Soft-DVB

A Fully-Software DVB-T Modulator Based on the GNU-Radio framework

September 16, 2008

Contents

Introduction

Full-hardware solutions:
A heavy burden for further DSP development

August 2007, DVB-SCENE, monthly issue of official DVB Project's magazine, states: " ... with the publication of the Commercial Requirements came voices from countries which are close to launching DVB-T services, and those which have yet to choose their preferred terrestrial broadcasting technology: Is there something wrong with DVB-T?", we heard, and, "Do I need to wait for DVB-T2 to be able to deliver HDTV?". Well, of course, the answer to both questions is a resounding NO! [...] So good luck to DVB-T2, but please don't interrupt your plans for DVB-T services!"

So, what better example of a hardware-imposed development bottleneck could be given than such an explicit call by such a standardizing body as ETSI ?
Actually, the problem is very well known and affects any transmission technology once its market has grown up to maturity and turned into the classical "consumer replacement market" model. Newer versions of transmission standards take up slowly, and typically obstruct further deployment of older ones.
Thus, conspicuous development efforts in signal processing and related technologies are often frustrated for not being revolutionary but just evolutionary. Disadvantages follow for both consumers and service providers, having to give up better technologies in favour of older, better spread ones.

Wouldn't it be much better if we could fully reconfigure all of our radiotransmission devices? Service providers could then remotely manage the components of their network, continuously evolving their transmission technologies and keeping them up with the constant advances in the field of digital signal processing.
Benefits yielded would include better spectrum usage, lower RF power emissions, lower costs and wider offer to the end consumer.

Aim of this *thesis* is to provide a proof of concept implementation of a fully functional, *realtime, realworld-deployable* ETSI DVB-T [1] modulator which is *entirely implemented using structured C++ code* within the GNURadio hybrid C++/Python framework [2].

Such *fully software* modulator is called *Soft-DVB* and is designed to reach realtime requiring just modest computational power, as shown in chapter 5.
We believe the implementation results reached during this research work and described within this thesis prove feasibility as well as peculiar advantages of fully software (SW) solutions for modulation of digital media broadcasting signals.

Soft-DVB modulator has been demonstrated as a fully functional prototype during the software radio dedicated conference WSR08, held on March 5th - 6th 2008, in Karlsruhe, Germany.

A real world example: FM Broadcast, persistence of an old technology

Probably the best known example of hardware-imposed development bottleneck is what happened to the Digital Audio Broadcasting (DAB) standard: standardized in 1993 and intended to quickly replace classic Frequency Modulation (FM) Broadcasts, today DAB is only implemented by 1,000 transmitting stations worldwide. Far less than a market success.
So, why that? Isn't DAB a good standard?
It surely is, as far as it provides significant multipath resistance, high spectral efficiency and flexibility in tayloring spectral resource consumption to the specific audio broadcasting application requirements.
Yet, even if no such feature is present in analogue Frequency Modulation broadcasting technique, FM is by far the most widely deployed technology for audio broadcasts, throughout the world, and looks likely to stay.

The reason for such apparent oddity is definitely that DAB advantages over FM, though undoubted, don't yield sufficient benefits to sustain the cost of replacing a whole transmission infrastructure.
A Software Defined Radio (SDR) approach to media broadcasting involving both transmit and receive sides would instead make this cost negligible.
Also, the average lifetime of hardware (HW) equipment would be dramatically increased, making additional economical resources available for research and implementation in the field of signal processing.
The pace of the media broadcasting market would therefore be highly boosted by new standards, applications and services being deployed with unprecedentedly short *time-to-market*. Moreover, players in the development arena could be much more confident about their research investment not being frustrated by the market inertia they can't control.
Such a fully *software-radio* dominated media broadcasting scenario, we believe, is very likely to result in much higher growth rates for the entire broadcasting industry.

Modern, Competitive Multimedia Broadcasting Infrastructure: a Wishlist

In times of widespread Internet Protocol (IP) communications, and at the dawn of highly promising peer to peer TV technologies, media broadcasting simply can no longer be about presenting the user a few images that were shot quite away from him.
Technologies such as the Internet that today are just complementary, could turn into scary competitors as network capacity increases and users' home computers are turned by new codecs, and by smart software in general, into powerful, customizable media collection and playback devices.
In order to keep useful and profitable, a media broadcasting network must then evolve and enhance its peculiar advantages over IP based infrastructure such as reliability and unique intrinsic scalability, which both are vital in the case, for example, of nationwide (or worldwide) realtime events.
Yet this is not sufficient, because, if these were the only interesting services obtainable from a broadcast network, then just a fistful of broadcast channels would be enough to serve a whole national community.
In fact a competitive broadcast infrastructure must also be able to keep up with evolution in compression and modulation techniques, in order to provide the final user with state of the art multimedia experience or it would surely be defeated by any well packaged new high-definition codec that can be easily downloaded from the network and installed on a home computer.
Novel network architectures too are a remarkable need. Reducing the emitted power along with increasing the number of transmitting stations could open up interesting possibilities to use Ultra High Frequency (UHF) spectrum to deliver a Video On Demand (VOD) offer through a cellular DVB-T network.
This would allow end users to customize the media content they buy, which is not possible with current broadcast networks.
It is also clear how such architectures must rely on low cost transmitters, given the high number of them that would be required, which, again, calls for fully software implementation of broadcasting standards.

The Full-Software proposal: a bridge between research effort and deployment

Then what? Of course, the word *software* is often pronounced with too much ease when people look for the cause or the solution of every “evil”. Though, in this case we believe that a SDR approach to Digital Multimedia Broadcasting is not just viable but also reliable and shows good performance.
In fact, by combining SDR technology with good code optimization techniques, an accurate

mathematical description becomes enough to obtain practically any real-world radiocommunication signal, and people with robust signal processing background can directly implement, test and debug their desired waveform.
Obtaining deployment-ready transmission systems is then just a matter of some more on-field benchmarking.

Then, in a mature SDR Broadcasting scenario, once a technology or a standard has been validated on-field, software updates for all of the network components can be packaged and remotely installed on the devices that are to be affected. Such a process can be fully transparent to the final user, that will simply notice an improved multimedia experience.

At this point, some may object that, given the finite computational resources available for our SDR transmission devices, hardware still has to be replaced when deploying a new standard and that, therefore, SDR Broadcast is not the solution.
This is not true, as, by wisely designing the processing power of single machines in early design stages, as well as by developing smart, optimized code, the hardware components of our SDR-based network can live through many releases of the transmission standard they implement.
Actually, once the transmission standard they were thought for has grown beyond their computational possibilities, they could be recycled to perform different functions. Also, being able to reconfigure all the devices, it becomes possible, when defining next generation signals, to make them “scalable”. In other words, we could enable newer, high computational power devices to act on the whole signal and to fully exploit its capacity, while reprogramming older, less powerful systems to handle totally new signals but just to the extent permitted by their computing power.

This yields a novel concept of backwards compatibility, which no longer constrains next generation transmission signals to include everything needed by older equipment. Actually development of new standards is completely free as backwards compatibility is provided by the possibility to redefine the behaviour of older, slower systems.

The Soft-DVB concept implementation: A feasibility proof

Research work carried out within this thesis is aimed at exploring real-world feasibility of fully software modulation for multimedia broadcasting. In such perspective the fully software modulator called Soft-DVB has been written as a proof of concept implementation demonstrating SDR technologies applied to ETSI DVB-T broadcasting.

After studying the ETSI DVB-T standard [1], implementation work has been divided into the three phases of selecting the appropriate hardware and software framework, implementing a

draft modulator, optimizing the code (both single functional blocks and general architecture) in order to reach realtime performance.
Chapters 1, 3 and 4 respectively cover such work stages. In Chapter 5 the performance of our final implementation is described in detail and results of laboratory tests carried out at DSP for Communications Lab, University of Pisa, Italy are provided.

Finally, chapter 6 explores possible deployment scenarios and analyzes the impact of imminent advances in computer technology upon SDR-based broadcasting techniques.
The conclusions reached during our work as well as further research directions suggested by our results are outlined in chapter 7.

Advantages of a Fully-Software solution for ETSI DVB-T signal generation

In our specific case, immediate advantages of implementing an ETSI DVB-T modulator using standard, fully portable C++ code are easy to identify, even without considering a mature SDR-based broadcasting scenario.
Actually, just to mention a few, major benefits of an SW implementation include:

i) affordable camp equipment for emergency broadcast;

ii) affordable Single Frequency Network covering limited land areas

iii) easy upgrade to DVB-T standard evolution;

iv) support for high-complexity cell-based multimedia content-distribution networks;

v) potential for using DVB-T as the transport layer technology for IP-based networks [3] in rural communities to address digital-divide impairments.

All this would already be achievable with no (or just little) development effort, that is just by deploying current (or a slightly modified) version of Soft-DVB software modulator.

Actually, the strong cost reduction yielded by performing modulation wholly via standard C++ code also opens up a broad span of possibilities in reshaping the broadcast network architecture towards higher complexity models. In fact: the lower the cost of the single transmitter, the higher the number of transmitters that can be deployed, supporting new services to the general audience.

Chapter 1

The DVB-T standard for broadcasting terrestrial digital television

Main features

Digital video broadcasting (DVB), in its terrestrial version (DVB-T), is currently the most widely deployed system for delivering standard and high definition video content to digital TV users worldwide. Although born as a European initiative, with the standardization process led by the European Telecommunications Standards Institute (ETSI) [1], DVB-T is today already deployed or adopted in more than 70 countries [4]. Such broad set of nations include most countries from Europe, Africa, the Middle East, and Oceania, and some Asian countries, including India and Singapore. As a consequence, the worldwide application, along with the extension to handheld devices, makes ETSI DVB-T the mainstream broadcasting system for both free-to-air (FTA) transmissions and conditional access content.

The main reasons for the almost worldwide application of ETSI DVB-T lie in the high spectral efficiency and in the robust multipath resistance due to the Orthogonal Frequency Division Multiplexing (OFDM) modulation technique.

The ETSI DVB-T system allows one baseband motion picture expert group-2 (MPEG2) transport stream (TS) to be transmitted over the air. (Optionally, two TSs can be combined through the use of hierarchical modulation techniques.) Each TS is designed to carry multiple audio/video and/or data channels. Distribution over single frequency networks (SFNs) is also supported.

