組み込み機器用 GNU/Linux 開発・テストシステムの開発

Development and test system for embedded oriented GNU/Linux system

小島 一元 [1)]　　杉岡 利信 [2)]
Kazumoto KOJIMA　Toshinobu SUGIOKA

1)（〒244-0002 横浜市戸塚区矢部町 946-31　E-mail: kkojima@rr.iij4u.or.jp ）
2)（株）アイ・ティー・オー（〒573-0163 枚方市長尾元町 7-4-8 E-mail: sugioka@itonet.co.jp）

ABSTRACT. DODES is the network system for cross development in GNU/Linux system. Because of its features, source code availability and the world wide cooperation, GNU/Linux is getting used in many environments, scale up to the mainframe and down to embedded system. For the development of embedded targets, the performance of target system (which is relatively slow) could be the bottle-neck of development and major problem. The cross compiling environment on high performance server is used to generate the binary and the binary runs transparently on target connected with network, which is quite convenient and resolve the bottle-neck. Clustering targets and multi-architecture handling makes DODES to be very flexible.

１．背景

これまで GNU/Linux システムは PC のプラットフォームで稼働してきており、この分野では技術的側面、社会的側面の双方において高い実績と定着した評価がある。そして、新たな技術的進展として、組み込み機器の分野においても GNU/Linux システムを用いることが可能となってきており、家電製品などの分野に急速に応用が広がりつつある。

この背景には、組み込み機器に利用されるハードウェアが GNU/Linux システムにとって必要十分の機能を有するようになってきたこと、また、組み込み機器に要求されるソフトウェア技術が複雑で高機能のものとなり GNU/Linux システムのような OS を必要とするようになってきたこと、さらには GNU/Linux システム自身のスケーラビリティの向上など多くの要素がある。

一般に組み込み機器においては、開発効率をあげるためにクロス開発を行うことが多いが、 GNU/Linux システム全体のような大規模ソフトウェアをビルドする場合、その途中でターゲット用の実行ファイルを実際に動作させる必要のあることが多く、従来の手法ではクロス開発環境との整合性が悪いため、開発効率を低下させる原因となっている。

２．目的

DODES は複数のターゲットマシンを、ホストマシンからネットワークを用いて利用するクロス開発環境のネットワークシステムを構築し、組み込み機器向けの GNU/Linux システムにおいて効率よくクロス開発できるようなシステムの基盤技術を開発することを目的とし、GNU/Linux システムが動作する複数の組み込み機器用プロセッサターゲットをネットワークに接続した高速 PC をホストとするハイブリッド型の開発・テストシステムを構築することを目指している。このためすでに平成 12 年度未踏ソフトウェア創造事業において SuperH プロセッサターゲットと高速 PC ホストによって構成される開発・テストシステム DODES の開発を行った。

今回の開発はこの DODES を発展させ、プロセッサやエンディアンが異なるターゲットの混在した環境や既存のネットワーク上での利用を安全に行うためにホスト-ターゲット間の通信プロトコルを修正するなどして多様なシステムの構成で、運用や管理面も含めて容易な利用を可能にすることを目的としている。

組み込み機器用プロセッサの多くはそのプロセッサファミリ内にさらに細分化されたアーキテクチャを持ち、その間での実行バイナリの互換性は基本的に存在しない。また組み込み機器開発においてプロセッサファミリやファミリ内でのプロセッサ変更は珍しいことではないので複数のアーキテクチャへの対応は必要性が高いものであった。

３．DODES システム

（1）概要

DODES では、ホスト計算機からの遠隔実行要求を受け、ターゲット計算機上でプログラムが稼働する。この際、複数のプログラムの全体の実行時間を短縮させるために、複数あるターゲット計算機において、負荷分散を行う。この負荷分散はプロセスの実行を単位とする荒い粒度の負荷分散であり、プロセスのマイグレーションは行わず、負荷の分散はプロセスの投入時にその時点での各ターゲット計算機の負荷の情報に基づいて行われる。

複数の異なるアーキテクチャを持つターゲット計算機が存在する場合にはそれぞれのアーキテクチャごとに独立した負荷分散が行われる。

（2）実装

DODES ではホスト計算機上であるプログラムを実行する場合にそれがターゲット計算機のためのものであることを検知し、複数あるターゲット計算機のどれかの上での遠隔実行を行う。ターゲット計算機とホスト計算機は NFS によってファイルシステムを共有しておりターゲット計算機におけるファイルのパス名がホスト計算機での対応するファイルのパス名が同一または簡単なマッピングで対応するように構成される。

