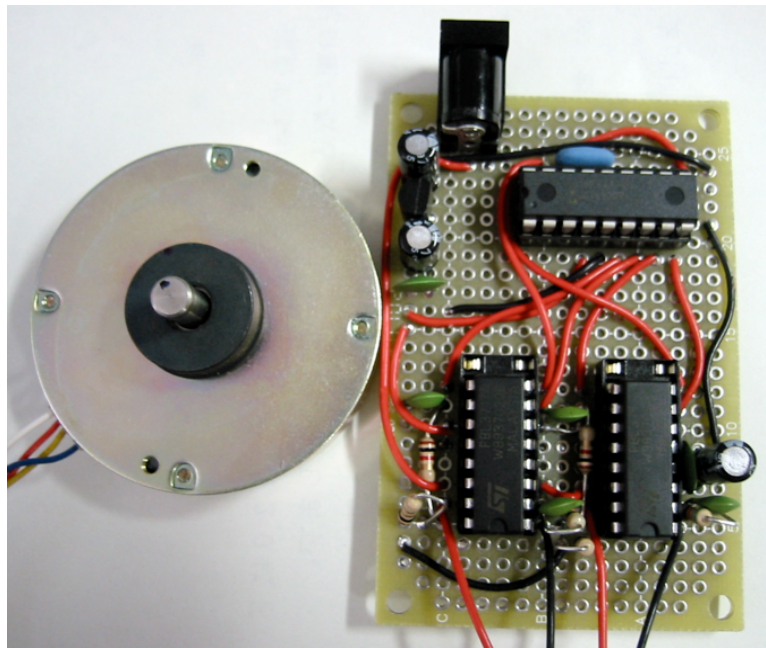


バイポーラ駆動ステッパモータ制御装置の製作

神崎正美（固体地球研究センター）
mkanzaki@misasa.okayama-u.ac.jp



SPring-8の実験でステッパモータをちょっと使う予定があったので、制御装置を作成した。モーターはロープロファイルのものが必要だったので、探したらRS（アールエスコンポーネンツ）のカatalogにあったが、駆動方法がバイポーラでしか使えないと書いてあった。バイポーラとはモータに流れる電流の極性が変わるタイプの駆動方式であり、ユニポーラは極性が1つに決まっているタイプである。ユニポーラだと秋月電子などでドライバーキットがあるし、実際いくつか手持ちもあるのだが、バイポーラは複雑なためかキットもなく、自作することにした。ドライバー回路の自作は面倒そうなので、ドライバーICを探したところSGS-Thomson MicroelectronicsのPBL3717Aがあって、アプリケーションノートもpdfでウェブ上から入手できるし、RSから購入できることが分かったのでこれを使った。ドライバーに制御信号を与える部分はワンチップマイコンであるPIC16F84Aを使った。これにプログラムを書き込むことでいろいろな動作をさせることが可能になる。PIC16F84Aへのプログラムの書き込みには秋月電子のAKI-PICプログラマーを使った。AKI-PICプログラマーとPIC16F84Aの利用については遠藤敏夫さんの「わかるPICマイコン制御」誠文堂新光社刊がよい入門書である。

