平成 17 年度秋田工業高等専門学校卒業論文

『２足歩行ロボットの設計と製作』

平成 18 年 3 月

報告者　　機械工学科５年　13-12　　菊地　賢

指導教員　　木澤　悟

目　　次

第一章　緒言

1－1　研究背景

近年,人間との協調・共存を目指したロボットの研究開発が盛んである.ロボットがそのような状況で作業を行う状況を想定したとき常に作業環境,作業方法などの情報が事前に十分に与えられるとは考えづらい.そのためロボットが環境や作業に適応しなければならない.

また少子高齢化社会に伴う労働人口の減少や要介業者の増加に備え,人間自身のメカニズムを早急に解明することも必要となってきており、愛知万博で展示されるほどロボットへの関心は高まってきている.

そこで本研究では人間の行動で代表的な「二足歩行」を解析するため，昨年度の卒業研究の課題であった軽量化と発展性を持たせることを目標に二足歩行ロボットを製作し、設計や製作における妥当性を検討することが目的である.

第二章　実験装置システムの概要

２－１　マイコンボード（SH2―7045F）

二足歩行ロボットを実現させるため,タイマやA/D変換器といった機能を十分に有した制御用マイクロコンピュータが必要である.そこで本研究では昨年度同様に，日立製作所製32ビットマイコンであるSH2-7045Fを使用した.仕様については表2-1に示す.また実際の写真をFig.2-1に示した.さらにFig2-2にSH2-7045Fのブロック図を示した.

表2-1.CPUの仕様

CPU	HD64F7045F28 ○内部32ビット構成 ○汎用レジスタマシン －汎用レジスタ 32ビット×16本 －コントロールレジスタ 32ビット×3本 －システムレジスタ 32ビット×4本 ○RISCタイプの命令セット －命令長：16ビット固定長によるコード効率の向上 －ロードストアアーキテクチャ －遅延分岐命令の採用で，分岐時のパイプラインの乱れを軽減 －C言語指向の命令セット ○命令実行時間　1命令／1サイクル（28MHz） ○乗算器内蔵 ○パイプライン　5段パイプライン方式
キャッシュメモリ	○1kB命令キャッシュ ○命令コード及びPC相対読み出し・データをキャッシング ○内蔵RAMと兼用，キャッシュイネーブル時は内蔵RAMのうち2kBをアドレスアレイ・データアレイとして使用
割り込みコントローラ	○外部割り込み端子×9本 ○外部割り込み要因44要因
その他	大容量ROM・RAM・タイマ・A/D変換器・I/Oポート・ シリアルコミュニケーションインターフェース等を内蔵

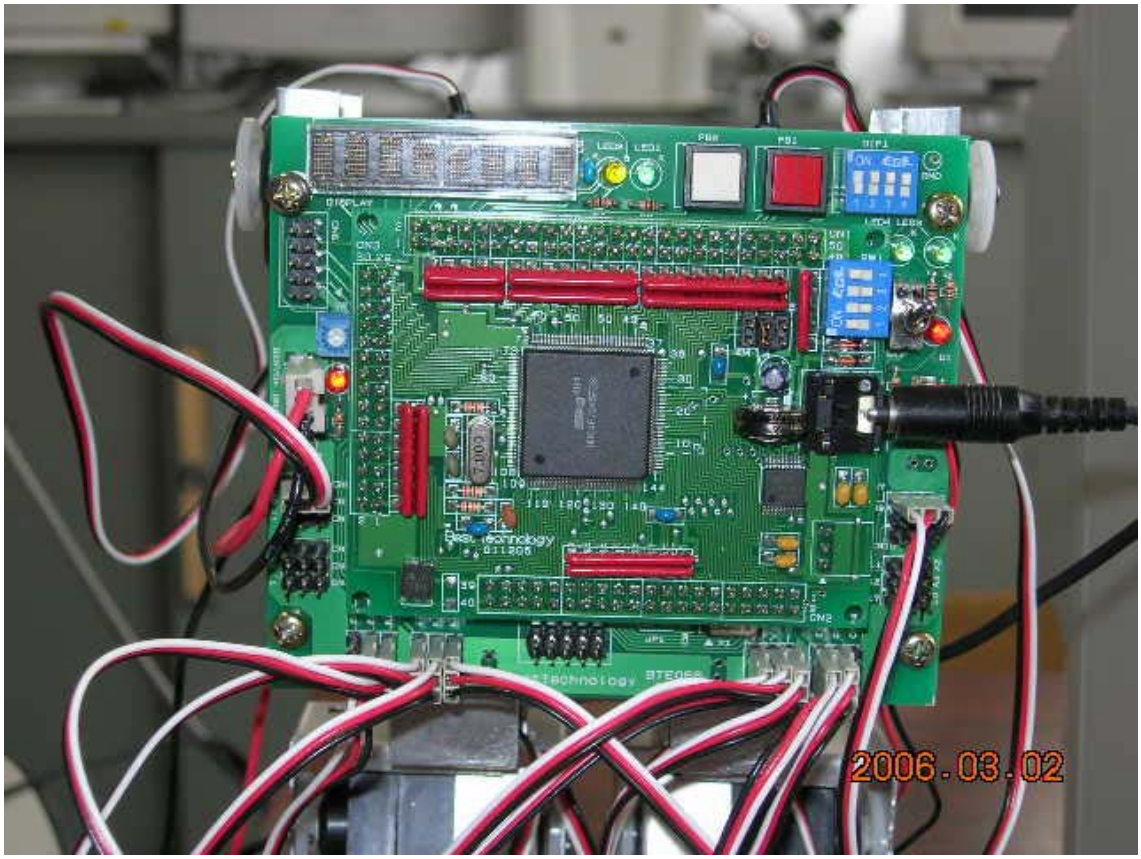


Fig.2-1 SH2－7045F

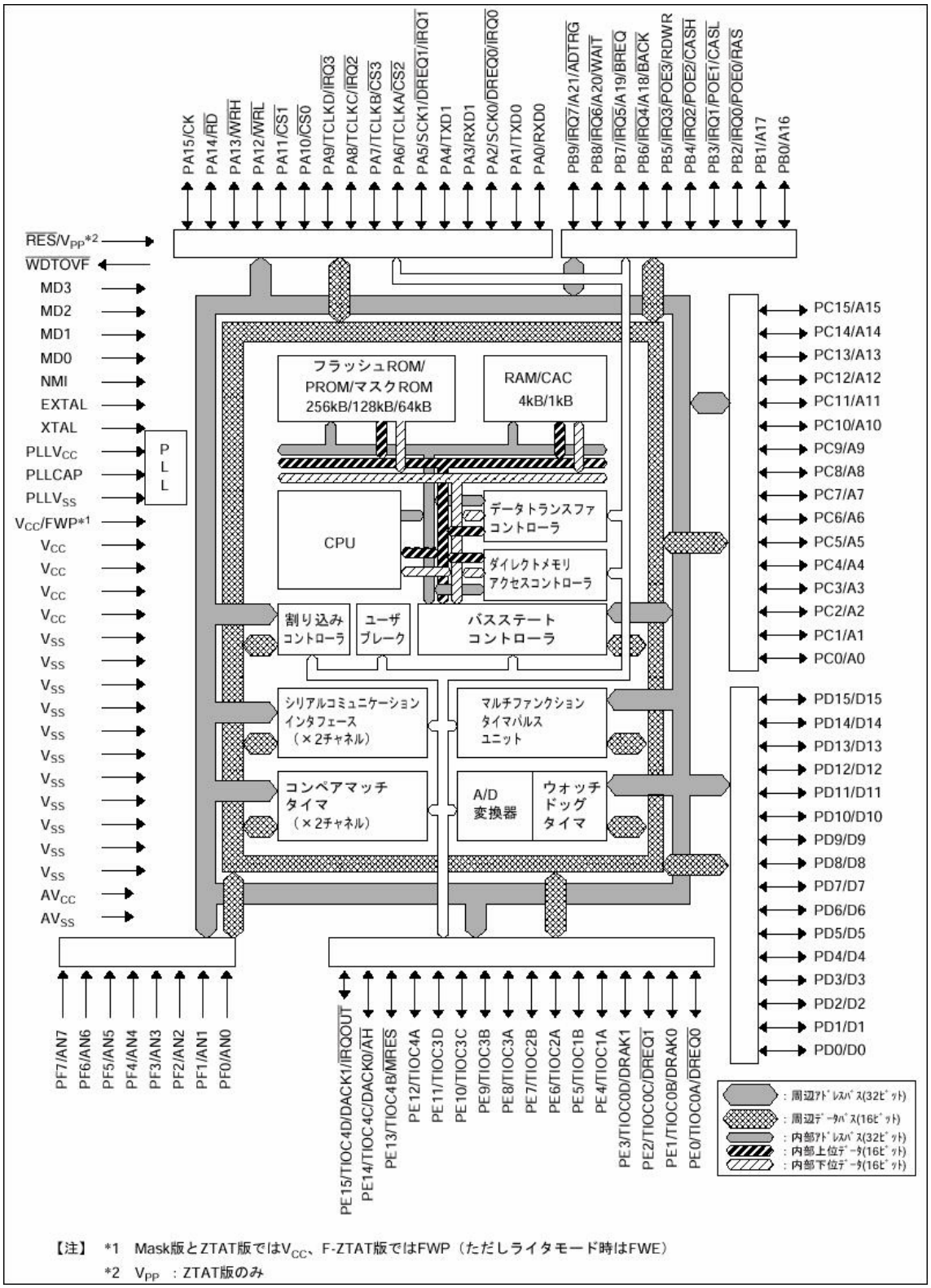


Fig.2-2 SH2-7045F のブロック図

２－２　開発環境

SH2-7045F マイコンボード用のプログラム開発環境としては，ボードに付属されている統合開発環境ソフトウェア GCC Developer Lite 及びコンパイラ等ツールソフトウェア GNU Pro Toolkit を Windows 上で動作させ，これを用いることで，プログラムの編集，コンパイル，CPU への書き込みを行う．

本研究では､コンピュータと SH2-7045F とを, Fig2-3 のように通信ケーブル(RC-232C)を用いて繋ぎ, コンピュータで C 言語を用いて作成したプログラムを SH2-7045F におくり,それらに対応するモータに送られたプログラムを実行することで制御している．
転送手順を表２－２に示した．

表 2-2 プログラム転送方法

１．ツールの SIMPLE　TERM をクリック．
２．マイコンボート上の「ディップスイッチ４」をオンにしたまま，マイコンボートの電源を供給する．(「ディップスイッチ４」をオフにしたままだと，以前のデータが保護されている状態にある．)
３．すると文字列が表示される．
４．マイコンボードの電源をオフにする．
５．「通信」の「ポート　オープン」をクリック
６．マイコンボート上の「ディップスイッチ４」をオンにしたままマイコンボートに電源を供給する．(制御電源を ON にする．)（マイコンボードは受信待機中になっている.）
７．「SIMPLE　TERM」の「転送」をクリック．(転送開始)

プログラム編集後のコンパイル手順
1.「ツール」の「GCC オプション」の順にクリック
2.「設定リストを SH2　７０４５F」増設 SRAM にして「OK」をクリック．
3. コンパイルする．

Fig.2-3　二足歩行ロボット開発環境

第三章　二足歩行ロボット

３－１　基本構造

本研究で製作した二足歩行ロボットは，軽量化を目指すため、厚さ１ｍｍのアルミ板を加工して製作した．これは低価格で比重が小さく，単位重量あたりの強度があるアルミ板を使用し，曲げ加工を多く用いることでネジによる接合を極力抑えるためである．ロボットの全長は 302ｍｍ，重量は 1.11ｋｇであった．

