

Lecture 8: Programming for the Arduino

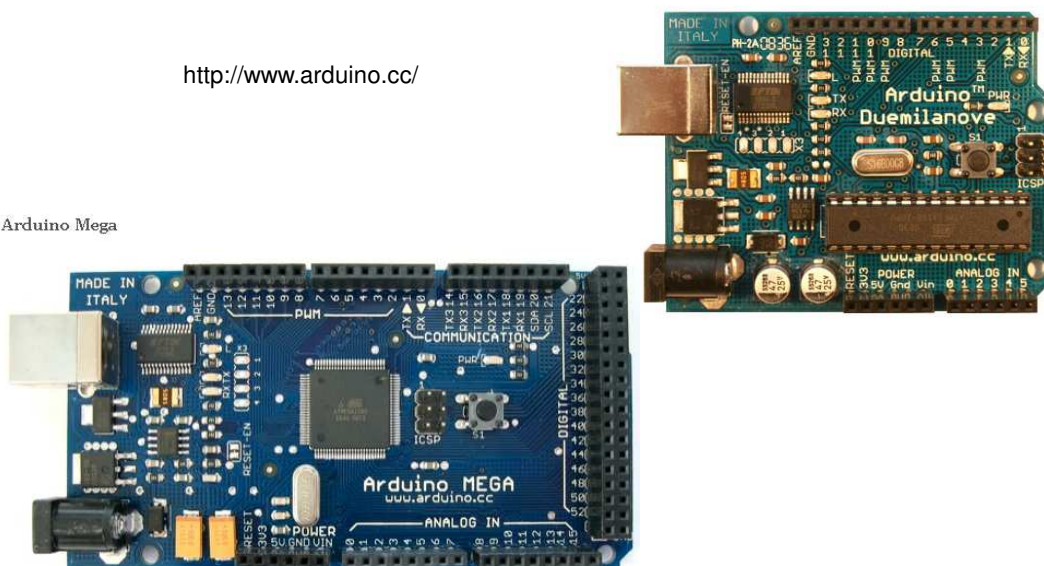
- The hardware
- The programming environment
- Binary world, from Assembler to C
- Programming C for the Arduino: Basics
- Programming C for the Arduino: more ...
- Programming style

The hardware

Arduino Duemilanove

<http://www.arduino.cc/>

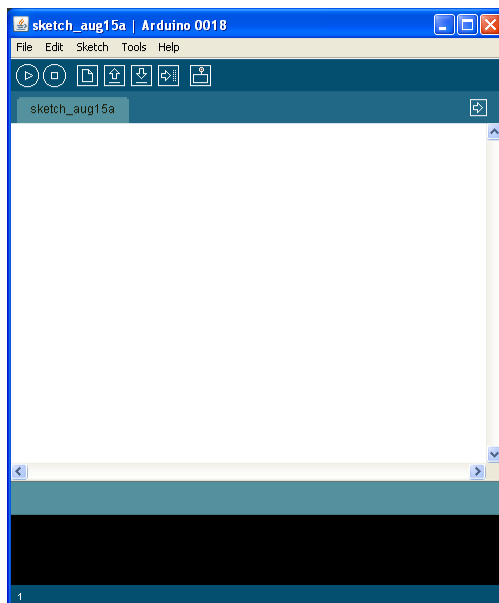
Arduino Mega



Lecture 8: Programming for the Arduino

- ✓ The hardware
 - The programming environment
 - Binary world, from Assembler to C
 - Programming C for the Arduino: Basics
 - Programming C for the Arduino: more ...
 - Programming style

Programming environment



Download from:
<http://arduino.cc/en/Main/Software>

Tools	
Auto Format	Strg+T
Archive Sketch	
Fix Encoding & Reload	
Serial Monitor	Strg+Umschalt+M
Board	▶
Serial Port	▶
Burn Bootloader	▶

Sketch	
Verify / Compile	Strg+R
Stop	
Show Sketch Folder	Strg+K
Import Library...	▶
Add File...	

Edit	
Undo	Strg+Z
Redo	Strg+Y
Cut	Strg+X
Copy	Strg+C
Copy for Forum	Strg+Umschalt+C
Copy as HTML	Strg+Alt+C
Paste	Strg+V
Select All	Strg+A
Comment/Uncomment	Strg+Slash
Increase Indent	Strg+Close Bracket
Decrease Indent	Strg+Open Bracket
Find...	Strg+F
Find Next	Strg+G

File	
New	Strg+N
Open...	Strg+O
Sketchbook	▶
Examples	▶
Close	Strg+W
Save	Strg+S
Save As...	Strg+Umschalt+S
Upload to I/O Board	Strg+U
Page Setup	Strg+Umschalt+P
Print	Strg+P
Preferences	Strg+Comma
Quit	Strg+Q

Lecture 8: Programming for the Arduino

- ✓ The hardware
- ✓ The programming environment
 - Binary world, from Assembler to C
 - Programming C for the Arduino: Basics
 - Programming C for the Arduino: more ...
 - Programming style

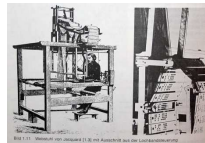
Binary world, programming from Assembler to C

