Dr. Jürg M. Stettbacher
Neugutstrasse 54
CH-8600 Dübendorf

Telefon: +41 43 299 57 23
Email: dsp@stettbacher.ch

Übung

CRC

1. Codierung und Decodierung mit CRC

   In einem Kommunikationssystem können Codeworte mit 16 Bit übertragen werden. Von den 16 Bits sollen 12 Bit Nutzdaten sein, die restlichen Bits sind für die CRC-Prüfsumme reserviert.

   (a) Wählen Sie ein geeignetes CRC-Polynom[1].

   (b) Es soll das Nutzdatenwort $\underline{u} = (B1D_h)$ übertragen werden. Bestimmen Sie das zu übertragende Codewort $\underline{c}$.

   (c) Machen Sie die Auswertung auf der Empfängerseite für den Fall, dass das Codewort korrekt übertragen wurde.

2. Fehlerhafte Übertragung

   Mit einem geeigneten CRC-Polynom vom Grad p werden mit Sicherheit alle Burstfehler der Länge $b \leq p$ erkannt. Nehmen Sie an, im korrekten Codewort $\underline{c} = (B1D2_h)$ mit CRC-4 trete bei der Übertragung ein Burstfehler der Länge drei auf. Verfälscht werden die Bits[2] 4, 3 und 2.

   (a) Berechnen Sie den Rest auf der Empfängerseite, der die fehlerhafte Übertragung anzeigt.

   (b) Erzeugen Sie für das Generatorpolynom CRC-4 einen Fehlervektor $\underline{e}$ mit drei isolierten[3] 1-Bit Fehlern, so dass die Fehler vom Empfänger nicht erkannt werden können. Verifizieren sie Ihre Lösung.

3. Hardware

   Eine CRC-Prüfsumme soll in Hardware berechnet werden. Zur Auswahl stehen CRC-5-ITU und CRC-7.

   (a) Wählen Sie diejenige Variante, die weniger XOR-Gatter benötigt.

   (b) Zeichnen Sie das Schaltbild mit dem Schieberegister und den XOR-Gattern.

   (c) Welches ist der Rest, wenn die Hardware-Lösung das Bitmuster $\widetilde{\underline{c}} = (134_h)$ überprüft?

[1] Sie können die Tabelle mit den Generatorpolynomen aus dem Skript benutzen.

[2] Die Bits seien von rechts nach links nummeriert, beginnend bei 0.

[3] *Isoliert* heisst, dass die Fehlerbits im Fehlervektor nicht direkt nebeneinander stehen, sich also nicht berühren.

Antworten

1. Codierung und Decodierung mit CRC

   (a) Sollen vier Bits für die Prüfsumme verwendet werden, so muss das CRC-4 Polynom gewählt werden:

   $$G(z) = z^4 + z + 1$$

   In der Polynomdarstellung: $\underline{g} = (10011) = (13_h)$.

   (b) Das Codewort $\underline{c}$ ergibt sich aus dem nach links geschobenen Nutzdatenwort $\underline{u}$ und der angehängten Prüfsumme $\underline{p}$. Um die Prüfsumme zu berechnen, sind dem Nutzdatenwort $\underline{u}$ vier Nullen (Grad p des CRC-Polynoms) anzuhängen. Dies führt zur folgenden Polynomdivision:

```
1011000111010000 : 10011 = ...
10011
-----
  10100
  10011
  -----
    11111
    10011
    -----
     11001
     10011
     -----
      10100
      10011
      -----
        11110
        10011
        -----
         11010
         10011
         -----
          10010
          10011
          -----
            0010    <== Rest
```

   Die gesuchte Prüfsumme wird $\underline{p} = (0010) = (2_h)$ und das gesuchte Codewort resultiert zu $\underline{c} = (B1D2_h)$.

   (c) Bei einer korrekten Übertragung muss der Empfänger folgende Polynomdivision ausführen:

   $(B1D2_h) : (13_h)$

   Da der Rest null wird, nimmt der Empfänger an, die Übertragung sie (vermutlich[4]) fehlerfrei abgelaufen.

2. Fehlerhafte Übertragung

   (a) Das fehlerhafte Bitmuster heisst $\widetilde{\underline{c}} = (1011000111001110) = (B1CE_h)$. Der Test sieht so aus:

```
1011000111001110 : 10011 = ...
10011
-----
  10100
  10011
  -----
    11111
    10011
    -----
     11001
     10011
     -----
      10100
```

[4] Eine 100 % sichere Übertragung gibt es nie. Es sind immer Fehlerfälle denkbar, die nicht erkennbar sind. Das Ziel ist es, die Wahrscheinlichkeit für derartige Fälle zweckmässig klein zu machen.

```
10011
-----
  11101
  10011
  -----
   11101
   10011
   -----
    11101
    10011
    -----
     11100
     10011
     -----
      1111      <== Rest
```

Der Rest ist ungleich null und zeigt die Fehlübertagung an.

(b) Um das Muster mit isolierten, nicht erkennbaren 1-Bit Fehlern zu finden, bilden wir ein Vielfaches[5] des Generatorpolynoms $G(z)$. Nach etwas Probieren finden wir zum Beispiel diese Lösung:

```
    10011
   10011
10011
---------
100000101
```

Zur Verifikation wird das Fehlerpolynom zu einem gültigen Codewort addiert:

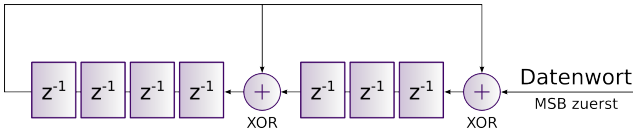
$\widetilde{\underline{c}} = (B1D2_h) + (105_h) = (B0D7_h)$

Achtung: Es handelt sich um die bitweise MOD2-Arithmetik. Die Polynomdivision $(B0D7_h) : (13_h)$ zeigt, dass der Rest null ergibt und der Empfänger die fehlerhafte Übertragung nicht erkennen kann.

3. Hardware

(a) Die Anzahl der XOR-Glieder entspricht der Anzahl gesetzter Einsen im Generatorvektor $\underline{g}$ minus 1. CRC-7 benötigt mit 3 Termen nur 2 XOR-Glieder, während CRC-5-ITU derer 3 benötigt. Wir wählen also CRC-7.

(b) Das Schaltbild sieht so aus:



(c) Im Schieberegister läuft folgendes ab:

```
   SR         Bitmuster     Kommentar
-------------------------------------------------------------
00000000 <-- 100110100     Startzustand, SR = 0.
00000001 <-- 00110100
00000010 <-- 0110100
00000100 <-- 110100
00001001 <-- 10100
00010011 <-- 0100
00100110 <-- 100
01001101 <-- 00
10011010 <-- 0             MSB = 1, XOR wird ausgeführt.
10010011 <-- 0
00100110                   Rest.
```

Der gesuchte Rest ist also $(0100110) = (26_h)$.

[5] Alle Fehlermuster, die Linearkombinationen von $G(z)$ sind, können nicht erkannt werden.