DE0_CycloneIII_Eval_Board (Nr. 11.35)

Hardware-Beschreibung

Pin-Belegungen

Demo- und Testprojekte (mit/ohne NIOS) (mit/ohne µCOSII)

1 Historie

Platinen-Version	Datum	Release Note	Autor
1.2	23.07.2012	Korrektur im Kapitel 11 (Testsignale)	JM
1.1	07.03.2012	Erweiterung mit zusätzlichen I/O-Leitungen für das CycloneIII-Board V2.0	JM
1.0	28.2.2012	First Release	JM

Dieses Dokument ist identisch abgelegt im

- **Verzeichnis VHDL Projekte für *DE0_CycloneIII_Eval_Board (Nr. 11.35)* und**
- **Platinenverzeichnis für *DE0_CycloneIII_Eval_Board (Nr. 11.35)***

Das Dokument beschreibt die Hardware-Konfigurationen und beinhaltet Demo- und Testprojektdateien für stand-alone-VHDL und NIOS/µCOSII-Applikationen.

2 Projektüberblick

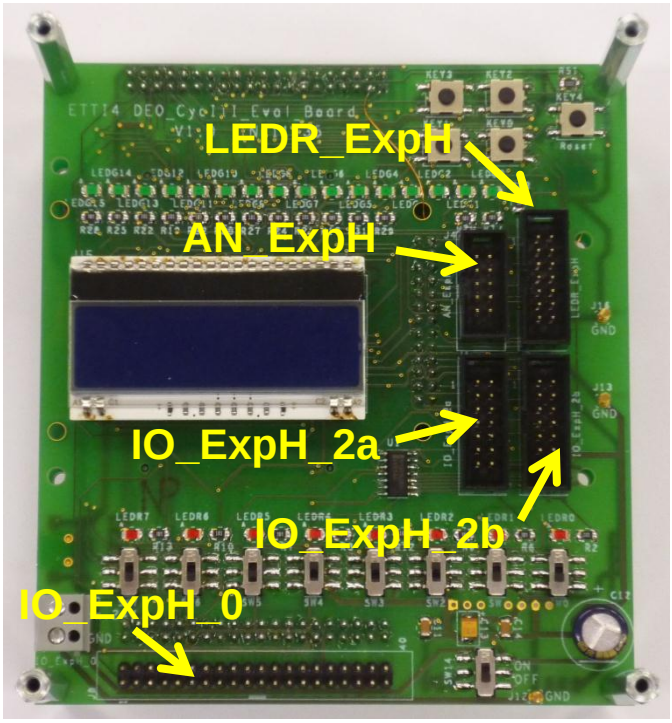
Das DE0_CycloneIII-Evaluation-Board adaptiert die HW-Aufsteckmodule und erlaubt über Bedien- und Anzeigeelemente ein komfortableres Testen und Prototyping.
Das Eval-Board ist als „Doppelpack“ aufgebaut:

- Top: Bedien- und Anzeigeelemente, Anschlussstecker für Steuerungs- und Messzwecke
- Bottom: Steckplatz für DE0-Nano-Board oder CycloneIII-Board

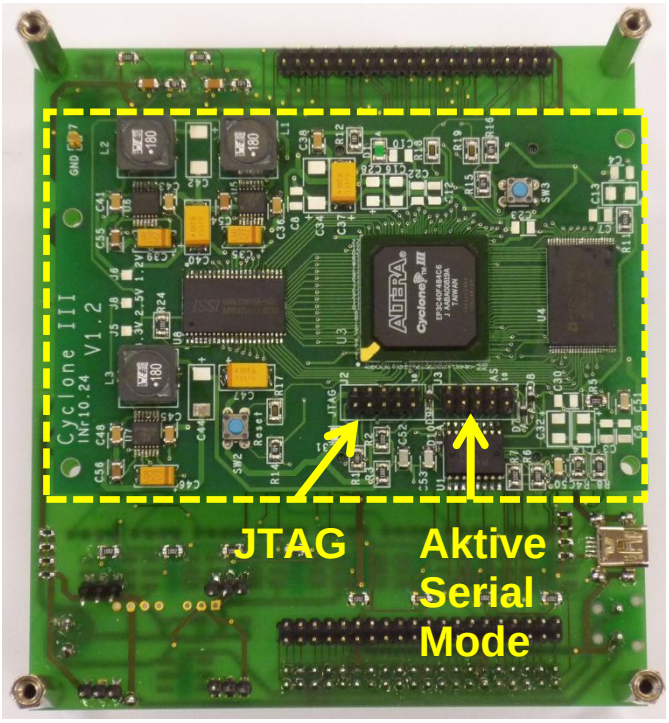
2.1 Spezifikation

- Steckplatz für CycloneIII-Board, Nr. 10.24
  alternativ
  Steckplatz für DE0-Nano-Board (mit/ohne DE0_Nano-Flash-Board Nr. 11.29)
- LCD DOGM162B-A (16 x 2) mit Hintergrundbeleuchtung weiß
- 4 Taster (entprellt, low-aktiv)
- separater Resettaster (low-aktiv)
- 8 Schiebeschalter
- 16 LED grün, über Schieberegister angesteuert (MSB first)
- 8 LED rot, direkt einzeln angesteuert (high-aktiv)
- 10-pol. Stiftleiste (ExpansionHeader) mit Parallelbelegung der LED-rot-Signale (I/O-Ext.)
- 40-pol. Stiftleiste (ExpansionHeader) für GPIO (Pinbelegung abhängig vom Aufsteckboard)
- 10-pol. Stiftleiste (ExpansionHeader) für 8 Analog-In-Ports (nur bei DE0)
- zwei 14-pol. Stiftleisten (ExpansionHeader) für je 8 GPIO (nur bei DE0)
- Spannungsversorgung 5V über USB-B-Mini (nicht schaltbar)
- optional: Spannungsversorgung über Schaltregler, Vin ca. 9...36 V (schaltbar über SW14)
- I/O-Standard: 3,3V TTL

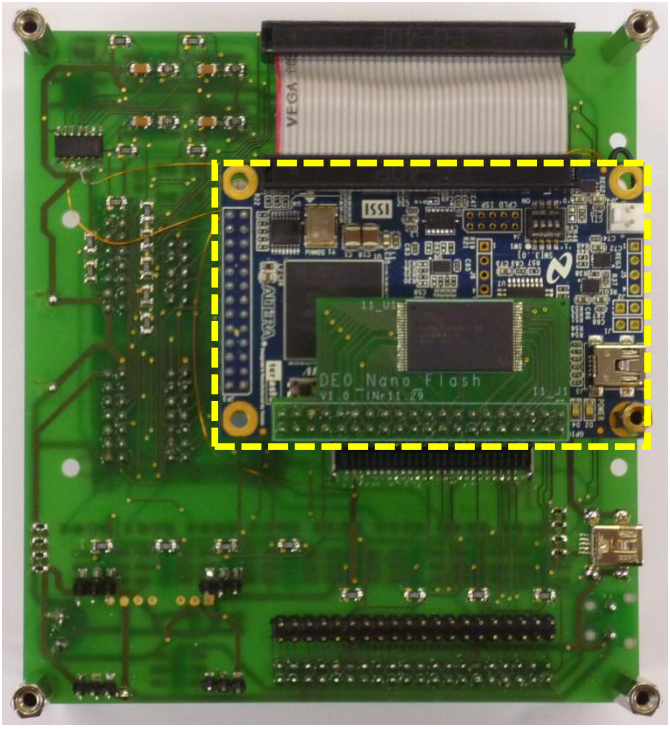
Für die Ansteuerung der HW-Resourcen des Eval-Boards in einer User Application müssen spezifische Module eingebunden werden!
(siehe auch separater „Nios-Baukasten“ bzw.
Projektunterverzeichnis „eval_board_common_c_sources“)



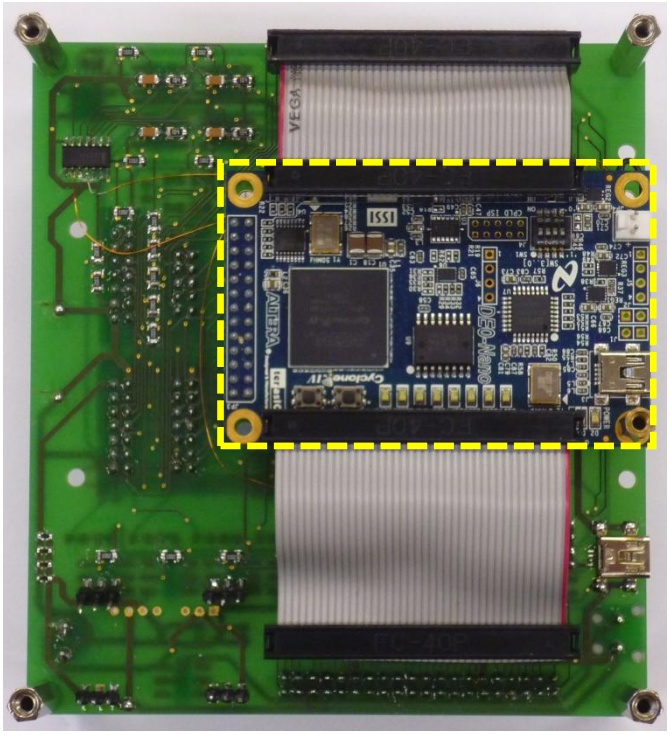
Eval-Board top



Eval-Board bottom, mit Cyclone III Board



Eval-Board bottom, mit DE0-Nano Board
mit Flashboard



Eval-Board bottom, mit DE0-Nano Board
ohne Flashboard

2.2 Bedienung

In den Demo-Projekten **mit NIOS** können über Tasten und Schalter Einstellungen gemacht werden und an LEDs und am LCD angezeigt werden.

Key0 startet Shifting der 16 grünen LEDGs
die LEDGs sind ein „doppeltes gespiegeltes“ Abbild des LEDR-Datenmusters bzw. der Schalterstellung
(die LEDR 7..0 werden als LEDR 0..7 auf die LEDG-Position 15..8 geschoben)

Key1 übernimmt Schalterstellung in rote LEDRs

Key2 nur Funktion mit Testsoftware zur Board-Inbetriebnahme

Key3 nur Funktion mit Testsoftware zur Board-Inbetriebnahme

Key4 Reset

Wenn **alle** Schiebeschalter ON sind, schreibt das LCD periodische Testdaten.

