PIC16F648A による文字表示回路の製作

山﨑　勝翁[1]・手島　規博[2]

[1]技術センター，[2]制御情報工学科

マイクロコントローラPIC16F648Aを使用した，8×8ドットマトリクス表示器のダイナミック点灯制御方式による文字表示回路を製作した．言語には，マイクロチップテクノロジー社から提供されている統合開発環境MPLAB IDE付属のアセンブラMPASMを使用した．この文字表示回路により，ハードウェアの構造と動作を理解させ，ソフトウェアによりダイナミック点灯制御させるためのアルゴリズムを示した．また部品点数が少なく簡素化した回路とすることで，製作時間の短縮およびコストの削減に努めた．さらに，特別な動作環境を選ばず，単体で動作することにも配慮した．以上のことについて試作を試み，改良の余地があるものの動作することがわかったので，工学実験に導入した．

キーワード： PIC，アセンブラ，ダイナミック点灯制御，ドットマトリクス表示器

１．緒言

従来，制御情報工学科 3 年の工学実験で 74 シリーズ TTL-IC, ROM を使った文字表示回路の製作を行っていた．この文字表示回路は, 74HC14 による 2 段インバータ形 RC 発振回路，4017B デコーダ内蔵 10 出力 10 進カウンタ，HC541，HC540 のバッファ，インバータ，EPROM27C256 などで構成された回路である．しかし，回路を端子付き基板にハンダ付けするだけでは動作できない．5×7 ドットマトリクス 表示器を使った文字表示装置が別にあり，製作した端子付き基板を差し込むことで，文字表示回路が動作する．その文字表示装置がある場所以外では，回路が動作しないという制限があった．この回路は配線が多く，配線ミスやハンダ付け不良などで動作不良がおこり，時間外製作を余儀なくされる結果にもつながっていた．また，回路演習 I では，ライントレースカー製作の予備実験として，セラロックによる発振回路，フォトセンサ回路，DC 電源回路，PWM によるモータドライブ回路で，ハードウェア関連の基礎的な実験を行った．ソフトウェア関連では，MPASM のアセンブラプログラムを通じて，PIC16F84A を使用した文字表示を行った．こちらの文字表示は，プログラム作成後，PIC に書き込み，準備された文字表示装置で表示を確認するものである．どちらの文字表示回路にも制限があることがわかり，動作環境の変更が必要と思われた．平成 20 年度，工学実験内容の変更に伴い，文字表示回路に，8×8 ドットマトリクス表示器と PIC16F648A の組み合わせを考えた．　PIC16F648A は発振回路を内蔵しており，外部回路をほとんど必要とせず動作する．この PIC の 16 ピンは入出力に設定できるため，8×8 ドットマトリクス表示器を制御できる．今回の文字表示回路のコンセプトは，最少の部品で構成し，時間内に製作できるよう簡素化した回路としたことである．また論理が変わらぬよう論理値 1 が点灯，論理値 0 が消灯とし，プログラムで好みの文字表示制御ができるものとした．そのうえ動作環境を選ばず，コストを最小限に押さえたものをと考え，工学実験に導入したので，その方法について報告する．

２．方法

開発環境は Windows2000,XP，MPLAB IDE，PIC ライタである．詳細は(1)ソフトウェア，(2)ハードウェアに示す．**図-1** が製作した文字表示回路である．パーツが少なく最少の部品で構成されている．基板にはドットマトリクス表示器，　PIC16F648A，抵抗ネットワーク，IC ソケット以外のパーツはなく，シンプルな回路である．製作は 3h×4 週

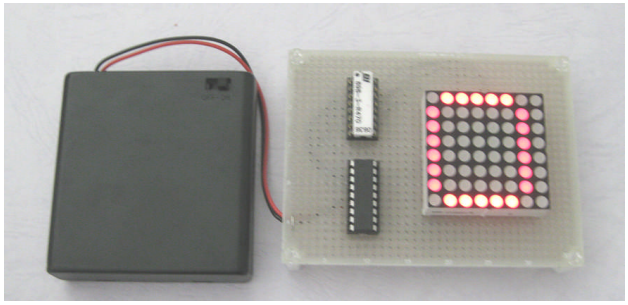

図-1　文字表示回路

で行い，2週がハードウェアの製作，2週がアセンブラプログラムの作成とした．

(1) ソフトウェア

使用したソフトウェアを開発環境として**表-1** に示す．MPLAB IDE Ver8 は Microchip 社ホームページから無償でダウンロードでき使用できる．IDE(Integrated Development Environment)とは統合環境を指し，エディタによるソースファイルの作成からアセンブル(コンパイル)まで行い，オブジェクトやアセンブルリストが生成できるものである．PIC ライタは秋月電子通商の Aki-PIC プログラマーキット Ver.4 を使用した．このキットには書き込みソフト PIC ProgrammerV4 Beta が付属している．

