In the 2004 International Symposium on Computer Architecture, Munich, Germany

Synchroscalar: A Multiple Clock Domain, Power-Aware, Tile-Based Embedded Processor

John Oliver [†], Ravishankar Rao [†], Paul Sultana [†],
Jedidiah Crandall [†], Erik Czernikowski [†], Leslie W. Jones IV [‡],
Diana Franklin [‡], Venkatesh Akella [†], and Frederic T. Chong [†]
University of California, Davis [†]
California Polytechnic State University, San Luis Obispo [‡]

Abstract

We present Synchroscalar, a tile-based architecture for embedded processing that is designed to provide the flexibility of DSPs while approaching the power efficiency of ASICs. We achieve this goal by providing high parallelism and voltage scaling while minimizing control and communication costs. Specifically, Synchroscalar uses columns of processor tiles organized into statically-assigned frequency-voltage domains to minimize power consumption. Furthermore, while columns use SIMD control to minimize overhead, data-dependent computations can be supported by extremely flexible statically-scheduled communication between columns.

We provide a detailed evaluation of Synchroscalar including SPICE simulation, wire and device models, synthesis of key components, cycle-level simulation, and compiler- and hand-optimized signal processing applications. We find that the goal of meeting, not exceeding, performance targets with data-parallel applications leads to designs that depart significantly from our intuitions derived from general-purpose microprocessor design. In particular, synchronous design and substantial global interconnect are desirable in the low-frequency, low-power domain. This global interconnect supports parallelization and reduces processor idle time, which are critical to energy efficient implementations of high bandwidth signal processing. Overall, Synchroscalar provides programmability while achieving power efficiencies within 8-30X of known ASIC implementations, which is 10-60X better than conventional DSPs. In addition, frequency-voltage scaling in Synchroscalar provides between 3-32% power savings in our application suite.

1. Introduction

Next-generation embedded applications demand high throughput with low power consumption. Current approaches often use Application-Specific Integrated Circuits (ASICs) to satisfy these constraints. However, rapidly evolving application protocols, multi-protocol embedded devices, and increasing chip NRE costs all argue for a more flexible solution. In other words, we want the flexibility of a programmable Digital Signal Processor (DSP) with energy efficiency more similar to an ASIC. We propose the Synchroscalar architecture, a tile-based DSP designed to efficiently meet the throughput targets of applications with multi-rate computational subcomponents. We focus upon next-generation signal processing applications which can not be efficiently supported on today's DSPs, including Orthogonal Frequency Division Multiplexing (OFDM) for 802.11a, MPEG4 encoding, stereo feature extraction and correlation, and software radio digital down conversion. Contrary to traditional microprocessor design goals of the highest performance possible, our goal is to design the lowest power solution for set performance targets. Consequently, our metric of success is the lowest system power to achieve a solution, not raw performance.

While conventional wisdom credits the low power of ASIC implementations to their low area per operation [8], Synchroscalar invests area to achieve programmability while compensating with voltage scaling to achieve low power. Specifically, Synchroscalar uses multiple processor tiles and wide buses to exploit parallelism in order to achieve performance targets while running at low frequencies. Ideally, linear gains in performance translate to quadratic reductions in power due to voltage scaling.

In designing Synchroscalar, we focused on several key features of ASICs that lead to their energy efficiency – high parallelism, low control overhead, and custom interconnect. Our design achieves power efficiencies within 8-30X of known ASIC implementations, which is 10-60X better than conventional DSPs. The success of the Synchroscalar design stems from the nature of its target application class – exploiting their multi-rate structure, intra-task data parallelism, and statically predictable control and communication. To this end, Synchroscalar uses a column-oriented 2D tile structure that follows three design principles.

First, Synchroscalar exploits parallelism to perform voltage scaling. We minimize hardware complexity by scaling voltages spatially rather than temporally. Columns of processors are statically assigned voltages rather than dynamically varying voltage for each processor. Computations are mapped to the appropriate frequency and voltage, and communication facilitates moving from one voltage domain to another.

Second, Synchroscalar amortizes control overhead by grouping each column of processors into a single thread of control, implemented with a single SIMD control unit and program memory.

Third, Synchroscalar minimizes communication overhead through substantial investment in statically configurable interconnect. Specifically, the Synchroscalar's low clock frequencies enable the use of wide segmented buses. Because communication can be heavily data dependent and consequently inefficient to manage under SIMD control, we introduce a decoupled communication controller in each column to orchestrate data motion using static schedules. This enables extremely low overhead register-to-register inter-tile communication which allows us to compete with the dedicated interconnects of ASICs.

In the remainder of this paper, we provide an overview of the Synchroscalar architecture and our multi-rate applications to establish the context of our study. Then we describe our evaluation methodology, including power models, SPICE simulations, VHDL synthesis, software tool chain, and cycle-level simulation. We analyze our results and discuss our intuitions from this analysis. We then conclude with related and future work.

2. The Synchroscalar Architecture

The high level observation that led to this design was that if an application can be parallelized efficiently on an architecture, then the clock and voltage can be scaled down in order to reduce power consumption. The column-oriented nature of Synchroscalar allows us to greatly reduce the complexity of control, communication, clock distribution and voltage scaling. SIMD controllers are used to amortize the control overhead and support efficient application parallelization in each column, while Data Orchestration Units (DOUs) provide communication flexibility by supporting statically-scheduled zero-overhead irregular communication. Interconnect bandwidth is highest within and between columns, in order to provide high-speed communication within an application. Lower bandwidth is required for communication between components. Additionally, each column of four tiles is supported by a specific clock generator and voltage and are configured at startup.

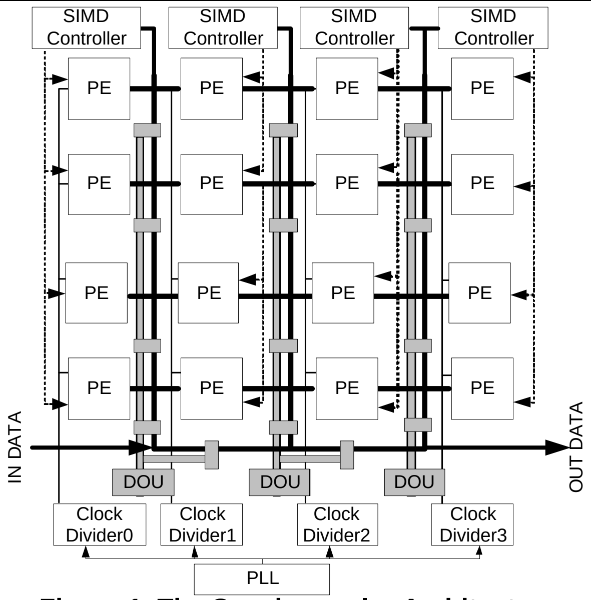

Figure 1. The Synchroscalar Architecture

We will use the Digital Down Converter (DDC) application as an example of how the parallelization and mapping process works. Parallelization begins by recognizing stages in the application with a specific data rate between each stage. The first two stages of this application are the digital mixer and the CIC integrator (see Section 3 for full application descriptions). After exploring the trade-offs between computation and communication with varying levels of parallelization (described in Section 5) we find that the first stage, the mixer, should run on 8 tiles and the integrator on 8 tiles to minimize power consumption. The mixer is then mapped to the first two columns and the integrator to the third and fourth columns.