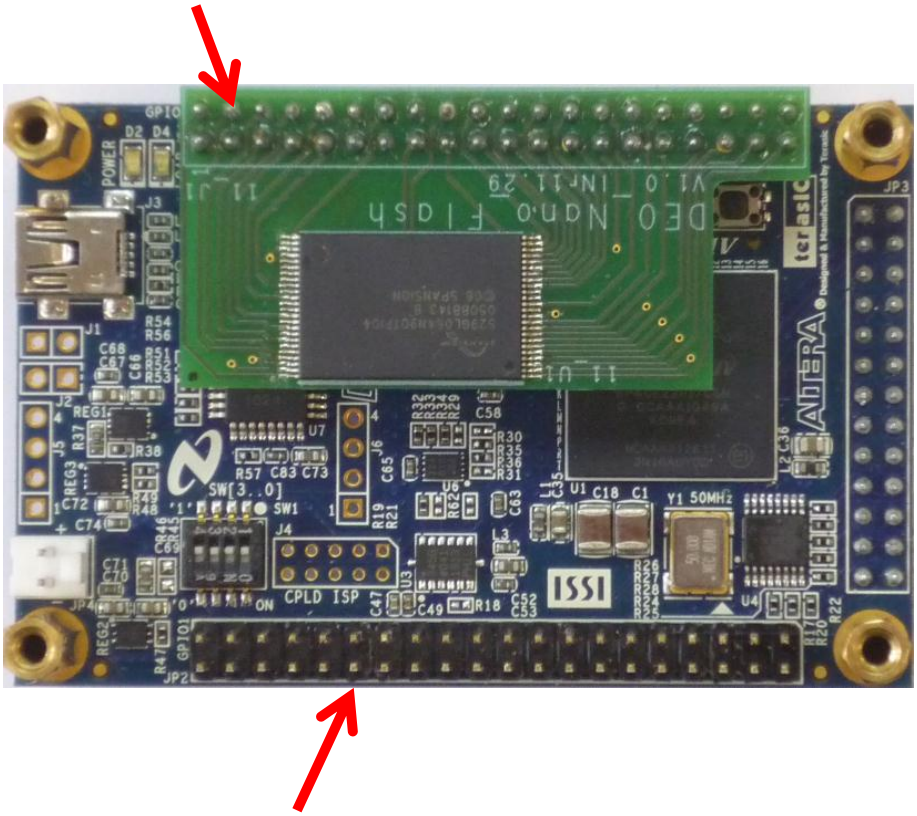
Wenn **nicht alle** Schiebeschalter ON sind, wird im LCD die Stellung der Schalter abgebildet (äquivalent zur Anzeige der grünen LEDGs).

3 Hardware Eval-Board

- DE0-Nano-Board
  ggf. mit DE0_Nano-Flash-Board Nr. 11.29
- CycloneIII-Board, Nr. 10.24

Wenn das DE0_Nano-Flash-Board (Nr. 11.29) für das DE0-Nano verwendet wird, reduziert sich durch die Belegung des GPIO-0-Headers die verfügbare Anzahl an externen GPIOs.

Das DE0_Nano-Flash-Board (Nr. 11.29) darf bei Verwendung auf dem Eval-Board NUR auf den GPIO-0-Header des DE0-Nano-Boards aufgesteckt werden!
Pin-Belegung für GPIO-0 verwenden !!!!



Der GPIO-1-Header ist mit Eval-Board-internen Signalen belegt!

Bei neuem DE0-Nano bzw. CycloneIII-Board oder bei unbekanntem Software-Inhalt:
Das Board VOR dem Einstecken in das Eval-Board programmieren, da sonst durch vorhandene Softwareaktionen Schäden entstehen könnten !!!!!!

3.1 Board-Aufbau und Stecker

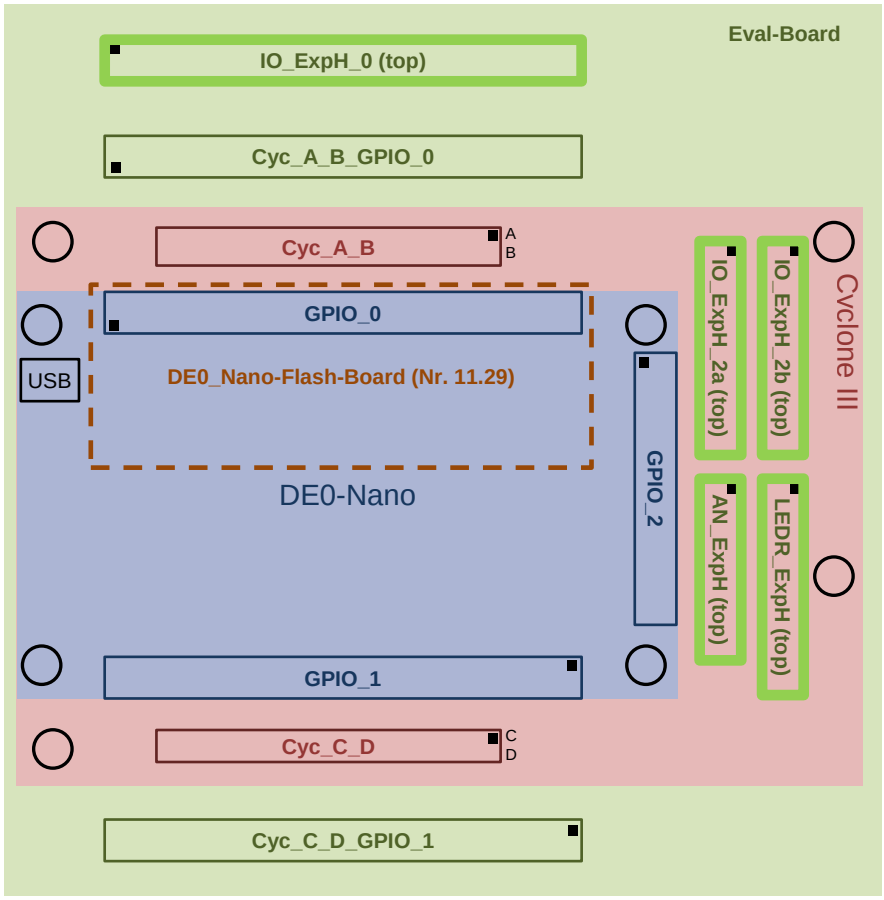
Schematische Anordnung der einzelnen Stecker zur besseren Orientierung bei der Pinbelegung
!!! Bottom-View und überlagerte Sicht auf die beiden Aufsteckboards !!!
Kennzeichnung von Pin1 --> ■

GPIO_0	DE0-Nano
GPIO_1	DE0-Nano
GPIO_2	DE0-Nano
Cyc_A_B	Cyclone III
Cyc_C_D	Cyclone III
Cyc_A_B_GPIO_0	kombinierte Belegung
Cyc_C_D_GPIO_1	kombinierte Belegung (für Eval-Board-Logik)
IO_ExpH_0	**für externe Beschaltung (dig. IO 3,3V TTL)**
IO_ExpH_2a	**für externe Beschaltung (dig. IO 3,3V TTL)**
IO_ExpH_2b	**für externe Beschaltung (dig. IO 3,3V TTL)**
AN_ExpH	**für externe Beschaltung (Analogsignale)**
LEDR_ExpH	**für externe Beschaltung (parallele LEDR-Signale 3,3V TTL)**

Die IO-Signale beider Aufsteckboards werden an den **IO-ExpH**-Steckern überlagert.

Hinweis: VCC_Sys = VCC_50

!!!! Bottom-View !!!!



Anschlussseite zur Board-Logik

4 Empfehlungen für Rapid Prototyping

Dieser Abschnitt gibt Hinweise für ein schnelles Erstellen von eigenen Projekten.
Die Vorgehensweise ist eine Empfehlung und nicht verbindlich.

1. Download des Demo-Projektes bzw. der Projekt-Templates aus dem „Nios-Baukasten“ und Einrichten der eigenen Projektstruktur

2. Festlegung des verwendeten Aufsteck-Boards (DE0-Nano oder CycloneIII)

3. Festlegung der genauen HW-Konfiguration (z.B. DE0-Nano mit/ohne Flashboard, mit/ohne SDRAM-PLL, Port-Belegung)

4. Festlegung der genauen SW-Konfiguration (z.B. mit/ohne NIOS, mit/ohne µCOSII)

5. Aufbau der eigenen QSys-Projektdatei auf Basis der entsprechenden Datei aus dem Demo-Projekt bzw. Template-Projekt
   ggf. Anpassung / Erweiterung etc. an die applikationsspezifischen Anforderungen

6. Aufbau der Quartus/VHDL-Filestruktur, ggf. mit Anpassung der eigenen VHDL-Dateien auf Basis der vorhandenen (TopLevel)-VHDL-Files

7. Einbindung der zwingend erforderlichen Eval-Board-spezifischen VHDL-Module

8. Erzeugung eines lauffähigen Quartus-Systems

9. Aufbau des eigenen NIOS-Projektes auf Basis der entsprechenden Demo-Projekte

10. Einbindung der (zwingend) erforderlichen Eval-Board-spezifischen C-Module und Konfiguration für Betrieb *mit/ohne µCOSII* (in Header *general.h*)

11. Erzeugung eines lauffähigen NIOS-Systems

12. last, but not least: Doku nicht vergessen !

5 Pinbelegungen (ausführliche Eval-Board-interne Sicht)

5.1 GPIO-0 .... Cyc_A_B .... IO_ExpH_0

DE0-Nano GPIO-0 (ohne Flashboard)						Cyc_A_B_GPIO_0				IO_ExpH_0				Cyclone III Cyc_A_B (* Pin erst ab V2.0)					
Cycl. IV	GPIO-0	Pin	Cycl. IV	GPIO-0	Pin	Cyc_A_B _GPIO_0	Pin	Cyc_A_B _GPIO_0	Pin	IO_ExpH_0	Pin	IO_ExpH_0	Pin	J1A	Pin	Cycl. III	J1B	Pin	Cycl. III
A8	GPIO-0-IN0	1	D3	GPIO-00	2	N.C.	1	GPIO-00	2	to Cyc_C_D _GPIO_1-1	1	GPIO-00	2				ExpH_U_12	B16*	U_12
B8	GPIO-0-IN1	3	C3	GPIO-01	4	GPIO-0-IN1	3	GPIO-01	4	GPIO-0-IN1	3	GPIO-01 ExpH_R_1	4				ExpH_R_1	B24	R1
A2	GPIO-02	5	A3	GPIO-03	6	N.C.	5	GPIO-03	6	to Cyc_C_D _GPIO_1-5	5	GPIO-03 ExpH_P_1	6				ExpH_P_1	B23	P1
B3	GPIO-04	7	B4	GPIO-05	8	GPIO-04	7	GPIO-05	8	GPIO-04 ExpH_AB_3	7	GPIO-05 ExpH_N_1	8	ExpH_AB_3	A24	AB3	ExpH_N_1	B22	N1
A4	GPIO-06	9	B5	GPIO-07	10	GPIO-06	9	GPIO-07	10	GPIO-06 ExpH_A_23	9	GPIO-07 ExpH_M_1	10	ExpH_A_23	A23	AA5	ExpH_M_1	B21	M1
--	VCC-Sys	11	--	Gnd	12	VCC-Sys	11	Gnd	12	VCC-Sys	11	Gnd	12						
A5	GPIO-08	13	D5	GPIO-09	14	GPIO-08	13	GPIO-09	14	GPIO-08	13	GPIO-09	14	ExpH_T_12	A16*	T_12	ExpH_U_14	B17*	U_14
B6	GPIO-010	15	A6	GPIO-011	16	GPIO-010	15	GPIO-011	16	GPIO-010	15	GPIO-011	16	ExpH_U_13	A15*	U_13	ExpH_U_9	B15*	U_9
B7	GPIO-012	17	D6	GPIO-013	18	GPIO-012	17	GPIO-013	18	GPIO-012	17	GPIO-013	18	ExpH_U_10	A14*	U_10	ExpH_E_16	B12*	E_16
A7	GPIO-014	19	C6	GPIO-015	20	GPIO-014	19	GPIO-015	20	GPIO-014	19	GPIO-015	20	ExpH_U_8	A13*	U_8	ExpH_U_7	B14*	U_7
C8	GPIO-016	21	E6	GPIO-017	22	GPIO-016	21	GPIO-017	22	GPIO-016	21	GPIO-017	22	ExpH_E_15	A12*	E_15	ExpH_N_16	A22*	N_16
E7	GPIO-018	23	D8	GPIO-019	24	GPIO-018	23	GPIO-019	24	GPIO-018	23	GPIO-019	24	ExpH_E_14	A11*	E_14	ExpH_T_15	A21*	T_15
E8	GPIO-020	25	F8	GPIO-021	26	GPIO-020	25	GPIO-021	26	GPIO-020	25	GPIO-021	26	ExpH_G_10	A10*	G_10	ExpH_G_11	B11*	G_11
F9	GPIO-022	27	E9	GPIO-023	28	GPIO-022	27	GPIO-023	28	GPIO-022	27	GPIO-023	28	ExpH_G_9	A9*	G_9	ExpH_E_12	B10*	E_12
--	VCC_33	29	--	Gnd	30	VCC_33	29	Gnd	30	VCC_33	29	Gnd	30						
C9	GPIO-024	31	D9	GPIO-025	32	GPIO-024	31	GPIO-025	32	GPIO-024	31	GPIO-025	32	ExpH_E_11	A8*	E_11	ExpH_F_2	B3*	F_2
E11	GPIO-026	33	E10	GPIO-027	34	GPIO-026	33	GPIO-027	34	GPIO-026	33	GPIO-027	34	ExpH_M_7	A7*	M_7	ExpH_G_4	B4*	G_4
C11	GPIO-028	35	B11	GPIO-029	36	GPIO-028	35	GPIO-029	36	GPIO-028	35	GPIO-029	36	ExpH_M_3	A6*	M_3	ExpH_J_7	B5*	J_7
A12	GPIO-030	37	D11	GPIO-031	38	GPIO-030	37	GPIO-031	38	GPIO-030	37	GPIO-031	38	ExpH_K_7	A5*	K_7			
D12	GPIO-032	39	B12	GPIO-033	40	GPIO-032	39	GPIO-033	40	GPIO-032	39	GPIO-033	40	ExpH_G_3	A4*	G_3			
														VCC-Sys	A1		VCC-Sys	B1	
														Gnd	A2 A25		Gnd	B2 B25	
														UReset_n	A3				