(2) ハードウェア

使用した主なパーツを**表-2** 示す．

a) PIC16F648A

図-2 に PIC16F648A（PDIP）のピン配置[1)]を示す．PDIP タイプを使用した．（PDIP :Plastic Dual inline Package）この PIC は 8 ビットのワンチップマイコンである．CPU, ROM, RAM，I/O ポートなどが 1 つのチップに収められ，プログラムにより異なる動作をさせることができる．命令がわずかに 35 個[2)]であり，バイト処理命令（18），ビット処理命令（4），リテラル処理命令（6），ジャンプ命令（5），その他（2）など 5 つの命令に分類される．() 内は命令の個数を表す．動作電圧 2.0V～5.5V の範囲で動作する．I/O ポートは I/O ピン RA0～RA7 がポート A，RB0～RB7 がポート B となる．I/O ピンはシンク電流 20mA, ソース電流 25mA まで流すことができる．シンク電流とは外部回路から I/O ピンに取り込める電流をいい，ソース電流とは I/O ピンから外部回路へ取り出せる電流をいう．内部発振器の発振周波数約 4MHz で動作させた場合，各命令の実行時間は約 1μsec となる．デバイス名に F が付いているフラッシュタイプは，何度でも電気的に書き換えて使うことができる．一般的には, USB フラッシュメモリと同じタイプである．

b) ドットマトリクス表示器

図-3 に 8×8 ドットマトリクス表示器[3)]を示す．内部構造は LED が 64 個配置されている．LED のアノード側が共通の線で接続されているアノード・コモン型である．Microchip 社のデータシートより，PIC16F648A の 4 ピン RA5 が Input port 専用ということがわかった．よって製作した回路では 8×8 ドットの点灯が無理なことがわかり，8×7 ドットの 56 個の LED の点灯制御に変更した．

c) 抵抗ネットワーク(8 素子独立)

470Ω の抵抗が 8 個組み込まれた 8 素子独立の抵抗アレイである．通常使用する炭素皮膜抵抗は使用していない．その理由は，このように形状の違う IC パケージ型抵抗もあることを認識させることと，簡素化することで見栄えを良くすることである．

表-1　開発環境

	Software	Version
統合環境	MP LAB　IDE	Ver8
PIC ライタ	Aki PIC PIC Programmer v 4	V6.61.48

表-2　パーツリスト

品　名	規　格
PIC マイコン	PIC16F648A-I/P
ドットマトリクス表示器	パラライト社製 TOM-1588BH-B
抵抗ネットワーク	8 素子独立(470Ω×8)
IC ソケット	18P，16P
電池 BOX	単三×4　スイッチ付き
ユニバーサル基板	95mm×72mm
スペーサー	3mm プラネジ+14mm（4 本）

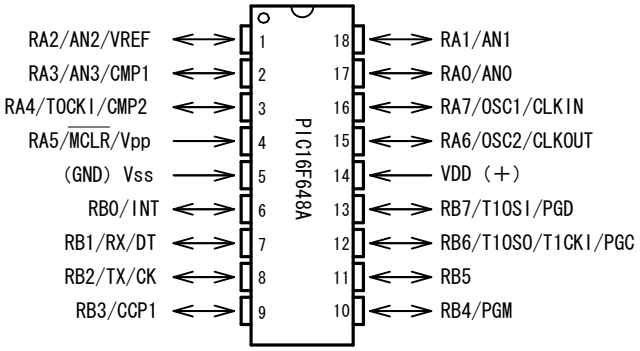

図-2　PIC16F648A（PDIP）のピン配置

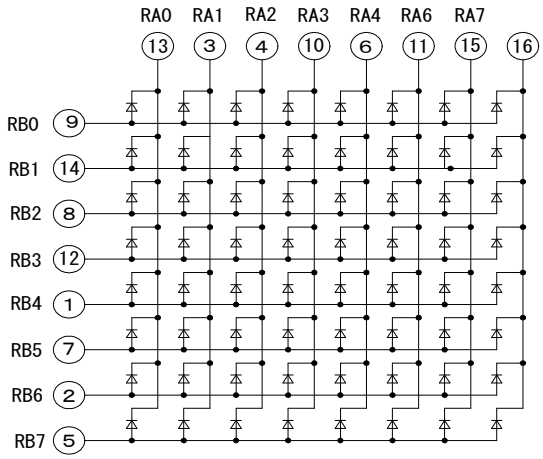

図-3　8×8 ドットマトリクス表示器

d) IC ソケット

PIC 用に 18 ピン，抵抗ネットワーク用に 16 ピンの IC ソケットを使用した．IC を基板に直接ハンダ付けすると熱により破損する場合があるため，IC を保護する役割がある．また IC の差し替えが可能となる．

e) 電池 BOX

単三×4 本のスイッチ付きのものを使用した．電源電圧約 DC6V として使用した. スイッチはメインスイッチとして代用した.

f) ユニバーサル基板

95mm×72mm のものを使用した.

g) スペーサー

樹脂 3mm プラネジ（7mm）+6 角スペーサ（14mm）を 4 セット使用した.

