R8Cマイコン ASCII文字入力で、モールス符号出力

モールス符号とは、電信で用いられている可変長符号化された 文字コードです。
モールス符号を使った信号は モールス信号と呼ばれます。（ Wikipedia 参照： ちょっと堅苦しい表現が続きますが、我慢して下さい。）

概要：
日本では、総務省令無線局運用規則別表第1号に和文と欧文の符号が定められ、総合無線通信士は、無線従事者国家試験において和文および欧文の送受信の、国内電信級陸上特殊無線技士は、国家試験および養成課程修了試験において、和文の送受信の電気通信術の実技試験があり、また第一級・第二級・第三級アマチュア無線技士では、試験および 修了試験の法規において、モールス符号に関する知識が問われる。

日本語では、モールス符号の短点を「トン」（あるいは「ト」）、長点を「ツー」と 表現することが多いため、俗に「トンツー」とも呼ばれる。
短点と長点の組み合わせだけで構成されている単純な符号であることから、修得者は無線通信に限らず音響や発光信号でも会話や通信に活用している（ 投光・遮光が 一挙動で自由に出来て信号を送れる 回光通信機を持つ大型船舶が存在する ）。

歴史：
アメリカ合衆国の発明家サミュエル・フィンレイ・ブリース・モールスは、1837年9月4日にニューヨーク大学で、まずは現在のものと全く異なった符号で 電信実験を行った。 次にジョセフ・ヘンリー（ プリンストン大学教授 ）の指導と 協力の下で 改良した符号と 電信機に関する特許を1840年6月20日に 取得した。

さらに改良した符号を使って　1844年5月24日には デモンストレーションを行い、ワシントン（ の B&O Mount Clare Station ）から　ボルチモアへ向けて “What Hath God Wrought”と送信することに成功した。

1849年に　フリードリヒ・クレメンズ・ゲールケが改良した符号を もとにしたものが、DOTV（ドイツ語: Deutsch-Osterreichischer Telegraphen-Verein ）の　1851年10月ウィーン会議において　標準規格とする条約が結ばれた。 その後、1868年7月にウィーンで開催されたUTI（ フランス語: Union Telegraphique Internationale、万国電信連合　ITUの前身の一つ ）において国際規格として承認され、現在のものの原型となった。

船舶の無線室に備えられていた時計。
毎時0・15・30・45分から3分間の間に色が塗られているが、
これは聴守態勢をとらねばならない時間帯（沈黙時間）を表している。
画像はアメリカ製の物。日本製の物は緑の部分が青になっている。 陸上同士の通信においては、20世紀前半まで電報などの文字通信で多く使われた。

1920年代あたりからテレタイプ端末による電信・1930年代からテレックス・　1980年代からファクシミリ・1990年代後半から電子メールなど、他のデジタル通信方式の発達により、次第に使われなくなった。

一方、遠洋航海の船舶間、または船舶と陸上との通信においては、通常の通信から万一の際の　遭難信号（SOS）まで、長い間中波および短波を使ったモールス通信が行われ、映画などで船舶内の無線室でモールス通信を行うシーンも良く出ていた。

通信衛星の登場によって 短波によるモールス通信は縮小し、非常用の通信手段としても国際海事機関(IMO)の決定により、国際的な船舶安全通信がGMDSSに1999年2月に完全に移行したため、モールス通信は 基本的に使われなくなった。

日本では、1996年に海上保安庁がまた1999年までにNTTグループやKDD（現KDDI）もモールス符号を用いた通信業務を停止した。残るのは、一部の漁業無線（遠洋漁業）・自衛隊の一部の通信・アマチュア無線である。

以上のように　双方向の通信に用いられることは 稀になったが、同報通信における識別信号の送信にはいまだに利用される。

航空無線航行用の　DME、ILS、VOR、NDB（ 無線局の種別は 無線航行陸上局又は 無線標識局 ）は モールス符号により　標識符号を送信するものと、短波を用いて海洋観測をする海洋レーダー（ 無線標定陸上局 ）は　モールス符号により　呼出符号を 送信するものとされる。

JJY（ 標準周波数局 ）も　呼出符号の送信はモールス符号による。 実験試験局でも電気通信大学の HFD観測用実験試験局 JG2XAなどがある。 電気通信術の訓練は、陸上自衛隊通信学校や　海上自衛隊第1術科学校、水産高等学校で行われている。

