資　料

組込み技術を用いた機器制御システムの研究開発（第１報）

秋元英二

Development of machine control system that uses Embedded Technology (Part1)

AKIMOTO Eiji

組込み技術で利用可能なリアルタイム OS の Linux と μTRON に着目して、その組込みシステムを構築をするとともに、機器制御を目的とし通信ポートからの外部機器を動かすための実験装置とそれを動かすプログラムを作り評価を行った。この結果、各組込み OS 上で動くプログラムと実験装置がうまく連動して動作することが検証できた。

キーワード：組込み、Linux、μTRON、制御

はじめに

情報家電や自動車産業に代表される日本の産業の柱となっている技術の１つとして「組込み技術」がある。この組込み技術は、今後の我が国産業を担う重要な基幹技術として位置づけられている。[1)]しかしながら、組込み技術の開発者不足が叫ばれており、全国的にみると現在 19 万３千人ほどがいるものの、ソフトウェア開発者の中の 4.1%に過ぎず、今後、さらに９万４千人が不足すると予想されている。[1)] また、近年は、OS 部分とアプリケーション部分を切り分けた「組込み OS」を用いた開発手法が盛んになってきている。

一方、県内の機械製造メーカに対して「組込み技術」について聞き取り調査したところ、①本格的な取り組みをしたいが、従来の自社独自の方法で慣れている。②取り入れたいが、人と時間の余裕がない。③取り入れることで効率化が期待できるが、「組込み技術」のノウハウがない。などの意見が大勢であった。

そこで本研究では、今後、県内の機械製造業に必要とされる組込み技術について、機械制御を有効に行う上で重要となる「組込み OS」の構築と外部機器制御のモデルを製作し、ソフトウェアとハードウェアの両面からその有用性について検証したので報告する。

実験方法

１．実験装置

今回実験に用いた「組込み OS」は、Linux と μTRON を採用した。これは、全国の公設試での取り組みにおいて、この２者が拮抗して大勢であったり、県内企業の聞き取り調査においても、この２者の認知度が高く、利用希望が多かったためである。

図１に使用した装置を示す。左側が、Linux を組み込む装置((株)FASE)で、CF(コンパクトフラッシュ)に Linux OS を組み込むことで動作可能になる。入出力端子としては、モニタ端子のほかに通信ポートとしてシリアルポートと USB ポートを備えている。

図１　Linux 装置(左)と μTRON ボード(右)

右側は、TRON ボード(（株）中央製作所)で、ボードの中に μTRON OS を導入し、各種の操作を実行する。ボード上には、４個の DIPSW と３個の LED 及び、シリアルポートと AD ポートなどを備えている。

２．開発環境

(1) Linux について

組込み用 OS に Linux を導入するにあたり、ターゲットを作る Linux の構築が必要となってくる。そこで、今回は Fedora5 で作り上げた。

Fedoar を採用したのは、Linux でシェアの高い Red Hat Linux の後継であり、もっとも安定的に情報を入手できるためである。

図２は、Windows 上で任意の仮想マシンを構築する VMware の中に開発環境の Fedora5 を乗せて開発しているところである。VMware を仲介したのは、組込み OS 構築時に失敗や致命的なエラーなどが生じても、開発環境自体を容易に削除し再構築が可能であるほか、開発環境の容量を簡単に指定できるためである。つまり最小の容量にすることで領域確保などの開発環境構築にかかる時間が少なくてすむというメリットがある。しかし、仮想マシンであるために処理スピードが若干遅くなるデメリットがある。

