Lichtwürfel

Revision 2.0

Michael Frey
ridcully at cccmz dot de

26. Juli 2008

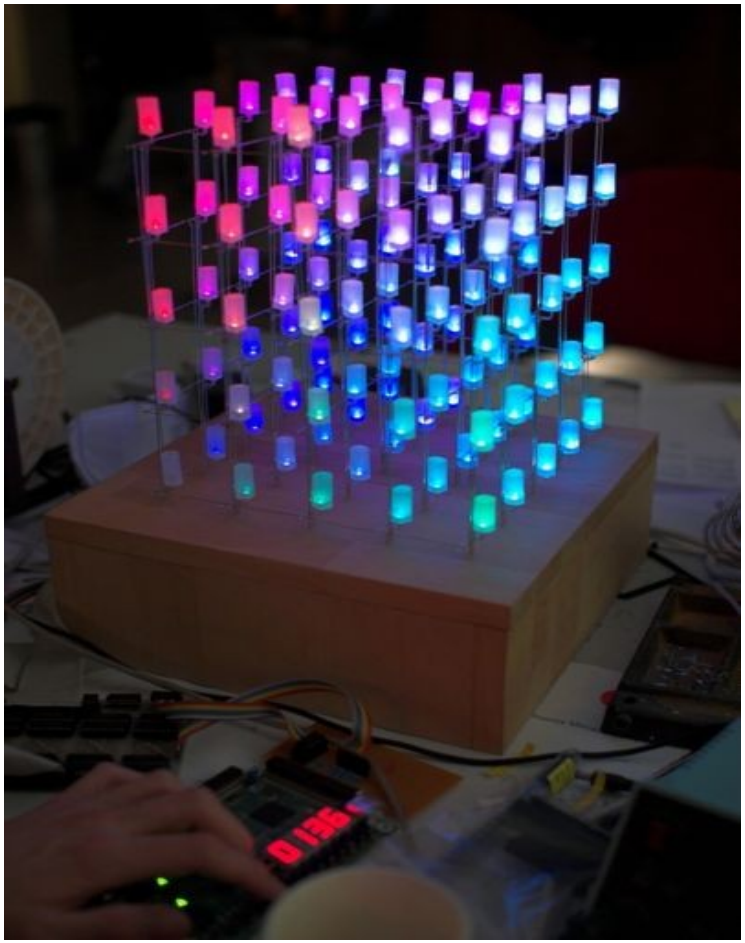
Überblick

- Ziel
- Motivation
- Idee
- Lichtwürfel Revision 1.0
- Hardware
- Software
- Ausblick
- Zusammenfassung

Ziel

Ziel des Projektes ist es einen Lichtwürfel aus roten diffusen LEDs zu bauen. Die LEDs sind dabei in einem dreidimensionalen Gitter angeordnet. Jede LED soll einzeln ansteuerbar sein, womit es möglich ist Animationen und 3D Grafiken anzuzeigen. Die visuellen Fähigkeiten des Lichtwürfels sollen demonstriert werden.

Motivation



- Inspiration: James Klar (3D Display Cube, 2005), Network Wizards (Cubatron, 2006)
- Wenig bis gar keine Bauanleitungen und dann meistens mit Lücken
- Ziel: 5x5x5 Lichtwürfel mit einer guten Bauanleitung

