The AsTeRICS Academy

for cross-cultural education in Assistive Technology

AsTeRICS academy

MA23 Wirtschaft, Arbeit Statistik

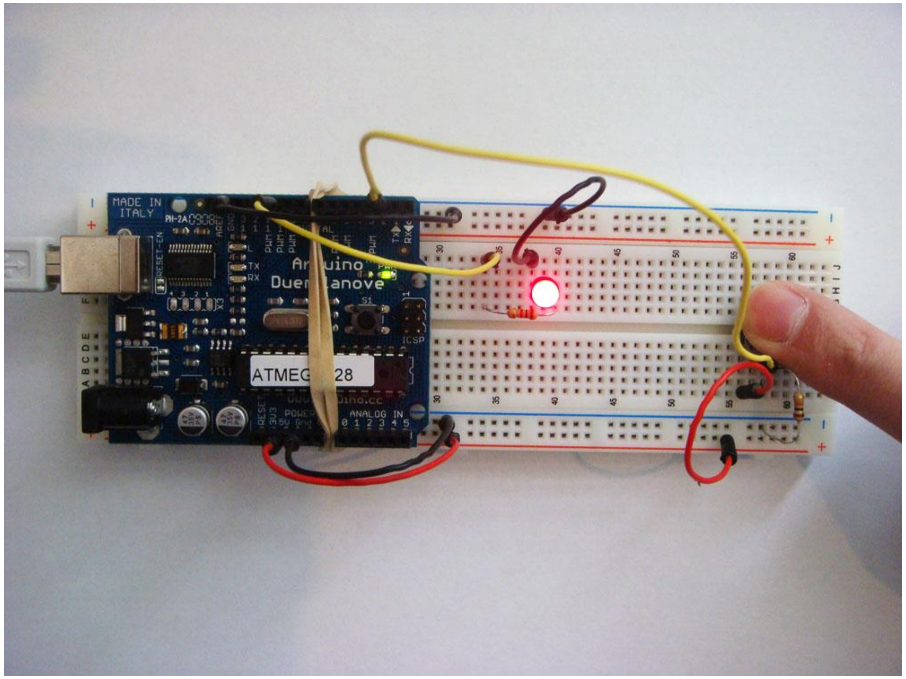
Stadt Wien

FH University of Applied Sciences TECHNIKUM WIEN

Module 4: Microcontrollers

using the Arduino and the Flexible Assistive Button Interface (FABI)

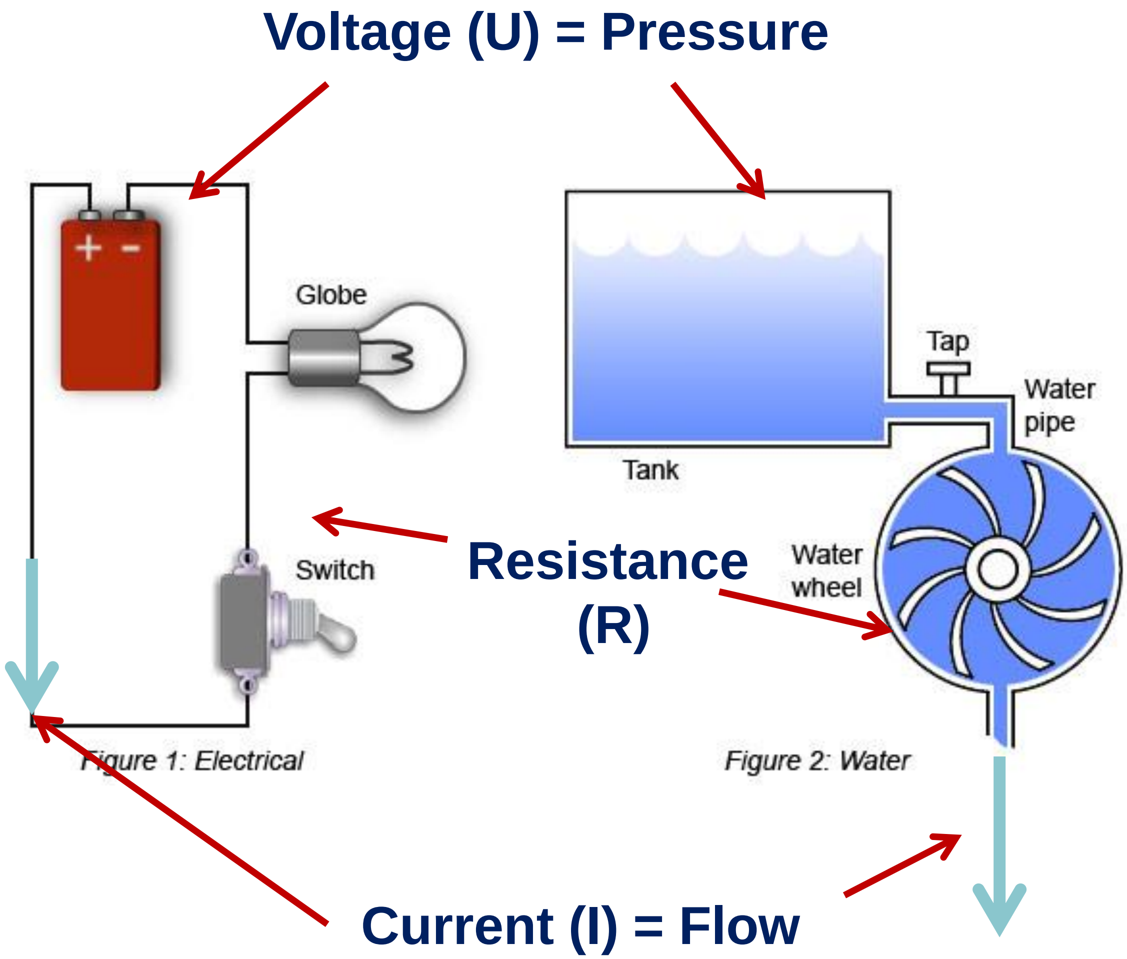
First steps in microcontroller programming !



… with applications in Assistive Technology and bioelectric signal processing ….

Basic concepts

Voltage and current… and their water-„analogies“



Voltage (U) = Pressure

Resistance (R)

Current (I) = Flow

Figure 1: Electrical

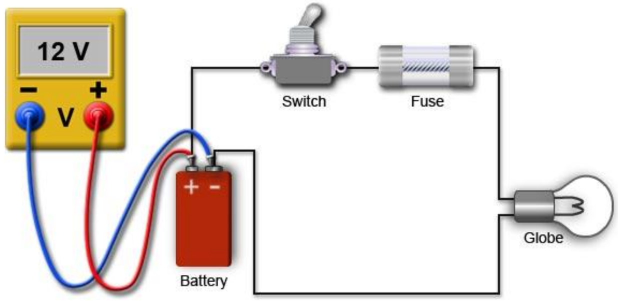
Figure 2: Water

Units:

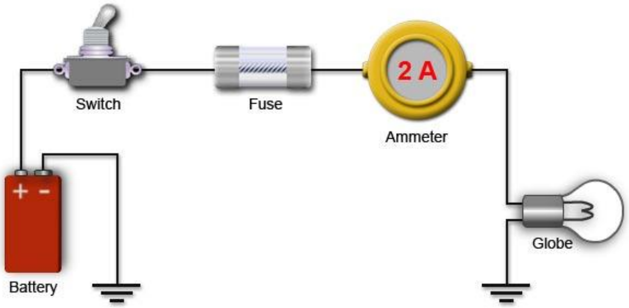
[I] = Ampere (A, mA)
[U] = Volt (V)
[R] = Ohm

Ohm‘s Law: $I = \frac{U}{R}$

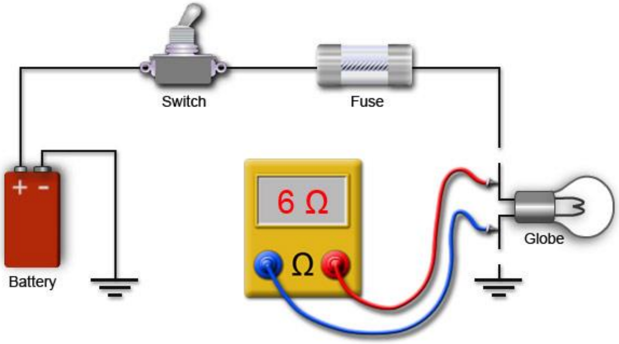
Using a Multimeter to measure U, I, R



measure voltage



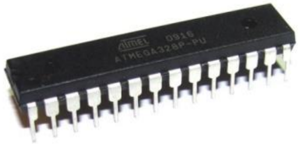
measure current



measure resistance

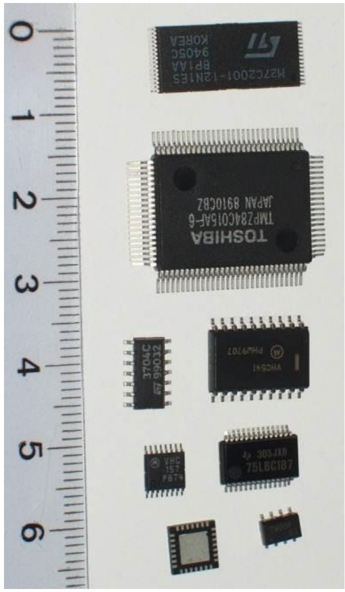
http://www.ladyada.net/learn/metertut/

What is a microcontroller?



- Microcomputer
  - Microprocessor (CPU) +memory
  - external peripheral devices (monitor, printer, user interaction devices)

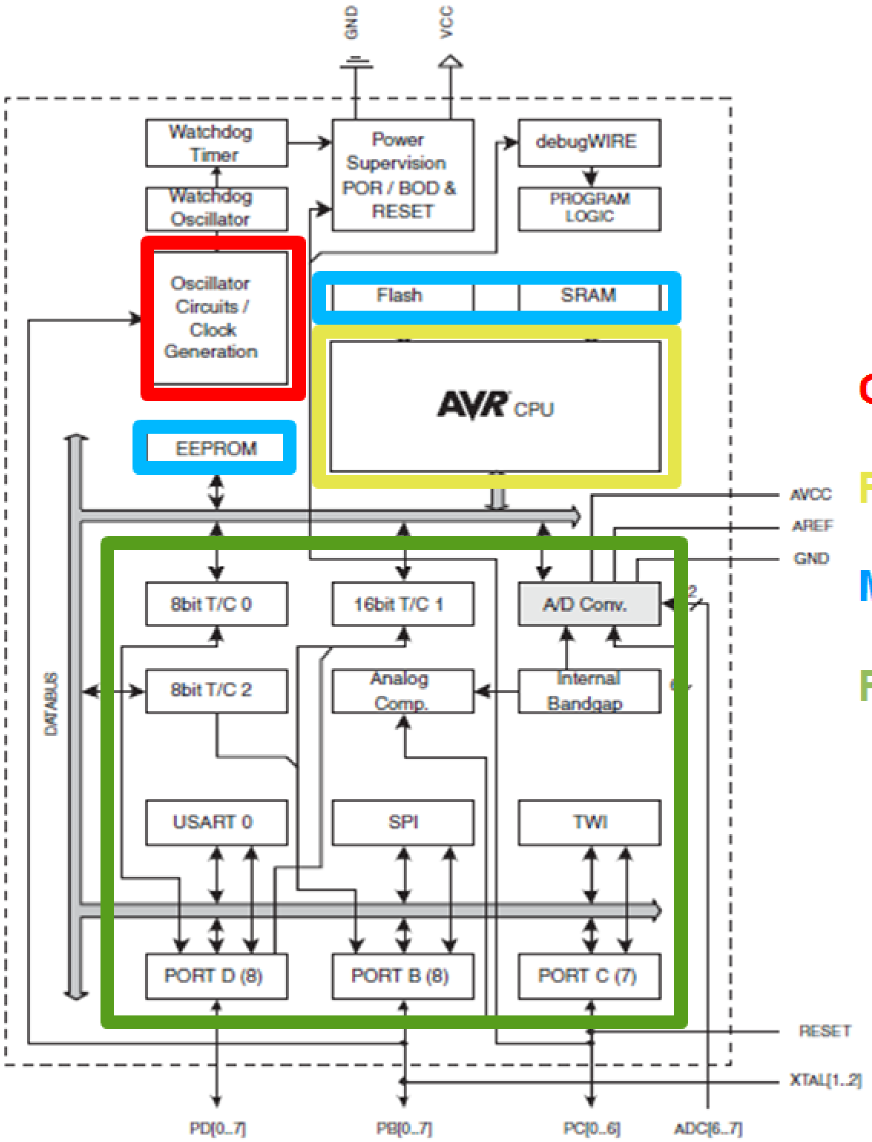
- Mikrocontroller (Microcontroller Unit, MCU)
  - „Small Microcomputer on a chip“
  - CPU + memory + internal peripheral devices
    for interfacing of sensors/actuators/ other modules
  - Examples: Intel 805, Microchip „PIC“, Atmel „AVR“, ARM „Cortex“
- Digital Signal Processor (DSP)
  - Optimised for fast implementation of digital filters etc.
    (MAC instruction = Multiply/Accumulate)
  - Bsp: AD „Blackfin“, TI C6000
- (Programmable) System-On-Chip, (SoC, pSoC)
  - Various flexible HW-blocks (graphical/sound controller, encryption, …



What is a microcontroller?

Example:

AVR MCU block diagram



CLOCK

PROCESSOR

MEMORY

PERIPHERALS

Microcontrollers …

… enable embedded systems that:

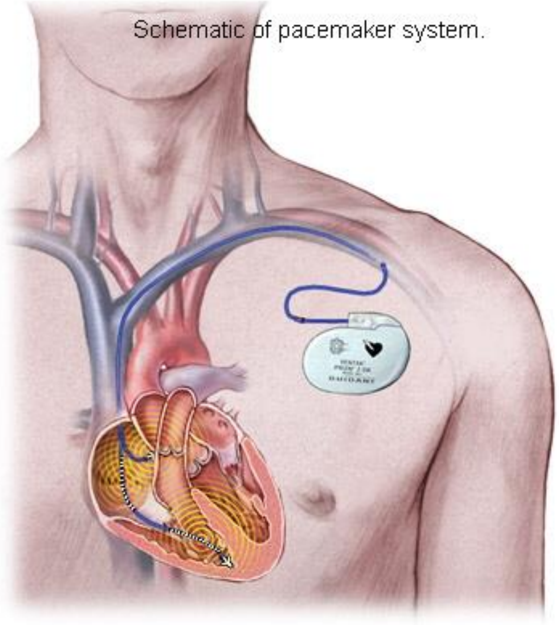
- are reliable
- are small and lightweight
- have a low power consumption
- allow reaction to events in a defined time (real-time)

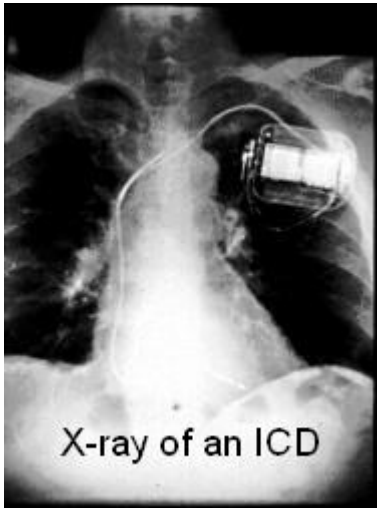
These issues are critical e.g. in body implants !

Examples for uC-based devices

Pace Makers and Functional Electro-Stimulation

http://www.hgcardio.com/HRhythm/Treatments/a_pacemaker_schematic.jpg





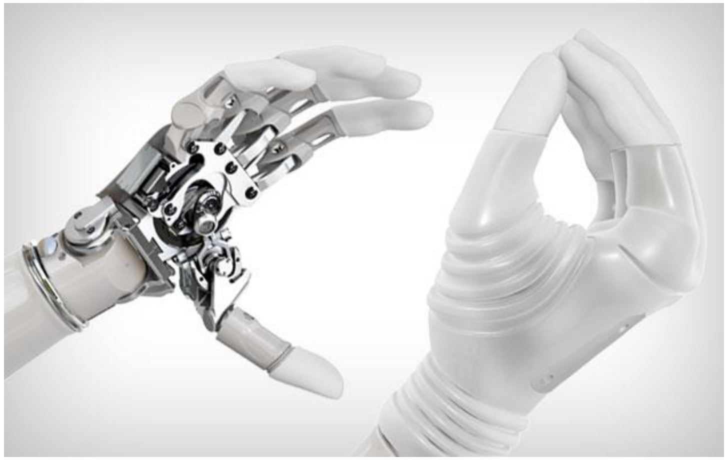
- current pacemakers have 5-7 yrs. battery lifetime
- feedback loops -> adapt to physical needs
- multichannel stimulation and measurement electrodes

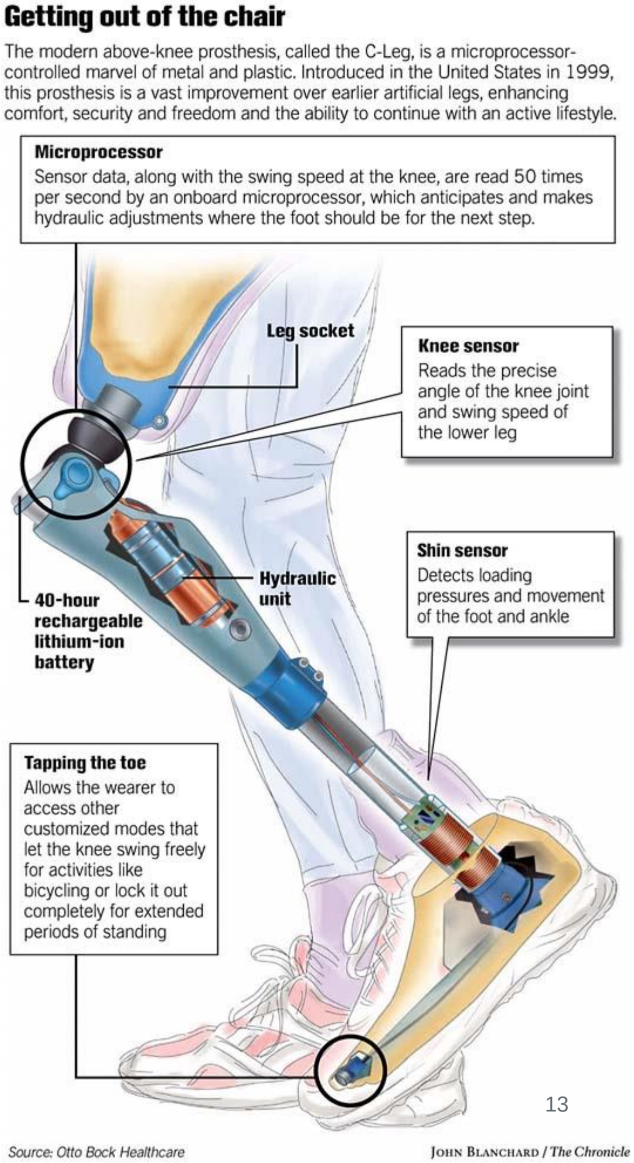
Examples for uC-based devices

- Alternative computer input devices
- Implants and FES-devices
- Active prosthesis, Orthesis and artificial limbs
- Environmental control systems
- Braille displays

Examples for uC-based devices

Otto Bock “C-Leg” and “Michelangelo Hand”





Getting out of the chair

The modern above-knee prosthesis, called the C-Leg, is a microprocessor-controlled marvel of metal and plastic. Introduced in the United States in 1999, this prosthesis is a vast improvement over earlier artificial legs, enhancing comfort, security and freedom and the ability to continue with an active lifestyle.

