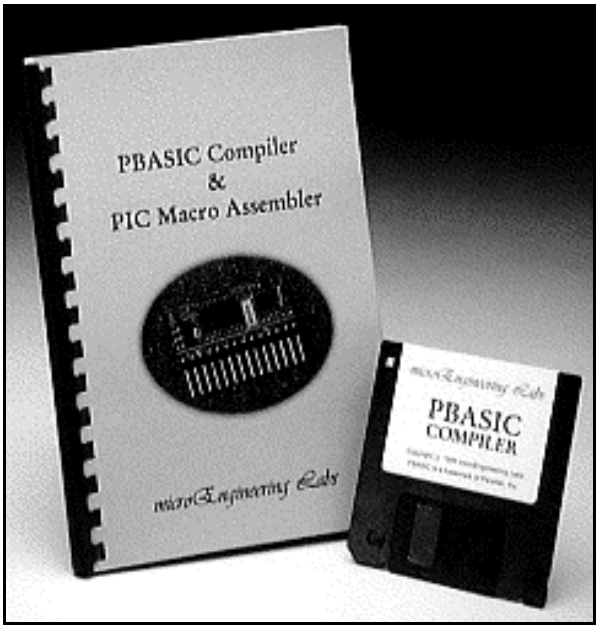
PIC Basic Compiler

- BASIC COMPILER FOR PIC MICROCONTROLLERS
- EXPANDED BASIC STAMP I (PBASIC 1*) COMPATIBLE INSTRUCTION SET
- TRUE COMPILER PROVIDES FASTER PROGRAM EXECUTION AND ENABLES LONGER PROGRAMS THAN BASIC INTERPRETERS
- I^2CIN AND I^2COUT INSTRUCTIONS TO ACCESS EXTERNAL SERIAL EEPROMS AND OTHER I^2C DEVICES
- PEEK AND POKE INSTRUCTIONS TO ACCESS ANY PIC REGISTER FROM BASIC
- SERIAL SPEEDS TO 9600 BAUD
- IN-LINE ASSEMBLER AND CALL SUPPORT
- SUPPORTS PIC16C55X, 6X, 7X, 8X, 92X AND PIC14000 MICROCONTROLLERS
- PRODUCES DIRECT HEX CODE.



Write your PIC programs in BASIC! This PicBasic Compiler converts Basic programs into hex or binary files that can be programmed directly into a PIC microcontroller. The easy-to-use BASIC language makes PIC programming available to everyone with its English-like instruction set.

The PicBasic Compiler instruction set is compatible with the Parallax BASIC Stamp 1 (BS1).
BS1 programs can be compiled into PIC code ready for programming directly into a PIC, eliminating the need for a BASIC Stamp module. The benefits of the PicBasic Compiler include faster program execution than BASIC interpreters, longer programs possible and substantial cost savings over a BASIC Stamp.

Peek and Poke instructions let you use additional PIC features not available on the BASIC Stamp I. These include access to PORTA, B, C, D and E (if the particular PIC has them), A/D converters, hardware serial ports and other on-PIC features in BASIC, foregoing the need to use assembly language.

Assembly language instructions may be mixed with BASIC instructions through the use of the PicBasic Compiler's in-line assembler and Call instruction. Our PIC macro assembler is included and automatically invoked by the PicBasic Compiler.

The new I^2C commands let the PIC talk to external I^2C devices, such as serial EEPROMs, using only a 2-wire interface. Two PortA pins have been dedicated to the task (the particular pins assigned may be changed if so required), thus saving the need to use the special purpose PortB pins.

