Software Radio: A Review of Design Considerations and Digital Hardware Choices

Parul Gupta[1], Pradipta Dey[2], Shivkumar Kalyanaraman[1], Wang Qing[3], Chen Jianwen[3], Lin YongHua[3]

[1]IBM India Research Lab, Bangalore Email: fparulgupta, shivkumar-kg@in.ibm.com

[2]IBM India Research Lab, Delhi Email: pradipta.de@in.ibm.com

[3]IBM China Research Lab, Beijing Email: fwangqing, jianwenc, linyhg@cn.ibm.com

Abstract: This paper reviews the requirements for Software Defined Radio (SDR) systems for high-speed wireless applications and compares how well the different technology choices available- from ASICs, FPGAs to digital signal processors (DSPs) and general purpose processors (GPPs) - meet them.

Key words: ASIC; FPGA; DSP; GPP; processor; software radio; SDR; wireless networks

I. INTRODUCTION

The case for Software Defined Radio systems is universally accepted now, whether it be to keep pace with rapidly evolving wireless standards, support for multiple applications or as a key enabler for cognitive radios. First proposed by Joseph Mitola in 1991[1], the SDR concept has been around for some time, but with recent advancements in electronics, SDR systems are becoming increasingly feasible. The focus now is shifting towards building maximally flexible and efficient systems. This paper presents a review of the design requirements for software radios for high-speed wireless application and a comparison of how well different technologies meet them.

II. DIGITAL HARDWARE CHOICES

Broadly, the spectrum of digital hardware choices available for implementing a communication system range from the very specialized, inflexible but super-efficient Application Specific Integrated Circuits (ASICs) to highly programmable General Purpose Processors (GPPs) which sacrifice some cost, area and power efficiency. Between the two lie DSPs, FPGAs and many hybrid systems. Digital Signal Processors (DSPs) are microprocessors with architecture, instructions and features suited specifically for signal processing applications. Examples of GPPs are ARM, PowerPC etc and of DSPs are TI C64x series, Freescale MSC8144 etc. Field Programmable Gate Arrays (FPGAs) are semiconductor devices consisting of low-level logic blocks and interconnects which can be programmed for a desired functionality. FPGAs allow very high degree of customization and parallelization and can be used for a variety of applications, including signal processing and control. Over time, boundaries between these different families have blurred with hybrid devices coming in. Now, one can find GPPs with DSP features, soft-microprocessors implemented in programmable logic as well as multicore and multithreaded processors which combine the parallelism and computation capacity of ASICs/FPGAs with the programmability of processors. The suitability of any hardware technology over another needs to be measured on many criteria and depends heavily on the application. The sections below vpresent such a discussion for software radio systems for highrate wireless communication systems.

III. SDR SYSTEM REQUIREMENTS

The first natural requirement that follows from the central concept of software radio is flexibility, and configurability. The common trend between emerging wireless technologies is higher data-rates, better quality of service, better security and often features like high adaptability and support for mobility.These requirements translate into more complex algorithms and hence the demand for higher computation capacity. Also important for communication systems is real-time operation. There are other desirable aspects like small area, cost and power consumption, but they can be less or more important depending on the target application, e.g. less when you are building a base station and more for a consumer device. We shall discuss each of these requirements and how well different digital hardware choices meet them in detail in the sections that follow.

A. Computation capacity

The propagation effects of wireless medium together with the need for ever-higher data rates and spectral efficiency place many demands on signal processing inwireless systems. Because of random fading and delays in multipath channels, wireless receivers need sophisticated synchronization and channel-tracking algorithms. Further, to correct any errors introduced by the wireless channel, forward error correction (FEC) is needed. The Viterbi decoder for convolutional codes has traditionally been the most compute-intensive module in wireless receivers. Turbo codes used in Wimax have concatenated convolutional codes with even more complex decoders! For supporting higher throughputs and more users,multiplexing techniques like CDMA, OFDM/OFDMA and spatial multiplexing are being used which need complex signal processing. The high data rate required in those new standardsis also adding the pressure on computation capacity, e.g. up to 75 Mbps in Wimax and 500 Mbps in 802.11n. To show this trend, a coarse estimate of the computation requirement for the physical layer of some wireless standards are summarized in Table I [1].

