

組み込み Linux におけるソフトウェア更新方式(1)

- 全体構成 -

小谷 亮 清原 良三 攝津 敦 橘高 大造

三菱電機株式会社 情報技術総合研究所

1. はじめに

近年、開発・原価のコスト上のメリットから組み込み機器へLinuxを適用するケースが増えている。その一方、組み込み機器のソフトウェアは大規模化・複雑化が進んでいるため、不具合のない状態での出荷が困難となっており、出荷後の不具合修正のためのソフトウェア更新技術が検討されている[1]。組み込みLinux上でのソフトウェア更新には特有の課題が存在する。本論文では組み込みLinux上でのソフトウェア更新の実現方法に関し、Linuxにおける課題およびその解決方法について述べる。

2. Linux におけるソフトウェア更新の前提

2.1. フラッシュ ROM 書き換え方式

本論文では、ストレージとしてフラッシュROMを搭載した組み込み端末を対象とする。通常のLinuxにおいては、プロセスの実行はRAM上で行われる。したがって、ソフトウェア更新の方法としては、RAM上でのプロセス実行中にフラッシュROMを書き換える方法が考えられる。しかし、LinuxシステムによってはROM上でプロセスの実行をおこなう機能を持つものもあり、特にRAM容量の節約等のためROM実行が必要なケースがある。この場合、ROM上でのプロセス実行中にROM内容を書き換えることはできない。したがって、汎用的な枠組みとして、ここではフラッシュROM書き換え専用の小規模なOSを別に用意し、書き換え時にはこのOSで再起動した上でLinuxシステムを書き換える方法を前提に考える。

2.2. ソフトウェア更新の手順

ソフトウェア更新は端末上で次の手順に従い行う。

- (1) ソフトウェア更新用データを通信機能によりサーバからダウンロードする。
- (2) 通常の Linux の起動状態から書き換え専用 OS へ再起動し、ROM 上の Linux システムを書き換え可能な状態にする。
- (3) 書き換え専用 OS 上でソフトウェア更新用データをROMに対して適用し、書き換えを行う。
- (4) 新版の Linux システムによって再起動し、後処理を行う。

上記のような方法を実装する場合には、以下のような課題が存在する。

A Software Update Method for Embedded Linux Systems
Akira Kotani, Ryozyo Kiyohara, Atsushi Settsu,
Taizo Kittaka
Mitsubishi Electric Corporation
Information Technology R&D Center

- (1) ファイルシステムの書き換えにおける書き換え時間とROM消費量の削減
 - (2) ファイルシステム上のリードライト対象データの扱い
 - (3) 書き換え中断時の対応
- 以下ではこれらの課題への対策について述べる。

3. ファイルシステムへの対応

3.1. 書き換え時間とROM消費量の削減

組み込みLinuxにおいては、CramFS[2]とJFFS2[3]という2種類のファイルシステムが広く用いられる。CramFSとJFFS2では、以下のような特徴の相違がある。

- ・ JFFS2 はリードライトが可能であるのに対し、CramFS は読み込み専用である。
- ・ 同一のデータ量を格納する場合、CramFS の方がJFFS2 に比較してROM消費量が小さい。

ソフトウェア更新の方法としては、JFFS2上にすべてのファイルを配置し、書き換えOSからJFFS2をマウントして、ファイルシステムの機能を用いてファイルを書き換える方法が考えられる。しかしこの方法では、ROM消費量および更新時間が増大するという問題点が存在する。

そこで、ソフトウェア更新の対象はCramFS領域とし、ソフトウェア更新の際には、CramFS領域のROMイメージに対する新版と旧版の差分データを適用することにより最新版に更新する。このような方式を用いることにより、JFFS2の機能を用いた場合に比較して更新時間の短縮・ROM容量削減の効果が得られる。また、差分データを利用することによりソフトウェア更新用データの容量を削減することができる。

なお、CramFSにはデータ圧縮機能があり、これに対応した差分抽出方式を用いる必要がある[5]。また、ROM上の書き換え対象エリアに対してあらかじめ余裕領域を挿入することにより、一定サイズ以内の書き換え単位に分割する[6]。このことによりROM書き換え量削減が可能となる。

3.2. リードライト対象データの扱い

システムファイル(/etc等に置かれるLinuxのファイルおよびアプリケーション依存のファイルがある)の中には、端末の使用中にリードライトされるものと読み込み専用のものがある。リードライト対象データに不具合があった場合の対応方法としては、端末ローカルな修正内容を保ちながら更新内容をマージする方法も考えられるが、マージ後のファイルの正当性を保証することは困難である。

そこで、リードライト対象のシステムファイルに更新が

生じた場合、端末ローカルな修正の保持よりも不具合修正を優先し、以下の対応とする。あらかじめシステムファイルを使用中にリードライトされるものと読み込み専用のものを分類し、前者をJFFS2に、後者をCramFSに配置する。また、JFFS2に配置するシステムファイルの初期ファイルをCramFSに配置する。更新が生じた場合には初期ファイルを更新するとともに、新版Linuxシステム再起動後の後処理として、初期ファイルの内容をJFFS2にコピーする。この方法により、ローカルな修正は消去されるが、リードライト対象データの不具合修正が可能となる。

