

ECE508

Analog and Digital Communications:
Laboratory Explorations using a
Software-defined Radio Approach

Lee C. Potter
Ohio State University

Draft v0.7
©September 21, 2009

Contents

| | | |
|----------|---|-----------|
| 1 | Introduction | 1 |
| 1.1 | Software Radio | 1 |
| 1.2 | IF Transceiver | 3 |
| 1.3 | Background | 4 |
| 1.3.1 | Frequency up-conversion | 5 |
| 1.3.2 | Frequency down-conversion | 6 |
| 1.3.3 | Sampling and aliasing | 7 |
| 1.3.4 | Digital up-conversion | 9 |
| 1.3.5 | Digital down-conversion | 11 |
| 1.4 | Explorations | 12 |
| 1.4.1 | Experiment 1.1: Getting Started | 12 |
| 1.4.2 | Experiment 1.2: Audio file input | 16 |
| 1.4.3 | Experiment 1.3: RF front end | 16 |
| 1.4.4 | Experiment 1.4: AM transmitter | 17 |
| 2 | Noncoherent AM | 19 |
| 2.1 | Background | 19 |
| 2.1.1 | Amplitude Modulation | 19 |
| 2.1.2 | Example: Single tone modulation | 21 |
| 2.1.3 | Envelope detector | 22 |
| 2.1.4 | Square-law detector | 23 |
| 2.2 | Experiment: Construct a Crystal Radio | 24 |
| 2.2.1 | Construction | 24 |
| 2.2.2 | Questions | 27 |
| 2.3 | Experiment: AM Transmission with Overmodulation | 28 |
| 2.4 | Experiment: Programmable waveform | 29 |
| 2.4.1 | Procedures | 29 |
| 2.4.2 | Questions | 30 |

| | | |
|----------|--|-----------|
| 2.5 | Experiment: Square-law detector | 30 |
| 2.5.1 | Procedures | 31 |
| 2.5.2 | Questions | 32 |
| 3 | Coherent AM | 33 |
| 3.1 | Background | 33 |
| 3.1.1 | Quadrature amplitude modulation | 33 |
| 3.1.2 | Complex baseband representation | 36 |
| 3.1.3 | DSB-SC AM demodulation: Costas loop | 37 |
| 3.2 | Explorations | 40 |
| 3.2.1 | Exercise: QAM signals | 40 |
| 3.2.2 | Exercise: Phase offset | 41 |
| 3.2.3 | Exercise: Memoryless phase recovery for DSBSC-AM | 42 |
| 3.2.4 | Exercise: Closer look at the phase offset | 43 |
| 3.2.5 | Exercise: Frequency and phase recovery | 44 |
| 4 | SSB-AM | 47 |
| 4.1 | Background | 47 |
| 4.1.1 | Hartley modulator | 47 |
| 4.1.2 | SSB demodulation | 49 |
| 4.1.3 | Vestigial sideband | 50 |
| 4.2 | Explorations | 50 |
| 4.2.1 | Exercise: Hartley modulator | 50 |
| 4.2.2 | Exercise: SSB coherent demodulation | 51 |
| 5 | Frequency Modulation | 55 |
| 5.1 | Background | 55 |
| 5.1.1 | Modulation | 55 |
| 5.1.2 | Demodulation | 57 |
| 5.2 | Explorations | 58 |
| 5.2.1 | Exercise: Analog discriminator | 58 |
| 5.2.2 | Exercise: Digital discriminator | 60 |
| 5.2.3 | Exercise: FM chirp waveform | 61 |
| 5.2.4 | Exercise: Range/Doppler radar | 61 |
| 5.2.5 | Exercise: Commercial FM radio | 61 |

| | | |
|----------|--|-----------|
| 6 | Matched Filter | 63 |
| 6.1 | Background | 63 |
| 6.1.1 | Maximizing SNR | 63 |
| 6.1.2 | Time-of-arrival estimation | 65 |
| 6.1.3 | Timing recovery | 65 |
| 6.1.4 | Channel estimation | 66 |
| 6.2 | Explorations | 66 |
| 6.2.1 | Exercise: Echolocation | 67 |
| 6.2.2 | Exercise: Channel estimation | 69 |
| 6.2.3 | Exercise: Phase recovery | 70 |
| 7 | BPSK | 71 |
| 7.1 | Background | 71 |
| 7.1.1 | Fractional-Rate Baseband Processing | 71 |
| 7.1.2 | Pulse Shaping and Intersymbol Interference | 72 |
| 7.2 | Explorations | 76 |
| 7.2.1 | Exercise: Pulse Shaping | 76 |
| 7.2.2 | Exercise: Implement BSPK modem | 76 |
| 7.2.3 | Exercise: ISI | 78 |
| 8 | PSK | 79 |
| 8.1 | Design Specifications | 79 |
| 8.2 | Implementation | 79 |
| 8.3 | Reporting | 80 |
| A | NI 5640R | 81 |
| A.1 | Rx and Tx Packets | 90 |
| A.2 | Brief overview of working of NI PCI-5640R | 91 |

Chapter 1

Introduction: Software Radio

The goals of this introductory exercise are to introduce the hardware and software tools to be used during the course and to explore frequency up-conversion and down-conversion.

