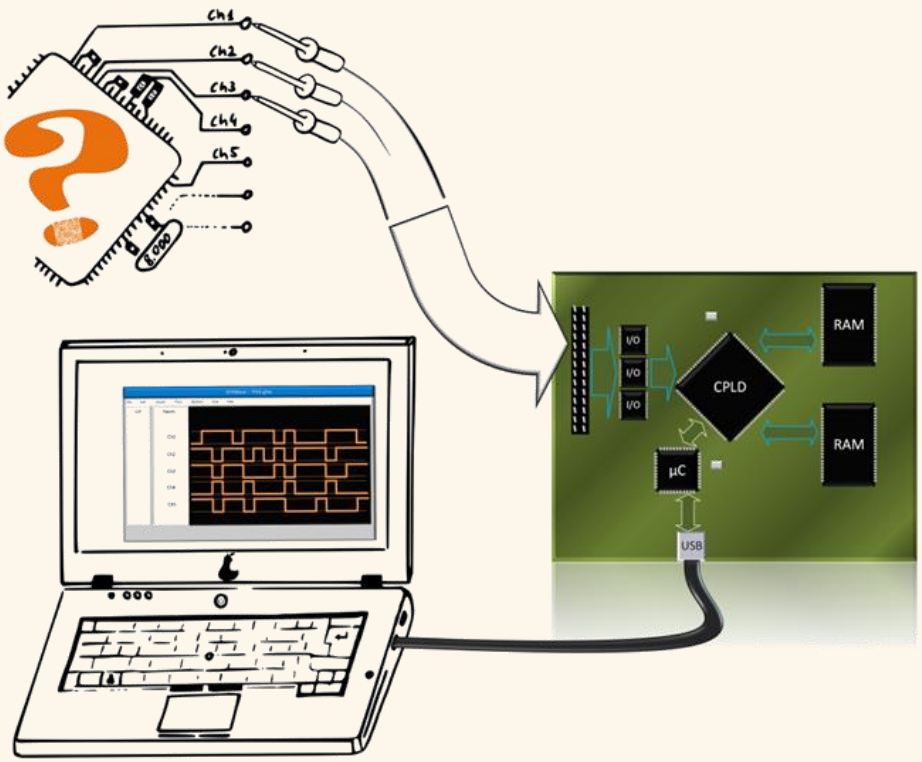
Logikanalyse für Alle

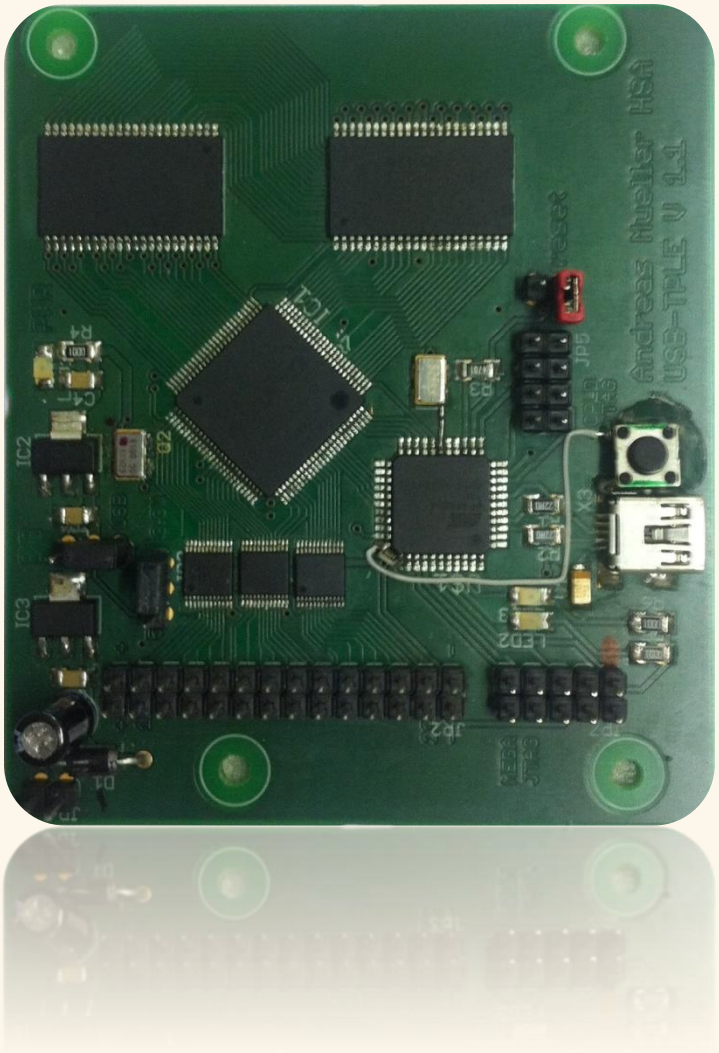


Projektarbeit des Studiengangs
Technische Informatik
Semester 5
Hochschule Augsburg
WS 2013/2014

