2005－EVA－15（6）
2005／11／22

模型を使った組込みソフトウェア設計教育試行とその評価

中西 恒夫[1)]　野田 厚志[2)]　北須賀 輝明[1)]　福田 晃[1)]

[1)] 九州大学大学院システム情報科学研究院
[2)] 九州大学大学院システム情報科学府

概要

さまざまなハードウェアを介して外部環境に接する組込みシステムのソフトウェアは，これらハードウェア，外部環境に因る問題によって，しばしば予期しない変更を強いられる．ゆえに，組込みソフトウェア開発の教育プログラムは，学生にハードウェアや外部環境を意識させるものであることが望ましい．著者らは，計算機工学に関する基礎知識を有する学部 4 年生を対象に，鉄道模型を使った組込みソフトウェア開発に関する教育プログラムを実施した．本報告では当該教育プログラムの概要と評価を述べる．当該教育プログラムは，学生にハードウェアと外部環境を意識させるもので，また適度の複雑さを有し，学生の能力に応じて複雑さを変えやすいものであった．

Trial and Evaluation of an Education Program on Embedded Software Development with Using a Model

Tsuneo Nakanishi[1)]　Atsushi Noda[2)]　Teruaki Kitasuka[1)]　Akira Fukuda[1)]

[1)] Faculty of Information Science and Electrical Engineering, Kyushu Univ.
[2)] Graduate School of Information Science and Electrical Engineering, Kyushu Univ.

Abstract

Embedded software, which interacts with external environment via various hardware devices, is often required unexpected modification to resolve problems caused by hardware and external environment. Therefore, the education program for embedded software development should force the students to mind hardware and external environment matters. The authors planned and performed an education program on embedded software development with using model trains for undergraduate students equipped with basic knowledge on computer engineering. This paper reports what the authors did in the education program. The author found the education program had appropriate complexity for undergraduate students, forced the students to mind hardware and external environment matters, and could change complexity depending on ability of the students.

1　はじめに

最近数年の間，日本のコンピュータ産業界は自動車，家電製品，製造機械等の組込みシステム分野に自身の競争力を見出し，多くの資源を割り当てている．産業界のこのような動きに応じて，組込みソフトウェア教育の重要性が認識されるようになってきた．

組込みシステムとは対極にある PC システムは，標準化されたハードウェア上で動作する寡占的オペレーティングシステムの存在を前提とできるため，ハードウェアによってソフトウェアの開発が左右されることはあまりない．一方，組込みシステムは，ハードウェアとソフトウェアが一体となって特定のサービスをユーザに提供するものであり，ハードウェ

アとソフトウェアの開発は並行して進められる．ソフトウェアが何らコストを発生させることなく複製できるのに対して，ハードウェアは製品の製造台数にしたがってコストが発生し，特に大量生産がなされる製品においては製品 1 個当たりのハードウェアのコストは問題となる．製造や調達に生じるハードウェアコストの問題は，ハードウェア設計の自由度を制約し，ハードウェアの設計変更が余儀なくされるが，こうしたハードウェアの設計変更の問題はソフトウェアの設計に影響を与える．また，ハードウェアとソフトウェアが結合される開発の後段にあっては，ハードウェアのバグ，ハードウェア，ソフトウェアの仕様の認識の誤りが顕在化される．この段階でのハードウェアの変更は極めて難しいため，ソフトウェアがハードウェアにあわせなければならない．さらには，ハードウェアとのインターフェースのみならず，ハードウェアとその外部環境とのインターフェースで生じる問題についても，ソフトウェアによる対処が要求されることは少なくない．

組込みソフトウェア開発の難しさの理由のひとつとして，ハードウェアや外部環境によってソフトウェアの設計変更が強いられることが挙げられよう．ゆえに，組込みソフトウェアの教育プログラムは，単に組込みソフトウェア開発において伝統的に使用されるテクニックを与えるのみならず，設計，開発，実装の過程において，ソフトウェアのみならず，ハードウェアや外部環境を考慮させるべきである．

上述の組込みソフトウェア開発の特異性を鑑み，筆者らは，研究室に配属された学部生に，鉄道模型を用いた組込みソフトウェア開発プログラムを実施したので，以下報告する．

2　教育対象

今回，教育プログラムを受けた学生は，著者らの研究室に配属された，九州大学工学部電気情報工学科 4 回生の学部生 4 名である．本教育プログラムに参加した全ての学生は，PC 上でのゲームプログラム等の作成経験がある者はいたものの，いわゆる組込みプログラミングについては全員がその経験がなかった．