Programmable, mechanical calculation machines (19th/early 20th cent. AD)

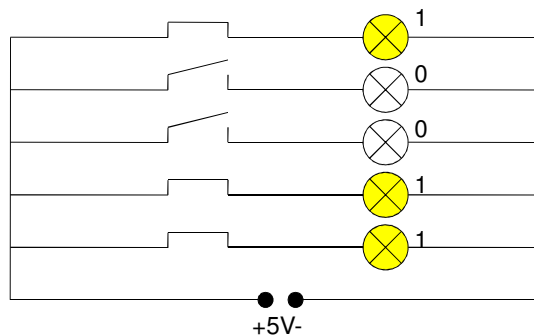
Falcon
(1728)

**Joseph-Marie
Jacquard**
(1805)

-> punchcard looms
for work steps
(program)



- 1 Origins at China and also developed by the mathematician Leibniz (17th century AD)
- 1 Positional numeral system which represents each number just with 2 symbols, 0 and 1
- 1 These values can be represented by voltage levels in electronic circuits
- 1 For human use very inefficient but with electronic circuits it is possible to create very efficient arithmetic and logic units (ALUs) for the basic operations addition, subtraction, multiplication and division

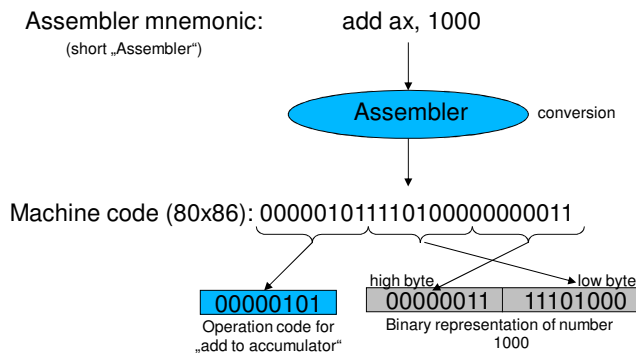


See: Rembold, Ulrich et. al.: Einführung in die Informatik für Naturwissenschaftler und Ingenieure. Hanser München Wien 1991
 See: <http://privat.svol.dia/Sven/Bandell/index.html>, Download 01.09.2007
 See: <http://en.wikipedia.org/wiki/Image:Boullier1.JPG>, 01.09.2007

See: Rembold, Ulrich et. al.: Einführung in die Informatik für Naturwissenschaftler und Ingenieure. Hanser München Wien 1991

Binary world, programming from Assembler to C

The machine instructions



Binary world, programming from Assembler to C

Programming paradigm of the problem oriented computer language C

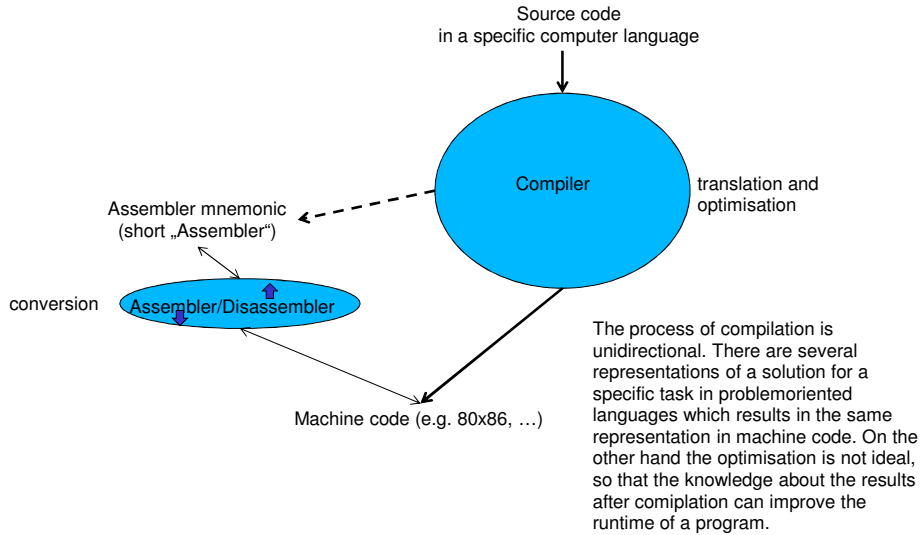
- Procedural programming with structured programming as subset
 - Code is splitted into several, reusable sections called procedures or functions with own scopes, which can be called at given code positions
 - Logical procedure units are combined to modules
 - Jumps (like goto) are not allowed
 - E.g. C
 - Case sensitive

```

Blink | Arduino 0018
File Edit Sketch Tools Help
Blink $
int ledPin = 13; // LED connected to digital pin 13
// The setup() method runs once, when the sketch starts
void setup()
{
  // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);
}
// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
  digitalWrite(ledPin, HIGH); // set the LED on
  delay(1000); // wait for a second
  digitalWrite(ledPin, LOW); // set the LED off
  delay(1000); // wait for a second
}
    
```

