Aufgabe 1

(a) Ein 10-Bit-Carry-Select-Addierer mit der Aufteilung 5-5 ist in Abbildung 1 dargestellt:

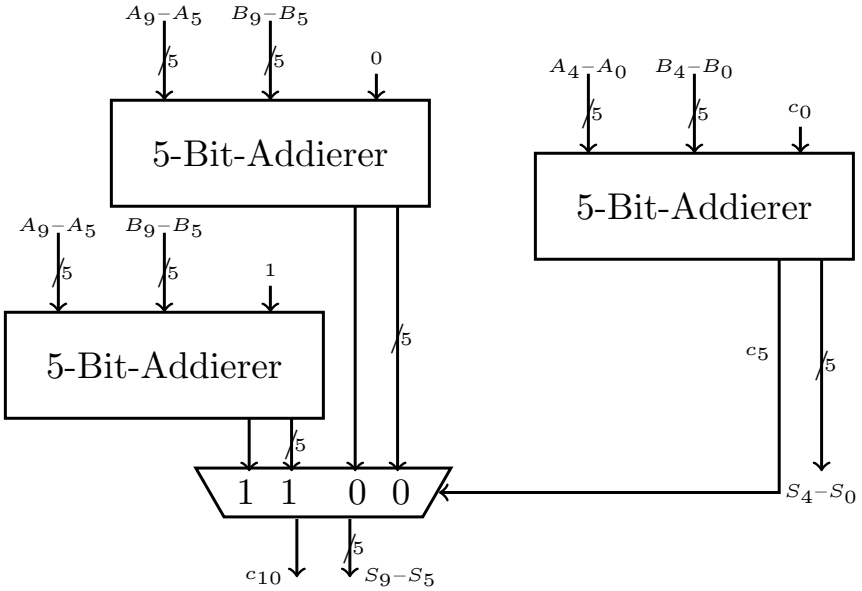

Abbildung 1: 10-Bit-Carry-Select-Addierer

Gesamtfläche:

$$A = \underbrace{1 \cdot 5FE}_{\text{Addierer Bit 0-4}} + \underbrace{2 \cdot 5FE}_{\text{Addierer Bit 5-9}} + \underbrace{5 \cdot 1FE}_{\text{Multiplexer Bit 5-9}} + \underbrace{1 \cdot 1FE}_{\text{Multiplexer Carry}} = 21FE$$

Gesamtzeit:

$$T = \underbrace{5ZE}_{\text{alle 5-Bit-Addierer}} + \underbrace{1ZE}_{\text{Multiplexer}} = 6ZE$$

Zeit-Flächen-Produkt:

$$AT = 21FE \cdot 6ZE = 126FEZE.$$

(b) Ein 10-Bit-Carry-Select-Addierer mit der Aufteilung 3-3-4 ist in Abbildung 2 dargestellt.
Gesamtfläche:

$$A = \underbrace{1 \cdot 3FE}_{\text{Addierer Bit 0-2}} + \underbrace{2 \cdot 3FE}_{\text{Addierer Bit 3-5}} + \underbrace{2 \cdot 4FE}_{\text{Addierer Bit 6-9}} + \underbrace{7 \cdot 1FE}_{\text{Multiplexer Bit 3-9}} + \underbrace{2 \cdot 1FE}_{\text{Multiplexer Carry}}$$
$$= 26FE$$

