

ライトレースロボット

OAKS - LABO

デモプログラム(Oklabo_3 & Oklabo_4)の使い方

Ver2.0

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. デモプログラムの構成	2
2. デモプログラムの走行	3
(1) モード選択スイッチの割り当てと機能	3
(2) 走行テスト手順	4
3. デモプログラムの概要	7
(1) ソフトウェアタイマ方式	7
4. T M 統合化開発環境での使い方	9
(1) インストールと注意点	9
(2) エントリー版 (新バージョン) の仕様について	10
(3) T M 統合化開発環境の使い方	11
5. 割り込み処理	13
6. 割り込みベクタテーブルへの登録	14
7. 便利な関数の活用	16

1. デモプログラムの構成

ディレクトリ「Demo2」に入っている「Ok1abo_3」ファイルは、デバugg「KD30」で使用するためのデバugg用ファイルです。また、ディレクトリ「Demo2_ROM」に入っている「Ok1abo_4」ファイルは、フラッシュROMに書き込むためのROM化用ファイル「Ok1abo_4.mot」を作成します。商品には表1で示されるファイルが入っています。

表1 「Demo2」ディレクトリのファイル一覧

ディレクトリ名	ファイル名	内 容
Deno2	Ok1abo_3.c Ok1abo_3.a30 Ok1abo_3.r30 Ok1abo_3.map Ok1abo_3.lst Ok1abo_3.x30 Ok1abo_3.tmk Ok1abo_3.mak M16io.h Lcdtime.h ncrt0.a30 sect30.inc lcc.bat	デバugg用タイマ割込み方式ソースプログラム。 コンパイル出力アセンブラファイル。 アセンブラ出力ファイル。 マップファイル。 リストファイル。 デバugg用出力ファイル。 プロジェクトファイル。 コンパイル用メイクファイル。 I/O記述ヘッダファイル。 LCD表示とタイマ用ヘッダファイル。 スタートアップルーチンファイル。 スタートアップルーチン用インクルードファイル コンパイル用バッチファイル。
Demo2_rom	Ok1abo_4.c Ok1abo_4.a30 Ok1abo_4.r30 Ok1abo_4.map Ok1abo_4.lst Ok1abo_4.x30 Ok1abo_4.mot Ok1abo_4.tmk Ok1abo_4.mak M16io.h Lcdtime.h ncrt0.a30 sect30.inc lcc.bat	ROM化用タイマ割込み方式ソースプログラム。 コンパイル出力アセンブラファイル。 アセンブラ出力ファイル。 マップファイル。 リストファイル。 ***** ROM化用オブジェクトファイル。 プロジェクトファイル。 コンパイル用メイクファイル。 I/O記述ヘッダファイル。 LCD表示とタイマ用ヘッダファイル。 スタートアップルーチンファイル。 スタートアップルーチン用インクルードファイル コンパイル用バッチファイル。

2. デモプログラムの走行

デモプログラム「Ok labo__3」または「Ok labo__4」を使用してOAKS - LABOを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ユーザーズマニュアル付録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トの引っ掛かりなど、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、付属のバッテリーパックの充電は、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、DC 14V ~ 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2のように4つのスイッチの状態によって、速度の設定、オールセンサ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。

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。

スイッチ3番の機能

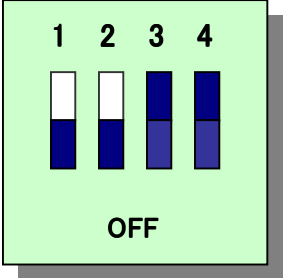
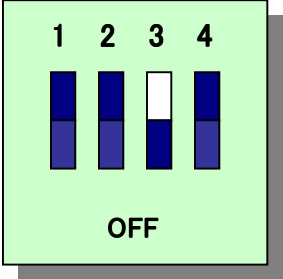
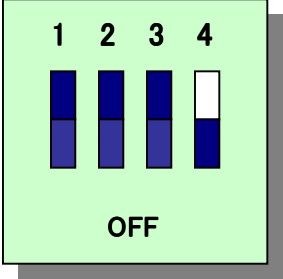
3番のスイッチは、オールセンサ時での動作を選択します。“ON”で使用する場合には、オールセンサの無いコース(交差点の無いコース)で走行して下さい。ストップライン(4つのセンサが同じに消灯(黒ライン走行)または点灯(白ライン走行))で停止します。

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サ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。