Once the parallelization and mapping is complete, the clock and voltage can then be scaled down based on the application needs. Given the DDC's target execution rate of 64 million samples per second, the mixer tiles need to run at 120 MHz and the integrator tiles at 200 MHz. These clock rates are generated from reference clocks which are fed into clock dividers in each column as shown in Figure 1. Supply voltages are also externally supplied, and SPICE simulations from Section 4 indicate that the mixer tiles can operate at 0.8V and the integrator tiles at 1.0V. With this simple example and overview in mind, the remainder of this section describes the major components of Synchroscalar in greater detail.

2.1. Parallelism

Parallelism is critical to the success of Synchroscalar, for it is through parallelism that we can reduce the clock frequency, and thus voltage, while continuing to meet performance targets. In this we are greatly aided by the statically-predictable, highly data-parallel nature of signal processing. While our applications are all hand-parallelized in this study, future work will focus on automated tools. We believe that automation is realistic, since our applications fit the Synchronous Dataflow (SDF) model of computation used in existing DSP design tools such as Ptolemy from UC Berkeley [6, 7, 9] and Simulink from Mathworks and SPW from Cadence.

The dataflow models allow for two forms of parallelism - within a Synchroscalar column and between columns. SDF also provides predictability by restricting the number of data values produced and consumed by a task to be a constant. This restriction imposed by the SDF model offers the advantage of static scheduling and decidability of key verification problems such as bounded memory requirements and deadlock avoidance [21].

2.2. SIMD Control

Low-overhead control is critical to the efficiency of ASICs. The data-parallel nature of signal processing applications allows a reduction in the cost of instruction fetch and decode through a single SIMD controller that sends instructions to the tiles in a column. The SIMD controller performs all control instructions, only forwarding computation instructions to the tiles. To communicate data for conditional branches, the SIMD controller is connected to the segmented bus with the tiles.

In order to support branch prediction, there would need to be a mechanism to squash instructions that have already been sent to the processing elements. Instead, we provide a short pipeline in the control unit to calculate branches quickly, and delay instructions from reaching the processing elements. This introduces a single-cycle stall for each conditional branch. For zero-overhead loops, there is still no delay, because the PC is used for decision making, not the actual instruction.

Note that applications do not always parallelize evenly into columns of 4 tiles, requiring occasional idle tiles. Idle tiles are assumed to consume negligible power through supply gating, so we sacrifice their area to simplify our design. Idle tiles are decided at startup.

2.3. Reconfigurable Interconnect

Synchroscalar exploits parallelism to increase efficiency, but these gains must not be lost to the communication overhead to support this parallelism. In particular, latency-critical communication must not be allowed to increase the idle time of power-hungry processor tiles, especially when idle times are too small to shut down tiles.

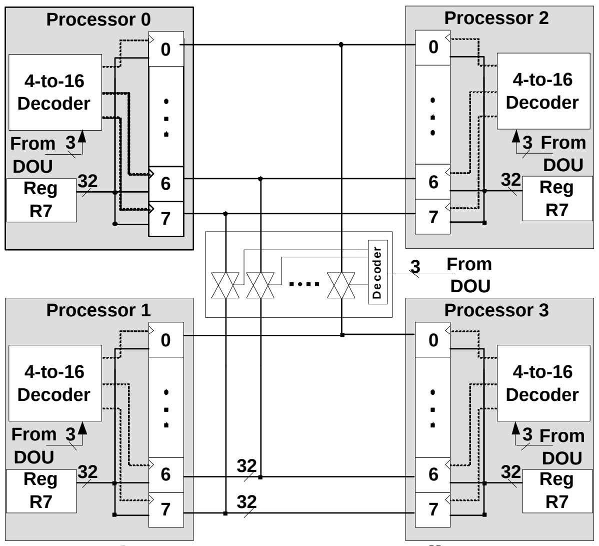

Figure 2. Segment Controllers

Given our design goal of low system clock rates, we find that we can approximate the specialized interconnects of ASICs through a combination of segmented buses and a decoupled communication controller called the Data Orchestration Unit (DOU). Refer back to Figure 1 to see these buses and controllers arranged in each column. We note that signal processing applications exhibit much higher communication requirements within computational blocks than between blocks. Consequently, we allocate only a single horizontal bus between columns, which both meets bandwidth requirements and facilitates gather-scatter operations.

Synchroscalar buses are 256 bits wide, grouped into 8 32-bit separable vertical buses that are segmented in between each of the tiles. Although 256 bits wide might seem power-hungry, we shall see in Section 4 that the power consumption of the busses is small compared to the cost of supporting a higher frequency tile.

In addition, by suitably controlling the segment controllers, the bus can perform several parallel communications. For instance, if all the controllers are turned on, the bus becomes a low-latency broadcast bus, and all tiles able to receive the same data in a single cycle. Alternatively, two messages can pass between neighboring tiles using the same wires in different segments, achieving the approximate bandwidth of a mesh if code is allocated to the tiles intelligently. The segmentation of the bus allows Synchroscalar to achieve higher levels of local bandwidth for very little cost in area and power and reduces tile idle time due to remote data dependencies.

Data Orchestration Unit (DOU) A key feature of Synchroscalar is statically-scheduled communication provided by the DOU decoupled controllers located in each column. The goal of the DOUs are to provide zero-overhead

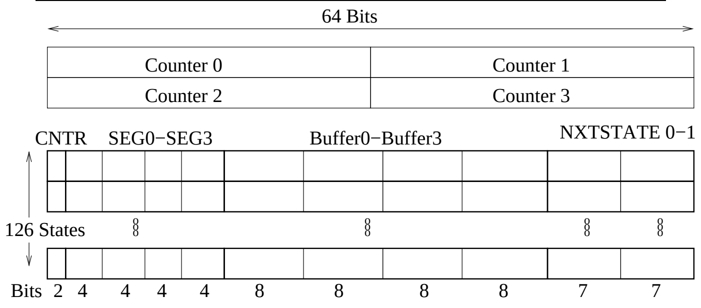

Figure 3. DOU Implementation

data movement between producer and consumer tiles. A producer writes to a special register, and, at a statically-scheduled time, a consumer can read that value from a receive register. The DOUs provide separate, cycle-by-cycle control of data motion and interconnect configuration. This flexibility facilitates irregular data motion and allows our applications to be efficiently scheduled in SIMD tile computations. The DOU operates at the maximum frequency, the frequency of the bus. Since the DOUs are very small the power contribution of the DOUs is minimal.

There is one DOU for each of the columns on Synchroscalar. The gray boxes that overlap the data bus in Figure 1 represent the segmenters, and the gray lines that connect DOU to the segmenters are the control lines that are necessary to control each of the 8 splits. Figure 2 depicts a detailed logical diagram of the segmenters.

The DOU is simply a state machine, where each of the DOU's state's outputs control the segmenters. The DOU must be programmed with the desired communications patterns for the column-bus it controls. There are 128 states in the DOU. Each state entry in the DOU has five types of fields, CNTRi , SEGi, Bufferi, NXTSTATE0i, NXTSTATE1i,as shown in Figure 3.

The CNTR field specifies which of the four DOU down counters should be checked for a given state in the DOU. If the counter specified by the CNTR field is zero, then the next state is the state pointed to by the NXTSTATE0 field of that given state and the down counter is reset to its initial value. If not, the DOU state machine proceeds to the state pointed to by the NXTSTATE1 field and decrements CNTR. There are four 32-bit down counters that are preprogrammed with the dynamic instruction count of the associated loop, allowing four nested loops. The SEG and Buffer fieldsare the outputs of a given state. They control the bus segmenters for a given column and the communications buffers for each tile in a given column, respectively.

