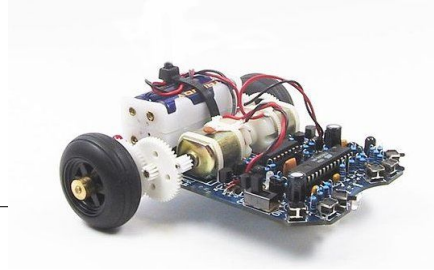
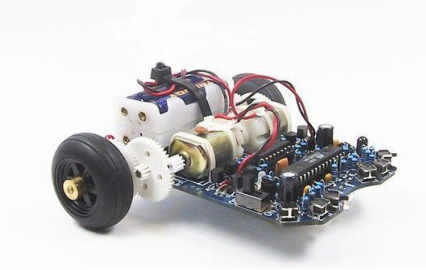
Mikrocontroller selbst programmieren

Uwe Berger

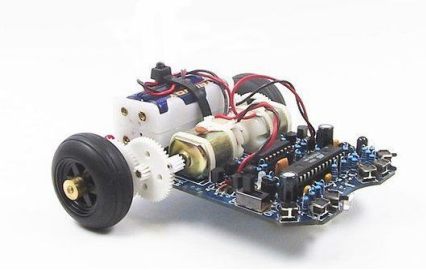


Zurück zu den Anfängen...

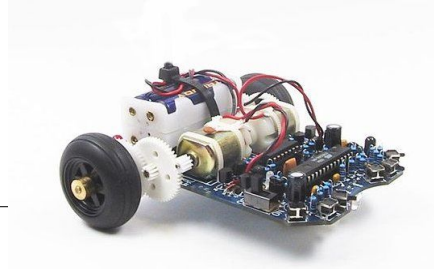


Inhalt

- Was sind Mikrocontroller
- AVR-Mikrocontroller
- Voraussetzungen für Mikrocontroller-Projekte
- Projektbeispiele

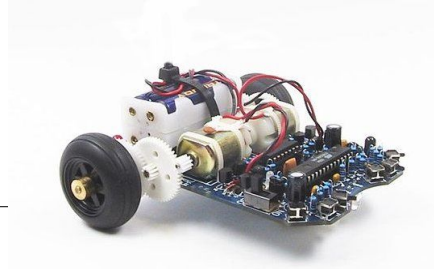


- Was sind Mikrocontroller
- AVR-Mikrocontroller
- Voraussetzungen für Mikrocontroller-Projekte
- Projektbeispiele



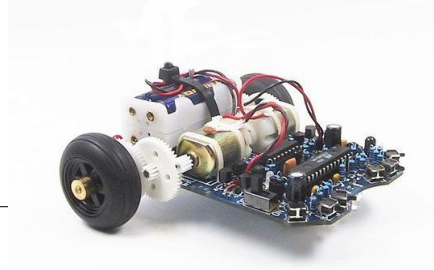
Zwei Definitionen

- Mikrocontroller:
  - Mikrorechner, bei denen viele Komponenten eines Computers auf einem Schaltkreis integriert sind ("Ein-Chip-Computer")
  - Prozessor, Speicher, Interruptcontroller, diverse Ein-/Ausgabe-Einheiten usw.
- Mikroprozessoren:
  - sämtliche Komponenten eines Prozessors mit seinem Steuerwerk auf einem Chip
  - spezielle Mechanismen zu effizienten Befehlsabarbeitung
  - Speicher, Ein-/Ausgabe-Einheiten usw. extern



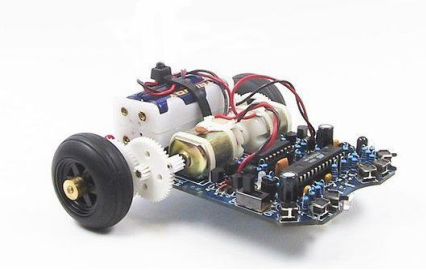
Mikrocontroller vs. Mikroprozessoren

- Mikrocontroller:
  - begrenzte Ressourcen
  - geringe Rechenleistung
  - bereits mit wenig Peripherie lauffähig
  - meist für spezielle Anwendungsgebiete
  - Messen, Steuern, Regeln
  - ideal für Hobby-Projekte
- Mikroprozessoren:
  - skalierbare Ressourcen
  - hohe Rechenleistung
  - ohne zusätzliche Peripherie nicht lauffähig
  - universelle Rechen-maschine
  - PCs, Großrechner, Supercomputer



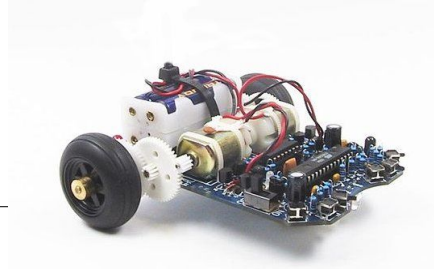
Andere Prozessorfamilien

- DSP (Digital signal processor)
  - spezialisierte Chips zur digitalen Verarbeitung von analogen Signalen
  - Filter, Effekte, Datenkompression, Signalanalyse
  - meist mit A/D- und D/A-Wandler ausgestattet
  - meist echtzeitfähig
  - Audio-/Video-Bearbeitung
  - Hersteller z.B. Analog Devices, Texas Instruments, Motorola...



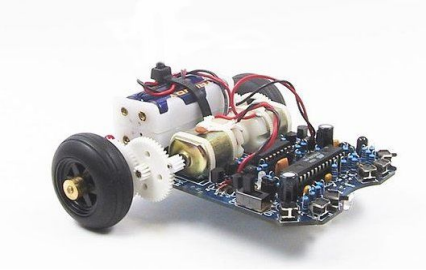
Andere Prozessorfamilien

- FPGA (Field programmable Gate Array)
  - "vor Ort modifizierbarer Logikbausstein"
  - frei, mittels einer Beschreibungssprache, konfigurierbare und kombinierbare Hardwarekomponenten
  - jederzeit rekonfigurierbar
  - CPLD (Complex Programmable Logic Device)
  - ASIC (Application specific integrated circuit)



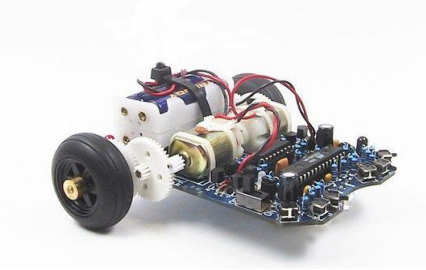
Verbreitete Mikrocontroller-Familien

- MSP430
  - spezielle MC-Reihe der Firma Texas Instruments
  - 16-Bit Prozessorkern
- ARM
  - spezielle 32-Bit RISC-Prozessorkerne der Firma ARM
  - diverse Hersteller verwenden diesen Kern für eigene Produkte
  - Vorteil: einheitlicher Befehlssatz

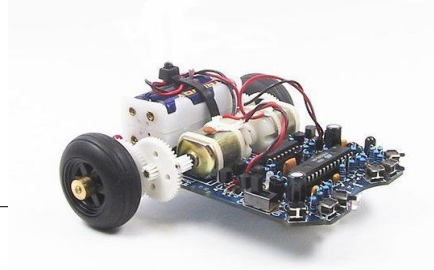


Verbreitete Mikrocontroller-Familien

- PIC
  - Hersteller: Microchip Technology Inc.
  - 8-, 16-, 32-Bit RISC-Prozessoren
  - spezielle integrierte Komponenten: LCD, USB, Ethernet etc.
  - ebenfalls bei Hobby-Elektronikern sehr beliebt
- 8051
  - Prozessorarchitektur von Intel, diverse Derivate von verschiedenen Herstellern
  - 8-Bit CISC-Prozessorkern
- AVR --> Mikrocontroller-Reihe der Firma Atmel...

