Intelligente Himbeere

Der Raspberry Pi

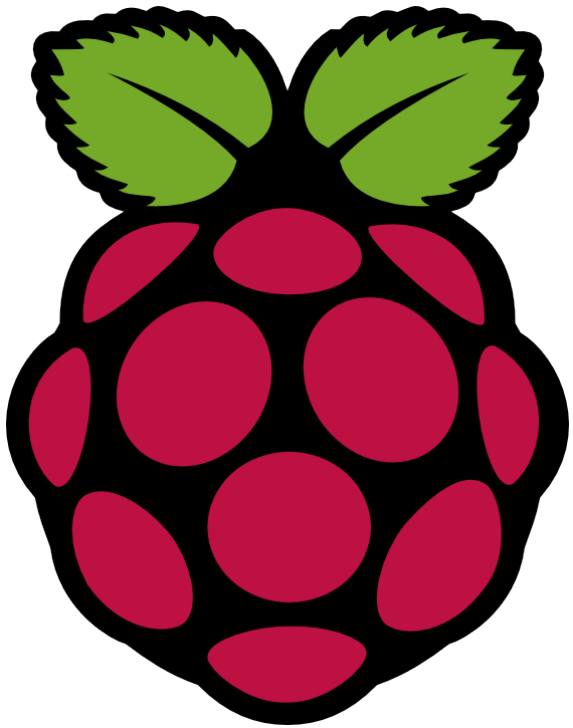
Ralph Sontag, Andreas Heik

TU-Chemnitz

19. April 2013

Agenda

- Raspberry Pi Foundation
- Anatomie einer Himbeere
- Quickstart
- Der Raspi im WLAN
- Media Center
- Low-Level Peripherie
- Physical Computing - Roboter
- Visionen



Motivation

Raspberry Pi Foundation

„Die **Raspberry Pi Foundation** ist eine Stiftung und in Großbritannien als Wohltätigkeitsorganisation eingetragen. Sie hat sich zum Ziel gesetzt, das Studium der Informatik und verwandter Themen zu fördern, insbesondere im Schulbereich. Dazu entwickelt sie mit dem *Raspberry Pi* einen erschwinglichen kreditkartengroßen Computer, der zu Experimenten reizen soll. Dabei stützt man sich, wie in der Anfangszeit der Heimcomputer (z. B. IMSAI 8080, Apple I, Sinclair ZX80), bewusst auch auf den Spaßfaktor beim Erlernen der Computergrundlagen und -programmierung.

Die Raspberry Pi Foundation wurde am 5. Mai 2009 in Caldecote, South Cambridgeshire, Großbritannien gegründet.“ [1]

[1] http://de.wikipedia.org/wiki/Raspberry_Pi

Equipment

- **Raspberry Pi**
- SD-Karte ab 2 GB (Speed-Class beachten!)
- Netzteil mit Micro-USB Stecker (wenigstens *1 A*)
- USB-Tastatur und optional Maus
- Monitor (HDMI, DVI oder SCART)
- aktiver USB-Hub (auch als Stromversorgung)
- Internetverbindung (Kabel oder WLAN)
- Gehäuse
- Breadboard und elektronische Bauteile
  (LED, Taster, Widerstände, Sensoren, . . .)
- Lizenzen für MPEG2- und VC1-Codecs

Anatomie einer Himbeere

Raspberry Pi - Modell B

- SoC (System on a Chip) Broadcom 2835
  ARM1176JZF-S (700 MHz) + VideoCore IV
- 512 MB SDRAM (gestapelt unter SoC) *)
- 2 USB-2.0 Ports *)
- Netzwerk 10/100 MBit (onboard) *)
- *Video:* HDMI, Composite Video
- *Audio:* HDMI, 3.5mm Klinke
- SD/MMC Kartenslot
- Low-Level Peripherie (GPIO, UART, I2C- und SPI-Bus)
- Leistungsaufnahme ca. 3.5 W *)

