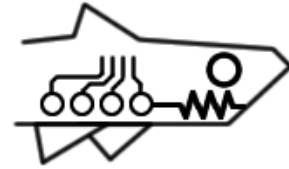


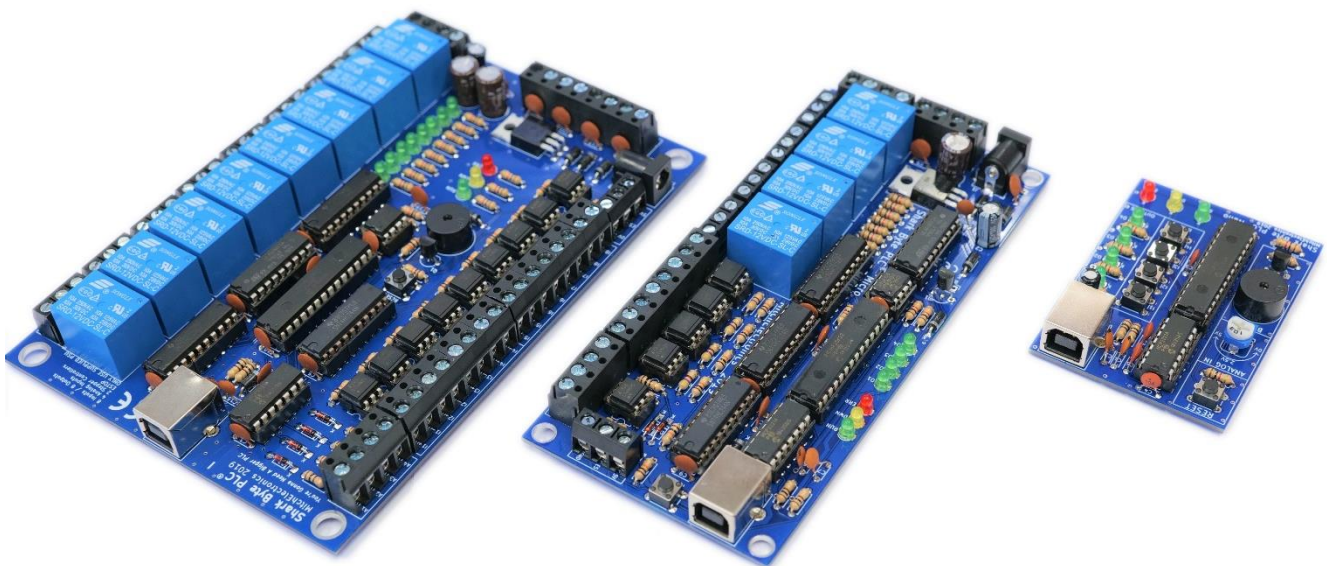
# Shark Byte PLC<sup>®</sup>

Educational Industrial PLC

MitchElectronics<sup>®</sup> 2019



Revision 1



*Shark Byte PLC<sup>®</sup> and MitchElectronics<sup>®</sup> are registered trademarks*

## Contents

Introduction.....	4
The Product Range.....	5
Hardware Specification .....	5
Software Specification .....	5
The Shark Byte PLC Nano .....	6
The Shark Byte PLC Micro .....	7
The Shark Byte PLC I.....	8
Basic Hardware Usage (Micro and I only).....	9
Inputs.....	9
Outputs.....	9
ESTOP .....	9
Analog Input .....	9
Stepper Motor Output .....	9
Power .....	10
USB .....	10
The Shark Core.....	11
L2 Software.....	14
Introduction .....	14
Connecting your PLC .....	15
Open an example .....	15
Basic controls .....	15
PLC Stats .....	16
Element drop-down list.....	16
Changing Element Values / Seeing Values.....	17
Ladder Logic – How it works .....	18
Transparent Elements .....	19
Ladder Logic Elements .....	20
L2 Examples .....	24
Simple Input Example .....	24
AND Gate Example.....	24

---

OR Gate Example .....	24
NOT Gate Example .....	25
One Second Timer .....	25
Count To ten Example .....	26
Analog Example.....	26
Other Information.....	27

## Introduction

The Shark Byte PLC range of products are educational and prototyping PLC controllers that include many of the commonly used functions on industrial PLCs such as those sold by Allen-Bradley and Siemens. However, unlike industrial PLCs, the Shark Byte PLC range is designed with students and makers in mind and as such differs in a number of ways. Firstly, the entire range of PLCs use only through-hole technology which makes them available in kit form as well as being easy to understand (i.e. students can see each component and understand its purpose). Secondly, some advanced features such as scripts, custom coded functions, and networking are not supported. Thirdly, the PLC range is not certified for industrial use but instead designed to be used in DIY and educational projects to give students and engineers a better understanding of what PLCs do.

The Shark Byte PLC range of products have many advantages over industrial PLCs including

- USB COM port that provides a superior connection experience over industrial PLCs (which can be intermittent)
- Considerably cheaper than their industrial counterparts allowing each student their own personal PLC
- Software is intuitive, simple to use, and entirely free with no licensing required



The Shark Byte PLC range of products themselves can work on Windows, Apple, and Linux but the software runs only on Windows. To run the software on other platforms such as Linux then a Windows emulator (such as Wine) will be needed.

This manual describes how the Shark Byte PLC range of products work as well as how to use the L2 software.

## The Product Range

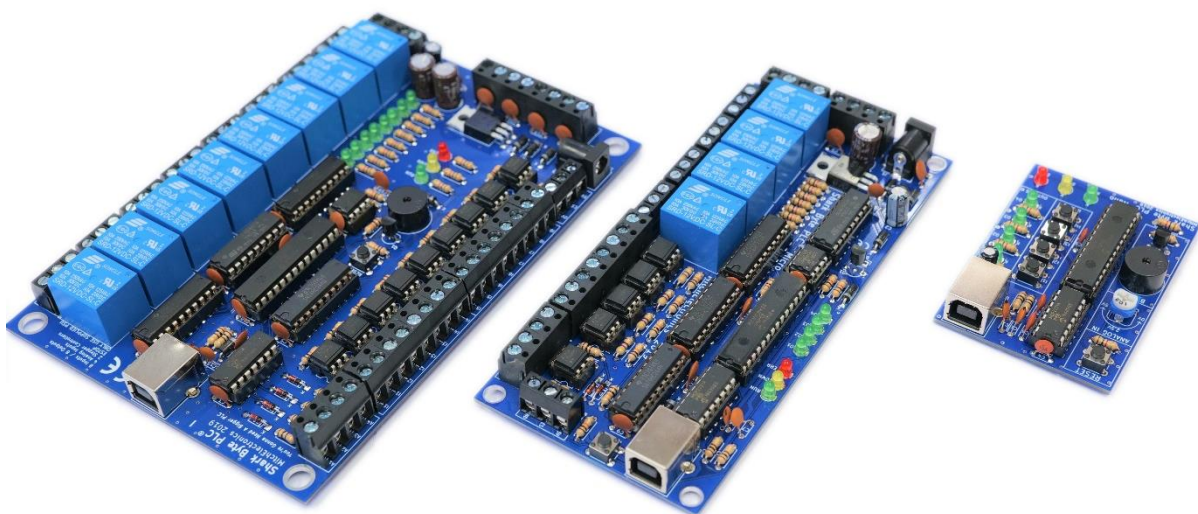
As the product range currently stands (August 2019), there are three PLC systems; the Nano, the Micro, and the I. The Micro and I are fully functional education PLCs that include output relays for controlling loads, input isolators for reading signals, analog inputs for sensors, and stepper motor output signals for controlling stepper motors. The Nano, however, is a training PLC that is designed for sketching designs and quickly prototyping ideas and as a result only has input switches and output LEDs.

