Beispiellösungen in Kurzform zur Übung 3

Aufgabe 1

Teil a

O_1	O_2	L
0	0	0
0	1	0
1	0	0
1	1	1

Es ergibt sich die AND-Funktion. Deshalb wird diese Art Schaltung auch als 'Wired-AND' bezeichnet.

Teil b: Zeitverhalten

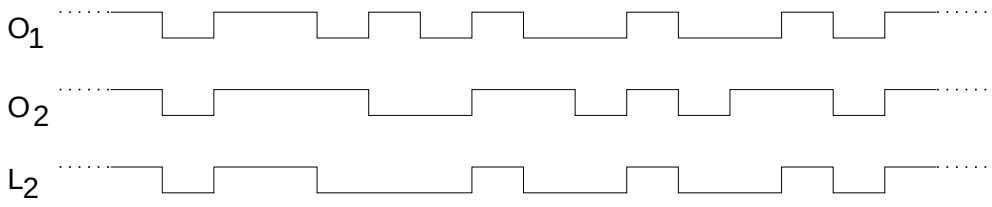

Abbildung 1: Verlauf der Signale O_1, O_2, L_2 über die Zeit t.

Das Signal für L_2 (siehe Abblidung 1) ergibt sich durch schrittweise Anwendung der Wahrheitstabelle von Teil a. Zu erkennen ist, daß keiner der Sender sein Signal auf der Leitung durchsetzen kann. Da jeweils die Null auf der Leitung L erscheint, wird die Null auch als 'dominanter Pegel' bezeichnet.

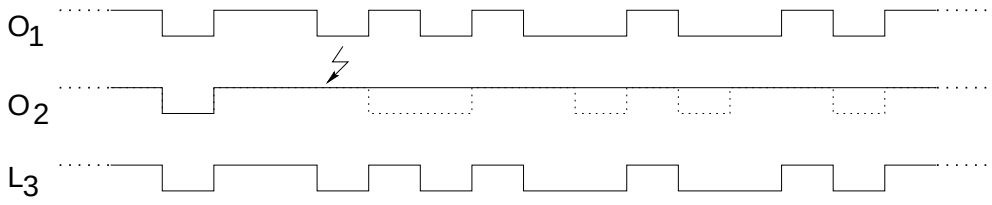

Abbildung 2: Verlauf der Signale O_1, O_2, L_3 über die Zeit t.

Teil c: System

Für Teil c ergibt sich der Signalverlauf von L_3 (Abblidung 2). Durch die 'Rückzugseigenschaft' wird das Signal O_2 auf '1' geschaltet. O_1 wird nun nicht mehr überschrieben und kann seine Signale unbeeinflußt senden. IC1 kann nicht mal

die Existenz eines weiteren Teilnehmers feststellen. Es gilt: Wer zuerst eine '1' sendet, verliert den Zugriff auf die Leitung L.

Teil d: Protokoll

Damit ein Sender sich zurückzieht und die Leitung L freigibt, muß innerhalb der ersten 4 Bit ($A3$-$A0$) eine Kollision festgestellt werden. Durch die 4 Bit ergeben sich 16 verschiedene Arbitrierungscodes. Diese Codes müssen so auf die Sender verteilt werden, daß kein Code zweimal vergeben wurde.

Weiterhin müssen beide Sender synchron mit der Übertragung beginnen, damit sich die Arbitrierungsphasen exakt überlappen. (Deshalb führende Null als 'start of frame'-SOF.) Es werden die Daten bevorzugt, die einen niedrigen Arbitrierungscode haben $\rightarrow$ Code=0 hat höchste Priorität.

Bei einem dritten Teilnehmer muß wieder gelten, daß jeder Code nur einmal an einen Sender vergeben wurde.

Das Prinzip, des hier vorgestellten Protokolls, wird im CAN-System (Controller Area Network) verwendet, welches hauptsächlich im KFZ-Bereich eingesetzt wird. Das Protokoll ist allerdings etwas komplexer und besteht aus 11 bzw. 29 Arbitrierungsbits, 0-8 Datenbytes und weiteren Bitfeldern zur Datensicherung.

Aufgabe 2

Teil a:

Die Kodierung der Steuerleitungen c_0, c_1 erfolgt mit Hilfe von Folie 13 aus dem Kombinatorikteil der Vorlesung[1] nach der folgenden Tabelle:

	AND	NAND	OR	NOR	Bool'sche Formel
Fkt-Kode	0	1	2	3	
c_0	0	1	0	1	
c_1	0	0	1	1	
x_0	0	1	0	1	$x_0 = c_0$
x_1	0	1	1	0	$x_1 = c_0\overline{c_1} + \overline{c_0}\,c_1$
x_2	0	1	1	0	$x_2 = x_1$
x_3	1	0	1	0	$x_3 = \overline{c_0}$

Bemerkung: c_1 selektiert die Grundoperation AND bzw. OR und c_0 gibt an ob das Ergebnis noch zu invertieren ist.

