

A ROBUST SOFTWARE RADIO TESTSET FOR RESEARCH AND
LABORATORY INSTRUCTION

Ratha An

A Thesis
Submitted to
the Graduate Faculty of
Auburn University
in Partial Fulfillment of the
Requirements for the
Degree of
Master of Science

Auburn, Alabama
December 12th, 2005

A ROBUST SOFTWARE RADIO TESTSET FOR RESEARCH AND

LABORATORY INSTRUCTION

Except where reference is made to the work of others, the work described in this thesis is my own or was done in collaboration with my advisory committee.

Ratha An

Certificate of Approval:

Thomas Denney
Associate Professor
Electrical and Computer Engineering

Thaddeus Roppel
Associate Professor
Electrical and Computer Engineering

A. Scottedward Hodel
Associate Professor
Electrical and Computer Engineering

Stephen L. McFarland
Acting Dean, Graduate School

A ROBUST SOFTWARE RADIO TEST SET FOR RESEARCH AND

LABORATORY INSTRUCTION

Ratha An

Permission is granted to Auburn University to make copies of this thesis at its discretion, upon request of individuals or institutions and at their expense. The author reserves all publication rights.

Signature of Author

Date of Graduation

VITA

Ratha An, the oldest daughter of Chhean An and Ny An, was born on December 12, 1973 in Cambodia. The family suffered during the Pol Pot regime in 1975 –1979 and became refugees in 1989 after living through hardship and lack of human rights in their homeland. Their escape led them to Site II camp, Thailand, and finally to the United States in 1993.

Ratha graduated from Pensacola High School in Pensacola, Florida in 1996. She received dual degrees of Bachelor of Electrical Engineering and Bachelor of Computer Engineering from The University of Florida in 2001 with honors, and continued on to graduate school to further her education in wireless systems.

THESIS ABSTRACT

A ROBUST SOFTWARE RADIO TEST SET FOR RESEARCH AND LABORATORY INSTRUCTION

Ratha An

Master of Science, November 7th, 2005
(B.E.E., University of Florida, Spring 2001)

84 Typed Pages

Directed by Thaddeus A. Roppel

Since the mid-1990s, communication technology has advanced dramatically, making it possible to achieve high data rates for both enterprise and individual users. The rapid and widespread deployment of wireless communications systems employing a variety of regional standards has led to the software radio (SWR) concept. The essential idea of SWR is to incorporate the entire functionality of a wireless transceiver into software running on a digital signal processor (DSP) or microprocessor, resulting in devices that are more compact, power-efficient, and readily adaptable to new standards and environments. In reality, the state of current technology is such that this goal can be only partially realized. It is still necessary to implement some of the RF functionality in hardware due to speed limitations of existing DSP's. This intermediate stage of development is referred to as software-defined radio (SDR). This thesis will focus on a robust, flexible SDR design suitable for use in undergraduate laboratory experiments, as well as in graduate research. The design presented here uses a Motorola 68HC11 microcontroller for baseband functions and a Cirronet WIT2410 RF front-end module. The 68HC11 is studied by most students prior to taking a wireless laboratory course at Auburn University, and is therefore a "known" quantity with reduced learning curve. The WIT2410 is a well documented, field-proven RF front-end module providing considerable flexibility to the user. Importantly, it is available in a fairly rugged package which provides a large degree of immunity to electrostatic discharge (ESD). The proposed system design obtained by coupling the 68HC11 with the Cirronet WIT2410, is presented, as well as a sample experiment that can be run in an undergraduate laboratory setting.

ACKNOWLEDGEMENTS

To Mom and Dad as always. I would like to begin by thanking to my parents Chhean and Ny, and my siblings for loving and believing in me, which are the core elements in life that lead to accomplish my goal. My sincere thanks to my uncle Chan and my aunt Than, who had sponsored my family to United States, that made education is possible to me. My second thank go to American people and Government that have granted permissions for refugees begin their new life here.

First and foremost I would like to thank Professor Thaddeus Roppel, whose insight, guidance and advice have always been the cornerstones of my studies and research. His character and integrity is what motivated me to work harder. I would also like to thank Dr. Richard C. Jaeger for supporting the software radio project. Also, special thanks to Professor A. Scottedward Hodel and Professor Thomas Denney for agreeing to be on my graduate committee and reviewing this thesis.

I also would like to acknowledge the wonderful persons Mr. Joe Haggerty and Miss Linda Barresi for advice and help on electronics equipment. Finally, I would like to thank Mr. Bill Weber at UWF who allowed me to access lab equipments for circuits testing. His consistent help has always been the greatest incentive for me to carry on.

Style manual or journal used IEEE Transactions on Wireless Communications

Computer software used Microsoft Word 2002

TABLE OF CONTENTS

LIST OF TABLES.....	xi
LIST OF FIGURES.....	xii
1 INTRODUCTION.....	1
1.1 Wireless Communications.....	3
1.2 Cellular Communications.....	4
1.3 Application Requirements.....	5
1.3.1 Requirements for Undergraduate Laboratory Experiments.....	6
1.3.2 Additional Requirements for Graduate Research.....	6
1.4 Thesis Organization.....	6
2 BACKGROUND AND LITERATURE REVIEW.....	8
2.1 General Considerations.....	8
2.2 Receiver Architectures.....	9
2.2.1 Receiver building blocks.....	10
2.2.2 Super-Heterodyne Receiver.....	11
2.2.2.1 The Image Frequency Problem.....	12
2.2.3 Direct Conversion Receiver (DRC).....	13
2.2.4 Image-Reject Receivers.....	14
2.2.4.1 Image-Rejection Architectures.....	15
2.2.4.2 Double Conversion Wide-band IF Receiver.....	15
2.3 Radio Frequency Transceiver.....	16
2.4 Receiver Challenges.....	18
2.5 Spread Spectrum Modulation.....	20
2.6 Software Radio.....	22
2.6.1 DSR Architecture.....	23
3 MATERIALS AND METHODS USED IN THE AUSDR-2 PROJECT.....	26
3.1 RF Front End Using the Cirronet WIT2410 Module.....	26
3.2 Baseband Processor-Motorola MC68HC11 Microcontroller.....	27
3.2.1 MC68HC11 Port Overview.....	28
3.2.2 Serial Communication Interface (SCI) Set Up.....	29
3.3 SDR-1 versus SDR-2.....	31
4 MEASUREMENTS AND RESULTS.....	33
4.1 Serial Communication Interface (SCI) using the Motorola MC68HC11.....	33
4.2 Interfacing the CME11E9-EVBU to the WIT2410.....	37
5 SUMMARY AND CONCLUSIONS.....	39
A CME9_EVBU.....	41
A.1 CME9-EVBU Overview.....	41
A.2 Port Connection.....	42
A.3 HC11 CODE.....	43
A.3.1 Transmitter SCI code.....	43

	A.3.2	Receiver SCI code.....	48
A.4		HC11 Code testing for different Baud Rate.....	52
A.5		Serial Communications Interface with WIT2410.....	58
B		AUSR-1 Overview.....	64
	B.1	RF Front End Architecture.....	64
	B.2	DSP's for Baseband Processing.....	67
	B.3	Code Development.....	67
	B.4	Results form AUSR-1.....	67

LIST OF TABLES

3.1	WIT2410 pins.....	27
3.2	Baud rate selection.....	30
A.1	Port connection.....	43

LIST OF FIGURES

1.1	Software Define Radio general Architecture.....	2
1.2	Structures of the main three cellular communications.....	5
2.1	Conventional Super-Heterodyne Receiver.....	9
2.2	Super-heterodyne receiver architecture.....	11
2.3	Down-conversion mixer.....	12
2.4	Direct conversion receiver.....	14
2.5	DC offset problem of DCR.....	14
2.6	Double conversion Wide-Band IF receiver.....	15
2.7	Direct-conversion receiver circuit.....	18
2.8	Heterodyne receiver.....	20
2.9	The basis of direct sequence spreading.....	21
2.10	FHSS and DSSS Power Spectral Densities.....	22
2.11	Dimensions of Software Radio Implementation.....	23
2.12	Traditional Model of a Radio Communications System.....	24
3.1	WIT2410 board.....	27
3.2	M68HC11 E-Series Block Diagram.....	29
3.3	SIC link photo.....	31
4.1	SCI via two 68HC11 boards.....	34
4.2	Key 6 is pressed (10111101 => rwo 2 and column 3 are shorted).....	35
4.3	Key 1 is pressed (00110000 = key 1).....	36
4.4	Key 9 is pressed (01110011 = key 9).....	36
4.5	WIT2410 connected to a signal generator with 1 Mhz square wave connected to TxD input.....	37
4.6	Software radio testset using MC68HC11 microcontroller evaluation board and WIT2410 modem. The 'HC11 can be programmed to exert a significant amount of detailed control over the operation of the modem, in addition to providing baseband data and encoding.....	38
A.1	CME11E9-EVBU board.....	41
A.2	Key pad row and column pins out.....	42
A.3	7-Segment LED pins out.....	42
B.1	RF front end design using RFMD WLAN chipset.....	66
B.2	RF Receiver front-end diagram.....	67
B.3	RF Transmitter front-end diagram.....	68
B.4	RF transmitter photo.....	69
B.5	RF receiver photo.....	69
B.6	Signal received through RF link.....	70

CHAPTER 1

INTRODUCTION

In wireless telecommunication there is an ongoing search for new technologies for improvement of bandwidth, capacity, interference rejection, voice quality, and so on. Many achievements have been made regarding modulation techniques and coding to find reliable and more efficient ways to send information wirelessly. As wireless technology encompasses multiple standards around the globe, the software defined radio (SDR) has emerged as a sort of "universal translator." Additionally, the financial impact of SDR is potentially considerable, since wireless handsets and basestations could be provided with significant upgrades to meet new standards and to take advantage of new modulation and encoding technologies by downloading software rather than by replacing hardware. Given the impending explosion of development in this area, it is important for students of wireless engineering to have relevant classroom and laboratory instruction. This thesis presents the design of a laboratory testset suitable for use by undergraduate and graduate students performing experiments and research in the field of SDR. Auburn University has introduced wireless laboratories to undergraduates in which students learn about practical aspects of signal processing, radio-frequency (RF) design, and measurement principles. Graduate-level research seeks to expand the knowledge of modern wireless systems.

The ultimate goal of SDR is to eliminate the analog RF front-end in radio systems; however, this goal has not yet been achieved because of the heavy digital processing load required. The general architecture of modern SDR shown in Figure 1.1 consists of antenna, front-end, transmitter, receiver, and baseband. These terms will be discussed in more detail in Chapter 2.

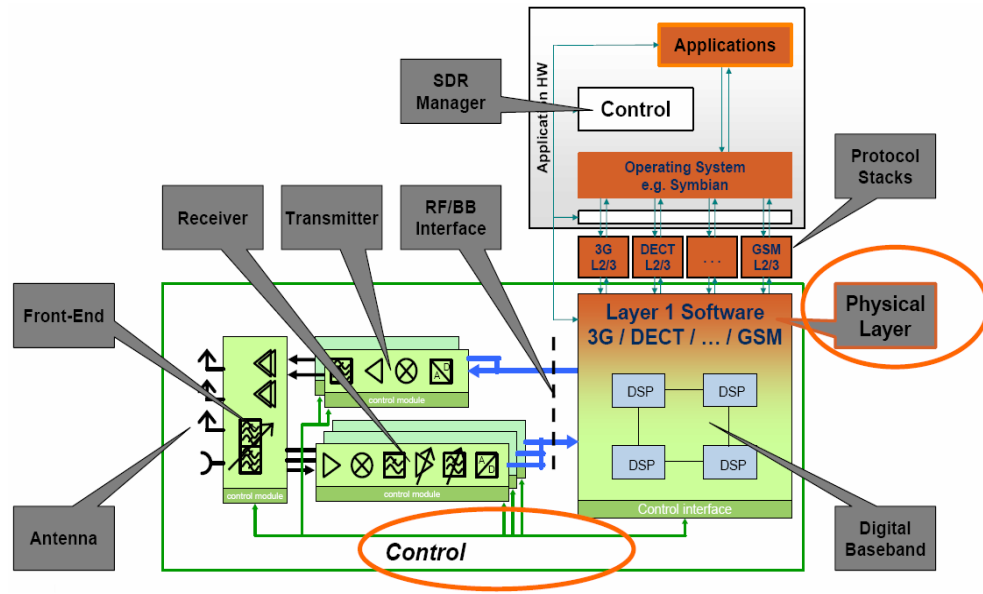


Figure 1.1- Software Define Radio General Architecture [55]

SDR technology draws on many different disciplines as can be seen in Figure 1.1. This includes mainly software, hardware, digital signal processing, and RF design. In recent developments of SDR, the design is focused on optimizing the application of hardware and software, whereas in the past, the approach of mainly software or entirely hardware (Application-Specific Integrated Circuit (ASIC), Field-Programmable Gate Array (FPGA) or Digital Signal Processor (DSP)) design had been practiced. However, at present, neither a purely hardware nor purely software approach can handle all the requirements proposed by standards such as the US Department of Defense Software Communication Architecture (SCA) [1].

This work is an outgrowth and extension of extensive collaboration with Mr. Fraidun Akhi (MSEE, Auburn University, 2003). The author collaborated with Mr. Akhi during the design, construction, and testing of a previous SDR version, hereinafter referred to as AUSDR-1 [52, 53]. The design of AUSDR-1 was based on the use of a set of commercially-available modular boards implementing various transceiver functions: LNA/mixer, QPSK modem, RF power amplifier, and bandpass filter. The baseband processing was accomplished using DSP evaluation boards and an op-amp based analog signal processor. The author participated extensively in the testing of the RF components, wrote the DSP code, and designed and constructed the op-amp circuits for AUSDR-1. As a first-generation design, AUSDR-1 was successful in many respects, but suffered from extreme sensitivity to ESD and difficulties with noise

suppression. These problems were mainly due to the physical layout and packaging of the individual boards, and the lack of a suitable ESD workstation in our laboratory. The present work (referred to as AUSDR-2) is focused on development of a more practical and rugged design suitable for use in an undergraduate wireless engineering laboratory, such as ELEC 3060 at Auburn University.

The novel aspects of the present work compared with AUSDR-1 are as follows:

- (a) AUSDR-2 has increased robustness due to the use of a commercially available RF module (Cirronet WIT2410) which is packaged in a single, shielded housing.
- (b) In AUSDR-2, the baseband processing is performed on a Motorola MC68HC11 microprocessor, rather on a DSP. Although a DSP offers more flexibility in the choice of encoding schemes, the MC68HC11 is considerably less expensive and is likely to have been used by most students prior to taking a wireless lab, so the time to achieve working code may be significantly reduced.
- (c) A typical experiment is presented that might be performed in an undergraduate wireless lab. For completeness, a review of the AUSDR-1 project is provided in Appendix B. Only the aspects of the author's work that are distinct from AUSDR-1 are presented in the main body of the thesis.

1.1 Wireless Communications

During the last decade, the wireless communications market has grown significantly. This expansion has led to advances in wireless technology that have reduced the size and cost of mobile radios while improving performance. The increasing level of integration in wireless circuits has contributed to many of these improvements. However, increasing integration is dependent upon the development of novel transceiver architectures that allow the designer to eliminate large discrete electronic components and to combine multiple circuit blocks on a single chip. At the same time, numerous wireless standards have been introduced which dictate the performance specifications of the hardware in wireless devices, where hardware requirements differ substantially between wireless communications applications. As of 1998, no cellular phone manufacturer in North America had a mass-production dual-mode (Advanced Mobile Phone

System (AMPS) or Code Division Multiple Access (CDMA)) handset that met the requirements for both standards [2], despite the existence of numerous handsets that met specifications for one of the standards. The implementation of multiple cellular standards in a single architecture also requires novel transceiver design approaches. Most wireless systems use multiplexing transmission schemes that provide a means for many users to share a medium with minimal or no interference. For example, Frequency Division Multiplexing (FDM) is used in analog cellular; Time Division Multiple Access (TDMA) is used in digital cellular and personal communications service (PCS); and Code Division Multiple Access (CDMA) is used mostly for PCS. In the next section, those schemes will be broadly discussed.

1.2 Cellular Communication

The earliest cellular phone system is known as AMPS (advanced mobile phone system). AMPS is an analog system operating in the frequency band from 800-MHz to 900-MHz. It was released in 1983 by Ameritech. The system uses frequency modulation (FM) and depends on frequency division multiple accesses (FDMA) to allow simultaneous service to multiple users. Each user in the FDMA system is allocated 30-KHz bandwidth for each channel during the call duration. AMPS is the least efficient cellular system as a result of its limited spectrum, no room for spectrum growth, poor data communications, minimal privacy, low capacity, and inadequate fraud protection. To improve the FDMA system, time division multiple access (TDMA) was incorporated. TDMA system is built on FDMA by dividing conversations by frequency and time to support synchronous traffic. This system offers improvements in system capacity, security, performance, and voice quality. Its channel is shared between several users, which easily boosts the capacity of the system. Although TDMA is a digital system, it is still inefficient since it has no flexibility for varying digital data rates. In a competitive development, the Qualcomm Corporation introduced IS-95, the standard that uses code-division multiple access (CDMA) in its wireless cellular system. CDMA allows all users to share a common wideband channel that is separated by orthogonal spreading codes. This technique is especially targeted at improving the communication error resulting from multi-path effects.

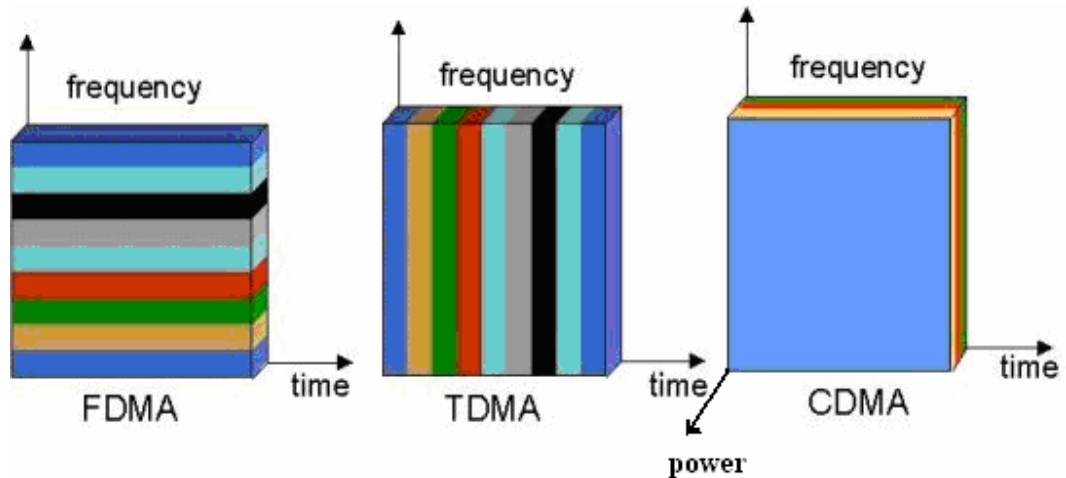


Figure 1.2- User allocations in the main three cellular communications protocols

FDMA assigns a channel to each of the N users for the duration of their respective calls while TDMA partitions each of the N channels into M time slots, and CDMA applies the orthogonal spreading codes technique to occupy the same channel.

