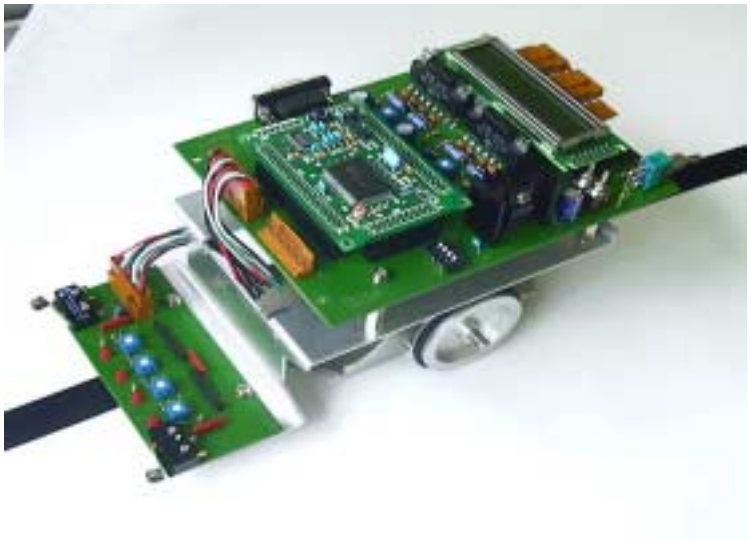
ライントレースロボット

OAKS－LABO

ユーザーズマニュアル　Ver2.1

ＴＭ統合環境　＋　Ｎｃ30ｗａ　Ｖｅｒ４.００　Ｒｅｌｅａｓｅ２対応版



オークス電子株式会社

目　　次

1.はじめに・・・

11.ＴＭ統合化開発環境での使い方 ・・・・・・・・・・・・・・・・・・・・・３４
(1)インストールと注意点 ・・・・・・・・・・・・・・・・・・・・・・３４
(2)エントリー版(新バージョン)の仕様について ・・・・・・・・・・・３４
(3)ＴＭ統合化開発環境の使い方 ・・・・・・・・・・・・・・・・・・・３６
12.プログラムの作成手順 ・・・・・・・・・・・・・・・・・・・・・・・・３７
(1)プログラムの実行手順 ・・・・・・・・・・・・・・・・・・・・・・３７
(2)コンパイル方法 ・・・・・・・・・・・・・・・・・・・・・・・・・３８
(3)コンパイル用バッチファイル例 ・・・・・・・・・・・・・・・・・・４０
(4)I／0 ポートの記述例 ・・・・・・・・・・・・・・・・・・・・・・４０
(5)割り込み処理 ・・・・・・・・・・・・・・・・・・・・・・・・・・４１
(6)割り込みベクタテーブルへの登録 ・・・・・・・・・・・・・・・・・４２
13.Flash ＲＯＭへの書き込み ・・・・・・・・・・・・・・・・・・・・・・４４

付　録

付録 ― 1　ドライバボード回路図 ・・・・・・・・・・・・・・・・・・・４６
付録 ― 2　センサボード回路図 ・・・・・・・・・・・・・・・・・・・４６
付録 ― 3　ドライバボード構成図 ・・・・・・・・・・・・・・・・・・４７
付録 ― 4　センサボード回路図 ・・・・・・・・・・・・・・・・・・・４８
付録 ― 5　ＯＡＫＳ－ＬＡＢＯ外形図 ・・・・・・・・・・・・・・・・・４８
付録 ― 6　コネクター覧 ・・・・・・・・・・・・・・・・・・・・・・４９
付録 ― 7　オプション(ＬＡＢＯ無線ユニット) ・・・・・・・・・・・・５１
付録 ― 8　停止ラインの作り方・・・・・・・ ・・・・・・・・・・・・５２
付録 ― 9　ライントレースロボットのコース例 ・・・・・・・・・・・・５３

安全上の注意

１）万一、動作中及び充電中煙が出ている、変なにおいがする、発熱するなどの異常状態がみられる場合はすぐに電源を切ってください。感電や火傷に注意しながら速やかにコネクタをはずしてください。充電器は別売り専用急速充電器をご使用ください。

２）電池から漏れた液が肌に触れると、火傷の原因になります。破損した電池に触れた場合は、すぐに水で洗い流してください。

３）電池は絶対にショートさせないでください。発熱、液漏れ、破裂、発火の原因になります。

４）液晶モニターが破損した場合、中の液晶には十分注意してください。体に付着したときはきれいな水で十分洗い流し、特に目に入った場合は医師の診断を受けてください。

５）本製品はメカトロニクス学習用です。幼児、子供の手の届かない所に置いてください。

１．はじめに

ロボットブームの今日、アイボ（ホンダ製）に代表される人間型ロボットや工場等での搬送用ロボットや介護用ロボットからホビー用ロボットまで、様々なロボッが次々に紹介されております。この様なロボットブームの２１世紀を迎えた今日、ロボットの仕組みを知りたいと言う願望を持つ一般ユーザ人口が急激に増加しております。

ライントレースロボットは、床に貼った白又は黒いテープをなぞって走るロボットです。最も基本的なロボットですが、ロボットの原理を学ぶのに最も適したロボットと言えます。

初心者でも経験者でもライントレースロボットの製作に取り組む人にとって、その楽しさを経験し、より優秀なロボットの製作が容易に実現できます。

2.OAKS-LABO の特徴

最近ではＰＩＣマイコンシリーズ、Ｈ８マイコンシリーズ、Ｚ８０マイコンシリーズ等をＣＰＵボートに採用したライントレースロボットが多数紹介されています。しかし、これらのＣＰＵボードは、必ずしもユーザ側から見ると使い易いとは言えないのが現状です。理由は次の点にあります。

① パラレルポートやタイマが足りずに増設が必要となる。
② ＣＰＵの周辺にＲＯＭとＲＡＭを必ず接続し、ロムライタを用いてＲＯＭに書き込みが必要となる。
③ デバッガやコンパイラが高価でソフト開発には容易に使用できない。
④ 処理速度が遅く、ロボット制御には向かない。
⑤ ＲＯＭやＲＡＭの容量が少なく、満足なプログラムが開発できない。

これらの問題を全て解決したライントレースロボットがＯＡＫＳ－ＬＡＢＯです。ＣＰＵにはＭ１６Ｃ／６２シリーズのＭ３０６２０ＦＣＡＦＰと呼ばれる１６ビット･ワンチップマイコンを搭載したＣＰＵボート「ＯＡＫＳ１６」が用いられています。

このような特徴以外に、次のような多くの特徴をもっています。

- プログラム作成に必要なコンパイラ、デバッガ、フラッシュライタソフトが付属している。
- ステッピングモータ駆動回路にはＰＷＭ ＩＣを使用し、高速回転を可能にしている。
- センサ基板は取り外しが可能となっているので、拡張コネクタ(Port_3)を併用することにより、マイクロマウス用センサ等が搭載可能となる。
- 拡張用ＲＳ２３２Ｃコネクタ(チャンネル０：Ｔｘ０，Ｒｘ０)にオプションの双方向ＲＳ２３２Ｃ無線ユニット又は単方向無線ユニットを取り付けると、無線による遠隔操作が可能となる。

3.OAKS-LABO の仕様

ＯＡＫＳ－ＬＡＢＯの仕様は、表１で示されるように、ＣＰＵには１６ビットマイコンを搭載したＯＡＫＳ１６マイコンボートがコネクタを介して取り外し可能となっています。

表１　ＯＡＫＳ－ＬＡＢＯの仕様

機　構	内　　容
ＣＰＵ	ＯＡＫＳ１６マイコンボード 三菱電機 30620FCAFP 使用 フラッシュ ROM 128KB、RAM 10KB 内臓
モータ	日本サーボ株式会社 KH39GM2-801　6.4V/0.47
モータドライブ	サンケン LSA7020M　2 個使用
センサ	ローム RPR-220　4 個使用
接触センサ	オムロン SS-01GL2-F　2 個使用
シャーシ	アルミニウム板組み立て一体型 ボールキャスタ 2 個付き
車輪	直径 48 ㎜　ゴムタイヤ付き
ソフトウェア	コンパイラＮＣ３０ＷＡ(V4.00 Release2 エントリー版)付き リモートデバッガ KD30 付き フラッシュライタソフト M16CFlsh 付き
電源	12[V] ／1100[mA]　ニッカド電池
マニュアル	ユーザーズマニュアル
重量・サイズ	７８０g(バッテリ無し)　バッテリ:２４０ｇ 横:１１０㎜　長さ:２０５㎜　高さ:１０７㎜

また、表２のようなオプションが別価格として用意されています。(詳細は付録―７をご覧下さい。)

表２オプション仕様

オプション	構　　成	備考覧
単方向ＲＳ２３２Ｃ ＬＡＢＯ無線ユニット	・送信ユニット：No ９１５２ ・受信ユニット：No ９１５３ ・取り付けケーブル、アンテナ付き	チャンネル０:遠隔操作用
双方向ＲＳ２３２Ｃ ＬＡＢＯ無線ユニット	・送受信ユニット：No９１５４ ・送受信ユニット：No９１５５ ・取り付けケーブル、アンテナ付き	チャンネル０:遠隔操作用 チャンネル１:デバッグ用

４.モータドライブボード

　モータドライブボードの構成図を図１に示します。サンケン社製のＰＷＭ(Pulse Width Modulation)制御によるチョッパ(Chopper)方式の駆動回路 IC であるＳＬＡ７０２０Ｍを使用し、高速回転と高トルクを実現します。電源電圧をモータ定格の数倍にし、モータにはパルス状の電圧を加え、流れる電流の調整はパルスの幅で行うので PWM 制御と呼ばれます。

　ステッピングモータは小型で高速回転が可能な中から定格電圧６．４[Ｖ]、電流０.４７[Ａ]用の日本サーボ(株)製ＫＨ３９ＧＭ２－８０１を使用しています。

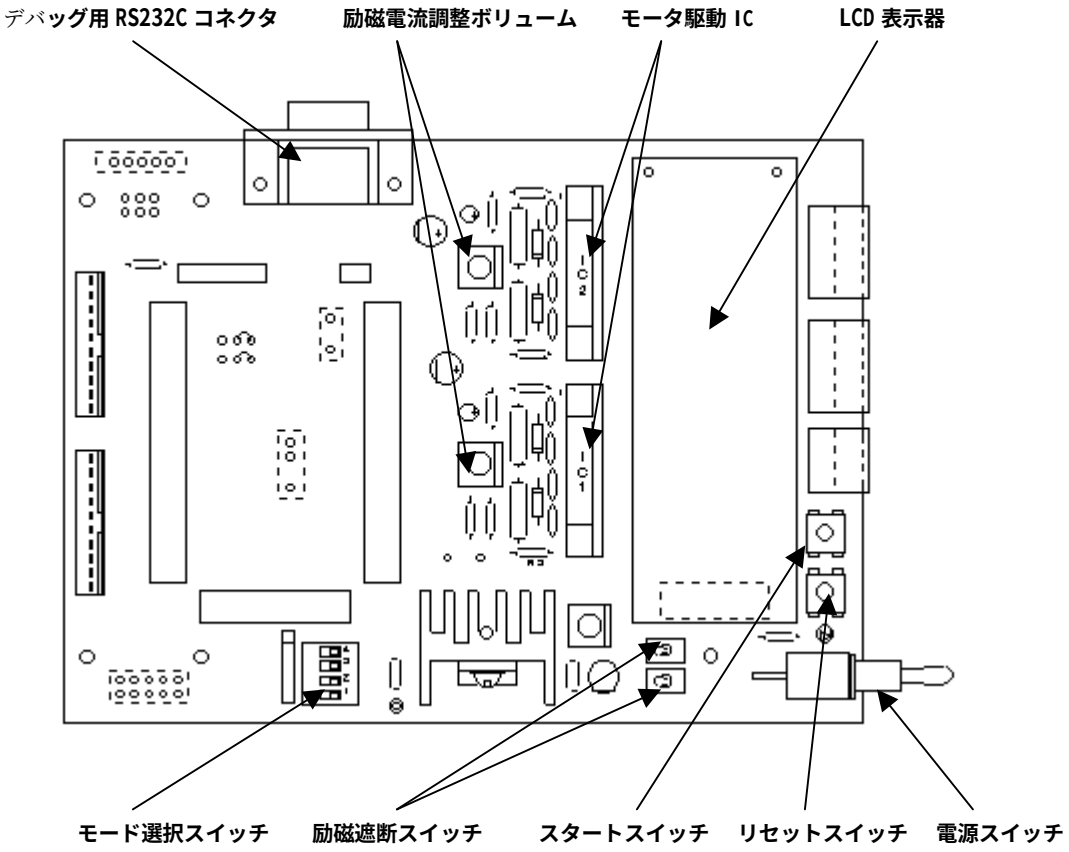


図１　モータドライブボード構成図

(1)、励磁電流調整ボリューム

　左右のモータに流れる励磁電流を調整するのが１００Ωの半固定ボリュームです。この電流は、ドライバＩＣの７番と９番ピンに接続されている１Ωの抵抗電圧を計測すれば電圧値が電流値と１対１に対応する事から判ります。ＯＡＫＳ－ＬＡＢＯでは、この電圧をＡ／Ｄ変換器を介してＬＣＤ表示器でモータ電流として表示します。左右のモータ電流は約５００［mA］程度になるように調整します。詳しくは、調整方法の項目をご覧下さい。

(2)、励磁遮断スイッチ

励磁遮断スイッチは、停止時やプログラムのデバッグ時でのモータを駆動する必要がない場合にバッテリの消耗を防ぐ為のスイッチで、左右のモータ駆動回路に取り付けられています。スイッチをオフにするとモータ電流はゼロになります。

(3)、ＬＣＤ表示器

ＬＣＤ表示器は、２行×１６桁タイプのキャラクターコード付きを使用しています。このＬＣＤ表示器の仕様とキャラクターコード表は付録４をご覧下さい。図２は、ＬＣＤ表示器とＣＰＵポートの接続を示したものです。８ビットのデータは、ポート７のＰ７０～Ｐ７３の４ビットのデータ線を通して下位４ビットと上位４ビットに分けて２回転送する方式です。

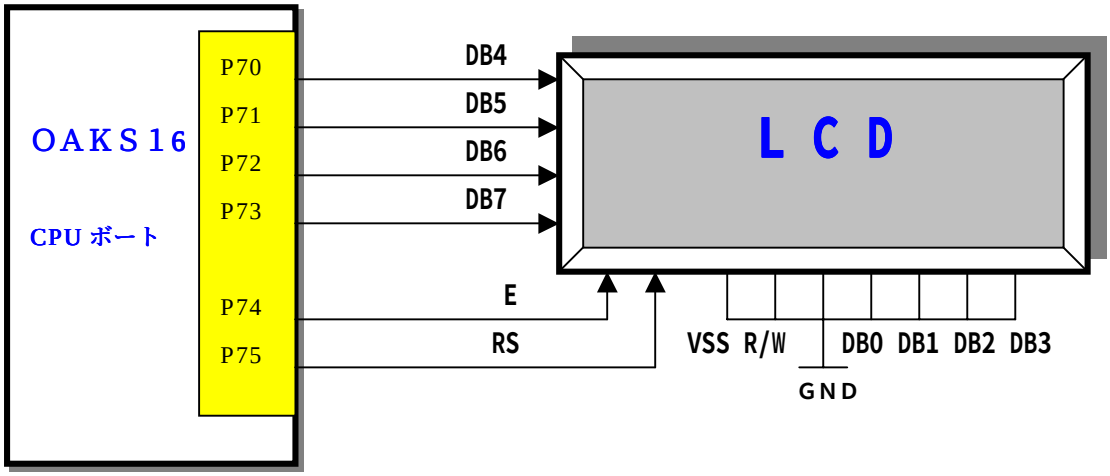


図２　ＬＣＤ表示器とＣＰＵポートの接続

ＬＣＤ表示には次のような情報を表示します。

① 左右モータ電流値の表示
② モード選択スイッチ状態の表示
　速度選択、停止ライン選択、白ライン走行と黒ライン走行選択
③ 走行状態表示
　ライントレース走行情報、右接触センサ情報、左接触センサ情報
④ 走行時間表示

(4)、その他のスイッチ

① 電源スイッチ

電源スイッチをＯＮすると、バッテリから電圧１２[Ｖ]がモータドライブ回路に供給されます。また、３端子レギュレータを介して電圧５[Ｖ]がＣＰＵボートに供給します。

② スタートスイッチ
ＯＡＫＳ－ＬＡＢＯをスタートするスイッチです。

③ リセットスイッチ
このスイッチを押すとＣＰＵボート「ＯＡＫＳ１６」にリセットがかかります。

5.センサボード

センサボードの構成図を図３に示します。センサボードは独立した基板になっており、取り外しが可能です。回路図は付録３でご覧下さい。ロームのフォトレフレフクタＲＰＲ－２２０を４個使用し、反射出力をＯＰアンプによるコンパレータ回路を介してマイコンに入力します。

(1)、感度調整ボリューム

反射光が入力されるとフォトトランジスタに電流が流れ、エミッタに接続された抵抗に電圧が生じます。この電圧と感度調整用ボリュームの電圧をコンパレータに入力し、感度調整を行います。コンパレータの出力側には赤色発光ダイオード(ＬＥＤ)を接続してありますので、センサボード単独で感度調整と動作確認ができます。