1.1 Software Radio

As an evolving technology, Software Defined Radio is a work in progress. Its definition likewise continues to evolve. Software Defined Radio, also known as Software Radio or SDR, is broadly defined as “radio in which some or all of the physical layer functions are software defined.”¹

A radio is any kind of device that wirelessly transmits or receives signals in the radio frequency (RF) part of the electromagnetic spectrum to facilitate the transfer of information. In today’s world, radios exist in a multitude of items such as cell phones, computers, car door openers, vehicles, and televisions.

Traditional hardware-based radio devices limit cross-functionality and can only be modified by physically changing hardware components. This results in minimal flexibility to support multiple waveform standards. In contrast, SDR technology provides an efficient and comparatively inexpensive solution to this problem, allowing multi-mode, multi-band and/or multi-functional wireless devices that can be enhanced using software upgrades.

¹“SDRF Cognitive Radio Definitions,” Working Document SDRF-06-R-0011-V1.0.0, November 2007. <http://www.sdrforum.org>.

SDR defines a collection of hardware and software technologies where some or all of the radios operating functions (also referred to as physical layer processing) are implemented through modifiable software or firmware operating on programmable processing technologies. These devices include field programmable gate arrays (FPGA), digital signal processors (DSP), and general purpose processors (GPP). The use of these technologies allows new wireless features and capabilities to be added to existing radio systems without requiring new hardware.

While an ideal, fully programmable SDR may be realized at low transmission frequencies without hardware for frequency up-conversion or down-conversion, most existing SDRs use mixers at the front end to perform analog up-conversion and down-conversion. The receiver chain for such a system is depicted in Figure 1.1.

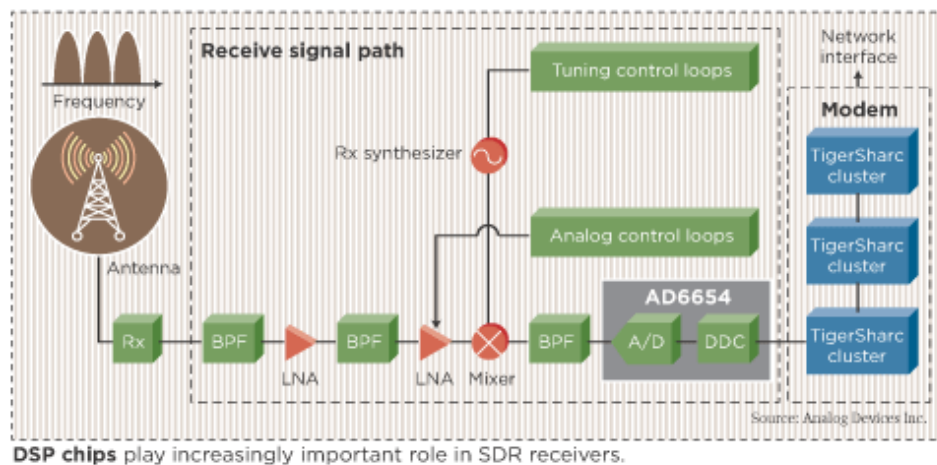


Figure 1.1: SDR receiver chain. Source: Analog Devices, Inc.

On receive, a mixer down-converts the signal to achieve an intermediate frequency (IF) that can be handled by today's ADCs. The I/Q mixer format preserves phase and frequency information contained in most digital

modulation schemes².

Digital down-converters (DDCs) are commonly used after the ADC to further lower the data rate so that memory requirements may be relaxed and processing speeds are more moderate. These devices come as individual ICs, but their functions also can be implemented in the ADC or in the baseband processor. A DDC does what an analog down-convert mixer does, but in a digital manner. It takes the ADC samples and multiplies them by samples of the carrier frequency generated by a numerically controlled oscillator (NCO) or direct digital synthesizer (DDS). Multiplying is the same as mixing. Multiplying the signal by a carrier frequency produces signals that are the sum and difference of the signal and local oscillator frequencies.

The difference signal is the baseband waveform; the higher-frequency sum signal is filtered out. Next, the resulting signals are decimated, meaning that only one of each N samples is retained and the others are discarded. This lowers the sample rate commensurate with the baseband bandwidth, making it far easier to process the data in a DSP or FPGA.

A similar set of steps occurs on transmission. Digital up-conversion (DUC), used at the transmitter, takes a lower set of frequencies and boosts them up to an IF; an up-convert mixer takes the DAC signal to be transmitted and converts it to the final transmission frequency.