In general, the number of multi-path signals in a wireless channel is unknown and difficult to predict. Spread spectrum (SS) technology aims to spread the information signal over a wider bandwidth to make jamming and interception more difficult while a receiver allows each arriving multi-path signal to be individually demodulated and then combined to produce a stronger and more accurate signal. The spread spectrum modulation techniques were originally developed for use in military and intelligence communications systems because of their resistance against jamming signals and low probability of interception.

1.3 Application Requirements

This section addresses the key features required for application of the proposed system to undergraduate laboratory experiments and to graduate research.

1.3.1 Requirements for Undergraduate Laboratory Experiments

In order to be suitable for use in an undergraduate laboratory, an SDR testbed must meet the following minimal requirements:

- (a) Rugged enough to survive repeated use and mis-use by students in a learning environment.
- (b) Reasonably simple to interface a variety of external signal sources and test equipment.
- (c) Inexpensive to power and instrument
- (d) Operationally flexible to support various experiments
- (e) Readily available replacement parts
- (f) Minimal learning curve for software development

1.3.2 Additional Requirements for Graduate Research

In order to be useful for graduate research, the SDR testset must also be accessible. That is, the user should be able to exert some degree of control over the operation of the key functional modules. For example, a researcher might want to interface his own low-noise amplifier to the existing system in order to validate his design.

1.4 Thesis Organization

A global overview of different receiver architectures is included in a literature review of Chapter 2. The relative merits of these architectures are discussed with respect to integration and supporting multiple wireless standards. It will deal with the definition, the architecture, technological challenges and commercial aspects of software radio.

The discussion of the methodology employed in the testing and evaluation of the WIT2410 and the 68HC11 microprocessor evaluation board, which together comprise the hardware basis for the proposed SDR, is illustrated in Chapter 3.

The results of some experiments with the software radio transceiver are presented in Chapter 4. Testing of the WIT2410 transceiver is discussed, and the serial communication interface using the CME11-EVBU evaluation board for the 68HC11 with different baud rates is demonstrated.

A summary of the work performed and results achieved are presented in Chapter 5 along with recommendations for further research.

A complete description of the serial communication interface (SCI) and code used for the example undergraduate laboratory experiments proposed is given in Appendix A.

A review of the original SDR testset, AUSDR-1 upon which this present work was based, is presented in appendix B.

CHAPTER 2

BACKGROUND AND LITERATURE REVIEW

This chapter begins by presenting a brief review of the theory of modern wireless communications as well as a review of the literature in the field of software radio and software-defined radio.

2.1 General Considerations

In the wireless communications environment a limited spectrum is allocated to each user, which translates to a limited rate of information and requires signal coding, compression and efficient modulation. The small available bandwidth also affects RF design since the receiver must be able to process a desired wireless channel while sufficiently rejecting neighboring interferers. For example, in the DCS-1800 Standard, a band-pass filter must have a pass-band 200KHz wide and provide 60 dB of attenuation 500 KHz from the center of a desired channel. For the filter, obtaining its high Q (quality factor) is important.

A filter's Q indicates the amount of resistance at resonance. Q-factor is defined as $Q = \frac{f_o}{f_2 - f_1} = \frac{f_o}{\Delta f}$,

where f_o is the resonant frequency, and f_2 is the upper and f_1 is the lower cutoff frequency. To have a high Q-factor such as the order of 10^5 , Surface Acoustic Wave filter (SAW) or similar technology must be employed. However, insertion loss (IL) (the loss resulting from the insertion of a device in a transmission

line, where $IL = 10 \log \left(\frac{P_{out}}{P_{in}} \right)$) increases with Q, which increases the noise figure of successive stages

by the loss factor. If a band-pass filter has an insertion loss of 3 dB and the low-noise amplifier that follows has a noise figure of 2 dB, then the combined noise figure of the two cascade devices rises to 5 dB, which reduces the receiver's overall sensitivity. Therefore, receiver design incorporates numerous trade-offs to simultaneously maximize receiver sensitivity, selectivity, and the level of integration. The following sections briefly discuss receiver and transmitter architecture.

2.2 Receiver Architectures

Currently, receivers are based on the traditional super-heterodyne scheme (Figure 2.1) where the RF and IF stages are completely analog with only the base-band stage is digital and is usually built in dedicated hardware. There are three main RF front end architectures in popular use today. These are super-heterodyne, direct conversion (or zero IF), and low IF (or image-rejection). The super-heterodyne receiver is a traditional design that has passed the test of time. As shown in figure below, a received message carrying an RF signal is down-converted (or mixed down) to base-band in multiple stages.

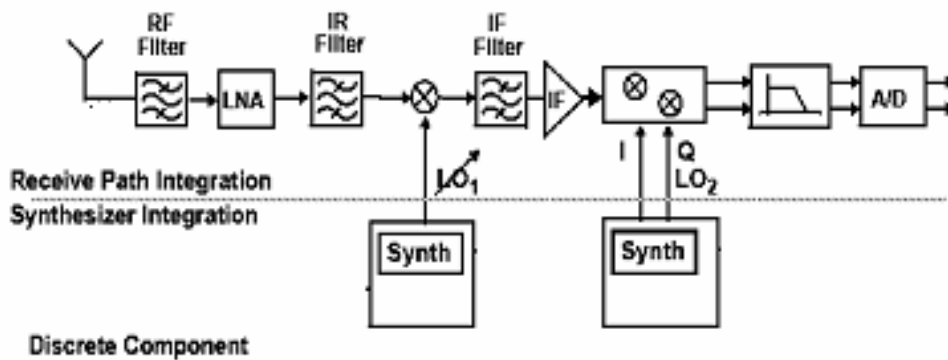


Figure 2.1- Conventional Super-Heterodyne Receiver. [Courtesy J. Rudell]

In order to provide good channel selectivity and receiver sensitivity, the super-heterodyne requires high- Q filters, which make it difficult to integrate an entire receiver and to support multiple standards. The direct-conversion receiver eliminates the need for several high- Q filters, but it suffers from significant DC offset problems. Image-rejection receivers offer greater flexibility in supporting multiple standards and lend themselves to increased integration, but their performance is limited by inherent circuit mismatches. The performance of these three receiver architectures will be discussed with respect to the level of integration, ability to implement multiple standards, and ease of circuit implementation.

2.2.1 Receiver building blocks

An RF receiver typically contains the following building blocks [9].

Antenna: The antenna is the interface between the receiver and the free air. The antenna has many characteristics such as gain, bandwidth, radiation efficiency, beam width, and beam efficiency. The antenna is the interface between air and receiver and therefore the signals must be transferred in good condition as possible. The antenna should be impedance matched between the free air and the receiver input.

Low-noise amplifier: The low-noise amplifier (LNA) is the first amplifier in the receiver chain. Its influence on the noise figure is strong compared to the subsequent amplifiers. The amplifier must have a high gain, linear characteristic, and a very low noise figure. To design high gain LNA, the circuit may lose its linearity. Therefore, a tradeoff must be made between gain and noise figure.

RF filters: The RF filter is necessary to filter the desired signal from out-of-band noise. Especially the so-called image frequency, which has the same frequency difference to the LO (Local Oscillator) as the desired signal, can distort the signal. This way only the desired signal is transferred to the IF (intermediate frequency).

Mixer: The mixer is a circuit that is injected with the received signal and a reference signal from an oscillator. The mixer converts the desired frequencies to the IF band. This requires an RF filter as stated before. A special type of mixer is the image rejection mixer, which eliminates the image band in the mixer itself.

Local Oscillator: An oscillator that generates the reference signal for the mixer. The LO consists of a phased lock loop and a fixed reference oscillator. The fixed oscillator can be a digital circuit or a crystal oscillator. The phase locked loop will lock to a divided version of the reference oscillator. By using a low pass filter in the phase locked loop, phase noise is filtered out to get a steady oscillator. This loop filter

slows down the lock-time of the loop to the divided reference oscillator. This means that the loop filter should be narrow enough to limit oscillator spurs, but wide enough to have a fast lock time.

IF filter: The Intermediate Frequency is also known as the beat frequency. It is the difference between the LO frequency and the carrier frequency. Only the IF frequency range is passed by the IF filter. The summed result of the LO and the carrier frequency is removed as well as out of band noise.

Detector: The final stage is a detector to convert the signal to a suitable base-band signal. The type of detector depends upon the modulation technique used. For FSK (Frequency Shift Keying) an FM detector is used, which converts the frequency shifted signal to an analog NRZ (Not Return to Zero) signal. The I/Q (I: in phase; Q: quadrature phase) demodulator is necessary if a form of quadrature modulation is used.

2.2.2 Super-Heterodyne Receiver

In Figure 2.2, the information signal is fed by the RF antenna and is followed by a band-pass filter (BPF) and then amplified with a low-noise amplifier (LNA). In the super-heterodyne architecture, the signal band is translated to a lower frequency, which relaxes the Q requirements for the channel selection filter. The translation of the carrier signal to a lower frequency is accomplished using a mixer, which can be viewed as an analog multiplier. The advantages of this type of receiver are low power, no dc offset, and high performance.

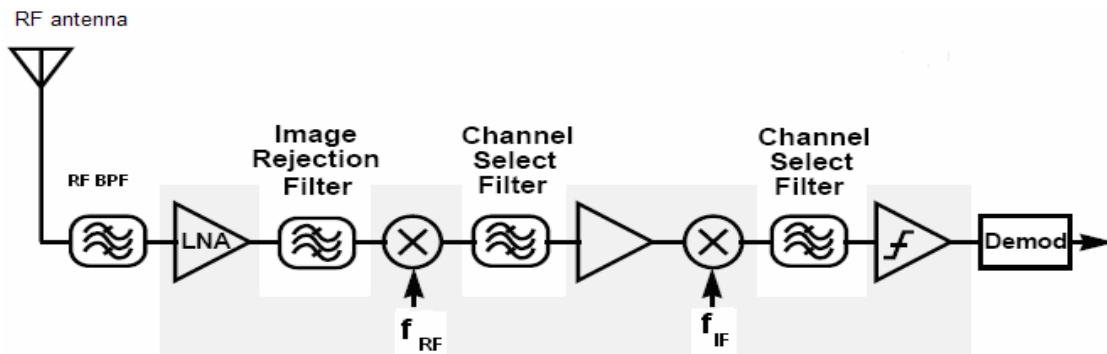


Figure 2.2- super-heterodyne receiver architecture

If a signal centered at ω_{RF} is to be translated to a frequency ω_{IF} , it is mixed with a sinusoid $\omega_{LO} = \omega_{RF} - \omega_{IF}$ generated by a local oscillator, which yields a signal at ω_{IF} , and a signal at $2\omega_{RF} - \omega_{IF}$ is removed through low-pass filtering. The lower frequency is known as the intermediate frequency (IF). This classic radio architecture has been used for decades because of its excellent channel sensitivity and selectivity characteristics. However, this comes at the expense of more complexity and cost. In particular it requires an off-chip surface-acoustic wave (SAW) filter in combination with additional IF circuitry. Since SAW filters are fabricated using different material technology, they cannot be integrated on the IC. To replace SAW filters in the heterodyne receiver, the low IF architecture can be used, where the RF signal is converted to a very low IF frequency, and then filtered using passive or active in-circuit filters. However, this architecture requires careful attention to issues of image rejection and channel selectivity, as discussed in the next section.

2.2.2.1 The Image Frequency Problem

The frequency translation process creates unwanted signal artifacts, which can interfere with the desired signal. It is helpful to consider a simple time-domain representation of the mixing process. Figure 2.2 shows a down-conversion mixer.

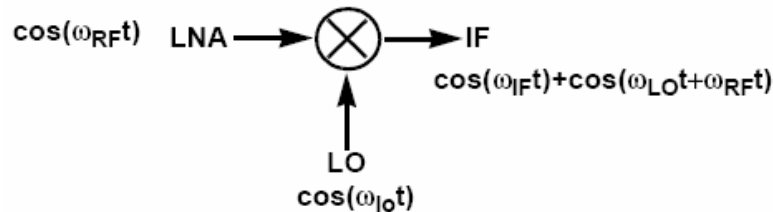


Figure 2.3- down-conversion mixer

The down-conversion mixing process generates two output signals, one at a desired IF, and a second at a higher, undesired frequency. If the local oscillator is at frequency $\omega_{LO} = \omega_{RF} - \omega_{IF}$, then both the signal band at ω_{RF} and the signal band at $2\omega_{RF} - \omega_{IF}$ are translated to the IF. The signal at

$2\omega_{RF} - \omega_{IF}$ is known as the image frequency. The image frequency problem is significant because, while wireless standards restrict the output power of their own users, they have no control over adjacent bands. The power of the image signal can be much higher than that of the desired signal, which forces the wireless designer to include some form of image rejection in receiver architecture. The most common method of image suppression in a super-heterodyne architecture is to place an image-reject filter immediately before the mixing stage in the receive path (Figure 2.2). This filter must have low loss in the signal band, and large attenuation in the image band. Provided the receiver's IF is large enough, the filter's Q will be small enough for a practical filter implementation; however, the receiver's IF must be small in order for an IF channel selection filter to be practical. This results in a significant trade-off in the design of super-heterodyne receivers since a low IF allows great suppression of adjacent channel interferers, while a high IF gives substantial image rejection [5].

2.2.3 Direct Conversion Receiver (DCR)

Radio receivers with a direct-conversion architecture, also known as zero-IF architecture, are more suitable for integration compared with the super-heterodyne architecture. They convert the RF signal all the way down to zero frequency. The DCR uses the fact that a signal and its image are separated by twice the intermediate frequency (IF). Thus, a down conversion architecture with zero IF produces an image at zero frequency, where the LO coincides with the incoming carrier frequency giving an IF of zero Hz. i.e., the desired signal is its own image. This is an important advantage over the heterodyne architecture since no image filter is required, and the low-noise amplifier (LNA) need not drive a 50 ohm load, which is common when an off-chip image filter is used. The "homodyne" or "direct-conversion" receiver architecture eliminates many discrete components in the receive signal path. In this approach, the RF signal is translated directly to base-band using a single mixer stage (Figure 2.4). This removes the IF stage from the receiver, and eliminates the need for image rejection. Energy from undesired channels is removed with on-chip low-pass filtering. The advantage of this receiver type is the reduced hardware requirement.

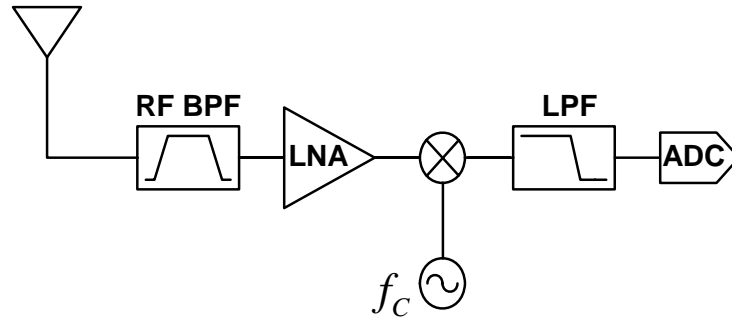


Figure 2.4- direct conversion receiver

There are problems (see figure 2.5) associated with the direct conversion architecture despite its greater degree of integration than a super-heterodyne system. Since the Local Oscillator (LO) is at the same frequency as the RF carrier, the receiver may have LO leakage directly to the mixer input. The LO signal may also leak to the antenna, be transmitted and “re-received,” which results in self-mixing and gives a time-varying DC offset at the mixer output [6]. Inherent circuit offsets and this time-varying DC offset reduce the receiver’s dynamic range.

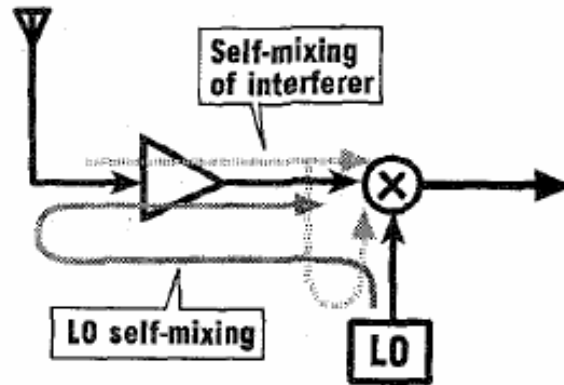


Figure 2.5- DC offset problem of DCR

2.2.4 Image-Rejection Receivers

The use of integrated image-reject filters limits the flexibility of the heterodyne receiver architecture. In contrast, image-reject receivers suppress image frequencies by processing the signal and its image differently. Using several phase shifts, it is possible to generate a replica of the input signal that contains the desired signal and the negation of the image signal. The image signal may then be cancelled by a simple summation operation at base-band.

2.2.4.1 Image-Rejection Architectures

Image-reject receivers can typically be divided into single-conversion and double-conversion architectures. The single-conversion architecture is based on the Hartley modulator [7] and uses a single quadrature mixing stage followed by a pair of quadrature phase shifters. The main drawback of this design is that it is difficult to build low-loss quadrature phase shifters at high frequencies. An alternative approach is based on the double-conversion Weaver architecture [8], which uses a second quadrature mixing stage to eliminate the phase shifters in the Hartley architecture. An image-rejection architecture that lends itself to the Weaver architecture and facilitates full receiver integration is the double conversion wide-band IF receiver architecture.

2.2.4.2 Double Conversion Wide-band IF Receiver

The receiver system proposed in [4] (Figure 2.6) translates the entire channel band from RF to IF using a single mixer. A low-pass filter at IF removes up-converted frequency components, and passes the channel band to the second mixing stage. The channels are then translated to base-band using a tunable channel-select frequency synthesizer. A variable-gain base-band filtering network is used to remove unwanted channels.

