「C,RTOS（μITRON）、ZIPC による開発手法の比較」

エンベデット・エジュケーション有限会社 教育システムエンジニア

青柳　良介

１．はじめに

私たちの身の回りで MCU を使用していない製品はほとんどないくらいに、MCU は使用されるようになってきました。

また、半導体の集積度の飛躍的な向上に加えて RISC 方式の採用により最近の MCU はより高性能のものが出てきて、携帯電話、PDA 機器、デジタルカメラ、等から大きな製品では自動車までと、ますます多くの製品に MCU は使用されるようになってきました。

しかし、システムの構成は基本的に図１．のようになり、どのような製品でも大差はありません。ただし、規模の小さいものは、全ての要素がひとつの IC になってしまうこともあります。

そして、高速性を要求される基幹部分は ASIC（ハード）として機能を用意します。ただし、現状でも高機能な ASIC（ハード）の開発には３～５年程度が必要とされております。そのために、ASIC（ハード）を共通に使用して、製品の展開をソフトウエアで行うことが多く行われております。

また、大雑把な機能分担の比率は ASIC（ハード）が３０%、ソフトが７０%となっております。そして、多くの機能をソフトで行う為にワンチップの MCU でも２５６K バイト（ROM）は一般的になっております。また、メモリーが外付けの場合はメモリー（ROM/RAM）が数 M バイト～数十 M バイトと数年以前のパソコンと変わらない程度に大きくなってきております。そして、この傾向は今後とも変わらずに続くと思われます。

また、製品展開は、ほぼ同時に数種類行われるのが一般的です。そして、開発の期間は３ヶ月から１年程度がもっとも多くなっております。商品によっては 1 年から数年のものもあります。

そのために、大きなサイズのソフトを多種に渡り、短期間で開発を行うことが企業には要求されてきております。

しかしながら、「CASE」ツール等を含めた新しい手法の導入はあまり進んでおりません。

ひとつは、多くの「CASE」ツール等は「ディスクトップ」系向けが多く、「組込

み」系には向きません。

　また、新しいツール、手法を導入する場合は従来のツール、手法と大きく異なります。そのために担当者がなれる為にかなりの時間を必要とします。時間に追われている開発の現場では、開発効率を上げるために新しいツール、手法を導入をしたいが、担当者がなれる為の時間がとれないというジレンマに陥り導入に二の足を踏んでいることが多いと思います。

　そこで、導入の参考にということで「新しいツール、手法」と従来の手法を比較いたしました。

図1．システム構成

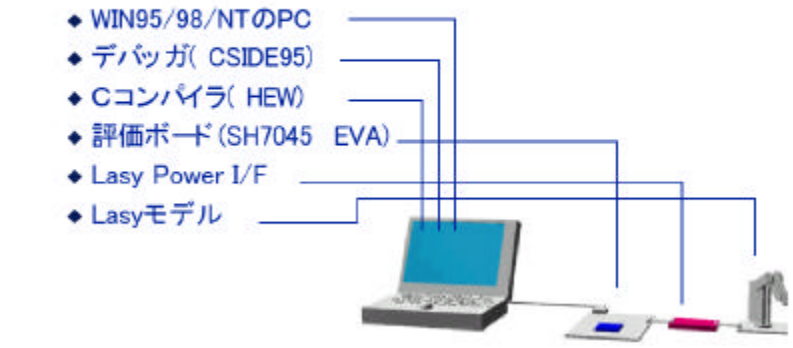
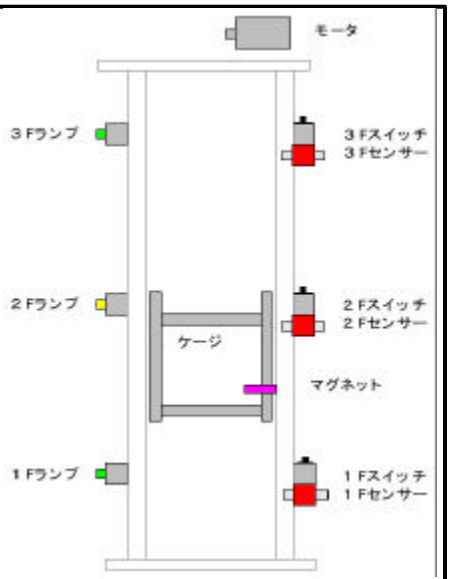


