

FPGA BASED REAL TIME SYNTHETIC APERTURE SONAR PROCESSING FOR AUVS

Simon Banks Bloomsbury DSP Ltd
Simon Charles Bloomsbury DSP Ltd.
Allan Willcox Bloomsbury DSP Ltd.

1 INTRODUCTION

Bloomsbury DSP has implemented a number of real-time Synthetic Aperture Sonar (SAS) algorithms using FPGA, General Purpose Processor (GPP) and a combination of GPP and FPGA technology. This paper compares the use of FPGA and GPP technology for the implementation of SAS algorithms. Discussion of other technologies such as Digital Signal Processors (DSPs) and Application Specific Integrated Circuits (ASICs) is also given.

It is shown that for certain algorithms, FPGAs can provide significant benefits over modern GPPs, such as an embedded Single Board Computer (SBC) or desktop PC, in terms of processing power per watt, physical volume and price. GPPs are shown to complement FPGA technology for control and non-systematic processing functions. The operational benefits of integrating FPGA based real time SAS processing into Autonomous Underwater Vehicles (AUVs) is discussed.

Over recent decades, SAS has received increasing attention for high-resolution imaging of targets on the seabed for mine countermeasures and the offshore industry. Not only does SAS offer improved seabed images, it also provides very accurate navigation information based on acoustic motion estimation techniques. Such information is complementary to other motion estimation techniques such as Inertial Navigation Systems (INS) and Global Positioning Systems (GPS).

Recent developments in AUVs have enabled unmanned survey missions to be carried out leading to potential cost savings in the offshore industry and reduced risk to personnel for mine countermeasures operations. A large number of research organisations and companies are now developing AUV technology, including the National Oceanography Centre in Southampton [1], Bluefin Robotics [2] and Kongsberg [3].

Despite the obvious synergy, the operational use of real time SAS equipped AUVs is far from widespread today. Indeed, the SAS equipped AUVs currently in existence are predominately research and prototype technology demonstrator vehicles.

One factor holding back the widespread use of SAS equipped AUVs is a scarcity of compact and low power computing platforms that are computationally powerful enough to process SAS data in real time. After looking at the need for real time SAS equipped AUVs, a comparison of SBC / CPU (Central Processing Unit) and FPGA technology is given. The paper then discusses using a combination of conventional CPU based processing and FPGA devices to facilitate a low power and physically compact real time SAS processing solution that is suitable for integration into a modern AUV.

Comparisons are given between the computation times that can be achieved using FPGAs and those achievable using a low power Pentium M based GPP. It is shown that some algorithms are more efficiently implemented on an FPGA, whereas other algorithms are more efficiently implemented on a GPP.

Why real time SAS and AUVs?

Synthetic Aperture Sonar offers the highest available resolution for long-range acoustic imaging: around ten times better than conventional and multi-beam side-scan sonar systems. In addition, SAS offers constant resolution with range, the ability to image around convex and concave curved

platform paths and the ability to produce multi-aspect images of a given target from the same data set [4].

Using acoustic motion estimation techniques [9], a Synthetic Aperture Sonar system can also act as one of the most accurate acoustic motion estimation sensors available. Indeed, without this accurate estimate of platform motion, SAS imaging over large synthetic apertures is not possible.

With so many potential advantages, it is little surprise that there is a growing interest in the use of SAS technology for many sub-sea applications. This includes both defence applications for the detection and classification of mines, and commercial survey applications in the Oil and Gas and offshore industries. The ability to provide both accurate motion estimation and high-resolution images makes SAS systems an ideal sensor payload for use on AUVs.

In the defence industry, an AUV facilitates the remote detection, classification, and in some cases neutralisation of bottom and tethered mines. As a result, many world navies are now actively looking at the use of AUV technology for use in Mine-Counter-Measures (MCM) operations. For the US navy, details are given in the UUV Master Plan [17]. To minimise mission time, there is a strong motivation to develop AUVs with the ability to automatically detect and classify mines, and make automatic navigation decisions based on the output of classification algorithms. Such decisions may for example involve circling a suspected mine-like-object to facilitate more accurate classification of the target. Other decisions may involve avoiding targets that are positively identified as a mine.

In the offshore industry, towed array and ROV technology is frequently deployed for survey and the monitoring of undersea infrastructure. Such platforms necessitate the use of many personnel and the hire of large survey vessels. AUVs provide a means to carry out surveys remotely, reducing survey time and costs.

