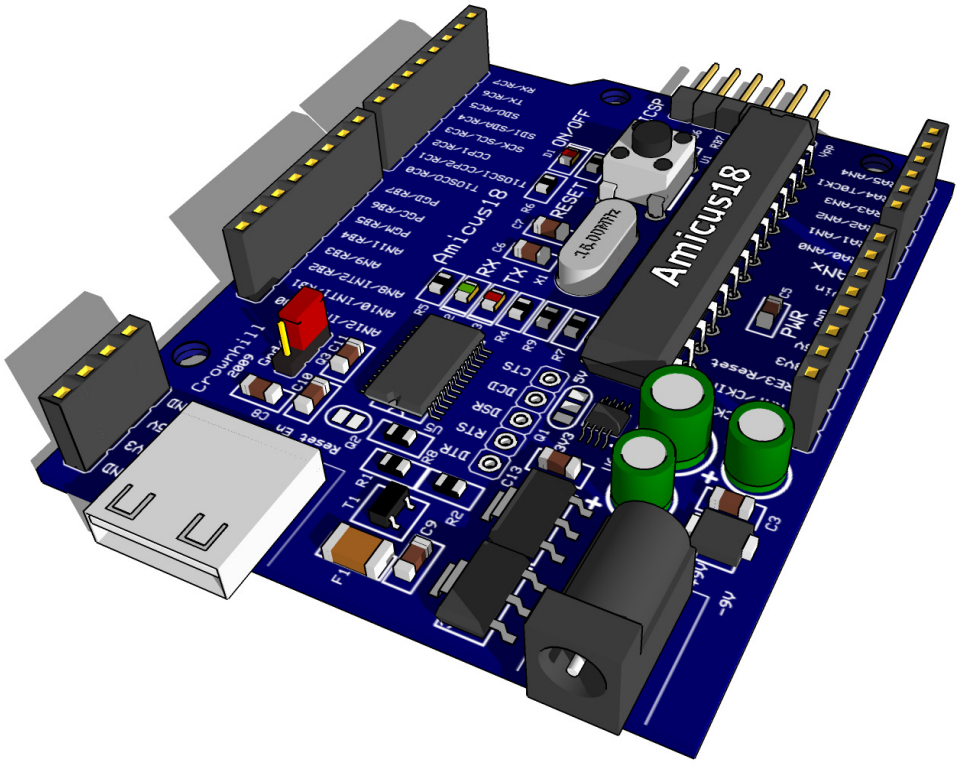
Amicus18
Hardware Overview

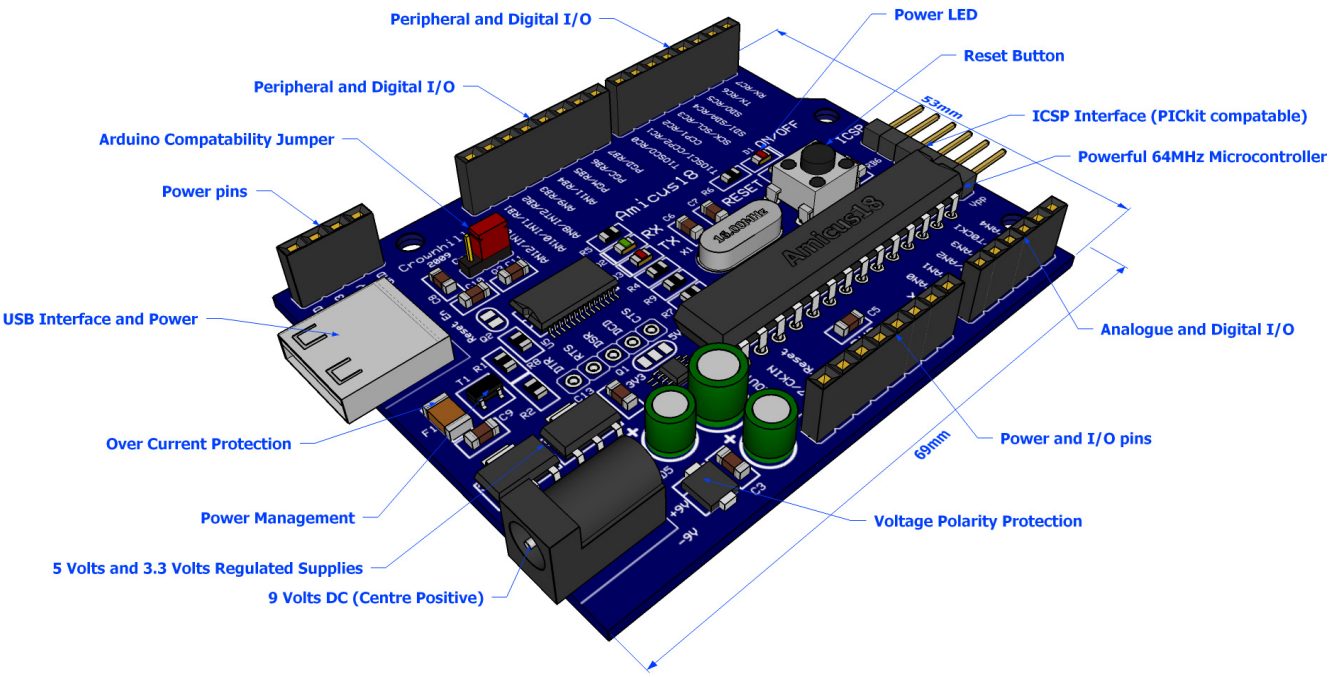


Amicus18 Hardware

Amicus18 Hardware

Amicus18 Hardware Overview

The Amicus18 hardware is based upon the world famous Arduino board, however, the Amicus18 board uses a Microchip PICmicro[tm] microcontroller instead of an Atmel AVR type.



It has exactly the same dimensions as the Arduino, and all Arduino shields will physically fit on the Amicus18 board.

The microcontroller used on the Amicus18 is the Microchip PIC18F25K20, which has 32768 bytes of flash memory, 1536 bytes of RAM, and operates at 64MHz, which equates to 16 MIPS (Million Instructions per Second).

There are nine 10-bit ADC (Analogue to Digital Converter) inputs, and two 10-bit PWM (Pulse Width Modulation) outputs, as well as two comparators, a USART (Universal Synchronous Asynchronous Receiver Transmitter), SPI (Serial Peripheral Interface), I^2C (Inter-Integrated Circuit), and four timers, each with various internal operations attached to them.

Each of the microcontroller's I/O lines are brought out for use with external devices such as LEDs, Servos, Potentiometers, LCDs etc…

Communication with the Amicus18 board is through a USB interface, which presents itself as a standard serial port on the PC. The microcontroller can be programmed directly through this port so there is no need for a dedicated device programmer, however, if the need arises, there is an ICSP (In Circuit Serial Programming) interface suitable for all programmers, but tailored for the Microchip PICkit2[tm] programmer.

Power can be supplied to the board either via the USB port, or an external 9 Volt DC source. When powered from the USB port, a maximum of 500mA (milliAmp) may be drawn, and the USB port is protected by a resetable fuse. When powered via a 9V source, a maximum of 800mA may be drawn.

The microcontroller is a 3.3 Volts type, however, there is also a 5 Volt supply always available.