### Hardware Specification

Product	Number Inputs	Number Outputs	Number Analog inputs	Number stepper	Buzzer output	Protected inputs	Relay outputs	Output LEDs	USB	Memory
Nano	4 (Switches)	4 (LEDs)	1 (Potentiometer)	0	Yes Integrated	No	No	Yes	Yes (Type B)	No
Micro	4	4	2	2	Yes External	Yes	Yes	Yes	Yes (Type B)	Yes
I	8	8	4	2	Yes On board	Yes	Yes	Yes	Yes (Type B)	Yes

### Software Specification

Product	Number elements	Number counters	Number timers	Number latches	Number Edges
Nano	32	2	2	2	2
Micro	64	4	4	4	4
I	256	16	16	16	16

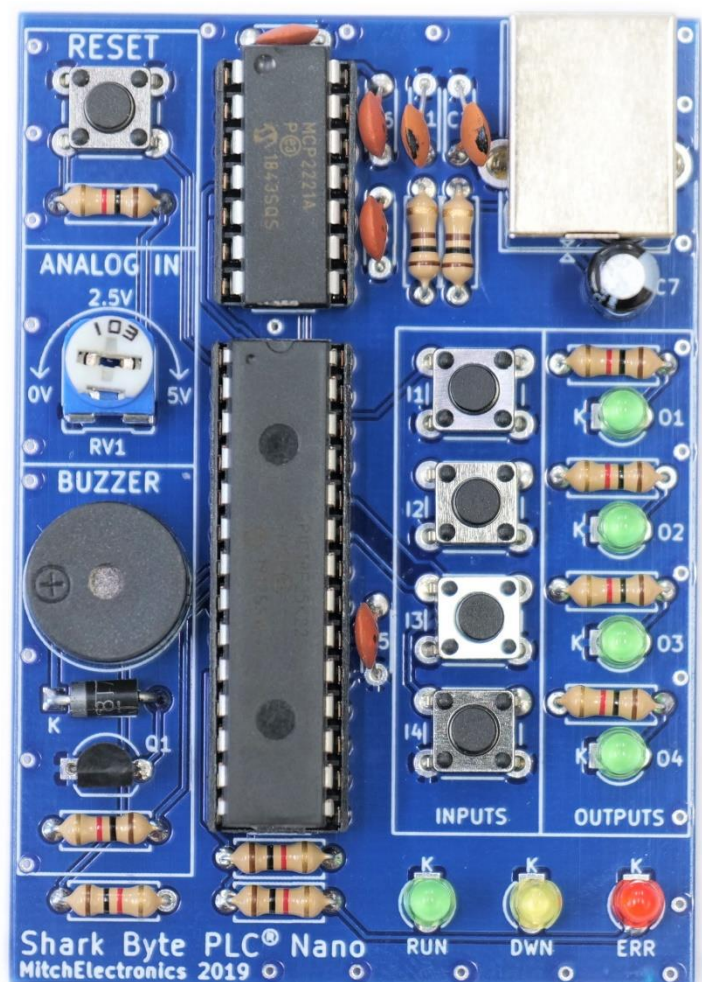
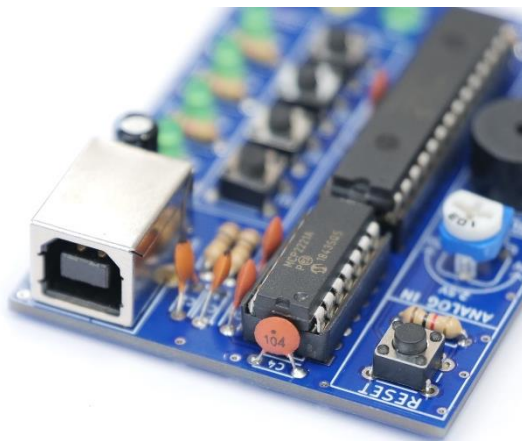
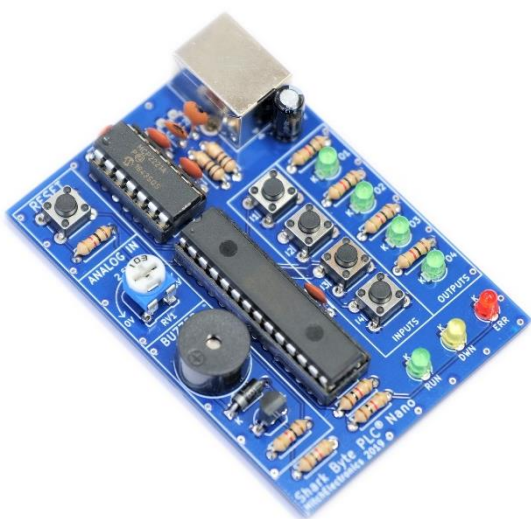


*The three main products: Shark Byte PLC I, Shark Byte Micro, and Shark Byte Nano*