九州大学工学部電気情報工学科では，学部 3 年生までに情報工学に関する講義や実験の形式で学部単位での集団的な教育を受ける．その後，学部 4 年生に進む際に研究室に配属され，いわゆる卒業研究に取り組む．

九州大学電気情報工学科のうち，情報工学を専攻する学生は，論理回路，コンピュータアーキテクチャ，アルゴリズム，オペレーティングシステム，コンパイラ，データベース，プログラミング言語，ネットワーク，ソフトウェア工学等，コンピュータのハードウェアやソフトウェアの構造や設計，設計の方法論に関する講義を受けている．

また，学部 3 年生では，i) C 言語によるネットワークプログラミング，ii) CMOS ゲートならびに FPGA による順序回路の実装 [4]，iii) Verilog を用いた FPGA ベースの RISC プロセッサの設計と実装 [4]，iv) 同期機構を有する簡単なマルチタスク OS の実装等の演習を受けている．しかしながら，これら演習は，いずれも講義で教授されたハードウェア，ソフトウェア個別の技術の理解と習得に重心が置かれており，これらの技術を複合的に活用してひとつのシステムを作り上げるような指導は行われていない．

3　教材: 鉄道模型制御システム

組込みシステムの開発においては，複数のハードウェア，ソフトウェア技術を立体的に理解し，複合的に活用する能力が欠かせない．著者らは，前節に述べた学部における学生教育の現状を鑑みて，さらに下記の要件を満たす題材を探し，鉄道模型制御システムを選択した．

- ハードウェアや外部環境を考慮に入れたソフトウェア設計を強いるものであること
- コンピュータ以外の高度な知識を必要としないこと
- 学生一人で実施できる程度の規模と複雑さであること
- 同一のシステムで課題の難易度を自由に変えられること

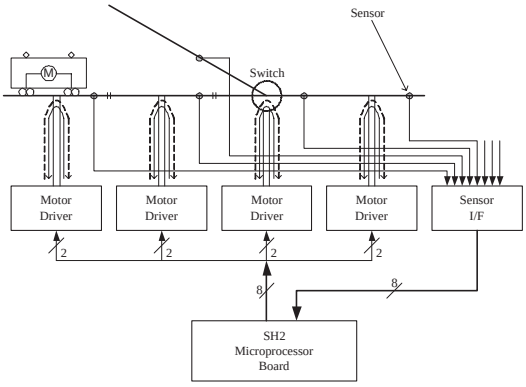

図 1: 鉄道模型制御システム

- ソフトウェアの挙動を肉眼で容易に観測できること
- 細かいマイルストーンの設定が容易であること

市販の鉄道模型セットは，列車やレール，ポイント，信号機，踏切等の各種パーツがシステム化されて販売されており，これらの組み合わせによってシステムの規模や複雑さを変えられる．また，鉄道模型の動作原理は極めて単純でわかりやすいため，比較的早い段階で列車を制御するようなソフトウェアを作成でき，その後はシステム要求を順次追加していくことでマイルストーンを漸進的に設定していくことができる．細かいマイルストーン設定と題材の楽しさは，学生のモチベーションを高く保つために重要な要素である．

図 1 は本プログラムで使用した鉄道模型制御システムの概要図である．システムの中核となるのはマイコンボードである．このマイコンボードには，マイクロプロセッサに SH2 を使用し，トグルスイッチ，プッシュスイッチ，7 セグメント LED，8bit パラレル入力ポート，8bit パラレル出力ポート等の周辺デバイスが搭載されている．リアルタイム OS として ITRON を採用している．鉄道模型制御プログラムは PC 上でクロス開発し，シリアルポートを介してマイコンボードに転送，実行する．

