組み込みシステム向けのダイナミックリンク手法に関する研究

野原 史朗†, 横山 孝典†, 志田 晃一郎†, 兪 明連†
武蔵工業大学大学院 工学研究科†

1 はじめに

近年，組み込みシステムに搭載されるソフトウェアが大規模化してきており，1つのシステム上で複数のアプリケーションを動作させるのが普通である．そのため，プログラムの共通部分をライブラリ化し，複数のアプリケーションで共有するようになった．このため，プログラムの不具合が発覚した場合や，アップデートを行いたい場合などには，当該箇所のみを動的に差し替えて利用できるようにしたいという要求がある．

ところが，一般に組み込みシステム向けのプログラムでは静的リンクが用いられている．リンク処理は，ソースファイルから実行可能イメージを生成する際に，アドレス解決などを行う処理である．静的リンクでは，単独で動作する実行可能ファイルを作成するために，必要となるすべてのオブジェクトを1つのファイルに結合する[1]．この際，組み込みシステムでは，すべての関数のアドレスを静的に決める．このため，リンク時に指定したアドレス以外で動作させることはできず，ライブラリやアプリケーション単位でのプログラム交換も行えないといった欠点がある．

UNIXなどの一般の計算機システム向けのOSでは，動的にライブラリへのリンクを行うダイナミックリンクが用いられている．しかしこの方法を資源の限られた組み込みシステムにそのまま適用することは容易ではない．

本研究の目的は，コスト要求が厳しく，資源に制約のある組み込みシステムに適したプログラム更新を実現することである．具体的には，リンク処理によるオーバーヘッドを抑えるとともに，ライブラリ呼び出しにかかる処理時間の変動を抑えたダイナミックリンクを提案する．

2 課題

2.1 従来のダイナミックリンク方式

図1に一般の計算機システムで用いられているダイナミックリンクの概要を示す．静的リンクでは，ライブラリも含めてすべてのプログラムをひとつに結合していた．これに対し，ダイナミックリンクでは，作成される実行可能イメージはライブラリなどを含まない[2]．図1の例では，1つのアプリケーションが複数のライブラリを呼び出している．アプリケーションから実行可能イメージを生成する際には，ライブラリ本体ではなく，呼び出したい関数を示す情報が組み込まれる．

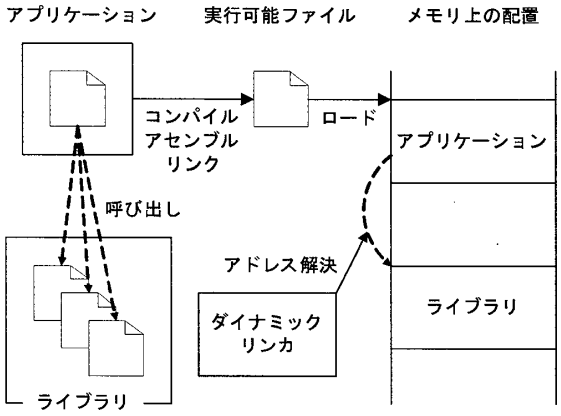

図1: ダイナミックリンク

生成した実行可能イメージをメモリ上にロードすると，ダイナミックリンカが呼び出される．ダイナミックリンカは，アプリケーションが呼び出す関数をライブラリから検索し，呼び出せるようにする．このように，ダイナミックリンクはロード時にアドレス解決をするため，ライブラリやアプリケーション単位でのプログラム差し替えが可能というメリットがある．

また，リンク処理をロード時ではなく，呼び出し時に行うことで，プログラムの起動を高速化する手法がある．この手法は遅延リンクと呼ばれ，現在Unix系OSやWindowsなど多くのOSで広く使われている[3]．

2.2 組み込みシステム向けダイナミックリンクの課題

ダイナミックリンクは実行時にリンク処理を行うため，実行要求が起きてから実際に実行が開始されるまでのオーバーヘッドが大きい．このため，CPU性能の限られた組み込みシステムには向かず，よりオーバーヘッドの小さいダイナミックリンク方式を実現する必要がある．

Dynamic linking for embedded systems
†Shiro NOHARA, Takanori YOKOYAMA, Koichiro SHIDA and Myungryun YOO
†Research Division in Engineering, Graduate School of Musashi Institute Of Technology

