Minimizing the Power Consumption of an Asynchronous Multiplier

Yijun Liu, Steve Furber

Advanced Processor Technologies Group
Department of Computer Science
University of Manchester
Manchester M13 9PL, UK
{yijun.liu, sfurber}@cs.man.ac.uk

Abstract. As the demand for low power electronic products continues to increase there is a need for the designers of CMOS circuits to find ways to reduce the power consumption of their circuits. This paper introduces a practical approach to improve power-efficiency based upon the analysis of a breakdown of the power consumption of an existing design. The breakdown is used to identify the most promising subcircuits for improvement. A 32×32 asynchronous pipelined integer multiplier is used as a case-study. Following the proposed methodology, the redesigned multiplier uses less than 40% of the energy per instruction of its predecessor. An asynchronous latch controller is also proposed which is smaller and faster than previous 4-phase fully-decoupled latch controllers.

1 Background

The semiconductor industry has witnessed an explosive growth in very large-scale integrated circuits over several decades. Only recently has the rapid progress of CMOS technology made integrating a complete system-on-a-chip possible. The rapid increase in the number of transistors on a chip dramatically improves system performance. However, the improvements in transistor density and performance also result in a rapid increase in power dissipation, which has become a major challenge in the use of modern CMOS technology. In order to prolong battery life and reduce the need for expensive cooling systems, significant effort has been focused on the design of low-power systems for portable applications, digital signal processors and ASIC implementations. Power consumption has emerged as an important parameter in VLSI design and is given comparable weight to or, for some applications, more weight than performance and silicon area considerations.

Dynamic power dissipation is the dominant factor in the total power consumption of a CMOS circuit and typically contributes over 80% of the total system power [1], although leakage is also becoming a major problem in deep sub-micron CMOS. Three factors determine the dynamic power dissipation of a CMOS circuit: the supply voltage, capacitance and switching activity. The supply voltage of a CMOS circuit is decided by the characteristics of the CMOS technology used to fabricate the circuit, so we will ignore techniques based on reducing V_{dd}.

Multiplication is an essential function in microprocessors and digital signal processing systems and contributes significantly to the overall system power consumption, thus low-power design is important here. A number of techniques have been proposed to minimize the power dissipation in multipliers [2][3][4]. These approaches can be divided into two categories. The first category focuses on high-level algorithms to decrease the switching activity. The modified Booth's algorithm [5] is the most popular algorithm for low-power multipliers, because it scans multipliers two bits at a time, thus decreasing the number of partial products by a factor of two compared to a one-bit-at-a-time algorithm and reducing the number of compressor cells. The modified sign-generate algorithm (MSG) [4] is another power-efficient multiplication algorithm; instead of extending the sign bit to the most significant bit of the array, the MSG algorithm needs only two sign-extension bits, thus saving power and silicon area.

The second category of techniques used in power-efficient multipliers put their emphasis specifically on details of the CMOS implementation. Different Booth's encoders and partial product generators have been proposed to minimize the number of glitches [4][6]. The structure of compressors is another area of interest. 3-2 compressors (CSAs), 4-2 compressors, and even 5-2 compressors are claimed to have their advantages in low-power design [7][8][9]. Pass-transistor logic [10] is often used because it is very efficient for building small compressors.

The low-power methodology used in this paper is presented in the context of a pipelined multiplier — the Amulet3 multiplier [11], but can be used to reduce the power dissipation of other pipelined systems. The remainder of the paper is organized as follows: Section 2 presents the context of the work by introducing the architecture and power analyses of the Amulet3 multiplier. Section 3 describes the low-power strategies derived from the analyses of Section 2. Section 4 gives the detailed circuit implementation of the new multiplier and the experimental results. Section 5 summarizes the low-power techniques used in the new multiplier.