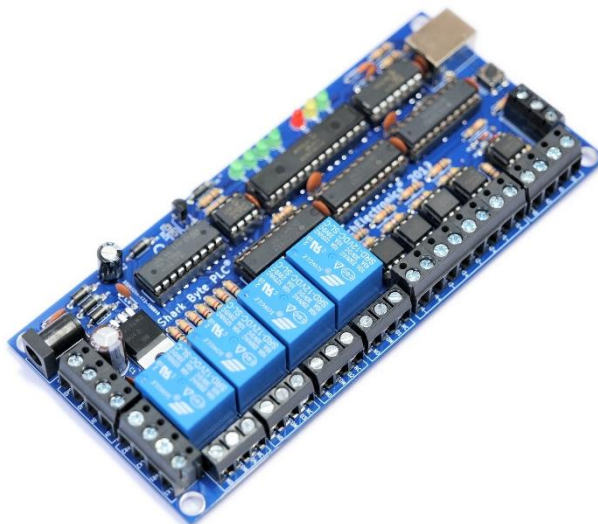
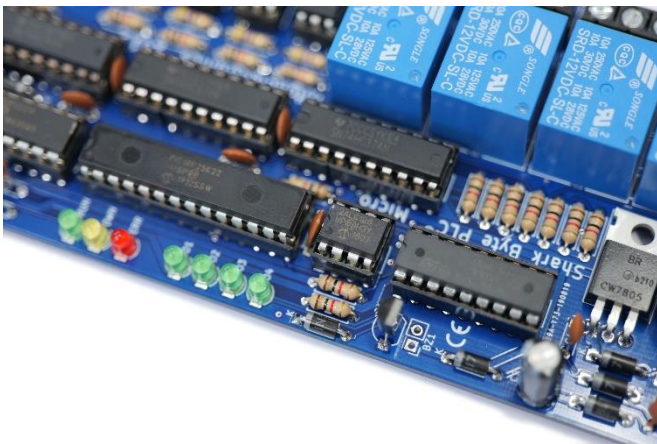
## The Shark Byte PLC Nano

The Shark Byte PLC Nano is a small handheld sketch PLC that includes 4 input buttons, 4 LED outputs, a potentiometer analog input, and an output buzzer. The PLC does not require an external 12V power supply and instead obtains its power from a USB connection. Unlike the other PLCs, the Nano does not have a memory chip and therefore forgets its program when disconnected from USB. The purpose of the Nano is to provide a low-cost solution for students that wish to use a PLC system at home who only need to write small ladder logic programs. The Nano is also an ideal kit for students who are required to solder for an electronics course as there are only a handful of components. Unlike the Micro and I, the Nano does not use a standard PLC core but instead a totally custom core that reads input and writes to outputs directly without the need for input and output buffers.



## The Shark Byte PLC Micro

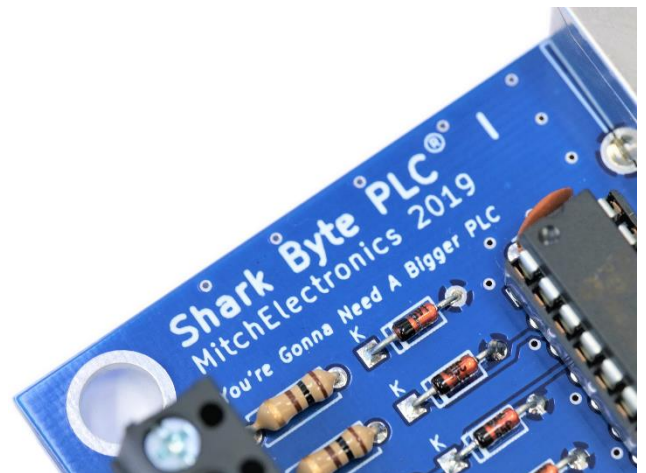
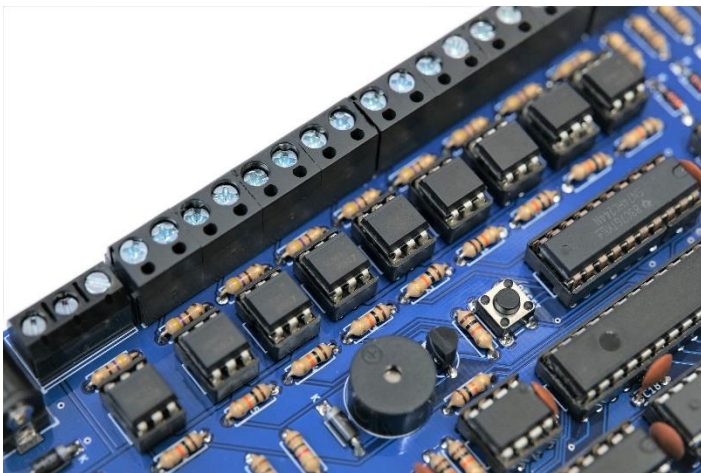
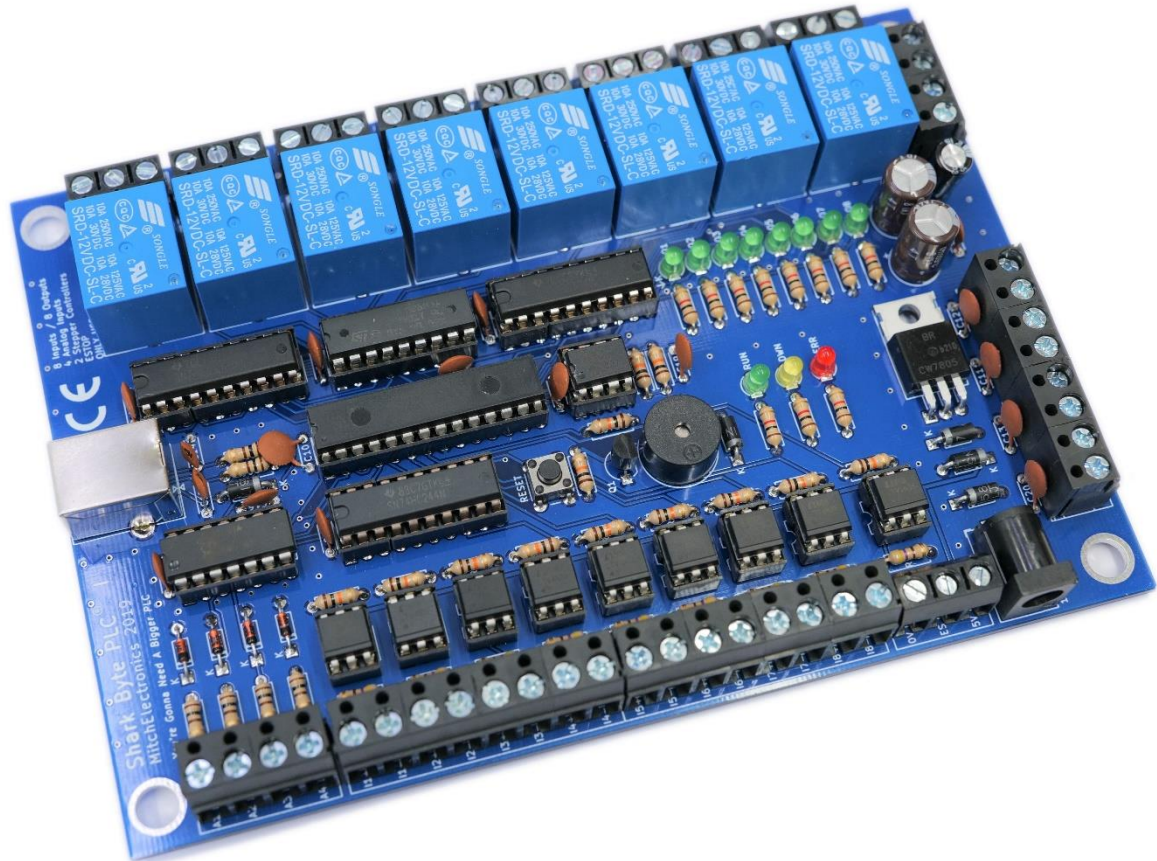
The Shark Byte PLC Micro is the smallest PLC in the mainstream range which are targeted for education and makers. The Micro includes 4 inputs, 4 outputs, 2 analog inputs, an output buzzer signal, two stepper motor controllers, and an ESTOP. Unlike the Nano, all the inputs and outputs to the Micro are via screw terminals which makes integrating into projects simplified. The inputs to the Micro are opto-isolated which protects the Micro from voltage spikes as well as making it compatible with a wide range of logic levels. Each output of the Micro has a relay that can be used to control motors, lamps, and other high-powered devices and each relay has a corresponding LED which makes identifying relay signals trivial.





