Embedded Systems

Master 's Degree in Informatics Engineering

Open Source Electronic Prototyping Platform

Why this platforms?

Why is openness important in hardware?

"*Because open hardware platforms become the platform where people start to develop their own products*”

Massimo Banzi

co-creator of Arduino

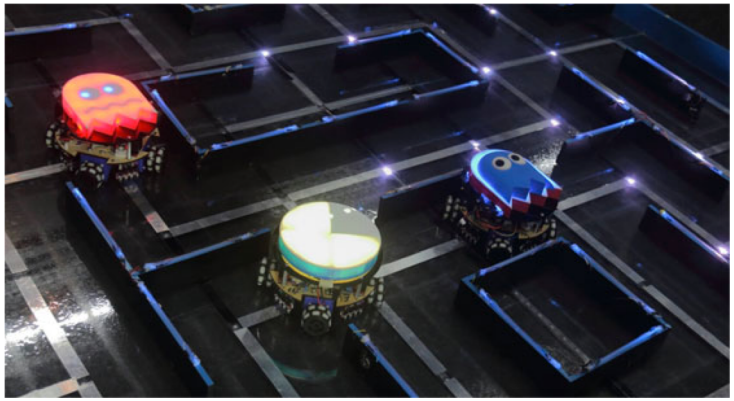
Why this platforms?

- Starting Point – Software
  - Currently, software development is a “common” skill.
  - Free distribution software licenses allow the users to have the control.
  - The target platform is based on computer systems

- Starting Point – Hardware
  - One step beyond – We want to develop the hardware components
  - Electronic skills aren’t “common”
  - These platforms are program and simple wired based

Why this platforms?

- What can we do?
  - Develop a physical prototype
  - Computational capabilities
  - Program based ➔ Dynamic behavior



- Future?
  - Internet of things is a reallity
  - OMG but what about mechanical skills? → 3D printers are the solution

How many platforms?

PanStamps

Description:
"panStamps are small wireless modules programmable from the Arduino IDE. Each module contains an Atmega328p MCU and a Texas Instruments CC1101 RF interface, providing the necessary connectivity and processing power to create autonomous low-power wireless motes".

panStamp NRG: $18.95
- relies on a powerful CC430F5137 SOC

Price: $18.55

Specs & Features:

- **Atmel Atmega328P** at 8MHz
- Operating voltage: from 2.5 VDC to 3.6 VDC
- Current consumption: 1 uA when in deep sleep mode

Connectivity: Sub-1 GHz RF Transceiver 868/915 MHz
Open Source: **Yes GNU GPL v2**

TinyDuino

Description:
"The TinyCircuits TinyDuino is an Arduino compatible board in an ultra compact package. Imagine the possibilities of having the full power of an Arduino Uno in a size less than a quarter!"

Price: $19.95

Specs & Features:

- Atmega328P processor 32KB Flash, 2KB RAM, 1KB EEPROM
- Arduino and LilyPad Compatible
- 20 I/Os (14 Digital, 6 Analog / Digital I/O) - All the signals on the Arduino Shield connectors are supported

Connectivity: Via add-on shields (Bluetooth, WiFi, etc)
Open Source: Yes

How many platforms?

Arduino Uno

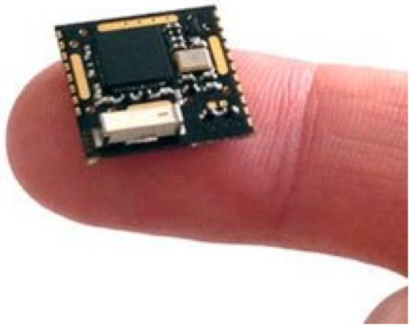
Description:
"Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software. It's intended for artists, designers, hobbyists, and anyone interested in creating interactive objects or environments."

Price: $29.95

Specs & Features:

- **ATmega328** microcontroller
- 14 Digital I/O Pins (6 PWM outputs)
- 32k Flash Memory

Connectivity: Can be extended with shields (Wifi, GSM, Bluetooth, etc)
Open Source: **Yes** GPL and the C/C++ microcontroller libraries are under the LGPL



RFduino

Description:
"A finger-tip sized, Arduino compatible, wireless enabled microcontroller, low cost enough to leave in all of your projects! "

Price: **$21 and up**

Specs & Features:

- **Nordic 32 bit ARM Cortex-M0 processor**
- 7 GPIO and fully software selectable and can be remapped as you wish
- Your standard Arduino sketches run on the RFduino

Connectivity: Bluetooth Low-Energy 4.0 built-in
Open Source: Yes

How many platforms?

XinoRF

Description:
"The XinoRF is an Arduino UNO R3 compatible electronics development board with an onboard 2-wayCiseco SRF data radio, which supports over-the-air programming."

See Also: RFμ-328

Price: £30.00

Specs & Features

- ATmega328 P-PU micro-controller (32kb flash, 2kb RAM, 1kb EEPROM)
- Lots of I/O - 14 DIO (6 PWM, 6 analog, 40ma output)
- Over the air programming

Connectivity: SRF-U wireless
Open Source: Yes

Ciseco: OpenKontrol Gateway

Description:
"The OpenKontrol Gateway is the product everyone (including us) has been eagerly waiting for.

It supports WiFi, low power RF (many types), Ethernet and Bluetooth. It's designed to be used 24 hours a day so it's incredibly low power at just half a watt (based on XRF radio, XV wifi module, SD, RTC and SRAM)."

Price: £30.90-

Specs & Features:

- **Atmel 328** (with pre-loaded UNO bootloader)
- SD Card Reader Kit (Surface mount SD card socket)
- 32k SRAM Memory
- RTC Kit (DS1307, socket, crystal, CR2032 coin cell, coin cell holder, 5V regulator, capacitor)

Connectivity: Ethernet and expandable to GSM, XRF radio module, RN-XV WiFi module
Open Source: Yes

How many platforms?

Pinoccio

Description:
"A wireless, web-ready microcontroller with WiFi, LiPo battery, & built-in radio. An API to get your board talking to the Web right out of the box."

Price: **$49.00-99.00**

Specs & Features:

- **Atmel ATmega256RFR2** with built-in radio
- 17 digital I/O pins
- 8 analog input pins
- 16MHz MCU32k SRAM
- LiPo rechargeable battery
- On-board temperature sensor

Connectivity: 802.15.4 & WiFi
Open Source: Yes

Raspberry Pi

Description:
"The Raspberry Pi is a single-board computer developed in the UK by the Raspberry Pi Foundation. The Raspberry Pi is a credit-card sized computer that plugs into your TV and a keyboard. It's a capable little PC which can be used for many of the things that your desktop PC does."

Price: $35

Specs & Features:

- **Broadcom BCM2835** 700MHz **ARM1176JZFS** processor with FPU and Videocore 4 GPU
- 512 Megabytes of RAM
- HDMI
- SD Card socket

Connectivity: Ethernet. Expandable with USB and shields to other options (See: **Raspberry Pi Wireless Options**)
Open Source: Partial

How many platforms?

BeagleBone Black

Description:
"BeagleBone Black is a community-supported development platform for developers and hobbyists. Boot Linux in under 10 seconds and get started on development in less than 5 minutes with just a single USB cable."

Price: $45

Specs & Features:

- **AM335x 1GHz ARM® Cortex-A8**
- 2GB of on-board flash and a microSD card reader
- HDMI
- 2x 46 pin headers

Connectivity: Ethernet
Open Source: Partial

CubieBoard

Description:
"The Cubieboard is a new Allwinner A10 based developer board, with a very wide range of IO options. The board is set apart by offering SATA and an extended pin interface for low level access to the SOC."

Price: $49.00

Specs & Features:

- 1G **ARM cortex-A8 processor**, NEON, VFPv3, 256KB L2 cache
- 512M/1GB DDR3 @480MHz
- 96 extend pin including I2C, SPI, RGB/LVDS, CSI/TS, FM-IN, ADC, CVBS, VGA, SPDIF-OUT, R-TP.
- HDMI

Connectivity: Ethernet
Open Source: Partial

How many platforms?

Nanode

Description:
"Nanode is an open source Arduino-like board that has in-built web connectivity.

It is a low cost platform for creative development of web connected ideas. It's like an Arduino with an Ethernet shield built in. Easily upgradable to wireless by adding a 433Mhz radio kit".

Price: $39-$56

Specs & Features:

- ATmega328P microcontroller
- Up to 14 digital I/O lines
- 32KB ISP flash memory
- SPI expansion memory (SRAM, Flash or FRAM)

Connectivity: Ethernet & optional 868MHz 433MHz Boards and devices
Open Source: Yes

WeIO

Description:
"WeIO is an innovative open source hardware and software platform for rapid prototyping and creation of wirelessly connected interactive objects using only popular web languages such as HTML5 or Python.

With WeIO making connected objects becomes as simple as making websites.".

Price: 59 euro

Specs & Features:

- WeIO is based on Python powered Tornado WebSocket server
- 16 MB Flash and 64 MB DDR2 RAM
- 32 x GPIO • 1 x UART

Connectivity: WiFi IEEE 802.11bgn 1x1 2.4 GHz integrated into AR9331 processor
Open Source: GPL Licence 3

How many platforms?

Arduino Yun

Description:
"Arduino Yún is the combination of a classic Arduino Leonardo with a Wifi system-on-a-chip running Linino.

When the Yún is turned on for the first time, it becomes an Access Point, creating a Wi-Fi network named “Arduino”.

Price: $69

Specs & Features:

- Atmel ATMega32u4 @ 16 MHz with 2.5KB SRAM and 32KB flash
- microSD card slot
- 14 digital input/output pins

Connectivity: Ethernet & WiFi
Open Source: **Yes** GPL and the C/C++ microcontroller libraries are under the LGPL

mbed - LPC1768

Description:
"The mbed Microcontrollers are a series of ARM microcontroller development boards designed for rapid prototyping.

The platform includes a standards-based C/C++ SDK, a microcontroller HDK and supported development boards."

Price:**$49.95**
- See Also: **mbed FRDM KL25Z** ($12.95)

Specs & Features:

- 32-bit **ARM Cortex-M3** core running at 96MHz
- Web-based C/C++ programming environment
- 512KB FLASH, 32KB RAM
- USB Host/Device, 2xSPI, 2xI2C, 3xUART, CAN, 6xPWM, 6xADC, GPIO

Connectivity: Ethernet
Open Source: Partial

How many platforms?

Wi-Go Module

Description:
"A complete wireless data acquisition system with multiple sensors, flash storage and 800mAh battery."

Price: $108

Specs & Features:

- Processor: **Freescale KL25Z** (ARM Cortex M0+) at 48MHz
- Sensors: Accelerometer (MMA8451Q), Magnetometer (MAG3110), Altimeter (MPL3115A2), Ambient Light Sensor (TEMT6200)
- Data Storage: 2 MB SPI Serial Flash (S25FL216K)
- Battery: 800mAh Lithium-Polymer

Connectivity: Wi-Fi 802.11bg (**Murata LBWA1ZZVK7**)
Open Source: Yes (Keil 4.70 source project)

pcDuino

Description:
"pcDuino is a high performance, cost effective mini PC platform that runs PC like OS such as Ubuntu and Android ICS. The platform could run full blown PC like OS with easy to use tool chain and compatible with the popular Arduino ecosystem such as Arduino Shield. "

Price: $59.95

Specs & Features:

- **1GHz ARM Cortex A8 CPU**
- 1GB DRAM
- Onboard Storage: 2GB Flash, microSD card
- Arduino-Style Peripheral Headers
- HDMI

Connectvitiy: Ethernet
Open Source: Partial

How many platforms?

OpenPicus Flyport WiFi

Description:
"Flyport is a powerful and low-cost system on module (SOM) with embedded Internet connectivity.
Flyport transforms a sensor into an internet datalogger, a simple relay - into a remote controlled automation and much more."

Price: € **39.00**

Specs & Features:

- **Microchip PIC24FJ256 16bi**
- Low power - hibernation mode supported
- Peripherals: up to 18 Digital I/O, 4 Analog Inputs (10bits ADC), 4 UARTs, SPI, I2C
- Flash memory: 16Mbit external flash

Connectivity: WiFi, Other options include Ethernet and GPRS
Open Source: Yes

Hackberry

Description:
"Based on the popular 1.2Ghz ARM Allwinner A10, the Hackberry A10 developer board is a powerful, hackable Android / Linux PC. The Hackberry A10 has both WiFi and Ethernet."

Price: $**65.00**

Specs & Features:

- **1.2GHz Allwinner A10 ARM Cortex A8**
- DDR3 512MB / 1GB
- 4GB NAND storage
- HDMI

Connectivity: 10/100 Ethernet, Realtek 802.11n WiFi
Open Source: No

UDOO

Description:
"UDOO is a multi development platform solution for Android, Linux, Arduino™ and Google ADK 2012. The board is designed to provide a flexible environment that allows to explore the new frontiers of the Internet of Things."

Price: $109-129.00

Specs & Features:

- **Freescale i.MX 6 ARM Cortex**-A9 CPU Dua/Quad core 1GHz
- Atmel SAM3X8E ARM Cortex-M3 CPU
- RAM DDR3 1GB
- 54 Digital I/O + Analog Input (Arduino-compatible R3 1.0 pinout)
- HDMI and LVDS + Touch (I2C signals)

Connectivity: Ethernet, WiFi
Open Source: Yes - CC Attribution Share-Alike license

Libelium Wasmote

Description:
"Waspmote is an open source wireless sensor platform specially focused on the implementation of low consumption modes to allow the sensor nodes ("motes") to be completely autonomous and battery powered, offering a variable lifetime between 1 and 5 years depending on the duty cycle and the radio used."

Price: €153.00-

Specs & Features:

- **ATmega1281**
- Over the air programming
- **60 sensors available to connect to Waspmote**
- On-Board Temperature and Accelerometer
- Hibernate mode consumers just 0.06μA.

Connectivity: 8 different wireless interfaces including long range (3G / GPRS), medium range (802.15.4, ZigBee, WiFi) and short range (Bluetooth, RFID, NFC)
Open Source: Yes - LGPL license

How many platforms?

The Rascal

Description:
"The Rascal is a small computer that you can use to monitor and control the world remotely. It's like the brains of an iPhone, without the corporate overlord. The Rascal is powerful enough to handle real web traffic, but you don't have to be a professional electrical engineer to use one."

Price: $199 (W/4 GB memory card and power supply)

Specs & Features:

- Web server includes a built-in editor
- **Atmel AT91SAM9G20** 400 MHz, 64 MB RAM.
- Program in Python
- 2 32MByte SDRAM chips

Connectivity: Ethernet
Open Source: **Yes**

But there is also ….

- Mini-ITX platforms

Introducing… 5x5

- Board is 5.5” x 5.8” (140mm x 147mm)
  - Smallest socketed board standard
  - 29% smaller area than Mini-ITX standard
- Fully Featured
  - Supports LGA based CPUs
  - Feature rich I/O selection
  - 2 channels SODIMM
  - 2.5” SATA* or M.2 storage
  - Wired & wireless networking options
  - Support up to 65W CPU thermal design power
- Fixed length and width board dimensions
- Fixed CPU XY location on motherboard
- Value proposition
  - Enables sub-1 liter socketed solution
  - Provides Intel® Celeron® to Intel® Core™ i7 processor scalability
  - Targeted to support both 35W and 65W TDP CPUs

5x5 measures 140x147 mm at 20,580 mm²
Mini-ITX measures 170x170 mm at 28,900 mm²

But there is also ….

- Pico PC boards

Product	ZBOX PI320 pico
SKU	Please consult your regional sales representative
Memory	2GB DDR3L
Storage	32GB eMMC (integrated) Expandable via micro SD/SDHC/SDXC (up to 128GB)
CPU	Intel Baytrail (quad-core)
GPU	Intel HD Graphics
Video Memory	Shared Memory
Display Options	HDMI
Card Reader	3-in-1 (micro SD/SDHC/SDXC)
SATA	N/A
Ethernet	10/100Mbps
WiFi	Onboard 802.11n Wi-Fi & Bluetooth 4.0
USB Ports	3 USB 2.0
Audio	HDMI audio (bitstream) 3.5mm output
DirectX Support	DirectX 11
Other Features	Intel Virtualization Technology (VT-x)
HDCP:	Yes
Windows	Windows 8.1 with Bing (x86) preinstalled

Selecting a platform!!

- Prototype applied to …..
- Which will be the cost?
- Is there any free development IDE?
- Connectivity? (pins, digital/analogic, comm ports, …)
- Available shields (sensors, actuators, ….)
- Documentation and Development community?

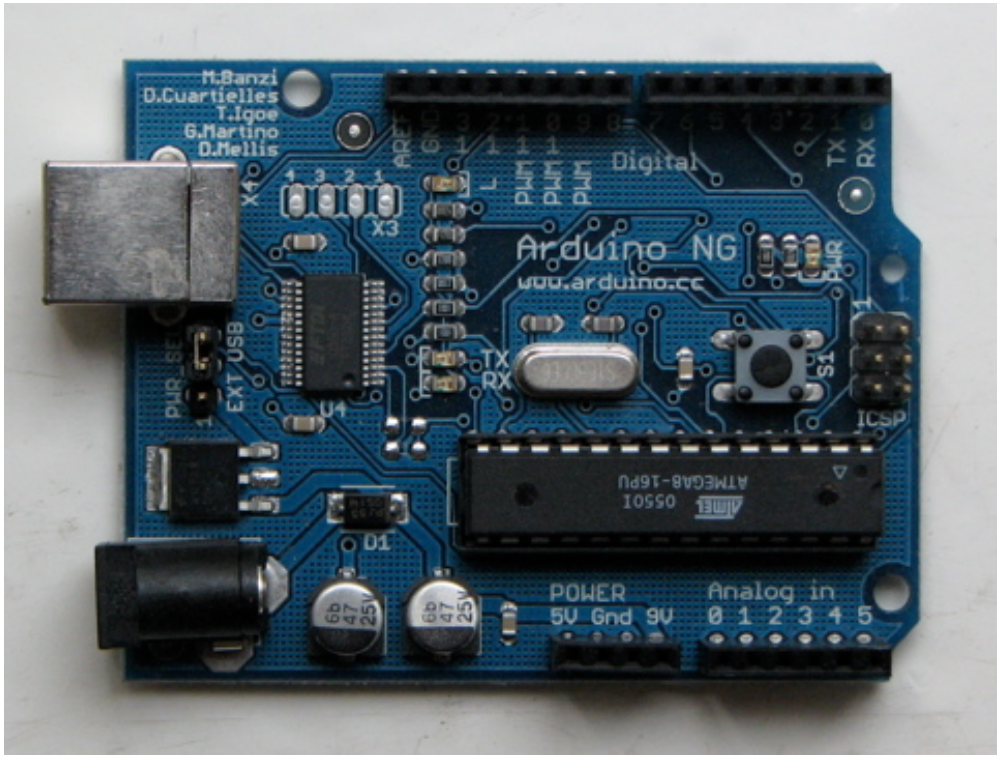
Embedded Systems

Master 's Degree in Informatics Engineering

ARDUINO Platform

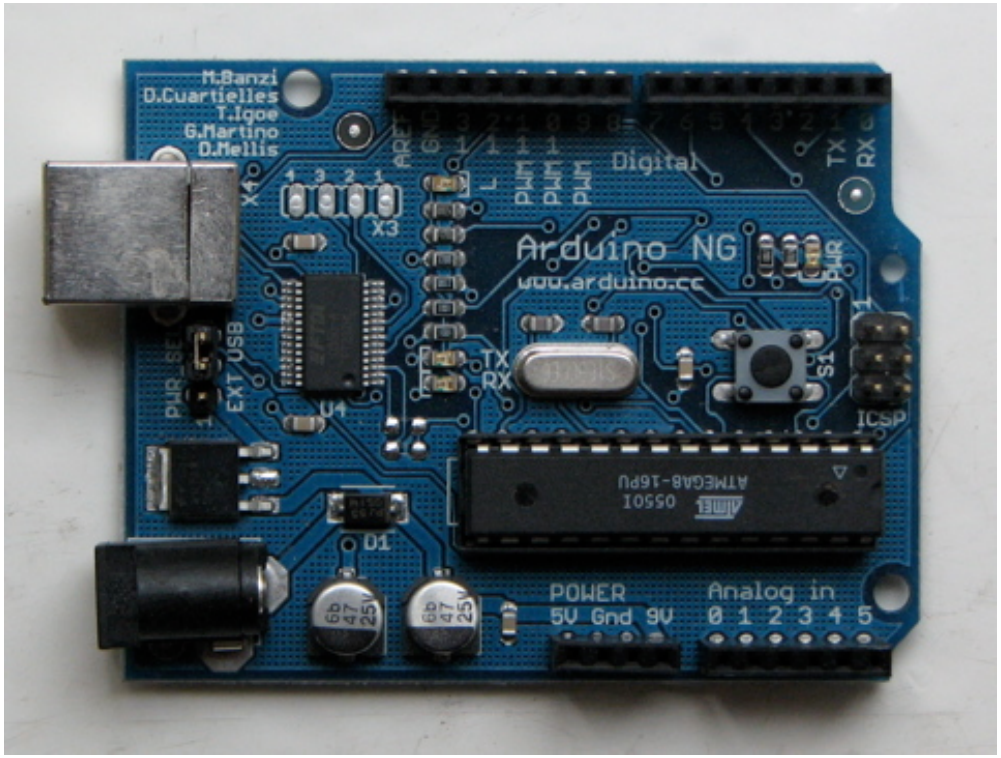
What is Arduino?

