AsTeRICS Workshop – HandsOn Part

Welcome !!

- In this Workshop we will use the AsTeRICS System
- First we make sure that everybody has ARE and ACS installed
- Then we will build some easy models together
- After that you can choose an Assistive Technology project and build it together in small groups

The Basics

- AsTeRICS can be downloaded from the official site: http://www.asterics.org
- The latest source code is available at GitHub https://github.com/asterics/AsTeRICS
- ACS is the graphical editor, which sends models with connected plugins to the ARE (the runtime environment)
- Documentation is avaliable:
  - User Manual with step-by-step model creation guide
  - Developer manual for creating new plugins
  - Use **F1** in ACS when plugin is selected to display plugin-help !
- Let‘s start ACS and ARE and have a look !

Some useful plugins – short examples:

- Sensors
  - ComputerVision: FaceTrackerLK
  - Generic: DigitialIn, AnalogIn
  - GUI: ButtonGrid, Slider, Cellboard
  - IntertialMeasurement: Acceleration
  - Simulation: SignalGenerator, Timer

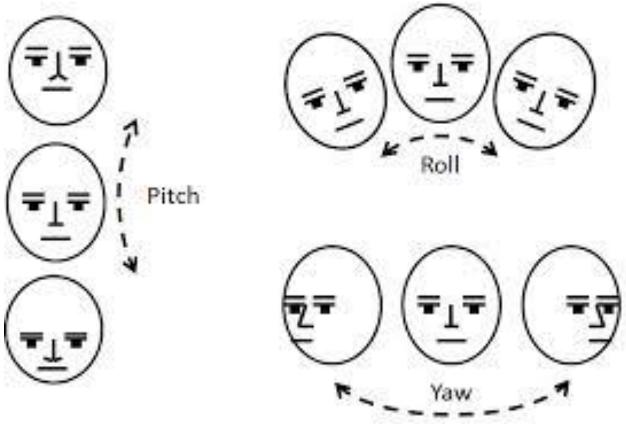
- Actuators
  - Audio: WavePlayer, SpeechSyntheziser
  - GenericOutput: DigitalOut
  - GUI: Oscilloscope, BarDisplay, EventVisualizer
  - HomeControl: FS20, IRTrans
  - InputDeviceEmulation: Mouse, Keyboard

- Processors
  - Audio&Voice: Speechprocessor
  - BasicMath: MathEvaluator, Threshold, Averager, Constant Dispatcher, Comparator, Differentiate / Integrate
  - DataConverters: IntToString
  - SignalShaping: Deadzone, SignalTranslate
  - Event&StringProcessing: StringDispatcher
  - Microcontroller: Arduino
  - OSKA
  - SignalPathways: PathSelector

* Needs dedicated HW or SW / driver

Example 1: Mouse-Control by Head Movements

- In this example we will show how you can control the mouse cursor via head movements
- We use a webcam and the FacetrackerLK sensor plugin (which tracks face movements)
- Mouse clicks are generated via a period of inactivity (no movements)



just another example

Note: you find a similar description in the AsteRICS User manual

Example 1: Mouse-Control by Head Movements

How to provide headtracking-controlled mouse alternatives for computer input

- The x- and y-position of the local mouse will be controlled by the user‘s head movement

Requirements:

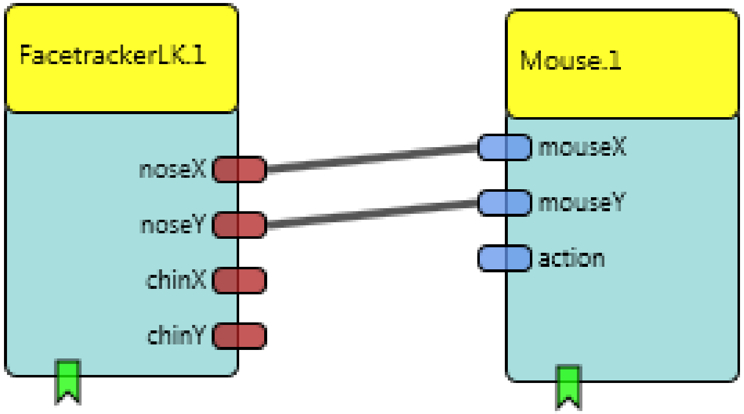
- A webcam

Remarks:

- Camera position directly in front of the user, distance 70-120cm
- No other persons face should be in the field-of-view of the camera

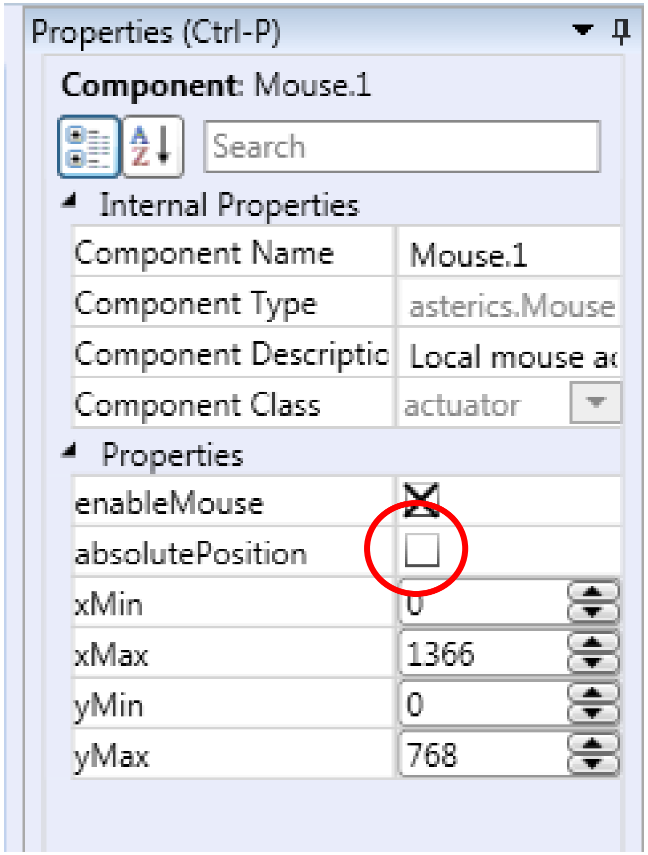
Example 1: Mouse-Control by Head Movements

- Create a new model in the ACS (System → **New Model**)
- Insert the „**FacetrackerLK**“ sensor component
(Components → Sensors → ComputerVision → FacetrackerLK)
- Insert the „**Mouse**“ actuator
(Components → Actuators → Input Device Emulation → Mouse)
- Connect ports „noseX“ to „mouseX“ and „noseY“ to „mouseY“



Example 1: Mouse-Control by Head Movements

- Mouse plugin properties:
  - adjust **xMax** and **yMax**
    to the desired values, e.g. screen resolution
  - enter 0 for xMax and yMax to enable automatic detection of screen resolution
  - **deselect** „absolutePostition“
    this defines that the X and Y input values are relative changes → fits the output of the Facetracker component



Properties (Ctrl-P)

Component: Mouse.1

Search

Internal Properties

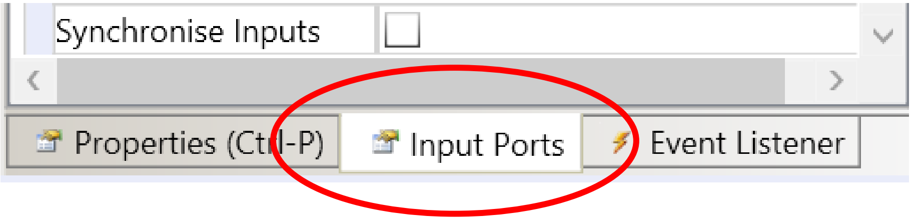
Component Name	Mouse.1
Component Type	asterics.Mouse
Component Descriptic	Local mouse ac
Component Class	actuator

Properties

enableMouse	☒
absolutePosition	☐
xMin	0
xMax	1366
yMin	0
yMax	768

Example 1: Mouse-Control by Head Movements

- Mouse plugin, Input Port rider:

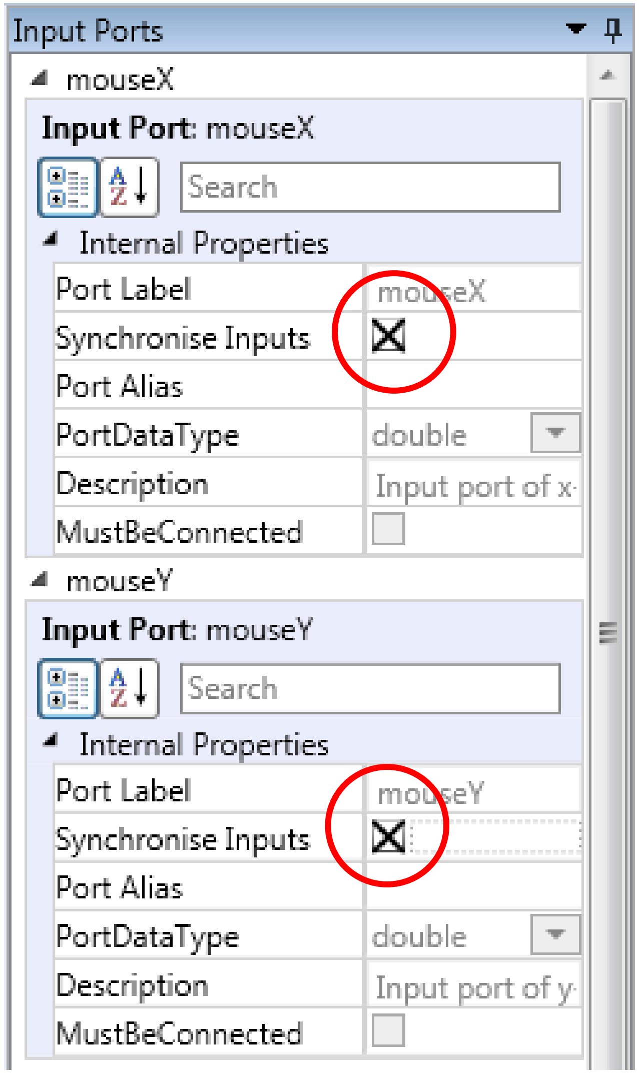


Synchronise Inputs □

  - Select „**Synchronize Inputs**“ option in the Input Port Riders for both inputs (mouseX and mouseY)

    → this will wait for both input coordinates to arrive before the mouse position is updated

- You can try out your model by uploading it to the ARE and clicking „Start Model“



Input Ports

mouseX

Input Port: mouseX

Internal Properties

Port Label	mouseX
Synchronise Inputs	☒
Port Alias	
PortDataType	double
Description	Input port of x
MustBeConnected	□

mouseY

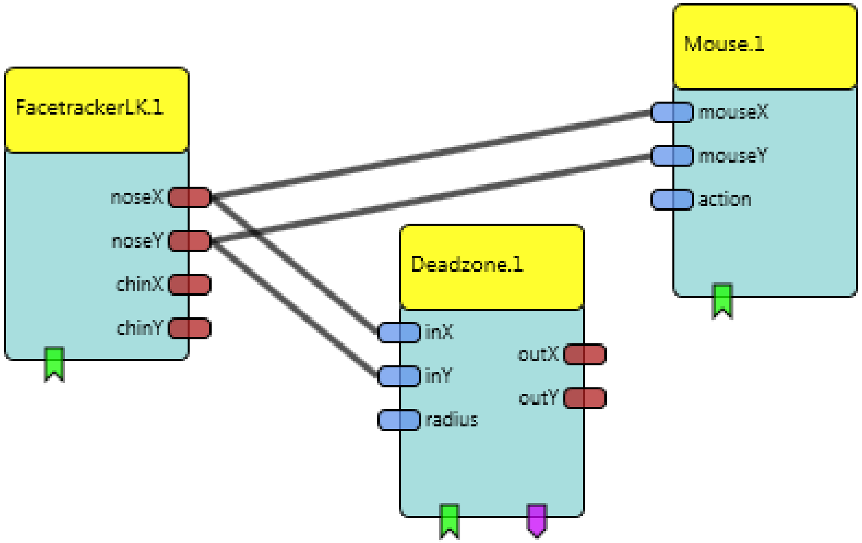
Input Port: mouseY

Internal Properties

Port Label	mouseY
Synchronise Inputs	☒
Port Alias	
PortDataType	double
Description	Input port of y
MustBeConnected	□

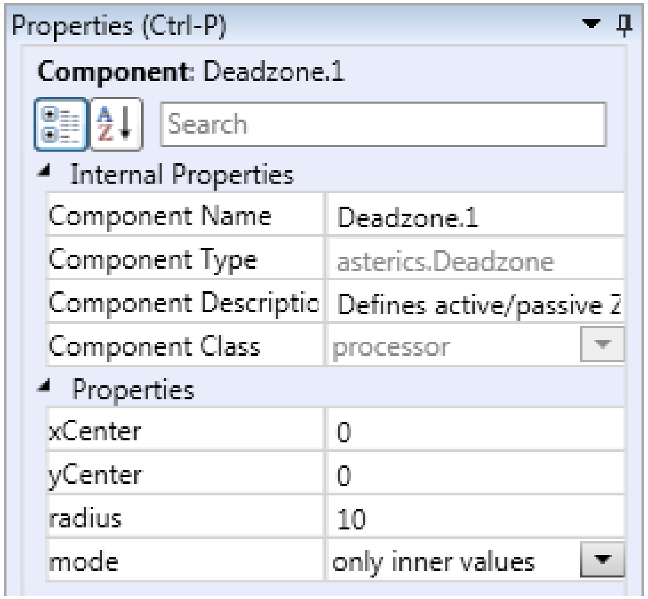
Example 1: Mouse-Control by Head Movements

- No clicking function implemented yet
- → further development: Add **dwell clicking**
- Adding processing component „**Deadzone**“
  (Components – Processors – Signal Shaping – Deadzone)
- Connect noseX/noseY outputs of the FacetrackerLK to the inX/inY inputs of the Deadzone



Example 1: Mouse-Control by Head Movements

- Use the **Deadzone** component to define a desired movement level to start or stop the timing for the dwell click
- Deadzone component fade out x/y signal values in an adjustable range and generate event trigger if the x/y values are in- or outside this range
- Parameter „**radius**“ defines this range → here it is the amount of nose movement
- Leave the radius at the default value of 10



Properties (Ctrl-P)

Component: Deadzone.1

Internal Properties

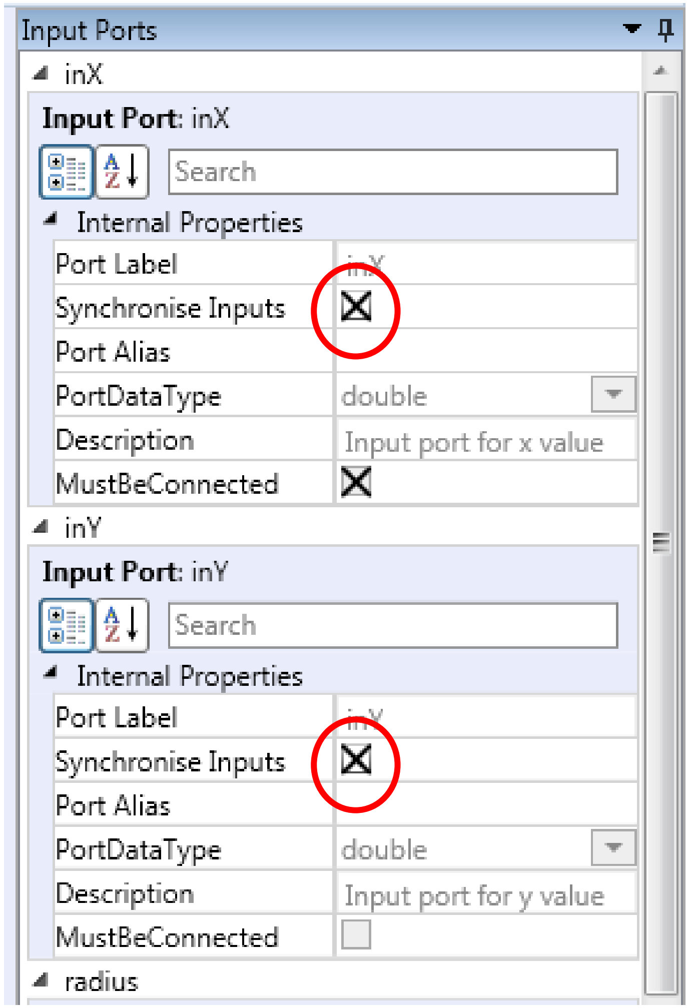
Component Name	Deadzone.1
Component Type	asterics.Deadzone
Component Descriptic	Defines active/passive Z
Component Class	processor

Properties

xCenter	0
yCenter	0
radius	10
mode	only inner values

Example 1: Mouse-Control by Head Movements

- Default value 10 – movement range is set to 10 pixels from previous to current position
- Select „**Synchronize Inputs**“ option in the Input Port Riders of the Deadzone plugin for inX and inY



Input Ports

inX

Input Port: inX

Search

Internal Properties

Port Label	inX
Synchronise Inputs	☒
Port Alias	
PortDataType	double
Description	Input port for x value
MustBeConnected	☒

inY

Input Port: inY

Search

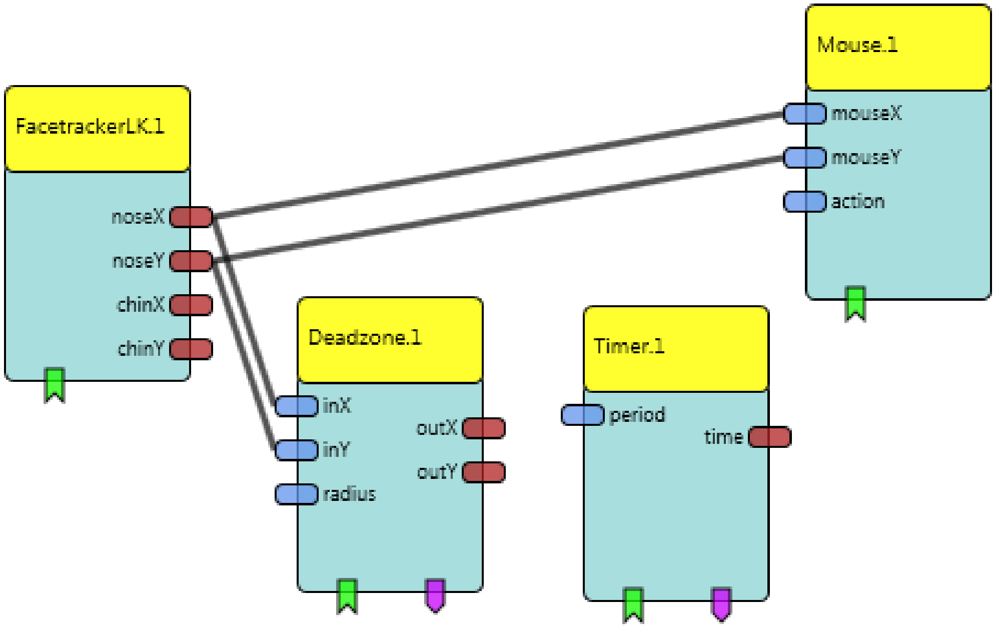
Internal Properties

Port Label	inY
Synchronise Inputs	☒
Port Alias	
PortDataType	double
Description	Input port for y value
MustBeConnected	☐

radius

Example 1: Mouse-Control by Head Movements

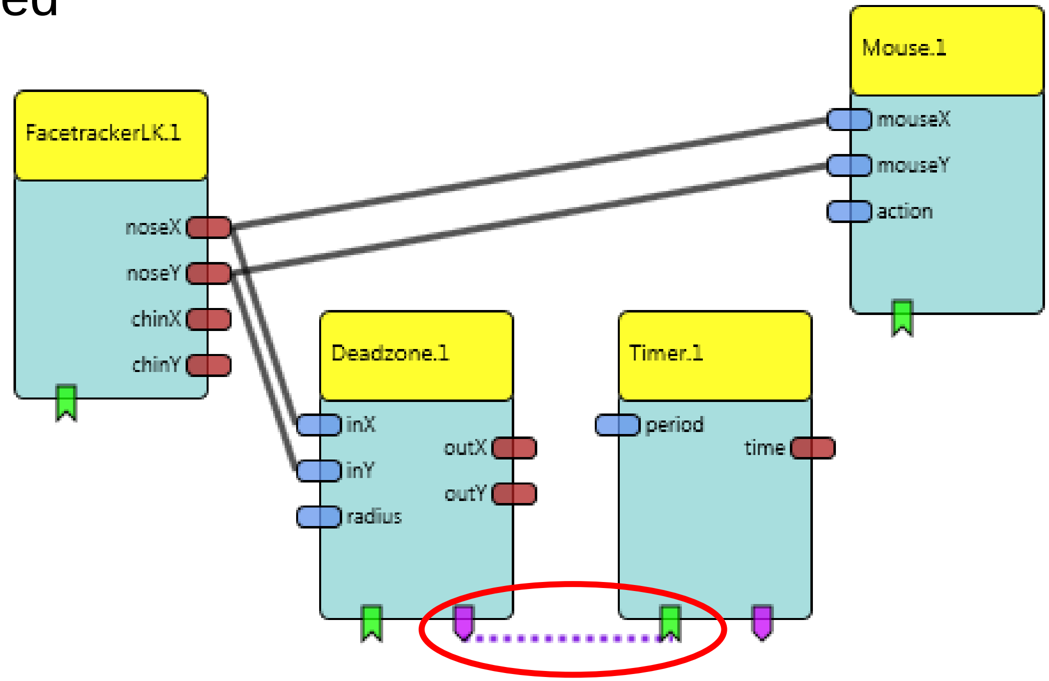
- How can we measure a certain timespan of low movement (to generate a dwell click) ?
- → Insert a **Timer** sensor component (Sensors – Simulation – Timer)
  - this component measures time, generates events if a time period has passed, perfoms timing loops
- Set time-period to 1000 in the components properties



Example 1: Mouse-Control by Head Movements

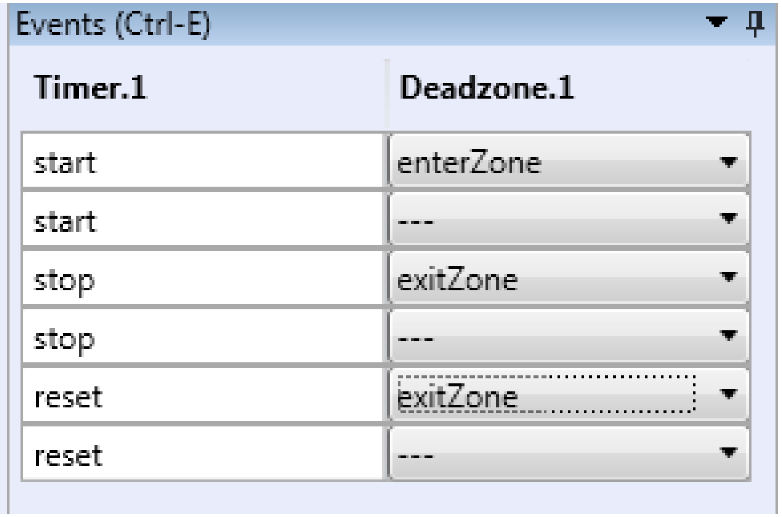
- Connect event trigger port of the Deadzone component (purple) to the event listener of the Timer component (green)
- Click on the event channel (connection line is now purple dotted)

→ Events can be selected from dropdown menu



Example 1: Mouse-Control by Head Movements

- Select „**enterZone**“ event for the „start“ function
- Select „**exitZone**“ event for the „stop“ and „reset“ function
  - These event connections control the Timer components
  - If nose movements stay below selected level of 10 pixels, the Timer is started
  - Else, the Timer is resetted to 0 and stopped
  - If the movement stays low for the full time period, the timer will generate its „**periodFinished**“ event.

