L-Università ta' Malta

ESE 2023 Summer Training:

Automation and Prototyping

L-Università ta' Malta

ESE 2023 Summer Training:

Day 2:
Control Systems

Pneumatic Control Systems

The pneumatic system we have considered is the basic building block around which, a wide range of assembly processes depend.

However a pneumatic system without a controller is practically useless!
A complete system will need a controller that will determine the sequence and mode of operation.

In this short, 3-day course, it is impossible to enter into detail about controllers and programing, so we will just provide an overview of two main categories of controllers.

PLC (Programmable Logic controller)

The PLC has been around for about 40 years and it is still dominant in the control of pneumatic and various other systems. The reason for its continued success, is that major PLC manufacturers such as Siemens, Allen Bradley, Mitsubishi etc. invested heavily in continuously upgrading their line of products such that today's PLCs can control practically any manufacturing process.

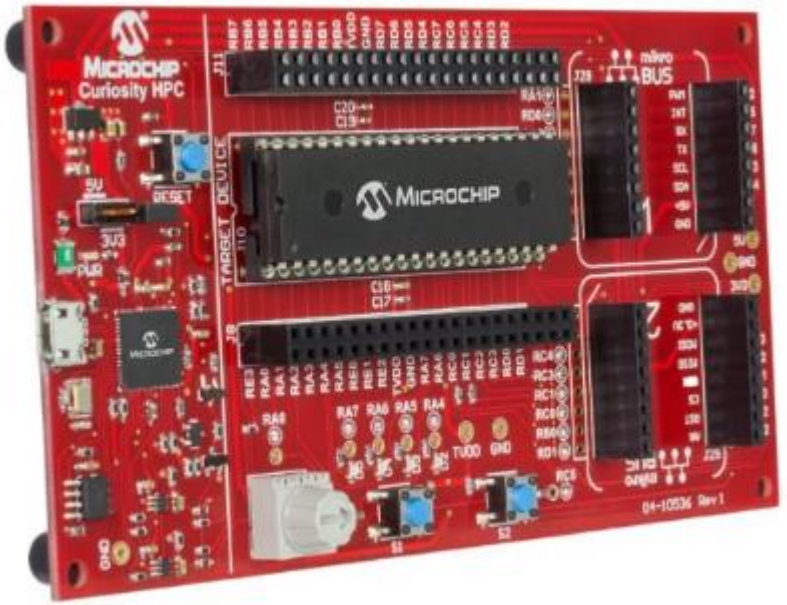
The main advantage of the PLC is that it is robust; very reliable and therefore well-suited for industrial operation, even in harsh environments. The main disadvantage of PLC systems is that, as always, Quality comes at a price, and as such, PLC systems tend to be quite expensive.

I have personally been involved in the setting-up of PLCs systems that have worked for over 20 years, working two shifts - and at times, even 3 shifts, with no trouble at all !

Other microcontroller systems:

Here we find a long list of controllers and among the more well-known makes, we could mention the Arduino; PIC and the Raspberry pi, (though the latter bears more resemblance to a mini-computer).

These low-cost systems are widely used for educational purposes, but it is also a fact that they are also extensively used as a controller in several non-critical applications.



Other microcontroller systems *(cont.)*

Provided that the application is not one that is safety-critical, then these controllers provide an excellent; and in many cases, a reliable, low-cost solution.

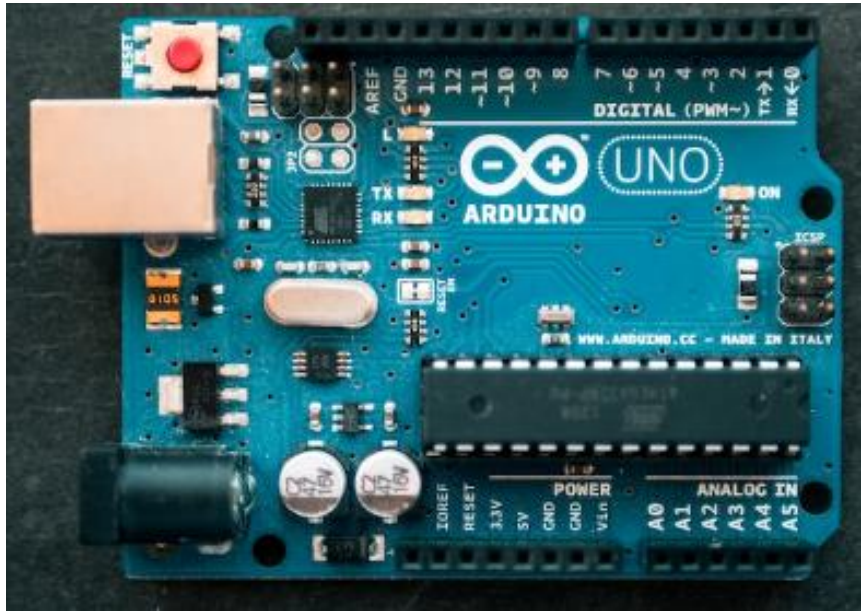
These microcontrollers are especially useful in Prototype systems. Concepts and ideas can be tested without having to commit to using an expensive PLC system which could be much better-utilised in an actual manufacturing process.

However, these should never be used for the control of industrial processes since they are not sufficiently mechanically and electrically robust to withstand harsh environments.

