RS1 HA zum 09.12.11

Claas Jaehrling, Sven-Hendrik Haase

December 9, 2011

Aufgabe 6.1

(a)

Die Minimaldistanz dieses Codes beträgt 4: Ändert sich irgendwo ein Bit, so auch das entsprechende Zeilen- und Spaltenparitätsbit und das Paritätsbit der Spaltenparität. Insgesamt 4 Bit.

(b)

Mit diesem 2D-Paritätscode können alle Einbitfehler eindeutig erkannt und korrigiert werden, weil er quasi "von 3 Seiten" überprüft wird: Von der Spalte, der Zeile und der Überprüfung der Parität der Spaltenparität her. Einzelne Bitfehler bzw. pro Zeile oder Spalte einmal vorkommende Bitfehler führen zu "falschen" Paritätsbits in der Spalte und Zeile, in der sie auftreten. Doppelte Bitfehler nebeneinander werden entweder in der Spalte oder in der Zeile als falsch erkannt; immerhin werden nur höchstens 2 Blöcke als fehlerhaft bezeichnet. Bei dreifachen Bitfehlern gilt: in einer Reihe werden sie leicht erkannt, da die Parität dieser Reihe den falschen Wert liefert, ansonsten gilt das gleiche wie für 1 oder 2 fehlerhafte Bits.

(c)

Wenn vier Bits fehlerhaft übertragen werden und diese in der Tabelle ein Quadrat bilden, also je 2 Zeilen und Spalten betroffen sind, dann können sie nicht als Fehler erkannt werden.

(d)

Es handelt sich um Ziehen ohne Zurücklegen und ohne Beachtung der Reihenfolge, also berechnet sich die Anzahl der Möglichkeiten als n über k, wobei n hier 49 und k 4 entspricht. Es gibt also insgesamt 211876 Möglichkeiten. Die Relation von nicht erkennbaren Vierbitfehlern zur möglichen Gesamtanzahl ist 9 zu 52.969.

Aufgabe 6.2

(e)

Wenn die Anzahl der Einsen, die ein Parity-Bit checkt, gerade ist, wird dieses auf 0 gesetzt, bei einer ungeraden Anzahl auf 1. Also müssen wir momentan gucken, ob das beim korrekten Code funktioniert: 0001111 stimmt also. Verändern wir c_6 ergibt sich 0001101. Parity-Bits p_2 und p_3 sind für c_6 verantwortlich und müssten nun theoretisch beide geflippt sein, damit der Code nicht als fehlerhaft erkannt wird. Da sie das aber nicht sind, muss der Code fehlerhaft sein. Wir können auch die Position des fehlerhaften Bits errechnen, indem wir die Position der kaputten Parity-Bits aufaddieren: $2 + 4 = 6$. Also ist das 6. Bit kaputt.

Prüfworte:

$$x_a = (0 + 0 + 1 + 1) mod 2 = 0$$

$$x_b = (0 + 0 + 0 + 1) mod 2 = 1$$

$$x_c = (1 + 1 + 1 + 1) mod 2 = 0$$

(f)

Durch das Addieren der jeweiligen Positionen der falschen Parity-Bits ergibt sich die Position des kaputten Bits. Sind also Position 2 und 8 falsche Parity-Bits, ist das Bit an Position 10 kaputt.

Aufgabe 6.3

(a)

$$f(x) = (\overline{x_3} \vee x_2) \wedge (\overline{x_2} \vee x_3) \tag{1}$$
$$Vereinfachung(3-bit): \tag{2}$$

Menge	x3	x2	x1
NOT x3	0	1	1
2	0	1	0
NOT x3 OR 2	0	1	1
NOT 2	1	0	1
3	1	0	0
NOT 2 OR 3	1	0	1
(NOT 3 OR 2) AND (NOT 2 OR 3)	0	0	1

$$\tag{3}$$
$$KDNF: \tag{4}$$
$$!(x_2 \vee x_3) \tag{5}$$
$$KKNF: \tag{6}$$
$$x_2 \wedge x_3 \tag{7}$$
$$RMF: \tag{8}$$

(b)

$$g(x) = \overline{x_3} \oplus \overline{x_2} \tag{9}$$
$$KDNF: \tag{10}$$
$$KKNF: \tag{11}$$
$$RMF: \tag{12}$$