6-28-2010

Software Defined Radio: Inexpensive Hardware and Software Tools

Thad B. Welch
Boise State University

Cameron H. G Wright
University of Wyoming

Michael G. Morrow
University of Wisconsin–Madison

SOFTWARE DEFINED RADIO: INEXPENSIVE HARDWARE AND SOFTWARE TOOLS

Thad B. Welch

Dept. of Elec. and Comp. Engineering
Boise State University, ID
t.b.welch@ieee.org

Cameron H. G. Wright

Dept. of Elec. and Comp. Engineering
University of Wyoming, WY
c.h.g.wright@ieee.org

Michael G. Morrow

Dept. of Elec. and Comp. Engineering
University of Wisconsin–Madison, WI
morrow@ieee.org

ABSTRACT

DSP topics such as software defined radio are more easily taught if appropriate demonstrations and laboratory experiences are provided. This paper describes a new, inexpensive software defined radio educational platform based upon MATLAB and the Texas Instruments C6713 digital signal processing starter kit. We describe the hardware and software issues and briefly describe recommended classroom use.

Index Terms— digital signal processors, education, sampling methods

1. INTRODUCTION

Software defined radio (SDR) is a topic that is becoming increasingly necessary as part of either a digital signal processing (DSP) course or a digital communications course at most universities today, with the target audience being both undergraduate and graduate students. In the past, even when the theoretical aspects of SDR have been covered in such courses, actual demonstrations and lab exercises for students are rare, and even when given they are typically limited to a simple MATLAB simulation. We believe that adding a hardware aspect to the presentation of SDR, with hands-on opportunities for students, greatly enhances their mastery of this important topic, as we have advocated for similar DSP topics in the past [1, 2]. This paper describes a new, inexpensive SDR educational platform based upon custom MATLAB data acquisition code capable of real-time operation with the Texas Instruments C6713 DSP starter kit (DSK).

2. TEACHING SOFTWARE DEFINED RADIO

2.1. How Much Detail?

One of the challenges in teaching a topic such as SDR to such a diverse student audience is to decide how much detail to include. One or more entire courses could be devoted to SDR and multirate digital communications [3, 4], but we are often limited to just a few lectures and perhaps one demonstration and/or lab exercise. In this situation, we consider the ideas of rate conversion and first-order bandpass sampling to be fundamental to getting students comfortable with the general ideas

of SDR [5, 6]. More detail can be added if time is available in the course, or can included in a follow-on course for interested students.

With this in mind, students must be convinced of three things: 1) that the "gospel" of $F_s \geq 2f_{\max}$ (i.e., sampling at least twice the highest signal frequency) that they learned regarding lowpass sampling is only a special case, 2) that for bandpass signals the selection of sampling frequency is more complicated, and that 3) aliasing is not always a bad thing. If we choose F_s properly, then aliasing places the signal spectrum where we want it, and we can avoid the potential overlapping of spectral replicas that would render the signal useless [7, 8].

One useful form of the expression for predicting the range of acceptable sample frequencies is

$$2B \left(\frac{Q}{n} \right) \leq F_s \leq 2B \left(\frac{Q-1}{n-1} \right) \tag{1}$$

where $B = f_{\mathrm{U}} - f_{\mathrm{L}}$ is the bandwidth of the signal, $Q = \frac{f_U}{B}$, and n is an integer such that $1 \leq n \leq \lfloor Q \rfloor$. In most real-world examples, the signal's frequency content is already specified, leaving n as the first choice the students must learn to make. Choosing $n = 1$ is the same as the lowpass sampling case and provides no instructional benefit; choosing higher values of n allows lower acceptable ranges of sampling frequencies. In general, as n gets larger F_s get lower but the range of acceptable values of F_s gets more narrow; this makes practical engineering issues such as sample jitter more and more obvious [7]. Furthermore, choosing n to be an even number results in "spectral reversal" or "flipping" of the downconverted aliased spectral copies; this is what Liu calls "inverse spectral placement" [8]. This result is often unexpected by students and can thus be useful for instructional purposes. Finally, if Q is an integer, the lowest possible sample frequency of $F_s = 2B$ is an allowable value (when $n = Q$), albeit within an essentially "zero tolerance" frequency range of $2B \leq F_s \leq 2B$. We will take advantage of these ramifications in the teaching example we describe below.

