Design of Arithmetic Circuits Using Resonant Tunneling Diodes and Threshold Logic

Christian Pacha and Karl Goser
Lehrstuhl Bauelemente der Elektrotechnik,
Universität Dortmund
D-44221 Dortmund, Germany
E-mail: pacha@luzi.e-technik.uni-dortmund.de

Abstract

This paper describes the design of arithmetic circuits based on hybrid integrated resonant tunneling diodes and heterostructure field-effect transistors. The key components are depth-2 parallel counters consisting of multiple terminal threshold gates. In particular, we propose a novel parallel addition scheme by combining threshold logic and systolic VLSI-algorithms for bit-level computations. The approach is motivated by the demand for locally interconnected circuit modules to solve the wiring problem in nanoelectronic circuits.

1 Introduction

During the last decades the progress in microelectronics primarily results from the scaling of the devices and the development of information processing systems such as microprocessors and memories. Today, originating from the possibility to fabricate semiconductor heterostructures with atomic layer thickness, there are tremendous activities in research and development of devices so small that quantum mechanical effects become relevant for their operation.

In the field of resonant tunneling structures the experimental research concentrates on different kinds of three terminal devices. Significant examples are resonant hot electron transistors [22], and gated resonant tunneling diodes [19]. In addition, several hybrid microelectronic-nanoelectronic devices have been developed. They are composed of resonant tunneling diodes (RTD) which are integrated together with heterostructure field-effect transistors (HFET) [16]. During the initial phase of nanoelectronics this is a first way to get some experience with novel circuit architectures of lower complexity until lateral nanostructured devices are available.

Although at present most of the research is technologically oriented a further preliminary to implement complete systems with these novel devices is the investigation of circuit architectures and the development of a kind of design framework [9]. Recapitulating some milestones the first example that has demonstrated how to build a logic circuit with quantum-effect devices is a 1-bit full adder based on X-NOR gates. It has been proposed by Capasso in 1989 [3] and reduces the circuit complexity of the adder by taking advantage from the multistate behavior of a RTD. Other applications following this approach are different kinds of multiple-valued logic gates and resonant tunneling diode memory cells [15], [23]. The common idea of these circuits is to decrease the number of devices which are required to implement a specific logic functionality [20].

Another essential aspect and the principal topic of this paper is the question if these functionally integrated circuits could be combined with adequate VLSI algorithms for bit-level computations. The fundamental relevance of this aspect has been emphasized by Ancona [1] in context with single electron transistor circuits for multiplication and Fast Fourier Transform. Independent of the technological realization and the operating principles of the different families of nanoelectronic devices, important design principles for an architectural approach towards nanoelectronics are:

- A regular layout with a small number of different circuit modules.
- The use of quantum effects to reduce the logic depth of a circuit.
- Local interconnections on the circuit and the system level to solve the wiring problem.

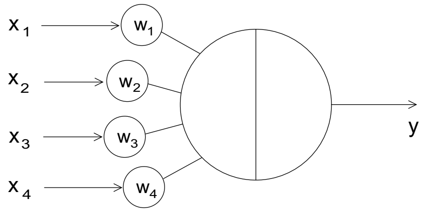

Figure 1: Linear threshold gate

- Concurrent computation and pipelining at the bit-level to achieve a low latency and a high data throughput.

Even today most of these principles are a substantial part of modern CMOS-VLSI and it is obvious that the problem if quantum-effect devices will be useful has to be investigated from that point of view, too.

The paper is structured as follows: Section 2 describes the implementation of a linear threshold gate with RTDs and HFETs. After this on higher level we utilize the RTD-based threshold gates to design a parallel counter, that is a special combinatorial network for adding multiple operands. In section 3 it will be shown that parallel counters could be used in several arithmetic circuits for parallel addition with an increased logic density. This part of our work bases on the theoretical investigations of threshold logic circuits and applications done by Vassiliadis, Dadda and Swartzlander [24], [8], [21]. A disadvantage of these threshold logic circuits is that regularity and local interconnections were not explicitly considered from the beginning as design principles because the algorithms are optimized only with regard to the logic depth. To overcome this problem we take up the ideas of bit-level systolic arrays and regular, tree-like structures for carry lookahead addition [14], [2]. As a result we propose a novel systolic addition scheme for RTD-based threshold gates in section 4. To estimate the performance of the proposed circuits the area and time complexity is analyzed for various operand lengths in section 5.

2 RTD-based threshold gates

A linear threshold gate is a multiple terminal device that calculates the weighted sum χ of the digital inputs $x_k, k = 1, \ldots, N$. Afterwards the gate converts this sum into a digital output y by comparing χ with a given threshold value Θ (figure 1). Adapting the weights $\{w_1, \ldots, w_N\}$ and the threshold value Θ, a linear threshold gate computes any linear separable Boolean function of the N inputs. Compared with a Boolean logic gate, a threshold gate combines an internal analog computation of the weighted sum with digital encoded input and output states. The output y of a threshold gate is given by

$$y(\chi) = \mathrm{sign}(\chi - \Theta) = \begin{cases} 1 & \text{if} \quad \chi \geq \Theta \\ 0 & \text{if} \quad \chi < \Theta \end{cases} \tag{1}$$

$$\chi = \sum_{k=0}^{N} w_k \cdot x_k, \tag{2}$$

$$x_k = \{0, 1\}, \tag{3}$$

$$w_k = \{0, \pm 1, \ldots, \pm w_{max}\}, \tag{4}$$

$$\Theta = \{0, \pm 1, \ldots, \pm \Theta_{max}\}. \tag{5}$$

Recently, a RTD-based threshold gate has been proposed which consists of two serially connected RTDs and multiple parallel HFETs (figure 2) [5]. Here, the HFETs enable the weighting of the digital inputs. The RTDs are used to generate a digital output and to compare the positive and negative weighted inputs with the threshold value Θ. The advantage of this circuit configuration is that the complex functions of a linear threshold gate (weighting, summation and comparison) are implemented with a few number of devices only. In the following two subsections we give a short summary how this RTD-based threshold gate works. A more detailed investigation of the underlying operating principles can be found in [5], [18] and [17].