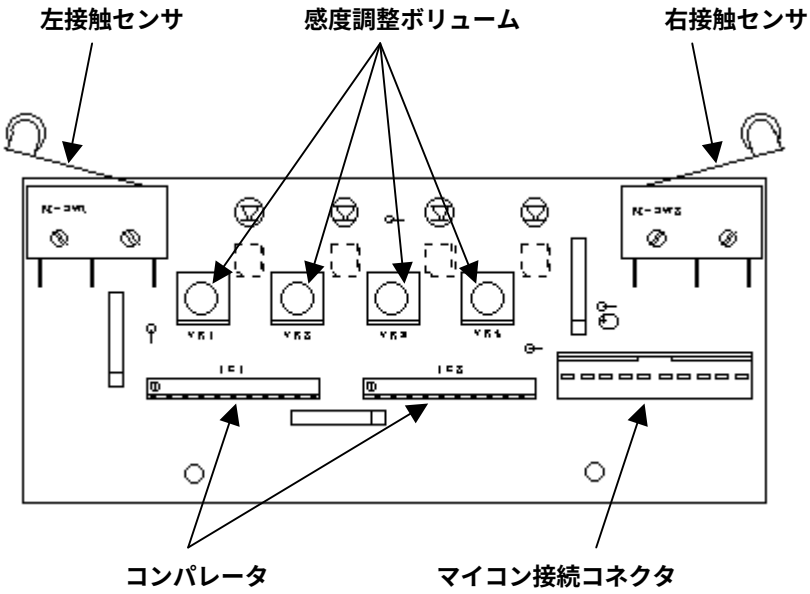


図３ センサボード構成図

(2)、フォトレフレフクタの接続

センサボードに取り付けられている４個のフォトレフレフクタは、図４に示されるように、コンパレータを介してＣＰＵボードのポート４(Ｐ４０～Ｐ４３)に接続しています。

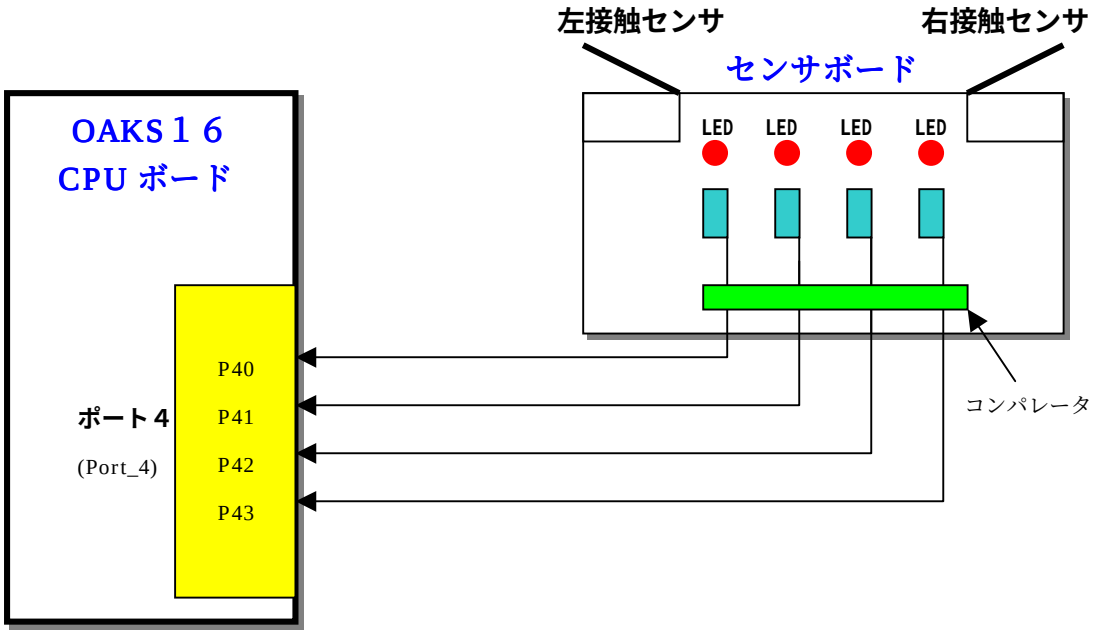


図４　フォトレフレフクタの接続

(3)、接触センサ

接触センサには、オムロンのリミットスイッチ SS-01GL2-F が２個それぞれ基板の両端に取り付けられています。

図５で示されるように、左端の接触センサ出力は、ＣＰＵボードの外部割込みＩＮＴ０に接続されます。また、右端の接触センサ出力は、ＣＰＵボードの外部割込みＩＮＴ１に接続されます。

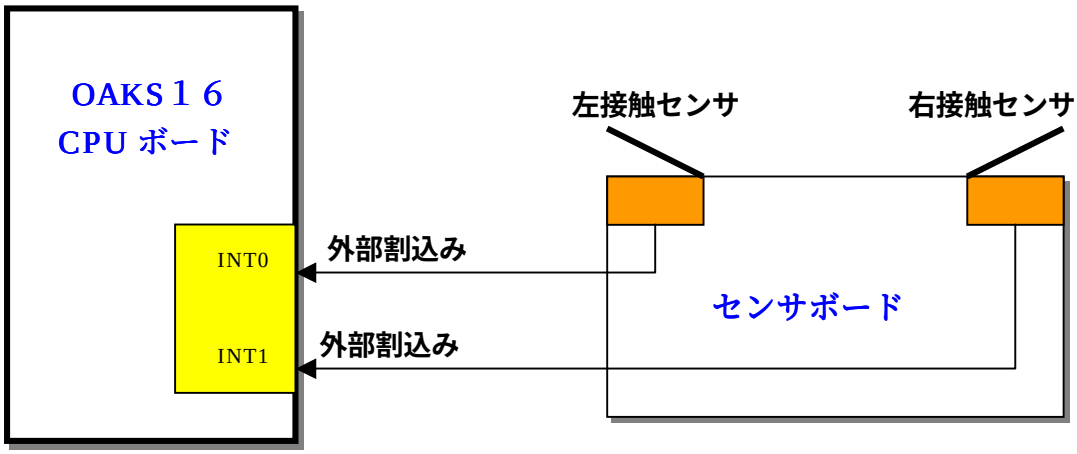


図５　接触センサとＣＰＵボードの接続

6. ＣＰＵボード

図 6 はＭ１６Ｃ／６２シリーズのＭ３０６２０ＦＣＡＦＰを搭載したＯＡＫＳ１６マイコンボードです。このボードには１２８ＫＢのフラッシュＲＯＭと１０ＫＢのＲＡＭがＣＰＵに内蔵されています。

ＲＡＭ領域は、４００ｈ～０２ＢＦＦｈまでの１０ＫＢ空間に割り当てられます。ただし、後部の１.８ＫＢ空間はデバッガのＭｏｎｉｔｏｒ　Ｐｒｏｇｒａｍとして使用されるのでユーザは使用できません。また、フラッシュＲＯＭ領域は、Ｅ０００００ｈ～ＦＦＦＦＦｈまでの１２８ＫＢ空間に割り当てられます。ただし、ユーザ領域はＥ０００００ｈ～ＦＢＥ００ｈ(１１１．５ＫＢ)空間に限定されます。ＦＢＥ００ｈ以降はデバッガが使用するモニタ領域、割り込みの為のベクタエリアとして使用されています。

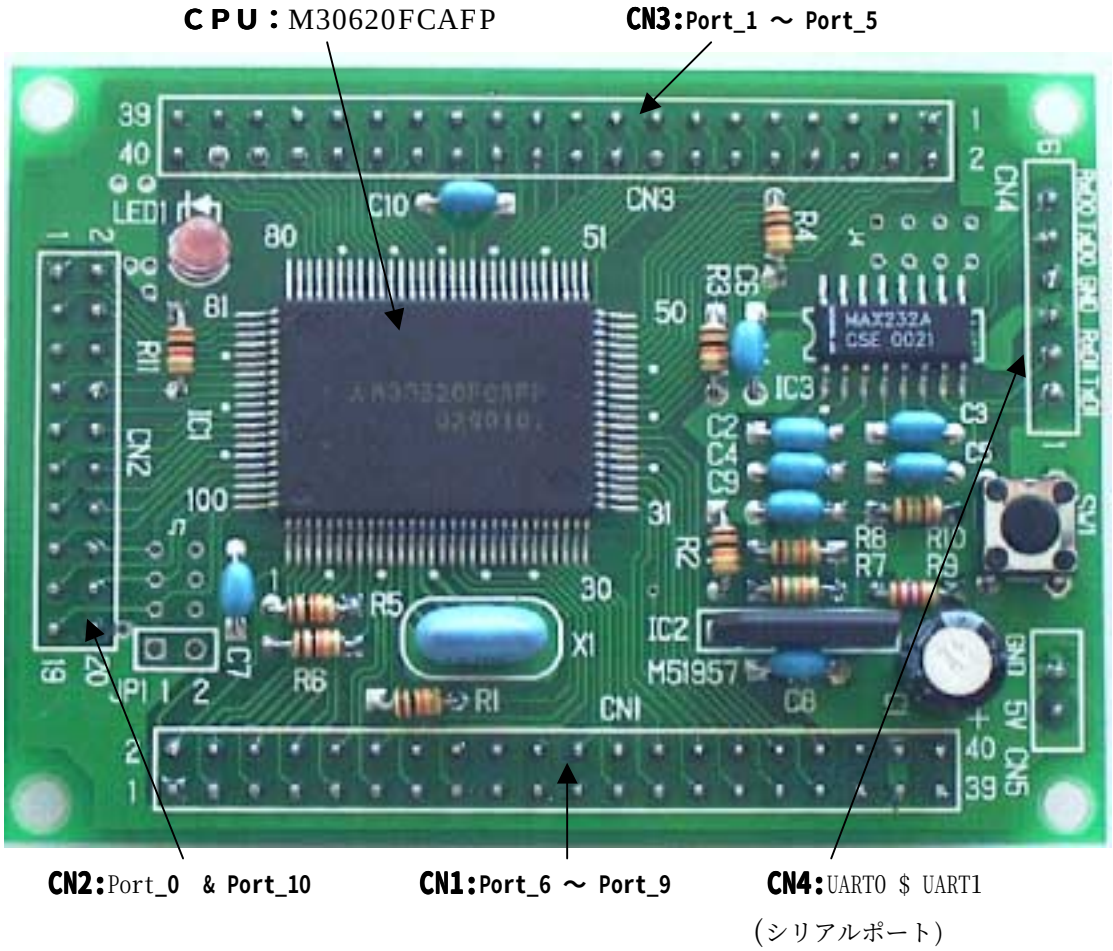


図6 ＯＡＫＳ１６　ＣＰＵボード

(1)、パラレル入出力ポート

パラレル入出力ポートは、最大で１１ポート(８８ｂｉｔ)使用できます。実際は、ポート端

子がシリアルポート、Ａ／Ｄコンバータの入力端子、Ｄ／Ａコンバータの出力端子と併用されている端子は、その使い方で使用できない場合があります。

表３はＯＡＫＳ－ＬＡＢＯが使用するパラレル入出力ポートの割り当てを一覧表で表したものです。

ポート１(Port_1)のビットＰ１０とビットＰ１１は、右側モータの駆動用として使用します。また、ポート２(Port_2)のビットＰ２０とビットＰ２１は、左側モータの駆動用として使用します。

走行腺からのラインセンサ情報は、ポート４(Port_4)のビットＰ４０～ビットＰ４３から入力します。

表３　入出力ポートの割り当て

入出力ポート名		割り当て機能
Port_1	P10 ～ P11	右側モータドライバ
	P12～P17	未使用
Port_2	P20 ～ P21	左側モータドライバ
	P22～P27	未使用
Port_3		拡張用空きポート
Port_4	P40～P43	ラインセンサ
	P44～P45	未使用
	P46～P47	接触センサ
Port_5	P50～P53	モードスイッチ
	P54～P56	未使用
Port_7	P70～P53	LCD 表示器
	P73	未使用

(2)、シリアルポート

シリアルポートは、２チャンネル(ＲＳ２３２Ｃドライバ実装済み)が標準装備されています。チャンネル１（Ｔｘ１，Ｒｘ１）は、リモートデバッガＫＤ３０で使用できます。また、チャンネル０（Ｔｘ０，Ｒｘ０）は、オプションの双方向ＲＳ２３２Ｃ無線ユニットまたは単方向ＲＳ２３２Ｃ無線ユニットを用いた遠隔操作用として使用します。図７は、シリアルポートの接続を示したものです。チャンネル１は９ピンのＤ－ＳＵＢコネクタが標準で実装されています。パソコンのＲＳ２３２Ｃとストレートケーブルで接続し、リモートデバッガＫＤ３０を起動します。また、チャンネル０は、オプション使用で、無線ユニットに専用の接続ケーブルが付属しております。

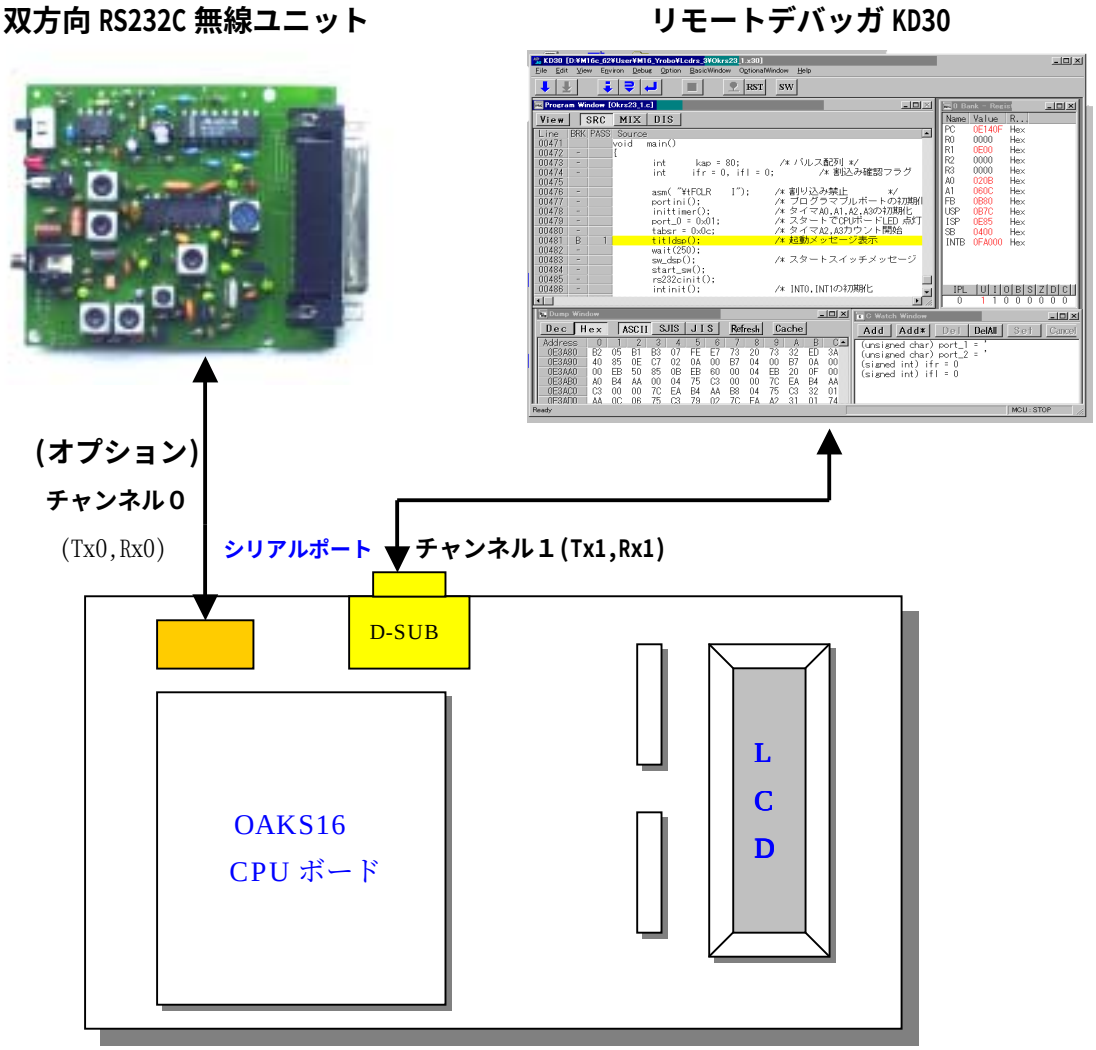


図７　シリアルポートの使用例

7.OAKS-LABO の調整方法

(1)、モータ電流の調整

ＯＡＫＳ－ＬＡＢＯには、バッテリに１２[Ｖ] １１００[mA]ニッカド電池を使用します。モータを２相励磁方式で走行させる場合の電源電流を求めます。

このモータを２相励磁方式で使用すると定格電流は０．４７×２ ＝ ０.９４[Ａ]になります。しかし、チョッパ回路により電流を供給しているので電源からの電流が０.９４[Ａ]流れるわけではありません。電源電圧とモータに流れる電流の関係を図８に示します。電源電圧が高い程、電源から出る平均電流は低くなります。電源電圧１２[Ｖ]を与えると、この電源より流れる励磁電流をモータ定格電流の２分の１(０.９４÷ ２　　０．５[Ａ])に調整します。電流波形は図のように平均値から±４０［mA］程度変動しています。

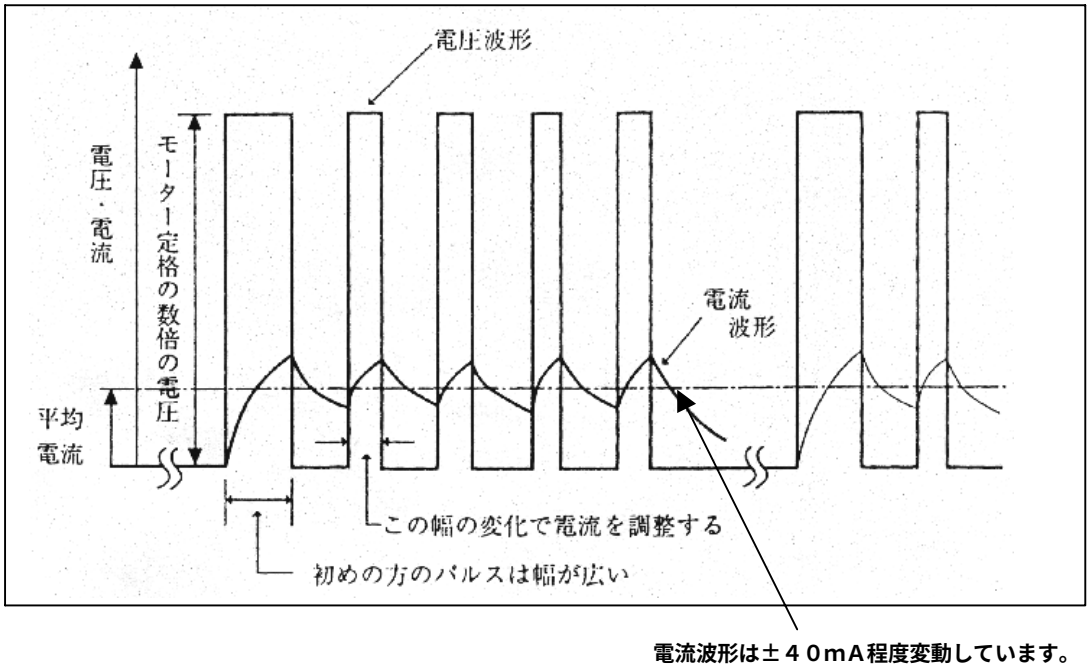


図８　ＰＷＭ波形のモータ電流

■次の手順で調整して下さい。

① 電源スイッチをオンにします。ＬＣＤ表示器には「Ｗｅｌｃｏｍ　ｔｏ　ＯＡＫＳ－ＬＡＢＯ」メッセージが数秒間表示後、図９のような表示画面に切り替わります。

② 左右モータ用の２つの励磁遮断スイッチを共にＯＦＦします。この時、ＬＣＤ表示器の電流値は「Ｌ＝０００mA　Ｒ＝０００mA」のように共に０［mA］となっている事を確認して下さい。

③ 左モータ用励磁遮断スイッチをＯＮにします。図９で示されるように、左モータ

励磁電流設定ボリュームを調整し、左モータ電流が約「Ｌ＝５００mA」となるように調整して下さい。電流値は、常に±４０［mA］程度変動変動しています。ロボット自身を走行させるだけですがらトルクは最大を得る必要しありません。したがって、５００mA程度が表示されればＯＫです。

④ 右モータ用励磁遮断スイッチをＯＮにします。図９で示されるように、右モータ励磁電流設定ボリュームを調整し、右モータ電流が約「Ｒ＝５００mA」となるように調整して下さい。電流値は、常に±４０［mA］程度変動変動しています。変動の中間値として５００［mA］程度が表示されればＯＫです。