Here is a quick example of the DOU's operation. Figure 4 shows a nested pseudo-code loop. It requires two DOU counters for I and J. The I loop counter would need to be 4*A and the J loop counter would need to be 2*B, assuming the FOR instruction loop can be encoded in a single assembly instruction. The output pattern would need to

```
For(i=0; i<A; i++){
     Outer_Instruction1;
     Outer_Instruction2;
     For(j=0; j<B; j++){
          Inner_Instruction1;
          Inner_Instruction2;
     }
     Outer_Instruction3;
}
```

Figure 4. Example DOU code

be programmed for each of the instructions that access the global data bus. the output pattern is a "don't care".

Synchroscalar Tiles are based on the ADI/Intel Blackfin DSP ISA [20], but with control provided by the SIMD controller instead of in each tile. Additionally, each of the tiles has a read and a write buffer as shown in Figure 2. These buffers have a dual purpose. Their first function is to adapt the tile voltage to the bus voltage, as tile voltages across the Synchroscalar design may be different. Secondly, the buffers align a word of data onto the desired split of the global data bus. Register R7 is the designated communications register on each of the tiles. The DOU controls the alignment of this register on the data bus. Only three bits are required from the DOU to control the placement of the data on the data bus for each of the 8 splits of the bus.

2.4. Clock and Voltage Domains

All of the design decisions above come together to provide clock and voltage scaling per column. Since the applications have known performance needs, the SDF model provides predictable performance and distinct tasks that are exploited by our SIMD column-based design. These columns become separate clock and voltage domains. Each task is performed at the lowest frequency that meets the application constraints and the corresponding voltage, down to the chosen voltage and frequency floors of 0.7 V and 100 MHz.

To further reduce complexity, we support only a small set of frequencies and voltages for a given design. The computational rates of each algorithmic block implemented in each column, however, must be matched to the target data rates. If one block runs too fast, then the subsequent block will not be able to keep up with the data produced.

A simple procedure for matching rates is to choose the minimum frequency necessary for each column, then add *nops* to throttle the computational rate. Unfortunately, adding nops to application code may not be convenient if the throttling rate is not a good multiple of the existing loop structure. Instead, we introduce a simple mechanism for flexible computation throttling in our multi-rate system called *Zero Overhead Rate Matching*. We add a simple programmable counter to each SIMD controller. This counter allows us to periodically dynamically insert nops to the tiles in each column in any period of cycles, thus allowing perfect rate matching.

3. Applications

To drive the design of Synchroscalar, we selected four signal-processing applications, each of considerable complexity involving several computational subcomponents. Each cannot be executed at the required rate by any known commercial DSP at this time. These applications are: Digital Down Conversion, Stereo Vision, 802.11a, and MPEG-4. The next four sections briefly describe each of these applications.

Digital Down Conversion(DDC) Digital Down Conversion (DDC) is an integral component of many communication systems, and functions primarily to convert a received signal to baseband such that the signal of interest can be processed. This particular DDC was configured to support GSM cellular requirements of up to 64 M Samples per second. It is comprised of a Numerically Controlled Oscillator, digital mixer, Cascaded-Integrator-Comb (CIC) filter and a two-stage filter in the form of a compensating 21-tap filter (CFIR) and a 63-tap filter (PFIR).

Stereo Vision (SV) , used in the Mars Rover[26], has two stages: point feature extraction and point feature correlation. Each frame processed is 256 by 256 pixels in monochrome and is processed at a rate of 10 frames per second. Tomasi and Kanade's [10] algorithm for point feature extraction was employed and for point feature correlation, singular value decomposition [30] was used.

802.11a is an end-to-end application. This IEEE standard for wireless communications supports data rates up to 54 Mbps. It is coded using OFDM and employs up to 12 20-MHz channels in the 5 GHz frequency range. The four major components in the 802.11a receiver are the FFT, Demodulation, De-Interleaving and a K=7 Viterbi Decoder.

MPEG-4 is an ISO/IEC standard adopted in 1998. For encoding, we implement Motion Estimation, DCT and Quantization which constitute about 90video encoder [36]. For Synchroscalar, both CIF and QCIF MPEG4 encoding was performed at 30 frames per second.

4. Methodology

We now present an evaluation framework for Synchroscalar including: an application mapping methodology, tile and interconnect power models, VHDL synthesis, and cycle-accurate simulation.

4.1. Implementing Applications on Synchroscalar

In this section, we will outline the procedure used to map applications to Synchroscalar and evaluate their performance and power efficiency. The process involves finding an efficient mapping of an application on the Synchroscalar architecture, validating it for functional correctness and then determining the appropriate frequency and voltage of operation of each column. The frequency and voltage values are plugged into an empirical power model for Synchroscalar to evaluate the power consumption for that mapping. The detailed procedure is outlined below.

1. Start with the description of the application on a single tile.
2. Choose the number of tiles, N, that minimizes power.
3. Partition the application among the N tiles and insert data transfer operation to model the communication between the tiles.
4. Assume every data transfer takes one clock cycle. Statically schedule all the data transfers.
5. NOPS are introduced appropriately to avoid structural hazards due to bus conflicts.
6. Use the cycle-accurate simulator to determine the number of clock cycles required per input data sample. Code and data are in local tile memories when computing the clock cycle count.
7. Given the input data rate and number of cycles required by each tile, frequency of operation for each column of tiles is computed. Let f_i be the frequency of operation of the i^{th} column.
8. Using SPICE and the Berkeley Predictive Technology Models we find the required supply voltage (V_i) for a given frequency and voltage for an assumed critical path delay of 20 FO4s.
9. The total power is estimated using the following equations

$$Power = P_{tile} + P_{interconnect} + P_{leakage}$$

$$P_{tile} = \sum_{i=1}^{k} U * (V_i/V_{ref})^2 * f_i$$

$$P_{interconnect} = 0.5 * C * V_i^2 * f_i$$

where U is defined as the *normalized* power in milliwatts per MHz (mw/MHz) at the reference voltage V_{ref}, and it includes the active power consumed by the tile (including the data memory) and the DOU and the SIMD controller in each column, C is the average bus capacitance switched per cycle, and k is the number of tiles.

Based on the procedure outlined above, it is clear that the key factors that influence our model are - the power model for the tile, the power model for the buses or interconnect and the *leakage power*. Next we will describe how we model these parameters and their validation.

4.2. Tile Model

To model the power of the tile we need two things. First, is the parameter U that that represents the *normalized* power of the tile and its associated components. The second parameter needed is the relationship between the frequency of operation of the tile and the operating voltage.