AUVs are increasingly being used for remote sensing applications. In recent years the industry has seen significant advances in enabling technologies such as batteries and navigation systems. As a result, there are now a significant number of companies and research organisations developing and deploying AUVs for many different applications. Such AUVs range from cheap systems costing less than seventy thousand pounds to larger systems costing upwards of a million pounds.

There are therefore a number of very compelling reasons to integrate a real time SAS payload on an AUV. Not only does SAS provide high-resolution images, it provides a very accurate platform motion estimate that can either augment on-board navigation systems, or reduce the required accuracy, hence cost, of on-board navigation systems.

In order to effectively integrate real time SAS into an AUV, a processing solution that is both low power and compact is required. Because many SAS algorithms are very computationally intensive, this is a tall order. The following sections discuss the use of FPGA technology to perform systematic data processing tasks in order to produce a real time SAS system that is both low power and compact.

2 FPGA AND GPP TECHNOLOGY

2.1 Technology overview

Devices that process data in a similar manner to modern GPPs have existed for over half a century: the Von Neumann architecture, the most commonly used today, being first proposed in 1945 [5]. Granted, the first GPPs used valves and were high power, un-reliable, physically large and expensive.

A GPP is designed to be a single processing unit that can be easily programmed to perform any computational process by implementing common mathematical operators such as addition, multiplication and division in silicon (or multiple valves). The GPP is therefore extremely flexible,

and has found uses in almost every aspect of modern life. A single device can be used to perform a multitude of different functions without the need to modify the physical hardware.

On one hand, a GPP is designed to do everything. But looking at it another way, a GPP is not optimised for anything. In general, only one computational operation can occur at once meaning silicon for other tasks is idle for much of the time. Exceptions to this include Super Scalar Single Instruction Multiple Data (SIMD) architectures, which have multiple execution units with vector operands so can complete more than one computation per clock cycle. The Intel Integrated Performance Primitives (IPP) library, for example, can make use of SIMD for vector operations. Despite this, the level of parallelism with SIMD is still orders of magnitude less than that possible with an FPGA.

The performance of a GPP is often limited by the available memory IO. For data larger than the CPU cache, this means the system main memory bandwidth. For a typical PC this can be up to 10 Gbyte/second (burst rate). For small data sets, the faster CPU cache IO limits the performance. FPGA devices provide multiple internal memories; meaning internal memory IO at up to 153 Gbyte/second sustained is possible with for example a Xilinx Virtex 4 SX35 device.

Also, for a specific data processing application, a GPP comes with a large amount of unnecessary complexity, such as the requirement for an operating system, hard drive, motherboard chipset and unused interfaces.

Alternatives to a GPP include DSP devices and ASICs. DSPs are normally Harvard architecture CPU type devices that facilitate fast real time DSP applications. Whilst five years ago a DSP device would be a natural choice for real time data processing, more modern devices do not provide a significant improvement in performance. Over the same period, FPGA devices have shown significant improvements in performance for DSP applications, now exceeding the power of DSP device. This trend is expected to continue.

An ASIC is a custom piece of silicon that is optimised to perform one specific task. For many computational tasks, an ASIC will provide the highest available performance. However, there are obvious penalties such as lengthy design times, NRE costs, and the inability to modify the device if an algorithm change is required. Many ASIC designs cost in excess of a million pounds to develop.

Somewhere between the GPP and an ASIC lies the FPGA, first produced by Xilinx in the mid 1980's [6]. At a similar time, Altera [7] began producing Electronically Alterable Gate Array (EAGA) devices, which were later re-branded FPGA devices.

These devices can be re-configured to perform specific systematic computational processes in a fast, compact and low power device. Although not originally intended for DSP, modern FPGAs contain many features suitable for use in DSP applications. For example, a modern Virtex 4 device can perform up to 256G Multiply Accumulates (MACS) per second; around two orders of magnitude greater than that possible with a modern Intel Pentium D device. Although FPGAs are not as flexible as GPPs, for some tasks they can provide a significant increase in performance.

Other vendors such as Actel, Lattice Semiconductor and QuickLogic now produce similar re-programmable digital ICs. Instead of going through the costly and risky process of developing an ASIC, the FPGA allows engineers to rapidly develop application specific data processing.

For many designs the theoretical maximum FPGA performance is not reached. This is partially due to routing problems in the device that necessitate running the device at a reduced clock rate, and the fact that most algorithms cannot be structured such that all computational elements are operating concurrently. In spite of this, we can generalise that a modern FPGA device provides around 100 times the computational power per watt than a modern GPP for systematic processing tasks.

