ロボット（ソフトウェアサーボ）

コンピュータ ＋ センサ ＋ アクチュエータ ＝ ロボット

実験のねらい

既存の機能部品を組み合わせてシステムを作成する際に必要なインターフェースの設計についての基礎知識を身につける。

コンピュータ、センサ、アクチュエータ等の機能部品を組み合わせてロボット（ソフトウェアサーボ）等のシステムを作成する際に、これら機能部品の間に置かれて信号の電気的性質やタイミング等の調整を行なうものがインターフェース回路である。このインターフェース回路について、電子回路の構成やコンピュータのプログラミングによりさまざまな設計が可能なことを、実験を通して理解する。

実験内容

第１章　直流モータの駆動と制御

１.１　Hブリッジ回路による直流モータの駆動

１.２　マイクロコントローラＰＩＣ

１.３　ＰWMを用いた直流モータの速度制御

１.４　ＡＤ変換、ＤＡ変換

１.５　センサを用いたソフトウェアサーボ

テキスト等のweb上の教材（URL）

http://www.ops.dti.ne.jp/~yanaka/Robot/

http://www.ops.dti.ne.jp/~yanaka/

〒371-8530　前橋市鳥羽町580

群馬工業高等専門学校

電子メディア工学科 谷中勝

TEL　027-254-9161(直通)

E-mail　yanaka@gunma-ct.ac.jp

第１章　直流モータの駆動と制御

１.１　Ｈブリッジ回路による直流モータの駆動

図１に示すように、４個のスイッチを用いた回路で、直流モータの回転方向を制御できる。

スイッチ①と④のみを同時にオンにすると、図２に示すように電流が流れて、モータは一方向に回転する（便宜上、この回転方向を正転と定義する)。スイッチ②と③のみをオンにすると、今度はモータには逆方向に電流が流れて、モータは逆方向に回転する（図３：逆転と定義する)。スイッチ③と④のみをオンにすると、モータにはブレーキがかかった状態になる（図４)。すべてのスイッチがオフのときは、モータはフリーとなる（図１)。つまり、回転していた場合には惰性で回転し続け、止まっている場合にはモータの回転軸を容易に手で回すことができる。

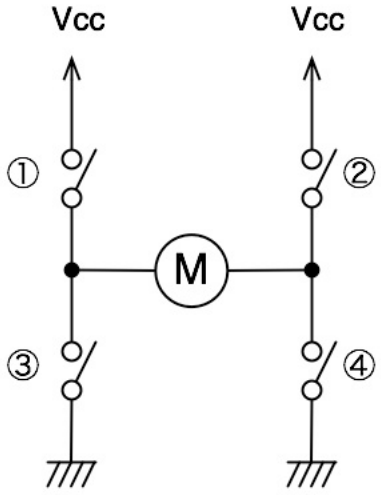


図１　Ｈブリッジ回路の原理（フリー）

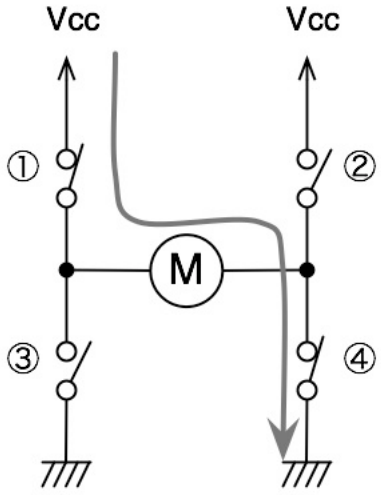


図２　正転

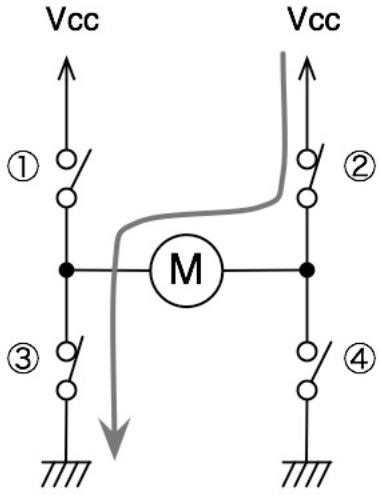


図３　逆転

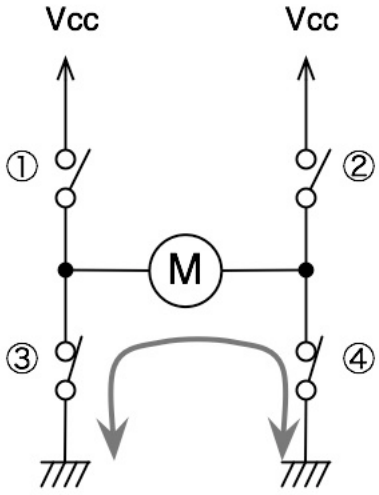


図４　ブレーキ

この回路は、構成がアルファベットのＨの形をしているので、Ｈブリッジ（H-Bridge）と呼ばれる。実際の回路では、スイッチにはトランジスタが用いられ、全体が１つの集積回路（IC）として製作され、モータドライバＩＣとして、さまざまな種類のものが各メーカから市販されている。

今回は、東芝セミコンダクタ社製の TA7291S を用いて、実験を行なう。データシートの一部をメーカ等のウェブページ（URL を下に示す）から抜粋して、次ページ以降に示す。

https://toshiba.semicon-storage.com/jp/product/linear/motordriver/detail.TA7291S.html

https://www.marutsu.co.jp/contents/shop/marutsu/datasheet/ta7291.pdf

東芝バイポーラ形リニア集積回路　シリコン　モノリシック

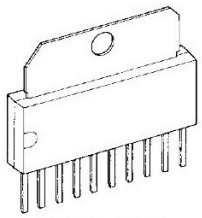
TA7291P,TA7291S/SG,TA7291F/FG

DC モータ用フルブリッジドライバ (正・逆切り替えドライバ)

TA7291P/S/SG/F/FG は、正・逆転切り替え用としてブリッジドライバで正転・逆転・ストップ・ブレーキの 4 モードがコントロールできます。

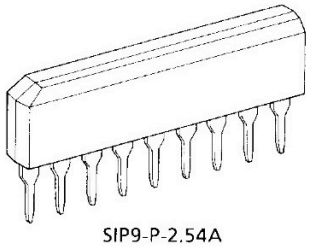
出力電流は、1.0A (AVE.) および 2.0A (PEAK) (TA7291P) 、0.4 A (AVE.) および 1.2A (PEAK) (TA7291S/F) を取り出せます。特に VTR のフロントローディング・テープローデイング・キャプスタン・リール用として最適な回路構成であり、出力側と制御側の二系統電源端子かつ出力側にはモータ電圧を制御できる V_{ref} 端子を持っており、モータへの印加電圧調整ができます。また入力電流が少なく CMOS との直結が可能です。