2 The Amulet3 multiplier

Amulet3 [12] is a 32-bit asynchronous processor core that is fully instruction set compatible with clocked ARM cores. The Amulet3 multiplier uses a radix-4 modified Booth's algorithm [5] to decrease the number of partial products by a factor of two. An early termination scheme is also used to improve performance and save power. The early termination algorithm is based on the idea that if the multiplier is very small so that several of the most significant bits are all 1s or 0s, the multiplication can be sped up by ignoring these bits. Using a radix-4 modified Booth's algorithm and 4-2 compressors, 8 bits of multiplier are processed in each cycle and it takes 4 cycles to complete a multiplication. So for a 32-bit multiplier, if $s_{31}..s_{24}$ are all 0s or 1s, one cycle of the multiplication can go faster. Similarly, two or three cycles can go faster if the next one or two groups of eight bits are all the same as the top group. The multiplication cycles can be separated into two kinds — normal cycles and early termination cycles. In normal cycles, the multiplier operates as usual, but during early termination cycles the 4-2 compressors are idle and shift registers simply shift the output to the proper position. Consequently, power can be saved in early termination cycles.

The Amulet3 multiplier is an asynchronous multiplier, which cooperates with other parts of the Amulet3 microprocessor using an asynchronous handshake protocol. However, from the internal architecture point of view, the Amulet3 multiplier is a synchronous pipeline because its internal clock signal is generated by an inverter oscillator and this clock signal is distributed throughout the whole multiplier to provide the timing reference. The architecture of the Amulet3 multiplier is shown in Figure 1.

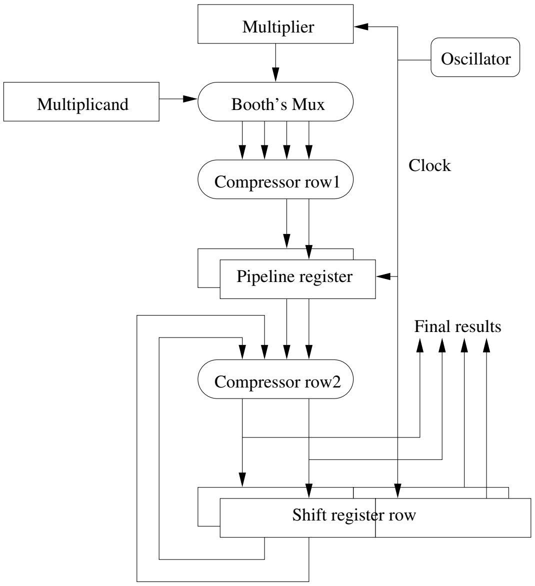

Fig. 1. The architecture of the Amulet3 multiplier

The Amulet3 multiplier does not use a tree or array architecture because of the tight silicon budget; instead an iterative architecture is used. Pipelining improves the speed of the multiplier by a factor of two.

In order to find the critical areas of power dissipation, we need to define some specific multiplication samples. The special features of the Amulet3 multiplier — a pipelined iterative multiplier with an early termination scheme — mean that its power dissipation is not constant. Multiplications without early termination consume the most power (the full load) and those with 3 cycles of early termination use the least power (the lightest load). Both situations must be analyzed.

The power dissipation contributions of the different components are presented in Figure 2(a) with a full load and Figure 2(b) shows the power breakdown with the lightest load.

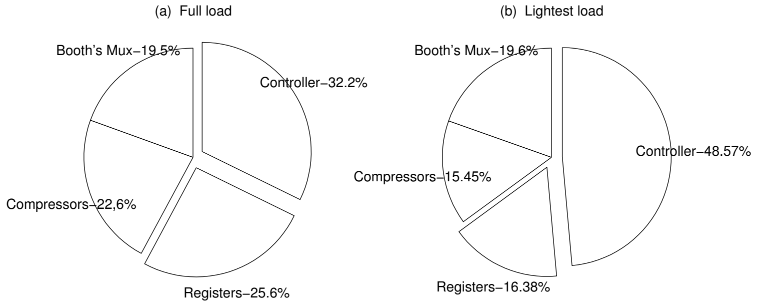

Fig. 2. The power breakdown of the Amulet3 multiplier

As can be seen from Figure 2, the control circuit uses the largest share of the system power, followed by the registers. The registers in the Amulet3 multiplier are substantial — they include more than 200 edge-triggered flip-flops which put a heavy load on the clock signal. Therefore large buffers are needed in the output of the clock generator to drive the registers, and these buffers themselves consume a lot of power.