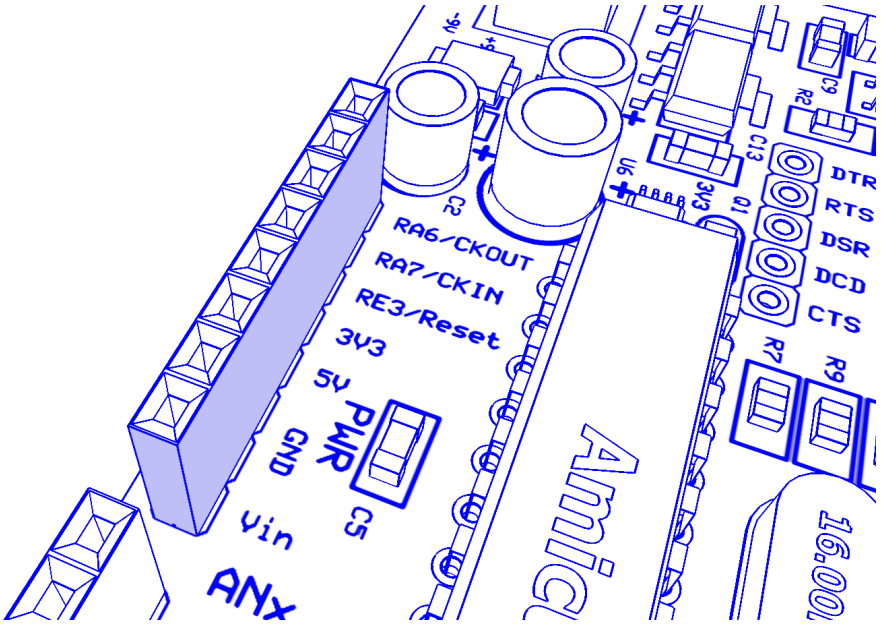
The Amicus18 board is extremely easy to use, in fact, no previous microcontroller experience is required in order to get your first project up and running, as you'll find out later.

Amicus18 Hardware

Amicus18 Sockets

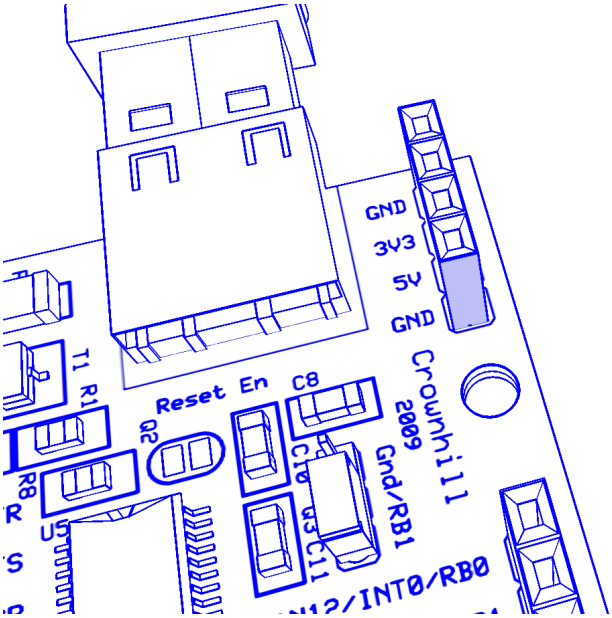
As mentioned earlier, each of the microcontroller's I/O lines is brought to the outside world via 2.54mm (0.1") SIL sockets on the Amicus18 board. The operation of each block of pins is outlined below:

The 8-pin Power header socket:



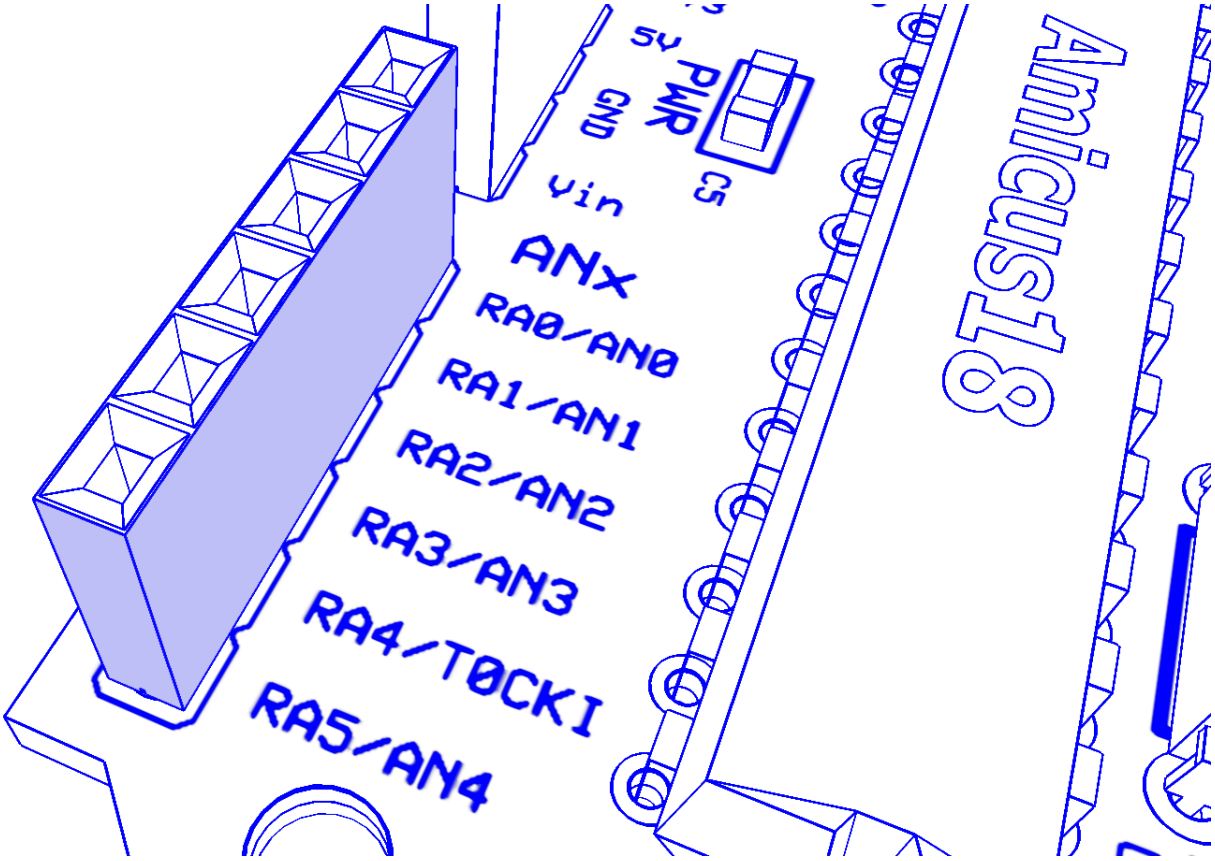
- RA6 which is bit-6 of PortA. This pin defaults to the Clock Output Pin where the crystal is connected. It may be used as an I/O pin only when an internal oscillator setting is chosen.
- RA7 which is bit-7 of PortA. This pin defaults to the Clock Input Pin where the crystal is connected. It may be used as an I/O pin only when an internal oscillator setting is chosen.
- Microcontroller's reset line, which also acts as bit-3 of PortE (RE3), and is also the voltage input for a device programmer such as the PICkit2.
- 3.3 Volts output. 500mA when powered via USB, or 800mA when powered by an external 9 Volts source.
- 5 Volts output. 500mA when powered via USB, or 800mA when powered by an external 9 Volts source.
- Ground (0 Volts).
- DC 9 Volts input. This may be used to power the board.

The 4-pin Power header socket:



- Ground (0 Volts)
- 3.3 Volts output. 500mA when powered via USB, or 800mA when powered by an external 9 Volts source.
- 5 Volts output. 500mA when powered via USB, or 800mA when powered by an external 9 Volts source.

The PortA (Anx) socket:



- RA0 which is bit-0 of digital PortA. This pin can also be configured as Input 0 (AN0) of the 10-bit ADC (*Analogue to Digital Converter*). It can also be configured as the negative (-) input pin to either Comparator 1 or 2.