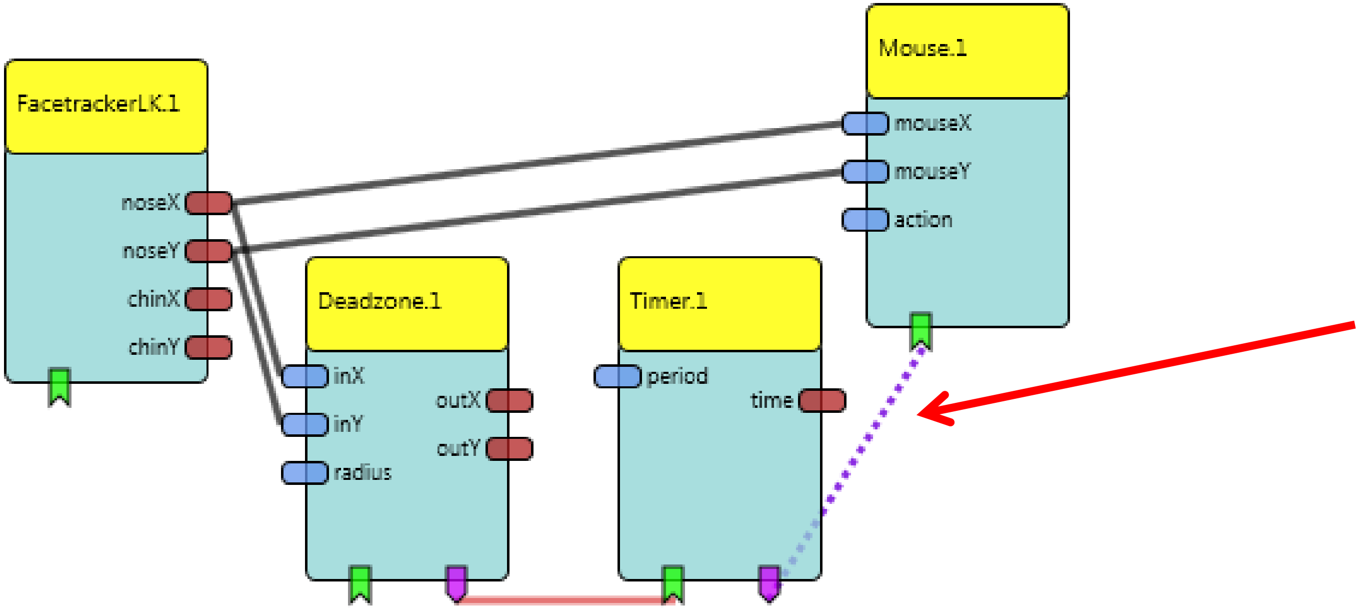


Events (Ctrl-E)

Timer.1	Deadzone.1
start	enterZone
start	---
stop	exitZone
stop	---
reset	exitZone
reset	---

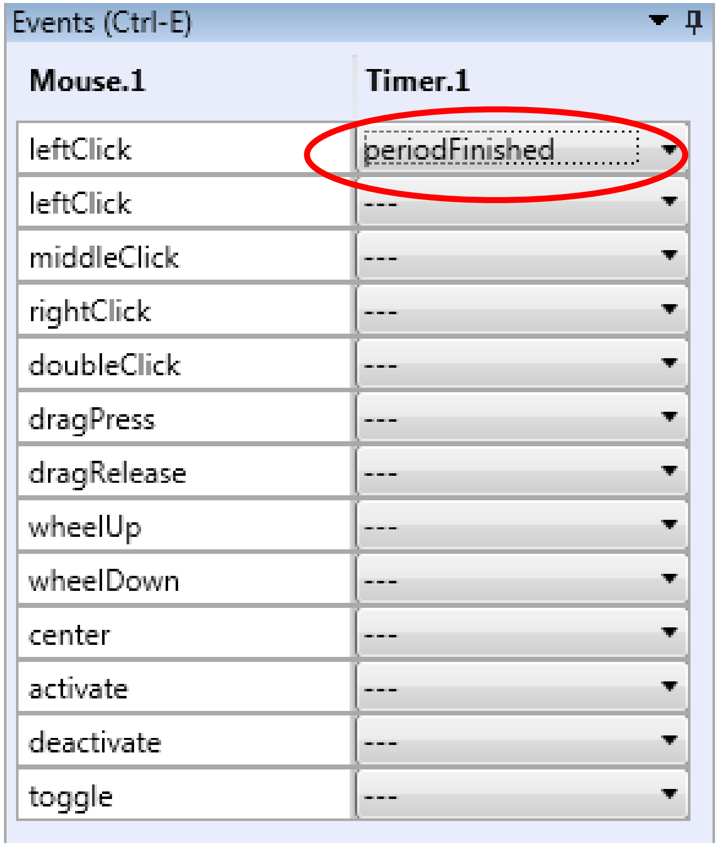
Example 1: Mouse-Control by Head Movements

- Draw a channel from the Timer‘s event trigger port (purple) to the event listener port of the Mouse (green)



Example 1: Mouse-Control by Head Movements

- By clicking on the new channel (line is now purple dotted), the Events can be adjusted
- Assign the „periodFinished“ event to the „leftClick“ function
- After these settings, the model is usable and provides left-click

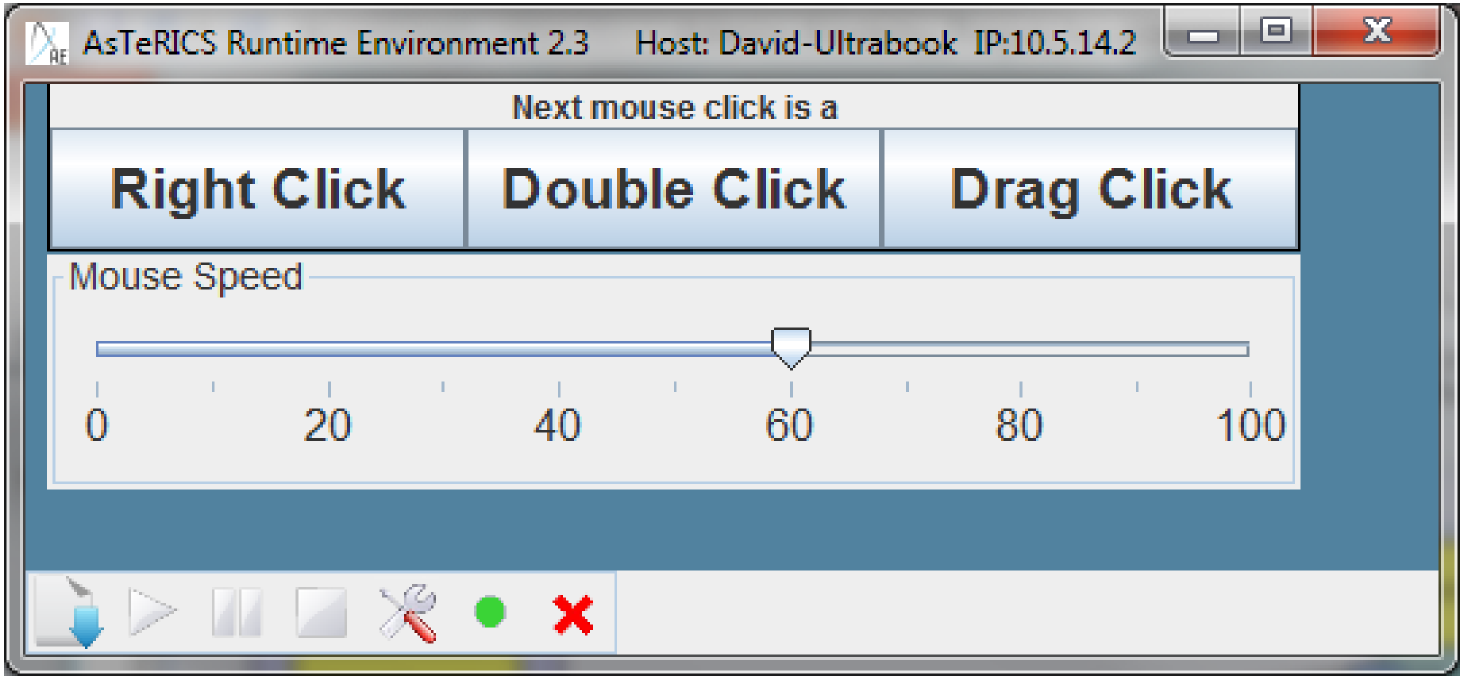


Events (Ctrl-E)

Mouse.1	Timer.1
leftClick	periodFinished
leftClick	---
middleClick	---
rightClick	---
doubleClick	---
dragPress	---
dragRelease	---
wheelUp	---
wheelDown	---
center	---
activate	---
deactivate	---
toggle	---

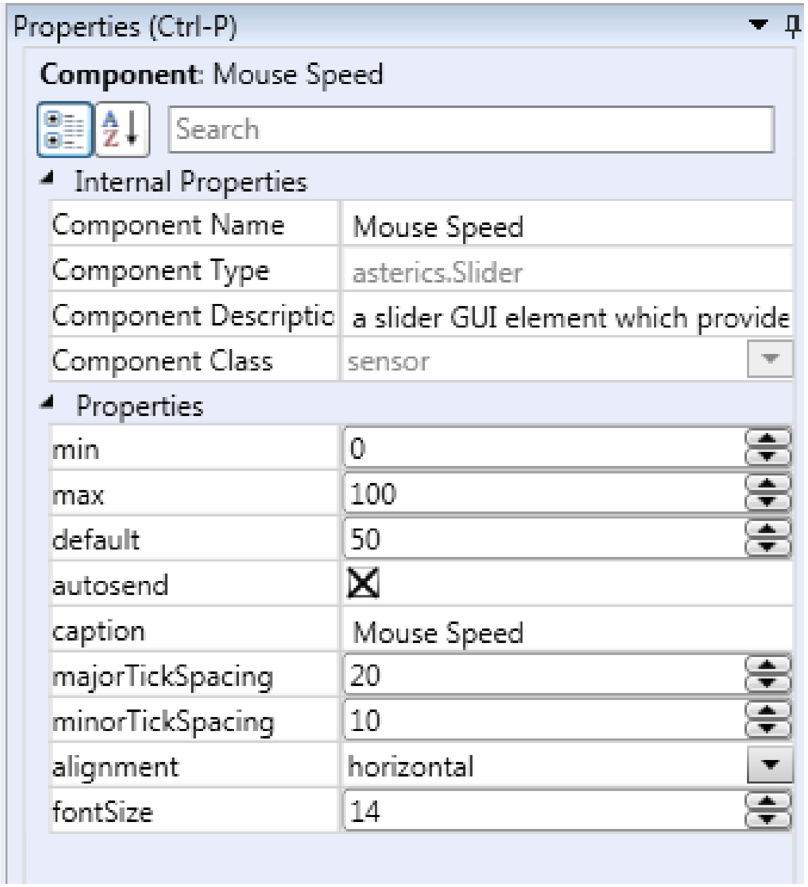
Example 1: Mouse-Control by Head Movements

Include GUI elements for adjustable mouse acceleration and dwell timing as well as different click-actions



Example 1: Mouse-Control by Head Movements

- Add a **Slider** component (Components – Sensors – Graphical User Interface – Slider)
- Slider properties:
  - range of value can be defined (we can leave it at 0-100)
  - set slider’s component name to “Mouse Speed”
  - set minorTickSpacing to “10”



Properties (Ctrl-P)

Component: Mouse Speed

Search

Internal Properties

Component Name	Mouse Speed
Component Type	asterics.Slider
Component Descriptic	a slider GUI element which provide
Component Class	sensor

Properties

min	0
max	100
default	50
autosend	☒
caption	Mouse Speed
majorTickSpacing	20
minorTickSpacing	10
alignment	horizontal
fontSize	14

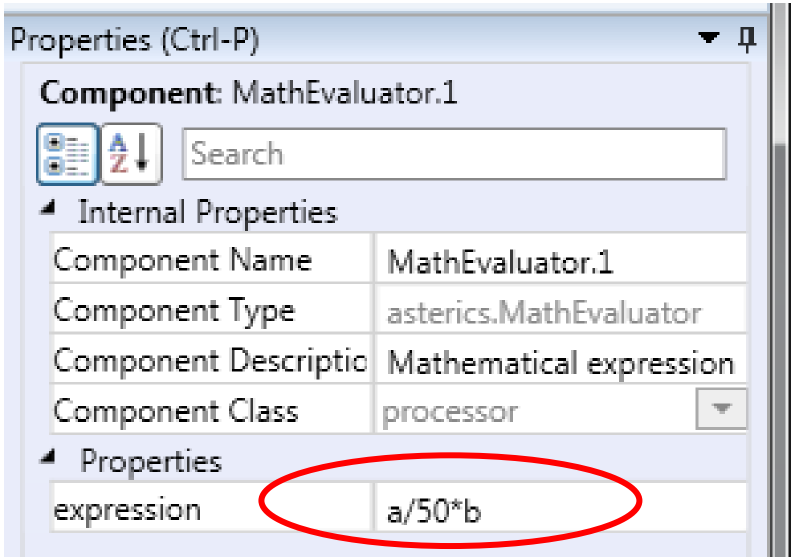
Example 1: Mouse-Control by Head Movements

- To modify the x/y mouse speed with the slider’s values, a **MathEvaluator** processing component is needed (Components - Processors – Basic Math – MathEvaluator)
- First, the x-signal is modified:
  - Delete port connection from noseX to mouseX
  - Draw new port connection from “value” (Slider) to “inA” (MathEvaluator)
  - Draw new connection from noseX (Facetracker) to “inB” (MathEvaluator)
  - Draw a new connection from output port (MathEvaluator) to mouseX input port

Example 1: Mouse-Control by Head Movements

- MathEvaluator properties:
  - Adjust “**expression**” property of the MathEvaluator – this defines what will be done with the inputs – in our case we will multiply inA and inB
  - Slider position <50 shall slow down mouse speed, slider positions >50 shall increase mouse speed

→ a/50*b



Properties (Ctrl-P)

Component: MathEvaluator.1

Search

Internal Properties

Component Name	MathEvaluator.1
Component Type	asterics.MathEvaluator
Component Descriptio	Mathematical expression
Component Class	processor

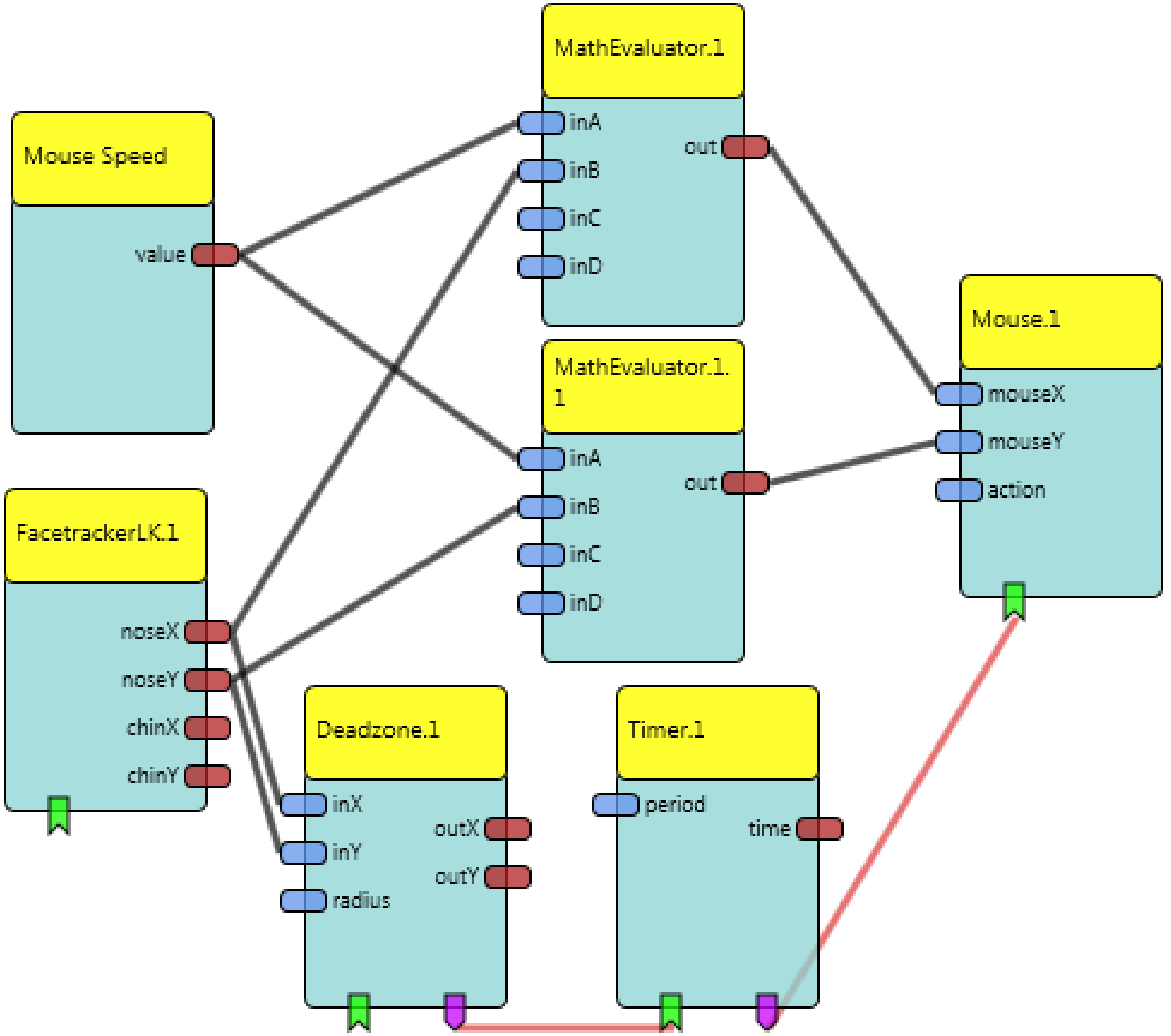
Properties

expression	a/50*b

Example 1: Mouse-Control by Head Movements

- For modifying the y-signal, copy and paste (Ctrl+C & Ctrl+V) the MathEvaluator
- New connections as for the x-direction:
  - Delete port connection from noseY to mouseY
  - Draw new connection from “value” (Slider) to “inA” of the second MathEvaluator
  - Draw new connection from noseY (Facetracker) to “inB” of the second MathEvaluator
  - Draw a new connection from output port of the second MathEvaluator to mouseY input port

Example 1: Mouse-Control by Head Movements



Example 1: Mouse-Control by Head Movements

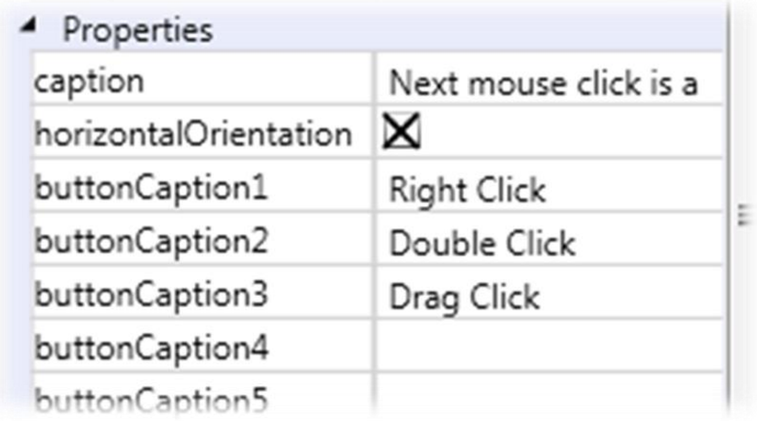
Adding different mouse click activities via GUI by

- adding a **ButtonGrid**
  - to select next click type
- informing mouse element about the next desired mouse click
  - by sending an “action string” to the Mouse element

    Action strings contain commands which are understood by a number of specialized actuator elements. These string contain the addressed component and the desired command
    e.g. “@MOUSE:nextclick,right”

Example 1: Mouse-Control by Head Movements

- Add the **ButtonGrid** component (Component – Sensors – Graphical User Interface – ButtonGrid)
- ButtonGrid properties:
  - Set “buttonCaption” properties of button 1, 2 and 3 to
    - RightClick
    - DoubleClick
    - DragClick
  - Choose “horizontalOrientation” for the ButtonGrid
  - Set a desired caption, e.g. “Next click is a”



Properties

caption	Next mouse click is a
horizontalOrientation	☒
buttonCaption1	Right Click
buttonCaption2	Double Click
buttonCaption3	Drag Click
buttonCaption4	
buttonCaption5	

Example 1: Mouse-Control by Head Movements

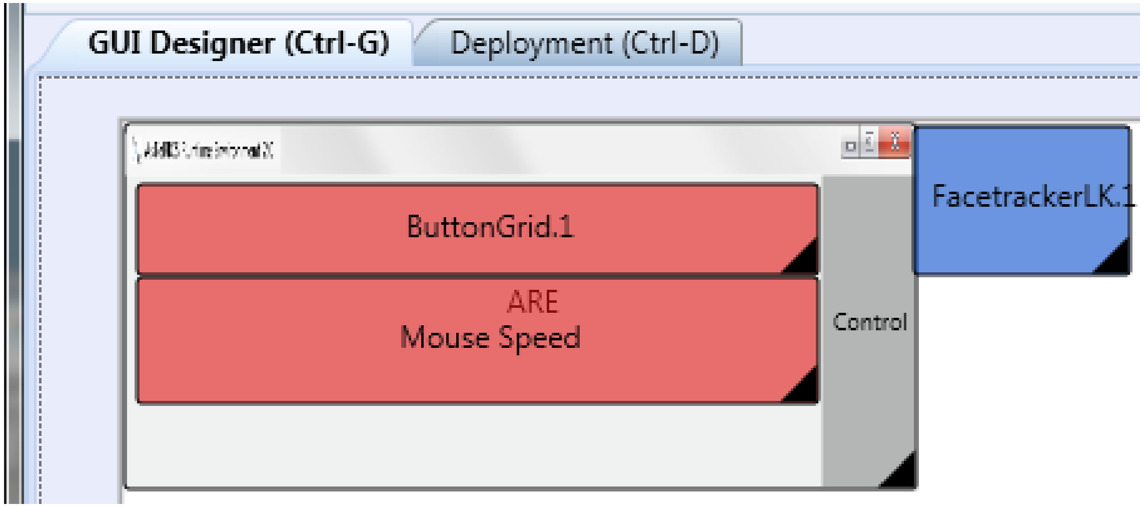
- Add the **StringDispatcher** component (Component – Processors – Event and String Processing – StringDispatcher)
  - Translates incoming events into outgoing strings
  - If buttons are pressed, desired actions strings are generated for the Mouse components
- Connect event trigger port of ButtonGrid (purple) to the event listener port of the StringDispatcher (green)
- Click on the event channel and attach
  - button1 to dispatchSlot1
  - button2 to dispatchSlot2
  - button3 to dispatchSlot3

Example 1: Mouse-Control by Head Movements

- Define the strings for slot1-slot3 in the StringDispatcher properties:
  - slot1(button1): “@MOUSE:nextclick,right”
  - slot2(button2): “@MOUSE:nextclick,double”
  - slot3(button3): “@MOUSE:nextclick,drag”
- Connect output port of the StringDispatcher to the “action” input port of the mouse

Example 1: Mouse-Control by Head Movements

- Change to GUI Designer tab to define a desired position for the ButtonGrid
  - Possible positions:

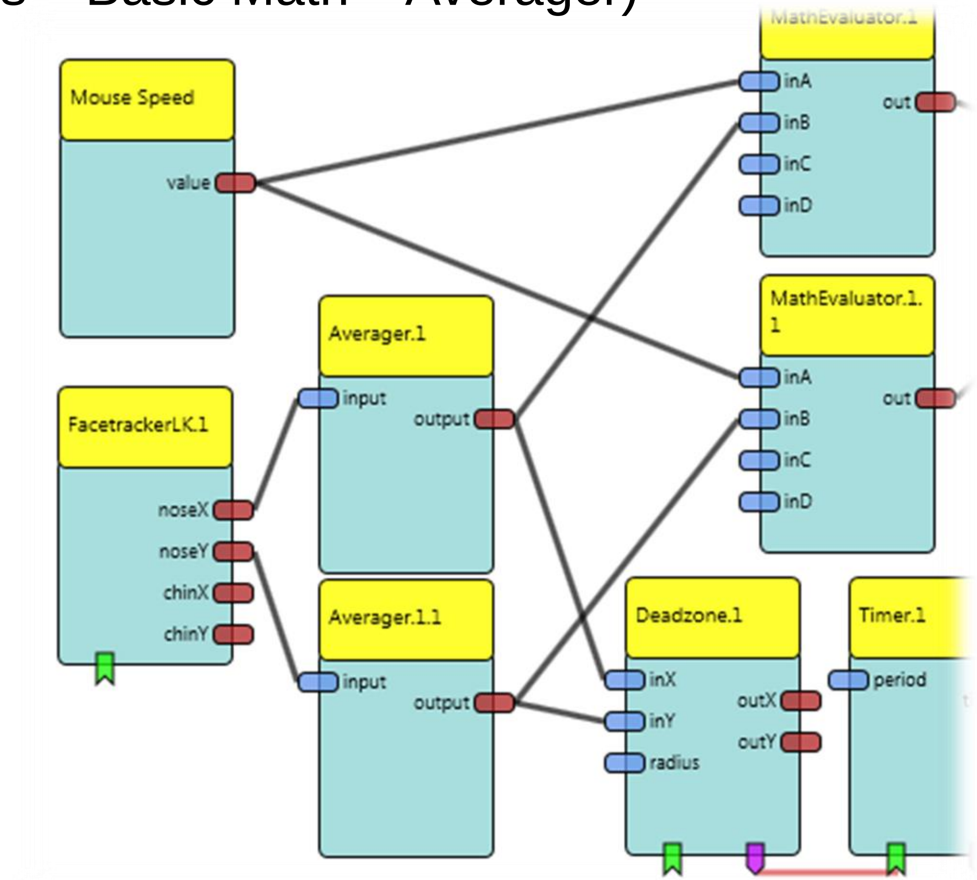


- ”Upload Model” and “Start Model” to try it out!

