Benotete Hausaufgabe

Bei diesem Dokument handelt es sich um die Lösungen von Adrian Khairi aus dem 21C-Kurs zur benoteten Hausaufgabe Nummer 1 „LogischeGatter_A.pdf“.

Aufgabe: Speichern und Rechnen

a)

Wie viele Binary Cells und wie viele RS-Flip-Flops werden benötigt, um diesen Speicher zu realisieren?

Hierbei gibt es 2 verschiedene Lösungsansätze, die im Folgenden beide berechnet werden.

1. Ansatz:

   1 Byte = 8 Bit
   1 Kilobyte = 1024 Byte
   1 Megabyte = 1024 Kilobyte
   1 Gigabyte = 1024 Megabyte

   Das heißt 1 Gigabyte in Bit wären:
   1*8*1024*1024*1024 = 8.589.934.592 Bit

   Nun rechnet man das Ergebnis noch für 8 Gigabyte, statt wie zuvor für 1 Gigabyte, aus:
   8589934592*8 = 68.719.476.736 Bit

2. Ansatz:

   1 Byte = 8 Bit
   1 Kilobyte = 1000 Byte
   1 Megabyte = 1000 Kilobyte
   1 Gigabyte = 1000 Megabyte

Das heißt 1 Gigabyte in Bit wären:
1*8*1000*1000*1000 = 8.000.000.000

Nun rechnet man das Ergebnis noch für 8 Gigabyte, statt wie zuvor für 1 Gigabyte, aus:
8.000.000.000*8 = 64.000.000.000

Antwort:
Jede Binary Cell kann 1 Bit speichern, weil diese einen RS-Flip-Flop in sich enthält. Somit benötigt man nach dem ersten Ansatz 8.589.934.592 und nach dem zweiten Ansatz 64.000.000.000 Binary Cells mit jeweils einem in ihr enthaltenen RS-Flip-Flop.

b)

In heute üblichen Systemen ist der RAM als Tabellenspeicher mit 64Bit Wortbreite realisiert. Welches Bauteil ermöglicht es jede Zeile des Tabellenspeichers anzusprechen, ohne dass für jede eine eigene Adressleitung von Prozessor zum Speicher führt?

Hierbei handelt es sich um einen Demultiplexer. Dieses Bauteil ermöglicht es jede Zeile des Tabellenspeichers anzusprechen, ohne dass für jede eine eigene Adressleitung von Prozessor zum Speicher führt. Durch den Demultiplexer kann man mit n-Schaltleitungen 2^n –Ausgabeleitungen ansprechen.

c)

Wie viele Leitungen muss der Adressbus zu diesem Speicher mindestens haben, um jede Zeile ansprechen zu können?

Aus den bei Aufgabenteil a und b gewonnen Ergebnissen lassen sich die folgenden zwei Antworten auf diese Frage herleiten. Der Unterschied beruht hierbei auf den bei Aufgabenteil a beschriebenen zwei Ansätzen. Da man aus dem Aufgabentext von b entnehmen kann, dass die Wortbreite 64 Bit entspricht, muss man diese bei der

Berechnung der Mindestanzahl an Leitungen, um jede Zeile ansprechen zu können, integrieren.

1. Ansatz nach Aufgabenteil a:

$$\log_2\left(\frac{68.719.476.736}{64}\right) = 30$$

2. Ansatz nach Aufgabenteil a:

$$\log_2\left(\frac{64.000.000.000}{64}\right) = 29{,}8973 \approx 30$$

Der Adressbus zu diesem Speicher braucht mindestens 30 Leitungen, um jede Zeile ansprechen zu können.

Zum Rechenwerk:

a)

Ein Addierwerk soll zwei 64Bit-Zahlen addieren. Wie viele Halbaddierer müssen für diese Rechenoperation in dem Addierwerk vorhanden sein?

