



# FPGA für Anfänger spring(); break; 2019

Mark Hoffmann

# Inhaltsverzeichnis

- 1 Beispielanwendung
  - LED Lauflicht
- 2 Was ist ein FPGA?
  - Hardware Interna
- 3 Entwicklungsboard
  - Cyclone IV E FPGA
  - Development Kit

- 4 Entwicklungssoftware
  - Quartus Prime
  - ModelSim
  - VHDL & Verilog
- 5 Demonstration
  - Live!

# LED Lauflicht

## LED Lauflicht

Möglichst einfache  
Beispielanwendung

Lauflicht mit vier LEDs (rot)  
und einer Status LED (grün)



Box mit Lauflicht

# FPGA

**Ein Field Programmable Gate Array** ist eine im Feld, also vor Ort, beim Kunden, reprogrammierbare Logikgatter Anordnung



Xilinx XC2064 FPGA

# FPGA

## Logikgatter

Ein Logikgatter ist ein Bauelement, das mehrere digitale Eingangssignale zu einem digitalen Ausgangssignal umsetzt - eine Boole'sche Funktion realisiert

NICHT	
A	Y
1	0
0	1

UND		
A	B	Y
1	1	1
1	0	0
0	1	0
0	0	0

NAND		
A	B	Y
1	1	0
1	0	1
0	1	1
0	0	1

XNAND (Ungleichheit)		
A	B	Y
1	1	0
1	0	1
0	1	1
0	0	0

Oder		
A	B	Y
1	1	1
1	0	1
0	1	1
0	0	0

NOR		
A	B	Y
1	1	0
1	0	0
0	1	0
0	0	1

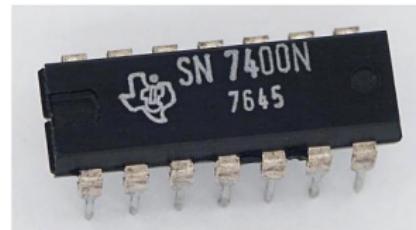
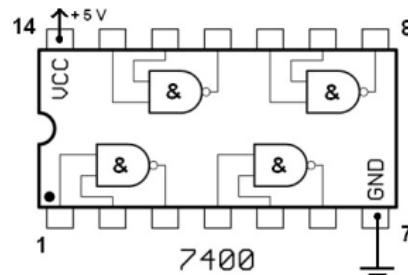
XNOR (Gleichheit)		
A	B	Y
1	1	1
1	0	0
0	1	0
0	0	1

