

## 組込み計算機を拡張するためのリモートデバイスの実現

## Implementation of Remote Devices for Embedded Computer Extension

明神智之† 佐藤喬† 多田好克†  
Tomoyuki Myojin Takashi Satou Yoshikatsu Tada

## 1 はじめに

近年、組込み計算機の利用は情報家電の普及、ユビキタス社会の発展に伴って、業務用製品にとどまらず広く拡大してきている。組込み計算機には省スペース、省電力、高耐久性といった特徴がある。今後、組込み計算機の利用はますます大きくなるものと予想される。

組込み計算機は上記のような特徴を持つ反面、拡張性が乏しいという欠点を持つ。これは省スペース性と拡張性が物理的にトレードオフの関係にあるためである。拡張性の不足は、入出力デバイスを貧弱にして、組込み計算機上での開発環境を大きく阻害する要因となる。また、一旦システムを構築すると後から変更を加えることが困難で、特にメンテナンスが困難な場所に設置された組込み計算機では、システムの再構築にかかるコストが膨大になる。これらのことから、組込み計算機の省スペース性を生かしつつ、柔軟な拡張性が求められる。

## 2 概要

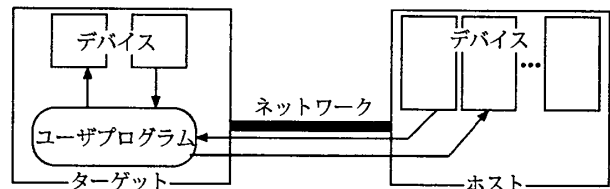
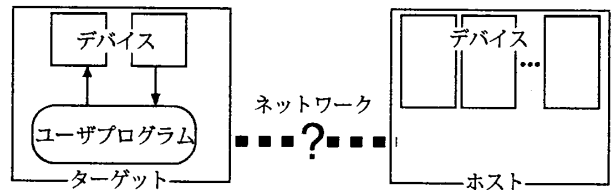
本研究では、リモートデバイスを用いて組込み計算機の開発環境を支援するシステムを構築する。ネットワーク上に接続された別の計算機(以下ホストと呼ぶ)から資源を借り、あたかも組込み計算機(以下ターゲットと呼ぶ)がその資源を持っているかのように見せる。この手法で利用できるデバイスのことをリモートデバイスと呼ぶ。リモートデバイスの利用に際しては、組込み計算機を物理的に変更する必要はない。リモートデバイスを利用することで、組込み計算機の省スペース性を保ったまま柔軟な拡張性を持たせられ、ネットワーク越しにデバイスの入出力が可能となる。その結果、ターゲットで動くユーザプログラムの開発者(以下ターゲット開発者と呼ぶ)は開発時に十分な入出力機能を得られ開発が容易になり、組込み計算機を用いたシステムの開発にかかるコストの低減が実現される。

## 3 設計

## 3.1 要求仕様

- ターゲットとホストの関係には、開発時の状態(状態- $\alpha$ )と運用時の状態(状態- $\beta$ )がある
- 状態- $\alpha$ のとき(図1)
  - ターゲットは必ずネットワークを通じてホストへ接続されている。
  - ターゲット開発者は開発の対象となるユーザプログラム中から、ホストが持つデバイス(リモートデバイス)を利用できる。
  - ターゲット開発者は本システムが用意するインタフェースを用いて、明示的にリモートデバイスを利用できる。
- 状態- $\beta$ のとき(図2)

- ターゲットは必ずしもネットワークを通じてホストへ接続されているとは限らない。
- いかなるリモートデバイスも利用しない。
- ターゲット開発者は状態- $\alpha$ で作成したユーザプログラムのリモートデバイスに関する記述を変更する必要はない。
- リモートデバイスに関して、ユーザプログラムは実行時にオーバーヘッドを伴わない。
- 開発を終了すると状態- $\beta$ へ遷移し、再度ユーザプログラムの変更の必要が生じたとき、容易に状態- $\alpha$ へ遷移できる。
- ターゲットの物理的なハードウェア変更は行わない。

図1: 開発時(状態- $\alpha$ )図2: 運用時(状態- $\beta$ )

## 3.2 設計

## 3.2.1 インタフェース

本システムは、ターゲット開発者がリモートデバイスを利用するためのインタフェースを、ライブラリの形で提供する。例えば、ターゲット開発者は `printf` の出力先をリモートデバイスへ宛てたい場合、`rprintf` を利用する。さらに、ローカルデバイスとリモートデバイスの双方へ同時に出力したい場合、`xprintf` を利用する。また、直接リモートデバイスをローカルデバイスファイルとしてオープンできる。

## 3.2.2 環境

本システムではターゲット、ホスト、及びその間のネットワークについて次のような環境を想定する。

## ■ターゲット

- UNIX OS が稼働
- 入出力デバイスが貧弱、もしくはそれらを持たない
- イーサネットポートを装備

## ■ホスト

- OS は問わない
- 十分な機能の入出力デバイスを持つ
- イーサネットポートを装備

† 電気通信大学 大学院情報システム学研究所

## ■ターゲット・ホスト間のネットワーク

- LANのような高帯域のネットワーク

### 3.2.3 ネットワーク

ホストとターゲットの間はネットワークで接続する。ネットワークを用いることで、ターゲットのハードウェアの物理的な変更を行う必要がなくなる。ユーザプログラムからインタフェースを通じたデータの送信要求があると、TCP/IPのsocketを用いてホストへデータを送信する。逆にユーザプログラムがデータの受信待ちになると、socketからのデータ転送を待つ。