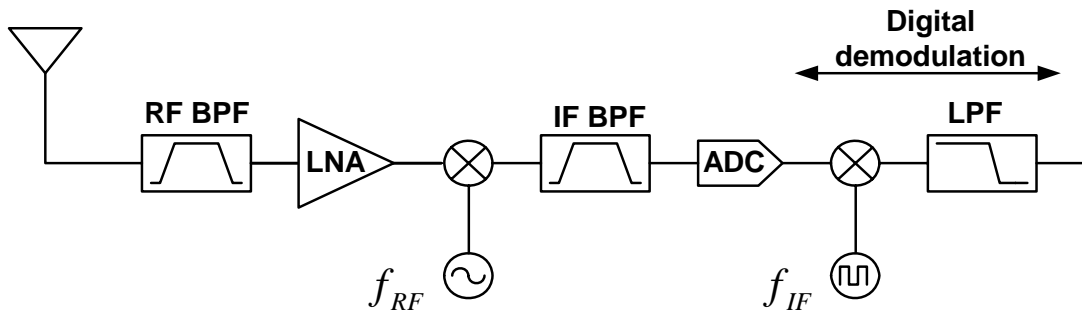


Figure 2.6- Double Conversion Wide-Band IF Receiver

This is similar to the super-heterodyne approach in that it uses several stages of frequency translation. Unlike a conventional super-heterodyne receiver, the first mixer translates the entire receive

band, which results in a large signal bandwidth at IF. Channel selection is performed at a second lower-frequency mixer with a tunable local oscillator. In direct conversion, channel filtering is done at base-band, where digital filters can be used to provide multi-standard receiver features. The use of digital filter components can provide superior performance in the super-heterodyne receiver, which increases the size of the receiver and tailors it to a particular standard. Compared to the Direct-Conversion approach, the Wide-Band IF architecture offers several potential advantages. However, both the Direct-Conversion and Wide-Band IF receivers perform channel filtering at base-band, where it is possible to implement an integrated programmable channel filter with multi-standard capabilities. The most significant feature of Wide-Band IF architecture is that channel tuning takes place at the second mixing stage and does not use the first LO, which can then be implemented using a Phase-Locked Loop (PLL) and a fixed-frequency crystal oscillator. This reduces the VCO's contribution to phase noise and allows it to be implemented using low-Q on-chip components. In addition, since channel tuning is performed by the IF oscillator, the PLL's divider ratio can be reduced. This reduces the contributions of the reference oscillator, phase detector and divider circuit to the phase noise of the frequency synthesizer. A lower divider ratio also reduces spurious tones generated by the PLL. The wide-band IF architecture is also insensitive to LO retransmission and self-mixing since there is no local oscillator at the same frequency as the RF carrier. The wide-band IF system facilitates high integration, but several non-idealities limit overall receiver performance. The first local oscillator is fixed in frequency, and channel selection is performed at the second mixing stage which increases the relative frequency tuning range required of the IF synthesizer. The harmonics of the second LO may also down-convert unfiltered interferers from IF to base-band.

2.3 Radio Frequency Transceiver

The RF part of a wireless communication system generally provides a conversion between the radio frequency and base-band or intermediate frequencies (IF). This is called up and down-conversion, which is respectively related to up and down shifting in the frequency spectrum of the signal that needs to be received or transmitted. Many chip manufacturers develop integrated circuits for RF applications covering all analog and digital communication standards in the world based on the growth of wireless communication. These products range from fully integrated circuits for digital standards, to flexible

building blocks for custom RF products. The ICs are fully integrated, providing a complete receiver, transmitter or transceiver to be implemented in a mobile communication device. The mobile communication standards DECT, GSM, IS-95, D-APMS, are just a few from the long list of mobile communication technologies that exist all over the world [54]. Certain systems will be easier to implement and certain radio architectures are more suitable than others. From the theory outlined previously, it can be seen that for the commonly-used algorithms magnitude and phase information are both important. The receiver not only receives and recovers the in phase and quadrature signals, but also preserves phase and amplitude information of the RF signal. In a receiver, several effects appear in analog and digital stages which must be compensated for. Therefore, the RF front-end should be designed according to the following two statements.

Phase and amplitude preservation

All receiver architectures that have discussed above should be able to preserve the carrier's phase and amplitude from RF to base-band. This requires the use of highly linear receivers and transmitters. An RF transceiver is a very complex device that is subject to non-linearities, noise, jitter etc.

Synchronization

Due to manufacturing dissimilarities and component mismatches, receiver architectures will suffer from differences in frequency and phase of input signals. If the translation of I/Q (*in-phase* (I) and *quadrature* (Q) components of the downconverted signal) signals to RF signals is subjected to non-stationary phase mismatches, the received signals will not synchronize.

2.4 Receiver Challenges

One of the most common building blocks of a receiver is the mixer. Generally the two inputs of the mixer are the received signal and a locally generated signal called Local Oscillator (LO). The basic function of a mixer is a translation of an input signal to a different frequency. The math is formed by the trigonometric relations.

$$\begin{aligned}\cos(\omega_1 t + \phi) \cos(\omega_2 t) &= \frac{1}{2} [\cos((\omega_1 - \omega_2)t + \phi) + \cos((\omega_1 + \omega_2)t + \phi)] \\ \cos(\omega_1 t + \phi) \sin(\omega_2 t) &= \frac{1}{2} [\sin((\omega_1 - \omega_2)t + \phi) + \sin((\omega_1 + \omega_2)t + \phi)]\end{aligned}\tag{2.3.0}$$

The angular frequencies of any arbitrary signal that has a form as above equations are represented by ω_1 and ω_2 where ϕ is the phase difference. Relating the above equations to the function of the mixer is the translation of both frequencies to sum and difference frequencies. In receiver or transmitter architectures, only one of the two output frequencies is interesting, and therefore the mixer will be combined with an output filter, filtering either the sum or difference frequency. The concept of the direct-conversion receiver is visible in Figure 2.7.

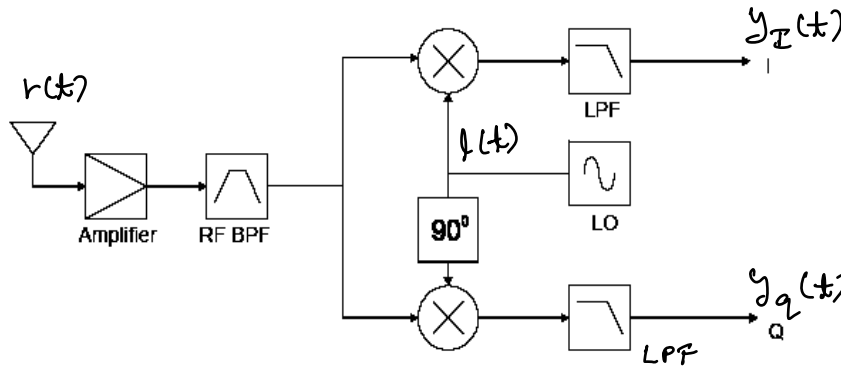


Figure 2.7- Direct-conversion receiver circuit

The received signal at the antenna is amplified and then injected to the mixer stage after going through RF band pass filter. The mixer stage usually consists of two mixers, which demodulate the received signal. The LO is offered to both mixers with a 90 degree phase shift for the demodulation of the

in phase and quadrature component of $r(t)$. Consider the received signal $r(t)$ a quadrature modulated signal represented by:

$$r(t) = m_I(t) \cos(2\pi f_c t + \phi) + m_Q(t) \sin(2\pi f_c t + \phi) \quad (2.3.1)$$

$m_I(t)$ and $m_Q(t)$ represent respectively the in phase and quadrature message components. Then, the receiver in phase component can be expressed as

$$y_I(t) = [m_I(t) \cos(2\pi f_c t + \phi) + m_Q(t) \sin(2\pi f_c t + \phi)] \times [\cos(2\pi f_c t) + \hat{\phi}] \quad (2.3.2)$$

In (2.3.2) $y_I(t)$ is the output of the receiver for the in phase message component. ϕ and $\hat{\phi}$ are respectively the phase of the received signal and the phase of the local oscillator. By applying the trigonometric relations (2.3.0), $y_I(t)$ can be written as

$$y_I(t) = \frac{1}{2} m_I(t) \cos(\phi - \hat{\phi}) + \frac{1}{2} m_I(t) \cos(4\pi f_c t + \phi + \hat{\phi}) + \frac{1}{2} m_Q(t) \sin(\phi - \hat{\phi}) + \frac{1}{2} m_Q(t) \sin(4\pi f_c t + \phi + \hat{\phi}) \quad (2.3.3)$$

If the high frequency component is applying lowpass filter (LPF), the equation of (2.3.3) becomes

$$y_I(t) = \frac{1}{2} m_I(t) \cos(\phi - \hat{\phi}) + \frac{1}{2} m_Q(t) \sin(\phi - \hat{\phi}) \quad (2.3.4)$$

By applying the above procedure with the LO signal phase shifted by 90 degree to obtain the quadrature output, $y_Q(t)$, in (2.3.5) as

$$y_Q(t) = \frac{1}{2} m_Q(t) \cos(\phi - \hat{\phi}) + \frac{1}{2} m_I(t) \sin(\phi - \hat{\phi}) \quad (2.3.5)$$

When $\phi = \hat{\phi}$, $y_I(t)$ and $y_Q(t)$ become

$$y_I(t) = \frac{1}{2} m_I(t) \quad (2.3.6)$$

$$y_Q(t) = \frac{1}{2} m_Q(t) \quad (2.3.7)$$

The effect of a mismatch in phase is clear from (2.3.4) and (2.3.5). If there is a mismatch between the phase $\hat{\phi}, \phi$, the quadrature component $y_Q(t)$ leaks to the in phase component $y_I(t)$ and visa versa. In heterodyne receiver architecture, its principle is to use of more than one mixer stage, to convert the radio frequency to the base-band. The concept of the heterodyne receiver is visualized in Figure 2.8.

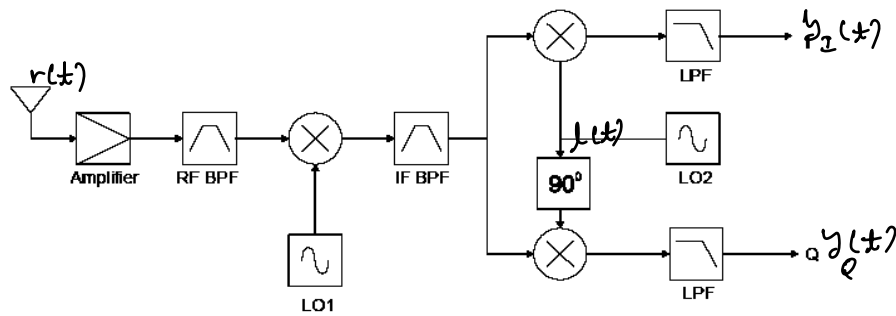


Figure 2.8- heterodyne receiver

Instead of converting the received signal directly to the base-band, the first stage converts the signal to an intermediate frequency range. Next step, the signal is band-pass-filtered to filter one of the resulting images then the signal is converted to in phase and quadrature signals. The last conversion is based on exactly the same principle as the direct conversion receiver. Certain modulation types do not need an implementation where the signal is converted to I/Q plane to finally detect the bits.

2.5 Spread Spectrum Modulation

A number of modulation techniques have been developed to generate spread spectrum signals. These can be generally classified as direct sequence (DS), frequency hopping (FH), time hopping (TH), chip modulation and the hybrid combination modulation. The generation of a DS signal using an exclusive-OR (XOR) operation is illustrated in Figure 1.2, where the data symbols, with period T_S , are modulo-two added (XORed) to the PN (pseudo-random noise) sequence with chip period T_C and T_S at the transmitter, where $T_S \gg T_C$. The basis of the DS spreading technique is depicted in Figure 2.7.

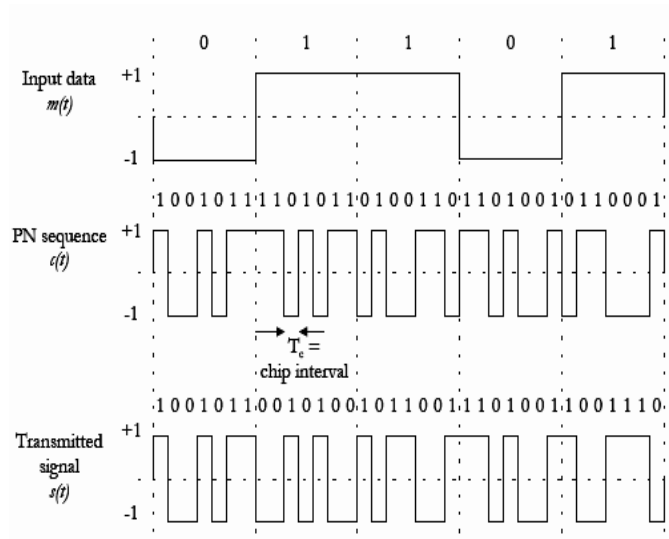


Figure 2.9- The basis of direct-sequence spreading

Direct Sequence Spread Spectrum (DSSS) is produced by multiplying the transmitted data stream by a pseudo random sequence or performs an XOR operation of PN with spread sequence. The processing gain (PG) is the ratio by which this modulating pattern exceeds the bit rate of the base-band data is defined as

$$PG = 10 \log \left(\frac{T_S}{T_C} \right).$$

The PG is used to determine the number of users that can be allowed in a system, the

amount of multi-path effect reduction, and the difficulty to jam or detect a signal. For the DS system, it is advantageous to have a PG as high as possible [3]. Figure 2.8 illustrated FH performance versus DS. When applying FH, the carrier frequency is ‘hopping’ according to a unique sequence. FH performs by varying the carrier frequency rapidly according to a PN over a broad range of channels while it keeps transmitting the data signal as usual. Figure 2.8 is presented the power spectrum of FHSS versus DSSS method.

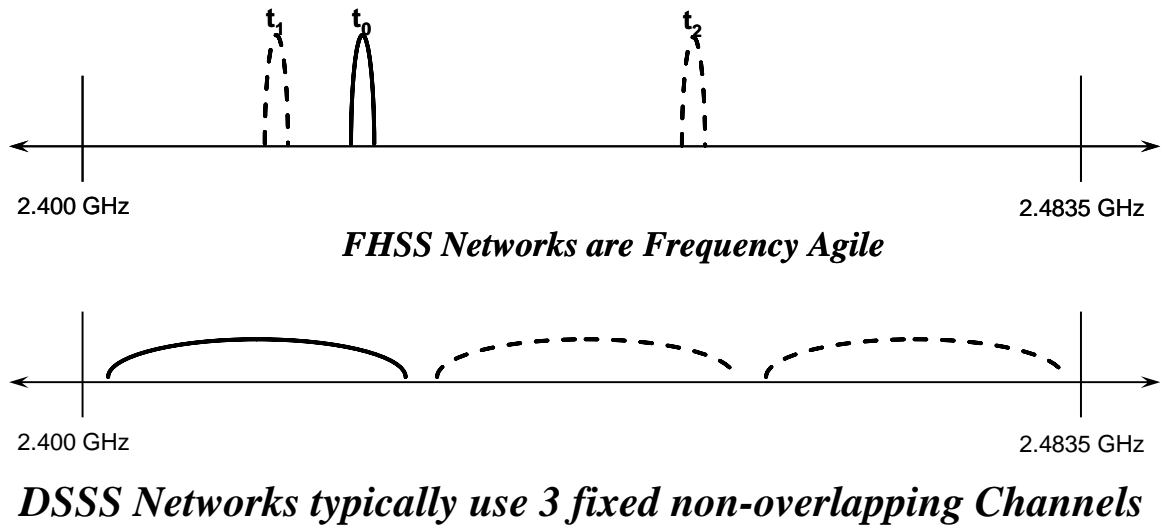


Figure 2.10- FHSS and DSSS Power Spectral Densities

A disadvantage of FH as opposed to DSSS is that obtaining a high processing gain is difficult. To obtain high PG in FH, a frequency-synthesizer is needed. For more detail of FH versus DSSS, see [35]. As the RF signal is spread over a large frequency-band (see Figure 2.8 FHSS), the power spectral density gets very small unlike DSSS networks. This system is providing higher user capacity, random access possibilities, data rates, and increasing securities because of unknown random codes.

2.6 Software Radio

Since the early 1980's the world market has seen the emergence of numerous cellular mobile systems. Correspondingly, numerous regional standards have been developed such as TACS, GSM, DCS-1800, IS-95 CDMA, etc. To overcome this roadblock to worldwide interoperability, the software-radio concept is introduced as a potential pragmatic solution for the user's terminal to be able to dynamically adapt to the radio environment in which it is located. The software radio concept can be viewed as a means to make users, service providers, and manufacturers more independent of standards, and to add a degree of flexibility to the radio platform. Overall, there is not one unique definition for the software radio concept. We use the definition proposed in [3]: "software radio is an emerging technology, aimed to build flexible radio systems, which are multiple-service, multi-standard, multi-band, reconfigurable and re-

programmable, by software.” Likewise, a software-defined radio (SDR) is a radio that can accommodate a significant range of RF bands and air interface modes through software [18]. A commercially viable SDR may be implemented as a mix of Application-Specific Integrated Circuits (ASICs), Field Programmable Gate Arrays (FPGAs), Digital Signal Processors (DSPs), and general-purpose microprocessors. ASIC parameters also may be defined in software, but an ideal software radio has no ASICs. Instead, it implements all the radio’s RF conversions, filtering, modem and related functions in software. The range of SDR implementations as a function of digital access bandwidth and processor programmability is shown in Figure 2.11. Implementation points A through D in the figure below are contemporary SDRs (see [19]).

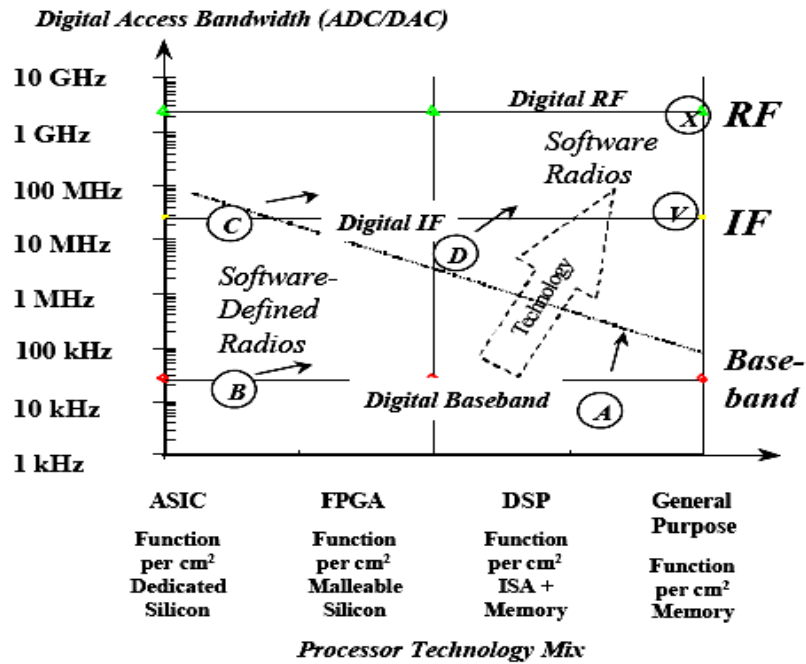


Figure 2.11- Dimensions of Software Radio Implementations

2.6.1 SDR Architecture

Although various technical standards exist in the current global marketplace, all communication processes require the same general steps to decode a radio signal into speech or data and vice versa for transmitters as well as for receivers. From the radio engineering perspective, the transmitter communication devices must first encode the information from the source into a bit stream. This bit stream

must be small enough to save channel bandwidth, which increases the capacity of the system network. Then it transforms this bit stream into a waveform compatible with the radio frequency (RF) communications channel. The next step is to cipher the bit stream before it is modulated so that one channel bandwidth of modulation signal is used. After modulation, the base band signal is converted into the appropriate part of the system bandwidth that marks the border between base band processing and intermediate-frequency (IF) processing. Finally, this band is converted into the real radio frequency (RF) band, which is used by the system. The process is illustrated in Figure 2.10.