2.2 Design flow

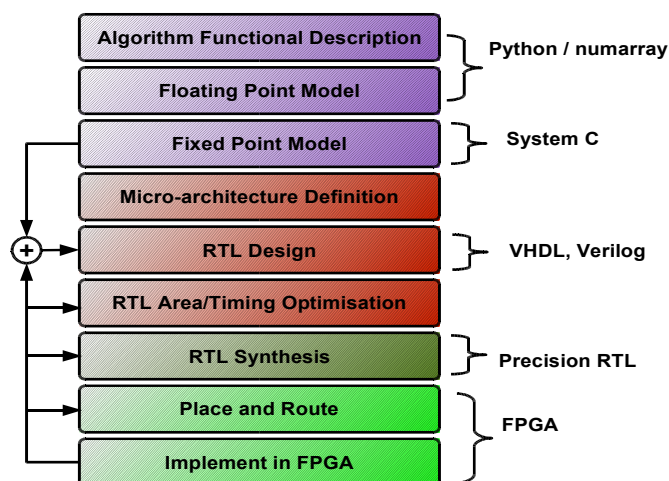


Figure 1 Typical design flow for FPGA development

The typical design flow for developing FPGA code is shown in figure 1. For new or complex algorithms, it is usual practice to define the algorithms in a fixed-point software model. This is generally done using C or C++/SystemC. A functionally identical design is then implemented in a hardware description language such as VHDL or Verilog.

The implementation of algorithms on an FPGA device can be significantly more time consuming than writing software for a CPU. This is because of a number of reasons, including the fact that data is almost always represented using integer or fixed point data types. The use of floating point data consumes large amounts of FPGA resources and seldom leads to an efficient design. Block floating point processing, where the same exponent is used for a large section of data, is commonly used as a compromise between full floating point and the limited data range of fixed point numbers.

2.3 Summary

When designing a system for modern high performance computing, it is prudent to account for trade-offs between different technologies such as CPUs/SBCs and FPGAs, DSPs and ASIC. The most appropriate solutions are often provided by heterogeneous systems made from a combination of, for example, GPP and FPGA technology. For this reason, many companies are now producing high performance computing platforms with FPGA devices closely coupled to a CPU. Such companies include Cray[15] and Spectrum Signal Processing [16] amongst others.

3 SYNTHETIC APERTURE SONAR ALGORITHMS

By integrating sonar data from several pings, SAS processing achieves increased azimuth resolution compared with conventional side-scan sonar. Using fast time domain imaging techniques such as FFBP, SAS also offers the possibility to image using non-linear apertures, such as circular apertures [4]. As described above, such techniques are of significant interest to SAS equipped AUVs used for mine countermeasures operations. SAS also offers the possibility of multi-aspect views of a target, which can significantly aid the performance of classification algorithms for radar [8] and potentially sonar too.

To form a good SAS image, it is essential to have an accurate measurement of the shape of the synthetic aperture. Compensation must also be made for changes in the medium. Unwanted vehicle movement affects the shape of the synthetic aperture with a free-swimming vehicle (such as an AUV). It is common practice to use acoustic based motion estimation techniques, sometimes

termed micro-navigation [9] to provide accurate motion estimates. INS systems are sometimes combined with acoustic micronavigation to further enhance the motion estimate.

SAS processing is almost always used with broadband sonar pulses, which are matched filtered to achieve comparable range and azimuth resolutions.

A typical SAS processing chain considered in this paper is shown in figure 2.

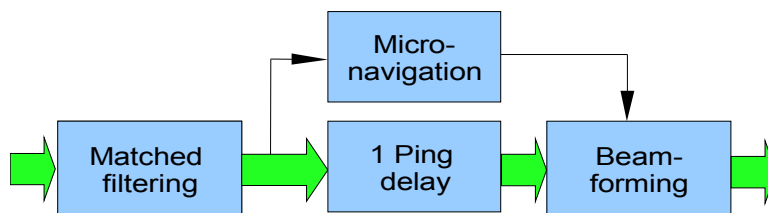


Figure 2 Typical SAS processing chain

This section presents performance figures for a Pentium M and an FPGA implementation of matched filtering. An overview of Micro-navigation and Beam-forming is also given.

3.1 Matched filtering

Matched filtering involves convolving the received sonar data with the transmitted pulse replica. The process is exactly the same as that used in an FIR filter, however the transmitted pulses are generally much longer than a typical FIR impulse response. For long pulses (over 128 samples) this process is most efficiently performed in the frequency domain [10]. Two common techniques for performing this process are: overlap and add, and overlap and save. These process are described in many textbooks on signal processing, including [10] and [11].