Microprocessor
Sensor data, along with the swing speed at the knee, are read 50 times per second by an onboard microprocessor, which anticipates and makes hydraulic adjustments where the foot should be for the next step.

Knee sensor
Reads the precise angle of the knee joint and swing speed of the lower leg

Shin sensor
Detects loading pressures and movement of the foot and ankle

Tapping the toe
Allows the wearer to access other customized modes that let the knee swing freely for activities like bicycling or lock it out completely for extended periods of standing

Source: Otto Bock Healthcare

JOHN BLANCHARD / The Chronicle

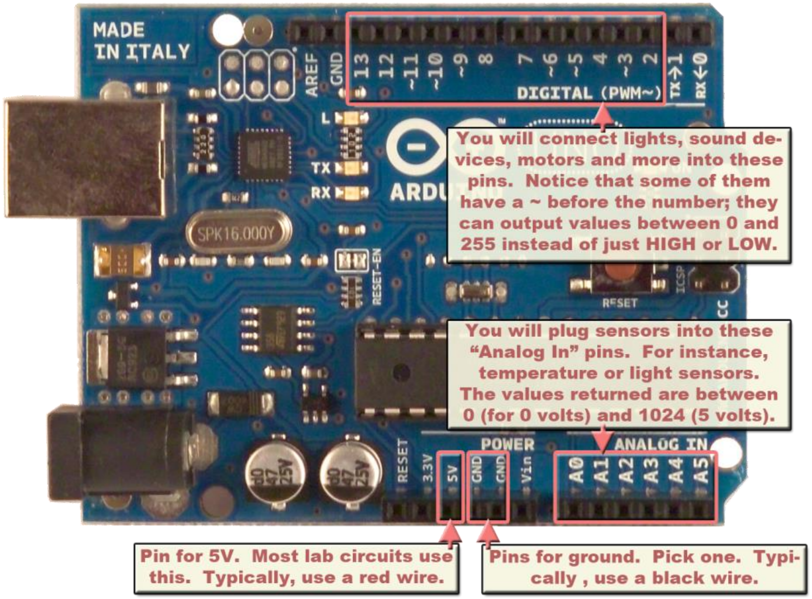
Examples for uC-based devices

- various sensors or meters:
  Body temperature, Blood Pressure,
  Blood Sugar Level, …
- Implants and prosthetics
- Pacemakers (for heart, breathing, ...)
- functional Electrostimulation
- Orthesis and artificial limbs
- Biosignal acquisition equipment

Adam blood glucose meter

www.heartratemonitor.co.uk

Introducing the Arduino



You will connect lights, sound devices, motors and more into these pins. Notice that some of them have a ~ before the number; they can output values between 0 and 255 instead of just HIGH or LOW.

You will plug sensors into these "Analog In" pins. For instance, temperature or light sensors. The values returned are between 0 (for 0 volts) and 1024 (5 volts).

Pin for 5V. Most lab circuits use this. Typically, use a red wire.

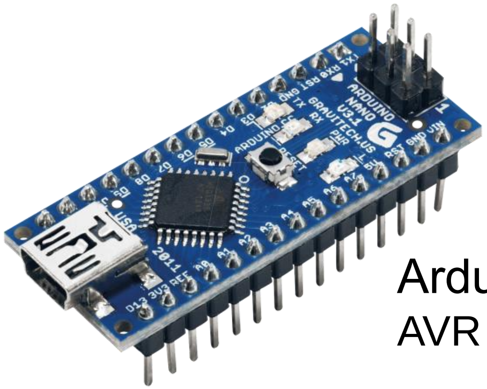
Pins for ground. Pick one. Typically , use a black wire.

What is Arduino ?

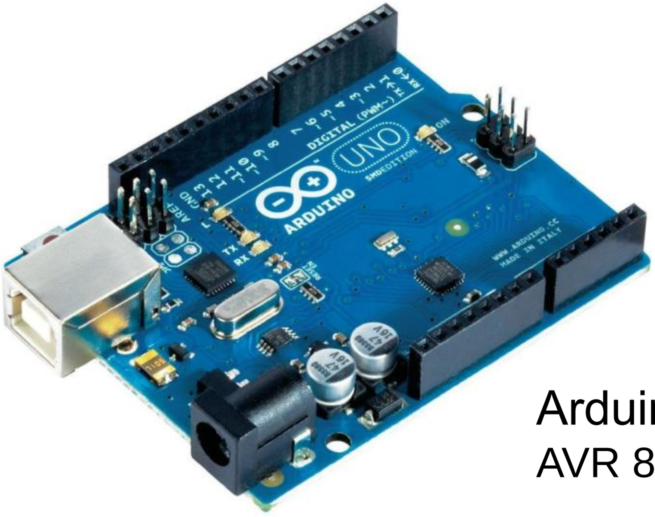
- Arduino is an integrated development environment (IDE) for microcontrollers, supporting many different microcontroller application boards
- The software support makes it easy to use various hardware modules (e.g. LC-displays, servo motors etc.)

Introducing the Arduino

Different board examples:



Arduino „nano“
AVR 8 bit

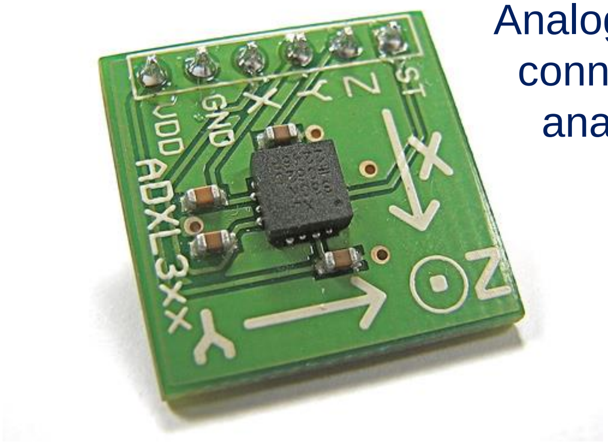


Arduino „UNO“
AVR 8 bit

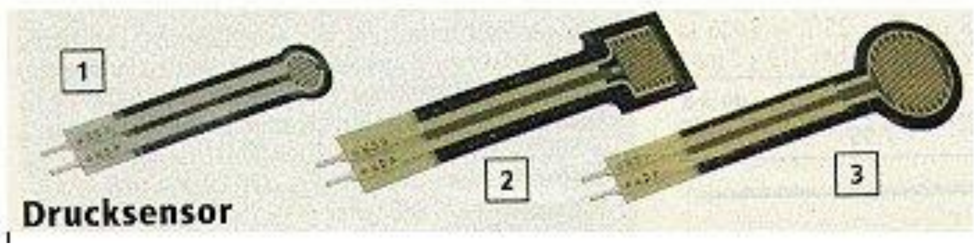
Arduino „Yun“
AVR+ Atheros 32 bit, Linux

Arduino „pro micro“
AVR 8 bit, HID-capable

Other useful sensors:

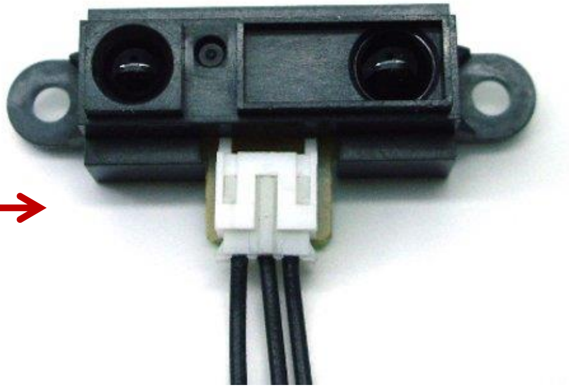


3-axis accelerometer
Analog version:
connect x/y/z to
analog inputs



Resistive force sensors / stain gauges / bend sensors

Proximity sensors using
ultrasonic waves (up to 8m)
or infrared light (up to 1m)



Other useful Actuators

Hydraulic or pneumatic
cylinders and valves

Relais to switch
higher loads
e.g. 220V devices

LC-Displays (graphical or alphanumeric),
usually connected via SPI or I2C interfaces

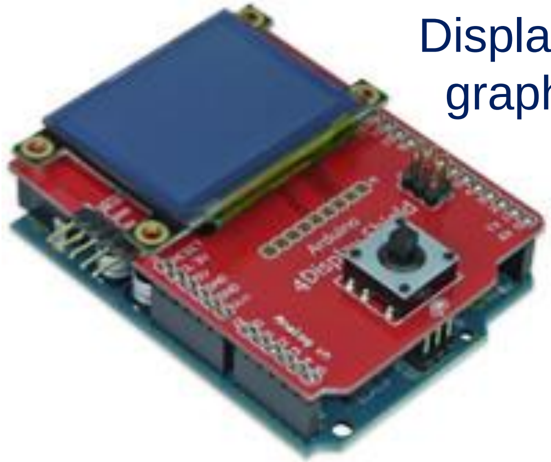
Solenoids / Valves
for fluid control

Stepper Motors
can be controlled precisely in
single steps of defined angles

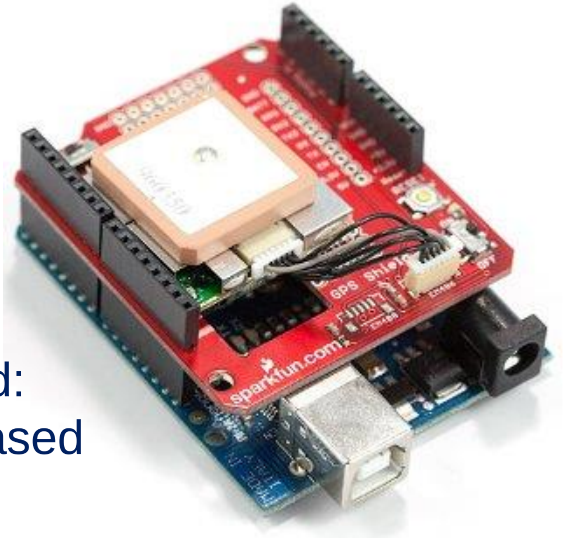
Arduino shields (for „Uno“ form factor)

Ethernet shield:
Run a http-server
send Twitter messages
receive emails etc.

Sound Wave shield:
play audio data
from SD card



Display shields:
graphical output

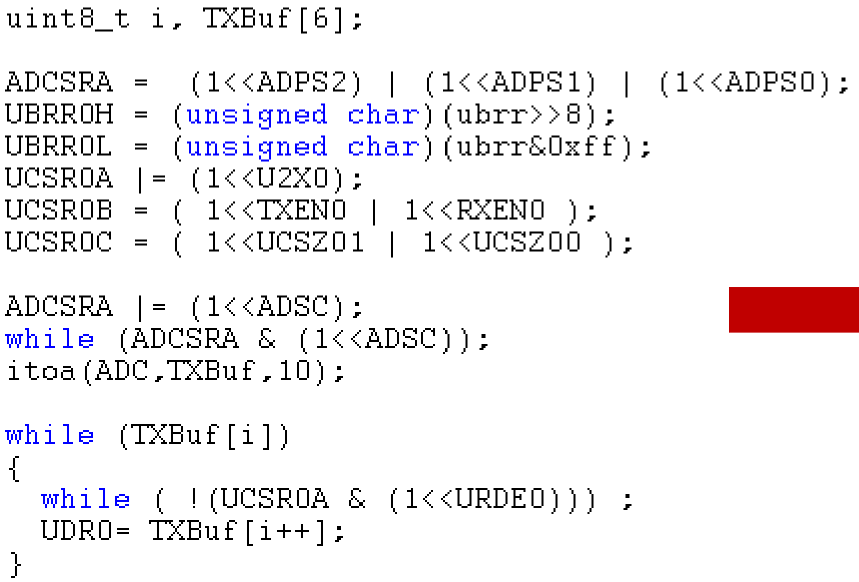


GPS shield:
location-based
activities

Introducing the Arduino

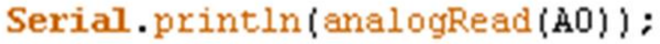
Classes for easy programming:

Without Arduino-libraries:
no hardware abstraction,
„register-level programming“



```
uint8_t i, TXBuf[6];

ADCSRA =  (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);
UBRR0H = (unsigned char)(ubrr>>8);
UBRR0L = (unsigned char)(ubrr&0xff);
UCSR0A |= (1<<U2X0);
UCSR0B = ( 1<<TXEN0 | 1<<RXEN0 );
UCSR0C = ( 1<<UCSZ01 | 1<<UCSZ00 );

ADCSRA |= (1<<ADSC);
while (ADCSRA & (1<<ADSC));
itoa(ADC,TXBuf,10);

while (TXBuf[i])
{
  while ( !(UCSR0A & (1<<URDE0)) ;
  UDR0= TXBuf[i++];
}
```

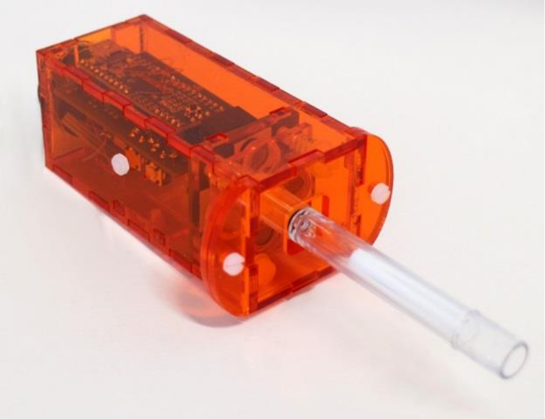
With Arduino-libraries for
hardware abstraction:



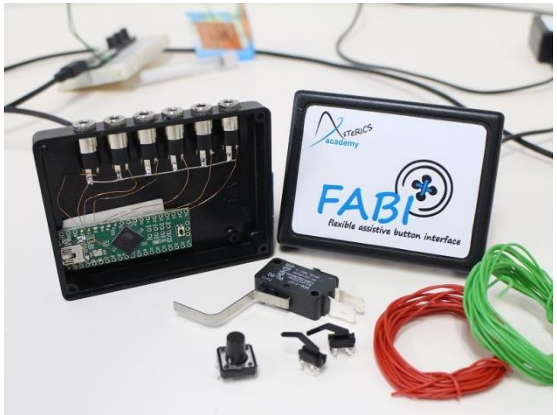
```
Serial.println(analogRead(A0));
```

AsTeRICS Academy Assistive Tools

- Two affordable, uC-based open source modules for computer control with Graphical User Interface



- **FLipMouse**
  zero-way joystick, mouse- and keyboard activities are created via finger- or lip/mouth interaction



- **FABI** - Flexible Assistive Button Interface
  control of a computer‘s mouse and typing by using up to 6 buttons or specialized input methods

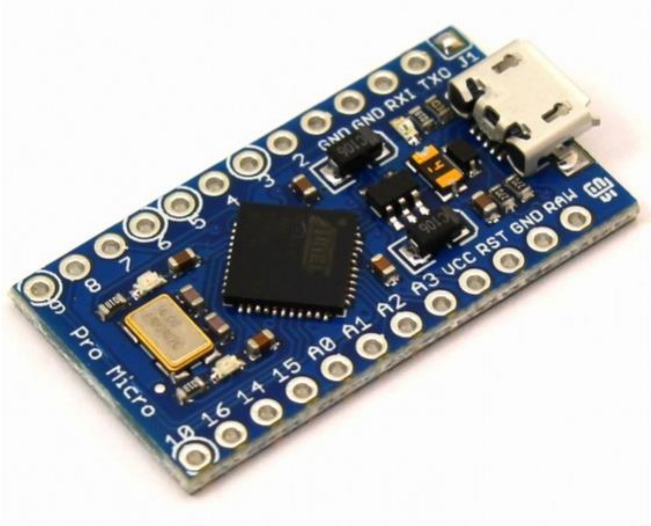
Download User Manuals and software:
http://www.asterics-academy.net/tools

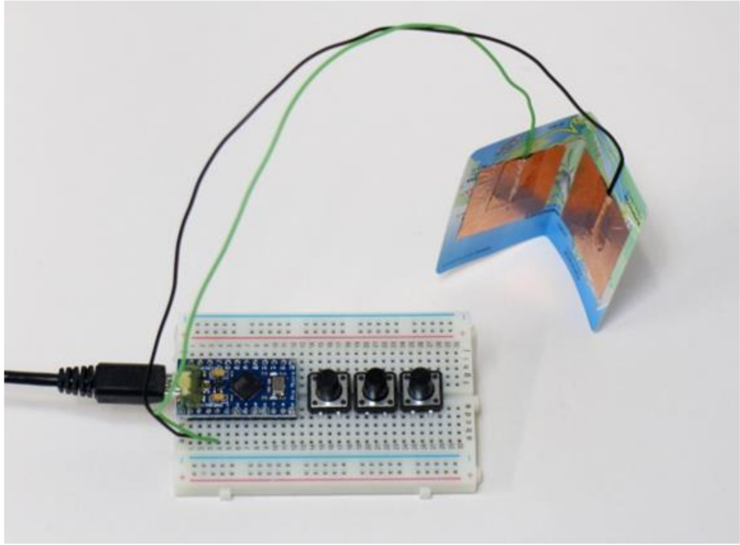
FLipMouse and FABI …

- .. can be used as „stand-alone modules“
  - works on many computers (Windows, Linux, Mac, Android)
  - no driver software needed
- .. can be used together with AsTeRICS
  - combinations with other sensors
  - music / sound creation, environmental control, ….
- .. are open source designs
  - we use them in our workshop and provide them for evaluations
- .. work with Arduino-compatible Microcontrollers

The AsTeRICS Academy FABI …

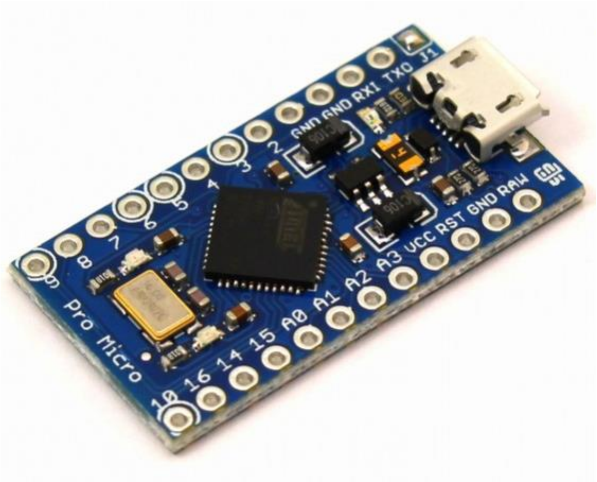
- Uses the Arduino Pro Micro microcontroller which can behave as HID device (mouse, keyboard, joystick) !
- Available for 4 $ (via chinese suppliers)
- We developed a firmware and GUI for customizing HID actions (mouse clicks, key presses, …)





FABI: What do we need ?

