

Schriftliche Prüfung dt2

Kurs 1 und 2

Datum: 18. April 2016

21 Pkt

5.0

Zeit: 15:15 – 17:15

Name: _____

USB-Nr: _____

Bedingungen:

- Erlaubte Hilfsmittel: Unterrichtsunterlagen, VHDL-Buch und Übungen
- Die Prüfungsaufgaben 3 bis 5 sind lokal auf Ihrem Computer zu lösen. Machen Sie zu diesem Zweck auf Ihrem Rechner ein leeres VHDL-Projekt mit Ihrem **NameVorname** als Projektnamen und importieren Sie danach die Files/Directories vom USB-Stick. Am Ende der Prüfung ist das gesamte Projekt mit den Files
integrate.vhd
integrate_generics.vhd
top.vhd
signal_gen.vhd
simulation\force_integrate.do
- wieder auf den USB-Stick zu kopieren und abzugeben
- Setzen Sie als erstes Ihren Namen und Vornamen in die Dateien
- Nichtkomplilierbarer Code gibt maximal die Hälfte der möglichen Punktzahl
- Gegenseitiges Abschreiben in irgendeiner Form führt zu Note 1
- Die Beilage muss abgegeben werden
- Verboten ist jegliche Art von Kommunikation

Bewertung:

Die maximale Punktzahl beträgt 26 Punkte.

- 1) Unter welchen Umständen wird in einem PROCESS mit einem IF-Statement etwas gespeichert? Wann ist das gewünscht, wann nicht? (3 Punkte)

• falls der Process abhängig von c/k z.B. (`rising-edge(c/k)`)
→ alle Signale links des Zuweisungsoperators oder alle Variablen, welche gelesen werden, bevor sie beschrieben werden.

- 2) Zustandsmaschinen (FSM): Wieso nutzt man bei der Kodierung der Zustände bei FPGAs "one hot" und nicht z.Bsp. Binärkodierung? (2 Punkte) 1.5

FPGAs haben viele FlipFlops zur Verfügung, weshalb man aus Geschwindigkeitsoptimierungsgründen "one hot" codiert. Dies bedeutet, dass pro State nur 1 FF aktiv ist, alle anderen spielen dabei keine Rolle. Außerdem ist das System weniger störungsausfälliger.

- 3) Entwickeln Sie ein VHDL-Modell für einen Integrator, der die ankommenden Signale "tx" aufaddiert, wenn sie grösser als die Schwelle "threshold" sind. Die aufaddierten Daten sollen in einem unsigned -Vektor "integrate_value" gespeichert und gleichzeitig als "int_value" ausgegeben werden.
- J Überschreitet "integrate_value" die Schwelle "limit", wird der Error "int_error" ausgegeben (siehe Abbildung 1).
- Das Signal "rst_int" setzt "integrate_value" und "int_error" synchron zurück (siehe Abbildung 2).

Gegeben ist die Entity von "integrate" des Integrators:

Name	Typ	Bedeutung
rst_n	std_ulogic	Eingang: Asynchroner Reset, aktiv low.
clk	std_ulogic	Eingang: Systemclock, 100 MHz.
rst_int	std_ulogic	Eingang: Löscht den Integrator synchron, wenn '1'. Mindestens während einer Clockperiode aktiv.
tx	std_ulogic_vector (7 downto 0)	Eingang: Datenwort, das "integriert" wird.
threshold	std_ulogic_vector (7 downto 0)	Eingang: Schwelle ("threshold") für den Integrator: ist "tx" grösser als diese Schwelle, wird integriert. D.h. der Wert von "tx" wird zum aktuellen Integratorwert addiert.
limit	std_ulogic_vector (11 downto 0)	Eingang: Endwert des Integrators. Überschreitet der interne Integratorwert diesen Endwert, stoppt die Integration.
int_value	std_ulogic_vector (15 downto 0)	Ausgang: der aktuelle Integratorwert.
int_error	std_ulogic	Ausgang: Error, wenn der Endwert des Integrator überschritten wird.

