ASIC Design of Shared Vector Accelerators for Multicore Processors

Spiridon F. Beldianu and Sotirios G. Ziavras
Department of Electrical and Computer Engineering
New Jersey Institute of Technology
Newark, NJ 07102, USA

***Abstract* -- Vector coprocessor (VP) resources are often underutilized due to the lack of sustained DLP (data-level parallelism) or the presence of vector-length variations in application code. Our work is motivated by: a) the omnipresence of vector operations in high-performance scientific and embedded applications; b) the need for performance and energy efficiency; and c) applications that must often handle various vector sizes. Our design for VP sharing in multicores enhances performance while maintaining low area and energy costs. Our 40nm ASIC design yields 16.66 GFLOPs/Watt. Also, a detailed clock and power gating analysis further proves the viability of our approach.**

Keywords* – *ASIC design; vector processor; multicore processor; power management.

I. INTRODUCTION

Hardware accelerators (e.g., VPs) are performance and energy efficient for many applications due to their specialization [1-2]. Vector operations apply the same instruction simultaneously to multiple array elements. Vector supercomputers became the epitome of this mode of computation but, due to technological advances, conventional microprocessors as well currently implement some form of this parallelism (i.e., SIMD instructions).

VIRAM [3], SODA [4] and AnySP [5] are vector microprocessors. VIRAM targets multimedia applications using vector execution lanes. SODA and AnySP consist of a single scalar and SIMD integer pipelines that target software defined radio. Matrix-oriented SIMD instruction extensions for microprocessors have been developed as well [6]. Logic cell densities have facilitated soft vector processors on FPGAs [7-10]. It's interesting to note that Intel's data center division announced in June 2014 the single-package integration of its Xeon processor with an FPGA since "based on industry benchmarks FPGA-based accelerators can deliver more than 10X performance gains" [14]. We started intensive relevant research in our laboratory eleven years ago [15].

However, these vector-oriented designs do not address: **a)** the need to share resources in multicores for higher utilization while releasing silicon for the implementation of more cores or the enhancement of existing cores; **b)** runtime resource management of vector resources assigned to the cores since the collective needs of simultaneously running applications are normally in a fluid state; and **c)** runtime energy saving techniques that take into account individual application needs for vector processing [12, 13].

The sharing of VPs by many cores residing on the same processor chip can deal effectively with these issues. VP sharing increases efficiency and lowers energy consumption. We present here the 40nm ASIC VP realization of a shared VP design that we first proposed in [12, 13] in order to demonstrate its feasibility, and also investigate interesting design tradeoffs for embedded-system implementations. Sections II and III summarize the shared VP architecture and the ASIC design flow. Section IV presents its benchmarking.

II. SHARED VP ARCHITECTURE

Fig. 1 shows VP sharing for two cores. The VP, memory crossbar (MC), banked vector memory (VM) and vector memory controller (VMC) are in-house developed IPs modeled in VHDL. The vector lane conforms to the VIRAM vector unit for a single core [3]. Any vector lane can be shared dynamically by multiple cores; it contains a subset of the elements from a vector register distributed along the lanes, an FPU and a memory load/store (LDST) unit.

A distinct vector controller (VC) is attached to each core. It receives vector instructions from it that are of two types: a) instructions to move and process vector data, which are forwarded to vector lanes; and b) control instructions which are forwarded to the attached Scheduler. Control instructions enable communications between cores and the Scheduler for acquiring VP resources, getting the current VP status, and changing the vector length for processing [1]. The core always receives an acknowledgement in response to an issued control instruction. Vector lanes do not compete for resources except to access a memory module. Upon core request, the Scheduler assigns one or more lanes. Their number depends on the application, any pending requests, any currently executing vector processes, and the need to minimize the overall energy consumption.

The VC forms a two-stage pipeline; stage one carries out decoding whereas stage two is for hazard detection and vector register renaming. The VC pushes vector instructions to FIFOs in the chosen lane(s) along with vector element ranges.

LDST and ALU instructions follow separate lane paths. LDST instructions can use non-unit stride, and can carry out indexed memory accesses via MC. Shuffle instructions transfer elements between lanes using communication patterns stored in a vector register. Without loss of generality, FPUs execute IEEE754 single-precision floating-point +, - and *, and can evaluate absolute and negate. We often instantiate additional units for target applications, such as division, square root, multiply-and-add, etc.

