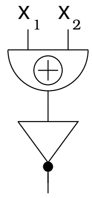
Lösungen zum 6. Übungsblatt Informatik II (SS 96)

Jürgen Zimmer (jzimmer@cscip.uni-sb.de)
Michael Mehl (mehl@dfki.uni-sb.de)

Aufgabe 6.1 [Addition zweier 2er-Komplement Zahlen]



1. Ein Schaltkreis, der die Funktion $\equiv$ berechnet, läßt sich leicht aus einem $\sim$ - und einem $\oplus$ -Gatter aufbauen.

2. Die Aufgabenstellung war leider nicht ganz korrekt. Man sollte zeigen, daß für $a, b, s \in \{0,1\}^n, c \in \{0,1\}$ und $\langle c, s\rangle = \langle a\rangle + \langle b\rangle$ folgendes gilt:

$$
\begin{array}{rcllllll}
[a]+[b] \in \{-2^{n-1}, \ldots, 2^{n-1}-1\} & \Leftrightarrow & [a] < 0 & \textit{und} & [b] < 0 & \textit{und} & [s] < 0 \\
& \textit{oder} & [a] > 0 & \textit{und} & [b] > 0 & \textit{und} & [s] \geq 0 \\
& \textit{oder} & [a] \leq 0 & \textit{und} & [b] \geq 0 & & \\
& \textit{oder} & [a] \geq 0 & \textit{und} & [b] \leq 0, & &
\end{array}
$$

was äquivalent ist zu der Aussage:

$$
\begin{array}{rcllllll}
[a]+[b] \notin \{-2^{n-1}, \ldots, 2^{n-1}-1\} & \Leftrightarrow & [a] < 0 & \textit{und} & [b] < 0 & \textit{und} & [s] \geq 0 \\
& \textit{oder} & [a] > 0 & \textit{und} & [b] > 0 & \textit{und} & [s] < 0
\end{array}
$$

Zum Beweis seien $a = (a_{n-1}, \ldots, a_0), b = (b_{n-1}, \ldots, b_0) \in \{0,1\}^n$,
c_i der Übertrag von Stelle i, $R_n := \{-2^{n-1}, \ldots, 2^{n-1}-1\}$, dann gilt:

$\Rightarrow$: Sei $[a]+[b] \notin R_n$

1.Fall $[a], [b] < 0$
$\Rightarrow a_{n-1} = b_{n-1} = 1$
und da $[a]+[b] \notin R_n$, gilt $[a]+[b] < -2^{n-1}$, also
$\langle a_{n-2}, \ldots, a_0\rangle + \langle b_{n-2}, \ldots, b_0\rangle \leq 2^{n-1}$ und damit
$s_{n-1} = c_{n-2} = 0$, also $[s] \geq 0$.

2.Fall $[a], [b] > 0$
$\Rightarrow a_{n-1} = b_{n-1} = 0$
und da $[a]+[b] \notin R_n$, gilt $[a]+[b] \geq 2^{n-1}$, also
$2^{n-1} \leq \langle a_{n-2}, \ldots, a_0\rangle + \langle b_{n-2}, \ldots, b_0\rangle < 2^n$ und damit
$s_{n-1} = c_{n-2} = 1$, also $[s] < 0$.

$\Leftarrow$:

1.Fall $[a] < 0$ und $[b] < 0$ und $[s] \geq 0$

$\Rightarrow s_{n-1} = c_{n-2} = 0$

$\Rightarrow \langle a_{n-2}, \ldots, a_0 \rangle + \langle b_{n-2}, \ldots, b_0 \rangle < 2^{n-1}$

$\Rightarrow [a] + [b] < -2^n + 2^{n-1} = -2^{n-1}$

$\Rightarrow [a] + [b] \notin R_n$

2.Fall $[a] > 0$ und $[b] > 0$ und $[s] < 0$

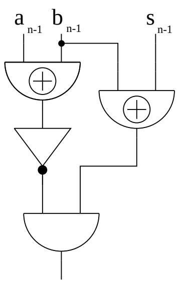
$\Rightarrow s_{n-1} = c_{n-2} = 1$

$\Rightarrow \langle a_{n-2}, \ldots, a_0 \rangle + \langle b_{n-2}, \ldots, b_0 \rangle \geq 2^{n-1}$

