PIC-BASIC 完成モジュール

AKI-PIC877モジュールに画期的な
ベーシックコンパイラをプロクラムしたモジュール。

PIC-BASIC

Version 1.0

PIC の開発環境に新たに"BASIC"が加わりました。今となっては PIC を知らない人はいないほど、PIC マイコンが普及してきており、開発環境も MPLAB やサードパーティから多数のツールが販売されています。しかし、自分にあったツールがないとか、Visual Basic のように簡単なものがないとかいう人も多いのではないでしょうか？

そういう方にうってつけなのがこの PIC-BASIC です。専用の開発環境とデバッガ、ライターソフトなどが一式揃っていて、短期間で目的のプログラムを作成することができます。

今まで、PIC アセンブラに躊躇していた方や、アセンブラでガリガリ書いていた方にもオススメです。プログラムの書き込みは専用開発環境による、自己書き込み方式を採用していますので、PIC ライター・ハードウェアをお持ちでなくても開発可能です。

※本ソフトウェアは当社の PIC-BASIC モジュールにのみ対応しています。それ以外のボード、PIC16F877 でも BASIC インタプリタが書き込まれていないボードでは動作しませんのでご注意ください。

■動作環境

・Windows95,98,ME,NT,2000,XP （2002/04/01 現在）

・COM ポートの空きが１つ以上

■特長・仕様

◇PIC-BASIC モジュール

言語	PIC 専用オリジナル BASIC
処理方式	BASIC インタプリタ
ＣＰＵ	PIC16F877（超うす型パッケージ） BASIC インタプリタ書き込み済
クロック	20.0MHz
プログラム容量	インタプリタ：約 4k ワード ユーザ領域：約 4k ワード
Ｉ／Ｏポート	トータル 33 ポート 内、最大８本を A/D 入力として使用可能 （RS232C，I2C ポート含む）
A/D コンバータ	分解能：１０ビット 最大８チャンネル
シリアル通信	１チャンネル 最大 115,200bps
動作電圧	５Ｖ～１２Ｖ推奨 ３端子レギュレータ内蔵（バイパスすることも可能）

◇PIC-BASIC 開発環境

変数の種類	BYTE,WORD,LONG 型（いずれも無符号）
配列	１次元配列をサポート
特殊機能	液晶モジュール シリアル通信 EEPROM コントロールなど
書き込み・デバッグの方式	・RS232C によるシリアル通信 ・専用ソフトによる自己書き込み方式を採用 （AKI-PIC プログラマーなどのライターは必要としません）
（インライン）アセンブラの使用	不可
デバッグ機能	ブレークポイント，ウォッチ・ウィンドウ，デバッグ・ウィンドウ，ターミナルウィンドウ

■BASIC モジュールの組み立て

・PIC-BASIC モジュールにヘッダピンを半田付けします。※始めから半田付けされている場合もあります。

◆PIC-BASIC モジュールを単独で使用する場合

・回路図のように RS232C コネクタ、リセットSW、ジャンパー（RUN/PGM）ピンと配線してください。点線枠内はオプションとなっており、PIC-BASIC の動作に必ずしも必要というものではありません。

・右点線枠…液晶モジュールの接続方法
・左点線枠上…大容量 E^2PROM の接続方法
・左点線枠下…LED の接続方法（接続例）
※液晶モジュールと E^2PROM は回路図以外の結線で配線しても正しく動作いたしません。