Our runtime VP sharing techniques are: coarse-grain temporal (*CTS*), fine-grain temporal (*FTS*) and vector lane (*VLS*) sharing. CTS temporally multiplexes sequences of vector instructions from different threads. Cores and the Scheduler run processes to lock-unlock VP resources. A thread is run to completion or until it stalls due to an expensive event. Compared to dual-cores with private VPs of the same size as the shared VP, CTS halves the monetary and energy costs and increases the VP efficiency [1]. FTS executes simultaneously within a lane instructions from different threads that use different resources. It resembles simultaneous multithreading (SMT) for conventional processors, thus increasing resource utilization and VP throughput. Finally, VLS assumes a divisible VP with clusters of lanes assigned simultaneously to the threads. It facilitates the runtime resizing of VP resources assigned to cores.

We first created an FPGA prototype. The results show that FTS exhibits the highest speedup and smallest energy consumption, followed by VLS. Under low resource utilization, FTS doubles the speedup and reduces the energy by about 50% as compared to cores with VP exclusive access. Performance and power estimation models were developed to aid the runtime system. As expected, we can improve the performance and reduce the energy consumption by increasing either the application’s vector length (e.g., via loop unrolling) or the thread population. Further analysis shows that the last technique is superior to the former. Thus, the lack of adequate DLP in applications can be overcome via VP sharing. Finally, VP sharing for a dual-core yields speedups of 1.2-2 and halves the energy needs as compared to a system having a single core with an attached VP. With the performance expressed in clock cycles, VP sharing demonstrates 3.62-7.92 speedups compared to compiler-optimized Xeon runs.

III. ASIC DESIGN

To apply a 40nm TSMC ((Taiwan Semiconductor Manufacturing Company) ASIC process, we replaced several Xilinx IP cores of the FPGA prototype. The FPU +, - and * operations were optimized to a pipelined latency of 6 clock cycles. This latency is inherited from the prototype due to our intention to compare directly the FPGA and ASIC consumptions. LDST address computation employs Synopsys DesignWare Basic Block integer multipliers. Big register files for the VP require SRAM to retain the state. However, the available Synopsys and TSMC libraries do not provide SRAM models for feature sizes smaller than 90 nm. CACTI 6.0 [1] is used to deduce area/delay/power figures for the VRF (vector register file) and VM SRAM banks. Table I shows values for 1 GHz and 40nm. The VRF has four banks in each lane; a bank has 128 32-bit elements. A VM bank has 2048 32-bit elements. The chosen VRF size can store all the vector registers for the most register consuming benchmark in Section IV. It corresponds to FFT with 8 vector lanes and 8 memory banks (8x8 VP configuration); it requires 18 vector registers of 32 or 64 elements each. Similarly, the chosen VM bank can fit enough data for matrix multiplication at the available bus bandwidth.

To extract accurate performance and power figures, the rest of the environment was implemented with a Verilog testbench capable of issuing the traces of VP instructions from the chosen benchmarks. Due to tight performance constraints, as shown in Table I, six access ports (4 read and 2 write) were implemented in each VRF bank. This results in 1.061 Volts and 1.10 GHz figures. Pairs of read and write ports are present in each VM bank, requiring a VDD of only 0.661 Volts at 1.24 GHz.

Synopsys VCS-MX is used to simulate-verify the RTL design (the netlist is produced after synthesis). The Synopsys Design Compiler is used for synthesis, with Synopsys PrimeTime for timing and power analysis. Synopsys front-end design and power flow involves: i) simulation of RTL logic using Synopsys VCS-MX for performance; ii) synthesis with the Synopsys Design Compiler; iii) netlist simulation with Synopsys VCS-MX and testbench stimuli; and (iv) extraction of power consumption using Synopsys Primetime-PX, based on node activities.

Power estimation with Primetime-PX requires: i) the netlist generated after synthesis; ii) the Switching Activity Interchange Format (.SAIF) file generated after gate-level simulation; iii) time constraints and the wire load model for parasitic capacitance and resistance; and iv) the vendor’s technology libraries.

Our RTL models infer fine-grain clock gating extensively during Design Compiler synthesis, thus capturing most of the remaining clocked elements. The power associated with the clock distribution network is substantially reduced. Fig. 2 plots a lane’s FPU power consumption for various activity rates (numbers of operations in 100 clock cycles). It shows the benefits of clock gating. The standby power is only 40% of that consumed without gating. As the activity rate increases, more gates are used in clock gating and more clock routes are open.

We target the 40 nm TSMC High Performance (HP) process for personal computers, networking and communications. It has Multi-Voltage support with Low, Nominal and High Voltage thresholds (Vth). By varying the timing constraints imposed on blocks, the Design Compiler produces a multitude of logic topologies, synthesis mappings and gate sizes that differ in area, power and delay. Tight constraints imply aggressive optimizations and gates with wider transistors that increase the drive strength; this approach results in higher area-power figures.