- RA1 which is bit-1 of digital PortA. This pin can also be configured as Input 1 (AN1) of the 10-bit ADC (*Analogue to Digital Converter*). It can also be configured as the negative (-) input pin to either Comparator 1 or 2.

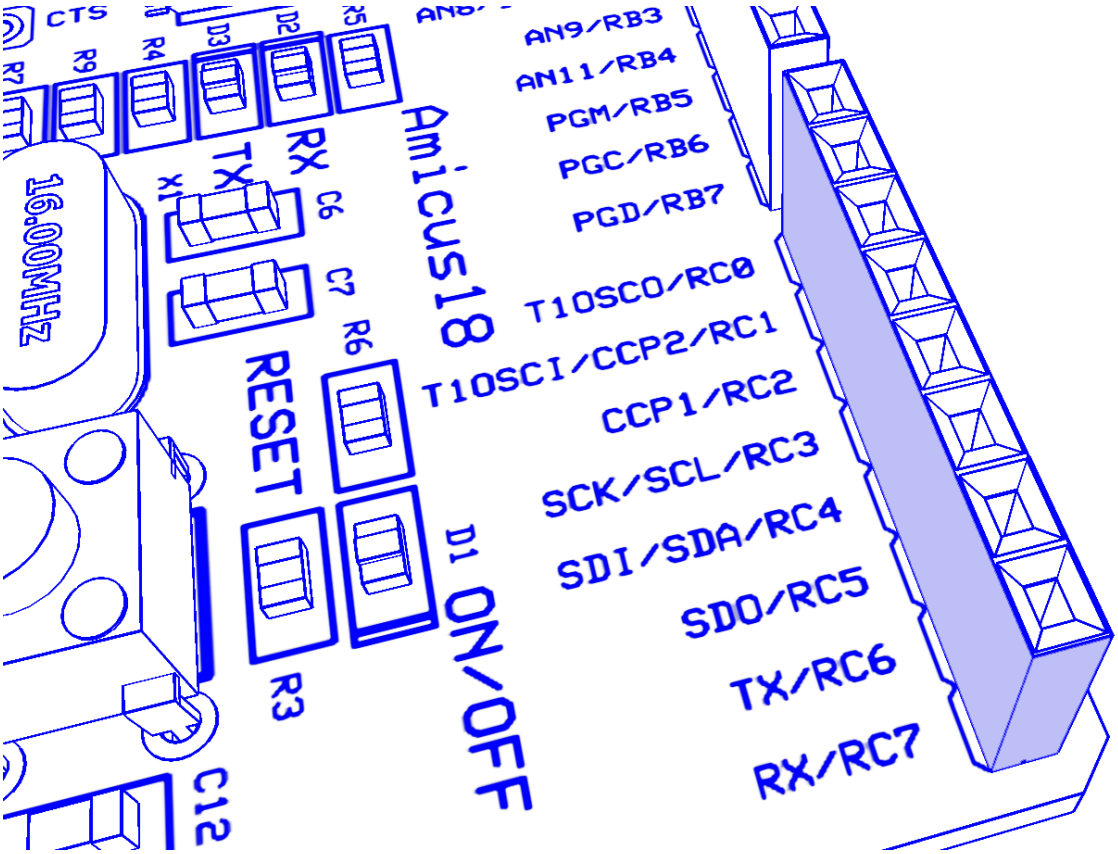
- RA2 which is bit-2 of digital PortA. This pin can also be configured as Input 2 (AN2) of the 10-bit ADC (*Analogue to Digital Converter*). It can also be configured as the positive (+) input pin to Comparator 2, or the output for the internal voltage reference.

- RA3 which is bit-3 of digital PortA. This pin can also be configured as Input 3 (AN3) of the 10-bit ADC (*Analogue to Digital Converter*). It can also be configured as the positive (+) input pin to Comparator 1.

- RA4 which is bit-4 of digital PortA. This pin can also be configured as the input trigger for Timer 0. It can also be configured as the output pin of Comparator 1.

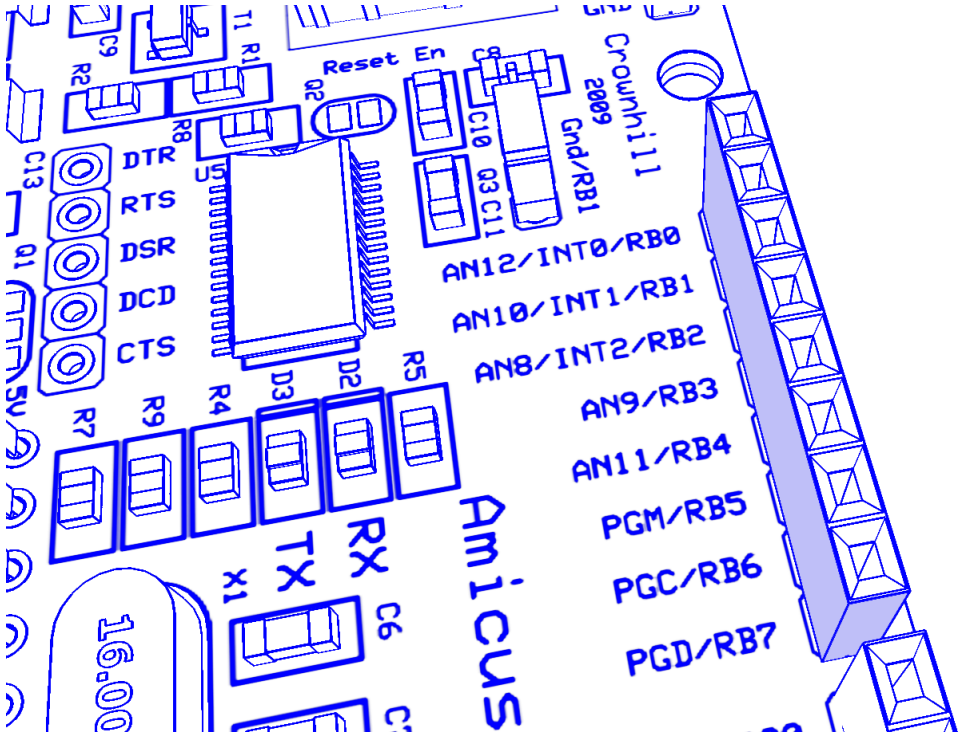
- RA5 which is bit-5 of digital PortA. This pin can also be configured as Input 4 (AN4) of the 10-bit ADC (*Analogue to Digital Converter*). It can also be configured as the output pin of Comparator 2.

The PortC socket:



- RC0 which is bit-0 of digital PortC. This pin can also be configured as the input for Timer 1.
- RC1 which is bit-1 of digital PortC. This pin can also be configured as the input for Timer 1, or a PWM (*Pulse Width Modulation*) output.
- RC2 which is bit-2 of digital PortC. This pin can also act as a PWM (*Pulse Width Modulation*) output.
- RC3 which is bit-3 of digital PortC. This pin can also be configured as the clock source for I^2C (*Inter-Integrated Circuit*) or SPI (*Serial Peripheral Interface*) communications.
- RC4 which is bit-4 of digital PortC. This pin can also be configured as the data source for I^2C (*Inter-Integrated Circuit*) or the data output for SPI (*Serial Peripheral Interface*) communications.
- RC5 which is bit-5 of digital PortC. This pin can also be configured as the data input for SPI (*Serial Peripheral Interface*) communications.
- RC6 which is bit-6 of digital PortC. This pin can also be configured as the USART (*Universal Synchronous Asynchronous Receiver Transmitter*) output for serial communications.
- RC7 which is bit-7 of digital PortC. This pin can also be configured as the USART (*Universal Synchronous Asynchronous Receiver Transmitter*) input for serial communications.