For the purpose of teaching first-order bandpass sampling for SDR to our students, we find that using a signal with which the students are familiar is most effective. To that end, we chose to use the intermediate frequency (IF) signal of a

commercial FM radio, which for the U.S. is defined as having a center frequency of $f_C = 10.7$ MHz, and a bandwidth B of 200 kHz. Thus $f_L = 10.6$ MHz and $f_U = 10.8$ MHz. Since $Q = \frac{f_U}{B} = 54$ is an integer, $F_s = 2B = 400$ kHz is an allowable value if we choose $n = 54$. Note that since this value of n is even, the aliased spectral copies exhibit inverse spectral placement, and there will be "zero tolerance" for the value of F_s, providing many more teaching opportunities than would be available with a more benign choice of n (i.e., an odd integer lower than 54). This choice of n is also consistent with the recommendation of fred harris in [4] to set F_s such that $f_C = F_s(k \pm 0.25)$ for integer k, if we choose $k = 27$. Another significant reason to use $F_s = 400$ kHz is that we know how we can achieve this in relatively inexpensive hardware.

2.2. Teaching with Software and Hardware

How might we teach such SDR concepts to our students? Computer-based demonstrations can be effective with students for many DSP topics [9]. We can take advantage of the fact that the software package MATLAB and its related toolboxes have become a mainstay in most ECE programs. Given our students' familiarity with MATLAB, computer exercises that implement bandpass sampling theory for SDR would seem to be a natural approach. But we have found with many DSP topics that our students are often not impressed with a software-only "canned demo," and adding a hardware component greatly improves the effectiveness of the demo and/or lab exercise. With properly chosen software and hardware, we also have the opportunity to create a real-time system to use as a teaching tool.

In the past, proceeding beyond a MATLAB-only simulation to a real-time hardware implementation has been impeded by a very abrupt transition, in terms of both cost and the learning curve of unfamiliar systems and software. By developing a software and hardware bridge between MATLAB and real-time DSP hardware, we have made it possible to smoothly and incrementally transition from simulation to a full hardware implementation, all while retaining the impressive capabilities of the MATLAB display engine [10]. Using this approach, students are able to develop and enhance their own DSP system that demonstrates bandpass sampling.

2.3. System Requirements

Our students are already very familiar with MATLAB, but we also want them to learn more about hardware-based digital signal processing (DSP). For the primary DSP hardware, our main criteria were low cost, sufficient processing power, and a versatile software development environment. The authors have considerable experience with various DSP evaluation modules and DSKs, and we chose to construct our bandpass sampling educational platform around MATLAB and the inexpensive Texas Instruments (TI) C6713 DSK, which we have
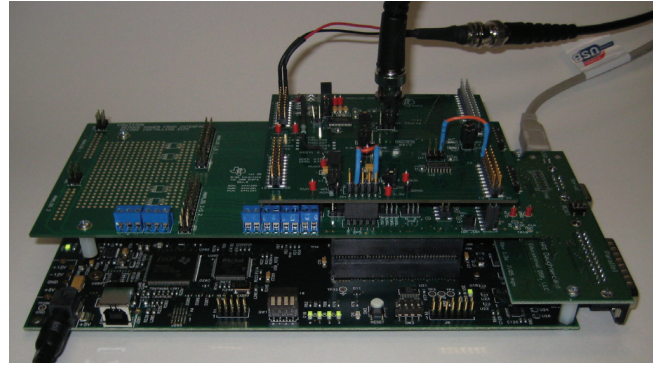
Fig. 1. The TI C6713 DSK (bottom) with the HPI daughtercard (at right end of DSK), the THS1206 ADC (top), and the 5-6K Interface board (middle).

described elsewhere (e.g., [5, 10]). High speed data communication between a PC and the DSK is provided by the inexpensive HPI daughtercard (see [10] or [11] for details).

