HIGH PERFOMANCE DIGITAL SIGNAL PROCESSING

Tom Curtis[1], Michael Curtis[2]

Curtis Technology (UK) Ltd, Weymouth, Dorset, DT4 7BS

[1]tomc@curtistech.co.uk
[2] michael.curtis@curtistech.co.uk

1. INTRODUCTION

Recent improvements in the speed and complexity of commercially available digital signal processing (DSP) devices have allowed digital techniques to be extended to a number of new areas. Now many applications in RF and communications are within the bandwidth capability of current generation DSP systems. This paper demonstrates a system developed recently for such applications. It uses standard PC bus architectures and interface chipsets, together with commercially available analogue-to-digital converters (ADCs), digital-to-analogue converters (DACs), and field programmable gate arrays (FPGAs), to realise wide bandwidth, high performance systems at relatively low cost.

The system outlined provides multiple independent communications processing channels, each with in excess of 100 MSPS real time bandwidth. Analogue interfaces are provided via ADCs and DACs and the module supports frequency domain processing via real time forward, inverse fast fourier transforms (FFTs) and various matrix manipulation algorithms. An industry standard peripheral component interface (PCI) is also implemented on the system, so that raw or processed time-series and/or frequency domain data can be examined and displayed as required, using a standard personal computers (PCs). This PC interfacing also provides the necessary hooks for simple adaptive processing on the high speed data stream.

2. TECHNOLOGY BACKGROUND

Until relatively recently the choice of architectures available to system designers for high performance DSP was limited. Mainstream integrated circuit manufacturers such as Analog Devices [1], Texas Instruments [2], Motorola [3], etc, each produce families of programmable DSP parts for the commercial market. In the main, these devices are derivations of architectures that have been around for some time, with incremental improvements in performance being achieved by successive geometry shrinks to improve clock speeds and reduce dynamic power consumption. The mainstream devices are general purpose, in the sense that their architectures were developed to cope with a number of different potential markets, with the customisation for particular applications being achieved by software programming.

Manufacturer	Part No	Clock Rate	1k complex FFT speed
Analog Decices	**ADSP-BF53x**	**600 MHz**	**16.2 microseconds**
Motorola	**Star Core SC140**	**300 MHz**	**15.8 microseconds**
Texas Instruments	**TMS320C64x**	**720 MHz**	**8.34 microseconds**

Table 1(a) – Some Available Main-Stream Digital Signal Processors

Proceedings of the Institute of Acoustics

A further approach, by what could be described as the more "niche" DSP players in the IC manufacturing area, such as Sharp [4], DSP Architectures [5], etc, has been to produce more specialised DSP engine parts, with architectures more closely tuned to particular applications. Whilst most of the mainstream programmable DSP parts provide mainly scalar arithmetic functions, these DSP engines were designed to handle complex vector arithmetic directly. This gives them some advantage in terms of lower clock rates and system software requirements but at the expense of a more limited application area.

Manufacturer	Part No	Clock Rate	1k complex FFT speed
Sharp Microelectronics	**LH91V24**	**80 MHz**	**38.4 microseconds**
DSP Architectures	**DSP24**	**80 MHz**	**27.5 microseconds**

Table 1(b) – Some DSP "Processing Engines"

These two "flavours" of DSP device architecture offer the system designer sufficient flexibility for most applications but there are still some systems that benefit from a more customised processor design. The cost of custom design for small geometry systems-on-chip (SoC) has always been a problem but is cost effective in high value added or large user volume applications.

However, the improvements in performance of SoCs that has been achieved by transistor geometry scaling in recent years are starting to hit some fundamental limits. Recent geometry scalings from 130 nm to 90 nm have not produced the increase in operating speed that previous scaling steps have realised. Excessive transistor leakage at the smaller feature sizes on some processes has resulted in die where the resultant dc dissipation has caused major thermal problems, which sets the limit to device operating frequencies rather than the more normal dynamic power dissipation limits that dominated earlier larger geometry processes [6].

