

Vorname: _____

Name: _____

Matr.-Nr.: _____

Note: _____

27.02.2006 - 10⁰⁰ Uhr bis 12⁰⁰ Uhr

UNIVERSITÄT KARLSRUHE
Institut für Industrielle Informationstechnik
- Prof. Dr.-Ing. habil. K. Dostert -

Vordiplomprüfung im Fach

Mikrorechnertechnik

Die Prüfung umfasst **10 Aufgaben auf 19 Seiten**.

Bitte schreiben Sie auf **alle** Lösungsblätter Ihren Namen und Ihre Matrikelnummer.

Geben Sie bei allen Aufgaben den kompletten Lösungsweg an! Die alleinige Nennung des Endresultats ist zur Erlangung der vollen Punktzahl einer Aufgabe nicht ausreichend.

Die Verwendung eigenen Papiers ist nicht erlaubt.

Bei Bedarf kann zusätzliches Schreibpapier bei der Aufsicht angefordert werden.

Als Hilfsmittel sind Schreib- und Zeichenzeug sowie Taschenrechner mit zu Beginn der Klausur gelöschtem Speicher zugelassen.

Aufgabe:	1	2	3	4	5	6	7	8	9	10	gesamt
Punkte:											
Erreichbare Punktzahl:	9	10	9	10	10	10	11	10	10	11	100

Aufgabe 1: Zahlendarstellung in Mikrorechnerprogrammen

9 Punkte

Zur Zahlendarstellung in Mikrorechnerprogrammen finden verschiedene Zahlenformate Verwendung. Zunächst wird der folgende Bitstring betrachtet, wobei die Abstände zwischen den Vierergruppen lediglich der besseren Lesbarkeit dienen:

$$b = 1001\ 0110\ 0110\ 1001\ 0001\ 0101\ 0010\ 1000$$

Geben Sie den jeweiligen Dezimalwert dieses Bitstrings an, wenn man ihn wie folgt deutet:

- a) Zahl zur Basis 2 im 2er-Komplement!
- b) BCD-Zahl!
- c) Fraktalzahl!
- d) Gleitkommazahl gemäß IEEE 754! Geben Sie die Gleitkommazahl als gekürzten Bruch und nicht als gerundete Kommazahl an!

Stellen Sie die Zahl $-\left(12 + \frac{3}{64}\right)$ wie folgt dar:

- e) Als Gleitkommazahl gemäß dem IEEE-P754 Single-Precision-Standard!
- f) Als **2er-Komplement-Festkommazahl** mit 16 Vorkomma- und 16 Nachkommabits!

Aufgabe 2: FSM mit Beschreibung durch Zustandsgraph/Flow-Table

10 Punkte

Ein endlicher Automat („Finite State Machine“, FSM) kann durch einen Zustandsgraphen oder eine Ablauf-tabelle („Flow-Table“) beschrieben werden. Im Folgenden ist das Verhalten einer einfachen Garagentorsteuerung mit Hilfe dieser beiden Methoden zu beschreiben.

Per Fernbedienung können die drei Befehle „Auf“, „Zu“ und „Stopp“ erteilt werden, die mit a , z und s abgekürzt werden. Die Endstellungen des Tors werden durch die Signale ea (Tor auf) und ez (Tor zu) signalisiert.

Der Zustandsgraph für die Modellierung enthält drei Zustände ($s1$, $s2$, $s3$). Im Zustand „Auf“ wird das Tor durch einen laufenden Motor geöffnet, im Zustand „Zu“ geschlossen. Im Zustand „Stopp“ ist der Motor ausgeschaltet. In Abbildung 2.1 ist der unvollständige Zustandsgraph für die beschriebene Garagentorsteuerung abgebildet.

In Abbildung 2.2 ist ein Ausschnitt aus der Ablauf-tabelle der Steuerung angegeben. Dabei sind die beschriebenen Zustände mit $s1$ bis $s3$, die Folgezustände mit $f1$ bis $f3$ bezeichnet. Der Eingangsvektor $x = (a, z, s, ea, ez)$ bestimmt, ausgehend vom aktuellen Zustand $s1$ bis $s3$, jeweils den Folgezustand $f1$ bis $f3$. Falls eine Variable den Wert „1“ annimmt, ist die zugehörige Bedingung erfüllt, im Fall „0“ ist sie nicht erfüllt. Irrelevante Eingangsvariablen sind durch „X“ gekennzeichnet.

