PC-BASED REAL-TIME DOPPLER SIGNAL PROCESSING AND DISPLAY USING TMS320C62 AND USB INTERFACING

Karim El-Laithy, Abou-Bakr M. Youssef and Yasser M. Kadah
Biomedical Engineering Department, Cairo University, Giza, Egypt

Abstract – **This study demonstrates a DSP, Digital Signal Processor, and USB, Universal Serial Bus, based system for Doppler signal real-time analysis. The system output is the velocity profile of the vessel's section under examination. The target of the ultrasound waves is a Doppler string phantom designed and implemented specially for this study. DSP kit is responsible of all the processing applied to the digitized input Doppler signal. USB kit is responsible of transferring the processed data blocks from the DSP kit memory to the PC memory for display and reporting. Simultaneously, USB kit controls the string phantom parameters to simulate certain blood flow profiles. The results show that DSP cops with the required processing power. However, the use of USB kit for both data transfer and phantom control lead to poor display at the PC.**

Keywords – **Doppler, DSP, USB, signal processing**

I. INTRODUCTION

Doppler Ultrasound is used as a technique for non-invasive assessment and measurement of the velocities of moving structures inside the human body, particularly blood flow [1]. The combination of anatomical imaging with vascular imaging makes the use of this technique a routine in common clinical diagnosis procedure in many areas.

The spectrogram, or velocity profile, shown in Fig 1, is a common evaluation tool for the Doppler data. It displays as a 2D image the frequency or the corresponding velocity versus time at a particular spatial region of interest. The brightness of the point, or the third dimension, indicates how many particles is moving with this velocity at the section under examination. To calculate this spectrogram, the data should be viewed in the frequency domain to get red of the undesired frequency bands, and then calculate the power spectrum. Further processing is needed to reconstruct the filtered Doppler signal to output it as an audible signal.

The real time velocity estimation process, basically the evaluation of the velocity profile, has a great role in the diagnosis of the circulatory system improper functioning especially in cardiac applications. The processing power needed to implement such a real-time signal processing system is quiet substantial [1]. It involves the acquisition and processing of huge amounts of raw data to extract an accurate estimate of the velocity. Given the variability of velocity profiles with time especially near the heart because of the pulsatile nature of blood circulation, the processing time available to follow such variations is rather limited and usually should be complete in a few milliseconds.

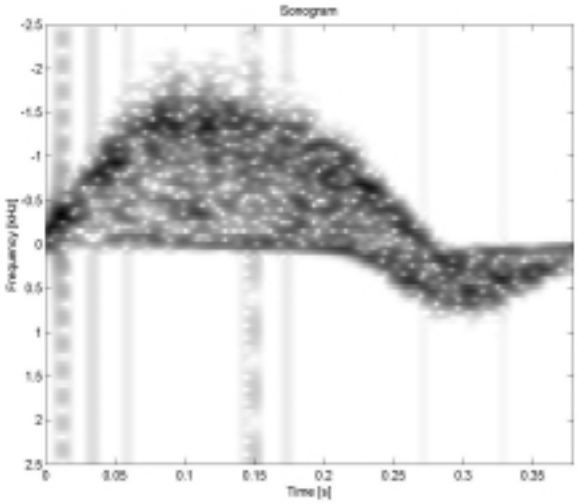

Fig. 1. Sample spectrogram

In a number of applications, engineers sought the integration of the versatility of PCs in generating user-friendly graphical interfaces. The progress in PC hardware has added to even its most modest configurations a formidable processing power that can be utilized for various purposes. Among the foremost of these is its use in medical imaging consoles and in particular ultrasound imaging. Even though such configurations were proven sufficient for such applications as B-mode ultrasound imaging, the processing needed for Doppler signal processing was not possible to implement in real-time.

