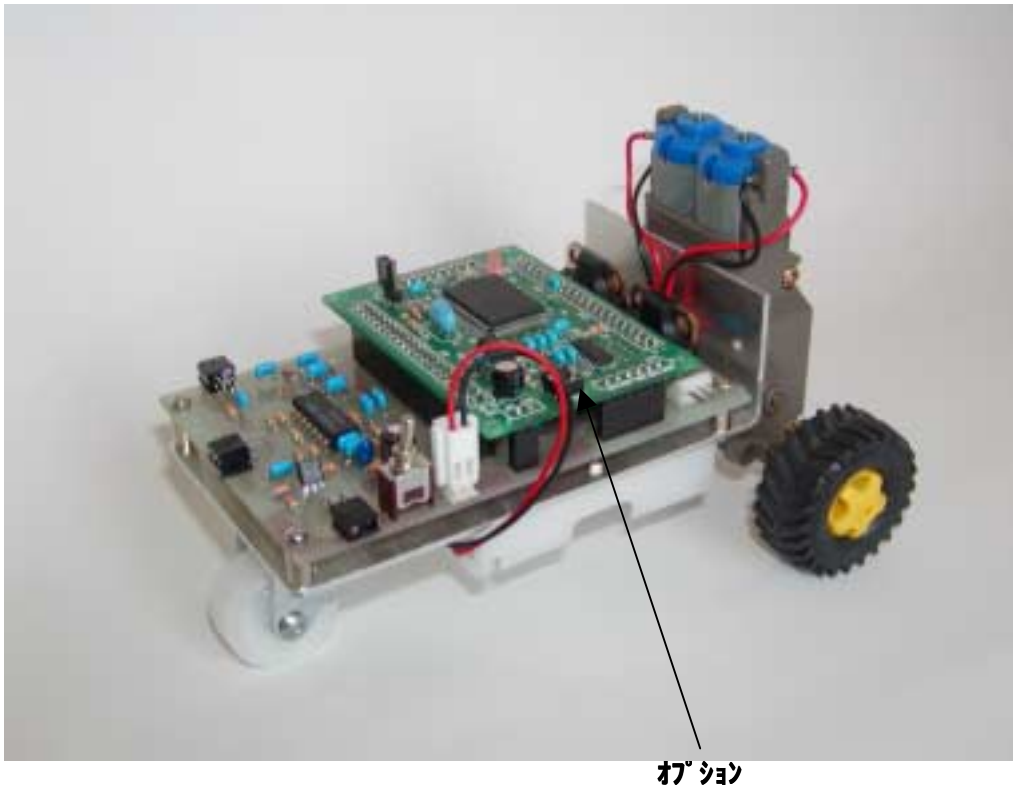
センサーロボット

OAKS16−SENSOR　LABO

ユーザーズマニュアル　Ver1.01

ＴＭ統合環境　＋　Nc30wa Ver4.00 Release２ Entry 版対応



オークス電子株式会社

目　次

1.はじめに・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・・３
2.ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯの特徴・・・・・・・・・・・・・・・３
3.ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯの仕様・・・・・・・・・・・・・・・４
4.ＤＣモータドライブ部・・・・・・・・・・・・・・・・・・・・・・・・・５
　(1) ＤＣモータドライブＩＣ「ＴＡ８４４０Ｈ」の仕様 ・・・・・・・・・・・・５
　(2) ＤＣモータドライバ「ＴＡ８４４０Ｈ」の使い方 ・・・・・・・・・・・・・６
　(3) ＣＰＵボード(ＯＡＫＳ１６)との接続 ・・・・・・・・・・・・・・・・７
5.光距離センサ部・・・・・・・・・・・・・・・・・・・・・・・・・・・８
　(1) 光距離センサ部の接続 ・・・・・・・・・・・・・・・・・・・・・・９
　(2) 障害物からの距離とＤＣ出力電圧の関係 ・・・・・・・・・・・・・・・９
　(3) 光距離センサ回路の基本構成 ・・・・・・・・・・・・・・・・・・・１０
6.ＣＰＵボード(オプション)・・・・・・・・・・・・・・・・・・・・・・１１
　(1) パラレル入出力ポート ・・・・・・・・・・・・・・・・・・・・・・１２
　(2) シリアルポート ・・・・・・・・・・・・・・・・・・・・・・・・・１３
7. ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯのしくみ ・・・・・・・・・・・１４
　(1) ＤＣモータの駆動方法 ・・・・・・・・・・・・・・・・・・・・・・１４
　(2) ＰWM制御とは ・・・・・・・・・・・・・・・・・・・・・・・・・１７
　(3) ＰWMモードの動作 ・・・・・・・・・・・・・・・・・・・・・・・１７
　(4) ＤＣモータの特性 ・・・・・・・・・・・・・・・・・・・・・・・・２１
　(5) 光距離データとＡ／Ｄ変換器の使い方 ・・・・・・・・・・・・・・・２２
8. 添付ファイルの構成と使い方 ・・・・・・・・・・・・・・・・・・・・２６
　(1) ＤＣモータとギャの使用上の留意点 ・・・・・・・・・・・・・・・・２６
　(2) サンプル「Ｒｏｂｏ＿１」の使い方 ・・・・・・・・・・・・・・・・２６
　(3) サンプル「Ｒｏｂｏ＿２」の使い方 ・・・・・・・・・・・・・・・・２７
　(4) サンプル「Ｒｏｂｏ＿３」の使い方 ・・・・・・・・・・・・・・・・２８
　(5) 距離の検出方法 ・・・・・・・・・・・・・・・・・・・・・・・・・３０
9. フラッシュＲＯＭへの書き込み方法 ・・・・・・・・・・・・・・・・・３１
　(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・・・・・３２
　(2)デバッガの起動方法・・・・・・・・・・・・・・・・・・・・・・・・３４
10.ＴＭ統合化開発環境での使い方 ・・・・・・・・・・・・・・・・・・・３６
　(1)エントリー版（新バージョン）の仕様につい ・・・・・・・・・・・・３７
　(2)ＴＭ統合化開発環境の使い方 ・・・・・・・・・・・・・・・・・・・３８
11.プログラムの作成手順・・・・・・・・・・・・・・・・・・・・・・・・３８
　(1)プログラムの実行手順 ・・・・・・・・・・・・・・・・・・・・・・３９
　(2)コンパイル方法 ・・・・・・・・・・・・・・・・・・・・・・・・・４０
　(3)コンパイル用バッチファイル例 ・・・・・・・・・・・・・・・・・・４３

(4)I／0 ポートの記述例　・・・・・・・・・・・・・・・・・・・・・・・・４３
(5)割り込み処理 ・・・・・・・・・・・・・・・・・・・・・・・・・・４４
(6)割り込みベクタテーブルへの登録 ・・・・・・・・・・・・・・・・・４５
12.Ｆｌａｓｈ　ＲＯＭへの書き込み ・・・・・・・・・・・・・・・・・・４７

付　録

付録 ― 1　ドライバボード回路図 ・・・・・・・・・・・・・・・・・・・４９
付録 ― 2　ドライバボード構成図 ・・・・・・・・・・・・・・・・・・・４９
付録 ― 3　ＯＡＫＳ１６－１６ＳＥＮＳＯＲ　ＬＡＢＯ外形図 ・・・・５０
付録 ― 4　モータ配線図 ・・・・・・・・・・・・・・・・・・・・・・５０
付録 ― 5　シャシー組立て図 ・・・・・・・・・・・・・・・・・・・・５１
付録 ― 6 ＲＳ－２３２Ｃ結線図・・・・・・・・・・・・・・・・・・・５１
付録 ― 7　部品表 ・・・・・・・・・・・・・・・・・・・・・・・・・５２

安全上の注意

１）乾電池以外の外部電源を接続中に変なにおいがする、発熱するなどの異常状態がみられる場合はすぐには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。また、外部電源は６[V]またはそれ以下で必ずご使用下さい。

２）万一、乾電池値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）乾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）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下さい。

ご使用上の注意

１）　使用しているＤＣモータは、ＰＷＭ制御によって走行速度の制御を行いますが、乾電池の消耗によっても走行速度が遅くなります。したがって、電池の消耗によって設計通りの動作ができなくなります。十分にご注意下さい。

２）　使用されているツインギャーボックスは、構造上の問題から左右のモータのトルクに差が出ます。設定したＰＷＭデータが左右同じであっても直線上を走行せず、何れかの方向にカーブしますのでご注意ください。

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。

センサーロボット「ＯＡＳＫＳ１６－Ｓｅｎｓｏｒ　Ｌａｂｏ」は、前方及び左右の障害物を光距離センサーによって捕らえ、それぞれの障害物を避けて走行する自走ロボットです。

初心者でも経験者でも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する為の参考となります。

2.0AKS16-SENSOR 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ボートに採用した学習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ロボットがＯＡＳＫＳ１６－Ｓｅｎｓｏｒ　Ｌａｂｏです。ＣＰＵにはＭ１６Ｃ／６２シリーズのＭ３０６２０ＦＣＡＦＰ(ルネサステクノロジ社製)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」(オプション)がコネクタを介して接続できる仕様になっています。次のような多くの特徴を持っています。

- オプションのＣＰＵボート「ＯＡＫＳ１６」を用いると、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使用できます。
- ＤＣモータ駆動回路にはＰＷＭ制御ＩＣを使用し、速度制御を可能にしています。
- 光距離センサは、障害物までの距離に対応したＤＣ電圧が得られます。この電圧をＡ／Ｄコンバータに取り込みＣＰＵボートで障害部まで距離を得ます。

3.OAKS16-SENSOR LABO の仕様

ＯＡＫＳ１６－ＳＥＮＳＯＲ　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(オプション)がコネクタを介して取り外し可能となっています。

表１　ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯの仕様

機　構	内　　容
モータ／ギヤー	宮模型　Ｎｏ９７ ﾂｲﾝﾓｰﾀ･ｷﾞﾔｰﾎﾞｯｸ
タイヤ	田宮模型　Ｎｏ１０１　ﾄﾗｯｸﾀｲﾔｾｯﾄ(２個使用)
モータドライブ	東芝　ＴＡ８４４０Ｈ　２個
距離センサ	ロームＲＰＲ-２２０　３個
シャーシ	アルミニウム板一体型、キャスタ１個付き
電源	６[V] ／３００[mA](電池ＭＵ－３×４は付属しません。)
マニュアル	ユーザーズマニュアル(ＰＤＦファイル形式)
重量サイズ	２５０ｇ(電池無し) 横：１１０mm　長さ：１４５mm　高さ：９０mm

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。商品に添付されているＣＤ－Ｒにはオプションのｃｐｕボードで動作するサンプルデモプログラムが入っています。デバッグ用として、汎用のＲＳ－２３２Ｃケーブルを使用する場合には、付録―６のＲＳ－２３２Ｃ結線図に従って変換ケーブルを作成して下さい。

表２オプション仕様

オプション	構　　成	価　　格
専用 RS-232C ケーブル OAKS16-PLC 用	９ｐｉｎ－３ｐｉｎケーブル D-SUB 9 ﾒｽ/H3P-SHF-AA L=1.8m	￥３，０００
ＣＰＵボード	ＯＡＫＳ１６マイコンボード ルネサステクノロジ社製 30620FCAFP 使用 フラッシュ ROM 128KB、RAM 10KB 内臓 ・コンパイラ付き （ＮＣ３０ＷＡ V4.00 Release2 Entry 版） ・ 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	￥５，４００

４.ＤＣモータドライブ部

モータドライブの構成図を図１に示します。東芝社製のＰＷＭ(Pulse Width Modulation)制御によるＤＣモータ駆動回路ＩＣであるＴＡ８４４０Ｈを使用し、ＯＡＫＳ１６－ＳＥＮＳＯＲ ＬＡＢＯの前進走行や後進走行を実現します。

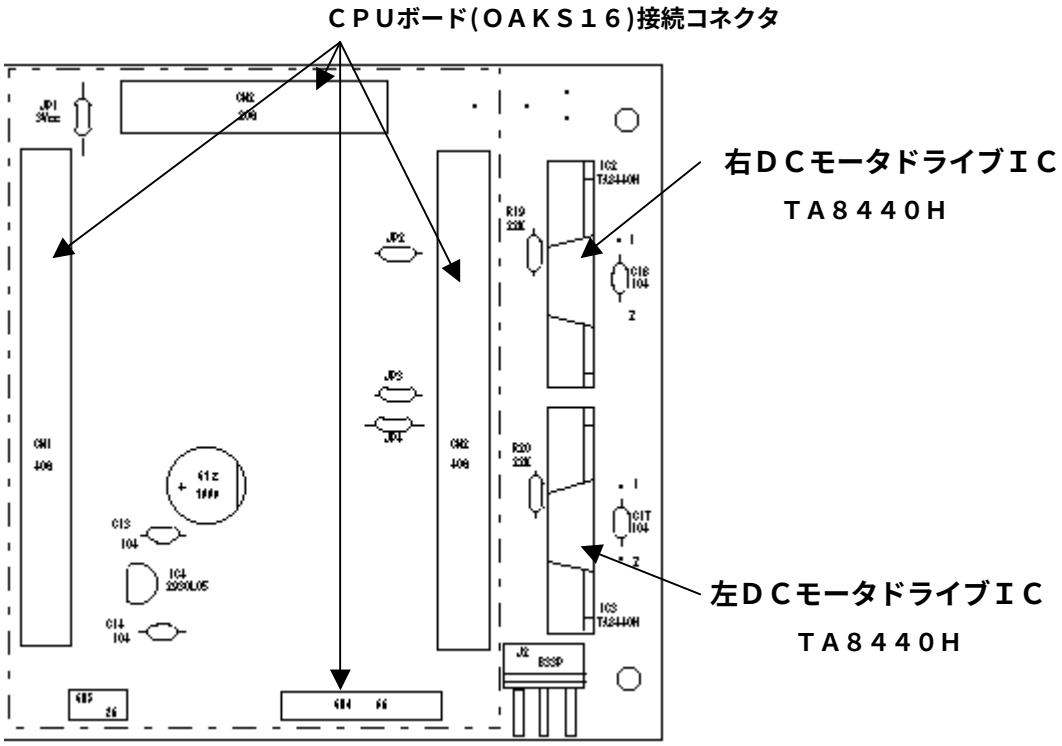


図１　モータドライブ部の構成図

(1)、ＤＣモータドライブＩＣ「ＴＡ８４４０Ｈ」の仕様

ＤＣモータドライブＩＣのＴＡ８４４０Ｈ（東芝製）は、ブラシ付きモータの正・逆転切り替え用のフル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。また、モータ駆動部とコントロール部は独立しており、ステッピングモータドライバとしても使用できます。

図２は、ＴＡ８４４０ＨとＤＣモータの接続を示したものです。ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯでは、１番端子(Ｅｎａｂｌｅ)を６番端子(制御系電源)と接続し、常にＨレベルを与えています。１番端子(Ｅｎａｂｌｅ)と２番端子(回転方向制御)はＩＣ内部でプルダウン抵抗(１００ＫΩ)が内蔵されています。これらの端子をマイコンボートのＩ/Ｏポート端子と接続した場合、プログラムによってＩ/Ｏポートをプルアップ(Pull Up)に設定すると、各ポートのプルアップ抵抗は５０ＫΩ(平均値)となりＴＡ８４４０Ｈ側のプルダウン機能が働かずにプルアップされ突然モータが回りだすなどの誤動作をする可能性があります。そこで、１番端子(Ｅｎａｂｌｅ)は常にＨレベルを与えています。

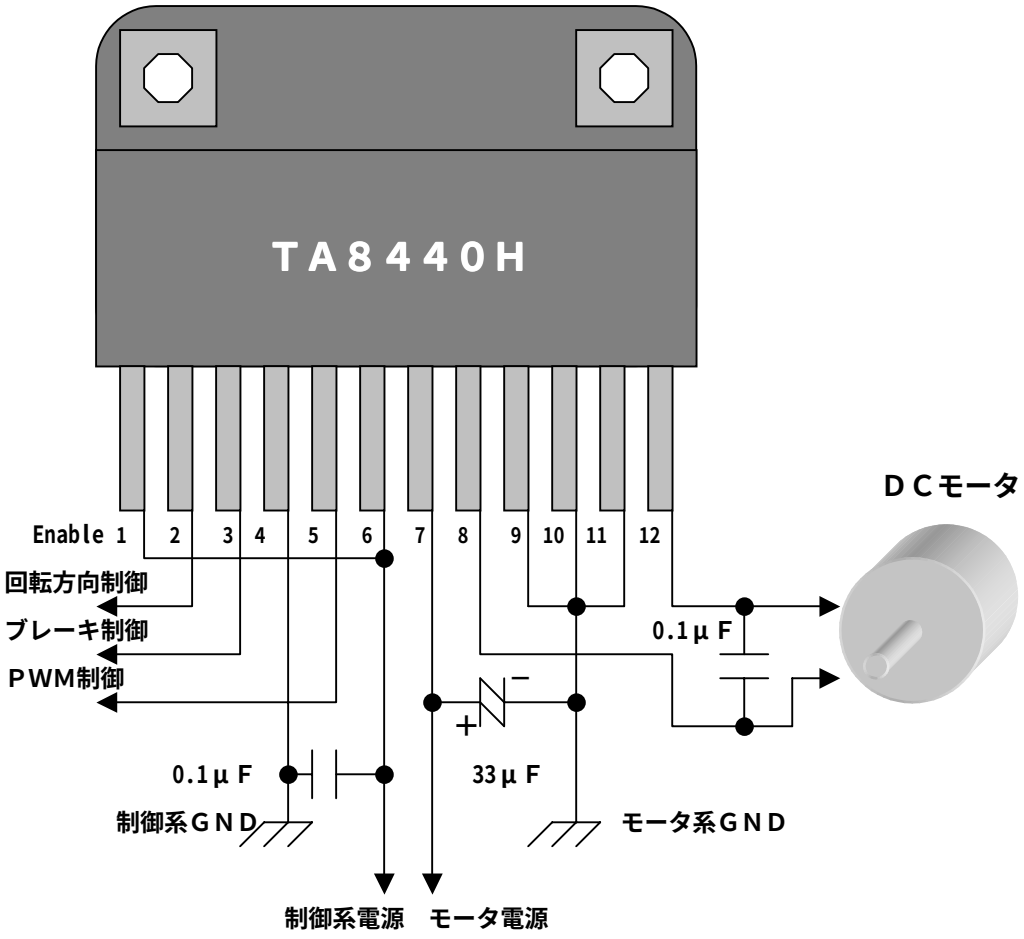


図２　ＴＡ８４４０Ｈとモータの接続

(2)、ＤＣモータドライバ「ＴＡ８４４０Ｈ」の使い方

ＥＮＢＬＥ端子(１番端子)が常に‘１’の状態で表３で示されるように、回転方向制御端子(２番端子)に‘０’、ブレーキ制御端子(３番端子)に‘１’を与え、ＰＷＭ制御端子(５番端子)が‘０’になるとＤＣモータは正転します。また、回転方向制御端子(２番端子)に‘１’、ブレーキ制御端子(３番端子)に‘１’を与え、ＰＷＭ制御端子(５番端子)が‘０’になるとＤＣモータは逆転します。

