A two-item floating point fused dot-product unit with latency reduced

Mingjiang Wang, De Liu, Ming Liu, and Boya Zhao
Harbin Institute of Technology Shenzhen Graduate School,
Xili University Town, Shenzhen 518055, Guangdong, China
a) mjwang@hit.edu.cn
b) liude19832006@126.com

Abstract: This paper presents a floating point fused dot-product (FDP) unit with latency reduced. The proposed FDP unit performs the dot-product operation of four floating point numbers: \( ab \pm cd \) and is implemented with dual-path algorithm. The proposed FDP is modeled in Verilog-HDL and synthesized using TSMC 65 nm technology library. Synthesis results show that our proposed FDP unit is 24~30\% faster and 36.4\% less area than the fastest FDP in previous work. We also use the proposed FDP unit and our previously designed FAS (fused add-subtract) unit to implement a FFT Radix-2 Butterfly (R2BF) unit. The latency of our proposed R2BF unit is improved roughly by 34\% and the area is reduced by 41.6\%, compared to the fastest 2’s-complement butterfly unit.

Keywords: floating point arithmetic, fused dot-product unit, fused FFT-butterfly unit

Classification: Integrated circuits

References

[1] J. Sohn and E. E. Swartzlander: “Improved architectures for a fused floating-point add-subtract unit,” IEEE Trans. Circuits Syst. I, Reg. Papers 59 (2012) 2285 (DOI: 10.1109/TCSI.2012.2188955).
[2] D. Liu, et al.: “Delay-optimized floating point fused add-subtract unit,” IEICE Electron. Express 12 (2015) 20150642 (DOI: 10.1587/elex.12.20150642).
[3] E. E. Swartzlander and H. H. M. Saleh: “FFT implementation with fused floating-point operations,” IEEE Trans. Comput. 61 (2012) 284 (DOI: 10.1109/TC.2010.271).
[4] J. H. Min, et al.: “A low-power dual-path floating-point fused add-subtract unit,” 46th ACSSC Conf. Rec. (2012) 998 (DOI: 10.1109/ACSSC.2012.6489167).
[5] T. Yao, et al.: “A multi-path fused add-subtract unit for digital signal processing,” IEEE Int. Conf. CSAE (2012) 448 (DOI: 10.1109/CSAE.2012.6272991).
[6] H. Saleh and E. E. Swartzlander: “A floating-point fused add-subtract unit,” 51st MWSCAS (2008) 519 (DOI: 10.1109/MWSCAS.2008.4616850).
[7] T. Yao, et al.: “Dual-path architecture of floating-point dot product computation,” ICCSNT (2011) 2272 (DOI: 10.1109/ICCSNT.2011.6182427).
[8] J. Sohn and E. E. Swartzlander: “Improved architectures for a floating-point
1 Introduction

Floating-Point (FP) fused arithmetic units have been researched recently to reduce the accumulated error and improve the performance, such as fused add-subtract (FAS) \[1, 2, 3, 4, 5, 6\] unit, fused dot-product (FDP) \[3, 7, 8, 9, 10\] unit and fused FFT butterfly \[3, 11, 12\] unit.

This paper presents a latency-reduced two-item FP fused dot-product (FDP) unit. The proposed FDP is designed with the dual-path algorithm \[13\] and the injection-based rounding method \[14\]. But we modified and improved these two algorithms to reduce the delay and area.

As feature size of CMOS transistor shrinks below 130 nm, using FO4 to compare the performance of two designs implemented in different technology processes is not very precise in quantity. So we use the number of logic levels to represent the latency. We define that the delay of one 2-input gate (AND, OR, XOR, MUX) is 0.5L. Fig. 1 shows the delay of an 8-bit adder with Kogge-Stone architecture is 4L: the delays of \(G_i\) (carry generation \(\lor\)), \(P_j\) (carry propagation \(\lor\)), \(G_{i:j}\) (dot-operation, \(\land\)) and \(S_f\) (final sum) are respectively 0.5L, 0.5L, 1L and 0.5L. It is noted that the delay of \(C_8\) (the carry signal of the most significant bit) is 3.5L, which is 0.5L faster than the sum signals (\(S_7\)~\(S_4\)). So we conclude that the number of logic levels of a \(W\)-bit adder with Kogge-Stone architecture is \(\lceil \log_2 W \rceil + 0.5 + 0.5 \times L\). \(\lceil \cdot \rceil\) is the ceiling function, for example, when \(W\) is equal to 24, the delay is \(\lceil \log_2 24 \rceil + 0.5 + 0.5 = 6\)L.
Section 2 summarizes the structure and analyzes the latency of the fastest FDP presented in recent research work [8]. In Section 3, the overall architecture of the proposed FDP unit is firstly presented, then the techniques we employed to decrease the latency are described, and finally the details of implementing each component of the proposed FDP are provided. In Section 4, the synthesis results and comparisons are presented.