サンプルプログラムでこれらの機能を使っているものも多いので、接続しておいたほうが PIC-BASIC の機能をより楽しむことができます。

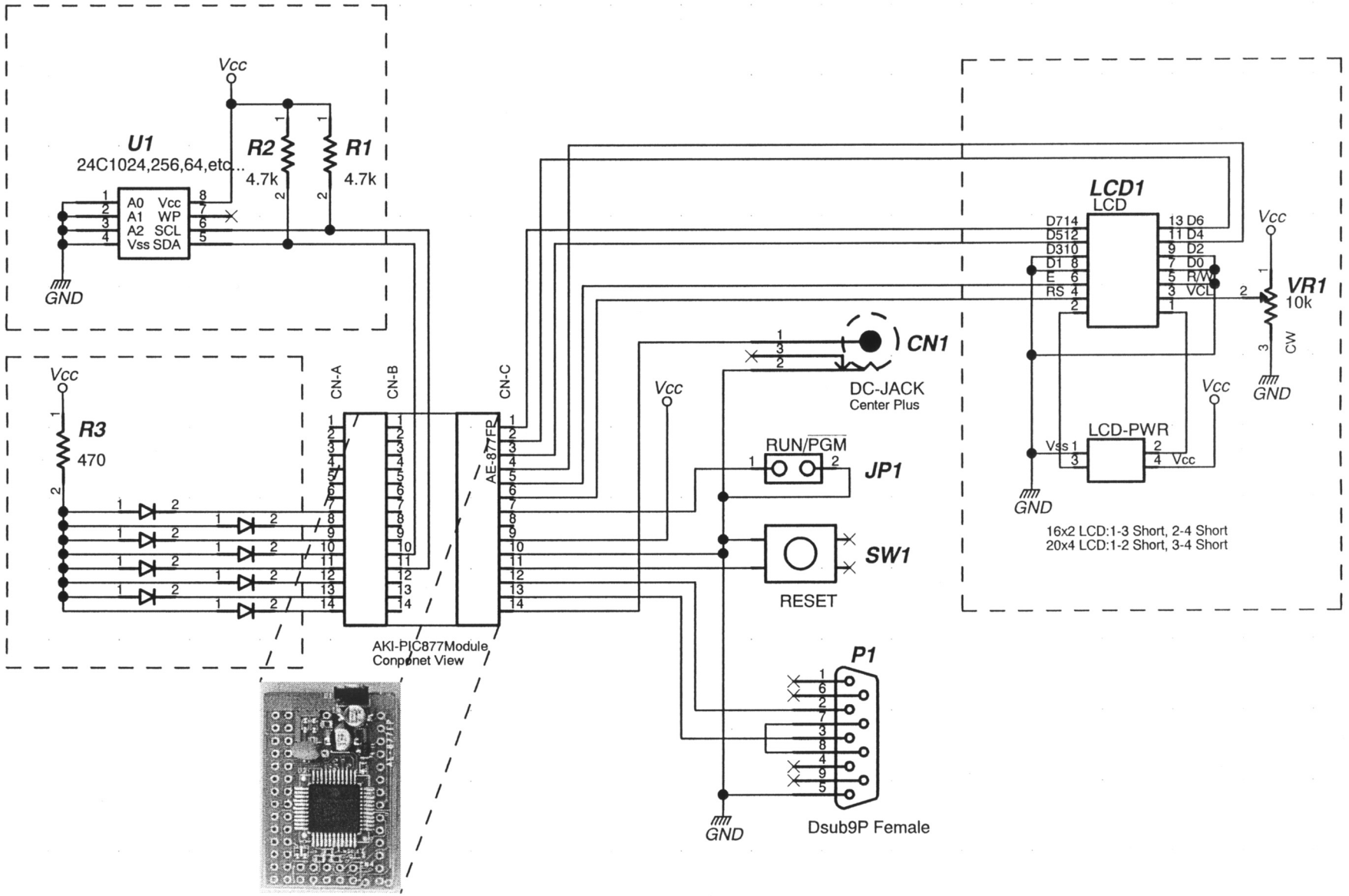
◆PIC-BASIC ベースボードと共に使用する場合

・ベースボートでご利用の方は PIC-BASIC モジュールをベースボートに差し込んでください。

■PIC-BASIC ピンアサイン表

CN-A		CN-B		CN-C	
1	RE0 ポート	1	RA0 ポート	1	RB7 ポート
2	RE1 ポート	2	RA1 ポート	2	RB6 ポート
3	RE2 ポート	3	RA2 ポート	3	RB5 ポート
4	未使用	4	RA3 ポート	4	RB4 ポート
5	未使用	5	RA4 ポート	5	RB3 ポート
6	GND	6	RA5 ポート	6	RB2 ポート
7	RD0 ポート	7	RC0 ポート	7	RB1 ポート
8	RD1 ポート	8	RC1 ポート	8	RB0 ポート
9	RD2 ポート	9	RC2 ポート	9	Vdd(≒5V)
10	RD3 ポート	10	RC3 ポート（EEPROM の SDA へ）	10	GND
11	RD4 ポート	11	RC4 ポート（EEPROM の SCL へ）	11	MCLR（リセットピン）
12	RD5 ポート	12	RC5 ポート	12	TxD（D サブコネクタへ）
13	RD6 ポート	13	未使用	13	RxD（D サブコネクタへ）
14	RD7 ポート	14	未使用	14	PWR（電源端子７～１２Ｖ）

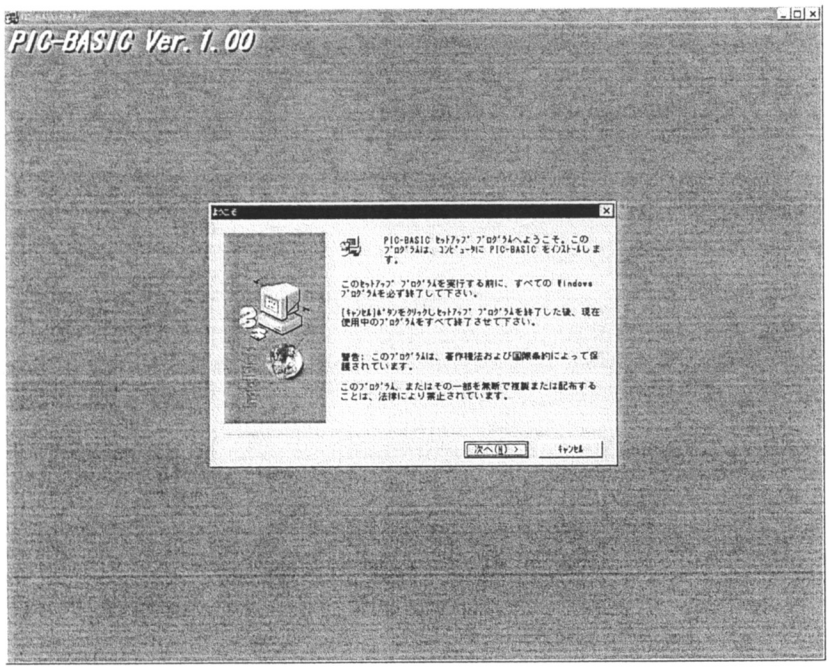
※ＣＮ４，ＣＮ５は PIC-BASIC としては使用しません。


U1
24C1024,256,64,etc...
A0 Vcc
A1 WP
A2 SCL
Vss SDA
Vcc
GND
R2
4.7k
R1
4.7k
R3
470
CN-A
CN-B
CN-C
AE-877FP
AKI-PIC877Module
Conponet View
CN1
DC-JACK
Center Plus
RUN/PGM
JP1
SW1
RESET
P1
Dsub9P Female
LCD1
LCD
D714
D512
D310
D1 8
E 6
RS 4
13 D6
11 D4
9 D2
7 D0
5 R/W
3 VCL
VR1
10k
CW
LCD-PWR
Vss 1
4 Vcc
16x2 LCD:1-3 Short, 2-4 Short
20x4 LCD:1-2 Short, 3-4 Short