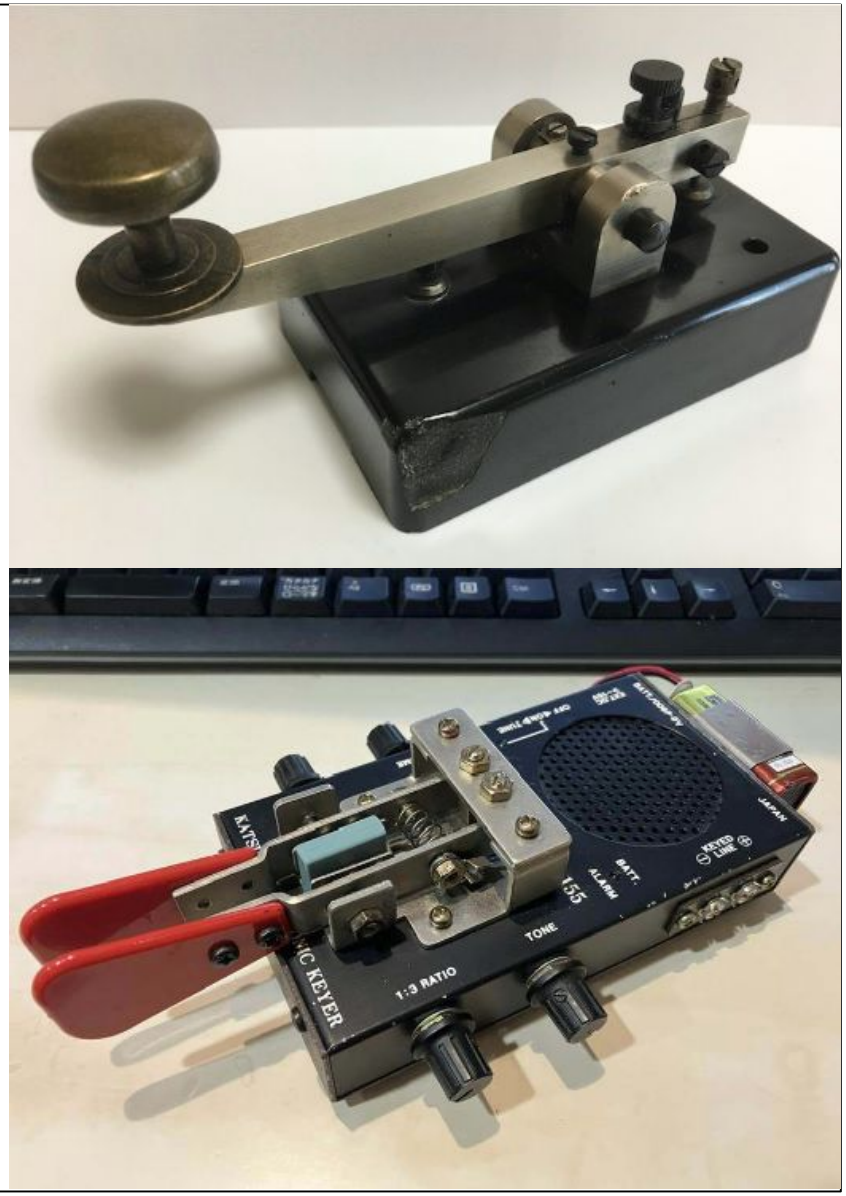
初期の送受信機

モールスの送信機は、機械式スイッチ（電鍵）の接点を手動で 開閉するものであった。 紙テープを事前に穿孔してそれにより接点を開閉する方式の自動送信機を 1846年にベインが発明した。 1866年からイギリスのチャールズ・ホイートストンが 製作した自動送信機が 広く使われた。

受信機としては、1837年にトミーが発明した、紙テープに電磁石で動かした針の圧力で刻むエンボッシング方式が最初に使われたが、紙の巻き取りなどで 鮮明でなくなり判読に 苦労するものであった。1854年に トーマス・ジョンがインクで 印を付ける方式を 考案した。

また1860年代には、紙テープを動かして固定したペンに接触させたり離したりする方式に 改良された。

ちなみに、右の画像は アマチュア無線で使用する用途の 標準的な手動式の電鍵（上）で、俗称：コメツキバッタとか 名前が 付いていたと思います。 下は、赤いパドルを左右に動かして使うもので、エレキーパドルとか名前が付いています。 中に電子回路が、入っていて、短点、長点の長さを 正確に出せる半自動の電鍵だったと思います。



この印字機を用いてモールス符号を視覚化しそれを文字に直す方法は、通信量が多くなると対応が難しくなる。機械式継電器（音響器）の音で符号を判別する音響受信は最初禁止されていたが、同時筆記が可能で高速通信が行えるので、後には広く行われるようになった。

有線と無線の通信方法　20世紀初頭に、電波を断続してモールス符号を送受する無線電信が実用化された。 有線電信と比較すると、送信のための 電鍵操作は 基本的に同一であるが、受信の方法は 両者で 異なる。