(3) 文字表示回路の設計

はじめに，LED1 個の点灯回路の原理を理解させ，次にマトリクス LED とするため，**図-4** の 2×2 ドット LED の点灯回路を考えた．これが理解できると発展させた 8×8 ドット LED の点灯制御ができる. PIC の出力ポートと 8×8 ドットマトリクス表示器の関連付けは, 表計算ソフトにならい行と列を用いた．各 LED を行と列で示すと分かりやすいので，左上の角の LED が 0 行 0 列と設定した．ポート A の RA0～RA7 を 0 列～6 列へ, ポート B の RB0～RB7 を 0 行～7 行へと対応させた．またポート A，ポート B の I/O ピン番号を bit0～bit7 と対応させ数字を揃えた．RA0 が bit0，RA7 が bit7 である．残念ながら RA5 が出力として使用できないため 5 列目を RB6, 6 列目を RB7 として使用した．そのため**図-3** の右端の列の LED8 個はこの文字表示回路では使わないことにした. RA5 の bit5 の設定は，点けない設定の 1 とした．データの表記は B’00000011’のように，2 進数で示すと点灯が 1，消灯が 0 を表すことになり，理解しやすい．**リスト-1** に 2×2 ドット LED の全点灯プログラムの例を示す. ①MOVLW でデータをセットし，②MOVWF でポート A に出力する．ポート B も同様に③④の操作を行う．⑤が表示時間となる．**図-5** が製作した文字表示回路の回路図である.

３．製作手順

(1) ブレッドボード上で動作確認

確実に製作するためハンダ付けを行う前に, ブレッドボードを利用した．ブレッドボードにパーツを配置し，PIC のピン番号と8×8ドットマトリクス表示器のピン番号の接続を確認しながら，文字表示回路を製作した．その後，動作確認済みプログラムを書き込んだPICを取り付け，文字表示の確認を行った. ブレッドボードは, パーツを配置し，ジャンプワイヤーを接続して回路を完成させるもので, 一時的に回路の動作を確認する手段である.

(2) ハンダ付け

ハンダ付け作業の工程は，まずポートA周辺から行った. 接続の確認後，問題がなければ次にポートB周辺のハンダ付けを行った．ブレッドボードでは，表側から見たTOP VIEWの配線で行う．ハンダ付けでは，裏側から見たBOTTOM VIEWの配線で行う. このことを理解しておかなければならない．間違えると全く逆の配線となる.

リスト-1　2×2 ドットの全点灯プログラム

```
MOVLW     B’11111100’     ;①
MOVWF     PORTA           ;②
MOVLW     B’00000011’     ;③
MOVWF     PORTB           ;④
CALL      TIMER           ;⑤
```