1.2 IF Transceiver

The central tool in most laboratory exercises will be the National Instruments PCI-5640R Intermediate Frequency (IF) transceiver, with power and flexibility that make it well-suited for teaching and research involving software-defined radio and other communications applications. The PCI bus board installs into a desktop PC and is fully programmable with intuitive NI LabVIEW graphical programming. The board offers multiple options for processing received signals or preparing signals for transmission. A user may engage an onboard Xilinx Virtex-II Pro FPGA for inline processing or choose host-based processing by streaming signals to and from the host PC. And, MATLAB code may be conveniently and directly inserted into the LabVIEW graphical programs.

²L. E. Frenzel, "The Elusive Software-Defined Radio," *ElectronicDesign.com*, article #13380, September 2006. Penton Media, Inc.

The NI PCI-5640R offers two 100 megasample per second (MS/s), 14-bit input channels with built-in digital down-conversion and two 200 MS/s, 14-bit output channels with built-in digital up-conversion. Each channel can be operated with up to 20 MHz real-time bandwidth and 250 kHz to 80 MHz analog frequency range.

The Xilinx Virtex II Pro FPGA connects to the input/output resources on the device: analog-to-digital converter (ADC), digital-to-analog converter (DAC) and Clock Distribution Circuit. The AD9857 provides the high-speed DAC; it is intended to function as a universal I/Q modulator and agile up-converter, single-tone direct digital synthesizer (DDS), or interpolating DAC for communications applications. The AD6654 is a mixed-signal IF to base band receiver and is intended for use as part of a radio system that digitally demodulates and filters IF sampled signals. The ADC and DAC functions are discussed in Sections 1.3.4 and 1.3.5.



Figure 1.2: NI PCI-5640R Software-Defined Radio IF Transceiver

1.3 Background

This section provides a brief review of analytical concepts explored in the laboratory exercises. First, up-conversion and down-conversion are defined for the real-valued signal case. Second, sampling and aliasing are reviewed. (Complex-valued sampling and quadrature modulation are explored in Chapter 3.) Third, digital up-conversion and digital down-conversion are summarized.

1.3.1 Frequency up-conversion

up-conversion (amplitude modulation or mixing) of a real-valued message $m(t)$ is given by

$$s(t) = m(t) \cos(2\pi f_c t) \quad (1.1)$$

and serves to translate the message to desired frequency band centered on the carrier frequency, f_c . up-conversion is graphically depicted in Figure 1.3. From the Euler identity,

$$\cos(2\pi f_c t) = \frac{1}{2} [e^{j2\pi f_c t} + e^{-j2\pi f_c t}] \quad (1.2)$$

the spectrum, $S(f)$ is found via the Fourier transform

$$\begin{aligned} S(f) &= \int_{-\infty}^{\infty} m(t) \cos(2\pi f_c t) e^{-j2\pi f t} dt \\ &= \frac{1}{2} \int_{-\infty}^{\infty} m(t) e^{-j2\pi(f-f_c)t} dt + \frac{1}{2} \int_{-\infty}^{\infty} m(t) e^{-j2\pi(f+f_c)t} dt \\ &= \frac{1}{2} M(f - f_c) + \frac{1}{2} M(f + f_c). \end{aligned} \quad (1.3)$$

Because $m(t)$ is real-valued, $|M(f)|$ is symmetric around $f = 0$, implying the up-converted spectrum is symmetric about f_c and the spectrum lower sideband (below f_c) is redundant with the upper sideband (above f_c). The spectrum is illustrated in Figure 1.4.

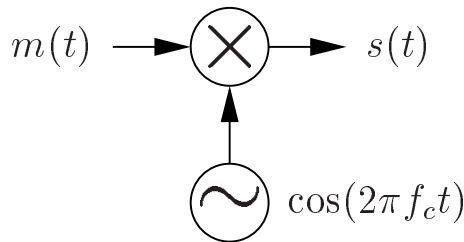


Figure 1.3: Amplitude modulation performs a translation in frequency.

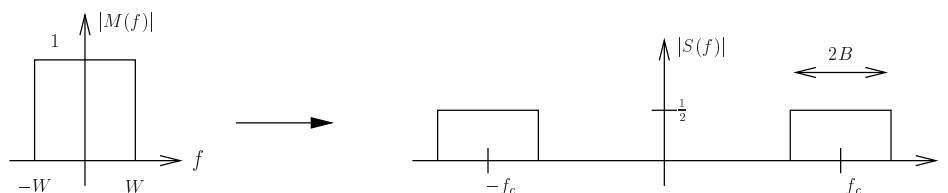


Figure 1.4: Spectrum resulting from amplitude modulation.

1.3.2 Frequency down-conversion

With f_c known, amplitude demodulation can be accomplished by mixing and lowpass filtering

$$\begin{aligned}
 v(t) &= \text{LPF}\{s(t) \cdot 2 \cos(2\pi f_c t)\} \\
 &= \text{LPF}\{m(t) \cdot \underbrace{2 \cos^2(2\pi f_c t)}_{1 + \cos(2\pi \cdot 2f_c t)}\} \\
 &= \text{LPF}\{m(t) + m(t) \cos(2\pi \cdot 2f_c t)\} \\
 &= m(t)
 \end{aligned} \tag{1.4}$$

This process is depicted in Figure 1.5.

