Supporting De-normalized Numbers in an IEEE Compliant Floating-Point Adder Optimized For Speed

Yariv Levin Tel-Aviv University
Electrical Engineering Department
69978 Tel-Aviv Israel

July 2001

Advised by: Dr. Guy Even

Contents

List of Figures

Chapter 1

Abstract

This project deals with an IEEE floating point addition algorithm of Seidel and Even [ES]. We provide correctness proofs for parts of the design, extend the design to support denormal numbers, and construct a verification procedure. We show that the overhead in latency required for supporting denormalized operands and results in double precision is one logic level. We presented a parameterized design in which the exponent and significand string lengths are given as parameters. An exhaustive test was conducted on a very small precision version of the design. This test revealed a few errors in the design on Seidel and Even. Corrections of these errors are presented in this report.

Chapter 2

Introduction

Floating-point (FP) addition and subtraction are the most frequent FP operations. Both operations use a FP-adder. Therefore a lot of effort has been spent on designing FP-adders and, in particular, on reducing the latency of FP-adders. Our starting point is the FP-adder design of Seidel and Even [SE]. We refer to this design as the SE FP-adder. The SE FP-adder has a latency of roughly 24 logic levels and is easily partitioned into two pipeline stages each with 12 logic levels.

The description of the SE FP-adder in [SE] does not deal with denormal operands and results, lacks correctness proofs of several sub-modules, and lacks a description of the circuitry for computing exponent and sign of the sum.

This paper deals with these three issues as follows:

1. We extended the SE FP-adder design to handle denormal operands and results according to the IEEE 754 Standard [IEEE]. We consider two alternatives for computing the normalization shift amount. The hardware and delay overheads are analyzed. We conclude that the additional delay required for supporting denormal operands and results is only one logic level. The additional hardware is almost linear in the precision (i.e. an OR gate per significand bit and a decoder).

2. We specify in detail the functionality of the sub-modules of FP-adder with support of denormal numbers. We prove the correctness of some of these sub-modules.

3. We describe in detail the circuitry for computing the exponent and sign of the sum. This description confirms the conjecture that the computation of the exponent and sign of the sum does not lie on the critical path.

The above design is specified in a parametric fashion. Namely, the number of bits used for representing the significand and the exponent are parameters. This parametric design is used in the construction of a simulation environment. This simulation environment was utilized to conduct an exhaustive test with small precisions (i.e. 3 bits for the exponent and 5 bits for the significand). This exhaustive test helped reveal several errors in the SE FP-adder.

Organization. The paper is organized as follows. In section 2, notations are presented. In section 3, we describe the "vanilla" algorithm for FP addition. In section 4, high-level

description of the SE FP adder is described. In section 5, we describe in detail all the sub-modules of the modified SE FP adder and we prove the correctness of some of the sub-modules. In section 6, we focus on the modification we did to the SE-FP adder, to support de-normalized numbers. In section 7 we describe the testing environment we used for the exhaustive tests. We conclude with a summary and discussion in section 8. In order to complete the documentation of the modified SE FP adder design we added in the appendix several block diagrams of additional sub-modules that are used by the SE-FP adder.

Chapter 3

Notations

Upper cases, lower cases and representations.
During the description of the addition algorithms (the SE-FP adder and the modified SE FP adder) we specify the sub-modules and for some of them we prove the correctness. In order to do it we use both upper cases and lower cases.

Upper cases
We use upper case letters to :

- Represent the unsigned value of a variable, and
- Represent the binary string of a variable.

For example:

- $F[-1:p-1]$ is a string of $p+1$ bits ($F[-1:p-1] \in \{0,1\}^{p+1}$) and its value is the unsigned value of the string:
  $F[-1:p-1] = \sum_{i=-1}^{p-1} F[i] \cdot 2^{-i}$
  (The indexing in the significands is opposite directed - higher on the left bits and lower on the right bits)
- $E[n-1:0]$ is a string of n bits ($E[n-1:0] \in \{0,1\}^{n}$) and its value is the unsigned value of a the string:
  $E[n-1:0] = \sum_{i=0}^{n-1} E[i] \cdot 2^{i}$

On upper case values we can do arithmetic operations and sting manipulations, for example:

- Bitwise not (ones complement): $Y[n-1:0] = \overline{X[n-1:0]}$
  In unsigned values: $Y = 2^{n-1} - X$.
- Concatenation:$Y[n+m-1:0] =< X[n-1:0], Z[m-1:0] >$
  In unsigned values: $Y = X \cdot 2^m + Z$.
- Shifting : $F_1 = F_2 \cdot 2^k$ where k is integer.
- Unsigned addition : $E_1[n:0] = E_2[n-1:0] + E_3[n-1:0]$

Lower cases

We use lower case letters to represent the real value of a variable, which can be a rational number. On lower cases we can not do string manipulations but only arithmetic operations.

Relations between lower cases and upper cases

We differ between exponents and significands.
The exponents are represented with bias and therefore:
$e = E[n-1:0] - bias_n$
where:

- $bias_n = 2^{n-1} - 1$
- e is the real value of the exponent.
- $E[n-1:0]$ is the representation of the exponent.

For example, e_{min} (that equals $2 - 2^{n-1}$) is represented by $E[n-1:0] =< 0^{n-1} \cdot 1 >= 1$.
The significands (which are, by definition, not negative) are without any transformation:
$f[0:p-1] = F[0:p-1] = \sum_{i=-1}^{p-1} F[i] \cdot 2^{-i}$
If a variable is negative then ones complement representation is used, for example:
if $F[-1] = 1$ then: $f[-1:p-1] = -(2^2 - \sum_{i=-1}^{p-1} F[i] \cdot 2^{-i})$

FP-numbers with parameterized width.

1. $S \in \{0,1\}$ donated the sign bit.
2. $E[n-1:0] \in \{0,1\}^n$ donates the exponent string. The value represented by the exponent is:
$$e = \begin{cases} e_{min} & if\ E = 0^n \\ exception & if\ E = 1^n \\ \sum_{i=0}^{n-1} E[i] \cdot 2^i - (2^n - 1) & otherwise \end{cases}$$
where $e_{min} = 2 - 2^{n-1}$ and its representation is: $E_{min}[n-1:0] = \{0^{n-1} \cdot 1\}$
3. $F[1:p-1]$ donates the significand string that represents a fraction in range $[0,2)$. When representing significands, we use the convention that bit positions to the right of the binary point have positive indices and bit positions to the left of the binary point have negative indices. Hence, the value represented by the string $F[0:p-1]$ is:
$$f = \begin{cases} \sum_{i=0}^{n-1} F[i] \cdot 2^{-i} & if\ E = 0^n \\ exception & if\ E = 1^n \\ 1 + \sum_{i=0}^{n-1} F[i] \cdot 2^{-i} & otherwise \end{cases}$$
4. The value represented by a FP-number (S,E[n-1:0],F[1:p-1]) is:
$$f = \begin{cases} (-1)^s \cdot 2^e \cdot f & if\ E \neq 0^n \\ (-1)^S \cdot \infty & if\ E = 1^n\ \&\ F = 0^{p-1} \\ NaN & if\ E = 1^n\ \&\ F = 1^{p-1} \end{cases}$$

5. The values of the parameters n and p in the IEEE standard are:

   - In single precision: n = 8 , p = 24, and
   - In double precision: n = 11, p = 53.

Inputs.
The inputs of the FP-addition are:

1. Operands donated by: (SA,EA[n-1:0],FA[0:p-1]) and (SB,EB[n-1:0],FB[0:p-1]), and
2. IEEE rounding mode.

Outputs.
The output is a FP-number (S,E[n-1:0],F[0:p-1]). The value represented by the output respects the equation:

$$(-1)^s \cdot 2^e \cdot f = round\{(-1)^{sa} \cdot 2^{ea} \cdot fa + (-1)^{sb} \cdot 2^{eb} \cdot fb\}$$

Chapter 4

The Naive FP-Addition Algorithm

In this section we overview the naive FP-addition algorithm.
Let (sa, ea, fa) and (sb, eb, fb) donate the inputs, A and B, to the FP addition. The requested computation is the IEEE FP representation of the rounded sum:

$$sum = round(A + B) = round\{(-1)^{sa} \cdot 2^{ea} \cdot fa + (-1)^{sb} \cdot 2^{eb} \cdot fb\}$$

Let $S.EFF = xor(sa, sb)$. The case that $S.EFF = 0$ is called *effective addition* and the case that $S.EFF = 1$ is called *effective substraction*.
We define the exponent difference $\delta = ea - eb$.
The "large" operand (sl, el, fl) and the "small" operand (ss, es, fs) are defined as follows:

$$(sl, el, fl) = \begin{cases} (sa, ea, fa) & if \delta \geq 0 \\ (sb, eb, fb) & \text{otherwise} \end{cases}$$

$$(ss, es, fs) = \begin{cases} (sb, eb, fb) & if \delta \geq 0 \\ (sa, ea, fa) & \text{otherwise} \end{cases}$$

The expected sum can be written as :

$$sum = round\{(-1)^{sl} \cdot 2^{el} \cdot [fl + (-1)^{s.eff} \cdot fs \cdot 2^{-|\delta|}]\}$$

The stages of the naive algorithm:

1. Exponents substraction : $\delta = ea - eb$.
2. Operands swapping : computation of (sl,el,fl) & (ss,es,fs).
3. Limitation of the alignment shift amount :

$$\delta_{lim} = min\{D, |\delta|\}$$

where D is a constant greater then p (the width of the significand).

4. Normalizing of the significand.
   let :
   $$f_{prenorm} = fl + (-1)^{s.eff} \cdot fs \cdot 2^{-|\delta|} \qquad |f_{prenorm}| < 4$$

   The absolute value of the sum before rounding is :
   $$|A + B| = |f_{prenorm}| \cdot 2^{el}$$

   There are two cases:

   - If $|A + B| \geq 2^{e_{min}}$ then:
     find integer $K \in [-1..(el - e_{min})]$ such that $|f_{prenorm}| \cdot 2^K \in [1, 2)$.

   - If $|A + B| < 2^{e_{min}}$ then:
     $K = el - e_{min}$ and $|f_{prenorm}| \cdot 2^K \in [0, 1)$.

   In both cases : $f_{prernd} = |f_{prenorm}| \cdot 2^K$.

5. Compensation of the normalization in the exponent.
   $$e_{prernd} = \begin{cases} el - K & if\ |A + B| \geq 2^{e_{min}} \\ e_{min} & \text{otherwise} \end{cases}$$

6. rounding and post-normalization.

7. Computation of the sign.
   The sign of the result is: s = xor(sl,s')
   where
   $$s' = sign\{fl + (-1)^{s.eff} \cdot fs \cdot 2^{-|\delta|}\}$$

Chapter 5

The SE FP-Adder

In this section we present a high level description of the SE FP-Adder [SE] depicted in figure 5.1. The algorithm includes two parallel paths: R-path and N-path and the result is selected

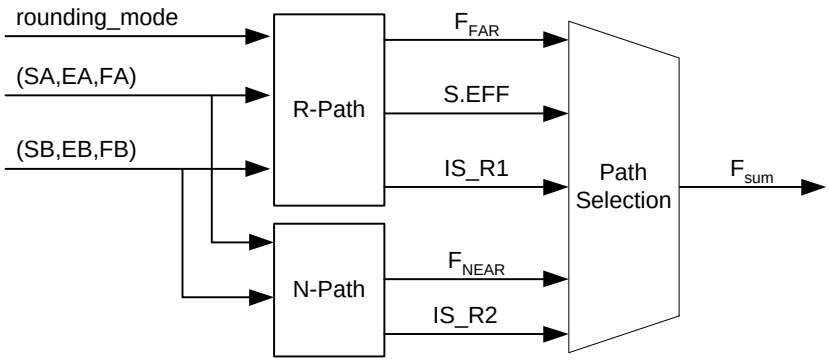

Figure 5.1: High level description of the SE FP-Adder.

from one of the paths according to indications $S.EFF$, IS_R1 and IS_R2.
The Inputs (SA, EA, FA) and (SB, EB, FB) are assumed to be in unpacked format.
The N-path calculates the unpacked format of the sum with assumption that all the following conditions hold:

- effective subtraction takes place ($S.EFF = 1$).
- The exponents difference is small: $|\delta| \leq 1$.

The R-path calculates the packed format of the sum for the rest of the cases.

