システムソフトウェアと
オペレーティング・システム 84－8
（２０００．５．２５）

電車模型制御用ソフトウエアシステムの設計

森若和雄 * 久住憲嗣 * 中西恒夫 * 片山徹郎 * 最所圭三 † 福田晃 *

あらまし

組み込みシステムの多機能化、複雑化により、開発工期の短縮が重要な課題となっている。そのためソフトウエアの再利用のために、組み込みシステムにおいてもオペレーティングシステムが利用されるようになってきている。

組み込みシステムのOSは、多様なハードウエアに対応する必要があり、なおかつオーバーヘッドや資源の浪費を避ける必要がある。そのためOSをハードウエアにあわせてカスタマイズすることで専用化を行う。

本稿ではこのカスタマイズによる専用化に注目した組み込みシステム向けOSを試作する。必要な部品を入手しやすいこと、扱い易いことなどを考慮して、電車模型の制御を行うシステムを題材とする。

Design of a Train Model Control System with an Embedded Operating System Prototype

Kazuo Moriwaka * Kenji Hisazumi * Tsuneo Nakanishi *
Tetsuro Katayama * Keizo Saisho † Akira Fukuda *

Abstract

In recent multifunctional and complicated embedded systems extension of time to market becomes a serious problem.

Use of embedded operating systems is one of promising solutions for this problem. Embedded operating systems are required to adapt various kinds of hardware and to be small and light. We construct an easy-customizable embedded operating system. This paper discusses design of a train model control system with such an embedded operating system.

*奈良先端科学技術大学院大学 情報科学研究科
†香川大学工学部
*Graduate School of Information Science, Nara Institute Science and Technology
†Faculty of Engineering, Kagawa University

1 はじめに

近年、半導体技術の進展やハードウエア開発工程のソフト化などにより、マイコンを利用した組み込みシステムを従来より安価に製造することができるようになった。その結果、自動車や玩具、掃除機や扇風機といった家電製品でも組み込みシステムが利用されるようになった。製品サイクルの短期化により、組み込みシステムを短かい納期で設計・開発する必要がでてきた。組み込みシステムの開発においても開発工数の削減が重要になり、開発効率の向上が重要な課題となっている。

そこで従来はOSを利用せずに開発されることの多かったマイコンを利用した小規模な組み込みシステムにおいても、ソフトウエアの再利用性などを高め、ハードウエアの変化への対応を集中させることで開発工数を削減するために、OSが利用されるようになってきている。

ところで、組み込みシステムは製造コストを下げるために利用できるメモリやCPU速度など、資源の制約が厳しい。また機器の制御やセンサーなどでリアルタイム性が必要になることも多い。組み込みシステムで利用できるハードウエアには多くの選択肢があるので、様々なハードウエアに対応することが望まれる。そこで、組み込みシステム向けのOSは以下のような制約を満たす必要がある。

- 消費するメモリ量などの資源が少ないこと
- リアルタイム性を持つこと
- 様々なハードウエアに対応すること

様々なハードウエアに対応する為にHAL(Hardware Abstraction Layer)をOSに導入すると、メモリなどの資源が浪費されたり、オーバーヘッドがかかる。