■インストール

・CD-ROM をドライブにセットすると自動的にインストーラが起動します。
（自動起動しない場合は CD-ROM 内の setup.exe を実行させてください）

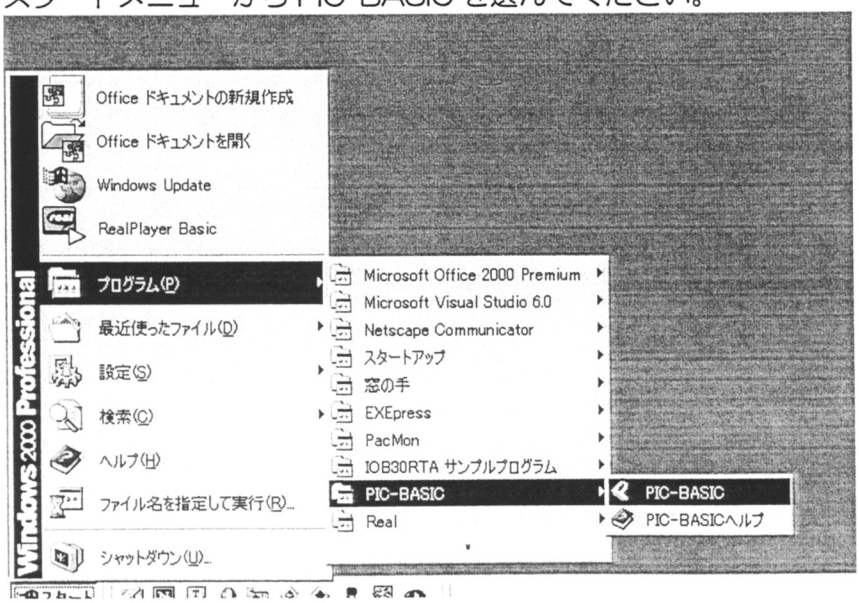
・インストーラが起動すると以下の画面が現われます。



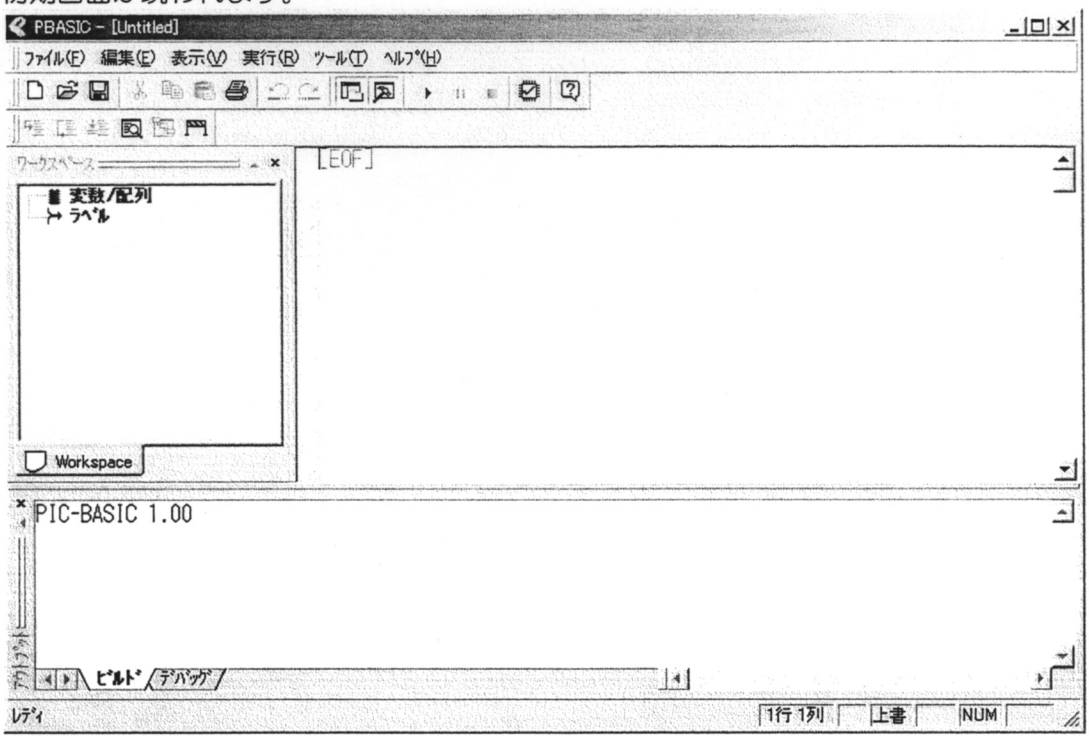
・あとは画面の指示に従って進めていってください。インストールが終了するとスタートメニューに PIC-BASIC が現れます。

