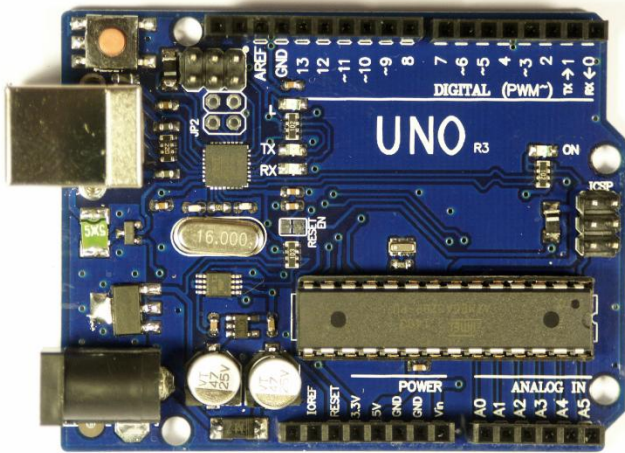


**314492**

## RFID Learning Kit (Contents)

1x 50K $\Omega$ Potentiometer
1x 7-seg LED 1x module
1x 7-seg LED 4x module
1x 8x8 dot LED array
1x 9v battery cable
1x Buzzer (active)
1x Buzzer (passive)
1x Flame sensor
1x IC 74HC595N 16-pin DIP
1x IR receiver
1x IR remote control
1x Joystick module
1x LED - RGB
1x LM35 Temp Sensor
1x Microphone sound sensor
1x Relay 5v
1x RFID card
1x RFID fob
1x RFID sensor
1x RTC module
1x Servo Motor
1x Stepper module
1x Stepper Motor
1x Temp & Humidity
1x USB cable
1x Water Level
2x Ball tilt sensor
3x Photo Resistor
4x Large button switch
5x 10K $\Omega$ resistor
5x 1K $\Omega$ resistor
5x LED - Blue
5x LED - Red
5x LED - Yellow
830-pin Breadboard
8x 220 $\Omega$ resistor
Uno R3 compatible board
Dupont connector wires
1x 4*4 button switch module
1x 2x16 LCD display

## Inland Uno R3:



### UNO R3 Summary:

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz
Length	68.6 mm
Width	53.4 mm
Weight	25 g

See <http://arduino.cc> for detailed specifications, overviews, schematics, etc. Core functions, code examples, and links to many of the device libraries can be found in the learning section; refer to the manufacturer's site if using other add-on shields or sensors.

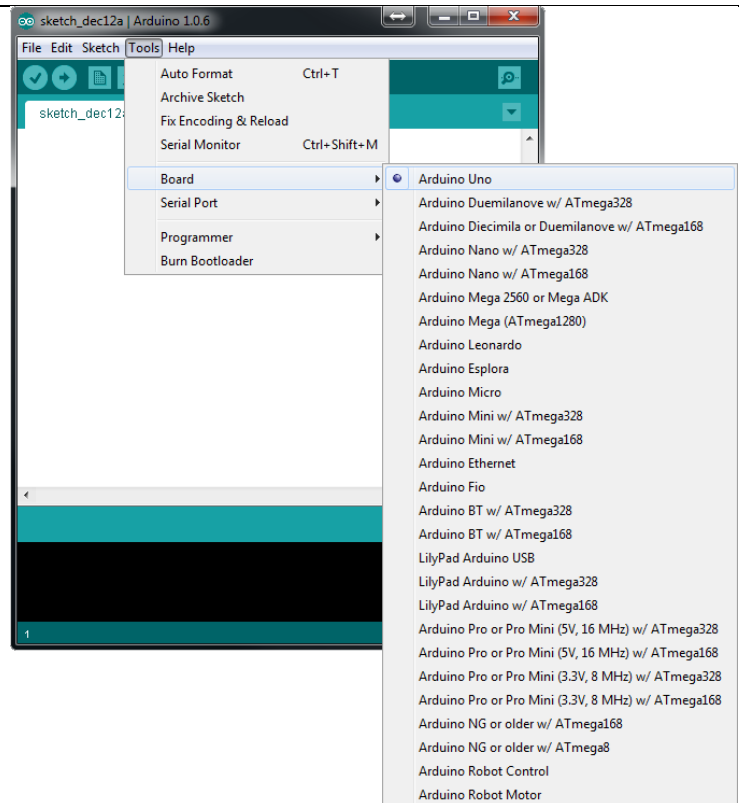
The latest Arduino Integrated Development Environment (IDE) necessary for programming your UNO R3 board can be obtained at <http://arduino.cc/en/Main/Software> (the **Download** menu choice on Arduino.cc)

