OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

Computer Architektur

Sommersemester 2025

Projektzielsetzung

Version 1.0

Teammitglieder:

Fabian Becker _____

Jendrik Jürgens _____

Nicolas Koch _____

Franz Krempl _____

Daniel Sowada _____

Michael Specht _____

Professor:
Dr. Daniel Münch

Abgabedatum:
30.04.2025 11:30 Uhr

26. April 2025

Inhaltsverzeichnis

Abbildungsverzeichnis **2**

Tabellenverzeichnis **3**

1 Projektzielsetzung **4**

2 PmodCLP **7**

2.1 Ansatz Custom IP 8

2.2 Registermapping 9

2.2.1 I/Os 9

2.2.2 Registerbereich 9

3 Sensor **14**

Abbildungsverzeichnis

1.1 Systemblockbild 5

2.1 Startup Sequence 8

Tabellenverzeichnis

2.1 PmodCLP Überblick I/O 9
2.2 PmodCLP Controller Register Space Overview 9
2.3 General/Global Control and Status Register (GCSR) 10
2.4 Global Interrupt Enable Register (GIER) 10
2.5 IP Interrupt Enable Register (IPIER) 11
2.6 IP Interrupt Status Register (IPISR) 11
2.7 ID Register (IDR) 11
2.8 Version Register (VERR) 11
2.9 Special Control and Status Register (SCSR0) 12
2.10 Character Data Register (CDR) 12
2.11 Format and Mode Register (FMR) 13
2.12 Sensor Data Register (SDR) 13

1. Projektzielsetzung

Projekttitel:
Integriertes Demosystem für Sensor- und Displayansteuerung auf FPGA-Basis
Teammitglieder:
Fabian Becker, Jendrik Jürgens, Nicolas Koch, Franz Krempl, Daniel Sowada, Michael Specht
Projektbeschreibung:
Ziel des Projekts ist die Entwicklung und Integration eines funktionsfähigen Demosystems, das zwei getrennte Komponenten – den Sonarsensor (PMOD MAXSONAR) und das LCD-Display (PMOD CLP) – auf einer gemeinsamen FPGA-Plattform (Digilent Arty A7-100) vereint. Die Messwerte des Sensors sollen in Echtzeit auf dem Display ausgegeben werden. Dazu werden eigene IP-Cores für beide Komponenten entworfen, implementiert, getestet und in die Hardwareplattform integriert.

Geplante Arbeitspakete und Zuständigkeiten:

1. Systemintegration bestehender Demosysteme
   - Integration der Software- und Hardware-Demoprojekte zu einem Gesamtsystem (HW & SW)

     Zuständig: Alle Teammitglieder
2. Entwicklung Custom IP für Sensor (UART / MAXSONAR)
   - Konzept (Blockdiagramm, Registermapping) auf Basis der `tut_int` Vorlage
   - Reduzierte Funktionalität orientiert an Xilinx UART Lite IP

     Zuständig: Fabian Becker, Nicolas Koch
3. Entwicklung Custom IP für Display (PMOD CLP, Nibble-Mode)
   - Umsetzung von Timinganforderungen in Hardware
   - Anzeige von Daten auf LCD über FSM und spezifizierte Steuerregister

     Zuständig: Jendrik Jürgens, Michael Specht
4. Erstellung und Test der IP-Testbenches (Core und AXI)
   - Entwicklung mit Fokus auf Polling (Interrupt optional bei Zeitreserve)
   - Simulation und Validierung des Verhaltens

     Zuständig: Alle Teammitglieder
5. Treibersoftware
   - Schreiben von Treibern für die beiden IPs unter Verwendung der SW-Templates von `tut_int`
   - SW-basierte Initialisierung und Datentransfer

Zuständig: Franz Krempl, Daniel Sowada

6. Integration in vollständiges Demosystem
   - Zusammenführung aller Komponenten zu einem lauffähigen System
   - Validierung auf der realen Hardwareplattform

   Zuständig: Alle Teammitglieder

Systemblockbild:

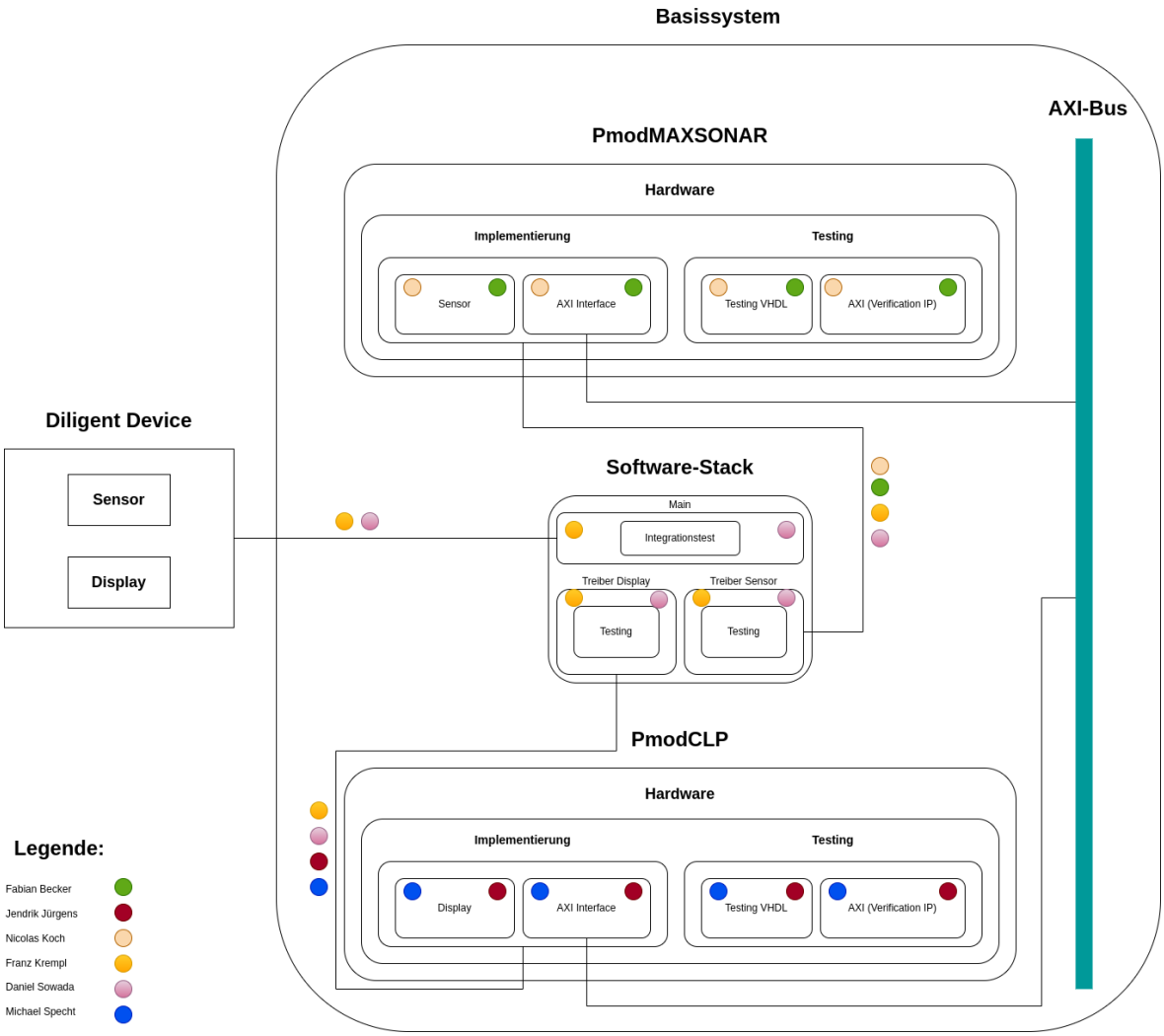

Abbildung 1.1: Systemblockbild

- **Exakte Timings für das Display (PmodCLP) in VHDL:**
  Die Ansteuerung im Nibble-Mode erfordert die präzise Umsetzung aller Timingbedingungen (Setup, Hold, Enable) in einer FSM, da keine automatische Verzögerung durch die CPU gegeben ist.

- **Entwicklung AXI4-Lite-kompatibler IP-Cores:**
  Für Sensor und Display werden eigenständige IPs mit klar strukturiertem Registermapping und AXI-Anbindung erstellt, basierend auf einer gemeinsamen IP-Vorlage.

- **Zuverlässiger Datenfluss zwischen Sensor, CPU und Display:**
  Die Software muss synchronisiert mit den IPs arbeiten, um Sensordaten korrekt auszulesen und anzuzeigen – inklusive Fehlerbehandlung und Statusabfrage.

- **Simulation und Verifikation:**
  Funktion und Schnittstellen werden über VHDL-Testbenches (Core + AXI) geprüft, um Designfehler frühzeitig zu erkennen.

