

Energy and Performance Models for Clocked and Asynchronous Communication

Kenneth S. Stevens
Strategic CAD Labs, Intel Corporation, Hillsboro, OR

Abstract

Parameterized first-order models for throughput, energy, and bandwidth are presented in this paper. Models are developed for many common pipeline methodologies, including clocked flopped, clocked time-borrowing latch protocols, asynchronous two-cycle, four-cycle, delay-insensitive, and source synchronous. The paper focuses on communication costs which have the potential to throttle design performance as scaling continues. The models can also be applied to logic. The equations share common parameters to allow apples-to-apples comparisons against different design targets and pipeline methodologies. By applying the parameters to various design targets, one can determine when unlocked communication is superior at the physical level to clocked communication in terms of energy for a given bandwidth. Comparisons between protocols at fixed targets also allow designers to understand tradeoffs between implementations that have a varying degree of timing assumptions and design requirements.

1. Introduction

A number of effects are impinging on the way we do design. A fundamental change is in the requirement of adding pipeline stages in the interconnect. This increased complexity is due to a number of factors including physical scaling as well as the desire to increase performance. This communication pipelining results in some circuit and architecture advantages and overheads. Various communication methodologies have differing local and architectural performance implications and CAD requirements. This work models pipelines using various protocols, including clocked and asynchronous, and makes a first-order comparison of energy and performance of these models. These models show that asynchronous communication can be comparable or superior to clocked communication, even under worst case conditions.

Global synchronous design requires the expenditure of a large design effort to create a low skew clock. While this can result in many efficiencies, including no synchroniza-

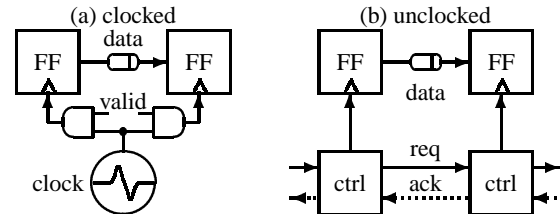


Figure 1. Clocked and unlocked topologies

tion overhead and uniform performance targets, there are also significant drawbacks. A single global frequency may not be optimal for CPU architectures because each module has different optimal power/performance points. Further, a global frequency requires repipelining every module as frequencies change. If efficient unlocked communication can be designed, two significant benefits arise. First, each module can be designed for its best frequency and power. Second, the ability to interconnect components with different frequencies will be vastly enhanced resulting in higher design reuse, faster time to market, and easier ability to customize designs.

The goal of this work is to study communication methodologies to determine if and when unlocked communication can be competitive at the physical level with clocked styles.

2. Handshaking and Background

Figure 1 shows the basic interconnection for clocked and unlocked design methodologies. In clocked design, data is normally transmitted every clock cycle between the latches and flops. Data transmission in unlocked domains is based on handshake protocols, allowing transactions to be dynamic based on data validity. Delay insensitive (DI) protocols encode data validity with each bit, whereas bundled protocols employ the independent request (req) signal. When data is not available, the communication links remain idle. This is emulated in a clocked design by applying clock gating, which passes a data valid bit with similar functionality to the req signal indicating the validity of the data, allowing one to disable or “gate” the clock to the flops. The backward acknowledge (ack) signal in the asynchronous domain

restricts the max cycle time in such a way that min-delay failures cannot occur.

Clocking methodologies and bundled data protocols place restrictions on how and when the data can change relative to the clock or handshake signals [6]. The data must have sufficient time to become valid before the clock or request signal enables its usage. The “ack” signal is used in asynchronous protocols to indicate the data has been received, and that new data can be transmitted. The protocols are maintained in a clocked system with setup and hold requirements. Most unclocked protocols have backwards handshakes and return-to-zero (RZ) phases. Two-cycle (or NRZ) protocols halve the number of transitions per handshake but usually require more logic overhead and delay. Timing assumptions can be made using pulsed and source synchronous protocols to entirely remove the backward handshake path [8].

When controlling local data paths, the communication delays for control signals are small and the handshaking overhead can largely occur concurrently with the data function, hiding the overhead. However, for communication, the req and ack control signals are exposed to the same power and delay costs as the data because they must also propagate from sender to receiver. Thus propagation of the control lines is an expensive operation for most asynchronous protocols in both power and delay when compared to clock methodologies where the distribution costs can be shared amongst many logic blocks. This makes data communication the most *inefficient* application for asynchronous protocols and creates a worst-case comparison between clocked and unclocked implementations.

Wire sizing and shielding can be used to somewhat mitigate control wire overhead. Nonetheless, the handshaking significantly limits the throughput of the four-phase protocols. If the delays of the data and control wires are roughly the same, then a four-cycle protocol can at best transmit data at a rate one-fourth the propagation time between the latches.

The remainder of this paper reports the comparison of common protocols against each other for a configuration typical of a microprocessor bus. Models are then developed that were used in the comparison for energy and delay in a hierarchical manner. These models include most first order effects such as clock gating, coupling, process variation, voltage variations, data and bus activity factors, etc.

3. Comparison

Seven representative protocols are modeled in this paper. This includes two clocked protocols, two DI protocols, two bundled protocols, and one pulse protocol. The clocked protocols include the flopped design (clk_flop) and latched with time borrowing (clk_latch). The dual-rail (2-rail) and

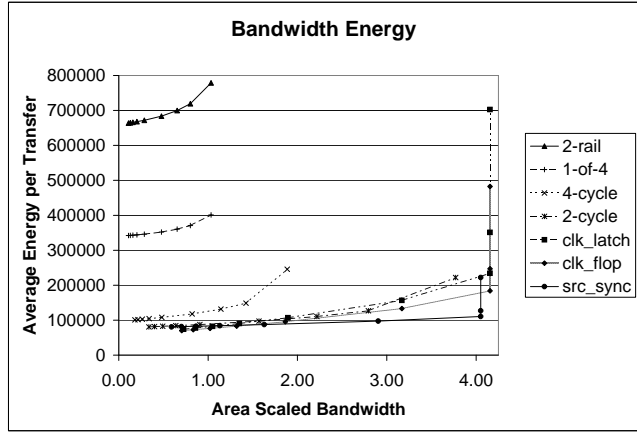


Figure 2. Complete Energy/Bandwidth Graph

one-of-four (1-of-4) DI protocols are modeled, as well as NRZ two-cycle (2-cycle) and RZ four-cycle (4-cycle) bundled data protocols. Finally the source synchronous protocol (src_sync) is modeled, which sends the request as a pulse along with the bundled data. The source synchronous protocol has no acknowledgment.

