The Exploration of the Software-Defined Radio Concept by Prototyping Transmitter and Receiver Functions on a Digital Signal Processor

Roel Schiphorst, Sabih H. Gerez and Cornelis H. Slump

Department of Electrical Engineering, University of Twente, The Netherlands
E-mail: r.schiphorst@el.utwente.nl, s.h.gerez@el.utwente.nl, c.h.slump@el.utwente.nl

Abstract— **An ideal "software radio" is a system that performs analog-to-digital conversion directly after the antenna and then does all signal processing required in the digital domain on a platform that supports reconfiguration. "Software-defined radio" (SDR) is the term used for a more realistic approach in which part of the processing is still done in the analog domain.**

The use of mobile telephony has shown a spectacular growth in the last 10 years. A side effect of this rapid growth is an excess of mobile system standards. Therefore, the SDR concept is emerging as a potential pragmatic solution. It aims to build flexible radio systems, which are multiple-service, multi-standard, multi-band, re-configurable and re-programmable, by software.

First, this paper presents a global overview of SDR. Furthermore, it explains the implementation of an SDR transmitter and receiver that have been simplified for the purpose of illustration. The source code has been written in C and is running on a DSP evaluation module of Texas Instruments. The algorithms are based on a modified version of the China Wireless Telecommunication Standard (CWTS). The correctness of the implemented system has been verified and measurements of the *bit error rate* **versus the bit-energy-to-noise-energy ratio are reported.**

I. Introduction

Since the early 1980's the use of cellular mobile systems have grown enormously. Nowadays, mobile communication has become a major worldwide business. A side effect of this rapid growth is an excess of analog and digital mobile system standards such as TACS, GSM, DCS-1800, IS-95 CDMA, etc. In fact, every major country has its own standard(s). Efforts to define a unique worldwide standard result often in a new, extra standard. The excess of standards is not only bad for manufacturers but also for consumers. Manufacturers have to develop a new telephone for each standard. This results in extra development costs and small divided markets. It is also bad for consumers because they cannot use their mobile telephones abroad.

A unique common worldwide standard has benefits, but the industrial competition between Asians, Europeans and Americans makes it very difficult. It is for this reason that the *software-radio concept* is emerging as a potential pragmatic solution: a software implementation of the user terminal able to dynamically adapt to the radio environment in which the terminal is located [1]. Aside of the standardization issues, one should also view the software radio concept as a means to make users, service providers, and manufacturers more independent of standards. The benefits of this approach are that air interfaces may, in principle, especially be tailored to the specific needs of a particular service for a particular user in a given environment at a given time. For a manufacturer, a single design is sufficient for the whole world and consumers can use their telephones in every country.

Given the analog nature of the air interface, a software radio will always have an analog frontend. In an ideal software radio, the analog-to-digital and digital-to-analog (A/D/A) converters should be positioned directly behind the antenna. Such an implementation is infeasible due to the power that such device would consume and other physical limitations [2, 3]. It is therefore a challenge to design a system that preserves most properties of the ideal software radio while being realizable with current-day technology. Such a system is called a *software-defined radio* (SDR). The first serious attempts to build an SDR were made in the context of military applications (see e.g. [4]). In the last few years, a strong interest in the civil application of SDR has grown. This becomes clear from the long list of commercial companies that have joined efforts to cooperate on the standardization of such systems in the non-profit organization *SDR Forum* [5]. In addition, several consortia are actively involved in research in the field. An example is the *SORT* project financed by the European Union *Advanced Communications Technology and Services* (ACTS) programme [6].

An SDR will have one or more of the following properties [7]:

- a flexible transceiver architecture that can be controlled and programmed by software;
- radio functions that are mainly computed by digital signal processing;
- reprogrammability (the possibility to download new software) through the air interface;
- support of multiple modes and standards.

The reprogrammability can be used for:

- the frequency band and the channel bandwidth;
- the modulation and coding scheme;
- the radio resource and mobility management proto-

cols;

- user applications.