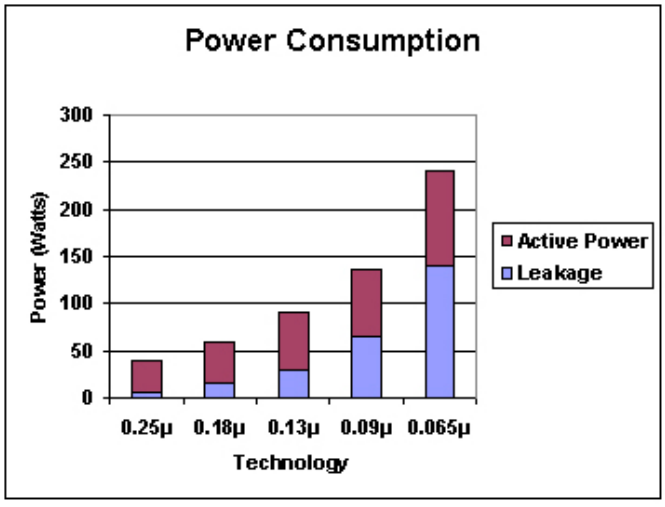

Figure 1 – Power Dissipation vs. Process Geometry – from [6]

In order to circumvent these process problems, exotic technology tweaks like using strained silicon and high-k dielectric gate materials are being incorporated into the process flow by the major foundries. Consequently, it is becoming increasingly difficult for the smaller systems houses to get access to these technologies and develop custom DSP parts on "leading edge" processes:

Proceedings of the Institute of Acoustics

the up-front costs for computer aided design (CAD) tools for that level of design and the foundry charges are just too prohibitive.

In the past, one way around this problem with foundry access was to develop custom hardware at printed circuit board (PCB) level using "bit-slice" parts [7], for example using high-speed static memory and various multiplier or multiplier/accumulator devices. However, with the increasing complexity available on chip, these low complexity DSP "building block" parts have fallen off the technology "trailing edge". Consequently it was difficult for a time to build efficient custom architecture DSP systems for niche applications, as the basic components were just not available.

However, recent advances in FPGA technology have started to fill that gap. FPGAs with random access memory (RAM) based configurable logic blocks (CLBs) have been around for some time, from the major players such as Xilinx [8] and Altera [9]. The combination of programmable logic and the ability to use the CLB RAM blocks as distributed memory made these devices immediately attractive in many DSP applications.

More recently the manufacturers have started to add features to their basic FPGA architectures in order to differentiate their products in the market place. Xilinx, for example, initially added blocks of larger fast dual port RAM to their devices, making the chips useful in areas such as corner turning for DSP. Arrays of embedded dedicated multipliers were soon added to this block memory feature and current generation FPGA devices are available with hundreds of 4kbyte memory blocks and 18x18 bit multipliers, together with upwards of 60,000 programmable logic slices. Table 2 outlines the DSP related resources available on current generation FPGA families.

Manufacturer	FPGA Family	Device	Block RAM	Multipliers
Altera	**Stratix II**	**EP2S60**	**128k bytes**	**144**
Altera	**Stratix II**	**EP2S130**	**300k bytes**	**252**
Xilinx	**Virtex II**	**XC2V3000**	**216k bytes**	**96**
Xilinx	**Virtex II**	**XC2V8000**	**378k bytes**	**168**
Xilinx	**Virtex IIP**	**XC2VP125**	**1251k bytes**	**556**

Table 2 – Typical DSP Resources on FPGAs

Using these programmable parts, the system designer has much more freedom in architecture design and the vast amount of DSP-related resources available on chip allows processing engines with massive throughput to be considered. DSP-enabled FPGAs, together with current generation ADCs [10] and DACs [11], provide the resources to implement processing hardware for very high bandwidth systems. As an example of the capability of this type of device, the following section outlines the development of a multiple channel parallel pipelined FFT for professional RF applications that integrates several channels of frequency domain processing, with bandwidths in excess of 100 MSPS.

3. ARCHITECTURE CONSIDERATIONS

The availability of massive resources, in terms of dedicated multipliers and memory, provides considerable flexibility for DSP design. However, to use these devices efficiently, the designer needs detailed knowledge of the algorithms to be implemented and how to map them onto the programmable hardware. In general terms, this type of programmable hardware design requires a significantly different “skill-set” than that required for systems using main-stream DSP parts. To reduce the "learning curve" associated with programmable hardware design, a number of

Proceedings of the Institute of Acoustics

companies have started to offer "black box" designs for various DSP functions [12] that can be readily integrated into the FPGA design flow. However, this “encapsulated IP” approach to system development can be restrictive (and expensive!!), so the approach taken here was to design the DSP engine from the ground up, building on previous DSP architectures and designs [13].