Binary world, programming from Assembler to C

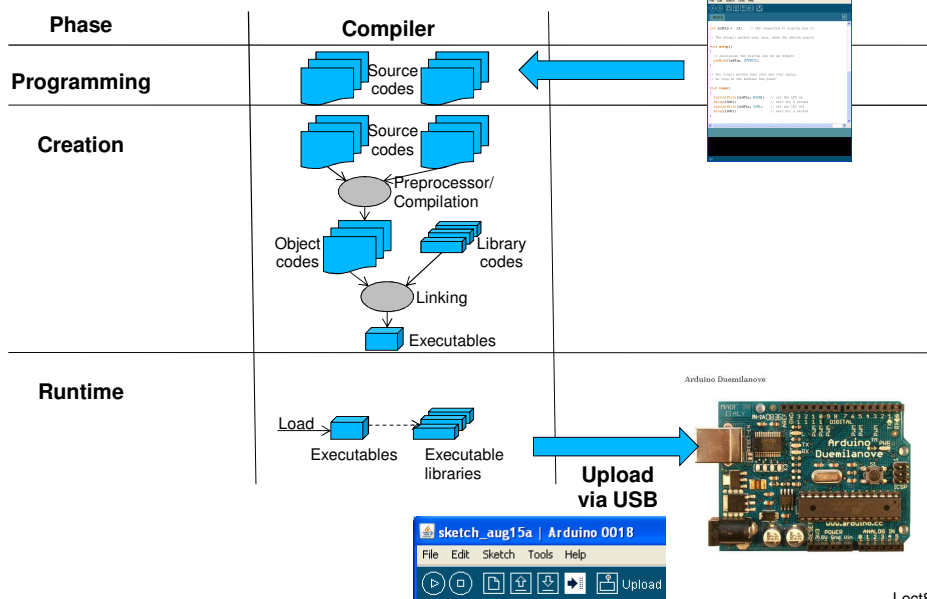
From source code to machine instructions



See: <http://vs.cs.uni-magdeburg.de/bs/lehre/sose99/bs1/seminare/assembler.shtml>, Download 30.08.2007

Lect8-Page9

Binary world, programming from Assembler to C



Lect8-Page10

Lecture 8: Programming for the Arduino

- ✓ The hardware
- ✓ The programming environment
- ✓ Binary world, from Assembler to C
 - Programming C for the Arduino: Basics
 - Programming C for the Arduino: more ...
 - Programming style

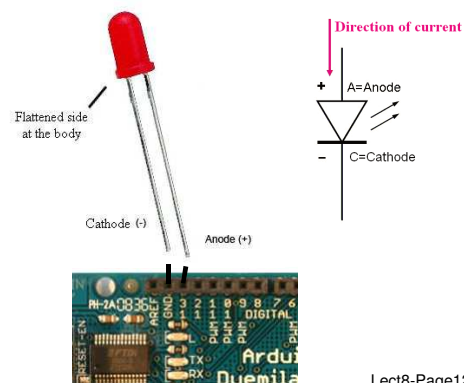
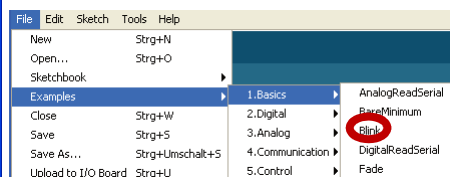
Programming C for the Arduino: Basics

Our first program: Blink (LED)

```

Blink $
Arduino 0018
File Edit Sketch Tools Help

Blink $
int ledPin = 13; // LED connected to digital pin 13
// The setup() method runs once, when the sketch starts
void setup()
{
  // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);
}
// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
  digitalWrite(ledPin, HIGH); // set the LED on
  delay(1000); // wait for a second
  digitalWrite(ledPin, LOW); // set the LED off
  delay(1000); // wait for a second
}
    
```



Programming C for the Arduino: Basics

```

Blink | Arduino 0018
File Edit Sketch Tools Help
Blink $
int ledPin = 13; // LED connected to digital pin 13

// The setup() method runs once, when the sketch starts

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

// the loop() method runs over and over again,
// as long as the Arduino has power

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

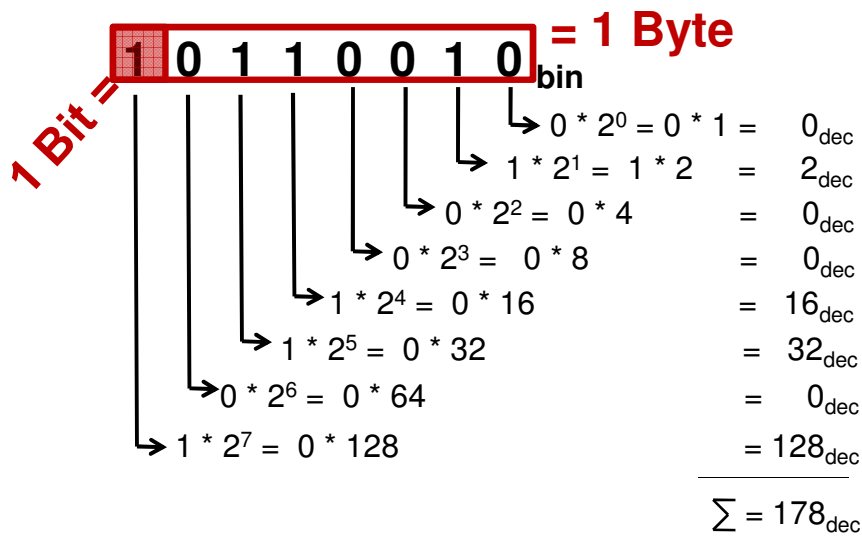
Memory Elements



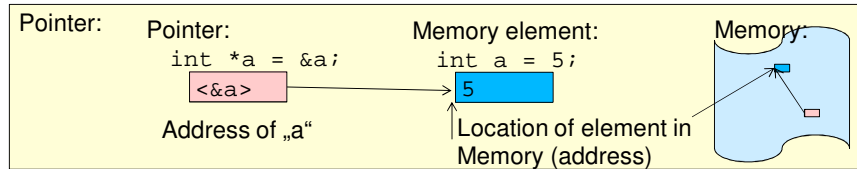
short, int, long,
float, double,
char,
structures,
...

Variables
Arrays (e.g. int Ar[5];)
Indexes start with 0!!!
Pointers

Programming C for the Arduino: Basics



Programming C for the Arduino: Basics



Programming C for the Arduino: Basics