Spezifikation (siehe auch Abbildung 1)

- Der asynchrone Reset "rst_n" setzt alle Register zurück auf den Wert 0 ✓
- Die Clockfrequenz beträgt 100 MHz ~~-dforce ...~~
- Alle Eingangssignale ausser "rst_n" sind clock-synchron
- Es sollen möglichst wenig Speicherelemente verwendet werden

- Timing:

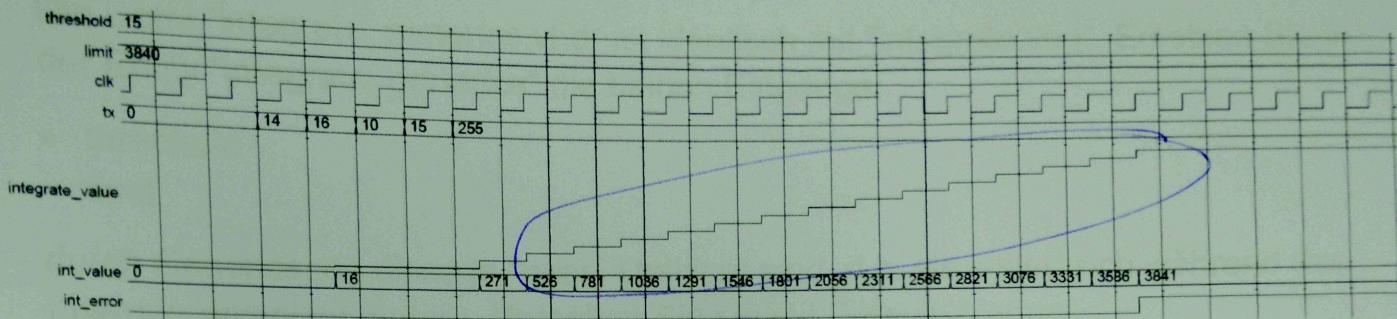


Abbildung 1

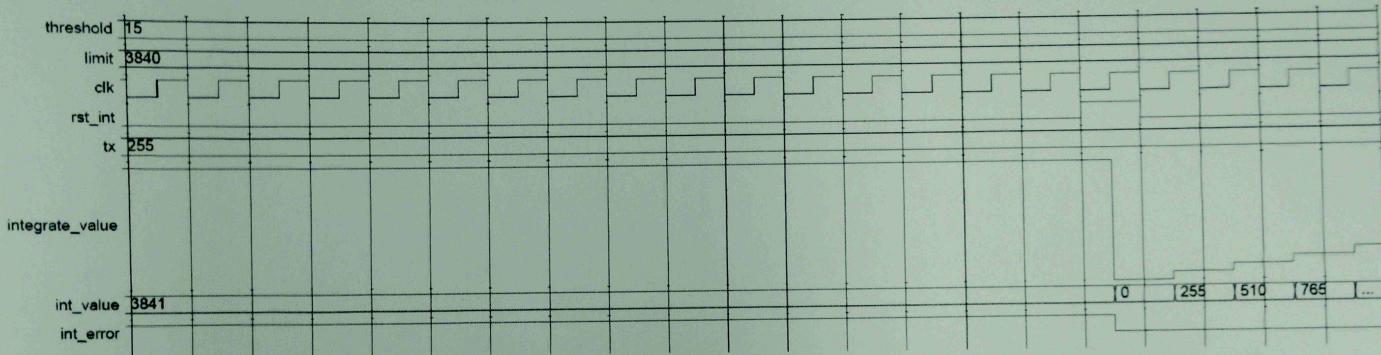


Abbildung 2

a) Datei *integrate.vhd*

(5 Punkte) 4.5

Ergänzen Sie den Code so, dass die oben definierten Spezifikationen und das Timing erfüllt sind (Abbildung 1 und 2).

b) Auf welchem Abstraktionslevel liegt diese Beschreibung?

