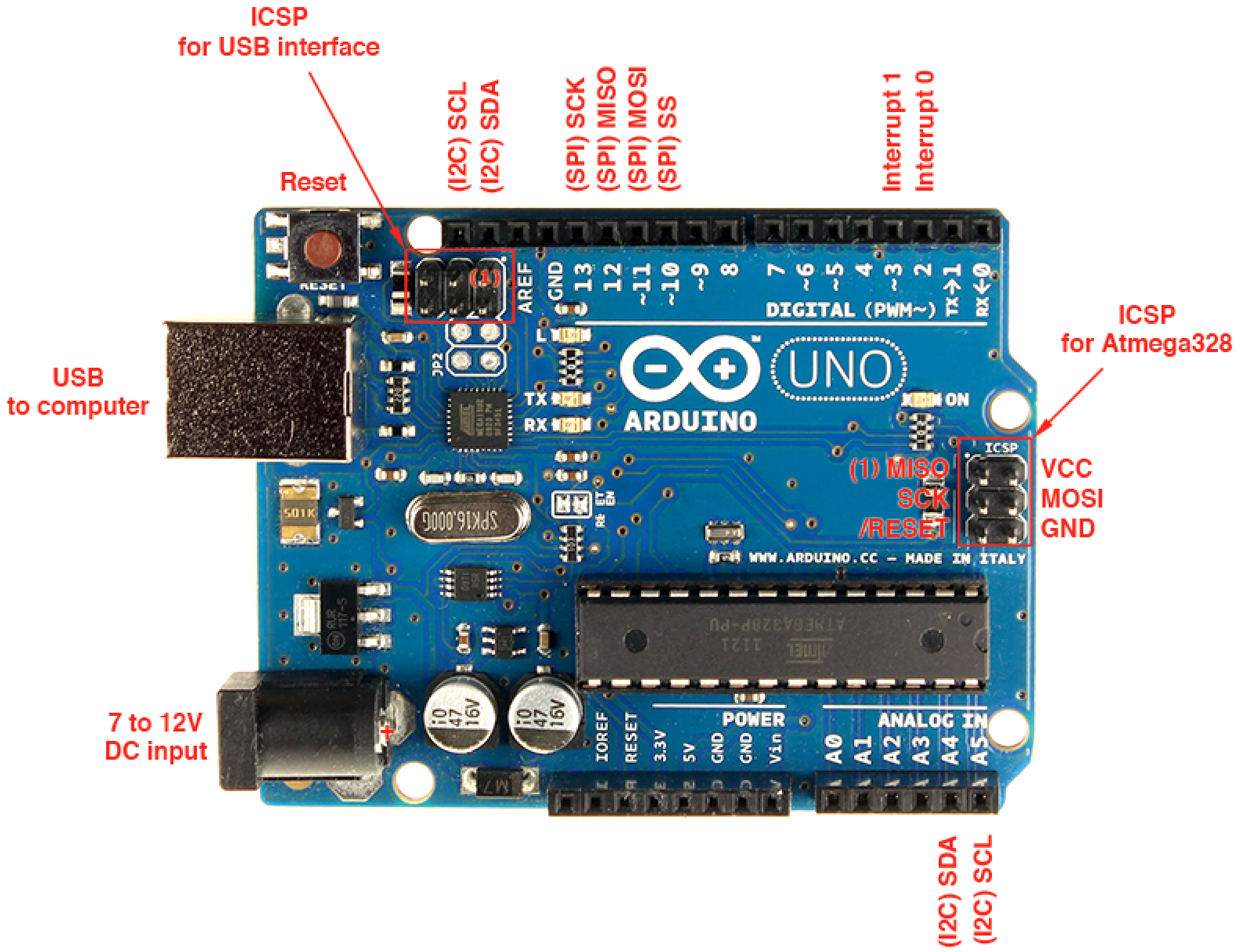
ICSP for USB interface
Reset
(I2C) SCL
(I2C) SDA
(SPI) SCK
(SPI) MISO
(SPI) MOSI
(SPI) SS
Interrupt 1
Interrupt 0
ICSP for Atmega328
USB to computer
(1) MISO
SCK
/RESET
VCC
MOSI
GND
7 to 12V DC input
(I2C) SDA
(I2C) SCL

SPI

SPI

Serial Peripheral Interface.

http://gammon.com.au/spi

A serial protocol that uses a clock line (SCK) along with master-to-slave data (MOSI) and slave-to-master data (MISO) to transfer information. Also commonly used is a slave select (SS) line to select one slave out of multiple ones.

SCK

Serial Clock (used by SPI).

A clock line generated by the master to "clock" data to the slave.

MOSI

Master Out, Slave In (used by SPI).

The data going from the master to the slave.

MISO

Master In, Slave Out (used by SPI).

The data going from the slave to the master.

SS

Slave Select (used by SPI).

Used to select a slave. Typically is brought low to be active.

I2C

I2C

http://gammon.com.au/i2c

A serial protocol that uses a clock line (SCL) along with a data line (SDA) to transfer information.

SCL

Serial Clock (I2C)

The clock line used by I2C to indicate data is ready on the data line.

SDA

Serial Data (I2C)

The data line (bidirectional) used by I2C.

Programming

ICSP

In Circuit Serial Programming.

Used for programming either the main Atmega328 processor (most common) or the Atmega16U2 USB interface processor (only if required).

The ICSP system uses SPI with a protocol built into the chips.

Note: programming is typically also done by using the async serial interface (pins D0/D1) and an installed bootloader program.

Power

VCC

The "power" line. Generally +5V on the Uno.

IOREF

The IO voltage reference (connected to +5V). Intended to allow shields to see if the board is running at 5V or 3.3V.

Vin

The "voltage in" to the board. This will generally be about 1V less than the voltage supplied to the "power plug" due to the reverse polarity protection diode.

It is not recommended to draw power from Vin if you are powering the Arduino from the USB port.

GND

The "ground" line.

Other abbreviations

USB

Universal Serial Bus.

Used to communicate via the USB protocol with a host computer (for programming or sending/receiving serial data).

PWM

Pulse Width Modulation.

Some pins, marked with a "~" symbol on the board, support PWM, that is rapid pulses that can be used to control motors at various speeds, or play music (by varying the pulse frequency).

Analog reading

AREF

Analog read reference.

You can put some voltage source here (0 to 5V) as an accurate "voltage reference" for the analogRead function calls. By default, this is not required.

Analog pins

Pins A0 to A5 can be used to do "analog" reads, giving a reading between 0 (for 0V) and 1023 (for 5V) using the analogRead function. The tested range can be altered by suitable programming and/or supplying a different voltage to the AREF pin.

The analog pins can also be used for digital reading/writing. For this purpose you can refer to them as pins 14 to 19. (eg. `digitalWrite (19, HIGH);`)

Async serial

Pins D0 (marked Rx) and D1 (marked Tx) can be used for async serial comms.

http://www.gammon.com.au/forum/?id=10894

The Rx pin is "inwards", that is from host computer to Arduino, and the Tx pin is "outwards", that is from the Arduino to another device.

These pins are also internally connected via 1K resistors to the USB chip, so that data from the USB interface can be sent/received to pins 0 and 1.

Maximum ratings

The IO pins have an absolute maximum rating of 40 mA per pin. However in practice you should design for a rating of 20 mA to be on the safe side.

Also the following **groups** of pins should not **source** than 150 mA (each group):

- Digital pins 0 to 4 plus analog pins A0 to A5
- Digital pins 5 to 13

Also also the following **groups** of pins should not **sink** than 100 mA (each group):

- Digital pins 0 to 4
- Digital pins 5 to 13
- Analog pins A0 to A5

In addition to that, the **entire processor chip** has a maximum rating of 200 mA current consumption.

These figures generally refer to pins in output mode (input mode won't consume much power). So you can see that, configured as outputs, you could only have 10 pins sourcing 20 mA each (to avoid exceeding the chip maximum) and even then they would need to be spread out so that you don't consume more than 100 mA from a group mentioned above.

Current limiting resistors

As a rule of thumb, if you hook up an LED to an output pin, you would want something like a 330 ohm resistor in series with the diode. This is because you want something like 10 mA current drawn, with a 3V voltage drop (the diode drops about 2V, so the resistor drops the other 3V from a supplied 5V). Using Ohms Law, 3/0.010 = 300 ohms. The closest standard value then is 330 ohms, although a bit higher wouldn't hurt.

Even a 1K resistor gives a reasonably bright LED output, and only consumes around 3 mA.

Power supply

The board can be powered from the USB connector for testing while connected to the computer.

If you use an external supply a voltage range of 7V to 12V is recommended (e.g., a 9V battery could be used). Bear in mind the higher the voltage the more work the voltage regulator has to do to "throw away" the excess voltage, so with a 12V supply, and powering quite a few "peripherals" the voltage regulator is likely to get hot.

On the other hand, the internal circuitry is designed to switch from USB to external if the external supply exceeds 6.6V, so supplying much less than 7V probably won't work properly.