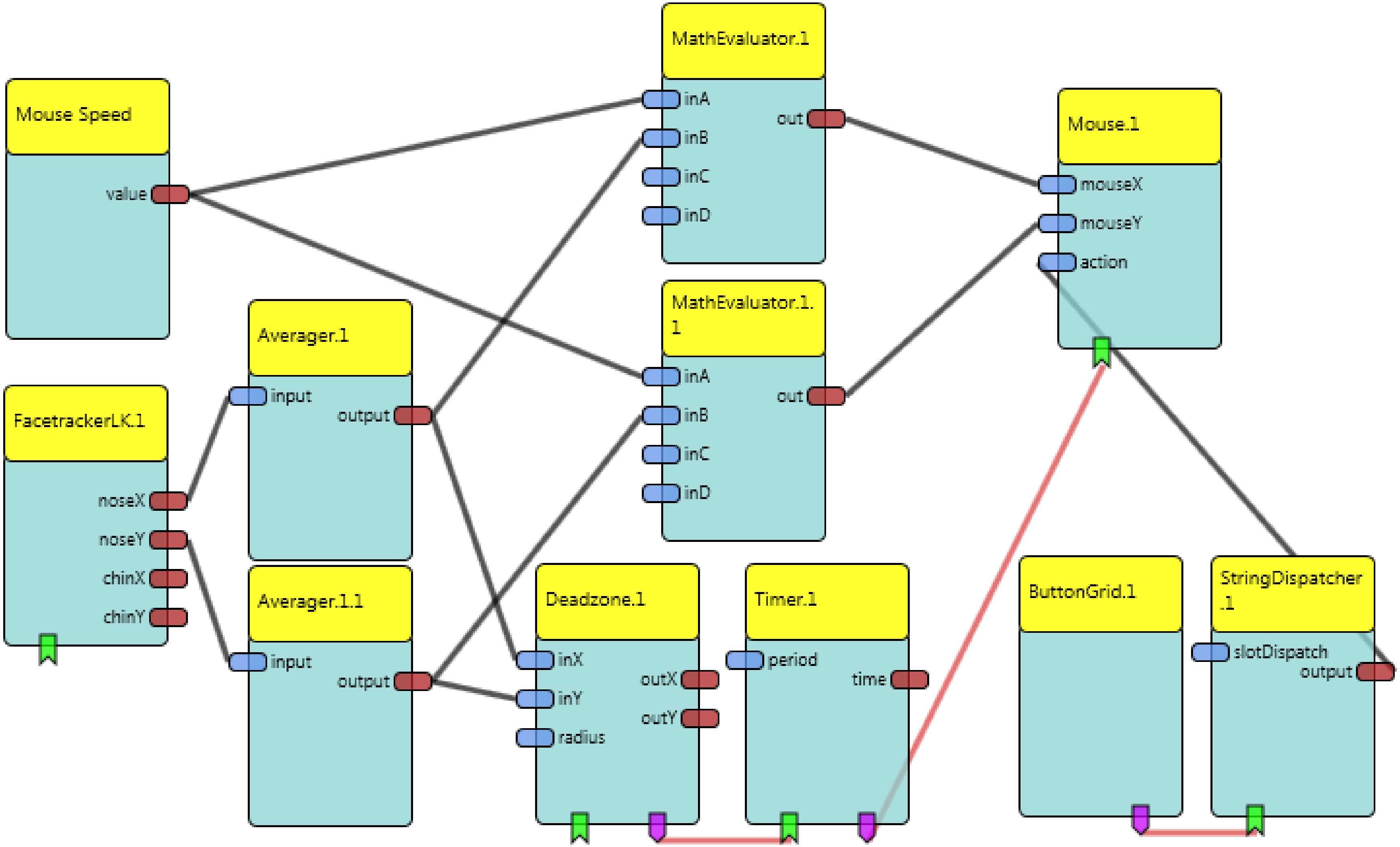
Example 1: Mouse-Control by Head Movements

Further improvement tip

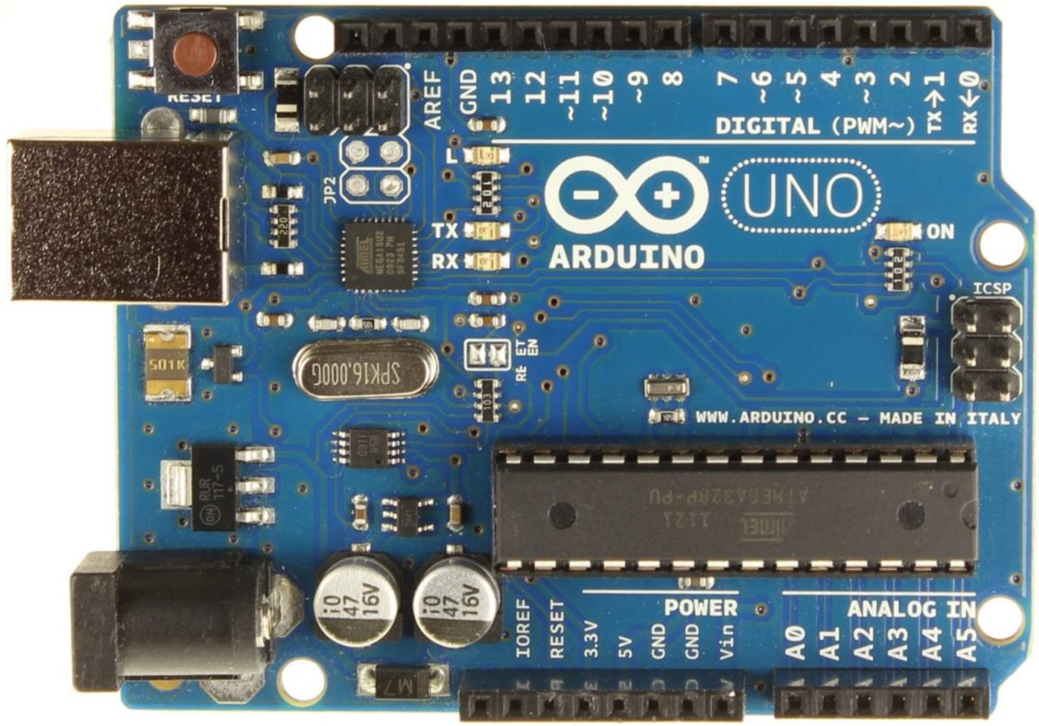
- If the mouse cursor is jittering a lot, use an averager (Components – Processors – Basic Math – Averager) with a BufferSize of 5 after Facetracker Outputs



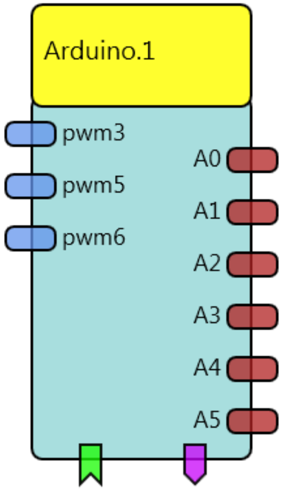
Example 1: Mouse-Control by Head Movements



Example 2: Using the Arduino for interfacing



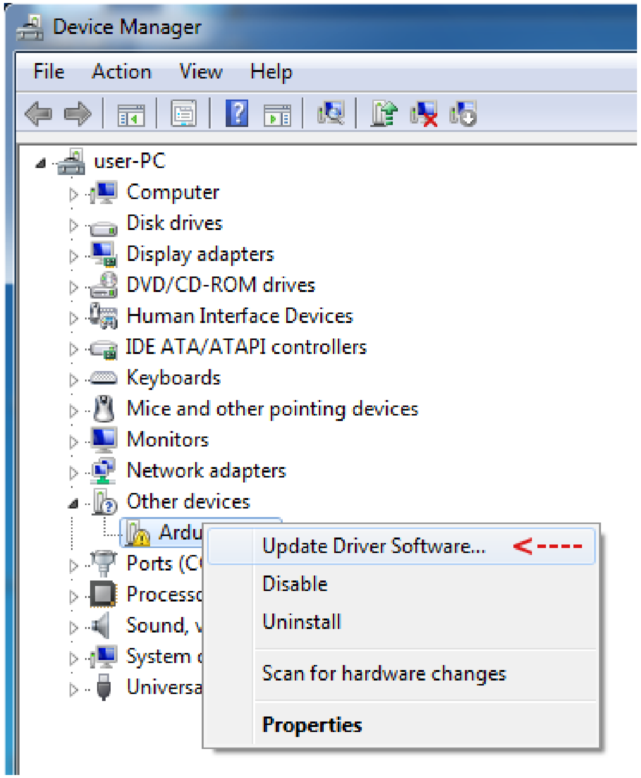
- Based on Atmel ATmega328
- 6 Analog Input Pins
- 14 Digital I/O Pins
- 32 KB Flash Memory
- 2 KB SRAM
- 1 KB EEPROM
- 16 MHz Clock Speed



- **Special Firmware** supports ARE – plugin
- Processors → Microcontroller Interface → Arduino

Example 2: Using the Arduino for interfacing

- Installation of **COM Port**:
  - Attach Arduino to USB Port
  - Driver must be updated in Device manager to get COM Port
  - Select location of „Arduino.inf“ (folder: CIMs/Arduino/driver)
  - Click „Install this driver software anyway“



- For Windows 8:

  Reboot without Driver Signature:

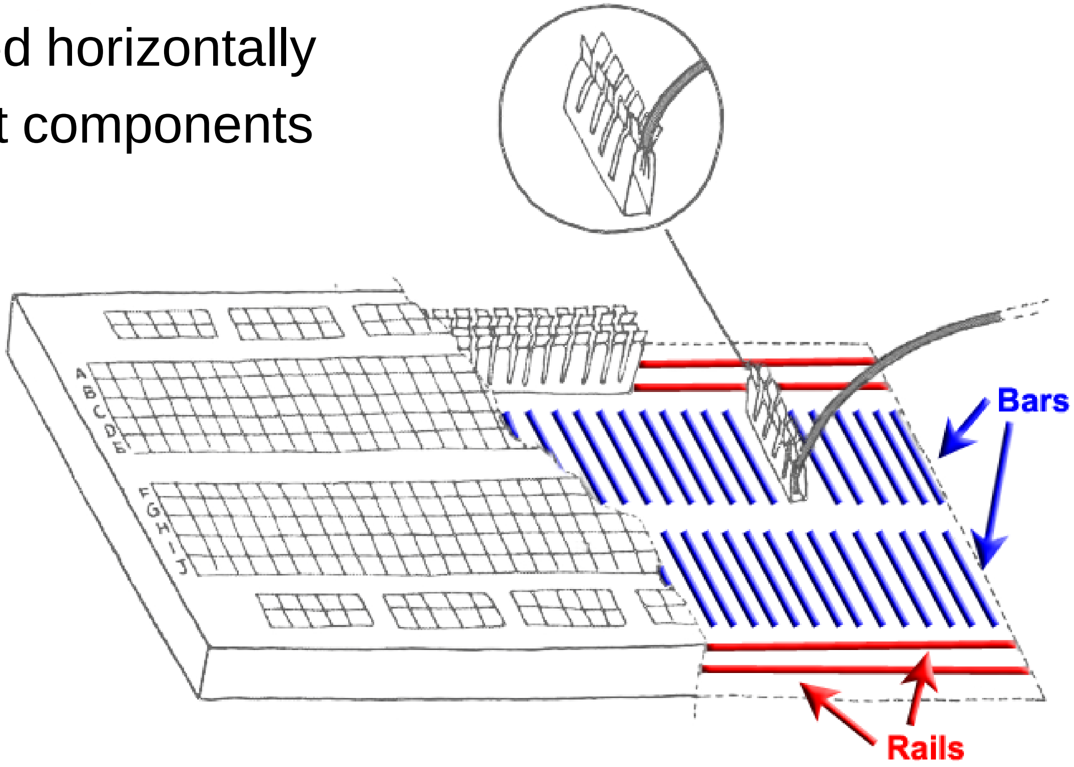
  Enter „shutdown /r /o /f /t 00“ in shell-window

  Choose: troubleshooting -> advanced options
  -> startup settings -> reboot without driver signature enforcement

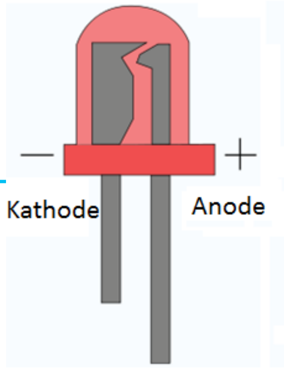
See: http://www.makeuseof.com/tag/how-can-i-install-hardware-with-unsigned-drivers-in-windows-8/

Example 2: Using the Arduino for interfacing

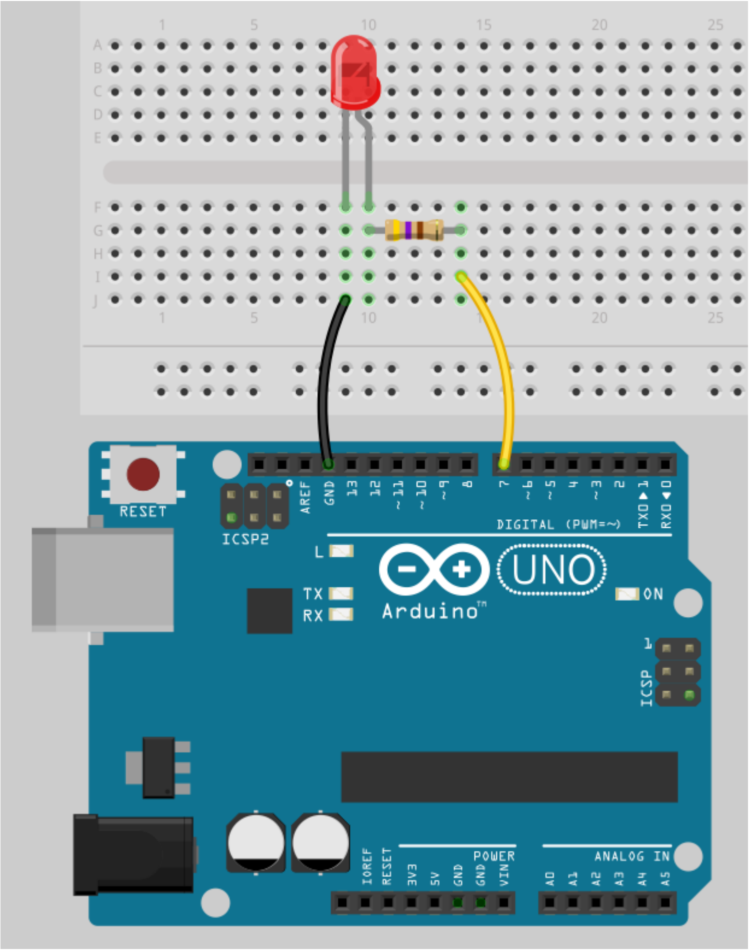
- **Breadboard** connections
  - allow flexible building test circuits
  - the bars are connected vertically
  - the rails are connected horizontally
  - use cables to connect components and Arduino Pins

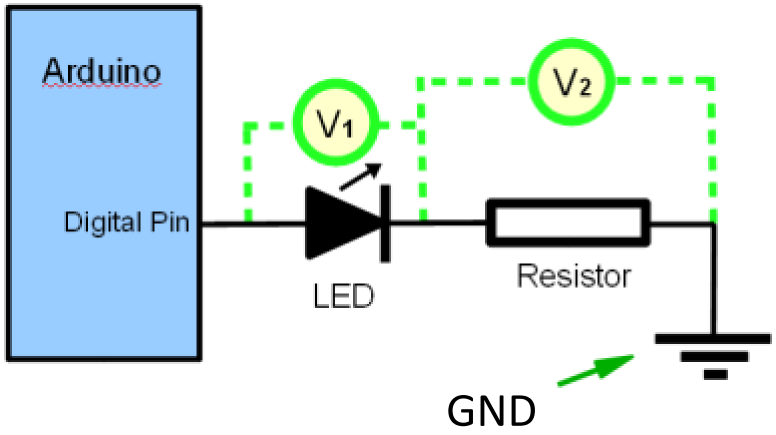


Example 2: Connecting a LED to Arduino



- Connect the **Anode** of the Led to Pin7 (positive lead, usually the long leg)
- Connect the **Kathode** of the Led to a 470 Ohm resistor
- Connect the resistor‘s other end to GND (0 Volt)

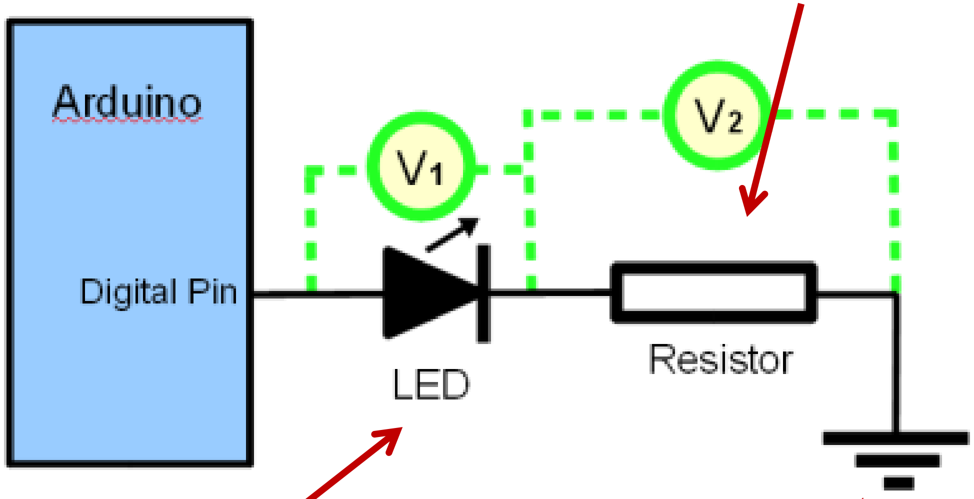




Example 2: Connecting a LED to Arduino

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

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

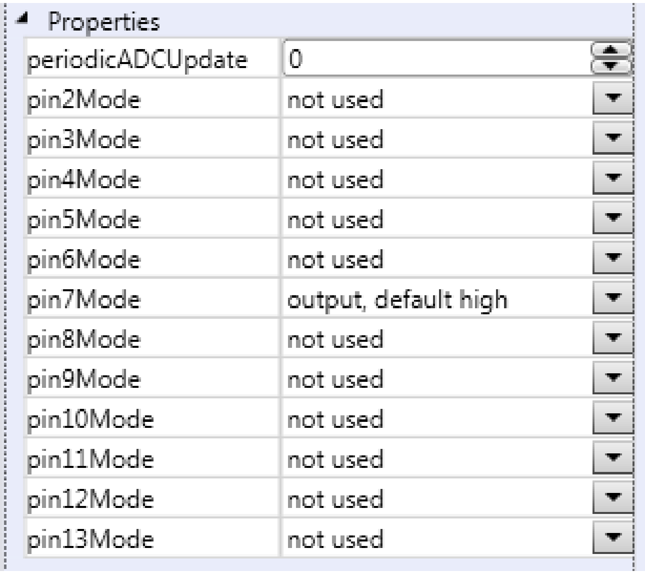


LED Kathode
connected to
lower potential
(usually Ground)

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

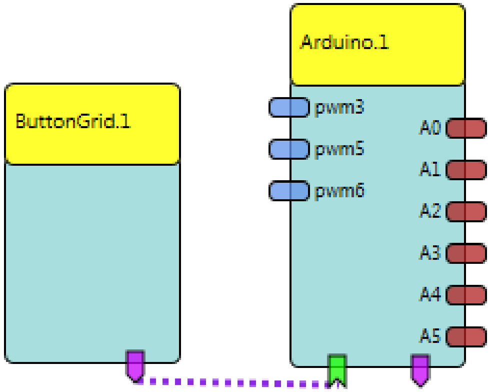
Example 2: Connecting a LED to Arduino

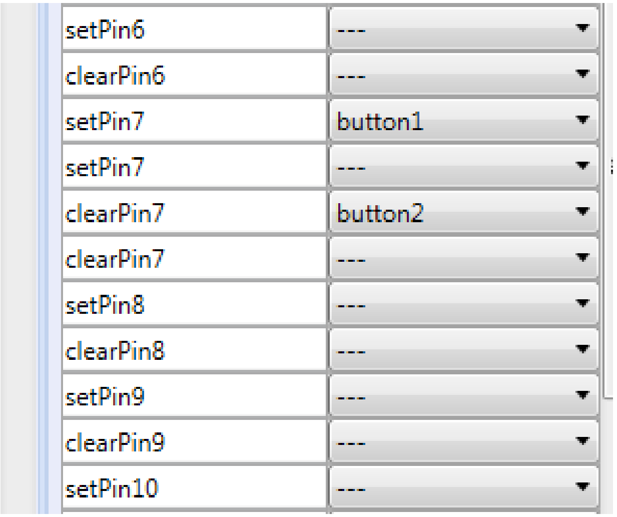
- In the Arduino processor plugin‘s properties:
  - Define pin7Mode „output, default high“ or „output, default low“
  - Use Events Listeners to set output high or low (turn on/off)