組み立て図を Fig3-1 に示し，　ロボットの写真を Fig3-2 に示す．

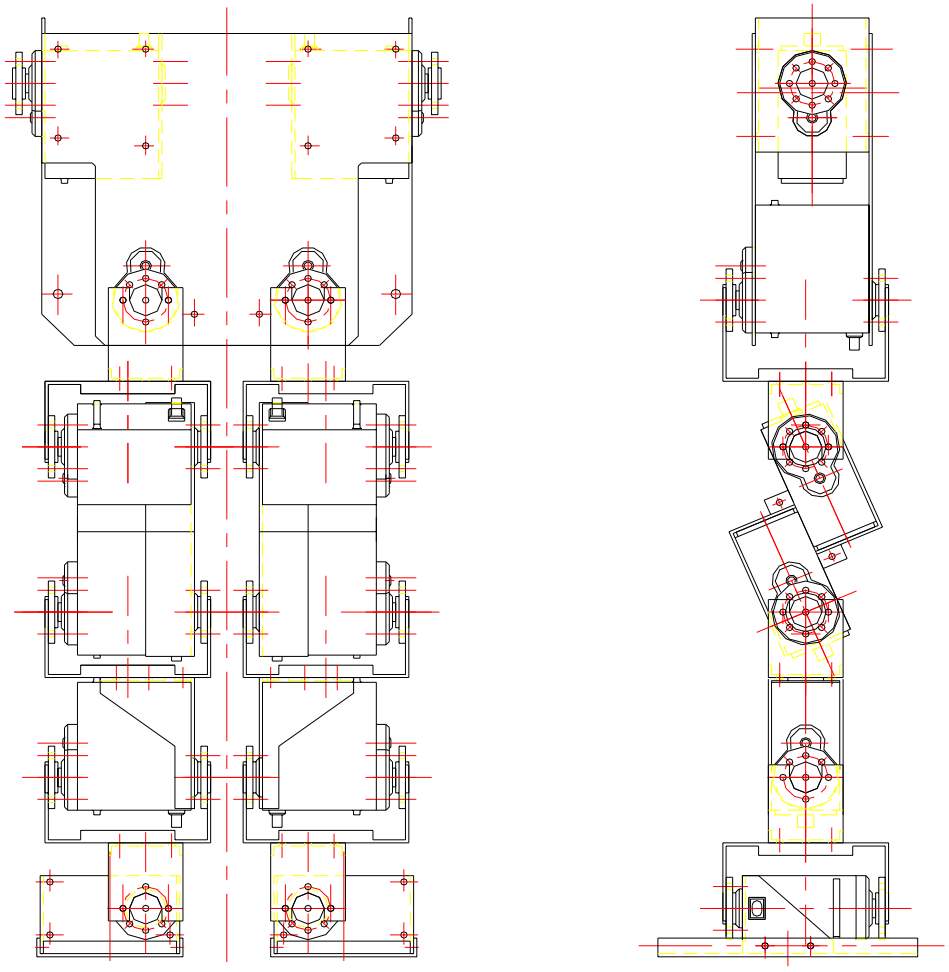


Fig３－１　二足歩行ロボットの組み立て図

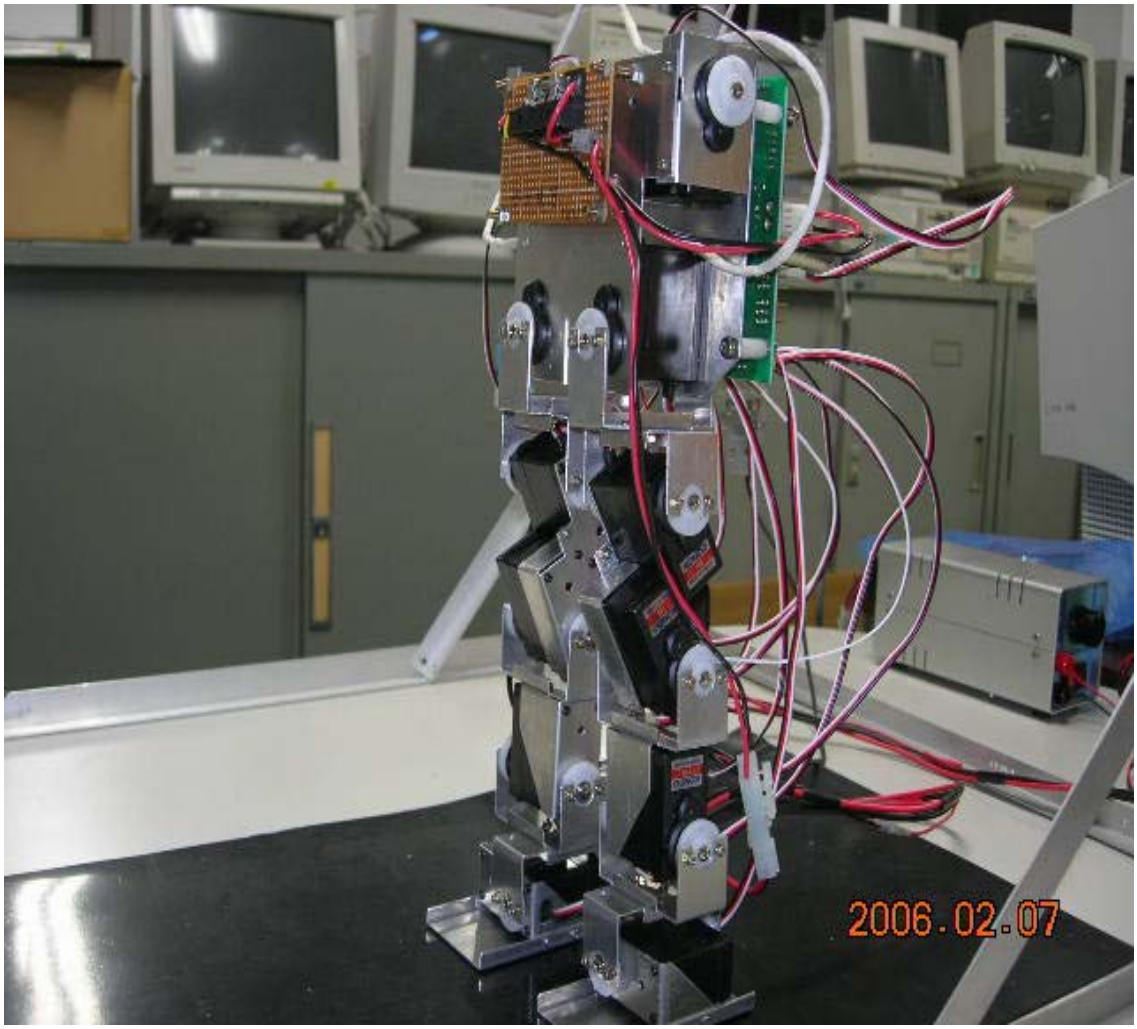


Fig３－２　二足歩行ロボット外観

３－２　ブラケット

ブラケットとはサーボモータ同士を繋ぐ部品である．つまりこのロボットは大きく分けるとサーボモータと制御基盤とブラケットでできている事になる．

昨年度卒業研究では市販ブラケットも使用していたが，今回はすべて自作する事となった.そこで二足歩行ロボットに関する本やインターネット上の HP などを参考に，ロボットのブラケットの形状を決定，今回使用するサーボモータの寸法に合わせてブラケットの寸法を決め，設計した．

３－３　駆動部

今回は Fig3-1 からわかるように、駆動部が片足５カ所の合計１０カ所ある．駆動部として用いられているサーボモータは、近藤科学社(株)の KRS786ICS を用いている．

３－３－１　サーボモータ

制御回路を含むモータモジュールである。この制御回路に、制御パルスを送ると、そのパルス幅に比例した角度にまで回転する。パルスの幅と回転角度の対応は、それぞれのサーボモータで、少しずつ違ってくる.

３－３－２　サーボモータの仕様

サーボモータの図と仕様を以下に示した.

Fig3-３ KRS786ICS

表 3-1　KRS786ICS の仕様

販売元	近藤科学
低格電圧	6.0(V)
スピード	0.17(sec/60°)
トルク	8.7(kg/cm)
重量	45(g)
寸法	41×35×21(㎜)

３－４　サーボモータの制御

サーボモータを任意の角度まで回転させるためには，制御パルスのパルス幅を変調させることが必要となる．これを PWM（パルス幅変調）方式といい，サーボモータの制御に不可欠なものといえる．使用した CPU である SH2-7045F にはインターバルタイマ機能*が内蔵されているため，インターバルごとに任意のパルス幅をサーボモータに与えることで PWM 出力を行うことが可能であり，これを用いて制御を行う．なお，この制御は一方的に制御パルスを与えフィードバックは行わないオープン制御である．

サーボモータと SH2-7045F マイコンボードの接続基盤の構造は Fig3-4 のようになっている．また，I/O ポートのサーボモータ用のピン配列は Fig.3-5 の示す．なお，図中の S 表示はシングル，D 表示は，切り替えによるポートである．この切り替え方法は本論文の P12 にて説明する．

(注＊インターバルタイマについては，４章にて説明．)

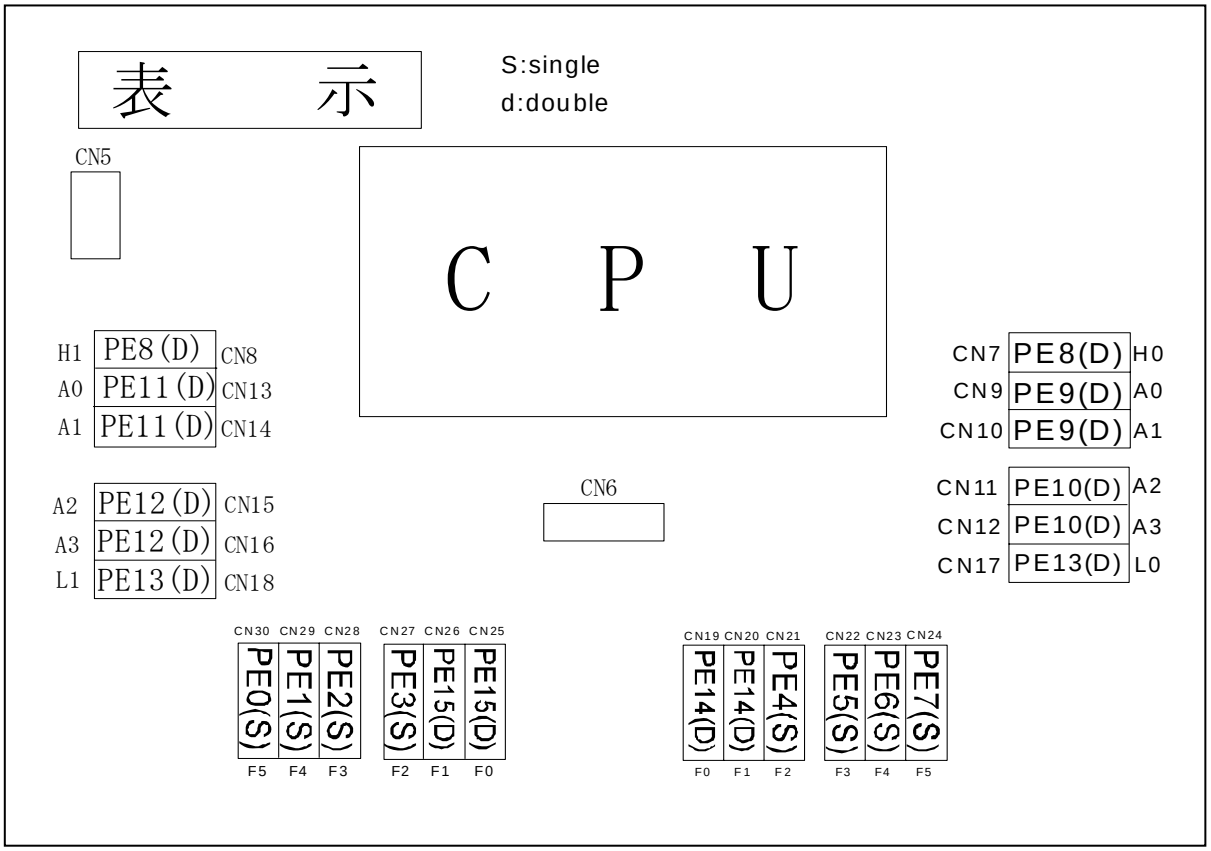


Fig.3-4　SH2―7045F マイコンボードの接続基盤の構造

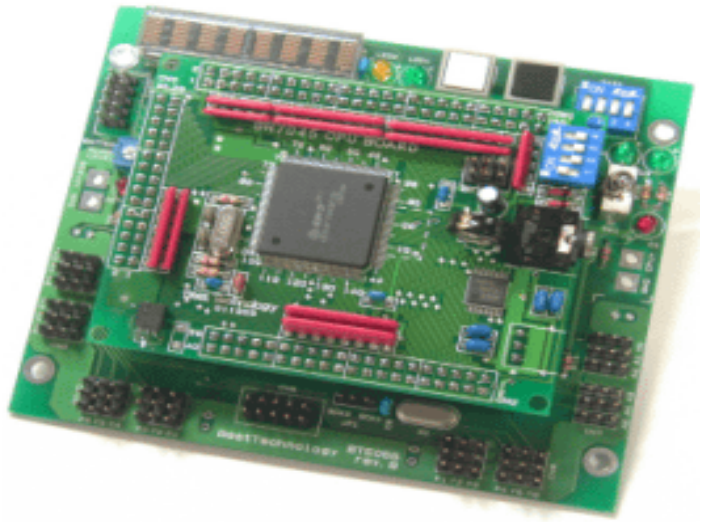


Fig3-5　I/O ボード

このボードでは，16 本の PWM しか使うことができないが，足のモータだけで，片足５本，両足で計１０本の PWM を使うことになる。今は 16 本あれば，十分足りるが，今後改良すると 16 本だけでは，足りないということが考えられる．そこで，PWM を増やしてやる必要がある．PWM を増やす方法には次の 2 つがあげられる．

I/O ポートで PWM を出力する方法

PWM を切り替えて出力する方法

今回は PWM を切り替えて出力する方法を使用する．サーボモータの PWM については，Fig.3-6 のようになっている.図中のパルス幅は、1.５ms になっているが、これはサーボモータのニュートラル位置に対応している．

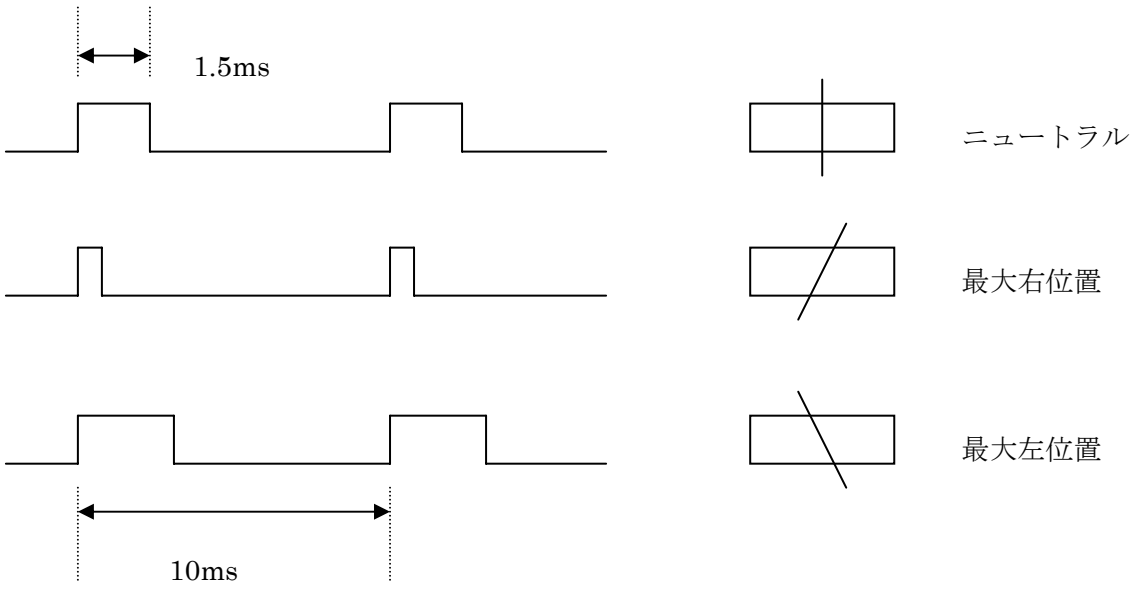


Fig.3-6　サーボモータの PWM

PWM の周期は一般に 10msec であり，サーボではこの周期を短くすることができる．今回はモータと CPU の相性から 10ms を使用している．

次にサーボ出力のためのチャンネル数を増やす方法を述べる.Fig.3-7 に示すような時間的な割り込み技術を用いればハードウェア的な切り替え方法が使用できる．つまり，1ch の PWM 出力を２つに分けてチャンネル数を増加させようというものである．

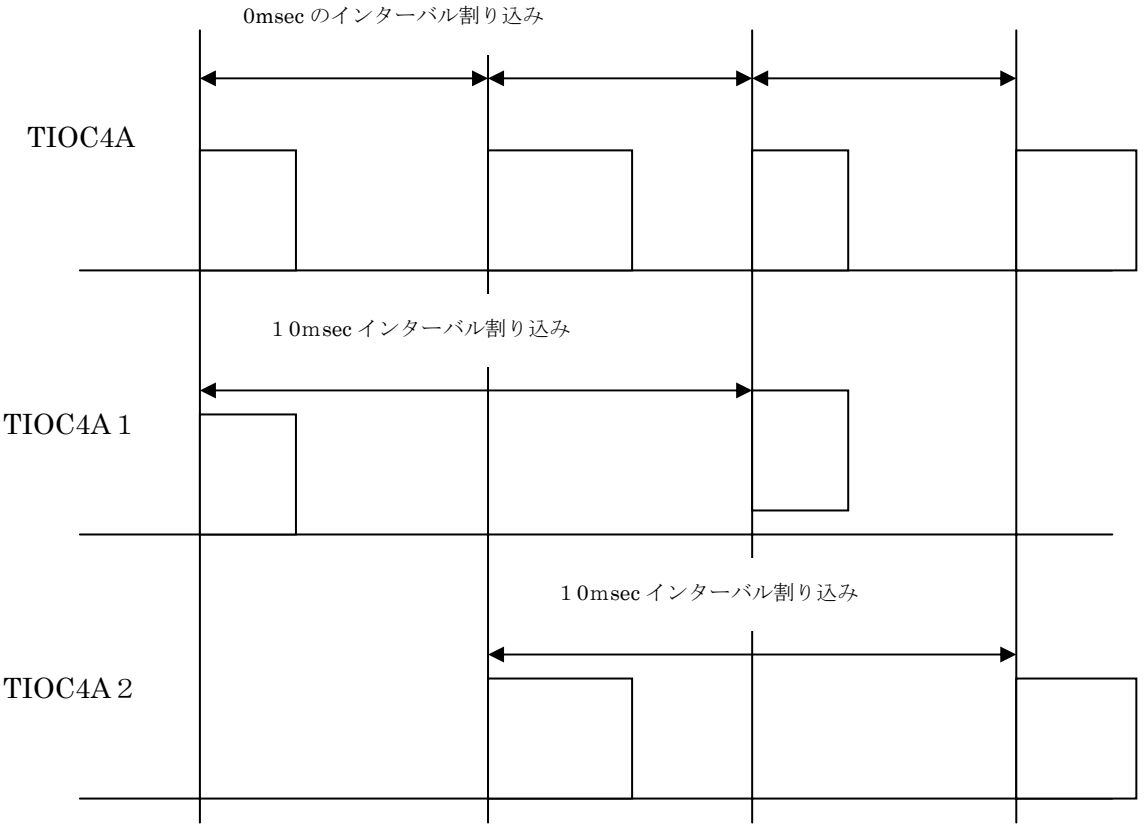


Fig.3-7　切り替え方法

Fig3-8 はチャンネルを切り替えるための回路図である．これには，AND の演算素子が使用されており．この回路を使用することでチャンネルを増加することができる．つまり，図中の TIOC4A1 から PWM 信号を出力したい場合は，PC16 を high にすればよい．TIOC4A2 から PWM 信号を出力したい場合は PC17 を high にすればよく，TIOC4A ポートを切り替えることで，二つの出力を可能としている．本来であれば，16 本が使用可能であるが，このうち 8 本のチャンネルを，二倍とし，計２４個のチャンネルを使用することができる．また出力端子の構造を Fig3－8 に示す．またその内の点線で囲まれた部分を Fig3-9 に示す。