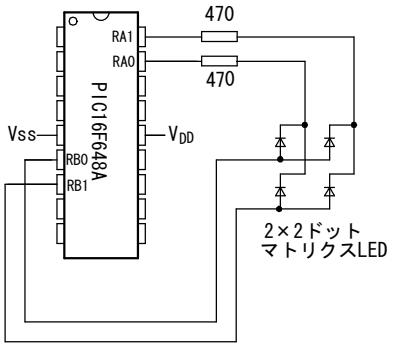

図-4　2×2 ドット LED の点灯回路

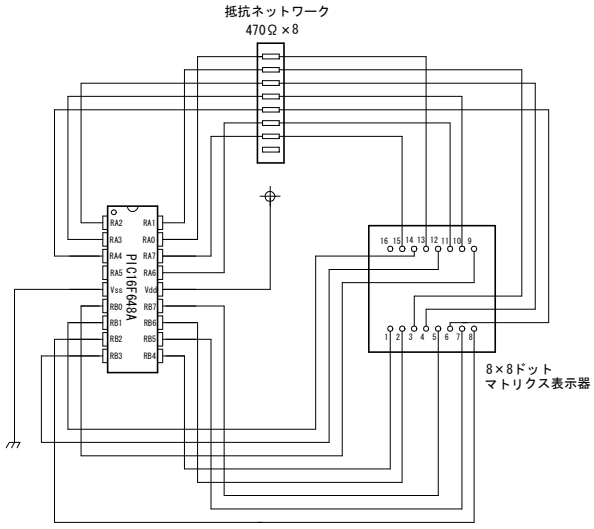

図-5　文字表示回路図（TOP VIEW）

(3) プログラム

今回サンプルプログラムとして示したのは，1 秒間隔で「ONCT」と 4 文字表示するプログラムである．

(4) 動作確認

実行中の文字表示の様子を**図-6** に示す．文字の明るさが一定でないのは，電流不足のためである．この対策は**5. 考察**に示す．

4．文字表示回路

(1) ダイナミック点灯制御

点灯制御には，スタティック点灯制御とダイナミック点灯制御[4]がある．**図-7 (a) (b)** は電源，スイッチ，抵抗，LED 1個により，スタティック表示とダイナミック表示の違いを簡単に示したものである．スタティック表示方式は**図-7 (a)** のSWをONの状態とし，LEDを常時点灯させる方法である．ダイナミック点灯制御方式は**図-7 (b)** のように，スイッチのON/OFF切り換えによりLEDの点灯を制御している．製作した文字表示回路では，列を左から右へ高速に順次切り換えて，点灯させている．実はLEDは点いたり消えたりして点滅している．人間の目には残像現象があり，1msecともなると判断ができないため，常時点灯しているように見える．

a) 基本回路

(1)に示したようにダイナミック表示は高速にスイッチの ON と OFF を切り換えて表示する方法で，LED は点滅した状態である．手動で切り換えていたのでは，点滅しているようにしか見えないので，電気的に切り換える．電気的に切り換えるため 0 と 1 または 0V と＋5V がある周期で切り替わるパルスと呼ばれる信号を用いる．555 などのタイマ IC や発振回路で取り出すことができる．このようなパルスを SW の替わりとして接続すると，電気的にスイッチの ON と OFF を切り換えることになる．この接続の場合，電源は不要となる．数 10msec 以下の速さで繰り返すと連続して点灯しているように見える．

b) タイムチャート

図-8 (a) にスタティック表示，**図-8 (b)** にダイナミック表示のタイムチャートを示す．横軸に時間，縦軸に電圧または ON/OFF を表したもので，タイムチャートと呼ばれる．縦軸が電圧の場合，面積が平均電圧を表すことになり，**図-8 (b)** ダイナミック表示は半分の面積，すなわち平均電圧は 1/2 となる．時間進行は左から右となる．タイムチャートで，時間進行に伴う動作の状態がわかる．**図-9** は ON 時間と OFF 時間の比率を%か数値で示したもので，デューティー比と呼ばれる．繰り返しサイクル 1 周期が 10msec で，ON 時間 5msec，OFF 時間 5msec の場合，デューティー比は 50%または 0.5 となる．ON 時間 1msec，OFF 時間 9msec の場合，デューティー比は 10%または 0.1 である．

モータの回転数を制御する場合にも使用されることが多く，この比率を変更するとモータの回転数が変わる．

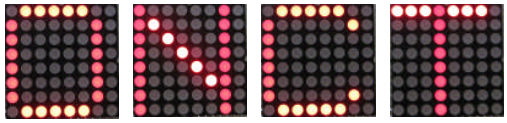

図-6　「ONCT」の文字表示

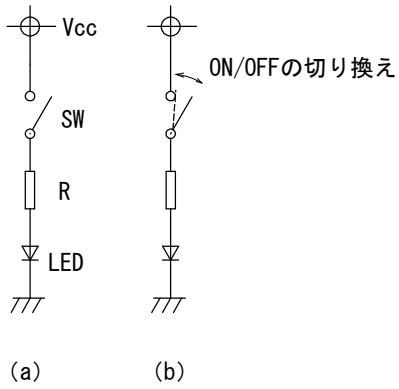

図-7　スタティック表示とダイナミック表示

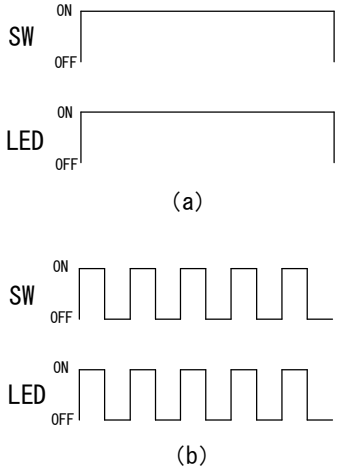

図-8　タイムチャート

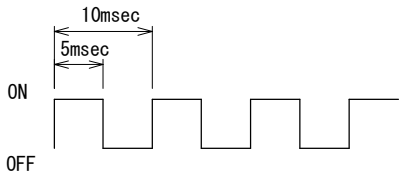

図-9　デューティー比

５．アセンブラプログラム

(1) MPASM

統合環境 MPLAB のエディタを使う場合，デフォルトのフォントサイズが 8 ポイントと小さいため，プロパティ設定でサイズを大きくすると見やすくなる．エディタがアクティブの状態で，メニューの Edit から Properties を選択すると EditorOption ダイアログボックスが開く．Text タブの SelectFont から好みのフォント名，スタイル，サイズを決定する．

