Estimation of Energy Consumption in Speed-Independent Control Circuits

Peter A. Beerel and Cheng-Ta Hsieh and Suhrid Wadekar
EE-Systems Department
University of Southern California
Los Angeles, CA 90089

Abstract: We describe a technique to estimate the energy consumed by speed-independent asynchronous (clockless) control circuits. Because speed-independent circuits are hazard-free under all possible combinations of gate delays, we prove that an accurate estimate of their energy consumption is independent of relative component gate delays and can be determined by simulating only a small number of input patterns proportional to the size of the circuit's Signal Transition Graph (STG) specification. Specifically, we calculate the *average energy per external signal transition* consumed by a circuit. This can be used to compare the energy consumption between two different circuit implementations of the same specification, to calculate average energy for a given high-level operation, and to provide average circuit power when combined with delay information.

1 Introduction

Because asynchronous circuits do not have a global clock, *energy consumption per clock cycle*, a common means of quantifying energy dissipation in synchronous circuits, can not be used. Instead, it has been suggested to quantify energy consumption in asynchronous circuits using *energy per operation* (cycle of activity) [6]. This approach has been applied to asynchronous circuits that use a two-phase signaling protocol for request/acknowledge handshaking. The control circuits addressed in [6], however, are assumed to be in a small set of pre-defined macro gates whose energy consumption *per output transition* is pre-calculated using SPICE. The energy per output transition for all control circuits is then incorporated into the overall estimate for energy and power consumption [6].

This paper, on the other hand, considers the problem of measuring energy consumption of a more general class of control circuits to support energy and power estimation of other types of asynchronous circuits. Specifically, we consider control circuits that are netlists of gates rather than pre-defined macro gates. These circuits are speed-independent, *i.e.*, they are hazard-free under any set of component gate delays. Moreover, the circuits can exhibit input choice, in which the environment can non-deterministically decide what external inputs to change, driving the circuit into different *modes* of operation, such as a read/write mode in a memory interface controller.

Since different internal signals may change in different modes, each mode may have a different energy consumption. To obtain a unique figure of merit, it is necessary to combine the energy consumption in each of the different modes of operation into a single quantity. To do this, we assume the availability of input-choice statistics, which describe the relative probabilities of different environmental choices. In practice, these input-choice statistics may be given by the user or estimated through behavioral simulation.

Our procedure assumes the control circuit is specified using a *Signal Transition Graph (STG)*. Starting with the STG, we derive a *Sequential Signal Transition Graph (SSTG)* that defines a small set of input sequences that are simulated to obtain the switching activity of circuit nodes. The SSTG also defines a Markov chain which under reasonable circuit assumptions is irreducible, recurrent, and periodic (see section 4). Using Markov chain analysis, we obtain a small system of linear equations whose solution, when combined with the simulation results of the SSTG, yields the average energy per external signal transition.

The organization of the paper is as follows. Section 2 gives a more complete introduction to calculating energy in asynchronous circuits. Section 3 describes the classes of circuits and specifications considered in this paper. Section 4 presents a proof that the order of concurrent transitions does not affect the energy consumption of the circuit. Section 5 describes our SSTG. Section 6 describes how the SSTG defines a Markov chain which leads to a system of linear equations whose solution is used to calculate the average energy. Section 7 describes how this work relates to previous work on the energy consumption of synchronous circuits. Section 8 describes some preliminary results and conclusions.

2 Calculating energy consumption

The lack of a global clock in an asynchronous circuit and the concurrent operation of the circuit and its environment obscures any obvious time frame to measure energy consumption. If the circuit repeatedly performed the same operation, such as asynchronous datapaths, then energy per operation would be a useful measure of energy consumption [6].

Control circuits, however, do not perform a well-defined single operation repeatedly and hence energy per operation is not suitable for quantifying energy use. Kudva *et al.* [6] suggest measuring the *average energy per output transition* using SPICE simulation on the final layout, which is practical when the circuits are single pre-defined gates. Yet, for other asynchronous design styles the control circuits are synthesized automatically or semi-automatically using basic gates (*e.g.,* [2, 9]). SPICE simulation is impractical for these circuits since energy consumption estimates are desired before their physical design. Moreover, since energy might be expended in response to input signal changes prior to any output signal change, we estimate a slightly different figure of merit: average *energy per external (input and output) signal transition.*

To illustrate the issues involved in calculating energy per external signal transition, let us consider the speed-independent control circuit and its STG specification depicted in Figure 1. This control circuit, called *sbuf-send-ctl*, has 5 output signals, 3 input signals, and 9 internal signals. The arrows in the STG specification describe sequencing requirements between transitions of input and output signals. For example, the arrow between *rejpkt+/1* and *y1+/1* means that after the environment raises *rejpkt*, the circuit should raise *y1*. The two arrows into *rejpkt-* from *idlebar+* and *latchaddr+/1* respectively mean that once both *idlebar+* and *latchaddr+/1* occur, the environment can lower *rejpkt*. In addition, the STG specifies environmental choice using arrows that emanate from circles, called *places*. For example, in the