## The Shark Byte PLC I

The Shark Byte PLC I is the upgraded version of the Micro and has 8 inputs, 8 outputs, 4 analog inputs, an onboard buzzer, two stepper controllers, and an ESTOP. Like the Micro, all inputs are opto-isolated protecting the I while all outputs have relays for controlling high powered devices. Not only does the I include more hardware but it is also expanded in its software and includes more timers, counters, and latches allowing for complex designs.





## Basic Hardware Usage (Micro and I only)

Using the Shark Byte PLC range has been designed with students and makers in mind. With the exception of USB and power input, all connectors are screw terminal making them easy to integrate into any project.

### Inputs

The inputs to the PLC use opto-isolators and each input has a series current limiting resistor. Traditional inputs will expect a digital signal (typically 5V and 0V) but the use of opto-isolated inputs allows for a wide range of input voltage (3V to 20V). The use of isolators also allows for a wide range of sensor types to access the PLC without interfering with other sensors. Each input has a + and – terminal which related to the polarity of the signal. Devices that connect to an input need to apply power across these two terminals to produce a valid input signal and the circuit diagram below shows how this can be achieved (typically, - will be grounded and a digital signal sent to the + pin).

### Outputs

Outputs of the PLC range are relay switch contacts including the common, normally-open, and normally-closed. As such, the outputs are electrically isolated from the PLC which means they can be made to work at any polarity and voltage range (so long as the voltage range does not go beyond the limits of the relays). Outputs can be read (using the ladder logic software), however the output relay itself is not read; the control signal to the relay is read. Each output on the PLC range also has an indicator LED to show which outputs are currently on / off.

### ESTOP

The PLC range (except the nano) have an external ESTOP signal which **MUST BE CONNECTED** for the system to work. The ESTOP signal is a normally high signal and as such must be connected to a switch that, when pressed, breaks the connection and latches itself. For prototyping the ESTOP can be simply shorted to 5V (provided on the connector) and the ESTOP signal is isolated from the main PLC via an opto-isolator.

### Analog Input

Analog inputs are one of the few connectors that are not isolated and as such users must take care when using them. Protection circuitry is provided (using a current limiting resistor and zener diode per channel), but this is not a guarantee. Analog signals are between 0V and 5V with any voltage greater than 5.1V being clamped by the zener diode.

### Stepper Motor Output

Stepper motor outputs are not isolated from the PLC and as such care should be taken when using them. These outputs provide two connections per stepper motor channel; a direction and a clock. The direction signal indicates which way the stepper motor should rotate while

the clock signal indicates a single step on the motor. This output is ideal for use with stepper motor drivers such as the A4988.

### Power

Power to the PLC is provided with the use of a 5.5mm x 2.1mm DC jack that supplies 12V and 2A minimum. It is essential that only the supplied power supply is used otherwise the warrantee of the PLC is voided. The board can operate from only the USB connection but external relays do not operate (as they require the 12V). However, this allows designers to upload designs to the PLC before having to have the relays operate. Multiple power connectors are also available on the PLC and these are either 5V or 0V which are designed for use with external sensors.

### USB

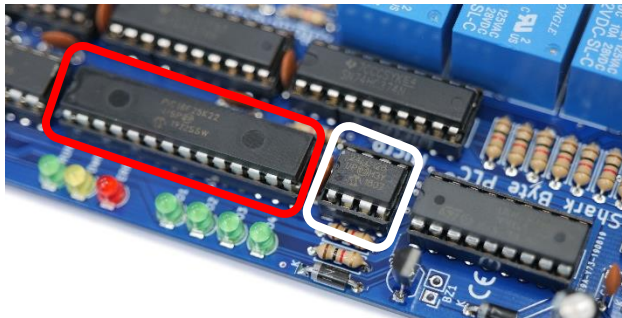
The USB input to the PLC range is a USB B type and the cable length should be less than 5M. This input provides 5V to the PLC which allows the PLC to be programmed, reading of inputs, and controlling stepper motors but it cannot be used for powering the relays. When the PLC is connected to an external 12V DC jack the USB connection is no longer required.

## The Shark Core

At the heart of the PLC is the Shark Core which can be thought of as the brain of the system. This core is a microcontroller that executes ladder logic programs and performs all the important tasks including

- Reading Inputs
- Writing Outputs
- Checking internal statuses and errors

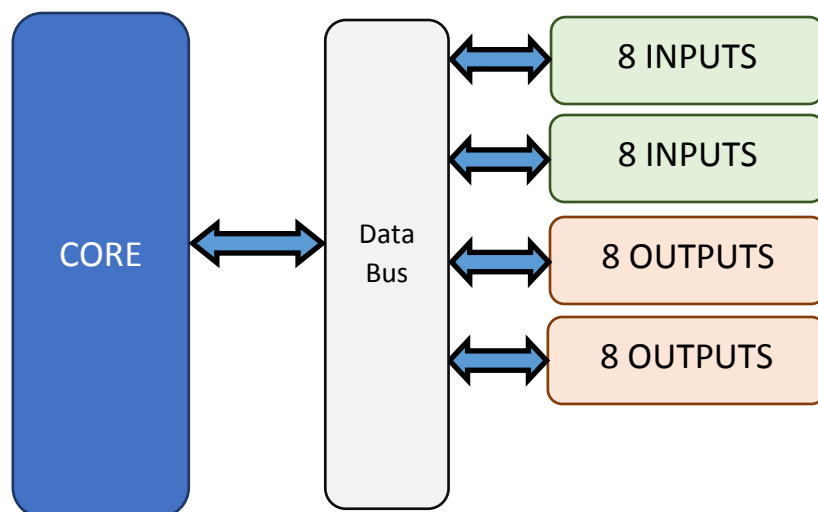
When the core first starts it looks to see if there is an external memory chip (using the I2C protocol) and if one is detected, it will load the ladder logic program stored in that chip. If no chip is detected then the core will halt and throw an error via the error LED output (the system cannot function without a memory chip). Once the ladder logic program is loaded it infinitely executes the ladder logic program until either the power is removed or an ESTOP signal is detected.



