LEGO Mindstorms RCX reaktivieren: Entwicklungen im Webbrowser

maehw

20. Augsburger Linux-Infotag

20.04.2024

Gliederung

Kurze Entwicklungsgeschichte von LEGO Mindstorms

Kooperation von LEGO mit dem MIT

- Seit 1960er: Forschenden-Gruppe am Massachusetts Institute of Technology (MIT) entwickelt Robotik-Bausätze für Kinder
- unter anderen: Dr. Jean Piaget, Dr. Seymour Papert, Dr. Mitchel Resnick
- u.a. Papert: Programmiersprache *Logo* für Kinder, später auch „Turtle Graphics“
- Papert: Buchautor von *Mindstorms: Children, Computers, and Powerful Ideas* (1980)
- ab ~1985: Kooperation des MIT mit LEGO („idea-sharing relationship“)

LEGO + Logo (1/2)

- Ende 1980er: LEGO Technic + LOGO $\rightarrow$ LEGO/Logo
- Logo-Programme steuern LEGO-„Maschinen“ vom Computer
- LEGO TC Logo (Dacta) plus Interface-Karte für Computer (Adapter für Apple II, BBC Micro, C64 und IBM PC)

Abbildung: LEGO TC Logo - u.a. Auto mit Berührungssensor *(TLG)*

LEGO + Logo (2/2)

- u.a. Sensoren auslesen, Aktoren ansteuern, Töne abspielen
- Beispielprogramm:

„Turtle in a Box“

```
1  to wander
2  gofd                  // tell turtle to go forward
3  listento 6
4  waituntil [sensor?] // wait until turtle sees a "wall"
5  offturtle             // turn off the turtle
6  tone 400 10           // sound a beep (pitch, duration)
7  tlt 10                // tell the turtle to turn
8  wander
9  end
```

Control Center und Control Lab

(a) TECHNIC Control Center (*TLG*)

- 1990
- für Zuhause, ohne PC
- 3 Motor-Anschlüsse
- batteriebetrieben

(b) LEGO Dacta Control Lab (*TLG*)

- 1993
- für die Schule
- wieder mit PC-Anbindung

Der „Programmable Brick“ vom MIT

- alles bis dahin „angekettet“ (kabelgebunden)
- „**P**rogrammable **Bricks**“: Computer in einem LEGO-Baustein
- damit jetzt mobiler Betrieb!
- Textbasierte und grafische Programmierung (Logo, LogoBlocks)
- grauer und roter „PBrick“ sind Vorläufer von LEGO Mindstorms

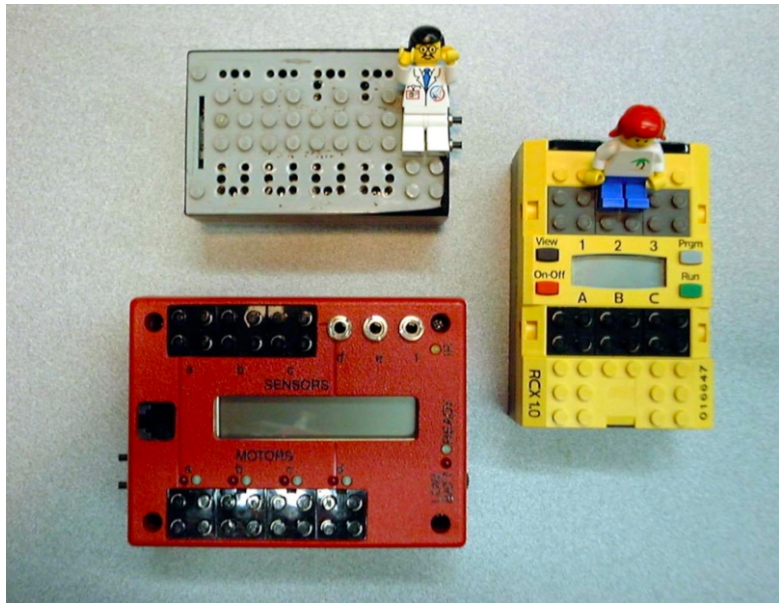


Abbildung: „MIT Programmable Bricks“, gelb: RCX *(MIT)*

MINDSTORMS-Generationen

(a) **R**obotics **C**ommand **E**xplorer / Robotics Control System (RCX) (*TLG*)

- 1998
- Teil des Sets: Robotics Invention System (RIS)
- Zielgruppe: 10-14 j. Jungs

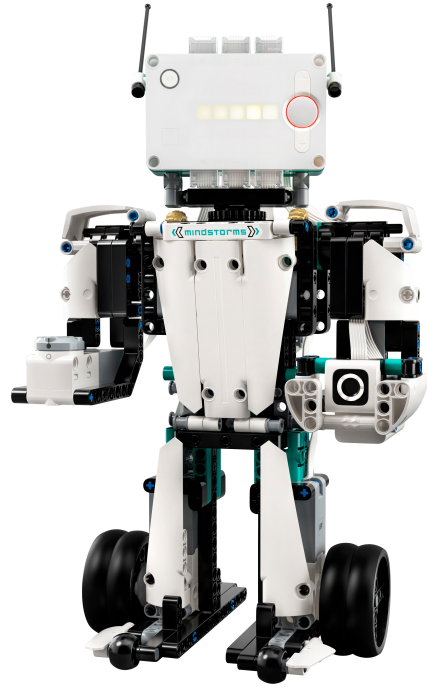
(b) NXT (*TLG*)

- 2006
- USB 2.0
- Bluetooth

MINDSTORMS-Generationen

(a) EV3 (*TLG*)

- 2013
- microSD-Karte
- USB Host(!) Port
- hier läuft ein Linux! 🐧



(b) 51515 Robot Inventor (*TLK*)

- 2020
- scheinbar letzte Mindstorms-Generation

Funktionsweise des LEGO MINDSTORMS RCX

Anatomie des RCX: von außen (1/2)

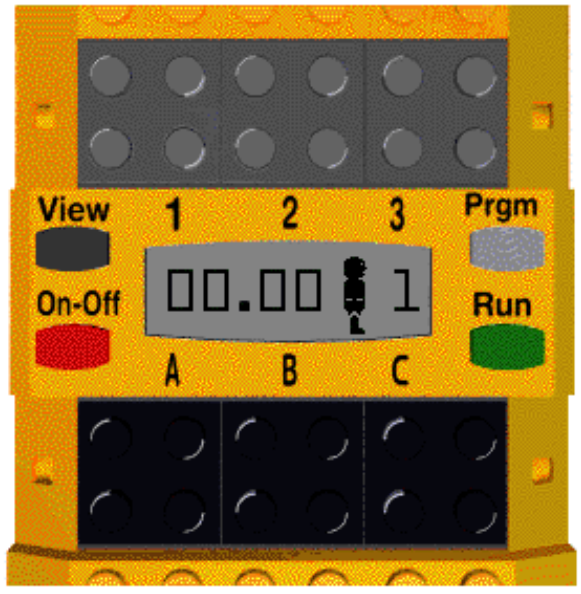


Abbildung: Darstellung RCX PBrick (*TLG*)

- LC-Display
- Vier Buttons: On-Off View Prgm Run
- 3 Eingangs-Ports für Sensoren *(Grey/Red PBrick: 8/6)*
- 3 Ausgangs-Ports für Aktoren *(Grey/Red PBrick: 4)*

Anatomie des RCX: von außen (2/2)