While the the C6713 DSK is an excellent platform, it's CD-quality codec is suitable only for audio frequencies. To achieve $F_s = 400$ kHz, we chose to add two other inexpensive TI parts: a THS1206 high speed analog-to-digital converter (ADC) evaluation module (EVM) and the 5-6K Interface Board that connects the THS1206 EVM to the C6713 DSK. These parts are described in more detail below.

For the software part of this educational platform, we have developed a versatile data acquisition library for MATLAB that works seamlessly with the C6713 DSK and is described more fully below.

3. HARDWARE INTERFACE

The THS1206 EVM connects to the C6713 DSK via the the 5-6K Interface Board (see Fig. 1). The versatile THS1206 is a 12-bit, four-channel (four single-ended or two differential inputs), simultaneous sampling, ADC that is capable of operating at up to 6 Msps divided between the maximum of 4 channels. For the purposes of this teaching platform, we need only one channel, and to allow the higher range of input frequencies present in an FM IF signal the antialiasing RC filter in THS1206 EVM was modified. Specifically, the filter capacitor was changed from 1 nF to 100 pF. The cost for the THS1206 and the 5-6K board is approximately $50 each (see http://focus.ti.com/docs/prod/folders/print/ths1206.html for the THS1206 ADC and http://focus.ti.com/docs/toolsw/folders/print/5-6kinterface.html for the 5-6K). The HPI daughtercard costs $85 (see http://www.educationaldsp.com/).

4. DESIRED EDUCATIONAL PROCESS

We've briefly described the DSK-based hardware part of our educational platform that can implement first-order bandpass sampling of an FM IF signal as part of an SDR process. With the IF signal thus aliased down to baseband (albeit with inverse spectral placement), remaining DSP steps would include correcting the spectral placement (if desired) and demodulation of the FM signal. If our students were all proficient at DSP programming, they could just program the C6713 to implement these steps for the SDR system. However, while our students know how to use MATLAB, they are typically not what we would call proficient using TI's Code Composer Studio to program a C6713 in C. What we needed was a tool that allows for algorithm development in MATLAB. Once the students are comfortable with what they had learned using MATLAB, such a tool would facilitate the migration of the algorithm—in part or whole—into C and onto the DSP hardware. The desired progression would be as follows: 1) Study the traditional DSP theory, 2) use MATLAB with simulated data, 3) use MATLAB with real-world data, 4) implement the process (in part or whole) in real-time on the TI DSK hardware using C, and 5) repeat to improve the design or to develop new features.

The third step of this process presents a practical problem. While MATLAB has available a very capable data acquisition (DAQ) toolbox that is compatible with a number of different DAQ hardware boards and which allows for direct data acquisition into the MATLAB workspace, that toolbox does *not* support programmable DSP systems such as a C6713 DSK. Even if the DAQ Toolbox could be used with a DSK, you could not avoid the fact that this method would be too slow to allow the transition to step four: a real-time implementation. Since we wish to minimize multiple software environments in the interest of time (for students and faculty), a single development environment solution is highly desirable. For this reason, we developed a direct MATLAB-to-DSK interface.