有線電信では、音響器を用いた聴覚による受信方法が基本である。電流が流れ始めた時と断たれた時に衝撃音が発せられるので、これの音調と 間隔により 短点と 長点を判別する。

無線電信においても（最初期以外は）聴覚受信が　行われてきたが、短点と長点は　持続音で表現され有線電信の　カタカタ音とは異なる。

そのため有線と無線の通信士では　訓練課程も異なることが多く、どちらか片方の操作だけに従事するのが普通だったが、有線モールスの後期においては　電信信号でブザー（持続音）を鳴らすことにより、無線通信士も　従事できるようになった。 また有線通信士を　このブザー通信に習熟させ、無線通信士に転換するかも？
軍事通信では　有線と 無線が　混在する場合が多く、特に地上戦では　通信兵は どちらも操作できる必要があった。

ブザーのほかに、低周波発振器を直流電信信号で制御する機器もある。
実例「日本陸軍 九五式電信機」

無線のモールス通信には混信や雑音もあり、信号だけが 受信できる場合は 稀であるが、SN比が マイナス、つまり信号強度のほうが小さい場合も、熟練者なら 目的の信号音を聞き分けられる。 無線電話や データ通信は到底 行えないような通信環境でも、最低限の情報交換が可能であり、モールス通信が21世紀の 今日でも使われるのは、これが理由である。

符号化方式

国際モールス符号は 短点(・)と 長点(ー)を組み合わせて、アルファベット・ 数字・ 記号を表現する。 長点1つは 短点3つ分の長さに相当し、各点の間は 短点1つ分の 間隔をあける。

また、文字間隔は 短点3つ分、語間隔は 短点7つ分 あけて 区別する。

策定については、標準的な英文におけるアルファベットの出現頻度に応じて 符号化されており、よく出現する文字ほど 短い符号で 表示される。 例を挙げると、Eは(・)、Tは(ー)と それぞれ 1符号と 最短である。

逆に 使用頻度が少ないと 思われる Qは(ーー・ー)、Jは(・ーーー)と 長い符号が制定されている。

これに対して、和文のモールス符号では 出現頻度が まったく考慮されておらず、通信効率に劣ったものとなっている。 和文モールス符号で(・)と(ー)が 意味するのはそれぞれ 「へ」と「ム」 で ある。

国際モールス符号ではなく、DOTVの モールス符号 (1854年4月版)を 基に イロハを 当てはめている。

通信速度の表記には、字/分のほか、短点50個分(1ワード)の 1分間当たりの出現回数WPM (words per minute) が 用いられる。短点 50個の 基準として「 PARIS 」の 符号を用いることから PARIS 速度とも 呼ばれる。

例えば 10 WPM は 50字/分に相当する。符号の速度が 同じであっても、英語の平文では 出現頻度の多い文字ほど 符号が短いため、実際の文字数は 多くなることが ある。

大変長い説明で、視聴者の皆様も お疲れ様でした。

最初に、モールスさんが 符号を作ってから186年 経過していますね。 使用される機会は激減していますが、一部の分野では、まだ使用されています。 特に無線通信で 微弱な電波を受信する場合、ノイズの中に信号が入っている状況で、そのようなS/Nの悪い環境でも、熟練の無線通信士の方は、モールス信号を 聞き取れるというのは、すごいなと思います。

話は、飛びますが、昔ジグビーとかいう無線通信モジュール ２台で、マイコンと シリアル通信で接続して、通信を行わせるテストを した事があります。 ２台の ジグビー間の距離が、1mとか近いと、大量のデータを 転送しても、相手に転送する事が出来ます。 が、20～30m 離すと 安定した通信を行うのは、かなり難しくなります。

微弱電波だからしょうがない。 といえば それまでですが、まともなアンテナも 付いて無いので仕方ない部分も あります。

それと、Wikipediaの　モールスの説明の中に JJY の 話が入ってましたが、JJYは、だいぶ前に 無くなったと思ってましたが…。 昔、JJYの電波の時報で、時刻合わせを行っている装置が、あったようですが、その時刻情報取得の電波として、電波時計の基地局が、福島と福岡に設置されたと思います。かなり低い長波で 福島が、40KHz、福岡が 60KHz です。

秋月電子にて、電波時計キットを 10数年前に、購入しましたが、設計、製造したのは、北海道のトライステートという会社です。

で、そのキットに電波時計に送られてくる電波の　時刻フォーマットが、書いてありました。 1分／1フレームの ゆっくりした時刻フォーマットです。 但し、電波法の関係で、毎時　15分と 45分の2回、電波時計基地局の　コールサインを　モールスで送ってくる事になっています。