The PortB socket:



- RB0 which is bit-0 of digital PortB. This pin can also be configured as input 12 (AN12) of the 10-bit ADC, or an external interrupt trigger.
- RB1 which is bit-1 of digital PortB. This pin can also be configured as input 10 (AN10) of the 10-bit ADC, or an external interrupt trigger.
- RB2 which is bit-2 of digital PortB. This pin can also be configured as input 8 (AN8) of the 10-bit ADC, or an external interrupt trigger.
- RB3 which is bit-3 of digital PortB. This pin can also be configured as input 9 (AN9) of the10-bit ADC, or an alternative PWM (*Pulse Width Modulation*) output.
- RB4 which is bit-4 of digital PortB. This pin can also be configured as input 11 (AN11) of the 10-bit ADC, or an external interrupt trigger.
- RB5 which is bit-5 of digital PortB. This pin can also be configured as an external interrupt trigger.
- RB6 which is bit-6 of digital PortB. This pin can also be configured as an external interrupt trigger, and is also the clock line for a device programmer such as the PICkit2.
- RB7 which is bit-7 of digital PortB. This pin can also be configured as an external interrupt trigger, and is also the data line for a device programmer such as the PICkit2.

Each pin of the microcontroller is capable of sourcing or sinking 25mA, with a maximum of 100mA per port.

The microcontroller's architecture is very versatile, allowing several internal peripherals to share the same pin, thus maximising the flexibility, but keeping the size of the device small. Each internal peripheral can be enabled, disabled and configured very easily from within the free BASIC compiler environment.

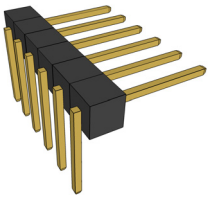
Although the microcontroller has a 3.3 Volts operating voltage, all I/O pins are 5 Volt tolerant.

Device Programming Header

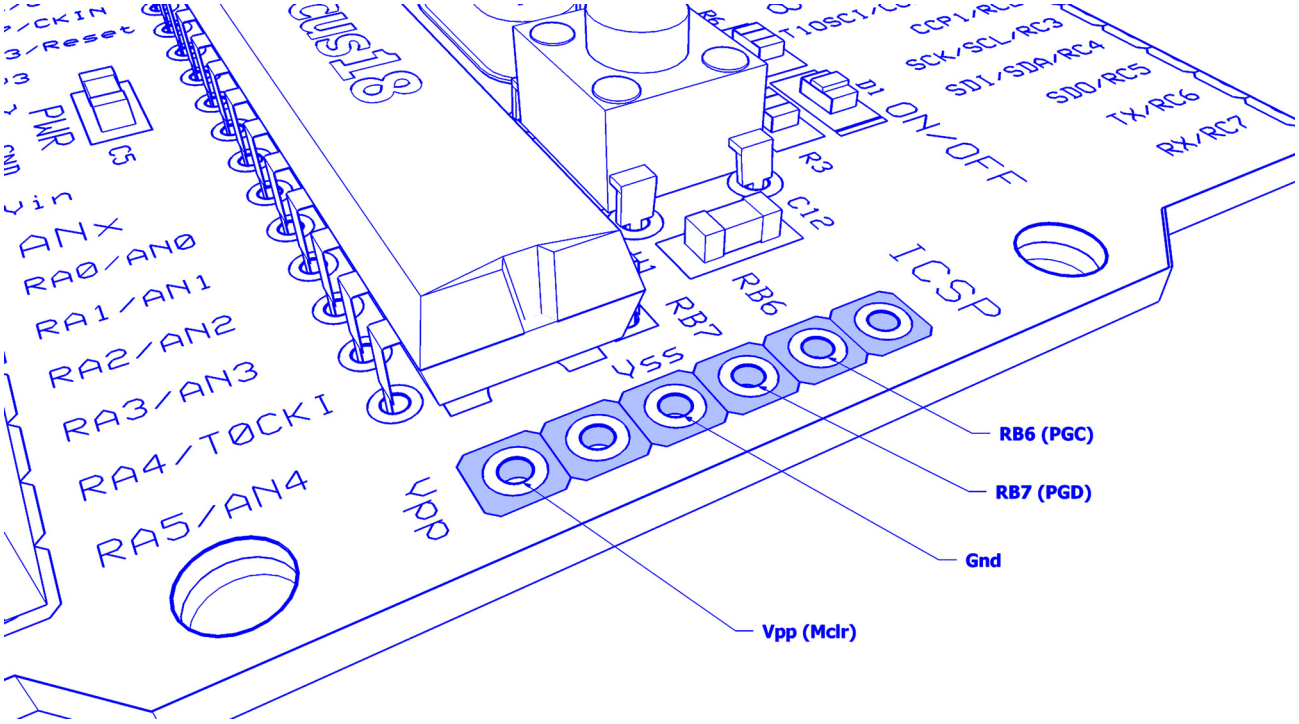
The Amicus18 board has the ability to be programmed via ICSP (In Circuit Serial Programming). This bypasses the built in bootloader, and indeed, will overwrite it.

The header has been designed for a PICkit2[tm] programmer to fit straight onto it, however, any other device programmer may be used with a suitable adapter. It must be remembered that the microcontroller is a 3.3 Volt PIC18F25K20 type, therefore if a programmer other than a PICkit2 [tm] is used, ensure that it supports this device, as a 5 Volt only programmer may damage the microcontroller.

The header requires a right angle 2.54mm (0.1") SIL:



The programming header's location is shown below:

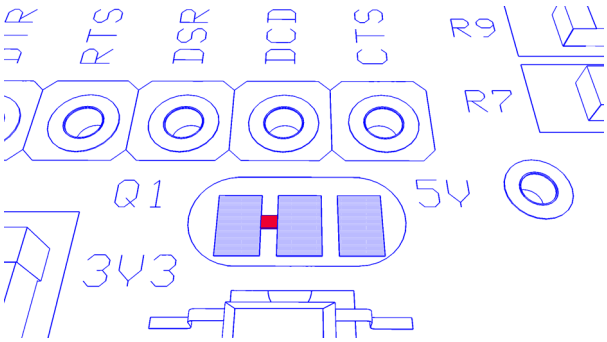


Jumper and Pad Settings

The Amicus18 board has a jumper and two pads that can alter it's characteristics.

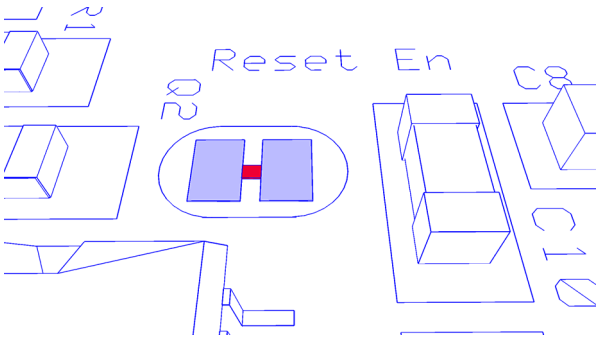
Pad Q1

This allows a 5 Volts type microcontroller to be used with the board instead of the supplied 3.3 Volt type.