Properties

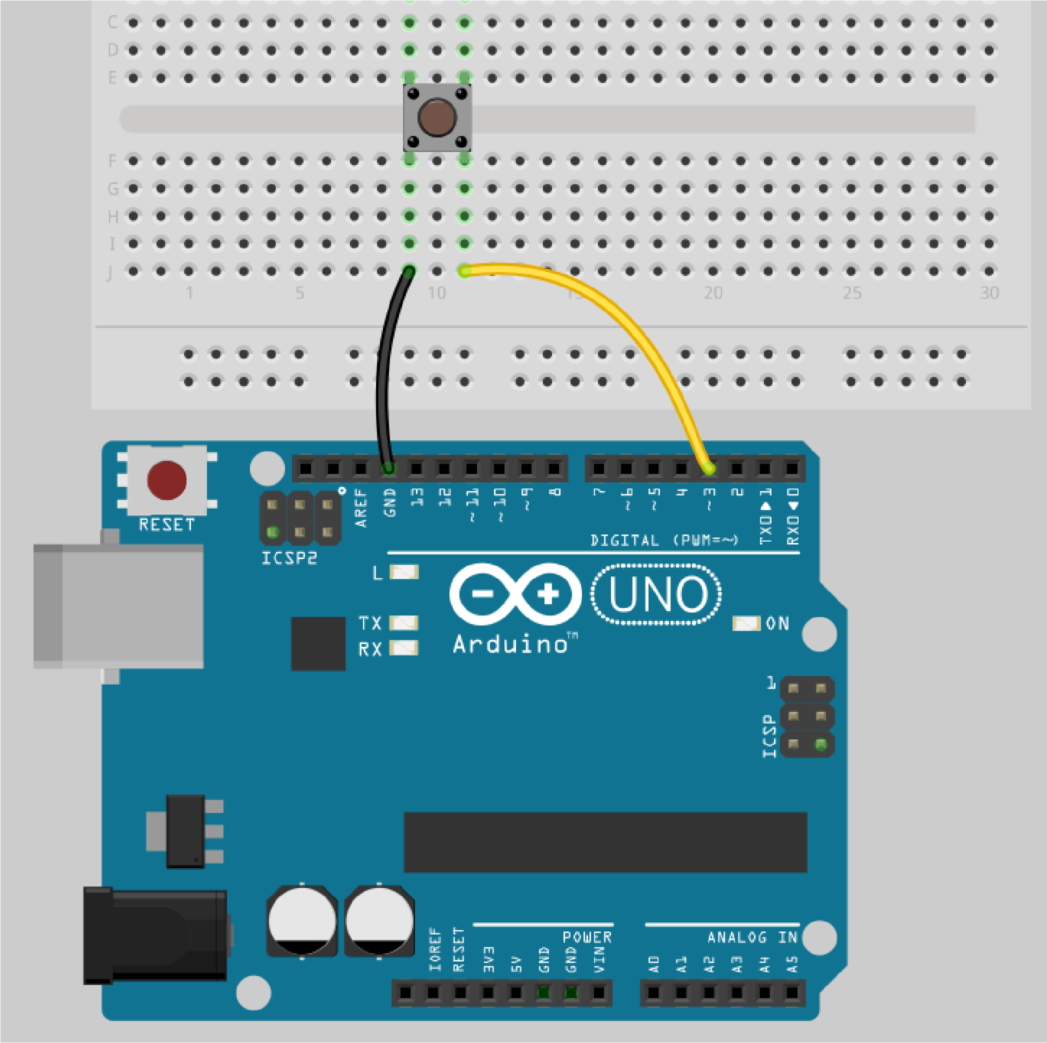
periodicADCUpdate	0
pin2Mode	not used
pin3Mode	not used
pin4Mode	not used
pin5Mode	not used
pin6Mode	not used
pin7Mode	output, default high
pin8Mode	not used
pin9Mode	not used
pin10Mode	not used
pin11Mode	not used
pin12Mode	not used
pin13Mode	not used



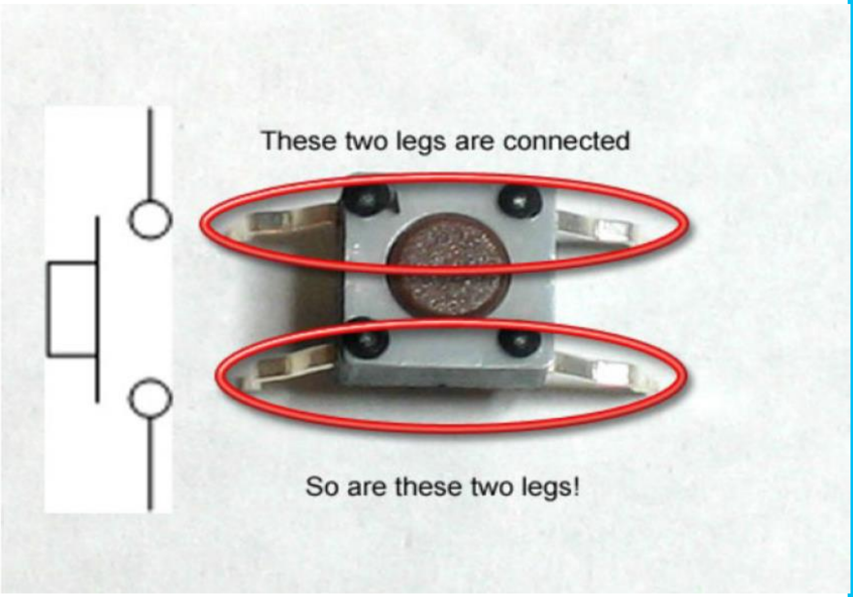


setPin6	---
clearPin6	---
setPin7	button1
setPin7	---
clearPin7	button2
clearPin7	---
setPin8	---
clearPin8	---
setPin9	---
clearPin9	---
setPin10	---

Example 2: Connecting a PushButton to Arduino

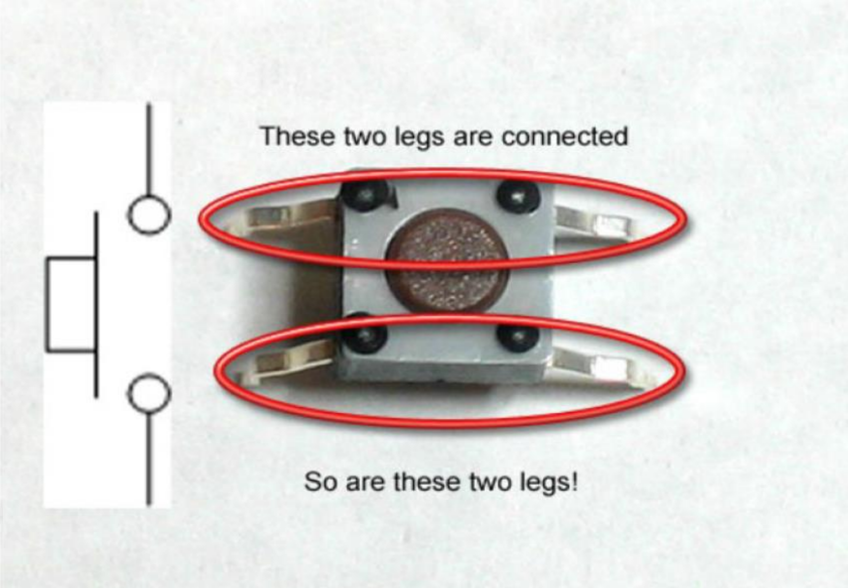


- Connect one side of switch to GND and the other side to Pin 3

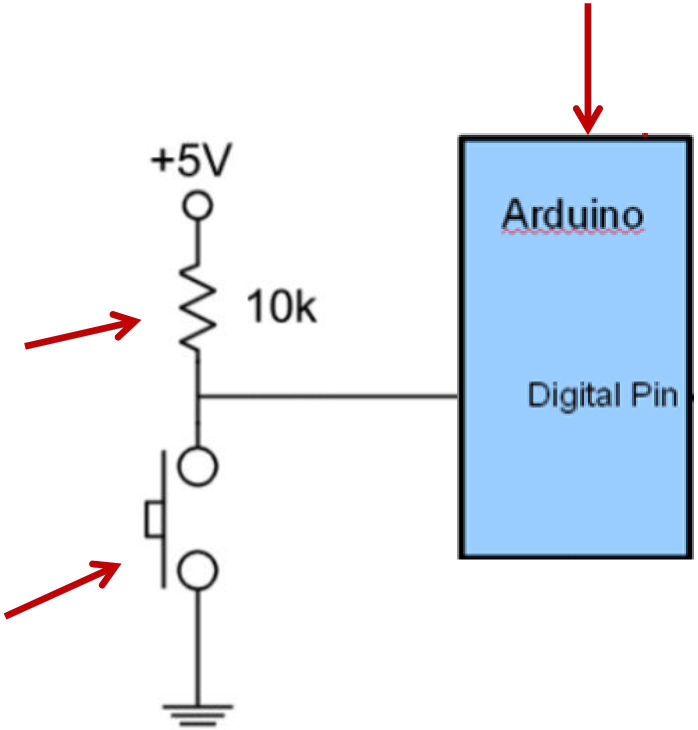


- When switch is pressed: all 4 leads connected !

Example 2: Connecting a PushButton to Arduino



Microcontroller can measure voltage (high or low) at a digital <u>input</u> Pin

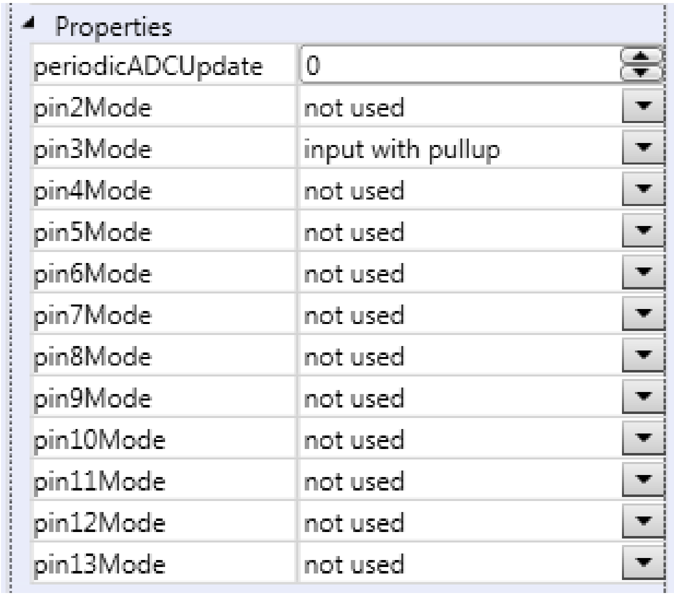


Resistor pulls Pin-voltage to 5V (high) if the switch is not pressed

Button connects Pin to 0V (low) if pressed

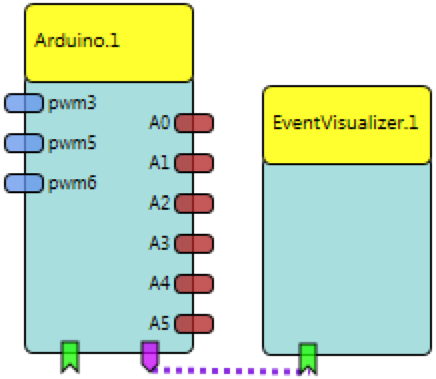
Example 2: Connecting a PushButton to Arduino

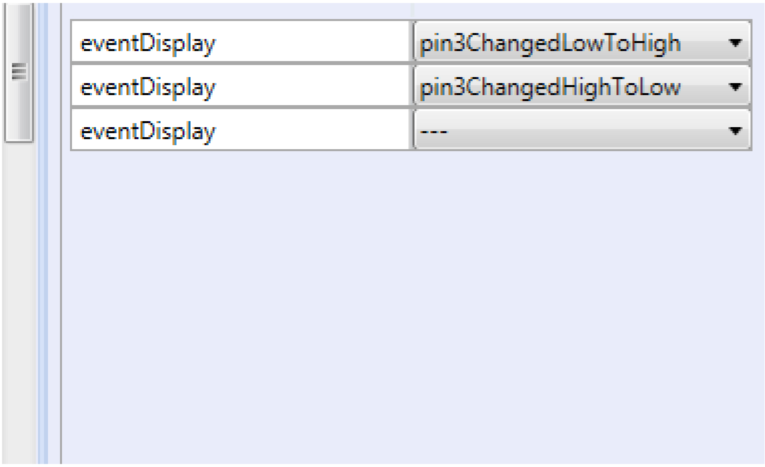
- In the Arduino processor plugin:
  - Select pin3Mode: “input with pullup”
  - Event is triggered when PushButton is pressed (high -> low)
  - Event is triggered when PushButton is relesed (low -> high)



Properties

periodicADCUpdate	0
pin2Mode	not used
pin3Mode	input with pullup
pin4Mode	not used
pin5Mode	not used
pin6Mode	not used
pin7Mode	not used
pin8Mode	not used
pin9Mode	not used
pin10Mode	not used
pin11Mode	not used
pin12Mode	not used
pin13Mode	not used

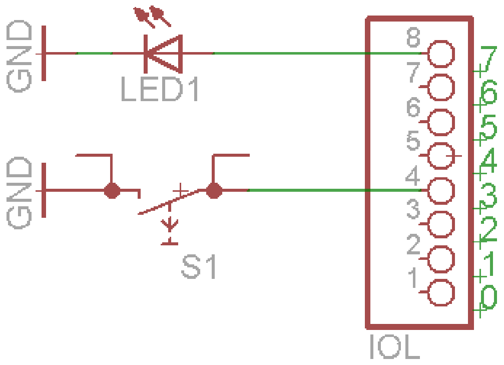


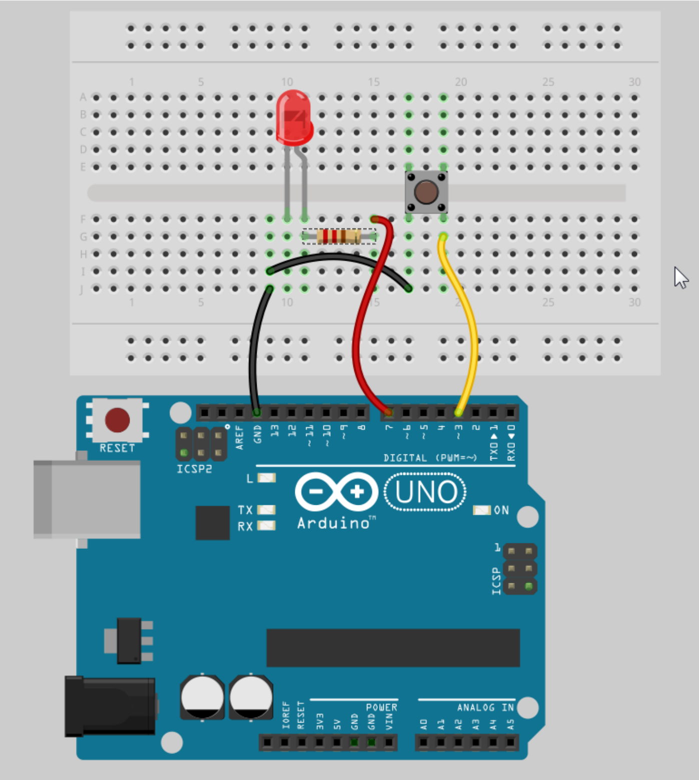


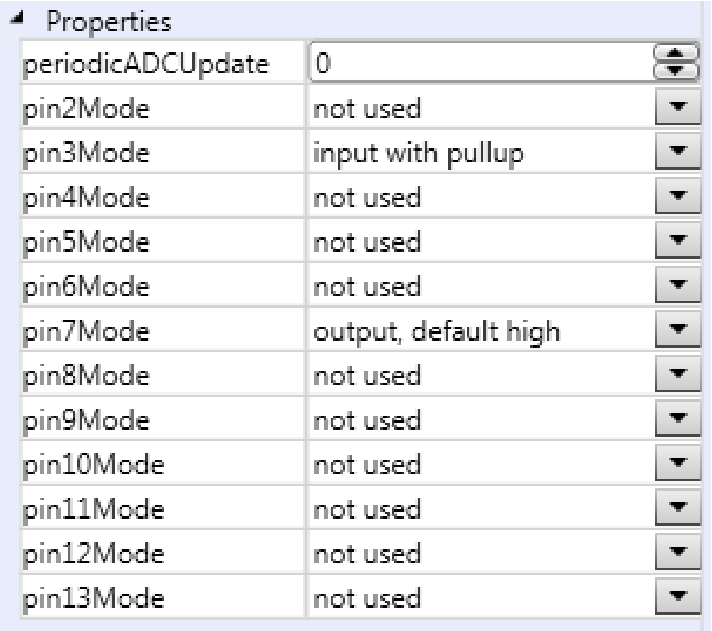
eventDisplay	pin3ChangedLowToHigh
eventDisplay	pin3ChangedHighToLow
eventDisplay	---

Example 2: Switching a Led with a Pushbutton

- Switch is connected to Pin3 (internal PullUp)
- LED is connected to Pin7
- Switch Open → LED on
- Switch Closed → LED off



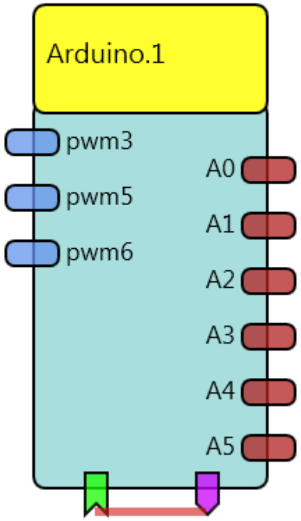


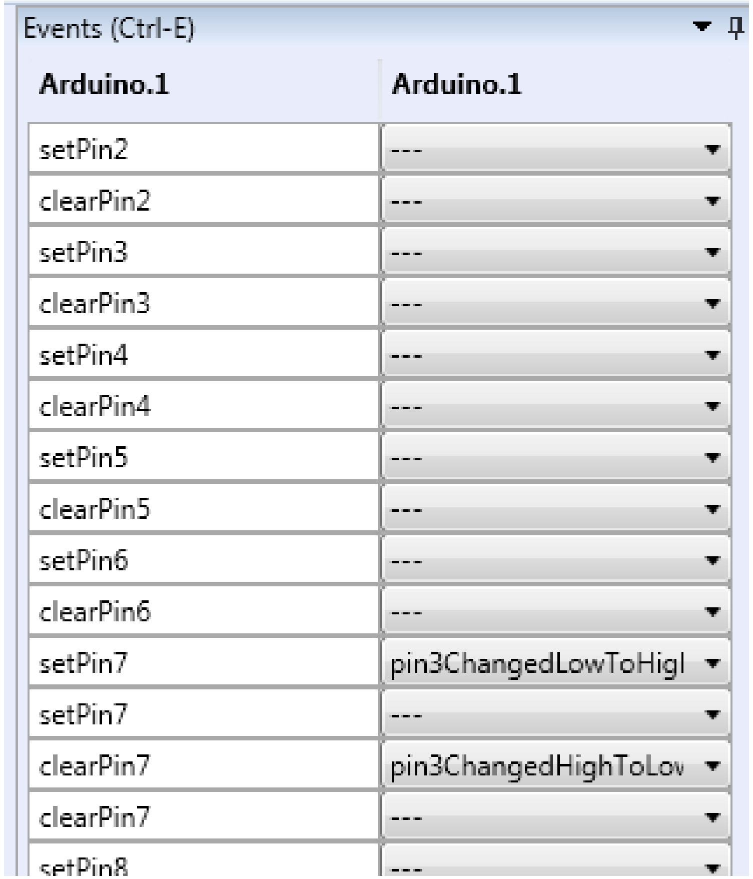


Properties

periodicADCUpdate	0
pin2Mode	not used
pin3Mode	input with pullup
pin4Mode	not used
pin5Mode	not used
pin6Mode	not used
pin7Mode	output, default high
pin8Mode	not used
pin9Mode	not used
pin10Mode	not used
pin11Mode	not used
pin12Mode	not used
pin13Mode	not used

Example 2: Switching a Led with a Pushbutton



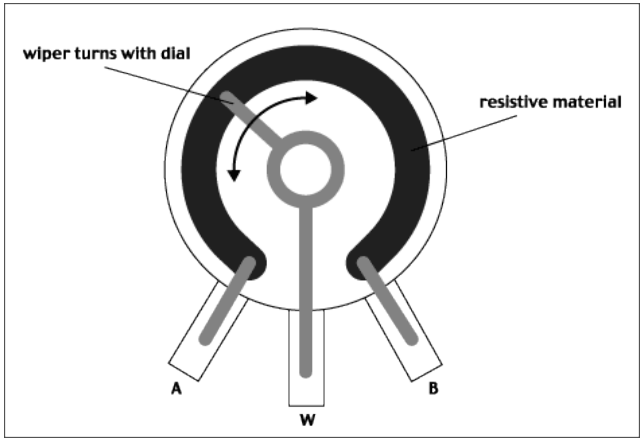

Events (Ctrl-E)

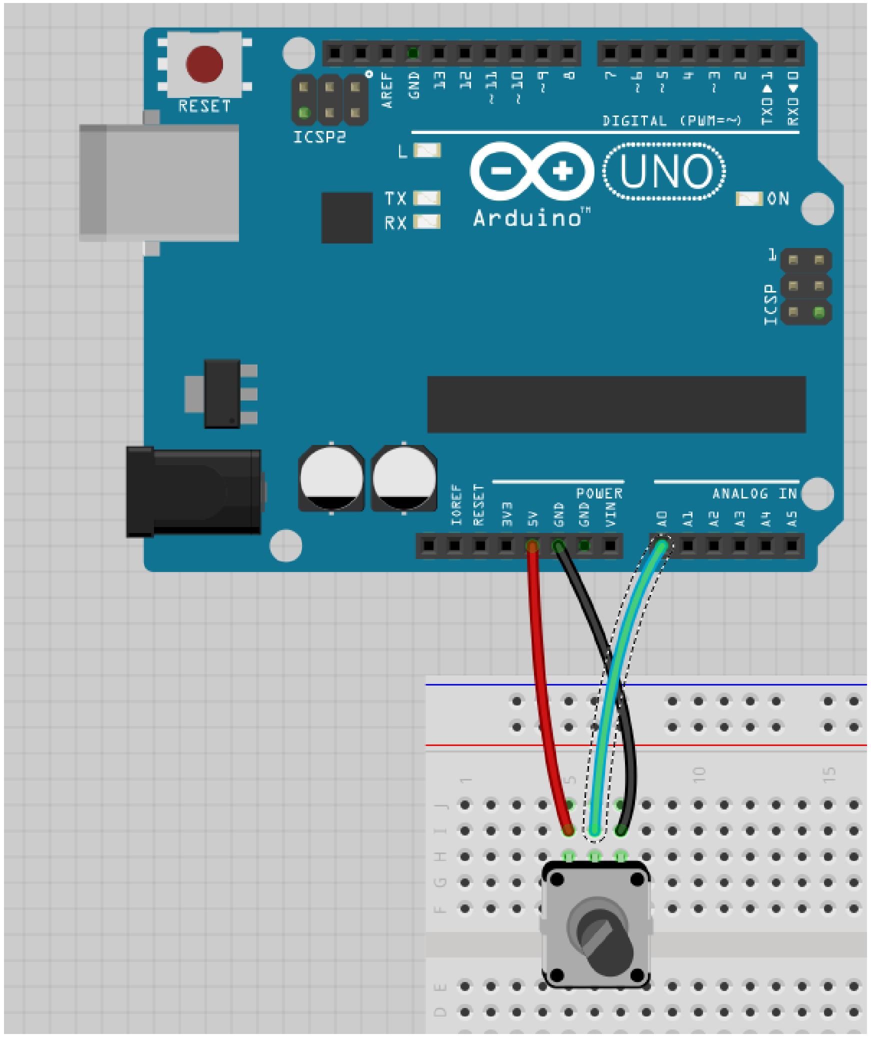
Arduino.1	Arduino.1
setPin2	---
clearPin2	---
setPin3	---
clearPin3	---
setPin4	---
clearPin4	---
setPin5	---
clearPin5	---
setPin6	---
clearPin6	---
setPin7	pin3ChangedLowToHigl
setPin7	---
clearPin7	pin3ChangedHighToLov
clearPin7	---
setPin8	---

Example 2: Reading Analogue Values

- Connect a Potentiomenter
  - Right lead goes to 5 V
  - Middle lead goes to A0
  - Left leads goes to GND

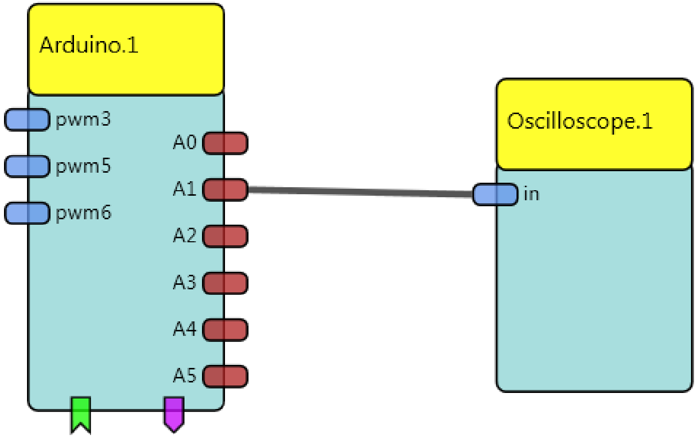
- Voltage at A0 will vary from 0V to 5V when turning

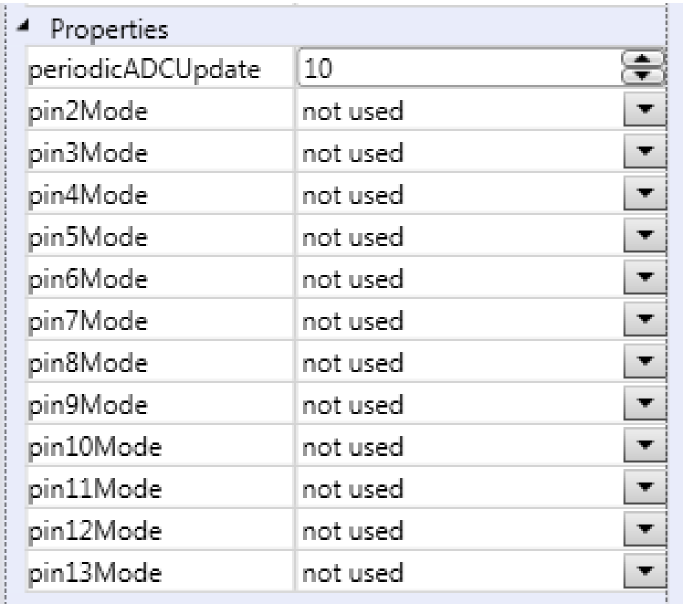




Example 2: Reading Analogue Values

- Arduino Analog-Digital Converter
  - 10 bit → 1024 Values
  - Zero Volt = Value 0
  - 5 Volt = value 1023
- Important: Set periodicADCUpdate to a positive Time (in miiliseconds) otherwise the ADC is off !!!