コールサインを送ってくる時間は、電波時計は、受信情報を　無視していると 思います。

余談のついでです。 あまり 知られて無い事と思いますが、電波時計の電波は、昼と、夜とで時刻が　若干 ズレています。夜の方が、遅れます。 基地局は、常時　正確な時刻情報を 出しています。 でも、受信機というか電波時計には、夜は、昼間より、若干遅れた時刻が 設定されます。何故かというと、長波は、地球上空にある電離層に当たり反射して地表に戻ってきます。 地球には、外側から、A、B、C、D、E　という 5つの電離層があるのですが、通常一番地表に近い E層に　当たって反射します。しかし　E層は、夜になると、消滅します。 よって夜は、一つ上の D層に当たって反射して地表に戻ってきます。

という事で、夜は、E層ではなく　D層という　より高い所の電離層に当たり反射して地表に戻る関係で、送信所から、受信場所までの距離が、夜は長くなる。 という事は　電波が到着するまでの時間が、昼と夜では　夜の方が　伝搬時間が長く 到着時刻が遅れるという事です。

当然、受信場所により、遅延時間は変わります。 私の住んでる場所は、熊本なので
福岡の はがね山の電波 60KHzを 受信します。 私のところでは、昼と夜との 時間差は 60ms ぐらいでした。 昼間も当然遅れて届くので、夜は それに 60ms の遅れが上乗せされます。 よって 電波時計の電波は ミリセコンド単位のシビアな時刻情報は期待出来ません。ふつうの腕時計レベルの使い方であれば、問題ない精度と思います。

シビアな 絶対時刻精度が 必要な場合は
1秒パルス出力機能の付いた GPS受信機がいいと思います。

GPSの話を 書き始めると また長くなるので、今回は 見送ります。

何でモールスの話から、電波時計、GPSの話に なったのかな。？

と思ったら JJYの話からずれてきたのですね。

私が 思うに モールス通信は、今のコンピュータの通信機能の 土台を作ったのではないかと考えます。

私、個人はアマチュア無線の免許も持ちませんし、当然モールス符号を出すとか、聞きとる事はやった事が ありません。 遥か昔、中一の時、2つ年上の先輩が、アマチュア無線を やっていて、よく遊びに行ってました。

高校生になってアマチュア無線の部長に誘われて、あの当時 電話級だったかな。試験を受けて合格したのですが、その後の 開局手続き等が分からず、流してしまった。 という失敗がありました。

その後、無線関係とは縁遠くなりましたが、コンピュータと、モデムを接続したデータ通信とか仕事で やっていたので、データ通信の知識はいろいろ得る事が、出来ました。

で、今回　R8C/Mマイコンで、モールスを出力するには、モールス符号の約束事を　理解する必要が　あります。 先ほど出て来た資料の一部ですが、

国際モールス符号は　短点(・)と 長点(ー)を組み合わせて、アルファベット・ 数字・ 記号を表現する。　長点1つは　短点3つ分の長さに相当し、各点の間は　短点1つ分の 間隔をあける。 また、文字間隔は 短点3つ分、語間隔は 短点7つ分 あけて　区別する。

以上、青、茶 の 色を付けた部分を、しっかり理解する。 プログラムを作る時、必要です。

細かい時間の基準は、短点の長さになります。 短点の長さを3倍すれば、長点に　なります。間隔も、短点、短点x3、短点x7 に　なります。

後は、101の動画で　R8C/Mマイコンで　音楽を鳴らしましたが、基本　同じ要領で　モールス信号を　出力出来るはずです。

前回、音楽のテンポは　固定でしたが、今回のモールス信号出力は、速さを可変出来るようにしようと思います。 両端を　0V、5Vに 接続したボリュームの中点を、A/D入力端子 に 接続します。 A/D変換した ボリューム中点の電圧により、モールスの　時間の基準になる短点の長さを　可変出来るようにすれば　いいと思います。

テラタームから　送られて来る ASCII文字列の受信は　前回の、ヘキサダンプの コマンド入力の機能を　利用出来ます。 ちょっと面倒なのは　アルファベット 26文字分と　記号の分の　モールス文字信号を 出すサブルーチンを作成する事です。 大雑把にこのような考え方で　プログラムを 作成します。

モールス符号 フォーマット

A	■ ▬	N	▬ ■
B	▬ ■ ■ ■	O	▬ ▬ ▬
C	▬ ■ ▬ ■	P	■ ▬ ▬ ■
D	▬ ■ ■	Q	▬ ▬ ■ ▬
E	■	R	■ ▬ ■
F	■ ■ ▬ ■	S	■ ■ ■
G	▬ ▬ ■	T	▬
H	■ ■ ■ ■	U	■ ■ ▬
I	■ ■	V	■ ■ ■ ▬
J	■ ▬ ▬ ▬	W	■ ▬ ▬
K	▬ ■ ▬	X	▬ ■ ■ ▬
L	■ ▬ ■ ■	Y	▬ ■ ▬ ▬
M	▬ ▬	Z	▬ ▬ ■ ■