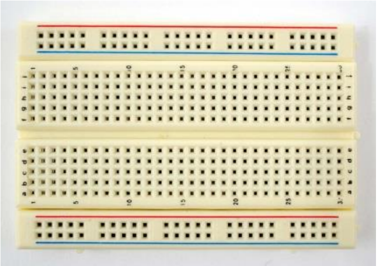
- „Arduino Pro Micro“ microcontroller
- Arduino Software to install the FABI software (firmware) on the board
- Pushbuttons or any kind of electrical contact which can be connected to the microcontroller
- Soldering iron & solder or a breadboard to connect the buttons
- Cables, pliers (to remove isolation)
- The „FabiGUI“ software application to configure the desired functions of the pushbuttons



Arduino Pro Micro:

- Atmel ATmega32U4 microcontroller
- 32 kB Flash Memory
- 16 MHz Clock Speed
- USB interface, HID compatible (it can be a mouse or keyboard)

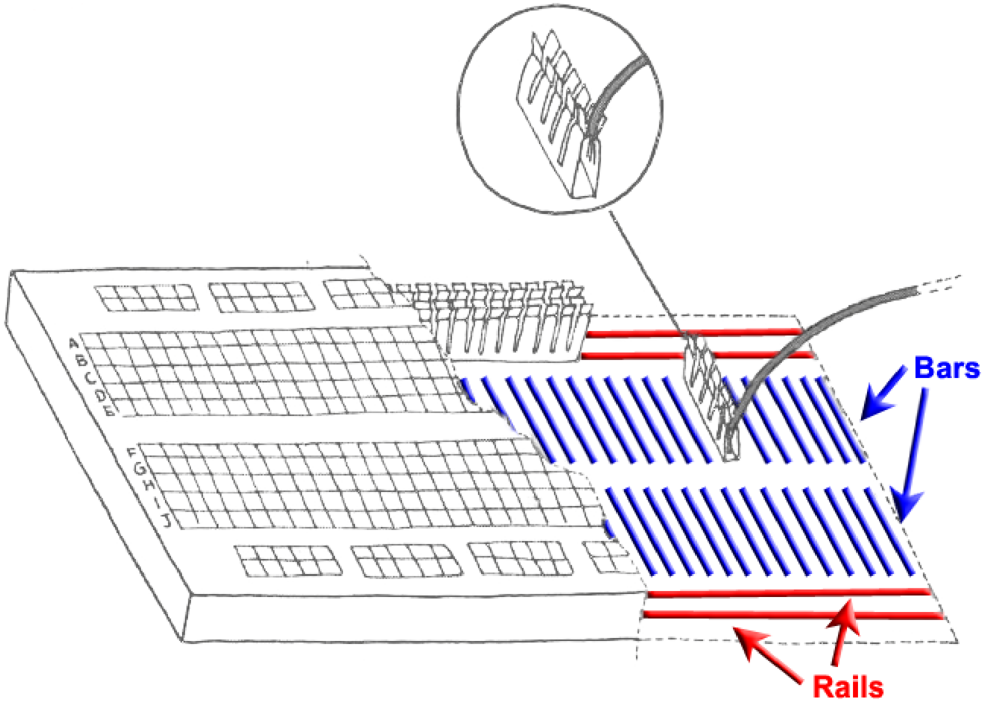
Connecting things….

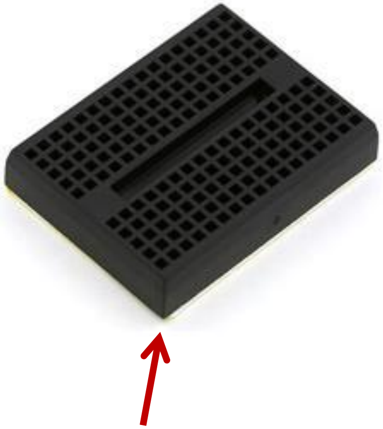


Breadboards for prototyping:

the bars are vertically connected,
the rails are horizontally connected

→ insert components / cables to make temporary electrical connections !

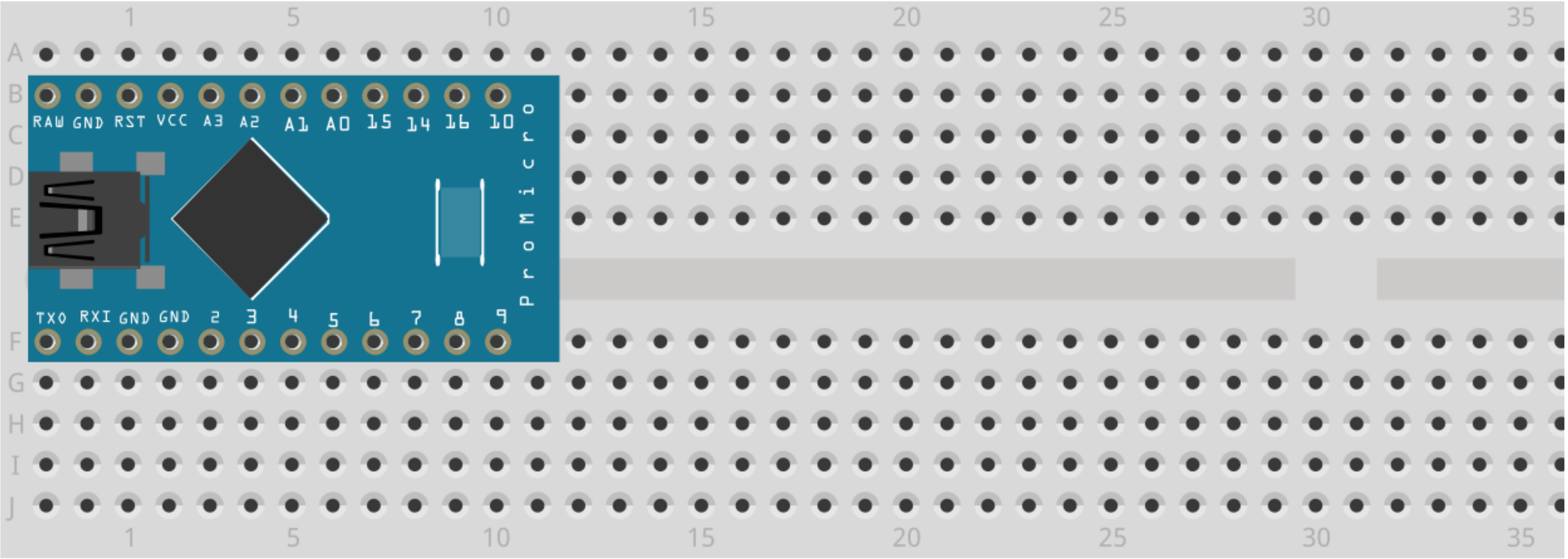




small breadboards
do not have rails …

Connecting things ….

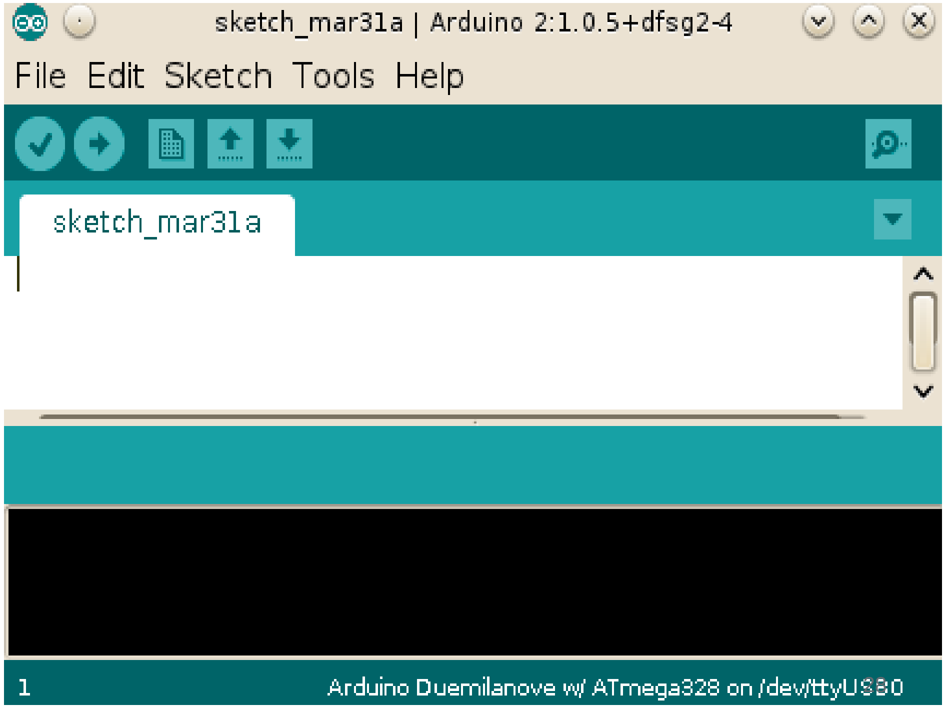
- Insert the Arduino microcontroller into the breadboard
  (we need space aside and below the Arduino for the buttons)


1
5
10
15
20
25
30
35
A
B
C
D
E
F
G
H
I
J
RAW GND RST VCC A3 A2 A1 A0 15 14 16 10
TX0 RXI GND GND 2 3 4 5 6 7 8 9
Pro Micro

Installing the Arduino IDE

- First, download and install/unzip the Arduino software
  Software/SoftwareDevelopment/arduino-1.6.7-windows.exe or
  http://www.arduino.cc

- After the installation, you can start the
  Integrated Development Environment (IDE):


sketch_mar31a | Arduino 2:1.0.5+dfsg2-4
File
Edit
Sketch
Tools
Help
sketch_mar31a
1
Arduino Duemilanove w/ ATmega328 on /dev/ttyUSB0

Connecting the microcontroller

- Standard MicroUSB cable (A to B)

- The left connector fits into the Arduino Pro microcontroller
  Please be careful: cheap board, connector is not very stable :-)

- The right connector fits into the PC

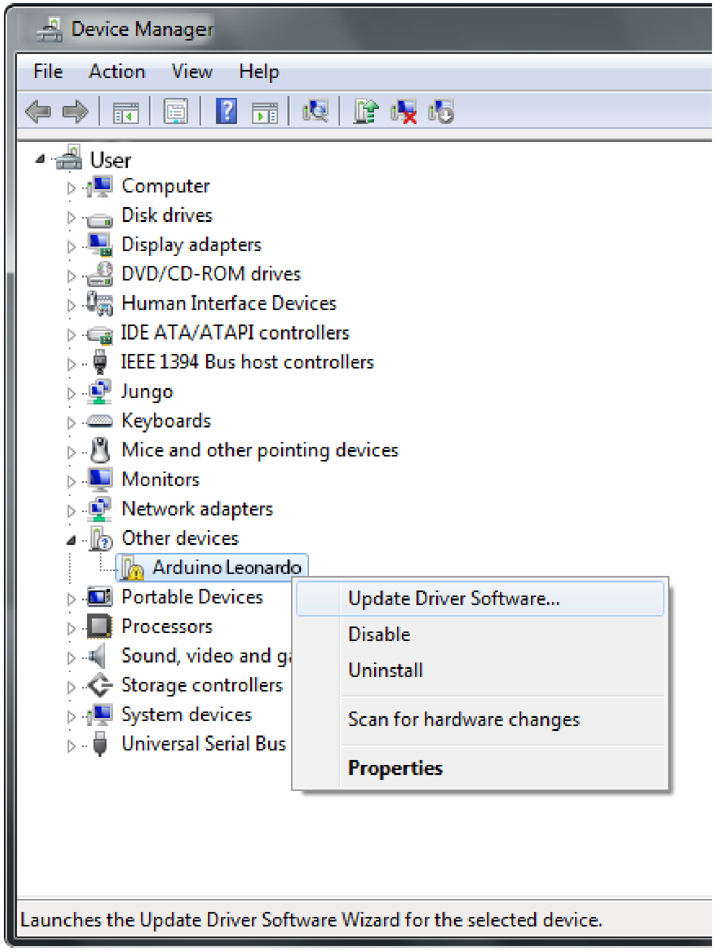
After plugging in, the PC/Windows tries to find a driver (COM-Port)

Installing the COM Port driver

If the driver installer does not start automatically:

- Open windows control panel → hardware → Device Manger
- Find the Arduino device
- Right click it and choose „Update Driver Software“

For instructions for other operating systems please refer to http://www.arduino.cc

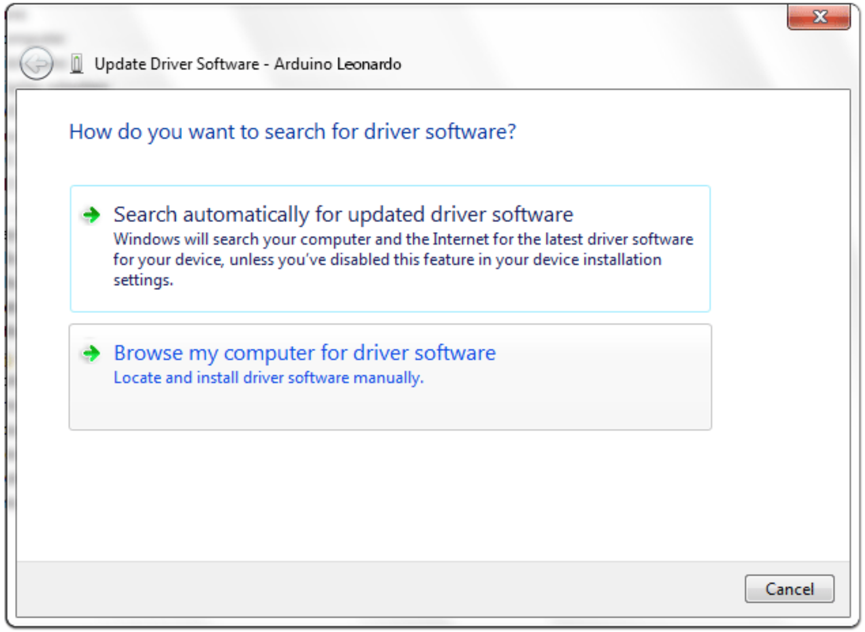


Launches the Update Driver Software Wizard for the selected device.

Installing the COM Port driver

- Select „Browse my computer for driver software“
- Select the installation path of the Arduino IDE
- Activate „Include subfolders“
- After a click on „Next“, Windows should install the necessary drivers

→ finished, close the dialog

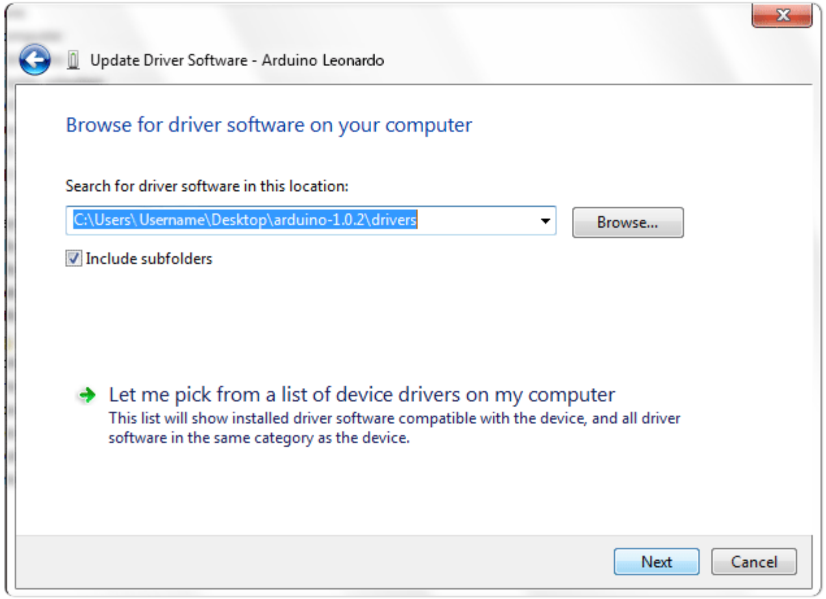


Update Driver Software - Arduino Leonardo

How do you want to search for driver software?

Search automatically for updated driver software
Windows will search your computer and the Internet for the latest driver software for your device, unless you've disabled this feature in your device installation settings.

Browse my computer for driver software
Locate and install driver software manually.



Update Driver Software - Arduino Leonardo

Browse for driver software on your computer

Search for driver software in this location:
C:\Users\Username\Desktop\arduino-1.0.2\drivers

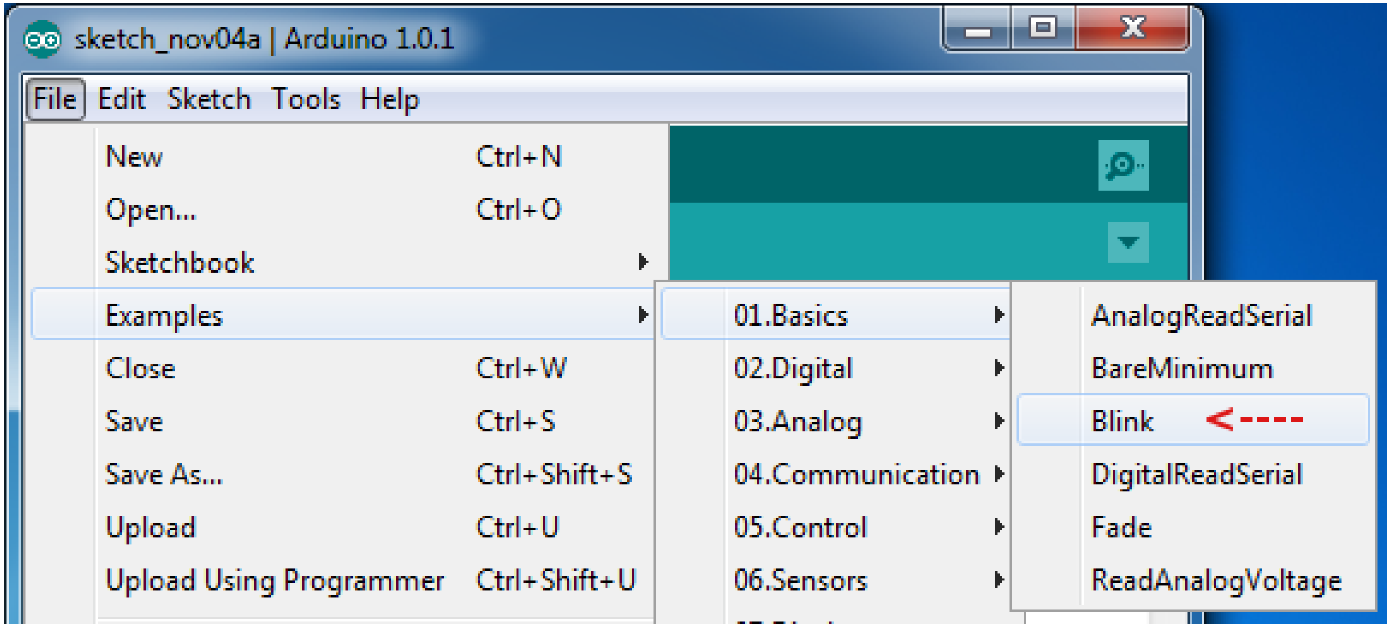
☑ Include subfolders

Let me pick from a list of device drivers on my computer
This list will show installed driver software compatible with the device, and all driver software in the same category as the device.