A SDR approach also suits particularly well The DVB-T environment as in 2006, just two years after the release of currently deployed DVB-T standard [1], work began for defining the forthcoming DVB-T2 system, which is planned to be reaching the market in 2009.
Actually, next generation DVB-T should provide a 30% increased payload capacity and support

High Definition TeleVision (HDTV) delivery via Motion Picture Experts Group 4 (MPEG4) codecs. Techonlogies adopted should include Multiple Input Multiple Output (MIMO) and OFDM as well as Low Density Parity Check (LDPC) forward error correcting codes.

Typical DVB-T propagation scenarios: Rayleigh and Ricean Channels

Designed to provide digital TV services to both urban and rural areas, the DVB-T standard has been developed in order to offer good performance in both Line of Sight (LOS) and No Line of Sight (N-LOS) multipath channels. The system was validated by ETSI against the typical multipath channel model, with both Rayleigh (N-LOS) and Ricean (LOS) fading. Also, performance over the plain Gaussian channel was tested.

Fading coefficients taken into account have the form: $a = \rho e^{j\phi}$ where ρ and ϕ are random variables distributed as follows:
for the Rayleigh channel:

$$f_{ra}(\rho) = 2\rho e^{-\rho^2} u(\rho) \tag{1.1}$$

$$f_{ra}(\phi) = \frac{1}{2\pi} rect\left(\frac{\phi - \pi}{2\pi}\right) \tag{1.2}$$

for the Ricean channel:

$$f_{ri}(\rho) = 2\rho(k-1)e^{-(k+1)\rho^2 - k} I_0(2\rho\sqrt{k(k+1)})u(\rho) \tag{1.3}$$

$$f_{ri}(\phi) = \frac{1}{2\pi} rect\left(\frac{\phi - \pi}{2\pi}\right) \tag{1.4}$$

where $k = \frac{LOS-received-Power}{N-LOS-received-Power}$ and $I_0(x) = \frac{1}{2\pi}\int_0^{2\pi} e^{x cos(\theta)}\, d\theta$

System behaviour for such propagation scenarios, as well as for the plain Gaussian, has been analyzed in [1] in terms of required signal to noise ratio ($\frac{C}{N}$) to ensure Quasi Error Free (QEF) demodulation of the transmitted Transport Stream (TS).
Table 1.1 clearly shows the flexibility offered by different modulation and channel coding options in order to sustain the various severity conditions that our propagation scenario might impose.

Actually as low SNR as 3.1 (linear) can be used, providing up to 6.03 Mbps useful bitrate, while a robust SNR of 20.1 (linear) can afford up to 31.67 Mbps.
Such good performance, especially in N-LOS multipath environments such as densely populated urban areas has allowed ETSI DVB-T to outperform its American competitor, namely the Advanced Television Systems Committee Standard (ATSC), in such scenarios.
ATSC relies upon the 8-VSB modulation which is in fact much less robust to multipath propagation than DVB-T's OFDM.

Table 1.1: *System performance over Gaussian, Ricean and Rayleigh Channel*

Modulation	*Coding rate*	*Required* $(\frac{C}{N})$*, Gaussian Channel*	*Ricean Channel*	*Rayleigh Channel*
QPSK	1/2	3.1	3.6	5.4
QPSK	2/3	4.9	5.7	8.4
QPSK	3/4	5.9	6.8	10.7
QPSK	5/6	6.9	8.0	13.1
QPSK	7/8	7.7	8.7	16.3
16-QAM	1/2	8.8	9.6	11.2
16-QAM	2/3	11.1	11.6	14.2
16-QAM	3/4	12.5	13.0	16.7
16-QAM	5/6	13.5	14.4	19.3
16-QAM	7/8	13.9	15.0	22.8
64-QAM	1/2	14.4	14.7	16.0
64-QAM	2/3	16.5	17.1	19.3
64-QAM	3/4	18.0	18.6	21.7
64-QAM	5/6	19.3	20.0	25.3
64-QAM	7/8	20.1	21.0	27.9

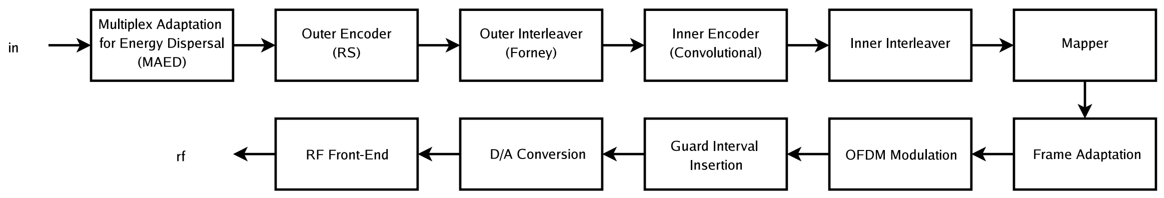


Figure 1.1: *Functional block scheme for the standard ETSI DVB-T transmitter [1].*

Analysis of ETSI DVB-T functional blocks

In this section we will briefly analyze the structure of the functional blocks that adapt the baseband MPEG2 signal to the typical multipath radio frequency channel, as described in [1]. Typically, only the transmitter device will be considered, as receive chain blocks are mainly symmetrical to the transmitting ones. Though, when needed, differences and peculiarities will be explicitly outlined.

Fig. 1.1 shows the functional block diagram of a DVB-T modulator. Each function block is shortly described as follows:

- the *multiplex adaptation for energy dispersal (MAED)* is the first block of the system, thus it receives the baseband MPEG2 input stream organized in fixed lenght packets. It then removes time domain correlation from its byte-wise input by performing a bit-level XOR with a pseudo-random binary sequence (PRBS). The proper PRBS is generated via a linear feedback shift register (LFSR) by the generator polynomial:

$$p(x) = 1 + x^{14} + x^{15} \tag{1.5}$$

  The initialization of both scrambler and descrambler is bound to the MPEG2 SYNC byte.

- the *outer encoder* performs Reed-Solomon (RS) encoding at byte level and in systematic form. The chosen code is an RS(204,188), obtained by shortening an RS(255,239). The main purpose of the RS code is to mitigate the impact of the error bursts produced at the receiver side by the Viterbi algorithm. Such RS code is capable of correcting up to 8 random erroneous bytes in a received RS word that is 204 bytes long.
  The $GF(2^8)$ Galois Field the code operates within is generated by the following field generator polinomial:

$$p(x) = x^8 + x^4 + x^3 + x^2 + 1 \tag{1.6}$$

  The code is instead generated by:

$$g(x) = (x + \lambda^0)(x + \lambda^1)(x + \lambda^2)...(x + \lambda^{15}), \lambda = 02_{HEX} \tag{1.7}$$

- the *outer interleaver* aims to minimize correlation between the residual errors at decoding time, it is implemented by a convolutional interleaver based on the Forney approach.
  The interleaved data bytes are composed of error protected packets and delimited by inverted or non inverted MPEG2 sync bytes;

- the *inner encoder* performs convolutional encoding and puncturing to achieve quasi-error-free (QEF) transmission.
  The mother convolutional code, running at a coding rate $r = 1/2$ has generators:
  $G_1 = 171_{OCT}$ and $G_2 = 133_{OCT}$;

- the *inner interleaver* provides support for hierarchical modulation and fairness in assigning the payload bits to the OFDM carriers, i.e., it prevents certain bits from the original TS from being constantly assigned to the same set of OFDM carriers with unsatisfactory signal-to-noise ratio (SNR). It is made up of two different interleavers, named "bit-wise interleaver" and "symbol interleaver", each driven by different permutation laws and working on different data blocks;

- the *mapper* modulates the encoded bits onto QPSK, 16-QAM, and 64-QAM constellations. Both non-hierarchical and hierarchical modulations are supported, thus allowing two different TSs with different error performance to be transmitted simultaneously;

- *frame adaptation (FA)* provides insertion of all needed reference signals: pilot carriers, used for synchronization and channel estimation, and carriers conveying information upon the transmission parameters being implemented. This operation is called transmission parameter signaling (TPS);

- the *OFDM modulation* block adds virtual carriers, reference signals and performs an inverse fast Fourier transform (IFFT) with 2048 (2k) or 8192 (8k) subcarriers.;

- *guard interval insertion* inserts the guard interval (cyclic prefix) required by the duration of the channel impulse response.

- the *digital-to-analog (D/A) conversion* interpolates the samples, thus producing an analog signal;

- the *RF front-end* shifts the basesband I/Q analog signal to the proper carrier frequency of the desired TV channel.

Chapter 2

The OFDM Modulation

Overview

As previously stated, ETSI DVB-T ability to outperform competitor video broadcasting systems lies in the usage of OFDM modulation technique, providing high spectral efficiency and the required robustness to multipath propagation.
Thus, this chapter aims to explore peculiarities and advantages provided by OFDM modulation in the fixed and nomadic video broadcasting scenario ETSI DVB-T operates within.
Named Orthogonal Frequency Division Multiplexing, the OFDM modulation technique belongs to the class of multi-carrier modulations. Thus, several thousands of closely spaced orthogonal sub-carriers are modulated by as many sub-flows of information bits. The input to an OFDM functional block, is then a serial flow of information bits which gets immediately parallelized into the needed sub-flows, as depicted in Figure 2.1.
The parallel sub-flows obtained immediately undergo constellation mapping. The resulting symbols are considered as frequency-domain complex samples and, therefore, assembled into the input vector to the IFFT block. As will be better outlined in the following sections, performing an IFFT on such vector guarantees the needed modulation on every sub-carrier. It can be intuitively assumed that frequency-domain complex samples yield the spectral shape of the signal whose time-domain shape is directly visible at the output of the IFFT block.
The real and the imaginary output of the IFFT block are directly fed to the Digital to Analogue Converters (DACs) of in-phase and quadrature transmitter branches. This produces the baseband OFDM modulated signal.

The demodulator structure is exactly dual to the modulator one (see Figure 2.2. It is interesting to note how the FFT block, dual to the IFFT one present in the modulator system, receives the time domain signal from the communication channel and returns, yet affected by channel noise and distorsion, the spectral shape that frequency-domain samples were meant to impose.
Though not depicted in Figure 2.2, the typical OFDM demodulator also includes a channel equalization function. Actually some of the carriers, called pilots, convey deterministic symbols whose received value allows for channel estimation in the frequency domain at the receiver side. The estimated channel frequency response is then used to compensate the distorsion introduced by the communication channel.