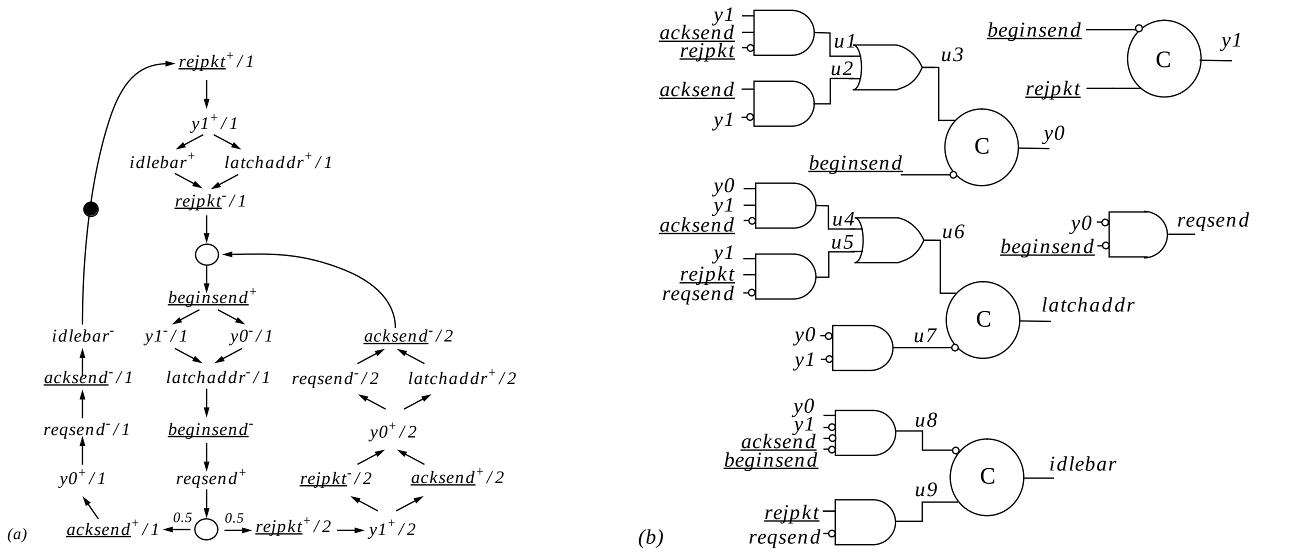


Figure 1: a) An STG specification of the *sbuf-send-ctl*circuit [4, 7] and b) a corresponding speed-independent implementation.

above STG, there exists two arrows emanating from a place to *acksend+/1* and *rejpkt+/2* with an arrow from *reqsend+* leading into the place. This means that after *reqsend+* occurs, the environment chooses between raising *acksend* or *rejpkt*, and thereby driving the circuit into different cycles of activity. The dark dot on the transition between *idlebar-* and *rejpkt+/1* denotes the initial state of the system; the first transition that can fire is *rejpkt+/1*. (An STG's formal semantics is given in Section 3.1.)

To define energy per external signal transition for such a circuit, the following definitions are needed: a *trace* T of the circuit is a sequence of STG transitions $\langle t_1, \ldots, t_n \rangle$; $\mathcal{T}$ is the set of all traces that start at the initial state and are specified to occur by the STG; and, $\mathcal{T}^k$ is a set of all traces in $\mathcal{T}$ that have k external signal transitions. For example, in the *sbuf-send-ctl* STG, one trace in $\mathcal{T}^5$ is $T_1 = \langle$*rejpkt+/1, y1+/1, idlebar+, latchaddr+/1, rejpkt-/1*$\rangle$.

Note that there may be two signals in the STG specified to change concurrently. For example, the above specification allows *idlebar+* and *latchaddr+/1* to fire concurrently after *y1+/1* fires. In this case, each ordering of these two transitions represents one *interleaving* of the two concurrent transitions. For each interleaving there exists a different trace in $\mathcal{T}$. Hence, $\mathcal{T}^5$ also contains the trace $T_2 = \langle$*rejpkt+/1, y1+/1, latchaddr+/1, idlebar+, rejpkt-/1* $\rangle$.

Second, we define the energy consumed by a circuit during the execution of a trace T, denote $En(T)$. A trace describes only the behavior of the external circuit signals, while the energy consumed during the execution of a trace is also dependent on the behavior of the internal signals in the circuit. The number of times gates switch during execution, the order of arrival of input changes to a gate, the relative directions of the changes, and the capacitive load on a gate output all determine the energy consumption. This energy is computed *assuming the internal signals are allowed to settle before and after the execution of the trace*. For simplicity, we assume this energy depends only on the capacitance at the output of the gates that switch during the execution of T. The well-known formula for energy consumption is:

$$En(T) = \sum_{\text{gates}} \frac{1}{2} \cdot C_{\text{gate-load}} \cdot Vdd^2 \cdot (\text{\# of gate switches}). \quad (1)$$

A more accurate model of $En(T)$ would consider the energy consumed through charging/discharging of capacitance that is internal to gates in the circuit along with other factors, such as short-circuit current.