Only the FFT/IFFT engine design will be considered in any detail, although the performance of the overall unit will be outlined in later sections. The main aim will be to illustrate the basic architecture development and the level of performance that can be realised, rather than providing a detailed formal description of FPGA hardware design.

3.1 Architecture Background

In earlier hardware systems, using for example LSI multipliers and bit slice devices, one of the major constraints on processor design was the need to reduce the complexity of control logic in the system, as typically this was implemented using MSI TTL. Considerable efforts were made to minimise the overall system cost, and balance the control complexity to arithmetic complexity ratio, often by heavily massaging the processing algorithms to match the chosen implementation technology [14,15]. Often, various number theory and matrix manipulation techniques were used to achieve this balance.

Similar massaging is needed to realise efficient performance on FPGA based systems. However, the constraints are now very different. Control complexity is no longer an issue: there are sufficient CLB resources for complex control designs. The main constraints are now in developing algorithms that map readily onto the interconnection architecture provided on the device. Interconnection routing tends to be faster and more efficient between neighbouring components on FPGA chips, leading naturally to the choice of heavily pipelined architectures for fast, densely packed systems.

A number of previous designs [4,5,13] have used higher radix FFT butterflies but with pipelining used only in the processor core itself, i.e. at the time they were designed, the technology could not support enough resources on chip, in terms of memory and multipliers, to fully “unwrap” the FFT algorithm.

For example, the Complex Vector Processor outlined in Reference 13, used a three-port pipelined butterfly engine, with three real multipliers and four arithmetic units, to implement the Radix-4 based system. Clocking at 25 MHz, this design achieved 1k-point complex FFT times of around 205 microseconds. The Sharp LH9124 used a four-port pipelined engine, with six multipliers and eleven arithmetic units to implement a pseudo radix-16 system, with 1k-point transform times of around 38 microseconds when run at a clock rate of 80 MHz.

Similarly, bit slice designs have been reported [16] that used fully pipelined memory architectures, but with simpler radix-2 FFT butterflies, to minimise the overall processor resources.

3.2 FPGA MAPPINGS

3.2.1 Dual Port RAM Based Systems

Current generation FPGAs have enough routing and resources to consider fully pipelined, high-radix FFT designs. The combination of dual port memory and hardware multipliers on the Xilinx FPGAs, for example, leads naturally to radix-2 pipeline designs, such as that shown schematically in Figure 3. Typically, for 1k-point complex FFT, this design can be clocked at around 150MHz, giving a transform time of about 3.5 microseconds.

Proceedings of the Institute of Acoustics

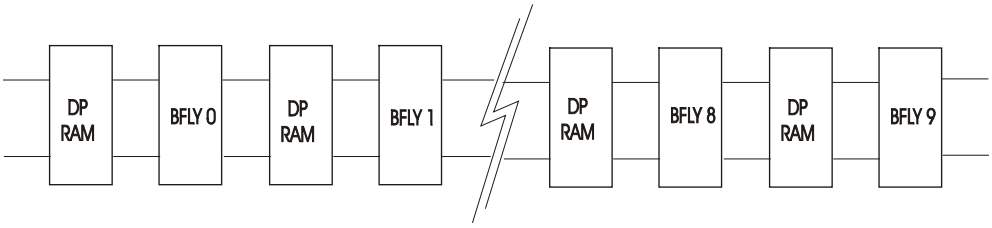

Figure 3 – Pipelined Radix-2 FFT Schematic

Even this simplistic design approach is faster than the best currently available main stream DSP parts. However significant gains in performance are available by massaging the engine design to fit the FPGA resources - there is around a factor of two difference in speed between the dual-port block memory access time and the fall through time for the multipliers on the Xilinx devices. Faster RAM access time can be traded to provide pseudo four-port RAM, as shown in Figure 4, giving four port access speeds matched to the multiplier fall through time.