First demo example: „Blink“

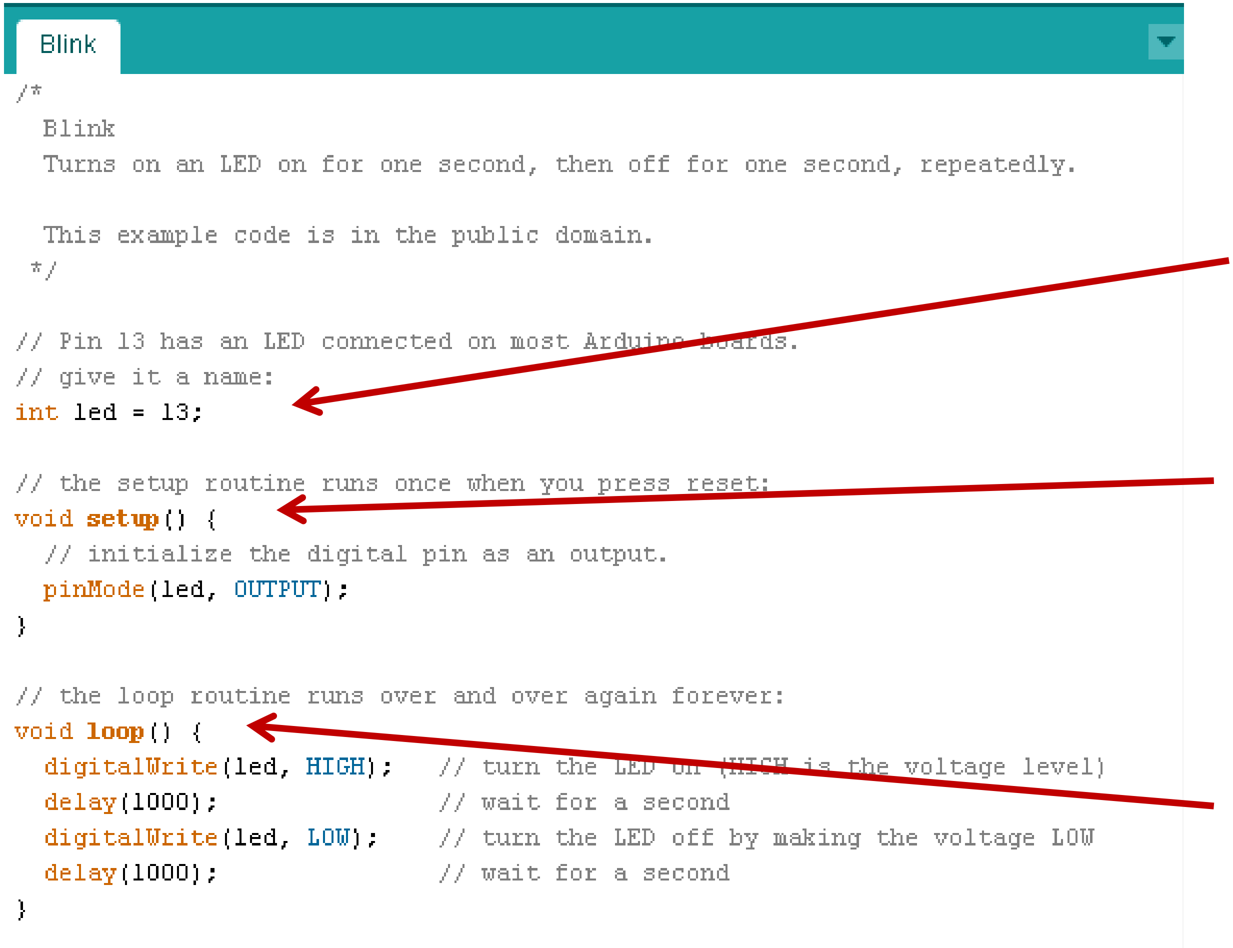
Open the „Blink“-demo program:

Examples -> 01.Basics -> Blink



… the Blinking Led – „Hello World“ for microcontroller programmers !

The structure of an Arduino program:



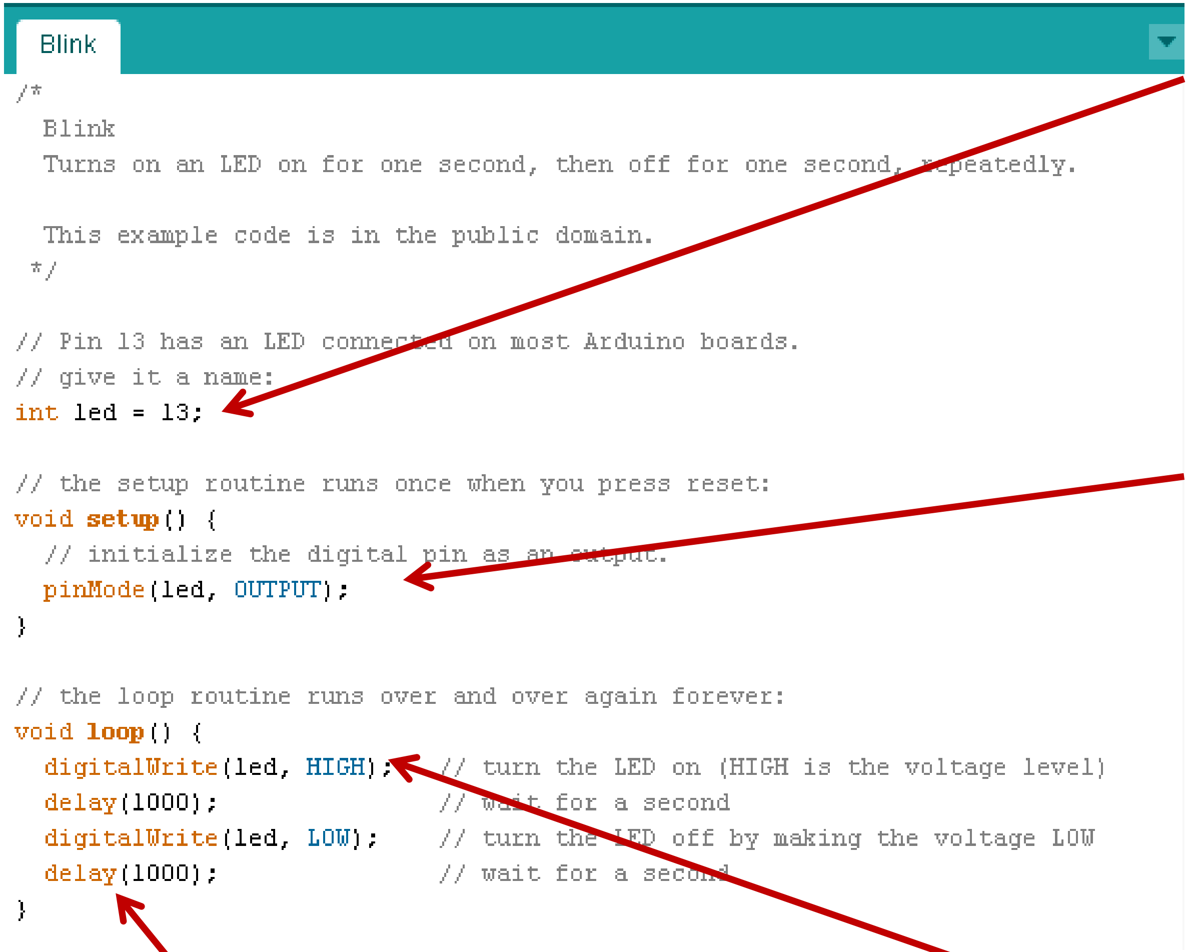
```
/*
  Blink
  Turns on an LED on for one second, then off for one second, repeatedly.

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

variable definitions

a setup() section
which is executed once
When the program starts

a loop() section
which is endlessly looped
after setup() has been done

What do these commands mean ?



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

  This example code is in the public domain.
 */

// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);               // wait for a second
}
```

int <variable_name>
creates an integer variable
that can store changing
non-fractional numbers
Change this number to 9
for the Arduino Pro Micro !

pinMode (<pinnr>, <mode>)
defines if a pin is used as
OUTPUT or INPUT

OUTPUT is used to drive
voltages to actuators (e.g LED)

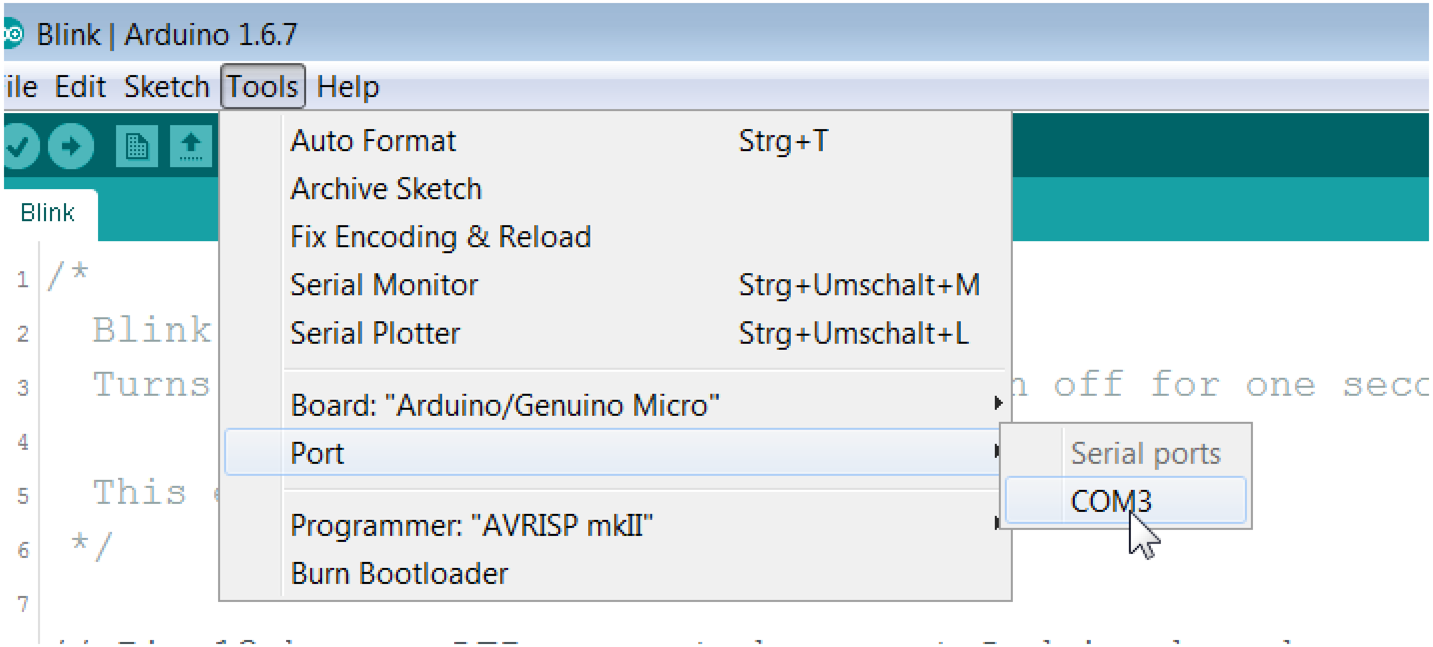
INPUT is used to read voltages
(eg. from a sensor / button)

delay (<milliseconds>)
waits a given time in milliseconds

digitalWrite (<pinnr>, <value>)
is used to set an output Pin
LOW (0 Volt) or HIGH (5 Volt)

Upload the Blink Demo

- Select the correct COM port
  which has been installed and is now visible
  in the Device Manager (example here: COM3)



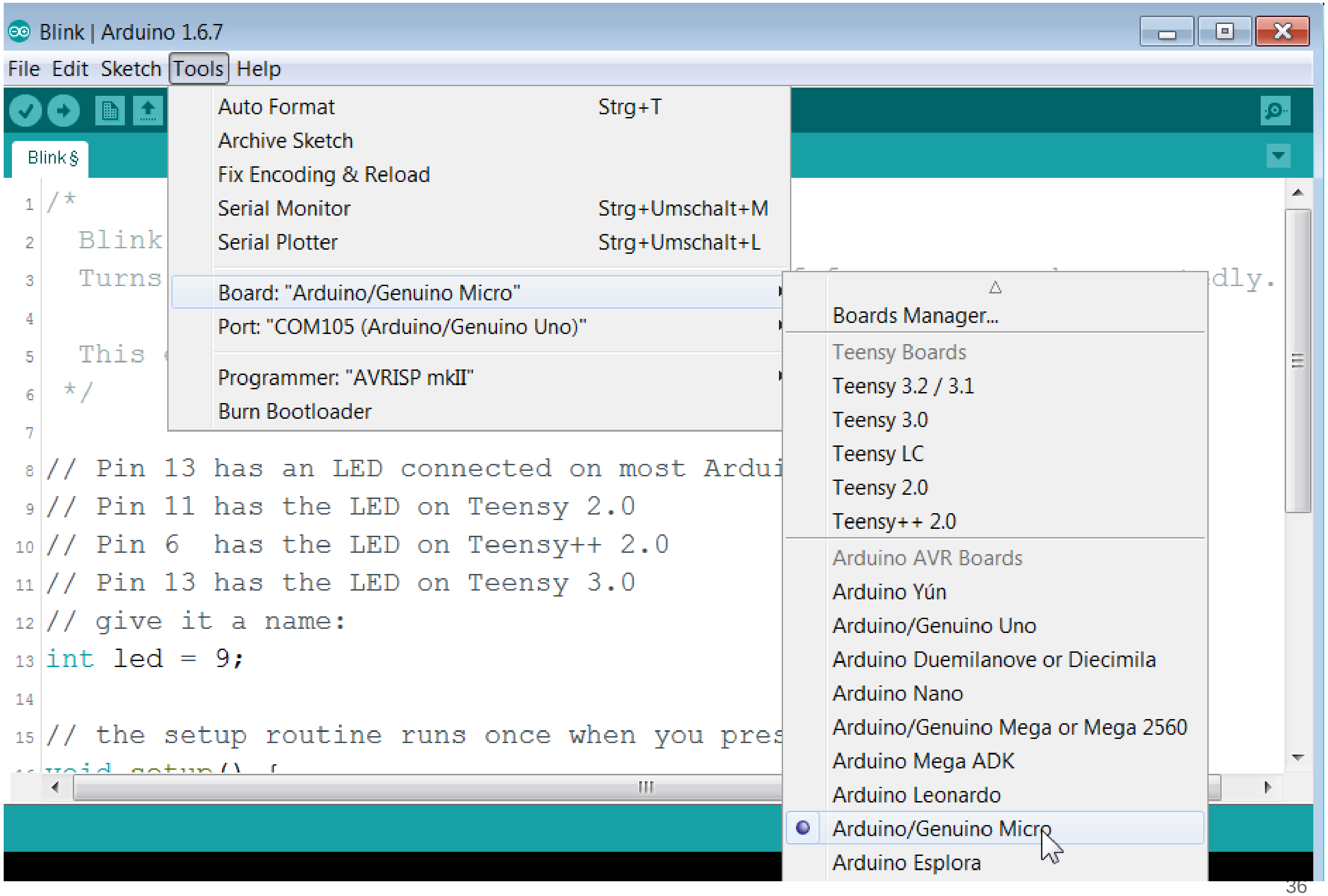
Blink | Arduino 1.6.7

ile Edit Sketch Tools Help
Blink
Auto Format Strg+T
Archive Sketch
Fix Encoding & Reload
Serial Monitor Strg+Umschalt+M
Serial Plotter Strg+Umschalt+L
Board: "Arduino/Genuino Micro"
Port
Programmer: "AVRISP mkII"
Burn Bootloader
Serial ports
COM3

```
/*
  Blink
  Turns                                  n off for one secc

  This 
 */
```

Upload the Blink Demo

- Select the correct Arduino board (Arduino/Genuino Micro):



Board: "Arduino/Genuino Micro"

Port: "COM105 (Arduino/Genuino Uno)"

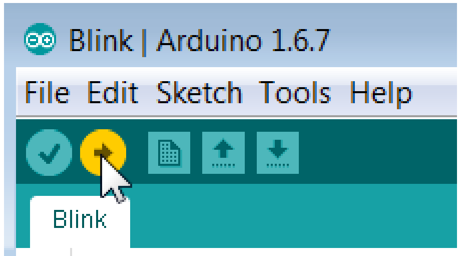
Programmer: "AVRISP mkII"

● Arduino/Genuino Micro

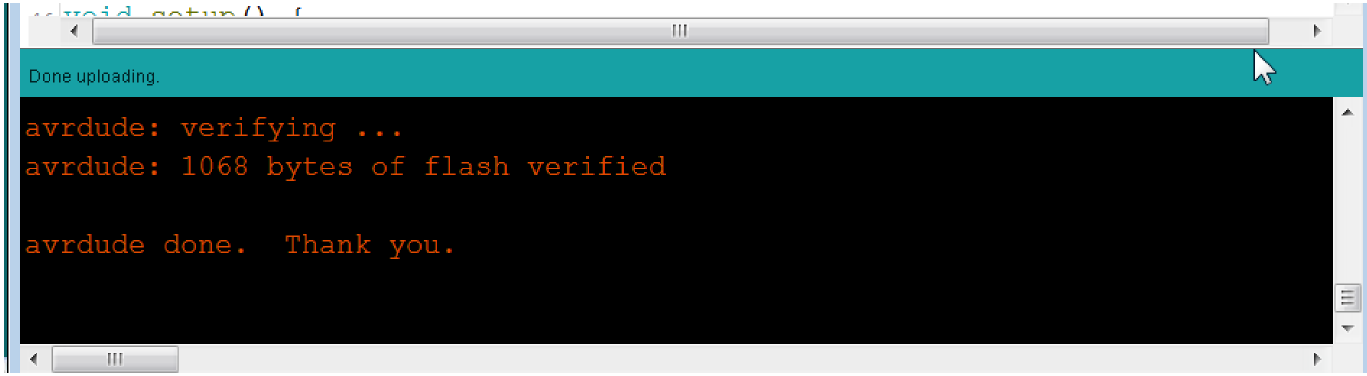
```
/*
  Blink
  Turns                                              dly.

  This
 */

// Pin 13 has an LED connected on most Ardui
// Pin 11 has the LED on Teensy 2.0
// Pin 6  has the LED on Teensy++ 2.0
// Pin 13 has the LED on Teensy 3.0
// give it a name:
int led = 9;

// the setup routine runs once when you pres
void setup() {
```

Upload the Blink Demo

- press the „Upload“ button:



- if everything works, you should see the rs/tx led flashing and after some seconds the status window shows:

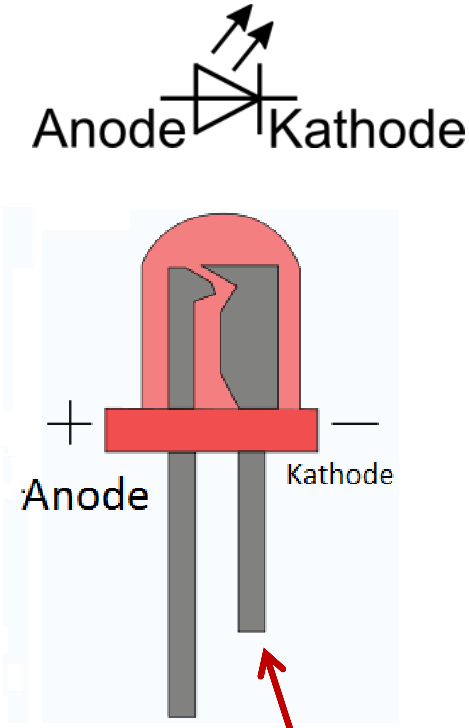


Done uploading.

```
avrdude: verifying ...
avrdude: 1068 bytes of flash verified

avrdude done.  Thank you.
```

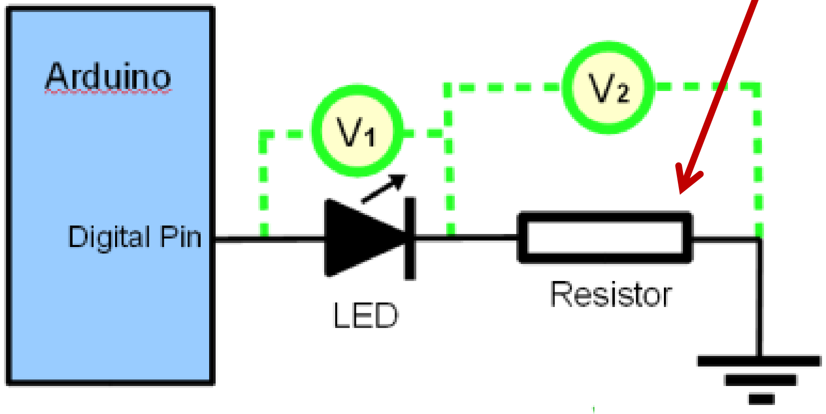
- now your program is running on the Arduino Board

