目次

1. 安全上の注意事項

⚠警告

半田ゴテの高熱とやけどに注意

半田ゴテは 300℃を超えるかもしれない超高温です。触れると皮膚に水ぶくれができるだけではなく、ヒリヒリして夜も眠れません。絶対に触れないようにしてください。やけどした場合は冷水ですぐに冷やしてください。

①取手のみを握るのは大丈夫　②発熱部には電源を切っても絶対に触れない
③半田ゴテを落としそうになった時はコテを掴まない　④ふざけて振り回さない
⑤使わない時はコテ台に置く　⑥人に向けない

ホットボンドの高熱とやけどに注意

ホットボンドも超高温です。ホットボンドだけではなく、溶けた樹脂もかなりの高温になっています。溶けた樹脂が皮膚に付いた場合は素早く取り除いてください。その後、冷水ですぐに冷やしてください。

⚠注意

今回の工作実験では高温になる道具を使用します。やけどに注意し、スタッフの言う事をよく聞いて、行動してください。

今回の工作実験ではパソコンなど高価な道具を使用します。愛を持ってそれらを使用し、決して乱暴に道具を扱わないようにしてください。

もし、やけどなど体に異常がある場合はすぐにそばのスタッフに声をかけてください。トイレも我慢しないように。

①LED 発光ダイオードや②電池ボックスの接続、③電池の挿入には向きに気を付けてください。もし、間違っていると回路が破壊する恐れがあります。

2. 電子オルゴール各部の名称

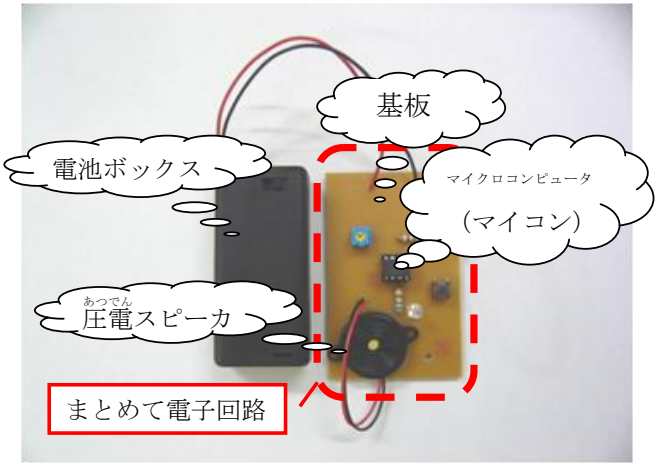


図 1　電子オルゴールの各部の名称

今日の工作で作成する電子オルゴールの各部の名称を図 1 に示します。

電子オルゴールには PIC マイコンという小さなコンピューターが搭載されています。イメージとしては小さなパソコンと思ってください。

このマイコンが圧電スピーカを使って音楽を演奏します。♪～

3. 使用する工具

図 2～図 4 に工作に使用する工具を示します。半田ゴテとホットボンドは高熱になりますので注意してください。

① 半田ゴテは半田という鉛と錫の低融点合金を使って半田付けに使用します。半田付けとは、2 つの金属を接合する事を言います。

② ホットボンドは接着剤の代わりに溶けたプラスチックを使用します。固まると接着力と共に適度な硬さが得られます。

③ ワイヤストリッパは電線の絶縁体である導線の周りのゴムを剥くのに使用します。剥き出しになった銅を基板に半田付けします。

④ ニッパは導線を切ったり、基板に付けた部品の余計な部分を切断するのに使用します。

⑤ プラスドライバーはネジを回すのに使用しますが、今回は可変抵抗器の調整に使います。

図 2　左から、ワイヤストリッパ 1, ワイヤストリッパ 2, ニッパ, プラスドライバー

図 3　半田ゴテとコテ台

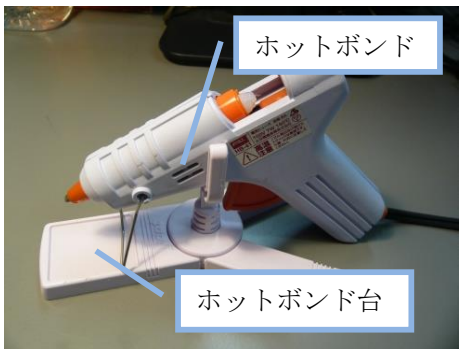


図 4　ホットボンドとその台

3.1. 工具の使い方

半田の付け方（コテの当て方）

半田付けを行う手順を説明します。良い半田付けが出来るかどうかが良い電子回路になるかにかかっています。図 5 は以下の手順を説明しています。図 6 は半田付けの良い例と悪い例を示しています。半田は付け過ぎてはなりませんが、多過ぎてもいけません。十分にパターンを温めて半田を付けないと何時まで経っても付かないばかりか、部品を破損させたり、接触不良の原因になったりします。

① 半田ゴテの電源を入れ、十分に熱くなるまでコテ台に置きます。
② 半田ゴテの先端部を濡れたスポンジか、金たわしでこさぎ、酸化物(汚れ)を落とします。
③ 半田を流す前に、半田ゴテを十分にくっつけて温めます。
④ 十分に温まった頃に半田を付けて溶かします。
⑤ 半田が溶けたら、パターン面と部品の足の間に半田が流れるのを確認して半田ゴテを離します。

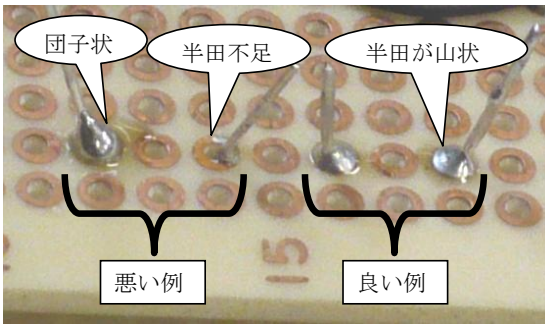


図５　半田付けの良い例，悪い例

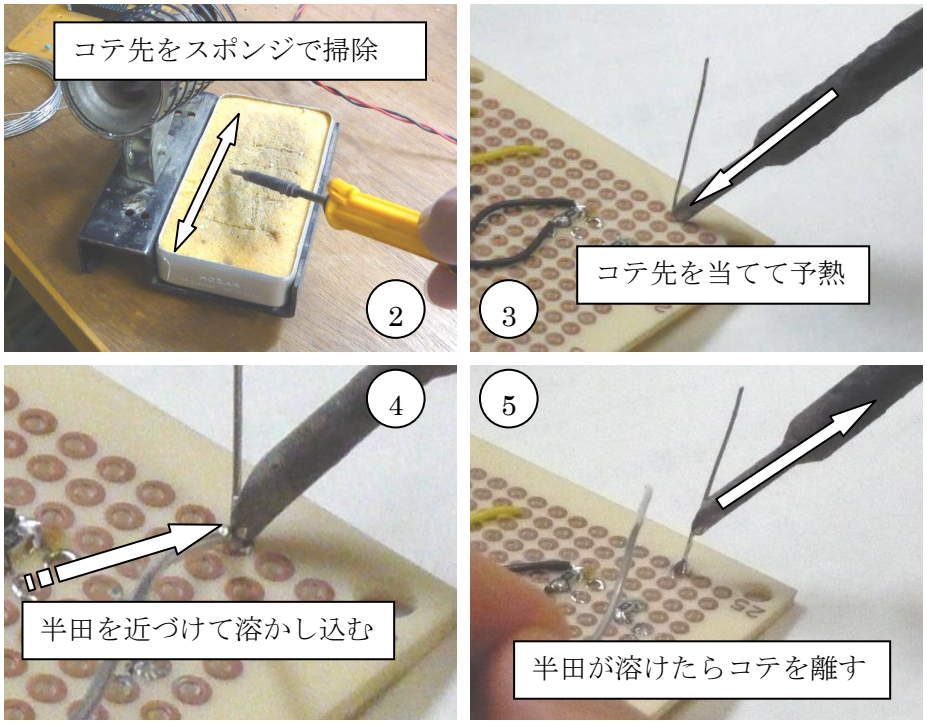


図６　半田付けの手順（番号は説明と一致）

ホットボンドの使い方

ホットボンドはコテと同じように電源を入れるとヒーターから発熱します。その熱で中の樹脂を溶かしてしまいます。溶けた樹脂は高温ですので触れないでください。使い方は、ホットボンドのトリガーを引いて樹脂を出し、目的のものを接着します。

● ニッパによる、足の切断時のテクニック（足を飛ばさない）

部品を半田付けした後は裏に出た余計な足を切断する必要があります。切断する時は、切断後の足が飛ばない様に指で押さえてください。足が飛ぶと、近くの人が危険ですし、掃除(そうじ)が大変です。