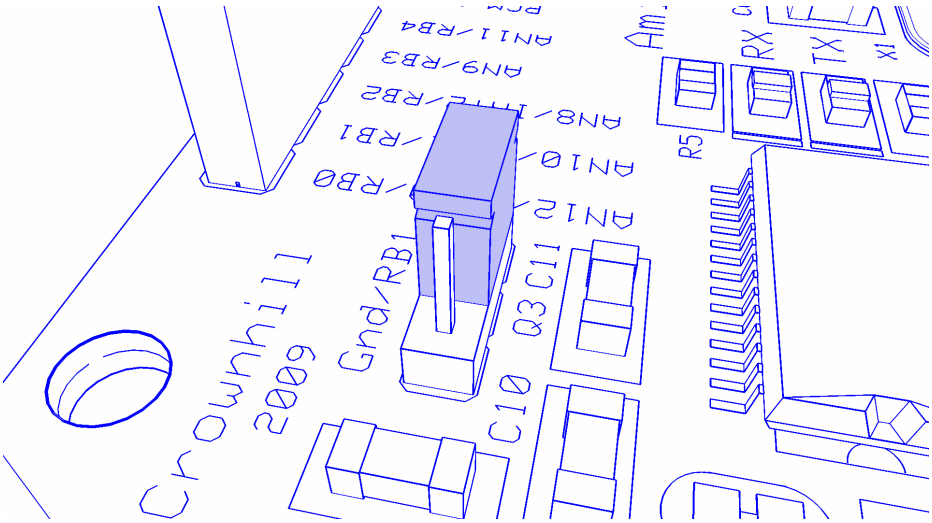
Pad Q2

This allows disconnection of the internal Reset for the microcontroller from the USB bootloader.



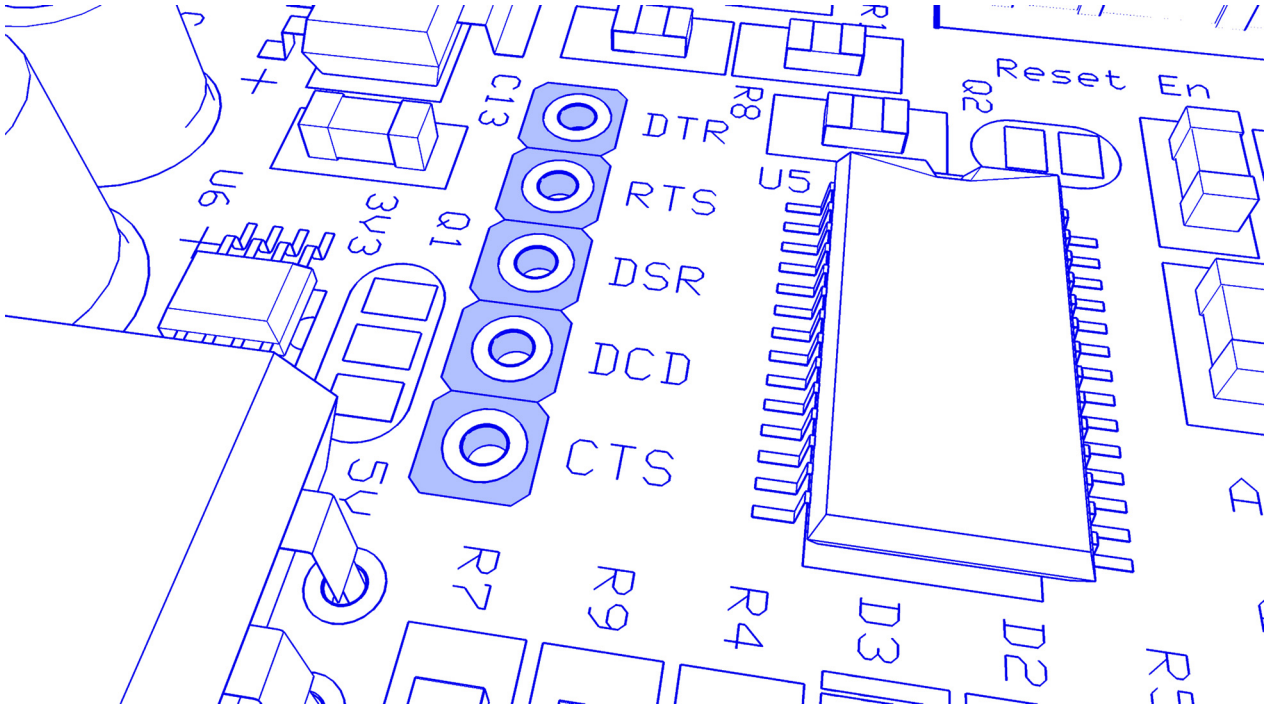
Jumper Q3

This allows maximum compatibility with existing Arduino shields. The PIC18F25K20 microcontroller has more I/O lines than that of an Atmel, therefore, two of the pins on the PortB socket operate differently on the Amicus18. RB1 is a Ground pin on the Arduino board, but this would waste a valuable I/O pin if it were simply grounded. Instead, Jumper Q3 can be configured for RB1 or Ground.



Serial Handshake Connections

The USB to serial device also emulates the handshaking lines of a conventional serial port. These are shown below:



The Amicus18 board uses the DTR line in-order to reset the microcontroller, however, the other lines are available to use. The direction of each line is shown below:

- DTR This is an output from the PC to the Amicus18 board.
- RTS This is an output from the PC to the Amicus18 board.
- DSR This is an input to the PC from the Amicus18 board.
- DCD This is an input to the PC from the Amicus18 board.
- CTS This is an input to the PC from the Amicus18 board.

Free BASIC compiler

There's a free compiler specially adapted for the Amicus18 development platform, based upon the Proton BASIC compiler.

The compiler has no restrictions or time limits and is fully functional. It can be downloaded from www.MyAmicus.co.uk.

Here's a very small sample of the BASIC language:

```
' Flash an LED connected to RB0
  While 1 = 1          ' Create an infinite loop
    High RB0           ' Bring the LED pin high (illuminate the LED)
    DelayMs 500        ' Wait 500ms (half a second)
    Low RB0            ' Pull the LED pin low (Extinguish the LED)
    DelayMs 500        ' Wait 500ms (half a second)
  Wend                 ' Close the loop
```

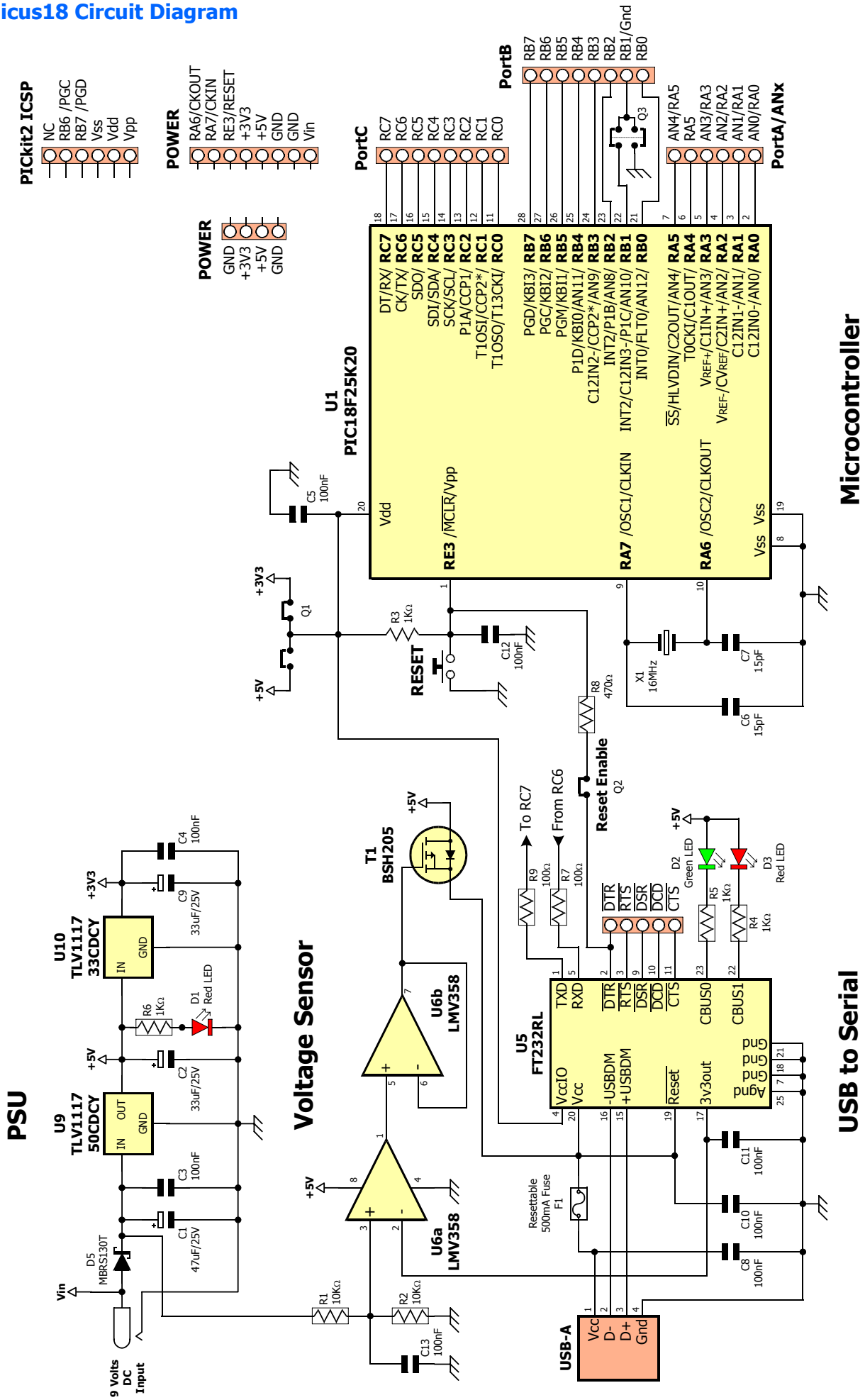
As can be seen, tha language is very simple to understand, but has a powerful command set, and produces true assembler code that talks to the microcontroller directly.