2.1 Weighting of the digital inputs

Figure 2b shows that the drain to source current of the HFETs modulates the peak current of the parallel RTD. Since this drain to source current depends linearly on the transistor width, a weighting of the input signals is obtained by varying the transistor geometry. The HFETs connected in parallel to the top RTD perform a positive weighting of the inputs whereas HFETs in parallel to the bottom RTD are negative weighted inputs. The threshold value of the gate is an additional negative weight controlled by the gate voltage V_Θ. Thus, the internal signal of the weighted sum χ is represented by the total modulation current at the output node according Kirchhoff's current law. If the threshold gate has M_p positive weighted and M_n negative weighted inputs, the total modulation current is

$$\Delta I = \sum_{k=1}^{M_p} w_k \cdot I_F(V_{Gk}) - \sum_{k=1}^{M_n} w_k \cdot I_F(V_{Gk}) - w_\Theta \cdot I_F(V_\Theta). \tag{6}$$

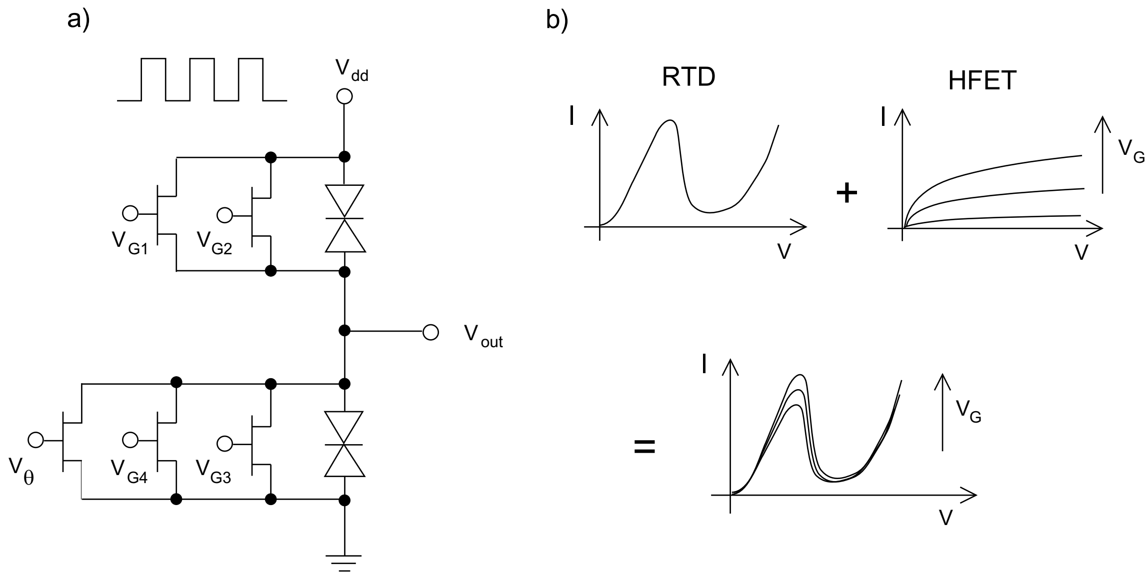

Figure 2: RTD-based threshold gates (a) and modulation of the RTD-current (b).

Here I_F is the drain to source current of a HFET with minimum width. The weight factors w_k and w_Θ express the width ratios between the HFETs with weighted inputs and the HFET with minimum width $w = 1$. The inputs x_k of the threshold gate are set by the gate source voltage V_{Gk}. If the gate voltage V_{Gk} exceeds the threshold voltage of the HFET the current $w_k \cdot I_F$ is added to the RTD current.

2.2 Threshold operation and switching

In the following we assume that the total modulation current ΔI is small compared to the peak current of the RTDs and can be neglected to calculate the bistable output voltage V_{out}. The most characteristic feature of the circuit in figure 2 is the oscillating bias voltage V_{dd} on top of the two RTDs. Thus, together with the nonlinear current-voltage characteristics of the RTDs, the output behavior of the circuit is either bistable or monostable. The bistable configuration occurs at a bias voltage larger than twice the peak voltage and generates two self-stabilizing digital output states. In figure 3 the two logic states appear at a bias voltage of $V_{dd} > 0.8$V where the central equilibrium point becomes unstable. The equilibrium points are the intersection points of the RTD-currents and indicated by dots. At a larger bias voltage there is a small region with three stable equilibrium points and two unstable equilibrium points. If the bias voltage exceeds five times the peak voltage the circuit becomes monostable again. In the example chosen here (figure 3, left) the peak voltage and the peak current of the RTDs are $V_p = 0.4$V and $I_p = 5$mA with a peak to valley ratio of $PVR = 5.5$. To understand the switching into a logic high or low state it is important to notice that metastable transition point where the output behavior changes from monostability to bistability is very sensitive to the small modulation current ΔI. Using this sensitivity is an area efficient way to implement the comparison function of a threshold gate because the sign of the modulation current is equivalent to the sign of the weighted sum. The logic state high (low) corresponds to a positive (negative) sign of the modulation current. Thus, after the bias voltage has produced a bistable output and a short relaxation phase has finished the circuit converts the internal weighted sum (i.e. the modulation current) into a digital output:

$$V_{out} = \mathrm{sign}(\Delta I) = \begin{cases} 0.95\mathrm{V} & \text{if} \quad \Delta I \geq 0 \\ 0.05\mathrm{V} & \text{if} \quad \Delta I < 0 \end{cases} \qquad (7)$$

Since we will later design a multilayer network and operate the threshold gates in a pipelined way the oscillating bias voltage enables a clocking of the threshold gate to synchronize these networks. Due to the fact that the output of a layer is available only at the time when the bias voltage is larger than twice the peak voltage this dynamic circuit technique requires two overlapping clocking schemes.

