エレキーをプログラムする

2020.11.29　JA1VCW

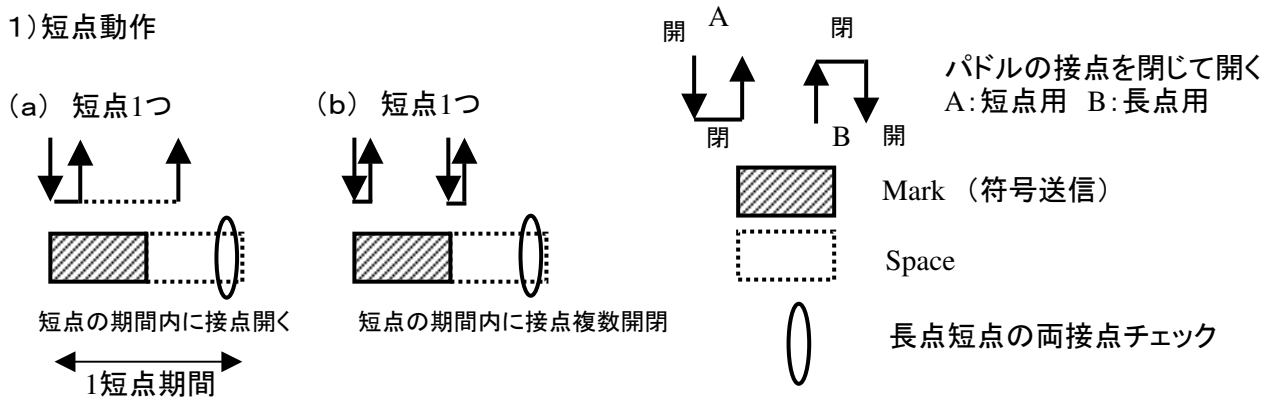
1．　初めに

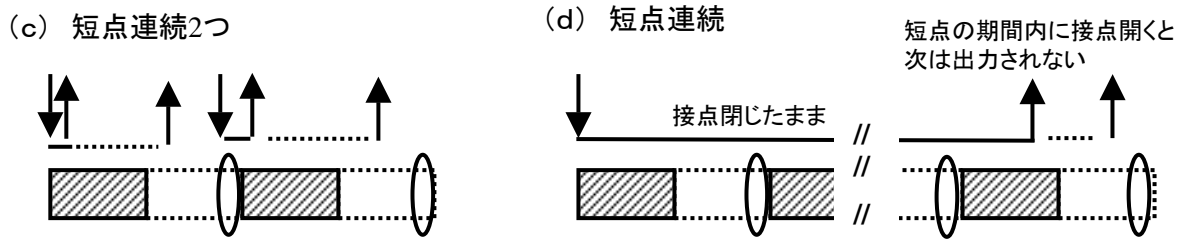
　電信が得意なわけでは無いのですが、現用のエレキーは74-TTLを使用した古いものです。
新しくしたいと思ってワンチップマイコンで作ってみました。 現用がスクイーズ・キーヤー
（squeeze keyer またはiambic keyer）で機能はそのままで良いのでその動作をプログラムしました。
私の備忘録です。（形式を整えて書いておかないと後でわからなくなるので）

2．　機能

　まず動作を調べました。 現用を詳しく調査したわけではありませんが、まあこんな所だろうと。

1）短点動作





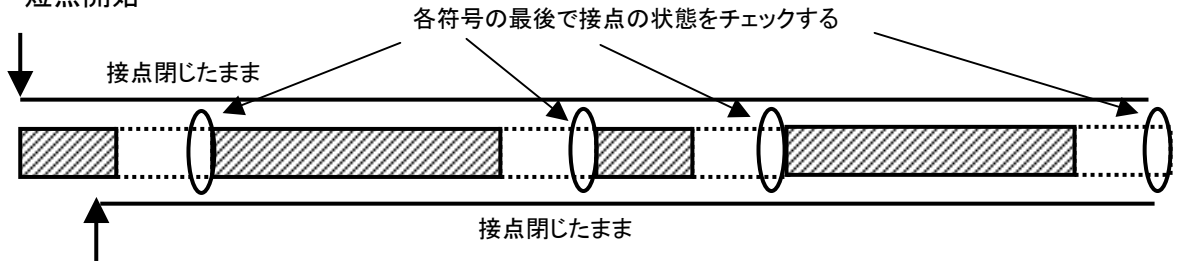
markとspaceで1短点とし、spaceの最後の時点で接点の状態をチェックして、接点がが閉じていると連続して次の符号が送出されます。 すべてこのタイミングの接点の状態で次の動作が決まります。