*) Unterschiede zum Modell A

Quickstart

- Raspberry Pi Foundation empfielt die auf Debian basierte Distribution *Raspbian Wheezy*[2]
- Installation des Image auf SD-Karte:

```
sudo dd if=2013-02-09-wheezy-raspbian.img \
        of=/dev/SD-Karte bs=1M
```

  → */boot* (vfat) + *Root-Filesystem* (ext4)
- **Power On!**
- Konfigurationsmenü nach ersten Start
  Tastatur, *Spracheinstellung*, *Zeitzone*, *expand_rootfs*
- Anmeldung mit Loginnamen `pi` und Passwort `raspberry`
- *. . . und nun?*

[2] http://www.raspbian.org/

Quickstart 2

- System aktualisieren:

```
sudo apt-get update
sudo apt-get upgrade
```

- Multimedia-Fähigkeiten der GPU mit Video im H.264-Format testen:

```
omxplayer --adev hdmi big_buck_bunny_1080p_h264.mov
```

- grafische Oberfläche starten:

```
startx
```

Der Raspberry Pi mit *wheezy*-Distribution ist ein **vollwertiges System** mit Entwicklungsumgebung (`Compiler, ...`), aber weniger als Desktop-Ersatz geeignet.

Stolpersteine

- Stromversorgung (Belastbarkeit des Netzteil)
- Video-Modes und die Monitor-Auflösung[3]

  `/opt/vc/bin/tvservice`

- Computer-Monitor und Audio über HDMI

  `config.txt: hdmi_drive=2`

- siehe auch: R-Pi Troubleshooting[4]

[3] http://elinux.org/RPiconfig#Video
[4] http://elinux.org/R-Pi_Troubleshooting

Der Bootprozess

first stage bootloader	fest in SoC programmiert, montiert FAT32-Partition der SD-Karte
second stage bootloader `(bootcode.bin)`	lädt die Firmware der GPU und startet die GPU
GPU firmware `(start.elf)`	GPU startet die CPU, konfiguriert SDRAM für GPU und CPU
User code `(kernel.img)`	typischerweise der Linux-Kernel
`(config.txt)`	System Konfigurationsparameter[5] ✎ GPU Memory ✎ Videomode ✎ Lizenzen (Videocodecs) ✎ Boot-Parameter, Overclocking

[5] http://elinux.org/RPiconfig

Raspberry Pi im WLAN

- *wheezy*-Distribution enthält bereits Treiber und Firmwarepakete für gängige USB-WLAN Sticks
- Informationen über den USB-WLAN Stick im `/var/log/syslog` und mittels `/usr/bin/lsusb`
- Konfiguration des *wpa_supplicant*:

```
wpa_passphrase "essid" "passphrase" \
     >> /etc/wpa_supplicant/wpa_supplicant.conf
```

- Funknetz starten `sudo ifup wlan0`
- Autostart des Netzwerkinterface `wlan0` in `/etc/network/interfaces` konfigurieren:

```
auto wlan0
iface wlan0 inet dhcp
wpa-conf /etc/wpa_supplicant/wpa_supplicant.conf
```

Raspberry Pi als Access Point

- Access Point mit Hilfe des `hostapd`-Daemon
- ☞ Treiber des USB-WLAN Stick muß den Access Point-Mode[6] unterstützen
- Konfiguration als Bridge:

```
ifconfig eth0 0.0.0.0 up
ifconfig wlan0 0.0.0.0 up
hostapd /etc/hostapd/hostapd.conf
brctl addbr br0
brctl addif br0 eth0
brctl addif br0 wlan0
ifconfig br0 192.168.0.20 netmask 255.255.255.0 up
```

- Konfiguration als Router:
  - z.B. mittels `iptables` (NAT) und `dnsmasq`

[6] http://wireless.kernel.org/en/users/Drivers

Bunt und Laut - Media Center

Der Broadcom „full HD, multimedia applications processor“[7] und der HDMI-Ausgang prädestinieren den Raspi für den Einsatz als Media Center.