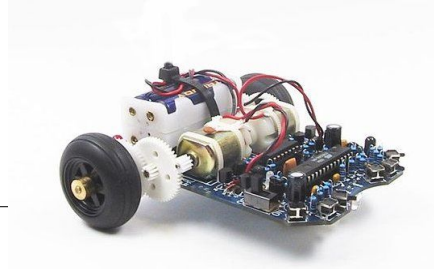


- Was sind Mikrocontroller
- AVR-Mikrocontroller
- Voraussetzungen für Mikrocontroller-Projekte
- Projektbeispiele



AVR-Mikrocontrollerfamilien

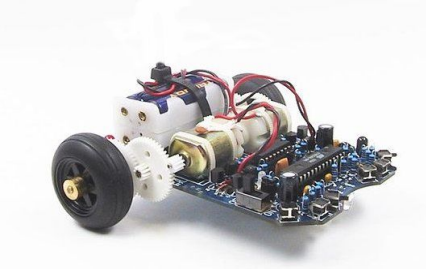
- AVR: offiziell nur ein Eigenname
- Übersicht: http://www.avr-praxis.de/content/view/34/53/
- unterscheiden sich in:
  - Prozessortakt
  - Speicher
  - Anzahl der I/O-Pins, AD-Wandler, Timer, Interrupt
  - Stromverbrauch
  - integrierte Spezialkomponenten



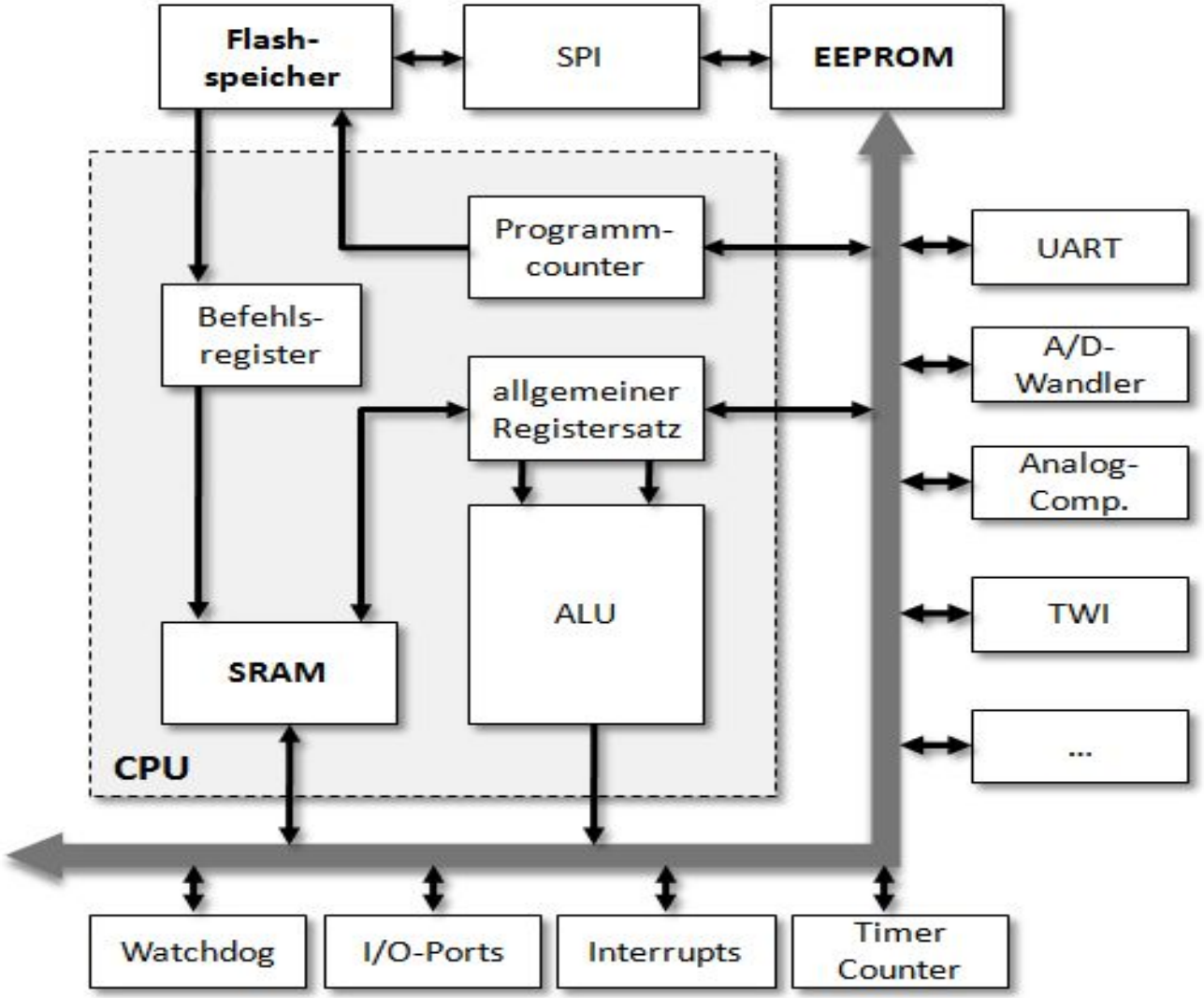
AVR-Mikrocontrollerfamilien

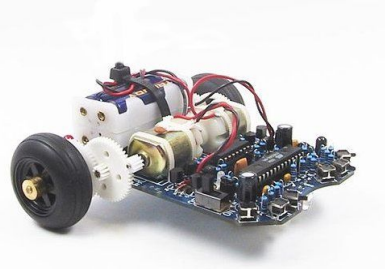
- AT90Sxxx: veraltete, "klassische" AVR-Reihe
- AT90xxx: Nachfolger der Classic-Reihe (auch USB, CAN)
- ATtiny: "kleine" AVR-Reihe; wenige I/O-Pins, kleiner Speicher
- ATmega: "große" AVR-Reihe; teilweise bis 256kB Flash, 86 I/O-Pins, 16 AD-Wandler

- AVR32 (32-Bit RISC-Prozessorkern) gehört nicht zur AVR-Reihe



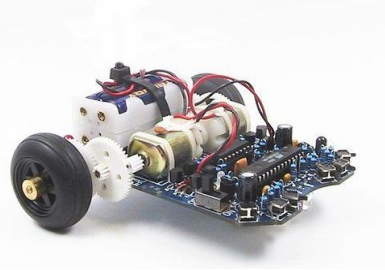
AVR-Mikrocontroller





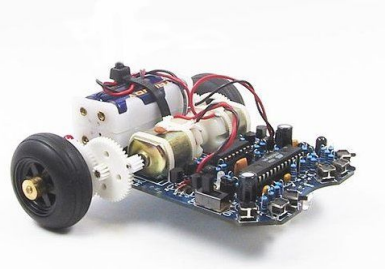
Mikrocontroller-Kompomenten (AVR)

- Prozessorkern
  - 8-Bit RISC-Prozessorkern
  - meist 1-2 Taktzyklen pro Befehl
  - Taktfrequenz bis 20MHz
- Taktgeber
  - interner Oszillator
  - externer Takteingang
  - Steuerung über Fuse-Bits



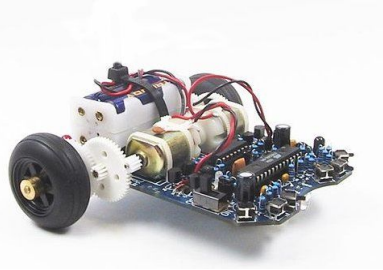
Mikrocontroller-Kompomenten (AVR)

- Speicher
  - Harvard-Architektur (getrennte Daten-/Befehlsspeicher)
  - Flash (nicht flüchtiger Programmspeicher)
  - SRAM (flüchtiger Datenspeicher, u.a. Stack, schnell)
  - EEPROM (nichtflüchtiger Datenspeicher, langsam)
- Interruptsteuerung
  - Unterbrechung des Hauptprogramms durch spezielle Serviceroutinen (ISR)
  - Zustand des Hauptprogramms wird vollständig gesichert
  - diverse Interruptquellen



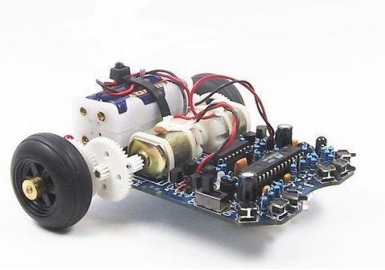
Mikrocontroller-Komponenten (AVR)

- I/O-Ports
  - Pins die als Ein- oder Ausgänge konfigurierbar sind
  - Zustand via spezieller Register ein-/setzbar
  - Interruptquelle
- UART (Universal Asynchronous Receiver Transmitter)
  - asynchrone serielle Schnittstelle
  - Kommunikation mit der Außenwelt
  - bis 115kBit/s
  - Interruptquelle



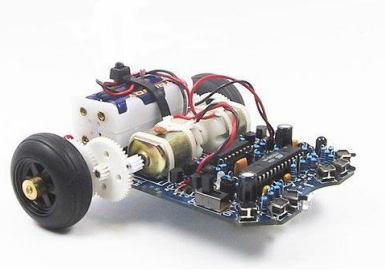
Mikrocontroller-Komponenten (AVR)

- SPI (Serial Peripherial Interface)
  - schnelle synchrone serielle Schnittstelle (bis ca. 1MBit/s)
  - Kommunikation zwischen MCs oder anderen externen Komponenten
  - Interruptquelle
- A/D-Wandler (Analog/Digital-Wandler)
  - Umwandlung analoger Spannungen in Digitalwerte (max. 10-Bit; max. Wandlerrate ca. 200kHz)
  - Analogkomperator
  - Interruptquelle



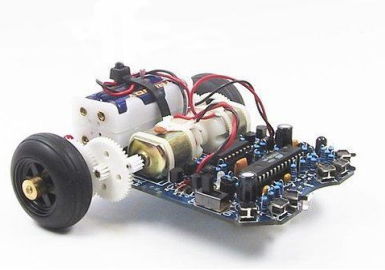
Mikrocontroller-Komponenten (AVR)