2）長点動作

　短点動作と全く同じで、markの時間のみ短点の3倍の長さになる。

3）スクイーズ動作

（a） 短点開始

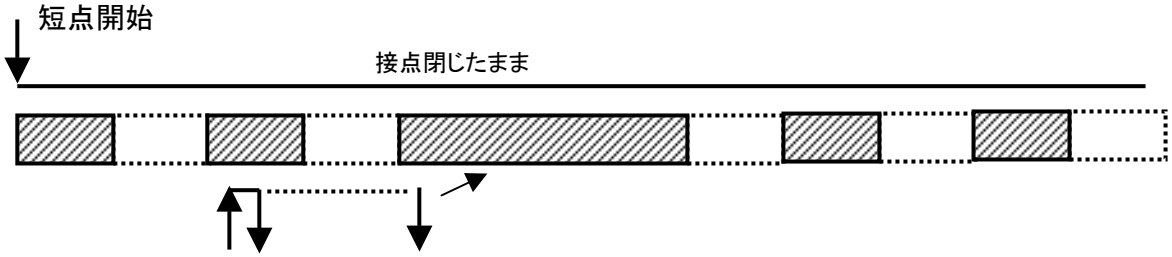


短点から開始し接点を閉じたままにすると、最初に短点を発生した後は長、短点の繰り返し。
長短点の各スペースの最後の部分の接点の状態で動作が決まる。　終了は1，2）と同じ。

（b） 長点スタート

開始が長点から始まる以外（a）と同じ。

（c） 連続短点に長点割り込み



短点接点閉じたままで長点を一瞬閉じると、次の符号のタイミングで長点が割り込む。
割り込みの場合は符号の最終部分のチェック点までずっと接点が閉じている必要は無く、一瞬でも閉じれば短点の後に長点が割り込む。 長点接点を閉じたままにすると、長、短点の繰り返し。
（a）（b）と同様。

（d） 連続長点に短点割り込み

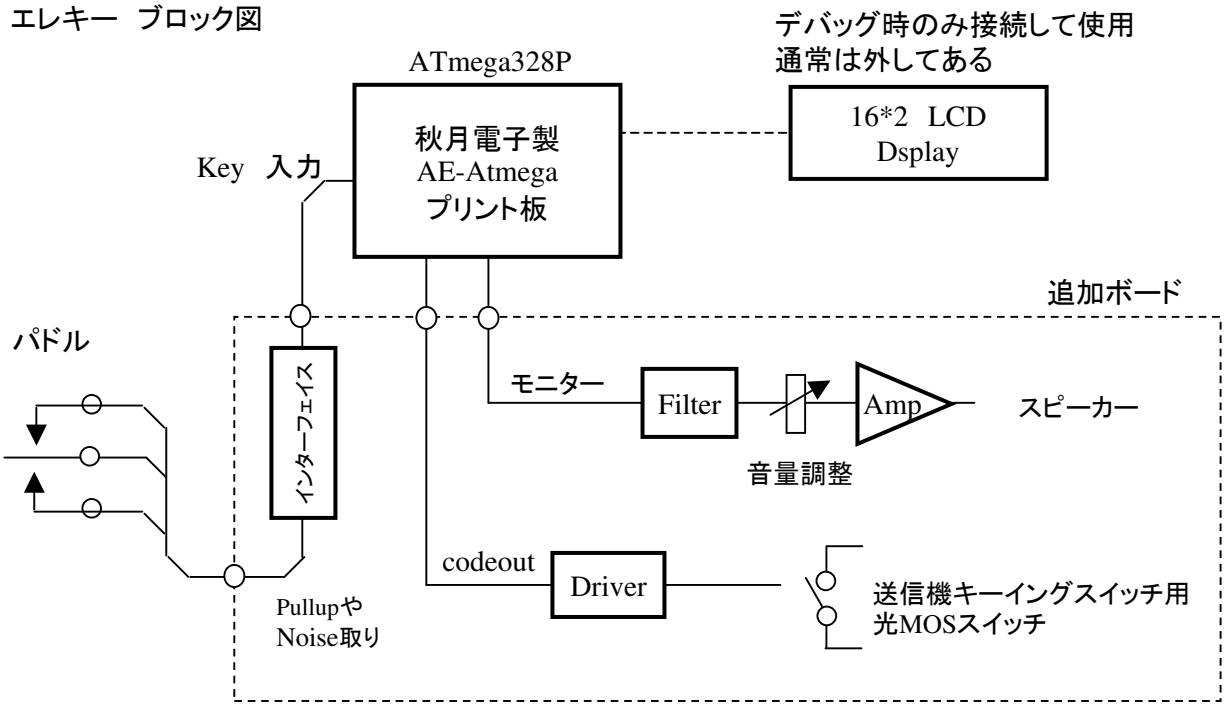
（c）において長点と短点を逆にした動作。

スクイーズの動作が正しいかどうかわかりませんが、これを仕様と考えてプログラムを組みました。

3． ハードウェア

CPU、key入力、モニター用アンプ および送信機キーイング用スイッチ これで全部です。
CPUは ATmega328P に決めていますので、秋月のプリント板が便利です。
それに実配による周辺回路を付けました。 これらは単純なものです。