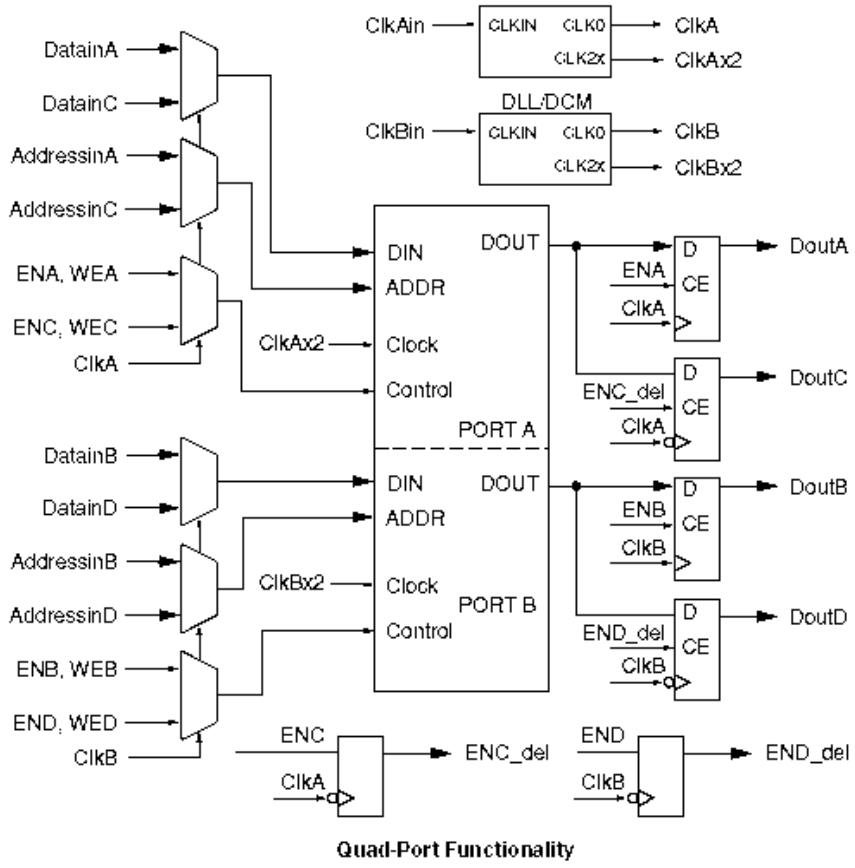

Figure 4 – Pseudo 4-port RAM Schematic

So, pipelined Radix-4 systems, such as shown in Figure 5, can be clocked at similar speeds to the Radix-2 design in Figure 3. These achieve a 1k-point complex FFT speed of around 1.7 microseconds. This design uses around 48 multipliers and 30 block RAMs and represents around 50% of the multiplier and 30% of the BRAM available, on say a Xilinx XC2V3000.

Proceedings of the Institute of Acoustics

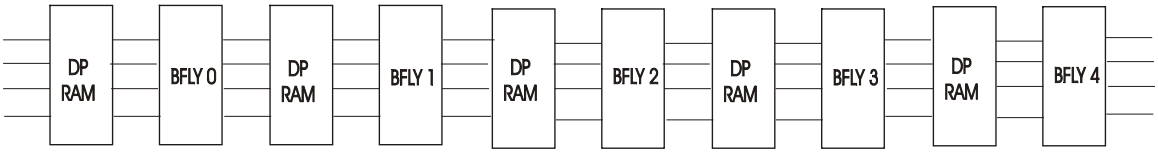

Figure 5 – Pipelined Radix-4 1k FFT Schematic

This resource requirement assumes a 4 multiply/2 add implementation for each complex multiply: using a 3 multiply/5 add design (See Figure 6) reduces the multiplier requirement to around 36 multipliers, with no speed penalty. The resources used to implement this basic radix-4 design then represent about 30% of the total available on the XC2V3000.

Four multiply/two add implementation:-

$$(a+jb)(x+jy) = ax-by + j(bx+ay) \quad \ldots\ldots (1)$$

Three multiply/five add implementation:-

Writing

$$X1 = (a+b)x$$

$$X2 = (x+y)b$$

$$X3 = (x-y)a$$

Gives

$$(a+jb)(x+jy) = (X1-X2) + j(X1-X3) \quad \ldots\ldots (2)$$

Figure 6 – 4 Multiply/2 Add vs 3 Multiply/5 Add Complex Multiplier

The hardware requirement can be reduced further using a pseudo radix-16 butterfly and a mixed radix-16/radix-4 architecture. The pseudo radix-16 butterfly uses the radix-4 butterfly, with the some additional real multipliers. This reduces the hardware needed for a 1k-point complex transform to 26 multipliers and 18 BRAMS with no speed penalty.