Fig. 3 shows Pareto trade-off curves involving performance-area-power for the ALU datapath and the process corners of Table II. These corners are identified by varying the fabrication parameters that integrate the design on the wafer. They represent extreme variations covering wide power-performance ranges. Presenting the results in a trade-off manner involving performance, area and power provides a comprehensive picture of the design exploration space. This space spans performance and power ranges of about 4.5× (i.e., 0.44-2 GFLOP/s) and 8× (i.e., 7.6-60 mW). Fig, 3 does not contain the same numbers of points for all process corners because some processes cannot reach respective performance values.

We conclude: i) for a given process corner, the area and power consumptions increase with performance increases; ii) area Pareto points for the high speed process (1.21 V and Low Vt) dominate other Pareto process points even at low performance; iii) at low performance, low speed process corners dominate high speed corners in power consumption. E.g., at 0.66 GFLOP/s, PC_03 dominates PC_01 and PC_02; iv) finally, only PC_04 provides performance greater than 1.1 GFLOP/s.

IV. BENCHMARKING

Table III shows the maximum frequency of main components. The wire load model used in synthesis is “TSMC512K_Lowk_ Conservative“; this non-linear model assumes high parasitic capacitance and resistance. The frequency can be improved with a relaxed model. Since the register banks, which are the slowest components, can operate at 1.1 GHz, simulations are carried out for 1.0 GHz. Table IV shows area and power results for all components. Per Fig. 4, more than three quarters of a lane are occupied by the four SRAM banks that implement the VRF. Only 13.7% of the VP area is taken by custom logic; the rest is occupied by memory blocks within the VRF and VM. More than 66% of the consumption is due to the ALU and LDST units. In addition to our implementation of clock gating that reduces the dynamic power consumption, entire VRF banks in a lane are power gated if they cannot be used by combinations of vector kernels running simultaneously on the cores. Thus, the static power of unused banks is eliminated from the total power budget.

Fig. 5 focuses on the area and power scalability of the crossbar switch. This module is the only component that does not scale linearly; its area and power consumptions increase quadratically with the number of lanes. In fact, for more than 16 lanes the area increases more than quadratically because of tight timing constraints. However, custom interconnect fabrics [11] can be used for VPs with more than 16 lanes. It is future work.

Our benchmarking involved the vector kernels: 32-tap FIR, 32-point complex FFT, 1024x1024 matrix multiplication (MM), LU decomposition (LU), and sparse matrix-vector multiplication (SpMV) consisting of two stages. In the first stage (SpMV_k1), array values are multiplied with the corresponding elements from the vector. In the second stage (SpMV_k2), additions are performed along each row.

SpMV_k1 requires a higher number of irregular vector VM accesses (using vector index loads). Due to its high complexity, this is the only sparse kernel shown in Fig. 6 and Table V. For better performance, loop unrolling is sometimes employed (e.g., FIR32, MM and SpMV_k1). Without loss of generality, the cores run the same vector kernel for FTS and VLS, but not for CTS.

Fig. 6 shows power consumption results for FTS, which generally produces better performance than CTS and VLS. FIR and FFT exhibit higher dynamic ALU power consumption as compared with the LDST unit. MM, LU and SpMV demonstrate high LDST utilization, and thus high consumption for the LDST controller and the VM banks. The leakage power is 13.39-19.24% of the total power consumption for FIR32 and SpMV.

A major power consumer is the clock distribution network, even for an idle VP. It is the result of flip-flops (FFs) that are not clock gated, the clocked gates inferred by the synthesis tool and relevant interconnections. The impact can be reduced by decreasing the FFs in pipelines and controllers. If the leakage power is reduced by increasing the circuit speed to the maximum working frequency, the clock network power consumption will not keep up since it increases with the clock frequency. A possible solution is to use, on top of fine-grain clock gating forced by the synthesis tool, architecture-level clock gating; i.e., coarse-grain on-off control of clock signals that feed VP units which are idle for non-negligible periods of time (e.g., more than 4-10 clock cycles).

The standby power consumption is about 162 mW and accounts for 38-54% of the total power consumption. With the power gating of unused VRF banks, the MM under FTS and the loop unrolled once is the best case; the standby power accounts for 28.95% of the total consumption. For a given application, the lowest dynamic energy is almost the same for all VP sharing techniques as it depends primarily on the types of operations and their numbers. If the standby power is also included, the advantage of FTS and VLS over CTS is substantial, especially under low average VP utilization (e.g., SpMV). It achieves up to 20 pJ/FLOP (multiply unit). Table IV shows that the shared VP can achieve 55 pJ/FLOP (FIR32 and LU under FTS). The difference of 35 pJ/FLOP represents data supply (controllers, VRF and memory) and instructions (VCs, and instruction queues and processing). The largest consumption results from operations on sparse matrices since substantial energy is spent to move data between LDST and VM. For a VP with 8 lanes, we get 8 GFLOPs at 1 GHz and 480 mWatts. The power efficiency is 16.66 GFLOPs/Watt. The AMD Radeon HD 5450 GPU at 40 nm consumes 19.1 Watts at 104 GFLOPs peak (5.44 GFLOPs/Watt).