In this training program, we will be using an Arduino and we shall see how it can be used to:

1. Operate our pneumatic cylinder according to a predetermined sequence.

2. Perform a simple functional test on a push-button switch.



As we have already mentioned, the scope of this exercise is to provide you with a hands-on experience of how a complete, functional system can be put together.

We will only be guiding you through the set-up sequence to link your laptop to the Arduino.

The Arduino board – a quick Overview

1. Atmega Controller chip.

2. Digital input & output pins - these pins can be configured through the program to be either inputs or outputs.

3. Analog pins - to process analog signals from sensors such as thermistors, loadcells etc.

4. USB Connection for power up and communication via a PC or laptop.

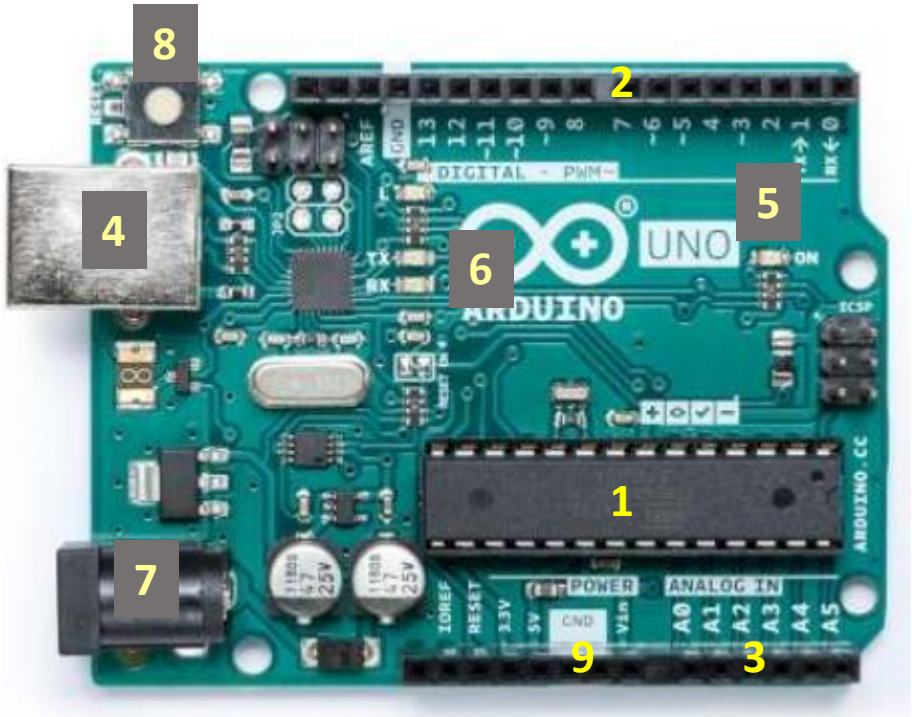
5. LED Power ON indicator.

6. TX/RX these LEDs flash when software is downloaded.

7. dc Power jack - used when the Arduino is powered from a PSU.

8. Reset button - very useful if you need to do a general reset and run the program from a fresh start.

9. Pins marked GND are ground, (0 Volt), connections.



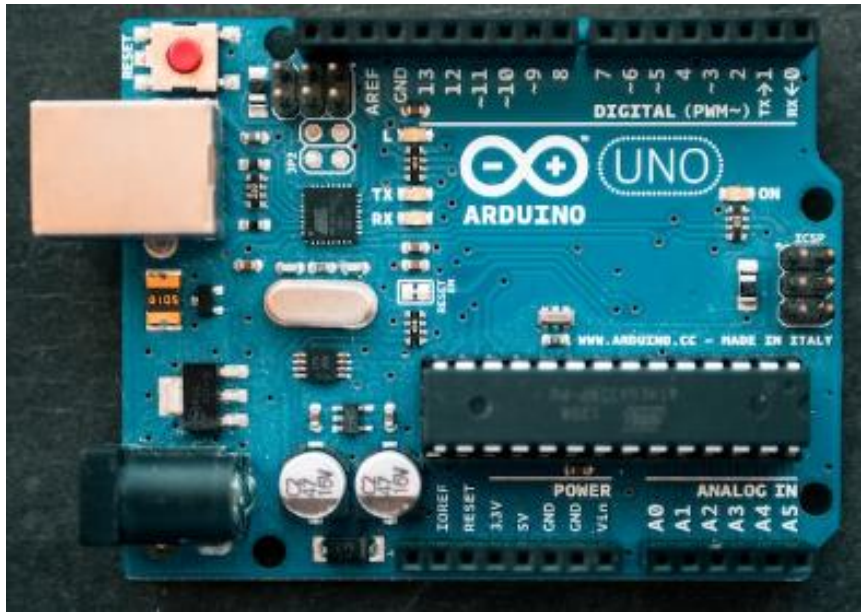
A few important notes about the Arduino.

Powering up:

The recommended input voltage range of the Arduino is 7V to 12V.

Applying a supply voltage of less than 7V, may lead to unstable operation while exceeding 12V, may result in overheating of the on-board voltage regulator. In any case, the supply voltage should not exceed 15V.

In our case, we will be supplying the Arduino with a 9V, regulated supply.