5.2 GPIO-1 .... Cyc_C_D .... Eval-Board-Logik

DE0-Nano GPIO-1						Cyc_C_D_GPIO_1				Cyclone III Cyc_C_D					
Cycl. IV	GPIO-1	Pin	Cycl. IV	GPIO-1	Pin	Cyc_C_D _GPIO_1	Pin	Cyc_C_D _GPIO_1	Pin	J1C	Pin	Cycl. III	J1D	Pin	Cycl. III
T9	GPIO-1-IN0	1	F13	GPIO-10	2	to IO_ExpH_0-1	1	LEDG_SER_DATA	2	ExpH_A_20	C3	A20	D_OE_IN	D14	N22
R9	GPIO-1-IN1	3	T15	GPIO-11	4	UReset_n	3	LEDG_SER_CLK	4	ExpH_B_21	C4	B21	D_CLRn_IN	D15	N21
T14	GPIO-12	5	T13	GPIO-13	6	to IO_ExpH_0-5	5	LCD_RW	6	ExpH_C_21	C5	C21	ExpH_B_20	D3	B20
R13	GPIO-14	7	T12	GPIO-15	8	KEY3	7	LCD_EN	8	ExpH_D_21	C6	D21	ExpH_O_22	D4	C19
R12	GPIO-16	9	T11	GPIO-17	10	KEY2	9	LCD_RS	10	ExpH_E_21	C7	E21	ExpH_I_22	D5	N20
--	VCC-Sys	11	--	Gnd	12	VCC-Sys	11	Gnd	12	--			--		
T10	GPIO-18	13	R11	GPIO-19	14	KEY1	13	LCD_DQ7	14	ExpH_F_21	C8	F21	ExpH_Q_22	D6	M20
P11	GPIO-110	15	R10	GPIO-111	16	KEY0	15	LCD_DQ6	16	ExpH_H_21	C9	H21	ExpH_Z_22	D7	F20
N12	GPIO-112	17	P9	GPIO-113	18	LEDR7	17	LCD_DQ5	18	ExpH_J_21	C10	J21	ExpH_H_22	D8	H22
N9	GPIO-114	19	N11	GPIO-115	20	LEDR6	19	LCD_DQ4	20	ExpH_K_21	C11	K21	ExpH_J_22	D9	J22
L16	GPIO-116	21	K16	GPIO-117	22	LEDR5	21	LCD_DQ3	22	ExpH_L_21	C12	L21	ExpH_K_22	D10	K22
R16	GPIO-118	23	L15	GPIO-119	24	LEDR4	23	LCD_DQ2	24	ExpH_M_21	C13	M21	ExpH_L_22	D11	L22
P15	GPIO-120	25	P16	GPIO-121	26	LEDR3	25	LCD_DQ1	26	ExpH_P_21	C14	P21	ExpH_P_22	D16	P22
R14	GPIO-122	27	N16	GPIO-123	28	LEDR2	27	LCD_DQ0	28	ExpH_R_21	C15	R21	ExpH_R_22	D17	R22
--	VCC_33	29	--	Gnd	30	VCC_33	29	Gnd	30	--					
N15	GPIO-124	31	P14	GPIO-125	32	LEDR1	31	SW0	32	ExpH_T_22	C16	M22	ExpH_U_22	D18	U22
L14	GPIO-126	33	N14	GPIO-127	34	LEDR0	33	SW1	34	ExpH_U_21	C17	U21	ExpH_V_22	D19	V22
M10	GPIO-128	35	L13	GPIO-129	36	SW7	35	SW2	36	ExpH_V_21	C18	V21	ExpH_W_22	D20	W22
J16	GPIO-130	37	K15	GPIO-131	38	SW6	37	SW3	38	ExpH_W_21	C19	W21	ExpH_Y_22	D21	Y22
J13	GPIO-132	39	J14	GPIO-133	40	SW5	39	SW4	40	ExpH_Y_21	C20	Y21	ExpH_AA_22	D22	AA22
										VCC-Sys	C1		VCC-Sys	D1	
										Gnd	C2 C25		Gnd	D2 D25	

Hinweis zum Cycl.III-Board: Signal *D_CLRn_IN* könnte als Input verwendet werden, allerdings geht Signal *UReset_n* parallel auf Cycl.III/E4 = int.Reset.
---> d.h. Verwendung als Input ist relativ sinnlos bei gleichzeitigem Reset !!!

5.3 GPIO-2 .... Cyc_A_B .... IO_ExpH_2a

DE0-Nano GPIO-2						IO_ExpH_2a				Cyclone III Cyc_A_B (* Pin erst ab V2.0)					
Cycl. IV	GPIO-0	Pin	Cycl. IV	GPIO-0	Pin	IO_ExpH_2a	Pin	IO_ExpH_2a	Pin	J1A	Pin	Cycl. III	J1B	Pin	Cycl. III
	VCC_33	1	E15	GPIO-2-IN0	2	Gnd	1	VCC_50	2						
E16	GPIO-2-IN1	3	M16	GPIO-2-IN2	4	VCC_33	3	GPIO-2-IN1	4				ExpH_T_13	A17*	T_13
A14	GPIO-20	5	B16	GPIO-21	6	GPIO-20	5	GPIO-210	6	ExpH_E_10	B9*	E_10			
C14	GPIO-22	7	C16	GPIO-23	8	GPIO-22	7	GPIO-212	8	ExpH_N_7	B8*	N_7			
C15	GPIO-24	9	D16	GPIO-25	10	GPIO-24	9	Gnd	10	ExpH_P_3	B7*	P_3			
D15	GPIO-26	11	D14	GPIO-27	12	GPIO-26	11	Gnd	12	ExpH_K_8	B6*	K_8			
F15	GPIO-28	13	F16	GPIO-29	14	GPIO-28	13	Gnd	14						
F14	GPIO-210	15	G16	GPIO-211	16										
G15	GPIO-212	17		Analog_In5	18										
	Analog_In6	19		Analog_In7	20										
	Analog_In3	21		Analog_In2	22										
	Analog_In4	23		Analog_In0	24										
	Analog_In1	25		Gnd	26										

(siehe auch User Manual und Doku zum DE0-Nano-Board und zum CycloneIII-Board)

5.4 GPIO-2 .... Cyc_A_B .... IO_ExpH_2b

DE0-Nano GPIO-2						IO_ExpH_2b				Cyclone III Cyc_A_B (* Pin erst ab V2.0)					
Cycl. IV	GPIO-0	J1A	J1A	J1A	J1A	J1A	J1A	IO_ExpH_2b	Pin	J1A	Pin	Cycl. III	J1B	Pin	Cycl. III
	VCC_33	1	E15	GPIO-2-IN0	2	Gnd	1	VCC_50	2						
E16	GPIO-2-IN1	3	M16	GPIO-2-IN2	4	VCC_33	3	GPIO-2-IN2	4				ExpH_V_14	A18*	V_14
A14	GPIO-20	5	B16	GPIO-21	6	GPIO-23	5	GPIO-2-IN0	6						
C14	GPIO-22	7	C16	GPIO-23	8	GPIO-25	7	GPIO-21	8	ExpH _T_14	A20*	T_14	ExpH_U_15	A19*	U_15
C15	GPIO-24	9	D16	GPIO-25	10	GPIO-27	9	Gnd	10						
D15	GPIO-26	11	D14	GPIO-27	12	GPIO-29	11	Gnd	12						
F15	GPIO-28	13	F16	GPIO-29	14	GPIO-211	13	Gnd	14						
F14	GPIO-210	15	G16	GPIO-211	16										
G15	GPIO-212	17		Analog_In5	18										
	Analog_In6	19		Analog_In7	20										
	Analog_In3	21		Analog_In2	22										
	Analog_In4	23		Analog_In0	24										
	Analog_In1	25		Gnd	26										

5.5 GPIO-2 .... Cyc_A_B .... AN_ExpH

DE0-Nano GPIO-2						AN_ExpH			
Cycl. IV	**GPIO-0**	**Pin**	**Cycl. IV**	**GPIO-0**	**Pin**	**AN_ExpH**	**Pin**	**AN_ExpH**	**Pin**
	VCC_33	1	E15	GPIO-2-IN0	2	Gnd	1	Analog_In7	2
E16	GPIO-2-IN1	3	M16	GPIO-2-IN2	4	VCC_33	3	Analog_In6	4
A14	GPIO-20	5	B16	GPIO-21	6	Analog_In5	5	Analog_In4	6
C14	GPIO-22	7	C16	GPIO-23	8	Analog_In3	7	Analog_In2	8
C15	GPIO-24	9	D16	GPIO-25	10	Analog_In1	9	Analog_In0	10
D15	GPIO-26	11	D14	GPIO-27	12				
F15	GPIO-28	13	F16	GPIO-29	14				
F14	GPIO-210	15	G16	GPIO-211	16				
G15	GPIO-212	17		Analog_In5	18				
	Analog_In6	19		Analog_In7	20				
	Analog_In3	21		Analog_In2	22				
	Analog_In4	23		Analog_In0	24				
	Analog_In1	25		Gnd	26				