We implemented a fused FFT Radix-2 Butterfly (R2BF) unit by using the proposed FDP and our previous work FAS [2]. Similarly, the R2BF is synthesized and compared with other research works in current literatures in Section 4.

2 Architecture of the fastest FDP in recent research works

The main frame of the fastest FDP [8] is shown in Fig. 2. It is also implemented with dual-path algorithm. The design is divided into three stages and two paths, and the delay in terms of logic levels of each stages and component is marked in Fig. 2. The inputs are four FP numbers: $a$, $b$, $c$ and $d$.

1) STAGE1

The first stage consists of three blocks: one ExpComp and two MulTrees. The ExpComp is used to compare the exponents of $a \times b$ and $c \times d$, and produce the comparing signal swap. The two MulTrees are used to generate compressed mantissa partial-products $hs_{ab}$, $hc_{ab}$, $hs_{cd}$ and $hc_{cd}$. The ExpComp is implemented in 2 levels of 8-bit adders, so its delay is $8L$; the MulTree is designed by using Booth transformation ($2L$) and Wallace compress tree ($7L$).

2) STAGE2

The second stage is divided into two path: FAR and CLOSE path. The FAR path consist of four blocks: SigSwap, Align, Sticky and 4:2 CSA. The SigSwap block swaps $hs_{ab}$, $hc_{ab}$, $hs_{cd}$ and $hc_{cd}$ according to the logic value of signal swap and has $1L$ delay. According to the description of [8], the Align block is a 74-bit barrel shifter and its delay is $6L$. The Sticky is an OR tree to computing the sticky bit and its delay is $[\log_2 48] \times 0.5L = 3L$. The 4:2 CSA is a compressor and its delay is $2L$, the reason is explained in Section 3.

The CLOSE path consist of five blocks: SmallAlign, 4:2 CSA, LZD, SigComp and PreNorm. The smaller partial-product is shifted to the right by 0 or 1 or 2 bits through the SmallAlign block which has the delay of $1L$. The 4:2 CSA is used to compress the four shifted partial-product: $f1$, $f2$, $f3$ and $f4$. The SigComp block is
used to compare and decide which partial-product is bigger. The SigComp block can be implemented by a 50-bit comparator or a subtractor, so its delay is 
\[ \log_2 50 + 1 + 0.5 = 7.5L \]. The PreNorm block is used to pre-normalize the compressed partial-products, shifting sigL and sigS to the left by the amount of \( 12n \) bits. The LZD is a leading zero detector.

It is obviously seen that the critical path is the CLOSE path of which the total delay is \( 16.5L \). The delay of FAR path is \( 12L \).

![Diagram](image_url)

Fig. 2. The main frame of the FDP proposed in [8]

3) STAGE3

The aligned and compressed mantissa partial-products \( hcF, hsF \) of FAR path and \( hcC, hsC \) of CLOSE path is multiplexed through the PathSelect block to determine the correct result. Then the most 24 significant bits of sig1 and sig2 are added through a 24-bit compound adder [15] whose delay is 
\[ \log_2 24 + 0.5 + 0.5 = 6.5L \] (the delay of a compound adder is \( 0.5L \) bigger than a common adder). Since the sum of sig1 and sig2 may overflow, according to [8, 15], the Round block is used to generate the rounding select signal sel, and its delay is \( 3L \).
Finally, the exponent is adjusted through the $\text{Exp Adj}$ block which contains several 8-bit adders and four levels of multiplexers. As is seen, the delay of the third stage is $14.5L$.