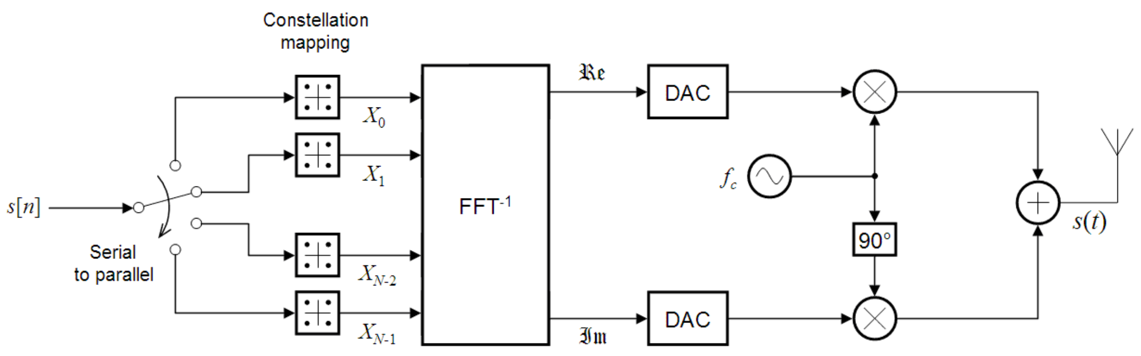


Figure 2.1: *Functional block scheme for the ideal OFDM modulator, source: wikipedia.org*

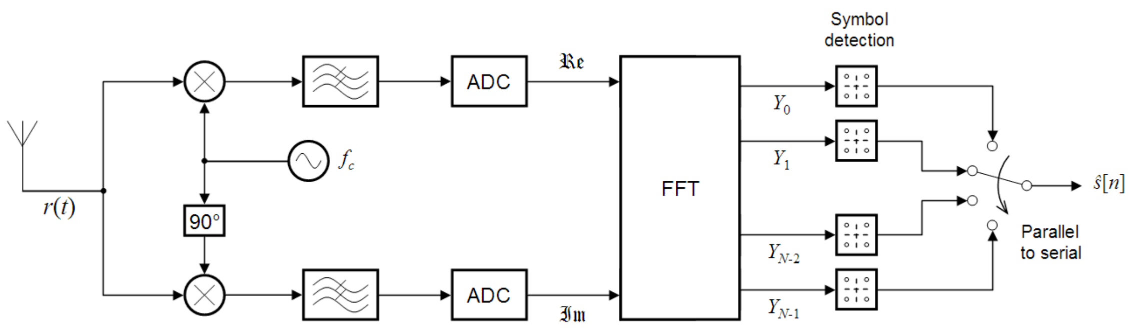


Figure 2.2: *Functional block scheme for the ideal OFDM demodulator, source: wikipedia.org*

Currently OFDM modulation technique is adopted in a variety of system, which is not limited to media broadcasting networks. Examples are DVB-T, DAB, Digital Radio Mondiale (DRM), Terrestrial Digital Media Broadcasting (T-DMB) but there are also wireless network access technologies such as 802.11a, 802.11g (commercially named WiFi) and 802.16 (commercially named WiMax).

G.DMT is the typical modulation used in Digital Subscriber Line systems (xDSL) and, though meant for network access over copper wire, it implements an FFT-based multi-carrier technique which is almost identical to what happens in OFDM. This is because of substantial similarities between frequency selectivity of radio multi-path channel and copper wire.

Usage within ETSI DVB-T: benefits and implementation choices

As seen in Chapter 1 ETSI DVB-T standard concatenates OFDM modulation with two stages of error correction coding. This is typically known as COFDM (Coded OFDM) and, by varying the overall coding rate, enables the system deployer to adapt the information speed to the actual channel capacity imposed by the contingent propagation severity, therefore keeping Bit Error Rate (BER) under control.

A relevant advantage of using OFDM lies in the modulation's intrinsic compatibility with SFNs. Actually a receiver that has visibility of two transmitters located in different spots, broadcasting the same TS, sees the two signals that approach it as two different rays of a multipath channel. Thus it addresses the frequency selectivity problem that this yields the same way it usually deals with real differently delayed replicas originating from the same transmitter.

Usage of orthogonal sub-carriers, spaced just enough to be such, provides state-of-the-art efficiency in spectrum usage.

The pilot carrier driven, frequency domain channel equalization is a highly efficient solution for exploiting the heavily frequency selective, strongly multi-path channel, that inevitably affects broadband transmissions in the typical urban scenario.

As shown in [1] the FFT size can be chosen to be 2 048 or 8 096, this is has no impact over the system's spectral efficiency but is meant to provide two different trade-off choices between the accuracy of channel's frequency response estimation and the time required to obtain it. Actually a smaller FFT size allows for a quicker channel estimation, suitable for mobile devices. A longer FFT size is instead useful for a fixed receiver which doesn't see a rapidly varying channel and can therefore spend more time to obtain a better estimate, yielding a stronger multi-path protection.

Another tunable parameter available to the system designer is the guard interval duration, which directly impacts the spectral efficiency performance. Such parameter can be set to 1/4, 1/8, 1/16 or 1/32 of the length of the useful symbol part.
From an implementation viewpoint, the longer the guard interval duration, the stronger the system robustness against significant delay spreads imposed by the transmission channel.

Orthogonality of Signals and Spectral Efficiency

Considering parallel information sub-flows to be modulated, within suitable symbol intervals, upon sub-carriers spaced of f_Δ, our ideal signal can be described by the following expression

$$x_{OFDM}(t) = \sum_{m=-\infty}^{+\infty} \sum_{k=0}^{+\infty} c_k^{(m)} p(t - mT_s) e^{j2\pi k f_\Delta t} \tag{2.1}$$

where k is the sub-carrier index, m the index of the OFDM symbol, $p(t)$ the time domain pulse shape for each sub-carrier and T_s the duration of the OFDM modulation symbol.

The received signal $r(t)$ will include a certain amount of noise that we assume to be additive, white and Gaussian distributed (AWGN). Therefore:

$$r(t) = x(t) + w(t) \tag{2.2}$$

Accordingly, the optimal decision variable for the k-th information sub-flow within the symbol m=0 is given by

$$z_k^{(0)} = \frac{1}{T_s} \int_0^{T_s} r(t) e^{-j2\pi k f_\Delta t}\, dt \tag{2.3}$$

It is then worth to examine the level of interference from the generic sub-carrier i upon our decision variable $z_k^{(0)}$ which we call $I_{i,k}^{(0)}$

$$I_{i,k}^{(0)} = \int_0^{T_s} c_i^{(0)} e^{j2\pi i f_\Delta t} e^{-j2\pi k f_\Delta t}\, dt = c_i^{(0)} \frac{e^{j2\pi(i-k) f_\Delta t} - 1}{j2\pi(i-k) f_\Delta} \tag{2.4}$$

It is immediately evident from 2.4 that, if we look for the values of f_Δ which will ensure orthogonality among sub-carriers, we obtain $f_\Delta = \frac{q}{T_S}$, q integer.
Obviously we choose $q = 1$ to avoid unnecessary spectrum consumption.

As promised, the result of all this is a very high spectral efficiency with no interference among sub-flows.
The "Orthogonal" adjective within the OFDM acronym is actually meant to stress this concept which substantially differentiates OFDM from traditional frequency division modulations.

Figure 2.3 shows a numerically simulated OFDM signal spectrum.
The blue plot is obtained by using 64 sub-carriers, the red one, by using 2 048.
In Figure 2.4 the actual spectrum of a real-world OFDM signal can be seen.

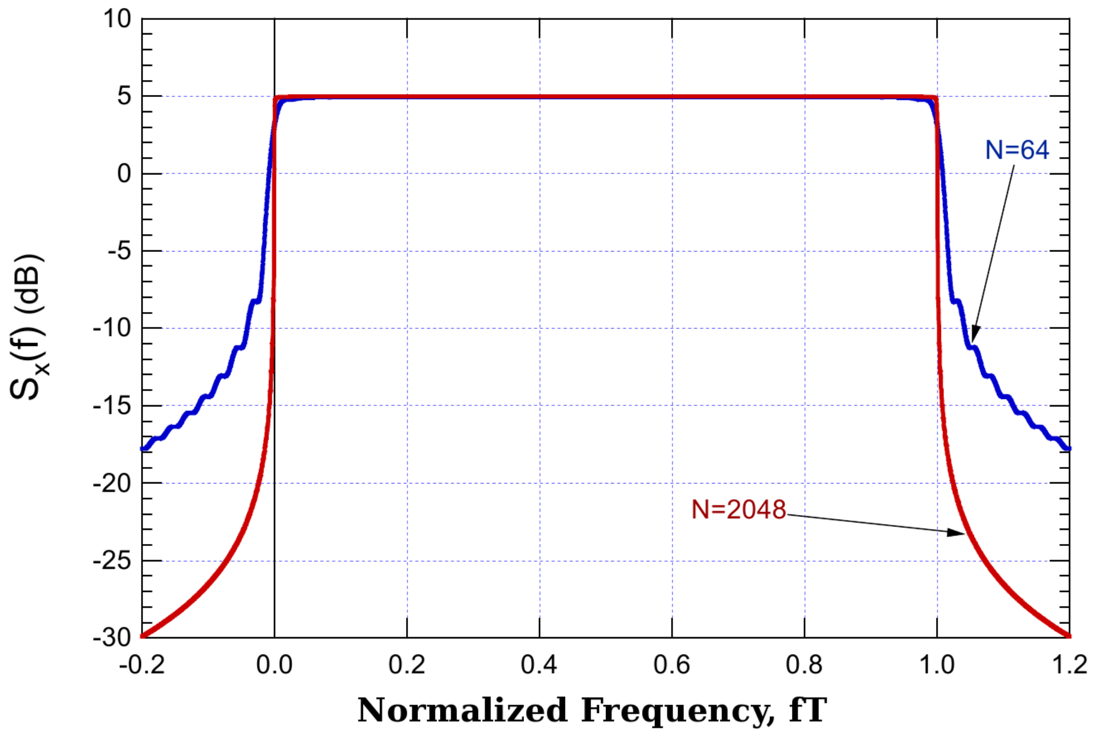


Figure 2.3: *Numerically simulated OFDM signal spectrum. The blue plot is obtained by using 64 subcarriers, the red one, by using 2 048, [5]*