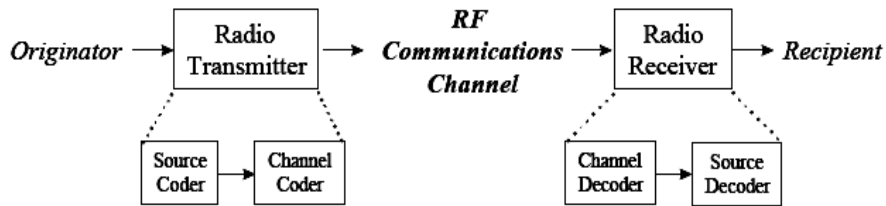


Figure 2.12 - Traditional Model of a Radio Communications System

BellSouth defined the SDR in order to call for an evolution towards greater programmability of wireless infrastructure [20]. This evolution includes programmable multi-band multimode radios. The IS-95/AMPS handset, for example, uses a Code Division Multiple Access (CDMA) chip-set for communication between mobiles and base stations. First, the bit stream is modulated using the quadrature-phase-shift-keying (QPSK) – modulation technique. This technique modulates two bits at a time; the phase

of a QPSK-signal has four values: $\frac{\pi}{4}, \frac{3\pi}{4}, \frac{5\pi}{4},$ and $\frac{7\pi}{4}$; the bit stream is then multiplied with a

pseudo-noise (PN) code. Because the bit stream is spread by the PN code, it uses a relative large frequency band. Therefore, the capacity of CDMA in theory is unlimited, but the PN codes are not purely orthogonal, thus the noise floor increases as users are added. The entire SDR system consists of an Analog Mobile Phone System (AMPS) chip set; a dual mode analog RF Integrated Circuit (RFIC) chip-set [21]; a DSP chip [22] for filtering, voice coding and other computationally intensive tasks; and a microcontroller for user interface and system control. Many of the functions are programmable, but the CDMA modem, for

example, is defined in an ASIC and may not be changed in the field. Over time, functions initially implemented in hardware (e.g., the base-band modem) will migrate to software. Conversely, functions initially defined in software (e.g., in a simulation) may migrate to hardware (e.g., an ASIC).

CHAPTER 3

MATERIALS AND METHODS USED IN THE AUSDR-2 PROJECT

This chapter describes the hardware used in the AUSDR-2 software-defined radio testset. The RF front end is provided by the Cirronet WIT2410 frequency-hopping spread spectrum module. The baseband processing is performed on a CME11E9-EVBU, which is a commercially available evaluation board for the Motorola MC68HC11 microcontroller.

3.1 RF Front-End Using the Cirronet WIT2410 Module

The WIT2410 Wireless Module from Cirronet, Inc. was proposed primarily because it has extremely low latency, good bandwidth, reasonable size and low power consumption, a package that eliminated most of the RF design challenges, and on-board management of the wireless protocol. This device is a commercially available product intended primarily for use as an OEM part in wireless industrial transceivers. Such modules are deployed around the world for factory automation, supervisory control and data automation (SCADA), medical telemetry, mining vehicle control, and fleet management. It uses TDMA for multichannel operation, and the RF section uses frequency hopping spread spectrum (FHSS) in the unlicensed ISM band in the range from 2.4 GHz to 2.483 GHz. It has a maximum data transfer rate of 460 kilobits per second with a specified range up to 3000ft indoors. With a high gain antenna, the range of operation can be up to 20 miles under ideal conditions outdoors. Its RF bandwidth is 750 KHz, the transmit power output is 10mW or 100mW, and the receiver sensitivity is -93dBm. A photograph of the board is shown in Figure 3.1. The pinout is given in Table 3.1.

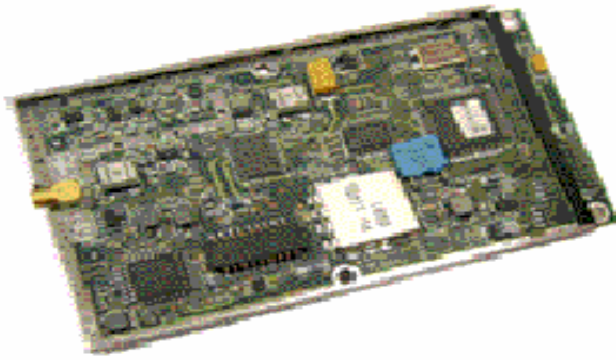


Figure 3.1- WIT2410 board

Table 3.1-WIT2410 PINS

<i>Pin</i>	<i>Signal</i>	<i>Type</i>	<i>Description</i>
1	GND		Signal and chassis ground
2	TXD	input	Input for serial data to be transmitted
3	RXD	output	Output for received serial data
4	/CFG	input	Used to switch between control and data modes
5	/RTS	input	Request to send
6	SLEEP	input	Sleeps/wakes radio transceiver
7	/DCD	Output	Data carrier detect
8	/CTS	Output	Transmit flow controller
9	NC		
10	reset	input	Resets the radio
11..15	NC		
16	VCC (3.3v)		Min 3.3V ; Max 10.0V

The results of testing of this device are provided in Section 4.2.

3.2 Baseband Processor – Motorola MC68HC11 Microcontroller

This section begins by detailing the serial communication interface (SCI) of the CME11E9-EVBU board, which is a commercially available evaluation board for the Motorola MC68HC11 microcontroller. The serial communication interface of the CME11E9-EVBU that is about to be described was an example of how to set up a demonstration of the two boards. If the WIT2410 is employed for both transmitter and receiver, the SCI can be implemented wirelessly.

3.2.1 MC68HC11 Port Overview

The MC68HC11 operates at nominal bus clock speed of 2 MHz. It has forty input and output pins which are arranged into five ports: A, B, C, D, and E. Ports A, B, and E are fixed direction ports, which means they have been configured for either input or output and they cannot be changed. Ports C and D are general purpose input/output registers which can be configured to allow input and output on their pins. The main two ports of interest for this experiment are ports B and C. Port B is automatically configured as an output port. When the pins of port B are read, the value will be exactly what has been sent to the port. The address of port B is \$1004. The following instruction sequence illustrates outputting the hex value 30 to port B.

```
REGBAS    EQU    $1000
PORTB     EQU    $04
          LDX    #REGBAS
          LDAA  #$30
          STAA  PORTB,X
```

Since port C is a general purpose input/output register, it has to be configured to allow data to flow in the desired direction. Port C has an associated data direction register called DDRC. To designate the data direction flow, the pins which the user wishes to specify as input need to be set to 1. The address of the DDRC is \$1007. For example, to program the upper four pins for output and the lower four pins for input the value \$F0 has to be stored in the DDRC. The instruction sequence for the operation would look like:

```
REGBAS    EQU    $1000
DDRC      EQU    $07
          LDX    #REGBAS
          LDAA  $F0
          STAA  DDRC, X
```

The CPU includes 8 Kbytes of read-only memory (ROM), 512 bytes of electrically erasable programmable ROM (EEPROM), and 256 bytes of random-access memory (RAM). An eight-channel

analog-to-digital (A/D) converter is included with eight bits of resolution; see [39] for more information on peripheral devices. Figure 6.1 gives an overview of the MC68HC11.

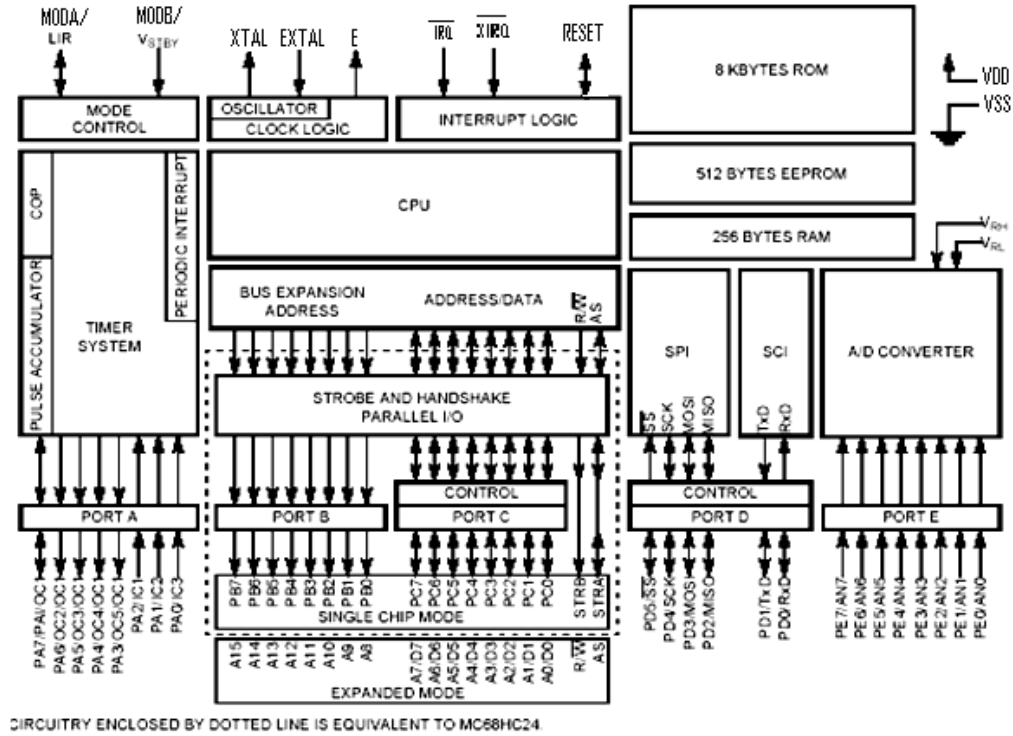


Figure 3.2- M68HC11 E-Series Block Diagram

3.2.2 Serial Communication Interface (SCI) Set Up

Two CME11E9-EVBU boards are used to demonstrate the serial communication interface (SCI) using the code given in Appendix A. The boards could be interfaced wirelessly using a pair of WIT2410's using the same code. For SCI specific detail, please refer to Chapter 9 of the Reference Manual [39]. The SCI is an independent serial I/O subsystem in the MCE11E9-EVBU. Port D pins TxD(PD1) and RxD(PD0) are used as Transmitter and Receiver data pins. The SCI subsystem is used primarily to connect a PC to the MCU and is configured/controlled by the following five registers.

- BAUD Baud-RateControl Register
- SCCR1 SCI Control Register 1
- SCCR2 SCI Control Register 2

SCSR SCI Status Register

SCDR SCI Data Register

The baud rate of the SCI system is derived by dividing the E clock signal at address \$102B as indicated in table 3.2.

Table 3.2- Baud rate selection

SCR2	SCR1	SCR0	Division Factor	Baud Rate
0	0	0	1	9600
0	0	1	2	4800
0	1	0	4	2400
0	1	1	8	1200
1	0	0	16	600
1	0	1	32	300
1	1	0	64	150

For establishing SCI between two boards, the two boards need common ground. The transmit routine is downloaded to one board while the receive routine is downloaded to other board. The PE0 pin on both boards need to be connected to Vcc for operation on the single mode that is operated/run the program upon pressing the reset button. A typical setup is shown in Fig. 3.3

operation for this device: control mode and data mode. While in data mode, the transmitter will only transmit the data. For the control mode, the modem parameters can be modified. The WIT2410 can also conserve power by enabling a duty cycle feature. Overall, SDR-2 is likely to be better than SDR-1 for use in an undergraduate laboratory environment in term of cost, maintenance, and ruggedness.

CHAPTER 4

MEASUREMENTS AND RESULTS

4.1 Serial Communication Interface (SCI) using the Motorola MC68HC11

The SCI interface capability of the Motorola MC68HC11 is designed for relatively slow data rate serial communication, but it is suitable for laboratory experiments at the undergraduate level. The board is not as sensitive to ESD as the DSP boards used in AUSDR-1 (the previous software define radio project). Two evaluation boards were interconnected as described in Chapter 3. The objective is to develop a protocol for transmitting a keypad-generated input from one board to a display connected to the other board. Figure 4.1 illustrates the connection of computer, digital logic analyzer, and 68HC11 boards. Although the boards are hardwired in this setup, the ultimate objective for a wireless laboratory experiment is to use a WIT2410 connected to each board to provide a wireless interface.

On the transmitter board, bits 0 - 3 of port C are configured as inputs and wired to the column pins of the keypad with 10 k Ω pull-up resistors. The upper bits of port C are configured as outputs and connected directly to the row pins of the keypad. Port C was programmed to scan for any key press. If no key is pressed, the program goes into an infinite loop looking for a key press. The HC11 code and wiring diagrams are presented in Appendix A.

Figures 4.2 through 4.4 show the logic analyzer display as sample keys are pressed.



Figure 4.1- SCI via two 68HC11 boards

Figure 4.2 is an example of SCI which can be used as a wireless experiment if the transmitter is connected to a WIT2410 and the receiver is connected to another WIT2410. The figure shows the logic analyzer display when key 6 is pressed on the keypad connected to the transmitter. PC0-PC7 are the inputs which represent which key is pressed (port C on the transmitter board), while PB6-PB0 are the outputs obtained after transmission (port B on the receiver board). E-clk is the clock signal of the 68HC11 board operating at 2 MHz. SCDR goes high when data transmission is completed.



Figure 4.2- Key 6 is pressed (10111101=>row 2 and column 3 are shorted)

As seen in Figure 4.2, port C data and Port B data happened to be the same; they both contain the data captured when key 6 was pressed.

In Figure 4.3, key 1 was pressed prior to capturing the data; therefore, port C is already in a loop looking for a new key press, while the data in port B represents that key 1 was pressed. Figure 4.4 shows a similar situation corresponding to key 9 being pressed. The translation from row / column encoding on the keypad to the 7-segment display encoding is performed in software on the transmitter board. The code was designed to operate at different SCI baud rates such as 9600, 4800, 1200, 600, 300, and 150. Displaying different baud rates is impossible on the logic analyzer; however, another section of code was written to observe operation at various baud rates. The code can be seen in Appendix A.2. This code displays a sequence of characters on the LED display, so the baud rate can be observed by how fast the display changes.

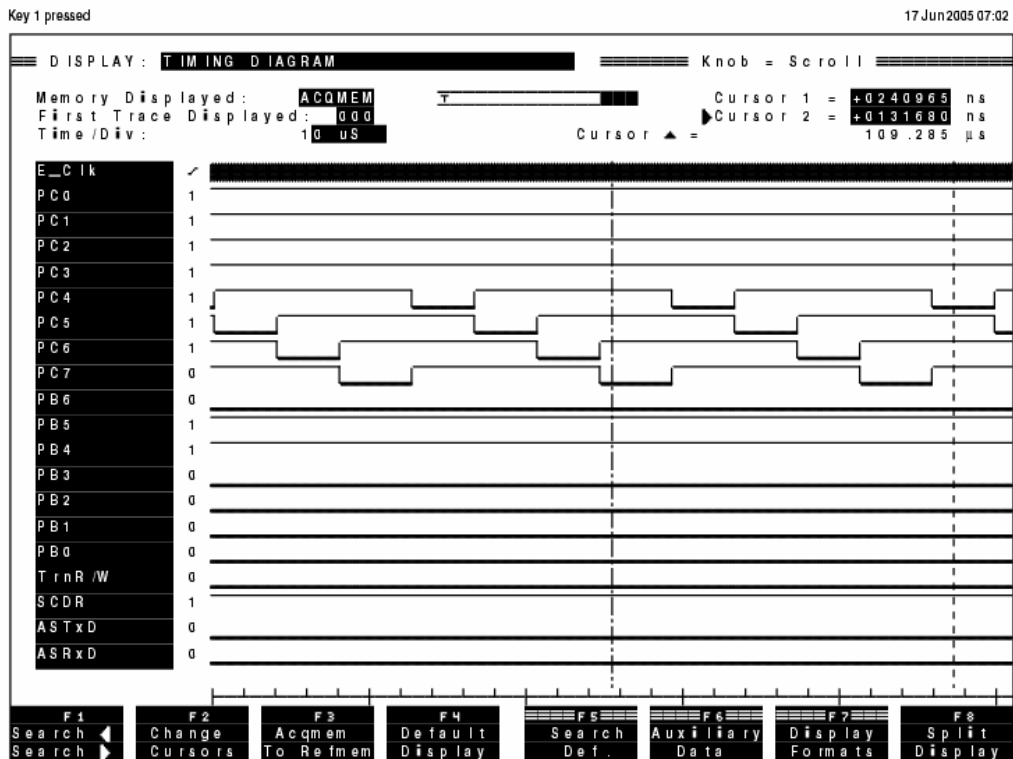


Figure 4.3- Key 1 is pressed (00110000 = key 1)

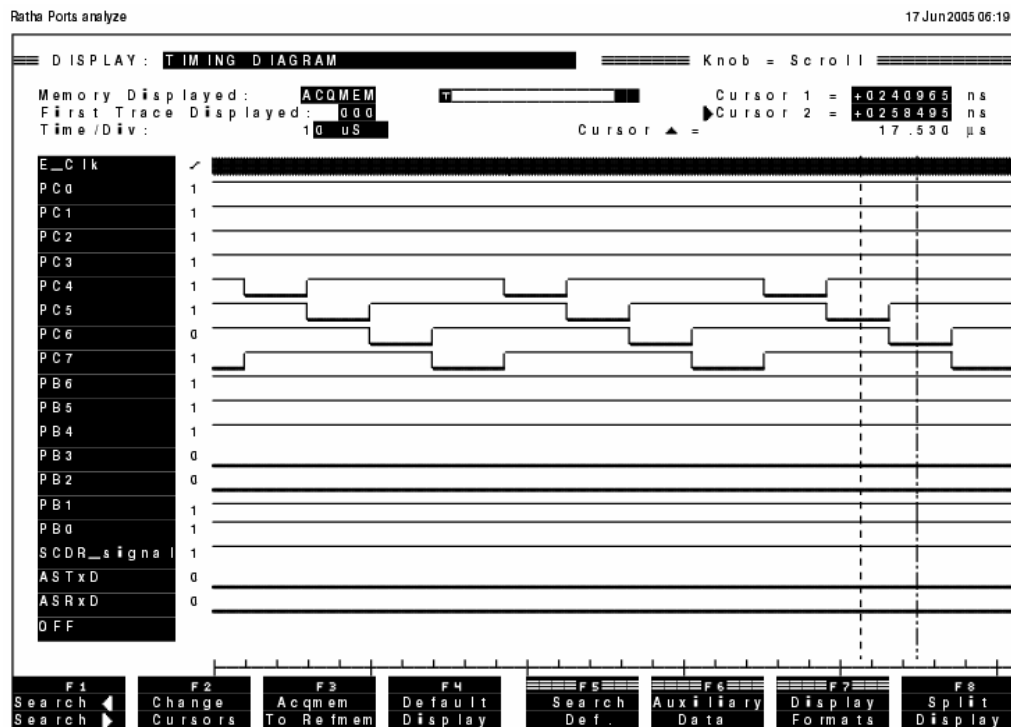


Figure 4.4- Key 9 is pressed (01110011 = key 9)

The code in Appendix A.2 contains several options, transmit or receive, word length (8/9 bits), baud rates, and parity odd/even for error detection. This assembly code may be useful for future implementation using a wireless link.

