The eLabtronics USB PORT Module: 2 in 1 PIC Programmer Controller
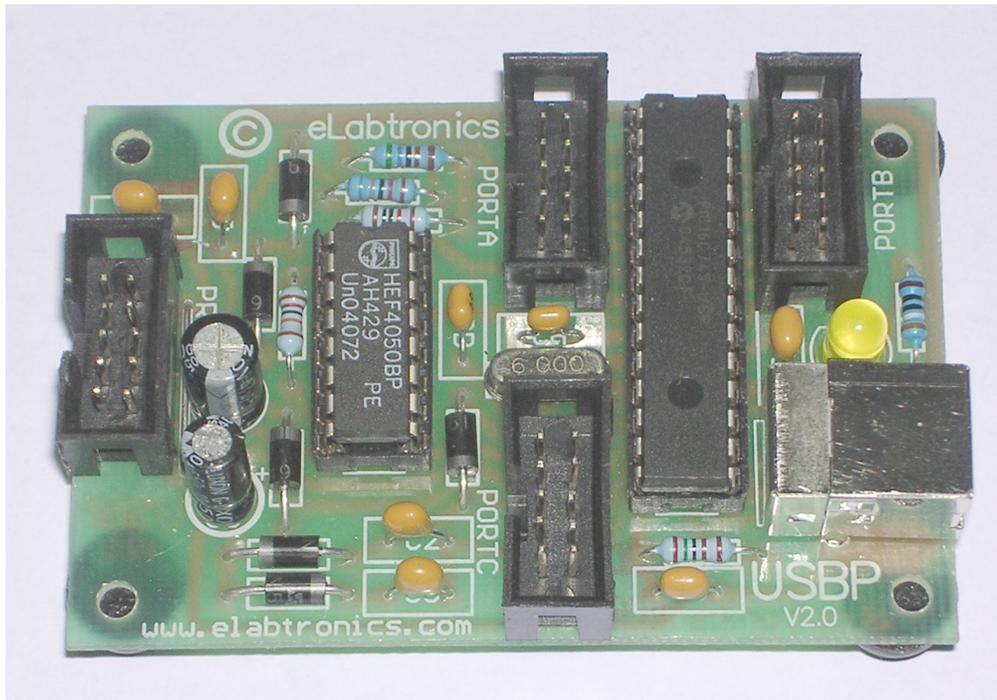
The compact USB PORT Module from eLabtronics can be used as a PIC programmer and a controller. It programs user PIC projects using the eLabtronics industry strength CoreChart graphical assembler or PIC machine code (hex) files. You can find a full version 30day FREE trial copy of CoreChart at www.elabtronics.com

Nowadays a great deal of electronic products have USB connections. Some new laptops have only USB connections. The eLabtronics USB PORT Module Programmer Controller enables you to build your own USB port into electronic projects.

There are various USB projects that power torches, fans etc. These projects make use of the 5V power on the USB but not the data transfer to the PC. The USB PORT Module allows you to add USB port to your project and also can program the PIC chips on your new microcontroller invention!

The eLabtronics USB PORT Module includes:

1. A USB PIC Programmer so that you can program PIC chips on your own PIC project PCB without removing them from the circuit (in circuit programming). The programmed PIC chip on your PCB can then communicate with the PC through the same connection.
2. Three re-programmable connectors (19 pins are usable). Most of the 19 pins can be used for Digital Input / Output signals. Some can be used as analogue (voltage level) inputs.
3. Powerful firmware for totally accessible read and write to chip registers on the USB PORT Module.

The USB PORT Module is available from eLabtronics in 3 forms:

    a) Pre-programmed USB Chip only (PCB layout provided).
    b) A kit including PCB.
    c) Fully built and tested.

Construction instructions are included in the "PCB and Kit Construction" folder of the USB PORT Module CD.

In Circuit PIC Programming

Most PIC microcontrollers can be programmed while in the application circuit. This is done with two lines for clock and data. Another three lines are for power, ground and the programming voltage. The USB PORT Module Programmer supports ICSP (In Circuit Serial Programming) via a 10 pin connector marked "PROG". Please refer to the individual PIC datasheet for the ICSP pins of the chip you wish to program.