モータを停止するには、回転方向制御端子(２番端子)の状態に関係なく、ブレーキ制御端子(３番端子)に‘１’を与え、ＰＷＭ制御端子(５番端子)を‘１’にするとＤＣモータは停止します。

表３　ＴＡ８４４０Ｈの真理値表

入　力			出　力		
回転方向制御	ブレーキ制御	PWM制御	ＯＵＴＡ	ＯＵＴ$\overline{A}$	モータ動作
0	1	0	0	1	正転
1	1	0	1	0	逆転
0／1	1	1	∞	∞	ストップ
0／1	0	0／1	0	0	ブレーキ
0	1	1	0	∞	ＣＨＯＰ
1	1	1	∞	0	ＣＨＯＰ

(注)　∞：ハイインピーダンス

(3)、ＣＰＵボート(ＯＡＫＳ１６)との接続

図３はＣＰＵボート「ＯＡＫＳ１６」とＤＣモータドライバ「ＴＡ８４４０Ｈ」の接続を示したものです。右モータの回転方向制御は、ＣＰＵボートのＩ/ＯポートＰ１０とＰ１１を図３に示されるように接続します。また、左モータの回転方向制御は、ＣＰＵボートのＩ/ＯポートＰ３０とＰ３１を図３に示されるように接続します。

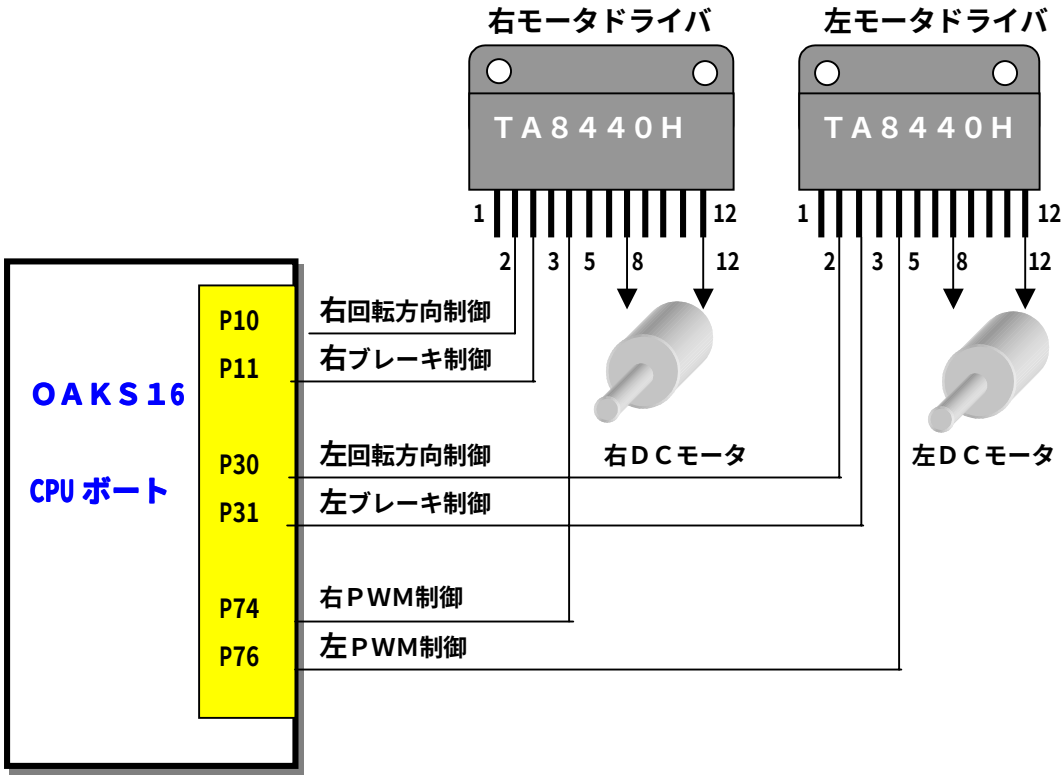


図３　ＣＰＵボートとＴＡ８４４０Ｈの接続

右モータ用のＰＷＭ制御は、Ｉ/ＯポートのＰ７４端子がＰＷＭ出力端子(ＴＡ２ｏｕｔ)と兼用端子で、プログラムによって使用を切替えます。また、左モータ用のＰＷＭ制御は、Ｉ/ＯポートのＰ７６端子がＰＷＭ出力端子(ＴＡ３ｏｕｔ)と兼用端子で、プログラムによって使用を切替えます。

5.光距離センサ部

光距離センサ部の構成図を図４に示します。光距離センサ部は図１で示されるＤＣモータドライブ部と一体化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。光距離センサーにはローム社のフォトレフレフクタＲＰＲ－２２０を３個使用します。交流方式により、障害部からの反射出力をＯＰアンプによる増幅し、この出力をダイオードで検波(整流)します。得られた直流電圧をマイコンのＡ／Ｄコンバータから取り込み障害物からの距離を得ています。

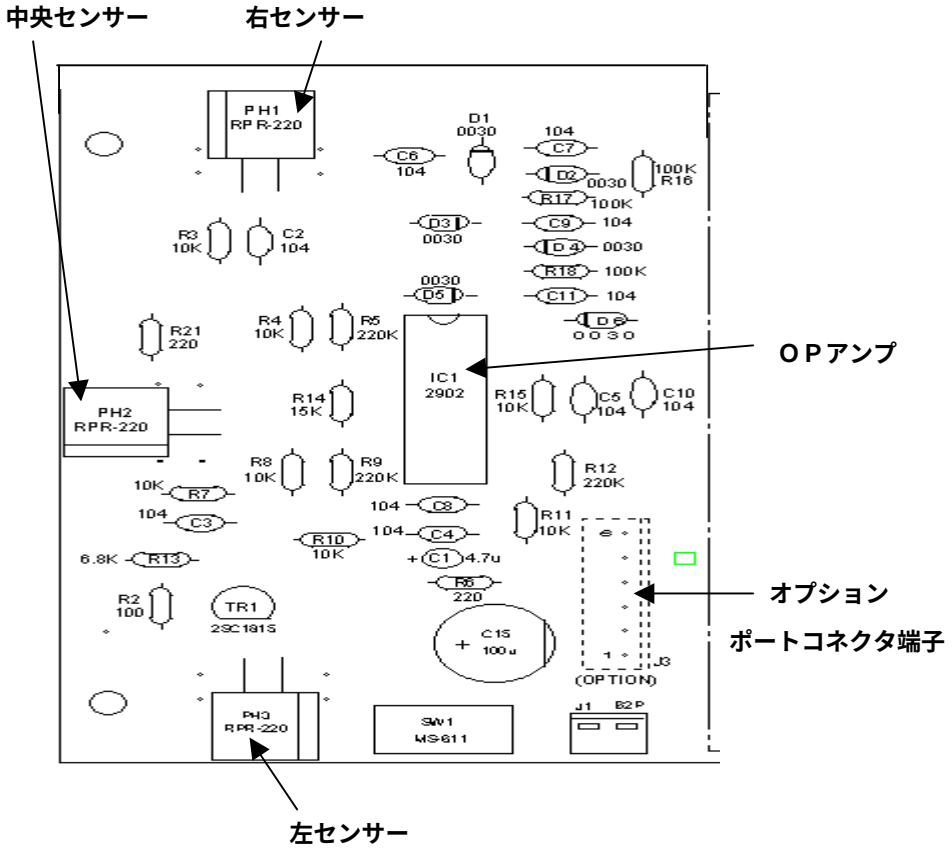


図４　光距離センサ部の構成図

（１）、光距離センサー部の接続

光距離センサー部に取り付けられている３個のフォトレフレフクタからの距離情報は、図５に示されるように、ＯＰアンプと検波回路を介してＣＰＵボードのＡ／Ｄ変換器端子ＡＮ０、ＡＮ１、ＡＮ２にそれぞれ入力します。

また、フォトレフレフクタを距離センサとして使用する為に、Ｉ／０ポートＰ０４端子をトランジスタのベースに接続します。この端子からプログラムによってソフト的に８００Ｈｚのパルスを作りフォトレフレフクタＲＰＲ－２２０に与えています。

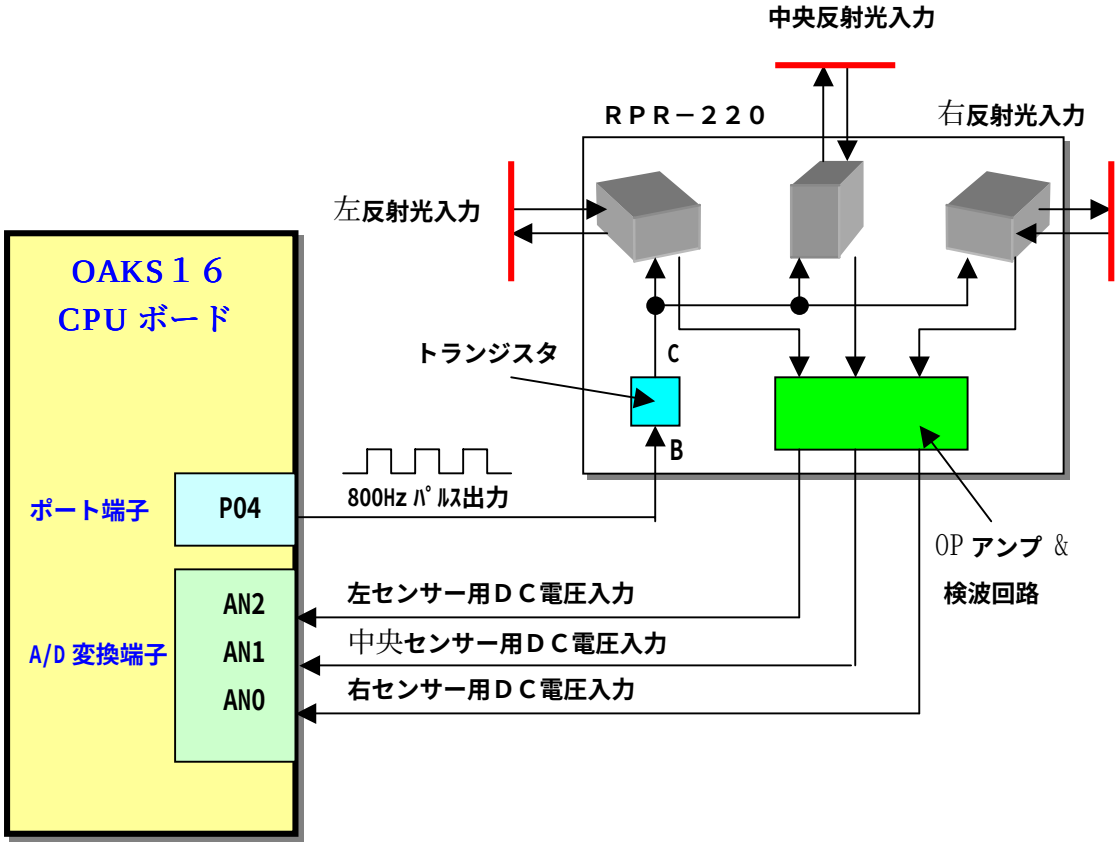


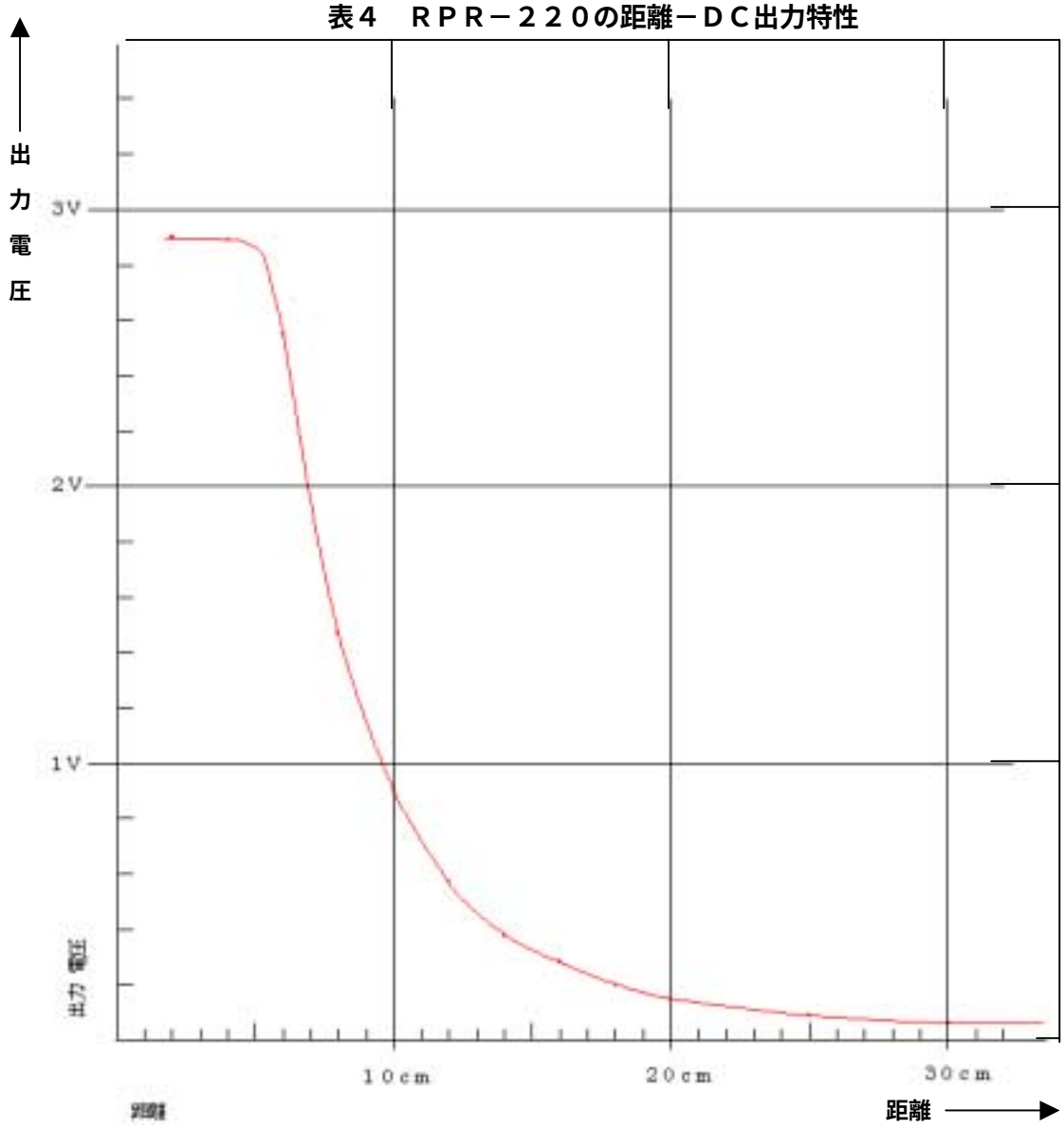
図５　光距離センサ部の接続

(２)、障害物からの距離とＤＣ出力電圧の関係

表４は、フォトレフレフクタＲＰＲ－２２０を距離センサーとして使用した場合の障害物までの距離とその時に得られるＤＣ電圧出力の関係を表した特性図です。

連続して正しい距離を検出するには、距離－ＤＣ出力電圧特性からできる限り直線部分を使用します。表４では、距離が６ｃｍから１２ｃｍの間を使用するとその距離に比例したＤＣ出力電圧が得られます。

表４　ＲＰＲ－２２０の距離－ＤＣ出力特性



(３)、光距離センサー回路の基本構成

図６は光距離センサー部の基本構成回路です。フォトレフレフクタＲＰＲ－２２０からの発射光は、８００Ｈｚのパルス波として発射されます。このパルス波は、Ｉ／０ポートＰ０４端子からトランジスタ２ＳＣ１８５のベースに与えられます。コレクタ側には発光ダイオードのカソードに接続すると、トランジスタのＯＮ／ＯＦＦに対応して光が発射されます。８００Ｈｚのパルス列は、プログラムによってソフト的に作り、Ｉ／０ポートＰ０４端子から出力します。

