DAC がなくてもオケオケオッケー♪はしご回路でつくる

エレクトリックきいちゃんの製作

著者 ： T-ボーンナム

キーワード：MSP430,DAC,サンプリング周波数,量子化,R-2R ラダー抵抗,
テブナンの定理,重ね合わせの原理,反転増幅回路,オケオケオッケー♪

【はじめに】

著者が現実世界で疲れたとき,たまたま観たアイカツ！で冴草きいちゃんが発する「オケオケオッケー♪」というセリフに元気をもらいました.それからというもの,仕事などでうまくいったときに「オケオケオッケー」と呟いていることに気がつきました.「きいちゃんの『オケオケオッケー♪』に飢えているのだ」と自覚したとき「そうだ!いつでもどこでもきいちゃんに肯定してほしい!」という思いから始まったエレクトリックきいちゃん作成の記録です.

ここでは,オケオケオッケーを発するための原理,実際の作成方法およびプログラムも示していきます.

【用意するもの：気持ち】

今回はマイクロコンピュータこと,マイコンを使用します.マイコンとは、~~パソくん~~パーソナルコンピュータでマイコンへの命令をプログラミング言語で書き,その命令を実行する LSI[1]です。すなわちプログラミングをやります.そうです.2020 年にはみなさんの小学校でも必修化されるアレですね.「いきなり不安だな」とこれから始めるみなさんも,「普段から書いていますわ」とマイコンも嗜むみなさんも,きたるべき 2020 年に備えましょう.

【用意するもの：マイコン】

今回は,Texas Instruments（テキサス・インスツルメント以下 TI 社)の MSP430 シリーズを使用して説明します.世の中にはラズベリーパイでアイカツ！筐体ができるハイスペックなものや,アードゥイノなど使用者も多く充実しているものなどなど,マイコンを嗜むみなさんの中にはすでにほかのマイコンを使っているかと思います.しかし今回は，あの秋葉原ですら数種類程度しか置いてない,はっきり言ってマイナーマイコンである MSP430 を使用します.
このマイコンで TI 社が挙げる利点は

・駆動電圧が低い(3.3～1.8V)
・低消費電流である
・MSP430 シリーズ間で互換性がある

とされています.

とにかく低消費電力を実現し,電池駆動で持ち運びができるデバイスなどに利用されます.今回は「MSP430-EXP430G2」を用いた説明をします.

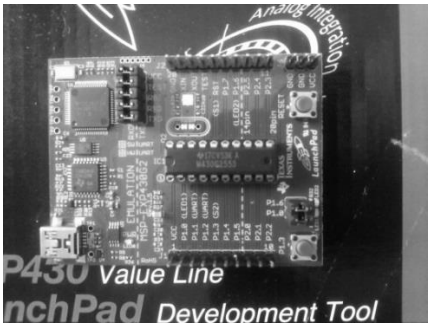

図１．MSP430-EXP430G2 の外観

[1] LSI (Large Scale Integration)は、トランジスタやダイオード等々の電子素子を１つの IC に詰め込んだものです。

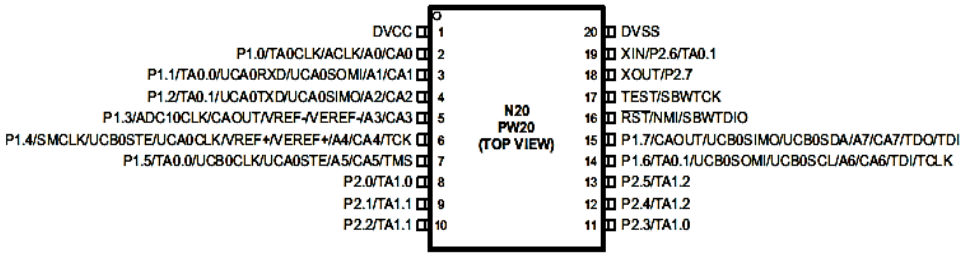

図 2. MSP430G2553 ピン配置(DIP-20pin)
(MSP430G2553 データシートより引用)

プログラムは統合開発環境(IDE)と呼ばれるパソコン上の環境で書いていきます.MSP430-EXP430G2 などの MSP430 は Eclipse という統合開発環境をベースとした「Code Composer Studio IDE 」(以下 CCS)を使用します.ここでは紹介のみに留め,**別途何らかの形で紹介しようと思います.**

(※注意： CCS で MSP430-EXP430G2 の開発を行う際の OS は WINDOWS でのみサポートされているそうです.MAC や Linax 64bit をお使いの方は,サポートされていません.CCS を使う場合は,ほかのサポートされている MSP430 の開発ボードを用いるか, Arduino のスケッチのように開発できる「 Energia 」を用いるとよいでしょう.)

「MSP430-EXP430G2」には MSP430G2553 というマイコンがすでに乗っています.
最大動作周波数 16MHz, コードなどを記録する ROM は 16KB (= 16×1024B) を有し,演算処理等で使用する RAM は 512B という,他と比べたら小さいマイコンであります.「MSP430-EXP430G2」は秋月電子などの電気パーツ屋さんでお求めできるかと思いますが先にも書いたようにマイナーなのであまりないかもしれません….このように,マイナーだの小さいだの書いておきながら,なぜ MSP430 を紹介するのか『MSP430 が大好きだから. 』です.

マイコンにはそれぞれ特色があり,アイドルのように自分の応援したいマイコンがあるはずです.著者にとってそれが MSP430 だったのです.というわけで,本書では,MSP430 でこんなことができるぞ!というプロデュースをしたい気持ちも含まれています.

【音を出すには？DAC？】

さて,本題に入ります.マイコンを嗜むみなさんであれば,音を出力するためには DAC を使用するかと思います.ここでは DAC(Digital to Analog Converter)について簡単にではありますが説明します.

音声のような一次元空間のアナログ信号は途切れるところはなく,時間に対して連続に振幅が出力されたものです.言い換えれば,どれだけ細かくしても連続してデータがあります.アナログ信号はデジタル信号に変換するとき,ある程度の時間間隔で振幅を飛び飛びに取り出します.データを取り出すことを「サンプリング(標本抽出)」といいます.このときの周波数を「サンプリング周波数」といいます. しかし,デジタルといえばアニメや映画でおなじみのように 0 と 1 が羅列する 2 進数で表現されますね.マイコンも同様に 0 と 1 の 2 進数しか分らないので変換します.この値は一定間隔で飛び飛びにしか配置できず, 必ず 2 進数のいずれかの値に変換されます.