By adding the numbers of logic levels of each stage, the overall latency of the FDP proposed by Sohn [8] is $40L$.

3 The proposed FDP in this work

3.1 Architecture and functionality

The overall architecture of our proposed FDP is illustrated in Fig. 3. The proposed FDP can perform $ab - cd$ or $ab + cd$. The proposed design is also divided into three pipeline stages in vertical direction and two paths in horizontal direction.

![Fig. 3. The overall architecture of the proposed FDP unit](image_url)

STAGE1 is mainly used to obtain the larger exponent $E_{\text{max}}$ and the exponent difference $E_{\text{diff}}$ of $ab$ and $cd$ through the $\text{Exp Comp}$ block, compute carry-save formatted partial-products $hs_{ab}$, $hc_{ab}$, $hs_{cd}$ and $hc_{cd}$ of mantissas $Fa$, $Fb$, $Fc$ and $Fd$ through the $\text{PPR}_{AB}$ and $\text{PRP}_{CD}$ blocks, and determine the smaller...
partial-product $hs_{small}$ and $hc_{small}$ which are shifted to the right by the amount of 0, 1 or 2 bits through the $Shift2Bits$ block. This stage also produces two injection signals $inj0$ and $inj1$ for the purpose of rounding.

STAGE2 is divided into two paths: the FAR and Close path. The FAR path is used to align the shifted partial-products $hs_{sh2}$ and $hc_{sh2}$ through two aligning blocks $AlignS$ and $AlignC$. The injection signals $inj1(0)$, larger partial-products $hs_{large}$, $hc_{larger}$ and aligned partial-products $hs_{align}$, $hc_{align}$ are reduced through two compressors $5:2$ CSA. The CLOSE path is used to compress $hs_{large}$, $hc_{large}$, $hs_{sh2}$ and $hc_{sh2}$ through the $4:2$ CSA block, compute the mantissa difference $mdiff1$ of $ab$ and $cd$ through the $FPFA$ [16] block, and count the leading zero number $ln$ of $mdiff1$ through the $LZD$ block. Besides, this stage generates overflow signals $ovf(2,1,0,m)$ and underflow signals $udf(2,1,0,m)$ for four cases.

STAGE3 computes and rounds the sum or difference $mant1$($mant0$) of mantissas’ partial-products through the $MantAdd1$ and $MantAdd0$ blocks in FAR path, normalizes the mantissa’s difference $mdiff1$ through the $NORMALIZE$ block and rounds the normalized result $mdiffz$ through the $Round$ block in CLOSE path. Simultaneously, the exponent $expf$ of CLOSE path is computed through the $ExpAdjClose$ block. Finally the exponent $expf$ and mantissa $mantf$ of FAR path is multiplexed through the $MantSelect$ block, and the final result is selected through the multiplexer $PathSelect$. The $Exception$ block produce the final exception signals $overflow$ and $underflow$. The cases that the result is a NaN or infinity can be detected in STAGE1, and are not drawn in Fig. 3 for clarity.

The detailing implementation of each block and the techniques we employed to improve the area and latency are described in section 3.2.

3.2 Implementation and techniques

The circuit structure of STAGE1 is shown in Fig. 4. The product $ab$ (or $cd$) of FP number $a$ and $b$ (or $c$, $d$) can overflow, but the sum of $ab$ and $cd$ may not if the effective operation is $|a \times b| - |c \times d|$. So 10 bits are needed to represent the value of the sum of the exponents. The adder $AddAB$ and $AddCD$ are used to compute the exponents $Eab$ and $Ecd$ of $ab$ and $cd$ respectively, and the $CSA$ is used to compute the exponent difference $Ediff$ and the indicating signal $swap$. The $PPAR_AB$ and $PPR_CD$ blocks are exactly the same as the $Multrees$ in Fig. 2. The techniques we used to reduce the delay and area are as follows:

- **Technique 1.** To balance the delay between pipeline stages, the mantissa swap logic $(SigSwap$ in Fig. 2) are moved from STAGE2 to STAGE1, forming the $MantSwap$ block.