■起動

・スタートメニューから PIC-BASIC を選んでください。



■初期画面が現われます。



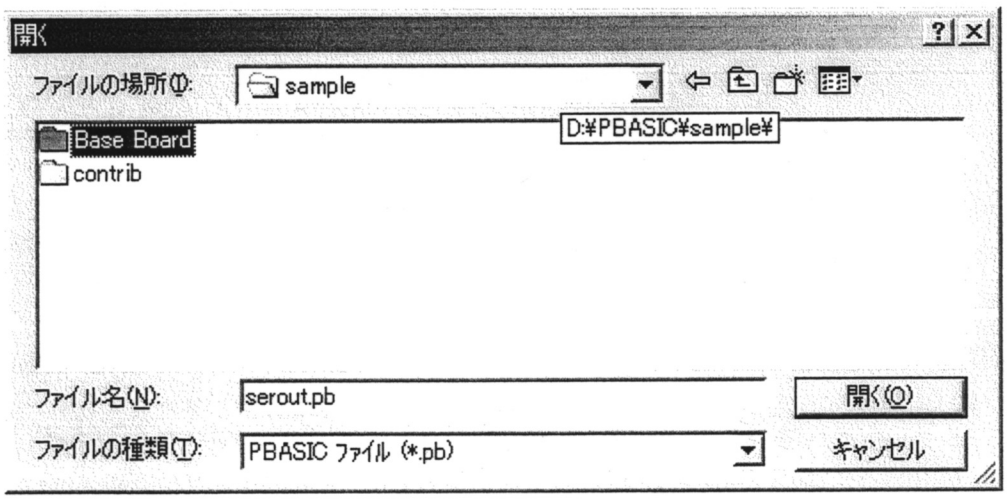
■サンプルプログラムを動かす

■サンプルを動作させてみましょう。

デフォルトのフォルダにインストールした場合は

C:¥Program Files¥PIC-BASIC¥samples

の下にサンプルプログラムが入っています。



・Base Board フォルダは PIC-BASIC ベースボード用のサンプルプログラム集です。contrib フォルダはフリーソフト開発者などから無料で提供されているプログラムです。こちらのプログラムの全てについて、動作確認がとれている訳はありませんが、面白いものもありますのでぜひご覧下さい。

・始めの第一歩として Base Board フォルダの中の demo1.pb ファイルを開いてみます。

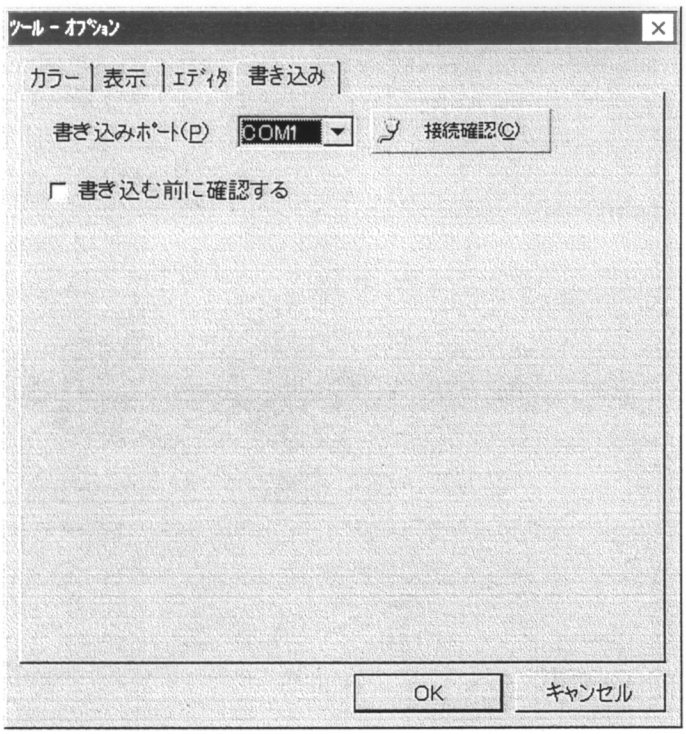
プログラムは出来上がっていますので、これを PIC-BASIC に書き込み（転送）して走らせてみましょう。このサンプルプログラムは液晶モジュールが必要です。

■BASIC モジュールとの接続

RS232C ケーブルで PIC-BASIC とパソコンを接続します。接続ケーブルは普通のＲＳ２３２Ｃストレートのケーブルです。接続したＣＯＭポートを覚えておきます。

RUN/PGM ピンをジャンパー（接続）してベースボードの電源を入れます。

ＣＯＭ１以外に接続されているかたはＣＯＭポートの設定が必要です。
メニューバーの［ﾂｰﾙ(T)］→［ｵﾌﾟｼｮﾝ(O)］を選択します。



上部の書き込みタブを選びます。すると、ＣＯＭポート選択できる画面が現れますので接続したポートを選んでください。このとき［接続確認］ボタンを押して実際に通信できるか確認を行うことができます。問題なければ、［ OK ］を押してダイアログを閉じて下さい。

メニューバーの［実行(R)］から［ﾃﾞﾊﾞｯｸﾞ実行(R)］を選択します。（F5 キーを押しても同じ動作をします）

書き込み確認ダイアログが開きます。ここで［ OK ］を押してみてください。液晶モジュールにメッセージが表示され、ベースボートを利用している方はさらにＬＥＤがついたり・消えたりするのが確認できるはずです。

その他のサンプルが同フォルダに入っていますので、そちらも試してみてください。

■PIC-BASIC Ver.1.0 命令リファレンス

こちらのリファレンスはオンラインヘルプの抜粋です。ＣＤ－ＲＯＭ内にあるヘルプが最新でかつ詳しく書かれております。

◆基本動作

DIM 変数・配列の定義

Dim [変数名] As [変数の型]
Dim [配列名] ([配列要素の個数]) As [変数の型]

一度に複数の変数・変数を定義することもできます。
As [変数の型] は省略可能です。省略すると Word 型で定義したものとみなします。

LET 変数・配列への数値の代入（省略可）

LET 変数 = 式
変数 = 式
LET 配列 (式) = 式
配列 (式) = 式

変数・配列に式の結果を代入します。LET のキーワードは省略して書くことができます。通常は省略して書きます。式には数値や変数を使った計算式を自由に書くことができ、掛け算・割り算が先に計算されます。