エレキー ブロック図



4．　キー操作速度と短点の時間

キー操作速度としては　毎分10文字～150文字位を可変できれば良いでしょう。
目安の数値を考えます。

・1長点＝3 短点
・各点･線の間＝1 短点分の間隔をあける
・文字間隔＝3 短点分の間隔をあける
・語間隔＝7 短点分の間隔をあけて区別する

{ 1語5文字（50短点）＋間隔（7短点）}＝57短点とした時、キー操作速度と短点の時間は
　毎分10文字では57短点*2=114短点　→ 1短点当たり約530ms
　毎分150文字では57短点*30=1710短点　→ 1短点当たり約35ms
wait命令を使えばこの時間は簡単に設定でき、且つ命令実行中にインターラプトが使用できます。
今回のように時間精度が要らない場合は比較的単純にプログラムできそうです。

5．　ハードウェアおよびプログラムの方針

5．1　マイコン

1）マイコンはAVR。 私はAVRはATmega328Pー種類に決めていますので、秋月のプリント板とともに使用しました。
2）プログラムはBasicコンパイラを使います。 BascomAVRです。
　今回のプログラムの容量は4kbyte以下です。
3）プログラムの書き込みは、アイテンドーのAVRライタ[USB-ASP]と指定の書き込みプログラムProgISPを使用しました。
4）AVRライタと秋月の基板ではコネクタが合わないので、変換ケーブルを自作しました。
5）I/Oの割り当てです。

信号名称	I/O	端子	機能	ｺﾈｸﾀ	論理	機能記述
dot_input	I	PD2	INT0	J1-3	負	dotの接点入力端子
dash_input	I	PD3	INT1	J1-4	負	dashの接点入力端子
codeout	O	PB5	SCK	J2-6	正	キーイングリレー動作出力
speedcont	I	PC5	ADC5	J3-6	ー	speed設定用の電圧入力
moniout	O	PD6	OC0A	J1-7	ー	モニター用700Hz出力

　他にLCDのピン割り当があります（回路図参照）。　LCDはデバッグ用です。
6）電源はケータイ充電用の5Vのスイッチング電源のコネクタを取り替えたものです。

5．2　プログラムの方針

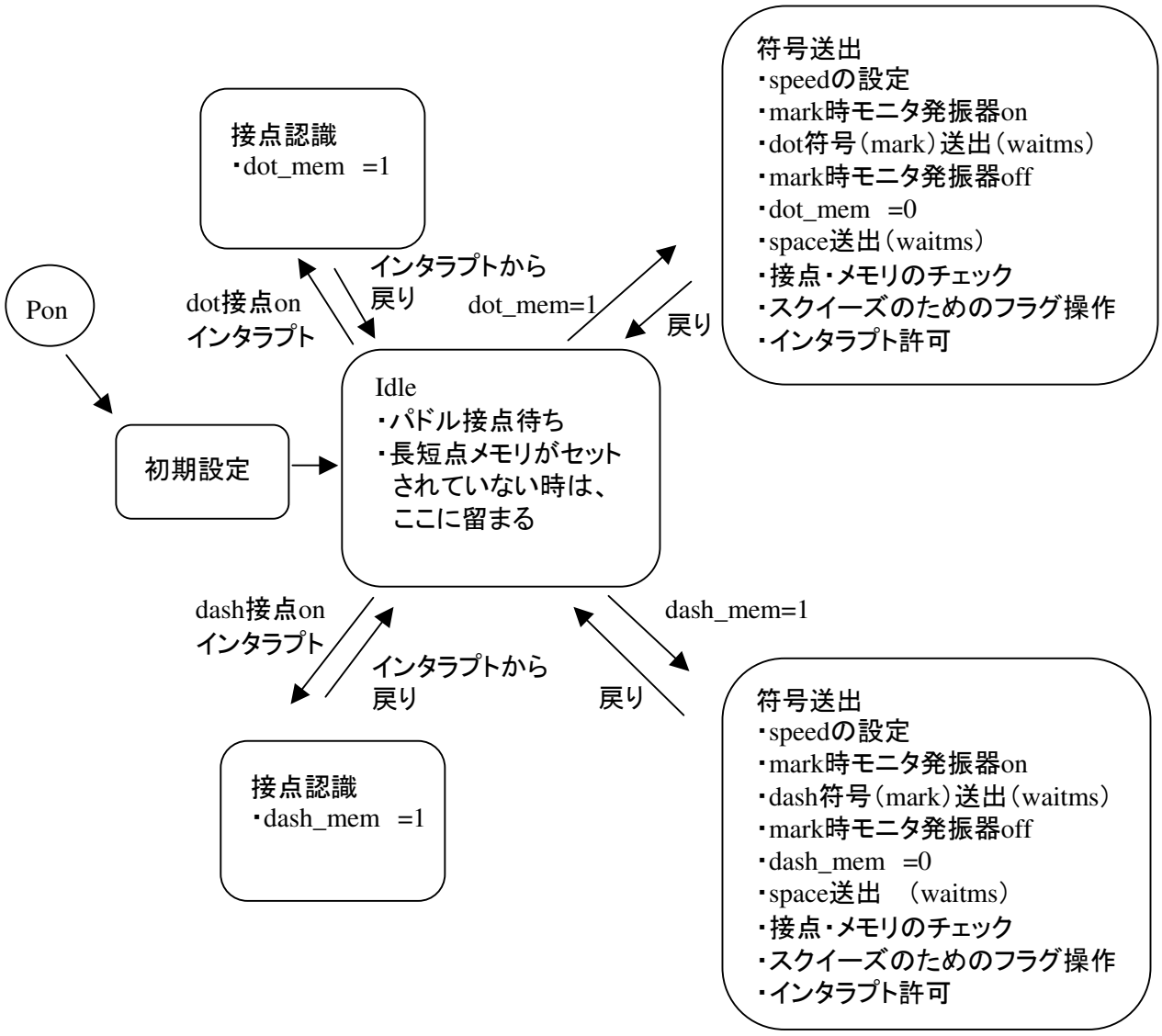
1）長点、短点、スペースの時間はソフトウェアのwait命令を使用します。 理由は
　（a）正確な時間間隔は不要です。
　（b）動作中に割り込みが使えます。
2）長点、短点の接点の情報は割り込みによって取り込みます。
3）ハードウェアのカウンタ／タイマー（Timer0）を使って、モニター用の発振器の動作をさせます。
4）電池を使用しての運用等での消費電力を少なくするような方策は行っていません。