図2．エレベータ仕様書

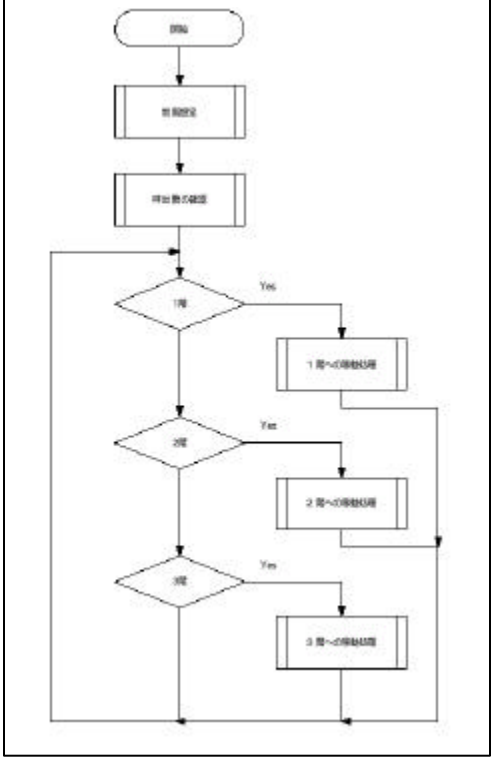


1．初期状態
ケージを１階に移動。
1階のランプを点灯。

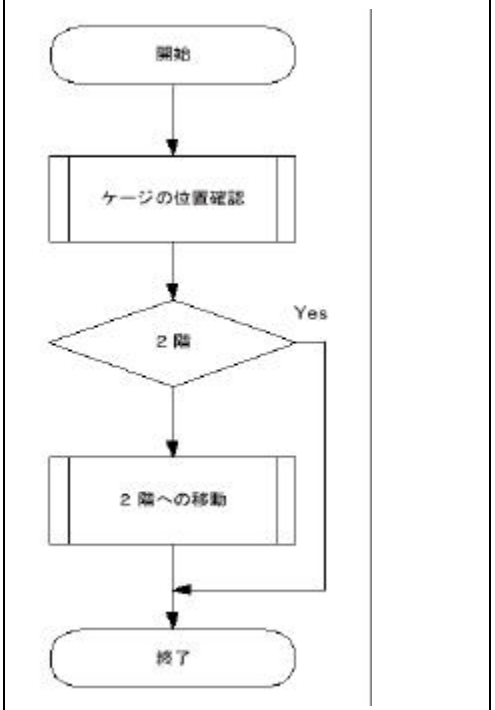
2．待ち状態
各階の呼出し SW が押されるのを待つ。

3．移動状態
停止階のランプを消灯。
呼出し階を検出後にケージを停止。
呼出し階のランプを点灯。

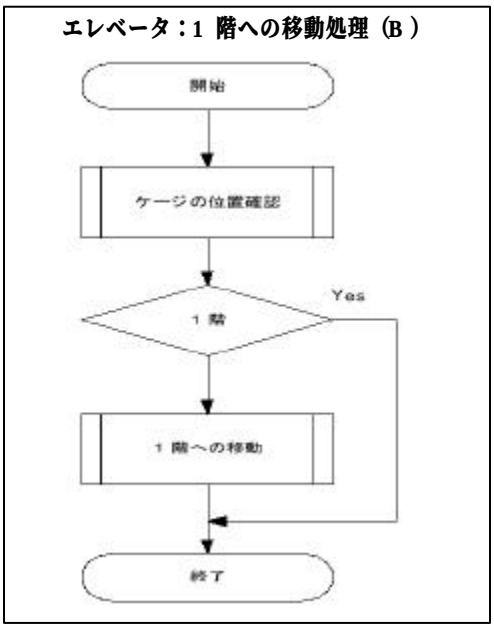