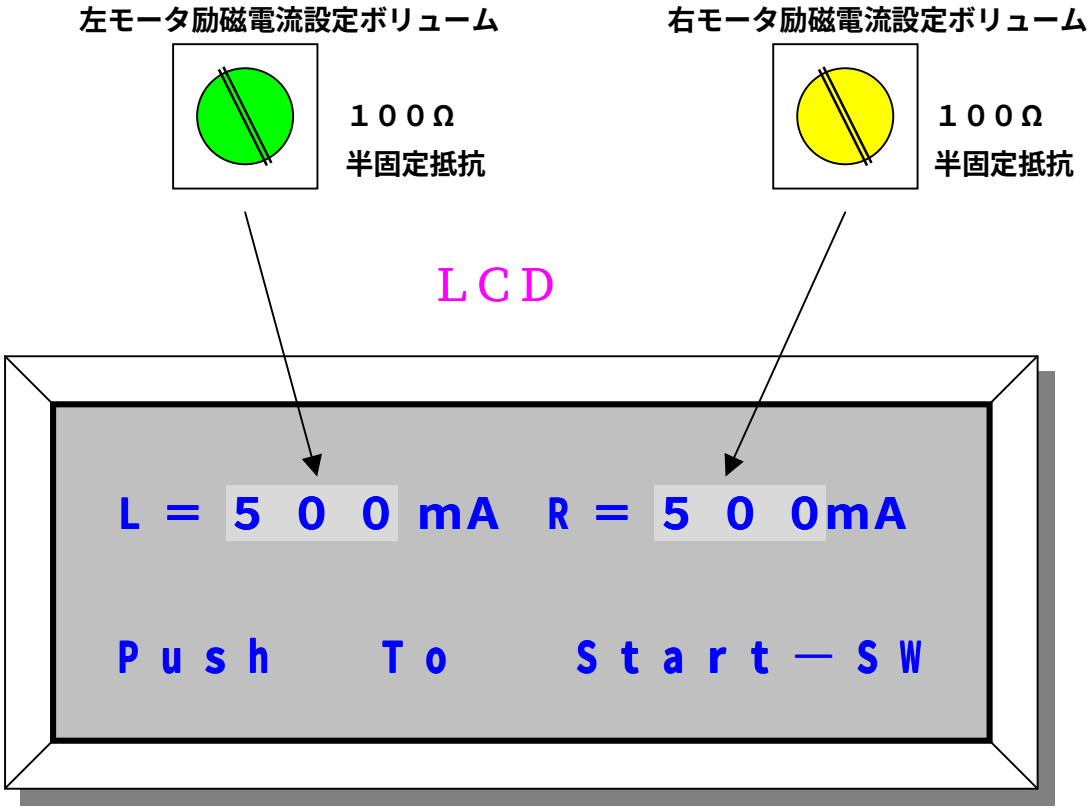


図９　モータ電流の調整

(2)、ラインセンサの調整

ラインセンサボートを車体に取り付けた状態でコースからセンサの先端までの間隔が約５[mm]ら７[mm]間になっているか確認します。また、４つのセンサが全てコースと平行になっている事も確認して下さい。確実に走行するために２つのボールキャスタが少し浮き気味になっています。次の手順で４つのラインセンサを調整して下さい。調整が”あまい”とカーブでのコースアウトの原因となります。確実に調整して下さい。

■　白ラインコースの場合。

① ロボットがコースから外れた位置すなわち黒エリアの所に置き、図１０で示されるＮｏ１～Ｎｏ４の感度調整ボリュームを調整し、センサボード上の各センサに対応するＬＥＤが点灯から消灯する位置に調整して下さい。ロボットをボールキャスタの浮いている範囲で前後に揺らして全てのＬＥＤが点灯しない事を確認して下さい。

② 次に、ロボットのＮｏ１センサを白ライン上に置き、ＬＥＤが点灯する事を確認します。もし、点灯しない場合には、Ｎｏ１感度調整ボリュームを再度調整し、ＬＥＤが消灯から点灯する位置に調整します。

③ 同様にＮｏ２からＮｏ４センサを順番に白ライン上に置き、ＬＥＤが点灯する事を確認します。

④ 白ライン上でもロボットをボールキャスタの浮いている範囲で前後に揺らして全てのＬＥＤが点灯する事を確認して下さい。

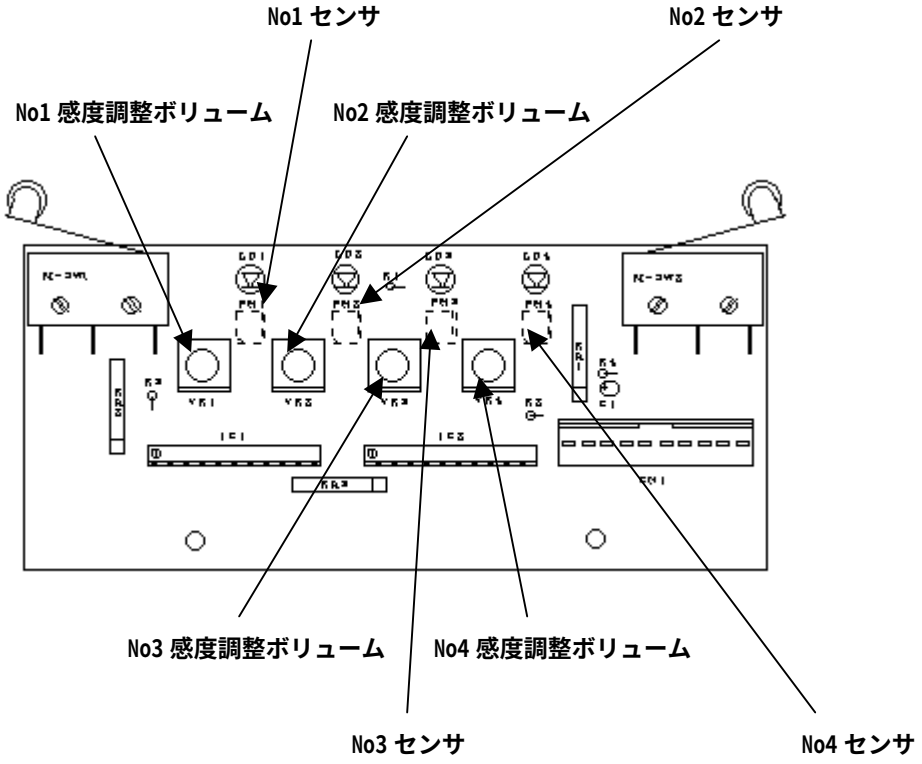


図１０　ラインセンサの調整

■　黒ラインコースの場合。

⑤ ロボットがコースから外れた位置すなわち白エリアの所に置き、図１０で示されるＮｏ１～Ｎｏ４の感度調整ボリュームを調整し、センサボード上の各

センサに対応するＬＥＤが消灯から点灯する位置に調整して下さい。ロボットをボールキャスタの浮いている範囲で前後に揺らして全てのＬＥＤが点灯する事を確認して下さい。

⑥　次に、ロボットのＮｏ１センサを黒ライン上に置き、ＬＥＤが消灯する事を確認します。もし、消灯しない場合には、Ｎｏ１感度調整ボリュームを再度調整し、ＬＥＤが点灯から消灯する位置に調整します。

⑦　同様にＮｏ２からＮｏ４センサを順番に黒ライン上に置き、ＬＥＤが消灯する事を確認します。

⑧　黒ライン上でもロボットをボールキャスタの浮いている範囲で前後に揺らして全てのＬＥＤが消灯する事を確認して下さい。

以上の調整手順を二三回繰り返し、完全にラインセンサを調整します。また、異なったコースで走行させる場合にも、その都度、確実に調整して下さい。

8.OAKS-LABO の走行テスト

ＯＡＫＳ－ＬＡＢＯには、走行テストとデモを兼ねたプログラムが書き込まれています。ライントレースロボットとして走行させる場合には、コースが必要です。床または卓上が白色の場合には、黒のビニールテープを付録５のコース例を参考に貼って下さい。また、床または卓上が黒色の場合には、白のビニールテープを付録８のコース例を参考に貼って下さい。

コースはなるべく段差を少なくして下さい。段差があると車輪の空回りやボールキャスタの引っ掛かりなど、ロボットがうまく動かない場合があります。また、紙でコースを作ると車輪のゴムに紙の繊維が付着してグリップ力が無くなり空回りの原因になります。

走行テストの前にモータ電流とラインセンサの調整をしておきます。また、付属のバッテリパックの充電は、できる限り専用の充電器を使用して下さい。もし、手元に無い場合には、ＤＣ１４Ｖ～１６Ｖ、２００～３００mA程度（定格１１０mA　１５時間）でゆっくり充電して下さい。

(1)、モード選択スイッチの割り当てと機能

モード選択スイッチは、表４のように４つのスイッチの状態によって、速度の設定、オールセンスの判定による停止または通過の設定、白ライン走行または黒ライン走行の設定ができます。

①　スイッチ１番と２番の機能

スイッチ１番と２番の４通りの組み合わせによって、表４のように低速、中低速、中高速、高速の４段階に設定できます。設定変更は、電源を投入する前またはＬＣＤ表示器に表示されるスタートまでの１０秒間以内に行います。

②　スイッチ３番の機能

３番のスイッチは、オールセンス時での動作を選択します。“ＯＮ”で使用する場

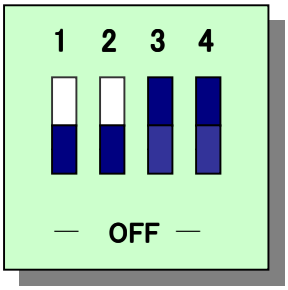
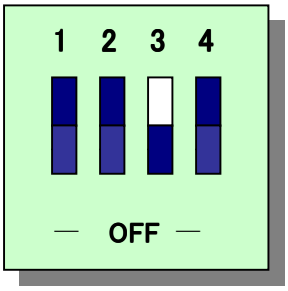
合には、オールセンスの無いコース(交差点の無いコース)で走行して下さい。ストップライン(４つのセンサが同じに消灯(黒ライン走行)または点灯(白ライン走行))で停止します。

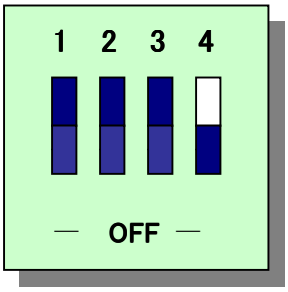
スイッチを“ＯＦＦ”で使用する場合には、交差点などオールセンスのあるコースを走行させることができます。設定変更は、電源を投入する前またはＬＣＤ表示器に表示されるスタートまでの１０秒以内に行います。

③ スイッチ４番の機能

４番スイッチは､“ＯＮ”にすると白地に黒ラインのコースで走行できます。また、ＯＦＦ”で使用すると黒地に白ラインで走行できます。設定変更は、電源を投入する前またはＬＣＤ表示器に表示されるスタートまでの１０秒以内に行います。

表４　モード選択スイッチの割り当てと機能

<table>
<tr><th>モード選択スイッチ</th><th>機　　能</th></tr>
<tr><td> 
</td><td>モード選択スイッチの１番と２番は走行速度を選択します。
<table>
<tr><th>1</th><th>2</th><th>LCD 表示</th><th>走行速度</th></tr>
<tr><td>OFF</td><td>OFF</td><td>Lo</td><td>低速</td></tr>
<tr><td>OFF</td><td>ON</td><td>Ml</td><td>中低速</td></tr>
<tr><td>ON</td><td>OFF</td><td>Mh</td><td>中高速</td></tr>
<tr><td>ON</td><td>ON</td><td>Hi</td><td>高速</td></tr>
</table></td></tr>
<tr><td> 
</td><td>モード選択スイッチの３番は通過と停止を選択します。

ＯＮ・・・減速・停止します。(LCD 表示:　“St”)
オールセンスの無いコースで、ゴール線で停止。
ＯＦＦ・・通過します。(LCD 表示:　“Pa”)
交差点等、オールセンスが在るコース。</td></tr>
<tr><td> 
</td><td>モード選択スイッチの４番は白ラインと黒ラインを選択します。

ＯＮ・・・白地に黒ラインの場合。
(LCD 表示:　“Bk”)
ＯＦＦ・・黒地に白ラインの場合。
(LCD 表示:　“Wh”)</td></tr>
</table>

(2)、走行テスト手順

① 電源スイッチを“ＯＮ”またはリセットスイッチを押すと図１１に示される表示が数秒間表示されます。

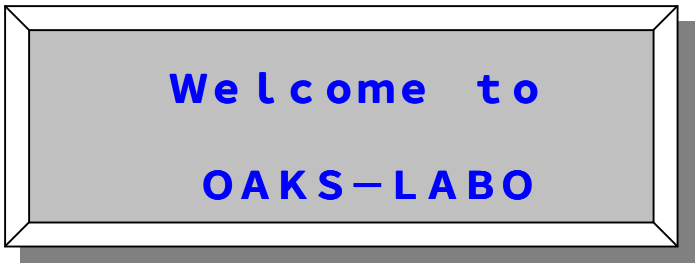


図１１ メッセージの表示

② 数秒後に図１２で示されるモータ電流調整の為の表示に切り替わります。励磁遮断スイッチが共に“ＯＦＦ”にすると「Ｌ＝０００mA　Ｒ＝０００mA」が表示されます。左側の励磁遮断スイッチを“ＯＮ”にしてから左モータ用の励磁電流調整ボリュームを廻して、図１２のように約Ｌ＝５００mAとなるように調整します。電流値は±４０mA程度常に変動しています。故障ではありません。次に右側の励磁遮断スイッチを“ＯＮ”にしてから右モータ用の励磁電流調整ボリュームを廻して、図１２のように約Ｒ＝５００mAとなるように調整します。電流値は同様に±４０mA程度常に変動しています。故障ではありません。調整が終了後、スタートスイッチを押して下さい。この場合、画面が切り替わる迄しばらく押し続けて下さい。

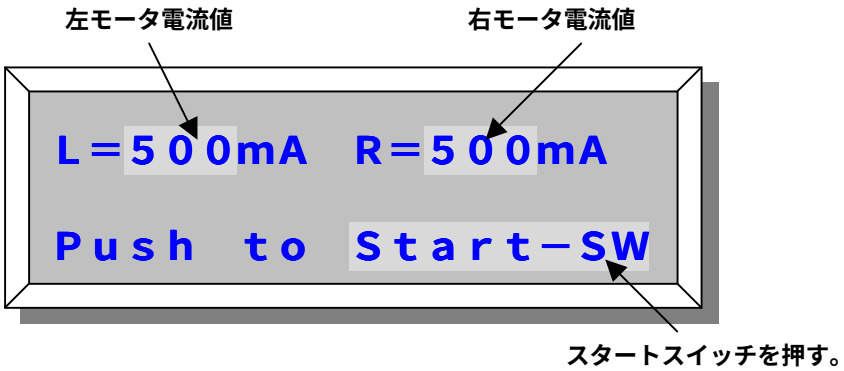


図１２　モータ電流の調整表示

④ スタートスイッチを押すと、図１３で示されるモード表示画面に切り替わります。上段には、走行開始までの残り時間が表示されます。時間は、１０秒から開始し、０秒になると走行を開始します。下段は、表４で表記されるように、モード選択ス

イッチの状態を表示します。変更を行う場合には、走行カウンタが０秒になる迄にモード選択スイッチで行って下さい。

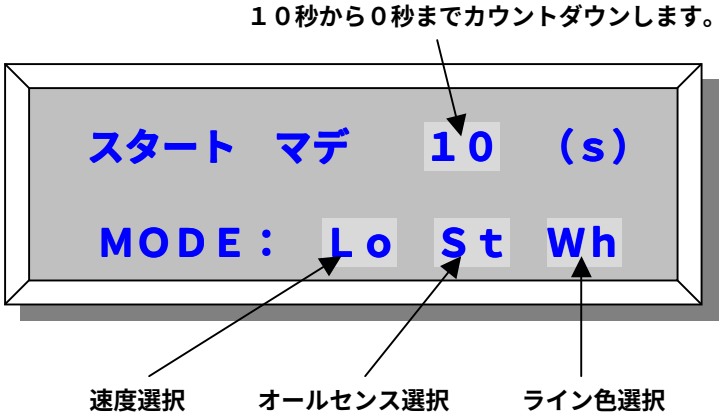


図１３　モード表示画面

⑤　走行カウンタが０秒になると、　図１４で示される走行表示画面に切り替わります。

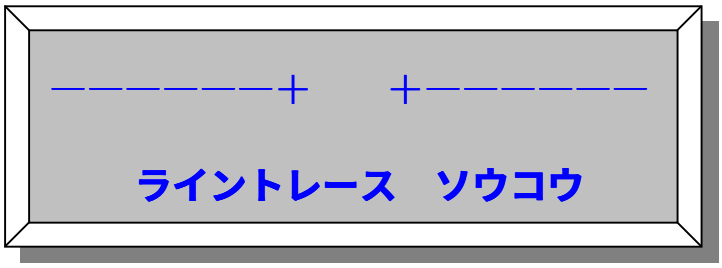


図１４　走行表示画面

⑥　モード選択スイッチで走行開始前に、予め停止ラインの減速・停止の設定がされていると、図１５で示されるように、走行開始から停止迄の所要時間が表示されます。
　再び走行させる場合には、スタートスイッチを押して下さい。③項の画面表示に戻ります。

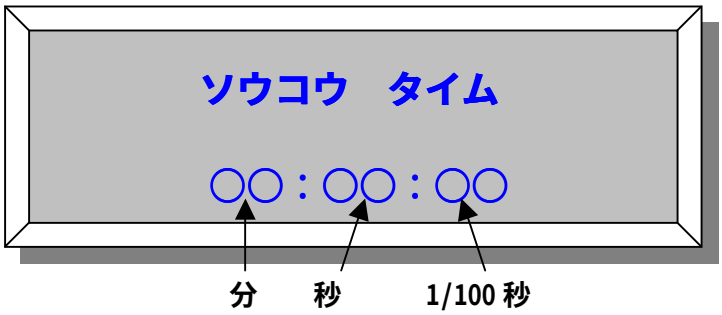


図１５　走行表示画面

⑦　左接触センサが障害物に接触すると、図１６で示されるように、左側が障害物に接触した表示を行い、急停止します。その後、数十センチ程度バック走行を行い完全停止します。
再び走行させる場合には、コースに戻し、スタートスイッチを押して下さい。③項の画面表示に戻り再走行を開始します。

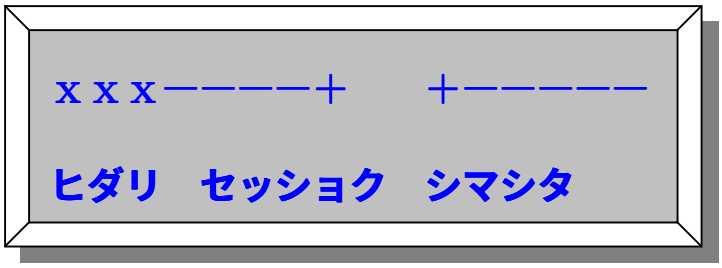


図１６　左側接触の表示画面

⑧　右接触センサが障害物に接触すると、図１７で示されるように、右側が障害物に接触した事を知らせる表示を行い、急停止します。その後、数十センチ程度バック走行を行い完全停止します。
再び走行させる場合には、コースに戻し、スタートスイッチを押して下さい。③項の画面表示に戻り再走行を開始します。

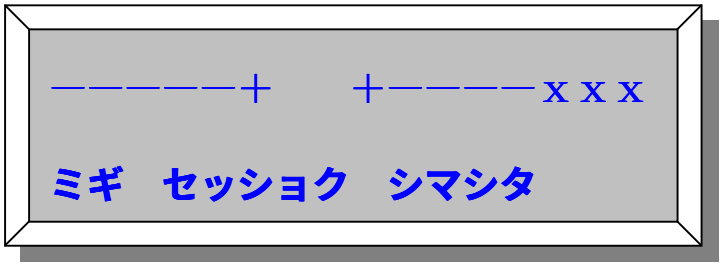


図１７　右側接触の表示画面

9.ＯＡＫＳ－ＬＡＢＯのしくみ

ＯＡＫＳ－ＬＡＢＯには２個のステッピングモータを使用しています。ステッピングモータは、パラレルポートから１と０のデータを与えるだけで簡単に回転制御が可能です。また、ロボットの位置は４個のラインセンサの位置を検出し、その値から右方向に制御するか左方向に制御するかを判断し、走行ラインに沿って走行させる事ができます。