To investigate how the amplitude of the oscillating bias voltage influences the circuit operation the right part of figure 3 illustrates the multistable output be-

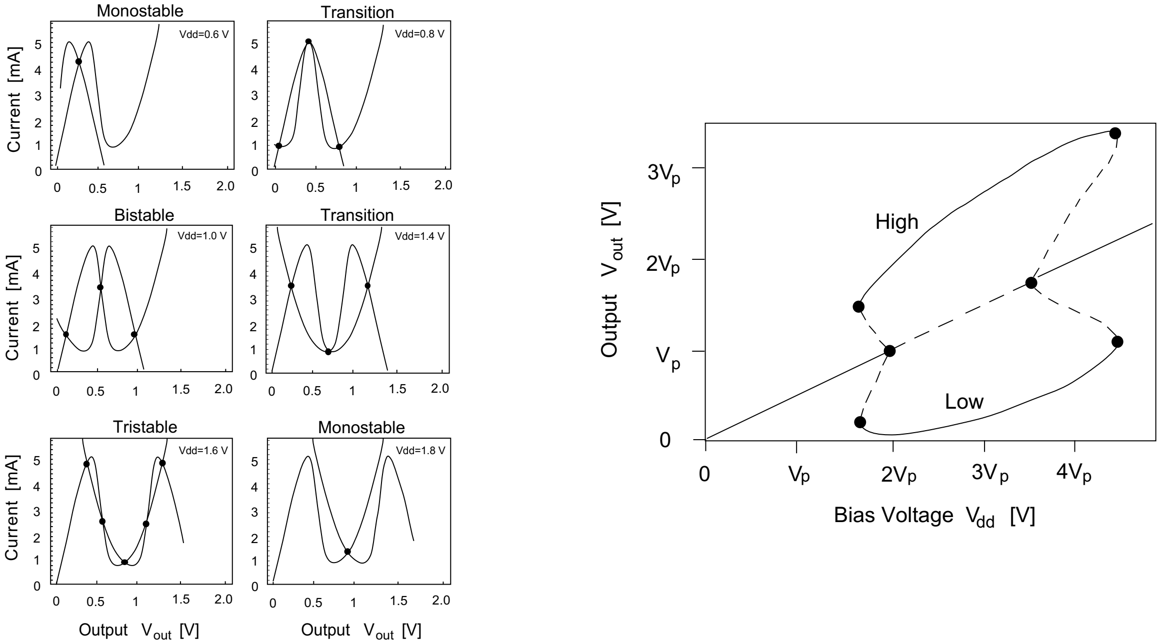

Figure 3: Monostable-bistable behavior of two serially connected RTDs.

havior obtained from a nonlinear circuit analysis of the RTD-pair. The six transition points where the stability of the output changes are marked by dots. Stable equilibrium points are indicated by solid lines while the unstable equilibrium points are indicated by dashed lines. If the electrical parameters of an RTD are given this allows to maximize the noise margin, that is the difference between the high and low state, by adapting the bias voltage. In larger circuits with a multilayer architecture the noise margin and the location of the logic states are important to switch the gates of the HFETs in the subsequent layer. If we choose the RTD-parameters mentioned above an appropriate bias voltage has to be about $2.5V_p$ to obtain a noise margin of about 0.9V.

Switching a logic gate by means of a monostable-bistable transition in a symmetric configuration of two tunneling diodes is a well-known technique in nonlinear dynamic circuits and has been investigate first by Goto in 1960 [10]. Since field-effect transistors to implement the terminals of the gate were not available at that time this circuit architecture has not played an important role in microelectronics. Today, the technological development and the more profound understanding of nonlinear phenomena are the reason for the comeback of these circuit configurations in nanoelectronics. A similar configuration of RTDs and HFETs is used in a high speed and low power static memory cell by Texas Instruments [23]. Very recently, a high frequency RTD-Schottky gate with the same operating principle as our RTD-threshold gate has been demonstrated to operate at a maximum frequency of 12GHz [11]. Therefore, these RTD-circuits might be potential candidates for future high speed signal processing.

3 Parallel counters for circuits with reduced logic depth

The implementation of arithmetic functions, especially parallel addition schemes, is one possible application of RTD-based threshold gates. In this section we describe four different kinds of parallel adders and analyze how they are designed. The basic components of the adders are generalized parallel counters. A parallel counter is a combinatorial network that receives n digital inputs and computes the number of active inputs, that is the number of ones (figure 4). The reason for designing arithmetic circuits with parallel counters is that they lead to a very efficient implementation of addition and multiplication schemes with a reduced logic depth [8], [6]. Regarding the notation, a $k_{r-1}, \ldots, k_0|m$ generalized counter receives r columns of $k_l, l = \{0, \ldots, r-1\}$ digital inputs and computes a digital output of word length m [21]. Each input row k_l is weighted by 2^l. Figure 4 shows three parallel counters with a differ-

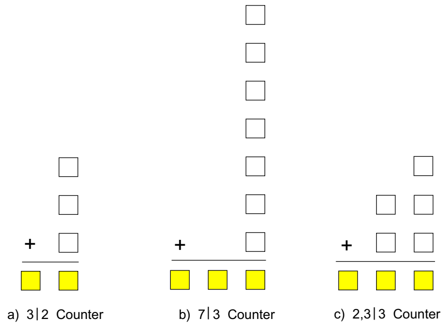

Figure 4: Different configurations of parallel counters.