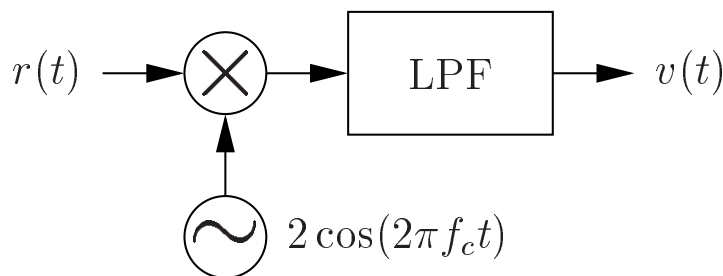


Figure 1.5: Amplitude demodulation.

In Figure 1.5, the lowpass filter (LPF) has a passband cutoff $B_p \geq W$ Hz and stopband cutoff $B_s \leq 2f_c - W$ Hz, where W is the one-sided bandwidth of the baseband signal, $m(t)$. The filter's magnitude response is illustrated in Figure 1.6.

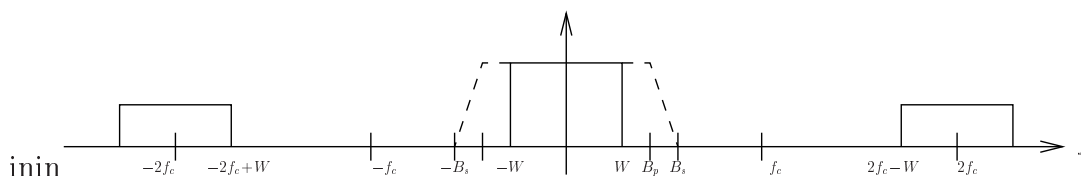


Figure 1.6: Lowpass filter (dashed line) used in amplitude demodulation.

When the receiver oscillator has frequency offset γ or phase offset ϕ relative to the up-conversion, the demodulated signal is distorted:

$$\begin{aligned}
 v(t) &= \text{LPF} \left\{ m(t) \underbrace{\cos(2\pi f_c t) \cdot 2 \cos(2\pi(f_c + \gamma)t + \phi)}_{\cos(2\pi\gamma t + \phi) + \cos(2\pi(2f_c + \gamma)t + \phi)} \right\} \\
 &= m(t) \underbrace{\cos(2\pi\gamma t + \phi)}_{\text{time-varying attenuation}}. \tag{1.5}
 \end{aligned}$$

The time-varying attenuation can be viewed as an up-conversion of $m(t)$ to a frequency equal to the offset, γ .

1.3.3 Sampling and aliasing

The basic intuition of sampling and aliasing is evident in Figure 1.7: given only samples of a sinusoidal waveform, there exist tones of infinitely many possible frequencies that pass through the samples. The frequencies of these possible interpolating waveforms differ by a multiple of the sampling frequency. In the figure, the solid line shows $m(t) = \sin(2\pi 20t)$, a sine wave at 20 Hz. Samples, shown by dots-on-sticks, are taken every 0.01 s, for $f_s = 100$ S/s. The top panel shows that $\sin(2\pi(20 + f_s)t) = \sin(2\pi 120t)$ also is consistent with the same set of samples; likewise, the bottom figure