The interesting phenomenon we found when analyzing the lightest load situation is the fraction of the power used by the Booth's Mux. It is no different from that on full load. We might expect that during early termination cycles the Booth's Mux is idle and it should use less power. Moreover, the share of the power used by the shift registers is the same as that under full load, while the total power is decreased, so the power percentage of the shift registers should be bigger than that under full load. The most likely explanation is that during early termination cycles the datapath, including the Booth's Mux and compressor rows, still does some unnecessary work. In another words, the datapath is not fully idle during early termination cycles.

3 Low power strategies

3.1 Asynchronous control

From the power analysis of the Amulet3 multiplier we can see that the power consumption of the edge-triggered registers and the control circuits is the crux of the problem. One efficient way to decrease the power consumption of the multiplier is to use smaller registers or latches and to simplify the control circuit. However, the TSPC register is very small compared to other edge-triggered registers. To get any smaller we must use level-sensitive transparent latches. With level-sensitive transparent latches, synchronous pipelines have only 50% occupancy.

Fortunately, with fully-decoupled asynchronous pipeline latch controllers [13], we can use normal level-sensitive latches to construct an asynchronous pipeline with 100% occupancy. As is well known, level-sensitive latches present half the capacitance of edge-triggered registers to their clock signals. Moreover, in asynchronous circuits several locally-generated signals are used instead of the global clock signal, which means that the load on the global signal is shared by several local signals and each local signal drives fewer gates and needs smaller driver buffers. As a result, changing the architecture of the Amulet3 multiplier from a synchronous pipeline to an asynchronous pipeline has the potential to reduce its power consumption.

The major cause of wasted power is unnecessary switching activity. It has been argued that asynchronous logic offers a more flexible power management strategy, which is that an asynchronous system only performs processing 'on demand' [14]. With asynchronous latch controllers we can easily isolate the datapath of the Amulet3 multiplier as soon it has finished performing normal cycles, so during early termination cycles only the shift registers are active to shift the output to the proper position, thus saving power by suppressing unnecessary activity.

The asynchronous latch controller used in the multiplier is shown in Figure 3(a). The latch controller is constructed from two C-elements [15]. The Signal Transition Graph (STG) description of the latch controller is illustrated in Figure 3(b). The dashed arrows indicate orderings which are maintained by the environment; the solid arrows represent orderings which the circuit ensures by itself. As we can see from Figure 3(b), the latch controller is normally closed, which means that the latch opens only briefly when a new stable data value is available on its inputs and it is closed in the idle state which eliminates unnecessary glitches. Normally-closed latch controllers are good for low power [16]. A normally-open latch controller, on the other hand, holds its latches in the transparent state when idle, so a glitch at the front of the datapath may propagate through the whole datapath. As a result, noise and glitches can easily cause unnecessary switching activity in the whole system and dissipate a lot of power. To achieve low power we should minimize the switching activity; the latches should therefore be closed as soon as possible after they finish passing data to avoid unnecessary activity throughout the system. The normally-closed latch controller can thus yield a significant power advantage in low-performance systems.

The latch controller is also fully-decoupled [13], and we can use normal level-sensitive latches to construct an asynchronous pipeline with 100% occupancy. Clocked logic requires edge-triggered registers to construct pipelines with 100% occupancy. Level-sensitive latches present half the capacitive load of edge-triggered registers to clock signals. As a result, level-sensitive latches controlled by the fully-decoupled latch controller shown in Figure 3 require about half as much area and power as edge-triggered designs. In the new multiplier, the pipeline registers are level-sensitive latches and the iterative registers are edge-triggered (because the iterative registers act both as pipeline registers and as shifters).