- Tragen Sie die Bedingungen für die Übergänge aus den Zuständen „Auf“ und „Zu“ in den Zustand „Stopp“ in den Zustandsgraphen in Abbildung 2.1 ein und verwenden Sie dazu die Informationen aus der Ablauf-tabelle in Abbildung 2.2!
- Tragen Sie die Bedingungen für das Verbleiben in den Zuständen „Auf“ und „Zu“ in die Ablauf-tabelle in Abbildung 2.2 ein und verwenden Sie dazu die Informationen aus Abbildung 2.1! Kennzeichnen Sie irrelevante Bedingungen durch „X“!

Ein Übergang aus den Zuständen „Stopp“ und „Zu“ in den Zustand „Auf“ soll erfolgen, wenn der Befehl „Auf“ gesendet wird. Gleichzeitig darf keiner der beiden anderen Befehle gesendet werden und der Endschalter ea darf nicht geschlossen sein. Entsprechend soll ein Übergang aus den Zuständen „Stopp“ und „Auf“ in den Zustand „Zu“ erfolgen, wenn nur der Befehl „Zu“ gesendet und der Endschalter ez nicht geschlossen ist.

- Beschriften Sie die zu diesen beiden Übergängen gehörenden Pfeile in Abbildung 2.1 mit den entsprechenden booleschen Gleichungen für die Eingangsvektoren und tragen Sie diese Übergänge in die Ablauf-tabelle in Abbildung 2.2 ein! Kennzeichnen Sie irrelevante Bedingungen durch „X“!

Fortsetzung der 2. Aufgabe:

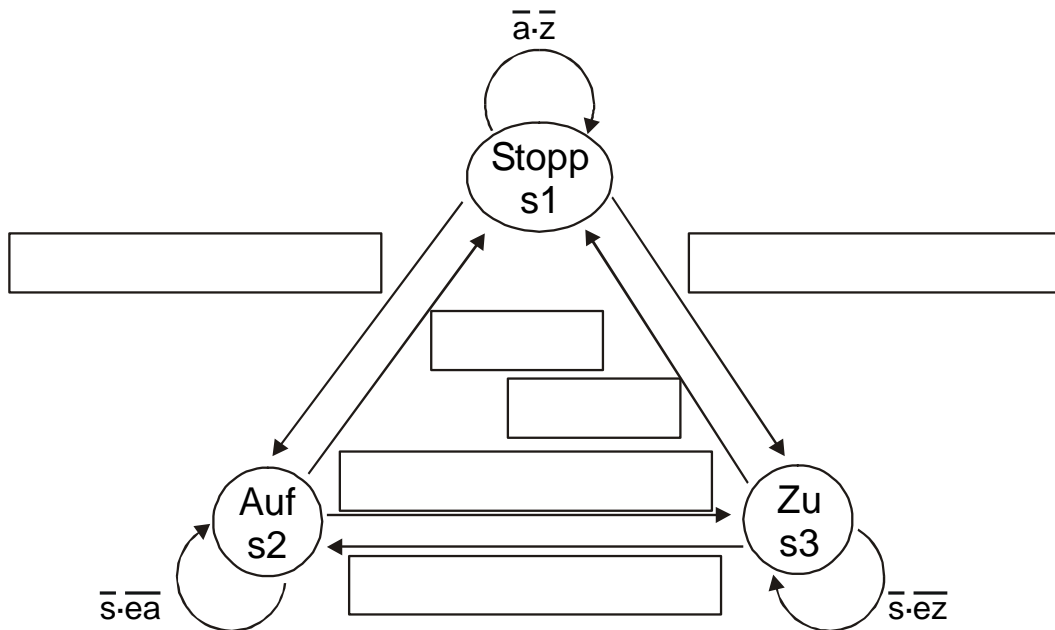


Abbildung 2.1: Zustandsgraph für Garagentorsteuerung

relevant =	a,	z,	s,	ea,	ez;	
s1,	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	f2;
s1,	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	f3;
s2,	X	X	1	X	X,	f1;
s2,	X	X	X	1	X,	f1;
s2,	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	f2;
s2,	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	f3;
s3,	X	X	1	X	X,	f1;
s3,	X	X	X	X	1,	f1;
s3,	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	f2;
s3,	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	f3;

Abbildung 2.2: Ausschnitt aus Ablauf- und Ablaftabelle für Garagentorsteuerung

Aufgabe 3: MOS-Technologie

9 Punkte

- a) Gegeben sei die folgende Schaltung aus Abbildung 3.1. Beschriften Sie *Gate*, *Drain* und *Source* der Transistoren, sowie die Betriebsspannung U_B , die Eingangsspannung U_E und die Ausgangsspannung U_A an den entsprechenden Leitungen! Kennzeichnen Sie eindeutig, bei welchem der Transistoren es sich um einen n-Kanal- bzw. einen p-Kanal-Typen handelt!