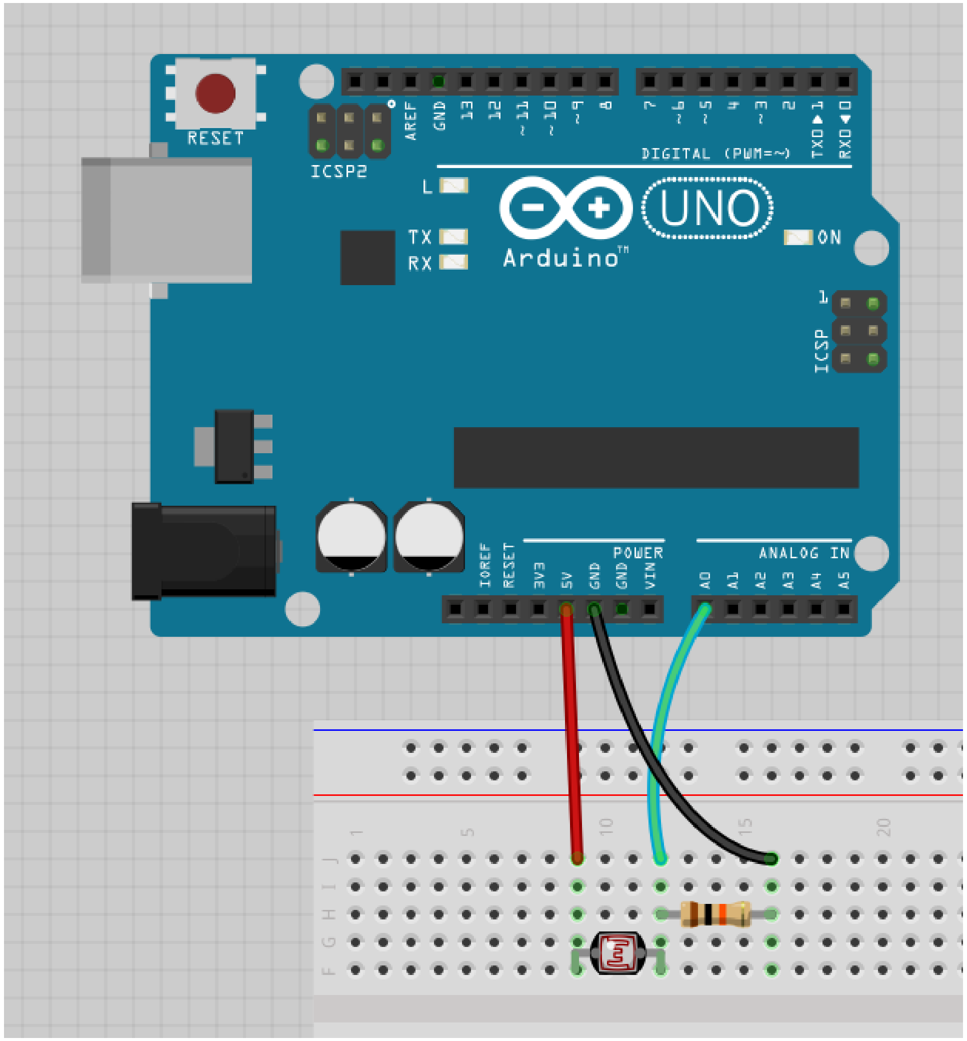


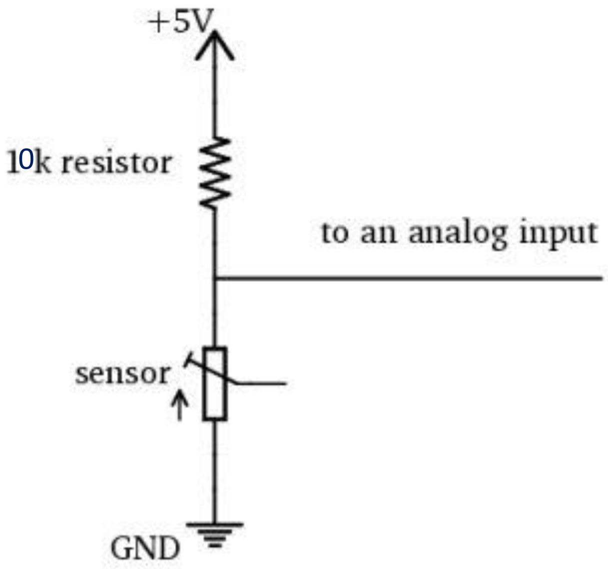


Properties

periodicADCUpdate	10
pin2Mode	not used
pin3Mode	not used
pin4Mode	not used
pin5Mode	not used
pin6Mode	not used
pin7Mode	not used
pin8Mode	not used
pin9Mode	not used
pin10Mode	not used
pin11Mode	not used
pin12Mode	not used
pin13Mode	not used

Example 2: Reading Analogue Values



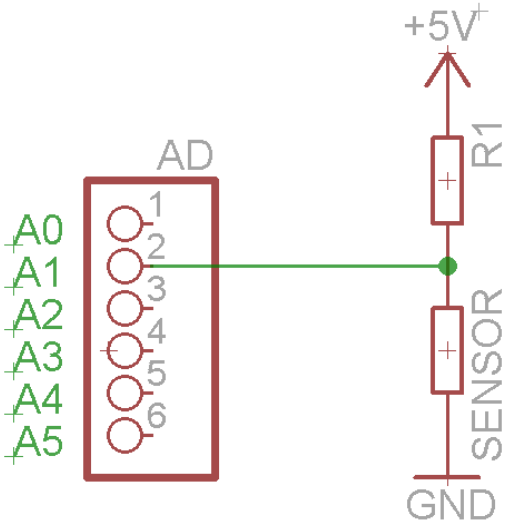


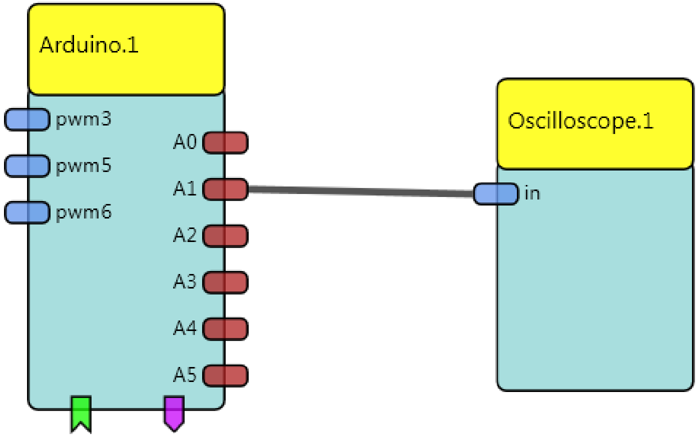
A voltage divider with a resistive sensor (e.g. LDR – Light dependent resistor) could also be a force sensor, temperature sensor etc.

Try: to print the brightness value in the ARE Window !

Example 2: Reading Analogue Values

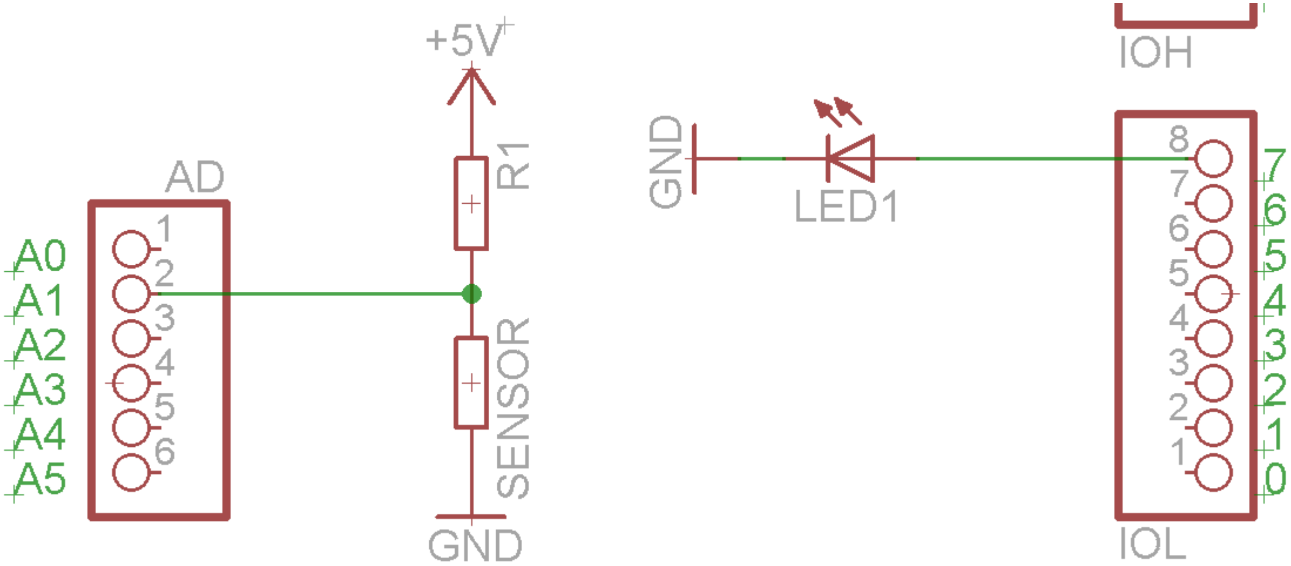
- Read a resistive sensor:
  - build a voltage divider!
  - R1 depends on the Sensor



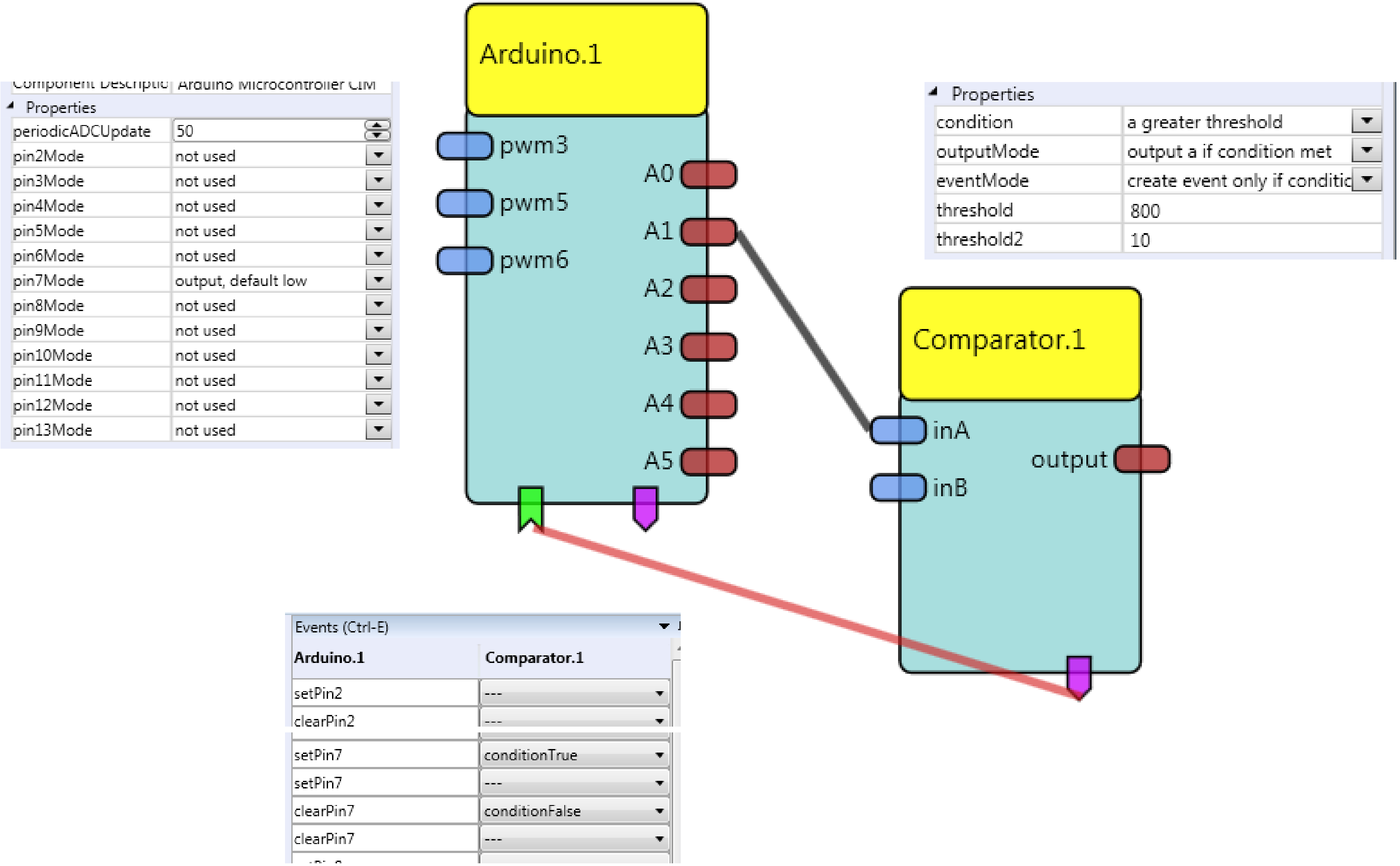


Example 2: Control Led with Sensor

- Try: Switch LED on
  if Sensor value reaches threshold



Example 2: Control Led with Sensor



Component Descriptio	Arduino Microcontroller CIM
Properties	
periodicADCUpdate	50
pin2Mode	not used
pin3Mode	not used
pin4Mode	not used
pin5Mode	not used
pin6Mode	not used
pin7Mode	output, default low
pin8Mode	not used
pin9Mode	not used
pin10Mode	not used
pin11Mode	not used
pin12Mode	not used
pin13Mode	not used

Properties	
condition	a greater threshold
outputMode	output a if condition met
eventMode	create event only if conditic
threshold	800
threshold2	10

Events (Ctrl-E)

Arduino.1	Comparator.1
setPin2	---
clearPin2	---
setPin7	conditionTrue
setPin7	---
clearPin7	conditionFalse
clearPin7	---

Hands-On Projects

HANDS ON !!

- Select a project and solve it in small groups !
- We have some sets of hardware which can be used
- 6 Project topics are available

Project 1: Digitally controlled Mouse

- Realize a fully working mouse
  with 3 push buttons

  – 2 buttons: left/right or up/down
    pressing both buttons: selection of axis (X or Y)

  – 1 button: mouse click
    short press: left click
    long press: right click

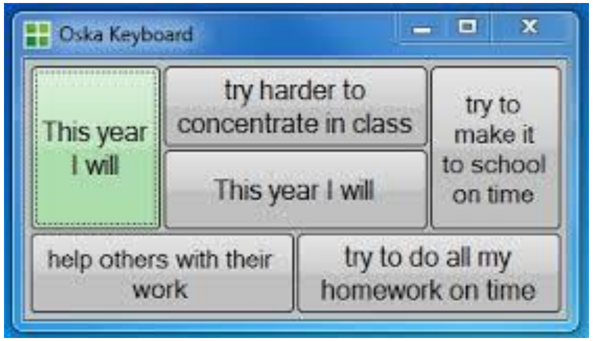
Project 2: Analog-controlled Mouse

- Realize a fully working mouse with 2 push buttons and 1 potentiometer
  - potentiometer: cursor movement with
    2 directions (e.g. +x, -x)
  - 1 button: toggle axis (X or Y)
  - 1 button: mouse click

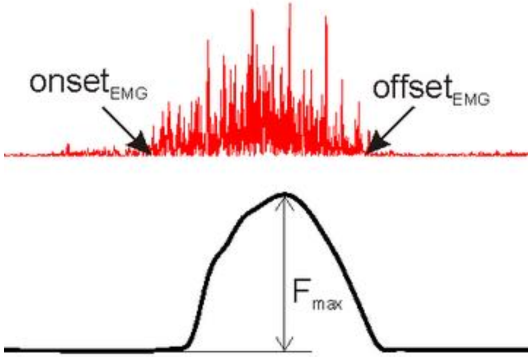
    short press: left click

    long press: right click

Project 3: Virtual Keyboard


Oska Keyboard
This year I will
try harder to concentrate in class
try to make it to school on time
This year I will
help others with their work
try to do all my homework on time

- Realize a keyboard with scanning via the Acceleration Sensor and/or EMG input
  - Use OSKA with a keyboard grid for writing
  - Use automatic scanning for key selection
  - Use EMG sensor for controlling the scanning
    - Minimal muscle movement selects key



Project 4: Environmental Control

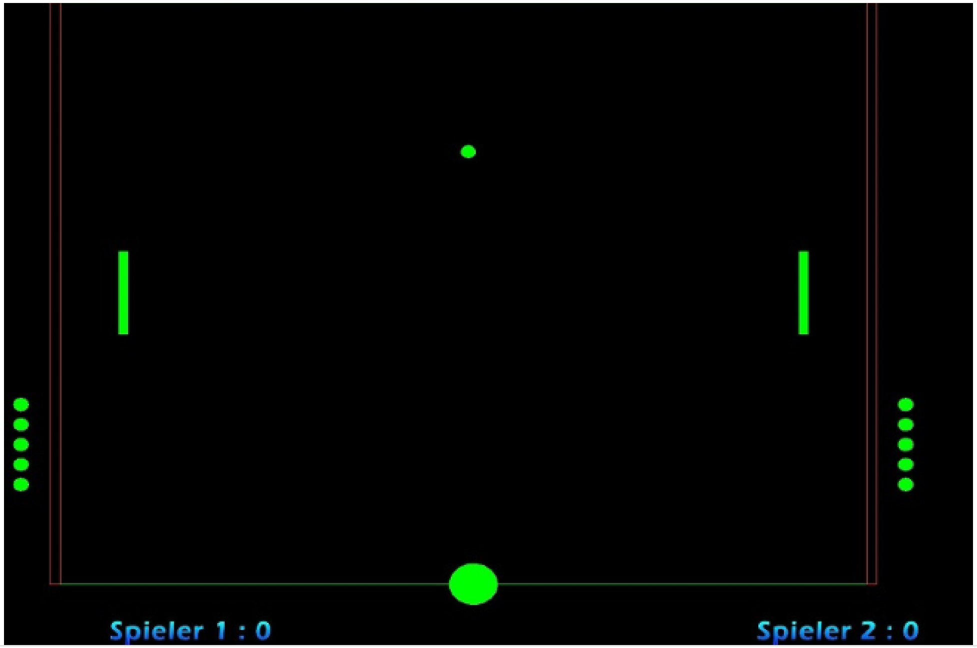
- Realize an environmental control system
  - Speech Recognition input
  - FS20 for controlling a 220V light
  - Abotic door opener connected via GPO module

Project 5: Infrared Control

- Control a GhettoBlaster via IR device
  - GUI (Cellboard) with automatic scanning
  - IR Trans for infrared remote control
  - Send commands to HiFi Stereo Radio:
    - On/Off
    - Play / Stop
    - Volume control
    - Next/Previous song

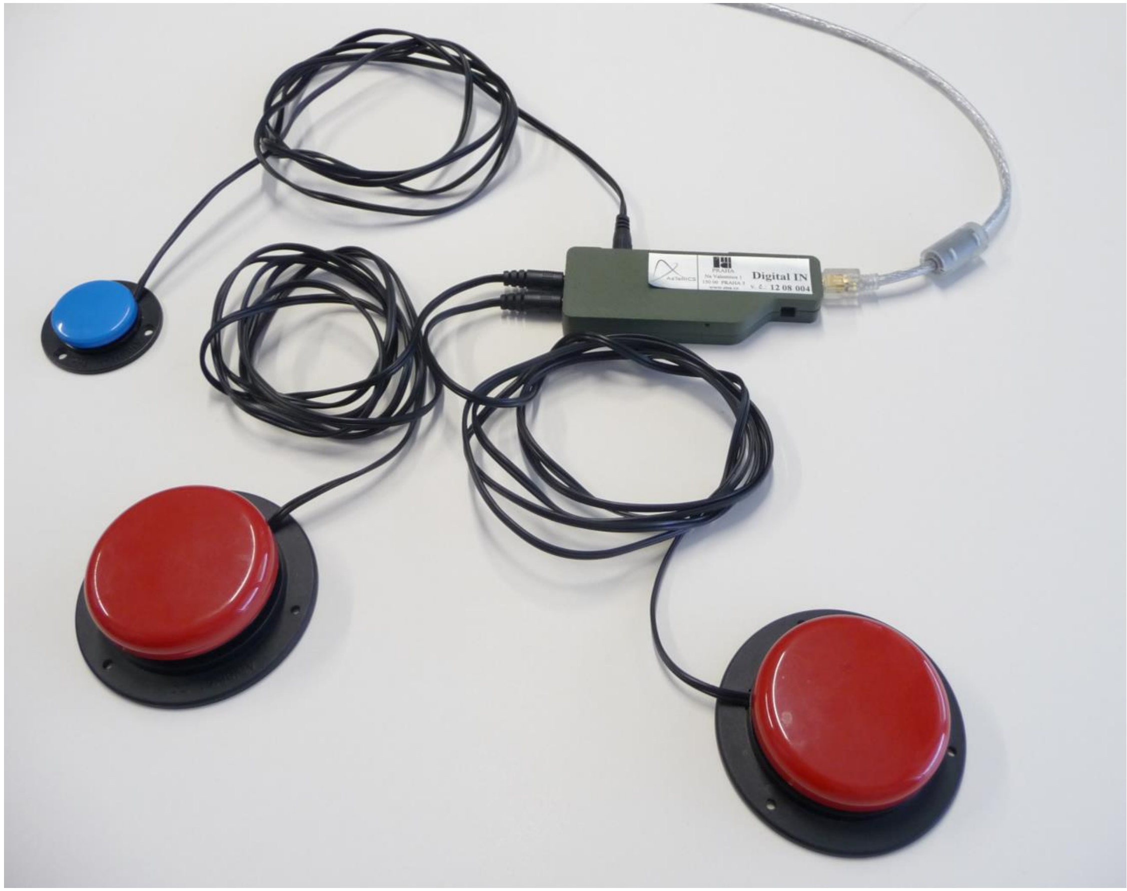
Project 6: Accessible Pong Game

- Realize a Pong Game Interface
  via desired input sensors
  - Player 1: uses Accelerometer: tilting controls paddle 1
  - Player 2: uses WebCam / FaceTracking:
    up/down movement of head controls paddle 2



