Hysteretic Threshold Logic and Quasi-Delay Insensitive Asynchronous Design

Mark Neidengard, *Student Member, IEEE*, and Bradley A. Minch, *Member, IEEE*

***Abstract*—We introduce the class of hysteretic linear-threshold (HLT) logic functions as a novel extension of linear threshold logic, and prove their general applicability for constructing state-holding Boolean functions. We then demonstrate a fusion of HLT logic with the quasi-delay insensitive style of asynchronous circuit design, complete with logical design examples. Future research directions are also identified.**

***Index Terms*—Asynchronous VLSI, digital logic, hysteresis, linear-threshold logic, quasi-delay insensitive (QDI).**

I. Introduction

BOOLEAN logic is the well-established foundation upon which most current digital systems rest. It is both easily manipulated and easily implemented with networks of transistors on chip—its expressive power lies in its simplicity. However, that expressive power has been proven strictly inferior to other classes of logic, in particular linear-threshold (LT) logic [1]. LT logic promises decreased logical complexity for certain computations, and mathematical differences between LT and Boolean logic may result in certain implementation advantages in silicon.

Another feature common to most current digital systems is a global clock used for sequencing operations and coordinating the holding of state. Clocked circuits can be very simple to design, but generating and distributing the clock itself is a task whose difficulty rises in proportion to the clock's frequency and the amount of circuitry it controls. Asynchronous circuits eliminate this difficulty by completely decentralizing the sequencing task and distributing it among all of the functional modules. Synchronization complexity is thereby limited to the more easily solved regime of the fan-in/fan-out of individual modules. Asynchronous methodologies that include delay insensitivity also permit the designer to focus on verifying the logical correctness of the design rather than on the timing of signals, usually resulting in a shorter design time.

In this paper, we establish the theoretical foundations underlying what we believe to be a novel design style, by incorporating a generalization of LT logic into a certain asynchronous circuit style. We also present a small number of elementary complexity results using our theory, and offer a simple theoretical design example. Future publications will describe designs for more complex systems, details of silicon implementation, and silicon performance results.

Manuscript received August 15, 2001; revised March 4, 2002. This paper was recommended by Associate Editor Y. Ismail.

The authors are with the Department of Electrical and Computer Engineering, Cornell University, Ithaca, NY 14853-5401 USA.

Digital Object Identifier 10.1109/TCSI.2002.803360.

After the research described in this paper began, we became aware of work by Theseus Logic, Inc. with certain parallels to our own [2], [3]. In particular, their NULL convention logic (NCL) methodology utilizes threshold gates to detect input arrival and feedback to add hysteresis to those gates. While a "NULL" or reset state is required of any asynchronous circuit to be used more than once [4], our work does not require symbolic completeness via an explicit "Intermediate" state, and does not limit hysteresis to cases where all inputs to an operator must reset between valid states. Our choice of the quasi-delay insensitive (QDI) design methodology lends a different flavor to our circuit derivations as compared to the NCL methodology, but the generality of the theorem presented below does allow individual NCL operators to be expressed in hysteretic LT (HLT) logic.

Throughout this paper, we adopt a convention from complexity literature for describing sets of functions. If G denotes a class of operators (e.g., G = AON for the class of AND, OR, and NOT circuits), Gn is the set of all functions computable by a depth-n network of such operators.

II. Threshold Functions and Hysteresis

A general threshold function F: $T^m \rightarrow S$ can be defined by

$$F = A \cdot Hv\left(-Th + \sum_{i=1}^{n}(w_i \cdot f_i(X)) + B\right) \qquad (1)$$

where Hv is the Heaviside function, $X = (x_1, x_2, \ldots, x_m)$, and A and B are constants. Because Hv can only assume two values, the cardinality of S must be 2. To allow cascading of threshold functions, it is customary to stipulate $T \equiv S$. The sets $\{-1, 1\}$ and $\{0, 1\}$ are common choices for S; to avoid implementation difficulties later (e.g., negative voltages), we shall assume the latter in this paper; hence, we take $A = 1$ and $B = 0$. Th is the *threshold* of F, and, without loss of generality, we shall assume in this paper, that it is in $\mathbf{Q}$. In general, $w_i \in \mathbf{R}$; although, for our purposes, we can require $w_i \in \mathbf{I}$. Again, with a view toward implementability, we define a *directly implementable* threshold function as one in which all $w_i \geq 0$.

Different functional forms of the subfunctions $f_i(X)$ generate different families of threshold functions. The most elementary and well-studied type comprises the LT functions, where $f_i(X) \equiv x_i$. This class of functions forms the basis for conventional neural networks. Our current inquiries are confined to LT logic, although the enhanced power of polynomial threshold logic, in which the $f_i(X)$ are product functions of x_i, shows promise as a future research direction [5].

It has been demonstrated that feedback in LT systems leads to a set of stable states [6], but using that information for sequential logic design has been relatively clumsy compared to production rules or other Boolean formalisms. To separate state-holding behavior from computational topology, we define a *hysteretic threshold* (HT) function by

$$*\left[F' := F;\ F = A \cdot Hv\left(-Th(F') + \sum_{i=1}^{n} (w_i \cdot f_i(X)) + B\right)\right] \tag{2}$$

where $Th(0) \equiv T_{\rm lo}$ and $Th(1) \equiv T_{\rm hi}$, and $T_{\rm lo}$, $T_{\rm hi} \in \mathbf{Q}$. Once again, we take $A = 1$ and $B = 0$. In this context, the sequencing implied by the semicolon models the fact that the feedback is not instantaneous, which is consistent with wire delay or iterative simulation techniques.

Note that threshold functions are a strict subset of HT functions for which $T_{\rm lo} = T_{\rm hi}$. In particular, LT functions are a strict subset of HLT functions.

We now prove by construction that any HT function, and hence any *noninterfering* Boolean function (state-holding or not), can be implemented by threshold functions with a fixed feedback configuration. Noninterfering Boolean functions have *set* and *reset* production rules that are never simultaneously enabled. For convenience, we also define

$$\begin{aligned} S_{\rm gl} &\equiv \min\left(\sum (w_i \cdot x_i) > T_{\rm lo}\right) \\ S_{\rm ll} &\equiv \max\left(\sum (w_i \cdot x_i) < T_{\rm lo}\right) \\ S_{\rm gh} &\equiv \min\left(\sum (w_i \cdot x_i) > T_{\rm hi}\right) \\ S_{\rm lh} &\equiv \max\left(\sum (w_i \cdot x_i) < T_{\rm hi}\right). \end{aligned} \tag{3}$$

Theorem 1 (General Hysteresis Theorem): Any HT function can be implemented by a threshold function with a fixed-weight feedback connection from output to input.

Proof: First, we write HT function F as

$$*\left[F' := F;\ F = Hv\left(\sum_{i=1}^{n} (w_i \cdot f_i(X) - T_o + w_o + H \cdot F')\right)\right] \tag{4}$$

where T_o is a parametric constant. We must show that *hysteresis*, H, and *offset*, w_0, are well-defined. We observe that

$$S_{\rm gl} + w_0 > T_o > H + w_0 + S_{\rm lh} \tag{5}$$

and

$$S_{\rm gh} + H + w_0 > T_o > S_{\rm ll} + w_0 \tag{6}$$

which together yield the constraints that

$$S_{\rm gl} - S_{\rm lh} > H > S_{\rm ll} - S_{\rm gh}. \tag{7}$$

We can always satisfy this inequality, because $S_{\rm gl} > S_{\rm ll}$ and $S_{\rm gh} > S_{\rm lh}$. Next, we define

$$\begin{aligned} S_{\rm gt} &\equiv \min(S_{\rm gl},\ S_{\rm gh} + H) \\ S_{\rm lt} &\equiv \max(S_{\rm ll},\ S_{\rm lh} + H). \end{aligned} \tag{8}$$

We then have that

$$S_{\rm gt} + w_0 > T_o > S_{\rm lt} + w_0. \tag{9}$$

Logical advantage will be maximized when T_o is centered between the two sums, implying that we should take

$$w_0 = T_o - (S_{\rm gt} + S_{\rm lt})/2. \tag{10}$$

Finally, we equate Th in (1) with $T_o - w_0$, which completes the proof. ∎

Corollary: All noninterfering Boolean functions $B(X)$ are in HT2.

Proof: We start by writing B as production rules in disjunctive normal form (DNF) as

$$\begin{aligned} F_{t1}(X) \vee F_{t2}(X) \vee \cdots \vee F_{tu}(X) &\rightarrow B\uparrow \\ F_{f1}(X) \vee F_{f2}(X) \vee \cdots \vee F_{fv}(X) &\rightarrow B\downarrow. \end{aligned} \tag{11}$$

The operator B being noninterfering implies that all $F_t(X)$ are mutually exclusive with all $F_f(X)$. Further, the DNF of B implies all $F_t(X)$ and $F_f(X)$ are strictly conjunctive (i.e., Boolean AND) in X. Now, define $F'_{fi}(X) \equiv \neg(F_{fi}(X))$. AND and NAND are known to be in LT1, hence all $F_t(X)$ and $F'_f(X)$ are in LT1 and thus in HLT1.

Now, consider the function

$$S(X) = \sum_{i=1}^{u} (F_{ti}(X)) + \sum_{i=1}^{v} \left(F'_{fi}(X)\right). \tag{12}$$

When B is state holding, all $F_t(X) = 0$ and $F'_f(X) = 1$; hence, we have that $S(X) = v$. Further, when B is set to 1, all $F'_f(X)$ must remain 1, and some $F_{ti}(X) = 1$; hence we have that $S(X) \geq v + 1$. Likewise, we have that when B is set to 0, all $F_t(X)$ must remain 0 and some $F'_{fi}(X) = 0$; hence, we have that $S(X) \leq v - 1$.

The outputs of $F_t(X)$ and $F'_f(X)$ can be fed into a single HLT element with all $w_i = 1$, $v + 1 > T_{\rm lo} \geq v \geq T_{\rm hi} > v - 1$. Thus, all Boolean functions are in HLT2 and therefore also in HT2. ∎

III. Example Applications of the General Hysteresis Theorem

In conventional Boolean Logic, AND, OR, and NOT are typically taken to be the simplest combinational operators. In the same spirit, the C-element can be said to be the simplest noncombinational (i.e., state-holding) operator. The C-element is normally defined by

$$\begin{aligned} x_1 \wedge x_2 \wedge \cdots \wedge x_n &\rightarrow y\uparrow \\ \neg x_1 \wedge \neg x_2 \wedge \cdots \wedge \neg x_n &\rightarrow y\downarrow. \end{aligned} \tag{13}$$

It is frequently more convenient to compute the dual of this function because of direct implementability concerns in CMOS. Because the C-element is neither purely conjunctive nor purely disjunctive, it cannot be in AON1 (although the inverting C-element can be implemented with one stage of conventional transistor logic). To show that the normal C-element is in HLT1, consider an n-input HLT function with all inputs weighted by unity. Then, from the definition given above, simply set $0 < T_{\rm hi} < 1 \leq n - 1 < T_{\rm lo} < n$. The General Hysteresis Theorem then gives us a concise implementation in which $H = n - 1$ and $w_o = T_o - n + 1/2$.

Now, consider the *asymmetric C-element* given by

$$x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge C \rightarrow y\uparrow$$
$$\neg x_1 \wedge \neg x_2 \cdots \wedge \neg x_n \rightarrow y\downarrow . \tag{14}$$

Boolean C can be replaced by a variety of logic functions, making this a template for precharge, return-to-zero logic. Such operators have been used extensively in such contexts as the Caltech MiniMIPS processor [7]. To see that the asymmetric C-element is also in HLT1, set all $w_i = 2$ and $w_c = 1$, yielding $1 < T_{\rm hi} < 2 \leq 2n < T_{\rm lo} < 2n + 1$. the General Hysteresis Theorem then gives us that $H = 2n$ and $w_o = T_o - 2n - 1/2$.

Next, we consider the *co-asymmetric C-element*, given by

$$x_1 \wedge x_2 \wedge \cdots \wedge x_n \wedge C \rightarrow y\uparrow$$
$$\neg x_1 \wedge \neg x_2 \wedge \cdots \wedge \neg x_n \wedge \neg D \rightarrow y\downarrow . \tag{15}$$

We prove by contradiction that this function is not in HLT1. Assume there exist $T_{\rm lo}$ and $T_{\rm hi}$ satisfying the HLT definition. Then, we have that

$$\sum_{i=1}^{n}(w_i \cdot x_i) + w_c > T_{\rm lo} > \sum_{i=1}^{n}(w_i \cdot x_i) + w_d \Rightarrow w_c > w_d \tag{16}$$

and

$$w_d > T_{\rm hi} > w_c \Rightarrow w_d > w_c \tag{17}$$

which is a contradiction. Therefore, co-asymmetric C-elements are not in HLT1. However, the corollary to the General Hysteresis Theorem guarantees that they are in HLT2.

We note that the General Hysteresis Theorem is not restricted to operators for which $T_{\rm lo} \geq T_{\rm hi}$. Such operators are *stable* in the sense that once they set their output as a function of their inputs, they will not alter the output again as long as the inputs remain constant. If $T_{\rm lo} < T_{\rm hi}$, the possibility exists for the operator to switch its output true and, after the delay implied by the semicolon in (2), immediately back to false again even if the inputs have not changed in the interim. Such unstable operators have *negative* hysteresis which can either be implemented with a negative hysteretic weight, or with a positive hysteretic weight driven by the inverse of the function's output (we will assume the latter). Unstable operators appear in digital logic in such places as pulse generators for latches [8], or in neural networks in the form of pulsed neurons [9].

IV. A Brief Word About CMOS Implementation

The purpose of this paper is to establish a theoretical foundation for HLT circuits and how they may be used to implement QDI systems. We here briefly touch on the subject of HLT circuit implementation in CMOS. We will present further implementation details and experimental results from test silicon in future publications.

Production rules are readily realized in silicon: each guard corresponds directly to a series-parallel network of FETs (p-type for set guards and n-type for reset guards). Logical correctness is independent of such factors as transistor sizing and permuting transistors in series, although those factors may impact electrical performance. Electrical limitations, such as on the maximum allowable number of FETs in series, may make implementing certain production rules unfeasible, requiring us to decompose them into sets of simpler production rules.

Several different schemes exist to implement LT elements. Proposals to implement such elements with multi-input transformers whose turn ratios constitute input weights has existed for some time [10]. A more suitable method for integrated circuits involves a bank of voltage-controlled current sinks fighting a fixed current source. If all the input voltages switch between the same two logic values, the relative strength of each current sink (e.g., its W/L ratio if implemented with FETs) determines its input weight. The current source can readily be set by multiplying the desired threshold by the current drawn by a unity-sized current sink. If the sinks and source are ideal and have infinite output impedance, the voltage at the summing node will reach whichever power supply rail corresponds to the side with the greater current drive. Level shifting can then be employed to recover the high and low logic voltages as the operator's output. This arrangement is a generalization of the pseudo-NMOS NOR arrangement, and has seen considerable use in multiple-valued logic [11].

An alternate strategy based on capacitive voltage division came to prominence via Shibata and Ohmi's work with *neuron-MOS*, or *floating gate*, transistors [12]. Circuit inputs are connected to capacitor top plates, whose common bottom plate drives a CMOS inverter. Assuming that charge on the floating node is fixed, the voltage at the gate is a sum of the input voltages, weighted by the relative capacitor sizes, and offset by the total capacitance times the bottom plate charge. In the ideal case, the inverter compares the weighted sum with its switching threshold and outputs one of the power supply voltages, which can again be level shifted to recover the logic voltages. Unlike the arbitrary threshold current source above, the inverter's switching threshold and initial floating gate charge may be constrained by implementation details (e.g., the inverter's switching threshold is frequently fixed at half the power supply voltage).

While neither of these schemes suffers from major conventional CMOS fanin limitation, series chain limits due to high channel resistance and charge sharing, they are limited by summing junction noise which scales as a function of the number of current sinks or capacitors present.

V. QDI Systems and HLT Logic

A thorough introduction to asynchronous design methodologies is beyond the scope of this paper; several different styles have been described in other publications [13]–[16]. All of these styles arise from various compromises with respect to true delay insensitivity (DI), in which logical sequencing of events is guaranteed regardless of any finite, positive delays in wires and operators. The compromises, in the form of timing assumptions, are necessary, because the class of truly DI systems has been shown to be unacceptably small [4]. The QDI style developed at Caltech makes the minimum timing assumption required to achieve Turing completeness, maintaining very strong theoretical guarantees about proper logical sequencing of events. In

order to inherit that strength, we have adapted the QDI style for constructing our own asynchronous, HLT circuits.

QDI circuits can be compiled from communicating hardware processes (CHP) notation into production rules by identifying and strengthening syntactic guards [17]. These syntactic guards are equally valid for constructing HLT functions, either directly or by converting production rules into HLT functions through the procedure in the corollary to the General Hysteresis Theorem. The enhanced power of LT logic allows direct implementation of certain syntactic guards, such as the majority and parity functions, that would be quite cumbersome to implement with Boolean logic [18].

An operator whose outputs become valid when all its inputs are valid and neutral when all its inputs become neutral is called *weak condition* (WC) [19]. WC operators are the most natural way to syntactically compile most QDI programs, because they require no additional circuitry to implement proper sequencing. However, the resulting production rules may contain unacceptably many conjuncts or disjuncts, prompting the designer to decompose the operator into smaller suboperators each with lower fan-in. Such decomposition increases the complexity (and implementation difficulty) of the overall system. HLT implementations may be subject to different complexity constraints, possibly offering the designer an alternative that avoids this decomposition issue.

Additionally, QDI design assumes that state-holding operators are perfect, holding state indefinitely. It is implicitly expected that state-retaining circuitry (like staticizers) will be added to state-holding production rules upon translation into actual transistor networks. In HLT logic, state preservation is an explicit, not implicit, property of the feedback from output to input. HLT circuits should therefore be easier to logically verify against high-level designs, and less prone to designer oversight.

VI. Design Example: Half-Buffering and Module

In this section, we illustrate these advantages with a design example, based on the standard QDI approach. For the sake of clarity, we shall omit reset circuitry in this example.

One possible CHP description of a pipeline stage performing the AND function is given by

$$*[C!(\text{AND}(A?, B?))] \tag{18}$$

where channels A, B, and C contain data signals t and f and enable signal e. One possible handshaking expansion, utilizing the *half-buffer* reshuffling [20], is given by

$$\begin{aligned}
*[\ & [c_e \wedge a_t \wedge b_f \rightarrow c_f\uparrow \\
& [\!] \ c_e \wedge a_f \wedge b_t \rightarrow c_f\uparrow \\
& [\!] \ c_e \wedge a_f \wedge b_f \rightarrow c_f\uparrow \\
& [\!] \ c_e \wedge a_t \wedge b_t \rightarrow c_t\uparrow \\
&]; (a_e\downarrow \| b_e\downarrow); \\
& [\neg c_e \wedge \neg a_f \wedge \neg a_t \wedge \neg b_f \wedge \neg b_t \rightarrow c_f\downarrow \| c_t\downarrow]; \\
& (a_e\uparrow \| b_e\uparrow); \\
]. &
\end{aligned} \tag{19}$$

Syntactic compilation into production rules yields

$$\begin{aligned}
c_e \wedge a_t \wedge b_f &\rightarrow c_f\uparrow \\
c_e \wedge a_f \wedge b_t &\rightarrow c_f\uparrow \\
c_e \wedge a_f \wedge b_f &\rightarrow c_f\uparrow \\
c_e \wedge a_t \wedge b_t &\rightarrow c_t\uparrow \\
\neg c_e \wedge \neg a_f \wedge \neg a_t \wedge \neg b_f \wedge \neg b_t &\rightarrow c_f\downarrow \\
\neg c_e \wedge \neg a_f \wedge \neg a_t \wedge \neg b_f \wedge \neg b_t &\rightarrow c_t\downarrow \\
c_f \vee c_t &\rightarrow en\downarrow \\
\neg c_f \wedge \neg c_t &\rightarrow en\uparrow
\end{aligned} \tag{20}$$

where $en = a_e = b_e$. In order to achieve direct implementability, the rules for c_f and c_t must be altered, for instance, as follows:

$$\begin{aligned}
c_e \wedge a_t \wedge b_f &\rightarrow \overline{c_f}\downarrow \\
c_e \wedge a_f \wedge b_t &\rightarrow \overline{c_f}\downarrow \\
c_e \wedge a_f \wedge b_f &\rightarrow \overline{c_f}\downarrow \\
c_e \wedge a_t \wedge b_t &\rightarrow \overline{c_t}\downarrow \\
\neg c_e \wedge \neg a_f \wedge \neg a_t \wedge \neg b_f \wedge \neg b_t &\rightarrow \overline{c_f}\uparrow \\
\neg c_e \wedge \neg a_f \wedge \neg a_t \wedge \neg b_f \wedge \neg b_t &\rightarrow \overline{c_t}\uparrow \\
\overline{c_f} &\rightarrow c_f\downarrow \\
\neg\overline{c_f} &\rightarrow c_f\uparrow \\
\overline{c_t} &\rightarrow c_t\downarrow \\
\neg\overline{c_t} &\rightarrow c_t\uparrow .
\end{aligned} \tag{21}$$

Note that $\overline{c_f}$ and $\overline{c_t}$ are state-holding, and will require staticizing when implemented with actual transistors. To implement weak conditioning, the set rules for $\overline{c_f}$ and $\overline{c_t}$ contain five conjuncts, which is most naturally implemented with five P-type transistors in series. In cases where this series-chain length is unacceptable, the standard QDI solution (as used in the MiniMIPS) is to switch from WC logic to precharge logic, as follows:

$$\begin{aligned}
en \wedge c_e \wedge b_f &\rightarrow \overline{c_f}\downarrow \\
en \wedge c_e \wedge a_f &\rightarrow \overline{c_f}\downarrow \\
en \wedge c_e \wedge a_t \wedge b_t &\rightarrow \overline{c_t}\downarrow \\
\neg en \wedge \neg c_e &\rightarrow \overline{c_f}\uparrow \\
\neg en \wedge \neg c_e &\rightarrow \overline{c_t}\uparrow \\
\overline{c_f} &\rightarrow c_f\downarrow \\
\neg\overline{c_f} &\rightarrow c_f\uparrow \\
\overline{c_t} &\rightarrow c_t\downarrow \\
\neg\overline{c_t} &\rightarrow c_t\uparrow \\
a_t \vee a_f &\rightarrow av\downarrow \\
\neg a_t \wedge \neg a_f &\rightarrow av\uparrow \\
b_t \vee b_f &\rightarrow bv\downarrow \\
\neg b_t \wedge \neg b_f &\rightarrow bv\uparrow \\
av \wedge bv &\rightarrow iv\downarrow \\
\neg av \wedge \neg bv &\rightarrow iv\uparrow \\
\neg\overline{c_f} \vee \neg\overline{c_t} &\rightarrow ov\uparrow \\
\overline{c_f} \wedge \overline{c_t} &\rightarrow ov\downarrow \\
iv \wedge ov &\rightarrow en\downarrow \\
\neg iv \wedge \neg ov &\rightarrow en\uparrow .
\end{aligned} \tag{22}$$

This circuit reduces the maximum number of conjuncts in all set rules to two. This reduction is purchased with additional circuitry to detect the neutrality (and validity) of the inputs separately from the operators that generate the outputs. While this transformation allows us to simplify the $\overline{c_f}$ reset somewhat, the overall circuit now has four extra operators, three of which are state holding, and we will need to add staticizers. Such circuits are considerably harder to design, build, and test than are their weak-condition counterparts.

Now, consider the following HLT compilation:

$$\begin{aligned} 2*c_e + a_f + b_f > 2.5 &\rightarrow c_f \uparrow \\ 2*c_e + a_f + b_f < 0.5 &\rightarrow c_f \downarrow \\ c_e + a_t + b_t > 2.5 &\rightarrow c_t \uparrow \\ c_e + a_t + b_t < 0.5 &\rightarrow c_t \downarrow \\ a_f + a_t + b_f + b_t + c_f + c_t > 2.5 &\rightarrow en \downarrow \\ a_f + a_t + b_f + b_t + c_f + c_t < 0.5 &\rightarrow en \uparrow . \end{aligned} \tag{23}$$

Observe that the signal en is logically inverted relative to c_f and c_t. For this paper, we follow standard practice in the LT literature of considering this inversion to be "free."

Like the precharge logic production rules just given, this circuit detects input and output validity/neutrality separately from the actual output generation circuitry. However, the set and reset rules are entirely symmetrical, and the output operators are actually weak-condition. By further exploiting the properties of LT logic and using the fact that the t and f signals are mutually exclusive true, we arrive at a form which uses only WC operators, analogous to the first weak-condition example

$$\begin{aligned} 2*a_f + 2*b_f + a_t + b_t + 2*c_e > 4.5 &\rightarrow c_f \uparrow \\ 2*a_f + 2*b_f + a_t + b_t + 2*c_e < 0.5 &\rightarrow c_f \downarrow \\ c_e + a_t + b_t > 2.5 &\rightarrow c_t \uparrow \\ c_e + a_t + b_t < 0.5 &\rightarrow c_t \downarrow \\ c_f + c_t > 0.5 &\rightarrow en \downarrow \\ c_f + c_t < 0.5 &\rightarrow en \uparrow . \end{aligned} \tag{24}$$

For instance, suppose that we are constrained to set $Th = 1/2 \cdot \sum_{i=0}^{m} w_i$ by a floating-gate transistor implementation. Substituting that choice into the General Hysteresis Theorem, we arrive at the following HLT circuit:

$$\begin{aligned} c_f &:= 2*a_f + 2*b_f + a_t + b_t + 2*c_e + 4*c_f + 3 > 7.5 \\ c_t &:= c_e + a_t + b_t + 2*c_t > 2.5 \\ en &:= c_f + c_t \leq 0.5. \end{aligned} \tag{25}$$

There are now only three HLT operators, compared to three conventional operators and two inverters in the weak-condition Boolean case, or seven conventional operators and two inverters required by in the precharge Boolean case. Recall that both Boolean circuits will also require staticizers for all stateholding nodes, which are unnecessary for the HLT circuits. Also observe the increased symmetry in the HLT set and reset rules relative to those in the Boolean case. All these properties make HLT circuits both easier to design and to debug.

Note that additional constraints may exist for HLT elements, such as an upper limit on the maximum allowable fan-in m. Such restrictions would require decomposing large operators into cascades of simpler operators. Researchers have reported examples of successful silicon implementation styles for LT circuits with fan-ins above 16 [21], so circuits like the ones presented in this section should be implementable directly as given.

VII. Conclusion

We have presented an alternative to Boolean logic that interfaces cleanly with a particular asynchronous design methodology. The General Hysteresis Theorem actually extends beyond this particular methodology, and could potentially allow HLT circuits into any digital design style meeting its (weak) requirements. Examining the ramifications of such synergy will be a direction for ongoing research.

What we have not yet presented is evidence that HLT functions can be implemented cleanly and efficiently in silicon. We are in the process of testing current-summing and capacitive voltage-summing HLT systems with no more than two CMOS devices between the power supply rails, capable of rapid switching speeds and very low-voltage operation. The results of these investigations will be the subject of future publications.

Acknowledgment

The authors would like to acknowledge the pioneering work of Dr. A. Martin and his research group at the California Institute of Technology in the area of asynchronous circuit design, which served as the inspiration for this research; Dr. R. Manohar, himself a former member of Dr. Martin's group, for valuable insight and advice on silicon implementation details; and the MOSIS program for providing silicon fabrication services in support of this research.

References

[1] K. Siu and J. Bruck, "Neural computation of arithmetic functions," *Proc. IEEE*, vol. 78, pp. 669–1675, Oct. 1990.

[2] K. M. Fant and S. A. Brandt, "NULL Convention Logic: A complete and consistent logic for asynchronous digital circuit synthesis," in *Proc. 1996 IEEE Int. Conf. Application-Specific Systems, Architectures, and Processors*, 1996, pp. 261–273.

[3] G. E. Sobelman and K. Fant, "CMOS circuit design of threshold gates with hysteresis," in *Proc. IEEE Symp. Circuits and Systems*, vol. 2, 1998, pp. 61–64.

[4] A. J. Martin, "The limitations to delay-insensitivity in asynchronous circuits," in *Proc. 6th MIT Conf. Advanced Research in VLSI.* , 1990, pp. 263–278.

[5] J. Bruck, "Harmonic analysis of polynomial threshold functions," *SIAM J. Discr. Math.*, vol. 3, no. 2, pp. 168–177, 1990.

[6] J. Bruck and V. P. Roychowdhury, "On the number of spurious memories in the Hopfield Model," *IEEE Trans. Inform. Theory*, vol. 36, pp. 393–397, Mar. 1990.

[7] A. J. Martin *et al.*, "The design of an asynchronous MIPS R3000," in *Proce. 17th Conf. Advanced Research in VLSI*, 1997, pp. 164–181.

[8] J. M. Rabaey, *Digital Integrated Circuits: A Design Perspective*. Englewood Cliffs, NJ: Prentice-Hall, 1996.

[9] S. Churcher, A. F. Murray, and H. M. Reekie, "Pulse-firing vlsi neural circuits for fast image pattern recognition," in *Proc. Conf. VLSI for Artificial Intelligence and Neural Networks*, 1991, pp. 235–244.

[10] M. Karnaugh, "Pulse-switching circuits using magnetic cores," *Proc. IRE'95*, vol. 43, pp. 570–584, May 1955.

[11] Z. G. Vranesic, K. G. Smith, and A. Druzeta, "Electronic implementation of multiple-valued logic networks," in *Proc. 1974 Int. Symp. Multiple Valued Logic*, 1974, pp. 59–77.

[12] T. Shibata and T. Ohmi, "A functional MOS transistor featuring gate-level weighted sum and threshold operations," *IEEE Trans. Electron Devices*, vol. 39, pp. 1444–1455, June 1992.

[13] A. J. Martin, "Compiling communicating processes into delay-insensitive vlsi circuits," *Distrib. Comput.*, vol. 1, no. 4, pp. 226–234, 1986.
[14] C. H. van Berkel and R. W. J. J. Saejis, "Compilations of communicating processes into delay-insensitive circuits," in *Proc. 1988 IEEE Int. Conf. Computer Design: VLSI in Computers and Processors*, 1988, pp. 157–162.
[15] A. Takamura, M. Kuwako, M. Imai, T. Fujii, M. Ozawa, I. Fukasaku, Y. Ueno, and T. Nanya, "Titac-2: An asynchronous 32-bit microprocessor based on scalable-delay-insensitive model," in *Proc. 1997 IEEE Int. Conf. Computer Design: VLSI in Computers and Processors*, 1997, pp. 288–294.
[16] K. Stevens, S. Rotem, S. M. Burns, J. Cortadella, R. Ginosar, K. Kishinevsky, and M. Roncken, "Cad directions for high performance asynchronous circuits," in *Proc. 1999 Design Automation Conf.*, 1999, pp. 116–121.
[17] R. Manohar, T. Lee, and A. J. Martin, "Projection: A synthesis technique for concurrent systems," in *Proc. 5th Int. Symp. Advanced Research in Asynchronous Circuits and Systems*, 1999, pp. 125–134.
[18] R. C. Minnick, "Linear-input logic," *IRE Trans. Electronic Computers*, no. <AU: ISSUE NO:?>, 1961.
[19] C. Mead and L. Conway, *Introduction to VLSI Systems*. New York: Addison Wesley, 1980.
[20] U. V. Cummings, A. M. Lines, and A. J. Martin, "An asynchronous pipelined lattice structure filter," in *Proc. Int. Symp. Advanced Research in Asynchronous Circuits and Systems*, 1994, pp. 126–133.
[21] Y. Leblebici, F. K. Gurkaynak, and D. Mlynek, "A compact 31-input programmable majority gate based on capacitive threshold logic," in *Proc. 1998 IEEE Int. Symp. Circuits and Systems*, vol. 2, 1998, pp. 105–108.

Mark Neidengard (S'94) earned the B.S. and M.S. degrees in computer science from the California Institute of Technology (Caltech), Pasadena, in 1997 and 1998, respectively, and the Ph.D. degree in electrical and computer engineering (focusing on extending QDI formal techniques to encompass exotic logic styles) from Cornell University, Ithaca, NY in 2002, under Dr. B. Minch.

While at Caltech, he participated in the MiniMIPS asynchronous microprocessor project under Dr. A. Martin, and was a Student Operator, Assistant System Administrator, and Programmer for the Center for Advanced Computing Research.

Bradley A. Minch (S'89–M'97) received the B.S. degree in electrical engineering with distinction from Cornell University, Ithaca, NY, in May 1991 and the Ph.D. degree in computation and neural systems from the California Institute of Technology, Pasadena, in June 1997.

He is currently an Assistant Professor in the School of Electrical and Computer Engineering at Cornell University. His research interests include the analog and digital integrated circuit design, translinear circuits, log-domain filters, and adaptive floating-gate MOS circuits.

Dr. Minch received the IEEE Electron Devices Society's Paul Rappaport Award in 1996. He is a Member of the Tau Beta Pi, Eta Kappa Nu, and Phi Kappa Phi honor societies.