Maris Koch 20.12.2021

Einführung in die Informatik | Übung 02

Speicher und Rechnen:

a)

Wenn man „Gigabyte“ nach dem Dezimalpräfix interpretiert, so kommt man auf folgende Werte:

1 Byte	8 Bit
1Kilobyte	8.000 Bit
1 Megabyte	8.000.000 Bit
1 Gigabyte	8.000.000.000 Bit
8 Gigabyte	64.000.000.000 Bit

Wenn man „Gigabyte“ nach dem Binärpräfix gemäß IEC interpretiert, so kommt man auf folgende Werte:

1 Byte	8 Bit
1 Kibibyte	8.192 Bit
1 Mebibyte	8.388.608 Bit
1 Gibibyte	8.589.934.592 Bit
8 Gibibyte	68.719.476.736 Bit

Jede Binary Cell kann 1 Bit speichern, durch das in ihr enthaltene RS-Flipflop. Man benötigt also für 8 GB RAM 64.000.000.000 Binary Cells mit jeweils einem in ihr enthaltenes RS-Flipflop.

Nach der Norm von vor 1999 enthält ein GB 68.719.476.736 Bit, da der Umrechnungsfaktor nicht 1000, sondern 1024 war. Dem entsprechend würde man 68.719.476.736 Binary Cells mit jeweils einem in ihr enthaltenes RS-Flipflop benötigen.

b)

Der Demultiplexer ermöglicht, jede Zeile des Tabellenspeichers anzusprechen, ohne dass für jede eine eigene Adressleitung von Prozessor zum Speicher führen muss.

c)

Im Folgenden geschrieben ist meine Auffassung der Aufgabe:

Wie viele Steuerleitungen werden von Prozessor zu Demultiplexer mindestens benötigt, um alle Zeilen des Tabellenspeichers anzusprechen?

Bei *n* Steuerleitungen kann man *z* Zeilen ansteuern. Aufgrund von der Wortbreite von *64 Bit*, enthält jede Zeile *64 Binary Cells*. Daraus ergibt sich folgende Anzahl an Zeilen:

$$z_{Dezimal} = \frac{64.000.000.000}{64} = 1.000.000.000$$

$$z_{IEC} = \frac{68.719.476.736}{64} = 1.073.741.824$$

Allgemein lässt sich die Anzahl an ansteuerbaren Zeilen berechnen durch:

$$z = 2^n$$

Daraus ergibt sich:

$$n = \log_2(z)$$

Eingesetzt ergibt dies folgende Werte:

$$n_{Dezimal} = \log_2(1.000.000.000) \approx 29{,}90$$

$$n_{IEC} = \log_2(1.073.741.824) = 30$$

Für $n_{Dezimal}$ muss aufgerundet werden, dass es keine Teilsteuerleitungen gibt. Daraus ergibt sich für beide Fälle, dass der Adressbus mindestens *30 Leitungen* zum Speicher benötigt.

Zum Rechenwerk:

a)

Das Addierwerk beginnt mit einem Halbaddierer, darauf folgen 63 Volladdierer. Jeder Volladdierer beinhaltet 2 Halbaddierer. Daraus folgt, dass man $63 * 2 + 1 = 127$ Halbaddierer für das Addierwerk benötigt.

b)

Die Subtraktion von 2 Binärzahlen b_1 und b_2 erfolgt durch die Addition von b_1 mit dem 2er Komplement von b_2. Diese Komplementbildung erfolgt in den ALUs.

Belegung der Steuerleitungen:

u	v	m
0	1	1

Darüber hinaus muss in die erste ALU ein $c_{alt} = 1$ zugeschaltet werden, da ansonsten das 1er Komplement gebildet wird anstelle des 2er Komplements.

Die Komplementbildung wird im Folgenden erklärt:

Die einzelnen Stellen von b_2 werden in z_2 eingeführt und durch $u = 0$ und $v = 1$ invertiert. Dies geschieht durch zwei Schritte:

1. Durch $u = 0$ gibt das eine AND-Gate der Verarbeitung von z_2 permanent eine 0 aus.
2. Durch $v = 1$ und die Negierung von z_2 gibt das andere AND-Gate immer $\neg z_2$ aus.

Diese beiden Outputs sind durch ein OR-Gate verknüpft, woraus sich die Invertierung von z_2 für z_2' logisch ableiten lässt:

$$z_2' = \neg z_2$$

Durch das Einführen von $c_{alt} = 1$ wird 1 addiert, was in Kombination mit der Invertierung von b_2 zum 2er Komplement von b_2 führt.

Des Weiteren muss $m = 1$ geschaltet sein, um die Überträge durchzuschalten zu der folgenden ALU. Dies ist notwendig für die korrekte Addition.

Es wird also b_1 mit dem 2er Komplement von b_2 addiert, was äquivalent zu $b_1 - b_2$ ist, wodurch mit ALUs eine Subtraktion durchgeführt werden kann.

Maris Koch 20.12.2021

Optimieren einer Schaltung:

a)

a	b	c	d	$A =$ $a \wedge c$	$B =$ $\neg a \wedge b \wedge c$	$C =$ $\neg a \wedge c \wedge \neg d$	$D =$ $\neg(c \vee d)$	$y =$ $A \vee B \vee C \vee D$
0	0	0	0	0	0	0	1	1
0	0	0	1	0	0	0	0	0
0	0	1	0	0	0	1	0	1
0	0	1	1	0	0	0	0	0
0	1	0	0	0	0	0	1	1
0	1	0	1	0	0	0	0	0
0	1	1	0	0	1	1	0	1
0	1	1	1	0	1	0	0	1
1	0	0	0	0	0	0	1	1
1	0	0	1	0	0	0	0	0
1	0	1	0	1	0	0	0	1
1	0	1	1	1	0	0	0	1
1	1	0	0	0	0	0	1	1
1	1	0	1	0	0	0	0	0
1	1	1	0	1	0	0	0	1
1	1	1	1	1	0	0	0	1

b)

Verbale Formulierung:

$$y = (a\ AND\ c)\ OR\ (NOT(a)\ AND\ b\ AND\ c)\ OR\ \big(NOT(a)\ AND\ c\ AND\ NOT(d)\big)\ OR\ \big(NOT(c\ OR\ d)\big)$$

Logische Formulierung:

$$y = (a \land c) \lor (\neg a \land b \land c) \lor (\neg a \land c \land \neg d) \lor (\neg(c \lor d))$$

c)