制御対象である鉄道模型では，左右のレールに電流を流し，列車は車輪を通してレールからの給電を受けて自身のモータを回して前後に動く．列車の進

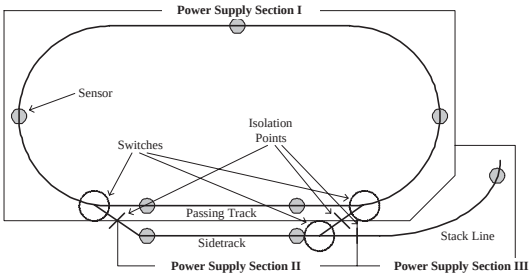

図 2: レイアウト

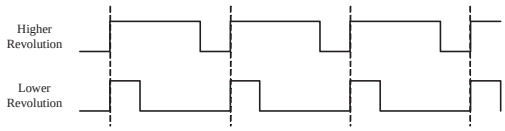

図 3: PWM 制御

行方向，すなわちモータの回転方向は，モータ駆動回路によって電流を流す向きを変えることで制御する．また，電車の速度，すなわちモータの回転速度は PWM 制御により変える．図 3 のようにモータにパルス状の給電を行う．パルスのデューティ比を高くするとモータの回転速度は増し，低くするとモータの回転速度は減る．図 2 は今回の実験で用いた線路のレイアウトである．レイアウト中には，複数の絶縁点が設けられているが，これらの絶縁点で閉じられる 3 つの通電区間それぞれに別々のモータ駆動回路が接続されている．したがって，各通電区間に対して異なった給電を行うことにより，各通電区間に存在する列車に異なった制御を施すことができる．

レイアウト中にはポイントもある．ポイントは，電磁石による磁力を用いた機構により，電磁石に流す電流の向きに応じて，レールの向きが変わる原理となっている．ポイントの制御は，モータ駆動回路を用いて，単に一定時間，ポイントの電磁石に通電するのみでよい．

列車の通過を検知するためにレイアウト中には CdS セルによる光センサが 8 個設けられており，これらの光センサの出力はマイコンボードの 8bit パラレル入力ポートに接続されている．

列車制御やポイント制御のための，モータ駆動回路による給電のタイミング制御は，マイコンボード上のソフトウェアで行う．また，列車の通過は光パラレル入力ポートをポーリングすることにより検知する．

4　教育プログラム内容とその評価

本節では，鉄道模型を用いた組込みソフトウェア教育プログラムについて，その概要，時系列に沿った実施過程，ならびに評価を述べる．

4.1　予備教育

この教育プログラムでは，最初の 5 日間をかけて，学生に対して C 言語の一般的文法知識の講義と小演習を施した．講義内容の大部分は組込みソフトウェアに限られない一般的なものであるが，ビット演算子やシフト演算子を用いたビット操作，I/O レジスタを直接アクセスするテクニックなど，組込みソフトウェアを作成する上で欠かせないテクニックを，ハードウェアの知識とともにいくつか教授した．午前中に 90 分程度の講義を施し，講義の最後に講義内容に基づく演習課題を与え，午後は全てその演習課題の実施に充てた．4 日目の最後には，やや複雑な仕様の演習課題（祝日／振替休日表示つき万年暦）を与え，5 日目には全学生を交えたコードレビューを行い，作成されたプログラムの良否を評価した．

4.2　周辺回路の実装

続いて 2 日を使って，学生に，モータ制御 IC をはじめとする部品一式，オシロスコープ，テスタ等の計測装置，工具一式を与え，モータ制御回路，電車通過センサ回路を製作させた．

この作業はソフトウェアとは何の関係もないが，この作業がもたらす「負の結果」の教育的効果は小さくなかった．製作時の半田不良，結線ミス，部品の損傷等により，電車模型制御プログラム完成後に動作テストをしても，電車模型が期待通りの動作をしないことがよく起こる．ソフトウェア実装の複雑さゆえに多くの学生が自身のソフトウェアのバグを疑うが，実際には上述のようなハードウェアに起因する問題も少なくない．学生は，問題の解決にあたって，問題の原因を複数推定し，計測器やデバッガを用いてハードウェアからソフトウェアに至る各所を検査し，ありえない推定原因を排除し，原因を特定する必要がある．発見した問題の原因を解消には，ハードウェアの修正以外に手がない場合を除いては，できる限りソフトウェアの修正で解消させる．

この作業は，問題の発生源をソフトウェアのみならずハードウェアに広げ，学生のハードウェア，ソフトウェア双方の問題解決力を養成する上で，当初予想していなかった教育的効果があった．

4.3　手動制御プログラムの実装

続く 2 日間をかけて，マイコンボード上のスイッチの設定に従って，電車模型を制御する電車模型制御プログラムを作成させた．ユーザは，スイッチの設定により，各通電区間における電車模型の進行方向と走行スピード，ならびにポイントの向きを設定できる．設定状態はマイコンボードの 7 セグメント LED により表示される．