Consequently, it is possible to pack a number of separate FFT units, working in parallel on different data streams, with their control, memory addresses and coefficients running in lock-step from a common control system.

3.2.2 Shift Register Based Systems

Using larger radix transforms becomes difficult using multi-port memory directly. However, using a combination of fixed length delays and multiplexers [17], as shown in Figure 7 allows high radix transforms to be implemented directly.

Proceedings of the Institute of Acoustics

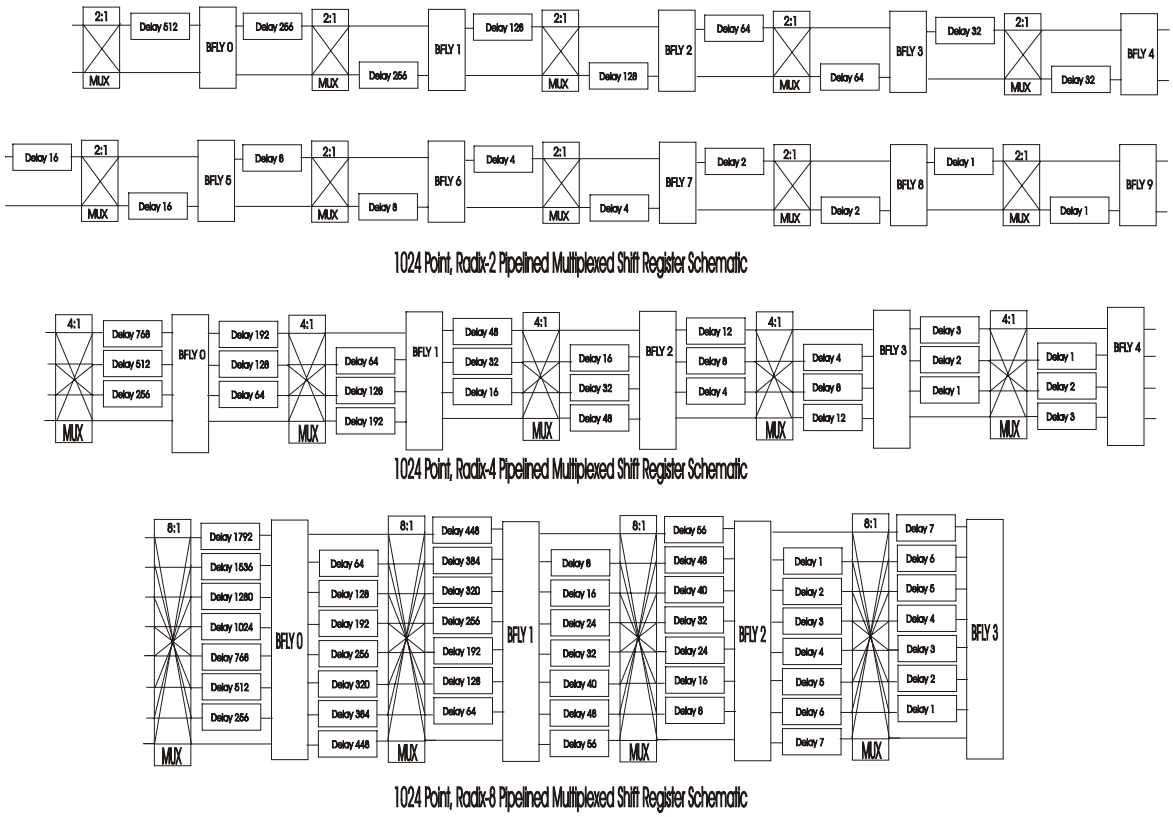

Figure 7 – Pipelined Register/MUX based High Radix FFT Schematics

This approach uses a reasonable mixture of CLK, multiplier and BRAM resources and various mapping tricks can be used to implement the higher radix transform cores efficiently, using for example small length low complexity Winograd transforms [18] .