Table 1 Required phy layer computation capability in defferent standards W-CDMA 802.11a

	W-CDMA	802.11a	WiMAX
Throughput	2Mbps	24Mbps	20Mbps
Required resources	5.48G cycles/s	7.82G cycles/s	16G cycles/s

Even though ASICs have the best power-efficiency and computation performance, their inflexibility and long, complex development cycles make them unsuitable for SDR. However, some sections where performance requirements can’t yet be supported on other platforms, esp. the RF front end, continue to be in ASICs. The computation capacity of processors has not scaled up to keep pace with the demand because increasing operating frequencies steadily gets difficult and also consumes much power. Such computation needs to be split on processor banks consisting of many processors. A preferred alternative is processors with multiple cores and multiple threads with each core running at a lower frequency. Multi-core processors have an advantage over processor banks as they give more processing capacity for less area and power. Table II shows a comparison of a bank of 16 MSC8103 StarCore DSPs and 4 quad-core MSC8122 StarCore DSP, both operating at 300 MHz, highlighting this advantage[2].

For speeding up computation, an alternative to the above homogenous multi-core processors are systems with multiple specialized processing units. E.g., QuickSilver Technology's Adaptive Computing Machine[3] has three different kind of processing units - each specialized for arithmetic operations, bit-manipulation and control logic. A more common example is DSPs with hardware accelerators or dedicated co-processors for key modules like Viterbi, FFT. Heterogeneous systems provide a trade-off between overall system flexibility and individual module computational efficiency. While a homogeneous system can distribute the system workload among processing units, a heterogeneous system must provide enough units for the worst case workloads of each type of PE. Therefore, heterogeneous systems are more-likely to underutilize their hardware.

Table 2 Comparison of
single core vs multi-core. source [2]

	Single-Core	Multi-core
Package Size	272mm x 272mm	80mm x 80mm
Power Consumed	13.8 W	4.6 W [a]
Number of Pins	5312	1724
Performance	19.2 GMACs	1 9.2 GMACs

[a]Power measurement for multi-core DSP operating at 400 MHz vs 300 MHz for single core case

For the same number of cores, operating at the same frequency, instruction set architecture can help speed up computation a lot. Some operations occur very frequently in signal processing algorithms, e.g. multiply accumulate (MAC) in correlation, filtering, FFT etc or performing repetitive computations in a loop. Each computation needs a combination of simpler instructions like fetching of instructions and operands from memory, performing a multiplication followed by an addition, incrementing loop counter, calculating address for the next computation etc, which could take several cycles on a simple RISC processor. Specialized architecture which pipelines hardware components to perform multiple instructions parallely (instruction-level parallelism) or which perform an operation on vector data simultaneously (data-level parallelism) can speed up computation. For example, the Cell Broadband Engine (Cell BE) has one Power processor element (PPE) for control and 8 synergistic processing elements (SPEs) for computation intensive processing running at 3.2 GHz. Each SPE has its own Single Instruction Multiple Data (SIMD) vector execution unit, supporting a 128-bit vector. Overall the Cell BE can deliver more than 100 GMACs per second. In our implementation of the WiMAX PHY layer optimized for Cell BE platform[4,5], about 30% of the operations are implemented with SIMD instruction. After accounting for overheads and different operand-widths used in the SIMD code, e.g. 8 bits and 32 bits, we could get more than 3 times speedup by using SIMD.

Other architectures like superscalar and Very Long Instruction Word (VLIW) provide instruction-level parallelism by allowing different hardware units of the processor like memory, ALU etc to be used in parallel.