端末のボリューム構成は表1のようにになる。なお、ユーザファイル(カーナビにおける登録地点情報、携帯電話における電話帳データ・カメラ画像等)はJFFS2に配置し、ソフトウェア更新対象外とする。

表1: ボリューム構成

ボリューム	内容	S/W更新対象
ブートローダ		-
Linux カーネル		
CramFS 領域	実行ファイル・共有ライブラリ・システムファイル(読み込み専用・R/W 対象の初期値)	
JFFS2 領域	システムファイル(R/W 対象)・ユーザデータ	初期値 上書き
書き換え OS	ソフトウェア更新用	-

4. 書き換え中断時の対応

書き換え専用OSによるフラッシュROMの書き換え中に、電池消耗・停電等の理由で中断が発生した場合、書き換え対象エリアの内容が破壊され、起動不可能となる場合がある。そこで、このような状態から復旧するための機能(フェイルセーフ機能)が必要となる。フェイルセーフを実現するための1つの方法としては、ROM領域を二重化し、一方を新版に更新してから切り替える方式が考えられる。しかし、この方法ではROM容量が通常の数倍必要となる。そこで、ここでは以下の方法によりフェイルセーフ機能を実現する。Linuxシステム上でサーバ～ダウンロードする際に、ソフトウェア更新用データは書き換え専用OSからも参照可能なROM領域に格納する。ソフトウェア更新時には、書き換え単位ごとに以下の処理を行う。

- 書き換え単位が書き換え前・書き換え中・書き換え完了のいずれの状態であるかを書き換え単位状態テーブルに記録する。
- 書き換え単位をバックアップ ROM 領域にコピーする。
- 書き換え領域ごとに旧版に戻すための逆差分データを作成する。
- RAM ワーク領域上で書き換え領域の新版への書き換えを行い、領域バックアップ・逆差分生成後にROMを書き換える。

書き換え手順を図1に示す。本手順に従えば、書き

換え中の任意の時点において、それまでに書き換えられたROM領域の逆差分データと、現在書き換え中の書き換え領域のバックアップコピーがROM上に存在する。したがって、書き換え中断時には、以下のいずれかの対応が可能となる。

- 中断が発生した書き換え単位から新版への書き換え処理の続行を行う。
- 逆差分データを用いて書き換え済みの領域を再度書き換えることにより、旧版への復旧を行う。

なお、基本的にユーザは新版への書き換えを望むため、新版への書き換えを続行し、最後まで書き換えを実施したがチェックサムが一致しなかった場合のみ旧版への復旧を行う。以上の方式によりフェイルセーフ機能を実現することができる。

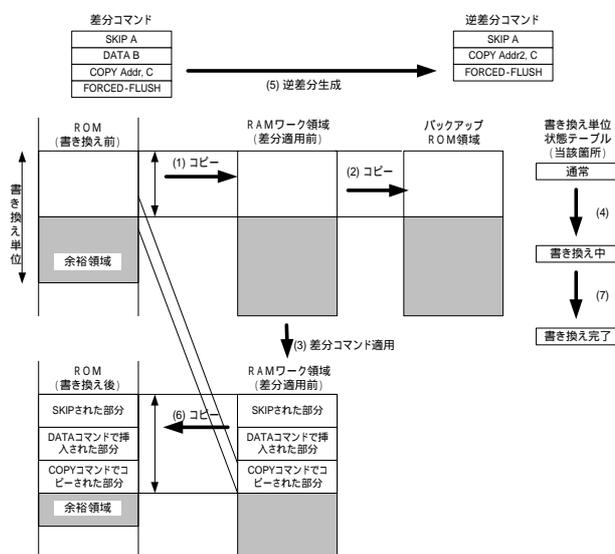


図1: フェイルセーフな書き換え手順

5. おわりに

組み込みLinux上でソフトウェア更新の実現における課題・およびその解決のための方式概要について述べた。今後は細部を設計し、試作により本方式の有効性を確認する。

参考文献

- 清原他, 「携帯電話のSW更新に関する検討」, 情報処理学会MBL 22-13, pp93-100, 2002.
- 栗原他, 「携帯電話SWのバージョン間差分データに関する検討」, 情報処理学会 DICOMO2003 4A-074
- <http://sourceforge.net/projects/cramfs/>
- <http://sources.redhat.com/jffs2/>
- 河相他, 「組み込みLinuxにおけるソフトウェア更新方式(2)」, 情報処理学会 第66回全国大会論文集
- 深澤他, 「組み込みLinuxにおけるソフトウェア更新方式(3)」, 情報処理学会 第66回全国大会論文集