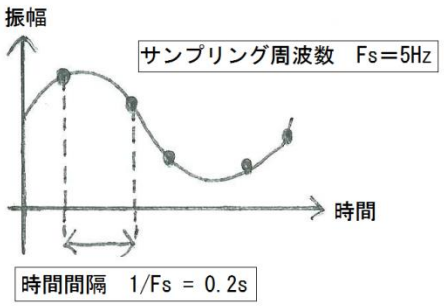

図 3. サンプリング概要図

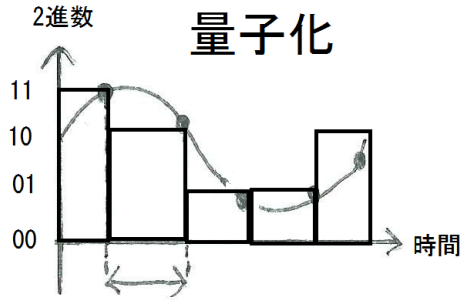

図 4. 量子化概要図

例えば 2bit で表現する場合は,

[00,01,10,11]の 4 通りの値のいずれかになり,たとえサンプリングした値がその中間にあったとしても,[00,01,10,11]のどれかになります.これを「量子化」といいます.

先に紹介したことは ADC(Analog to Digital Converter)の機能で行います.DAC の説明をするつもりが ADC の説明になっていますが,DAC は ADC の逆をやります.量子化されているデータをサンプリング周波数の時間間隔で出力して音を出します.この「サンプリング周波数」と「量子化」というのが重要になります.量子化ビット数が多ければ多いほど振幅が,サンプリング周波数がより高ければ高いほど,時間方向になめらかになり,アナログ信号に近づきます.ちなみにサンプリング周波数は,目標の周波数の 2 倍以上が必要です. そうしないと高い周波数をサンプリングするはずが,サンプリング周波数の半分の周波数で折り返ってきてしまう現象が発生します[2].逆に言えば,サンプリング周波数の半分の周波数までしか音を表現できません.[3]例えば,サンプリング周波数が 8000Hz であれば最大で 4000Hz まで表現できます.人間の音声のほとんどのエネルギーが 4000Hz 以下に詰まっているため,このサンプリング周波数がよく使われます.

一方で,量子化する際の指標は量子化ビット数として表現します.例えば 8bit の量子化ビット数の場合は,10 進数でいうと 2 の 8 乗, すなわち 256 個に量子化されます.ほかには「ダイナミックレンジ[4]」という指標があり,振幅を「dB(デシベル)」で表現することもあります.

そこで,エレクトリックきいちゃんの性能は,

・量子化ビット数 ： 8bit
・サンプリング周波数 ： 8000Hz

で作成します.

【GPIO しかない！困った！】

MSP430G2553 に限らずマイコンには GPIO(General-purpose input/output)とよばれるポートが必ずあります.日本語に訳すと「汎用入出力」です.この GPIO は単純に電圧の値 High または Low により 2 進数で 1/0 を判断します.逆に出力は High または Low の電圧を出すことで 1/0 を表現します.(High および Low の電圧はマイコンの仕様に依ります.)

前の量子化の説明から考えると,GPIO だけでは 0 と 1 の 1bit でしか音を表現できないことになります.そこで,「R-2R ラダー回路」という有名な回路を用いると DAC のように出力できます.

---

2 エイリアシングと言います.

3 ナイキスト周波数と言います.

4 1bit あたり 6 dB(振幅で 2 倍)分細かく表現できるので,8bit のダイナミックレンジは 48dB となります.(あくまで理想値なので歪み等で少し低くなります.)

ちなみにラダー回路のラダーとは「はしご」という意味であり,抵抗をはしごのように組んでいくと実現できます.計算しても面白いですし,はんだ付けも面白い回路です.

【R-2R ラダー回路】

はしご状に抵抗を接続することで GPIO が 1/0 の 1bit でしか表現できなくても,ビット表現を増やすことができます.P2.0～P2.7 は,マイコン(ここでは MSP430)に接続される GPIO ポートを示します.そのうえで, 2R , R という抵抗値をはしご状に組んでいきます.たとえば, R = 10kΩ という抵抗値を設定すると,横または GND には 2R = 20kΩを,縦方向には R = 10kΩを接続します.これを R-2R ラダー回路[5]といいます.

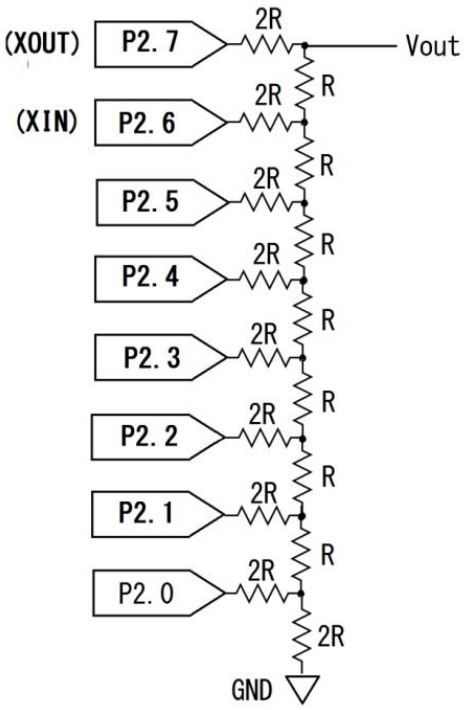

図 5. R-2R ラダー回路

この回路は DAC の原理の一つとしても有名で,実際に組んでみてもわかるように,抵抗が 2 種類だけなので簡単にできます.今回紹介する MSP430 に限らず DAC 機能がないマイコンでも GPIO が数本あれば音を出すことができます.

【はしごの魅力】

はしご状に抵抗値を組むことで,なぜ 2 進数表現ができるのか.これを説明するには「テブナンの定理」と「重ね合わせの原理」が大活躍します.

「テブナンの定理」とは,複雑な回路のとある部分を切りとって,それを電源電圧と内部抵抗だけの簡単な等価回路にしてしまおうという定理です.

一方,「重ね合わせの原理」とは,複数の線形的な回路の電圧源は,1 個の電圧源の回路とみなし,他の電圧源は短絡(ここでは GND に接続)させ,それぞれで求めた出力電圧を足し合わせることができるという原理です.

この 2 つをうまく組み合わせると,R-2R ラダー回路の導出が容易になります.例として図 6.のような 2 段の R-2R ラダー回路を解いていきます簡単のために V0,V1 の電圧が 0 または 1 が出力するものとします.

