Digital Signal Processing with the SHARC

Rulph Chassaing / Roderick Ayers
Roger Williams University / Naval Undersea Warfare Center Division
Bristol, RI 02809/ Newport, RI 02841

Abstract

This paper introduces the use of Analog Devices' latest generation of floating-point digital signal processors, the Super Harvard Architecture Computer (SHARC). The SHARC is well suited for a wide range of digital signal processing (DSP) applications. It is particularly useful for implementing complex algorithms such as the fast Fourier transform (FFT) and for developing applications requiring multiple processors in order to satisfy large processing requirements. Features of the SHARC along with available hardware and software support tools are presented. Examples in FIR/IIR filtering and the FFT, implemented using both simulation and real-time execution, further illustrate the use of the SHARC.

Introduction

Digital signal processors are special-purpose fast microprocessors with specialized instruction sets appropriate for signal processing. These devices, made possible through advances in integrated circuit technology, are found in a wide range of applications such as telecommunications, speech processing, etc. In recent years, both fixed and floating-point digital signal processors have emerged from a number of companies such as Texas Instruments, Motorola, and Analog Devices. Those digital signal processors now have architectures that facilitate the use of high-level language compilers. C has become very popular as the high-level language of choice. The use of a high-level language often results in a substantial reduction in execution speed. Alternatively, carefully prepared assembly language programs produce faster executable code, but they are more difficult to document and maintain. The compromise is to code time-critical routines in assembly language that can be called from C. Companies such as Ixthos provide libraries of optimized functions for the SHARC that can be called from C.

Over the last ten years, senior students at Roger Williams University (RWU) have implemented a wide range of real-time DSP applications in areas such as communications, controls, and adaptive and multirate filtering. Those applications, based on both Texas Instruments' floating-point TMS320C30 and fixed-point TMS320C25 digital signal processors, are described in references 1 and 2.

Investigation of the SHARC was undertaken at the Naval Undersea Warfare Center (NUWC) to determine its suitability for large processing requirements.

The SHARC and Supporting Tools

The SHARC, one of today's most powerful general purpose digital signal processors, is manufactured by Analog Devices Inc. (ADI), and is the latest generation of the ADSP-21OOO family of floating-point digital signal processors. Two versions of the SHARC exist: the ADSP-2106O with 128K x 32 bit words of on-chip RAM and the ADSP-21062 with 64K x 32 bit words of on-chip RAM. Key features include a large on-chip memory, six link ports which can be used to interconnect to other SHARCS for multiprocessing, and a 25-ns instruction execution time, for a peak performance of 120 MFLOPS. Figure 1 shows a functional block diagram of the ADSP-2106O SHARC. Additional features follow:

The program memory (PM) address bus is 24 bits wide which allows access of up to 16M words of mixed instructions and data. The data memory (DM) address bus is 32 bits wide which allows access to 4G words of data.

A large set of 16 primary and 16 secondary data registers which allows for support of 32 circular buffers of arbitrary size. A circular buffering scheme is used to update samples in filtering as well as for bit reversal in an FFT algorithm.

"DO loops", as in Fortran programming, are supported without any overhead.

Delayed branch instructions eliminate overhead, effectively executing as a single-cycle instruction. Branching instructions such as CALL, JUMP, or RETURN can be executed in parallel with a computation.