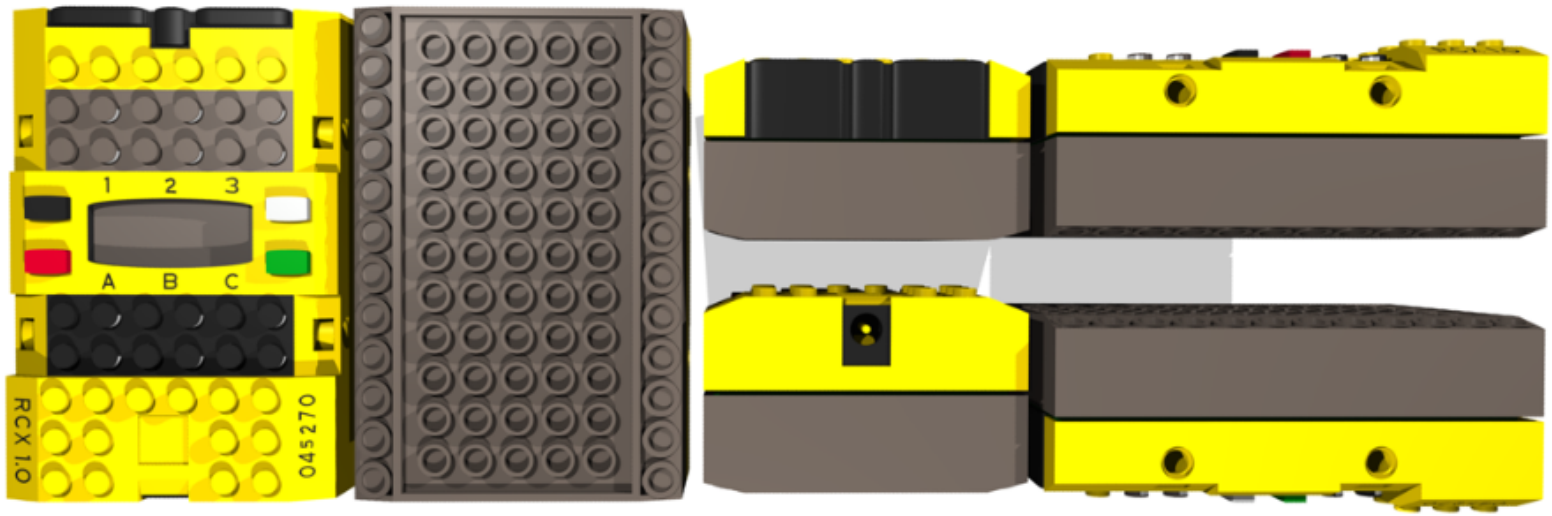


Abbildung: Ansichten RCX 1.0 (*J. Holbrook*)

- Infrarot-Schnittstelle (serielle Kommunikation)
- Drei RCX-Generationen: 1.0 / 1.5 / 2.0

Anatomie des RCX: von innen

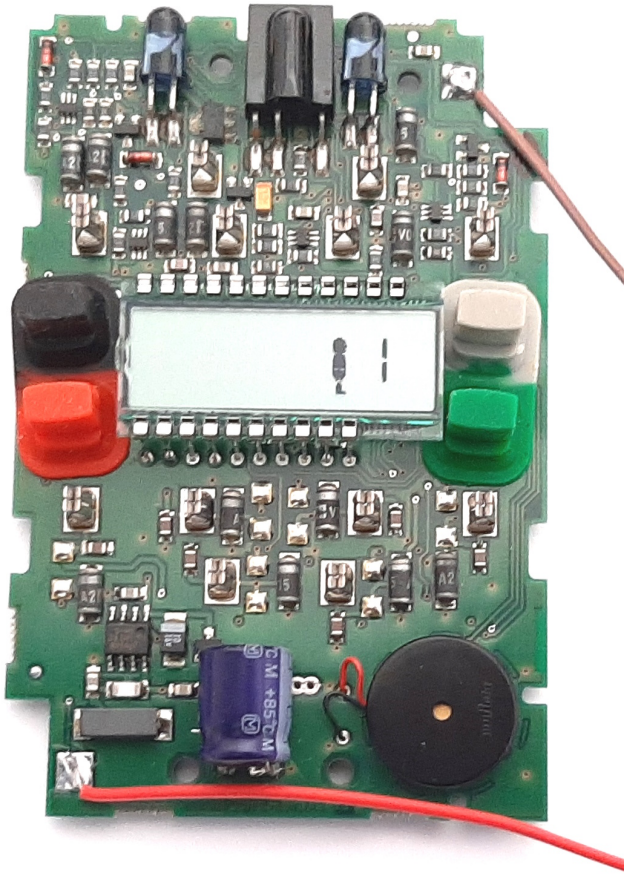


Abbildung: RCX-Leiterplatte
(*maehw*)

- (teilw.) batteriebetrieben (6x 1,5V)
- stabilisierte 5V-Spannungsversorgung
- Lautsprecher!
- Hitachi (Renesas) H8/3292 Mikrocontroller
  - H8/300H RISC Core
  - Taktung: 16 MHz
  - Instruction Length: 2-4 Bytes
  - 16- und 8-Bit-Register
  - 512 Byte(!) int. SRAM, 32 kByte ext. RAM
  - 16 kByte ROM
  - 10-bit Analog-Digital-Wandler
  - Serielle Schnittstelle

Sensoren

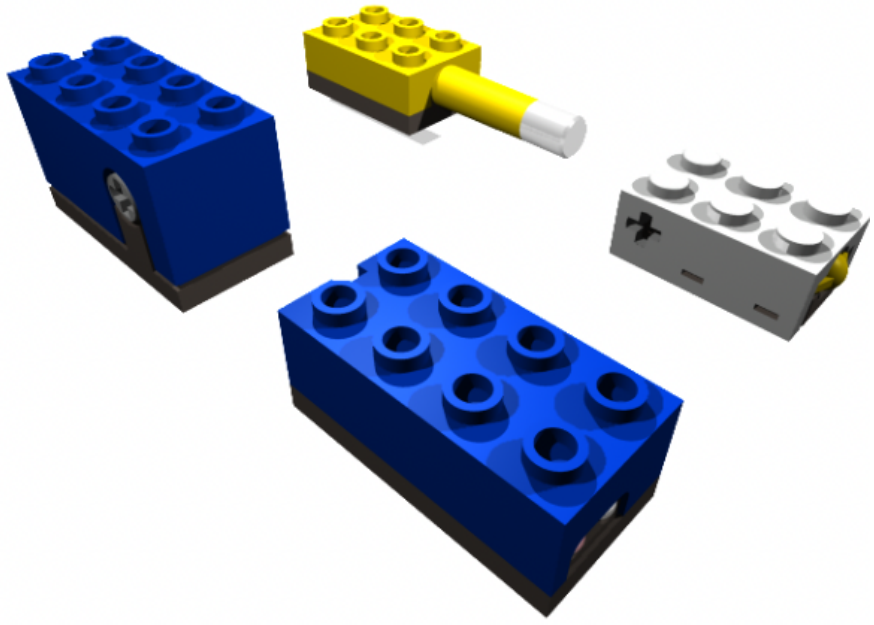


Abbildung: Sensoren im UZS: Rotation, Temperatur, Berührung, Licht
(*John Holbrook*)

- zwei Typen: passive and aktive (mit Versorgung)
- Sensor-Typ muss in Software konfiguriert werden
- auch Fremdkomponenten und Eigenbauten