Consider the simulation of the trace T_1. Initially, the inputs and outputs are held fixed at the circuits initial state and the internal signals are allowed to settle. Since, in the initial state, only signal *y0* is high, the only internal signal high is *u8*. Starting from this state, the trace is simulated and the number of circuit signals that switch is recorded. In this case, signals *y1* and *idlebar* go high, signal *u8* goes low, and signals *u5*, *u6*, and *u9* first go high and then go low. Each circuit signal transition is guaranteed to be monotonic because the circuit is hazard-free [1, 2].

Third, we associate a probability for each trace in $T \in \mathcal{T}^k$, denoted $Pr_{STG}(T)$, subject to the constraint that $\sum_{T \in \mathcal{T}^k} Pr_{STG}(T) = 1$ for all k. This probability depends on gate delays and the probability of various environmental choices. For example, consider the two traces T_1 and T_2 in $\mathcal{T}^5$. Since they are the only two traces of length 5, the sum of their probabilities should equal 1. Their exact probabilities, however, depend on the relative delays of the sub-circuits that control them. For example, since the sub-circuit for *latchaddr* has 3 levels of logic and the sub-circuit for *idlebar* only has 2 levels, *idlebar+* will most likely fire first. Since *idlebar+* fires first in T_1, a delay analysis might generate a probability of 0.8 for T_1 and 0.2 for T_2.

Finally, we can present our definition for average energy per external signal transition, denoted $\overline{E}$ as follows:

$$\overline{E} = \lim_{k \to \infty} \frac{\sum_{T \in \mathcal{T}^k} En(T) \cdot Pr_{STG}(T)}{k}. \quad (2)$$

The focus of the paper is to provide an efficient technique to calculate the above equation for a large class of speed-independent control circuits. To do this, we prove that for the class of speed-independent circuits we consider, any pair of traces T and T' that differ only in the order of concurrent transitions consume the same energy, *i.e.*, $En(T) = En(T')$. For example, the traces T_1 and T_2 discussed in the last section differ only in the order of concurrent transitions. Hence, both traces consume the same amount of energy.

As a result of this property, an *equivalence relation* that identifies traces that consume equal energy, differing only by the ordering of concurrent transitions, can be defined. This relation partitions the traces $\mathcal{T}$ into a set of *equivalence classes* Φ. Each class $\phi \in \Phi$ contains all traces in which the same input choices are made by the environment. We define the probability of an equivalence class $Pr_{STG}(\phi)$ to be the sum of the probabilities of all traces inside the class. The energy consumed by an equivalence class, $En(\phi)$, is equal to the energy consumed by any trace T in ϕ. Average energy can then be computed based on the probability of each equivalence class $Pr_{STG}(\phi)$ as follows:

$$\overline{E} = \lim_{k \to \infty} \frac{\sum_{\phi \in \Phi^k} En(\phi) \cdot Pr_{STG}(\phi)}{k}. \qquad (3)$$

This result is the key justification of our power estimation procedure. This equation refers to the probability of equivalence classes rather than that of individual traces. In the presence of concurrent transitions, the probability of individual traces can depend on relative delays inside the circuit and the environment, in contrast, the probability of an equivalence class depends only on the statistics of various input choices made by the environment. Hence, this result enables us to estimate energy consumption without performing a complicated delay analysis on the circuit or its environment.

Moreover, this equation enables us to calculate average energy by simulating the circuit using a derived sequential signal transition graph that defines a subset of the traces of the STG and does not model any concurrency. We will show that because of the lack of concurrency, the SSTG defines a Markov chain. Using Markov chain analysis, the proportion of each STG transitions that fires in an average trace in the SSTG can be calculated. This result leads to a small system of linear equations that can be used to calculate $\overline{E}$.

3 Specifications and implementations

In this paper we restrict ourselves to free-choice STG specifications and their speed-independent implementations.

3.1 Free-choice STG

An STG [3] is a specification formalism for asynchronous sequential circuits. It is an interpreted Petri net, and as such, can explicitly capture *causality*, *concurrency*, and *choice*.

A Petri net is a triple $N = \langle P, R, F \rangle$, where P is a set of *places*, R is a set of *transitions*, and $F \subseteq (P \times R) \cup (R \times P)$ is the *flow relation*. A place $p \in P$ is a *predecessor* of a transition $t \in R$, and t is a *successor* of p if $(p, t) \in F$. Conversely, a transition $t \in R$ is a predecessor of a place $p \in P$, and p is a successor of t if $(t, p) \in F$.

A *free-choice net* (FC net) is a Petri net where if a place p has more than one transition as its successors, then p must be the only predecessor of its successor transitions. Such a p is called a *free-choice* place.

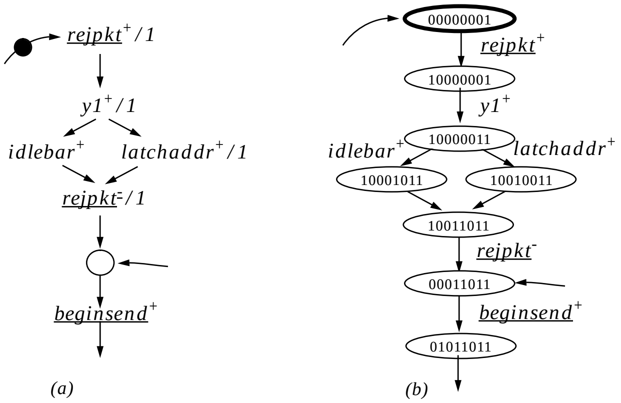
An STG is an *interpreted free-choice Petri net*: transitions of the net are interpreted as value changes on external signals of the specified circuit. *Positive* transitions (labeled with a "+") represent $0 \rightarrow 1$ changes, *negative* transitions (labeled with a "-") represent $1 \rightarrow 0$ changes. *Input transitions*, which are underlined for emphasis in Figure 1, are those that occur on circuit input signals I, *output transitions* are those that occur on circuit output signals O.