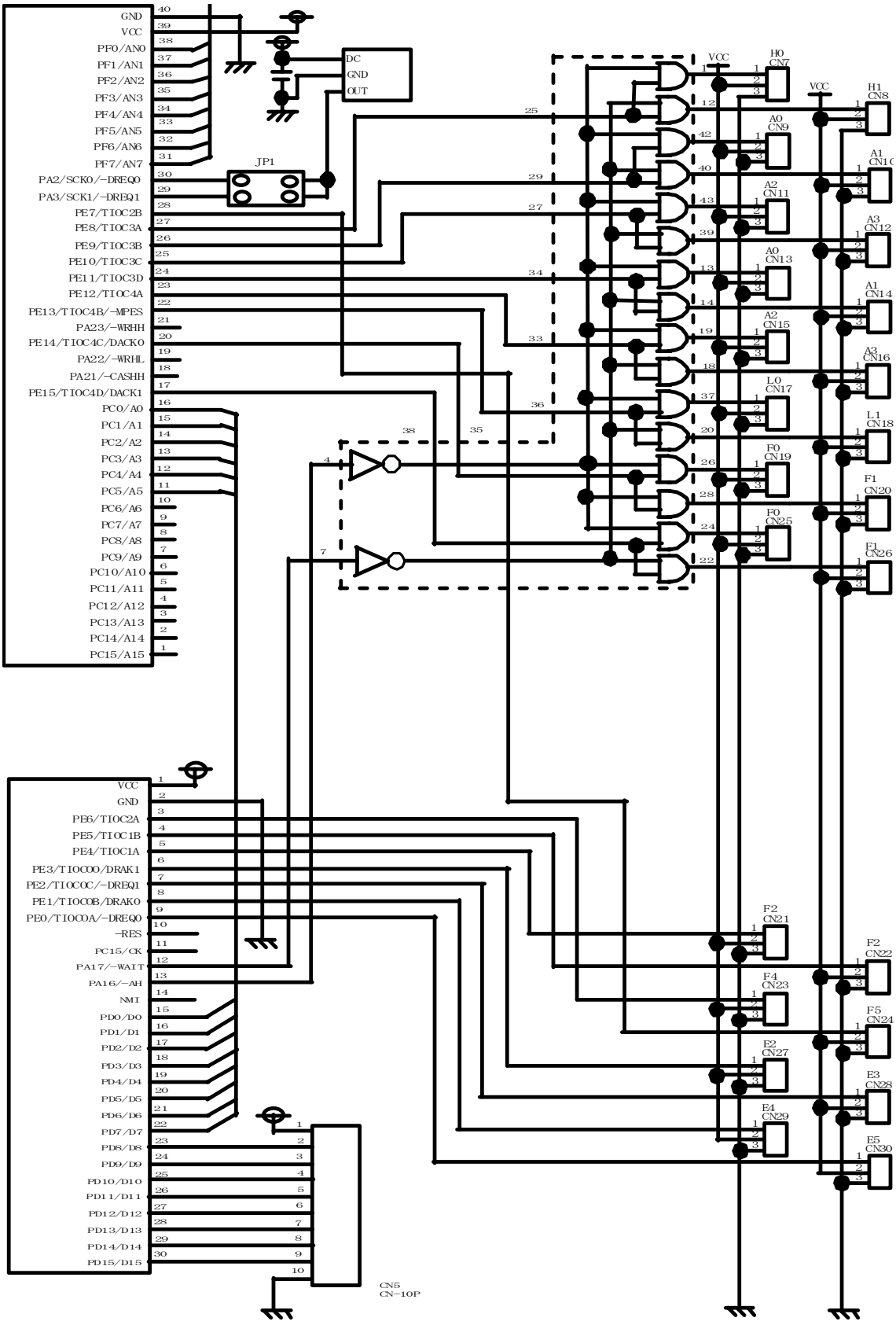


Fig.3-8 出力端子割り当て

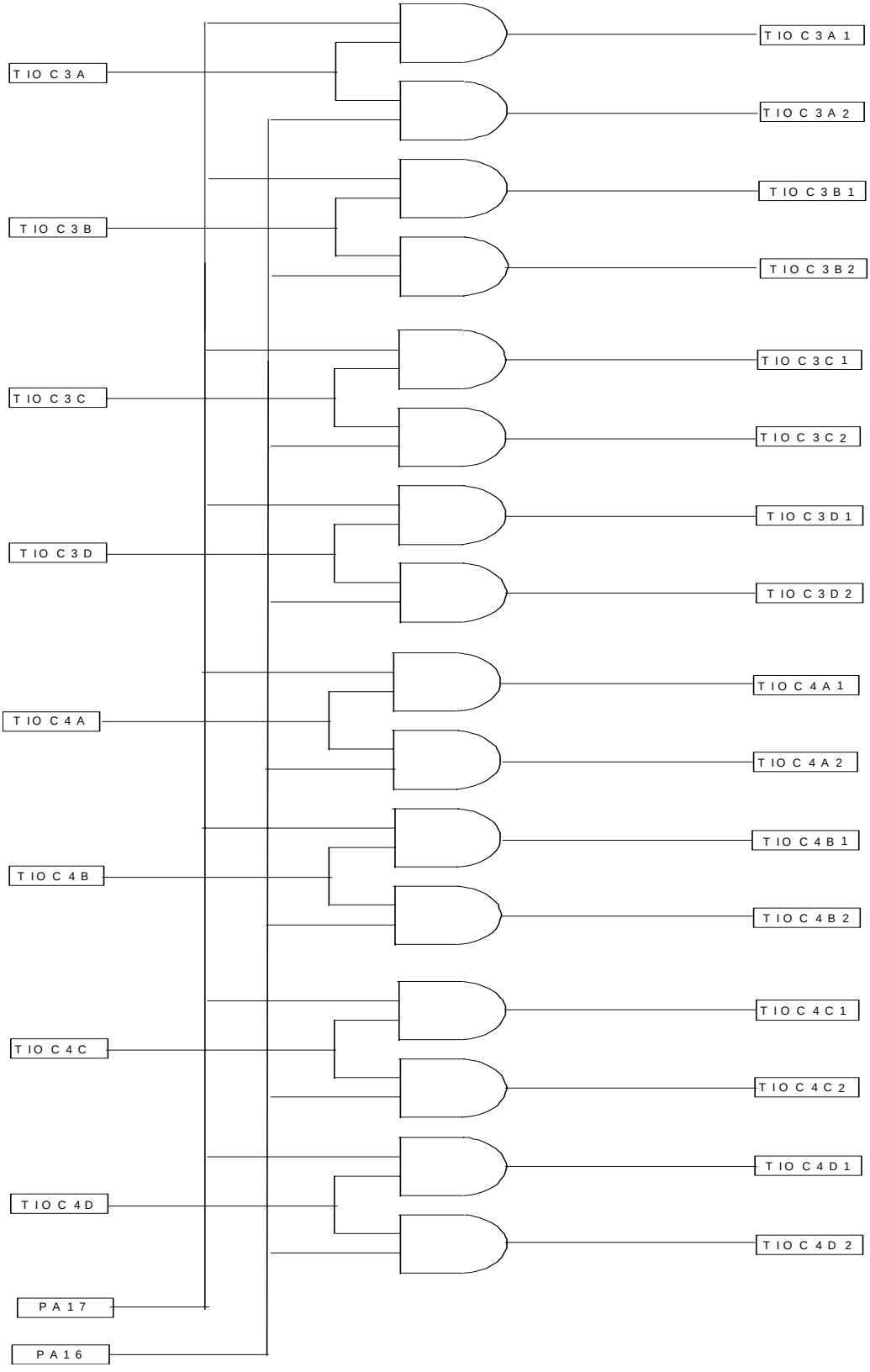


Fig.3-9 出力端子割り当て（拡大）

3－5 バッテリー

本研究ではサーボモータ用と制御用の二つのバッテリーを使用している.サーボモータ用のバッテリーの仕様を表 3-2 に制御用のバッテリーを表 3-3 に示す.

表 3-2　サーボモータ用バッテリーの仕様

種類	ニッカド電池
定格電圧	7.2V
定格電流	600mA
重量	125g

表 3-3　制御用バッテリーの仕様

種類	ニッカド電池
定格電圧	8.4V
定格電流	150mA
重量	45g

第四章　制御方法

４－１　制御方法

今回，本論文第 3 章　3. サーボモータ. で説明したサーボモータの制御方法は PWM 制御【パルス幅変調（Pulse Width Modulation）】とよばれる手法である.

４－２　PWM 制御

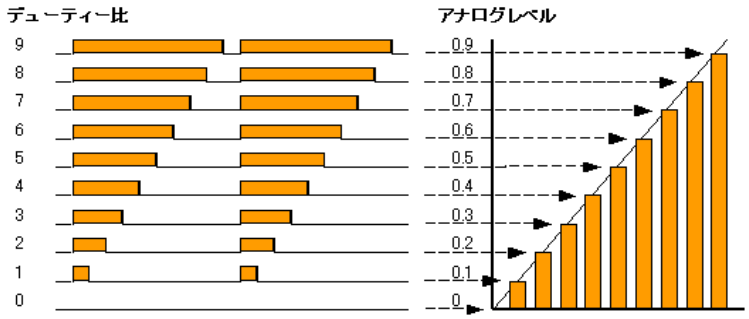


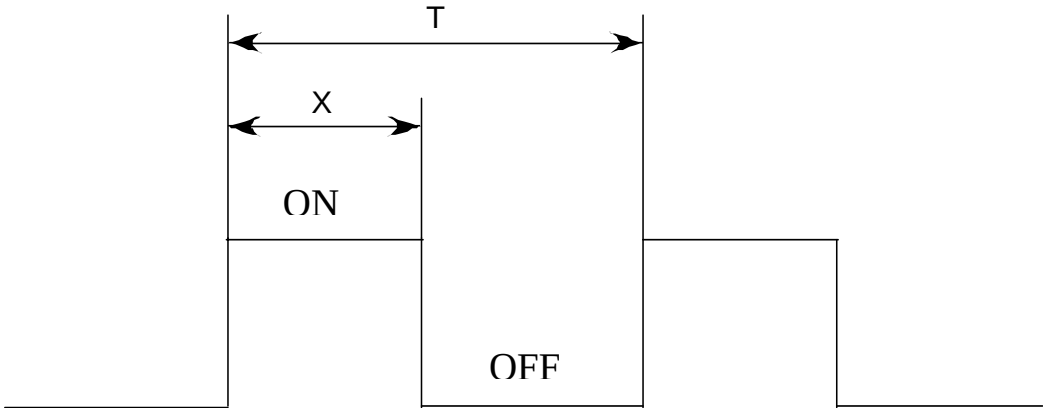
Fig4-1 PWM 制御

図の左側では，デューティー比が高いほどＯＮになっている期間が長く，デューティー比がゼロではずっとＯＦＦのままである様子を示している.

モータにこのようなタイミングで電源を与えると，あたかもアナログ的に電源電圧を可変したかのように，角度の制御が可能になる. これが PWM による制御である.

４－３　デューティー比

PWM 制御を用いるにあたって，デューティー比が必要となってくる．ディーティー比は周期 T と，on 時間 x の比である．
これより以下のようになる．



４－４　デューティー比とサーボモータの移動角度

$$duty = \frac{X}{T} \times 100[\%]$$

本研究では，デューティー比とサーボモータの移動角度の関係を調べるため，移動角度 5 度毎に時間ｄｔとパルス数をデータにとりさらに，それをデューティー比に変換した．その表を，表 4-1 に示す．

表４－１　デューティー比とサーボモータの移動角度（近藤科学　KRS786ICS）

角度(°)	パルス数		時間dt	デューティー比
−90	870	1340	0.73	7.3
−85	940	1270	0.77	7.7
−80	1020	1190	0.82	8.2
−75	1100	1110	0.87	8.7
−70	1180	1030	0.91	9.1
−65	1240	970	0.94	9.4
−60	1320	890	0.99	9.9
−55	1400	810	1.04	10.4
−50	1470	740	1.08	10.8
−45	1550	660	1.12	11.2
−40	1620	590	1.16	11.6
−35	1700	510	1.2	12
−30	1770	440	1.25	12.5
−25	1850	360	1.29	12.9
−20	1920	290	1.33	13.3
−15	1990	220	1.38	13.8
−10	2070	140	1.42	14.2
−5	2140	70	1.46	14.6
0	2210	0	1.5	15
5	2300	−90	1.55	15.5
10	2360	−150	1.58	15.8
15	2430	−220	1.62	16.2
20	2520	−310	1.67	16.7
25	2580	−370	1.71	17.1
30	2650	−440	1.75	17.5
35	2730	−520	1.8	18
40	2790	−580	1.83	18.3
45	2870	−660	1.88	18.8
50	2960	−750	1.93	19.3
55	3040	−830	1.97	19.7
60	3120	−910	2.02	20.2
65	3190	−980	2.06	20.6
70	3260	−1050	2.1	21
75	3330	−1120	2.14	21.4
80	3410	−1200	2.18	21.8
85	3490	−1280	2.23	22.3
90	3560	−1350	2.27	22.7

横軸にディーティー比，縦軸にサーボモータの移動角度をとったグラフに Fig4-3 に表した．ただしこのデータは無負荷状態における関係でありまたデューティ比が 15%の時をニュートラル値として表している．ここで T＝10msec である．

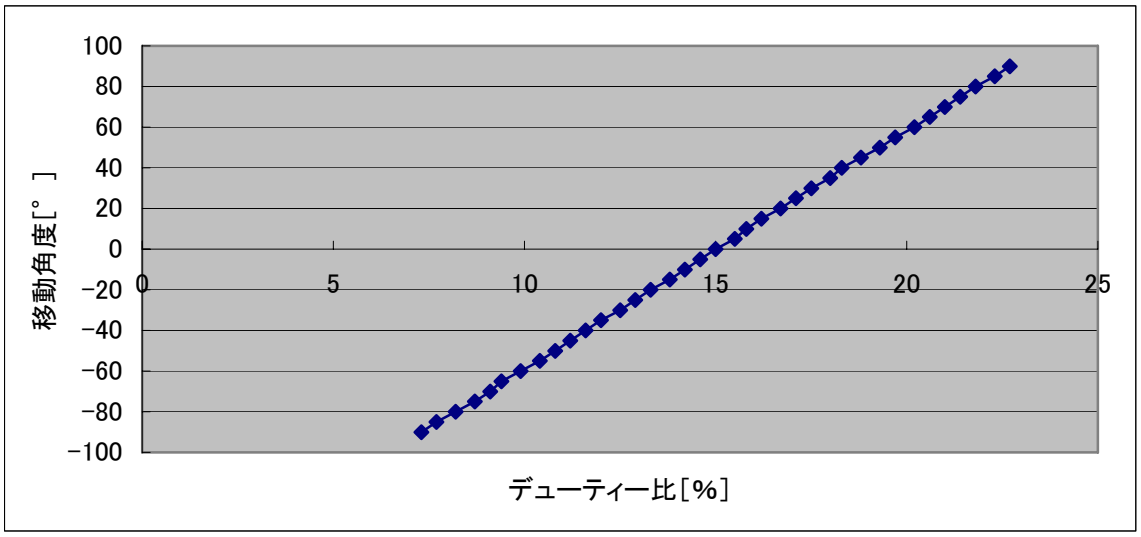


Fig4-3 デューティー比とサーボモータの移動角度の関係

これよりデューティー比とサーボモータの移動角度の関係は比例関係になっているとする．

４－５　パルス数とサーボモータの移動角度

本論文 4-4 節でデューティー比とサーボモータの移動角度の関係を説明したが, デューティー比では CPU が読み取ることができないため，CPU が読み取れる値に変えることが必要となってくる．