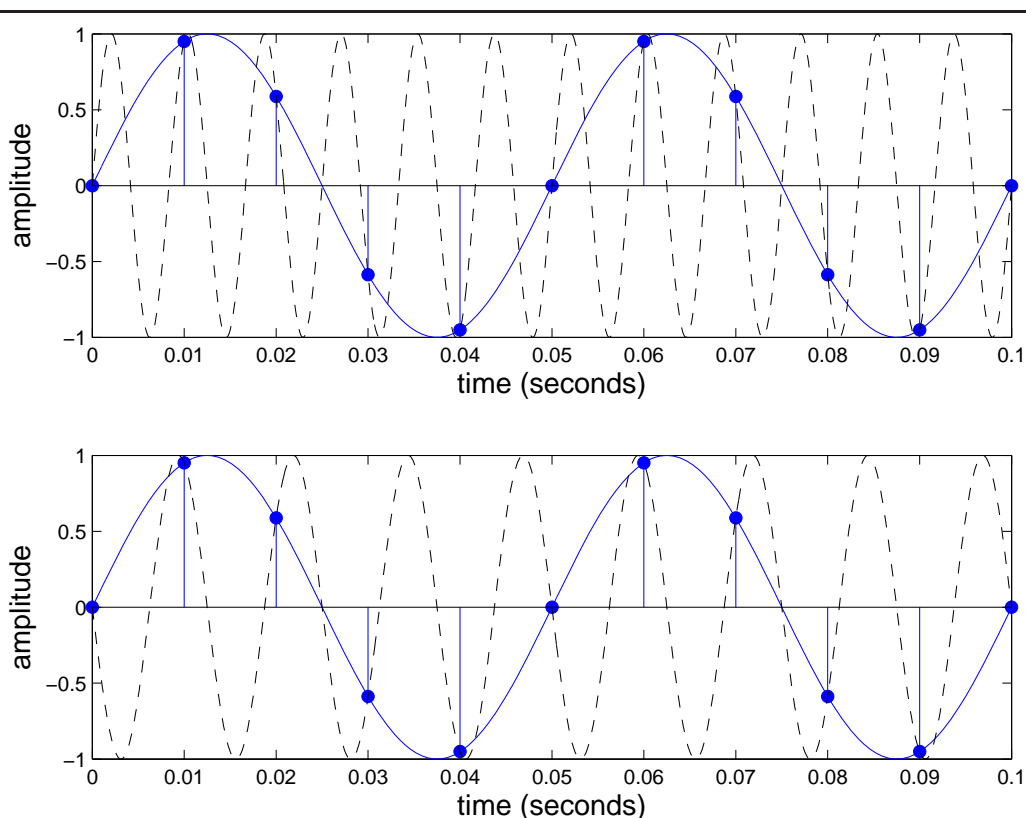


Figure 1.7: Illustration of sampling and aliasing.

illustrates that $\sin(2\pi(20 - f_s)t) = \sin(2\pi(-80)t) = \sin(2\pi 80t + \pi)$ is another tone yielding the same samples.

The Fourier spectrum of the sampled signal is the sum of these possibilities. Let $x(t) = m(t) * \sum_n \delta(t - nT)$ be the sampled signal; then,

$$X(f) = \frac{1}{T} \sum_{k=-\infty}^{\infty} M(f - kf_s) \quad (1.6)$$

Each shifted replica of the spectrum $M(f)$ is called an *image*. To eliminate confusion among the possible signals yielding a sampled waveform, the *Nyquist criterion* stipulates that $M(f) = 0$ for all $|f| \geq f_s/2$; that is, the sampling rate gives at least two samples per cycle for any frequency present in $m(t)$.

More generally, bandpass sampling requires that the two-sided bandwidth

of the spectrum $M(f)$ be less than the sampling rate.

1.3.4 Digital up-conversion

On the NI PCI-5640R, the digital-to-analog converter (DAC) is combined with a digital up-converter (DUC) on a single integrated circuit. The AD9857 integrates a high-speed direct-digital synthesizer (DDS), a high-performance, high-speed 14-bit digital-to-analog converter (DAC), clock multiplier circuitry, digital filters, and other DSP functions onto a single chip, to form a complete quadrature digital up-converter device. The AD9857 is intended to function as a universal quadrature modulator and agile up-converter, single-tone DDS, or interpolating DAC for communications applications. The basic operation is abstracted in Figure 1.8.

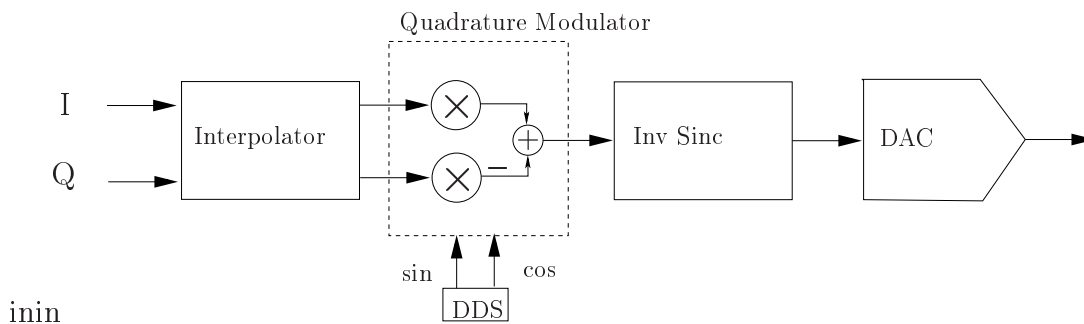


Figure 1.8: Basic operation of a quadrature digital up-converter.

Direct digital synthesis

A direct digital synthesizer (DDS) is the core of the AD9857 and is a numerically controlled oscillator (NCO). A phase accumulator provides a selectable modulus counter that increments with each clock pulse; the increment is determined by a digital word, and the value in the accumulator is used to index a sinewave look-up table to produce accurate, stable, and phase-continuous frequency tuning. When used as a quadrature synthesizer, both cosine and

sine outputs are produced, yielding excellent matching of inphase (I) and quadrature (Q) outputs.

Interpolation

Interpolation is a digital filtering procedure that increases the sampling rate by an integer factor L

$$y[n] = \sum_{k=-\infty}^{\infty} x[k] \frac{\sin(\pi(n - kL)/L)}{\pi(n - kL)/L}. \quad (1.7)$$

The processing can be viewed as interlacing $L - 1$ zeros between consecutive samples of $x[n]$, then lowpass filtering with gain L and cutoff frequency $\frac{f_s}{2L}$ to produce the interpolated samples. See Figure 1.9.

