

モータを動かそう

光永 法明

第4章では、デジタル入出力(PORTAからPORTE)の基本的な使い方を、第5章ではA-Dコンバータ、シリアル通信について説明しましたが、周辺デバイスを上手に使うことで、プログラムの負担を軽くすることができます。同じ入出力ピンを使う機能や、内部で同じハードウェアを使う機能は同時に使えないので、注意します。

本章に関係する部分の回路図を図6-1に示します。ここでは、デジタル入出力、複数のタイマ、PWM出力を組み合わせて使います。PIC用とは別にモータ用の電源を用意してください。

6-1 ラジコン・サーボを動かしてみよう

今度はラジコンのサーボ(写真6-1)を動かしてみましょう。このアイコンは、章末に用語解説があります

ラジコンのサーボは、パルスを与えるとその長さに比例した角度に回転します。パルスの幅は約1[ms]から3[ms]、パルスとパルスの間隔は約20[ms] (こちらは角度に影響しない)です(図6-2)。パルスの幅が約2[ms]のとき、サーボの角度は可動範囲のほぼ中央になります。

図6-2の回路図では、サーボの線は青が信号線(PICにつなぐ)、黒がグラウンド、赤が電源です。電源の電圧は4.8Vから6V程度です。

コネクタはラジコン・サーボを扱っているお店で延長用コネクタなどを購入するか、ピン・ヘッダを使



写真6-1 ラジコン・サーボの例

3種類のラジコン・サーボ(左から双葉電子工業製S3101, 三和電子機器製ERG-VB, 双葉電子工業製S9450)。いずれも同じようにパルスの幅で角度を指令する。双葉電子工業のものは赤と黒が電源、白が信号線、三和電子機器のものは赤と黒が電源、青が信号線になっている。

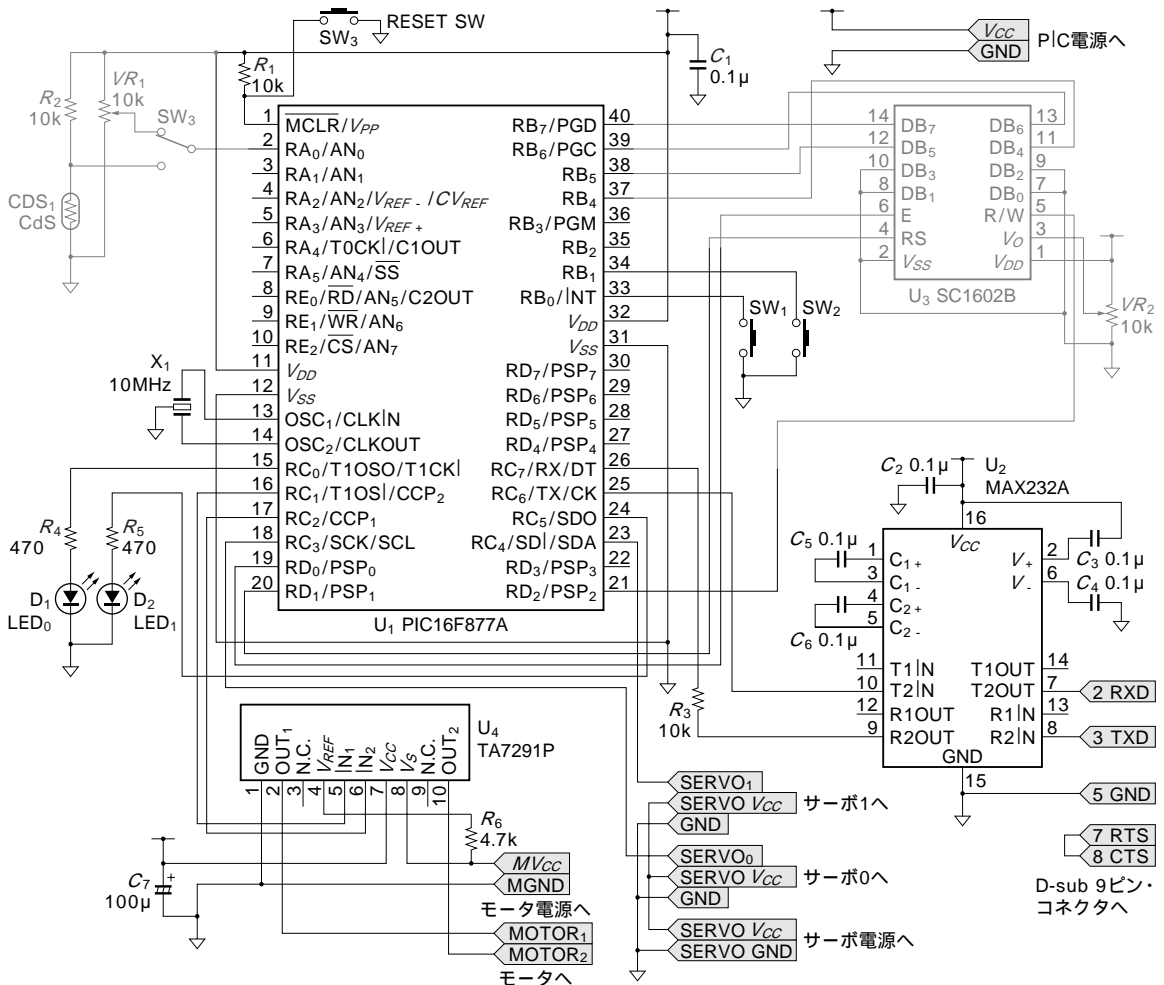


図6-1 ラジコン・サーボと模型モータに関連した部分の回路図

モータの電源はMV_{CC}、MGNDに、サーボの電源はSERVO V_{CC}、SERVO GNDにつなぐ。MOTOR₁、MOTOR₂はモータの両端へつなぐ。サーボへの信号はヘッダ・ピンを使うと、サーボのコネクタをそのまま挿すことができる。

うことができます。ピン・ヘッダの場合には逆に挿さないように注意します。電源については、PICとは別に用意してください。サーボの動作でPICの電源電圧が乱れて誤動作することがあります。

実際に動かしてみる

パルスはLEDと同様に作ることができます。ディレイ・ループを使って長さを決め、全体の20[ms]はタイマを使ってつくっています。必ず1[ms]の長さが必要なので、1[ms]部分と残り0から2[ms]に分け、1から3[ms]のパルスを作ります。ここではポートCにつないだ二つのサーボに順にパルスを送っています(図6-3)。

リスト6-1にプログラムを示します。初期化ルーチン内で、角度の初期値ANGLEを127にし、2[ms]のパルスを出すようにします。メイン・ルーチンでは、まず20[ms]タイマになるようタイマ0を設定します()。

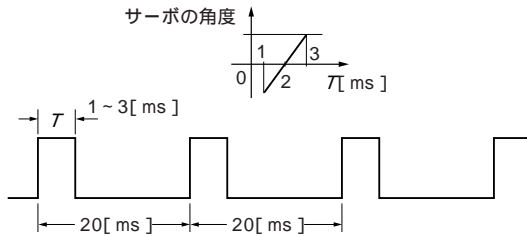


図6-2 ラジコン・サーボに送る信号

ラジコン・サーボに送る信号は、20[ms]周期のデジタル信号，“H”になっている時間に応じてサーボの角度が決まる。“H”の時間は1[ms]から3[ms]程度だが、サーボによって若干違うので実際に動かして確かめる。

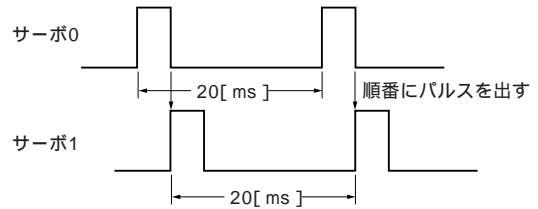


図6-3 二つのサーボへ信号を送る

二つのサーボへ信号を送るために、サーボ0の次にサーボ1へ信号を送ることにする。信号には時間差があるが、実際の動きはほぼ同時になる。同時に信号を送らないのは、サーボ0とサーボ1で“H”から“L”へ変化するタイミングがほぼ同じときに信号が乱れないようなプログラミングが難しいため。

つぎにANGLEの更新を行っています。ここではANGLEが50から200の間を1ずつ増加・減少を繰り返すようにしています。COUNTレジスタのビット0が1のときは、ANGLEを1ずつ引き()、50になったかチェックします()。

50になったらCOUNTレジスタのビット0を0にし()、次回からは1ずつ増加します。COUNTレジスタのビット0が0のときは、ANGLEを1ずつ足し()、200になったかチェックします()。200になったらCOUNTレジスタのビット0を1にし()、次回からは1ずつ減少します。

次にパルスを作ります。まず一つ目のサーボのつながったポートCのビット3を1にし()、1[ms]+(ANGLE)×8[μs]待ちます()。そしてポートCのビット3を0にします()。

二つ目のサーボも同様に、サーボのつながったポートCのビット4を1にし()、1[ms]+(ANGLE)×8[μs]待ちます()。そしてポートCのビット4を0にします()。最後に20[ms]経過するのをタイマ0のフラグをチェックして待ちます()。

サブルーチンdelayn()では、まずwレジスタの値が0かチェックし、0の場合にはすぐに戻ります()。0でない場合には(wレジスタの値)×8[μs]待つ()ことで、0から2[ms]をつくっています。

サーボによっては1[ms]、3[ms]のパルスでは可動範囲を超えるため、パルスの長さは1.4[ms](ANGLE=50)から2.6[ms](ANGLE=200)の範囲で増加、減少させています。サーボの可動範囲全体で動かすためには少しずつパルスの範囲を変えて試してみてください。

このプログラムでは一つ目のサーボのパルスを出し終わったら、二つ目のサーボのパルスを出しているため、二つのサーボへの指令時刻はずれていますが、動きからはわからないと思います。

一つのサーボに送るパルスは最大3[ms]なので、順次パルスを出す方法で動かせるサーボは6個までです。スイッチに反応するようしたり、ボリュームに反応させることは、LEDの点滅と同様な方法で実現することができます。

パソコンから角度を指令する(割り込みを使ってパルスを作る)

今度はパソコンから角度を指令するようにしてみます。パソコンからは、サーボの番号(1文字) パルス幅(2文字)で送ることにします。

サーボの番号はx(サーボ0)かy(サーボ1)、パルス幅は16進数とします。

プロトコルは第5章と同じボーレート9600 bps(9.6 kbps)、8ビット、スタート・ビット:1ビット、ストップ・ビット:1ビット、パリティ・チェックなし、フロー制御なしとします。

リスト6-1 ラジコン・サーボを動かす

```

list p=16f877a          ; PIC16F877A 用のプログラムであることを宣言
#include p16f877a.inc    ; PIC16F877A 用のヘッダ・ファイルを読み込む
__config _HS_OSC & _CP_OFF & _PWRTE_ON & _WDT_OFF & _LVP_OFF

CT_DELAY1MS equ 0x20
CT_DELAYN    equ 0x21
PORTC_       equ 0x22
COUNT       equ 0x23
TMP          equ 0x24
MSG_LOOP     equ 0x25
ANGLE        equ 0x26
} アドレスの割り当て

org 0x0
nop
goto start

org 0x4
retfie

start
; 初期設定をする
bcf STATUS, RP0 ; この2行でバンク0にする } バンク0に
bcf STATUS, RP1

clrf INTCON      ; 割り込み禁止
clrf PORTC       ; LEDが点灯しない, サーボが動かないように0に } バンク0
clrf PORTD       ; 出力は0にしておく } の設定
clrf PORTE       ; 出力は0にしておく

bsf STATUS, RP0 ; バンク1にする
movlw 0xff
movwf TRISB     ; ポートBはデジタル入力
movlw B'00000111' ; PORTBのウィーク・プルアップを有効に
                  ; プリスケーラはタイマに割り当て
                  ; プリスケーラは1/256にする } バンク1
movwf OPTION_REG ; の設定
clrf TRISC      ; 全ピン出力に
clrf TRISD      ; 全ピン出力に
bcf STATUS, RP0 ; バンク0に戻す

movlw B'00100000' ; LED0は消灯, LED1は点灯
movwf PORTC_
movwf PORTC
movlw D'127'      ; 最初のパルスの長さ1000 + 127 × 8 = 2016[ μs ] = 2[ ms ] } 初期値
movwf ANGLE       ; の代入
; 初期設定は終了

; ここからメイン・ルーチン
main
bcf INTCON, T0IF ; フラグのクリア
movlw D'61'      ; 20[ ms ]タイマの設定 } ①
movwf TMR0       ; 20[ ms ]タイマの設定

```

リスト6-1 ラジコン・サーボを動かす(つづき)

```

btfss    COUNT, 0
goto     DEC_COUNT

;; ANGLE から 1 を引いていく
decf     ANGLE, f ← ② 1 を引く
movlw   D'50'
subwf   ANGLE, w } ③
btfsc   STATUS, Z } 50 になったか
bcf     COUNT, 0 } ④
          50 になったら, ビット 0 を 1 に
          }

goto     SERVO_PULSE
;; ANGLE に 1 を足していく
DEC_COUNT
incf     ANGLE, f ← ⑤ 1 を足す
movlw   D'200'
subwf   ANGLE, w } ⑥
btfsc   STATUS, Z } ⑦
          200 になったら, ビット 0 を 1 に
          }

SERVO_PULSE
;; 一つ目のサーボのパルスを作る
bsf     PORTC_, 3 } ⑧
movfw   PORTC_   } ビット 3 を 1 にし,
call    delay1ms } ⑨
movfw   ANGLE    } 1[ms]+(ANGLE)× 8[μs] 待ち,
call    delayn   }
bcf     PORTC_, 3 } ⑩
movfw   PORTC_   } ビット 3 を 0 にする
movwf   PORTC

;; 二つ目のサーボのパルスを作る
bsf     PORTC_, 4 } ⑪
movfw   PORTC_   } ビット 4 を 1 にし,
call    delay1ms } ⑫
movfw   ANGLE    } 1[ms]+(ANGLE)× 8[μs] 待ち,
call    delayn   }
bcf     PORTC_, 4 } ⑬
movfw   PORTC_   } ビット 4 を 0 にする
movwf   PORTC

btfss   INTCON, T0IF } ⑭
goto    $-1          } 20[ms] 経過
                    } するのを待つ

goto    main

delay10us:
goto    $+1          ; 2
goto    $+1          ; 2
goto    $+1          ; 2

```

メイン・ルーチン

④ サイクル数

リスト6-1 ラジコン・サーボを動かす(つづき)

```

goto    $+1    ; 2
goto    $+1    ; 2
        } 10[ μs ]待ちサブルーチン
goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2

nop     ; 1
return  ; 2

delay1ms:
movlw   D'89'   ; 1
movwf   CT_DELAY1MS ; 1
delay1msl1:
call    delay10us ; 25
decfsz  CT_DELAY1MS, f ; 1(次の行を実行), 2(次の行はとばす)
goto    delay1msl1 ; 2
        } 1[ ms ]待ち
        サブルーチン

nop     ; 1
goto    $+1    ; 2

return  ; 2

delayn:
movwf   CT_DELAYN
movf    CT_DELAYN, f
btfsc   STATUS, Z
return
        } ⑩
        wが0のときは,
        すぐに帰る

delaynl1:
goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2
        } ⑮
        w × 8[ μs ]待ち,
        サブルーチン

goto    $+1    ; 2
goto    $+1    ; 2
goto    $+1    ; 2
nop     ; 1
        } ⑰
        8[ μs ] × wの待ちを
        行うループ

decfsz  CT_DELAYN, f ; 1
goto    delaynl1    ; 2

return

end

```