その読み取れる値パルス数と，デューティー比の関係は Fig4-4 のようになる．ただしこの状態は無負荷状態である．

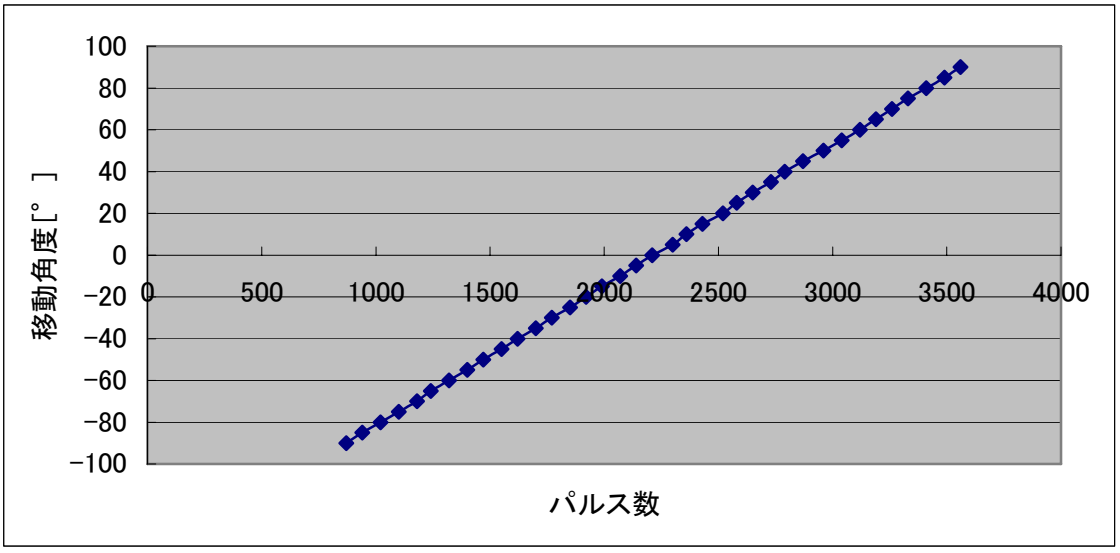


Fig4-4　パルス数とサーボモータの移動角度の関係

図中の直線は,

$$y = \frac{1}{14.5}(x - 2210) \quad \cdots (4\text{-}1)$$

の関係にある．

ただしここでは

ｙ：角度

ｘ：パルス数

である．また，2210 はニュートラルの値である．

４－６　割り込み処理プログラム

```
WDT．WRITE．TCSR＝0x5abc ――――――――――④
a＝WDT．READ．TCSR．BYTE ――――――――――⑤
WDT．WRITE．TCSR＝0xa53e ――――――――――⑥
```

プログラム④では，使用したい時間を指定している．なお今回のプログラムでは，割り込みを，１０ｍｓで発生させたいために，プログラム①で最大時の時間を３７．４ｍｓとしている．

・５ａはアドレスの指定をしている．

・最後のｂｃは１０ｍｓを表している．

１０ｍｓの算出方法

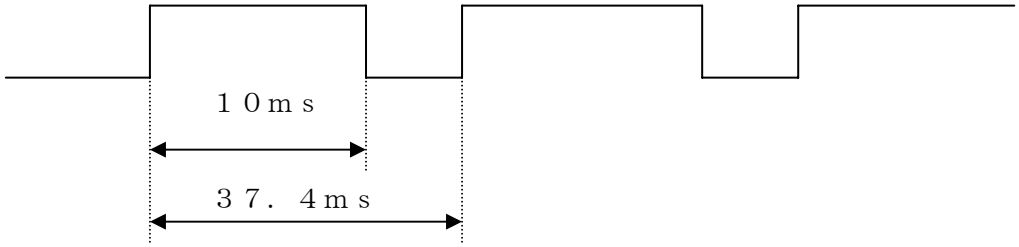
ｂｃは１６進数なので（０ｘより），１０進数に変換すると，１８８になる．

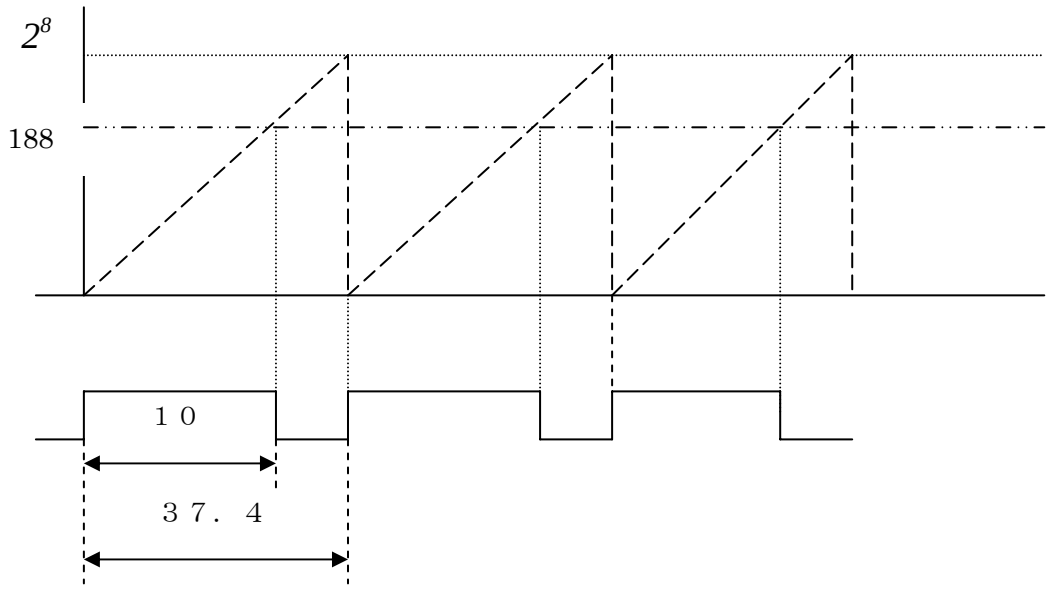
※ｂｃ（１６進数）⟶ １０１１　１１００（２進数)⟶ １８８（１０進数）

$$37.4 \times \left(\frac{256-X}{256}\right) = \text{使用したい時間}$$

Ｘに１８８を入れると今回のプログラムで使っている値が出てくる．

つまり１０ｍｓを使用したい場合は，上式の使用したい時間に１０ｍｓを代入しＸを求め，その値をプログラムに入力する（ただしＸは１０進数でてくるため．１６進数を使用する場合は，変換をしなければいけない．）





後章で説明するがここで作った時間 10msec は割り込み時間であり、ワンショットパルスを発生させるための周期時間を作り出すためのものである．このことを Fig4-5 に示す.

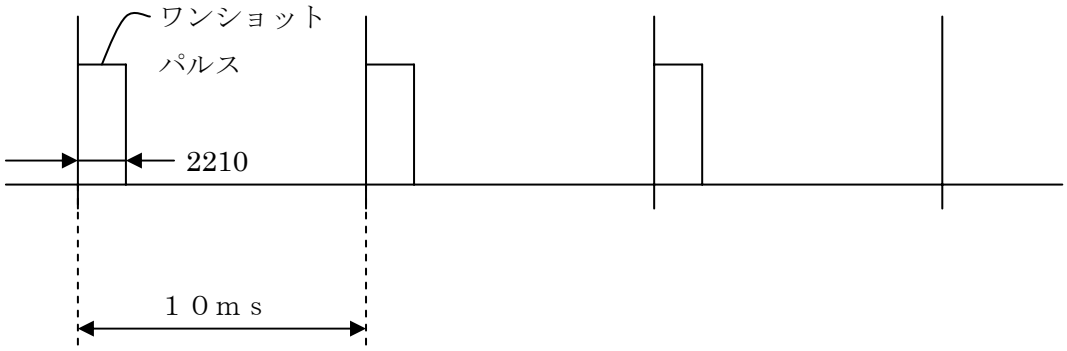


Fig4-5

プログラム⑤は VOF クリアする

プログラム⑥でプログラムの左辺はプログラム④と全く同じである．プログラム⑥ TCSR への書き込みであるが関数が同じである．そのため右辺の最初で a 5 とあるこれで TCSR への書き込みをしている.

４－７　　ウォッチドックタイマ

ウォッチドックタイマは本来は,システムの監視を行うためのタイマであるが,それと同時にインターバルタイマの機能があるのでこれを使用することができる.

インターバルタイマは,TCNT がオーバーフローするごとに,インターバル割り込みを発生する．この割り込みが発生したとき,１６本のタイマをワンショットで起動すれば,PWM を発生させることができる.

レジスタには**タイマコントロール／ステータスレジスタ（TCSR）,タイマカウンタ(TCNT),リセットコントロール／ステータスレジスタ(RSTCSR)**がありこれらを設定する必要がある.

1) タイマカウンタ(TCNT)

TCNT は,８ビットの読み書き可能なアップカウンタである．TCNT の値がオーバフロー（H'FF から H'００）すると,TCSR の OVF フラグが１にセットされる。ここで割り込みが発生するので,オーバーフローフラグを読み取り, TCNT に５msec のカウント値を書き込んでやる.

2) タイマコントロール／ステータスレジスタ（TCSR）

TCNT は,８ビットの読み書き可能なレジスタで,TCNT に入力するクロックやモードの選択を行う.

タイマモードセレクトは０でインターバルタイマである.

タイマイネーブルは１でカウントをスタートする.

クロックセレクトの CKS2－CKS０ビットで選択された内部クロックにより,カウントアップを開始する.

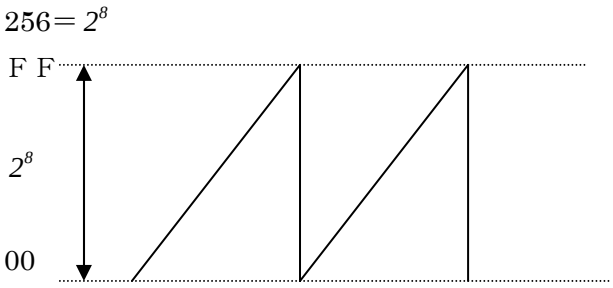
CKS2	CKS1	CKS0	
0	0	0	φ/2
		1	φ/64
	1	0	φ/128
		1	φ/256
1	0	0	φ/512
		1	φ/1024
	1	0	φ/4096
		1	φ/8192

OVF	WT/T	TME	---	---	CKS2	CKS1	CKS0

例えば，ここに 00111110Ｂ（＝3e（１６進数））を入れてやると

0	0	1	1	1	1	1	0

となる．これに対応しているＣＫＳをみると，φ/4096 となる



$$\frac{28MHz}{4096} = 6835.9375\,H_z$$

$$T = \frac{1}{6835.9375}\ \mathrm{S} = 0.000146285\ \ \mathrm{S}$$

$$= 0.146285\ \ \mathrm{ms}$$

よってフルカウントでの時間インターバル

$$T_f = 0.146285\ \ \mathrm{ms} \times\ \ 2^8\ \ \fallingdotseq\ \ 37.4\ \ \ \mathrm{ms}$$

例

12msec を使用する場合

WDT.WRITE.TCSR = 0x5aac

という設定をする.

この結果を計算により，算出してみる

5a はアドレスの指定のためここでは，下位 2 ビットの ac を使用する.

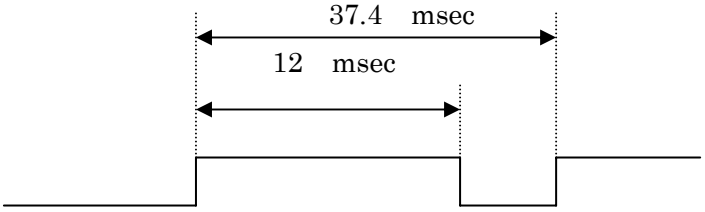
まず，ac を 2 進数に変換する.

ac＝1010 1100　（H）

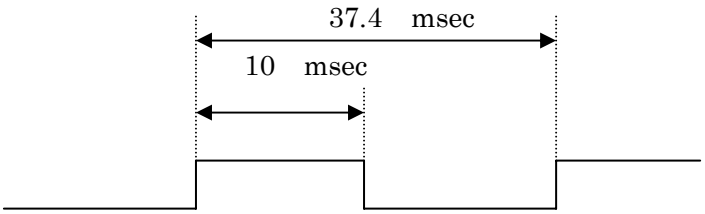
となる．これを，10 進数に変化する.

1010 1100　＝　174

$$37.4 \text{ msec} \times \left(\frac{256-174}{256}\right) = 12 \text{ msec}$$



周期を 10msec にするためには



$$37.4 \text{ msec} \times \left(\frac{256-X}{256}\right) = 10\text{msec}$$

X　＝　188

この値を，二進数に変換する。1011 1100　となる.

この二進数を１６進数に変換すると，bc

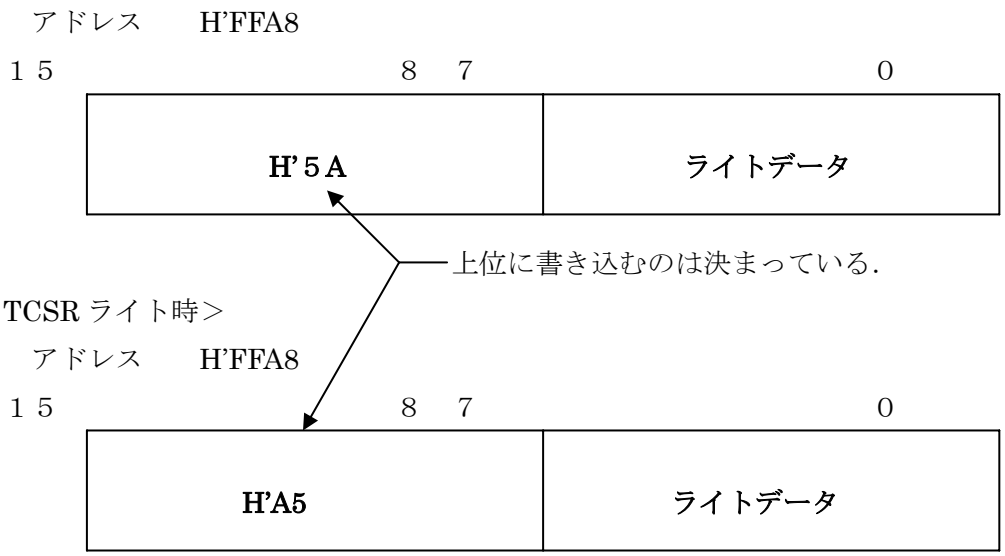
WDT.WRITE.TCSR = 0x5abc

とすれば，周期　10ms となる.

3) **TCSR, TCNT への書き込み**

TCSR, TCNT ともに同一アドレスになっているため,TCNT, TCSR へ書き込むときには,下位バイトを書き込みデータに,**上位バイトを H’５A（TCNT のとき）**または **H’A5（TCSR のとき）**にしてワード転送を行う．プログラム例をプログラム１に示す.

＜TCNT ライト時＞



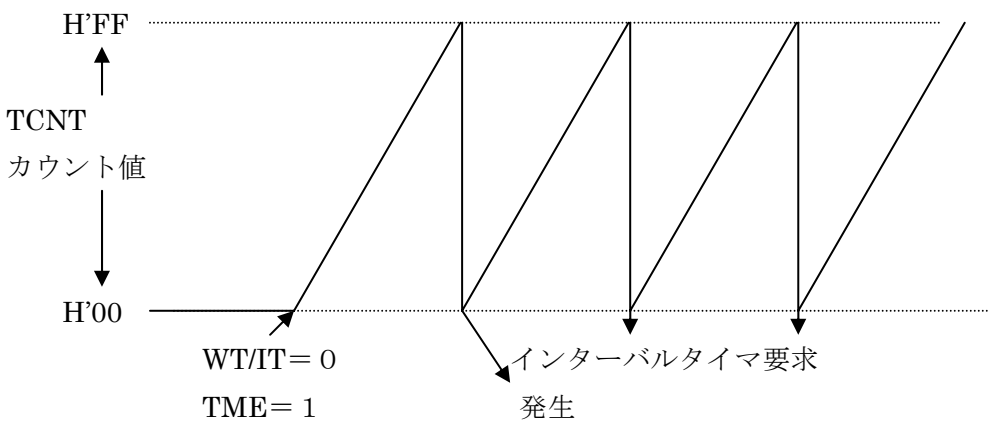
4)**インターバルタイマ時の動作**

インターバルタイマとして使用するには,**TCSR** の **WT/IT** ビットを０にクリアし,**TME** ビットを１にセットする.

TCNT がオーバフローするごとに,インターバルタイマ割り込み要求が発生する．これにより，一定時間ごとにインターバルタイマ割り込みを発生させることができる.

プログラム１

```
Char a;
WDT.WRITE.TCSR=0x5a26; //TCNT=117=0x75 5msec     5a で TCNT に書き込み
A=WDT.READ.TCSR>BYTE;//OVF クリア
WDT.WRITE.TCSR=0xa53d;//a5 で TCSR への書き込み
```



WDTの設定プログラム

```
WDT．WRITE．TCSR＝0xa53e ――――――――①
INC．IPRH．WORD ＝0xf000 ――――――――②
SetSRReg （0） ――――――――③
```

①のプログラムでは，ＴＣＳＲ（タイマコントロール/ステータスレジスタ）の指定をしている．ＴＣＳＲは読み出し/書き込み可能な８ビットのレジスタで，タイマカウンタ（TCNT）に入力するクロック，モードの選択などを行います．

7	6	5	4	3	2	1	0
OVF	WT/IT	TME	――	――	CKS2	CKS1	CKS0
0	0	1	1	1	1	1	0
R/(W)	R/W	R/W	R	R	R/W	R/W	R/W

ビット７：オーバフローフラグ（OVF）

ビット７	説明
OVF	
0	インターバルタイマモードで TCNT のオーバフローなし （初期値） ［クリア条件］ OVF を読み出してから０を書き込む
1	インターバルタイマモードで TCNT のオーバフロー発生

ビット７～０には，①のプログラムの　3e　が対応している。よってここでは０が入力されている．

ビット６：タイマーモードセレクト（WT/IT）

ウォッチドックタイマとして使用するか，インターバルタイマとして使用するかを選択します．この選択によって，TCNT がオーバフラウしたとき，インターバルタイマ割り込み（ITI）が発生するか，WDTOVF 信号が発生するかが決まります．

ビット６	説明
WT/IT	
0	インターバルタイマモード：TCNT がオーバフラウしたとき CPU へインターバルタイマ割り込みを要求 （初期値）
1	ウォッチドックタイマモード：TCNT がオーバフラウしたとき WDTOVF 信号を外部へ出力

今回のプログラムでは，０を選択している．

ビット５：タイマイネーブル（TME）

タイマ動作の開始または停止を設定します．

ビット５	説明
TME	
0	タイマディスエーブル：TCNT を H’００に初期化し，カウントアップを停止（初期値）
1	タイマイネーブル：TCNT はカウントアップを開始。TCNT がオーバフローすると WDTOVF 信号または割り込みを発生

今回のプログラムでは１を選択している.

ビット４，３：予約ビット

読み出すと常に１が読み出されます。書き込む値もつねに１にすること.

ビット２～０：クロックセレクト２～０（CKS２～CKS０）

システム（φ）を分周して得られる８種類の内側クロックから，TCNT に入力するクロックを選択します.

ビット２	ビット１	ビット０	説明	
CKS２	CKS１	CKS０	クロック	オーバーフロウ周期（φ＝28MHz の場合）
0	0	0	φ/2	0.018m s
		1	φ/64	0.585m s
	1	0	φ/128	1.2m s
		1	φ/256	2.3m s
1	0	0	φ/512	4.7m s
		1	φ/1024	9.4m s
	1	0	φ/4096	37.4m s
		1	φ/8192	74.9m s

今回のプログラムでは,１１０を選択

②のプログラムでは優先レベルの指定を行っている。割り込み優先レベル設定レジスタ A～H（ＩＰRA～ＩＰRH）は，それぞれ読み出し/書き込み可能な１６ビットのレジスタで構成されている．今回のプログラムでは，ＩＰRH を使用している．これはWDＴを使用するためである．

レジスタ	ビット			
	１５～１２	１１～８	７～４	３～０
割り込み優先レベル設定レジスタ A	IRQ０	IRQ１	IRQ２	IRQ３
割り込み優先レベル設定レジスタ B	IRQ4	IRQ5	IRQ6	IRQ7
割り込み優先レベル設定レジスタ C	DMAC０	DMAC1	DMAC2	DMAC3
割り込み優先レベル設定レジスタ D	MTU0	MTU0	MTU1	MTU1
割り込み優先レベル設定レジスタ E	MTU2	MTU2	MTU3	MTU3
割り込み優先レベル設定レジスタ F	MTU4	MTU4	SCI0	SCI1
割り込み優先レベル設定レジスタ G	A/D	ＤＴＣ	ＣＭＴ０	ＣＭＴ１
割り込み優先レベル設定レジスタ H	WDT，BSC	I/O	予約	予約

③のプログラムは割り込みマスククリアをしている

また，PWM の切り替えず Fig．4-6 に示す．

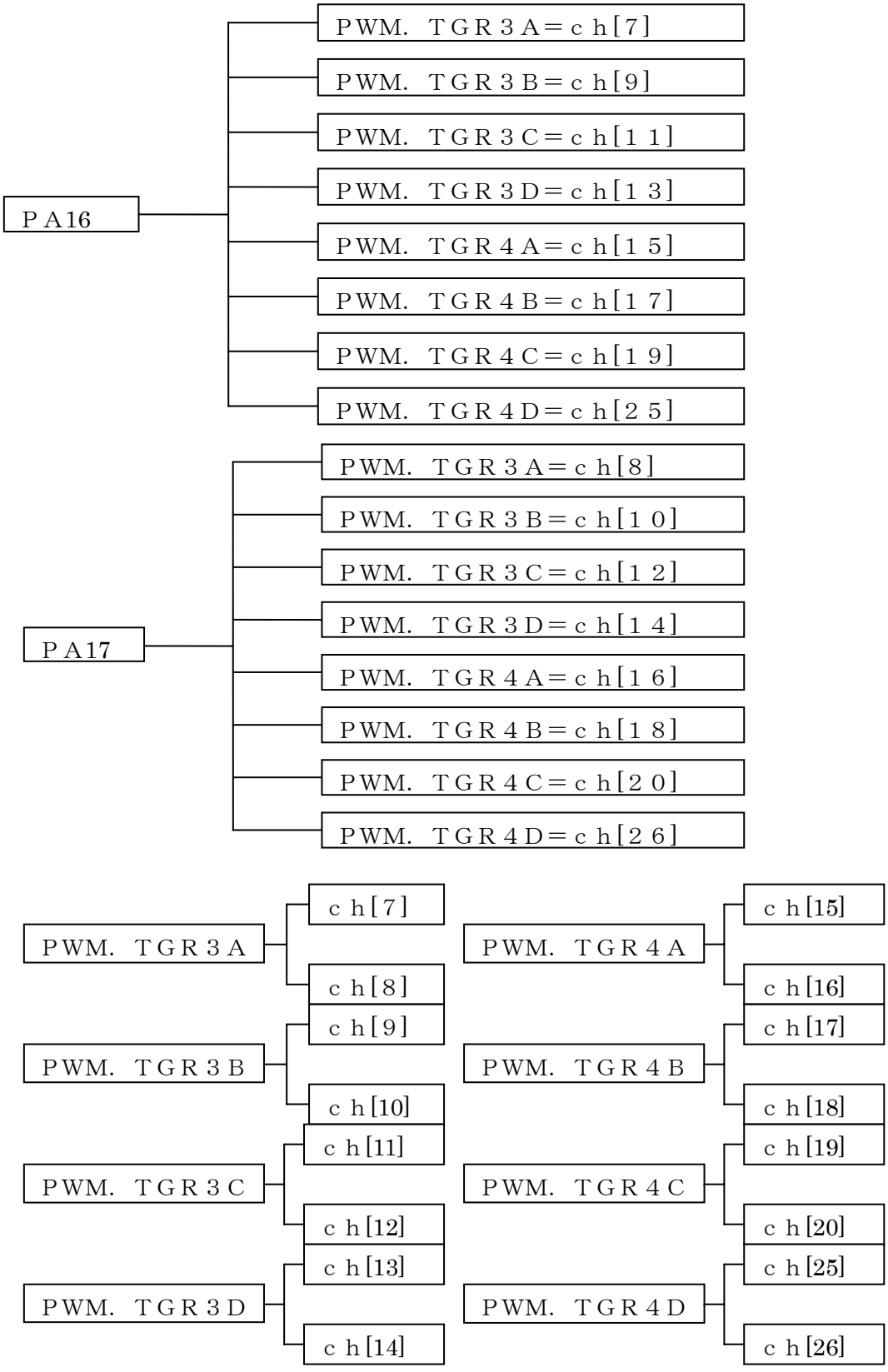
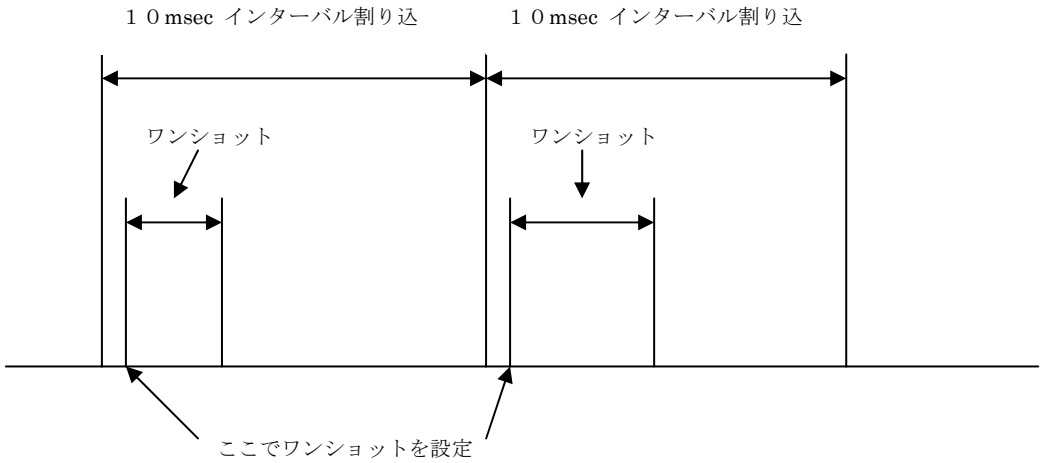


Fig 4- 6　PWM の切り替え図

４－8　　１６軸の PWM を出力する

うまく組み合わせても出力可能な最大の PWM 出力数は１２となる．そこで図１に示すように,WDT のインターバルタイマを使用して周期を決定する．



割り込みと同時にワンショットタイマを起動し,デューティーを決定する．こうすることによって１６ｃｈすべてのポートから PWM を出力することができる．つまり,普通は一つの信号で一つのモータを制御するところを,一つの信号をプログラムによって分割し一つの信号で複数のモータを制御しようというものである．

4-9　MTU について

MTU0．TCR．BYTE　＝　０ｘ０２

TCR：タイマコントロールレジスタ　（８ビットレジスタ）

タイマコントロールレジスタ（TCR）は各チャンネルの TCNT カウンタを制御するレジスタです．MTU には，チャンネル０～４に各一本，計５本の TCR レジスタがあります．TCR レジスタは，８ビットの読み出し/書き込み可能なレジスタです．パワーオンリセットまたはスタンバイモードで H’00 に初期化されます．マニュアルリセットでは初期化されません．

CCLR2	CCLR1	CCLR0	CKEG1	CKEG0	TPSC2	TPSC2	TPSC2
0	0	0	0	0	0	1	0

CCLR2～0：TCNT のクリアの禁止

CKEG1～0：立ち上がりエッジ

TPSC2～0：内部クロック　φ/16 でカウント

φ/16＝28［MHz］/16≒0．57μsec

MTU0．TMDR．BYTE　＝　０ｘＣ０

TMDR：タイマモードレジスタ，８ビットレジスタ

タイマモードレジスタ（TMDR）は各チャンネルの動作モードの設定を行います．MTU には，各チャンネル１本，計５本のレジスタがあります．TMDR レジスタは，８ビットの読み出し/書き込み可能なレジスタです．パワーオンリセットまたはスタンバイモードで H’C0 に初期化されます．マニュアルリセットでは初期化されません．

—	—	BFB	BFA	MD3	MD2	MD1	MD0
1	1	0	0	0	0	0	0

MTU0．TIOR．WORD　＝　０ｘ５５５５

MTU0．TIOR．BYTE．H＝０ｘ５５

MTU0．TIOR．BYTE．L＝０ｘ５５

TIOR：タイマ　I/O　コントロールレジスタ

タイマ　I/O　コントロールレジスタ（TIOR）は TGR を制御するレジスタです．チャンネル０，３，４に各二本，チャンネル１，２に各一本，計８本の TIOR レジスタがあります．

TIOR レジスタはパワーオンリセットまたはスタンバイモードで H'00 に初期化されます. マニュアルリセットでは初期化されません.

MTU0. TIOR. BYTE. H＝０ｘ５５　に関して

IOB3	IOB2	IOB1	IOB0	IOA3	IOA2	IOA1	IOA0
0	1	0	1	0	1	0	1

IOB3～0：TGROB に関する設定　　初期値１でコンペアマッチで０を出力
IOA3～0：TGROA に関する設定　　初期値１でコンペアマッチで０を出力

MTU0. TIOR. BYTE. L＝０ｘ５５　に関して

IOD3	IOD2	IOD1	IOD0	IOC3	IOC2	IOC1	IOC0
0	1	0	1	0	1	0	1

IOD3～0：TGROD に関する設定　　初期値１でコンペアマッチで０を出力
IOC3～0：TGROC に関する設定　　初期値１でコンペアマッチで０を出力

チャンネル　MTU０，３，４　は　２チャンネルの出力
チャンネル　MTU１，２　は　１チャンネルの出力　　} モード１の場合

MTU0, TCNT　＝　０　について
TNCT：タイマカウンタ　１６ビット

タイマ TNCN カウンタ (TCNT) は１６ビットのカウンタです. 各チャンネルに一本, 計５本の TCNT カウンタがあります. TCNT カウンタはパワーオンリセットまたはスタンバイモードで H'00 に初期化されます. マニュアルリセットでは初期化されません. TCNT カウンタの８ビット単位でのアクセスは禁止です. 常に１６ビット単位でアクセスしなくてはいけません.

MTU. TSTR. BIT. CST0＝１
TSTR：タイマスタートレジスタ　８ビット

CST4	CST３	—	—	CST2	CST1	CST0
0	0			0	0	1

チャンネル０→４本
チャンネル１→２本
チャンネル２→２本
チャンネル３→４本
チャンネル４→４本

} 計１６本をコントロール

第五章　設計の検討

5-1　機械部品

二足歩行ロボットを製作する上でブラケットは必要不可欠なものである．ブラケットとはモータとモータをつなぐ部品で，ロボットはモータと CPU とブラケットでできているといえる．昨年度と違いブラケットはすべて自作することになっている．厚さ１㎜の１×２ｍのアルミ板（材質：A5052P-H32）から切り出して製作した．Fig.5-1 に全体図を示す．Fig.5-1 の①と②の軸間距離に精度がかなり必要となる．これは，後に制御する際に股関節と膝関節間距離と膝関節と足首間距離が同じだと，制御が行いやすくなるためである．ほぼすべての部品に曲げ加工を施すため，曲げ加工に重点を置いた．１つ１つの部品に誤差があると組み上げた際かなりの誤差が生まれてしまうので部品を作成する際には，精度を出すために注意を払った．

5-1-1　曲げ加工

プレス加工においても同じような部品を作る際，金型を作り大量に生産する．本研究に置いてブラケット１は現段階でも１０個必要であるためすべて手で作るには多すぎる．また，ブラケット２，３，４では１つ型を作ると使い回しが効く．したがってそこの部分について治具を作り，曲げ加工を施す．その他の曲げ加工はその部品にあうブロックを用いて曲げることとした．

ブラケット 1，2，3，4 にあいたφ3 の穴は曲げる際に固定するための穴で，治具（その穴のある面の寸法に合わせたブロックと、皿ネジの頭を埋めるための鉄板）を作らなければならない．

ブロックは，まずφ3 穴のある面の寸法に合わせてブロックを作り，φ3 の穴の位置にあわせてブロックに下穴をあけ，M3 のネジを切る． ネジ穴の深さは，皿ネジの長さから鉄板の厚さ 5 ㎜とアルミ板の厚さ１㎜を引いた値で，かつドリルが壊れない深さの範囲内の深さとする．皿ネジの頭を埋めるための鉄板は，厚さ 5 ㎜程の鉄板をブロックの寸法より前後左右に１㎜ずつ大きくして作り，ブロックと同様の位置にφ3 の穴をあけ，皿ネジの頭を埋めるために面取りしておく．

ブロックと鉄板でアルミ板を挟み，皿ネジで固定した後，万力でかんでハンマーで叩き，各面を曲げる．ハンマーで叩く際は鉄製のブロックなどを介して叩き，できるだけ叩く回数を減らすようにする．叩く際に介するブロックは曲げる面より大きな寸法のものにし，曲げる面の根本に力を加えるようにすると良い．この方法ではうまくするとほぼ直角に曲げることができ，多少失敗しても-1,-2 度の誤差ですむ．