The conventional graphical representation of an STG is slightly different than conventional Petri net representations. STGs are represented using a directed graph, where transitions are identified by their name, places are denoted by circles, and directed edges represent elements of the flow relation. Places with only one predecessor and one successor are omitted. Directed edges whose successor is a transition represent sequencing constraints, either on the circuit to be synthesized (if their successor is an output transition), or on the environment (if their successor is an input transition). They specify what set of transitions cause each transition.

A *token marking* of a Petri net is a non-negative integer labeling of its places. A transition is *enabled* (*i.e.*, the corresponding event can happen in the circuit) whenever all its predecessor places are marked with at least one token.

An enabled transition may *fire*. This means that the corresponding external signal changes value in the circuit. When it fires, a token is removed from every predecessor place, and a token is added to every successor place.

If a place marked with only one token has more than one enabled successor transition, then only one of them may fire non-deterministically. The other transitions are *disabled* by its firing. Hence, such successor transitions are referred to as *choice transitions*.



State: [rejpkt beginsend acksend latchaddr idlebar reqsend y1 y0]

Figure 2: A portion of the STG and SG for *sbuf-send-ctl*.

A marking M'' is *reachable* from another marking M' if there exists a sequence of enabled transition firings that produces M'' starting from M'. A marking M' is *live* if for all markings M'' reachable from M', every transition can be enabled through some sequence of firings from M''. A marked net is live if its initial marking is live. A marking M' is bounded if the number of tokens that any place can be holding after any sequence of firings from M' is bounded. A marking M' is *safe* (sometimes referred to as 1-bounded) if its is 1-bounded. A marked net is 1-bounded (safe) if its initial marking is 1-bounded (safe).

Two transitions are said to be *concurrent* iff there exists a marking, reachable from the initial marking, in which both transitions are enabled.

An STG is live iff 1) the underlying free choice net is live and safe, 2) no two transitions of the same signal are concurrent, and 3) if two up-transitions of a signal are fired, a down-transition of the same signal should always be fired in between (and vice versa). In this paper, we restrict ourselves to live STGs.

We annotate choice transitions with choice probabilities that designate the probability of the environment firing one of multiple mutually-exclusive input transitions. In our example, we annotated both choice transitions *acksend+/1* and *rejpkt+/2* with a probability of 0.5.

The reachability graph of a Petri net is a directed graph where each node corresponds to a marking and an edge joins a pair of marking M' and M'' if there exists a transition t that firing from M' produced M'' (the transition labels the edge).

The *state graph* (SG) [3] of an STG is the reachability graph of the underlying net where each node (henceforth called *state*) is labeled with a vector of signal values. For a given state s, the value of signal u is given by $s(u)$. This node labeling must be consistent with the SG edge labeling, in other words for each edge $s \rightarrow s'$ and each signal u: if the edge is labeled u^+, then $s(u)$ must be 0 and $s'(u)$ must be 1; if the edge is labeled u^-, then signal $s(u)$ must be 1 and $s'(u)$ must be 0; and otherwise $s(u)$ must equal $s'(u)$. The labeling can always be done, as proven in [3], if the STG is live. The procedure to find a SG from a STG is called *token flow* and is given in [3]. A portion of the SG along with the corresponding portion of the STG is depicted in Figure 2.

Notice that the SG can be exponentially larger than its STG. The size difference is dependent on the amount of concurrency expressed in the STG. In the STG, concurrency is expressed implicitly with independent parallel paths of transitions, while, in a SG, each different interleaving is explicitly represented by different paths through the SG. The two different interleaving T_1 and T_2 that were discussed earlier correspond to the two paths through the portion of the SG depicted in Figure 2.

Our procedure for estimating power does not explore the various interleavings and hence avoids the exponential complexity associated with the state graph.

3.2 Speed-independent circuits

We model a circuit with a five-tuple $G = \langle I, O, N, E, F \rangle$. The sets I, O, and N are the input, output and *internal* signals of the circuit respectively. They are collectively called the *circuit signals* and denoted A_{Impl}. $E \subseteq A_{Impl} \times A_{Impl}$ is a set of directed edges that defines the connectivity of the circuit signals. An edge $e = (u, v)$ means that signal u is a fanin of signal v. F is a labeling function that labels each internal and output signal u with a binary function f_u that describes the function of the gate that drives this signal. Two-output gates such as mutual exclusion elements can not be modeled in this framework. Consequently, circuits that contain such elements are not considered.