A flexible transceiver architecture also opens the possibility for more sophisticated signal processing than in rigid hardware-based solutions. The transmitter can characterize the transmission channels and adapt its power, modulation scheme, etc. to the circumstances. The receiver can also apply a whole range of techniques (detect energy distribution in the channel and adjacent channels, null interference, deal with multi-path, correct errors) to receive the signal with the lowest possible *bit error rate* (BER) [1].

In order to achieve the required flexibility, the architecture should consist of *reprogrammable* and *reconfigurable* components. As far as reprogrammable components are concerned, one can observe that the performance of *digital signal processors* (DSPs) has increased significantly in recent years [8]. Tasks that required multiple DSPs functioning in parallel a few years ago can nowadays easily be handled by a single DSP.

Reconfigurability is provided by *field-programmable gate arrays* (FPGAs). Technological developments also have boosted the performance provided by these devices which makes their use in SDR attractive [9, 10]. They consume less power than DSPs, they have the advantage that variable word lengths can be used throughout an algorithm, etc. *Dynamically reconfigurable* FPGAs exist that allow the device to be reconfigured on-the-fly: while part of the device is busy doing some signal processing, another part can receive a new functionality targeted to a next task. While it sounds positive that FPGAs “put the silicon back under the control of the DSP system architect” [10], design tools that are much more powerful than currently available, are necessary to take advantage of the many degrees of freedom. Yet another disadvantage of FPGAs is that they consume far more power than equivalent ASICs. ASICs that have some degree of reconfigurability are therefore certainly an option in SDR.

A recent of review of the challenges involved in building an SDR was given by Mitola [11]. The most important issues are:

- Defining a suitable layered architecture. At the lowest level, one finds the hardware platform consisting of DSPs, FPGAs and ASICs. The next layer implements radio primitives such as voice and data channels, control threads, etc. Higher levels provide the selection of the radio standard to be used and user services.
- Making portable unit. Low power consumption is the key issue here. Power savings are possible from design considerations at the circuit level up to the software level [12]. In portable terminals, important savings can be achieved by *dynamic* power management [13], i.e. by making use of different degrees of *sleep modes.* Receiver complexity is typically four or more times the transmitter complexity. Thus, the receiver architecture has a first order impact on handset cost.

This paper reports on the DSP implementation of radio functions in order to gain some experience on SDR complexity. The *China Wireless Telecommunication Standard* (CWTS) was chosen for this purpose because its specification is freely available on the Internet [14]. In the rest of this paper, first the standard will be shortly introduced and the parts that were chosen for a DSP implementation will be presented. Next, some attention to the implementation itself will be given and experimental results will be presented.

II. The China Wireless Telecommunication Standard

A wireless telecommunication standard is very complex; apart from the physical layer, there are many other aspects, such as the billing system, service channels, etc. This section only describes the physical layer of the communication between base station and user terminal of the CWTS standard. The physical layer consists of the following parts: multiple access, frame structure, modulation, and filter characteristics.

The CWTS standard uses a combination of *code-division multiple access* (CDMA) and *time-division multiple access* (TDMA). The used CDMA component is direct-sequence code division multiple access (DS-CDMA). In DS-CDMA systems [15], the spreading code is a sequence of bits (known as chips). First an XOR operation is carried out between the message and the spreading code. This XOR operation is also known as chipping. So a `'0'` is represented by a chip sequence and a `'1'` is represented by the inverse of this chip sequence. Instead of transmitting the message bits the accompanying chip sequences are transmitted.

The chip rate of the DS-CDMA technique in CWTS is equal to 1.28 Mchips/s which results in a bandwidth of approximately 1.6 MHz. Furthermore, the system uses a 200-kHz carrier raster. So, eight channels utilize the same 200-kHz band. Apart from DS-CDMA, CWTS uses a TDMA component, namely TDD (time-division duplex). This allows to use the same radio frequency for both the forward and the reverse link transmissions by reserving separate time intervals for both links.