ターゲットとホストの間の接続が切断されると、一定の時間内であれば転送すべきデータはバッファリングされ、接続が回復すると転送される。一定の時間以上ホストとの接続が切断されると、開発が終了したとみなして状態- $\beta$ に遷移する。状態- $\beta$ のとき、リモートデバイスに関する処理は行われないのでユーザプログラムの実行時にはオーバーヘッドは伴わない。

## 4 実装方式の検討

本システムのターゲット上での実装方式について、ライブラリによる実装、カーネルによる実装、デバイスファイルによる実装の三つの方式で検討した。ここでは簡単のために出力デバイスに限りて議論する。各実装の概略図を図3に示す。

ライブラリによる実装では、ターゲットのユーザ空間ですべて実装するため実装が容易で、ユーザプログラムへインタフェースも提供できる。しかし、ユーザ空間で処理を行うため、システムコール呼び出しが頻発しオーバーヘッドが大きくなる。

カーネルによる実装では、ターゲットのカーネル空間内で処理を行うため、オーバーヘッドは小さくなる。しかし、ユーザプログラムに対してのインタフェースを提供するための実装が困難で、ターゲット開発者はデバイスがリモートかローカルであるかを明示的に指示できない。

デバイスファイルによる実装では、カーネルによる実装と同様にターゲットのカーネル空間内で処理を行うため、オーバーヘッドは小さくなる。また、複数のデバイスファイルを作ることで、ユーザプログラムへインタフェースを提供できる。

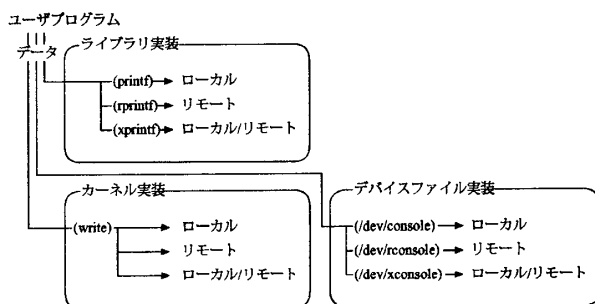


図3: 各実装の概略

## 5 実装

本システムでは、

- 明示的に出力先を指示できるようなインタフェースが存在
- 実装が容易
- オーバヘッドが小

という点において、要求仕様を満たすデバイスファイルによる実装を採用した。

今回は/dev/rconsoleのプロトタイプをLinux 2.4上で、Loadable Kernel Module機構を用いて実装した。実装したデバイスファイルは以下の機能を持つ。

- open** 指定されたホストへsocketによるネットワーク接続を確立する。
- close** 指定されたホストとのsocketによるネットワーク接続を閉じる。
- write** 指定されたホストへsocketによるメッセージの送信を行う。

また、ホスト側には以下の動作を行うモニタを実装した。

- 指定されたターゲットからのsocketによるネットワーク接続を待つ。
- 接続が確立されると、ターゲットからのメッセージを待つ。
- メッセージを受信すると、ホストの端末に受信したメッセージを出力する。
- 接続が閉じられると、再び接続を待つ。

このプロトタイプでは状態- $\alpha$ と状態- $\beta$ の遷移は考慮しない。

## 6 関連研究

Etherconsole[1]はLinux上でのコンソールへのアクセスをイーサネットインタフェースへリダイレクトする。対象はキーボード、ビデオ、マウスであるため、それ以外のデバイスには対応していない。そのためリモートにあるファイルシステムやディスクを利用することはできない。

佐藤らはネットワーク透過な周辺機器制御の枠組み[2]を提案し、Linuxカーネルにリモートデバイスを利用するための実装をしたが、本研究における状態- $\alpha$ と状態- $\beta$ を相互に遷移するような事態は考慮されていない。

## 7 まとめと今後の課題

本研究では、リモートデバイスを用いた組込み計算機の拡張を提案した。これにより、入出力機能が貧弱な組込み計算機においても、デバッグ情報をリモートデバイスに出力するなど、リモートにある豊富な資源を利用することが可能となった。今回はリモートデバイスをデバイスドライバによる方式で実装し、その有用性を確認した。

今後は、入力やファイル、ブロックデバイスの実装やネットワーク接続の自動認識の実現を行う予定である。

## 参考文献

- [1] Kistler, M., Hensbergen, E. V. and Rawson, F., Console over Ethernet, in *Proceedings of the FREENIX Track:2003 USENIX Annual Technical Conference* (2003).
- [2] 佐藤友隆, 中山健, 小林良岳, 前川守, カーネルレベルで実現したネットワーク透過な周辺機器制御の枠組み, 情報処理学会システムソフトウェアとオペレーティングシステム研究会 (2001).