Self-Timed SRAM with Smart Latency Bundling

Abdullah Baz, Delong Shang, Fei Xia, Reza Ramezani, Robin Emery,

Alex Yakovlev

Technical Report Series

NCL-EECE-MSD-TR-2010-161

November 2010

Contact: abdullah.baz@ncl.ac.uk

NCL-EECE-MSD-TR-2010-161

School of Electrical, Electronic & Computer Engineering
Merz Court, Newcastle University
Newcastle upon Tyne, NE1 7RU
UK
http://async.org.uk

Self-Timed SRAM with Smart Latency Bundling

Abdullah Baz, Delong Shang, Fei Xia, Reza Ramezani, Robin Emery, Alex Yakovlev
Microelectronic System Design Group, School of EECE
Newcastle University
Newcastle Upon Tyne, United Kingdom
{Abdullah.baz,delong.shang,fei.xia,reza.ramezani,r.a.emery,alex.yakovlev}@ncl.ac.uk

Abstract— **In energy-aware design, especially for systems with uncertain power sources, asynchronous computation loads which can function under variable power supply have many potential advantages. Fully safe asynchronous loads based on delay insensitivity (DI), however, tend to suffer power and size penalties. As a compromise, delay bundling has been widely used in asynchronous computation, but traditional delay elements have been shown to be unsuitable for bundling memory components, whose latency behaviour varies differently under variable Vdd from other types of logic. This paper proposes an intelligent delay bundling method for SRAM working under Vdd which unpredictably varies over a wide range (e.g. 200mV to 1V for 90nm technology), based on the principle of using matching delay bundling elements. A fully speed independent (SI) SRAM design is investigated in depth and a self-timed SRAM architecture using such SI SRAM cells as delay bundling elements is demonstrated through comprehensive analysis.**

Keywords-component; SRAM, failure rate, energy harvesting systems, timing variations.

I. INTRODUCTION

Energy harvesting (EH) has emerged as an alternative power source to batteries and EH systems continue to grow at a rapid pace. Such self-powered systems can be used where maintaining and replacing batteries are impossible, inconvenient, costly or hazardous. While EH systems overcome difficulties associated with batteries, they pose new challenges in the process of power delivery and management. Normally, in battery powered systems, where the supply voltage is relatively constant but ultimate energy is limited, the strategy is the traditional low-power systems approach, i.e. to decrease the power dissipation whilst maintaining the target system performance. However in EH systems, energy is infinite but power and voltage nondeterministic. The strategy is to maximize the performance for the supplied amount of energy and under all circumstances not consuming more than the harvested energy. Such a system must have the ability to work under a wide range of supply voltage since the harvester supplies nondeterministic power and Vdd which depends not only on the specification of the harvester but also on the amount of energy that can be harvested from the environment which varies from one place to another and even from time to time. Systems with these features can be called energy adaptive rather than low-power.

When the high variability of the harvester supply voltage is combined with process variations, which is becoming a significant factor in VLSI [1], timing variations would normally follow which could cause system malfunctions.

In on-chip computation systems, SRAM tends to occupy the majority, e.g. more than 90%, of the die area, and this trend is continuing as predicted by the ITRS [1]. It therefore has a great impact on the power, area, reliability and performance of almost all digital systems ranging from handheld devices to high-performance processors.

Normally SRAM works based on timing assumptions where the timing control block is in charge of regulating the timing relationship between different blocks in the SRAM system (memory cells, precharger, write driver, sense amplifier, row decoder and column multiplexer) to guarantee safe and successful reading and writing operation. Some examples of the timing hazards that may cause functional failures in the memory systems are: 1) if, during reading, the precharger has not been deactivated before opening the access transistor, the data in the memory cell may be upset, ending up with reading failure; 2) if, during writing, the access transistors have been de-asserted before flipping the data in the cell, writing failure occurs; 3) if, during reading, the access transistors have been turned off before one of the bit lines is discharged to the threshold values, access time failure occurs.

In general, the timing control block is built from basic gates, where functions and latencies depend on the switching of both the NMOS and PMOS devices. In contrast, the operation of the memory depends only on the NMOS device during reading to discharge one of the bit lines and on the PMOS device during writing to flip the memory cell [2]. This difference in operation dependency on the NMOS and/or PMOS devices makes the memory cells and the timing control block change in different ways with variations of the supply voltage. Other factors which exist in the SRAM but not in normal logic gates also contribute to this mismatch. These include the number of cells connected to the bit lines and the data values (0 or 1) in these cells. The worst case scenario happens when the data stored in all cells along the same bit line is opposite to the data in the cell being read or written, the leakage currents of all these other cells may decrease the level of I_{on} of the cell under operation, and make it indistinguishable from the level of I_{off} [17]. As the number of cells in the bit line

increases the problem gets worse. This discrepancy increases under PROCESS variations and becomes significant under the high variations of the supply voltage in the EH environment [11].

In synchronous systems, the timing control block of the memory system needs to be calibrated to work under the worst case conditions in order to eliminate the effect of the timing hazard [2]. Intuitively, a wide operational voltage range will result in a large difference between the worst and best case conditions. This in turn will lead to a significant conflict between safety and performance, where safe designs must incur heavy performance penalties.

In systems employing dynamic voltage scaling (DVS), voltage changes are fully under control. In such systems timing control blocks may be pre-programmed to change the clock frequencies according to the operating voltage carefully so that different worst cases are used for different voltage modes. For systems where the change of voltage is not fully under control but depends on the environment (e.g. EH systems), however, effectively determining the appropriate worst case dynamically in real time is needed. Moreover, in general, synchronous designs are not best suited for systems experiencing Vdd variable over a wide range not because of environmental factors.

On the other hand, asynchronous systems can potentially tolerate Vdd variations over a wide range, by trading speed for correctness automatically so that, in the long run, the systems demonstrate average case performance whilst maintaining correctness. However, for memory circuits there are additional complications.

The safest asynchronous circuits are delay-insensitive (DI) where the systems work correctly irrespective of latency behaviours of any component, including logic and wires. Such systems should in general work correctly under nondeterministic Vdd variations with average case performance. However, DI techniques usually require complex coding (e.g. dual-rail) and have power and size disadvantages [4]. For memory such as on-chip SRAM, even small unit penalties will get multiplied to intolerable scales. In any case, for many applications, DI solutions cannot be found [5].