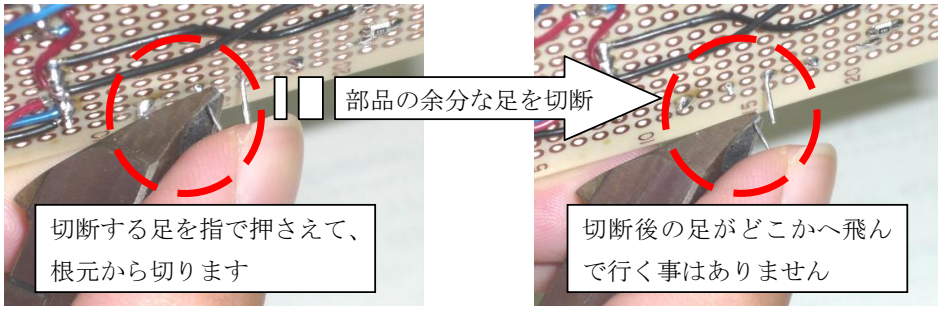


図 7　切断した後、足が飛んで行かない安全な切断方法

4.　わたしのオルゴールの製作

4.1.　部品を基板に半田付けする

部品を組み立てて、電子オルゴールを作ります。

4.1.1.　部品の説明

部品の一覧を図 8 に示します。部品の内、接続するのに向きがあるのは①電池②電池ケース③発光ダイオード④ワンチップマイコン(PIC マイコン) ⑤IC ソケットの 5 点です。部品の取り付けには十分注意してください。

図 8　使用する部品一覧

(ア) **プリント基板**：紙フェノール樹脂の裏に銅パターンが付いています。銅は電気を通す導体です。工作に使用する基板は、八代高専の手作りです❤。

(イ) **単三電池**　　：電池電圧 1.5V を 2 つ使って、電子回路を 3.0V で動作させます。

(ウ) **タクトスイッチ**：スイッチの一種で、押すとスイッチが入り、電流を流すことが出来ます。

(エ) **半固定抵抗**　：可変抵抗とも言います。抵抗値が調整によって 0～100ｋΩまで変化します。

(オ) **スピーカ**　　：スピーカの一種で、本当は圧電スピーカと言います。圧電素子が使われています。

(カ) **電池ケース**　：単三電池 2 個直列専用の電池ケースです。電源スイッチ付。

(キ) **発光ダイオード**：LED とも言います。電球や蛍光灯とは全然違う原理で光ります。使うのは黄色です。青色発光ダイオードを発明した人は会社から 20 億円を貰(もら)った事でも有名です。極性があるので向きを間違えないようにしてください。足の長いほうがプラス側です。

(ク) **ワンチップマイコン**：使うのは PIC12F675 と言うマイコンです。マイコンとはマイクロコンピュータの略で、小さなパソコンっぽい物です。向きがあるので注意してください。

(ケ) **炭素抵抗**　　：抵抗器の一種です。電流が流れにくい性質があり、決まった抵抗値の物が製造されています。極性はありません。

(コ) **IC ソケット**　：PIC12F675 マイコンを差し込むソケットです。マイコンは頻繁(ひんぱん)にプログラムを書き換えるので抜き差しできる様にソケットを使って接続します。マイコンの向きに合わせます。

4.1.2.　部品の組み立て

部品の取り付け順番は、取り付けた状態で頭の低い順から取り付けるという原則があります。

① 1/4W 炭素抵抗を着けます。発光ダイオード(LED)と抵抗はセットにしているので、他の抵抗を着けてしまわない様に気を付けてください。余った足はニッパで切断します。

② IC ソケットを着けます。図 9 の上方向と欠けた方向を合わせてください。**方向に気を付けて**ください。

③ 半固定抵抗を着けます。

④ 発光ダイオード(LED)を**方向に気をつけて**取り付けます。IC 側が短い脚（カソード）です。

⑤ タクトスイッチを取り付けます。奥までしっかり挿してください。

⑥ 圧電スピーカを着けます。もし、電線が長い場合は切断してください。切断する場合は、近くのスタッフに言ってください。

⑦ 電池ケースを取り付けます。＋V と書いてあるほうに赤い線(プラス極)を取り付けてください。

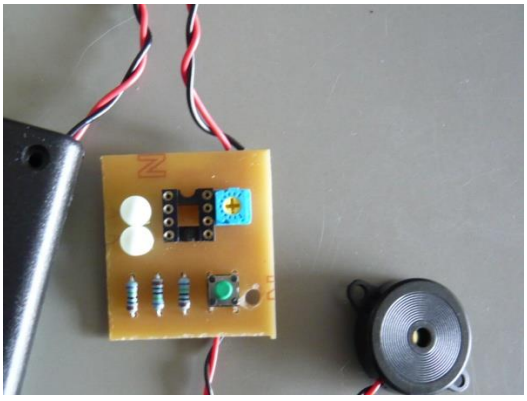


図 9　部品の配置(おもて)

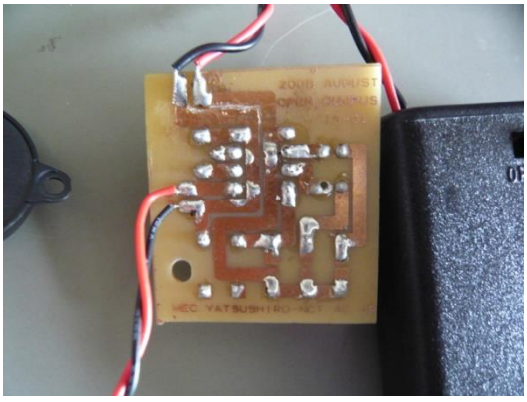


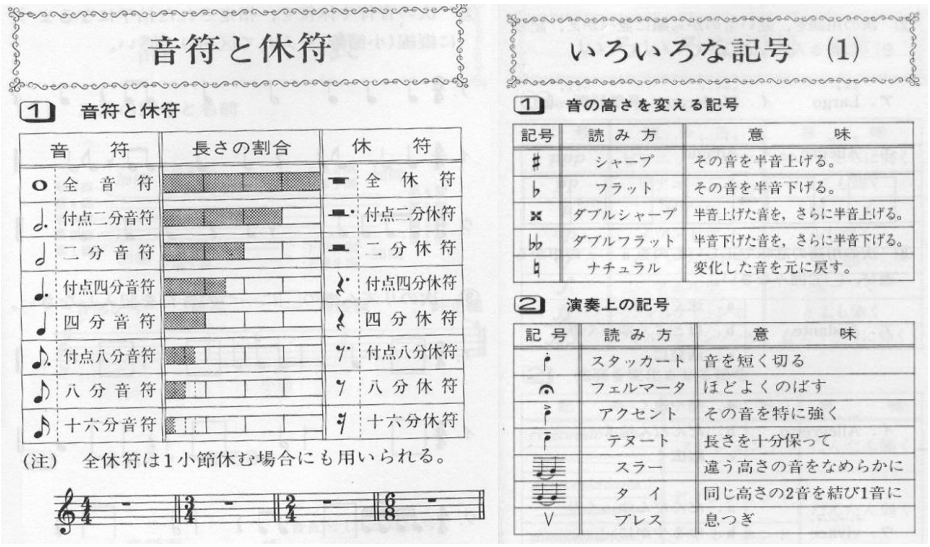
図 10　部品の配置(うら)

4.2.　音符をエクセルに入力する

4.2.1.　音符の読み方

下に示す表 1 を参考に音符を読みます。エクセルへ入力の際は、“音の長さ”という列に全音符(休符)を 1、二分音符(休符)を 2、四分音符(休符)を 4、八分音符(休符)を 8、十六分音符(休符)を 16 として代入してください。また、＃（シャープ）や♭（フラット）などの記号のエクセルへの入力は、“半音”という列で指定してください。

表 1　音符・休符と記号の読み方



音符と休符

1 音符と休符

音符	長さの割合	休符
全音符		全休符
付点二分音符		付点二分休符
二分音符		二分休符
付点四分音符		付点四分休符
四分音符		四分休符
付点八分音符		付点八分休符
八分音符		八分休符
十六分音符		十六分休符

(注)　全休符は1小節休む場合にも用いられる。

いろいろな記号　(1)

1 音の高さを変える記号

記号	読み方	意味
♯	シャープ	その音を半音上げる。
♭	フラット	その音を半音下げる。
𝄪	ダブルシャープ	半音上げた音を，さらに半音上げる。
𝄫	ダブルフラット	半音下げた音を，さらに半音下げる。
♮	ナチュラル	変化した音を元に戻す。

2 演奏上の記号

記号	読み方	意味
	スタッカート	音を短く切る
	フェルマータ	ほどよくのばす
	アクセント	その音を特に強く
	テヌート	長さを十分保って
	スラー	違う高さの音をなめらかに
	タイ	同じ高さの2音を結び1音に
V	ブレス	息つぎ

4.2.2.　エクセルを使った音楽データの作成

楽譜から、音符の音階と音の長さを読み取り、図 11 に示すエクセルの表へ入力します。もし、他の人がパソコン(以下 PC)を使用していて、使えない時は印刷されたシートへ手書きで記入しておくと良いと思います。

では、図 12 に示す楽譜を例に、データの入力方法を説明したいと思います。データの入力手順は、次の通りです。

(ア)　先ず、音階を読み取ります。最初の「ド」は音階が“ド”で、音の長さが 4 分音符です。

(イ)　読み取った音程に合わせて図 11 中の①枠の中に入力していきます。音階の入力は図 13 に示す様に行います。ここでは、「ド 0」を選びます。もし、♭や＃があった場合は半音欄でそれらを選びます。