4.2 Interfacing the CME11E9-EVBU to the WIT2410

Only one WIT2410 module was available which prohibited a functional RF test. Nevertheless, the specifications and interface requirements were studied sufficiently to ensure that this module would interface suitably with the baseband processor. In Figure 4.5, a 5V peak to peak square wave with a frequency of 1MHz was fed to the transmit pin (pin 2) and measurements were taken from the RF output. Each pin of the WIT2410 was examined to verify functionality. However, due to resource limitations, it was not possible to conduct a full functional test. This testing is to verify the capability of WIT2410 in transmitting mode.

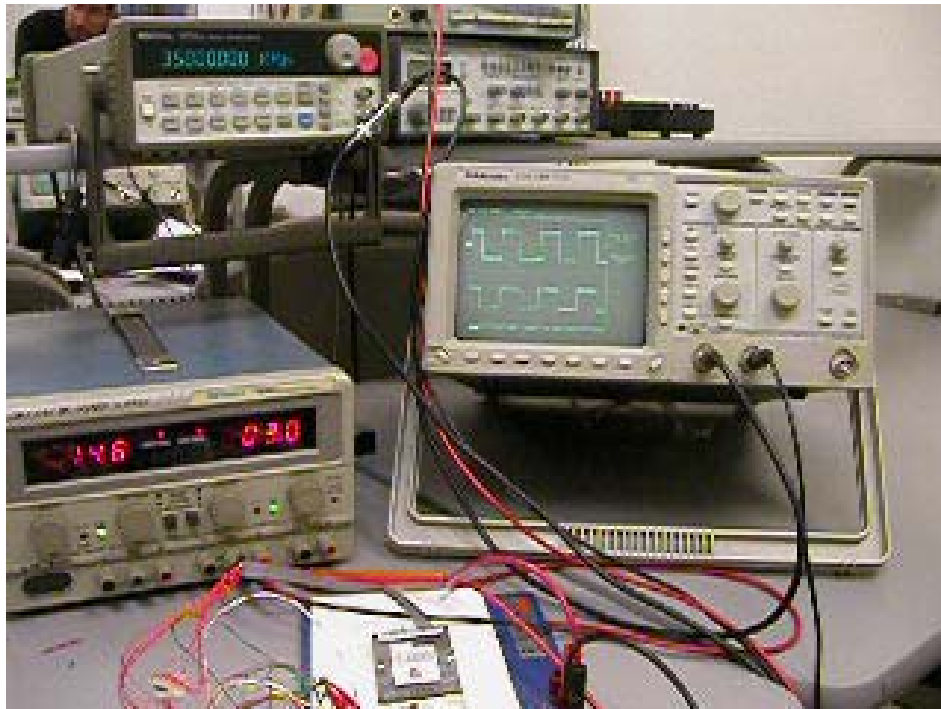


Figure 4.5- WIT2410 connected to a signal generator with 1 MHz square wave connected to TxD input.

Figure 4.6 shows a diagram of the proposed testset for performing undergraduate laboratory experiments and graduate research in software radio. The CME11E9-EVBU board is used to interface user input to the WIT2410 modem. The user will also write software to run on the evaluation board that will exercise control over the RF parameters of the WIT2410. The details of the control scheme are provided in [51]. They are summarized here.

The control mode of the WIT2410 permits the user to control the following parameters of the RF behavior:

- Frequency range (US, Europe, Asia, Canada)
- Frequency hopping pattern and duration
- Transmit power
- TDMA parameters (dynamic vs. static slot assignment, slot size)
- Network mode (star, roaming, or point-to-point)
- Data packet length
- ARQ (automatic retransmit request) parameters
- Report relative received signal strength
- Security code

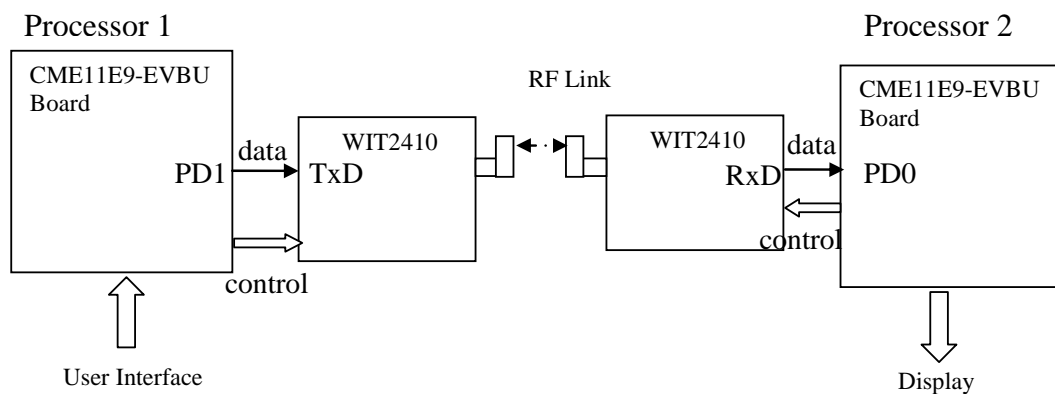


Figure 4.6- Software radio testset using MC68HC11 microcontroller evaluation board and WIT2410 modem. The 68HC11 can be programmed to exert a significant amount of control over the RF parameters of the modem, in addition to providing baseband data and encoding.

CHAPTER 5

SUMMARY AND CONCLUSIONS

In Chapters 1 and 2, motivation for this project was provided by outlining the growing popularity of the software radio concept in industry and academia. With constant improvement in digital signal processing, data conversion, and RF technology, the proliferation of software radio systems will undoubtedly accelerate. Thus, a research and education laboratory dedicated to this technology is highly attractive, if not essential for academic institutions that have wireless communications programs, such as Auburn University.

Most undergraduate wireless communications laboratories fall into two categories, based on the emphasis of their content. Some emphasize digital signal processing algorithm design and testing [44, 45], while others emphasize RF circuit design and testing [46-49]. A software-radio-based laboratory can bridge the gap between these two approaches. It will necessarily have a rugged, reliable, and readily accessible testset. The design proposed here consisting of a Motorola MC68HC11 coupled with a WIT2410 RF front-end can provide such a testset. An examples program using SCI has been written in assembly language to be used with the WIT2410 for testing. Experiments can be set up by measuring the bit error rate, signal to noise ratio, and the data loss in receiving mode.

It is the hope of the author that the software radio project undertaken as part of this thesis work will be continued. One of the main goals of the Wireless Engineering Initiative at Auburn University is to create a program that will be at the cutting edge of wireless education and technology. To reach this goal, the software radio project should be continued with a variety of experiments designed for undergraduate laboratory studies, and sufficiently detailed design information provided for use by graduate researchers.

Recommendations for Future Work

In continuation of this work, a WIT2410 module should be interfaced to each of the two Motorola MC68HC11 evaluation boards. The transmission of data from one to the other should be verified. Physical parameters, especially transmission distance, should be determined quantitatively. Possible interference between testsets should be considered, and eliminated by appropriate shielding if necessary. Failure modes of the components should be examined, and eliminated to the degree possible by suitable protective measures. A set of lab sheets should be developed describing the operation of the equipment, and suggesting various experiments that may be performed. In the long term, the development of a ruggedized in-house RF front-end may be considered in order to provide more accessibility to individual components for taking measurements. A specific example experiment is given in Appendix A-5.

APPENDIX A

CME9_EVBU

A.1 CME9-EVBU overview

The Axiom CME11E9-EVBU single board is a development system for the Motorola 68HC11 microcontrollers that has a wall plug power supply and a serial cable. The board layout is shown in the figure below. For more detail, see the User Manual [37].

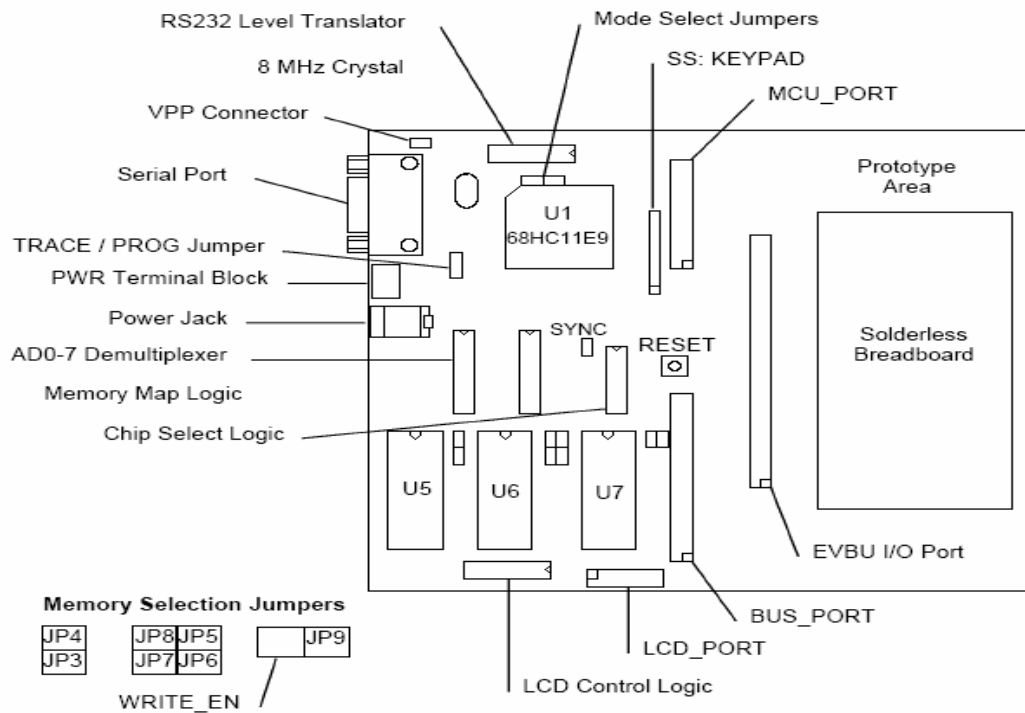


Figure A.1- CME11E9-EVBU board

Single chip mode is used for this experiment. To set the 68HC11 in Single-Chip mode with the EVBU boards' MODA jumper must be ON, MODB jumper must be OFF, and VCC must feed to PE0 pin after downloading the program into the board. To download the program the following procedure has been used:

```
>load t      <press enter>
              < select filename .s19>
>done       <done will be displayed after the program finishes loading>
>g         <to run the program>
```

More information about operating the 68HC11 board is provided in [38.]

A.2 Port Connection

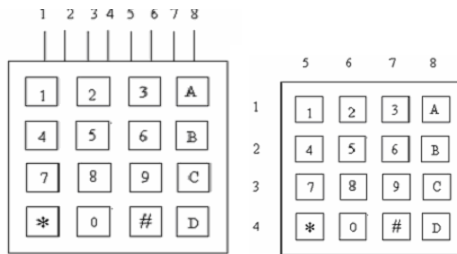


Figure A.2- Key Pad row and column pins out

Key pad operation: when key is pressed, the row and column pins are shorted together

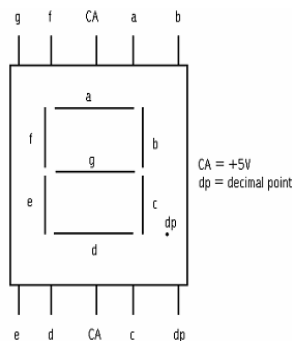


Figure A.3- 7-segment LED pins out

Table A-1 Ports Connection

PortC connection	Keypad Pins	PortB connection	LED Pins
PC0	1	PB0	g
PC1	2	PB1	f
PC2	3	PB2	e
PC3	4	PB3	d
PC4	5	PB4	c
PC5	6	PB5	b
PC6	7	PB6	a
PC7	8	PB7	NC

A.3 HC11 CODE

A.3.1 Transmitter SCI code

```

*-----*
* Overview
* Use the Serial Control Interface (SCI) to
* communicate with another 68HC11 EVBU, using
* differing bauds, and paritys
*-----*
* Transmitter
*-----*
* Global Variables
*-----*
Registers      EQU          $1000          ; bases address
Stack          EQU          $30           ; stack pointer at $30 hex
*-----*
* BUFFALO Subroutines
*-----*
WriteStr       EQU          $FFC7         ;prompt string for user command
InChar        EQU          $FFCD         ;store prompt command
*-----*
* SCI Registers and Variables
*-----*

SPCR           EQU          $28
BaudReg        EQU          $2B          ;baud rate register
SCCR1          EQU          $2C          ;SCI control register 1
SCCR2          EQU          $2D          ;SCI control register 2
SCSR           EQU          $2E          ;SCI status register
SCDR           EQU          $2F          ;SCI data register
DDRC           EQU          $07          ;configure port C direction
DDRD           EQU          $09          ;configure port D direction
PORTC          EQU          $03          ;port C address
    
```

```

PORTB      EQU          $04                ;port B address
TDRE       EQU          %10000000         ;transmitter notification
RDRF       EQU          %00100000         ;reciever notification
IDLE       EQU          %00010000         ;idle line detected flag
TC         EQU          %01000000         ;Transmit complete flag
TE         EQU          %00001000         ;Transmit enable
RE         EQU          %00000100         ;Receiver enable
T8         EQU          %01000000         ;Transmit bit 8
R8         EQU          %10000000         ;Receive bit 8
SEVEN5     EQU          $3A98             ;Bounce delay value
*-----
* Variables: use for storage purpose
*-----

BaudRate   RMB          1                ;reserve baud rate space
Temp       RMB          1                ;miscellenous
Parity     RMB          1                ;reserve parity space
C          RMB          1                ;reserve column value
R          RMB          1                ;reserve row value

*****
* PROGRAM START:  the character of key-pad is stored at address $00 while at address $100
*                 the stack value of $30 is loaded to allocated the stack space
*                 then the program is began to set up the baud rate AskBd subroutine
*                 the main purpose of this program is to transmit certain value of key pressed to display
*                 in 7-segment display; this program can be used as an experiment for transmitter to
*                 WIT2410 with other WIT2410 as a receiver
*
*****

CHAR       ORG          $00
           FCB $30,$6D,$79,$77,$33,$5B,$5F,$1F,$70,$7F,$73,$4E,$01,$7E,$01,$3D

           ORG          $0100
           LDS          #Stack            ;begin to load stack at $30

*-----
* Query for Baud:  prompt the baud rate from the user
*-----

AskBd     LDX          #AskBaud           ;display message for baud rate operation
           JSR          WriteStr          ;go to Write String subroutine in Buffalo
AB2       JSR          InChar             ;store the value in ACCA
           CMPA        #$30              ;compare the value of user enter with 0
           BLO         AB2               ;if the user's value below 0 go back to AB2
           CMPA        #$36              ;compare the value of user enter with 6
           BGT         AB2               ;if above 6 go back
           STAA        BaudRate          ;store the value in BaudRate memory

*-----
* Query for Parity:  prompt the parity dectection from the user
*-----

AskPar    LDX          #AskParity        ;display prompt message
           JSR          WriteStr          ;go to write string in buffalo
AP2       JSR          InChar             ;store in ACCA
           CMPA        #$31              ;parity bit 0/1
           BEQ         SetPar            ;if equal to 1 parity is set
           CMPA        #$30              ;parity bit 0
           BEQ         SetPar            ;if equal to 0 parity is set to 0

```

```

SetPar      BRA      AP2      ;if not 0/1 go back
           ANDA     #%00001111 ;and ACCA with $0F
           STAA     Parity    ;store in Parity reserve memory

*-----
* Query For Reset: prompt for reset operation since this program is operated with reset button is pressed
*-----
           LDX      #AskReset ;allert for reset to press
           JSR      WriteStr  ;when reset is pressed prg will go $b600
           STOP

*****
* MAIN PROGRAM START: this program will start to begin SCI communication between two boards
*****
*-----
* Program Will Jump Here When RESET is pressed
*-----
           ORG      $B600
           LDS      #Stack
           LDX      #Registers ;load X with base address
           JSR      Init      ;begin the process
           JSR      SetTxParity ;jump to set transmit parity subroutine
           JSR      IRQ_Send  ;jump to set IRQ for sending information
           SWI      ;software interrupt

*****
* Name:   Init
* Function: Set BAUD rate to value base on user selection
*****

Init

*-----
* Set Lower part of porTC as Input and Upper part as output
*-----
           LDAA     #$F0      ;set upper half of port C as output
           STAA     DDRC,X    ;store to Port C directional adrs

*-----
* Set TxD an output
*-----
           LDAA     #$02
           STAA     DDRD,X    ;store to Port D directional adrs

*-----
* Baud Rate Modification
*-----
           LDAA     BaudRate  ;load Baud rate value of user
           STAA     BaudReg,X ;store to baud rate register

*-----
* Enable Transmit
*-----
           LDAA     #$08      ;load $08 to put into transmitter address
           STAA     SCCR2,X   ;store the value to transmit register 2
           RTS              ;return from the initialize subroutine