Aktoren

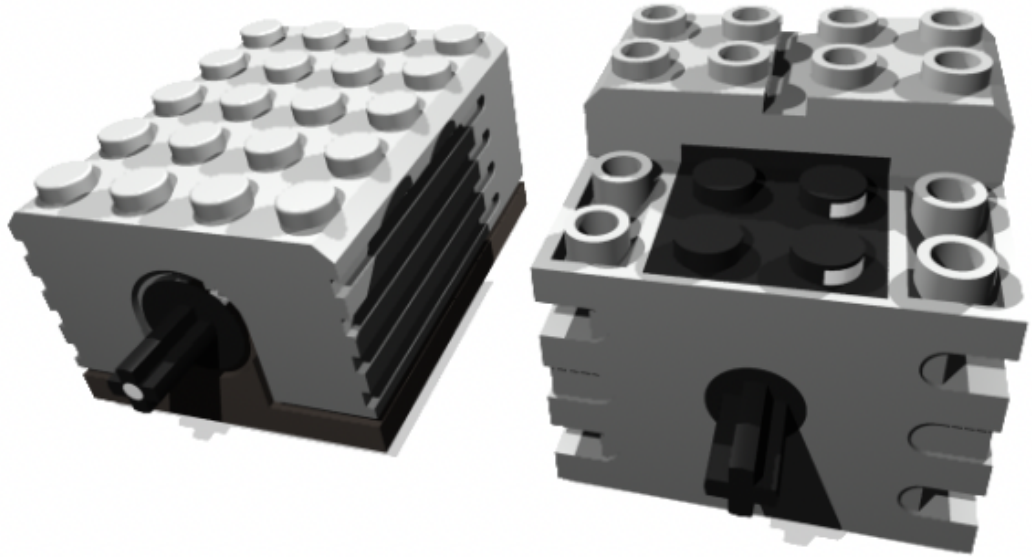


Abbildung: „rechteckiger“ und „quadratischer“ Getriebemotor
(*John Holbrook*)

- Motoren
  - Vier Modi: Vorwärts / Rückwärts / Stopp / Leerlauf
  - Acht Leistungsstufen (über PWM)
- (Lämpchen & Sirenen)

Kommunizieren mit dem RCX

- LEGO Mindstorms RCX lässt sich über Infrarot **programmieren** oder/und **direkt interaktiv fernsteuern**
- ... dazu braucht der Computer aber auch eine Infrarot-Schnittstelle:

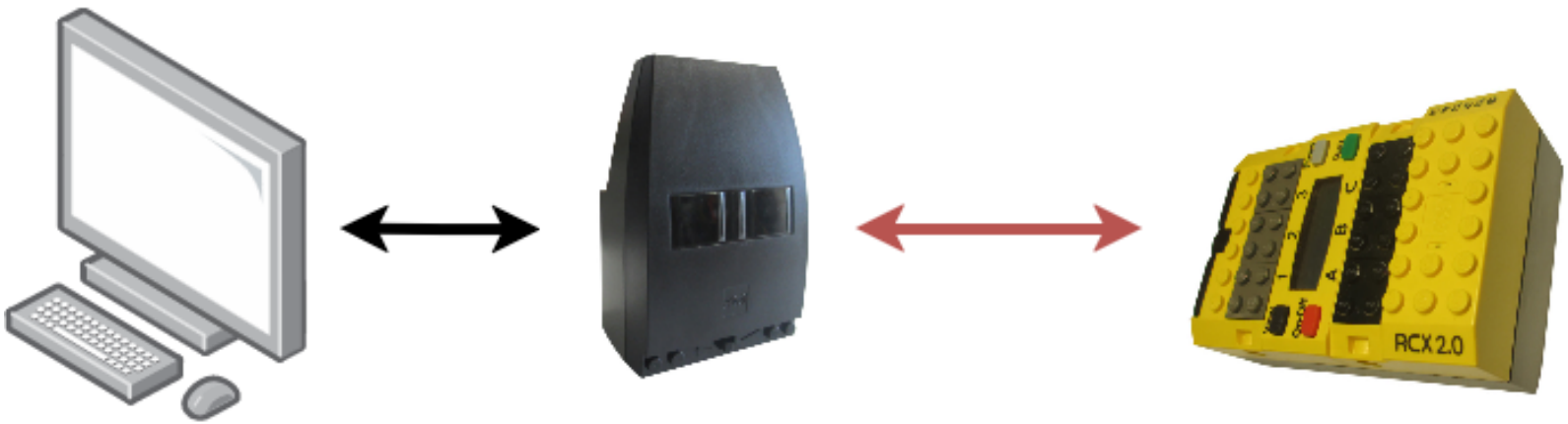


Abbildung: Computer „spricht“ mit RCX über IR Tower

- Kommunikation ist bidirektional und Halbduplex
- RCX kann sogar mit anderen RCX über Infrarot Daten austauschen

Serielle Schnittstelle

- Daten werden seriell übertragen
- also Bit für Bit hintereinander
- es gibt einen Ruhezustand (engl. „idle“)
- Beispiel: Buchstabe „S“
  - Encodierung als 7-bit ASCII „S“ (0x52) = 0b101'0011
  - Erweiterung auf 8-bit: 0b0101'0011 (führende Null)
  - gesendet in umgekehrter Reihenfolge „LSB first“
  - ggf. Paritäts-Bit für einefache Fehlererkennung (gerade oder ungerade Anzahl an Bits, die 1 sind)

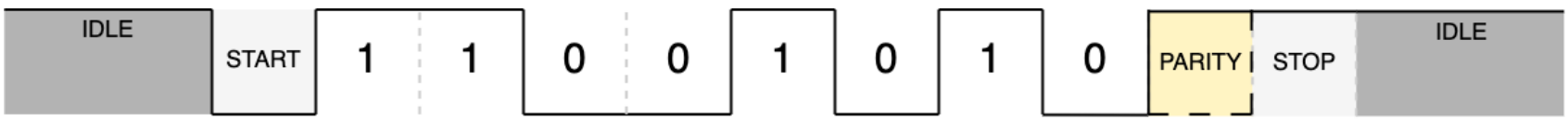


Abbildung: Serielle Schnittstelle

Serielle Schnittstelle über Infrarot

- Umwandlung elektrische $\rightarrow$ optische Signale ($\lambda \approx 950$ nm)
- Übertragungsrate: 2400 Baud ($\approx$ Bit/Sekunde)
- Konfiguration: 8O1: 8 Datenbits, „Odd Parity“ Bit, 1 Stoppbit
- jede 0 wird Infrarot-Licht ausgesendet, jede 1 nicht
- Licht wird „gepulst“ (38 kHz, OOK)

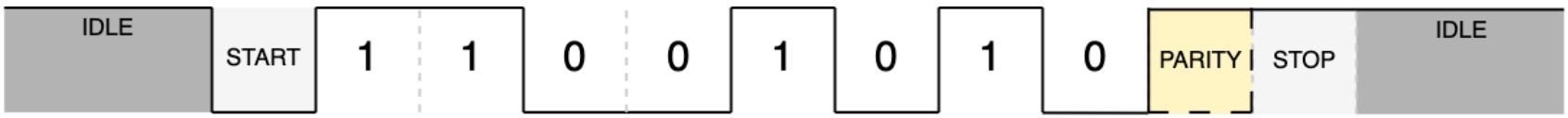


Abbildung: Serielle Schnittstelle

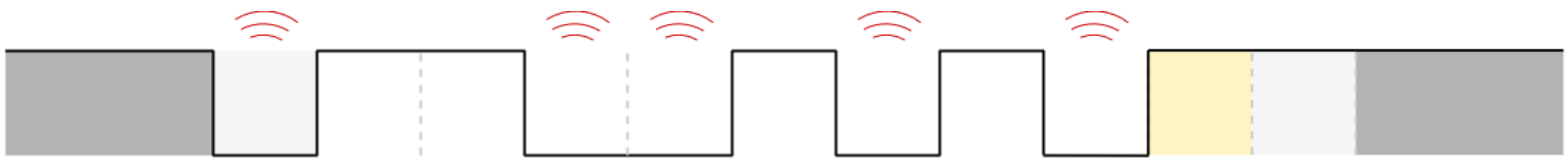


Abbildung: Infrarot-Aussendung

LEGO IR Tower

(a) LEGO IR Serial Tower (*Brickipedia*)

- Anschluss: DE-9 / serielle Schnittstelle
- Versorgung: 9V DC (Blockbatterie)
- „short/long distance“
- in früheren Sets

(b) LEGO IR USB Tower (*Brickipedia*)

- Anschluss: USB Typ A
- Versorgung: über USB 5V Bus
- USB 1.1, VID 0x0694, PID 0x0001
- Windows-Treiber: 32-bit / ⚠ 64-bit
- Linux-Treiber!