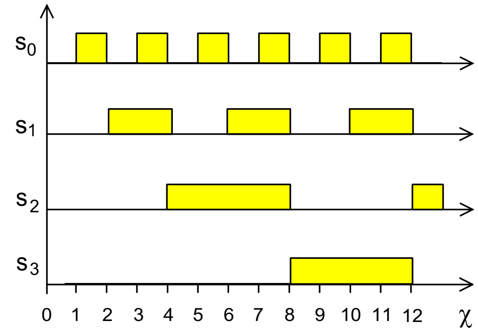

Figure 5: Periodical dependency of the output bits s_j.

ent number of inputs and outputs . As depicted, a 3|2 counter is equivalent to a full adder.

The design of a parallel counter with a depth-2 linear threshold network is relatively simple because each output bit $s_j, j = \{0, \ldots, m-1\}$ is a periodic symmetric function. The characteristic feature of a periodic symmetric function is that the output of the function depends only on the weighted sum χ of the n inputs. As illustrated in figure 5 the output $s_j(\chi)$ consists of a periodic sequence of high and low intervals. Each interval of $s_j(\chi)$ has an equal length 2^j. For example, the linear threshold network of a 7|3 counter can be described by

$$\begin{aligned} s_0(\chi) &= [1]+[3]+[5]+[7] \\ s_1(\chi) &= [2,4]+[6[\\ s_2(\chi) &= [4[. \end{aligned} \tag{8}$$

Here high intervals of the sum χ are defined by

$$\begin{aligned} [a] &= 1 \quad \text{if} \quad \chi = a \\ [a,b] &= 1 \quad \text{if} \quad a \leq \chi \leq b \\ [a[&= 1 \quad \text{if} \quad a \leq \chi \end{aligned} \tag{9}$$

where a and b are the boundaries of the intervals. Thus, the corresponding threshold network has to detect the boundaries of the high intervals and to perform an OR operation afterwards. This is achieved by a suitable selection of the threshold values Θ_i in the first layer of the network. In the case of a 7|3 counter the first layer comprises seven gates having the threshold values $\{1, 2, 3, 4, 5, 6, 7\}$. To compute the OR operation in the second layer of the network we use three-valued weights $w_i = \{0, \pm 1\}$ only. Other linear threshold networks, such as the Kautz-network [12] or Telescopic Sums [24] are not regarded here since they are more difficult to implement within the proposed circuits.

In the following, $2,3|3$ counters are frequently used to group the input operands of an adder into blocks of 2-bits at a time. The intention is to improve the delay in parallel addition schemes. The two most significant bits (MSB) of the $2,3|3$ counter are weighted by $w_i = 2$ while the two least significant bits and the incoming carry are weighted by $w_i = 1$. Apart from the different weighting of the input operands a $2,3|3$ counter and a 7|3 counter have the same input capacity of $\chi_{max} = 7$ and therefore the same output behavior.

Concerning the implementation costs of a parallel counter important features are the number of gates, the depth of the network, the magnitude of the weights and the fan-in. While the delay time directly follows from the logic depth of the network, the area of the circuits depends on the number of linear threshold gates as well as on the magnitude of the weights. In this context one has to make a compromise between networks with a very small depth and the boundary conditions of an implementation. The most critical conditions are a reliable operation of the gate and the limitation of the dynamic power dissipation. The dynamic power dissipation is primarily affected by a large fan-in (i.e. many inputs and large weights) because a large number of inputs increases the total input capacity of the RTD-based threshold gate. In addition, if the number of internal states of the weighted sum χ increases the gates becomes more prone to fluctuations and parameter variations. Consequently, linear threshold gates with small weights and small fan-in are basically of interest. Based on SPICE circuit simulations we expect that an upper boundary for the input capacity of a parallel counter is $\chi_{max} = 12$ [18].

3.1 Ripple carry addition

The most simple algorithm for adding two n-bit numbers in parallel is a ripple carry adder. It consists of n serially connected 3|2 counters (i.e. full adders) and

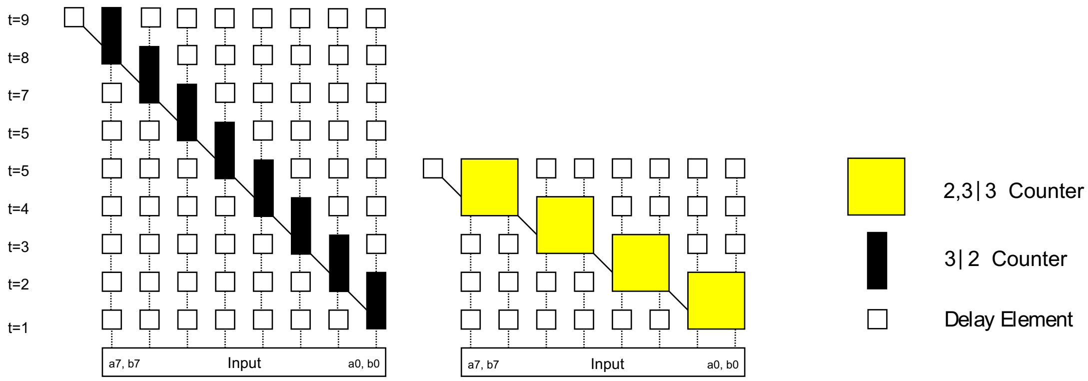

Figure 6: Ripple carry adder with 3|2 counters (left) and $2,3|3$ counters (right).

has a logic depth of $d = n + 1$. To decrease the delay time T which is directly proportional to the logic depth d (under the assumption that the wiring delay can be neglected if the cells are locally interconnected) Cotofana has proposed a δ-bit serial adder [6] where the operands are grouped into blocks of length δ. Using parallel counters the advantage is a speed up of the carry propagation since the carry of a δ-bit block is available after a delay of one. Figure 6 shows a ripple carry and a δ-bit adder with 2-bit grouping for 8-bit unsigned numbers. The δ-bit adder is composed of $2,3|3$ counters. In both adders the carries are propagating along the diagonal lines. The δ-bit serial adder has a logic depth of $d = \lceil n/\delta \rceil + 1$ and thus for an 8-bit adder with 2-bit grouping the delay is reduced from $d = 9$ to $d = 5$. To obtain a correct timing additional delay elements, indicated by the white squares, have to be inserted. Since there are only local interconnections the ripple carry adder and the δ-bit adder are in convenience with the design principles discussed in the introduction. However, the linear dependency of the time complexity from the operand length is an obvious disadvantage. It should be emphasized that we have considered the logic depth of each adder explicitly in the computation graphs to obtain a realistic estimation of the performance. In many theoretical investigations of VLSI algorithms the delay of a single cell is defined as $d_{cell} = 1$ and thus the final evaluation is only based on the asymptotic behavior of the algorithm. Often this makes it difficult to compare different algorithms for operand lengths up to 64-bit being relevant for practical applications.