TA7291P



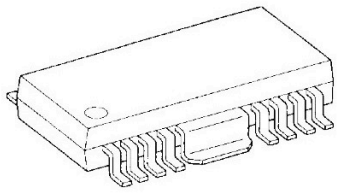
HSIP10-P-2.54

TA7291S/SG



SIP9-P-2.54A

TA7291F/FG



HSOP16-P-300-1.00

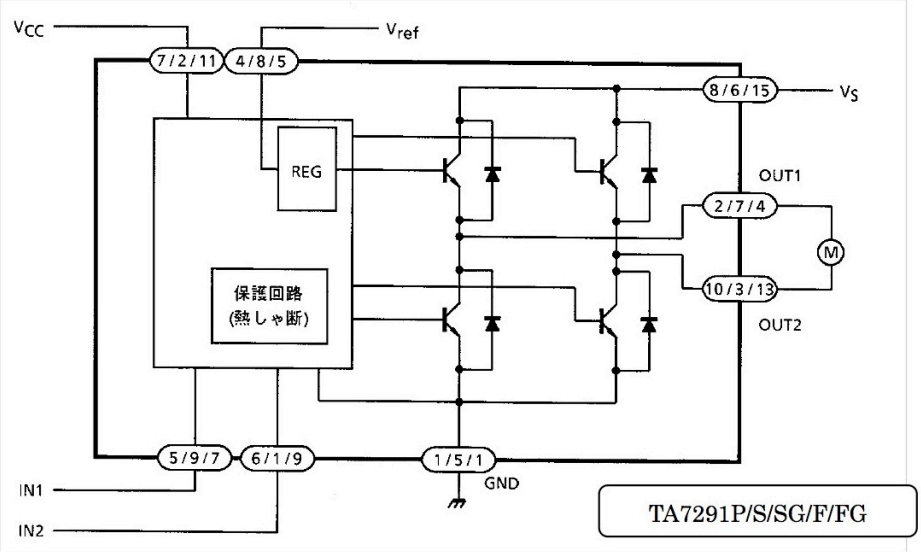
特　長

- 動作電源電圧範囲　：V_{CC} (opr) = 4.5～20 V
  ：V_S (opr) = 0～20 V
  ：V_{ref} (opr) = 0～20 V
  * V_{CC}、V_S はどのような大小条件でも誤動作しません。
  ただし、$V_{ref} \leqq V_S$ となるように使用してください。
  * V_S は 0V からですが、内部ロス(Vsat)を考えた電圧を印加しませんと、負荷は駆動できません。
- 出力電流　：P タイプ　1.0 A (AVE.) 2.0 A (PEAK)
  S/F タイプ　0.4 A (AVE.)1.2 A (PEAK)
- 熱しゃ断回路内蔵、出力端子プロテクタ回路内蔵
- 逆起電力吸収用ダイオード内蔵
- 入力ヒステリシス回路内蔵
- スタンバイ回路内蔵

質量

HSIP10-P-2.54	:	2.47 g (標準)
SIP9-P-2.54A	:	0.92 g (標準)
HSOP16-P-300-1.00	:	0.50 g (標準)

ブロック図



端子説明

端子記号	端子番号			端　子　説　明
	P	S/SG	F/FG	
V_{CC}	7	2	11	ロジック側電源端子
V_S	8	6	15	出力側電源端子
V_{ref}	4	8	5	制御電源端子
GND	1	5	1	GND
IN1	5	9	7	入力端子
IN2	6	1	9	入力端子
OUT1	2	7	4	出力端子
OUT2	10	3	13	出力端子

P タイプ　：　③⑨ピンは NC 端子
S/SG タイプ：　④ピンは NC 端子
F/FG タイプ：　②③⑥⑧⑩⑫⑭⑯ピンは NC 端子
なお F タイプの FIN は、GND にショートすることを推奨します。

ファンクション

入力		出力		モード
IN1	IN2	OUT1	OUT2	
0	0	∞	∞	ストップ
1	0	H	L	CW / CCW
0	1	L	H	CCW / CW
1	1	L	L	ブレーキ

∞: ハイインピーダンス

注: 入力は“H”アクティブ

絶対最大定格 (Ta = 25℃)

項目			記号	定格	単位
ロジック側電源電圧			V_{CC}	25	V
出力側電源電圧			V_S	25	V
制御電源電圧			V_{ref}	25	V
出力電流	PEAK	Pタイプ	$I_{O\ (PEAK)}$	2.0	A
		S/Fタイプ		1.2	
	AVE.	Pタイプ	$I_{O\ (AVE.)}$	1.0	
		S/Fタイプ		0.4	
許容損失		Pタイプ	P_D	12.5 (注 1)	W
		Sタイプ		0.95 (注 2)	
		Fタイプ		1.4 (注 3)	
動作温度			T_{opr}	−30～75	℃
保存温度			T_{stg}	−55～150	℃

注 1: Tc = 25℃

注 2: IC 単体

注 3: 基板実装時 (PCB 面積 60×30×1.6mm 銅箔面積 50%以上)

動作電源電圧範囲: V_{CC} (opr) = 4.5～20 V
V_S (opr) = 0～20 V
V_{ref} (opr) = 0～20 V
$V_{ref} \leqq V_S$

TA7291S には２つの入力端子があり、データシートのファンクションに示すように、この入力端子にディジタル信号を入力することによって、正転、逆転、ブレーキ、フリー（ストップ）のモードを切り替えることができる。

〔実験１〕図５に示すように、TA7291S に２個のタクトスイッチを接続して、ギア付き直流モータの回転方向を制御してみよ。特に、ブレーキとフリーの違いについて確認せよ。配線は、ブレッドボード上に部品を挿入して、部品の端子間をジャンパワイヤで接続すること。ブレッドボードの穴は、図６にマゼンタ色の線で示したように、横につながっている部分と縦につながっている部分とがあるので注意せよ。

表１　使用器具・部品リスト〔実験１〕

品　名	規　格	数　量	備　考
モータドライバＩＣ	TA7291S	1	
タクトスイッチ		2	
抵抗器（保護用）	3Ω	1	
ギア付き直流モータ	ZMP オリジナル	1	
直流電源（12V，5V）	ZMP オリジナル	1	
ブレッドボード		1	
ジャンパワイヤ		多数	