```

* Name: IRQ_Send

* Function: Transmit Prompt IRQ

IRQ_Send

CLM0	LDX	#Registers	;load register X with \$1000
	CLRA		;clear ACCA
	STAA	C	;initialize C
	LDAA	#\$EF	;load ACCA with \$EF for port C
	STAA	PORTC,X	;send 11101111 to port c
	BRCLR	PORTC,X,\$01,R0	;check Row 0 if it is low
	BRCLR	PORTC,X,\$02,R1	;check Row 1 if it is low
	BRCLR	PORTC,X,\$04,R2	;check Row 2 if it is low
	BRCLR	PORTC,X,\$08,R3	;check Row 3 if it is low
CLM1	LDAA	#1	;load ACCA with value 1
	STAA	C	;store to C space
	LDAA	#\$DF	;load ACCA with \$DF
	STAA	PORTC,X	;send 11011111 to port c
	BRCLR	PORTC,X,\$01,R0	
	BRCLR	PORTC,X,\$02,R1	
	BRCLR	PORTC,X,\$04,R2	
	BRCLR	PORTC,X,\$08,R3	
CLM2	LDAA	#2	
	STAA	C	
	LDAA	#\$BF	
	STAA	PORTC,X	;send 10111111 to port c
	BRCLR	PORTC,X,\$01,R0	;check Row 0 if it is low
	BRCLR	PORTC,X,\$02,R1	;check Row 1 if it is low
	BRCLR	PORTC,X,\$04,R2	;check Row 2 if it is low
	BRCLR	PORTC,X,\$08,R3	;check Row 3 if it is low
CLM3	LDAA	#3	
	STAA	C	
	LDAA	#\$7F	
	STAA	PORTC,X	;send 01111111 to port c
	BRCLR	PORTC,X,\$01,R0	
	BRCLR	PORTC,X,\$02,R1	
	BRCLR	PORTC,X,\$04,R2	
	BRCLR	PORTC,X,\$08,R3	
	BRA	CLM0	
R0	CLRA		;clear ACCA
	STAA	R	;store to Row 0
	BRA	CAL	;jump to CAL
R1	LDAA	#1	;load ACCA with value 1
	STAA	R	;store to Row 1
	BRA	CAL	;jump to CAL
R2	LDAA	#2	;load ACCA with value 2
	STAA	R	;store to Row 2
	BRA	CAL	;jump to CAL
R3	LDAA	#3	;load ACCA with value 3

```

                STAA      R                ;store to Row 3
                BRA      CAL              ;jump to CAL

CAL             JSR      BOUNCE           ;label CALL is to transmit value
                LDY      #CHAR
                LDAA     R                ;load ACCA with Row value
                ASLA                     ;arithmetic shift left
                ASLA                     ;arithmetic shift left
                LDAB     C                ;load ACCB with Column value
                ABA                     ;add ACCB to ACCA
                TAB                     ;transfer ACCA to ACCB
                ABY                     ;add ACCB to index Y
HERE           BRCLR    SCSR,X,TDRE,HERE ; is transmit data reg empty?
                NOP                     ;delay for 2 cycl
                LDAA     0,Y              ;load ACCA with Y-index pointer
                STAA     SCDR,X          ;store ACCA value to SCDR reg
                JMP      CLM0            ;jump to Column 0
                RTS                     ;return from subroutine

```

```

*-----
*Name: BOUNCE is a delay routine to make sure the key is pressed properly from user
*Function: make sure the key is pressed
*-----

```

```

BOUNCE         PSHX                     ;reserve X-index to be used
                LDX      #SEVEN5         ;load delay number
HOLD           NOP                       ;idle for 2 cycl
                NOP
                DEX                     ;decrement index-X by 1
                BNE     HOLD             ;check if X=0 if not go to HOLD
                PULX                    ;get back X-index value
                RTS                     ;return from subroutine

```

```

*****
* Name: CalcPar is to calculate the parity value from the user enter 8bits
* Function: Calculate Parity
*****

```

```

CalcPar

                STAA     Temp            ;store ACCA to Temp
                LDY     #$0              ;load Y-index with 0
                CLRB                    ;clear ACCB
CP1            INY                       ;increment Y-index by 1
                CPY     #$8              ;compare Y to $8
                BEQ     CPDone           ;if equal $8 go to CPDone
                LSR     Temp             ;logic shift right Temp value
                BCS     ChkPar           ;go to check parity if carry bit is set
                BRA     CP1              ;otherwise to back to PC1
ChkPar         CMPB    #$00              ;compare ACCB to 0
                BEQ     AddPar           ;if 0, go to increment ACCB by 1
                DECB                    ;otherwise decrement by 1
                BRA     CP1              ;always go to CP1
AddPar         INCB                    ;increment ACCB by 1
                BRA     CP1              ;always go to CP1
CPDone        RTS                     ;return from subroutine

```

```

*****
* Name: SetTxParity is a subroutine to set transmit parity according to the user selection
* Function: Set Transmit Parity
*****
SetTxParity
                JSR          CalcPar                ;jump to CalcPar subroutine
*-----
* ACCB=0 (Even) ACCB=1 (Odd)
*-----
SetTxPar       CMPB        Parity                ;compare ACCB to parity value
                BEQ          ClrT8                ;if equal then go to ClrT8 label
SetT8          BSET        SCCR1,X,T8           ;set SCCR1 reg to T8
                BRA          STPDone             ;otherwise finish
ClrT8          BCLR        SCCR1,X,T8           ;clear the SCCR1 flag
STPDone        RTS
                ;return from subroutine

```

```

*****
* Data: this section is used for data display on the screen for user making a selection
*****
Prompt         FCB          "Ratha> ",$FF
AskBaud        FCB          "Baud: 9600 (=0); 4800 (=1); 2400 (=2); "
                FCB          "1200 (=3); 600 (=4); 300 (=5); 150 (=6)", $04
AskParity      FCB          "Parity: Even (=0); Odd (=1)", $04
AskReset       FCB          "Set Jumper to PE0; Press Reset", $04

```

A.3.2 Receiver SCI Code

```

*-----*
* Overview
* Use the Serial Control Interface (SCI) to
* communicate with another 68HC11 EVBU, using
* different bauds, and parity
*-----*
* Receptor
*-----*
* Global Variables
*-----*

Registers      EQU          $1000
Stack          EQU          $30

*-----*
* BUFFALO Subroutines
*-----*

WriteStr       EQU          $FFC7                ;buffalo command
InChar         EQU          $FFCD                ;buffalo command

```

```

*-----*
* BUFFALO Output Port
*-----*
PortB      EQU      $04                ;output port B
*-----*
* SCI Registers and Variables
*-----*

SPCR       EQU      $28
BaudReg    EQU      $2B
SCCR1      EQU      $2C
SCCR2      EQU      $2D
SCSR       EQU      $2E
SCDR       EQU      $2F
TDRE       EQU      %10000000
RDRF       EQU      %00100000
IDLE       EQU      %00010000
TC         EQU      %01000000
TE         EQU      %00001000
RE         EQU      %00000100
T8         EQU      %01000000
R8         EQU      %10000000

*-----*
* Data to Receive
*-----*

*****
* PROGRAM START: the program section is to prompt inputs from user
*                 to set up the communication between the two boards
*                 such as parity detection, baud rate selection, and word length
*****

                ORG      $0100
                LDS      #Stack
*-----*
* Query for Baud
*-----*

AskBd        LDX      #AskBaud          ;prompt baud rate from user
                JSR      WriteStr       ;display the options for user
AB2          JSR      InChar            ;store user value in ACCA
                CMPA     #$30           ;compare ACCA with $30
                BLO      AB2            ;if result is below go to AB2
                CMPA     #$36           ;compare ACCA with $36
                BGT      AB2            ;if result is greater go to AB2
                STAA     BaudRate       ;store value ACCA to baud rate
*-----*
* Query for Parity
*-----*

AskPar       LDX      #AskParity        ;prompt user input for parity
                JSR      WriteStr       ;display options to user
AP2          JSR      InChar            ;store user value to ACCA
                CMPA     #$31           ;compare ACCA with $31
                BEQ      SetPar         ;if result is equal go to set parity
                CMPA     #$30           ;compare ACCA with $30

```

```

                BEQ      SetPar      ;if result is equal go to set parity
                BRA      AP2         ;if not go to AP2
SetPar          ANDA     #%00001111 ;and ACCA with $0F
                STAA    Parity      ;store ACCA to parity

```

```

*-----

```

```

* Query For Reset

```

```

*-----

```

```

                LDX     #AskReset    ;prompt user input for reset
                JSR     WriteStr     ;display option and get value
                STOP                    ;stop the program

```

```

*-----

```

```

* Variables

```

```

*-----

```

```

BaudRate       RMB      1           ;allocate baud rate memory
Temp           RMB      1           ;allocate temporary register
Char           RMB      1           ;allocate character space
Parity         RMB      1           ;allocate parity space

```

```

*****

```

```

* PROGRAM START: this program will display back to the user to any key was pressed to 7-segment
*                 display. It can be used with wit2410 on the receiver part. The program performs
*                 similarly to transmit part. The only different is that it will enable the receive
*                 register in other board to obtain information from the transmitter board.

```

```

*****

```

```

*-----

```

```

* Program Will Jump Here When RESET

```

```

*-----

```

```

                ORG     $B600        ;B600 is the address for Buffalo
                LDS     #Stack       ;load stack to allocate space

```

```

*-----

```

```

* Process Begins Here

```

```

*-----

```

```

                JSR     Init         ;jump to initialize subroutine
                JSR     RecvStr     ;jump to receive string subroutine

```

```

*****

```

```

* SUBROUTINES

```

```

*****

```

```

*****

```

```

* Name: Init

```

```

* Function: Set BAUD and Mode

```

```

*****

```

```

Init

```

```

                LDX     #Registers   ;load index X with reg address

```

```

*-----
* Baud Rate Modification
*-----
                LDAA      BaudRate      ;load ACCA to Baud rate
                STAA      BaudReg,X     ;store ACCA to Baud rate register
*-----

* Enable Receive
*-----
                LDAA      #$04          ;load ACCA with $04
                STAA      SCCR2,X       ;store ACCA to SCCR2 reg
                JSR       SetRxParity   ;jump to set receiver parity routine
                RTS                    ;return from subroutine initialize

*****
* Name: RecvStr—this subroutine is to put back the string from transmitter into output port for later
*         to display on the 7-segments LED
* Function: Receive String
*****
RecvStr

                LDX      #Registers     ;load index_X with register value
Loop           NOP                    ;idle for 2 clock cycle to avoid
                                         ;missing data
RS1           BRCLR     SCSR,X,RDRF,RS1 ;branch to RS1 if RDR is clear
*-----
* Clear Status Flag
*-----
                LDAA      SCDR,X        ;load SCDR value into ACCA
                STAA      PORTB,X       ;store ACCA into portB
                BRA       Loop          ;branch always to loop
*****
* Name: CalcaPar—this subroutine is performed parity setting base on user request
* Function: Calculate Parity
*****
CalcPar

                STAA      Temp          ;store ACCA value in Temp
                LDY      #$0            ;compare index-Y to $0
                CLRB
CP1           INY                    ;clear ACCB for future increment
                CPY      #$8           ;increment index-Y by 1
                BEQ      CPDone         ;compare index-Y to value $8
                LSR      Temp          ;if result is equal go to CPDone
                BCS      ChkPar         ;otherwise shift temp to the left
                BRA      CP1           ;branch to ChkPar if carry bit is set
ChkPar       CMPB      #$00           ;branch always to CP1
                BEQ      AddPar         ;compare ACCB to $00
                DECB
                BRA      CP1           ;if result is equal go to AddPar
AddPar       INCB
                BRA      CP1           ;decrement ACCB by 1
CPDone      RTS                    ;branch always to CP1
                                         ;increment ACCB by 1
                                         ;branch always to CP1
                                         ;return from subroutine

```

```

*****
* Name: SetRxParity—this subroutine is to set the parity according from the user request
* Function: Set Receive Parity
*****
SetRxParity

                PSHY                ;reserve index-Y value
                JSR                  CalcPar                ;jump to CalcPar routine
*-----*
* ACCB=0 (Even) ACCB=1 (Odd)
*-----*
SetRxPar        CMPB                Parity                ;compare ACCB to parity value
                BEQ                  ClrR8                ;if result equal go to ClrR8
SetR8           BSET                SCCR1,X,R8            ;otherwise set SCCR1 R8
                BRA                  SRPDone              ;branch always SRPDone
ClrR8           BCLR                SCCR1,X,R8            ;clear SCCR1 with R8 value
SRPDone        PULY
                RTS
                ;return from subroutine

*****
* Data—display to the screen for the user
*****
Prompt         FCB                  "Ratha> ",$FF
AskBaud        FCB                  "Baud: 9600 (=0); 4800 (=1); 2400 (=2); "
                FCB                  "1200 (=3); 600 (=4); 300 (=5); 150 (=6)", $04
AskParity      FCB                  "Parity: Even (=0); Odd (=1)", $04
AskReset       FCB                  "Set Jumper to PE0; Press Reset", $04

*****

```

A.4 HC11 Code testing for different Baud Rate

The code in this section is similar to appendix A-3, but a modify version to make possible for baud rate measurement in logic analyzer. Transmitter and receiver code are integrated in once. The program will provide the user to enter (0) for transmitting and (1) for receiving.

```

*-----*
* Global Variables
*-----*

Registers      EQU                  $1000
Stack          EQU                  $30

*-----*
* BUFFALO Subroutines
*-----*

WriteStr       EQU                  $FFC7
InChar         EQU                  $FFCD

```

```

*-----*
* SCI Registers and Variables
*-----*

```

```

SPCR      EQU      $28
BaudReg   EQU      $2B
SCCR1     EQU      $2C
SCCR2     EQU      $2D
SCSR      EQU      $2E
SCDR      EQU      $2F
TDRE      EQU      %10000000
RDRF      EQU      %00100000
IDLE      EQU      %00010000
TC         EQU      %01000000
TE         EQU      %00001000
RE         EQU      %00000100
T8         EQU      %01000000
R8         EQU      %10000000
PortB     EQU      $04

```

```

*-----*
* Data to Transmit/Receive
*-----*

```

```

*****

```

```

* PROGRAM START

```

```

*****

```

```

                ORG      $0100
                LDS      #Stack

```

```

*-----*
* Query for Transmit or Receive Mode
*-----*

```

```

                LDX      #AskMode           ;prompt mode from user
                JSR      WriteStr          ;jump to write subroutine of buffalo
AskM            JSR      InChar            ;store value into ACCA
                CMPA    #$30              ;compare ACCA to 0
                BEQ     SaveMode          ;if equal then save mode value
                CMPA    #$31              ;compare ACCA to 1
                BEQ     SaveMode          ;if equal go to save mode
                BRA     AskM              ;otherwise always jump to askM
SaveMode        STAA    Mode              ;store ACCA value into Mode

```

```

*-----*
* Query for Word Length
*-----*

```

```

                LDX      #AskLength        ;prompt word length from user
                JSR      WriteStr          ;jump to buffalo subroutine
TR1            JSR      InChar            ;store value into ACCA
                CMPA    #$30              ;compare ACCA with $30
                BEQ     SetLen            ;if ACCA equal to zero go to setlen
                CMPA    #$31              ;compare ACCA with $31
                BEQ     SetLen            ;if ACCA equal go to SetLen
                BRA     TR1              ;jump always to TR1
SetLen         STAA    Length            ;store ACCA into length

```



```

*-----
* Query for Baud
*-----
AskBd      LDX      #AskBaud      ;prompt baud rate from user
           JSR      WriteStr    ;display the string to user
AB2        JSR      InChar      ;store user value to ACCA
           CMPA     #$30        ;compare ACCA with $30
           BLO     AB2          ;if result is below go back to AB2
           CMPA     #$36        ;compare ACCA to $36
           BGT     AB2          ;if result is greater go to AB2
           STAA    BaudRate     ;otherwise store to BaudRate
*-----
* Query for Parity
*-----
AskPar     LDX      #AskParity   ;prompt parity value
           JSR      WriteStr    ;display option for user
AP2        JSR      InChar      ;store user value in ACCA
           CMPA     #$31        ;compare ACCA to $31
           BEQ     SetPar      ;if result is equal go to set parity
           CMPA     #$30        ;compare ACCA with $30
           BEQ     SetPar      ;if result is equal go to set parity
           BRA     AP2          ;jump always to AP2
SetPar     ANDA    #%00001111   ;and result with $0F
           STAA    Parity      ;store result to Parity
*-----
* Query For Reset
*-----
           LDX      #AskReset   ;prompt for reset pressed
           JSR      WriteStr    ;display the option to user
           STOP     ;stop the program
*-----
* Variables
*-----
Buffer     RMB     20          ;allocate memory for buffer
BaudRate   RMB     1           ;allocate memory for baud rate
Mode       RMB     1           ;allocate memory for mode
Temp       RMB     1           ;allocate memory for temporary
Char       RMB     1           ;allocate memory for character
Parity     RMB     1           ;allocate memory for parity
Length     RMB     1           ;allocate memory for word lenght
*****
* PROGRAM START: this program is to initialize the baud rate, parity, and world length correspond to the
* user request
*****
*-----
* Program Will Jump Here When RESET
*-----
           ORG     $B600        ;buffalo address
           LDS     #Stack      ;load stack for allocation
*-----
* Process Begins Here
*-----
           JSR     Init         ;jump to initialization routine
           STOP     ;execute the program from initialize

```

```

*****
*           SUBROUTINES
*****
*****
* Name:    Init
* Function: Set BAUD and Mode
*****

Init
      LDX          #Registers          ;load index X with register value
*-----
* Baud Rate Modification
*-----
      LDAA         BaudRate            ;load ACCA from baud rate
      STAA         BaudReg,X          ;store ACCA value to baud reg

*-----
* Enable Both Transmit and Receive
*-----
      LDAA         #$0C                ;load ACCA with value $0C
      STAA         SCCR2,X            ;store ACCA to SCCR2

*-----
* Transmit or Receive Selection
*-----
      LDAA         Mode                ;load ACCA with Mode
      CMPA         #$30                ;Transmit Mode
      BEQ          Xmit                ;if equal to zero then transmit
Recv   JSR         RecvStr             ; Receive Mode
      BRA         InitDone            ;always go to initDone
Xmit   JSR         SendStr            ;jump to transmit mode
InitDone RTS                          ;get out of subroutine init

*****
* Name:    SendStr
* Function: Transmit Prompt And String
*****

SendStr
      LDX          #Registers          ;load index-X with register value
      JSR         SetWL                ; Set Word Length

*-----
* Send Prompt
*-----
SS0    LDY         #Prompt             ;load index-Y with input prompt
SS1    LDAA         0,Y                ;ACCA point to index[0] of Y addr
      CMPA         #$FF                ;Terminator by comparing ACCA
      BEQ          SS3                 ;if ACCA = $FF go to SS3
      JSR         SetTxParity          ;jump to subroutine SetTxParity
      STAA         SCDR,X              ;store ACCA to SCDR reg
SS2    BRCLR       SCSR,X,TDRE,SS2    ;branch to SS2 if TDRE bit is clear
      INY                          ;otherwise increment Y index
      BRA         SS1                 ;always branch to SS1

```

```

*-----
* Send String to transmitter
*-----
SS3      LDY      #XmitString      ;load index-Y with XmitString
SS4      LDAA     0,Y              ;ACCA point to index[0] of Y reg
          CMPA    #$FF            ; Terminator by comparing ACCA
          BEQ     SSDone          ;if equal then go to SSDone
          JSR     SetTxParity     ;otherwise set TxParity
          STAA   SCDR,X          ;store ACCA to SCDR
SS5      BRCLR   SCSR,X,TDRE,SS5 ;loop here if TDRE is not clear
          INY     ;otherwise increment index Y by 1
          BRA    SS4            ;always branch to SS4

```

```

*-----
* Clear Status Flag: reset the flag after word is transmitted
*-----