Ziel:
Ein lauffähiges Demosystem mit eigenentwickelten, erweiterbaren IP-Cores, das Messwerte des Sonar Sensors auf einem LCD-Display ausgibt – mit strukturierter Dokumentation, Tests und einer funktionalen Ergebnisvorführung.

2. PmodCLP

Der PmodCLP besteht aus einem Samsung KS0066 LCD Controller und einem Sunlike LCD Panel, worüber Informationen dargestellt werden können lcd-desc. Es ist möglich 32 Positionen auf dem 16x2 LCD Panel zu nutzen. Pro Position werden die Zeichen dabei mit einer Auflösung von 5x8 angezeigt.

Das System besteht im Wesentlichen aus drei Komponenten. Der character-generator ROM (CGROM) hält 192 vordefinierte Zeichen, darunter 93 ASCII Charaktere. Anschaulich gesehen können die Zeichen über eine matrixartige Struktur indiziert werden, welche im Datenblatt festgelegt ist. Neben den nicht-volatilen Daten im CGROM ist es möglich bis zu 8 eigene Zeichen volatil im character-generator RAM (CGRAM) zu halten. Um nun Zeichen aus diesen beiden Repositories auf dem Panel anzeigen zu können, gibt es den data RAM (DDRAM). Hier können bis zu 80 Zeichencodes gespeichert werden. Er fungiert als Indexspeicher für Daten innerhalb des CGROM oder CGRAM. Wird ein Index aus der matrixartigen Struktur in den DDRAM geladen, erscheint das entsprechende Zeichen auf dem Display.

Das Display selbst verfügt über 2 Zeilen á 16 Positionen. Insgesamt stehen jedoch nicht 32 Speicherplätze zur Verfügung, sondern 39, um beispielsweise Scrolling zu verwenden.

Wichtige Schnittstellen des Samsung KS0066 LCD Controller sind

- *DB4-DB7*: Datenbits im Nibble-Mode zur Codierung von Befehlen/Zeichen
- *RS (Register Select)*: High für Daten, Low für Instruktionen
- *RW (Read/Write)*: High = Read, Low = Write
- *E (Enable)*: High für Read, Falling Edge für Write

Um diese nutzen zu können wird folgendes Mapping auf dem FPGA hinterlegt:

```
set_property -dict {PACKAGE_PIN D13 IOSTANDARD LVCMOS33}[get_ports{clp_db_tri_io[4]}];
    #db04
set_property -dict {PACKAGE_PIN B18 IOSTANDARD LVCMOS33}[get_ports{clp_db_tri_io[5]}];
    #db05
set_property -dict {PACKAGE_PIN A18 IOSTANDARD LVCMOS33}[get_ports{clp_db_tri_io[6]}];
    #db06
set_property -dict {PACKAGE_PIN K16 IOSTANDARD LVCMOS33}[get_ports{clp_db_tri_io[7]}];
    #db07
set_property -dict {PACKAGE_PIN E2 IOSTANDARD LVCMOS33}[get_ports{clp_cb_tri_o[0]}];
    #lcd_rs
set_property -dict {PACKAGE_PIN D2 IOSTANDARD LVCMOS33}[get_ports{clp_cb_tri_o[1]}];
    #lcd_rw
set_property -dict {PACKAGE_PIN H2 IOSTANDARD LVCMOS33}[get_ports{clp_cb_tri_o[2]}];
    #lcd_e
```

Listing 2.1: Pin-Zuordnung im Constraints-File

Die Zuordung in Aufzählung 2.1 greift auf zwei Header des PmodCLP-Boards zurück. Die Daten-Bits db04-db07 sind an die untere Hälfte des Headers J1 gebunden. Die Steuersignale Register Select, Read/Write und Enable hingegen an Header J2.

2.1. Ansatz Custom IP

Im ersten Schritt soll die IP funktional fertiggestellt werden. Sie soll dabei mittels Polling eingesetzt werden. Sobald die Funktionalität des Gesamtsystems vollumfänglich gegeben ist, wird anstatt Polling über Interrupts kommuniziert.

Das Projektteam hat sich auf folgenden Entwurf geeinigt:

- *Submodul 1*: LCD-Controller (FSM)
  Mittels einer FSM wird aus den Registern, welche von Microprozessor gesetzt werden können, der Befehl interpretiert.
- *Submodul 2*: Timing Controller
  Dies kann bspw. mit einem Zähler, der als Timer in Abhängigkeit vom Systemtakt fungiert, umgesetzt werden. Diverse Statusflags sollen den aktuellen Stand zeigen.