DSPs proved excellence in performing stable and huge amount of embedded signal processing with minimum time delays [1]. Instead of attempting to create the user interface from embedded components, it makes more sense to combine DSP chips and PCs to complement the current PC-based ultrasound imaging configurations. Here, we present a report of a successful integration of this kind. In particular, we acquire the data using an A/D converter connected to a DSP chip where it is processed and the desired spectrogram line is generated. Then, using a modern USB interface, the data are transferred to the PC where it is displayed on the screen. This study integrates all these aspects to come out with a prototype for real-time Doppler signal analysis.

II. System Components and Setup

The experimental setup was designed using the following basic components:

- DSP kit: Texas Instruments DSP Kit, C6211 Platform, Fixed Point Precision TMS320 C6000 series [3]
- USB kit: Cypress EZ-USB kit. [4]
- PC: IBM Compatible PC, Pentium III 640 MHz, 64 MB RAM.
- Doppler String Phantom

The system interconnections are parallel-like from the abstract point of view. The input signal is fed into the DSP kit via the microphone input socket as an audible signal. Then, all the processing is applied to the raw data to extract the features for further display. This output data are transferred from the DSP kit to the PC for display via USB interface. The output data is transferred from the DSP chip to the USB interface via parallel interface with 16-bit data word. The 16 digital lines carrying this data were taken from the expansion slot on the DSP chip board. This slot is used basically to connect the kit to a Daughter board. This expansion slot is HW mirror for the EMIF, expansion memory interface, in the DSP chip. The USB ports are 8 bit, so the data word is read using two ports each one at a time. Then the data are carried to the PC Memory via USB standard interface.

Instantaneously, the USB interface, with a stored program on its EEPROM, was controlling the speed and direction of the steeper motor, which is the control part of the Doppler string phantom. However, as a detailed look, for controlling and monitoring all these actions two more PCs were needed for controlling the two system components independently. The DSP chip is controlled via the TI Code Composer Studio. The Ez-USB kit is controlled via Ez-USB Cypress Tool Kit. But this need is not supposed to be needed in fully developed system.

III. Doppler Signal Processing

The reflected ultrasound signal is received from the body or the phantom under examination. This signal carry the Doppler information, the signal with different frequency component depends on the target velocity. The signal is demodulated in the receiving electronics, not implemented in this case, and output a signal carrying only the Doppler information. As the Doppler shift is a change in the transmitted frequency depending on the target velocity [2]; the target in this case is the Doppler string phantom or the biological blood in the human body with max velocity of 100 cm/s and according to the Doppler shift relation [1]. Hence, the maximum frequency shift lies in the audible range. This audible signal is fed as an input to the system, raw material.

The output of the system is: a) the velocity profile, or sonogram, on a PC monitor and b) an audible Doppler signal on a loudspeaker. The data, audible signal, is fed into the DSP chip. At this stage all the coming steps are handled with in the DSP kit itself.

1. The analog input signal is digitized with an on board analogue to digital converter at sampling frequency 8 KHz, which implies that, the maximum frequency component may be contained in the signal is 4 KHz
2. Applying windowing to the data stream by taking the data as segments that each one length is 128 samples and sent it for further processing.
3. Apply the FFT transform to the data segment to get the underlying frequency spectrum in the signal.
4. By setting the cutoff frequency of the WMF, wall motion filter, and the DLC, delay line canceller, and use it on the data from the FFT; hence the signal is free of irrelevant components. The filters used here were frequency domain filters.
5. Here the data flow split into two pathways, the first, is calculating the spectrogram data by calculating the power spectrum of the FFT data. This power spectrum could be obtained by squaring the absolute data value of the output from the FFT process. Second, the data is input to the inverse Fourier transform process (IFFT) to reconstruct the signal. This signal is carried to the digital to analogue converter to be output as analogue output to the audio speaker out
6. The power spectrum data is sent digitally via certain paths in the DSP chip to be gathered by an ordinary PC to display the results in an acceptable form.