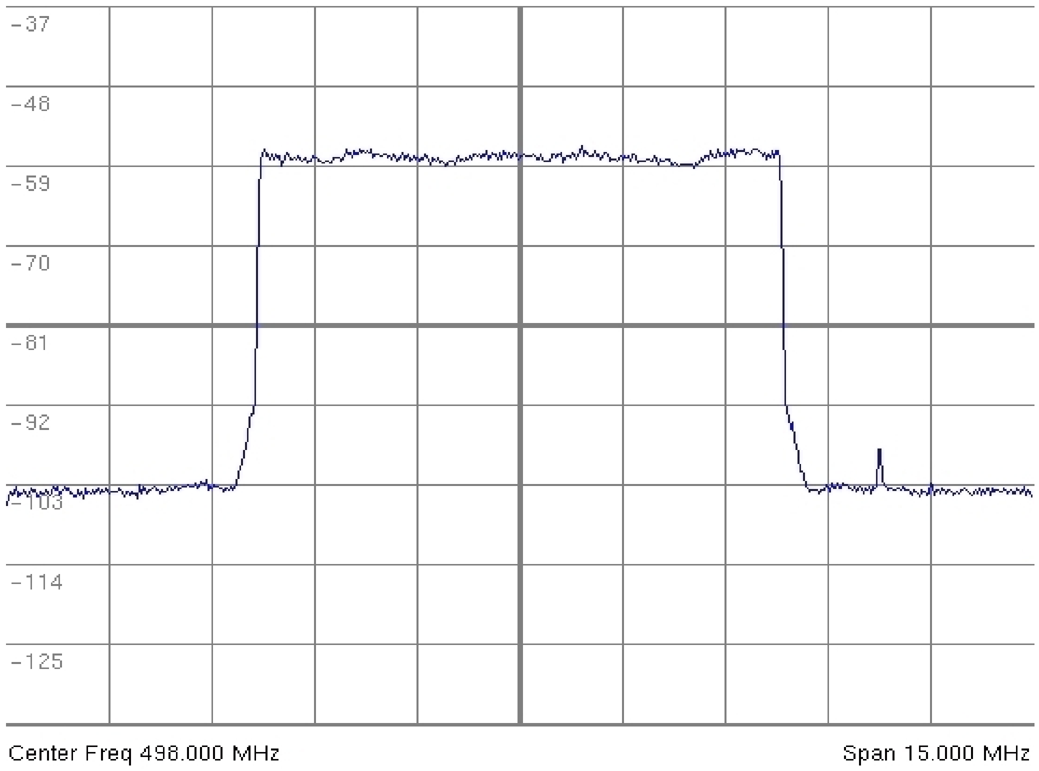


Figure 2.4: *Real world, undistorted OFDM signal, source: wikipedia.org*

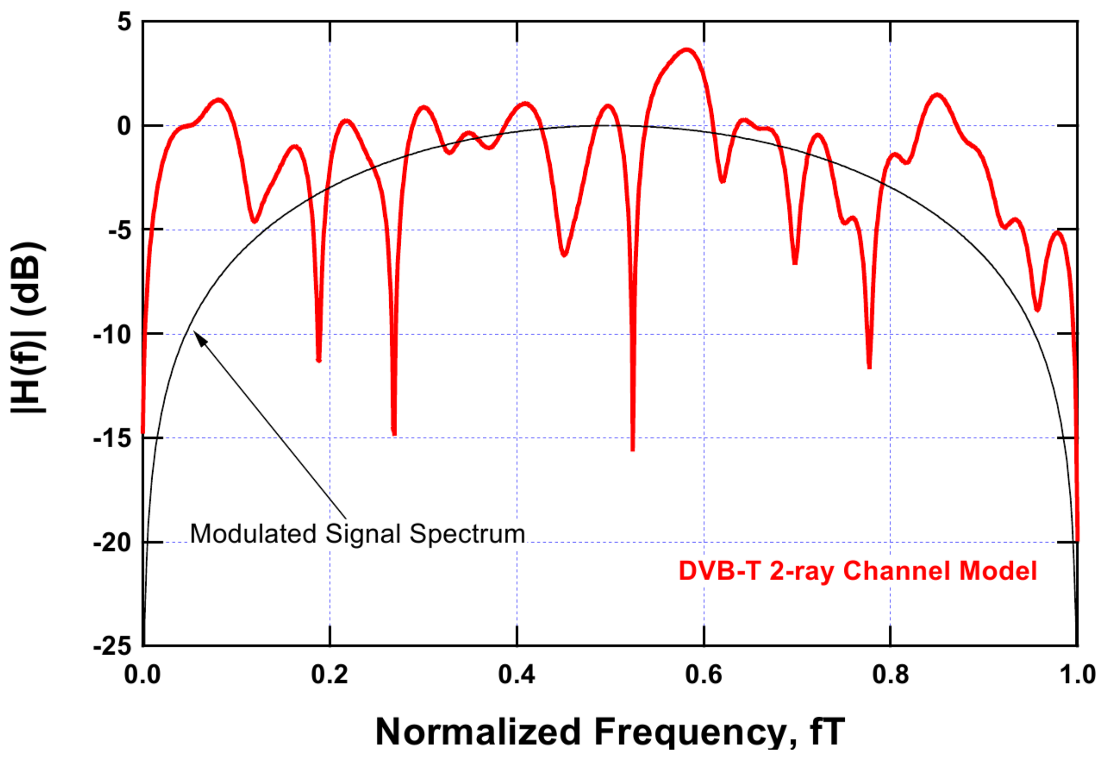


Figure 2.5: *Numerically simulated frequency selectivity of a 2-Ray multipath channel, [5]*

Multipath Resistance

The other major benefit of OFDM modulation is its robustness against the frequency selectivity of multipath channels.

Figure 2.5 clearly shows the severity of frequency selectivity on an optimistic, two-rays-only multipath channel. By inserting deterministically modulated sub-carriers within its symbol, OFDM measures the channel frequency response $H_c(f)$ both in fixed and deterministically moving points of the spectrum. Such measured values are then interpolated in order to obtain a reliable estimate of $H_c(f)$ over all sub-carriers. Such estimate is then used to remove channel distorsion as follows: the received signal spectrum being R(f), contributions to it can be expressed by

$$R(f) = H_c(f) \cdot X_{OFDM}(f) + W(f) \tag{2.5}$$

where W(f) is the gaussian noise spectrum.
Therefore, dividing (carrier by carrier, in the discrete frequency domain) the received signal spectrum by $H_c(f)$, we obtain:

$$R_e(f) = X_{OFDM}(f) + \frac{W(f)}{H_c(f)} \tag{2.6}$$

where channel-imposed distorsion is clearly compensated, even if significant Noise-Enhancing phenomena might affect sub-carriers where $H_c(f)$ assumes very low amplitudes.

Sub-carriers used for channel estimation are called *pilot carriers* and, being deterministically modulated, do introduce overhead into the OFDM transmission.
Pilot carrier modulation is performed according to a PRBS generated using a LFSR, which is both known at transmitter and receiver side.

Pilot carriers are also used for synchronization purposes.

Chapter 3

Soft-DVB

Preliminary choices: The USRP system and the GNURadio Framework

Amidst all the possible solutions to implement Soft-DVB, the GNURadio paradigm [2] has been selected for the SW section, whereas the Universal Software Radio Peripheral (USRP) produced by Ettus Research LLC [6] has been chosen for the HW section. The remainder of this chapter reports the main motivations and implications of both choices.
The main motivations that led us to choose the GNURadio framework lie in the efficiency and high flexibility provided by this solution. GNURadio is an open source project aimed at developing SDR terminals. In particular, this project provides a collection of software modules to perform the main operations for signal processing. It is interesting to note that the GNURadio paradigm currently gathers a very active worldwide community of developers building their projects upon this platform. [2]

The GNURadio framework is split in two wide areas. The first one makes use of the C++ programming language, while the latter one exploits the flexibility provided by the Python scripting language. Although C++ and Python are both objected-oriented languages, the first one is compiled, whereas the latter one is interpreted. Due to the different performance they provide, C++ is devoted to all the operations concerning signal processing within a block, whereas Python is used to instantiate and connect different blocks together.
The use of Python is particularly attractive, since it makes the whole project flexible and easily reconfigurable.

The USRP is a device designed within the GNURadio project [2] and produced by Ettus Research [7]. As a consequence, it perfectly fits the GNURadio framework considered for the SW section. Furthermore, the USRP has been selected for Soft-DVB due to the good efficiency of the device and because of its high quality/pricing ratio. In the following, a short description of the main operations performed by the USRP is given. The output of the transmission chain depicted in Fig. 1.1 is the stream of I/Q samples computed by the host computer. The interface between the SW and the HW section is represented by the universal serial bus (USB), which

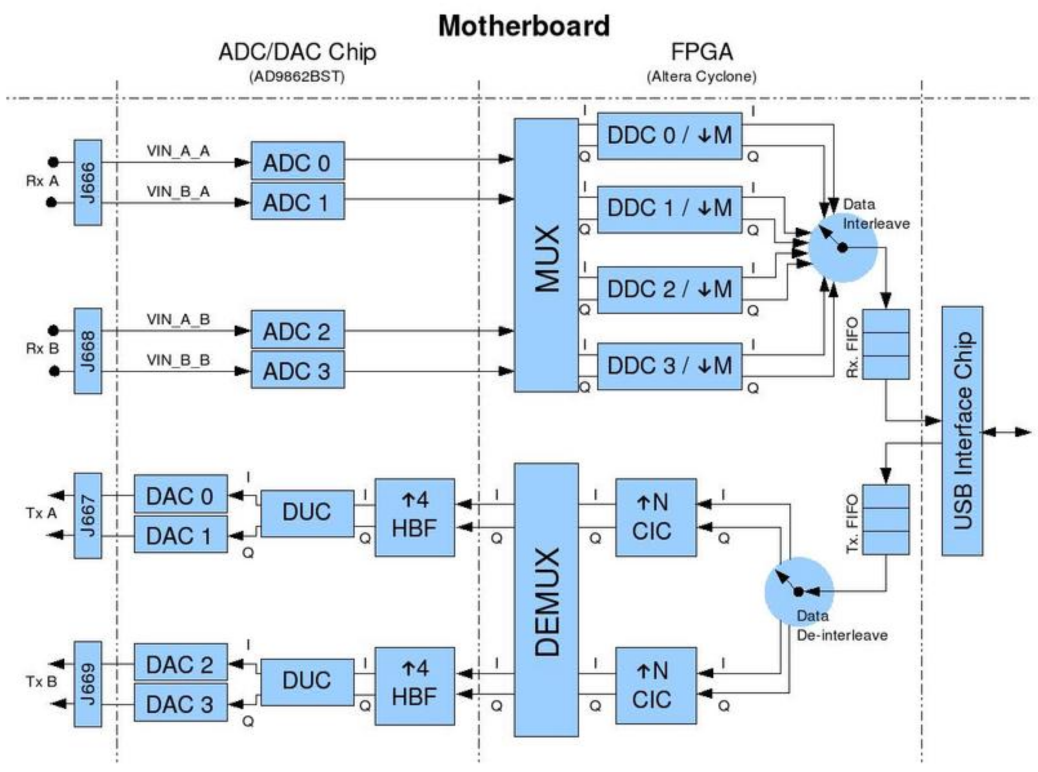


Figure 3.1: *Functional block scheme for the USRP motherboard, source: gnuradio.org*

conveys the output samples to the HW section, actually the USRP.