V. CONCLUSIONS

This paper presented the ASIC design of a shared vector coprocessor (VP) for multicores using a TSMC 40nm process. Clock gating and power gating were applied in order to minimize the energy cosnumption. The importance of VP sharing for high performance and high resource utilization in multicores was shown earlier using simulations [1]. The ASIC-based evaluation results in this paper involved performance and power tradeoffs as well as a diverse set of vector benchmarks. The vector coprocessor can achieve a high power efficiency of 16.66 GFLOPs/Watt.

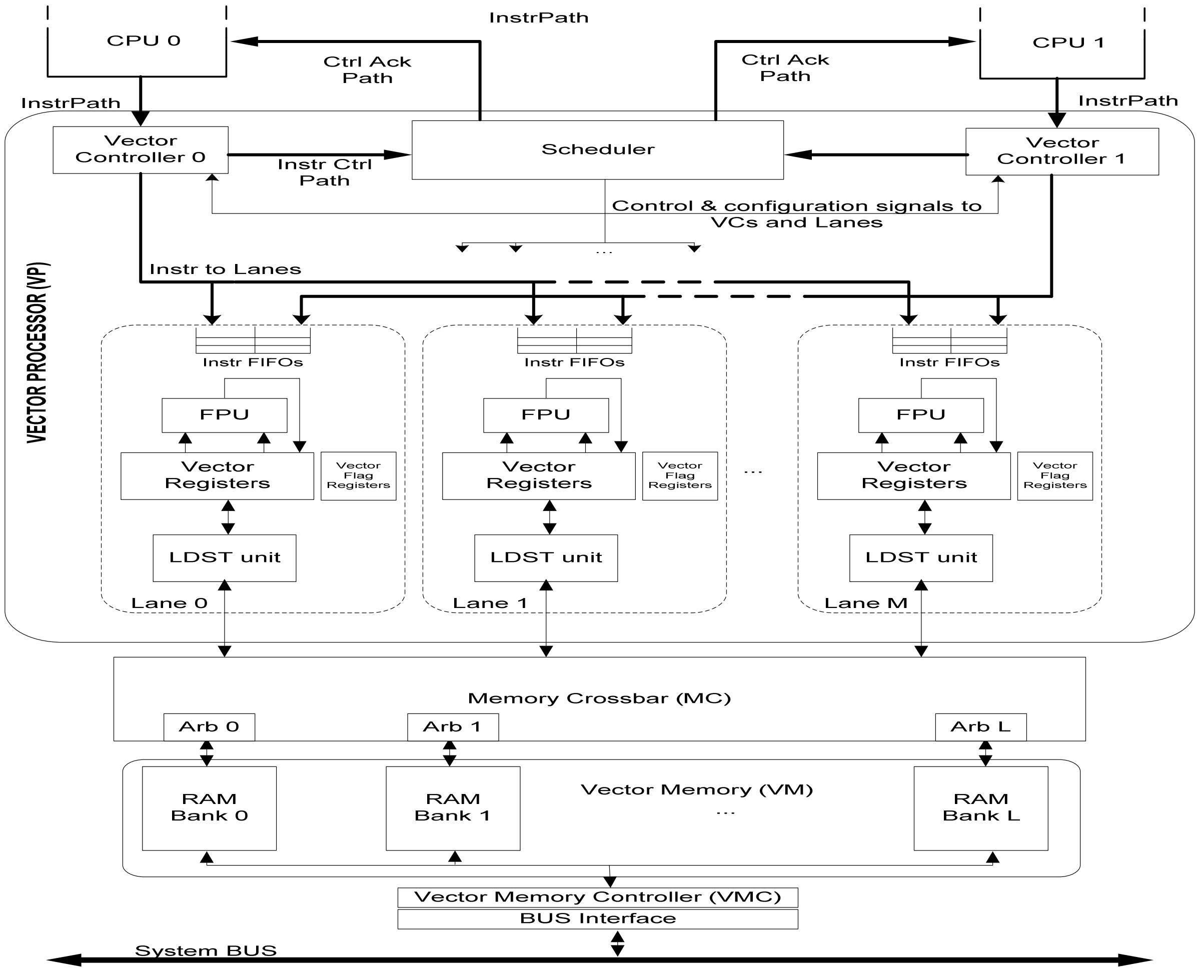

Fig. 1. MxL VP (M vector lanes and L memory banks) shared by two cores. LDST: Load/Store unit.

TABLE I. VRF AND VM AREA AND POWER CONSUMPTION FIGURES.