$\Rightarrow [a] + [b] \geq 2^{n-1}$

$\Rightarrow [a] + [b] \notin R_n$

3. Gemäß der in 6.1.2 bewiesenen Aussage, berechnet ein Schaltkreis das Überlaufsignal durch Vergleich der Vorzeichenbits von a, b und s :



Aufgabe 6.2 [$\circ$]

Gegeben ist die Funktion $\circ : \{0,1\}^{2\times 2} \longrightarrow \{0,1\}^2$

$$(x_1, x_2) \circ (y_1, y_2) := (x_1 \vee (x_2 \wedge y_1), x_2 \wedge y_2)$$

Die Funktion $\circ$ ist assoziativ, denn für beliebige $x_1, x_2, y_1, y_2, z_1, z_2 \in \{0,1\}$ gilt:

$$\begin{aligned}
[(x_1, x_2) \circ (y_1, y_2)] \circ (z_1, z_2) &= (x_1 \vee (x_2 \wedge y_1), x_2 \wedge y_2) \circ (z_1, z_2) \\
&= ((x_1 \vee (x_2 \wedge y_1)) \vee ((x_2 \wedge y_2) \wedge z_1), (x_2 \wedge y_2) \wedge z_2) \\
&= (x_1 \vee (x_2 \wedge y_1) \vee x_2 \wedge (y_2 \wedge z_1), x_2 \wedge (y_2 \wedge z_2)) \\
&= ((x_1 \vee (x_2 \wedge (y_1 \vee (y_2 \wedge z_1)))), x_2 \wedge (y_2 \wedge z_2)) \\
&= (x_1, x_2) \circ (y_1 \vee (y_2 \wedge z_1), y_2 \wedge z_2) \\
&= (x_1, x_2) \circ [(y_1, y_2) \circ (z_1, z_2)]
\end{aligned}$$

Anmerkung: Die Funktion $\circ$ ist nicht kommutativ, denn

$$(1,0) \circ (0,0) = (1,0) \neq (0,0) = (0,0) \circ (1,0)$$

Aufgabe 6.3 [Carry-Lookahead-Adder]

Leider waren auch hier die Behauptungen auf dem Übungsblatt nicht korrekt, die zu beweisenden Aussagen sollten so lauten ($0 \leq i \leq j < n$):

i) $p_{ii} = 1 \Rightarrow c_j = c_{i-1}$

ii) $g_{0j} = 1 \Leftrightarrow c_j = 1$

Bei Aussage i) gilt die Umkehrung nicht! (Gegenbeispiel: 0+0)
Bei Aussage ii) soll der Übertrag natürlich genau dann eins sein, wenn beim Addieren der ersten $j+1$ Stellen ein Übertrag generiert wurde und nicht erst bei den Stellen $i \dots, j$. (Gegenbeispiel: $i = j, a_j = 1, b_j = 0, c_{j-1} = 1 \Rightarrow c_j = 1$, aber $g_{jj} = a_j \wedge b_j = 0$) Bei letzterer Aussage hätte auch das Hinzufügen der Bedingung $i < j$ gereicht.
Die Beweise:

ad i) Seien $0 \leq i \leq j < n$ und $p_{ij} = 1$, dann gilt gemäß der Definition von p_{ij}:

$$\begin{aligned} p_{ij} &= p_{jj} \wedge p_{ij-1} \\ &= p_{jj} \wedge p_{j-1j-1} \wedge p_{ij-2} \\ &= \dots \\ &= \bigwedge_{i \leq k \leq j} p_{kk} = 1 \end{aligned}$$

$$\begin{aligned} &\Rightarrow \quad p_{kk} = 1 \qquad \forall k : i \leq k \leq j \\ &\Rightarrow \quad g_{kk} = 0 \qquad \forall k : i \leq k \leq j \text{ aufgrund der Definitionen von } p_{kk} \text{ bzw. } g_{kk} \end{aligned}$$

Behauptung: $c_k = c_{i-1} \qquad \forall k : i \leq k \leq j$
Beweis durch Induktion über k: Für $k = i$ ist $c_i := \underbrace{g_{ii}}_{=0} \vee \underbrace{p_{ii}}_{=1} \wedge c_{i-1} = c_{i-1}$
Für $i < k \leq j$ gilt : $c_k := \underbrace{g_{kk}}_{=0} \vee \underbrace{p_{kk}}_{=1} \wedge c_{k-1} = c_{i-1}$ nach Induktionsannahme.