Here's a slightly more complex program:

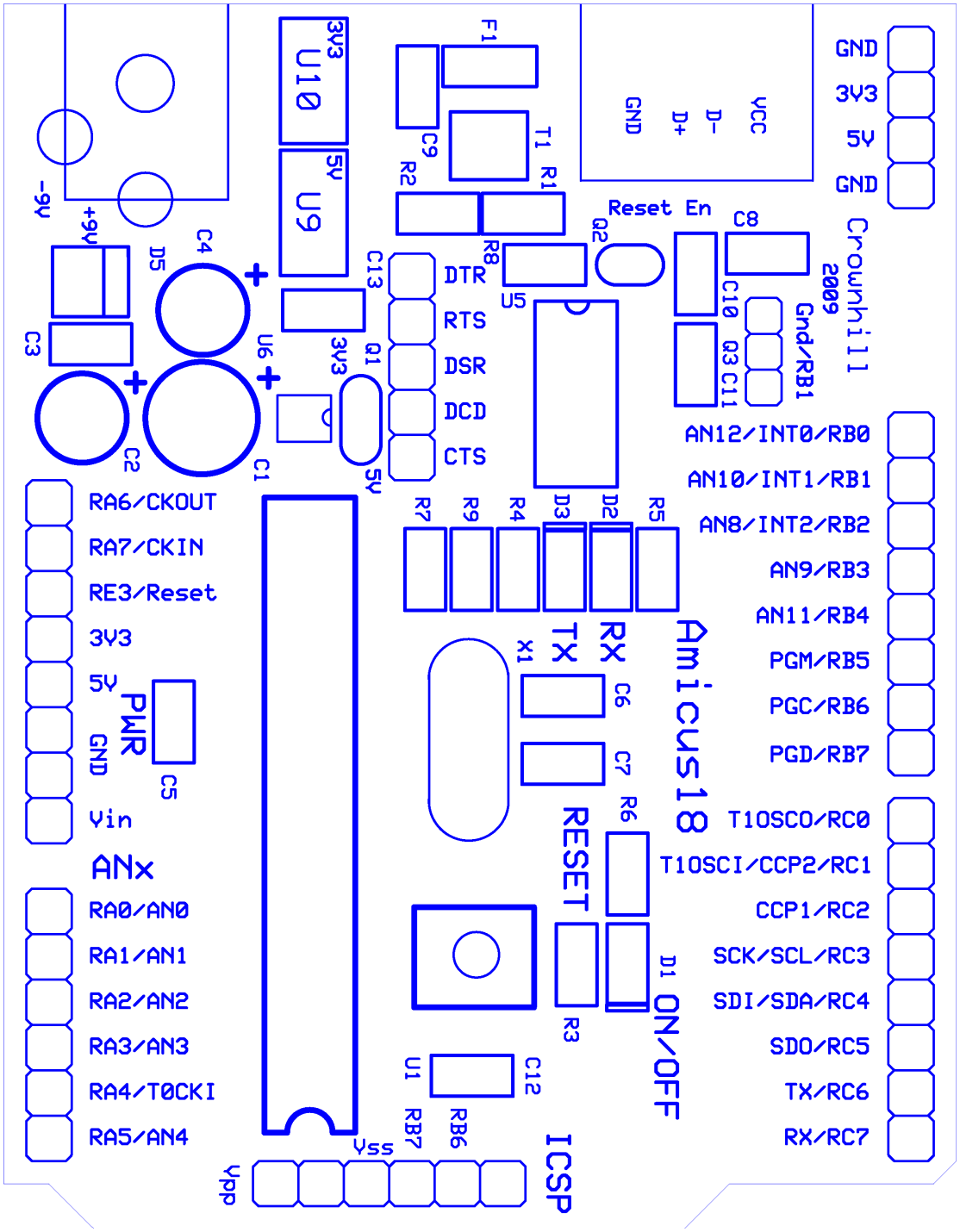
```
' Pulse both LEDs, one decreases while the other increases brightness
'
  Include "Hpwm10.inc"                ' Load the 10-bit PWM macros into the program

  Dim wDutyCycle As Word              ' Holds the duty cycle of the PWM pulses
  While 1 = 1                         ' Create an infinite loop
    '
    ' Increase LED1 illumation, while decreasing LED2 illumination
    '
    For wDutyCycle = 0 To 1023          ' Cycle the full range of 10-bits
      WriteAnalog1(wDutyCycle)          ' PWM on CCP1 (Bit-2 of PortC) (0 to 1023)
      WriteAnalog2(1023 - wDutyCycle)   ' PWM on CCP2 (Bit-1 of PortC) (1023 to 0)
      DelayMS 5                         ' A small delay between duty cycle changes
    Next                                ' Close the loop
    DelayMS 5
    '
    ' Decrease LED1 illumation, while increasing LED2 illumination
    '
    For wDutyCycle = 1023 To 0 Step -1  ' Cycle the full 10-bit range (reversed)
      WriteAnalog1(wDutyCycle)          ' PWM on CCP1 (Bit-2 of PortC) (1023 to 0)
      WriteAnalog2(1023 - wDutyCycle)   ' PWM on CCP2 (Bit-1 of PortC) (0 to 1023)
      DelayMS 5                         ' A small delay between duty cycle changes
    Next                                ' Close the loop
  Wend                                  ' Do it forever
```


