# １１　組み込みソフトウェアの開発事例

## ― 地域企業への技術支援の一環として ―

山形大学工学部技術部
情報技術室　鈴木貴彦

### １．背景

当技術部では、技術職員による学内外からの技術支援および技術相談に応える体制をとっている。この度、県内の企業より新規開発のＯＡ機器に係わる組み込みソフトウェア開発を受託する機会があったので、公表可能な内容に限定して紹介する。具体的にはマイクロコントローラを用いてＬＥＤを点灯制御するだけの単純な機能である。もちろんこの程度のソフトウェアであれば同社のソフトウェア開発部門にて難なく対応可能とのことである。しかし問題なのは、本件を依頼された方が所属する製品企画部門とソフトウェア開発部門とは互いに遠く離れているので、製品仕様を詰めるために頻繁に改変されるような作業をソフトウェア部門に頼んでしまっては、遠隔地同士がゆえに意思疎通に大きな負担と時間が費やされてしまうことである。そこで可能であれば、ＬＥＤの点灯パターン等の「人間の感性に関わる製品仕様」に関しては、製品企画部門において実働させながら仕様検討ができるようにし、仕様が確定した後にソフトウェア開発部門へ仕様書を渡せる体制にしたいとのことであった。今回のような場合、一般的には任意波形発生器を用いれば容易にＬＥＤの点灯制御を行うことができる。しかしこのためだけに高価な機器を用意するのは難しいことと、出張時の運搬や製品企画会議の席上でのデモを考慮して、今回のようなマイクロコントローラを用いたＬＥＤ点灯制御プログラムの開発を行うこととなった。

### ２．開発目標

前述の要求事項を考慮して、開発目標を下記の通り設定した。

(1) ＬＥＤの明るさを制御可能なこと。
(2) 現場にてＵＳＢインターフェース経由でＬＥＤ点灯パターンを変更可能なこと。

上記(1)は、単純な ON/OFF だけではなく中間的な明るさも制御できることであり、これは一般的な手法であるパルス幅変調（ＰＷＭ)によって実現する。(2)は、筆者の手を離れた後も当該企業において特別な技術を要しないで所望の点灯パターンを設定できることを意味する。そのためにＬＥＤ点灯パターンを定義するコマンドを独自に定義し、これをＵＳＢインターフェース経由でマイコンボードにダウンロードできるようにする。

### ３．開発環境

図１に、今回使用したマイクロコントローラボード(以下「マイコンボード」)と開発環境を示す。このマイコンボードは、ＮＥＣエレクトロニクス社から発売されている同社の８ビットマイクロコントローラ 78K/LG2 シリーズのスターターキットである [1)]。開発に必要なソフトウェアは全て同梱されており、最新版は同社のホームページから無償でダウンロードできる。オブジェクトサイズには制限があるものの、コンパイルからデバッグまでシームレスに開発を行うことが可能である。