Pinout for PROG	
1	DATA I/O
2	CLK
3	-
4	-
5	-
6	-
7	-
8	MCLR Vpp
9	VDD +5V
10	VSS 0V

10 pin Header
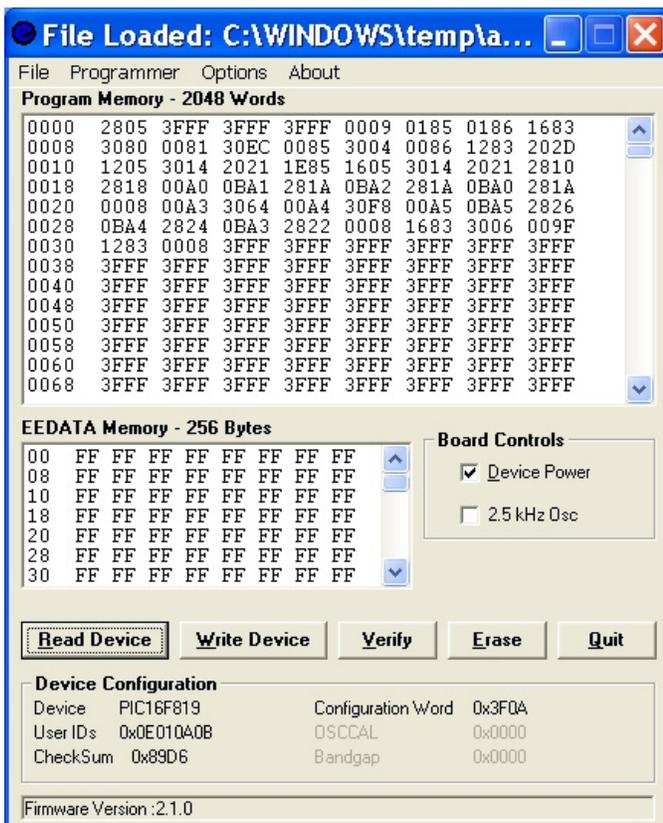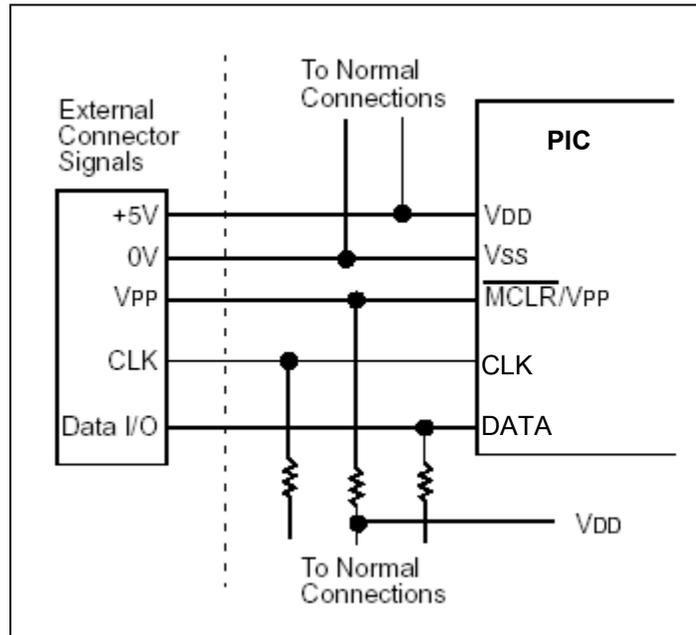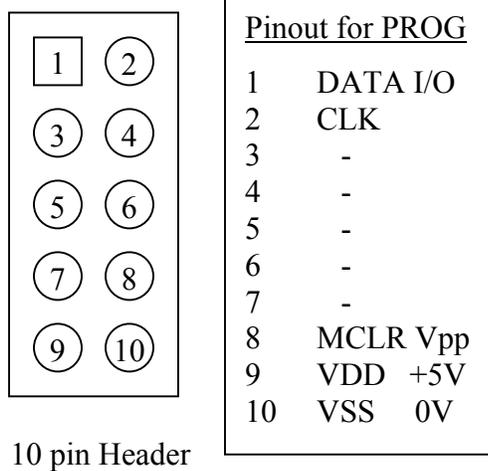
Ensure that your project PIC ICSP pins are correctly connected to the "PROG" Connector on eLabtronics USB PORT Module. Then run the eLabtronics USB Programmer software as shown on the left. This software will attempt to identify the your project PIC microcontroller and display the device model.