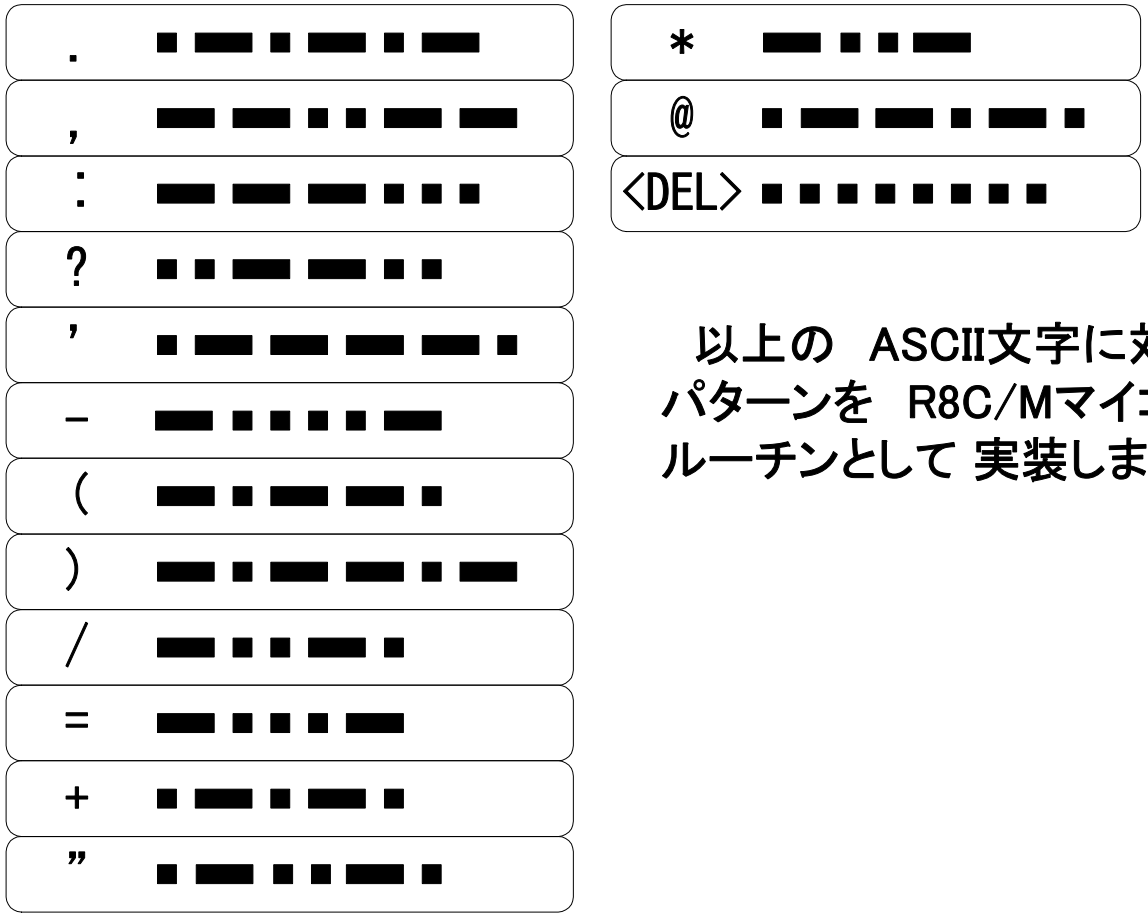
因みに、英字は 大文字 小文字の 区別は無いようです。 次は、数字です。 略体の数字もある様ですが、一部の英字と モールス符号のパターンが同じなので、略体は、見送ります。

0	▬ ▬ ▬ ▬ ▬	5	■ ■ ■ ■ ■
1	■ ▬ ▬ ▬ ▬	6	▬ ■ ■ ■ ■
2	■ ■ ▬ ▬ ▬	7	▬ ▬ ■ ■ ■
3	■ ■ ■ ▬ ▬	8	▬ ▬ ▬ ■ ■
4	■ ■ ■ ■ ▬	9	▬ ▬ ▬ ▬ ■

次に、欧文記号というのが ありますが、15文字あるので、次のページに 表示します。

乗算の記号 × は、ASCII 文字の ＊ に 割り当てます。 それと短点 8つの 訂正記号が、ありますが、ASCII 制御文字の DEL (7Fh) に 割り当てます。