5.6 LEDR_ExpH

LEDR_ExpH			
LEDR_ExpH	**Pin**	**LEDR_ExpH**	**Pin**
Gnd	1	N.C.	2
VCC_33	3	LEDR7	4
LEDR5	5	LEDR6	6
LEDR3	7	LEDR4	8
LEDR2	9	Gnd	10
LEDR1	11	Gnd	12
LEDR0	13	Gnd	14

Hinweis zur Pinbelegung von CLOCK und RESET:

DE0-Nano-Board:

- Reset-Taster (Key4) ---> CycloneIV-Pin R9
- Systemclock ---> CycloneIV-Pin R8

CycloneIII-Board:

- Reset-Taster (Key4) ---> CycloneIII-Pin E4
- Systemclock ---> CycloneIII-Pin G1

6 Pinbelegungen (Nutzer-Sicht für DE0_Nano bzw. CycloneIII)

Es wurde möglichst die Namenskonvention aus dem UserManual bzw. der Board-Doku übernommen.
Die Pinbelegungen für die Board-interne Logik ist nicht aufgeführt, dafür sollen die Quartus-pinning-Files importiert werden.
Die Eingänge in Quartus auf *„unused pins = input tri-stated with weak pull-up“* stellen, damit ungenutzte IN-Pins definiert auf „H“ liegen.
Bei allen anderen (genutzten) Eingängen muß der Anwender selbst ggf. für die Pullup/Pulldown-Widerstände in der Hardware sorgen!!

6.1 Pinbelegung für DE0_Nano (ohne Flashboard)

IO_ExpH_0					
Pin Cycl. IV	**IO_ExpH_0**	**Pin**	**Pin Cycl. IV**	**IO_ExpH_0**	**Pin**
T9	GPIO-1-IN0	1	D3	GPIO-00	2
B8	GPIO-0-IN1	3	C3	GPIO-01	4
T14	GPIO_12	5	A3	GPIO-03	6
B3	GPIO-04	7	B4	GPIO-05	8
A4	GPIO-06	9	B5	GPIO-07	10
---	VCC-Sys	11	---	Gnd	12
A5	GPIO-08	13	D5	GPIO-09	14
B6	GPIO-010	15	A6	GPIO-011	16
B7	GPIO-012	17	D6	GPIO-013	18
A7	GPIO-014	19	C6	GPIO-015	20
C8	GPIO-016	21	E6	GPIO-017	22
E7	GPIO-018	23	D8	GPIO-019	24
E8	GPIO-020	25	F8	GPIO-021	26
F9	GPIO-022	27	E9	GPIO-023	28
---	VCC_33	29	---	Gnd	30
C9	GPIO-024	31	D9	GPIO-025	32
E11	GPIO-026	33	E10	GPIO-027	34
C11	GPIO-028	35	B11	GPIO-029	36
A12	GPIO-030	37	D11	GPIO-031	38
D12	GPIO-032	39	B12	GPIO-033	40

Bei Verwendung des Flashboards fallen alle

GPIO-0-Signale weg - es sind dann nur noch zwei

rote Signale (von GPIO-1) sind verfügbar!

IO_ExpH_2a					
Cycl. IV	**IO_ExpH_2a**	**Pin**	**Cycl. IV**	**IO_ExpH_2a**	**Pin**
---	Gnd	1	---	VCC_50	2
---	VCC_33	3	E16	GPIO-2-IN1	4
A14	GPIO-20	5	F14	GPIO-210	6
C14	GPIO-22	7	G15	GPIO-212	8
C15	GPIO-24	9	---	Gnd	10
D15	GPIO-26	11	---	Gnd	12
F15	GPIO-28	13	---	Gnd	14

IO_ExpH_2b					
Cycl. IV	**IO_ExpH_2b**	**Pin**	**Cycl. IV**	**IO_ExpH_2b**	**Pin**
---	Gnd	1	---	VCC_50	2
---	VCC_33	3	M16	GPIO-2-IN2	4
C16	GPIO-23	5	E15	GPIO-2-IN0	6
D16	GPIO-25	7	B16	GPIO-21	8
D14	GPIO-27	9	---	Gnd	10
F16	GPIO-29	11	---	Gnd	12
G16	GPIO-211	13	---	Gnd	14

AN_ExpH			
AN_ExpH	**Pin**	**AN_ExpH**	**Pin**
Gnd	1	Analog_In7	2
VCC_33	3	Analog_In6	4
Analog_In5	5	Analog_In4	6
Analog_In3	7	Analog_In2	8
Analog_In1	9	Analog_In0	10

LEDR_ExpH					
Cycl. IV	**LEDR_ExpH**	**Pin**	**Cycl. IV**	**LEDR_ExpH**	**Pin**
---	Gnd	1	---	N.C.	2
---	VCC_33	3	N12	LEDR7	4
L16	LEDR5	5	N9	LEDR6	6
P15	LEDR3	7	R16	LEDR4	8
R14	LEDR2	9	---	Gnd	10
N15	LEDR1	11	---	Gnd	12
L14	LEDR0	13	---	Gnd	14

Wichtig !!!!!!!!!!!!!!!!
Die LEDR-Signalleitungen können als zusätzliche Eingänge verwendet werden.
Dazu muss aber sichergestellt sein, dass im Quartus diese Leitungen auch auf INPUT geschaltet sind.
In diesem Fall leuchten die LEDR entsprechend dem Signalpegel an diesen Eingängen.

6.2 Pinbelegung für CycloneIII

IO_ExpH_0					
Pin Cycl. III	**IO_ExpH_0**	**Pin**	**Pin Cycl. III**	**IO_ExpH_0**	**Pin**
N22	D_OE_IN	1	U12	ExpH_U_12	2
N21	(D_CLRn_IN)	3	R1	ExpH_R_1	4
B20	ExpH_B_20	5	P1	ExpH_P_1	6
AB3	ExpH_AB_3	7	N1	ExpH_N_1	8
AA5	ExpH_A_23	9	M1	ExpH_M_1	10
		11			12
T_12	ExpH_T_12	13	U_14	ExpH_U_14	14
U_13	ExpH_U_13	15	U_9	ExpH_U_9	16
U_10	ExpH_U_10	17	E_16	ExpH_E_16	18
U_8	ExpH_U_8	19	U_7	ExpH_U_7	20
E_15	ExpH_E_15	21	N_16	ExpH_N_16	22
E_14	ExpH_E_14	23	T_15	ExpH_T_15	24
G_10	ExpH_G_10	25	G_11	ExpH_G_11	26
G_9	ExpH_G_9	27	E_12	ExpH_E_12	28
		29			30
E_11	ExpH_E_11	31	F_2	ExpH_F_2	32
M_7	ExpH_M_7	33	G_4	ExpH_G_4	34
M_3	ExpH_M_3	35	J_7	ExpH_J_7	36
K_7	ExpH_K_7	37			38
G_3	ExpH_G_3	39			40

Diese Pins sind erst ab CycloneIII-Platinenversion V2.0 verfügbar !!!

Hinweis:
Signal *D_CLRn_IN* an IO_ExpH_0-3 könnte prinzipiell als Input (dual purpose pin) verwendet werden, allerdings liegt das Signal parallel zu *UReset_n* vom Reset-Taster, was über eine separate Reset-Leitung direkt an den Cycl. III / Pin E4 = interner Chip-Reset führt.

IO_ExpH_2a					
Pin Cycl. III	IO_ExpH_2a	Pin	Pin Cycl. III	IO_ExpH_2a	Pin
---	Gnd	1	---	VCC_50	2
---	VCC_33	3	T_13	ExpH_T_13	4
E_10	ExpH_E_10	5			6
N_7	ExpH_N_7	7			8
P_3	ExpH_P_3	9	---	Gnd	10
K_8	ExpH_K_8	11	---	Gnd	12
		13	---	Gnd	14

IO_ExpH_2b					
Pin Cycl. III	IO_ExpH_2b	Pin	Pin Cycl. III	IO_ExpH_2b	Pin
---	Gnd	1	---	VCC_50	2
---	VCC_33	3	V_14	ExpH_V_14	4
		5			6
T_14	ExpH_T_14	7	U_15	ExpH_U_15	8
		9	---	Gnd	10
		11	---	Gnd	12
		13	---	Gnd	14

Diese Pins sind erst ab CycloneIII-Platinenversion V2.0 verfügbar !!!

LEDR_ExpH					
Pin Cycl. III	LEDR_ExpH	Pin	Pin Cycl. III	LEDR_ExpH	Pin
---	Gnd	1	---	N.C.	2
---	VCC_33	3	H22	LEDR7	4
K22	LEDR5	5	J22	LEDR6	6
P22	LEDR3	7	L22	LEDR4	8
R22	LEDR2	9	---	Gnd	10
U22	LEDR1	11	---	Gnd	12
V22	LEDR0	13	---	Gnd	14

Wichtig !!!!!!!!!!!!!!!! Die LEDR-Signalleitungen können als zusätzliche Eingänge verwendet werden.
Dazu muss aber sichergestellt sein, dass im Quartus diese Leitungen auch auf INPUT geschaltet sind.
In diesem Fall leuchten die LEDR entsprechend dem Signalpegel an diesen Eingängen.

7 Demo- und Testprojekte

Das User-Projekt kann für Cyclone IV (DE0-Nano) oder für Cyclone III (ETTI4-eigenes Board) konfiguriert werden
Entsprechende Assignments bzw. Importierungen sind dafür in Quartus durchzuführen:

- zuerst ALLE Assignments entfernen
- dann die gewünschte qsf-Datei importieren, mit ALLEN GLOBALEN Parametern
- die zugehörige Pin-Belegung importieren (*.csv)

Bei der DE0-Nano-Board-Konfiguration gibt es QSys-Dateien für zwei Varianten:

- SDRAM ohne PLL
- SDRAM mit PLL (ggf. erforderlich wg. Clock-Phase-Shift für SDRAM-Chip)

Für die jeweilig gewünschte Konfiguration sind die entsprechenden Projekt- bzw. Konfig-Dateien zu verwenden bzw. zu importieren.
(teilweise liegen die Dateien auch im Unterverzeichnis \atom_netlists bzw. im Source-Verzeichnis)

Normalerweise unterstützen alle DE0-Nano-Konfigurationen das Flash-Board (dann sind aber keine GPIO_0-Ports verfügbar)!

Bei DE0-Nano-Konfigurationen **OHNE** Flash-Board (also nur mit SDRAM) sind GPIO_0-Ports verfügbar.