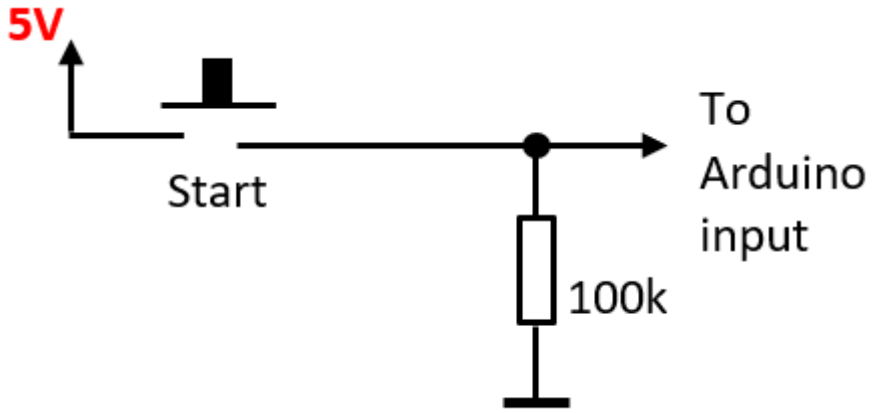
Input & Output Voltage levels

Note that any High (1) input signal voltage to the Arduino should never exceed 5V.
The Low (0) voltage level should be less than 1V.

Remember, also, that all inputs to the Arduino should be pulled down to ground, (0V), via a 100k resistor.

This is necessary to ensure that in the absence of a High signal, the Arduino input will be securely tied to the ground, (0V), level, otherwise the input will remain floating and operation is likely to become unstable.

In general, this principle applies in all digital logic work.

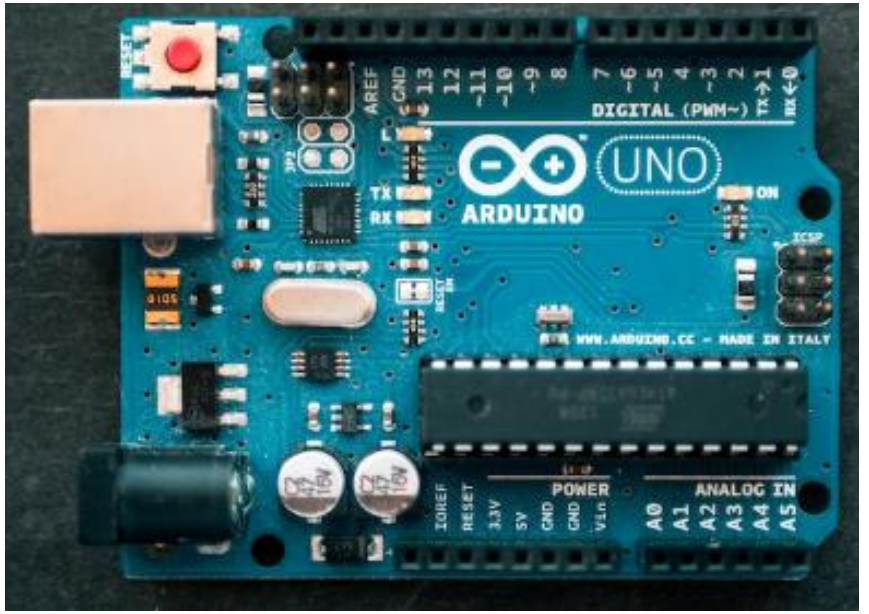


Inputs & Outputs (I.O.) – Operating voltage

Arduino’s onboard regulator sets the I.O. voltage levels to 5V.

This means that a ‘High’ at an Arduino output pin will be ≈ +5V while a ‘Low’ will be near to 0V.

Similarly, an input signal to an Arduino that represents a High, should not exceed 5V. Exceeding 5V can damage an Arduino’s input.



Important Considerations

In most industrial applications, the solenoids of the valves operate at a voltage of 24V dc.

Similarly, quite often, a sensor attached to a cylinder will also operate at 24V dc.

These voltage levels are obviously not compatible with the Arduino's 5V levels.

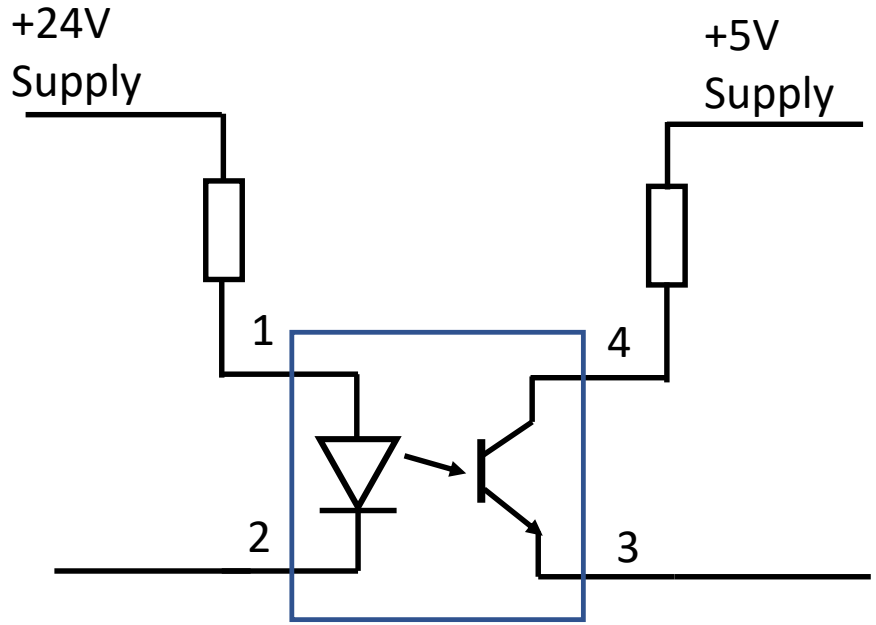
Ignoring these considerations might not have the catastrophic effect of the picture shown here, but may nevertheless, damage your Arduino!