モールス符号　記号フォーマット



以上の　ASCII文字に対応する　モールス出力パターンを　R8C/Mマイコンに 文字単位のサブルーチンとして 実装します。

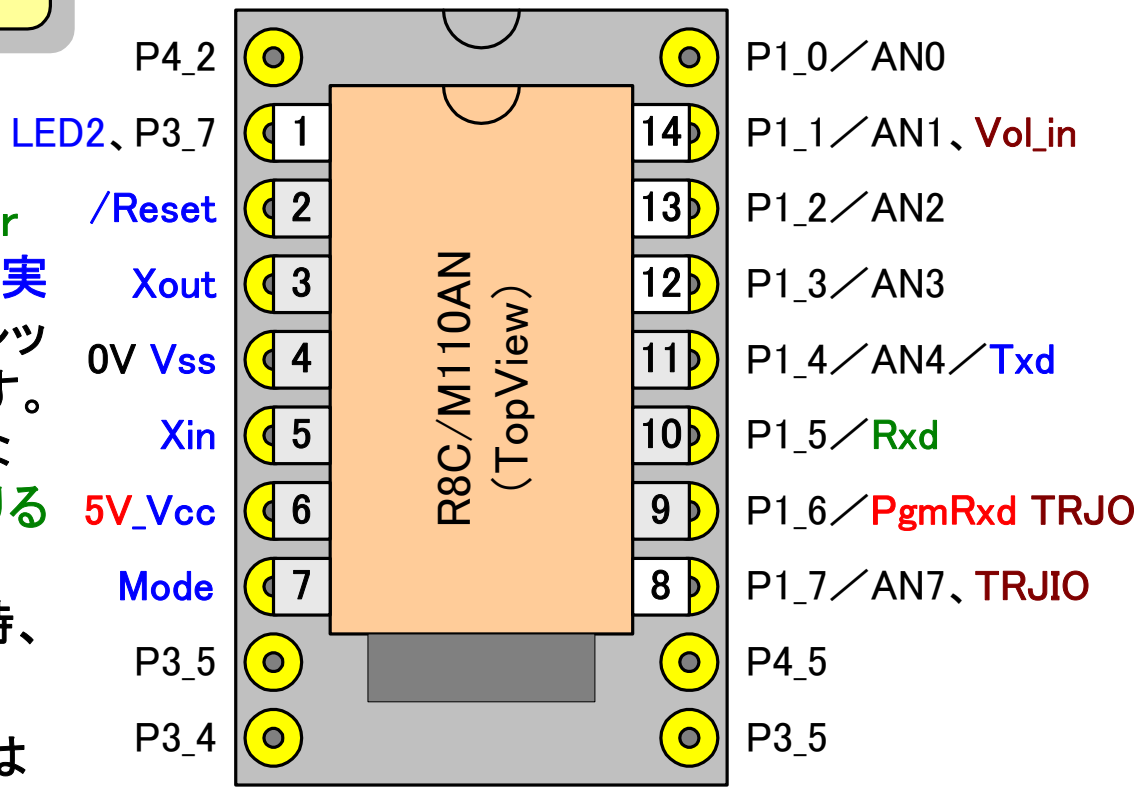
R8C/M小基板にて回路を構成する

107 動画で、作成した小基板を用いて
今回の回路を 構成します。

シリアル通信と、CPUモード切替え、Power Onリセット、CPUクロック水晶は 小基板に 実装されているため、それ以外の回路を、ブレッドボード上で、構成すれば よい事になります。