  The value range of the dot-product of four mantissas can be derived as follows:

  $0 \leq Fa, Fb, Fc, Fd < 2 \rightarrow 0 \leq Fa \times Fb < 4, \quad 0 \leq Fc \times Fd < 4 \rightarrow 0 \leq Fa \times Fb \pm Fc \times Fd < 8$, so there are four cases for the dot-product $Fa \times Fb \pm Fc \times Fd$, shown in Fig. 5:

  I. $4 < Fa \times Fb \pm Fc \times Fd < 8$, the 25th bit $msb2$ is logic 1;
  II. $2 < Fa \times Fb \pm Fc \times Fd \leq 4$, $msb2 = 0$ and the 24th bit $msb1$ is logic 1;
  III. $1 < Fa \times Fb \pm Fc \times Fd \leq 2$, $msb2 = 0$, $msb1 = 0$, and the 23th bit $msb0$ is logic 1;
  IV. $0 \leq Fa \times Fb \pm Fc \times Fd \leq 1$, $msb2 = 0$, $msb1 = 0$, $msb0 = 0$. 
For dot-product, if \( msb1 \) is 1 for \( Fa/C2 Fb \), \( msb1 \) is 0 for \( Fc/C2 Fd \) and the exponent difference \( Ediff \) is 2, then the mantissa difference \( 2^{-Ediff} \times Fa \times Fb - Fc \times Fd \) may have many leading zeros. So we proposed a new method to divide the circuit structure into two paths: if the effective operation is subtraction and the exponent difference \( Eidff = 0, 1, 2 \) and \( msb = 0, msb1 = 0, msb0 = 0 \), then the result is computed through the CLOSE path, otherwise the FAR path.

For the FAR path, the smaller mantissa partial-product \( hs\_small \) and \( hc\_small \) need a full-length right shifting (\( AlignS \), \( AlignC \) in Fig. 3); for the CLOSE path, they need to be shifted to the right by the amount of 0, 1 or 2 bits.

- **Technique 2.** To decrease the area, the Shift2Bit block is shared by both FAR and CLOSE path to shift \( hs\_small \) and \( hc\_small \) to the right by the amount of 0, 1, 2 or 3 bits. To balance the delay between pipeline stages, the Shift2Bit block (equivalent to the SmallAlign block in Fig. 2) is placed in STAGE1. The detailing implementation of the Shift2Bit is illustrated in Fig. 6. It is consist of two levels of 50-bit multiplexers and its delay is 1L.

In Fig. 3 and Fig. 6, the signals \( hs\_sh2 \) and \( hc\_sh2 \) are passed to the CLOSE path and continued to be aligned (shift to the right) through the \( AlignS \) and \( AlignC \) blocks respectively.

When the effective operation is \( |a \times b| + |c \times d| \), the mantissa of the dot-product can be in case I, II and III; when the effective operation is \( |a \times b| - |c \times d| \), it can be in case II, III and IV. To keep the delay as small as possible, we used the injection-based [14] rounding method. In our proposed structure, the guard, round and sticky bit of FAR path is in position \( 0(g1), -1(r1) \) and \( -2(s1) \) for case I and II; and in position \( -2(g0), -3(r0) \) and \( -4(s0) \) for case III and IV, shown in Fig. 6.

- **Technique 3.** To decrease the area, the sticky bit is computed in parallel with the shifter. In Fig. 6, the least 26 and 24 significant bits of \( hs\_sh2 \) and \( hc\_sh2 \)
are ORed through two OR logic gate trees to generate the sticky bit. The “\(\gg\!\!\!\!\!\!\!\!>4\)” block is a row of multiplexers and used to shift \(f_3\) to the right by 4 bits if \(\text{Editf][2}\) is 1. Its delay is 0.5L, smaller than that of \(\text{ORtree1}\) \((\log_2 26 \times 0.5 + 0.5 = 3.5L)\). Using the same method, the aligned smaller partial-product \(hs_{\cdot}\text{align}\) and \(hc_{\cdot}\text{align}\) are obtained through the “\(\gg\!\!\!\!\!\!\!\!>8\)” and “\(\gg\!\!\!\!\!\!\!\!>16\)” block. The width of each shifter is 28 bits, which greatly reduces the area compared to the \(\text{Align}\) block whose width is 74 bits in Fig. 2. The delay of the \(\text{AlignS}(C)\) block is 7.5L.