Quelle (Bild): Bochum “Das Labor: Wiki

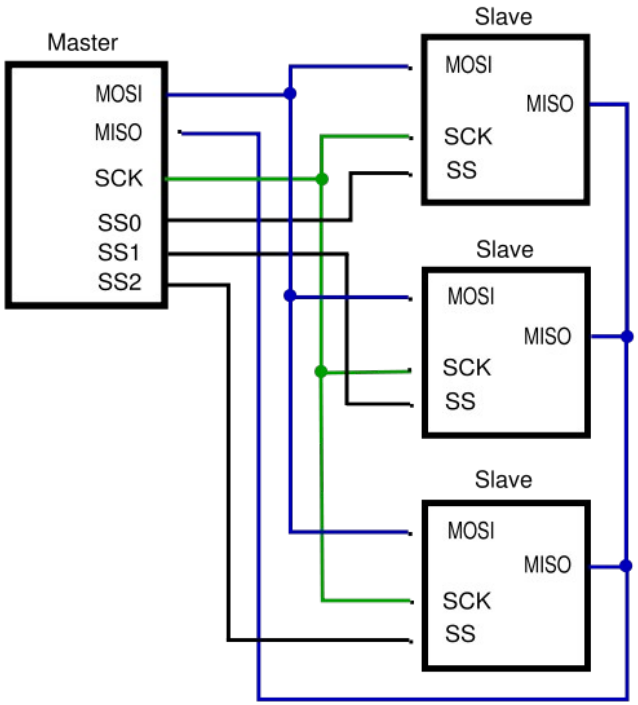
Idee

- Alle Anoden einer Ebene sind verbunden (am Decoder Ausgang sind 100 LEDs)
- Alle Kathoden übereinander sind miteinander verbunden (an einem Atmega Ausgang sind 10 LEDs)
- Setzen von 5 Volt an der Ebene und 0V an der Säule lassen die entsprechende LED leuchten

Hardware

- SPI
- Arduino
- Ebenensteuerung
- Säulensteuerung

Serial Peripheral Interface - I



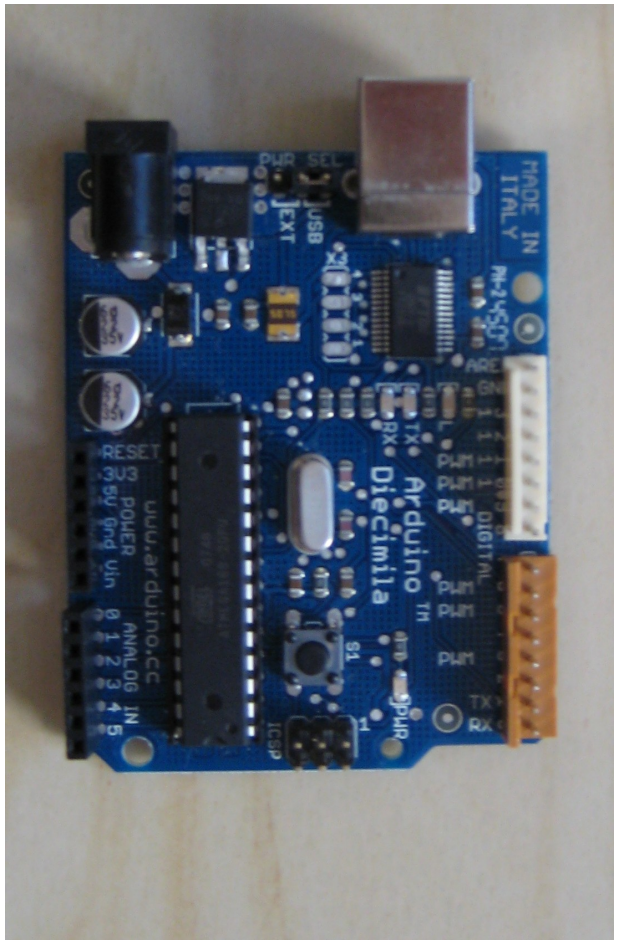
- Synchroner serieller Bus von Motorola
- Master/Slave Prinzip
- I/O Pins: SS, MISO, MOSI und SCK
- Register: SPDR, SPSR und SPCR

Quelle (Bild): Wikipedia

Serial Peripheral Interface - II

- Funktionsweise:
  - SS auf Low (Signalisierung für den Slave zur Übertragung)
  - Schreiben der Daten:
    - Byte ablegen in SPDR (SPI Data Register)
    - Solange warten bis im SPCR (SPI Control Register) am Bit SPIF (SPI Interrupt Flag) eine 1 steht
    - Vorgang wiederholen bis alle Daten geschrieben sind
  - SS auf High (Signalisiert für den Slave das Ende der Übertragung)

Arduino - I



- Open Source Board
- ATmega168 mit 20 Mhz
- USB Schnittstelle
- Serielle Kommunikation mit einem Steuerrechner
- SPI Kommunikation mit den Säulen
- “Pin” Kommunikation mit den Ebenen
- Erweiterungen für Bluetooth, ZigBee, Ethernet und GPRS/GPS verfügbar