(2) 文字表示

プログラム作成の前に**図-10** に示す 7 枚の GIF 画像を使ったアニメーション GIF を作成した．ダイナミック点灯制御のイメージを膨らませるため，ドットマトリクス表示器の左から右へ点灯パターンが 0.1sec 間隔で移動するものを示したものである．ポート A はカソード側に接続されているため，点灯させる列のピンのみ論理値 0 とすることで 1 列の LED8 個を点灯させる準備が整う． ポート B はアノード側に接続されているので，点灯させる行のピンを論理値 1 とする．カソードが論理値 0，アノードが論理値 1 の場合 LED が点灯する．**図-11** に文字「O」のタイムチャートを示す．このタイムチャートから 1 文字を 1 回表示させるには，7msec の時間がかかる．1 秒間（1000msec）表示させるには式(1)のとおり約 143 回繰り返せばよいことがわかる．16 進数では 8FH となる．

$$\frac{1000\text{msec}}{7\text{msec}} = 142.85 \fallingdotseq 143 \qquad (1)$$

図-12 は 1 文字を表示させるフローチャートである．X=6 の判定は X を列番号としたもので， 0～6 列のポート B の文字表示データを 1msec 出力することを表している．RA5 は除く．実際は**リスト-2** のような順次構造処理を 7 回させている．CNT1-1=0 の判定は，143 回（8FH）を判定するループである．このループを抜けた時，2 文字目の表示に移る．**リスト-2** の内容は，①RA0 を論理値 0 とし，0 列目が点灯可能の状態にする．②ポート A へ出力．③0 列目の点灯データをセットする．④ポート B へ出力する．⑤1msec タイマサブルーチン TIME1 の呼び出しである．

リスト-2　0～6 行の 0 列目を 1msec 点灯するプログラム

```
MOVLW    B'11111110'    ;①
MOVWF    PORTA          ;②
MOVLW    B'01111110'    ;③
MOVWF    PORTB          ;④
CALL     TIME1          ;⑤
```

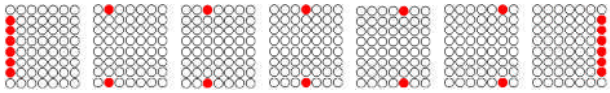

図-10　ダイナミック点灯制御のイメージ（文字「0」）

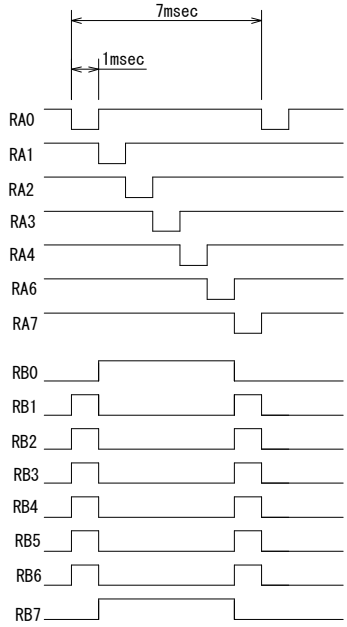

図-11　ダイナミック点灯制御によるタイムチャート

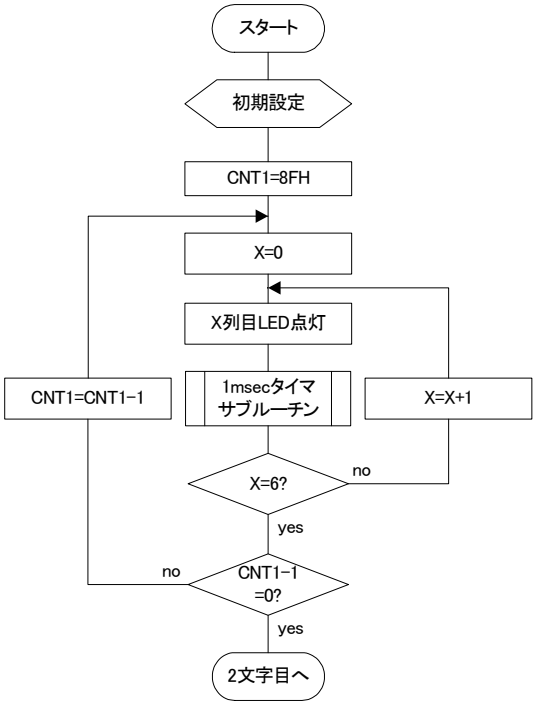

図-12　1 文字を表示させるフローチャート

(3) 1msec タイマ

すべての命令は実行するのに一定の時間がかかり，それを繰り返すことで遅延時間[5]を作る．したがって命令の実行時間がわかれば，時間を正確に決定することができる．命令の 1 サイクルの実行時間は式(2)のようになる．使用した PIC16F648A の内部クロック 4MHz の場合，式(3)の結果が得られ，1 サイクル 1μsec の実行時間を必要とする．