Various approximations to true DI have been proposed. For instance, if the delays on wires are assumed to be zero or do not matter, systems can be designed so that any delay variations in the gates do not compromise correctness. Systems designed in this way are known as speed-independent (SI). SI systems tend to be easier and cheaper to design and implement than DI ones but in general still incur significant overheads. In addition, circuits of significant size usually include long wires whose latencies cannot be ignored, making SI only practical for small components.

A cheaper still approach to this problem is through delay-bundling, by making the latencies of timing control blocks automatically track the latencies of bundled logic when voltages change. This, the bundled data approach, is a standard method in computation logic design [6]. The common technique in bundled data design is to use conventional delay elements, such as inverter chains, for timing control. As delay elements usually consist of basic gates, similar to computation logic, the timing control blocks slow down or speed up more or less at the same rate as the bundled logic when Vdd changes. This ensures correct system operation over a potentially wide Vdd variation range.

In order to use delay bundling for memory systems, timing control blocks need to slow down or speed up at the same rate as memory cells when Vdd changes. Because of the operational discrepancy between normal logic gates and memory, such timing control blocks cannot be constructed entirely from basic gates. Alternative solutions have been proposed to improve the latency tracking for memory [2][3]. These employ a redundant column of memory cells, which are the same as the cells in the main memory bank, and use the signals generated from the cells in this column to enable and disable the sense amplifier and the row decoder to avoid timing hazards in the memory system. Although these solutions have shown good timing tracking, they have only addressed the timing hazards during the reading operation, leaving the writing timing hazard problem open. This is because generating completion signals for writing in SRAM type cells was assumed to be impractical.

Asynchronous design methods have been applied to SRAM design so that the latter can operate under a wide range of Vdd [3][7][8][9][11]. These designs implement various degrees of approximation to true DI with different degrees of success in reducing timing assumptions. The safest is the design proposed in [11] which resulted in a fully SI SRAM solution based on the standard 6T SRAM cell. This design has been shown to work correctly under a wide range of Vdd without timing assumptions. Since each SRAM cell is of limited size whose technology and layout are under the full control of the designer who can therefore easily keep wire delays predictable, SI approximates DI well in this application.

Failure rates of synchronous SRAM have been analyzed comprehensively in the literature [13][14]. However, all these analyses investigate the failure rate of the SRAM cell alone without including its timing control block because this is not needed under assumptions of synchronous operation. For asynchronous SRAM such combined system analysis is necessary, but such analysis was not carried out in [11].

Fully SI SRAM banks of large size multiply the overheads of SI, resulting in size costs not usually acceptable for energy adaptive systems.

This paper describes a smart latency bundling approach to asynchronous SRAM using fully SI SRAM cells as latency bundling elements for memory banks consisting of normal SRAM cells. The SI SRAM cells provide full completion detection capability for both reading and writing operations. This makes it possible to provide full latency bundling and completely remove the need for timing assumptions, resulting

in a fully self-timed SRAM with minimal overheads.

The contributions of this work are: 1) the SI SRAM design in [11] is thoroughly analysed under different values of Vdd; this analysis includes behaviour studies under all process corners, as well as failure rate investigations under PROCESS VARIATION assumptions, compared with its conventionally-bundled counterpart; 2) the validity and advantages of this type of SI SRAM having been demonstrated in the previous studies, a new self-timed SRAM system using it in an intelligent latency bundling scheme is designed and implemented; 3) the new self-timed SRAM method is demonstrated through further comparative studies.

II. Self-timed SRAM with Smart Latency Bundling

In general, for bundled data designs, the timing control circuits which provide the latency bundling tend to be much smaller than the bundled logic; otherwise the designer might as well choose to implement the main logic directly in SI or even DI. Conventional latency bundling elements are therefore usually based on the lightest possible basic logic units such as inverters. In memory banks, reading and writing operations usually involve multiple cells each time along an entire row. This can be bundled using a single memory cell in the timing control block. This is schematically shown in Figure 1. In this scheme, although the timing control block is heavier than one regular memory cell, it is much lighter than the entire row in memory banks with reasonable word lengths. This latency bundling method is "smart" or "intelligent" because it is based on extracting the relevant latencies in the most accurate manner possible – from components of exactly the same type as the bundled logic.

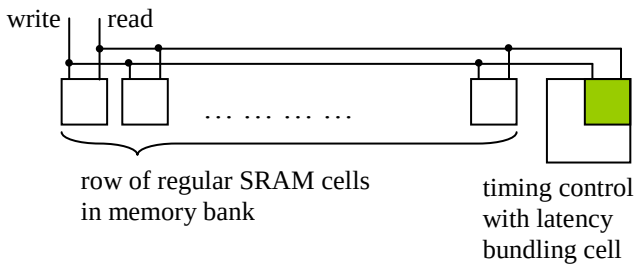

Figure 1. Timing control of SRAM row using latency bundling cell.

III. SI SRAM Cell Operations

The SRAM cell used for latency bundling in the timing control block needs to provide completion signals for all row operations, i.e. both reading and writing. Those used previously cannot generate writing completion [2][3] therefore writing had to depend on timing assumptions. The SRAM design in [11] is fully SI, making it possible to extract all relevant completion information from such a cell.

An example of SI SRAM with a normal 6T cell controlled by an SI controller is shown in Figure 2. It consists of a conventional 6T SRAM cell with writing and reading operations all controlled by an SI asynchronous controller.

The signal transition graph (STG) [10] specifications for reading and writing operations of the SI asynchronous controller are shown in Figures 3 and 4 respectively. These STGs specify completely sequential operations for both reading and writing, which are directly translated from synchronous SRAM operations by replacing clock signals with the appropriate corresponding handshakes.

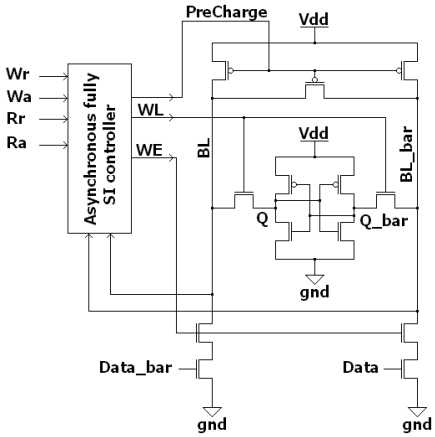

Figure 2. 6T SRAM cell with fully SI controller.