この研究は、「組込み技術を用いた機器制御システムの研究開発」の予算で実施した。

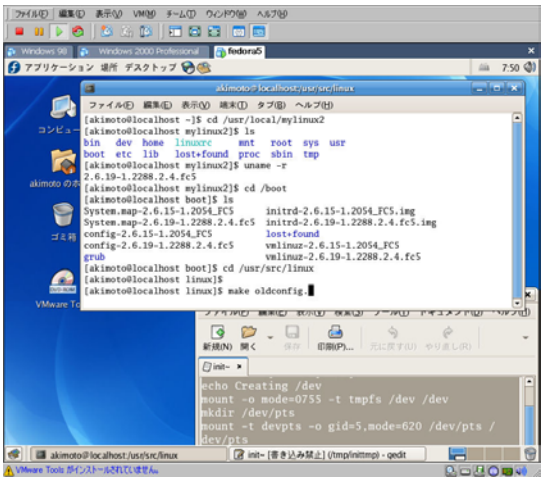

図２　VMware で組込み Linux を構築

今回の開発マシンの仕様は、CPU2.5GHz、メモリ１GB であった。Fedora の構築容量は、CF にあわせて１GB とした。

(2) μ TRON について

使用したボードは、リアルタイム OS(TOPPERS/JSP) の μ TRON の仕様となっている。開発環境としては、Windows 上に Cygwin を構築し、この中で TRON 用のプログラムを開発、コンパイルしていく。Cygwin の他に構築に利用したプログラムは表１のとおりである。

表１　μ TRON 開発環境

プログラム名	バージョン	説　明
binutils	20040312-1	GNU assembler
bison	20030307-1	parser generator
gcc、gcc-g++	3.3.1-3	C complier
make	3.80-1	`make`utility
perl	5.8.2-1	`perl`utility

また、出来上がったプログラムの書込みには、Flush Fire を用いた。make により構築された exe ファイルと同時に bin ファイルが作られ、この bin ファイルを直接ボードに送り込むことで実行が可能となる。

２．外部機器

図１の各装置に入れ込んだ組込み OS を用いて、その上でのアプリケーションソフトを開発し動作検証を行った。機器制御のためには、図１の各本体装置と外部機器との連動(入出力)を検証する必要があり次の実験を行った。

(1) Linux について

ｉ)USB を用いた入力実験

本ポートを入力としてデータの取り込みを行った。

ii)シリアルポートを用いた LED、SW の操作実験

シリアルポートは、標準的な通信ポートで、D-SUB9pin 中のいくつかの通信ポートを利用した。

(2) μ TRON について

ｉ)DIPSW と LED の連動実験

ボード上の DIPSW と LED をプログラムにより関連付けた動作をさせた。

ii)小型モータ回路の製作

速度制御をするために、シリアルポートを経由した送受信を試みた。その制御対象として小型モータ回路の試作を行った。

結 果 と 考 察

１．組込み Linux の構築

Linux の組み込みには、以下(図３)の手順を踏んだ。

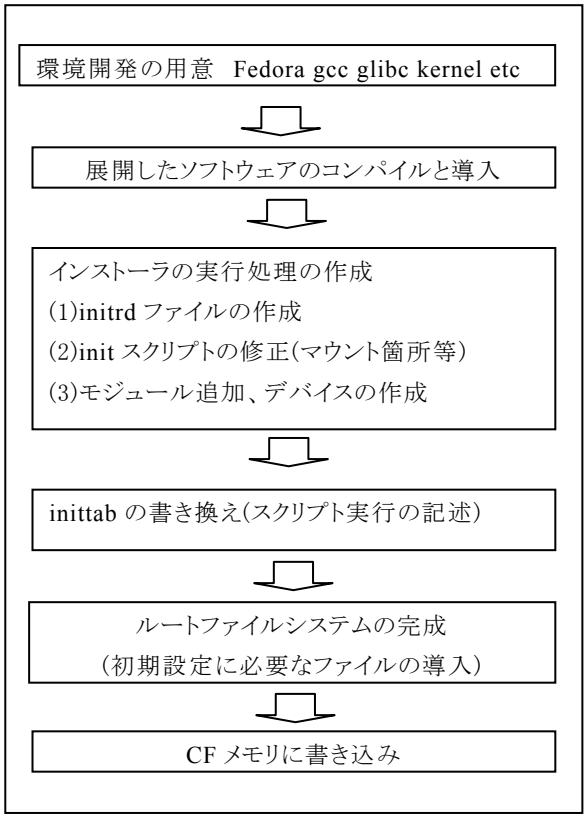

図３　組込み用 Linux の構築フロー

これら一連の作業を、正確に実行することで、組込み OS としての Linux が完成する。その結果、CF 内に容量 341MB で構築することができた。(ただし、GNOME や KDE などの GUI 環境は含まない。)

PC に Linux を構築する場合には、少なくとも 500～600MB 程度が必要とされるほか、上述の GUI などを入れ込むとさらにその容量がギガ単位にまで拡張する。