- Physical computing platform
- Open source
- “Hardware Abstracted” Wiring Language
- USB programmable
- Large community
- Inexpensive ($31.95 from Sparkfun)

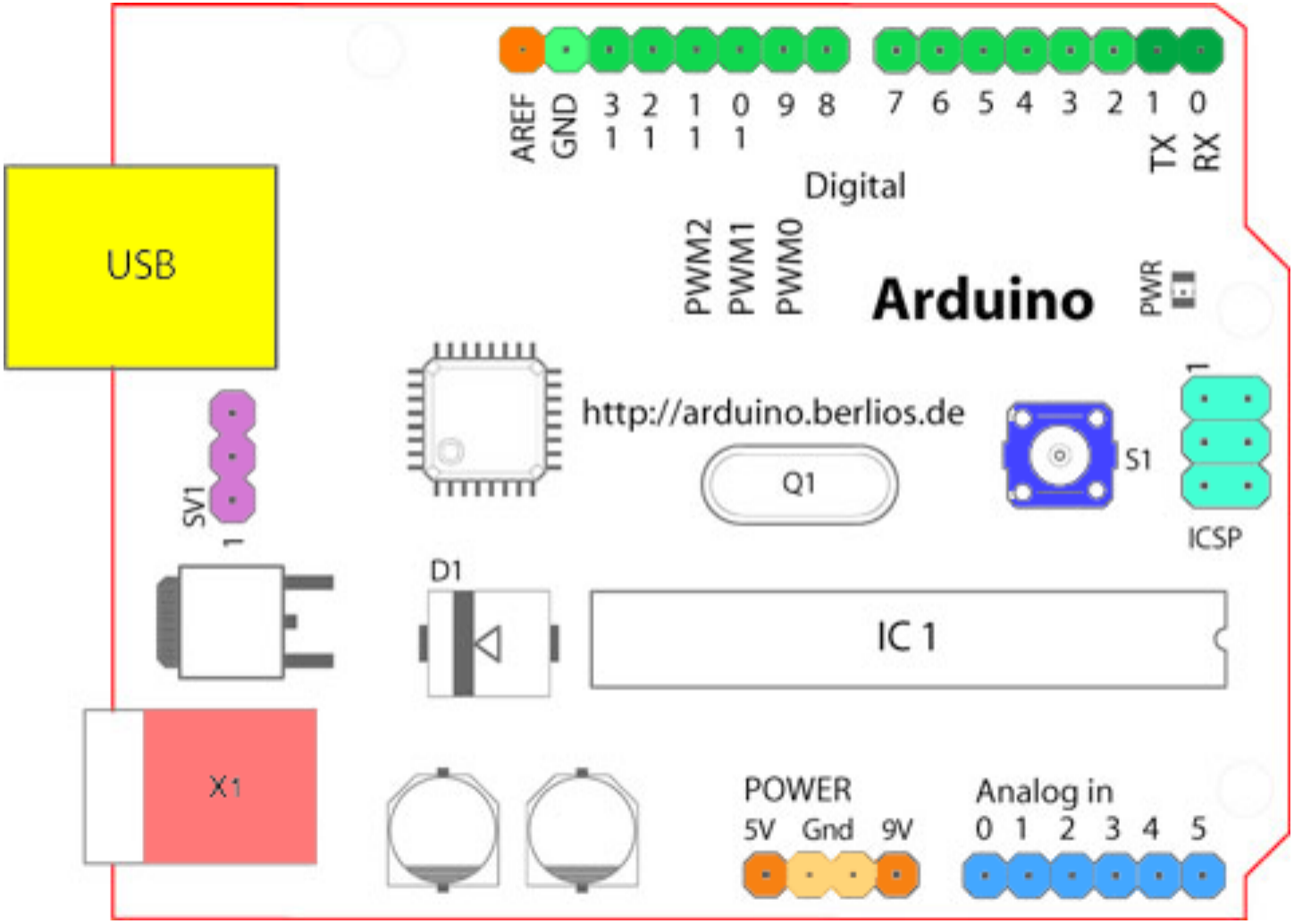


What is Arduino?

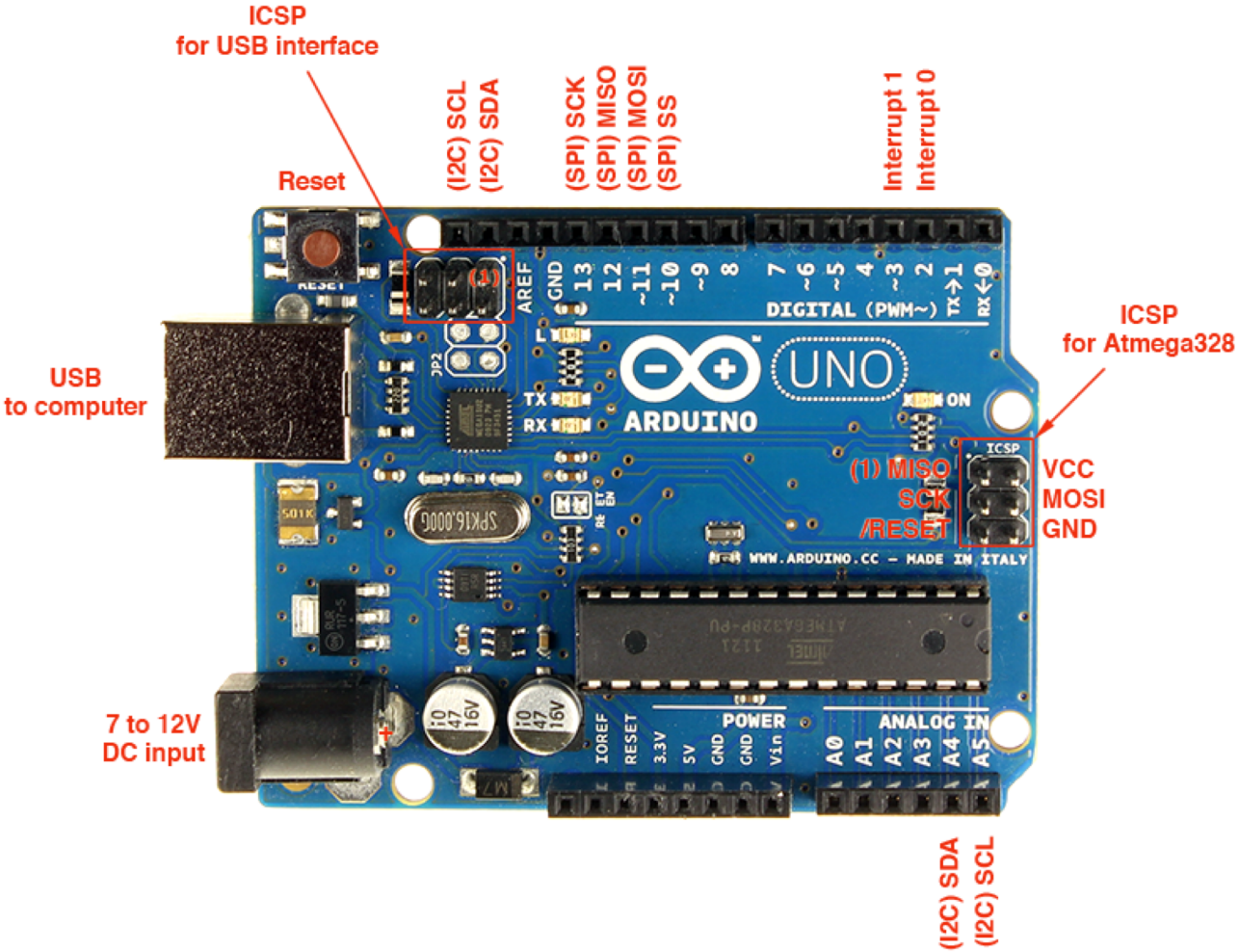
- Based on ATmega8
- USB interface
- Voltage regulator
- The “power” is in:
  - Standard board design
  - Wiring language
  - Open Source



Arduino Board Overview

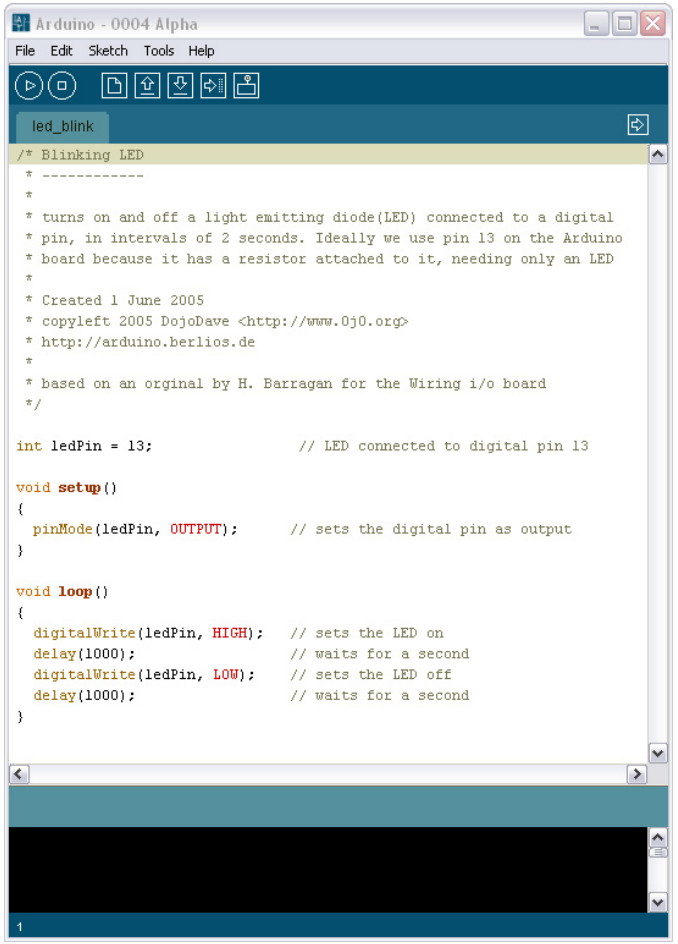


Arduino Board rev.3



Arduino is a platform

- It also includes an Integrated Development Environment (IDE).
- The language itself is based in C / C++ but is largely modeled upon the www.processing.org language.



What is it used for?

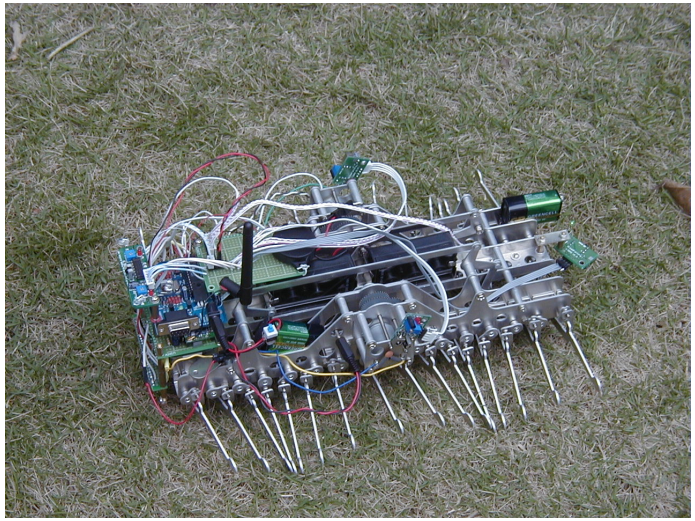
- Physical Computing projects / research
- Interactive Installations
- Rapid prototyping

- When you wish to move beyond the traditional Mouse, Keyboard and Monitor to develop novel and custom interactions in your project work.

What can it do?

- Sensors (to sense stuff)
  - Push buttons, touch pads, tilt switches.
  - Variable resistors (eg. volume knob / sliders)
  - Photoresistors (sensing light levels)
  - Thermistors (temperature)
  - Ultrasound (proximity range finder)

- Actuators (to do stuff)
  - Lights, LED's
  - Motors
  - Speakers
  - Displays (LCD)

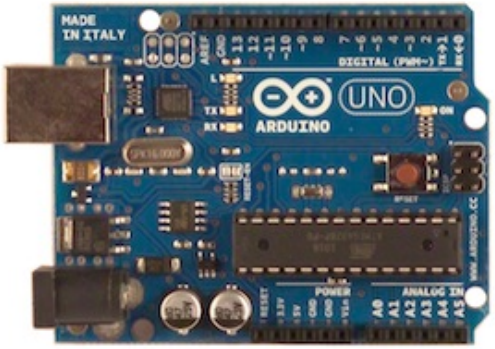


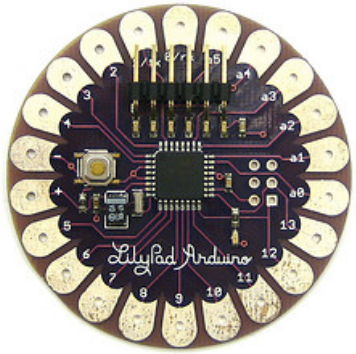
Basic Electrical knowledge

- A fantastic guide to electronics in theory, practice and of course safety is available as a PDF at:

  http://www.ibiblio.org/obp/electricCircuits/

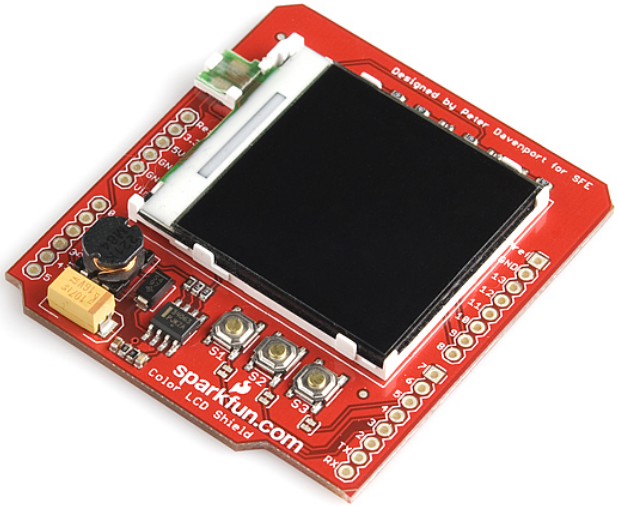
Arduino I/O Boards

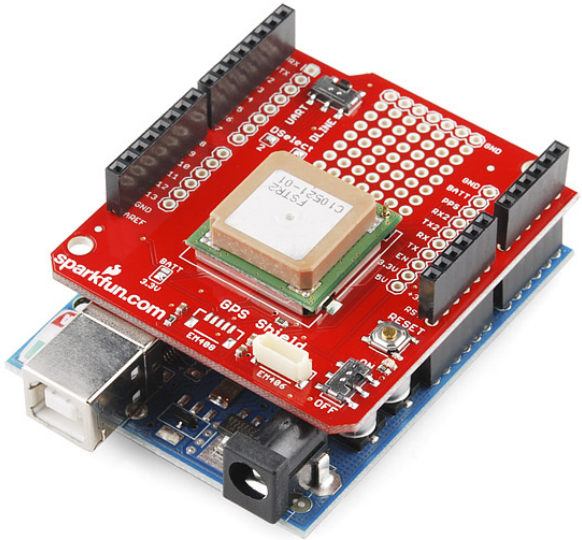


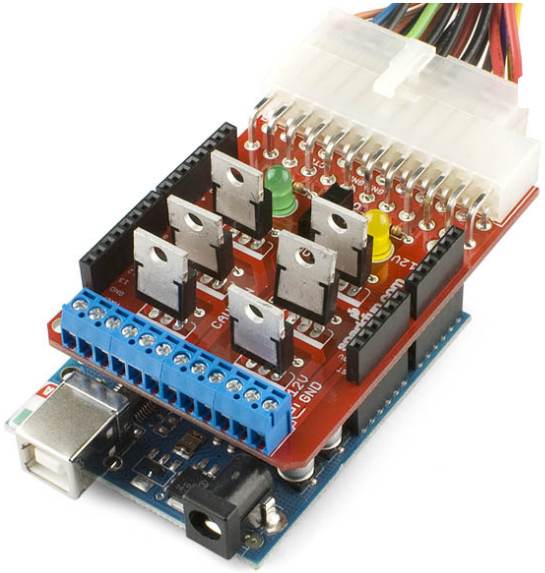


“Shields”


MAC NO.
FCC ID : T9JRN41-1
RoHS compliant


Designed by: Peter Davenport for SFE
sparkfun.com
Color LCD Shield

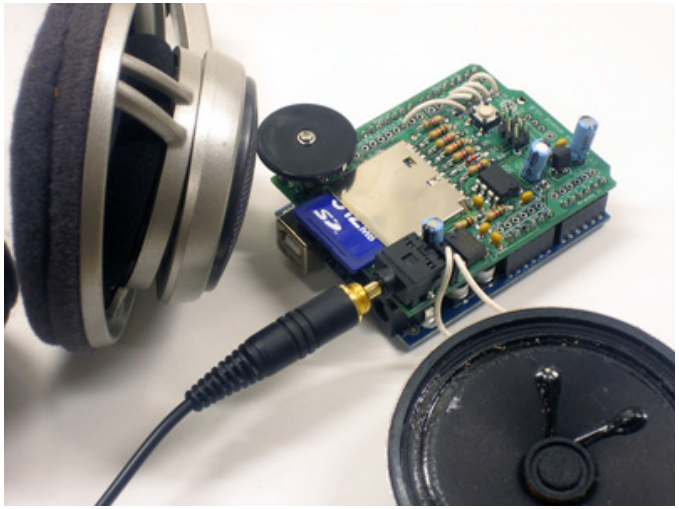

sparkfun.com
GPS Shield



“Shields”