Examples for many basic components can be found under the Examples menu. As you install libraries for additional shields, new examples may be available.

Follow the getting started guide found on the arduino.cc web site. Click **Learning**, and select **Getting started**. Click on the link for Windows, Mac OS X, or Linux for more specific directions.

### Getting Started:

1. Download the Arduino Environment (IDE) and install or unzip/extract the application directory.
2. Connect the UNO board to one of your computer's USB port.
3. Install the drivers (If the computer does not automatically download and install the necessary USB drivers, point the hardware setup to the "**drivers**" directory of the Arduino IDE application.)
4. Launch the Arduino IDE application
5. Open a sketch example such as "Blink"
6. Select your **Board** from the Tools menu.
7. Select the **Serial Port** used by the board
8. Upload the sketch to the board

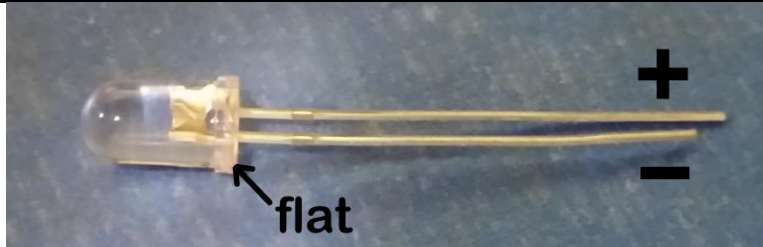


Sketch (code) Examples are included as part of the IDE. If you install device libraries for other components or shields, additional examples may be included and will show up in the list under the IDE File menu.

(See: <http://arduino.cc/en/Tutorial/HomePage> for an overview of the core functions and libraries.)

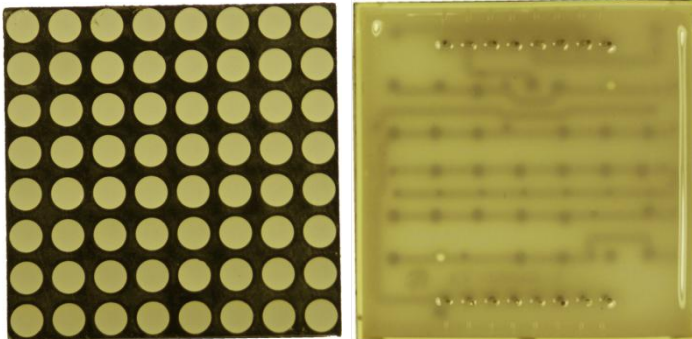
## Components:

### LEDs

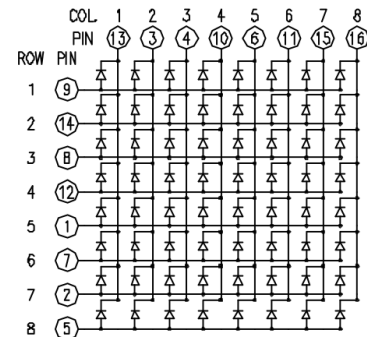


#### LED - Light Emitting Diodes

- 1) Connect a current-limiting resistor (220 ohm) between the LED's positive pin and the 5v pin. Connect the LED's negative pin directly to your Arduino output pin. -OR-
- 2) Connect a current-limiting resistor (220 ohm) between the Arduino output pin and the LED's positive pin. Connect the LED's negative pin directly to a Ground (GND) pin.