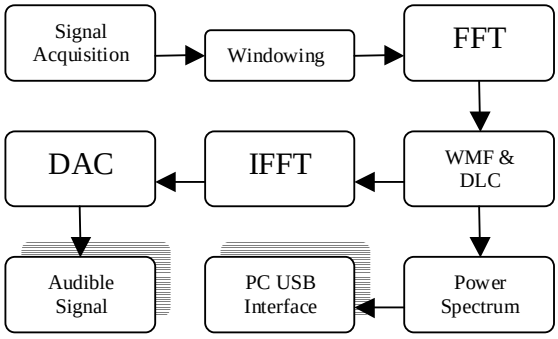

Fig 2. Overall system overview

The digital signal processor requires basic scheduling and I/O services to handle high-speed arithmetic, I/O and interrupt processing. The DSP/BIOS foundation software, included in Code Composer Studio, furnishes a small firmware kernel with basic run-time services that software developers can embed on target DSP hardware. Mainly threading and scheduling along with a data pipe managers are designed to manage block I/O (also called stream-based or asynchronous I/O. moreover, The SWI module manages software interrupt service routines, which are patterned after HWI hardware interrupt routines, are

triggered programmatically through DSP/BIOS API calls, such as SWI_post, from client threads. Once triggered, execution of a SWI routine will strictly preempt any current background activity within the program as well as any SWIs of lower priority; HWI hardware interrupt routines on the other hand take precedence over SWIs and remain enabled during execution of all handlers, allowing timely response to hardware peripherals with the target system. Software interrupts or SWIs provide a range of threads that have intermediate priority between HWI functions and the background idle loop.

The DSP/BIOS Buffered Pipe Manager or PIP Module manages block I/O (also called stream-based or asynchronous I/O) used to buffer streams of program input and output typically processed by embedded DSP applications. Each pipe object maintains a buffer divided into a fixed number of fixed length frames, specified by the number of frames and frame size properties.

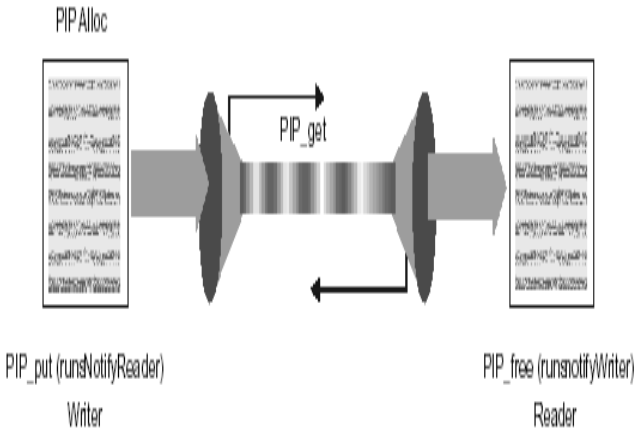

Fig 3. DSP/BIOS Data Pipes (PIP Module)

All I/O operations on a pipe deal with one frame at a time. Although each frame has a fixed length, the application may put a variable amount of data in each frame (up to the length of the frame). Note that a pipe has two ends. The writer end is where the program writes frames of data. The reader end is where the program reads frames of data.

Data notification functions (notifyReader and notifyWriter) are performed to synchronize data transfer. These functions are triggered when a frame of data is read or written to notify the program that a frame is free or data is available. These functions are performed in the context of the function that calls PIP_free or PIP_put. They may also be called from the thread that calls PIP_get or PIP_alloc. After PIP_alloc is invoked, DSP/BIOS checks whether there are more full frames in the pipe. If so, the notifyReader function is executed. After PIP_alloc is invoked, DSP/BIOS whether there are more empty frames in the pipe. If so, the notifyWriter function is executed.