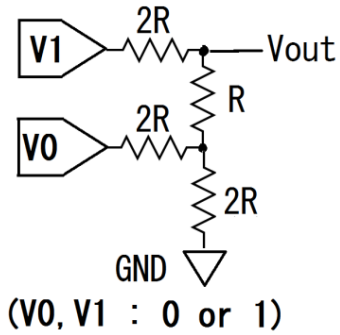

図 6. 2 段の R-2R ラダー回路

5 学生時代,テブナンの定理の演算問題として出題されていましたが「こんなん何の役になんのや」と思ったものです.まさかアイカツ！に通じていたとは夢にも思いませんでした.

まず，V0 と接続している抵抗と GND に接続されている抵抗の回路で切ります．

次に，切り取ったところの接点電圧を求めます．このときは，V0 側の 2R の抵抗値と GND 側の 2R の抵抗値から抵抗分圧により電圧は V0/2 となります．

さらに内部抵抗を求めるときは，電圧は考えないようにするため，V0 は短絡状態にします．すると，GND に接続された状態になり，その時の切り取った回路上の内部抵抗は 2R と 2R の並列抵抗とみなせるので，抵抗値は R となります．

最後に，上記で求めた接点電圧と内部抵抗をテブナンの定理の等価回路に当てはめます．これで 1 段目はできあがりです．

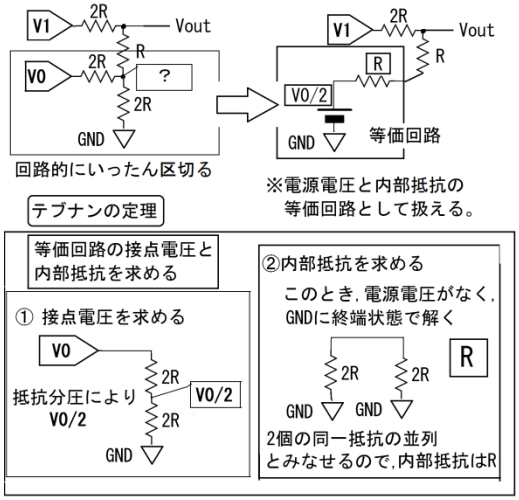

図 7. 1 段目の解き方

続いて 2 段目を解く際は，1 段目と同様にして V1 までの範囲の回路を切りとり，ここで，重ね合わせの原理を用います．

まず，等価回路側の V0/2 の電圧を無いものとし，GND に短絡させて接点電圧 V1_out を求めます．このとき(R+R)の直列抵抗と，V1 側の 2R の抵抗で抵抗分圧できているので V1_out = V1/2 と求められます．

次に，V1 側の電圧を無いものとして，これも GND に短絡させます．このときも同様にして抵抗分圧で V0/2 の半分となり V0_out = V0/4 と求められます．

最後に V1_out で V0_out の和がまさに重ね合わせの原理が適応できるので，このときの Vout = V1/2 + V0/4 となります．これで，テブナンの定理的にいうと電圧を求めることができました．

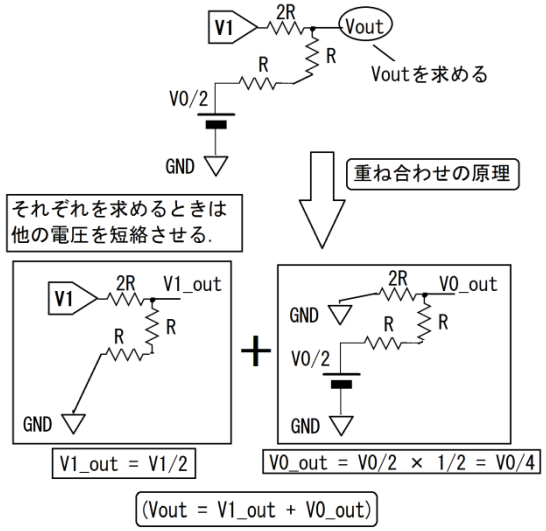

図 8. 2 段目重ね合わせの原理解き方

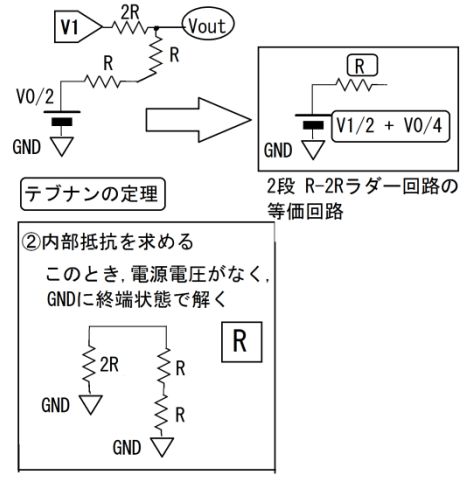

図 9. 内部抵抗の解き方

内部抵抗は図 9 のようにお互いの電圧を無いものとして GND に短絡させれば 2R どうしの並列抵抗になるので内部抵抗の抵抗値は R となります．このように最終的にはテブナンの定理の等価回路の回路になりました．

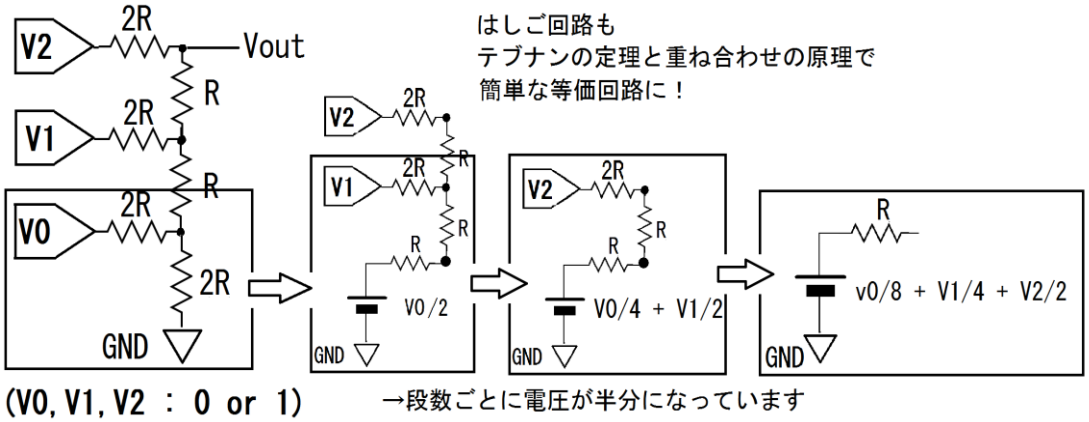

図 10. 3 段の R-2R ラダー回路の解き方

テブナンの定理と重ね合わせの原理を用いると,図 10. のように連鎖的に求められることがわかります. つまり,抵抗分圧により階段の段数の分だけ出力電圧が半分になり,内部抵抗は並列抵抗によりRのままとなります.