The parameter U is estimated as follows. The tile consists of 2 40-bit ALUs, 4 8-bit video ALUs, 2 40-bit accumulators, 2 16x16 multipliers, 1 40-bit barrel shifter, a 32x32 register file with four read ports and 2 write ports, 32KB data memory and glue logic. It was modeled in VHDL and synthesized using the Synopsys Design compiler. The multipliers, register file and memory were not synthesized. We mapped the design to a 0.25μ ASIC library, at a supply voltage of 2.5V, and used Design Power to estimate the power from the gate-level netlist. We scaled the results to 130 nm geometry and found

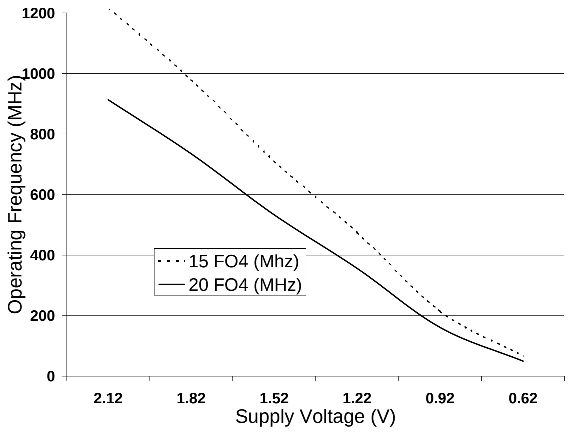

Figure 5. Voltage-Frequency curve for a 20 FO4 pipelined processor

that the normalized power of the datapath was approximately 0.03mW/MHz. To this we added the contribution of the register file (0.11mW/MHz), [27], and the data memory (1.75mW/MHz) [28], by scaling the data appropriately. Hence, the total normalized power of the tile was estimated to be 1.89mW/MHz. To this we add the amortized overhead from the DOU and the SIMD controller. Assuming that there are four tiles per column, the contribution of the SIMD controller and the DOU to the power of each tile is roughly 0.25mW/MHz, for a total normalized of 2.14mW/MHz, which corresponds to the parameter U in the equation above.

We assume that by doing a custom logic implementation with appropriate transistor sizing we would cut the power of the synthesized portions of the logic in the SIMD controller and the tile by around 30%. With this assumption, we estimate the normalized power to be approximately 0.642mW/MHz, which reduces to 0.1mW/MHz at 1V supply. Although no Blackfin core power numbers are available, we can compare our estimate to a similar core from NEC the SPXK5 [37], which consumes 0.07mw/MHz in 130 nanometer technology. Given that we are using an estimate, however, we will discuss the sensitivity of our results to tile power at the end of the results section.

The relationship between operating frequency and supply voltage of a column is found as follows. We assume the critical path is 20 F04 gates, which is pessimistic, but appropriate for an embedded DSP core [16]. Using the Predictive Berkeley Technology Models [17] we SPICE a 20 FO4 critical path and plot the relationship between frequency and voltage. The graph in Figure 5 shows the variation of the frequency and voltage for the 130 nm process assuming critical paths of 15 and 20 FO4 lengths. This graph is captured as a look-up table to determine the appropriate voltage of operation of a tile given the frequency.

Parameter	Value	Source
Technology	130 nm	
Minimum Voltage	0.7V	Blackfi n DSP [20]
Maximum Voltage	1.65 V	Estimated [17]
Threshold Voltage	0.332 V	[17]
Temperature	40 C	Assumed
Oxide Thickness	3.3 nm	[17]
Dielectric Strength of Oxide	5e6 V/cm	[17]
Max Frequency	600 MHz	SPICE using [17]
Tile Power	0.1mW/MHz	See estimate above
Tile Size	1.82 mm^2	From Section 4.6
Wire Cap.	387 fF/μm	Semi global [16]
Wire pitch	16 λ	Semi global wiring [16]

Table 1. Technology Parameters

4.3. Interconnect Model

The interconnect model is largely based on the data from ”The Future of Wires” paper [16]. In 0.18μ tech, the gate capacitance of a minimum sized transistor is about 1-2fF [16]. This value is expected to remain constant over shrinking process technologies. The projected value of wire capacitance for a semi-global wire in 0.13 μ technology is per unit length is 387fF/mm. Assuming length of the chip is about 10mm (that corresponds to the length of the bus) the wire capacitance is about 3870fF. This suggests that even if the drivers and repeaters are 10-times the minimum size, their capacitance is about 20fF. If there are 8 drivers for each bus, it adds only 160fF to the wire capacitance. Also, we find that the gate and drain capacitances are orders of magnitude smaller than the wire capacitance per unit length. The drain-source capacitance of the segmenters and the gate and drain capacitances of the drivers are ignored. Thus the interconnect is modeled by the wire capacitance to a first order approximation. A summary of the key parameters of our model and their sources is given in Table 1.

4.4. Leakage Power Estimation

Given that we are scaling the supply voltage aggressively, it is important to include the contribution of the leakage current in our estimations. Additionally, the fact that we trade area for power in Synchroscalar makes our leakage analysis even more critical. We use an analytical model to compute the leakage current I_{off}

$$I_{off} = I_{on}.e^{\frac{V_{GS}-V_{th}}{\alpha * V_T}}$$

where I_{on} is the *on current* that depends on the process but is roughly equal to 0.3 μA per micron width, $V_T = kT/q$ which is roughly 26 millivolts at room temperature and α depends on the devices structure but is roughly between 1.3 to 1.5 and V_{th} is the threshold voltage.

The leakage current increases with decrease in threshold voltage and increase in temperature. In order to model the leakage, we make the following assumptions:

1. All the devices are operating at the threshold voltage of 0.332V
2. The temperature is 80 degrees Celsius
3. A transistor density of 1 million transistors per mm^2 in 0.13μ technology
4. Tiles that are not used in an application are assumed not to contribute to the leakage current

Using these numbers we calculate the leakage current, I_{off} which happens 830 pA per transistor for a minimum sized transistor. This leakage correlates well with the number published by Intel on their 130 nm process where leakage current varies from 0.65 nA per transistor to 32.5 nA per transistor depending if the threshold voltage of the transistor is high or low, respectively [41].

We estimate 1.8 million transistors per tile, so we believe the leakage power to be around 1.5 mAmps assuming 830 pA of leakage per transistor. Of course, this estimate makes several assumptions, such as the average transistor width. While all results in this paper will assume 830 pA of leakage per transistor, we will present a leakage sensitivity analysis in the results section of this paper.

4.5. Cycle-Accurate Simulation

To obtain cycle-accurate performance measurements, we adapted an object-oriented variant of SimpleScalar to model the Synchroscalar architecture. The instruction set was retargeted to the Blackfin ISA [20] and communication mechanisms were added.

The applications were compiled down to assembly, and the inner-loops hand-optimized. Inter-tile communication is hand-scheduled, and appropriate nops are inserted for synchronization between different clock domains.

4.6. Tile Area Estimation

The tile, the SIMD controller and the DOU were modeled in VHDL and synthesized using Synopsys Design compiler for a 0.25μ ASIC library and scaled to 0.13μ. The various components of the tile and the SIMD controller are shown in Table 2. Memory, register file, and multipliers were not synthesized. Their area was estimated from [15] which has technology independent models for various components. We assume 32KB SRAM of data memory per tile and 2KB SRAM for instruction memory. The $Rest$ field models the glue logic and the wiring overhead between the top-level blocks. The area of the tile is 1.82 mm^2, the area of the SIMD controller and the DOU, which are shared by the whole column of four tiles, is approximately 0.25mm^2 and 0.0875 mm^2 respectively.

5. Results

Since all of our applications have set performance targets, our metric is the system power required to achieve those targets. Table 3 summarizes the primary success of Synchroscalar: software implementation of challenging signal processing applications with energy efficiency generally within 8-30X of ASIC solutions and 10-60X better than DSPs performing even reduced data rate versions of the applications. The remainder of this section describes the benefits of Synchroscalar's unique column-oriented voltage scaling, parallelization's influence on the system power, interconnect costs and leakage current.