Also gilt auch $c_j = c_{i-1}$ *q.e.d*

ad ii) $\underline{\Rightarrow}$ (Induktion über j):

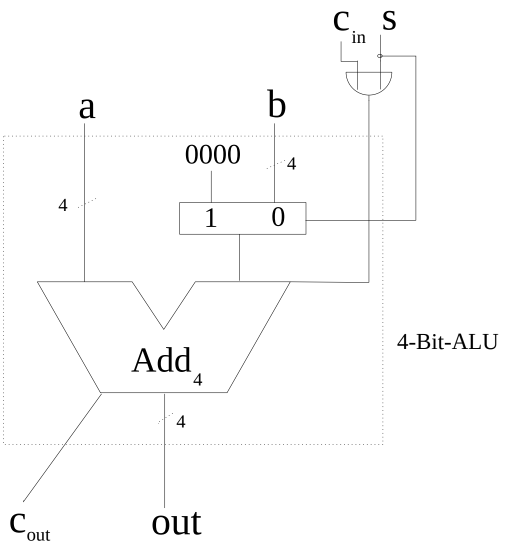
$j = 0$: $g_{00} = 1 \Rightarrow c_0 = 1$
$0 \leq j < n-1$: Induktionsannahme: $g_{0j} = 1 \Rightarrow c_j = 1$
Sei also $g_{0j+1} \stackrel{\text{Def}}{=} g_{j+1j+1} \vee (g_{0j} \wedge p_{j+1j+1}) = 1$
$\Leftrightarrow g_{j+1j+1} = 1$ oder $g_{0j} \wedge p_{j+1j+1} = 1$
Falls $g_{j+1j+1} = 1$, dann ist auch $c_{j+1} = 1$. Aus $g_{0j} = 1$ folgt nach I.A. $c_j = 1$ und damit mit $p_{j+1j+1} = 1$ auch $c_{j+1} = 1$.

$\underline{\Leftarrow}$ (Induktion über j):