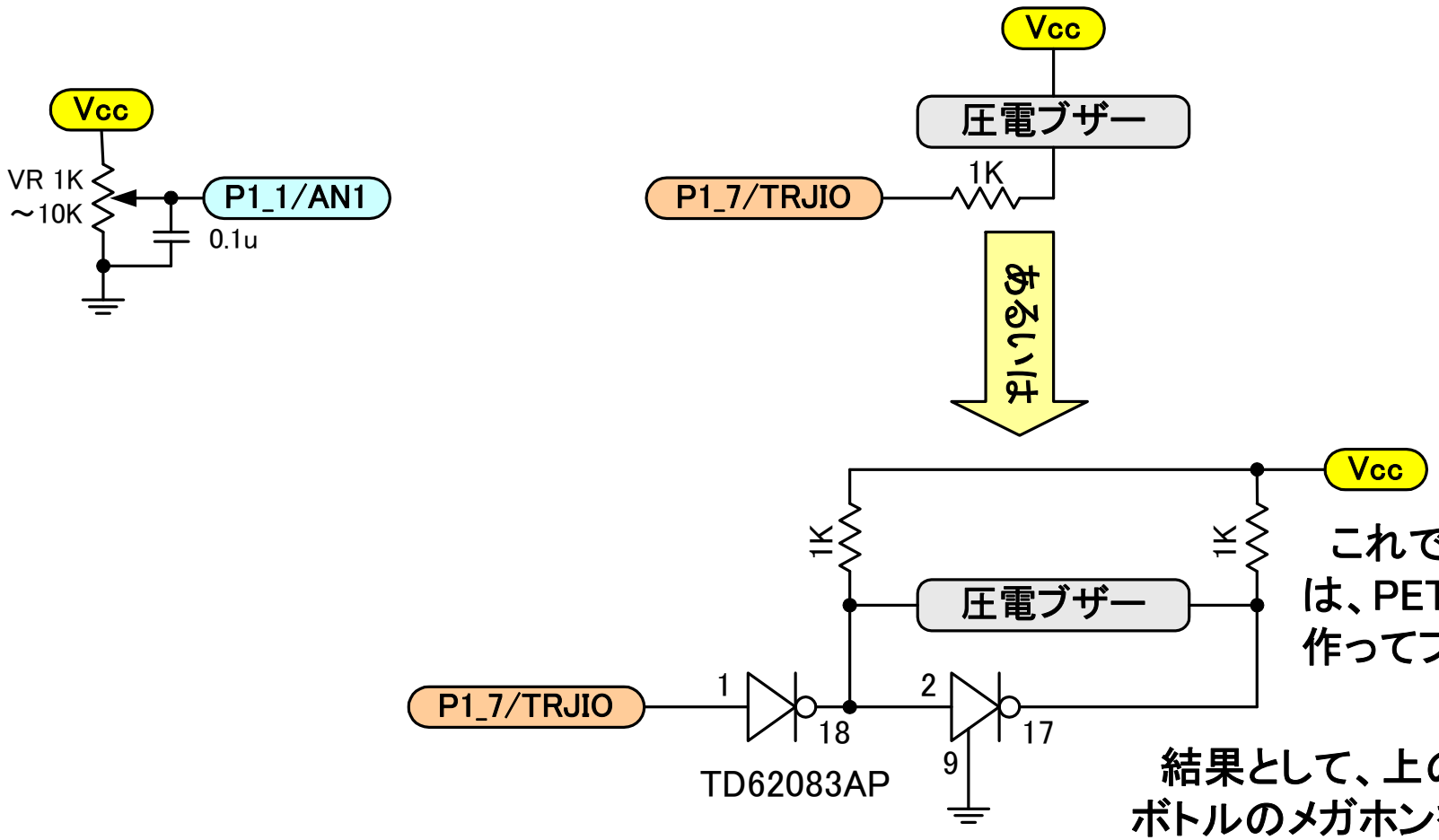
圧電ブザーと、A/D入力は 1chしか使わないので、R8C/M110ANでも 足ピン数は 足りると思います。 それと、小基板に付いている LED2(P3_7)にて、ブザーを鳴らしている時、LED2も 同時に点灯させようと思います。

ボリュームの中点入力の A/D入力端子は P1_1/AN1(M110ANの14Pin)に 接続します。 ブザー出力は、タイマーRJ2の出力を使います。 出来ればバイポーラ接続にして音量を上げたいところですが、TRJOと PgmRxd がぶつかっているので、TRJIOのみ使用します。