Datalogging Shield

Touch Screen Shield

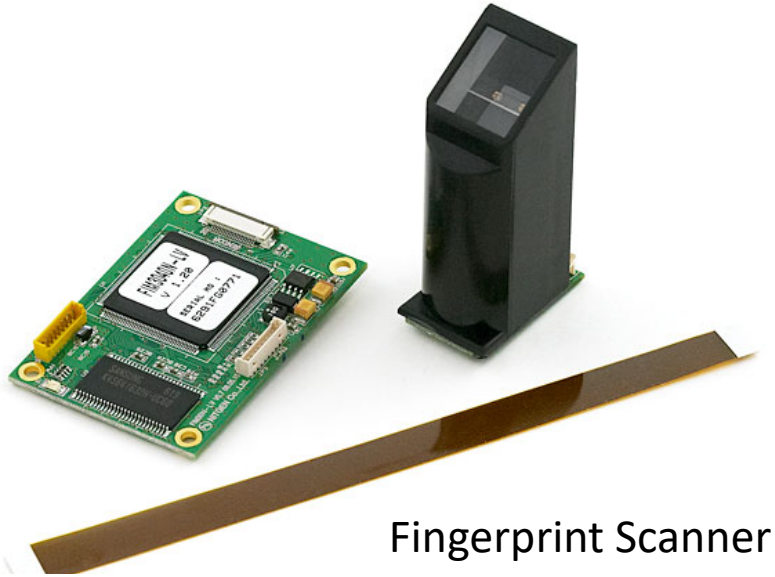


Wave Shield

“Sensors”

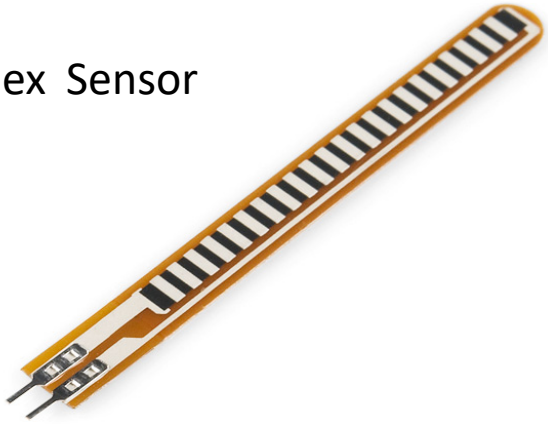
Gas Sensor

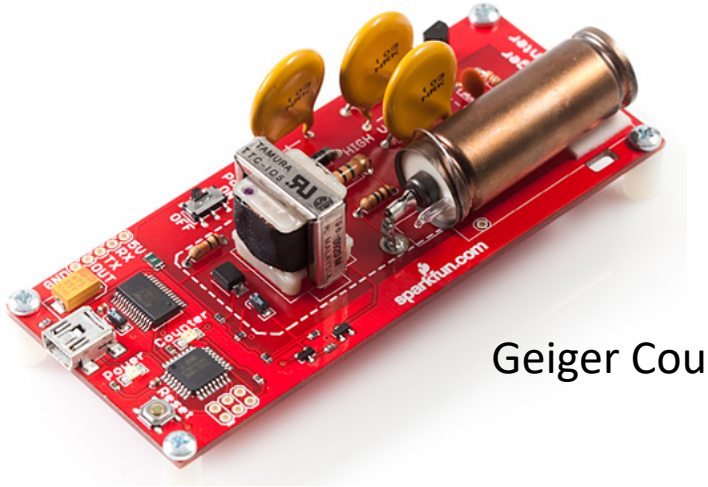
Temp & Humidity



Fingerprint Scanner

Flex Sensor

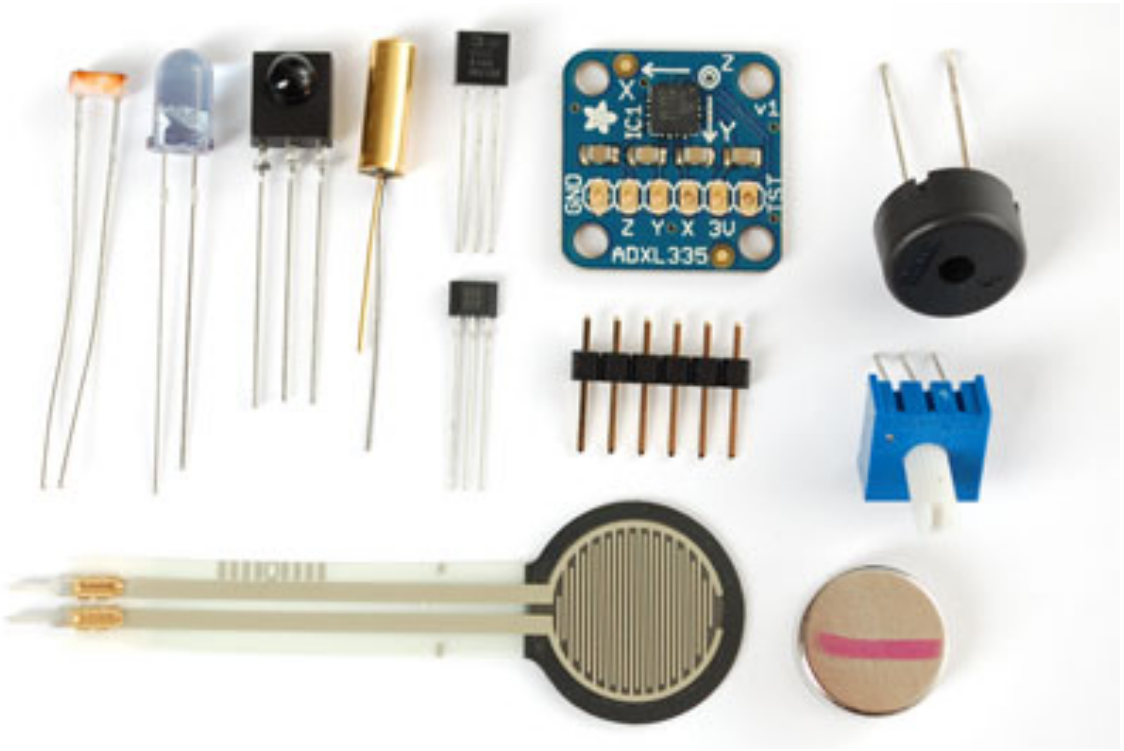




Geiger Counter

“Sensors”

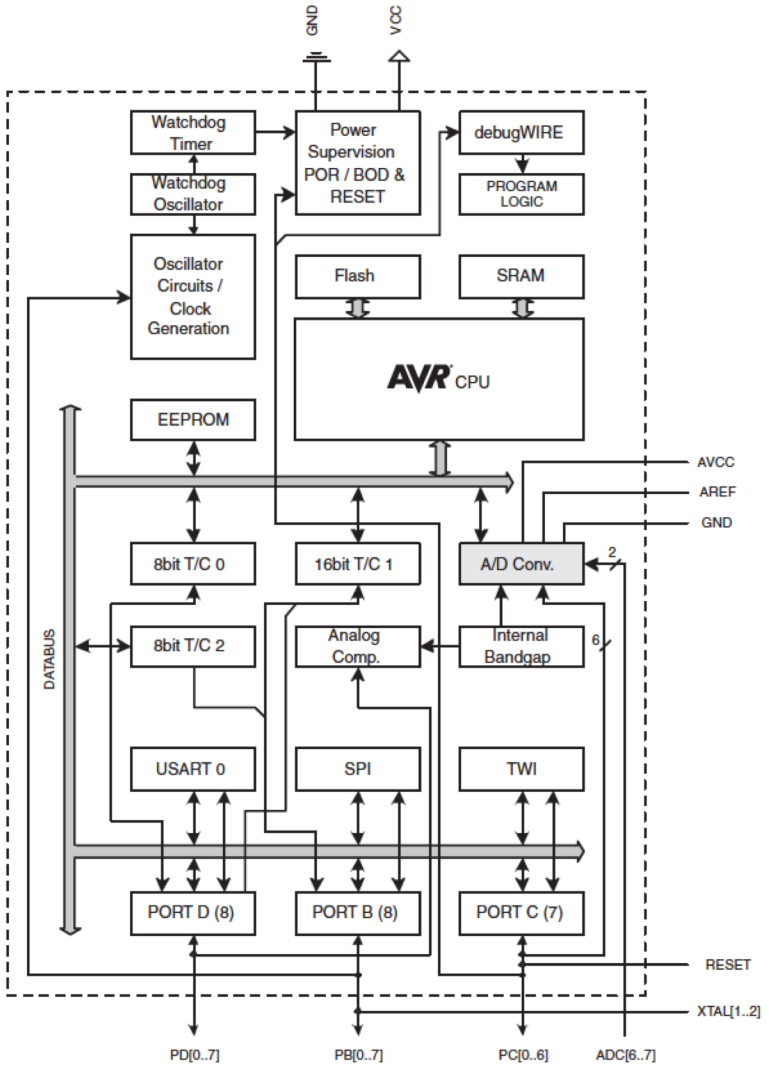
Photo/thermistor, infared, force sensitive resistor, Hall effect, Piezo, tilt sensor..



Arduino microcontroller Atmega328

Atmega328 Internals

AVR Architecture

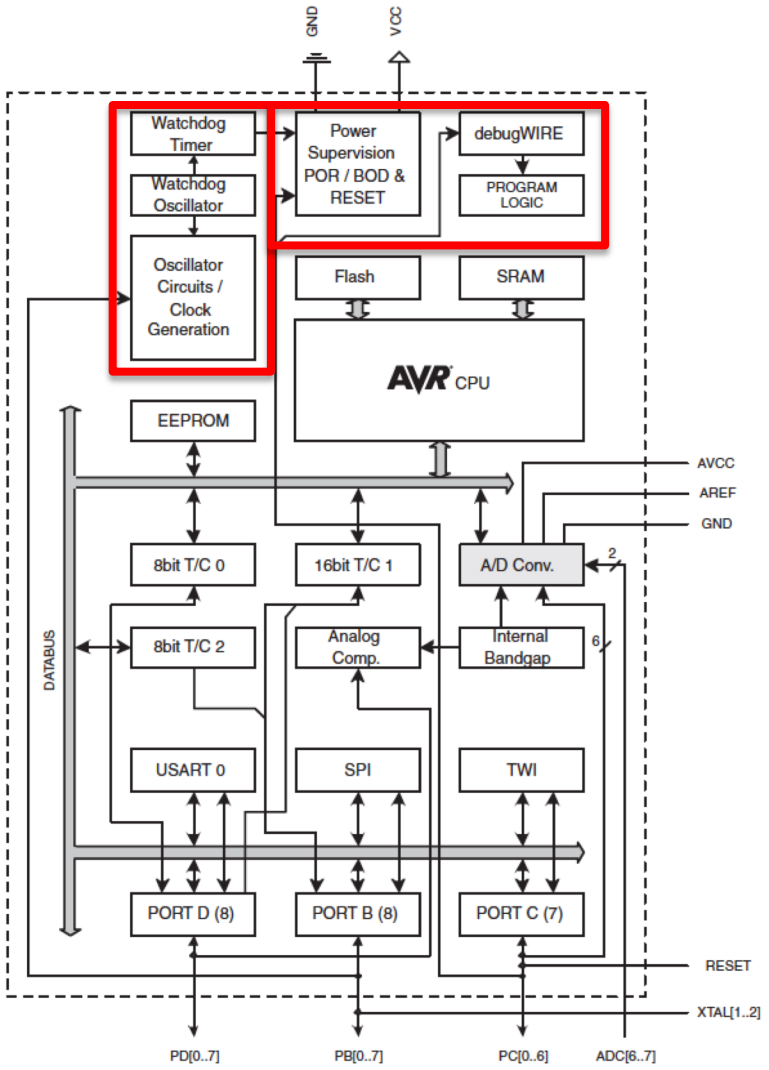


Atmega328 Internals

AVR Architecture

Clock and Power

- 16MHz CPU speed
- Operating Voltage – 5V
- Input voltage (Recommended) – 7-12V
- Input voltage (Limits) – 6-20V

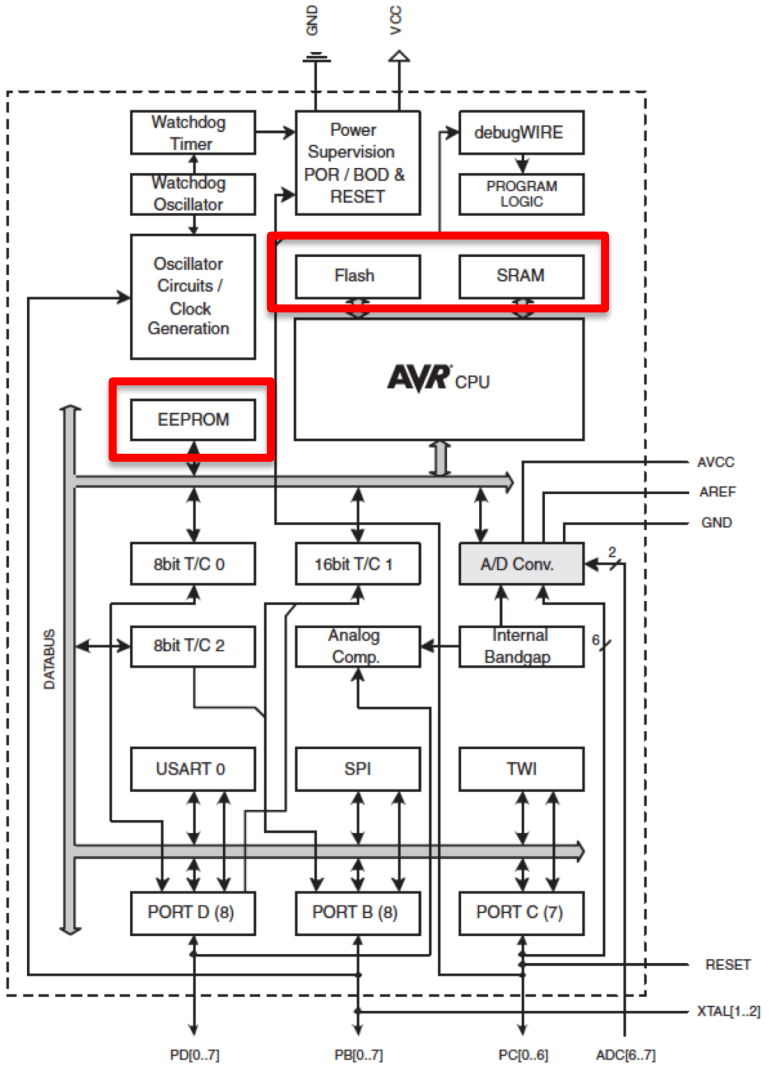


Atmega328 Internals

AVR Architecture

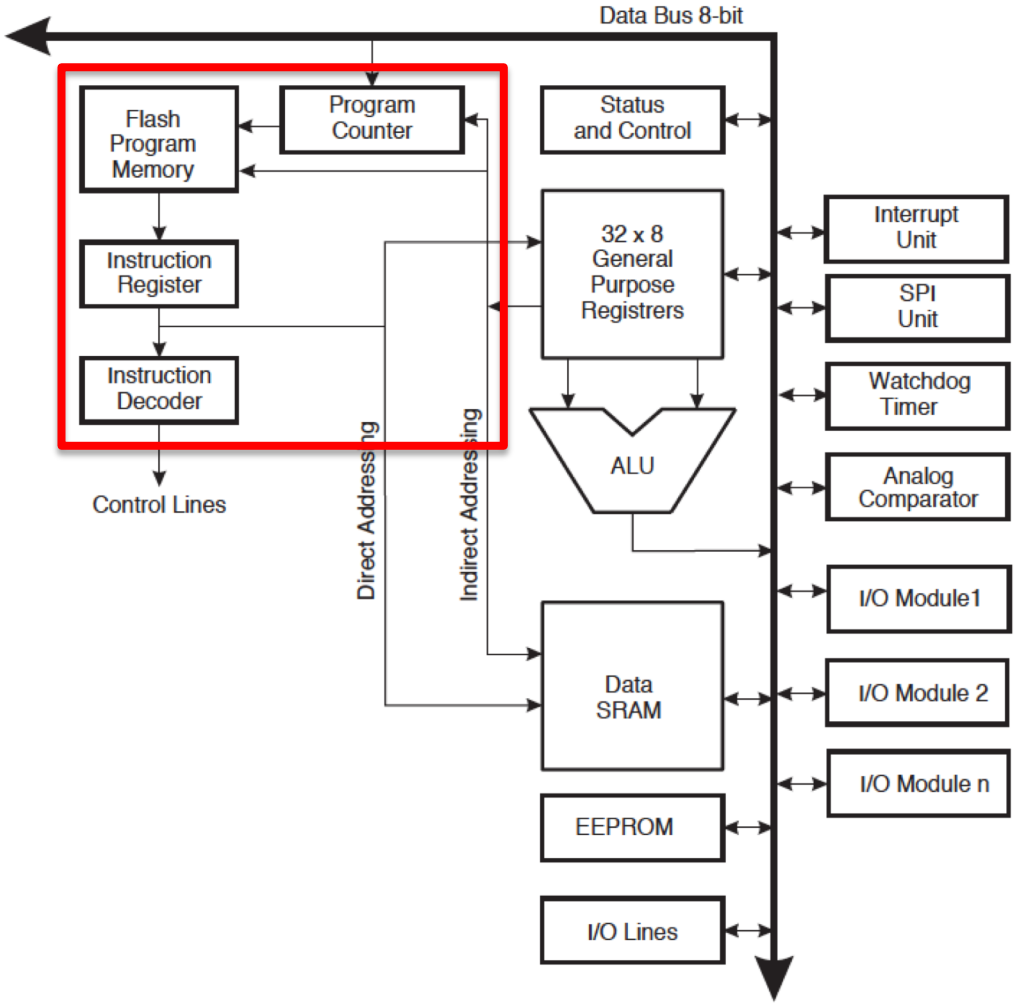
Harvard Architecture

- Flash memory program – **32K**
  - 15bits addressing
  - Read only
  - Non-volatile
  - Able to allocate data (PROGMEM)
- SRAM memory – **2K**
  - Temporary values, stack, etc.
  - Volatile
- EEPROM (long term data) – **1K**
  - Long-term data