図 10. では 3 段しか求めていませんが,8 段でも同様にして求められます. ちなみに 図 5. R-2R ラダー回路の場合, P2. x の端子電圧を Vx とおくと式①になります. 式からもわかるかと思いますが, ちょうど 8bit の表現ができます. 一つ注意したいのが,Vout の最大の電圧は, 255/256 となります.

$$Vout = \frac{V0}{2^8} + \frac{V1}{2^7} + \frac{V2}{2^6} + \frac{V3}{2^5} + \frac{V4}{2^4} + \frac{V5}{2^3} + \frac{V6}{2^2} + \frac{V7}{2^1}$$

(V0～V7 ： 0 or 1)

・・・ ①

つまり端子電圧の最大である 1 より 1/256 少なく出力されます. (今回は後で増幅させたりゲイン調整するのであまり気にしていません.)

このように,GPIO を 8 本使うことで出力電圧は8bit で表現できることがわかりました. ただし, この R-2R ラダー回路での注意点は,抵抗値の誤差に影響されやすいので,できるだけ精度の高い抵抗(例えば金属皮膜抵抗)を使います.

【測定の準備 ①はしご回路の作成】

R-2R ラダー回路の波形をオシロスコープで確認するため,R-2R ラダー回路を作成します. 図 5. の回路図にならってラグ板に対して 20kΩ と 10kΩ をはんだ付けしました.

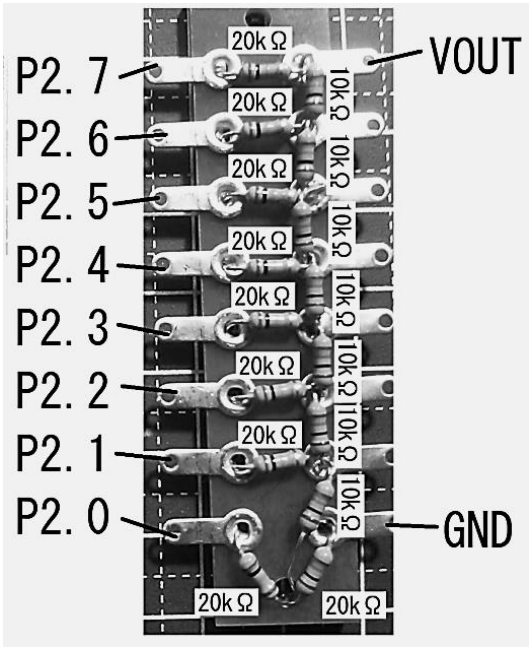

図 11. R-2R ラダー回路制作例[6]
(撮影の都合上,カーボン抵抗です.)

6 実は悪い例です. 8bit だからといって,8P のラグ板をうっかり買うとこうなります. 10P のラグ板などを買うと,図 5. のようにきれいなはしごになります.

図 11 の R-2R ラダー回路を図 1 で紹介した MSP430-EXP430G2 へ接続するための配線も紹介します.

図 12.のように,片側をメスのジャンパワイヤというものにラグ板をはさむためのクリップにつなぎかえています.
(部品は巻末の部品表を参照願います.)

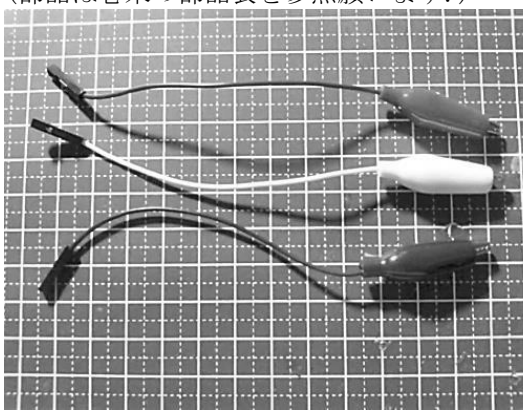

図 12.接続用配線

これを図 11. R-2R ラダー回路制作例で作成したラグ板の端子P2.0～P2.7に対応する箇所と MSP430-EXP430G2 に対応する箇所を接続用配線で接続します.

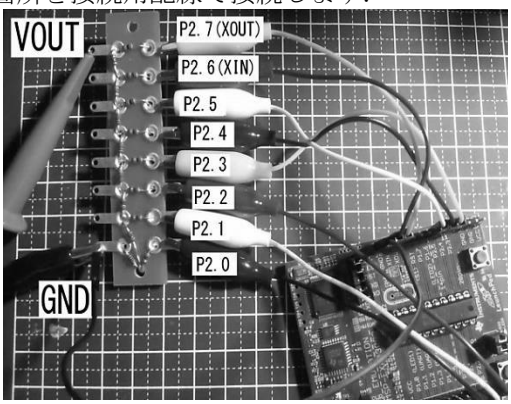

図 13.全体の接続例

※MSP430-EXP430G2 上では P2.6 は「XIN」,P2.7 は「XOUT」が対応します. また, MSP430-EXP430G2 の GND とラグ板との接続も忘れがちなので注意します.

【測定の準備　②プログラムの作成】

ここでは,プログラム上でどういった処理をして動いているのか,エレクトリックきいちゃんのアルゴリズムを説明します.これは MSP430 に限らずほかのマイコンでも同様の処理になります.

- MSP430 の設定を行う.
- CPU をオフ状態にする.
- ボタンが押されたら,サンプリング周波数の時間間隔で P2.0～P2.7 の GPIO を動かす.
- 音声データがあるまで P2.0～P2.7 の GPIO を操作する.
- 音声データが終わったら, CPU はオフ状態.

音声を出すための処理として,GPIO を High/Low とさせることで量子化された電圧を出力し,サンプリング周波数の時間間隔でGPIOを変化させることで音声を表現できるのでプログラミング自体はとても単純なものとなります.

また,アルゴリズムを示すためのフローチャートも示します.こちらも大きく分けて以下になります.

- main 処理
- ボタン割り込み処理
- タイマー割り込み処理

ボタンが押されるのをじっと待つ,タイマーが完了するのをひたすら待つという処理を行わなくても,その時が来たら処理するようにすれば消費電流も抑えられて効率的です.これらを割り込み処理といいます.詳しくは下手ではありますが, 巻末おまけとしてソースコード[7] を記しますのでコメントとあわせて参照いただければと思います.

---

7 ソースコード自体は 130 行程度ですが,きいちゃんボイスで 13KB ほどあります.したがって,ここで紹介したマイコンのほとんどは,きいちゃんでできていると言っても過言ではありません.