注意点

- 曲げ加工を施す部品で，φ３の穴のある面以外の穴あけは曲げ加工を施した後に，φ３の穴のある面を基準面にとってケガき，穴あけをする．曲げる前にすべての穴あけをしてしまった場合曲げ加工時にずれる可能性があるので，必ず曲げた後で加工すること．
- また，図でφ12.5 の穴の中にφ3 の穴がある場合，先にφ3 の穴を開け，曲げ加工を施した後にφ12.5 になるように穴を拡げること．

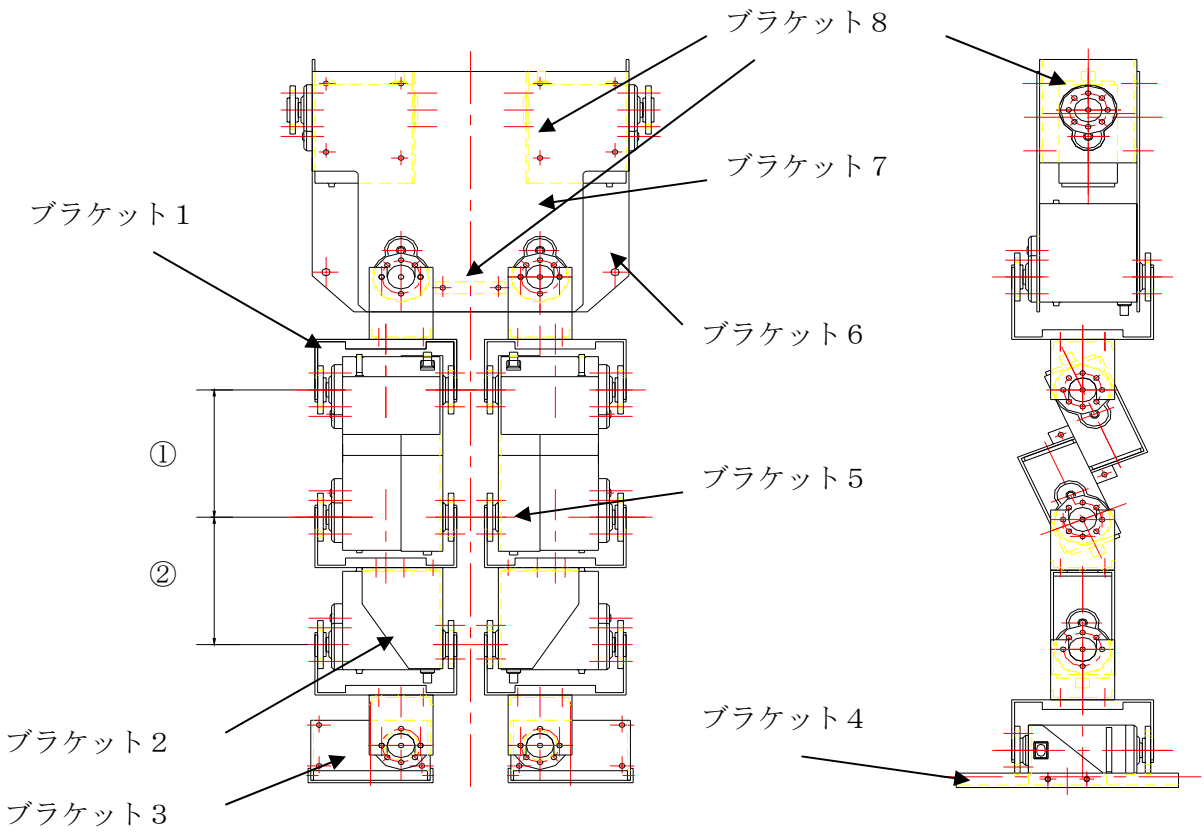


Fig.5-1　二足歩行ロボットの全体図

Fig5-2　ブラケット１概観

5-2　ブラケット１（コの字型ブラケット）

Fig.5-1 のブラケット１は，肩部モータ以外のモータの軸受けとなる部品である．このブラケットで重要なのは軸受けとなる R５の部分とその面と曲げが直角であることである．まず曲げた際に軸受け部分のある面が斜めに曲がったり，歪んでしまった場合，モータの位置がずれ，制御が難しくなる．R5 の部分がずれていた場合，関節が斜めに曲がるようになってしまうため，これも制御が難しくなってしまう．曲げが直角にできなかった場合サーボホーンやフリーホーンに固定した際に歪む可能性がある．

部品図は Fig5-3 のようになる．製作する行程はまず Fig5-4 のように展開した状態まで板を加工する．この際穴の位置（Fig5-5）以外は同じ寸法なので（ただし，RL0，LL0 につけるブラケット１の高さは 25 ㎜ではなく 30 ㎜），フライス盤でまとめて切削するとよい．ただしアクリル板などでアルミ板を挟んで，ネジで固定し切削する事．挟まない場合アルミ板が曲がってしまう可能性がある． 次に 5-1-1 で述べた方法で曲げ加工を施す．皿ネジの頭を埋めるための鉄板は５１×２５㎜にすると良い．製作する際は 5-1-1 の注意点を必ず守ること．

なお，Fig5-4 の図面でピンク色の線は折り曲げるところである．

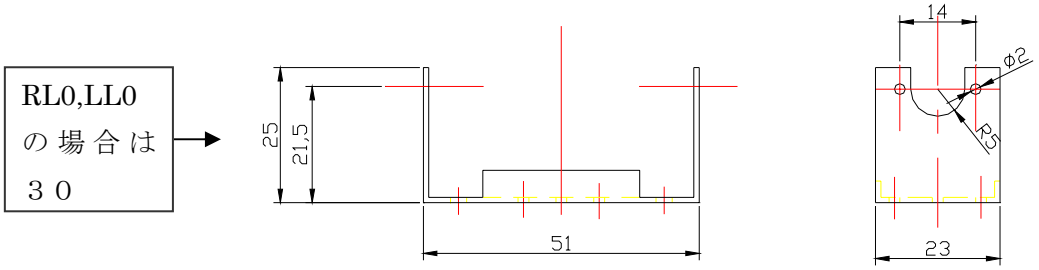


Fig5-3　ブラケット１の図面

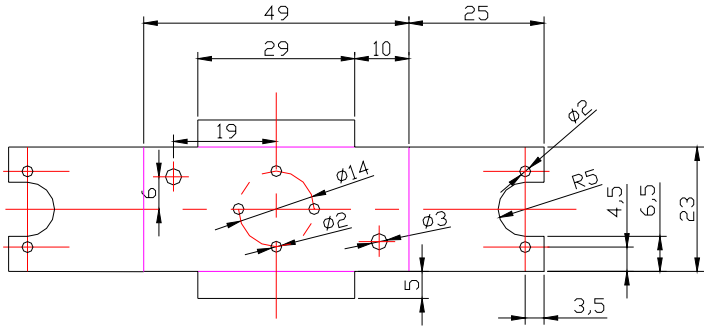


Fig5-4　ブラケット１の展開図

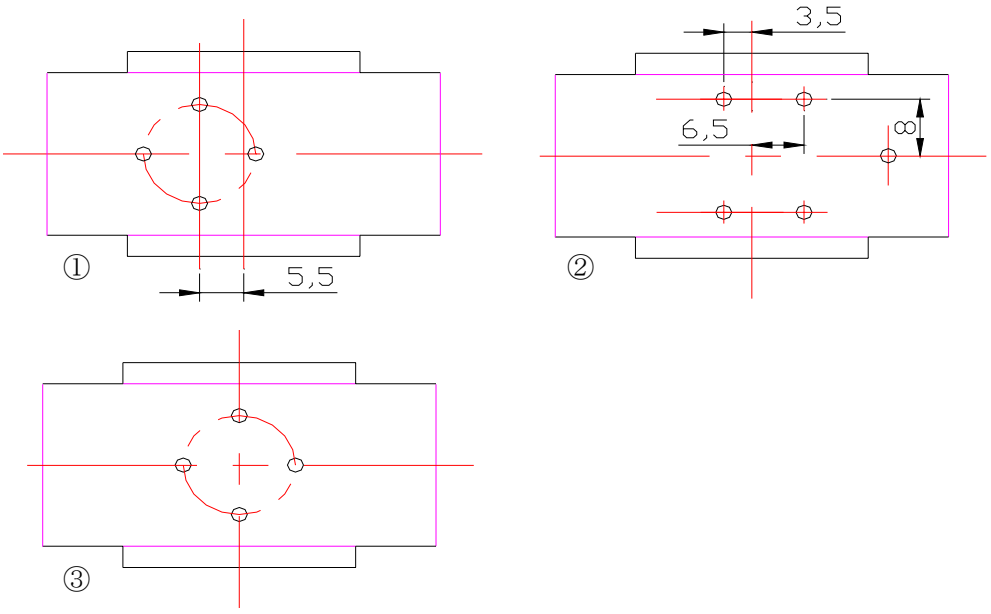


Fig5-5　穴の位置

Fig5-4①は RL1，LL1，RL3，LL3 につけるブラケット１の穴の位置

Fig5-4②は RL2，LL2 につけるブラケット１の穴の位置

Fig5-4③は RL0，LL0，RL4，LL4 につけるブラケット１の穴の位置

Fig5-6　ブラケット 2 概観

5-3　ブラケット２（膝関節下のブラケット）

Fig.5-1 のブラケット２は，膝関節のブラケット１と足首につながるサーボモータをつなげるためのブラケットである．このブラケットで重要なのは穴の位置である．ブラケット１とつなげる穴がずれていると股関節の前後動作部や膝関節と軸が平行にならなくなるので注意しなければならない．またフリーホーンが出るためのφ12.5 の穴がずれているとブラケットにモータがはまらない場合が生じる．

部品図は Fig5-7 のようになる．製作する行程はまず Fig5-8 のように展開した状態まで板を加工する．次に先に述べた方法で曲げ加工を施す．治具は今回使用するサーボモータの突起がない場合を想定した寸法でブロックを作る（４１×２１×３５）．皿ネジの頭を埋めるための鉄板は 43×23 ㎜にすると良い．製作する際は 5-1-1 の注意点を必ず守ること．なお，Fig5-8 の図面でピンク色の線は折り曲げるところである．

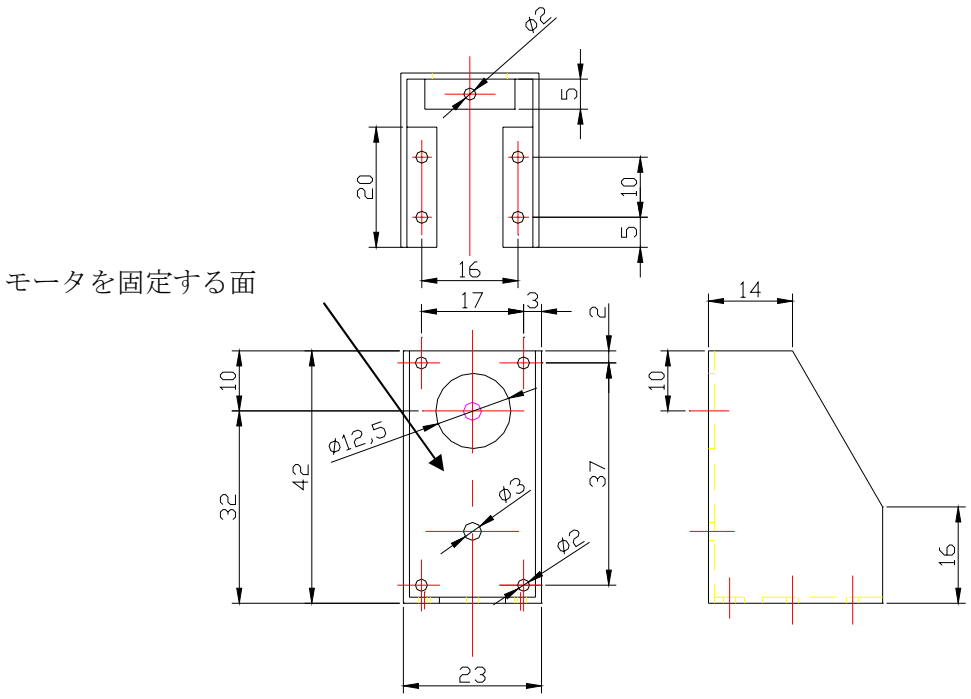


Fig5-7　ブラケット２の図面

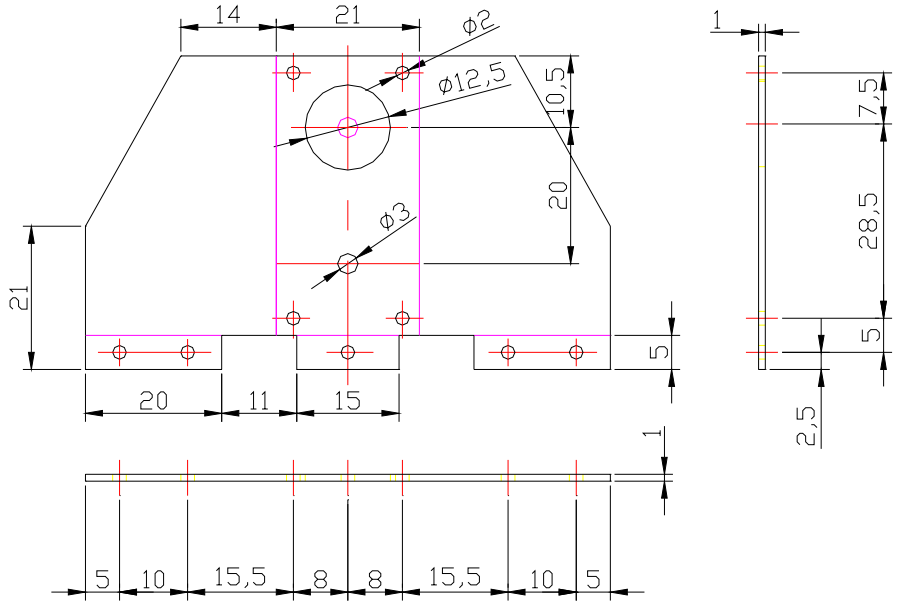


Fig5-8　ブラケット２の展開図

Fig5-9　ブラケット 3 概観

5-4　ブラケット３（足首のブラケット）

Fig.5-1 のブラケット３で重要なのは穴の位置と曲げである．このブラケットは足の裏になる部品と直接接合するので, 穴の位置や曲げの失敗等によってブラケットのモータを固定する面が地面と垂直にならないと制御が難しくなる．

部品図は Fig5-10 のようになる. 製作する行程はまず Fig5-11 のように展開した状態まで板を加工する．次に 5-1-1 で述べた方法で曲げ加工を施す．曲げ加工に使うブロックと鉄板はブラケット２を作る際に用いたブロックを使う．製作する際は 5-1-1 の注意点を必ず守ること．なお，Fig5-11 の図面でピンク色の線は折り曲げるところである．