Logikanalyse für Alle

Ablauf:

1. Vorstellung des Teams
2. Motivation
3. Überblick
4. CPLD
5. Mikrocontroller
6. PC-Seite
7. Ausblick
8. Fragen?



Logikanalyse für Alle

Das Team und die Aufgabenbereiche

Gareis Andreas	CPLD-Firmware
Vockinger Stefan	CPLD-Firmware
Weber Matthias	JTAG, sigrok-Anbindung
Krafft Bernd	µC-Firmware
Bunje Nils	PC-Kommunikations-Software
Echter Patrick	µC-Firmware, Projektmanagement

Logikanalyse für Alle

Motivation:

Es sind viele verschiedene Themen der Technischen Informatik enthalten.

- Hardware
- Software für den Mikrocontroller in C
- Hardware-Beschreibungssprache VHDL für den CPLD
- Softwareentwicklung unter Linux

Das gesamte Projekt ist OpenSource.

Logikanalyse für Alle

Überblick

Logikanalysator zur zeitlichen Erfassung und Anzeige von digitalen Signalen.

Eigenschaften:

- 8 Kanäle
- Abtastung bis 6,25MSamples/s
- Auflösung maximal 160ns
- Speicher für 262.144 Messungen
- Einfache Ansteuerung über USB
- Firmware-Update per USB
- Open Hard- und Software
- Preiswert
- Handelsübliche Bauteile
- Eigenentwicklung

Logikanalyse für Alle

CPLD

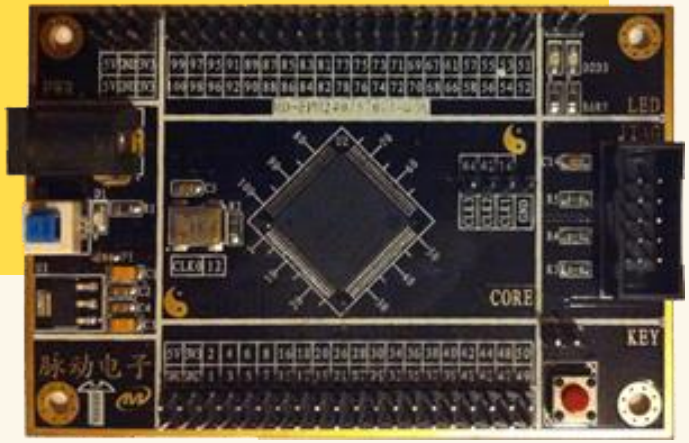
- Complex Programmable Logic Device
- Funktion durch Verschaltung von internen Logikelementen
- Möglichkeit zur Erstellung von schnellen, parallel arbeitenden Modulen
- Messdaten müssen möglichst schnell in den Speicher geschrieben werden
- Altera Max II mit 240 Logikelementen
- Programmierbar über JTAG