障害物からの反射光は、フォトトランジスタから発光ダイオードのパルス波に同期して受信されます。このパルス波をＯＰアンプＮＪＭ２９０２で増幅し、ダイオードＤ１とＤ

２及びコンデンサ０．１μと抵抗１００Ｋで構成される検波回路で検波し、直流出力電圧を得ています。

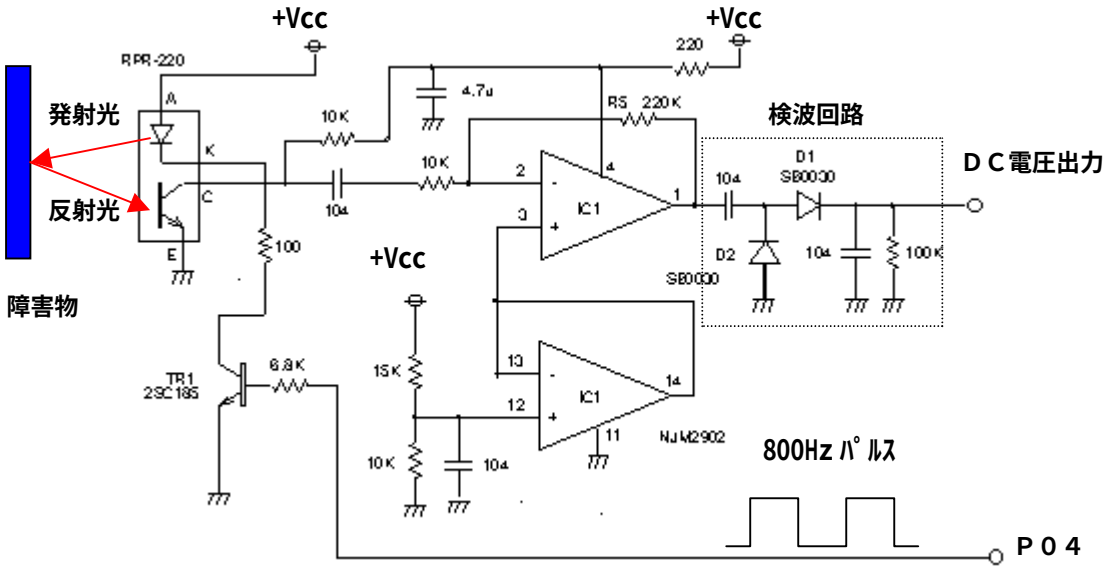


図６　光距離センサーの基本回路構成

6.ＣＰＵボード(オプション)

図７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、Ｅ００００ｈ～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Ｅ００００ｈ～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。

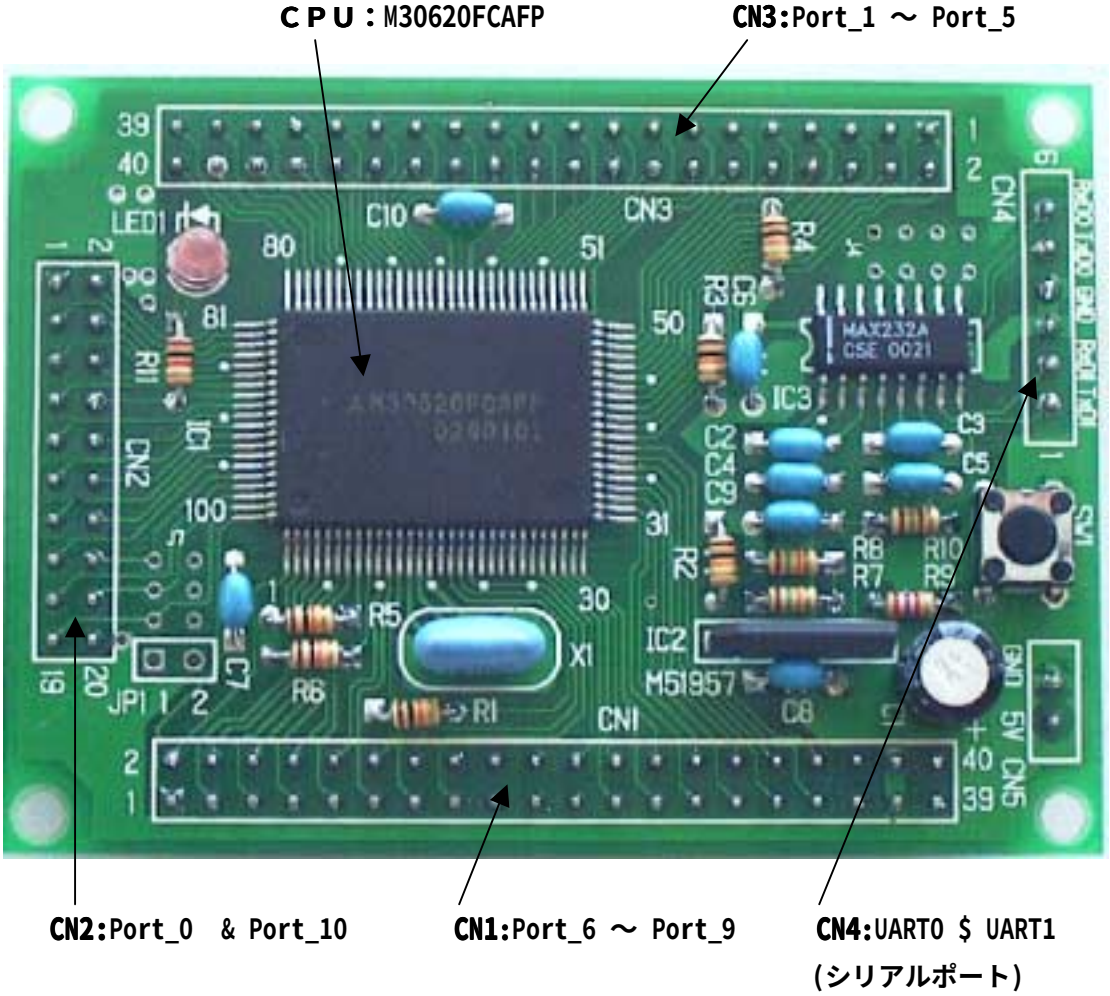


図７ ＯＡＫＳ１６ ＣＰＵボード

(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変換器の入力端子、Ｄ／Ａ変換器の出力端子などが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５はＯＡＫＳ１６－ＳＥＮＳＯＲ 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３(Ｐｏｒｔ＿３)のビットＰ３０とビットＰ３１は、左側モータの駆動用として使用します。

ポート７(Ｐｏｒｔ＿７)のビットＰ７４はＴＡ２ｏｕｔ端子として使用し、右モータ用のＰＷＭを出力します。ビットＰ７６はＴＡ４ｏｕｔ端子として使用し、左モータ用のＰＷＭを出力します。

また、ポート０(Ｐｏｒｔ＿０)のビットＰ０４は、８００Ｈｚのパルス波の出力用として使用します。

表５　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	右モータ回転方向制御
	Ｐ１１	右モータブレーキ制御
Port_3	Ｐ１０	左モータ回転方向制御
	Ｐ１１	左モータブレーキ制御
Port_0	Ｐ０４	８００Ｈｚパルス波出力
Port_10	ＡＮ０	右光距離センサＤＣ電圧入力
	ＡＮ１	中央光距離センサＤＣ電圧入力
	ＡＮ２	左光距離センサＤＣ電圧入力
Port_7	Ｐ７４	右モータＰＷＭ出力
	Ｐ７６	左モータＰＷＭ出力

(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とフラッシュライタソフトによるフラッシュＲＯＭへの書き込みで使用します。また、チャンネル０（Ｔｘ０，Ｒｘ０）は、未使用チャンネルです。オプションのＲＳ２３２Ｃ無線ユニットを使用すると無線による遠隔操が可能です。

チャンネル１（Ｔｘ１，Ｒｘ１）は、図８で示されるようにＪ２(ポストサイド型)は日圧(ＢＳ３Ｐ－ＳＨＦ－１ＡＡ)コネクタが取り付けられています。パソコンとの接続は、オプションのＯＡＫＳ１６－ＰＬＣ用９ｐｉｎ－３ｐｉｎケーブル(￥３，０００)を使用するか、付録－６で示される接続ケーブルを自作してご使用下さい。

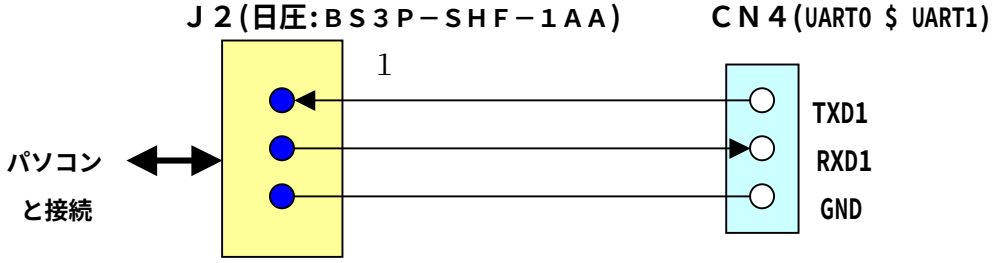


図８　ＲＳ－２３２Ｃの接続

7.ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯのしくみ

ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯには２個のＤＣモータを使用しています。ＤＣモータは、ＤＣモータドライバＴＡ８４４０Ｈを用いてＰＷＭ制御によりモータの速度制御が簡単に行えます。また、光距離センサーによって障害物まで距離を検出し、障害物を避けながら走行できます。

(1)、ＤＣモータの駆動方法

図１０は、ＣＰＵボードとＤＣモータドライバ及びＤＣモータの接続を表したものです。右モータは、Ｉ／Ｏポート１(Ｐｏｒｔ＿１)のビットＰ１０で回転方向制御、ビットＰ１１でブレーキ制御を行います。また、Ｉ／Ｏポート７(Ｐｏｒｔ＿７)のビットＰ７４をＴＡ２ｏｕｔ使用でＰＷＭを出力し、速度制御を行います。

左モータは、Ｉ／Ｏポート３(Ｐｏｒｔ＿３)のビットＰ３０で回転方向制御、ビットＰ３１でブレーキ制御を行います。また、Ｉ／Ｏポート７(Ｐｏｒｔ＿７)のビットＰ７６をＴＡ３ｏｕｔ使用でＰＷＭを出力し、速度制御を行います。

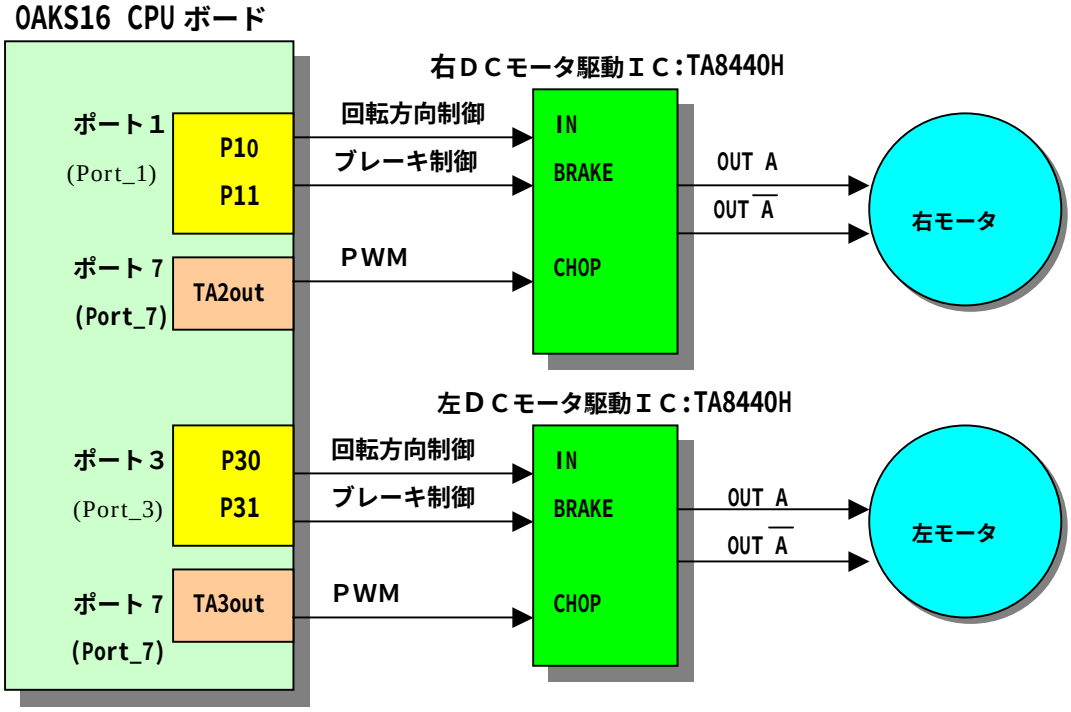


図１０　ＣＰＵボードとＤＣモータドライバとＤＣモータの接続

表６は、ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯを前進走行する為の真理値表です。左右のモータはお互いに向かい合って取り付けてあります。従って、表６に示されるように、右モータを正転させ、左モータを逆転させる事により前進走行が可能になります。

表６　前進走行真理値表

	入　力			モータ動作
	ブレーキ	回転方向	１６進数	
右モータ	Ｐ１１	Ｐ１０	０２ｈ	正転
	１	０		
左モータ	Ｐ３１	Ｐ３０	０３ｈ	逆転
	1	1		

表７は、ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯを後進走行する為の真理値表です。表７に示されるように、右モータを逆転させ、左モータを正転させる事により後進走行が可能になります。

表７　後進走行真理値表

	入　力			モータ動作
	ブレーキ	回転方向	１６進数	
右モータ	Ｐ１１	Ｐ１０	０３ｈ	逆転
	１	１		
左モータ	Ｐ３１	Ｐ３０	０２ｈ	正転
	1	0		

ＣＰＵボートのポート１（Ｐｏｒｔ＿1）のＰ１０から‘０’、Ｐ１１から‘１’を出力します。また、ポート３（Ｐｏｒｔ＿3）のＰ３０から‘１’、Ｐ３１から‘１’を出力すると、ロボットは前進走行します。図１１は、前進走行のプログラムを記述したものです。

前進走行の動作を記述すると次のようになります。

```
port_1 = 0x02;
port_3 = 0x03;
```

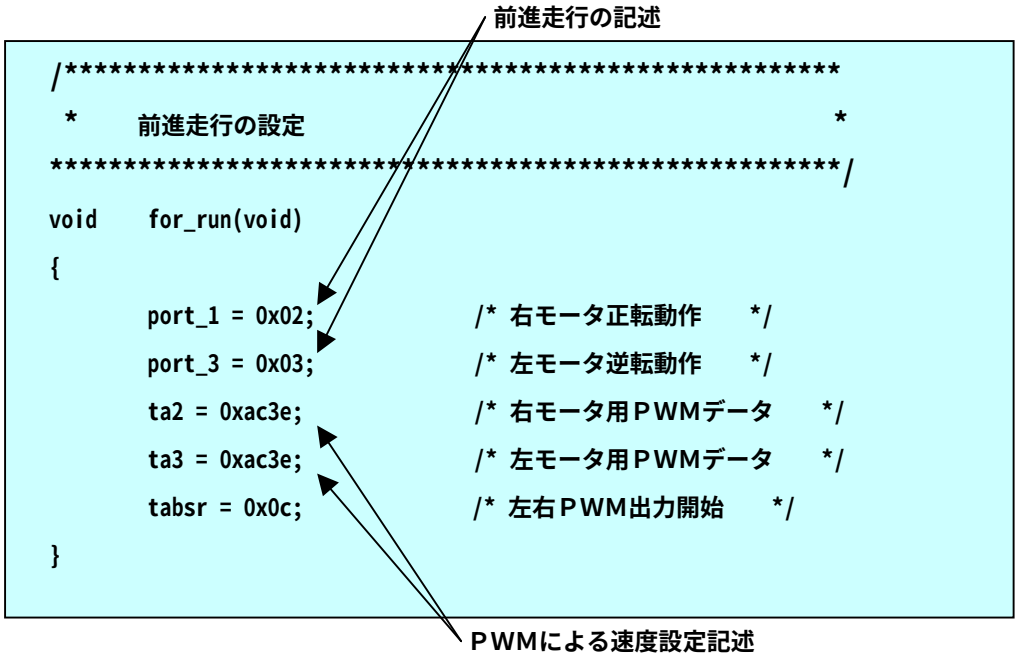


図１１　前進走行の記述例

また、ＣＰＵボートのポート１（Ｐｏｒｔ＿１）のＰ１０から‘１’、Ｐ１１から‘１’を出力します。また、ポート３（Ｐｏｒｔ＿３）のＰ３０から‘０’、Ｐ３１から‘１’を出力すると、ロボットは後進走行します。図１２は、後進走行のプログラムを記述したものです。後進走行の動作を記述すると次のようになります。

```
ｐｏｒｔ＿１　＝　０ｘ０３；
ｐｏｒｔ＿３　＝　０ｘ０２；
```

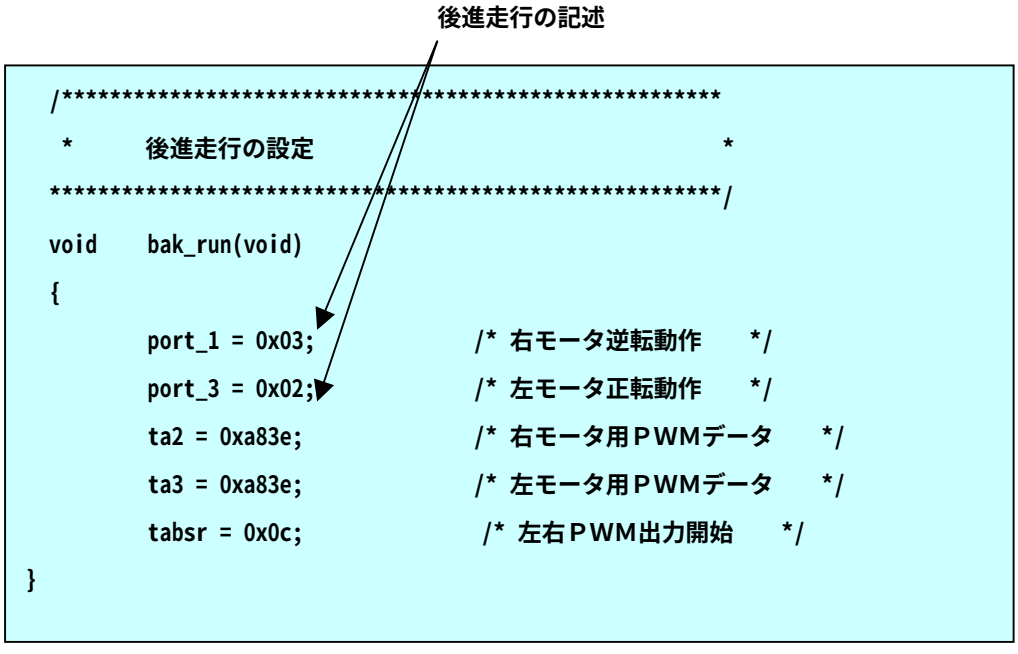


図１２　後進走行の記述例

(２)、PWM制御とは

PWM（Ｐｕｌｓｅ　Ｗｉｄｔｈ　Ｍｏｄｕｌａｔｉｏｎ：パルス幅変調）の原理を説明します。PWM信号は図１３で示されるような一定周期の信号となっていますが、(ａ)図と（ｂ）図を比べると、それぞれの信号の‘Ｈ’レベルの幅と‘Ｌ’レベルの幅が異なっています。

