Universität Paderborn

Institut *Elektrotechnik und Informationstechnik*

Fachgebiet *Datentechnik*

Prof. Sybille Hellebrand

Klausur

Digitaltechnik / Grundlagen Technische Informatik

3. August 2009

Punkteverteilung							
Aufgabe	1	2	3	4	5	6	Σ
maximale Punkte	15	10	20	10	15	20	90
erreichte Punkte							

Note:	

Aufkleber

Name:	
Matrikelnummer:	
Studienrichtung:	

Für die Lösung der Klausuraufgaben sind ausschließlich die Aufgabenblätter zu verwenden. Lösungsangaben außerhalb der Aufgabenblätter („Schmierzettel“, etc.) werden bei der Bewertung nicht berücksichtigt!

Mit Bleistift oder der Korrekturfarbe rot angefertigte Lösungen werden nicht bewertet!

Die Verwendung von „Tipp-Ex“ oder „Tintenkiller“ ist untersagt.

Es sind keine Hilfsmittel zugelassen!

Aufgabe 1: (VHDL) 15 Punkte

Ein linear rückgekoppeltes Schieberegister (engl. Linear Feedback Shift Register, LFSR) ist ein Schieberegister, bei dem in jedem Takt das nachzuschiebende Bit aus einer XOR-Verknüpfung mehrerer anderer Stellen des Schieberegisters berechnet wird.
Abbildung 1 zeigt ein 8-Bit Linear Feedback Shift Register mit den Bit-Positionen Y(7)..Y(0). Bei einem Reset wird das Register auf den Wert "01011011" gesetzt. In jedem Takt wird das Register um ein Bit nach rechts geschoben. Das nachzuschiebenne Bit $Y(7)$ bekommt den Wert Y(4)⊕Y(2)⊕Y(0), wobei ⊕ der XOR Operator ist.

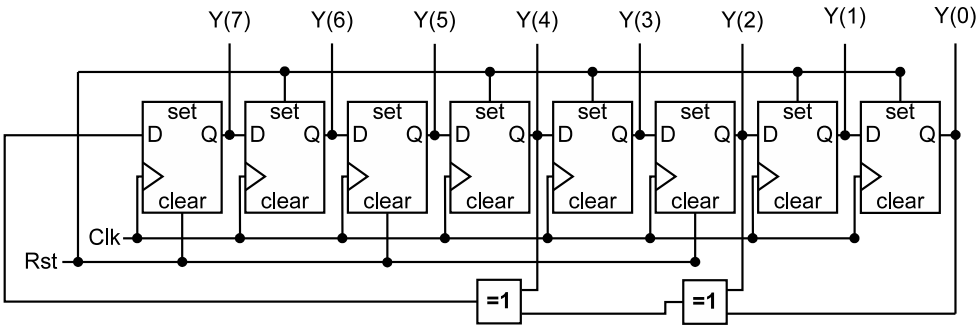
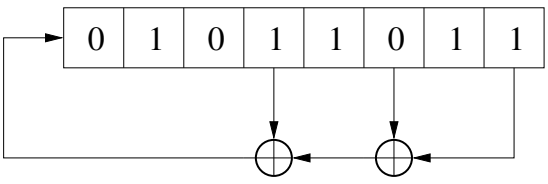


Abbildung 1: Linear Feedback Shift Register

a) In Abbildung 2 sehen Sie den Zustand des 8-Bit LFSRs nach einem Reset. Ergänzen Sie in der nebenstehenden Liste den Zustand des Registers nach einem, zwei, drei und vier Takten.

b) Weiter unten finden sie ein Code-Fragment für das VHDL Modul LFSR. Ergänzen Sie sowohl die `entity` als auch die `architecture` so, dass das Modul LFSR die Funktionalität der Schaltung aus Abbildung 1 implementiert.



t0: Y= „01011011“

t1: Y=

t2: Y=

t3: Y=

t4: Y=

Abbildung 2: Startzustand und folgende vier Ausgaben des LFSRs

Entity LFSR

```
library IEEE;
use IEEE.std_logic_1164.ALL;

entity LFSR is
  port(

    Clk ____________________________________________

    Rst ____________________________________________

    Y   ____________________________________________
  );
end LFSR;

architecture Functionality of LFSR is

  ____________________________________________________________

  ____________________________________________________________
begin

  Y <= ___________________________________________

  lfsr_proc : process( ____________________________________ )
  begin

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________
  end process;

  ____________________________________________________________

  ____________________________________________________________
end architecture Functionality;
```

Entity LFSR (Ersatz)

```
library IEEE;
use IEEE.std_logic_1164.ALL;

entity LFSR is
  port(

    Clk ____________________________________________

    Rst ____________________________________________

    Y   ____________________________________________
  );
end LFSR;

architecture Functionality of LFSR is

  ____________________________________________________________

  ____________________________________________________________
begin

  Y <= ___________________________________________

  lfsr_proc : process( ____________________________________ )
  begin

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  ____________________________________________________________

  end process;

  ____________________________________________________________

  ____________________________________________________________
end architecture Functionality;
```

(ungültige Lösung streichen!)

Aufgabe 2: (Zahldarstellung) 10 Punkte

a) Gegeben sei die Zahl 1101 in 4-bit Zweierkomplementdarstellung. Welchen Wert hat die Zahl? Kreuzen Sie das richtige Ergebnis in folgender Tabelle an.

- ☐ 13
- ☐ -5
- ☐ -3
- ☐ Keine Lösung ist richtig

b) Gegeben sei wieder die Zahl 1101 in 4-bit Zweierkomplementdarstellung. Die Zahldarstellung soll jetzt auf eine 8-bit Zweierkomplementdarstellung erweitert werden. Wie sieht die korrekte Zahldarstellung aus? Kreuzen Sie das richtige Ergebnis in folgender Tabelle an.

- ☐ 00001101
- ☐ 11111101
- ☐ 10001101
- ☐ Keine Lösung ist richtig

c) Gegeben sei wieder die Zahl 1101 in 4-bit Zweierkomplementdarstellung. Wie sieht die Zahl im 4-bit Vorzeichen/Betrag Format aus?