Other protocols were modeled but not included in the paper for clarity, such as pulsed clock protocols and asynchronous pulse-mode, one-hot, and the PC2/2 protocol [7]. Other styles, such as those using n-of-m codes [1][5], can be modeled in the same manner as the other protocols in this paper. Most of the protocols will be similar to, or bounded by, the models presented in this paper in terms of throughput, energy, and bandwidth.

Figures 2 and 3 show the results of these models applied to a long $10,000\mu$ bus with a critical repeater distance of 600μ . The distances and parameters are typical of what might be found on a microprocessor fabricated in a 65nm process. A very long bus was chosen to allow a wide range of pipelining and frequencies to be graphed. The y-axis shows the average energy per transaction, measured in relation to the energy of driving a minimum sized inverter. The x-axis shows the effective bandwidth in terms of the number of concurrent transactions that can be sent down this path. The bandwidth is area scaled based on the number of wires used for control and data paths.

Each tick mark on any protocol line in the graphs indicates a particular frequency or pipeline granularity. For the parameters used in this paper, the unpipelined $10,000\mu$ wire will contain 16.67 repeaters. The far left point of each protocol is the condition where all of the repeaters are inverters. Pipelining is increased moving right along each protocol by replacing some of the inverters with flops or latches. As more of the inverters of the communication path are replaced with pipeline latches, the frequency and bandwidth increases across the x-axis due to increased pipelining. The

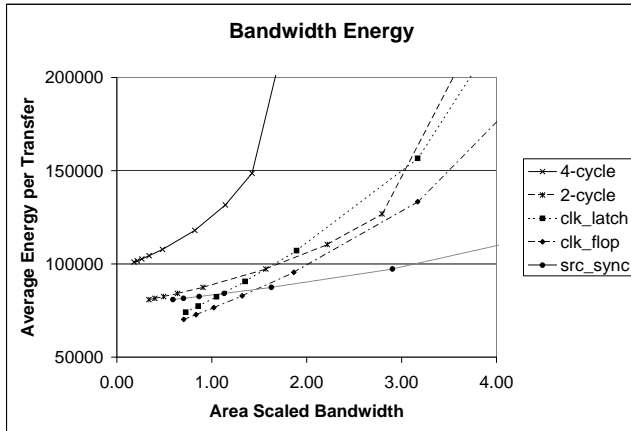


Figure 3. Efficient Protocols

rightmost tick is where all of the inverters have been replaced with flopped repeaters or pipeline control, achieving the maximum bandwidth for the protocol.

Achieving the target bandwidth can come at different frequencies (or pipeline granularity) based on the protocol being used. This can be seen by determining the pipelining for every protocol that achieves a particular bandwidth target in the graphs. In general, these results show that the asynchronous protocols require higher pipelining to achieve the same bandwidth as the clocked protocols.

Increased bandwidth comes at the cost of increased energy per transaction for a fixed bus width. Fine-grain pipelining, or the right-most point on each protocol, is the most power-hungry operational mode for any protocol. For optimal power and performance, the bandwidth should be matched by appropriate pipeline granularity or frequency. The graphs also show that for the clocked and source synchronous designs, a wall exists beyond which increasing the frequency achieves no further performance gains.

The protocols with acknowledgment handshaking show a significantly reduced bandwidth range when compared with the clocked and source synchronous protocols. This is due to the delay overheads in propagating request and acknowledgment signals across long wires.

Figure 3 zooms in on the efficient protocols. Clocked protocols exhibit the best power and bandwidth values for coarse grained pipelining having a low clock frequency. The source-synchronous protocol is the best for highly pipelined designs. The bulk of the energy for all protocols is dominated by the wires. However, as the data is increasingly pipelined, the average energy per transaction increases. The slopes of the four-cycle asynchronous protocols that have backward handshaking (ack signaling) are steeper, indicating a larger energy penalty for increased bandwidth. Clocked flop designs are more energy efficient than latch designs, even though the latch designs allow a

marginally higher throughput for any fixed latch/flop placement on the communication path. Note that asynchronous designs adapt to the current fabrication and environment conditions. Since the graphs show the worst case conditions, a vast majority of the asynchronous chips will operate at a significantly improved frequency, reducing the slopes on these protocols which makes them more competitive to clocked design.

The dual-rail and 1-of-4 DI protocols exhibit significantly higher average energy per transaction and the lowest bandwidth ranges. The bandwidth limitations are largely due to the inefficient use of wires. Each bit requires two wires in these protocols, effectively halving the bandwidth. The high energy consumption of these protocols can mostly be attributed to the high activity factors that result from the data encodings. However, because these protocols are delay insensitive the CAD requirements are greatly reduced, allowing quicker time to market and higher robustness to operational parameters. Hence they may still be good choices depending on the design requirements.

4. Approach

The remainder of this paper details the method used to generate the models and results using the following steps:

1. Determine which set of parameters characterize first order variations in delays.
2. Model the parameterized latencies of latches and flops on both data and clock pins.
3. Characterize the energy of latches and flops.
4. For a selected set of protocols, create parameterized first order models for throughput based on the latch, flop and control logic overheads.
5. Create parameterized energy models for the protocols.
6. Create bandwidth models based on wire efficiency.

5. Models

5.1. Parameters

The timing of all circuit elements we fabricate will be faster or slower than the scalar value of an “ideal” element due to process variations, capacitive coupling, and other effects. Variation from ideal devices and wires is considered an overhead in this work, whether it manifests itself as clock skew or device and wire delay variation.

Tables 1 and 2 show the parameters and their associated values that are used in this paper. The values are based

Name	Equation	Value
V_c	Cross coupling variation of wire	0.3
V_p	Process variation	0.16
V_{pw}	Pulse width variation	0.2
V_{vt}	Voltage & temperature variation	0.12
V_{cad}	Margin for CAD (PV Accuracy)	0.05
D_i	Ideal stage delay (as FO4)	30 ... 1
T_{cq+}	Delay from clock to output	1.5
T_{cq-}	$T_{cq+}(1 - V_p)$	1.26
T_{su}	Setup time	0.25
T_{skj}	Clock skew and jitter	1
T_{sk}	Clock skew	1
T_{skp}	Time borrow phase skew	0.5
T_h	Hold time	0.25
T_{dqf}	Flop data-to-output delay	3
T_{dql}	Latch data-to-output delay	1.5
T_{pw}	Minimum pulse width	3.5
T_{pw+}	$T_{pw}(1 + \frac{V_{pw}}{2})$	3.85
T_{fsm+}	Control latency	2.5
T_{fsm-}	$T_{fsm+}(1 - V_p)$	2.1
T_{dr}	Domino reset & propagation	0
T_{cad}	$D_i V_{cad}$	1.5–0.05
T_l	Part of D_i attributed to logic	0 ... 0
T_{lc+}	$T_l(1 + \frac{V_p}{2} + V_{vt})$	0
T_{l+}	$T_l(1 + \frac{V_p}{2})$	0
T_c	$D_i - T_l$	30–1
T_{cc+}	$T_c(1 + \frac{V_c}{2} + \frac{2V_{vt}}{1+D_c})$	37.1–1.24
T_{c+}	$T_c(1 + \frac{V_c}{2})$	34.5–1.15
T_{csh+}	$T_c(1 + \frac{V_c}{4})$	32.3–1.08

Table 1. Parameter variables and derivatives.

on the design constraints and process technology, loosely based on a 65nm process for this work. Changing the value of any parameter may improve or degrade the results.