例えば、Ｈ’レベルの区間でモータが回転し、‘Ｌ’レベルの区間でモータが停止するとすれば、平均の回転する為のエネルギーは、‘Ｈ’レベル区間の長い(デューティの大きい)(ｂ)図のほうが大きくなります。

この信号の繰り返し周期が、モータの回転数より十分早い速さ(高い周波数)であれば、モータの回転エネルギーとしては平均電力で考えると(ｂ)図のほうが高速回転することになります。このような事を利用して、‘Ｈ’レベル区間の幅を制御することで、平均電力を制御してモータの速度を制御することができます。

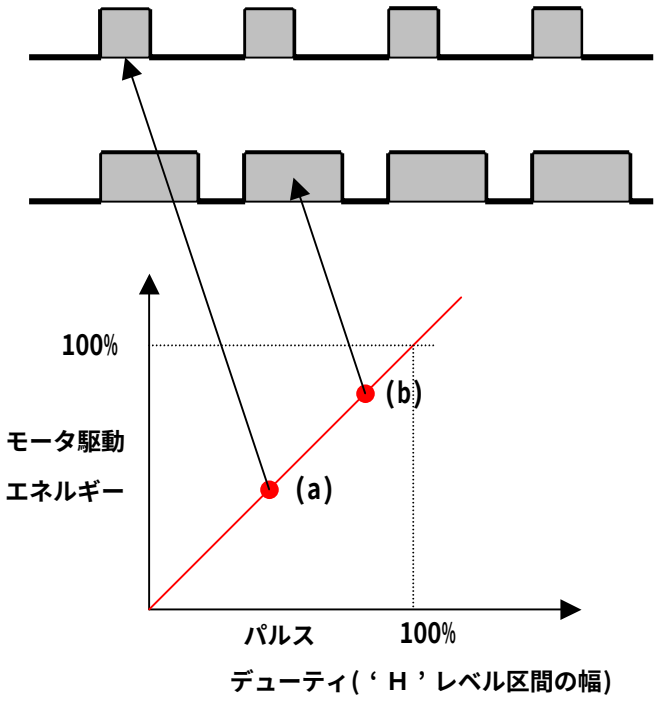
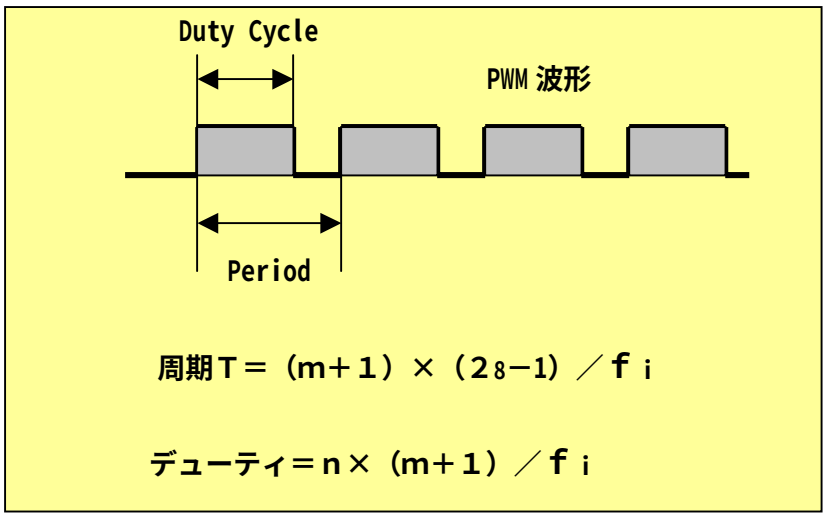


図１３　PWM制御の原理

(３)、PWMモードの動作

ＯＡＫＳ１６ ＣＰＵボードには、タイマＡ動作の８ビットＰＷＭモード選択ができます。このモードでは、図１４で示されるように‘Ｈ’レベル区間の長い(デューティサイクル：Ｄｕｔｙ　Ｃｙｃｌｅ）と繰り返し周期(Ｐｅｒｉｏｄ)は、次式で表されます。



上式より、mはプリスケラー値(分周値)で００ｈ～ＦＥｈの範囲で設定が可能です。また、ｎはタイマ値で００ｈ～ＦＥｈの範囲で設定が可能です。また、ｆｉはカウントソースの周波数を表し、ＣＰＵボードのクロック周波数１６ＭＨｚとなります。

ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯでは周期Ｔを１ｍｓに設定し、上式からmの値をを計算します。

$$m = (f_i \times T / 256 - 1) - 1$$

６２（＝３Ｅｈ）

また、ｎの値は、プログラムによってモータの速度を見ながらその値を決定します。ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯでは、Ａ０ｈからＥ０ｈの範囲でモータの速度を見ながら選びます。例えばｎ＝Ｂ０ｈ（＝１７６）を用いた場合のデューティは上式より

$$\text{デューティ} = 176 \times (62+1) / 16 \times 10^{6}$$

０．７ｍｓ

となり、ここでＬレベル区間の幅を計算すると次のようになります。

Ｌレベル区間の幅　＝　周期Ｔ　－　デューティ
＝　１　－　０．７
＝　０．３ｍｓ

ＤＣモータドライバＴＡ８４４０Ｈでは、ＰＷＭの‘Ｌ’レベル区間の幅でモータが回転します。従って、‘Ｌ’レベル区間の幅が大きくなるとモータの速度が速くなります。表８は、ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯが走行可能なｎ(タイマ値)に対するデューティと‘Ｌ’レベル区間の幅を上記の計算式から計算した一覧表です。光距離センサが補足する障害物までの距離設定が近すぎる場合、速度をあまり早くすると、障害物に衝突する可能性があります。原因は、使用されているモータとギャの構造上の問題や慣性の影響等により急停止ができません。ユーザプログラムを作成する場合に十分にご注意下さい。

表８　ＰＷＭのｎ値とデューティの一覧表（Ｔ＝１ｍｓ）

ｎ(タイマ値)	デューティ	Ｌ’レベル区間の幅	モータ速度
90h(144)	0.567ms	0.433ms	高速回転
a0h(160)	0.63ms	0.37ms	↕
b0h(176)	0.693ms	0.307ms	
c0h(192)	0.756ms	0.244ms	
d0h(208)	0.819ms	0.181ms	低速回転

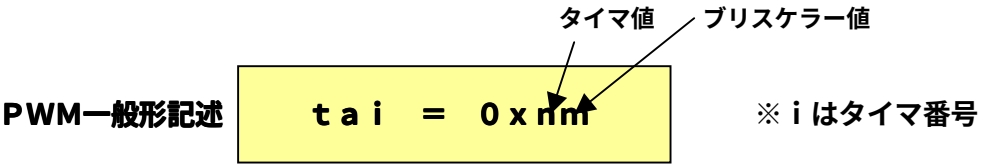


図１１と図１２は、以上の計算数値を用いてＰＷＭによる走行プログラムです。ＰＷＭ制御は、ポートＰ７４端子をタイマＡのＴＡ２ｏｕｔで使用し、右モータの速度制御用です。また、ポートＰ７６端子をタイマＡのＴＡ３ｏｕｔで使用し、左モータ速度制御用です。ＰＷＭの初期化プログラムを図１４に示します。次の手順でプログラムを記述します。

```
*****************************************************
* PWMの初期化                *
*****************************************************/
void     pwminit(void)
{
         ta2mr    = 0x27;      /* A1: 8ﾋﾞｯﾄPWM ﾓｰﾄﾞ  fi=f 使用 */
         ta3mr    = 0x27;      /* A4: 8ﾋﾞｯﾄPWM ﾓｰﾄﾞ  fi=f 使用 */
          ta2       = 62;       /* A0 初期値 n=0, m=3Eh  T=1ms    */
          ta3       = 62;       /* A1 初期値 n=0, m=3Eh  T=1ms    */

         tabsr     = 0x00;     /* A2 & A3 ｶｳﾝﾄ開始ﾌﾗｸﾞ  停止   */
}
```

図１４　ＰＷＭの初期化記述例

①　ＰＷＭモードの選択および各機能の選択

タイマＡのモードレジスタで右モータはｔａ２ｍｒ（＝３９８ｈ）、左モータはｔａ３ｍｒ（＝３９９ｈ）が対応します。これらのレジスタの各ビット図１５で示される意味を持ちます。

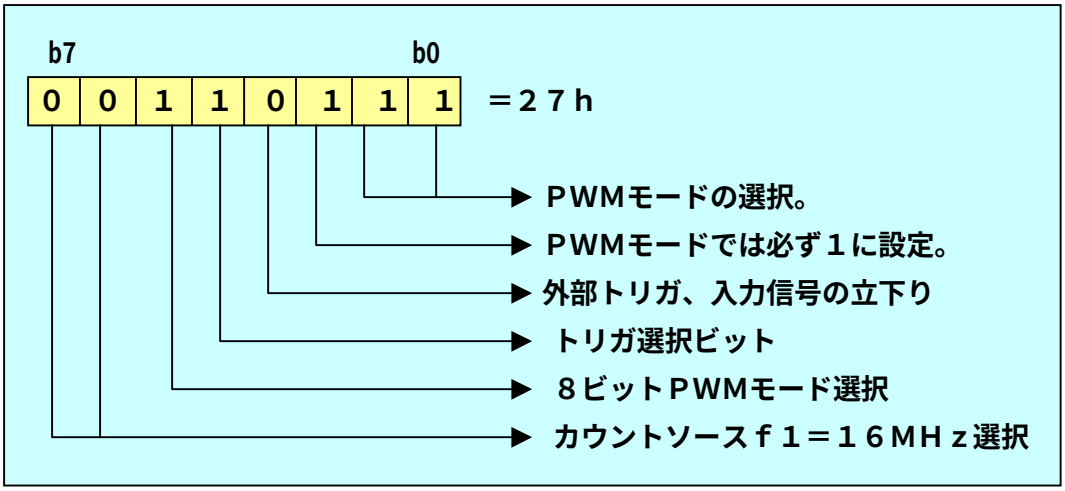


図１５　ＰＷＭモードの選択および各機能の選択

②　ＰＷＭパルスの周期および‘Ｈ’レベル幅の設定

図１６で示されるように、右モータはタイマＡ２レジスタｔａ２（＝３ＢＢｈ番地）、左モータはタイマＡ３レジスタｔａ３（＝３ＢＤｈ番地）が対応します。

図１６より、上位８ビットはタイマ値ｎの値を１６進数で設定します。また、下位８ビットはプリスケラー値(分周値)ｍの値を１６進数で設定します。

ｔａ２とｔａ３の上位８ビットにｎ＝００ｈを設定すると、ＰＷＭは動作せず、それぞれのＰＷＭ出力端子ＴＡ２ｏｕｔとＴＡ３ｏｕｔの出力レベルは‘Ｌ’レベルのままです。従って、初期設定では上位８ビットをｎ＝００ｈに設定します。

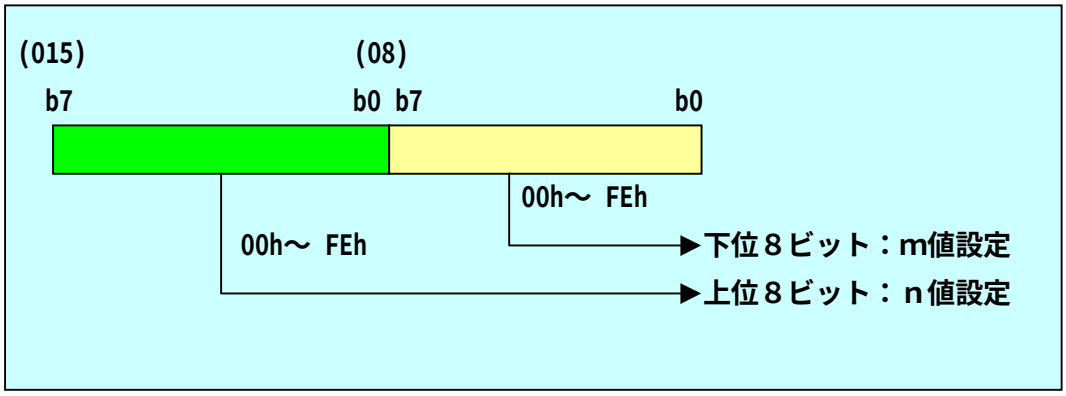


図１６　ＰＷＭパルスの周期および‘Ｈ’レベル幅の設定

③　カウント開始フラグの設定

カウント開始フラグｔａｂｓｒ（＝０３８０ｈ）は、右モータではタイマＡ２カウント開始フラグ、左モータではタイマＡ３カウント開始フラグが対応します。図１７で示されるようにタイマＡ０からタイマＡ４に該当するビットを１にするとＰＷＭ波

が出力されます。また、ビットを０に設定するとＰＷＭ出力は停止します。

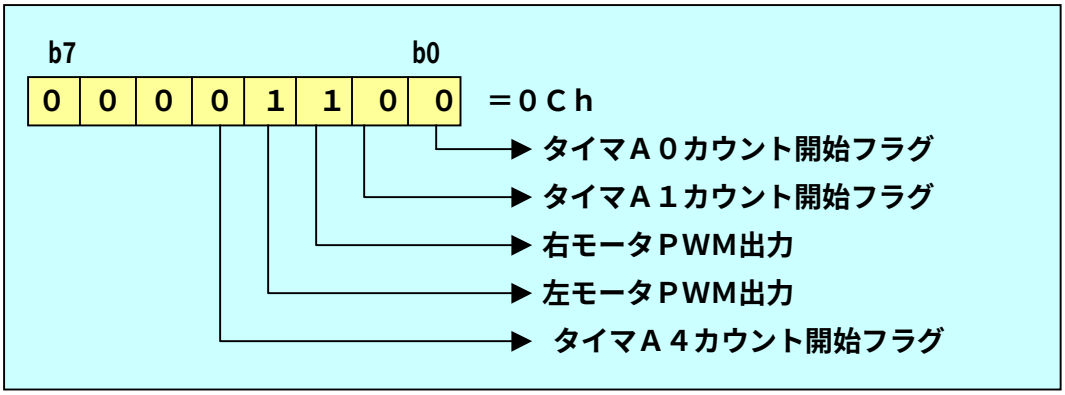


図１７　カウント開始フラグの設定

(４)、ＤＣモータの特性

ＤＣモータを使って何らかの制御を行う場合、ＤＣモータの等価回路を考慮する必要があります。図(ａ)はモータと電源を接続した回路例です。この回路を等価回路で置き換えると図(ｂ)のようになります。図(ｂ)より下記の関係式が成り立ちます。

$$Eb = Ra \times Ia + VB + Ec$$

上式よりＥｂ：電源電圧［Ｖ］、Ｒａ：電機子抵抗［Ω］、Ｉａ：モータ電流［Ａ］、 ＶＢ：ブラシ－整流子間の接触電圧［Ｖ］、Ｅｃ：モータの誘導電圧［Ｖ］を表します。

ここで、ブラシ－整流子間に発生する接触電圧ＶＢはＥｂ≫ＶＢ、Ｅｃ≫ＶＢから、これを無視すると上式は

$$Eb = Ra \times Ia + Ec$$

のようになります。従って、ＤＣモータは、一定の回転数で回転している場合、図(ｃ)のような等価回路で表すことができます。

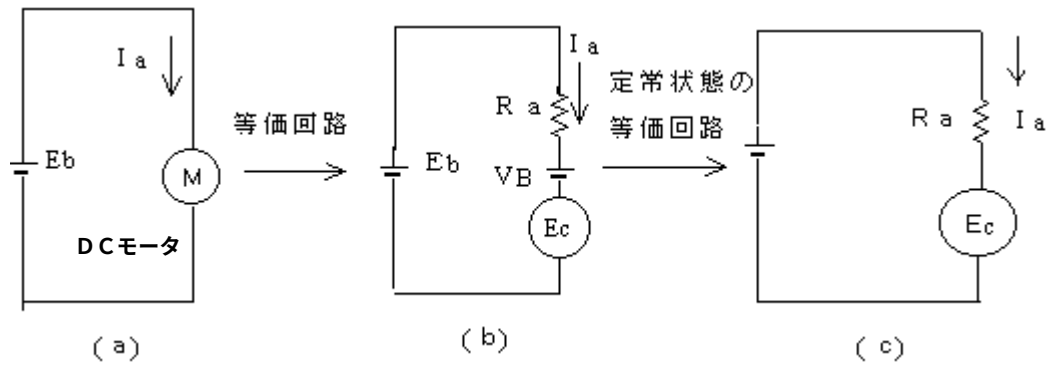


図１８　ＤＣモータの等価回路

これらの関係から下図のようなＤＣモータの特性が得られます。図１９より電圧を上げるとこれに比例して回転数が速くなります。モータのトルクＴが大きくなり、より大きな力が得られます。また、電圧を下げると回転数ウが遅くなり、これに比例してトルクＴも小さくなります。

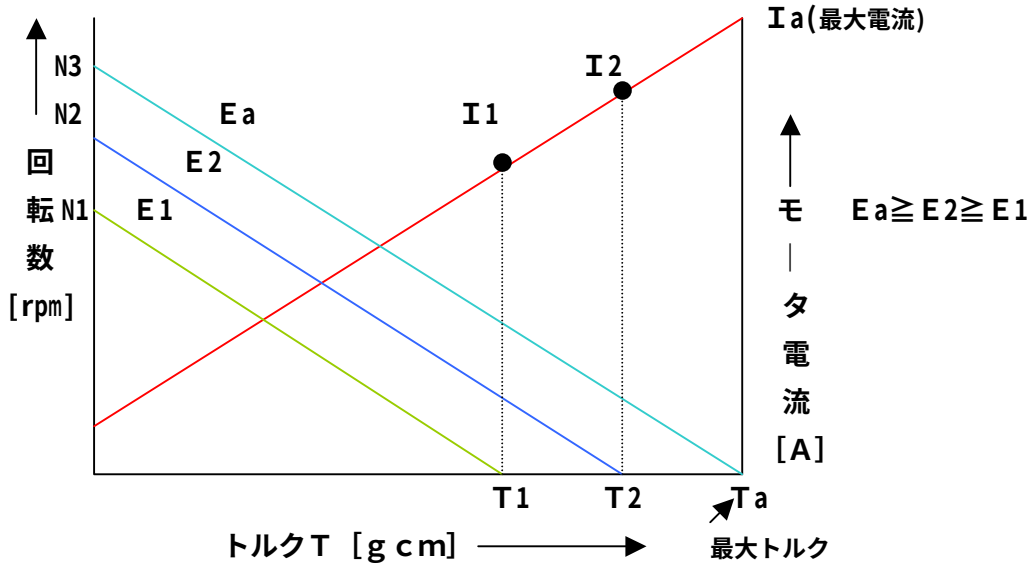


図１９　ＤＣモータの特性

ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯでもＰＷＭの制御によって自走速度を自由に変えられますが、ＰＷＭの設定値を変えてモータの速度を遅くした場合、ある設定値以下になると、車体を持ち上げたときにタイヤが回転してもモータのトルクと車体の重量により走行できなくなります。ご注意下さい。