- Zähler/Zeitgeber
  - unabhängige und frei konfigurierbare Zähler (8-/16-Bit)
  - Genauigkeit abhängig von der Taktquelle
  - interne und externe Taktquelle
  - Interruptquelle (Compare, Capture, Overflow)
- PWM (Pulse Width Modulation)
  - Erzeugung von digitalen Signalen mit bestimmten Tastverhältnis bei fester Grundfrequenz
  - z.B. Regelung eines Gleichstromverbrauchers
  - D/A-Wandler (Tiefpass nachschalten)



Mikrocontroller-Komponenten (AVR)

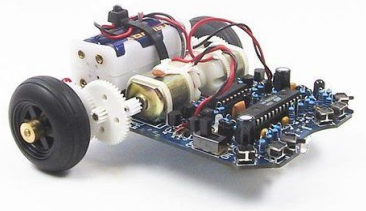
- I2C (Inter IC Bus)/TWI (Two wire Interface)
  - synchroner serieller Bus (2 Drähte)
  - mind. 1 Master und max. 128 adressierbare Slaves
  - bis max. 3,4 Mbit/s
- Watchdog
  - zuschaltbarer unabhängiger Timer zur Programmüberwachung
  - löst nach konfigurierbarer Zeit und "Nichtauffrischen" einen Reset aus
  - sinnvoll z.B. zum Schutz von Hardware



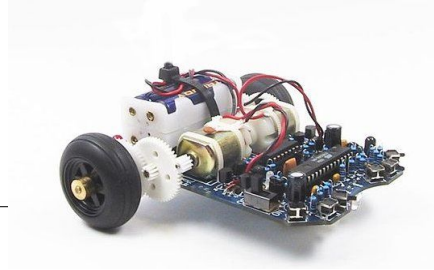
Mikrocontroller-Komponenten (AVR)

- JTAG/ISP (In-System-Programming)
  - spezielle Schnittstelle zum Laden der Firmware und Debuggen
  - JTAG: Standard IEEE 1149.1 der "Joint Test Action Group"

- und einige weitere Spezialkomponenten...



- Was sind Mikrocontroller
- AVR-Mikrocontroller
- Voraussetzungen für Mikrocontroller-Projekte
- Projektbeispiele



Minimal-Voraussetzung MC-Hardware

- ein MC mit einigen weiteren Bauteilen
- Stromversorgung
- ein paar LEDs, Taster, Widerstände
- Programmiergerät

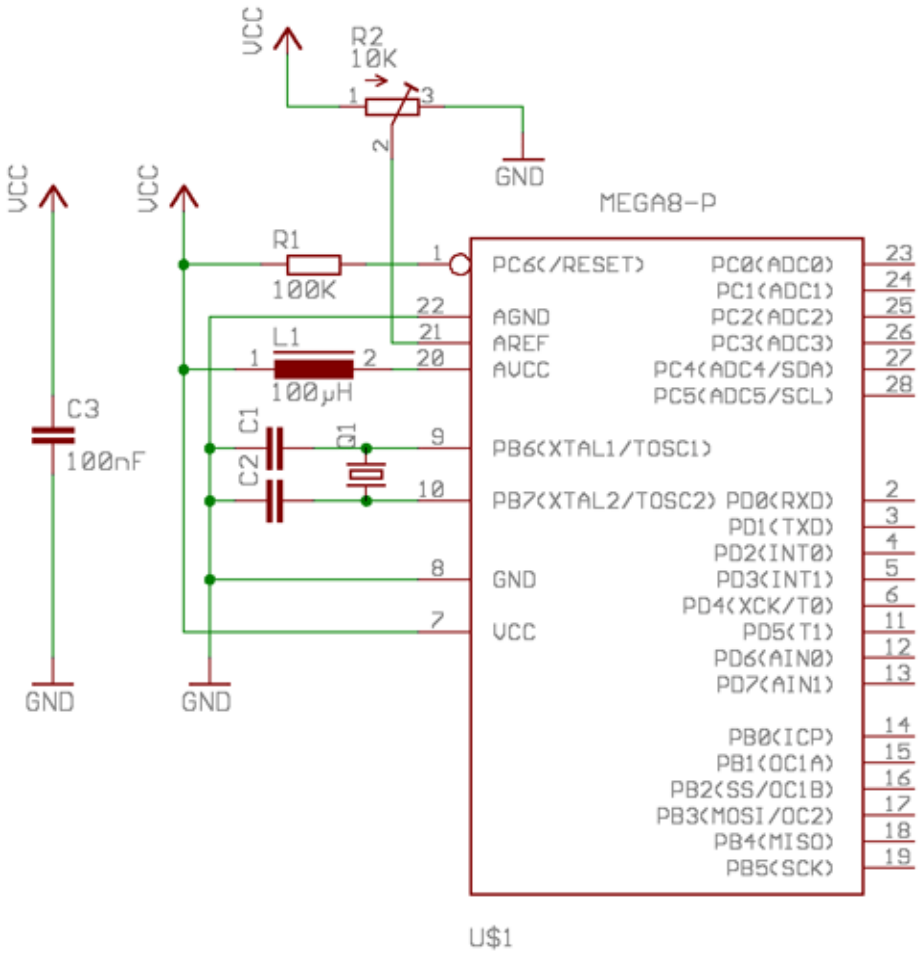
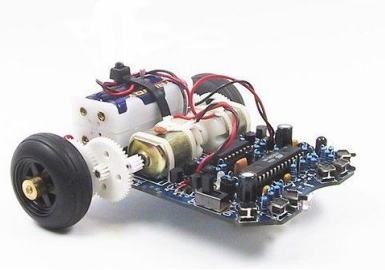
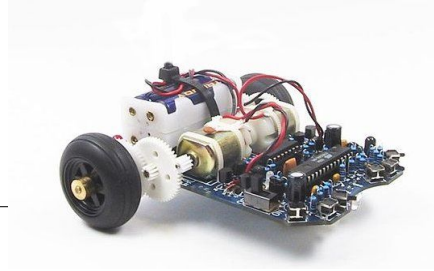


Bild:http://kreatives-chaos.com



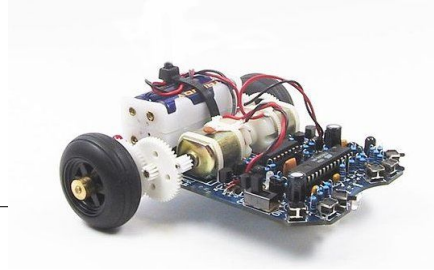
Entwicklerboards

- fertige Baugruppen, die bereits mit diversen I/O- und Programmierschnittstellen ausgestattet sind
- Referenzboards der Firma Atmel: STK200, STK500, STK1000, AVR Butterfly
- zahlreiche kommerzielle und freie Boards: myAVR, Etherrape, RN-Control u.v.m.
- meist werden diverse Zusatzbaugruppen zur Erweiterung angeboten



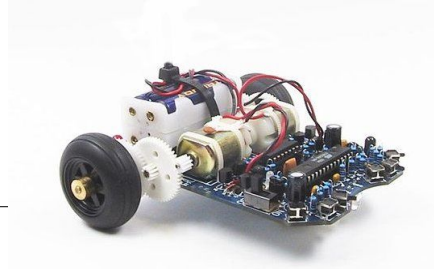
Software zur MC-Programmierung

- Programmiersprachen: Assembler, C/C++, Basic, Pascal, Java, Forth u.v.m.
- Entwicklungsumgebungen für C:
  - avr-gcc, avr-libc, avr-binutils
  - für Windows WinAVR
  - AVR Studio (Firma Atmel), komplette Entwicklungs-umgebung für Windows
  - AVR Eclipse Plugin
- AVRPascal, Bascom AVR, NanoVM etc.



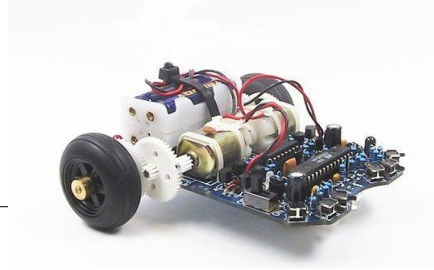
Flashen von Mikrocontrollern

- irgendwie muß der Maschinencode in den Programmspeicher des MC
- Möglichkeiten:
  - Programmieradapter: Hardware, die den MC über speziell dafür vorgesehene Anschlüsse programmiert (USBasp, USBprog, USBisp etc.)
  - Bootloader: Software, die sich in einem geschützten Bereich auf dem MC befindet und z.B. eine serielle Schnittstelle zum Flashen initialisiert
- Programmiertools: avrdude, PonyProg



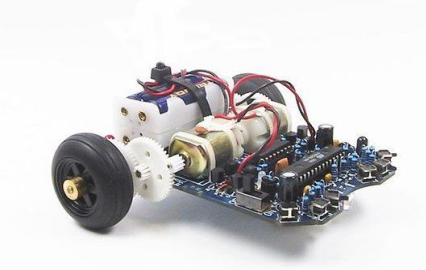
Debuggen von MC-Programmen

- Debuggen "in der Schaltung": JTAG (spezielle Hardware erforderlich)
- Debuggen/Simulation auf dem PC:
  - Windows: z.B. SimulAVR/GDB (WinAVR), AVRStudio
  - Linux: gdb-avr/simulavr
- Fehlersuche durch gezielte Ausgaben:
  - serielle Schnittstelle u.ä.
  - LEDs, LC-Display, Taster/Schalter
  - Multimeter, Oszilloskop, Logikprüfer etc.