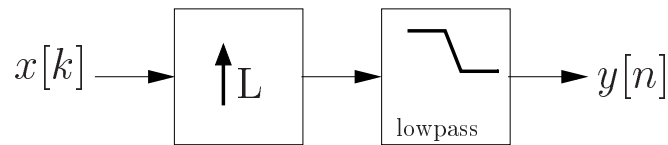


Figure 1.9: Block diagram for interpolation by L ; the image reject lowpass filter has cutoff frequency $f_s/2$, where f_s is the original sampling rate and Lf_s is the output sampling rate.

The action of sinc interpolation filter is illustrated in Figure 1.10, for the limiting case of producing a continuous-time output

$$y(t) = \sum_{k=-\infty}^{\infty} x[k] \frac{\sin(\pi(t - kT)/T)}{\pi(t - kT)/T} \quad (1.8)$$

where T is the sampling period for $x[k]$. The summation is the convolution of a lowpass filter impulse response (a scaled sinc function) with the impulse train of samples, $x[k]\delta(t - kT)$. This summation is depicted in the figure, where the solid circles denote the samples, the dashed lines denote each shifted sinc function, as in Equation 1.8, and the summation is shown by the solid line. The zero crossings of the sinc function, occurring every T seconds, ensure the faithful interpolation of the original samples. In practice, the sinc impulse response is approximated using a finite impulse response.

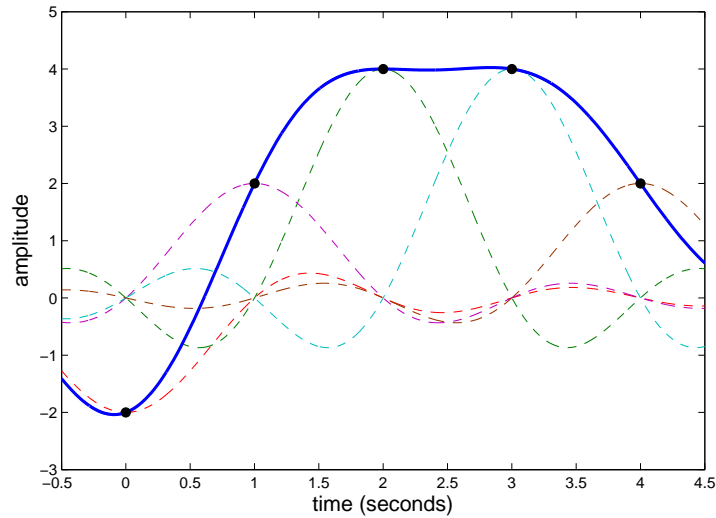


Figure 1.10: Illustration of sinc interpolation.

1.3.5 Digital down-conversion

The AD6654 is a mixed-signal IF to base band receiver consisting of a 14-bit, 92.16 MSPS analog-to-digital converter (ADC) and a multi-channel digital down-converter (DDC). It has been optimized for wideband standards such as CDMA2000, UMTS, and TD-SCDMA. The AD6654 is used as part of a radio system that digitally demodulates and filters IF sampled signals. The basic operation is abstracted in Figure 1.11.

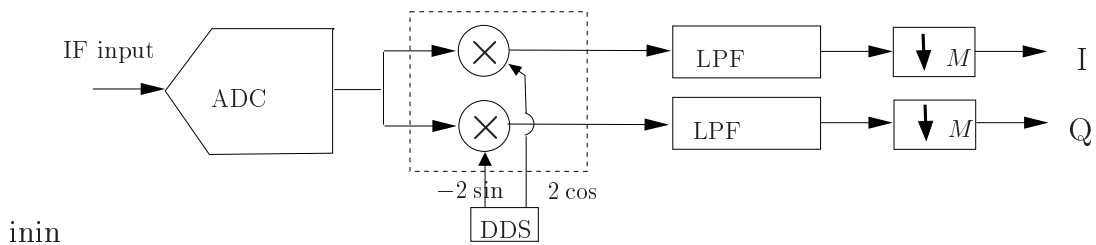


Figure 1.11: Basic operation of a quadrature digital down-converter.

Decimation is a digital filtering procedure that decreases the sampling rate. Given an original sampling rate of f_s , decimation by a positive integer M produces a new sampling rate of f_s/M . The signal is first lowpass filtered, typically in stages, with a cutoff frequency of $\frac{f_s}{2M}$. The filtering prevents aliasing from resulting when the signal is down-sampled. Only every M th sample of the filtered output is computed and retained. Refer to Figure 1.12. Note that the lowpass filtering is an averaging that can reduce noise, particularly noise due to quantization error.