(５)、光距離データとＡ／Ｄ変換器の使い方

右側光距離センサのＤＣ電圧出力は、Ａ／Ｄ変換器のアナログ入力ＡＮ０端子から取り込みます。中央右側光距離センサのＤＣ電圧出力は、Ａ／Ｄ変換器のアナログ入力ＡＮ１端子から取り込みます。また、左側光距離センサのＤＣ電圧出力は、Ａ／Ｄ変換器のアナログ入力ＡＮ２端子から取り込みます。図２０は、光距離センサ回路とＡ／Ｄ変換器のアナログ入力端子の接続を示したものです。

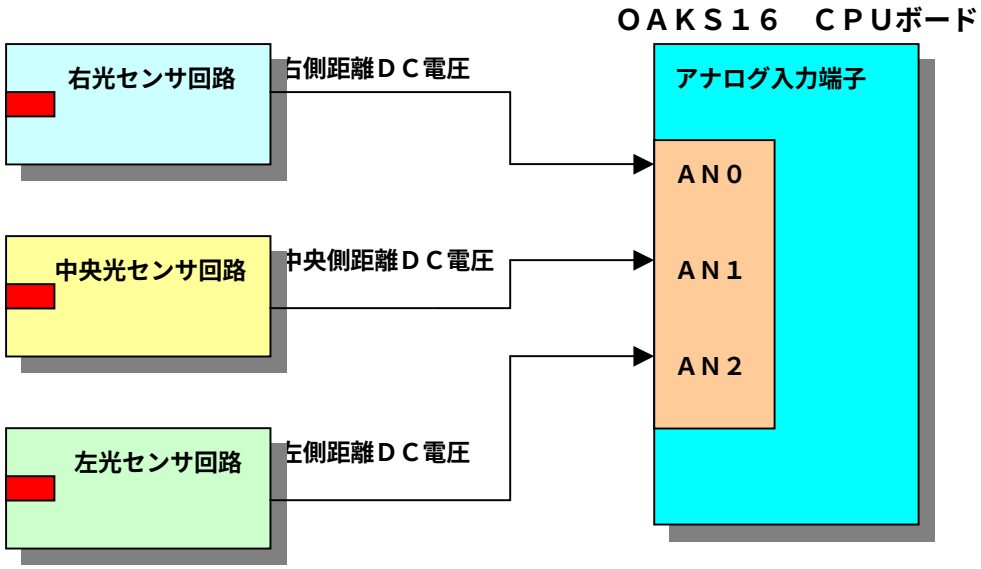


図２０　光距離センサ回路とＡ／Ｄ変換器の接続

Ａ／Ｄ変換器の初期設定について順を追って説明します。図２１は、Ａ／Ｄ変換器の初期設定を記述したプログラム例です。

```
/************************************************************
*  Ａ／Ｄ変換器の初期設定                                    *
************************************************************/
void  adinit(void)
{
        adcon2 = 0x01;     /*  サンプルホールド有り     */
        adcon0 = 0x80;     /*  fAd/2, 単発モード,AN0 端子使用   */
        adcon1 = 0x28;     /*  EXIT OP AMP NO, fAD/2, 10bit mode, Vref */
        adcon2 = 0x01;     /*  サンプルホールド有り     */
        adcon0 = 0x81;     /*  fAd/2, 単発モード,AN1 端子使用     */
        adcon1 = 0x28;     /*  EXIT OP AMP NO, fAD/2, 10bit mode, Vref */
        adcon2 = 0x01;     /*  サンプルホールド有り     */
        adcon0 = 0x82;     /*  fAd/2, 単発モード,AN2 端子使用     */
        adcon1 = 0x28;     /*  EXIT OP AMP NO, fAD/2, 10bit mode, Vref */

        adic   = 0x03;     /* ad 割り込みレベル３の設定 */
}
```

図２１　Ａ／Ｄ変換器の初期設定プログラム例

① サンプル＆ホールドの選択

サンプル＆ホールドは、Ａ－Ｄ制御レジスタ２「ＡＤＣＯＮ２」(３Ｄ４ｈ)のビット b0 に１を書き込むと使用が可能となります。図２２は、サンプル＆ホールドの選択をありに設定した例です。

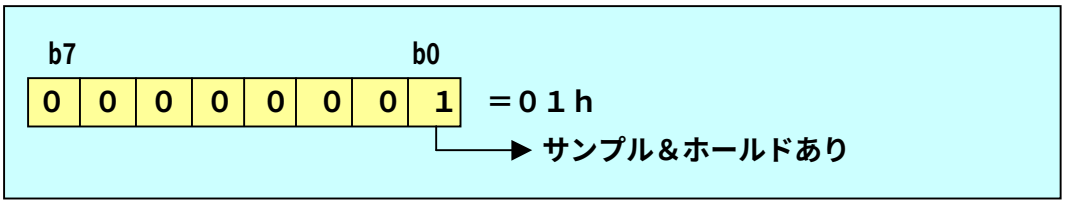


図２２　サンプル＆ホールドの選択

② Ａ－Ｄ制御レジスタ０の設定

Ａ－Ｄ制御レジスタ０は、図２３で示されるように、Ａ－Ｄ制御レジスタ０「ＡＤＣＯＮ０」(３Ｄ４ｈ)の各ビット毎にＡ／Ｄ変換器の使用方法を決めます。

Ａ／Ｄ変換器は単発モードでします。また、Ａ／Ｄ変換器の動作クロックを ｆAD／２で使用します。

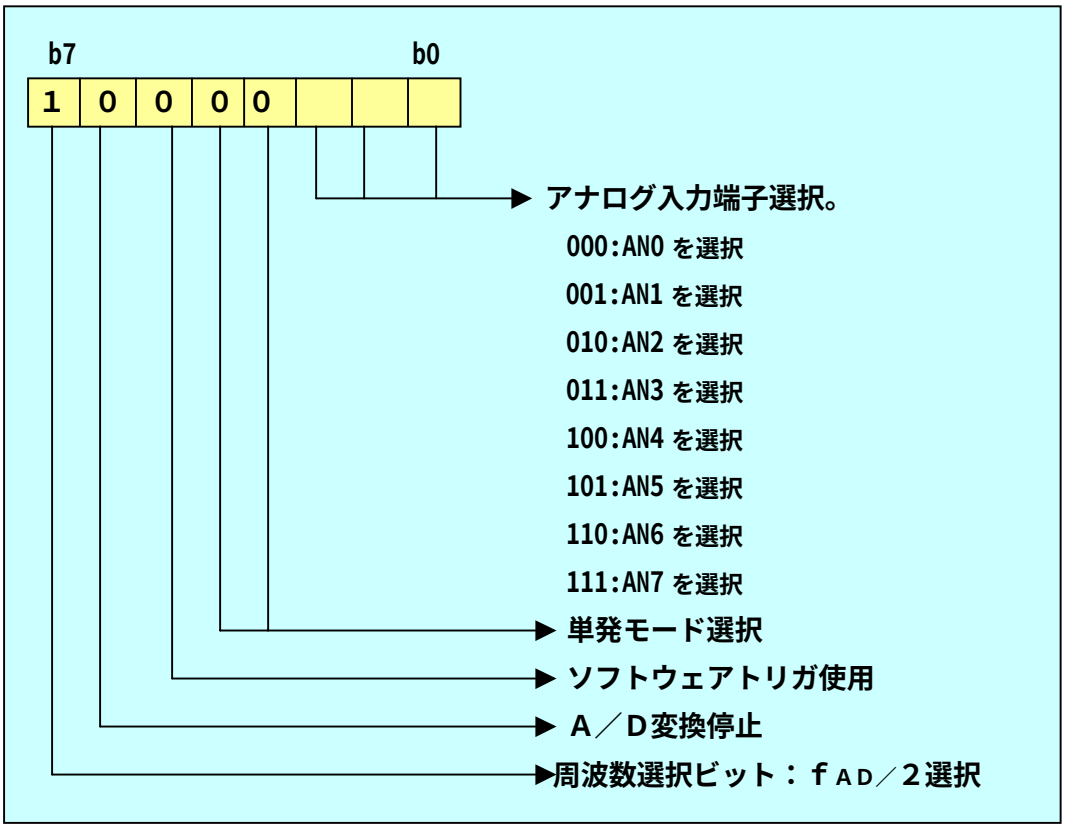


図２３　Ａ－Ｄ制御レジスタ０の設定

③　Ａ－Ｄ制御レジスタ１の設定

Ａ－Ｄ制御レジスタ１は、図２４で示されるように、Ａ－Ｄ制御レジスタ１「ＡＤＣＯＮ１」(３Ｄ７ｈ)の各ビット毎にＡ／Ｄ変換器の使用方法を決めます。
ここでは、１０ビットＡ／Ｄ変換器として使用します。

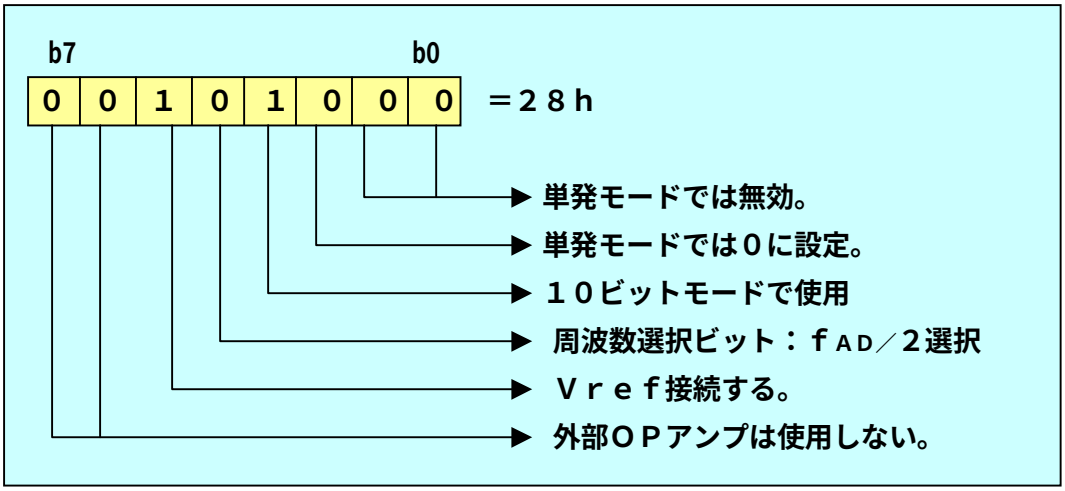


図２４　Ａ－Ｄ制御レジスタ１の設定

③　Ａ／Ｄ変換開始

Ａ－Ｄ制御レジスタ０は、図２５で示されるように、Ａ－Ｄ制御レジスタ０「ＡＤＣＯＮ０」(３Ｄ４ｈ)のビット６を１に設定するとＡ／Ｄ変換器を開始します。

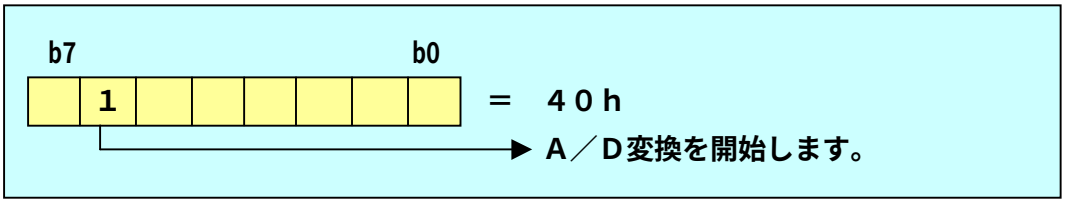


図２５　Ａ－Ｄ変換の開始

④　変換結果の読み出し

図２６で示されるＡ／Ｄ変換レジスタａｄ０(右光距離データ)、ａｄ１(中央光距離データ)、ａｄ２(左光距離データ)に１０ビットのデジタルデータとして読み出されます。

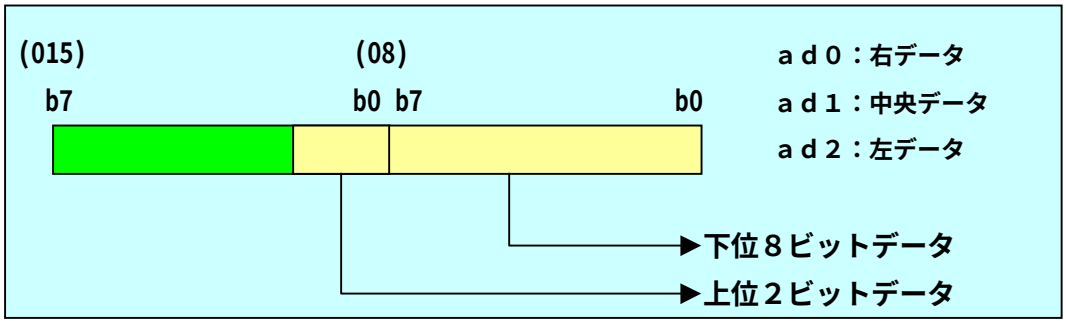


図２６　変換結果の読み出し

８.添付ファイルの構成と使い方

センサロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」には表９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９　ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ添付ファイル一覧表

ディレクトリ名	ファイル名	内　　容
Robo_1	Robo_1.c Robo_1.x30 Robo_1.mot	前進走行－後進走行の繰り返しソースプログラム デバッグファイル。 フラッシュＲＯＭ書き込みファイル。
Robo_2	Robo_2.c Robo_2.x30 Robo_2.mot	前進－Ｕターン走行の繰り返しソースプログラム。 デバッグファイル。 フラッシュＲＯＭ書き込みファイル。
Robo_3	Robo_3.c Robo_3.x30 Robo_3.mot	障害物からの距離を検出し障害物を避けるソースプログラム。 デバッグファイル。 フラッシュＲＯＭ書き込みファイル。

(1)、ＤＣモータとギャーの使用上の留意点

センサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」には、田宮模型のツインモータギヤーボックスを使用しております。添付されているサンプルデモプログラムやユーザプログラムで走行させる場合、次の点につきましては十分に留意されてからご使用下さい。

① 使用されているＤＣモータは、ＰＷＭ制御によって走行速度の制御を行いますが、乾電池の消耗によっても走行速度が遅くなります。したがって、電池の消耗によって設計通りの動作ができなくなります。十分にご注意下さい。

② 使用されているツインギャーボックスは、構造上の問題から左右のモータのトルクに差が出ます。設定したＰＷＭデータが左右同じであっても直線上を走行せず、何れかの方向にカーブしますのでご注意ください。

（２)、サンプル「Ｒｏｂｏ＿1」の使い方

サンプルデモプログラム「Ｒｏｂｏ＿１」は、電源スイッチをＯＮにすると、一定時間の間、前進走行を行い一旦停止します。次に、同じ一定時間で後進走行で戻ってきます。停止

後に前進走行を繰り返します。図２７は、前進走行と後進走行を繰り返すｍａｉｎ（）関数のプログラムです。一定時間は、ソフトタイマｗａｉｔ（ｓｈｏｒｔ　ｋ）関数で行います。定数ｋの値で時間を変えられます。

```
void      main()
{
        pwminit();                  /*  PWMの初期化 */
        portinit();                 /*  ポートの初期化 */
        port_0 = 0x00;              /* LED の点灯       */
        while(1)
        {
                for_run();                  /* 前進走行     */
                wait(600);                  /* 走行距離の設定  */
                stp_run();                  /* 両モータ停止     */
                wait(200);
                bak_run();                  /* 後進走行     */
                wait(600);                  /* 走行距離の設定  */
                stp_run();                  /* 両モータ停止     */
                wait(200);
        }
}
```

図２７　「Ｒｏｂｏ＿1」のｍａｉｎ（）関数記述例

（３）、サンプル「Ｒｏｂｏ＿２」の使い方

サンプルデモプログラム「Ｒｏｂｏ＿２」は、電源スイッチをＯＮにすると、一定時間の間、前進走行を行い一旦停止します。次に、Ｕターンを行って一旦停止します。次に、同じ一定時間で前進走行で戻ってきます。

図２８はＵターン動作を行う関数「ｕ＿ｒｕｎ()」の記述例です。Ｕターン動作を行う場合、何れか一方のモータを停止し、他方を一定時間回転させるとターンをしますが、この方法では、同じコース上に戻ってきません。同じコース上に戻すには、左右のモータを一定時間逆転させます。

図２８のように、右モータをｐｏｒｔ＿1＝０Ｘ０２記述によって正転させます。また、左モータをｐｏｒｔ＿３＝０Ｘ０２記述で正転させます。左右のモータはお互いに向かい合っているので、お互いに反対方向に回転してＵターンを行います。

```
void      u_run(void)
 {
          port_1 = 0x02;          /* 右ターン走行     */
          port_3 = 0x02;          /* 左ターン走行     */
          ta2 = 0xb03e;           /* 右モータ用ＰＷＭデータ        */
          ta3 = 0xb03e;           /* 左モータ用ＰＷＭデータ        */
          tabsr = 0x0c;
 }
```