Vector Register File (VRF) in a vector lane • 4 Read ports and 2 Write ports (total: 6 ports) • 128 32-bit elements per bank, with 4 banks. Total: 2 KBytes • VDD: 1.061 V • Access time: 0.9096 ns; max. frequency 1.10 GHz • Total dynamic energy per read port operation: 1.92 pJ • Total standby leakage power per bank: 2.098 mW • Total area: 28017.44 μm^2
Vector Memory (VM) bank • 2 Read and 2 Write ports • 2048 32-bit elements (8 KBytes)

- VDD: 0.661 V
- Access time: 0.8069 ns; max. frequency 1.24 GHz
- Total dynamic energy per read port operation: 3.16 pJ
- Total standby leakage power per bank: 3.102 mW
- Total area: 82835.40 µm^2

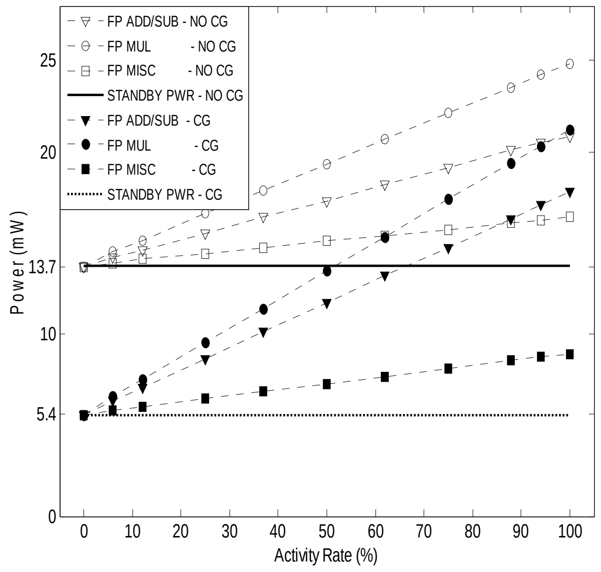

Fig. 2. Total power consumption in a lane for floating-point (FP) +, -, * and average MISC(ellaneous) operations under various activity rates. 40 nm TSMC process. VDD=1.21V. Low voltage threshold. f=1 GHz. Steady state. MISC: Absolute, Negate, Move or Intra-lane Shift. NO CG: No Clock Gating in synthesis. STANDBY PWR: Power w/o operations.

TABLE II. TSMC HP 40nm PROCESS CORNERS (PCs).

PC	**Description**	
PC_01	• High Performance non-well biased with UPF (unified power format) and multi-voltage support • Threshold voltage range: 40 – 85 mV • Temperature: 125 °C	VDD: 1.21 V Vth: Low
PC_02		VDD: 1.21 V Vth: Nominal
PC_03		VDD: 1.21 V Vth: High
PC_04		VDD: 0.99 V Vth: Low