So far, we have discussed various methods which can increase computation capacity of processor systems. Now the task of interest is to compare the computation capacity of different platforms for implementation of a wireless standard. Typically, wireless standards give specifications like coding, modulation, spectral mask, sensitivity requirements etc but leave the exact implementation to the vendor. Actual computation requirements depend heavily on the algorithms as well as the software design. In general, one can use more complexsig-

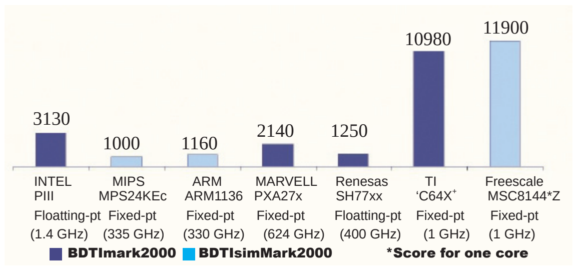

Fig. 1 A comparison of the computation speeds of different processors based on BDTI benchmarks[7]. Higher is faster.Source[8]

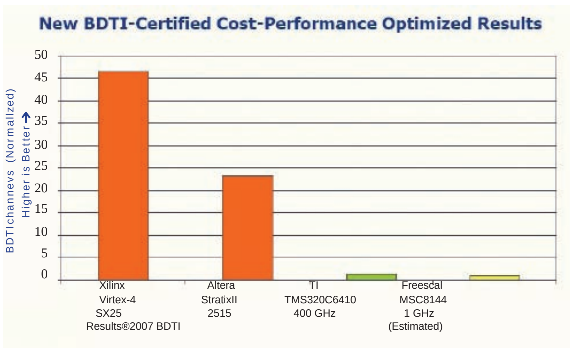

Fig. 2 A comparison of the cost-performance of FP GAs vs DSPs based on the BDTI Communi-cations benchmark for OFDM[10]. Source[9]

nal processing algorithms for achieving better bit error performance. Even for a chosen algorithm, an intelligently optimized implementation can reduce the computation requirement by as much as 10 times[4]. Hence, the comparison of computation rate of different platforms is fair only when done for a common implementation. BDTI Inc. provides one such reference benchmark, BDTImark2000[2], based on common signal processing modules like filtering, FFT, Viterbi decoder etc[7]. Figure 1 provides a comparison of performance of different processors on the BDTI benchmark.

Note that the above speed comparison is meaningful for processors in which computations are predominantly serial. In FPGAs, the computation resources can be used in a serial, parallel or hybrid implementation of the system. For FPGAs, the computation capacity is measured in terms of the number of logic cells. In addition to the algorithms and software architecture, the number of cells needed by a system also depend on the bit-widths chosen for each computation. Processors are fixed width, but FPGAs provide the extra degree of optimization by allowing customization of bit-width for each operation, with significant saving in complexity, area and power. Figure 2 shows a comparison of the cost-performance of FPGAs vs DSPs based on the BDTI Communications benchmark for OFDM[10].

B. Computation accuracy

The error performance of communication systems depends heavily on the precision used for physical layer signal processing. Fixed-point processors allow a fixed range and precision but the range of values along the processing chain can vary a lot. The designer has to pay careful attention to scaling of values to maximize accuracy and account for overflow. The precision needed for computations in different modules can also be very different - e.g. threshold detection or some filter coefficients might need only 4 bits of precision but the FFT-values might need 16 bits or higher. Since processors have fixed width (e.g. 8/16/24 bit for DSPs or 32/64 bit for GPPs), they either provide insufficient precision or are too wasteful. In contrast, FPGAs give complete flexibility to the systems designer for right-sizing functional components, making them very efficient in terms of area and power. But this comes at the cost of significant complexity in system design and verification. For floating point processors, computations are high-precision and attention to numeric effects is far less required.

C. Memory architecture

The intensive signal processing for high-rate wireless standards naturally needs a lot of data movement, e.g. fetching instructions and operands from memory and passing results of one module as inputs to the next. Movement of large words (because of high-precision computation) very often (because of high-throughput) requires high bandwidth memory access. In the memory architecture, both the number and width of accesses allowed per cycle is important. DSPs have Harvard architecture, i.e. memory is segmented into program and data memory, so that operands and instructions can be fetched simultaneously in a single cycle. Earlier GPPs had Von Neumann architecture, i.e. a single program+data memory but now even high-end GPPs have Harvard architecture. Further, memory access in signal processing blocks often follow wellknown patterns like sequential, circular (e.g. filtering) or bitreversed (e.g. in FFT). DSPs provide high-memory access through address generation coprocessors and special instructions which leverage known addressing patterns.