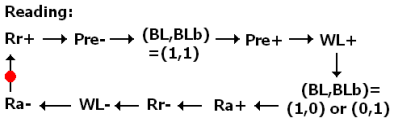

Figure 3. STG specification of the reading operation.

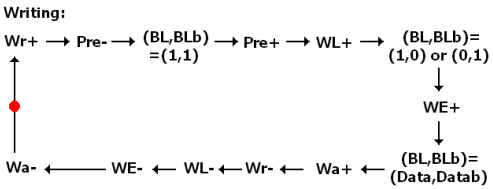

Figure 4. STG specification of the writing operation.

One of the possible realizations of the previous STGs is shown in Figure 5.

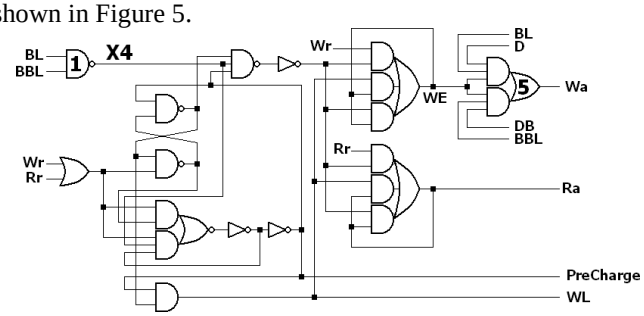

Figure 5. Possible realization of the controller.

A fully SI SRAM bank can be implemented by using one such controller to manage the whole SRAM bank. Normally the cells in each row form a word, which are usually written and read together. So, to control each row (word), an amended decoder is required which is a normal address decoder

triggered by WL to generate each individual word selection signal. The block diagram of the whole system is shown in Figure 6. As the address is stable during reading and writing, the change does not affect the SI property of the entire circuit. In addition, as each word contains multiple bits, gate 1 needs to be replicated. The number of the replication equals the number of bits in a memory word (the number of cells in a row). The inputs of each replicated gate are a pair of bit lines corresponding to each bit of the memory word. All outputs of the replicated gates 1 are collected in a C element. The output of the C element is used to replace signal x4. Gate 5 is also replicated. All outputs of these replicated gates are collected in another C element and the output of this C element is the new Wa signal. In this way complete reading and writing cycles are controlled in a closed-loop fashion with the bit lines from the SRAM cells serving as feedback signals into the controller.

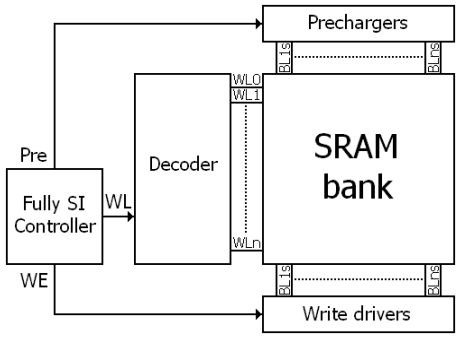

Figure 6. Possible realization of the system.

All descriptions in this section are adapted from [11].

IV. Corner Analysis of the SI SRAM

In this section we further explore the behaviours of this SI SRAM design in the context of constructing completely SI SRAM banks, because detailed analysis is lacking in [11]. This is done by implementing a fully SI SRAM bank and setting it into a testing chip. The structure of the chip is shown in Figure 7. The general function of this chip is to write, read and compare the written data with the read data in an asynchronous and autonomous manner without any interactions with the user or environment.

The design of this testing chip facilitates on-chip self test or prototyping where the SI SRAM may be tested on the same chip with other circuits, reducing the availability of input and output pins. In addition, the testing may be run at a wide range of Vdd variance, where the tested circuits could distribute a wide range of speed changes, making it impractical to depend on input and output pins to extract all the information. In this design, therefore, the tested RAM is surrounded with its entire test operating environment on the same chip.

This chip contains a couple of asynchronous counters to generate address and data and three D elements to manage the write and read cycle handshakes. The D element is a popular handshake interface circuit, which encloses a slave handshake inside a master one [12]. The self-timed counters are based on the design found in [12]. There is also a comparator, based on an XOR gate triggered by the a1 signal of the second D element, which determines the consistency or the lack thereof between the written and read data. The outputs of all XOR gates are combined by a C element to provide the consistency confirmation signal. If there is at least a single bit of disagreement, a negative result is returned.

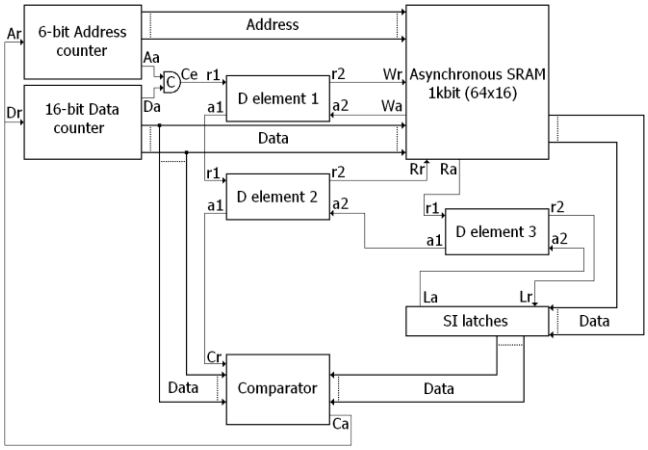

Figure 7. SRAM testing chip.

The signal flow of the chip is as follows: The first counter generates the address and sends its acknowledgement signal to the C element. The second counter generates the data and sends its acknowledgement signal to the C element. Once both counters settle, the C element sends its acknowledgement signal to the first request (r1) of the first D element. The first D element working as a writer sends the Wr signal to the controller in the SRAM to request writing. Once writing is finished, the controller sends the Wa signal to the first D element. The first D element withdraws the Wr signal and sends a request signal to the second D element. The second D element as a reader sends an Rr signal to the controller in the SRAM to request reading. Once reading is finished, the controller sends its Ra signal (see Figure 5) to download the data to the reading SI-latches. Here the third D element is used for this data download to guarantee the data is stable during processing. The reading SI-latches latch the data from the SRAM and then the third D element passes the control (Ra's acknowledgement) to the second D element to report the completion of reading. The second D element withdraws the Rr signal and sends a request signal to the comparator. The comparator compares the data from the SI latches with the data from the second counter and if they are equal, it sends an acknowledgment signal Ca to the counters to start another round. This means that a round of writing following by reading has been performed correctly as the data written was correctly read. If an inconsistency is discovered, however, the entire circuit stops at the comparator. The testing chip operation is captured in the STG in Figure 8.

