# Evaluation of Bit-Pipelined Array Circuits for Datapath DSP Applications

Pasin Israsena

Thailand IC Design Incubator (TIDI)
National Electronics and Computer Technology Center (NECTEC)
99/25 Unit C, Software Park Building
Pakred, Nonthaburi THAILAND 11120
Email: pasin@nectec.or.th

**Abstract**: This paper discusses issues in VLSI design and implementation of high performance datapath circuits. Of particular concern will be various types of multiplier and adder, which are fundamental to DSP operations. Performance comparison will be provided in terms of sampling speed, layout area, and in particular, power consumption, with techniques that may be applied to reduce power dissipation also suggested. As an example, a low power, high performance recursive filter achieved through bit-level pipelining technique is illustrated

## 1. Introduction

In the last three decades, there has been a tremendous growth in the field of digital signal processing (DSP). DSP techniques have found applications in fields as diverse as communications, speech, audio, video, medicine and vision. In earlier days, high-performance DSP designs meant a high sampling speed and minimal layout size, but low power consumption [1] is becoming more and more important as a design target. Much of the demand has been fuelled by the promise of an exciting array of new multimedia products and services such as video games, car navigators, cellular and video phones, or personal digital assistants. Considering hardware realisation, this paper will focus on standard-cell implementation, and therefore the emphasis is on gate-level designs of DSP building blocks such as adders and multipliers. Various techniques that have been used to implement these functions will be briefly reviewed, with performance comparison between the structures also provided. Power consumption of various multipliers and adders and the effect of glitches, in particular, will be further analysed here. As an example, a simple DSP system which is, in this case, a recursive filter will be used to show how power reduction can be reduced using simple techniques such as pipelining, particularly at the bit-level.

In what follows the various types of adders and multipliers and their performance in terms of area, time, and power will be discussed in section 2. Section 3 then deals particularly with power analysis and power reduction technique, with an example of such a low-power system shown in section 4. Section 5 concludes the work.

## 2. Performance Comparison of Adders and Multipliers

### 2.1 Adders

One of the most basic types of adders is the Ripple Carry Adder (RCA)[2-3]. Given that the speed of the RCA is determined by the carry propagation time that is a linear function of the wordlength $d$ bits such as $T_D=O(d)$, this implies a slower sampling speed for systems with large data wordlength. Accordingly a lot of researches have been directed into designing a fulladder with short carry-delay [2-3]. This option, however, is only possible with the designer having an access to the circuit level implementation whereas gate-level designs are considered here. RCS also suffers from erroneous computations that result in unnecessary switchings that are glitches [1][4]. Because of these wasted switch operations, power consumption can be higher than necessary. In the carry lookahead adder (CLA), the carry propagation time is reduced to $O(log(d))$ by using a tree-like circuit to rapidly compute the carry signal. Several designs exist such as [5-8]. Because the fan-in of logic gates increases linearly with the number of bit in the adder, the CLA is usually partitioned into blocks with carry lookahead. The block size is usually selected in the range 4-5.

Other types of parallel adders include Carry select adder (CS) [9]. It consists of blocks; each executing two additions, with one assuming that the carry-in is '1' and the other assumes the carry-in is '0'. The real carry-in is computed from the previous block and selects one of the two sum outputs with multiplexer. Its optimal staging, however, will depend on the technology. CS also generally uses larger area than RCA. The conditional-sum adder (CSA) [10], on the other hand, generates all possible sums and carries in a manner similar to the carry-select adder. It uses a modified half-adder to generate sums and carries in the first phase. The second phase uses $log_2 d$ level of multiplexers to conditionally combine neighbouring bits into the final sum [10]. Although generally proved to be fast, its performances in both speed and power have usually been enhanced by employing pipelining or special circuit styles such as the complementary pass-transistor logic (CPL) [11], which may not suit standard-cell implementation. The CSA also has complex interconnections that require large chip area.