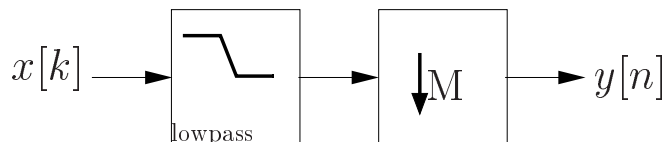


Figure 1.12: Block diagram for decimation by M ; the antialiasing lowpass filter has cutoff frequency $f_s/(2M)$, where f_s is the original sampling rate and f_s/M is the output sampling rate.

1.4 Explorations

1.4.1 Experiment 1.1: Getting Started

In this experiment, you will familiarize yourself with the operation of the NI PCI-5640R IF transceiver. You will explore the operation of the digital up-converters and down-converters. For this first experiment, explicit instructions will be given to speed your exploration. In subsequent chapters, exercises will be less scripted, with freedom for creative detours and experimentation.

Begin

To begin, copy template LabVIEW Virtual Instruments (vi's) to your workspace and run them using LabVIEW 8.2. More detail regarding the templates files is found in Appendix A.

1. Login to the benchtop PC and create a subdirectory for your ECE508 laboratory work.
2. From `C:/508 templates` copy into your subdirectory two vis's: `Rx StreamingIntro.vi` and `Tx Streaming.vi`.
3. On the NI-PCI 5640R board, connect output AO-0 to input AI-0 using a one-foot cable with SMA connectors. (The connections are accessible at the back of the PC chassis; details are given in Appendix A.)
4. From your work directory, double click on `Rx StreamingIntro.vi` and `Tx Streaming.vi` to execute them using LabVIEW 8.2.
5. The Front Panel of each ve should be visible on the computer screen and gives a graphical user interface to the software control of the IF transceiver. Use `CTRL e` to toggle between the Front Panel and the underlying Block Diagram, which shows the graphical programming to implement the vi.

Tx Front Panel

Four inputs are available on the Tx Front Panel, as listed in Table 1.1.

Table 1.1: Tx Front Panel Inputs

| Item | Entry | Description |
|-------------------------|-------------|---|
| Data | menu select | tone or audio |
| center frequency, f_c | numeric | IF up-conversion frequency (Hz) |
| IQ rate, $1/T$ | numeric | sample rate at which data are read from buffer (Hz) |
| Data block size | numeric | size of DMA transfer from host to FPGA buffer (Samples) |

The `Data` select menu allows the user to create a tone at IF or to amplitude modulate an audio-frequency waveform to an IF carrier. The `center frequency` is user selectable; for proper operation, the suggested range is 250 kHz to 75 MHz. The `IQ rate` defines the sampling rate at which data

are read from the buffer by the digital up-converter (DUC). This value must be

$$\frac{50 \text{ MHz}}{L}$$

where L is an integer from 2 to 63. The DUC performs an interpolation by L and another interpolation by a fixed factor of 4 to arrive at a sampling rate of 200 MHz prior to digital-to-analog conversion.

Toggle to the Block Diagram and peruse the graphical program. Note in particular the MATLAB SCRIPT NODE on the left, which in the `tone` mode is used to create the constant baseband signal $m[k] = 1$ using the `ones` command. This constant waveform is then mixed to f_c by the DUC.

1. Select `tone`
2. Enter 1.6M (or 1600k) for center frequency.
3. Enter 780k for IQ rate. This entered value will be set to an actual value of $50000/63 = 793.65 \text{ k}$ upon execution of the VI.
4. Enter 4k for block size.

Rx Front Panel

Four inputs are available on the Rx Front Panel, as listed in Table 1.2. The

Table 1.2: Rx Front Panel Inputs

| Item | Description |
|-------------------------|---|
| center frequency, f_c | IF down-conversion frequency (Hz) |
| IQ rate, $1/T$ | sample rate at DDC output (Hz) |
| Data block size | size of DMA transfer from FPGA buffer to host (Samples) |
| Buffer size | (Samples) |

`center frequency` is user selectable; for proper operation, the suggested range is 250 kHz to 75 MHz. The `IQ rate` defines the sampling rate at which data are produced by the ADC, after decimation. This value must be

$$\frac{50 \text{ MHz}}{M}$$

where M is a power of 2 from 4 to 2048. The DDC performs an decimation by M from the base rate of 100 MHz.

On the bottom left, the virtual instrument computes and displays the magnitude spectrum. The spectrum is displayed from $-f_s/2$ to $f_s/2$, where f_s is the Rx IQ rate. On the bottom right of the Front Panel, the VI displays the sampled waveform, as one would see on an oscilloscope.

1. Enter 1.6M (or 1600k) for center frequency.
2. Enter 500k for IQ rate. This entered value will be set to an actual value of $100000/64 = 781.25$ k upon execution of the VI.
3. Enter 4k for data block size.
4. Enter 2097152 for buffer size, corresponding to 2^{21} samples.