This chip has been implemented in UMC90nm process technology and tested under all process corners and a wide

range of supply voltages from 1V down to 400mV. Meanwhile the latency of writing, reading and checking, of the whole memory bank 1kbit (64x16), has been measured and the results are shown in Figure 9. In terms of latency the order of the worst corners is: SS (slow NMOS, slow PMOS), SNFP (slow NMOS, fast PMOS), FNSP (fast NMOS, slow PMOS), TT (typical NMOS, typical PMOS) and FF (fast NMOS, fast PMOS).

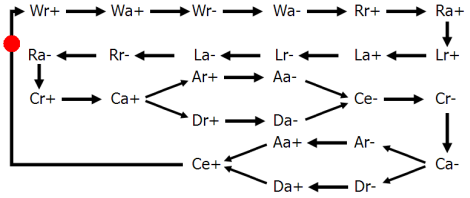

Figure 8. STG specification of the SRAM testing chip.

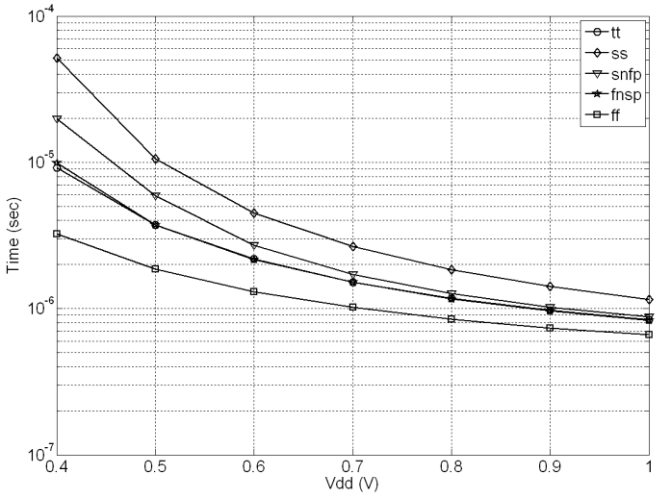

Figure 9. Corner analysis of the SRAM chip.

At 1V the chip can write, read and check the whole memory bank in 1.1µsec, 662nsec and 836nsec for the worst, best and typical cases respectively. At 400mV these operations can be completed in 51µsec, 3.2µsec and 9.2µsec for the worst, best and typical corner respectively.

Under these testing conditions, all simulation cycles completed without incident. This demonstrated that such an SI SRAM bank works correctly under all process corners across a wide range of Vdd variance.

The SI SRAM is further compared with two other types of SRAM, both of the same 6T basic structure. One is a conventionally bundled SRAM whose timing is controlled by a chain of inverters. The other is a fully synchronous SRAM controlled by a clock. Write (Figure 10) and read (Figure 11) energy consumptions are compared among the three SRAM banks. The amounts of energy reported pertain to single write and read operations on a single row of cells and the figures are in logarithmic scale. Clearly the SI SRAM shows great advantages over the other technologies in terms of energy consumption under all Vdd values.

The high energy consumption of the conventionally bundled synchronous SRAM partially results from the bundling inverter chain timing control being designed to work under the worst case appropriate for 190mV, thus being far from optimized at higher Vdd values. The even higher energy consumption of the fully synchronous SRAM is the result of running it always under a clock frequency suitable for the worst case of 190mV, which means that its operations take a much longer time to complete at higher Vdd than both of the other SRAM solutions (in comparison, the frequencies of conventionally bundled SRAM are 250KHz for 190mV and 125MHz for 1V, in other words it was able to run much faster at higher Vdd). This is the only way in which the three solutions can be fairly compared, i.e. all three have to work across the entire Vdd range without modifications half way through the simulation.

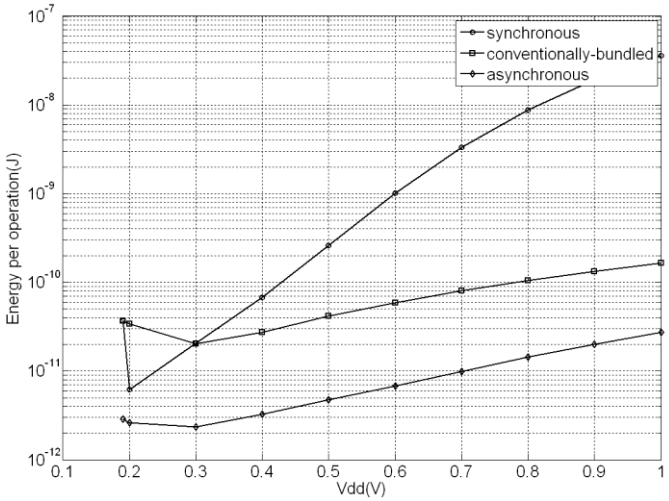

Figure 10. Write energy consumption comparison between SI and conventionally bundled synchronous SRAM banks.

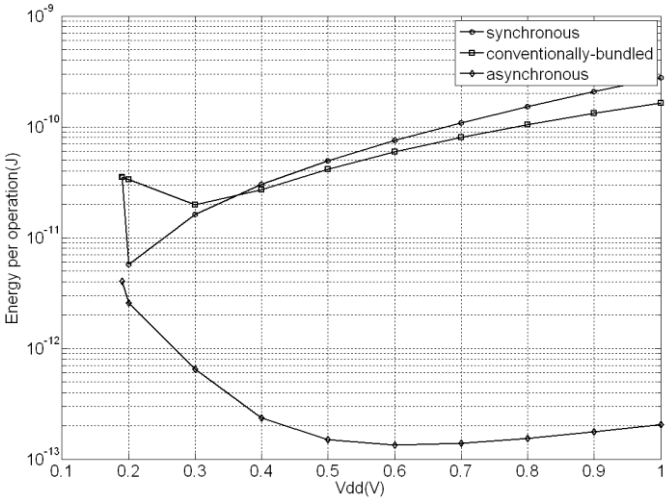

Figure 11. Read energy consumption comparison between SI and conventionally bundled synchronous SRAM banks.