```

```

          STAA   SCDR,X          ;store ACCA to SCDR
SSDone   RTS                    ;return from the subroutine

```

```

*****

```

```

* Name: SetWL—this function is to store user word length selection

```

```

* Function: Set Word Length

```

```

*****

```

```

SetWL

```

```

          LDAA   Length          ; Set Word Length
          CMPA   #$30           ;compare ACCA to $30
          BEQ   SetBit8         ;if equal go to set 8 bit
SetBit9   BSET   SCCR1,X,T8     ;if not 8 bit then set to 9 bit
          BRA   SWEnd           ;always go to return from routine
SetBit8   BCLR   SCCR1,X,T8     ;clear SCCR1 at T8 bit
SWEnd     RTS                    ;return from subroutine

```

```

*****

```

```

* Name: RecvStr—this function is to receive the information from the user

```

```

* Function: Receive String

```

```

*****

```

```

RecvStr

```

```

          LDX    #Registers     ; Set Word Length
          JSR    SetWL          ; Set Word Length
ResetBuf  LDY    #Buffer        ;load index-Y with buffer value
RS1      BRCLR  SCSR,X,RDRF,RS1 ;if RDRF not full loop here

```

```

*-----
* Clear Status Flag
*-----

```

```

          LDAA   SCDR,X          ;load ACCA with SCDR
          JSR    SetRxParity     ;jump to SetRxRarity
          STAA   0,Y            ;store ACCA in to index-Y
          STAA   PortB,X        ;store ACCA into port B to disp
          INY     ;increment for the next Y
          CPY    #Buffer+21     ;compare to see if Buffer end
          BLO    RSBranch       ;if not finish go to RSBranch
          BRA    ResetBuf       ;otherwise always go to ResetBuf
RSBranch  BRA    RS1            ;always jump to RS1
RSDone    RTS                    ;return from subroutine

```

```

*****
* Name: CalcPar—this function is to determine if bit 0/1 parity is detected
* Function: Calculate Parity
*****

```

CalcPar

```

                STAA      Temp                ;store ACCA in temp
                LDY      #$0                 ;load index-Y with value 0
                CLRB                    ;clear ACCB
CP1             INY                    ;increment Y-index
                CPY      #$8                 ;compare Y-value with $8
                BEQ      CPDone             ;if equal go to CPDone
                LSR      Temp              ;logic shift left Temp
                BCS      ChkPar            ;branch to ChkPar if carry bit is set
                BRA      CP1               ;branch always to CP1
ChkPar         CMPB      #$00              ;compare ACCB with $00
                BEQ      AddPar            ;if equal go to AddPar
                DECB                    ;decrement ACCB by 1 unit
                BRA      CP1               ;branch always to CP1
AddPar         INCB                    ;increment ACCB by 1 unit
                BRA      CP1               ;branch always to CP1
CPDone        RTS                    ;return from subroutine

```

```

*****
* Name: SetTxPar—this function is set the transmit parity as selection of user
* Function: Set Transmit Parity
*****

```

SetTxParity

```

                PSHY                    ;reserve index-Y for future used
                JSR      CalcPar           ;jump to subroutine CalcPar
*-----
* ACCB=0 (Even) ACCB=1 (Odd)
*-----
SetTxPar       CMPB      Parity           ;compare ACCB to Parity
                BEQ      ClrT8            ;if equal go to ClrT8
SetT8          BSET      SCCR1,X,T8       ;otherwise set SCCR1 at T8 bit
                BRA      STPDone          ;branch always to finish
ClrT8          BCLR      SCCR1,X,T8       ;clear SCCR1 at T8 bit
STPDone        PULY                    ;pull back reserve Y-value
                RTS                    ;return from subroutine

```

```

*****
* Name: SetRxParity—this function is to set the parity bit for receiver unit as user selection
* Function: Set Receive Parity
*****

```

SetRxParity

```

                PSHY                    ;reserve index-Y address
                JSR      CalcPar           ;jump to subroutine CalcPar
*-----
* ACCB=0 (Even) ACCB=1 (Odd)
*-----
SetRxPar       CMPB      Parity           ;compare ACCB with parity value
                BEQ      ClrR8            ;if equal go to clrR8
SetR8          BSET      SCCR1,X,R8       ;otherwise set SCCR1 at R8 bit

```

	BRA	SRPDone	;branch always to SRPDone
ClrR8	BCLR	SCCR1,X,R8	;clear SCCR1 at R8 bit
SRPDone	PULY		;pull reserve Y-index address
	RTS		;return from subroutine

* Data

*\$30,\$6D,\$79,\$77,\$33,\$5B,\$5F,\$1F,\$70,\$7F,\$73,\$4E,\$01,\$7E,\$01,\$3D

Prompt	FCB	"Ratha> ", \$FF
XmitString	FCB	"~0myw3[-psN=myw30myw3[psN=0myw3sN=ym", \$FF
AskMode	FCB	"Mode: Xmit (=0) ; Receive (=1)", \$04
AskLength	FCB	"Length: 8-bit (=0) ; 9-bit (=1)", \$04
AskBaud	FCB	"Baud: 9600 (=0); 4800 (=1); 2400 (=2); "
	FCB	"1200 (=3); 600 (=4); 300 (=5); 150 (=6)", \$04
AskParity	FCB	"Parity: Even (=0); Odd (=1)", \$04
AskReset	FCB	"Set Jumper to J2; Press Reset", \$04

A.5 Serial Communications Interface with WIT2410

OBJECTIVES

The objectives of this experiment are to understand asynchronous serial communications of 68HC11 boards, and to use the board as processors with WIT2410 devices as transmitter/receiver.

MATERIALS

(2) Motorola MC68HC11EVBU Kit: this included a series port, a +5V power supply, and software CD

(AxIDE)

(2) 5-Volt Regulated Power Supply

(2) WIT2410 devices

(2) antennas

(1) keypad

(1) 7-segment LED

(2) 741s04

(4) 10k-ohm resistors

(7) 330-ohm resistors

INTRODUCTION

1. WIT2410

Refer to user manual in the Carronet's CD, it is given when buy the board, for specific details of setting up the boards and for the pinout of the WIT2410.

2. 68HC11

Refer to Chapter 9 of the Reference Manual for specific details of the SCI. The SCI is an independent serial I/O subsystem in the M68HC11. Port D pins TxD (PD1) and RxD (PD0) are used as Transmitter and Receiver data pins. The SCI subsystem is used primarily to connect a PC to the MCU and is configured/controlled by the following five registers:

BAUD Baud-RateControl Register

SCCR1 SCI Control Register 1

SCCR2 SCI Control Register 2

SCSR SCI Status Register

SCDR SCI Data Register

Read Section 9.2 of the Reference manual to understand the function of each register given above and how they relate to the SCI subsystem.

PROGRAM REQUIREMENTS

For this experiment, two programs will be created, one for each 68HC11 board. Program One will capture data from the 68HC11 user interface board by configured half of port C as inputs and another half as outputs, this will allow the key to scan properly (more detail of how to connect keypad to the board is given

in Appendix A.2), and transmit the data using a WIT2410 configured as a transmitter. The second WIT2410 configured as a receiver will wirelessly receive the corresponding user data. The data will be stored into the second 68HC11's memory by the SCI in port B. The WIT2410's should be configured for point to point operation. The transmit routine should allow the user to press the key pad and transmit the corresponding characters at different baud rates, different modes (refer to sscr1 reg.), and different parity settings (even or odd). The transmit program should contain four basic routines:

- (1) INIT_SCI - initialize all regs. that are involved with the SCI (ex. : baud, sscr1, sscr2).
- (2) TRANSMIT - should transmit a character out using the SCI. (if the MODE = 1, then the 9th bit should be a the parity bit (even or odd).)
- (3) PARITY - should calculate the parity of a byte (even or odd).
- (4) PRESS_KEY – should read input from user when key is pressed.

Program Two will receive the data from the transmitter board using the SCI. The receiving routine should allow the user to receive at different baud rates, different modes, and different parity settings. The program should also contain a circular buffer by which you should store the received character (if mode = 1, then all 9 bits). The receive program should contain three basic routines:

- (1) INIT_SCI - initialize all register that are involved with the SCI (ex: baud, sscr1, sscr2).
- (2) REC_CHAR - should read a character using the SCI (if the MODE = 1, then the 9th bit should be a parity (even or odd).)
- (3) DISPLAY_KEY – should outputting the key to 7-segment LED base on the key pressed from transmitter

Writing Program One and Two

- **Type** in source code using a text editor (e.g. **Notepad**).
- Save source code (**xxxx.asm**).

Compiling Program One and Two

- Install IASM11 onto computer (this program is available on line for free down load)
- Using the IASM11 Assembler
 - Double click on the **iasm11** icon. This start up the assembler code into the editor of the assembler.
 - Select **F3 (Load)**. Type in the file name (xxxx.asm) and hit Enter. This loads the source code into the editor of the assembler.
 - Select **F10 (Menu)**. **Move** the cursor to the **Assemble** (use the right arrow) and hit Enter. This brings up the Assemble submenu. Use the Enter Key to toggle **Listing** to **On**, **Object to S19**, and **Cycle Cntr to On**. These section prompt the assembler to
 - Generate a list file xxxx.lst
 - Generate object code in Motorola S-record format xxxx.S19
 - Include a cycle count with the program listing
 - Hit **F4 (Assemble)**. This assembles the program. If there are error in the program, they may be corrected using editor.
 - **Exit iasm 11** by hitting **F5**.
- Install AxIDE onto computer (this program is included when buying the 68HC11 board) then created a short cut name as AxIDE.

PROCEDURE

1. Connect power supply to all boards.
2. Connect a wire to common ground for all boards.
3. Down load the Transmit routine in the transmit board.
4. Down load the Receive routine in the receiver board.
5. To test programs: (at the monitor prompt)

(NOTE: Do NOT hit return after steps k and l.)

Testing 68HC11 board:

- (a) connected series port from the board to a computer
- (b) double click on AxIDE and select COM1 or COM2
- (c) press reset button on 68HC11 boards
- (d) type: mm 102b; this allow to change baud rate of the board if using start address at \$B600
(more detail regarding to using AxIDE can be found in the AxIDE's CD documentation)
- (e) type: 30; this allow 68HC11 board to operate at 300 baud rate when down load the code
- (f) press enter twice
- (g) type: load t <press enter>; then click on "upload" and select file.s19
- (h) after "done" is displayed on the screen, click on "red check mark" on the left to change the baud rate back to 9600; this is the default baud rate.
- (i) do the same to another board
- (j) type: go <starting address> on one terminal.
- (k) type: go <starting address> on second terminal.
- (l) connect a wire from transmit pin of the transmit routine board to the receive pin of the receive routine board (Pins found on Port D).
- (m) Press Enter on both terminals.
- (n) Press any key on keypad, then hit RESET on receive board.
- (o) Disconnect wire from step l.
- (p) Press RESET on the transmit board.
- (q) Examine characters transmitted.

Connecting 68HC11 Transmitter to WIT2410 Transceiver

- (1) Connect pin PD1 from 68HC11 of transmitter board to pin 2 of transmitter WIT2410.
- (2) Connect RF output of WIT2410 to an oscilloscope to make sure both devices are operating
- (3) Connect the RF output to antenna.

Connecting 68HC11 Receiver to WIT2410 Receiver

- (1) Connect pin 3 of WIT2410 to an oscilloscope to make sure that the signal is actually received
- (2) Connect pin 3 of WIT2410 Receiver to pin PD0 of 68HC11 receiver board.

APPENDIX B

AUSDR-1 OVERVIEW

This appendix provides an overview of the original SDR testset, AUSDR-1. The author has collaborated extensively on the design, construction, and testing of this testset. The experience thus gained led to the development of the second generation testset, AUSDR-2 reported in the main body of this thesis.

B.1 RF Front End Architecture

In this design, a super-heterodyne RF front end is used. Figure B.1 shows the RF front-end architecture with two up and down-conversion stages that consist of the whole picture of RFMD's WLAN chipset of wireless communication IEEE 802.11b standards. The chipset includes the RF2948b QPSK (quadrature phase-shift keying) modem, the RF5117, RF power amplifier, the RF2494 LNA/mixer, and the RF3000 baseband processor which is not utilized in the project. Since this RF front end is a quadrature modulator/demodulator that include two channels for transmission and reception, these two channels are used to transmit the data and synchronization bit-streams from the transmitter DSP (digital signal processing) to the receiver DSP. TI's DSPs feature is configure as master/slave (SPI) standard that is operated at 30 Mb/s. To aid in the code development process, Code Composer Studio's DSP/BIOS configuration utility was used to configure the McBSP (Multi-channel Buffered Serial Port). The transmitting McBSP is configured as a serial port master; therefore, it provides the clock and frame sync pulses to the receiving McBSP, which is configured as a slave. For result of the simulation, see Akhi's thesis.

The RF part of a wireless communication system generally provides a conversion between the radio or microwave frequency and base-band or intermediate frequencies (IF). This is called up and down-conversion, which is respectively related to up and down shifting in the frequency spectrum of the signal that needs to be received or transmitted. In AUSDR-1, a super-heterodyne RF front-end was used. Figure

B.1 shows the RF front-end architecture with two up and down-conversion stages based on RFMD's WLAN chipset for the IEEE 802.11b standard. The chipset includes the RF2948b QPSK (quadrature phase-shift keying) modem, the RF5117 RF power amplifier, the RF2494 LNA/mixer, and the RF3000 baseband processor, which was not utilized at this stage of the project (DSP's were used instead). The RFMD's WLAN chipset is geared toward the IEEE 802.11b by applying a DSSS method for the WLAN transceiver market. The input signals are the outputs of a digital to analog converter with proper peak-to-peak amplitude and DC bias.

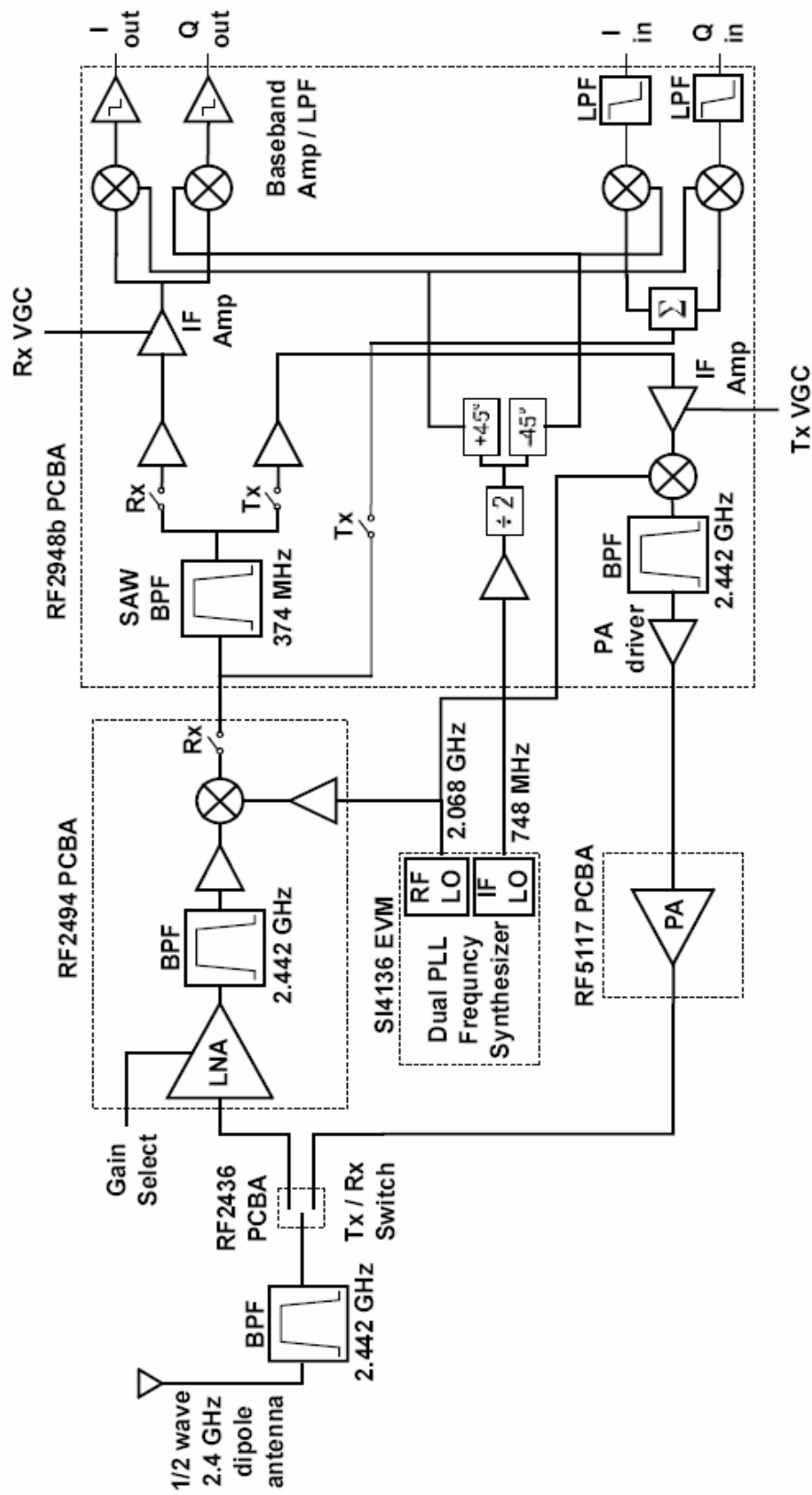


Figure B.1- RF front end design using FFMD WLAN chipset

B.2 DSP's for Baseband Processing

Two Texas Instruments DSP's were evaluated for use in AUSDR-1, the TMDS320C5416, and the TMDSC6711 DSK boards. The principle difference is that the '6711 is a floating-point machine, whereas the '5416 has a fixed-point processor.

B.3 Code Development

DSP code was written using Code Composer Studio. The author wrote the majority of the code that was used to perform the baseband processing for AUSDR-1.

B.4 Results from AUSDR-1

The experimental setup and results for AUSDR-1 is shown below in Fig'. B.2 through B.6.

The RF front end was successfully built and tested. Figures B.2 and B.3 are diagrams of the RF Receiver front-end and RF transmitter front-end circuit respectively.