DIY IR Tower (1/2)

- „DIY“ = Do It Yourself → Eigenbau
- USB/Seriell-Wandler + µC + IR-Diode + IR-Receiver + OSS

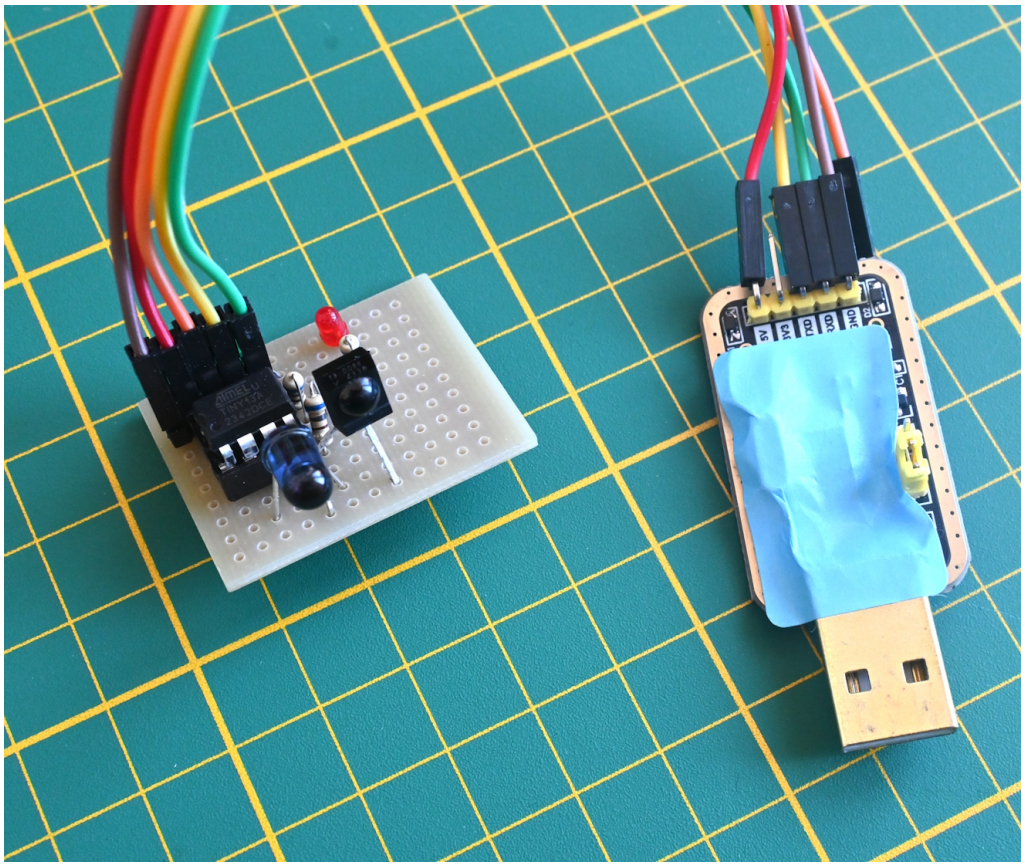


Abbildung: DIY IR (USB) Tower

DIY IR Tower (2/2)

- kompakt
- wie LEGO IR Serial Tower - nur direkt mit USB-Anschluss
- Kosten: ca. 10 €
- Mikrocontroller: Microchip ATtiny13 (ehem. Atmel)
- Power- und IR-Aktivitäts-LEDs optional
- Anschluss am Computer:
  - je nach Wandler, typ. USB Typ A
  - Versorgung: 3,3V/5V über USB
  - ggf. Treiber für USB/Seriell-Wandler
- Direktanschluss (GPIOs) an anderer Elektronik, z.B.
  - Arduino, Raspberry Pi, Pi Pico
  - Bluetooth/Seriell-Adapter
  - ...
- Firmware: Open Source (GPLv3),
  github.com/maehw/DiyIrTower

Embedded Software

- RCX-ROM-Code
  - „Hardware-Routinen“ / „Basic Input/Output System“ (BIOS)
  - Low-Level Basisfunktionen für Buttons, Motoren, Batterie-Überwachung, LC-Display, Sound, etc.
  - ermöglicht das Nachladen von Firmware im „Boot-Modus“
- RCX-Firmware
  - „Betriebssystem“ (engl. „Operating System“)
  - Hauptaufgabe: Ausführung der Anwenderprogramme ...
  - ... in einer Multi-Tasking-Umgebung
  - Ansteuerung der Hardware gemeinsam mit dem ROM-Code
- (Anwender-)Programme
  - ... von Anwender:in programmiert </>
  - haben keinen *direkten* Zugriff auf Buttons, Display oder Lautsprecher
  - schauen wir uns noch näher an!

Firmware-Download

- Firmware von LEGO ursprünglich auf CD-ROM, aber auch online: z.B. [pbrick.info] RCX Firmware
- wird ins RAM geladen (volatiler Speicher!)
- muss bei Batteriewechsel über IR neu ins RAM geladen werden
- Dateinamen mit Endung `.lgo`
- Motorola-S-Record-Format: „ASCII-basiertes Datenformat zur Kodierung von Binärdateien“
- RCX möchte noch die Prüfsumme über die ersten 19 kByte: `sum(data[0:19455] % 65536);`

`firm0332.lgo Snippet`

```
1 S00F00006669726D303333322E6C676F0A
2 S11380005E0082E4550254706DF06DF16DF26DF313
3 S11380106DF46DF56DF61B877901F1007903D700D6
4 ...
```

Original-Entwicklungsumgebung und -Programmiersprache

RCX Code

- grafische, blockbasierte Programmierumgebung
- angelehnt am LogoBlocks (MIT)
- enthalten auf CD im MINDSTORMS RIS Set
- läuft unter Microsoft Windows (98..XP) 
- gibt es in den Versionen 1.0, 1.5 und 2.0
- einfache Bedienung
- enthält auch viele (Zwangs-)Tutorials
- zahlreiche Einschränkungen beim Programmieren:
  - nur 1 Variable
  - nur 1 Timer
  - kein Zugriff auf das Datalog
- Programme als menschenlesbar Skripte abgespeichert