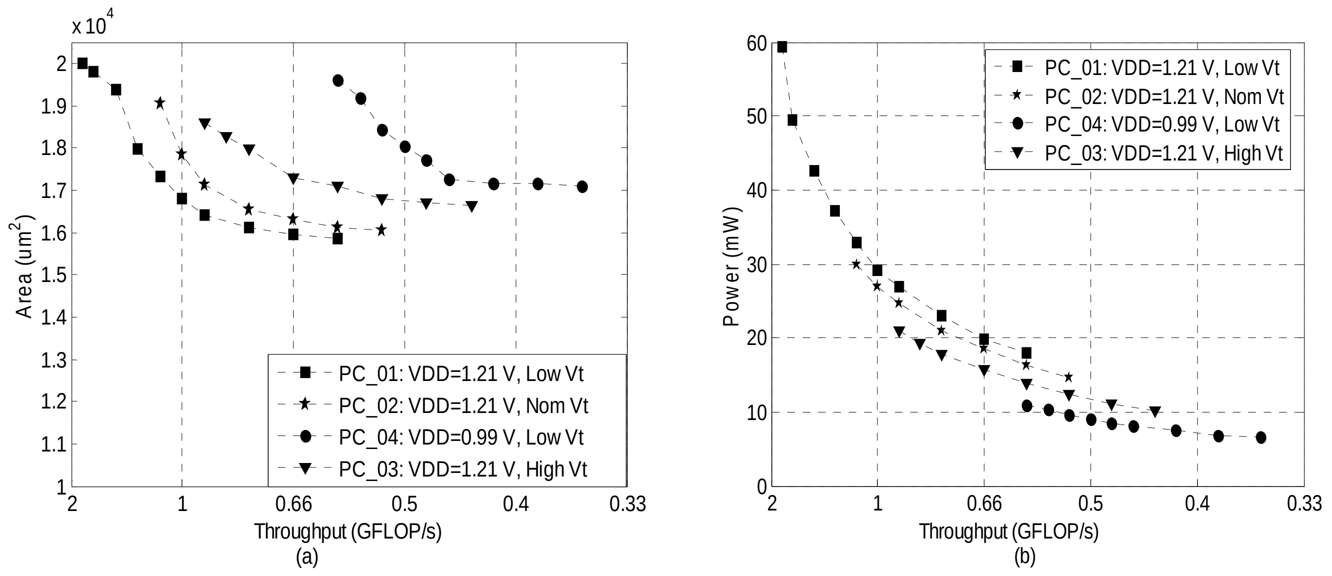

Fig. 3. Pareto trade-off curves for the lane ALU involving: (a) performance and area; (b) performance and power.

TABLE III. MAX. FREQUENCY OF MAIN VP COMPONENTS.

		Max. GHz
ALU	ALU_CTRL	2.08
	ALU_ADD/SUB	1.98
	ALU_MUL	1.78
LDST	LDST_CTRL	1.97
VC		2.12

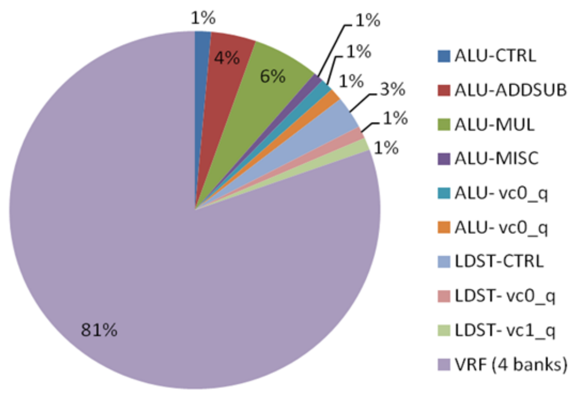

Fig. 4. Vector lane area consumption breakdown.

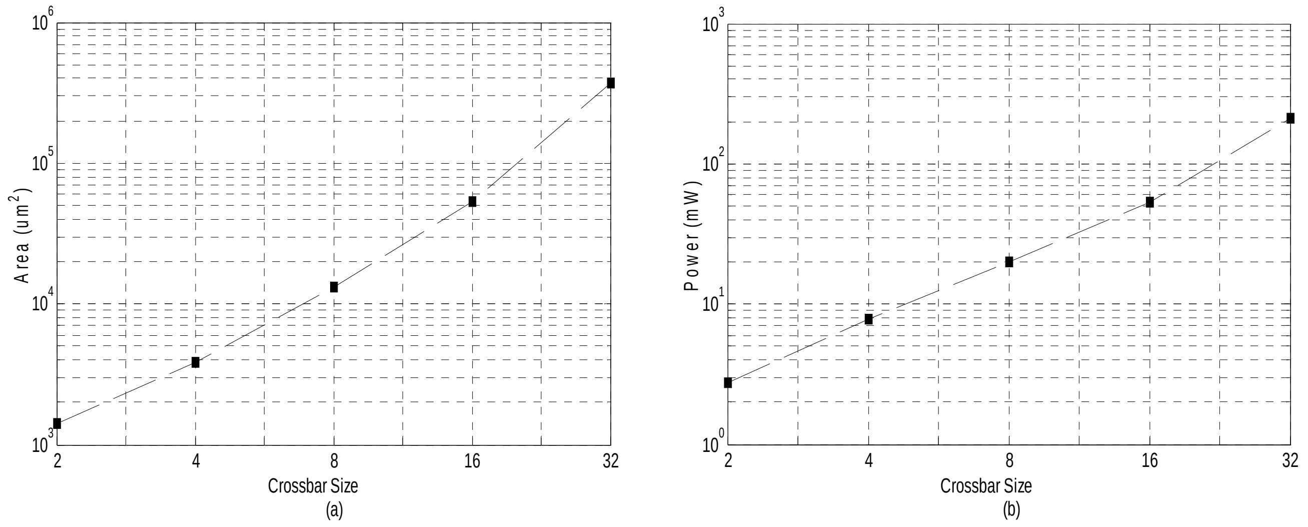

Fig. 5. (a) Area and (b) power consumption of an NxN crossbar switch in the N×N VP. The crossbar contains arbiters and logic for data shuffling. f=1GHz. Max. LDST utilization assumed.

REFERENCES

[1] N. Muralimanohar, R. Balasubramonian and N. Jouppi, "Optimizing NUCA Organizations and Wiring Alternatives for Large Caches with CACTI 6.0," *40th Inter. Symp. Microarch*., 2007, pp. 3-14.

[2] C. Lemuet, *et al.*, "The Potential Energy Efficiency of Vector Acceleration," *ACM/IEEE Superc. Conference*, 2006, pp. 73-90.