$$1\text{サイクル実行時間}=\frac{4}{\text{クロック発振子の周波数}} \tag{2}$$

$$1\text{サイクル実行時間}=\frac{4}{4MHz}=1\times10^{-6}=1\mu\,\text{sec} \tag{3}$$

$$\frac{1\text{msec}}{1\mu\,\text{sec}}=1\times10^{3}=1000\text{サイクル} \tag{4}$$

1 サイクル 1μsec の実行時間がかかることを考慮し，遅延時間 1msec を作るには，式(4)のように 1000 サイクルが必要となる．よって 1msec タイマのサイクル合計が 1000 サイクルとなるようルーチンを繰り返せばよいことになる．PIC16F648A のレジスタは 8 ビットのため，変数に設定できる最大値は 255 までとなる．1000 サイクルの内訳は，4 サイクル×250，5 サイクル×200，10 サイクル×100 などが考えられるが，250×4 サイクルが最も適切なため採用した．ただしサブルーチン前後の CALL, RETURN 命令のサイクルを考えて-1 少ない 249 サイクルと設定するのがよい．**リスト-3** に 1msec タイマサブルーチンを示す．**図-13** は 1msec タイマのフローチャートである．TIME1 を CALL することで，実行する．CNT2-1=0 の判定は，249 回(F9H)を判定するループであり．RETURN で戻る．**リスト-3** の内容は，①1msec タイマサブルーチンをラベル名 TIME1 と定義し，定数データ F9H(10 進数の 249)をワーキングレジスタにセットする．②カウンタ用変数 CNT2 へ定数データを格納する．③から繰り返しルーチンで，ラベル名を TIME1LP と定義した．NOP は何もしない 1 サイクル命令であるが，実行させると 1μsec の時間がかかる．また繰り返しサイクルが NOP を 1 つ記述することで計 4 サイクル(③+④+⑤)となり，4 サイクル×250 の設定が適用できる．

リスト-3　1msec タイマサブルーチン

```
;1msec  Timer  Subroutine
 TIME1      MOVLW   F9H          ;①
            MOVWF   CNT2         ;②
 TIME1LP    NOP                  ;③
            DECFSZ  CNT2,1       ;④
            GOTO    TIME1LP      ;⑤
            RETURN               ;⑥
```

NOP はサイクルの微調整の時に使うことが多い．④減算命令 DECFSZ は 1 サイクル命令で CNT2 の値を-1 し，結果が 0 なら次の命令をスキップする．ただしスキップする時は，次の命令を NOP に変えて実行するため，2 サイクル命令 GOTO を 1 サイクル命令 NOP に変えて実行する．そのためこの時は 1 サイクルとなり全体から-1 サイクルする処置が必要となる．ここで減算命令 DECFSZ は 249 を-1 し，結果が 0 でなければ次の 2 サイクル命令 GOTO TIME1LP へジャンプし減算を繰り返す．249 回目には結果が 0 となり，GOTO　TIME1LP をスキップし，⑦2 サイクル命令 RETURN が実行されサブルーチンから戻る．1msec タイマサブルーチンは 2 サイクル命令 CALL TIME1 でジャンプしている．①+②で 2 サイクル，③＋④＋⑤で 4 サイクルを 249 回繰り返す．1msec タイマサブルーチンのサイクル数は，式(5)のようになり，式(6)により約 1msec の遅延時間とした．ちなみに 5 サイクル×200 でも式(7)(8)のとおり同じ遅延時間が得られた．どちらを使っても遅延時間は同じであるが，使用したのは NOP が 1 個の 4 サイクル×249 である．　式(6)(7)の 996+2 の+2 サイクルは CALL 命令である．

4 サイクル×250

$$2+4\times249-1=996 \tag{5}$$

$$\begin{aligned}996+2&=998\times1\mu\,\text{sec}\\&=0.998m\,\text{sec}\\&\fallingdotseq1\text{msec}\end{aligned} \tag{6}$$

5 サイクル×200

$$2+5\times199-1=996 \tag{7}$$

$$\begin{aligned}996+2&=998\times1\mu\,\text{sec}\\&=0.998m\,\text{sec}\\&\fallingdotseq1\text{msec}\end{aligned} \tag{8}$$