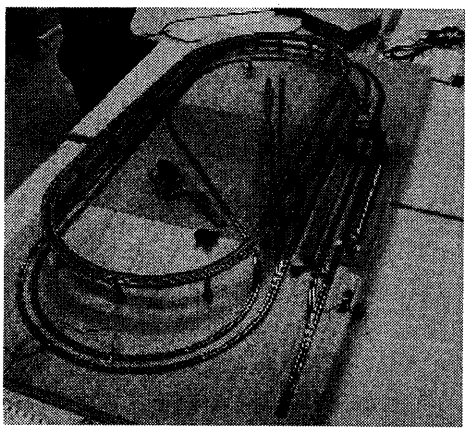

図 1: 実験用レイアウト

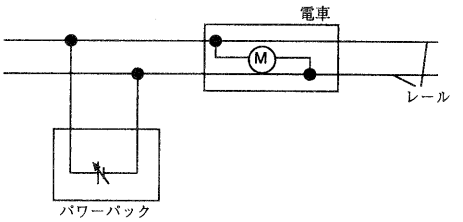

図 2: 電車模型の概要

一旦作成された組み込みシステムは、ハードウエアの一部を互換性のないものと交換するということはあまりない。そこでOSのソースをコンパイルする前に、ハードウエアに依存する部分をプリプロセッサなどで切り貼りしてカスタマイズすることで、そのハードウエアに専用化されたOSを生成できる。ソースの切り貼りによるカスタマイズによって、オーバーヘッドを抑えつつ多くのハードウエアに対応するOSを作ることができる。

そこで、本研究ではこのカスタマイズによる専用化に注目した組み込みシステム向けOSを試作する。題材としては、工作の必要が少ないことと、必要な部品を入手しやすいこと、扱い易いことを考慮し、図1のような電車模型の制御を行うシステムとする。

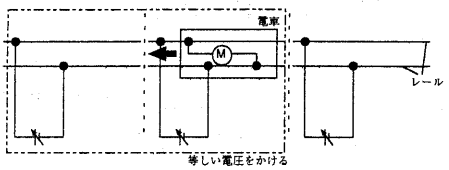

図 3: レールの区間分割

2 電車模型制御システム

2.1 電車模型

電車模型制御システムの説明にはいる前に、本研究で使用した電車模型「Nゲージ」の構造について説明する。

Nゲージでは図2のように、パワーパックと呼ばれる電源装置から線路の2本の金属レールに電圧をかけ、レールを介して電車に給電し、電車に搭載されている直流モータを駆動する方式をとっている。パワーパックで電圧を制御することによって電車の速度と方向を制御する。この線路は必要に応じ絶縁することができ、図3のように、区間ごとに異なる電圧をかけることが可能になっている。また、ポイントにはレールの方向を変える電磁石が入っており、その電磁石に電流を流すことにより電車の進行方向を切り替えることができる。

2.2 システムの概略

このシステムでは、パワーパックを人が操作する部分を置き替えて、あらかじめ定められたダイヤに従って、複数の電車を制御する。

複数の電車を制御する方法として、電車に制御回路を積み制御する方法と、線路を互いに絶縁されたいくつかの区間に分割し、それぞれの区間に異なる電圧をかけ制御する方法が考えられる。今回は電車模型になるべく手を加えないで作成するために、後者の方法で制御することにする。

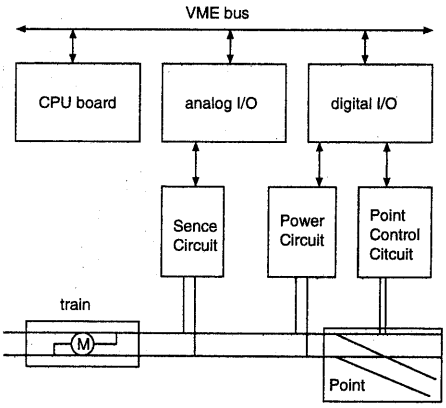

図 4: 制御コンピュータの概要

電車を走行させる際には、図3のように制御の対象とする電車がいる区間と、その電車の進行方向の次の区間に等しい電圧をかけておく。これは、次の区間に電車が進入する際に電車の動きがぎこちなくなるのを防ぐためである。

区間ごとに異なる電圧をかけることにより複数の電車を制御するので、それぞれの区間に進入できる電車は1本のみである。2本目以上が進入しようとしたときは電車を停止する。

ひとつの電車に2つの区間を割り当てるので、同時に制御できる電車の台数は最大$\lfloor \frac{区間数}{2} \rfloor$(但し区間数 ≥ 2) となる。

2.3 制御コンピュータ

制御用のソフトウエアを動作させるプラットホームは、図4のようにCPUボードとアナログI/Oボード(本設計ではA/Dコンバータを使用)、パラレルI/OボードをVMEバスで接続したものである。各I/Oボードは各種制御回路に接続されており、CPUボードからI/Oボードを操作することにより電車模型を制御することが可能となる。CPU