プログラミングは全てＣ言語を用いて行った。随所に割り込み処理を用いているが、特にアセンブリ言語を用いなくとも割り込み処

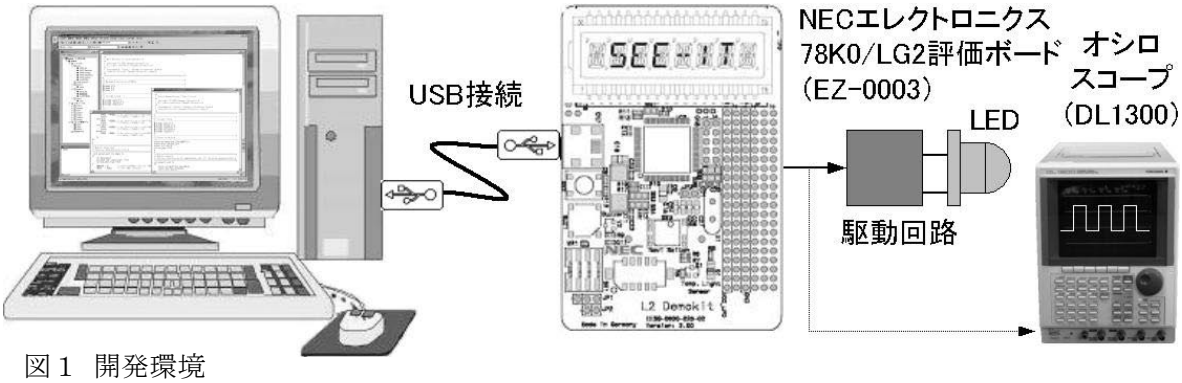


図１ 開発環境

理ルーチンを書くことができる。コンパイルされたオブジェクトファイルはデバッガによってＵＳＢインターフェースを介してマイコンボード上のマイコンチップ(μPD78F0397D)のフラッシュメモリに転送され、実行される。更にＵＳＢインターフェース経由でオンチップデバッグも可能である。マイコンチップの汎用 I/0 ポートから出力されたＬＥＤ制御用(ロジック)信号は、オシロスコープによってそのタイミング等を確認した。

## ４．ＬＥＤの制御方法

ＬＥＤの明るさは、一般的な方法であるパルス幅変調(ＰＷＭ)によって制御した(図２)。

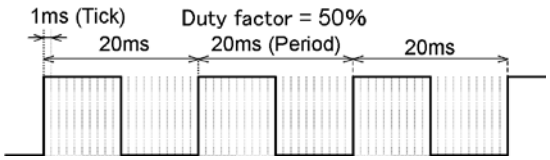


図２ LED 駆動用のパルス幅変調(PWM)

これは人間の目の残像を利用するものであり、一定以上の周波数で点滅しているＬＥＤは連続点灯として認識され、更にその Duty Factor を変化させることによって連続的にその明るさを変化させることが可能である。ＰＷＭの制御可能な最小時間(Tick)と制御周期(Period)、および各周期毎の Duty Factor は、ＵＳＢインターフェース経由で設定可能とした。マイコンチップからの信号はロジック信号であるため、トランジスタによる駆動回路を介してＬＥＤを駆動している。

## ５．ＬＥＤ制御パターンの書き込み方法

前述の通りＬＥＤの制御方法は決定したが、これを実際にマイコンチップのメモリへ書き込むための具体的な方法が必要となる。考えられる方法として下記の２つがある。

(1) プログラム中の配列に制御パターンを直接書き込み、コンパイル後の実行形式ファイルを毎回マイコンチップへ書き込む。

(2) プログラムの書き込みやデバッグに用いているＵＳＢインターフェースを利用し、これを通してパソコンと通信を行うようなプログラムを作成し、パソコンから制御パターンを転送する。

(1)はいわゆる「リテラル値」を埋め込むものであり、最も単純な方法である。制御パターンの変更にはプログラムのソースコードの変更と再コンパイル、書き込みが必要となるが、開発ソフトが無償で提供されているので当該企業においても筆者と同様の開発環境を整えて実行することは可能である。しかし制御パターンを変更する度に上記の作業を繰り返す必要があり、時間がかかり過ぎる欠点がある。また、ソースコードの変更とコンパイルという操作にどうしても忌避感を捨てきれない方々も多く、歓迎される方法ではない。

他方(2)は、プログラム自身にＵＳＢインターフェース経由でパソコンと通信できる機能を持たせておき、プログラムが動作した後にパソコンから制御パターンを何らかのデータ形