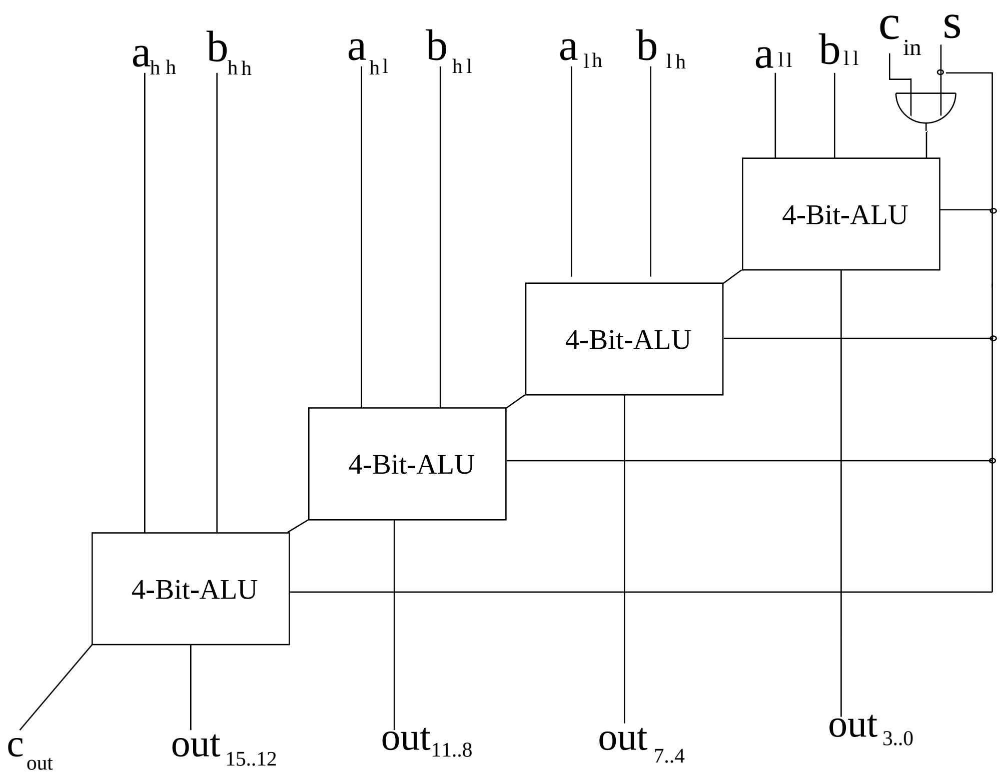
$j = 0$: $c_0 = 1 \Rightarrow (a_0 \wedge b_0) = 1$ oder $(a_0 \oplus b_0) \wedge c_{-1} = 1$
$\Rightarrow g_{00} = 1$.
$0 \leq j < n-1$ Induktionsannahme: $c_j = 1 \Rightarrow g_{0j} = 1$
Sei $c_{j+1} = 1$, d.h. $g_{j+1j+1} \vee (p_{j+1j+1} \wedge c_j = 1$
Es gilt: $g_{0j+1} \stackrel{\text{Def}}{=} g_{j+1j+1} \vee (g_{0j} \wedge p_{j+1j+1}) = 1$
Ist also $g_{j+1j+1} = 1$, dann auch g_{0j+1}
Falls $(p_{j+1j+1} \wedge c_j) = 1$, dann ist nach I.A. auch g_{0j} und damit g_{0j+1}.

Aufgabe 6.4 [Kaskadierbare ALU?]

- Ein 4-Bit-ALU die die angegebenen Funktionen berechnet, kann so aufgebaut werden...

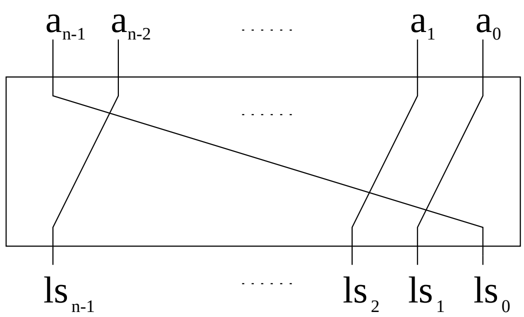


- ...damit funktioniert dann das Kaskadieren fast wie im Buch beschrieben. Beim Inkrement einer 16-Bit Zahl, darf nämlich nur zu den untersten 4 Bits 1 addiert werden.

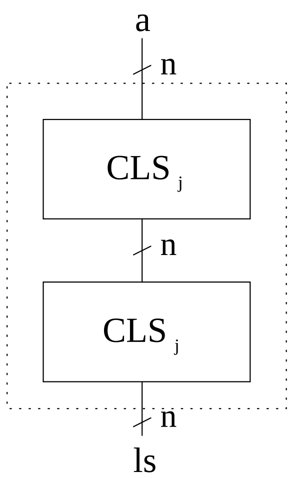


Aufgabe 6.5 [Shift]