(1)、ステッピングモータの駆動方法

図１８は、ＣＰＵボードとステッピングモータの接続を表したものです。右モータは、パラレルポート１(Port_1)のビットＰ１０とビットＰ１１の２本で制御します。また、左モータは、パラレルポート２(Port_2)のビットＰ２０とビットＰ２１の２本で制御します。

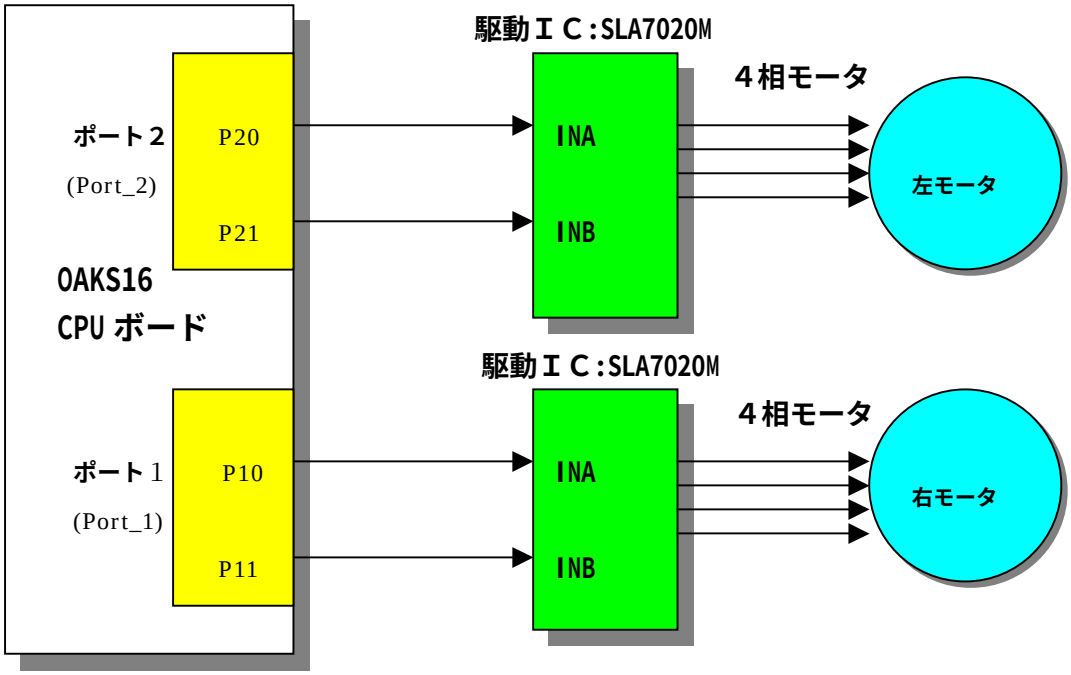


図１８　ＣＰＵボードとステッピングモータの接続

各ポートの出力は、モータ駆動ＩＣ（ＳＬＡ7020M）のＩＮＡ端子とＩＮＢ端子から入力し、ＩＣ内部で２相励磁パターンを作ります。

図１９は、２相励磁パターンを作る為に各ポートから出力するタイムチャートです。

ポート		CW 回転			
Clock		0	1	2	3
P0	INA	1	1	0	0
P1	INB	0	1	1	0

(a) 時計回転

ポート		CCW 回転			
Clock		0	1	2	3
P0	INA	0	0	1	1
P1	INB	0	1	1	0

(b)反時計回転

図１９　励磁信号タイムチャート

ＣＰＵボートのポート１（Ｐｏｒｔ＿１）とポート２（Ｐｏｒｔ＿２）の各Ｐ０（ビット０）とP1（ビット 1）を用いているので、モータをＣW回転させるには(ａ)図のように、０１ｈ→０３ｈ→０２ｈ→００ｈの順番に繰り返し出力します。また、ＣＣW回転させるには(ｂ)図のように、００ｈ→０２ｈ→０３ｈ→０１ｈの順番に繰り返し出力します。

図２０は、励磁信号タイムチャートを配列を用いてＣ言語で記述した例です。

時計方向回転データ関数名　　時計方向回転データ配列

```
char cwdata(char dr)
{
        char    cw_tbl[4] = {0x01,0x03,0x02,0x00};
        return  cw_tbl[dr];
}

char ccwdata(char dl)
{
        char  ccw_tbl[4]= {0x00,0x02,0x03,0x01};
        return  ccw_tbl[dl];
}
```

反時計方向回転データ関数名　　反時計方向回転データ配列

図２０　モータ回転データのプログラム記述例

モータの回転速度を制御するには、ＣＰＵボードから励磁信号を出力してから次の励磁信号を出力するまでの時間間隔与えます。この時間は、一般的に次のような方法で作ることができます。

① ソフトウェアタイマ方式

ステッピングモータのパルス間隔をプログラムのループによる時間稼ぎによって回転を制御する方式です。図２１はソフトタイマ方式のフローチャートです。この方式では、それぞれのモータのタイマデータを交互にデクリメントしてゼロになった方のモータに回転データを出力します。モータカウント値が０になるまでループし、０になつたらモータを駆動し、カウント値を初期値に戻して再びカウントダウンを行います。この方法では簡単にプログラムは記述できますが、両モータを交互に制御する過程でプログラムの時間ループが終わるまで次の処理ができないので高速走行には不向きです。

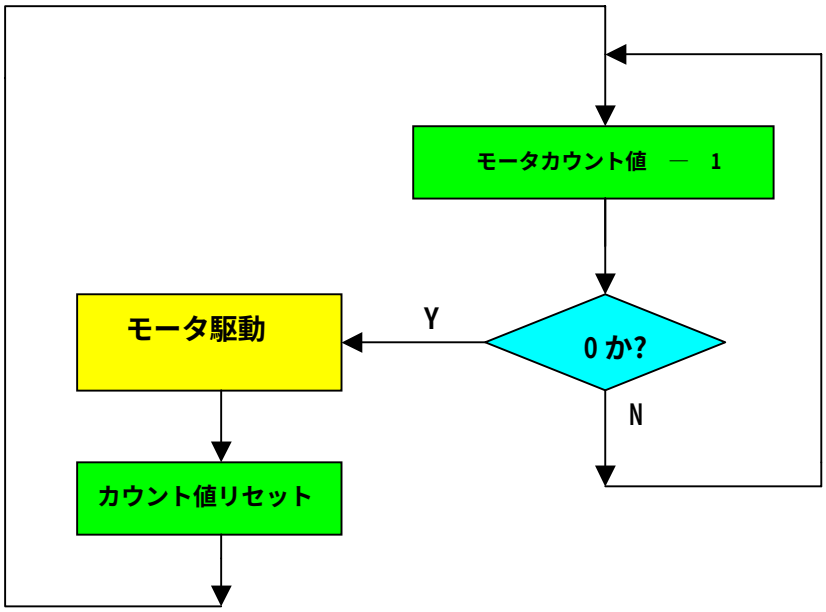


図２１ ソフトタイマ方式でのモータ駆動

図２２は、図２１で示されたソフトタイマ方式フローチャートをＣ言語で記述したプログラム例です。右モータのカウント値を入れる変数がＲ＿ＣＯＮＳＴＰです。また、左モータのカウント値を入れる変数はＬ＿ＣＯＮＳＴＰです。それぞれのカウント値が“０”になるとモータを駆動し、カウントデータを初期値ＣＯＮＳＴＰに戻して無限ループから抜け出します。

右モータ値から１を引く　　右モータ駆動関数

```
void m1(void)
{
        for(;;){
                --R_CONSTP;         /* 右パルス間隔データディクリメント */
                if(R_CONSTP==0)
                {
                        rmout();
                        R_CONSTP=CONSTP;
                        break;
                }
                --L_CONSTP;         /* 左パルス間隔データディクリメント */
                if(L_CONSTP==0)
                {
                        lmout();
                        L_CONSTP=CONSTP;
                        break;
                }
}
```

左モータ値から１を引く　　左モータ駆動関数

図２２　ソフトタイマ方式のプログラム記述例

② タイマ割り込み方式

左右のモータをそれぞれ別々のタイマ割り込みによって回転速度を制御する方式です。図２３はタイマ割り込み方式のフローチャートです。

この方式は、プログラムにタイマ割り込みの知識が必要となりますが、左右のモータに割り当てたタイマの時間定数を変更することで独立して回転速度を制御できます。割込みのあり、なしの判断に割込みフラグを使用します。タイマＡをタイマモードに設定し、その割込み先をモータ駆動にします。時間定数値をセットし、割込みがある迄待ちます。割込みが発生し、モータを駆動してフラグを１にセットします。時間定数に加速データを与える事により、より高速走行が可能となります。

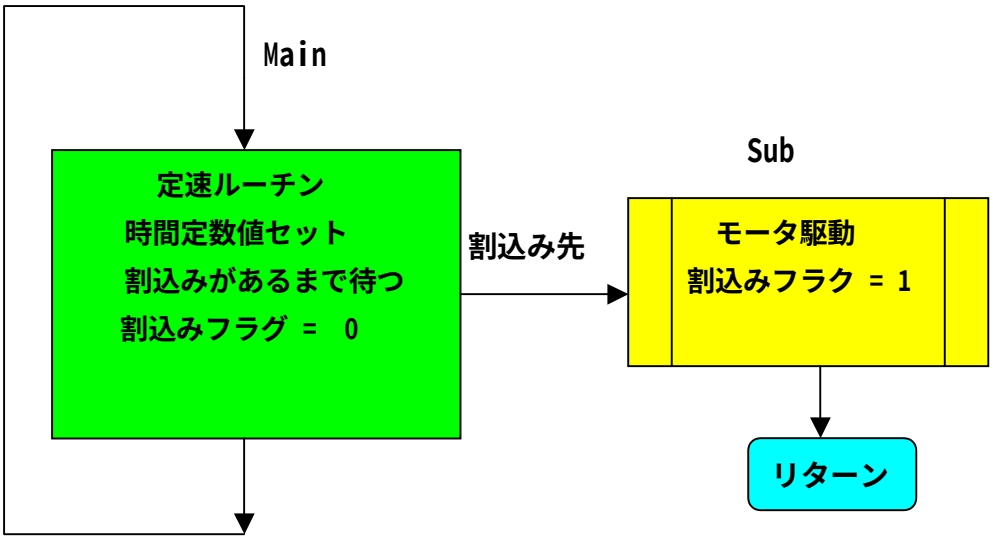


図２３ タイマ割込み方式でのモータ駆動

図２４（ａ）は、図２３図で示されるフローチャートを C 言語で記述したプログラム例です。右モータの時間設定を行うタイマ変数がｔａ０です。また、左モータの時間設定を行うタイマ変数はｔａ１です。直進する場合には、2 つのタイマ変数に規定値 MAXSPEED を設定します。右モータ駆動割込みが終了すると割込みフラグ「ｉｌｒ」を”１”にします。また、左モータ駆動割込みが終了すると割込みフラグ「ｉｆｌ」を”１”にします。したがって、無限ループで何れかのフラグが”１”になるまで待ちます。最後に 2 つのフラグを”０”に戻して割込みを可能にします。

規定値

```
Void  noncr(void)  /* 直進 ルーチン */
{
        ta0 = MAXSPEED;    /* 右定速度回転  */
        ta1 = MAXSPEED;    /* 左定速度回転 */
         for(;;){
                f(ifr == 1 | ifl == 1) break;
         }
         ifr=0;
         ifl=0;
}
```

（ａ）直進ルーチン

図２４（ｂ）は、ロボットを少し右に修正する場合のプログラム例です。直進する場合には、右モータのタイマ変数には規定値 MAXSPEED と可変値 COR１D__L

を加えます。また、左モータのタイマ変数は規定値 MAXSPEED と可変値 COR１D__H を加えます。
可変値は、COR１D__L ＞ COR１D__H となる値を与えるので、右モータのタイマ値が左モータのタイマ値よりも大きくなります。したがって、右モータの回転速度が左モータの回転速度よりも遅くなり、ロボットは右方向に進みます。
左右２つの可変値を組み合わせると、方向を自由に変える事が可能になります。

ta0 > ta1 となるように設定。

```
void  rc1(void)        /* 少し右修正 ルーチン */
{
        ta0 =MAXSPEED+COR1D_L;  /* 右モータを少し低速に設定  */
        ta1 =MAXSPEED+COR1D_H;
         for(;;){
                 if(ifl == 1) break;
         }
         ifl=0;
}
```

（b）少し右修正ルーチン

図２４　タイマ割込み方式のプログラム記述例

(2)、ラインセンサ入力

コース上のロボットの位置と 4 個のセンサ反応の位置関係を図２５に示します。センサの反応は、コース上(白)で ”０”(Ｌレベル)、コースから外れた場所では ”１”(Ｈレベル)の出力になります。このデータをビット反転してコース上で ”1”、 コースから外れた場所では ”０”にして判定した真理値表が表５です。
ロボットがコースに対して ”かなり右寄り ”の位置では、センサ出力は一番左側のセンサがコースに位置しているので、 ”０”となり、残りの３のセンサ出力は ”１”となります。したがって全出力は ”０１１１”となりますが、表５では、このデータをビット反転したデータ ”１０００”で表現します。
ロボットを停止させるには、４個全てのセンサが ”１１１１“ となるように、コースに対して直角に４個のセンサの距離長分のラインテープを貼れば、ロボットはその位置で停止します。
制御の考えは車の運転と同じで右に寄ったら左側のモータの回転を下げればロボットは左に曲がります。また逆に、左に寄ったら右側のモータの回転を下げればロボットは右に曲がります。

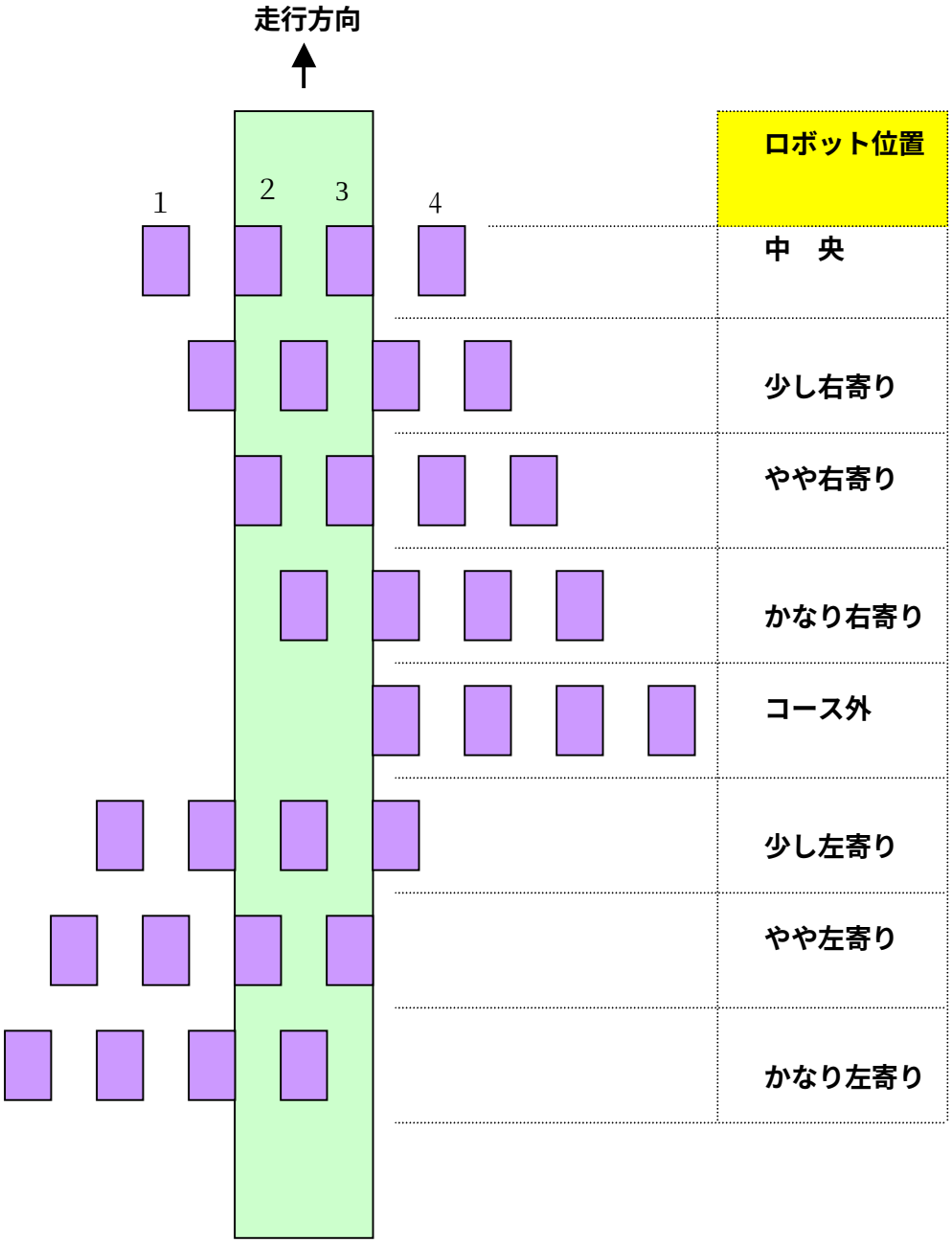


図２５　ロボットの位置とセンサの関係

図２６は、表５を元にＣ言語で記述した位置修正プログラム例です。センサからの入力はｓｅｎｃｅ（）関数です。このデータをｓｗｉｔｃｈ文でロボットの位置を判別します。例えば、センサデータが”１”ならば、ｃａｓｅ1文を実行します。ロボットは、かなり右に寄っているのでｌｃ3（）関数でかなり左に修正します。

表５　コース位置判定真理値表

ロボットの位置	センサ出力データ（ビット反転出力）				
	２進数				１６進数
	0	1	2	3	
中　央	0	1	1	0	０６ｈ
少し右寄り	0	1	0	0	０２ｈ
やや右寄り	1	1	0	0	０３ｈ
かなり右寄り	1	0	0	0	０１ｈ
はずれ	0	0	0	0	００ｈ
少し左寄り	0	0	1	0	０４ｈ
やや左寄り	0	0	1	1	０Ｃｈ
かなり左寄り	0	0	0	1	０８ｈ

センサからのデータ　　スイッチ文で位置を判別

```
void teisoku()                          /* 定速ルーチン */
{
        int     sed;
        sed = sence();
        switch(sed)
        {
                case 1: lc3();  break;          /* かなり左修正 */
                case 2: lc1();  break;          /* 少し左修正 */
                case 3: lc2();  break;          /* やや左修正 */
                case 4: rc1();  break;          /*少し右に修正 */
                case 6: noncr();break;          /* 中央 */
                case 8: rc3();  break;          /* かなり右修正 */
                case 12: rc2(); break;          /* やや右に修正 */
                case 15: stop_line(); break;    /* ストップライン  */
                default:
                        noncr(); break;         /* 何れでもない */
                }
}
```

図２６　位置修正プログラム例

10.添付ファイルの構成と使い方

ＯＡＫＳ-ＬＡＢＯには表６で示されるファイルがＣＤ-Ｒに入っています。目的に応じて自由に取り出してご使用下さい。ただし、添付されている全てのＣ言語プログラムには著作権が存在しますので商用として使用する事はできません。