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2 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	機 能																				
	<p>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。</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 10%;">1</th> <th style="width: 10%;">2</th> <th style="width: 20%;">LCD 表示</th> <th style="width: 60%;">走行速度</th> </tr> </thead> <tbody> <tr> <td>OFF</td> <td>OFF</td> <td>Lo</td> <td>低速</td> </tr> <tr> <td>OFF</td> <td>ON</td> <td>MI</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> </tbody> </table>	1	2	LCD 表示	走行速度	OFF	OFF	Lo	低速	OFF	ON	MI	中低速	ON	OFF	Mh	中高速	ON	ON	Hi	高速
1	2	LCD 表示	走行速度																		
OFF	OFF	Lo	低速																		
OFF	ON	MI	中低速																		
ON	OFF	Mh	中高速																		
ON	ON	Hi	高速																		
	<p>モード選択スイッチの3番は通過と停止を選択します。</p> <p>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。</p> <p>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。</p>																				
	<p>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。</p> <p>ON・・・白地に黒ラインの場合。 (LCD表示: “Bk”)</p> <p>OFF・・・黒地に白ラインの場合。 (LCD表示: “Wh”)</p>																				

(2)、走行テスト手順

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



図1 メッセージの表示

数秒後に図2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2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2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2 モータ電流の調整表示

スタートスイッチを押すと、図3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。

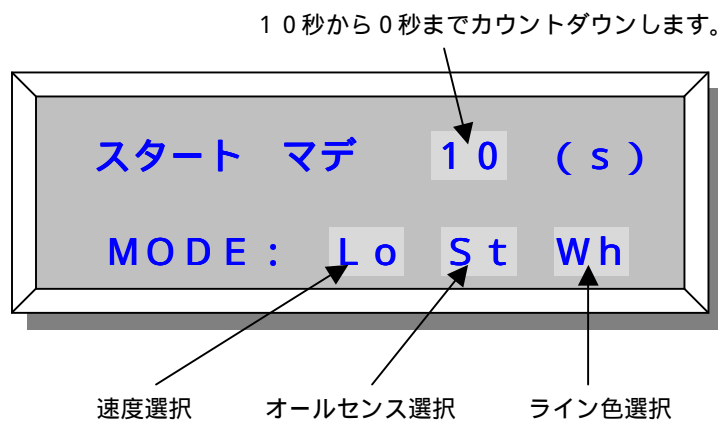


図3 モード表示画面

走行カウンタが0秒になると、 図4 で示される走行表示画面に切り替わります。

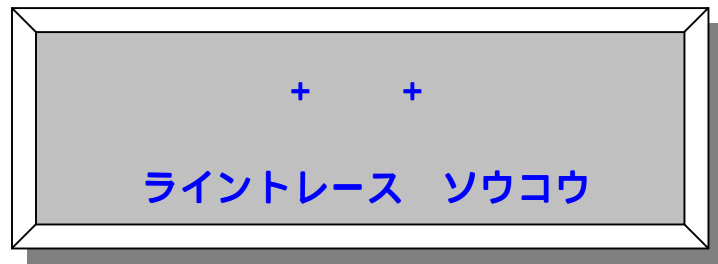


図4 走行表示画面

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5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。 項の画面表示に戻ります。

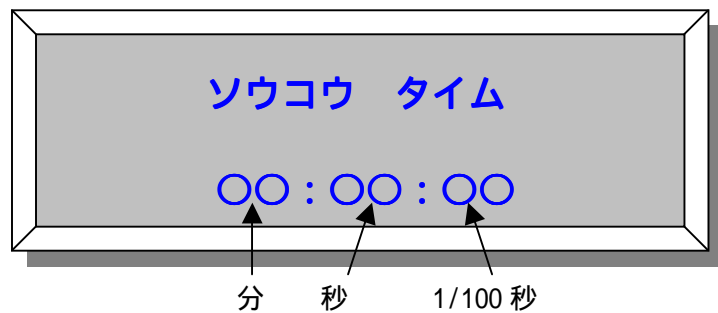


図5 走行表示画面

左接触センサが障害物に接触すると、図6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。 項の画面表示に戻り再走行を開始します。