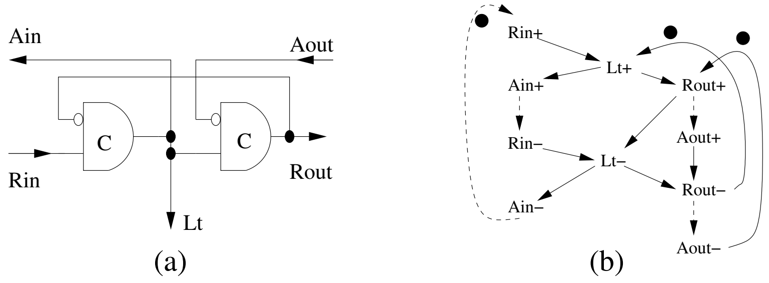

Fig. 3. A normally-closed latch controller

The multiplier is a 2-stage pipeline, so its control circuit includes three latch controllers, controlling the multipliers, the pipeline registers and the iterative registers respectively. The latch controller is not truly speed-independent, because we should not allow the input data to change until after *Ain−*, which indicates that the latch in the next stage has closed. Here, *Rout−* enables *Lt+*, violating the protocol at the next stage; a speed-independent latch controller should have *Aout−* → *Lt+* instead of *Rout−* → *Lt+*. However, since the delay of the logic block is much longer than the delay of a C-element, the new latch controller operates correctly despite this violation of speed-independence. As a result of this timing assumption, the proposed latch controller is smaller and faster than previous normally-closed and/or fully-decoupled latch controllers [13][17].

The shifter shifts the result of multiplication right 8 bits per cycle. Initially the least-significant 32 bits of the shifter are empty, so we can put the multiplier in the least significant 32 bits of the shifter to save an additional 32 bits of register for storing the multiplier.

We can further increase the speed of the latch controller by eliminating two inverters' delay as shown in Figure 4. With this latch controller, we get an improved performance because the driver delay is no longer included in the critical path of the control circuit. Again, the timing assumptions upon which this optimisation is based must be checked carefully.

3.2 Non-Booth's algorithm

We have argued [18] that the modified Booth's algorithm may have power-efficiency disadvantages as a result of its inverting operations: with the modified Booth's algorithm, partial products have a 50% probability of being $-1\times$ or $-2\times$ the multiplicand. These inverting operations introduce many transitions into the adder tree. This phenomenon is especially critical in the multiplication of two small integer numbers.

In order to overcome the disadvantage of the modified Booth's algorithm, we should avoid the inverting operations. We can use a much simpler algorithm by scanning the multiplier bit by bit. If the bit is 0, the respective partial product is 0; otherwise, the partial product is the value of multiplicand. This simple algorithm is not very efficient when the multiplier is negative; as a result of the two's-complement format, the multiplier sign bit will be propagated to the most significant bit. As a result, the multiplication has 32 partial products. Fortunately, we can modify this simple algorithm using the following equations:

$$a \times b = \overline{a} \times \overline{b} + \overline{a} + \overline{b} + 1 \tag{1}$$

$$a \times b = \overline{\overline{a} \times b + b - 1} \tag{2}$$

$$a \times b = \overline{a \times \overline{b} + a - 1} \tag{3}$$

We first check the sign bits of the two operands. If both multiplicand and multiplier are negative (say a and b), we invert both multiplicand and multiplier and send them to the non-Booth's multiplier. Then we add the result to $\overline{a}+\overline{b}+1$ to get the final result. If only one of the operands is negative, we invert it. We send two positive numbers to the non-Booth's multiplier and get the result. Then we invert the sum of the result, the positive operand and -1 to get the final result.

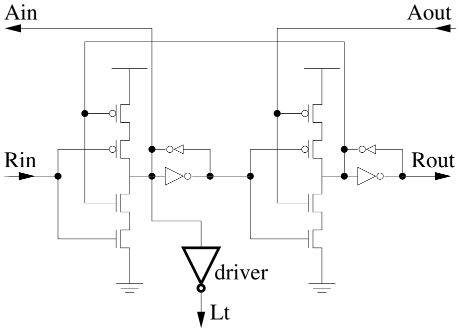

Fig. 4. A fast implementation of the latch controller

3.3 Split register

Multiplication operands have the characteristic of unbalanced distribution, which means that the least significant bits switch more frequently than the most significant bits. Some of the least significant bits toggle at approximately half the maximum frequency (uniform white noise). The most significant bits have a very low toggle rate. In our non-Booth's algorithm, all operands sent into the multiplier are positive, so we can draw the effects of data correlation on bit switching frequency as shown in Figure 5. The n in Figure 5 is about 8 [19]

Because of the unbalanced distribution of input vectors, we can split the 32-bit registers into small groups. We only drive the 'significant bits' and ignore a series of 0s on the most significant bit side. In our design, we split the registers