If the Device Configuration shows "Not Present" it means:

1. The PIC chip you are programming is not able to identify itself. Some PIC chips do not have this Device ID facility even though the PIC is connected.
2. The PIC chip is not supported by the eLabtronics USB Programmer software.
3. Your project PIC ICSP pins are wrongly connected.

2

CoreChart Software

A full version 30day trial copy of eLabtronics industry strength CoreChart graphical assembler software is included in the CD. CoreChart has a user-friendly graphical interface. This enables shorter product development cycles. It makes microcontroller programming much more accessible.

Since CoreChart contains the eLabtronics USB Programmer software, you can program your project PIC chip using the USB PORT Programmer in one step.

To set up CoreChart in order to use the USB PORT Module programmer, go to the menu bar at the top and select "Options / Hardware Settings" and set the "Programmer" selection to "USB Programmer". Make sure that no tick boxes are selected on the tick options. Before you send your program to the PIC chip, first make sure that the USB PORT Module programmer is connected to the PC and your project PIC chip is connected to the "PROG" Connector correctly. Then select 'Tools / Send to Chip'. The eLabtronics USB Programmer software will appear on the screen with the program loaded and ready to transfer. Click on the "Write Device" button to start programming your project PIC chip.
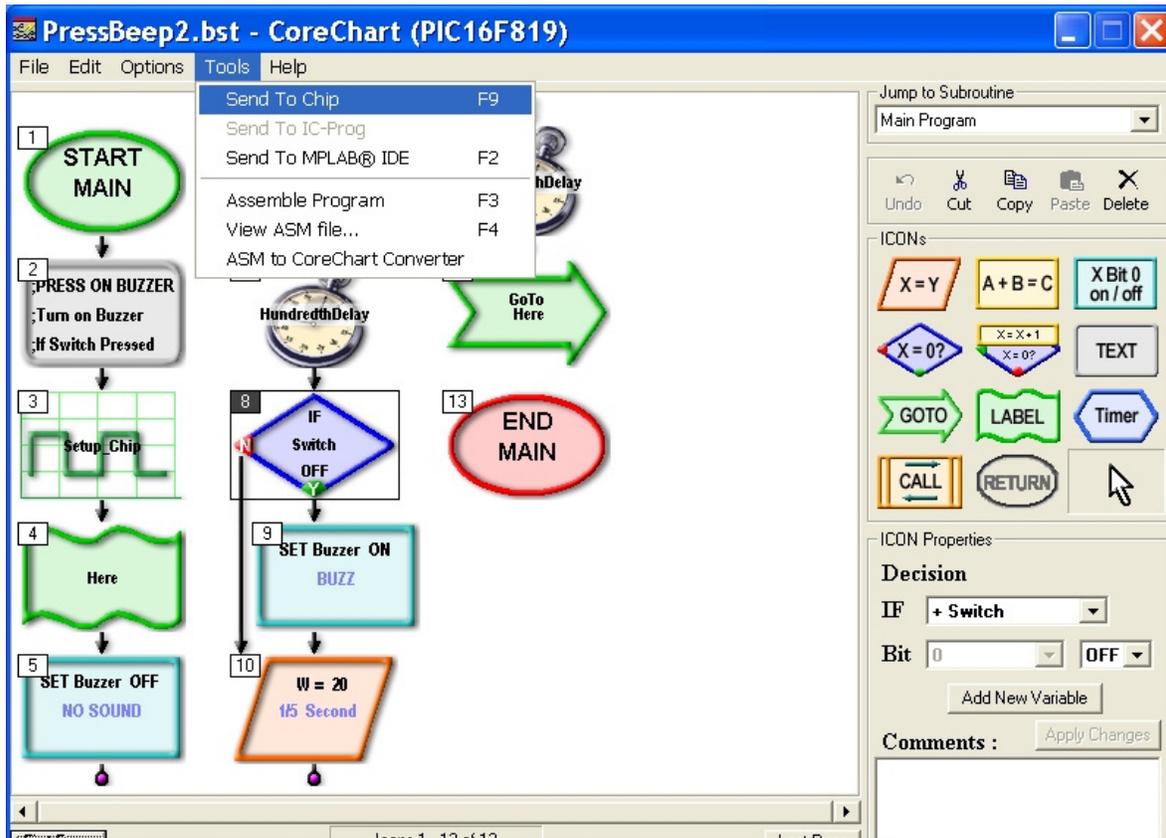
Connecting the PORTS for PC Control