図２８　Ｕターン関数の記述例

（４）、サンプル「Ｒｏｂｏ＿３」の使い方

サンプルデモプログラム「Ｒｏｂｏ＿３」は、電源スイッチをＯＮにすると、前方、左右に取り付けられている光距離センサーによって障害物までの距離をＡ／Ｄ変換器で計測し、予めプログラムされている距離まで接近すると、次の回避動作を行います。障害物が黒色の場合には、光が反射しませんので障害物を検地できません。できる限り白色の障害物をご用意下さい。

① 中央センサーが障害物に接近した場合
中央センサーが障害物に予めプログラムされている距離まで接近すると、一定時間で後進動作を行い、次にＵターン動作を行います。一旦停止後に前進走行を行います。

② 右側センサーが障害物に接近した場合
右側センサーが障害物に予めプログラムされている距離まで接近すると、一旦停止後、約９０度左にターンし、障害物を避けます。次に、一旦停止後に前進走行を行います。

③ 左側センサーが障害物に接近した場合
左側センサーが障害物に予めプログラムされている距離まで接近すると、一旦停止後、約９０度右にターンし、障害物を避けます。次に、一旦停止後に前進走行を行います。

フォトレフレフクタＲＰＲ－２２０を距離センサーとして使用する為に、Ｉ／０ポートＰ０４端子からトランジスタ２ＳＣ１８５のベースに８００Ｈｚのパルス波を与えます。

図２９は、ソフトで８００Ｈｚのパルス波を作るプログラムの記述例です。

```
*********************************************************************
*       タイマ割込み関数 : ＴＡ０
*********************************************************************/
void  ta0int(void)
{
        time400Cnt--;                       /* 400Hz カウンタ-1         */
        if(time400Cnt == 0){
                port_0 ^= 0x10;             /* P04 のオン <-> オフ    */
                time400Cnt = TA01SCNT;      /* 400Hz カウンタ初期化(8)   */
        }
        ta0 = TA0CNT;                       /* タイマ値の初期化       */
}
```

図２９　８００Ｈｚのパルス波出力プログラム例

周波数８００Ｈｚの周期は１．２５ｍｓとなります。ソフトで半周期(０．６２５ｍｓ)の間隔でポートＰ０４端子からＨレベルとＬレベルを交互に出力すれば、８００Ｈｚのパルス波が得られます。

従って、タイマ割込みによって半周期毎に割込み関数でポートＰ０４端子をＯＮとＯＦＦを繰り返します。タイマ設定値はボートのクロック周期が２μｓになっているので次の数式から算出します。

$$\text{タイマ設定値} = \frac{０．６２５ \times １０\text{-}3 \ / \ ２ \times １０\text{-}6}{３１２．５}$$

ここでタイマＡ０を使用します。このタイマは１６ビットカウンタなので上記のタイマ値を３１×１０＝３１０として２つの変数に分け、プログラムの中で最初に定義しておきます。

```
#define     TA0CNT    31-1          /* A0:31:タイマカウンタ値      */
#define     TA01SCNT 10             /* A0:秒カウンタ値 31*10=310   */

 int        time400Cnt;             /* 400Hz 用カウンタ       */
```

（５）、距離の検出方法

光距離センサで得られたＤＣ電圧をＡ／Ｄ変換器に入力し、距離を検出するサンプルプログラムを図３０に示します。アナログデータは補正値を用いて次のようにＡ／Ｄ変換し、デジタル変換データを変数ａｎ__0 に格納します。

```
an_0 = (ad0 & 0x03ff)  / 5 ;
```

Ａ／Ｄ変換器は１０ビットモードで使用するので、変換結果の読み出しレジスタａｄ０の値を上位５ビットをマスクするために０Ｘ０３ｆｆとのＡＮＤをとります。

距離に対するＤＣ出力電圧特性は、表４で表されている通り反比例します。従って、定数１００から減算して比例して増加するように変更します。この値を障害物までの距離データとして使用します。

```
an_0 =   100   -   an_0;
```

```
void     ad0_int(void)              /* A／D変換割込み          */
{
         unsigned _far int  an_0, an_1, an_2;
         _asm( "\tFCLR      I");             /* 割り込み禁止    */

         adcon0= 0x40;
         an_0 = (ad0 & 0x03ff)  / 5 ;        /* A/D ch0 データの取得: 補正値= 5 */
         an_0 = 100 - an_0;

         adcon0= 0x41;                       /* A/D ch1  変換開始        */
         an_1 = (ad1 & 0x03ff)  / 5 ;        /* A/D-1 データの取得: 補正値= 5 */
         an_1 = 100 - an_1;

         adcon0= 0x42;                    /* A/D ch2  変換開始        */
         an_2 = (ad2 & 0x03ff)  / 5 ;  /* A/D-2 データの取得: 補正値= 5 */
         an_2 = 100 - an_2;
```

図３０　距離の検出プログラム例

９.フラッシュＲＯＭへの書き込み方法

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。

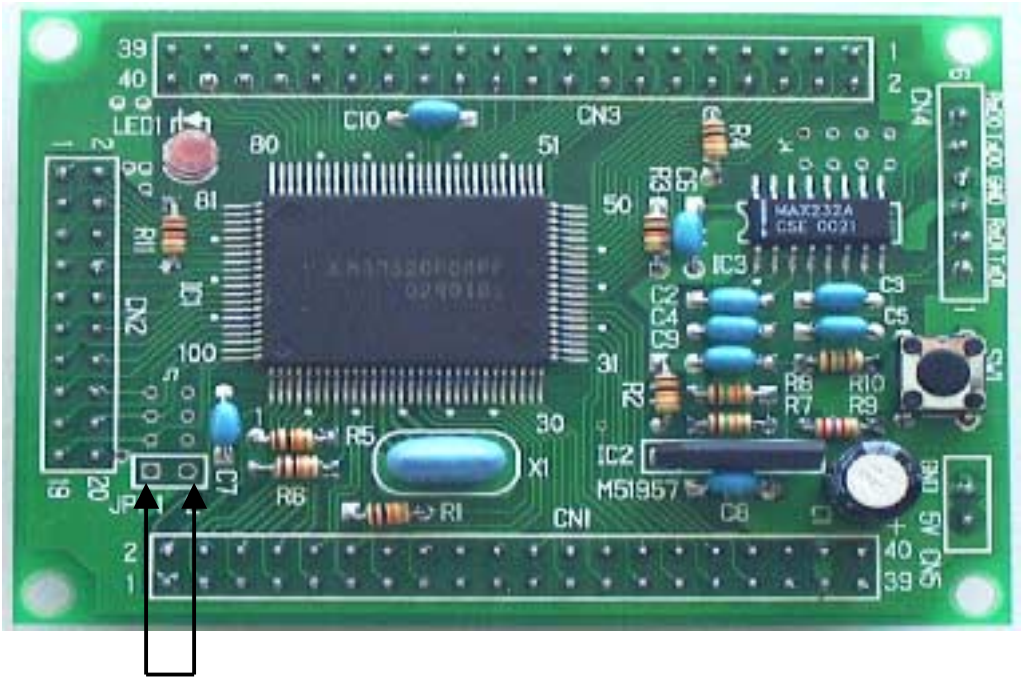
センサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」には、ＯＡＫＳ１６ＣＰＵボードは付属しておりません。お持ちでない場合には、別途ご購入下さい。

商品に添付されているＣＤ-Ｒに入っているサンプルデモプログラムをフラッシュＲＯＭに書き込んで実行する場合とリモートデバッガＫＤ３０からサンプルデモプログラムを実行する方法をご説明します。

(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＿ｕａｒｔ．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。

センサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」とパソコン間をＲＳ２３２Ｃケーブルで接続します。また、図３１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３１　ＣＰＵボード「ＯＡＫＳ１６」のＪＰ1端子

ＯＡＫＳ１６－ＳＥＮＳＯＲ　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３２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。

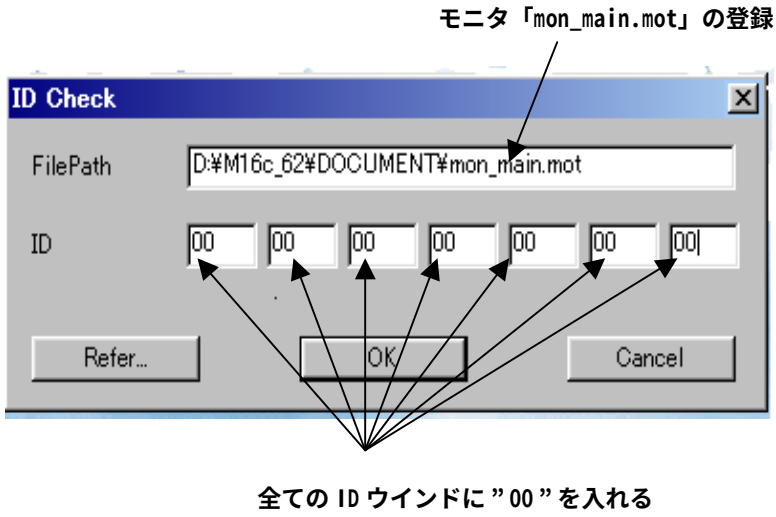


図３２　「ID Check」　ウインドウ

④　図３３で示される「Ｍ１６Ｃ　ｆｌａｓｈ　Ｗｒｉｔｅ」ウインドウが表示します。

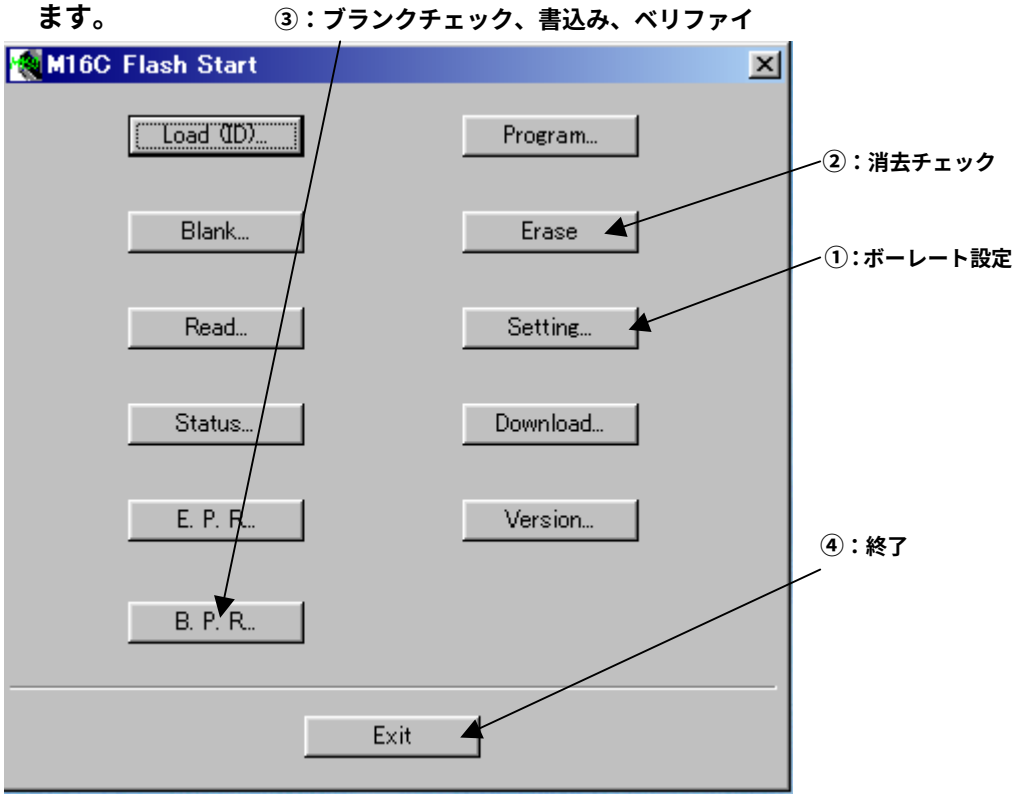


図３３　「Ｍ１６Ｃ　ｆｌａｓｈ　Ｗｒｉｔｅ」ウインドウ

⑤　「Ｓｅｔｔｉｎｇ」ボタンを押します。図３４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。

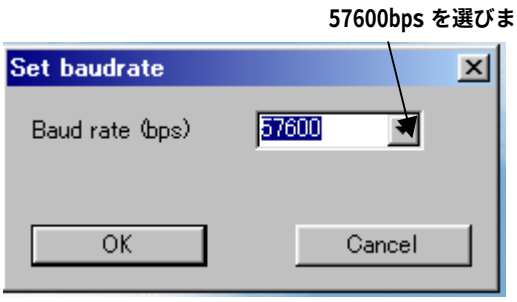
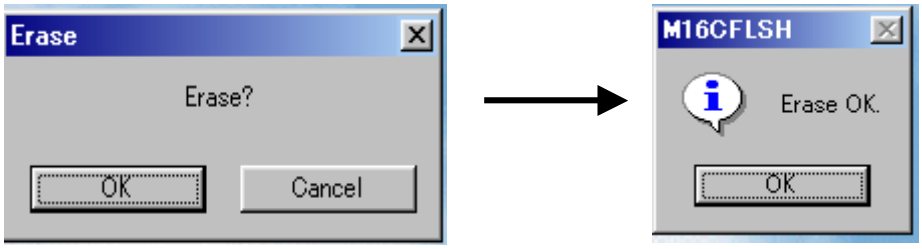


図３４　「Ｓｅｔ　ｂａｕｄｒａｔｅ」ウインドウ

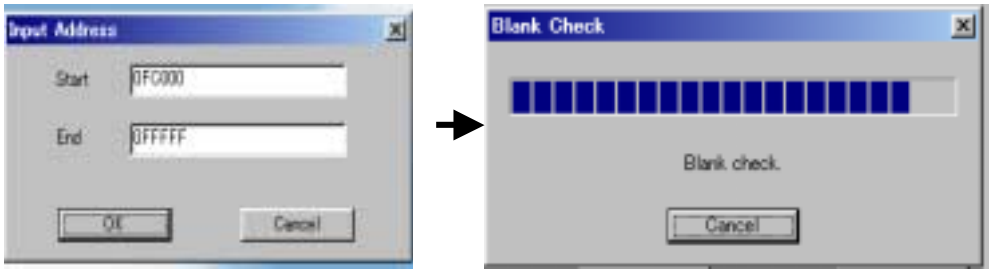
⑥　「Ｅｒａｓｅ」ボタンを押します。図３５（ａ）で示されるウインドウの「ＯＫ」ボタンを押すと、（ｂ）図で示される「Ｅｒａｓｅ　ＯＫ」ウインドウが表示します。



（ａ）　　（ｂ）

図３５　「Ｅｒａｓｅ」チェックウインドウ

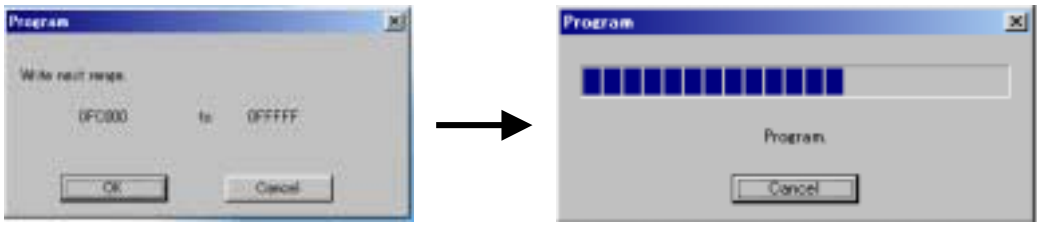
⑦　「Ｂ．Ｐ．Ｒ」ボタンを押します。このボタンは、ブランクチェック、プログラムの書込み、書込み確認を一連で実行します。図３６（ａ）で示されるプログラムのメモリ範囲が表示されます。確認後、「ＯＫ」ボタンを押します。図（ｂ）のウインドウが開きブランクチェックを開始します。



（ａ）　　（ｂ）

図３６　「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３７（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３７　「Ｐｒｏｇｒａｍ」ウインドウ

①　プログラムが正常に書き込まれると、図３８（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３８　「Ｒｅａｄ　ｃｈｅｃｋ」ウインドウ

⑩　最後に、図３３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、ＯＡＫＳ１６－ＳＥＮＳＯＲ　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ケーブルで接続します。

デバッガ「KD30」アイコンをクリックすると図３９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３９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。

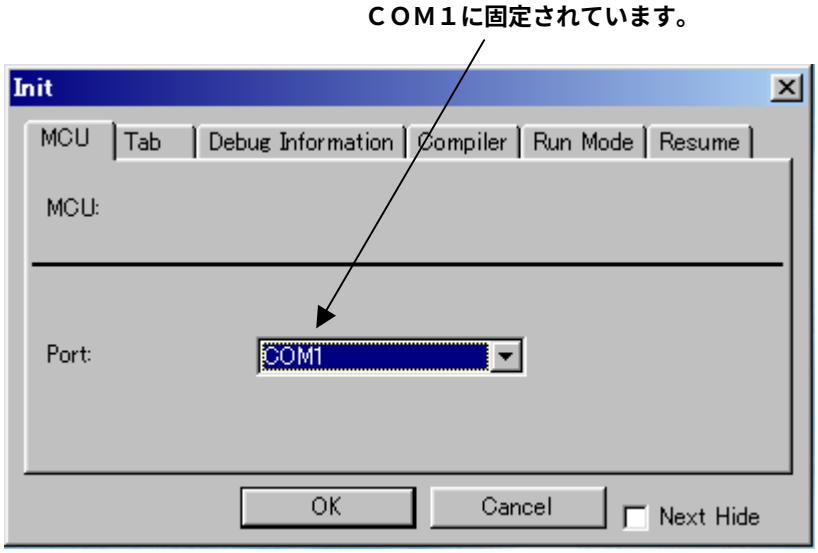


図３９　デバッガ起動ウインドウ

「ＯＫ」ボタンを押すと、図４0 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「Ｒｏｂｏ＿１．　ｘ30」と「Ｒｏｂｏ＿２．ｘ30」と「Ｒｏｂｏ＿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４０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。