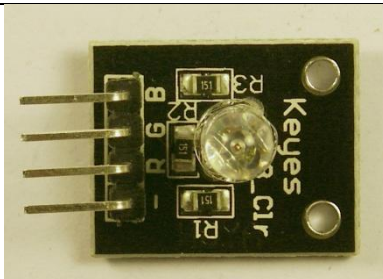


#### LED - 8x8 Matrix (1588BS or similar)



Connect Columns to Arduino Data pins that can be pulled to ground, connect columns using current limiting resistors to pins that will output positive voltage to illuminate the selected LED. See:

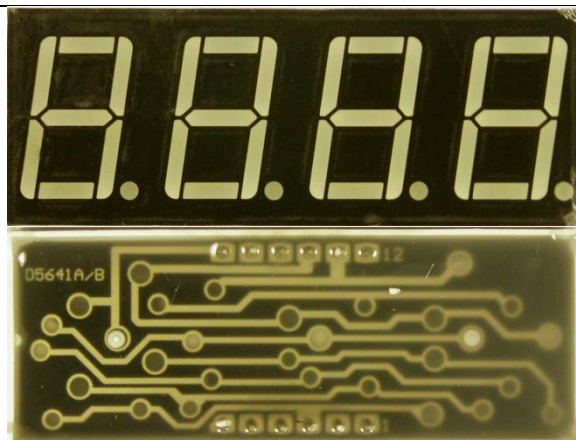
<http://arduino.cc/en/Tutorial/RowColumnScanning>



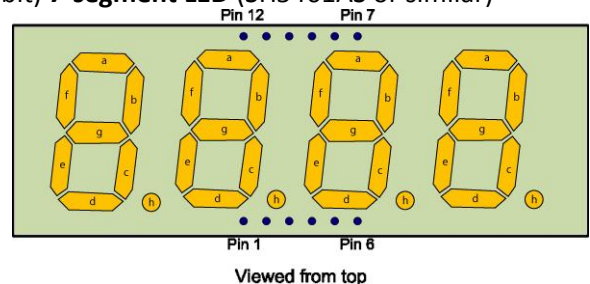
#### LED - RGB Module

Current-limiting resistors are already attached to the positive Red, Green, and Blue pins of the LED. Connect the negative (-) pin to your ground, and the R, G, and B pins to your Arduino output pins.

If using PWM (Pulse Width Modulation) capable outputs, you can effectively mix the RGB primary colors to produce thousands or different output colors in the single LED. (See Examples, 01.Basic, **Fade** sketch example in the IDE)



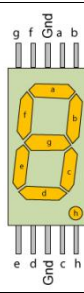
#### (4-bit) 7-segment LED (SH5461AS or similar)



Pins 12, 9, 8, 6 are grounds for each segment; LED segments share pins 11(a), 7(b), 4(c), 2(d), 1(e), 10(f), 5(g), 3(h). Transistors are recommended to handle current that could exceed the maximum output of the Arduino pins. See:

<http://learn.parallax.com/4-digit-7-segment-led-display-arduino-demo>





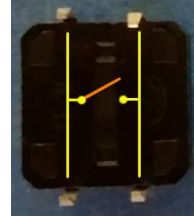
### (1-bit) 7-segment LED (TOS5121AS or similar)

Pin 1 is bottom left. Pins 3 and 8 are a common ground. Connect other pins to your Arduino with a current limiting resistor.

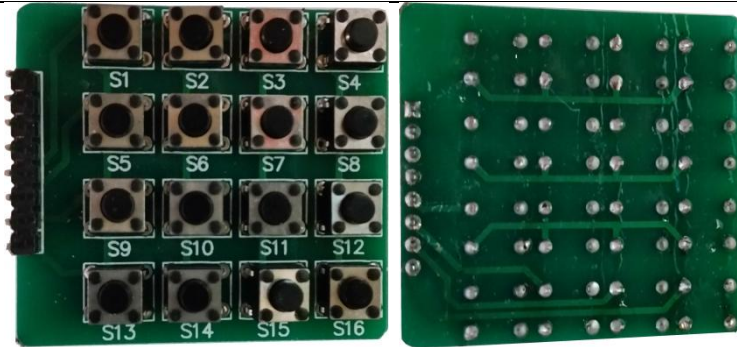
## Switches



### Large button switch - momentary contact, NO



For the switch, you can use either pair located on one side. The connection is Normally Open (off) until the button is pushed.