In addition to memory bandwidth, the amount of memory is also very important. Processors have hierarchical memory with L1/L2/L3 caches and possibly off-chip memory with different access times for each of these components. For modules with streaming computations (e.g. filters), small FIFO queues suffice but for modules which operate on blocks (e.g. interleaver) large memory banks are needed. The tradeoff in the choice of platform is that if you don't have sufficient onchip memory, you will spend more cycles fetching data fromexternal memory.

The memory requirements in FPGAs are smaller as they need only data memory, no program memory. The program functionality is burnt onto the FPGAs - i.e. fixed interconnects between logic cells are created at startup - eliminating the need to store the program code as well as all the processing needed for instruction fetches from memory.

D. Communication capacity at I/O interfaces

For Software Radio system, the data communication between the Baseband processing board and the RF module is characterized by high throughput. The system designer needs to ensure that the I/O interfaces of the chosen platform can support such required throughputs. For example, in the WiMAX PHY system with 20Mbps throughput for both uplink and downlink, if an 8 bit DAC is used, the output of downlink is about 658Mbps. For 3 sectors, the overall downlink throughput is about 1.975Gbps. On the CELL blade server QS21, there are two independent duplex Gbps Ethernet adapters, which can support 3 sectors uplink and downlink. Moreover, a mass of resources will be consumed for receiving, parsing and sending packets which should be accounted in the processing capacity.

E. Real-time operation

For communications systems, data streams coming in at high rates need to be processed in real-time and under strict latency constraints. This entails that bounds on computation as well as memory access times be deterministic. In ASICs and FPGAs, computations are triggered by a uniform hardware clock and throughput is deterministic. GPPs have dynamic behavior because of OS-controlled features like branch prediction and thread scheduling which can lead to throughput jitter. To solve this problem, de-jitter buffers on the input and output interfaces and a uniform clock module is used to controlthe throughput.

F. Development effort and flexibility

Development on processors is usually fast as they have good tool support (compilers, IDE etc) and can be programmed in high-level languages. Optimization in assembly is needed only for the most compute-intensive modules and often hardware accelerators or optimized libraries for these modules are supplied by the processor vendors. Third party libraries and programming skills are also easier to find. For signal processing components of communication systems, the above holds especially true in the case of DSPs. With the GPP and DSP families mixing, the availability of tools, skills and libraries for signal processing on GPPs is bound to improve.

A caveat - the computation needs of higher-rate standards like 4G can't be met by any single processor today. Multicore/ mutithreaded processors and even multiple processor banks are needed. It is easier to split the processing of streams of data where the interaction between individual streams is limited, e.g. parallel processing for multiple antennas or independent computational modules like filtering and FFT. If the computation capacity of individual processing elements necessitates further partitioning of algorithms, synchronization and verification can be a big challenge.

FPGAs need to be programmed in Hardware Definition Language (HDL) at a much lower level of logic blocks. Tools like SystemC, AccelDSP which generate Verilog/VHDL from C or MATLAB do not give very optimized code and are used for proof-of-concept prototypes rather than end products. After coding, an extra step in FPGA development is synthesis and timing simulation. Computations in FPGAs are driven at precise clock edges rather than at the end of previous computation. Hence if any operation takes longer than designed for (i.e.timing is not met), data integrity can be compromised. Meeting timing is more difficult at higher operating frequencies and can need significant design changes, making the development cycle longer as well as more complex. Development of the same system can take up to 4x or longer on an FPGA than on a processor.

G. Other factors

There are many other factors like smaller cost, size, power efficiency and system integration which are desirable, but can be more or less important depending on the application. All of the above would be very critical for a consumer handheld device, size and power less so for a fixed, powered device (e.g. a desktop or a game station) and all of them less so for network appliances.

The cost aspect has many components, many of which are linked to other criteria: 1. Non recurring expenditure (NRE) for the initial setup - this involves the tool as well as human resource costs. 2. The recurring per-unit manufacturing costs which depend on the area, processing capacity etc. 3. Poor power efficiency leads to indirect expenditure in the form of bigger power supplies or on-chip cooling systems. Also, device failures escalated due to excessive heating add to indirect costs.