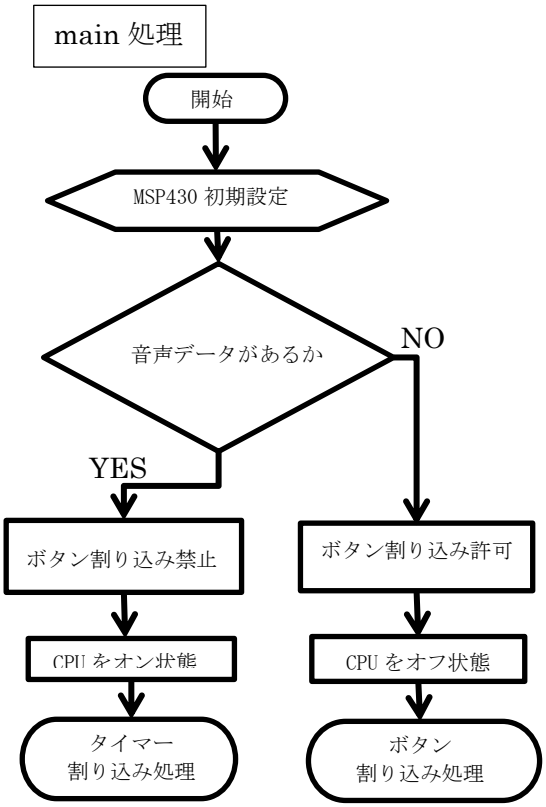

図 14. main 処理

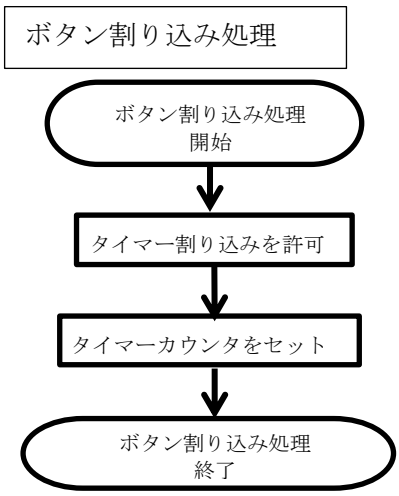

図 15. ボタン割り込み処理

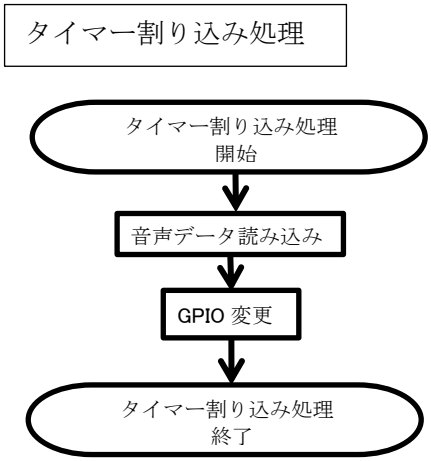

図 16. タイマー割り込み処理

【オシロスコープで確認】

プログラミングを終えて実行したら，波形を確認しましょう. 巻末おまけとして載せたソースコードを動かしています. ソースコード上では，0 から 255 の数字を 1 つずつ加算して表しています. これをサンプリング周波数の時間間隔で送出するとどうなるでしょうか? 図 13. の全体の接続例のようにオシロスコープのプローブに接続し, そのときの波形はのこぎり波になりました.

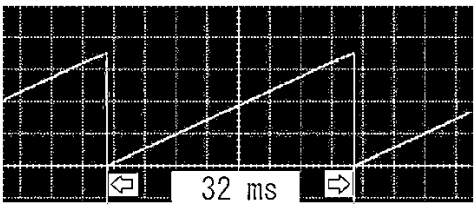

図 17. 観測波形(のこぎり波)

図 17. ではサンプリング周波数 8000Hz の時間間隔で 0 から 255 の数字を 8bit で表現し, ラダー回路を通じた結果です. のこぎり波の 1 周期のデータ数は 256 個なので, この波の 1 周期はサンプル数とサンプリング周波数の商で算出できます.

$$\frac{256}{8000Hz} = 32\text{ms} \quad \cdots \quad ②$$

また，1 周期の時間は周波数の逆数であることが知られているので

$$\frac{1}{32ms} = 31.25\text{Hz} \quad \cdots \quad ③$$

のこぎり波の周波数が 31.25Hz と算出できます.

【あなたがドなら、私はレ】

また,任意に音の高さを変えることもできます.音の高さを変えるのは,音声データの配列のカウンタをいじるのが簡単です.例として巻末おまけのソースコード(128～132 行目あたり)から抜粋します.配列のカウンタである 「wav_counter」は巻末例では 1 つずつ動かしていますが,2 つずつ,または 3 つずつにカウンタの間隔を変えたプログラムを実行すると,図 20 のように変化します.

```
126
127     }else{
128         /*** 音用カウンタがデータ数未満のとき ***/
129         temp_arr = (unsigned char)(wav_arr[wav_count
130         P2OUT = temp_arr;    // データをそのまま P2OUT にい
131         wav_counter++;       // 次の配列数のデータに移動する
132     }
```

図 18.　音の高さを変える
(巻末おまけから抜粋)

```
wav_counter +=2;  // 1つ飛ばしで配列数のデータに移動
wav_counter += 3;  // 2つ飛ばしで配列数のデータに移動
```

図 19.　変更例
(いずれかにコードを換える)

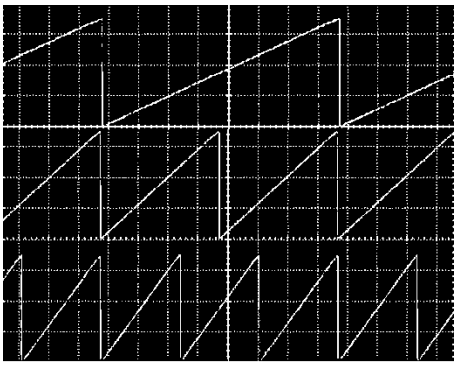

図 20. のこぎり波の周波数変化

のこぎり波は上から配列のカウンタを 1 つずつ,2 つずつ 3 つずつ変化させたとき，のこぎり波の数が比例して増えており,周波数が 2 倍，3 倍になっていることがわかりました.

このように高い音を出すこともできるので,絶対音感や音楽センスがある人はラララと作ってみるのもよいでしょう.

ただし,高い音を出そうとしてもサンプリング周波数の半分の周波数,ここでは 4000Hz 以上は出せないので,のこぎり波の 1 周期のデータ数の半分である 128 以上ごとで「wave_counter」を数えても目標の音にはなりません.
(試してみるとよいでしょう.逆に音が低く聞こえてしまいます.)