Optocoupler - overview

Apart from ensuring that voltage levels are within limits, circuit protection would be greatly enhanced if we could isolate the inputs and outputs of the Arduino from the harsh environment of solenoids or other devices which generate considerable back emf, especially when de-energised.

An optocoupler IC will do just that job for us!

Pins 1 and 2 are the input side. When powered-up, the LED will shine on an optically sensitive transistor that will then conduct with pins 4 and 3 acting like a closed circuit.



Note that there is complete electrical isolation between the diode circuit and the transistor output.

Interface

Clearly then, we need some kind of interface that will provide the right voltage levels, at both ends:

1. Input signal levels from the cylinder sensors that work at 24V have to be brought down to 5V on the input side of the Arduino.

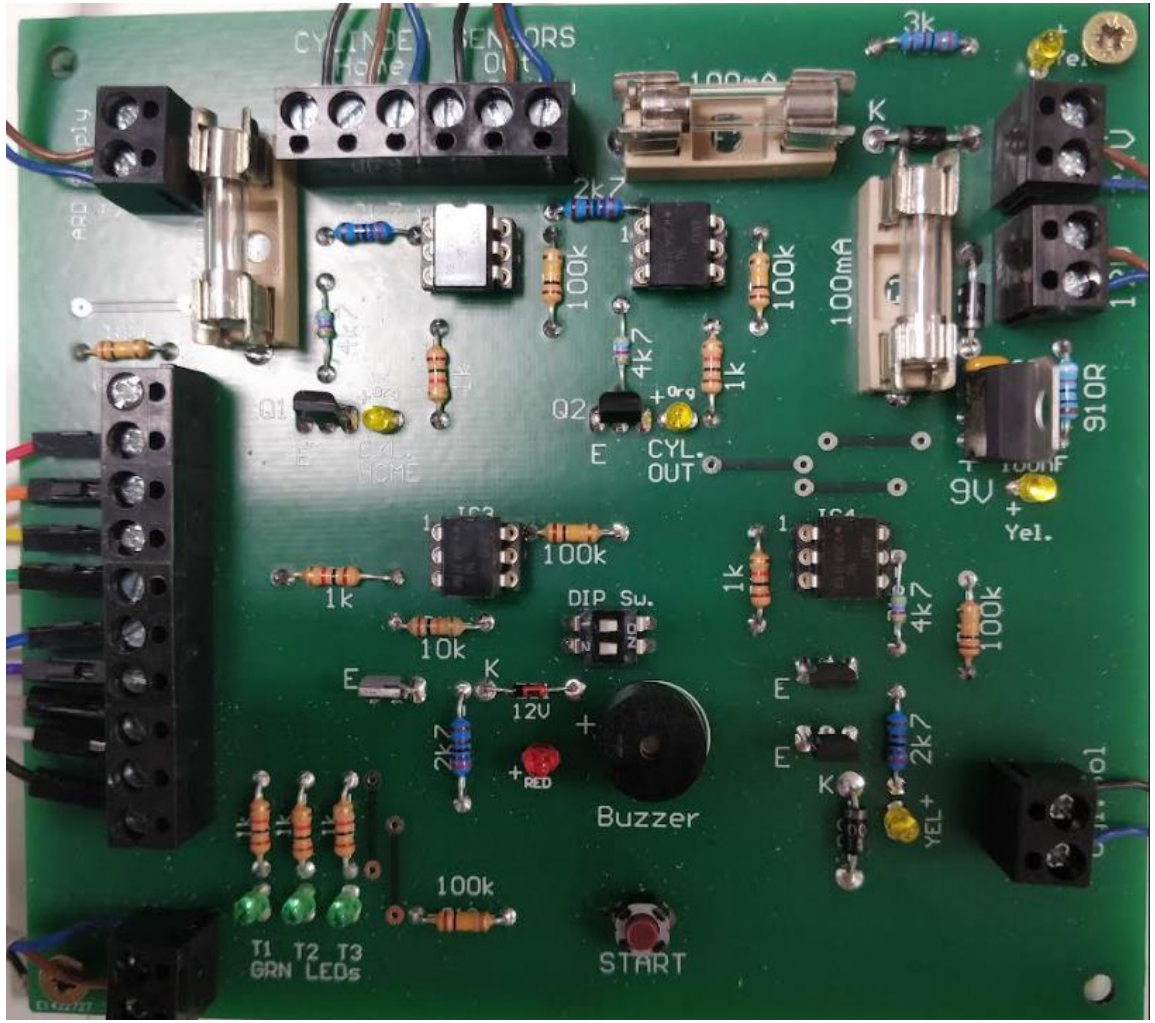
2. The Arduino 5V output has to be increased to 24V to drive the solenoid of the cylinder.

For this course we have designed our own interface board with the following considerations in mind:

1. The sensors on the cylinder operate at 24V. This is way beyond what the Arduino can handle as an input voltage. The board uses optocouplers to step down that 24V to the 5V level Arduino input requirement.

2. Similarly, we will be changing the 5V output voltage from the Arduino, to the 24V which the cylinder needs to operate.