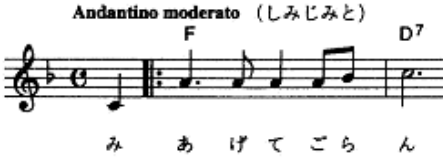


図 12　楽譜の例

(ウ)　次に、音の長さを図 11 中の②枠の中に入力します。最初の「ド」は 4 分音符ですので、同様にして 4 を入力します。もし、付点がつく場合は付点欄で付点を選んでください。

(エ)　最初の「ド」を入力し終わったら、次は「ソ」です。これは 4 分音符ですが、付点が付いている事に注意してください。入力の手順は最初の「ド」の時と同じです。手順の（ア）に戻って、すべての音程を入力し終わるまで繰り返しこの作業を行います。

(オ)　全ての入力が終わったら、音階欄の最後に「おしまい」を選択してください。

(カ)　図 14 に示す様に、「書き込みデータ作成」ボタンを押して、HEX ファイルを作ります。

(キ)　図 15 の様に HEX ファイルが出来た事を確かめます。

エクセルの表へのデータ入力手順はこれで以上です。分らない事があったら、スタッフへ聞いてください。

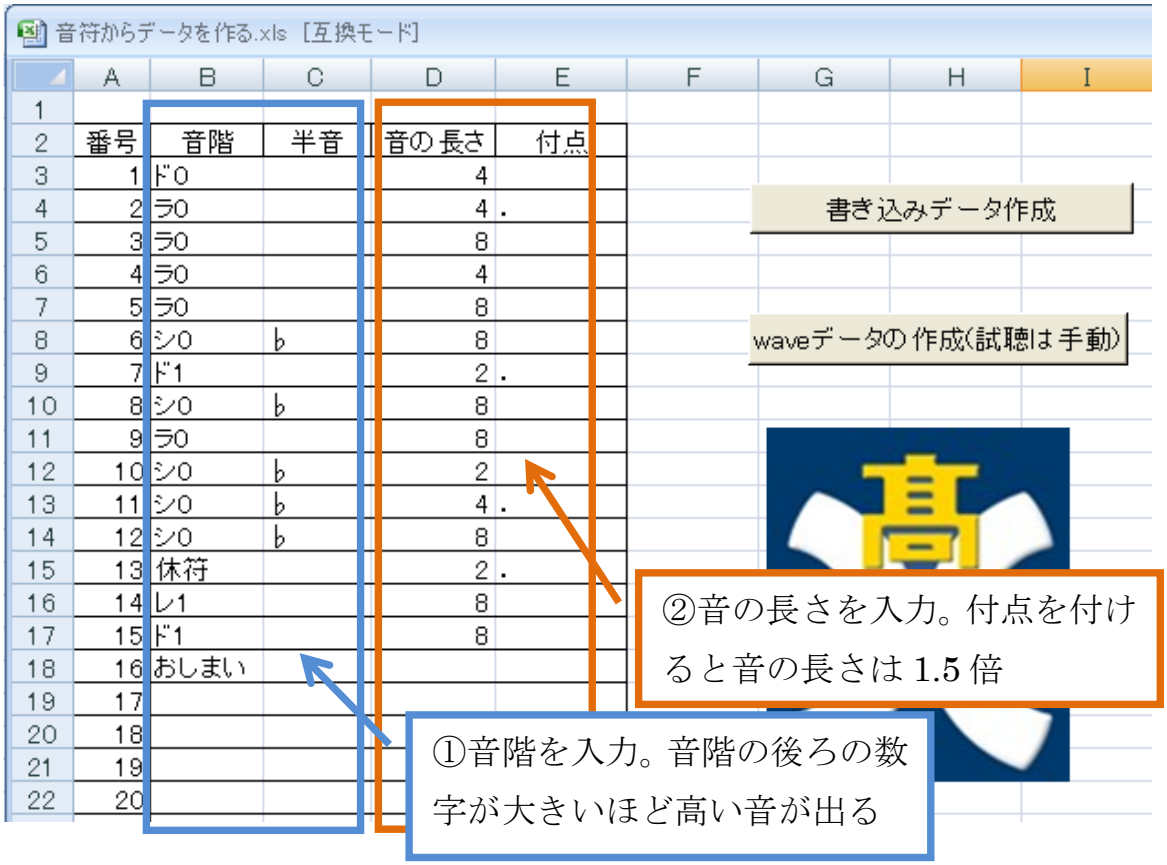


図 11　音符入力シート

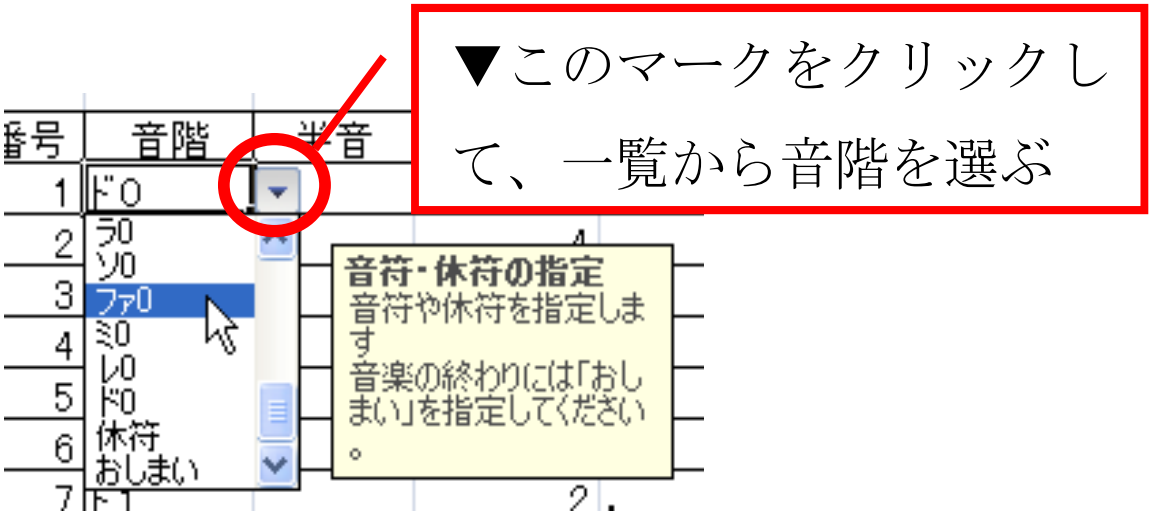


図 13　音階の入力。音の長さも同じ

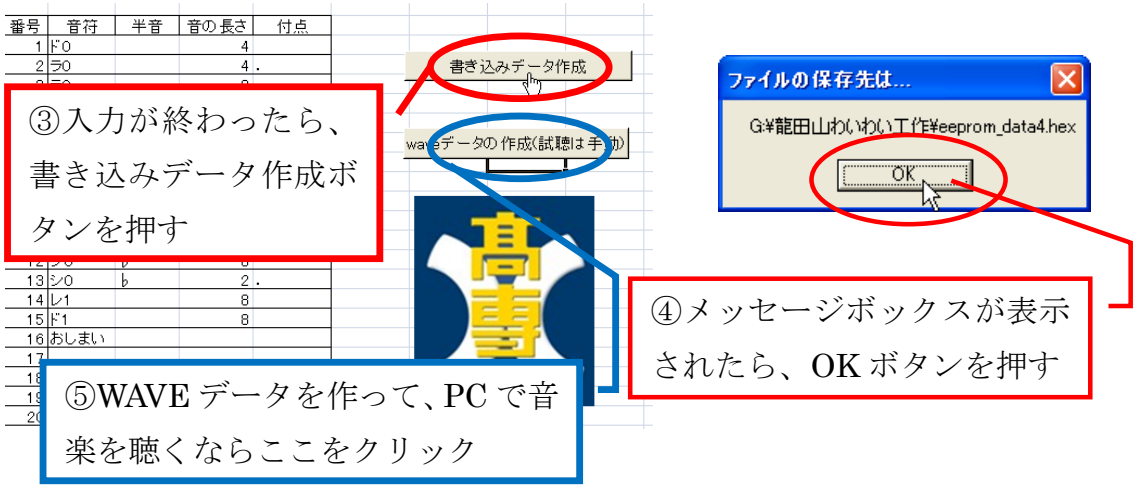


図 14　データ作成ボタンで HEX ファイルを作る

図 15　WAVE ファイルを作っておくと、PC で試聴できる

4.3.　作ったデータを PIC マイコンへ書き込む（不揮発性メモリへのデータ入力）

では、作った HEX ファイルを PIC マイコンへ書き込んでみましょう。

(ア) PIC マイコンを PIC ライターボードへセットし、パソコンと接続します。

(イ) 次に、Windows メニューのプログラム一覧から[Akizuki]→[PIC programmerV4]を起動させてください。このソフトの外観は図 16 の様になっているはずです。PIC の EEPROM へ作った HEX ファイルを書き込む手順は以下の通りです。

① 作った HEX ファイルを読み込みます。図 16 中の指示されたボタンをクリックして、ファイルを選択してください。

② 次に、「デバイス選択」です。今回使用するのは PIC12F675 ですので「PIC12F675」を選択してください。

③ 右上にある「拡張機能」ボタンを押して、図 16 中の右下のウィンドウを開きます。

④ 「拡張プログラミング機能」ウィンドウが開いたら、図 16 に示す様に下側の「プログラム」ボタンをクリックしてください。

⑤　うまくいけば PIC の EEPROM（不揮発性メモリ）へ作った音楽データが書き込まれます。旨く行ったかどうかは、一応メッセージが出ますが、動かしてみるのが一番です。