5.1 The R-Path

We specify the functionality of the R-path. The R-Path top hierarchy is depicted in figure 5.2.
The first cycle calculates the values:

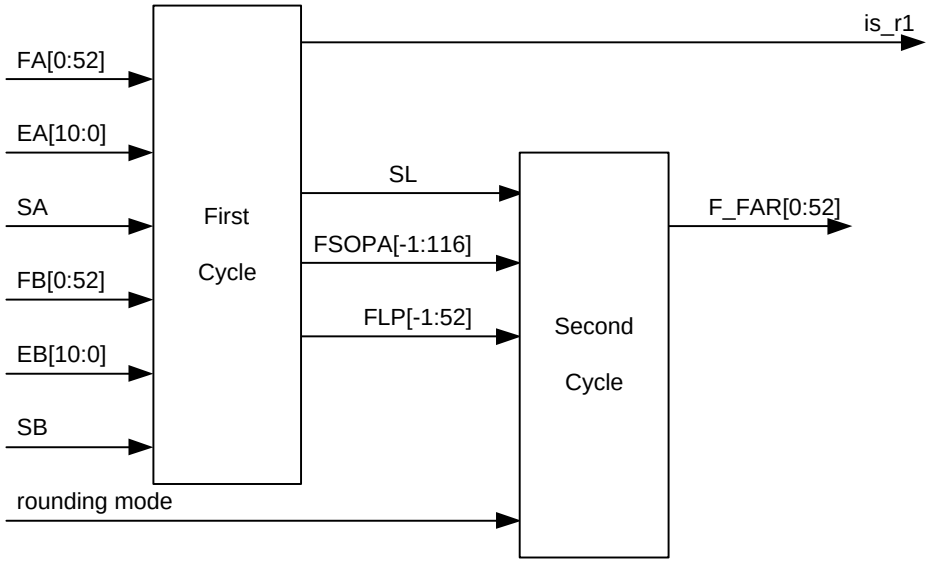

Figure 5.2: The top hierarchy of the R-Path of the SE FP-Adder

- $sl = \begin{cases} sb & if\ \delta \leq 0 \\ sa & \text{otherwise} \end{cases}$
- $flp = fl \cdot 2^{s.eff}$
- $fsopa = \begin{cases} fs \cdot 2^{-\delta_{lim}} & if\ s.eff = 0 \\ -fs \cdot 2^{-\delta_{lim}+1} & \text{otherwise} \end{cases}$
- $is\_r1 = (\delta \geq 2)$

where:

- $\delta_{lim} = min\{\delta, 64\}$

The sum $flp + fsopa$ in the two cases of $S.EFF$ is:

- S.EFF = 0
  $flp + fsopa = fl + fs \cdot 2^{-\delta_{lim}} = f_{prenorm}$
- S.EFF = 1
  $flp + fsopa = 2{\cdot}fl - fs{\cdot}2^{-\delta_{lim}+1} = 2{\cdot}(fl - fs{\cdot}2^{-\delta_{lim}}) = 2{\cdot}f_{prenorm}$

The second cycle of R-Path calculates F_FAR by rounding and post-shifting (if necessary) of the sum flp+fsopa.

5.2 The N-path

We specify the functionality of the N-path. The N-Path top hierarchy is depicted in figure 5.3.

The first cycle calculates the values:

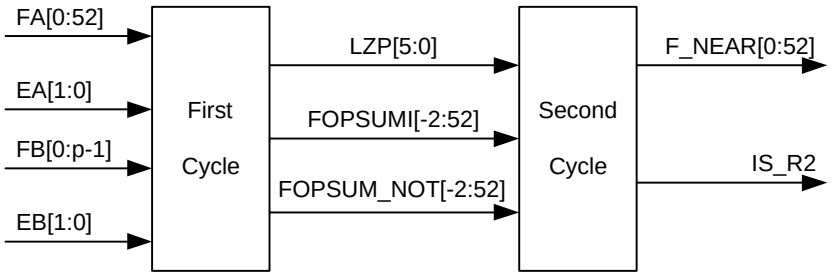

Figure 5.3: The top hierarchy of the N-Path of the SE FP-Adder

- $fopsumi = \begin{cases} |f'_{prenorm}| & if\ f'_{prenorm} > 0 \\ don'tcare & \text{otherwise} \end{cases}$
- $\overline{fopsum} = \begin{cases} |f'_{prenorm}| & if\ f'_{prenorm} \leq 0 \\ don'tcare & \text{otherwise} \end{cases}$
- $FOPSUMI[-2] = f'_{prenorm} < 0$
- $lzp \in \{min(ea - e_{min}, eb - e_{min}, \alpha), min(ea - e_{min}, eb - e_{min}, \alpha + 1)\}$

where:

- $f'_{prenorm} = fl{\cdot}2^{|\delta|} - fs$.
- α is the number of leading zeros in $|f'_{prenorm}[0 : p-1]|$.

In the second cycle of the N-path the significand of the result f_{near} is calculated. In addition, it produces the output is_r2, which is used to the path selection, and its value is:

- $is\_r2 = (fl{\cdot}2^{|\delta|} - fs \geq 2) = (f'_{prenorm} \geq 2)$.

5.3 Parameterization of The SE FP-Adder

The IEEE definition for double-precision floating point numbers is that:

- The width of the exponent is $n = 11$, and
- The width of the significand (un-packed) is $p = 53$.

Therefore, the SE-FP adder is designed with fixed width of I/O and internal signals. As a result, exhaustive testing of the SE-FP adder is impossible since there are $o(2^{128})$ (!!!) different possible combination of inputs.
In order to reduce the input space size, and do exhaustive testing, we changed the width of the SE-FP adder signals (including the I/O signals) to be parameters. Then, by defining small values to the width of exponent and significand, we could test the modified SE-FP adder exhaustively. For examples if $n = 3$ and $p = 4$ there are only $o(2^{14})$ different possible combinations of inputs which is reasonable size of inputs space to be tested exhaustively.
We parameterized the SE-FP adder by the following stages:

1. Defining parameters q, w and D in addition to n and p.

2. replacing the indexes of the signals in the SE-FP adder with expressions that depends on n, p, q, w, D.

The meanings and values of the additional parameters we defined are:

- $q = \lceil \log_2(p+2) \rceil$ - The width of the unsigned representation of δ_{lim}
- $D = 2^q - 1$ - The maximum value of δ_{lim} (D is also defined in the vanilla algorithm)
- $w = p + D$ - The index of the L.S.B of $fsopa$

The reason for the expression of q is that q bits should be minimal and sufficient for unsigned representation of δ_{lim} which is the amount of the alignment shift that is done to fs.
Since all alignment shifts that are beyond the sticky bit are equivalent we can limit the alignment shift. The maximum alignment shift is shifting bit $fsop[-1]$ (see figure 6.2) to the sticky bit that its index is $p+1$. Therefore the maximum shift amount is $p+2$, and the number of bits that are required to describe $p+2$ are $\lceil \log_2(p+2) \rceil$ as defined.
Since δ_{lim} is represented by q bits its value is limited to D as defined.
The alignment shift produces $fsopa[-1:w]$ (see figure 6.2) that must include enough bits to represent precisely the alignment shift result. Therefore, if the index of the L.S.B of $fsop$ (the input to the alignment shifter) is p and the maximum shift is D then $w = p + D$ as defined.

Chapter 6

Detailed Description of The Modified SE FP-Adder

In this section we describe the sub modules of the modified SE FP adder.We specify each module and depict its design diagram. In addition we prove or explain the correctness of the submodule. The sub-modules with the modifications are explained briefly in this section and in detail in section "The modifications of SE FP adder".

6.1 The exponent difference

This module is part of the first cycle of R-path. It receives the exponents of the addends and produces indications about the difference between the exponents. The module is depicted in figure 6.1.

6.1.1 Specification

The "exponent-difference" module receives the unpacked exponents of the addends and calculates the following values:

- $is\_big = (\delta \leq -2^q)\ or\ (\delta \geq 2^q + 1)$
- $sign\_big = (\delta \leq 0)$
- $is\_r1 = (|\delta| \geq 1)$
- $mag\_med = \begin{cases} |\delta| - 1 & if\ \overline{is\_big}\ \&\ (\delta > 0) \\ |\delta| & if\ \overline{is\_big}\ \&\ (\delta \leq 0) \\ don't care & \text{otherwise} \end{cases}$
- $sign\_med = \begin{cases} (\delta \leq 0) & if \overline{is\_big} \\ don't care & \text{otherwise} \end{cases}$
- el = max{ea,eb}

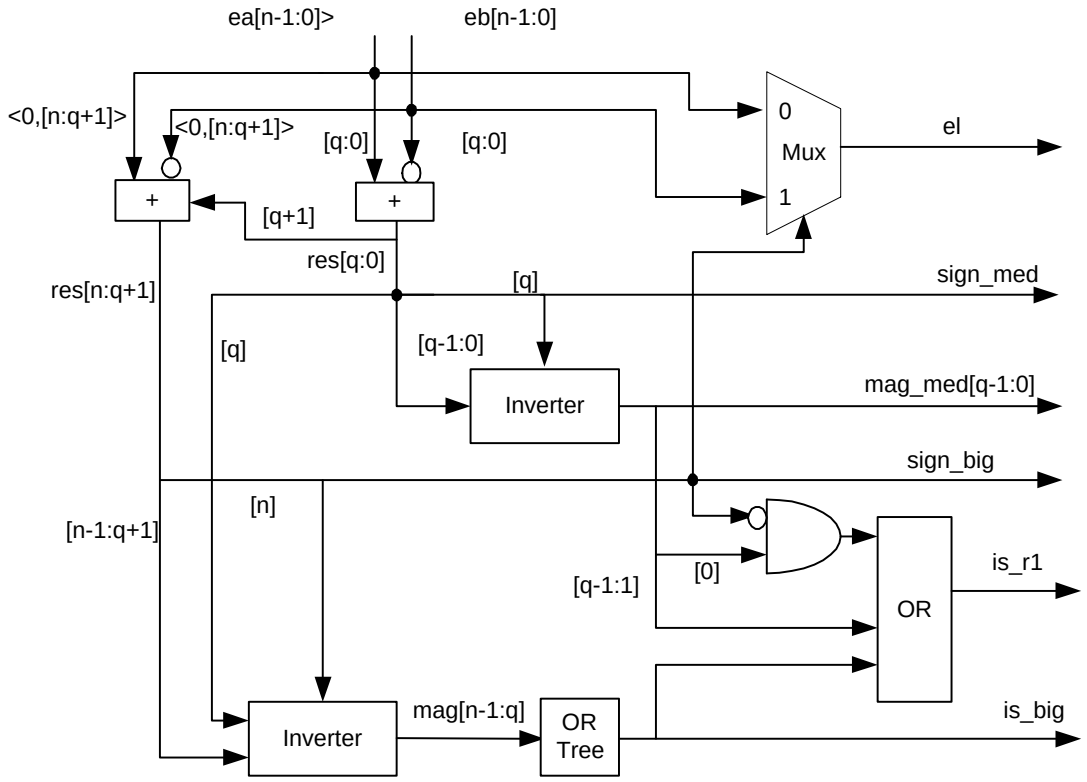

Figure 6.1: The exponent difference module

6.1.2 Correctness

We prove the correctness of the module (namely, that the module depicted in figure 6.1 computes the correct outputs values).
The proof stages:

1. Finding boundaries for δ.
2. Expressing $RES[n:0]$ as a function of δ.
3. Proving the correctness of is_big and $sign\_big$.
4. Proving the correctness of mag_med and $sign\_med$.
5. Proving the correctness of is_r1.

The correctness of el can be verified immediately from the module diagram.

stage 1: Boundaries of δ
Each addend exponent includes **n** bits, therefore:

$$EA \in [0, 2^n - 1] \tag{6.1}$$

$$EB \in [0, 2^n - 1] \tag{6.2}$$

The definition of δ:

$$\delta = ea - eb \tag{6.3}$$

The exponents are represented with bias, therefore:

$$\begin{aligned} \delta &= ea - eb \\ &= (EA - bias_n) - (EB - bias_n) \\ &= EA - EB \end{aligned} \tag{6.4}$$

According to equations (6.1),(6.2) and (6.4) δ is limited:

$$\delta \in [-(2^n - 1), 2^n - 1] \tag{6.5}$$

stage 2: $RES[n:0]$ as a function of δ
The equation of $RES[n:0]$ in the "exponent difference" module is:

$$\begin{aligned} RES[n:0] &= mod(< 0, EA[n-1:0] > + \overline{< 0, EB[n-1:0] >}, 2^{n+1}) \\ &= mod(ea + 2^n + \overline{EB[n-1:0]}, 2^{n+1}) \\ &= mod(ea + 2^n + 2^n - 1 - eb, 2^{n+1}) \\ &= mod(2^{n+1} + \delta - 1, 2^{n+1}) \\ &= mod(\delta - 1, 2^{n+1}) \end{aligned} \tag{6.6}$$

stage 3: Proof of *is_big* and *sign_big*
We will check 2 cases of δ:

1. $\delta \in [1, 2^n - 1]$.
2. $\delta \in [-(2^n - 1), 0]$.

case 1:
if $\delta \in [1, 2^n - 1]$ then $(\delta - 1) \in [0, 2^n - 2]$. We proved that $RES[n:0] = mod(\delta - 1, 2^{n+1})$ (equation 6.6). Therefore:

$$\begin{aligned} RES[n:0] &= \delta - 1. \\ &\in [0, 2^n - 2] \end{aligned} \tag{6.7}$$

$RES[n:0]$ is small enough such that:

$$\begin{aligned} sign\_big &= RES[n] \quad \text{(signals } SIGN\_BIG \text{ and } RES[n] \text{ are} \\ &\qquad \text{the same wire in the design)} \\ &= 0 \end{aligned} \tag{6.8}$$

and we can write:

$$\begin{aligned} RES[n-1:0] &= RES[n:0] \\ &= \delta - 1 \end{aligned} \tag{6.9}$$

The signal IS_BIG is an output of an or-tree feed by mag[n-1:q]. Therefore:

$$is\_big = or(MAG[n-1:q]) \tag{6.10}$$

The equation of $MAG[n-1:q]$ in the "exponent difference" module is:

$$MAG[n-1:q] \quad = \begin{cases} RES[n-1:q] & \text{if RES[n]} = 0 \\ \overline{RES[n-1:q]} & \text{otherwise} \end{cases} \tag{6.11}$$

According to equations (6.8),(6.10) and (6.11):

$$\begin{aligned} is\_big &= or(MAG[n-1:q]) \quad \text{(equation 6.10)} \\ &= or(RES[n-1:q]) \quad \text{(equations 6.8 and 6.11)} \\ &= (RES[n-1:0] \geq 2^q) \\ &= (\delta - 1 \geq 2^q) \\ &= (\delta \geq 2^q + 1) \end{aligned} \tag{6.12}$$

case 2:
if $\delta \in [-(2^n-1), 0]$ then $(\delta - 1) \in [-2^n, -1]$. We proved that $RES[n:0] = mod(\delta - 1, 2^{n+1})$ (equation 6.6). Therefore:

$$\begin{aligned} RES[n:0] &= 2^{n+1} + \delta - 1. \\ &\in [2^n, 2^{n+1} - 1] \end{aligned} \tag{6.13}$$

$RES[n:0]$ is large enough such that:

$$\begin{aligned} sign\_big &= RES[n] \quad \text{(signals } SIGN\_BIG \text{ and } RES[n] \text{ are} \\ &\qquad \text{the same wire in the design)} \\ &= 1 \end{aligned} \tag{6.14}$$

and we can write:

$$\begin{aligned} RES[n-1:0] &= 2^{n+1} + \delta - 1 - 2^n \\ &= 2^n + \delta - 1 \end{aligned} \tag{6.15}$$

According to equations (6.11) and (6.14):

$$MAG\_MED[n-1:q] = \overline{RES[n-1:q]} \tag{6.16}$$

and:

$$\begin{aligned} is\_big &= or(\overline{MAG[n-1:q]}) \quad \text{(equation 6.10)} \\ &= or(\overline{RES[n-1:q]}) \quad \text{(equation 6.16)} \\ &= or(2^n - 1 - RES[n-1:0] \geq 2^q) \\ &= or(2^n - 1 - (2^n - 1 + \delta) \geq 2^q) \\ &= or(-\delta \geq 2^q) \\ &= or(\delta \leq -2^q) \end{aligned} \tag{6.17}$$

summary of case 1 and case 2:
In case (1) $sign\_big = 0$ and $is\_big = (\delta \geq 2^q + 1)$ and in case (2) $sign\_big = 1$ and $is\_big = (\delta \leq -2^q)$ which is identical to:

$$sign\_big = (\delta \leq 0) \tag{6.18}$$

$$is\_big = (\delta \leq -2^q) \text{ or} (\delta \geq 2^q + 1) \tag{6.19}$$

as required.

stage 4: Proof of mag_med and $sign\_med$

According to the specifications of the "exponent difference" module:
If $is\_big = 1$ then the values of mag_med and $sign\_med$ are "don't care" (since they are not in use by the SE - FP adder if $is\_big = 1$). Therefore it is enough to prove that:
if $is\_big = 0$ then:

$$sign\_med = (\delta \leq 0) \tag{6.20}$$

$$mag\_med = |\delta| - (\delta \geq 1) \tag{6.21}$$

According to equation (6.19):
if $is\_big = 0$ then $-2^q < \delta \leq 2^q$, therefore we check the following cases of δ:

1. $\delta \in [1, 2^q]$
2. $\delta \in (-2^q, 0]$

case 1:

We proved that if $\delta \in [1, 2^q]$ then:

$$\begin{aligned} RES[n-1:0] &= \delta - 1 \quad \text{(equation 6.9)} \\ &\leq 2^q - 1 \end{aligned} \tag{6.22}$$

$$RES[n-1:0] \quad \in < 0^{n-q} \cdot \{0,1\}^q > \tag{6.23}$$

and therefore:

$$\begin{aligned} sign\_med &= RES[q] \quad \text{(signals } SIGN\_MED \text{ and } RES[q] \text{ are the same wire in the design)} \\ &= 0 \end{aligned} \tag{6.24}$$

$$\begin{aligned} mag\_med[q-1:0] &= RES[q-1:0] \\ &= RES[n-1:0] \\ &= \delta - 1 \end{aligned} \tag{6.25}$$

case 2:

We proved that if $\delta \in (-2^q, 0]$ then $RES[n:0] = 2^{n+1} + \delta - 1$ (equation 6.13). Therefore:

$$\begin{aligned} sign\_med &= RES[q] \\ &= (mod(RES[n:0], 2^{q+1}) \geq 2^q) \\ &= (mod(2^{n+1} + \delta - 1, 2^{q+1}) \geq 2^q) \\ &= (mod(\delta - 1, 2^{q+1}) \geq 2^q) && (2^{n+1} \text{ is an integer multiplication of } 2^{q+1}) \\ &= (2^{q+1} + \delta - 1 \geq 2^q) && (\text{if } is\_big = 0 \text{ then } \delta \in (-2^q, 0]) \\ &= (\delta \geq -2^q + 1) \\ &= 1 \end{aligned} \tag{6.26}$$

$$
\begin{aligned}
mag\_med[q-1:0] &= \overline{RES[q-1:0]} \\
&=< 0^{n-q+1}, \overline{RES[q-1:0]} > \\
&= \overline{< 1^{n-q+1}, RES[q-1:0] >} \\
&= \overline{RES[n:0]} && (RES[n:0] = 2^{n+1} + \delta - 1 > 2^{n-1} - 2^q \\
& && \text{and therefore } RES[n:q] =< 1^{n-q+1} >) \\
&= 2^{n+1} - 1 - (2^{n+1} + \delta - 1) \\
&= -\delta \\
&= |\delta|
\end{aligned}
\tag{6.27}
$$

summary of case 1 and case 2:
In case 1 $sign\_med = 0$ (equation 6.24) and $mag\_med = \delta - 1$ (equation 6.25) and in case 2 $sign\_med = 1$ (equation 6.26) and $mag\_med = |\delta|$ (equation 6.27) which is identical to:

$$sign\_med = (\delta \leq 0) \tag{6.28}$$

$$mag\_med = |\delta| - (\delta \geq 1) \tag{6.29}$$

as required.

stage 5: Proof of is_r1
The equation of is_r1 is :

$$is\_r1 = is\_big \; or \; (mag\_med[q-1:1] \geq 1) \; or \; (mag\_med[0] \; \& \; \overline{sign\_big}) \tag{6.30}$$

We will check the values of is_r1 in a few intervals of δ:

1. $\delta \leq -2^q$
   According to equation (6.19), if $\delta \leq -2^q$ then $is\_big = 1$,
   and therefore, according to equation (6.30): $is\_r1 = 1$.

2. $-2^q < \delta \leq -2$
   According to equation (6.21), if $-2^q < \delta \leq -2$ then $mag\_med = |\delta|$.
   In this interval of δ : $mag\_med = |\delta| \in [2, 2^q - 1]$, and therefore $1 \leq mag\_med[q-1:1]$.
   According to equation (6.30): $is\_r1 = 1$.

3. $-1 \leq \delta \leq 0$
   According to equation (6.21), if $-1 \leq \delta \leq 0$ then $mag\_med = |\delta|$.
   In this interval of δ : $mag\_med = |\delta| \in [0, 1]$ and Therefore $mag\_med[q-1:1] = 0$.
   According to equation (6.18) sign_big = 1, and therefore, $(mag\_med[0] \; \& \; \overline{sign\_big}) = 0$.
   According to equation (6.19) is_big = 0, and therefore, according to equation (6.30): $is\_r1 = 0$.

4. $\delta = 1$
   According to equation (6.21), if $\delta = 1$ then $mag\_med = 0$, and according to equation (6.19) $is\_big = 0$.
   Therefore, according to equation (6.30): $is\_r1 = 0$.

5. $\underline{\delta = 2}$
   According to equation (6.21), if $\delta = 2$ then $mag\_med = 1$, and according to equation (6.18) $sign\_big = 0$.
   Therefore:
   $mag\_med[0]$ & $\overline{sign\_big} = 1$, and according to equation (6.30): $is\_r1 = 1$.

6. $\underline{3 \leq \delta \leq 2^q}$
   According to equation (6.21), if $3 \leq \delta \leq 2^q$ then $mag\_med = \delta - 1$.
   In this interval of δ : $mag\_med = \delta - 1 \in [2, 2^q - 1]$ and therefore $1 \leq mag\_med[q-1:1]$.
   According to equation (6.30): $is\_r1 = 1$.

7. $\underline{2^q < \delta}$
   According to equation (6.19), if $2^q < \delta$ then $is\_big = 1$
   and therefore, according to equation (6.30) : $is\_r1 = 1$.

We can see that $is\_r1 = (|\delta| \geq 2)$ as required.

6.2 R-path first cycle

The module receives the addends (SA, EA, FA) and (SB, EB, FB) and produces the intermediate values FLP and $FSOPA$ which are one's complement representation of flp and $fsopa$. The module is depicted in figure 6.2.

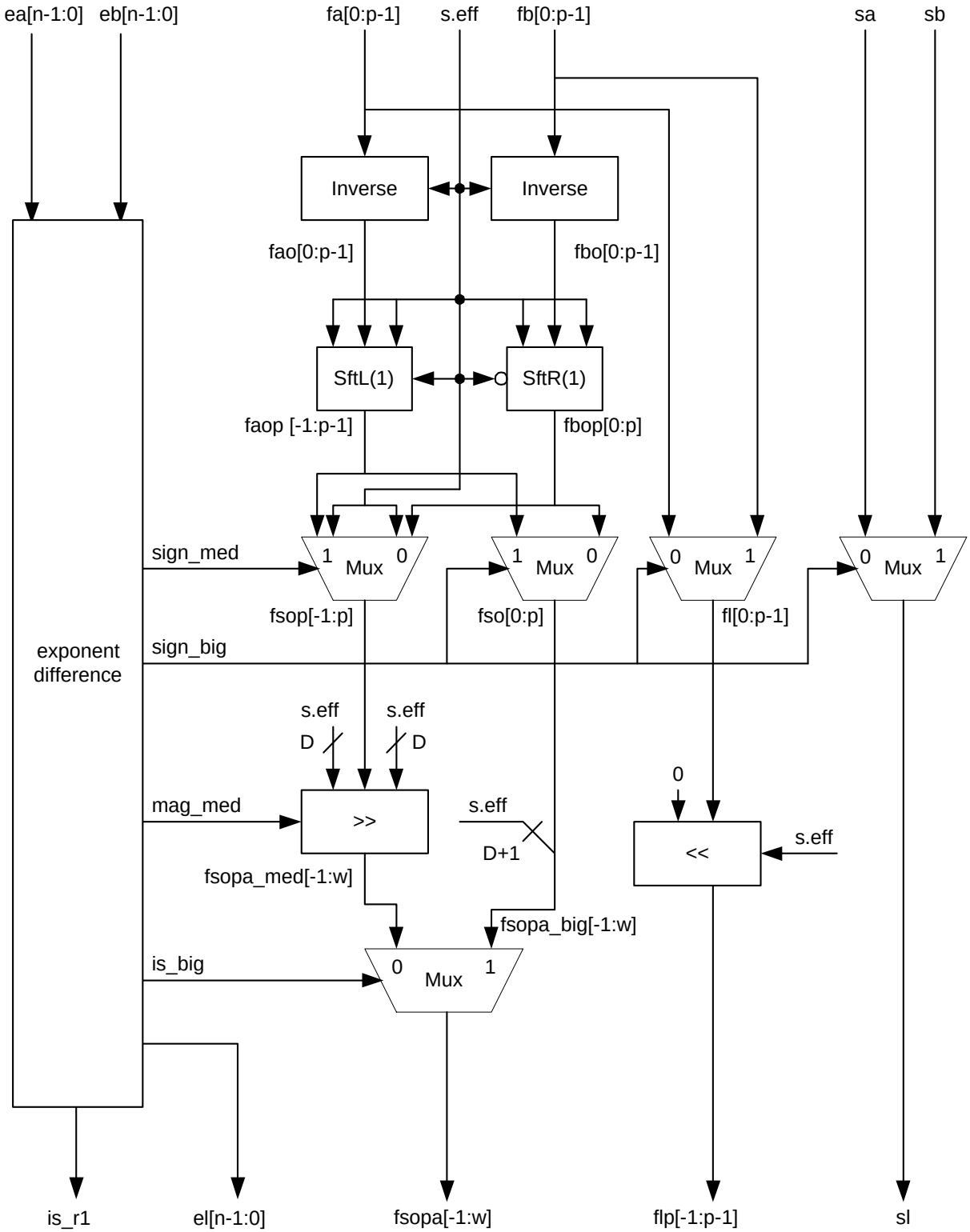

Figure 6.2: The R-path of the modified SE FP-adder

6.2.1 Specification

The first cycle of R-path receives the signs, exponents and significands of the addends (in unpacked format), and calculates the following values:

- $sl = \begin{cases} sb & if\ \delta \leq 0 \\ sa & \text{otherwise} \end{cases}$
- $flp = fl \cdot 2^{s.eff}$
- $fsopa = \begin{cases} fs \cdot 2^{-\delta_{lim}} & if\ s.eff = 0 \\ -fs \cdot 2^{-\delta_{lim}+1} & \text{otherwise} \end{cases}$
- $is\_r1 = (\delta \geq 2)$
- $el = max\{ea, eb\}$

where:

- $\delta_{lim} = min\{\delta, 2^q\}$

6.2.2 Correctness

The correctness of sl and flp can be verified immediately in the R-path diagram. We prove the correctness of $fsopa$ in the following stages:

1. listing the equations of the module.
2. Calculating $FSOPA$ for all 4 combinations of $s.eff$ and is_big.
3. Calculating $FSOPA$ for $s.eff = 0$ and $s.eff = 1$ in the following ranges of δ:
   - $\delta \leq -2^q$,
   - $-2^q < \delta \leq 0$,
   - $1 \leq \delta \leq 2^q$, and
   - $2^q < \delta$.
4. Summarizing the expressions of $FSOPA$ for $s.eff = 0$ & $s.eff = 1$.