The USRP implements the last two blocks in the functional scheme of Fig. 1.1, namely D/A conversion and RF front-end. It is composed of a motherboard (for the D/A conversion) and a daughterboard (for the RF upconversion). The functional scheme for the motherboard is sketched in Fig. 3.1. Once the samples are delivered to the URSP, they are interpolated by means of the field programmable gate array (FPGA) section, then they are upconverted by the digital up converters (DUCs), and finally they are sent to the two digital-to-analog converters (DACs) to produce a signal in the analog domain. The analog signal is then upconverted and amplified by the daughterboard. During our test activity, and still at the time of writing this thesis, no front-end has been made available from Ettus Reserach LLC that can cover the DVB-T VHF UHF spectrum without requiring modifications to the HW. A daughterboard covering such needs is currently under development. Therefore, given the contingent lack of proper HW solutions, we use the USRP in combination with the BasicTX daughterboard, which is not meant to operate above 250 MHz [6].

Even though this solution is heavily suboptimal, we use the fourth spectral replica of the DAC output, applying a preliminary shift of $f_{DUC} = 42$ MHz via the DUC (Fig. 3.2). The DACs

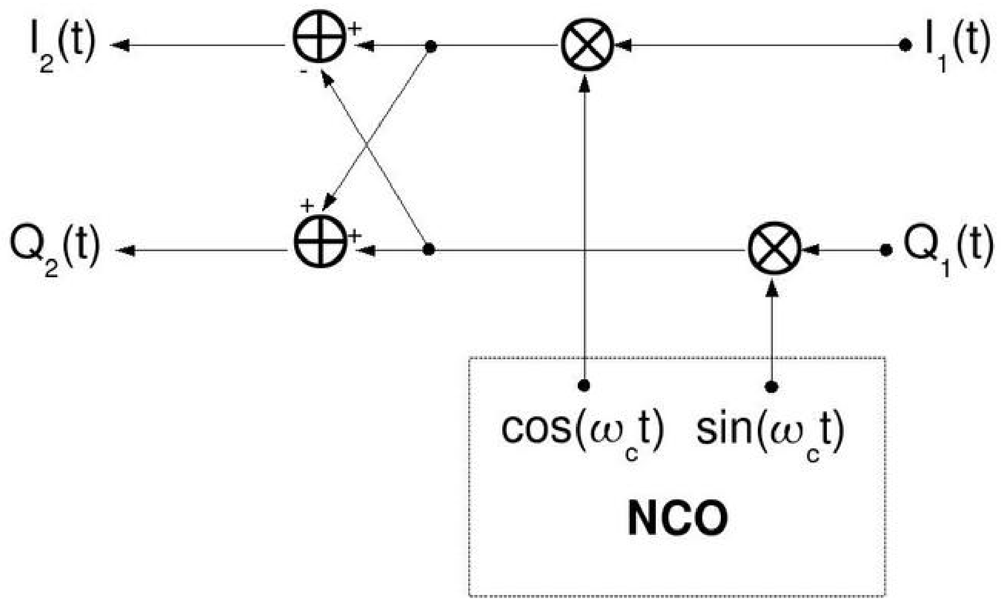


Figure 3.2: *Functional scheme of the Digital Up Converter (DUC), source: gnuradio.org*

operate at $f_{DAC} = 128$ MHz, thus yielding an RF central frequency:

$$f_{RF} = 4f_{DAC} + f_{DUC} = 554 \text{ MHz}, \tag{3.1}$$

which corresponds to channel 31 UHF. As a consequence, we use this frequency for our DVB-T test transmissions.

Another limitation in using the USRP is given by the speed of the interface with the host computer. In fact, its performance is currently limited by the USB 2.0 bus transfer rate. For commercial chipsets, the maximum reliable transfer speed over this bus is[1] $R_{max} = 32\,\text{Mbyte/s}$. When sampling an 8- MHz channel using a sampling frequency $f_s = 8\,\text{Msample/s}$, the most accurate sample representation is

$$N_{bit} = \frac{R_{max}}{2f_s} = 2\,\text{byte/sample} = 16\,\text{bit/sample}. \tag{3.2}$$

In order to represent a 16- MHz bandwidth, the sample resolution can be reduced from 16 to 8 bits. However, this yields a loss in quantization SNR equal to 48 dB, which is definitely not acceptable for any RF transmission. Thus, according to [1], an 8- MHz band forces us to implement a 7- MHz DVB-T channel. Nonetheless, it is interesting to note that the next version of USRP, which is currently under development, will remove the USB bottleneck by using gigabit ethernet. This feature will allow Soft-DVB to broadcast a full 8- MHz channel just by slightly upscaling the sampling frequency, without any change to the code.

[1]Many bargain chipsets actually provide a maximum throughput far lower than this limit.

Table 3.1: *Mapping of DVB-T blocks into Soft-DVB C++ blocks.*

DVB-T block	*Soft-DVB C++ block*
MAED	dvb.maed_bb
outer encoder	dvb.ocoder_bb
outer interleaver	dvb.ointerleaver_bb
inner encoder	dvb.icoder_bb
inner interleaver	dvb.iinterleaver_bb
mapper	dvb.mapper_bc
frame adaptation	dvb.insert_refsig_cc
OFDM	gr.fft_vcc, dvb.insert_vica_cc
guard interval insertion	dvb.insert_gi_cc

Soft-DVB Architecture

As stated in the introduction, Soft-DVB represents a proof-of-concept DVB-T modulator designed following the software-defined radio (SDR) paradigm. In a SDR architecture, most (if not all) signal processing is performed digitally via SW routines. By inspecting Fig. 1.1, it is possible to identify the last two blocks as the only strictly HW components required. Hence, in the spirit of designing a fully-SW DVB-T modulator, the architecture selected for Soft-DVB only consists of the front-end as the HW section, whereas all the other functional blocks are implemented through the already mentioned C++ SW modules.

All such signal processing blocks have been developed from scratch by the authors to have full control over the computational efficiency of the modulator.
The only exception is represented by the OFDM block, in which the fast Fourier transform (FFT) is performed by making use of the algorithm developed in [8]. As shown in Table 3.1, each digital signal processing (DSP) block reported in Fig. 1.1 is directly mapped into one of the Soft-DVB C++ DSP blocks.

DSP blocks are assembled and configured via a Python script named Soft-DVB.py, which also controls the USRP interpolation rate and the RF channel center frequency.

The necessary (forward-only) communication within blocks is provided by the GNURadio run-time system, which ensures the data flow to proceed reliably throughout the entire processing chain. GNURadio also provides Soft-DVB with the appropriate interface towards the USRP, namely the GNURadio block *usrp.sink_c()*

Development Stages: the check-dumps

Soft-DVB development has been a "step by step" process: the task of any single block was at first read from the ETSI standard [1], then it was sketched on paper in order to define an appropriate efficient algorithm to implement it.
Afterwards, the corresponding C++ block was written following the GNURadio framework recommendations and it was given a known input. The resulting output was checked against a manually computed output, in order to validate the behaviour of the block.

Once this validation step was passed, the block was operated over a much longer test input, the resulting output was then validated against the output produced by the equivalent Matlab script belonging to the Matlab open-source DVB-T implementation by Gordon Cichon [9].

Such procedure was iterated for each Soft-DVB GNURadio block, until the end of the processing chain was reached.
Obviously, when our blocks' boundaries happened to be different from Mr Cichon's, only the first validation step could be performed.

Suitable software routines were written to perform the comparison between each Soft-DVB block's output and the one from Mr Cichon's Matlab scripts.
Such routines had to take into account the difference in data representation between Soft-DVB processing chain and the Matlab simulator. Actually, Soft-DVB's I/Q signal samples are represented using the GNURadio data type named *gr.complex*, which assigns 32 bits to the in-phase channel and 32 to the quadrature one. The Matlab simulator instead uses 64 bits for both I and Q channels.

However, given the same input, at the end of the processing chains of both systems (i.e. the Matlab simulator and Soft-DVB first Draft), differences between calculated signals were negligible, i.e. below 10^{-5} on every complex sample.

Figure 3.3: *The test setup for Soft-DVB's first draft. The 7200rpm Seagate Barracuda disc is visible on top of the pc*

TPS Implementation

The Transmission Parameters Signalling (TPS) subsystem output could not be checked against any preexisting implementation of any kind. It was thus validated only through "hand-made" computation of its output.
Such output is however fully deterministic and depending only on the transmission scheme implemented and on the evolution of a dedicated PRBS.

The TPS block payload was computed off-line and then hard-coded into the block's executable.

The BCH (Bose Chaudhuri Hocquenghem) redundancy of TPS information was also precomputed because of the deterministic nature of the transmission parameters to be signaled.