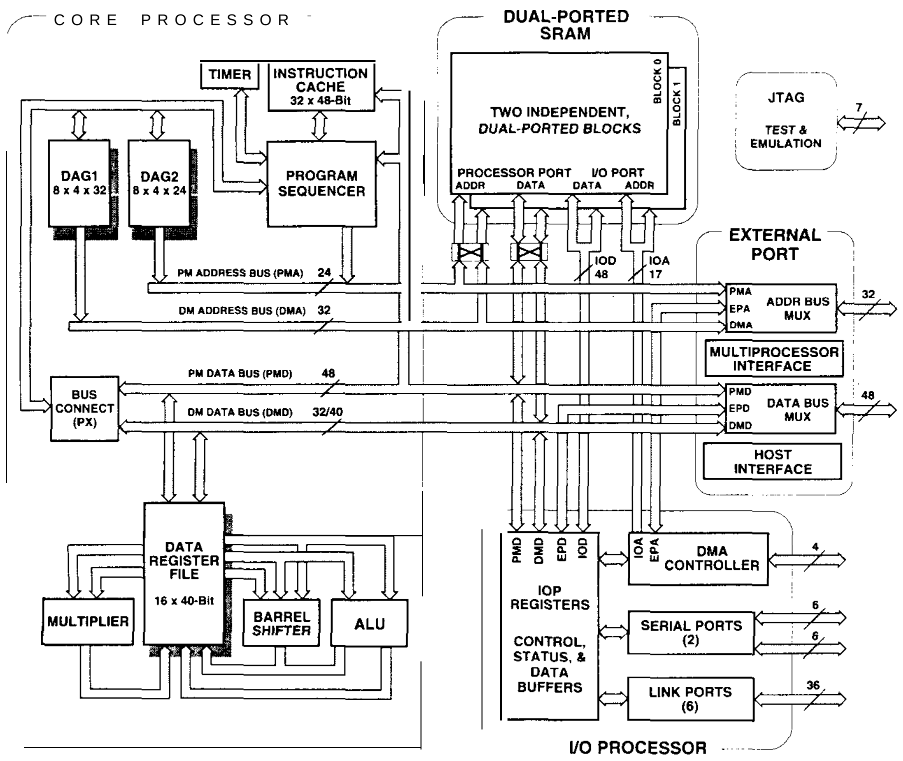

Figure 1. ADSP-2106O SHARC functional block diagram

The DM stores data operands and the PM stores both instructions and data, allowing dual data fetches with the instruction supplied by cache, achieving single-cycle execution of dual-operand instructions. For an instruction requiring two data accesses, the PM bus is used to access data from the mixed block and the DM bus is used to access data from the data-only block, with the instruction fetched from cache. A 32-word instruction cache enables a 3-bus operation for fetching an instruction and two data values by caching instructions that conflict with data access in program memory 3.

Single-cycle execution is maintained when one of the data operands is transferred to or from off-chip memory via the external port. The external port provides access to off-chip memory, as well as access to the internal memory of other processors.

A unified address space allows direct interprocessor accesses of each SHARC'S internal memory. A cluster of up to six SHARCS can directly access each other's RAM's using an ID scheme of addressing. Using the ID of another processor, one can write to the internal memory of that processor. Broadcast writes allow simultaneous transmission of data to all SHARCS in a single cluster.

Communication between processors is performed either through point-to-point communication channels via six link ports or through a single shared, global memory via a parallel bus (cluster multiprocessing).

There are two serial ports, each with a maximum data rate of 40 Mbits/see, which are used for 1/0 access to external devices.

External memory space is divided into four banks. Mapping peripherals into different banks can accommodate 1/0 devices with different timing requirements, since each bank has its own wait state generator.

Ten DMA channels allow for basic operations of external port block data transfers and 1/0 (link, serial) port data transfers.

There are a number of support tools currently available for the SHARC[3-11]. These tools include an ADSP-21062 SHARC-based board (EZ-LAB) which plugs into a full slot on an IBM compatible PC. The board, developed by Bittware Research Systems, contains an Analog Devices AD1 847 codec for real-time input and output. An emulator and a board with one or two SHARCS are also available from Bittware. Software tools, available from ADI, include a C compiler, an assembler, a linker, and a simulator.

Hypersignal for Windows Block Diagram, a visual design tool which can be used to implement and test DSP algorithms, is available from Hyperception. DSP designs can be executed on a host PC or on the SHARC-based board using a real-time driver option. Drivers are also available for other processors such as the TMS320C30. Block Diagram contains a wide and extensive range of functional building blocks such as the FFT, correlation, filtering, as well as image processing and communications functions. The user can select the functional block icons, specify their parameters, interconnect them to implement a desired algorithm while monitoring and testing its functionality. A/D and D/A functional blocks for real-time input and output are also supported with the SHARC-based board. Different types of displays allow observation of waveforms at different points within the design. User controls enable parameters to be changed while the design is executing, providing a powerful and interactive visual design environment. An optional C source code generator produces the code for the complete design which can then be cross-compiled and ported to the SHARC or to another platform such as the TMS320C30.

Implementation

Several examples using both the simulator and the SHARC-based board were developed to establish familiarity with the SHARC and supporting tools.