Benchmarks for various algorithm mappings are given in Table 3

Radix Size	Transform Size	Speed (microseconds)
2	1024	3.41
4	1024	1.71
8	512	0.43
16	256	0.11
32	1024	0.21

Table 3 – Higher Radix Speed Benchmarks

4. PRACTICAL SYSTEMS

A number of practical systems designs based on the architectures outlined above have been built. In all cases, these have use arithmetic word widths of at least 24 bits for the data paths through the system, with 18 bit wide coefficient data. The 24 bit x 18 bit multipliers for this were realised by using the embedded 18 x 18 bit multipliers for the MS product and using CLB resources to

Proceedings of the Institute of Acoustics

generate the LS product and combine the results. This approach was needed to achieve the dynamic range required for the application.

Figure 8 shows a photograph of one such practical system module. It uses two 14-bit, 105 MSPS ADCs and a dual 14-bit, 125 MSPS DAC for analogue data interfacing. Analogue input and outputs are transformer coupled for isolation. A transformer-coupled trigger input is also provided for use in applications where synchronous snap-shot data is required. The module uses a Xilinx XC2v3000 FPGA and three blocks of 64k x 64 fast synchronous static RAM. The FPGA can be configured either via the module interface (using PCI bus on mother board), by a J-tag test interface or using a dedicated flash memory.

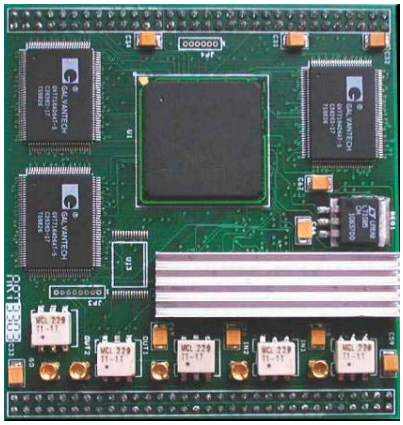

Figure 8 – FPGA based FFT Module

The module normally sits on a cPCI mother board (as shown in Figure 9) to interface it to the rest of the DSP system. Commercial PCI motherboards are also used – see Figure 10. This figure also shows the format of the demonstration software used to exercise the system.

Figure 9 – CPCI Processor Card

Whilst this module was developed for an RF processing application, it has also been used as the processing engine in high throughput sonar surveillance systems, as outlined in Reference [19].