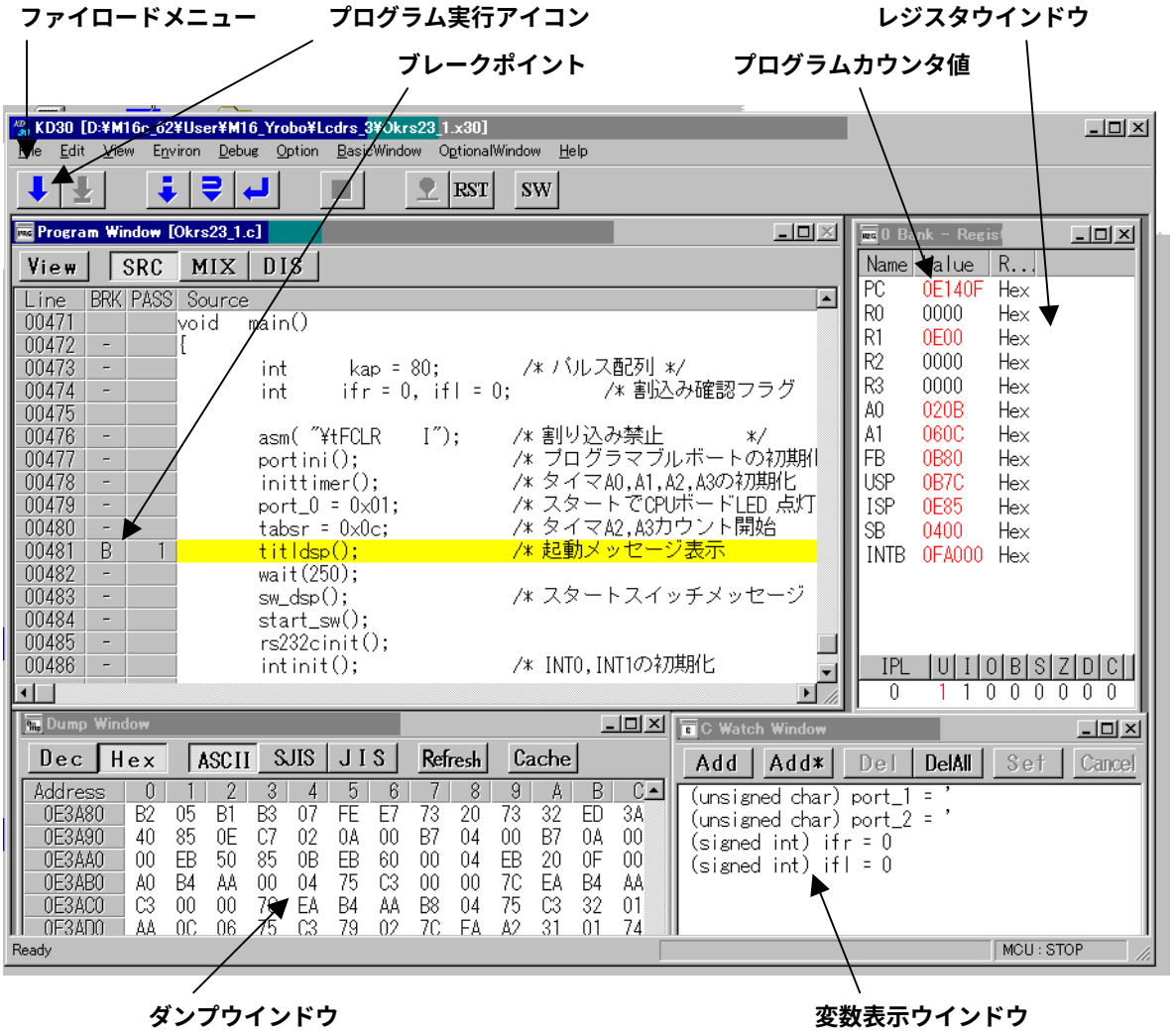


図４０　デバッガＫＤ３０デバッグ画面

１０．ＴＭ統合化開発環境での使い方

別途、センサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」に必要なＯＡＫＳ１６ＣＰＵボートには、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。

センサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」に添付されているＣＤ－ＲにはＴＭ統合開発環境「ＴＭＶ３．１１」は付属しておりません。

（１）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パイルする際、表１０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１０　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。

③　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１１　使用できないソフトウェアおよびユーティリティー覧

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１２　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１３　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

（２）ＴＭ統合化開発環境の使い方

「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４１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。

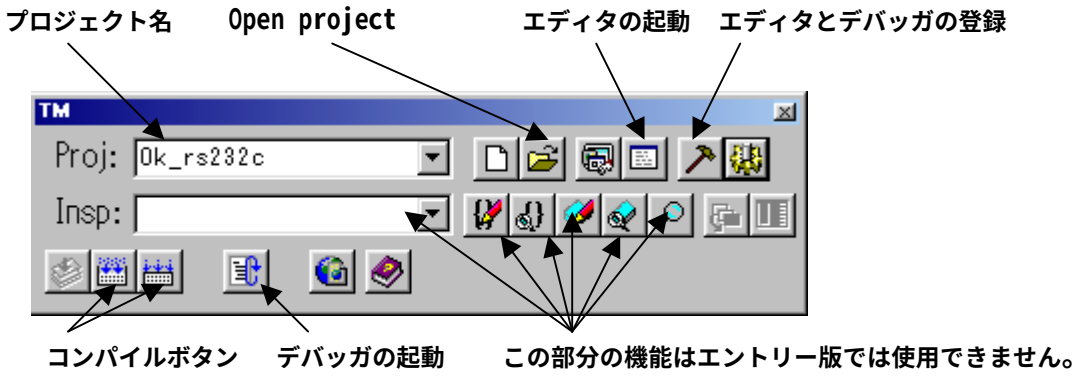


図４１　プロジェクトバー

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４１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「Ｒｏｂｏ＿３．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「Ｒｏｂｏ＿３．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４１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、ＯＡＫＳ１６ＣＰＵボードに付属のＣＤ－Ｒに入っているユーザーズマニュアルをご覧下さい。

１１.プログラムの作成手順

センサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」には、プログラム開発の為の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おりません。別途ＯＡＫＳ１６　ＣＰＵボードに付属しております。

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４２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「ＮＣ３０WA」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。

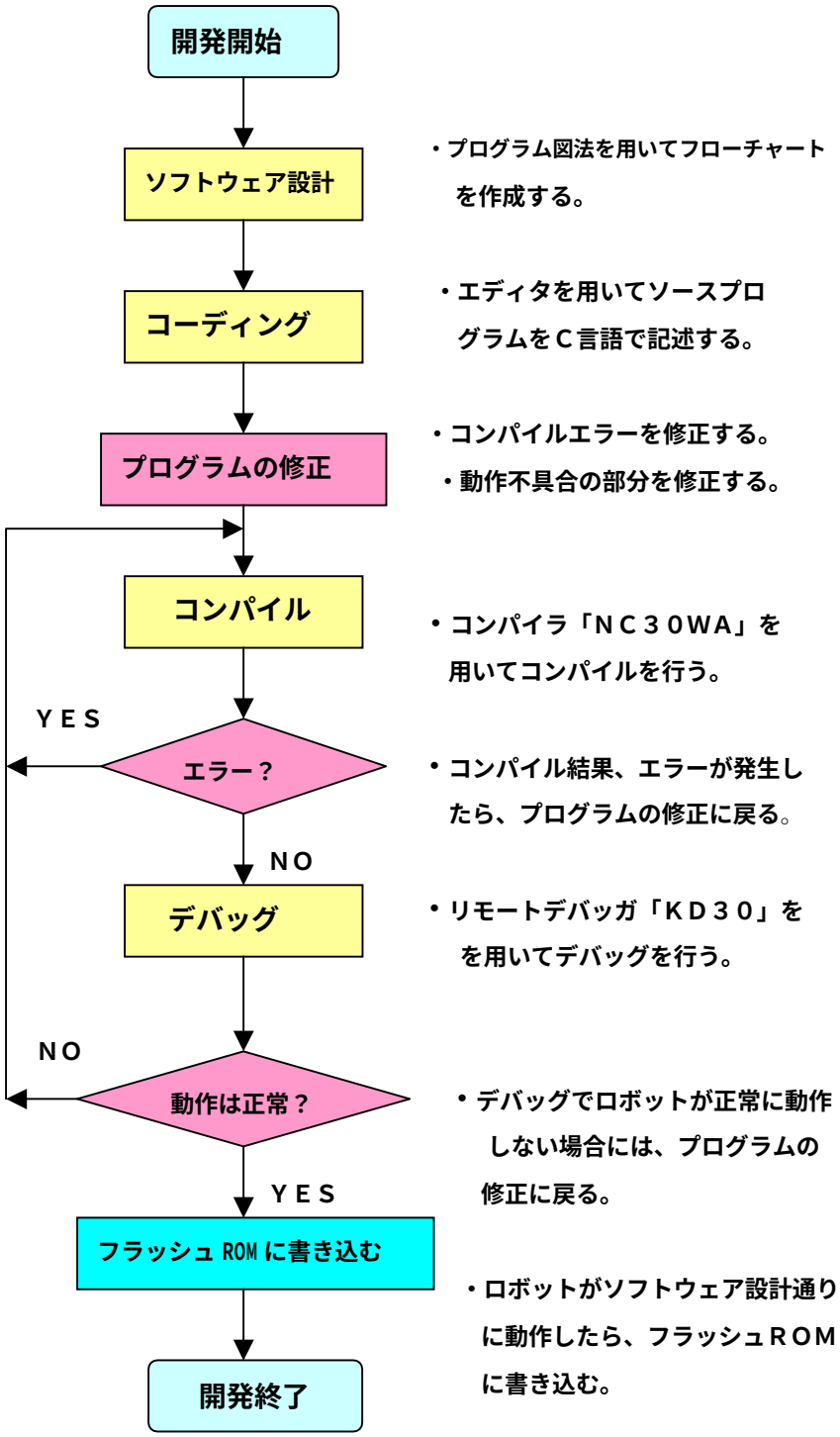


図４２プログラム開発の手順

(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４３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「Ｒｏｂｏ＿３．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「Ｒｏｂｏ＿3．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「Ｒｏｂｏ＿３.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。

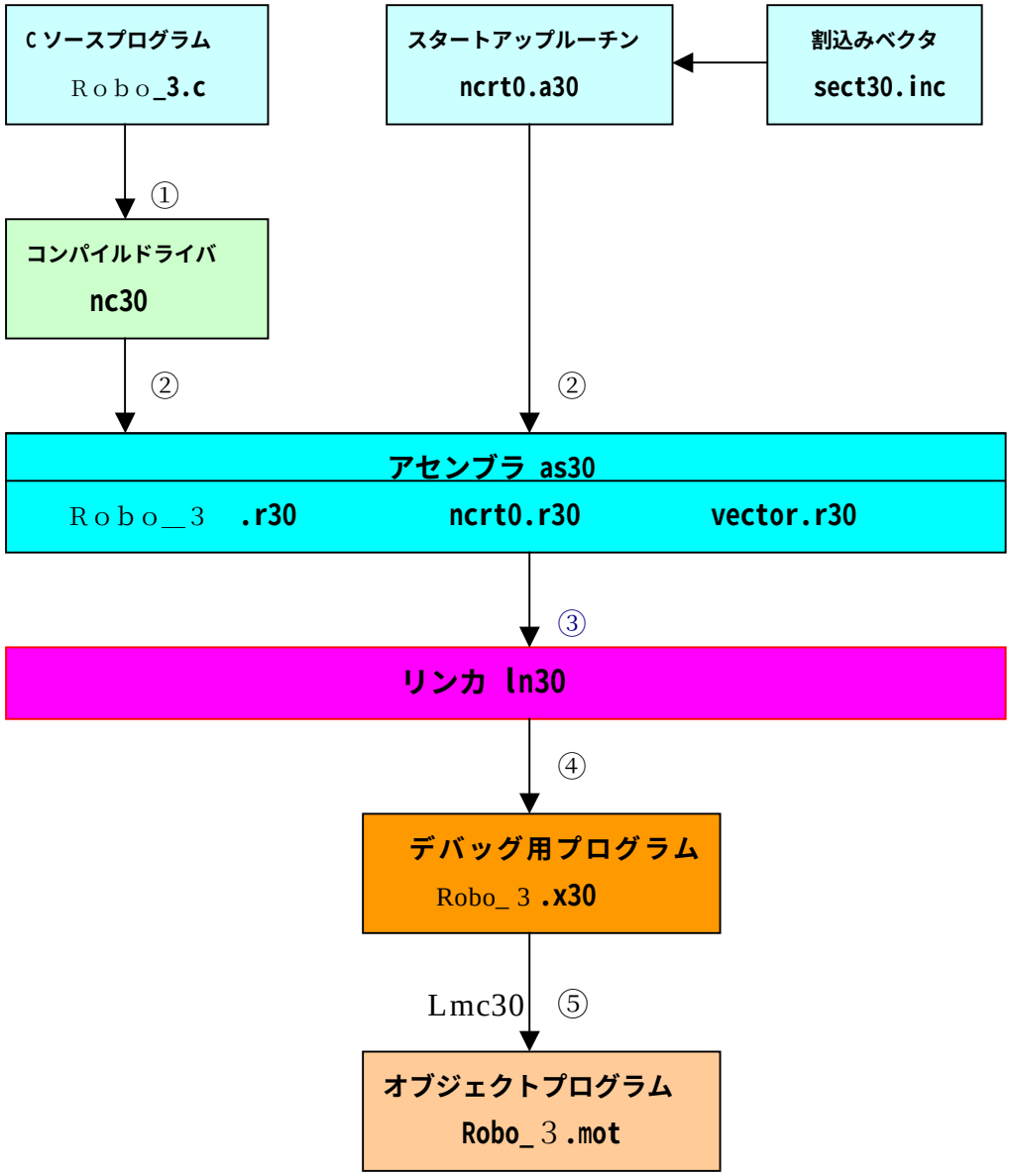


図４３ コンパイラ「ＮＣ３０ＷＡ」のコンパイル手順

(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４４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。

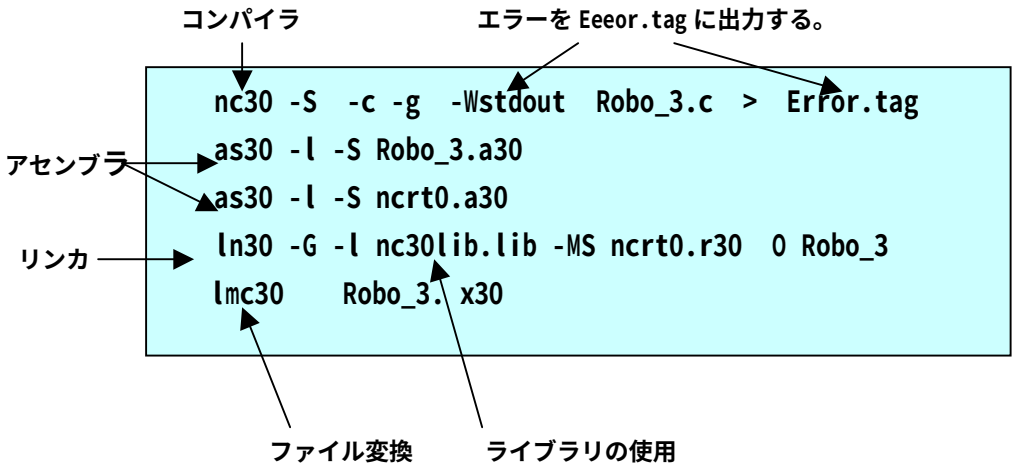


図４４ コンパイル用バッチファイル記述例

図４４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「Ｒｏｂｏ＿３．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４４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／０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４５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。

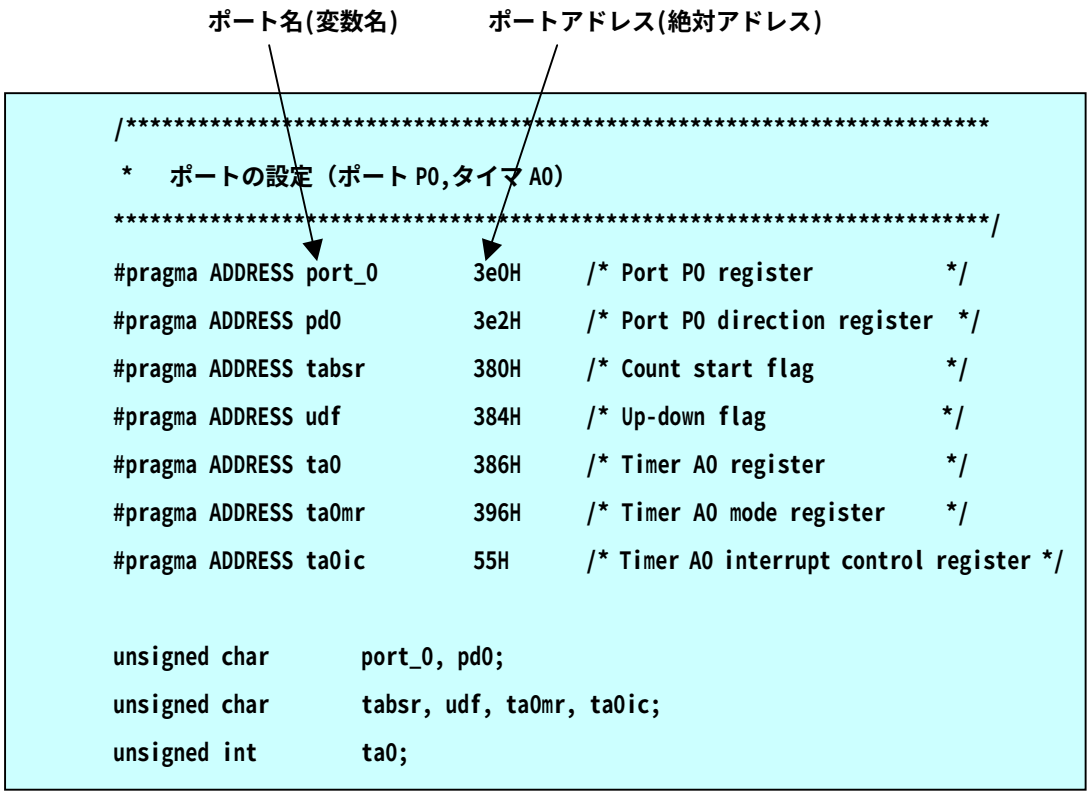


図４５　Ｉ／Ｏポートの記述例

(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４６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。

