

# SINGLE FPGA IMPLEMENTATION OF A DIGITAL MODULATOR FOR DVB-S

Henk Medenblik, PE1JOK

Werner Damman, PE1OBW

<http://www.d-atv.com>

[pe1jok@d-atv.com](mailto:pe1jok@d-atv.com), [pe1obw@d-atv.com](mailto:pe1obw@d-atv.com)

## Abstract

*A single FPGA solution for a digital modulator is presented. The modulator is suited for DVB-S compliant amateur television transmissions. The architecture of the FPGA is discussed. A rate adapting transport stream multiplexer connects two external MPEG2 transport streams together with a special real-time audio interface and internal DVB-SI table generator to the different signal processing blocks of the modulator.*

*DVB-S compliant forward error correction is performed with selectable puncturing ratios. Finally pulse shaping and programmable interpolation is performed which results in two 10 bit output words. These output words are clocked at a rate of 100 MHz and fed to the external dual Digital to Analog Converters for the I- and Q channel. The FPGA design also implements a memory mapped microcontroller interface which is used for the setting and readout of several parameters. The final implementation which was written in VHDL fits in a cheap 144 pin Xilinx XC2S100E Spartan IIE series FPGA.*

## 1. Introduction

Historically radio amateurs have been using analog modulation techniques for the transmission of audio and video signals. In years past amateur television transmissions required the use of Amplitude Modulated transmitters. Later these were replaced by the much simpler Frequency Modulated TV transmitters which required the same techniques as being used for the now outdated Analog Satellite TV systems.

The upcoming popularity of amateur television has lead to a highly populated frequency spectrum at the lower microwave bands. In order to find newer

techniques which occupy less frequency space and the fact that commercial developments in the field of Digital Television became widely accepted a new dimension was born in the field of Amateur Television.

The digital amateur television techniques, which are presently used, are based on the standards that currently exist for commercial digital television. These standards are defined by the DVB organization. Three different transmission media are defined and therefore three different standards are developed. DVB-T(terrestrial) is the standard for terrestrial transmissions where multipath effects dominate the transmission trajectory. This technique uses OFDM (Orthogonal Frequency Division Multiplexing) as the basic modulation format. It requires hardware consuming Inverse Fast Fourier Transformation processing which at present time requires too much hardware and is too expensive for amateur implementations. DVB-C(able) is the standard which has been developed for cable transmissions. Because cable transmission media are more or less well defined environments, it is possible to use higher order Quadrature Amplitude Modulation techniques and simpler forward error correction blocks. Finally, for the transmission of Satellite to Earth signals, a third standard has been developed which is known as DVB-S(satellite). The latter one uses QPSK as modulation format but also uses (besides the DVB-T standard) the heaviest form of forward error correction as defined by the DVB organization. When looking at the most suitable standard which can be used for amateur television then DVB-S becomes the winner. This is based on a tradeoff between transmission path impairments, required hardware complexity and reuse of commercial

available cheap settop-boxes. Currently most of the Digital Amateur Television realizations are based on this DVB-S standard [1].

This paper presents the design of a digital modulator according to the DVB-Satellite standard which we have developed in a single FPGA.

## 2. Architecture

The basic architecture of the FPGA design can be divided in four main blocks. This is given in Fig. 1.

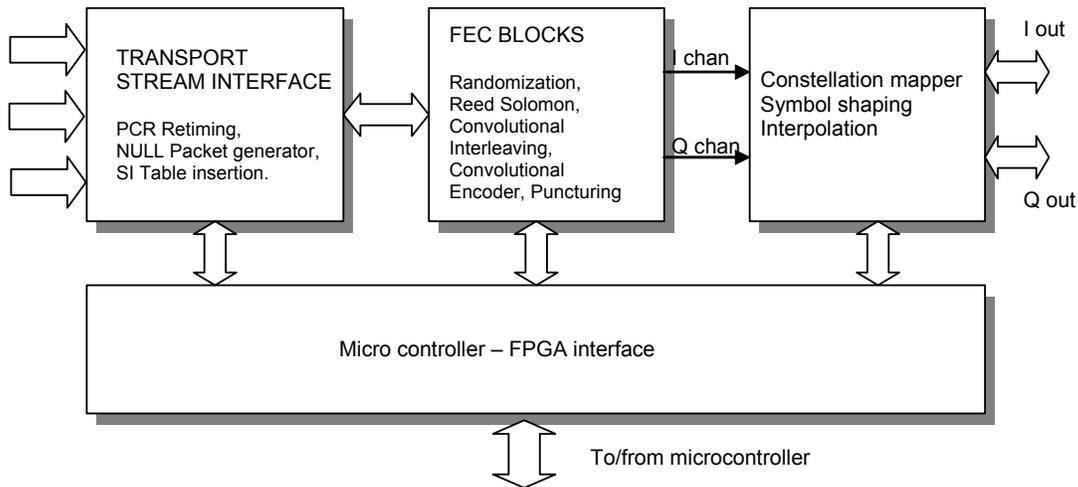


Fig. 1. Basic architecture of the modulator implementation.