1. Konstruiere zunächst für alle Zweierpotenzen 2^j mit $0 \leq j < i$ einen zyklischen Linksshifter CLS_{2^j}:

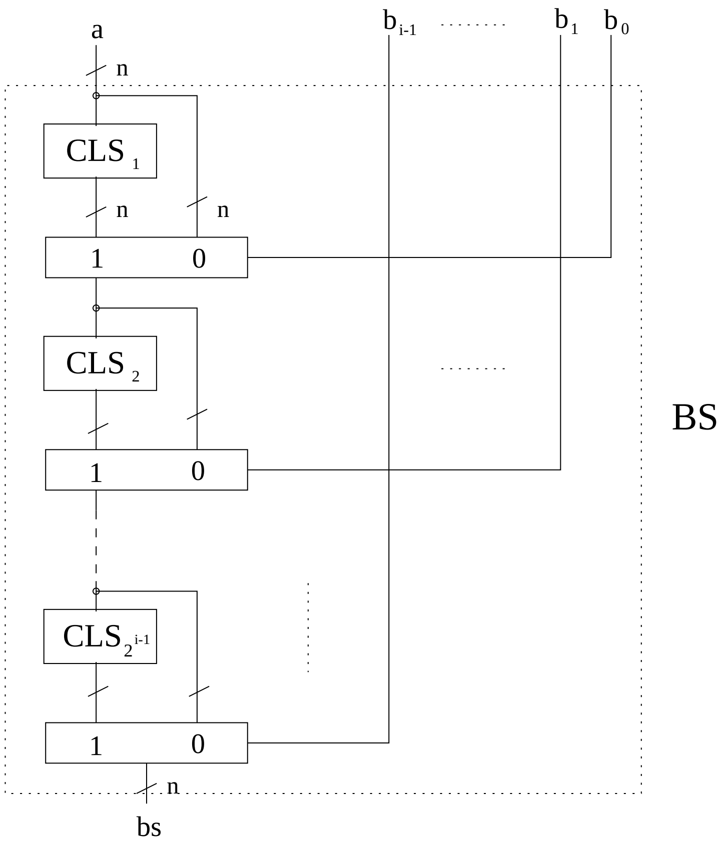


CLS 1



CLS 2 j

Mit diesen kann man sich gemäß der Darstellung $\langle b\rangle = \sum_{0\leq j<i} b_i 2^i$ den gewünschten Barrelshifter zusammensetzen:

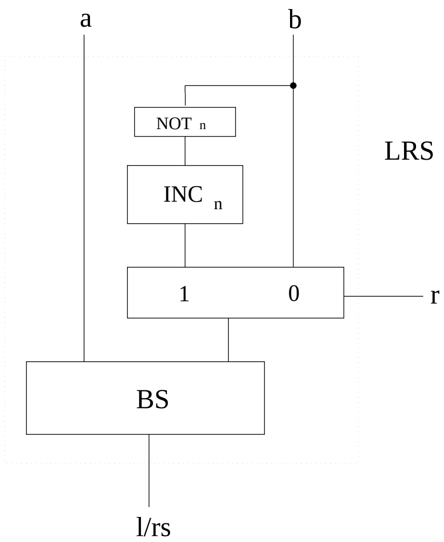


Die Kosten des Barrelshifters ergeben sich also aus den Kosten von i n-Bit Mutliplexern. Die zyklischen Linksshifter besitzen Kosten 0.
$C(BS) = i \cdot C(n - BitMUX) = i \cdot (3n + 1) = O(n \cdot i)$

2. Ein zyklischer Rechtsshift um k Bits entspricht einem zyklischen Shift um $n-k$ Bits nach links. Günstigerweise gilt:

$$\begin{aligned} n - \langle b \rangle &= 2^i - \sum_{j=0}^{i-1} b_j 2^j \\ &= \sum_{j=0}^{i-1} 2^j + 1 - \sum_{j=0}^{i-1} b_j 2^j \\ &= 1 + \sum_{j=0}^{i-1} (1 - b_j) 2^j \\ &= 1 + \langle \overline{b} \rangle \end{aligned}$$

Mit dem Eingang r wird demnach entschieden, ob a um $\langle b \rangle$ oder um $1 + \langle \overline{b} \rangle$ nach links geshiftet wird:



An zusätzliche Gatter werden nur i NOT-Gatter ein i-Bit MUX und ein i-Bit Incrementierer benötigt, also bei Verwendung eines Carry-Chain-Adders:
$C(LRS) = C(BS) + i + (3i + 1) + O(i) = C(BS) + O(i)$.

3. Ein nichtzyklischer Rechtsshifter, der von links Nullen nachführt, sieht natürlich so aus: