

Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

Aufgabe 1 Verständnisfragen

(10 Punkte)(5 × 2)

Beantworten Sie die folgenden Fragen mit *wenigen kurzen Sätzen*.

- a) Wie unterscheiden sich *Pseudo-nMOS* von CMOS-Schaltungen? Nennen Sie einen Vor- und einen Nachteil von Pseudo-nMOS Schaltungen.

Bei Pseudo-nMOS Schaltungen wird das komplementäre pMOS-Transistornetzwerk durch einen einzelnen, immer eingeschalteten schwachen pMOS-Transistor ersetzt. Das spart Fläche, sorgt aber für eine statische Leistungsaufnahme.

1 Punkt für Konzept von Pseudo-nMOS, 0.5P je Vor-/Nachteil [V3F43]

- b) Warum lässt sich jede boole'sche Funktion ausschließlich mit *NAND-Gattern* realisieren?

Jede boole'sche Funktion lässt sich zweistufig (bspw. als DNF) mit NOT, AND und OR Gattern realisieren. Wegen $\text{NOT}(a) = \text{NAND}(a, a)$, $\text{AND}(a, b) = \text{NAND}(\text{NAND}(a, b), \text{NAND}(a, b))$ und $\text{OR}(a, b) = \text{NAND}(\text{NAND}(a, a), \text{NAND}(b, b))$ genügen dafür NAND-Gatter.

1 Punkt für zweistufige Realisierung, 1 Punkt für Abbildung auf NOT/AND/OR, [V15F43]

- c) Wie unterscheiden sich *kombinatorische und sequentielle* Schaltungen?

Die Ausgänge kombinatorischer Schaltungen hängen ausschließlich von den aktuellen Eingängen ab. Sequentielle Schaltungen haben hingegen einen internen Zustand, welcher deren Ausgabe beeinflusst.

1 Punkt je Charakterisierung der Schaltungstypen [V7F15]

- d) Welche *Arten der Parallelität* können zur Anwendungsbeschleunigung eingesetzt werden? Wodurch unterscheiden sich diese?

Bei der räumlichen Parallelität wird die Hardware zur Bearbeitung einer Aufgabe vervielfältigt, um mehrere Aufgaben gleichzeitig behandeln zu können. Bei der zeitlichen Parallelität wird eine Aufgabe in (voneinander abhängige) Teilprobleme (Stufen) zerlegt. Die Stufen werden parallel ausgeführt, so dass mehrere Probleme gleichzeitig (überlappend) bearbeitet werden können, ohne die Hardware zu vervielfältigen.

0.5 Punkte je Typ, 0.5 Punkte je Charakterisierung [V9F28]

- e) Nennen Sie eine *Alternative zur Ripple-Carry* Implementierungen der binären Addition. Wie versucht diese, den kritischen Pfad zu verkürzen?

Der Conditional-Sum Adder berechnet die obere Worthälfte für beide möglichen Carry-Ausgänge der unteren Worthälfte, und wählt am Ende nur noch das richtige Teilergebnis aus. Der Carry-Lookahead Adder berechnet Propagate- und Generate-Flags für ganze Eingabeblocke, so dass die Übertragskette nur noch zwischen den Blöcken verläuft.

1 Punkt für den Addierer-Typ, 1 Punkt für die Implementierungsidee [V13F13, V13F18]

Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____

Matrikelnummer: _____

Aufgabe 2 Zahlendarstellungen und binäre Subtraktion

(10 Punkte)(5 + 5)

- a) Vervollständigen Sie die folgende Tabelle vorzeichenloser Zahlendarstellungen. Alle Einträge einer Zeile sollen dabei den gleichen numerischen Wert repräsentieren. Verwenden Sie möglichst kurze Ziffernfolgen (ohne führende Nullen).