Project 1 – Tips

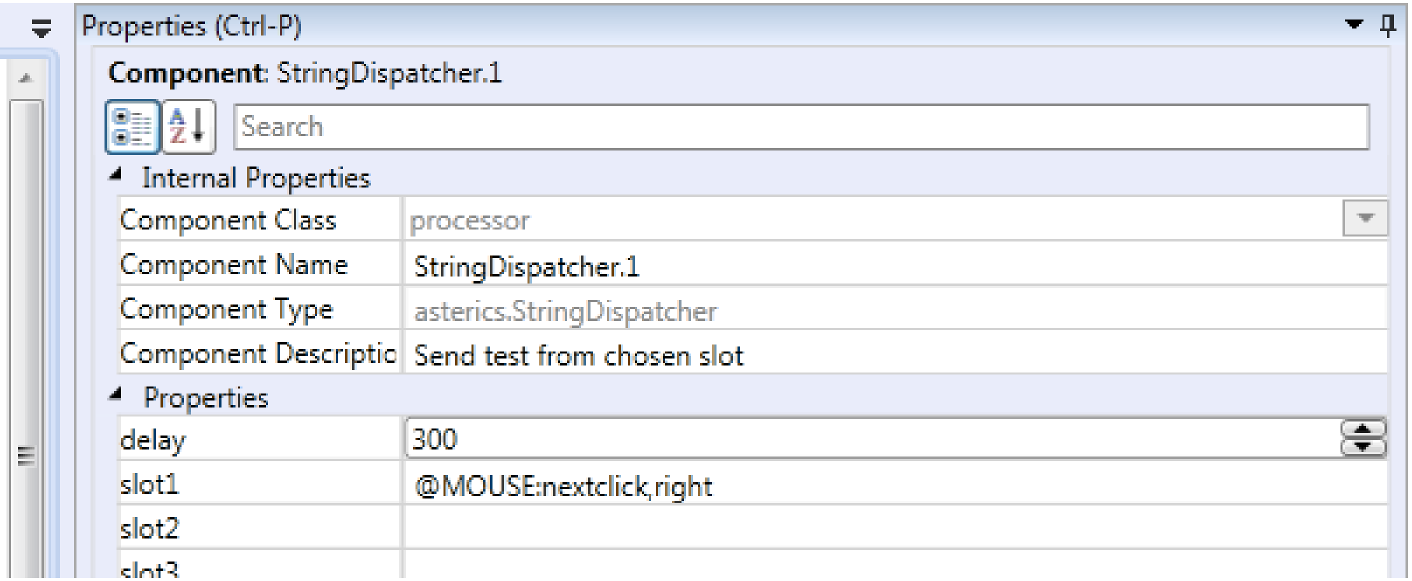
- Connect 3 buttons to your DigitalIn as shown:
- Alternatively, you can use an Arduino and use 3 pushbuttons



Project 1 – Tips

- To generate 1 right-click with the Mouse actuator, you can send the string „**@MOUSE:nextclick,right**“
to the action input port

- Use a **StringDispatcher** processor to send desired strings to other plugins (see HeadMouse example)



Component: StringDispatcher.1

Internal Properties

Component Class	processor
Component Name	StringDispatcher.1
Component Type	asterics.StringDispatcher
Component Descriptic	Send test from chosen slot

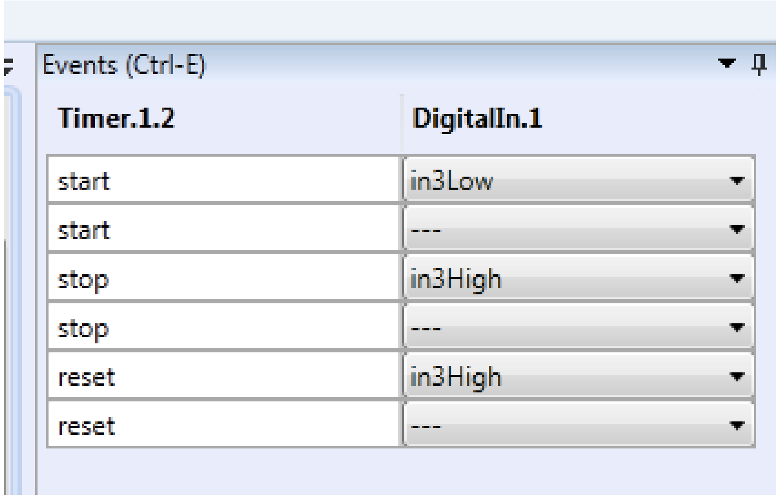
Properties

delay	300
slot1	@MOUSE:nextclick,right
slot2	
slot3	

Project 1 – Tips

- Use a **Timer** sensor to detect a long button press
- If the button is held for a desired time (defined in the properties) the timer can send an event
- Attach Events from the DigitalIn to the Timer plugin so that:
  - Pressing the button starts the timer
  - Releasing the button stops (and resets) the timer.

The example shows events from a button connected to DigitalIn 3:

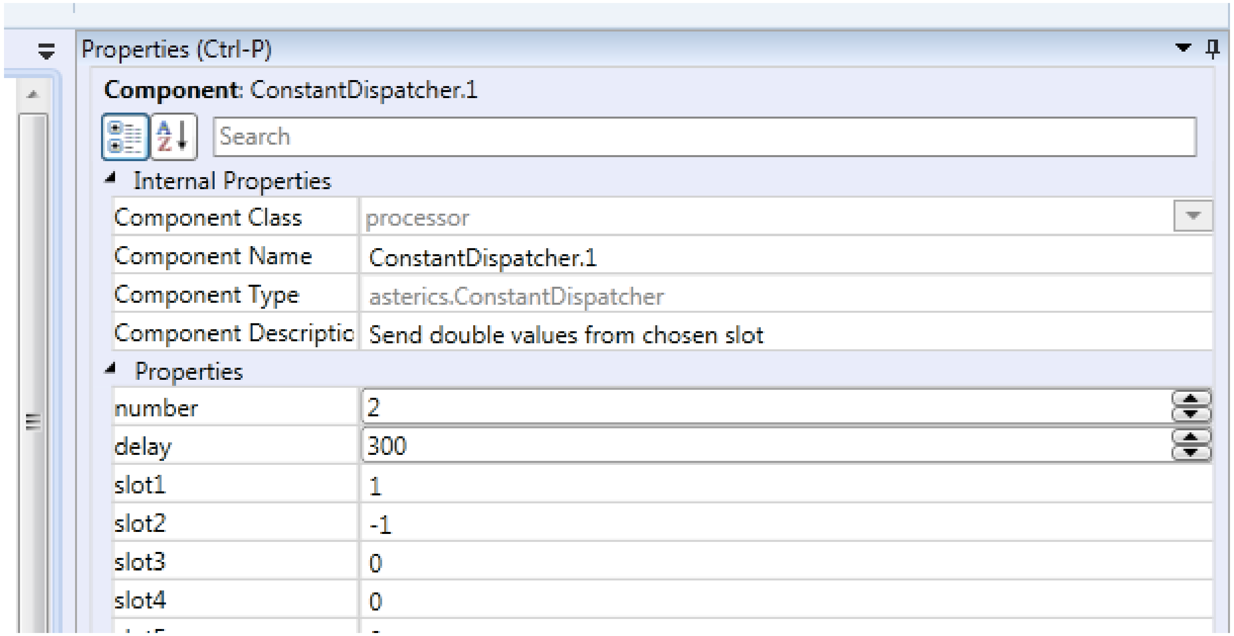


Events (Ctrl-E)

Timer.1.2	DigitalIn.1
start	in3Low
start	---
stop	in3High
stop	---
reset	in3High
reset	---

Project 1 – Tips

- Use a ConstantDispatcher processor to create movement:
  - Send negative value (-1) to Mouse actuator x/y for up/left
  - Send positive value (1) to Mouse actuator x/y for right/down



Properties (Ctrl-P)

Component: ConstantDispatcher.1

Internal Properties

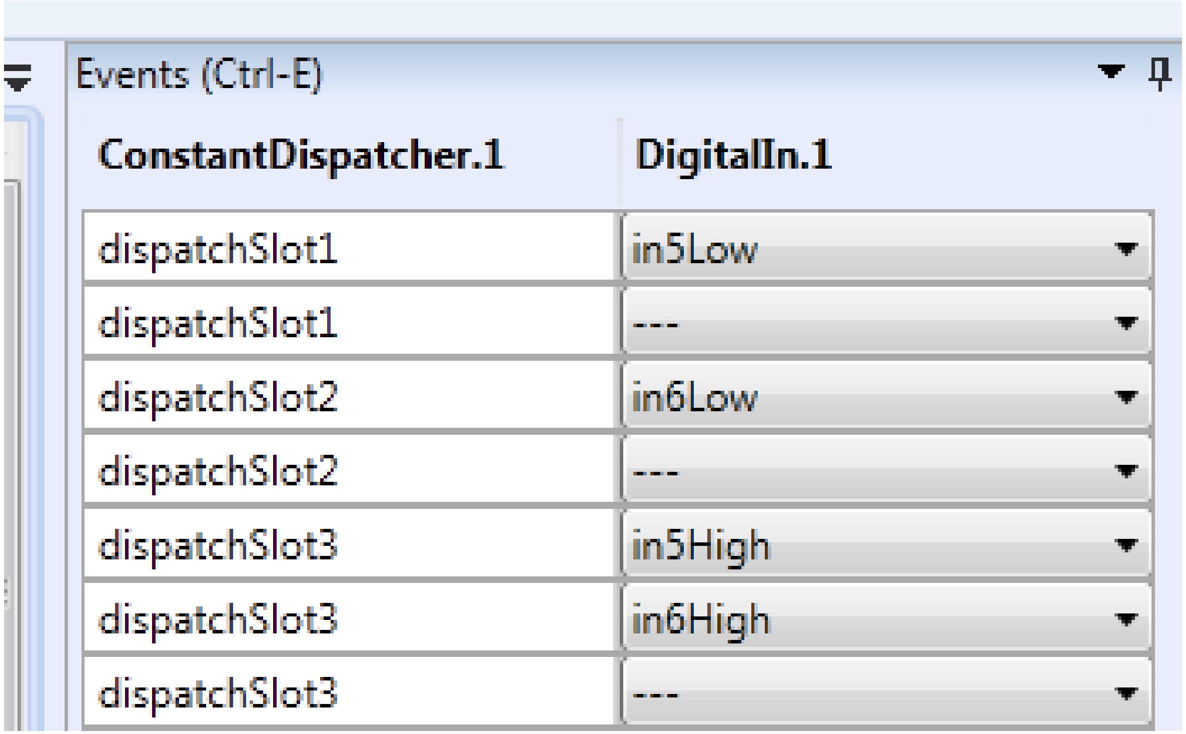
Component Class	processor
Component Name	ConstantDispatcher.1
Component Type	asterics.ConstantDispatcher
Component Descriptic	Send double values from chosen slot

Properties

number	2
delay	300
slot1	1
slot2	-1
slot3	0
slot4	0

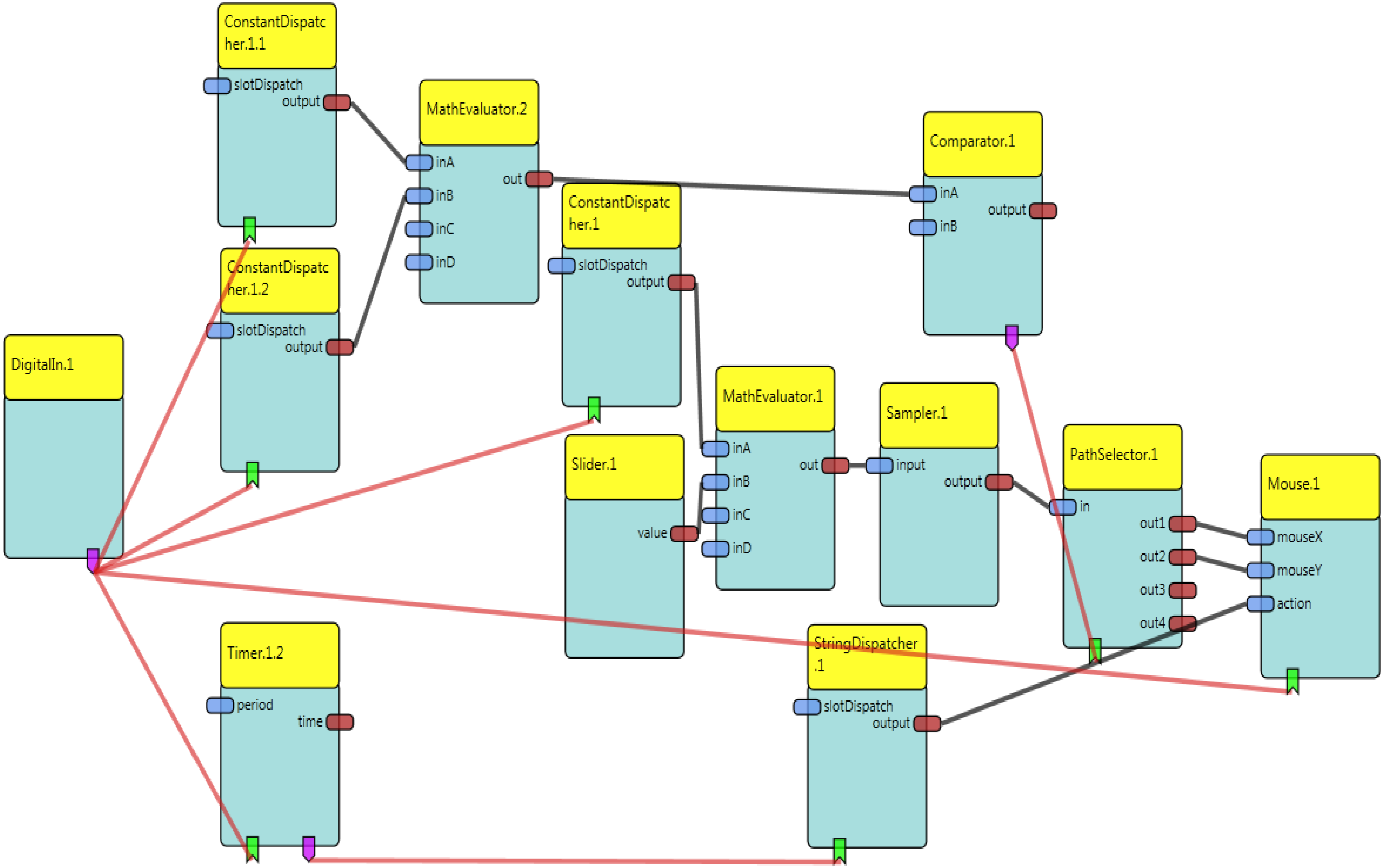
Project 1 – Tips

- Dispatch up/left or right/down movement via connected pushbuttons (here: DigitalIn Pins)

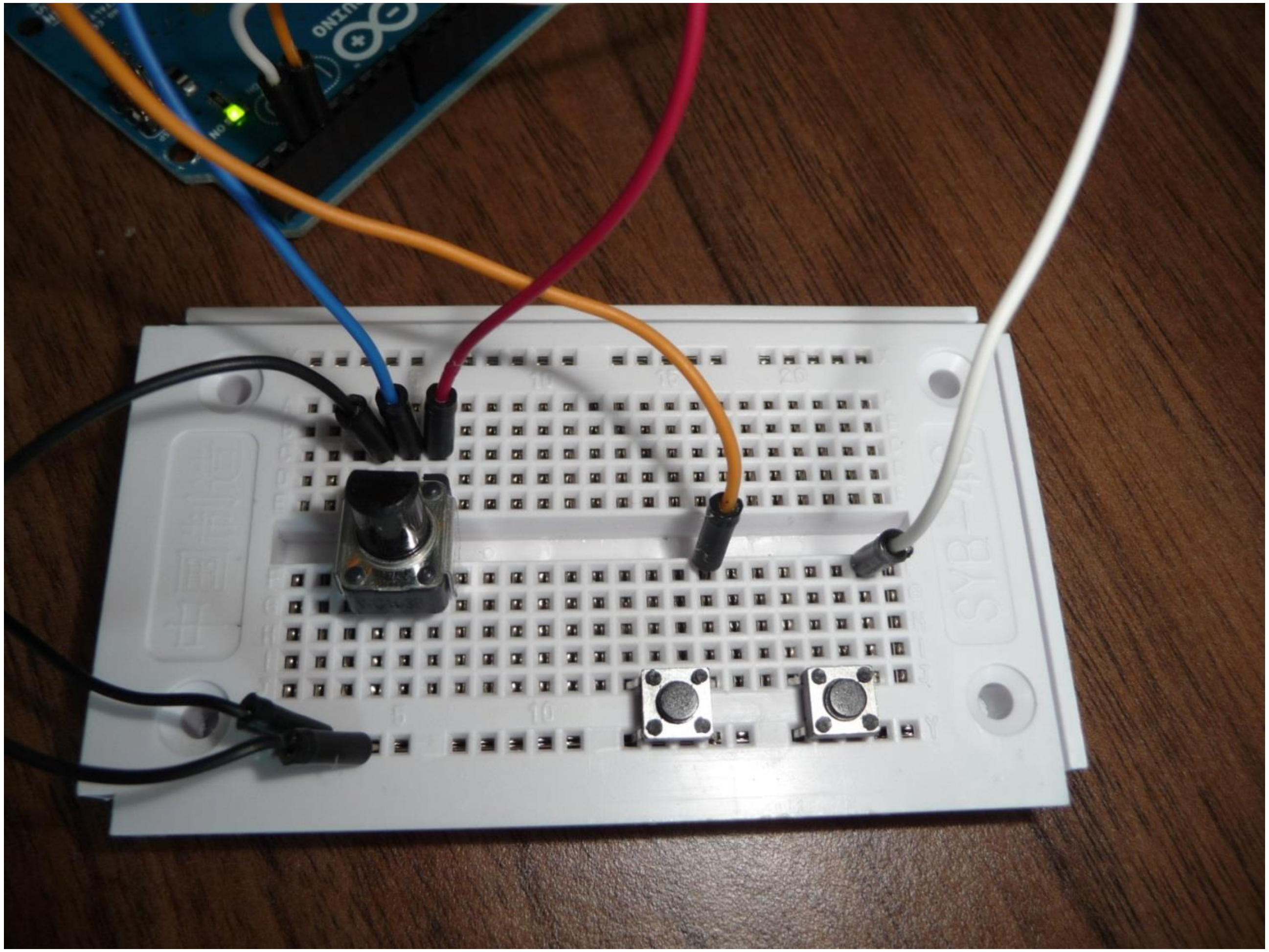


ConstantDispatcher.1	DigitalIn.1
dispatchSlot1	in5Low
dispatchSlot1	---
dispatchSlot2	in6Low
dispatchSlot2	---
dispatchSlot3	in5High
dispatchSlot3	in6High
dispatchSlot3	---

Project 1: a possible solution …

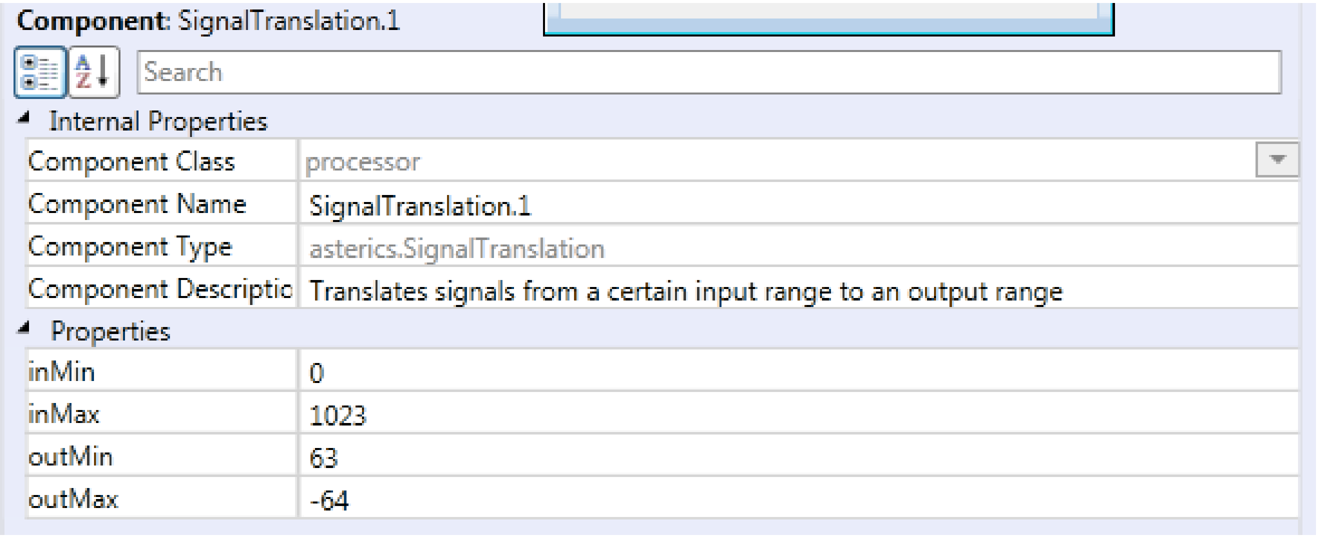


Project 2 – Tips



Project 2 – Tips

- Different value ranges of mouse movement and Arduino ADC:
  - Mouse: relative, positive and negative values
  - ADC: absolute, only positive values (0-1023, 12bit)
- Solution: SignalTranslation plugin



Component: SignalTranslation.1

Search

Internal Properties

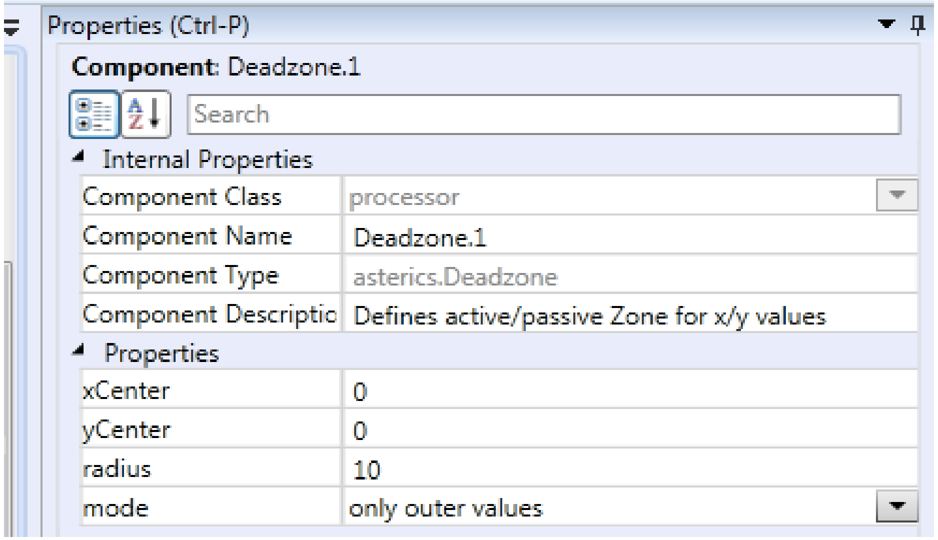
Component Class	processor
Component Name	SignalTranslation.1
Component Type	asterics.SignalTranslation
Component Descriptio	Translates signals from a certain input range to an output range

Properties

inMin	0
inMax	1023
outMin	63
outMax	-64

Project 2 – Tips

- It‘s hard to find the exact centre of the potentiometer
  - > the Cursor is always moving
- We need a window around the centre, where the cursor does not move:
  - > Deadzone plugin



Properties (Ctrl-P)

Component: Deadzone.1

Search

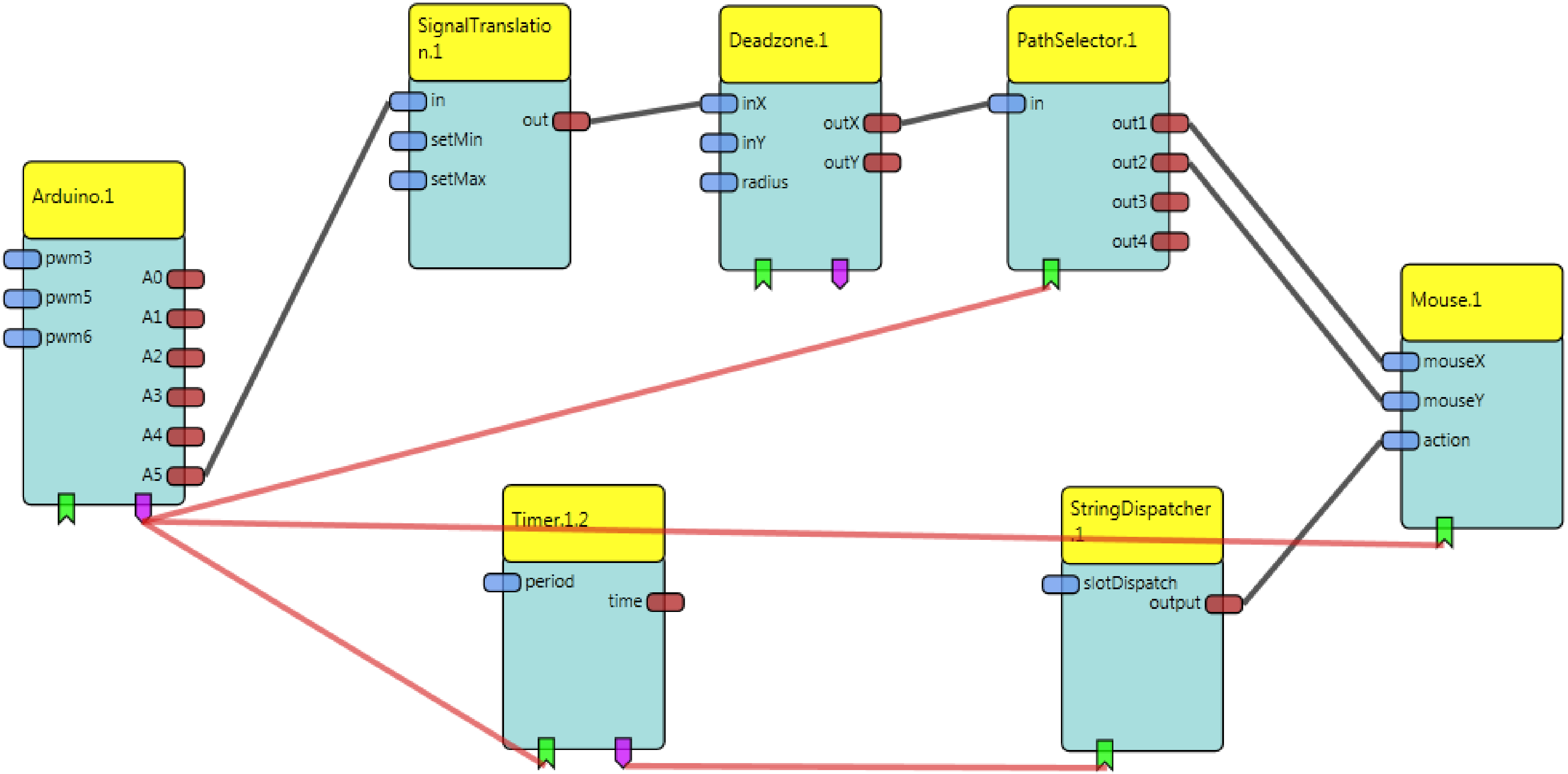
Internal Properties

Component Class	processor
Component Name	Deadzone.1
Component Type	asterics.Deadzone
Component Descriptic	Defines active/passive Zone for x/y values

Properties

xCenter	0
yCenter	0
radius	10
mode	only outer values

Project 2: a possible solution ….



