Single Electron Tunneling Delay Insensitive and Fluctuation Based Computation Paradigms and Circuits

Saleh Safiruddin, and Sorin Dan Cotofana, Senior Member IEEE
Computer Engineering Laboratory, Delft University of Technology
Delft, The Netherlands

Ferdinand Peper, Member IEEE
Nano ICT Group - National Institute of Information and Communications Technology
Kobe, Japan

***Abstract*— This paper presents two computational paradigms in Single Electron Tunneling (SET) technology that address the unpredictability caused by the stochastic nature of tunneling, which is a widely recognized weakness of SET. The first paradigm is based on circuits that are insensitive to their signal delays. Called Delay-Insensitive, these circuits work without a clock, and are thus free of any timing restrictions that may interfere with the SET stochastic behavior. The second paradigm goes one step further and allows its signals to be affected by fluctuations, which are expected to be a major problem in nanometer-scale SET circuits. We discuss the SET-based implementations of the basic building blocks required for each paradigm and the techniques for constructing larger circuits from them, taking into account the pros and cons with respect to circuit complexity (area) and technological applicability. Our analysis clearly indicates that, if utilized in the right way, SET probabilistic behavior can become an asset in building predictable SET based circuits and systems, rather than an impediment.**

Index Terms **— Single Electron Tunneling, Brownian circuits, delay insensitive circuits**

I. Introduction

It is generally expected that current semiconductor technologies, i.e., CMOS, cannot be pushed beyond a certain limit because of problems arising in the area of power consumption and scalability. A promising alternative to CMOS is Single Electron Tunneling (SET) technology [1], which has the potential of performing computation with lower power consumption than CMOS and which is scalable into the nanometer region and beyond [2].

SET technology is fundamentally different from CMOS as it is based on tunneling of electrons. This difference opens up avenues for new computational paradigms, e.g., Single Electron Encoded Logic, Electron Counting, [3], [4], [5], [6], which try to effectively use the basic SET properties. Theoretical results on the area and delay complexity of arithmetic operations using those new paradigms indicate great potential. However, electron tunneling is stochastic in nature. Tunneling through a junction becomes possible when the junction's actual voltage V_j exceeds the junction's critical voltage $V_c = \frac{q_e}{2(C_j+C_e)}$, where $q_e = 1.602 \cdot 10^{-19}C$, C_j is the capacitance of the junction, and C_e is the capacitive value of the remainder of the circuit as seen from the junction. The delay of such circuits cannot be analyzed in the traditional sense. Instead, for each transported electron one can describe the switching delay as $t_d = \frac{-\ln(P_{error})q_e R_t}{|V_j|-V_c}$, where R_t is the junction resistance and P_{error} is the chance that the desired charge transport has not occurred after t_d seconds. This probabilistic delay complicates the direct utilization of SET-based computation in building synchronously timed computation units.

In this paper we investigate two classes of SET based architectures and circuits that aim to exploit, rather than suffer from, the purported weaknesses of SET technology, i.e., its probabilistic delay, and the indeterministic behavior induced by background charge, temperature, etc.

First, we combine SET with Delay-Insensitive Circuits (DICs), which are a type of asynchronous circuit that are robust to signal delays in wires and operations. We focus on a specific type of DIC, namely circuits in which signals have a discrete character. Called *token-based*, this type of signal is implemented in a natural way as individual electrons in the framework of SET. Due to their delay-insensitivity, DICs have the potential to be robust against the probabilistic delays occurring in SET circuits. Previous work [12] has determined some universal sets of basic primitive modules from which any DIC can be built. We choose primitive modules that are in many such universal sets and propose building blocks that implement these modules in terms of SET technology. We demonstrate by computer simulations that the SET building blocks function correctly, and that they can be used as a basis to construct larger circuits, which are tolerant to delay uncertainties. In the SET designs of the building blocks we utilize the tunneling effect and the ability of SET technology to control individual electrons. We also propose circuit topologies such that the movements of a positive charge around the circuit follow certain states, depending on the input values and input combinations. We then extract these signals and combine them to form the desired output.

Second, we investigate ways to exploit the signal fluctu-

978-1-4244-2553-2/08/$20.00

ations in SET logic and arithmetic circuits. This approach finds its ancestor in an earlier proposal that employs signal fluctuations actively in a simulated annealing scheme for Boltzman machines [9]. Being focused on neural network implementations, however, this proposal has found no follow-up in the context of traditional arithmetic circuits. Recently, two building blocks operating on a similar principle have been proposed in the context of the so-called *Brownian circuits* [8], [7]. The idea of Brownian circuits is to use fluctuations to guide signals through a circuit. That is, fluctuations drive a search process through a Brownian maze formed by the topology of the circuit [8]. The two proposed blocks have been shown to form a universal set from which any desired functionality can be constructed [7]. We propose SET-based circuit topologies for this class of circuits and verify via simulation that they deliver the expected behavior and that they can be connected in order to build more complex circuits.

The remainder of the paper is organized as follows: In Section II we briefly introduce the basic set of Delay Insensitive Circuit we have chosen for implementation. Section III presents circuit topologies and simulation results for the considered DIC primitives. In Section IV we introduce a buffering technique that can be utilized in conjunction with the proposed primitives to build larger DICs and an example circuit. Section V introduces circuit topologies for Brownian circuits basic building blocks while Section VI discusses the implementation of larger fluctuation based. Finally, in Section VII we draw some conclusions and discuss future work.

II. Building Blocks for Delay Insensitive Circuits

The behavior of the universal set of DI primitives can be better described in terms of inputs and outputs transitions rather than in terms of absolute logic levels as is common in conventional digital logic. For the following delay-insensitive components the environment has to wait for a pending output transition to occur before allowing new input signals to the component. The DI primitives are as follows:

- Merge(*a,b ; c*) - functions as a conventional *XOR* gate, as a transition of either *a* or *b* causes a transition of *c* and only when *c* transitions can new transitions of *a* or *b* be accepted;
- C-Element(*a,b ; c*) - is a state holding element. It waits until both *a* and *b* have undergone a transition before causing a transition of *c*, and is only then ready to accept new transitions of *a* and *b*;
- Sequencer(*a,b,c ; ao,bo*) - here *a* and *b* are inputs and *c* is the control. A transition of *a* and *c*, not necessarily in that order, causes *ao*, and a transition of *b* and *c* causes *bo*. A transition of *a* and *b* and then of *c* causes a transition of *ao* or *bo*, chosen arbitrarily, and a further transition of *c* causes the transition of *ao* or *bo* depending whether *ao* or *bo* transitioned previously, respectively.
- Tria(*a,b,c ; d,e,f*) - is a further extension with similar functionality where *a* and *b* cause *d*, *b* and *c* cause *e* and *a* and *c* cause *f*; simultaneous signals on all three inputs should be prohibited by the environment.
- Toggle(*a ; b,c*) - functions like the conventional toggle flip-flop. A transition on *a* causes a transition on *b* and the next transition of *a* causes a transition on *c* and so on.

In [12] and [13] it was demonstrated that various universal sets can be defined based on these primitives.

III. DIC Implementations

To ensure that the charge of an electron is quantized on each specific island the tunnel junctions must have a sufficiently high tunneling resistance, so that the charging energy, also called the Coulomb energy, dominates over the quantum charge fluctuations [10]. Thus, in the following implementations, all tunneling junctions have a resistance of $100k\Omega$. When a tunnel junction appears in the figure it is designated with a 'J' and a number, and when the circuit parameters are described the capacitance of the tunnel junction is referred to with a 'C' and the same number. An input signal designated with an appended ˆ means that an inverted signal is used, for example V_aˆ . This inverted signal is produced with an inverting buffer being applied to the original signal, as proposed in [4] and graphically depicted in Figure 1. The source voltage V_s is set at 16 mV and logic '1' is encoded as 16mV and logic '0' as 0 mV [4]. All simulations were done using SIMON 2.0 software [11].

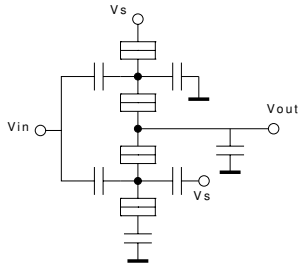

Fig. 1. Static inverting buffer

A. Merge

Figure 2 presents a SET circuit topology which provides the required behavior for a *Merge*. The input signals are V_a and V_b and the output signal is V_c. Since the functionality is equivalent to that of an *XOR* gate an *AND-OR-INVERT* implementation is the most appropriate. The *AND* gate and the *OR* gate are built by utilizing the generic design of the linear threshold gate described in [5]. The circuit comprises of 7 tunneling junctions and 17 capacitors. The circuit parameters correspond to those in [5], specifically, $C_a = C_b = C_t = 0.5aF; C_{s1} = 9.5aF; C_{s2} = 10.5aF; C_g = 10aF; C_1 = C_2 = C_3 = 0.1aF$. The inverter is implemented using the inverting buffer in [5].

The circuit operates as follows. When V_a and V_b are low V_aˆ and V_bˆ are high causing an electron to tunnel through $J2$ leaving a positive charge on $n2$. This in turn causes an electron to tunnel through $J3$ leaving a positive charge on $n3$. This is inverted and so V_c becomes low. If V_a or V_b undergoes

a transition then the voltage of $J2$ (and $J1$) becomes lower than the critical voltage causing the electron to tunnel back leaving no charge on $n2$ (and $n1$). This reduces the voltage over $J3$ to under the critical voltage and the electron tunnels back leaving no charge on $n3$. This value is complemented afterwards by the output inverter thus V_c becomes high. If the second signal also undergoes a transition then the voltage over $J1$ becomes higher than the critical voltage causing an electron to tunnel. This causes an electron to tunnel through $J3$ leaving a positive charge on $n3$ corresponding to a low output. If subsequently one of the signals transitions again, going low this time, the circuit goes into the previous state with no charges on $n1$, $n2$, and $n3$. If after that the other signal also transitions the circuit returns to the original charge neutral state.

The simulation results are displayed in Figure 3 and one can observe that the circuit exhibits the expected *Merge* behavior.

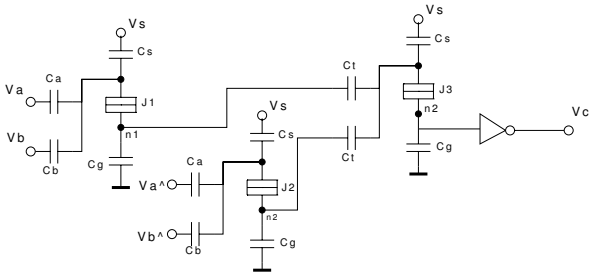

Fig. 2. Merge circuit diagram

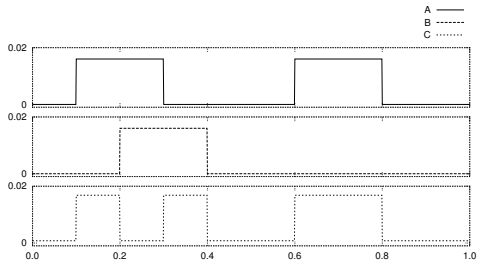

Fig. 3. Simulation results for the Merge

B. C-Element

Figure 4 presents a SET circuit topology that is designed to provide the required behavior for a *C-Element*. The input signals are V_a and V_b and the output signal is V_c. The circuit comprises of 2 tunneling junctions and 4 capacitors. The circuit parameters are as follows: $C_a = C_b = 3aF; C_s = C_g = C_1 = 10aF; C_2 = 0.1aF$.

The circuit operates as follows. If both input signals are low the circuit is in its initial neutral state and no charge transport occurs. If either V_a or V_b undergoes a transition the voltage over $J2$ increases but does not become critical yet. If the second signal also goes high then the voltage over $J2$ becomes larger than its critical voltage and an electron tunnels from $n3$ to $n2$. This causes the voltage over $J1$ to exceed the critical voltage and the electron further tunnels from $n2$ to $n1$. V_c is now high, its value being determined by the charge on $n3$. Subsequently, when V_a or V_b goes low the voltage over $J1$ decreases, while the voltage over $J2$ increases but not above the critical voltage. Only when the second signal also goes low does the voltage over $J2$ exceed the critical voltage and an electron is transported from $n2$ to $n3$ leaving $n2$ positively charged. This charge on $n2$ causes the voltage over $J1$ to exceed the critical voltage and the electron residing on $n1$ tunnels through into $n2$. The circuit is then back in its initial neutral state.

The simulations results can be seen in Figure 5. As it can be observed the proposed SET circuit produces the correct *C-Element* functionality. The output goes high only when both input signals have gone high, and subsequently goes low only when both input signals have gone low.

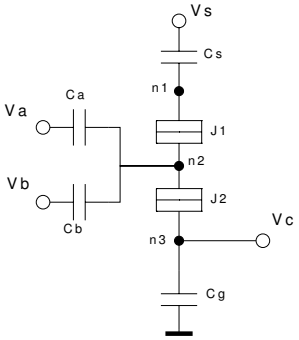

Fig. 4. C-Element circuit diagram

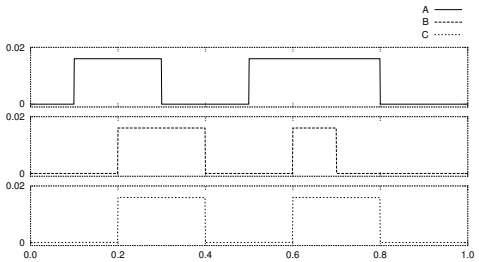

Fig. 5. Simulation results for the C-Element

C. Sequencer

Figure 6 presents a SET circuit topology which provides the required behavior for a *Sequencer*. The input signals are V_a, V_b, and V_c and the output signals are V_{ao} and V_{bo}. The circuit comprises of 10 tunneling junctions and 20 capacitors. The circuit parameters are follows: $C_a = C_b = C_c = 3aF; C_1 = C_3 = C_5 = C_7 = 10aF; C_2 = C_4 = C_6 = C_8 = 0.1aF; C_g = C_s = 10aF; C_{s1} = C_{s2} = 10.5aF; C_9 = C_{10} = 0.1aF; C_t = 0.5aF$.

The circuit is made up of 4 *C-Element* structures and operates as follows. With all inputs low the circuit is in a neutral state with no transported charge. If the two inputs V_a and V_c go high an electron tunnels through $J1 - J2$ and propagates from node $n2$ to $n1$ due to the transport mechanism built in such a *C-Element*, leaving a positive charge on $n2$ and a negative one on $n1$. From here, either V_a and V_c would go low causing the electron to go back to $n1$ resulting in the neutral state, or V_b would go high and V_c would go low meaning $V_c\hat{}$ going high resulting in an electron tunneling through $J3 - J4$. With $n2$ positive and V_b and $V_c\hat{}$ high an electron propagates through $J3 - J4$ resulting in no charge on $n2$ and a positive one on $n3$. Subsequently, with inputs V_a and V_b high and V_c low, either V_a would go low and $V_c\hat{}$ low, or V_b would go low and $V_c\hat{}$ low resulting in an electron propagating from $n4$ or $n2$ into $n3$ through $J7 - J8$ or $J3 - J4$ respectively, leaving a positive charge on $n4$ or $n2$. From there the aforementioned mechanisms would apply.

The net effect is thus that the positive charge moves around the circuit depending on input signal transitions, until it reaches node $n1$ where the circuit returns to it's initial neutral state. The nodes $n2$ and $n3$ correspond to V_{ao}, which is evaluated by an *OR* gate implemented using a linear threshold gate as introduced in [5] and taking them as inputs. The nodes $n3$ and $n4$ correspond to V_{bo}. If, from the neutral state V_a and V_b go high and then V_c, then one of the propagation routes is chosen arbitrarily and either node $n2$ or $n4$ becomes positively charged. Subsequently, the circuit is in the state as described above after the first propagation and the extra signal which went high is pending until the transtion of V_c.

The simulation results are displayed in Figure 7. As one can observe from the first segment of the simulation, which corresponds to the first V_a pulse, the basic *Sequencer* functionality is delivered. Input V_a going high or low followed by or preceded by input V_c going high or low produces a toggling of V_{ao}. The same holds for input V_b. In the second part of the simulation we can see the additional *Sequencer* functionality showing that if V_a and V_b go high before V_c going high two subsequent transitions of V_c cause a toggling of V_{ao} and V_{bo} in an arbitrary order.

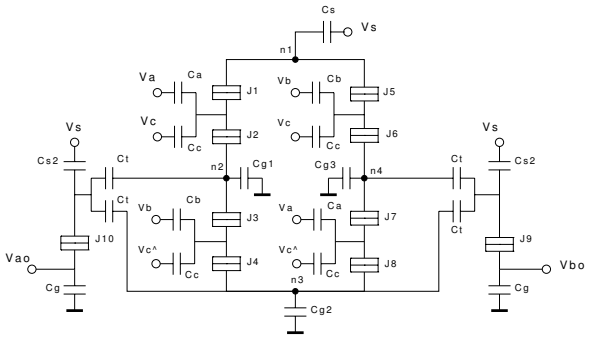

Fig. 6. Sequencer circuit diagram

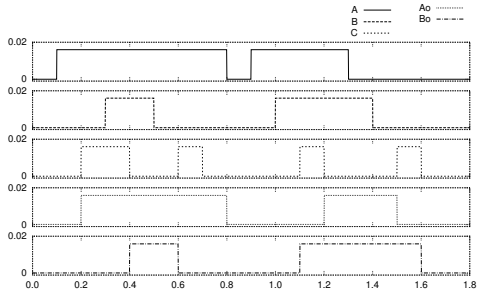

Fig. 7. Simulation results for the Sequencer

D. Tria

Figure 8 presents a SET circuit topology that provides the required behavior for a *Tria*. The input signals are V_a, V_b, and V_c and the output signals are V_d, V_e, and V_f; all input gate capacitances are Ci. The circuit comprises of 27 tunneling junctions and 50 capacitors. The circuit parameters are as follows: $C_i = 3aF; C_g = C_s = 10aF; C_1 = 10aF; C_2 = 0.1aF; C_t = 0.21aF; C_3 = 0.1aF; C_{s2} = 10.5aF$.

The circuit operates in a very similar way to the *Sequencer*. An electron moves to node $n1$ from $n2$, $n3$ or $n4$ depending on which input signals go high. Only if the same two signals then go low again does the electron propagate back. This is in effect the *C-Element* functionality. Subsequently, the node which has the positive charge has three nodes from which an electron may potentially arrive, depending on which input signals undergo transitions. Thus, each node has a pathway to other 3 nodes opening up only due to transitions which are possible if that node has been reached. As with the *Sequencer*, the positive charge moves from node to node depending on the transitions made by the input signals. Each node contributes to one or more of the outputs as follows. $n2$, $n5$, $n6$, and $n8$ correspond to output V_d and a 4-input *OR* gate is used to combine these node outputs to get the output V_d. Similarly, output V_e is formed from $n3$, $n5$, $n7$, and $n8$ and output V_f is formed from $n4$, $n6$, $n7$, and $n8$.

The simulation results are displayed in Figure 9. The first two transitions are of V_a and V_b going high, and $n2$ signals the output V_d. Subsequently, V_b goes low and V_c goes high, and $n5$ signals outputs V_e and V_d. The rest of the simulation proceeds in a similar fashion and clearly indicates that the topology in Figure 8 is able to deliver the *Tria* functionality.

E. Toggle

Figure 10 presents a SET circuit topology which provides the required behavior for a *Toggle*. The input signal is V_a and the output signals are V_b and V_c. The circuit comprises of 10 tunneling junctions and 17 capacitors with the following values: $C_a = 4aF; C_r = 3aF; C_b = 3.6aF; C_{g1} = 11aF; C_{g2} = 13aF; C_{g3} = 18aF; C_s = C_g = 10aF; C_{s2} = 10.5aF; C_t = 0.5aF; C_1 = C_3 = C_5 = C_7 = 10aF; C_2 = C_4 = C_6 = C_8 = 0.1aF$. The *Toggle* is constructed using 4 *C-Element* structures but with only one input and which only propagate electrons in one direction. The circuit operates as follows. It

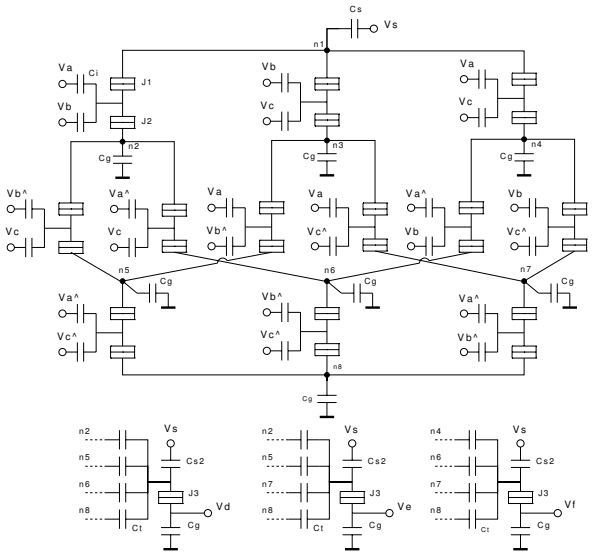

Fig. 8. Tria circuit diagram

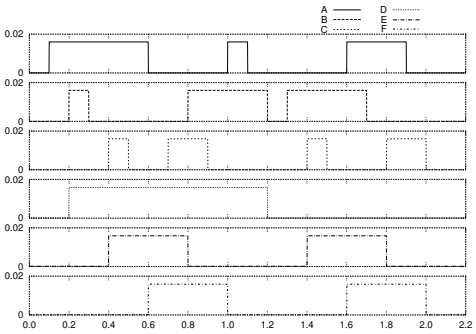

Fig. 9. Simulation results for the Tria

starts in a neutral state with the input low and with no charge transported. If V_a goes high an electron propagates from $n2$ to $n1$ through $J1 - J2$, leaving a positive charge on $n2$. The electron doesn't return if V_a goes low, as the critical voltage of either $J1$ or $J2$ is not exceeded. When V_a goes low, $V_a\hat{}$ goes high and an electron is transferred from node $n3$ to $n2$ through $J3 - J4$. With a positive charge now on $n3$ if V_a goes high again an electron tunnels through $J5 - J6$ from $n4$ to $n3$. Finally $V_a\hat{}$ going high causes the electron residing on $n1$ to tunnel through $J7 - J8$ to neutralize the positive charge on $n4$ and bring the circuit back to its initial neutral state. The charge on node $n2$ corresponds to the first pulse of V_a and the charge on node $n3$ corresponds to the signal till the second pulse, and therefore these two nodes contribute together to the first output of the *Toggle*, V_b. These nodes are therefore combined using a linear threshold *OR* gate[5]. Nodes $n3$ and $n4$ correspond to output V_c .

The simulation results are displayed in Figure 11. As is expected, output V_b toggles for each positive edge of the input and output V_c toggles for each negative edge of the input. After the second pulse the circuit is in its initial neutral state again and none of the outputs are high. The cycle then repeats itself.

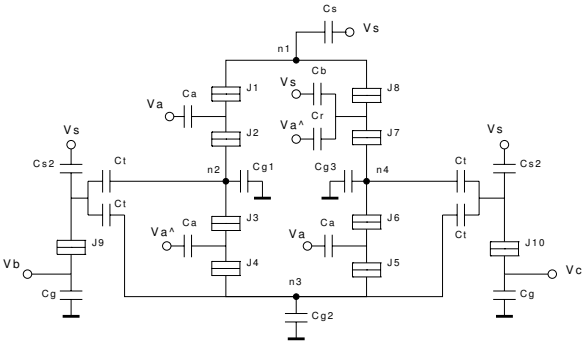

Fig. 10. Toggle circuit diagram

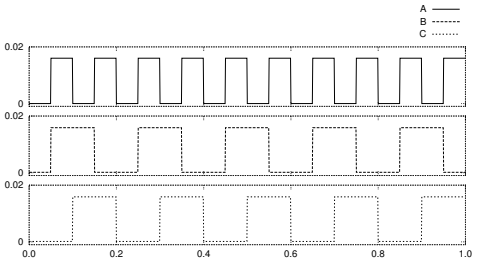

Fig. 11. Simulation results for the Toggle

IV. Networks of DIC Building Blocks

A. Buffering Between Passive Networks

Due to the passive nature of the proposed building blocks feeback problems arise when attempting to connect the blocks to form circuits with a particular desired functionality. This problem was previously discussed in the context of Single Electron Encoded Logic (SEEL) circuits in [4] and a non-inverting buffer was proposed to be utilized as a front-end to SEEL gates with an input gate capacitance of $0.5aF$. In the case of SET based DICs this buffer can be directly utilized only for the *Merge* block. The proposed implementations of the *C-Element*, *Sequencer*, and *Tria* all have input gate capacitances of $3aF$ and the *Toggle* has $4aF$ thus the SEEL specific buffer cannot be utilized as it doesn't function correctly in these cases. We propose an extension for this buffer such that it can properly drive the higher input gate capacitances, as depicted in Figure 12. The circuit parameters are as follows: $C_1 = 10aF; C_2 = 0.1aF; C_i = 1aF; C_o = 0.5aF; C_g = 10aF$. The design is essentially the same except that there are 6 islands instead of one, with each island connected to the input of the next block. The total input gate capacitance is thus the required $3aF$. This proposal is scalable and by varying the number of islands connected to the input various capacitances

can be accommodate. In the case of the *Toggle* which has $4aF$ input capacitance the network in Figure 12 is extended with two more islands.

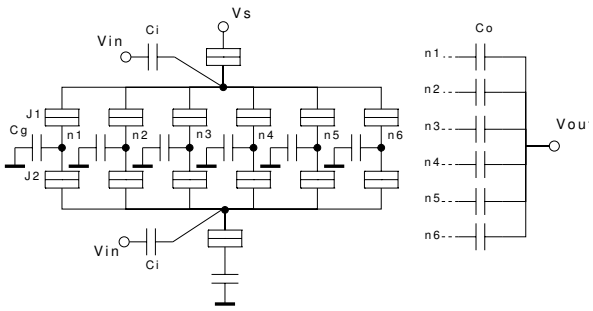

Fig. 12. Extended buffer circuit diagram

B. Example: QR42

With the proposed buffering technique, networks of delay insensitive circuit building blocks can be constructed. As an example we present the *Quick Return Conversion Unit*, or *QR42*, which repeatedly converts a four-phase handshake from it's left side environment to a two-phase handshake with it's right side environment [13]. The construction is shown in Figure 13. Buffers are placed between the DI components, depicted as blocks labelled *B*. With *a* and *d* as inputs and *b* and *c* as outputs the *QR42* functions as follows: a transition of *a* causes *b* and *c* to transition. Subsequently, a transition of *a* and *d* causes *b* to transtion. The results of the simulation can be seen in Figure 14.

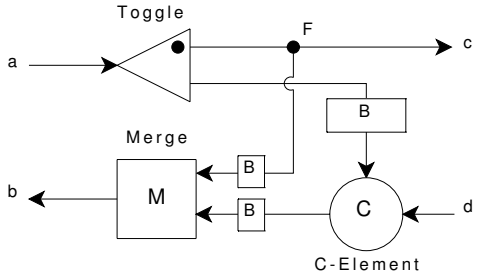

Fig. 13. QR42 block diagram

V. Building Blocks for Fluctuation Based Calculations

Circuits in which signals can fluctuate have a fascinating advantage over circuits with more "straight" signals: the fluctuation can be exploited to backtrack signals out of deadlock situations, thereby adding powerful functionality to the circuit for free [8]. It turns out that in this case merely two primitives suffice to form a universal set [7]:

- *Hub* - contains three wires that are bidirectional. There will be at most one signal at a time on any of the *Hub*'s wires, and this signal can randomly walk between any of the wires and off to other primitives due to its fluctuations.
- *Conservative Join* - has two input wires and two output wires. The *CJoin* can be interpreted as a synchronizer of two signals passing through it. Signals may fluctuate on the input wires of a *CJoin*, but once processed by the *CJoin*, they will be placed on the output wires and there is no way back, even though fluctuations on the output wires are allowed. When connecting *CJoin*s to each other, we should make sure that input terminals face output terminals. *Hub*s, having bi-directional wires, may be connected in any way to *CJoin*s or other *Hub*s.

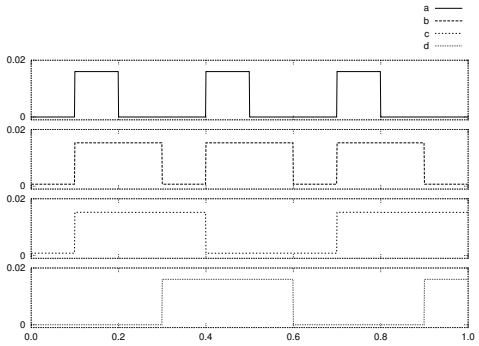

Fig. 14. Simulation results for the QR42

A. Hub Implementation

Figure 15 presents a SET circuit topology that exhibits the *Hub* functionality. The circuit was simulated at $1K$. The circuit parameters are as follows: $C_{s1} = 10aF; C_{s2} = 0.5aF; C_{s3} = 0.2aF; C_g = 10aF; C_1 = C_3 = C_5 = 10aF; C_2 = C_4 = C_6 = 0.1aF$.

The circuit operates as follows. When V_a is high an electron travels away from $n1$ through $J2$ and $J1$. Once there is a charge on $n1$, an electron is supplied to node $n1$, from $n2$ or $n3$ through $J3 - J4$ or $J5 - J6$ respectively. However, C_{s3} is chosen with a value so that the voltages over the junctions are close enough to their critical voltages so that an increase in energy due to thermal energy would randomly cause the electron to tunnel back into $n2$ or $n3$. The charge jumps from $n1$ to $n2$ or $n3$ and then back into $n1$ and then again randomly into $n2$ or $n3$. The thermal energy is thus effectively used as a random control voltage.

The results of the simulation can be seen in Figure 16. The circuit is stable when there is no supply of a charge. After time step 0.1, when the charge is supplied, we can see a random travelling of the charge through nodes $n2$ and $n3$, via $n1$.

B. Conservative Join

Figure 17 presents a SET circuit topology that implements the Conservative Join. The circuit was simulated at $1K$. The circuit parameters are as follows: $C_a = C_b = 1aF; C_{s1} = C_{g1} = 10aF; C_{s2} = C_{n3} = 0.5aF; C_{n1} = 0.25aF; C_{n2} =$

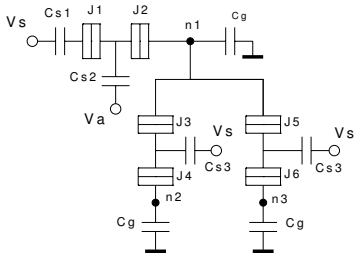

Fig. 15. Hub circuit implemenation

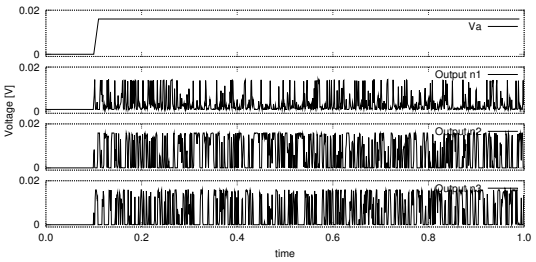

Fig. 16. Hub circuit simulation results

$0.3aF; C_{g2} = 11.5aF; C_{g3} = 10.5aF; C_{g4} = 11aF; C_1 = C_3 = C_5 = C_7 = C_{11} = 10aF; C_9 = C_{14} = 5aF; C_2 = C_4 = C_6 = C_8 = C_{10} = C_{12} = C_{13} = 0.01aF$.

The circuit operates as follows. When inputs V_a and V_b go high, electrons tunnel through the junctions $J1-J2$ and $J5-J6$ leaving positive charges on nodes $n1$ and $n3$, respectively. When $n1$ and $n3$ simultaneously have a charge, an electron tunnels from node $n5$ into the source through junctions $J9$ and $J10$. Subsequently, an electron tunnels from $n6$ to $n5$ due to the SET transistor $J11-J12$, which acts as a buffer and seperates the influences of the input C_{n1} gate capacitors and the C_{n2} driving gate capacitors. Now the charge on node $n6$ causes the charge on $n1$ to be transferred into $n2$ through $J3$ and $J4$ and the charge on $n3$ to be tranferred into $n4$ through $J7$ and $J8$. Nodes $n2$ and $n4$ are the output nodes and the input tokens, represented as positive charges, are thus transferred to the output. To bring the circuit back to a reusable state, the charge remaining on $n6$ has to be removed. This is achieved by connecting $n6$ to ground via a reversed transistor structure, $J14-J13$. If there is a positive charge residing on $n6$ and the charges on $n1$ and $n3$ become zero an electron tunnels from ground into $n6$, resetting the circuit. Once the output charges on $n2$ and $n4$ have been consumed the circuit is then ready to accept new input tokens.

The results of the simulation can be seen in Figure 18. Charges appear on the output nodes $n2$ and $n4$ only when both V_a and V_b go high, but not when only one of them goes high.

VI. Networks of fluctuation based circuits

Even though the two circuit topologies function as required at the same temperature, simply connecting them results in feedback, and to retain the random fluctuations in the *Hub*

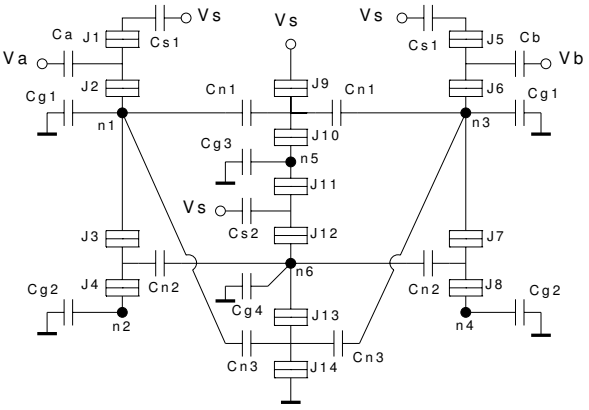

Fig. 17. Conservative Join circuit implemenation

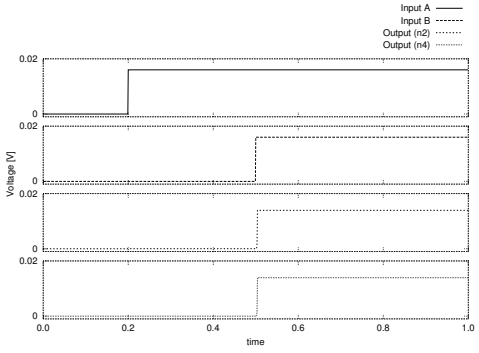

Fig. 18. Conservative Join circuit simulation results

certain parameters have to be adjusted. The circuit parameters of the *Hub* that have to be adjusted are C_{s2}, C_{s3}, C_2, C_4, and C_6. C_{s2} and C_{s3} have to be changed to $0.7aF$ and C_2, C_4, and C_6 have to be changed to $0.01aF$. Also, the grounded capacitance C_g, corresponding to the unconnected node of a *Hub* ($n2$ or $n3$) has to be changed to $10.4aF$. The circuit parameters of the *CJoin* remain unchanged.

To demonstrate that these building blocks can be combined into one functional circuit we use an example network of two *CJoin*s and three *Hub*s (Figure 19), with one *Hub* connected to both *CJoin*s. With signals present on all *Hub*s it is a matter of chance as to whether it is the first *CJoin* that fires or the second one. The common token (at the center) is consumed by the firing *CJoin*, leaving the other (non-firing) *CJoin* with only one token. A number of simulations were done using different random generator seeds and the results of two of these simulations are presented in Figures 20 and 21. In both simulations outputs which remain '0' are left out. Fig. 20 shows the charge distributions on A and B (corresponding to the left *Hub*) and C and D (right *Hub*); we see fluctuations on both *Hub*s. The central *Hub* is not supplied with a token until time 0.5 and at this point the *CJoin* with outputs E and F fires and the charges are trapped at its output. Figure 21 shows similar behaviour in the second simulation, except that

now the *CJoin* with output G and H fires, trapping the charges at its respective outputs.