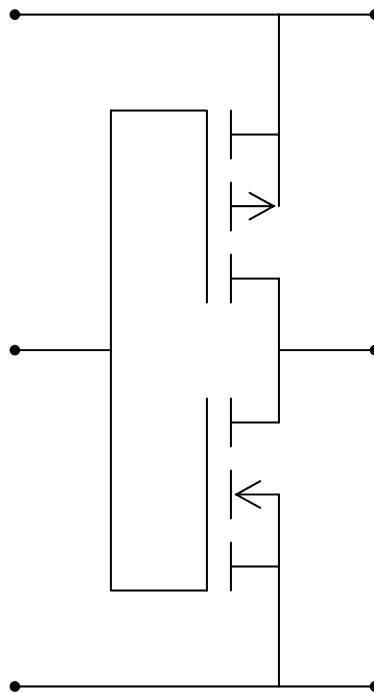


Abbildung 3.1: Schaltung mit MOS-Transistoren

- b) Welche logische Funktion führt die Schaltung in Abbildung 3.1 aus?
- c) Erläutern Sie die Funktionsweise der Schaltung aus Abbildung 3.1 in Stichworten! Erläutern Sie anhand des oberen Transistors, wann dieser Transistor in Abhängigkeit von U_E sperrt bzw. leitet!

Aufgabe 4: CMOS-Transferrates

10 Punkte

Die CMOS-Technologie ermöglicht den Aufbau besonderer Schaltungsstrukturen mit Hilfe von Transferrates. Im Folgenden soll ein CMOS XNOR-Gatter unter Verwendung von Transferrates entworfen werden.

- a) Vervollständigen Sie die Wahrheitstabelle 4.1, so dass die Verknüpfung $c = a \text{ XNOR } b$ realisiert wird!

a	b	c

Tabelle 4.1: Zustandsbelegung eines XNOR-Gatters

- b) Ergänzen Sie alle Verbindungen in der nachfolgenden Abbildung 4.1, so dass das Schaltbild eines CMOS XNOR-Gatters realisiert wird!
(**Hinweis:** Verwenden Sie hierzu ausschließlich die beiden bereits vorhandenen Inverter und Transferrates! Verwenden Sie **keine** zusätzlichen Transferrates!)

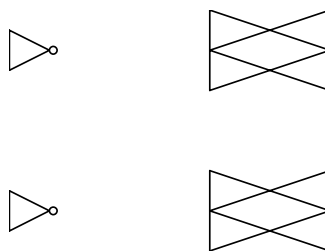


Abbildung 4.1: CMOS XNOR-Gatter

Im Folgenden werde die Schaltung aus Abbildung 4.2 betrachtet.

Fortsetzung der 4. Aufgabe:

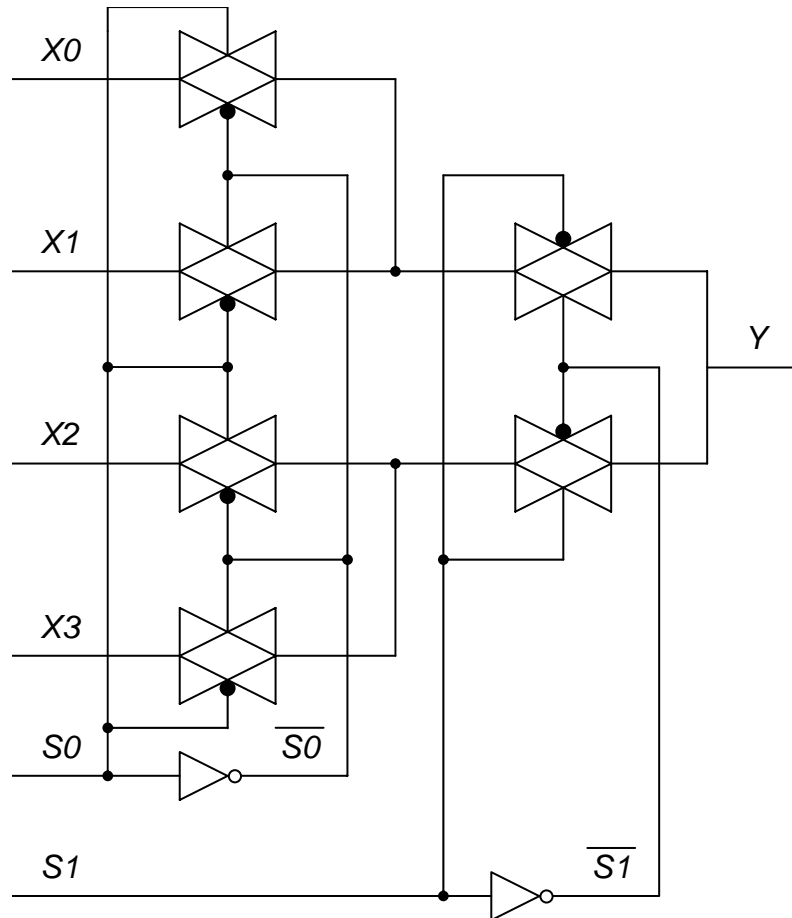


Abbildung 4.2: Schaltung mit Transferrgates

c) Vervollständigen Sie Tabelle 4.2 gemäß der Schaltung aus Abbildung 4.2!

S_0	S_1	Y
0	0	
0	1	
1	0	
1	1	

Tabelle 4.2: Zustandsbelegung

d) Welche Funktion wird durch die Schaltung in Abbildung 4.2 realisiert?

Aufgabe 5: Arithmetik-Schaltungen

10 Punkte

Die Funktionstabelle eines 1-bit-Halbsubtrahierers für zwei Binärzahlen a und b , mit der positiven Differenz d sowie dem „Borgebit“ e lautet:

a	b	d	e
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

Tabelle 5.1: Zustandstabelle eines Halbsubtrahierers

Dabei gilt:

Differenz: $d = a \oplus b = a \cdot \bar{b} + \bar{a} \cdot b$

Borgebit: $e = \bar{a} \cdot b$.

Im Folgenden soll daraus ein Vollsubtrahierer entwickelt werden. Dieser hat 3 Eingänge, bestehend aus dem Bit a , dem Bit b , welches von a abgezogen wird und dem Borgeübertrag c aus eventuell zuvor durchgeführten Subtraktionen. Die beiden Ausgänge stellen das Ergebnis der Subtraktion d und das Borgebit e dar.

- a) Vervollständigen Sie die Zustandstabelle 5.2 eines Vollsubtrahierers.

a	b	c	d	e
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Tabelle 5.2: Zustandstabelle eines Vollsubtrahierers

- b) Durch welche booleschen Ausdrücke können die Differenz d sowie das Borgebit e des Vollsubtrahierers beschrieben werden?

Fortsetzung der 5. Aufgabe:

- c) Der Vollsubtrahierer soll nun in Hardware umgesetzt werden. Ergänzen Sie hierzu in geeigneter Weise das Schaltnetz in Abbildung 5.1 durch Ausfüllen der Schaltungsblöcke!
(**Hinweis:** Der Vollsubtrahierer lässt sich einfach durch geeignete Kombination eines Halbaddierers und eines Halbsubtrahierers realisieren!)