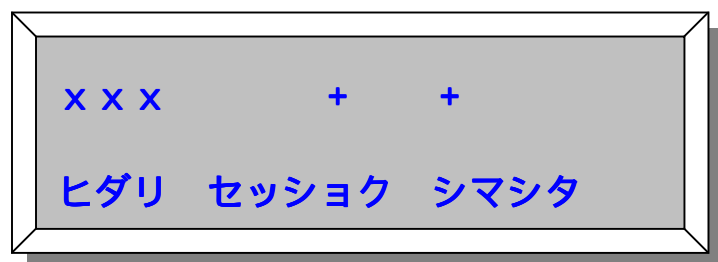


図6 左側接触の表示画面

右接触センサが障害物に接触すると、図7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。頂の画面表示に戻り再走行を開始します。

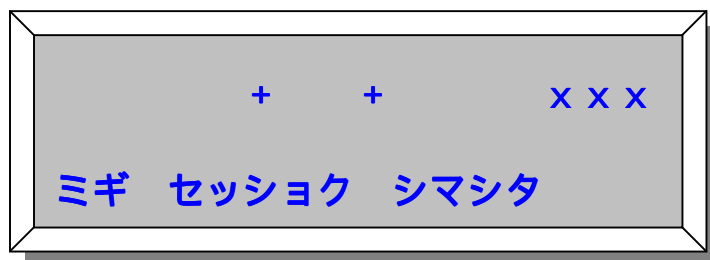


図7 右側接触の表示画面

3. デモプログラムの概要

(1) タイマ割り込み方式

デモプログラム「Ok labo_3」と「Ok labo_4」は、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8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。割り込みのあり、なしの判断に割り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、その割り込み先をモータ駆動にします。時間定数値をセットし、割り込みがある迄待ちます。割り込みが発生し、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。

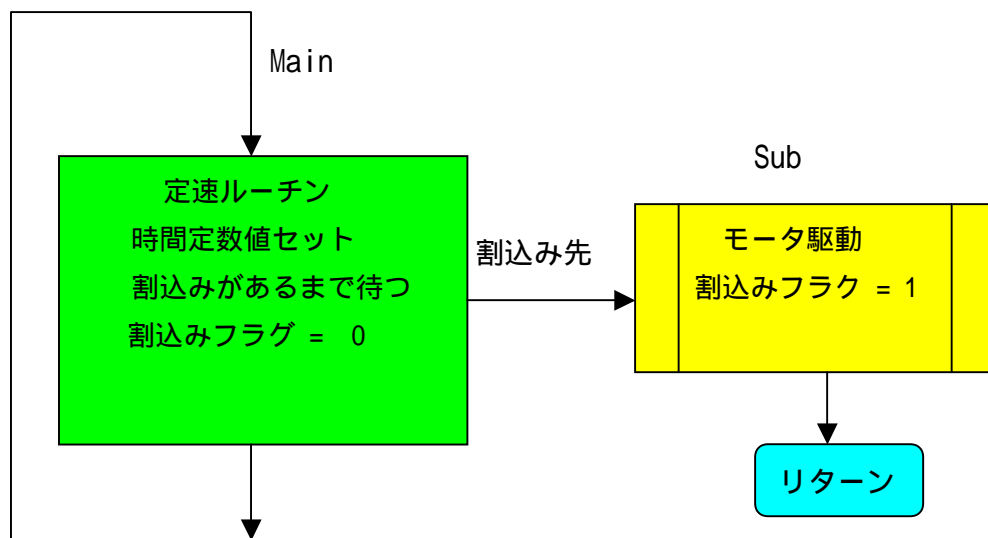


図8 タイマ割り込み方式でのモータ駆動

図9 (a) は、図8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 t a 0 です。また、左モータの時間設定を行うタイマ変数は t a 1 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。右モータ駆動割り込みが終了すると割り込みフラグ「 i l r 」を " 1 " にします。また、左モータ駆動割り込みが終了すると割り込みフラグ「 i f l 」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 " に戻して割り込みを可能にします。

```

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9 (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、方向を自由に変わる事が可能になります。