The CWTS standard uses a four-layer structure. The first level consists of superframes, which contain 72 radio frames of 10 ms. Radio frames are divided into two 5-ms subframes. In each subframe, there are 7 main time slots and 3 special time slots. The complete physical channel signal format is presented in Figure 1. The 7 main time slots can be used for down-link communication (DL#n) or up-link communication (UL#m). Between the down-link and up-link communication there are 3 special time slots. The first one is the *down-link pilot symbol* (DwPTS) that is used for down-link synchronization. The next time slot is a *guard period* (GP) used to separate down and up-link communication. The last special time slot is *up-link pilot symbol* (UpPTS) which is used for up-link synchronization.

The CWTS standard uses the *quadrature phase-shift keying* (QPSK) modulation technique for transmission. QPSK uses two channels for transmission, an in-phase (I) and quadrature phase (Q) channel. The I channel is multiplied with a cosine and the Q channel with a sine. Because a sine is orthogonal to a cosine, both carriers can use the

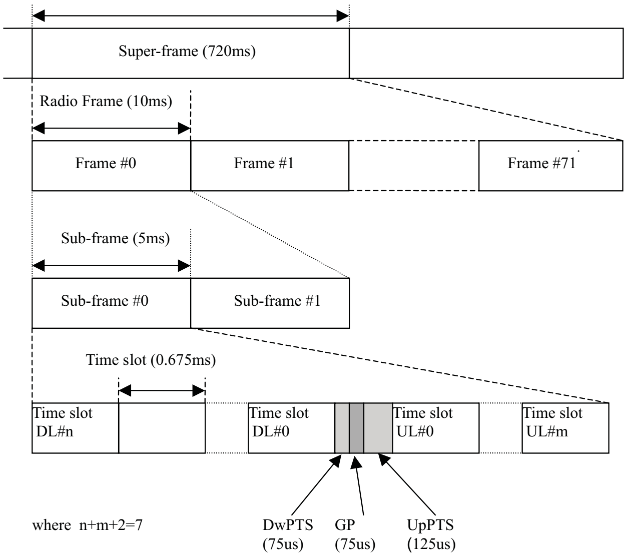

Fig. 1. The physical channel structure in CWTS.

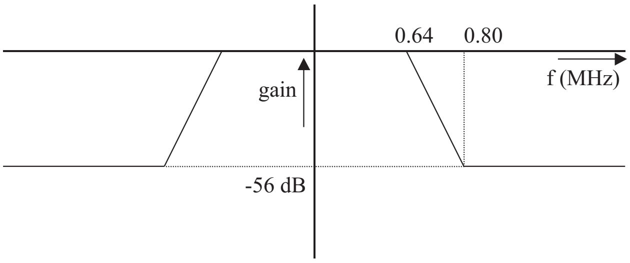

Fig. 2. The CWTS receiver filter characteristics.

same frequency. Data is sent by changing the phase of the carriers, A binary `'1'` leads to a phase shift of 180 degrees, whereas a binary `'0'` leads to no phase shift. The modulated signal is the multiplication of the carriers with the data signals. To reduce *inter-symbol interference*, the output of the QPSK modulation is led through a *root-raised-cosine filter* with a roll-off factor α of 0.22.

The input signal at the receiver can have a large input range. The minimal (reference) sensitivity is -135 dB when the data rate is 12.2 kbps. An input signal at the reference sensitivity level should not have a BER larger than 0.001. The maximal input level on the other hand cannot be described by an absolute value. It depends on the BER which should not be larger than 0.001. The filter characteristics of the receiver cannot be obtained directly from the CWTS standard. However, the standard presents several definitions and requirements from which the filter characteristics can be estimated. The estimations lead to a filter with a pass band of 0.64 MHz, a transition band of 0.16 Mhz and a minimal stop band attenuation of 56 dB [16]. The filter characteristics are illustrated in Figure 2.