また、512MB の比較的容量の小さな CF に入れ込む事ができるのでシステムの価格も安くできる。あるいは逆に、1 GB やそれ以上の大容量の CF にすると空き領域に

大きなアプリケーションを導入することが可能となる。

さらに OS の容量をコンパクトにする方法を検討すると、コマンドやライブラリの中に含まれているデバッグなどに利用するシンボル情報を管理するシンボル・テーブルについて strip を使って削除し、shrink を実施することにより小さくすることが可能である。この作業を１つ１つのファイルについてチェックし適合させることで、OS 自体のサイズを減らすことが可能と思われる。

２．外部機器との連携

(1)Linux について

ⅰ)温度センサからのデータの取り込み

外部からの入力として、今回ディジタル温度計を使用した。これを図４に示す。USB 接続により Linux 装置に接続し、標準デバイス(モニタ)にリアルタイムに数値を出すことができた。

図４　USB 接続型の温度計モジュール(Media Lab.)

これを発展させると標準デバイスから通信ポートに変更して外部に発信することも可能である。

ⅱ)シリアルポートを用いた LED と DIPSW 操作

Linux 装置には、USB の他に外部との通信手段としてシリアルポートが標準装備されている。これを使い、LED の点滅指示と SW 操作による ON/OFF 情報の取り込みができるようにした。この目的は、上記 USB ポート以外の外部出力の検証として LED を点滅することと逆に入力の検証として SW での外部状態の変化をキャッチすることにある。

シリアルポートには、D-SUB9 ピンの場合、表２に示すような役割が各々に割り振られている。

表２　シリアルポートのピンタイプ

ピン	ﾗﾍﾞﾙ	制 御 信 号	タイプ
1	DCD	Carrier Detect	Control
2	RXD	Received Data	Date
3	TXD	Transmitted Data	Date
4	DTR	Data Terminal Ready	Control
5	GND	Signal Ground	Ground
6	DSR	Data Set Ready	Control
7	RTS	Request to Send	Control
8	CTS	Clear to Send	Control
9	RI	Ring Indicator	Control

シリアルポートは、通信回路として通常用いられるが、これらの制御信号は個別にアクセスできるので、今回は DIO として使用した。この表中において、LED 点灯用として DTR と RTS を、SW からの受信用としては、DCD と RTS を割り振った。図５の回路図が今回試作したものである。(電源回路は省略している)

シリアルポートは、RS232C レベルであるので TTL レベルへの相互変換として MAX232CPE を挿入した。

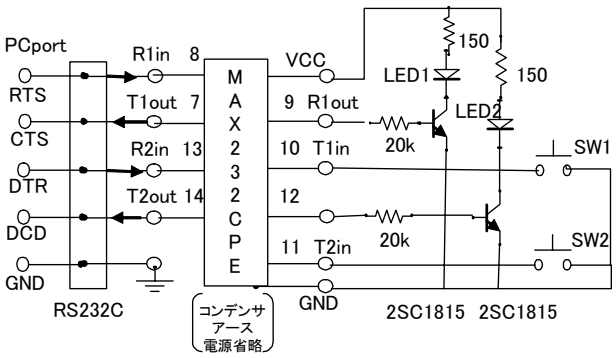

図５　LED と SW の動作回路

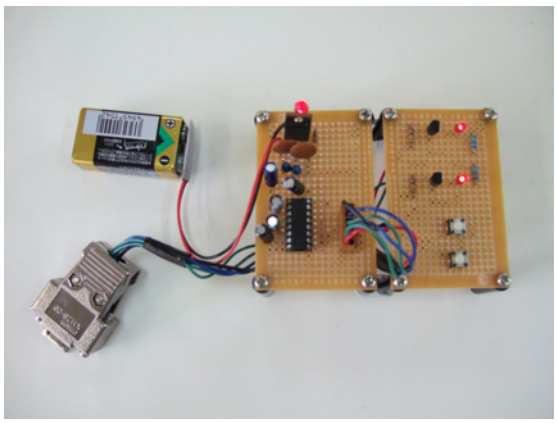

図６　試作したボード

図５を元に図６を試作した。プログラムは、C 言語で作成した。入出力ともに指定どおりの結果を得た。入力ポートに対しては、入力待ちに無限ループで常時待機するようにすることで信号を取り込むことができた。