Atmega328 Internals

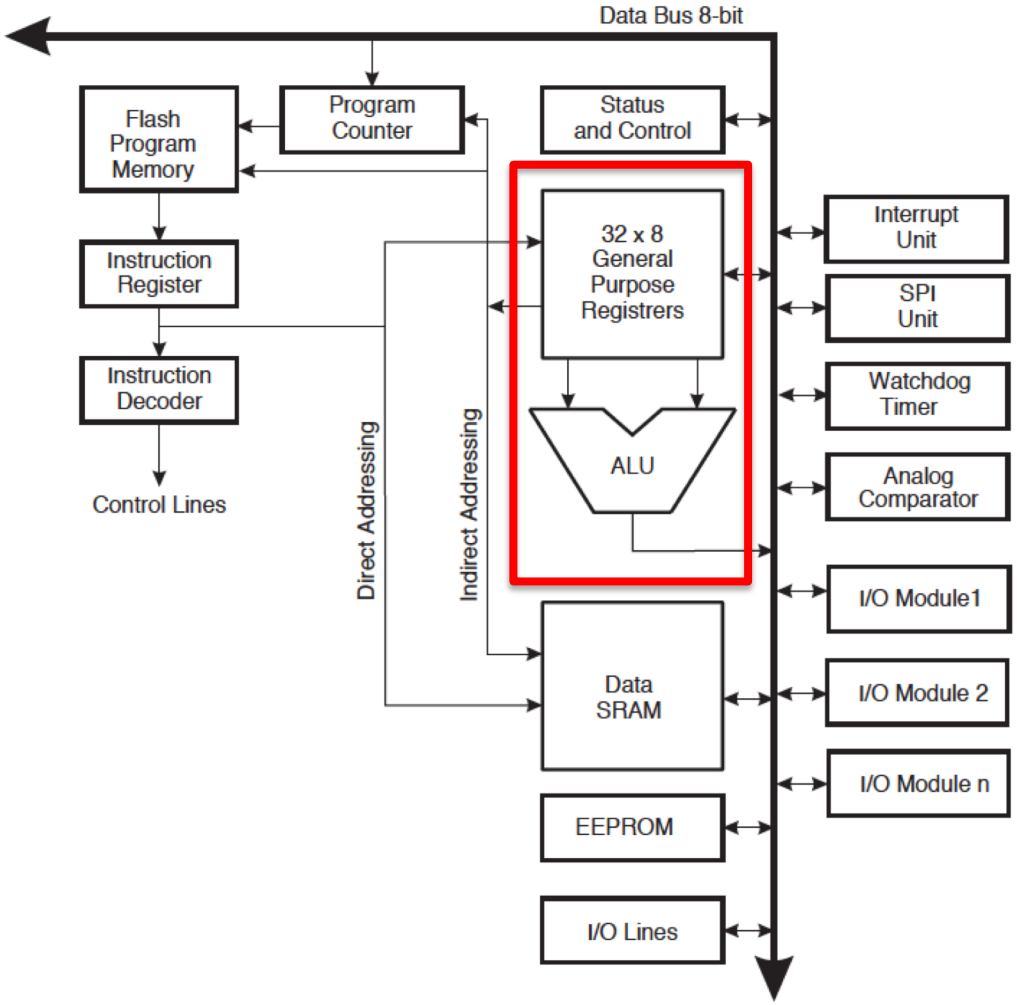
Instruction Fetch and Decode



Atmega328 Internals

Instruction Fetch and Decode

ALU Instructions

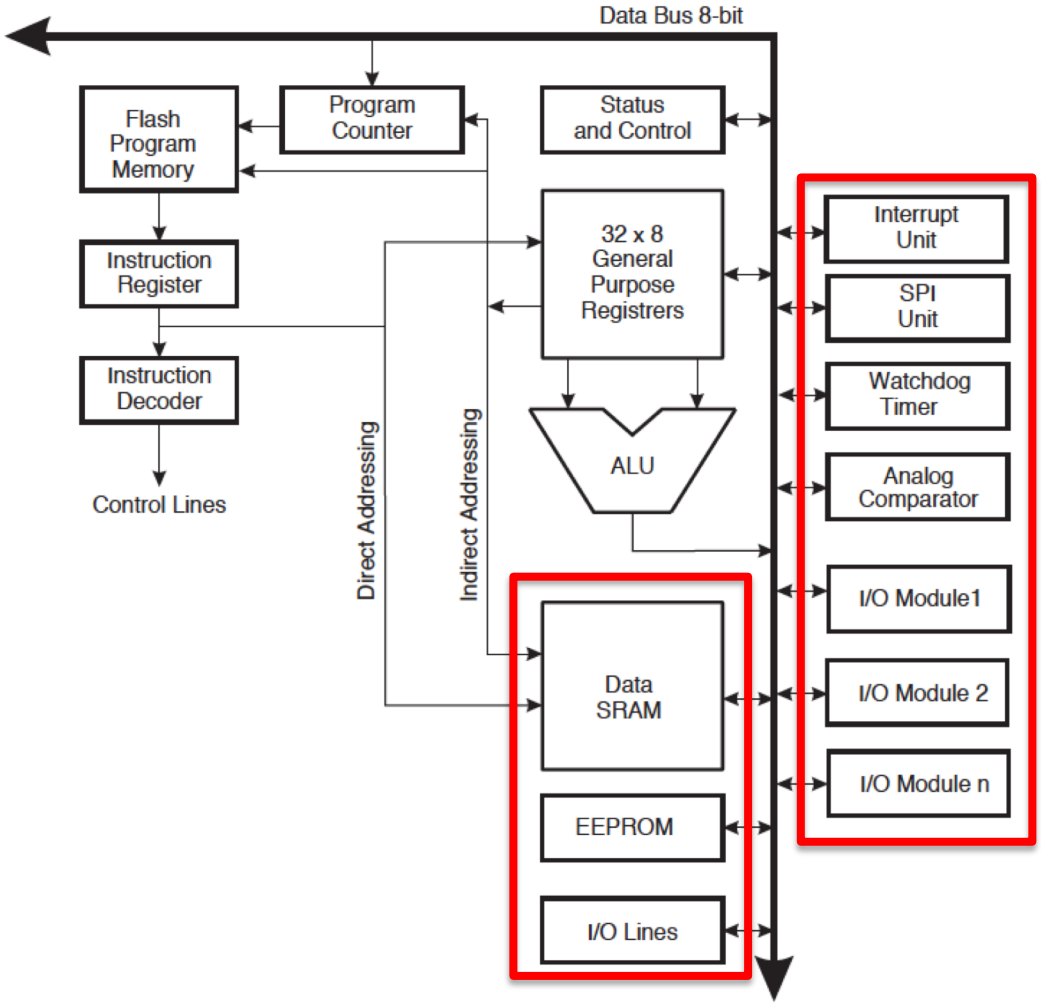


Atmega328 Internals

Instruction Fetch and Decode

ALU Instructions

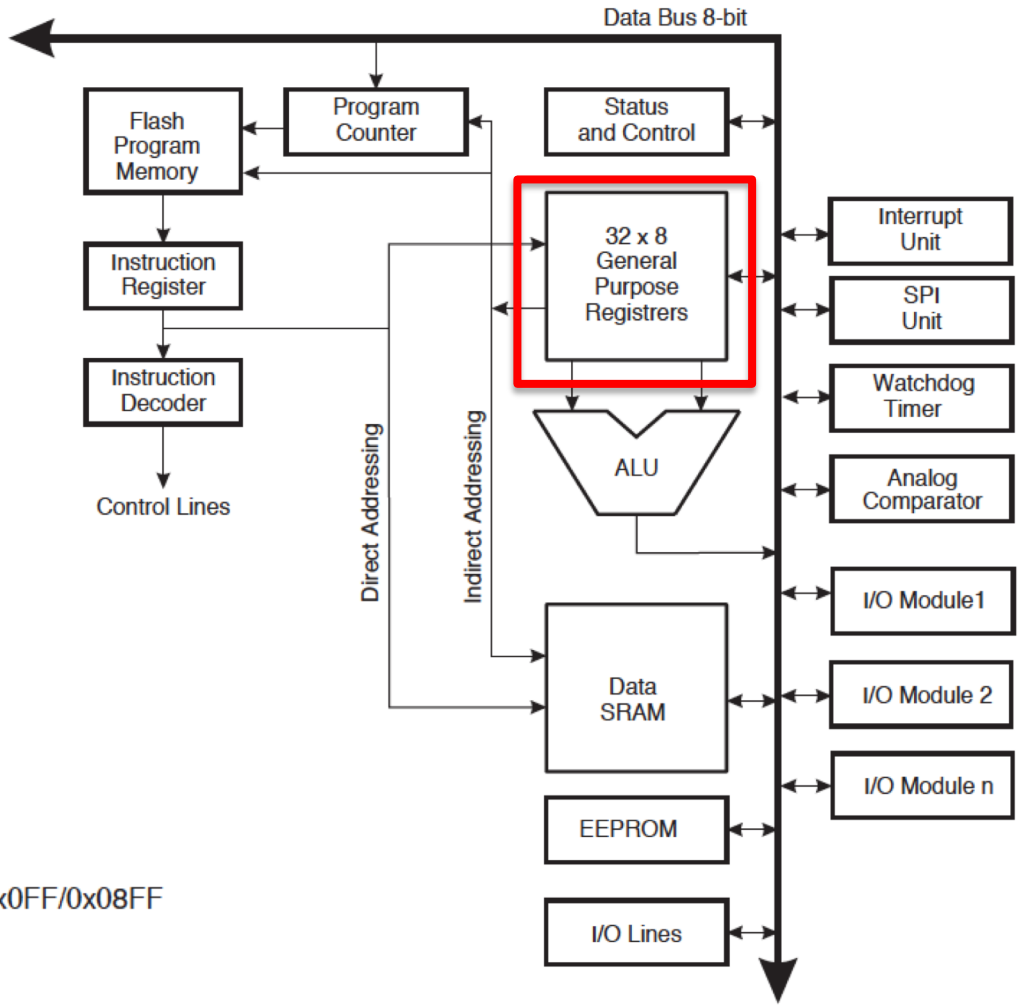
I/O and Special functions



Atmega328 Internals

32 8-bits General Purpose registers [R0..R31]

Mapped on the SRAM addressing space



Data Memory

32 Registers	0x0000 - 0x001F
64 I/O Registers	0x0020 - 0x005F
160 Ext I/O Reg.	0x0060 - 0x00FF
Internal SRAM (512/1024/1024/2048 x 8)	0x0100
	0x04FF/0x04FF/0x0FF/0x08FF

Atmega328 Internals

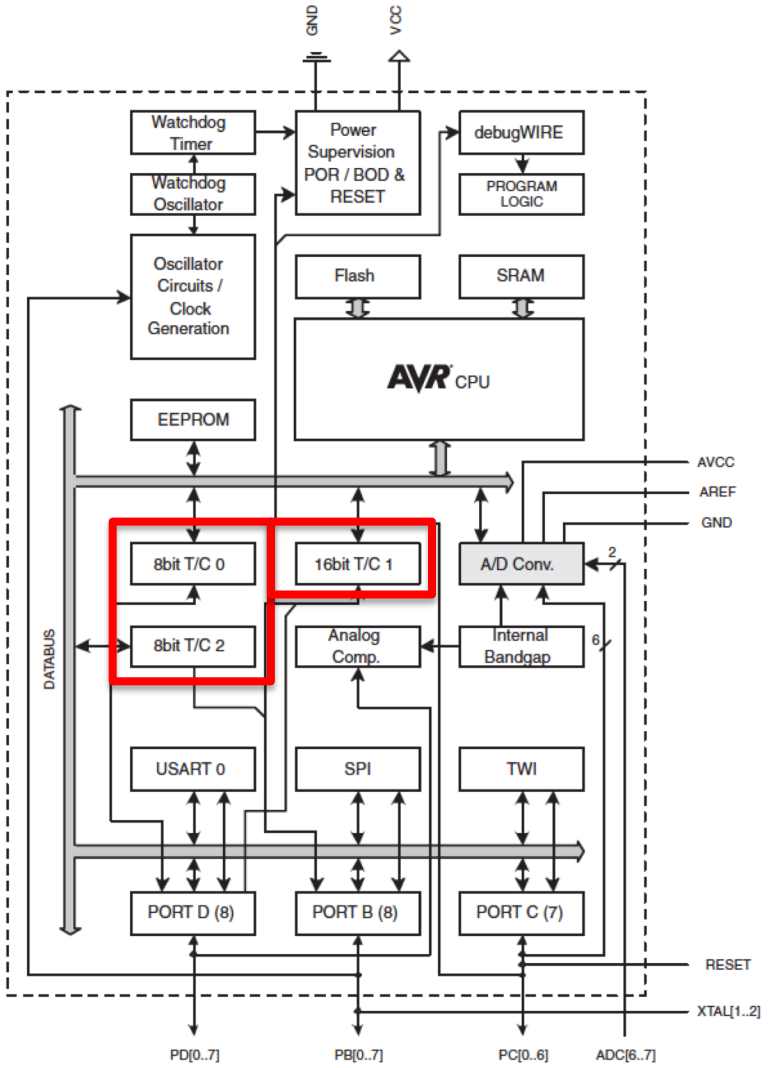
AVR Architecture

Three Programmed Timers

- Choose clock rate
- Choose “roll-over” value
- Generate interrupts
- Generate PWM signals

Timing functions:

- delay(ms)
- delayMicroseconds(us)
- unsigned long millis()
- unsigned long micros() (resolution 4us)

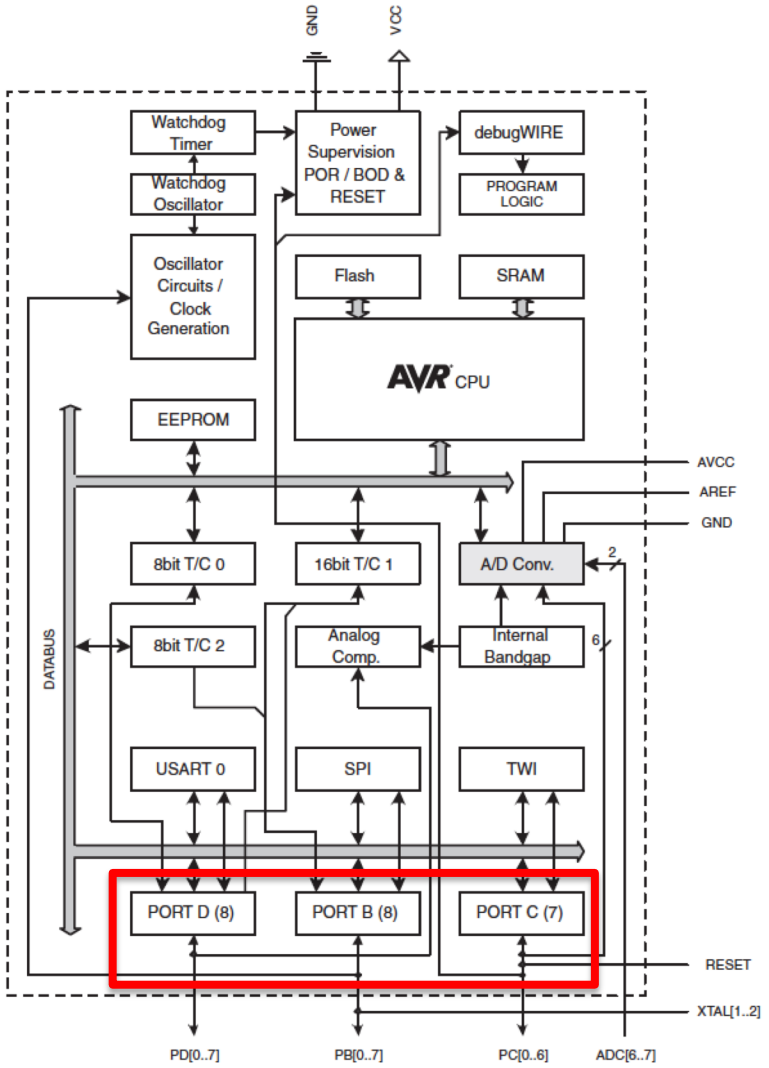


Atmega328 Internals

AVR Architecture

Interface to I/O

- Each pin directly programable
  - Input/Output
  - Program value (LOW/HIGH)
  - Program pull-ups
- Some pins are special
  - Analog vs. Digital
  - Clocks
  - Reset

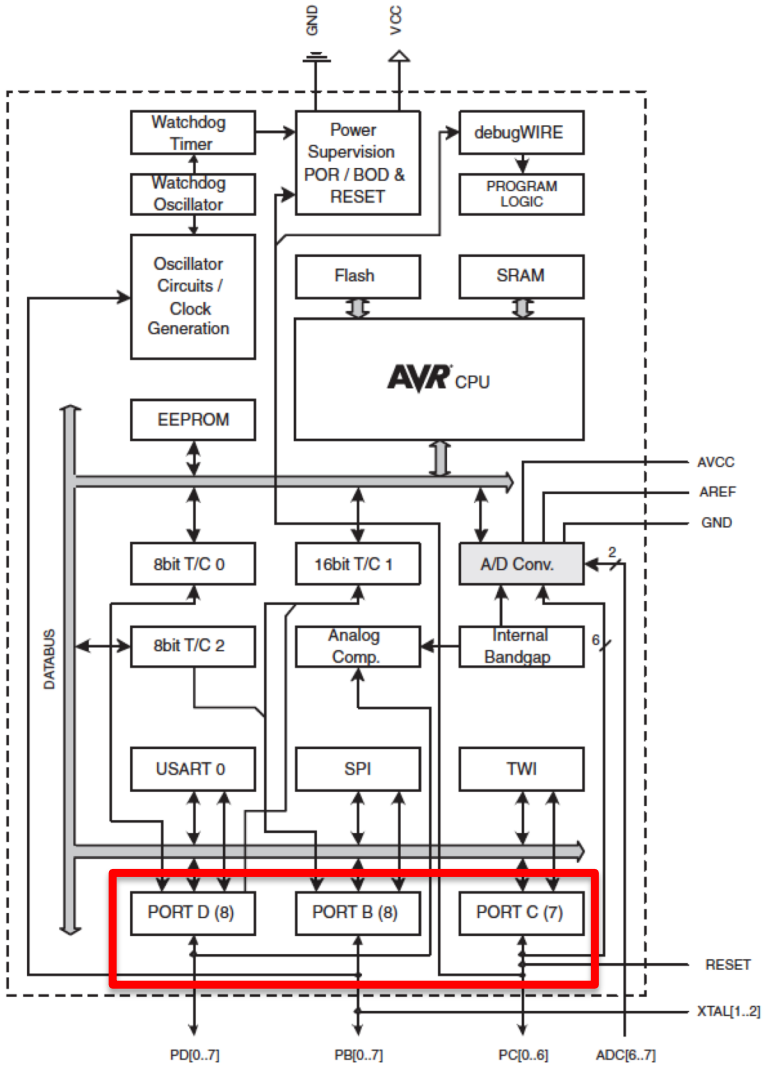


Atmega328 Internals

AVR Architecture

Interface to I/O

- Two 8-bit **PORT**s B & C and 7-bit **PORT** D
- Each one controlled by 3 registers (each bit controls one I/O pin)
  - **DDRx** – Direction register
    - Input – 0
    - Ouput - 1
  - **PINx** – Pin input value
  - **PORTx** – Pin output value
- Using the "Arduino" functions only one pin can be changed – Setting multiple pns requires access to the pin registers

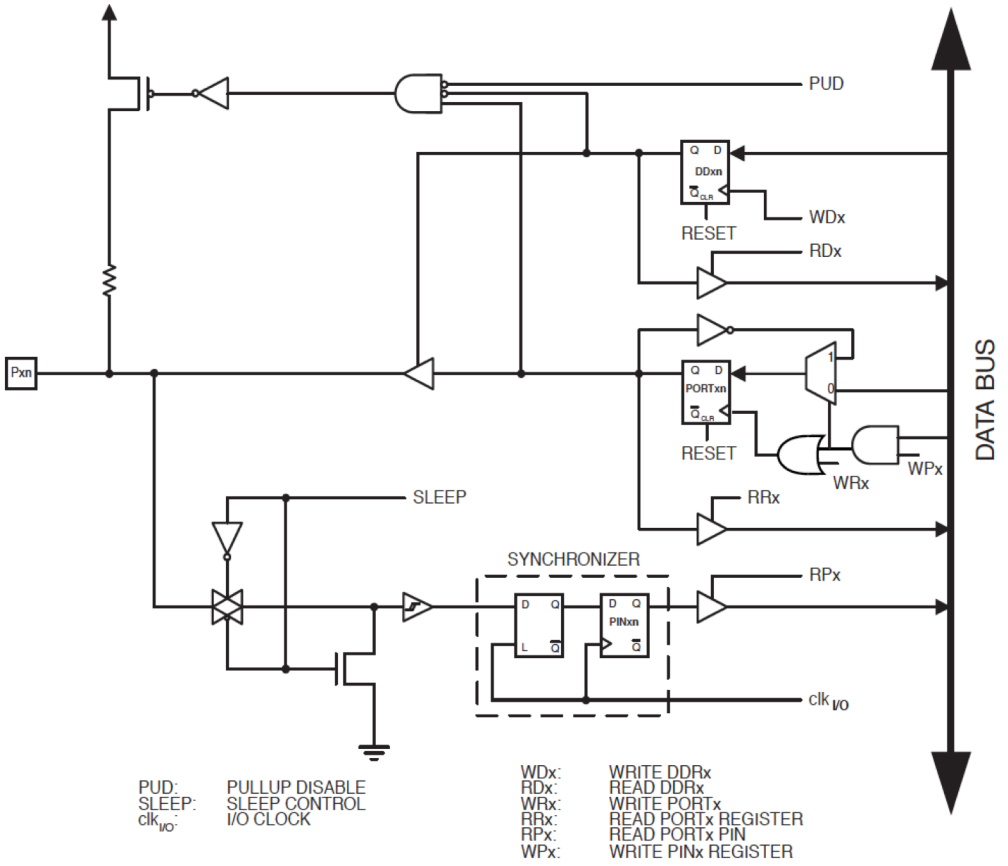


Atmega328 Internals

PIN Circuitry

Interface to I/O

- DDRx – Direction register
  - Input – 0
  - Ouput - 1
- PINx – Pin input value
- PORTx – Pin output value



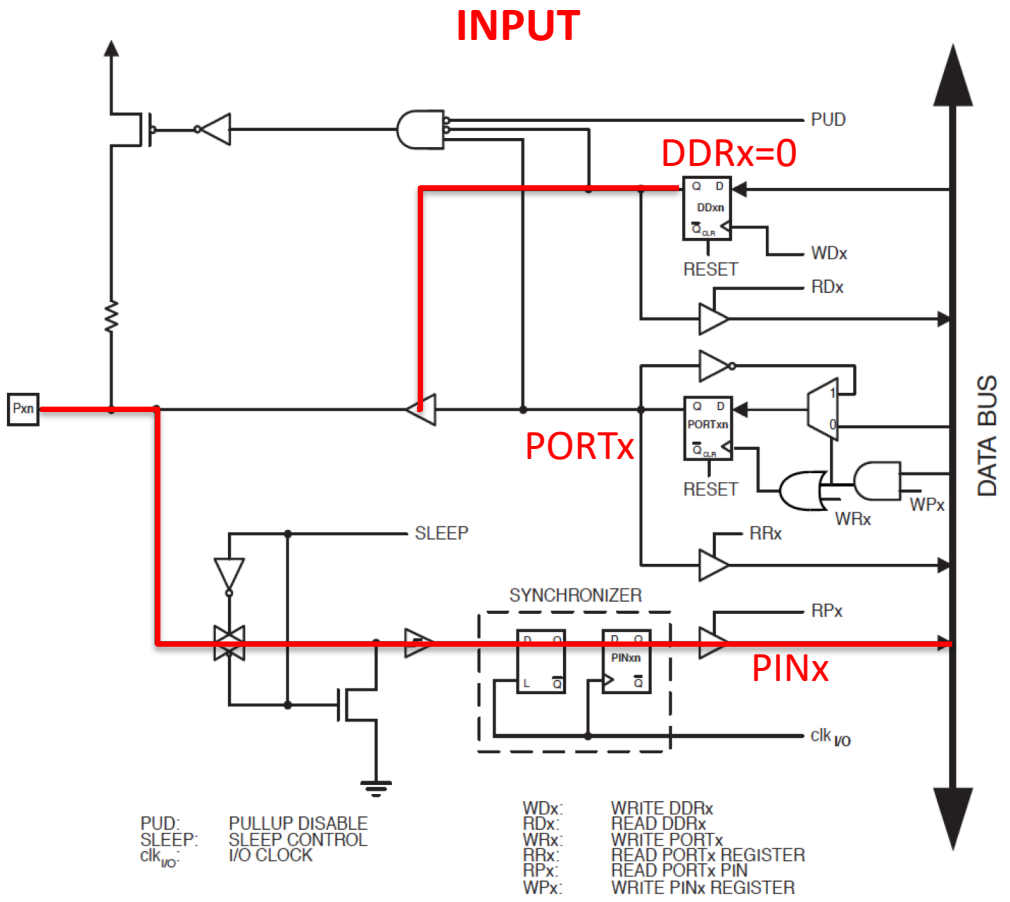
Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

Atmega328 Internals

PIN Circuitry

Interface to I/O – Input

- DDRx = 0
- PORTx = 0 – Pin is floating
- PORTx = 1 – Pull-up (allows OR bus)



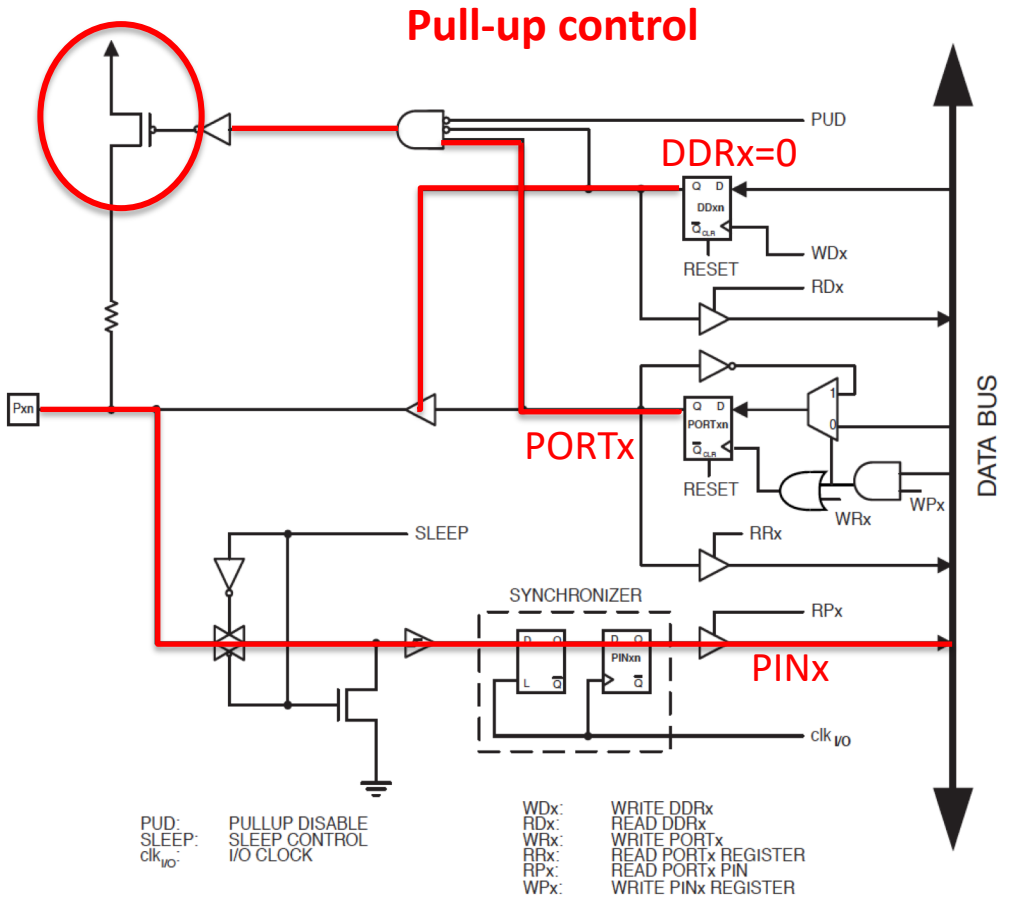
Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

Atmega328 Internals

PIN Circuitry

Interface to I/O – Input

- DDRx = 0
- PORTx = 0 – Pin is floating
- PORTx = 1 – Pull-up (allows OR bus)



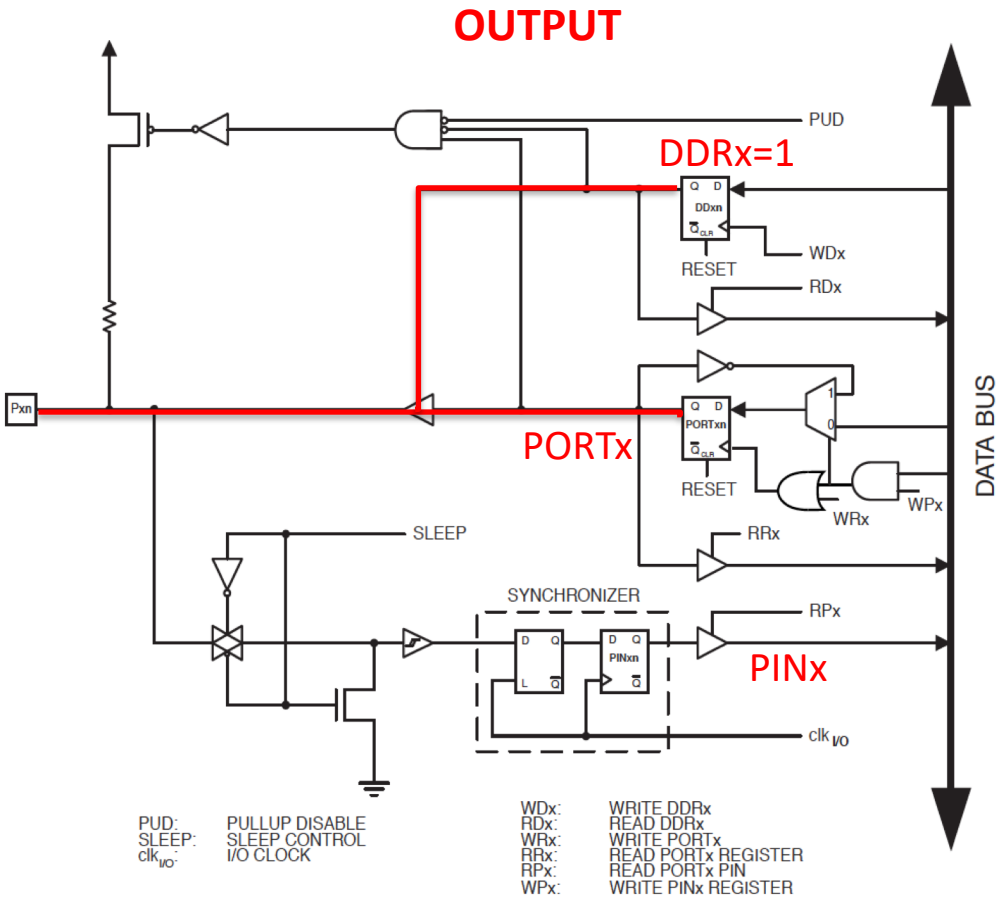
Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

Atmega328 Internals

PIN Circuitry

Interface to I/O – Output

- DDRx = 1
- PORTx = Pin output value



Note: 1. WRx, WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

Atmega328 Internals

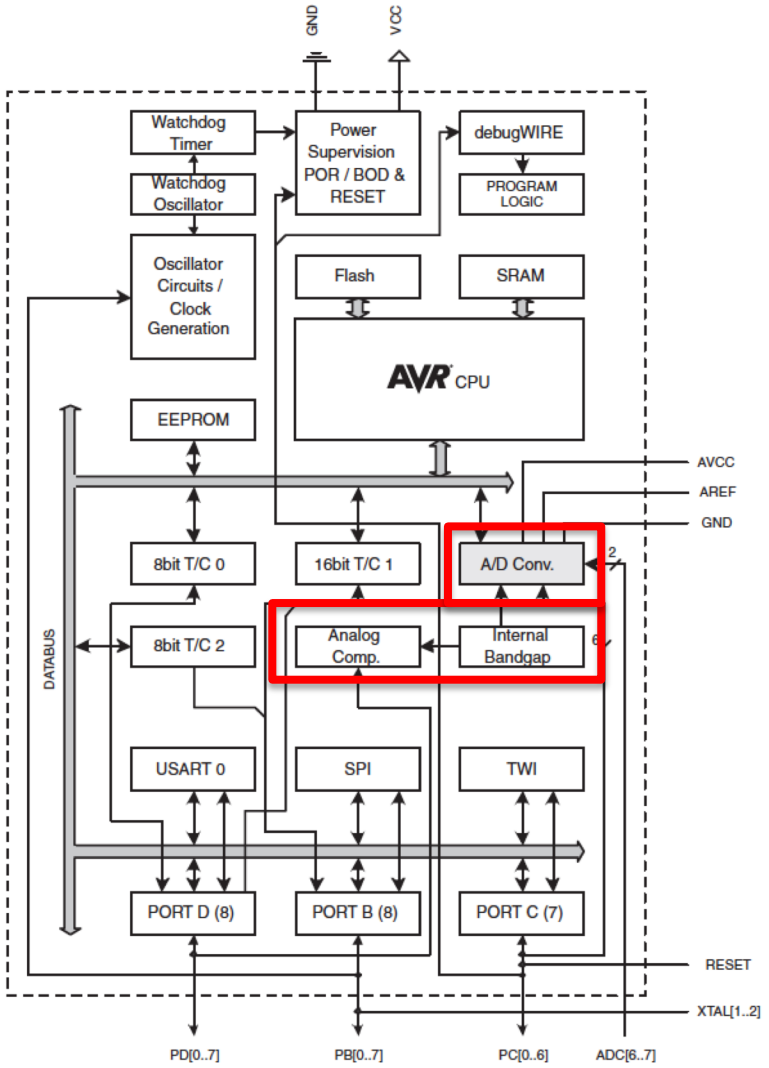
- Digital pins:
  - Pins 0 – 7: PORT D [0:7]
  - Pins 8 – 13: PORT B [0:5]
  - Pins 14 – 19: PORT C [0:5] (Arduino analog pins 0 – 5)
  - digital pins 0 and 1 are RX and TX for serial communication
  - digital pin 13 connected to the base board LED
- Digital Pin I/O Functions
  - pinMode(*pin*, *mode*)
    - Sets pin to INPUT or OUTPUT mode
    - Writes 1 bit in the DDRx register
  - digitalWrite(pin, value)
    - Sets pin value to LOW or HIGH (0 or 1)
    - Writes 1 bit in the PORTx register
  - int value = digitalRead(pin)
    - Reads back pin value (0 or 1)
    - Read 1 bit in the PINx register

Atmega328 Internals

AVR Architecture

Analog inputs

- Convert voltage to a 10-bit digital value
- Available reference value



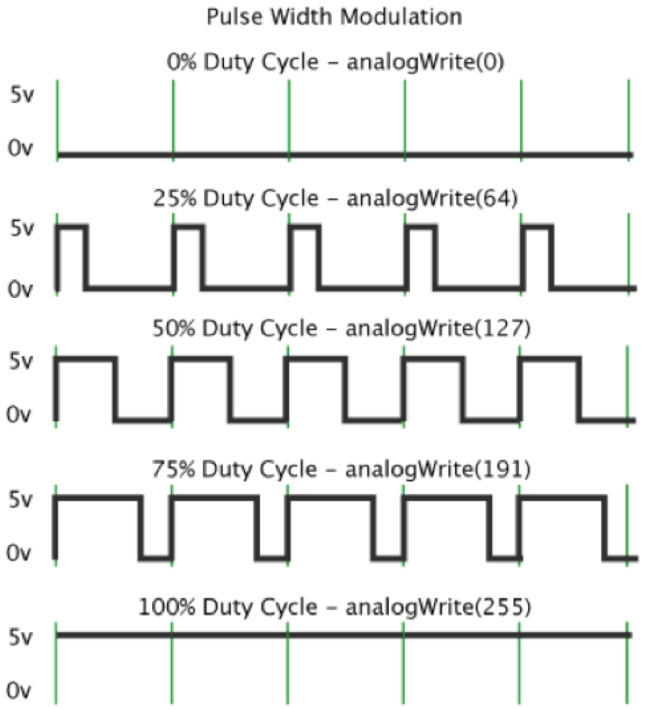
Atmega328 Internals

- Analog input pins: 0 – 5
- Analog output pins: 3, 5, 6, 9, 10, 11 (digital pins)
- Analog input functions
  - int val = analogRead(*pin*)
  - Converts 0 – 5v. voltage to a 10-bit number (0 – 1023)
  - Don't use pinMode
  - analogReference(*type*)
    - Used to change how voltage is converted (advanced)
- Analog output
  - analogWrite(*pin*, *value*)
    - value is 0 – 255
  - Generates a PWM output on digital pin (3, 5, 6, 9, 10, 11)
  - @490Hz frequency

Atmega328 Internals

Pulse Width Modulation

- Represents analog values by means digital outputs
- Clock with variable duty cycle that represents a value

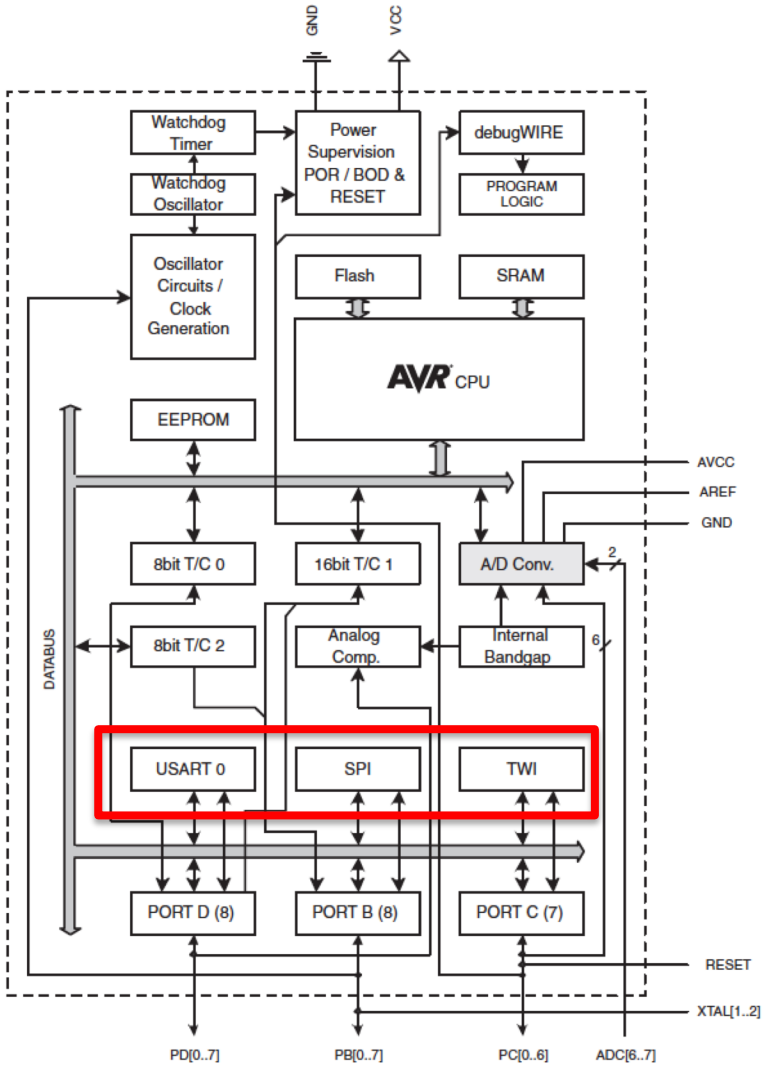


Atmega328 Internals

AVR Architecture

Special I/O support

- Serial protocols (SPI/I2C)
- Use special pins
- Requires the use of timers



Electronic concepts

Electronic concepts

Some theoretical concepts:

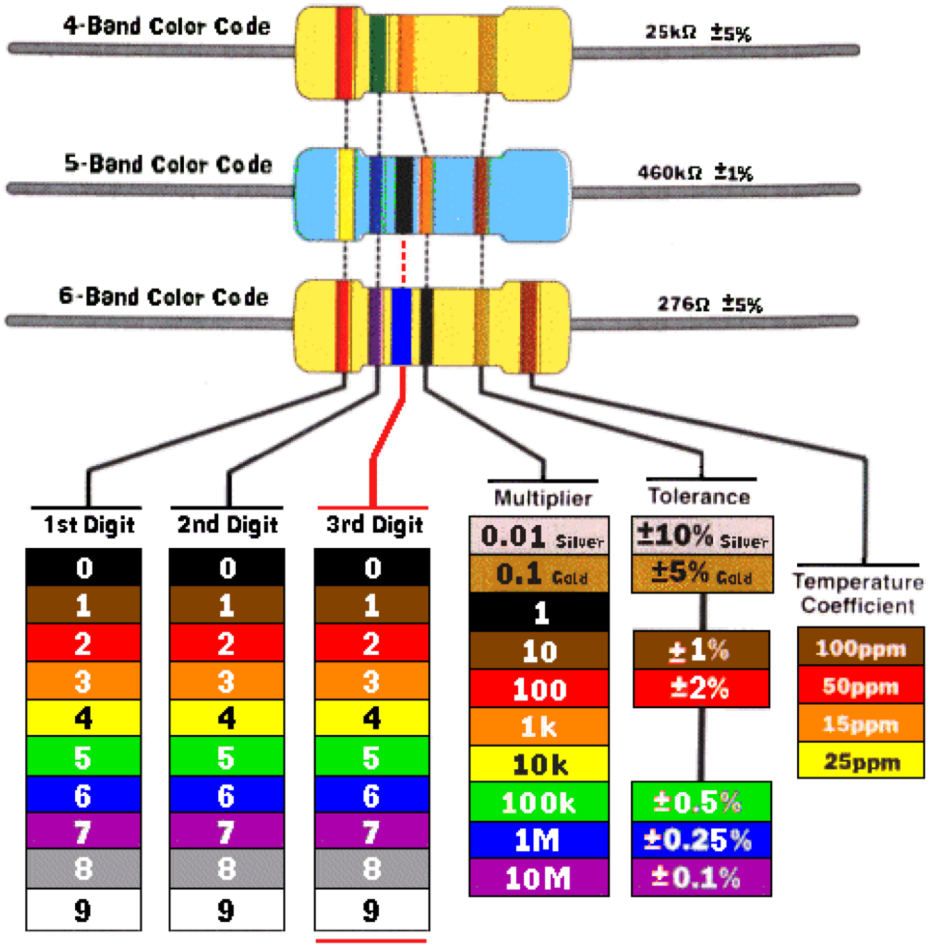
Voltage (V): Also known as electrical potential difference or electric tension. May represent either a source of energy.

Electric current (I): is a flow of electric charge through an electrical conductor when there is voltage.

Electric resistance (R): is the opposition to the passage of an electric current through that conductor

$$V = I \times R$$

Electronic concepts



Electronic concepts

Arduino pin voltage ➔ HIGH (5V) and LOW (0V) TTL levels

Arduino max. electric current ➔

- DC Current per I/O Pin - 40.0 mA
- DC Current VCC and GND Pins - 200.0 mA
- 1 VCC pin: Means these Arduinos can Source a total of 200mA
- 2 GND pins: Means these Arduinos can Sink a total of 400mA

Electronic concepts

Pin SOURCE Current Limitations:

NOTE: Although each I/O port can source more than the test conditions (20 mA at VCC = 5V, 10 mA at VCC = 3V) under steady state conditions (non-transient), the following must be observed.

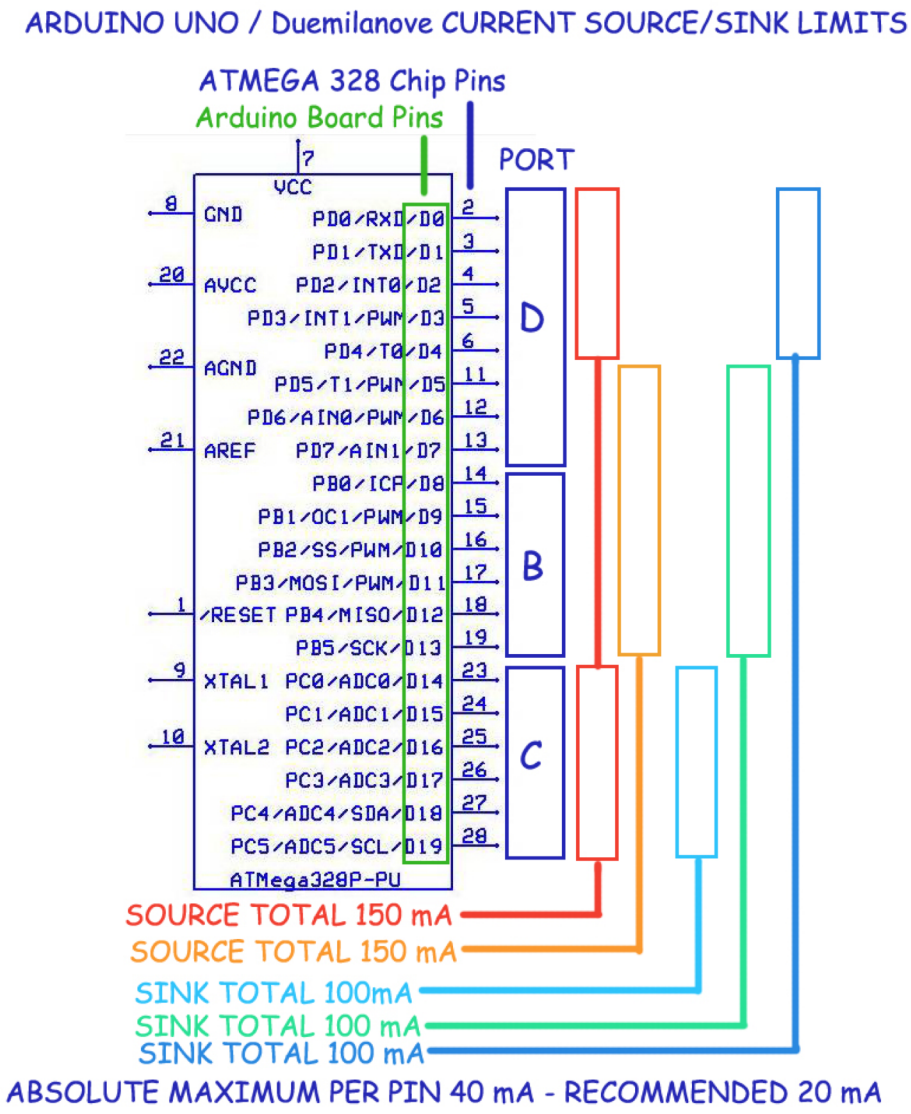
- The sum of all IOH, for ports C0 - C5, D0- D4, ADC7, RESET should not exceed 150 mA.
- The sum of all IOH, for ports B0 - B5, D5 - D7, ADC6, XTAL1, XTAL2 should not exceed 150 mA.
- If IOH exceeds the test condition, VOH **may exceed** the related specification. Pins are not guaranteed to source current greater than the listed test condition.

Electronic concepts

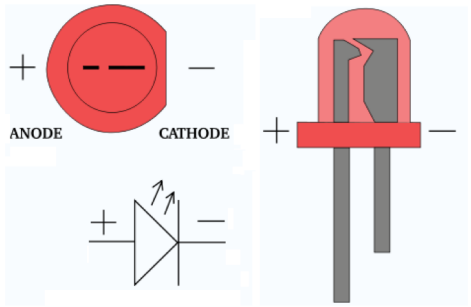
Pin SINK Current Limitations:

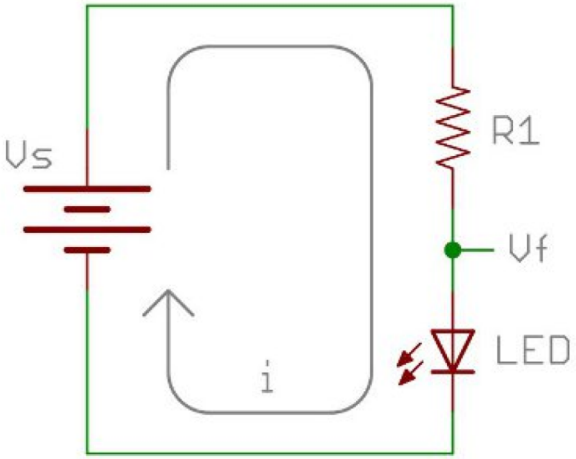
NOTE: Although each I/O port can sink more than the test conditions (20 mA at VCC = 5V, 10 mA at VCC = 3V) under steady state conditions (non-transient), the following must be observed:

- The sum of all IOL, for ports C0 - C5, ADC7, ADC6 should not exceed 100 mA.
- The sum of all IOL, for ports B0 - B5, D5 - D7, XTAL1, XTAL2 should not exceed 100 mA.
- The sum of all IOL, for ports D0 - D4, RESET should not exceed 100 mA.
- If IOL exceeds the test condition, VOL may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test condition.


ARDUINO UNO / Duemilanove CURRENT SOURCE/SINK LIMITS
ATMEGA 328 Chip Pins
Arduino Board Pins
PORT
7
VCC
8
GND
20
AVCC
22
AGND
21
AREF
1
/RESET
9
XTAL1
10
XTAL2
PD0/RXD/D0
PD1/TXD/D1
PD2/INT0/D2
PD3/INT1/PWM/D3
PD4/T0/D4
PD5/T1/PWM/D5
PD6/AIN0/PWM/D6
PD7/AIN1/D7
PB0/ICP/D8
PB1/OC1/PWM/D9
PB2/SS/PWM/D10
PB3/MOSI/PWM/D11
PB4/MISO/D12
PB5/SCK/D13
PC0/ADC0/D14
PC1/ADC1/D15
PC2/ADC2/D16
PC3/ADC3/D17
PC4/ADC4/SDA/D18
PC5/ADC5/SCL/D19
2
3
4
5
6
11
12
13
14
15
16
17
18
19
23
24
25
26
27
28
D
B
C
ATMega328P-PU
SOURCE TOTAL 150 mA
SOURCE TOTAL 150 mA
SINK TOTAL 100mA
SINK TOTAL 100 mA
SINK TOTAL 100 mA
ABSOLUTE MAXIMUM PER PIN 40 mA - RECOMMENDED 20 mA

Working with LEDs





$$R = \frac{Vs - Vf}{i}$$

LED Color	Potential Difference
Infrared	1.6 V
Red	1.8 V to 2.1 V
Orange	2.2 V
Yellow	2.4 V
Green	2.6 V
Blue	3.0 V to 3.5 V
White	3.0 V to 3.5 V
Ultraviolet	3.5 V

Using a Red LED

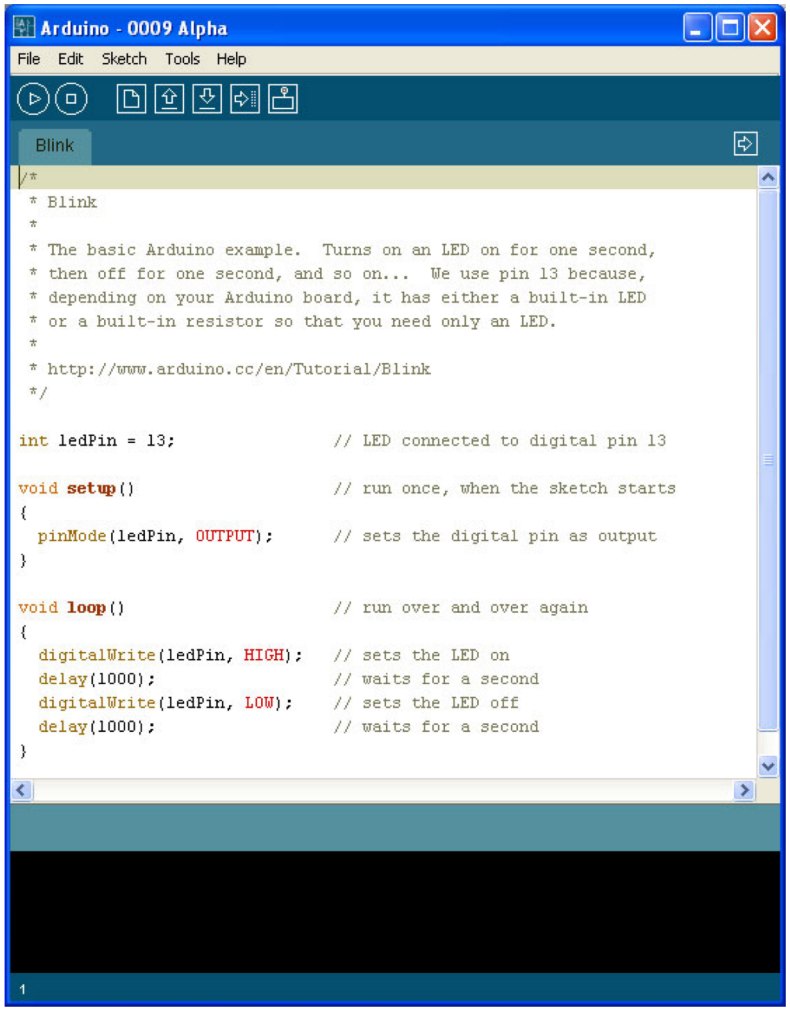
Vf = 2V aprox.

Vs = VCC = 5V

Current = 20mA

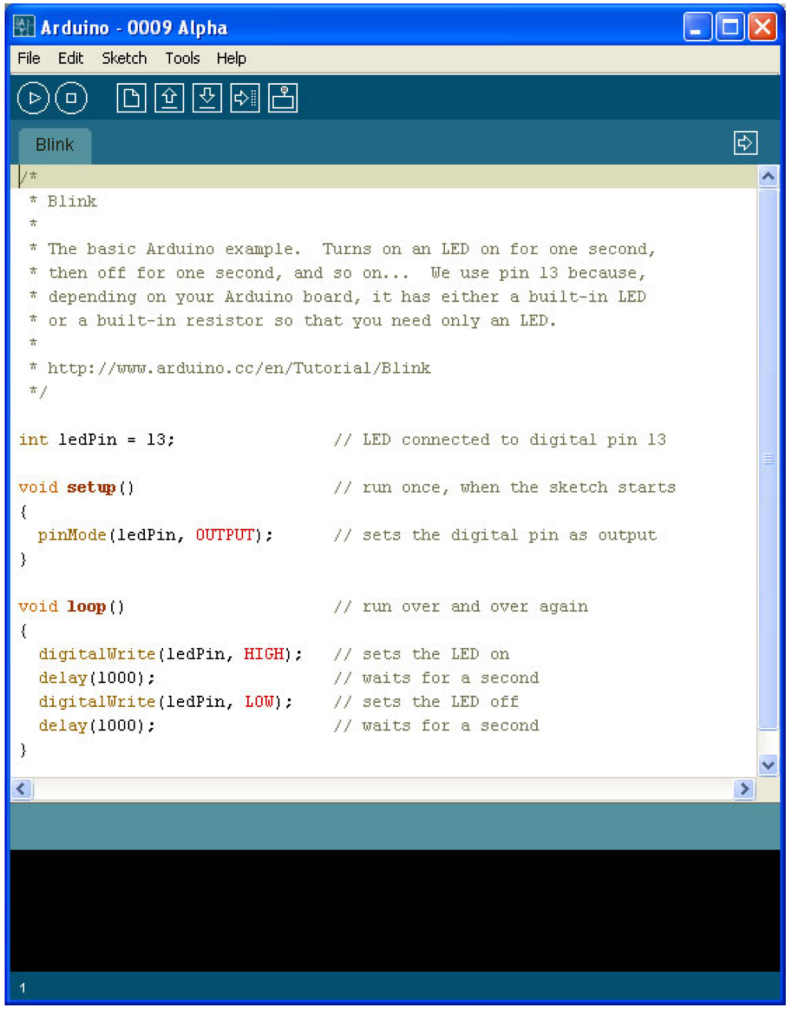
$$R = \frac{5V - 2V}{20mA} = 150\Omega$$

Arduino IDE Environment



Programming ARDUINO

- Programs are known as Sketchs
  Can be C or C++, but C provides much more efficient code.
- There is no debug option – Online monitoring by the serial monitor
- The sketch is composed by:
  - Setup procedure
    - Run once
    - Determines the initial parameters and configuration
  - Loop procedure
    - Main code to be executed continuously



http://www.ladyada.net/learn/arduino/index.html
http://arduino.cc/playground/

Programming ARDUINO

Data Types

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- float
- double
- string - char array
- String - object
- array

Constants

- HIGH | LOW
- INPUT | OUTPUT
- true | false
- integer constants
- floating point constants

Variable Scope & Qualifiers

- variable scope
- static
- volatile
- const

Programming ARDUINO

Control Structures

- if
- if...else
- for
- switch case
- while
- do... while
- break
- continue
- return
- goto

Conversion

- char()
- byte()
- int()
- word()
- long()
- float()

Arithmetic Operators

- = (assignment operator)
- + (addition)
- - (subtraction)
- * (multiplication)
- / (division)
- % (modulo)

Comparison Operators

- == (equal to)
- != (not equal to)
- < (less than)
- > (greater than)
- <= (less than or equal to)
- >= (greater than or equal to)

Programming ARDUINO

Digital I/O
- pinMode()
- digitalWrite()
- digitalRead()

Analog I/O
- analogReference()
- analogRead()
- analogWrite() - *PWM*

Advanced I/O
- tone()
- noTone()
- shiftOut()
- shiftIn()
- pulseIn()

Time
- millis()
- micros()
- delay()
- delayMicroseconds()

Math
- min()
- max()
- abs()
- constrain()
- map()
- pow()
- sqrt()

Trigonometry
- sin()
- cos()
- tan()

Random Numbers
- randomSeed()
- random()