A signal is *enabled* if its current value does not equal the value of its function given the current value of its fanin. For example, if the current value of u is 0 and it is driven by an AND gate whose inputs are all 1, then u is enabled. When the signal changes value, the signal *fires*. A signal is *disabled* if, before it fires, one or more of its fanins change such that it is becomes no longer enabled. In practice, the disabling of a signal can manifest itself as a voltage spike or runt pulse. For a broad class of circuits using a conservative delay model, there always exists a set of delays such that this spike can propagate to a primary output [1]. Therefore, such a disabling is generally considered a *hazard* and should be avoided. For the sequel, a circuit is *speed-independent* iff the circuit is guaranteed to be void of hazards under all possible gate delays (assuming negligible wire delay) and the circuit satisfies its specification [1].

4 Concurrency and energy consumption

This section formalizes our claim that energy consumption can be calculated without analyzing all possible interleavings. First, we prove that two traces that differ only in the order of two concurrent transitions consume the same energy. Then, we extend the result to all members of an equivalence class of interleavings.

Let a binary relation $W \subseteq \mathcal{T} \times N$ be the set of pairs (T, i) such that in trace T transition t_i and t_{i+1} are concurrent. Let the function *swap*(T, i) return the trace obtained from T by swapping the i^{th} and $i + 1^{\text{st}}$ transitions. Then, *swap*(T, i) is *legal* if $(T, i) \in W$.

Theorem 1 *If swap*(T, i) *is legal, then* $En(T) = En(swap(T, i))$.

Proof: (By contradiction, sketch) For the purpose of reaching a contradiction, presume the energy consumed in the two different traces is different. Then there exists a gate u which fires in the simulation of one but not both traces. Starting from a simulation point in which signal u is enabled, simulate the remaining part of the trace without letting u fire. The circuit status after this simulation trace should be the same as after simulations in which u does not fire. By definition of our simulations, at this simulation point no internal signal is enabled. Hence, gate u must have been disabled and the circuit must not be speed-independent. A contradiction. ■

To extend this result to any set of transitions that differ only in the interleavings of concurrent transitions, we define a relation $C \subseteq \mathcal{T} \times \mathcal{T}$ such that $(T, T') \in C$ iff there exists a sequence (including the zero length sequence) of legal swaps that transform T into T'. Then, any two traces in C consume the same amount of energy. Moreover, it is easy to prove that C is an equivalence relation. Therefore, it partitions a trace set $\mathcal{T}$ into a set of equivalence classes Φ. We let Φ^k represent the equivalence classes of the set of traces of length k.

The equation for average energy can be recast in terms of these equivalence classes rather than individual traces to obtain Equation 3 repeated here for convenience:

$$\overline{E} = \lim_{k \to \infty} \frac{\sum_{\phi \in \Phi^k} En(\phi) \cdot Pr_{STG}(\phi)}{k}. \tag{4}$$

Algorithm 5.1 (Find SSTG)

```
/* Let $N = \langle P, R, F \rangle$, be the specification STG. */
/* Let $M$ be a marking of the STG. */
/* Let $t$ be a transition that drives the circuit into $M$. */
/* Let fire$(M, t')$ be the marking obtained from $M$
    after firing enabled transition $t'$. */
/* Let set $S$ contain transition-marking pairs $(t, M)$. */
/* Let $i$ be a time stamp. */
Find_SSTG$(N, M)$ {
  initialize $S$ to include $(null, M)$ and set $i = 0$
  while $S$ is not empty
    remove some transition-marking pair $(t, M)$ from $S$.
    call add_state$(t)$ to add a node for $t$ to the SSTG
    if there exists an unseen non-choice transition $t'$
        enabled in $M$
      call add_edge$(t, t')$ to connect nodes for $t$ and $t'$
      mark $t'$ as seen at time $i$ and increment $i$
      add the transition-marking pair $(t', fire(M, t'))$ to $S$
    else if there exists a set $D$ of unseen mutually-exclusive
          choice transitions enabled in $M$
      foreach choice transition $t'$ in $D$
        call add_edge$(t, t')$ to connect nodes for $t$ and $t'$
        mark $t'$ as seen at time $i$ and increment $i$
        add the transition-marking pair $(t', fire(M, t'))$ to $S$
    else if there exists seen transitions enabled in $M$
      let $t'$ be the seen transition with the lowest time stamp
      call add_edge$(t, t')$ to connect nodes for $t$ and $t'$
  call add_places() to add free-choice places at transitions
      with multiple predecessors or successors.
}
```

Figure 3: The algorithm to find the SSTG from a given STG.

The advantage of this alternative representation is that the probabilities of each equivalence class depend only on input-choice statistics. Hence, a delay analysis of the circuit to determine relative probabilities of different interleavings caused by concurrent output signals is unnecessary.

5 The sequential signal transition graph

The energy of an equivalence class, $En(\phi)$, is measured by simulating a *characteristic trace* T_ϕ of ϕ This characteristic trace is defined by imposing an order on concurrent STG transitions which is the same for all equivalence classes. Such an ordering ensures that equivalence classes that contain different subtraces corresponding to the same part of the STG will have the same choice of interleavings. We specify this ordering by taking the original STG and deriving a new STG, called a *sequential signal transition graph* (SSTG), in which all concurrency is serialized.