◆ループ・条件分岐

FOR...NEXT　　FOR 文

(1)　FOR [変数|配列] = 式 TO 式

　　NEXT {[変数|配列]}
(2)　FOR [変数|配列] = 式 TO 式 STEP 式

　　NEXT {[変数|配列]}
(3)　FOR [変数|配列] = 式 TO 式 STEP - 式

　　NEXT {[変数|配列]}

FOR...NEXT 内のプログラムを指定した回数繰り返します。繰り返し変数には任意の変数を利用できます。
STEP 項の記述がない場合は STEP 1 と同様に処理されます。

WHILE...WEND　　WHILE 文

WHILE 条件
　命令など
WEND

WHILE...END 文は条件が真の間、WHILE～WEND に書かれた命令を実行します。もし、はじめから条件が偽の場合は WHILE～WEND 間のプログラムは一度も実行されず、WEND の後から実行します。

DO...UNTIL　　DO 文

DO
　命令など
UNTIL 条件

DO...UNTIL 文は条件が真になるまで、DO～UNTIL の間に書かれた命令を実行します。条件分岐が最後にあるので、条件に関わらず DO～UNTIL 内に書かれたプログラムは少なくとも１回は実行されます。

IF...THEN　　IF 文

(1)　IF 条件式 THEN 命令 [: 命令]
(2)　IF 条件式 THEN 命令 [: 命令] ELSE 命令 [: 命令]
(3)　IF 条件式 THEN ラベル
(4)　IF 条件式 THEN
　　命令...
　　ENDIF
(5)　IF 条件式 THEN
　　命令...
　　ELSE
　　命令...
　　ENDIF

条件分岐を行います。

GOTO　　GOTO 文

GOTO [ﾗﾍﾞﾙ]

プログラムを無条件に分岐します。GOTO の後にラベルを記述します。

GOSUB...RETURN　　GOSUB...RETURN 文

GOSUB [ﾗﾍﾞﾙ]

サブルーチンを呼び出します。GOSUB の後にラベルを記述します。ラベルの後の : (コロン) は不要です。

RETURN

サブルーチンからリターンします。

◆Ｉ／Ｏ制御　ディジタル入出力の制御

OUTPUT　　ピンを出力に変更

OUTPUT { RA | RB | RC | RD | RE } . [0～7]

指定したピンを出力ピンにします。出力ピンにした時点でHになるかLになるかは RA,RB,RC,RD,RE に設定されている値によって異なります。

INPUT　　ピンを入力に変更

INPUT { RA | RB | RC | RD | RE } . [0～7]

指定したピンを入力ピンにします。指定したピンはハイインピーダンス(Z)になります。この命令を実行すると RA,RB,RC,RD,RE を使ってI/Oピンの状態を読み取れるようになります。

HIGH　ピンをハイ(H)にする

HIGH { RA | RB | RC | RD | RE } . [0～7]

指定したピンを出力ピンにして、ハイ"H"を出力します。
現在入力ピンになっていても、この命令によって(指定されたピンのみ)自動的に出力に変更されます。

LOW　ピンをロー(L)にする

LOW { RA | RB | RC | RD | RE } . [0～7]

指定したピンを出力ピンにして、ロー"L"を出力します。
現在入力ピンになっていても、この命令によって(指定されたピンのみ)自動的に出力に変更されます。

TOGGLE　ピン出力の入れ替え(H)⇔(L)

TOGGLE { RA | RB | RC | RD | RE } . [0～7]

指定したピンの状態を反転します。現在がHならLに、LならHにします。
指定したピンは出力ピンになっていないと、何も起きません。

REVERSE　入力⇔出力ピンの入れ替え

REVERSE { RA | RB | RC | RD | RE } . [0～7]

指定したピンの入力・出力を反転します。現在が入力なら出力に、出力なら入力にします。

◆アナログ　アナログデータの取り込み

ADC　ＡＤ変換

ADC [変換ﾁｬﾝﾈﾙ], [変換ﾓｰﾄﾞ], [変数]

CPU内蔵のADコンバータを使ってアナログ→ディジタル変換します。
分解能は10ビットです。変換結果は0～1023までの範囲となります。精度は電源電圧や電源リップル，リファレンス精度、入力インピーダンスによって変化します。
変換ﾁｬﾝﾈﾙ,変換ﾓｰﾄﾞについてはオンラインマニュアルをご覧ください。

◆シリアル　ＲＳ２３２Ｃ通信用

SERIN　RS232C データの受信

SERIN [ﾎﾞｰﾚｰﾄ定数], [ﾀｲﾑｱｳﾄ時間(ms)], [変数] { , [変数]... }

RS232C データの受信を行います。通信条件はデータ8ビット，ストップ1ビット，パリティなしで、ボーレートは引き数で変えることができます。
受信したデータは引き数で与えた変数に順にセットされます。
タイムアウト時間内に全てのデータが受信できなかったときは受信処理を中断し、処理が戻ります。タイムアウト時間は引き数で与えた変数それぞれのタイムアウト時間ではなく、変数全てを受信する時間を表しています。受信できなかった変数は命令を実行する前の変数値をそのまま持っています。どこまで受信できたかは変数値を見て判断してください。
ﾎﾞｰﾚｰﾄ定数についてはオンラインマニュアルをご覧ください。