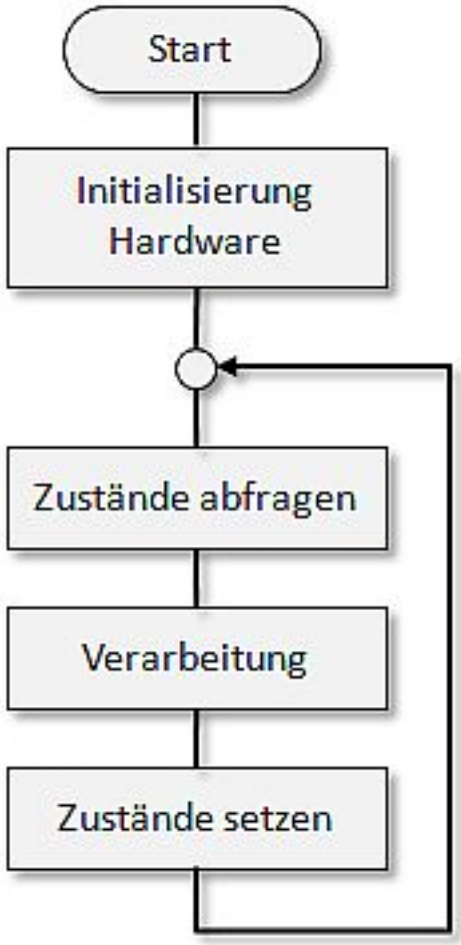
Mikrocontroller-Programme

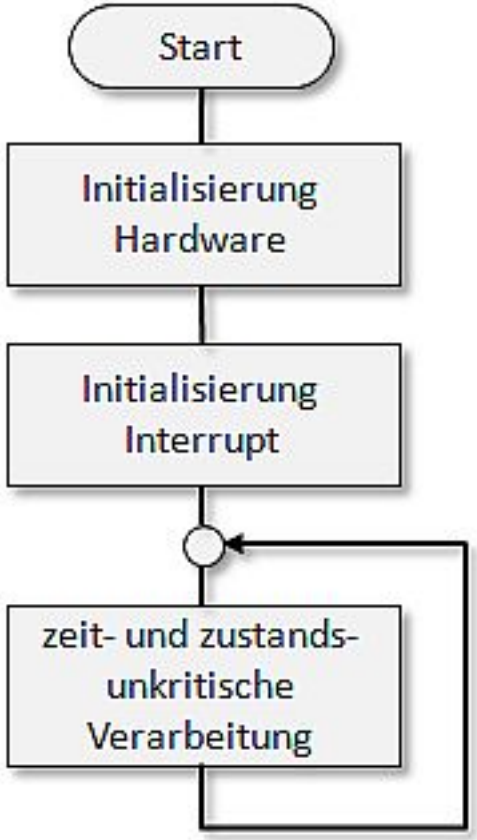
- alles ist EIN Programm, es gibt keine nachladbaren Module
- man muß sich um alles selbst kümmern: Hardware-/ Schnittstelleninitialisierung, Programmsteuerung, Ein-/ Ausgabesteuerung etc.
- sämtliche Programmteile müssen "kooperativ" untereinander sein
- begrenzte Programm- und Datenspeicherbereiche, es gibt keine "Swap-Partition"
- möglichst nicht mit gebrochenen Zahlen rechnen

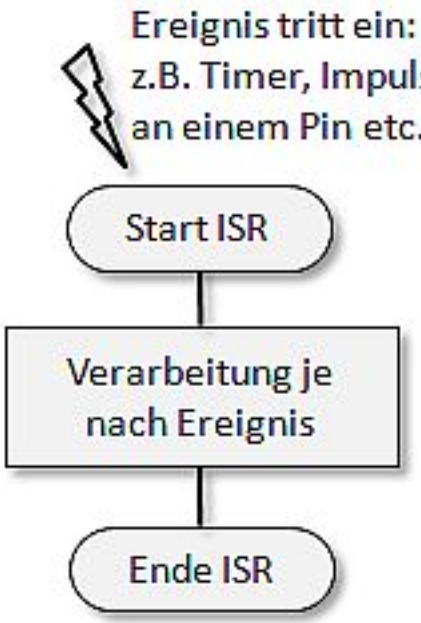


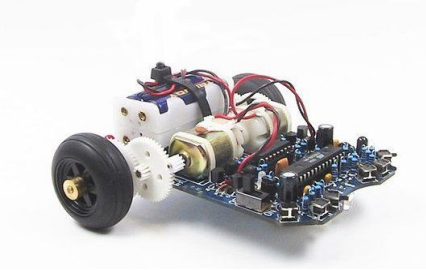
Mikrocontroller-Programme (Struktur)

- Endlosschleife vs. Interruptroutinen

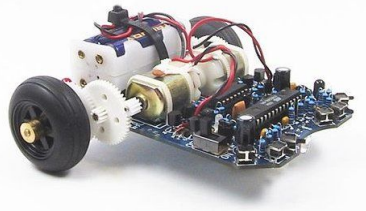








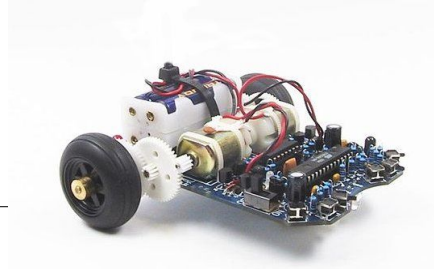
- Was sind Mikrocontroller
- AVR-Mikrocontroller
- Voraussetzungen für Mikrocontroller-Projekte
- Projektbeispiele



Asuro

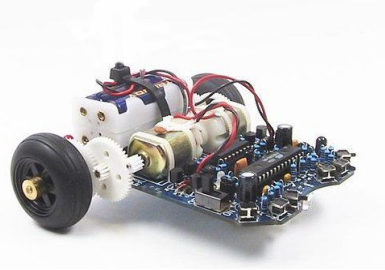
- einfacher mobiler Roboter-Bausatz, entwickelt vom Deutschen Zentrum für Luft- und Raumfahrt (DLR)
- Hersteller: Arrexx Engineering
- http://www.asurowiki.de

Bild:http://www.wikipedia.de



Asuro (Hard-/Software)

- ideal für Einsteiger
- Hardware: ATMega8 (8MHz, Flash 8kB, SRAM 1kB, EEPROM 512B), 2 Motorbrücken, 6 Taster, 2 Fototransistoren, 2 Odometriesensoren, IR-Schnittstelle, diverse LEDs)
- Bootloader bereits vorhanden (Flashen via RS232 -> Infrarot-Schnittstelle)
- Programmierung in C (eigene C-Bibliothek)
- zahlreiche Hardware-Erweiterungen (z.B. USB, Ultraschallortung, LCD etc.)



Etherrape

- einfach aufzubauender Mikrocontroller-Bausatz
- u.a. Ethernet-Schnittstelle
- http://www.lochraster.org