Core (red box)

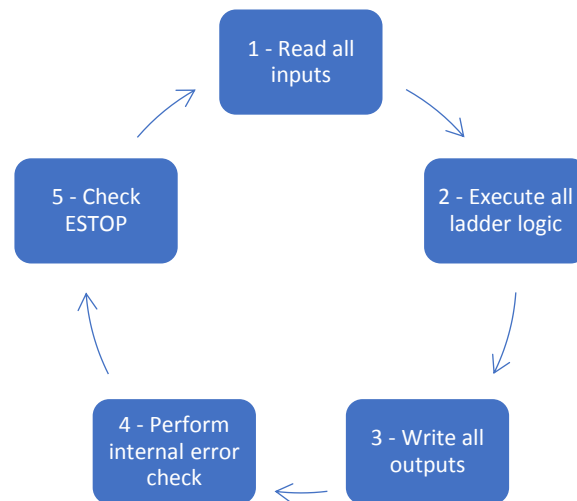
Memory (white box)

The core is capable of reading many inputs as well as writing many outputs but the core only has 28-pins. To solve this issue, the core operates in a very similar fashion to a CPU whereby inputs and outputs share a common data bus that uses latches and buffers to enable / disable IO ports. Below is a simplified diagram of the Shark Byte PLC architecture demonstrating the data bus and the different IO ports.





If, for example, an input needs to be read, the core will deselect all ports and then enable the input port for reading. The input data is then transferred to the data bus which in turn gets transferred internally to the core. If an output needs to be written to then the core will deselect all ports, place the data that is to be written to the desired output, and then send a write signal to the output port that is being written to. This allows for the output to retain its value even when the core is performing other tasks and controlling different ports. Execution of the ladder logic program is done as a cycle and the execution cycle is shown below.



## L2 Software

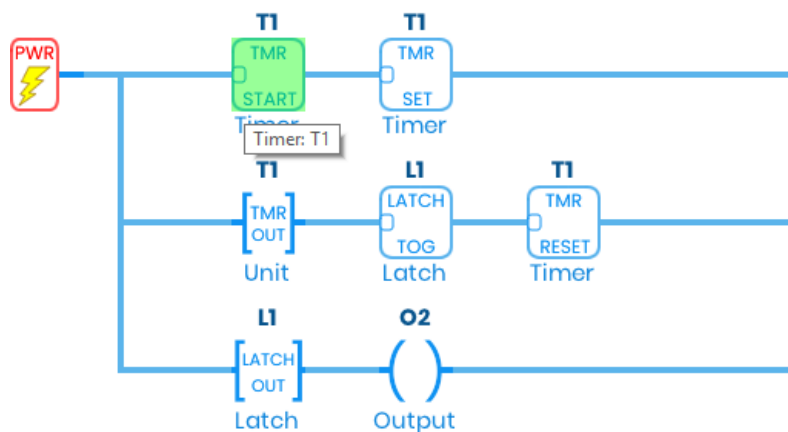
### Introduction

L2 (Ladder Logic) is the programming environment for the Shark Byte PLC range and uses ladder logic to represent industrial control processes. Unlike other software packages, L2 is designed to be easy to use and as such has the following features

- Large modern icons
- Snap to grid functional
- Auto-joining of neighbouring wires
- Simplified PLC connection
- Auto-scan on start up for connected PLC
- One button run – Saves, compiles, and downloads in one operation

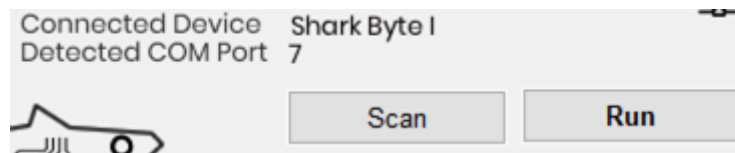
L2 offers a wide range of commonly used elements to provide designers with flexibility and each function specific is dedicated into a single element. For example, other PLC software suites will often have all timer signals (such as reset and stop) on a single element whereas L2 has one element for each function. This makes designs easier to read as well as allowing a single element type to span an entire design with ease.

**Note – L2 requires a minimum screen size of 1280 px**



## Connecting your PLC

The first step when using the L2 software is to connect your PLC to the computer running L2. The PLC uses the MCP2221 USB bridge which is automatically recognised by Windows 10 so there should be no need for any driver installations. Once connected, launch the L2 software so that it automatically connects to it on start up. If you do not connect your PLC to the computer before launching L2 then you will need to press the scan button on the L2 software once you have connected your PLC to a valid USB port.



## Open an example

L2 comes with multiple examples for each device in the Shark Byte PLC range and these can be found in the Examples folder in the installation directory. Normally, L2 is installed under C:\Program Files\L2 but this may be different depending on the folder selected during the installation. Once an example has been loaded the program can be downloaded and executed by pressing the Run button found next to the Scan button (top right of the window). A black window with text will appear (this is the compiler / downloader) and this window will close automatically. Once it closes the PLC will run the program automatically. Be aware that the PLC will stop execution if you scan the PLC with the L2 software. In this case, press the reset button on the PLC to restart program execution.

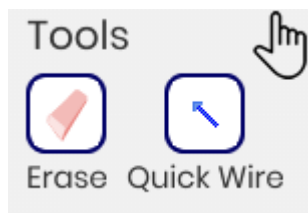
## Basic controls

L2 is designed to be used with a mouse and keyboard and has an intuitive set of controls. Most actions are done with the mouse while the keyboard is mainly used for moving the design.

- Left click element – Pick up / drop
- Right click element – Open menu
- Right click element when picked up – Delete element
- Down arrow – Scroll down
- Up arrow – Scroll up
- Mouse wheel up – Scroll up
- Mouse wheel down – Scroll down
- Ctrl + S – Save as

Two additional tools are provided with L2 which greatly decrease the time needed to make ladder logic designs. The first tool is the eraser which, when selected, can be used to rub out elements very quickly by simply holding the left mouse button and moving the eraser icon over elements that need to be removed.

The second tool is the quick draw which is a very convenient wire drawing tool. When selected the tool will draw wire pieces wherever the mouse pointer is when the left mouse button is held.