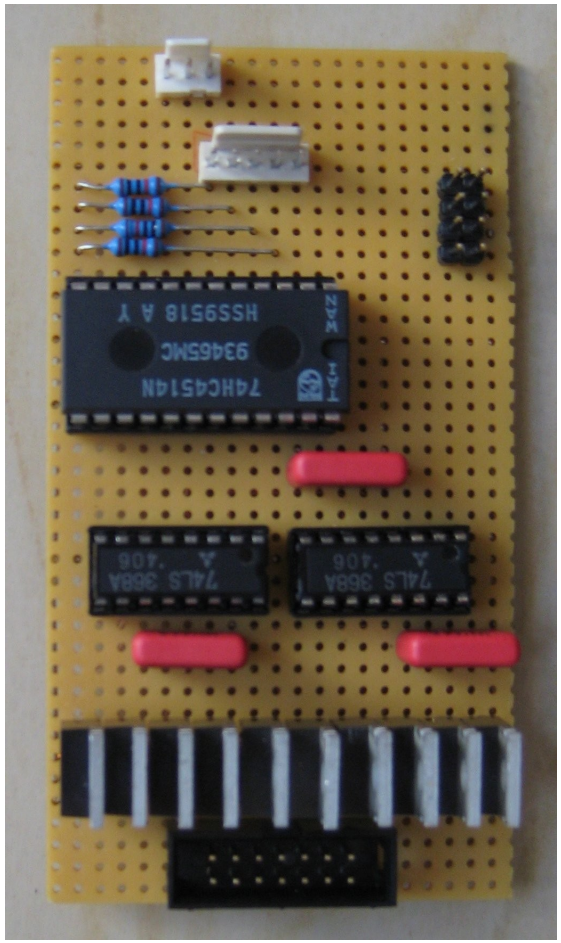
Arduino - II

- Routinen auf dem Arduino:
  - setup(void)
    - Definiert I/O Pins als Input/Output
    - Setzt den Prescaler für das SPI
    - Setzt das SPI Control Register
    - Setzt den Prescaler für den Timer Overflow
    - Aktiviert den Interrupt Timer Overflow
  - loop(void)
    - Liest Säulendaten von der seriellen Schnittstelle
    - Schreibt Säulendaten per SPI (1 Byte Header, 250 Byte Säulendaten)
  - writeData(unsigned char data)
    - Schreibt ein Byte per SPI

Arduino - III

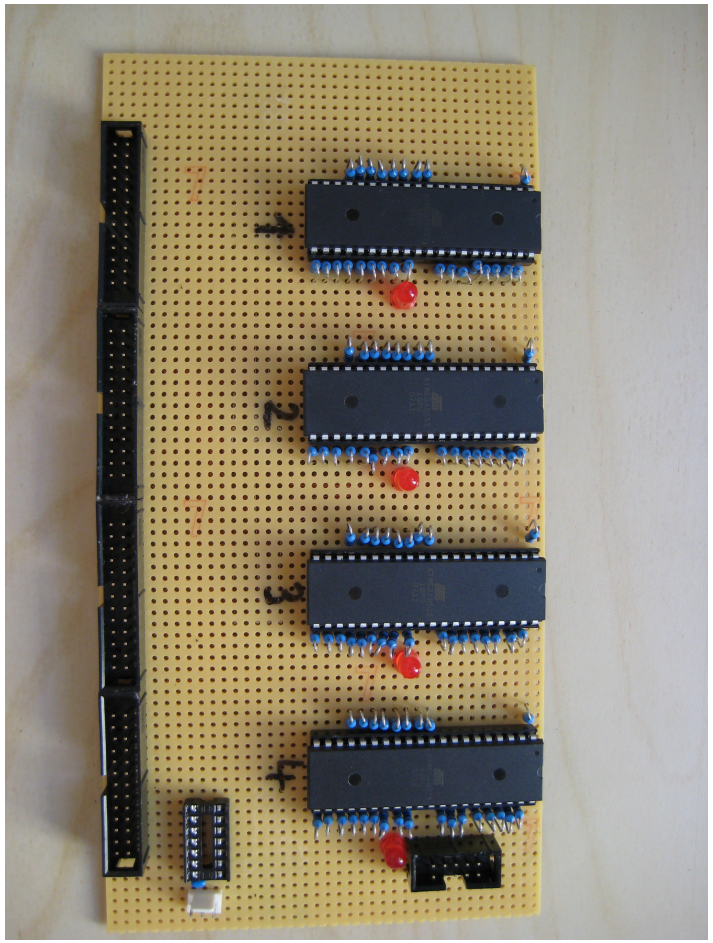
- Fortsetzung Routinen:
  - ISR(TIMER2_OVF_vect)
    - Schreibt Ebenendaten per SPI (1 Byte Header, 1 Byte Ebene) an alle Slave Devices
    - Schaltet Ebenen
      - Zählervariable wird mit 1 initialisiert (static)
      - Prüfung ob Bit gesetzt sind (Maskierung mit 0x01, 0x02, 0x03, 0x04). Falls das Bit gesetzt ist wird der entsprechende Pin auf High “gezogen”, ansonsten Low
      - Nach dem zehnten Durchlauf wird die Zählervariable auf 1 zurückgesetzt