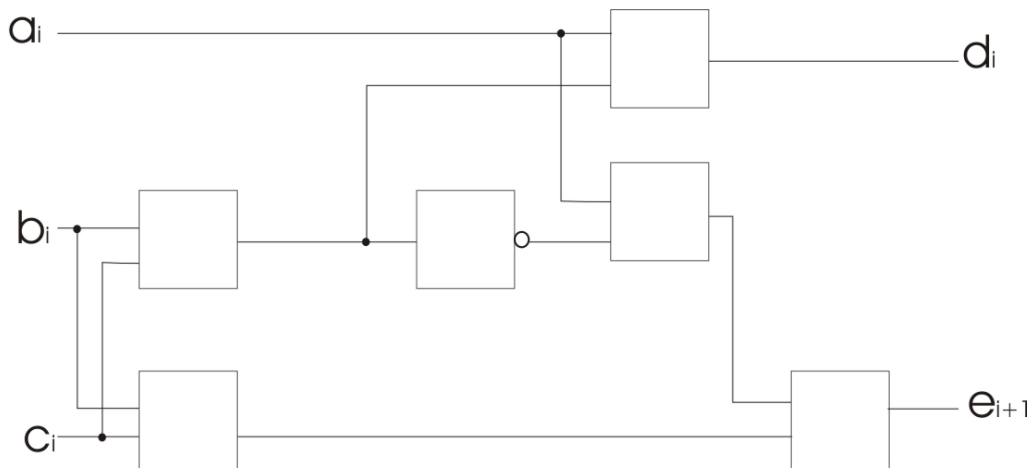


Abbildung 5.1: Schaltnetz eines Vollsubtrahierers

Aufgabe 6: Mikrocontrollerprogrammierung mit dem 80C51

10 Punkte

Im ersten Aufgabenteil soll ein Unterprogramm für den 80C51-Mikrocontroller entwickelt werden, dessen Flussdiagramm in Abbildung 6.1 gegeben ist. Dem Unterprogramm werden in den Registern $R0$, $R1$, $R2$ beliebige vorzeichenlose 8 bit-Dualzahlen übergeben.

Hinweis: In der beiliegenden Hilfsblattsammlung finden Sie eine Übersicht aller Assemblerbefehle des 80C51.

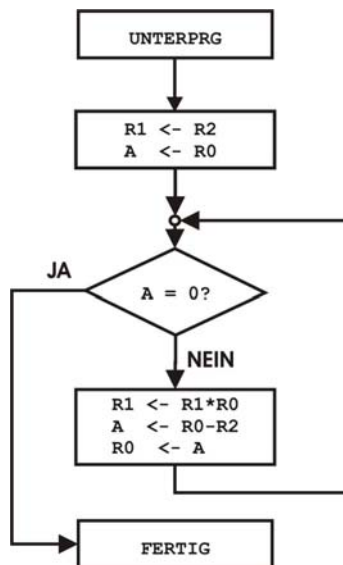


Abbildung 6.1: Flussdiagramm des Assemblerprogramms

- a) Schreiben Sie das entsprechende Assemblerprogramm für den in Abbildung 6.1 gegebenen Abschnitt! Vergessen Sie dabei nicht den Rücksprung ins Hauptprogramm und benutzen Sie die Bezeichnungen $R0$, $R1$, $R2$, A , *FERTIG* und *UNTERPRG* aus Abbildung 6.1.

UNTERPRG:

Fortsetzung der 6. Aufgabe:

- b) Welche Dualzahlen stehen nach Ablauf des Programms in $R0$, $R1$ und $R2$, wenn in den Registern zu Beginn des Programmsegments vorzeichenlose 8 bit-Dualzahlen mit folgenden Dezimalwerten stehen:

Vorher: $R0: 6_d$, $R1: 4_d$, $R2: 3_d$

Nachher:

- c) Welche arithmetische Operation wird durch das Programmsegment ausgeführt, wenn zu Anfang $R2$ den Wert 1_d hat und das Ergebnis in $R1$ in Abhängigkeit von $R0$ ausgedrückt wird?

Aufgabe 7: Timer/Counter des Mikrocontrollers 80C52

11 Punkte

Ein Mikrocontroller vom Typ 80C52 soll zur Überwachung eines Impulsgebers eingesetzt werden, der bis zu 20.000 Impulse pro Sekunde liefern kann. Dazu wird Timer 0 als Zähler verwendet, der die Impulse am T0-Pin jeweils während einer Zeit von 10 ms zählt. Timer 1 hat die Aufgabe, alle 10 ms einen Interrupt auszulösen. Die Taktfrequenz des Mikrocontrollers beträgt 12 MHz.

- a) Berechnen Sie den Reloadwert für Timer 1! Vernachlässigen Sie hierbei die Zeitverzögerung, die durch den Aufruf der Interruptroutine entsteht!
- b) Geben Sie in Tabelle 7.1 die notwendigen Einstellungen der Spezialfunktionsregister an, um die beiden Timer zu initialisieren und zu starten! Kennzeichnen Sie irrelevante Bits durch „X“!

Bit	7	6	5	4	3	2	1	0
IE								
TMOD								
TCON								

Tabelle 7.1: Belegung der Spezialfunktionsregister

In der Interruptroutine von Timer 1 wird der Zählerstand von Timer 0 (TL0) ausgelesen und in das Register R0 geschrieben. Wurde der Wert von 15.000 Impulsen pro Sekunde überschritten, wird der Portpin P1.0 auf Low gesetzt, ansonsten auf High. Außerdem muss Timer 1 neu geladen werden.