[3] C. Kozyrakis and D. Patterson, " Scalable Vector Processors for Embedded Systems," *IEEE Micro*, vol. 23, no. 6, 2003.

[4] Y. Lin, *et al.*, "SODA: A Low-power Architecture for Software Radio," *33rd Int. Sym. Computer Arch*itecture, 2006, pp. 89-101.

[5] M. Who, *et al.*, "AnySP: Anytime Anywhere Anyway Signal Processing," *IEEE Micro*, vol. 30, no. 1, 2010, pp. 81-91.

[6] F. Sanchez, *et al.*, "On the Scalability of 1- and 2-dimensional SIMD Extensions for Multimedia Applications," *IEEE Intern. Symp. Perf. Analysis Syst. Softw*., 2005, pp. 167-176.

[7] H. Yang and S.G. Ziavras, "FPGA-based Vector Processor for Algebraic Equation Solvers," *IEEE Int. System on Chip Conf.*, 2005, pp. 115-116.

[8] J. Cho, *et al.*, "An FPGA Based SIMD Processor with a Vector Memory Unit," *IEEE Int. Symp. Circ. Systems*, 2006, pp. 525-528.

[9] P. Yiannacouras, *et al.*, "VESPA: Portable, Scalable, and Flexible FPGA-based Vector Processors," *ACM Int. Conf. Comp. Archit. Synth. Emb. Syst*., 2008, pp. 145-166.

[10] J. Yu, *et al.*, "Vector processing as a soft processor accelerator," *ACM Trans. Reconfig. Technology Systems*, vol. 2, no. 2, 2009.

[11] S. Satpathy, *et al.*, "SWIFT: a 2.1Tb/s 32x32 Self-arbitrating Manycore Interconnect Fabric," *Symp. VLSI Tech. Circuits, 2011*.

[12] S.F. Beldianu and S.G. Ziavras, "Multicore-based Vector Coprocessor Sharing for Performance and Energy Gains," *ACM Trans. Embedded Computing Systems*, *v*ol. 13, no. 2, Sept. 2013.

[13] S.F. Beldianu and S.G. Ziavras, "Performance-Energy Optimizations for Shared Vector Accelerators in Multicores," *IEEE Trans. on Computers*, accepted for publication, 2013 (preprint: http://www.computer.org/csdl/trans/tc/preprint/06718035.pdf)

[14] P. Clarke, "Intel to Package FPGA with Xeon Processor," http://www.design-reuse.com/news/exit/?id=34847&url=http%3A%2F%2Felectronics360.globalspec.com%2Farticle%2F4313%2Fintel-to-package-fpga-with-xeon-processor, *Design & Reuse*, June 19, 2014.

[15] X. Wang and S.G. Ziavras, "Performance Optimization of an FPGA-Based Configurable Multiprocessor for Matrix Operations ," *IEEE Intern. Conf. Field-Program. Techn.*, Tokyo, Japan, Dec. 2003.

TABLE IV. AREA-POWER CONSUMPTION OF COMPONENTS, AND TOTAL AREA FOR VARIOUS VP CONFIGURATIONS. STANDBY POWER: VP IS IDLE. MAX. POWER INCLUDES STANDBY POWER. % VALUES RELATIVE TO THE FIRST MODULE IN THE HIERARCHY (I.E., ALU OR LDST). f= 1.0 GHZ. THE TOTAL AREA INCLUDES THE VM. **AN EQUIVALENT GATE HAS 4 TRANSISTORS.*

Component		**Area (µm^2)**	**Power (mW)**		
			Leakage	**Standby**	**Max**
ALU		20659 (100%)	2.756	9.504	31.3
	ALU_CTRL	1951 (9.4%)	0.236	2.713	4.58

	ALU_ADD/SUB	5482 (26.5%)	0.717	2.311	21.20
	ALU_MUL	8126 (39.3%)	1.212	2.817	14.70
	ALU_MISC	1271 (6.2%)	0.144	0.239	3.62
	ALU_vc0_queue	1571 (7.6 %)	0.177	0.308	0.617
	ALU_vc1_queue	1571 (7.6 %)	0.182	0.308	0.617
LDST		7213.35 (100 %)	0.884	6.307	12.31
	LDST_CTRL	4081.91 (56.6 %)	0.66	5.68	11.21
	LDST_vc0_queue	1563.0879 (21.7 %)	0.177	0.307	0.758
	LDST_vc1_queue	1558.32 (21.6 %)	0.182	0.312	0.758
VRF - Latch based (one bank)		35637	3.891	3.891	8.979
VRF - SRAM (CACTI, one bank)		28017	2.098	2.098	2.098+ 4.177/port
VC		3076	0.341	1.553	3.61
Scheduler		2427	0.281	1.134	
Crossbar Switch		14196	1.017	8.369	14
Vector Memory - SRAM (1 bank)		82835	3.102	0	3.102+ 7.763/port