エレベータ：ゼネラルフロー（A）



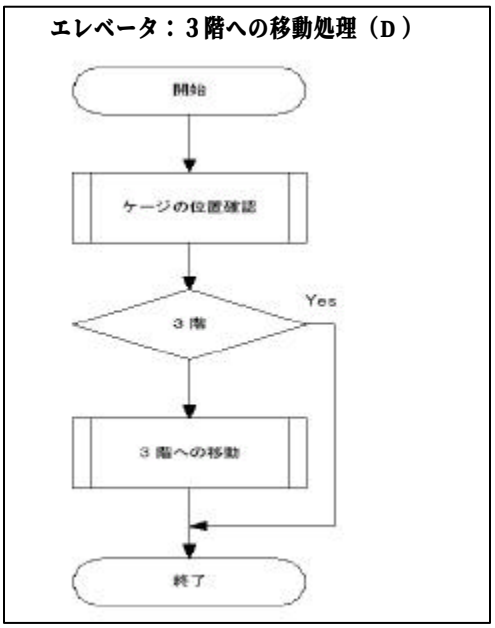
エレベータ：2階への移動処理（C）



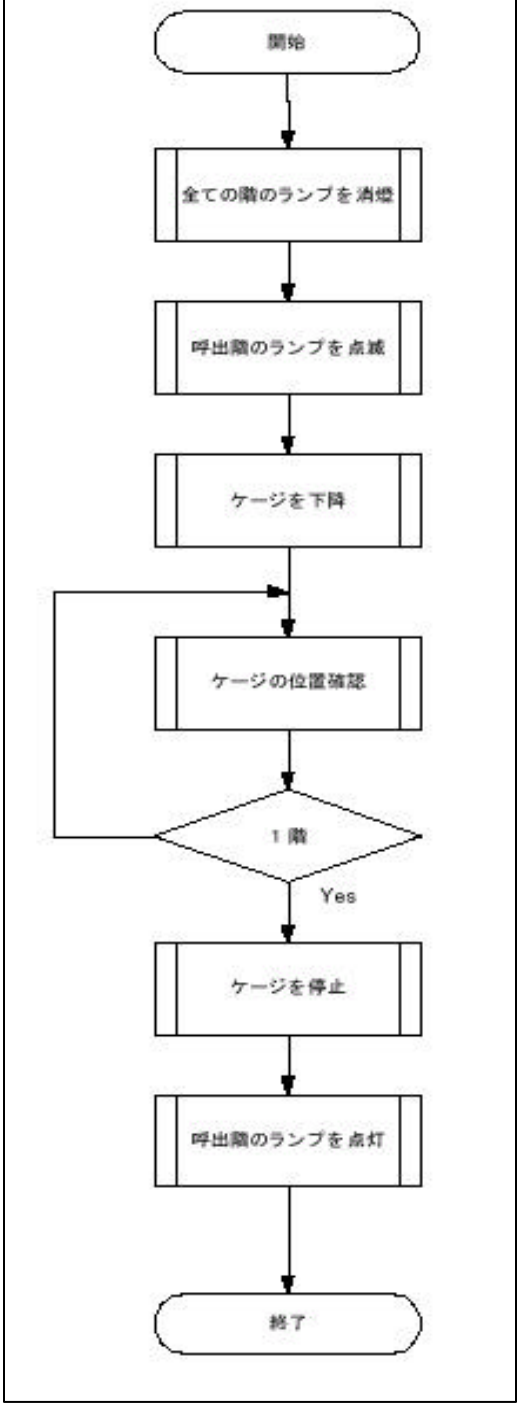
エレベータ：1 階への移動処理（B）



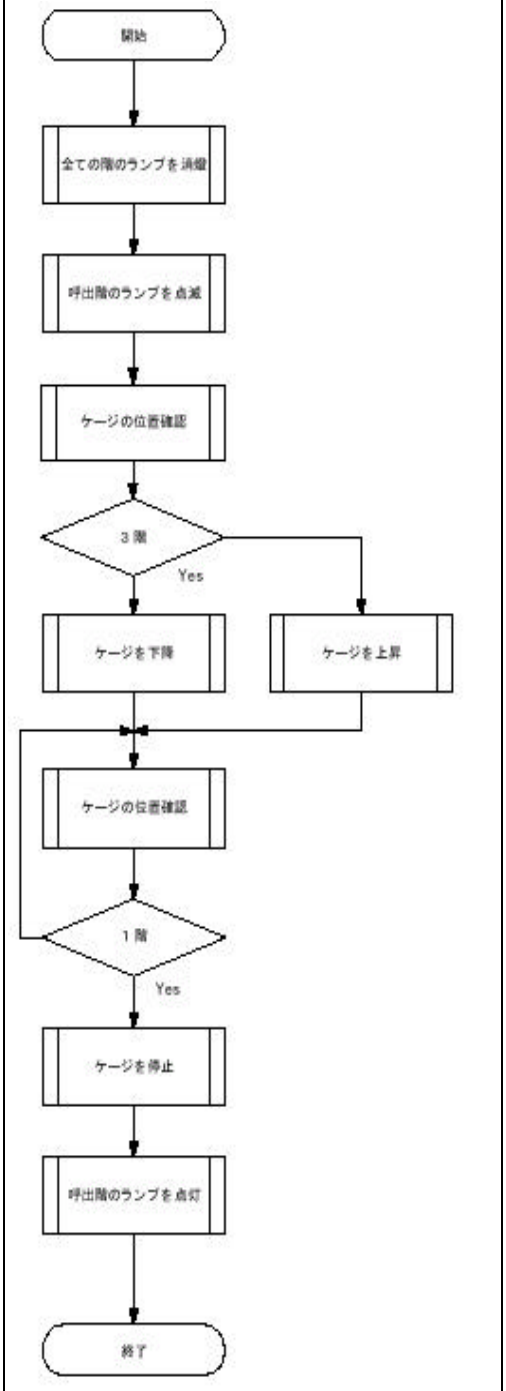
エレベータ：3階への移動処理（D）