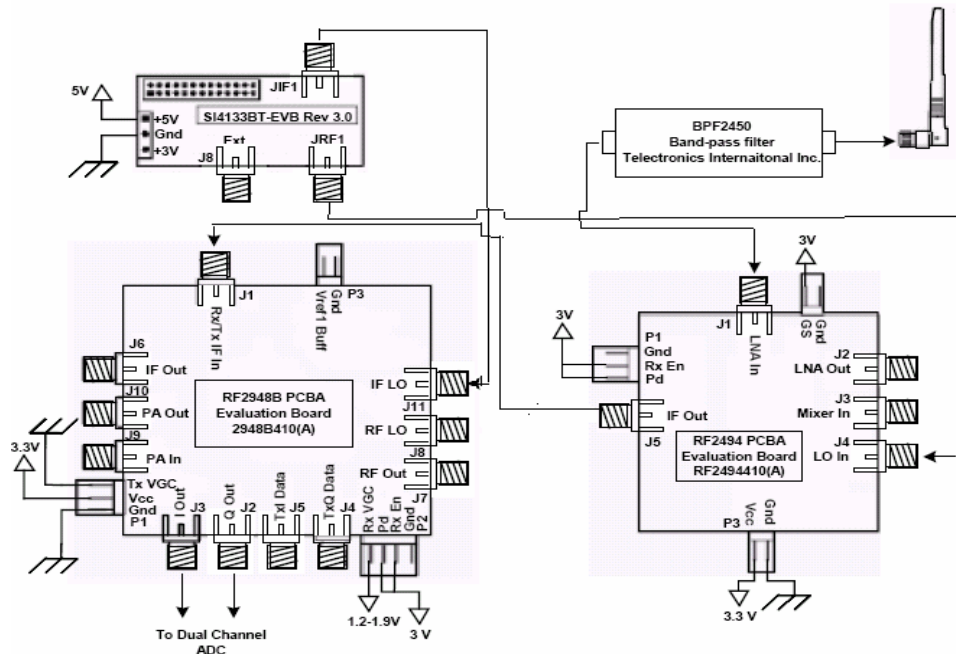


Figure B.2- RF Receiver front-end diagram

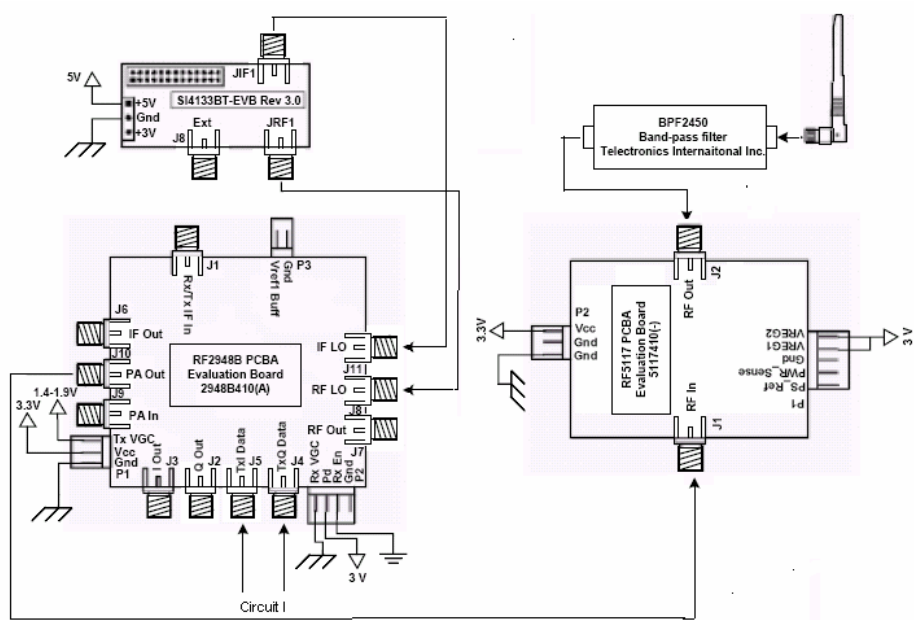


Figure B.3- RF Transmitter front-end diagram

Figures B.4 and B.5 is illustrated the transmitter and receiver hardware. Figure B.6 is displayed the transmission of two sinusoids through the I and Q channels of the front end during testing with an input signal having an amplitude of 100mV and a frequency of 100k Hz.

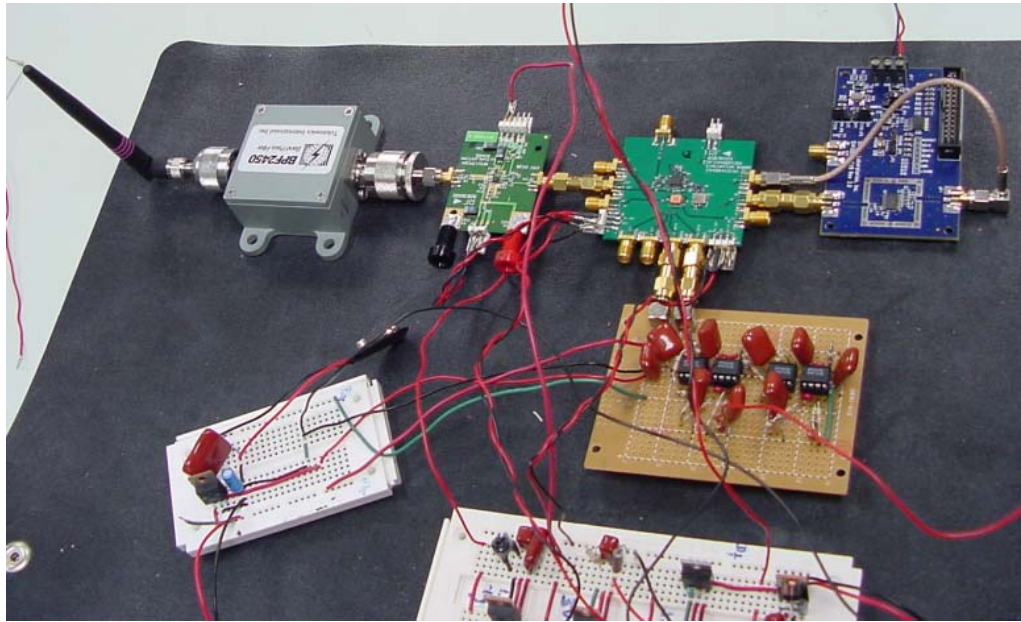


Figure B.4- RF transmitter photo

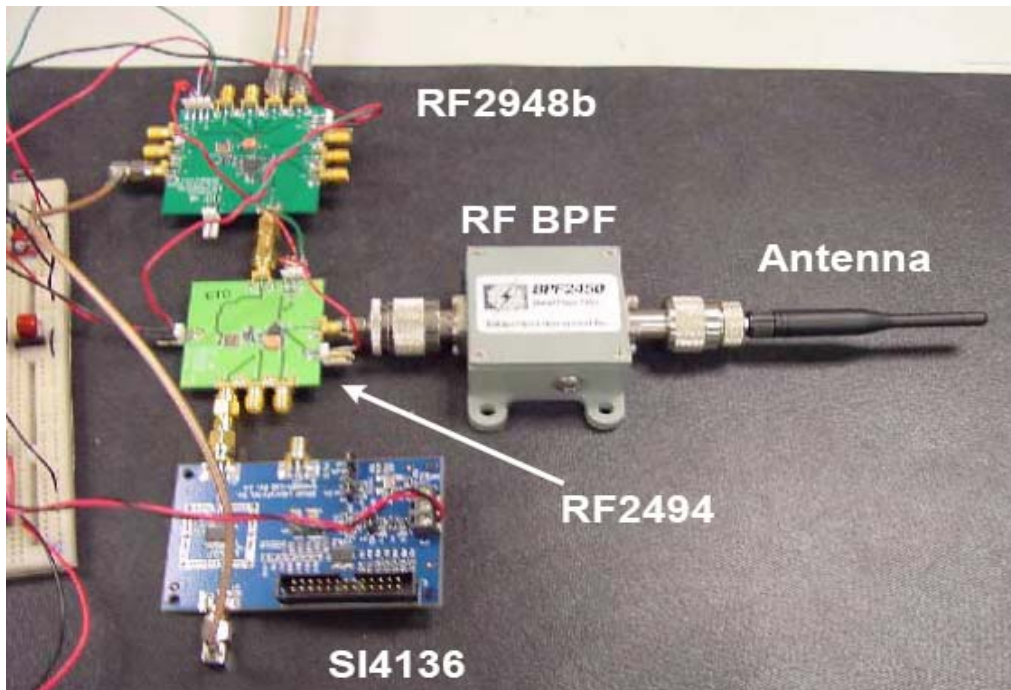


Figure B.5- RF receiver photo

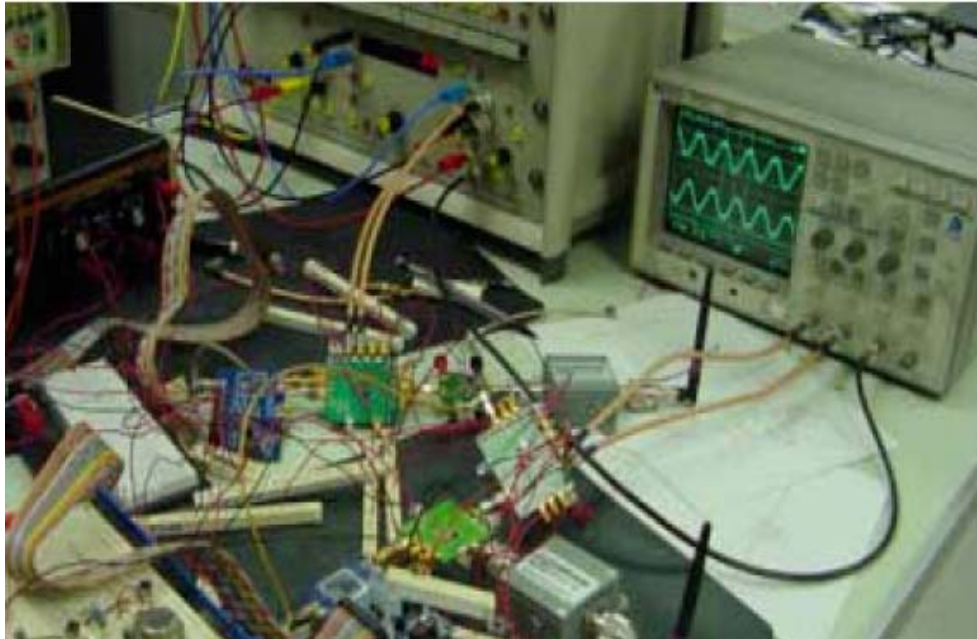


Figure B.6- Signal received through RF link

During the construction of AUSDR-1, several of the RFMD boards failed and had to be replaced before final testing could be performed. After all devices were connected, AUSDR-1 functioned for only a very short duration. After operating for approximately 1 hour, the DSP boards failed, as well as the RF2948b. ESD sensitivity was determined to be the primary culprit.

BIBLIOGRAPHY

- [1] L. Doyle, P. Mackenzie, D. O'Mahony, K. Nolan, D. Flood, "A General Purpose Processor Component Based Software Radio Engine", Proceedings of the Second European Colloquium on Reconfigurable Radio, June 2002
- [2] M. Coyne and G. Desjardins, "RF Performance of Dual-Mode Handsets," *Technical Report for Sony of Canada*, Aug. 1998.
- [3] Buracchini, E., SORT & SWRADIO concept <http://www.ifn.et.tu-dresden.de/~sort/>
- [4] Brad Brannon, and Chris Cloninger, "Redefining the Role of ADCs in Wireless," *Applied Microwave & Wireless*, March 2001, Page(s): 95-105.
- [5] B. Razavi, "RF Microelectronics," Upper Saddle River, NJ: Prentice Hall PTR, 1998, Chapter 5.
- [6] H. Tsurumi, and Y. Suzuki, "Broadband RF stage architecture for software-defined radio in handheld terminal applications," *IEEE Communications Magazine*, pp. 90-95, Feb. 1999.
- [7] R. Hartley, "Modulation System," US Patent 1,666,206, April 1928.
- [8] D.K. Weaver, "A Third Method of Generation and Detection of Single-Sideband Signals," *Proc. IRE*, Vol. 44, pp. 1703-1705, Dec. 1956.
- [9] J.Min, "Analysis and Design of a Frequency-Hopped Spread-Spectrum Transceiver for Wireless personal Communications", Final Report, 1996
- [10] Texas Instruments, "TMS320C6000 Chip Support Library API reference Guide (Rev. D)", Literature No. SPRu401D, <http://www.ti.com/sc/psheets/spru401d/spru401d.pdf>
- [11] "Using the TMS320C6000 McBSP as a High Speed Communication Port," <http://focus.ti.com/lit/an/spra455a/spra455a.pdf>.
- [12] Texas Instruments, "Applications using the TMS320C6000 Enhanced DMA", Literature No. SPRA636A, <http://www.ti.com/sc/psheets/spra636a/spra636a.pdf>
- [13] <http://www.ncjrs.org/pdffiles1/nij/185030b.pdf>
- [14] <http://www.ce-mag.com/archive/02/Spring/cutler2.html>. visited on 22nd February 2004.
- [15] Cellular Communication by International Engineering Consortium http://www.iec.org/online/tutorials/acrobat/cell_comm.pdf
- [16] Learn more about TDMA and other cellular technologies
http://cellphones.about.com/library/glossary/bldef_tdma.htm

- [17] The principles of Spread Spectrum communication <http://cas.et.tudelft.nl/~glas/ssc/techn/>
- [18] J. Mitola III, "Software Radio: Technology and Prognosis" Proceedings of the IEEE National Telesystems Conference (NY: IEEE Press) May 1992
- [19] Software Radio Architecture Evolution *Foundations, Technology Tradeoffs, and Architecture Implications*
- [20] The Software Defined Radio Request for Information, (Atlanta, GA: BellSouth) Dec 1995
- [21] Pengelly, R. "Low Cost GaAs MMIC Chip Set for Dual Mode AMPS/CDMA Phones", WESCON 96, p 96
- [22] Oh, S. "DSP Technology for a Hands-Free Car Cellular Phone", WESCON 96, p 821
- [23] P.G. Cook, and W. Bonser, "Architectural overview of the SPEAKeasy system," IEEE Journal on Selected Areas in Communications, Volume: 17 Issue: 4 April 1999, Page(s): 650 –661
- [24] <http://www.rfmd.com/DataBooks/db97/2948B.pdf>
- [25] Behzad Razavi, RF Microelectronics, Prentice Hall, New Jersey, 1998.
- [26] Texas Instruments, "TMS320C6000 Peripherals Reference Guide", Feb. 2001
- [27] Texas Instruments, "TMS320C6000 Technical Brief", Feb. 1999
- [28] Theodore S. Rappaport, Wireless Communications, Principles & Practice, Prentice Hall, PTR, New Jersey, 1996.
- [29] Texas Instruments, "TMS320C62x/67x Programmer's Guide", May 1999
- [30] Cirronet, WIT2410 2.4GHz Spread Spectrum Wireless Industrial Transceiver. Integration Guide, Oct. 2001.
- [31] <http://www.reed-electronics.com/ednmag/index.asp?layout=article&articleid=CA45858>
- [32] Philip Mackenzie, Trinity College, Dublin, Ireland, Lina Doyle, Keith Nolan, Donal O'Mahony, "Selecting Appropriate Hardware for Software Radio Systems."
- [33] Volume: 9, Issue: 6, Year: Dec. 2002 **Guest Editorial - Wireless LANs**
Bing, B.; Heegard, C.; Heile, B.
- [34] <http://www.tmeg.com/tutorials/antennas/antennas.htm>
- [35] http://www.arcelect.com/DSSS_FHSS-Spead_spectrum.htm
- [36] <http://www.lightball.dk/download/dk/pdf/LM6172.pdf>
- [37] Motorola, "CME11E9-EVBU", 1999; <http://www.axman.com>
- [38] Motorola, "BAFFALO MONITOR", 1999; <http://www.axman.com>
- [39] Motorola, "HC11 REFERENCE MANUAL", 1999; <http://www.axman.com>

- [40] K.S. *Shanmugan*, "Estimating the power spectral density of ultra wideband signals," In Proceedings Of the 2002 IEEE Conference on Personal Wireless Communications, 15-17 December 2002, Page(s): 124 –128.
- [41] T. *Shono*, H. *Shiba*, Y. *Shirato*, K. *Uehara*, K. *Araki* , and M. *Umehira*, " Performance of IEEE 802.11 wireless LAN implemented on software defined radio with hybrid programmable architecture, Communications," In Proceedings of the 2003 IEEE International Conference on Communications, Volume: 3, 11-15 May 2003, Page(s): 2035 – 2040.
- [42] Leon W. Couch II, "Digital and Analog Communication Systems," Upper Saddle River, NJ: Prentice-Hall, 1997, Page(s): 277 - 288
- [43] Peter Spasov, "Microcontroller Technology: The 68HC11,"Upper Saddle River, NJ: Prentice-Hall, 2002, Page(s): 370-389.
- [44] B. *Kamali*, " Development of an undergraduate structured laboratory to support classical and new base technology experiments in communications," IEEE Transactions on Education, Volume: 37 Issue: 1, February 1994, Page(s): 97 – 100.
- [45] P.M. *Shankar*, and B.A. *Eisenstein*, " Project-based instruction in wireless communications at the junior level," IEEE Transactions on Education, Volume: 43 Issue: 3, August 2000, Page(s): 245 –249.
- [46] T.M. *Weller*, P.G. *Flikkema*, L.P. *Dunleavy*, H.C. *Gordon*, and R.E. *Henning*, " Educating tomorrow's RF/microwave engineer: a new undergraduate laboratory uniting circuit and system concepts," 1998 IEEE International Microwave Symposium Digest, Volume: 2, 7-12 June 1998, Page(s): 563 –566.
- [47] M.A. *Jensen*, R.H. *Selfridge*, and K.F. *Warnick*, " System-level microwave design projects," IEEE Antennas and Propagation Magazine, Volume: 43 Issue: 5, October 2001 Page(s): 138 – 142
- [49] C.D. Ferris, "Development of a practice-oriented radio-frequency design course," IEEE transactions on Education, Volume: 34 Issue: 1, February 1991, Page(s): 118 –122.
- [50] *Ruly Lai-U Choi* and *Ross D. Murch*, "New Transmit Schemes and Simplified Receivers for MIMO Wireless Communication Systems," IEEE transactions on wireless communications, Volume: 2 Issue: 6, November 2003 Page(s):1217-1230.
- [51] "WIT2410 2.4 GHz Spread Spectrum Wireless Industrial Transceiver Integration Guide," Cirronet, Inc., www.cirronet.com, available on CD provided with the WIT2410.
- [52] Akhi's thesis: Fraidun Akhi, "Design and implementation of a software radio testset for research and laboratory instruction." MS Thesis, Auburn University, 2003
- [53] ASEE paper: Ratha An and Thaddeus Roppel, " Laboratory Technology for Educating Engineers in the Wireless Information Age," ASEE-SE Conference, Auburn, AL, April 2004
- [54] <http://www.ifip.or.at/con2000/icct2000/icct325.pdf>, visit on April 24, 2005
- [55] <http://www.cotsjournalonline.com/home/article.php?id=100191>
- [56] <http://ieeexplore.ieee.org/iel1/4/10262/00482187.pdf?tp=&arnumber=482187&isnumber=10262>