A 1024-point FFT example demonstrates a program in C calling an optimized assembly function. A set of optimized assembly functions, callable from C, are available from Ixthos. An abridged version of Ixthos' library functions, included with the SHARC-based board from Bittware, contains a 1024-point complex FFT function. The 1024-point FFT was tested using different inputs and implemented using both the simulator and the SHARC-based board. Figure 2 shows the resulting simulator plot of the FFT of a 1024-point sine input function represented with 16 points/cycle. Figure 2 shows a “delta” function at the frequency of the input sinusoid (N=64) and another “delta” function at its folding frequency (N=960). The following parallel instruction, executed in a single cycle, is an example of the code syntax of the SHARC, used for implementing “butterflies” in an FFT algorithm section:

```
R12=RO*R7, R13=R8+R12, R1O=R8-R12, PM(I1O,M1O)=R3, R6=DM(I0,M0);
```

The execution of this instruction performs, in a single cycle, a multiply and the sum and difference of the same two inputs, stores one number in program memory and fetches another number from data memory. The “I” and “M” registers are used to post-increment or pre-increment a memory address.

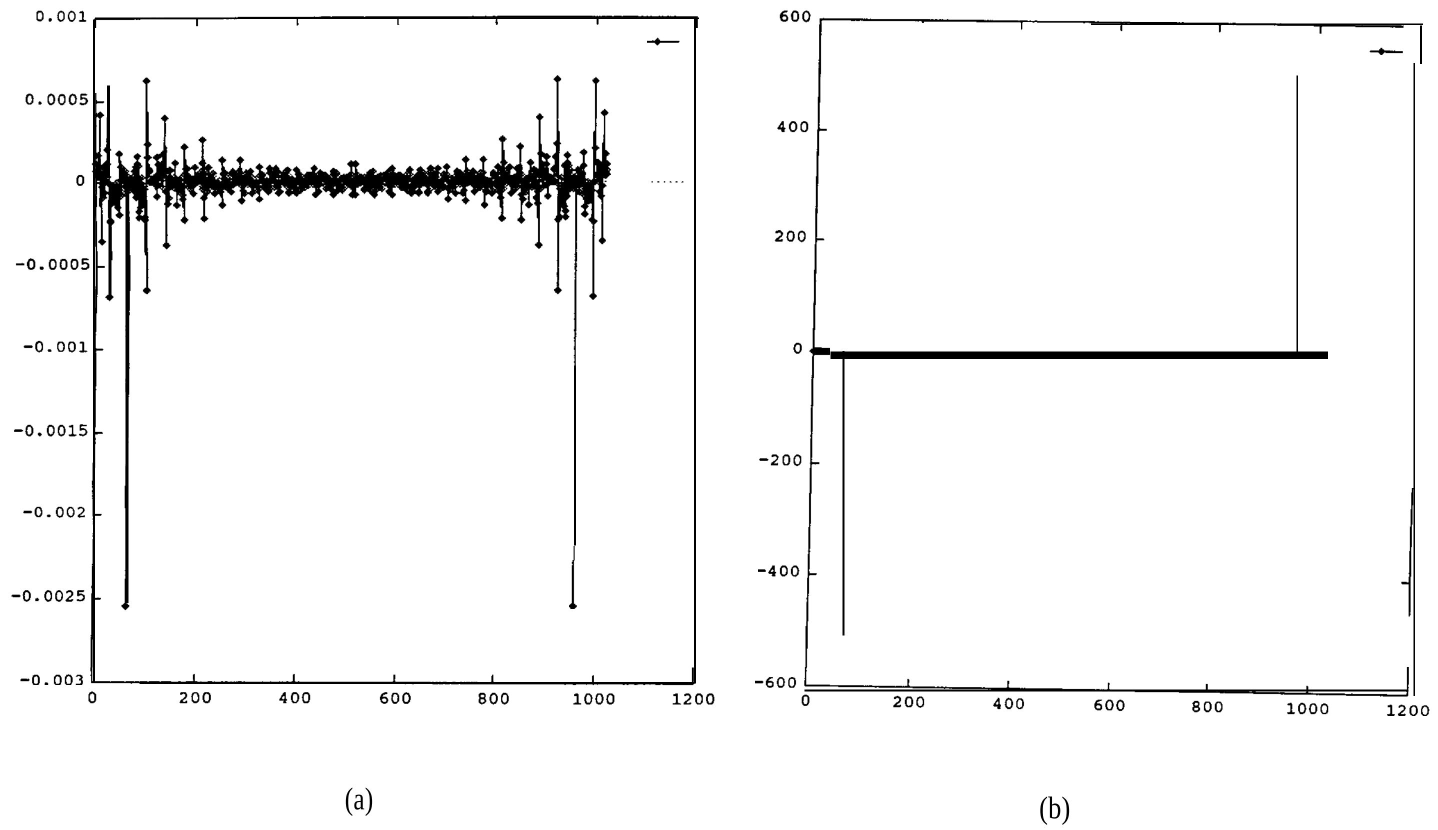

Figure 2. 1024-Point FFT of sine input function: (a) real output component; (b) imaginary output component

FIR and IIR filter examples were developed and tested on the EZ-Lab board. The following is an excerpt from an assembly code program which implements the convolution equation of an FIR filter:

```
                lcntr=Rl, do mac until Ice;
mat:            F12=FO*F4, F8=F8+F12, FO=dm(IO,MO), F4=pm(I8,M8); {mat/fetch}
```

The first line of code performs a "Do loop" command to execute the subsequent instruction R1 times, with R1 set previously to the number of filter taps less 1. In the second line of code, register F12 contains the product of an input sample value (stored in register FO) and a coefficient value (stored in F4). Accumulation is performed in F8. Within the same cycle, the next input sample value from data memory and coefficient value from program memory are fetched and stored in FO and F4, respectively, in preparation for the next product and accumulation.

C programs for FIR and IIR filters were also developed. Those programs are very similar to the C programs previously used with Texas Instruments' TMS320C30 processor, listed and described in reference 1 (pages 131,185). An input/output (1/0) communication routine was used as a "black box" and included in those filter programs. The 1/0 capability of the EZ-Lab board was tested by verifying, on a spectrum analyzer, the expected frequency responses of the filters.