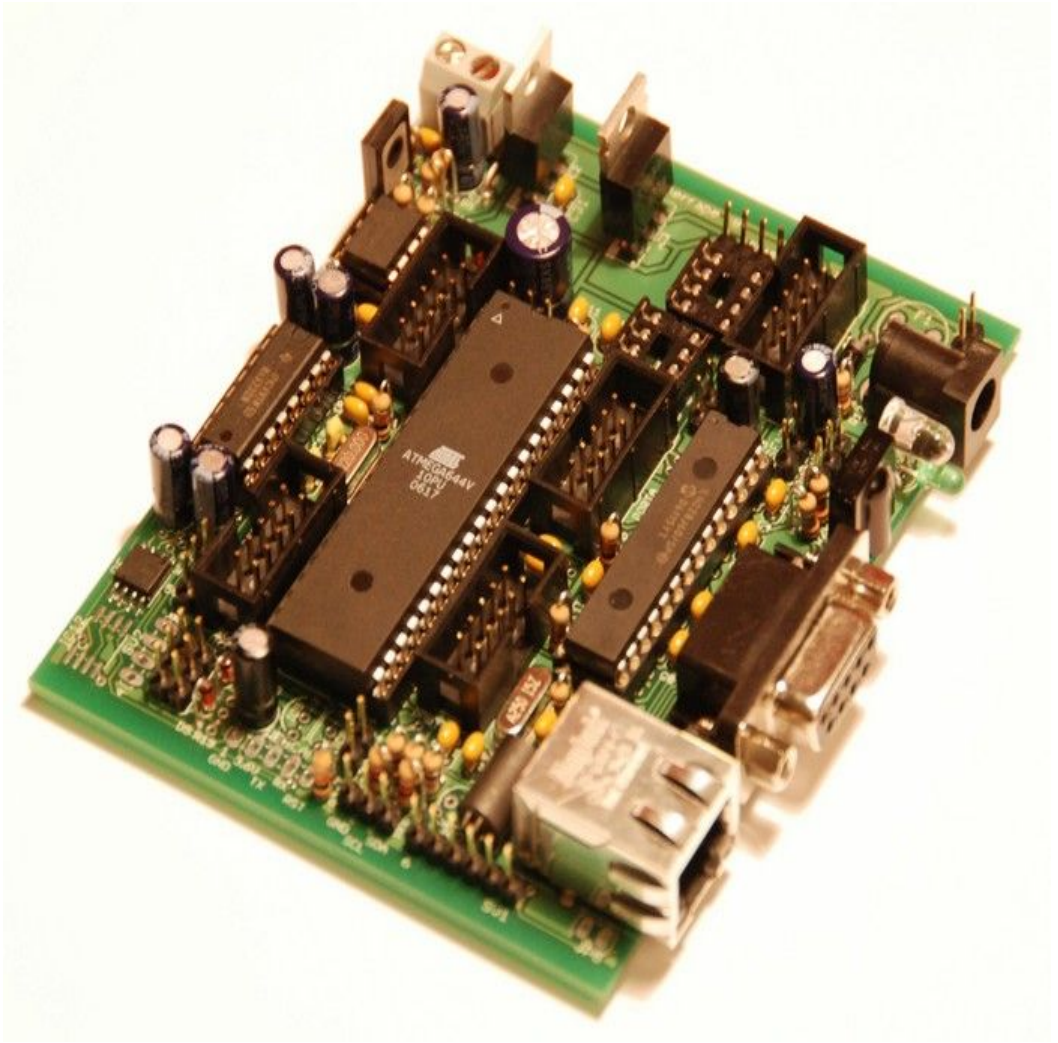
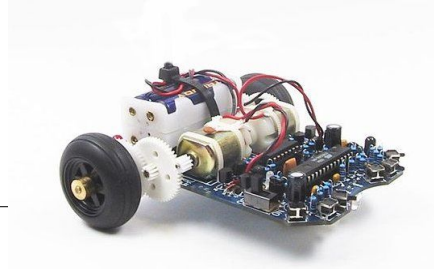
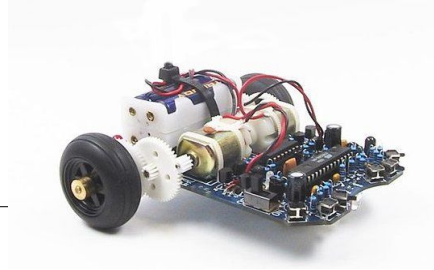


Bild:http://www.lochraster.org



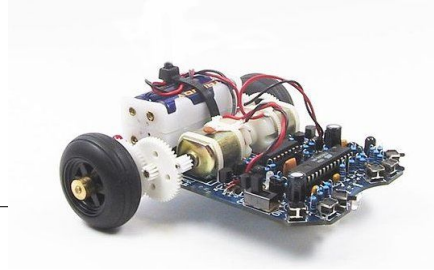
Etherrape-Hardware

- Mikrocontroller: Mega644 (20MHz, EEPROM 2KByte, Flash 64KByte, SRAM 4KByte), Data-Flash (2MByte)
- Schnittstellen:
  - Ethernet-Schnittstelle (10MBit)
  - RS232-Schnittstelle
  - IR-Empfänger/-Sender
  - I2C
  - SPI
- optional: RS485/422, OneWire-Bus, Handy-Cam MCA-25, LC-Display u.v.m



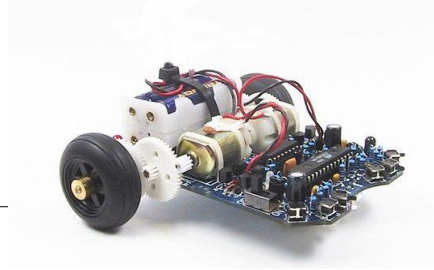
Etherrape-Software (original)

- Bootloader (via RS232)
- originale Firmware
  - Hardwareansteuerung
  - TCP/IP-Stack (uIP)
  - FS20-Protokoll (Hausautomatisierung von ELV, Conrad)
  - RC5 (IR-Fernbedienungsprotokoll)
  - Kommandozeilen-Tool
  - Syslog
  - LCD-Ansteuerung
  - in Entwicklung: Webserver, Filesystem



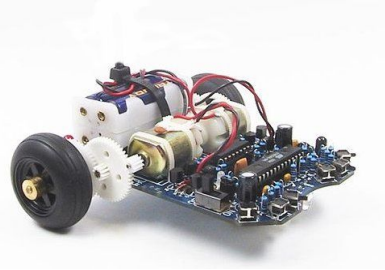
Etherrape-Software (Erweiterungen)

- "ethersex" (http://www.ethersex.de)
  - TFTP
  - IPv6
  - verschlüsselte Kommunikation
  - DNS, DynDNS, SNTP
  - und einiges mehr
- librape (http://brokenpipe.de/cgi-bin/gitweb.cgi)
  - Zusammenfassung vieler Grundfunktionen der Originalfirmware in einer Bibliothek



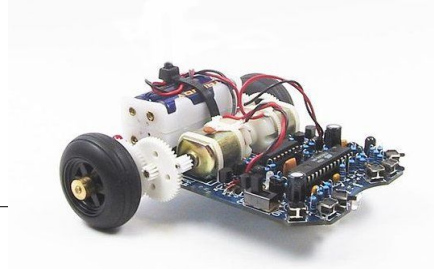
TCP/IP-Stack mit uIP

- ein extrem kleiner TCP/IP-Stack, speziell für embedded 8Bit-Mikrocontroller
- Autor: Adam Dunkels
- http://www.sics.se/~adam/uip/index.php/Main_Page
- BSD-Lizenz
- TCP und UDP
- IPv4 (vorbereitet für IPv6)
- sehr einfach in eigene Programme einzubinden



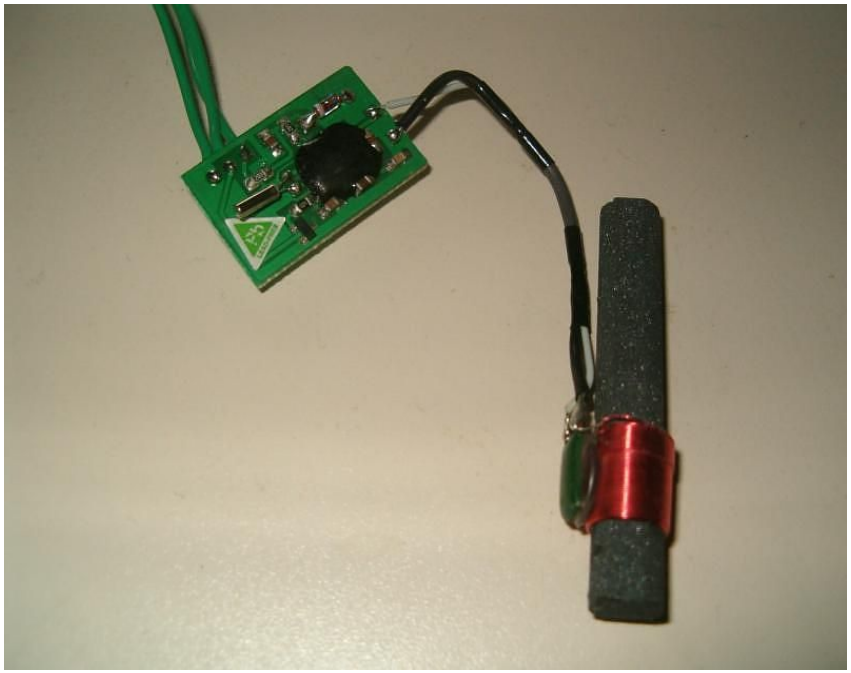
Projekt: Etherrape-Uhr

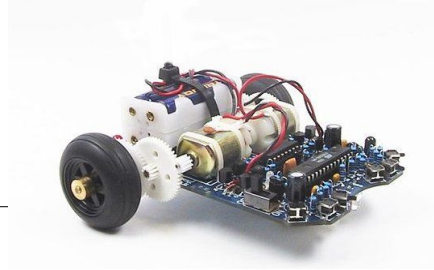
- Warum die genaue Zeit?
  - Backups, Jobsteuerung, Zeitstempel etc.
- Woher die genaue Zeit?
  - DCF77
    - Langwellensender in der Nähe von Frankfurt/M.
    - Zeitinformation wird von einer Atomuhr gespeist
    - Datum-/Zeitinformation ist in 59, über eine Minute verteilte Austastlücken codiert
  - NTP und SNTP