Amicus18 Circuit Diagram



Amicus18 PCB Layout



Installing the Amicus18 USB Driver

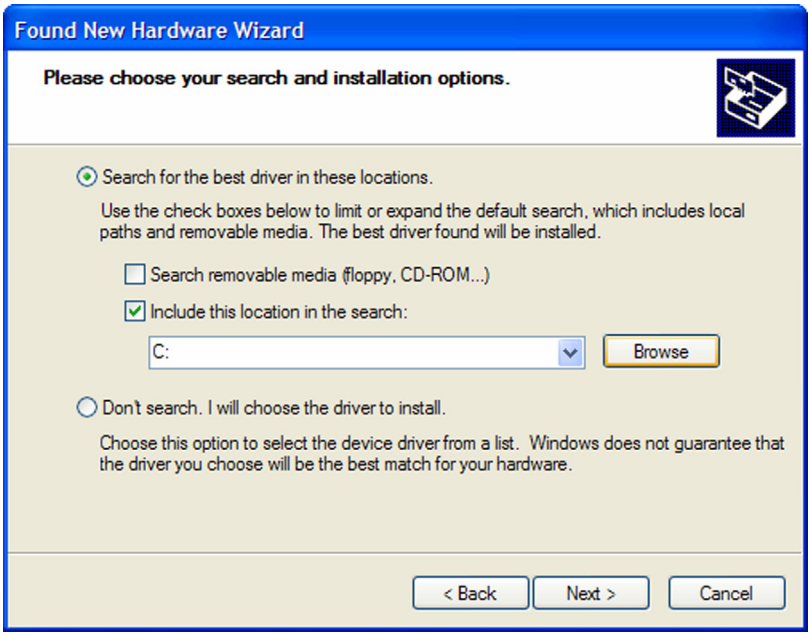
The Amicus18 board uses an FTDI serial to USB device, which presents itself as a standard com port on the PC. However, this requires USB drivers to be installed the first time the Amicus18 board is connected to your computer. This is a simple process and a step by step guide is outlined below for a Windows XP system. Note that Vista systems use the same principle, only windows and dialogues will change:

Plug the USB cable into a free USB port on the PC, and then into the Amicus18's USB port.

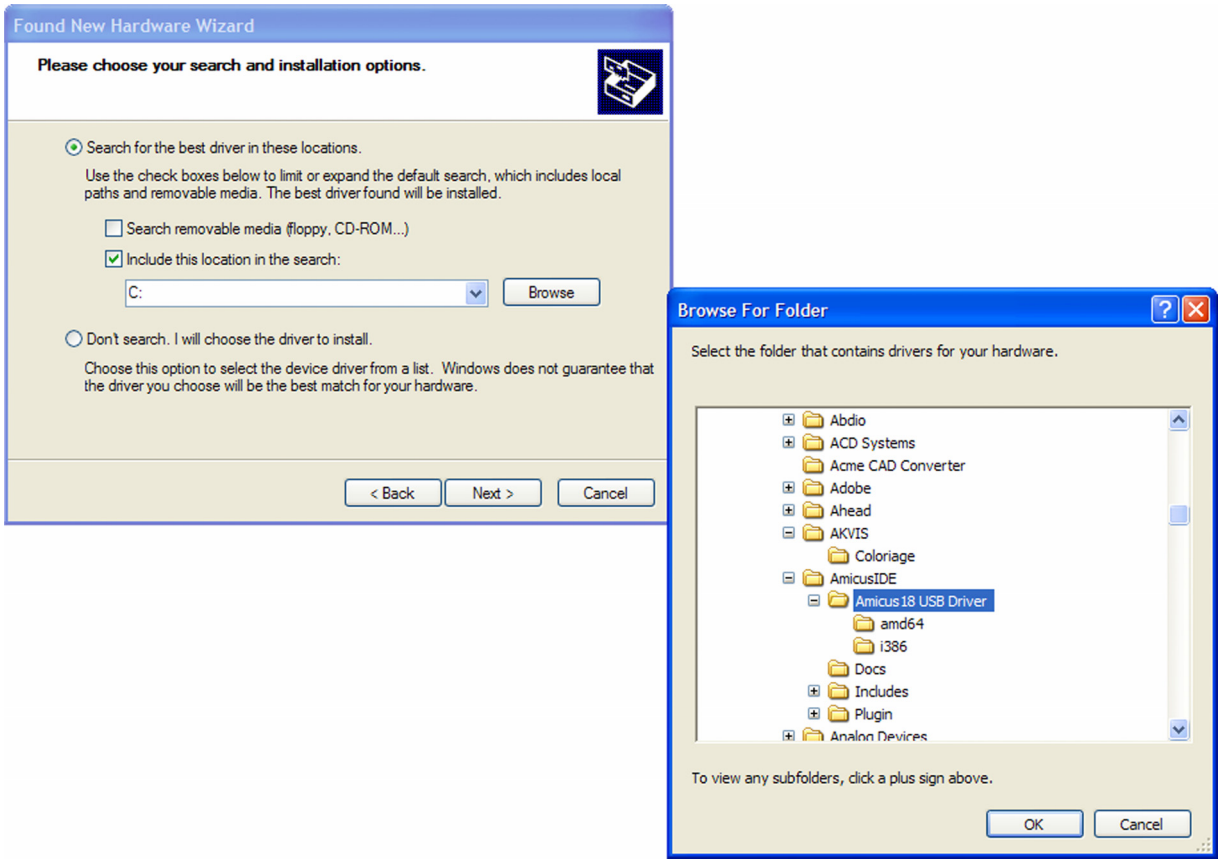
Note. Make sure you plug the Amicus18 board into a powered USB HUB or direct to the PC's USB port, as un-powered HUBs can only supply 100mA of power, instead of 500mA for powered HUBs.

The first window will inform you that a new device has been found on the USB port:

Choose the option "**Install from a list or specific location**" and click ***Next***:

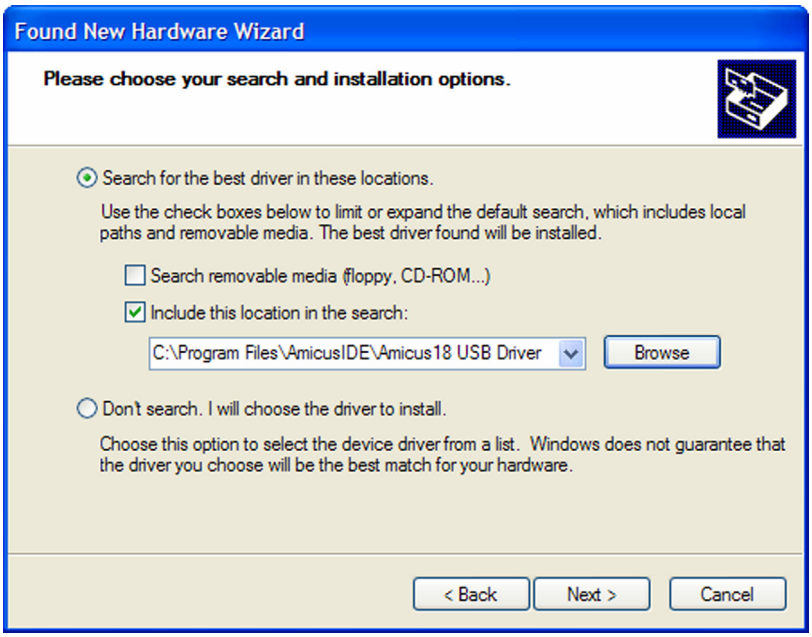


Make sure the options are ticked as in the previous window and click on the ***Browse*** button:



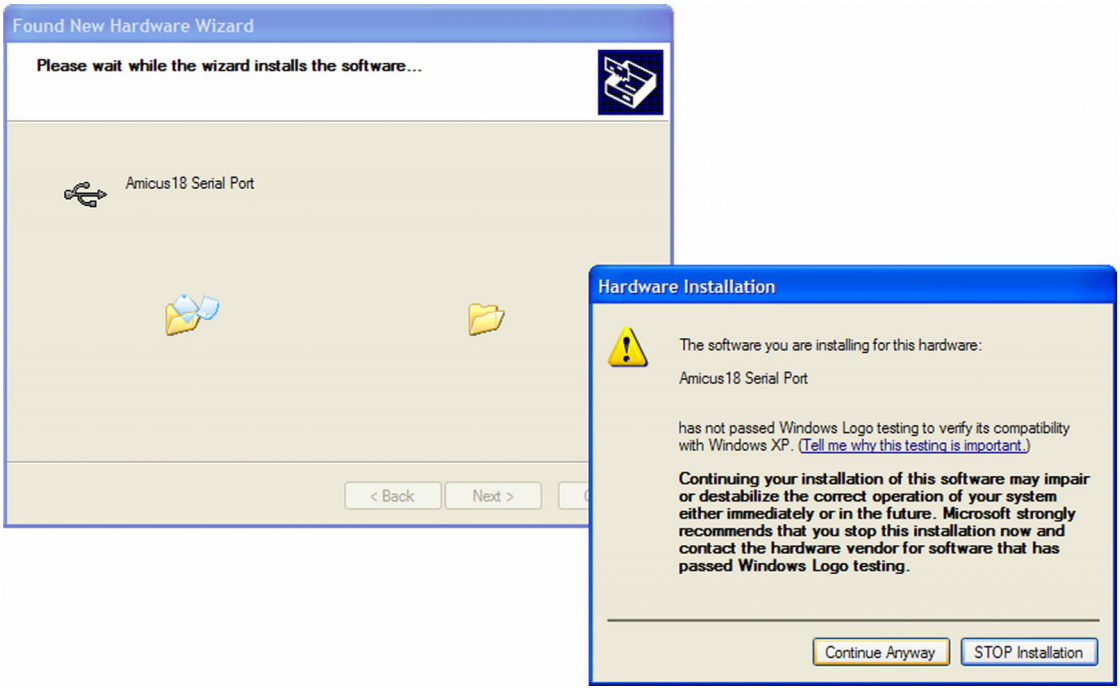
Navigate to the Amicus18's install path which it defaults to "**C:\Program Files\AmicusIDE**" and choose the "**Amicus18 USB Drive*r***" folder. Click **OK**:

The windows should look like the image below:

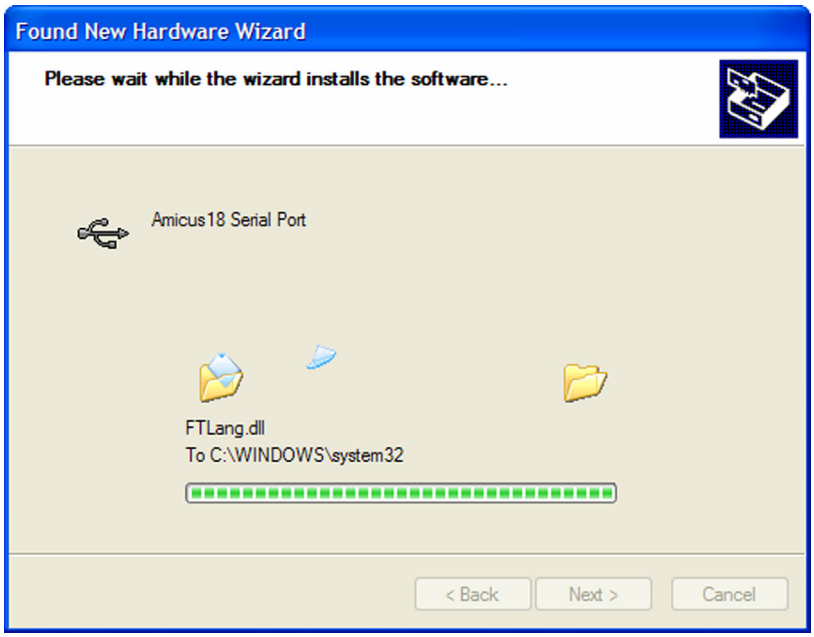


Click the ***Next*** button and the driver will begin to install.

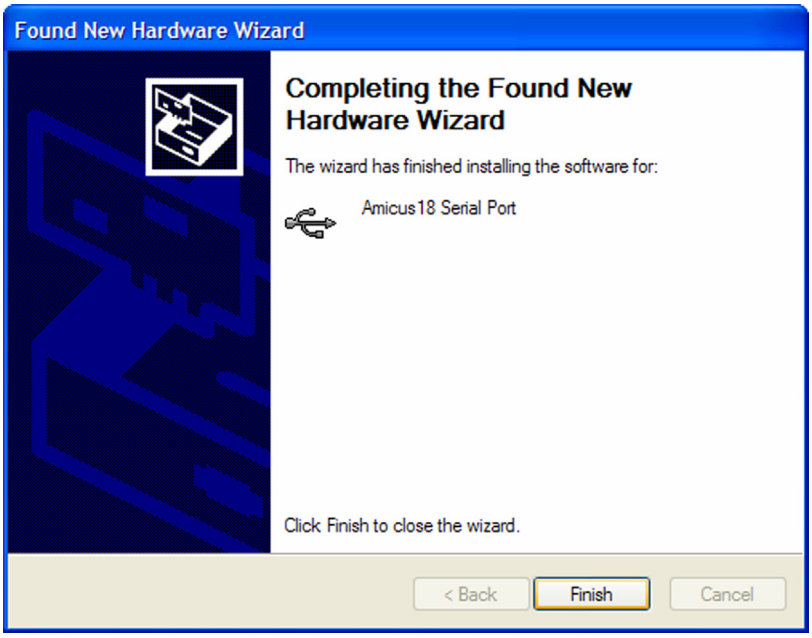
You will see a windows message stating that the drivers have not been certified by Microsoft. This is quite normal and nothing to be worried about, just click the ***Continue Anyway*** button:



The driver will continue to install:

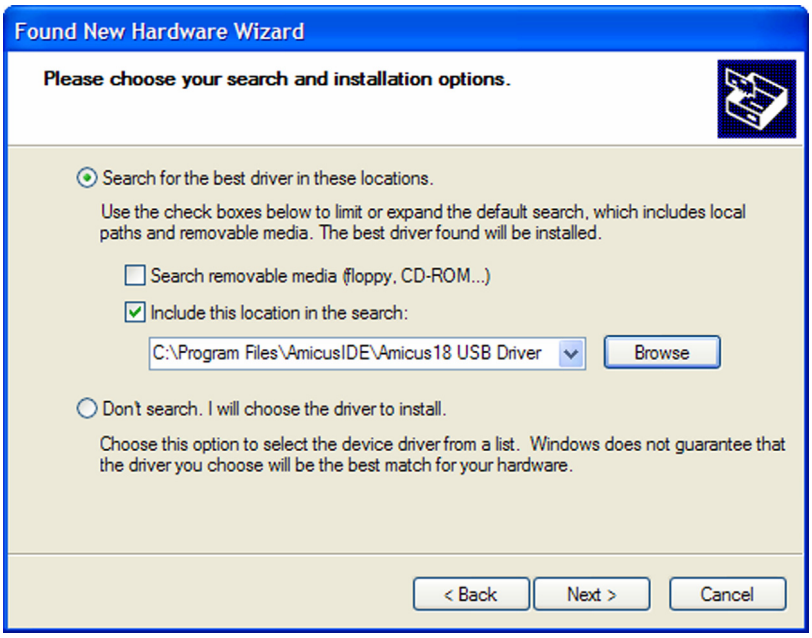


Once the driver is complete it will show the window below:



Click on the ***Finish*** button.

Note that the above procedure will need to be carried out twice for the driver to be fully installed, however, the second time, the files will have already been located on the hard drive, so it may not be necessary to navigate to the driver folder:



The USB drivers are now installed and will not require re-doing, unless the Amicus board is inserted into a different USB port on the computer, in which case, choose the "**Install the software automatically**" option on the initial driver install window.