### A. Transport Stream Interface

The first block is used for the interfacing with the incoming MPEG2 transport streams. One of our objectives was to be able to interface with two different external MPEG2 encoded transport streams. Besides the interfacing with two independent MPEG2 transport streams we also felt a need to develop a real-time audio interface. The real-time audio interface can be used when real-time audio transmissions are desired (for example during ATV duplex transmission) which do not undergo the inevitable delay of approximately 500ms due to MPEG2 video encoding. The real-time audio interface is not compliant to the DVB-S standard but is based on a user extension on the MPEG2 transport stream definition.

The transport stream interface also implements the rate adaption between the MPEG transport stream bursts and the actual modulator block which runs at a selectable constant symbolrate. The interface

consists of a transportstream multiplexer with automatic MPEG2 null packet insertion. This multiplexer sequentially selects one of the available transport stream inputs or packets coming from the real-time audio interface when the appropriate TS buffer is filled. In the particular case when no packets are available the multiplexer automatically inserts MPEG2 null packets which are generated inside the FPGA.

This way there will always be a constant input datastream, which is required for proper operation of the modulator. One problem with multiplexing is that it introduces a variable delay for the video and audio packets arriving at the decoder. Multiplexing causes PCR timestamp jitter on the actual original MPEG2 streams. PCR timestamps are special clock references which are generated by the transport stream multiplexer during encoding. They enable the decoder to recover the encoder clock, assuring correct frame ordering and synchronous presentation of all MPEG streams, audio and video,

belonging to the program. The MPEG2 standard defines a maximum value for PCR jitter. In our design we have solved this problem by regenerating the PCR timestamp information at the time that packets are pulled out of the transport stream buffers in the multiplexer before they are fed to the first signal processing blocks of the modulator. This way we can achieve the required specification of time stamp jitter.

At last the multiplexer contains an extra input to the internal SI table packet buffer. A subset of the DVB specification defines Service Information signaling [2]. The SI tables provide information about the MPEG2 streams, about stream content, video and audio PID values, EPG info etc. The SI tables are written in FPGA memory by an external microcontroller. This is established by creating a memory mapped microcontroller interface which makes it possible to write and read several parameters.

### B. FEC Blocks

Following the transport stream multiplexer we find the DVB-S specific building blocks. The data coming from the multiplexer is fed to the randomizer block which takes care for the bitwise randomization process. The randomization is performed by a PRBS generator with polynomial  $1 + X^{14} + X^{15}$ . Although the actual randomization process is a bitwise sequence, we decided to implement this block as a byte oriented operation. This is due to the fact that the preceding and following blocks are byte oriented. The randomizer

also performs MPEG2 sync byte inversion at every first packet in a group of eight packets.

Following the randomizer we find the first FEC block which is a Reed Solomon RS(204,188,T=8) encoder. This block adds 16 checksum bytes (2T) at the end of a 188 byte input packet. Reed Solomon codes are block correcting codes and therefore ideal for correcting burst errors. The Reed-Solomon codeword is generated using the polynomial:

$$g(x) = (x + \alpha^0)(x + \alpha^1)(x + \alpha^2) \dots (x + \alpha^{15})$$

where:  $\alpha = 02_{\text{HEX}}$  (1)

Using the field generator polynomial which is defined as:

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1$$
 (2)

the generator polynomial can be expanded to :

$$g(x) = x^{16} + \alpha^{120}x^{15} + \alpha^{104}x^{14} + \alpha^{107}x^{13} + \alpha^{109}x^{12} + \alpha^{102}x^{11} + \alpha^{161}x^{10} + \alpha^{76}x^9 + \alpha^3x^8 + \alpha^{91}x^7 + \alpha^{191}x^6 + \alpha^{147}x^5 + \alpha^{169}x^4 + \alpha^{182}x^3 + \alpha^{192}x^2 + \alpha^{225}x^1 + \alpha^{120}$$
 (3)

The codeword is constructed using:

$$c(x) = g(x).i(x)$$
 (4)

where  $i(x)$  is the input information block,  $g(x)$  is the generator polynomial and  $c(x)$  is a valid codeword.

The final implementation of the Reed Solomon coder requires 16 Galois Field multipliers which we implemented very efficiently by expanding the mathematical representation of a multiplication of two GF(256) elements [3].