Algorithm 5.1, presented in Figure 3, finds an SSTG for a given STG. The resulting SSTG is comparable in size with the initial STG. When run on the STG given in Figure 1, the algorithm derives the SSTG illustrated in Figure 4.

Note that each trace of the SSTG T_ϕ is a characteristic trace of the equivalence class ϕ in the original STG. Infact, the purpose of the SSTG is that the probability of the equivalence class ϕ, which recall is defined as the sum of $Pr_{STG}(T)$ for all traces T in ϕ, is equal to the probability of T_ϕ in the SSTG, *i.e.*,

$$Pr_{STG}(\phi) = \sum_{T \in \phi} Pr_{STG}(T) = Pr_{SSTG}(T_\phi) \tag{5}$$

Using this fact, the average energy of the circuit can be computed as follows:

$$\overline{E} = \lim_{k \to \infty} \frac{\sum_{T_\phi \in \mathcal{T}^k_{\text{SSTG}}} En(T_\phi) \cdot Pr_{SSTG}(T_\phi)}{k}, \tag{6}$$

where $\mathcal{T}_{SSTG}$ is the set of all traces in the SSTG.

6 Calculating average energy

To calculate the average energy using the SSTG we define the *long term proportion* of an SSTG transition t, denoted

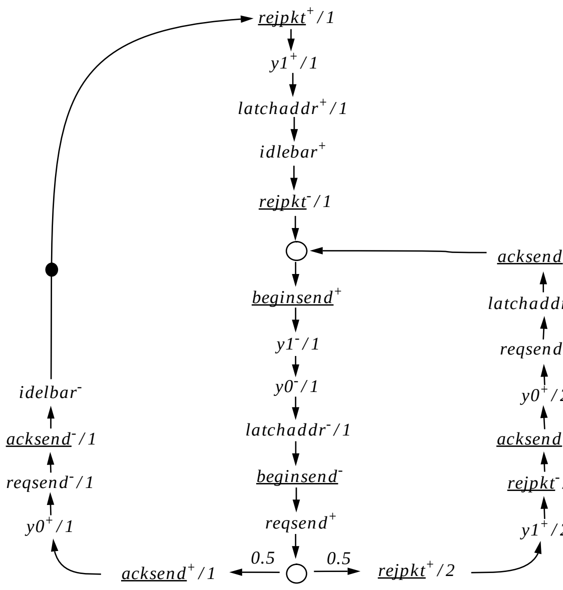


Figure 4: The SSTG for the circuit *sbuf-send-ctl.*

Π_t, as follows:

$$\Pi_t = \lim_{k \to \infty} \frac{\sum_{T_\phi \in \mathcal{T}^k_{\mathrm{SSTG}}} (\#\text{ of } t \text{ in } \phi) \cdot Pr_{SSTG}(T_\phi)}{k}. \quad (7)$$

Then, the equation for average energy can be recast as follows:

$$\overline{E} = \sum_{t \in R} \Pi_t \cdot En(t), \quad (8)$$

where R is the set of STG transitions and $En(t)$ is the energy consumed simulating transition t. Note that each SSTG transition t has a unique corresponding SG state s in which t is enabled in which simulation begins. (This state can be obtained through a reachability analysis of the SSTG.) We begin simulation of the transition assuming all internal circuit signals settle in this state. Then, we fire the external transition and simulate the circuit until all internal signals have once again settled, recording which circuit signal change. During simulation, a zero delay model can be used for circuits which have no internal feedback since internal signals can change at most once per external signal transition. Many automatically synthesized circuits have this property, *e.g.,* [2]. When internal feedback can exist, on the other hand, internal signals may change multiple times per external signal transition, requiring the use of a more realistic delay model.

Developing this equation from the SSTG is important because now *Markov chain* theory can be used to obtain the long-term proportions of STG transitions. To demonstrate this, we first review Markov chains, then, explain why the original STG does not generally define a Markov chain, and demonstrate that, under the assumption of *uncorrelated choices*, the traces of the SSTG define a Markov chain.

Consider a stochastic process $\{X_n, n = 0, 1, 2, \ldots\}$. If $X_n = i$ then we say the process is in state i at time n. If the conditional distribution of any future state X_{n+1} is independent of the past states and depends only on the present state, then the process is a Markov chain.

First, consider the stochastic process $\{T_n, n = 0, 1, 2, \ldots\}$ that models all possible traces in the *original* STG. In each "time step", T_n can take on a *finite* set of values, the STG transitions. If $T_n = t$, then the n^{th} state of the process is the STG transition t. The traces *do not* define a Markov chain because concurrency is allowed. Consider, for example, two concurrent transitions t and t'. The probability of a future state T_{n+1} being transition t' given that $T_n = t$ depends on whether or not t' was visited earlier (such as in state T_{n-1}), *i.e.*, the conditional distribution depends not only on the current state of the process, but also on past states. This violates the Markov model.

Now, consider the traces in the SSTG which exhibits no concurrency. The only decision points made during the trace are at choice places. If there is no correlation between consecutive input choices, the conditional distribution of the future states T_{n+1} depends only on the present state. The transition probability between states corresponding to STG transition t and t', denoted $P_{tt'}$, are as follows:

- zero if there is no edge from t to t';
- one if the only edge into t' is from t; and
- the choice probability of t, if t is a choice transition.