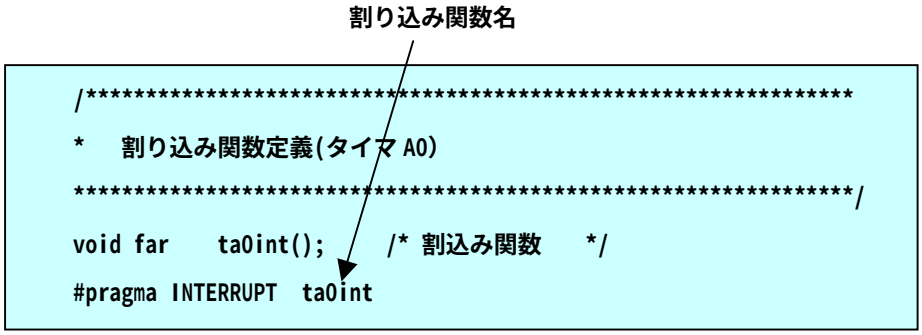


図４６　割り込みの記述例

(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４７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４８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４７　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４８　「ｓｅｃｔ３０．ｉｎｃ」ベクターテーブルの記述例

１２．Flash ROM への書込み

デバッガによってセンサーロボット「ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ」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１４は、図３３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。

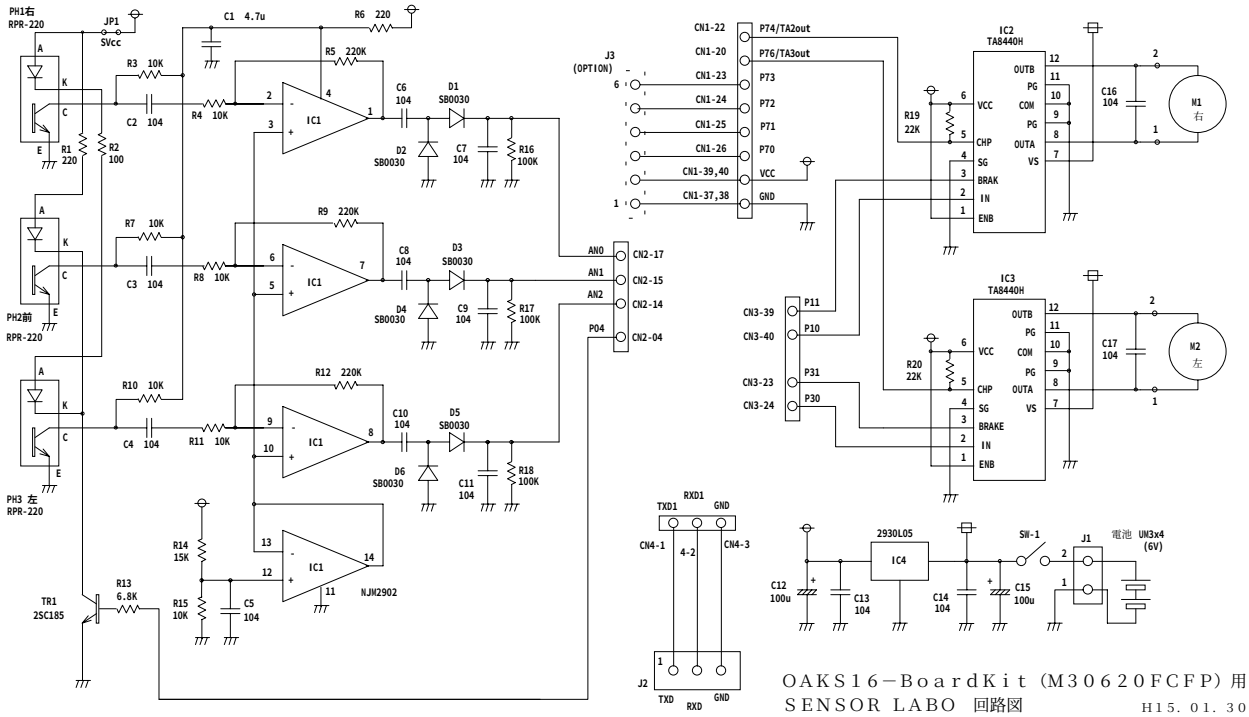
表１４ 各メニューボタンの機能

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

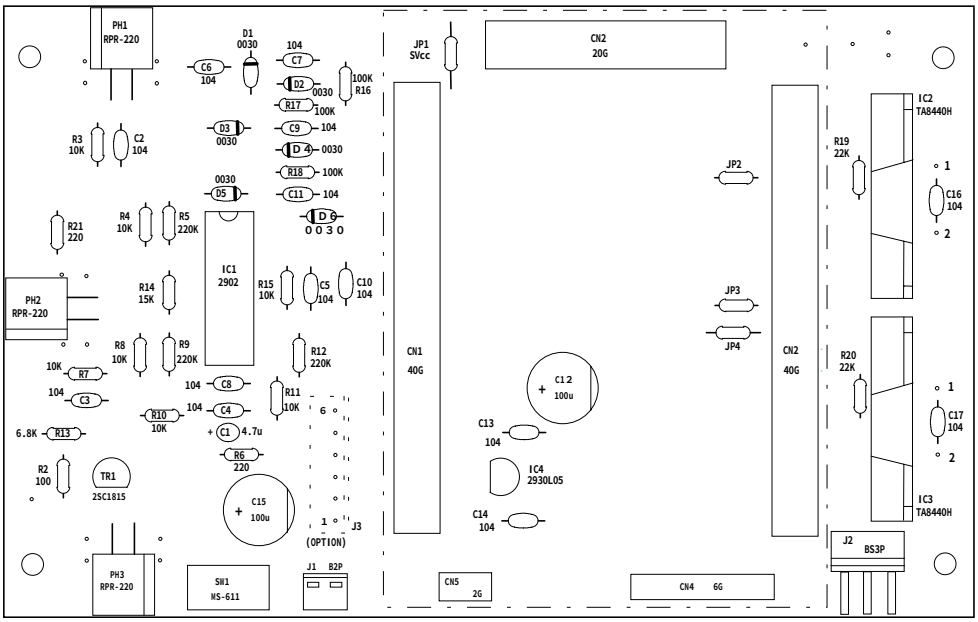
●拡大してご覧下さい。



OAKS16－BoardKit（M30620FCFP）用
SENSOR LABO　回路図　H15. 01. 30

付録―2　ドライバーボード構成図

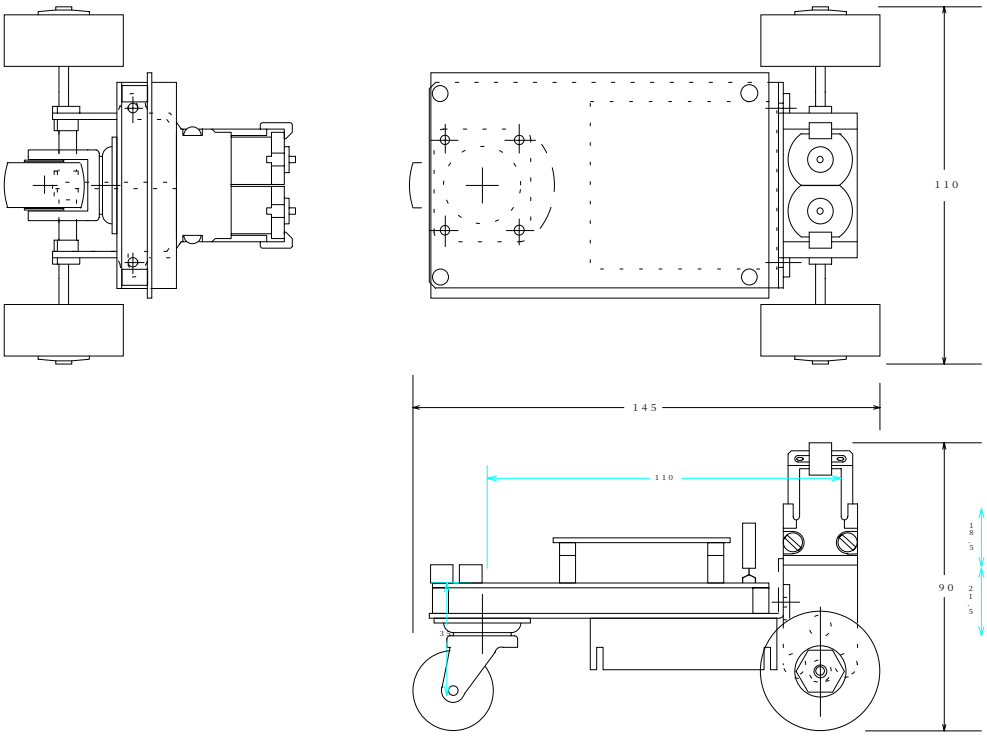
●拡大してご覧下さい。



OAKS16－BoardKit（M30620FCFP）用
SENSOR　LABO　基板実装図　H15. 02. 20

付録―３　ＯＡＫＳ１６－ＳＥＮＳＯＲ　ＬＡＢＯ外形図

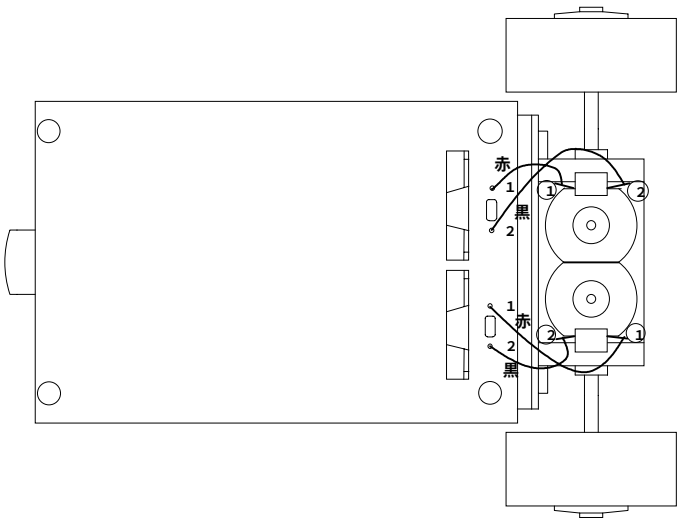
●拡大してご覧下さい。



付録―４　モータ配線図

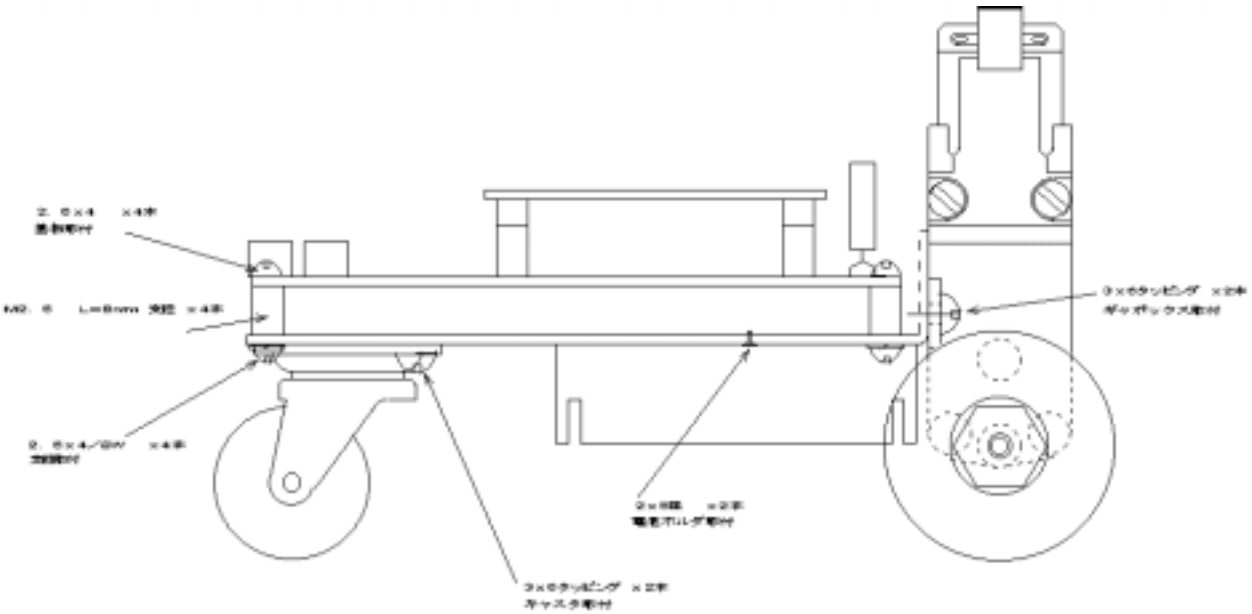
●拡大してご覧下さい。

モータ　配線



付録－５ シャシー組み立て図

●拡大してご覧下さい



2×6段	2本	電池ホルダ
2．6×4 +ナベ	2本	基板取付支柱
2．6BW（スプリングワッシャ）	4本	基板取り付支柱シャーシ側
3×8 +ナベ タッピング	4本	ボックス、キャスタ取付
M2．6 L8mm 支柱	4本	基板取付

シャシー組立図

付録－６ ＲＳ－２３２Ｃ結線図

付録－７ 部品表

記　号	型　番	ﾒｰｶｰ	数量	備　　考
IC1	NJM2902	JRC	1	4ヶ入 OP AMP（相当品）
IC2,3	TA8440H	東芝	2	ﾓｰﾀﾄﾞﾗｲﾊﾞ IC
IC4	NJM2930L05	JRC	1	5V 3端子ﾚｷﾞｭﾚｰﾀ （相当品）
PH1-3	RPR-220	ﾛｰﾑ	3	反射型ﾌｫﾄｾﾝｻ（相当品）
D1-6	SB0030-04A	ｻﾝﾖｰ	6	ｼｮｯﾄｷﾊﾞﾘｱﾀﾞｲｵｰﾄﾞ（相当品）
TR1	2SC1815GR	東芝	1	ﾄﾗﾝｼﾞｽﾀ（相当品）
R1,6	RD16S-220	ｺｰｱ	2	220 ｵｰﾑ炭素皮膜抵抗（相当品）
R2	100	"	1	100 ｵｰﾑ炭素皮膜抵抗（相当品）
R3,4,7,8,10,11,15	10K	"	7	10K ｵｰﾑ炭素皮膜抵抗（相当品）
R5,9,12,	220K	"	3	220K ｵｰﾑ炭素皮膜抵抗（相当品）
R13	6.8K	"	1	6.8K ｵｰﾑ炭素皮膜抵抗（相当品）
R14	15K	"	1	15K ｵｰﾑ炭素皮膜抵抗（相当品）
R16,17,18	100K	"	3	100K ｵｰﾑ炭素皮膜抵抗（相当品）
R19,20	22K	"	2	22K ｵｰﾑ炭素皮膜抵抗（相当品）
JP1-4	Z16	"	4	0 ｵｰﾑ抵抗(相当品)又はﾘｰﾄﾞ線
C1	SS1C475M	ｴﾙﾅ	1	4.7u/16v ﾀﾝﾀﾙｺﾝﾃﾞﾝｻ（相当品）
C2-11,13,14,16,17	RPE132F104Z50	村田	14	0.1u/50v ｾﾗﾐｯｸｺﾝﾃﾞﾝｻ（相当品）
C12	SRA25VB100M	ﾆｯｹﾐ	1	100u/25V 電解ｺﾝﾃﾞﾝｻ（相当品）
C15	SRM25VB100M	"	1	100u/25V 電解ｺﾝﾃﾞﾝｻ（相当品）
SW1	MS-611A	ﾐﾔﾏ	1	ﾄｸﾞﾙｽｲｯﾁ（相当品）
CN1,3	OX-114-DS-40G	OAKS	2	ｺﾈｸﾀ（相当品）
CN2	OX-114-DS-20G	"	1	ｺﾈｸﾀ（相当品）
CN4	OX-114-SS-6G	"	1	ｺﾈｸﾀ（相当品）
CN5	OX-114-SS-2G	"	1	ｺﾈｸﾀ（相当品）
電池ﾎﾙﾀﾞ	MC-304-3 (UM3x4)	ﾀｶﾁ	1	電池ﾎﾙﾀﾞ（相当品）
ﾊｳｼﾞﾝｸﾞ	H2P-SHF-AA	日圧	1	電池ﾎﾙﾀﾞ用ｺﾈｸﾀ（相当品）
ｺﾝﾀｸﾄ	SHF-001T-0.8SS	"	2	電池ﾎﾙﾀﾞ用ｺﾝﾀｸﾄ（相当品）
J1 (ﾎﾟｽﾄ ﾄｯﾌﾟ型)	B2P-SHF-1AA	"	1	ｺﾈｸﾀ（相当品）
J2 (ﾎﾟｽﾄ ｻｲﾄﾞ型)	BS3P-SHF-1AA	"	1	ｺﾈｸﾀ（相当品）
ｼｬｰｼｰ	ATK-9146-05		1	
ｷｬｽﾀ	NO.420-G-N25-1	ﾄﾁｷﾞﾔ	1	
ﾂｲﾝﾓｰﾀｰｷﾞﾔｰﾎﾞｯｸｽ	No.97	ﾀﾐﾔ	1	
ﾄﾗｯｸﾀｲﾔｾｯﾄ(36mm)	No.101 (4ヶ入)	"	2/4	
PCB	9164		1	

ご注意

（１） 本書の内容の一部または全部を無断転記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、万一ご不審な点や誤り、記載洩れなどお気付きなことがありましたら、お買い求めの販売代理店にご連絡下さい。
（４） 運用した結果の影響については（３）項にかかわらず責任を負いかねますのでご了承下さい。

● 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) デバッガ「ＫＤ３０　ルネサステクノロジ社
(3) コンパイラ「ＮＣ30ＷＡ Ver4.00 Release 2 Entry」 ルネサステクノロジ社
(4) フラッシュライタ「Ｍ１６ＣＦｌｓｈ」　ルネサステクノロジ社
(5) 統合開発環境「ＴＭ」　ルネサステクノロジ社

● 商標

ＯＡＫＳ１６－ＳＥＮＳＯＲ ＬＡＢＯ ユーザーズマニュアル

２００３年３月２３日　第一冊発行
２００３年４月　１日　第一冊改訂版発行

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