表６　ＯＡＫＳ－ＬＡＢＯ添付ファイル一覧表

ディレクトリ名	ファイル名	内　　容
Demo_1	Oklabo_1.c Oklabo_1.x30	ソフトタイマ方式ソースプログラム。 ソフトタイマ方式デバッグファイル。
Demo1_rom	Oklabo_2.c Oklabo_2.mot	ソフトタイマ方式ＲＯＭ化ソースプログラム。 ソフトタイマ方式ＲＯＭ化ファイル。
Demo_2	Oklabo_3.c Oklabo_3.x30	タイマ割込み方式ソースプログラム。 タイマ割込み方式デバッグファイル。
Demo2_rom	Oklabo_4.c Oklabo_4.mot	タイマ割込み方式ＲＯＭ化ソースプログラム。 タイマ割込み方式ＲＯＭ化ファイル。
Document	OAKSLABO 取扱説明書	OAKS-LABO ユーザーズマニュアル。 (PDF ファイル形式)
Monitor	Mon_main.mot	デバッガ起動用モニタプログラム。

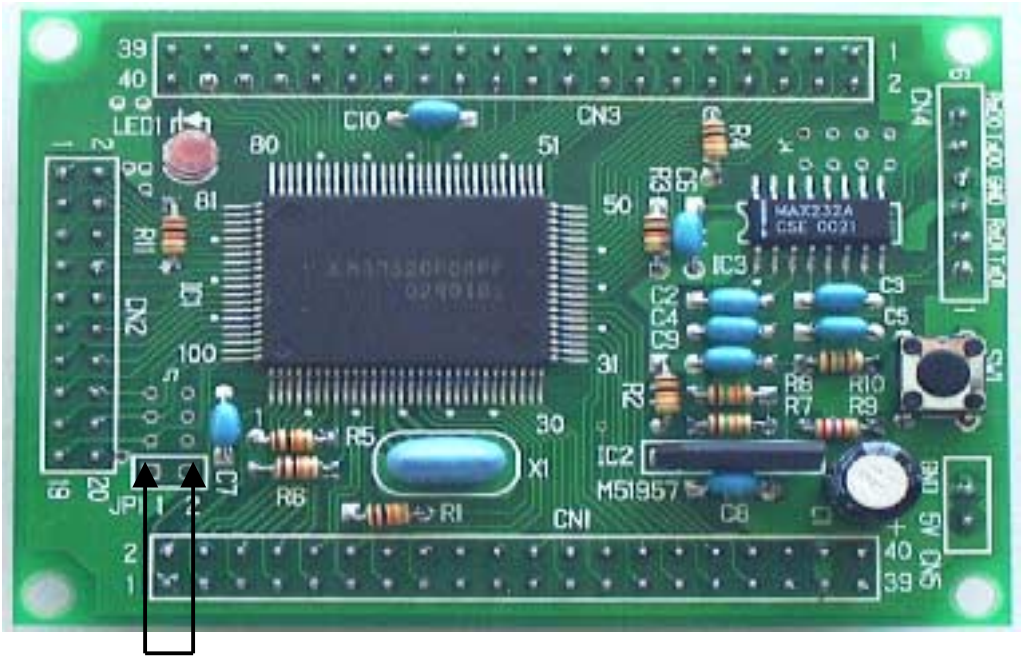
ファイルの拡張子が「．ｃ」はＣ言語ソースプログラムです。「.ｘ３０」はデバッガが使用するオブジェクトファイルです。「.ｍｏｔ」は「.ｘ３０」ファイルをモトローラ２形式に変換したファイルで、フラッシュＲＯＭに書き込むファイルです。

出荷時には、タイマ割込み方式のＲＯＭ化ファイル「Ｏｋｌａｂｏ＿４．ｍｏｔ」がフラッシュＲＯＭに書き込まれております。

(1)、デバッガを使用する為のフラッシュライタソフトの使い方

デバッガ「ＫＤ３０」を用いてデモプログラムを実行するには、予めフラッシュＲＯＭにモニタファイル「Ｍｏｎ＿ｍａｉｎ．ｍｏｔ」ファイルを書き込まなければなりません。次の手順で書き込んで下さい。

ＯＡＫＳ-ＬＡＢＯとパソコン間をＲＳ２３２Ｃケーブル(ストレート線) で接続します。また、図２７に示されるＣＰＵボード「ＯＡＫＳ１６」のＪＰ1 端子１ －２間を何らかのショート端子を用いてショートして下さい。



ＪＰ１の１－２間をショート端子でショートします。

図２７　ＣＰＵボード「ＯＡＫＳ１６」のＪＰ1 端子

ＯＡＫＳ－ＬＡＢＯの電源をオンにし、フラッシュライタソフト「M16ＣＦｌｓｈ」アイコンをクリックすると図２８で示されるウインドウが表示します。

① 「Ｒｅｆｅｒ」ボタンをクリックし、モニタプログラム「ｍｏｎ＿ｍａｉｎ．ｍｏｔ」を登録します。

② 空欄になっている七つのＩＤウインドウに ”00 ” を書き込んで下さい。

③ 「ＯＫ」ボタンをクリックします。

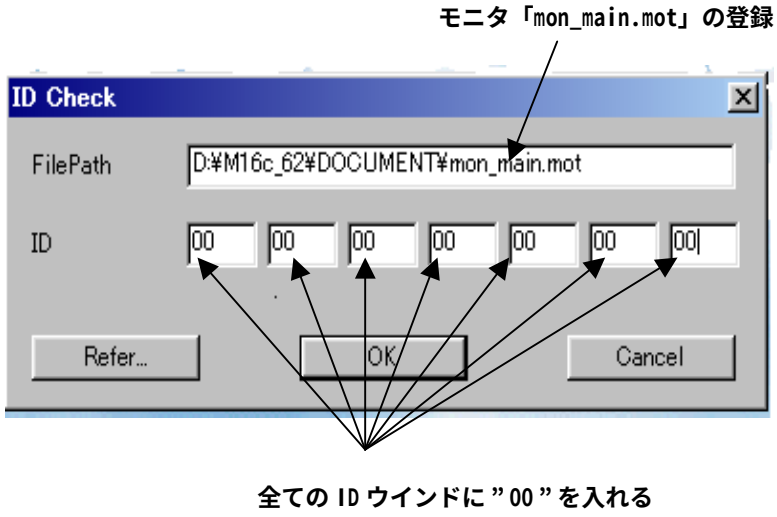


図２８　「ID Check」　ウインドウ

④　図２９で示される「Ｍ１６Ｃ　ｆｌａｓｈ　Ｗｒｉｔｅ」ウインドウが表示します。

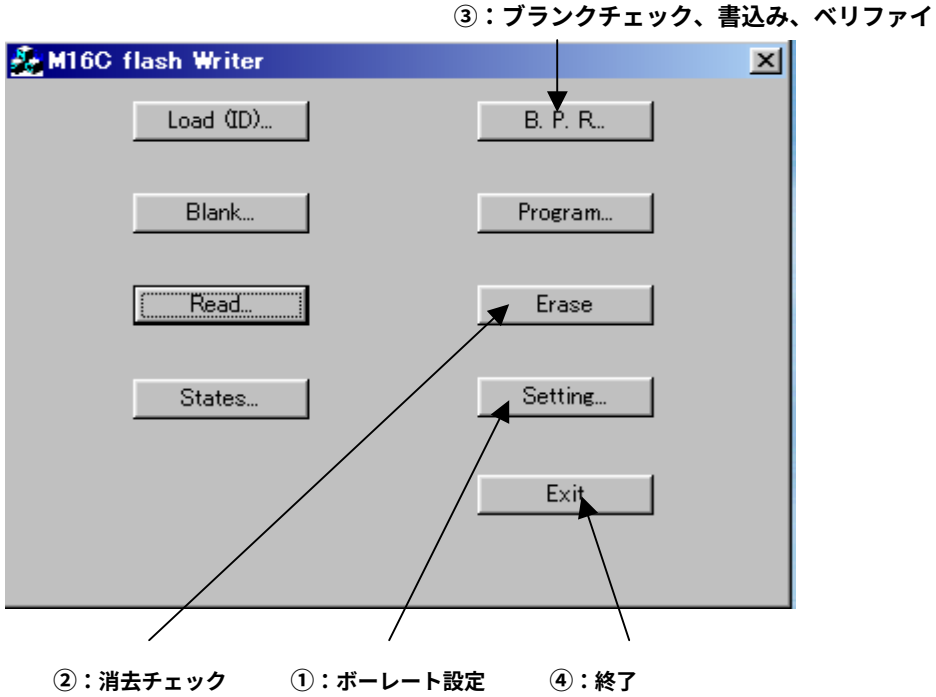


図２９　「Ｍ１６Ｃ　ｆｌａｓｈ　Ｗｒｉｔｅ」ウインドウ

⑤　「Ｓｅｔｔｉｎｇ」ボタンを押します。図３０で示されるウインドウ内の「Ｂａｕｄｒａｔｅ」設定プルダウンメニューからボーレートを図のように５７６００を選んで下さい。設定後、「ＯＫ」ボタンを押して下さい。

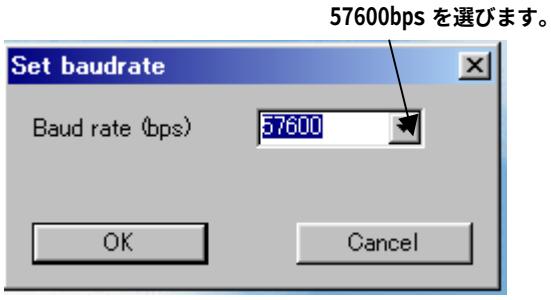
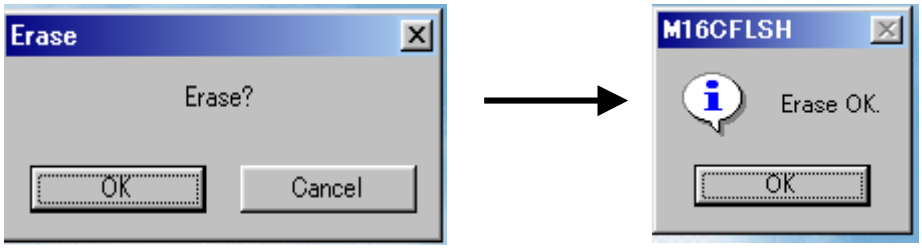


図３０　「Ｓｅｔ　ｂａｕｄｒａｔｅ」ウインドウ

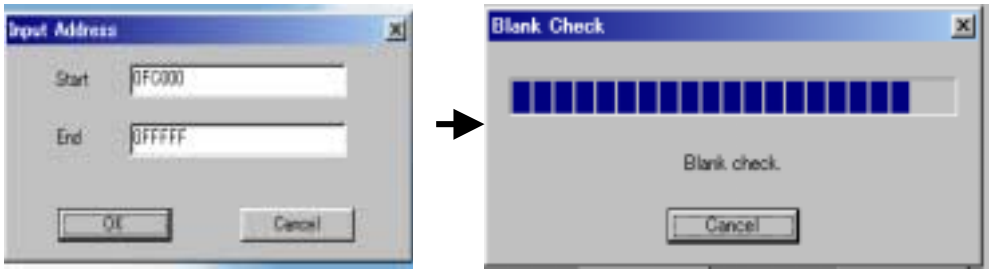
⑥　「Ｅｒａｓｅ」ボタンを押します。図３１（ａ）で示されるウインドウの「ＯＫ」ボタンを押すと、（ｂ）図で示される「Ｅｒａｓｅ　ＯＫ」ウインドウが表示します。



（ａ）　　（ｂ）

図３１　「Ｅｒａｓｅ」チェックウインドウ

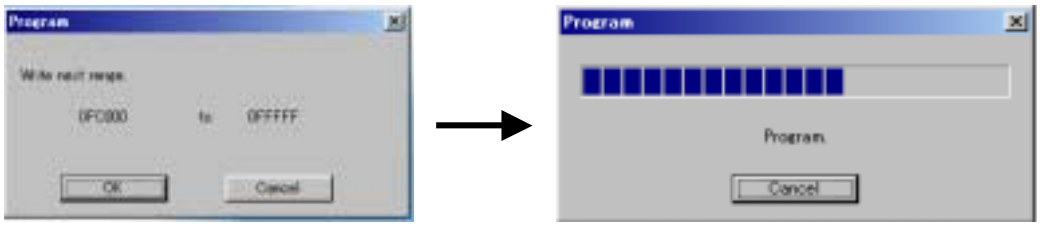
⑦　「Ｂ．Ｐ．Ｒ」ボタンを押します。このボタンは、ブランクチェック、プログラムの書込み、書込み確認を一連で実行します。図３２（ａ）で示されるプログラムのメモリ範囲が表示されます。確認後、「ＯＫ」ボタンを押します。図（ｂ）のウインドウが開きブランクチェックを開始します。



（ａ）　　（ｂ）

図３２　「Ｂｌａｎｋ　Ｃｈｅｃｋ」ウインドウ

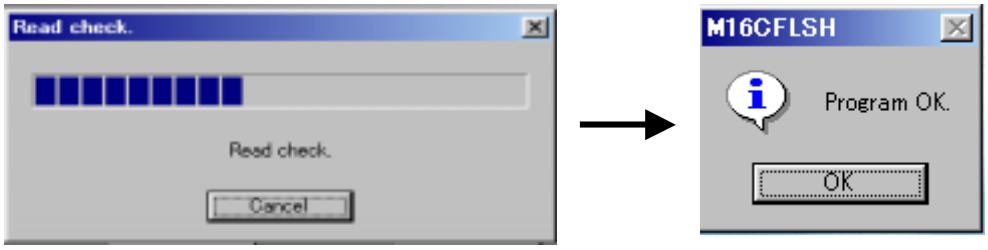
⑧　ブランクチェックが正常に終了すると、図３３（ａ）に示されるウインドウが表示されます。「ＯＫ」ボタンを押すと図（ｂ）のウインドウが開き、書込みを開始します。



（ａ）　　（ｂ）

図３３　「Ｐｒｏｇｒａｍ」ウインドウ

⑨　プログラムが正常に書き込まれると、図３４（ａ）に示されるウインドウが表示され、書込みの確認を行います。正常に書き込まれていると図（ｂ）のウインドウが開き、書込み確認が終了します。ＯＫ」ボタンを押して下さい。



（ａ）　　（ｂ）

図３４　「Ｒｅａｄ　ｃｈｅｃｋ」ウインドウ

⑩　最後に、図２９で示される「Ｍ１６Ｃ　ｆｌａｓｈ　Ｗｒｉｔｅ」ウインドウから「Ｅｘｉｔ」ボタンを押して下さい。

(2)、デバッガの起動方法

デバッガ「KD30」を起動するには、ＯＡＫＳ－ＬＡＢＯのシリアルポートからチャンネル１とパソコン間をＲＳ-２３２Ｃケーブル（スレートケーブル）で接続します。

デバッガ「KD30」アイコンをクリックすると図３５に示される起動ウインドウが表示されのす。パソコン側のシリアル端子はＣＯＭ１に固定されています。

ロボットの電源をＯＮにして下さい。また、ＲＳ-２３２Ｃケーブルの接続を確認後、「ＯＫ」ボタンを押します。もし、エラーウインドウが表示される場合には、ケーブルの接続や電源を再確認します。また、予めフラッシュＲＯＭにモニタファイル「Ｍｏｎ_ｍａｉｎ．ｍｏｔ」ファイルが書き込まれていなければなりません。

図３５で示される起動ウインドウの詳しい使用方法は「KD30」に添付されているマニュアルをご覧下さい。

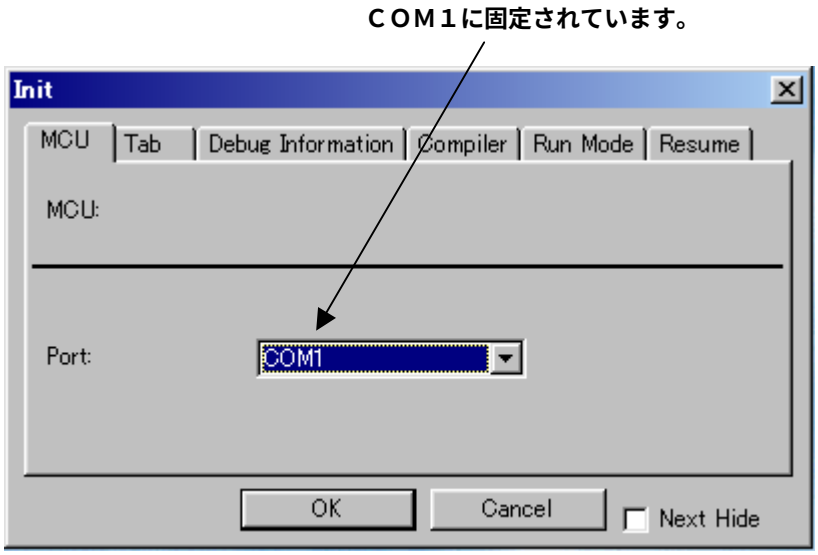


図３５　デバッガ起動ウインドウ

「ＯＫ」ボタンを押すと、図３６で示されるデバッグウインドウが表示されます。

①　ファイルのロード方法

ファイルをロードするには、メニューから左隅の「Ｆｉｌｅ」をクリックします。プルダウンメニューから「Ｄｏｗｎｌｏａｄ」を選びます。更にこの中の「Ｌｏａｄ　Ｍｏｄｕｌｅ」を選ぶとファイルロードウインドウが開きます。ロード可能のファイルは、拡張子に「．ｘ30」です。デモ用のファイルから「Ｏｋｌａｂｏ_1．ｘ30」と「Ｏｋｌａｂｏ_3．ｘ30」がロード可能です。

② プログラムの実行方法

プログラムを実行するには、通常プログラムカウンタにロードしたユーザプログラムの先頭番地を予め設定する必要があります。しかし、デモプログラムの場合には、ロードの段階で自動的にプログラムカウンタに先頭番地が設定されます。