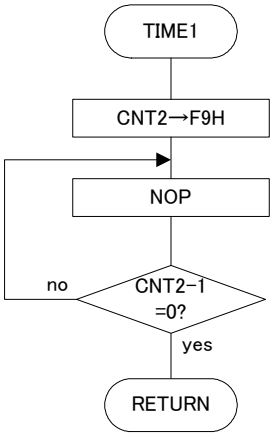

図-13　1msec タイマフローチャート

1msec タイマを利用して 0.1sec タイマ（1msec タイマ×100），0.5sec タイマ（0.1sec タイマ×5），1sec タイマ（0.1sec タイマ×10）など発展させ，用途によって必要な遅延時間を作ることができる．外部振動子を使った場合も，クロック発振子の周波数から 1 サイクルの実行時間が計算できるので，同様に遅延時間を作ることができる．ここで，アセンブラを使う理由として，プログラムの実行時間があらかじめ計算でき，μsec 単位の高速な処理が必要なプログラムには有利という点が挙げられる．

(4) コンフィグレーション・ビットの設定

18 ピンの PIC16F648A には 1 ピンに最大 4 つの機能が割り振られており，各機能を設定で使い分けることができる．この文字表示回路では各ピンを I/O ピンとして使うため，I/O ピン以外の機能はプログラムの冒頭で使わない設定 [6] とした．またリセット時にデフォルトで I/O ピンとして使える設定の場合もあり，I/O ピンとして意識せずに使うことができる．電源(V_{DD}，V_{SS})以外のピンを I/O ピンとして使う場合の設定を**表-3** に示す．**リスト-4** に示す記述でコンパレータが OFF となる．

(5) 疑似命令について

コンフィグレーション・ビットの設定は，プログラムで疑似命令「__CONFIG」に続き，パラメータを記述する．「CONFIG」の前はスペース 1 個以上とアンダーバーが 2 個である．行の先頭から 1 個以上のスペースを入れなければラベルと認識するので注意する．各パラメータの区切りはスペースであり，「&」を入れて続けてパラメータを記述する．各パラメータの先頭はアンダーバーが 1 個である．これらの記述を間違えるとエラーになる．**リスト-5** に設定した例を示す．**表-4** にコンフィグレーション・ビットのパラメータを示す．

(6) リセット

PIC マイコンには，数種類のリセット回路が内蔵されている．内部リセット回路と外部リセット回路 [7] に分類される．リセットが必要な理由は，電源電圧 V_{DD} が正常範囲内かの確認，暴走する可能性がある場合，電源 OFF の時に，格納したデータを書き換えてしまう場合があるなどの理由で，暴走を防ぐことができる．

a) パワーアップタイマ

パワー・オン・リセット後に起動し，約 72msec 間だけリセットする．このタイマはデフォルトで使う場合が 0，使わない場合が 1 となっており，OFF とすると使う設定となるので，ON と記述する．

表-3　I/O ピンとして使う場合の設定

ピン	内容
RA0 RA1 RA3 RA4	CMCON レジスタの CMx ビット CM0～CM2 をすべて論理値 1 とすると，コンパレータは電源 OFF となり，動作しない．この設定により RA0，RA1，RA3，RA4 が I/O ピンとなる
RA2	リセット時に無効になっているので，I/O ピンとなる
RA5	マスタークリア　外部リセットを使わない OFF 設定にすると RA5 が I/O ピンとなる
RA6 RA7	内部振動子を使い，外部振動子を使わない．また内部クロックを外部へ出さない設定とすると I/O ピンとなる
RB0	外部割り込みリセット時に I/O ピンとなっている
RB1 RB2	RS-232C 送受信機能であるが，リセット時に I/O ピンとなっている
RB3	リセット時に I/O ピンとなっている
RB4	Low Voltage Programming．PIC を基板に取り付けたままプログラムの書き込みを行う機能である．この機能は OFF 設定で I/O ピンとなる
RB5	Input port．入力ピンとしてのみ使用できる．出力ピンとして使うことはできない
RB6 RB7	リセット時に I/O ピンとなっている

リスト-4　コンパレータの OFF

```
MOVLW   07H
MOVWF   CMCON
```

リスト-5　コンフィグレーション・ビットの設定

```
__CONFIG _INTRC_OSC_NOCLKOUT &
_CP_OFF & _WDT_OFF & _PWRTE_ON &
_BODEN_OFF & _LVP_OFF &
_MCLRE_OFF
```