Typically, the power consumption and device costs of FPGAs end up lower than DSPs because they can be highly customized for the target application and trim all the overheads.This can be seen in the significantly higher BDTIChannel/$ of FPGAs seen in Figure 2. Note that it does not take into account the NRE, especially the cost due to longer and more complex design and verification cycles which can be higher for FPGAs and needs to be amortized over volumes. There is a “break even” volume at which the NRE and unit costs balance, determining the volumes at which ASIC, FPGA or DSP makes sense. According to an analysis done in [12], the total costs can be comparable for DSPs and FPGAs for some applications.

Often, hybrid systems using a combination of ASIC, FPGA, and processors have been used to

Table 3 Comparison of different platforms for a SDR implementation

	Freescale MSC8144 (4-core DSP)	Power6 (GPP)	Cell BE	Xilinx Virtex4-SX25 FPGA [15]
Operating Frequency	1 GHz	4.7 GHz	3.2 GHz	500 MHz
Computation Rate[a]	16 GMAC/s	120 GFlop/s (8 core) [13]	100 GMAC/s	256 GMAC/s
Numerical Precision	16-bit Fixed Point	64-bit Floating Point	Variable (8-128 bit) Floating Point	Variable, Fixed Point
On-chip Memory (values are per core in case of multi-core)	L1 cache (I:16KB, D: 32KB) L2 cache: 128 KB 10.5 MB Internal RAM	L1 cache (I:64KB, D: 64KB) L2 cache: 4MB	PPE: L1 cache(I:32 KB, D:32KB) L2(512 KB) 256 KB store per SPE	128 RAM blocks 18 Kbits each
Memory Bandwidth	N. A.	32 GB/sec [14]	25.6 GB/s	21–32 GB/s [b]
Power Consumption [c]	4.5 W	N.A.[d]	110 W	4 W
Size[e]	841 mm^2	341 mm^2	221 mm^2	729 mm^2
Development ease	High	High	Medium	Low
Design flexibility	Medium	Medium	Medium	High

[a]As well known, number of MAC/s is an oversimplified metric. Not all operations need MAC instructions, famously the Viterbi decoder, which is one of the most compute-intensive modules. In the absence of full-reference implementations on all platforms under comparison here, we quote this.

[b]Virtex-4 supports many memory interfaces like DDR2 SDRAM, QDR II SRAM. Memory bandwidths are different for different interface.

[c]The power consumption of a device depends heavily on the implementation. The numbers here are coarse estimates for peak-loading conditions. Also, since the computation capacities of these platforms are different, a fairer metric is energy consumption or W/MOPS. For instance, MSC8144 DSP and Xilix V4 FPGA might look comparable in power at first look, but the FPGA delivers 16x more performance at the same power!

[d]Power consumption numbers were not available for the Power6 chip, only for servers based on Power6 which will be much higher. As per one estimate, a 4-core p570 server consumes about 1400W.

[e]The size quoted for Cell BE and Power6 is die area whereas those for Xilinx FPGA and MSC8144 are packaged chip areas.

combine their strengths. This choice doesn't come without its challenges - design is more complicated as components are distributed across multiple architectures, with different tool chains and modes of behavior, also needing a larger skill set. Further, fewer components and better system integration can bring savings in power, area, hence cost. An example of a highly integrated Software Radio system is Vanu's multi-standard Anywave implementation which combines all the functionality of Base Transceiver Station(BTS) and Base Station Controller(BSC), including signal processing, on a single server[11]. Ideally, it would be desirable to have massively parallel devices where advanced programming and verification tools accelerate development and the high computational density reduces power and unit price.