Hardware trivia and Headaches

Once the firs Soft-DVB draft was completed, it was time to get it to an off-the-shelf receiver to validate the entire modulation process and to debug, if and where necessary, the signal.
Such first draft of Soft-DVB code ran in about 7 times the real time on a 3.0 GHz Intel Pentium 4 CPU. Therefore it could only be used to pre-calculate the signal, which had to be sent out towards the receiver afterwards, playing it back from a disc.
Thus, a high throughput disk was needed. It was in the end found that a 7200 rpm Seagate Barracuda SATA2 hard disc (see Fig. 3.3) could do the job and reach the necessary sustained throughput of 32MB/s.
The format used to store the samples onto the disc was GNURadio's *gr.interleaved_short* data type, which is consistent with the above mentioned throughput.

Another relevant issue was the USB 2.0 chipset. At first, a high number of test transmission were unsuccessful because the outwards nominal, and advertized :-), throughput was not reached by the USB chipset installed on the used machine.
Once this was solved by resorting to an Intel 945 series chipset, the first correctly-shaped OFDM signal was transmitted using the Basic-Tx frontend and displayed on screen by a spectrum analyzer at DSPCOLA.
This was the first real-world check obtained by Soft-DVB signal.

It was then time to start transmitting our 7 MHz DVB-T channel to a commercial set-top-box to validate the modulator. First, we had to check if the work-around of using the forth DAC replica could really be used.

To do this, we recorded a standard DVB-T signal transmitted by the national Italian broadcaster RAI, and re-sent it towards our set-top-box, namely an Access Media STBL3012, through a Basic-TX transmission front-end. This was again quite critical, as the TVRX receiver front-end we used in this phase to capture the RF signal heavily cuts an 8 MHz wide signal at the band margin.
Nevertheless, using this technique it was found that, at 554 MHz (channel 31 UHF), we could obtain acceptable performance from the Basic-Tx. Actually, re-transmitting the recorded RAI DVB-T signal, we found that we could have the STBL3012 to sporadically lock to the MPEG2 SYNC byte.

On Air... pardon, on Coax

At this time we could really attempt to demodulate our draft signal by playing it back from the right "quick" hard disk, through the right "quick" USB 2.0 chipset, in the right spectral

position.
Doing exactly this we found that the signal could be demodulated almost correctly: actually channel allocation tables could be demodulated, read and stored into the set-top-box memory and the audio/video stream could be presented on screen.
Still, something wasn't yet all right, as frequent discontinuities were noticeable in the audio/video flow.

Signal Debugging

The causes of such problem were investigated and it was found that the first signal draft was affected by a mistaken initialization of the TPS modulation. This was corrected and the audio/video flow started to be displayed flawlessly.

The next step was the validation of the signal also against a different kind of receiver. A bargain DVB-T USB-STICK receiver was used for such purpose. The actual model was the Pinnacle PCTV 70e. Again, audio/video presentation was flawless and we started considering our signal correct and reliable.

At this point we had implemented a not-yet-realtime ETSI DVB-T modulator, with:

- 11.612 Mbps useful bitrate
- 16-QAM mapping
- 2/3 Coding rate
- 1/4 Guard interval

The subsequent aim was to reach realtime operation, while keeping the already implemented transmission parameters. Nevertheless, before proceeding towards this target, a deep need of our soul had to be satisfied... :-)

On Air, Actually!

Really, there is nothing cool about implementing a radio frequency communication standard from bottom up if your work does not, sooner o later, interact with the real electromagnetic spectrum.

In other words, after so much time and so many energies have been spent, you really need to actually transmit over the air what results from your efforts. Thus, this is what we did by using an RFX900 USRP system daughterboard, which provided us with roughly 200mW of RF power. As aerials we used a couple of self-built $\lambda/2$ dipoles.

The rest of the transmitting and receiving system remained unchanged.
This was enough to achieve a Quasi Error Free (QEF) demodulation within an indoor environment, at a distance of several tens of metres, actually with up to six brick walls in between.
Tests in an outdoor scenario showed better performance with connectable distances of about 1.3 kilometres.
Further development activities will deal with identifying appropriate techniques that make it possible to boost Soft-DVB signal by using power amplifiers (often scarcely linear) with as small a back-off margin as possible.

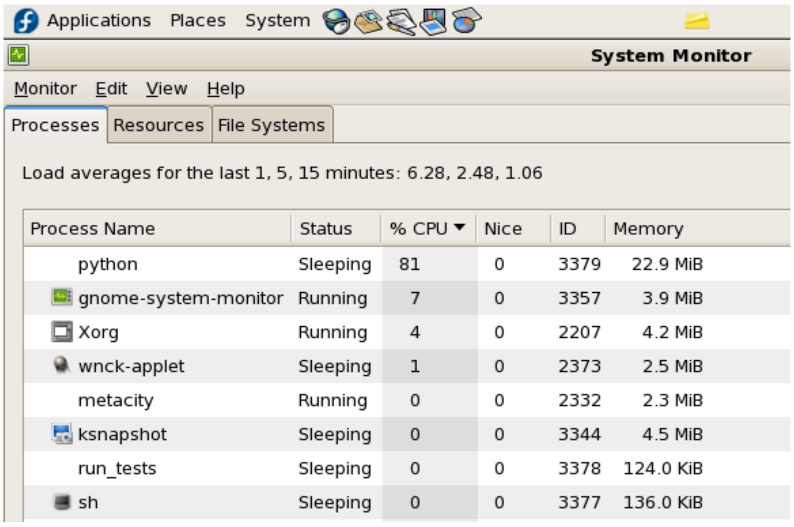
Chapter 4

Optimization

CPU-time vs Memory Trade-Off

The critical resource to achieve real time performance is the computational power of the host computer, whereas Soft-DVB is definitely *not* a memory-demanding application. Evidence of such consideration is provided in Figure 4.1 which is just a simple screenshot of Gnome system load analyzer on our Fedora Core 6 test machine. Memory usage is always below 24 MiB, central processor unit (CPU) load is instead (in the optimized, realtime version) about 75% of the overall computational power of a single-core 3.0 GHz Pentium IV processor.
Therefore, the most effective optimizations are those that trade off CPU time for memory occupancy.

A typical example is the Galois finite field (GF) multiplier involved in RS encoding, where the computation is really heavy but the set of possible result is limited — just 256 possible values in a $GF(2^8)$. In such a case, the best thing is to save computational power by avoiding the on-line calculation of GF multiplication based on the real stream of data.
What we do in Soft-DVB is to pre-calculate all the possible results of GF products within the suitable GF and store them into a look up table from which they are collected at runtime.
This is dramatically faster than calculating those products on-line and is done just once, as soon as the constructor method of the class implementing the Reed-Solomon GNURadio block is called at block's instantiation time. Using a precalculated array impacts very profitably in terms of performance gain also when implementing several other DSP function.
In fact, the same principle is applied to functions such as PRBS generation, TPS data and redundancy calculation, constellation mapping, multiple permutations in the interleaver, and also when slicing bytes into the single bits for blocks working at bit level.



Load averages for the last 1, 5, 15 minutes: 6.28, 2.48, 1.06

Process Name	Status	% CPU	Nice	ID	Memory
python	Sleeping	81	0	3379	22.9 MiB
gnome-system-monitor	Running	7	0	3357	3.9 MiB
Xorg	Running	4	0	2207	4.2 MiB
wnck-applet	Sleeping	1	0	2373	2.5 MiB
metacity	Running	0	0	2332	2.3 MiB
ksnapshot	Sleeping	0	0	3344	4.5 MiB
run_tests	Sleeping	0	0	3378	124.0 KiB
sh	Sleeping	0	0	3377	136.0 KiB

Figure 4.1: *System status measured on a Fedora Core 6 PC while a single Soft-DVB instance is running. The "python" entry displays the resources consumed by Soft-DVB.*

Actually, when two or more cascaded interleaving stages are required by the DVB-T standard, the overall resulting permutation is calculated at instantiation time and then applied at run-time. This also impacts positively on CPU time absorption.

Bit-Level Operations and the convolutional encoder example

Another major performance boost was obtained by finding suitable ways to perform multiple bit level operations just by acting on a single char.

This was actually crucial, as the char is the smallest data unit you can work upon by writing code at the pure C++ level.

An example of this technique, widely adopted within Soft-DVB code, is our convolutional encoder implementation.
Figure 4.2 shows the DVB-T convolutional encoder as described in [1].
It complies with the typical description of any convolutional encoder. Figure 4.3 shows Soft-DVB's fast convolutional implementation.

The first two blocks on the left extract single bits from the bytewise data flow that the block receives as an input. Such extraction is performed using a look-up table, in compliance with the principle described just above.
Also, at this stage, extracted bits are conveniently reassembled into a char so that the next block

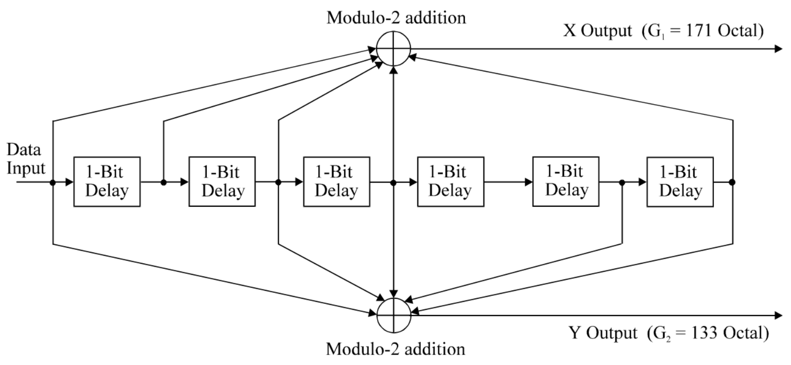


Figure 4.2: *The DVB-T convolutional encoder as described in [1]*

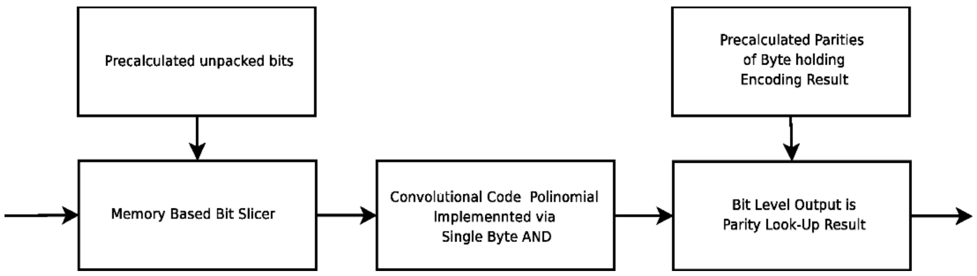

Precalculated unpacked bits
Precalculated Parities of Byte holding Encoding Result
Memory Based Bit Slicer
Convolutional Code Polinomial Implemennted via Single Byte AND
Bit Level Output is Parity Look-Up Result

Figure 4.3: *Soft-DVB's fast convolutional implementation*

can perform a single AND operation, thus processing ”one-shot” the six required bits shown in figure 4.2.

At last, the parity of the processed char, which constitutes the convolutional encoder’s output, is obtained by means of another look-up table and sent to the subsequent puncturing block that provides the chosen 2/3 coding rate.

Vectorized vs Scalar Structures

Further benefits are achieved by improving the way DSP blocks exchange data. For instance, preferring vectorized data structures to scalar structures reduces the number of input/output connections of single blocks. The vectorized data structures are assembled and disassembled within each block, without relying on external tasks, though they are present (and generally useful) in GNURadio.
Assembling and processing large vectors was found to be the best choice (even though not the only possible one) whenever big blocks of data must undergo a permutation, as it happens for example with the ”inner interleaver” DVB-T block.

As long as the GNURadio framework must take care of communication within interconnected blocks, it is self-evident that keeping the number of GNURadio blocks making up an application low has a certain impact on performance, although this is really not the first issue to take into account when it gets to optimizing the code.

Nevertheless, we found that a good trade-off between application flexibility and code clarity VS real-time performance, was to implement exactly as many GNURadio blocks as the functional blocks described in [1] are. This is shown in Table 3.1.

Chapter 5

Implementation Results

Laboratory Benchmarks

Once this thesis work produced a real-world, fully working, prototype ETSI DVB-T modulator, the entire system consistency and performance was validated at DSP for Communication Lab (DSPCOLA, see Figure 5.1), University of Pisa, where Soft-DVB proved to be capable of several hours of flawless continual transmission.

The spectrum of the output digital signal is presented in Fig. 5.2. As it can easily be verified, its shape is consistent with typical spectra of OFDM transmission schemes.

Experimental tests have been performed using two different receivers: i) an off-the-shelf digital TV set-top-box, namely the Access Media® STBL3012; and ii) a typical USB-pen DVB-T receiver, namely the Pinnacle® PCTV USB Stick 70e. The payload used to test Sof-DVB video broadcasting system is a TS obtained by recording a transmission by RAI (the Italian public broadcaster). Images from such TS are visible in Fig. 5.3.

Also the output of the Access Media STBL3012 is shown in Figure 5.3. Both receivers can easily and quickly acquire the MPEG TS synchronization byte and report a BER below 10^{-9}.

GNURadio "realtime scheduling" functionality, directly implementend via the Python script *Soft-DVB.py*, prevents Soft-DVB from being interfered by other processes. Hence, Soft-DVB can regularly feed the USRP with a continuous sample stream, ensuring real-time operation in the presence of concurrent applications.

As already stated, in its current version, Soft-DVB runs flawlessly real-time on a 3.0- GHz Intel® Pentium IV processor, averagely draining 83% of its computational power.

Further encouraging results have been obtained by testing Soft-DVB on an entry-level general purpose laptop, namely an HP® Compaq nx6310. The current version of Soft-DVB runs real-time on this HW, equipped with a 1.46- GHz Intel® Celeron M CPU.

Figure 5.1: *The entrance of DSPCOLA lab at Pisa University. Here Soft-DVB was validated*

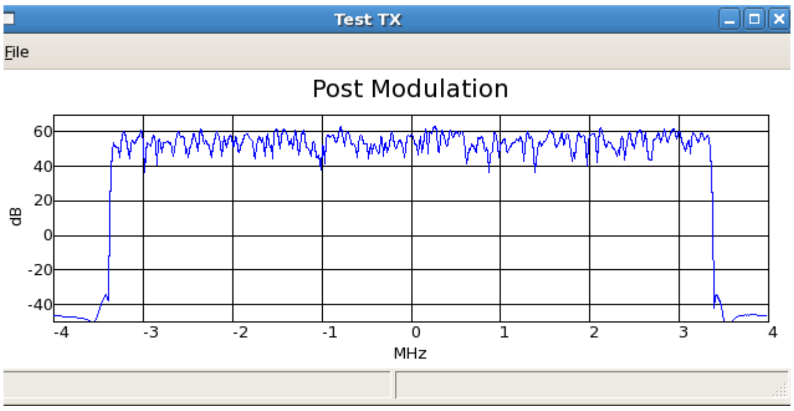


Figure 5.2: *Spectrum of the digital signal at the output of the host computer.*

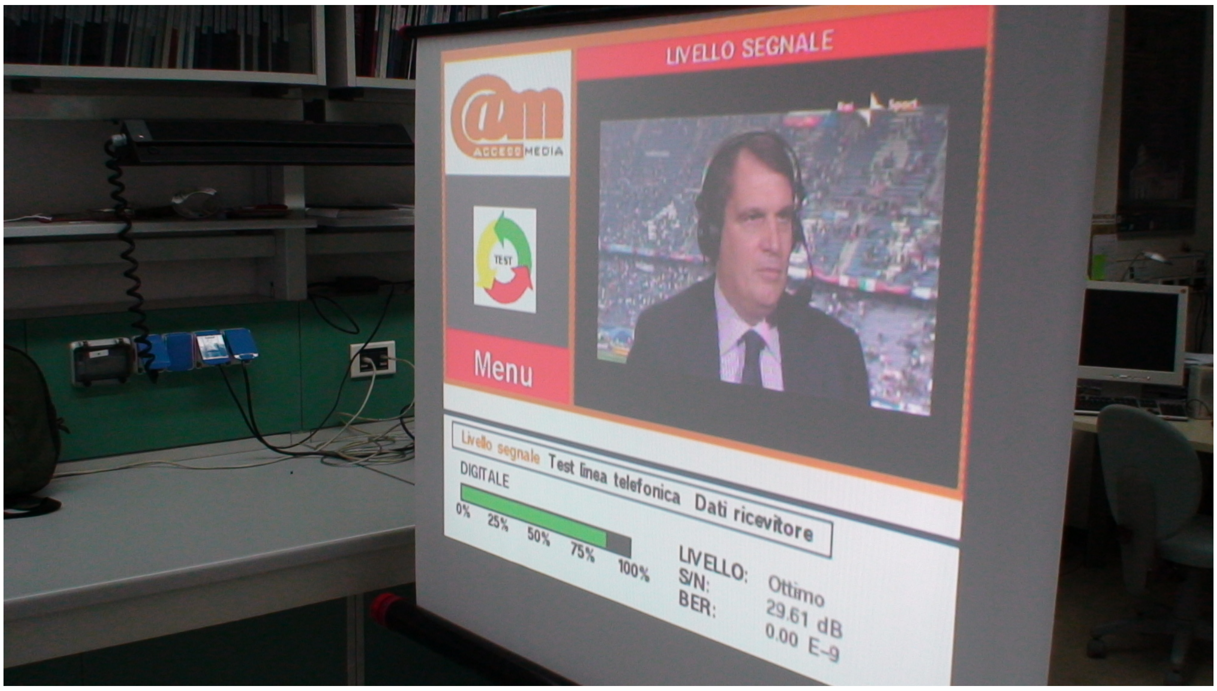


Figure 5.3: *Output of the Access Media STBL1230 test receiver fed by the Soft-DVB transmitted signal.*

Final Implementation Results

At the end of this thesis work we have finally produced a fully functional, fully software ETSI DVB-T modulator that requires only about 83% of the total computational power of a 3.0 GHz Pentium IV CPU and less than 24 MiB of RAM. The modulator can surely run continuously for several hours without showing any problem.
Also, a good number of laptops ranging from a Compaq nx6310 with a single core 1.46- GHz Intel® Celeron M CPU to a gaming level ASUS G1S proved capable of broadcasting TV in real-time using Soft-DVB and an USRP.

If the chosen RF frontend for the USRP is an RFX 900 from ETTUS Research LLC and directional receiving aerials are used, the Soft-DVB system can be used to implement a real ETSI DVB-T broadcasting station with covered distances up to a few kilometres.

Chapter 6

Real World Deployment scenarios and Perspectives

Emergency Broadcasts

One application for which Soft-DVB would be immediately useful as is (i.e. in its current 11.612 Mbps real-time version) is emergency camp broadcast.
Civil protection agencies could easily use a laptop and a USRP to broadcast emergency information in the aftermath of a disaster or when a known or supposed threat to the population is present. Viewers could watch such broadcast in their home TV sets if electricity is still present in the area.

If electrical power is not available due to the emergency situation (which is possible but does not necessarily happen in all emergencies), the broadcast can still be received by means of a DVB-T enabled laptop, a mobile, battery operated, DVB-T receiver, one of the many GPS receivers that integrate DVB-T tuner, or even a DVB-H smartphone that also support 7 MHz DVB-T channels.

The content of such broadcast could range from critical life support information (i.e. exact location of dangers, points of assistance...) to the behaviour that citizens should observe in order to minimize their risks.

Mission-Dedicated Broadcasts

Another interesting application is suitable for all agencies that have large missions operating abroad.

This is the case of armies in war or peacekeeping missions, but also of International Cooperation agencies and Non Governmental Organizations (NGOs).
Actually, also large engineering firms with many workers employed abroad could benefit from the application described in the following.

In such a scenario, a Soft-DVB based system could rebroadcast locally satellite channels arriving from homeland, therefore removing the need for many satellite dishes to be installed. Workers could see their favorite shows just on their laptop whether in a single building or in many posts scattered around a given land area.
Also, some channels within the transmitted multiplex could carry locally generated data for dedicated computer applications as well as local audio/video signals of interest for the system user.

Highway Traffic Information Systems

Currently the reports of highway traffic sensors are collected in monitoring centres and then provided to suitable websites and FM radio programs.
This is clearly not enough as, typically, just a few drivers are reached by traffic information in useful time to avoid being caught into a traffic jam.
Many car-aimed GPS navigators already integrate DVB-T.

Then why not to use a roadside network of many low power (and very low cost) software DVB-T transmitters to reach all cars with an MPEG2 transport stream featuring datacasting of all traffic info as well as entertainment audio/video channels aimed at other passengers in the car (and, obviously, not at the driver.. :-)).