Wahrheitstabellen für Gatter

# FPGA

## Entwicklung zum FPGA

- 7400er Serie
- PAL / GAL

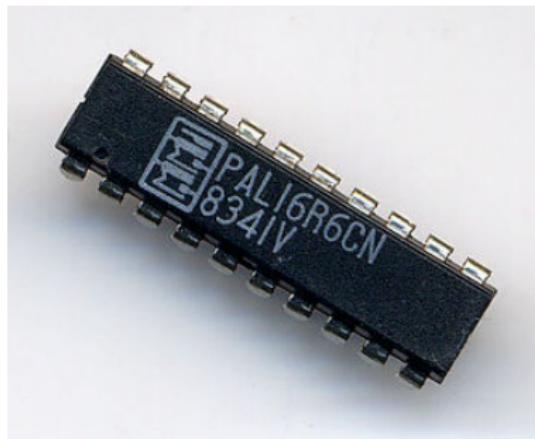


TI SN7400N

# FPGA

## Entwicklung zum FPGA

- 7400er Serie
- PAL / GAL
- CPLD

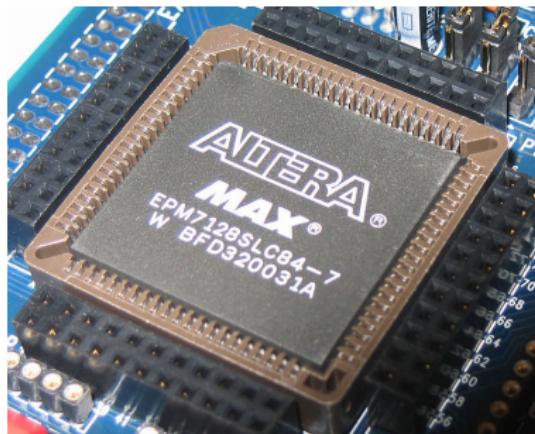


MMI 16R6 PAL

# FPGA

## Entwicklung zum FPGA

- 7400er Serie
- PAL / GAL
- CPLD
- FPGA



Altera MAX 7000 CPLD

# FPGA

## Entwicklung zum FPGA

- 7400er Serie
- PAL / GAL
- CPLD
- FPGA
- ASIC

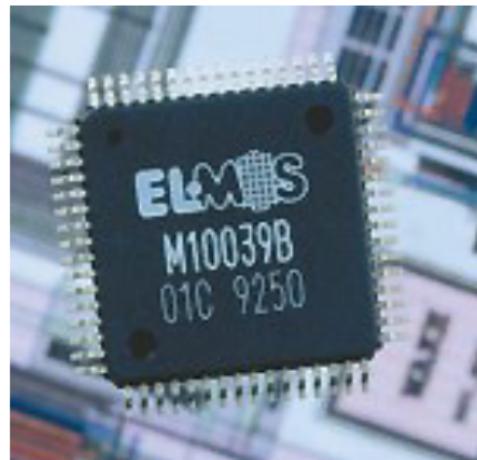


Altera Stratix IV FPGA

# FPGA

## Entwicklung zum FPGA

- 7400er Serie
- PAL / GAL
- CPLD
- FPGA
- ASIC

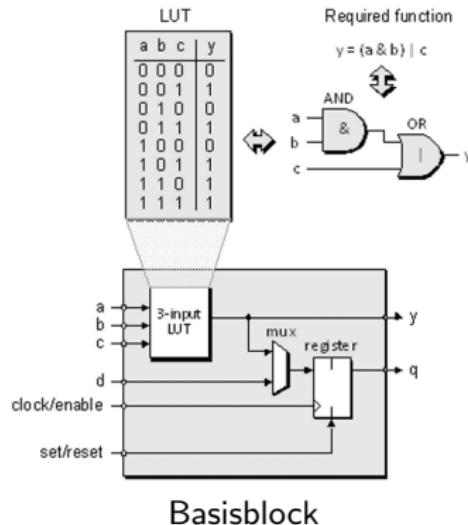


Elmos Asic

# FPGA

## FPGA Grundstruktur

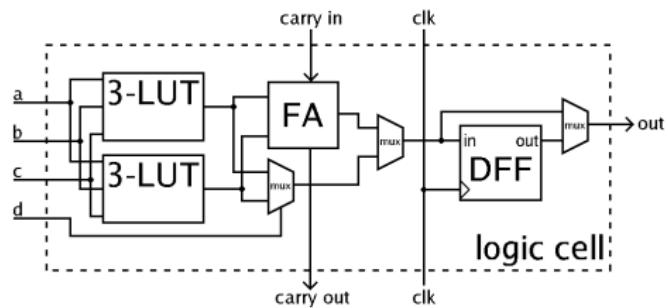
- Basisblock (LE)
- CLB



# FPGA

## FPGA Grundstruktur

- Basisblock (LE)
- CLB
- M9K Block RAM

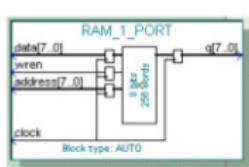


Configurable Logic Block mit Full Adder

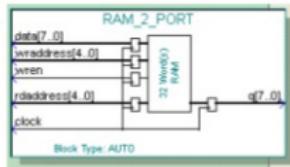
# FPGA

## FPGA Grundstruktur

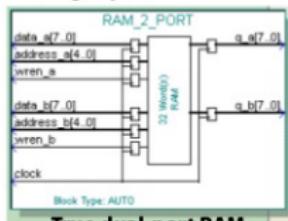
- Basisblock (LE)
- CLB
- M9K Block RAM
- DSP Blocks



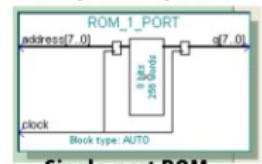
Single-port RAM



Simple dual-port RAM



True dual-port RAM



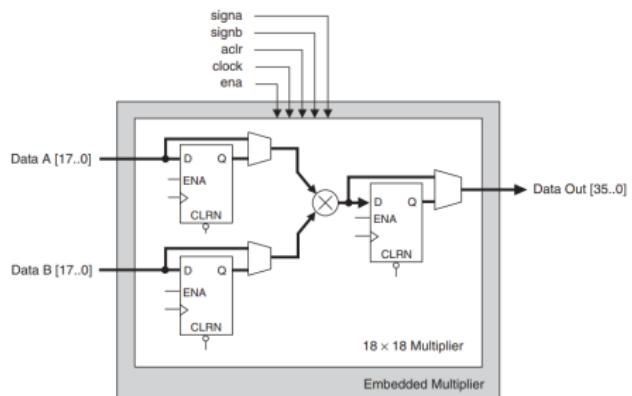
Single-port ROM

M9K Block RAM

# FPGA

## FPGA Grundstruktur

- Basisblock (LE)
- CLB
- M9K Block RAM
- DSP Blocks
- PLL

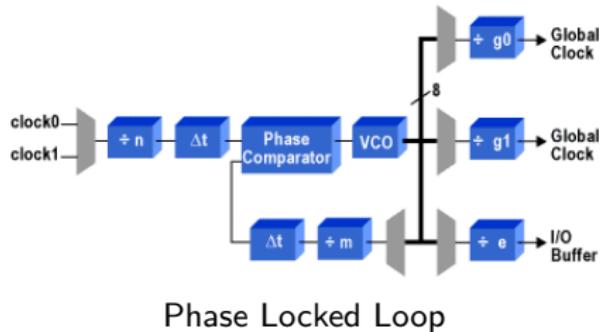


Embedded Multiplier (18 oder 9 Bit)

# FPGA

## FPGA Grundstruktur

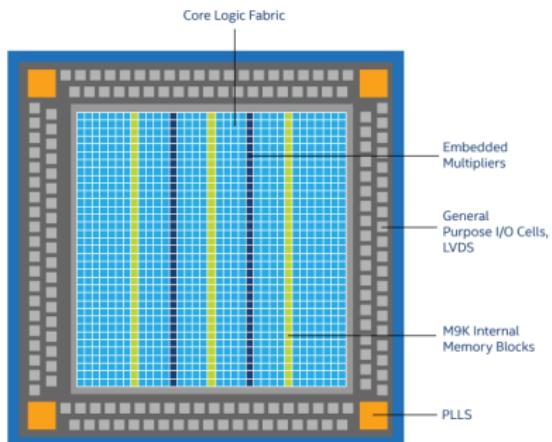
- Basisblock (LE)
- CLB
- M9K Block RAM
- DSP Blocks
- PLL
- Floor Plan