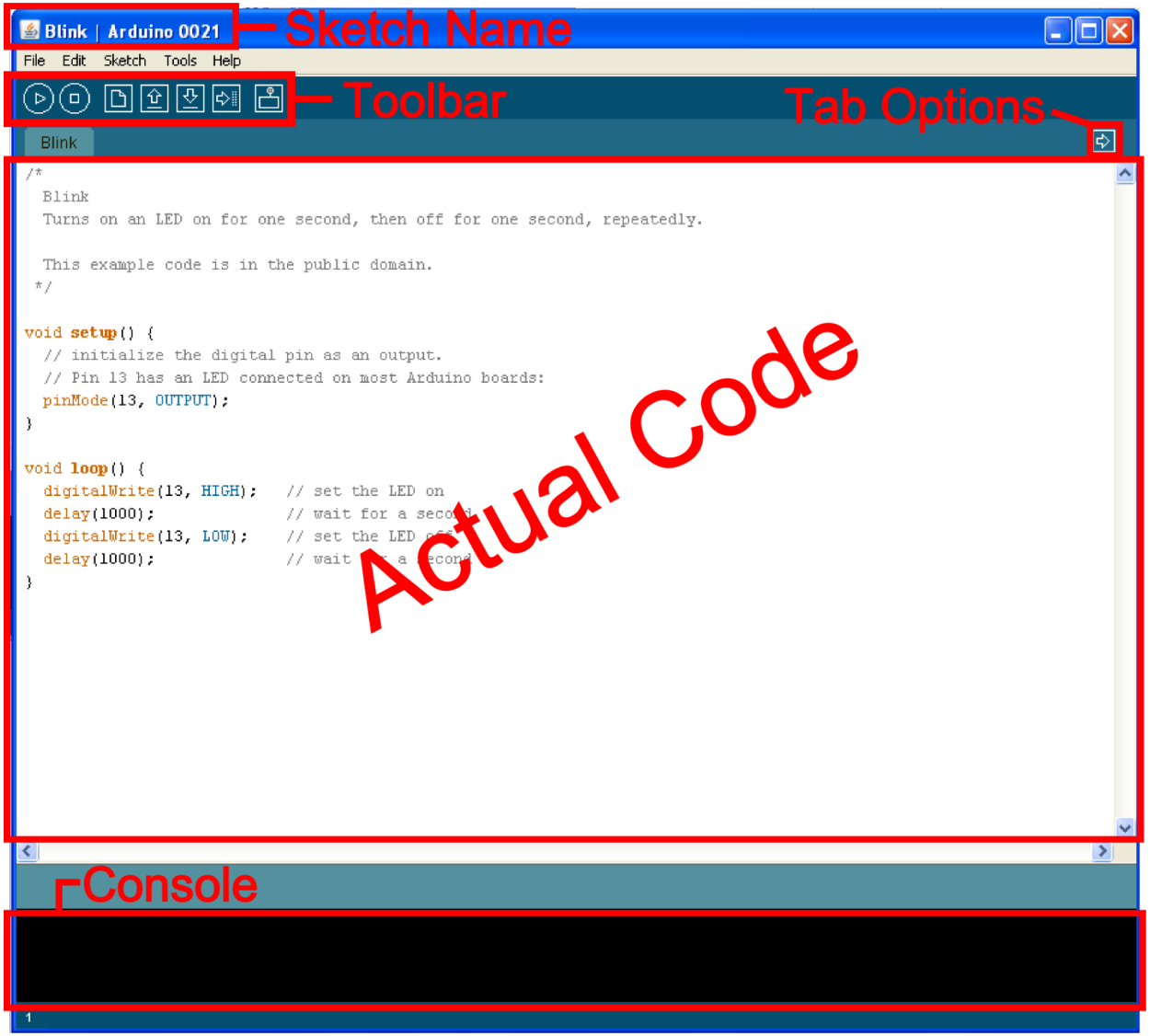
Bits and Bytes
- lowByte()
- highByte()
- bitRead()
- bitWrite()
- bitSet()
- bitClear()
- bit()

External Interrupts
- attachInterrupt()
- detachInterrupt()

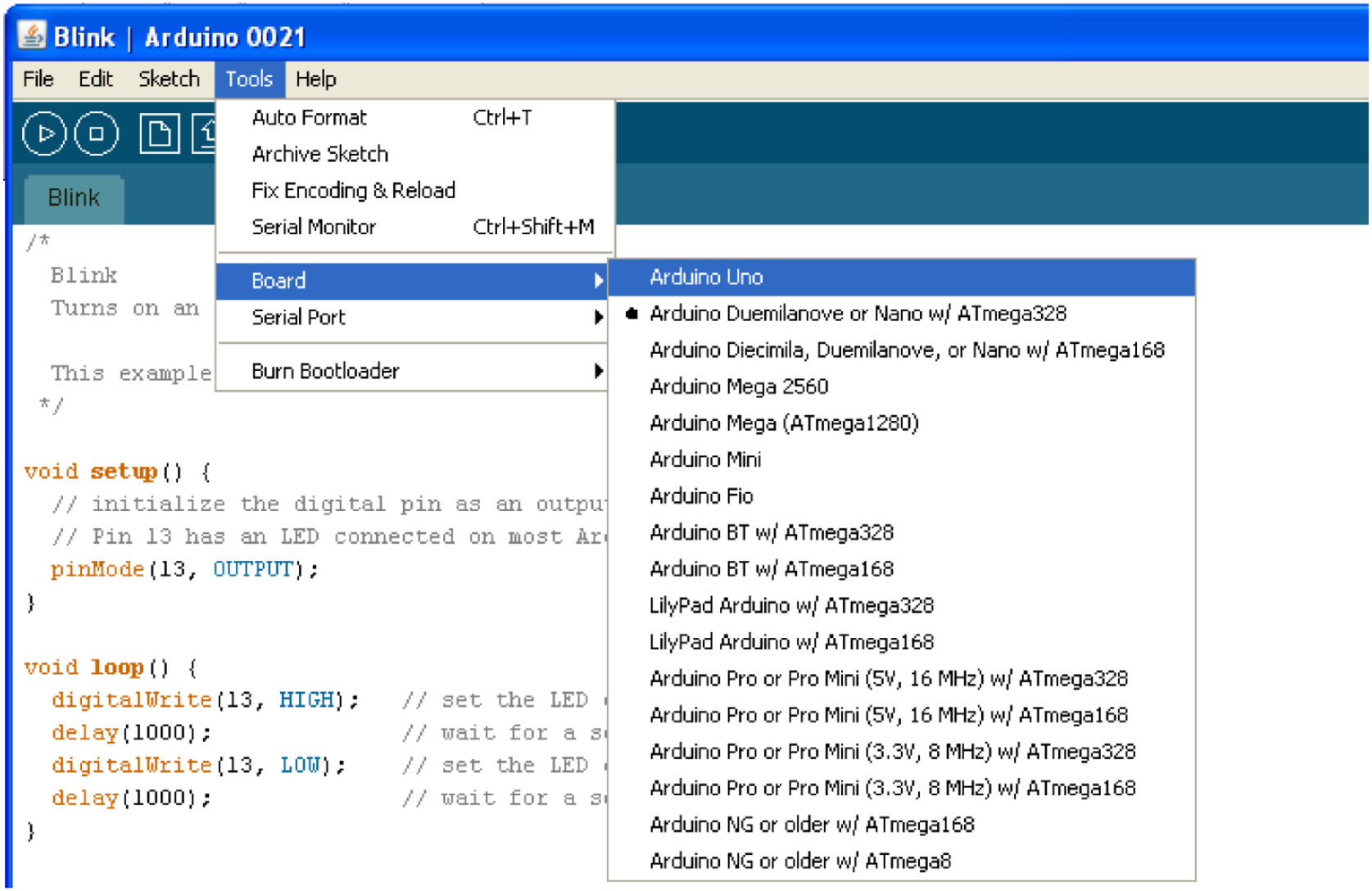
Interrupts
- interrupts()
- noInterrupts()

Communication
- Serial
- Stream

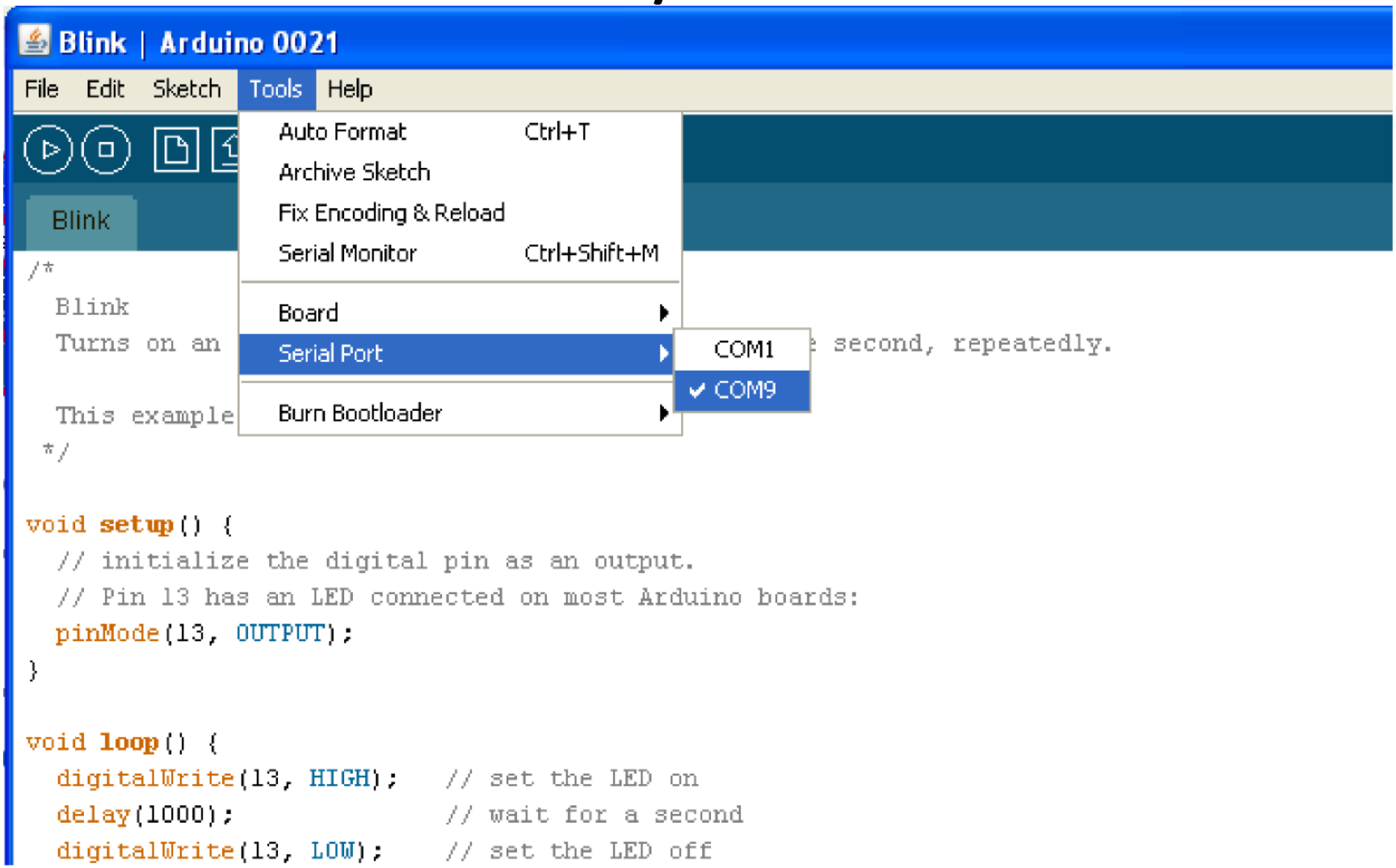
http://arduino.cc/en/Reference/HomePage
http://arduino.cc/en/Reference/Libraries


Blink | Arduino 0021
Sketch Name
File Edit Sketch Tools Help
Toolbar
Tab Options
Blink
/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.
This example code is in the public domain.
*/
void setup() {
// initialize the digital pin as an output.
// Pin 13 has an LED connected on most Arduino boards:
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH); // set the LED on
delay(1000); // wait for a second
digitalWrite(13, LOW); // set the LED off
delay(1000); // wait for a second
}
Actual Code
Console
1

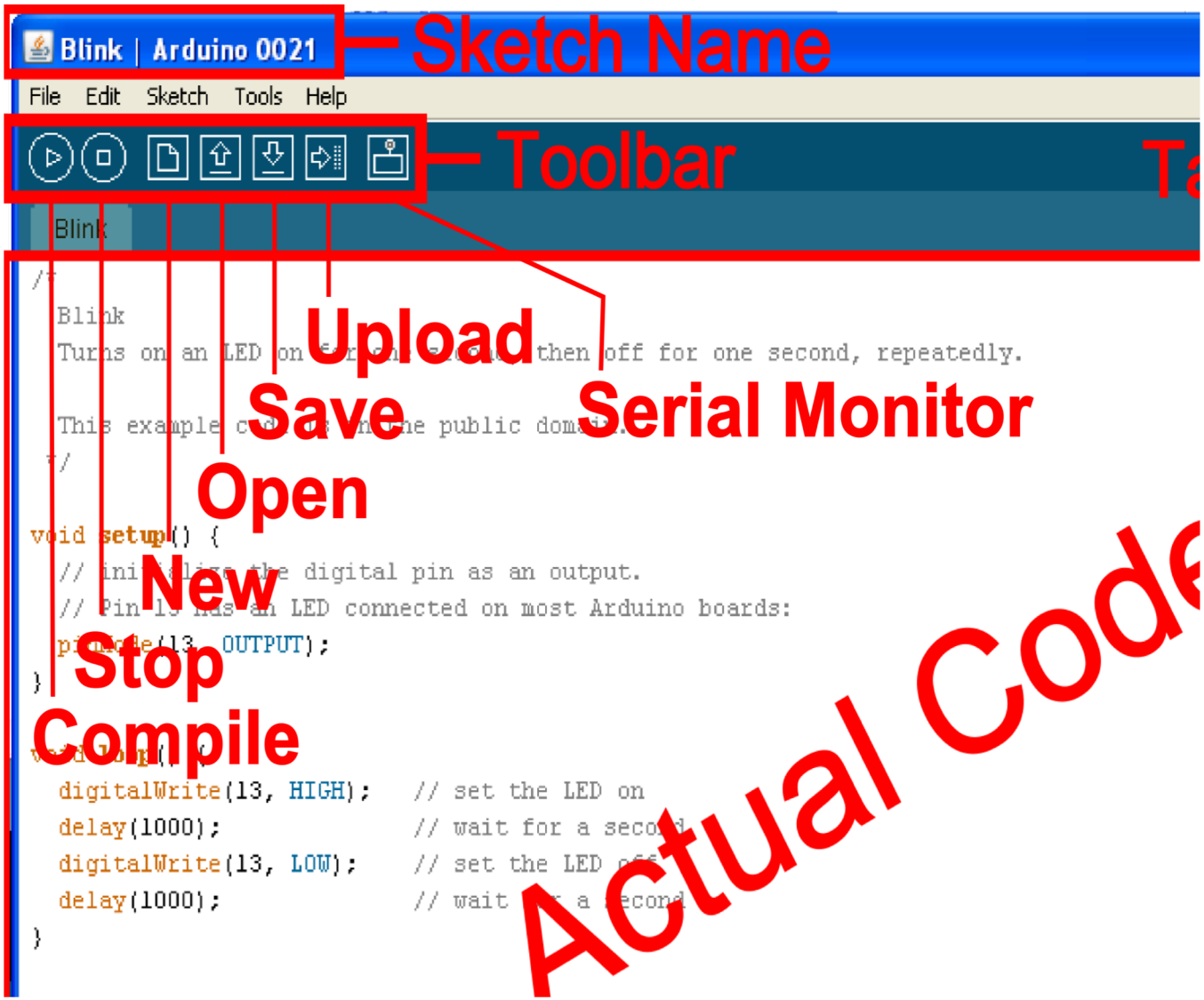
Board Type


Blink | Arduino 0021
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Ctrl+Shift+M
Board
Serial Port
Burn Bootloader
Arduino Uno
Arduino Duemilanove or Nano w/ ATmega328
Arduino Diecimila, Duemilanove, or Nano w/ ATmega168
Arduino Mega 2560
Arduino Mega (ATmega1280)
Arduino Mini
Arduino Fio
Arduino BT w/ ATmega328
Arduino BT w/ ATmega168
LilyPad Arduino w/ ATmega328
LilyPad Arduino w/ ATmega168
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega328
Arduino Pro or Pro Mini (5V, 16 MHz) w/ ATmega168
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega328
Arduino Pro or Pro Mini (3.3V, 8 MHz) w/ ATmega168
Arduino NG or older w/ ATmega168
Arduino NG or older w/ ATmega8
Blink
/*
Blink
Turns on an
This example
*/
void setup() {
// initialize the digital pin as an output
// Pin 13 has an LED connected on most Ar
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH); // set the LED
delay(1000); // wait for a s
digitalWrite(13, LOW); // set the LED
delay(1000); // wait for a s
}

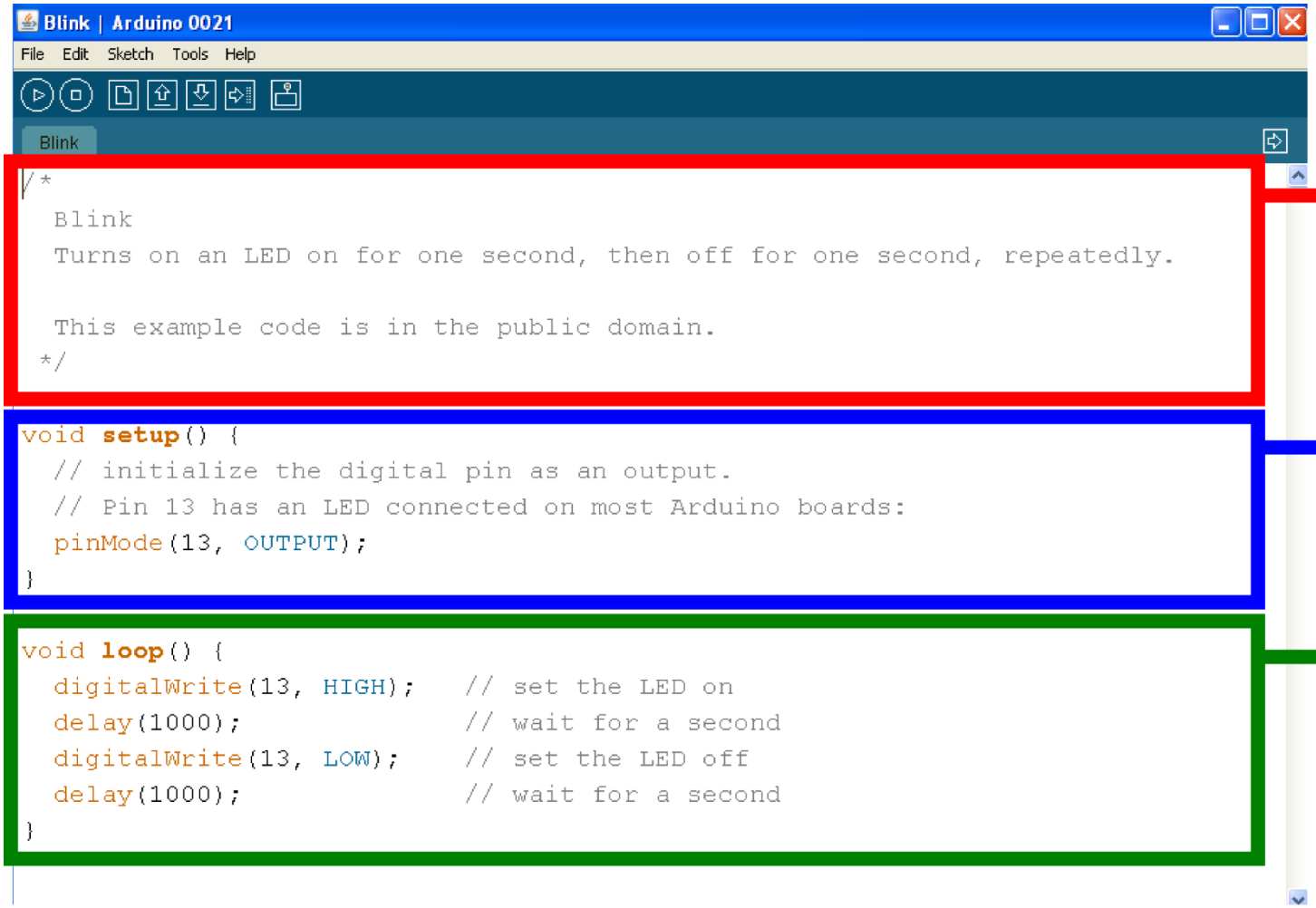
Serial Port / COM Port



The Environment



Parts of the Sketch



Comments / Explaining the game

Setup / Stretching or tying shoes

Loop / Playing the game

Setup

void setup ( ) { }

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

The setup function comes before the loop function and is necessary for all Arduino sketches