3.2 Carry lookahead addition

To decrease the delay time for 16 or 32-bit operands the carry propagation could be accelerated by means of a carry look ahead computation or by operating an 8-bit adder in a pipelined way. For example, a pipelined addition scheme of 32-bit numbers splits the operands into 4 blocks of 8-bits before the blocks are added successively. Obviously, such a bit-level pipelining enables a high data throughput of larger operands without an significant increase of area. The second way is a carry lookahead adder where the gates are connected in a regular way as proposed by Brent and Kung [2]. The basic idea is an algorithm with a binary tree structure to compute the group generate and propagate carries $G_{i,j}$ and $P_{i,j}$ according to the recursive equations

$$G_{i,j} = G_{i,k} + P_{i,k} \cdot G_{k-1,j} \tag{10}$$

$$P_{i,j} = P_{i,k} \cdot P_{k-1,j} \tag{11}$$

for

$$i \geq k \geq j + 1. \tag{12}$$

The derivation of these equation is straightforward: Dividing a block comprising the bits from position j to i into a MSB-group from k to i and a LSB-group from j to $k-1$ the complete group generates a carry if either the MSB-group generates a carry or the LSB generates a carry. This LSB-carry is afterwards propagated by the MSB-group (figure 8). The complete block propagates a carry if both the LSB-group and the MSB-group propagate an incoming carry. At the final level of the binary tree

$$c_i = G_{i-1,0} \tag{13}$$

and

$$p_i = P_{i,i} \tag{14}$$

are computed before one obtains the sum bit

$$s_i = p_i \oplus c_i. \tag{15}$$

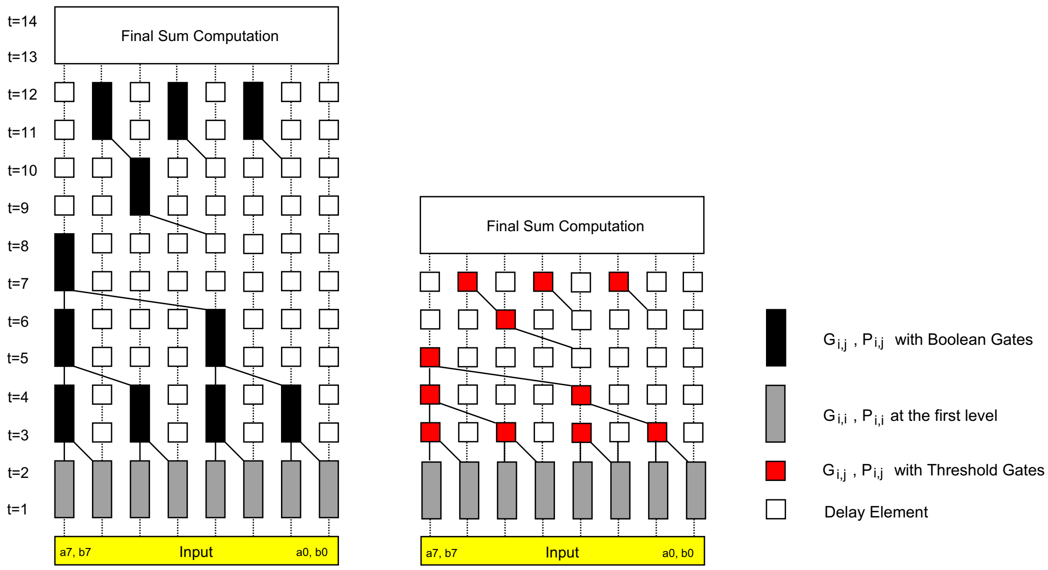

Figure 7: Brent and Kung CLA adder with boolean gates (left) and a threshold logic version (right).

At the first level the group generate carry $G_{i,i}$ and group propagate carry $P_{i,i}$ are given by

$$G_{i,i} = a_i \cdot b_i \tag{16}$$

and

$$P_{i,i} = a_i \oplus b_i \tag{17}$$

The first step is now to implement this algorithm with linear threshold gates instead of Boolean gates as used in the original version of Brent and Kung. Both, the original algorithm and the threshold logic version are illustrated in figure 7. The advantage of the threshold logic version is that the logic depth of a single $(G_{i,j}, P_{i,j})$-cell is reduced from $d_{cell} = 2$ to $d_{cell} = 1$ because the group propagate and generate carries can be computed with two linear threshold gates with a delay of $d_{G_{i,j}} = d_{P_{i,j}} = 1$:

$$\begin{aligned} G_{i,j} &= G_{i,k} + P_{i,k} \cdot G_{k-1,j} \\ &= \text{sign}(2G_{i,k} + P_{i,k} + G_{k-1,j} - 2) \end{aligned} \tag{18}$$