# FPGA

## FPGA Grundstruktur

- Basisblock (LE)
- CLB
- M9K Block RAM
- DSP Blocks
- PLL
- Floor Plan

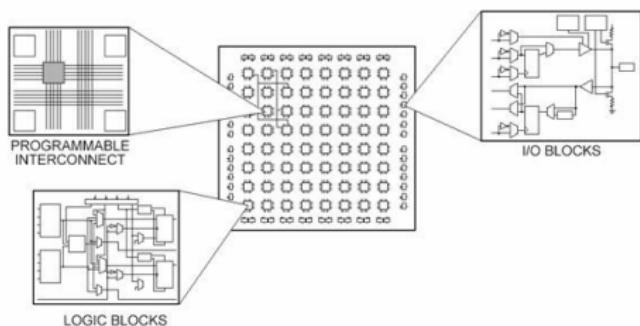


Floor Plan

# FPGA

## FPGA Grundstruktur

- Logikblöcke
- Verbindungsgeflecht

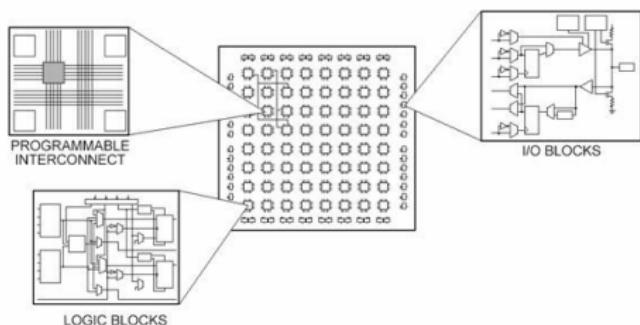


Funktionsblöcke

# FPGA

## FPGA Grundstruktur

- Logikblöcke
- Verbindungsgeflecht
- I/O-Blöcke

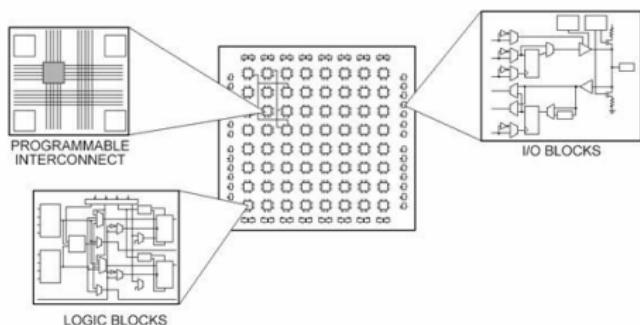


Funktionsblöcke

# FPGA

## FPGA Grundstruktur

- Logikblöcke
- Verbindungsgeflecht
- I/O-Blöcke



Funktionsblöcke

# Cyclone IV E FPGA

## Altera Cyclone IV E FPGA

- Cyclone IV E FPGA: EP4CE6E22C8N
- Anzahl der Logikzellen: 6272
- Anzahl der LABs: 392
- Embedded M9K Memory (Block RAM):  
270 Kbit (33 Kbyte)
- I/O-Spannung: 3,3 V
- I/O-Pin-Anzahl: 91
- Spannungsversorgung: 1,0 V bis 1,2 V
- Maximale Betriebsfrequenz: 200 MHz
- Gehäuse: QFP-144

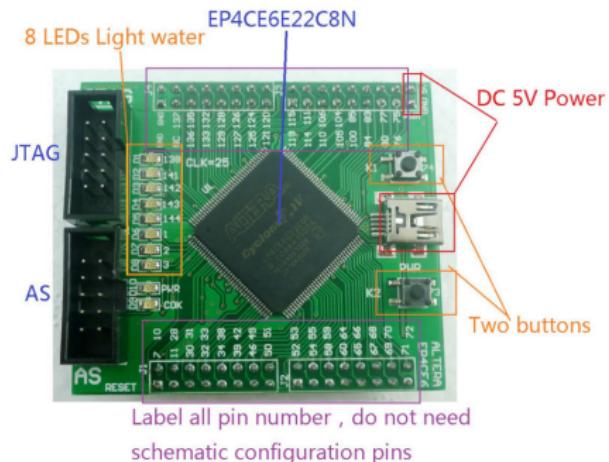


Altera Cyclone IV E

# Entwicklungsboard

## Entwicklungsboard

- FPGA: Cyclone IV E EP4CE6E22C8N (Altera)
- LEDs: 8

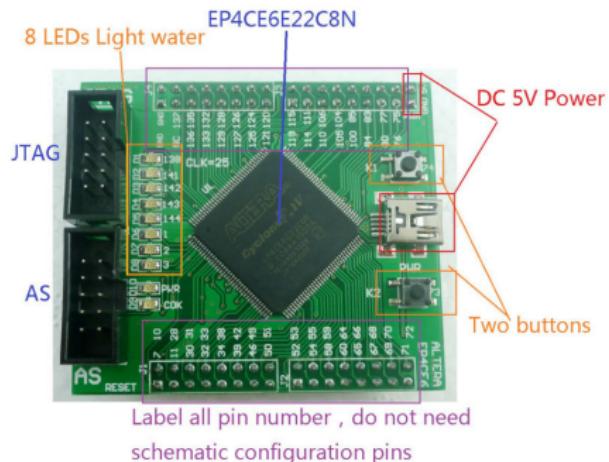


Vorderseite

# Entwicklungsboard

## Entwicklungsboard

- FPGA: Cyclone IV E EP4CE6E22C8N (Altera)
- LEDs: 8
- Push-Buttons: 2

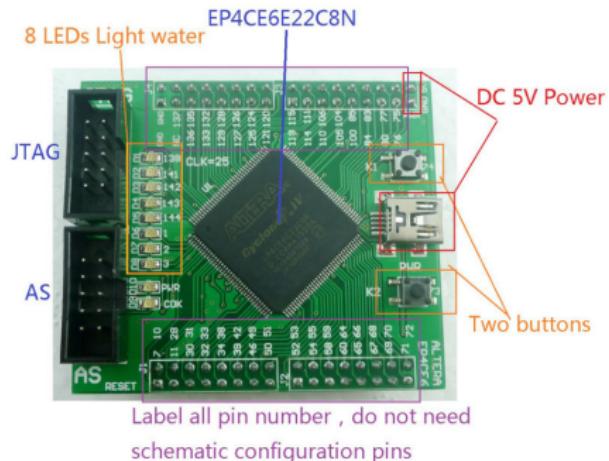


Vorderseite

# Entwicklungsboard

## Entwicklungsboard

- FPGA: Cyclone IV E EP4CE6E22C8N (Altera)
- LEDs: 8
- Push-Buttons: 2
- I/O-Pins: 91