Fig. 6. The detailing circuit of the \(\text{Shift2Bit}\) and the \(\text{AlignS}(C)\) block

- **Technique 4.** To decrease the area, the injection-base rounding algorithm is used and improved. According to [14], the injection signal should have 24 bits. By analyzing, we found that 3 or 5 bits are enough to store the value of the injection. This improvement not only decreases the width (area), but also reduces the delay of the adders producing the final guard, round and sticky bit. The \(\text{RMDEC}\) block in STAGE1 converts the four IEEE rounding modes into three: rounding to zero (RZ), rounding to infinity (RI) and rounding to nearest even (RN), and produces the injection signals \(\text{inj1}\) and \(\text{inj0}\) which are used for case I, II and III, IV respectively. The \(\text{inj1}\) and \(\text{inj0}\) are shown in Eq. (1).

\[
\text{inj1} = \begin{cases} 
00000 & \text{RZ} \\
10000 & \text{RN} \\
11111 & \text{RI}
\end{cases} \quad \text{inj0} = \begin{cases} 
00 & \text{RZ} \\
100 & \text{RN} \\
11 & \text{RI}
\end{cases}
\]  

(1)

The aligned mantissa partial-products \(hs_{\cdot}\text{align}\) and \(hc_{\cdot}\text{align}\), the larger mantissa partial-product \(hs_{\cdot}\text{large}\) and \(hc_{\cdot}\text{large}\), and \(\text{inj1}\), \(\text{inj0}\) are compressed through two 5:2 CSAs in Fig. 3 to half-sum \(hs_{\cdot}\text{f1}(0)\) and half-carry \(hc_{\cdot}\text{f1}(0)\) which are used for case I, II and III, IV respectively. The circuit of the 5:2 CSA and 4:2 CSA is
illustrated in Fig. 7. The 5:2 CSA block is consist of three levels of full adders $FA$, and its delay is $3L$.

In Far path, the compressed half-sum $hs_{f1}$ and half-carry $hc_{f1}$ are passed to the $MantAdd1$ block, $hs_{f0}$ and $hc_{f0}$ are passed to the $MantAdd0$ block, shown in Fig. 3. The circuit implementation of the $MantAdd1$ block is illustrated in Fig. 8. The $HalfAdd$ block is a row of half adders, and its delay is $0.5L$. The $FPPA$ is a 24-bit flagged parallel prefix adder [16], used to obtain the sum $S$ and $S+1$ of two operands $hs$ and $hc$. The delay of the $FPPA$ block is $\lceil \log_2 26 \rceil + 1 + 0.5 = 6.5L$. The “$>>1$” block is a multiplexer and used to right shift $S$ and $S+1$ for case I. The Adder is a 5-bit (3-bit for $MantAdd0$) adder to generate $g1$, $r1$ and $s1$ in Fig. 6. The $FIX1$ and $FIX0$ block is used to fix the LSB bit of the final result for case I and case II respectively. The $INCDES$ block is used to generate the enable signal $inc$ to select the correctly rounded mantissa through $M1$ and the correctly fixed LSB through $M2$. The total delay of the shifters “$>>1$” and multiplexers $M2(M3)$ is $1L$.

The details of the mechanism is explained in our previously proposed FAS [2].

**Technique 5.** To reduce the delay of the rounding process, a dual-way rounding structure is proposed. The $MantAdd1$ block is used to compute
and round the mantissas’ dot-product for case I and II, and the MantAdd0 block is used for case III and IV, shown in Fig. 3 and Fig. 5. The entire delay of the addition and rounding of the half-sum and half-carry is $8L$. If case I is considered and handled using the rounding method of [8, 15], then the delay of the Round block in Fig. 2 will increase and two extra 24-bit adders are needed to compute $sum + 3$ and $sum + 4$. So the proposed dual-way rounding structure not only decrease the delay, but also helps to reduce the area.