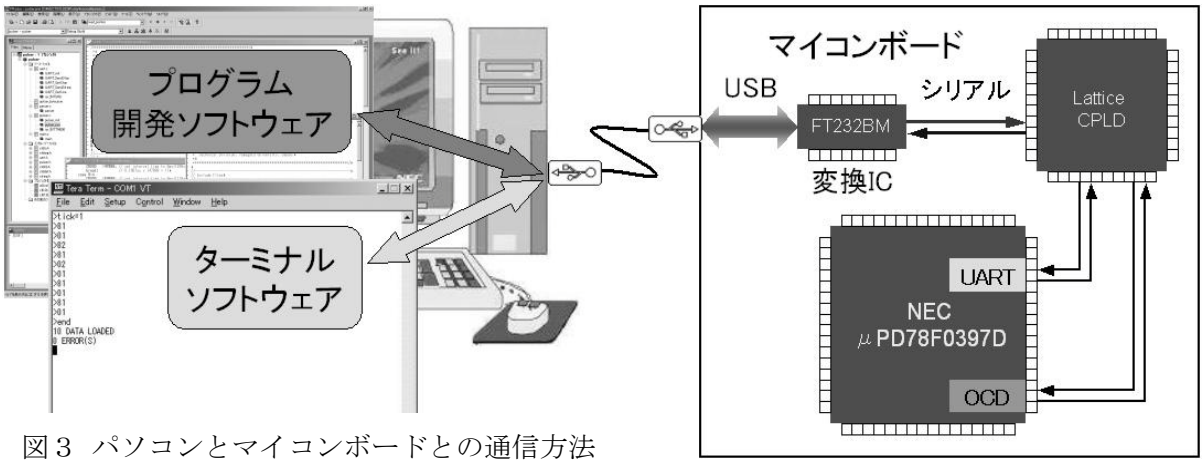


図３　パソコンとマイコンボードとの通信方法

式で送り込み、これをマイコンチップのメモリ上に展開し、次いでこれを読み取りながら然るべき信号を出力する方法である。この方法は(1)と比較して使いやすいが、通信機能のためのコーディングが必要となり、開発に手間と時間がかかる欠点がある。しかし利用者の利便性を最優先事項と判断して、(2)の方法を採用した。幸いにもこのＵＳＢインターフェースは、パソコン上のソフトウェアからはシリアルポート(仮想ＣＯＭポート)として見えるようになっている。図３は、パソコンとマイコンボードとの通信方法を説明した図である。今回用いたマイコンチップにはＵＳＢインターフェースは付いていない。そこでＵＳＢ－シリアル変換ＩＣ(FT232BM)を介してマイコンチップ上のＵＡＲＴ(調歩同期式シリアル通信)または書き込み回路とシリアル通信を行うように設計されている。よってこのＵＡＲＴインターフェースを用いてパソコン上で動作するターミナルソフトウェア(以下「ターミナルソフト」)と通信できるようにプログラミングを行えば、新たにパソコン上で動作するＵＳＢインターフェース専用のソフトウェアを開発する必要がない。次にこの仮想ＣＯＭポート経由で任意の制御パターンをマイコンチップに送り込む方法であるが、これにはテキストベースのコマンドを新規に定義し、ターミナルソフトから手打ちもしくはファイル転送によって送り込むこととした。また、ややトリッキーな方法ではあるが、デバッガが実行開始した後でもディップスイッチ(SW1-S3)の切り替えとデバッガリセットとを行うと、デバッガはプログラムを停止させることなく仮想ＣＯＭポートを開放してくれるので、ターミナルソフトがＵＡＲＴと通信可能となる。再びプログラムの修正を行う場合には、ターミナルソフトにＣＯＭポートを開放させてから、一連のビルド、デバッガの起動を行えばよい。