Logikanalyse für Alle

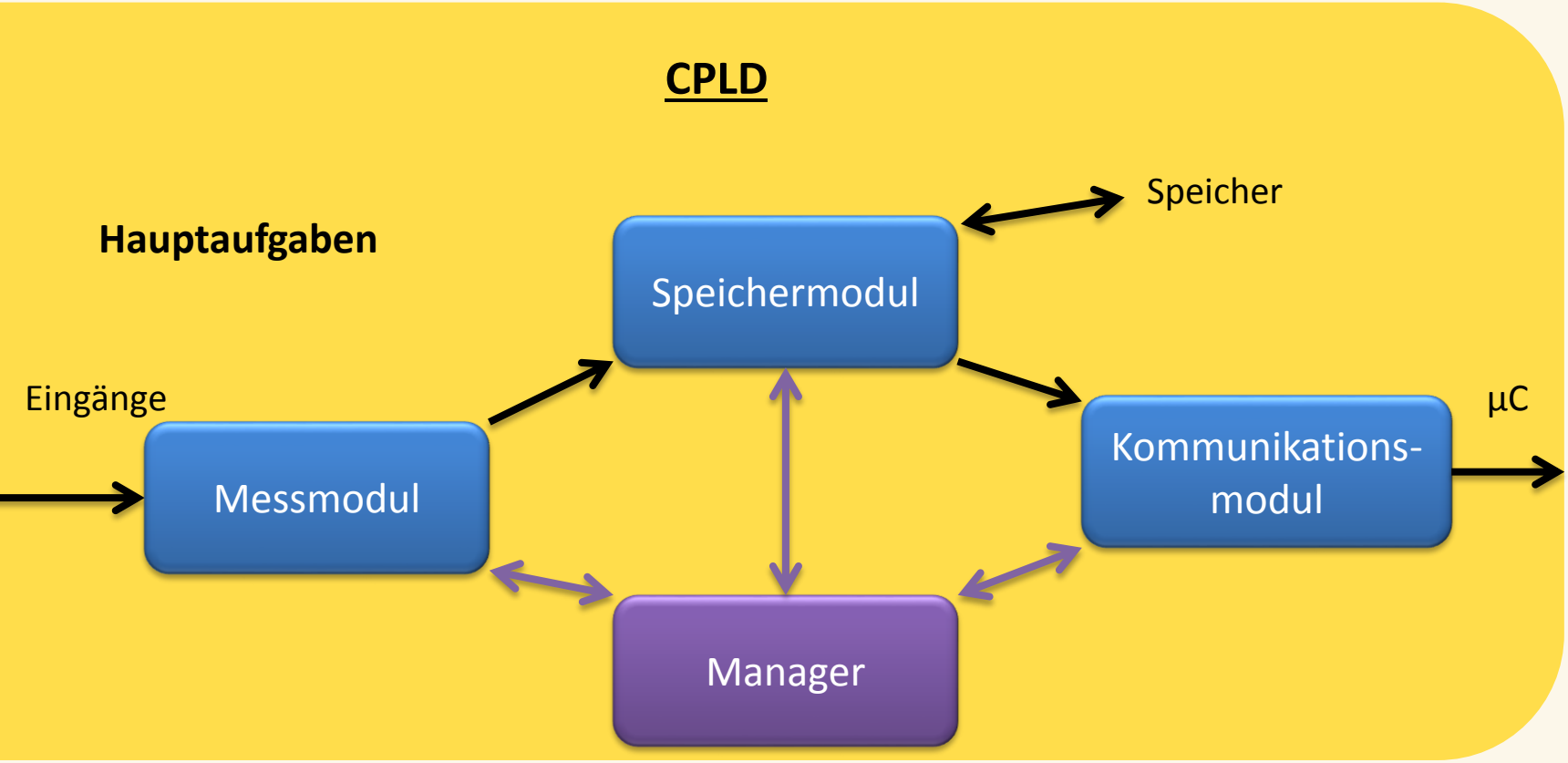
CPLD

Entwicklungsumgebung

- Hardwarebeschreibungssprache VHDL
- Simulation mit Modelsim von Mentor Graphics
- Schaltungssynthese mit Quartus von Altera
- Test einzelner Module auf einem Evaluationsboard



Logikanalyse für Alle



Logikanalyse für Alle

CPLD

Manager
- Eigentliche Programmsteuerlogik
- Auswertung der internen Statussignale
- Ansteuerung der internen Module

Logikanalyse für Alle

CPLD

Messung

- 8 digitale Eingänge
- Abtastung der Eingänge mit 6,25 Megasamples pro Sekunde
- Speicherung eines Datensatzes bei Signaländerung
- Zeitstempel
- Aufbau Datensatz:

Logikanalyse für Alle

CPLD

Speicher

- 2 x SRAM 256K x 16 (= 1 MegaByte); 12 ns Zugriffszeit
- Anzahl möglicher Messungen: 1 MB / 4 Byte = 262.144
- Speicherung eines Datensatzes benötigt 16 Takte
- Abtastrate: 100 MHz / 16 = 6,25 MHz

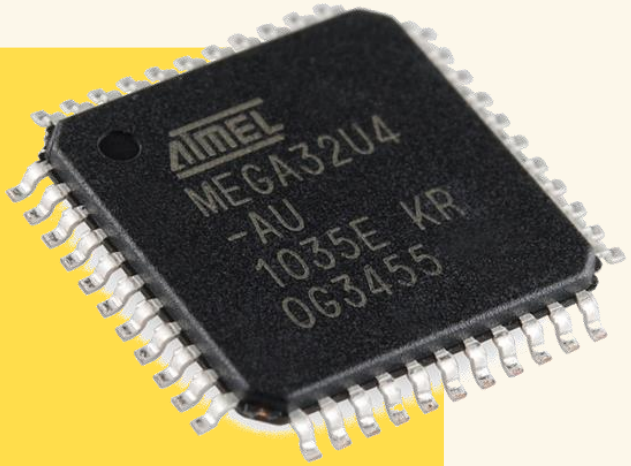
Logikanalyse für Alle

CPLD

Kommunikation
- Empfang von Steuerbefehlen des Mikrocontrollers
- Mikrocontroller kann Speicherinhalt und Status auslesen
- Schnittstelle zu µC: 4 bidirektionale Datenleitungen + 6 Steuerleitungen