The USB PORT Module can be used as a PC based controller.
The three Port Connectors are labeled PORTA, PORTB and PORTC. These names are the same as the USB PORT Module PIC chip registers which control them. A 10 pin ribbon crimp header plug and a 10 way ribbon cable connects between the USB PORT Module and your project application. These could be LEDs, relays, motor drivers, switches, sensors or other PIC chips. Notice that the pin numbering on the three Port Connectors are all different – take care and double check your connections.

Pin for PORTA		Pin for PORTB		Pin for PORTC	
1	VSS	1	VSS	1	VSS
2	VDD	2	B7	2	VDD
3	A0	3	B6	3	C7
4	A0	4	B5	4	C0
5	A2	5	B4	5	C6
6	A1	6	B3	6	C1
7	A4	7	B2	7	C2
8	A3	8	B1	8	-
9	VDD	9	B0	9	-
10	A5	10	VDD	10	-

Pin Header
(Top Down)

Controlling the USB PORT Module from the PC

The sample programs provided show how the USB PORT Module can be controlled from the PC. They serve as examples for users to develop their own applications. Compiled examples are provided which are ready to run. "Raw" Visual Basic (V6.0) source codes are available for users to create their own interfaces. More technical details are provided later in this article. Ensure that the USB PORT Module is connected to the PC before running any of the sample programs. The title of the program will read: "Programmer not found" if the USB PORT Module was not detected.

Analog Input Example

This example shows you how to use the USB PORT Module to obtain analog values and display them on the screen. The PIC16C745 MCU on the USB PORT Module has an 8 bit Analog to Digital Converter that can be utilized to obtain readings of voltages between 0 to 5V. Note that the value obtained will be a discrete value ranging from 0 to 255. To convert this value back into a voltage, divide the value by 255 and multiply by 5. In this example, the A/D converter is triggered every 100ms and the results are displayed on the form, both in "raw" data and voltage readings.

> **Analog Input Example - Programmer V2.1.0**
>
> ATD Value: 69
>
> Approx Voltage: 1.35V
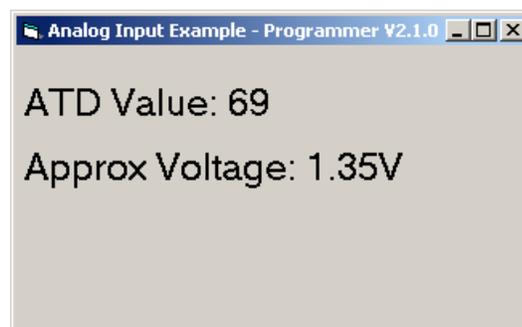
Digital Input / Output Example

This example shows you how to read and write digital data on the USB PORT Module. The program window contains two tabs, one for Input and the other for Output. This example uses PORTB, which has built in pull-up resistors so that a pin will be 5V (hence showing at '1') when it is not connected to 0V. In the output mode, click on the digits to toggle the value between '0' (0V) and '1' (5V). The decimal and hexadecimal values are displayed.
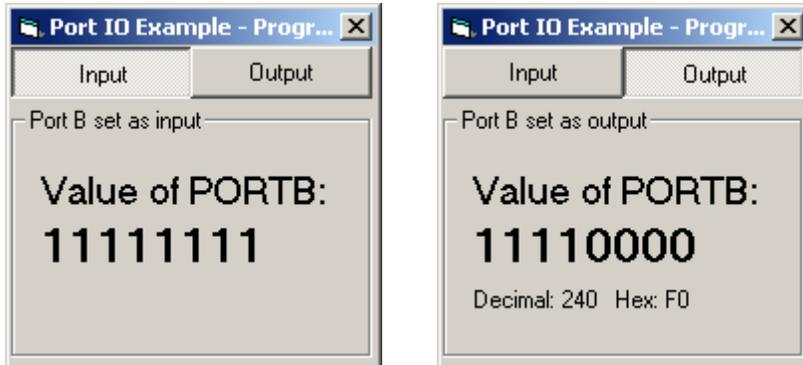
Combined Example