3. Bipolar transistors are used at each optocoupler output to ensure that the O/P current limit of these ICs is not exceeded.



Interface board walk through:

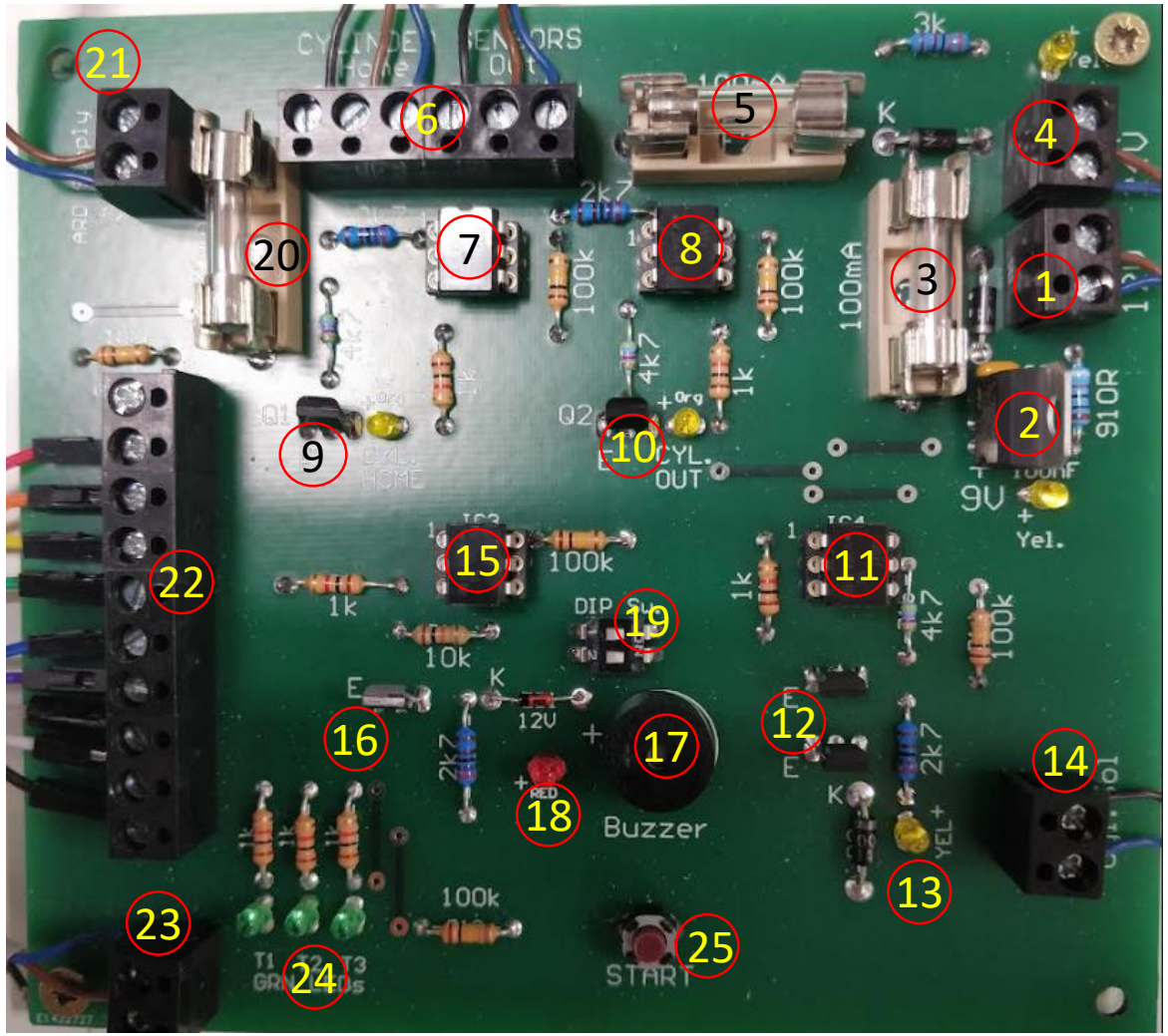
1. 12 V input terminal.

2. 9V voltage regulator IC supplies the Arduino with a stable, 9V supply voltage. The LED next to the regulator, indicates whether or not the 9V output is present.

3. A 100mA fuse protects the 9V regulator circuit.

4. 24 V terminal block powers cylinder sensors and the valve solenoid.