The capillarity of such a network, made up of many low power, low cost transmitters, would allow to efficiently reach any single spot of the highway as well as to insert into the transmitted stream touristic and promotional information concerning the single region that each car is driving through.

Quick and Cheap Network: Ideas for a new Media Broadcasting Model

If we assume the capacity of wired IP networks to keep growing at current rates, it will probably soon be able to support both video broadcasting and VOD applications, even if IP based architectures are inherently inefficient and definitely not scalable enough for such purposes.
However, in order to keep competitive, traditional broadcasting networks should move towards innovative multimedia distribution schemes that allow user choice and customization possibilities to be added to the traditional total scalability that they can already provide.

The possibility to deploy a high capillarity, cellular multimedia distribution network that is made up of many very low cost, fully independent low power transmitters can be a great advantage for broadcasters.

In fact, the ability to set up remotely, and at any time, the exact content mix that is being broadcast by a small cell, can be useful to reach each user (or group of users) with exactly the content he/she requires and is willing to pay for. Still, using the remainder of the transmission capacity of the cell for traditional broadcasting remains a possibility.
A further advantage, compared to competitor solutions, is that the user-side equipment would remain untouched and that existing cell-phone base station sites could be used to install the low power DVB-T software transmitters.
Therefore, a mobile network operator with interests in TV broadcasting and multimedia distribution would be the perfect provider for such an application.

On Site Broadcasting

During sport events like formula one races, golf competitions and often even football matches, the interesting action and detail lie far away from the audience that is present ”on-site”. Therefore, ”On Site Broadcasting” (OSB) might be an interesting solution to improve the way such audiences experience the show. OSB mainly consists of locally transmitting video content to handheld terminals that viewers can watch when the interesting action is far away, unclear or when a slow-motion replay is needed in order to better understand what happened in a disputed situation.

As long as the broadcast is being performed on-site, good Signal to Noise Ratios (SNR) are easy to obtain, therefore 64-QAM constellation, high coding rates and short guard intervals can be used.
This yields a DVB-T channel capacity of 27.710 Mbps on a 7 MHz bandwidth which is enough to accommodate about 11 good resolution handheld MPEG2 broadcasts.

Such a wealth of channels means that many different services can be provided to the user but also that the same content can be replayed many times for a long time.
This is highly useful as long as the user won’t need to keep watching the service all the time but will just have the opportunity to switch on his receiver when he wants to see something and he will surely find the content he’s looking for on one of the channels being continuously broadcast.

Of course, one of the provided channels can still show a standard, continuous coverage of the event.

Another good application scenario for OSB is local broadcast of video content to DVB-T/H enabled smartphones, laptops, GPS receivers and other devices in business class transports such as high speed trains or intercontinental flights.

Parallel CPU Architectures: a huge computing power boost

As to the perspectives of further development it is worth to take into account that, in the last few months, developers within the GNURadio project (particularly Eric Blossom) have been working on a new parallel architecture of this framework. Such architecture should provide software radio programmers with the tools needed to access the huge raw computational power of the Cell BE processor [10]. This architecture is expected to be part of GNURadio releases planned for the near future. On the Cell BE platform, the accessible computing power will be approximately $3 \cdot 10^{10}$ floating point operations per second (flops), which is approximately 30 times the computing power of the machine used to test the current version of Soft-DVB. Thus, assuming the number of flops as a proper metric for the system computational capability, some hypotheses about the potentiality of a Cell BE-based Soft-DVB system can be drawn.

Considering the current Soft-DVB computational needs to be fully parallelized and distributed among the various processing cores of the Cell BE, a bunch of USRPs is supposed to allow about 30 11.612-Mb/s DVB-T multiplexes to be broadcast. This solution yields a total throughput of 348.36 Mb/s, which is suitable for broadcasting about 70 standard definition MPEG2 channels. Higher throughputs are achievable by upscaling the sampling frequency to that required for a 8- MHz channel and by using different constellations, coding rates, guard intervals. In terms of commercial applications, a large-sized HW transmission facility can be replaced by a high computing power platform (such as the Cell BE) running multiple Soft-DVB instances at full DVB-T channel capacity.

Actually the CELL-BE processor features 8 3.2GHz, 128bit RISC CPUs designed for SIMD operation called Sinergistic Processing Elemnents(SPEs).
The usage of such a heavily parallel computing platform can really provide Soft-DVB, as well as any other software radio system, with an enormous performance boost.

It is quite likely that, by developing new algorithms that are wise enough to use up the huge raw processing power of parallel CPU architectures, it will be possible to make software radio systems performance-competitive with their hardware counterparts.
If this will prove to be true as we believe it is, then all implementations of RF communication systems will be software radio implementations. This would happen because of the unachievable flexibility, ease of implementation, and great affordability of such a technology.

We do believe that Soft-DVB is a first, significant step in this direction.

Chapter 7

Conclusions

In this master thesis work, a fully-software, low-cost, real-time DVB-T modulator named Soft-DVB was developed using the GNURadio hybrid C++/Python framework.

The developed code can turn any average, consumer level PC into a simple but reliable ESTI DVB-T broadcasting station.

Results obtained during this work show that a full software solution for ESTI DVB-T signal generation is not just feasible, but also shows high flexibility, good performance and ease of implementation.

Such results have been demoed at the International SDR conference (WSR08) that took palce in Karlsruhe, Germany in March 2008.

Actually, Soft-DVB is intended to be a starting point in a research aiming to develop a full new class of multimedia distribution systems, entirely based on software defined radio technology. The research activity based upon Soft-DVB is also aimed at studying and exploring new possible application scenarios for such systems, taking the highest possible advantage from the peculiarities of the software radio paradigm.

We believe that Soft-DVB achievements are an early proof of both the feasibility and the profitability of all this.

Parallel re-interpretation of existing radio signal processing algorithms has proved to be the key strategy in the effort to make SDR systems competitive and preferable to their hardware counterparts.

Finally, it is possible to state that software radio technology offers unprecedented opportunities for multimedia broadcasting and datacasting applications, both on small and on large scale distribution networks.

Further research effort is needed especially in the area of parallel algorithm design.

Nevertheless, players holding as their exclusive the know how needed to deploy fully software-radio broadcasting networks would have a dramatic competitive advantage over others.
This would be true not just because of the strong cost reduction but also as long as entirely new classes of applications would then become possible.
Moreover, the total flexibility of an SDR system would yield for such players a highly significant protection level for the investment they make in their broadcasting infrastructure.
An SDR-based broadcasting network would in fact allow its owner to constantly, smoothly evolve and upgrade the service provided to the end user without any need to be replaced. Changing user requirements and new business ideas could therefore be addressed with minimal effort.

Acknowledgments

I wish to thank the DSPCOLA staff — especially Professor Marco Luise and Giacomo Bacci, PhD — and all the guys on the GNURadio mailing list for the precious help and support received while developing Soft-DVB.
A special thanks also goes to Gordon Cichon, whose check dumps were very precious in the early validation process of Soft-DVB.

List of Figures

List of Tables

List of Acronyms

DVB-T	Digital Video Broadcasting - Terrestrial version
HDTV	High Definition TeleVision
MIMO	Multiple Input Multiple Output
LDPC	Low Density Parity Check
HW	Hardware
SW	Software
ETSI	European Telecommunication Standards Institute
RF	Radio Frequency
GNU	GNU's Not Unix (Recursive Acronym) :-)
FM	Frequency Modulation
IP	Internet Protocol
SDR	Software Defined Radio
DSP	Digital Signal Processing
VOD	Video On Demand
UHF	Ultra High Frequency
DAB	Digital Audio Broadcasting
MPEG2	Motion Picture Experts Group2
TS	Transport Stream
SFN	Single Frequency Network
PRBS	Pseudo Random Binary Sequence
LFSR	Linear Feedback Shift Register
RS	Reed-Solomon (Codes)
LOS	Line of Sight
NO-LOS	No Line of Sight
ATSC	Advanced Television Systems Committee
FFT	Fast Fourier Transform
IFFT	Inverse Fast Fourier Transform
DAC	Digital to Analogue Converter
ADC	Analogue to Digital Converter

DMT	Discrete MultiTone
DRM	Digital Radio Mondiale
T-DMB	Terrestrial Digital Multimedia Broadcasting
xDSL	x Digital Subscriber Line
COFDM	Coded OFDM
BER	Bit Error Rate
AWGN	Additive White Gaussian Noise
USRP	Universal Software Radio Peripheral
TPS	Transmission Parameters Signalling
BCH	Bose Chaudhuri Hocquenghem
CPU	Central Processor Unit
GF	Galois Field
DSPCOLA	DSP for Communication Lab
NGO	Non Governmental Organization
SNR	Signal to Noise Ratio
SPE	Sinergistic Processing Elemnent
QEF	Quasi Error Free

Bibliography

[1] *Digital Video Broadcasting (DVB); Framing Structure, channel coding and modulation for digital terrestrial television*, ETSI Std. EN 300 744 V1.5.1, Nov. 2004.

[2] GNURadio website. [Online]. Available: http://gnuradio.org/trac

[3] *Digital Video Broadcasting (DVB); DVB specification for data broadcasting*, ETSI Std. EN 301 192 V1.4.1, June 2004.

[4] DVB Project, "Official DVB-T deployment data," 2007. [Online]. Available: http://www.dvb.org/dvb-deployment-data.xls

[5] P. M. Luise, "Introduzione al dvb e all'ofdm," 2001.

[6] M. Ettus, "TX and RX daughterboards for the USRP radio system," 2006. [Online]. Available: http://www.ettus.com/downloads/miscdboards-v3b.pdf

[7] Ettus Research LLC website. [Online]. Available: http://www.ettus.com

[8] M. Frigo and S. G. Johnson, "FFTW: An adaptive software architecture for the FFT," in *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP)*, Seattle, WA, May 1998, pp. 1381–1384.

[9] Radio Network Processor website. [Online]. Available: http://www.radionetworkprocessor.com

[10] S. Williams, J. Shalf, L. Oliker, P. Husbands, S. Kamil, and K. Yelick, "The potential of the cell processor for scientific computing," in *Proc. ACM Conf. on Computing Frontiers*, Ischia, Italy, May 2006, pp. 9–20.