Dezimal	Binär	Hexadezimal
202_{10}	$1100\ 1010_2$	CA_{16}
45_{10}	$10\ 1101_2$	$2D_{16}$
2018_{10}	$111\ 1110\ 0010_2$	$7E2_{16}$

Ein Punkt je Dezimal- und Binäreintrag. 0.5 Punkte je Hexadezimaleintrag. [V2F20, Ü2.3.1]

- b) Wandeln Sie -40_{10} und 77_{10} in 1 Byte breite Zweierkomplement-Zahlen um. Subtrahieren Sie die Binärdarstellungen voneinander. Wandeln Sie das Ergebnis ins Dezimalformat und ins 12 bit Hexadezimalformat um. Der Lösungsweg wird bewertet.

- Umrechnung ins Zweierkomplement:
 $-40_{10} = \overline{0010\ 1000}_2 + 1 = 1101\ 1000_2$
 $+77_{10} = 0100\ 1101_2$

- Subtrahend negieren:
 $-77_{10} = \overline{0100\ 1101}_2 + 1 = 1011\ 0011_2$

- Addition:
$$\begin{array}{r} 1101\ 1000_2 \\ + 1011\ 0011_2 \\ \hline = 1000\ 1011_2 \end{array}$$

- Umrechnen ins Dezimalformat:
 $1000\ 1011_2 = -128_{10} + 8_{10} + 2_{10} + 1_{10} = -117_{10}$ ✓

- Umrechnen ins Hexadezimalformat nach vorzeichenbehafteter Bitbreitenerweiterung:
 $1000\ 1011_2 = 1111\ 1000\ 1011_2 = F8B_{16}$

Je ein Punkt pro Teilschritt. [V2F31-35, Ü2.5]

Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

Aufgabe 3 Realisierung von logischen Funktionen

(10 Punkte) (2 + 2 + 6)

Die folgenden Teilaufgaben hängen *nicht* voneinander ab.

- a) Geben Sie die in folgender Wahrheitstabelle spezifizierte boole'sche Funktion in *konjunktiver Normalform* an.

A	B	C	Y
0	0	0	1
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	0

$$Y = (A+B+\bar{C})$$

$$\cdot (A+\bar{B}+\bar{C})$$

$$\cdot (\bar{A}+B+C)$$

$$\cdot (\bar{A}+\bar{B}+\bar{C})$$

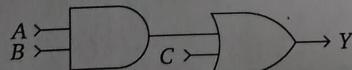
0.5 Punkte je Maxterm. [V4F23, Ü4.3]

- b) Formen Sie $Y = \overline{(A+B)(\bar{A}+C)}$ in eine *disjunktive Darstellung* (Summe von Implikanten) um. Geben Sie für jeden Umformungsschritt die angewandte Regel der boole'schen Algebra an.

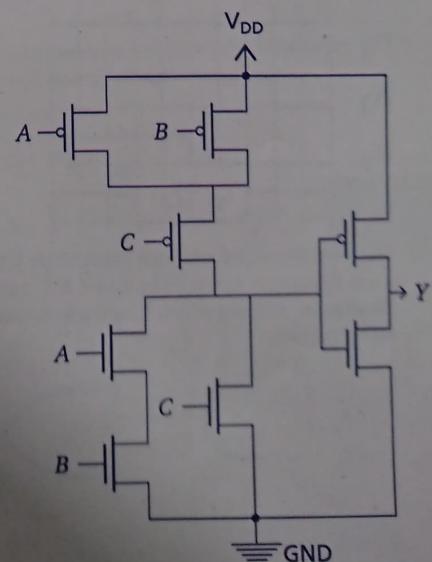
$$\begin{aligned}
 Y &= \overline{(A+B)(\bar{A}+C)} && \text{De Morgan} \\
 &= \overline{A+B} + \overline{\bar{A}+C} && \text{De Morgan} \\
 &= \overline{A}\overline{B} + \overline{\bar{A}}\overline{C} && \text{Involution} \\
 &= \overline{A}\overline{B} + A\overline{C}
 \end{aligned}$$

0.5 Punkte je Regelanwendung, 0.5 Punkte für korrekte Disjunktion. [V4F19+40, Ü4.4, Ü4.5]

- c) Realisieren Sie die folgende Gatter-Schaltung in CMOS. Verwenden Sie ausschließlich *positive Eingangsliterale*.