音量が、小さい場合は、TRJIO出力に、オープンコレクタバッファを 2段付けて
ブザーを バイポーラ接続にしようと思います。

小基板の外に追加する回路

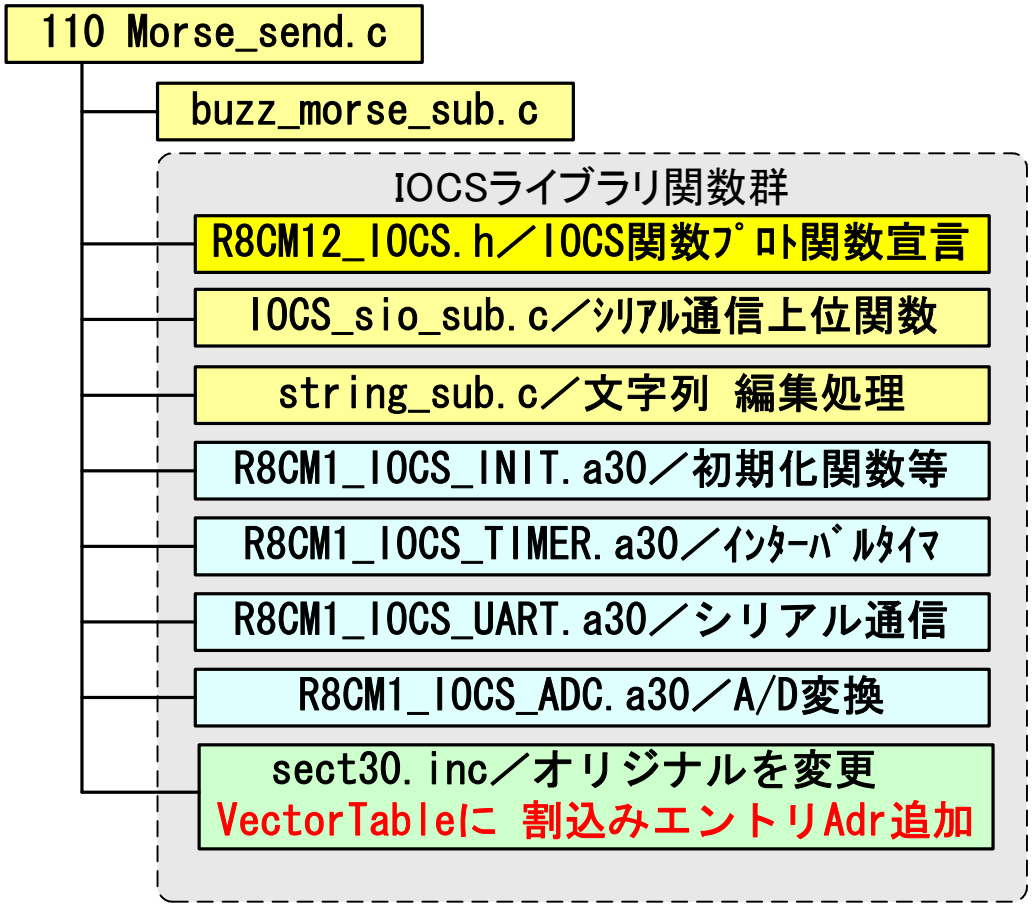


これでも音量が小さい場合は、PETボトルで メガホンを作ってブザーに付けて下さい。

結果として、上の簡単な回路に PETボトルのメガホンを付けました。 この用途では、十分な音量でした。

HEWプログラム概要の 説明

今回の HEWプロジェクトの ソースファイルの モジュール構成を説明します。



"110 Morse_send.c"が、main関数が 入っているソースです。

その下の "buzz_morse_sub.c"が、今回のモールス信号出力処理の サブルーチン群が、入っているソースです。

その下の IOCSライブラリ関数群は、R8Cの周辺回路、I/O回りの関数、及び文字列編集の関数群のソースです。
拡張子 a30 は、アセンブラです。

IOCSに、どのような関数が含まれているかは R8CM12_IOCS.h を 参照して下さい。
中身は、関数のプロトタイプ宣言と、右側に簡単なコメントが 付いています。
各関数が、どのソースに含まれるか、分かる様にしているので、細かい仕様は ソースを見て下さい。

main関数内の ループ処理

```
sio_print( "* R8C/M  Send Morse Signal. ( v_1.1 )" );  // open message
beep_1();          // ビープ 音鳴らす

while( 1 )
{
    sio_txt_input( ">", Txt, 80 );    // 文字列の受信処理
    oomoji( Txt );                    // 文字コード大文字化
    ptr = Txt;                        // ポインタ設定
    while( *ptr != NULL )
    {
       morse_char_send( *ptr );       // モールス １文字送信
        ptr++;                        // ポインタインクリメント
    }
    sio_put_crlf();                   // 改行コード出力
}
```

外側の while ループが 行単位のループに なります。 内側の while ループが 文字単位の ループに なります。 青文字の関数 morse_char_send 関数が、1文字モールス出力関数です。 morse_char_send 関数の実態は buzz_morse_sub.c 内の 下の方に あります。

buzz_morse_sub.c 内の関数

```
//**  モールス文字出力                    **
//** ------------------------------- **
//**  引数 :  C = 出力文字コード        **
//*************************************
void  morse_char_send( char c )
{
    sio_send( c );              //  1文字 送信

    if(('@' <= c) && (c <= 'M'))
    {
        switch( c )
        {
            case '@':  morse_atmark();  break;
            case 'A':  morse_A();       break;
            case 'B':   morse_B();      break;
            case 'C':  morse_C();       break;
```

```
//*****************************
//**  モールス信号  B 出力  **
//*****************************
static  void  morse_B( void )
{
    dah_sound( 1 );       // ツー
    dot_sound( 1 );       // ト
    dot_sound( 1 );       // ト
    dot_sound( 0 );       // ト
    time_interval_3();    // 長点の時間待ち
}
```

赤の矢印は、文字 B の出力関数の 呼び出し部分と、呼び出される その実態を 示しています。 dah_sound関数が、長点を出力する関数です。 dot_sound関数が、短点を出力する関数です。

morse_char_send関数内には、A～Zの文字と、数字 0～9と 記号 15文字が、case 文と if 文で縦に 並んでます。