- c) Vervollständigen Sie das Flussdiagramm in Abbildung 7.1 für die Interruptroutine von Timer 1! Geben Sie dabei neben dem Kommentar jeweils auch den entsprechenden Assemblerbefehl an.

Fortsetzung der 7. Aufgabe:

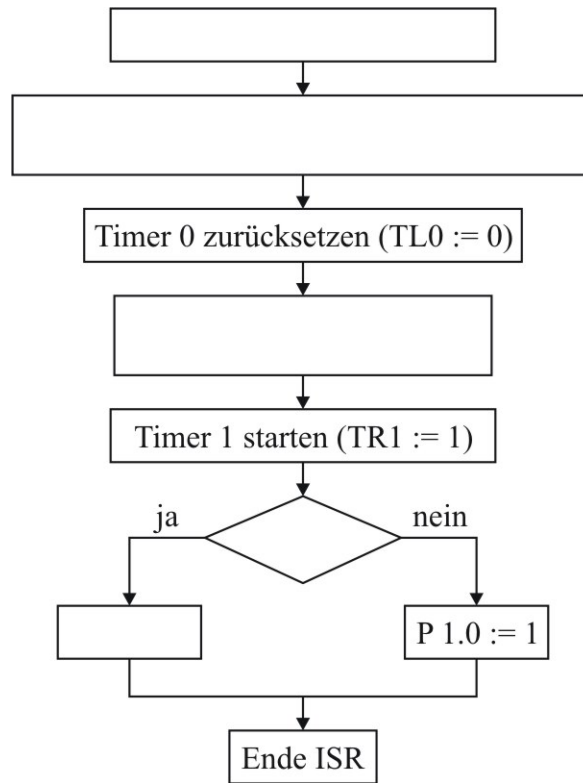


Abbildung 7.1: Flussdiagramm der Interruptroutine von Timer 1

Aufgabe 8: Architektur von DSPs und Analyse von Assemblercode

10 Punkte

Im Folgenden soll die Architektur von digitalen Signalprozessoren untersucht werden.

- a) Wie wird die Bus- und Speicherstruktur eines digitalen Signalprozessors bezeichnet?

- b) Skizzieren Sie die typische Bus- und Speicherstruktur eines digitalen Signalprozessors!

Gegeben sei nun der nachfolgende Befehl: `MOVE $100, M2`

- c) Welche Funktion haben die Register M_n ?

- d) Was wird durch den Befehl `MOVE $100, M2` bewirkt?

Fortsetzung der 8. Aufgabe:

Der X-Speicher eines DSP56001 sei gemäß Tabelle 8.1 belegt.

Adresse	Inhalt
\$14F	\$153
\$150	\$152
\$151	\$151
\$152	\$150
\$153	\$14F

Tabelle 8.1: Speicherbelegung des X-Speichers eines DSP56001

- e) Ergänzen Sie die freien Felder der nachfolgenden Tabelle 8.2!

(**Hinweis:** In der Tabelle bezeichnet $R0(S)$ den Startwert des Adresszeigers, $R0(E)$ den Endwert des Adresszeigers nach Abarbeitung des Befehls.)

	$R0(S)$	$N0$	$R0(E)$	A	Bezeichnung
MOVE X:(R0),A				\$150	
	\$153	\$2		\$14F	
	\$153	\$1		\$154	

Tabelle 8.2: Adressierungsarten eines DSP56001

Aufgabe 9: Beschreibung von Digitalschaltungen mit VHDL

10 Punkte

Gegeben sei das Blockschaltbild eines 4-zu-1-Multiplexers nach Abbildung 9.1 mit den Steuerleitungen $S0$ und $S1$, den Eingängen $E0$, $E1$, $E2$ und $E3$, sowie dem Ausgang A . Die Wahrheitstabelle des 4-zu-1-Multiplexers ist in Tabelle 9.1 gegeben.