This example is a combination of the above examples and utilizes all the ports available to the USB PORT Module. The information is refreshed every 100ms.
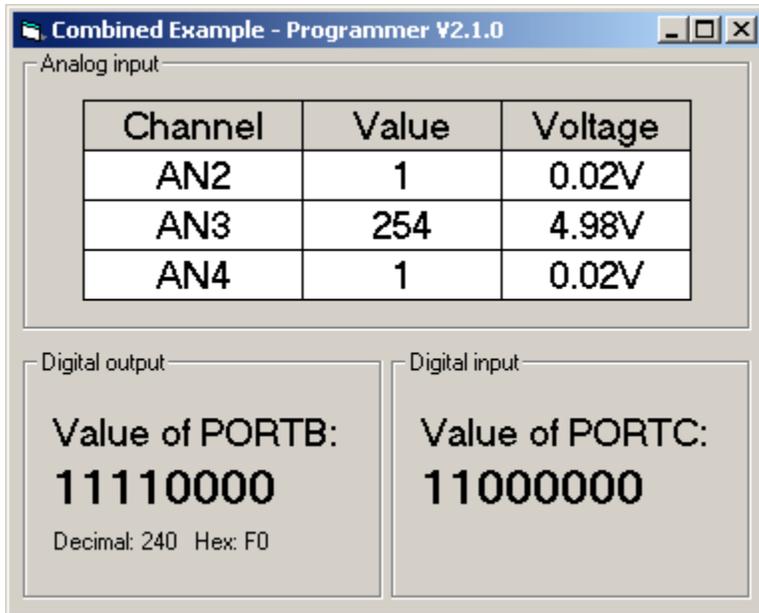
RegViewer program to view Registers on the USB PORT Module

This program provides easy access to the first 256 bytes of data memory on the PIC16C745 MCU on the USB PORT Module. The grayed out Symbols indicate that the address is not used or unavailable for use. Clicking on the Symbol Name will highlight the register. You can then change the value of the selected register. It is recommended that the registers values are not altered unless you are clear how these changes will alter the USB PORT Module functions.

Register Viewer

Programmer Ver2.1.0

Warning - changing register values without knowledge of what it is used for may crash the firmware and cause the USB PORT programmer to stop responding. If it happens, just unplug and replug the USB PORT programmer and restart the this software.

Refresh View | Exit

File Registers (0x00 to 0xFF)

Addr	Symbol Name	Val	Addr	Symbol Name	Val	Addr	Symbol Name	Val	Addr	Symbol Name	Val
00	INDF	--	40	CheckSum	EE	80	INDF	--	C0		F4
01	TMR0	35	41		14	81	OPTION_REG	07	C1		98
02	PCL	71	42		6F	82	PCL	71	C2		96
03	STATUS	19	43	TempWord	A1	83	STATUS	19	C3		02
04	FSR	04	44		6C	84	FSR	84	C4		7F
05	PORTA	00	45	ProgramCntr	B8	85	TRISA	02	C5		40
06	PORTB	7F	46		FB	86	TRISB	7F	C6		FE
07	PORTC	C6	47	CommandTable	00	87	TRISC	C2	C7		40
08		--	48		02	88		--	C8		E3
09		--	49		03	89		--	C9		40
0A	PCLATH	04	4A		04	8A	PCLATH	04	CA		F7
0B	INTCON	45	4B		05	8B	INTCON	45	CB		64
0C	PIR1	40	4C		06	8C	PIE1	0A	CC		B6
0D	PIR2	00	4D		08	8D	PIE2	00	CD		20
0E	TMR1	FF	4E		18	8E	PCON	00	CE		BF
0F	TMR1H	00	4F		0A	8F		--	CF		09
10	T1CON	00	50		09	90		--	D0		2B
11	TMR2	02	51		0B	91		--	D1		38
12	T2CON	7C	52		FF	92	PR2	3F	D2		66
13		--	53		FF	93		--	D3		04
14		--	54		FF	94		--	D4		EB
15	CCPR1	20	55		FF	95		--	D5		03
16	CCPR1H	20	56	Point5Sec	FF	96		--	D6		F7
17	CCP1CON	2F	57	DelayPeriod	02	97		--	D7		00
18	RCSTA	00	58	DelayCntr	9F	98	TXSTA	02	D8		FF
19	TXREG	00	59	PowerState	88	99	SPBRG	00	D9		60