- Music Player Daemon (MPD)[8]
- verschiedenen XBMC-Distributionen[9]
- Lizenzen für *MPEG2-* und *VC1-Codecs* von Raspbmc unterstützt
- Fernsteuerung via Infrarot, Web und App
- ...

[7] http://www.broadcom.com/products/BCM2835
[8] http://www.musicpd.org/
[9] http://wiki.xbmc.org/index.php?title=Raspberry_Pi

Low-Level Peripherie

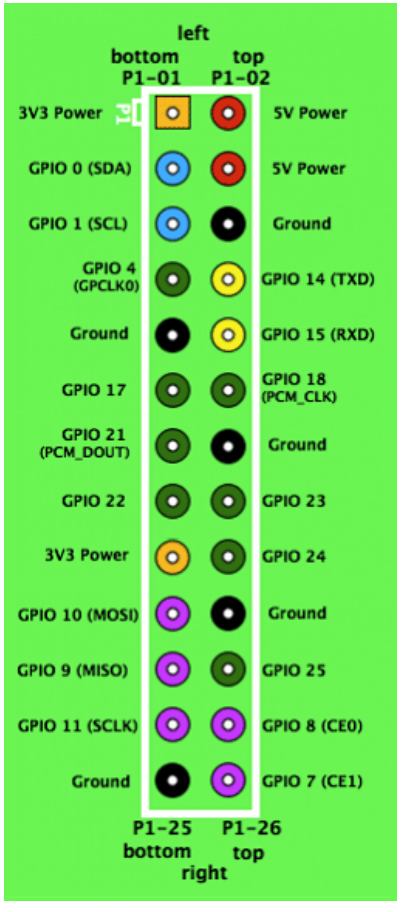
Die Stiftleiste mit herausgeführten GPIO-Pins[10] macht den Raspberry Pi zu einer interessanten Plattform für Bastelpojekte.
☞ Vor dem Anschluß von Tastern, LEDs, Sensoren oder anderen Bausteinen unbedingt die Hinweise zu Belastbarkeit und Spannungsfestigkeit beachten!

- ✗ 3.3V Spannung mit maximal 50*mA* belastbar
- ✗ 5V wird vom Netzteil direkt versorgt
- ✗ Signalpegel der GPIOs ist 3.3V
  (Eingänge sind nicht 5V-tolerant)
- ✗ Pinbelegung von der Board-Revision abhängig

[10] http://elinux.org/RPi_Low-level_peripherals

Low-Level Peripherie

- Low-Level Peripherie liegt im physischen Adressraum `0x20000000` bis `0x20FFFFFF`[a]
- unterstützt durch Kernelmoduln mit korespondierenden Files im `/dev` und `/sys` Filesystem
- Bibliotheken für verschiedene Programmiersprachen verfügbar
  - Python-Module *quick2wire*[b]
  - C-Bibliothek *bcm2835*[c]
  - ...



[a] siehe: Dokument BCM2835-ARM-Peripherals

[b] http://quick2wire.com/

[c] http://www.open.com.au/mikem/bcm2835/

GPIO-Pins

- frei programmierbar als Ein- oder Ausgang
- integrierte Pullup/Pulldown Widerstände
- Eventbehandlung, z.B. Flankenwechsel (rising/falling edge)
- 1 Hardware PWM Ausgang (GPIO 18)