Hence, the stochastic process is a finite-state Markov chain. In fact, in Section 7, we will argue that for speed-independent circuits, correlations among input choices do not affect the average energy consumption of the circuit.

Before we can present our final result, we must demonstrate the our Markov chain satisfies a few properties. A state t in a Markov chain is *recurrent* if, starting in state t, the process will ever reenter state t. Since the STG is live we are guaranteed after firing t there will exist a trace in the original STG that refires t. Since we do not allow zero choice probabilities we are guaranteed that the probability of this trace is non-zero. Hence, our Markov chain is recurrent.

A Markov chain is *irreducible* if every state can be reached from every other state. Because the original STG is live, from every marking there exists a sequence of transitions which enables every transition. Since this property extends to our SSTG, our Markov chain is irreducible.

State t in a Markov chain is *periodic* with *period* d if the probability of being in state t after n transitions starting in state t is 0 whenever n is not divisible by d and d is the largest integer with this property [11]. In other words, starting in state t, it may be only possible to enter state t at times 2, 4, 6, 8, etc, in which case state t will have period 2. State t is called *aperiodic* if d equals 1. Since a STG transition t cannot fire twice without firing a complete cycle of the transitions in the STG, our Markov chain is periodic.

Because our finite-state Markov chain is irredundant, recurrent, and periodic, it can be shown that the Π_t are the unique non-negative solutions to the following equations [11]:

$$\Pi_{t'} = \sum_{t \in R} \Pi_t \cdot P_{tt'}, \qquad \text{for all } t' \in R \quad (9)$$

$$\sum_{t \in R} \Pi_t = 1 \quad (10)$$

Because most of the transitions in the SSTG have a single predecessor, many of the above equations are trivial. For example, since *y1+* has a single predecessor, namely *rejpkt+/1*, one of the equations is $\Pi_{y1+} = \Pi_{rejpkt+/1}$. In other words, many transitions are equiprobable and can be combined into a *sequence of transitions* L_m. The different sequences β form a partition of the signal transitions in the SSTG. For the SSTG depicted in Figure 4, we illustrate this collapsing of equiprobable transitions in Figure 5.

The new system of linear equations based on this partition of sequences of equiprobable transitions are as follows:

$$\Pi_{L_m} = \sum_{L_l \in \beta} \Pi_{L_l} \cdot P_{L_l L_m}, \qquad L_m \in \beta \quad (11)$$

$$\sum_{L_m \in \beta} (\#\text{ of transitions in } L_m) \cdot \Pi_{L_m} = 1. \quad (12)$$

In our example, this transformation reduces the number of needed equations from 24 to 4. Average energy can then be calculated in terms of these transition lists as follows:

$$\overline{E} = \sum_{L_m \in \beta} \Pi_{L_m} \cdot En(L_m), \quad (13)$$

where $En(L_m)$, the energy consumed in the transition list L_m, is simply the sum of the energy consumed in its composite transitions. Hence, the exact value of $\overline{E}$ can be obtained by simulating each STG transition once!

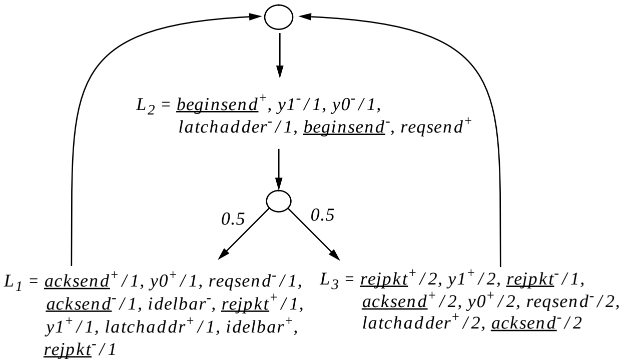

$L_2 = \underline{beginsend}^+, y1^-/1, y0^-/1,$
$latchadder^-/1, \underline{beginsend}^-, reqsend^+$
0.5
0.5
$L_1 = \underline{acksend}^+/1, y0^+/1, reqsend^-/1,$
$\underline{acksend}^-/1, idelbar^-, \underline{rejpkt}^+/1,$
$y1^+/1, latchaddr^+/1, idelbar^+,$
$\underline{rejpkt}^-/1$
$L_3 = \underline{rejpkt}^+/2, y1^+/2, \underline{rejpkt}^-/1,$
$\underline{acksend}^+/2, y0^+/2, reqsend^-/2,$
$latchadder^+/2, \underline{acksend}^-/2$

Figure 5: The collapsed SSTG for the circuit *sbuf-send-ctl*.

7 Relationship to related work

Because the performance of a synchronous circuit is fixed by the period of the global clock, the average circuit power can be obtained from the energy consumed per clock cycle. As a result, the *switching activity* (the expected number of times a gate switches in one clock period) of circuit signals determines the energy and power consumption estimates. The goal of current research is to obtain the most accurate estimate of switching activity while maintaining low complexity.