RCX Code: Beispiel

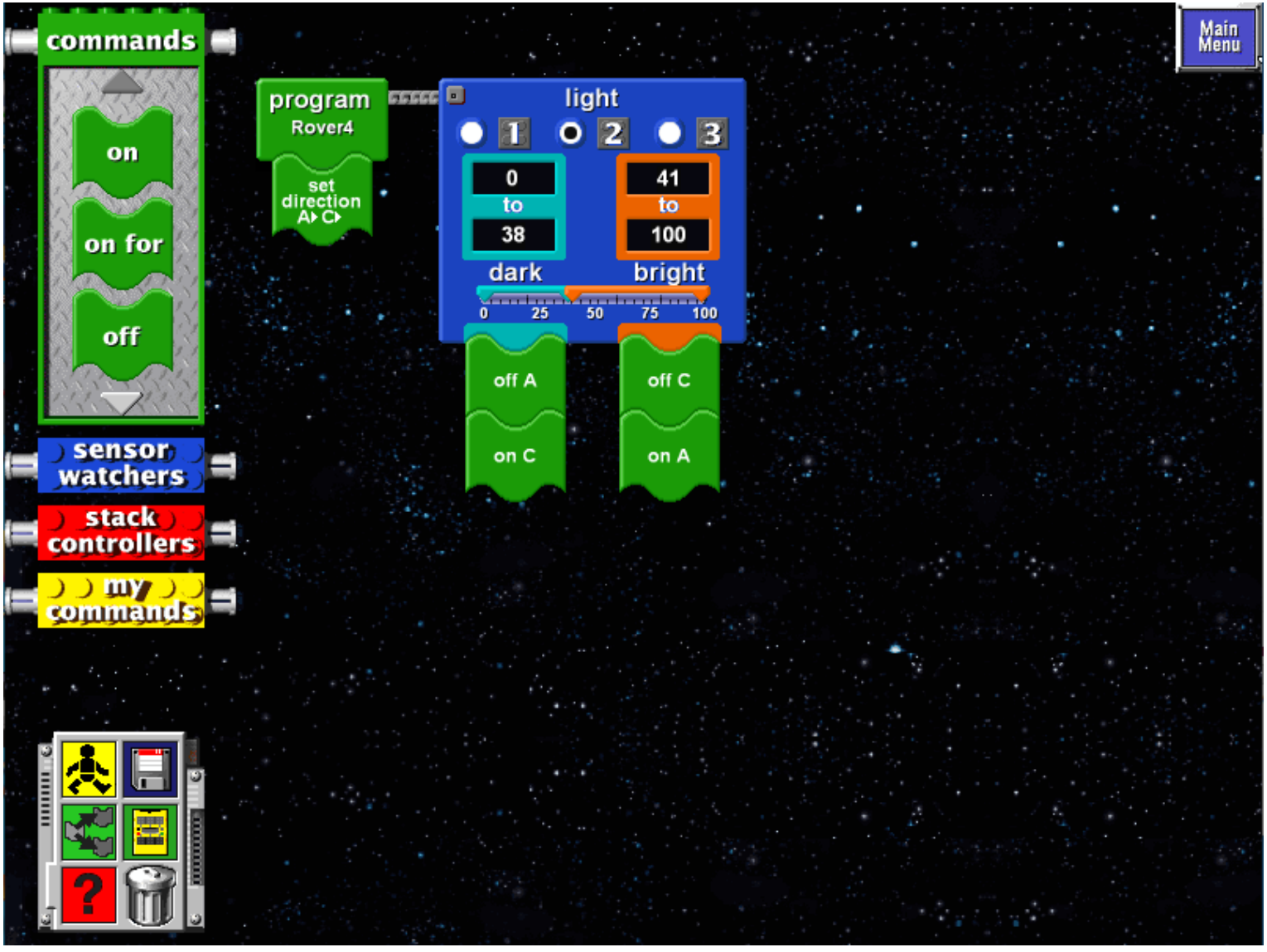


Abbildung: Linienfolger-Code *(TLG/maehw)*

ROBOLAB

- von LEGO Dacta (jetzt LEGO educational)
- für Schulen gedacht
- muss(te) kostenpflichtig erworben werden (€€)
- basiert auf National Instruments LabVIEW
- initial von der Tufts University (Boston, USA) entwickelt
- ROBOLAB (Standalone) $\neq$ ROBOLAB for LabVIEW (Add-on)
- Zugriff auf das Datalog
- zwei „Programmierebenen“ mit unterschiedl. Oberflächen
  - ROBOLAB Pilot: einfach, nutzt nicht alle Features des RCX
  - ROBOLAB Inventor: mehr Flexibilität und weniger Grenzen
- Versionen für Windows und MacOS