DODES におけるターゲット負荷分散は、一台のホスト計算機上で動作するサーバプログラム(DODES サーバ)を中心として機能する。 各ターゲット(DODES ノード)は指定した DODES サーバに対して一定間隔で自身の状態を報告し続ける。状態データの送出には単一の UDP パケットを用いている。 DODES サーバでは、この各 DODES ノードからの報告を保持し、遠隔実行を行うホスト計算機(DODES クライアント)からの問い合わせに対して、適切な遠隔実行サーバの所在を回答する。DODES クライアントからの DODES サーバへの問い合わせは FTP や SMTP などの一般的 TCP サーバと同様のプロトコルを用いている。

DODES の透過的遠隔実行機能は、ダイナミックリンカの共有オブジェクトプリローディング機能を用いて、GNU libc の execve() システムコールをラッパー関数で置き換えることによって実現されている。

アクセス制御機能として TCP ラッパーライブラリを使用し、状態やノードの取得の問い合わせや遠隔実行の制限が行なえるようにしている。

（3）稼動状況の取得

DODES ではターゲットアーキテクチャバイナリの実行がホストシステム上透過的に行われる。つまり実際にどの DODES ノード上で遠隔実行されたかは問わない。そこで通常の実行中にはわからない各ターゲット計算機の負荷状況や遠隔実行記録などをホストマシン上で取得可能とし可視化して示す基本機能を実装した。また複数アーキテクチャへの対応に伴いノードのアーキテクチャ情報もこの機能で取得、表示できるようにした。

負荷分散サーバでの遠隔実行記録はノード要求時に得られた情報をサーバのログ情報として扱うことで行う。この情報は正確には遠隔実行の予備的要求の記録で本来の遠隔実行記録ではないが、遠隔実行記録が求められる状況では充分な近似だといえる。記録自体は通常のサーバのロギングをおこなう syslogd への記録の枠組を用いている。

（4）応用例

2002 年 1 月から産業技術総合研究所内において、1 台のホストと 8 個のノードで構成される DODES システム（dodeser）の試験運用を開始した。このシステムに含まれる DODES ノードは SH-3, SH-4 の 2 種の CPU で 2 通りのエンディアンのものを含んでおり、Linux/SH でサポートされているアーキテクチャを網羅している。このシステムは GNU Compiler Collection (GCC)における C コンパイラの SH 対応部分の開発およびテストをはじめ、Perl パッケージのブートストラップなど、従来はクロス開発環境では不可能であったか、または困難であった作業を著しく効率化した。

小規模な応用として PC とゲームコンソール上で利用できる GNU/Linux システム PS2Linux との組合せで双方が DODES ホストかつノードとなるようなシステムを、主に GCC-3.0 のリグレッションテストを目的として構成した。 標準的な PC を使った場合でも DODES システム利用時、C コンパイラのビルドに要する時間については約 5 倍、そのリグレッションテストに要する時間では約 2 倍の高速化を行なうことができた。

これらの応用例はいずれも組み込み機器用 Linux 開発ツールの整備に大きな効果を上げた。 特に SuperH アーキテクチャに対する GNU toolchain は GCC-3.0.3 において実質的に C 言語での GCC testsuite でリグレッションが全て解消され、また C++ 言語においてもリグレッションは残り 1 つとなるなど、格段に向上した。 さらに GNU foreign function interface library の SuperH ポート、GCC-3.0.3 の EmotionEngine ポートの整備なども短期間で可能になった。

（5）結論と課題

DODES のアイデア自体は非常にシンプルでその構成要素自体はよく知られたものであるが、組み込み GNU/Linux システムではターゲットとホストでほぼ同一の GNU/Linux という環境が用意できるという特性から、このようなシステム向けのクロス開発環境にとって実用性の高い手法と考えられる。複数アーキテクチャを含むシステムの構築と運用が可能になったことでその適用範囲を広げることができた。

DODES システムの最新バージョンのソースやドキュメントはその他の情報とともに
http://www.dodes.org/dodes/index.ja.html
からたどることができ CVS による最新のリソースの提供を行っている。

DODES における課題として、大規模なシステムの運用に際しての支援機能の充実をあげることができる。このような機能としてはメンテナンス目的で DODES ノードを明示的にシステムから切り離す機能などをあげることができる。

４．参加企業及び機関

なし

５．参考文献

［1］GNU/Linux on SuperH プロジェクト, "GNU/Linux on SuperH プロジェクト",
http://lc.linux.or.jp/lc2001/papers/linux-superh-paper.pdf
［2］http://www.m17n.org/superh/