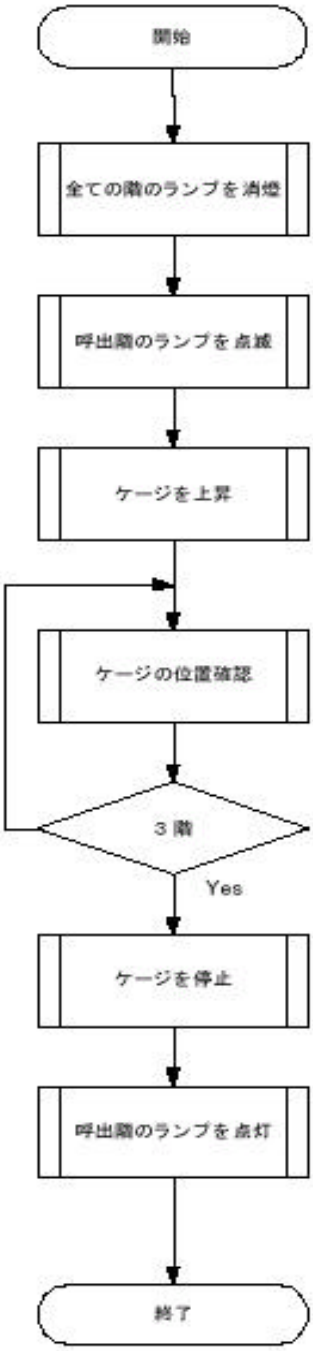
エレベータ：1 階への移動（E）



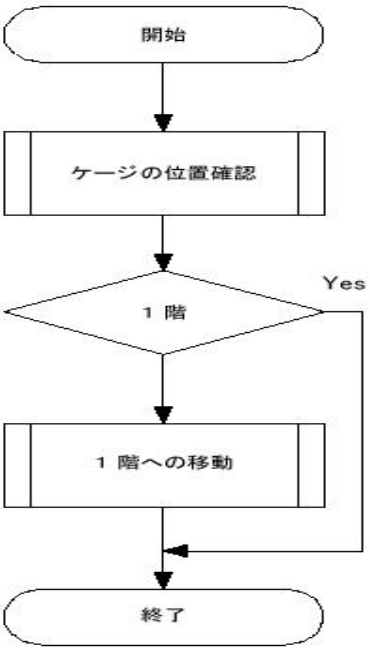
エレベータ：2階への移動（F）



エレベータ：3階への移動（G）



エレベータ：初期設定（H）



2．目的

「新しいツール、手法」としては、「ZIPC」を使用します。また、比較する対象としては、従来の「C 言語のみのプログラミング」と「RTOS（μITRON）による」とします。