(1 Punkt)

RTL → Beschreibend ~~→ also was es macht: v~~
Logik

c) Datei *simulation\force_integrate.do*

(3 Punkte)

Schreiben Sie ein force-File, das das geforderte Integrier-Verhalten überprüft

3

d) Datei *integrate_generics.vhd* ✓

(2 Punkte)

A.5

Die Datei "integrate_generics.vhd" ist identisch mit "integrate.vhd". Ersetzen Sie in der Entity "integrate_generics" die beiden Eingänge:

- threshold
- limit

durch Generics, da diese ja "quasi-statisch" sind, d.h. sie sollen sich während des Intergrierens nicht ändern. Versehen Sie sie mit Default-Werten

- 4) Implementieren Sie folgendes Blockdiagramm in VHDL (Abbildung 3). Die Entity von "top" sowie die Komponenten "a" bis "d" sind im importierten Projekt vorhanden:

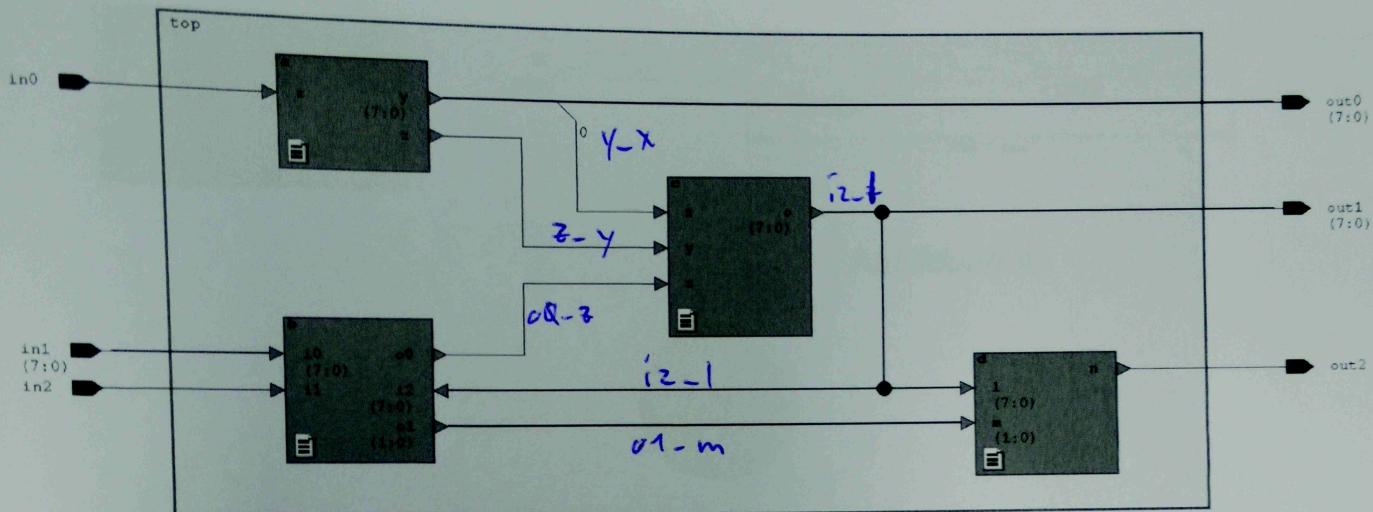


Abbildung 3

a) Datei *top.vhd*

(4.5 Punkte)

Verbinden Sie in der Architektur von "top" die vier Komponenten "a", "b", "c" und "d"

b) Um was für VHDL-Beschreibungsart handelt es sich?