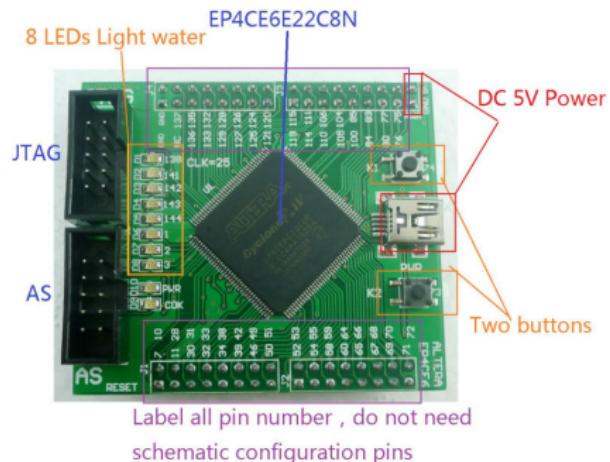


Vorderseite

# Entwicklungsboard

## Entwicklungsboard

- FPGA: Cyclone IV E EP4CE6E22C8N (Altera)
- LEDs: 8
- Push-Buttons: 2
- I/O-Pins: 91
- JTAG und AS Port

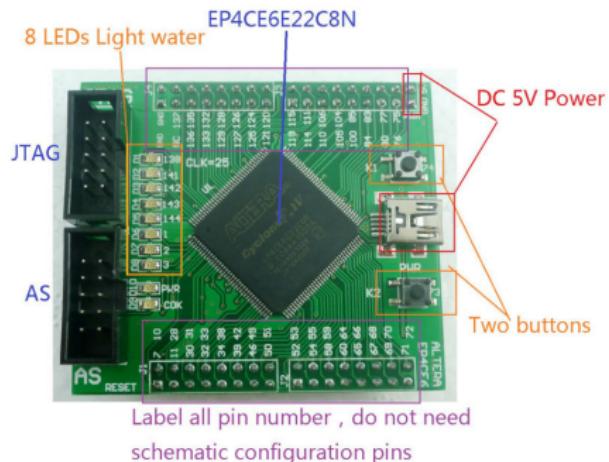


Vorderseite

# Entwicklungsboard

## Entwicklungsboard

- FPGA: Cyclone IV E EP4CE6E22C8N (Altera)
- LEDs: 8
- Push-Buttons: 2
- I/O-Pins: 91
- JTAG und AS Port
- Betrieb mit 5 V (USB)

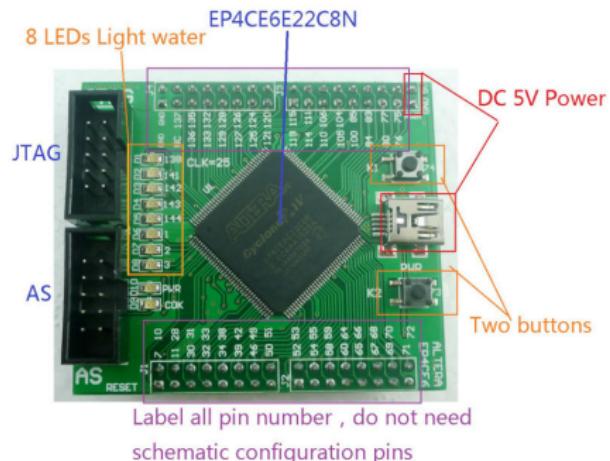


Vorderseite

# Entwicklungsboard

## Entwicklungsboard

- FPGA: Cyclone IV E EP4CE6E22C8N (Altera)
- LEDs: 8
- Push-Buttons: 2
- I/O-Pins: 91
- JTAG und AS Port
- Betrieb mit 5 V (USB)



Vorderseite

# Entwicklungsboard

## Entwicklungsboard

- Quarzoszillator: 25 MHz
- Serial-Flash-Memory:  
M25P40 4 MB

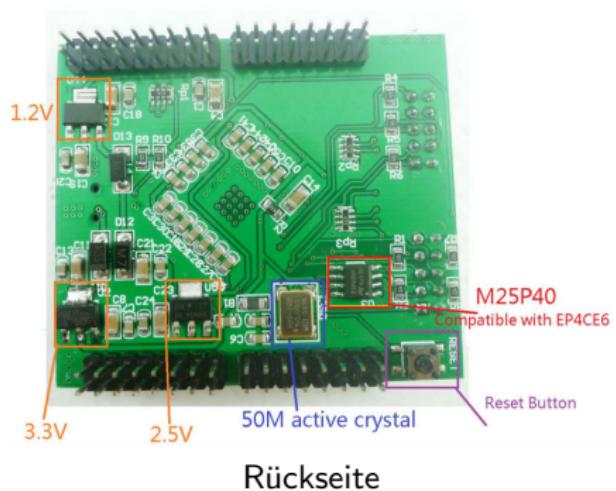


Rückseite

# Entwicklungsboard

## Entwicklungsboard

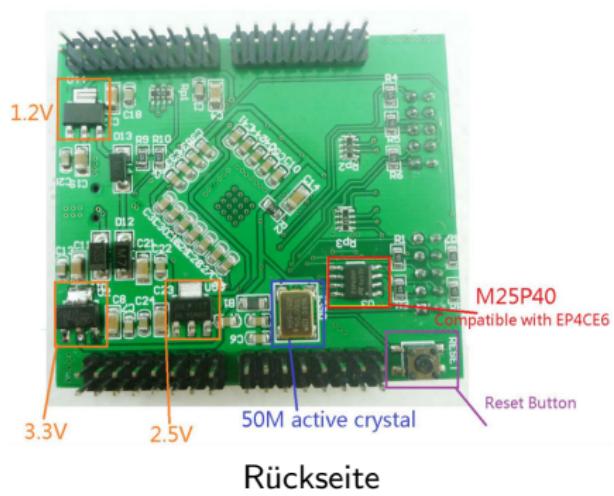
- Quarzoszillator: 25 MHz
- Serial-Flash-Memory:  
M25P40 4 MB
- Reset Button