To compare the speed and area performance of the adders, Table 1 shows the predicted maximum delay and transistor count. The table is summarised from [12]. In this work

Pirsch uses a simple CMOS transistor RC model to describe the delay of each adder as a function of the numbers of bit $n$ and a fixed variable $\tau_L$. It is clear that the non-linear relationship of $n$ in some designs makes direct comparison difficult. Recent works by C. Nagendra et al. [13-14] have studied several adders and compared their area, time, and power performance. It was shown that the RCA has the smallest area compared to the other classes and the lowest power consumption in many cases. The carry-lookahead adders, especially the ELM [7], have been shown to be the fastest, although their power dissipation can be relatively high. Carry select adders, on the other hand, are widely used as an optimum compromise between the high speed of CLA and the low area of RCA. The conditional sum adder, with variable block staging, can also be very fast if well optimised. Its power consumption can be comparable or may be less than that of the RCA because it uses a reduced internal swing and a relatively small transistor count if the CPL-like style is used. It is interesting to note from [13] that, apart from its relatively slow speed, RCA offers a low area and power solution. Although high-speed DSPs can be constructed simply as a discrete connection of high speed adders and multipliers, it will be seen later that RCAs can generally be integrated with array multipliers to form a structure that overcomes this speed problem while retaining the low power and area advantages.

**Table 1.** Comparison of parallel adders [12]

| Type | Delay | Trans Cunt |
|---|---|---|
| RCA | $\tau_L(24n-1)$ | 56n |
| CLA | $\tau_L[2\log_2^2 n+16\log_2 n+17.8]$ | $64n-16\log_2 n-32$ |
| CS<br><br>Opt | $\tau_L[4n/m+15m+34.4]$<br>$m_{opt}$=sqrt(4n/15)<br>$\tau_L$(5sqrt(15n)+34.4) | $58n+8n/m+2(n/m)^2$ |
| CondSm | $\tau_L[2n+13\log_2(n+1)+11]$<br>$n=2^k-1$, n>7 | $(10n-2)\log_2(n+1)+34n$<br>$n=2^k-1$ |

## 2.2 Multipliers

The ripple carry array multiplier is probably the simplest structure of all parallel multipliers. It is derived directly from equation

$$P = \sum_{i=0}^{m-1}\sum_{j=0}^{n-1}(a_i b_j)2^{i+j} \qquad (1)$$

The multiplier can have an array structure as shown in Figure 1.

**Carry Save Array Multiplier:** In a ripple carry multiplier (Fig 1) the carry signals go horizontally, whereas in a carry save multiplier the carry signals go diagonally. The carry save multiplier also needs a summation of the final partial products on the last row using a vector merging adder (VMA). Usually, the term Braun array multiplier is used to describe multipliers whose final VMA is implemented as a ripple carry adder. Although the cost in terms of logic element is similar to a ripple carry multiplier, Braun array multipliers have particular advantages when used in pipeline, as will be discussed later. In a multiplier array, a balanced sum and carry delay is desirable because sum and carry signals are both on the critical path. This is different from the case of a parallel adder where, as already stated, only the carry path needs to be optimised to speed up the circuit.

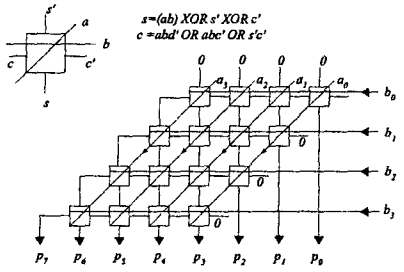


**Figure 1.** Ripple carry multiplier

**Tree-based multipliers:** In tree-based multipliers, compression is obtained based on the fact that a fulladder is used to perform each bit-product. These adders can, in turn, be considered as counters, with a full adder being a (3,2)-counter that adds three inputs and form a two bits result. The output equals the number of 1s in the inputs. A half-adder, accordingly, is denoted as (2,2)-counter. In 1964 Wallace [15] showed that a tree structure of such counters is an efficient method, $O\log_2 d$, to add the bit-products. Several alternative tree-based addition schemes also exist such asDadda [16]. Wallace tree multiplier should therefore only be used in designs with large wordlength and where performance is critical.

**Pezaris Multiplier:** Multipliers discussed so far are used in unsigned systems. In 2's complement number representation, however, the most significant bit (MSB) is weighted negatively. In realising such a system, Pezaris [17] generalises the fulladders into four types as shown in Fig. 2. In type *A0*, which represents a normal adder, all three inputs $x,y,z$ are weighted positively and the result lies in the range {0,3}. This result is represented by a 2-bit binary number $cs$ where $c$ and $s$ are also weighted positively. In the other three types there are some signals, indicated by the dots, that are weighted negatively. It can be verified that, in spite of different weightings, these adders actually represent almost identical function [17].