Ebenensteuerung - I



- Eingangspins verbunden mit Arduino Board
- Codierung entsprechend Datenblatt
- Jumper zum testen der Eingänge/Ausgänge
- Umschaltvorgang über den Arduino
- Inverter invertieren das Ausgangsignal des Decoders
- MOSFETs haben eine inverse Logik

Säulensteuerung - I



- Vier ATmega8515 zur Steuerung der Säulen
- Jeder ATmega8515 ist für 25 Säulen verantwortlich
- Empfang der Daten über SPI, aktuell werden keine Daten zurückgesendet
- Externe Taktquelle geplant, aber FUSEs noch nicht gesetzt
- Debug LEDs zur Kontrolle

Säulensteuerung - II

- Routinen auf den Säulen:
  - init(void)
    - Initialisiert die Ports (Input/Output)
    - Initialisiert den Buffer der die Säuleninformationen hält (250 Bytes)
  - receiveData(void)
    - Funktion zum empfangen von Daten per SPI
  - main(void)
    - Empfängt Daten und unterscheidet anhand des Headers ob ein weiteres Byte oder 250 Bytes folgen. Werden Informationen zur Ebenenumschaltung geliefert so wird die Funktion zur umschaltung der Ebenen aufgerufen
    - Debug LED blinkt sobald Daten per SPI eintreffen

Säulensteuerung - III

- Fortsetzung der Routinen:
  - switchRow(unsigned char row)
    - Index wird ermittelt anhand des Parameters ((rowNumber -1) * 25)
    - Für jeden Port wird in einer Schleife geprüft ob eine LED an ist, falls ja wird eine 0 auf die entsprechende Stelle geshiftet und auf den Port “ge-odert”
    - Der Index wird dabei in achter Schritten erhöht (0, 8, 9, 17)

Software

- Bibliothek zur Kommunikation mit dem Würfel
  - Kommunikation mit der seriellen Schnittstelle
  - Thread der Daten periodisch aus einem Buffer liest und auf die serielle Schnittstelle schreibt
  - Thread gesichert über Semaphore
- MP3 Player Plugin zur Spektrum Analyse
  - Plugin vorhanden
  - Dokumentation nicht vorhanden
  - Magic Numbers im Code

Demo

Ausblick

- Modellierungssoftware für Animationen
- Erweiterung der Bibliothek um Schriftschnittstelle
- Netzwerkzugriff auf den Lichtwürfel
- Pong/Snake mit der Wiimote und dem Lichtwürfel
- Modifikationen am Projekt:
  - Steuerung realisieren durch FPGA
  - RGB Leuchtdioden
  - Lichtwürfel für die Luminale

It's not a bug, it's a feature!

- Anode/Kathode
- Programmierboard:
  - “Wieso hat das Programmierboard einen 8 Pin Eingang, aber der ISP nur ein 6 Pin Kabel?”
- Säulensteuerung:
  - “So jetzt noch den Slave Select Eingang an Pin 44 - moment Pin 44 bei einem DIL-40 Sockel?”
  - Umsetzen der Widerstände und ändern der Kupferlackdrahtverbindungen. Yeah!
- Testplatine
  - Wieso leuchtet da nichts? VCC Leitung mit GND Leitung verwechselt, Debug LED falsch herum gelötet.

Zusammenfassung

- Zahlen:
  - 130 Meter Silberdraht
  - 1000 Leuchtdioden
  - 100 Widerstände, 10 Kondensatoren, ...
  - 4 ATmega8515, 1 ATmega168, 1 Decoder
  - Materialkosten: 320 Euro
- Interessantes Projekt
- Spannende Erweiterungsmöglichkeiten

Vielen Dank für die Aufmerksamkeit!