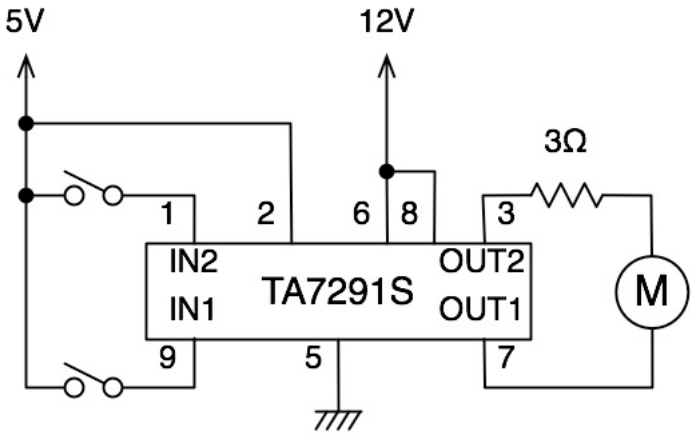


図５　〔実験１〕実験回路図

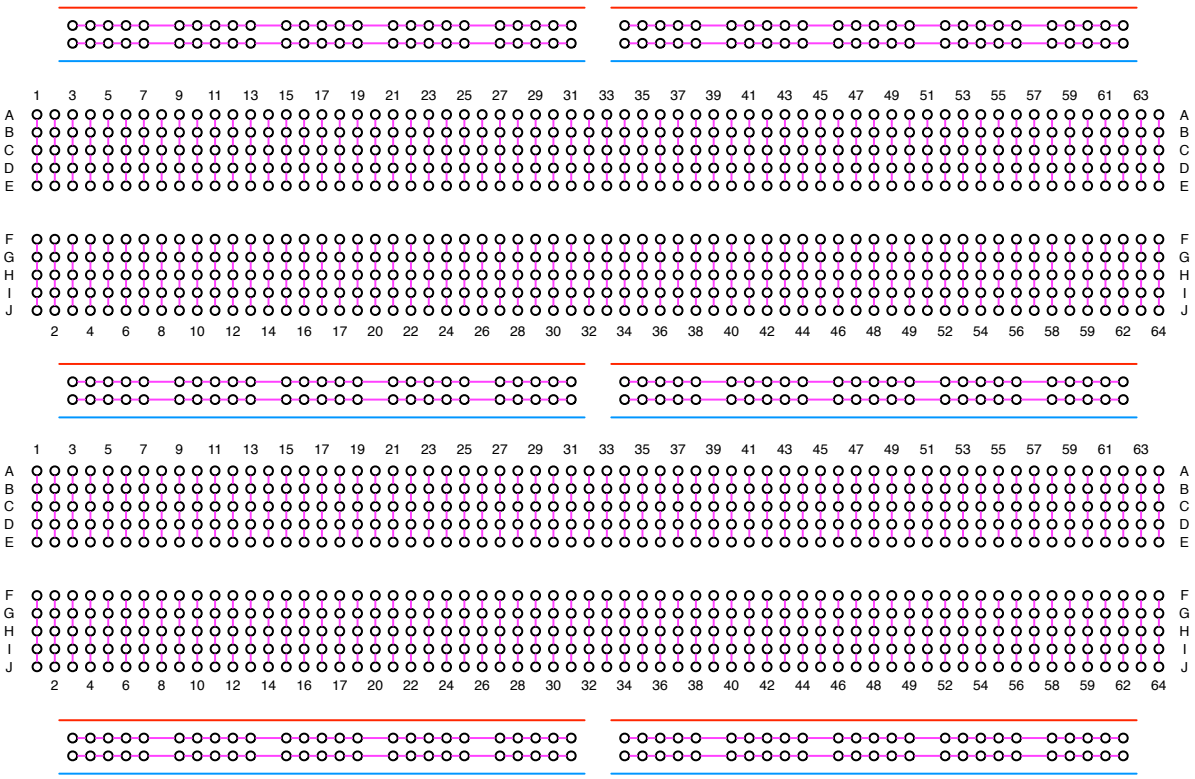


図６　ブレッドボード（裏面での穴の接続：マゼンタ色の線）

１.２　マイクロコントローラＰＩＣ

実験２以降で使用する PIC（Peripheral Interface Controller）について簡単に解説する。PIC はワンチップ化されたコンピュータである。

コンピュータのワンチップ化、ひとつの IC（Integrated Circuit）への集積化は 1971 年に初めて実現されたが、初期のものは演算や制御といった計算機の中枢部である中央処理装置（CPU；Central Processing Uinit）のみを集積化したのであって、メモリ IC や入出力のための周辺回路は外付けであった。そのため、これらはマイクロプロセッサ（microprocessor）とか MPU（Micro Processing Uinit）と呼ばれた。それでも、時代とともに IC の集積度は上がり、やがて MPU とメモリ IC や周辺回路を基板上に搭載した個人向けの計算機（パーソナルコンピュータ；パソコン）が市販されるようになった。現在では、その当時（1970 年代）の大型計算機の性能を上回るようなパソコンが広く普及しているのは周知の通りである。

一方で、メモリや周辺回路を MPU と同一の IC 上に集積化したものも、1978 年に初めて開発され、プログラム可能な制御用 IC として、家電や自動車に組み込まれて使用されてきている。これらは組み込みシステム（embedded system）とかワンチップマイコン（one-chip microcomputer）と呼ばれる。パソコンの陰に隠れて目立たないが、組み込みシステムの数と種類は非常に多く、パソコンの比ではない。

ワンチップマイコンの詳細については、ウィキペディア（ウェブ上に構築されている百科事典）を参照するとよい。

http://ja.wikipedia.org/wiki/マイクロプロセッサ

http://ja.wikipedia.org/wiki/組み込みシステム

今回は、マイクロチップ社製の PIC16F877 とを用いて、実験を行なう。データシートの一部をメーカのウェブページ（URL を下に示す）から抜粋し、教材 web 上にも置いたので参考にして欲しい。

http://ww1.microchip.com/downloads/en/DeviceDoc/30292aj.pdf

また、マイクロチップ・テクノロジー・ジャパンの URL を下に示す。

http://www.microchip.co.jp/

今回の実験では、PIC の入出力ポートとＡＤ変換機能、PWM 出力機能を使用する。PIC には８ビットの入出力のための端子が何組かあり、これらを入出力ポートという。各ポートには豊富な機能が用意されているが、プログラムで機能を選択してから使用する。例えば、入力と出力のどちらにするかを設定したり、アナログ入力とディジタル入力とを切り替えて利用する。単にディジタル信号を入力／出力するのであれば、対応するレジスタ（データの置き場所）に対して読み／書きをする命令を使って、プログラミングすればよい。