Project 3 – Tips

- Connect the EMG shield to the Arduino
  - Attach electrodes to forearm
- Use an Arduino plugin to view analog values of AD0
  - Select an update rate of 5ms (200Hz) for the Arduino ADC
- Use oscilloscope to view raw singal (try it out !)

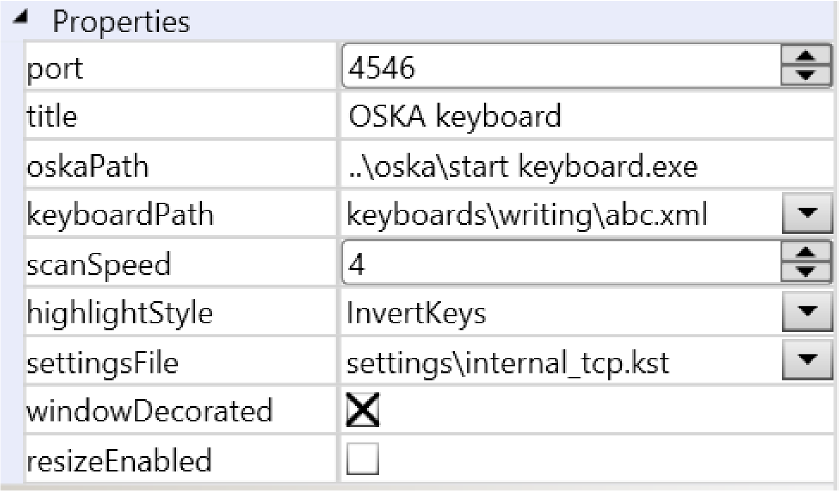
Project 3 – Tips

- To calculate the EMG force:
  - With a **Differentiate** processor you to get rid of the DC-offsets and slow changes in the EMG signal. (You can also experiment with the Filter Plugin)
  - use a **MathEvaluator** to get the absolute values of the signal enter **abs(a)** as expression in the MathEvaluator properties
  - use an **Averager** to calculate the average of 30 samples

- Use a **Threshold** processor to create events if the EMG force exceeds a certain level
  - Check reasonable values with oscilloscope or BarDisplay

Project 3 – Tips

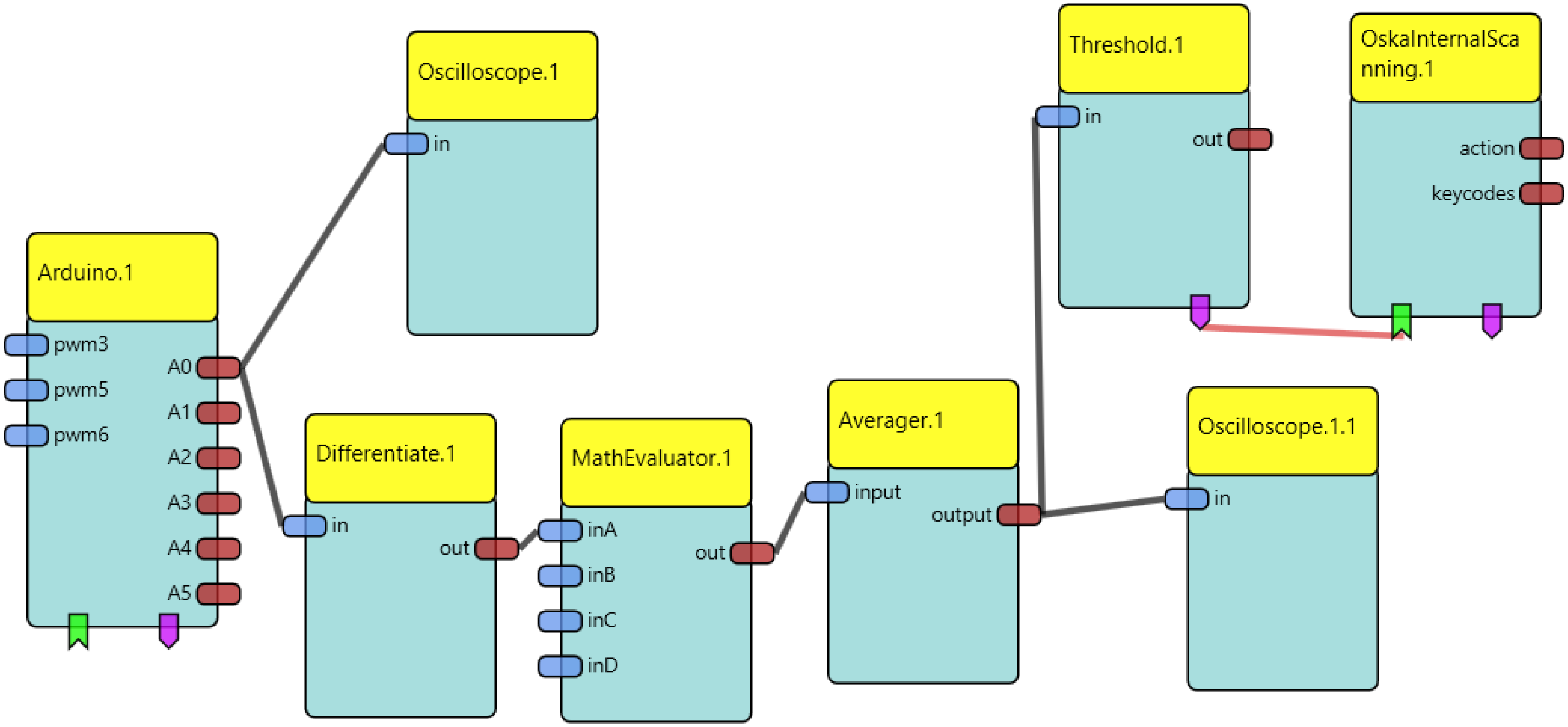
- Add a „**OSKAInternalScanning**“ plugin
  - this launches the OSKA on-screen keyboard in automatic scanning mode
  - Upload model to ARE (to update list of available keyboards)
  - Select an appropriate keyboard grid for writing for example: „**keyboards\writing\abc.xml**“
  - Select scan speed 4 and highlightStyle „InvertKeys“



Properties

port	4546
title	OSKA keyboard
oskaPath	..\oska\start keyboard.exe
keyboardPath	keyboards\writing\abc.xml
scanSpeed	4
highlightStyle	InvertKeys
settingsFile	settings\internal_tcp.kst
windowDecorated	☒
resizeEnabled	□

Project 3: a possible solution …



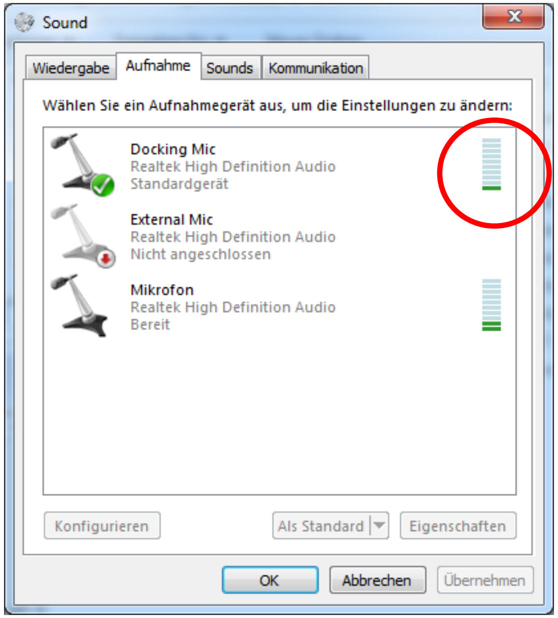
Project 4 – Tips

Requirements:

- Abotic Door Opener
  - http://abotic.com/en/
  - DigitalOut module
  - Plug In Abotic Door Remote to **DigitalOut port 1**

- Voice recognition
  - Microphone (Check **signal level**!!)
  - **Microsoft Speech Platform Server version 11**, the **SR language** and the **TTS language pack** must be installed too.
  - http://www.microsoft.com/en-us/download/details.aspx?id=27225
- FS20Sender device
  - http://www.elv.at/fs20-funkschaltsystem.html

Project 4 – Tips



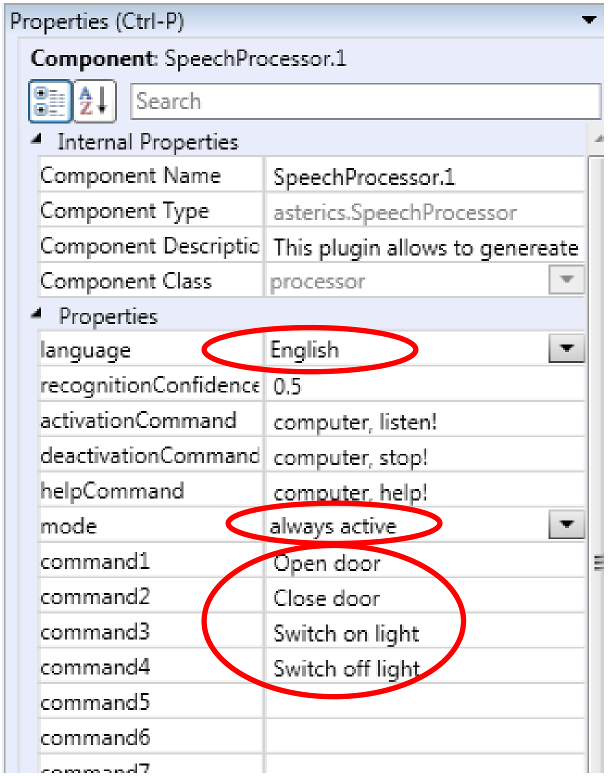
Sound

Wiedergabe | Aufnahme | Sounds | Kommunikation

Wählen Sie ein Aufnahmegerät aus, um die Einstellungen zu ändern:

- Docking Mic
  Realtek High Definition Audio
  Standardgerät
- External Mic
  Realtek High Definition Audio
  Nicht angeschlossen
- Mikrofon
  Realtek High Definition Audio
  Bereit

Check the signal level of the microphone!



Properties (Ctrl-P)

Component: SpeechProcessor.1

Internal Properties

Component Name	SpeechProcessor.1
Component Type	asterics.SpeechProcessor
Component Descriptio	This plugin allows to genereate
Component Class	processor

Properties

language	English
recognitionConfidence	0.5
activationCommand	computer, listen!
deactivationCommand	computer, stop!
helpCommand	computer, help!
mode	always active
command1	Open door
command2	Close door
command3	Switch on light
command4	Switch off light
command5	
command6	
command7	

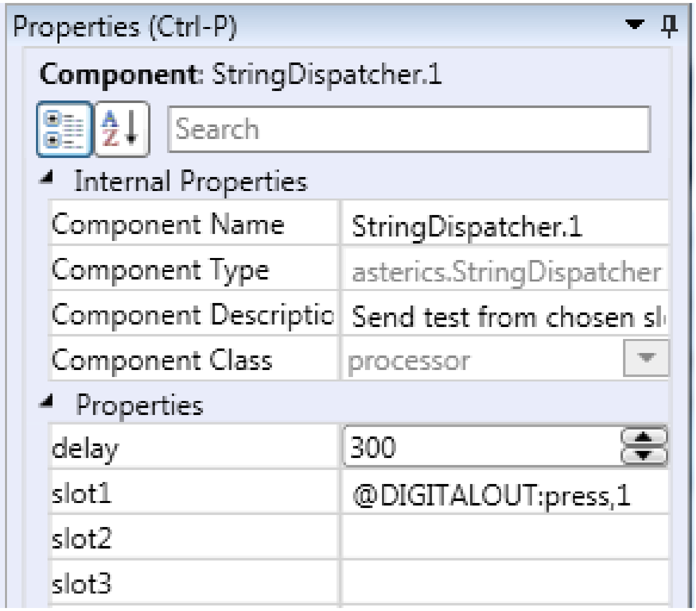
Select the recognition **language**. In the **mode** „always active“ the recognition is performed continuously.

command1-commandN: Enter voice commands

Project 4 – Tips

- The **DigitalOut** plugin expects action strings: "set", "clear", "toggle" and "press". The command has to be followed by a comma and the port number, for example:
  - "@DIGITALOUT:set,1" or
  - "@DIGITALOUT:toggle,2“.
  - The "press"-command toggles the given output port two times with a delay of 500ms.

- **StringDispatcher** can be used to construct the string.



Component: StringDispatcher.1

Internal Properties

Component Name	StringDispatcher.1
Component Type	asterics.StringDispatcher
Component Descriptic	Send test from chosen sl
Component Class	processor

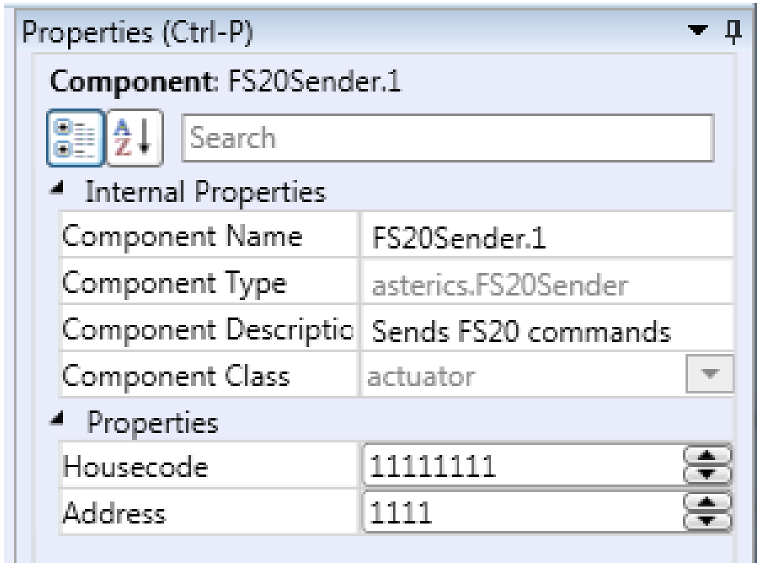
Properties

delay	300
slot1	@DIGITALOUT:press,1
slot2	
slot3	

Project 4 – Tips

- Enter the **housecode** and **address** of the target device in the **FS20Sender** properties.

- Our FS20Receiver Power Switch is already configured to housecode 11111111 and address 1111



Properties (Ctrl-P)

Component: FS20Sender.1

Search

Internal Properties

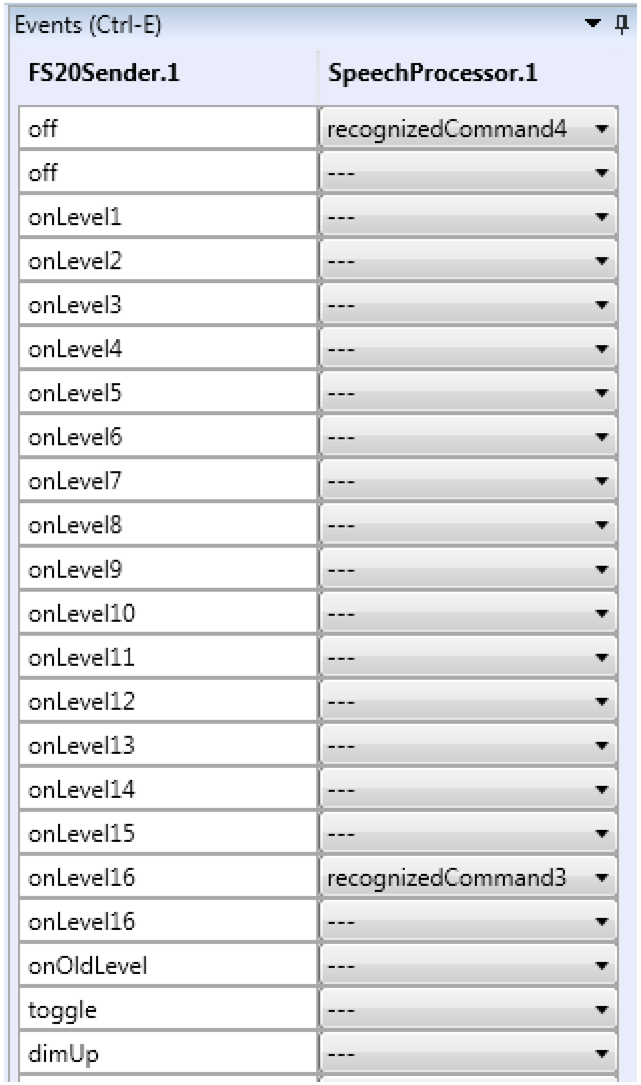
Component Name	FS20Sender.1
Component Type	asterics.FS20Sender
Component Descriptio	Sends FS20 commands
Component Class	actuator

Properties

Housecode	11111111
Address	1111

Project 4 – Tips

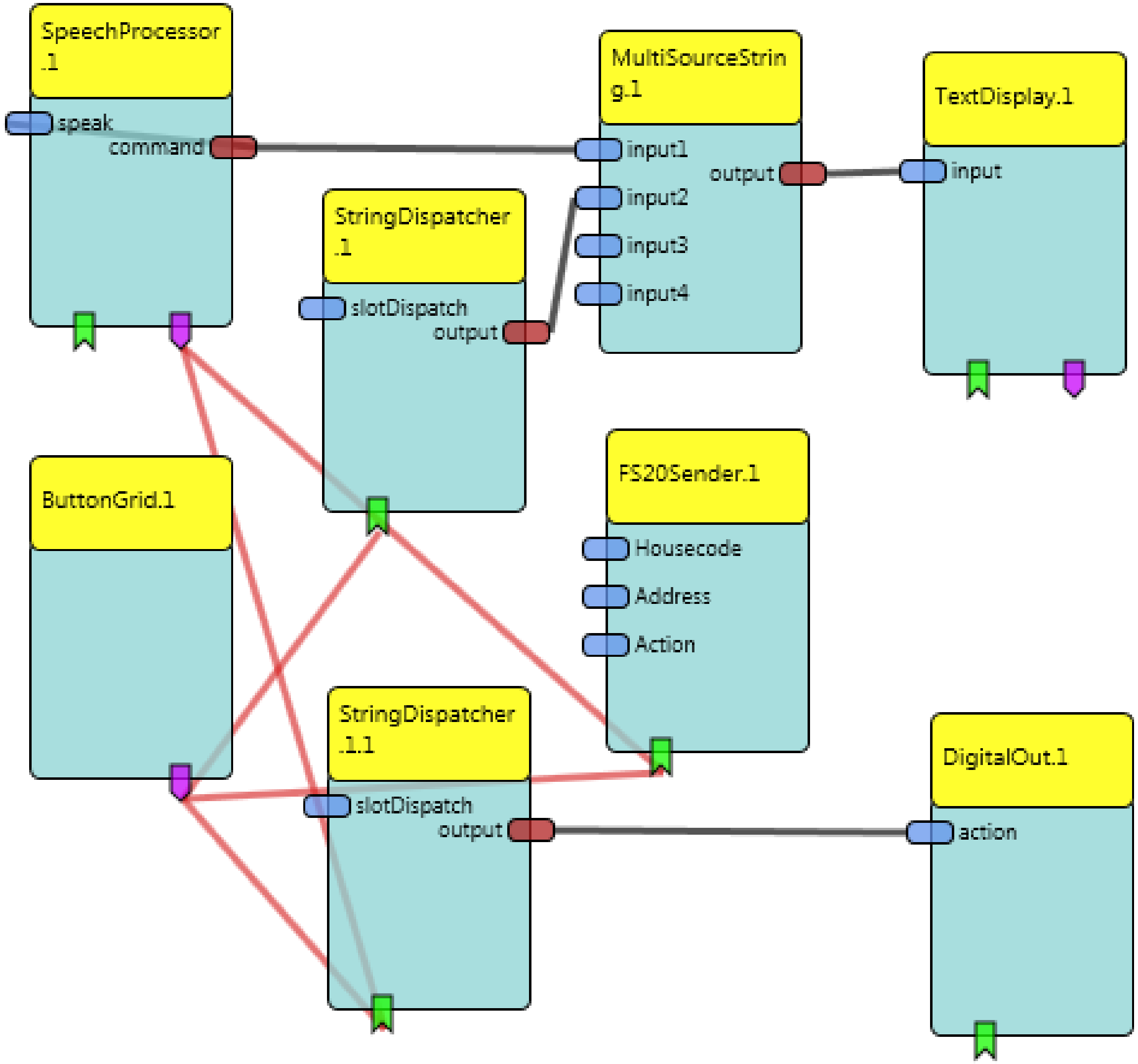
- In the event editor of **FS20Sender** map recognized commands with FS20 actions:
  - **off**: Switch off device
  - **onLevel1-onLevel16**: Switch on device to a certain level
  - **toggle**: Toggles switching (on/off)