# Entwicklungsboard

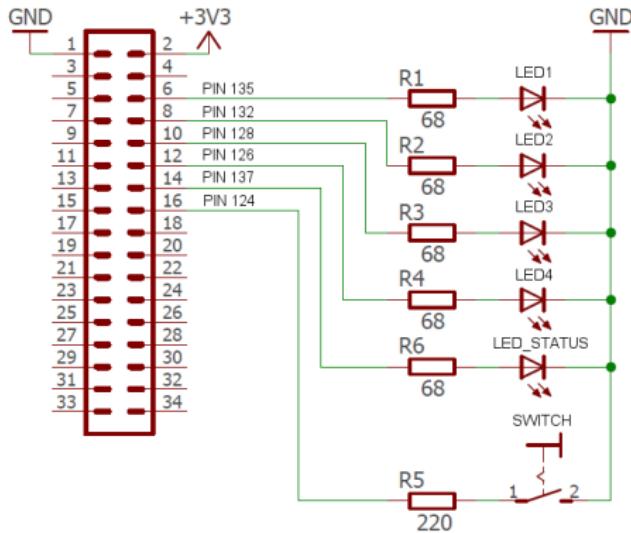
## Entwicklungsboard

- Quarzoszillator: 25 MHz
- Serial-Flash-Memory:  
M25P40 4 MB
- Reset Button



# Entwicklungsboard

## Schaltungsdiagramm Box - LEDs und Schalter



# Entwicklungsboard

## Entwicklungsboard

- JTAG-Adapter für Flashen des Bitstreams

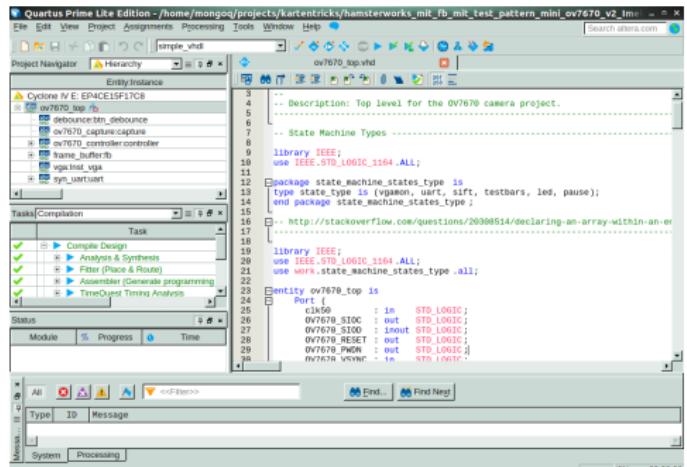


JTAG-Adapter

# Quartus Prime

## Quartus Prime

- Eingabe von VHDL/Verilog Code
- Synthese (Kompilieren)  
Vorgang mit Fehlermeldungen und Warnungen



The screenshot shows the Quartus Prime Lite Edition interface. The top menu bar includes File, Edit, View, Project, Assignments, Processing, Tools, Window, Help, and a search bar. The main window has tabs for Project Navigator and Hierarchy, with the Hierarchy tab selected. A tree view shows a project structure under 'Cyclone IV E: EP4CE15F17C8'. The code editor displays VHDL code for an 'ov7670\_top' entity. The code includes declarations for various components like 'frame\_buffer\_fb', 'vga\_inst', and 'sys\_uartuart'. It also includes a state machine declaration and an entity definition for 'ov7670\_top'. The status bar at the bottom shows '0% 00:00:00'.

```
entity ov7670_top is
    port(
        cik50 : in  STD_LOGIC;
        OV7670_SDIO : out STD_LOGIC;
        OV7670_SINR : inout STD_LOGIC;
        OV7670_RESET : out STD_LOGIC;
        OV7670_PWDN : out STD_LOGIC;
        OV7670_VSND : in  STD_LOGIC);
end entity;

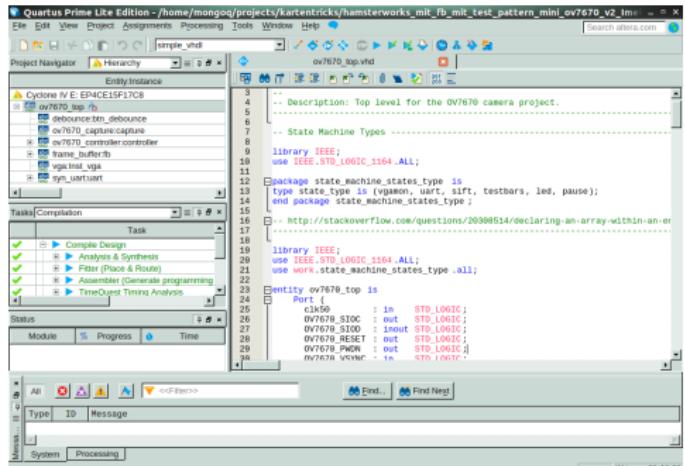
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD.TEXTIO.all;
use work.state_machine_states_type.all;
library IEEE;
use IEEE.STD.TEXTIO.all;
use work.state_machine_states_type.all;
entity ov7670_top is
    port(
        cik50 : in  STD_LOGIC;
        OV7670_SDIO : out STD_LOGIC;
        OV7670_SINR : inout STD_LOGIC;
        OV7670_RESET : out STD_LOGIC;
        OV7670_PWDN : out STD_LOGIC;
        OV7670_VSND : in  STD_LOGIC);
end entity;
```