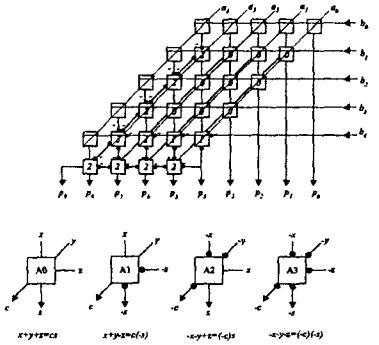

**Figure 2.** Pezaris multiplier

Other types of signed multipliers include Baugh-Wooley (BW) and Booth multipliers. Although in an array form

similar to Pezaris, BW only requires one type of adder [18]. Booth multipliers, on the other hand, are generally used in systems where the operands are longer than 16-bit [19]. The structure is based on recoding the 2's complement operand (i.e. multiplier) in order to reduce the number of partial products to be added. This makes the multiplier faster while using less hardware. Detail of both types can be read further in [18-19].

Similar to the performance comparison of adders discussed earlier, Table 2 shows the estimated delays and transistor counts for various multiplier structures. For simplicity, all the multipliers are n×n. It can be said that generally the faster designs employ Booth-encoded or tree-based techniques such as Dadda arrays. For operands of 16-bit and over, the modified Booth's algorithm is even more attractive as it reduces the partial product's number by half, resulting in a faster and smaller layout. Extremely fast designs have been shown to adopt the Wallace tree structure with modified Booth encoding. The array structures such as Baugh-Wooley, on the other hand, can still perform at a reasonable speed especially in small wordlength designs.

**Table 2.** Performance comparison of array multipliers

| Unsigned | Delay | Trans Count |
|---|---|---|
| Ripple Carry | (64n-72.4) $\tau_L$ | $32n^2$-36n |
| Braun | (46.4n-54.8) $\tau_L$ | $32n^2$-36n |
| Dadda 8x8 | 280.2$\tau_L$ | 1760 |
| Dadda 16x16 | 544.2$\tau_L$ | 7616 |
| Signed | | |
| Pre-post comp | (56.2n-89.1) $\tau_L$ | $32n^2$+40n+64 |
| Pizaris | (46.4n-38.4) $\tau_L$ | $32n^2$-26n |
| Baugh-Wooley | (46.4n-14.8) $\tau_L$ | $32n^2$-32n+88 |
| Booth | (41.5n-1.9) $\tau_L$ | $23n^2$+5n-58 |

In terms of power consumption, Table 3 shows a recent result by Keane et.al [20]. It can be seen that the carry-save based array structures such as Pezaris or Braun multipliers consume less power than the tree-based or Booth encoded counterparts. Although the number of transistors is reduced in the Booth encoded array, its total power remains high due to the overhead in wiring, so as in the tree-based structure. The carry-save array was also shown to use less area [20]. Thus, given an already good power-area characteristic of the carry-save array structures, they can prove to be highly efficient implementations if their speed can be improved further.

**Table 3.** Power of various multipliers @20Mhz [20]

| Type | Random Input | | | 16-bit Input | | |
|---|---|---|---|---|---|---|
| | 8-bit | 16 | 24 | 8-bit | 16 | 24 |
| CS(reg) | 5.99 | 23.19 | 56.92 | 23.19 | 7.69 | 5.66 |
| CS(flat) | 3.23 | 25.94 | 67.96 | 25.94 | 9.58 | 6.41 |
| 2'c CS | 3.65 | 27.31 | 80.79 | 27.31 | 10.0 | 9.12 |
| Booth | 5.09 | 27.95 | 88.58 | 27.95 | 8.43 | 8.93 |
| BW | 5.50 | 37.10 | 93.60 | 37.10 | 12.8 | 16.0 |
| 2'c BW | 4.00 | 32.24 | 86.12 | 32.24 | 9.32 | 10.4 |

## 3. Further Power Analysis

Having shown to be power-competitive by earlier works, the power consumption of the ripple-carry adder and carry-save multiplier structures is investigated further in this work. The two structures of various sizes have fully been implemented using 1μm CMOS and their power consumption evaluated using [21]. The result is shown in Figure 3. The *total* power consumption is shown as a combination of power dissipations due to glitches (glitch) and the rest which is termed an algorithmic power (*algo*). The total power values are obtained through normal simulation procedures, whereas the algorithmic power is equivalent to power consumption based on zero-delay model, and is achieved by counting the activity of the nodes only once during each clock. The power due to glitch is then simply *total-algo*.