	TOTAL VP AREA	**Equivalent Gate Count***
VP 2 (lanes) × 2 (memory banks)	0.466 mm^2	1166638
VP 4×4	0.911 mm^2	2276331
VP 8×8	1.798 mm^2	4495745
VP 16×16	3.573 mm^2	8934571
VP 32×32	7.125 mm^2	17812148

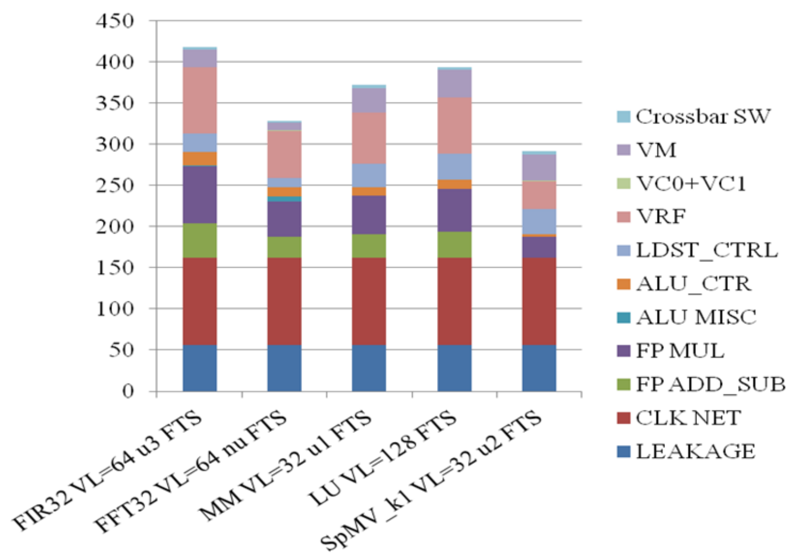

Fig. 6. Power breakdown (mW) for a 8x8 VP under FTS. The total leakage and clock distribution network consumptions are shown separately. f=1.0 GHz. (u*i*, for *i*= 1,2,3: loop unrolled *i* times. nu: no loop unrolling. Crossbar SW: application-driven crossbar consumption. VC0+VC1: consumption of vector controllers. LDST_CTRL, ALU_CTR: LDST, ALU controller. CLK NET: clock network.)

TABLE V. PERFORMANCE-POWER CONSUMPTION OF APPLICATION KERNELS. 8X8 VP. THE THIRD COLUMN SHOWS THE NUMBER OF ACTIVE VRF BANKS USED IN A LANE. STEADY STATE OPERATION WITH f=1.0 GHZ. (nu: NO LOOP UNROLLING; u1: LOOP UNROLLED ONCE.)

		Ungated / Total banks	**Average Utilization (%)**		**Time (µs)**	**Dynamic mW**	**Total mW**	**Dynamic nJ**	**Total nJ**	**pJ/ FLOP**
			ALU	**LDST**						
FIR32 VL=128 nu	CTS	1 / 4	39.24	19.94	0.0207	106.21	218.13	2.19	4.515	71
	FTS	2 / 4	75.66	38.24	0.0107	204.78	333.5	2.20	3.568	55
	VLS	2 / 4	49.51	25.29	0.0164	134.00	262.72	2.19	4.308	67
FFT32 VL=32 nu	CTS	2 / 4	43.29	23.38	0.408	89.64	218.36	36.57	89.091	139
	FTS	4 / 4	76.28	42.39	0.230	157.95	320.27	36.40	73.662	115
	VLS	4 / 4	62.74	35.11	0.274	129.91	292.23	35.59	80.071	125
MM VL=128 u1	CTS	1 / 4	68.3	69.51	0.376	221.67	333.59	83.40	125.43	60
	FTS	2 / 4	97.32	98.91	0.264	315.86	444.58	83.46	117.36	56
	VLS	2 / 4	81.88	83.4	0.311	265.75	394.47	82.84	122.68	59
LU VL=128 nu	CTS	1 / 4	36.17	36.53	0.079	116.357	228.27	8.683	18.033	70
	FTS	2 / 4	72.05	72.92	0.0395	231.781	360.5	8.634	14.240	55
	VLS	2 / 4	47.23	47.67	0.059	151.936	280.65	8.641	16.558	64
SpMV_k1 VL=32 u1	CTS	1 / 4	9.35	38.2	422.25	68.53	180.45	28937	76195	454
	FTS	2 / 4	18.22	73.39	213.87	133.54	262.26	28562	56089	334