```
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;
}
```

ta0 > ta1 となるように設定。

(b) 少し右修正ルーチン

図9 タイマ割込み方式のプログラム記述例

4. 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(エンタープライズ版)」が入っています。既に、ライトレースロボット「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」でコンパ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

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3示される語の前にアンダースコア（ ）を付加して下さい。

表3 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
<code>inline</code>	<code><u> </u>inline</code>
<code>near</code>	<code><u> </u>near</code>
<code>far</code>	<code><u> </u>far</code>
<code>asm()</code>	<code><u> </u>asm()</code>

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。

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4 使用できないソフトウェアおよびユーティリティー一覧

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)

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5 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), -fansj, -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6 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10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。

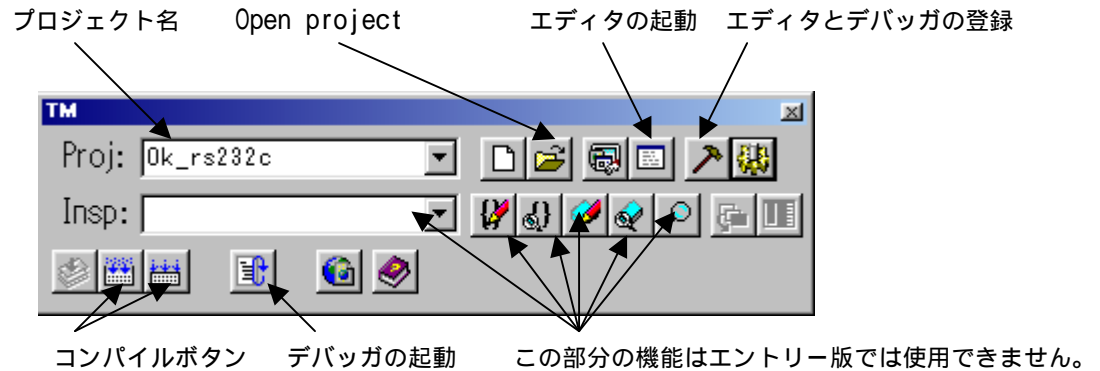





図 1 0 プロジェクトバー


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 1 9 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。

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 1 9 で示される  (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 1 9 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。

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。

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。

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、 (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10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ください。

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。

割り込み処理関数の定義。

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切な名前を付けて、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11は、デモサンプルプログラム「Oakslabo_4」で割り込みに使用する為の割り込み処理関数の記述です。

割り込み関数名

```
*****
* 割り込み関数定義
*****/
void    int0int();          /* INT0 割り込み関数 */
        #pragma INTERRUPT int0int
void    int1int();          /* INT1 割り込み関数 */
        #pragma INTERRUPT int1int
void    intr_count(void);   /* LCDタイマ値割り込み関数 */
        #pragma INTERRUPT intr_count
void    ta3int(void);       /* 秒カウント割り込み関数 */
        #pragma INTERRUPT ta3int
void    rmout(void);        /* 右モータ割り込み関数 */
        #pragma INTERRUPT rmout
void    lmout(void);        /* 左モータ割り込み関数 */
        #pragma INTERRUPT lmout
```

図11 「Oakslabo_4」の割り込みの記述

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。

割り込み処理関数名を疑似命令 “.glb” で外部定義します。

使用する割り込みのダミー関数 “dummy__int” を、割り込み処理関数名に変更します。

図12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」内に割り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      _rmout
```

また、タイマA0を割り込みで使用する場合には、図13に示されるインクルードファイル「sect30.inc」のベクタテーブル見出し名「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   _rmout
```

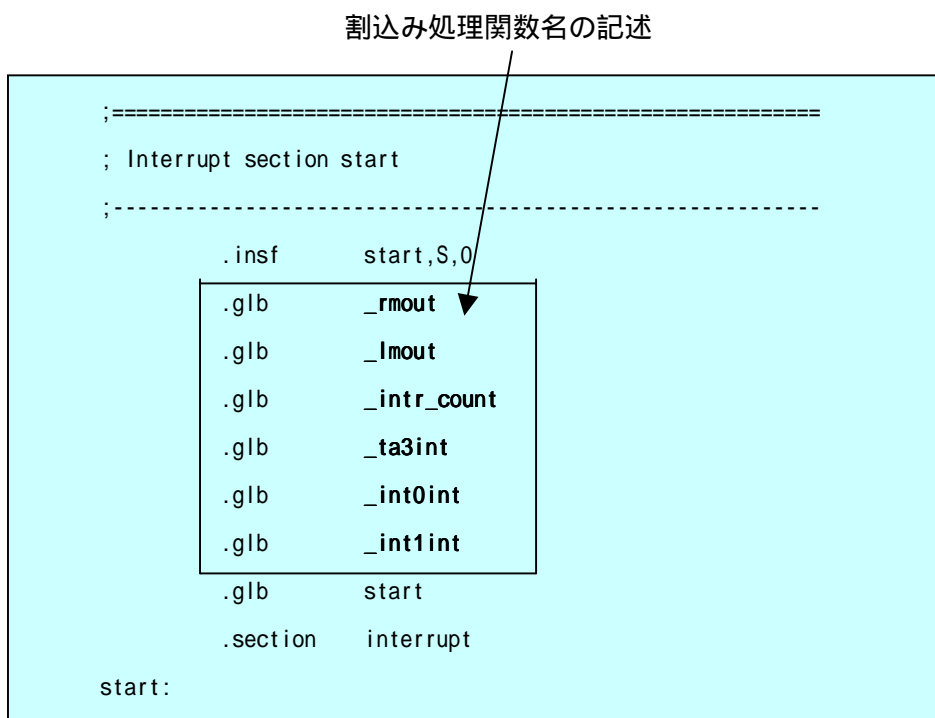


図12 スタートアップルーチンに割り込み処理関数名を記述する例