Folgende Anpassungen sind erforderlich:

Qsys-Dateien:

für DE0-Nano mit Flash-Board	NIOS-Reset-Vector auf “Flashcontroller” setzen
für DE0-Nano ohne Flash-Board	NIOS-Reset-Vector auf “SDRAM-Controller” setzen

Quartus-TopLevel-Datei “top_level_nios_shift_leds_DE0_Nano_Flash.vhd”

in Entity:	entsprechend mit/ohne Flashboard die Ports ein/auskommentieren
in Architecture:	gewünschte Component ein/auskommentieren (mit/ohneSDRAM-PLL)
	in Component PLL-Ports ein/auskommentieren
	Verdrahtung DRAM_CLK ein/auskommentieren
	in Port Map PLL-Verdrahtung ein/auskommentieren

****

(Für die Cyclone III-Konfiguration gibt es keine Variante, das Flash ist integriert, es ist nichts ein/auszukommentieren,
es ist lediglich der NIOS-Reset-Vektor einzustellen)

7.1 Projektdateien

7.1.1 Qsys / SoC / Nios

- DE0_NANO_FLASH_Eval_Board.qsys — für DE0-Nano mit/ohne Flash-Board, ohne PLL für SDRAM, ohne µCOSII
- DE0_NANO_FLASH_SDRAM_PLL_Eval_Board.qsys — für DE0-Nano mit/ohne Flash-Board, mit PLL für SDRAM, ohne µCOSII
- CycIII_Eval_ResVecFlash_memSRAM_ucosii.qsys — für Cyclone III-Board (mit integriertem Flash, NIOS-Resetvektor=Flash, für µCOSII)
- CycIII_Eval_ResVecSRAM_memSRAM_ucosii.qsys — für Cyclone III-Board (mit integriertem Flash, NIOS-Resetvektor=SRAM, für µCOSII)

- DE0_NANO_FLASH_Eval_Board.sopcinfo
- DE0_NANO_FLASH_SDRAM_PLL_Eval_Board.sopcinfo
- CycIII_Eval_ResVecFlash_memSRAM_ucosii. sopcinfo
- CycIII_Eval_ResVecSRAM_memSRAM_ucosii. sopcinfo

- DE0_NANO_FLASH_Eval_Board.qip — Konfiguration DE0-Nano, mit/ohne Flash-Board, ohne PLL für SDRAM
- DE0_NANO_FLASH_SDRAM_PLL_Eval_Board.qip — Konfiguration DE0-Nano, mit/ohne Flash-Board, mit PLL für SDRAM
- CycIII_Eval_ResVecFlash_memSRAM_ucosii.qip — Konfiguration Cyclone III-Board (mit integriertem Flash)
- CycIII_Eval_ResVecSRAM_memSRAM_ucosii.qip — Konfiguration Cyclone III-Board (mit integriertem Flash)

- nios_interface_shift_leds_hw.tcl — Nios-User-Component für LED-Ansteuerung

7.1.2 Quartus Setting File

- DE0_CycIII_Eval_Board.qsf — Temporäres Settings-File (wird für jede Konfiguration überschrieben!!)

- top_level_shift_leds_DE0_Nano.qsf — Konfiguration für DE0-Nano, ohne Nios
- top_level_shift_leds_CycIII.qsf — Konfiguration für Cyclone III, ohne Nios

- DE0_Nano_Flash_Eval_Board.qsf — Konfiguration für DE0-Nano, mit Nios, mit/ohne Flash-Board, mit/ohne PLL für SDRAM
- CycIII_Eval_Board.qsf — Konfiguration für Cyclone III, mit Nios

Hinweis:
Beim Importieren sind ALLE Assignments zu entfernen und dann ALLE wieder mit Global Settings zu importieren, damit die Chip-Einstellungen korrekt sind!

7.1.3 Pin Assignments

- pinning_shift_leds_DE0_Nano.csv — Pinbelegung für DE0-Nano, ohne Nios
- pinning_shift_leds_CycIII.csv — Pinbelegung für Cyclone III, ohne Nios

- pinning_DE0_Nano_Flash_Eval_Board.csv — Pinbelegung für DE0-Nano, mit Nios, mit/ohne Flash-Board, mit/ohne PLL für SDRAM
- pinning_CycIII_Eval_Board.csv — Pinbelegung für Cyclone III, mit Nios

7.1.4 Quartus VHDL / TopLevel

- top_level_shift_leds_DE0_Nano.vhd — Top-Level für DE0-Nano, ohne Nios
- top_level_shift_leds_CycIII.vhd — Top-Level für Cyclone III, ohne Nios

- top_level_nios_shift_leds_DE0_Nano_Flash.vhd — Top-Level für DE0-Nano, mit Nios, (für mit/ohne Flash-Board und mit/ohne PLL anpassen)
- top_level_nios_shift_leds_CycIII.vhd — Top-Level für Cyclone III, mit Nios

- shift_leds.vhd — LED-Ansteuermodul
- nios_interface_shift_leds.vhd — Nios-Interface für LED-Ansteuermodul

7.1.5 Quartus-Programmierfiles

- DE0_CycIII_Eval_Board.sof — Temporäres Programmierfile (wird für jede Konfiguration überschrieben!!)
- DE0_CycIII_Eval_Board.pof — Temporäres Programmierfile (wird für bestimmte Konfigurationen überschrieben!!)

- top_level_shift_leds_DE0_Nano_prog.sof — Top-Level für DE0-Nano, ohne Nios, auch für Eval-Board-HW-Test
- top_level_shift_leds_DE0_Nano_prog.jic — Top-Level für DE0-Nano, ohne Nios, auch für Eval-Board-HW-Test
- top_level_shift_leds_DE0_Nano_jic_conversion.cof — Setup zur Jic-Conversion (**J**TAG **I**ndirect **C**onfiguration für EPC16-Progr.)

- top_level_shift_leds_CycIII_prog.sof — Top-Level für Cyclone III, ohne Nios, auch für Eval-Board-HW-Test
- top_level_shift_leds_CycIII_prog.pof — Top-Level für Cyclone III, ohne Nios, auch für Eval-Board-HW-Test

- DE0_Nano_ohne_Flash_SDRAM_PLL_Eval_Board_prog.sof — für DE0-Nano mit Nios, ohne Flash-Board, mit PLL für SDRAM

- DE0_Nano_ohne_Flash_SDRAM_PLL_Eval_Board_prog.jic — für DE0-Nano mit Nios, ohne Flash-Board, mit PLL für SDRAM
- DE0_Nano_ohne_Flash_SDRAM_PLL_Eval_Board_jic_conversion.cof — Setup zur Jic-Conversion (**J**TAG **I**ndirect **C**onfig für EPC16-Progr.)

- DE0_Nano_mit_Flash_SDRAM_PLL_Eval_Board_prog.sof — für DE0-Nano mit Nios, mit Flash-Board, mit PLL für SDRAM
- DE0_Nano_mit_Flash_SDRAM_PLL_Eval_Board_prog.jic — für DE0-Nano mit Nios, mit Flash-Board, mit PLL für SDRAM
- DE0_Nano_mit_Flash_SDRAM_PLL_Eval_Board_jic_conversion.cof — Setup zur Jic-Conversion (**J**TAG **I**ndirect **C**onfig für EPC16-Progr.)

- CycIII_Eval_ResVecFlash_memSRAM_ucosii_prog.sof — für Cyclone III mit Nios, ResetVektor = Flash
- CycIII_Eval_ResVecFlash_memSRAM_ucosii_prog.pof — für Cyclone III mit Nios, ResetVektor = Flash
- CycIII_Eval_ResVecSRAM_memSRAM_ucosii_prog.pof — für Cyclone III mit Nios, ResetVektor = SRAM
- CycIII_Eval_Board_pof_conversion.cof — Setup zur pof-Conversion (Active Serial Programming Mode)

7.1.6 NIOS-Programmierfiles

- DE0_NANO_ohne_Flash_SDRAM_PLL_EVAL.elf — Temporäres Nios-Programmierfile (wird beim Compilieren überschrieben)
- DE0_NANO_ohne_Flash_SDRAM_PLL_EVAL_prog.elf — Nios-Programmierfile, ohne Flash-Board, mit PLL für SDRAM (anderes BSP)

- DE0_NANO_mit_Flash_SDRAM_PLL_EVAL.elf — Temporäres Nios-Programmierfile (wird beim Compilieren überschrieben)
- DE0_NANO_mit_Flash_SDRAM_PLL_EVAL_prog.elf — Nios-Programmierfile, mit Flash-Board, mit PLL für SDRAM (anderes BSP)

- CycIII_Eval.elf — Temporäres Nios-Programmierfile (wird beim Compilieren überschrieben)
- CycIII_Eval_ResVecFlash_memSRAM_prog.elf — Nios-Programmierfile (ResetVektor=Flash, ohne µCOSII)

- CycIII_Eval_ucosii.elf — Temporäres Nios-Programmierfile (wird beim Compilieren überschrieben)
- CycIII_Eval_ResVecFlash_memSRAM_ucosii_prog.elf — Nios-Programmierfile (ResetVektor=Flash, mit µCOSII)
- CycIII_Eval_ResVecSRAM_memSRAM_ucosii_prog.elf — Nios-Programmierfile (ResetVektor=SRAM, mit µCOSII)

- CycIII_Eval_prog_TdoT_2012.elf — Nios-Programmierfile speziell zum Tag der offen Tür 2012 (andere LCD-Texte)

(Die C-Source-Dateien werden beim Nios-Abschnitt beschrieben)

7.2 Programmier-Interfaces

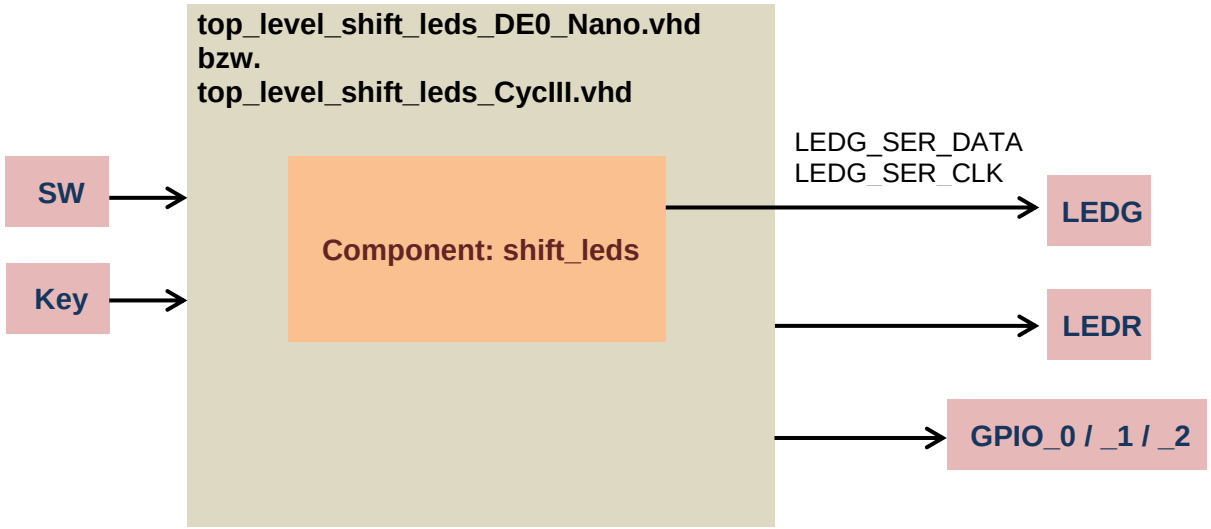
Die Programmierung der verschiedenen Aufsteckboards ist etwas unterschiedlich.