The details of the CLOSE path, ExpAdjFar, ExpAdjClose, and Exception blocks in Fig. 3 are illustrated in Fig. 9. The “$+$2”, “$+$1” and “$-1$” blocks respectively compute the results of $E_{\text{max}} + 2$ for case I, $E_{\text{max}} + 1$ for case II and $E_{\text{max}} - 1$ for case IV. Then four AND gates are used to detect whether the corresponding exponent is maximum ($E = 255$, overflow) or less than zero (underflow). The Exception block is consist of eight multiplexers $M_6 \sim M_{13}$ and used to produce the final exception signals underflow and overflow.

The CLSOE path shown in Fig. 9 mainly consists of a 50-bit 4:2 CSA, a 50-bitflagged parallel prefix adder FPPA, two 50-bit leading zero detectors LZDPos and LZDneg, a 50-bit left shifter NORMALIZE and a 24-bit incrementer INC. The FPPA is used to compute the difference of half-sum $h_s.C$ and half-carry $h_c.C$. The signal carry indicates whether the difference $mdiff_0$ is positive or negative. If $mdiff_0$ is negative, the leading zero number $lzn.n$ is selected through $M4$ and the correct result is obtained by inverting $mdiff_0$. The NORMALIZE block shifts the difference $mdiff_1$ to the left by the amount of $lzn$ bit. Then the normalized mantissa $mdiff_2$ is rounded through the Round block. The most 24 significant bit [49:26] of $mdiff$ are
incremented through the INC block according the enable signal rnd produced by the GRS block. The INC block has the similar structure as an adder but with a smaller delay \( \log_2 24 = 5L \). The ADD1 and ADD0 blocks are two 10-bit adders to compute the exponents of the CLOSE path.

### 4 Results and comparisons

The proposed FDP is modeled in Verilog-HDL and verified through extensive simulations. Also the FDP is synthesized with TSMC 65 nm technology using Synopsys Design Compiler (version: vl-2013.12). Table I shows the latency, area and power of the proposed FDP unit, and Table II gives the delay of each stage. The data with a “#” suffix in Table I is the raw data of corresponding literature.

Table I shows that the latency of the proposed FDP is 1.92 ns. As it is seen in Fig. 2 and Fig. 3, the latency of the proposed FDP is 30.5L, and the latency of the fastest FDP in [8] is 40L. The proposed FDP is 23.7% faster than the FDP in [8]. When FO4 is used to compare the performance, our proposed design is 29.8% faster than the FDP in [8] (87FO4 vs 124FO4 in Table I). Generally the value of FO4 of a process is \( 1/3 \sim 1/2 \) of the drawn channel length (feature size). Although using FO4 to compare the latency of two designs implemented with different processes is not very precise in quantity, it still makes sense to compare performance in quality, since the improvement ratio in terms of FO4 is 70.2% which is roughly close to the ratio 76.3% in terms of logic levels. So FO4 can be used to qualitatively compare the performance of different designs. Table I shows that the FDP [3] designed with single-path algorithm has the worst performance, of which the latency is 181FO4.

**Table I.** The synthesis results of the proposed FDP unit and comparisons

|               | Proposed | [8] (DP 45 nm) | [3] (SP4 5 nm) | [7] (DP 180 nm) | [9] (DP 45 nm) |
|---------------|----------|----------------|----------------|----------------|----------------|
| Latency (ns)  | 1.92     | 1.87#          | 2.72#          | 11.38#         | 2.25#          |
| Latency (L)   | 30.5 (76.3%) | 40 (100%)     | 124 (100%)     | 181            | 126            |
| Latency (FO4) | 87 (70.2%) | 124 (100%)     | 16104#         | 419252#        | 11043#         |
| Area (µm²)    | 41740    | 31472#         | 16104#         | 419252#        | 11043#         |
| Area (65 nm)  | 41740 (63.6%) | 65664# (100%) | 33600         | 54671          | 23040          |
| Power (mW)    | 30.34    | 16.16#         | —              | —              | —              |

The scaled area is computed: Area (65 nm) = Area (45 nm) \( \times (65/45)^2 \), 1FO4 \( \approx 0.022 \) ns for 65 nm, 1FO4 \( \approx 0.015 \) ns for 45 nm, DP and SP means dual-path and single-path.