SEROUT　RS232C データの送信

SEROUT [ﾎﾞｰﾚｰﾄ定数], [変数] { , [変数]... }

RS232C データの送信を行います。通信条件はデータ8ビット，ストップ1ビット，パリティなしで、ボーレートは引き数で変えることができます。
データ引数で与えた順に送信されます。
ﾎﾞｰﾚｰﾄ定数についてはオンラインマニュアルをご覧ください。

SERCLEAR RS232C 受信バッファのクリア

SERCLEAR

RS232C の受信ﾊﾞｯﾌｧをクリアします。PIC-BASIC モジュールはリセット直後や、データ通信中や直後などにゴミデータを受信してしまうことがあり、データの先頭文字が正しく受け取れないことがあります。このような場合はこの命令を使って受信ﾊﾞｯﾌｧをクリアして、次に届くRS232C データを確実に受信できるようにします。

◆液晶モジュール　液晶モジュール用

INITLCD　液晶モジュールの初期化

INITLCD

引数はありません。キャラクタタイプの液晶モジュールを初期化します。推奨液晶モジュールは秋月電子で販売されているSC1602BS*B です。他社製品の場合はピンアサインや電源極性が逆の場合はありますのでデータシートで確認してください。AKI-PIC877 ベースボードを利用すると 16×2行,20×4行の両方に対応した電源パターンが引かれていますので便利です。
液晶モジュールを使用する場合は下図のように接続するI／Oピンが予め決められています。INITLCD 命令を実行すると液晶に接続したピンは全て出力ピンに自動的に変更されます。以降、液晶モジュールを正しく動作させるにはピンを入力にしたり、出力を変化させたりしないでください。正しくコントロールできなくなります。

CLEARLCD　液晶モジュールの表示クリア

CLEARLCD

引数はありません。キャラクタタイプの液晶モジュールの表示をクリアして、カーソルを左上に移動させます。この関数を利用する前にINITLCD を使って液晶モジュールを初期化しなければいけません。

PUTLCD　文字・数値の表示

PUTLCD [文字列 | 式 | CHR$(式)] [, [文字列 | 式 | CHR$(式)]]...

現在のカーソル位置から液晶モジュールに任意の文字や数値を表示させる命令です。この関数を利用する前に INITLCD を使って

液晶モジュールを初期化しなければいけません。引数には文字列や変数・式を混在させることが可能です。文字列はそのまま液晶に表示され、変数の値は10進数に変換されてから表示されます。

変数の表示桁数は変数の値によって変化します。例えば変数値が 10 のときは2ケタ、9000000 の時は7ケタの文字が液晶に表示されます。

詳細はオンラインマニュアルをご覧ください。

SETPOS　　カーソル位置変更

SETPOS [式],[式]

液晶モジュールのカーソル位置を引き数で与えた位置に移動します。第1引数がX座標、第2引数がY座標です。

16×2行の液晶モジュールで指定できる座標の範囲は(0,0)～(15,1)です。存在しない座標を指定した場合、X座標は16で割った余り、Y座標は4で割った余りの座標が指定されたものとして動作します。

20×4行の液晶モジュールで指定できる座標の範囲は(0,0)～(19,3)です。

HOMELCD　　カーソルをホーム位置に移動

HOMELCD

液晶モジュールのカーソル位置を(0,0)[→画面左上]に移動します。画面表示はクリアされません。

◆スリープ・ウェイト

SLEEP　　指定時間の待機（ウェイト）

SLEEP [式]

指定した時間だけプログラムの進行を遅らせます。ウェイトさせるための命令です。

ウェイト時間はms(1／1000秒)単位で指定します。

式には定数や変数・演算式などを記述することができます。

0を与えられるとウェイトせずすぐに次の文を実行します。

END　　プログラムの終了

END

プログラムを終了します。プログラムを終了したいポイントに記述します。

特に何も書かなくても、プログラムが最終行に到達すると END を実行したものと同じように処理されます。

END 文は実際には無限ループに入るだけの命令です。

一度 END 命令を実行したら、リセットするまでプログラムは一切実行されません。

◆テーブル参照

LOOKUP　　テーブル参照

LOOKUP [式], [変数], [値1] { , [値2] , [値3] , ... }

([式]の計算結果＋1)番目の値を [変数]に代入します。

[式]の結果が 0 なら[値1]を、1 なら[値1]を、同様に 2 なら[値2]を[変数]にセットします。

該当する値nがない場合は何も変数には代入しません。命令を実行する前の値を保持したままです。

◆EEPROM

READ　　内蔵 EEPROM の読み出し

READ [ｱﾄﾞﾚｽ], [変数] { , [変数]... }

PIC 内蔵の EEPROM からデータの読み込みを行います。

ｱﾄﾞﾚｽは EEPROM のアクセス開始アドレスです。指定したアドレスからデータを読み込みます。開始アドレスの指定範囲は&H00～&HFF(0~255)までです。

変数には読み込んだデータを入れる変数名あるいは配列名を指定します。複数個記述することもできます。

変数の型に注意:

書き込んだ時の変数の型と読み込んだ時の変数の型が一致しないと、全く異なる値を返すことがあります。1つの変数を1バイトで保存するのか、2バイトで保存するのか、あるいは4バイトで保存するのかによって EEPROM の保存アドレスがずれてくるためです。読み書きは同じ変数型で行うようにしてください。

WRITE　　内蔵 EEPROM への書き込み

WRITE [ｱﾄﾞﾚｽ], [変数] { , [変数]... }

PIC 内蔵の EEPROM へデータを書き込みます。

ｱﾄﾞﾚｽは EEPROM のアクセス開始アドレスです。指定したアドレスからデータを読み込みます。開始アドレスの指定範囲は&H00～&HFF(0~255)までです。