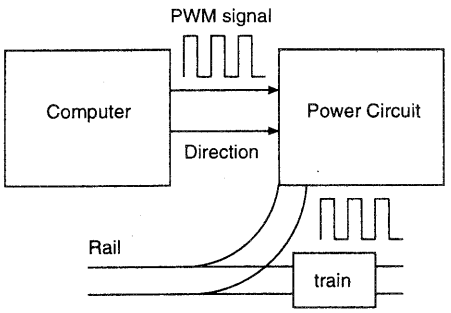

図 5: 速度制御機構の模式図

ボードとしては、CPU に 68000 を使ったものと SH-3 を使ったものの 2 種類を用意する。

2.4 電車の速度制御 (PWM)

電車の速度及び方向制御はレールに与える電圧によって行われる。各区間のレールに電圧を加える方法としては、区間ごとに電源を持つ方式と、ハードウエアで指定の区間へ給電先を切り替える電源選択方式が考えられるが、後者は電車の台数の増減に応じてハードウエアを増減させる必要があること、ハードウエアの構成が複雑になることなどの理由から、前者を採用する。

各区間の電源を、制御コンピュータが出す PWM 信号が High の時は ON、Low の時は OFF となるようにスイッチングすることにより速度制御を行う。この PWM 信号は実験的に $10kHz$ 程度が適正である。PWM 信号のデューティ比が小さければ電車は低速で走行し、デューティ比が大きければ電車は高速で走行することになる。また、電車の進行方向も制御コンピュータにより与え、信号が High の時はレールに正電圧、Low の時は負電圧をかける (図 5)。

2.5 電車の位置検出

電車の位置を検出する方法としては、各区間ごとに流れている電流を計る方法がある。電流が流れていれば電車が存在し、流れていなければ電車は存在しない。この方法は特別なハードが要らないという利点があるが、区間単位でしか電車の位置を把握できず、細い制御には向かない。そこで、細い制御を必要とする区間には、タッチセンサを設置し、2 つの方法を併用する。

2.6 ポイント切り替え

ポイントは電磁石により構成されており、電磁石に一定の電流を流すことにより電車の進行方向を制御できる。ポイントに決められた時間だけ電流を流すために、ワンショット回路を組み、その信号により電源をスイッチしポイントを変えることにする。制御コンピュータの信号によりスイッチする方が簡単な回路で良いのだが、制御コンピュータが誤動作した場合にポイントを破壊しないためにこの回路を採用する。

3 オペレーティングシステム

このシステムの OS は、以下の機能をサポートする。

- タスク管理
- タスクのスケジューリング
- メモリ管理
- 割り込み処理

このうちスケジューリングとメモリ管理に関しては、複数の実装を行う予定である。それぞれの実装は、特別な抽象化はしないで OS に記述するが、コンパイル前にカスタ

マイズすることでそれぞれの方式を選択するように作る。また、割り込み処理などのCPUボードの仕様に依存する部分もカスタマイズできるように作る。カスタマイズは`#ifdef`などのプリコンパイル指令を用いてソースコードを切り貼りする方式で行う。

3.1 スケジューリング

スケジューリング方式は、ラウンドロビンとEDF(Early Deadline First)の2方式を実装する。

3.2 メモリ管理

メモリ管理は固定長のページ単位での確保と解放を行う方式と、複数のブロックサイズを用意してブロック単位で確保と解放を行う方式の2つの方式を実装する。後者の方法は内部フラグメンテーションによるメモリ空間の無駄を減らすことを目的としている。

3.3 割り込み管理

このシステムで利用する割り込みは、タイマ割り込みとVMEバスからの割り込み入力(I/Oボードからの割り込み)のみである。

ハードウエアの仕様記述(I/OボードのIRQ番号、タイマ割り込みの割り込み番号など)とテンプレートから、割り込みハンドラの登録、削除、割り込みからの復帰 などを行うシステムコールのコードを生成するシステムを作成する。