【ほかの波形はどうか】

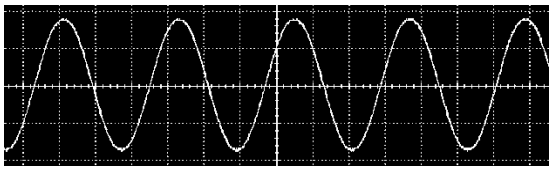

図 21. 正弦波を出力

巻末の「okokok.h」の中にも例として書きましたが,正弦波も出すことができます.これも,のこぎり波と同様にして周波数を 2 倍 3 倍にできます.

よって,「okokok.h」の配列内に 0～255 の数字を入れてもらえば好きな波形を作れます.きいちゃんボイスのほかにも 8bit サウンドのピコピコ音を楽しんでみるのもいかがでしょうか？

【オケオケオッケー】

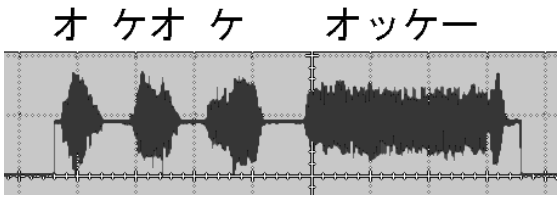

図 22. オケオケオッケーを出力

巻末おまけの「okokok.h」の配列データは諸事情によりのこぎり波や正弦波になっていますが,ビット数 8bit,サンプリング周波数 8000Hz の音声データを差し替えればオケオケオッケーとなります.きいちゃんのオケオケオッケーのデータは以下の手順で採取しました.

①「オケオケオッケー」をサンプリング周波数 8kHz で録音する.
②音声データの振幅の値を 0～255 以内にする.(正規化といいます)
③音声データを数値化したのち,配列の形式に従いカンマをつける.

①においては,高いサンプリング周波数で録音すると,4kHz 以上の音は通さないようにするローパスフィルタをかける処理も必要になります.
②においては,今回は 8bit で行うので,最大表現できる 0～255 の範囲にします.
③においては,配列の書き方に従います.またマイコンの ROM 容量にも気を付けましょう.

※音声データの容量見積もり

span × fs × bit_num = D ・・・ ④

span : 音声時間[s]
fs : サンプリング周波数[Hz]
bit_num : ビット数[bit]
D : データ[bit]

※おなじみの KB に変換

D ÷ (8 × 1024) = Data_byte [KB]
・・・ ⑤
(1K = 2^10 = 1024, 8bit = 1byte より)

もし,MSP430G2553(容量 16KB)等のマイコンで行う場合は,上記の値が 14KB 程度にします.手順では簡単にしか示していませんがこれを手でやるのがすごく手間です.**もっと簡単にできるようにするための今後の課題とさせてください.**

また,著作権や肖像権等を守って個人で楽しむ程度に留めるようにしましょう.

【アンプ】

さて,オケオケオッケーの波形が出ていることはわかりましたが,R-2R ラダー回路からの VOUT 出力だけでは電力が足りず,音を鳴らすことができません.そこでオペアンプを用います.今回のオペアンプは新日本無線の「NJM2113」を使用します.

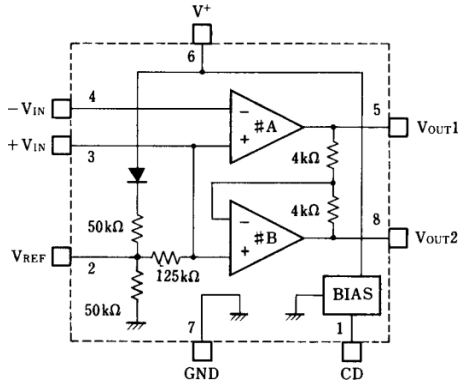

図 23. NJM2113_ブロック図
(引用 : NJM2113 データシート
http://www.njr.co.jp/products/semicon/PDF/NJM2113_J.pdf)

選定理由は,

- 電話音声帯域 (4kHz 程度) での使用
- 電源電圧 2V から動作すること
- パワーダウンとミュート兼用の CD 端子
- 差動出力形式なので出力側に大きなコンデンサが必要ない

今回のサンプリング周波数は 8000Hz なので,それに適したオペアンプを選定しました.電池駆動を想定しているので,音を鳴らさないときはパワーダウン端子による制御ができた方が消費電流の節約になります.また,電源電圧が低い方が,縦,横,高さが小さい電池を使えるので,持ち

運び可能なエレクトリックきいちゃんも作れます.持ち運びが可能なのを目指すので,差動出力形式による大きなコンデンサの部品を削減できるのは大きなメリットとなります.[8]

【ちょっとアンプの解説】

回路の解説に少しお付き合いください.ラダー回路周りは原理を含めて前で説明したので省略して,アンプ周りを少し説明します.

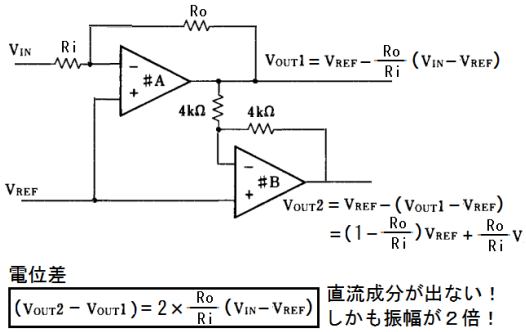

図 24. NJM2113 説明概要図

このアンプのブロック図をさらに簡単にしてみると,反転増幅を応用した構成になっていることがわかります.VREF で電源電圧の半分の電圧をバイアス電圧としてオペアンプ #A および #B の＋側に印加しています.応用回路例では,大きい容量のコンデンサを接続して電源のノイズを除去するように努めています.(本回路の C104,C105 も同様に使っています.)

また,#A が反転増幅回路, #B が 1 倍の反転増幅回路となっており,#A のオペアンプ出力を反転したものが#B の出力となっています.これで,出力 Vout1 および Vout2 の間で正と負の関係になり,直流成分は打ち消しあって,Vout1 と Vout2 の間においては交流になります.そのため,シングルオペアンプのように大きなコンデンサを接続して直流をカットする必要がありません. さらに交流は,#A と＃B のオペアンプの出力の差となるので,Vout1 の交流部分の出力より 2 倍の出力になります.[9]駆け足での説明になりましたが,選定理由の一つである「出力側に大きなコンデンサが必要ない」はこういうためです.

【アンプ周辺回路の解説】

NJM2113 との周辺回路について,巻末のおまけ回路でどういう構成になっているかも説明させてください.