```

Blink | Arduino 0018
File Edit Sketch Tools Help

Blink $

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

// The setup() method runs once, when the sketch starts
void setup()
{
  // initialize the digital pin as an output:
  pinMode(ledPin, OUTPUT);
}

// the loop() method runs over and over again,
// as long as the Arduino has power
void loop()
{
  digitalWrite(ledPin, HIGH); // set the LED on
  delay(1000); // wait for a second
  digitalWrite(ledPin, LOW); // set the LED off
  delay(1000); // wait for a second
}
    
```

Functions

Functiondefinition

```

int Add (int a, int b)
{
  return a+b;
}
    
```

Functioncall

```

X = Add (5, 6);
=> X=11
    
```

Programming C for the Arduino: Basics

Operators	Standard operator:
Assign	=,
Plus	+,
Minus	-,
Multiplication	*,
Division	/,
Modulo-Div.	%,
AND	&& (Bit-AND &)
OR	(Bit-OR)
NOT	!
	...
	<u>Additional operators:</u>
	Add one, subtract one:
	++,--, (e.g. i++; is i=i+1)
	and a lot of others:
	+=, -=, ..., [,], ->, *, ...

Lect8-Page17

Programming C for the Arduino: Basics



```

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

// The setup() method runs once, when the sketch starts

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

// the loop() method runs over and over again,
// as long as the Arduino has power

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

```

Comments

// This is a comment

/* This is also a comment */

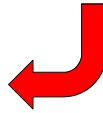
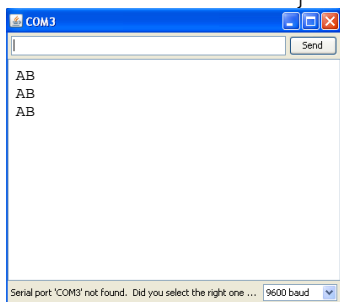
Lect8-Page18

Programming C for the Arduino: Basics

Interaction with users

Serial write

```
void setup() {
  // initialize serial communication:
  Serial.begin(9600);
}
void loop()
{
  Serial.print("On"); // Write without new line
  Serial.println("Off"); // Write with new line
}
```



Lect8-Page19

Programming C for the Arduino: Basics

Application workflow

Conditions

```
if (iIndex < 10)
{
  Serial.print("A");
}
else
{
  Serial.print("B");
}
```

Conditions

```
switch (iIndex)
{
  case 1:
    Serial.print("A");
    break;
  case 2:
    Serial.print("B");
    break;
  ...
}
```

Lect8-Page20

Programming C for the Arduino: Basics

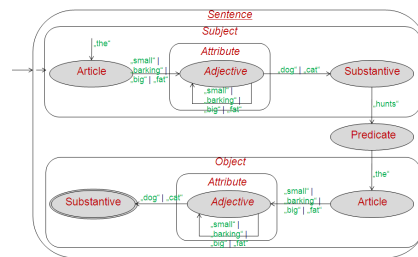
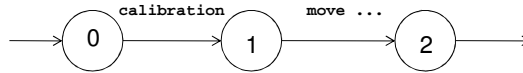
Realizing a state machine using switch condition

```

int iState;

void setup ()
{
  iState = 0;
}

void loop ()
{
  switch (iState)
  {
    case 0:
      //calibration
      iState = 1;
      break;
    case 1:
      // move just forward as long
      // as wall closer than 10 cm
      iState = 2;
      break;
    case 2:
      ...
  }
}
    
```



Programming C for the Arduino: Basics

Application workflow

Loops

```

int i = 0;
while (i < 10)
{
  Serial.print(i);
  i = i + 1;
}
    
```

Loops

```

int i;
for (i = 1; i < 10; i++)
{
  Serial.print(i);
}
    
```

Programming C for the Arduino: Basics

Digital Pins

Output

```
pinMode(iPingPin, OUTPUT);
digitalWrite(iPingPin, LOW);
delayMicroseconds(2);
digitalWrite(iPingPin, HIGH);
delayMicroseconds(10);
digitalWrite(iPingPin, LOW);
```

Input