PICは内蔵されたプログラムによって動作するが、このプログラムの開発とPICへの書込みの手順が必要になる。なお、以降の回路図では、電源、クロック、プログラム書込み用信号の配線を省略しているが、３年時の実験資料を参考にして、これらの配線を忘れないこと。

１.３　ＰＷＭを用いた直流モータの速度制御

直流モータは、定常状態では、回転速度が電圧に比例し、出力トルクが電流に比例する。いわば、アナログ部品である。しかし、Ｈブリッジのようなドライバ回路とディジタル制御回路を組み合わせて用いた場合には、電流のオン／オフによる回転／停止といった二者択一のディジタル的な制御しかできない。そこで、図７に示すように、短い一定時間周期 T の中でオンの時間 T1 とオフの時間 T2 を繰り返すようにすれば、T 中での T1 を増減させることによって、等価的に、直流電流を連続的に増減させることができ、モータの回転速度を制御することができる。（負荷が一定ならば、直流電圧が等価的に変化すると考えてもよい。）

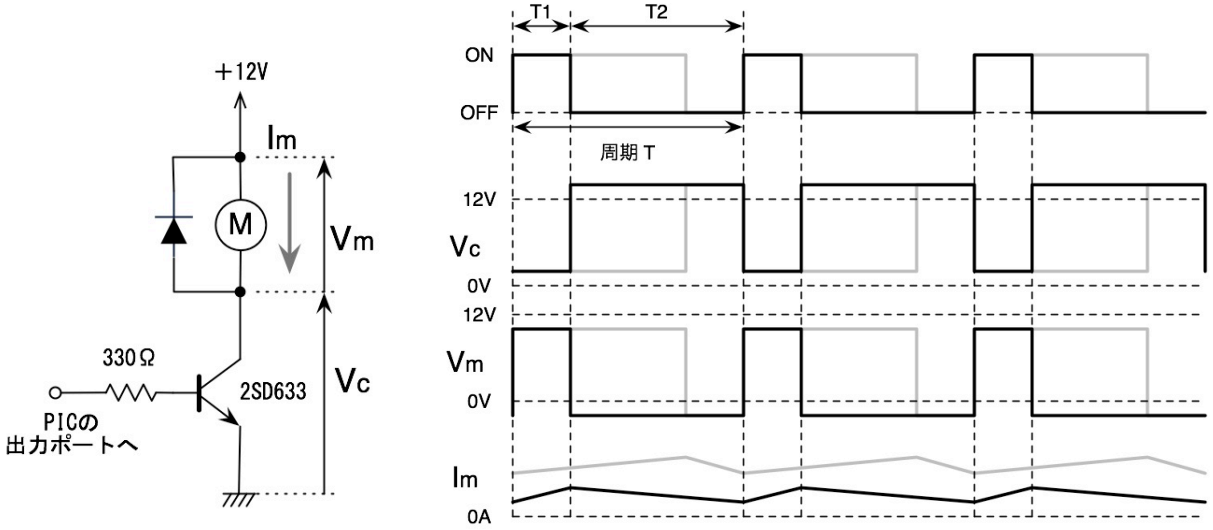


図７　モータのＰＷＭ制御回路と各部の電圧・電流波形

図７では、トランジスタスイッチのコレクタ電圧 Vc、モータの端子間電圧 Vm、モータに流れる電流（電源から供給される電流）Im を順に示して、オン時間 T1 の増減による変化を濃い実線と薄い実線とで表わしている。電圧や電流の変動は多少誇張してあるが、T1 の増減によって Im が増減することが分かる。オフの時間 T2 にも Im が流れるのは、モータに並列に接続した逆起電力吸収用ダイオードに、モータに蓄積された回転エネルギーによって発生する回生電流 Im が流れるためである。このとき、ほぼ一定速度でモータは回転し続けるが、電源からは電流が供給されないので、余分なエネルギー損失がない。そのため、このダイオードはフリーホイールダイオード（freewheel diode）とも呼ばれる。

ここで、T1のTに対する比（T1／T）をデューティ比（duty ratio）という。一定周期中のパルス幅、つまりはデューティ比を0%から100%まで連続的に変化させて、等価的に直流電流（電圧）値を変化させることができる。この方式をパルス幅変調（PWM；Pulse Width Modulation）と呼ぶ。周期Tは、１ミリ秒程度にすることが多いが、可聴域周波数を嫌ってもっと低周波数や高周波数にすることもある。

ところで、図７のドライバ回路において、ベース電流を制御して線形動作させれば、モータにかかる直流電圧を直接に調整できるが、この場合にはトランジスタでの電圧降下分がそのままエネルギーの損失になるので得策とはいえない。

また、図７のドライバ回路のトランジスタ（2SD633）の代わりに、モータドライバＩＣ（TA7291S）を用いることもできる。

〔実験２〕図８に示すように、PIC16F877 を TA7291S に接続して、スイッチの ON／OFF により直流モータの回転速度を制御してみよ。この PIC には、端子 RB0（33 番 pin）へ入力する信号の０／１によって、端子 RD0（19 番 pin）から出力する PWM 信号のデューティ比を変化させるプログラム（switch.asm）を書き込んで、使用すること（３年時の実験参照）。

表２　使用器具・部品リスト〔実験２〕

品　名	規　格	数　量	備　考
モータドライバＩＣ	TA7291S	1	
タクトスイッチ		1	
抵抗器（保護用）	3Ω	1	
ギア付き直流モータ	ZMP オリジナル	1	
直流電源（12V，5V）	ZMP オリジナル	1	
ブレッドボード		1	
ジャンパワイヤ		多数	
PIC	PIC16F877	1	
セラロック	CSTLS20M0	1	