Events (Ctrl-E)

FS20Sender.1	SpeechProcessor.1
off	recognizedCommand4
off	---
onLevel1	---
onLevel2	---
onLevel3	---
onLevel4	---
onLevel5	---
onLevel6	---
onLevel7	---
onLevel8	---
onLevel9	---
onLevel10	---
onLevel11	---
onLevel12	---
onLevel13	---
onLevel14	---
onLevel15	---
onLevel16	recognizedCommand3
onLevel16	---
onOldLevel	---
toggle	---
dimUp	---

Project 4: a possible solution …

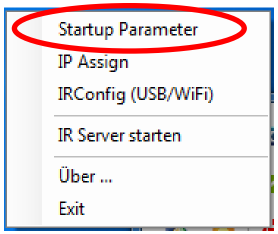


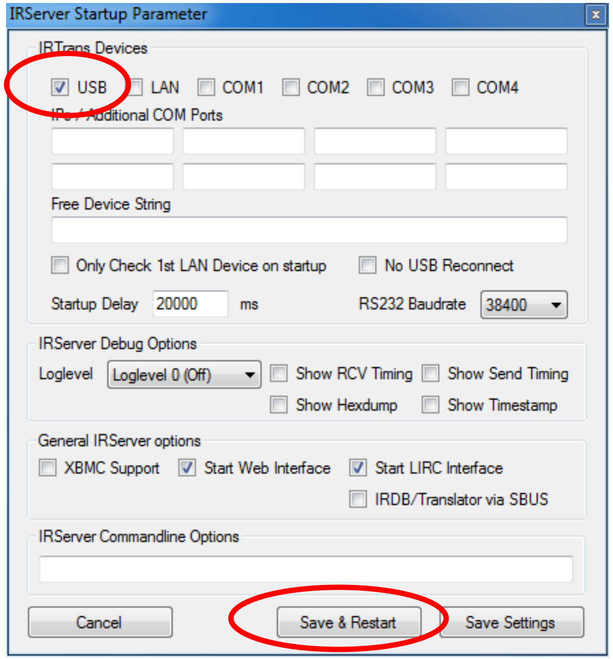
Project 5 – Tips

Requirements:

- Arduino
  - Connect **push button** as shown in Arduino slides, use **internal pullup resistor**

- IRTrans device
  - Install IRTrans driver and SW from http://www.irtrans.de/de/download/windows.php
  - Copy **“asterics.rem”** remote configuration file to **C:\ProgramData\IRTrans\remotes**
  - Start **IRTransServer.exe**
  - Right click on IRTrans symbol in system tray and choose **“Startup Parameter”**
  - Select **“USB”** and **“Save & Restart”**





IRServer Startup Parameter

IRTrans Devices

☑ USB ☐ LAN ☐ COM1 ☐ COM2 ☐ COM3 ☐ COM4

IPs / Additional COM Ports

Free Device String

☐ Only Check 1st LAN Device on startup ☐ No USB Reconnect

Startup Delay 20000 ms RS232 Baudrate 38400

IRServer Debug Options

Loglevel Loglevel 0 (Off) ☐ Show RCV Timing ☐ Show Send Timing ☐ Show Hexdump ☐ Show Timestamp

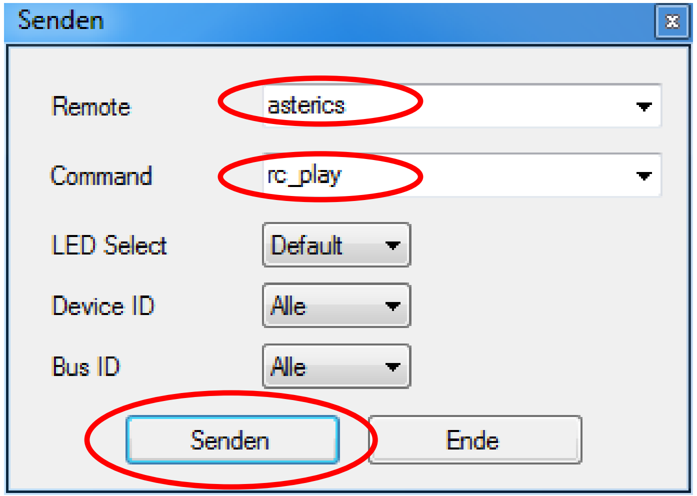
General IRServer options

☐ XBMC Support ☑ Start Web Interface ☑ Start LIRC Interface ☐ IRDB/Translator via SBUS

IRServer Commandline Options

Project 5 – Tips

- Test sending IR codes
  - Right click on IRTrans symbol in system tray
  - Select **“Send IR Code”**
  - Select **“asterics”** as remote configuration
  - Select a command
  - Click on **“Send”**



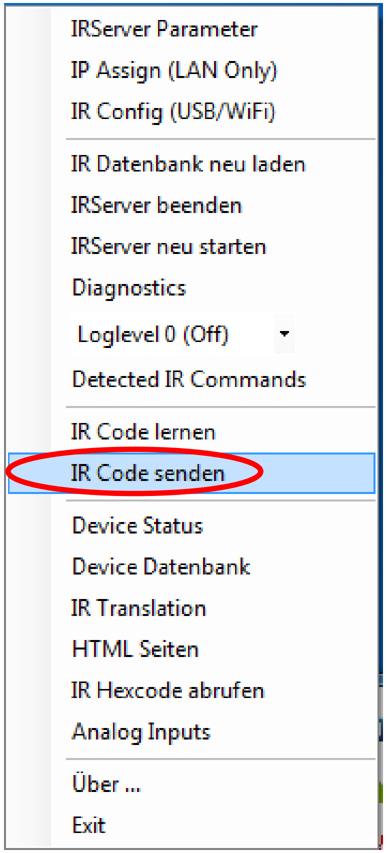
Remote: asterics

Command: rc_play

LED Select: Default

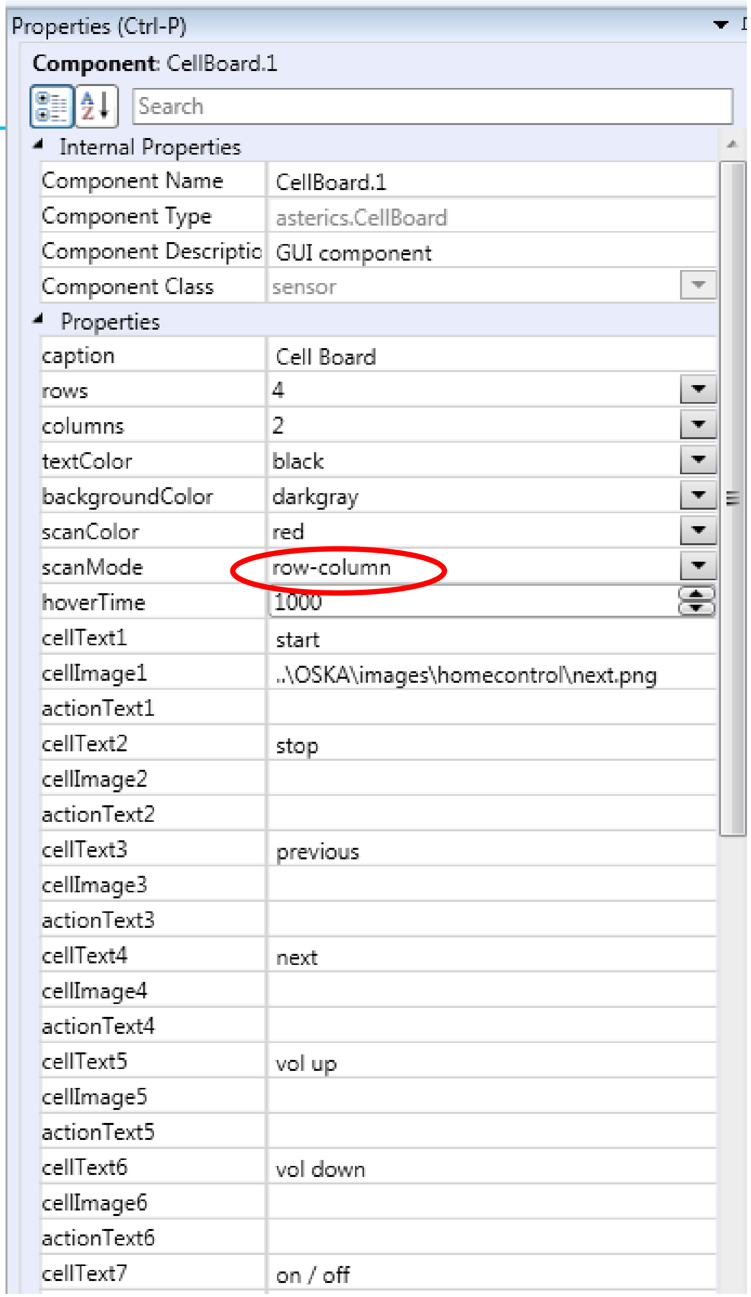
Device ID: Alle

Bus ID: Alle



Project 5 – Tips

- The **CellBoard** plugin can be used to make a „button-matrix“ with scanning selection.
- scanMode „**row-column**“ highlights rows first, then colums
- Selections are done via the **scanSelect** event listener.
- Enter caption and/or image for every used button
- Use a **Timer** plugin to trigger **scanMove** events periodically.



Properties (Ctrl-P)

Component: CellBoard.1

Search

Internal Properties

Component Name	CellBoard.1
Component Type	asterics.CellBoard
Component Descriptic	GUI component
Component Class	sensor

Properties

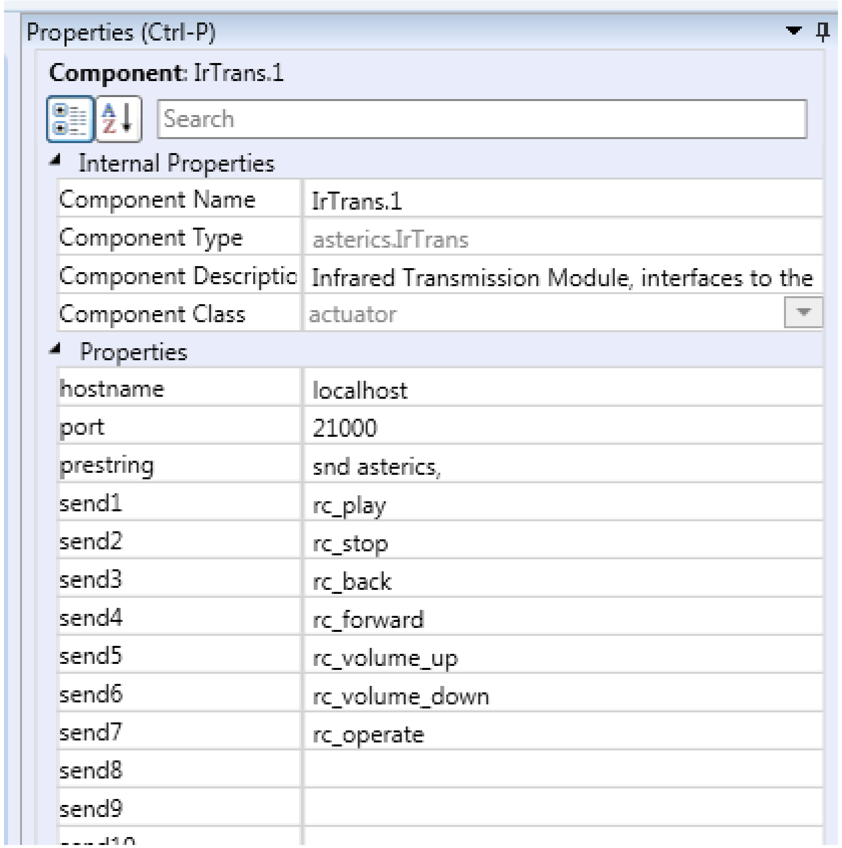
caption	Cell Board
rows	4
columns	2
textColor	black
backgroundColor	darkgray
scanColor	red
scanMode	row-column
hoverTime	1000
cellText1	start
cellImage1	..\OSKA\images\homecontrol\next.png
actionText1	
cellText2	stop
cellImage2	
actionText2	
cellText3	previous
cellImage3	
actionText3	
cellText4	next
cellImage4	
actionText4	
cellText5	vol up
cellImage5	
actionText5	
cellText6	vol down
cellImage6	
actionText6	
cellText7	on / off

Project 5 – Tips

- Use default **hostname** and **port** to connect to the local IRTrans

- **prestring** selects the remote control configuration to use, note the **comma** at the end !!

- **send1-sendN**:
  Enter the configured / stored IR codes



Properties (Ctrl-P)

Component: IrTrans.1

Search

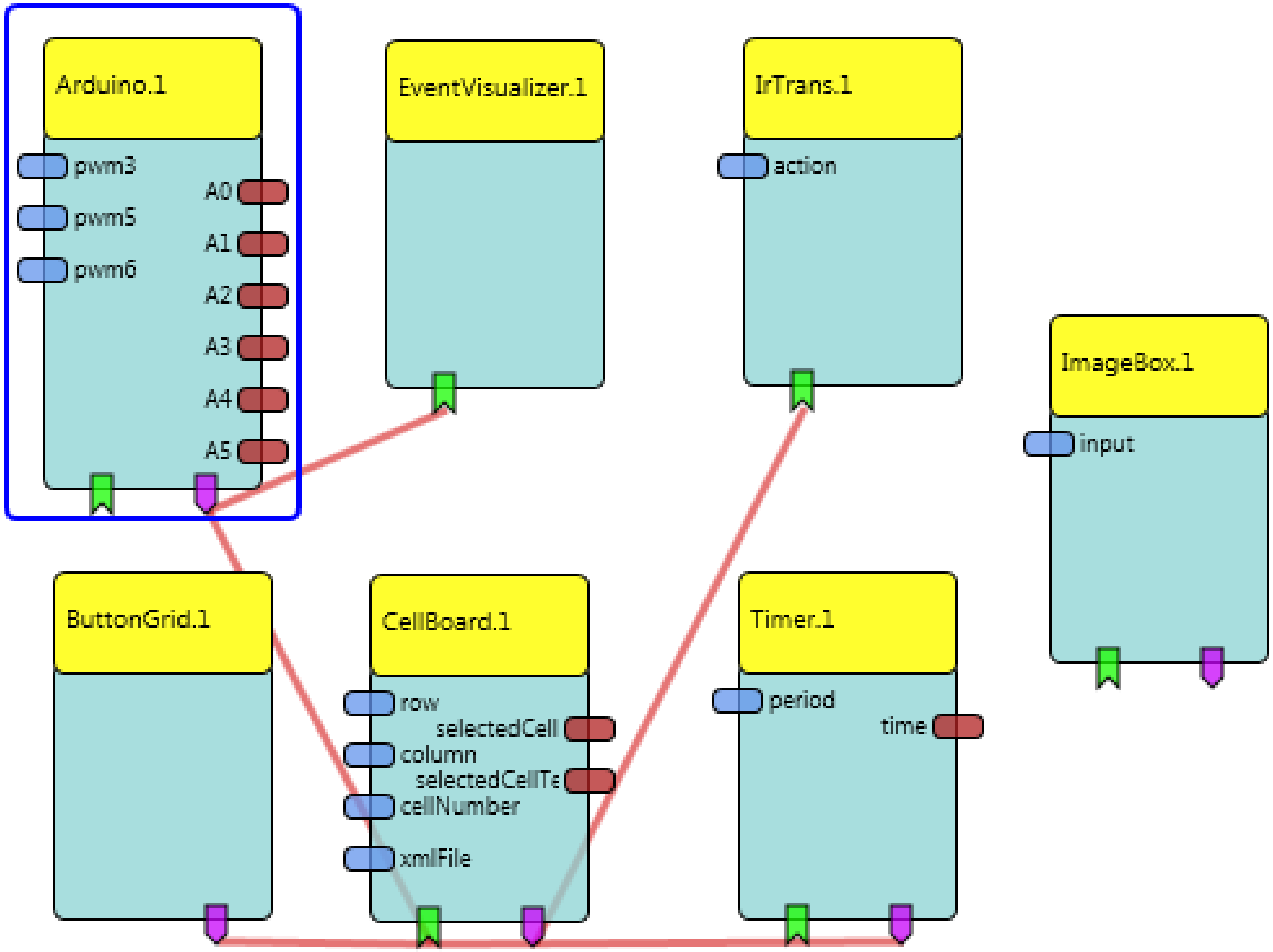
Internal Properties

Component Name	IrTrans.1
Component Type	asterics.IrTrans
Component Descriptio	Infrared Transmission Module, interfaces to the
Component Class	actuator

Properties

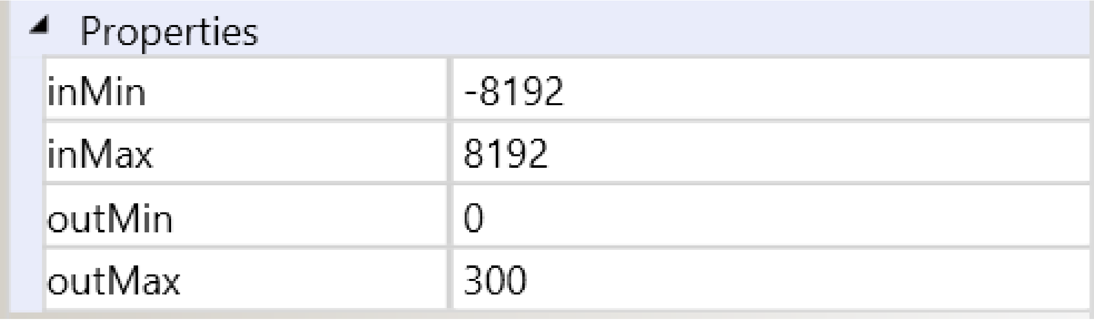
hostname	localhost
port	21000
prestring	snd asterics,
send1	rc_play
send2	rc_stop
send3	rc_back
send4	rc_forward
send5	rc_volume_up
send6	rc_volume_down
send7	rc_operate
send8	
send9	
send10	

Project 5: a possible solution …



Project 6 – Tips

- Add the **Ponggame** actuator and select the mode „**position**“
  - Input values of 0-300 are accepted for the paddles
  - The output of the used sensors must be adjusted to match this range of 0 – 300 to control the paddles
  - A „start“ event must be connected; this event could be generated via a GUI element (eg. **ButtonGrid** plugin)
- Add an **Accelerometer** sensor for paddle1
  - Use a **SignalTranslation** processor
    to map the range of the yAcc output (-8192 – 8192) to (0 – 300)



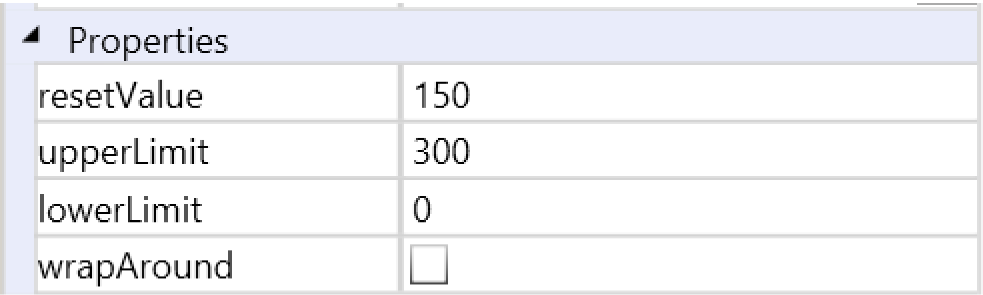
◢ Properties

inMin	-8192
inMax	8192
outMin	0
outMax	300

Project 6 – Tips

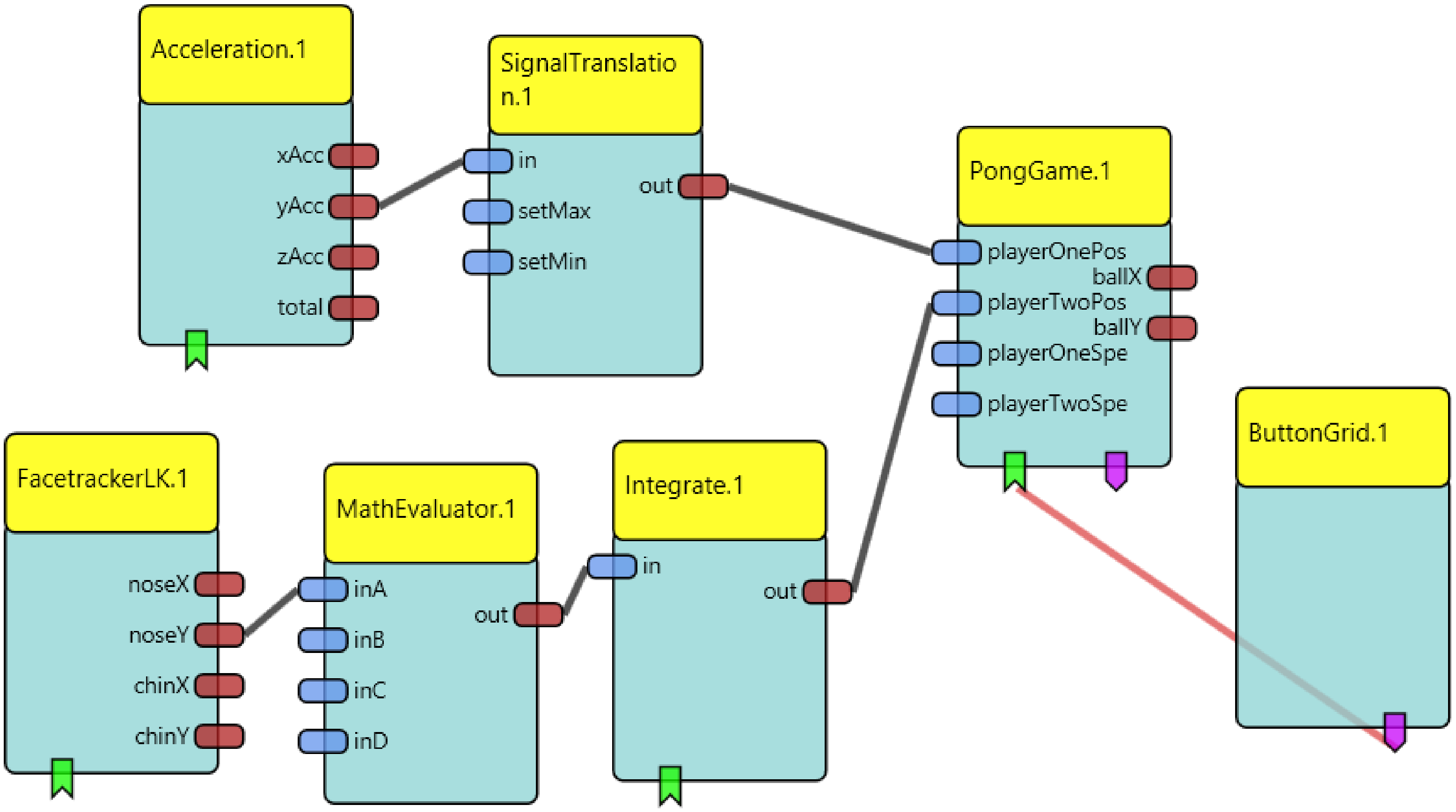
- Add the **FacetrackerLK** sensor plugin
  - The output represents head movements
    (relative changes of postion)
  - Use the nose-Y output port (for up/down movement)
  - Use a **MathEvaluator** to invert/scale the movement values as desired

- Add an **Integration** processor to accumulate the relative changes of the head movements
  - This creates an absolute position for the paddle 2 control
  - Set upperLimit and lowerLimit of the Integrator to 300/0



Properties	
resetValue	150
upperLimit	300
lowerLimit	0
wrapAround	□

Project 6: a possible solution …


Acceleration.1
xAcc
yAcc
zAcc
total
SignalTranslation.1
in
setMax
setMin
out
PongGame.1
playerOnePos
ballX
playerTwoPos
ballY
playerOneSpe
playerTwoSpe
ButtonGrid.1
FacetrackerLK.1
noseX
noseY
chinX
chinY
MathEvaluator.1
inA
inB
inC
inD
out
Integrate.1
in
out