Damit erhält man die Schaltung:

[1] Überschrift: Kodierte Multiplexer: Beispiel n = 2

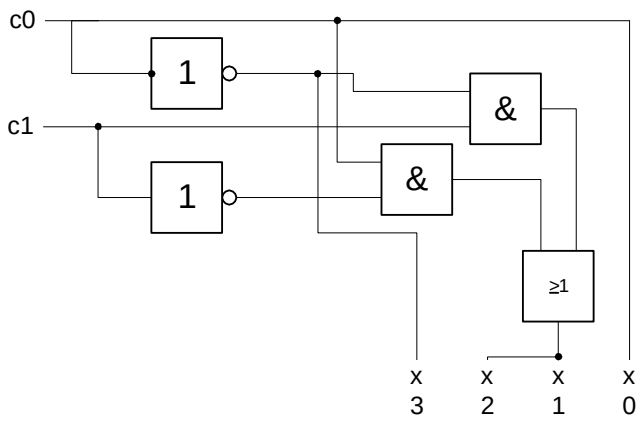

Abbildung 3: Schaltung für die Auswahl der Funktionen.

Teil b: 7 Segmentanzeige

Die Dekodierung wird standardgemäß nach dem Dualzahlsystem aufgebaut, d.h.

$$\begin{aligned} x_0 &= \overline{b_0}\,\overline{b_1}\,\overline{b_2}\,\overline{b_3} \\ x_1 &= b_0\overline{b_1}\,\overline{b_2}\,\overline{b_3} \end{aligned}$$

$s_1, \ldots, s_7$ bestimmt man, indem man sich für jedes Segment anschaut bei welchen Ziffern es einzuschalten ist, d.h.

$$\begin{aligned} s_1 &= x_0 + x_1 + x_2 + x_3 + x_4 + x_7 + x_8 + x_9 \\ s_2 &= x_0 + x_1 + x_3 + x_4 + x_5 + x_6 + x_7 + x_8 + x_9 \\ s_3 &= x_0 + x_2 + x_3 + x_5 + x_6 + x_8 + x_9 \\ s_4 &= x_0 + x_2 + x_6 + x_8 \\ s_5 &= x_0 + x_4 + x_5 + x_6 + x_8 + x_9 \\ s_6 &= x_0 + x_2 + x_3 + x_5 + x_6 + x_7 + x_8 + x_9 \\ s_7 &= x_2 + x_3 + x_4 + x_5 + x_6 + x_8 + x_9 \end{aligned}$$

Man beachte, daß die Dekodierung nur AND-Gatter verwendet, und der Konverter dagegen ausschließlich OR-Gatter. Daher einfache Realisierung mittels PLA:

Werden Werte außerhalb der Vorgaben (> 9) angelegt, so erscheint bei dieser Lösung eine 8 falls der Wert gerade ist, und eine 9 sonst.

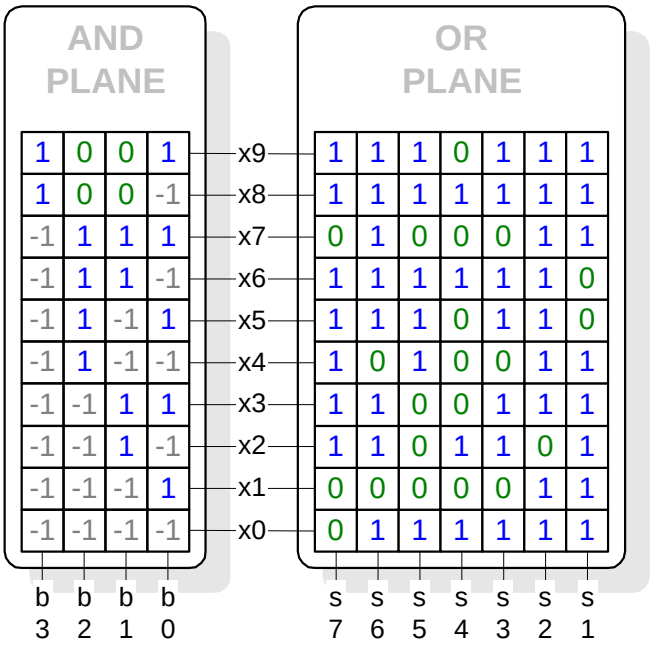

Abbildung 4: 7-Segment-Anzeigen Schaltung als PLA.