---

d) Auch Brüche können als Zweierkomplementzahlen dargestellt werden. Bei einem Zweierkomplementbruch $a_0, a_{-1}a_{-2}a_{-3}$ ... wird die Komplementzahl durch bitweises Invertieren und Addition einer 1 am Bit mit der niedrigsten Wertigkeit berechnet. Für den Bruch - 7/8 ergibt sich die Darstellung:

- ☐ 1,111
- ☐ 1,001
- ☐ 1,100
- ☐ Keine Antwort ist richtig.

e) Gegeben sei der Zweierkomplementbruch 1,111. Wie sieht der Zweierkomplementbruch aus, wenn statt der 4-Bit Darstellung eine 8-Bit Darstellung verwendet werden soll?

- ☐ 1,1111111
- ☐ 1,1110000
- ☐ 1,0001111
- ☐ Keine Antwort ist richtig.

f) Es werden nun die beiden Zahlen x = 1101 und y = 1110 in 4-bit Zweierkomplementdarstellung betrachtet. Bei der Addition der beiden Zahlen ergibt sich x + y = 11011. Kreuzen Sie alle richtigen Aussagen in der Tabelle an.

- ☐ Das Ergebnis liegt außerhalb des gültigen Zahlenbereichs (wegen Überlauf).
- ☐ In 4-bit Zweierkomplementdarstellung lautet das Ergebnis 1011.
- ☐ In 4-bit Vorzeichen/Betrag Darstellung ist das Ergebnis 1101
- ☐ Keine Antwort ist richtig

g) Bei Umrechnung des Dezimalbruchs 0,2 in einen Binärbruch ergibt sich folgende Darstellung:

- ☐ 0,10
- ☐ 0,001100110011....
- ☐ 0,00111000111....
- ☐ Keine Antwort ist richtig.

h) Gegeben sei die Dezimalzahl 1,5. Wie sieht die Zahldarstellung im IEEE Gleitkommaformat aus?

Vorzeichen (1 bit):

Mantisse (23 bit):

Exponent (8 bit, Exzessdarstellung mit Verschiebungskonstante 127):

Aufgabe 3: (Automat) 20 Punkte

Gegeben ist ein durch Abbildung 3 spezifizierter Mealy-Automat.

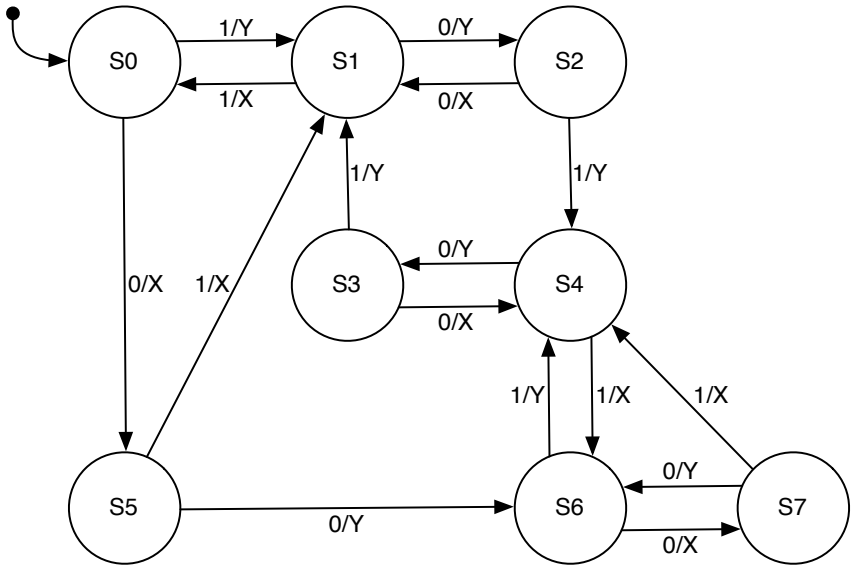


Abbildung 3: Mealy-Automat

1. Füllen Sie die gegebene Automatentafel zu dem in Abbildung 3 spezifizierten Mealy-Automaten.

   Automatentafel:

δ/λ	0	1

2. Bestimmen Sie den äquivalenten zustandsminimalen Mealy-Automaten mit Hilfe des Ginsburg/Huffman Verfahrens (Benutzen Sie die gegebenen Tabellen).

δ/λ	0	1

δ/λ	0	1

δ/λ	0	1

δ/λ	0	1

Die Automatentafel des zustandsminimierten Automat:

δ/λ	0	1

3. Zeichen Sie den äquivalenten zustandsminimalen Mealy-Automaten in Abbildung 4.

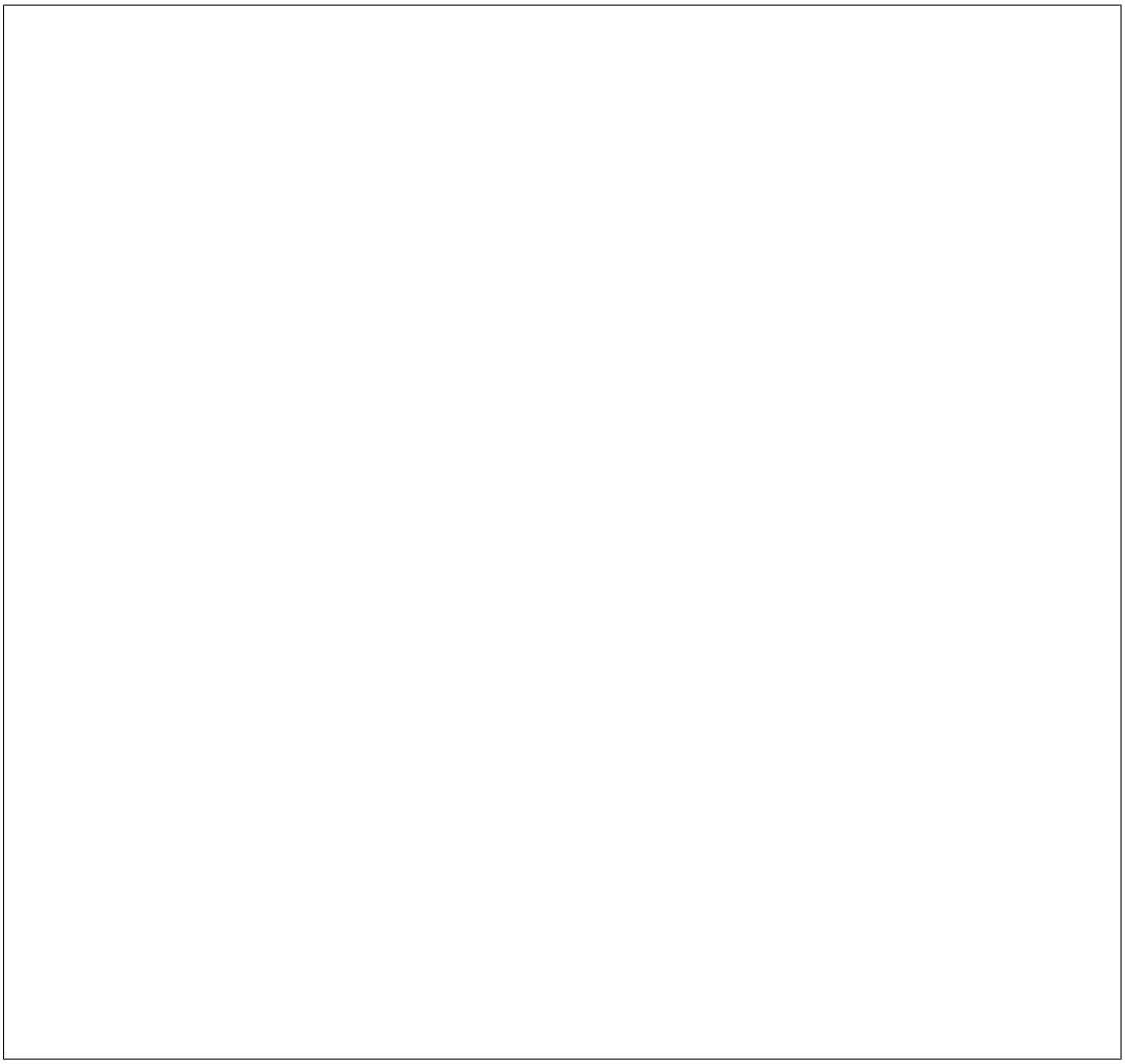


Abbildung 4: Äquivalenter Moore-Automat

4. Wandeln Sie den Mealy-Automaten in einen Moore-Automaten um und zeichnen Sie diesen in Abbildung 5.

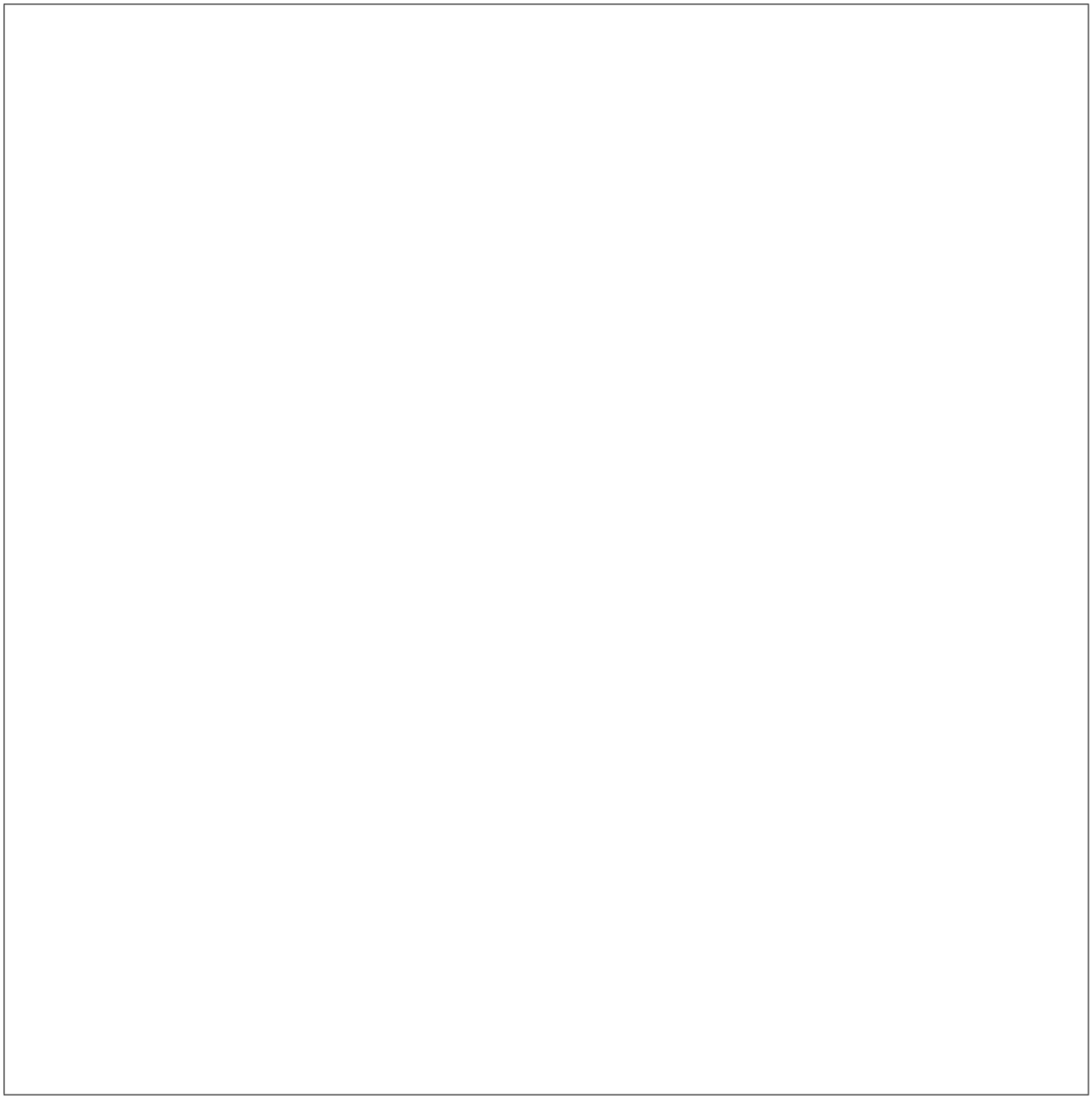


Abbildung 5: Äquivalenter Moore-Automat

Aufgabe 4: (Zähler) 10 Punkte

Entwickeln Sie einen Gray Code Zähler, der von **0 bis 5** zählt. Der Gray Code hat die Eigenschaft, dass sich bei jedem Zählschritt nur ein Bit ändert. Die Zahlencodierung kann nachfolgender Tabelle entnommen werden.

Dezimal	s_2	s_1	s_0
0	0	0	0
1	0	0	1
2	0	1	1
3	0	1	0
4	1	1	0
5	1	1	1

Gray Code Zahlencodierung

a) Spezifizieren Sie die Zustandsüberganfunktion $\delta : S \to S$, so dass der Zähler vorwärts von **0 bis 5** zählt und anschließend wieder bei 0 beginnt. Für die Ausgabefunktion soll $\lambda(s) = s$ gelten.

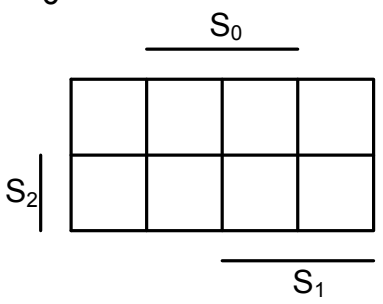
s			δ		
s_2	s_1	s_0	s_2^+	s_1^+	s_0^+
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

s			δ		
s_2	s_1	s_0	s_2^+	s_1^+	s_0^+
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

Ersatztabelle: (ungültige Lösung streichen!)

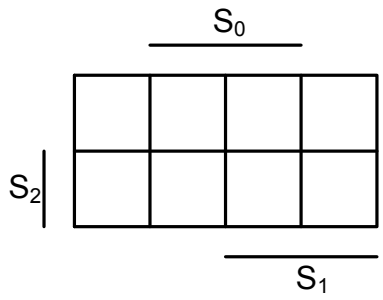
b) Vereinfachen Sie die Logik der Zustandübergangsfunktionen mit Hilfe von KV-Diagrammen und schreiben Sie die Lösungen in SOP-Form (**S**um **O**f **P**roducts) auf.

S_0^+:



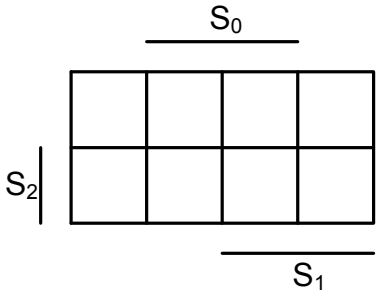
$S_0^+ =$ _____

S_1^+:



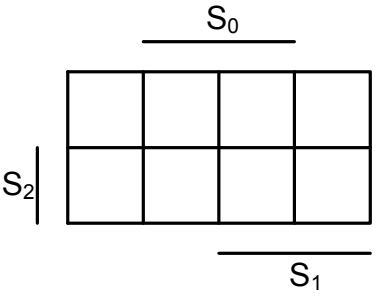
$S_1^+ =$ _____

S_2^+:



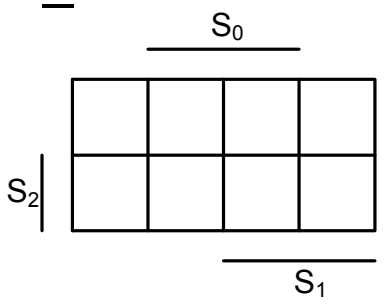
$S_2^+ =$ _____

$S\_^+$:



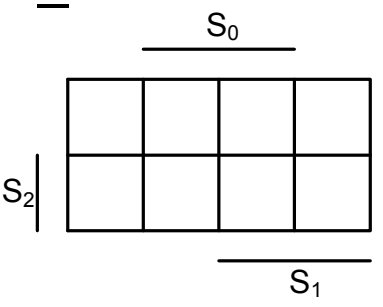
$S\_^+ =$ _____

$S\_^+$:



$S\_^+ =$ _____

$S\_^+$:



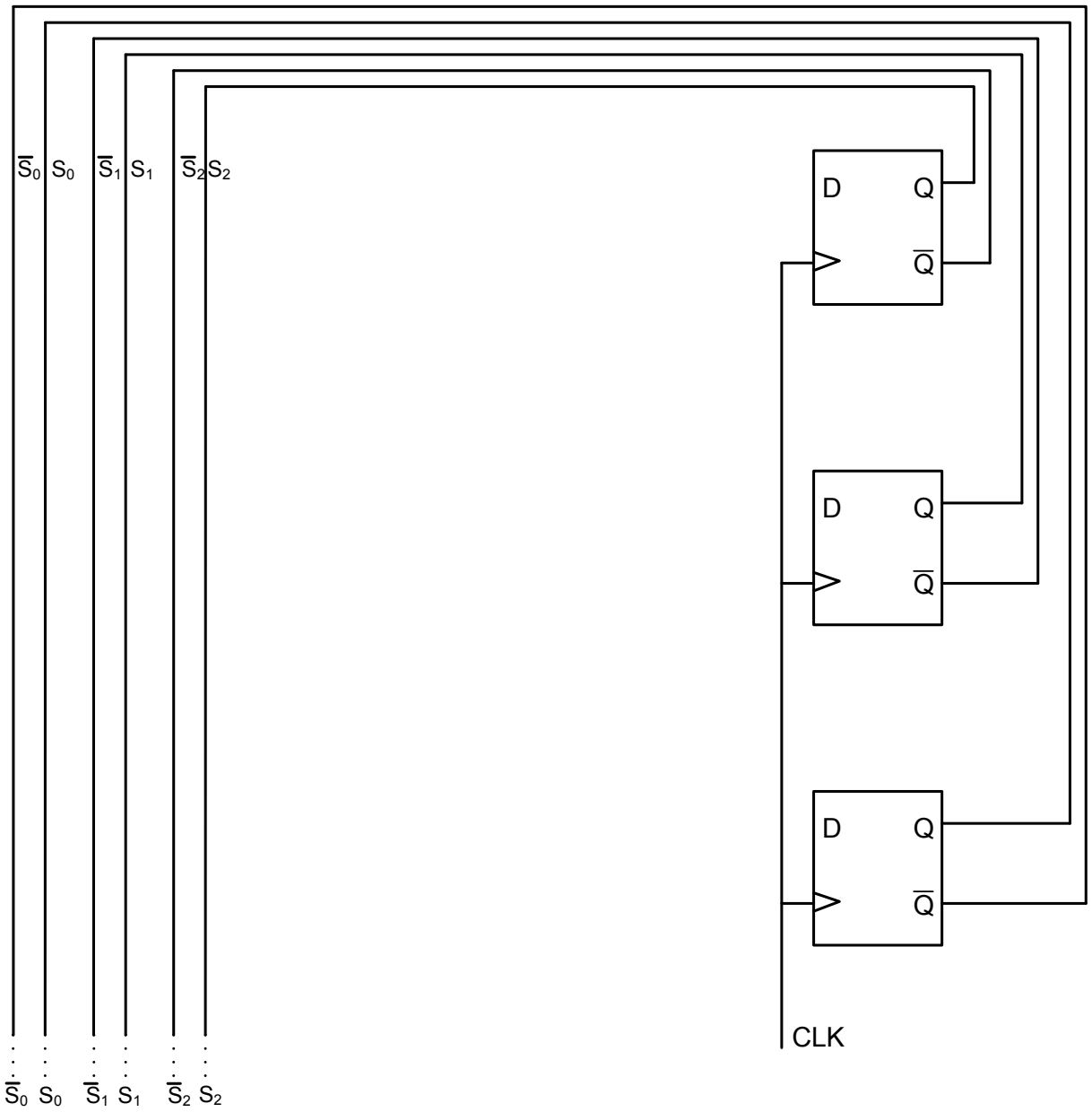
$S\_^+ =$ _____

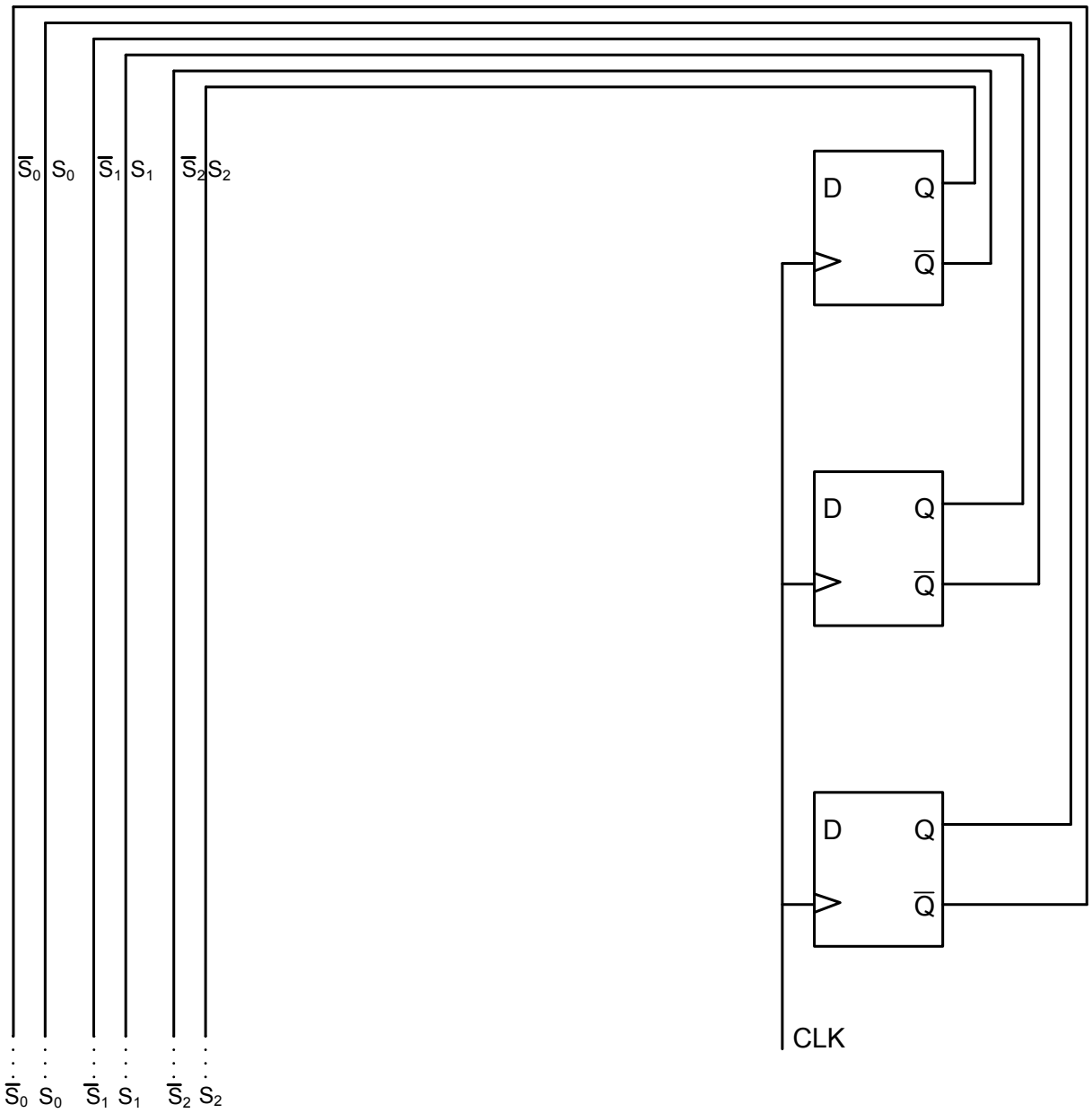
ungültige Lösungen streichen!

c) Zeichnen Sie die Logikschaltung für die im Punkt b) bestimmten minimalen SOP-Formen. Verwenden Sie dafür das nachfolgende Gerüst. Wieviele Gatttereingänge und wieviele Logikstufen benötigt die gesamte Schaltung?

Anzahl Gattereingänge: _____

Anzahl Logikstufen: _____





Ersatzgrafik: (ungültige Lösung streichen!)

Aufgabe 5: (Quine-McCluskey) 15 Punkte

a) Gegeben sei die Einsstellenmenge:

$$\mathcal{E} = \{0000, 0001, 0101, 1000, 1001, 1011, 1101, 1110, 1111\}$$

für $z = (x_3, x_2, x_1, x_0)$ Ermitteln Sie mit Hilfe des Quine-McCluskey Verfahrens die Menge $\mathcal{P}$ der Primimplikanten.

b) Gegeben sei die Menge der Primimplikanten durch:

$$\mathcal{P} = \{0--0, 0-0-, 01--, 10-1, -001\}$$

Bestimmen Sie mit Hilfe nachfolgender Tabelle die minimale SOP-Form (**S**um **O**f **P**roducts).

P_0									
P_1									
P_2									
P_3									
P_4									

Tabelle zu b)

P_0									
P_1									
P_2									
P_3									
P_4									

Ersatztabelle zu b)
(ungültige Lösung streichen!)

minimale SOP-Form:_____

Aufgabe 6: (RTL) 20 Punkte

In dieser Aufgabe soll eine Schaltung zur sequenziellen Berechnung des Skalarprodukts $\langle \vec{x}, \vec{y} \rangle := \sum_{i=0}^{n-1} x_i \cdot y_i$ zweier ganzzahliger n-dimensionaler Vektoren $\vec{x} = (x_0, ..., x_{n-1})$ und $\vec{y} = (y_0, ..., y_{n-1})$ entworfen werden.
Zur Realisierung des Datenpfads stehen Ihnen außer den Registern (x, y, result) genau ein Multiplizierer (mult) und ein Addierer (add) zur Verfügung. Die Eingabedaten werden der Reihe nach über einen 8-Bit-Bus (INBUS) zur Verfügung gestellt. Nach dem Ende der Berechnung soll das Ergebnis an einen zweiten Bus (OUTBUS) ausgegeben werden.

a) Vervollständigen Sie den Datenpfad in Abbildung 6 entsprechend.

b) Zeichen Sie für $n = 20$ die notwendigen Busbreiten ein, so dass es zu keinem Überlauf kommt und begründen Sie Ihre Entscheidung. Wie viele Bits muss dann das Register `result` speichen können.
Hinweis: Der Addierer kann Operanden mit unteschiedlicher Busbreite korrekt addieren.

c) Nun soll die Steuerung entworfen werden. Zeichen Sie dazu in den Datenpfad die notwendigen Kontrollpunkte und eventuell weitere notwendige Komponenten (z.B. Zähler) ein und benennen Sie alle Signale.

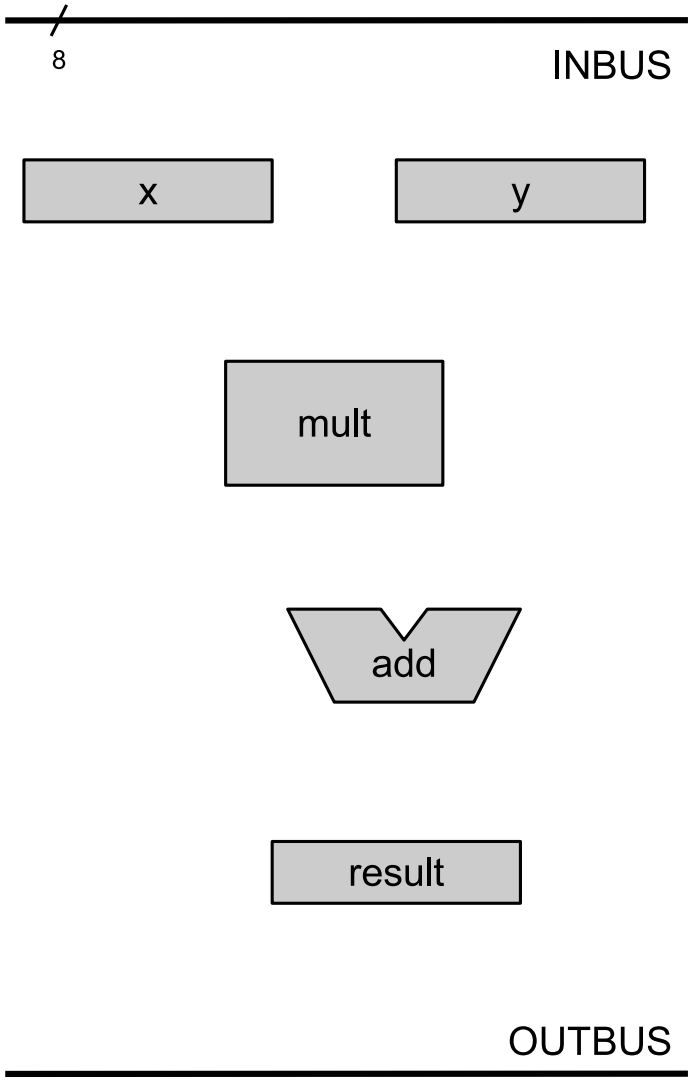


Abbildung 6: Blockschaltbild des Datenpfads

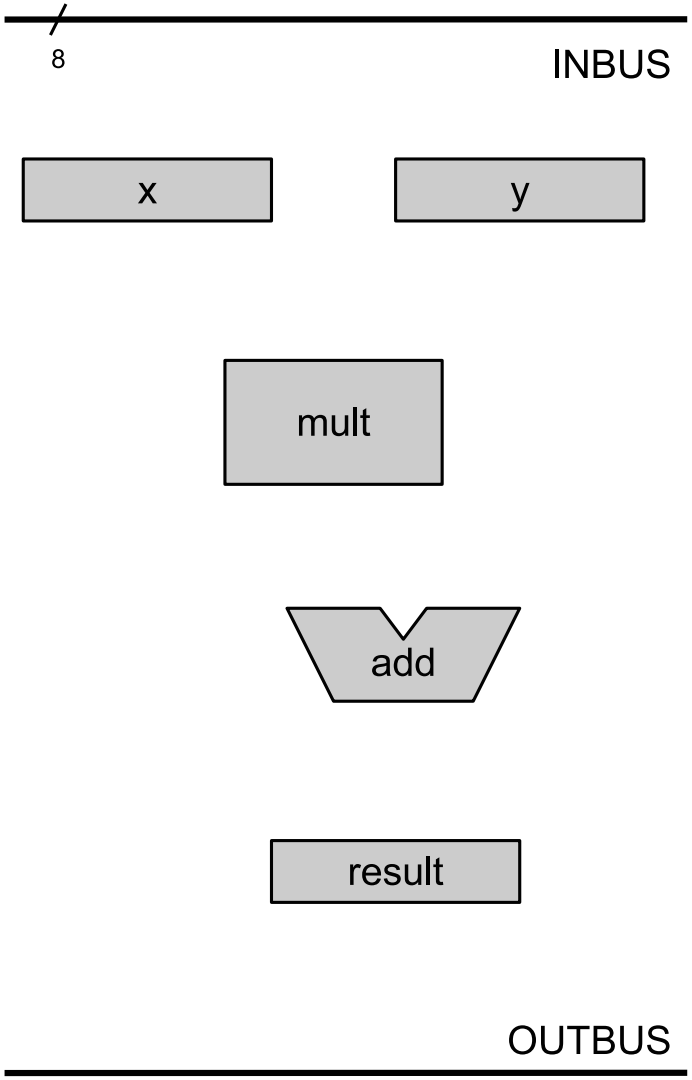


Abbildung 7: Blockschaltbild des Datenpfads (Ersatz)

(ungültige Lösung streichen!)

d) Zeichnen Sie dann den Zustandsübergangsgraph in den dafür vorgesehenen Freiraum (Abb. 8).
Hinweis: Sie können davon ausgehen, dass $n = 20$ fest vorgegeben ist und nicht eingelesen werden muss.

Abbildung 8: Zustandsübergangsgraph

Für eine andere Komponente sei bereits die Spezifikation der Steuerung gegeben (vgl. Abb. 9 und Abb. 10)

Abbildung 9: Steuerung des Datenpfads

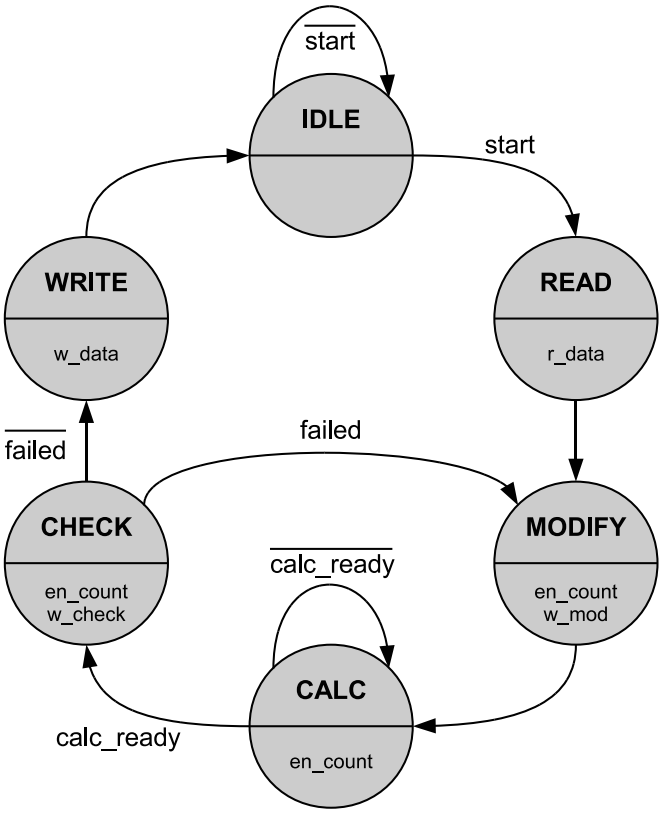


Abbildung 10: Zustandsübergangsgraph

Die Steuerung soll als μ-programmierbare Steuerung implementiert werden. Beantworten Sie zunächst die folgenden Fragen, um das μ-Befehlsformat festzulegen:

e) Wie viele Adressbits werden benötigt?

---

f) Wie viele Bits müssen für die Steuersignale reserviert werden?

---

g) Welche Sprungbedingungen gibt es? Geben Sie im folgenden für die Sprungbedingung jeweils eine Kodierung mit einer minimalen Anzahl von Bits an.

Sprungbedingung	Kodierung

h) Vervollständigen Sie dann den Speicherinhalt in Abbildung 11 für das μProgramm. Benennen Sie auch die Steursignale in der Kopfzeile der Tabelle.

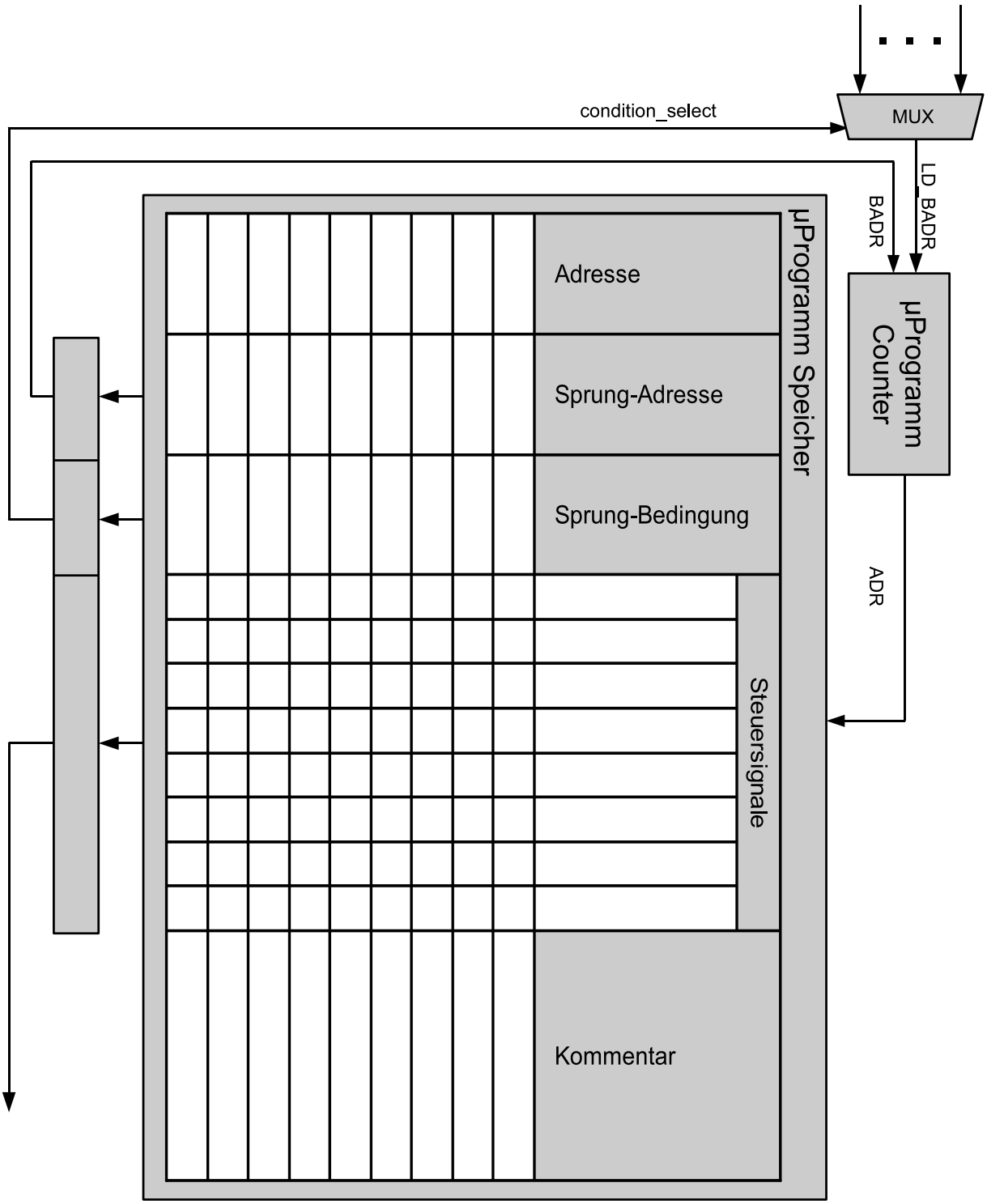


Abbildung 11: Inhalt des Mikroprogrammspeichers

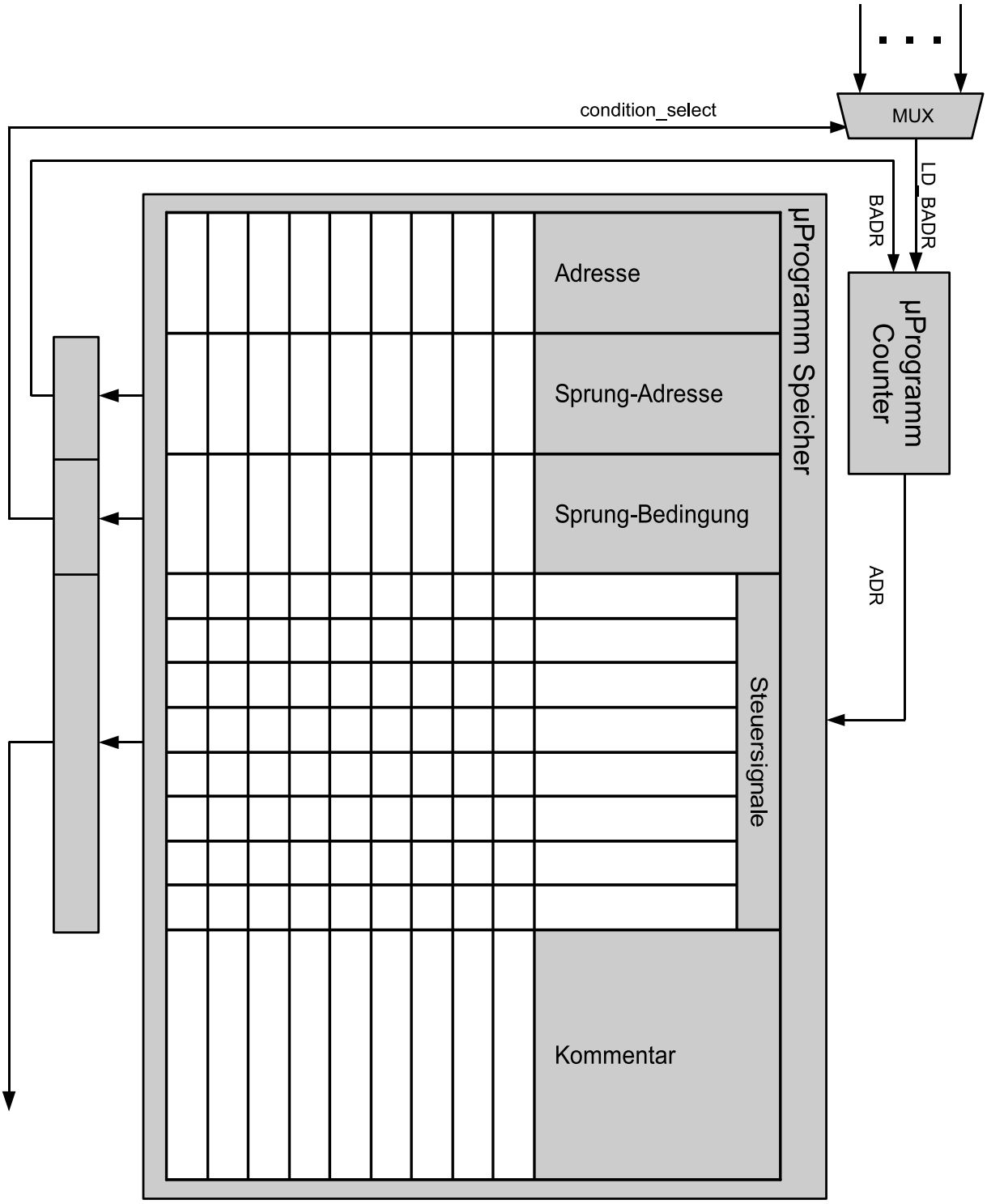


Abbildung 12: Inhalt des Mikroprogrammspeichers (Ersatz)
(ungültige Lösung streichen!)