5. A 100mA fuse protects the 24V power circuit.

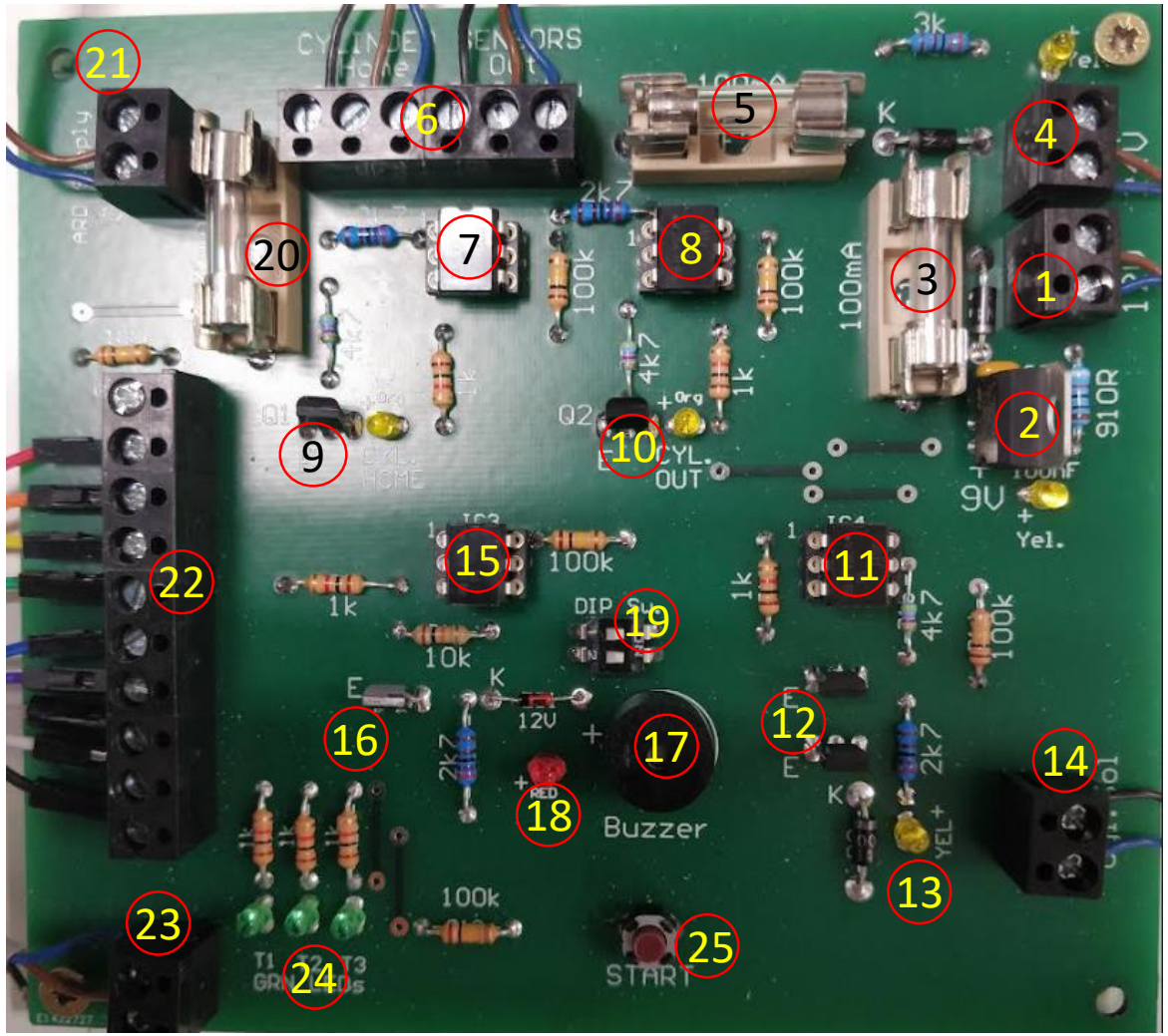


6. Cylinder sensor cables hook up to this terminal block.

7 & 8. Optocouplers provide the necessary isolation for the 24V sensors on the cylinder, to interface with the 5V input level used by the Arduino input pins.

9. The transistor, amplifies the current supplied from opto 7 to keep the current from the opto output to a low level. This ensures efficient operation of the opto. The LED next to the transistor indicates the state of the cylinder sensor.

10. Transistor - same function as above but controlled by opto 8.



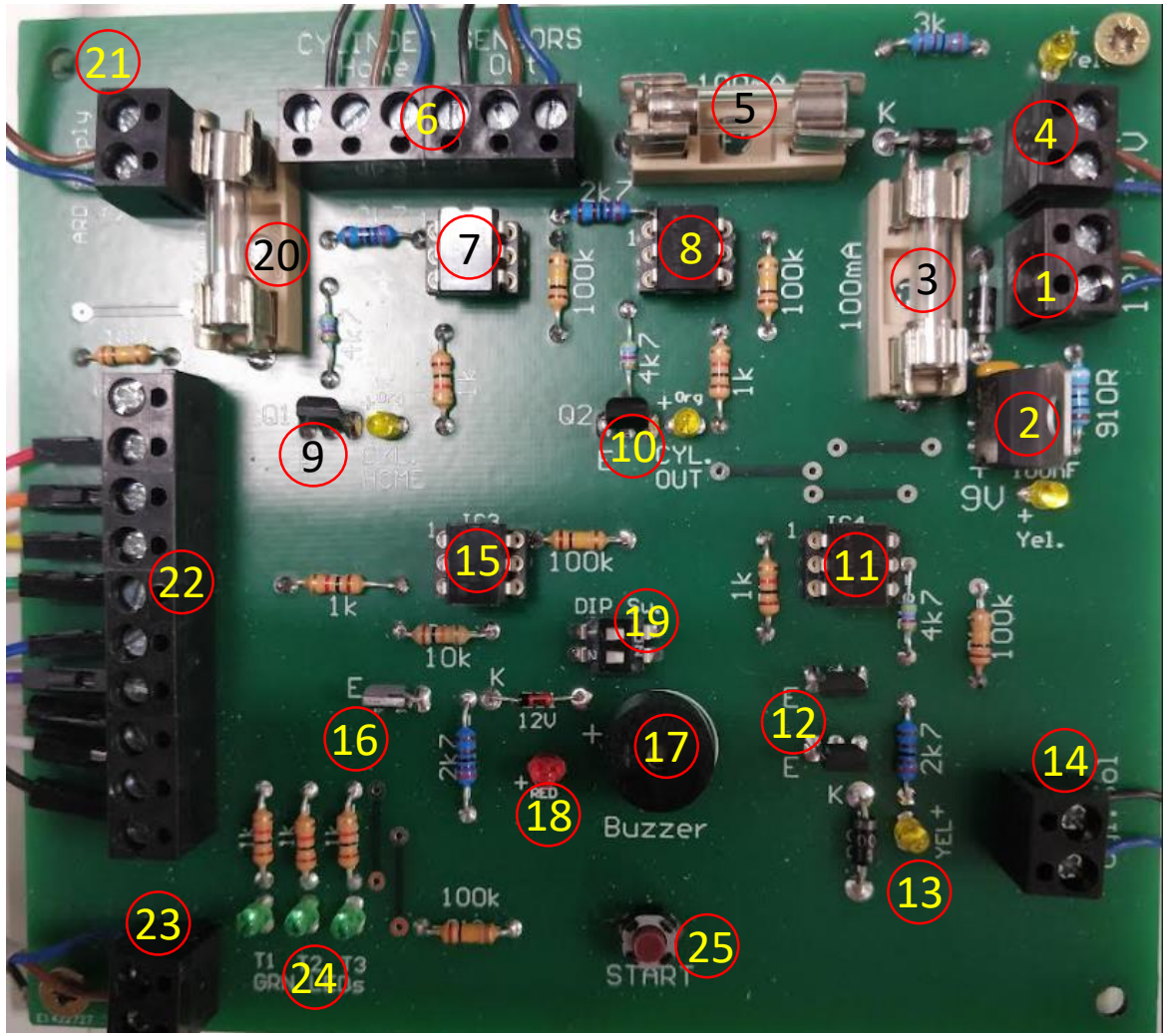
11. This opto also changes the 5V Arduino output to 24V, to power the valve solenoid.