### 4x4 button matrix keypad

Pin 1 is indicated by the square solder pad on the rear (closest to S13.)

Pins 1-4 connect to rows of buttons:

Pin 1 - S13, 14, 15, 16; Pin 2 - S9, 10, 11, 12; Pin 3 - S5, 6, 7, 8; Pin 4 - S1, 2, 3, 4

Pins 5-8 connect to columns of buttons:

Pin 5 - S1, 5, 9, 13; Pin 6 - S2, 6, 10, 14; Pin 7 - S3, 7, 11, 15; Pin 8 - S4, 8, 12, 16

For an example, see:

<http://arduinoexperiments.blogspot.com/2013/06/arduino-calculator.html>

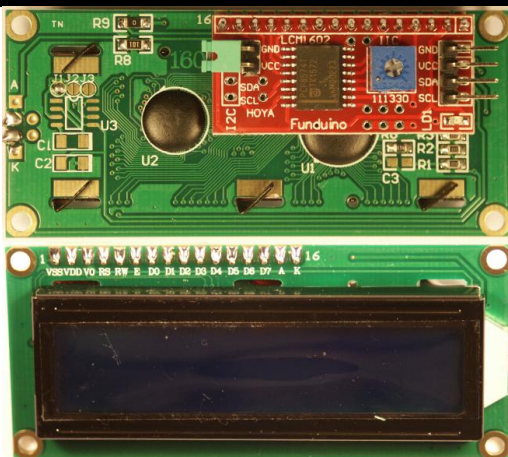


### 5 Volt Relay module

Three input pins: +, -, S ...connect - to ground, connect + to 5V, Connect S to your Arduino "signal" pin to trip the relay.

Three output (screw) pins: Center is common, NC indicates Normally Closed (ON), NO indicates Normally Open (OFF). When relay engages, the NC contact will open, the NO contact will close.

## LCD



### I2C 1602 LCD - 2-line, 16-character LCD display (I2C) with backlight.

4 pin connections are required: 5V (Vcc), Ground, and two Analog lines (i.e. SDA-A4, SCL-A5). For Arduino, you will need several libraries installed: Wire.h, LCD.h, LiquidCrystal\_I2C.h

See:

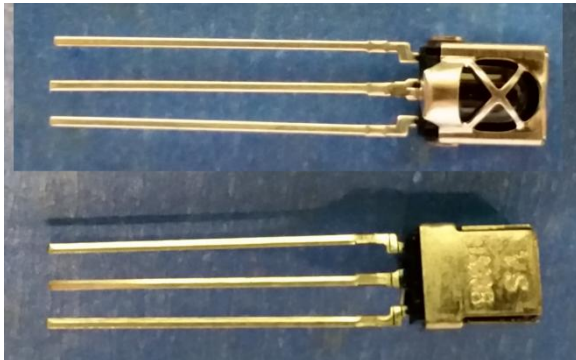
<http://www.hessmer.org/blog/2014/01/11/arduino-compatible-iic-i2c-serial-2-5-lcd-1602-display-module/>

## Sensors and modules



### Flame Sensor (YG1006 or similar)

The Flame sensor is a high-speed and highly sensitive NPN Silicon photo transistor based on the YG1006. It can be used to detect fire or other wavelength at 760nm ~ 1100nm light. Response time is 15us, supply voltage is 3.3-5V; output is analog.



Out Gnd Vcc

### IR Receiver (VS1838B or similar)

Connect the Vcc pin to your 5V pin and the Gnd to a Ground pin. The Out pin connects to an Arduino input pin and will change when the sensor detects an Infrared signal.

The IR remote control will send coded pulses based on which button you press, or an IR LED will produce a continuous illumination.



### Passive & Active buzzers

Use as a speaker, buzzer or other audible indicator. The Active buzzer has a protective tag over the opening, note the + identifies the positive pin of the device, as the rear is covered with epoxy. The passive buzzer does not have epoxy on the rear PCB, and the positive and negative connections are visible on the etched board.