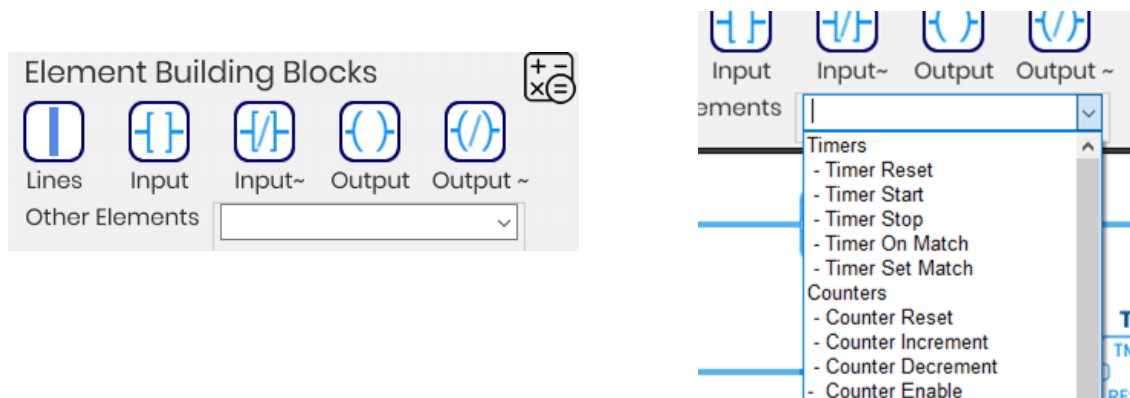
### PLC Stats

Different devices in the Shark Byte PLC Range have different capabilities with the nano being the smallest and the I (as of August 2019) being the largest. When a device is successfully detected by L2 the stats section will be updated with what that device contains. The stats section also shows how many elements the current design has used and this number must not exceed the max elements stat. Elements include all items in a design except for wires (such as timer reset, input, output, counter increment, and buzzer).

PLC Stats				
Digital Inputs	8	Timers	16	
Digital Outputs	8	Counters	16	
Analog Inputs	4	Latches	16	
Stepper Outputs	2	Max Elements	128	
Total Elements Used		7		

### Element drop-down list

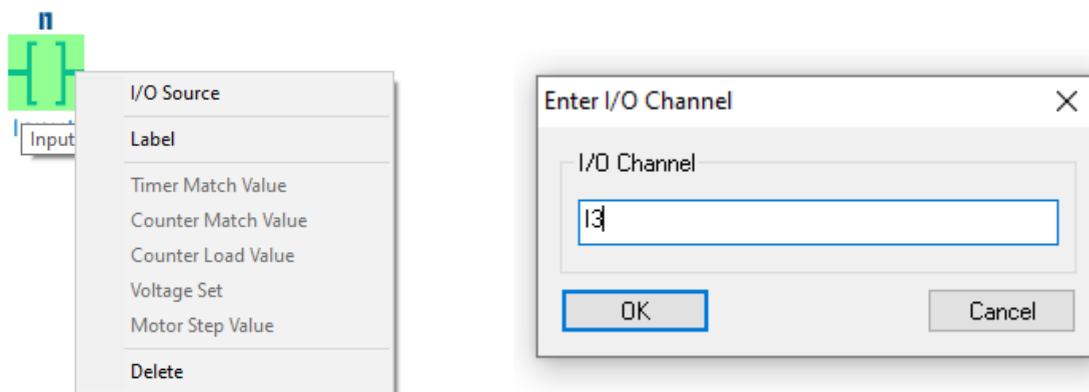
The most commonly used elements are found in the top left of the designer and include wires, inputs, and outputs. All other elements are found in a drop-down list just below these elements and these can be inserted into the design by expanding the drop-down list and then selecting the desired element.





## Changing Element Values / Seeing Values

Element values (such as IO number) can be changed via a pop-up menu. This menu is accessed by right clicking an element which has been placed on the designer.



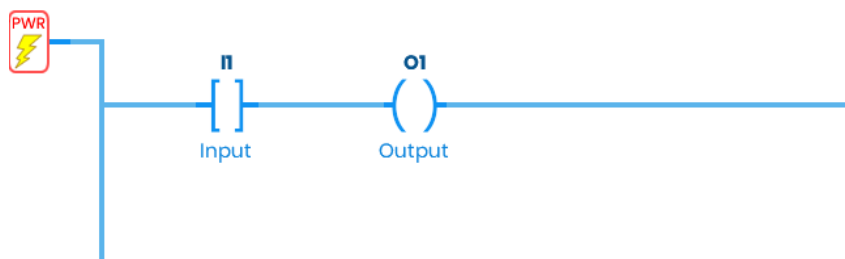
Element values (such as voltage set and counter load) can be see by hovering the mouse pointer over the element in question.



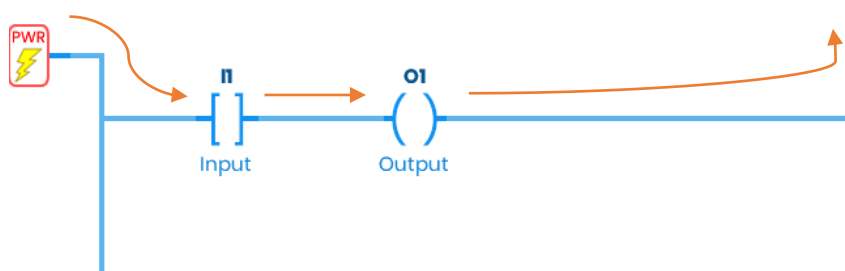
## Ladder Logic – How it works

Ladder logic is the “language of PLCs” and makes designing industrial control processes very easy. The first PLC systems would use standard microcontroller code such as assembler which was OK if the designer was a seasoned programmer or the process being controlled was trivial. However, as industrial processes became more complex the code-based system quickly became obsolete and inefficient. Instead, the industry turned towards a graphical method of programming which is not only easy to program but to also understand by both the designer and anyone in the future who needs to look at the design.

Ladder logic, as in the name, consists of ladder rungs that contain control logic symbols which can let signals pass through them if some condition is met or perform an action when a signal is received. The example below shows a simple switch connected to an output which, in this case, will be an LED.