5. MATLAB TO DSK SOFTWARE INTERFACE

The MATLAB-to-DSK real-time interface is a software tool that permits MATLAB to interface directly with a DSK. Data can be imported from the DSK inputs into MATLAB variables, and variables can be written to the DSK outputs. The data transfer capabilities are limited primarily by the bandwidth of the host PC to DSK connection, and the speed of the host computer. The software interface is encapsulated into a generalized command set that supports a variety of DSK models, multiple input and output channels, variable sample rates, various triggering configurations, and variable frame sizes. The interface was developed using MATLAB's "mex" facility and Microsoft Visual C++, and is centered around an object that encapsulates the hardware interface between the host PC and the DSK. The TI application programming interface furnished
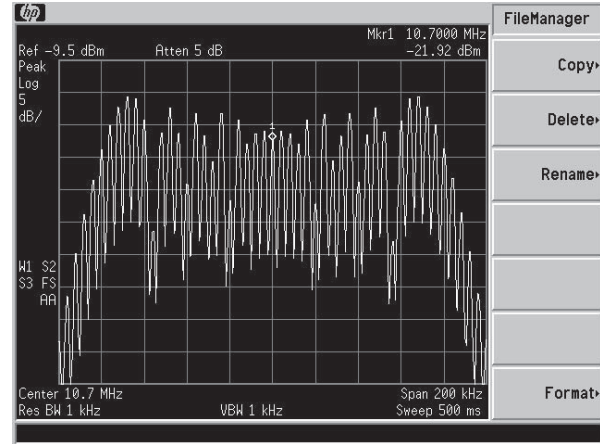

Fig. 2. Spectrum analyzer display of our simple FM IF signal: 10.7 MHz frequency modulated with a 4 kHz test tone.

with the DSK allows operation under Windows XP and Vista.

Our interface software requires that the DSK tools be installed on the computer, that the two files `C6X_DAQ.DLL` and `DAQ_SIMUL.OUT` be placed in a MATLAB-accessible directory, and that the HPI daughtercard is installed on the DSK. At the most basic level, this interface allows a novice user to operate the DSK as a data acquisition board with a simple command sequence, with no requirement to know how to use Code Composer or how to program in C. Initially, all signal processing can be done in the MATLAB environment using "live" data acquired from the DSK. As the students progress, they can move processing functions from MATLAB down to the DSK by altering the DSK code (that was used to create the `DAQ_SIMUL.OUT` file), and still continue to use MATLAB as a graphical display engine.

The MATLAB-to-DSK real-time interface driver software, example MATLAB scripts, and descriptions of the interface functions are provided with [10]. The software can also be downloaded from the website given at the end of this paper.

6. CLASSROOM USES OF THE SYSTEM

For the first use of this sofware/hardware educational platform as a vehicle to show bandpass sampling for SDR, we chose a very simple signal for demonstration. We frequency-modulated a 10.7 MHz "carrier" with a 4 kHz test tone and filtered the result to a 200 kHz bandwidth (see Fig. 2). We then sampled this signal at $F_s = 400$ kHz as discussed above using the DSK and the MATLAB-to-DSK real-time interface software in "data acquisition" mode to capture the sampled (and aliased) signal directly into the MATLAB workspace.

At this point, our students are at step 3 of the five-step process described above in Section 4, so the final demodulation of the bandpass sampled signal takes place in MAT-
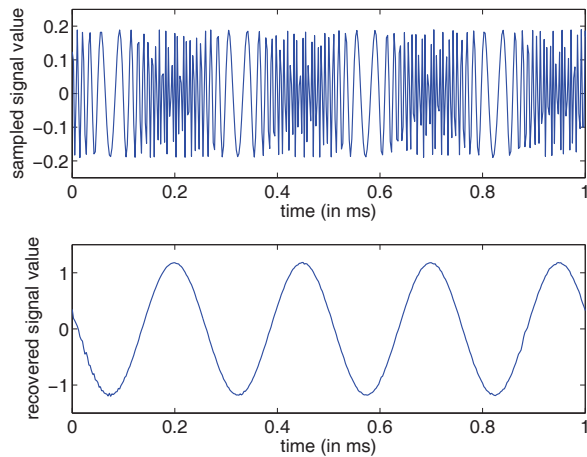
Fig. 3. Time domain display of FM IF signal modulated with 4 kHz test tone (top) and recovered test tone (bottom).