### Ball Tilt Sensor

This is a very simple switch with a ball inside of the tube. When the sensor is tipped upward past the horizontal, the ball will short the contacts, closing the switch. With the top (away from the pins) is tilted down relative to



BOTTOM VIEW  
LM35DZ

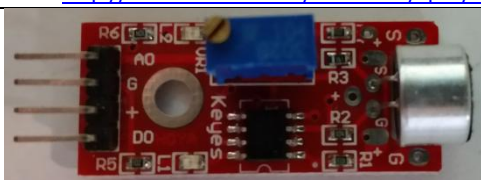
### LM35DZ Temperature Sensor (or similar)

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature.

Basic Temperature Sensor (+2° to +150°C):

+Vs=5V in, Ground, Vout = 0mV + 10.0mV/°C

See data sheet: [http://www.ece.usu.edu/ece\\_store/spec/lm35dt-3p.pdf](http://www.ece.usu.edu/ece_store/spec/lm35dt-3p.pdf)



### Microphone Sound Sensor

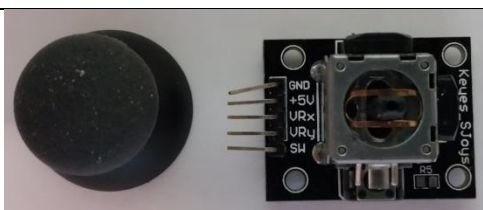

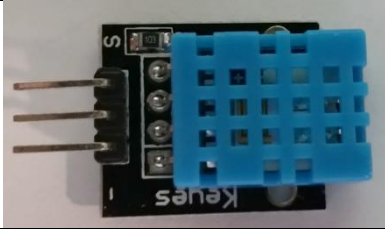
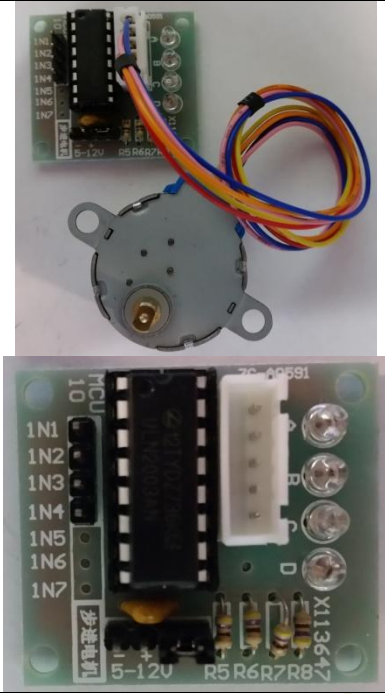

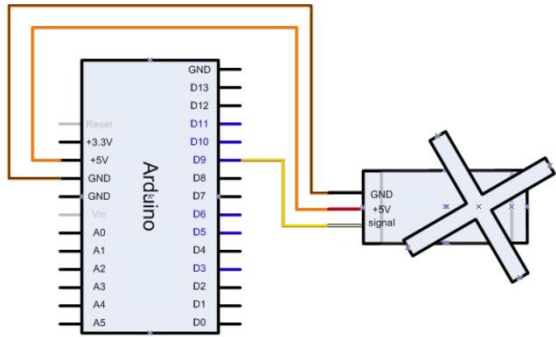
Four connections:

G - connect to Ground

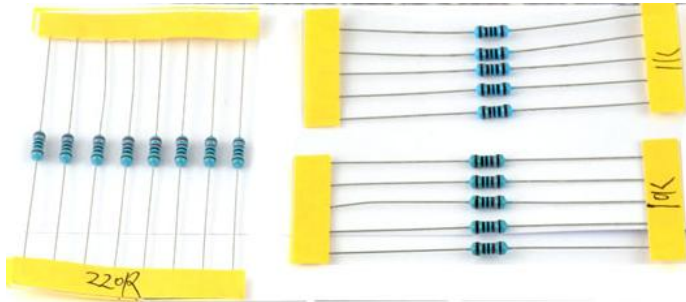
+ - connect to 5V

AO - Analog Out - connect to Arduino for analog input  
DO - Digital Out - Connect to Arduino as digital trigger input; adjust sensitivity via screw on potentiometer.