III. DSP Implementation

Two EVM320F549 DSP-evaluation modules from *Spectrum Digital* were chosen as the reconfigurable platform on which radio functions of the CWTS standard were going to be prototyped. One was meant for the receiver and the other for the transmitter. The main components of the evaluation module are the TMS320LC549 DSP, and the TLC320AD55 A/D/A chip, both from *Texas Instruments* (TI), as well as external memory for the DSP. Detailed technical specifications of the module can be obtained from [17]. The software for the DSPs was developed in the TI *Code Composer* environment [18]. For this project, at some occasions, functions from the TI TMS320C54x DSPLIB were used. It is an optimized DSP function library for TMS320C54x processors. It contains more than 50 assembly-optimized general-purpose signal-processing routines that can be called from C. DSPLIB is freely available [18].

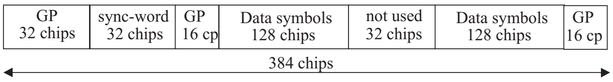

Fig. 3. The frame structure used for the prototype implementation.

The TMS320LC549 is a fixed-point DSP based on a *Harvard* architecture: instruction and data memories can be addressed simultaneously. A highly parallel data path containing a multiplier, an adder, an ALU and a barrel shifter, also contributes to the high performance of the DSP (100 MIPS) [18].

The TLC320AD55 is a high-resolution low-speed A/D/A converter. The maximum attainable sample frequency is 64 kHz (for more details, consult [18]). The software radio cannot utilize this maximum sample frequency because of the time requirements of the interrupt routine for the A/D converter of the software radio receiver. The receiver must detect a frame burst, which is carried out during this interrupt. If the interrupt routine is not ready when the next sample arrives, data is lost. Therefore the sample frequency is limited to 16 kHz.

Another limitation of the evaluation modules is the size of the data memory (64 kb). This size is too small for a direct implementation of the CWTS standard. Therefore a derived frame structure has been used, as shown in Figure 3. This frame structure contains only one sync word and one time slot. The sync word is chosen arbitrarily and is not based on the gold-code set defined by the CWTS standard.

The implemented transmitter has the structure shown in Figure 4. The `initialization` routine takes care of setting up the DSP and the A/D/A chips and their mutual communication that is based on interrupts. This is the only routine that was written in assembler rather than C. The `build frame` routine builds a frame according to the structure of Figure 3. The routines `QPSK modulation`, `CDMA` and `root-raised-cosine filter` perform exactly what their names indicate. The routine `interpolation` is necessary because the carrier frequency has a much higher sample rate than the symbol rate of the frame. Therefore, the I and Q channel are interpolated with a factor 16 in this block to match with the sample rate of the carrier frequency.

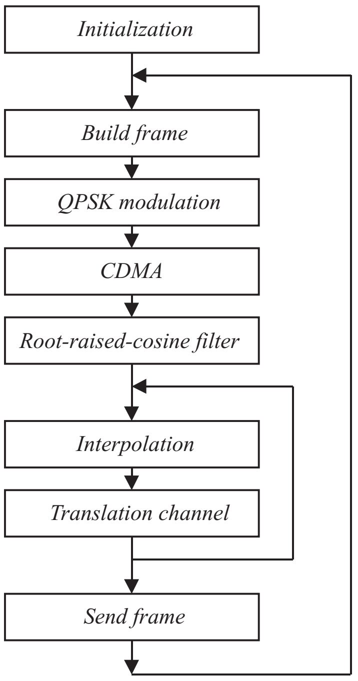

Fig. 4. Block diagram of the implemented transmitter.

The routine `translation channel` transforms the QPSK signal to a channel of the system band. So the signal is transformed to a higher frequency. The interpolated part of the I channel is multiplied with a cosine and the Q channel with a sine. First the I channel is multiplied with values from a cosine table and the result is stored in an output table. Then, the Q channel is interpolated and multiplied with values from a sine table. The result is added to the output table and transmitted to the A/D/A chip by the routine `send frame`.