Register Viewer

Register 0x01 (TMR0)
Value (Dec: 053   Bin: 00110101)
Enter new value: 35   Modify

Warning - chang
for may crash th
responding. If it
and restart the t

File Registers (0x00 to 0xFF)

Addr	Symbol Name	Val	Addr	Symbol Na
00	INDF	--	40	CheckSum
01	TMR0	35	41	
02	PCL	71	42	
03	STATUS	19	43	TempWord
04	FSR	04	44	
05	PORTA	00	45	ProgramCn

When you click on a Symbol Name it will be highlighted and you can then modify its value.

Using the Visual Basic code to access the USB PORT Module

A knowledge of Visual Basic (V6.0) is assumed in this summary.

Things to know:
- Only the first 256 bytes of data memory are accessible.
- Only 4 subroutines are used to interface with the USB PORT Module. The subroutines are packed into 2 Visual Basic software modules for ease of use. Examples on interfacing ports on the PIC16C745 chip are provided.

Visual Basic Software Modules

APIDeclarations.bas	Contains all the API function declarations and type declarations
USB_Module.bas	Contains private subroutines used to initialize communication with the USB PORT Module via the Windows HID driver and four public subroutines listed below.

Public Subroutines:

WriteToDevice	Used to send data to the USB PORT Module, 8 bytes at a time
Shutdown	Called when terminating the program, used to release USB handles
ReadTheVersion	Called at startup to determine if the programmer is connected
ReadReport	Call this to obtain data from the programmer. Data is placed in the ReadBuffer array

Start by examining the example Visual Basic programs. When you decide to write your own Visual Basic program to interface with the USB PORT Module import the two software modules into your project and use the Public Subroutines provided.

Syntax:

Reading from MCU data memory: WriteToDevice READREG, (Address)

This is followed by using **ReadReport** to read 8 bytes from the USB buffer on the PC. This subroutine will fill **ReadBuffer( )** with 8 bytes of data from the USB buffer on the PC, Index 1 to 8.

Note: Since the USB protocol only transmits when 8 bytes are filled, you need to ensure that 8 bytes are available for reading when **ReadReport** is called. If not this may cause **ReadReport** to take a long time to execute as it waits for timeout. This will generally cause the program to hang. Hence it is recommended that you save your work regularly.

Writing to MCU data memory: WriteToDevice WRITEREG, (Address), (Data)

Constants that you need to be aware of:

Constant	Value (decimal)
READREG	117
WRITEREG	85

Limitations

Communications bandwidth between the USB PORT Module and the PC is limited to 800 bps, which is 8 bytes every 10ms. This is an inherent limitation of the Windows HID device driver that is used by the USB PORT Module. WriteToDevice can only be executed every 10ms. Hence the Visual Basic program will wait until the next 10ms cycle before the WriteToDevice is executed. This holds true for **ReadReport** as well. Calling the subroutine in succession will cause the subsequent calls to the subroutine to stall until the 10ms interval is up.

Unavailable resources:

The following resources are constantly used by the USB PORT Module firmware and are not available for general use.

CCP1 – PORTC pin 2 (possible use of PORTC is with the CCP2 only)
PWM – Not available as it is used by the firmware as a voltage pump to generate 12V for ICSP.
AN0, AN1 – Analog to Digital Converter channel 0 & 1, PORTA pin 0,1

Available resources:

The following resources are not used by the USB PORT Module firmware at anytime and are always available for use.

PORTA – A2, A3, A4, A5 (Analog channels AN2, AN3, AN4)
PORTB – B0 – B6
PORTC – C0, C1

Shared Resources:

The following resources are rarely used by the USB PORT Module and are normally available for general use. C6 & C7 are only used during ICSP programming. B7 is connected to the LED and used as a power indicator. B7 is not used by the firmware after initialization.

B7, C6, C7

For more information email: USBP@elabtronics.com

eLabtronics
51 Byron Place, Adelaide
South Australia, 5000

Ph: +61 8 8231 5966   Fax: +61 8 8231 5266
www.elabtronics.com