In this paper we consider the use of the overlap and add technique. This technique is employed by the `fftfilt` function provided in Octave [12] and MATLAB [13] for example. The data flow for overlap and add is shown in figure 3 where typically N is 2K to 16K words and $P \leq N/2$.

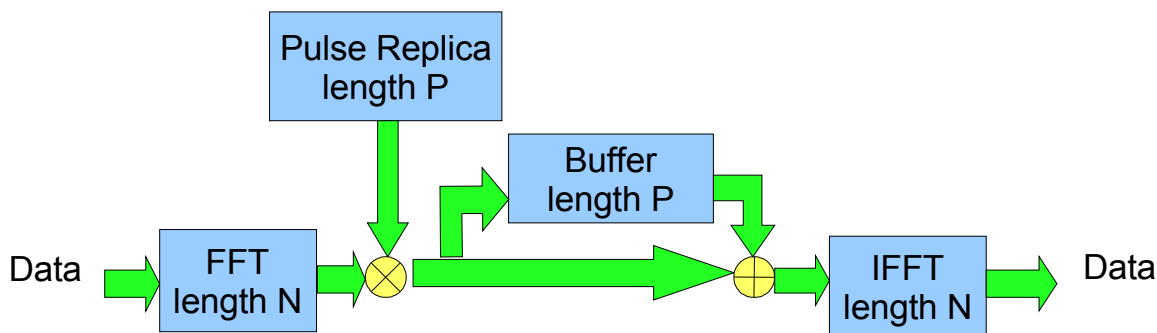


Figure 3 Data flow for overlap and add matched filtering

3.1.1 Assumptions on system parameters

The following parameters have been used for the for comparison of FPGA and Pentium M technologies:

Pulse length P : 1536 samples
 FFT Length N : 4096
 Baseband sampling frequency : 75 kHz
 Data representation : 32 bits per sample (16 bits real, 16 bits imaginary)

3.1.2 Software implementation

In software, each process shown in **figure 3** has to be performed separately. For each block of 4096 input samples, the CPU must perform a forward FFT, followed by a complex vector multiply (length 4096), a vector addition (1536 samples) and an inverse FFT (4096 samples). The process repeats for each block of 4096 range cells.

Two implementations are compared: one using the FFT published in Numerical Recipes in C [11], and a second using the Intel Intergrated Performace Primitives (IPP) library [14]. Each implementation was benchmarked by averaging the computation time over 2400 channels each of length 20,000 samples.

3.1.3 FPGA implementation

Each processing block shown in **figure 3** can be implemented separately in the FPGA. Due the nature of the algorithm, the whole system can stream data continuously at the specified clock rate. Xilinx quote a maximum clock rate of 500MHz for the Virtex 4 range of devices. However, this is difficult to achieve in a real design so we use the more conservative clock rate of 150MHz for the purposes of comparison. The performance of the FPGA implementation is deterministic and easy to calculate, because one sample is output every clock cycle. Using a 150 MHz clock, samples are therefore output at 150E6 times per second.

3.1.4 Results for matched filtering

	1.6 GHz Pentium M Single board computer§§		FPGA implementation Xilinx XC4VSX35-10FF668 C
	Standard	IPP optimised	
Sample throughput rate (MS/s)	2.823	9.411	150
Data throughput Assuming 4 bytes per sample, Input + output (MB/s)	22.584	75.288	300
Real time sonar channels at 75 kHz sampling	37	125	2000
Power consumption (W)	40 §		3**
Unit cost (GBP)	447.74 + OS		264*
Physical Size (mm)	203x106		32x32
Deterministic performance	Requires real time operating system		YES

*price of single device from Avnet converted to GBP using rate at time of writing. This does not include PCB.

**Estimate based on data from Xilinx

§ Measurement of similar device running at 100% CPU including hard drive

§§ Based on SBC - EBC-572 from Amplicon with 1.6 GHz Pentium M and 128M RAM from Scan computers.

Table 1: Performance comparison of FPGA and Pentium M single board computer pulse compression system

The computation rates for the two software implementations and the FPGA implementation of overlap-and-add pulse compression are shown in table 1. Table 1 also includes a comparison of price, power consumption and physical dimensions.

For the purposes of comparing technologies, it is interesting to note the speed increase of over three times using an optimised software implementation compared with a standard implementation. Similar differences can also be seen with FPGA technology, where poor design practices can lead to significantly reduced clock speeds.