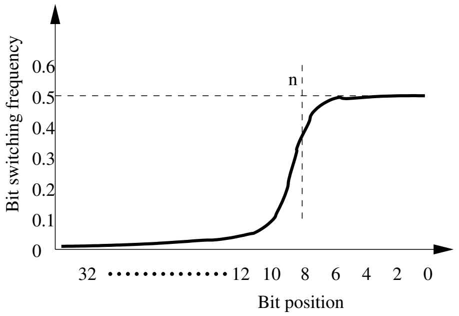

Fig. 5. The effects of data correlation on bit switching frequency

into 3 groups — $Bit_{31}..Bit_{16}$, $Bit_{15}..Bit_8$, $Bit_7..Bit_0$, as shown in Figure 6. We already have an 8-bit zero detector to decide the number of early termination cycles, so the only overhead incurred for splitting the registers is 2 clock gates. Testing the split registers by simulating the netlist we found a 12% energy saving. Because the netlist-based simulation does not include wire capacitance we think it likely that in a real circuit splitting the registers will offer a greater power saving.

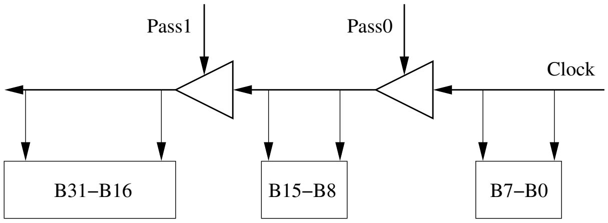

Fig. 6. 32-bit split register organization

The split-register technique is inspired by a low-power register bank [20] and can be used in any register- or latch-based CMOS circuit with an unbalanced distribution of input operands.

4 Circuit Implementation and Experimental Results

4.1 DPL 4-2 compressor

Using 4-2 compressors we can build a multiplier with a more regular structure than can be achieved with 3-2 compressors. A 4-2 compressor row reduces 4 partial products at once compared to only 3 partial products in a 3-2 compressor row. A 4-2 compressor can be constructed simply from two 3-2 compressors with four XOR-gates' delay. However, with careful design a 4-2 compressor can have only 3 XOR-gates' delay. So the multiplier using 4-2 compressors is faster than that using 3-2 compressors. Since the carry out signal is logically independent of the carry in, there is no carry propagation problem when several 4-2 compressors with the same weight are abutted into the same row; this is the key idea behind the use of the 4-2 compressor.

Pass-transistor logic is known to have an advantage over conventional static complementary CMOS logic in arithmetic circuits because it is well-suited to the construction of XOR-gates and multiplexers. Complementary pass-transistor logic (CPL) [10] and differential pass-transistor logic (DPTL) [21] both consist of two nMOS logic networks to increase the noise immunity which is potentially compromised by low signal swings. CPL and DPTL make full use of the advantages of nMOS transistors, which are faster and smaller than pMOS transistors. However, both of them have disadvantages in high performance and low power areas:

- The degraded output signals increase leakage power dissipation.
- Low signal swings decrease the speed of the circuit.
- A swing restoration circuit is needed in the output, which causes additional power dissipation.

Double pass-transistor logic (DPL) [22] retains the advantages of pass-transistor logic but avoids the drawbacks mentioned above because it provides full signal swings. The schematic of the DPL 4-2 compressor used in the new multiplier is shown in Figure 7. From the schematic we can see that the paths of *Sum* and *Carry* are well balanced, which decreases glitches thus minimizing unnecessary switch activity. The balanced delay also results in a lower worse-case latency.

4.2 Pipeline latches and registers

In this multiplier, because of the employment of the hybrid pipeline latch controllers, normal transparent latches are used to act as pipeline latches rather than edge-triggered registers. An edge-triggered register is often constructed from two latches with a common clock input, so the latency of an edge-triggered register is double the latency of a latch, and the register's capacitance attached to the clock signal is also doubled. As a result, this multiplier is lower power and faster than its synchronous counterparts which use edge-triggered registers.

Registers are used as a shifter to store the output of a multiplication. True single-phase clocking (TSPC) logic [23] is power-efficient when used to build the registers because a TSPC register is faster than a conventional static register and occupies less silicon area. Moreover, a TSPC register puts a smaller capacitance on the clock signal than a static register. PowerPC603 master-slave registers and latches [24] are well-known in the field of low-power design. In our experiment, we found that the PowerPC603 master-slave register is slightly better than the TSPC register when used in low-performance systems.