The parameters in the top section of Table 1 model variations from ideal delays that occur on on wires and devices. First-order effects are modeled, including coupling, process variation, voltage and temperature variations (V_c , V_p , and V_{vt} respectively). A larger margin is taken for pulse width variation V_{pw} over normal process variation. Other variations that occur in devices, such as multiple input switching and drafting are modeled as a margin for CAD tool inaccuracies (V_{cad}). Without this term, one assumes that the CAD can exactly model best and worst case delay conditions of fabricated silicon.

Values in the bottom section of Table 1 are scalar delays. These delays are all relative to the fanout of 4 (FO4) of a typical inverter in the selected technology. Clock skew is assigned a scalar of 1 FO4 delay, clock to output time T_{cq+} is 1.5, and the setup time T_{su} is 1, as shown in the table. The

Name	Equation	Value
$R_{g/t}$	Ratio of gate cap to total cap	0.2
A_w	Activity factor of bus wires	0.18
A_b	Activity factor of bus	0.05
O_{wd}	Control wire delay optimization	0.85
O_{wa}	Control wire area optimization	1.85
R_{ctl}	Control energy	0.5
X_c	Critical wire distance	600 μ
L_g	repeater size	120
L_p	$L_g(\frac{1}{R_{g/t}} - 1)$	480
W_b	Width of bus	32
D_c	Delay across repeated wire	1.8

Table 2. Parameter variables and derivatives.

ideal stage delay, D_i , is varied from 30 to 1 to allow evaluating pipelines ranging from coarse to very fine grain. This is the amount of ideal logic delay between latches of flops in the design, and it determines the pipelining or frequency of the communication link.

The target delay of any pipeline stage can be distributed as either functional logic delays T_l or communication delays T_c , which includes both repeater and wire delay. Therefore, ideal delays $D_i = T_l + T_c$. For the designs in this paper $D_i = T_c$. However, we have developed models that include function logic delays which significantly change the throughput results for some protocols. Variations are applied differently to logic and communication.

Delays with a maximum variant can append a '+' symbol, and min-delay values append a '-'. For example, T_{c+} is the max communication delay and T_{cq-} is the minimum clock to data output delay of a flop or latch.

T_{lc+} and T_{cc+} are the maximum logic delay and communication delays for clocked systems. Clocked protocols take first droop effects into account since the fixed frequency could otherwise cause the circuits to fail under the slower operation induced by the voltage droop. The asynchronous protocols will adapt to the operating conditions. Communication delay for the shielded links T_{csh+} of the DI protocols is smaller than for the non-shielded bundled data paths.

Finer specification of the parameters and their variation values could be made. For instance the communication delay is comprised of both a repeater device and resistive wire component. A single value V_c is used here to model process variation that applies to the device as well as coupling variation that only applies to the wire. A single delay value T_{fsm+} for the asynchronous control logic is also used. Some protocols, such as the 2-cycle NRZ protocols generally require more complicated control with larger latencies. This has not been modeled in the graphs in this paper. These variables could be split to give more accurate

Name	equation	description
E_{clkf}	$4 + i_2 + \max(\frac{4}{3}, \frac{L_g}{3.75}) + \max(\frac{4}{3}, \frac{L_g}{9.375}) + \max(1, \frac{L_1+i_2}{4})$	FF clock energy
l_1	$2 + \max(\frac{2}{3}, \frac{L_g}{18.75}) + \max(\frac{2}{3}, \frac{L_g+1}{7.5})$	internal flop load
i_2	$\max(1, \frac{\frac{2}{3} + \max(\frac{2}{3}, \frac{L_g}{18.75})}{4})$	internal flop inverter load
E_{datf}	$L_g + L_p + 5 + \max(\frac{2}{3}, \frac{L_g}{18.75}) + \max(2, \frac{L_g}{2.5}) + \max(1, \frac{L_g+1}{7.5})$	FF Data Energy
$E_{clk l}$	$2 + \max(\frac{4}{3}, \frac{L_g+1}{3.75}) + \max(1, \frac{\frac{2}{3} + \max(\frac{2}{3}, \frac{L_g+1}{7.5})}{4})$	Latch clock energy
$E_{dat l}$	$L_g + L_p + 3 + \max(1, \frac{L_g+1}{7.5})$	Latch data energy
E_{inv}	$L_g + L_p$	repeater and wire energy
E_{gf}	$\frac{1}{A_b} \max(\frac{4}{3}, \frac{E_{clkf}}{12}) + \max(1, \frac{E_{clkf}}{4})$	Flop clock gate energy
E_{gl}	$\frac{1}{A_b} \max(\frac{4}{3}, \frac{E_{clk l}}{12}) + \max(1, \frac{E_{clk l}}{4})$	Latch clock gate energy

Table 3. Energy variables and equations. Values in terms of input load of a minimum sized inverter.

values for each protocol.

The parameters in Table 2 include the critical wire distance of our process. This is the distance between optimally placed repeaters. Thus our $10,000\mu$ wire optimally requires 16.67 repeaters. The delay across each repeater stage D_c is approximately 1.8 times a nominal FO4 delay. The size of the repeater L_g in this segment is 120 times larger than a minimal inverter. The energy required to drive the wire can be calculated from the ratio of gate capacitance to total capacitance $R_{g/t}$. The parasitic load L_p of the repeater and the wire will be four times the gate load. Assuming that the parasitic capacitance of the gate is approximately equal to the gate cap, which seems to be relatively process independent, the parasitic wire capacitance for this repeater segment will be approximately three times the gate capacitance. The parameter A_b is activity factor of the the bus, or the percentage of clock cycles during which data is transmitted across the bus. Each flop is considered to be gated during idle cycles in the clocked protocols. A_w is the activity factor of data on the bus, or the probability that any bit will switch values for an average bus transaction. The parameter R_{ctl} is used to calculate the average energy for the control logic for the asynchronous protocols. It is typically multiplied by L_g . The sizing and spacing of the control wires in the asynchronous protocols may be optimized differently than the data wires. The parameters O_{wd} allow a reduced control wire delay for a larger area O_{wa} .