Proceedings of the Institute of Acoustics

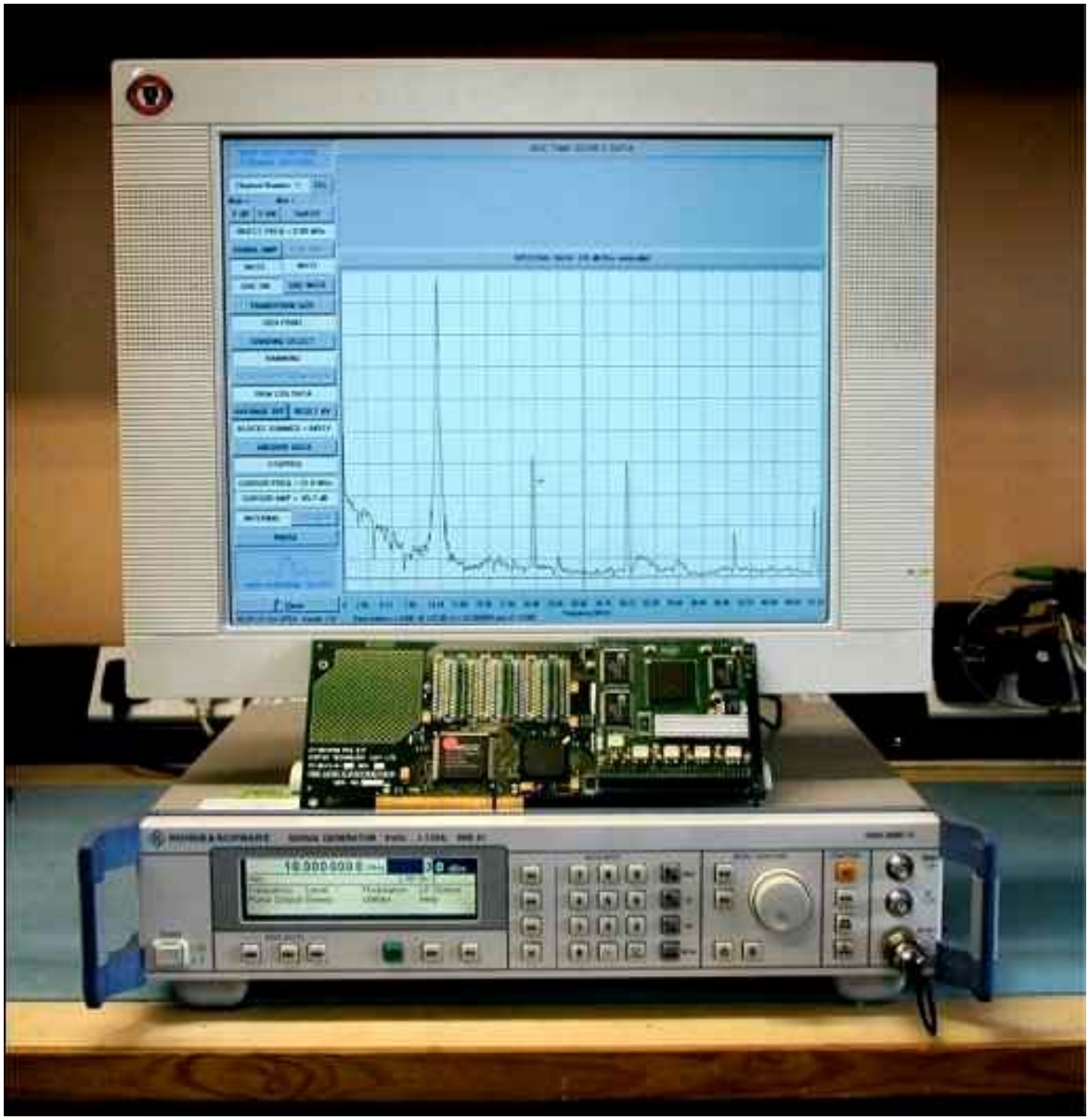

Figure 10 – FFT Module Demonstration Set-up

REFERENCES

[1] “ADSP-BF531/ADSP-BF532/ADSP-BF533: Blackfin® Embedded Processor Data Sheet”, Analog Devices, Norwood, MA.

[2] “TMS320C6414, TMS320C6415, TMS320C6416 Fixed-Point Digital Signal Processors”, Texas Instuments, Houston, Texas.

[3] “MSC8102 Quad Core 16-Bit Digital Signal Processor Technical Data”, Motorola.

[4] “LH9124 Digital Signal Processor Users Guide”, Sharp Electronics Corporation, Camas, WA, USA, 1992.

[5] “DSP24 – High Performance DSP Chip”, DSP Architectures, Vancouver, VA.

[6] “Low Power SoC Design”, EDA Weekly Review for May 24, 2004.

[7] "Control Ordered Sonar Hardware – COSH: A Distributed Processor Network for Acoustic Signal Processing", T E Curtis, A G Constantinides and J T Wickenden, Part F, Proc IEE, 1984.

[8] “Virtex-II Platform FPGA Handbook”, Xilinx, San Jose, CA.

[9] “Stratix II Device Family Data Sheet”, Altera Corperation, San Jose, CA.

[10] “AD6645: 14-bit 80/105 MSPS A/D Converter”, Data Sheet, Analog Devices, Norwood, MA.

[11] “AD9767: 14-bit, 125 MSPS Dual TxDAC D/A Converer”, Data Sheet, Analog Devices, Norwood, MA.

[12] See for example – “Xilinx IP Center” at www.xilinx.com

[13] “ A Fast 32-bit Complex Vector Processing Engine”, A J Kerr and T E Curtis, IOA Conference on Sonar Signal Processing, Loughborough, 1989.

[14] “The Interaction Algorithm and Practical Fourier Series”, I J Good, J Roy. Statist. Soc., Ser. B, 1953, 20 and Addendum, 22.

[15] “Hardware-based Fourier Transforms: Algorithms and Architectures”, IEE Proc, Part F, Communications, Radar and Signal Processing, 130.

[16] “Wafer Scale Integration for Improved Signal Processor Efficiency”, E E Schwartzlander, Proc of Digital Signal Processing-91, Florence, 1991.

[17] see for example – “Theory and Application of Digital Signal Processing”, L R Rabiner and B Gold, Prentice-Hall, NJ.

[18] “On Computing the Discrete Fourier Transform”, S Winograd, Math Comput, 32, 1978.

[19] “Compact Sonar Surveillance Processing Systems”, T E Curtis and M J Curtis, IOA Conference on Sonar Signal Processing, Loughborough, 2004.