Etherrape-Uhr (Hardware)

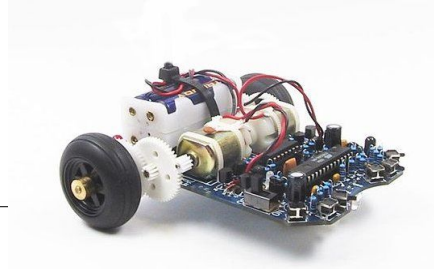
- Etherrape
- DCF77-Empfangsmodul (1 Port + 1 Port für Status-LED)
- LC-Display (8 Ports)



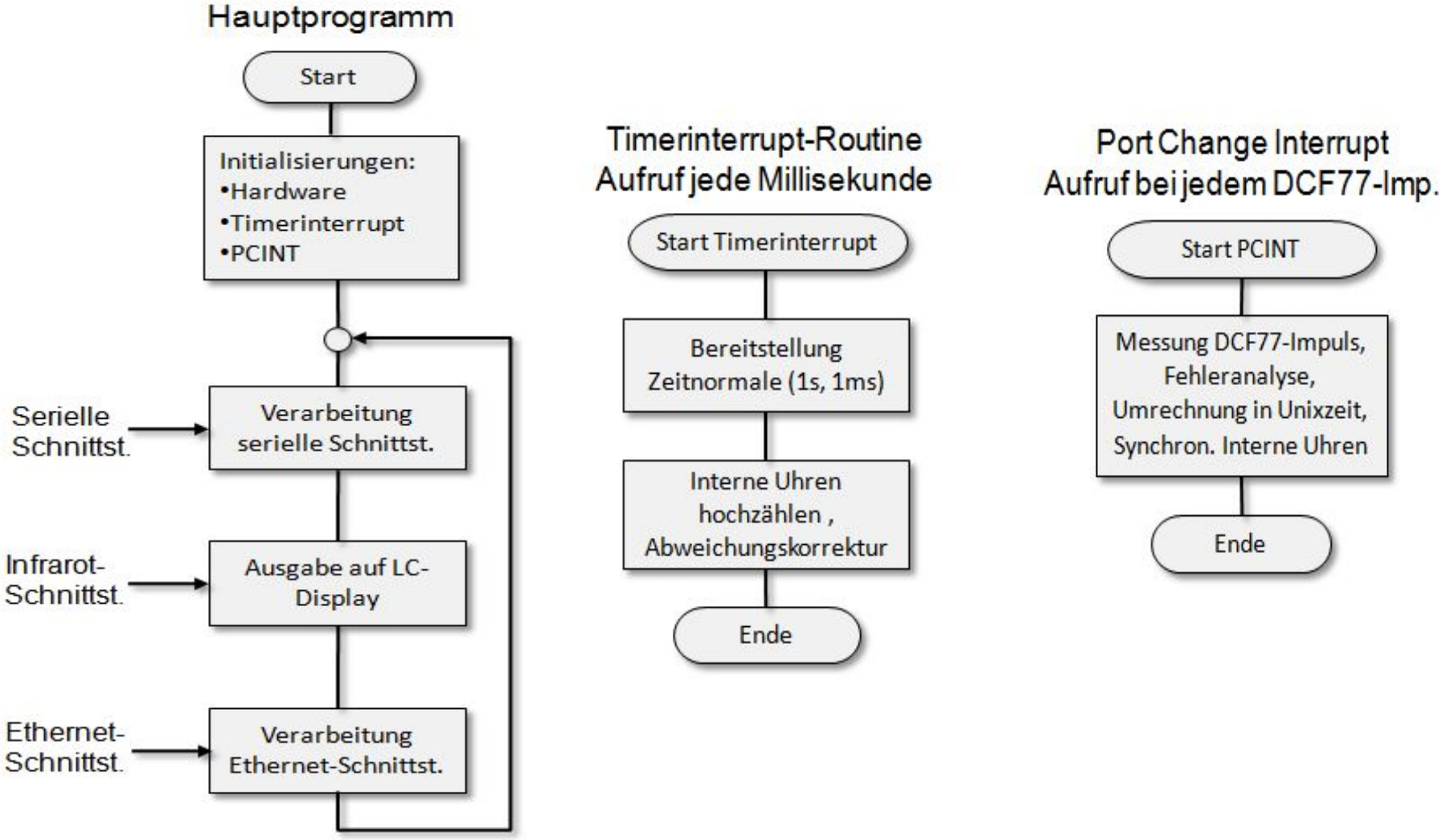


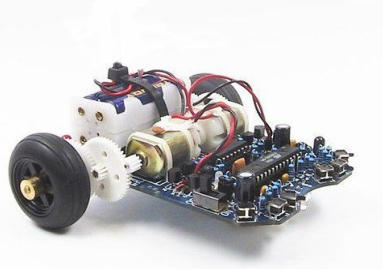
Etherrape-Uhr (Software)

- http://wiki.lochraster.org/wiki/DCF77
- zwei Interrupt-Routinen
  - fortlaufender Timer zur Impulsdauermessung des DCF77-Signal und Bereitstellung Sekundentakt
  - Port Change Interrupt Routine zur Erkennung und Verarbeitung des DCF77-Signals
- ein "Sekundenkorrektur-Algorithmus" für interne Uhr
- diverse Datum-/Zeit-Umrechnungsroutinen
- Kommandozeilen-Interface zum Steuern via Ethernet



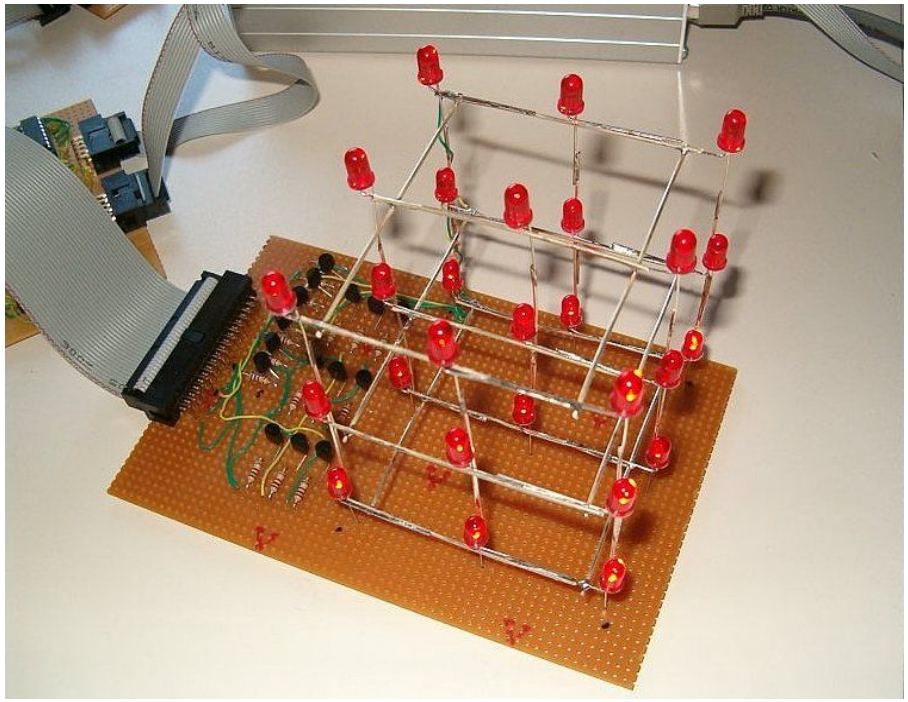
Etherrape-Uhr (Software)