6. プログラム

6．1 状態遷移図

フローチャートの代わりに状態遷移図（ステートダイアグラム）を書きました。
状態遷移図はネットで検索するといっぱい出てきます。 正式な書き方はあるようです。 今回は自己流ですが大きく違ってはいないと思います。
この手の図はどこまで細かく記入するかが難しいところで、あまり細かい所までかくとなるとプログラムそのものになってしまうし、省略すると見てもわからないという事になります。 フローチャートなども同様です。

エレキー状態遷移図

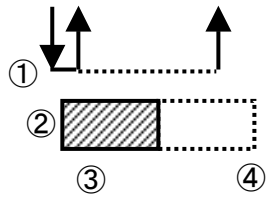


タイミングと合わせて考えると次のようになります。（次ページ）

6．2 詳細

短点(長点)の場合の処理の詳細。

長点の場合はmarkの時間が3倍になるのと、dot → dashに、短点→長点と読み変えてください。

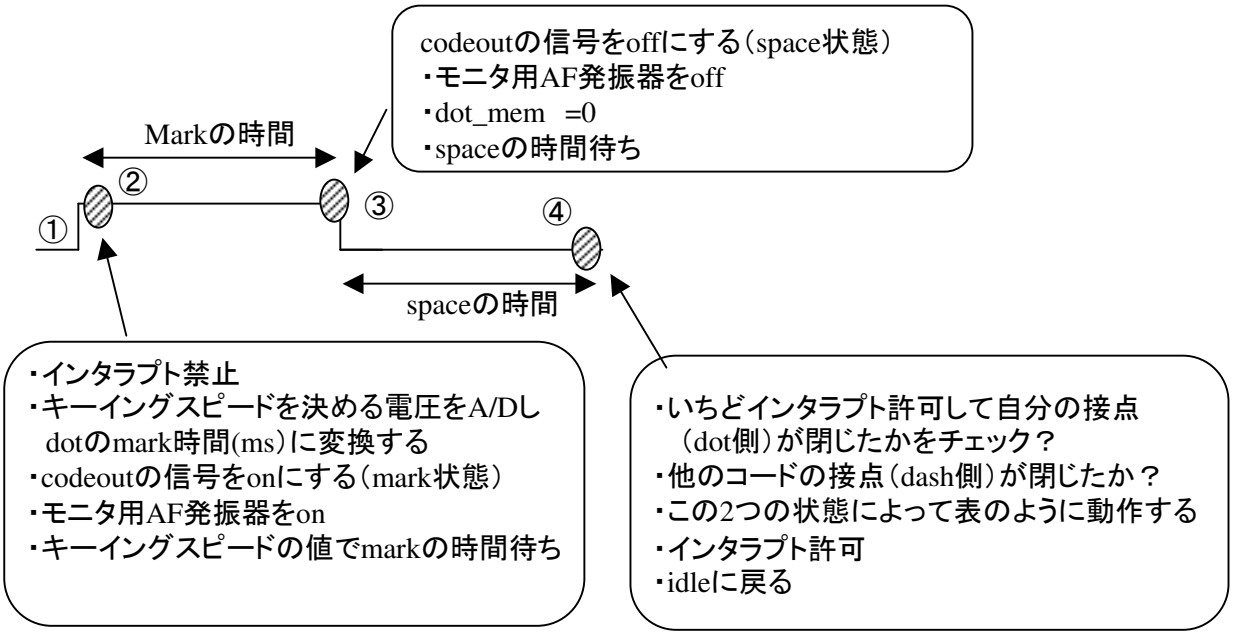


①接点が閉じてインタラプトがかかり、接点認識の状態で dot_mem =1 とし、その後戻ります。

②idleの状態で dot_mem =1 なので符号送出の状態に入ります。

③符号送出の状態でmarkの時間です。

④符号送出の状態中ですが、下図のようにコントロールします。 長短点メモリあるいはスクイーズ動作に必要です。



dotとdashの接点によって次のように動作します。

dot接点	dash接点	動作
open	open	処理は無し dot終了
close	open	処理は無し 結果的に次はdotを出力(連続dot)
open	close	処理は無し 結果的に次はdashを出力
close	close	強制的に dot_mem=0にする。 結果的に次はdashを出力