```
pinMode(iPingPin, INPUT);
long duration = pulseIn(iPingPin, HIGH);
```

Programming C for the Arduino: Basics

External modules (Libraries)

Creation

Copy library modules to here

Use module with #include <...>

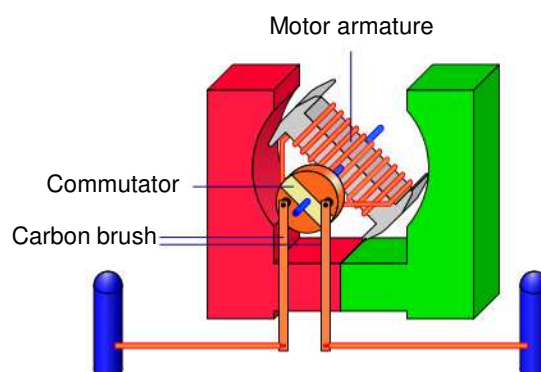
Lecture 8: Programming for the Arduino

- ✓ The hardware
- ✓ The programming environment
- ✓ Binary world, from Assembler to C
- ✓ Programming C for the Arduino: Basics
 - Programming C for the Arduino: more ...
 - Programming style

Lect8-Page25

Programming C for the Arduino: more ...

Motordriver for DC-motors: basic functionality



<http://de.wikipedia.org/w/index.php?title=Datei:Gleichstrommaschine.svg&filetimestamp=20070820012802>
<http://www.hpw-modellbahn.de/eisenbahntechnik/motoren.htm>

Lect8-Page26

Programming C for the Arduino: more ...

Motordriver for DC-motors: basic functionality

http://www.m-wissen.de/index.php/Getriebemotoren_Ansteuerung

Lect8-Page27

Programming C for the Arduino: more ...

Motordriver for DC-motors: basic functionality (with driver IC L293 D)

http://www.m-wissen.de/index.php/Getriebemotoren_Ansteuerung

Lect8-Page28

Programming C for the Arduino: more ...

Motordriver for DC-motors: basic functionality (with driver IC L293 D)

Enable = speed (PWM)

Rotation direction and On/Off

+/- rotate forward

-/+ rotate backward

+/- or +/- stop

Arduino Duemilanove

Beliebte H-Brücke für zwei Motoren

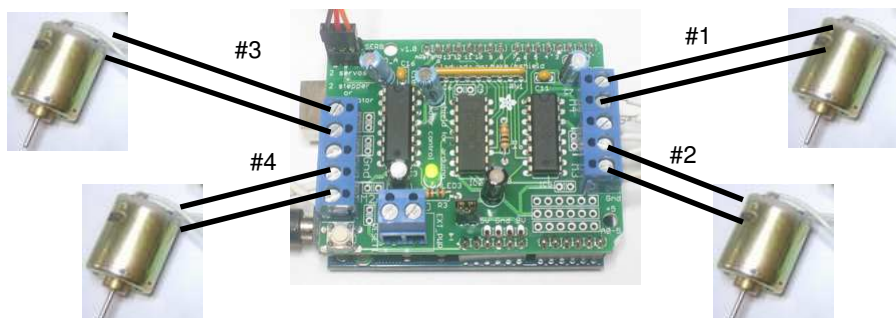
GND Motorspannung 6 - 36V

Schaltung von	roboternetz-wiki
www.roboternetz.de	hbruecke 1293d
	28.06.2005 21:19:25
	Sheet 1/1

http://www.m-wissen.de/index.php/Getriebemotoren_Ansteuerung
Lect8-Page29

Programming C for the Arduino: more ...

**Motordriver for DC-motors:
Adafruit Motor/Stepper/Servo Shield for Arduino kit - v1.0**



Programming C for the Arduino: more ...

**Motordriver for DC-motors:
Adafruit Motor/Stepper/Servo Shield for Arduino kit - v1.0**

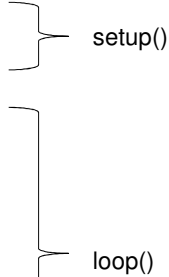
```

// Use Adafruit module
#include <AFMotor.h>

// create motor #2, 64KHz pwm
AF_DCMotor MotorRight(2, MOTOR12_64KHZ);

// set the speed
MotorRight.setSpeed(50);

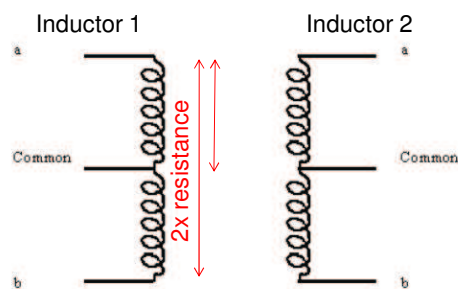
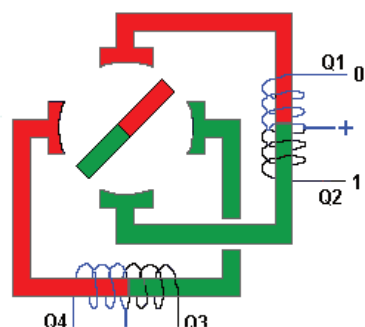
// Move forward
for (int i=0;i<500;i++)
{
    MotorRight.run(FORWARD);
}
// Move backward
for (int i=0;i<500;i++)
{
    MotorRight.run(BACKWARD);
}
    