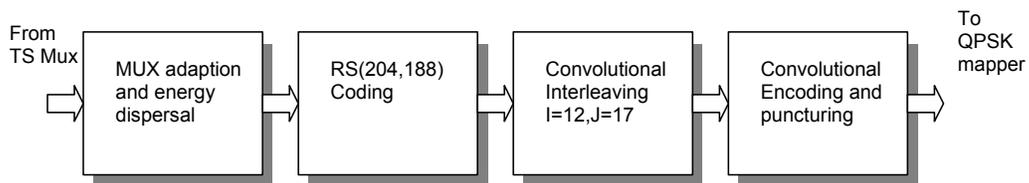


Fig. 2 Functional block diagram of Forward Error Correction section

The encoded 188 byte packets which result in 204 byte packets are interleaved by a convolutional interleaver with interleaving depth  $I=12$  and branch delay  $J=17$ . The interleaver assures spreading of burst errors by filling and reading a block of memory according to a certain pattern. We implemented the convolutional interleaver with dual ported memory, which is available on the Xilinx Spartan IIE FPGA devices. The total required amount of memory can be calculated from:

$$Memory = \frac{I \cdot (I-1)}{2} \cdot J \quad (5)$$

The interleaver therefore requires a total of 1122 bytes of memory.

The last block of the FEC is the convolutional encoder which basically consists of a  $\frac{1}{2}$  rate convolutional encoder with constraint length  $K=7$  followed by a programmable puncturer. The resulting combination is capable to perform convolutional encoding with code rates  $1/2$ ,  $2/3$ ,  $3/4$ ,  $5/6$  and  $7/8$ .

### C. Mapping, shaping and interpolation

The convolutional encoder from the FEC block outputs a 2-channel stream. These streams form the I- and Q inputs for the constellation mapper. It maps an I/Q pair onto a QPSK constellation. This means that the two bit combinations of the I/Q channel are mapped on to one of four symbols.  $I/Q=\{0,0\}$  maps to the vector  $\{1,1\}$ ,  $I/Q=\{0,1\}$  maps to  $\{1,-1\}$ ,  $I/Q=\{1,0\}$  maps to the vector  $\{-1,1\}$ ,  $I/Q=\{1,1\}$  maps to  $\{-1,-1\}$ .

Pulse shaping is performed on these symbols with a dual channel four times interpolating Root Raised Cosine FIR filter. If we look at the constellation mapping which results in symbols with values 1 and  $-1$  then we see that this would require a two bit input RRC filter (sign bit) which is quite inefficient for this purpose. A more efficient solution is to combine the constellation mapping and RRC filter. Therefore, a 'zero' at the input of the FIR is convolved with a positive impulse response at the output while a 'one' at the input is convolved with a

negative impulse response. Such a filter can be realized with a one bit input FIR.

Following the Root Raised Cosine filters we finally find the programmable interpolator. The programmable interpolator upsamples the outputs of the I- and Q Root Raised Cosine filters to the final system clock which in our case is 100 MHz. This is the final sample clock, which is used as the sample clock for the two-channel DAC's. Two Cascaded Integrator Comb interpolators perform the final interpolation [4]. These architectures can be realized very efficiently in hardware because such filters do not require multipliers. The interpolators can be built with simple adders. The final interpolation ratios are programmable by integer values between 2 and 12. Given the final system rate of 100 MHz and the fixed 4 times oversampling rate of the dual Root Raised Cosine filters, this results in user selectable symbol rates between 12.5 and 2,083 Msym/s.

One disadvantage of the original CIC structure is the fact that this structure introduces droop in its passband. This causes inter-symbol-interference (ISI) on the overall signal. The droop also varies slightly with the interpolation ratio. However, the droop can be compensated by pre-shaping the original filter coefficients of the preceding Root Raised Cosine filters. By calculating these compensated values for a midrange interpolation value we can achieve an overall flat passband. Simulations of final Error Vector Magnitude (EVM) for the modulator showed low EVM values ( $< 500\text{m}\%$ ) for all different interpolation ratios.

### 3. Conclusion

The presented architecture enables a high performance and efficient solution of a programmable digital modulator for use with DVB-S compliant amateur television transmissions. The given architecture limits the maximum supported symbol rate to  $1/8^{\text{th}}$  of the final system clock. Given the objective to find spectrum efficient transmission methods for analogue amateur television, this architecture provides a good, cheap, flexible and programmable alternative.

## References

- [1] EN 300 421 v1.1.2, "*Digital Video Broadcasting (DVB), Framing structure, channel coding and modulation for 11/12 GHz satellite services*", ETSI Standard, 1997-08.
- [2] EN 300 468 v1.5.1, "*Digital Video Broadcasting (DVB), Specification for Service Information (SI) in DVB systems*", ETSI Standard, 2003-05.
- [3] Ahlquist, C.G.C., B. Nelson, and M. Rice. "*Optimal Finite Field Multipliers for FPGA's*", Brygham Young University, 1999
- [4] E.B. Hogenauer, "*An economical class of digital filters for decimation and interpolation*", IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-29, pp. 155-162, Apr. 1981