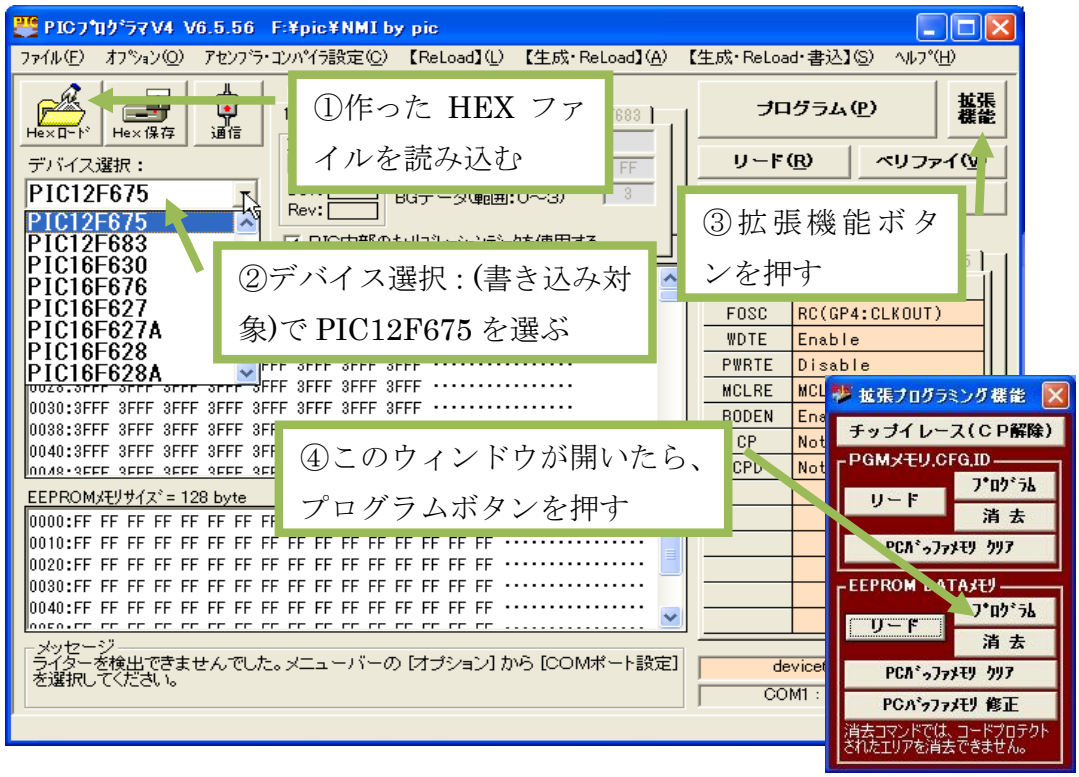


図 16　PIC programmerV4 を使った、PIC の EEPROM へ HEX ファイルを書き込む手順

4.4.　さっそく動かしてみよう

作った HEX ファイルを PIC の EEPROM へ書き込んだら、さっそく動かしてみましょう。

(ア) まず、データを書き込んだ PIC マイコンを自分の作った回路基板へ取り付けます。

(イ) 取り付けるにはコツがあり、IC ソケットに PIC マイコンを**深く刺しすぎない事**です。

(ウ) PIC マイコンには**極性と言って向きがあります**。差し込む方向を間違えないようによく確認してください。もし、まだ回路基板を作っていない人はテスト用の基板か友達の物で試しても良いです。

(エ) 電池を取り付ける前には、必ず、**スタッフを呼んで、一通りのチェックを行ってください**。

(オ) 単三電池を電池ボックスの中へ入れます。スイッチを入れて、音楽が流れれば OK です。もし、**動かない場合は素早く電源を切り**、電池を取り外した状態でスタッフのチェックをもう一度受けてください。

4.5. おしまいに

今日作った電子オルゴールは、音楽を書き換えるために ①エクセル（Microsoft Office Excel）と ②秋月電子通商製の PIC ライターボード（参考文献 3） ③ ②に付属のソフトウェアである PIC programmerV4 ④音符データを入力する「音符からデータを作る.xls」の 4 点が必要です。もし、音楽を書き換えたくても以上の①～③を用意できない人は、八代高専までデータを送信してください。送り先は次の通りです。

〒866-8501 熊本県八代市平山新町 2627
八代工業高等専門学校 機械電気工学科
入江 博樹（いりえ ひろき）
e-mail：irie@as.yatsushiro-nct.ac.jp

5. 参考資料

5.1. PIC マイコンについて

- **PIC は周辺装置を制御するために開発された IC**

ＰＩＣ(Peripheral Interface Controller：周辺機器接続制御用ＩＣ)とは元々メインのＣＰＵの機能を分散して周辺機器の制御を行うために開発されたＩＣです。PIC には CPU、メモリ（RAM、ROM)、I/O などが 1 チップに収められており、ワンチップマイコンとも言います。人間に例えると頭脳がメインＣＰＵで自律神経に相当する部分をＰＩＣが行うと言うことでしょうか。

図 17 PIC16F84A マイコン

PIC は ROM に書き込まれたプログラムにより制御されます。回路構成が簡単であり、安価なので電子工作を行う人の間で人気があります。趣味でマイコンをやっている人にとってはマイコン＝micro computer ≅ My con＝“わたしのコンピューター”と言って、自分自身で使える最小のコンピューターです。

- **PIC は小さなコンピューター**

ＰＩＣにもＣＰＵと同じように演算処理機能があり、メモリーも持っていて、ソフトウェアで制御します。

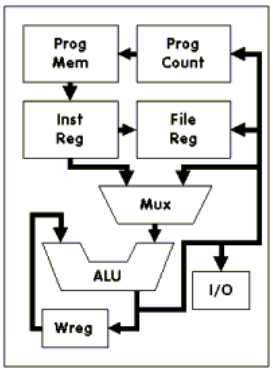


図 18 PIC マイコンの構造

しかし、処理能力、メモリ容量は大きくはありません。ＰＩＣの種類によって違いますが、最大動作クロック周波数が２０ＭＨｚ位、プログラムを書き込むメモリーは１Ｋワードから４Ｋワード位です。

クロック周波数はプログラムを読んで命令を解釈し、処理をする速さに関係します。クロック周波数だけでは処理能力が高い低いは論じられません。処理部分の構造（アーキテクチャ）によって変わり

ます。同じ構造であるなら、クロック周波数が高い方が処理能力は高くなります。
プログラムメモリーの容量にワードという言葉を使いましたが、１つの命令（インストラクション）を１ワードと言っています。メモリーの容量を表すのにバイトという単位を良く使います。１バイトというのは８ビットのことを表す言葉です。ビットと言うのは０か１を表す最小単位です。良く使われるＰＩＣに PIC16F84A がありますが、この場合１つの命令は１４ビットで構成されています。１Ｋワードをビットに換算すると 1 x 1,024 x 14 = 14,336 ビットになります。これをバイトに換算すると 14,336/(8 x 1,024) = 1.75Ｋバイトになります。バイトに換算する必要はないのですが、良く見慣れたバイトで感じを掴(つか)むために換算してみました。

余談ですが、メモリ容量で１Ｋバイト＝１，０２４バイト、１Ｍバイト＝１，０２４Ｋバイト、１Ｇバイト＝１，０２４Ｍバイトです。１０００倍ではありません。これは２進数の計算で２の１０乗が１，０２４になるところからきています。

PIC を使うとコンパクトな回路が出来る

ＰＩＣの便利なところは演算機能部、メモリ、入出力部などが一つのＩＣに組み込まれている点です。性能、機能は限定されていますが、いろいろなＩＣを組み合わせなくてもＰＩＣだけで制御部を構成できるので回路をコンパクトに作ることができます。

（この節の文章及び図は http://www.hobby-elec.org/pic1.htm と Wikipedia から参照しました。）

5.2. 音の出る仕組み

今回、スピーカから音楽を流してみましたが、一体どうやって音が出るのでしょうか?
そもそも音とは、空気の蜜と疎が連続して空間を伝わる振動です。言い換えれば、かなりの速度で気圧が高くなったり、低くなる状態が伝わっているのです。太鼓を叩けば、太鼓の皮の部分が震えて空気を押したり引いたりします。それが空気中に気圧の振動を引き起こし、空間を伝わって私たちの耳にある鼓膜を震わせます。スピーカが音を出す時にも同じ現象が起こっています。

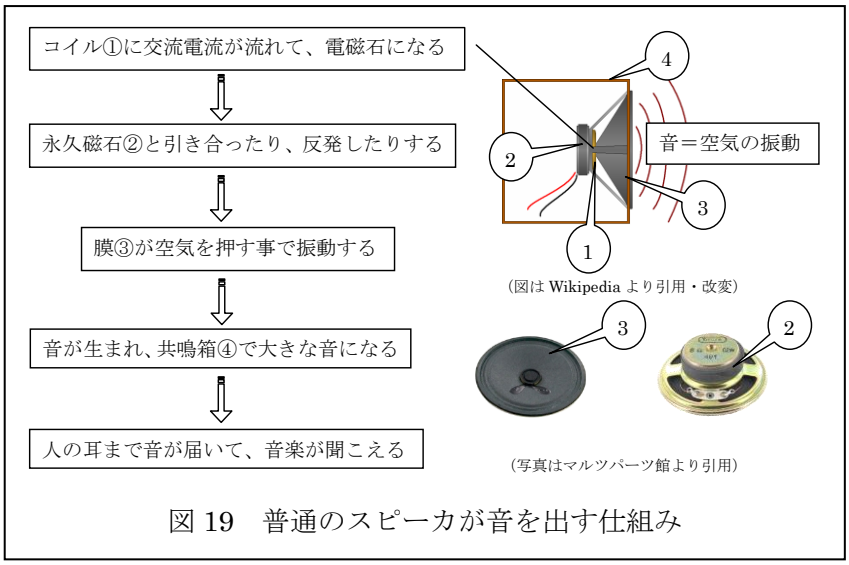


図 19　普通のスピーカが音を出す仕組み

図 19 は普通のスピーカが音を出す原理を示しています。普通のスピーカは、電磁石が磁石と引き合ったり反発する性質を利用しています。電磁石がスピーカの膜に取り付けてあるために電磁石の振動は容易に空気中に伝わっていきます。このように磁石を用いたスピーカは色々な音を出せます。

今回の工作に使用したのは圧電スピーカという物です。図 20 に動作原理を示します。これは圧電素子の性質を利用しています。圧電素子とは電圧をかけると変形する性質があります。時計に使われている水晶発振子も圧電素子の一種です。この変形によって空気を振動させて音が出ます。複雑な音は苦手ですが消費電流が少ない点で優れています。

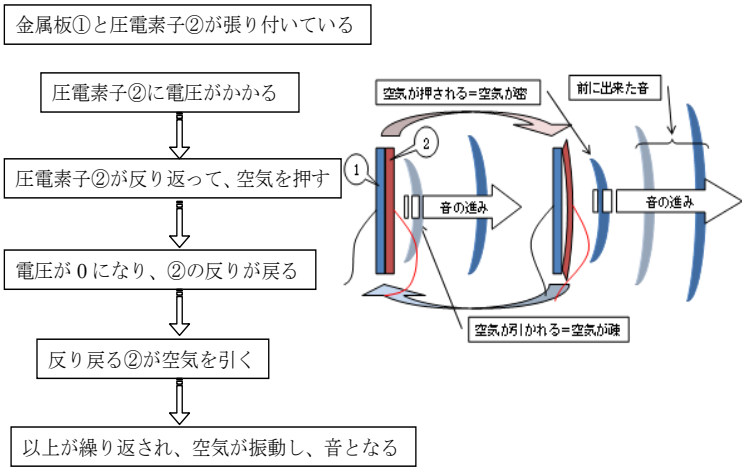


図 20　圧電スピーカが音を出す仕組み

PIC マイコンは圧電スピーカに図 21 の様に電圧を加えています。初めは 0V ですが、ある時間になると電圧が急に上がり、一定の電圧になります。また、一定時間が経(た)つと 0V になる・・・と言う事を繰り返します。この様な電圧の形を方形波(ほうけいは)又は矩形波(くけいは)と言います。方形波は図 22 に示す様に幾(いく)つもの単純な波が重なって出来ています。圧電スピーカは幾つかの波の内で最も大きな波の音を出しています。

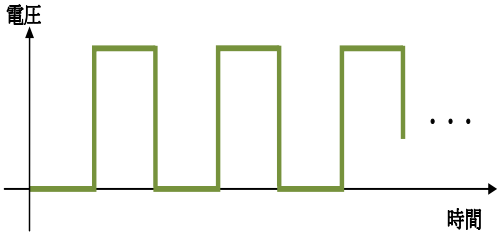


図 21　方形波(矩形波)の形

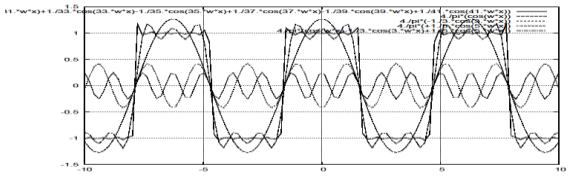


図 22　方形波は単振動が重なって出来ている

5.3.　その他の参考文献(インターネットに接続されたパソコンで調べよう)

(1)
(2)おもしろい PIC マイコン　PIC12F675 を使いこなす，オーム社，平成 16 年初版，¥2300.
(3)小ロット生産者とともに歩むー秋月電子通商ー，http://akizukidenshi.com/，2007.
備考：PIC12F675 のプログラムを変更するためには、次の 2 点が必要です。
①通販コード[K-31]，商品名「ＡＫＩ－ＰＩＣプログラマー　Ｖｅｒ．３．５　キット」
②通販コード[K-200]，商品名「ＰＩＣプログラマーＶｅｒ．４　バージョンアップキット」
(4)オルゴールの歴史について，http://www.sora-aonami.com/orgelhis01.html，2007.

6.　付録

6.1.　音楽の楽譜サンプル

音楽のサンプルとして、いくつか楽譜を用意しました。好きな音楽をマイコンに入れて遊んでみましょう。

6.2.　電子オルゴールの回路図

図 23 に電子オルゴールの回路図を示します。電流が流れすぎないように工夫しています。

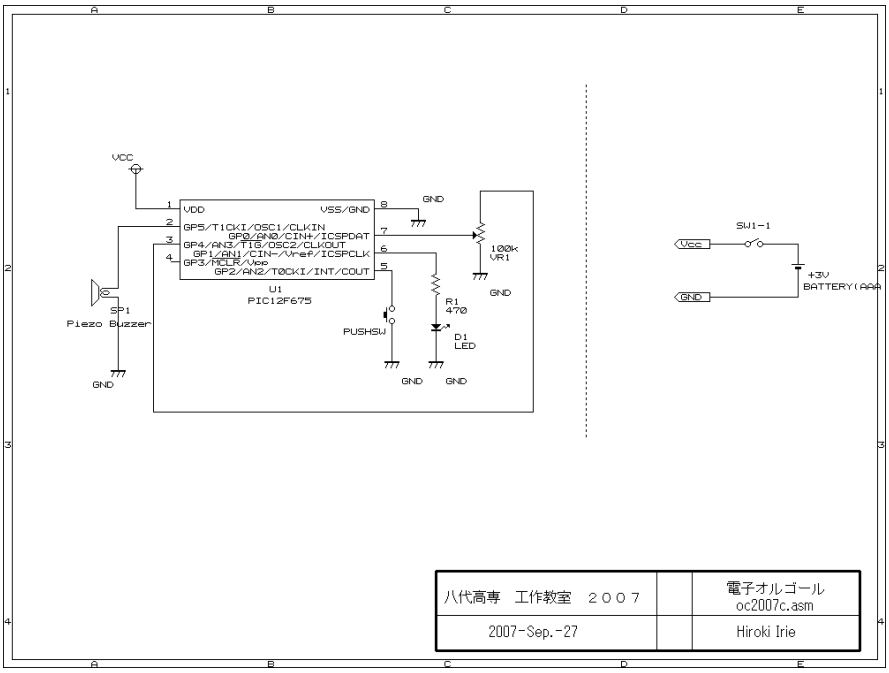


図 23　電子オルゴールの回路図

6.3.　マイコンのプログラムソース

電子オルゴールの PIC マイコンのプログラムです。このプログラムはアセンブラ言語という機械への命令語で出来ています。興味のある人は参考文献（2）を購入（こうにゅう）して、自分の思いのままに動作する PIC マイコンに改造しても良いと思います。

```
;===================================================================
;地域連携　製作用　電子オルゴール
;入江博樹 2007-June-23
; OC2007c.asm　　for PIC12F675　　オープンキャンパス２００７体験授業用　電子オルゴール
; GP5 に圧電ブザー（他励式）を接続し、「It's Small World」を演奏する。
; GP4 には LED を接続する。
;oc2007a.asm からの改良点は、次の通り
; (1) 演奏後にスリープモードに入る。　GP2(INT)に ON-OFF 入力があれば、スリープから復帰
; (2) GP0 に　可変抵抗器でアナログ電圧（0～5V)を入力、８ビットの分解能でデータを取得
;　　曲のテンポを可変できるようにした。　250 の時は通常の速度。
;　　数字が小さくなると速くなる。0 では 256 と同じとなり遅くなる。
; (3)GP1 にＬＥＤを付けた。電源を入れた直後とスリープから復帰後に、１回だけ点滅
;===================================================================
============

;ここから、プリアンブル部

        LIST    P=12F675
        INCLUDE P12F675.INC

CB      =       _CPD_OFF
CB      &=      _CP_OFF
CB      &=      _BODEN_ON
CB      &=      _MCLRE_OFF
CB      &=      _PWRTE_ON
CB      &=      _WDT_OFF
CB      &=      _INTRC_OSC_NOCLKOUT


        __CONFIG    CB
        __IDLOCS    H'0100'  ;バージョン 1.00

SOUND_TGL   EQU     B'00100000'     ;ビットマスク スピーカを接続するところを
１にする
LED_TGL     EQU     B'00000010'     ;ビットマスク LED を接
続するところを１にする

        CBLOCK  H'20'
                W_TEMP
                S_TEMP          ;割込処理時に、コンテキストを保存
                GPIO_OUT
                T0IF_CNT
                TONE
                ONPU
                CNT0
                CNT1
                CNT2
                TIM_AD
                CNT_ROM
        ENDC


RA0     EQU     D'114'      ;440Hz　A　ラ
SI0     EQU     D'129'
DO      EQU     D'136'      ;　　ド
RE      EQU     D'149'
MI      EQU     D'161'
FA      EQU     D'166'
FA_S    EQU     D'171'      ;F# ト長調ときは、Ｆ＃を使う
SO      EQU     D'176'
RA      EQU     D'185'      ;880Hz　A ラ
SI      EQU     D'193'
DO1     EQU     D'196'
RE1     EQU     D'203'

;ここまでがプリアンブル部

;これ以降が、プログラムとして、メモリに記憶される。
;ここから、I/O の初期化部と割り込み処理部

;-----------------------------------------
;起動時のプログラムを記述
        ORG     H'0'
        CALL    OSCCALIB
        CALL    IOINIT
        CALL    INTINIT
        GOTO    MAIN        ;メインプログラムが書いてあるルーチン
へジャンプ

;-----------------------------------------
```

```
;割り込み設定時のプログラムを記述
                ORG             H'4'                    ;04H 番地は、PIC の割り込みベクタになる
;W レジスタと STATUS レジスタの値をスタックへ逃がす。
                MOVWF           W_TEMP
                MOVF            STATUS,W
                CLRF            STATUS
                MOVWF           S_TEMP

;タイマー０による割り込みが発生した場合の処理
ISR_TMR0:
                BTFSS           INTCON,T0IF
                GOTO            ISR_TMR0_END
                BCF             INTCON,T0IF     ;割り込みフラグをクリア

                MOVF            GPIO,W
                MOVWF           GPIO_OUT
                MOVLW           SOUND_TGL       ;W レジスタに、ビットマスクを読み込む
                XORWF           GPIO_OUT,F      ;割り込みが入るごとにビットを反転

                CALL            IOPORT          ;値を出力

                MOVF            TONE,W          ;音階のデータ再設定
                MOVWF           TMR0            ;タイマ０に値をセット
ISR_TMR0_END:;ここまで

;INT ピン割り込み
ISR_INT:
                BTFSS           INTCON,INTF
                GOTO            ISR_INT_END
                BCF             INTCON,INTF
;(kokoni timer warikomi no puroguramu wo kaku)
ISR_INT_END:

;スタックから W レジスタと STATUS レジスタの値を戻す
                MOVF            S_TEMP,W
                MOVWF           STATUS          ;STATUS に値を戻す
                SWAPF           W_TEMP,F        ;W に値を戻す。
                SWAPF           W_TEMP,W        ;レジスタに変化が内容にするために、
SWAP を利用する

                RETFIE          ;割り込みによるサブルーチンからリターン

;PIC の割り込みサブルーチンがかいあるのは、ここまで。

;ここまでが、I/O の初期化部と割り込み処理部

;ここから、メインルーチン部
;-----------------------------------------------
;電源を入れた後は、03H 番地からここにジャンプしてくる
MAIN:
                CLRF            GPIO_OUT
                CLRF            T0IF_CNT
                CLRF            GPIO
                CALL            AD_DATA         ;可変抵抗の値で、タイマーの値を決める

MAIN_LOOP:

                CALL            LED_ONOFF025s ;GP2 に接続された LED を一回だけ点滅
                CALL            ONGAKU2         ; 音楽を演奏するサブルーチンへ
                CALL            LED_ONOFF025s

                BCF             INTCON,T0IE     ;TMR0 割り込みを不許可

                CALL SLEEP_IN
                SLEEP
                NOP
                CALL SLEEP_OUT

                BSF             INTCON,T0IE     ;TMR0 割り込みを許可

                ;ループを実行
                GOTO            MAIN_LOOP


;ここまでが、メインルーチン部

;ここから、サブルーチン部

SLEEP_IN
                BSF             STATUS,RP0
                MOVLW           H'FF'           ; 入力 GP0, GP2
                MOVWF           TRISIO                          ;       出   力       GP1
GP3,GP4,GP5
                                MOVLW           B'00000000'     ;
                                MOVWF           ANSEL           ;A/D ON, GP0

                BCF             STATUS,RP0
                RETURN

SLEEP_OUT
                BSF             STATUS,RP0
                MOVLW           B'00001101' ; 入力 GP0, GP2, GP3,
                MOVWF           TRISIO                          ;   出力   GP1 GP4,GP5

                                MOVLW           B'00000001'     ;
                                MOVWF           ANSEL           ;A/D ON, GP0

                                BCF             OPTION_REG,NOT_GPPU ;W P U を使う
とき 0
                                BSF             OPTION_REG,INTEDG  ;INT ピンの立ち
上がりを検出の時 1

                                MOVLW           B'00000100'
                                MOVWF           WPU             ;GP2 をウィークプルア
ップ


                BCF             STATUS,RP0
                RETURN

;-----------------------------------------------
; GPIO_OUT の値を I/O ポートにデータを出力する
IOPORT:
                MOVF            GPIO_OUT,W
                MOVWF           GPIO
                RETURN

;----------------------  内蔵発振子の設定
OSCCALIB:
                CALL            H'3FF'
                BSF                             STATUS,RP0
                MOVWF           OSCCAL
                BCF                             STATUS,RP0
                RETURN

;-----------------------------------------------
;I/O ポートを初期化する
IOINIT:
                BSF                             STATUS,RP0
                                MOVLW           B'00001101'     ; 入力 GP0, GP2,GP3
                                MOVWF           TRISIO          ;   出力   GP1,GP4,GP5

                                MOVLW           B'11010010'
                                ANDWF           OPTION_REG,F

                                MOVLW           B'00000111'     ;コンパレータＯＦＦ
                                MOVWF           CMCON

                                MOVLW           B'00000001'     ;
                                MOVWF           ANSEL           ;A/D ON, GP0

                                BCF             OPTION_REG,NOT_GPPU ;W P U を使う
とき 0
                                BSF             OPTION_REG,INTEDG  ;INT ピンの立ち
上がりを検出の時 1

                                MOVLW           B'00000100'
                                MOVWF           WPU             ;GP2 をウィークプルア
ップ

                BCF                             STATUS,RP0

                CLRF            TMR0                            ;タイマーを初期化

                BSF                             INTCON,T0IE     ;TMR0 割り込みを許可

                BSF                             INTCON,GIE      ;割込を可能にする

                RETURN

;----------------------  割り込みの設定
INTINIT:
                CLRF            INTCON
                CLRF            PIE1
                CLRF            PIR1
                                                                ;BSF で割り込み許可
                BSF                             INTCON,T0IE     ;タイマー 0 オーバフロ
ー割り込み
                BSF                             INTCON,INTE     ;INT ピン入力エッジ割
り込み
                BCF                             INTCON,GPIE     ;GPIO ポート入力レベル
変化割り込み

                BCF                             INTCON,PEIE     ;PIE 1 レジスタによる
割り込み設定
                BCF                             PIE1,EEIE       ;EEPROM 書き込み終了
割り込み
                BCF                             PIE1,ADIE       ;A/D 変換終了割り込み
                BCF                             PIE1,CMIE       ;コンパレータ出力割り
込み
                BCF                             PIE1,TMR1IE     ;タイマー 1 オーバフロ
ー割り込み

                BCF                             INTCON,GPIF     ;GPIO 入力データ変化割
り込みを初期化
                BCF                             INTCON,T0IF     ;タイマー 0 割り込みを
初期化
                BCF                             INTCON,INTF     ;INT ピン割り込みを初
期化

                BSF                             INTCON,GIE      ;割り込みを許可
                RETURN


;-----------------------------------------------
;LED   を   1 回だけ、点滅させる。
LED_ONOFF025s:
```

```
                BCF                             INTCON,T0IE     ;TMR0 割り込みを不許
可
                BCF                             INTCON,GIE      ;割込を不可能にする

                BSF                             GPIO,1          ;ＬＥＤを点灯
                CALL            TIM025s
                BCF                             GPIO,1          ;ＬＥＤを消灯

                BSF                             INTCON,T0IE     ;TMR0 割り込みを許可

                BSF                             INTCON,GIE      ;割込を可能にする
                RETURN
;-----------------------------------------------
;LED　を　呼び出すごとに点滅させる。
LED_ONOFF:
                MOVF            GPIO,W          ;GPIO の値を W レジスタへ読み込む
                MOVWF           GPIO_OUT        ;GPIO OUT へ値を移す
                MOVLW           LED_TGL         ;W レジスタに、ビットマスクを読み込む
                XORWF           GPIO_OUT,F      ;LED がつながっている場所のビットを反
転

                CALL            IOPORT          ;値を出力

                RETURN

;-----------------------------------------------
; timer Subroutine
;
;-----------------------------------------------
TIM2ms:
                MOVLW           D'250'
                MOVWF           CNT0
TIM02mL:
                GOTO $+1
                GOTO $+1
                NOP
                DECFSZ          CNT0,F
                GOTO            TIM02mL
    RETURN

;----------------------------------------- TIM_AD が 250 のとき 2ms
TIMADms:
                MOVF            TIM_AD,W
                MOVWF           CNT0
TIMADmsL:
                GOTO $+1
                GOTO $+1
                NOP
                DECFSZ          CNT0,F
                GOTO            TIMADmsL
    RETURN

;-----------------------------------------0.255sec ~ 0.001sec
TIM_ADs:
                MOVLW           D'125'
                MOVWF           CNT1
TIM_ADsL:
                CALL            TIMADms         ;A/D が 250 の時 2ms
                DECFSZ          CNT1,F
                GOTO            TIM_ADsL
    RETURN

;-----------------------------------------
TIM50ms:
                MOVLW           D'25'
                MOVWF           CNT1
TIM50mL:
                CALL            TIM2ms
                DECFSZ          CNT1,F
                GOTO            TIM50mL
    RETURN

;-----------------------------------------0.25sec
TIM025s:
                MOVLW           D'125'
                MOVWF           CNT1
TIM025L:
                CALL            TIM2ms
                DECFSZ          CNT1,F
                GOTO            TIM025L
    RETURN

;-----------------------------------------0.5sec
TIM05s:
                MOVLW           D'250'
                MOVWF           CNT2
TIM05L:
                CALL            TIM2ms
                DECFSZ          CNT2,F
                GOTO            TIM05L
    RETURN

;-----------------------------------------6μsec
TIM6us:
                GOTO $+1
                GOTO $+1
                GOTO $+1
    RETURN

;-----------------------------------------
:A/D データ　読み込み
AD_DATA:
                BSF             GPIO,4          ;GP4 を H レベルにする　(A/D 用の抵抗に
給電)
                MOVLW           B'00000001'
                MOVWF           ADCON0          ;A/D 変換器の動作を開始させる。


                                                ;A/D を設定した後、実際に動作を開始する
まで、しばらく時間をつぶす
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1
                GOTO            $+1

:A/D 変換の動作を開始、GP0 からデータを取り込む
                BSF                             ADCON0,GO

:A/D 変換が完了するまでは、時間がかかるので、完了までループする
                BTFSC           ADCON0,NOT_DONE ;A/D 変換が完了したら 0 になる
                GOTO            $-1             ;一つ前に戻る

                MOVF            ADRESH,W        ;A/D　上位 8 ビットのみを W レジスタに取
り出す。
                MOVWF           TIM_AD          ;それを TIM_AD レジスタ(汎用レジスタ)
に格納する。

                BCF             GPIO,4          ;GP4                を L レベルに戻す

                RETURN                          ;元の位置へ戻る。

;ここまでが、サブルーチン部

;-----------------------------------------------
;音楽データ を EEPROM から読み出し、演奏する

ONGAKU2:

                MOVLW           H'00'           ;音楽データが収められているＲＯＭの先頭
番地を EEADR に入れる
                MOVWF           CNT_ROM

ONGAKU_ROM:
                MOVF            CNT_ROM,W
                CALL READ_ROM                   ;２バイトを読み出し、TONE と ONPU に
入れる。　読み出しアドレスを次のバイトに移す
                INCF            CNT_ROM,F
                INCF            CNT_ROM,F

                CALL            LED_ONOFF

                MOVF            ONPU,F          ; ONPU のデータを取り出し、ONPU に戻
す、Ｚフラグが影響を受ける。
                BTFSC           STATUS,Z        ; ゼロだったら、このサブルーチンを終了す
る
                GOTO            END_ONGAKU2

;               MOVWF           TONE            ;音程を入力。　TONE が FF の時は休
符をあらわす
;               MOVWF           ONPU            ;音の長さを入力。　32 分音符 (0.0625 秒)
を基本とする

                CALL            ONPU_KYUFU      ;音を鳴らす。　休符の判定もサブルーチン
で実施　:OTODASHI&KYUFU

                GOTO            ONGAKU_ROM      :EEPROM からのデータの読み込みを続け
る

END_ONGAKU2:
                RETURN                          ;サブルーチンを終了する。


;-----------------------------------------------------
:EEPROM から２バイトのデータを取り出して、TONE と ONPU に代入する。
:EEADR のアドレスを　W レジスタにセットしてからコールする


READ_ROM:
                ;-------------------1 バイト目 (TONE 音の高さ)
                BSF             STATUS,RP0
                MOVWF           EEADR
                BSF             EECON1,RD
                MOVF            EEDATA,W
                BCF             STATUS,RP0
                MOVWF           TONE

                ;-------------------2 バイト目 (ONPU 音のながさ
                BSF             STATUS,RP0
                INCF            EEADR,F
                BSF             EECON1,RD
```

```
                MOVF        EEDATA,W
                BCF         STATUS,RP0
                MOVWF       ONPU

                RETURN

;=========================================================
;---------------------------------------------
;
ONPU_KYUFU:

                INCFSZ      TONE,W          ;もしも TONE が FF だったら　＋１をするとゼロになる
                GOTO        OTODASHI        ;ＦＦ以外だったら、音を出す命令が実行される
                GOTO        KYUFU           ;FF だったら、休符を実行する。
;---------------------------------------------
;
OTODASHI:
                BSF         INTCON,T0IE     ;TMR0 割り込みを許可
                BSF         INTCON,GIE      ;割込を可能にする

OTODASHI_L:                                 ;音の長さだけ､待ち時間を入れる 32 分音符が基準となる
                CALL        ONPU_AD
                DECFSZ      ONPU,F
                GOTO        OTODASHI_L

                CALL        AD_DATA         ;音の長さの基準値をＡ/Ｄ変換から取り込む
                CALL        KYUFU_6ms       ;音の切れをよくするために 6ms 間の休符を入れる
                CALL        LED_ONOFF       ;GP4 に接続された LED を交互に点滅

                RETURN
;---------------------------------------------------------
KYUFU:
                BCF                         INTCON,T0IE     ;TMR0 割り込みを不許可
                BCF                         INTCON,GIE      ;割込を不可能にする
KYUFU_L:
                CALL        ONPU_AD
                DECFSZ      ONPU,F
                GOTO        KYUFU_L

                RETURN

;---------------------------------------------
;
ONPU_AD:
                CALL        TIM_ADs         ;AD に応じて,0.001～0.255 秒間のウェイトを実施
                RETURN
;---------------------------------------------
;
KYUFU_6ms:
                BCF                         INTCON,T0IE     ;TMR0 割り込みを不許可
                BCF                         INTCON,GIE      ;割込を不可能にする
                CALL        TIM2ms
                CALL        TIM2ms
                CALL        TIM2ms

                RETURN

;---------------------------------------------
;
ONPU_2:
                CALL        TIM05s
                CALL        TIM05s
                GOTO        ONPU_END
ONPU_4:
                CALL        TIM05s
                GOTO        ONPU_END
ONPU_8:
                CALL        TIM025s
                GOTO        ONPU_END
ONPU_END:
                BCF                         INTCON,T0IE     ;TMR0 割り込みを不許可
                BCF                         INTCON,GIE      ;割込を不可能にする
                CALL        TIM2ms
                BSF                         INTCON,T0IE     ;TMR0 割り込みを許可

                BSF                         INTCON,GIE      ;割込を可能にする
                RETURN

KYUFU_4:
                BCF                         INTCON,T0IE     ;TMR0 割り込みを不許可
                BCF                         INTCON,GIE      ;割込を不可能にする
                CALL        TIM05s
                BSF                         INTCON,T0IE     ;TMR0 割り込みを許可

                BSF                         INTCON,GIE      ;割込を可能にする
                RETURN

KYUFU_8:
                BCF                         INTCON,T0IE     ;TMR0 割り込みを不許可
                BCF                         INTCON,GIE      ;割込を不可能にする
                CALL        TIM025s
                BSF                         INTCON,T0IE     ;TMR0 割り込みを許可

                BSF                         INTCON,GIE      ;割込を可能にする
                RETURN

KYUFU_32:
                BCF                         INTCON,T0IE     ;TMR0 割り込みを不許可
                BCF                         INTCON,GIE      ;割込を不可能にする
                CALL        TIM50ms
                BSF                         INTCON,T0IE     ;TMR0 割り込みを許可

                BSF                         INTCON,GIE      ;割込を可能にする
;プログラムの最後
                END
```