LAB. Once they are comfortable with their chosen MATLAB approach, they will then proceed to step 4 (real-time DSP implementation in C). While the students could take various approaches to demodulate the bandpass sampled signal using MATLAB, one elegant method is to create an analytic signal (i.e., I and Q components) using the Hilbert transform, then take the derivative of the angle of the $I + jQ$ vector. This requires just a few lines of MATLAB code as shown below.

```
zeroMeanData=data' - mean(data);
analyticSig=hilbert(zeroMeanData);
message=diff(unwrap(angle(analyticSig)));
message=message - mean(message);
```

See Fig. 3 for the before-and-after result of this algorithm.

This teaching example provides a good opportunity for asking the students to discuss the ramifications of the inverse spectral placement resulting from choosing an even value of n for Equation (1) above. After some lively discussion, and hands-on experimentation, they should conclude that for audio it makes no difference to the listener.

We plan to expand this to capturing commercial FM signals over the air, tapping the IF signal, and using similar techniques to extract RBDS information from the FM broadcast.

7. CONCLUSIONS

We have developed an educational platform that allows our students to better understand bandpass sampling and the basics of software defined radio using an inexpensive combination of software and hardware. Using this approach keeps our students interested, and allows real-world data to be gathered and used in the algorithm development and design process while maintaining a link to MATLAB. The transition to real-time DSP using the C6713 DSK is thus made much smoother than would otherwise be the case.

The authors freely distribute the associated software for educational, non-profit use, and invite user suggestions for improvement. See the URL `http://eceserv0.ece.wisc.edu/~morrow/software/`.

8. REFERENCES

[1] C. H. G. Wright, T. B. Welch, M. G. Morrow, and G. Vinyard, "CommFSK: A hardware approach to teaching FSK," *ASEE Comput. Educ. J.*, vol. XVIII, pp. 38–45, April–June 2008.

[2] T. B. Welch, C. H. G. Wright, and M. G. Morrow, "Caller ID: A project to reinforce an understanding of DSP-based demodulation," *ASEE Comput. Educ. J.*, vol. XVI, pp. 2–7, Oct. 2006.

[3] J. H. Reed, *Software Radio: A Modern Approach to Radio Engineering*. Prentice Hall, 2002.

[4] f. j. harris, *Multirate Signal Processing for Communication Systems*. Prentice Hall, 2004.

[5] T. B. Welch, C. H. G. Wright, and M. G. Morrow, "Teaching rate conversion using hardware-based DSP," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. III, pp. 717–720, Apr. 2007.

[6] T. B. Welch, T. Kent, C. H. G. Wright, and M. G. Morrow, "Teaching with software defined radios," in *Proceedings of the 2009 ASEE Annual Conference*, June 2009.

[7] R. G. Vaughan, N. L. Scott, and D. R. White, "The theory of bandpass sampling," *IEEE Trans. Signal Processing*, vol. 39, pp. 1973–1984, Sept. 1991.

[8] J. Liu, X. Zhou, and Y. Peng, "Spectral arrangement and other topics in first-order bandpass sampling theory," *IEEE Trans. Signal Processing*, vol. 49, pp. 1260–1263, June 2001.

[9] M. A. Yoder, J. H. McClellan, and R. W. Schafer, "Experiences in teaching DSP first in the ECE curriculum," in *Proceedings of the 1997 ASEE Annual Conference*, June 1997. Paper 1220-06.

[10] T. B. Welch, C. H. G. Wright, and M. G. Morrow, *Real-Time Digital Signal Processing: From MATLAB to C with the TMS320C6x DSK*. CRC Press, 2006.

[11] M. G. Morrow, T. B. Welch, and C. H. G. Wright, "A host port interface board to enhance the TMS320C6713 DSK," in *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing*, vol. II, pp. 969–972, May 2006.