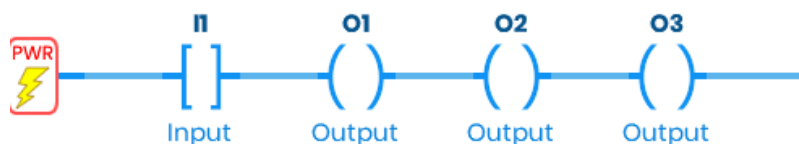
The left side of the ladder (connected to the PWR element) is always powered and represents a logical 1 (on). The input can pass this power onto its output (the right side) but this will only happen if the input is on (i.e. the switch is pressed). If the switch is pressed then the power from the PWR element is transferred through I1 into the input of O1. When this happens, the LED turns on (which represents O1), and O1 transfers this power to the right side of the ladder. Ladder logic flows up to down and left to right so in the example above power from the right rung cannot go back into O1.



### Transparent Elements

If an element does not process signals then it will be transparent to them. For example, the Output element reads the signal to its left and then turns the output on or off depending on this value. However, as it does not affect this signal it will pass it through itself to the right side as if it was a wire.

The example below demonstrates element transparency. When the input I1 detects a signal the output O1 turns on. However, O1 also passes this signal to its right side and this feeds the next output O2. Overall, all three outputs will turn on.



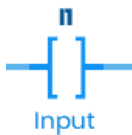

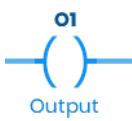





## Ladder Logic Elements










In L2, elements are the functional blocks including inputs, outputs, timers, and counters while nets are the wires that connect different elements together. Each PLC device can only hold n number of elements and processes m number of nets (where n and m depend on the PLC).


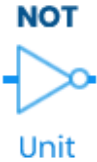







Each element has an IO number above which corresponds to the specific element. For example, all elements with the IO number “T1” will interact with timer 1. Another example would be “O1” which would be elements that interact with output 1. The table below shows the valid IO numbers for the different Shark Byte PLCs.


Element	Shark Byte Nano	Shark Byte Micro	Shark Byte I
<b>Input</b>	I1, I2, I3, I4	I1, I2, I3, I4	I1, I2, I3, I4, I5, I6, I7, I8
<b>Output</b>	O1, O2, O3, O4	O1, O2, O3, O4	O1, O2, O3, O4, O5, O6, O7, O8
<b>Analog Input</b>	A1	A1, A2	A1, A2, A3, A4
<b>Timer</b>	T1, T2	T1, T2, T3, T4	T1, T2, T3, T4, T5, T6, T7, T8, T9, T10, T11, T12, T13, T14, T15, T16
<b>Counter</b>	C1, C2	C1, C2, C3, C4	C1, C2, C3, C4, C5, C6, C7, C8, C9, C10, C11, C12, C13, C14, C15, C16
<b>Latch</b>	L1, L2	L1, L2, L3, L4	L1, L2, L3, L4, L5, L6, L7, L8, L9, L10, L11, L12, L13, L14, L15, L16
<b>Stepper Motor</b>	Not Available	M1, M2	M1, M2
<b>Edge</b>	E1, E2	E1, E2, E3, E4	E1, E2, E3, E4, E5, E6, E7, E8, E9, E10, E11, E12, E13, E14, E15, E16



Element	Looks Like	Function
<b>Input</b>		This element represents an input and will pass a signal if the specified input is a logical 1 (i.e. power signal into the input). If the input is a logical 0 then this element will not pass the signal to the left of the element.
<b>Input Inverted</b>		This element is identical to the input element except it is an inverted signal. The signal to the left of this element will be passed through if and only if the specified input is a logical 0 (i.e. no power signal into the input).
<b>Output</b>		This element represents an output and if the signal to the left of the output element is a logical 1 then the corresponding relay will switch on (this will connect the COM pin to the NO pin). If the signal to the left of this element is a logical 0 then the relay will switch off (this will connect the COM pin to the NC pin).
<b>Output Inverted</b>		This element is identical to the output element except that if the signal to the left of the output element is a logical 1 then the corresponding relay will switch off whereas if the signal to the left of this element is a logical 0 then the relay will switch on.
<b>Timer Reset</b>		This element will reset the specified timer when the signal to the left of the element is a logical 1.
<b>Timer Start</b>		This element will start the specified timer when the signal to the left of the element is a logical 1.
<b>Timer Stop</b>		This element will stop the specified element when the signal to the left of the element is a logical 1.
<b>Timer Out</b>		This element works similarly to the input element in that it will pass the signal from the left of the element to the right of the element when the specified timer value goes beyond a specified number (defined by the Timer Set element).

<b>Timer Set</b>		This element sets the timer match value which controls the timer out element. When the logic signal to the left of this element is 1 then the timer out element will pass a signal when the timer is greater or equal to the value specified by this element.
<b>Counter Reset</b>		This element will reset the specified counter to 0 when the signal to the left of the element is a logical 1.
<b>Counter Increment</b>		This element will increment the specified counter when the signal to the left of the element rises from a logical 0 to a logical 1. If the input remains a logical 1 the counter does not repeat counts (it is edge triggered).
<b>Counter Decrement</b>		This element will decrement the specified counter when the signal to the left of the element rises from a logical 0 to a logical 1. If the input remains a logical 1 the counter does not repeat counts (it is edge triggered).
<b>Counter Load</b>		This element will load the specified counter with a specified value when the signal to the left of the element is a logical 1. This is NOT an edge triggered device.
<b>Counter Equal</b>		This element works in a near identical way to the input element whereby the signal to the left of the element is passed through if the specified counter is equal to a specified value.
<b>Counter Less Than</b>		This element works in a near identical way to the input element whereby the signal to the left of the element is passed through if the specified counter is less than the specified value.
<b>Counter Greater Than</b>		This element works in a near identical way to the input element whereby the signal to the left of the element is passed through if the specified counter is greater than the specified value.
<b>Analog Less Than</b>		This element will pass the signal to the left of the element when the voltage at the specified analogue channel is less than the specified value.

<b>Analog Greater Than</b>		This element will pass the signal to the left of the element when the voltage at the specified analogue channel is greater than the specified value.
<b>Not Gate</b>		This element will invert the signal to the left of the element. For example, if the input is 0 then the output will be 1.
<b>Latch Set</b>		This element will set the specified latch when the signal to the left of the element is a logical 1. This is NOT an edge triggered element.
<b>Latch Clear</b>		This element will clear the specified latch when the signal to the left of the element is a logical 1. This element is NOT an edge triggered element.
<b>Latch Toggle</b>		This element will flip the state of the specified latch when the signal to the left of the element is a logical 1. This element is NOT an edge triggered element.
<b>Latch Out</b>		This element works in an identical way to the input element for the specified latch. If the latch is set then this element will pass the signal to the left of the element.
<b>Clockwise Step</b>		This element will step the specified stepper motor port once in a clockwise motion (via the DIR signal). This element is NOT edge triggered.
<b>Anticlockwise Step</b>		This element will step the specified stepper motor port once in an anti-clockwise motion (via the DIR signal). This element is NOT edge triggered.
<b>Buzzer</b>		This element will sound the external buzzer when the signal to the left of the element is a logical 1.

