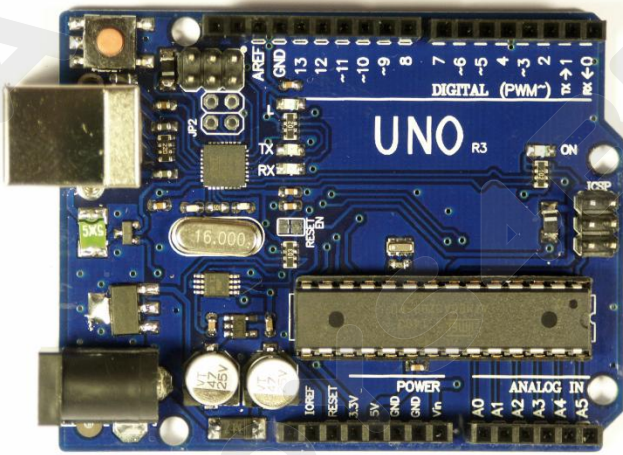




RFID Learning Kit (Contents)

- 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

**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.

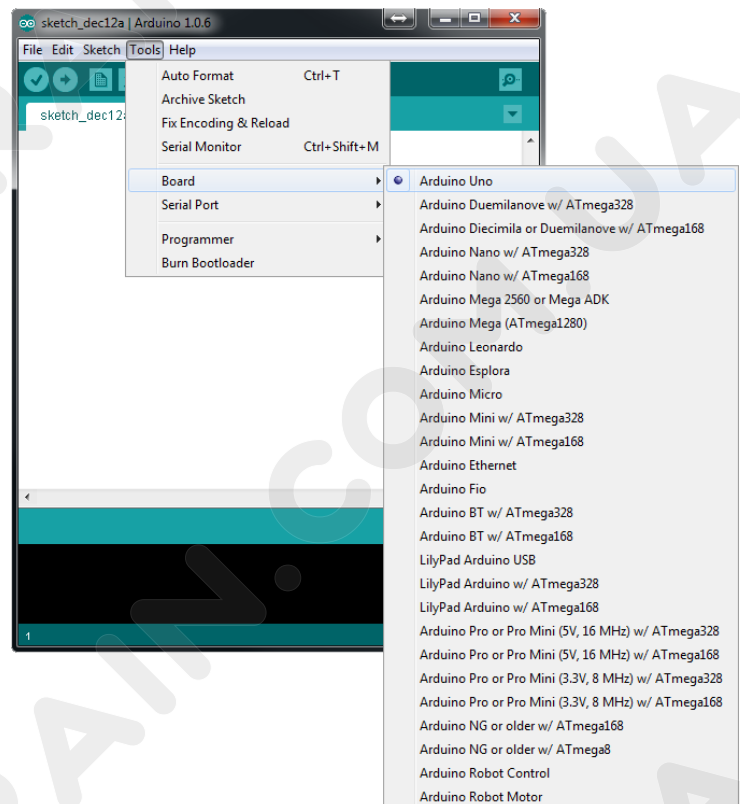
The latest Arduino Integrated Development Environment (IDE) necessary for programming your UNO R3 board can be obtained at <http://arduino.cc/en/Main/Software> (the **Download** menu choice on Arduino.cc)

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.)

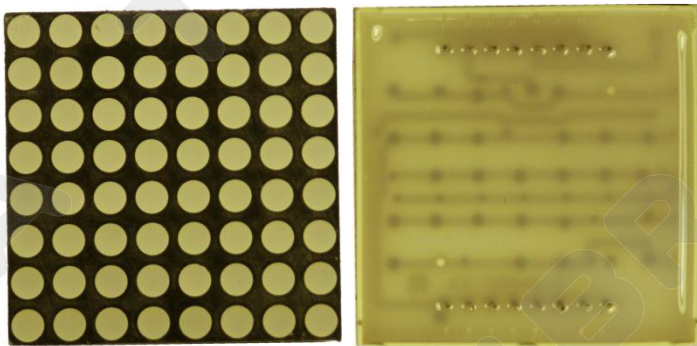
LEDs



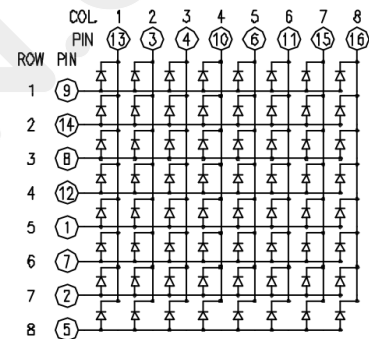
LED - Light Emitting Diodes

- 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.

Note: LEDs may have "water clear" or color tinted lens.

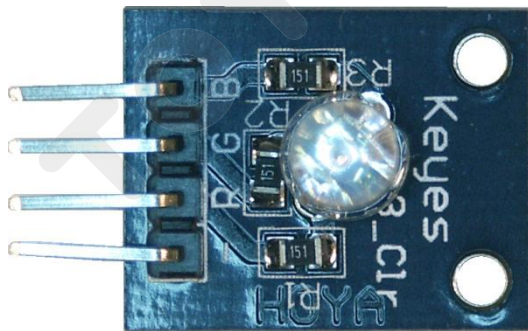


LED - 8x8 Matrix (1588BS or similar)



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>



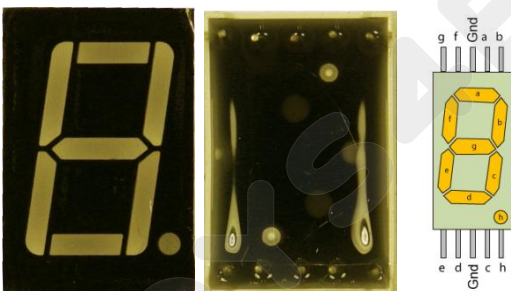
LED - RGB Module

Current-limiting resistors (151 = 1.5 \* 10 = 150 ohm) 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:

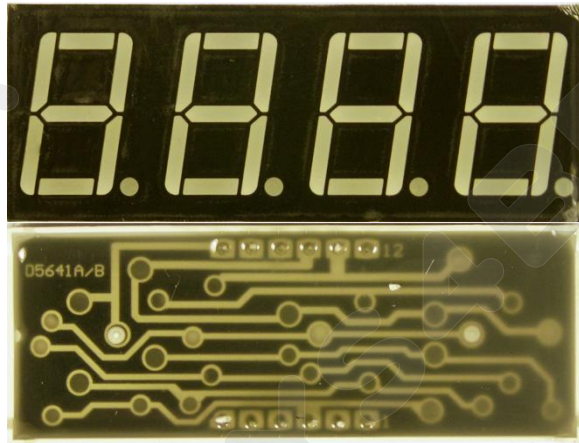
File, Examples, 01.Basic, **Fade** sketch example in the IDE

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

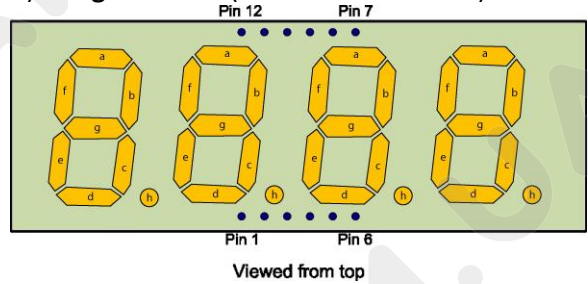


(1-bit) 7-segment LED (TOS5121AS or similar)

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



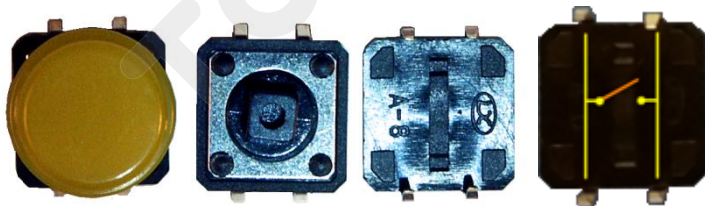
**(4-bit) 7-segment LED (SH5461AS or similar)**



Viewed from top  
 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:

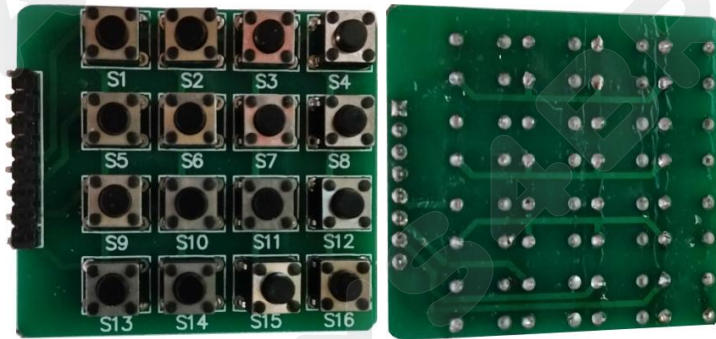
<http://learn.parallax.com/4-digit-7-segment-led-display-arduino-demo>

**Switches**



**Large button switch - momentary contact, NO**

For the switch connection, 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:

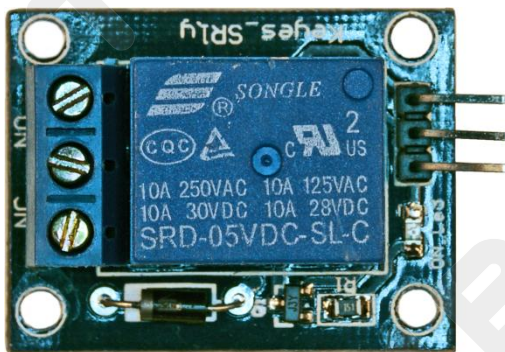
Pin 1 - S13, 14, 15, 16; Pin 2 - S9, 10, 11, 12; Pin 3 - S5, 6, 7, 8; Pin 4 - S1, 2, 3, 4

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:

(+) +5V

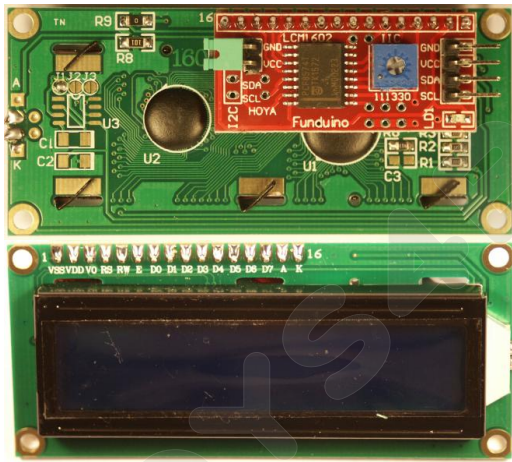
(-) Ground

(S) Signal - connect to your Arduino "signal" pin to trip the relay.

To use an external 5V source to power the relay coil or servo motors, connect the ground of the external source to the Arduino ground.

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.



**IR Receiver** (VS1838B or similar)

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

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

The surface-mount LED connects to the signal pin of the receiver. It will illuminate when the sensor detects an active infrared source.



**Passive & Active buzzers**

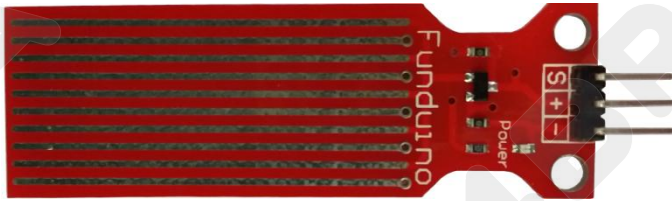
Use as a speaker, buzzer or other audible indicator.

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 active buzzer will generate a tone as soon as power is supplied to the device.

The passive buzzer does not have epoxy on the rear PCB, and the positive and negative connections are visible on the etched board. Passive buzzers must have a modulated signal supplied to the device (like a speaker) and would only generate a "click" if DC voltage is applied.

Play a tone or melody using the passive buzzer:  
<http://osepp.com/learning-centre/start-here/101-basic-starter-kit/tutorial-6-using-buzzer-to-play-a-melody/>  
<http://www.arduino.cc/en/Tutorial/melody>





### Water Level or Rain Sensor

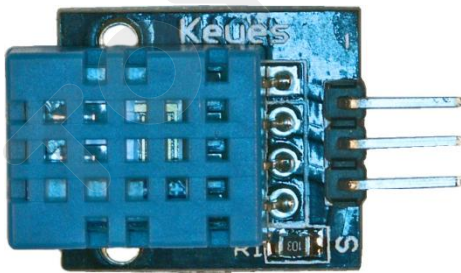
Three connections - = Ground, + = 5V, S = analog signal that will vary based on how much of the contacts are in contact with water.

See:

<http://www.instructables.com/id/Water-Level-Sensor-Module-for-Arduino-AVR-ARM-STM3/>

[http://www.seeedstudio.com/wiki/Grove\\_-\\_Water\\_Sensor](http://www.seeedstudio.com/wiki/Grove_-_Water_Sensor)

(Video): <http://www.wearerobots.co.uk/funduino-water-level-sensor-tutorial-and-application/>



### Temperature and Humidity sensor

Three connections:

(-) = ground (-) Note the square solder pad.

(center pin) = +5V

(S)= Signal (digital, serial output)

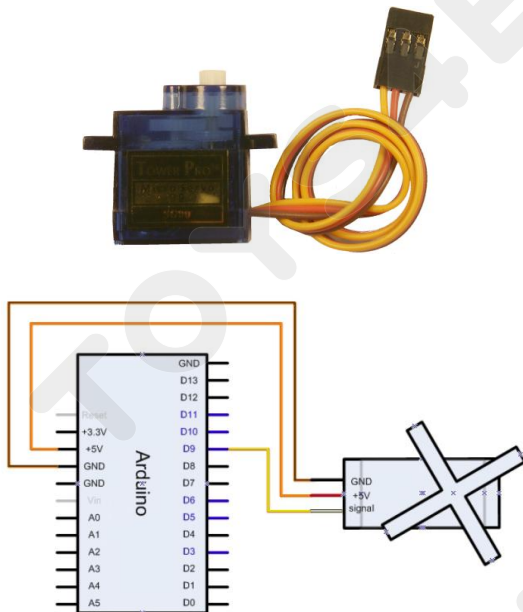
For the DHT11 library and information, see:

<http://playground.arduino.cc/main/DHT11Lib>

<http://www.johnboucha.com/arduino-dht11-temperature-humidity/>

[https://tkkrlab.nl/wiki/Arduino\\_KY-015\\_Temperature\\_and\\_humidity\\_sensor\\_module](https://tkkrlab.nl/wiki/Arduino_KY-015_Temperature_and_humidity_sensor_module)

## Motors



### Servo motor

Note that Servo motor color schemes may vary:

Brown / Black = ground

Orange / Red = +5V

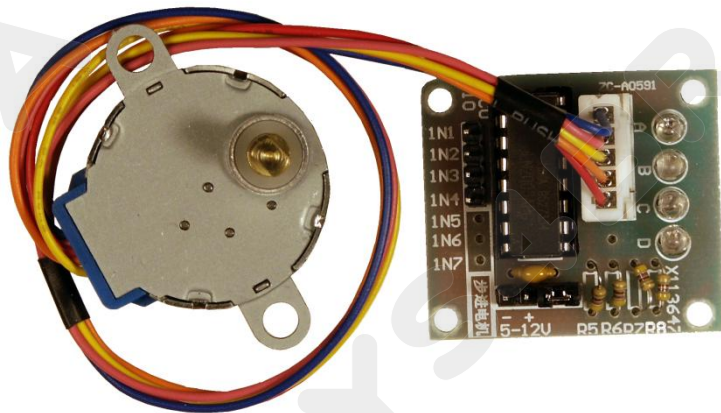
Yellow / White = signal (use digital PWM connection.)

Small servo motors may be connected directly to the Arduino controller, or you may wish to provide separate power to the servos and only connect the data and a common ground.

Servo examples are include with the IDE, see:

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

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



### Stepper motor and controller

Stepper IC = ULN2003AN (or similar)

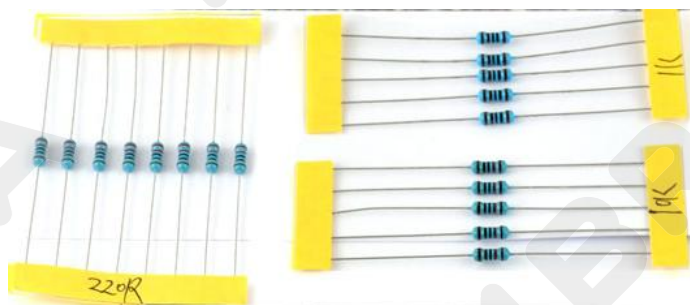
The stepper motor included in the kit connects to the controller through a white connector socket. Four inputs connect to your Arduino IN1, IN2, IN3, and IN4. Power for the motor and controller is provided through the ground (-) and 5-to-12V (+) pins.

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.

For sketch examples, see:

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

## Resistors

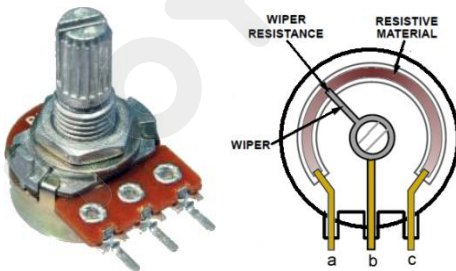


8pcs 220 ohm  
5pcs 1K ohm  
5pcs 10K ohm

### 1/4 Watt Resistors

Color	1st Band	2nd Band	3rd Band	Multiplier	Tolerance
Black	0	0	0	1Ω	
Brown	1	1	1	10Ω	± 1% (F)
Red	2	2	2	100Ω	± 2% (G)
Orange	3	3	3	1KΩ	
Yellow	4	4	4	10KΩ	
Green	5	5	5	100KΩ	± 0.5% (D)
Blue	6	6	6	1MΩ	± 0.25% (C)
Violet	7	7	7	10MΩ	± 0.10% (B)
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1	± 5% (J)
Silver				0.01	± 10% (K)

Resistors may come with 4 or 5 identifying color bands. (When in doubt, use a multimeter to verify the value.)



### 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.

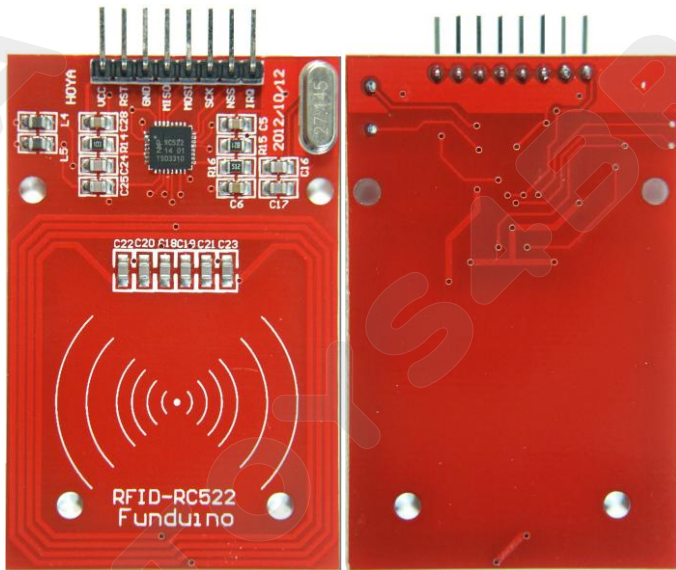


### Photo Resistor

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:

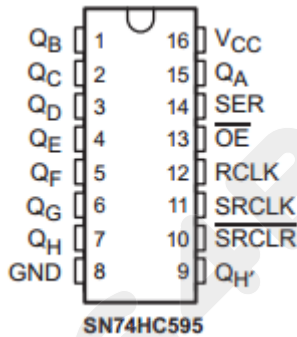
<http://playground.arduino.cc/Learning/MFRC522>

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

<https://sites.google.com/site/arduinomega2560projects/home/level-1/arduino-rfid-rc522>

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

other

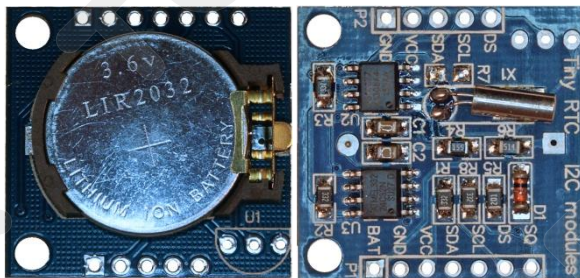


Tri-state 8-bit shift register IC SN74HC595N (or similar)

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 (for DS1302, see below...)