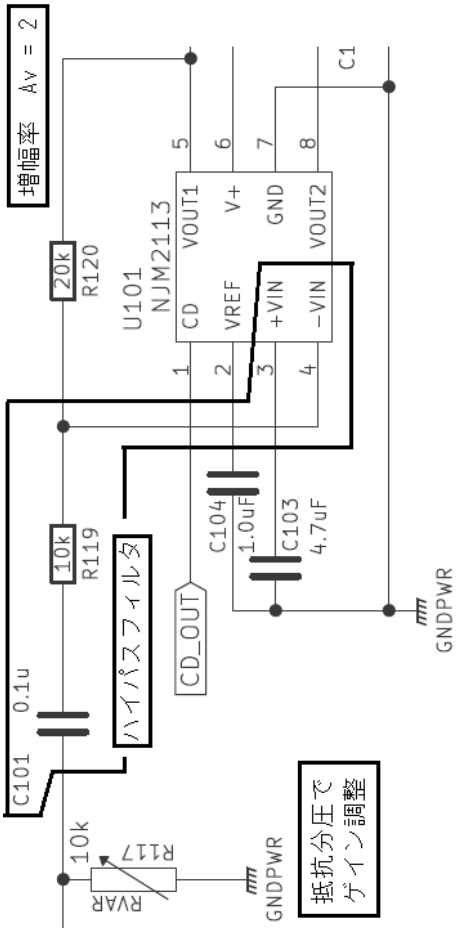

図 25. 回路図抜粋(縦ですみません)

8 出力側のインピーダンスにはスピーカ(8Ωなど)やイヤフォン(32Ωなど)が接続されるので,シングルオペアンプの場合は必然的にコンデンサ容量が大きくなってしまいます.

9 このような構成のアンプを BTL(Bridged Transless など)接続と言います.

まずは,入力信号(=Vinput としま す)は抵抗分圧でゲインを調整します.R-2R ラダー回路で解説したとおり,内部抵抗は R(=10kΩ)であるので,半固定抵抗は 10kΩを選びました.これなら入力信号は最大で Vinput/2 の入力電圧になります.このため,出力においては増幅率を 2 倍に設定しています.

ハイパスフィルタについては,NJM2113 のブロック図より,Vin+から GND にかけて抵抗分圧などがあるため,これが Vin- の入力インピーダンスに影響しています.(イマジナリショート[10]により Vin-と Vin+を同電位として扱う.)

データシートによると Vin+の入力インピーダンスが 100kΩ～220kΩであるので,図 25 の外付け抵抗である R119 の 10kΩを考慮してハイパスフィルタのカットオフ周波数を求めると

$$fc = \frac{1}{2\pi C_{IN} R_{IN\_max}} \fallingdotseq 6.92Hz \quad \cdots ⑥$$

CIN = 0.1uF
RIN_max = 10kΩ + 220kΩ

$$fc = \frac{1}{2\pi C_{IN} R_{IN\_min}} \fallingdotseq 14.5Hz \quad \cdots ⑦$$

CIN = 0.1uF
RIN_min = 10kΩ + 100kΩ

NJM2113 の入力インピーダンスのばらつきや,外部接続のコンデンサのインピーダンスを完全に無視しているのであくまでも概算程度に留めておいてください.

10 オペアンプの入力はインピーダンスがとても大きいので+側の端子と-側の間で電流がほとんど流れず,+側の端子と-側の端子の電位差を同じとして扱う考え方.これを覚えておくとオペアンプを扱いやすくなります.

【コラム　～アンプ～】

R-2R ラダー回路ははんだづけできたけれど,アンプを作るのは難しいときは,安いモノラルアンプを使ってみましょう.ラダー回路の VOUT に直流カットのコンデンサを直列に接続し,GND をクリップで接続します.

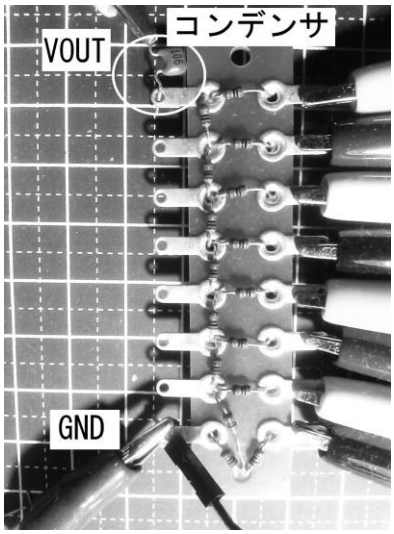

図 a.モノラルアンプの接続例


図 b.接続部の拡大図

モノラルアンプの先の細い方を R-2R ラダー回路の VOUT に,下の太い方を GND に接続します.モノラルアンプとの接続はクリップをはさむだけです.

アンプの先に 100 円ショップのミニスピーカを使い,音が鳴りました.

ここから,CIN の値をもう少し小さくして,カットオフ周波数を 300Hz 付近にもっていくのもよいかと思いましたが,部品の種類が増えるデメリットの方が大きいと判断したのでこの値にしました.このように,入力インピーダンスは出力インピーダンスより大きいので,ハイパスフィルタを構成する際はコンデンサの値は小さくすることができます.

【回路図と制作例】

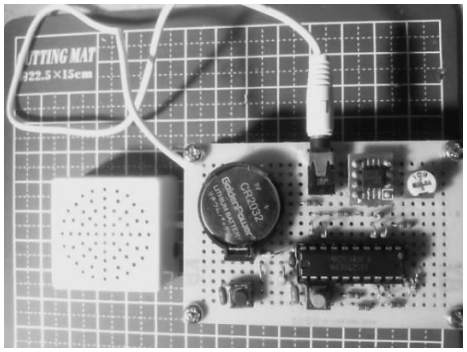

図 26. エレクトリックきいちゃん制作例

制作例では,CR2032 ボタン電池 1 個で駆動し,近所の 100 円均一で買ったミニスピーカを用いて鳴らしています.また,NJM2113 の特徴であるパワーダウン機能を回路図上では P1.0 の GPIO 割り当てたので,プログラム上で

・音を鳴らさないときは GPIO を HIGH
・音を鳴らす直前に GPIO を LOW
・音を鳴らし終わったら,GPIO を HIGH

という処理を追加してみてください.

この処理は行わなくても音を鳴らせますが,消費電流を抑えることができ,電池が長持ちしますので挑戦してみてください.(参考:待機中は 160uA 程度でした.)