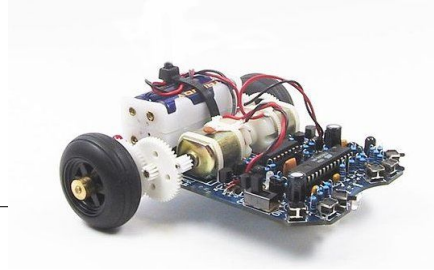




Projekt: 3D-LED-Würfel

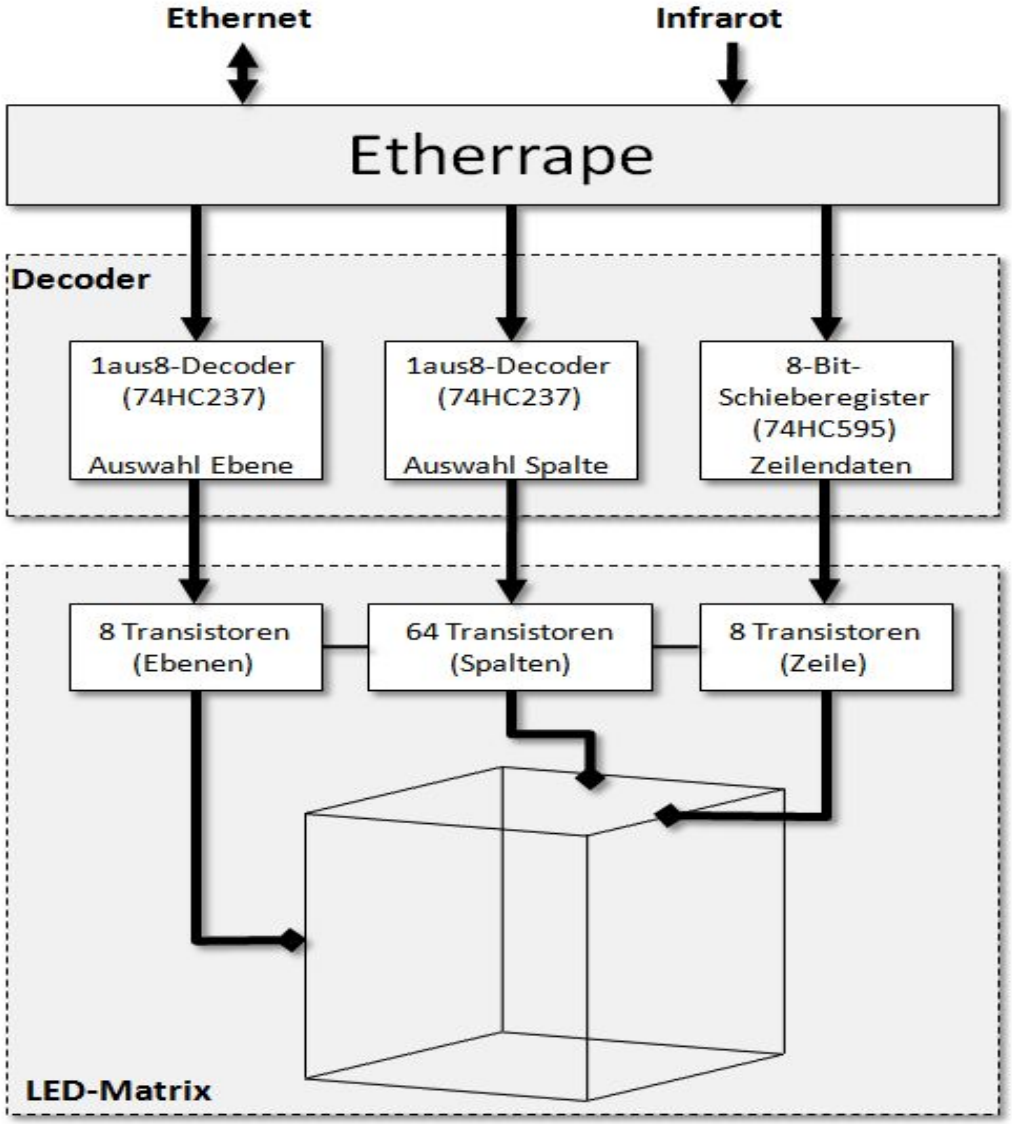
- http://www.bralug.de/wiki/3D-LED-Display
- 8x8x8 LEDs, einzeln ansteuerbar in einem Würfel (geplant)
- derzeit als 3x3x3-LED-Würfel realisiert

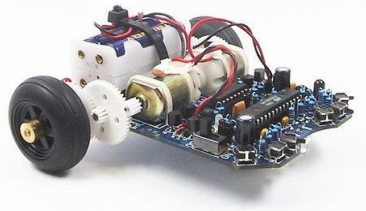




3D-LED-Würfel (Hardware)

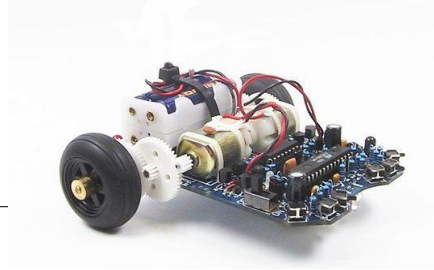
- Etherrape
- Decoder
- LED-Matrix
- IR-Fernbedienung



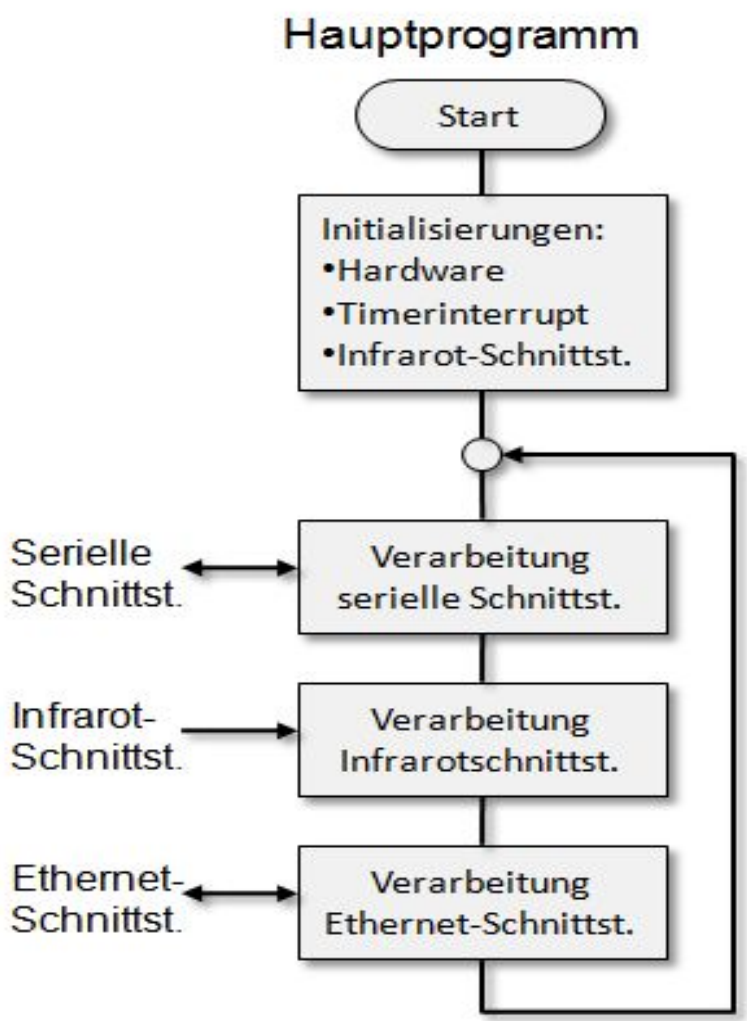


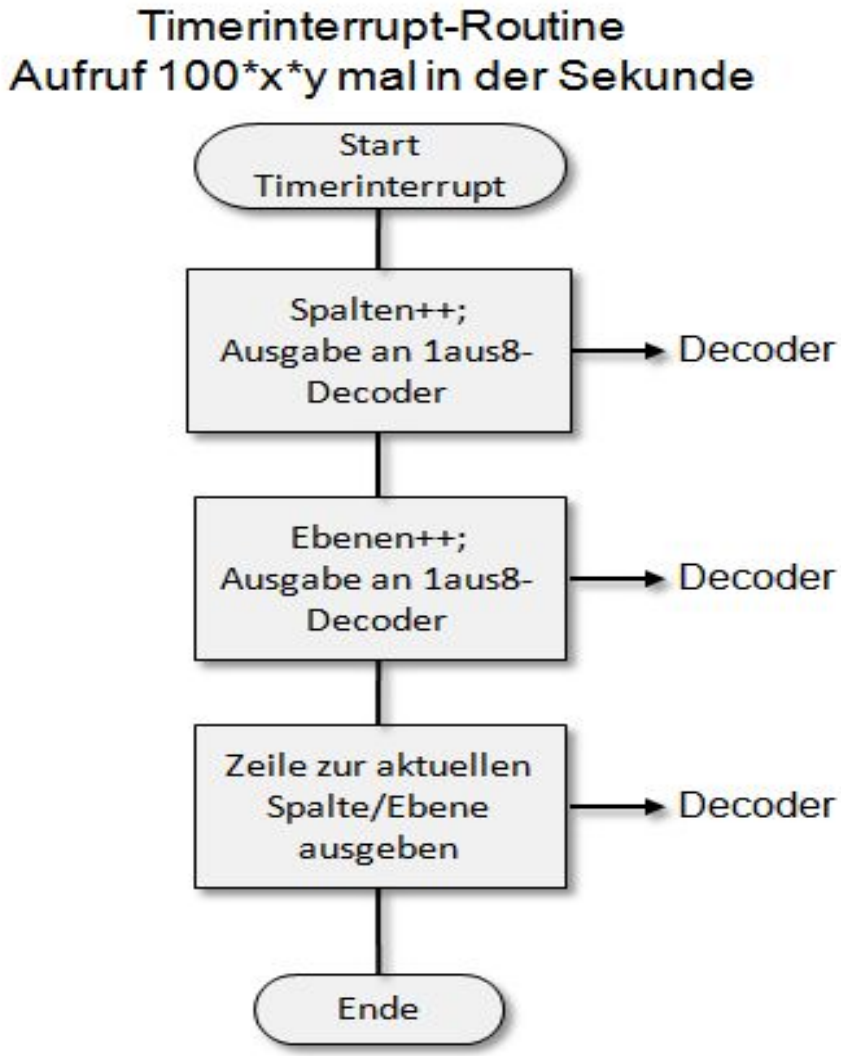
3D-LED-Würfel (Software)