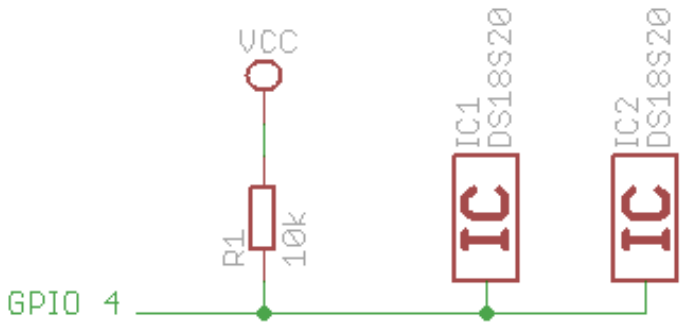
Programmierbeispiel in Shell:

```
# GPIO 17 als Ausgang
echo "17" > /sys/class/gpio/export
echo "out" > /sys/class/gpio/gpio17/direction

# GPIO 17 auf High-Pegel
echo "1" > /sys/class/gpio/gpio17/value
```

1-Wire-Bus

- Bausteine am 1-Wire-Bus mit Stromversorgung und **einer** Signalleitung angeschlossen (Pullup Widerstand erforderlich!)
- Datenübertragung asynchron (Sampling im μs-Bereich)
- Raspberry Pi als Master
- Adressierung der Slaves über deren eindeutige 64bit ROM-Adresse



1-Wire-Bus

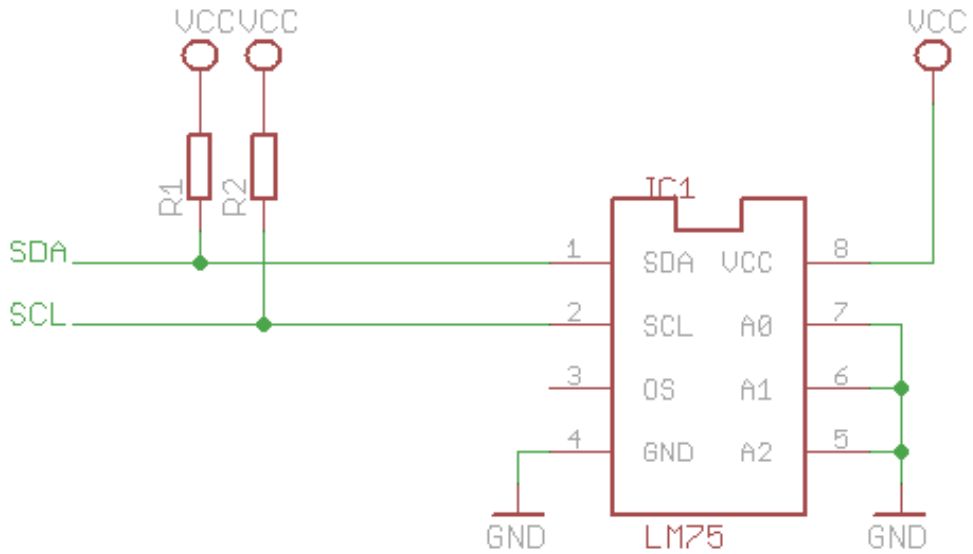
- Unterstützung durch Kernelmoduln für Bus und diverse Slaves *(„parasit power“ funktioniert nicht)*

```
modprobe wire
modprobe w1-gpio
modprobe w1-therm
cat /sys/bus/w1/devices/22-00000022a744/w1_slave
 58 01 4b 46 7f ff 08 10 f9 : crc=f9 YES
 58 01 4b 46 7f ff 08 10 f9 t=21500        <-- 21.5 Grad
```

- *Experiment:*
  Implementation des 1-Wire Protokolls im userland
  ca. 20% Übertragungsfehler (CRC)
  ☞ „parasit power“ funktioniert
  ✗ ROM_SEARCH nicht implementiert