上図④の時間に接点の状態をサンプリングして次の動作を決めています。

このサンプリングの時間を③に変えると、キーイング速度が速くなったときに短点が一つ多く出るミスが多くなります。 多分パドルの接点を開く操作が遅れて③よりも超えてしまうためと考えます。 それで現在④の位置にしています。

7． 結果

1）実際に使ってみて今まで使ってきたICのキーヤーと同等に使用できます。

2）参考のために回路図とプログラムを添付します。（後出）
プログラムについてはこれが最良かどうかわかりませんが、私が使用した時の動作としては特に問題なさそうです。

3）プログラムの容量はおよそ1.5kバイト強でした。 CPUは 32kバイトの容量があるので4.7%しか使用していません。 RAMなども同様です。 スカスカです。

8． ウエイト・コントロール（weight control）について

weight control という機能を付けたキーヤーがあります。

キー操作速度が速い時、遅い時等にスペースの時間を可変して聞き易い符号にする機能です。
本機では短点、長点、スペースの時間を独立に設定できますので、ハードとプログラムの変更で個性のある符号を作ることができます。 以下のようにすれば容易に実現できます。

（a） speed conrol の回路と同じように5VとGNDの間に可変抵抗（VR）を接続し、その中点を空いているADCに接続します。 VRの回転に応じた電圧値を読み取れるようにします。

（b） VRの中点が（短点:スペース）の比を（1:1）とし、それに対してVRをまわせば係数、例えば 0.xx～x.xx を算出します。

（c） サブルーチンspeeddata の中で 長点、短点、スペースの各時間に対してweight control が必要と思われる時間にVRの電圧から計算した係数を乗じます。

（d） あるいはキー操作速度の電圧によって係数を決めて各時間に係数を乗じます。
この場合はハードの追加は無くて自動ウェイト・コントロールになります。

私の場合 waight control は必要がないと考えたのでこの機能は入れていません。

9． 感想など

1）以前のキーヤーも40余年使いましたので元は取っている感じではあります。 長い間使いましたので愛着はあります。

2）スクイーズキーヤーをディスクリートで設計した方は尊敬に値します。 なかなかああいうロジックは簡単には設計できません。 少なくとも私には無理です。

3）インタラプト使用のプログラムはなかなかデバッグが難しいです。 まだ私が未熟なのでしょう。

4）エレキー以外にエレバッグ（短点は自動送信、長点は手動）、複式など、動作を切り替えられる様にすることも簡単にできます。 しかし私の場合エレキーはコンテスト以外にほとんど使用しないので他の動作は付けませんでした。（動作切替のスイッチを付けるのがおっくうでした）

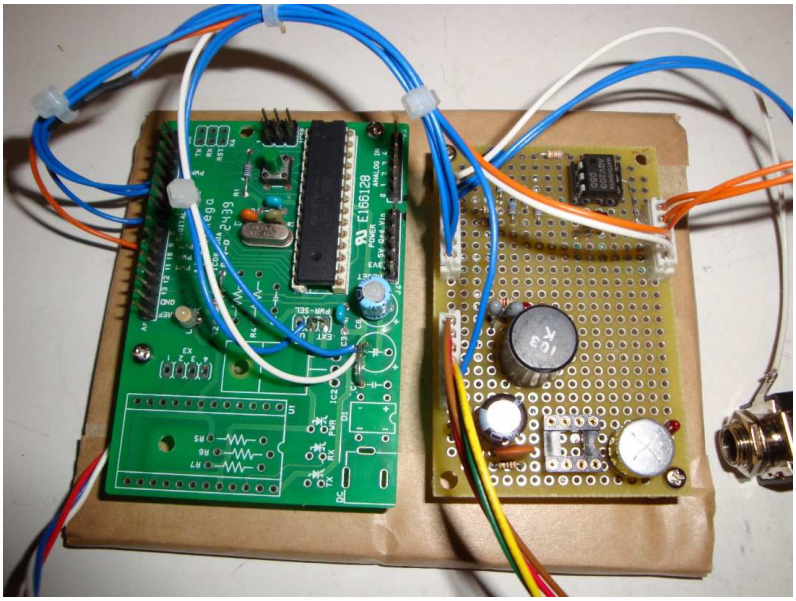
5）今回はプログラムの容量で4.7%しか使用していないのでもったいないです。 もっと小規模なCPUでも良いのですが、プリント板やそのほかCPU固有のパラメータを考慮しなければいけない等を考えると、少々規模が大きくなっても使い慣れたもののほうが良いと判断しました。
もし製品などを作るとなるとコストなど考える必要があるのですが、趣味で1個しかつくらないのでこれでよいと思います。