The receiver block diagram shown in Figure 5 contains many routines that are the inverse of the transmitter and some other routines as well. The routine `detect frame` inspects the average signal level and uses this value to establish whether noise or a frame is being received. Then, an entire frame is stored by `store frame` and a zero crossing is computed by `detect phase`. The phase is used in the routine `translate channel` that multiplies the signal with a cosine and sine to recover the I channel and Q channel respectively.

The routine `decimation` uses multiple stages. A multi-stage filter requires less computational power than a one-stage filter [19]. The receiver uses special filters in the first stages of the decimation (filtering) process. These special filters are *cascaded integrated comb* (CIC) filters. More information about CIC filters can be found in [20, 21]. CIC filters can only be used in the first steps of the decimation process because the frequency response is bad. An example is shown in Figure 6. CIC filters require only additions and subtractions as opposed to common FIR filters that require multiplications as well. This often leads to cheaper realizations. An example of a CIC filter is shown in Figure 7.

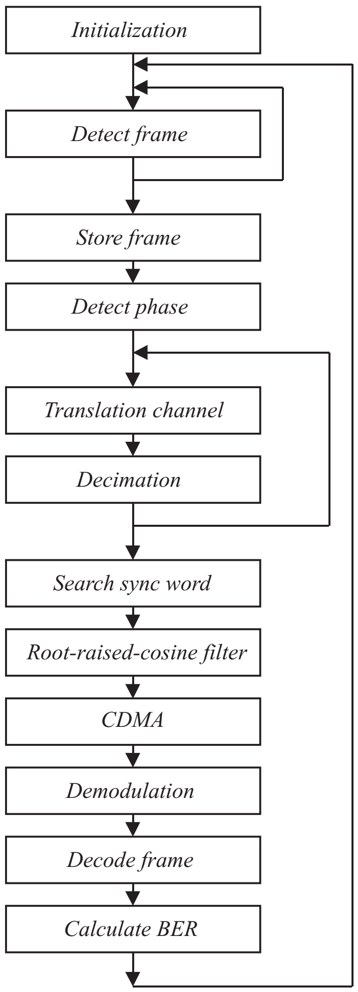

Fig. 5. The block diagram of the implemented receiver.

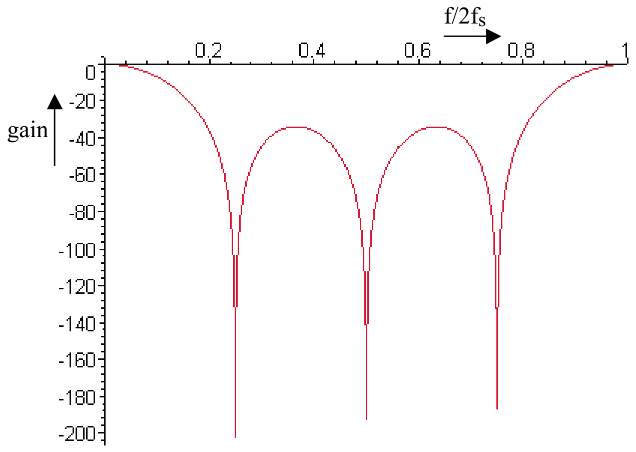

Fig. 6. The frequency response of a CIC filter.

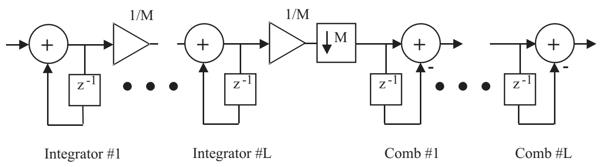

Fig. 7. The structure of a CIC filter.

The receiver routines that follow, `search sync word`, `root-raised-cosine filter`, `CDMA` and `demodulation` are quite straightforward and result in the routine `decode frame` delivering the data bits that were received. The quality of the transmission process is evaluated by the routine `calculate BER`. A more detailed account of the implementation of the transmitter and receiver routines can be found in [16].