All simulations in this paper, unless otherwise noted, are carried out in the following manner. For one round of writing, initial data values are stored in the memory bank and the entire

bank is written once, i.e. every word (row) is written once, with inverse data obliging the flipping of bits in every cell. For one round of reading, initial data values are stored in the memory bank and the entire bank is read once. Multiple rounds of writing and reading may be run in each simulation, especially in Monte Carlo analysis. Per row data, if presented, are the appropriate statistical mean values. Different initial data values are chosen and the worst case results reported, if experiments show dependencies to initial values. In general, it has been found that initial data values make very little difference (always less than 5%) to the numerical results.

V. Failure Rate Analysis of SI SRAM and Comparisons with Syncrhonous SRAM

A fully SI SRAM bank requires complex C elements with large numbers of inputs for the timing controller managing the timing of each row. These in turn potentially require large numbers of wires with significant and different lengths. The complex C element completion indication may cause significant latency penalties for the SRAM and the long and diverse wires may cause poor approximation to DI with safety penalties. On the other hand, a single fully SI SRAM cell does not need complex C elements or long wires and could be implemented compactly to achieve good DI approximation. Such a cell, from which completion information of all write and read steps can be easily extracted, can serve as the timing control block in the scheme of Figure 1.

Whether this SI SRAM design is suitable for use in this way, however, depends on its robustness and completion indication performance under target operating conditions. The soundness of this design can be seen from the results from the previous section, but a single cell should perform better than an entire bank, given the compromises required in the complex indication of the latter. For instance, the corner analysis in the previous section showed that the SI SRAM bank stopped fully working below 400mV, which casts some doubt as to whether bundling with this type of cell can work into the subthreshold region. As in-depth analysis of this design is lacking in [11], here further investigations are carried out before it is put to use in latency bundling.

The key aspect of SRAM operation is the timing relationship between the SRAM cells and the timing control block. For instance, in a fully SI SRAM bank, the feedback mechanism between the SRAM cells and the timing controller means that a full cyclic control is implemented and all temporal manifestations of process variation should be tolerated in theory. In synchronous SRAM solutions, however, there is no feedback and the system works in open loop. Variations need to be tolerated by worst-case condition estimations. In the scheme of Figure 1, the system works partially in closed loop. The bundling will fail if the timing control cell fails. The integrity of the timing control cell is therefore crucial.

In contrast to conventional SRAM analysis in the literature, where cells are tested under fixed clock inputs [13][14], here we investigate the entire SI SRAM cell including the 6T cell itself and the timing controller together. In keeping with conventional SRAM cell analysis and with the vision of Figure 1, the studies in this section cover a single cell only. A single cell in this context is the 6T SI SRAM cell in the form of Figure 2 where the timing controller is in the form of Figure 5, which is not shared with any other cells. Since the controller only needs to manage a single cell rather than an entire row, no replication of gates and complex C elements are needed.

This section concentrates on the investigation of failure rates of such an SI SRAM cell compared with those of a corresponding conventionally-bundled 6T SRAM cell. In a conventionally-bundled 6T SRAM the timing control block consists of a chain of inverters where the timing relationships between different blocks may rely either on the clock phases or on several delay lines triggered by the same clock phase. Here we chose to build our timing control block based on the clock phases.

Both cells are constructed in UMC90nm process technology using the same logic when applicable and have been tested under process variations of 6-σ (which covers 99.99% of the entire variation distribution) at different supply voltages using Monte Carlo methods with a thousand samples. Instead of showing the failure rate of the reading and writing operation, we define the functional yield as the proportion of non-failure occurrences from the distribution. Here, if the functional yield of the operation is below the constraint of 99.9%, the operation is regarded as a failure.

In order to provide fair comparative analysis the timing control block of the conventionally-bundled SRAM has been calibrated to work in the same range of supply voltage as the SI SRAM, from 190mV up to 1V. This calibration causes the bundling inverter chain to fit the worst case (at the lowest Vdd) and increases the average operation time, thus reducing the failure rate. In spite of this the SI SRAM shows better functional yield results than its conventionally-bundled counterpart. The numerical results are described in the following sub sections.

A. Reading Failure Rate Analysis of the SRAM cells

The reading operation consists of two consecutive processes, precharging the bit lines and opening the access transistors to discharge one of the bit lines. The results of the Monte Carlo analysis are shown in Table I for the conventionally-bundled and SI SRAM cells.

The data shows that the reading operation in the conventionally-bundled SRAM fails for all Vdd values lower than 600mV. However, in the SI SRAM the reading operation works from 1V down to 300mV with a functional yield of 99.9% and failure happens in the very low Vdd region of 200mV or below. This demonstrates the suitability of the SI SRAM for working under low Vdd into the subthreshold region. The minimum energy per operation point, which is very important for energy adaptive computing, can usually be found between 300mV and 500mV [15], where the SI SRAM

works fine whilst the conventionally-bundled SRAM does not.

TABLE I. FUNCTIONAL YIELD OF THE SRAM DURING READING

VDD(V)	*Functional Yield %*	
	Bundled	*SI*
1.00	99.9	99.9
0.90	99.9	99.9
0.80	99.9	99.9
0.70	99.9	99.9
0.60	99.9	99.9
0.50	99.8	99.9
0.40	98.1	99.9
0.30	90.5	99.9
0.20	81.7	94.8
0.19	80.7	86.2

B. Writing Failure Rate Analysis of the SRAM cells

Writing operation consists of three processes, precharing the bit lines, opening the access transistors to enable the write driver to discharge one of the bit lines, and flipping the cell.

Table II shows the results of the Monte Carlo analysis of the SRAM writing function yield for the conventionally-bundled and SI SRAM cells.

TABLE II. FUNCTIONAL YIELD OF THE SRAM DURING WRITING

VDD(V)	*Functional Yield %*	
	Bundled	*SI*
1.00	99.9	99.9
0.90	99.9	99.9
0.80	99.9	99.9
0.70	99.9	99.9
0.60	99.9	99.9
0.50	99.7	99.9
0.40	97.6	99.9
0.30	87.9	93.7
0.20	72.8	58.0
0.19	71.8	51.0

Again whilst the conventionally-bundled SRAM stopped successful writing below 600mV, the SI SRAM continues down to the subthreshold region and the minimum energy per operation point. Under very low Vdd values (190~200mV), it was found that the SI cell fails more often because it is not able to generate the appropriate acknowledgement signals after writing, partly because of the uncertain effects from long vertical bit lines. Up to the writing of the data bit, it leads the conventionally bundled design in the rate of success.