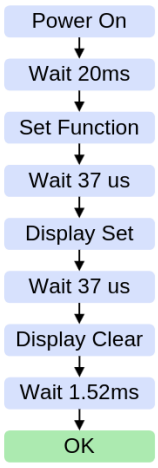

Abbildung 2.1: Startup Sequence

  Graphik 2.1 zeigt die Initialisierung des Displays und die damit verbundenen Timing-Anforderungen.

- *Submodul 3*: Character Transfomer
  Die vom Sensor erhaltenen Werte müssen in entsprechende Werte, die auf dem Display dargestellt werden können, umgewandelt werden. Eine lookup-table soll hier Abhilfe schaffen.
- *Submodul 4*: Memory Mapping
  Dieses Submodul soll eines standardisierte und zuverlässige DDRAM-Adressierung garantieren. Dabei muss das Display-Layout (16x2) beachtet werden. Mögliche Erweiterungen, wie bspw. Scrolling, sollen bei der Implementierung bereits stets bedacht werden.
- *Submodul 5*: LCD-Communication Interface
  Hier werden die Daten in die entsprechenden Register geschrieben, um die Werte anschließend auf dem Display darzustellen.

- *Submodul 6*: AXI Slave Interface
  Nachdem die Zuverlässigkeit der IP mittels Tests sichergestellt wurde, soll die IP an den internen Systembus (AXI) angebunden werden.

Das Registermapping wurde in Anlehnung an *at_ doc.pdf* aus *02b_ tut_ vhdl_ v03* erstellt.

2.2. Registermapping

2.2.1. I/Os

Signal Name	**I/O**	**Initial State**	**Description**
ap_clk(s00_axi_aclk)	I	NA	AXI Clock
ap_rst_n (s00_axi_aresetn)	I	NA	AXI Reset, active-Low
s_axi_control* (s00_axi*)	NA	NA	AXI4-Lite Slave Interface signals
interrupt	I	0x0	Indicates that the condition for an interrupt has occurred. (new sensor value available) 0 = No interrupt has occurred 1 = Interrupt has occurred
sensor_val_in	I	0xFF	Input value from PmodMAXSONAR
db4_7_out	O	0xFF	4 data bits, necessary in nibble mode.
register_select_out	O	0x1	Register Select: High for Data Transfer, Low for Instruction Transfer
read_write_out	O	0x1	Read/Write signal: High for Read mode, Low for Write mode
read_write_enable_out	O	0x1	Read/Write Enable: High for Read, falling edge writes data

Tabelle 2.1: PmodCLP Überblick I/O

2.2.2. Registerbereich

Address Offset	**Register Name**	**Description**
0x00	GCSR	General/Global Control and Status Register
0x04	GIER	Global Interrupt Enable Register
0x08	IPIER	IP Interrupt Enable Register
0x0C	IPISR	IP Interrupt Status Register
0x10	IDR	ID Register
0x14	VERR	Version Register
0x18	SCSR0	Special Control and Status Register
0x1C	CDR	Character Data Register

Tabelle 2.2: PmodCLP Controller Register Space Overview

Bit	Name	Access Type	Reset Value	Description
0x00 GCSR - General/Global Control and Status Register				
0	ap_start	R/W	0	Asserted when the kernel can start processing data. Cleared on handshake with ap_done being asserted.
1	ap_done	R	0	Asserted when the kernel has completed initialization operation. Cleared on read.
2	ap_idle	R	0	Asserted when the kernel is idle.
3	*reserved (ap_ready)*	R	0	Asserted by the kernel when it is ready to accept new data (used only by AP_CTRL_CHAIN)
4	*reserved (ap_continue)*	R/W	0	Asserted by the XRT to allow kernel keep running (used only by AP_CTRL_CHAIN)
5:6	reserved			
7	auto_restart	R/W	0	Used to enable automatic kernel restart. This bit determines whether the display reloads the last sensor value and continues running or clears the display.
8	lcd_initialized	R	0	Indicates the LCD has been properly initialized and is ready for commands.
9	display_busy	R	0	Indicates the display is currently processing a command.
10	error_flag	R	0	Indicates an error occurred during last operation.
31:11	reserved			

Tabelle 2.3: General/Global Control and Status Register (GCSR)