The Tiny RTC communicates with a microprocessor via the I2C serial interface. The real-time clock/calendar provides seconds, minutes, hours, day, date, month, and year information. Pin connections:

BAT: Battery voltage monitor (not used)

GND: Ground

VCC: +5V

SDA: I2C data (connect to A4)

SCL: I2C clock (connect to A5)

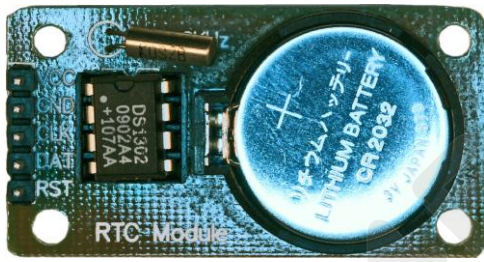
DS: DS18B20 Temp. Sensor output (connect to D2)

SQ: Square wave output (not used)

For library and code example, see:

[http://www.hobbyist.co.nz/?q=real\\_time\\_clock](http://www.hobbyist.co.nz/?q=real_time_clock)

<http://playground.arduino.cc/code/time>



**Real Time Clock Module** (provided in some kits)  
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:

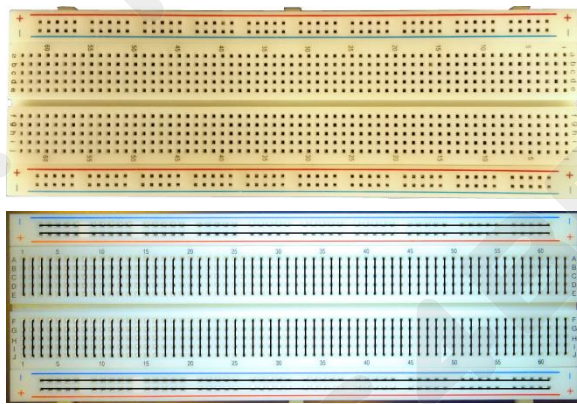
<http://playground.arduino.cc/Main/DS1302RTC>



**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.

## Additional Resources:

Several sites have hook-up and information and code examples on a variety of sensors, similar to, and including the ones found in this kit. Some sensors may be loose components or integrated into different board designs. If the documented sensor uses the same electronic component, then any code sketch documented may work with the sensors found in your kit. However, depending on the circuit design, the adjustments or sensitivity range may need to be modified slightly to achieve the desired result. Sites documenting these and other sensors include:

Arduino Playground Examples and additional libraries (code sketches available from the IDE *File, Examples* menu):

<http://www.arduino.cc/en/Tutorial/HomePage>

Arduino Playground Tutorials: <http://playground.arduino.cc/Learning/Tutorials>

Forum.HobbyComponents.com: <http://forum.hobbycomponents.com/viewtopic.php?f=73&t=1320>

LinkSprite Wiki - Advanced Sensors Kit for Arduino: [http://linksprite.com/wiki/index.php5?title=Advanced\\_Sensors\\_Kit\\_for\\_Arduino](http://linksprite.com/wiki/index.php5?title=Advanced_Sensors_Kit_for_Arduino)

TkkrLab.nl (Tukkerlab)Wiki: [https://tkkrlab.nl/wiki/Arduino\\_37\\_sensors](https://tkkrlab.nl/wiki/Arduino_37_sensors)

University of Rhode Island (PDF coursework): <http://www.ele.uri.edu/courses/ele205/Arduino%20-%20Learning.pdf>

Freeduino.org: <http://www.freeduino.org/>

Arduino for Projects (PDF with 1193 projects): <http://duino4projects.com/arduino-projects-pdf/>

Lady Ada - Introduction to Arduino- step-by-step lessons: <http://www.ladyada.net/learn/arduino/index.html>

Tronixstuf Arduino Tutorials: <http://tronixstuff.com/tutorials/>

Earthshine Electronics Beginners Guide to Arduino:

[https://docs.google.com/file/d/0Bw\\_ruMOtRDDgNXI3OTFGZXhI2c/edit?usp=sharing](https://docs.google.com/file/d/0Bw_ruMOtRDDgNXI3OTFGZXhI2c/edit?usp=sharing)

Sheepdog's Guide to Arduino Programming: <http://sheepdogguides.com/arduino/FA1main.htm>