TILE COMPONENT	Area $\mu(m^2)$
2 40-bit ALUs	48000
1 40-bit Shifter	500000
2 40-bit Accumulators	11060
2 16x16 mult	100000
32 KB SRAM	5,570,560
32x32 Regfi le 4 read and 2 write ports	650000
Rest	393000
Total	7,270,000
SIMD CONTROLLER and DOU	
DOU	350000
2 KB Instruction SRAM	350,000
Sequencer	225000
LBANK	59000
STACK32	180000
Rest	140000
Total	650000

Table 2. Tile and DOU and SIMD Control Area Estimation

5.1. Power Savings

The multiple column-oriented voltage domains yields advantages as shown by comparing the *Single Voltage* and *Multiple Voltages* columns in Table 4 and in Figure 5.1. Multiple voltages allow power savings of up to 81% for application components and up to 32% for full applications.

For applications where there are a few tiles that run at high frequencies that cannot be parallelized into multiple tiles, we see the greatest power saving due to the voltage scaling. The Stereo Vision application is one such application. In other applications, where there is not one computationally demanding algorithm with limited exploitable parallelism, the power saved due to the voltage frequency scaling is much smaller. The wireless 802.11a application is one such instance. The true benefits of voltage scaling can be better demonstrated when applications need to be composed. This can be seen in the data where we have composed an AES-based message authentication code with the 802.11a receiver.

5.2. Effects of Parallelism

Figure 7 shows how much power is consumed for different levels of parallelization of the our applications. By allocating more parallel resources we are able to run the applications at a lower frequency and a lower voltage, thereby saving power. However, there are diminishing returns for further parallelization in increased communica-

Application	Platform	Process (μ)	Area mm^2	Power (mW)	Voltage (V)	Notes
DDC	**Synchroscalar**	**0.13**	**139.88**	**2427.23**	**.7-1.3**	**Programmable, 64 MS/s**
	Intel Xeon 2.8 GHz [19]	0.13	146	71000	1.45	Programmable, only 19.0 MS/s, 1/3 required rate
	Blackfin 600 MHz [2]	0.13	2.5	280	1.2	Programmable, only 112.6 kS/s, 1/500 required rate
	Graychip [40]	UNK	UNK	250	3.3V	ASIC, 64 MS/s
Stereo	**Synchroscalar**	**0.13**	52.89	**857.40**	**1.2-1.5**	**Programmable, 10 f/s 256x256, stereo**
Vision	Intel Xeon 2.8 GHz [19]	0.13	146	71000	1.45	4.96 f/s, 1/3 required rate
	Blackfin 600 MHz [2]	0.13	2.5	280	1.2	Programmable, 1.46 f/s, 1/7 required rate
	FPGA [5]	UNK	UNK	15K-25K	UNK	30f/s 320x240, not stereo, no SVD, 1.75x rate
802.11a	**Synchroscalar**	**0.13**	**74.05**	**3930.53**	**0.7-1.7**	**Programmable, 54 Mbps RX only**
	Atheros [4]	0.25	34.68	203	2.5	ASIC
	Icefyre [32]	0.18	UNK	720	UNK	ASIC Chipset, including ADC
	IMEC [42]	0.18	20.8	146	1.8	ASIC, area includes ADC/DAC
	NEC [37]	0.18	119	474	1.5	ASIC, MAC+PHY layer, Core Power only
	D. Su [13]	0.25	22	121.5	2.7	PHY Layer only
	Blackfin 600 MHz [2]	0.13	2.5	280	1.2	Programmable, only 556 Kbps
MPEG4	**Synchroscalar**	**0.13**	**32.32**	**47.24**	**0.7**	**QCIF @ 30 f/s**
QCIF	Amphion [1]	0.18	110k gates	15	UNK	Application-Specific Core, QCIF @ 15 f/s
	Philips [23]	0.18	20	30	1.8	ASIP, QCIF @ 15 f/s
	Blackfin 600 MHz [2]	0.13	2.5	280	1.2	Programmable, QCIF @ 15f/s
MPEG4	**Synchroscalar**	**0.13**	**31.74**	**370.03**	**1.1, 0.7**	**CIF @ 30 f/s**
CIF	Toshiba [3]	0.13	43	160	1.5	SOC, CIF @ 15 f/s

Table 3. Power Comparison of Synchroscalar with other platforms.

Application	Algorithm	No. of Tiles	Frequency (MHz)	Voltage (V)	Power (mW)	Power (mW) Single Voltage	% Power Savings Due to Multiple Voltages
DDC	Digital Mixer	8	120	0.8	76.29	191.83	60 %
	CIC Integrator	8	200	1.0	241.54	403.58	40%
	CIC Comb	2	40	0.7	18.86	18.86	66%
	CFIR	16	380	1.3	1071.22	1071.22	0%
	PFIR	16	370	1.3	1031.75	1031.75	0%
	TOTAL	50			2427.23	2717.24	11%
Stereo	SVD	1	500	1.5	114.27	114.27	0%
Vision	PFE	16	310	1.2	742.68	1151.55	36%
	TOTAL	17			857.40	1266.28	32%
802.11a	FFT	2	90	0.8	16.74	79.60	79%
	De-mod/De-Interleave	1	60	0.7	4.71	28.45	83%
	Viterbi ACS	16	540	1.7	3848.01	3848.01	0%
	Viterbi Traceback	1	330	1.2	61.07	83.22	27%
	TOTAL	20			3930.53	4039.28	3%
802.11a +	FFT	2	90	0.8	14.80	49.36	75%
AES	De-mod/De-Interleave	1	60	0.7	4.71	28.45	83%
	Viterbi ACS	16	540	1.7	3848.01	3848.01	0%
	Viterbi Traceback	1	330	1.2	61.07	83.22	27%
	AES	16	110	0.8	159.50	556.56	71%
	TOTAL	36			2443.68	2866.14	11%
MPEG4 30f/s	Motion Estimation	8	70	0.7	42.53	42.53	0%
QCIF	DCT, Quant, IQ, IDCT	2	60	0.7	4.71	4.71	0%
	TOTAL	10			47.24	47.24	0%
MPEG4, 30f/s	Motion Estimation	8	280	1.1	351.21	351.21	0%
CIF	DCT, Quant, IQ, IDCT	8	60	0.7	18.82	46.48	60%
	TOTAL	16			370.03	397.68	7%

Table 4. Power Results Summary for DDC, SV, 802.11a and MPEG4 on the Synchroscalar Processor

tions requirements and leakage current. The 802.11a parallel implementations shown in Figure 7 is good example of how diminishing returns from additional communications requirements prevent us from further parallelizing the 802.11a application efficiently. This communications overhead negatively impacts our power efficiency and is represented by the dark portions of each of the application's bars in Figure 7. Another source of diminishing returns from further parallelization is a supply voltage floor. While tiles could operate at supply voltages lower that 0.7 V, due to leakage and noise constraints, we chose 0.7 V as the minimum supply voltage supported. Therefore, by further parallelizing an algorithm that is already running at the minimum supply voltage would not yield further power savings, and would likely increase the power consumption due to leakage and added communications cost.