6.4.　エクセルのプログラムソース

今回、音楽のデータを入力したエクセルのファイルには VBA と言う言語で書かれたプログラムが組み込まれています。たぶん、工作に参加した人で分かる人はいないと思いますが参考までに載せておきます。

VBA プログラミングは Microsoft Excel が使えるパソコンでならできます。

音階と周波数、そしてマイコンの設定値の関係を表したのが表 2 です。「ラ」の 440Hz を基準に決めていきます。

表 2　音階とマイコンのタイマー設定値の関係

階級k	音階	指　数 x =k/12	周波数 f[Hz]=440×2^x	周期 T[sec]=1/f	マイコンのサイクル cycle=T/φ×4	プリスケーラの値 N=cycle/n		タイマー値 四捨五入済 timer = 256−$N_{n=16}$
						8分の1 n=8	16分の1 n=16	
−10	シ	−0.83	246.9	0.0040495	4049.5	506.2	253.1	3
−9	ド	−0.75	261.6	0.0038223	3822.3	477.8	238.9	17
−8		−0.67	277.2	0.0036077	3607.7	451.0	225.5	31
−7	レ	−0.58	293.7	0.0034052	3405.2	425.7	212.8	43
−6		−0.50	311.1	0.0032141	3214.1	401.8	200.9	55
−5	ミ	−0.42	329.6	0.0030337	3033.7	379.2	189.6	66
−4	ファ	−0.33	349.2	0.0028635	2863.5	357.9	179.0	77
−3		−0.25	370.0	0.0027027	2702.7	337.8	168.9	87
−2	ソ	−0.17	392.0	0.0025511	2551.1	318.9	159.4	97
−1		−0.08	415.3	0.0024079	2407.9	301.0	150.5	106
0	ラ	0.00	440.0	0.0022727	2272.7	284.1	142.0	114
1		0.08	466.2	0.0021452	2145.2	268.1	134.1	122
2	シ	0.17	493.9	0.0020248	2024.8	253.1	126.5	129
3	ド	0.25	523.3	0.0019111	1911.1	238.9	119.4	137
4		0.33	554.4	0.0018039	1803.9	225.5	112.7	143
5	レ	0.42	587.3	0.0017026	1702.6	212.8	106.4	150
6		0.50	622.3	0.0016071	1607.1	200.9	100.4	156
7	ミ	0.58	659.3	0.0015169	1516.9	189.6	94.8	161
8	ファ	0.67	698.5	0.0014317	1431.7	179.0	89.5	167
9		0.75	740.0	0.0013514	1351.4	168.9	84.5	172
10	ソ	0.83	784.0	0.0012755	1275.5	159.4	79.7	176
11		0.92	830.6	0.0012039	1203.9	150.5	75.2	181
12	ラ	1.00	880.0	0.0011364	1136.4	142.0	71.0	185
13		1.08	932.3	0.0010726	1072.6	134.1	67.0	189
14	シ	1.17	987.8	0.0010124	1012.4	126.5	63.3	193
15	ド	1.25	1046.5	0.0009556	955.6	119.4	59.7	196
16		1.33	1108.7	0.0009019	901.9	112.7	56.4	200
17	レ	1.42	1174.7	0.0008513	851.3	106.4	53.2	203
18		1.50	1244.5	0.0008035	803.5	100.4	50.2	206
19	ミ	1.58	1318.5	0.0007584	758.4	94.8	47.4	209
20	ファ	1.67	1396.9	0.0007159	715.9	89.5	44.7	211
21		1.75	1480.0	0.0006757	675.7	84.5	42.2	214
22	ソ	1.83	1568.0	0.0006378	637.8	79.7	39.9	216
23		1.92	1661.2	0.0006020	602.0	75.2	37.6	218
24	ラ	2.00	1760.0	0.0005682	568.2	71.0	35.5	220
25		2.08	1864.7	0.0005363	536.3	67.0	33.5	222
26	シ	2.17	1975.5	0.0005062	506.2	63.3	31.6	224
27	ド	2.25	2093.0	0.0004778	477.8	59.7	29.9	226
基準周波数(ラ)		440	クロック周波数φ		4MHz =	4000000		

～特別付録～

```
‘----------ここより初期宣言。構造体の宣言部分----------
Option Explicit
Type indata
    Musical_Scale As String          '音階（ドレミファ）
    Musical_Notation As String       '音符（音の長さ）
End Type
Type Element_ST
    head As String                   '頭の文字列
    set_data(3) As indata            '真ん中の文字列
    derriere As String               'お尻部分の文字列
End Type
Type EEPROM_Data
    data(15) As Element_ST
End Type
‘----------ここより入力状況を配列へ格納----------
Public Sub GetMusicalData(ggg() As Integer)
'音階の情報を配列に格納する。 k は周波数を求める式の指数部の変数　cycle = 440 * 2 ^ (k / 12)
  Dim j As Integer, k As Integer
  Dim buff As String
  Dim times As Double

    For j = 0 To 62
        Select Case Worksheets(1).Cells(3 + j, 2)
            Case "シ 2": k = 26
            Case "ラ 2": k = 24
            Case "ソ 2": k = 22
            Case "ファ 2": k = 20
            Case "ミ 2": k = 19
            Case "レ 2": k = 17
            Case "ド 2": k = 15
            Case "シ 1": k = 14
            Case "ラ 1": k = 12
            Case "ソ 1": k = 10
            Case "ファ 1": k = 8
            Case "ミ 1": k = 7
            Case "レ 1": k = 5
            Case "ド 1": k = 3
            Case "シ 0": k = 2
            Case "ラ 0": k = 0
            Case "ソ 0": k = -2
            Case "ファ 0": k = -4
            Case "ミ 0": k = -5
            Case "レ 0": k = -7
            Case "ド 0": k = -9
            Case "休符": k = 100
            Case "おしまい": k = 101
            Case Else: k = 100
        End Select
        ggg(j, 0) = k
        '音の長さを決める　'1' = 0.0625sec
        buff = Worksheets(1).Cells(3 + j, 4)
        If (Worksheets(1).Cells(3 + j, 5) = ".  ") Then times = 1.5 Else times = 1
        If (buff <> "") Then                                                    '0 で除算するのを防止
            ggg(j, 1) = 16 / Val(buff) * 2 * times
        Else
            ggg(j, 1) = 0
        End If
    Next j
End Sub
‘----------ここより入力状況を格納した配列からデータを読み出して EEPROM へ保存するデータを作成----------
Sub  データ作成()
  Dim i As Integer, j As Integer, k As Integer, m As Integer
  Dim timer_value As Integer, music_end_flag As Integer, filenum As Integer, ggg(62, 1) As Integer
  Dim buff As String
  Dim temp As EEPROM_Data

    'データの初期化
    music_end_flag = 0
    For i = 0 To 15
        temp.data(i).head = ":1042" + Hex(i) + "0"
        Select Case i
            Case 0: temp.data(i).derriere = "006E"
            Case 1: temp.data(i).derriere = "005E"
            Case 2: temp.data(i).derriere = "004E"
            Case 3: temp.data(i).derriere = "003E"
            Case 4: temp.data(i).derriere = "002E"
            Case 5: temp.data(i).derriere = "001E"
            Case 6: temp.data(i).derriere = "000E"
            Case 7: temp.data(i).derriere = "00FE"
            Case 8: temp.data(i).derriere = "00EE"
            Case 9: temp.data(i).derriere = "00DE"
            Case 10: temp.data(i).derriere = "00CE"
            Case 11: temp.data(i).derriere = "00BE"
            Case 12: temp.data(i).derriere = "00AE"
```

```
            Case 13: temp.data(i).derriere = "009E"
            Case 14: temp.data(i).derriere = "008E"
            Case 15: temp.data(i).derriere = "009C"
        End Select
    Next i

    GetMusicalData ggg                                          '入力された音符のデータを読み出し。
    '音階からタイマーの値を決める。
    For j = 0 To 62
        i = Int(j / 4)
        m = j Mod 4
        If (music_end_flag = 0) Then
            k = ggg(j, 0)
            If (k < 100) Then
                If (Worksheets(1).Cells(3 + j, 3) = "♯") Then k = k + 1
                If (Worksheets(1).Cells(3 + j, 3) = "♭") Then k = k - 1
                buff = Hex(Round(256 - 1000000 / (16 * 440 * 2 ^ (k / 12)), 0)) 'タイマー設定値を 16 進変換
                If (Len(buff) = 1) Then buff = "0" + buff                       '1 桁のときは上の桁を「0」とする
                temp.data(i).set_data(m).Musical_Notation = "00" + buff
            ElseIf (k = 100) Then                                               '休符のとき
                temp.data(i).set_data(m).Musical_Notation = "00FF"
            ElseIf (k = 101) Then                                               'ここで音楽が終わりのとき
                temp.data(i).set_data(m).Musical_Notation = "0000"
                music_end_flag = 1
            End If
            '音の長さを決める　'1' = 0.0625sec
            buff = Hex(ggg(j, 1))
            If (Len(buff) = 1) Then buff = "0" + buff                           '1 桁のときは上の桁を「0」とする
            temp.data(i).set_data(m).Musical_Scale = "00" + buff
        Else
            temp.data(i).set_data(m).Musical_Notation = "0000"
            temp.data(i).set_data(m).Musical_Scale = "0000"
        End If
    Next j
    temp.data(15).set_data(3).Musical_Notation = "0000"                         'データの最後は必ず「0000」
    temp.data(15).set_data(3).Musical_Scale = "0000"
    'ファイル出力
    filenum = FreeFile
    Open CurDir() + "¥eeprom_data4.hex" For Output As #filenum
    For i = 0 To 15
        Print #filenum, temp.data(i).head;
        For m = 0 To 3
            Print #filenum, temp.data(i).set_data(m).Musical_Notation;
            Print #filenum, temp.data(i).set_data(m).Musical_Scale;
        Next m
        Print #filenum, temp.data(i).derriere                                   ' + vbCrLf
    Next i
    Close #filenum
    MsgBox CurDir() + "¥eeprom_data4.hex", , Title:="ファイルの保存先は..."
End Sub
```