IV. Experimental Results

The SDR functions as described in the previous section have been implemented. The implementation was verified to function correctly both by a direct interfacing of the transmitter and receiver as well as by using an air interface. For the former, the analog outputs of the evaluation boards were connected by a cable. For the latter, a 433 MHz FM transmitter-receiver combination from Radiometrix [22] was used.

The DSP evaluation module is capable of transmitting 0.73 frames per second. If the serial-port receive interrupt is disabled, such that no transmission takes place, the DSP is capable of calculating 1.33 frames per second. Given the fact that the DSP can process 100 million instruction cycles per second, about 75 million instructions are needed to compute one frame. The figures for the receiver are 0.33 respectively 1.27 frames per second leading to the conclusion that about 80 million instructions are necessary to decode one frame.

The BER of the SDR implementation has been measured with different levels of noise. There are different ways to introduce noise in the system. In this experiment, noise was added in the transmitter by software. Only noise is added to the data part of the frame. In this way the frame detection is unaffected by noise and only bit errors are counted which are caused by noise and not by wrong frame synchronization. Furthermore, the transmitter and receiver were interconnected directly by a cable. In this way, all noise other than the software-generated noise is eliminated.

According to theory, the BER is a function of the *bit energy* E_b and *noise energy* N_0. It is given by [23]:

$$BER = Q\left(\sqrt{2\frac{E_b}{N_0}}\right)$$

where:

$$Q(z) = \frac{1}{\sqrt{2\pi}}\int_z^\infty e^{-\frac{\lambda^2}{2}}d\lambda$$

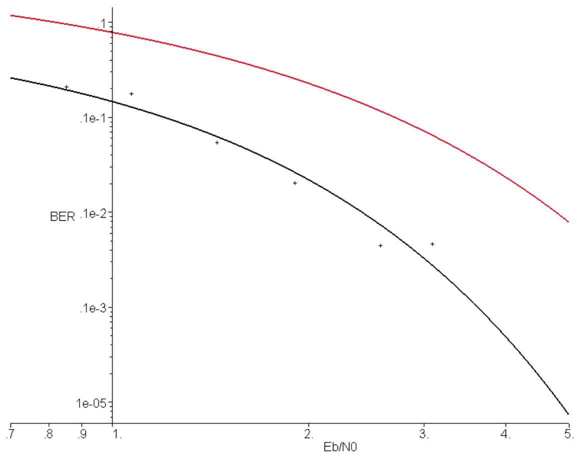

Fig. 8. Theoretical and measured behavior of the BER.

The theoretical curve following from these formulae and the results of the measurements performed are displayed in the logarithmic plot of Figure 8. Along the horizontal axis, $\frac{E_b}{N_0}$ is expressed in dB. To show the tendency of the measured results, a best-fit curve of the form $y = ae^{bx}$ is also included on the plot.

The theoretical and measured BER curves are different. They have almost the same shape, but the measured BER curve is much lower. The theoretical formula assumes a QPSK signal surrounded by white noise. The measured BER curve, on the other hand, has not been measured with white noise, because the noise has been generated by software. If noise is generated by software, only noise is added in the frequency band from zero to half the sample rate while white noise has equal energy in all frequencies. Furthermore, the sample rate of the software-radio is relatively low making that the generated noise cannot be considered as an approximation of white noise. In addition, the receiver uses low-pass filters to extract the QPSK signal and only noise in the pass band remains: the low-pass filters have eliminated other noise.

Other methods for generating noise are difficult to implement. For example a noise generator which adds noise at the analog output of the software-radio transmitter also affects the synchronization word and thus the synchronization. Because synchronization errors do not count for the BER, the BER would become difficult to measure.