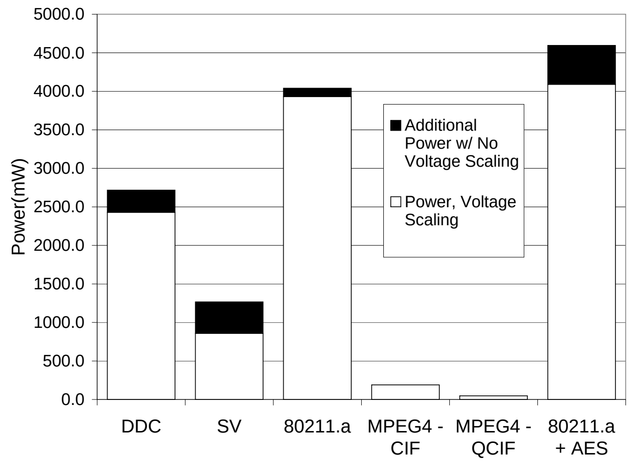

Figure 6. Power Consumption by Application

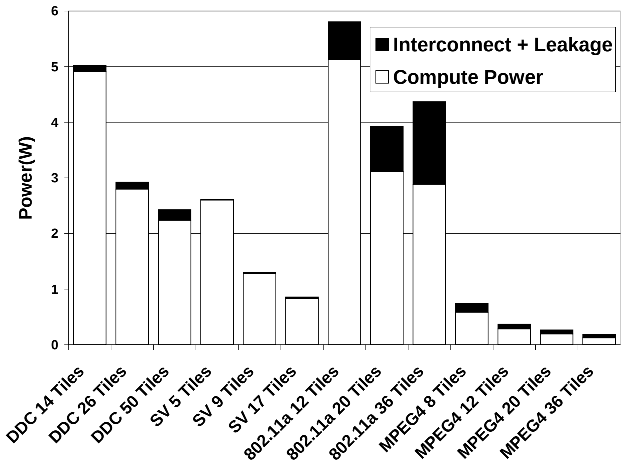

Figure 7. Power Consumption of Applications with varying parallelization

5.3. Effects of Interconnect

In Figure 8 we have mapped how the power-area efficiency of the Synchroscalar architecture scales for the Viterbi ACS with different sets of bus widths and different numbers of tiles. The Viterbi ACS is used here as the Viterbi Decoder has the most demanding communications requirements of any of the individual algorithms tested on the Synchroscalar architecture. The three curves on Figure 8 each represent a Viterbi ACS trellis being completed on 8, 16 and 32 tiles. Each of the curves are comprised of power results for a few different bus widths (32b, 64b, 128b, etc...). We can see from this figure that increasing the bus width from 128 to 256 bits significantly improves the power efficiency of Synchroscalar on the Viterbi Decoder for all three implementations. However, another such doubling of the bus width has a smaller reduction on our overall power consumption as the curves become less steep. This leads us to choose a 256 bit bus for Synchroscalar. While it would be possible to attain lower power consumptions for the Viterbi

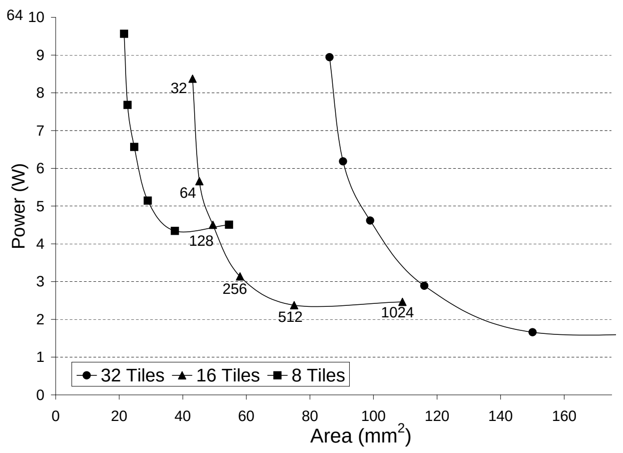

Figure 8. Power Consumption of Viterbi ACS with varying bus widths and parallelization

ACS it would come at a significant area cost. This trade-off is made in light of the fact that the other applications show little need for the increased bandwidth above a 256 bit bus.

5.4. Leakage Sensitivity Analysis

Since Synchroscalar trades spatial parallelism for temporal parallelism and the power dissipation due to leakage is proportional to the spatial parallelism, a careful analysis of leakage must be considered. Figures 9 and 10 show how different levels of parallelization of our four applications perform under varying levels of leakage currents. In the figures, the horizontal axis shows the leakage current per Synchroscalar tile, and the vertical axis shows the power consumption of the applications in mW. The lowest leakage current (1.5 mA/tile) corresponds to the leakage per tile as calculated in Section 4.4. The largest leakage current graphed corresponds to the leakage current if each tile used only low Vt transistors as published by Intel [41], which we believe represents the highest leakage current that we would consider in the development of Synchroscalar.

Of particular interest are the cross-over points between different levels of parallelization of an application, as in Figure 10 for MPEG4. Moving from eight to twelve tiles allows Synchroscalar to reduce the overall power consumption through frequency reduction and voltage scaling. These gains outweigh the leakage penalty and communications overhead. However, when moving from twelve to 36 tiles, the structure that has the best overall power consumption depends heavily on the leakage current. When tiles leak less than 14.8 mA (corresponding to 8.3 nA/transistor), the higher parallelized structure of 36 tiles is more efficient, but when tiles leak more than 14.8 mA, the twelve tile structure is more efficient.

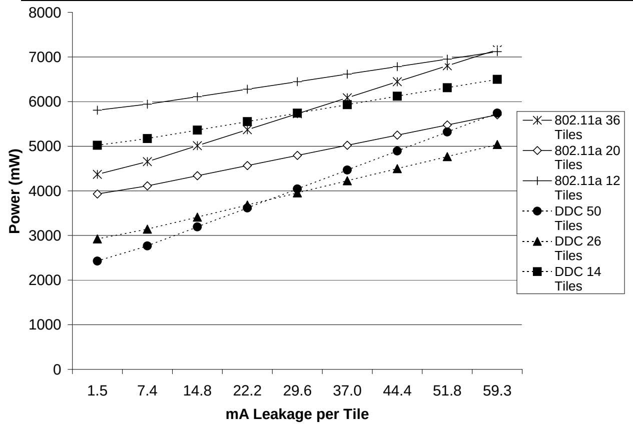

Figure 9. Leakage sensitivity for DDC, 802.11a

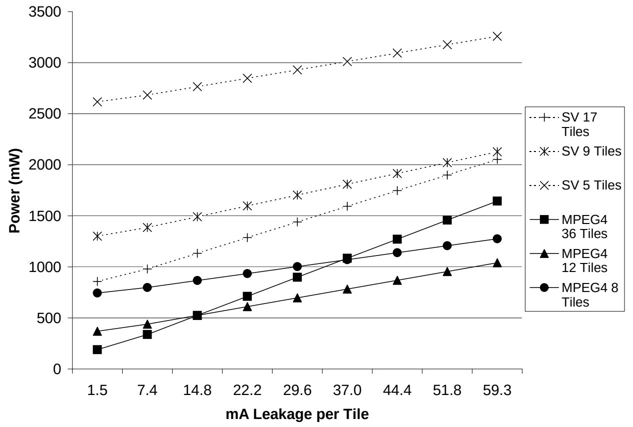

Figure 10. Leakage sensitivity for MPEG4, SV

5.5. Discussion

How much should one parallelize the applications? The factors that limit the amount of parallelization are the *voltage floor*, i.e. the minimum possible voltage that we could run a given tile at, the leakage current, and the structure of the application.

Additional parallel harware helps here to reduce power because we are scaling the voltage aggressively as well. Once all tiles are operating at voltage floor, parallelizing further is not advantageous, as further attempts for additional parallelization could increase the communiations overhead. It would the be the goa of a compilation tool for Synchroscalar to help parallelize applications so that they are running as close to the voltage floor as possible.