部品

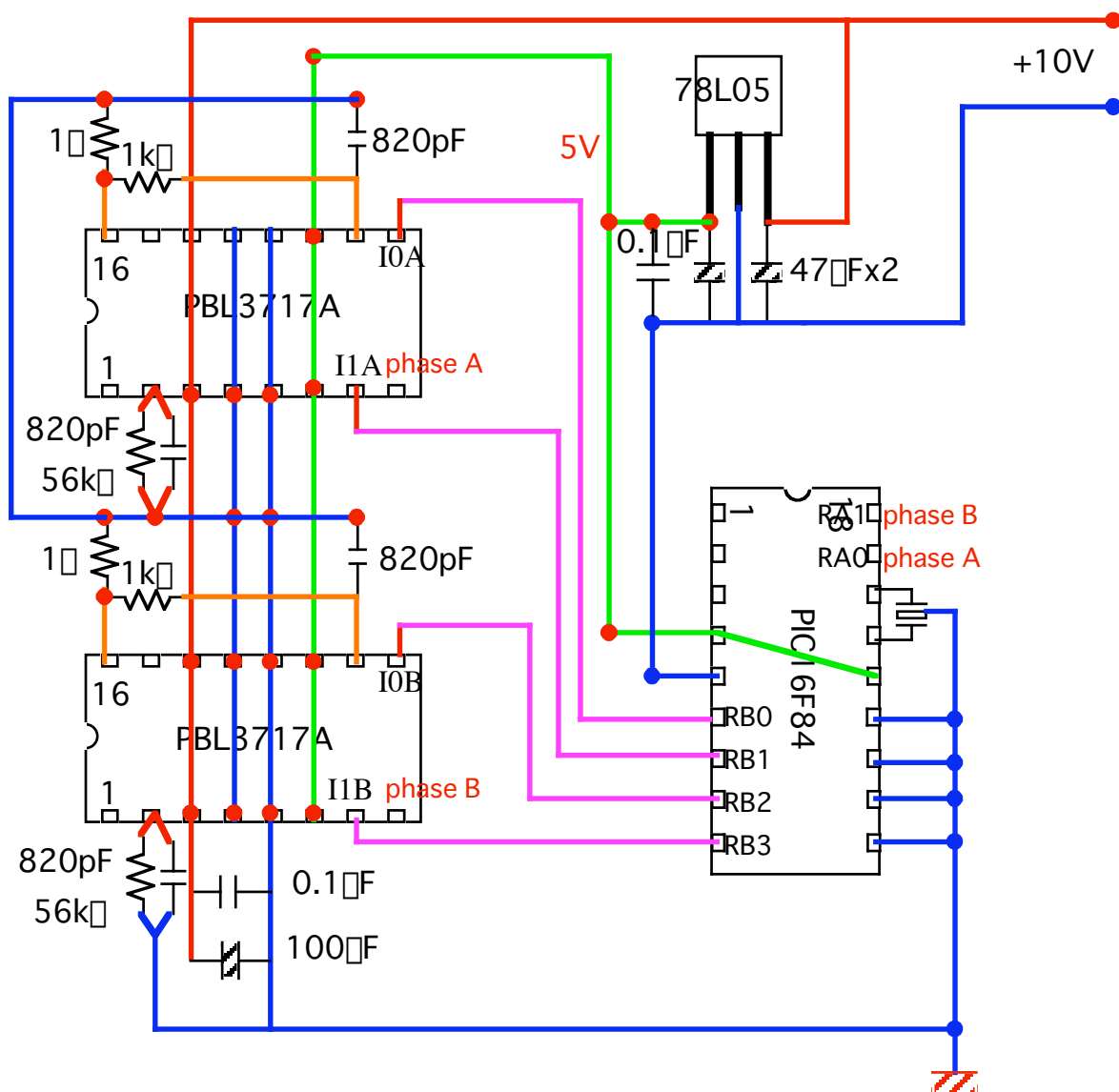
ACアダプタ電源以外全てRSから手に入る。ただしPIC16F84Aは秋月電子の値段のほぼ4倍近くする。ACアダプタは秋月で150円だったが、ちょっと重い。

回路図

ドライバー部分の回路はSGS-Thomson Microelectronicsのアプリケーションノートの回路図をそのまま使った。PIC側では、発振にはセラロックを使用。RA2,3ピンはSW入力用に空けている。SWを付ける場合は10kの抵抗を付けてSW開放時にグラウンドへ落ちるようにする。RB0-3はIOA等の制御に使う。RA0,1ピンは位相の制御に使った。RB4-8は不使用なのでグラウンドに落とした。PICはセラロック以外に外付け部品が必要なく、非常に便利である。

使用するステッパモーターは10VDC用でバイポーラ駆動のみ可能、ドライバーIC用の電源はmin10Vなので、DC10V出力のACアダプタ電源を使っている。PICとドライバーのロジック用電源はDC5Vが必要であるため、10VからレギュレータICの78L05を使って5Vを作っている（緑線部分）。

ドライバーICの1番と15番ピンがモーターのコイルにつながる。アプリケーションノートの回路図を見ると2つの出力はモーターの2つのコイルに対して逆相にするようだ。回路図のICのピン配置はtop view.



製作

汎用基板にICソケットを3つ載けて、その間を配線して作った。ドライバーIC周辺部分が結構混み合うので、プリント基板があった方がいい。アプリケーションノートにプリント基板の例が載っている。

ステッパーモータを動かしてみる

さて基板をよくチェックしたら、PICをプログラムして動かしてみる。その前に回転させるためにプログラムに必要な知識は...