V. Conclusions

This paper has reported on important issues for the design of SDR and the exploration of this concept by implementing characteristic transmitter and receiver functions of the CWTS standard. The prototype operates at a speed that is about 5000 times slower than required by the standard. The functions were implemented on evaluation modules that are meant for audio applications rather than SDR. It is postulated that the entire physical-layer CWTS functionality can be implemented on a reconfigurable platform with a few DSPs, provided that:

- newer generation DSPs are used;
- a platform tailored for SDR is developed;

- the platform has FPGAs for the implementation of the most critical functions;
- the platform has sufficient memory;
- more effort is spent in optimizing the generated DSP code (neglected in current implementation).

References

[1] J. Mitola, "The software radio architecture," *IEEE Communications Magazine*, pp. 26–38, May 1995.

[2] B. Nauta and C.H. Slump, "On the design of a front-end subsystem for software-radio applications," in *PROGRESS 2000 Workshop on Embedded Systems*, Utrecht, The Netherlands, October 2000.

[3] R.H. Walden, "Performance trends for analog-to-digital converters," *IEEE Communications Magazine*, pp. 96–101, February 1999.

[4] R.J. Lackey and D.W. Upmal, "Speakeasy: The military software radio," *IEEE Communications Magazine*, pp. 56–61, May 1995.

[5] "Software defined radio forum, a forum on open architecture reconfigurable wireless technology," `http://www.sdrforum.org/`.

[6] "Ac315: Sort – software radio technology," `http://www.ifn.et.tu-dresden.de/~sort/welcome.html`.

[7] E. Buracchini, "SORT and SWradio concept," in *4th ACTS Mobile Communications Summit (AMOS)*, Sorrento, Italy, 1999.

[8] D. Efsthatiou, J. Fridman, and Z. Zvonar, "Recent developments in enabling technologies in software defined radio," *IEEE Communications Magazine*, pp. 112–117, August 1999.

[9] M. Cummings and S. Haruyama, "FPGA in the software radio," *IEEE Communications Magazine*, pp. 108–112, February 1999.

[10] C. Dick and F.J. Harris, "Configurable logic for digital communications: Some signal processing perspectives," *IEEE Communications Magazine*, pp. 107–111, August 1999.

[11] J. Mitola, "Technical challenges in the globalization of software radio," *IEEE Communications Magazine*, pp. 84–89, February 1999.

[12] A.P. Chandrakasan and R.W. Brodersen, *Low Power Digital CMOS Design*, Kluwer Academic Publishers, Boston, 1995.

[13] L. Benini and G. De Micheli, *Dynamic Power Management, Design Techniques and CAD Tools*, Kluwer Academic Publishers, Boston, 1998.

[14] "China wireless telecommunication standard," `http://www.cwts.org/`.

[15] S.M. Schwartz, "Frequency hopping spread spectrum (FHSS) vs. direct sequence spread spectrum (DSSS) in the IEEE 802.11 wireless local area network arena," `http://www.breezecom.com/TechSupport/fhvsds.htm`, 1997.

[16] R. Schiphorst, "Demonstration of the software-radio concept," M.S. thesis, University of Twente, Department of Electrical Engineering, June 2000, EL-S&S-005.00.

[17] "Spectrum digital," `http://www.spectrumdigital.com/`.

[18] "Texas instruments," `http://www.ti.com/`.

[19] R.A. Roberts and C.T. Mullis, *Digital Signal Processing*, Addison-Wesley, Reading, Massachusetts, 1987.

[20] M.E. Frerking, *Digital Signal Processing in Communication Systems*, Kluwer Academic Publishers, Boston, 1994.

[21] A.Y. Kwentus, Z. Jiang, and A.N. Wilson, "Application of filter sharpening to cascaded integrator-comb decimation filters," *IEEE Transactions on Signal Processing*, vol. 45, no. 2, pp. 457–467, February 1997.

[22] "Radiometrix," `http://www.radiometrix.co.uk/`.

[23] L.W. Couch, *Modern Communication Systems: Principles and Applications*, Prentice Hall, Englewood Cliffs, New Jersey, 1995.