Setup

void setup ( ) { }

```
void setup() {
  // Initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

The setup header will never change, everything else that occurs in setup happens inside the curly brackets

Setup

void setup ( ) {

pinMode (13, OUTPUT); }

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}
```

Outputs are declare in setup, this is done by using the pinMode function

This particular example declares digital pin # 13 as an output, remember to use CAPS

Setup

void setup ( ) { Serial.begin;}

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  Serial.begin(9600);
}
```

Serial communication also begins in setup

This particular example declares Serial communication at a baud rate of 9600. More on Serial later...

Setup

void setup ( ) {digitalWrite (12, HIGH); }

```
void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
  Serial.begin(9600);
  digitalWrite(12, HIGH);
}
```

Activates de output line – HIGH/LOW

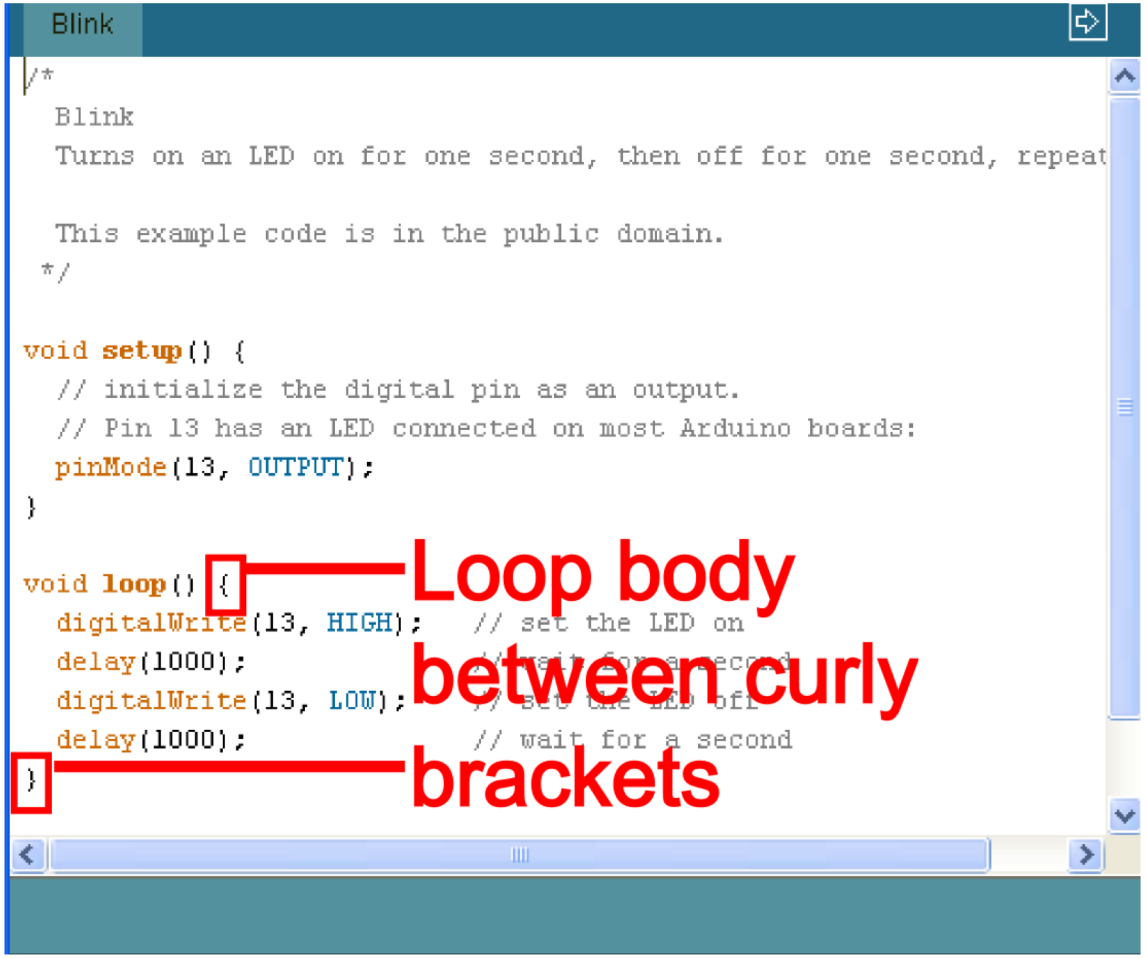
void loop ( ) { }

Blink

```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeat

  This example code is in the public domain.
 */

void setup() {
  // initialize the digital pin as an output.
  // Pin 13 has an LED connected on most Arduino boards:
  pinMode(13, OUTPUT);
}

void loop() {
  digitalWrite(13, HIGH);   // set the LED on
  delay(1000);              // wait for a second
  digitalWrite(13, LOW);    // set the LED off
  delay(1000);              // wait for a second
}
```

Loop header

void loop ( ) { }


Blink
/*
Blink
Turns on an LED on for one second, then off for one second, repeat
This example code is in the public domain.
*/
void setup() {
// initialize the digital pin as an output.
// Pin 13 has an LED connected on most Arduino boards:
pinMode(13, OUTPUT);
}
void loop() {
digitalWrite(13, HIGH); // set the LED on
delay(1000);
digitalWrite(13, LOW);
delay(1000); // wait for a second
}
Loop body
between curly
brackets

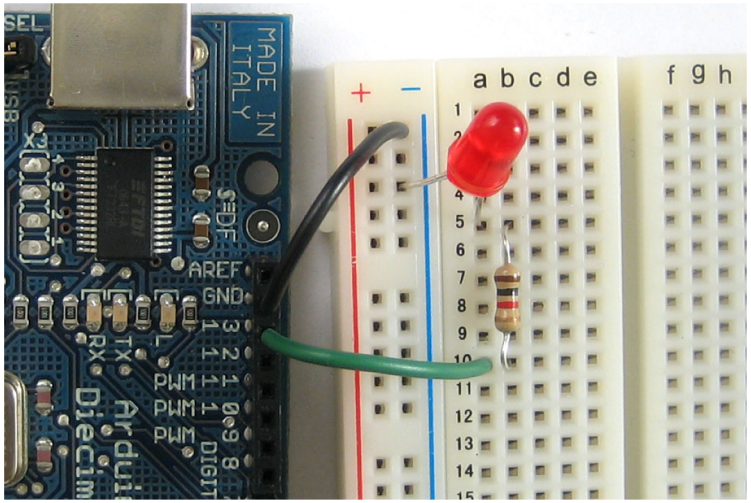
To compile your sketch, click the checkmark.

Make sure your Arduino is plugged into an available USB port.

Click the arrow to download the program to Arduino. If everything is attached correctly. The LED should blink.

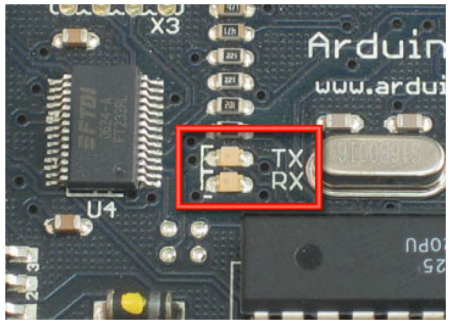
Basic example – Led blink

```
int ledPin = 13;            // LED connected to digital pin 13

void setup()
{
  pinMode(ledPin, OUTPUT);  // sets the digital pin as output
}

void loop()
{
  digitalWrite(ledPin, HIGH);// sets the LED on
  delay(1000);               // waits for a second
  digitalWrite(ledPin, LOW); // sets the LED off
  delay(1000);               // waits for a second
}
```



Serial Port communication

- Serial Port allows communicate Arduino with other devices
- It has associated two digital pins – D0 (RX), D1 (TX)
- These pins are also connected to the USB controller and they have associated two leds
- The Serial Monitor allows to monitor the send and received data



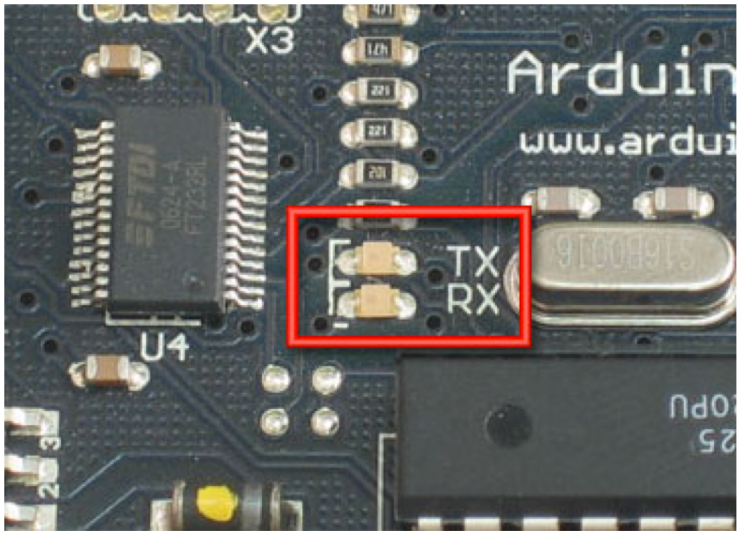
- **Programming the Serial port (**Serial object**)**
  - Main actions: Initialize, send, receive

    - *Initialize* → Serial.begin(baudrate) ➔ (9600bps, .., 115200bps)

    - *Send* → Serial.print(string)
      Serial.println(String)
      Serial.write(byte)

    - *Receive* → Serial.read(byte/char)
      Serial.readbytes(buffer, length)
      Serial.readBytesUntil(character, buffer, length)

Serial port communication



If (Serial)
available()
availableForWrite()
begin()
end()
find()
findUntil()
flush()
parseFloat()
parseInt()
peek()
print()
println()
read()
readBytes()
readBytesUntil()
readString()
readStringUntil()
setTimeout()
write()
serialEvent()

https://www.arduino.cc/reference/en/language/functions/communication/serial/

Serial port communication - Example

Three possible characters to be sent to the Arduino

- B – Makes the led blink
- S – Makes the led stop blinking
- A – Ask to the Arduino what is the led state – returns
  - B for blinking
  - S for stopped

Serial port communication – Activity

Guess the number - The Arduino-PC interaction will be done through the Serial Monitor.

- The Arduino generates a random number, in the range [1..1000], which the user must guess.
- The user enters and sends a number to the Arduino.
- The Arduino determines if the received value is higher or lower than the hidden one, then it returns a message indicating the current situation.
- The game continues until the user discovers the number.
- At the end, the Arduino will show the number of attempts needed and will ask to start a new game.

The program will include some user help functionality. Thus, when the user sends the character ‘h’ instead a number, the game will return a random clue as :

- "The number is odd”
- "The number has the digit X”
- etc....

Try to be imaginative with theses clues.

The Software-Serial port

- The Serial port is directly connected to the USB controller and it is used for accessing to the Arduino from the computer.
- There are devices that are based on the Serial protocol (Xbee, Serial LCD, …).
- The SoftwareSerial library allows to create a new Serial port mapped to digital lines with the same functionality than the Serial port.

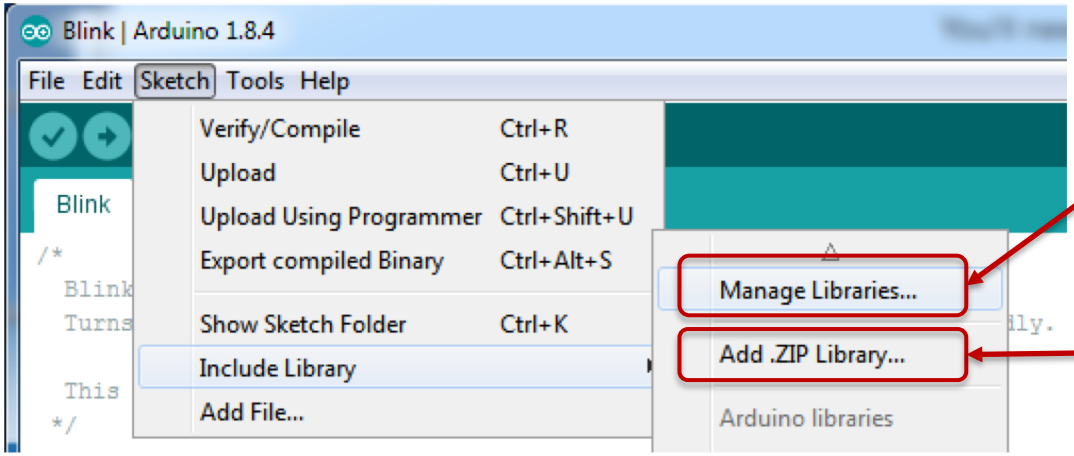
Example:

```
#include “SoftwareSerial.h”
SoftwareSerial sof_port(rx_pin, tx_pin); // Port declaration

void setup(){
  soft_port.begin(9600);                 // Port initialization
}
void loop(){
  soft_port.print(“Hello world\n”);      // working with the port
  ……
}
```

Arduino Libraries

Even it is possible to develop our own application to work wit any sensor/actuator Arduino has a large developer community.

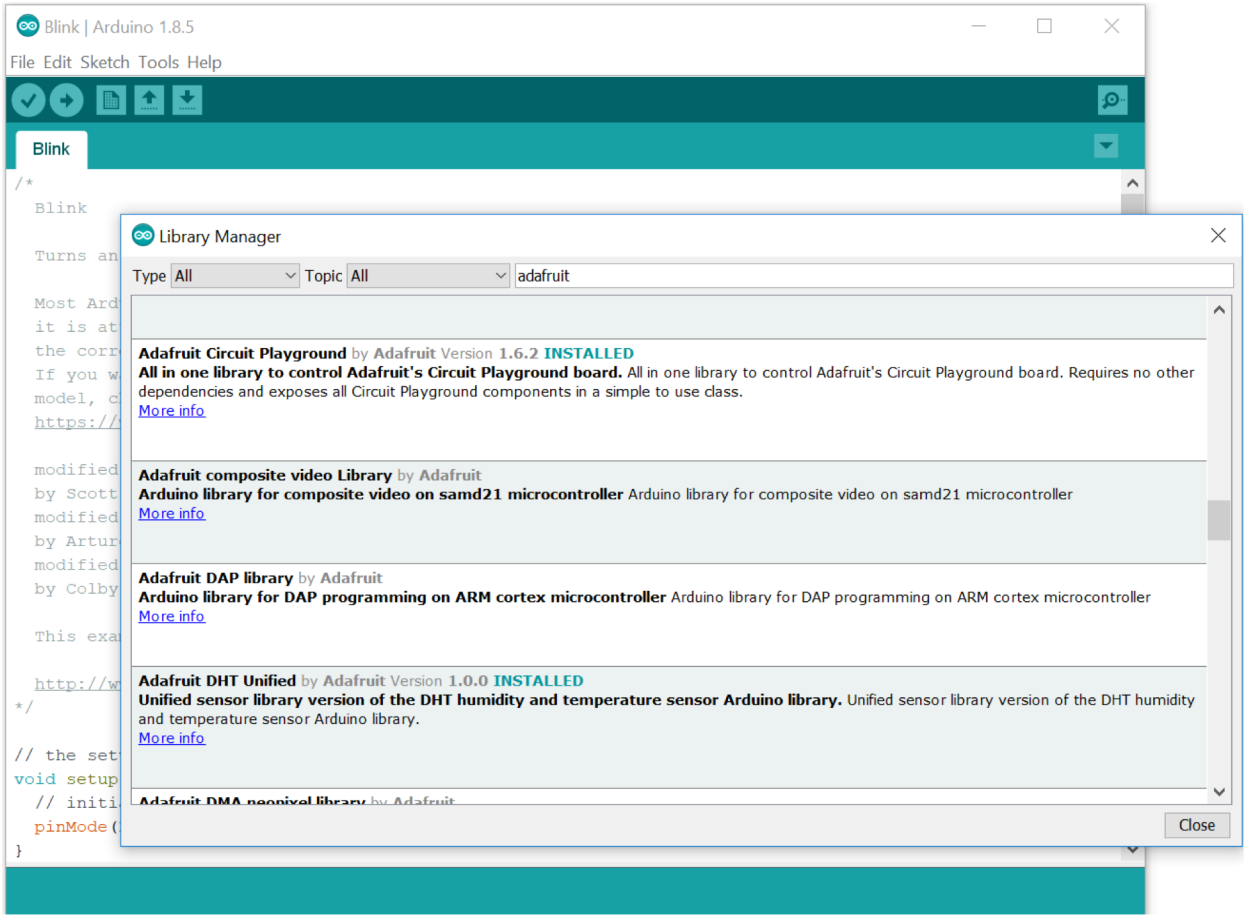
- Libraries simplify the development process
- There are many libraries for each sensor from different developers.
- Libraries use to include examples.



Access to the library repository

Install any local library

Arduino Libraries



Library Manager – Direct access to the respository

Arduino Libraries – Creating a new one

- `Header File` (.h) - Library's definitions
- `Source Code` (.cpp/c) - Library's code
- `Keyword file` (.txt) - Colour map for the keywords used in the library
- `Readme file` (.txt) - Information about the library (Revision Number, TODO's, etc.)
- `Examples` (.ino) - How to operate the library, inside the directory *examples*

All of them within the \Arduino\libraries\LIBRARY_NAME directory of your computer.

```
//Example_Library
//Header file

#ifndef PRLib_H
#define PRLib_H
  #include "Arduino.h"
  class PRLib{
    public:
        PRLib();
        begin(int led);
        led_on();
        led_off();
    private:
        int _led_pin;
  };
#endif
```

Header File – PRLib.h

```
//Example_Library
//Source file

#include "Arduino.h"
#include "PRLib.h"

PRLib::PRLib(){};

PRLib::begin(int led){
    pinMode(led,OUTPUT);
    _led_pin = led;
};

PRLib::led_on(){
    digitalWrite(_led_pin,HIGH);
};

PRLib::led_off(){
    digitalWrite(_led_pin,LOW);
};
```

Source File – PRLib.cpp

```
PRLib   KEYWORD1
begin   KEYWORD2
led_on  KEYWORD2
led_off KEYWORD2
```

Keywords.txt

1 – Orange
2 – Brown

Library – Activity

Create an Arduino library to control de Serial LCD* screen.

It must contain the following methods:

- **begin(**SoftwareSerial *sf_serial_port**)** > Defines the virtual port to use with
- **print**(int x, int y, char *str) → Prints the str string at the (x,y) location
- **sPixel** (int x, int y) → Set the pixel at coordinate (x,y)
- **clPixel** (int x, int y) → Clear the pixel at coordinate (x,y)
- **line**(int x1, int y1, int x2, int y2, int ink) → Draws a line from (x1,y1) to (x2,y2) with ink value → 0 clear, 1 set
- **rectangle** (int x, int y, int length, int height, int ink) → Draw a rectangle with ink value with ink value → 0 clear, 1 set
- **circle** (int x, int y, int radius) → Draw a circle with center at (x,y) and radius

Try to use the Serial LCD to show the "**Guess the number**" messages including some graphical features 😉.

* The datasheet of the ADM12864h LCD is in the virtual campus.