```



<http://www.ladyada.net/make/mshield/>
Lect8-Page31

Programming C for the Arduino: more ...

Motordriver for step-motors (steppers): basic functionality

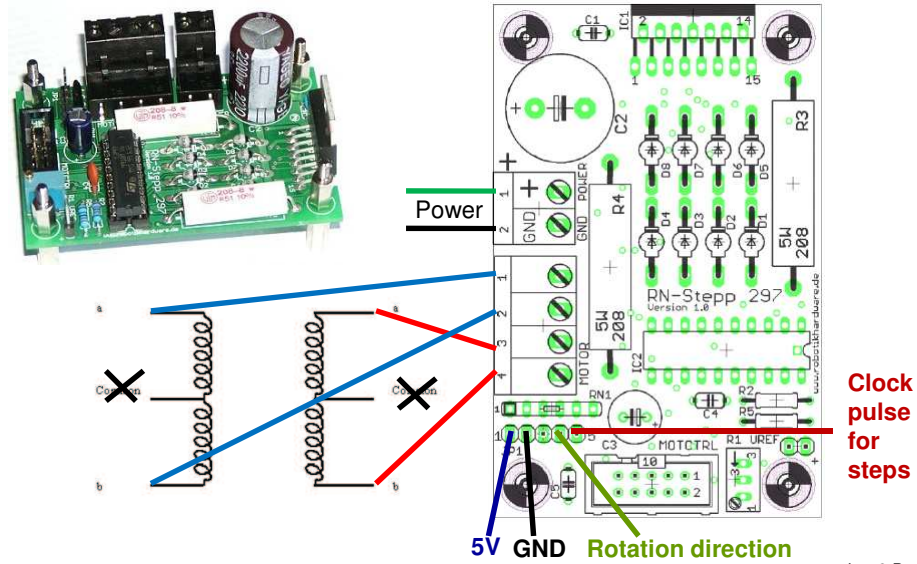


<http://www.shop.robotikhardware.de/>
<http://www.rn-wissen.de/index.php/Schrittmotoren>
<http://www.schule-bw.de/unterricht/faecher/physik/mess/soundkarte/hardware/interfsound/schrittschliff.htm>

Lect8-Page32

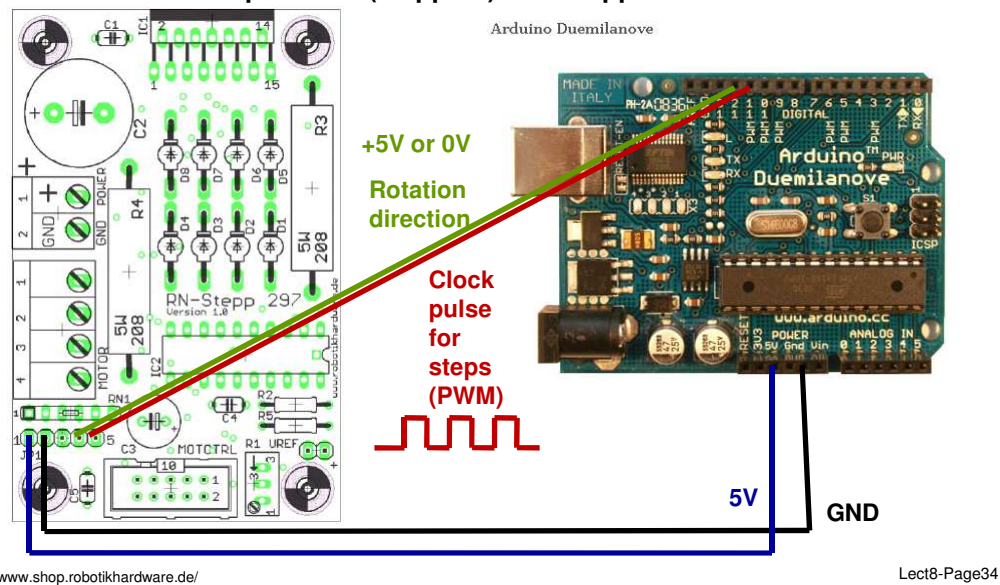
Programming C for the Arduino: more ...

Motordriver for step-motors (steppers): RN-Stepp297



Programming C for the Arduino: more ...

Motordriver for step-motors (steppers): RN-Stepp297



Programming C for the Arduino: more ...

Motordriver for step-motors (steppers): RN-Stepp297 - programming

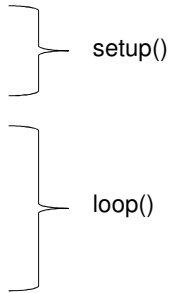
```
// Use stepper module
#include <Stepper.h>

// Create stepper class variable
Stepper StepperLeft (steps, pin_direction, pin_clockpulse);
  e.g.  StepperLeft (200, 8, 9)
       StepperRight (200, 7, 10)

// set the speed of the motor to 200 RPMs
StepperLeft.setSpeed(200);

// Move one step forward
StepperLeft.step(1);