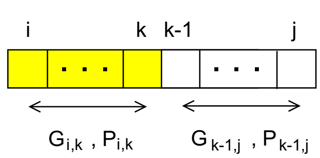

Figure 8: Computation of the group propagate and generate carries

and

$$\begin{aligned} P_{i,j} &= P_{i,k} \cdot P_{k-1,j} \\ &= \text{sign}(P_{i,k} + P_{k-1,j} - 2) \end{aligned} \tag{19}$$

if

$$i \geq k \geq j+1. \tag{20}$$

The maximum weight and the maximum threshold value are $w_{max} = \Theta_{max} = 2$. At the first level of the CLA-tree the group carries are computed with a delay of $d_{G_{ii}} = d_{P_{ii}} = 2$ directly from the operands according to (16) and (17). Although this CLA addition scheme leads to a regular layout there are three disadvantages:

- The binary tree structure includes several non-local interconnections and is therefore not systolic following the rigorous definition of a systolic VLSI algorithm in [13]. The effect of non-local interconnections is an increase of the delay time due to the signal propagation on the wire. Often this propagation time is not considered when estimating the time complexity of an VLSI algorithm. Assuming the realistic model that the propagation time on a wire is at least linear in the distance the delay of the Brent and Kung adder increases from $O(\log_2 n)$ to $O(n)$ as well as the theoretical lower bound for addition from $O(\log_2 n)$ to $O(\sqrt{n})$ [4].

- After the carry of the MSB has been computed a second binary tree is used to compute the carries of intermediate positions which are required for the final sum (figure 7, left, $t \geq 9$).

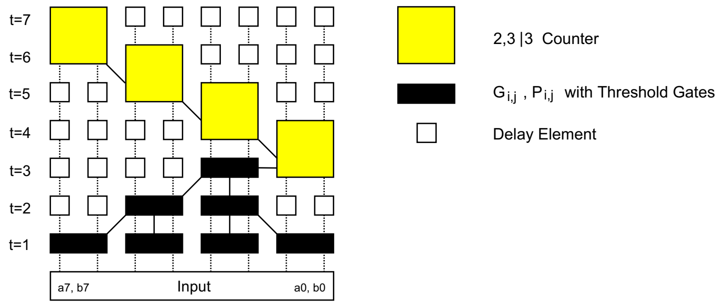

Figure 9: Systolic addition scheme with CLA computation and $2,3|3$ counters.

- If a pipelined operation is intended one has to feed back the MSB-carry of a block to the LSB of the following block. Obviously, the wire of the feed back line is as long as the complete adder and would violate our demand for a local interconnection. Apart from that, one has to introduce an extra delay before presenting the subsequent block of operands to consider because the propagation time that is required to compute the MSB-carry. Thus, the distance between two successive computations increases and limits the data throughput.

4 Systolic addition scheme

To solve the problems arousing from the non-local interconnections in the CLA-tree our task is now to implement the carry propagation tree and to maintain a local and regular design. Therefore, we propose a novel, purely systolic addition scheme that combines

- a threshold logic implementation of a carry lookahead tree,
- a δ-bit adder consisting of $2,3|3$-counters to compute the final sum and
- a pipelined operation based on an 8-bit block without a long carry feedback wiring.

The systolic 8-bit CLA/δ-Bit Adder comprises two parts (figure 9). The bottom part is a depth-3 CLA-tree that computes the carry of the MSB by means of the group generate and propagate carries. In contrast to the previous section here we have grouped the operands at the first level into 2-bit blocks and start with computing the group carries $G_{i,i-1}$ and $P_{i,i-1}$ [7]. A group generate carry $G_{i,i-1}$ is produced if the weighted sum of the 4 inputs exceeds a threshold value of $\Theta = 4$. In a similar way a group carry is propagated if the weighted sum exceeds a threshold value of $\Theta = 3$ and one obtains

$$G_{i,i-1} = \text{sign}(2a_i + 2b_i + a_{i-1} + b_{i-1} - 4) \tag{21}$$

$$P_{i,i-1} = \text{sign}(2a_i + 2b_i + a_{i-1} + b_{i-1} - 3). \tag{22}$$

The 2-bit grouping of the operands saves the first stage because $G_{i,i-1}$ and $P_{i,i-1}$ are computed from the input operands. In contrast to this in the Boolean logic version $G_{i,i}$ and $P_{i,i}$ have to be computed first. Since we have limited the word length to 8-bit the binary CLA-tree does not include any non-local interconnections and is therefore a purely systolic algorithm.

The top part of the adder computes the sum bits with a delay of $d = 5$ by 2-bit grouping using $2,3|3$ parallel counters. To operate this 8-bit adder in a pipelined fashion both parts are connected at the third level where the MSB-carry is used as LSB-carry for the next 8-bit block. Because the MSB carry is available exactly at the same time when the LSB of the next 8-bit block enters the first $2,3|3$ counter no extra delay has to be introduced between two subsequent blocks. Furthermore, the top part of the CLA-tree where the intermediate carries are computed is removed. Therefore, the pipelined operation can be done very efficiently and a high data throughput is achieved.

Up to now we have not mentioned how the different blocks used in the addition schemes are designed with threshold gates. Based on the threshold logic equations above the upper half of figure 10 shows the $(G_{i,j}, P_{i,j})$-cells of the CLA-tree and the delay elements. A delay element is a single threshold gate with one input and $w_1 = \Theta_1 = 1$. The bottom of half figure 10 contains the parallel counters. The $2|1$ counter is used in the last stage of the Brent and Kung adder to compute the final sum. While the $3|2$ counter belongs to the conventional ripple carry adder, the $2,3|3$ counters are used in the systolic adder and the ripple carry adder with 2-bit grouping. If the operands are grouped into 2-bit blocks 4 delay elements are required to obtain a right timing, otherwise 2 delay elements are sufficient.