この作業の目的は，タスクの生成やタスク間通信などリアルタイム OS の提供する機能の扱い，およびハードウェア制御に慣れることである．この作業は，ソフトウェアからハードウェアを介した環境への働きかけがあるのみであり，環境の状態を観測してソフトウェアの動きを変えることはなく，比較的難易度の低い課題と言える．また，この作業は，次の電車模型自動制御プログラムを作成する前に，学生に電車模型制御のノウハウを収集させる目的も有する．

4.4　自動制御プログラムの実装

最後の 6 日間をかけて，電車模型をあらかじめ決めたシナリオに従って自動制御する電車模型制御プログラムを作成させた．

最初の 1 日は，マイコンボード上のボタンの押下を契機に，1 編成の列車を自動的に 1 周させる自動

1. 普通列車が駅を出発する．
2. 普通列車が順次加速し，その後最高速度で走行する．
3. 普通列車が半周回ると，特急列車が普通列車を追って駅を出発する．
4. 特急列車は順次加速し，その後最高速度で走行する．
5. 普通列車は駅の待避線に進入し，順次減速，停止する．
6. 普通列車停止後，特急列車は駅の通過線に進入し，駅を通過する．(普通列車を追い越す.)
7. 特急列車通過後，普通列車はスタックに進入，停止する．
8. 特急列車は 1 周後，駅に停車する．
9. 普通列車はスタックから駅に進入し，停止する．

図 4: シナリオ例

制御プログラムを作成させた．具体的には，駅から出発して，線路前半途中のセンサに到達するまで順次加速し，そのセンサ通過後は最高スピードで走行し，線路後半途中のセンサを通過すると順次減速をはじめ，最後に駅にゆっくり進入し，停止位置のセンサに差し掛かると停止する．

残りの 5 日間は，学生各自に同時に 2 編成の列車を走らせるシナリオを複数作らせて，ユーザが選択したシナリオに従って列車を制御する，自動制御プログラムを作成させた．

シナリオの例を図 4 に示す．

いずれの課題も，センサを介して観測された環境の状態に応じて，ソフトウェアの動作を変える必要がある．学生にはイベント駆動型プログラムを作成させた．

電車模型制御タスクは，タイマならびに列車通過センサ監視タスクから上がってくるイベントに応じて，自身の状態を変え，各通電区間管理タスクやポイント管理タスクに制御指令を出す．学生には，シナリオ通りに列車を制御するために，電車模型制御タスクの状態遷移表を設計させ，その後，状態遷移表に従って電車模型制御タスクを実装させた．

4.5 評価

本教育プログラムの学生の評価は，特に課題の面白さという点において，非常によいものであった．

演習課題において，ハードウェアや外部環境の想定外の振舞によって，ソフトウェアの書き変えを強いられる状況が多く発生した．少なからぬ問題は，制御対象の調査不足やソフトウェア設計時の検討不足によるものであった．特に，列車通過を検知する光センサの絡む問題に，多くの学生が煩わされた．この課題において，当初学生たちは，列車の通過中に光センサは 1 を出力し続け，その他の時は 0 を出力するものと単純に考えて，ソフトウェアを開発した．しかしながら，実機テストの段階で，光センサを原因とする以下の問題が顕在化し，ソフトウェアの変更を強いられることとなった．

例 1: 列車の連結部で光が差し込むため，列車通過中でも車両の隙間の数だけ光センサの出力は 0 になる．この問題は，1 両編成の列車で実験していたときは現れなかったが，3 両編成の列車で実験したときに期待通りに列車が制御されず発見された．□

例 2: 列車の種類によって連結部の隙間の間隔が異なる．光センサとして使用した CdS セルは反応速度が遅いため，連結部の間隔の狭い列車では，連結部が光センサ上を通過しても光センサの出力は 0 には変わらない．また，列車の速度によっても，連結部を認識したり認識しなかったりする．従って，列車の種別や速度によって，列車通過検知の方法を変える必要が生じた．□

ハードウェアや外部環境に起因する問題によってソフトウェアの変更が強いられる，組込みソフトウェア開発の難点を再現するという，本教育プログラムの当初の目的は達成できた．また，本教育プログラムで用いた鉄道模型制御が，この教育目的を達成する上で適した題材であることも認められた．しかしながら，今回の演習では，実装を急ぐあまりに，ソフトウェア開発の進め方がかなり ad-hoc になっており，ソフトウェア設計時のハードウェアや外部環境に対する十分な検討が不足していたことは否めない．今後，分析・設計工程での指導を十分した上での，演習実施とその評価が望まれる．