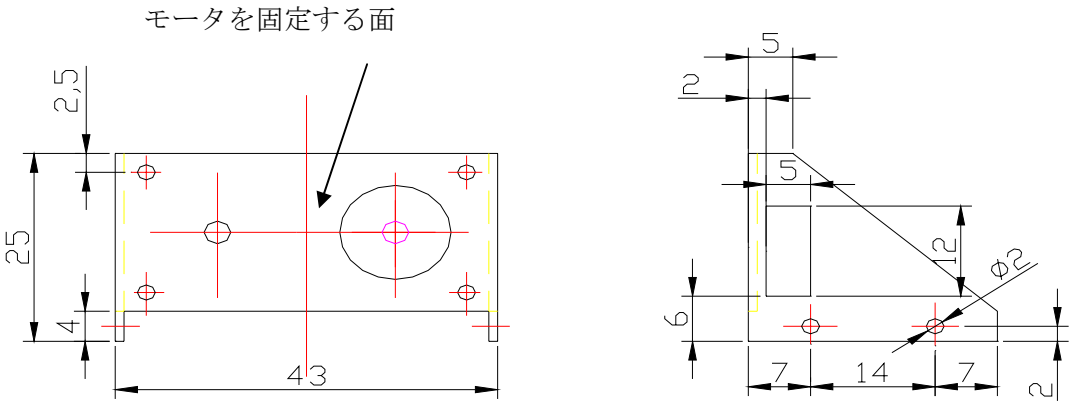


Fig5-10　ブラケット３の部品図

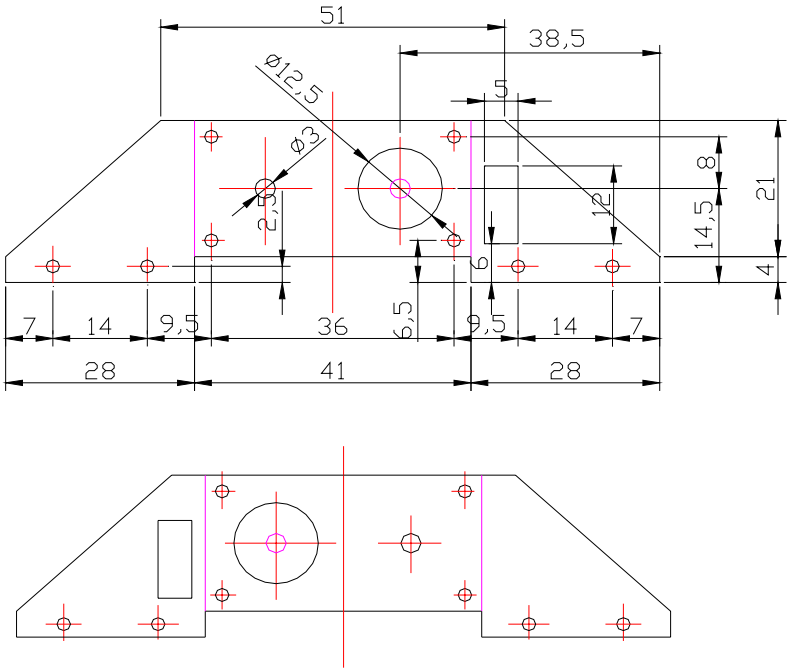


Fig5-11　ブラケット３の展開図

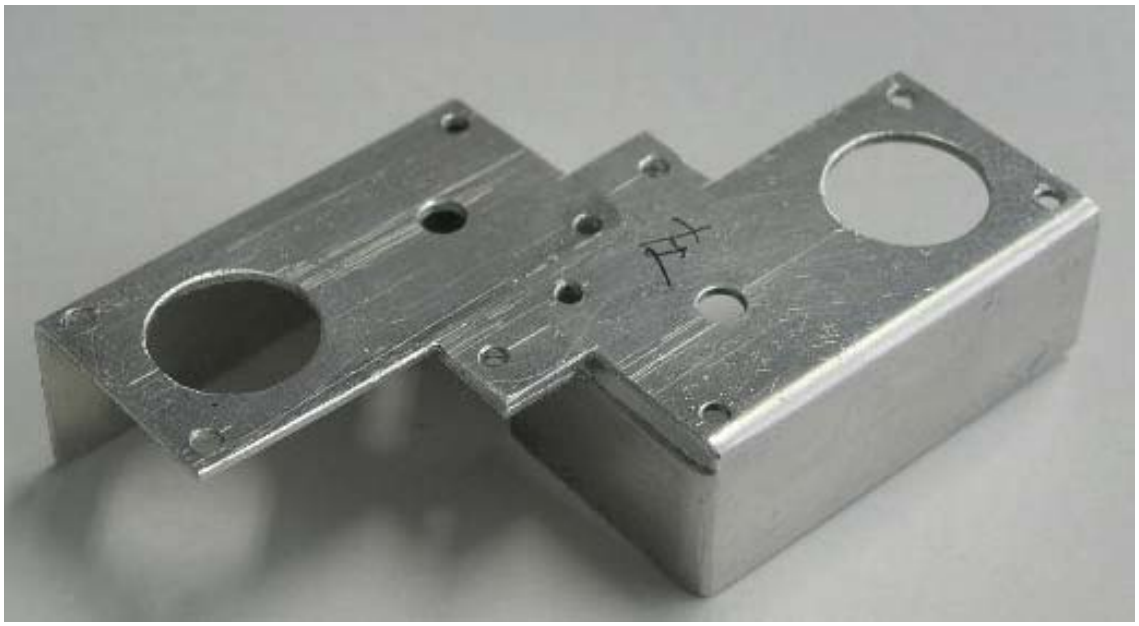


Fig5-12　ブラケット 4 概観

5-5　ブラケット４（太腿となるブラケット）

Fig.5-1 のブラケット 4 は，太腿となるブラケットである。このブラケットで重要なのは穴の位置と曲げ加工である。Fig5-13 で二つのφ12.5 の穴の中心間距離が Fig5-1 の①になる。それゆえ穴の位置を間違えると②と一致しなくなってしまう。また、モータを固定する面の穴の位置を間違えるとモータがはまらなくなる。曲げ加工を多く用いて複雑な形を作るので、曲げ加工で失敗するとモータがはまらなかったり、ブラケットに歪みができてしまう。

部品図は Fig5-13 のようになる. 製作する行程はまず Fig5-14 のように展開した状態まで板を加工する．次に先に述べた方法で曲げ加工を施す．治具はブラケット 2 で製作したブロックと板を使用する。製作する際は 5-1-1 の注意点を必ず守ること．なお，Fig5-14 の図面でピンク色の線は折り曲げるところである.

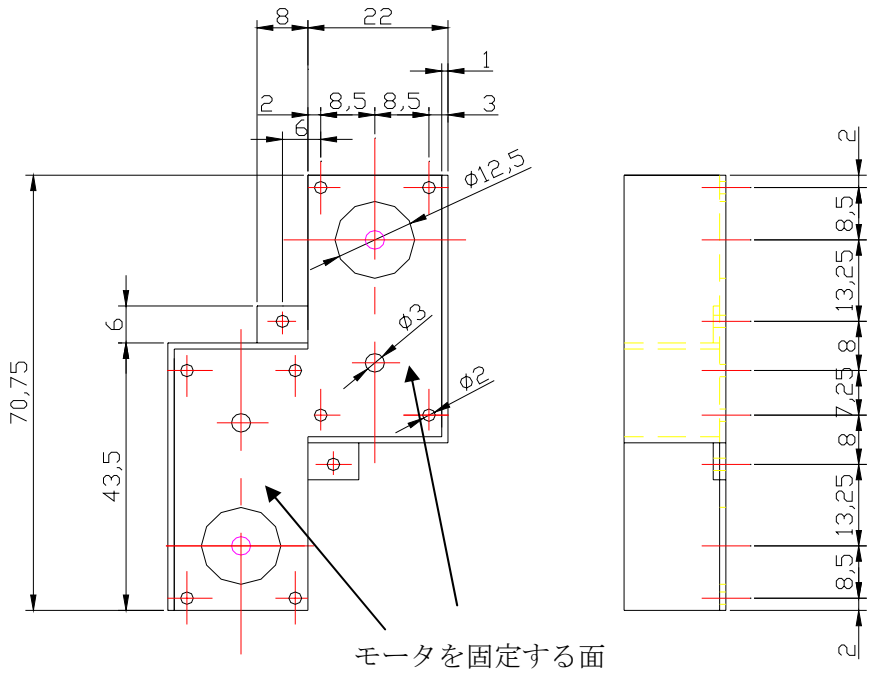


Fig5-13　ブラケット４の部品図

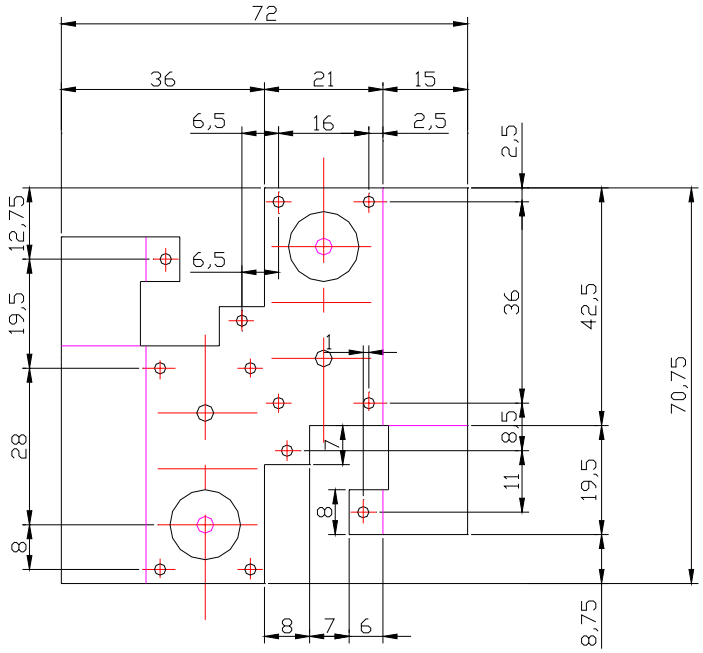


Fig5-14　ブラケット４の展開図

5-6　ブラケット５（足裏となるブラケット）

Fig.5-1 のブラケット 5 で重要なのは穴の位置と曲げである．このブラケットはロボットの土台となるので, 穴の位置や曲げの失敗等によってブラケット 3 のモータを固定する面が地面と垂直にならないと制御が難しくなる. 曲げに関しては曲げる面が長いので斜めに曲がるという可能性は極めて低いが, 曲げるときに使うブロックが小さいと万力で噛んだ際に足裏となる部分が曲がってしまう可能性があるので, 時間の都合上できなかったが, 型を作って製作するのが最善だろう．

部品図は Fig5-15 のようになる. 製作する行程はまず Fig5-16 のように展開した状態まで板を加工する．次に 5-1-1 で述べた方法で曲げ加工を施すが，製作する際は 5-1-1 の注意点を必ず守ること．なお，Fig5-16 の図面でピンク色の線は折り曲げるところである．

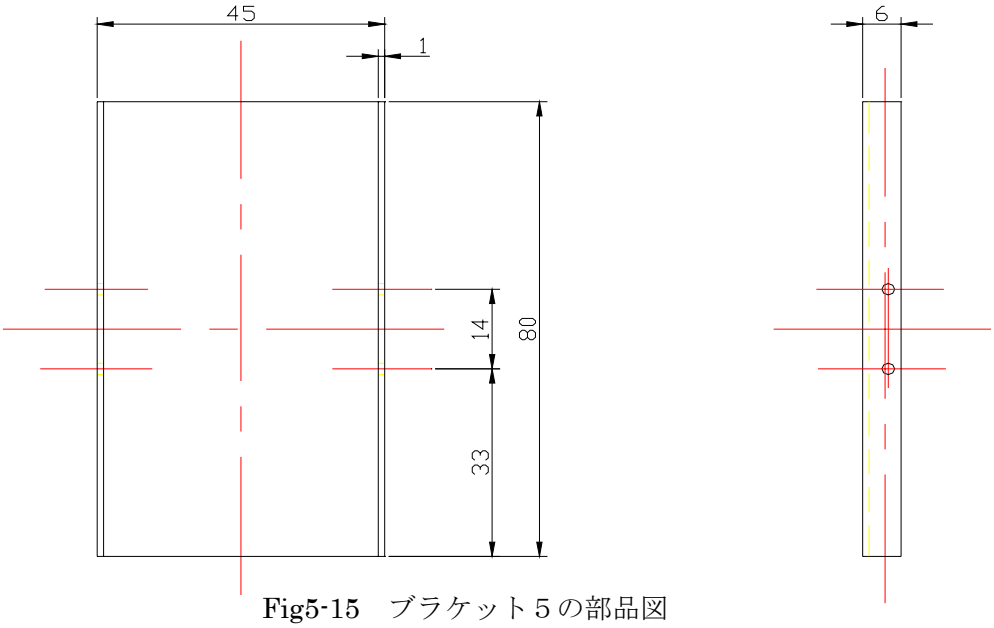


Fig5-15　ブラケット５の部品図

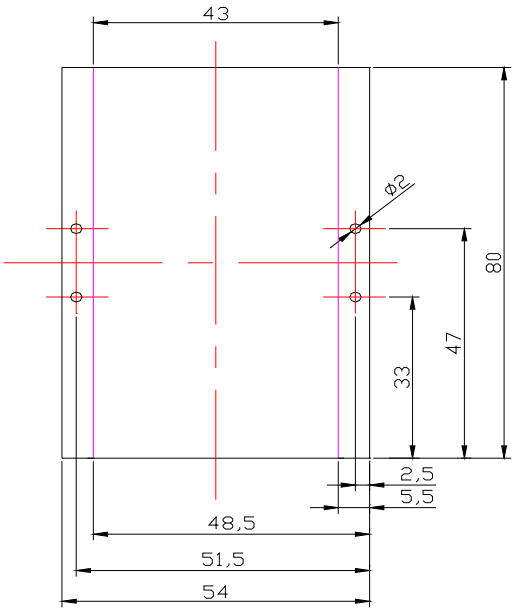


Fig5-16　ブラケット５の展開図

Fig5-17　ブラケット 6 概観

5-7　ブラケット 6 （ボディ背面となるブラケット）

Fig.5-1 のブラケット 6 で重要なのは穴の位置である．このブラケットはロボットのボディ背面となるので，穴の位置を間違えると RL0，LL0 や肩部のモータが斜めに取り付けられたりしてしまう．RL0，LL0 が斜めに取り付けられるとモータの可動範囲が変わってしまい好ましくない．肩部のモータが斜めに取り付けられると今後腕部を取り付けたときに支障がでるだろう．部品図は Fig5-18 のようになる．

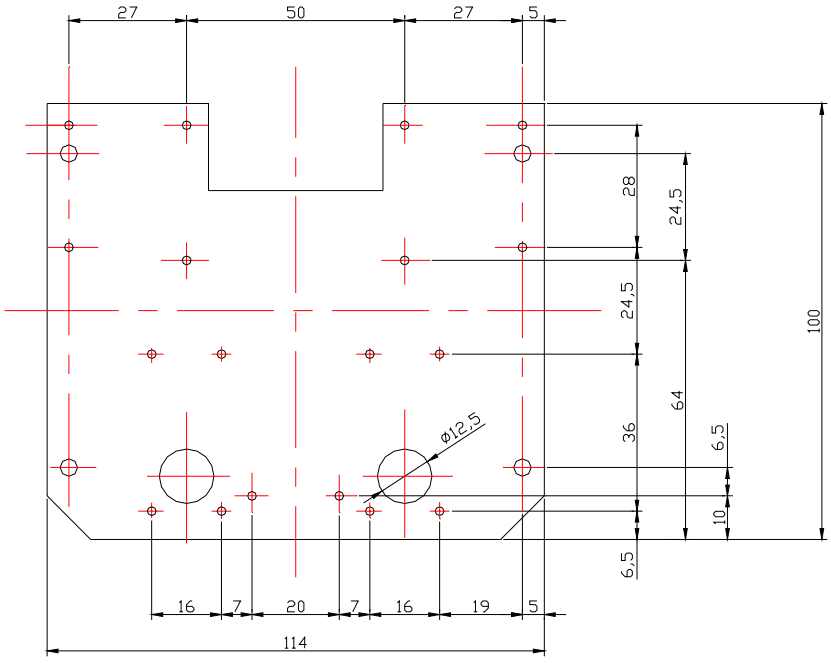


Fig5-18　ブラケット 6 の部品図

Fig5-19　ブラケット 7 の概観

5-8　ブラケット 7（ボディ前面となるブラケット）

Fig.5-1 のブラケット 7 で重要なのはブラケット 6 と同様に穴の位置である．このブラケットはロボットのボディ前面となるので，穴の位置を間違えると RL0，LL0 や肩部のモータが斜めに取り付けられたりしてしまう．RL0，LL0 が斜めに取り付けられるとモータの可動範囲が変わってしまい好ましくない．肩部のモータが斜めに取り付けられると今後腕部を取り付けたときに支障がでるだろう．部品図は Fig5-20 のようになる．

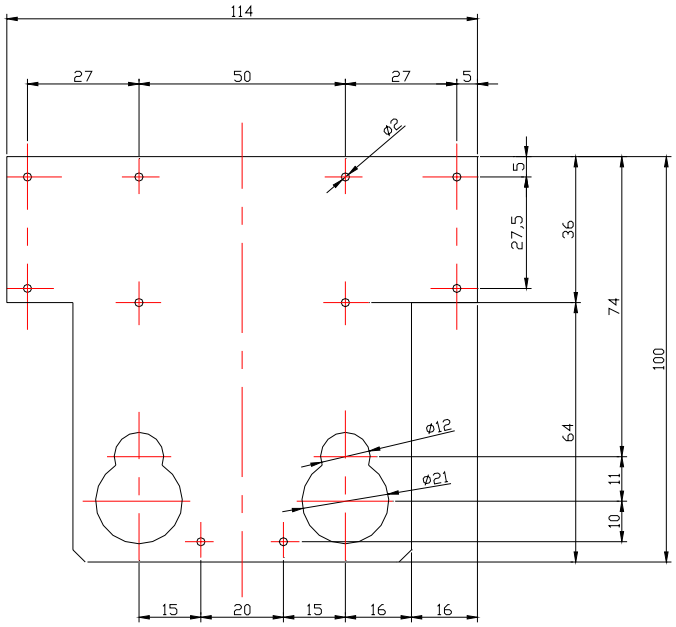


Fig5-20　ブラケット 7 の部品図

5-9　ブラケット 8（ブラケット 7-8 間のブラケット）

Fig.5-1 のブラケット 8 で重要なのは穴の位置と曲げである. このブラケットは Fig5-21 ①, ②は肩部の固定をするブラケットとなるので, 穴の位置や曲げの失敗等によってモータを固定する部分が地面と垂直にならないと肩部が斜めについた状態となり, 腕部を取り付けた際に制御するに当たって好ましくない. 曲げに関しては曲げる面が長いので斜めに曲がるという可能性は極めて低いが, 曲げるときに使うブロックが小さいと万力で噛んだ際にモータを固定する面が曲がってしまう可能性があるので, 時間の都合上できなかったが，型を作って製作するのが最善だろう.