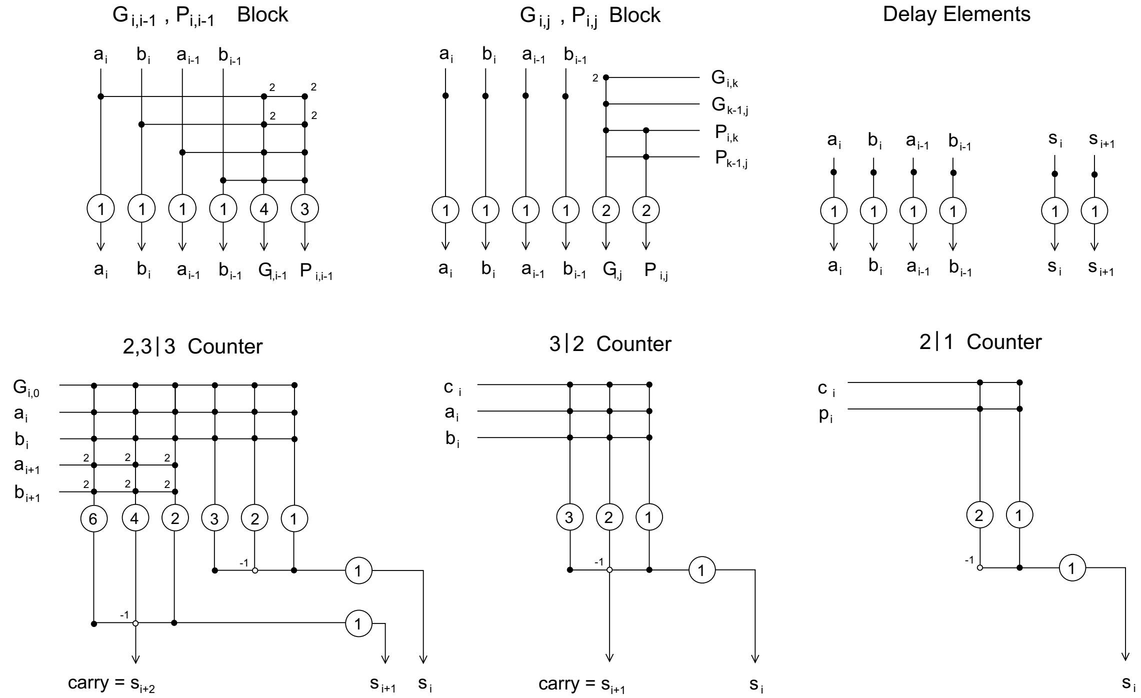

Figure 10: Synthesis of the different blocks with linear threshold gates.

Inputs with weights $w_i = 1$ and $w_i = -1$ are indicated by black and white dots. Due to the condition to use threshold gates with small fan-in the maximum weight and threshold value are $w_{max} = 2$ and $\Theta_{max} = 6$.

5 Comparison

In this section we compare the proposed addition schemes in regard to the delay time, the number of linear threshold gates and the area for operand lengths of $8, 16, 32$ and 64 bits. It is assumed that the four adders could be implemented with the proposed RTD-based threshold gates. The area of an adder is approximately given by the product of the total number of linear threshold gates and the area of a single threshold gate

$$A = N_{LTG} \cdot A_{LTG}. \tag{23}$$

This assumption can be justified because most of the area of a single linear threshold gate is covered by the two serial resonant tunneling diodes. Neglecting the area of the HFETs does not produce a large error if the threshold gates have small weights.

The total delay time T of an adder is the product of the number of steps S that are required to finish the computation and the delay time t_{LTG} of a single threshold gate. In the case of an addition scheme without pipelining the number of steps S is equal to the logic depth d of the circuit. In the pipelined addition scheme S is the difference between the input of the first block and the time when the last sum bit of final block has been computed. With reference to section 2 the delay time t_{LTG} is equivalent to the inverse clocking frequency $1/f_{V_{dd}}$, that is the frequency of the oscillating bias voltage:

$$T = S \cdot t_{LTG} = \frac{S}{f_{V_{dd}}} \tag{24}$$

To estimate the performance in a technological independent way we measure the delay time in terms of the number of steps S and the area in terms of the total number N_{LTG} of linear threshold gates.

Since there is always an area-time trade off which has to be minimized when implementing an arbitrary VLSI algorithm, we have estimated the AT-product according to

$$AT = (N_{LTG} \cdot A_{LTG}) \cdot (S \cdot t_{LTG}) \propto N_{LTG} \cdot S. \tag{25}$$

The results of the comparisons are summarized in the tables 1-3.

When comparing the AT-products for the different adders, one obtains the expected results that the Brent and Kung adder and our systolic addition scheme are

n	RCA	2-Bit	BK	SY
8	9	**5**	7	7
16	17	9	10	**8**
32	33	17	12	**10**
64	65	33	**14**	**14**

Table 1: Delay S.

n	RCA	2-Bit	BK	SY
8	783	**345**	728	798
16	5185	2097	3040	**1168**
32	36993	14433	8832	**1780**
64	278785	106689	24192	**3388**

Table 2: Area-Time complexity $N_{LTG}\ d$.

n	RCA	2-Bit	BK	SY
8	87	**69**	104	114
16	305	233	304	**146**
32	1121	849	736	**178**
64	4289	3233	1728	**242**

Table 3: Number of the linear threshold gates N_{LTG}.

These terms are used in the tables:

n	Operand length.
RCA	Ripple Carry Adder.
2-Bit	Ripple Carry Adder with $2,3\|3$ counters.
BK	Brent and Kung adder, threshold logic version.
SY	Systolic adder with CLA-tree and $2,3\|3$ counters.

advantageous for operand lengths larger than 16-bit. Due to the more complicated overhead the carry lookahead schemes loose this advantage for 8-bit operands where the ripple carry adder with 2-bit grouping should be preferred due to the smaller AT-product.

The reason for the good performance of our systolic addition scheme is that the distance between two successive computations is one clock cycle only. Therefore, a pipelined operation of an 8-bit adder is very efficient and limits the increase of the number of threshold gates for large operands. Compared with the simple ripple carry adder the AT-product of the systolic adder is about two order of magnitudes better for 64-bit operands. This is a strong indication that optimizing the logic design of future nanoelectronic circuits on the gate level enables an increase of performance.