stage 1: The equations of the R-path module

$$FAO[0:p-1] = \begin{cases} FA[0:p-1] & if\ s.eff = 0 \\ \overline{FA[0:p-1]} & \text{otherwise} \end{cases} \tag{6.31}$$

$$FBO[0:p-1] = \begin{cases} FB[0:p-1] & if\ s.eff = 0 \\ \overline{FB[0:p-1]} & \text{otherwise} \end{cases} \tag{6.32}$$

$$FAOP[-1:p-1] = \begin{cases} < 0, FAO[0:p-1] > & if\ s.eff = 0 \\ < FAO[0:p-1], 1 > & \text{otherwise} \end{cases}$$
$$= \begin{cases} < 0, FA[0:p-1] > & if\ s.eff = 0 \\ < \overline{FA[0:p-1]}, 1 > & \text{otherwise} \end{cases} \tag{6.33}$$

$$FBOP[0:p] = \begin{cases} < 0, FBO[0:p-1] > & if\ s.eff = 0 \\ < FBO[0:p-1], 1 > & \text{otherwise} \end{cases}$$
$$= \begin{cases} < 0, FB[0:p-1] > & if\ s.eff = 0 \\ < \overline{FB[0:p-1]}, 1 > & \text{otherwise} \end{cases} \tag{6.34}$$

$$FSOP[-1:p] = \begin{cases} < s.eff, FBOP[0:p] > & if\ sign\_med = 0 \\ < FAOP[-1:p-1], s.eff > & \text{otherwise} \end{cases} \tag{6.35}$$

$$FSO[0:p] = \begin{cases} FBOP[0:p] & if\ sign\_big = 0 \\ FAOP[-1:p-1] & \text{otherwise} \end{cases} \tag{6.36}$$

$$FSOPA\_MED[-1:w] =< s.eff^{mag\_med}, FSOP[-1:p], s.eff^{w+2-(p+1)-mag\_med} >$$
$$=< s.eff^{mag\_med}, FSOP[-1:p], s.eff^{2^q-1-mag\_med} > \tag{6.37}$$

$$FSOPA\_BIG[-1:w] =< s.eff^{2^q}, FSO[0:p] > \tag{6.38}$$

$$FSOPA[-1:w] = \begin{cases} FSOPA\_BIG[-1:w] & if\ is\_big = 1 \\ FSOPA\_MED[-1:w] & \text{otherwise} \end{cases} \tag{6.39}$$

stage 2: $FSOPA$ for all 4 combinations of $s.eff$ and is_big

$s.eff = 0, is\_big = 0$

According to equations (6.33),(6.34) and (6.35), if $s.eff = 0$ then:

$$FSOP[-1:p] = \begin{cases} < 0^2, FB[0:p-1] > & if\ sign\_med = 0 \\ < 0, FA[0:p-1], 0 > & \text{otherwise} \end{cases} \tag{6.40}$$

According to equations (6.37),(6.39) and (6.40), if $is\_big = 0$ then:

$$\begin{aligned} FSOPA[-1:w] &= FSOPA\_MED[-1:w] \\ &=< 0^{mag\_med}, FSOP[-1:p], 0^{2^q-1-mag\_med} > \\ &= \begin{cases} < 0^{mag\_med+2}, FB[0:p-1], 0^{2^q-1-mag\_med} > & if\ sign\_med = 0 \\ < 0^{mag\_med+1}, FA[0:p-1], 0^{2^q-mag\_med} > & \text{otherwise} \end{cases} \end{aligned} \tag{6.41}$$

$\underline{s.eff = 0, is\_big = 1}$

According to equations (6.33),(6.34) and (6.36), if $s.eff = 0$ then:

$$FSO[0:p] = \begin{cases} < 0, FB[0:p-1] > & if\ sign\_big = 0 \\ < 0, FA[0:p-1] > & \text{otherwise} \end{cases} \tag{6.42}$$

According to equations (6.38),(6.39) and (6.40), if $is\_big = 1$ then:

$$\begin{aligned} FSOPA[-1:w] &= FSOPA\_BIG[-1:w] \\ &=< s.eff^{2^q}, FSO[0:p] > \\ &= \begin{cases} < 0^{2^q+1}, FB[0:p-1] > & if\ sign\_big = 0 \\ < 0^{2^q+1}, FA[0:p-1] > & \text{otherwise} \end{cases} \end{aligned} \tag{6.43}$$

$\underline{s.eff = 1, is\_big = 0}$

According to equations (6.33),(6.34) and (6.35), if $s.eff = 1$ then:

$$\begin{aligned} FSOP[-1:p] &= \begin{cases} < 1, FBOP[0:p] > & if\ sign\_med = 0 \\ < FAOP[-1:p-1], 1 > & \text{otherwise} \end{cases} \\ &= \begin{cases} < 1, \overline{FB[0:p-1]}, 1 > & if\ sign\_med = 0 \\ < \overline{FA[0:p-1]}, 1^2 > & \text{otherwise} \end{cases} \end{aligned} \tag{6.44}$$

According to equations (6.37),(6.39) and (6.44), if $is\_big = 0$ then:

$$\begin{aligned} FSOPA[-1:w] &= FSOPA\_MED[-1:w] \\ &=< 1^{mag\_med}, FSOP[-1:p], 1^{2^q-1-mag\_med} > \\ &= \begin{cases} < 1^{mag\_med+1}, \overline{FB[0:p-1]}, 1^{2^q-mag\_med} > & if\ sign\_med = 0 \\ < 1^{mag\_med}, \overline{FA[0:p-1]}, 1^{2^q+1-mag\_med} > & \text{otherwise} \end{cases} \end{aligned} \tag{6.45}$$

$\underline{s.eff = 1, is\_big = 1}$

According to equations (6.33),(6.34) and (6.36), if $s.eff = 1$ then:

$$FSO[0:p] = \begin{cases} < \overline{FB[0:p-1]}, 1 > & if\ sign\_big = 0 \\ < \overline{FA[0:p-1]}, 1 > & \text{otherwise} \end{cases} \tag{6.46}$$

According to equations (6.38),(6.39) and (6.46), if $is\_big = 1$ then:

$$\begin{aligned} FSOPA[-1:w] &= FSOPA\_BIG[-1:w] \\ &=< 1^{2^q}, FSO[0:p] > \\ &= \begin{cases} < 1^{2^q}, \overline{FB[0:p-1]}, 1 > & if\ sign\_big = 0 \\ < 1^{2^q}, \overline{FA[0:p-1]}, 1 > & \text{otherwise} \end{cases} \end{aligned} \tag{6.47}$$

Stage 3: $FSOPA$ for $s.eff = 0$ and $s.eff = 1$ in 4 ranges of δ

$\delta \leq -2^q, s.eff = 0$

We proved in the "exponent difference" section that if $\delta \leq -2^q$ then

- $is\_big = 1$, and
- $sign\_big = 1$.

Therefore, according to equation (6.43):

$$FSOPA[-1:w] =< 0^{2^q+1}, FA[0:p-1] > \tag{6.48}$$

$$fsopa = fa \cdot 2^{-2^q} \tag{6.49}$$

$-2^q < \delta \leq 0, s.eff = 0$

We proved in the "exponent difference" section that if $-2^q < \delta \leq 0$ then

- $is\_big = 0$,
- $sign\_med = 1$, and
- $mag\_med = |\delta|$.

Therefore, according to equation (6.41):

$$FSOPA[-1:w] \quad =< 0^{mag\_med+1}, FA[0:p-1], 0^{2^q-mag\_med} > \tag{6.50}$$

$$fsopa \quad = fa \cdot 2^{-|\delta|} \tag{6.51}$$

$0 < \delta \leq 2^q, s.eff = 0$

We proved in the "exponent difference" section that if $0 < \delta \leq 2^q$ then

- $is\_big = 0$,
- $sign\_med = 0$, and
- $mag\_med = |\delta| - 1$.

Therefore, according to equation (6.41):

$$\begin{aligned} FSOPA[-1:w] \quad &=< 0^{mag\_med+2}, FB[0:p-1], 0^{2^q-1-mag\_med} > \\ &=< 0^{|\delta|+1}, FB[0:p-1], 0^{2^q-|\delta|} > \end{aligned} \tag{6.52}$$

$$fsopa \quad = fb \cdot 2^{-|\delta|} \tag{6.53}$$

$2^q < \delta, s.eff = 0$

We proved in the "exponent difference" section that if $2^q < \delta$ then

- $is\_big = 1$, and

- $sign\_big = 0$.

Therefore, according to equation (6.43):

$$FSOPA[-1:w] =< 0^{2^q+1}, \overline{FB[0:p-1]} > \tag{6.54}$$

$$fsopa = fb \cdot 2^{-2^q} \tag{6.55}$$

$\underline{\delta \leq -2^q, s.eff = 1}$
We proved in the "exponent difference" section that if $\delta \leq -2^q$ then

- $is\_big = 1$, and
- $sign\_big = 1$.

Therefore, according to equation (6.47):

$$FSOPA[-1:w] \quad =< 1^{2^q}, \overline{FA[0:p-1]}, 1 > \tag{6.56}$$

$$fsopa \quad = -fa \cdot 2^{-2^q+1} \tag{6.57}$$

$\underline{-2^q < \delta \leq 0, s.eff = 1}$
We proved in the "exponent difference" section that if $-2^q < \delta \leq 0$ then

- $is\_big = 0$,
- $sign\_med = 1$, and
- $mag\_med = |\delta|$.

Therefore, according to equation (6.45):

$$\begin{aligned} FSOPA[-1:w] \quad &=< 1^{mag\_med}, \overline{FA[0:p-1]}, 1^{2^q+1-mag\_med} > \\ &=< 1^{|\delta|}, \overline{FA[0:p-1]}, 1^{2^q+1-|\delta|} > \end{aligned} \tag{6.58}$$

$$fsopa \quad = -fa \cdot 2^{-|\delta|+1} \tag{6.59}$$

$\underline{0 < \delta \leq 2^q, s.eff = 1}$
We proved in the "exponent difference" section that if $0 < \delta \leq 2^q$ then

- $is\_big = 0$,
- $sign\_med = 0$, and
- $mag\_med = |\delta| - 1$.

Therefore, according to equation (6.45):

$$\begin{aligned} FSOPA[-1:w] \quad &=< 1^{mag\_med+1}, \overline{FB[0:p-1]}, 1^{2^q-mag\_med} > \\ &=< 1^{|\delta|}, \overline{FB[0:p-1]}, 1^{2^q+1-|\delta|} > \end{aligned} \tag{6.60}$$

$$fsopa \quad = -fb \cdot 2^{-|\delta|+1} \tag{6.61}$$

$\underline{2^q < \delta, s.eff = 1}$
We proved in the "exponent difference" section that if $2^q < \delta$ then

- $is\_big = 1$, and
- $sign\_big = 0$.

Therefore, according to equation (6.47):

$$FSOPA[-1:w] =< 1^{2^q}, \overline{FB[0:p-1]}, 1 > \tag{6.62}$$

$$fsopa = -fb \cdot 2^{-2^q+1} \tag{6.63}$$

Stage 3: Summarizing the expressions of $fsopa$ for $s.eff = 0$ & $s.eff = 1$

Summary for $s.eff = 0$

According to equations (6.49),(6.51),(6.53) and (6.55): $fsopa = \begin{cases} fa \cdot 2^{-2^{2^q}} & if\ \delta \leq -2^q \\ fa \cdot 2^{-2^{|\delta|}} & if\ -2^q < \delta \leq 0 \\ fb \cdot 2^{-2^{|\delta|}} & if\ 0 < \delta \leq 2^q \\ fb \cdot 2^{-2^{2^q}} & if\ 2^q < \delta \end{cases}$

The definitions of fs and δ_lim are:

- $\delta_{lim} = min\{|\delta|, 2^q\}$
- $fs = \begin{cases} fa & if\ \delta \leq 0 \\ fb & if\ 0 < \delta \end{cases}$

Therefore:
$fsopa = fs \cdot 2^{-\delta_{lim}}$

as required when $s.eff = 0$.

Summary for $s.eff = 1$

According to equations (6.57),(6.59),(6.61) and (6.63): $fsopa = \begin{cases} -fa \cdot 2^{-2^{2^q+1}} & if\ \delta \leq -2^q \\ -fa \cdot 2^{-2^{|\delta|+1}} & if\ -2^q < \delta \leq 0 \\ -fb \cdot 2^{-2^{|\delta|+1}} & if\ 0 < \delta \leq 2^q \\ -fb \cdot 2^{-2^{2^q+1}} & if\ 2^q < \delta \end{cases}$

The definitions of fs and δ_lim are:

- $\delta_{lim} = min\{|\delta|, 2^q\}$
- $fs = \begin{cases} fa & if\ \delta \leq 0 \\ fb & if\ 0 < \delta \end{cases}$

Therefore, using the definitions of fs and δ_lim :
$fsopa = -fs \cdot 2^{-\delta_{lim}+1}$

as required when $s.eff = 1$.

6.3 R-path second cycle

The module receives the intermediate values $flp, fsopa, sl, s.eff$ and el. In addition it receives the required rounding mode, and produces the significand and the exponent of the result - f_{far} *and* e_{far}. In this module we integrated between:

- The second cycle of R-path of SE FP-Adder.
- The ES rounding algorithm for FP multiplication (see ref [ES]).

The module is depicted in figure 6.3.

The module includes two pathes (marked as "path-1" and "path-2" in the module diagram).Path 1 computes the two candidates to be $F_{far}[0:p-2]$ ($FFARP$ and $FFARIP$ in the diagram). The second path do the following:

- Computes the two candidates to be $F_{far}[p-1]$,
- Makes the rounding decision, and
- Selects between the candidates to produce $F_{far}[0:p-1]$.

In the following sections we prove the computation of $FFARP$ and $FFARIP$ and we explain the computation of $< C[p-1], R', S' >$ which are the inputs to the rounding decision and the computation of $F_{far}[p-1]$. The computation of the rounding decision and $F_{far}[p-1]$ explained in detail in reference [ES].