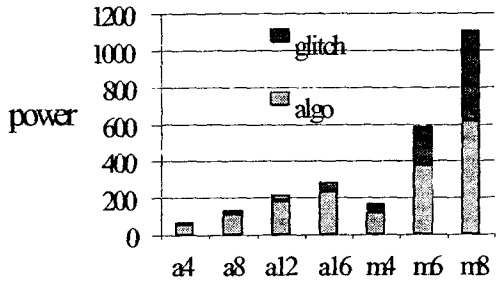


**Figure 3.** Power consumption (μW/MHz)of RCA adders(a) and CS multipliers(m)

Interestingly, it can be seen that glitches contribute more to the total power consumption in the larger structures, and the scope for further power reduction is considerable if these glitches can be removed. This larger portion of glitch power is understandable, as the larger designs have longer critical paths and hence more chances that internal signals are out of sync, resulting in more glitches. It is conceivable then that, if this critical path can be shortened, a structure that has lower power consumption than the original, already power-efficient, array design is possible. As will be seen in the next section, by employing the bit-level pipelining technique in an integrated bit-slice system which is a combination of the RCAs and array multipliers, the critical path is shortened and the effect on power reduction is as expected, with only a small penalty in terms of layout area. Even though Booth multiplier is also very glitch-sensitive, it is not further analysed here because it is not suitable for bit-level pipelining, meaning that the power consumption can not be reduced further that which is already higher than the non-pipelined carry save equivalent.

## 4. Example of Power Reduction Through Pipelining

Because of the restriction to standard-cell implementation, the pipelining technique is to be discussed here, as it only requires adding and re-arranging the latches which can easily be done at the gate-level. To use as an example to

illustrate the point, consider Figure 4 which shows a 2nd order allpass section which can be used to construct a wave digital filter WDF [22]. Based on a bit-sliced representation, the 2nd order allpass section with 20-bit data and 8-bit coefficient I/Os can be as illustrated in Figure 4. Because the structure is recursive in nature, so pipelining will need to be applied such that the functionality of the pipelined structure remains correct. In Figure 4, several levels of pipelining are applied (B3, B15, and B30) [23] and the corresponding power consumption is shown in Figure 5.

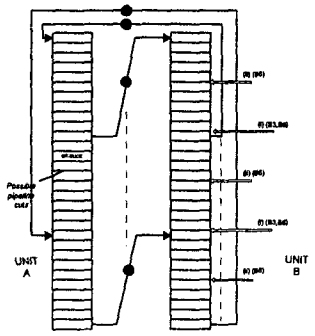


**Figure 4.** Pipelined WDF 2nd order section (20x8)

In Figure 5, the contributions to the power dissipation from different parts of the circuits have been isolated. *Pcell* (cell switching power) decreases with the number of bit slices in the register transfer paths. Because in each case the number of bit level operations required to complete the filter cycle is the same, this reduction is entirely attributable to a reduction in glitch activity. This effect competes with the increasing power used in the pipeline registers, *Pclk*, leading to, at least in principle, a minimum in the total power. This occurs here and to be seen for the architecture B15. Comparing the optimum architecture B15 to the starting architecture B1 we see that the power is approximately 50% and the power-are-delay product is increased by a factor of approximately 5 [23]. The results hence agree with analysis given in section 3.

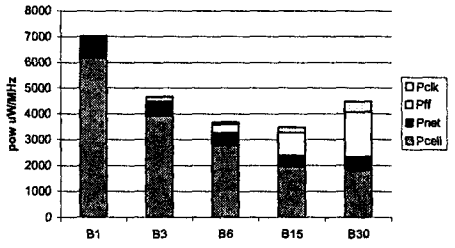


**Figure 5.** Power consumption of pipelined WDF

## 5. Conclusion

Gate-level implementations of various DSP structures were discussed, with their performance in all area, time, and power also compared. The structures considered range from adders and multipliers, which are fundamental to DSP functions, to specifically a pipelined digital filter used as a low-power design example. It has been shown that simple structures such as RCA adders or carry-save based multipliers already have a competitive power-area characteristic, and given that how their performance can be increased further when used in an integrated bit-slice pipelined system.