5.2. Latch and Flop Energy

The sizes of devices in this paper are all expressed using logical effort, which is a method of calculating the sizing and cost of a circuit topology that implements a logic function [9]. The energy numbers are all related to the baseline of the gate load of a minimal inverter which is assigned a value of 1.

The latch shown in Figure 4 is used as the sequential for

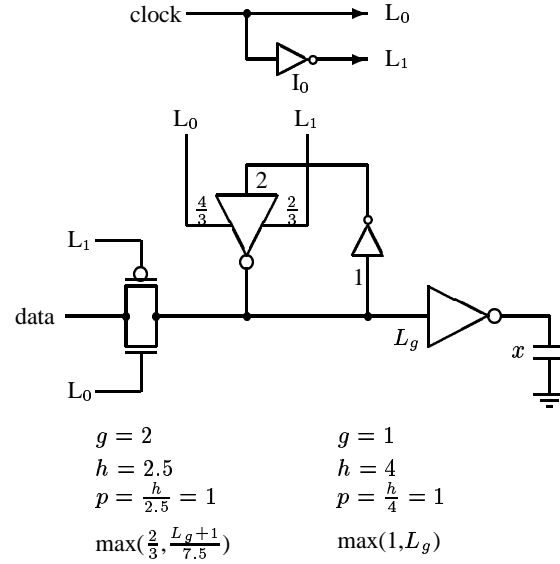


Figure 4. Sizing of the Latch (and half stage of a flop) for energy calculations.

this work [10]. The clock is directly wired to nodes L_0 and the inverter I_0 . The latch drives a parasitic load x . The keeper is a minimum sized tristate driver. The size of the inverter and pass gate depend on the load. The equations under these gates are their logical effort values, where g is the effort of the gate, h is the electrical effort (or gain), and p is the parasitic load of the gate. These equations return the size of the devices.

The resistance of long communication wires shields the total load x from the driving inverter. Sizing the driver solely based on the load is inefficient. Therefore the sizes of drivers in this paper refer to critical communication distance and a fixed driver size L_g . These are based on the parameters of the metal layer being used for communication. Hence, the performance and energy numbers are based on

Name		equation	description
C_{ff+}	\geq	$\max(2T_{pw}, (T_{lc+} + T_{cc+} + T_{su} + T_{skj} + T_{cq+} + T_{cad}))$	Clk_flop throughput
C_{l+}	\geq	$\max(2T_{pw}, T_{lc+} + T_{cc+} + 2T_{dql})$	Clk_latch throughput
A_{di+}	\leq	$4T_{fsm+} + 2T_{csh+} + 2O_{wd}T_{c+} + T_{l+} + T_{dr}$	2-rail and 1-of-4
T_{ps}	\leq	$(T_{fsm+} + T_{cq+} + T_{l+} + T_{c+} + T_{su}) - (T_{fsm+} + T_{c-} + T_{fsm-})$	Handshake path separation
T_{cdel}	\leq	$(1 + \frac{V_p}{2})\max(0, T_{ps}) - (1 - \frac{V_p}{2})\max(0, -T_{ps}) + T_{cad}$	Robust delay element size
T_{pdel}	\leq	$(1 + \frac{V_{pw}}{2})\max(0, T_{ps}) - (1 - \frac{V_{pw}}{2})\max(0, -T_{ps}) + T_{cad}$	Pulse delay element size
A_{2+}	\leq	$2T_{fsm+} + T_{c+} + T_{cdel} + O_{wd}T_{c+}$	2-cycle bundled throughput
A_{4+}	\leq	$4T_{fsm+} + 2T_{c+} + T_{cdel} + 2O_{wd}T_{c+}$	4-cycle bundled throughput
SS_+	\leq	$\max(2T_{pw}, (T_{cq+} + T_{l+} + T_{c+} + T_{su}), (T_{c+} + T_{pdel} + T_{fsm+}))$	Src_sync throughput

Table 4. Throughput equations. Cycle time measured in FO4 delays.

the driver size L_g rather than the load capacitance x .

The total load that switches with the clock is the sum of L_0 , L_1 , and the load of the inverter I_0 , where $L_0 = \max(\frac{2}{3}, \frac{L_g+1}{7.5}) + \frac{4}{3}$, $L_1 = \max(\frac{2}{3}, \frac{L_g+1}{7.5}) + \frac{2}{3}$, and $I_0 = \max(1, \frac{L_1}{4})$. The data load can be expressed in terms of minimum size inverter loads as $L_p + L_g + 3 + \max(1, \frac{L_g+1}{7.5})$, where L_p is the parasitic load. The last term in this equation models the parasitic of source and drain of the the pass gate input of the latch, assuming its value is $\frac{1}{3}$ the gain of the stage. The self-loading of I_0 and the tristate driver are not included.

Flip-flops were designed by combining two of these stages together with an efficient inverter tree for clocking. The equations are similar, with cascaded sizes based on output load. The data-to-output delay for a flop T_{dqf} is 3 and for a latch T_{dql} is 1.5. T_{cq} for both fops and latches is 1.5.

The equations that calculate the clock and datapath energy for latches and fops based on Figure 4 are shown in Table 3. Energy results are based on these equations parameterized by the size of the driving inverter. For example, the energy to drive the critical distance of a communication link through a repeater E_{inv} is the gate load L_g plus the parasitic load of the link L_p . The latch data energy E_{datl} is derived as the repeater load plus overhead for the pass gate and the parasitics of the keeper logic. The average energy per transaction for clock gating of a flop E_{gf} is calculated as a single transition on the clock driver, which is one-fourth the clock load of the flop, plus energy into the gating NAND for the average number of idle bus cycles per transaction.

5.3. Protocols

Equations to calculate throughput are derived for each communication methodology. The values derived are expressed in terms of FO4 delay through each pipe stage. The *ideal* delay represents the optimal performance achievable for any protocol using ideal devices and no overheads due

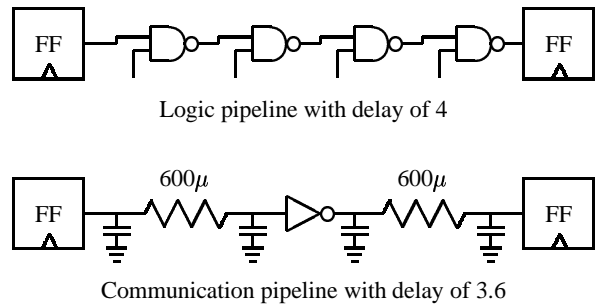


Figure 5. Logic and communication pipelines with ideal delays

to variations, buffering, or clock skew. The actual delay including timing uncertainties and flop delays for each protocol is measured against this idealistic value to determine the overhead.