(1 Punkt)

strukturaler code (mapping)

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;          0,5

entity integrate is
port(
    clk      : in std_ulogic;
    rst_n   : in std_ulogic;
    rst_int : in std_ulogic;
    tx       : in std_ulogic_vector(7 downto 0);
    threshold : in std_ulogic_vector(7 downto 0);
    limit    : in std_ulogic_vector(11 downto 0);
    int_value : out std_ulogic_vector(15 downto 0);
    int_error : out std_ulogic
);
end entity integrate;

```

4.5

```

architecture rtl of integrate is
    SIGNAL integrate_value : unsigned(15 downto 0);      0,5
begin

```

```

    p_integrate : process (rst_n,clk)
    begin
        if rst_n = '0' then
            integrate_value <= (Others => '0');           0,5
        elsif rising_edge(clk) then
            if rst_int = '1' then
                integrate_value <= (Others => '0'); } 0,5
            elsif tx > threshold then
                integrate_value <= integrate_value + resize(unsigned(tx),16); 0,5
            end if;
        end if;
    end process p_integrate;

    int_value <= std_ulogic_vector(integrate_value);
    int_error <= '1' when integrate_value > unsigned(limit) else '0'; 0,5
end architecture rtl;

```

AND <= limit

```

-- Name: Duerner
-- Vorname: Daniel

```

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

entity integrate is
generic(
    g_threshold : std_ulogic_vector(7 downto 0) := X"0F";
    g_limit : std_ulogic_vector(11 downto 0) := X"F00"
);
port(
    clk      : in std_ulogic;
    rst_n   : in std_ulogic;
    rst_int : in std_ulogic;
    tx       : in std_ulogic_vector(7 downto 0);
    --threshold : in std_ulogic_vector(7 downto 0);
    --limit    : in std_ulogic_vector(11 downto 0);
    int_value : out std_ulogic_vector(15 downto 0);
    int_error : out std_ulogic
);
end entity integrate;

```

1,5

```

    p_count : process (clk, rst) is
    begin
        if rst = '1' then
            cnt <= (OTHERS => '0');           0,5
        elsif rising_edge(clk) then
            cnt <= ((cnt + 1) mod 6); -- cnt: +3 Speicherelemente
        end if;
    end process p_count;

```

1 (anzen idle)

2,5

```

A:\data\ausbildung\bachelor_fhnw\module\dt2_digitaltechnik2\FS16\dt2_pruefungen\pruefung_1\studien
library ieee;
use ieee.std_logic_1164.all;

entity top is
  port(
    in0 : in std_ulogic;
    in1 : in std_ulogic_vector(7 downto 0);
    in2 : in std_ulogic;
    out0 : out std_ulogic_vector(7 downto 0);
    out1 : out std_ulogic_vector(7 downto 0);
    out2 : out std_ulogic
  );
end entity top;

architecture struct of top is
  component a_rtl
    port(x : in std_ulogic;
          y : out std_ulogic_vector(7 downto 0);
          z : out std_ulogic);
  end component a_rtl;

  component b_rtl
    port(i0 : in std_ulogic_vector(7 downto 0);
          i1 : in std_ulogic;
          i2 : in std_ulogic_vector(7 downto 0);
          o0 : out std_ulogic;
          o1 : out std_ulogic_vector(1 downto 0));
  end component b_rtl;

  component c_rtl
    port(x : in std_ulogic;
          y : in std_ulogic;
          z : in std_ulogic;
          o : out std_ulogic_vector(7 downto 0));
  end component c_rtl;

  component d_rtl
    port(l : in std_ulogic_vector(7 downto 0);
          m : in std_ulogic_vector(1 downto 0);
          n : out std_ulogic);
  end component d_rtl;

  -- Embedded configuration:
  FOR a_inst : a_rtl USE ENTITY work.a(rtl);
  FOR b_inst : b_rtl USE ENTITY work.b(rtl);
  FOR c_inst : c_rtl USE ENTITY work.c(rtl);
  FOR d_inst : d_rtl USE ENTITY work.d(rtl);

  -- Internal signal declaration:
  SIGNAL z_y : std_ulogic;
  SIGNAL o0_z : std_ulogic;
  SIGNAL o1_m : std_ulogic_vector(1 downto 0);