　プログラムの実行は、左隅の実行アイコン「↓」をクリックするとプログラムが“ＲＵＮ”します。

図３６で示されるデバッグウインドウの詳しい使用方法はリモートデバッガ「ＫＤ３０」に添付されているマニュアルをご覧下さい。

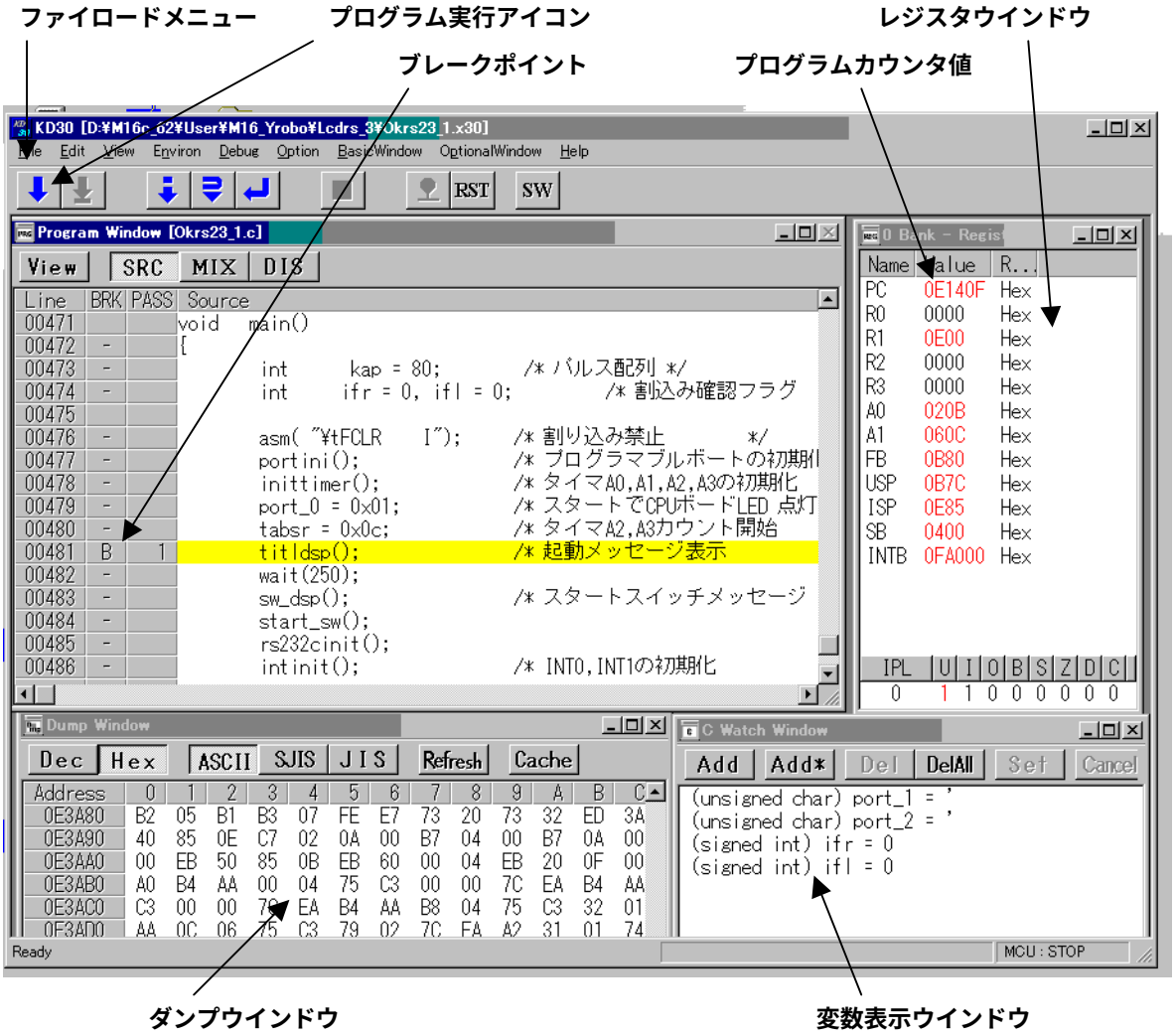


図３４　デバッガＫＤ３０デバッグ画面

11. ＴＭ統合化開発環境での使い方

ＯＡＫＳ－ＬＡＢＯ用ＲＳ-２３２Ｃ無線ユニットには、最新のＴＭ統合開発環境「ＴＭＶ３.１１」が付属しています。このＴＭ統合化開発環境を用いることにより、バッチファイルまたはメイクファイルによりＤＯＳ窓上でコンパイルを行っていた方式からＷｉｎｄｏｗｓ上でエディタの起動、コンパイラ、デバッガの起動等、統合操作を可能にしました。

（１）インストールと注意点

付属のＣＤ－Ｒには、ＴＭ統合化開発環境「ＴＭＶ３.１１」とコンパイラ「ＮＣ３０ＷＡ　Ｖ４.００　Ｒｅｌｅａｓｅ ２(エントリー版)」が入っています。既に、ライントレースロボット「ＯＡＫＳ－ＬＡＢＯ」に添付されているコンパイラ「ＮＣ３０ＷＡ　Ｖ３.００」がインストールしている場合、インストール中の「コンポーネントの選択」画面で「インストール先ディレクトリ」項目を「参照（Ｒ）」を使用して別のインストール先を指定してインストールをして下さい。もし、同じディレクトリにインストールする場合、必ず、既にインストールされているＮＣ３０ＷＡをアンインストールした後で、本製品のＮＣ３０ＷＡをインストールして下さい。

ＯＡＫＳ-ＬＡＢＯに添付されている全てのデモサンプルプログラムを旧バージョン「ＫＮＣ３０　Ｖ３.００」でコンバイルするとエラーが発生します。したがって、本製品に入っている全てのデモサンプルプログラムは、旧バージョンではコンパイルできません。

（２）エントリー版（新バージョン）の仕様について

① プログラムの記述について

旧バージョンのプログラムをエントリー版でコンバイルする際、表７で示される語の前にアンダースコア（＿）を付加して下さい。

表７　新旧バージョンの記述相違一覧

KNC30 旧バージョン版	NC30 エントリー版
ｉｎｌｉｎｅ	＿ｉｎｌｉｎｅ
ｎｅａｒ	＿ｎｅａｒ
ｆａｒ	＿ｆａｒ
ａｓｍ ()	＿ａｓｍ ()

② ＴＭ統合開発環境「ＴＭＶ３.１１」について

- インスペクタ機能はご使用になれません。
- ブラリプロジェクトは作成できません。

② ソフトウェアおよびユーティリティについて
下記に示すソフトウェアおよびユーティリティはご使用になれません。

表８　使用できないソフトウェアおよびユーティリティ一覧

ソフトウェア	STK ビューワ，MPA ビューワ，アセンブルオプティマイザ(aopt30)，ライブラリアン(lb30)，構造化記述アセンブラ(pre30),] 標準関数ライブラリソースファイル
ユーティリティ	utl30(SBDATA 宣言&SPECIAL ページ宣言ユーティリティ

④ ＮＣ３０とＡＳ３０のオプションについて
マニュアルに記載されている下記のオプションはご使用になれません。

表９　使用できないＣコンパイラオプション一覧

Ｃコンパイラ（ＮＣ３０）	
デバッグ用オプション	genter　-gno_reg
最適化オプション	-O[1-5], -OR, -OS, -Oconst(-OC), -Ono_bit(-ONB),-Ono_break_source_debug(-ONBSD), -Ono_float_const_fold(-ONFCF), -Osp_adjust(-OSA), -Ostack_frame_align(-OSFA), Oloop_unroll(-OLU), Ono_asmopt(-ONA), -Ono_logical_or_combine(-ONLOG), -Ocompare_byte_to_word(-OCBTW), -Ono_stdlib
生成コード変更オプション	-finfo, -fuse_DIV(-fUD), -fansi, -fnear_ROM(-fNROM), -fsmall_array(-fSA), -fno_align(-fNA)
アセンブル、リンクオプション	-as30, -ln30
その他のオプション	-dsource(-dS), -dsource_in_list(-dSL)

表１０　使用できないアセンブラオプション一覧

アセンブラ（ＡＳ３０）	
オプション	-finfo, -P, -M

（３） ＴＭ統合化開発環境の使い方

ＴＭ統合化開発環境「ＴＭ」を使用する場合、ライントレースロボット「ＯＡＫＳ－ＬＡＢＯ」に添付されているコンパイラ「ＫＮＣ３０」(旧バージョン)ではご使用になれません。「ＴＭ」を使用する場合には、本製品に入っているエントリー版コンパイラ「「ＮＣ３０」(新バージョン)を使用します。「ＴＭ」を起動すると細長いプロジェクトバーが表示されます。マウスによって右端を縮めると図３５で示されるフローティング状態のプロジェクトバーが表示されます。

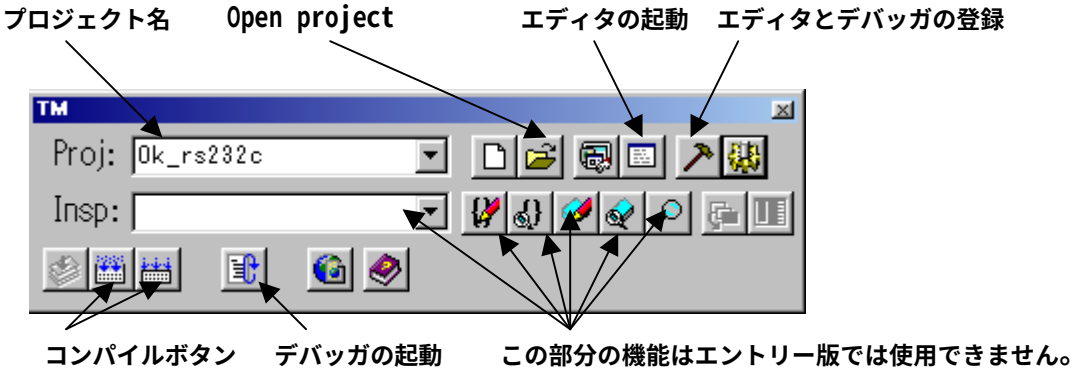


図３５　プロジェクトバー

① エディタとデバッガの登録

最初に (ツールの登録)ボタンをクリックし、使い慣れたエディタとリモートデバッガ「ＫＤ３０」を登録します。

ここで登録しないと、図１９図の (エディタの起動)ボタンと (デバッガの起動)ボタンが使用できません。

② プロジェクトファイルのロード

ＴＭ統合化開発環境「ＴＭ」を用いてＷｉｎｄｏｗｓ上でコンパイルを行うには、プロジェクトファイルを作成しなければなりません。新規のユーザプログラムからプロジェクトファイルを作成しコンパイルする方法は、付属のユーザズマニュアル（ＰＤＦファイル）をご覧下さい。本製品のデモサンプルプログラムにはプロジェクトファイル「Ｏｋ＿ｒｓ232ｃ．ｔｍｋ」が入っています。

プロジェクトファイルをロードするには図１９で示され (プロジェクトを開く)ボタンをクリックします。開くウインドウから付属のＣＤ－Ｒからコピーしたドライブとディレクトリを選び「Ｏｋ＿ｒｓ232ｃ．ｔｍｋ」をロードします。図１９で示されるプロジェクトウインドウにファイル名が表示されます。

③ コンパイル方法

プロジェクトで設定したソースプログラム「Ｏｋ＿ｒｓ232ｃ．ｃ」をコンパイル

するには、 (ビルドまたはリビルド)ボタンをクリックします。ウインドウにコンパイルメッセージが表示されます。スクロールバーによって前後のメッセージを読むことが可能です。

④ エディタの起動

(ツールの登録)ボタンによって前もって使い慣れたエディタを登録しておくと、(エディタの起動)ボタンでエディタが起動します。ソースプログラムをロードしてプログラムを修正できます。

⑤ デバッガの起動

(ツールの登録)ボタンによってリモートデバッガを登録しておくと、(デバッガの起動)ボタンをクリックするとリモートデバッガ「ＫＤ３０」が起動できます。

図３５において、２段目のインスペクタとその関連機能は、エントリー版では使用できません。また、ＴＭ統合化開発環境「ＴＭ」の詳しい使用方法は、本製品のＣＤ－Ｒに入っているユーザーズマニュアルをご覧下さい。

12.プログラムの作成手順

ＯＡＫＳ－ＬＡＢＯには、プログラム開発が容易に行えるようにＣコンパイラ、アセンブラ、リモートデバッガ、フラッシュライタソフトが付属しています。Ｃ言語プログラムは、他社のＣＰＵボートで開発したソースプログラムを、ＯＡＫＳ－ＬＡＢＯで使用するＭ１６Ｃ／６２シリーズ用プログラムに容易に移植することが可能です。また、アセンブラも熟知する必要がありませんので、Ｃ言語でプログラムする事をお勧めします。また、商品に添付しているデモプログラムは全てＣ言語プログラムとなっています。

(1)、プログラム開発の手順

プログラムを開発するには、図３６に示されるように次の手順でプログラムを開発します。

① プログラム図法を用いてフローチャート等のアルゴリズムを作成します。
② 使い慣れたエディタを用いてソースプログラムを記述します。
③ コーディングが終了したら、コンパイラ「ＫＮＣ３０ＷＡ」を用いてコンパイルを行います。
④ エラーメッセージが表示された場合には、②項に戻り、エラーを修正します。
⑤ コンパイルが終了したら、リモートデバッガ「ＫＤ３０」を用いて、ロボットがソフトウェア設計通りに動作するか確かめます。
⑥ 正常に動作しない場合には、原因を究明し、②項に戻りプログラムを修正します。
⑦ 正常にロボットが動作したら、フラッシュライタソフト「M16ＣＦｌｓｈ」を用いて

フラッシュＲＯＭにプログラムを書き込みます。

⑧　プログラム開発を終了します。

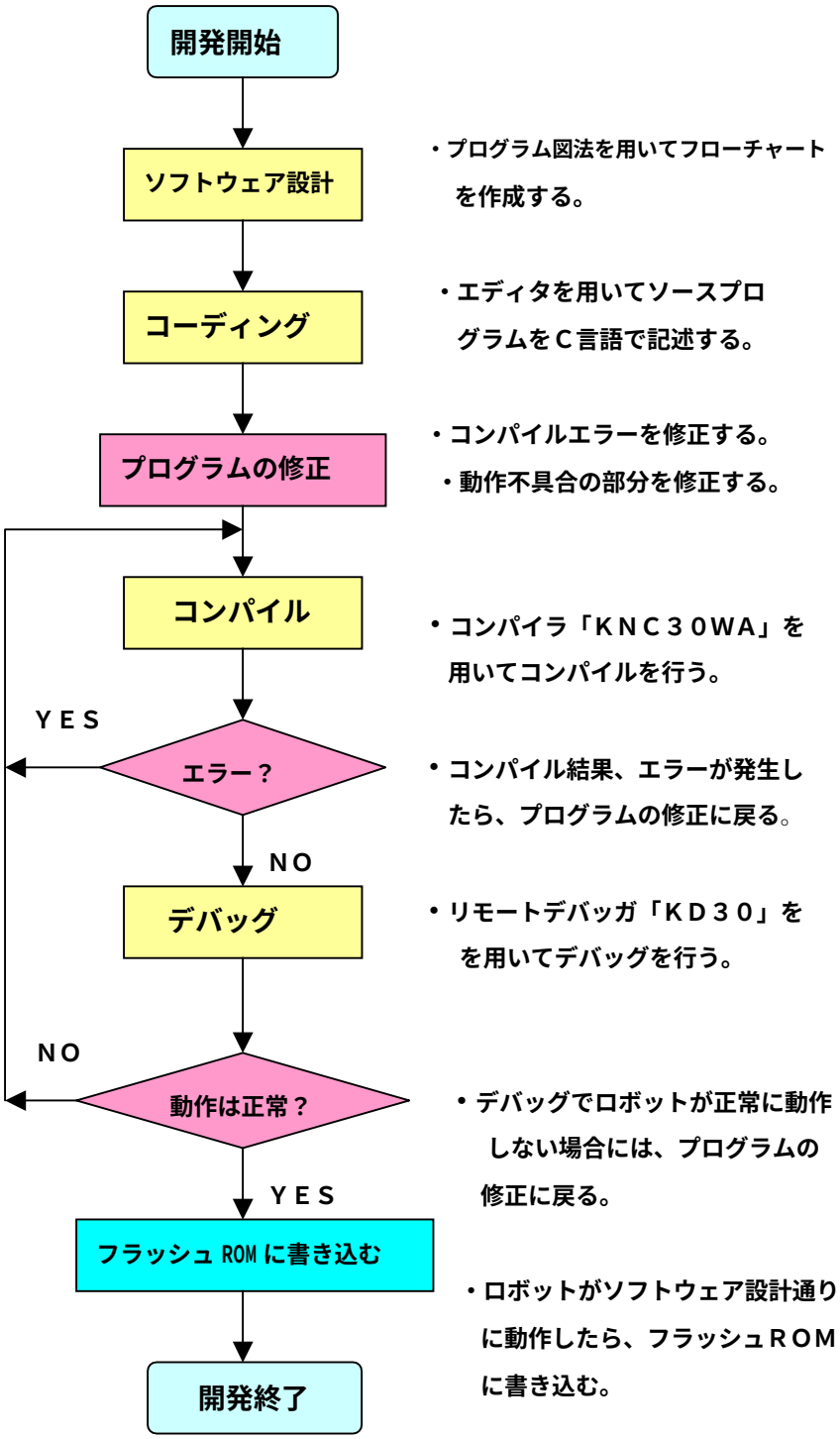


図３６プログラム開発の手順

(2)、コンパイル方法

M16C／６０シリーズ専用のコンパイラ「ＮＣ３０ＷＡ」にはＣコンパイラとアセンブラが標準で付属しております。図３７はコンパイル手順を示したフローチャートです。作成したＣ言語プログラム「Ｏｋｌａｂｏ＿４．ｃ」はコンパイルドライバ 「ｎｃ３０」によってアセンブラプログラム「Ｏｋｌａｂｏ＿４．ｒ３０」に変換します。

Ｃソースプログラムにエラーがなければ、スタックポインタ等を記述したスタートアップルーチン「ｎｃｒｔ０．ａ３０」と割り込みを使用する場合には、インクルードファイル「ｓｅｃｔ３０．ｉｎｃ」の割り込みベクタテーブルを書き換えてアセンブルします。全てのファイルにエラーがなければリンカでリンクします。リンク段階でエラーがなければデバッグ可能な実行ファイルが完成します。次に、リモートデバッガ「ＫＤ３０」を用いてプログラムが正常に動作する事を確認後、ファイル形式変換ソフト「Ｌｍｃ３０」によってモトローラ２形式のファイル「Ｏｋｌａｂｏ＿４．ｍｏｔ」に変換し、フラッシュライタソフト「M16ＣＦｌｓｈ」で書込み、プログラム作成は終了です。