The data IO rate has been shown, as using a PC the maximum rate is likely to be limited by the system's PCI bus. With standard 33MHz 32 bit PCI, the maximum sustained data transfer rate is around 80 MB/s, meaning even with a faster CPU it is likely that a SBC type system would be limited by IO and not processing speed.

The FPGA device is capable of significantly higher data IO rates. With the Virtex 4 XC4SX35 offering 448 IO pins, at 150 MHz clocking this provides 8.4 Gbytes/second single ended, 105 times faster than a typical SBC. The Xilinx Virtex 4 LX family also offers the possibility of serial IO, the largest device providing 24 x 250 Mbyte/s RocketIO transceivers.

3.2 Micro-navigation and beamforming

Bloomsbury DSP has also implemented micronavigation and beamforming algorithms for SAS processing using FPGA hardware. Details of this are left out of this paper due to space constraints. For these algorithms, a gain in performance of around ten times over a Pentium M implementation was also been seen.

It was still necessary to implement some parts of the algorithms using software running on a CPU. Such functions include interpolation for DPCA and system control. The use of a computer also facilitates easy debugging and flexibility.

4 CONCLUSIONS

In this paper we have shown a need for real time SAS equipped AUVs for use in a number of commercial and defence applications. Not only does SAS offer high-resolution images, it provides very accurate positioning information, which can aid the navigation of an AUV. To facilitate real-time SAS processing on an AUV, a high performance, low power and compact computing solution is required.

FPGA and SBC/GPP technology have been compared for performing different SAS related algorithms. We have shown that for systematic and computationally intensive algorithms, an FPGA can provide more than an order of magnitude better performance than a modern low power CPU. FPGAs can also be cheaper, physically smaller and consume less power. For integration into an AUV, where space and power are at a premium, this can be more graphically translated into: an FPGA can replace 10 single board computers for performing certain sonar algorithms.

Although in some situations FPGAs can provide a significant advantage, they are not suited to algorithms that are non-systematic. For this reason, the most suitable architectures for processing real time SAS data are based on a combination of FPGA and CPU technology. A number of companies are now producing off-the-shelf platforms with this architecture.

5 REFERENCES

1. National Oceanography Centre, Southampton, UK. See <http://www.noc.soton.ac.uk/>

2. Bluefin Robotics, Cambridge, MA, USA. See: <http://www.bluefinrobotics.com/>
3. Kongsberg Gruppen, Norway. See: <http://www.kongsberg.com/>
4. S. Banks, 'Studies in High Resolution Synthetic Aperture Sonar', PhD Thesis, University College London, September 2001.
5. J. von Neumann, 'First Draft of a Report on the EDVAC', Contract No. W-670-ORD-4926 Between the United States Army Ordnance Department and the University of Pennsylvania (June 30, 1945).
6. Xilinx. For a history, see : <http://www.xilinx.com/company/xilinxstory/history.htm> *
7. Altera. For a history, see : http://www.altera.com/corporate/about_us/innovation/abt-history.html *
8. M. Vespe, C. J. Baker and H. D. Griffiths, 'Multi-Perspective Target Classification', Radar Conference, 2005 IEEE International, 9-12 May 2005 pp877-882.
9. A. Bellettini and M. A. Pinto. 'Theoretical accuracy of synthetic aperture sonar microneavigation using displaced phase centre antenna'. IEEE Journal of Oceanic Engineering, Vol. 27, Issue 4 Oct 2002 pp780-789.
10. A. V. Oppenheim and R. W. Schaffer, 'Discrete-time signal processing', Second Edition, Prentice Hall Signal Processing Series, ISBN 0-13-754920-2.
11. W. H. Press, S. A. Teukolsky, W. T. Vetterling and B. P. Flannery, 'Numerical Recipes in C The Art of Scientific Computing', second edition, Cambridge University Press, ISBN 0-521-43108-5
12. Octave. See: <http://www.gnu.org/software/octave/> *
13. MATLAB. See: <http://www.mathworks.com/> *
14. Intel Integrated Performance Primitives. See: <http://www.intel.com/cd/software/products/asmo-na/eng/perflib/ipp/index.htm> *
15. Cray super computers. See: <http://www.cray.com/> *
16. Spectrum Signal Processing, Inc. See: <http://www.spectrumsignal.com/> *
17. B. Fletcher, 'UUV master plan: a vision for navy UUV development', Oceans 2000, Volume 1, 11-14 Sept 2000, pp65-71.

*Web addresses correct at time of writing.