Logikanalyse für Alle

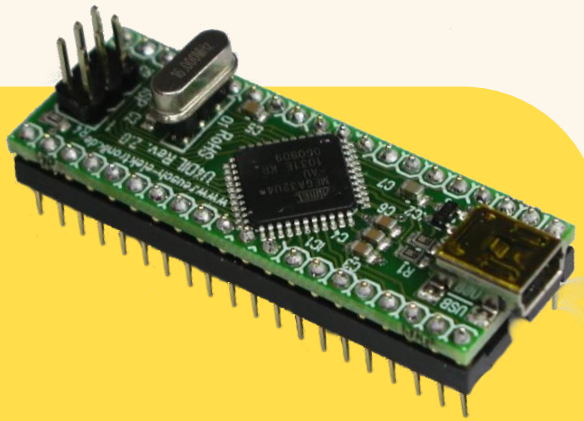
µC



Warum dieser µC?
Atmega32-U4 hat einen USB-2.0 Baustein fest integriert.
USB-Frameworks vorhanden

Aufgaben µC:
Kommunikation mit PC über USB (Device)
Aufspielen neuer µC-Firmware über USB (Bootloader)
Flashen des CPLD (Durchleiten der seriellen Signale vom STAPL-Player)
Steuerung des CPLDs über parallele Schnittstelle (Master)

Logikanalyse für Alle



µC

Entwicklungsumgebung: AVR-Studio von Atmel

USB-Framework (LUFA 130901):
Das Projekt hat es sich zur Aufgabe gemacht, ohne größere Kenntnisse über die Technik von USB, die Schnittstelle des ATmega verwenden zu können.
Die Bibliothek stellt verschiedene Device- und Host-Geräteklassen zur Verfügung.
Open-Source-Lizenz (MIT)

Logikanalyse für Alle

µC

Steuerung des CPLDs

Messung starten
Messung stoppen
Daten auslesen
Daten in VCD-Format auslesen
Reset von CPLD
Status von CPLD
Springe in µC-Bootloader
CPLD mit STAPL-Player flashen (JTAG)

Rueckmeldungen: 'r' run ; 's' stop ; 'f' RAM Speicher voll

Logikanalyse für Alle

µC

Daten in VCD-Format (Value Change Dump)

```
$date Jan 14, 2013 $end
$version USB -TPLE 1.0 $end
$comment 64 Bit Timestamp , 8Bit Data $end
$timescale 160ns $end
$scope module logic $end
$var wire 8 % data $end
$upscope $end
$enddefinitions $end

#0
b00000000 %
#2303
b00000010 %
#56843
b10000010 %
```

Logikanalyse für Alle

PC-Seite

PC-Software - Wofür?
- Steuern/Automatisieren von Messungen
- Automatisierung von Logik-, Protokoll- und Event-Analyse
- Verarbeitung und Darstellung der Messergebnisse/ Protokolldaten

Steuern/Automatisieren von Messungen - Ablauf einer Messung
- Konfiguration des Messgerätes mit den Messparametern (Messdauer, Samplerate, welche Kanäle, ...)
- Starten einer Messung
- Stoppen einer Messung
- Auslesen der Messdaten ("Dump")

Logikanalyse für Alle

PC-Seite

Ansteuerungsprotokoll der Hardware

- läuft über eine (virtuelle) serielle Schnittstelle [über USB com. device class]
- wurde von uns selbst festgelegt:
  * *Befehle* und *Statusmeldungen* sind ASCII-Tokens (einzelne Zeichen)
  * *Messdaten* können wahlweise als ASCII oder binär übertragen werden

- 3 Implementierungs-Level der Ansteuerung:
  * *einfach*: Ansteuerung durch ein Standard-Terminal (plattformübergreifend)
  * *mittel*: Implementierung eines eigenen Text User Interface (TUI) unter Linux (Befehle und Statusmeldungen werden ausgewertet und für den Benutzer aufbereitet, Automatisierung)
  * *schwer*: Integration in das sigrok-Projekt (Linux)