6 Conclusions

In this paper we have investigated several algorithms for parallel addition based on linear threshold gates with small input capacity. We have shown that the increased functionality of hybrid integrated nanoelectronic devices could be transfered to the circuit level. Using depth-2 parallel counters as basic circuits modules and adapting existing VLSI algorithms for carry lookahead addition, a novel systolic algorithm for 8-bit pipelined addition has been proposed. In addition, the implementation of a ripple carry adder and a tree-like carry lookahead adder with linear threshold gates has been analyzed. For a quantitative comparison in a technological independent way we have estimated the area and time complexity of the different algorithms. To overcome the serious interconnection problem, fundamental design principles such as modularity and local communication between different circuit modules have been considered as a guideline to design nanoelectronic circuits.

References

[1] M.G. Ancona. Systolic processor designs using single-electron digital circuits. *Superlattices and Microstructures*, 20(4):461–471, 1996.

[2] R. Brent and H.T. Kung. A Regular Layout for Parallel Adders. *IEEE Trans. on Computers*, 31(3):260–264, March 1982.

[3] F. Capasso, S. Sen, and F. Beltram. Quantum-effect devices. In S.M. Sze, editor, *High-Speed Semiconductor Devices*, pages 465–530. Wiley, NY, 1990.

[4] B. Chazelle and L. Monier. A Model of Computation for VLSI with Related Complexity Results. *Journal of the ACM*, 32(3):573–588, July 1986.

[5] K. J. Chen, K. Maezawa, and M. Yamamoto. InP-Based High-Performance Monostable-Bistable Transition Logic Elements (MOBILE's) Using Integrated Multiple-Input Resonant-Tunneling Devices. *IEEE Electron Device Letters*, 17(3):127–129, March 1996.

[6] S. Cotofana and S. Vassiliadis. Delta-Bit Serial Binary Addition with Linear Networks. *Journal of VLSI Signal Processing*, 3(14):249–264, December 1996.

[7] S. Cotofana and S. Vassiliadis. Low Weight and Fan-In Neural Network for Basic Arithmetic Operations. In *Proc. of 15th IMACS World Congress, Berlin, Germany*, August 1997.

[8] L. Dadda. On Parallel Digital Multipliers. *Alta Freq.*, 45:547–580, October 1976.

[9] K. Goser, C.Pacha, A. Kanstein, and M.L. Rossmann. Aspects of Systems and Circuits for Nanoelectronics. *Proc. of the IEEE: Special Issue on Nanoelectronics*, 85(4):558–573, April 1997.

[10] E. Goto et al. Esaki Diode High-Speed Logical Circuits. *IRE Transactions on Electronic Computers*, pages 25–29, 1960.

[11] W. Williamson III et al. 12 GHz Clocked Operation of Ultralow Power Interband Resonant Tunneling Diode Pipelined Logic Gates. *IEEE Journal of Solid-State Circuits*, 32(2):222–231, February 1997.

[12] W. H. Kautz. The Realization of Symmetric Switching Functions with Linear-Input Logical Elements. *IRE Transactions On Electronic Computer*, pages 371–378, 1961.

[13] L. Kuehnel. Optimal Purely Systolic Addition. Technical Report 9002, Christian-Albrechts-Universität Kiel, Germany, April 1990.

[14] J.V. McCanny and J.G. McWhirter. Implementation of signal processing functions using 1-bit systolic arrays. *Electronics Letters*, 18(6):241–243, March 1982.

[15] L. J. Micheel, A. H. Taddiken, and A. C. Seabaugh. Multiple-Valued Logic Computation Circuits using Micro- and Nanoelectronic Devices. In *Proc. of the 23rd Int.Symp. on Muliple Valued Logic, Sacrament, CA, USA*, pages 164–169. IEEE Computer Society Press, May 1993.

[16] C. H. Mikkelson, A. C. Seabaugh E. A. Beam III, and J. H. Luscombe. Coupled-Quantum-Well Field-Effect Resonat Tunneling Transistor for Multi-Valued Logic/Memory Applications. *IEEE Transactions on Electron Devices*, 41(2):132–137, February 1994.

[17] C. Moffat. Structures and Properties of Resonant Tunneling Device Circuits. Technical Report 96/4, Dept. of Physics and Astronomy, University College London, Image Processing Goup, 1006.

[18] C. Pacha and K. Goser. Functional Integration of Parallel Counters based on Quantum-effect Devices. In *Proc. of 15th IMACS World Congress, Berlin, Germany*, August 1997.

[19] W. C. B. Peatman et al. Novel Resonant Tunneling Transistor with High Transconductance at Room Temperature. *IEEE Electron Device Letters*, 15(7):236–238, July 1994.

[20] P. M. Petroff. Quantum Structure with Reduced Dimensionality and Quantum Functional Devices. *Journal of Future Electron Devices*, 5(2):5–19, 1994.

[21] E. Swartzlander. Parallel Counters. *IEEE Trans. Computers*, 22:1021–1024, 1973.

[22] M. Takatsu et al. Logic Circuits Using Resonant-Tunneling Hot-Electron Transistors (RHET's). *IEEE Journal of Solid-State Circuits*, 27(10):1428–1430, October 1992.

[23] P. v. d. Wagt, A. Seabaugh, and E. Beam. RTD/HFET Low Standby Power SRAM Gain Cell. *Technical Digest IEDM*, pages 425–428, 1996.

[24] S. Vassiliadis, J.Hoekstra, and S. Cotofana. Block Save Addition with Telescopic Sums. In *Proc. of the 21st EUROMICRO Conference, Como, Italy*, pages 701–707. IEEE Computer Society Press, September 1995.