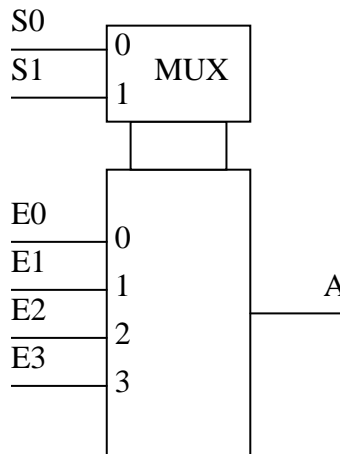


Abbildung 9.1: Blockschaltbild eines 4-zu-1-Multiplexers

S0	S1	A
0	0	E0
0	1	E1
1	0	E2
1	1	E3

Tabelle 9.1: Zustandsbelegung des 4-zu-1-Multiplexers

Vervollständigen Sie die nachfolgende ENTITY und die zugehörige ARCHITECTURE, so dass ein 4-zu-1-Multiplexer beschrieben wird! Verwenden Sie ausschließlich die Ein- und Ausgänge, sowie die Steuerleitungen entsprechend Abbildung 9.1! Die Zuweisung der Signale soll innerhalb des Prozesses geschehen!

```
library ieee;
use ieee.std_logic_1164.all;
use ieee.std_logic_arith.all;
use ieee.std_logic_unsigned.all;
```

```
entity mux_4_1 is
```

```
end mux_4_1;
```


Fortsetzung der 9. Aufgabe:

```
architecture mux_4_1_arch of mux_4_1 is
begin
    process(
        )

end mux_4_1_arch;
```

Aufgabe 10: Analyse von VHDL-Modellen

11 Punkte

VHDL stellt neben Signalen auch Variablen zur Verfügung. Im Folgenden sollen zwei VHDL-Codes auf ihr Zeitverhalten in entsprechenden Simulationen untersucht werden.

- a) Vervollständigen Sie Tabelle 10.1, indem Sie den nachfolgenden VHDL-Code untersuchen und die jeweils resultierenden Werte der Variablen in die Tabelle eintragen!
(**Hinweis:** Die Variable x wird extern entsprechend Tabelle 10.1 mit einer Taktrückflanke gesetzt und dem Prozess übergeben!)


```
process (clk)

variable y1, y2, y3: integer;

begin

    if (clk'event and clk='0') then
        y1 := y3;
        y2 := y1+1;
        y3 := x;
    end if;

end process;
```



x	7	3	6	1	8
y1					
y2					
y3					

Tabelle 10.1: Timingdiagramm unter Verwendung von Variablen

- b) Vervollständigen Sie Tabelle 10.2, indem Sie den nachfolgenden VHDL-Code untersuchen und die jeweils resultierenden Werte der Signale in die Tabelle eintragen!
(**Hinweis:** Das Signal x wird extern entsprechend Tabelle 10.2 mit einer Taktrückflanke gesetzt und dem Prozess übergeben!)

```
signal y1, y2, y3: integer;

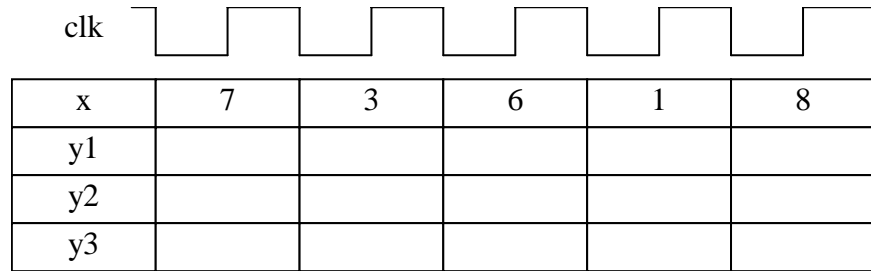
process (clk)

begin

    if (clk'event and clk='0') then
        y1 <= y3;
        y2 <= y1+1;
        y3 <= x;
    end if;

end process;
```

Fortsetzung der 10. Aufgabe:



clk	7	3	6	1	8
x	7	3	6	1	8
y1					
y2					
y3					

Tabelle 10.2: Timingdiagramm unter Verwendung von Signalen

- c) Nennen Sie drei Objekte, die VHDL zur Verfügung stellt!
- d) Nach welcher Anweisung kann in VHDL eine „Sensitivity-Liste“ folgen? Wozu wird diese Liste verwendet?
- e) Was wird mit VHDL beschrieben, d.h. wozu wird diese Sprache vorzugsweise eingesetzt? Welche Hauptunterschiede gibt es im Vergleich zu herkömmlichen Programmiersprachen wie C oder Pascal?