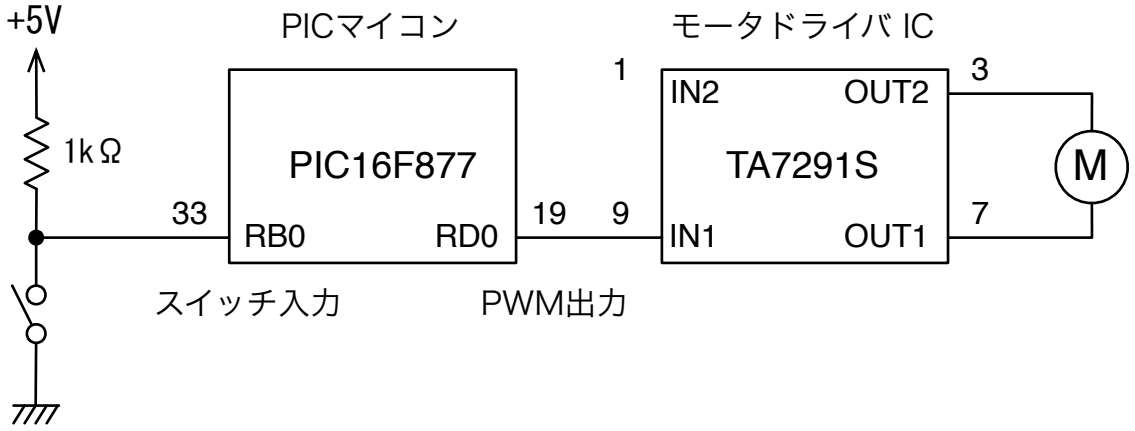


図８　〔実験２〕実験回路図

```
; switch.asm     for PIC16F877                                                  2018.06.20
; (led_2.asm)    for PIC16F877                                                  2017.11.01
;                for PIC12F675                                                  2011.06.25
;               for PIC16F648A                                                  2009.07.09
;                                                                              2005.01.05
;                                                                              2004.12.03
;                                                                              by M.Yanaka
;       1個のスイッチでモータの回転速度を切り替える（低速と高速）
;
        LIST    P=16F877        ; マイコンはPIC16F877
        INCLUDE P16F877.INC     ; インクルードファイルを指定
;*******************************
;       コンフィギュレーションビットの設定
;*******************************
        __CONFIG _HS_OSC & _WDT_OFF & _PWRTE_ON & _CP_OFF & _BODEN_OFF &  _WRT_ENABLE_ON
& _LVP_OFF & _DEBUG_OFF & _CPD_OFF
                                ;       _HS_OSC         HS高速発振
                                ;       _WDT_OFF ウォッチドッグタイマはOFF
                                ;       _PWRTE_ON       パワーアップタイマはON
                                ;       _CP_OFF         コードプロテクトはOFF
                                ;       _MCLRE_ON       マスタクリアはON
                                ;       _BODEN_OFF      低電圧リセットはOFF
                                ;       _WRT_ENABLE_ON  プログラムメモリ書き込みはON
                                ;       _LVP_OFF 低電圧書き込みはOFF
                                ;       _DEBUG_OFF      オンボードデバッグはOFF
                                ;       _CPD_OFF EEPRROMプロテクトはOFF
;--------------------------------
        CBLOCK  20H             ; ファイルアドレス20H番地から変数を割り当てる
        CNT0, CNT1, CNT2        ; 遅延用カウンタ
        ENDC                    ; CBLOCK終了
;--------------------------------
        ORG     0               ; リセットベクタ(プログラムアドレス0番地)
        GOTO    START           ; STARTへ跳ぶ
;--------------------------------
        ORG     4               ; 割り込みベクタ(プログラムアドレス4番地)
        GOTO    START           ; （今回は割り込みを使用しない）
;*******************************
;       メインルーチン
;*******************************
START:
        CALL    INIT_IOPORT     ; I/Oポートの初期設定
LOOP:
        BSF     PORTD,0         ; RD0 <- 1（モータ駆動）
        MOVLW   1               ; 約 2ms の遅延（低速）
        BTFSS   PORTB,0         ; RB0 = 1 ならば、つぎの命令をスキップ（1つ飛ばし）
        MOVLW   4               ; （RB0 = 0 ならば、）約 8ms の遅延（高速）
        CALL    DELAY
        BCF     PORTD,0         ; RD0 <- 0（モータ停止）
        MOVLW   4               ; 約 8ms の遅延（低速）
        BTFSS   PORTB,0         ; RB0 = 1 ならば、つぎの命令をスキップ（1つ飛ばし）
        MOVLW   1               ; （RB0 = 0 ならば、）約 2ms の遅延（高速）
        CALL    DELAY
        GOTO    LOOP            ; LOOPへ跳ぶ（始めから繰り返す）
;*******************************
;       I/Oポートの初期設定
;*******************************
INIT_IOPORT:
        BSF     STATUS,RP0      ; バンク1へ切り替えて、
;       MOVLW   H'FF'           ; ポートAは、
;       MOVWF   TRISA           ;       すべて入力ピンにする
```

リスト1　switch.asm

```
        CLRF    ADCON1          ; ポートAはすべてアナログ入力ピンにする
        MOVLW   H'FF'           ; ポートBは、
        MOVWF   TRISB           ;       すべて入力ピンにする
        CLRF    TRISC           ; ポートCはすべて出力ピンにする
        CLRF    TRISD           ; ポートDはすべて出力ピンにする
        CLRF    TRISE           ; ポートEはすべて出力ピンにする
        BCF     STATUS,RP0      ; バンク0へ切り替える
        RETURN
;*******************************
;       ソフトウェアタイマー        おおよそ (Wの内容) x 50 x 50 x 4 x 0.2μs
;                                       = (Wの内容) x 2ms だけ遅延させる
;*******************************
DELAY:
        MOVWF   CNT0            ; Wの内容をCNT0へ入れる
DELAY_0:
        MOVLW   0x32            ; CNT1の内容を50にする
        MOVWF   CNT1            ;
DELAY_1:
        MOVLW   0x32            ; CNT2の内容を50にする
        MOVWF   CNT2            ;
DELAY_2:
        NOP
        DECFSZ  CNT2,F          ; CNT2から1を引いて、
        GOTO    DELAY_2         ;       0以外ならDELAY_2へ跳ぶ
        DECFSZ  CNT1,F          ; CNT1から1を引いて、
        GOTO    DELAY_1         ;       0以外ならDELAY_1へ跳ぶ
        DECFSZ  CNT0,F          ; CNT0から1を引いて、
        GOTO    DELAY_0         ;       0以外ならDELAY_0へ跳ぶ
        RETURN
;-------------------------------
        END                     ; プログラムの終わり
```

リスト１　switch.asm（つづき）

１.４　ＡＤ変換、ＤＡ変換

実験２でスイッチをモータの回転速度制御のために使用したが、これは、スイッチがロボットのセンサとして利用できることを意味している。しかし、スイッチでは０／１のディジタル信号しか扱うことができない。連続的に変化できる量、例えば 0V～5V の値をとる電圧をセンサ入力してロボットの制御に使いたいときには、この連続量（アナログ信号）をディジタル信号へ変換してから PIC 等のコンピュータへ取り込むようにする。このように、アナログからディジタルへ変換したり、その逆にディジタルからアナログへ変換することを、それぞれ、ＡＤ変換、ＤＡ変換という。なお、PIC にはＡＤ変換機能が内蔵されている。