ROBOLAB Pilot: Beispiel

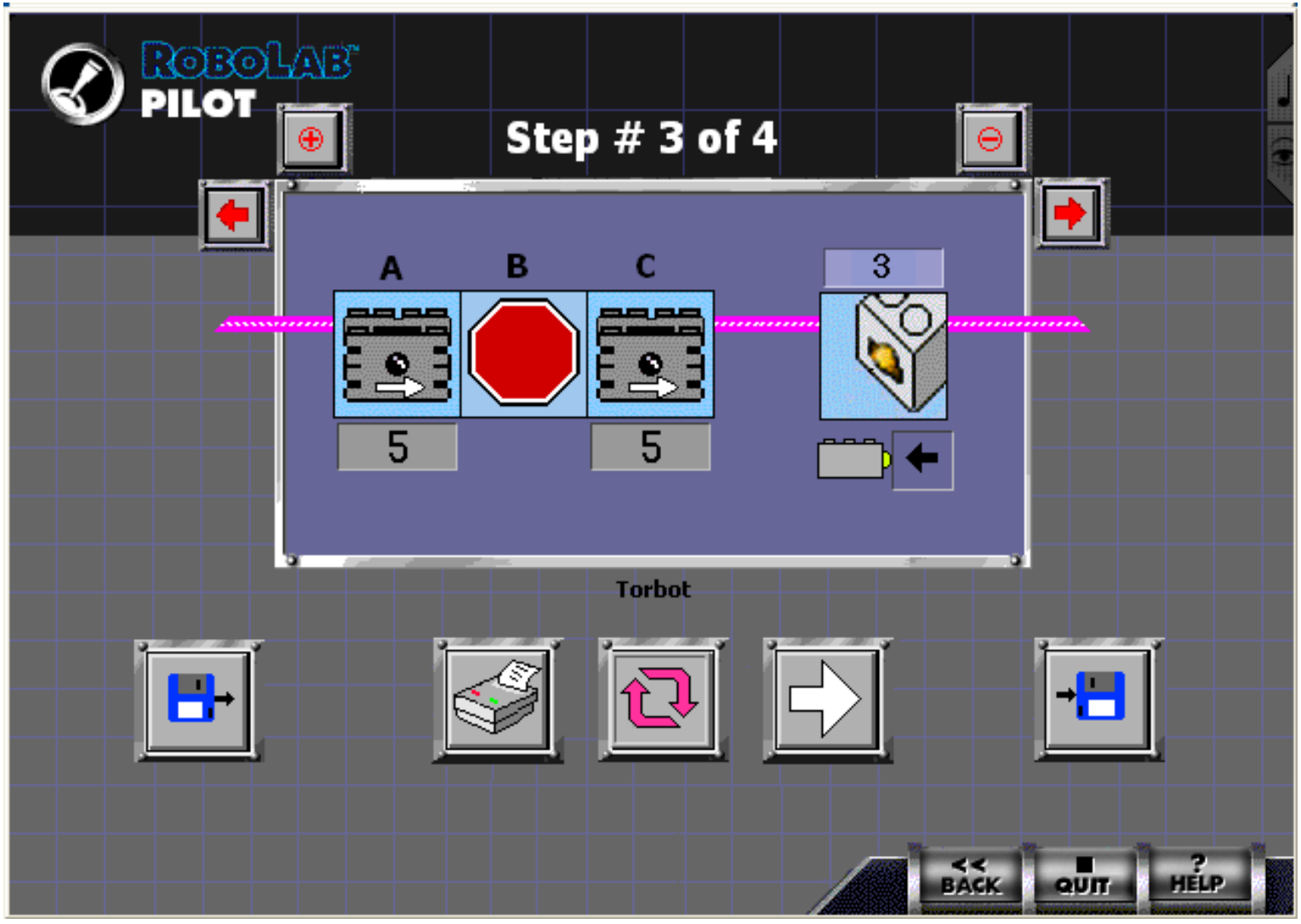


Abbildung: Beispiel-Code ROBOLAB Pilot *(TLG/maehw)*

ROBOLAB Inventor: Beispiel

Abbildung: Beispiel-Code ROBOLAB Inventor *(TLG/maehw)*

Alternative Programmiersprachen und Entwicklungsumgebungen

Alternative Programmiersprachen und Entwicklungsumgebungen - „3rd party“

- LEGO Java Operating System (leJOS) /'le.xos/
  - „originally forked out of the TinyVM project“ - **Java** ⚠Virtual Machine
  - Standard-Firmware muss ersetzt werden ⚠
- RobotC
  - **Windows** 7, 8/8.1, 10 ⚠
  - Version 4.56 unterstützt NXT und EV3
  - „Free Legacy Version 2.0.3 for RCX only“
- pbForth
  - **Forth**-Compiler(?) ⚠
  - Standard-Firmware muss ersetzt werden ⚠
  - Kommunikation mit Interpreter?
- BrickOS
- NQC

Brick Operating System (BrickOS, ~~legOS~~)

- Programmierung in **C**, **C++** (und Assembler)
- benötigt Cross-Compiler für die Hitachi H8/300-Prozessorserie (`gcc` + `binutils`)
- Standard-Firmware muss ersetzt werden ⚠
- „Vielzahl von Kernelaufrufen (...) dyn. Speicherverwaltung, Multithreading und Synchronisation“
- „Treiber für alle (...) Geräte des RCX“
- bspw. schreibt `cputs(''LIT24'');` auf das LC-Display
- Bibliotheken für Fließkommazahlen und Zufallszahlen
- sehr mächtig!
- mindestens lauffähig unter  und 

NQC: Not Quite C

- textbasierte Programmiersprache + Compiler
- entwickelt von Dave Baum
- ähnliche Syntax wie die Programmiersprache C
- kann mit Standard-Firmware von LEGO genutzt werden 👍
- Open-Source-Lizenz: Mozilla Public License (MPL 2.0)
- Anwenderprogramme über Kommandozeile übersetzen und aufspielen (🪟 🍎 🐧!)
- alternativ über IDEs wie BricxCC (🪟) oder MacNQC (🍎)
- damals sehr beliebt
  (viele Online-Ressourcen und auch Bücher!)
- „aktueller“ Release: 3.1 r6 (Juni 2007)

„Hello World“ in NQC

Sound abspielen, wenn Berührungssensor gedrückt wird:

helloworld.nqc

```
task main() {
  SetSensor(SENSOR_1, SENSOR_TOUCH);

  while (true) {
    until(SENSOR_1 == 0); // wait until button is pressed

    PlaySound(SOUND_DOWN);

    until(SENSOR_1 == 1); // wait until button is released
  }
}
```

Kompilieren und Herunterladen

```
nqc -d test.nqc
```

Einfacher Linienfolger in NQC

linebot.nqc

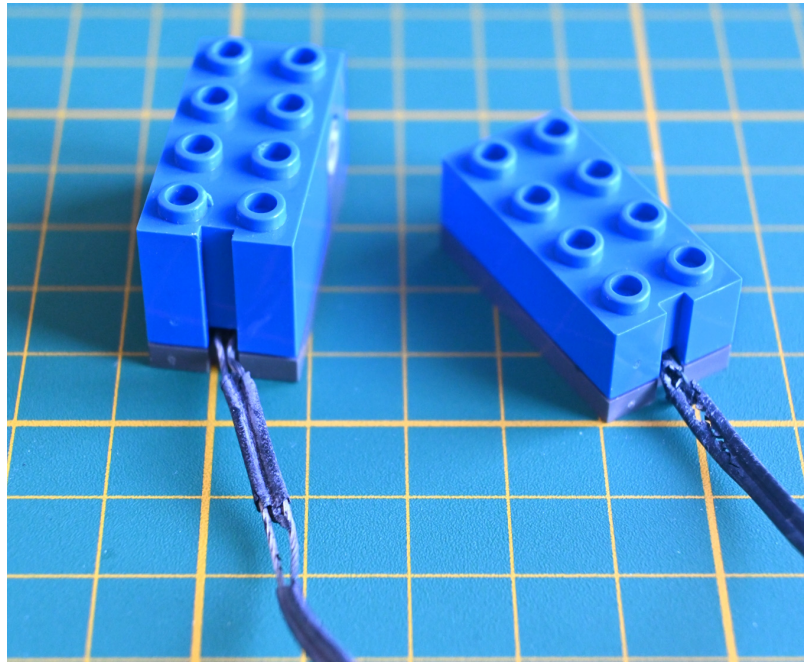
```
task main()
{
        SetSensor(SENSOR_2, SENSOR_LIGHT);
        On(OUT_A+OUT_C);

        while(true)
        {
                if (SENSOR_2 <= LEFT_THRESHOLD)
                {
                        Off(OUT_A);
                        On(OUT_C);
                }
                else if (SENSOR_2 >= RIGHT_THRESHOLD)
                {
                        Off(OUT_C);
                        On(OUT_A);
                }
                else
                {
                        On(OUT_A+RIGHT);
                }
        }
}
```

Herausforderungen bei der Reaktivierung nach 25 Jahren

Herausforderungen bei der Reaktivierung nach 25 Jahren

(a) Defekte Verbindungskabel

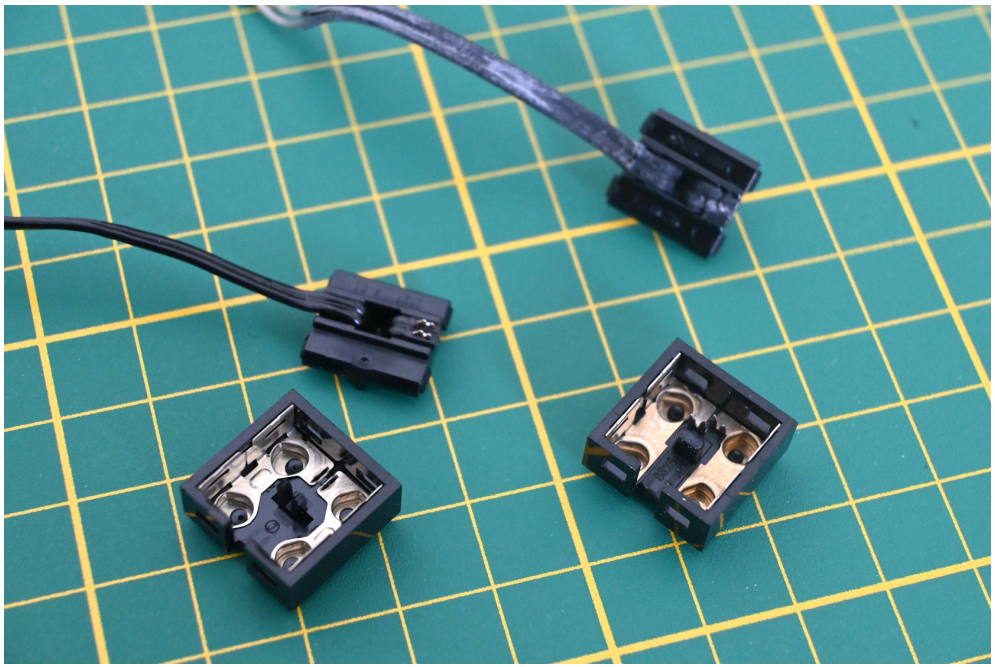
- ersetzbar (€€€)
- reparierbar (ohne Löten, €)