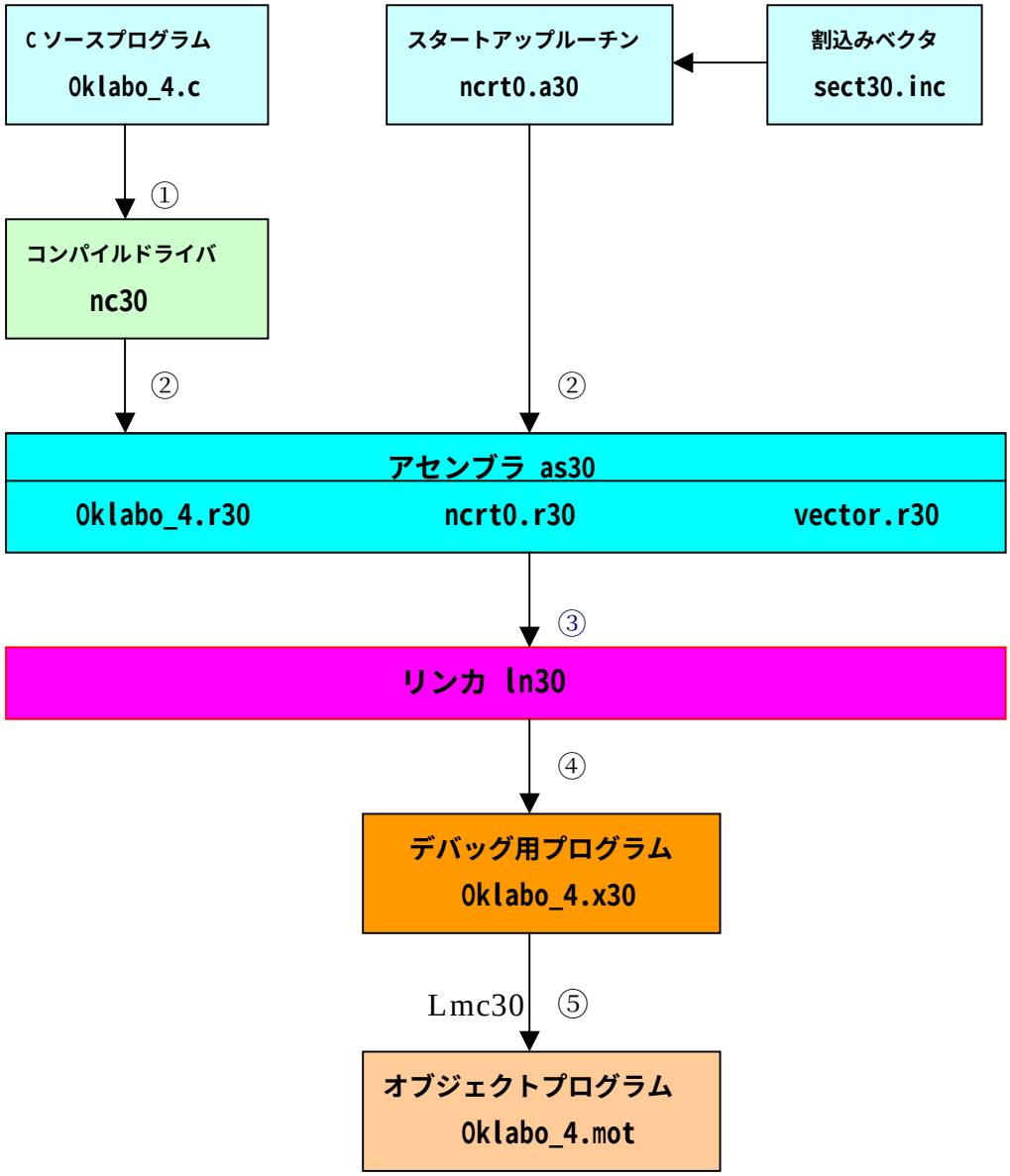


図３７ コンパイラ「ＮＣ３０ＷＡ」のコンパイル手順

(3)、コンパイル用バッチファイル例

コンパイラ「ＮＣ３０ＷＡ」は通常、ＴＭ統合化開発環境の元でコンパイルすると便利ですが、バッチファイルによってＤＯＳ窓上でコンパイルする事ができます。

図３８は、バッチファイルに予めコンパイルの手順を記述した記述例です。

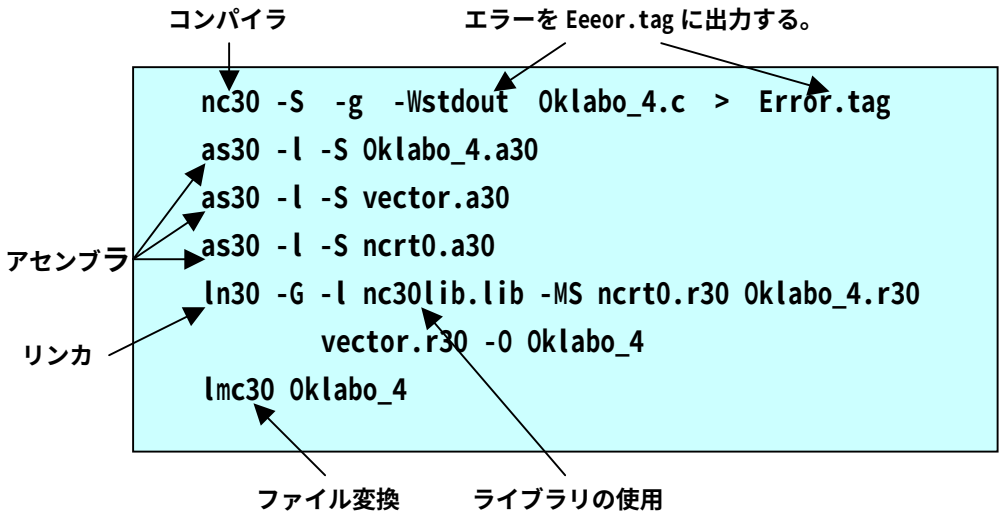


図３８ コンパイル用バッチファイル記述例

図３８で示されるバッチファイルは、出荷時にフラッシュＲＯＭに書き込まれているデモソフト「Ｏｋｌａｂｏ＿４．ｃ」をコンバイルする為のバッチファイルです。コンパイラのオプションに記述されている「―Ｗｓｔｄｏｕｔ」は、エラーリストを出力する指定です。行の最後に「　＞　Ｅｒｒｏｒ．ｔａｇ」のように記述すると、エラーリストをＥｒｒｏｒ．ｔａｇファイルに書き込まれます。このエラーファイルを見ながらソースプログラムを修正すると便利です。

図３７に使用されているオプションの詳しい使い方は、コンパイラ「ＮＣ３０ＷＡ」マニュアルをご覧下さい。

（４）Ｉ／Ｏポートの記述方法

Ｍ１６Ｃ／６２シリーズのＣＰＵは、メモリマップＩ／Ｏ方式で全てのＩ／Ｏアドレスを００００００番地から００４００番地に割り付けます。Ｃ言語でＩ／Ｏポートを定義するには、図３９のように拡張機能による絶対アドレス指定を用いると便利です。“＃ｐｒｏｇｒｍ　ＡＤＤＲＥＳＳ”は、関数外で定義された変数および関数内でスタティック宣言された変数のみ有効となります。ポート名(変数名)には、適当な名前を付けて、次のように記述します。

```
#ｐｒｏｇｒａｍ　ＡＤＤＲＥＳＳ　変数名　　絶対アドレス
```

上記の宣言により、変数名を絶対アドレスに配置します。このような拡張機能による絶対アドレス指定を用いると、メモリ使用量を節約できます。

ポート名(変数名)　　ポートアドレス(絶対アドレス)

```
/*************************************************************************
 *   ポートの設定（ポート P0,タイマ A0）
*************************************************************************/
#pragma ADDRESS port_0        3e0H     /* Port P0 register           */
#pragma ADDRESS pd0           3e2H     /* Port P0 direction register  */
#pragma ADDRESS tabsr         380H     /* Count start flag           */
#pragma ADDRESS udf           384H     /* Up-down flag               */
#pragma ADDRESS ta0           386H     /* Timer A0 register          */
#pragma ADDRESS ta0mr         396H     /* Timer A0 mode register     */
#pragma ADDRESS ta0ic         55H      /* Timer A0 interrupt control register */

unsigned char       port_0, pd0;
unsigned char       tabsr, udf, ta0mr, ta0ic;
unsigned int        ta0;
```

図３９　Ｉ／Ｏポートの記述例

(５)、割り込み処理

Ｍ１６Ｃ／６２シリーズのＣＰＵで割り込みをＣ言語関数として記述する手順は次の２つです。

① 割り込み処理関数の定義。

② 割り込みベクタテーブルの登録。

割り込み処理関数の定義は、割り込み関数名に適当な名前を付けて、次のように記述します。

```
#ｐｒｏｇｒａｍ　ＩＮＴＥＲＲＵＰＴ　割り込み関数名
```

上記のように記述すると、指定した関数の入り口と出口において、通常の関数の手続き以外に、全レジスタの退避、復帰と“ｒｅｉｔ”命令を生成します。割り込み処理関数の型は、引数／戻り値共にｖｏｉｄ型のみ有効です。図４０は、タイマＡ０を割り込みに使用する為の割り込み処理関数の記述例です。

割り込み関数名

```
/*****************************************************************
*   割り込み関数定義(タイマ A0)
*****************************************************************/
void far    ta0int();     /* 割込み関数    */
#pragma INTERRUPT  ta0int
```

図４０　割り込みの記述例

(６)、割り込みベクタテーブルへの登録

割り込みを正常に使用する為には、割り込み処理関数を定義すると共に割り込みベクタテーブルに登録する必要があります。
割り込みベクタテーブルの変更は、次の手順で行います。

① 割り込み処理関数名を疑似命令“.ｇｌｂ”で外部定義します。
② 使用する割り込みのダミー関数“ｄｕｍｍｙ＿ｉｎｔ”を、割込み処理関数名に変更します。

図４１は、スタートアップルーチン「ｎｃｒｔ０．Ａ３０」に記述されている見出し文「Interrupt section start」内に割込み処理関数を記述したものです。外部定義では次のように必ずアンダーバーを付けて記述して下さい。

```
.glb        _ta0int
```

また、タイマＡ０を割り込みで使用する場合には、図４２に示されるインクルードファイル「ｓｅｃｔ３０．ｉｎｃ」のベクターテーブル見出し名「variable vector section」２１番のダミー関数“ｄｕｍｍｙ＿ｉｎｔ”を次のように必ずアンダーバーを付けて記述して下さい。

```
.lword      _ta0int
```

割込み処理関数名の記述

```
;=========================================================
; Interrupt section start
;---------------------------------------------------------
    .insf          start,S,0
    .glb           _ta0int
    .glb           start
    .section       interrupt
start:
```

図４１　スタートアップルーチンに割込み処理関数名を記述する例

割込み処理関数名の記述

```
.else
    .lword   dummy_int          ; vector 0 (BRK)
    .org     (VECTOR_ADR +44)
    .lword   dummy_int          ; DMA0 (for user)
    .lword   dummy_int          ; DMA1 2 (for user)
    .lword   dummy_int          ; input key (for user)
    .lword   dummy_int          ; AD Convert (for user)
    .org     (VECTOR_ADR +68)
    .lword   dummy_int          ; uart0 trance (for user)
    .lword   dummy_int          ; uart0 receive (for user)
    .lword   dummy_int          ; uart1 trance (for user)
    .lword   dummy_int          ; uart1 receive (for user)
    .lword   _ta0int            ; TIMER A0 (for user) (vector 21)
    .lword   dummy_int          ; TIMER A1 (for user) (vector 22)
    .lword   dummy_int          ; TIMER A2 タイマ A2(vector 23)
    .lword   dummy_int          ; TIMER A3 タイマ A3(vector 24)
    .lword  dummy_int           ; TIMER A4 (for user) (vector 25)
    .lword   dummy_int          ; TIMER B0 (for user) (vector 26)
    .lword   dummy_int          ; TIMER B1 (for user) (vector 27)
    .lword   dummy_int          ; TIMER B2 (for user) (vector 28)
    .lword   dummy_int          ; INT0 (for user) (vector 29)
    .lword   dummy_int          ; INT1 (for user) (vector 30)
    .lword   dummy_int          ; INT2 (for user) (vector 31)
.endif
```

図４２　ベクターテーブルの記述例

13. Flash ROM への書込み

デバッガによってトレースロボットが正常に動作することが確かめられ後、最後にFlash ROM に書き込んで完成です。デバッグで使用したデバッグ用プログラムをコンパイラに添付されているファイル形式変換プログラム「ｌｍｃ３０」を用いてモトローラＳ２形式のＲＯＭ化ファイル(Ｏｋｌａｂｏ＿４.ｍｏｔ)に変換します。デバック時の状態でＲＳ２３２Ｃケーブルはそのまま接続しておきます。また、ＣＰＵボート上のＪＰ１をショートするとＦｌａｓｈ ＲＯＭへの書込みが可能となります。書込みにはフラッシュライタソフト「Ｍ１６ＣＦｌｓｈ」を起動します。使用方法は、第１０章の(1)項「デバッガを使用する為のフラッシュライタソフトの使い方」をご欄下さい。

表７は、図２９で示される「Ｍ１６Ｃ　ｆｌａｓｈ　Ｗｒｉｔｅ」ウインドウのメニューボタンの機能を表したものです。

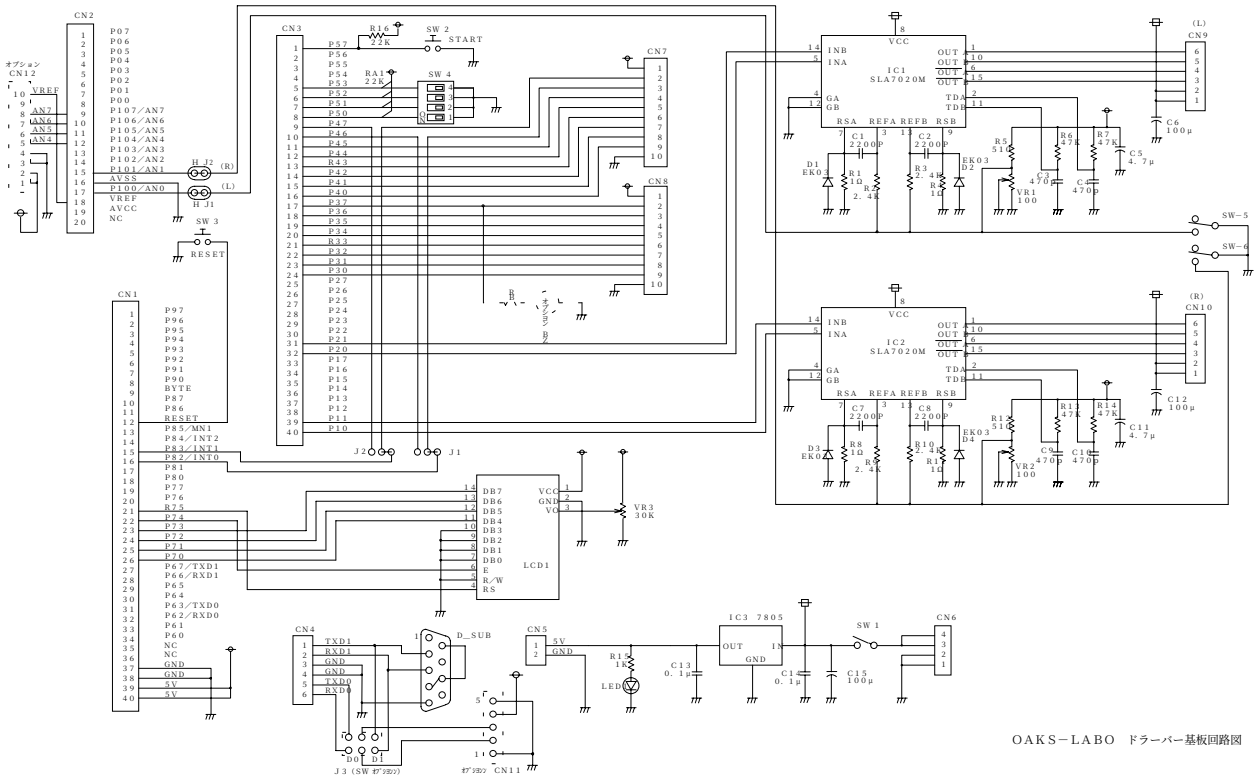
表１１ 各メニューボタンの機能

メニュー	機　　能
Load(ID)	ファイルの選択と ID チェックを行います。
Blank	指定した領域に対してブランクチェックを行います。
Read	指定したファイルの内容とを比較します。
Status	マイコンのステータスを表示します。
B.P.R	ブランクチェック、プログラム、ベリファイを順次行います。
Program	指定したファイルをフラッシュ ROM に書き込みます。
Erase	フラッシュ ROM の全領域をイレースします。
Setting	通信設定を行います。
Exit	書込みを終了します。

付　　録

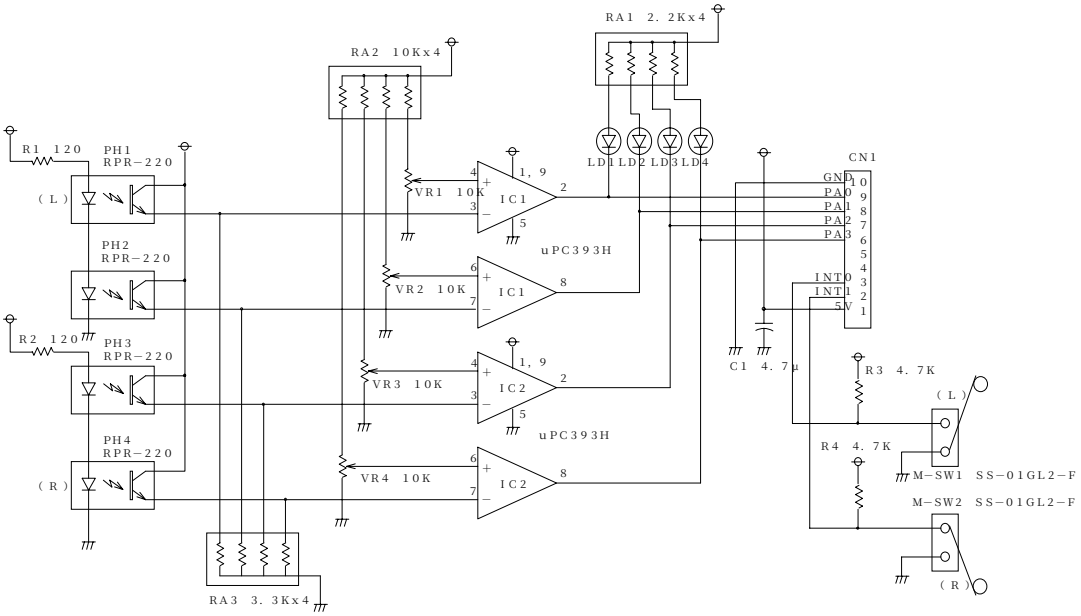
付録―1　ドライバーボード回路図

●拡大してご覧下さい。



付録―2　センサボード回路図

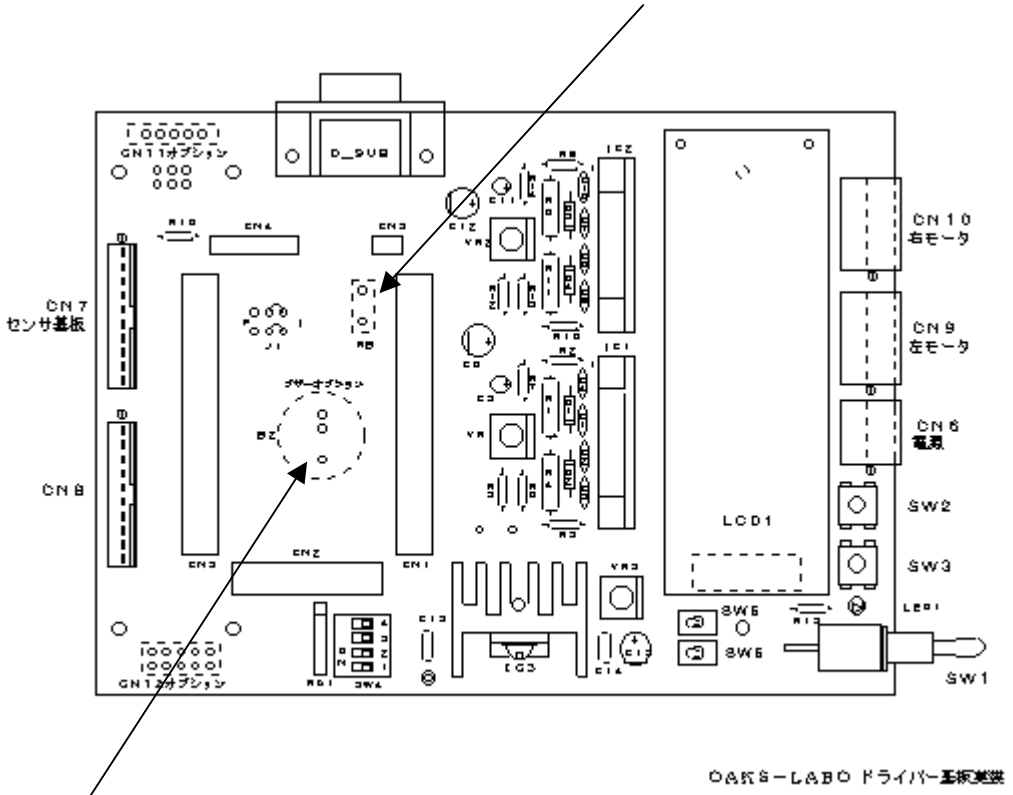
●拡大してご覧下さい。



付録－３　ドライバーボード構成図

●拡大してご覧下さい。

抵抗５００[Ω]程度を接続します。



オプション：圧電ブザーの取り付け

上図の位置に圧電ブザーと抵抗を取り付けます。圧電ブザー端子の一方に２つの穴が空いています。これは、７.５㎜タイプと１０㎜タイプの何れでも取り付けを可能にしています。デモプログラムでは、左右の接触センサに障害物が接触すると２回鳴ります。また、停止ラインで停止すると３回鳴るようにプログラムされています。