Conclusions

Applications in digital signal processing can be implemented on the SHARC using both C and SHARC assembly code. The SHARC can be efficiently used to process time-critical optimized assembly functions such as the FFT, callable from C, as well as for a wide range of applications.

Acknowledgements

The support of the National Science Foundation and Roger Williams University's Research Foundation to the DSP Lab at RWU is appreciated. Valuable support was provided by the Naval Undersea Warfare Center (NUWC) through two U.S. Navy -ASEE Summer Faculty Research Programs (1994, 1995). Bob Senko, from Bittware, has been very helpful in providing the input and output (1/0) communication routine used with the SHARC.

References

1. R. Chassaing, *"Digital Signal Processing with C and the TMS320C30"*, Wiley, 1992.
2. R. Chassaing, D.W. Homing, *"Digital Signal Processing with the TMS320C25"*, Wiley, 1990.
3. ADSP-2106X SHARC User's Manual, Analog Devices, 1995.
4. ADSP-21OOO Family Assembler Tools and Simulator Manual, Analog Devices, 1993.
5. ADSP-21OOO Family Applications Handbook, Vol. 1, Analog Devices, 1995.
6. ADSP-21 000 Family C tools Manual, Analog Devices, 1994.
7. AD SP-21OOO Family C runtime Library Manual, Analog Devices, 1994.
8. ADSP-2106X SHARC EZ-Lab Development System Manual, Analog Devices, 1995.

9. IXLibs-21k Optimized DSP Libraries for ADSP-21OXX User's Manual, Ixthos, 1995.
10. DSP/MSP Products Reference Manual, Analog Devices, 1995.
11. Serial Port 16-bit SoundPort Stereo Codec AD1847, Analog Devices, 1994.
12. P. Martin and R. Chassaing, "Parallel Processing with the TMS320C40", Proceedings of the 1995 ASEE Annual Conference.

RULPH CHASSAING, currently a professor at Roger Williams University, received the PhD (EE) from the Polytechnic Institute of New York. He is the author of the text: *"Digital Signal Processing with C and the TMS320C30"* and *coauthored with Dr. D. W. Horning "Digital Signal Processing with the TMS320C25",* both published by Wiley (1992, 1990). He also published a number of articles on real-time applications in DSP.

RODERICK AYERS received a Bachelor of Science degree in Electrical Engineering from the University of Maine in 1980. He is currently employed by the Naval Undersea Warfare Center Division of Newport, Rhode Island.