Table 4 shows the throughput models. Each equation was independently derived and is highly parameterized for delays and variations. The clocked values resemble the equations in [2]. Setup, skew, and flop delays create the overhead for clocking. The DI protocols are also relatively simple because they sequentially execute each logic and communication operation. The other asynchronous models are much more complicated because their handshaking protocols result in races between the data and the req handshake signal. These protocols require some calculation to ensure there is sufficient time between the control and data paths. This usually requires a delay element. The parameter T_{ps} is the worst case path separation in terms of FO4 delays between the control and data paths, assuming valid data is at the input of a latch. T_{cdel} is the number of gate delays that must be added to the control path to guarantee the data path arrives first under all skews and variations, including variation in the delay element itself. The worst-case values must be taken assuming that as pipe stages are composed, it

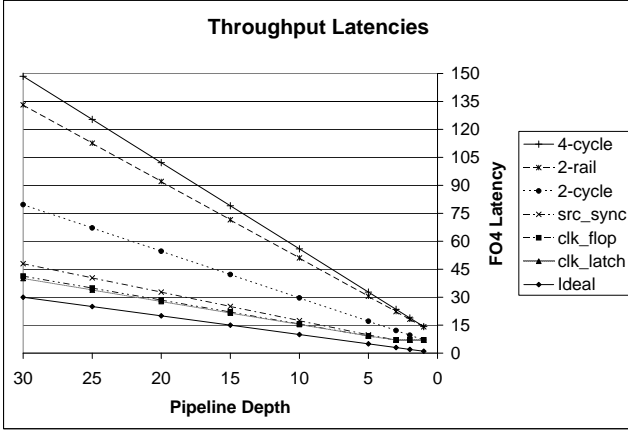


Figure 6. Worst-case protocol throughput

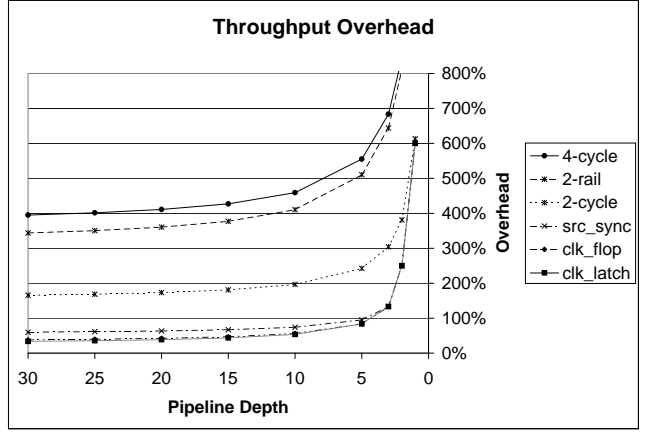


Figure 8. Throughput overhead against ideal

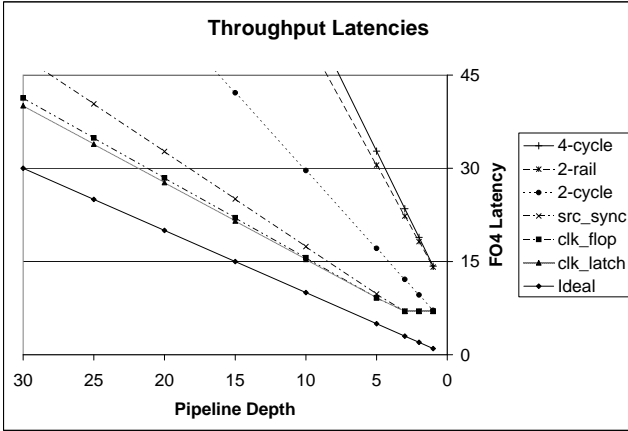


Figure 7. Throughput of efficient protocols

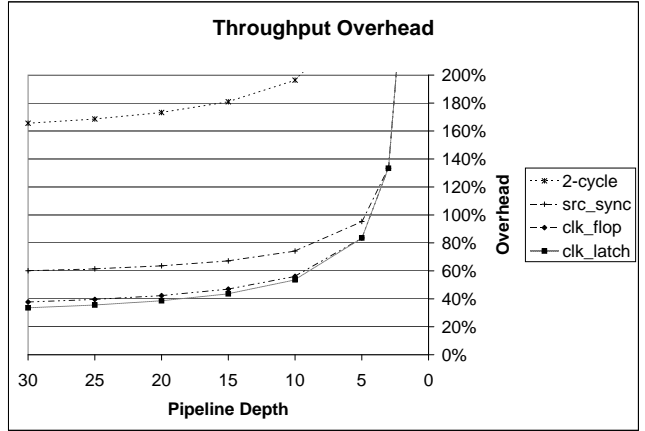


Figure 9. Overhead of efficient protocols

is possible to have worst-case skew between data and control in all stages. This creates a safe margin for the control and data race at the expense of throughput.

The clock parameters are best-case values, because the system clock cannot run faster than the slowest pipe stage. The unclocked numbers are worst-case because, based on variation, the fabricated circuits will likely run much faster than the values calculated by these equations.

Equation C_{ff+} is the maximum delay for a clocked flop protocol. The delay will be bounded by either of two conditions. First, the clock cycle cannot be shorter than twice the width of the minimum sized pulse $2T_{pw}$ that can be safely engineered and propagated. (This constraint results in the bandwidth wall reflected in Figure 2 for high frequencies.) Otherwise, the delay through a stage will be the sum of the maximum logic and communication delay from the source flop to the destination flop $T_{lc+} + T_{cc+}$. Clocked systems have additional overheads of setup time, clock skew and jitter, and other CAD related inaccuracies $T_{su} + T_{skj} + T_{cad}$. Upon arrival of the clock pulse, the data must also propagate

through the flop T_{cq+} .

Equation A_{di+} calculates the delay for the two DI protocols in this paper. For these protocols, the data code and acknowledgment operate on a four-cycle handshake moving from idle to valid then back to idle. Therefore, each encoding will have a data bit that rises and falls, and the acknowledgment will rise and fall. To be delay insensitive, these operations are sequential. Hence there are four cycles through the data acknowledge detection and propagation logic ($4T_{fsm+}$). The data transitions have less variation because the encoding result in the wires being half-shielded ($2T_{csh+}$). The acknowledge signal can be made wider to reduce delay on its transitions ($2O_{wd}T_{c+}$). When logic exists as well as communication, there is a delay for propagation through the logic (T_{l+}) and we assume this logic can be reset in parallel to generate the idle DI code (T_{dr}).