	<p><b>Joystick with push button module</b></p> <p>Five connections: GND (Ground), +5V, VRx = x-axis analog output, VRy = y-axis analog output, SW = Normally Open switch (push down on joystick to activate)</p>
	<p><b>Water Level Sensor</b></p> <p>Three connections - = Ground, + = 5V, S = analog signal that will vary based on how much of the contacts are under water.</p>
	<p><b>Temperature and Humidity sensor</b></p> <p>Three connections: pin 1 = ground (-), pin2 = +5V, pin 3 = Signal (digital, serial output)</p> <p>For DHT11 library and information, see: <a href="http://playground.arduino.cc/main/DHT11Lib">http://playground.arduino.cc/main/DHT11Lib</a></p>
<b>Motors</b>	
	<p><b>Stepper motor and controller</b></p> <p>Stepper IC = ULN2003AN (or similar)</p> <p>The stepper motor included in the kit connects to the controller through a white connector socket. Four inputs connect to your Arduino IN1, IN2, IN3, and IN4. Power for the motor and controller is provided through the ground (-) and 5-to-12V (+) pins.</p> <p>Use an external power supply for the motor to avoid damaging the Arduino. Connect the ground of your external supply to the ground of the Arduino and the signal IN# pins to digital outputs.</p> <p>For sketch examples, see: <a href="http://arduino.cc/en/Tutorial/MotorKnob">http://arduino.cc/en/Tutorial/MotorKnob</a></p>
 <p>Servo examples are include with the IDE, see: <a href="http://arduino.cc/en/Tutorial/Knob">http://arduino.cc/en/Tutorial/Knob</a> <a href="http://arduino.cc/en/Tutorial/Sweep">http://arduino.cc/en/Tutorial/Sweep</a></p>	<p><b>Servo motor</b></p> <p>Note that Servo motor color schemes may vary: Brown / Black = ground Orange / Red = +5V Yellow / White = signal (use digital PWM connection.)</p> 

## Resistors



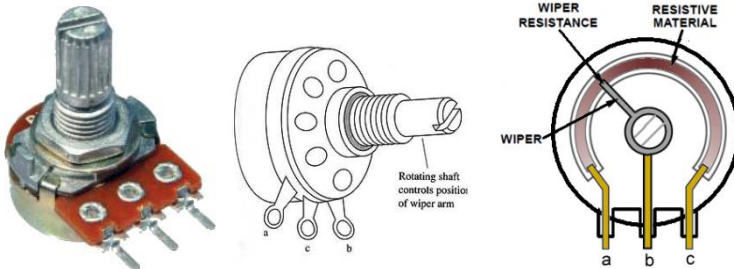
8pcs 220 ohm  
5pcs 1K ohm  
5pcs 10K ohm

## 1/8 Watt Resistors



0 - Black		tolerance
1 - Brown		none = 20%
2 - Red		silver = 10%
3 - Orange		gold = 5%
4 - Yellow		
5 - Green		
6 - Blue		
7 - Purple		
8 - Grey		
9 - White		

red, yellow, brown =  
2, 4, (1 x 0) =  
240 ohms



## 50K Potentiometer

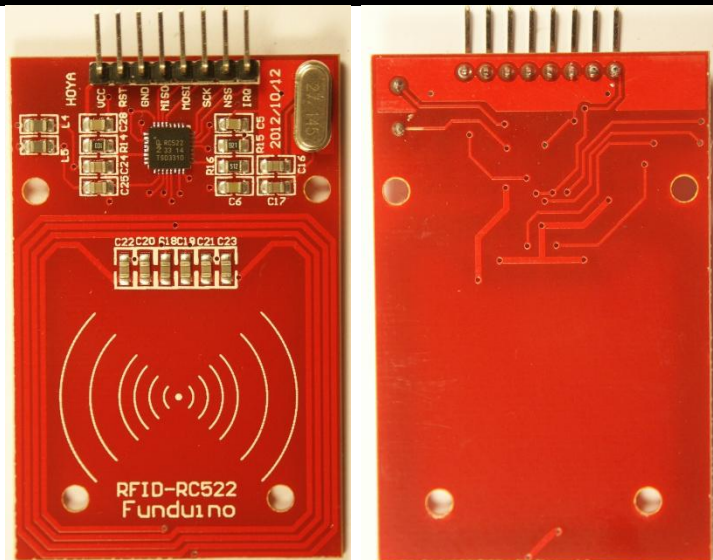
Resistance between outer pins is 50K ohms. Resistance between one outer pin and the center (wiper) pin is 0-50K ohms based on position.