```

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
    .lword    dummy_int        ; uart0 receive
    .byte     6bh              ; uart1 trance
    .byte     0cbh
    .byte     0fh
    .byte     00h
    .byte     6bh              ; uart1 receive
    .byte     0cbh
    .byte     0fh
    .byte     00h

    .lword    _rmout           ; TIMER A0 タイマ A0
    .lword    _lmout           ; TIMER A1 タイマ A1
    .lword    _intr_count      ; TIMER A2 タイマ A2
    .lword    _ta3int          ; TIMER A3 タイマ A3
    .lword    dummy_int        ; TIMER A4 (vector 25)
    .lword    dummy_int        ; TIMER B0 (vector 26)
    .lword    dummy_int        ; TIMER B1 (vector 27)
    .lword    dummy_int        ; TIMER B2 (vector 28)

    .lword    _int0int         ; INT0 (vector 29)
    .lword    _int1int         ; INT1 (vector 30)
    .lword    dummy_int        ; INT2 (vector 31)

    .endif

```

割込み処理関数名の記述

図 1 3 ベクターテーブルの記述例

7. 便利な関数の活用

LCD表示関数

デモプログラムが使用しているヘッダファイル「`lcdtime_1.h`」にはLCDを表示させるのに便利な関数が記述されています。

表示位置関数「`gotoxy(x, y)`」

表示位置関数の括弧内に記述するXは、表示する桁位置で0から15まで表示位置を記述します。また、Yは、表示する行位置で上段は0、下段は1を記述します。

`gotoxy` (桁位置 , 行位置)

表示関数「`lcdprint` (配列名)

表示関数の括弧内には直接に表示データを記述することはできません。一旦配列で記述してから括弧内に配列名で記述します。

`lcdprint` (配列名)

図13は、OAKS-LABOの電源を投入後、最初に表示するメッセージの表示プログラムです。表示データは「`titl_1`」と「`titl_2`」に配列で記述します。配列「`titl_1`」のデータは1行目0桁位置から表示し、また、配列「`titl_2`」のデータは2行目の0桁位置から表示します。

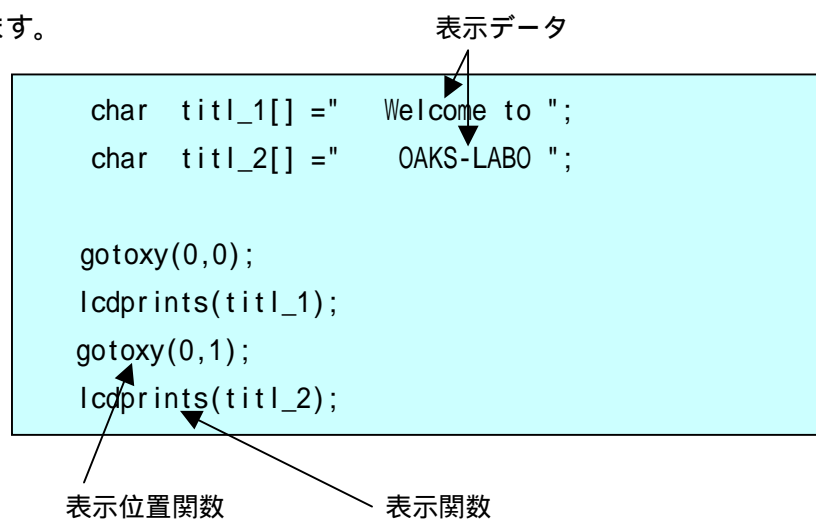


図13 LCDのメッセージ表示例

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の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 Ver4.0」 三菱電機

商標

Windows 95 / 98 / ME / NTは、米国マイクロソフト社の登録商標です。

デモプログラム(Oklabo_3 & Oklabo_4)の使い方

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>