Our results are sensitive to the P_{tile} number that we derived in the methodology section. Since tile power is the dominant factor in the total Synchroscalar power, our power results are roughly linear with the P_{tile} Our qualitative results are valid for a large range of realistic values of tile power. For instance, let us compare the Synchroscalar power consumption with the power consumption of the Blackfin DSP which are both in 0.13μ technology. Using the 0.1 mW/MHz estimate of power per tile for Synchroscalar, we have shown that the DDC application runs at 2.43 W for 64e6 samples/second or 38.0 nW/sample. The Blackfin DSP can run at 280 mW for 113e3 samples/second at 600 MHz or 2478 nW/sample - a factor of 60 difference. So clearly, even if our estimate of P_{tile} is off by a factor of two, we are still demonstrating significant power savings.

6. Related Work

The challenges presented by next generation applications in terms of higher data rates, lower power requirements, shrinking time-to-market requirements, and lower cost has resulted in tremendous interest in embedded architectures and platforms for communication appliances in the past few years. Researchers have approached the problem from several different angles. The DSP architecture companies have proposed highly parallel VLIW machines coupled with hardware accelerators or co-processors for the computation-intensive functions. The TI OMAP [18] is a good example of this category of solutions. However, this is not power efficient. You would need very high clock frequencies to meet the throughput constraints for the applications considered in this paper.

The SCORE project at UC Berkeley [11] uses a FPGA-like fabric with specially tailored interconnect to exploit parallelism and improve power efficiency. The PLEIADES project at UC Berkeley [44] proposes an interconnection of a low power FPGA, datapath units, memory, and processors, optimized for different application domains. The PLEIADES researchers conclude that a hierarchical generalized mesh interconnect structure [43] is most appropriate for their architecture as it balances both the global and the local interconnect. Our results are in agreement with this conclusion in general but given that we are targeting streaming computations, we have greater emphasis on near-neighbor communication and have stayed away from a general mesh. Other reconfigurable machines, such as RAPID [12] and Piperench [33], illustrate interesting alternatives to our choice of tiles, and may be amenable to our coarse-grained voltage-frequency scaling techniques.

The adaptive SOC project at University of Massachusetts [22] advocates an array of processors connected by a statically scheduled communication fabric. They allow different processors to operate at different clock frequencies and demonstrate significant power savings on video processing benchmarks. The key differences between this work and Synchroscalar are in the structure and contents of the tiles and the memory architecture. In aSOC the tiles are hardwired functional blocks such as Viterbi decoder, FFT, DCT etc., while in Synchroscalar we assume programmable DSPs as the building blocks for the tiles. As a result, the memory architecture of the system is radically different, changing the data transfer and communication scheduling problem as well. Intel's tile based architecture [14] shares the same objectives as ours, but the interconnection network is very different. Also, the tiles in [14] are much coarser grained, which means their power consumption will likely be higher. The tile-based architecture from University of Texas [29] resembles Synchroscalar structurally but it is designed for wire-delay scalability, not power efficiency given a data rate constraint, which is the unique feature of our work. Synchroscalar's use of spatial rather than temporal flexibility is somewhat inspired by the MIT RAW project [39] [38], but our mechanisms for ASIC-like performance are significantly different. The Imagine [31] processor approaches a similar problem domain from a stream-oriented perspective. The parallelization strategies used by Imagine are complementary to the voltage scaling, data orchestration, and multi-rate optimization used in Synchroscalar. The Smart Memories [24] project is another tile-based architecture whose reconfigurable tiles would also be complementary to Synchroscalar mechanisms. While the SIMD components of our applications are dominant, some phases could benefit from other models of computation.

Recently, there has been a revival of interest in the globally asynchronous and locally synchronous (GALS) approach to processor implementation [4] including the use of multiple clock domains and multiple voltages [25] [34]. The key difference between GALS approach and the Synchroscalar approach is the restriction of using only rationally related frequencies between different columns. This avoids the use of asynchronous FIFOs with their synchronization overhead. So, Synchroscalar is similar to Numesh [35], rather than the GALS approach.

7. Conclusion

The design principles of Synchroscalar – high parallelism, efficient interconnect, low control overhead, and custom voltage/frequency domains – will lead to a new set of embedded architectures with efficiency approaching ASICs and with the programmability of DSPs. Our study has shown a promising proof-of-concept through hand optimization and code development. Future work will focus on a software tool chain to automate and optimize application parallelization and communication scheduling.

8. Acknowledgments

This work is supported by NSF ITR grants 0312837 and 0113418, and NSF CAREER and UC Davis Chancellor's fellowship awards to Fred Chong. Jedidiah Crandall's work was supported in part by a United States Department of Education Government Assistance in Areas of National Need (DOE-GAANN) grant P200A010306. Diana Franklin's faculty position is funded by a Forbes Endowment. We would also like to thank Rajeevan Amirtharajah, Bevan Baas, Dean Copsey and Matthew Farrens at Univeristy of California at Davis, Mark Oskin at Univeristy of Washington and Timothy Sherwood at Univeristy of California at Santa Barbara.

References

[1] Amphion. *Amphion CS6701 Hybrid MPEG-4 Video Encoder*. http://www.amphion.com/cs6701.html.

[2] Analog Devices Press Release: Blackfi n. http://www. electronicstalk.com/ news/anc/anc199.html, March 2003.

[3] H. Arakida, M. Takahashi, Y. T. 1, T. Nishikawa, , H. Yamamoto, T. Fujiyoshi, Y. Kitasho, Y. Ueda, M. Watanabe, T. Fujita, T. Terazawa, K. Ohmori, M. Koana, H. Nakamura, E. Watanabe, H. Ando, T. Aikawa, and T. Furuyama. A 160mW, 80nA standby, MPEG-4 audiovisual LSI with 160mb embedded DRAM and a 50GOPS adaptive post fi lter. In *International Solid-State Circuits Conference, Digest of Technical Papers*, Feb. 2003.

[4] B. M. Baas. A parallel programmable energy-effi cient architecture for computationally intensive DSP systems. In *Conference Record of the Thirty-Seventh Asilomar Conference on Signals, Systems, and Computers*, nov 2003.

[5] A. Benedetti. Personal communication, 2003.

[6] S. Bhattacharya, P. Murthy, and E. Lee. Software synthesis from datafbw graphs, 1996.

[7] S. Bhattacharya, P. Murthy, and E. Lee. Synthesis of embedded software from synchronous datafbw specifi cations. *Journal of VLSI Signal Processing*, (21):151–166, June 1999.

[8] R. Brodersen. Low voltage design for portable systems. In *International Solid State Circuits Conference*, Feb. 2002.

[9] J. Buck et al. Ptolemy: A framework for simulating and prototyping heterogenous systems. *Int. Journal in Computer Simulation*, 4(2), 1994.

[10] C. Tomasi and T. Kanade. Detection and tracking of point features, 1991.

[11] E. Caspi, M. Chu, R. Huang, J. Yeh, J. Wawrzynek, and A. DeHon. Stream computations organized for reconfi gurable execution (SCORE). In *FPL*, pages 605–614, 2000.

[12] P. Cronquist, D.C.and Franklin, S. Berg, and C. Ebeling. Specifying and compiling applications for RaPiD. In K. L. Pocek and J. Arnold, editors, *IEEE Symposium on FPGAs for Custom Computing Machines*, pages 116–125, Los Alamitos, CA, 1998. IEEE Computer Society Press.