また、従来の評価の場合ですと実際の商品の開発等に使用されているために、よい点、悪い点等は発表されますが、実際のプログラム等は原則として発表されることはありません。

しかしながら、導入を検討しようとした場合は、プログラム等の具体的な内容が分からないと、的確な判断が下せない

ことが多くあります。

そこで、新しい試みとして「具体的なターゲット」を開発を行い、プログラム等をはじめとし、全ての内容を公開して従来の開発手法との違いを比較することにより、分かりやすくして、導入検討の参考材料にしたいと思います。

3．方法

「エレベータ」を制御するソフトを従来の「C 言語のみのプログラミング」、「RTOS（μITRON）によるプログラミング」、「ZIPC によるプログラミング」と比較を行います。

3.1 構成

図1．に環境の構成を示します。対象の MCU は「SH２」としました。

ソフトの開発環境は日立純正の「Hitachi Embedded Workbench」,デバッガは「CSIDE95」で評価ボードはコンピューテックス社の「SH7045　EVA」としました。

μITRON はミスポ社の NORTi、ターゲットのエレベータは Lasy 社の「Lasy Control」を使用いたしました。

3.2 手順

「C 言語のみのプログラミング」では図２．の仕様書をもとに「フローチャート」を作成し、C 言語でプログラミングを作成し、デバッグをおこないました。また、「RTOS（μITRON）によるプログラミング」では、図３．状態遷移図と図４．タスク関連図も元にプログラミングを作成し、デバッグをおこないました。

「ZIPC によるプログラミング」では、図２．の仕様書もと状態遷移表を作成して、プログラミングを作成して、デバッグをおこないました。

図３．エレベータ ：状態遷移図

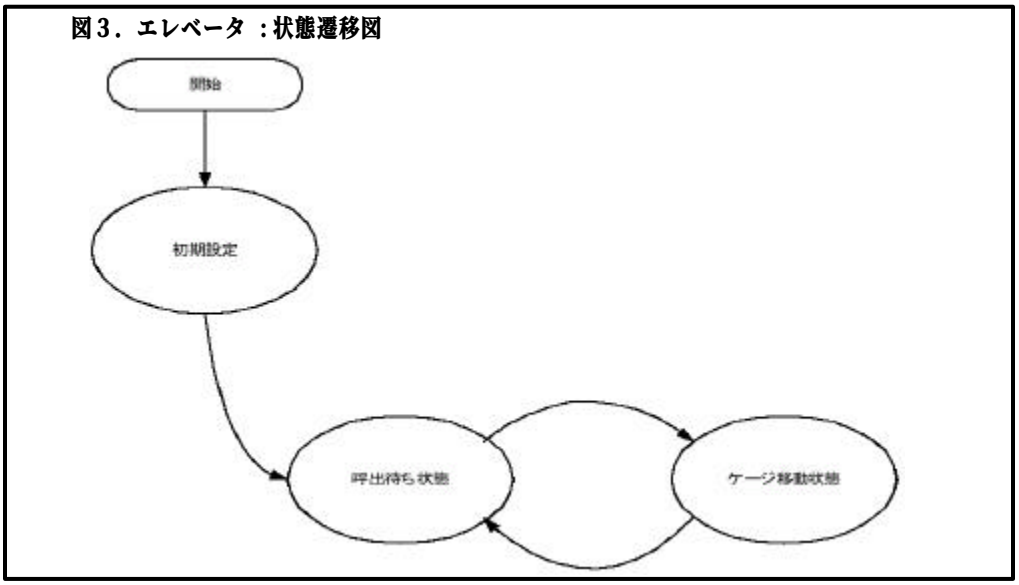
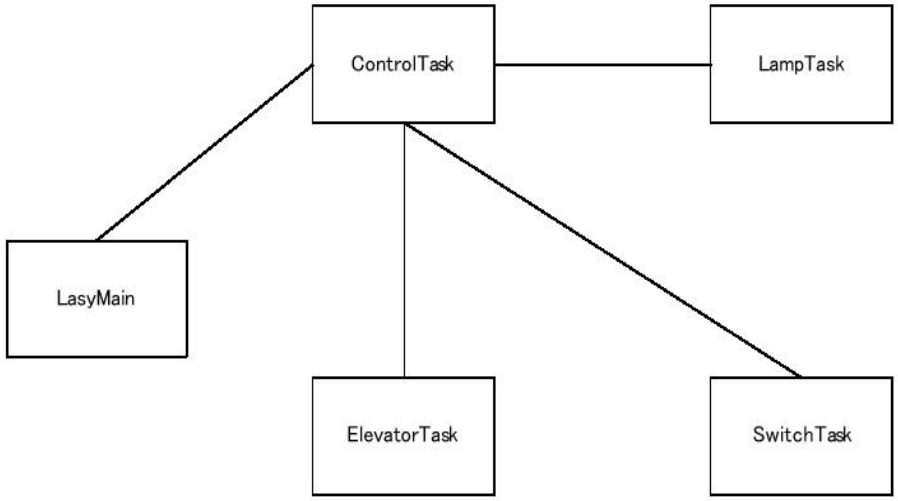