表-4　コンフィグレーション・ビット

疑似命令・パラメータ	内容
__CONFIG	コンフィグレーション・ビットの設定指定
_INTRC_OSC_NOCLKOUT	内部の RC 発振器を使い，内部のクロックパルスは取り出さない
_CP_OFF	コード・プロテクト
_WDT_OFF	ウォッチドッグ・タイマ
_PWRTE_ON	パワーアップタイマ
_BODEN_OFF	ブラウンアウトタイマ
_LVP_OFF	ローボルテージプログラミング
_MCLRE_OFF	マスタークリア 外部リセット入力

b）ブラウンアウトタイマ

ブラウンアウト・リセット電圧は約 2V になっている．電源電圧 V_{DD} が下がった場合や，接触不良などで電源電圧が ON/OFF を繰り返す場合などに約 100μsec 以上でリセットされる．

c）コード・プロテクト

プログラム・メモリをコード・プロテクトするかどうかを選択する．プログラム・メモリの内容が 0 としか読み出せなくなる．しかしコンフィグレーション・ビットなど一部プロテクトされない領域もある．

d）ウォッチドッグ・タイマ

タイムアウトすると PIC マイコンをリセットする．

e）ローボルテージプログラミング

PIC を基板に取り付けたままプログラムの書き込みを行う機能である．組み込みで使う場合に使用する．

f）マスタークリア

外部リセット入力回路を使うかどうかを選択する．このリセットのみ外部リセット回路である．

(7) 内部発振器 4MHz のキャリブレーション

内部発振器を使う場合，精度は 4MHz±1%である．プログラムのはじめにキャリブレーション・データを OSCAL レジスタに記述しておく．これで内部発振器の発振周波数が 4MHz に近くなる．Microchip 社の PIC16F648A のプログラム・メモリにキャリブレーション．データが書き込まれている．周波数精度をもっと良くしたい場合，EC モード，HS モード，XT モード，LP モードのどれかを選択し，外付けの発振子・発振器を使う．

５．考察

文字表示回路の製作について，改善が必要な問題点を以下に考察する．

(1) 入出力ポート

手頃な価格の PIC16F648A を使用したため，入出力ポートが不足した．仕様を満たす入出力ポートの多い PIC を使用する．

(2) 電流不足の対策

今回の回路では，シンク電流 20mA のため 1 列が全点灯した場合，20mA の 1/8 の電流で LED をドライブすることになり，やや薄暗くなる．対策として，トランジスタまたはトランジスタアレイを使い増幅することが考えられる．しかし，ハンダ付け箇所が増え，ミス・接触不良による製作時間の増加が考えられ，コンセプトから外れる．またコスト増にもつながり改良の余地がある．

(3) 電源電圧対策

PIC16F***の電源電圧仕様では対応範囲が広く，DC2.0～5.5V で動作するため，単三×4 本の電池を直列接続した約 6V の電圧を使用した．しかし，使用した電池・PIC の個体差により，正常動作するものや動作不良の症状もみられた．未使用の電池では過電圧の状態となり，1 文字表示したところで暴走しフリーズした．そのため 2 文字目の表示に進まなかった．この時の電圧は 5.82V である．同じ環境で使用済み電池と交換すると，正常動作した．また電池は交換せずに PIC を交換すると，正常動作した．しかし不安定なため，三端子レギュレータやツェナーダイオードを用い，定電圧を供給する回路を追加する必要がある．

(4) 開発環境

MPLAB の Active Toolsuite から選択できる，HITEC C コンパイラ，CCS コンパイラなどサードパーティから提供されているコンパイラがある．また統合環境の MikroBasic，MikroC も無料で提供されており，今後，その他の開発環境において動作検証を行う．

６．結言

PIC16F648Aによる文字表示回路の製作において，得られた結果をまとめると次のようになる．

(1) この文字表示回路の製作により，学生にハードウェアの構造と文字表示回路の動作原理を理解させた．

(2) MPLAB IDE および PIC ライタの開発環境で，PIC を使用した回路のプログラムが作成できる．

(3) ハードウェアを意識した基礎的なプログラミングの 1 つの方法として，アセンブラは有効である．

(4) 最少の部品で構成し，簡素化した回路としたので，学生が時間に余裕を持って回路の製作とプログラムの作成ができることが確認できた．

(5) 従来の文字表示回路と比較して，コストを約 1/3 に抑えることが可能である．

参考文献

1) PIC16F627A/628A/648A　Data Sheet MICROCHIP

2)たのしくできる PIC 電子工作　後閑哲也　東京電機大学出版局

3) TOM-1588BH-B データシート　パラライト社

4)ディジタル情報回路の基礎　宮本義博　技術評論社

5)電子工作のための PIC 活用ガイドブック　後閑哲也　技術評論社

6)C言語ではじめるPICマイコン　－フリーのCコンパイラではじめよう－　中尾　真治　オーム社／出版局

7)PIC マイコン活用ハンドブック　トランジスタ技術編集部　編　CQ 出版社

（2008.9.26受付）