DE0-Nano	*.sof	JTAG-Mode	Mini-USB am DE0_Nano	
DE0-Nano	*.jic	JTAG-Mode	Mini-USB am DE0_Nano	(die jic-Files aus den sof-Files konvertieren und im JTAG-Mode in das Config-Device laden)
CycloneIII	*.sof	JTAG-Mode	J2 am CycIII-Board	
CycloneIII	*.pof	Active Serial Mode	J3 am CycIII-Board	
DE0-Nano	*.elf	JTAG-Mode	Mini-USB am DE0_Nano	
CycloneIII	*.elf	JTAG-Mode	J2 am CycIII-Board	

Bei neuem DE0-Nano bzw. CycloneIII-Board oder bei unbekanntem Software-Inhalt:
Das Board VOR dem Einstecken in das Eval-Board programmieren, da sonst durch vorhandene Softwareaktionen Schäden entstehen könnten !!!!!!

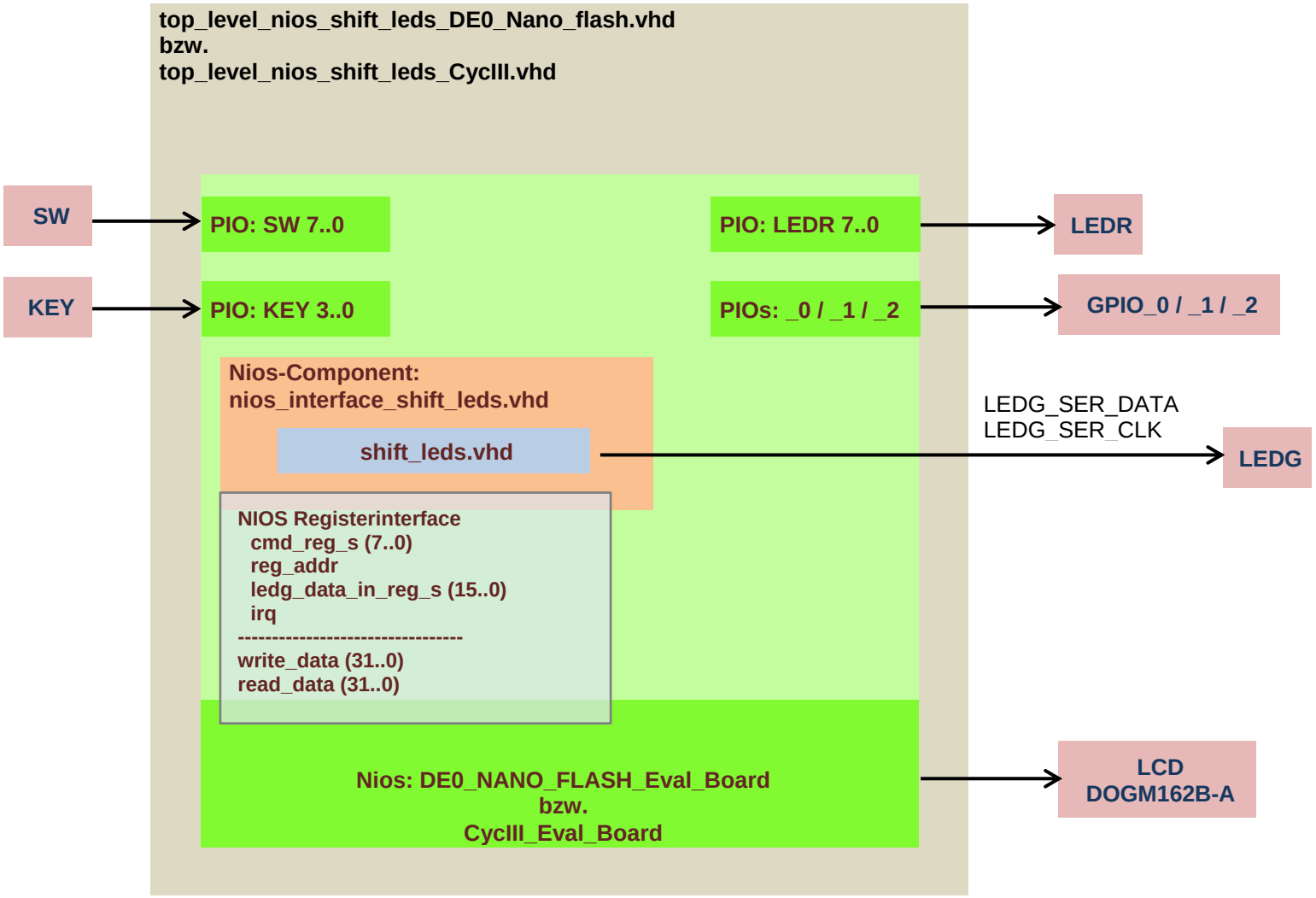
8 Grundsätzliche Funktion

8.1 VHDL-stand-alone (ohne Nios)



- Implementierung von reinen Demo- und Testfunktionen für das Eval-Board
- Abfrage der Schiebeschalter und Taster
- Ausgabe von Datenmustern an die LEDs und die GPIO-Leitungen (abhängig von der gewählten Konfiguration)
- GPIO-IN-Leitungen erzeugen spezielle Muster an den roten LEDs und müssen zum Test manuell belegt werden
- Aktionen für Taster
  Key 0: Start eines Bit-Shiftings zum Setzen der 16 grünen LEDs
  Key 1: Schiebeschalter-Stellung an die 8 roten LEDs durchschalten
  Key 2: GPIO-Datenmuster auf die GPIO-Ports durchschalten
  Key 3: Ändern der Taktrate der GPIO-Datenmuster

8.2 Register-Interface und Nios-Component



- Implementierung von reinen Demo- und Testfunktionen für das Eval-Board, **mit** Nios-Unterstützung
- Abfrage der Schiebeschalter und Taster
- Ausgabe von Datenmustern an die LEDs und die GPIO-Leitungen (abhängig von der gewählten Konfiguration)
- GPIO-IN-Leitungen müssen zum Test manuell belegt werden und erzeugen spezielle Muster, das am LCD angezeigt wird
- Aktionen für Taster
  Key 0: Start eines Bit-Shiftings zum Setzen der 16 grünen LEDs
  Key 1: Schiebeschalter-Stellung an die 8 roten LEDs durchschalten
  Key 2: GPIO-Datenmuster auf die GPIO-Ports durchschalten
  Key 3: Ändern der Taktrate der GPIO-Datenmuster

9 Beschreibung der VHDL-Module

Die TopLevel-Files für stand-alone-Demo (OHNE Nios) beinhalten den VHDL-Code zur Ansteuerung und zum Test der Board-HW.
Die TopLevel-Files für Nios-Demo dienen als Basis für eigene Projekte und beinhalten im Wesentlichen nur das PortMap für die NIOS-Components und die Hardware. (siehe dazu auch die DoxyGen-Dokumentation und separate Projekt-Templates im „Nios-Baukasten“)

9.1 Erforderliche Module

Die folgenden VHDL-Module sind für den Betrieb des Eval-Boards **zwingend erforderlich.**

9.1.1 shift_leds.vhd

Dieses Modul steuert die Ausgabe der Datenbits an die grünen LEDs auf Board-HW-Ebene und ist für **jede** Konfiguration des Eval-Boards (also **mit / ohne**) Nios erforderlich.

- Ansteuerung der 16 gruenen LEDs per Schieberegister über LEDG_SER_CLK und LEDG_SER_DATA (MSB first)
- Einlesen des parallelen LED-Datenwortes beim Start-Signal
- serielle Ausgabe an die Eval-Board-Schieberegister (MSB first)
- Handshake mit Top-Level:
  LEDG_SHIFT_STARTStartsignal vom Top-Level, Beginn des Shiftings, high-aktiv
  LEDG_SHIFT_BUSYAnzeige eines gerade laufenden Shiftings an den Top-Level, high-aktiv, wird eigenstaendig zurueckgesetzt nach Ende der Uebertragung
- Nach Setzen des Busy-Signals MUSS Top-Level die Start-Anforderung zuruecksetzen !!!
- Es wird immer ein kompletter Shift-Zyklus ausgefuehrt, danach wird die Start-Bedingung erneut geprueft
- Einstellen der Taktrate des ser. Daten-Clocks ueber Konstante clk_div_max (gibt die Zaehlweite fuer high/low-Phase an)
- Nach dem Reset wird jedesmal automatisch eine 0-Folge geschrieben, um evtl. gesetzte LEDs zu loeschen

9.1.2 nios_interface_shift_leds.vhd

Registerinterface für Nios-Konfigurationen

- NIOS-Anbindung zur Ansteuerung der 16 gruenen LEDs per Schieberegister
- wickelt einen kompletten Shift-Zyklus ab
- Einbindung des Moduls shift_leds.vhd
- Handshake mit shift_leds.vhd:
  - ledg_shift_start_s Start-Signal fuer Beginn des Shiftings
    Nach Setzen des Busy-bits durch shift_leds.vhd MUSS das Start-Signal zurueckgesetzt werden!!!
  - ledg_shift_busy_s Anzeige eines gerade laufenden Shiftings
    (wird von shift_leds.vhd eigenstaendig gesetzt und nach Ende der Uebertragung wieder zurueckgesetzt)
- NIOS-Registerinterface:
  addr NIOS-Adresse 0 Command-Register (Read/Write)
  1 Daten-Register fuer LEDs (Write)
  cmd_reg_s Command-Register
  ledg_data_in_reg_s Daten-Register fuer LEDs

- Struktur des cmd_reg_s:

bit	7	6	5	4	3	2	1	0
					busy	IRQ	IE	ShiftStart

ShiftStart: '1' vom NIOS startet Uebertragung durch shift_leds.vhd, wird automatisch zurueckgesetzt, sobald ledg_shift_busy_s aktiv (='1') ist
IE: Interrupt enable mit '1' vom NIOS
IRQ: Interrupt Request an NIOS, sobald Shift fertig ist (also busy '1'--->'0'), sofern IE enabled
muss vom NIOS nach Abholen des Interrupts zurückgesetzt werden
busy: '1' wenn Shift läuft, '0' wenn Shift fertig ist (entspricht ledg_shift_busy_s von shift_leds.vhd)
(Es macht keinen Sinn, vom NIOS darauf zu schreiben)

9.2 Empfohlene Module

Die folgenden VHDL-Module werden als Basis für eigene Projekte empfohlen (siehe auch separate Projekt-Templates im „Nios-Baukasten“)

9.2.1 top_level_shift_leds_DE0_Nano.vhd

Test-Top-Level-File für stand-alone-Konfiguration mit DE0-Nano (**OHNE** Flashboard)
Das Flashboard wird NICHT verwendet, weil sonst der GPIO-0-Port komplett wegfällt und der HW-Board-Test nur sehr unvollständig möglich wäre.
Bei „richtigen“ Applikationen wird das Flashboard wohl meistens verwendet werden.