In einem Addierwerk findet man bei zwei 64Bit-Zahlen einen Halbaddierer vor, auf welchen 63 Volladdierer folgen. Jeder Volladdierer enthält zwei Halbaddierer. Daraus lässt sich die Folgende Rechnung ableiten:

$$2 * 63 + 1 = 127\ Halbaddierer$$

127 Halbaddierer müssen für diese Rechenoperation in dem Addierwerk vorhanden sein.

b)

Wie kann mit einer ALU eine Binärzahl von einer anderen subtrahiert werden? Geben Sie bitte die Belegung der Steuerleitung an und erklären Sie in Stichpunkten warum dadurch die ALU eine Subtraktion durchführt.

Die Subtraktion funktioniert, indem man das Zweierkomplement des Subtrahenden in der ALU bildet. Dadurch kann die ALU im weiteren Verlauf eine Addition des Minuenden mit dem Zweierkomplement des Subtrahenden durchführen.

Die Belegung der Steuerleitung sieht wie folgt aus:

m	u	v	c_{alt}
1	0	1	1

Die Erklärung, warum die ALU durch die zuvor beschriebene Belegung der Steuerleitung eine Subtraktion durchführen kann, folgt nun:

- u steht auf 0, daher wird das AND-Gatter mit z_2 und u den Wert 0 ausgeben
- v steht auf 1, daher wird das NAND-Gatter mit ¬ z_2 und v den Wert von ¬ z_2 ausgeben
- Danach werden die Werte des zuvor beschriebenen AND-Gatters mit dem Wert des zuvor beschriebenen NAND-Gatters in einem OR-Gatter zusammengeführt. Das Ergebnis hieraus wird z_2' genannt. Weil das AND-Gatter den Wert 0 ausgibt, ist der Wert des OR-Gatters nur vom Wert des NAND-Gatters abhängig. Somit entspricht z_2' dem Wert von ¬ z_2 . Dieser Schritt ist die Invertierung von z_2. Damit hat man das Einerkomplement von z_2 gebildet
- c_{alt} steht auf 1 und entspricht somit der Addition des Einerkomplements von z_2' mit 1. Daraus resultiert das Zweierkomplement von z_2

- Die Werte von z_1, z_2' und c_{alt} gehen in den Volladdierer. z'_2 und c_{alt} bilden das Zweierkomplement von z_2. Dadurch kann eine Subtraktion durch die Addition des Minuenden beziehungsweise z_1 mit dem Zweierkomplement des Subtrahenden beziehungsweise des Zweierkomplements von z_2 durchgeführt werden
- m steht auf 1, daher ist das UND-Gatter vor c'_{neu} nur vom Wert von c_{neu} abhängig. m lässt den Übertrag also durchschalten
- Somit ergibt der Volladdierer einen Wert s, der für „Summe“ steht, und einen Wert c'_{neu}, der für den Übertrag aus diesem Volladdierer steht

Aufgabe: Optimierung einer Schaltung

a)

Erstellen Sie eine Wertetabelle, welche der Schaltung in Abbildung 1 entspricht. Mit den Knöpfen a bis d als Eingänge, leuchtet die LED als Ausgang y, wenn dieser 1 ist.

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

b)

Leiten Sie die Wahrheitsfunktion ab, welche genau der Schaltung in Abbildung 1 entspricht, ohne sie zu vereinfachen.

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

beziehungsweise

$$y = (a \wedge\ c) \vee (\neg a\ \wedge b \wedge c) \vee (\neg a \wedge c \wedge \neg d) \vee \neg(c \wedge d)$$

c)

Mit Hilfe des Karnaugh-Veitch-Diagramms kann die in Gleichung 1 dargestellte Funktion, welche einen vereinfachten Aufbau hat, als Optimierung abgeleitet werden.

$$y_{opt} = \neg d \vee (c \wedge d) \vee (a \wedge c)$$

Zeichnen Sie die vereinfachte Form der Schaltung.