0.5 Punkte je Transistor,
2 Punkte für korrekte Verbindungen.
[V3F41, V5F10, Ü3.5.2, Ü4.2]



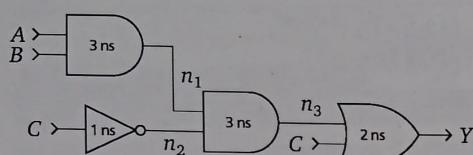
Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

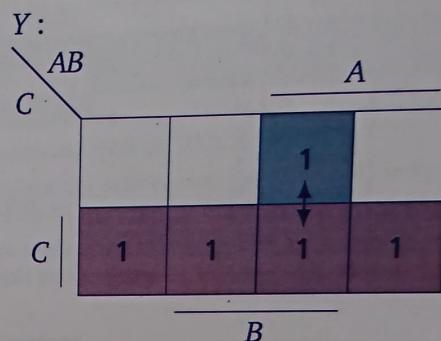
Aufgabe 4 Störimpulse (11 Punkte)(4 + 5 + 2)

Im Folgenden sei $t_{cd,AND} = t_{pd,AND} = 3 \text{ ns}$, $t_{cd,OR} = t_{pd,OR} = 2 \text{ ns}$ und $t_{cd,NOT} = t_{pd,NOT} = 1 \text{ ns}$.

- a) Identifizieren Sie die *kritischen Eingangstransitionen* (mit nur einer geänderten Eingangsvariable) der folgenden Schaltung, bei denen *Störimpulse* auftreten können. Verwenden Sie dazu ein *Karnaugh-Diagramm*.

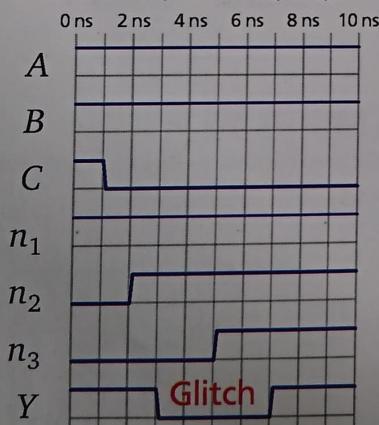


- 1 Punkt für Darstellung des Karnaugh-Diagramms,
1 Punkt für Implikanten im Karnaugh-Diagramm,
1 Punkt für Markierung oder Angabe des kritischen Übergangs
[V6F42, Ü7.2.1]



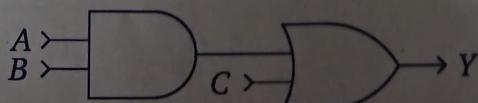
kritische Transition: $(1, 1, 0) \leftrightarrow (1, 1, 1)$

- b) Überprüfen Sie, ob ein Störimpuls bei der *fallenden Flanke* der identifizierten Eingangstransition auftritt. Füllen Sie dazu das folgende *Timing-Diagramm* aus:



- 0.5 Punkte insgesamt für A , B und n_1 ,
1 Punkt jeweils für C , n_2 , n_3 und Y ,
0.5 Punkte für die Kennzeichnung des Störimpulses
[V7F9, Ü7.2.1]

- c) Zeichnen Sie eine *funktional äquivalente* Gatterschaltung ohne Störimpuls. Begründen Sie kurz Ihr Vorgehen.
Nach Expansion des Implikanten $AB\bar{C}$ zum Primimplikanten AB überlappen sich alle Implikanten im Karnaugh-Diagramm. Alle einzelnen Eingangstransitionen innerhalb des On-Sets verlaufen daher auch innerhalb eines Primimplikanten.



- 1 Punkt für die Schaltung,
1 Punkt für die Begründung
[V6F43, Ü7.2.1]

Klausur zur Vorlesung Digitaltechnik

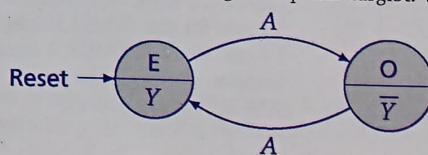
Name, Vorname: _____ Matrikelnummer: _____

Aufgabe 5 Endliche Automaten

(16 Punkte)(3 + 5 + 4 + 4)

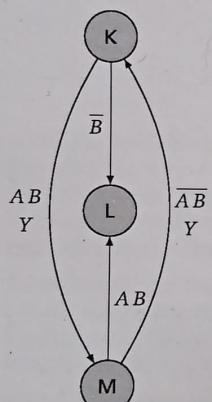
Die folgenden Teilaufgaben hängen *nicht* voneinander ab.

- a) Zeichnen Sie das Diagramm eines *Moore-Automaten* mit einem Eingang A und einem Ausgang Y , welcher die *ungerade Parität* der Eingabesequenz ausgibt. Verwenden Sie *möglichst wenige Zustände*.



0.5 Punkte je Zustand inkl. Ausgabe (Namen nicht relevant),
0.5 Punkte je Zustandsübergang,
1 Punkt für Korrektheit (inkl. Wahl des Startzustandes)
[V3F11+15, V8P8, Ü8.5]

- b) Geben Sie die *Zustandsübergangs-* und die *Ausgabetafel* für folgenden Automaten mit zwei Eingängen A, B und einem Ausgang Y an. Die Zustände müssen dabei *nicht binär codiert* werden. Verwenden Sie *Don't-Cares* für eine kompakte Darstellung. Um welchen *Automatentypen* handelt es sich?



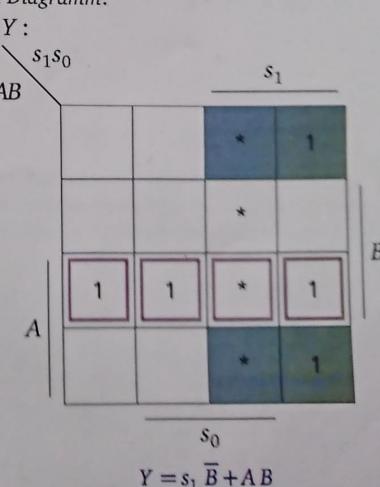
S	A	B	S'	Y
K	*	0	L	0
K	0	1	K	0
K	1	1	M	1
L	*	*	L	0
M	0	*	K	1
M	1	0	K	1
M	1	1	L	0

0.5 Punkte je Tabellen-Header ($S, A, B \leftrightarrow S'$ und $S, A, B \leftrightarrow Y$),
0.5 Punkte je Zeile (in vorletzter Zeile auch M, *, 0, K, 1 ok),
0.5 Punkte für Erkennen des Mealy-Automaten
[V8F14+32, Ü8.5]

- c) Geben Sie einen *minimierten boole'schen Ausdruck* für die *Ausgabefunktion* des durch die folgenden Tabellen spezifizierten Automaten an. Verwenden Sie dafür das vorgegebene *Karnaugh-Diagramm*.

Zustandsübergänge			Ausgabe			Zustandskodierung			
S	A	B	S'	S	A	B	Y	s_1	s_0
P	0	0	P	P	0	0	1		
P	1	0	Q	P	0	1	0		
P	*	1	R	P	1	*	1		
Q	0	0	P	Q	0	*	0		
Q	1	0	R	Q	1	0	0		
Q	0	1	R	Q	1	1	1		
Q	1	1	Q	R	0	*	0		
R	0	*	Q	R	1	0	0		
R	1	*	P	R	1	1	1		

S	s_1	s_0
P	1	0
Q	0	1
R	0	0



Die Zustandsübergangstabelle wird nicht benötigt.

1 Punkt für Einsen im Karnaugh-Diagramm,

1 Punkt für Don't Cares im Karnaugh-Diagramm,

(ungenutzte Zustandskodierung $s_1 = s_0 = 1$)

1 Punkt je Primimplikant

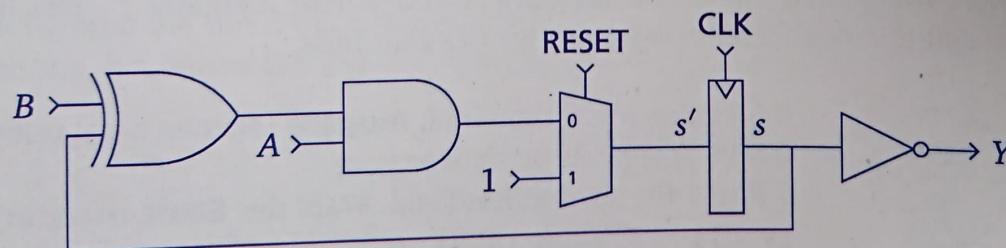
[V5F32+36, Ü5.4, Ü8.2]

Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____

Matrikelnummer: _____

- d) Zeichnen Sie das *Schaltwerk* eines endlichen Automaten mit zwei Eingängen A, B , zwei Zuständen und einem Ausgang Y . Realisieren Sie dabei die *Zustandsübergangsfunktion* $s' = A(s \oplus B)$ sowie die *Ausgabefunktion* $Y = \bar{s}$. Der Automat soll über einen *synchronen Reset* in den Startzustand $s = 1$ versetzt werden.



1 Punkt für Register mit CLK
(aber ohne asynchrones RESET),
1 Punkt für drei richtige Logikgatter,
1 Punkt für RESET-MUX, 1 Punkt für
korrekte Verbindungen
[V8F31, V11F32, Ü8.2b]

Klausur zur Vorlesung Digitaltechnik

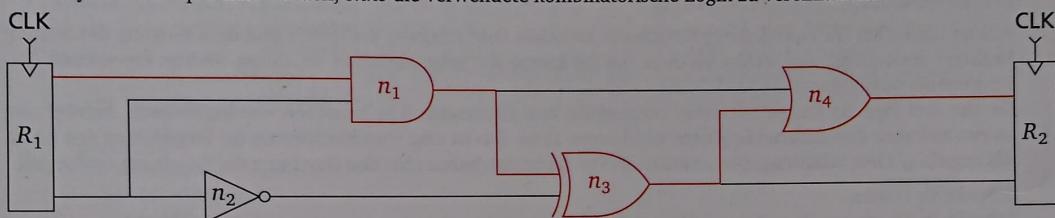
Name, Vorname: _____ Matrikelnummer: _____

Aufgabe 6 Synchrone Sequential Schaltungen (10 Punkte)(2 + 2 + 4 + 2)

Für diese Aufgabe werden ausschließlich die wie folgt spezifizierten Logikgatter verwendet:

	XOR	AND	OR	NOT
t_{cd}	3 ns	2 ns	1 ns	1 ns
t_{pd}	4 ns	3 ns	2 ns	1,5 ns

Darüber hinaus sei $t_{ccq} = 100 \text{ ps}$, $t_{pcq} = 300 \text{ ps}$, $t_{\text{setup}} = 700 \text{ ps}$ und $t_{\text{hold}} = 1 \text{ ns}$ für alle Register. Folgendes Schaltwerk soll analysiert und optimiert werden, ohne die verwendete kombinatorische Logik zu vereinfachen:



- a) Geben Sie den *kritischen Pfad* der Schaltung an. Mit welcher Frequenz kann das Schaltwerk *maximal getaktet* werden, ohne die *Setup-Bedingung* von R_2 zu verletzen? Begründen Sie Ihre Antwort.

Der kritische Pfad von R_1 nach R_2 verläuft durch das n_1 , n_3 und n_4 . Die maximale Gesamtverzögerung der kombinatorischen Schaltung beträgt $t_{pd} = 9 \text{ ns}$. Zusammen mit der Ausgabeverzögerung t_{pcq} von R_1 und t_{setup} von R_2 ergibt dies einen kritischen Pfad der Länge 10 ns. Dies begrenzt die Taktrate auf maximal 100 MHz.

0.5 Punkte für die Angabe oder Markierung des kritischen Pfades (welche Gatter),

1 Punkt für die Berechnung der minimalen Taktperiode,

0.5 Punkte für die Umrechnung in die maximale Taktrate,

[V9F21+32, Ü9.4, Ü9.5]

- b) Wird die *Hold-Bedingung* von R_2 eingehalten? Begründen Sie Ihre Antwort.

Der kürzeste Pfad von R_1 nach R_2 verläuft durch das AND und OR Gatter. Die minimale Gesamtverzögerung der kombinatorischen Schaltung beträgt $t_{cd} = 3 \text{ ns}$. Zusammen mit der minimalen Ausgabeverzögerung t_{ccq} von R_1 erreicht eine Signaländerung R_2 frühestens 3,1 ns nach der steigenden Taktflanke. Da R_2 eine geringere t_{hold} hat, ist die Hold-Bedingung erfüllt.

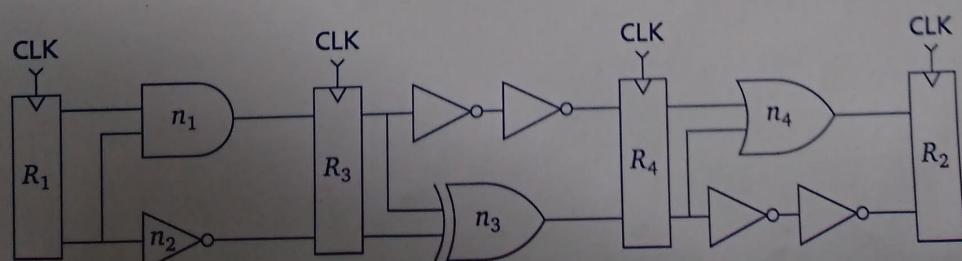
0.5 Punkte für die Angabe des kürzesten Pfades (welche Gatter),

1 Punkt für die Berechnung der frühestmöglichen Änderungszeit bei R_2 ,

0.5 Punkte für den Vergleich mit t_{hold} ,

[V9F21f, Ü9.4]

- c) Modifizieren Sie das Schaltwerk so, dass dieses mit 200 MHz getaktet werden kann, und dabei weiterhin die *Setup- und Hold-Bedingungen* aller Register erfüllt werden. Entfernen Sie dabei keines der bestehenden Logikgatter. Sie können die notwendigen Komponenten direkt in der gegebenen Grafik ergänzen. Begründen Sie kurz Ihr Vorgehen.



Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

Der kritische Pfad verläuft nun von R_3 nach R_4 durch n_3 . Die maximale Gesamtverzögerung der kombinatorischen Schaltung beträgt $t_{pd,XOR} = 4$ ns. Zusammen mit der Ausgabeverzögerung t_{peq} von R_3 und t_{setup} von R_4 ergibt dies einen kritischen Pfad der Länge 5 ns. Dies begrenzt die Taktrate auf maximal 200 MHz.

Die Hold-Bedingung von R_3 ist durch das bestehende NOT Gatter (n_2) gerade erfüllt. Die direkten Verbindungen zwischen R_3 und R_4 bzw. R_4 und R_2 müssen aber durch doppelte Inverter ersetzt werden, um die Hold-Bedingungen von R_4 und R_2 zu erfüllen.

- 1 Punkt je eingefügte Registerstufe,
- 1 Punkt für Analyse des neuen kritischen Pfades,
- 1 Punkt für Einfügen von extra Gattern für Hold-Bedingung,
[V9F23+33f, Ü9.5]

- d) Welche *Latenz* hat Ihr modifiziertes Schaltwerk zwischen dem Ausgang des ersten und dem Eingang des letzten Registers mindestens? Vergleichen Sie diese mit der Latenz der ursprünglichen Schaltung. Welche Eigenschaft des Schaltwerks wurde durch die Modifikation verbessert?

Da alle drei Pipeline-Stufen mit einer Taktperiode von mindestens 5 ns betrieben werden müssen, beträgt die Latenz zwischen den äußeren Registern mindestens 15 ns. Das ist eine Verschlechterung im Vergleich zu den 10 ns der ursprünglichen Schaltung. Die zusätzlichen Pipelinestufen haben aber den Durchsatz der Schaltung verdoppelt.

- 1 Punkt für Latenz,
- 1 Punkt für Durchsatz als verbesserte Eigenschaft,
[V9F29, Ü9.5]

Klausur zur Vorlesung
Name, Vorname: _____

Aufgabe 7 Systemverarbeitung
In dieser Aufgabe werden
Operanden als
a) Erweiterung

Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

Aufgabe 7 SystemVerilog extrem

(13 Punkte)(6 + 7)

In dieser Aufgabe soll eine *kombinatorische Schaltung* zur Berechnung des *Maximums oder Minimums* einer Liste von Operanden als *parametrisiertes SystemVerilog-Modul* implementiert und verifiziert werden.

- a) Ergänzen Sie die folgende Modulschnittstelle um die *rekursive (nicht iterative)* Implementierung des **extremum** Moduls. Für MAX==0 soll das Modul das Minimum aller im Eingabevektor A per *flattening* übergebenen Werte ermitteln. Für alle anderen Werte von MAX soll das Maximum ermittelt werden. Achten Sie darauf, dass auch eine ungerade Anzahl von Eingabewerten korrekt unterstützt wird. Organisieren Sie die Rekursion so, dass der resultierende *kritische Pfad* minimiert wird. Kommentieren Sie Ihre Lösung.

```
module extremum #(parameter WIDTH = 8,           // Bitbreite der Eingabewerte
                parameter DEPTH = 4,          // Anzahl der Eingabewerte
                parameter MAX    = 1)         // Wahl des Extremums
  (input logic [DEPTH*WIDTH-1:0] A,   // Eingabewerte (flattening)
   output logic      [WIDTH-1:0] Y); // ermitteltes Extremum

  generate
    // Rekursionsabbruch
    if (DEPTH == 1)
      assign Y = A;

    // Rekursion
    else begin

      // Divide ...
      localparam D1 = DEPTH/2; // floor(DEPTH/2.0)
      localparam D2 = DEPTH-D1; // ceil (DEPTH/2.0)
      logic [WIDTH-1:0] e1, e2;
      extremum #(WIDTH, D1, MAX) i1 (A[0           +: D1*WIDTH], e1); // untere Hälfte
      extremum #(WIDTH, D2, MAX) i2 (A[D1*WIDTH +: D2*WIDTH], e2); // obere Hälfte

      // ... and conquer
      assign Y = (MAX && (e1 > e2) || !MAX && (e1 < e2)) ? e1 : e2;
    end
  endgenerate
endmodule
```

1 Punkt für den Rekursionsabbruch,
1 Punkt für die Breite der oberen/unteren Hälfte (richtig Runden),
1 Punkt je Submodul-Instanziierung,
1 Punkt für Auswahl und Berechnung des Extremums,
1 Punkt für (sinnvolle) Kommentare
[V11F41, V12F8, V13F6, Ü12.3.1]

Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

- b) Ergänzen Sie die folgende Testbench so, dass die Berechnung des Minimums und Maximums für eine konkrete (durch die Parameter W und D spezifizierte) Konfiguration des Eingabevektors erschöpfend verifiziert wird. Es soll ein Eingabevektor pro Nanosekunde getestet werden. Zur besseren Visualisierung soll der Eingabevektor auch in ein Array der einzelnen Eingabewerte (list) konvertiert werden. Kommentieren Sie Ihre Lösung.

```
`timescale 1 ns / 10 ps
module extremum_tb;

localparam W=2;           // Bitbreite der Eingabewerte
localparam D=5;           // Anzahl der Eingabewerte

logic [D*W-1:0] a;        // Eingabevektor (nach flattening)
logic [W-1:0] max,         // Ausgabe von extremum mit MAX=1
            min,          // Ausgabe von extremum mit MAX=0
            list [0 : D-1], // Eingabearray (ohne flattening)
            emax,          // max(list)
            emin;          // min(list)

extremum #(W, D, 1) imax (a, max);
extremum #(W, D, 0) imin (a, min);

always_comb begin
    emax = {D{1'b0}};           // auf Minimalwert initialisieren
    emin = {D{1'b1}};           // auf Maximalwert initialisieren
    for (int k=0; k<D; k++) begin
        list[k] = a[k*W +: W];   // deflattening
        if (list[k] > emax) emax = list[k]; // erwartete werte anpassen
        if (list[k] < emin) emin = list[k];
    end
end

initial begin
    $dumpfile("extremum_tb.vcd");
    $dumpvars;

    // erschöpfender Test: alle Eingabekombinationen prüfen
    for (int i=0; i<(1<<(D*W)); i++) begin
        a = i;
        if (max != emax) $display("@%0t: unexpected max %0d != %0d", $time, max, emax);
        if (min != emin) $display("@%0t: unexpected min %0d != %0d", $time, min, emin);
        #1;
    end

    $display("FINISHED extremum_tb for W=%0d, D=%0d", W, D);
    $finish;
end
endmodule
```