IV. DISCUSSION

Table 3 shows a comparison of some commercially available platforms on the criteria discussed so far. All the platforms compared offer high computational capacity and can be considered for implementing a 4G software radio system (though a MSC 8144 implementation is likely to need multiple DSPs). Which platform is the best depends very heavily on the target application. For example, current processors are unsuitable for consumer portable devices as they consume too much power, leaving a combination of FPGAs and ASICs as the only choice. In comparison, for base stations, relatively larger size, cost and power can be tolerated. Depending on the need for quick system updates and time-to-market pressures, the longer development cycle for FPGAs can tilt the decision in favor of the more agile processors, despite FPGAs' obvious advantages in area, power and performance. With ongoing improvements in compute power and power efficiency, DSPs as well general purpose IT platforms are becoming good contenders for software radio systems due to the higher flexibility and better system integration they provide.

V. CONCLUSION

The case for Software Defined Radios (SDR) is universally accepted now and the focus has shifted to its efficient execution. The variety of technology choices available for implementing SDR systems range from ASICs, FPGAs, general and special purpose processors and everything in between. This paper outlines the many design requirements for SDR systems and compares how different technologies perform on them. FPGAs offer more customization and cost-performance-power efficiency than processors but at the cost of more complexity in system design. Along with the benefit of agile development and reconfigurability, processors now provide very high computation rates by using multiple cores and specialized architecture and instruction set. With ongoing improvements in compute power and power efficiency, DSPs as well general purpose IT platforms are becoming good contenders for software radio systems due to the higher flexibility and better system integration they provide. In summary, the space for 4G software radio systems comprises of network edge devices (base stations) and client devices. FPGAs, ASICs, DSPs still appear to be good choice for client devices, but it is worthwhile to investigate the usefulness of GPPs, specially the upcoming multicore processors from most vendors, in designing base stations.中国通信

Acknowledgement

The authors are grateful to friends and colleagues, Vimitha Kuruvilla, Vaibhav Saxena, Prateek Sharma, Rajesh Golani, Sudakshmina Mandal, Shobha R. for discussions which have greatly contributed to our understanding of processors, FPGAs and considerations in implementation of PHY and MAC components in wireless systems.

References

1. J. Mitola, Software radios–survey, critical evaluation and future directions,in Proc. National Telesystems Conference, pp. 13/15-13/23,Washington DC, USA, May 1992.
2. I. Scheiwe, The shift to multi-core DSP Solutions, http://www.dspfpga.com/articles/scheiwe/.
3. QuickSilver Technology: http://www.qstech.com/.
4. Q. Wang, D. Fan, Y.H. Lin, ”Design of BS Transceiver for IEEE 802.16E OFDMA Mode” International Conference on Acoustics, Speech, and Signal Processing (ICASSP), 2008.
5. D. Fan, Q. Wang, Y.H. Lin, Z.B. Zhu, ”Design and simulation of the BS transceiver for IEEE 802.16e OFDMA mode,” IEEE International Symposium on Broadband Multimedia Systems and Broadcasting, pp. 1-5, March 2008.
6. Y. Lin, et. al. A Low-power Architecture For Software Radio, Proc. 33rd International Symposium on Computer Architecture 2006, pp. 89 - 101.
7. The BDTImark2000: A Summary Measure of DSP Speed, a white paper by BDTI Inc., Sep 2004.
8. Jeff Bier, Use a Microprocessor, a DSP, or Both?, presented at Embedded Systems Conference, April 16, 2008.
9. Enabling Technologies for SDR: Comparing FPGA and DSP Performance, presented at SDR Conference, November 15, 2006.
10. BDTI Communications Benchmark for OFDM:http://www.bdti.com/products/services comm benchmark.html.
11. John Chapin, The Vanu Anywave Base Station Subsystem, white paper by Vanu Inc. May 2006.
12. R. Baines and D. Pulley, A Total Cost Approach to Evaluating Different Reconfigurable Architectures for Baseband Processing in Wireless Receivers, IEEE Communications Magazine, Jan 2003.
13. IBM Power Systems performance benchmarks: http://www-03.ibm.com/systems/power/hardware/benchmarks/hpc.html.
14. IBM System p570 Technical Overview and Introduction: http://www.redbooks.ibm.com/redpieces/pdfs/redp4405.pdf.
15. Xilinx Virtex-4 Multi-Platform FPGA: http://www.xilinx.com/products/silicon solutions/fpgas/virtex/virtex4.