I0AとI1Aの設定でモータのコイルに流れる電流を変えられる。I0A=High,I1A=Highで電流0, I0A=Low,I1A=Hで低電流, I0A=H,I1A=Lで中電流, I0A=L,I1A=Lで高電流となる。I0B, I1Bも同様。必要なトルクに応じて使い分ける。

フルステップドライブの場合は上記のようにI0A等で電流を最初に設定しておいて、Phase A,Bのみを変化させて制御する。Phase AとBで90度位相のずれた矩形波を与えると、モーターが回転する。4つのステップで1サイクルであるが、Phase A側で1100, Phase B側で0110と順番に信号を送る。この各ステップでステップ角だけ（このモータの場合は1.8度）回転する。上記回路の場合、上記の通り信号を与えるとCCW方向（モーター軸上から見て）に回転した。逆向きに動かすには位相を逆にすればよい。この駆動方式の場合は常に二つのコイルには電流が流れているのが特徴。

この通りのパルスシーケンスをPICで作ってやるとモータが回転した！各パルス間に待ち時間用のループを入れてあるが、その回数を変化させることで回転数を制御することができる。

ハーフステップドライブの場合はさらに電流のON/OFFも加わり、ステップの半分ではどちらかのコイルがOFFである状態が存在する。そのため8つのステップで1サイクルとなり、Phase A側は11110000, Phase B側は11000011の順に信号を送る、I0A等はドライブ電流値の応じて異なるが、A側は1と5ステップ目で電流がOFFになるように、B側は3と7ステップ目で電流がOFFになるようにする（デューティ比1/4）。この駆動方式の場合は2ステップで1ステップ角回転する。

この通りのパルスシーケンスをPICで与えてやるとモータはちゃんと回転した。こちらの方がより細かい角度制御となめらかな回転が可能であるが、トルクは少し弱い。

ホールドする場合はI0A等で電流を0以外に設定して、Phase A, Bを任意の値に固定すればよい。モータ軸を回転させようとするとうトルクがかかっていることがわかる。ホールド時には当然電流が流れているので、そのままで長時間放置するとモータ自体やドライバがかなり熱を持つので注意する。モータを回転していない時で、ホールドが不要な時はI0A等の設定で電流0となるようにプログラムする。

プログラム例

PICのプログラムは基本的にアセンブラで作る(Cも存在するが)。アセンブラを秋月のpa.exeでマシン語(hex形式)に変換して、picw(ライターソフト)を使ってマシン語になったプログラムを、AKI-PICプログラマーに刺したPIC ICに書き込む。

この例では1回転(360度)させて停止するだけのプログラムである。フルステップドライブ方式を使っている。基本的な部分は最初にRA, RBの該当するポートを出力用に設定して、I0A等で電流値設定のためにRB0-3のL,Hを設定する。それからループを使って4ス

テップ動作を50回だけ回す (1.8*4*50=360). 速度調整のためにwaitループを各ステップ後にcall している. ループ終了後はIA0等で電流を0に設定して終了.

```
; *** motor.asm ***
; Test program to drive bipolar stepper motor
; ra0,1: phase A,B
; rb0,1,2,3: I0A,I1A,I0B,I1B

        include 16f84.h

        .osc    hs
        .wdt    off

        org     0ch
tim1     ds      1      ; 変数
tim2     ds      1      ; 宣言
cnt1     ds      1      ;

        org     0
        goto    start

start    ;
        clr     rb      ; ポートBをクリア
        mov     !rb,#11110000b ; ポートB 下位4ビット出力用に
        mov     rb,#00001010b ; ポートB 下位4ビットで電流低に設定
        mov     !ra,#11111100b ; ポートA 下位2ビット出力用に

main
        mov     cnt1,#50      ; 50回転のためのカウンタ

lop1
        mov     ra,#00000001b ; PhA=1,PhB=0と設定 full step drive
        call    wait          ; 時間稼ぎ
        mov     ra,#00000011b ; PhA=1,PhB=1と設定
        call    wait          ; 時間稼ぎ
        mov     ra,#00000010b ; PhA=0,PhB=1と設定
        call    wait          ; 時間稼ぎ
        mov     ra,#00000000b ; PhA=0,PhB=0と設定
        call    wait          ; 時間稼ぎ
        djnz    cnt1,lop1     ; cnt1を1つ減じ, 0でないならlop1へ

        mov     rb,#00001111b ; 電流を0にする

loop
        goto    loop          ; 無限ループに入って終了.

wait     ; 「時間稼ぎ」のサブルーチン
        mov     tim1,#32      ; ここの数値を変えて速度を調整
wa1      clr     tim2          ; 0-1-> 255となる
wa2      nop                 ; 何もしない命令 (時間稼ぎのため)
        djnz    tim2,wa2      ; tim2を1つ減らして, 0でないならwa2へ
        djnz    tim1,wa1      ; tim1を1つ減らして, 0でないならwa1へ
        ret                  ; サブルーチンを終了
```