Logikanalyse für Alle

PC-Seite

GtKWave

- Anzeigeprogramm für den zeitlichen Verlauf von Signalen (Timingdiagramme/ Waveform)
- GTK+ basiert (GIMP Toolkit)
- für Unix/Linux, aber Portierungen für Win32 und Mac OSX
- unterstützt zahlreiche Dateiformate (LXT, LXT2, VZT, FST, GHW, ***VCD***)
  → *VCD ist weitverbreiteter Standard! (spezifiziert in IEEE-1364)*
- lizenziert unter der GNU GPL
- geschrieben in C

Logikanalyse für Alle

PC-Seite

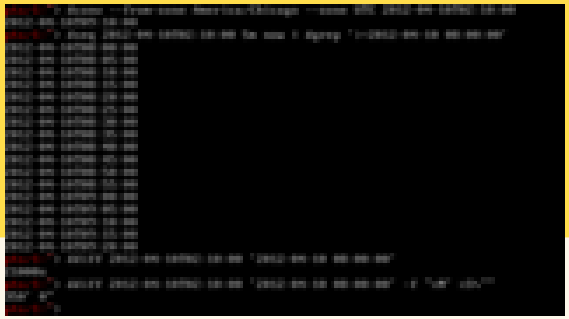
sigrok
- Software-Suite zur Signalanalyse
- portierbar, plattformübergreifend:
  Linux, Mac OS X, Windows, FreeBSD, OpenBSD,
  NetBSD (und unter x86, ARM, Sparc, PowerPC, ...)
- unterstützt verschiedene Gerätetypen verschiedener Hersteller
  Logikanalysatoren, Oszilloskope, Multimeter, Datenlogger, [...]
  → verallgemeinert: Messgeräte!
- "Free/Libre/Open-Source", lizenziert unter der GNU GPL
- "skriptisierbare" Protokoll-Dekodierung mit an Board!
- "stapelbare" Protokoll-Dekodierung

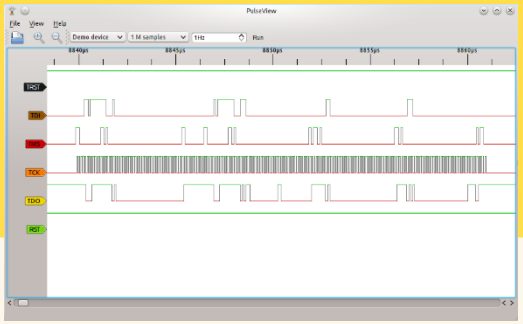
Logikanalyse für Alle

PC-Seite

sigrok

- unterstützt verschiedene Dateiformate (Binärdaten, ASCII, hex, CSV, gnuplot, VCD, ...)
- wiederverwendbarer Code mit verschiedenen Bibliotheken (libsigrok, libsigrokdecode, ...)
- Komponenten in C, C++ und Python mit GTK+ und Qt
- Verwendung mit verschiedenen Frontends/ GUIs (cli, PulseView)





Logikanalyse für Alle

PC-Seite

Firmware-Updates

- ***Firmware-Updates des Mikrocontrollers*** dank Bootloader per USB möglich!
- ***Firmware-Updates des CPLD*** dank eines JTAG-Adapters auf dem uC auch per USB möglich!
  * Jam Standard Test and Programming Language (STAPL) / JESD-71
  * JAM Stapl Player auf dem PC (JTAG-Daten)
  * Programmierdateien können direkt aus der Entwicklungsumgebung des CPLD exportiert werden
  * uC empfängt Bytes über die serielle Schnittstelle und setzt diese in Spannungspegel an den Portpins für JTAG um

Logikanalyse für Alle

Ausblick

- Re-Design der Platine um einen CPLD mit mehr Logikelementen zu integrieren.
- Einbindung in sigrok noch nicht ganz vollständig/ fehlerfrei
- GUI zur Bedienung des Logikanalysators unter Windows und Linux
- Erweiterung um Signalgruppengenerator-Funktion
- Weitere Triggermöglichkeiten und Abtastraten

Logikanalyse für Alle

Weitere Informationen

TRAC: https://io.informatik.fh-augsburg.de/trac/Logikanalysator
SVN: https://io.informatik.fh-augsburg.de/svn/Logikanalysator
WWW: https://io.informatik.fh-augsburg.de/projekte/Logikanalysator

Logikanalyse für Alle

Fragen?