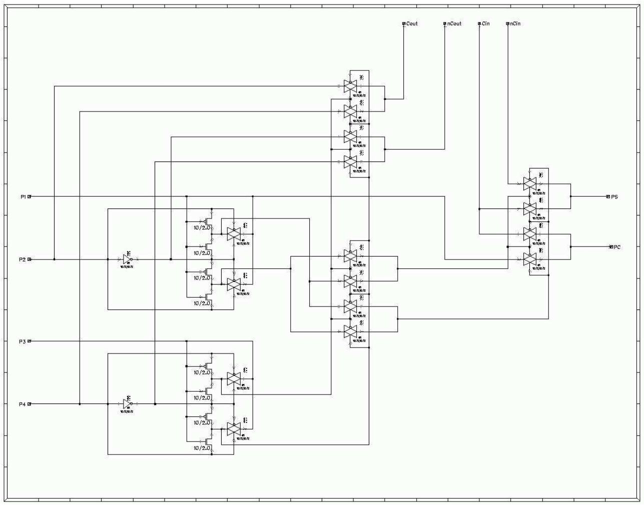

Fig. 7. The schematic of a DPL 4-2 compressor [11]

4.3 Experimental results

The new multiplier was compared with the original Amulet3 multiplier using HSPICE under the conditions of a 1.8 volt supply and 27 °C temperature on a 0.18 micron CMOS technology. The average power consumption of the new multiplier is 6.47 mW at 100 million operations per second, whereas the Amulet3 multiplier burns 16.17 mW at the same speed. Power dissipation alone is not enough to evaluate a circuit because we can simply make it very low power by reducing its performance. In this paper, *power* × *delay* or *energy per operation* is also used as a metric to evaluate the multipliers. Compared to the Amulet3 multiplier, the new multiplier is 10% faster than the Amulet3 multiplier because the delay of the transparent latches is less than that of edge-triggered registers, and we gain from the average performance of the two well-balanced pipeline stages. High performance is a side-effect of choosing asynchronous logic. Under the same conditions and with the same operands, the new multiplier uses only 37% of the *energy per operation* of the Amulet3 multiplier.

5 Conclusion

In the work described in this paper we succeeded in decreasing the power dissipation of an asynchronous pipelined iterative multiplier — the Amulet3 multiplier — starting from an analysis of the power breakdown. The new multiplier uses less than 40% of the *energy per operation* of its predecessor. The improvement in the low-power characteristics of the new multiplier is due to redesigning the control circuit and discarding large edge-triggered registers which contribute the most power dissipation in the Amulet3 multiplier. With the new asynchronous latch control circuits, more compact control circuits and smaller transparent level-sensitive latches are used in the new multiplier to minimize the load capacitance and remove the large driver buffer. The asynchronous control scheme causes the proposed multiplier to perform processing ‘on demand’, thus minimizing unnecessary switching activity. A low-power non-Booth’s algorithm is also used in this multiplier to avoid the inverting operations incurred by the modified Booth’s algorithm. Finally, to exploit the unbalanced bit switching frequency, we use a split register scheme to decrease the power dissipation of the large registers.