6.3.1 Computation of the candidates of $F_{far}[0:p-2]$

In path-1 of the second cycle of R-path the signals $FOPSUM[-1:p-2], FOPSUMI[-1:p-2]$ are calculated, and $FFARP, FFARIP$ are their overflow correction respectively. Therefore, in order to prove that:

$$F_{far}[0:p-2] \in \{FFARP[0:p-2], FFARIP[0:p-2] \tag{6.64}$$

we define a virtual signal:

$$FOPSUMV[-1:w] \quad = mod(FLP[-1:p-1] + FSOPA[-1:w], 4) \tag{6.65}$$

and prove that:

$$\begin{aligned} FOPSUMV[-1:w] \quad &= \begin{cases} FL + FS \cdot 2^{-\delta_{lim}} & if\ s.eff = 0 \\ 2{\cdot}(FL - FS \cdot 2^{-\delta_{lim}}) - 2^{-w} & \text{otherwise} \end{cases} \\ &= \begin{cases} F_{prenorm} & if\ s.eff = 0 \\ 2{\cdot}F_{prenorm} - 2^{-w} & \text{otherwise} \end{cases} \end{aligned} \tag{6.66}$$

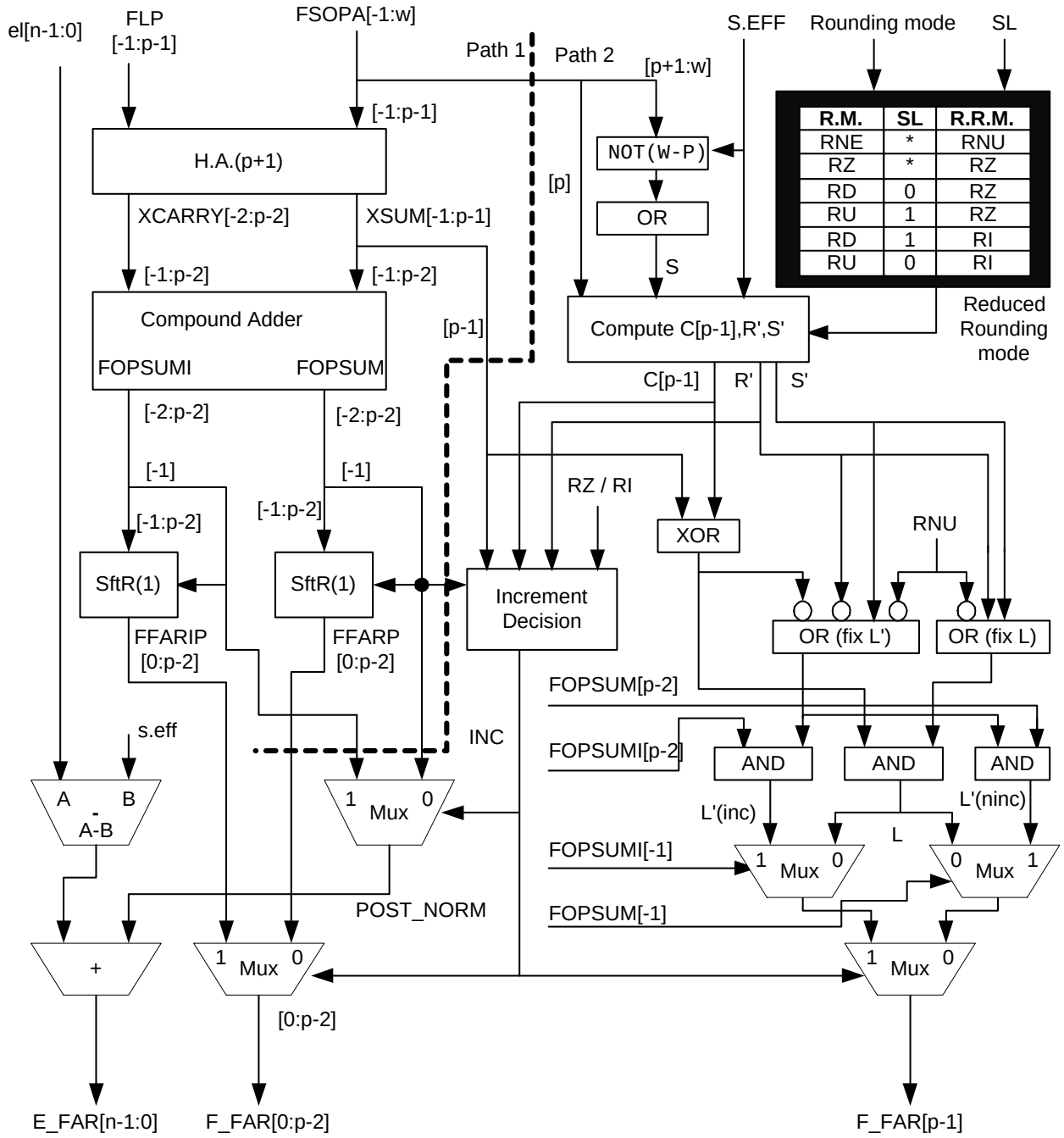

Figure 6.3: The second cycle of the R-path of the modified SE FP-adder

It is obvious from the implementation of $FOPSUM[-1:p-2]$ and $FOPSUMI[-1:p-2]$ that:

$$FOPSUM[-1:p-2] = mod(FLP + FSOPA, 4) - \beta \tag{6.67}$$

where $\beta \in [0, 2^{-(p-2)})$

¿From equations (65),(66) and (67) :

$$FOPSUM[-1:p-2] \quad = \begin{cases} F_{prenorm} - \beta & if\ s.eff = 0 \\ 2{\cdot}F_{prenorm} - \beta - 2^{-w} & \text{otherwise} \end{cases} \tag{6.68}$$

Since 2^{-w} is the minimal unit in the design and $\beta \in [0, 2^{-(p-2)})$:

$$FOPSUM[-1:p-2] \leq F'_{prenorm} \leq FOPSUMI[-1:p-2] \tag{6.69}$$

where:
$F'_{prenorm} = \begin{cases} F_{prenorm} & if\ s.eff = 0 \\ 2{\cdot}F_{prenorm} & \text{otherwise} \end{cases}$

We still have to prove that $F'_{prenorm}$ can be corrected by the overflow-correction:

$$0 \leq F'_{prenorm} < 4 \tag{6.70}$$

Proof of equation (6.66)
¿From the specifications of the first cycle of R-path:

$$FLP \quad = \begin{cases} FL & if\ s.eff = 0 \\ 2{\cdot}FL & \text{otherwise} \end{cases} \tag{6.71}$$

$$FSOPA \quad = \begin{cases} FS \cdot 2^{-\delta_{lim}} & if\ s.eff = 0 \\ \overline{FS \cdot 2^{-\delta_{lim}+1}} & \text{otherwise} \end{cases} \tag{6.72}$$

According to equations (6.65),(6.71) and (6.72):

$$\begin{aligned} &FOPSUMV[-1:w] = \\ &= \begin{cases} mod(FL[0:p-1] + FS[0:p-1] \cdot 2^{-\delta_{lim}}, 4) & if\ s.eff = 0 \\ mod(2{\cdot}FL[0:p-1] + (4 - 2^{-w} - FS[0:p-1] \cdot 2^{-\delta_{lim}+1}), 4) & \text{otherwise} \end{cases} \\ &= \begin{cases} mod(FL + FS \cdot 2^{-\delta_{lim}}, 4) & if\ s.eff = 0 \\ mod(2{\cdot}(FL - FS \cdot 2^{-\delta_{lim}}) - 2^{-w}, 4) & \text{otherwise} \end{cases} \end{aligned} \tag{6.73}$$

We check the two cases of $s.eff$:

$\underline{s.eff = 0}$

$$FOPSUMV[-1:w] = mod(FL + FS \cdot 2^{-\delta_{lim}}, 4) \tag{6.74}$$

The significands and δ_{lim} are defined such that:

$$FL, FS \in [0, 2) \tag{6.75}$$

$$\delta_{lim} \geq 0 \tag{6.76}$$

therefore:

$$0 \leq FL + FS \cdot 2^{-\delta_{lim}} < 4 \tag{6.77}$$

and :

$$FOPSUMV[-1:w] = FL + FS \cdot 2^{-\delta_{lim}} \tag{6.78}$$

as required.

$\underline{s.eff = 1}$

$$FOPSUMV[-1:w] = mod(2 \cdot (FL - FS \cdot 2^{-\delta_{lim}}) - 2^{-w}, 4) \tag{6.79}$$

In case of effective substraction, the R-path is selected only if $(is\_r1 \; or \; is\_r2) = 1$, which is equivalent to $(is\_r1 \; or \; (\overline{is\_r1} \; and \; is\_r2)) = 1$.
if $is\_r1 = 1$ then $\delta \geq 2$, therefore one addend must be normal ($FL \in [1, 2)$) and the other can be normal or de-normal ($FS \in [0, 2)$). This leads to :

$$FOPSUM[-2:w] \in (1, 4) \tag{6.80}$$

if $(\overline{is\_r1} \; \& \; is\_r2) = 1$ then:

- $0 \leq \delta_{lim} \leq 1$ (from $\overline{is\_r1}$)
- $FL \cdot 2^{\delta_{lim}} - FS \geq 2$ (This is the specifications of is_r2 of the N-path which is valid when $is\_r1 = 0$)

Therefore, even if we assume that both addends can be normalized or de-normalized $(FS, FL \in [0, 2))$ $FOPSUM[-2:w]$ obeys :

$$2 \cdot (FL - FS \cdot 2^{-\delta_{lim}}) - 2^{-w} \in [2, 4) \tag{6.81}$$

and:

$$FOPSUMV[-1:w] = 2 \cdot (FL - FS \cdot 2^{-\delta_{lim}}) - 2^{-w} \tag{6.82}$$

as required.

Proof of equation (6.70)
¿From equation (78) , if $s.eff = 0$:

$$F'_{prenorm} \in (1, 4) \tag{6.83}$$

From equation (81) , if $s, eff = 1$:

$$F'_{prenorm} \in [2, 4) \tag{6.84}$$

which mean that $F'_{prenorm} \in [0, 4)$ as required.

6.3.2 Computation of $F_{far}[p-1]$ and the rounding decision

The calculation of the rounding decision (the signal INC) and $F_{far}[p-2]$ is divided into 2 stages:

1. Encapsulation of $FSOPA[p:w]$ to $< C[p-1], R', S' >$
2. Calculation of INC and $F_{far}[p-1]$

Encapsulation of $FSOPA[p:w]$ to $< C[p-1], R', S' >$ in effective addition

A graphical description is depicted in figure 6.4.

In case of effective addition:

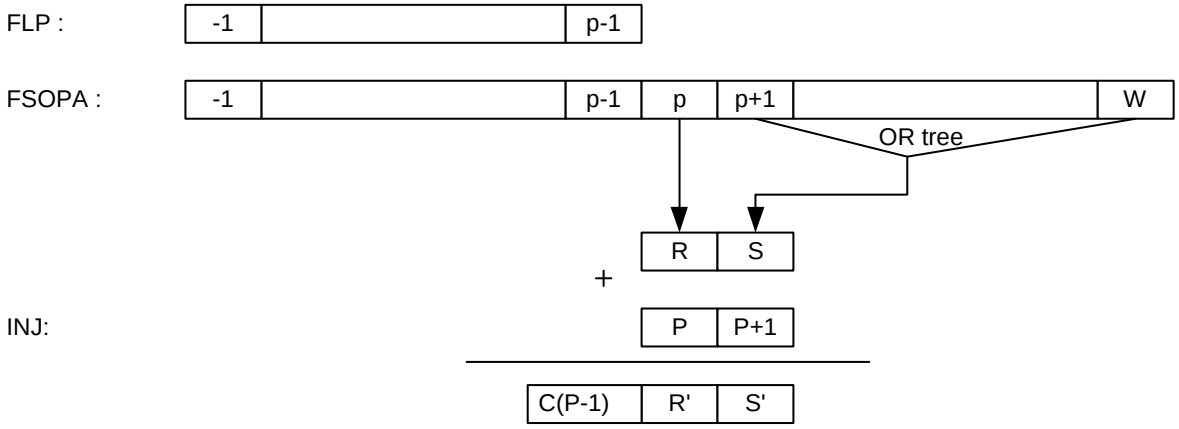

Figure 6.4: Effective addition in the second cycle of the R-path of the modified SE FP-adder

$FOPSUM[-1:w] = FLP[-1:p] + FSOPA[-1:w]$ so FOPSUM[p:w] = FSOPA[p:w] and the rounding and sticky bits are:

- $R = FSOPA[p]$,
- $S = OR(FSOPA[p+1:w])$.

The rounding algorithm is reduction to injection addition and rounding to zero, so injection that depends on the reduced rounding mode is added to produce $< C[p-1], R', S' >$ for the Calculation of INC and $F_{far}[p-1]$.

Encapsulation of $FSOPA[p:w]$ to $< C[p-1], R', S' >$ in effective substraction

A graphical description is depicted in figure 6.5 and simplification is depicted in figure 6.6.