Quartus Prime IDE

# Quartus Prime

## Quartus Prime

- Eingabe von VHDL/Verilog Code
  - Synthese (Kompilieren)  
Vorgang mit Fehlermeldungen und Warnungen
  - Verzweigung zu Unterprogrammen

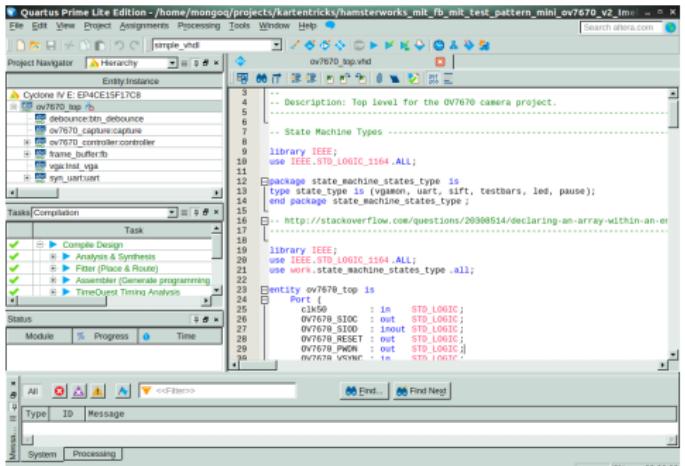


Quartus Prime IDE

# Quartus Prime

## Quartus Prime

- Eingabe von VHDL/Verilog Code
- Synthese (Kompilieren)  
Vorgang mit Fehlermeldungen und Warnungen
- Verzweigung zu Unterprogrammen



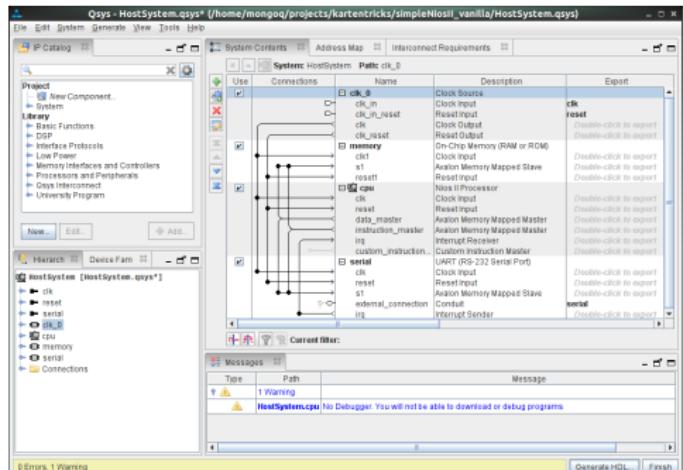
```
entity ov7670_top is
    port(
        cik50 : in  STD_LOGIC;
        OV7670_SDIO : out STD_LOGIC;
        OV7670_SINR : inout STD_LOGIC;
        OV7670_RESET : out STD_LOGIC;
        OV7670_PWDN : out STD_LOGIC;
        mV7670_VSND : in  STD_LOGIC);
end entity;
```

Quartus Prime IDE

# QSYS

## QSYS

- Instanziieren von:  
Softcores (NIOS II), RAM,  
Schnittstellen (JTAG / UART)
- Grafisches Verbinden der  
Funktionsblöcke



QSYS

# QSYS

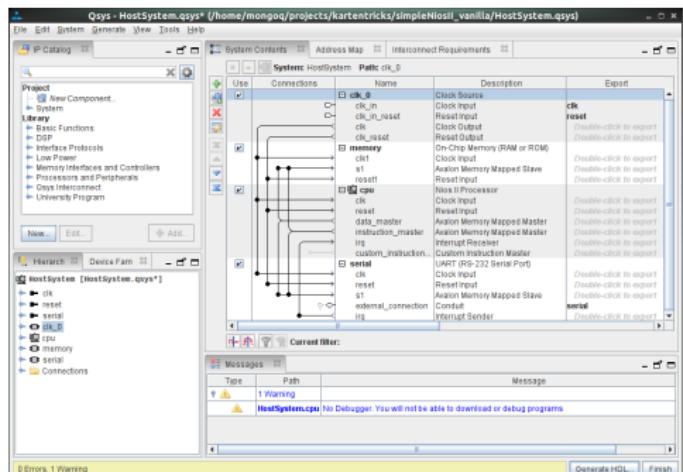
## QSYS

- Instanziieren von:

Softcores (NIOS II), RAM,  
Schnittstellen (JTAG / UART)

- Grafisches Verbinden der  
Funktionsblöcke

- Eclipse basierte  
Programmierumgebung

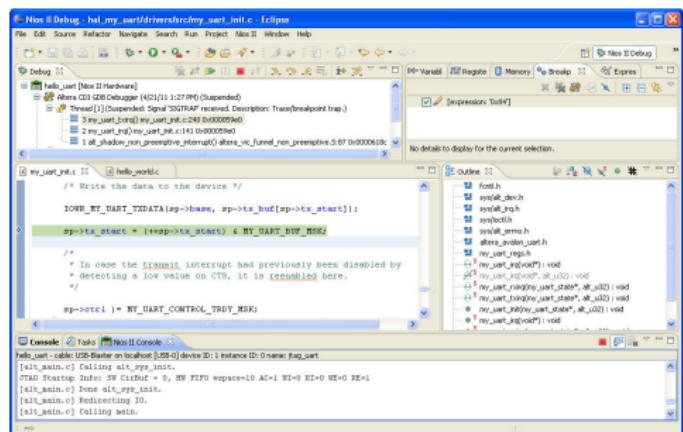


QSYS

# QSYS

# QSYS

- Instanziieren von:  
Softcores (NIOS II), RAM,  
Schnittstellen (JTAG / UART)
- Grafisches Verbinden der  
Funktionsblöcke
- Eclipse basierte  
Programmierumgebung
- Beispielsystem