4 本システムのタスク構成

本システムのタスク構成の概要とそれぞれのタスクについての検討事項を列挙する。

4.1 ユーザ入力

ユーザ入力に一つタスクをわりあてる。

入力を検知するための仕組みとしては、大きく分けて以下の2つの方法が考えられる。

- 周期的にプログラムがユーザ入力を見る。(ポーリング)
- ユーザ入力によって割り込みを発生させてユーザ入力を見る。(割り込み)

ユーザ入力は遅延が発生しても比較的問題が少ないと考えられるので入力の検出にはポーリングを使うことにする。

4.2 電車の位置検出

電車の位置検出についてのタスク構成について、以下の2つの方式が考えられる。

- 区間ごとにセンシングをするタスクを割り当てる方式(区間方式)
- 電車ごとにセンシングをするタスクを割り当てる方式(電車方式)

区間方式の利点

- 動作中のタスクの増減はない
- 動作中に新しく電車を載せた場合の検出が簡単にできる

区間方式の欠点

- タスク数は区間の数必要
- どの電車がどこにいるかは、別に管理する必要がある

電車方式の利点

- タスク数が電車の数だけになるので、区間方式に比べて少ない

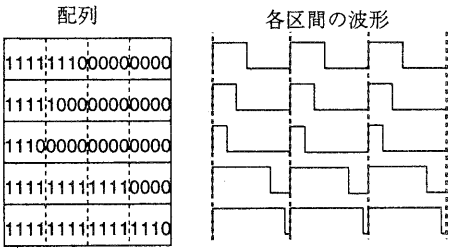

図 6: 波形と bit の対応

- 個々の電車がタスクに対応するので、区別するための仕組は不要

電車方式の欠点

- 一つのタスクで複数の区間を監視するので動作が複雑になる
- 電車の増減によって、動作中にタスクの増減がある
- 新しく電車を載せたことの検出の為の仕組みが別に必要

以上から、動作中に電車の数が変更されうる場合は、区間方式の方が単純に実行でき、逆に電車の数を変更しない場合は電車方式の方が単純であると思われる。

電車の数が変更されないことを仮定してシステムを構成すると、脱線などの事故の復旧など、電車の数が一時的に変更される場合の処理が面倒になるので、本システムでは区間方式を採用する。

4.3 電圧制御

本システムでは、制御用の PWM 信号出力処理を CPU を使って行う。この信号を安定して出力するために、タイマ割り込みを利用して信号の値を変化させる。

各区間に流す波形を表す各要素が 16bit 幅の配列を用意する。配列の各要素は区間に対応する。図 6 のように、要素の bit それぞれの 0/1 は電圧の Hi/Low に対応する。タイマ割り込みごとに配列のそれぞれの値をローテートさせ、最下位 bit の値によってパラレル出力を変更させる。

ユーザ入力、電車の位置検出と運行計画から、各電車に区間を割り当て、必要に応じて上で述べた配列を変更する為にタスクを一つ用意する。

5 おわりに

本稿では、現在我々が設計・製作している電車模型制御用システムの設計について、ハードウエアとソフトウエアそれぞれの概要と設計時の検討事項を報告した。

本システムのアプリケーション面での検討課題として以下のものが挙げられる。

- 脱線や停止などの事故への対応
- 新規に電車が配置された場合の扱い
- ダイヤグラムの自動的な編成
- 詳細な位置情報に基いた制御

また、OS についても、以下の課題がある。

- カスタマイズの半自動的な管理
- デバイスドライバのサポート

最終的には、OS の抽象化と性能の両立をカスタマイズによって実現することを目指している。

参考文献

[1] HITACHI SuperH Information
http://www.super-h.com/

[2] 東芝セミコンダクタ
http://www.semicon.toshiba.co.jp/

[3] アバールデータ
http://www.avaldata.co.jp/jpn/
index.html

[4] 株式会社トミー トミックス総合カタログ 1998-1999 (1998)

[5] 渡辺政彦, 羽生田栄一, 高田広章 情報処理 vol.41 No.2 インタラクティブエッセイ (2000)