また，遅延リンクでは初回呼び出し時はリンク処理が必要となり，2回目以降の呼び出しとは処理量が大きく違う．組み込みシステムの多くはタスクの実行時間を予測してスケジューリングを行う必要のあるリアルタイムシステムであり，遅延リンクを適用するには問題がある．実行時間のばらつきが小さいダイナミックリンク方式の実現が要求される．

3 組み込み向けダイナミックリンク方式

3.1 ライブラリ関数呼び出し

図2に，提案手法におけるライブラリ関数の呼び出しの流れを示す．リンクテーブルには，外部から呼び出し可能なライブラリ関数ごとに1つずつリンクエントリが存在する．あらかじめ，リンクテーブル内の何番目のエントリにどの関数のアドレスを格納するかを決めておき，実行時にリンクテーブル内の特定のエントリを参照して関数呼び出しを実行する．

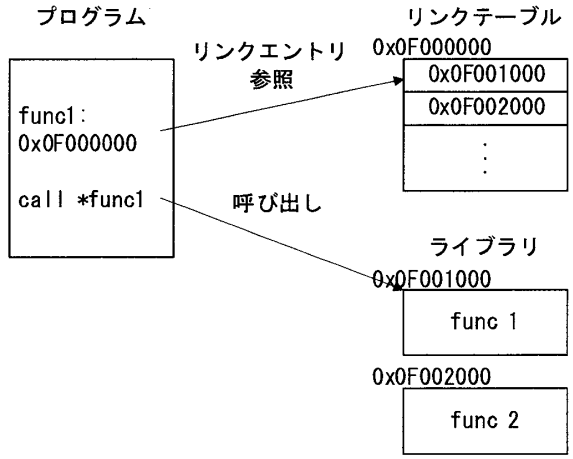

図2: ライブラリ関数の呼び出し

この図の例では，プログラムがライブラリ中の関数func1を呼び出そうとしており，ラベルfunc1はプログラム中で0x0F000000と定義されている．0x0F000000はリンクテーブルの0番目のエントリのアドレスであり，このエントリにはfunc1の開始アドレスである0x0F001000が格納されている．プログラムはこのエントリを参照することでfunc1の呼び出しが可能となる．

このように本方式では，ライブラリ関数毎に固定的なリンクエントリを提供する．これにより，リンクテーブルに記憶するアドレスを書き換えることで，ライブラリ関数のアドレス変更に対応できる．また，呼び出しは常にジャンプ1回で済むため，効率のよいライブラリ呼び出しが可能である．

3.2 リンクエントリの更新

ライブラリのアップデートなどによりアドレスが変化した場合，それに合わせてリンクエントリも更新する必要がある．

本方式ではホストコンピュータからライブラリを取得する．ターゲット上のダウンロードルーチンは，ホスト上のライブラリ配布ルーチンからライブラリをダウンロードし，メモリ上にロードする．その後，更新したライブラリ関数のリンクエントリを書き換える．これにより，アドレス変化後もリンクエントリを参照することでライブラリ関数が呼び出せる．

4 実装と評価

提案したダイナミックリンク方式を，CPUにSH3を用いた評価ボードを使用して，実装を行った．言語はCを用い，コンパイラはGCCを使用した．

実装したダイナミックリンク方式の関数呼び出し時のオーバーヘッドを評価した．従来の手法はライブラリ関数の呼び出しに2回のジャンプを必要とするが，今回の環境では呼び出しに4命令を要した．それに対し本手法は，3命令で呼び出を完了することができた．したがって従来よりも高速な呼び出しが可能になったと言える．

また，本手法では初回呼び出し時にリンク処理を必要としないので，遅延リンクのような実行時間のばらつきも発生しない．

5 おわりに

本論文では，オーバーヘッドの削減と実行時間の変動を抑えることを目的としたダイナミックリンク手法を提案した．実装・評価の結果，リンク処理によるオーバーヘッドを抑え，実行時間の変動がないダイナミックリンクが実現できたことを確認した．

参考文献

[1] 清水謙多郎, オペレーティングシステム, 岩波書店, 1992年.

[2] Kempf.J, Kessler.P.B, Cross-address space dynamic linking , Object Orientation in Operating Systems, 1992., Proceedings of the Second International Workshop on 24-25 Sept. 1992 Page(s):250 - 256.

[3] John R. Levine 著, 榊原一矢 監訳, ポジティブエッジ訳：Linkers & Loaders, オーム社, 2001年.