Bit	Name	Access Type	Reset Value	Description
0x04 GIER - Global Interrupt Enable Register				
0	gie	R/W	0	When asserted, along with the IP Interrupt Enable bit, the interrupt is enabled.
31:1	reserved			

Tabelle 2.4: Global Interrupt Enable Register (GIER)

Bit	Name	Access Type	Reset Value	Description
0x08 IPIER - IP Interrupt Enable Register				
0	ipie	R/W	0	When asserted, along with Global Interrupt Enable bit, the interrupt is enabled. (default: uses the internal ap_done signal to trigger an interrupt)
31:1	reserved			

Tabelle 2.5: IP Interrupt Enable Register (IPIER)

Bit	Name	Access Type	Reset Value	Description
0x0C IPISR - IP Interrupt Status Register				
0	ipis	R/W	0	Toggle on write. (write 1 to clear(W1C))
31:1	reserved			

Tabelle 2.6: IP Interrupt Status Register (IPISR)

Bit	Name	Access Type	Reset Value	Description
0x10 IDR - ID Register				
31:0	ID	R	0x80010744	Distinct ID for PmodCLP Controller

Tabelle 2.7: ID Register (IDR)

Bit	Name	Access Type	Reset Value	Description
0x14 VERR - Version Register				
31:0	VER	R	0x80001000	Version

Tabelle 2.8: Version Register (VERR)

Bit	Name	Access Type	Reset Value	Description
0x18 SCSR0 - Special Control and Status Register				
0	lcd_enable	R/W	0	Master enable for LCD display (0 = disabled, 1 = enabled)
1	clear_display	R/W	0	Write 1 to clear the display, auto-clears when operation completes
				Continued on next page

Bit	Name	Access Type	Reset Value	Description
2	return_home	R/W	0	Write 1 to return cursor to home position, auto-clears when operation completes
3	cursor_on	R/W	0	Enable cursor visibility (0 = off, 1 = on)
4	cursor_blink	R/W	0	Enable cursor blinking (0 = no blink, 1 = blink)
5	display_shift	R/W	0	Enable display shift (0 = no shift, 1 = shift)
6	shift_direction	R/W	0	Shift direction (0 = left, 1 = right)
7	auto_scroll	R/W	0	Enable automatic scrolling for long text
8	reset_lcd	R/W	0	Write 1 to reset LCD controller, auto-clears when operation completes
9	busy_flag	R	0	Indicates LCD controller is busy
10:15	reserved			
23:16	fsm_state	R	0	Current state of LCD controller FSM (for debugging)
31:24	error_code	R	0	Error code if error_flag is set in GCSR

Tabelle 2.9: Special Control and Status Register (SCSR0)

Bit	Name	Access Type	Reset Value	Description
0x1C CDR - Character Data Register				
7:0	char_data	R/W	0	Character data to write to LCD
15:8	char_addr	R/W	0	DDRAM address for character (00H-27H for line 1, 40H-67H for line 2)
16	write_char	R/W	0	Write 1 to initiate character write to specified address
17	read_char	R/W	0	Write 1 to read character from specified address
31:18	reserved			

Tabelle 2.10: Character Data Register (CDR)

TODO: Mit anderer Gruppe sprechen - wie Austausch von Daten funktioniert, über Input/Register/? –> SDR?? - Formatting-Switch Inch/cm siehe FMR??

Bit	Name	Access Type	Reset Value	Description
0x2C FMR - Format and Mode Register				

Continued on next page

Bit	Name	Access Type	Reset Value	Description
0	auto_format	R/W	0	Enable automatic formatting of sensor data
1	show_units	R/W	0	Show measurement units alongside values
2	leading_zeros	R/W	0	Show leading zeros (0 = hide, 1 = show)
3	fixed_point	R/W	0	Enable fixed-point display format
7:4	decimal_places	R/W	0	Number of decimal places to display (0-8)
15:8	display_mode	R/W	0	Display mode selection (0 = raw, 1 = formatted, etc.)
31:16	reserved			

Tabelle 2.11: Format and Mode Register (FMR)

Bit	Name	Access Type	Reset Value	Description
0x30 SDR - Sensor Data Register				
15:0	sensor_value	R/W	0	Raw sensor value from PmodMAXSONAR
23:16	sensor_status	R	0	Status code of sensor
31:24	scaling_factor	R/W	0	Scaling factor for sensor value display

Tabelle 2.12: Sensor Data Register (SDR)

3. Sensor