- Test-Top-Level-File als VHDL stand-alone-Modul, ohne Nios
- Ansteuerung der 16 gruenen LEDs per Schieberegister
- Einbindung des Moduls shift_leds.vhd
- LED-Aktivierung über Schiebeschalter SW7...SW0
- 8 rote LEDs parallel zu Schalter
- keine LCD-Ansteuerung
- GPIO-Ausgänge mit GPIO-Muster-Signalen belegt
- GPIO-Eingänge:
  GPIO_0_IN0 = '0' ----> alle LEDR on
  GPIO_0_IN1 = '0' ----> alle LEDR on
  GPIO_1_IN0 = '0' ----> jede zweite LEDR on
- KEY0: Auslösen eines LEDG-Shiftings
- KEY1: Schalterstellung an LEGR ausgeben
- KEY2: GPIO-Muster-Signale ausgeben
- KEY3: Taktrate der GPIO-Muster-Signale ändern

9.2.2 top_level_shift_leds_CycIII.vhd

Test-Top-Level-File für stand-alone-Konfiguration mit CycloneIII-Board

- Test-Top-Level-File als VHDL stand-alone-Modul, ohne Nios
- Ansteuerung der 16 gruenen LEDs per Schieberegister
- Einbindung des Moduls shift_leds.vhd
- LED-Aktivierung über Schiebeschalter SW7...SW0
- 8 rote LEDs parallel zu Schalter
- keine LCD-Ansteuerung
- GPIO-Ausgänge (ExpH_x) mit GPIO-Muster-Signalen belegt
- GPIO-Eingänge:
  D_OE_IN = '0' ----> alle LEDR on
  D_CLRn_IN = '0' ----> alle LEDR off
- KEY0: Auslösen eines LEDG-Shiftings
- KEY1: Schalterstellung an LEGR ausgeben
- KEY2: GPIO-Muster-Signale ausgeben
- KEY3: Taktrate der GPIO-Muster-Signale ändern

9.2.3 top_level_nios_shift_leds_DE0_Nano_Flash.vhd

Test-Top-Level-File für Nios-Konfiguration mit DE0-Nano (**MIT/OHNE** Flashboard, **MIT/OHNE** SDRAM-PLL, OHNE µCOSII)

- Konfiguration für DE0-Nano **MIT** Flash-Board ---> d.h. GPIO-0 ist **NICHT** verfügbar
- Ansteuerung der 16 gruenen LEDs per Schieberegister über Einbindung des Moduls shift_leds.vhd
- GPIOs als Muster-I/O, ggf. müssen die Nios-Pios geändert werden
- allg. Board-Funktionen (z.B. Schalter, Taster, LCD) werden vom Nios gesteuert

Das SDRAM auf dem DE0-Nano kann mit bzw. ohne PLL betrieben werden. (Alle Altera-Beispiele arbeiten MIT PLL)

Das File ist entsprechend vorbereitet, die gekennzeichneten Stellen müssen für die Verwendung mit/ohne Flashboard und mit/ohne SDRAM-PLL entsprechend aus- bzw. einkommentiert werden.

9.2.4 top_level_nios_shift_leds_CycIII.vhd

Test-Top-Level-File für Nios-Konfiguration mit CycloneIII-Board (MIT/OHNE µCOSII)

- Konfiguration für CycloneIII
- Ansteuerung der 16 gruenen LEDs per Schieberegister über Einbindung des Moduls shift_leds.vhd
- GPIOs als Muster-I/O, ggf. müssen die Nios-Pios geändert werden
- allg. Board-Funktionen (z.B. Schalter, Taster, LCD) werden vom Nios gesteuert

Das File ist entsprechend vorbereitet, die gekennzeichneten Stellen müssen für die Verwendung mit/ohne Flashboard und mit/ohne SDRAM-PLL entsprechend aus- bzw. einkommentiert werden.

10 NIOS-Software

Diese Nios-Software dient rein zu Test- und Demozwecken am Eval-Board und erfüllt keine „sinnvolle“ Applikation!
Bei Nutzung des Eval-Boards für „Rapid Prototyping“ ist es sinnvoll, die verfügbaren Ressourcen bzw. Module des Gesamtprojekts zu verwenden bzw. darauf aufzubauen!
ABER: Vorsicht bei HW-Anpassungen !!!!! Es droht der „Tod des Eval-Boards“

Name	Description
⊟ **clk**	Clock Source
clk_in	Clock Input
clk_in_reset	Reset Input
clk	Clock Output
clk_reset	Reset Output
⊟ **sysid_qsys**	System ID Peripheral
clk	Clock Input
reset	Reset Input
control_slave	Avalon Memory Mapped Slave
⊟ **jtag_uart**	JTAG UART
clk	Clock Input
reset	Reset Input
avalon_jtag_slave	Avalon Memory Mapped Slave
⊟ **nios2_qsys**	Nios II Processor
clk	Clock Input
reset_n	Reset Input
data_master	Avalon Memory Mapped Master
instruction_master	Avalon Memory Mapped Master
jtag_debug_module_reset	Reset Output
jtag_debug_module	Avalon Memory Mapped Slave
custom_instruction_master	Custom Instruction Master
⊟ **sdram**	SDRAM Controller
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
wire	Conduit
⊟ **SDRAM_PLL**	Avalon ALTPLL
inclk_interface	Clock Input
inclk_interface_reset	Reset Input
pll_slave	Avalon Memory Mapped Slave
c0	Clock Output
c1	Clock Output
areset_conduit	Conduit
locked_conduit	Conduit
phasedone_conduit	Conduit
⊟ **flash_controller**	Generic Tri-State Controller
clk	Clock Input
reset	Reset Input
uas	Avalon Memory Mapped Slave
tcm	Tristate Conduit Master
⊟ **flash_pin_sharer**	Tri-State Conduit Pin Sharer
clk	Clock Input
reset	Reset Input
tcm	Tristate Conduit Master
tcs0	Tristate Conduit Slave
⊟ **flash_conduit_bridge**	Tri-State Conduit Bridge
clk	Clock Input
reset	Reset Input
tcs	Tristate Conduit Slave
out	Conduit

⊟ **comp_nios_interface_shift_leds**	nios_interface_shift_leds
avalon_slave_0	Avalon Memory Mapped Slave
conduit_end	Conduit
clock_sink	Clock Input
reset_sink	Reset Input
⊟ **key3_0**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint
⊟ **switch7_0**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint
⊟ **LEDR7_0**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint
⊟ **lcd**	Character LCD
clk	Clock Input
reset	Reset Input
control_slave	Avalon Memory Mapped Slave
external	Conduit Endpoint
⊟ **GPIO_1_IN0**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint
⊟ **GPIO_12_out**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint
⊟ **GPIO_2_IN2_0**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint
⊟ **GPIO_2_out_12_0**	PIO (Parallel I/O)
clk	Clock Input
reset	Reset Input
s1	Avalon Memory Mapped Slave
external_connection	Conduit Endpoint

Die Ressourcen der NIOS-Projekte korrspondieren mit dem QSys-Aufbau.

Sofern QSys aus den Projekt-Dateien als Basis abgeleitet wird, können die Ressourcen einfach übernommen werden.

Andernfalls sind ggf. Namensanpassungen in den SW-Modulen durchzuführen.

Für den Einsatz von µCOSII ist noch ein zusätzlicher Interval-Timer (os_timer) erforderlich.

10.1 Nios-Demo-Projekte

Für das Eval-Board existieren mehrere Nios-Demo-Projekte, es handelt sich aber nur um eine exemplarische Auswahl:

• DE0-Nano-Board (**mit** Flashboard, **mit** SDRAM-PLL, **ohne** µCOSII)	DE0_NANO_mit_FLASH_SDRAM_PLL_Eval DE0_NANO_mit_FLASH_SDRAM_PLL_Eval_bsp	(C-Software) (HAL)
• DE0-Nano-Board (**ohne** Flashboard, **mit** SDRAM-PLL, **ohne** µCOSII)	DE0_NANO_ohne_FLASH_SDRAM_PLL_Eval DE0_NANO_ohne_FLASH_SDRAM_PLL_Eval_bsp	(C-Software) (HAL)
• CycloneII-Board (**ohne** µCOSII)	CycIII_Eval CycIII_Eval_bsp	(C-Software) (HAL)
• CycloneII-Board (**mit** µCOSII)	CycIII_Eval_ucosii CycIII_Eval_ucosii_bsp	(C-Software) (HAL)

Im jeweiligen C-Software-Projektteil finden sich folgende Source-Files:

- **main.c* (abhängig von Konfiguration)
  - einfaches Testprogramm mit allen wesentlichen Bausteinen zur Steuerung des Eval-Boards
  - Abfrage der Schalter und Taster
  - Ansteuerung der roten LEDs
  - Steuerung des Bit-Shiftings für die grünen LEDs
  - Durchschalten der GPIO-Mustersignale auf die Port-Ausgänge
  - Einlesen der GPIO-Eingangssignale und Anzeige am LCD
  - Ändern der Taktrate der GPIO-Mustersignale
  - mit 1 Musterkonfiguration für µCOSII (Projekt CycIII_Eval_ucosii)

- dazugehörige Header-Dateien **main.h* (abhängig von Konfiguration)

- *lcd_test_de0_nano.**c***
  - Ansteuerung des LCDs mit Testdaten

- dazugehörige Header-Datei *lcd_test_de0_nano.**h***

- Header-Datei *gpio_de0_nano.**h*** bzw. *gpio_cycIII.**h*** (abhängig von Konfiguration)
  - mit Makrofunktionen zur Abfrage und zum Setzen der GPIO-Ports

- **Das Projekt *CycIII_Eval_ucosii* verwendet den Betriebssystem-Kernel µCOSII, alle anderen Demo-Projekte sind „klassisch“ programmiert !**

10.2 Eval-Board-spezifische Ressourcen

Im Unterverzeichnis *„eval_board_common_c_sources“* finden sich in jedem Projekt die **gemeinsamen Eval-Board-spezifischen** C-Sourcen für alle Konfigurationen.
(Dieses Unterverzeichnis ist in alle Projektteile einkopiert sein, wird von DoxyGen aber nur 1x ausgewertet*)*

Für eine User Application mit dem Eval-Board sind einige Ressourcen zwingend erforderlich, andere sind Empfehlungen für Rapid Prototyping.
Diese Ressourcen korrespondieren mit dem SoC bzw. QSys-Aufbau, ggf. sind aufgrund anderer QSys-Bezeichnungen Anpassungen erforderlich.
Bei Übernahme der QSys-Dateien als Basis für eine User Application sollten diese Ressourcen auch direkt übernommen werden können.

10.2.1 Erforderliche Ressourcen

- Header-Datei *general.**h***
  Hier muss per define eingestellt werden, ob µCOSII verwendet wird oder nicht.

- *lcd_dogm_162_A.**c***
  - Funktionen zur Ansteuerung des LCDs (2 x 16), (Erweiterung gegenüber dem Altera LCD-IP-Core)
- dazugehörige Header-Datei *lcd_dogm_162_A.**h***

- *nios_interface_shift_leds.**c***
  - Funktionen zur Ansteuerung der grünen LEDs (Bit-Shifting, mit IRQ)
- dazugehörige Header-Datei *nios_interface_shift_leds.**h***

10.2.2 Empfohlene Ressourcen

- *ledr.**c***
  - Funktionen zur Ansteuerung der roten LEDs
- dazugehörige Header-Datei *ledr.**h***

- Header-Datei *sw_key.**h***
  - Makrofunktionen zur Abfrage der Schalter und Taster

(siehe dazu auch die DoxyGen-Dokumentation)

11 Inbetriebnahme und Board-Test, Prüfanweisung

Der Hardware-Test eines Eval-Boards wird in 2 Stufen durchgeführt:

- Stufe 1: ohne LCD, mit Testprogramm auf VHDL-Ebene (OHNE Nios) mit DE0-Nano-Board, weil da die maximale Zahl der IO-Pins zum Test zur Verfügung steht.
  Das DE0-Nano-Board muss OHNE Flashboard betrieben werden, sonst fehlen die GPIO-0-Ports !
- Stufe 2: mit LCD, mit CycloneIII-Board

Die Portdefinition in den Testprogrammen, sprich: Input / Output, ist willkürlich und praktischerweise auf OUT eingestellt.
Ausnahme: Die fixen INPUTS der Cyclone-Chips stehen auf dem Eval-Board natürlich ebenfalls nur als IN zur Verfügung.