変数には書き込みたい値が入った変数名あるいは配列名を指定します。複数個記述することもできます。

書き込みは1バイト当たり約10ms掛かります。

変数の型に注意:

書き込んだ時の変数の型と読み込んだ時の変数の型が一致しないと、全く異なる値を返すことがあります。1つの変数を1バイトで保存するのか、2バイトで保存するのか、あるいは4バイトで保存するのかによって EEPROM の保存アドレスがずれてくるためです。読み書きは同じ変数型で行うようにしてください。

◆I2C　　I2C通信

I2CREAD　　I2C EEPROM の読み出し

I2CREAD [ｺﾝﾄﾛｰﾙ], [ｱﾄﾞﾚｽ], [変数] { , [変数]... }

I2C バスからシリアル EEPROM の読み込みを行います。I2C EEPROM をモジュールの SDA,SCL に接続し、プルアップを必ず行ってください。

ｺﾝﾄﾛｰﾙは EEPROM のコントロールバイトのことで、2進数で1010□□□0を指定します。□部分は EEPROM のアドレスを示し、複数の EEPROM を SCL,SDA に接続した場合にどのデバイスかを選択するためのものです。3ビットありますので、EEPROM を最大8個まで制御することができます。

AKI-PIC877 ベースボードではボード上でアドレス&B000 に固定されていますので、&B10100000(=&HA0)を指定すれば、読み書きできます。

ｱﾄﾞﾚｽは EEPROM のアクセス開始アドレスです。指定したアドレスからデータを読み込みます。開始アドレスの指定範囲は外付け

の EEPROM によって異なります。
・24C64(24LC64)　&H0000～&H1FFF
・24C256(24LC256) &H0000～&H7FFF
・24C1024　　&H0000～&HFFFF

変数には読み込んだデータを入れる変数名あるいは配列名を指定します。複数個記述することもできます。

変数の型に注意：
書き込んだ時の変数の型と読み込んだ時の変数の型が一致しないと、全く異なる値を返すことがあります。1つの変数を1バイトで保存するのか、2バイトで保存するのか、あるいは4バイトで保存するのかによって EEPROM の保存アドレスがずれてくるためです。読み書きは同じ変数型で行うようにしてください。

※この I2CREAD コマンドは I2C バス EEPROM の読み書きを前提としています。その他の I2C デバイスでは正しく動かないことがあるかもしれませんのでご了承ください。
詳細はオンラインマニュアルをご覧ください。

I2CWRITE　　I2C EEPROM への書き込み

I2CWRITE [ｺﾝﾄﾛｰﾙ], [ｱﾄﾞﾚｽ], [変数] { , [変数]... }

I2C バスからシリアル EEPROM へ書き込みを行います。I2C EEPROM をモジュールの SDA,SCL に接続し、プルアップを必ず行ってください。

ｺﾝﾄﾛｰﾙは EEPROM のコントロールバイトのことで、2進数で1010□□□0を指定します。□部分は EEPROM のアドレスを示し、複数の EEPROM を SCL,SDA に接続した場合にどのデバイスかを選択するためのものです。3ビットありますので、EEPROM を最大8個まで制御することができます。
AKI-PIC877 ベースボードではボード上でアドレス&B000 に固定されていますので、&B10100000(=&HA0)を指定すれば、読み書きできます。

ｱﾄﾞﾚｽは EEPROM のアクセス開始アドレスです。指定したアドレスからデータを読み込みます。開始アドレスの指定範囲は外付けの EEPROM によって異なります。
・24C64(24LC64)　&H0000～&H1FFF
・24C256(24LC256) &H0000～&H7FFF
・24C1024　　&H0000～&HFFFF

変数には書き込みたい値が入った変数名あるいは配列名を指定します。複数個記述することもできます。

変数の型に注意：
書き込んだ時の変数の型と読み込んだ時の変数の型が一致しないと、全く異なる値を返すことがあります。1つの変数を1バイトで保存するのか、2バイトで保存するのか、あるいは4バイトで保存するのかによって EEPROM の保存アドレスがずれてくるためです。読み書きは同じ変数型で行うようにしてください。

※命令の大文字・小文字は区別しません。

※この I2CWRITE コマンドは I2C バス EEPROM の読み書きを前提としています。その他の I2C デバイスでは正しく動かないことがあるかもしれませんのでご了承ください。
詳細はオンラインマニュアルをご覧ください。

◆レジスタ制御

POKE　　ファイルレジスタ書き込み

POKE [addr], [value]

PIC 内部のファイルレジスタへアクセスして、そのレジスタに value の値を書き込みます。addr には&H000～&H1FF までの範囲を指定します。アドレスとそのレジスタの機能については PIC16F877 のデータブックをご覧ください。
この命令は主に PIC-BASIC の命令でできないような処理を行わせる場合に使われるものです。不要に使うと PIC-BASIC の命令が正しく動かなくなる可能性があるので、注意してください。

PEEK　　ファイルレジスタ読み込み

var = PEEK(addr)

PIC 内部のファイルレジスタへアクセスして、そのレジスタの値を読み込みます。addr には&H000～&H1FF までの範囲を指定します。アドレスとそのレジスタの機能については PIC16F877 のデータブックをご覧ください。
この命令は主に PIC-BASIC の命令でできないような処理を行わせる場合に使われるものです。不要に使うと正しくプログラムが動かなくなる可能性があるので、注意してください。
変数の戻り値の型は Byte 型です。(返却される値は0～255です)

◆デバッグサポート

DEBUG　　デバッグ命令

DEBUG [文字列 | 式], ...