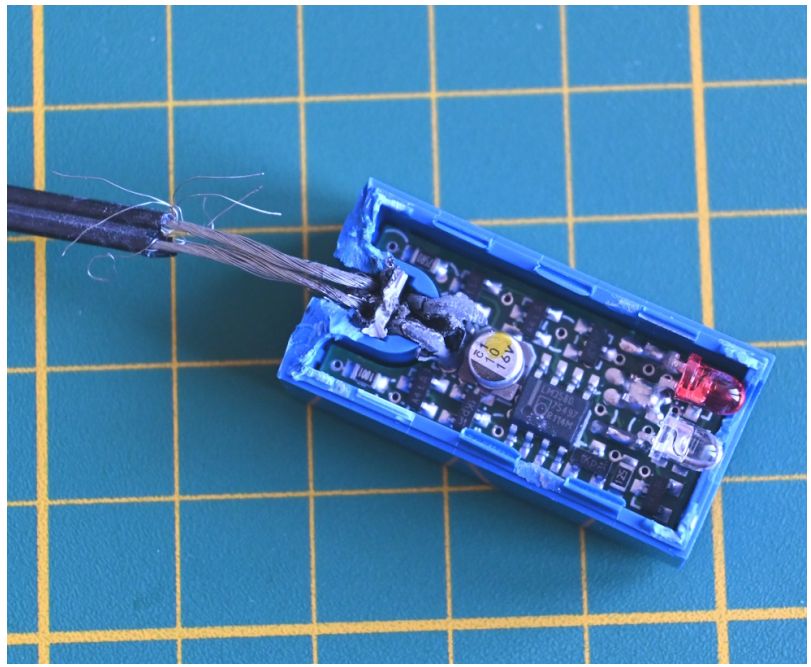
(b) Defekte Sensor-Anschlusskabel

- ersetzbar (€€)
- schwieriger reparierbar (mit Löten, €)

Herausforderungen bei der Reaktivierung nach 25 Jahren



(a) Geöffnetes Verbindungskabel



(b) Geöffneter Licht-Sensor
→ besser nicht öffnen! ⚠

Herausforderungen bei der Reaktivierung nach 25 Jahren

- Spröde, brüchige Kabel (Kurzschlussgefahr)
- Ausgelaufene Batterien
- generell: korrodierte Kontakte
- Lizenzen für Microsoft Windows ( , €, )
- CD-ROMs - optische Datenträger und Laufwerke dafür
- Anschlüssmöglickeiten „echter“ serieller Schnittstellen („COM-Ports“)
- 9V-System, Netzteil mit 9-12 V AC
- viele Webseiten nicht mehr verfügbar, tote Links

Projekt „WebPBrick“: Entwicklungen im Webbrowser

Web-IDE: WebPBrick.com

- Featues:
  - Firmware-Download auf den RCX
    (zur Reaktivierung oder für Alternativ-Firmwares)
  - Textbasierte Programmierung in NQC
  - Download der gebauten Anwenderprogramme auf den RCX

- Drei „moderne“ Technologien im Webbrowser:
  - WebAssembly (Wasm)
  - Web Serial API
  - WebUSB API

Demo der Web-IDE

webpbrick.com

WebAssembly (Wasm) - Compiler für das Web (1/3)

- **Webbrowser kann Bytecode ausführen**[1]
- WebAssembly ist Standard seit 2017
- NQC-Compiler Quellcode ist (auch) auf github, MPL 2.0
- Build mittels `make` hat auf Mac OS direkt funktioniert, sollte auch unter Linux
- Build für das Web
  - Nutung des Emscripten Compiler Frontend (`emcc`)
  - nur kleinere Anpassungen am urspr. NQC-Makefile
  - weitere kleinere Anpassungen am C-Code von NQC (z.B. unbenutzte Kommandozeilen-Optionen entfernt, IR-Kommunikation)
- Zwischenresultat: WebNQC , MPL 2.0

WASM bauen

```
1 make -f Makefile.mkdata
2 emmake make CXX=emcc
```

[1] caniuse.com/wasm

WebAssembly (Wasm) - Compiler für das Web (2/3)

- Emscripten-Output:
  - „the low level *compiled code module*“ +
  - „the ***JavaScript*** *runtime* to interact with it“ JS
- `nqc.wasm`: ca. 285 kByte WASM
- `nqc.js`: ca. 5000 Zeilen generierter JavaScript-Code JS
- `nqcWrapper.js`: ca. 150 Zeilen eigener Wrapper-Code („Shell-File“) JS
- ⚠ „*Several browsers (including Chrome, Safari, and Internet Explorer) do not support `file://` XHR requests, and can't load extra files needed by the HTML (like a `.wasm` file, or packaged file data (...).*“
- Website aufsetzen oder einen lokalen Webserver starten, z.B.:

Webserver starten

```
python -m http.server 8080
```

WebAssembly (Wasm) - Compiler für das Web (3/3)

- `stdout` und `stderr` können umgelenkt werden
- Zugriff auf Input- und Output-Dateien über ein Filesystem im Browser

JavaScript WASM-Shell

```
createWebNqc( { 'print': printFunction,
                'printErr': printErrFunction }
).then(instance => {
  nqc = instance;
});
...
async function clickConvert() {
    ...
    nqc.FS.writeFile(input_filename, txtInput.value+"\n");
    let retval = nqc.callMain(args);
    const out = nqc.FS.readFile(output_filename);
    ...
}
```

Web Serial API - Serielle Geräte im Webbrowser

- **Webbrowser kann auf serielle Schnittstellen des PC zugreifen**[2]
- viele Embedded Devices haben „noch“ serielle Schnittstellen
- auch der der LEGO IR **Serial** Tower und der **DIY IR** Tower
- Einstieg z.B. über [Code Lab] Getting Started with Web Serial
- dort Kommunikation mit einem BBC micro:bit (mit Espruino-Firmware)

JavaScript: seriellen Port öffnen

```
1  // Request a port and open a connection.
2  serialPort = await navigator.serial.requestPort();
3  // Wait for the port to open. Configure 2400 baud, 8-0-1
4  const serialParams = { baudRate: 2400, parity: "odd" };
5  await serialPort.open(serialParams);
6
7  const serialPortInfo = serialPort.getInfo();
```

[2] caniuse.com/web-serial

Web Serial API - Lesen & Schreiben

- ein geöffneter Port kann gelesen und beschrieben werden:

JavaScript: seriellen Port lesen + beschreiben

```
1  serialReader = serialPort.readable.getReader();
2  serialWriter = serialPort.writable.getWriter();
3
4  // generate message to transmit
5  txMsg = ...;
6  // ... and transmit it via serial
7  await serialWriter.write(txMsg);
8
9  // read received response message
10 result = await serialReader.read();
```

- RCX: LEGO hat Spezifikation veröffentlicht:
  „RCX 2.0 Firmware Command Overview“ (108 Seiten)
- Kommando-Pakete werden mit Response-Paketen beantwortet
- ⚠ Licht reflektiert, Licht strahlt ein, Toggle-Bit