11.1 Stufe 1: Test der Board-HW (mit DE0-Nano, ohne LCD)

11.1.1 Test-File

Zur Inbetriebnahme und zum Test ist folgendes File zu laden:

a) top_level_shift_leds_DE0_Nano_prog.sof — Top-Level für DE0-Nano, ohne Nios (Quartus-System erforderlich)

oder

b) top_level_shift_leds_DE0_Nano_prog.jic — Top-Level für DE0-Nano, ohne Nios (Programmierfile für autarken Betrieb)

Sicherheitshinweis:
Bei unbekanntem Programm- bzw. SW-Inhalt muss das DE0-Nano-Board VOR dem Stecken auf das Eval-Board ZUERST mit dem jic-File geladen werden, damit keine Schäden durch falsche I/O-Belegung entstehen.

11.1.2 Einschalten

Versorgung des Eval-Boards per USB-Kabel:
bei a) USB an DE0-Nano verwenden
bei b) es kann auch USB auf Eval-Board verwendet werden

ggf. das SOF-File über Quartus laden

Reset-Button (Key4) drücken ---> alle LEDR off, evtl. werden LEDG „auf off geschoben“.

11.1.3 Schiebeschalter und rote LEDs

Mit Schiebeschalter verschiedene Datenmuster einstellen ---> mit Key1 werden die Muster in LEDR übernommen, alle LEDR müssen mal geleuchtet haben.

11.1.4 grüne LEDs

Die 16 grünen LEDs sind ein „doppeltes gespiegeltes“ Abbild des LEDR-Datenmusters.
Die LEDR 7..0 werden als LEDR 0..7 auf die LEDG-Position 15..8 geschoben.
Mit Schiebeschalter Datenmuster einstellen ---> mit Key0 wird das LEDG-Shifting angestoßen.
Alle LEDG müssen mal geleuchtet haben.
(Es ist normal, dass während des Drückens von Key0 alle LEDs schwach leuchten)

Beispiel:
LEDR: 00000001 ----> LEDG: 10000000|00000001

11.1.5 Eingangs-Ports

Die Funktion der fixen IN-Ports kann mittels des Testprogrammes am LEDR-Verhalten abgelesen werden. Dazu sind die Ports mittels Drahtbrücke wahlweise auf 3,3V bzw. GND zu legen.
Diese Signale stehen an den Steckern IO_ExpH_2a/b, AN_ExpH und LEDR_ExpH zur Verfügung:
jeweils Pin1: GND
jeweils Pin3: 3,3V

11.1.5.1 Eingänge an IO_ExpH_0

Pin 1 ---> 3,3V (GPIO_1_IN0)
dann an Schiebeschalter Datenmuster einstellen und mit Key1 übernehmen
Pin 1 ---> GND ---> alle LEDR on
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster
Pin 1 ---> 3,3V
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster

Pin 3 ---> 3,3V (GPIO_0_IN1)
dann an Schiebeschalter Datenmuster einstellen und mit Key1 übernehmen
Pin 3 ---> GND ---> alle LEDR off
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster
Pin 3 ---> 3,3V
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster

11.1.5.2 Eingänge an IO_ExpH_2a

Pin 4 ---> 3,3V (GPIO_2_IN1)
dann an Schiebeschalter Datenmuster einstellen und mit Key1 übernehmen
Pin 4 ---> GND ---> alle LEDR off
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster

11.1.5.3 Eingänge an IO_ExpH_2b

Pin 4 ---> 3,3V (GPIO_2_IN2)
dann an Schiebeschalter Datenmuster einstellen und mit Key1 übernehmen
Pin 4 ---> GND ---> jede 2. LEDR on
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster
Pin 4 ---> 3,3V
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster

Pin 6 ---> 3,3V (GPIO_2_IN0)
dann an Schiebeschalter Datenmuster einstellen und mit Key1 übernehmen
Pin 6 ---> GND ---> jede andere 2. LEDR on
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster
Pin 6 ---> 3,3V
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster

11.1.6 Ausgangs-Ports

Das Testprogramm erzeugt für die (willkürlich) als Ausgang definierten Ports verschiedene Datenmuster (Rechtecksignale).
Es werden folgende Signale verteilt an den Pins ausgegeben:
ca. 98kHz
ca. 48kHz
ca. 24kHz
ca. 12kHz
ca. 6kHz

Zur Ausgabe der Rechtecksignale muß Key2 gedrückt werden!
Durch gleichzeitiges Drücken von Key3 wird die Frequenz des Signals verdoppelt.

11.1.6.1 IO_ExpH_0

IO_ExpH_0					
Signal (kHz)	**IO_ExpH_0**	**Pin**	**Pin**	**IO_ExpH_0**	**Signal (kHz)**
---	**GPIO-1-IN0**	1	2	GPIO-00	98
---	GPIO-0-IN1	3	4	GPIO-01	48
6	**GPIO_12**	5	6	GPIO-03	12
98	GPIO-04	7	8	GPIO-05	48
24	GPIO-06	9	10	GPIO-07	12
---	VCC-Sys	11	12	Gnd	---
98	GPIO-08	13	14	GPIO-09	48
24	GPIO-010	15	16	GPIO-011	12
98	GPIO-012	17	18	GPIO-013	48
24	GPIO-014	19	20	GPIO-015	12
98	GPIO-016	21	22	GPIO-017	48
24	GPIO-018	23	24	GPIO-019	12
98	GPIO-020	25	26	GPIO-021	48
24	GPIO-022	27	28	GPIO-023	12
---	VCC_33	29	30	Gnd	---
98	GPIO-024	31	32	GPIO-025	48
24	GPIO-026	33	34	GPIO-027	12
98	GPIO-028	35	36	GPIO-029	48
24	GPIO-030	37	38	GPIO-031	12
98	GPIO-032	39	40	GPIO-033	48

11.1.6.2 IO_ExpH_2a

IO_ExpH_2a					
Signal (kHz)	**IO_ExpH_2a**	**Pin**	**Pin**	**IO_ExpH_2a**	**Signal (kHz)**
---	Gnd	1	2	VCC_50	---
---	VCC_33	3	4	GPIO-2-IN1	---
98	GPIO-20	5	6	GPIO-210	24
24	GPIO-22	7	8	GPIO-212	98
98	GPIO-24	9	10	Gnd	---
24	GPIO-26	11	12	Gnd	---
98	GPIO-28	13	14	Gnd	---

11.1.6.3 IO_ExpH_2b

IO_ExpH_2b					
Signal (kHz)	**IO_ExpH_2b**	**Pin**	**Pin**	**IO_ExpH_2b**	**Signal (kHz)**
---	Gnd	1	2	VCC_50	---
---	VCC_33	3	4	GPIO-2-IN2	---
12	GPIO-23	5	6	GPIO-2-IN0	---
48	GPIO-25	7	8	GPIO-21	48
12	GPIO-27	9	10	Gnd	---
48	GPIO-29	11	12	Gnd	---
12	GPIO-211	13	14	Gnd	---

11.1.7 LEDR_ExpH

Die LEDR können auch über diesen Stecker als Eingänge beschaltet werden, **SOFERN im Quartus diese auch als IN-Port definiert sind und ein ANDERES Testfile geladen wird.**
Mit dem in 11.1.1 geladenen Testfile kann dieser IN-Test NICHT durchgeführt werden.

Bei diesem IN-Port-Test wird nur die Leiterbahnverbindung getestet, indem die Ausgangssignale überprüft werden.

Dazu:
geeignetes anderes Testfile laden (Sonderfall)
an Schiebeschalter Datenmuster einstellen und mit Key1 übernehmen
an IO_ExpH_0: Pin 1 ---> GND (GPIO_1_IN0)
alle LEDR sind jetzt off
bei Drücken von Key1 erscheint wieder das vorherige Datenmuster
für jede leuchtende LEDR muß an LEDR_ExpH der entsprechende Pin ebenfalls auf 1 liegen

11.2 Stufe 2: Test des LCD

Der LCD-Test eines Eval-Boards wird mit einem Testprogramm **mit Nios** durchgeführt.

Dazu ein **CycloneIII-Board** mit der Konfiguration
„CycIII_Eval_ResVecFlash_memSRAM_ucosii_prog.pof“
und
„CycIII_Eval_ResVecFlash_memSRAM_ucosii_prog.elf“
laden**.**

Wenn **alle** Schiebeschalter ON sind, geht das LCD in den Testzyklus und schreibt periodisch Testdaten.

Wenn **nicht alle** Schiebeschalter ON sind, wird im LCD die Stellung der Schalter abgebildet (gespiegelt und erweitert wie bei den grünen LEDs).

am LCD:
Pegelanzeige GPIO_1_IN0 ---> Pin 1 von IO_ExpH_0 auf 3,3V bzw. GND

Diese 3 Signale gibt es erst ab Platinenversion CycloneIII V2.0
Pegelanzeige GPIO_2_IN0 ---> Pin 6 von IO_ExpH_2b auf 3,3V bzw. GND
Pegelanzeige GPIO_2_IN1 ---> Pin 4 von IO_ExpH_2a auf 3,3V bzw. GND
Pegelanzeige GPIO_2_IN2 ---> Pin 4 von IO_ExpH_2b auf 3,3V bzw. GND

Mit Key0 wird das Shifting für die grünen LEDs gestartet.
Mit Key1 wird die Schiebeschalter-Stellung in die roten LEDs übernommen.

11.2.1 Ergänzung: Test CycloneIII I/O-Ports

Da in dieser Teststufe bereits ein CycloneIII-Board verwendet wird, können sinnvollerweise auch gleich die entsprechenden I/O-Ports getestet werden.

Die Layout-Pfade auf dem Eval-Board wurden schon mit verschiedenen Rechtecksignalen in Teil 1 getestet.
Hier geht es dann nur noch um cen Test der prinzipiellen Funktion des CycloneIII-Steckers zum Eval-Board.
Dafür wird an allen u.g. Pins das gleiche Rechtecksignal im 200Hz-Bereich ausgegeben.

IO_ExpH_0					
Signal (kHz)	**IO_ExpH_0**	**Pin**	**Pin**	**IO_ExpH_0**	**Signal (kHz)**
---	D_OE_IN	1	---	---	---
---	D_CLRn_IN	3	4	ExpH_R_1	~0,2
~0,2	ExpH_B_20	5	6	ExpH_P_1	~0,2
~0,2	ExpH_AB_3	7	8	ExpH_N_1	~0,2
~0,2	ExpH_A_23	9	10	ExpH_M_1	~0,2