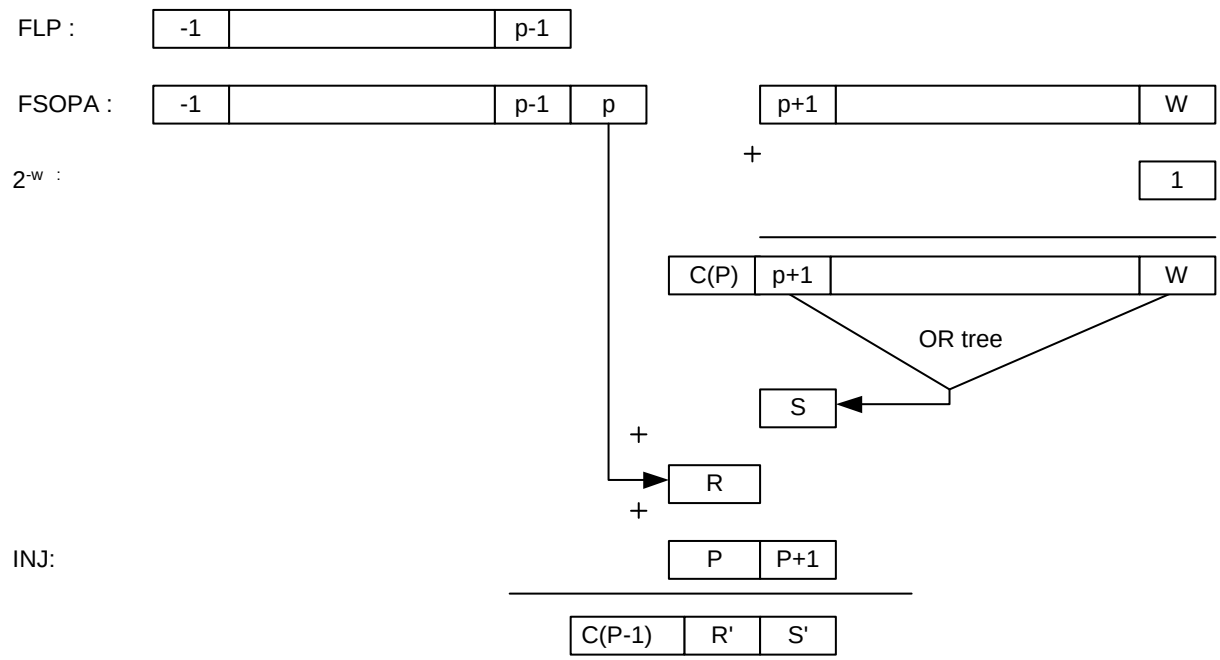

Figure 6.5: Effective substraction in the second cycle of the R-path of the modified SE FP-adder

In case of effective substraction
$FPSUM[-1:w] = FLP[-1:p] + FSOPA[-1:w] + 2^{-w}$, so
$FPSUM[p:w] = FSOPA[p:w] + 2^{-w}$ and the rounding and sticky bits are:

- $R = FSOPA[p]$,
- $S = OR(\overline{FSOPA[p+1:w]})$, and
- $C[p] = \overline{S}$.

The reason that $S = OR(\overline{FSOPA[p+1:w]})$ and $C[p] = \overline{S}$, is that in effective substraction the algorithm adds (virtually) a value of 2^{-w} to $FSOPA[-1:w]$ because of the ones complement representation. So, if $FSOPA[p:w] =$ "*all ones*", then the addition of 2^{-w} will cause $FOPSUM[p+1:w] =$ "*all zeros*" and a carry to bit $FOPSUM[p]$. The other option is that $FSOPA[p:w] \neq$ "*all ones*". In that case the addition of 2^{-w} will not propagate to bit $FOPSUM[p]$ so $FOPSUM[p+1:w] \neq$ "*all zeros*" and there is no carry to bit $FOPSUM[p]$.

6.4 N-path first cycle

The block receives the addends EA, FA, EB and FB (in unpacked format) and produces the intermediate values $FOPSUMI$,$\overline{FOPSUM}$ and LZP,under assumption that effective substraction is required ($SA \neq SB$). The first cycle of N-path is depicted in figure 6.7.

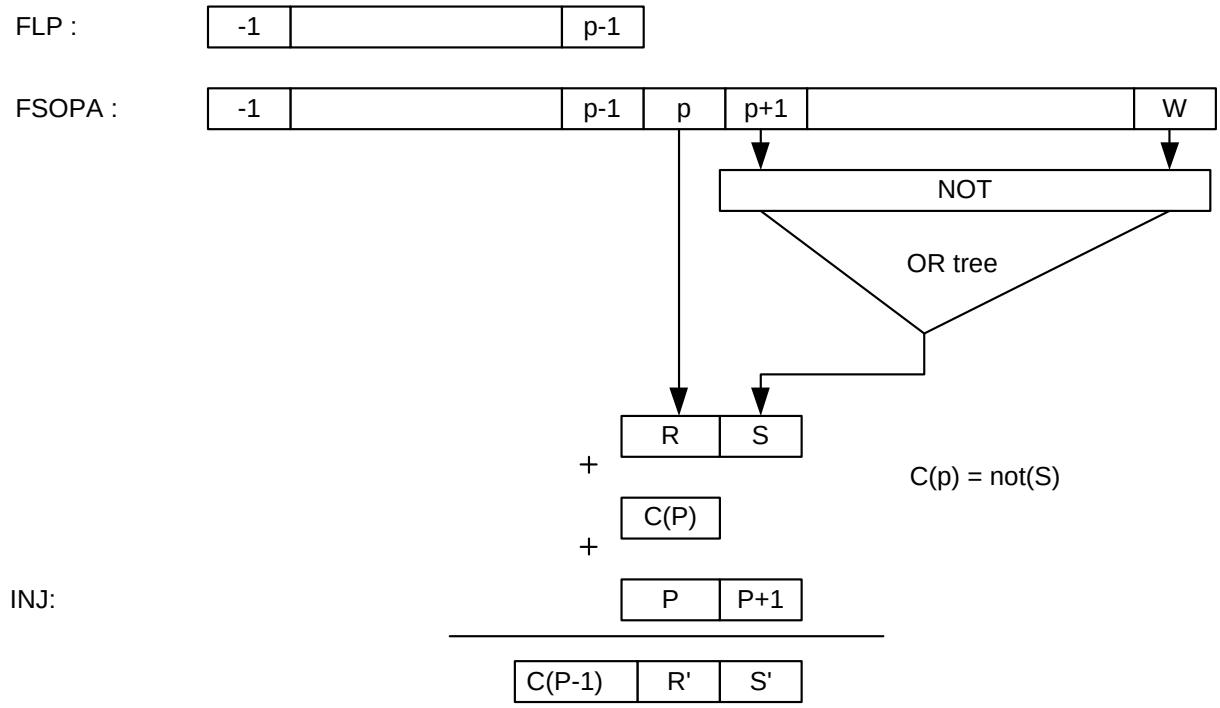

Figure 6.6: Simplified effective substraction in the second cycle of the R-path of the modified SE FP-adder

6.4.1 Specification

The specifications of the block are:

- $fopsumi = \begin{cases} |f'_{prenorm}| & if \quad f'_{prenorm} > 0 \\ don'tcare & \text{otherwise} \end{cases}$
- $\overline{fopsum} = \begin{cases} |f'_{prenorm}| & if \quad f'_{prenorm} \leq 0 \\ don'tcare & \text{otherwise} \end{cases}$
- $fopsumi[-2] = (f'_{prenorm} < 0)$
- $\overline{fopsum}[-2] = (f'_{prenorm} > 0)$
- $lzp \in \{min(EA-1, EB-1, \alpha), min(EA-1, EB-1, \alpha+1)\}$
- $min\_EA\_EB = \begin{cases} min(EA, EB) & if \quad \delta \leq 1 \\ don'tcare & \text{otherwise} \end{cases}$
- $min\_EA\_EB\_p1 = \begin{cases} min(EA, EB)+1 & if \quad \delta \leq 1 \\ don'tcare & \text{otherwise} \end{cases}$

where:

- $\delta = \begin{cases} EA - EB & if \ EA - EB \leq 1 \\ don'tcare & \text{otherwise} \end{cases}$
  (δ representation in the module is two's complement)
- $f'_{prenorm} = fl \cdot 2^{|\delta|} - fs$
- α is the number of leading zeros in $|f'_{prenorm}[0 : p-1]|$

6.4.2 Correctness

We prove the correctness of the module (namely, that the module depicted in figure 6.7 computes the correct outputs values) by the following stages:

1. Proving the correctness of δ.
2. Expressing $FOPSUM$ and $FOPSUMI$ as a function of $f'_{prenorm}$.
3. Proving the correctness of $fopsumi, \overline{fopsum}, fopsumi[-2]$ *and* $\overline{fopsum[-2]}$.

The correctness of min_EA_EB and $min\_EA\_EB\_p1$ can be verified immediately from the module diagram, and the correctness of LZP is explained in detail in section "The modifications of SE FP-Adder".

stage 1: Correctness of δ

For the calculation of δ, only bits [1:0] of EA and EB are used, and Δ (two's complement representation of δ) includes only bits [1:0]. Therefore, the equation of Δ is:

$$\begin{aligned}\Delta &= mod[EA[1:0] + \overline{EB[1:0]} + 1, 4] \\ &= mod[mod(EA, 4) + 3 - mod(EB, 4) + 1, 4] \\ &= mod[mod(EA, 4) - mod(EB, 4), 4] \\ &= mod(EA - EB, 4)\end{aligned} \tag{6.85}$$

if ($-2 \leq EA - EB \leq 1$) then $mod(EA - EB, 4)$ is the two's complement representation of δ as required.

stage 2: Expressing $FOPSUM$ and $FOPSUM$ as a function of $f'_{prenorm}$

The values of FLP and $FSOPA$ (see equations 169-170 on page 55) are:

$$FLP = FL{\cdot}2^{|\delta|} \tag{6.86}$$

$$FSOPA = 4 - 2^{-(p-1)} - FS \tag{6.87}$$

The inputs of the parallel prefix adder:

$$PPC\_A =< 0, FLP[-1 : p-1] > \quad (PPC\_A = FLP) \tag{6.88}$$

$$PPC\_B =< 1, FSOPA[-1 : p-1] > \quad (PPC\_B = 4 + FSOPA) \tag{6.89}$$

The function of the compound adder:

$$FOPSUM = PPC\_A + PPC\_B \tag{6.90}$$

$$FOPSUMI = FOPSUM + 2^{-(p-1)} \tag{6.91}$$

¿From equations (86-91):

$$\begin{aligned}FOPSUM &= FLP + FSOPA + 4 \\ &= FL{\cdot}2^{|\delta|} + 4 - 2^{-(p-1)} - FS + 4 \\ &= 8 - 2^{-(p-1)} + (FL{\cdot}2^{|\delta|} - FS)\end{aligned} \tag{6.92}$$

From equation (92) and the definition of $f'_{prenorm}$:

$$FOPSUM = 8 - 2^{-(p-1)} + f'_{prenorm} \tag{6.93}$$

$$FOPSUMI = 8 + f'_{prenorm} \tag{6.94}$$

stage 3:The correctness of $fopsumi, \overline{fopsum}, fopsumi[-2]$ and $\overline{fopsum[-2]}$

The equation of $FOPSUM[-2]$ is:

$$\begin{aligned}FOPSUM[-2] &= (FOPSUM \geq 4)\ and\ (FOPSUM < 8) \\ &= (FOPSUM < 8) \\ &= (8 - 2^{-(p-1)} + f'_{prenorm} < 8) \quad \text{(equation 6.93)} \\ &= (f'_{prenorm} < 2^{-(p-1)}) \\ &= (f'_{prenorm} \leq 0) \quad \text{(the l.s.b index of } F'_{prenorm} \text{ is [p-1])}\end{aligned} \tag{6.95}$$

and :

$$\overline{FOPSUM[-2]} \quad = (f'_{prenorm} > 0) \tag{6.96}$$

as required.

The equation of $FOPSUMI[-2]$ is:

$$\begin{aligned} FOPSUMI[-2] &= (FOPSUMI \geq 4) \; and \; (FOPSUMI < 8) \\ &= (FOPSUMI < 8) \\ &= (8 + f'_{prenorm} < 8) \\ &= (f'_{prenorm} < 0) \end{aligned} \tag{6.97}$$

as required.

$\overline{fopsum}$ is defined only for $f'_{prenorm} \leq 0$, therefore we prove it correctness for $f'_{prenorm} \leq 0$. According to equation (6.95), if $f'_{prenorm} \leq 0$:

$$FOPSUM[-2] = 1 \tag{6.98}$$

Therefore:

$$\begin{aligned} \overline{FOPSUM[-1:p-1]} &= \overline{\overline{FOPSUM[-2:p-1]}} \\ &= \overline{mod(FOPSUM, 8)} \\ &= 8 - 2^{-(p-1)} - mod(FOPSUM, 8) \\ &= 8 - 2^{-(p-1)} - mod(8 - 2^{-(p-1)} + f'_{prenorm}, 8) \quad \text{(equation 6.93)} \\ &= 8 - 2^{-(p-1)} - (8 - 2^{-(p-1)} + f'_{prenorm}) \quad (f'_{prenorm} \leq 0) \\ &= -f'_{prenorm} \\ &= |f'_{prenorm}| \end{aligned} \tag{6.99}$$

as required.

$fopsumi$ is defined only for $f'_{prenorm} > 0$, therefore we prove its correctness for $f'_{prenorm} > 0$. According to equation (6.97), if $f'_{prenorm} > 0$:

$$FOPSUMI[-2] = 0 \tag{6.100}$$

Therefore:

$$\begin{aligned} FOPSUMI[-1:p-1] &= FOPSUMI[-2:p-1] \\ &= mod(FOPSUMI, 8) \\ &= mod(8 + f'_{prenorm}, 8) \quad \text{(equation 6.94)} \\ &= f'_{prenorm} \\ &= |f'_{prenorm}| \end{aligned} \tag{6.101}$$

as required.

6.5 N-path second cycle

The block receives the sign of addend A (SA), and the intermediate values: $\overline{FOPSUM}, FOPSUMI, min\_EA\_EB, min\_EA\_EB\_p1$ and LZP, and produces the addition result $(S,E,F)_{near}$, and the signal is_r2, which is used to the path selection. The module is depicted in figure 6.8.