WebUSB API - USB-Geräte im Webbrowser ansprechen

- **Webbrowser kann auf USB-Geräte des PC zugreifen**[3]
- ⚠ leider(?) bisher nicht mit Mozilla Firefox
- Einstieg in WebUSB z.B. über [MDN Web Docs] WebUSB API - Web APIs oder [Chrome for Developers] Auf USB-Geräte im Web zugreifen

[3] caniuse.com/webusb

WebUSB API - LEGO IR USB Tower (1/2)

lsusb -vvv

```
Bus 003 Device 002: ID 0694:0001 Lego Group Mindstorms Tower
Device Descriptor:
  bLength                18
  bDescriptorType         1
  bcdUSB               1.10
  bDeviceClass          255 Vendor Specific Class
  bDeviceSubClass         0
  bDeviceProtocol         0
  bMaxPacketSize0         8
  idVendor           0x0694 Lego Group
  idProduct          0x0001 Mindstorms Tower
  bcdDevice            1.00
  iManufacturer           4 LEGO Group
  iProduct               26 LEGO USB Tower
  bNumConfigurations      4
    ...
```

→ USB vendor-specific device class

WebUSB API - LEGO IR USB Tower (2/2)

- Vier Konfigurationen
- zwei Interrupt-Endpoints (Host → Device, Device → Host)
- Geräte-Konfiguration über Vendor-spezifische Requests
- oben bereits erwähnt: Linux-Treiber
- LEGO hat auch hier Schnittstellenbeschreibung veröffentlicht: „LEGO USB Tower Interface Reference“ (36 Seiten)
- weitere Inspiration: github.com/hangrydave/InfraredBrickTower
- Work in progress: Vendor Requests funktionieren, IR-Übertragung nur in Senderichtung
- Ausprobieren: webpbrick.com/communication/webusb.htm

Ausblick & Outro

Ausblicke

- Interesse am RCX?
  - Entstauben oder günstig gebraucht online kaufen.
  - Auch vorher gerne schon hier ausprobieren!
- Ideen für Weiterentwicklungen
  - Unterstützung für LEGO USB IR Tower?
  - Open Source Hardware für DIY IR Tower?
    (Schaltplan und Layout mit KiCAD?)
  - Visuelle, blockbasierte Programmierung?
    (Blockly, vgl. Scratch)
  - Interaktive Fernsteuerung über Infrarot? Klavier?
  - NQC: Auto-Completion? Syntax-Highlighting?
  - Eigene NQC-Programme mit anderen teilen und „forken“?
  - Das Projekt ist Open Source! 
  - (Code-)Beiträge sind willkommen!

Ausblicke

- Kein RCX? Neue Mindstorms-Generationen:
  - viele, viele Community-Projekte für alle Generationen
  - LEGO hat die Mindstorms-Reihe eingestellt, aber: SPIKE Prime-Reihe kompatibel (neue Firmware flashen)
  - Pybricks: Programmierung in Python + blockbasiert möglich

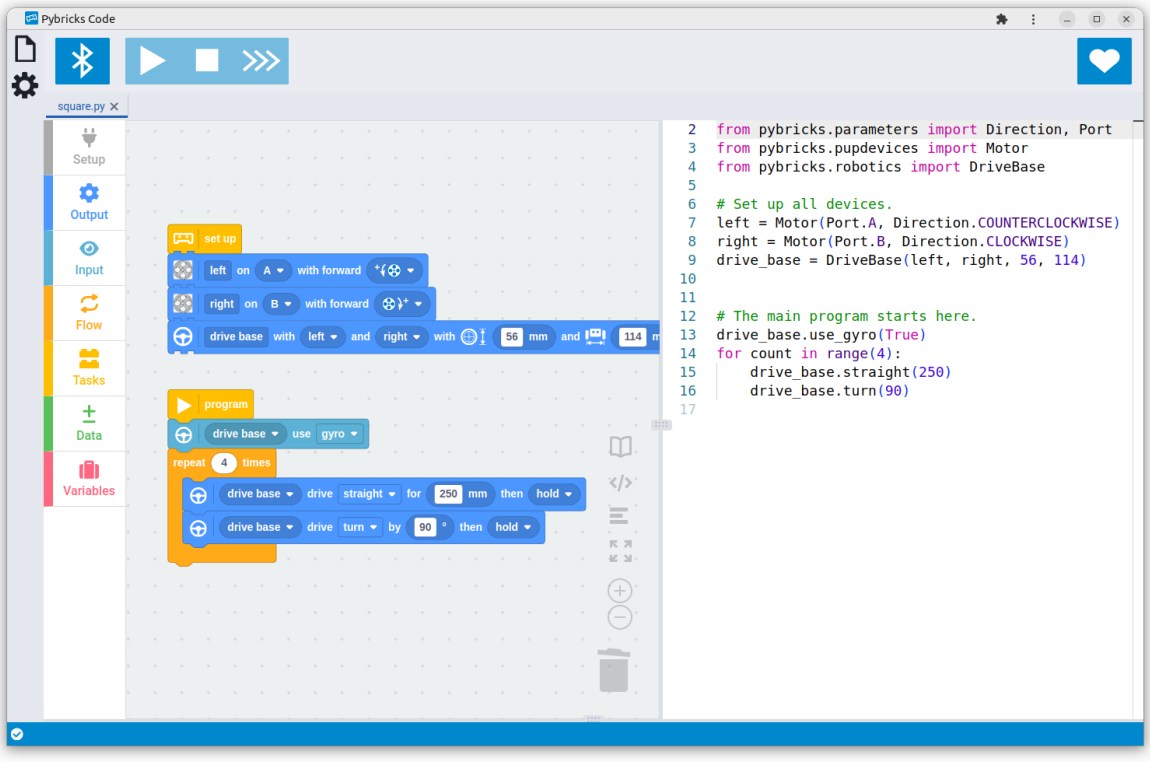


Abbildung: Pybricks - Blöcke und Code-Live-Preview *(Pybricks)*

Ressourcen

- Videos
  - [YouTube] BatterPoweredBricks - Replacing Lego 9v Wire / Repairing Mindstorms Sensors
  - [YouTube] Seymour Papert introduces LEGO TC Logo
- Bücher
  - Knudsen, J. B., & Noga, M. L. (2000). *Das inoffizielle Handbuch für LEGO-MINDSTORMS-Roboter.*
  - Baum, D. (2000). *Dave Baum's definitive guide to Lego Mindstorms.*
- Webseiten
  - [brothers-brick.com] A History of LEGO Education
  - [pbrick.info] RCX Firmware
  - Proudfoot, Kekoa: RCX Internals
  - [Code Lab] Getting Started with Web Serial
  - [MDN Web Docs] WebUSB API - Web APIs
  - [Chrome for Developers] Auf USB-Geräte im Web zugreifen

Ressourcen

- Artikel, (Reference) Manuals und Spezifikationen
  - Holbrook, J. (2017). *Using the LEGO Mindstorms RCX in 2017*
  - Baumann, C. (2023). *The 25th Anniversary of the LEGO® RCX®*
  - LEGO Technology Center (1999-2000). *LEGO USB Tower Interface Reference* Specification
  - LEGO (2000). *Specification: RCX 2.0 Firmware Command Overview*
  - MIT. (2000). *LEGO Mindstorms: The Structure of an Engineering (R)evoloution*
  - MIT. *To Mindstorms and Beyond: Evolution of a Construction Kit for Magical Machines*

Fragen?

Kontaktmöglichkeiten

- Mastodon: @maehw@chaos.social
- E-Mail: lit2024@webpbrick.com
- GitHub: github.com/maehw
- persönlich nach diesem Vortrag
- Einladung zum Ausprobierem am „Stand“