

# FPGA DVB-S Encoder

Die Idee ist, einen DVB-S-Encoder in VHDL zu realisieren.

## Berechnung Bitrate des MPEG2-TS

- gegeben: Symbolrate 4,5 MSym/s
- QPSK, also 2 Bit pro Symbol
  - aber: aus Faltungskodierer kommen 2 Bit pro Datenbit
  - d.h.:  $2 \times 4,5 \text{ Mbit/s}$  Datenstrom am Ausgang
- Durch Puncturing: Weglassen von Datenbits, damit geringere Bitrate
  - z.B.  $2/3$ 
    - 3 Ausgangsbits pro 2 Datenbits
    - Redundanz bedeutet Faktor 2
    - also: pro Datenbit 0,75 Ausgangsbits
    - $4,5 \text{ Mbit/s} / 0,75 = 6 \text{ Mbit/s}$
- RS erzeugt aus 188 Byte immer 204 Byte
  - Geringere Nutzdatenrate, Faktor  $188/204 = 0,921\dots$
  - $6 \text{ Mbit/s} * 0,921 = 5,529 \text{ Mbit/s}$
- Also Gesamtechnung: Sendebitrate / Bit pro Symbol / Puncturing factor \* RS-Faktor

## Links / Referenzen

- [ETSI-Standard DVB-S](#)
- [drmpeg gr-dvbs](#)
- [TS-Erzeugung CBR](#)

## Schnittstellen

- Schnittstelle zum PC: Ethernet (UDP)
- Schnittstelle zum I/Q-Modulator: 2xDAC

## Komponenten

Die geplante Komponentenstruktur wurde in KiCAD erstellt, was bei der Planung ungemein hilft:

- [Download Blockschaltbild](#)
- [KiCAD-Projekt und Schaltplanfiles](#)

Designfragen:

- Können die FrameSync-Eingänge einfach durch den Reset ersetzt werden / sind sie notwendig?

Bis nach dem Interleaver ist die Struktur byteweise, danach arbeitet sie bit-seriell. Die Pipeline muss vor dem RS-Encoder aller 188 Byte angehalten werden können, damit der RS-Encoder seine sechs

Paritätsbytes einschieben kann.

## Controller

Aufgabe:

- Datenstrom überwachen (Frame-Synchronität)
- Steuersignale für die einzelnen Komponenten erzeugen

## Netzwerk-RX

Aufgabe:

- Empfang von UDP-Paketen (Sanity-Check)
- Weiterreichen der Nutzdaten an FIFO

## Netzwerk-TX

Aufgabe:

- Auswerten der FIFO-Signale und Erzeugung von UDP-Nachrichten zur Datenflusskontrolle
- Wenn FIFO fast leer: „Mach schneller“ senden
- Wenn FIFO fast voll: „Mach langsamer“ senden

## FIFO

Aufgabe:

- MPEG-Datenstrom von Ethernet entgegennehmen und an Encoder weitergeben
- Signalisierung der noch vorhandenen Daten (zu viel / zu wenig)

## Scrambler

Aufgabe:

- Entsprechend der Position im Frame Scrambling anwenden
- MUX Adaptation (7 von 8 Sync Words invertieren)

## RS-Encoder

Aufgabe:

- Verkürzten RS-Code auf jeweils einen MPEG-Frame anwenden
- Paritätsbytes einfügen
- Bei jedem CE-Signal ein Ausgangsbyte erzeugen!

## Interleaver

Aufgabe:

- Vertauschen der Byte-Reihenfolge

## P/S-Converter

Aufgabe:

- Byteweisen Datenstrom in Bitweisen Datenstrom wandeln

## Convolutional Coder

Aufgabe:

- Faltungskode auf serielle Daten anwenden

## Mapping

Aufgabe:

- Mapping auf I/Q-Symbole
- optional: Puncturing - wird erstmal weggelassen

## (Interpolator)

Aufgabe:

- Einfügen von Nullen in den Datenstrom

## Baseband Filter

Aufgabe:

- Spectral Shaping mit RRC-Filter

From:

<http://www.loetlabor-jena.de/> - **Lötlabor Jena**

Permanent link:

<http://www.loetlabor-jena.de/doku.php?id=projekte:das:dvbs&rev=1422783942>

Last update: **2015/02/01 09:45**