表３　ＤＡ変換

ディジタル入力 a3,a2,a1,a0	アナログ出力 Vout／(Vcc/24)
０ ０ ０ ０	0
０ ０ ０ １	1
０ ０ １ ０	2
０ ０ １ １	3
０ １ ０ ０	4
０ １ ０ １	5
０ １ １ ０	6
０ １ １ １	7
１ ０ ０ ０	8
１ ０ ０ １	9
１ ０ １ ０	10
１ ０ １ １	11
１ １ ０ ０	12
１ １ ０ １	13
１ １ １ ０	14
１ １ １ １	15

ＤＡ変換回路の一例として、ラダー型ＤＡコンバータを図９に示す。図では、４個のスイッチによって、４桁（４ビット）の２進数 0100 を入力しているが、このときの出力電圧 Vout は Vcc の 1／6 となる。スイッチによって、a3, a2, a1, a0 の各点のディジタル入力信号を変えたときのアナログ出力信号 Vout は表３に示す通りとなる。

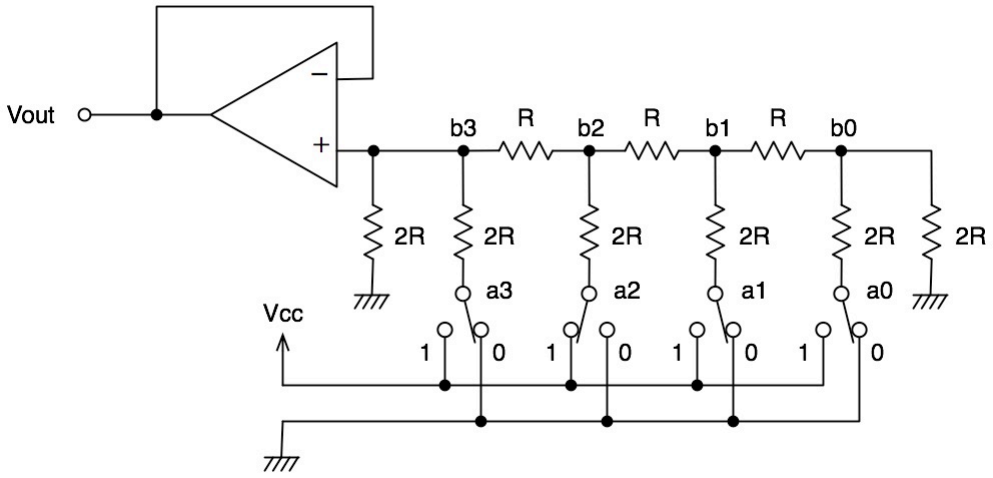


図９　ラダー型ＤＡコンバータ

ＤＡコンバータの各部の電流を図１０に示す。端子 a2 のみが電圧源 Vcc に接続され、他の端子は GND（0V）に接続されているが、a2 も Vcc を通して GND に接続されていると考えることができるので、b3, b2, b1, b0 の各点から下側を見た抵抗、左側を見た合成抵抗、

右側を見た合成抵抗はいずれも 2R となる。すると、a2 から流れてきた電流 I は b2 で I／2 ずつ右と左へと分かれ、b3 でさらに I／4 ずつ下と左へと分かれる（他の部分の電流も同様に求められる)。一方、a2 から見た全抵抗は 3R であるので、I＝ Vcc／(3R)である。したがって、Vout＝2R×I／4＝2R Vcc／(3R)／4＝Vcc／6 となる。

重ね合わせの理によって、複数のスイッチが ON のときの出力も、１個のスイッチのみが ON のときの出力電圧の算術加算によって求めることができる。また、ラダー（はしご）の数を増やすことで、４ビットより桁数の多いＤＡコンバータも同様に構成できる。ただし、ビット数が多ければ、ラダー回路部に使用される抵抗にはそれに応じた精度が要求される。

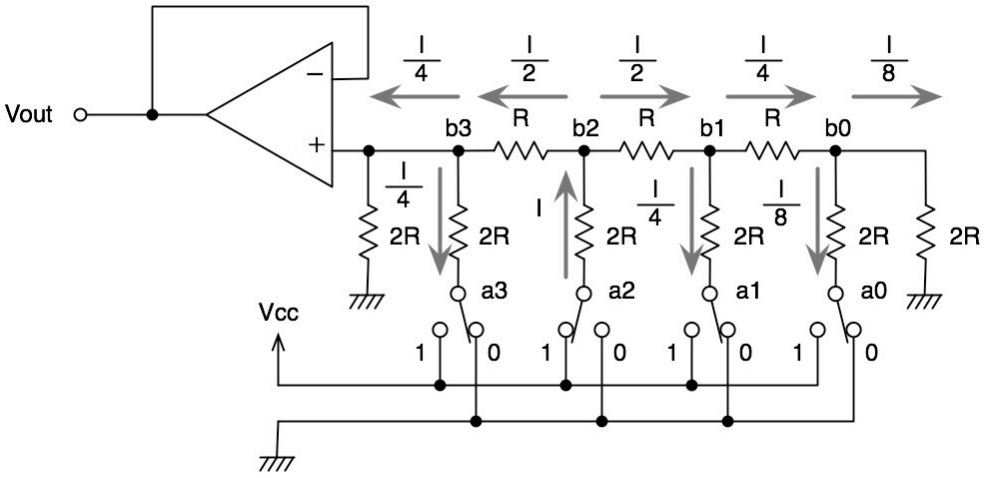


図１０　ＤＡコンバータの各部を流れる電流

ＡＤ変換回路の一例として、逐次比較型ＡＤコンバータ（図１１）について説明する。

逐次比較型ＡＤコンバータの内部にはラダー型ＤＡコンバータが内蔵されており、このＤＡコンバータの出力と測定対象のアナログ信号とをコンパレータで比較して、両者が等しくなるようにラダー回路部のスイッチを上位の方から逐次調整していく方式をとる。ＡＤコンバータ（ＤＡコンバータ)のビット数だけの比較と調整を繰り返せばよいので、比較的高速に変換が可能であるが、その変換精度はコンパレータとＤＡコンバータ部の精度によって決定される。

PICに内蔵されているＡＤ変換機能も逐次比較型である。

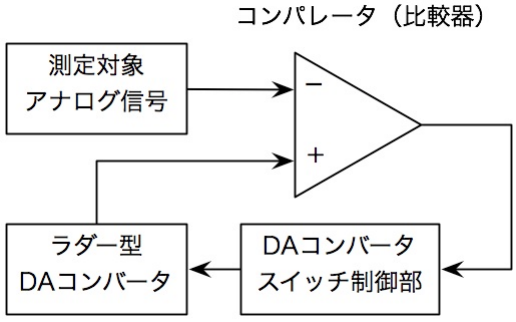


図１１　逐次比較型ＡＤコンバータ

〔実験３〕図１２に示すように、実験１における図５のスイッチに替えて、PIC16F877 を TA7291S に接続して、直流モータの回転方向と回転速度とを制御してみよ。この PIC には、端子 AN0（2 番 pin）へ入力しているアナログ電圧（0～5V）に応じて変化する PWM 信号を、端子 CCP1（17 番 pin）、CCP2（16 番 pin）から出力するプログラム（pwm.asm）を書き込んで、使用すること。

表４　使用器具・部品リスト〔実験３〕

品　名	規　格	数　量	備　考
モータドライバＩＣ	TA7291S	1	
可変抵抗器	10kΩ程度	1	
抵抗器（保護用）	3Ω	1	
ギア付き直流モータ	ZMP オリジナル	1	
直流電源（12V，5V）	ZMP オリジナル	1	
ブレッドボード		1	
ジャンパワイヤ		多数	
PIC	PIC16F877	1	
セラロック	CSTLS20M0	1	

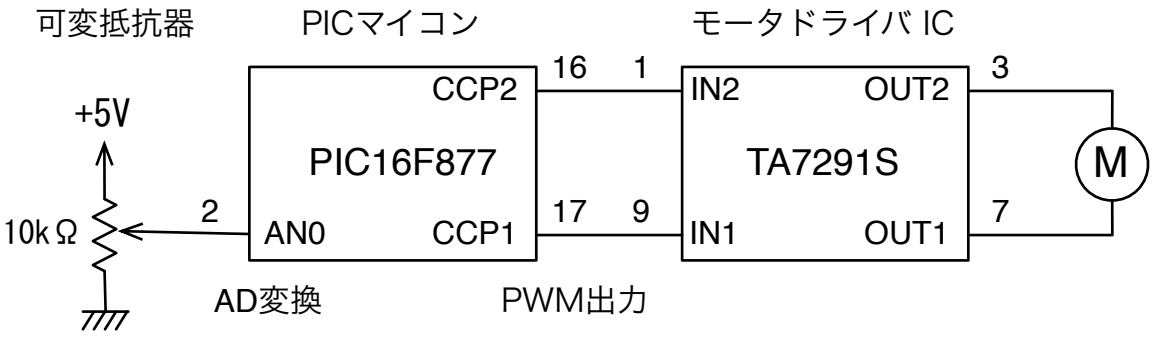


図１２　〔実験３〕実験回路図

```
; pwm_a.asm                                                                    2018.08.02
; pwm.asm                                                                      2018.06.20
;                                                                             2006.07.07
;                                                                             by M.Yanaka
;       Puls Width Moduration
;
        LIST    P=16F877 ; マイコンはPIC16F877
        INCLUDE  P16F877.INC      ; インクルードファイルを指定
;*******************************
;       コンフィギュレーションビットの設定
;*******************************
        __CONFIG _HS_OSC & _WDT_OFF & _PWRTE_ON & _CP_OFF & _BODEN_OFF &  _WRT_ENABLE_ON &
_LVP_OFF & _DEBUG_OFF & _CPD_OFF
                                ;       _HS_OSC           HS高速発振
                                ;       _WDT_OFF ウォッチドッグタイマはOFF
                                ;       _PWRTE_ON         パワーアップタイマはON
                                ;       _CP_OFF           コードプロテクトはOFF
                                ;       _MCLRE_ON         マスタクリアはON
                                ;       _BODEN_OFF        低電圧リセットはOFF
                                ;       _WRT_ENABLE_ON    プログラムメモリ書き込みはON
                                ;       _LVP_OFF 低電圧書き込みはOFF
                                ;       _DEBUG_OFF        オンボードデバッグはOFF
                                ;       _CPD_OFF EEPRROMプロテクトはOFF
;--------------------------------
Reg_20  EQU     H'20'           ; ソフトウェアタイマー用
AD_DATA EQU     H'21'           ; A/D変換結果格納用
;--------------------------------
        ORG     0               ; リセットベクタ(プログラムアドレス0番地)
        GOTO    START           ; STARTへ跳ぶ
;--------------------------------
        ORG     4               ; 割り込みベクタ(プログラムアドレス4番地)
        GOTO    START           ; （今回は割り込みを使用しない）
;--------------------------------
        ORG     8
;*******************************
;       start（初期設定）
;*******************************
START:
        CALL    INIT_PORT       ; 初期設定１（ポート入出力設定）をコール
        CALL    INIT_PWM ; 初期設定２（PWM設定）をコール
        CALL    INIT_AD         ; 初期設定３（ＡＤ変換設定）をコール
;*******************************
;       main loop
;*******************************
LOOP:
        CALL    AD_CONV         ; ＡＤ変換実行
        BTFSS   AD_DATA,7       ; 回転方向の識別
        GOTO    LEFT            ;       AD_DATA < 80h ならば、左回転
RIGHT:                          ; そうでなければ、右回転
        CLRF    CCPR2L          ; CCP2のPWMデューティーサイクルを最小(0)にする
        RLF     AD_DATA,W
        MOVWF   CCPR1L          ; CCP1のPWMデューティーサイクルを設定
        GOTO    LOOP
LEFT:
        CLRF    CCPR1L          ; CCP1のPWMデューティーサイクルを最小(0)にする
        COMF    AD_DATA,F
        RLF     AD_DATA,W
        MOVWF   CCPR2L          ; CCP1のPWMデューティーサイクルを設定
        GOTO    LOOP

;*******************************
;       初期設定１（ポート入出力設定）
;*******************************
INIT_PORT:
        BSF     STATUS,RP0      ; バンク1へ切り替えて、
        MOVLW   H'FF'           ; ポートAは、
        MOVWF   TRISA           ;       すべて入力ピンにする
```

リスト２　pwm.asm

```
        MOVLW   H'FF'           ; ポートBは、
        MOVWF   TRISB           ;       すべて入力ピンにする
        CLRF    TRISC           ; ポートCはすべて出力ピンにする
        MOVLW   H'FF'           ; ポートDは、
        MOVWF   TRISD           ;       すべて入力ピンにする
        CLRF    TRISE           ; ポートEはすべて出力ピンにする
        BCF     STATUS,RP0      ; バンク0へ切り替える
        RETURN
;*******************************
;       初期設定２（PWM設定）
;*******************************
INIT_PWM:
        BSF     STATUS,RP0      ; バンク1へ切り替えて、
        MOVLW   H'FF'           ; タイマー２（PWM用）の周期を、
        MOVWF   PR2             ;       最大(100h)にセットする
        BCF     STATUS,RP0      ; バンク0へ切り替える
        MOVLW   H'0C'
        MOVWF   CCP1CON         ; CCP1をPWMモードにセットする
        MOVWF   CCP2CON         ; CCP2をPWMモードにセットする
        CLRF    CCPR1L          ; CCP1のPWMデューティーサイクルを最小(0)にする
        CLRF    CCPR2L          ; CCP2のPWMデューティーサイクルを最小(0)にする
        MOVLW   H'06'           ; タイマー2のプリスケーラを、
        MOVWF   T2CON           ;       16にセットし、タイマーをオンにする
        RETURN
;*******************************
;       初期設定３（ＡＤ変換設定）
;*******************************
INIT_AD:                ; A/D 変換器の設定１(AN0)
        BSF     STATUS,RP0      ; バンク1へ切り替えて、
        MOVLW   H'02'           ; ADCON1(Reg_09F) <- 2
        MOVWF   ADCON1          ;       変換結果左詰、RE:Degital, RA:Analog
        BCF     STATUS,RP0      ; バンク0へ切り替える
        MOVLW   H'80'           ; ADCON0(Reg_01F) <- 80h
        MOVWF   ADCON0          ;       RA0/AN0 をアナログ入力に用いる
        RETURN                  ;       変換クロック <- システムクロック / 32
;*******************************
;       ＡＤ変換
;*******************************
AD_CONV:                ; A/D 変換実行１(AN0)
        MOVLW   H'81'           ; ADCON0(Reg_01F) <- 81h
        MOVWF   ADCON0          ;       A/D 変換モジュールオン(ADON(ADCON0.0) <- 1)
        CALL    WAIT_20         ; アクィジション時間待ち（ソフトウェアタイマー ：20.0μs)
        BSF     ADCON0,GO       ; A/D 変換開始(GO/DONE(ADCON0.2) <- 1)
WAIT:
        BTFSC   ADCON0,GO       ; A/D 変換終了(GO/DONE(ADCON0.2) = 0)を
        GOTO    WAIT            ;       待つ
        MOVF    ADRESH,W ; A/D 変換結果を取り込む(W_reg <- ADRESH(Reg_01E))
        MOVWF   AD_DATA         ;       結果を AD_DATA へ格納する
        RETURN
;*******************************
;       ソフトウェアタイマー
;*******************************
; Reg_20 はソフトウェアタイマー用作業レジスタ
WAIT_20:                ; ソフトウェアタイマー (20.0μs)
        MOVLW   H'20'           ; (0.2μs * 3 * 32) + (0.2μs * 4)
        MOVWF   Reg_20
        NOP
W_20:
        DECFSZ  Reg_20
        GOTO    W_20
        RETURN
;--------------------------------
        END                     ; プログラムの終わり
```

リスト２　pwm.asm（つづき）

１.５　センサを用いたソフトウェアサーボ

サーボシステムとは、物体の位置、方位、姿勢などを自動制御するシステムである。検出部（計測部）で制御対象の状態を調べてフィードバックし、それを目標値と比較してその差（偏差）が小さくなるように駆動部を制御する。ソフトウェアサーボとは、この内の一部分をコンピュータのプログラムによって実現するものである（図１３）。

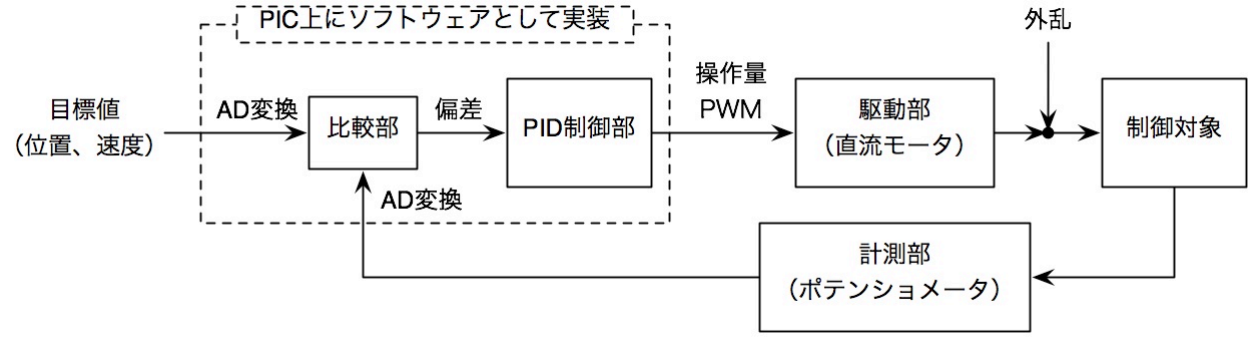


図１３　ソフトウェアサーボ

〔課題実験〕PIC16F877 のＡＤ変換機能と PWM 出力機能を利用して、ポテンショメータ内蔵ギア付き直流モータの角度制御を行なう。図１４に示すように、PIC16F877 の PWM 出力をモータドライバ TA7291S に接続し、角度の目標値を設定する可変抵抗器（半固定抵抗器）とモータ内蔵のポテンショメータとをそれぞれ PIC16F877 のＡＤ変換入力端子へ接続する。PIC には、２つのアナログ入力端子 AN0 と AN1 の信号をＡＤ変換入力して、その差を求め、差が正ならばモータを正転、負ならば逆転させるような PWM 信号を出力するプログラムを、各自で作成して、書き込み、使用すること。実験結果として、可変抵抗器を回すと、それに応じてギア付きモータの軸が回転することを確認せよ。

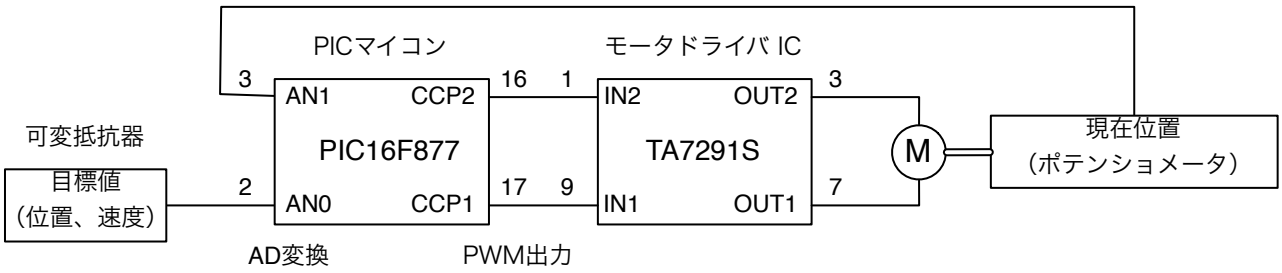


図１４　〔課題実験〕実験回路図