// Move one step backward
StepperLeft.step(-1);
```

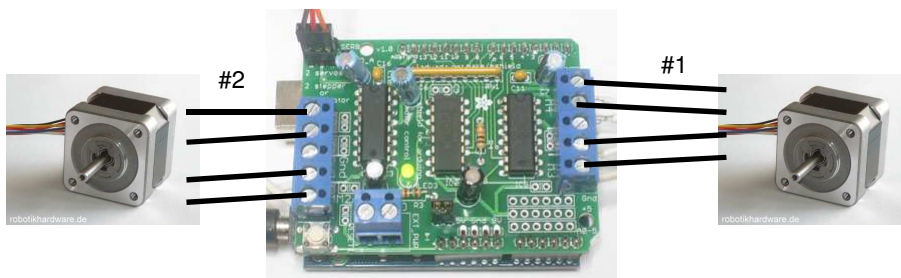


<http://arduino.cc/en/Reference/Stepper?from=Tutorial.Stepper>

Lect8-Page35

Programming C for the Arduino: more ...

Motordriver for step-motors (steppers): Adafruit Motor/Stepper/Servo Shield for Arduino kit - v1.0



Lect8-Page36

Programming C for the Arduino: more ...

**Motordriver for step-motors (steppers):
Adafruit Motor/Stepper/Servo Shield for Arduino kit - v1.0**

```

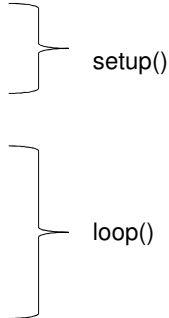
// Use Adafruit module
#include <AFMotor.h>

// create motor stepper #2 with 48 steps
AF_Stepper StepperLeft(48, 2);

// set the speed
StepperLeft.setSpeed(60);
StepperLeft.release();

// Move forward
StepperLeft.step(1, FORWARD, SINGLE);

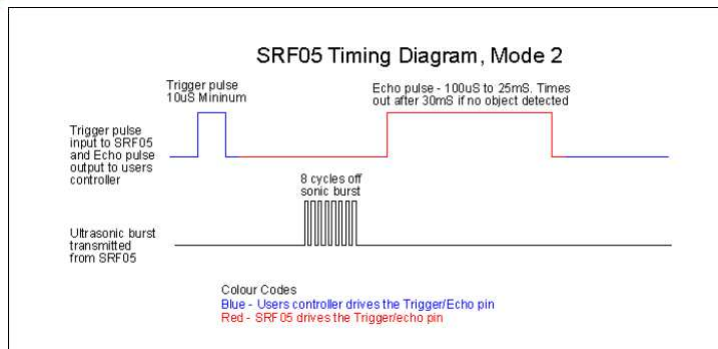
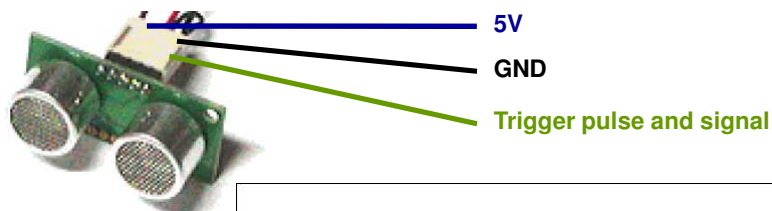
// Move backward
StepperLeft.step(1, BACKWARD, SINGLE);
    
```



<http://www.ladyada.net/make/mshield/>

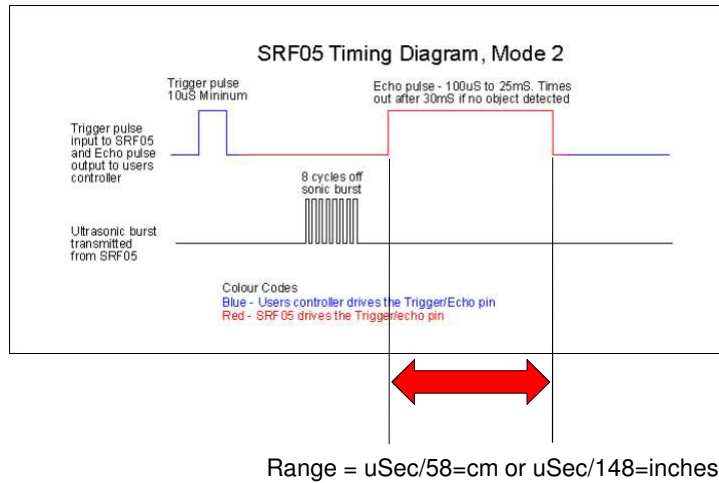
Programming C for the Arduino: more ...

Low Cost Ultra Sonic Range Finder



Programming C for the Arduino: more ...

Low Cost Ultra Sonic Range Finder



Programming C for the Arduino: more ...

Low Cost Ultra Sonic Range Finder

e.g. `int iPingPin = 13;`

SRF05 Timing Diagram, Mode 2

Trigger pulse 10µs Minimum

Echo pulse - 100µs to 25ms. Times out after 30ms if no object detected

Trigger pulse input to SRF05 and Echo pulse output to users controller

8 cycles off sonic burst

Ultrasonic burst transmitted from SRF05

Colour Codes
Blue - Users controller drives the Trigger/Echo pin
Red - SRF05 drives the Trigger/Echo pin

Range = uSec/58=cm or uSec/148=inches.