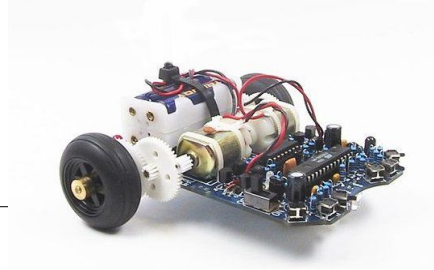
- per Konzept ein "Display-Server"
- ein, von allen anderen Programmteilen unabhängig laufender Timer-Interrupt
  - Ausgabe des "Bildes" ca. 100x in der Sekunde (Multiplexing)
  - Zerlegen der 512 LED-Zustände in Ebene/Spalte/Zeile
  - Ansteuerung des Decoders
- Kommandozeilen-Interface zum Steuern via Ethernet
- IR-Empfangsroutinen
- einige feste Animationen



3D-LED-Würfel (Software)

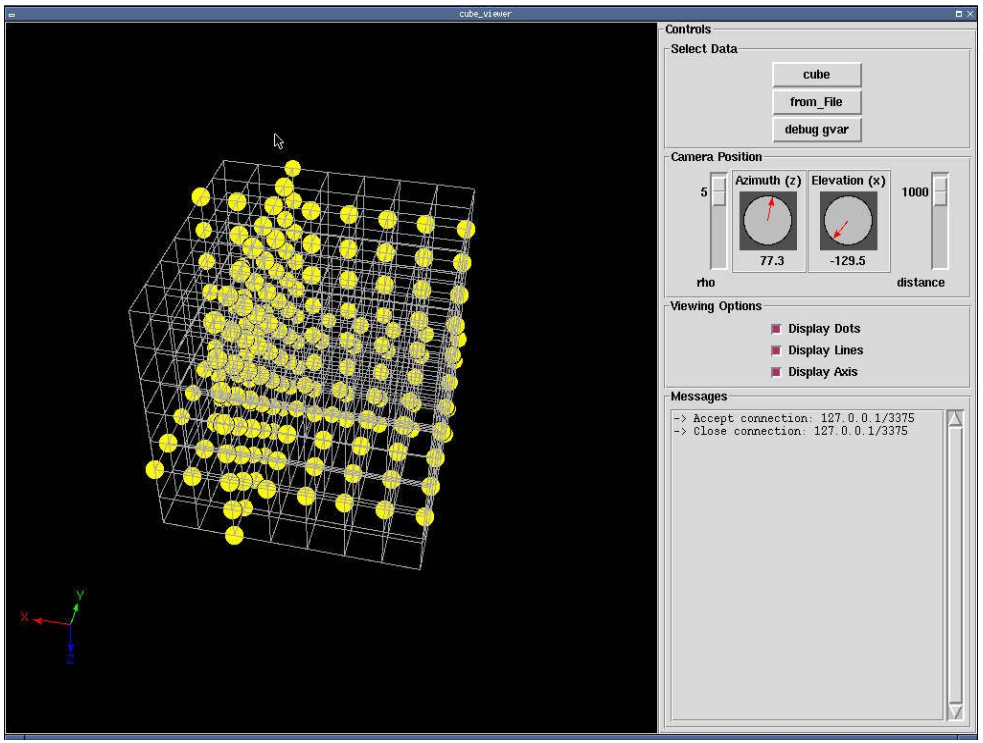


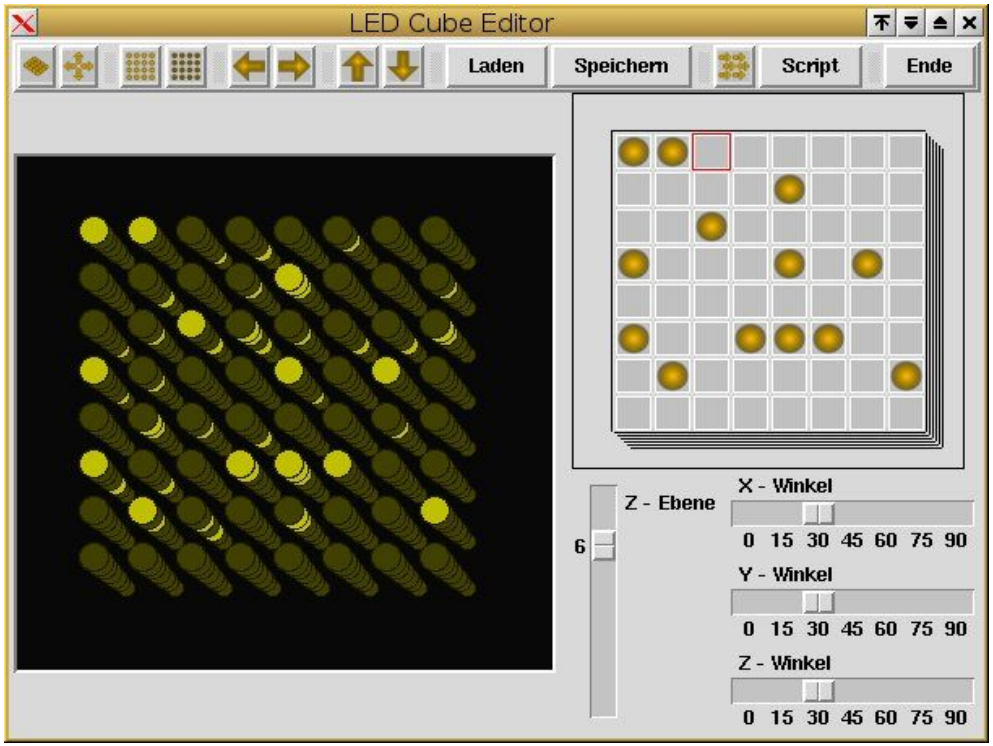


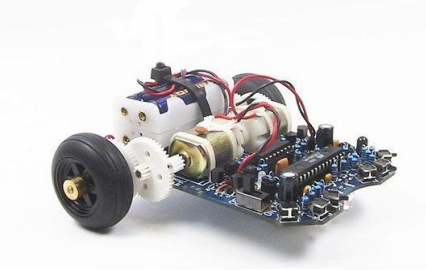


3D-LED-Würfel (Simulation)

- 3D-Würfel-Simulator
- 3D-Würfel-Editor

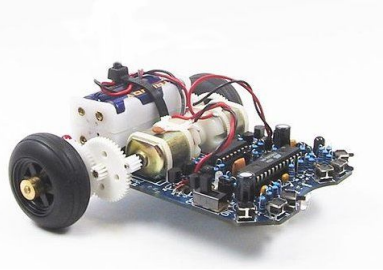






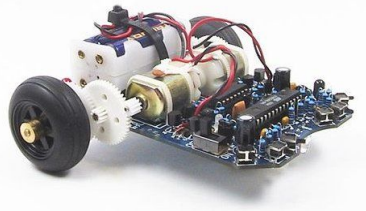
Betriebssysteme für MCs?

- Definition OS: Software die die Ressourcen eine Computers verwaltet und die Ausführung von Programmen steuert
- erweiterte Definition: ... sowie die wichtigsten Systemtools zur Verfügung stellt
- diverse Betriebssysteme für die verschiedensten MC-Familien:
  - unix-/linux-basierende OS (z.B. μCLinux, RTLinux)
  - spezialisierte OS (z.B. FreeRTOS, contiki, AvrX, GAOS)
- Befehlsinterpreter auf dem MC: Möglichkeit "externen" Code zu laden/auszuführen (z.B. uBasic)



Weiterführende Informationen

- Internet (u.a.):
  - http://www.mikrocontroller.net
  - http://www.roboternetz.de/
- Bücher (u.a.):
  - G.Schmitt; "Mikrocomputertechnik mit Controllern der Atmel AVR-RISC-Familie"; Oldenbourg Verlag
  - W.Trampert; "Messen, Steuern und Regeln mit AVR-Mikrocontrollern"; Franzis Verlag
  - Brinkschulte, Ungerer; "Mikrocontroller und Mikroprozessoren"; Springer Verlag



Danke für die Aufmerksamkeit!