- 0.5 Punkte je Submodul-Instanziierung,
1 Punkt für deflattening,
1 Punkt je Berechnung des erwarteten Wertes,
1 Punkt für erschöpfende Testschleife,
0.5 Punkte für mit Verzögerung in Testschleife,
0.5 Punkte je Test inkl. display,
0.5 Punkte für (sinnvolle) Kommentare
[V12F12, Ü12.1.2]

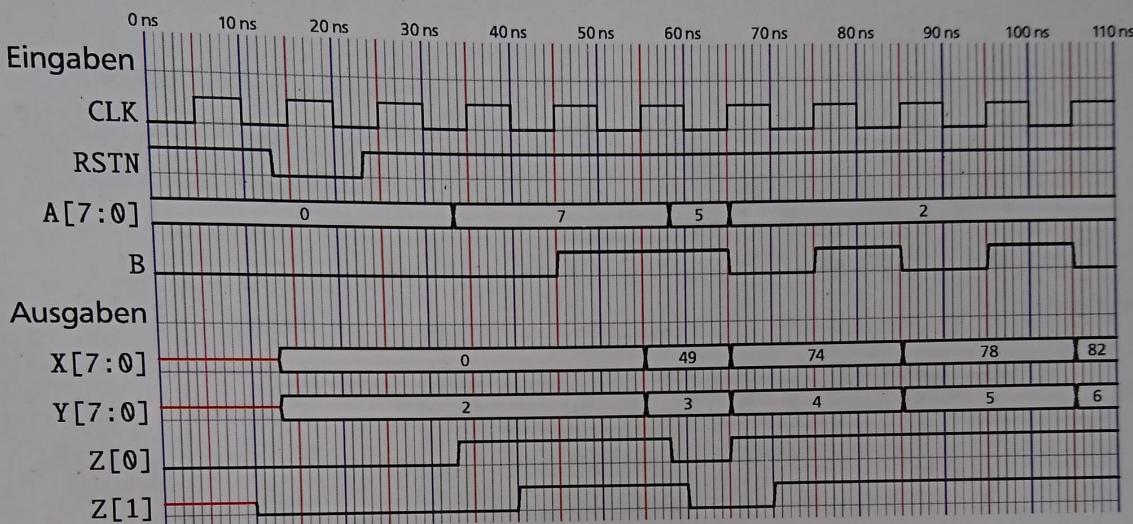
Klausur zur Vorlesung Digitaltechnik

Name, Vorname: _____ Matrikelnummer: _____

Aufgabe 8 Reverse-Engineering

(10 Punkte)(10)

Implementieren Sie ein SystemVerilog-Modul mit folgendem Ein-/Ausgabeverhalten. Kommentieren Sie Ihre Lösung.



```
module rev_eng(input logic CLK, RSTN,
                input logic [7:0] A,      input logic B,
                output logic [7:0] X, Y, output logic [1:0] Z);

    always_ff @(posedge CLK,           //posedge register (@55ns)
               negedge RSTN) begin   //async reset (@13ns)
        if (RSTN == 0) begin       //reset low active (13ns-23ns)
            X <= 8'd0;           //reset {X,Y} to {0,2} (@13ns)
            Y <= 8'd2;
        end else if (B) begin    //B is clock enable
            X <= X + A*A;        //accumulate square
            Y <= Y + 1;           //increment counter
        end
    end

    always_comb Z[0] <= ^A;          //combinatorial input parity
    always_ff @(negedge CLK) Z[1] <= Z[0]; //delayed parity on negedge (@40ns)

endmodule
```

funktional äquivalente Lösungen werden akzeptiert,
9 Punkte für die im Lösungsvorschlag kommentierten Stellen,
1 Punkt für (sinnvolle) Kommentare,
[Transferaufgabe, keine äquivalente Übung oder Vorlesungsunterlagen]