6.5.1 Specification

- $(S,E,F)_{near}$ are the unpacked format of the result in case of near path selection.
- $is\_r2 = (|f'_{prenorm}| \geq 2)$

where:

- $f'_{prenorm} = fl{\cdot}2^{|\delta|} - fs$

6.5.2 Correctness

The correctness $(S,E,F)_{near}$ is proved in section "The modifications of SE FP adder". We prove in this section the correctness of is_r2 by proving that:

$$ABS\_FOPSUM = |f'_{prenorm}| \tag{6.102}$$

Since $is\_r2 = ABS\_FOPSUM[-1]$, it is obvious that: $is\_r2 = (|f'_{prenorm}| \geq 2)$
as required.

Correctness of $ABS\_FPSUM[-1:p-1] = |f'_{prenorm}|$

The equation of $ABS\_FPSUM[-1:p-1]$ is:

$$ABS\_FPSUM[-1:p-1] = \begin{cases} FOPSUMI[-1:p-1] & if\ FOPSUM[-2] = 1 \\ \overline{FOPSUM[-1:p-1]} & \text{otherwise} \end{cases} \tag{6.103}$$

According to equation (6.95) of the proof of the first cycle of N-path:

$$\overline{FOPSUM[-2]} = (f'_{prenorm} > 0) \tag{6.104}$$

if $f'_{prenorm} \leq 0$ then

$$\overline{FOPSUM[-1:p-1]} = |f'_{prenorm}| \tag{6.105}$$

if $f'_{prenorm} > 0$ then

$$FOPSUMI[-1:p-1] = |f'_{prenorm}| \tag{6.106}$$

Therefore:

$$\begin{aligned} ABS\_FPSUM[-1:p-1] &= \begin{cases} |f'_{prenorm}| & if\ f'_{prenorm} > 0 \\ |f'_{prenorm}| & if\ f'_{prenorm} \leq 0 \end{cases} \\ &= |f'_{prenorm}| \end{aligned} \tag{6.107}$$

as required.

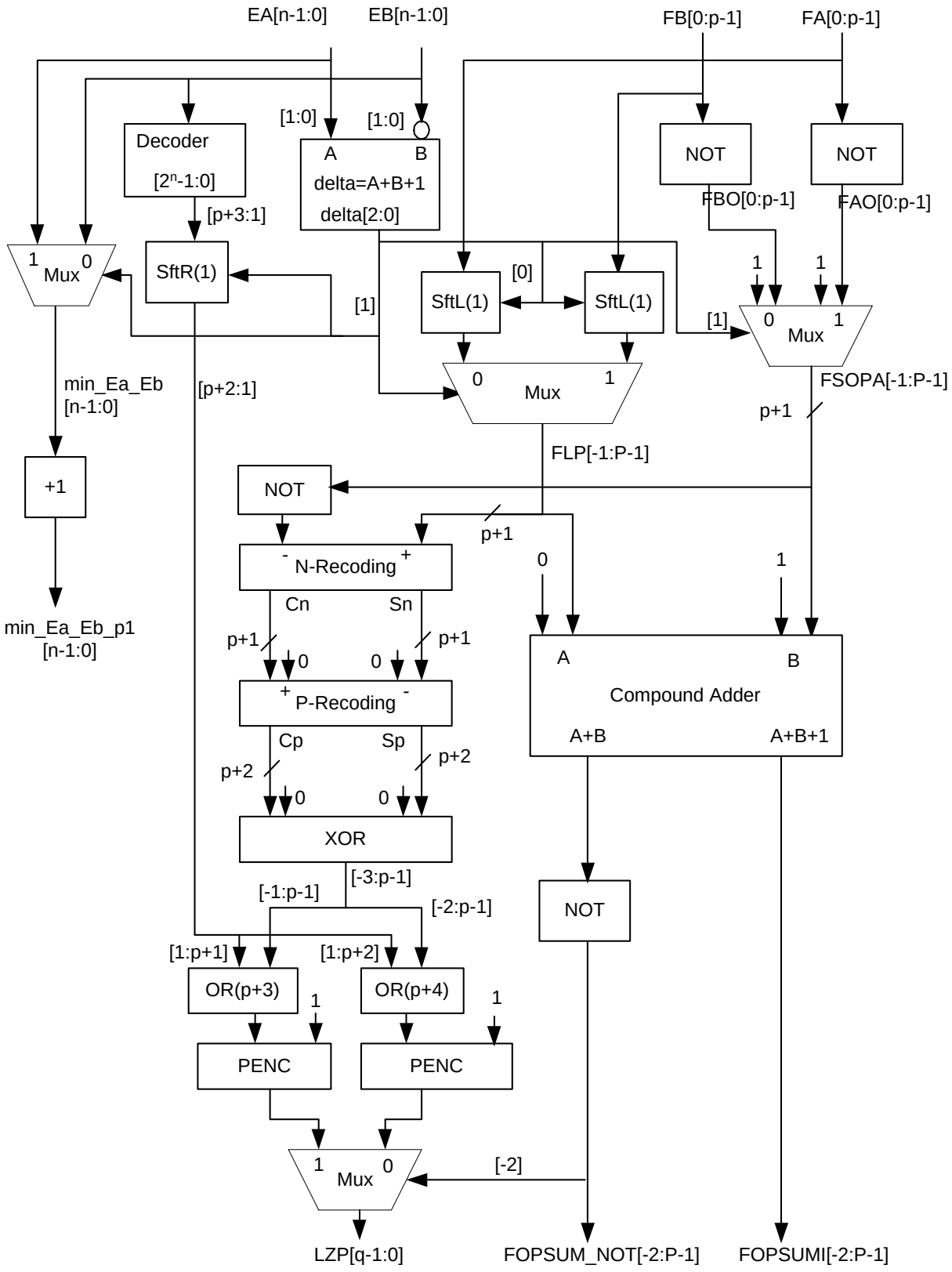

Figure 6.7: The first cycle of the N-Path of the modified SE FP-Adder

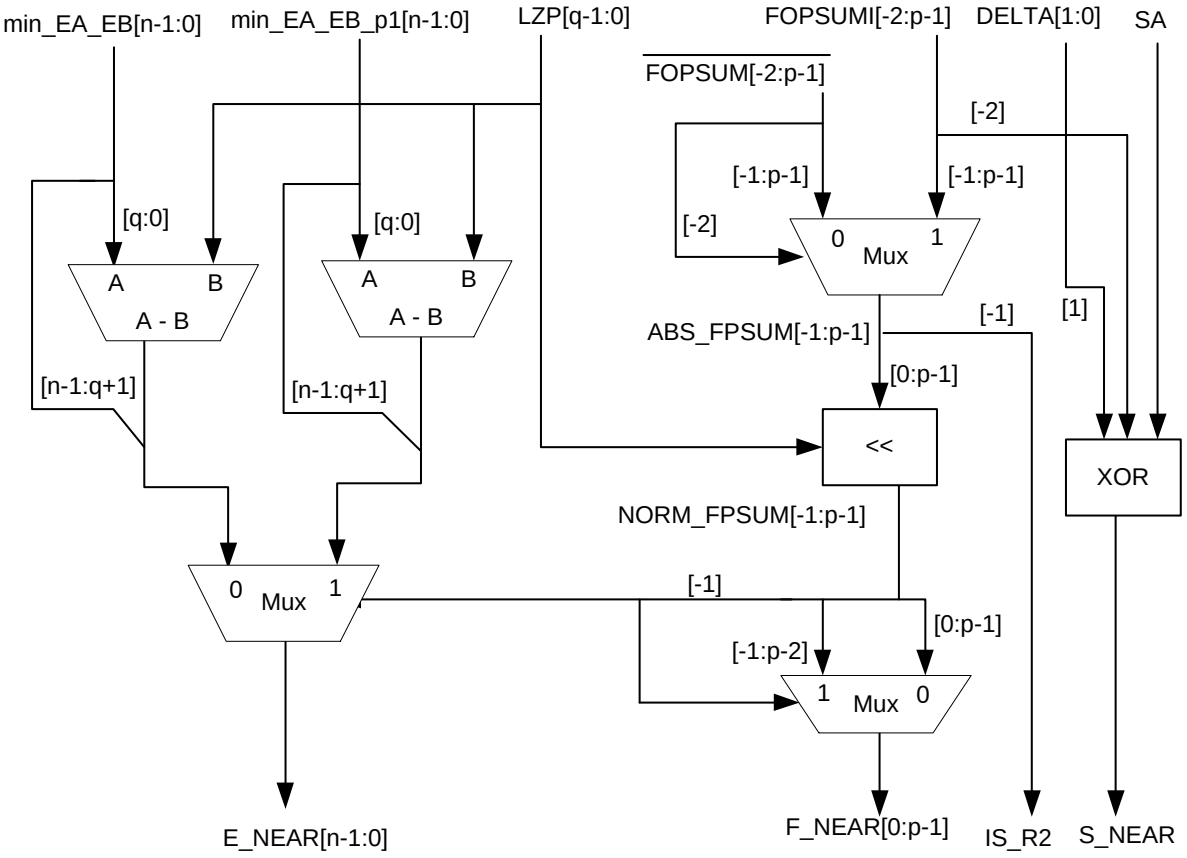

Figure 6.8: The second cycle of the N-Path of the modified SE FP-Adder

Chapter 7

Extending The SE FP-Adder To Support De-normals

In this section we focus on the modifications we did to the SE-FP adder in order to :

1. Support de-normalized numbers,
2. Compute the sign and the exponent of the result and not only the significand.

Supporting de-normal numbers requires enhancement of the control of the normalizing shifter in the second cycle of the N-path. In the first subsection we describe the problem of the SE FP adder and in the following subsections we present two options to solve this problem. For each option we give detailed description of the computation of the significand and the exponent, including a correctness proof. In addition, we analyze the timing (in terms of logic levels) of both options and find that option (1) is better.
The computation of the sign do not exist in the SE FP adder, and therefore, we added in to the algorithm.
Finally, we present the modification of computing is_r2.

7.1 The leading zeros prediction in the SE FP-adder

The values $ABS\_FPSUM[0:p-1]$ and min_EA_EB respect the equation:

$$ABS\_FPSUM[0:p-1]\cdot 2^{min\_ea\_eb} = |A+B| \tag{7.1}$$

We prove equation (7.1): If $ABS\_FPSUM[-1] = 1$ then R-path is selected and the context is the N-path,so we can assume that $ABS\_FPSUM[-1] = 0$. Based on equation (107) in sub-section "N-path second cycle":

$$ABS\_FPSUM[0:p-1] = |f'_{prenorm}| \tag{7.2}$$

We defined:

$$f'_{prenorm} = fl\cdot 2^{|\delta|} - fs \tag{7.3}$$

From equations (7.2-7.3):

$$\begin{aligned} ABS\_FPSUM[0:p-1]\cdot 2^{min\_ea\_eb} &= |f'_{prenorm}|\cdot 2^{min\_ea\_eb} \\ &= |fl\cdot 2^{|\delta|} - fs|\cdot 2^{min\_ea\_eb} \\ &= |fl\cdot 2^{|\delta|+min\_ea\_eb} - fs\cdot 2^{min\_ea\_eb}| \\ &= |fl\cdot 2^{el} - fs\cdot 2^{es}| \\ &= |A+B| \end{aligned} \tag{7.4}$$

as required.

According to equation (6.100), all we have to do to calculate $(E,F)_{near}$ is:

- Make the appropriate shift to $ABS\_FPSUM[0:p-1]$ to be IEEE compliment.
- Compensate the shift of ABS_FPSUM by decreasing min_ea_eb properly.

Practically, we have to find how many leading zeros are in $ABS\_FPSUM[0:p-1]$ meaning that if:

$$ABS\_FPSUM[0:p-1] \in < 0^{\alpha}, 1, \{0,1\}^{p-\alpha-1} > \text{ where } \alpha \in \{0,1,2,...,p\} \tag{7.5}$$

then α is the number of leading zeros in $ABS\_FPSUM[0:p-1]$.
The required shift left is given by:

$$required\ shift\ left = min\{\alpha, min\_EA\_EB - 1\} \tag{7.6}$$

because we want to make the maximum shift left of $ABS\_FPSUM[0:p-1]$ but we are limited by min_EA_EB-1, since:

$$E_{near} = min\_EA\_EB - (required\ shift\ left) \tag{7.7}$$

and $E_{near} \geq 1$ (in its un-packed format).

The estimation:

$$LZP' \in \{\alpha, \alpha+1\} \tag{7.8}$$

can be achieved by the design depicted in in figure 7.1, which is in use in the SE FP-adder: (Details about this design are in references [SE] and [DEM]). The uncertainty in LZP' is corrected in the second cycle of N-path by a conditional shifter that makes one shift right if $LZP' = \alpha + 1$. But the problem of using LZP' as a control to the normalizing shifter, as done in SE FP-Adder, is that it ignores restriction (6), and therefore will yield a mistaken results if the expected result is de-normal.
We present two approaches to add restriction (6) into the FP-adder with minimal influence on the timing.