How to connect a LED:



Microcontroller can apply 5V (high) or 0V (low) to a digital output Pin

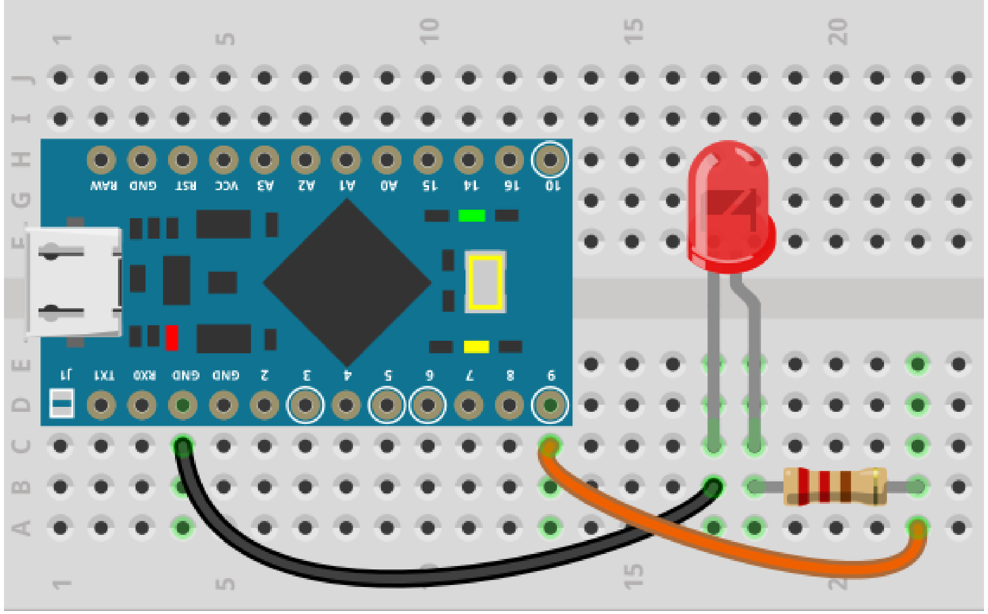
Resistor (470 Ohm) limits current to about 10mA to protect the LED

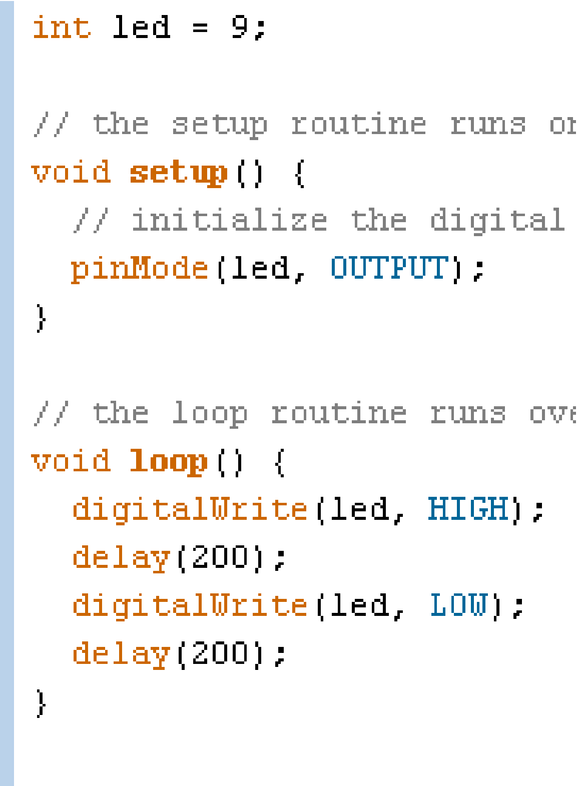


LED Cathode connected to lower potential (usually ground)

Ground Potential (GND) = 0 Volt (Battery minus pole)

Connect external LED to Arduino:

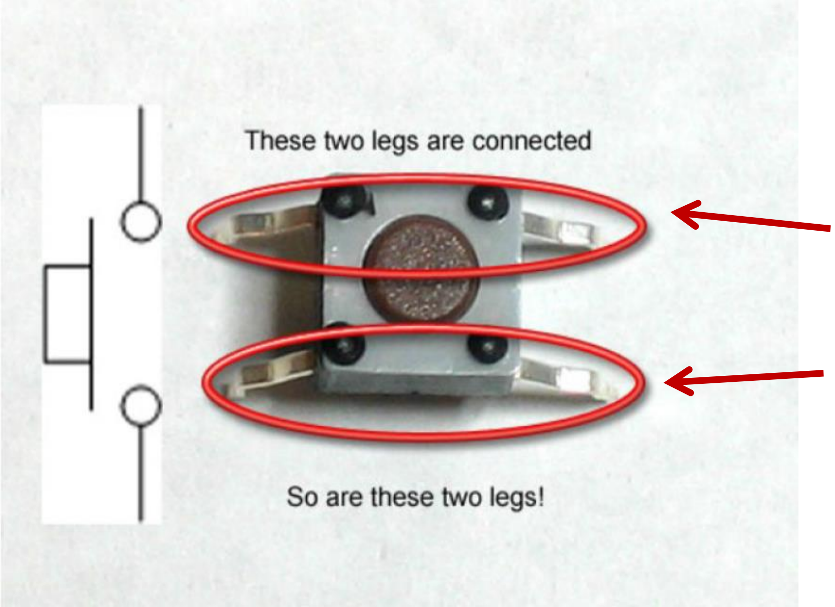




```
int led = 9;

// the setup routine runs o
void setup() {
  // initialize the digital
  pinMode(led, OUTPUT);
}

// the loop routine runs ove
void loop() {
  digitalWrite(led, HIGH);
  delay(200);
  digitalWrite(led, LOW);
  delay(200);
}
```

connected LED on breadboard, modified blink example (Pin 9) !

How does a pushbutton work ?

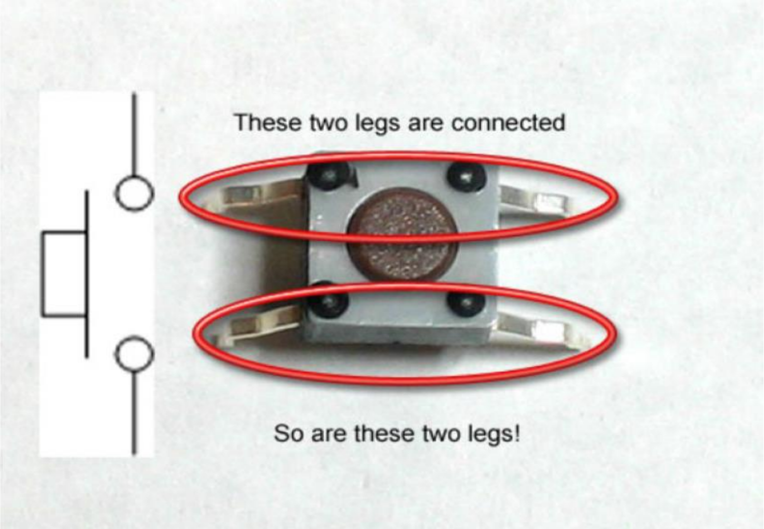


The top legs are always connected

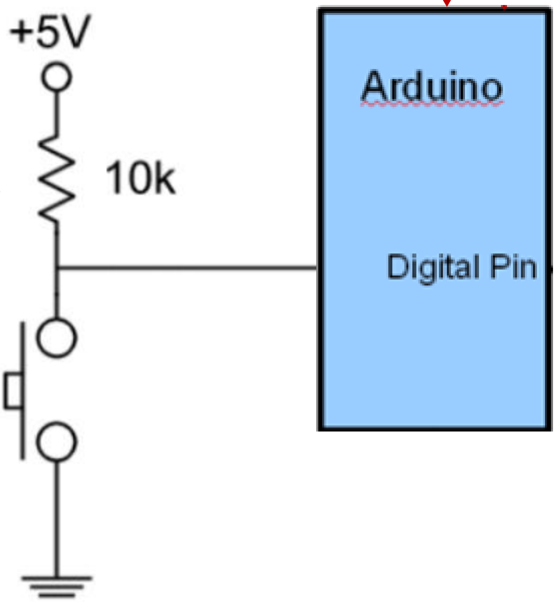
The bottom legs are always connected

- When the button is pressed, all 4 legs will be connected
- This can be used to lead a signal to the microcontroller:
  - Connect the top legs to the microcontroller
  - Connect the bottom legs to the zero volt signal (also called „ground“ or „GND“)

User Input ! - How to read a switch / button:



Microcontroller can measure voltage (high or low) at a digital input Pin

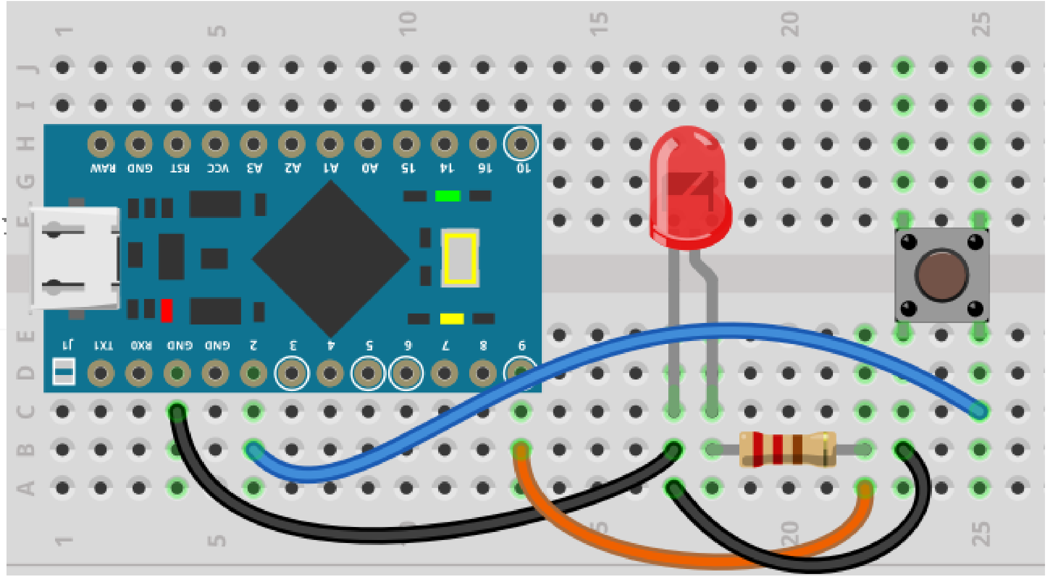


Resistor pulls Pin-voltage to 5V (high) if the switch is not pressed

Button connects Pin to 0V (low) if pressed

„If “- statements: using conditions!

```
int led = 9;
int button = 2;

// the setup routine runs once when you press reset:
void setup() {
  pinMode(led, OUTPUT);             // led pin is configured as ouptut
  pinMode(button, INPUT_PULLUP);    // button pin is configured as input
                                    // with internal pullup-resistor
}

// the loop routine runs over and over again forever:
void loop() {

  if (digitalRead ( button) == LOW)   {
      digitalWrite(led, HIGH);    // turn the LED on
  }
  else
  {
    digitalWrite(led, LOW);     // turn the LED off by
  }
}
```



Loops:

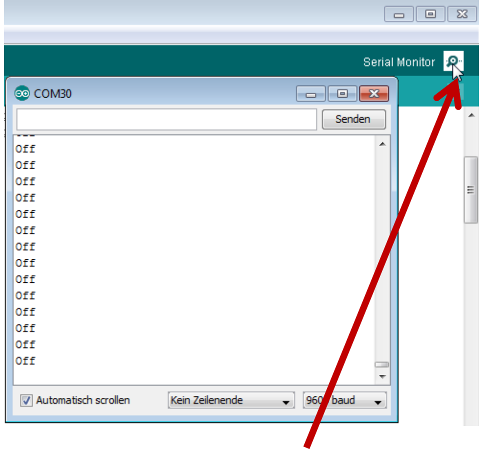
Performing statements several times, or until a condition occurs:

Examples:

```
for (x=0; x<10; x++)     // for loop: performs the loop-block 10 times
                         //  (x  increases from 0 to 9 )
{
     a=a+1;              // this is done 10 times !
}


while (x>10)       // while loop: performs loop-block as long as x > 10
{
    x=x-1;
    y=y*2;
}
```

Send and receive information:

```
int led = 9;
int button = 2;

// the setup routine runs once when you press reset:
void setup() {
  pinMode(led, OUTPUT);          // led pin is configured as ouptut
  pinMode(button, INPUT_PULLUP); // button pin is configured as input
                                 // with internal pullup-resistor

  Serial.begin (9600);    // begin communication with PC, 9600 Baud
}

// the loop routine runs over and over again forever:
void loop() {
  delay(100);
  if (digitalRead ( button) == LOW)   {
      digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
      Serial.println ("On");     // send "On" to the PC display window
  }
  else
  {
    digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
    Serial.println ("Off");    // send "Off" to the PC display window
  }
}
```



```
Off
Off
Off
Off
Off
Off
Off
Off
Off
Off
Off
Off
Off
Off
```

☑ Automatisch scrollen Kein Zeilenende 9600 baud

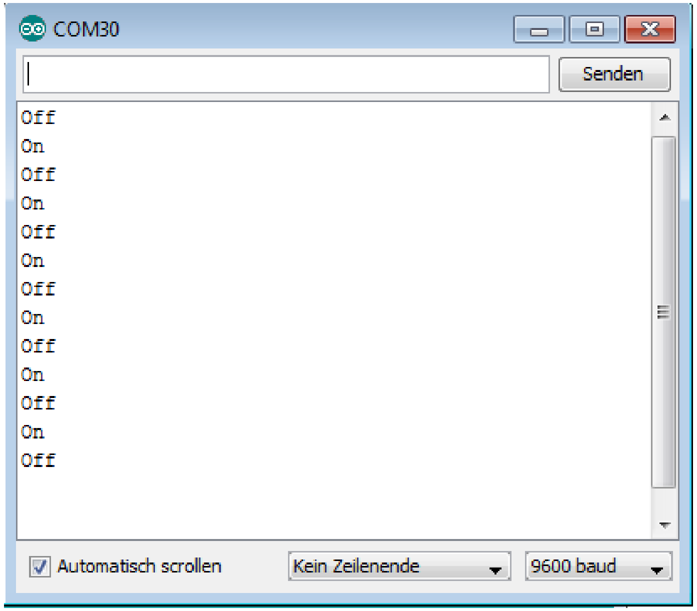
Click here to open the Serial Monitor window

This starts a communication with the PC via the COM Port

Serial.println(..) is used to send human-readable text (ASCII code) via the COM Port

Send only changing values:

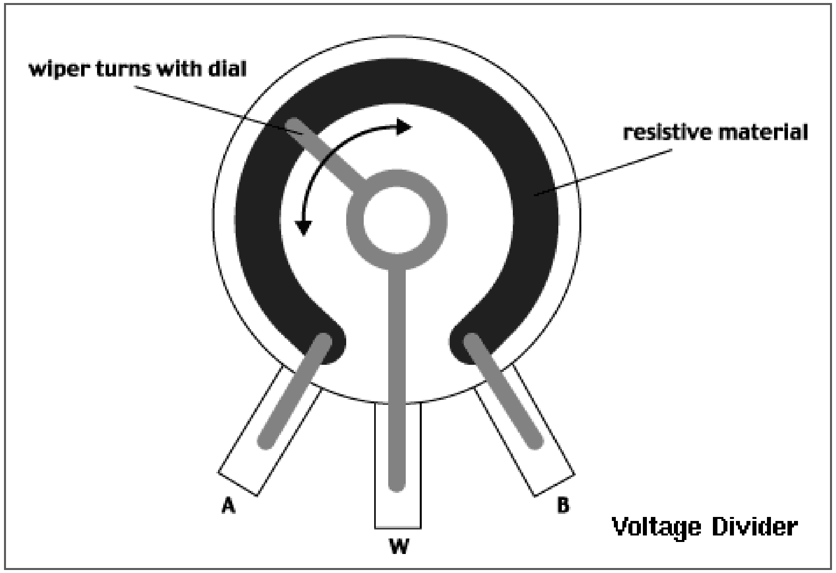
```
int led = 9;
int button = 2;
int act_state=0;

void setup() {
  pinMode(led, OUTPUT);
  pinMode(button, INPUT_PULLUP);

  Serial.begin (9600);   // begin communication with PC, 9600 Baud
}

// the loop routine runs over and over again forever:
void loop() {
  if (digitalRead ( button) == LOW)  {
    if (act_state==1) {          // only if actual state is 1
      digitalWrite(led, HIGH);
      Serial.println ("On");
      act_state=0;           // set actual state to 0
    }
  }
  else
  {
    if( act_state==0) {        // only if actual state is 0
      digitalWrite(led, LOW);
      Serial.println ("Off");
      act_state=1;         // set actual state to 1
    }
  }
}
```

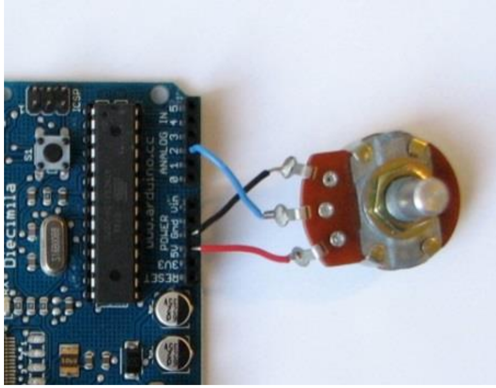


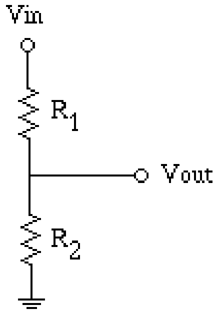
```
Off
On
Off
On
Off
On
Off
On
Off
On
Off
On
Off
```