<b>Edge</b>		<p>This element will output a brief logical 1 when the input rises from 0 to 1. Even if the input remains at 1, the output of the edge stays at a logical 0 and can be used for providing edge detection to elements that do not have an edge input. Note that edge elements are limited and their IO number indicates the edge element.</p>
-------------	---	--

## L2 Examples

The following examples show how to use the elements available in the Ladder Logic software.

### Simple Input Example

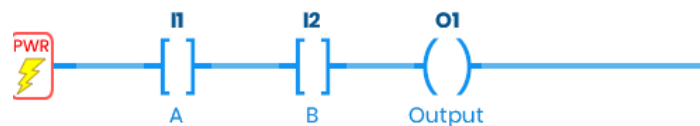
This example will turn on O1 when a signal is detected on I1.



*Simple Input Example*

### AND Gate Example

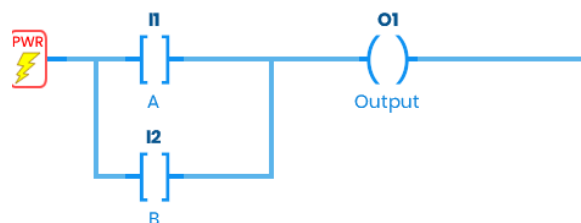
This example shows how two inputs (I1 and I2) can be used as an AND gate. The output will only be on if both inputs detect a signal.



*AND Gate Example*

### OR Gate Example

This example shows how two inputs (I2 and I2) can be used as an OR gate. The output will be on if either input detects a signal.

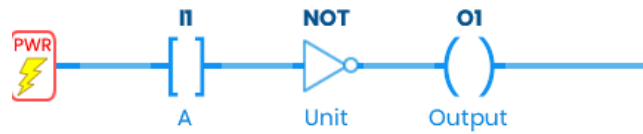


*OR Gate Example*



### NOT Gate Example

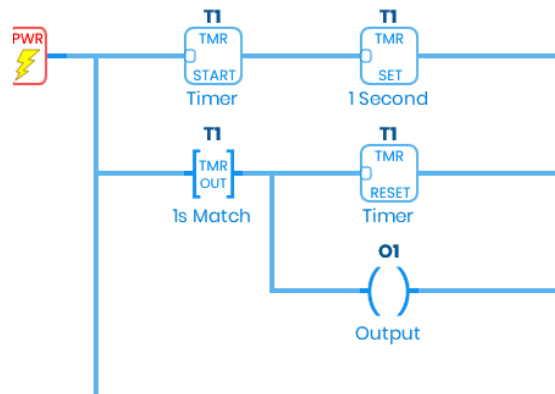
This example demonstrates how the NOT element can be implemented. When the input I1 detects a signal then the output will be off and if no signal is detected then O1 will be on.



*NOT Gate Example*

### One Second Timer

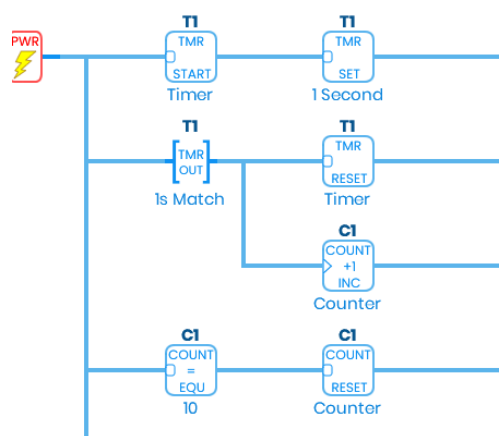
This example demonstrates a simple 1 second timer. The Timer Start and Timer Set elements are always triggered and when the timer reaches 1 second the TMR OUT element triggers the Timer Reset element as well as the output. When the timer reset element is triggered the timer is reset to 0 and this causes the output to pulse (similar to a clock signal).



*One Second Timer*

### Count To ten Example

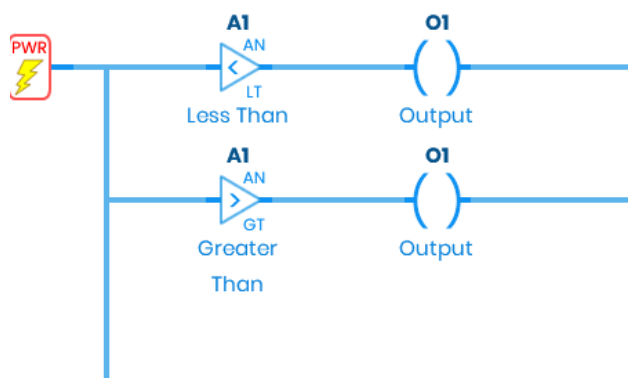
This example demonstrates how to use the counter element to count to 10 and then reset back to 0. The one second timer example (shown previously) is used to increment the counter (C1) and when the counter value is equal to 10 the Counter Reset element is triggered.



*Count-to-ten example*

### Analog Example

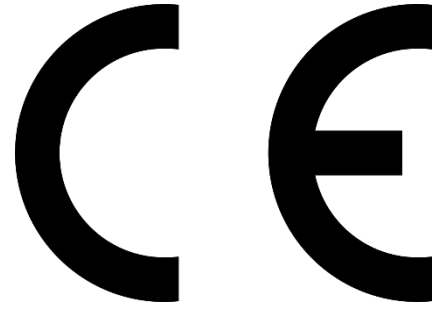
This example demonstrates how the analog input elements can be used to turn on outputs depending on the reading from the analog input port A1.



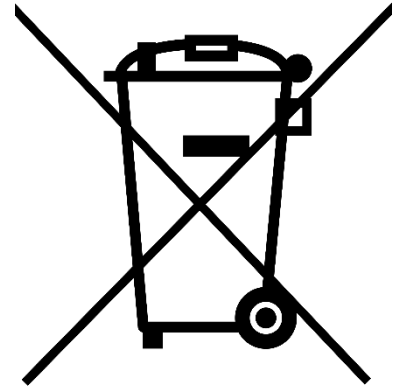
*Analog Example*

## Other Information

The range of Shark Byte PLCs was developed in compliance with the applicable European directives and therefore carries the CE mark. Its authorised use is described in this document and in the event of non-conforming use or modification of the product then you will solely be responsible for complying with any regulation that applies to your area (local law for example).



Waste electrical products should not be disposed of with household waste. Please recycle any MitchElectronics products responsibly and if in doubt, contact your local authority or retailer for recycling advice.



“Shark Byte PLC” and “MitchElectronics” are registered trademarks and are the sole property of MitchElectronics. These trademarks may not be reproduced without written permission from the trade mark owner.