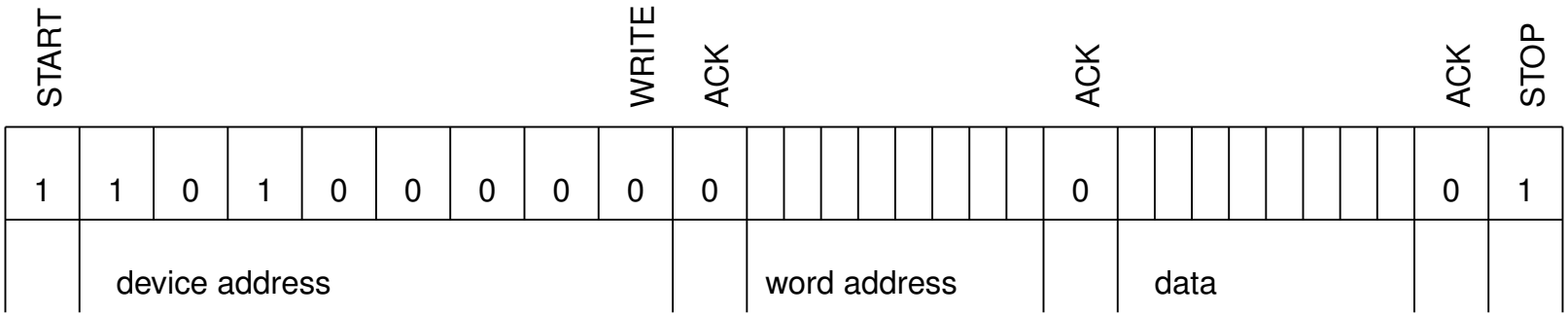
I²C-Bus

- häufig auch als Two-Wire-Interface (*TWI*) bezeichnet
- zwei Signalleitungen für Takt und Daten (`SCL`, `SDA`)
- Datenübertragung synchron (Takt: 100 kHz bis 3.4 MHz)
- Master-Slave-Prinzip
- Pullup-Widerstände sind auf dem Raspberry Pi-Board integriert