## Photo Resistor

Resistance across the pins will be 1 meg ohm or higher in darkness, dropping to 60 ohms or less in bright light.

## RFID



## RFID-RC522 Read/Write module

Typical connections:

Vcc - 5V or 3.3V

RST - Arduino pin 5

GND - Ground

MISO - Arduino pin 12

MOSI - Arduino pin 11

SCK - Arduino pin 13

(NSS, IRQ are not connected) See:

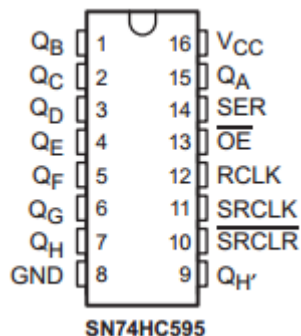
<http://playground.arduino.cc/Learning/MFRC522>

[https://labitat.dk/wiki/RFID\\_RC522-AN](https://labitat.dk/wiki/RFID_RC522-AN)

<https://sites.google.com/site/arduinomega2560/projects/home/lev-el-1/arduino-rfid-rc522>

<http://www.grantgibson.co.uk/2012/04/how-to-get-started-with-the-mifare-mf522-an-and-arduino/>

**other**

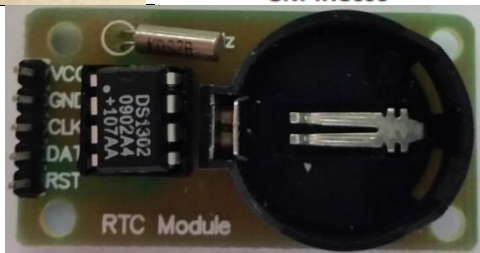


**Tri-state 8-bit shift register IC SN74HC595N (or similar)**

The datasheet refers to the 74HC595 as an "8-bit serial-in, serial or parallel-out shift register with output latches; 3-state." In other words, you can use it to control 8 outputs at a time while only taking up a few pins on your microcontroller.

See:

<http://arduino.cc/en/tutorial/ShiftOut>



## Real Time Clock Module

The DS1302 trickle-charge timekeeping chip contains a real-time clock/calendar and 31 bytes of static RAM. It communicates with a microprocessor via a simple serial interface. The real-time clock/calendar provides seconds, minutes, hours, day, date, month, and year information.

For library and code example, see:

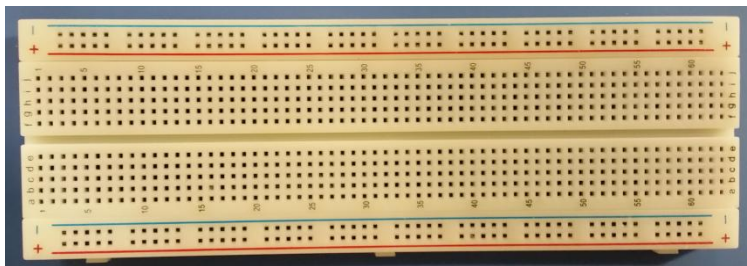
<http://playground.arduino.cc/Main/DS1302RTC>



### IR Transmitter (Remote Control)

Multiple tutorials are available for different remotes, use serial monitoring to identify inputs from your remote control and use those values in your programs. For a typical setup, see:

<http://arduino-info.wikispaces.com/IR-RemoteControl>



## 830-pin Breadboard

Power rails run the length of each side and are color coded blue for negative and red for positive. Inside rows of 5 pins each are connected together, but not to each other, and not to the power rails.