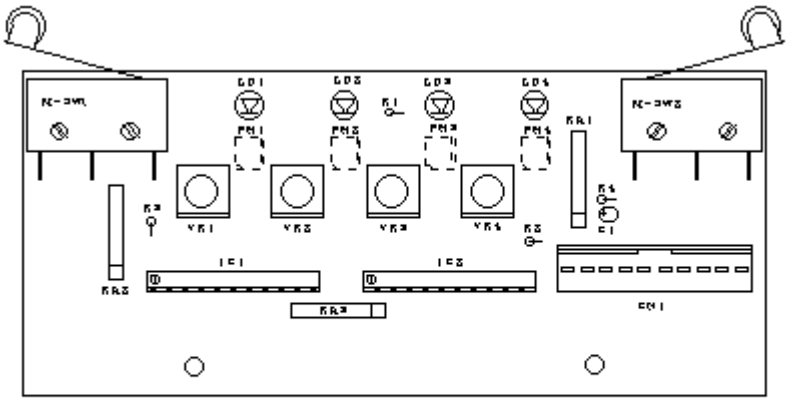
- **推奨圧電ブザー**
  - **①　１０㎜タイプ**
    - **村田製作所　ＰＫＭ１７ＥＰＰ－４００１**
    - **村田製作所　ＰＫＭ２２ＥＰＰ－４００１**
  - **②　７.５㎜タイプ**

    村田製作所　ＰＫＭ１３ＥＰＰ－４００２
- **抵抗**

  ５００[Ω]程度の抵抗を上図の図面位置に取り付けてください。

付録－４　センサボード構成図

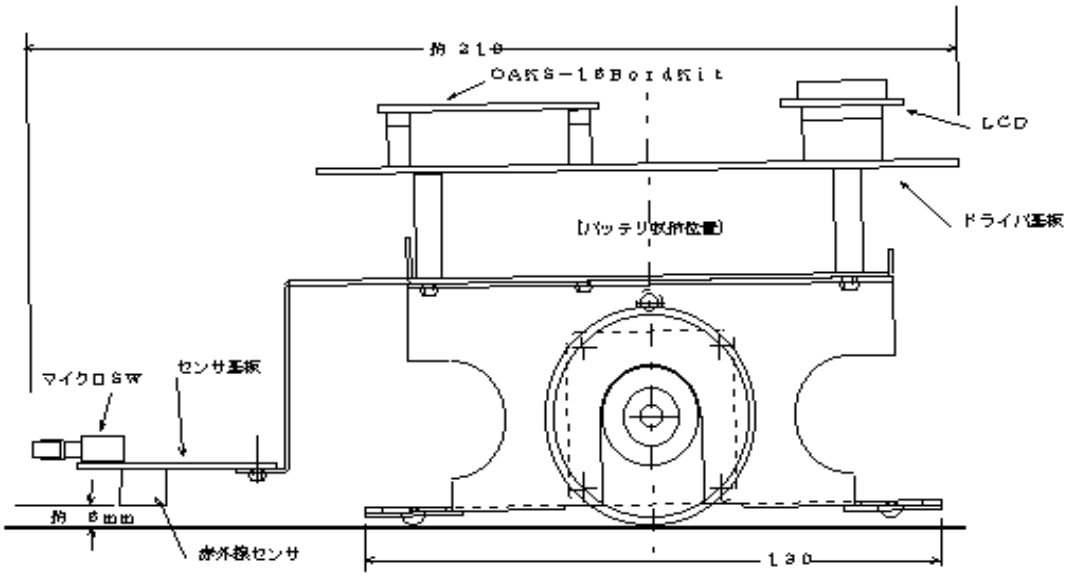
●拡大してご覧下さい。



ＯＡＫＳ－ＬＡＢＯ センサ基板図

付録―5　ＯＡＫＳ－ＬＡＢＯ外形図

●拡大してご覧下さい。



ＯＡＫＳ－ＬＡＢＯ外形図

付録―6　コネクター覧

CN１(OAKS16 CPU ボード)

1		2	
3		4	
5		6	
7		8	
9		10	
11		12	RESET
13		14	
15	右接触センサ	16	左接触センサ
17		18	
19		20	
21	P75 RS(LCD)	22	P74 E (LCD)
23	P73 DB7(LCD)	24	P72 DB6(LCD)
25	P71 DB5(LCD)	26	P70 DB4(LCD)
27	TXD1	28	RXD1
29		30	
31	TXD0	32	RXD0
33		34	
35		36	
37	GND	38	GND
39	+5V	40	+5V

CN３(OAKS16 CPU ボード)

1	P57 START	2	
3		4	
5	P53 4(SW4)	6	P52 3(SW4)
7	P51 2(SW4)	8	P50 1(SW4)
9	P47 JP2	10	P46 JP1
11	P45 4(CN7)	12	P44 5(CN7)
13	P43 6(CN7)	14	P42 7(CN7)
15	P41 8(CN7)	16	P40 9(CN7)
17	P37 ブザー	18	P36 3(CN8)
19	P35 4(CN8)	20	P34 5(CN8)
21	P33 6(CN8)	22	P32 7(CN8)
23	P31 8(CN8)	24	P30 9(CN8)
25		26	
27		28	
29		30	
31	P21 INB(左)	32	P20 INA(左)
33		34	
35		36	
37		38	
39	P11 INB(右)	40	P10 INA(右)

CN２(OAKS16 CPU ボード)

1		2	
3		4	
5		6	
7		8	
9		10	
11		12	
13		14	
15	AN1(右励磁電流)	16	AVSS
17	AN0(左励磁電流)	18	
19		20	

CN６(電源コネクタ)

1	GND
2	GND
3	+12[V]
4	+12[V]

CN７(センサコネクタ)

1	VCC　+5[V]
2	INT1 右マイクロ SW
3	INT0 左マイクロ SW
4	P45　未使用
5	P44　未使用
6	P43　右外光センサ
7	P42　右内光センサ
8	P41　左内光センサ
9	P40　左外光センサ
10	GND

CN8(未使用コネクタ)

1	VCC　+5[V]
2	P37　ブザー(OP)
3	P36　未使用
4	P35　未使用
5	P34　未使用
6	P33　未使用
7	P32　未使用
8	P31　未使用
9	P30　未使用
10	GND

CN9(右モータコネクタ)

1	12[V]電源
2	12[V]電源
3	$\overline{\text{OUT B}}$
4	OUT B
5	$\overline{\text{OUT A}}$
6	OUT A

CN10(左モータコネクタ)

1	12[V]電源
2	12[V]電源
3	$\overline{\text{OUT B}}$
4	OUT B
5	$\overline{\text{OUT A}}$
6	OUT A

CN11(オプションコネクタ)

1	GND
2	RXD
3	TXD
4	VCC　+5[V]
5	GND

CN12(オプションコネクタ)

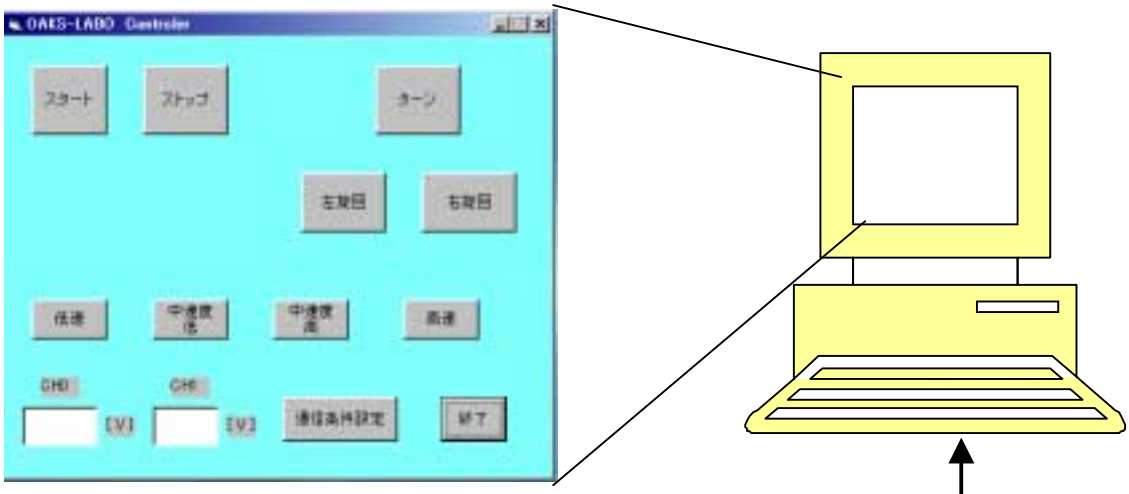
1	VCC　+5[V]
2	VCC　+5[V]
3	GND
4	GND
5	AN4　A/D-4
6	AN5　A/D-5
7	AN6　A/D-6
8	AN7　A/D-7
9	NC
10	V.REF

※　ＣＮ７を他のセンサで使用するとき、割り込み端子(INT0,INT1)を P37,P47 に変更することが可能です。この場合、J1,J2 ジャンパのＩ側をカットし、Ｐ側に接続して下さい。

※　ＣＮ８は未使用コネクタですが、P37 はオプションで圧電ブザーが使用できます。

付録－７　オプション(単方向　＆ 双方向無線ユニットボードの活用)

ＯＡＫＳ－ＬＡＢＯ制御ウインドウ例



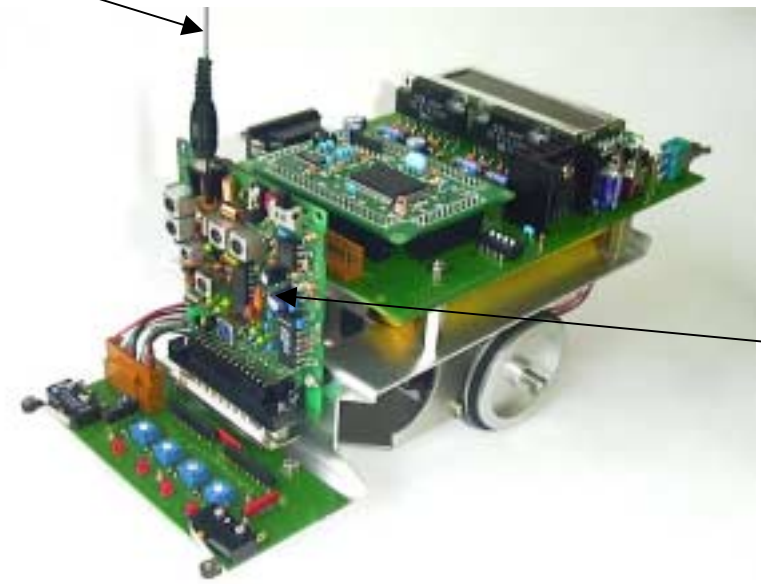
パソコン側無線ユニット

ＲＳ－２３２Ｃ
ケーブル

25P D-SUB コネクタ

無線通信

無線ユニット装着オプション



OAKS-LABO 側
無線ユニット

1.単方向無線ユニットの活用方法

単方向無線ユニットの送信ボードをパソコンのＲＳ－２３２Ｃと接続します。また、受信ボードをＯＡＫＳ-ＬＡＢＯの専用取り付け位置に取り付けます。

パソコンから次のような無線制御が可能です。

① 速度指令

低速、中速度(低)、中速度(高)、高速

② スタート指令

③ 停止指令

④ １８０度ターン指令

⑤ ９０度右ターン指令

⑥ ９０度左ターン指令

2.双方向無線ユニットの活用方法

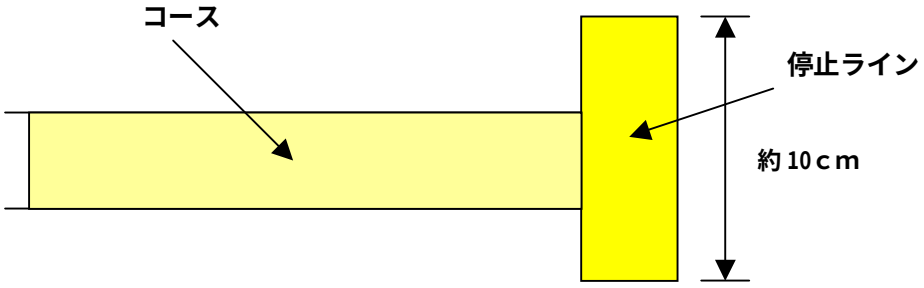
双方向無線ユニットは、お互いに周波数の異なる送信／受信ボートが２セットあります。一方のボードをパソコンのＲＳ－２３２Ｃと接続します。また、他方のボードをＯＡＫＳ-ＬＡＢＯの専用取り付け位置に取り付けます。

双方向無線ユニットを使用すると、１項の使い方の他に、リモートデバッガ「ＫＤ３０」を使用したデバッグをコードレスで行えます。ライントレースロボット等の移動体は、ＲＳ－２３２Ｃケーブルを取り付けたまま走行させせる事は困難です。このケーブルを無線に置き換える事により、デバッグがスムーズに行えます。

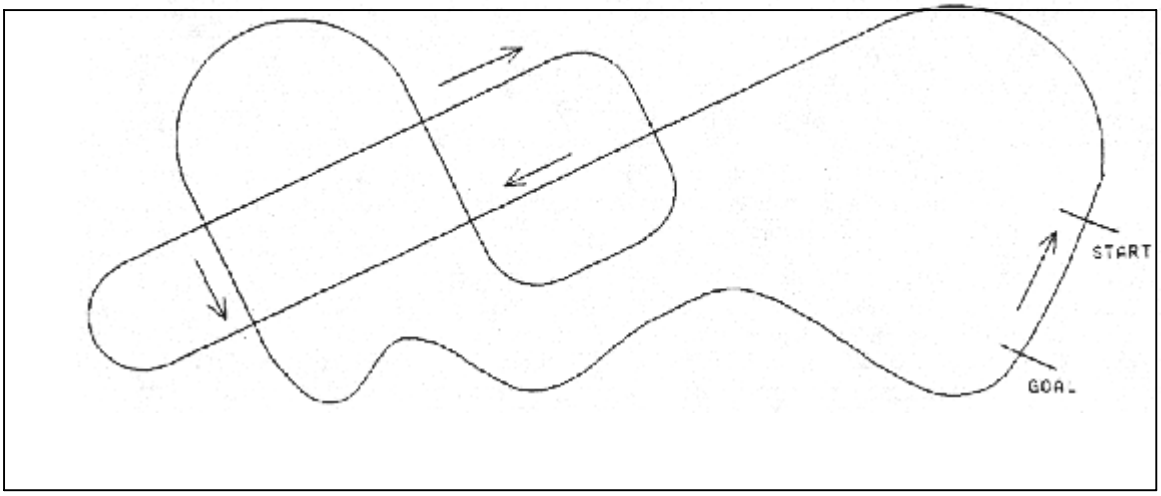
付録－８ 停止ラインの作り方

ＯＡＫＳ－ＬＡＢＯは、スタート・ストップ及びゴールマーカの検出機能はありません。ＯＡＫＳ－ＬＡＢＯでの停止ラインは、４個のラインセンサが全て検出した場合とします。

下図で示されるように、コースの終わりにコースと直角に１０ｃｍ程度のコースと同じ色のテープを貼っておきます。



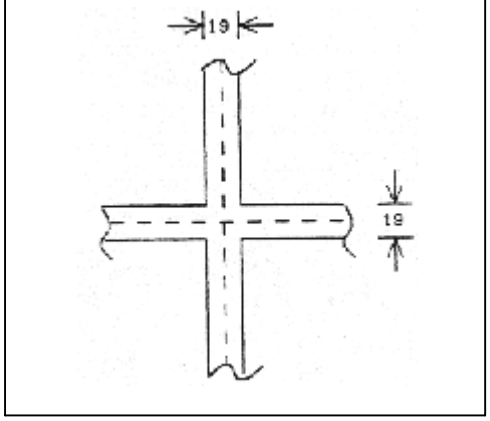
付録－９　ライントレースロボットのコース例



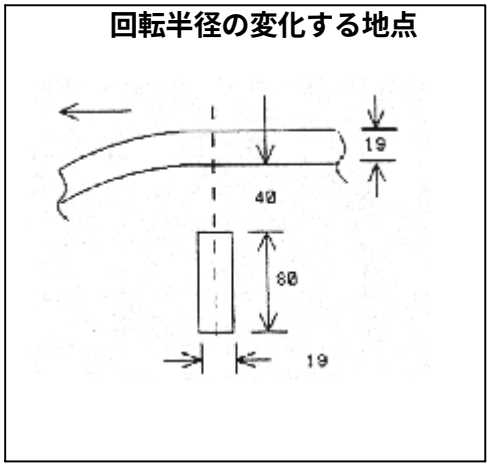
※　コースは黒地に白ラインまたは白地に黒ライン何れでも走行可能です。

※　Ｒの円弧曲率半径はＲ３０ｃｍからＲ６０ｃｍです。

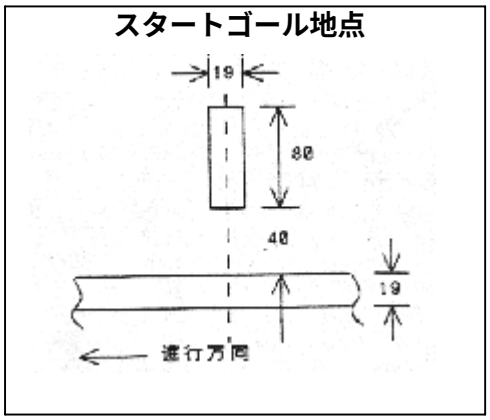
交差点



回転半径の変化する地点



スタートゴール地点



※　ＯＡＫＳ－ＬＡＢＯにはスタート、ストップ及びコーナー・マーカの検出はできません。

※　ＯＡＫＳ-ＬＡＢＯのストップ検出は、４個のセンサが全て”１”または”０”で停止します。デモプログラムでは、モードスイッチ３番を”ＯＮ”にします。

●ご注意

(1)　本書の内容の一部または全部を無断転載することは禁止されています。

(2)　本書の内容は将来予告なしに変更することがあります。

(3)　本書は、内容について万全を期して作成致しましたが、万一ご不振な点や誤り、記載漏れなどお気づきなことがありましたら、お買い上げの販売代理店にご連絡ください。

(4)　運用した結果の影響につきましては、(3)項にかかわらず責任を負いかねますのでご了承ください。

●参考文献

(1)「マイコン制御ロボットの製作」横山直隆著　シータスク発行

(2)　「Ｙ－ＲＯＢＯ取扱説明書」　サン・マイテック社

(3)　「コンパイラ　ＮＣ30ＷＡ　Ver1.0」　三菱電機

●商標

Ｗｉｎｄｏｗｓ９５／９８／ｍｅ／ＮＴは、米国マイクロソフト社の登録商標です。

「ＯＡＫＳ－ＬＡＢＯユーザーズマニュアル」

２００１年１０月２３日　第二冊発行

編集・発行　　オークス電子株式会社

〒101-0025 東京都千代田区神田佐久間町３－２１－３
第一千代田ビル３Ｆ

TEL (03)3863-1121　FAX (03)3863-1130

http://WWW.oaks-ele.com