The models for other protocols are shown in Table 4. Logic delays for the more complicated bundled protocols are not modeled for clarity. This does not affect these results because the logic component T_l is set to 0 for these results.

Name	equation	description
C_{fe}	$W_b(2(E_{gf} + E_{clkf}) + A_w(E_{datf} + \max(0, E_{inv}(\frac{T_c}{D_c} - 1))))$	Clk_flop energy
C_{le}	$W_b(4(E_{gl} + E_{clk}) + A_w(2E_{datl} + \max(0, E_{inv}(\frac{T_c}{D_c} - 2))))$	Clk_latch energy
A_{4e}	$4(R_{ctl}L_g + \frac{T_c}{D_c}E_{inv}) + W_b(2(\frac{E_{clk}}{4} + E_{clk}) + A_w(E_{datl} + \max(0, E_{inv}(\frac{T_c}{D_c} - 1))))$	4-Cycle energy
A_{sp}	$2(R_{ctl}L_g + \frac{T_c}{D_c}E_{inv}) + W_b(2(\frac{E_{clk}}{4} + E_{clk}) + A_w(E_{datl} + \max(0, E_{inv}(\frac{T_c}{D_c} - 1))))$	Src_sync & 2-cycle
A_{die}	$2(R_{ctl}L_g + \frac{T_c}{D_c}E_{inv}) + 2W_b(\frac{L_g}{20} + \frac{L_g}{5} + \frac{L_g}{4} + \frac{T_c}{D_c}E_{inv})$	2-rail energy
A_{dne}	$2(R_{ctl}L_g + \frac{T_c}{D_c}E_{inv}) + W_b(\frac{L_g}{20} + \frac{L_g}{5} + \frac{L_g}{4} + \frac{T_c}{D_c}E_{inv})$	1-of-4 energy

Table 5. Models for average energy per transaction for one pipe stage.

5.4. Throughput

Figure 5 shows two pipelines with an ideal logic delay of approximately 4. For the ideal pipelines there is no variation, and the flops have zero latency and no overhead. This would allow a new data item to be propagated through these pipe stages every four gate delays. We compare the actual throughputs and overheads of the protocols as calculated by the equations in Table 4 against the ideal pipeline delays. The equations calculate extra delays that are added due to device variation, latch delays, etc. The communication delays calculated by the delay across a repeater and wire pair may not equal an integer value equivalent to a logic delay pipeline, as is the case in Figure 5. Under this circumstance the models permit a fraction of an inverter and communication wire delay to be used. While this works mathematically, it would require some adjustments to the buffer location and wire distances in the physical design.

The throughput models applied to our parameter values are plotted in Figures 6 through 9. The x-axis plots delays based on the pipeline granularity in terms of the number of gate delays between flops or latches. The leftmost side contains pipelines with a logic pipeline depth of 30 ideal gate delays per stage. The rightmost point contains a single ideal gate delay per stage. The y-axis plots FO4 delays, or an overhead that is calculated by comparing to the modeled worst-case delay to ideal delay.

These plots allocate all of the delay to communication. As expected, the asynchronous protocols with acknowledgment are the least efficient for communication, with the 4-cycle protocol being the worst. The most efficient protocols are clocked. The two-cycle protocol shows significantly better performance due to the reduction in control transitions propagated between sender and receiver. The source synchronous protocol is the best asynchronous protocol because its request is propagated as a pulse and no ack is explicitly included. This protocol is marginally slower compared to clocked protocols largely due to the conservative margin in the delay elements. In a real design, one

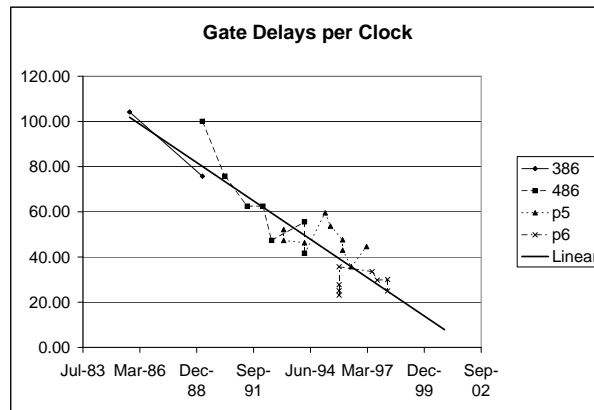


Figure 10. Gate delay scaling of products [3]

would expect the source-synchronous circuit to out-perform the clocked design due to its adaptive nature.

The cost of decreasing the amount of logic in each pipe stage to increase frequency can be inferred from these graphs. Figure 10 shows the historical trend in logic gates per pipe stage for Intel's recent microprocessors. As the amount of logic per stage is reduced, there is a considerable increase in power and performance sapping overhead. As shown in Figures 8 and 9, shortening the pipe depth has come at little cost in the past. As pipe depths continue to be shortened, the overheads start to dramatically increase. In a clocked latch design there is a 23% loss in efficiency as the pipe depth is decreased from 15 to 10 ideal gate delays. This balloons to a 56% loss if one moves from 10 to five ideal gate pipelines. Hence, future frequency increases for clocked designs will result in diminishing performance gains. This also comes at an increased power cost as is show in the next section. The same analysis applies to pipelining asynchronous systems, particularly if the pipeline has significant bubbles during full throughput operation.