## Reference


[1] A. P. Chandrakasan, S. Sheng and R.W. Brodesen, "Low-Power CMOS Digital Design", *IEEE J. of Solid State Circuits*, vol. 27, No.4, pp. 473-483 April 1992.

[2] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A systems Perspective*, Addison-Wesley:1994

[3] A. M. Shams and M. A. Bayoumi, "A Novel High-Performance CMOS 1-Bit Full-Adder Cell," *IEEE Transactions on Circuits & Systems II-Analog & Digital Signal Processing*, vol. 47, no. 5, pp. 478-481, May 2000

[4] A. Shen, A. Ghosh, S. Devadas, and K. Keutzer, "On Average Power Dissipation and Random Pattern Testability of CMOS Combinational Logic Networks," *IEEE/ACM Int. Conf. on Computer-Aided-Design*, pp. 402-407, CA, Nov. 8-12,1992

[5] R. P. Brent and H. T. Kung, "A Regular Layout for Parallel Adders," *IEEE Transo n Computers*, vol. C-31, pp. 280-284, 1982

[6] A. Weinbeger and J. L. Smith, "A Logic for High Speed Addition," *Nat. Bur. Stand. Circ*, vol. 591, pp. 3-12, 1958

[7] T. P. Kelliher, R. M. Owens, M. J. Irwin, and T-T. Hwang, "ELM-A Fast Addition Algorithm Discovered by a Program," *IEEE Trans. Comput*, vol. 41, no. 9, Sep 1992

[8] S. Hwang and A. L. Fisher, "Ultra Fast Compact 32-bit CMOS Adder in Multi-Output Domino Logic," *IEEE J. Solid-State Circuits*, vol. SC-24, pp. 35-369, 1989

[9] O. J. Bedrij, "Carry-Select Adder," *IRE Trans. lect. Comp*, vol. EC-11, pp. 340-346, 1962

[10] J. Sklanksy, "Conditional-Sum Addition Logic," *IRE Trans Elect. Comp*, vol. EC-9, pp. 226-231, 1960

[11] H. Lindkvist and P. Andersson, "Techniques for Fast CMOS-Based Conditional Sum Adder," *Proc. IEEE Int. Conf. on Computer Design*, Cambridge, MA, pp. 626-635,1994

[12] P. Pirsch, *Architectures for Digital Signal Processing*, Wiley: 1996

[13] Nagendra, R. M. Owens, and M. J. Irwin, "Power-Delay Characteristics of CMOS Adder," *IEEE Trans. on VLSI systems*, vol. 2, no. 3, Sep 1994

[14] C. Nagendra, M. J. Irwin and R. M Owens, "Area-Time-Power Tradeoffs in Parallel Adder," *IEEE Trans. On Circuits and Systems II*, vol. 43, no.10, pp. 689-702, Oct 1996

[15] C. S. Wallace, "Suggestion for Fast Multiplier," *IEEE Trans. on Electronic Computers*, vol. EC-13, pp. 14-17, 1964

[16] L. Dadda, "Some Schemes for Parallel Multipliers," *Alta Frequenza*, vol. 34, pp. 349-356, 1965

[17] S. D. Pezaris, "A 40 17-bit Array Multiplier," *IEEE Trans. on Computers*, vol. 20, pp. 442-447, 1971

[18] C. R. Baugh and B. A. Wooley, "A Two's Complement Parallel Array Multiplication Algorithm," *IEEE Trans. on Computers*, vol. C-22, pp. 1045-1047, Dec 1973

[19] A. D. Booth, "A Signed Binary Multiplication Technique," *Quarterly Journal of Mechanics and Applied Mathematics*, vol. 4, pp. 236-240, 1951

[20] G. Keane, J. R. Spanier, and R. Woods, "The Impact of Data Characteristics on Hardware Selection for Low-Power DSP," Proc. *IEEE/ACM ISLPED* 98, pp. 94-96, 1998

[21] P. Israsena and S. Summerfield, "Novel pattern-Based Power Estimation Tool with Accurate Glitch Modelling," *Proc. IEEE ISCAS'2000*, vol. 4, pp. .721-724, May 2000

[22] S.S. Lawson and A.R. Mirzai, *Wave Digital Filters*. 1990: Ellis-Horwood

[23] P. Israsena and S. Summerfield, "Power Minimisation of VLSI Wave Digital Filters through Systolic Block Size Selection," *IEE Electronics Letters*, vol. 35, no. 21, pp. 1795-7, 1999