[13] David Su and Masoud Zargari and Patrick Yue and Shahriar Rabii and David Weber and Brian Kaczynski and Srenik Mehta and Kalwant Singh and Sunetra Mendis and Bruce Wooley. A 5ghz cmos transceiver for ieee 802.11a wireless lan. In *International Solid State Circuits Conference*, 2002.

[14] E. Tsui and K. Ganapathy. A new distributed dsp architecture based on the intel ixs for wireless client and infrastructure. In *HOT CHIPS 14*, Aug. 2002.

[15] S. Gupta, S. Keckler, and D. Burger. Technology independent area and delay estimates for microprocessor building blocks. In *Technical Report TR2000-05, Department of Computer Science, University of Texas*, 2000.

[16] R. Ho, K. Mai, and M. Horowitz. The future of wires. In *Proceedings of the IEEE*, volume 89, pages 490–504, April 2001.

[17] C. Hu. Berkeley predictive technology model. http://www-device.eecs.berkeley.edu/ ptm/introduction.html.

[18] T. Instruments. Omap backgrounder. http://www.ti.com/ corp/docs/press/backgrounder/omap.shtml.

[19] Intel xeon processor 2.8 ghz datasheet. http://www. intel.com/design/xeon/datashts/298642.htm, March 2003.

[20] R. Kolagotla, J. Fridman, B. Aldrich, M. Hoffman, W. Anderson, M. Allen, D. Witt, R. Dunton, and L. Booth. High Performance Dual-MAC DSP Architecture. *IEEE Signal Processing Magazine*, July 2002.

[21] E. A. Lee and D. G. Messerschmitt. Static scheduling of synchronous dataflow programs for digital signal processing. *IEEE Transactions on Computers*, C-36(1), January 1999.

[22] J. Liang, S. Swaminathan, and R. Tessier. aSOC: A scalable, single-chip communications architecture. In *IEEE PACT*, pages 37–46, 2000.

[23] R. P. Llopis, R. Sethuraman, C. A. P. H. Peters, S.Maul, and M. Oosterhuis. A low-cost and low-power multi-standard video encoder. In *First IEEE/ACM/IFIP International Conference on Hardware/Software Codesign and System Synthesis*, 2003.

[24] K. Mai, T. Paaske, N. Jayasena, R. Ho, W. J. Dally, and M. Horowitz. Smart memories: A modular reconfigurable architecture. In *27th Annual International Symposium on Computer Architecture (27th ISCA-2000) Computer Architecture News*, Vancouver, British Columbia, Canada, June 2000. ACM SIGARCH / IEEE. Published as 27th Annual International Symposium on Computer Architecture (27th ISCA-2000) Computer Architecture News, volume 28.

[25] D. Marculescu and A. Iyer. Power and performance evaluation of globally asynchronous locally synchronous processors. In D. DeGroot, editor, *Proceedings of the 29th International Symposium on Computer Architecture (ISCA-02)*, volume 30, 2 of *Computer Architectuer News*, pages 158–170, New York, May 25–29 2002. ACM Press.

[26] L. Matthies, B. Chen, and J. Petrescu. Stereo vision, residual image processing and mars rover localization, 1997.

[27] I. Mavroidis. A low power 200 MHz multiported register file for the vector IRAM chip. In *Report No. UCB/CSD-01-1145, MS Thesis, University of California, Berkeley*, 2001.

[28] M. Mori, B. Amrutur, K. Mai, M. Horowitz, I. Fukushi, T. Izawa, and S. Mitarai. A 1V 0.9mW at 100 MHz 2kx16b SRAM utilizing a half-swing pulsed-decoder and write-bus architecture in 0.25 μm dual-Vt CMOS. In *IEEE International Solid-State Conference, Digest of Technical Papers*, 1998.

[29] R. Nagarajan, K. Sankaralingam, D. Burger, and S. W. Keckler. A design space evaluation of grid processor architectures. In *Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture*, pages 40–51. IEEE Computer Society, 2001.

[30] M. Pilu. A direct method for stereo correspondence based on singular value decomposition, 1997.

[31] S. Rixner, W. J. Dally, U. J. Kapasi, B. Khailany, A. Lopez-Lagunas, P. R. Mattson, and J. D. Owens. A bandwidth-efficient architecture for media processing. In *International Symposium on Microarchitecture*, pages 3–13, 1998.

[32] P. Ryan, T. Arivoli, L. de Souza, G. Foyster, R. Keaney, T. McDermott, A. Moini, S. Al-Sarawi, J. O'Sullivan, U. Parker, G. Smith, N. Weste, , and G. Zyner. A single chip PHY COFDM modem for IEEE 802.11a with integrated ADCs and DACs. In *International Solid State Circuits Conference*, 2002.

[33] H. Schmit et al. Pipeline reconfigurable FPGA. *Journal of VLSI Signal Processing Systems for Signal, Image and Video Technology*, 24(2):129–146, March 2000.

[34] G. Semeraro et al. Energy-efficient processor design using multiple clock domains with dynamic voltage and frequency scaling. In *HPCA*, pages 29–42, 2002.

[35] D. Shoemaker, F. Honore, C. Metcalf, and S. Ward. Numesh: An architecture optimized for scheduled communication. *Journal of Supercomputing*, 10(3), 1996.

[36] W. Stechele. Algorithmic complexity, motion estimation and a vlsi architecture for mpeg-4 core profile video codecs. In *International Symposium on VLSI Technology, Systems and Applications*, 2001.

[37] M. Y. T. Kumura, M. Ikekawa and I. Kuroda. VLIW DSP for Mobile Applications. *IEEE Signal Processing Magazine*, July 2002.

[38] M. Taylor, J. Kim, J. Miller, D. Wentzla, F. Ghodrat, B. Greenwald, H. Ho, m Lee, P. Johnson, W. Lee, A. Ma, A. Saraf, M. Seneski, N. Shnidman, V. Frank, S. Amarasinghe, and A. Agarwal. The raw microprocessor: A computational fabric for software circuits and general purpose programs, 2002.

[39] M. B. Taylor et al. The Raw microprocessor: A computational fabric for software circuits and general-purpose programs. *IEEE Micro*, 22(2):25–35, Mar./Apr. 2002.

[40] Texas Instruments GC4014 quad receiver chip datasheet. http://www-s.ti.com/sc/psheets/ slws132/slws132.pdf, April 1999.

[41] S. Thompson, M. Alavi, M. Hussein, P. Jacob, C. Kenyon, P. Moon, M. Prince, S. Sivakumar, S. Tyagi, and M. Bohr. 130nm logic technology featuring 60nm transistors, low-k dielectrics, and cu interconnects. *Intel Technology Journal*, 6(2):5–13, May 2002.

[42] W. Eberle and V. Derudder and L. Van der Perre and G. Vanwijnsberghe and M. Vergara and L. Deneire and B. Gyselinckx and M. Engels and I. Bolsens and and H. De Man. A digital 72Mb/s 64-QAM OFDM transceiver for 5GHz wireless LAN in 0.18um CMOS. In *International Solid State Circuits Conference*, 2002.

[43] H. Zhang, M. W. V. George, and J. Rabaey. Interconnect Architecture Exploration for Low Energy Reconfigurable Single-Chip DSP. In *Proceedings of the Workshop on VLSI, Orlando, Florida*, April 1999.

[44] H. Zhang, V. Prabhu, V. George, M. Benes, A. Abnous, and J. Rabaey. A 1-V heterogenous reconfigurable DSP IC for wireless baseband digital signal processing. *IEEE Journal of Solid State Circuits*, 35:1697–1704, November 2000.