This has interesting implications to the problem of reliability. Reliability is not an issue for either conventionally bundled or SI SRAM when Vdd is high. When Vdd is in the middle of the working range, down to about the minimum energy per operation point, the SI SRAM has clear advantages over the conventionally bundled design. Lower down still, the situation is reversed with the conventionally bundled SRAM showing an advantage over the SI solution. Interestingly, this lack of reliability at very low Vdd has nothing to do with the main function of memory, which is to maintain data storage. The reliability is affected by an inability to indicate successful writing after the fact.

In general, open-loop systems have higher reliability than closed-loop ones simply because the latter depend on feedback indication to progress, adding one element which can fail and whose failure causes catastrophic consequences (the system stops in the case of fully closed-loop implementations). The SI SRAM is "caught out" by this dependency on feedback which fails earlier than the main function. On the other hand, in engineering implementations, some mechanism of optionally breaking the dependency on feedbacks (e.g. stop listening to acks when Vdd drops beyond some value or stop waiting after some set time) can be used to eliminate complete system stoppages because of indication failure.

Although a full SI SRAM bank may not work properly under all variation conditions when Vdd goes below 400mV, a single cell is shown to be more robust.

VI. SMART LATENCY BUNDLING FOR SELF-TIMED SRAM

The in-depth investigations in the previous sections showed that the SI SRAM design, especially in single-cell form, robust under a wide range of Vdd and wide process variation assumptions. Therefore a single SI SRAM cell of this design is suitable as timing control for a self-timed SRAM of the form of Figure 1 to supply the latency bundling.

A bank of such a self-timed SRAM bundled with SI cells will include one bundling cell in each row. For consistency, implementation and layout practicality, and to compensate for the effect of leakage currents through the bit lines, these timing control cells should be put to the same position in each row. In other words, in a bank of such memory, one of the columns should be designated the timing control / latency bundling column. The delays of cells in this column must correctly bundle the delays of the whole memory bank under all conditions. It has been shown in [16] that due to interconnect latencies the furthest column from the decoder is slower than the other columns. This fact suggests that the furthest data column may be implemented with SI cells to serve both as the far end bits of memory words and as the timing control units. This approach is valid in purely timing assumption based design, as timing assumptions can be made data-independent.

However, in self-timed design, data independence cannot be assumed and needs to be verified. Intuitively, the worst case, where writing takes the longest time, is when the data in the memory cell needs to be flipped. An experiment is designed to clarify this matter. In this experiment, one normal SRAM cell is compared with one fully SI SRAM cell. The latency between writing start and data fully settling in the normal SRAM and the latency between writing request and writing acknowledgement in the SI SRAM are compared. In the bundling approach in the form of Figure 1, the writing acknowledgement in the SI SRAM is used to indicate the completion of writing, thus implying that data has settled in the row, including in the other normal cells. Experiments across all process corners were carried out for all data combinations.

Unfortunately, it has been found in these experiments that

when the normal cell has a bit flipping in writing and the SI cell does not, the writing acknowledgement signal from the SI cell comes before the written bit in the normal cell settled in all process corners at room temperature, and the difference in time cannot be covered by the additional interconnect delay of the longer wires. This means that bundling fails when the SI cell used for bundling does not experience bit flipping under writing. These experiments have also shown that when the SI cell experiences bit flipping, its acknowledgement signal always comes a safe distance after the data in the normal cell has settled.

These experiments highlight an important issue in the philosophy of bundling. The proposal of Figure 1 specifies that the bundling unit should be structurally equivalent to the bundled units. In fact it must exhibit behavioural equivalence in addition to structural equivalence. As a result so long as at least one of the cells in a word write flips (a high probability event in an SRAM row with a large number of bits), the bundling cell must display this behaviour and flip too. However, data dependency means that this is very unlikely.

Our solution is to construct the entire data memory from normal cells, and append an additional column of "professional" latency bundling SI cells to the far side of the furthest data column in the bank, where the bundling column will have the longest interconnect from the decoder. In this timing control column, whose stored data values have no operational functionality, alternating bit values are always written to cells to ensure that bit flipping happens in every write. To do that we need to read the data from the column, invert the data and then write it back to the column. Fortunately, the asynchronous controller in Figure 2 incorporates a reading action in its writing process (cf. Figures 3 and 4). Therefore there is no need to change the main parts of the controller, but the writing STG specification must be modified from the shape in Figure 4 to that shown in Figure 12 for the following reasons.

The STG in Figure 4, although correct for use in a system where all cells are SI, is not suitable for a bundling cell. For example, if the data being written into a memory cell is the same as the data already stored in the cell, according to the specification of the STG in Figure 4, as a round of reading is included in the writing process, so the (BL, BLb) presents the data stored in memory. As a result, the Wa+ signal will be generated as quickly as possible, and then the other control signals, such as WL, WE, will be withdrawn. This maximizes speed and reduces energy consumption, as the actual writing of a bit into the cell is not performed in such a case, but for a bundling cell, this speed-up is inappropriate.

Figure 12. Writing sequence STG for latency bundling cell.

As a result the specification for a bundling cell timing controller, derived from Figure 4, is developed (Figure 12). Here a forced writing of a different data bit is used to eliminate temporal inconsistencies. After the reading action in the writing process is complete (indicated by WE+), the data just read (the current stored data in the memory) is written into a storage, for example an SI latch, by Lr+, and after this writing to a latch is completed, La+ is generated. The resets of La and Lr immediately follow, and the complementary bit in the SI latch, which is the inverse of the original data bit, is written back into the cell as new data. This makes the data being written permanently different from the data stored in the cell. As the data being written into the latch needs to be stable during write back, the Lr and La signals should be reset before data writing. This is as shown in Figure 12, Lr+ followed by La+ and Lr-, and La-.

Based on this STG the controller must be updated as shown in Figure 13 with additional components. As the additional requirements do not affect the main part of the control, we have retained the design in Figure 5 and manually introduced the additional components to cover the difference between Figure 4 and Figure 8. These include a D element to manage the handshakes and an SI latch to store and invert the bit data. For the D element, {WE, WE1} is the master handshake and {Lr, La} is the slave handshake. The D element implements the WE+ → Lr+ → La+ → Lr- → La-→ WE1+ → WE- → WE1- handshake sequence.