12. These two transistors amplify the current from the opto to drive the cylinder solenoid.

13. The LED below the transistors indicates the state of this output.

14. The cylinder solenoid hooks up to this terminal block.

15. This opto provides the necessary isolation between the Arduino 5V output and the alarm buzzer (24V).



16. The transistor, amplifies the current supplied from the alarm opto (15), to drive the buzzer.

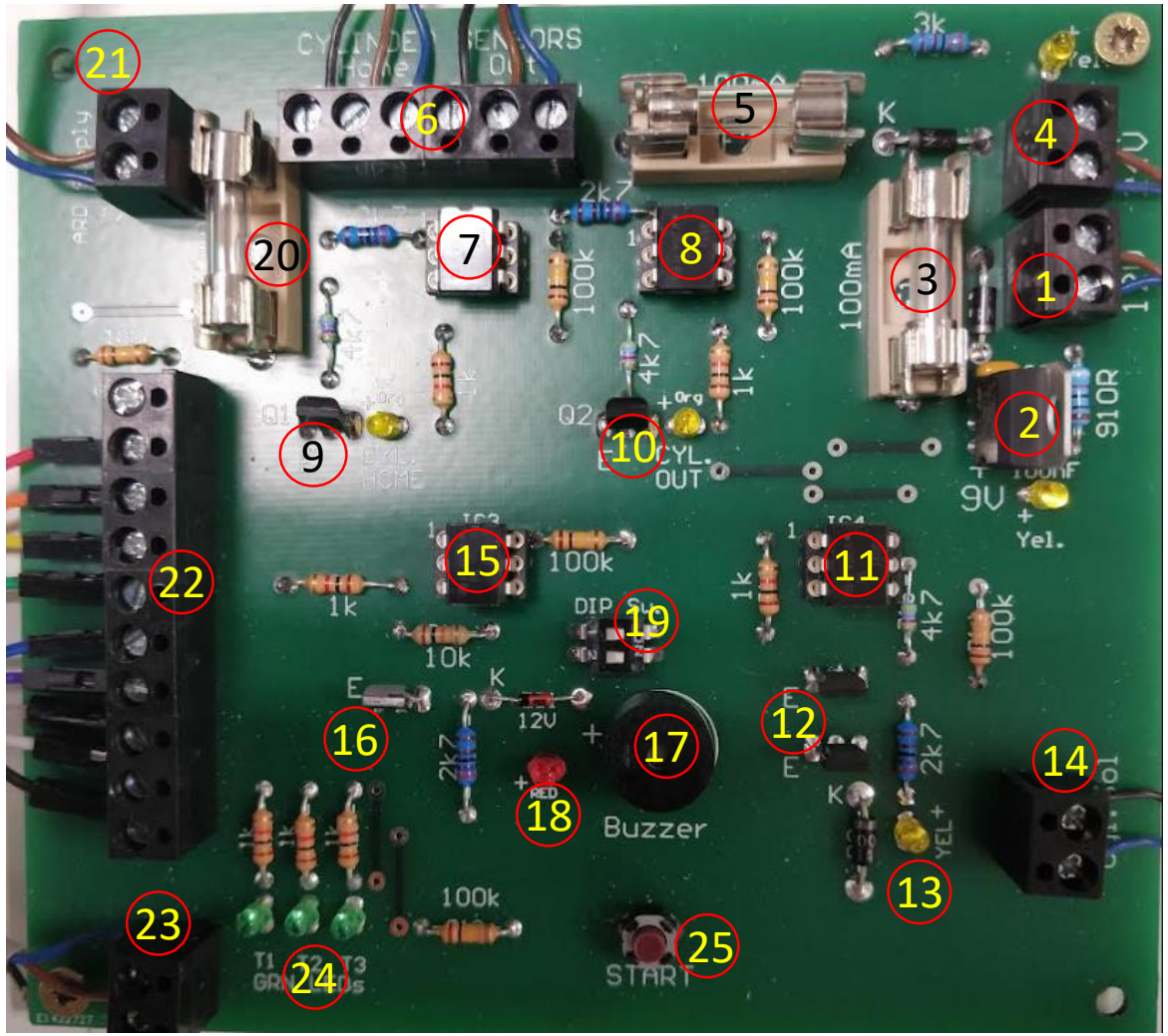
17. Alarm buzzer.