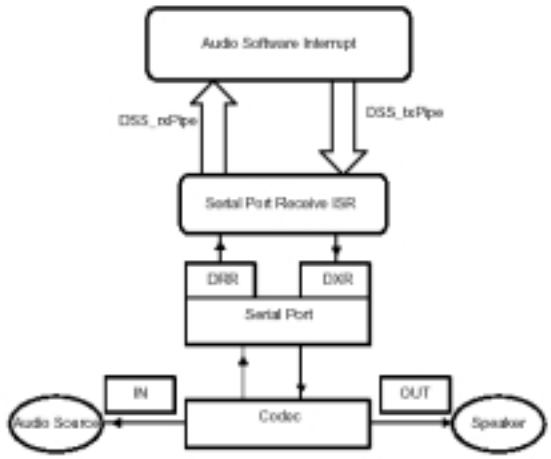

Fig 4. Program Flow, PIP based flow.

IV. THE DSP-USB INTERFACE

The most important registers are the EMIF global control register and the registers associated with memory spaces o f the expansion bus CE2 and CE3. We used the EMIF as asynchronous 16 bit output to the out world by setting the associated registers to the memory space and the global control register.

The most important signals were the Asynchronous Write Enable (AWE, Active LO) and the Chip Enable (CE, Active LO), which are used as control signals to control the flow of the data from the DSP kit to the USB development kit. The AWE signal used mainly to enable the latches (74LS373) that latch the valid data from the expansion slot on the DSP kit to the output pins of the latches. The AWE is a strobe signal and has a very high transition rate which makes it so difficult to be detected and sensed, so, we take it after buffering it by a bi-directional buffer stage via the (74LS244). Practically we take the inverted signal of the AWE (take it after inversion) to the buffer to match the requirements of the latch. But for the CE signal it is mainly generated to select the CS pins of another memory device, so, its electrical characteristics allow for it to be detected or sensed by another device. This signal is used in our system to enable the acquisition by the USB to a new 16-bit sample from the DSP.

Note that without the latches, it is almost impossible to read from the expansion bus of the DSP kit because of the very rapid level transition. Address decoding may be needed if the EMIF is used to communicate with more than one device, hence, to select the desired device, the address generated by the 20-bit address bus on the expansion bus could be used for this purpose.

V. Doppler String Phantom

The purpose of this device is to provide predefined Doppler velocity parameters as a target for the ultrasound beam in order to test the performance of the setup system. A stepper motor is mounted to control the speed and the direction of a string well tied to move around four bullies. The USB interface was used to control the motion of the string in fluid medium, basically water. The motor itself is driven by four power transistor for current supply. The logical levels out form USB kit board synchronize the four transistors operation to provide the desired flow parameters.

Two lookup tables are stored in the EEPROM of the USB interface for sinusoidal and Forward-Backward flows. The stepper motor used was not designed specially for this purpose. This limited the flow profiles that could be implemented with it. The main problem with this motor was the large inertia that blocked the smooth transition from one direction to another or the rise time for maximum speed.

VI. Conclusions

A real-time data processing system based on a combination of an embedded DSP chip and a PC interfaced using the standard USB interface was developed. The system allows the calculation of Doppler velocities in a timely manner while utilizing the PC as a versatile display device. The improvement of the current system by incorporating new floating point DSP chips and extending the USB interface to its version 2.0 enables the current system to address the most computationally demanding tasks including color Doppler and its variants. The simplicity and low cost of the hardware involved suggests the possibility of making many of the high-end Doppler system features available at a much lower price. Further investigation of this possibility is therefore warranted given the potential enhancement of health care delivery efficiency.

References

[1] D. H. Evans and W. N. McDicken, *Doppler Ultrasound Physics, Instrumentation and Signal Processing, 2nd Ed.*, John Wiley and Sons, New York, 2000.

[2] J. A. Jensen, *Estimation of Blood Velocities Using Ultrasound : A Signal Processing Approach*, Cambridge University Press, Cambridge, 1996.

[3] *TMS320C6211 Reference Guide*, Texas Instruments, 2000.

[4] *USB 1.1 Interfacing Using Cypress EZ-USB chipset*, Cypress Semiconductors, 2001.