Return to the Tx VI and execute by pressing the arrow at the top left of the menu bar. Then, execute the Rx VI.

down-conversion

Explore the operation of the digital down-conversion by stopping the Rx VI (red button at top left of menu bar), revising input values, then executing again. (Note: the TX VI may remain running while values are changed.)

1. Try six values of f_c : 1.5 M, 1.6 M, 1.7 M, 1.98 M, 2.00 M, and 2.05 M. Record the amplitude (dB) and frequency (kHz) of the peak frequency for each case by stopping the Rx program, and zooming the horizontal scale on the spectrum plot. You can zoom by clicking the end values on the horizontal axis, typing new values (including the suffix **k**), then pressing **Enter**. Record your data. Question 1.1
2. What do you see, and why? Try to explain the observations in terms of the functioning of the DDC. (Equation 1.5 may be particularly useful.) Question 1.2
3. At the Rx Front Panel, reset the IQ rate to a new value and explore with three or four choices of carrier frequency. Again note your observations and explain the behavior you observe.
4. Halt execution of the Tx and Rx Front Panels.

1.4.2 Experiment 1.2: Audio file input

Revisiting Experiment 1.1, use the Tx Front Panel **data** menu button to select **audio**. View the data flow from the Block Diagram (using CTRL **e** to toggle). The LabVIEW template reads an audio file and interpolates to match the sampling rate specified by **IQ rate**.

Again execute the Tx and Rx Front Panel vi's. Observe the Rx spectra with f_c at the Rx equal to 1.6 M and 1.7 M. What is the baseband bandwidth of the audio segment upon stopping execution to obtain a zoomed view of the spectrum?

1.4.3 Experiment 1.3: RF front end

Caution: Do not input a signal of more than 800mV into the NI PCI-5640R board. Always check the peak-to-peak value and offset of the signal using the oscilloscope before connecting as input to the NI board.

As shown in Figure 1.1, most radios use software-defined components to generate IF signals and process the baseband signal; fixed hardware is typically used for conversion between IF and radio frequency (RF) signals. In this exercise, we will use hardware for IF-to-RF up-conversion and RF-to-IF down-conversion.

First, consider down-conversion. Using the **Tx Streaming.vi** template, generate a tone at 20 MHz. Connect the output **A0-0** to the IF input of a Minicircuits mixer. Use the function generator to produce the local oscillator (LO) input to the mixer. Display the mixer RF output to the oscilloscope.

- Question 1.3 1. Mix down to an IF frequency of 10.5 MHz. What do you choose for the LO frequency?
- Question 1.4 2. As shown in Figure 1.5, insert a Mini-Circuits lowpass filter between the mixer IF output and the oscilloscope. Observe and describe the IF signal displayed on the oscilloscope.
- Question 1.5 3. Vary the function generator ± 3 MHz. Observe and describe the IF signal displayed on the oscilloscope.
- Question 1.6 4. Remove the lowpass filter and again observe the output of the mixer with LO as in (1). What differences do you observe with and without the filter? Explain.

5. Replace the lowpass filter. Take the output of the lowpass filter to analog input AI-0 of the NI PCI-5640R board. Repeat (2), using the Rx StreamingIntro.vi template to view the spectrum. What values of center frequency and IQ rate should you set to observe the signal? Question 1.7

Second, consider up-conversion.

1. Using the Tx Streaming.vi template, generate a tone at 6 MHz. Connect the output A0-0 to the IF input of a Minicircuits mixer. Use the function generator to produce the local oscillator (LO) input at 10 MHz to the mixer. Display the mixer output (labeled “RF”) to the oscilloscope.
2. Vary the function generator ± 5 MHz. Observe and describe the IF signal displayed on the oscilloscope. Question 1.8
3. Use the Rx StreamingIntro.vi template to view the spectrum. That is, take IF output of the mixer to the analog input AI-0 of the NI PCI-5640R board. What values of center frequency and IQ rate should you set to observe the signal? Question 1.9

1.4.4 Experiment 1.4: AM transmitter

Use the Tx Streaming.vi template to implement a AM transmitter at 1600 kHz.

1. First, implement an ideal Software Radio without an analog front end; this is possible at desired frequency band.
 - (a) Connect the A0-0 output to antenna and tune a commercial AM radio receiver to demodulate your broadcast.
 - (b) Try changing the transmit frequency (by amounts less than 5 kHz) and observe the demodulated sound output.
 - (c) Try other AM-band carrier frequencies. Is one carrier frequency better matched to the antenna you are using, thereby resulting in a stronger receive signal?
2. Second, implement an analog front end (for instruction purposes only, as we will actually down-convert the frequency to create the AM broadcast signal).

- (a) Transmit the the audio file at carrier frequency 12 MHz using the `Tx Streaming.vi` template. Mix the A0-0 output with a 13.6 MHz LO from the function generator and lowpass filter, and transmit using the antenna.
- (b) What differences do you observe between the received AM signal for parts (1) and (2)? Conjecture causes for any difference you observe.

Question 1.10