6）光MOS スイッチを送信機のキーイングスイッチとして使用しています。 このAQV203は高価です。たまたま手持ちにあったので使いましたが、最近はもっと安価なものがあります。

7）外付けのモニター回路が大きくなったためにもう1枚基板が増えてしまいました。 モニター回路にセラミックスピーカーなどを使用すればアンプが要らなくなるので、小さい基板に作ってCPU基板のUSB-シリアル変換モジュールの部分に載せてしまえば、CPU基板だけの大きさでまとまります。

8）たまたま百均を歩いていた時に、手ごろと思われるプラスティックの箱があったので買ってそれに入れてみました。 ちょっとチャチで少し小さめ。 キツキツです。 まあ、飽きたらちゃんとしたケースに入れます。　当面はこのままです。

10.　写真



モニター用のLM386の手持ちが
ちょうど無くて配線だけしてあります
（後で付けました）

という事で完成した姿ではありませんが
動作は確認してあります

左は秋月のボードとATmega328P

右の実配基板は秋月のユニバーサル
基板で次の回路が配線されています。

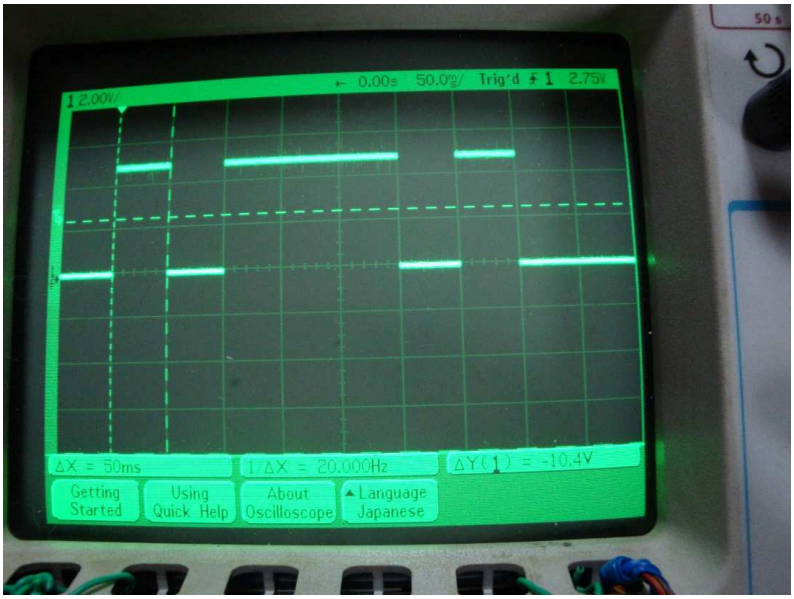
・キー入力インターフェイス
・送信機のキーイング出力
・モニター用アンプ（LM386）とフィルタ

上の写真のボードを、百均で買った
プラスティックの箱に入れました
大きさはおよそ　100*45*140　mm

電線が部分的に短いものがあって
うまくバインドできずに見た目が良くない

箱はスチロール系でひびが入りやすく
穴あけ、ネジ締めなどに注意が必要です

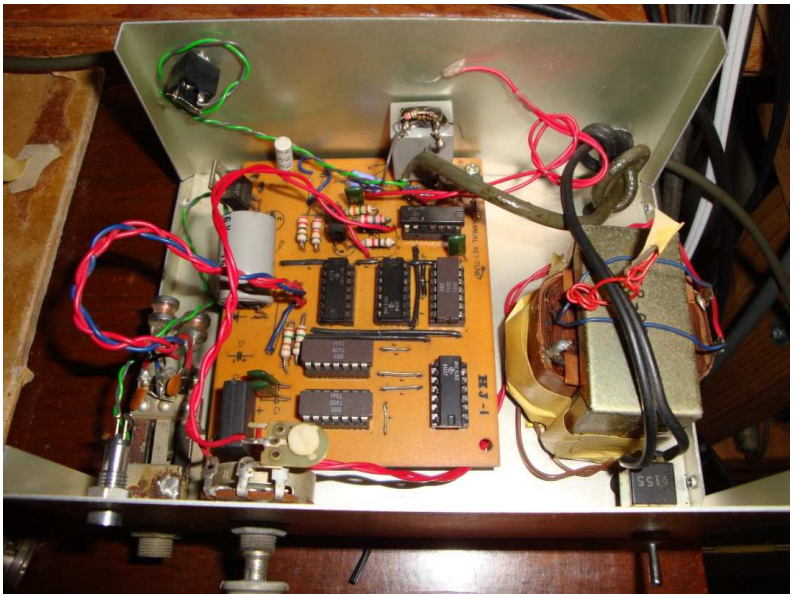
ツマミなどは左から
電源SW、スピードコントロール、
モニタ音量、キージャック
です。