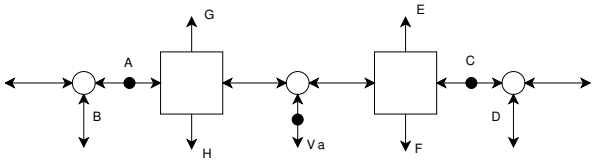

Fig. 19. Conservative Joins with Hubs Network Example

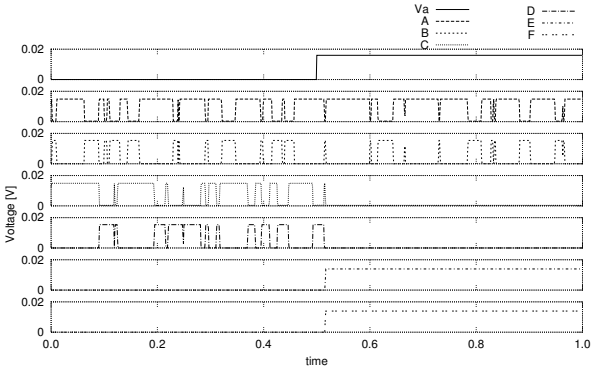

Fig. 20. Conservative Joins with Hubs simulation 1

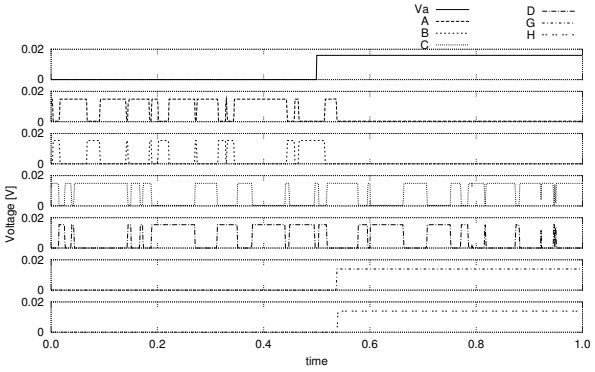

Fig. 21. Conservative Joins with Hubs simulation 2

VII. Conclusions and Future Work

In this paper we presented two computational paradigms in Single Electron Tunneling (SET) technology that address the unpredictability caused by the stochastic nature of tunneling, which is a widely recognized SET weakness. First we discussed a paradigm that is based on circuits that are insensitive to signal delays, known as Delay-Insensitive (DI), which work without a clock, and are thus free of any timing restrictions that may interfere with the stochastic behavior of SET. We introduced SET-based circuit topologies for a universal set of DI primitives and verified them by means of simulation. Afterwards we introduced a scalable buffering technique that can be utilized in conjunction with the proposed DI implementations to construct larger DI circuits. We presented a circuit example and verified it by means of simulations. The second paradigm goes one step further and allows its signals to be affected by fluctuations, which are expected to be a major problem in nanometer-scale SET circuits. We introduced SET-based circuit topologies for a universal set of the so-called Brownian-motion based constructions. With this we demonstrated that it is possible to have fluctuation-based circuits as well as deterministic ones co-exist at the same temperature. An example network was implemented and presented, which demonstrated that not only can they coexist at the same temperature, but also in the same connected circuit. In terms of circuit complexity, the DI implementations are more promising as with relatively few SET circuit elements more powerful circuit components can be built. The Brownian-motion based circuits however, taking into account that large combinations of C-Joins have to be utilised to construct useful functional circuits, would end up requiring more SET circuit elements. In terms of robustness and implementability in technolgy though, Brownian-motion based circuits are more promising as they are more error tolerant. Implementation at higher temperatures would not carry as many constraints as would for DI circuits, though it would be interesting to see if DI circuits could be developed for higher temperatures.

References

[1] R. Waser, “Ed. Nanoelectronics and Information Technology” ’Advanced Electronic Materials and Novel Devices’, 1st ed. Wiley-VCH, Berlin, 2003.

[2] ”International Technology Roadmap for Semiconductors, 2003 Edition, Executive Summary”.

[3] S. Cotofana, C. Lageweg, and S. Vassiliadis, ”Addition Related Arithmetic Operations via Controlled Transport of Charge,” *IEEE Transactions of Computers*, vol. 54, no. 3, pp. 243-256, March 2005.

[4] C. Lageweg, S. Cotofana, and S. Vassiliadis, ”Static Buffered SET Based Logic Gates”, in *Proceedings of the 2nd IEEE International Conference on Nanotechnology (NANO)*, pages 491 - 494 (Arlington, USA), August 2002.

[5] C. Lageweg, S. Cotofana, and S. Vassiliadis, ”A Linear Threshold Gate Implementation in Single Electron Technology”, in *Proceedings of the IEEE Computer Society Workshop on VLSI*, pages 93 - 98 (Orlando, USA), April 2001.

[6] N. Asahi, M. Akazawa, and Y. Amemiya, ”Single-Electron Logic Systems Based on the Binary Decision Diagram,” *IEICE Transactions on Electronics*, vol. E81-C, no. 1, pp. 49-56, January 1998.

[7] J. Lee and et. al. ”Brownian Circuits — Part II: Efficient Designs and Brownian Cellular Automata.” *In preparation*, 2008.

[8] F. Peper and et. al. ”Brownian Circuits — Part I: Concept and Basic Designs.” *In preparation*, 2008.

[9] T. Yamada, M. Akazawa, T. Asai, and Y. Amemiya. ”Boltzmann Machine Neural Network Devices using Single-Electron Tunnelling.” *Nanotechnology*, 12(1):60–67, 2001.

[10] C. Wasshuber, ”About single-electron devices and circuits,” Ph.D. dissertation, TU Vienna, 1998.

[11] C. Wasshuber, H. Kosina, and S. Selberherr. ”SIMON - A Simulator for Single-Electron Tunnel Devices and Circuits.” *IEEE Transactions on Computer-Aided Design*, 169:934-944, September 1997. http://www.lybrary.com/simon/

[12] J. Lee, F. Peper, S. Adachi and K. Morita, ”Universal Delay-Insensitive Circuits with Bidirectional and Buffering Lines” *IEEE Transactions on Computers*, vol. 53, no. 8, August 2004.

[13] P.G. Lucassen, ”A Denotational Model and Composition Theorems for a Calculus of Delay-Insensitive Specifications”, Ph.D. thesis, Dept. of C.S., Univ. of Groningen The Netherlands, May 1994