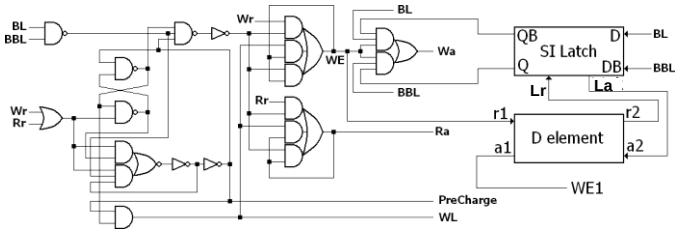

Figure 13. Latency bundling cell controller design.

The new controller is implemented in UMC 90nm CMOS technology in the Cadence toolkits. The operation of the new controller has been verified by SPECTRE simulation and it is fully functional from 1V down to 190mV.

The scheme of Figure 1 is designed for a bundling cell with a controller to manage the timing of a single row. In a multi-row bank, where only one row can be written or read at any moment in time, only a single bundling cell may be working at any time. This means that, whilst the bundling 6T cell should reside next to the row bundled by it for best layout consistency in latency, the timing controller in Figure 13 does not need to be replicated for every row. This is similar to the arguments leading to Figure 6.

A 1Kb (64×16b) bundled SRAM bank is implemented using one extra bundling SRAM per row as timing control blocks in the scheme of Figure 14. Each row in this bank therefore consists of 16 regular 6T SRAM cells plus one latency bundling SI SRAM cell at the far end. The latency bundling SI SRAM cells are controlled by the new controller

in Figure 12, with a single controller serving all 64 bundling cells. The size overhead compared with normal synchronous SRAM is therefore minimal.

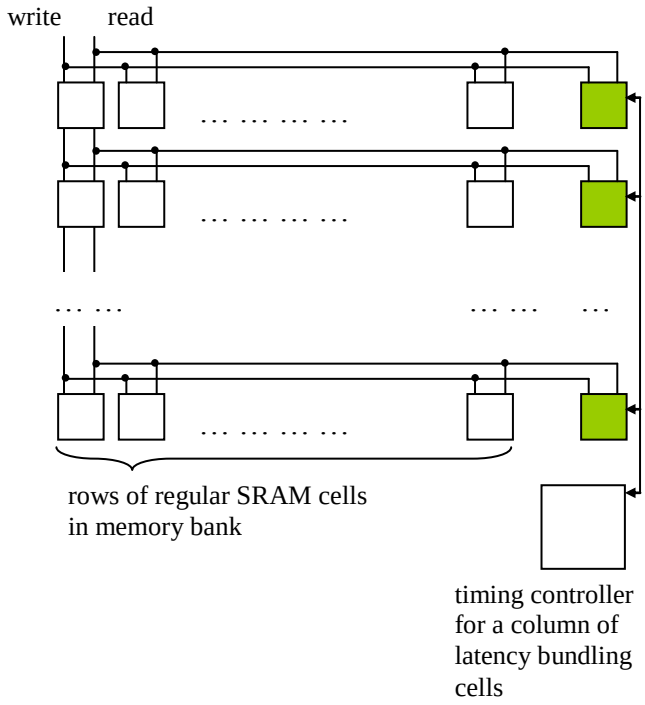

Figure 14. Smart-bundled self-timed SRAM bank.

A comparison is made between this bundled SRAM bank and a fully SI 1Kb SRAM bank which is constructed using the previous controller.

TABLE III. PERCENTAGES OF LATENCY AND ENERGY SAVED BY USING THE NEW TECHNIQUE

VDD(V)	*Time Saving %*		*Energy Saving %*	
	Writing zero	*Writing one*	*Writing zero*	*Writing one*
1.00	6.07	10.31	13.39	14.00
0.90	7.40	12.14	13.38	14.85
0.80	10.27	14.04	14.57	14.68
0.70	12.74	16.04	14.48	15.56
0.60	15.76	18.94	15.31	15.76
0.50	19.61	22.45	16.92	16.71
0.40	25.17	28.18	23.71	24.88
0.30	33.83	34.66	39.24	39.88
0.20	33.52	29.23	34.52	30.72
0.19	42.27	36.78	43.04	38.06

VDD(V)	*Time Saving %*		*Energy Saving %*	
	Reading zero	*Reading one*	*Reading zero*	*Reading one*
1.00	20.99	21.56	22.15	23.72
0.90	22.01	20.72	22.65	21.32
0.80	22.11	22.16	21.79	23.17
0.70	22.71	22.69	22.48	22.35
0.60	23.50	23.60	21.13	23.30
0.50	25.29	24.81	23.45	22.97
0.40	28.67	27.74	24.88	25.02
0.30	31.87	35.32	31.49	33.23
0.20	42.78	41.63	43.86	42.83
0.19	48.44	50.95	49.61	52.00

The latency and energy consumption of SRAM banks using the new and old controllers have been measured and the figures in Table III show the percentages of latency and energy savings the bundled technique can save over the fully SI method across different values of Vdd.

Under high voltage, the smart bundled self-timed SRAM can save up to 6% of the writing time and 13% of the writing energy. This saving increases as Vdd decreases to reach up to 30% of the writing time and 40% of the energy in the subthreshold region. The reading savings are even greater across the board. In view of Figures 10 and 11 which showed massive energy savings realized by the fully SI SRAM over SRAM with higher degrees of synchrony, these data represent further savings realized by the smart bundled self-timed design, and demonstrate the suitability of its use in energy adaptive systems across a wide voltage range extending to the subthreshold region.

VII. CONCLUSION AND FUTURE WORK

Computation loads which work based on absolute timing assumptions, such as synchronous systems, have difficulties when powered by nondeterministic power supplies and under process variations. This is because worst-case assumptions needed to ensure correctness imply operational inefficiencies. In contrast, fully asynchronous designs, for instance DI or approximately DI ones, by removing timing assumptions and regulating the data flow of the circuits based on the actual speed, work well under nondeterministic power supplies and in the face of temporal manifestations of process variations.

DI and its best approximation, SI, however, usually imply high costs and other limitations on the type of designs they can cover. For such systems as memory this problem is most acute, as memories have large numbers of replicated units where unit overheads are multiplied. Memories also necessitate long wires of different lengths, making SI designs problematic.