PicBasic Compiler enables the user to have more variables. The BS1 only provides variables from B0-B13 and W0-W6. Previous versions of the PicBasic Compiler allowed variables from B0-B21 and W0-W10. The latest version of the compiler allows variables from B0-B79 and W0-W39 when used with PICs having more RAM registers such as PIC16C622 and 16C74.
The PicBasic Compiler is a DOS command line application (it operates under Windows) and requires only a PC or compatible to run. It can create programs for the PIC16C55x, 6x, 62x, 64x, 66x, 7x, 71x, 8x, 92x and PIC14000 microcontrollers

ASM	Assembly language instructions follow .*	**NAP**	Power down processor for short period of time.
BRANCH	Computed GOTO (equivalent to ON..GOTO).	**OUTPUT**	Make pin output.
BUTTON	Debounce and auto repeat on specified pin	**PAUSE**	Delay (1mSec resolution).
CALL	Call assembly language subroutine at specified label.*	**PEEK**	Read byte from PIC register.*
DEBUG	Debugging statement (NOTE: Ignored by PBC).	**POKE**	Write byte to PIC register.*
EEPROM	Define initial contents of EEPROM.	**POT**	Read potentiometer on specified pin.
END	Stop execution and enter low power mode.	**PULSIN**	Measure pulse width (10uSec resolution).
ENDASM	Exit assembly language instruction mode.*	**PULSOUT**	Generate pulse (10uSec resolution).
FOR..NEXT	Repeatedly execute statement(s).	**PWM**	Output PWM pulse train to pin.
GOSUB	Call subroutine at specified label.	**RANDOM**	Generate pseudo random number
GOTO	Resume execution at specified label.	**READ**	Read byte from EEPROM.
HIGH	Make pin output high.	**RETURN**	Resume execution at statement following GOSUB.
I2CIN	Receive bytes from I2C device.*	**REVERSE**	Make output pin an input or an input pin an output.
I2COUT	Send bytes to I2C device.*	**SERIN**	Asynchronous serial input (N81).
IF..THEN	GOTO if specified condition is true.	**SEROUT**	Asynchronous serial output (N81).
INPUT	Make pin input.	**SLEEP**	Power down processor (1 Sec resolution).
[LET]	Perform math and assign result to variable.	**SOUND**	Generate tone or white noise on specified pin
LOOKDOWN	Search table for value.	**TOGGLE**	Make pin output and toggle state.
LOOKUP	Fetch value from table.	**WRITE**	Write byte to EEPROM.
LOW	Make pin output low.		

*PicBasic language extension not found in Parallax PBASIC 1.

ORDERING, PAYMENT, AND AVAILABILITY

Description	Part No
Basic Compiler Single User License	MEL-PBC

Payment : By Credit Card, COD, TT or Cheque.
Ordering : By phone, Email, or fax.
Availability : Orders are generally shipped the same day that they are received.

RF Solutions Ltd,
Unit 12,
Cliffe Industrial Estate,
Lewes,
E Sussex. BN8 6JL
Tel : +44 (0)1273 898 000
Fax : +44 (0)1273 480 661
E-mail sales@rfsolutions.co.uk

Web Page : http://www.rfsolutions.co.uk

Q. Does the PicBasic Compiler work with BASIC Stamp II programs?

A. The PicBasic Compiler is compatible with the BASIC Stamp I (PBASIC 1) instruction set. However, most BASIC Stamp users are using only a few of the BS2 specific instructions. BASIC Stamp II features such as more I/O pins can be accessed using the PicBasic Compiler's PEEK and POKE instructions or its in-line assembler function and assembler CALLs. These instructions allow any of the larger PICs expanded features to be accessed.

Q. Does the PicBasic Compiler work with the PIC16C5x (52, 54, 55, 56, 57, 58) parts?

A. The PicBasic Compiler does not support the older 5x or the 12C508/9 series PICs. Their 12-bit core only has a 2-level stack and the compiler requires the 8-level stack of the mid-range 14-bit core PICs. There are suitable pin-compatible replacements for the 54, 56 and 58. The PIC16C554, 556, 558, 620, 621 and 622 are an updated version of those 5x parts and are available in about the same price range. They have expanded features and are well suited to the PicBasic Compiler.