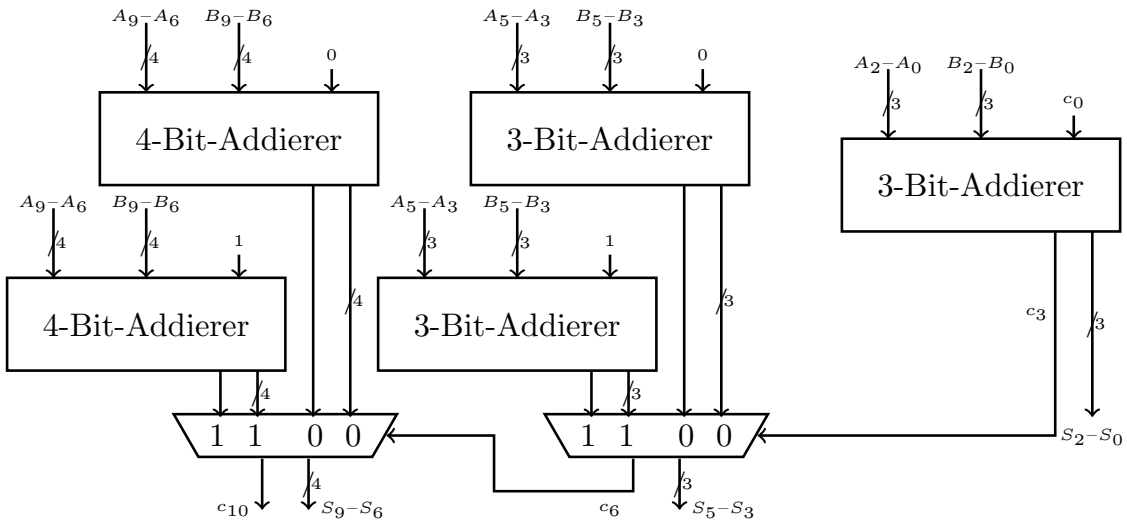

Abbildung 2: 10-Bit-Carry-Select-Addierer

Gesamtzeit:

$$T = \underbrace{3ZE}_{\text{3-Bit-Addierer}} + \underbrace{1ZE}_{\substack{\text{1. Gruppe Multiplexer} \\ \text{(\&4-Bit-Addierer)}}} + \underbrace{1ZE}_{\text{2. Gruppe Multiplexer}} = 5ZE$$

Zeit-Flächen-Produkt:

$$AT = 26FE \cdot 5ZE = 130FEZE$$

Aufgabe 2

(a) **Algorithmus von McClusky:**
Der Algorithmus von McCluskey liefert durch wiederholte Anwendung der ersten und zweiten Vereinfachungsregel ausgehend von der kanonischen Normalform:

$$
\begin{aligned}
f &= x_3\,x_2\,x_0 + x_3\,x_2\,\overline{x_0} + x_3\,\overline{x_2}\,x_1 + \overline{x_3}\,x_2\,x_1 \\
&= x_3\,x_2\overset{1}{x_1}\,x_0 + x_3\,x_2\overset{2}{\overline{x_1}}\,x_0 + x_3\overset{3}{x_2}\,x_1\,\overline{x_0} + x_3\,x_2\overset{4}{\overline{x_1}}\,\overline{x_0} \\
&+ x_3\overset{5}{\overline{x_2}}\,x_1\,x_0 + x_3\overset{6}{\overline{x_2}}\,x_1\,\overline{x_0} + \overline{x_3}\,x_2\overset{7}{x_1}\,x_0 + \overline{x_3}\,x_2\overset{8}{x_1}\,\overline{x_0} \\
&= x_3\overset{1,2}{x_2}\,x_0 + x_3\overset{1,3}{x_2}\,x_1 + x_3\overset{1,5}{x_1}\,x_0 + x_2\overset{1,7}{x_1}\,x_0 + x_3\,x_2\overset{2,4}{\overline{x_1}} \\
&+ x_3\,x_2\overset{3,4}{\overline{x_0}} + x_3\,x_1\overset{3,6}{\overline{x_0}} + x_2\,x_1\overset{3,8}{\overline{x_0}} + x_3\overset{5,6}{\overline{x_2}}\,x_1 + \overline{x_3}\,x_2\overset{7,8}{x_1} \\
&= [x_3\overset{1}{x_2}\,x_1 + x_3\,x_2\overset{2}{\overline{x_1}} + x_3\overset{3}{\overline{x_2}}\,x_1 + \overline{x_3}\,x_2\overset{4}{x_1}] + [x_3\overset{5}{x_2}\,x_0 + x_3\,x_2\overset{6}{\overline{x_0}}] \\
&+ [x_3\overset{7}{x_1}\,x_0 + x_3\,x_1\overset{8}{\overline{x_0}}] + [x_2\overset{9}{x_1}\,x_0 + x_2\,x_1\overset{10}{\overline{x_0}}] \\
&= [x_3\overset{1,2}{x_2} + x_3\overset{1,3}{x_1} + x_2\overset{1,4}{x_1}] + [x_3\overset{5,6}{x_2}] + [x_3\overset{7,8}{x_1}] + [x_2\overset{9,10}{x_1}] \\
&= x_3\,x_2 + x_3\,x_1 + x_2\,x_1
\end{aligned}
$$

Verfahren von Quine:
Es wird eine Tabelle erstellt, die für jeden Minterm in der DNF eine Zeile und für jeden Primterm, der durch das Verfahren von McClusky erzeugt worden ist, eine Spalte hat. In das Feld in der Zeile des Minterms M und der Spalte des Primterms P wird eine 1 eingetragen, wenn aus $M = 1$ folgt $P = 1$, also wenn P M überdeckt. Dann werden die dominanten Zeilen gesucht, also Zeilen, in denen nur eine einzige 1 steht. Diese Einsen in den dominanten Zeilen werden rot markiert und die Spalten, in denen rot markierte Einsen stehen, werden grau markiert. Es entsteht folgende Tabelle:

	$x_3\,x_2$	$x_3\,x_1$	$x_2\,x_1$
$x_3\,x_2\,x_1\,x_0$	1	1	1
$x_3\,x_2\,\overline{x_1}\,x_0$	1		
$x_3\,x_2\,x_1\,\overline{x_0}$	1	1	1
$x_3\,x_2\,\overline{x_1}\,\overline{x_0}$	1		
$x_3\,\overline{x_2}\,x_1\,x_0$		1	
$x_3\,\overline{x_2}\,x_1\,\overline{x_0}$		1	
$\overline{x_3}\,x_2\,x_1\,x_0$			1
$\overline{x_3}\,x_2\,x_1\,\overline{x_0}$			1

Da alle Spalten markiert wurden, wird das Verfahren beendet:

$$f = x_3\,x_2 + x_3\,x_1 + x_2\,x_1$$

(b) **CMOS-Komplexgatter:**
Um den n-Block mit der invertierten Funktion zu belegen, muss die Funktion invertiert werden. Durch Anwendung der Regeln von De Morgan (Boolsche Algebra Satz 11 und Satz 12) erhält man:

$$\begin{aligned}\overline{f} &= \overline{x_3\,x_2 + x_3\,x_1 + x_2\,x_1} \\ &= (\overline{x_3} + \overline{x_2}) \cdot (\overline{x_3} + \overline{x_1}) \cdot (\overline{x_2} + \overline{x_1})\end{aligned}$$

Ein OR-Gatter realisieren wir durch Parallelschaltung von Transistoren, ein AND-Gatter durch Reihenschaltung:

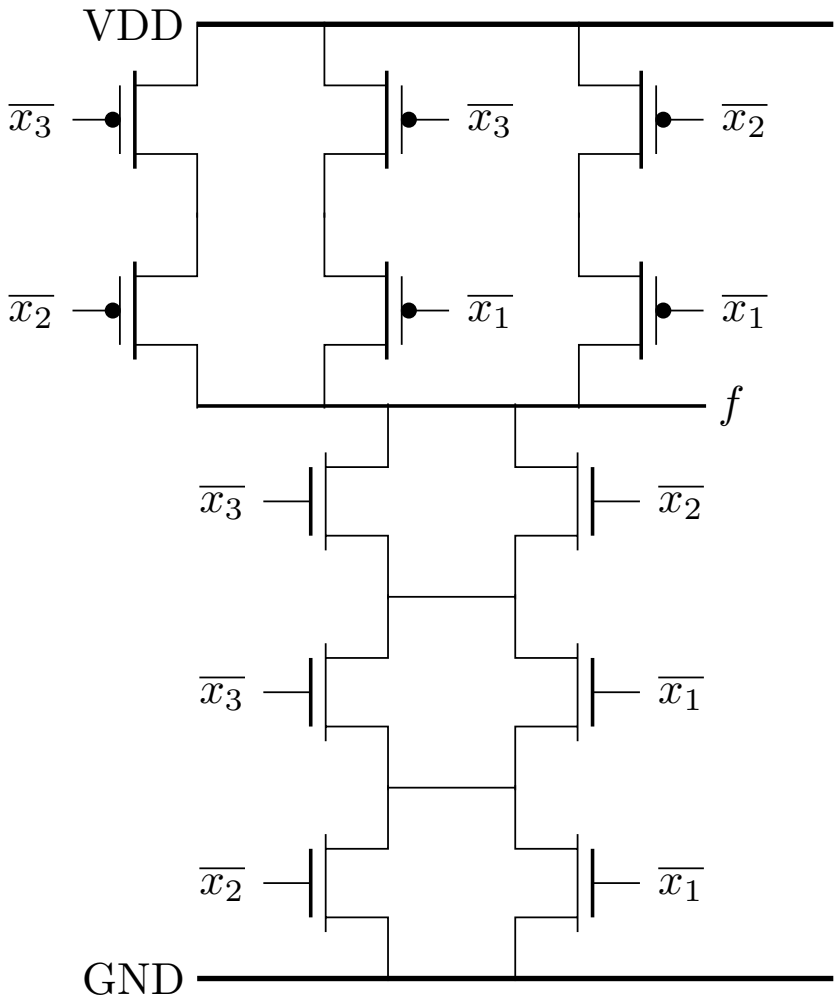

Abbildung 3: CMOS-Komplexgatter für f