“R”の信号を出力した時の波形

長点、短点、スペースの長さはOK
（5Vがmark）

スクィーズ動作もOK

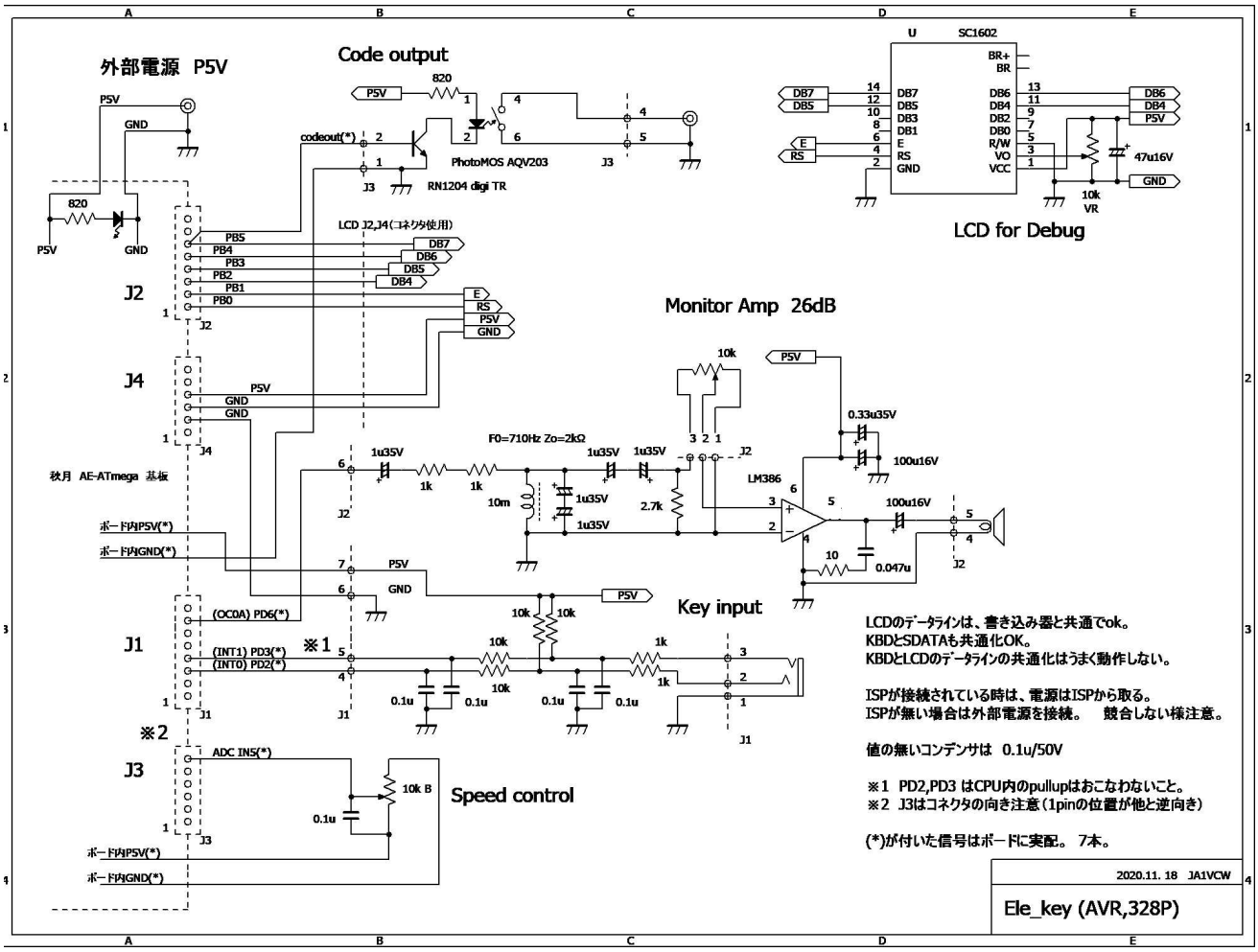


今まで使用してきたキーヤー
40年以上前から使用して来た
古いもので標準TTL使用
プリント板のみはどこかの製品です。

セラミックパッケージのSN7400が3個
使ってありますが、これなどもう入手
できないかもしれません（骨董品）
もちろんセラミックパッケージである
必要は全くありません

11. 回路図

参考回路図です。 この原稿を書いた時点の、動作している回路です。
万全を期していますが間違いがある可能性は否定できません。 その時はご容赦ください。
あと、今後いろいろと改善、改造を行って回路が変る可能性はあります。