☑ Automatisch scrollen Kein Zeilenende 9600 baud

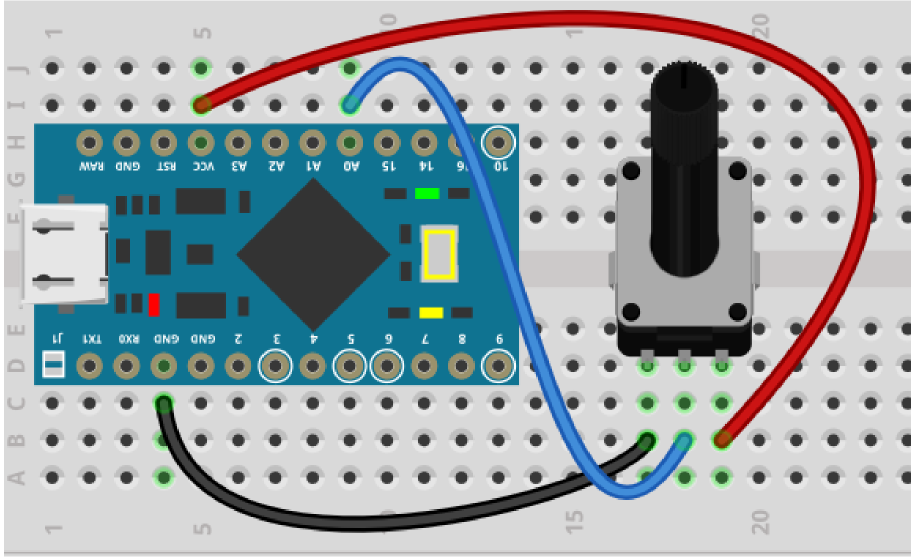
Connect a potientiometer





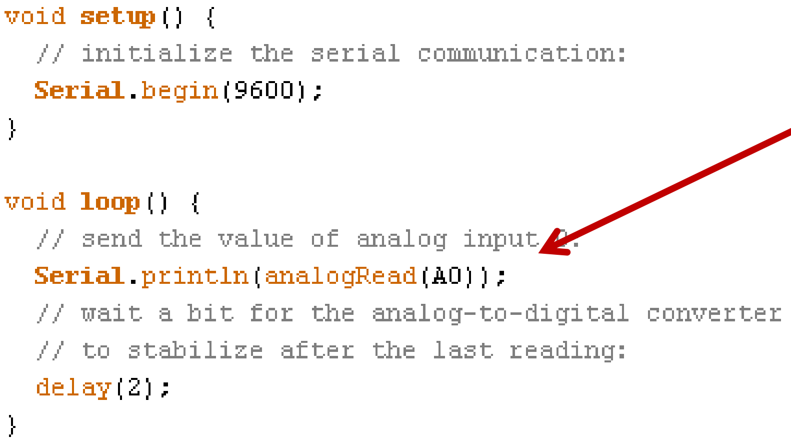


$$\text{Vout} = \frac{R_2}{R_1 + R_2}\,\text{Vin}$$



http://www.arduino.cc/en/Tutorial/Potentiometer

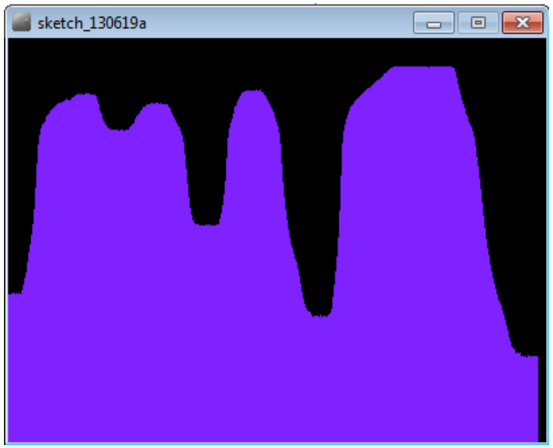
Read analog voltages



```
void setup() {
  // initialize the serial communication:
  Serial.begin(9600);
}

void loop() {
  // send the value of analog input 0:
  Serial.println(analogRead(A0));
  // wait a bit for the analog-to-digital converter
  // to stabilize after the last reading:
  delay(2);
}
```

The Arduino Code sends the voltage present at the analog input A0 to the COM Port

The range of the value is
0 (for 0 Volts) to 1023 (for 5 Volts)

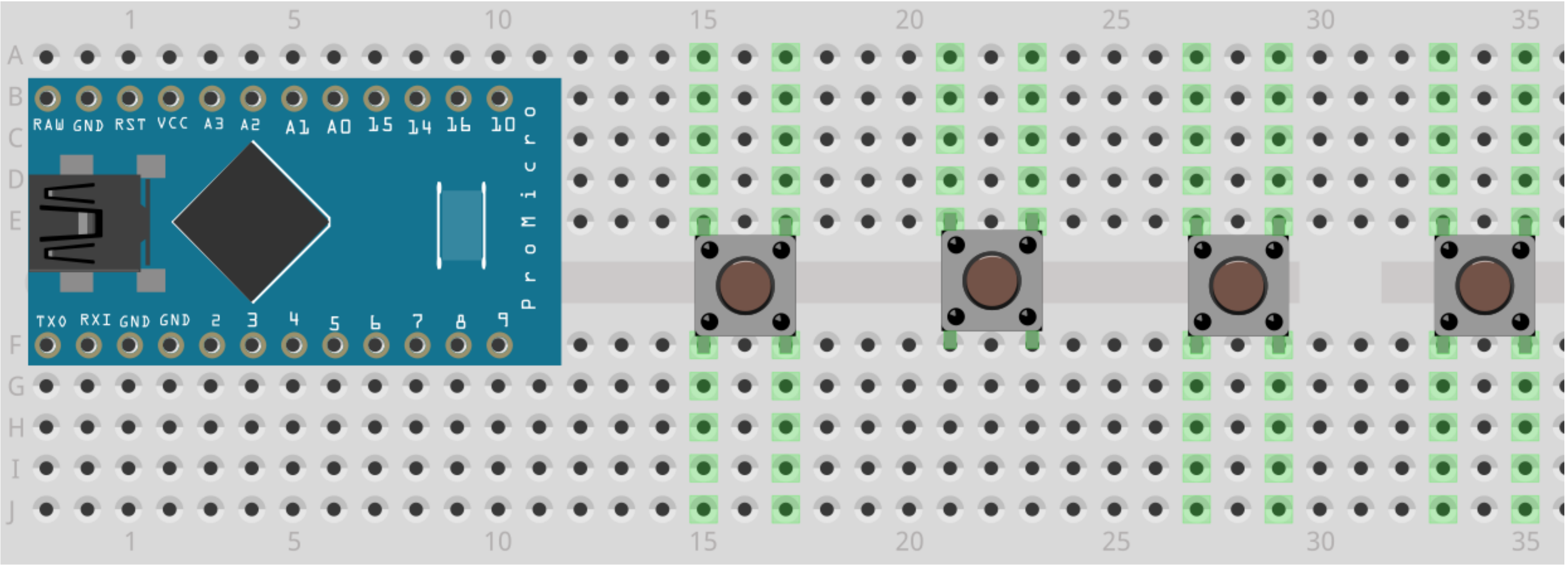


The processing code draws the values it receives from the Arduino in form as vertical lines.

The human readable output of println must be converted into a number

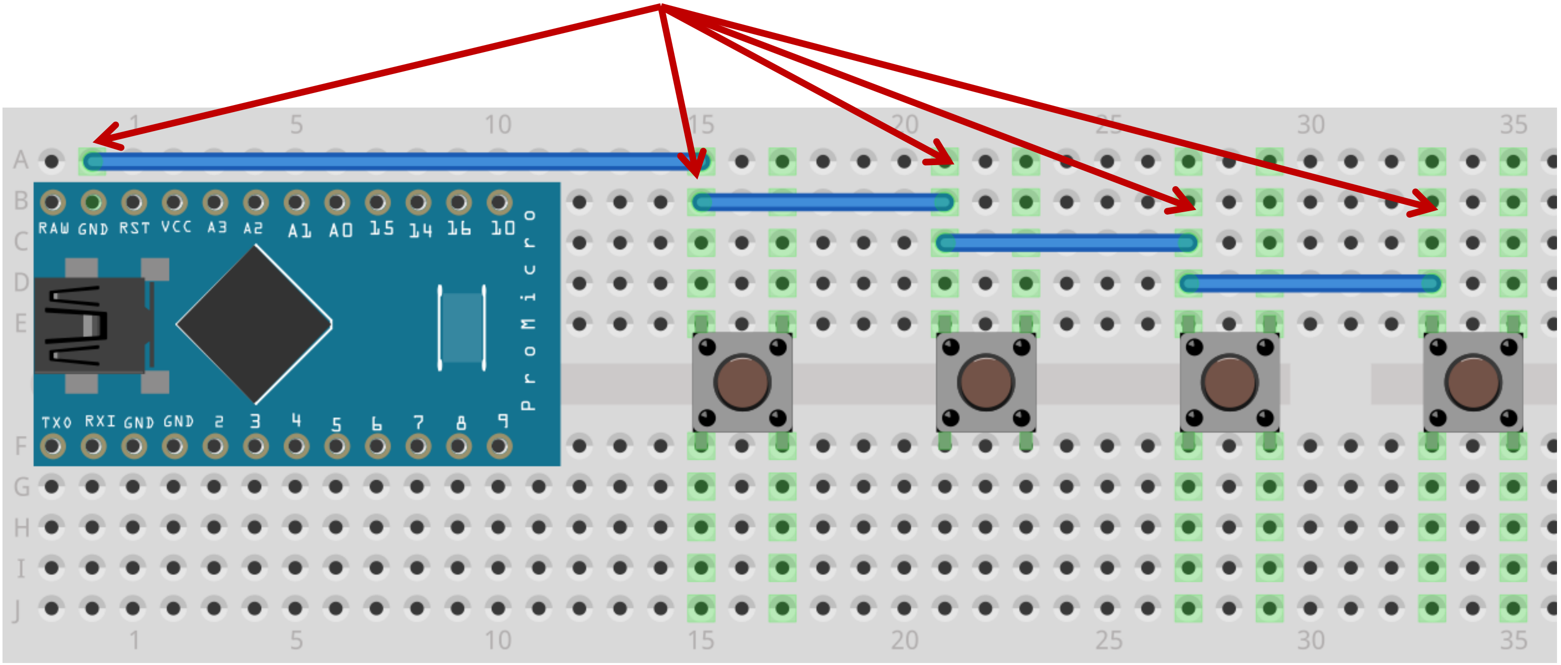
Connecting multiple pushbuttons

- Place buttons on the breadboard:
    - Using small buttons: up to 5 buttons can be placed
    - Using big buttons: up to 3 buttons can be placed
    - pay attention to the orientation: the legs connect 2 columns !



Connecting pushbuttons to FABI

- Now to connect all buttons to GND
- Remember: the upper and lower columns (green lines) are always connected internally in the breadboard



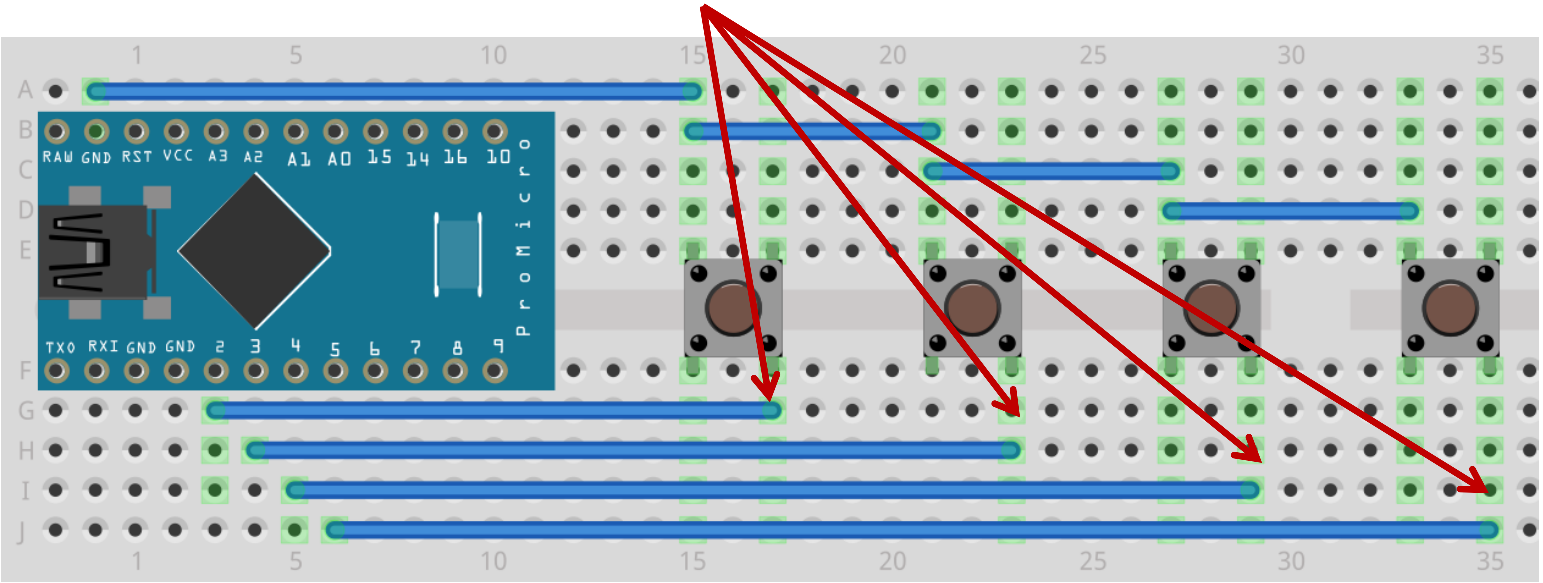
Every PushButton connects to 0V (low, Pin „GND“)

1
5
10
15
20
25
30
35
A
B
C
D
E
F
G
H
I
J
RAW GND RST VCC A3 A2 A1 A0 15 14 16 10
TXO RXI GND GND 2 3 4 5 6 7 8 9
Pro Micro

Connecting pushbuttons to FABI

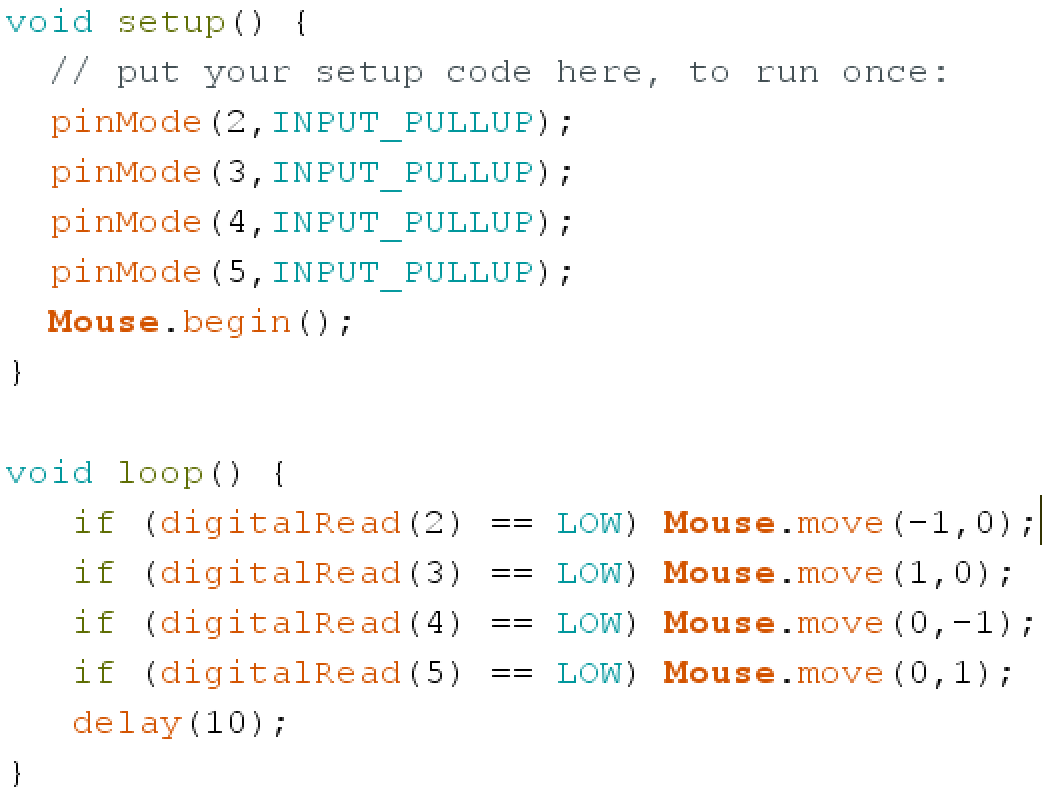
- Now connect the other connection of each button to one unique input pin of the Arduino microcontroller
- We use pins 2-5 in this example
- The FABI software supports pins 2-7 (for buttons 1 – 6)

Other side of PushButtons connect to Input pins (separate for each button)


1
5
10
15
20
25
30
35
A
B
C
D
E
F
G
H
I
J
RAW GND RST VCC A3 A2 A1 A0 15 14 16 10
TXO RXI GND GND 2 3 4 5 6 7 8 9
Pro Micro

Let the Mouse move !

(/SummerSchool/Module4…/ArduinoExamples/moveMouse)

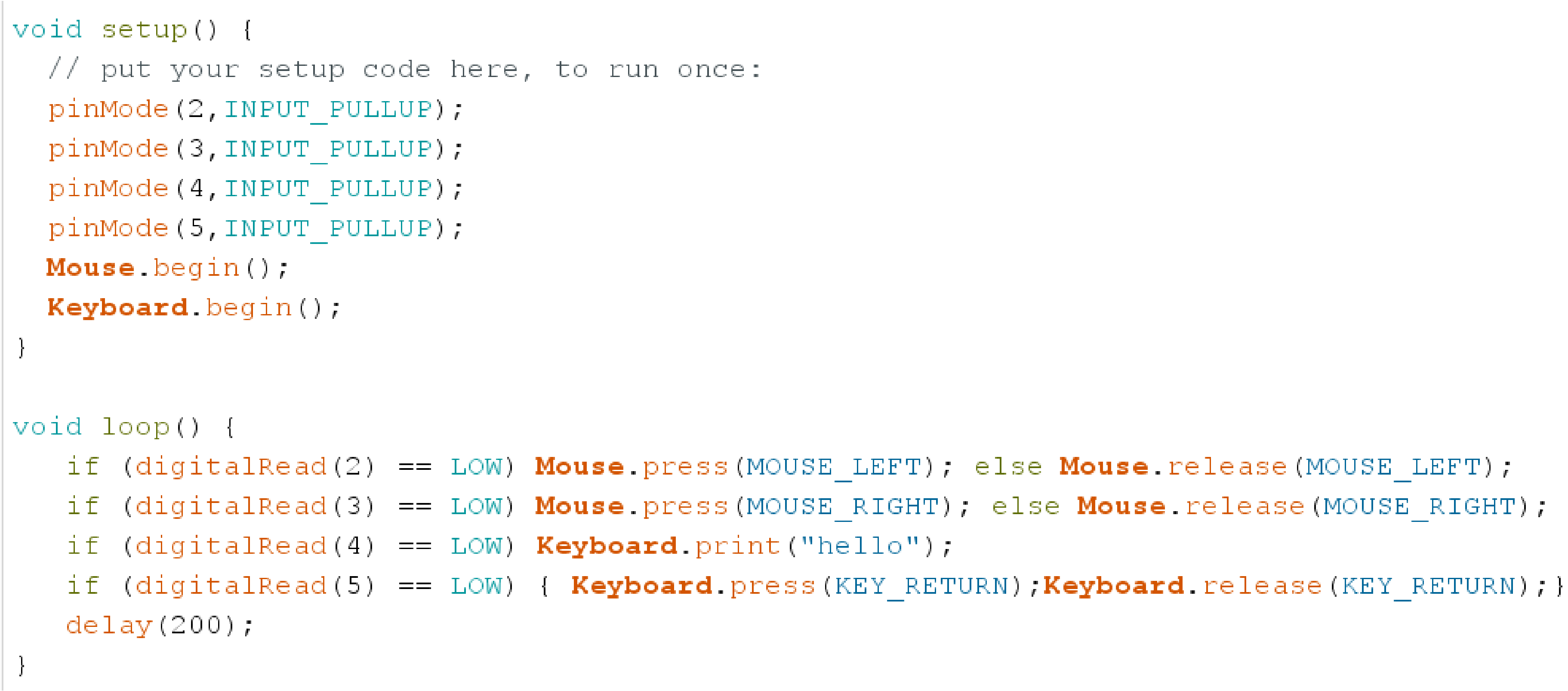


```
void setup() {
  // put your setup code here, to run once:
  pinMode(2,INPUT_PULLUP);
  pinMode(3,INPUT_PULLUP);
  pinMode(4,INPUT_PULLUP);
  pinMode(5,INPUT_PULLUP);
  Mouse.begin();
}

void loop() {
   if (digitalRead(2) == LOW) Mouse.move(-1,0);
   if (digitalRead(3) == LOW) Mouse.move(1,0);
   if (digitalRead(4) == LOW) Mouse.move(0,-1);
   if (digitalRead(5) == LOW) Mouse.move(0,1);
   delay(10);
}
```

Note: The Arduino UNO board doesn‘t support mouse or keyboard operation.
The Arduino „Pro Micro“ and „Leonardo“ have these capabilities.

Mouse clicks and keyboard input !

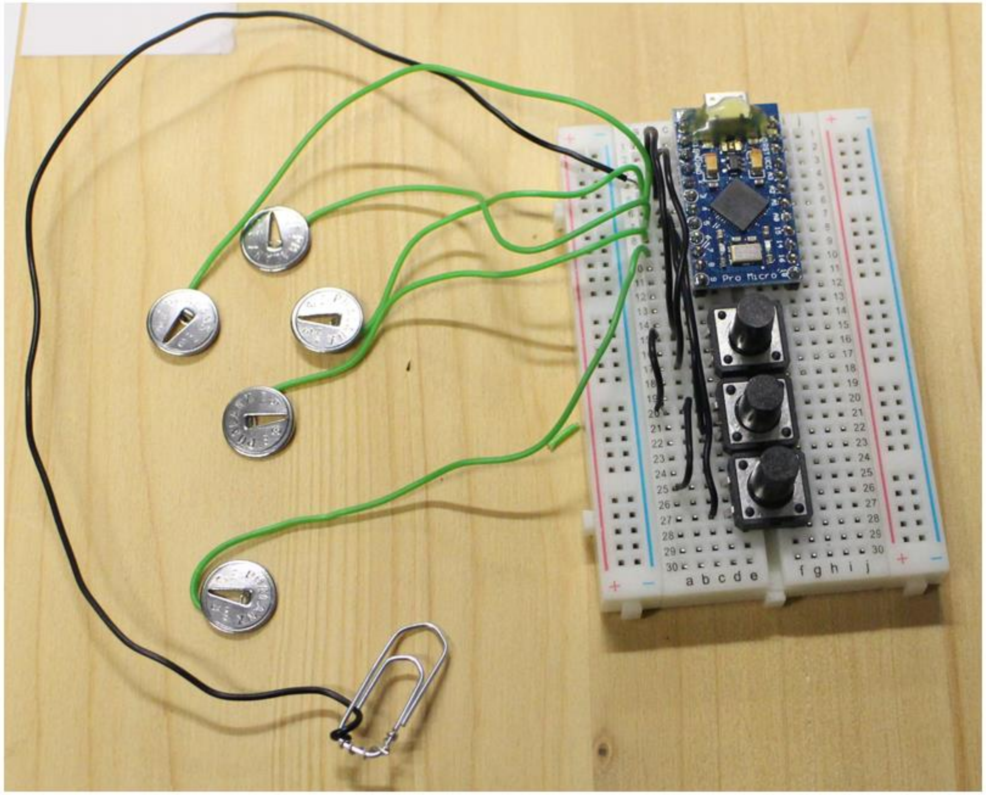
(/SummerSchool/Module4…/ArduinoExamples/MouseClick_KeyInput)



```
void setup() {
  // put your setup code here, to run once:
  pinMode(2,INPUT_PULLUP);
  pinMode(3,INPUT_PULLUP);
  pinMode(4,INPUT_PULLUP);
  pinMode(5,INPUT_PULLUP);
  Mouse.begin();
  Keyboard.begin();
}

void loop() {
   if (digitalRead(2) == LOW) Mouse.press(MOUSE_LEFT); else Mouse.release(MOUSE_LEFT);
   if (digitalRead(3) == LOW) Mouse.press(MOUSE_RIGHT); else Mouse.release(MOUSE_RIGHT);
   if (digitalRead(4) == LOW) Keyboard.print("hello");
   if (digitalRead(5) == LOW) { Keyboard.press(KEY_RETURN);Keyboard.release(KEY_RETURN);}
   delay(200);
}
```

Using external switches

- We could replace the PushButtons with our simple DIY switches:
- If placed well, those switches can be accessible for people with various motor disabilites !



Time for a Break !!

FABI – The flexible assistive button interface

- FABI consists of a microcontroller firmware and a PC GUI
- the PC GUI can be used to assign functions to 6 buttons
- the firmware can store and switch settings

AsTeRICS academy
FABI
flexible assistive button interface
AsTeRICS academy
FABI
flexible assistive button interface

Installing / using FABI

- the FABI frimware is offers more functions (selectable button actions, GUI configuration, saving configurations)
- Open the FabiWare firmware file (/Software/FABI/Source/FabiWare/FabiWare.ino) in the Arduino IDE
- Upload the program to the board
- Start the configuration software /Sofware/FABI/FabiGUI.exe
- if a error is shown you probably need to install the .Net framework, it is available in the folder /Software/SoftwareDevelopment/dotNetFx40_setup.exe

Customizing FABI

In the FABI GUI configuration software:

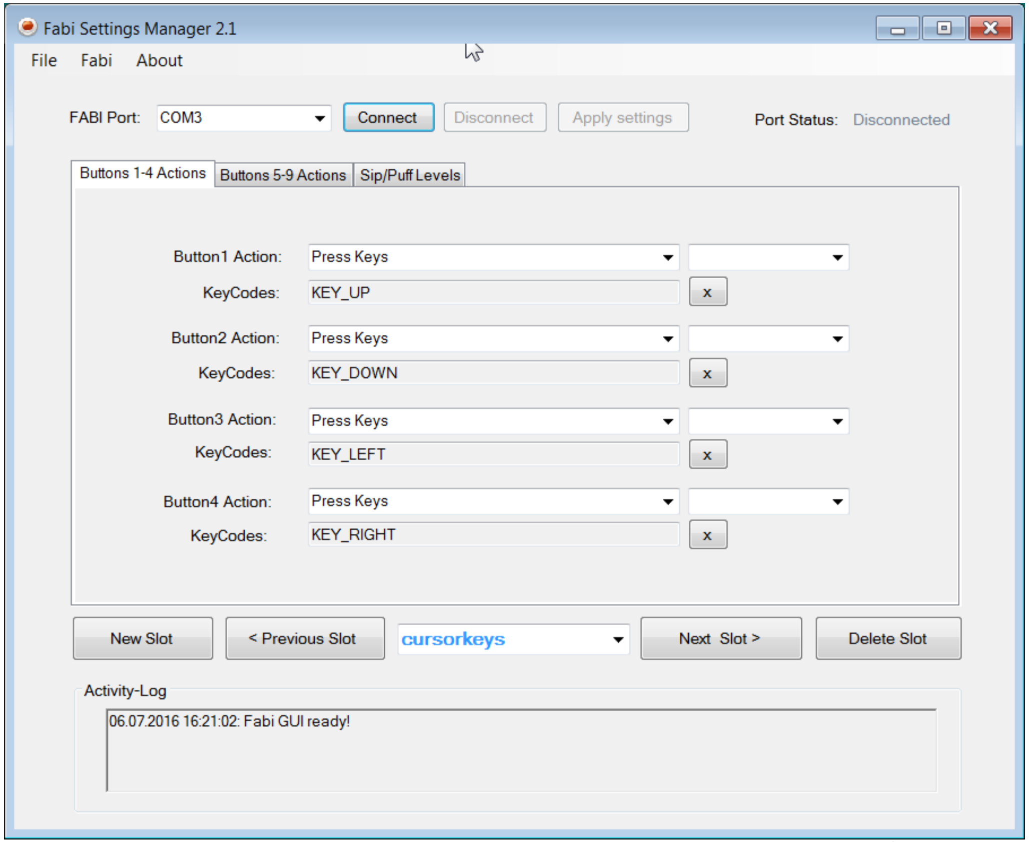
Select the correct COM Port

Press the Connect Button

Edit desired button functions

Press „Apply settings“ to test or „Fabi -> store settings to FABI“ to save the settings

Multipe settings can be saved into configuration „slots“ of desired names



FABI Port: COM3

Port Status: Disconnected

Button1 Action: Press Keys
KeyCodes: KEY_UP

Button2 Action: Press Keys
KeyCodes: KEY_DOWN

Button3 Action: Press Keys
KeyCodes: KEY_LEFT

Button4 Action: Press Keys
KeyCodes: KEY_RIGHT

cursorkeys

Activity-Log

```
06.07.2016 16:21:02: Fabi GUI ready!
```

Applying FABI

- You can now assign different functions to your pushbuttons
- Try to create certain key presses or mouse clicks - you can even move the mouse cursor or type complete words

Project outlook …

You can apply FABI as switch interface for many applications:

creating sounds, playing games, typing, speech synthesis
– all possible via single pushbutton presses !!

Application A: Trigger Sound Samples

By using the Soundplant application we can assign sounds to individual keyboard keys. Using FABI, people with special motor conditions can now play the drums !

We need:

- The Soundplant application, install:

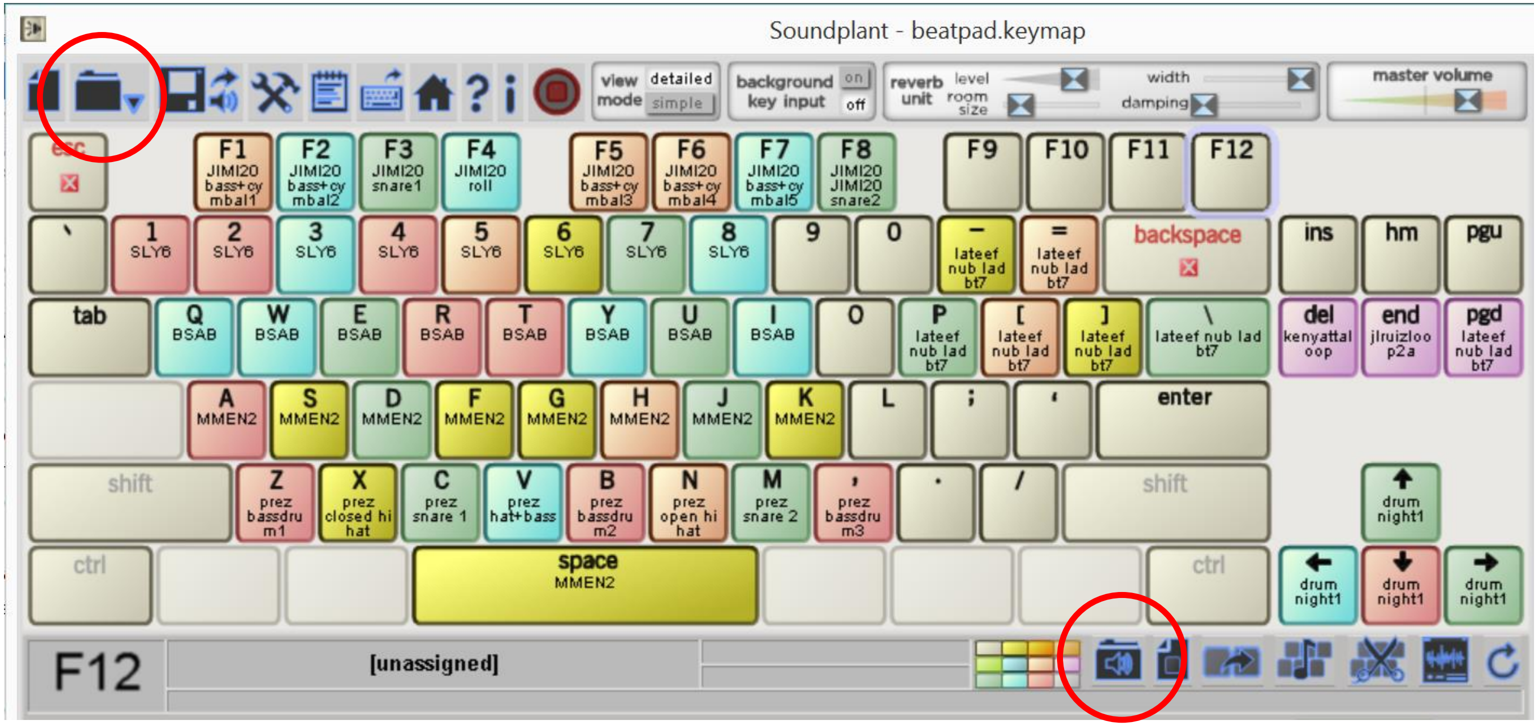
  /Software/3rdParty/Music&Sound/Soundplant/Soundplant42_win_setup.exe

- Keymap file and (optional) additional .wav sound files:

  /Software/3rdParty/Music&Sound/Soundplant/beatpad_keymap/beatpad.keymap

- FABI (with several connected pushbuttons)

Application A: Trigger Sound Samples



- After installing and running Soundplant, select the beatpad.keymap file in the “open” dialog
- Play around with the keys / sounds !
- assign new sounds using the open sample dialog

Application B: Typing and Speech

The „Special Access to Windows“ software (SAW)
is a configurable on-screen keyboard with scanning
Features: You can create own keyboards and select keys with just one or two switches.
Together with „Etriloquist“ we can also create speech !

We need:

- The SAW application, install:

/Software/3rdParty/On-Screen-Keyboards/SpecialAccessForWindows/SAW6.msi

- Keyboard files:

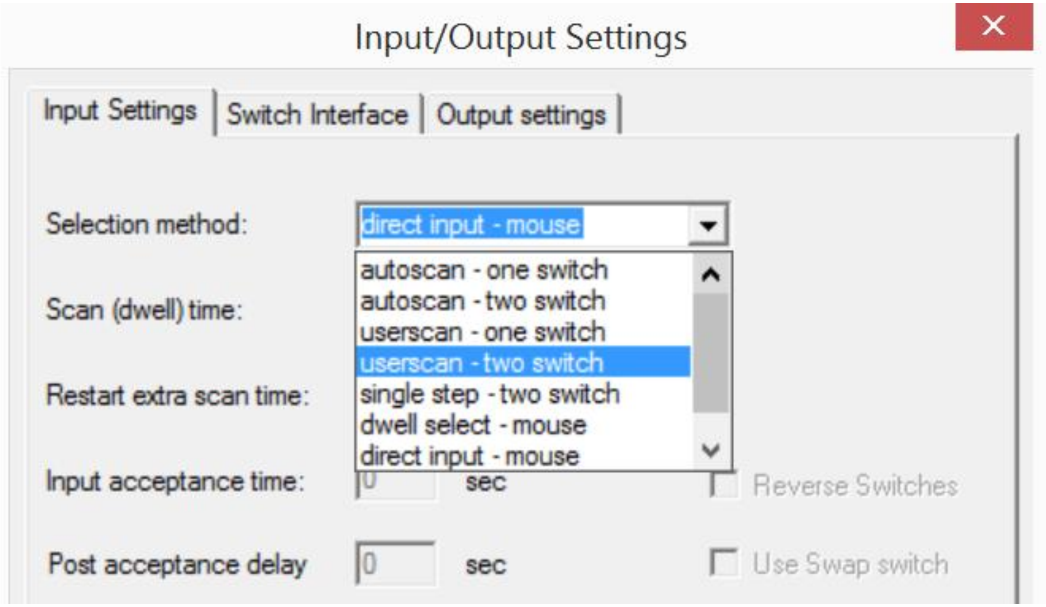
/Software/3rdParty/On-Screen-Keyboards/SpecialAccessForWindows/keyboards

- eTriloquist text-to-speech software, install:

/Software/3rdParty/SpeechCreation/ETriloquist/SetupEQ6300.msi

Application B: Typing and Speech

- Run SAW and open the keyboard (selection set) /keyboards/blade.sss



Input/Output Settings

Selection method: direct input - mouse

autoscan - one switch
autoscan - two switch
userscan - one switch
userscan - two switch
single step - two switch
dwell select - mouse
direct input - mouse

Scan (dwell) time:

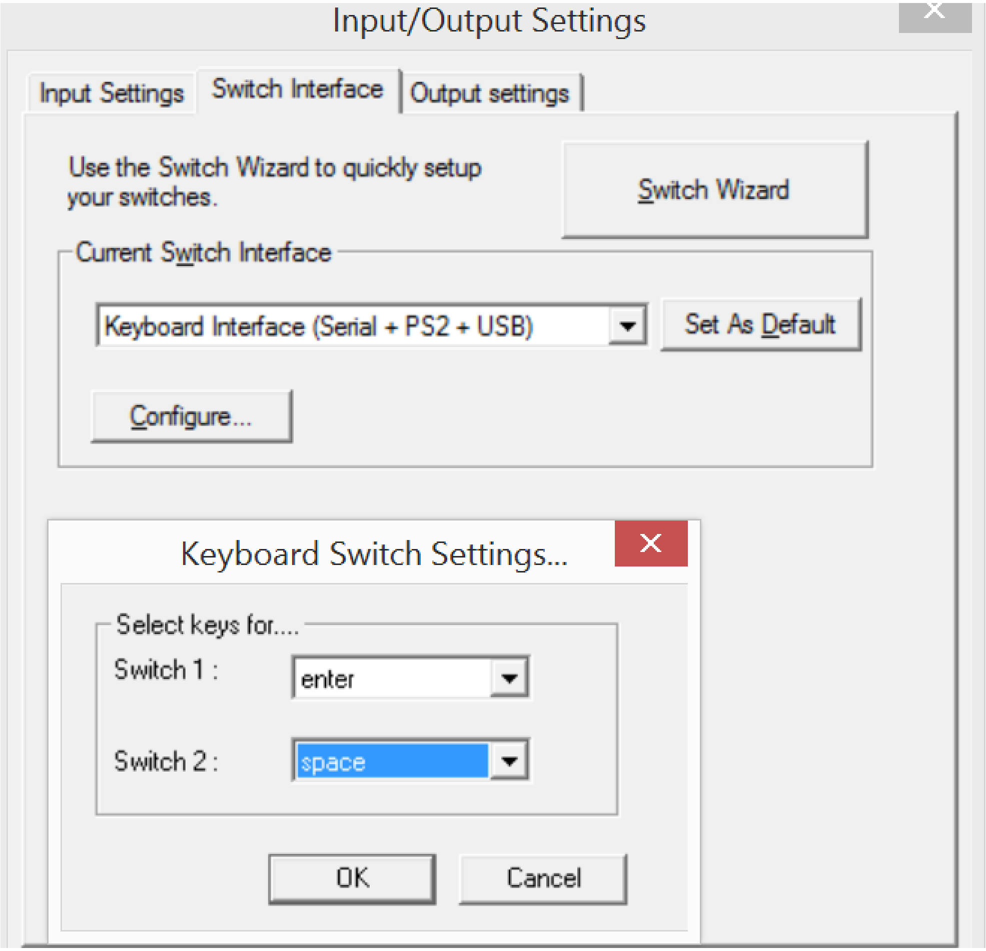
Restart extra scan time:

Input acceptance time: 0 sec □ Reverse Switches

Post acceptance delay 0 sec □ Use Swap switch

- In the settings menu of SAW, choose “Options (Ctrl+E)”
- In the InputSettings tab, select the input method “User scan – two switch”

Application B: Typing and Speech



Input/Output Settings

Use the Switch Wizard to quickly setup your switches.

Current Switch Interface: Keyboard Interface (Serial + PS2 + USB)

Keyboard Switch Settings...

Select keys for....

Switch 1 : enter

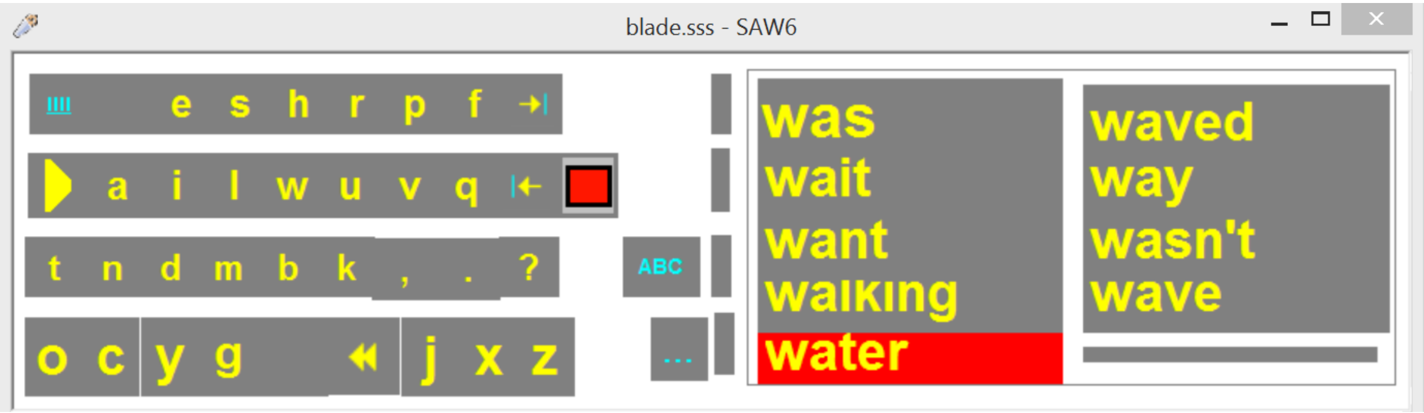
Switch 2 : space

- In the SwitchInterface tab, select “Keyboard Interface” as SwitchInterface
- Then choose two desired keys for the selection of row / column scanning

Here, “enter” and “space” were selected. FABI needs to be configured so that these two keys can be created via the the pushbuttons !

Application B: Typing and Speech

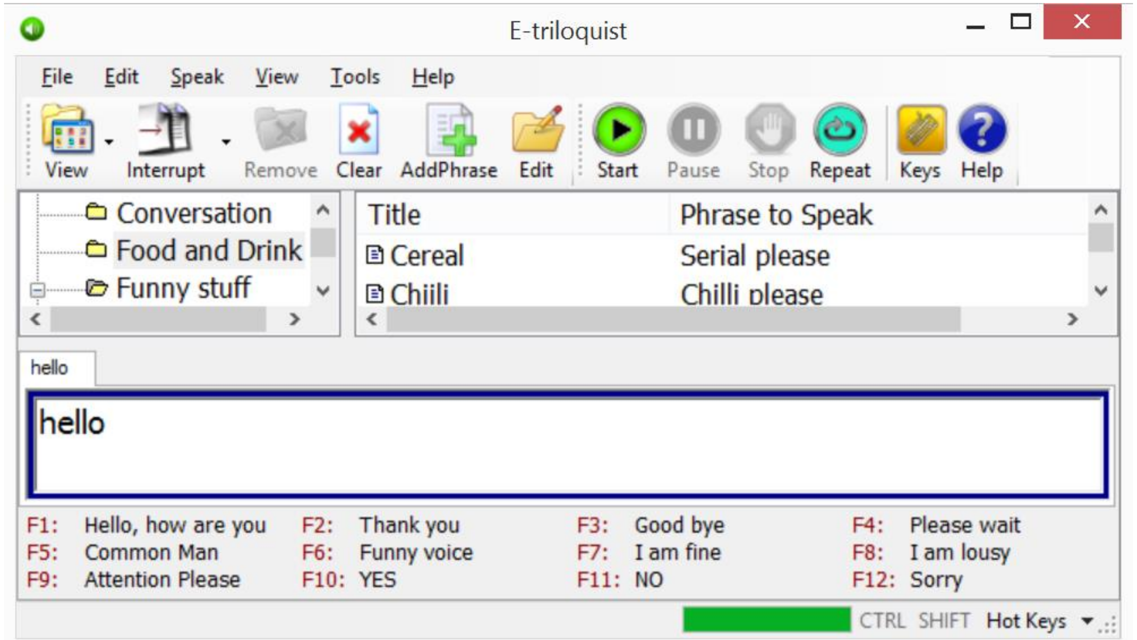
- Now select Settings->Run or press Ctrl+R



- Open an application where you want to type text into - e.g. Notepad or an email program
- Try the keys “enter” and “space” for navigation and selection of the desired keys !
- A word prediction window is available which makes typing more efficient. You can select it with the first key.

Application B: Typing and Speech

- Now start „Etriloquist“. A menu tree and a text input box should appear:



Title	Phrase to Speak
Cereal	Serial please
Chiili	Chilli please

hello

F1:	Hello, how are you	F2:	Thank you	F3:	Good bye	F4:	Please wait
F5:	Common Man	F6:	Funny voice	F7:	I am fine	F8:	I am lousy
F9:	Attention Please	F10:	YES	F11:	NO	F12:	Sorry

- You can now type into the textbox
- When you select the “enter key”, the typed words will be spoken out
- Using the menu tree you can select prepared phrases
- These phrases can be changed and extended

Application C: Play a retro console game

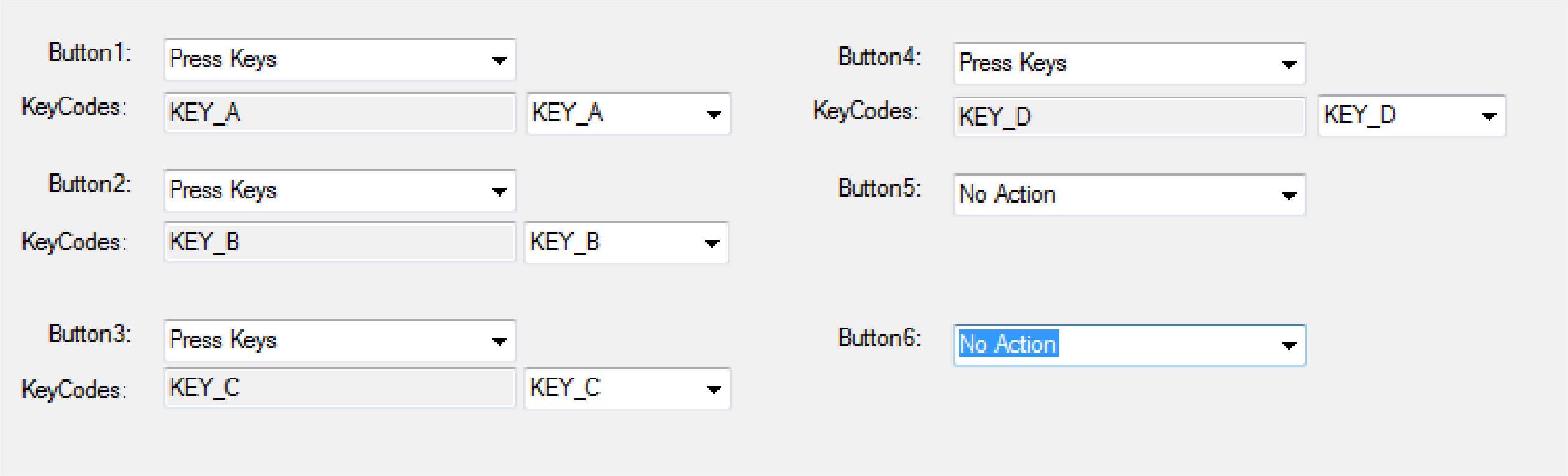
Play a game for the Nintendo-64 gaming-console (from 1996) with FABI as gamecontroller – using an N64 emulator! Even multiplayer is possible!

We need:

- The Project64 Nintendo64 emulator, install:
  /Software/3rdParty/Games&Learning/N64/Project64.zip
- ROM files - we will play “MarioKart”:
  /Software/3rdParty/Games&Learning/MarioKart64.z64
- FABI (with 4 connected pushbuttons)

Application C: Play a retro console game

Configure FABI to press different keys on 4 button inputs:



Button1: Press Keys
KeyCodes: KEY_A | KEY_A

Button2: Press Keys
KeyCodes: KEY_B | KEY_B

Button3: Press Keys
KeyCodes: KEY_C | KEY_C

Button4: Press Keys
KeyCodes: KEY_D | KEY_D

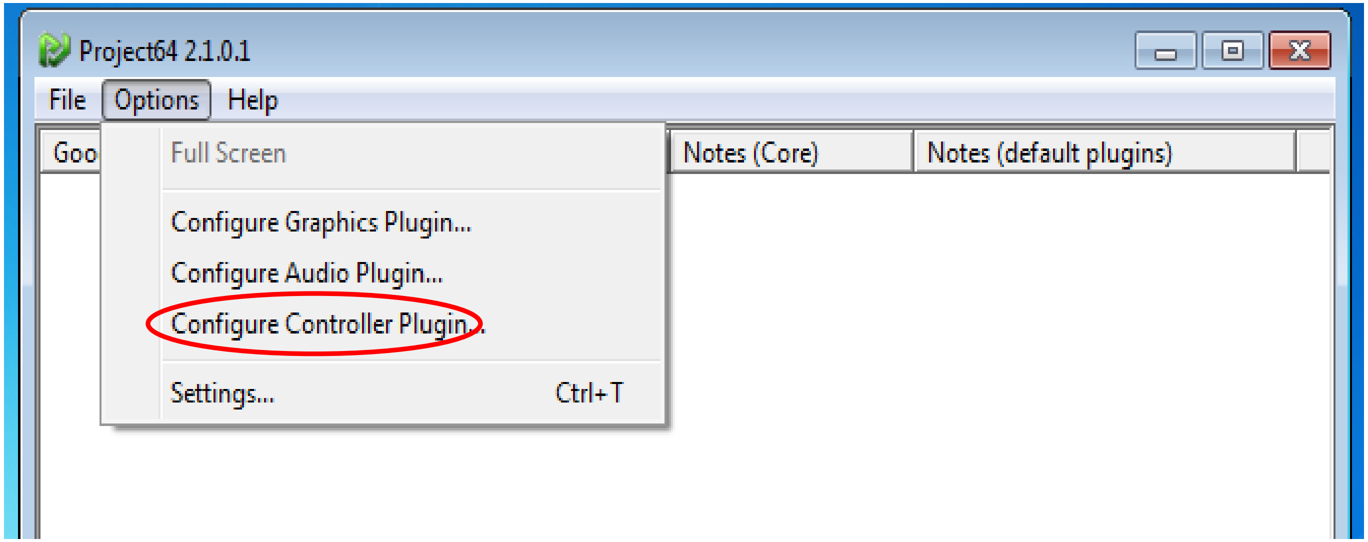
Button5: No Action

Button6: No Action

If you want multiple players, connnect two FABI devices and assign different keys for player 2

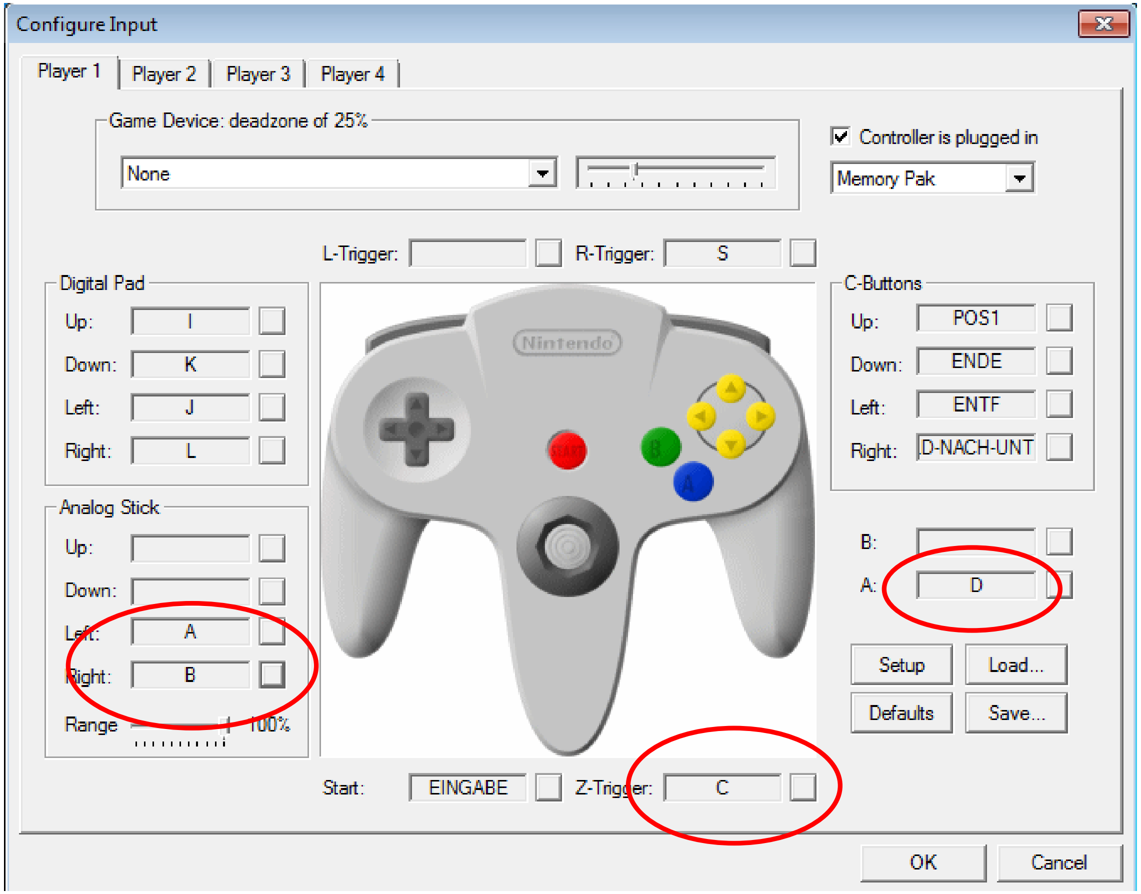
Application C: Play a retro console game

- Start the N64 emulator
- Configure the controller plugin of Project64 to use the previously defined keys (a, b, c and d):



Application C: Play a retro console game

To play MarioKart64, we need the following buttons:



Configure Input

Player 1 | Player 2 | Player 3 | Player 4

Game Device: deadzone of 25%

None

☑ Controller is plugged in

Memory Pak

L-Trigger:

R-Trigger: S

Digital Pad

Up: I

Down: K

Left: J

Right: L

Analog Stick

Up:

Down:

Left: A

Right: B

Range 100%

C-Buttons

Up: POS1

Down: ENDE

Left: ENTF

Right: D-NACH-UNT

B:

A: D

Start: EINGABE

Z-Trigger: C

A

for throttle

Analog stick left

for steering left

Analog stick right

for steering right

Z trigger

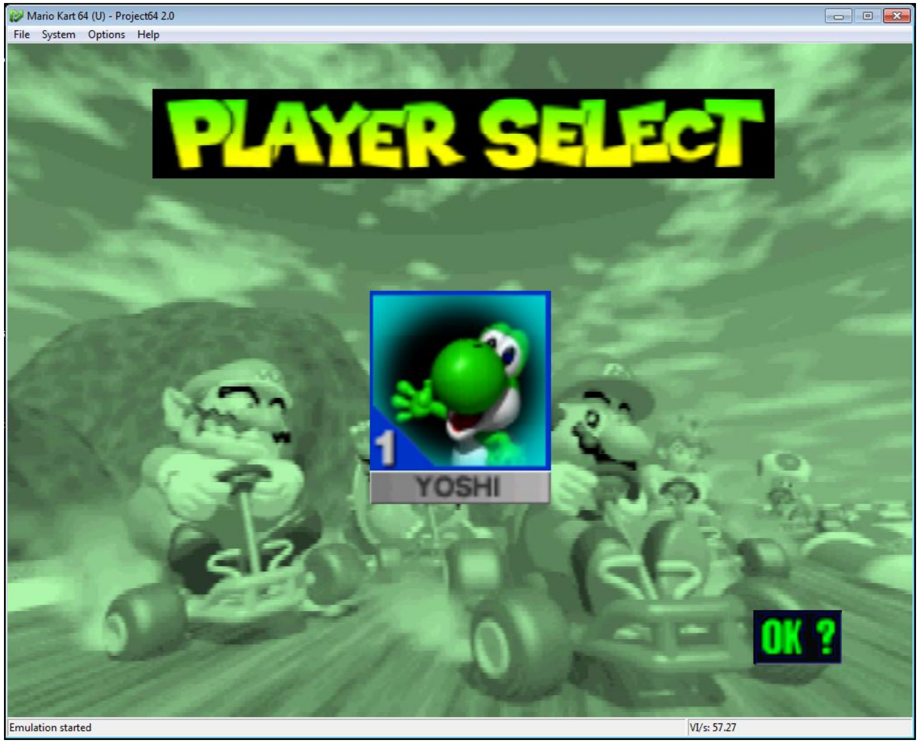
for fire/trigger item

Other games may need more buttons for full game experience

Application C: Play a retro console game

Now you can open the ROM file and enjoy the game !

/Software/3rdParty/Games&Learning/MarioKart64.z64



Application C: more retro games: C64 !

In a similar way, you can play 19
Commodore C64 retro games !
/Software/3rdParty/Games&Learning/C64/Setup_CCS64.msi

First: install and start CCS64

Application C: more retro games: C64 !

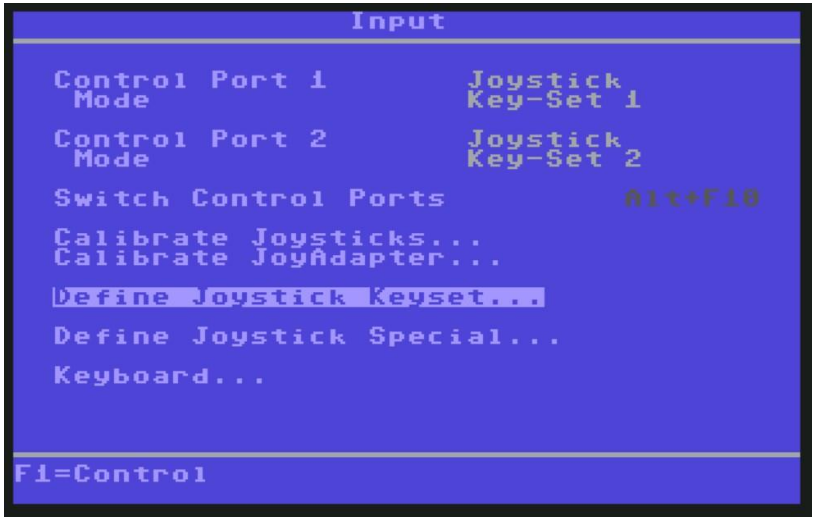
- Use menu “State -> Load” (or Alt+F11) to load one of the provided games

- Press Alt-I to define the Input Keys use Key-Set 1 and Key-Set 2 as control mode for the joysticks

- You might want a 5th button for fire, needed in many games

- During gameplay you can switch Keyset 1 and 2 by pressing Alt+F10

(some games use 1, others 2)



```
                     Input

  Control Port 1            Joystick
   Mode                     Key-Set 1
  Control Port 2            Joystick
   Mode                     Key-Set 2
  Switch Control Ports              Alt+F10
  Calibrate Joysticks...
  Calibrate JoyAdapter...
  Define Joystick Keyset...
  Define Joystick Special...
  Keyboard...

F1=Control
```

Links and resources:

Examples in Arduino, Processing and Fritzing

Fritzing PCB / Schematics Editor:
http://fritzing.org/

Sparkfun Electronics
https://www.sparkfun.com/

MAKE magazine and Hackaday blogs
http://blog.makezine.com/

Arduino Tutorials
http://arduino.cc/en/Tutorial/HomePage

LadyAda‘s Tutorials on Arduino and RaspberriPi
http://www.ladyada.net/learn/arduino/

Arduino Basic Connection Diagrams
http://www.pighixxx.com/abc-arduino-basic-connections/

Thank you for your attention!

All slides and further material (software, movies) is available on our USB stick.

If you are interested in our work or a cooperation please contact us:

office@asterics-academy.net
www.asterics-academy.net

You are welcome to participate
in our hands-on workshop
where we will use the FABI device to build
practical assistive solutions!