【今後の展望】

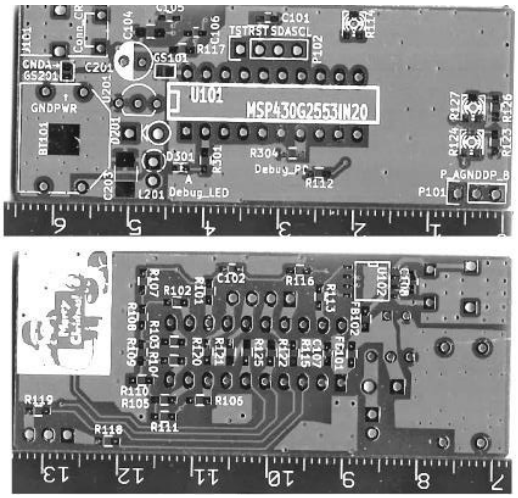

図 27. 自作基板

紹介したエレクトリックきいちゃんの回路図をさらに小さくして,とある清涼菓子(F○ISK)のサイズにおさめて持ち運びするのを最終目標にしています.実は以前にモールス符号の練習のためにエレクトリックキーヤー[11]という装置を作ったときの回路を流用したのがエレクトリックきいちゃんなのです.

この基板は,ボタン電池 1 個(LR44 1.5V)を DC-DC コンバータという回路で 3.3V まで昇圧し,あとは,これまでエレクトリックきいちゃんの原理で説明したとおり MSP430 で制御して R-2R ラダー回路で音をつくって NJM2113 で増幅しています.しかし,まだ完成ではなく,

・無駄な部品がある
・ラダー回路が 6bit 分しかない
・小さい部品を使いたい
・消費電流の低い部品を使いたい
・ DC-DC コンバータ自体で無駄に電力を消費するため, CR2032 電池にしたい

ため,改善の余地があります.

11 押すとモールス符号の音が出る機械.

特に,NJM2113 は待機電流が数百 uA ですが,他のシリーズ(NJM2135,NJM2149)では 1uA にまで落ちています.[12]

ソフトウェアもそれに合わせて改善できればと思います.改善の方向は,

- MSP430 のクロックを落としてさらに低消費電流を目指す
- 音声データの加工を容易にしたい
- 容易に音声データを読み書きしたい
- 外部記憶装置をつけて,いろんなきいちゃんボイスを楽しみたい

等々まだまだあります.

【基板作成ツールの宣伝】

ちなみに,巻末の回路図を描く際や,この基板作成にはトランジスタ技術でも取り上げられている KiCad[13]という完全フリーの基板作成ツールがあります.最大の魅力は,商用利用をしても料金を取らないので,同人ハード界隈で使っている人も多く見かけます.また,図 27 のように絵を描くのも bmp や png ファイルをちょっと加工して貼り付けるだけなので,絵を描ける皆様も楽しめるかと思います.[14]

個人でも基板が作れるすごい時代になったものです.

【実現のために】

また,【今後の展望】が実現できたら,共有できるように下記にサイトを紹介します

「https://github.com/korokorobone/Electric_kii」

Github という,ソースコードなどをプロジェクト毎に共有するサイトがありますので,著者もエレクトリックきいちゃんのプロジェクトを立ち上げました.

ここでは泣く泣く本誌に載せられなかった記事や,開発環境のインストール方法の紹介,回路図の更新や,ソースコードも載せていければと思います.ソースコードもコピー&ペーストでよいので,巻末のものを書き写さなくてもオケオケオッケーです.

【おわりに】

これまで紹介したオペアンプを用いたエレクトリックきいちゃんの回路図と部品表をソースコード同様に巻末おまけとして載せましたので,よろしければ参照願います.

R-2R ラダー回路と MSP430 で作る回路について色々と書き散らしてしまいましたが,これでいつでもどこでもきいちゃんに「オケオケオッケー」と言ってくれて心が癒されることでしょう.

また,ハードウェアを作ってしまえば,あとはみなさん思い思いの音を作成して楽しんでいただければ幸いです.

以上,最後まで読んでいただきありがとうございました. (終)

---

12 ブロック図を見る限り,バイアスにかかる電圧自体のスイッチを切っています.これで分圧のための抵抗自体に流れる電流自体を抑えているのでしょう.

13 日本語に翻訳等している支援サイト(http://kicad.jp/)もあります.また,twitter 上でもかわいいアイコンの人が情報提供しています.

14 ちなみに,著者は絵が描けないので,「いらすとや」さんからクリスマスの絵を引っ張ってきて貼り付けました.おかげでクリスマスキャンペーン中だった中国の基板屋さんにタダで基板を作ってもらいました.

【参考文献】
青木直史(2008)『C 言語ではじめる音のプログラミング-サウンドエフェクトの信号処理-』オーム社
(↑信号処理はじめるなら実例も多くてオススメ)

馬場清太郎(2016)『エレクトロニクス数式辞典』CQ 出版株式会社
(↑テブナンの定理,重ね合わせの原理などド忘れしていた公式などを実例交えて思い出させてくれました)

鈴木雅臣(2012)『回路の素 101』CQ 出版株式会社
(↑電子工作するなら手放せない１冊)

トランジスタ技術 SPECIAL 編集部(2014)『一人で始めるプリント基板作り「完全フリーKiCad 付き」』CQ 出版株式会社
(↑KiCad の使い方思い出すために引っ張りだしました)

【参考 web】
＜DAC 関連＞
Texas Instruments
『最新アナログ基礎用語集 - DA コンバーター- TI』
(http://www.tij.co.jp/lsds/ti_ja/analog/glossary/da_converter.page)
(↑DAC の簡単な概要がわかります)

アナログデバイセズ
『データ変換の基本ガイド』
(http://www.analog.com/media/jp/training-seminars/design-handbooks/ADI_Data_Conversion_Poster_F.pdf)
(↑これだけわかっていればADC/DAC データ変換マスターですわ)

＜ラダー回路関連＞
静岡大学
『R - 2R ラダー型ディジタル･アナログ変換器に関する研究』
(http://ir.lib.shizuoka.ac.jp/bitstream/10297/1435/1/24-0073.pdf)
(↑泣く泣くカットしたDAC の誤差を示す指標など詳しく書いています.)

＜DAC＋ラダー回路関連＞
村田製作所
『基本形 2 (バイナリ方式)』
(http://www.rohm.co.jp/web/japan/da_what6-j)
(↑DAC にもいろんな種類がありますね)

＜オペアンプ関連＞
新日本無線『NJM2113　データシート』
(http://www.njr.co.jp/products/semicon/PDF/NJM2113_J.pdf)
(↑いつもお世話になっております)

＜MSP430 関連＞
『超低消費電力マイコン MSP430 日本語技術資料』
(http://www.tij.co.jp/mcu/jp/docs/mcusplash.tsp?contentId=101282)
(↑英語がニガテな人でも, MSP430G2553 の開発に必要な情報はここで参照できます.)

そのほか, MSP430 Launchpad の情報は TI 社の web ページを参照に！