12. プログラム（参考）

```
Rem  Electoric keyer

Rem   memo Fuse bit    E1,99,FD  Lo,Hi,Ex       Default
Rem   memo Fuse bit    62,D9,FF  Lo,Hi,Ex       Default

Rem   Use this value Fuse bit    E7,D9,FF  Lo,Hi,Ex       @16MHz Xtal

Rem AE-ATmega board by Akizuki elec.
Rem ATmega328P 16MHz


$regfile = "m328pdef.dat"                    'ATmega328P
$crystal = 16000000                         '16MHz

'----- 16*2 LCD display for debag -----
Config Lcdmode = Port
Config Lcdbus = 4
Config Lcdpin = Pin , Db4 = Portb.2 , Db5 = Portb.3
Config Lcdpin = Pin , Db6 = Portb.4 , Db7 = Portb.5
Config Lcdpin = Pin , E = Portb.1 , Rs = Portb.0
Config Lcd = 16 * 2

'----- acoustic moniter -----
Config Timer0 = Timer , Prescale = 64 , Clear Timer = 1 , Compare A = Toggle
' use Timer0, Timer mode,prescale=64(4us),counter clear,compareA toggle
Stop Timer0

'----- ADC for speed control -----
Config Adc = Single , Prescaler = Auto , Reference = Avcc       'use ADC

Config Portd.2 = Input                      '*dot & int 0
Config Portd.3 = Input                      '*dash & int 1
Config Portd.6 = Output                      'OC0A moniter tone  output
Config Portb.5 = Output                      '*code output

'Set Portd.2                               'pullup
'Set Portd.3                               'pullup

Config Int0 = Low Level                      'Low Level  dot-contact
Config Int1 = Low Level                      'Low Level  dash-contact

Dim Dotmem As Byte                           'dot memory
Dim Dashmem As Byte                           'dash memory
Dim Keyspeed As Byte
Dim Dotlength As Word                        'dot length
Dim Dashlength As Word                        'dash length
Dim Dotspacelength As Word                     'dot space length
Dim Dashspacelength As Word                     'dash space length
Dim Wk0 As Byte
Dim Wk1 As Byte
Dim Wk0w As Word
Dim Wk1w As Word

Codeout Alias Portb.5

On Int0 Dotcontact                          'dot-contact
On Int1 Dashcontact                          'dash-contact
```

```
Enable Int0
Enable Int1

   Reset Codeout                        'Initial Code Off
   Dotmem = 0                           'initial value
   Dashmem = 0                           'initial value
   Compare0a = 184 - 1                      'about 680Hz


'       Dotlength = 70                    'initial value 70ms ***
'       Dashlength = Dotlength * 3
'       Dotspacelength = Dotlength
'       Dashspacelength = Dotlength

Cls
Waitms 500
Upperline
Lcd "Ele_key program"
Reset Codeout
Waitms 500

For Wk0 = 1 To 5
     Set Codeout
     Waitms 300                         'LED on/off 5 times for test
     Reset Codeout
     Waitms 300
Next

Enable Interrupts

'-----  main prog. starts here  -----

Mainstart:
Rtndash:
   If Dotmem = 1 Then Goto Dotmark
Rtndot:
   If Dashmem = 1 Then Goto Dashmark

   Goto Mainstart

'-----  dot-contact -----
Dotmark:
   Start Timer0                         'monitor on
   Set Codeout                          'dot on
   Gosub Speeddata                          'get speed data
   Waitms Dotlength
   Dotmem = 0                            'dot off
   Reset Codeout
   Stop Timer0                          'monitor off

'   Enable Int0
   Waitms Dotspacelength                      'dot space time
   Enable Int0
   Waitus 10
   If Dashmem = 1 Then Dotmem = 0
   Enable Int0
   Goto Rtndot
```

```
Dashmark:
   Start Timer0                          'monitor on
   Set Codeout                           'dash on
   Gosub Speeddata                        'get speed data
   Waitms Dashlength
   Dashmem = 0
   Reset Codeout                          'dash off
   Stop Timer0                           'monitor on

'   Enable Int1
   Waitms Dashspacelength                    'dash space time
   Enable Int1
   Waitus 10
   If Dotmem = 1 Then Dashmem = 0
   Enable Int1
   Goto Rtndash

'-----  speed control  -----
Speeddata:
   Start Adc
   Wk0w = Getadc(5)                        'VR voltage
   Dotlength = 40000 / Wk0w                  'Linearlize speed
   If Dotlength > 2000 Then Dotlength = 2000      'max length
   If Dotlength < 20 Then Dotlength = 20          'min length
   Dashlength = Dotlength * 3                'dash
   Dotspacelength = Dotlength                'space
   Dashspacelength = Dotlength               'space
'Upperline                                 'for debug
'Lcd Dotlength
    Return

'-----  dot-contact interrupt  -----
Dotcontact:
   Disable Int0
   Dotmem = 1                             'set dot memory
   Return

'-----  dash-contact interrupt  -----
Dashcontact:
   Disable Int1
   Dashmem = 1                            'set dash memory
   Return



End
```

参考プログラムです。
原稿執筆時で動作しているプログラムで、今後予告なく変わることがあります。
デバッグの為に命令を書いて、あとでコメントにして消してあるところがそのままになっている場所があります。

以上