Q. Which PICs does the PicBasic Compiler support?

A. The PicBasic Compiler works with all of the PIC16C55x, 6x, 7x, 8x, 92x, PIC12C67x and PIC14000 microcontrollers. These currently include the PIC16C554, 556, 558, 61, 62(A), 620, 621, 622, 63, 64(A), 641, 642, 65(A), 66, 661, 662, 67, 71, 710, 711, 715, 72, 73(A), 74(A), 76, 77, 84, 923, 924, PIC16F83, 84, PIC12C671, 672 and PIC14000. It is particularly suited to the EEPROM PIC16C84 and Flash PIC16F84 as they are a good match to the BASIC Stamp I feature set and may be erased and reprogrammed almost instantaneously.

Q. I want to run a serial port at 9600 baud. Can the PicBasic Compiler do this?

A. Yes. The latest versions of the PicBasic Compiler supports serial baud rates up to 9600. The supported baud rates are 300, 1200, 2400 and 9600.

Q. What about using a PIC with a hardware serial port?

A. The hardware USART on the PIC16C63, 65(A), 73(A) or 74(A) may easily be used by simply PEEKing and POKEing it, thereby avoiding the SERIN and SEROUT instructions. See the USART sample program for an example.

Q. How do I use the A/D converter on the PIC16C71?

A. The A/D converter on the PIC16C7x series parts may be accessed using either the PicBasic PEEK and POKE instructions or by using the in-line assembler feature. See the ADC71 sample program for an example.

Q. Can I use PortA (C, D, E) with the PicBasic Compiler?

A. As with the A/D converter above, PortA (and other ports on PICs with more pins) may be accessed using either the PicBasic PEEK and POKE instructions or by using the in-line assembler feature. See the PORTA sample program for an example.

Q. How do I use an LCD with PicBasic?

A. Of course, an LCD serial backpack from Scott Edwards Electronics may be used in conjunction with the SEROUT instruction, just as with a BASIC Stamp. Or an LCD module may be connected directly to 6 or 7 PIC I/O pins. See the LCD sample program for an example.

Q. But I really need the BS2 SHIFTIN and SHIFTOUT instructions.

A. These instructions, like many BS2 instructions, may simply be created as a PicBasic subroutine. See the SHIFT sample program for an example.

Example Programs

```
' Blinks LED using HIGH and LOW commands to control specified pin.

Symbol LED = 2                          ' LED Pin

Loop:   high LED                        ' LED On
        pause 1000                      ' Delay 1 second

        low LED                         ' LED Off
        pause 1000                      ' Delay 1 second

        goto Loop                       ' Forever
```

```
' Read and write PORTA using PEEK and POKE instructions.

Symbol  PORTA = 5                   ' PORTA address
Symbol  TRISA = $85                 ' PORTA data direction register
Symbol  SO = 0                      ' Serial output

        poke TRISA, 0               ' Set PORTA to all output
        poke PORTA, 31              ' Send a value to PORTA

        poke TRISA, 255             ' Set PORTA to all input
        peek PORTA, B0              ' Get PORTA inputs to variable B0

        serout SO, N2400, (#B0, 10)     ' Send variable to serial out

        end
```

```
' Access PIC16C71 A/D using PEEK and POKE instructions.

Symbol  ADCON0 = 8                  ' A/D Configuration Register 0
Symbol  ADRES = 9                   ' A/D Result
Symbol  ADCON1 = $88                ' A/D Configuration Register 1
Symbol  SO = 0                      ' Serial output

        poke ADCON1, 0              ' Set PORTA 0-3 to analog inputs
        poke ADCON0, $41            ' Set A/D to Fosc/8, Channel 0, On

Loop:   poke ADCON0, $45            ' Start conversion
        pause 1                     ' Wait 1ms for conversion
        peek ADRES, B0              ' Get result to variable B0

        serout SO, N2400, (#B0, 10)     ' Send variable to serial out

        goto Loop
```