Latency bundling based on relative timing is a practical compromise between fully synchronous and fully SI or DI designs and has been widely used in asynchronous computation systems. In order for such a scheme to work well, the latency bundling timing control circuitry need to behave similarly to the main system whose timing is being bundled. For regular computation logic simple delay elements such as inverter chains work well as latency bundling elements because both are based on basic logic gates.

Memories such as SRAM present a special challenge to this approach because it by and large does not behave similarly to basic gates under nondeterministic power supply and process variations.

In this paper we propose an intelligent or smart latency bundling philosophy which dictates that the latency bundling units need to be chosen for appropriate equivalent behaviour when the operation environment changes. For SRAM the best latency bundling element should be SRAM.

A fully SI SRAM design is explored in detail demonstrating its capabilities of working under a wide range of Vdd and

process variation conditions. An on-chip testing method for asynchronous memories is developed and demonstrated in this study of the fully SI SRAM. The fully SI SRAM is demonstrated to consume significantly less energy in both writing and reading operations compared with its fully synchronous counterpart.

A smart latency bundling unit, based on a single cell of such SI SRAM suitable for use in self-timed SRAM banks, is designed and comprehensively verified. A new self-timed SRAM design, using such bundling units for both reading and writing operations, is developed and thoroughly analysed. Comparative studies are carried out with fully SI and smart bundled solutions which demonstrate that the self-timed smart bundled design can realize additional energy and performance advantages over the fully SI solution.

The smart bundling method solves the problem of wrapping the latencies along horizontal rows. On the other hand, the charging and discharging of vertical bit lines could affect indication reliability at very low Vdd. This remains an interesting problem for further study.

The results of this paper represent significant advances towards the application of the practical method of latency bundling for asynchronous SRAM.

ACKNOWLEDGMENT

Abdullah Baz is supported by a scholarship from Umm Al-Qura University, Kingdom of Saudi Arabia.

This work was supported in part by the Engineering and Physical Sciences Research Council (EPSRC) under grant number EP/G066728/1 "Next Generation Energy-Harvesting Electronics: Holistic Approach," website: www.holistic.ecs.soton.ac.uk.

REFERENCES

[1] International Technology Roadmap for Semiconductors: http://public.itrs.net/.

[2] Amrutur, B.S.; Horowitz, M.A., "A replica technique for wordline and sense control in low-power SRAM's," Solid-State Circuits, IEEE Journal of, vol.33, no.8, pp.1208-1219, Aug 1998.

[3] Meng-Fan Chang; Sue-Meng Yang; Kung-Ting Chen, "Wide Embedded Asynchronous SRAM With Dual-Mode Self-Timed Technique for Dynamic Voltage Systems," IEEE Trans. CAS-I, vol.56, no.8, pp.1657-1667, Aug. 2009.

[4] Saito, H.; Kondratyev, A.; Cortadella, J.; Lavagno, L.; Yakovlev, A., "What is the cost of delay insensitivity?," Proc. ICCAD'99, San Jose, CA, pp. 316-323, Nov. 1999.

[5] Martin, A.J., "The limitations to delay-insensitivity in asynchronous circuits", In Willian J. Dally ed, Advanced Research in VLSI, pp.263-278, MIT press, 1990.

[6] Kearney, D.; Bergmann, N.W., "Bundled data asynchronous multipliers with data dependent computation times", ASYNC '97, pp.186-197, April 1997, Eindhoven, Netherlands.

[7] Nielsen, L.S.; Staunstrup, J., "Design and verification of a self-timed RAM," Design Automation Conference, 1995. Proceedings of the ASP-DAC '95/CHDL '95/VLSI '95., IFIP International Conference on Hardware Description Languages; IFIP International Conference on Very Large Scale Integration., Asian and South Pacific , pp.751-758, 29 Aug-1 Sep 1995.

[8] Sit, V.W.-Y.; Choy, C.-S.; Chan, C.-F., "A four-phase handshaking asynchronous static RAM design for self-timed systems," Solid-State Circuits, IEEE Journal of, vol.34, no.1, pp.90-96, Jan 1999.

[9] Dama, J.; Lines, A., "GHz Asynchronous SRAM in 65nm," Asynchronous Circuits and Systems, 2009. ASYNC '09. 15th IEEE Symposium on, pp.85-94, 17-20 May 2009.

[10] Rosenblum, L. Y.; Yakovlev, A., "Signal graphs: from self-timed to timed ones," In Proc. of international workshop on timed Petri nets, pp 199-207, Torino, Italy, July 1985.

[11] Baz, A.; Shang, D.; Xia, F.; Yakovlev, A., "Self-timed SRAM for energy harvesting systems," PATMOS'10, LNCS 6448, Grenoble, France, Sept. 2010.

[12] Varshavsky, V.I.; Kishinevsky, M.A.; Marakhovsky, V.B., Peschansky, V.A.; Taubin, A.R.; Tzirlin, B.S., Self-timed control of concurrent processes, Kluwer Academic Publishers, 1990.

[13] Mukhopadhyay, S.; Mahmoodi, H.; Roy, K., "Modeling of failure probability and statistical design of SRAM array for yield enhancement in nanoscaled CMOS," Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on, vol.24, no.12, pp. 1859- 1880, Dec. 2005.

[14] Mukhopadhyay, S.; Mahmoodi-Meimand, H.; Roy, K.; , "Modeling and estimation of failure probability due to parameter variations in nano-scale SRAMs for yield enhancement," VLSI Circuits, 2004. Digest of Technical Papers. 2004 Symposium on, pp. 64- 67, 17-19 June 2004.

[15] Ramadass, Y.K.; Chandrakasan, A.P., "Minimum energy tracking loop with embedded DC-DC converter enabling ultra-low-voltage operation down to 250mV in 65nm CMOS", IEEE Journal of Solid-state circuits, Vol. 43, No.1, January 2008.

[16] Amelifard, B.; Fallah, F.; Pedram, M., "Leakage Minimization of SRAM Cells in a Dual-Vt and Dual-Tox Technology," IEEE Trans. VLSI, vol.16, no.7, pp.851-860, July 2008.

[17] Calhoun, B.H.; Chandrakasan, A.P., "A 256-kb 65-nm Sub-threshold SRAM Design for Ultra-Low-Voltage Operation," Solid-State Circuits, IEEE Journal of, vol.42, no.3, pp.680-688, March 2007.