Many synchronous energy estimation techniques address taking into account correlations among different circuits signals. Correlations among signals can be related temporally (on the same signal at different times) [5], spatially (between different signals at the same time), and spatio-temporal correlations (different signals at different times) [8, 12, 10].

In our circuits, the only correlations that are not taken into consideration by our Markov model are correlations between different input choices. Correlations among the input choices affect the probability of each trace. More specifically, they affect the probability of sequences of transition lists. They do not affect, however, the expected number of occurrences of a particular transition list in an average trace. Since the energy consumed in a circuit depends only on the expected number of occurrences of each transition list, we can conclude that correlations do not affect average energy.

8 Results and Conclusions

We have applied our procedure to four circuits whose STG specifications are modified versions of those used in a large industrial circuit [4] and report the results in Table 1. Since these specification are taken from a benchmark set [7], their use in a large system is unknown. Hence, we had to make some assumptions about their use. First, we assumed that all mutually exclusive choice transitions are equiprobable. Since these circuits have not been layed out, we had to make some assumptions on the load capacitances of the gates in the circuit. Second, we assume each fanout of a signal contributes 25fF to the output load of the gate. The circuit netlist provides the fanout of internal signals while each primary output is assumed to have a total of 4 fanouts. Third, we assume a 5 volt power supply is used.

The column labeled $|R|$ shows the number of signal transitions in the original STG, the column labeled $|A_{Impl}|$ shows the number of circuit signals, the column labeled $|\beta|$ shows the number of different transition lists obtained by serializing and then collapsing the original STG. The last column $\overline{E}$ represents our final estimate for average energy per external signal transition.

The main purpose of the technique proposed in this paper is to facilitate an energy consumption comparison between two implementations of the same STG. Because the complexity of the approach is very low, this technique may guide logic optimizations (such as those described in [1]) to facilitate the synthesis for low power. In addition, when combined with the average number of transitions per high-level operation, the average energy per operation can be obtained. This will enable circuits with different STG specifications for the same application to be compared. Using additional techniques described in [6], we can also estimate the energy consumption per operation for asynchronous circuits that contain both datapath and control circuits.

Average Energy Estimates				
Circuit	$\lvert R\rvert$	$\lvert A_{Impl}\rvert$	$\lvert\beta\rvert$	$\overline{E}$
sbuf-send-ctl	8	17	3	1.08 pJ.
sbuf-send-pkt2	9	17	6	0.806 pJ.
pe-rcv-ifc	11	37	8	1.56 pJ.
pe-send-ifc	10	38	7	1.38 pJ.

Table 1: Calculated average energy estimates.

Acknowledgements

We would like to thank Massoud Pedram and Alice Parker of the University of Southern California and Chris Myers of Stanford University for stimulating discussions on power estimation. This research was funded in part by a Charles Lee Powell Foundation Research Equipment Grant for Young Faculty of the USC School of Engineering.

References

[1] P. A. Beerel. *CAD Tools for the Synthesis, Verification, and Testability of Robust Asynchronous Circuits.* PhD thesis, Stanford University, August 1994.

[2] P. A. Beerel and T. H.-Y. Meng. Automatic gate-level synthesis of speed-independent circuits. In *IEEE ICCAD* Digest of Technical Papers, pages 581–586, 1992.

[3] T.-A. Chu. *Synthesis of Self-Timed VLSI Circuits from Graph-theoretic Specifications.* PhD thesis, Massachusetts Institute of Technology, 1987.

[4] B. Coates, A. Davis, and K. Stevens. The Post Office experience: Designing a large asynchronous chip. *Integration, the VLSI journal*, 15(3):341–366, October 1993.

[5] A. Ghosh, S. Devadas, K. Keutzer, and J. White. Estimation of average switching activity in combinational and sequential circuits. In *Proc. ACM/IEEE Design Automation Conference*, pages 253–259, 1992.

[6] P. Kudva and V. Akella. A technique for estimating power in asyncronous circuits. In *International Symposium on Advanced Research in Asynchronous Circuits and Systems (ASYNC)*, pages 166–175, 1994.

[7] L. Lavagno, K. Keutzer, and A. Sangiovanni-Vincentelli. Algorithms for synthesis of hazard-free asynchronous circuits. In *Proceedings of the 28th ACM/IEEE Design Automation Conference*, 1991.

[8] R. Marcalescu, D. Marcalescu, and M. Pedram. Switching activiy analysis considering spatiotermporal correlations. In *Proc. International Conf. Computer-Aided Design (ICCAD)*, pages 294–299, 1994.

[9] A. J. Martin. Programming in VLSI: from communicating processes to delay-insensitive VLSI circuits. In C.A.R. Hoare, editor, *UT Year of Programming Institute on Concurrent Programming.* Addison-Wesley, 1990.

[10] J. Monteiro, S. Devadas, and B. Lin. A methodology for efficient estimation of switching activity in sequential circuits. In *Proc. ACM/IEEE Design Automation Conference*, pages 12–17,

[11] S. Ross. *Introduction to Probability Models.* Academic Press, 1985.

[12] C.-Y. Tsui, M. Pedram, and A. Despain. Exact and approximate methods for calculating signal and transition probabilities in fsms. In *Proc. ACM/IEEE Design Automation Conference*, pages 18–24, 1994.