18. Alarm, Red LED.

19. DIP switch disables the buzzer, if required.

20. 50mA fuse: Arduino 9V supply protection.

21. Arduino 9V supply terminal block.



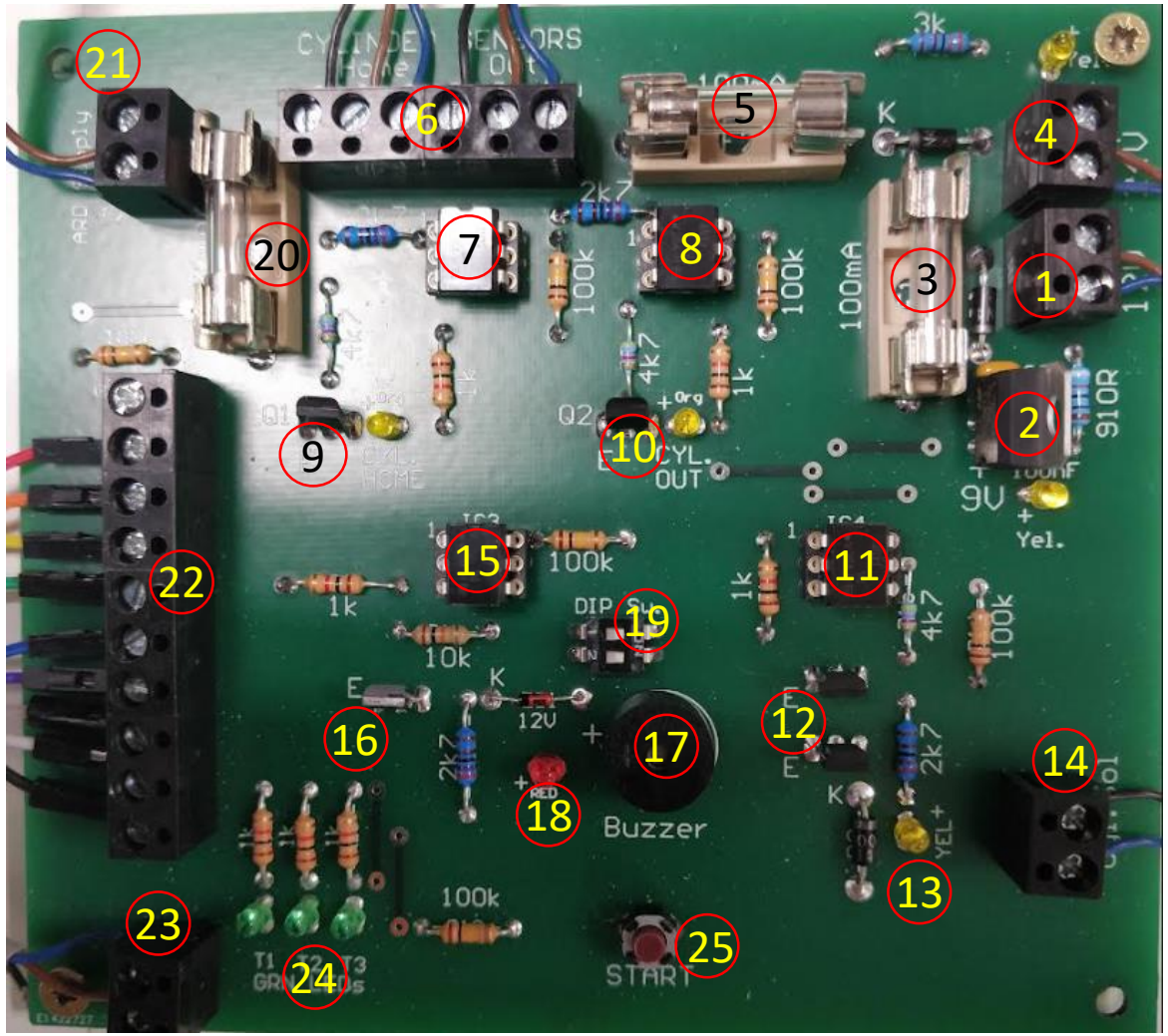
22. This terminal block connects the designated inputs and outputs of the Arduino.
Pin 1 goes to any of the ground pins on the Arduino.

Note that pins 2 to 9 on this terminal block have been purposely arranged to correspond to pins 2 to 9 on the Arduino connector.

23. The switch under test (DUT) connects to this terminal block.

24. These three green, (or transparent), LEDs indicate the successive test results of the DUT in each of the three tests: OFF - ON – OFF.
If the first test is ok, T1 lights up, etc.

25. Is the start button to initiate a test cycle.



Interface Circuit Diagram