## ６．ＬＥＤ制御パターンの定義と転送

図４は、マイコンチップ上でプログラムを動作させた後に、ターミナルソフトを用いて制御パターンを転送している様子である。紙数の都合上詳細は説明できないが、１バイト

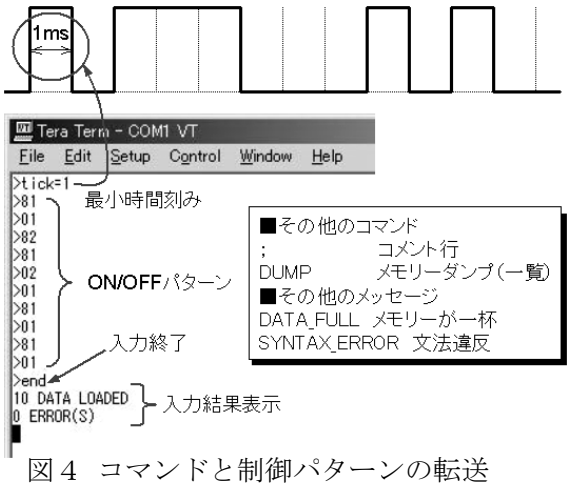


図４　コマンドと制御パターンの転送

の配列要素を１ステップとて、最上位ビットをＬＥＤの ON/OFF(1/0)とし、残りの 7 ビット(0～127)を ON/OFF の保持時間として、この１６進数表示の文字列を転送する。127 を超える長時間の保持が必要であれば、複数のステップを繋げればよい。手打ちでサンプルパターンを転送、動作させ、オシロスコープを用いて所定の制御パターンが出力されることを確認した。以上でマイコンチップ側の組み込みプログラムが完成した。

## ７．ＬＥＤ制御パターン生成プログラム

実際の使用においては、所望の制御パターンファイルを作成し、転送、動作させればよいが、実際の制御パターンは 400～600 ステップと長いために手作業で制御パターンファイルを作成することは現実的ではない。そこで図５のようなシーケンスを定義するファイル

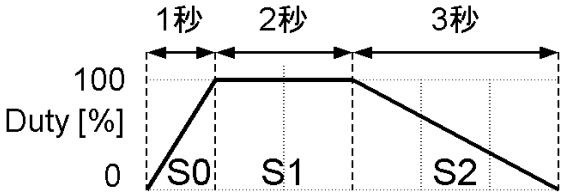


図５ シーケンスの例

```
RESOL:20
S0:1000,0-100
S1:2000,100-100
S2:3000,100-0
```

図６ シーケンスファイル

として、新たに図６に示すシーケンスファイルを定義した。例えば２行目は、S0 ステートは 1000ms かけて Duty factor を 0～100%まで変化させることを意味する。このシーケンスファイルを読み込んで図４に示したような制御パターンを出力する、パソコン上で動作するソフトウェアを作成した。

## ８．動作検証と問題の解決

動作検証の目的で、図７(a)のようなシーケンスに従ってＬＥＤを点灯させた。破線は Duty factor の変化を示している。その結果、

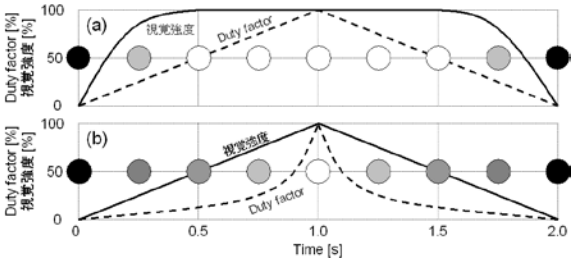


図７ Duty factor と発光強度の関係

線型に変化させた Duty factor に対して、実際の視覚強度(実線)は飽和したような変化を示した。実はこの原因は、人間の感覚が視覚も含めて物理的刺激量に対して対数関数的に反応することによる。従って視覚強度を直線的に変化させたい場合には、物理的刺激量を指数関数的に変化させればよい(図７(b))。前節のＬＥＤ制御パターン生成プログラムにこの効果を組み込むことによって、設計シーケンス(図５,図６)で期待される視覚変化が得られるようになり、本開発が完了した。

## ９．まとめ

(1) 本活動により当該企業の要望に応える組み込みソフトウェアを完成し、地域企業への技術支援を行うことが出来た。

(2) ＵＳＢ(仮想ＣＯＭポート)経由で動作パターンを転送できる組み込みソフトウェアを完成することがでた。

## １０．謝辞

一定の条件の基に本件の発表を承諾して下さいました当該企業様に心より感謝申し上げます。また、本活動への御理解を頂きました情報技術室の方々、廣瀬文彦教授、成田克助教、並びに研究室の学生諸君に感謝致します。

## １１．参考資料

1) 78K/LG2 用スターターキット(EZ-0003)
http://www.necel.com/micro/ja/development/asia/Evaluation_Board/Starter_Kit/ez-0003.html (2009 年 8 月 31 日現在)
(及び関連ドキュメント)