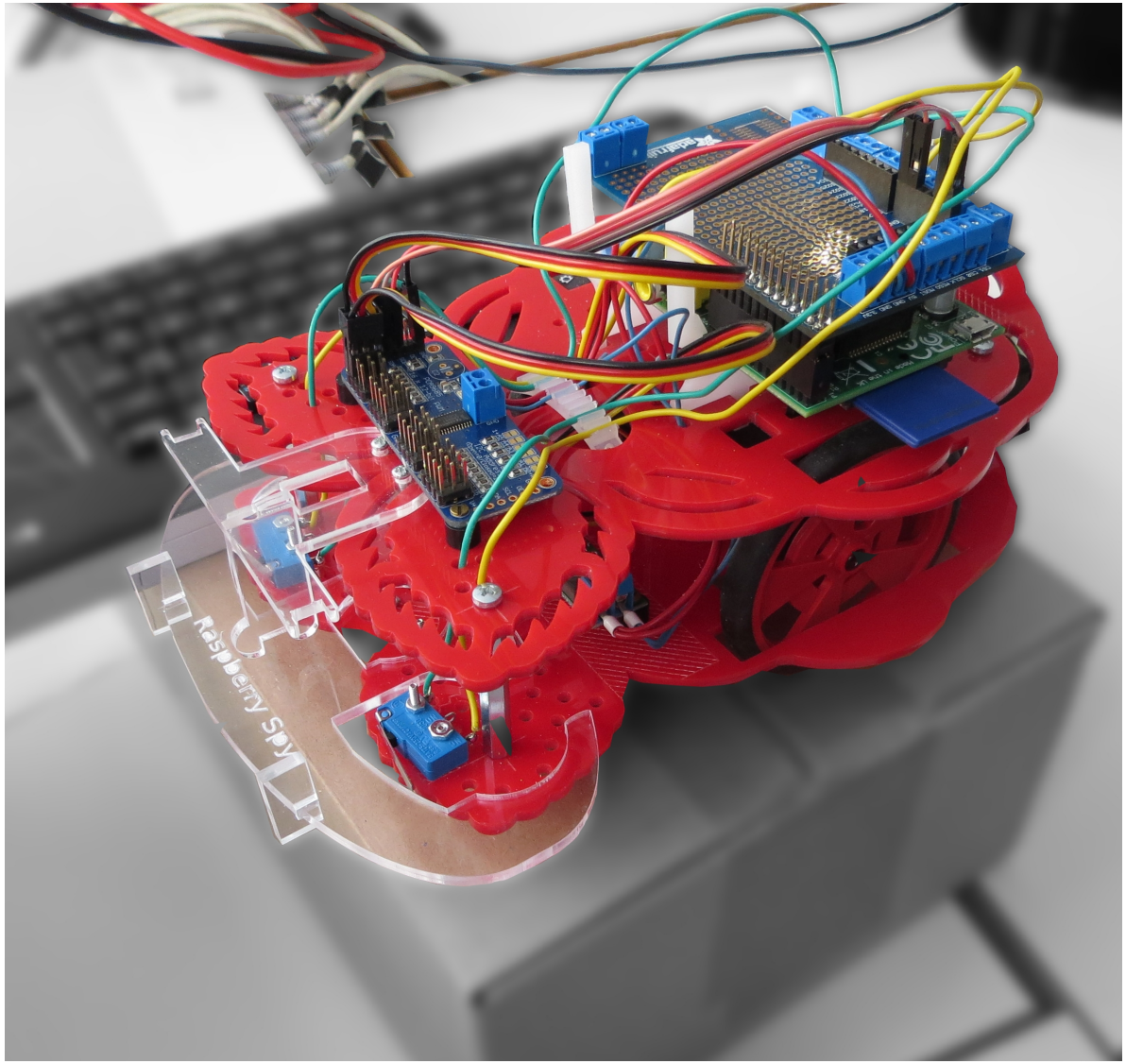
I²C-Bus

- Adressierung der Slaves erfolgt meist mit 8bit Adressen (LSB entscheidet über Lese- / Schreiboperation)




```
apt-get install i2c-tools
modprobe i2c-bcm2708
modprobe i2c_dev
# scan bus 1
i2cdetect -y 1
```

Physical Computing - Roboter



Physical Computing - Roboter

Bumper

- Tutorials[11]
- Anschluss an GPIO - melden Zustand
- RPi.GPIO-Library

Python-Code für Taster an GPIO 22:

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(22,GPIO.IN,pull_up_down=GPIO.PUD_UP)

input = GPIO.input(22)
```

[11] http://www.cl.cam.ac.uk/projects/raspberrypi/

Physical Computing - Roboter

Servos

- Einstieg: „Adafruit Learning System“[12]
- Direkt mit Pulsweitenmodulaton (PWM) über Kernel-Modul
- oder Adafruit 16 Channel Servo Driver[13]
  - theoretisch bis 992 Servos mit 62 Boards . . .
  - 3.3 V (vom Raspberry) für Prozessor, 5 V für Servos
  - SDA und SDL zur Steuerung
  - Software von Adafruit
  - Auflösung: 12 bit

[12] http://learn.adafruit.com/category/learn-raspberry-pi

[13] http://learn.adafruit.com/adafruit-16-channel-servo-driver-with-raspberry-pi/

Und weiter?

- Weitere Anschlüsse: P5 - I^2C, P6 - Reset
- LCD Display Connector (DSI): Module in Vorbereitung, bislang nur mit Basteln
- Kamera „Pi-Cam“: In Kürze, ca. 20 EUR.
- Übertakten? Bis 50% mehr Leistung möglich!
  (geschieht nur bei Bedarf, nicht bei Überhitzung)

Ziel erreicht?

- Lehr- und Ausbildungsrechner?
  Ja, aber bislang kaum für Schüler
- Vision: Raspis als Klassensatz, pro Schüler eine SD-Card
  - Motivation - Programmieren zu lernen
  - Zusammenhänge besser verstehen
  - Anregung für Experimente
- Weite Verbreitung sichert gute Chancen auf Support

Viele Ideen, die am Geldmangel oder fehlender Hardware scheiterten, werden umsetzbar.

VIELEN DANK FÜR
IHRE AUFMERKSAMKEIT!