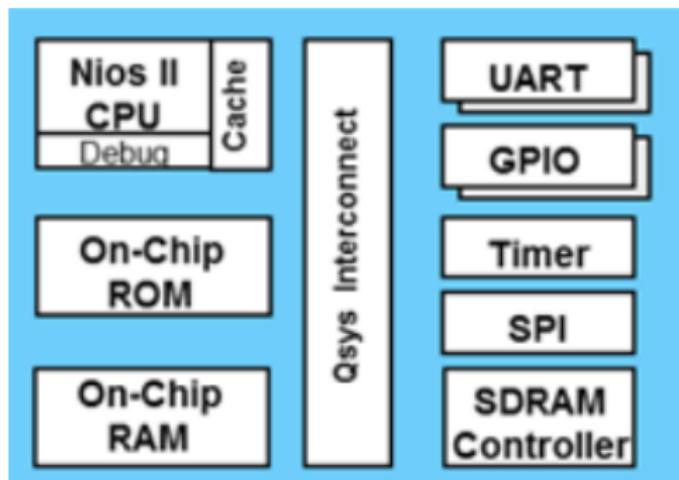


Nios II Embedded Design Suite (Eclipse)

# QSYS

## QSYS

- Instanziieren von:  
Softcores (NIOS II), RAM,  
Schnittstellen (JTAG / UART)
- Grafisches Verbinden der  
Funktionsblöcke
- Eclipse basierte  
Programmierumgebung
- Beispielsystem

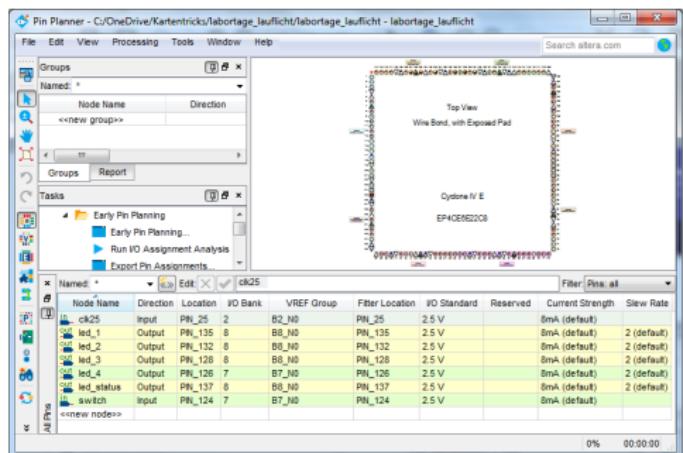


Beispielsystem (SoC - System-on-Chip)

# Pin Planner

## Pin Planner

- Zuordnung von logischen Ein- und Ausgängen zu physikalisch vorhandenen Pins

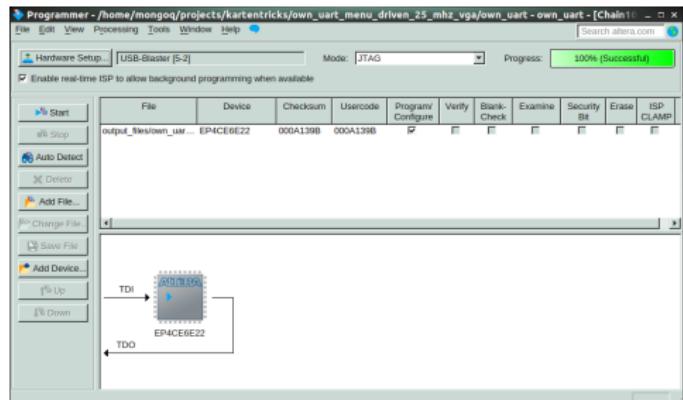


Pin Planner

# JTAG Programmer

## JTAG Programmer

- Flashen der FPGA Konfigurationsdatei - des Bitstreams
- Wahlweise in FPGA (flüchtig) oder Serial-Flash (permanent)

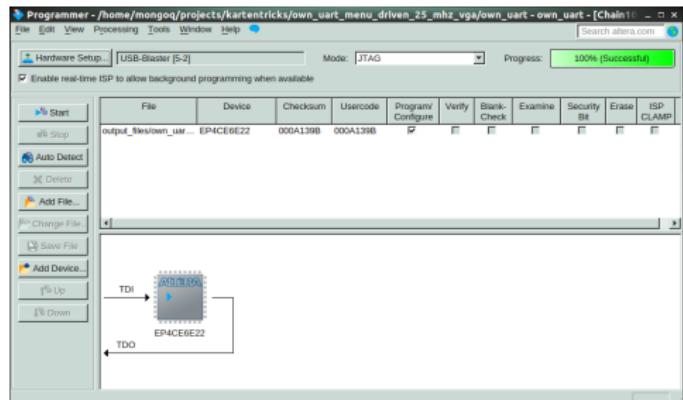


JTAG Programmer

# JTAG Programmer

## JTAG Programmer

- Flashen der FPGA Konfigurationsdatei - des Bitstreams
- Wahlweise in FPGA (flüchtig) oder Serial-Flash (permanent)