References

1. D. Soudris, C. Piguet and C. Goutis (eds). “Designing CMOS Circuits for Low Power”. Kluwer academic publishers, 2002.
2. T. Callaway and E. Swartzlander. “The Power Consumption of CMOS Adders and Multipliers”, in “Low power CMOS Design”, A. Chandrakasan and R. Brodersen (eds), IEEE Press, 1998.
3. L. Bisdounis, D. Gouvetas and O. Koufopavlou. “Circuit Techniques for Reducing Power Consumption in Adders and Multipliers”, in [1].
4. R. Fried. “Minimizing Energy Dissipation in High-Speed Multipliers”, International Symposium on Low Power Electronics and Design, 1997.
5. A. D. Booth, “A Signed Binary Multiplication Technique”, Quarterly J. Mech. Appl. Math., vol. 4, part 2, pp. 236 240, 1951.
6. N.-Y. Shen and O. T.-C. Chen, “Low power multipliers by minimizing switching activities of partial products”, IEEE International Symposium on Circuits and Systems, 2002, Volume: 4, 26-29 May 2002.
7. J. Gu and C.-H. Chang, “Ultra low voltage, low power 4-2 compressor for high speed multiplications”, International Symposium on Circuits and Systems, 2003, Volume: 5, 25-28 May 2003.
8. S.-F. Hsiao, M.-R. Jiang and J.-S. Yeh, “Design of high-speed low-power 3-2 counter and 4-2 compressor for fast multipliers”, Electronics Letters, Volume: 34, Issue: 4, 19 Feb. 1998.
9. K. Prasad and K. K. Parhi, “Low-power 4-2 and 5-2 compressors”, Conference Record of the Thirty-Fifth Asilomar Conference on Signals, Systems and Computers, 2001, Volume: 1, 4-7 Nov. 2001.
10. R. Zimmermann and W. Fichtner, “Low-Power Logic Styles: CMOS versus Pass-Transistor Logic”, IEEE Journal Of Solid State Circuits, 32(7):1079-1090, July 1997.
11. J. Liu, “Arithmetic and Control Components for an Asynchronous System”, PhD thesis, The University of Manchester, 1997.
12. S. B. Furber, D. A. Edwards and J. D. Garside, “AMULET3: a 100 MIPS Asynchronous Embedded Processor”, IEEE International Conference on Computer Design, 2000, 17-20th September 2000.
13. S. B. Furber and P. Day, “Four-Phase Micropipeline Latch Control Circuits”, IEEE Transactions on VLSI Systems, vol. 4 no. 2, June 1996, pp. 247-253.
14. S. B. Furber, A. Efthymiou, J. D. Garside, M. J. G. Lewis, D. W. Lloyd and S. Temple, “Power Management in the AMULET Microprocessors”, IEEE Design and Test of Computers Journal special issue, pp. 42-52 (Ed. E. Macii), March-April 2001.

15. J. Sparsø, S. Furber (eds). "Principles of Asynchronous Circuit Design: A systems perspective". Kluwer Academic Publishers, 2001.
16. M. Lewis, J. D. Garside, L. E. M. Brackenbury. "Reconfigurable Latch Controllers for Low Power Asynchronous Circuits". Internation Symposium on Advanced Research in Asynchronous Circuits and Systems, 1999, April 1999.
17. P. A. Riocreux, M. J. G. Lewis and L. E. M. Brackenbury, "Power reduction in self-timed circuits using early-open latch controllers", Electronics Letters, Volume: 36, Issue: 2, 20 Jan. 2000.
18. Y. Liu and S. B. Furber, "The Design of a Low Power Asynchronous Multiplier", International Symposium on Low Power Electronics and Design, 2004, Newport Beach, California, USA, August 9-11, 2004.
19. P. E. Landman and J. M. Rabaey, "Architectural power analysis: The dual bit type method", IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Volume: 3, Issue: 2, June 1995.
20. V. Zyuban and P. Kogge, "Split Register File Architectures for Inherently Low Power Microprocessor", Power Driven Microarchitecture Workshop at ISC98, June 1998.
21. J. H. Pasternak and C. A. T. Salama, "Differential pass-transistor logic", Circuits and Devices Magazine, IEEE , Volume: 9, Issue: 4, July 1993.
22. M. Suzuki, N. Ohkubo, T. Yamanaka, A. Shimizu and K. Sasaki, "A 1.5 ns 32 b CMOS ALU in double pass-transistor logic", IEEE International Conference on Solid-State Circuits , 40th ISSCC., 24-26 Feb. 1993.
23. J. Yuan and C. Svensson, "New TSPC latches and flipflops minimizing delay and power", IEEE International Symposium on VLSI Circuits, Digest of Technical Papers., 1996 , 13-15 June 1996.
24. G. Gerosa, S. Gary, C. Dietz, P. Dac, K. Hoover, J. Alvarez, H. Sanchez, P. Ippolito, N. Tai, S. Litch, J. Eno, J. Golab, N. Vanderschaaf, and J. Kahle, "A 2.2 W, 80 MHz superscalar RISC microprocessor", IEEE Journal on Solid-State Circuits, vol. 29, pp. 1440 1452, Dec. 1994.