5 関連研究

組込みシステムに関する教育の必要性が認識されるにつれ，組込みソフトウェア開発に関する教育活

動 [6, 7, 8]，あるいは教育事例 [2, 9, 11] の報告がなされるようになった．

学生にハードウェアや外部環境を意識させる組込みソフトウェア開発に関する教育の実施例も見られる．Salewski らは，車載スピードセンサをマイコンボードベースのシステム，ならびに CPLD ベースのシステムとして，異なった実装をさせる教育例を報告している [11]．SESSAME では電子ポットを題材に構造化分析手法等の設計手法を修得するための教育プログラム [6] を，大谷らはライントレースカーを制御対象として，オブジェクト指向分析技術を修得するための教育プログラムを実施している [9]．

6 まとめ

以上，鉄道模型を用いた組込みソフトウェア教育プログラムについて報告した．

変更容易な組込みソフトウェアは，ソフトウェア設計時には予期しないハードウェアや外部環境に因る問題により，ハードウェア／ソフトウェア結合後にその変更を余儀なく強いられる宿命にある．ゆえに，組込みソフトウェア教育プログラムは，ソフトウェアのみで完結するものでなく，ハードウェアや外部環境も設計，開発，実装において考慮させるものとすべきである．筆者らが題材として選択した鉄道模型制御プログラムは，上記の要件を満足し，かつ学生一人で 2 週間ほどで終えられる程度の規模と複雑さを持つ，大学教育向けのものであった．また，学生の能力に応じて要求を追加，削減することによって，容易に複雑さを増減することができる．

今日，多くのメーカは，世界各国の市場のさまざまの消費者の要求に対応するべく，一種類の製品についても幾種もの製品モデルを生産している．事前に定められた市場領域向けに，同一製品系列に属する複数の製品用のソフトウェアを効率的に開発するソフトウェア開発方法論として，プロダクトライン開発方法論が近年注目されている [1, 3, 5, 10]．レイアウトの変更や列車の編成数の追加，自動制御シナリオの追加等により，複雑度や規模を容易に変化できる鉄道模型制御システムの特徴を活かして，今後は本組込みソフトウェア教育プログラムにプロダクトライン開発方法論を盛り込んでいく予定である．現在は，Gomaa によるプロダクトライン開発方法論 PLUS[3]，ならびに韓国浦項工科大学による FORM[5] の採用を検討している．

参考文献

[1] P. Clements and L. Northrop, *Software Product Lines: Practice and Patterns*, Addison-Wesley, 2001.

[2] S. A. Edwards, “Experiences Teaching an FPGA-based Embedded Systems Class,” *Proc. 2005 Workshop on Embedded Systems Education (WESE2005)*, pp.52–58, 2005.

[3] H. Gomaa, *Designing Software Product Lines with UML: From Use Cases to Pattern-Based Software Architectures*, Addison-Wesley, 2004.

[4] 林田 隆則, 太田 喜一郎, 井上 創造, 松永 裕介, 倉爪 亮, 中西 恒夫, 藤田 博, 松崎 隆哲, 「九大における FPGA を用いたハードウェア設計教育」, DA シンポジウム 2004 論文集, pp.49–54, 2004 年.

[5] K. C. Kang, S. Kim, J. Lee, K. Kim, G. J. Kim, and E. Shin, “FORM: A Feature-Oriented Reuse Method with Domain-Specific Reference Architectures,” *Annals of Software Engineering*, No. 5, pp.143–168, 1998.

[6] 組込みソフトウェア管理者・技術者育成研究会, *http://www.sessame.jp/*

[7] 九州大学システム LSI 設計人材養成プログラム, *http://www.slrc.kyushu-u.ac.jp/qube/*

[8] 名古屋大学組込みソフトウェア技術者人材養成プログラム, *http://www.nexcess.itc.nagoya-u.ac.jp/*

[9] 大谷 芳寛, 大山 勝徳, 金子 正人, 武内 惇, 藤本 洋, 「オブジェクト指向分析技術修得得のための ET ロボットコンテストへの取り組み」, 組込みソフトウェアシンポジウム予稿集, pp.100–103, 2005.

[10] F. v. d. Linden, “Software Product Families in Europe: The Esaps & Café Projects,” *IEEE Software*, Vol.19, No.4, pp.41–49, 2002.

[11] F. Salewski, D. Wilking, and S. Kowalewski, “Diverse Hardware Platforms in Embedded Systems Lab Courses: A Way to Teach the Difference,” *Proc. 2005 Workshop on Embedded Systems Education (WESE2005)*, pp.66–70, 2005.