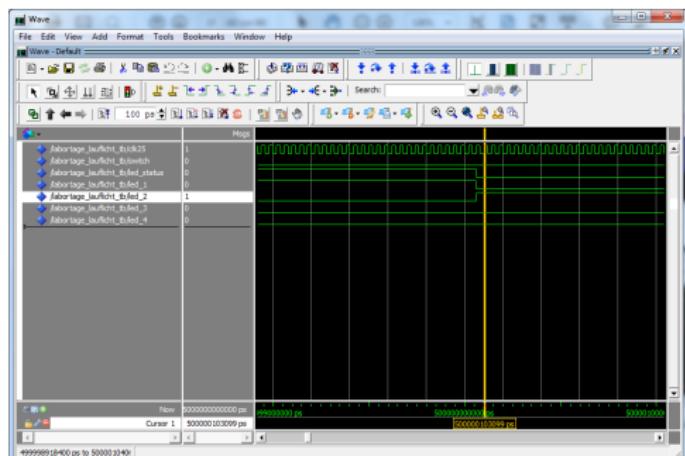


JTAG Programmer

# ModelSim

## ModelSim

- Simulation eines FPGA Programms
- Nutzung von Testbenches mit Stimuli

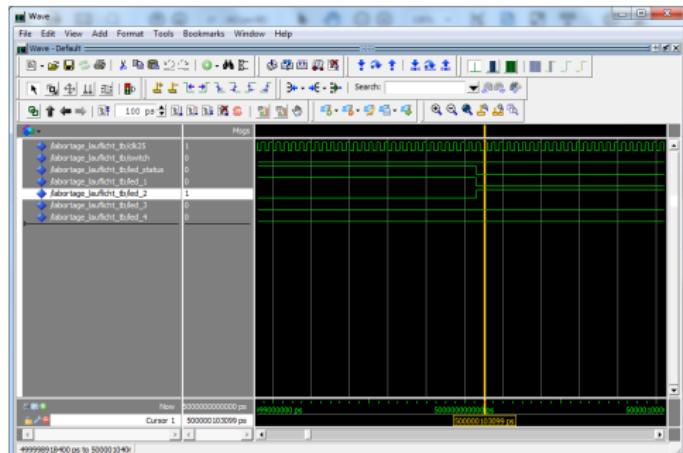


ModelSim

# ModelSim

## ModelSim

- Simulation eines FPGA Programms
- Nutzung von Testbenches mit Stimuli
- Ausgabe als Pegelwechsel-Diagramm oder Text

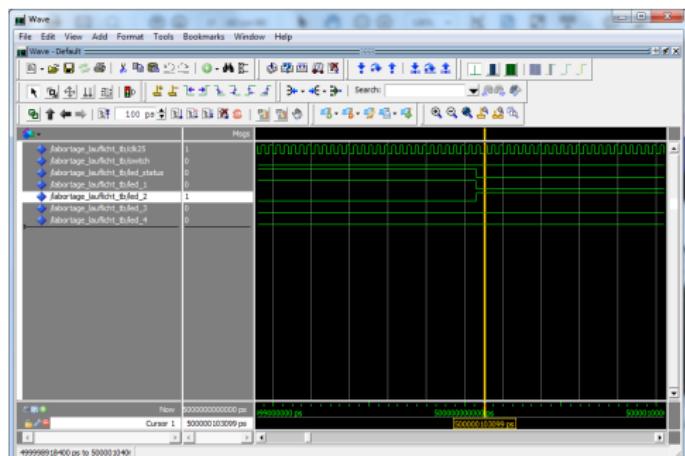


ModelSim

# ModelSim

## ModelSim

- Simulation eines FPGA Programms
- Nutzung von Testbenches mit Stimuli
- Ausgabe als Pegelwechsel-Diagramm oder Text
- Simulation erfolgt nicht in Echtzeit

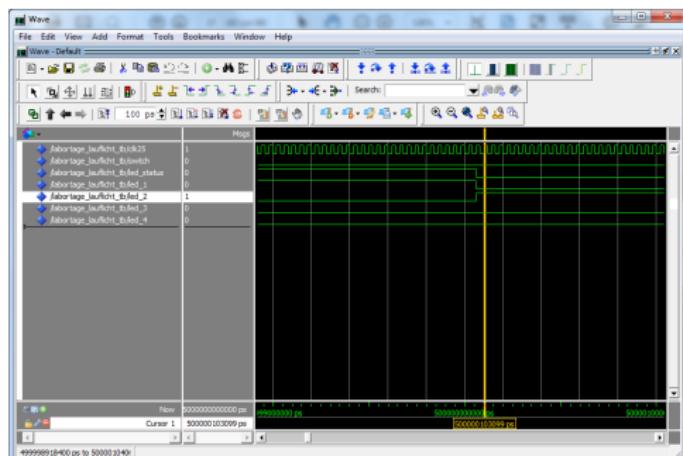


ModelSim

# ModelSim

## ModelSim

- Simulation eines FPGA Programms
- Nutzung von Testbenches mit Stimuli
- Ausgabe als Pegelwechsel-Diagramm oder Text
- Simulation erfolgt nicht in Echtzeit



ModelSim

# Programmierung

## VHDL (Europa)

Hardwarebeschreibungssprache  
Angelehnt an Ada

```
2 process  ({S0,S1},A,B,C,D)
3 begin
4     case {S0,S1}, is
5         when "00" => Y <= A;
6         when "01" => Y <= B;
7         when "10" => Y <= C;
8         when "11" => Y <= D;
9         when others => Y <= A;
10    end case;
11 end process;
```

Beispiel Code

## Verilog (USA)

Hardwarebeschreibungssprache  
Angelehnt an C

```
2
3 always @({S0,S1}, A, B, C, D)
4     case ({S0,S1})
5         2'b00: Y = A;
6         2'b01: Y = B;
7         2'b10: Y = C;
8         2'b11: Y = D;
9     endcase
10
```

Beispiel Code

## Demonstration - Live!

### Demonstration von Beispielcode

- Libraries
- Entity und Architecture
- Ein- und Ausgänge
- LED Auswahl
- Counter (zeitabhängig)
- Case Statement (Finite State Machine)
- Pin Planner
- JTAG Download
- ModelSim Simulation



Box mit Lauflicht

# Fragen?

Vielen Dank für Eure Aufmerksamkeit!

Hat vielleicht noch jemand Fragen?