デバッグのための命令です。PIC-BASIC デバッグ中にこの命令を実行すると、引数で与えた文字列・変数などをPCのデバッグウィンドウに表示します。引数の個数は可変長で容量が許す限り、いくつでも記述できます。文字列はそのままPC側に表示されますが、変数は10進数に変換されます。
10進数以外を表示させたい場合は CHR$( )を使います。

DEBUG 文ごとに表示が改行されます。改行せずにデバッグウィンドウに表示させたい場合は一文で記述してください。
デバッグ以外の時、この命令は何もしません。

■プログラミングのヒント

・変数（配列）はその変数（配列）を使う前に定義してあればどの行でも構いませんが、プログラムがわかりにくくなるので、プログラムの始めにまとめて記述したほうが良いです。

・For...Next や While...Wend, Do...Until といったループ命令の中はインデントした方が、コンパイルエラーを見つけやすく、また論理ミスを少なくすることができます。同様に If 文もそうしたほうが解りやすいです。

・TRIS_RE ポートの上位５ビットは PSP（Parallel Slave Port）のレジスタと共用になっています。そのため、ビット１を書き込むと PSP が有効になり、動作がおかしくなることがあるので注意しましょう。

・液晶関連の命令は RB ポートを勝手に操作します。そのため、液晶モジュールを接続しながら、RBポートを制御するプログラムは書かないほうが懸命です。同様に RS232C や外付け EEPROM で使用している I/O ピンも操作しないほうが安定して動作します。

・2/3 倍するとか、1.5 倍するというような場合は掛け算と割り算等を併用することで計算することができます。（数値が変数の表現範囲を超えないように注意してください）

- 2/3 倍

６６６７を掛けて、１００００で割る

```
a = a * 6667 / 10000
```

- 1.5 倍

その数自身にその１／２を加算する。

```
a = a + a / 2
```

・液晶モジュールへの出力は数値に応じて桁数が変わるので、前回の表示が消されないことがあります。例えば、12345 を表示した後、99 を表示したような場合は前の表示と重なって 99345 のような表示になってしまいます。そのときは末尾にスペースを出力すると正しく読み取ることができます。

```
Putlcd number,"      "
```

・PIC-BASIC で小数を扱うことはできませんが、近似的に処理をして小数のように見せかけることは可能です。

```
' 1667 という数値を "1.667V" と表示させるプログラム
Dim A
A = 1667
Putlcd A / 1000, ".", (A / 100) MOD 10, (A / 10) MOD 10, A MOD 10, "V"
```

・変数のビットを参照することはできますが、変数のビットを変更することはできません。

```
a = RA.Bit4
```

というのはできますが、

```
RA.Bit4 = a
```

というのはできません。それを実現するには

```
If a<>0 then Ra = Ra | &H10 Else Ra = Ra & (&H10 ^ &HFF)
```

と書く必要があります。対象ビットが Ra～Re のポートであれば次のように書くことも可能です。

```
If a<>0 then High Ra.Bit4 Else Low Ra.Bit4
```

■よくある質問

Q. PIC-BASIC モジュール以外の PIC16F877 で動作しますか？
A. いいえ。PIC-BASIC インタプリタが書き込まれていない PIC16F877 では動作致しません。

Q. プログラムは何回書き込むことができますか？
A. PIC-BASIC モジュールに使っているマイコン PIC16F877 の書き込み寿命に依存します。PIC16F877 の書き込み寿命は最低１,０００回が保証されています。１,０００回以上のプログラム書き換えが可能なのですが、書き込み以外にデバッグ中にも裏で書き込みを行っていますので、実際はそれよりも少なく見える場合があります。

Q. PIC-BASIC で開発したＰＩＣマイコンを 20MHz 以外の周波数で動作させることはできますか？
A. クロック周波数 20MHz に依存していますので、20MHz 以外では動作しません。

Q. 自分の開発した BASIC プログラムにプロテクトは掛けられますか？
A. プロテクトは掛けられません。

Q. A/D 変換値がおかしい
A. PIC マイコンの回路上、電源にダイオードが入っています。レギュレータの出力から、そのドロップ(約 0.3V)を引いた電圧が全体の動作電圧（＝Ａ／Ｄ変換のリファレンス）になるためです。プログラムで補正するか、外部リファレンス電圧を活用してください。

Q. 液晶に文字が出ません。
A. コントラストを右回りに回して（濃くして）ください。

Q. ファームウェア（PIC16F877 に書かれた BASIC インタプリタ）はバージョンアップできますか？
A. ＰＣ側の開発環境はバージョンアップ可能ですが、ファームウェアはバージョンアップできません。

Q. PIC-BASIC モジュールに書き込まれているインタプリタを消すことはできますか？
A. はい。インタプリタはプロテクトが掛かっておりますが、PIC プログラマーキットなどを使って削除すること（生の PIC16F877 の IC にすること）は可能です。但し、一度消してしまったインタプリタを復旧することはできません。

■免責事項

本キット（ソフトウェア，ハードウェア，ファームウェアを含む）は全てのソフトウェア・ハードウェア開発・動作環境において、必ずしも動作を保証するものではありません。
また、本ソフトウェア製品の使用または使用不能から生じた直接的または間接的損害に対し、一切の責任を負いません。予めご了承ください。

■当キットについての問い合わせ・質問は下記ホームページにお願いいたします。メールが出来ない場合は封書，往復はがきでも結構です。

http://picbasic.jp/

〒150-0036
東京都渋谷区南平台町 2-12　久保ビル７Ｆ（７０３）
株式会社　エスアイ創房

PIC-BASIC Ver.1.00 マニュアル第１版
2002/04/01