```

4.5

1

1.5

```

X:\data\ausbildung\bachelor_fhnw\module\dt2_digitaltechnik2\FS16\dt2_pruefungen\pruefungen.vhd
begin

    a_inst : a_rtl
    PORT MAP(
        x => in0,
        y => out0,
        z => z_y
    );
    0.5

    b_inst : b_rtl
    PORT MAP(
        i0 => in1,
        i1 => in2,
        o0 => o0_z,
        i2 => out1,
        o1 => o1_m
    );
    0.5

    c_inst : c_rtl
    PORT MAP(
        x => out0(0),
        y => z_y,
        z => o0_z,
        o => out1
    );
    0.5

    d_inst : d_rtl
    PORT MAP(
        l => out1,
        m => o1_m,
        n => out2
    );
    0.5

end struct;

```

```

# Name: Duerner
# Vorname: Daniel

# frequency: 100 MHz
force clk 0 0ns, 1 5ns -repeat 10ns      0.5
force rst_n 0 0 ns, 1 50 ns                0.5
3

force rst_int 0 0 ns, 1 700ns, 0, 710ns   0.5

force threshold 16#0F 0 ns
force limit 16#F00 0 ns
force tx 16#00 0 ns, 16#0E 25ns, 16#10 35ns, 16#0A 45ns, 16#0F 55ns, 16#FF 65ns
run 800 ns

```

```

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;          0.5

entity integrate is
port(
    clk      : in  std_ulogic;
    rst_n   : in  std_ulogic;
    rst_int  : in  std_ulogic;
    tx       : in  std_ulogic_vector(7 downto 0);
    threshold : in  std_ulogic_vector(7 downto 0);
    limit    : in  std_ulogic_vector(11 downto 0);
    int_value : out std_ulogic_vector(15 downto 0);
    int_error : out std_ulogic
);
end entity integrate;

architecture rtl of integrate is
begin
    SIGNAL integrate_value : unsigned(15 downto 0);      0.5

begin
    p_integrate : process (rst_n,clk)
    begin
        if rst_n = '0' then
            integrate_value <= (Others => '0');           0.5
        elsif rising_edge(clk) then
            if rst_int = '1' then
                integrate_value <= (Others => '0'); }      0.5
            elsif tx>threshold then
                integrate_value <= integrate_value + resize(unsigned(tx),16); 0.5
            end if;
        end if;
    end process p_integrate;

    int_value <= std_ulogic_vector(integrate_value);      0.5
    int_error <= '1' when integrate_value>unsigned(limit) else '0'; 0.5
end architecture rtl;

```

integrate.vhd

4.5/5

```

-- Name: Duerner
-- Vorname: Daniel

LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;

entity integrate is
generic(
    g_threshold : std_ulogic_vector(7 downto 0) := X"0F";
    g_limit : std_ulogic_vector(11 downto 0) := X"F00"
);
port(
    clk      : in  std_ulogic;
    rst_n   : in  std_ulogic;
    rst_int  : in  std_ulogic;
    tx       : in  std_ulogic_vector(7 downto 0);
    --threshold : in  std_ulogic_vector(7 downto 0);
    --limit    : in  std_ulogic_vector(11 downto 0);
    int_value : out std_ulogic_vector(15 downto 0);
    int_error : out std_ulogic
);
end entity integrate;

```

integrate-generics.vhd

1.5/2

```

p_count : process (clk, rst) is
begin
    if rst = '1' then
        cnt <= (OTHERS => '0');           0.5
    elsif rising_edge(clk) then
        cnt <= ((cnt + 1) mod 6); -- cnt: +3 Speicherelemente
    end if;
end process p_count;

```

signal-gen.vhd

2.5/3

1 (anmer idle)