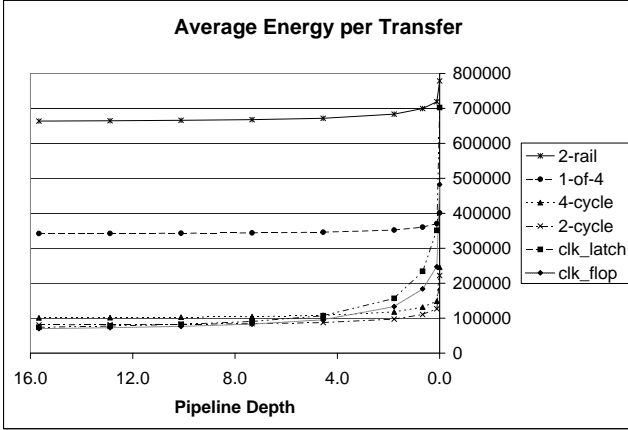


Figure 11. Average energy per transaction

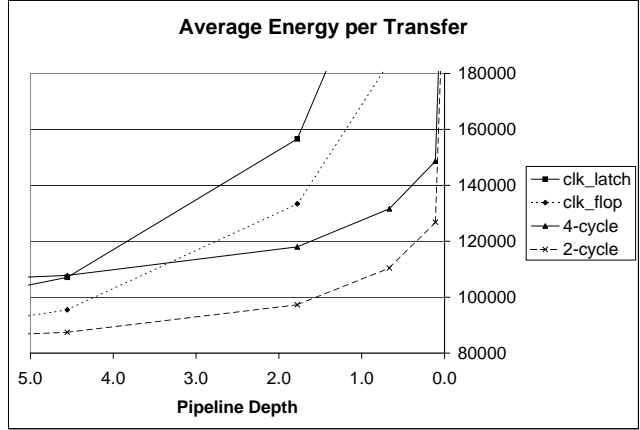


Figure 12. Highly pipelined transfer energy

5.5. Energy

Variables used for energy calculations are in Table 2. D_c is the total wire and gate delay to propagate a signal across the critical distance of interconnect. Wire delays account for close to half of the delay. Asynchronous protocols will expend more energy on each transmission than clocked protocols due to local clock generation and handshaking, but significantly less when the communication link is idle. Hence the size of the bus, the data activity factor, and how often the bus transmits data are key power parameters. Energy models for the protocol classes are shown in Table 5. These are based on the energy of the flops and latches from Table 3. Leakage power is not used in these calculations.

The clocked logic assumes perfect clock gating. Clock gating logic is modeled as a two-input NAND gate followed by an inverter for signal gain. The NAND is enabled only when data is to be actively transmitted through the flop. During idle times the only energy dissipated is the clock input to the gating NAND. This is optimistic because no energy is taken for the clock generation and distribution, the gating logic, for additional skew generated by gating, or for generating and transmitting valid bits across a link.

Both the clocked and unclocked protocols include buffers and drivers of sufficient energy to clock the datapath based on its width. Energy models for the asynchronous control also account for logic to drive the handshake signals and for internal logic transitions to calculate when control signals can switch. The control signals driving the repeater of size L_g are assumed to be buffered by an inverter of size $\frac{L_g}{4.8} = 25$. Three internal transitions are assumed to occur for each output transition. The transitions are assumed to be a gate of logical effort two, equivalent to a 4-input NAND. This results in a total control and driver power of $\frac{L_g}{4.8} + \frac{3L_g}{9.6} \approx 60$. Since this is half of L_g , the energy factor for control logic R_{ctl} is set to 0.5 in Table 2.

The equations in Table 5 are derived using energy values from Table 3 in similar fashion to the throughput equations. The equations calculate the energy dissipated between flops or latches in the segment, and may contain several repeaters. The bus width, latch clock and data energy profiles, activity factors of the data and control bits, number of repeater segments, and gain required to drive the latches and control wires are used to generate these models.

Clock Flop energy C_{fe} models the energy per bit times the bus width W_b . The two clock edges per cycle are multiplied by the gating energy E_{gf} and the energy to latch the flop E_{clkf} . The average bit activity factor A_w is multiplied by the data switching energy E_{datf} plus the drive through repeater segments between flops, if any. The asynchronous protocols are derived in the same way, with an additional set of numbers for control logic overhead R_{ctl} and propagation of the control signals, plus some extra energy for gain required to drive the latches. The delay insensitive protocols include state logic for one acknowledgment wire per bus, plus some additional logic per bit for latching and completion detection.

Figure 11 shows the energy dissipated to transmit data across a $10,000\mu$ bus with varying amounts of pipelining. Each equation in Table 5 calculates the energy for one flopped wire segment. This number is multiplied by the number of flopped segments $\frac{30}{T}$ in the bus, which scales the results for the same $10,000\mu$ distance. The x-axis shows the number of repeaters between the flops or control elements in the pipeline, the rightmost value when every inverter is replaced with a flop. The DI protocols have a much poorer power profile due to their activity factors. The rest of the protocols have fairly similar energy profiles. The curves are very flat except for ultra-fine grain pipelining, where the control overhead becomes a significant power drain on the circuits. For all of these circuits, the wire loads are the dominant power. Figure 12 shows that the bundled data

Name	equation	description
B_v	$1 + 1/W_b$	clocked valid bit
B_{da}	$2 + O_{wa}/W_b$	DI data encoding & ack
B_{ra}	$1 + 2O_{wa}/W_b$	req and ack signals
B_r	$1 + O_{wa}/W_b$	req signal

Table 6. Average wires per data bit.

asynchronous protocols are more energy efficient than the clocked protocols at high frequency.

5.6. Bandwidth

Figures 2 and 3 at the start of the paper were derived by using the latch and flop energy equations in Table 5 and the throughput equations of Table 4 scaled to a $10,000\mu$ bus. They were then area scaled for bandwidth as described in this section.

In order to effectively scale a chip, additional metal layers are added to support the increased bandwidth of the design. The effective utilization of wires in a process therefore determine the cost and bandwidth of the design. This work defines bandwidth to be proportional to the wire area. Table 6 shows how throughput is scaled based on the wire area to calculate the bandwidth of a design. The delay insensitive codes modeled here require two wires per data bit. Therefore, for the same throughput the bandwidth of a DI design would have roughly half the bandwidth. The penalty for control signal and valid bit overhead are also calculated for the studied designs as shown.

The final bandwidth models graphed in this paper are shown in Table 7. The value is the bandwidth-scaled number of data words that can be pipelined in the $10,000\mu$ wire. This allows us to compare bandwidth and area for various levels of pipelining across all the designs. The bandwidth values from this table is used to calculate the x-axis point for each protocol and the energy numbers are used for the y-axis values of Figures 2, 3, and 12.

6. Observations and Caveats

Accurate apples-to-apples comparisons are always challenging. These equations model the first-order effects of many implementation styles. The magnitude of many of these effects, such as first-droop, are arguable. Hence the models are highly parameterizable where the equations apply the parameters provided in a spreadsheet. This also allows one to study the results of an effect that is trending up or down as processes are scaled. The values used in this paper are loosely based on a 65nm process.

There are a number of effects that are difficult to model. The flop and latch design, for instance, will have impact on

Name	equation	description
B_{cf}	$30/(B_v C_{ff+})$	Clk_flop bandwidth
B_{cl}	$30/(B_v C_{l+})$	Clk_latch bandwidth
B_{di}	$30/(B_{da} A_{di+})$	2-rail and 1-of-4 b/w
B_2	$30/(B_{ra} A_{2+})$	2-cycle bandwidth
B_4	$30/(B_{ra} A_{4+})$	4-cycle bandwidth
B_{ss}	$30/(B_r S S_+)$	Src_sync bandwidth

Table 7. Bandwidth equations for the seven protocols. Value is the number of words that can be concurrently sent down a line with an ideal delay of 30 gates.

power and performance. Modeling all of the various choices is outside the scope of this work. This work picked a single simple flop and latch style and applied it to all protocols. This made the work relatively accurate between models, but the absolute value will be different based on the sequential used. However, the largest difference in most flop designs is the energy required in the clocking. Since this can be determined separately from data energy and delays the variation across flop styles should be rather simple to estimate.