(2) μTRON について

ⅰ)DIPSW と LED の連動

図１の μTRON ボードでは、ボード上に DIPSW と LED が配置されている(図７の赤丸)。そこで、この２つを連動させるプログラムを作り、動作検証を行った。

DIPSW 及び LED ともに、トロン OS 内ではハードウェアとのつながりにおいて、それぞれ固有番地を有している。それらの番地をプログラムで指定することにより４つの DIPSW と３つの LED とが任意に結びつけられるようになる。

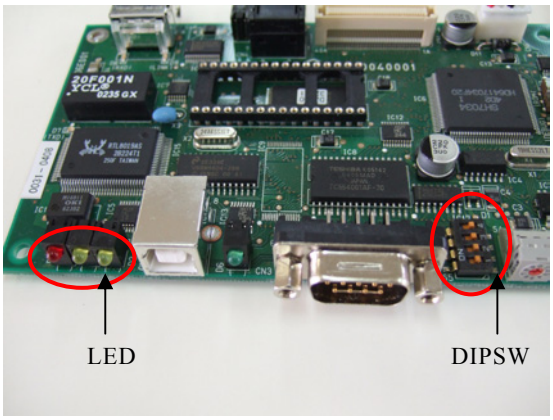

図７　DIPSW と LED の連動

Cygwin において作成したプログラムをコンパイルし、図８の Flush Fire にて、ターゲットに送り込んだ後、DIPSW の操作により LED の点滅を確認した。なお、図７に示すように、本ボードにはシリアルポートも装備されており、このポートを使った入出力も可能である。今後は、その利用方法について検討を行う。

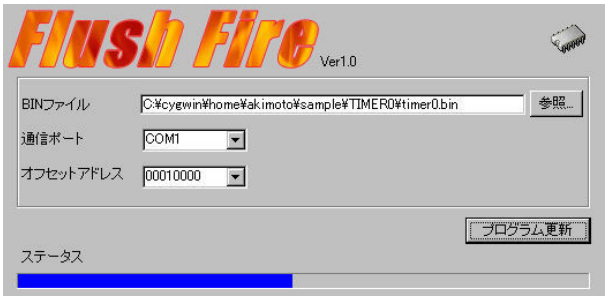

図８　Flush Fire によるプログラム転送

ii)小型モータ回路の製作

上述のシリアルポートを用いて、現在小型モータを駆動させる回路を検討した。DC モータドライバモジュールとしては、Motor Mind B(Slotions Cubed,LLC)を利用した。

図９　モータ制御のための通信解析

これは、コマンドを組み合わせるとモータのスピード制御ができるとともに、確認バイトも返信してくるのでデバッグには都合がよい。図９は、現在開発中の模様を示した。作成したプログラム中で発生させたコマンドが通信線上において、正常に伝送されているかどうかを解析しているところである。この図から、コマンドがヘキサ・データとして送信(青字)され、返信信号(赤字)が応答していることがわかる。

今後は、このプログラムを完成させるとともに回路を作り上げて、モータ制御できるシステムの構築を行う。

ま　と　め

今後、ますます必要となってくる「組込み技術」の根幹となる組込み OS を用いた各種のシステム構築を行った。その結果以下のことがわかった。

１．Fedora を使って組込み Linux の構築を行った。OS の容量として 341MB の小さなものとなった。

２．Linux OS では、外部接続として USB やシリアルポートを利用して温度情報の獲得や LED、SW の動作検証を行い、良好に動作することを確認した。

３．μTRON OS を使った実験では、DIPSW と LED を連動するプログラムを作成し、その有用性を確認できた。

今後は、より多くのデータ交換が可能な外部機器の制御モデルを作製し、即応性と信頼性を加味した組込みシステム構築の研究開発を実施する。

文　　献

1) 経済産業省:2006 年版組込みソフトウェア産業実態調査報告書.

2) 株式会社中央製作所:ZUNDA SH1 ガイド初版(2004).

3) 重松保弘:UNIX プログラミング入門講座(アスキー出版局)(1994).

4) 日経 BP 社:日経 Linux6(2006).