```

pinMode(iPingPin, OUTPUT);
digitalWrite(iPingPin, LOW);
delayMicroseconds(2);
digitalWrite(iPingPin, HIGH);
delayMicroseconds(10);
digitalWrite(iPingPin, LOW);

pinMode(iPingPin, INPUT);
digitalWrite(iPingPin, HIGH);
    
```

Programming C for the Arduino: more ...

Robot Compass Modul

CMPSD3
Kompassmodul



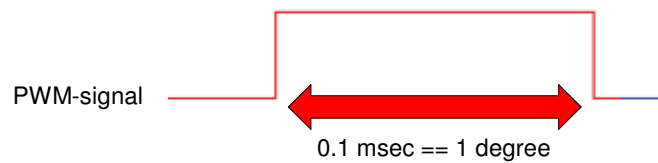
Pin 9 - GND (Masse)
Pin 8 - Nicht belegt
Pin 7 - 50 Hz Wechsellspannung Taktvorgang
(bei 50 Hz einfach unbelogt steuern)
Pin 6 - Nur bei Kalibrierung notwendig
(für jede Himmelsrichtung auf GND ziehen)
Pin 5 - nicht belegt
Pin 4 - Ergebnis (PWM)
Pin 3 - I2C - SDA
Pin 2 - I2C - SCL
PIN 1 - +5V

GND

Signal (PWM)

5V

Skizze / Übersetzung: www.robothardware.de



e.g. `int iCompassPin = 11;`

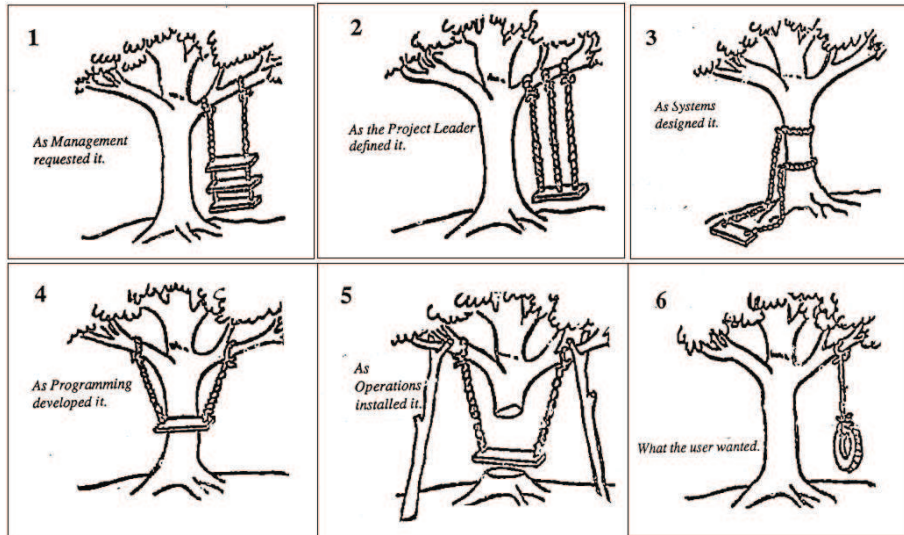
Lect8-Page41

Lecture 8: Programming for the Arduino

- ✓ The hardware
- ✓ The programming environment
- ✓ Binary world, from Assembler to C
- ✓ Programming C for the Arduino: Basics
- ✓ Programming C for the Arduino: more ...
- Programming style

Lect8-Page42

Programming style



See: <http://casablanca.informatik.hu-berlin.de/database/MDA-UML/VL/V1-MODSOFT-1.1.Ueberblick.pdf>, Download 07.11.2006

Lect8-Page43

Programming style

Try to write readable and efficient code:

- Add comments to explain the code
- Use understandable (variable, function) names
- Structure the code into logical units
- Initialize variables
- Search for existing code first
- ...

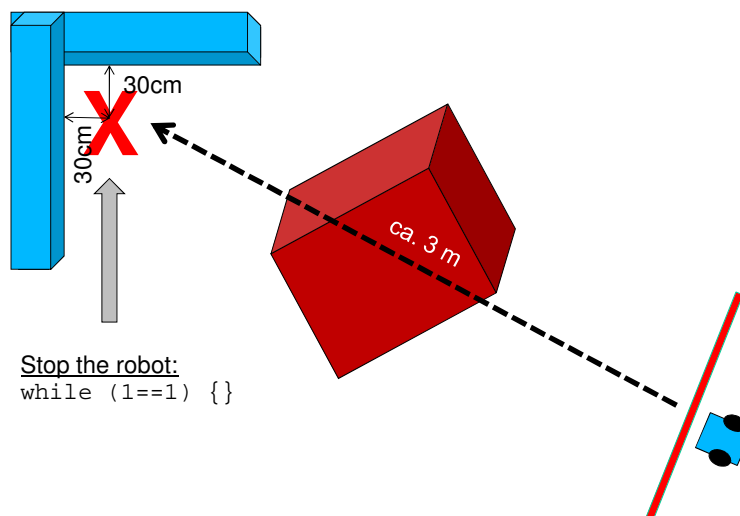
Lect8-Page44

Programming style

And for help look at:



The goal!



Thank you