The major logic blocks consuming the most area of the proposed FDP are two 24-bit multiplication trees (\( PPR_{AB}, PPR_{CD} \)), two 28-bit shifters (\( \text{AlignS}, \text{AlignC} \)), one 50-bit flagged parallel prefix adder (\( \text{FPPA} \)), two 50-bit leading zero detectors (\( \text{LZDPos}, \text{LZDNeg} \) in Fig. 9), two 24-bit \( \text{FPFA} \)s (\( \text{MantAdd1}, \text{MantAdd0} \) in Fig. 8) and one 50-bit barrel shifter (\( \text{NORMALIZE} \)). The proposed FDP consumes 36.4% less area than the fastest FDP of [8] in Table I. The reason is that the STAGE2 of the FDP of [8] in Fig. 2 contains components with big width: two 74-bit barrel shifters, one 50-bit comparators (\( \text{SigComp} \)) and two 76-bit shifters (\( \text{PreNorm} \)). Since the width of these components is big, many big buffers are inserted to strengthen the enable signals’ driving capability, which in turn greatly raises the area.
The delay ratio of each stage is 0.78:0.81:0.78, which is approximately 1:1:1, see Table II. As illustrated in Fig. 3, the delay ratio of STAGE1, STAGE2 and STAGE3 is 10.5L:10.5L:10.5L, which is also 1:1:1. This proves that analyzing the latency with logic level is appropriate, since it is more independent of specific manufacturing process than FO4.

| Table II. The delay of each pipeline stage of the proposed FDP unit |
|-----------------|-----------------|-----------------|-----------------|
|                 | STAGE1           | STAGE2           | STAGE3           |
| Delay (ns)      | 0.78             | 0.81             | 0.78             |

We also constructed a fused FFT R2BF unit by using the proposed FDP and our previously designed FAS [2]. Table III shows the latency, area and power of the proposed R2BF unit, and the comparisons with other research works. The fastest butterfly unit is constructed by the FDP of [8] and the FAS of [1]. The proposed R2BF has a latency of 124FO4 which is 34% faster than the fastest butterfly unit (186FO4) in [1] + [8]. The area of the proposed R2BF is 58% of that in [1] + [8] (104250 vs 178656 in Table III). Our proposed R2BF is 44% slower than the design in [11] (124FO4 vs 86FO4 in Table III), but consumes 46.7% less area (104250 vs 195783 in Table III). The reason is that the fused butterfly unit of [11] uses redundant signed-digit number to represent the mantissa of FP number and is not compatible to 2’s-complement format. The design of [11] has no sticky bit and is not compatible to IEEE754-2008 standard. When the butterfly unit of [11] is used in 2’s-complement systems, six 24-bit adders are needed to convert the 2’s-complement number into redundant signed-digit format at the input, and another six 24-adders are needed to do the reverse operation at the output.

| Table III. The latency, area and power of the proposed FFT R2BF unit |
|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
|                 | Proposed        | [1] + [8]       | [11]            | [12]            | [3]             |
|                 | (45 nm)         | (45 nm)         | (90 nm,)        | (45 nm)         | (45 nm)         | (45 nm)         |
| Latency (ns)    | 2.72            | 2.79            | 2.59#           | 3.70#           | 4.0#            |
| Latency (FO4)   | 124 (66%)       | 186 (100%)      | 86              | 247             | 286             |
| Area (µm²)      | 104250          | 85628#          | 375347#         | 25182#          | 47489#          |
| Scaled Area (µm²) | 104250         | 178656         | 195783          | 53792           | 99082           |
| Power (mW)      | 69.33           | —               | —               | —               | —               |

5 Conclusion

This paper presents a floating point fused dot-product (FDP) unit. Also a fused radix-2 butterfly unit is implemented using the proposed FDP and FAS in [2]. The basic algorithm we adopted is dual-path algorithm [13], but we propose a new criterion to divide the FDP into two paths to reduce the latency and area. Based on injection rounding method, we propose a novel dual-way rounding architecture in FAR path to greatly reduce the delay and area of the critical path. To further decrease the area and delay, the computing of sticky bit is performed in parallel with the alignment of mantissa partial-product.

Acknowledgments

This work is supported and funded by the project of Shenzhen Infrastructure Layout. Project No.: JCYJ20150827165024088.