図４．エレベータ：１タスク関連図



＜タスク説明＞

Lasy_main：タスクの生成等の初期化を行います。

ControlTask：エレベータの全体の制御を行います

ElevatorTask：Control Task からの要求に応じてケージの制御を行います。上昇、下降、停止

SwitchTask：呼び出しスイッチの状態を Control Task に伝え、ケージの移動を要求します。

LampTask：Control Task からの要求に応じて各階のランプを制御します。点灯、消灯、点滅。

3.3 結果

結果と書きましたが、今回は定量的な部分ではなく、使い勝手等の定量的に評価しにくい部分を比較した感想ということで書かせていただきます。

まず、従来の手法に比べて全体が大変分かりやすいものになりました。

具体的には図５．の「エレベータの状態遷移表」でフローチャートの（A）から（G）までの６個のフローチャートを 1 枚の状態遷移表で表現できることでお分かりいただけると思います。

また、イベントと状態を分けて表記するので、システムの動作についてより分かりやすいものとなっております。

ただし、「ZIPC」は従来の手法とことなる手法のために、独特の概念とツールの使用を覚える必要性が生じてきます。

4．まとめ

簡単な評価ですが、ソース・コード等を含めてオープンにすることにより、皆様の評価のもととなる材料を提供することで、評価の一助となれば幸いです。

図５．エレベータ状態遷移表

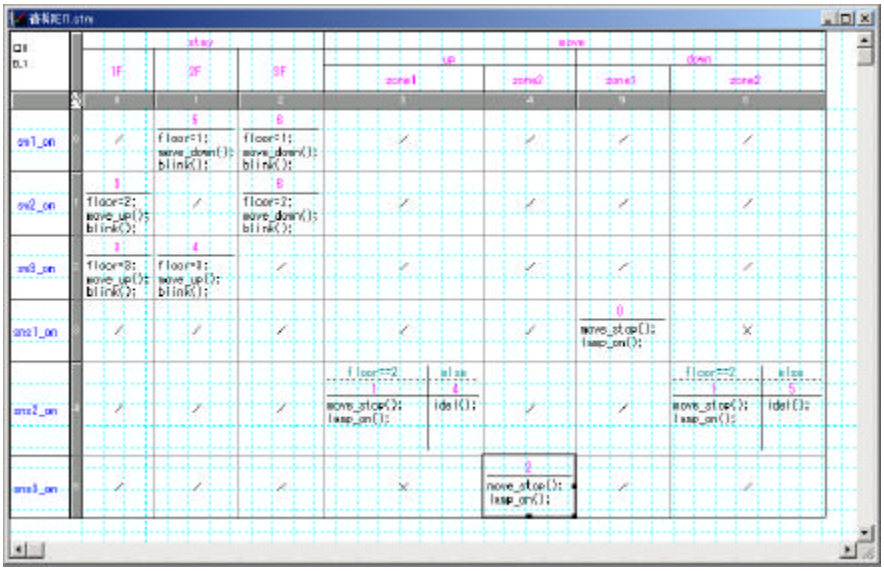


図１のシステム写真