展開図は Fig5-21 のようになる. 製作する行程はまず Fig5-21 のように展開した状態まで板を加工する．次に 5-1-1 で述べた方法で曲げ加工を施すが，製作する際は 5-1-1 の注意点を必ず守ること．なお，Fig5-21 の図面でピンク色の線は折り曲げるところである. ①は肩部の内側に, ②は肩部の外側に配置され, ③は RL0 と LL0 の間に配置されている.

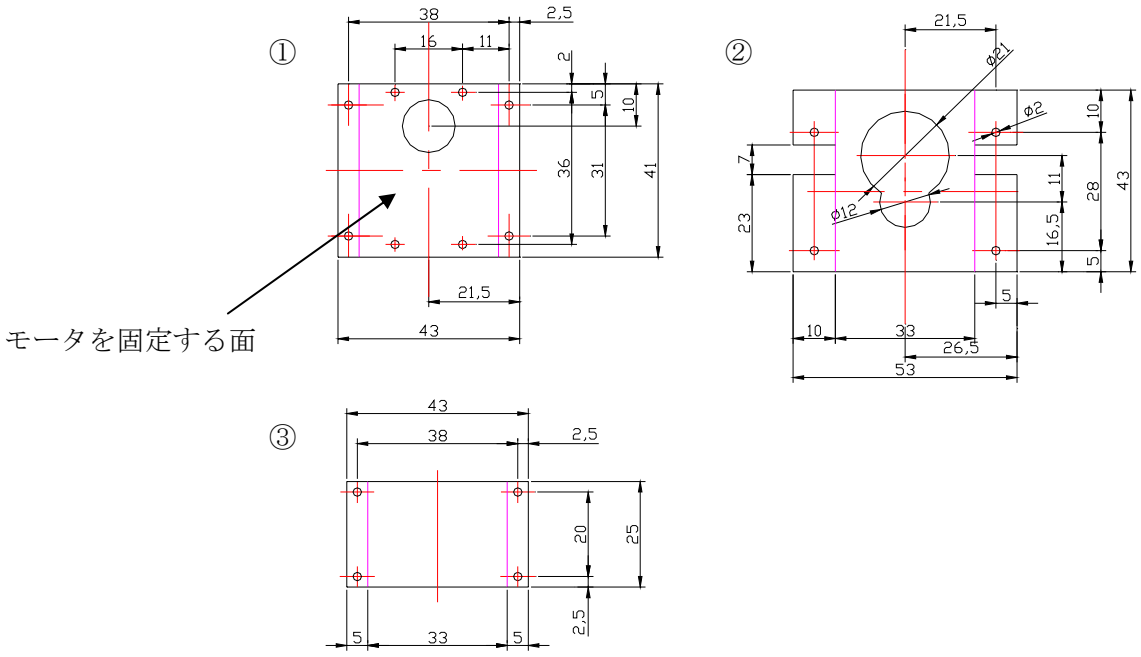


Fig5-21　ブラケット 8 の展開図

ここでは③のみ掲示する

Fig5-22　ブラケット 8 の展開図

第６章　屈伸プログラム

６－１　屈伸の概要

二足歩行ロボットに屈伸をさせるに当たって、下図に示すように股関節・膝関節・足首の両足あわせて６カ所を動かし屈伸をさせる。また、本研究で使用している二足歩行ロボットは設計する際、股関節のサーボモータから膝関節のサーボモータまた膝関節のサーボモータから足首のサーボモータまでの距離を等しくしているため重心移動は考えなくていいようになっている。

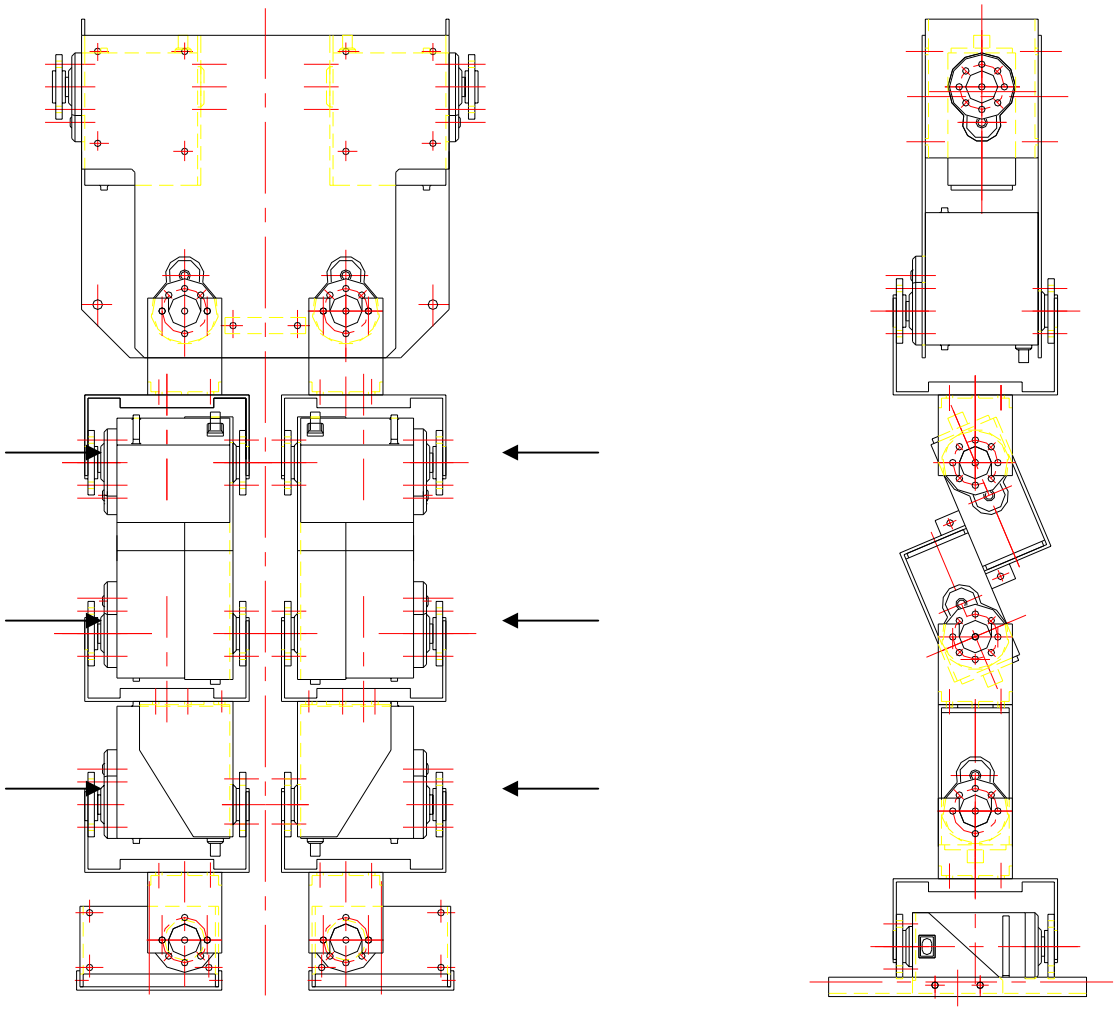


fig6-1 屈伸をさせる際に、使用するモータ

６－２　屈伸

屈伸の一つには、下図のようになる、立っている状態の値としゃがんでいる状態の値を与えてやり、それを分割することによりゆっくり移動させるものである。

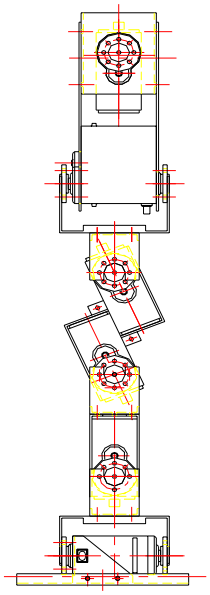


Fig6-2　直立した状態

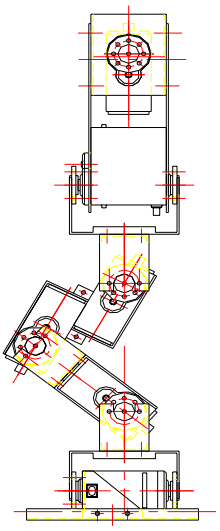


Fig6-3　屈伸した状態

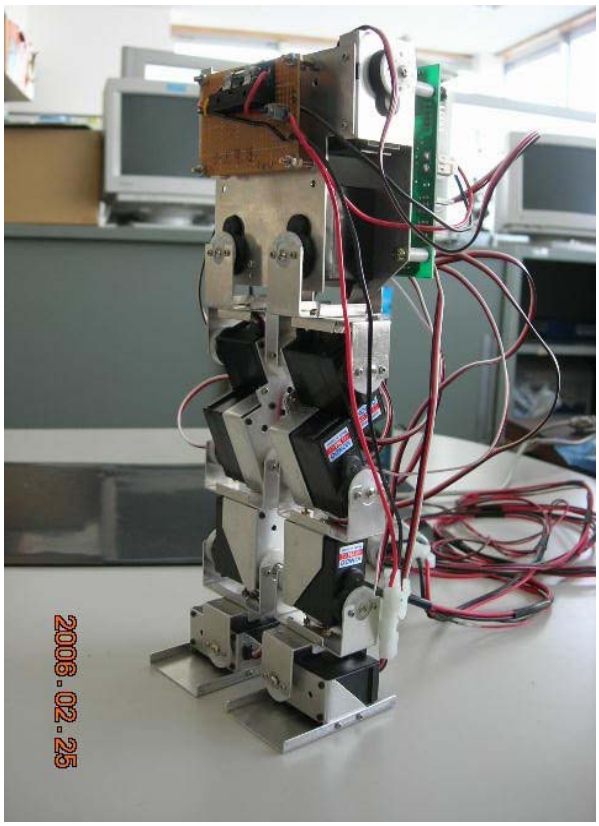


Fig6-3　直立した状態(実写)

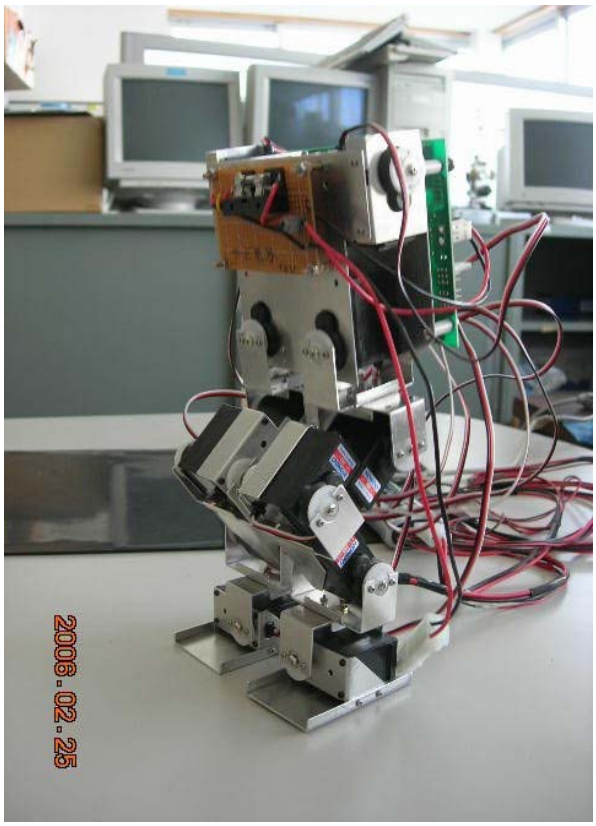


Fig6-5　屈伸した状態(実写)

またプログラムの流れとしてフローチャートを以下に表した。

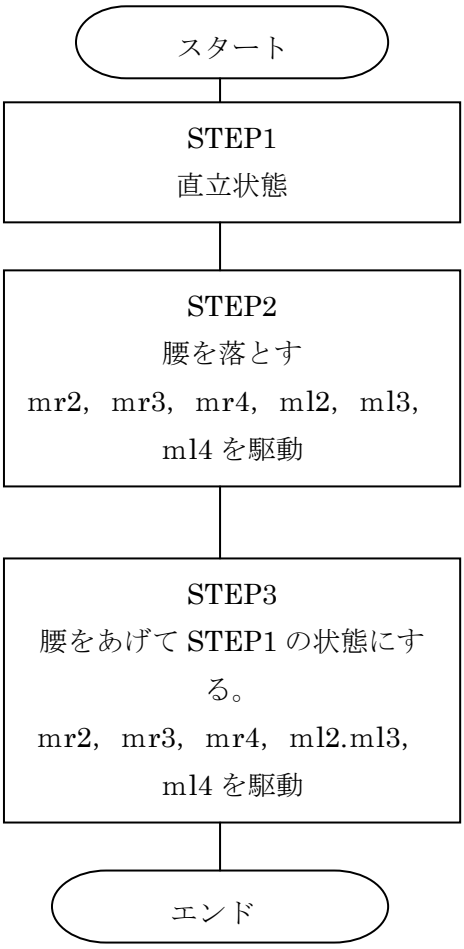


Fig.6-4　屈伸フローチャート

立っている状態としゃがんでいる状態の間を６００分割させ，移動させる．これは本研究で使用したサーボモータは０．１７秒間に６０°という速いスピードで変化する．そのため分割しない場合，または分割数が少ない場合にモータへ負荷がかかるので，それを防ぎスムーズな動きをさせるためにしているのである。

第七章　結言

本研究において，設計段階でサーボホーンを全て外側に向け、ポテンショメータなどの測定器を取り付けられるようにしたり、肩部をつくり腕部の取り付けを可能にしたので発展性は持たせることができたと考えられる。

また、製作した二足歩行ロボットは総重量 1.11kg、全長 302mm となった。昨年度のロボットは総重量 1.59kg、全長約 500mm であるので、0.48kg の軽量化と約 200mm の小型化ができた。よって本研究の目標である軽量化は成功したといえる。

制御面では屈伸動作まで製作することができた。しかし屈伸動作は初歩的な動作でありまだまだ開発していくべきプログラムがある。またその開発していく過程でロボット本体に不具合が発見される可能性もある。

したがって、今後の課題としては腕部の開発や、歩行プログラムの開発、動作を実行させた際に不具合が発生した場合の改善などが挙げられる。

参考文献

「ROBO-ONE のための二足歩行ロボット製作ガイド」
著者　ROBO-ONE 委員会
発行　株式会社オーム社

「ROBOCON　Magajine」
発行人　佐藤政次
発行　株式会社オーム社

「Teku Robo　工作室」
管理人　斉藤力弥
アドレス　http://homepage1.nifty.com/rikiya/

謝辞

本研究を行うに当たり絶えずご指導くださった木澤悟助教授、ならびに機械加工時等様々な場面でご指導くださった実習工場の方々に深く感謝申し上げます。