For simplicity, control blocks for all asynchronous protocols exhibited the same delay T_{fsm} of 2.5. In general, there can be a significant variation in the latency of control for these protocols. The two-cycle NRZ protocol, for instance, should arguably have a larger energy and delay penalty than control for a pulse or four-cycle RZ protocol. These effects can be modeled among the different protocols by providing a unique T_{fsm} parameter for each protocol that accurately models the average delay.

This work can also be applied directly to pipeline protocols for logic blocks, and some of the protocols include this modeling. However, the logic family, design and protocol style, and implementation aspects of the logic blocks have a much greater variability than a repeated wire, so the accuracy of such a model is much more difficult to compare and arguably less accurate. Min-delay issues become very important, but they can largely be ignored in communication since max and min signal propagation delays will be similar.

Fixed bandwidth can be achieved through combinations of throughput and parallelism. For instance doubling the throughput and halving the wires will achieve the same bandwidth, at a smaller area. This can be accomplished using these models. The activity factors for the serialized links may increase significantly.

The parameter values used in this paper represent a single design point. For example, increasing the activity factor of the bus by a factor of 10 makes no significant change to Figures 2 and 3. However, decreasing the activity factor by

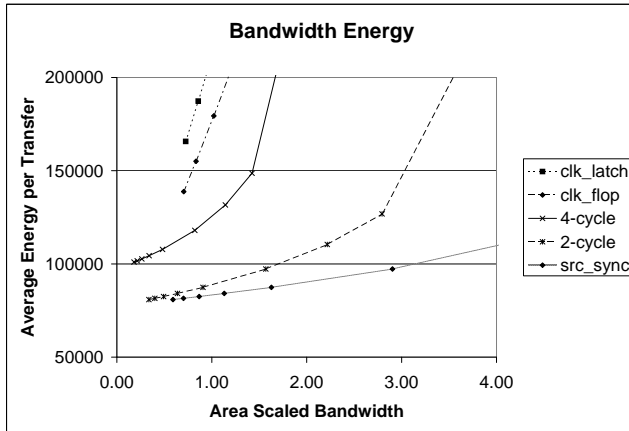


Figure 13. Bandwidth/energy for lower bus utilization

the same amount gives a significant advantage to the asynchronous protocols as shown in Figure 13. Here the 4-cycle, 2-cycle and source synchronous protocols show significant energy advantages at all bandwidths over clocked designs. Thus, the parameters need to be configured based on your target application.

Optimization based on these models has not been done. For instance, there is a tradeoff between faster control signal propagation and its deleterious effect on bandwidth.

7. Conclusions

Clocked and efficient unclocked protocols are shown to have similar results when measuring average transaction energy for a target bandwidth using parameters targeted for a microprocessor bus. This is one of the worst-case scenarios for asynchronous design.

Many design parameters favor either a clocked or unclocked style. Wider buses and high bit-level activity factors favor asynchronous communication. Higher bus utilization factors favor clocked designs. Changing the operating environment reflected in these parameters can result in asynchronous communication being far superior to clocked protocols.

The pipelining can be dynamically chosen in asynchronous designs, whereas in a clocked topology the distances and pipelining is fixed relative to the clock frequency. This implies that scalability and the ability to optimize for a particular power/performance point is enhanced in asynchronous designs. Furthermore, asynchronous designs can be repeated and pipelined at the optimal critical distance for the target topology without requiring overhead for future process scaling. The asynchronous protocols require increased pipelining to achieve similar bandwidths to the

clocked protocols. The physical level efficiency of the asynchronous communication is very dependent on the protocol.

Energy per transaction remains relatively flat until pipelining becomes aggressive. At that point there begins to be a considerable penalty for increasing the pipelining.

This study only compares the physical data transmission efficiencies. Communication effects on the overall processor performance and power are an important extension [4]. The benefits of implementing asynchronous communication in an otherwise globally synchronous processor must override the cost of synchronization with the destination frequency. Future designs - such as those with multiple on-die cores or designs with power islands - will be breaking the chip into different clock domains for power, thermal, and performance reasons. For such architectures, asynchronous communication exhibits significantly lower latency and has been shown, through this study, to have similar or better physical transport efficiencies when compared to clocked methodologies.

References

- [1] W. J. Bainbridge. *Asynchronous System-on-Chip Interconnect*. PhD thesis, Department of Computer Science, University of Manchester, Mar. 2000.
- [2] D. Harris. *Skew-Tolerant Circuit Design*. Morgan Kaufmann Publishers, 2001.
- [3] R. Ho, K. W. Mai, and M. A. Horowitz. The future of wires. *Proceedings of the IEEE*, 89(4):490–504, April 2001.
- [4] A. Iyer and D. Marculescu. Power-Performance Evaluation of Globally Asynchronous, Locally Synchronous Processors. In *Proc. Intl. Symposium on Computer Architecture (ISCA)*, pages 158–168, Anchorage, AK, May 2002.
- [5] S. J. Piestrak. Membership test logic for delay-insensitive codes. In *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 194–204, 1998.
- [6] C. L. Seitz. System timing. In C. A. Mead and L. A. Conway, editors, *Introduction to VLSI Systems*, chapter 7. Addison Wesley, 1980.
- [7] M. Singh and S. M. Nowick. High-throughput asynchronous pipelines for fine-grain dynamic datapaths. In *Proc. International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 198–209. IEEE Computer Society Press, Apr. 2000.
- [8] K. Stevens, S. Rotem, R. Ginosar, P. Beerel, C. Myers, K. Yun, R. Kol, C. Dike, and M. Roncken. An Asynchronous Instruction Length Decoder. *IEEE Journal of Solid State Circuits*, 36(2):217–228, Feb. 2001.
- [9] I. Sutherland, B. Sproull, and D. Harris. *Logical Effort: Designing Fast CMOS Circuits*. Morgan Kaufmann Publishers, Inc., San Francisco, 1999.
- [10] N. Weste and K. Eshraghian. *Principles of CMOS VLSI Design: A Systems Perspective*. VLSI Systems. Addison-Wesley, Menlo Park, CA, 1985.