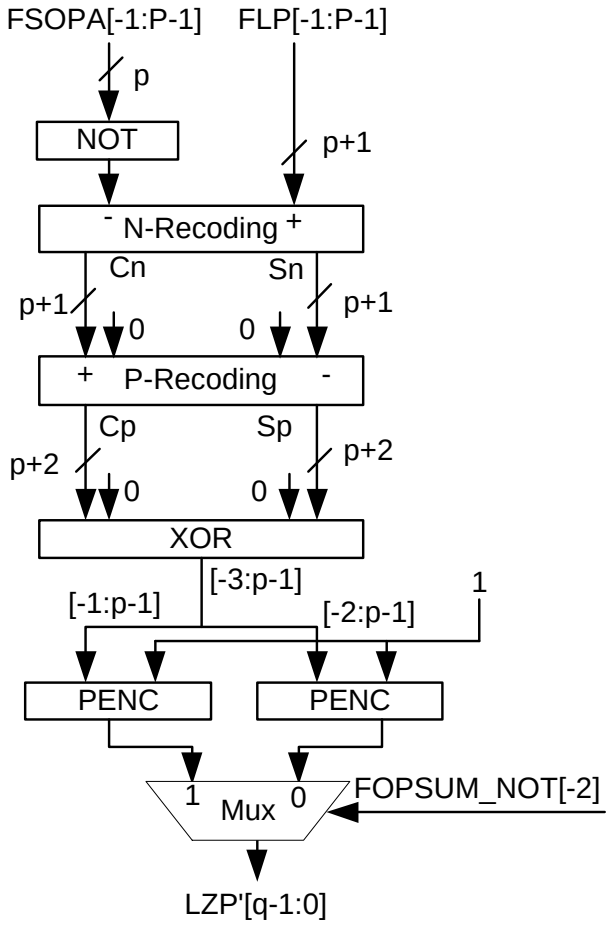

Figure 7.1: Leading zeros estimation in the SE FP-Adder

7.2 Option (1):Leading zeros prediction by modifying the first cycle

In order to support both normal and de-normal results we limit LZP' as depicted in figure 7.2.

$$LZP = min\{EA-1, EB-1, LZP'\} \tag{7.9}$$

We proved that equation (1).
The function of the decoder:

$$EB\_decoded[2^n-1:0] =< 0^{2^n-1-EB}, 1, 0^{EB} > \quad 1 \leq EB \leq 2^n-1 \tag{7.10}$$

if $EB > p{+}3$ then $EB > p{+}2$ (since in the N-path $|EA-EB| \leq 1$) and $min\_EA\_EB > p+2$. Its obvious that $\alpha \leq p$, therefore if $EB > p+3$ then $\alpha \leq p \leq min\{EA-1, EB-1\}$ and $LZP = min(min\_EA\_EB-1, LZP') = LZP'$.

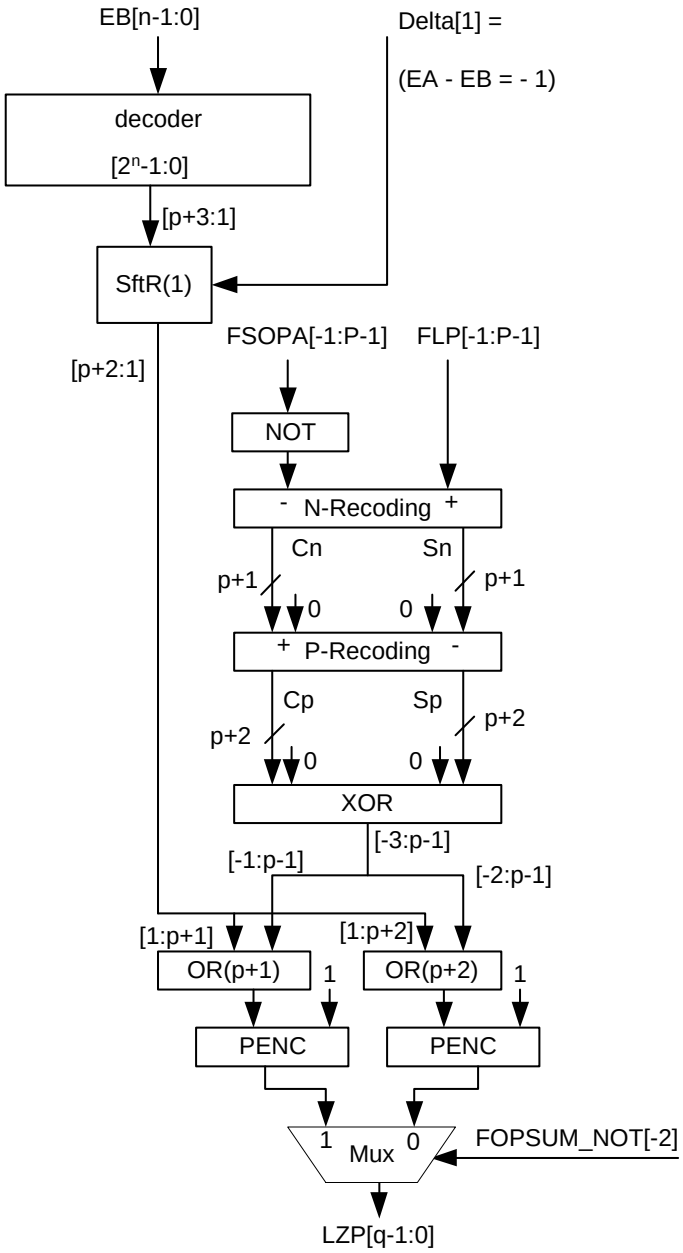

Figure 7.2: Leading zeros estimation in the modified SE FP-Adder

On the other hand, $EB \geq 1$, and therefore bit [0] of the decoder output is always zero. As a result, we are interested only in bits [p+3:1] of the decoder output:

$$EB\_decoded[p+3:1] =< 0^{p+3-EB}, 1, 0^{EB-1} > \quad 1 \leq EB \leq p+3 \tag{7.11}$$

The output of the conditional shifter (that is controlled by $\Delta[1]$):

$$\begin{aligned} min\_EA\_EB\_decoded[p+2:1] &= \begin{cases} <0^{p+3-EB},1,0^{EB-2}> & if\ EA-EB=-1 \\ <0^{p+2-EB},1,0^{EB-1}> & if\ EA-EB\in\{0,1\} \\ don'tcare & \text{otherwise} \end{cases} \\ &= \begin{cases} <0^{p+2-EA},1,0^{EA-1}> & if\ EA-EB=-1 \\ <0^{p+2-EB},1,0^{EB-1}> & if\ EA-EB\in\{0,1\} \\ don'tcare & \text{otherwise} \end{cases} \\ &=<0^{p+2-min(EA,EB)},1,0^{min(EA,EB)-1}> \end{aligned} \tag{7.12}$$

Flipping the order of the bits:

$$min\_EA\_EB\_decoded[1:p+2] =<0^{min(EA,EB)-1},1,0^{p+2-min(EA,EB)}> \tag{7.13}$$

$$min\_EA\_EB\_decoded[1:p+1] =<0^{min(EA,EB)-1},1,0^{p+1-min(EA,EB)}> \tag{7.14}$$

The inputs to both priority encoders are a result of bitwise or with $min\_EA\_EB\_decoded[1:p+2]$ and $min\_EA\_EB\_decoded[1:p+1]$ which have "1" in the bit k from the left where:
$k = min(EA, EB) - 1$. Therefore, the priority encoder outputs are limited to $min(EA, EB) - 1$ as required.

7.3 Computation of $(E, F)_{near}$ in option (1)

Option (1) was selected by us for the modified SE FP-adder, and we modified the N-path appropriately as depicted in diagrams 6.7 and 6.8 in section "Detailed description of the modified SE FP-Adder".
In this section we prove the correctness of $(e, f)_{near}$ in option (1), which is actually a proof of the correctness of the second cycle of the N-path in the modified SE FP-adder.
We proved in the previous section that the input LZP to the second cycle of N-path in option (1) equals:

$$LZP = min\{EA-1, EB-1, LZP'\} \tag{7.15}$$

where α is the number of leading zeros in $ABS\_FPSUM[0:p-1]$:

$$ABS\_FPSUM[0:p-1]\ \in <0^{\alpha},1,\{0,1\}^{p-1-\alpha}> \tag{7.16}$$

and LZP' is an estimation of α:

$$LZP' \in \{\alpha, \alpha+1\} \tag{7.17}$$

The relevant equations to F_{far} and E_{far} in the second cycle of N-path are:

$$NORM\_FPSUM[-1:p-1] =<0, ABS\_FPSUM[0:p-1]> \cdot 2^{LZP} \tag{7.18}$$

$$F_{near}[0:p-1] = \begin{cases} NORM\_FPSUM[0:p-1] & if\ NORM\_FPSUM[-1]=0 \\ NORM\_FPSUM[-1:p-2] & otherwise \end{cases} \tag{7.19}$$

$$E_{near} = \begin{cases} min\{EA,EB\}-LZP & if\ NORM\_FPSUM[-1]=0 \\ min\{EA,EB\}-LZP+1 & otherwise \end{cases} \tag{7.20}$$

We will check all the possible cases:

1. $LZP'=\alpha$ and $LZP'\leq min\{EA-1,EB-1\}$
2. $LZP'=\alpha$ and $LZP'>min\{EA-1,EB-1\}$
3. $LZP'=\alpha+1$ and $LZP'\leq min\{EA-1,EB-1\}$
4. $LZP'=\alpha+1$ and $LZP'>min\{EA-1,EB-1\}$

Case 1:$LZP'=\alpha$ and $LZP'\leq min\{EA-1,EB-1\}$
According to equation (7.15), in this case:

$$LZP = LZP' = \alpha \tag{7.21}$$

$$LZP \leq min\{EA,EB\}-1 \tag{7.22}$$

Therefore, according to equations (7.16) and (7.18):

$$\begin{aligned} MORM\_FPSUM[-1:p-1] &\in <0^{\alpha+1},1,\{0,1\}^{p-1-\alpha}>\cdot 2^{\alpha} \\ &= <0,1,\{0,1\}^{p-1-\alpha},0^{\alpha}> \\ &\in <0,1,\{0,1\}^{p-1}> \end{aligned} \tag{7.23}$$

Since $MORM\_FPSUM[-1]=0$, according to equations (7.19-7.20):

$$\begin{aligned} F_{near}[0:p-1] &= MORM\_FPSUM[0:p-1] \\ &\in <1,\{0,1\}^{p-1}> \end{aligned} \tag{7.24}$$

$$\begin{aligned} E_{near} &= min\{EA,EB\}-LZP \geq 1 \\ &\geq 1 \qquad\qquad\qquad\qquad (\text{equation } 7.22) \end{aligned} \tag{7.25}$$

which means that $f_{near}\in[1,2)$ and $e_{near}\geq e_{min}$ and therefore, they represent a normal, IEEE compliant, number.
We still have to prove that $f_{near}\cdot 2^{e_{near}} = |A+B|$: According to equations (7.18-7.20):

$$\begin{aligned} F_{near}\cdot 2^{E_{near}} &= MORM\_FPSUM\cdot 2^{min\{EA,EB\}-LZP} && (\text{equations } 7.19\text{-}7.20) \\ &= (ABS\_FPSUM\cdot 2^{LZP})\cdot 2^{min\{EA,EB\}-LZP} && (\text{equation } 7.18) \\ &= ABS\_FPSUM\cdot 2^{min\{EA,EB\}} \end{aligned} \tag{7.26}$$

We proved in section "N-path second cycle" that $ABS\_FPSUM = |F'_{prenorm}|$, therefore:

$$F_{near}\cdot 2^{E_{near}} = |F'_{prenorm}|\cdot 2^{min\{EA,EB\}} \tag{7.27}$$

In real values:

$$\begin{aligned} f_{near}\cdot 2^{f_{near}} &= |f'_{prenorm}|\cdot 2^{min\{ea,eb\}} \\ &= |fl\cdot 2^{|\delta|} - fs|\cdot 2^{min\{ea,eb\}} \\ &= |fl\cdot 2^{el} - fs\cdot 2^{es}| \\ &= |A+B| \end{aligned} \tag{7.28}$$

as required.

Case 2:$LZP' = \alpha$ and $LZP' > min\{EA-1, EB-1\}$
According to equation (7.15), in this case:

$$LZP = min\{EA-1, EB-1\} \tag{7.29}$$

$$\alpha > min\{EA, EB\} - 1 \tag{7.30}$$

Therefore, according to equations (7.16) and (7.18):

$$\begin{aligned} MORM\_FPSUM[-1:p-1] &\in < 0^{\alpha+1}, 1, \{0,1\}^{p-1-\alpha} > \cdot 2^{min\{EA,EB\}-1} \\ &= < 0^{\alpha+1-(min\{EA,EB\}-1)}, 1, \{0,1\}^{p-1-\alpha}, 0^{min\{EA,EB\}-1} > \\ &\in < 0^2, \{0,1\}^{p-1} > \end{aligned} \tag{7.31}$$

Since $MORM\_FPSUM[-1] = 0$, according to equations (7.19-7.20):

$$\begin{aligned} F_{near}[0:p-1] &= MORM\_FPSUM[0:p-1] \\ &\in < 0, \{0,1\}^{p-1} > \end{aligned} \tag{7.32}$$

$$\begin{aligned} E_{near} &= min\{EA, EB\} - LZP \\ &= min\{EA, EB\} - (min\{EA, EB\} - 1) \\ &= 1 \end{aligned} \tag{7.33}$$

which means that $f_{near} \in [0,1)$ and $e_{near} = e_{min}$ and therefore, they represent a de-normal, IEEE compliant, number.
The proof that $f_{near}\cdot 2^{e_{near}} = |A+B|$ is identical to case (1) (equations 7.26-7.28).

Case 3:$LZP' = \alpha + 1$ and $LZP' \leq min\{EA-1, EB-1\}$
According to equation (7.15), in this case:

$$LZP = LZP' = \alpha + 1 \tag{7.34}$$

$$LZP \leq min\{EA, EB\} - 1 \tag{7.35}$$

Therefore, according to equations (7.16) and (7.18):

$$\begin{aligned} MORM\_FPSUM[-1:p-1] &\in < 0^{\alpha+1}, 1, \{0,1\}^{p-1-\alpha} > \cdot 2^{\alpha+1} \\ &= < 1, \{0,1\}^{p-1-\alpha}, 0^{\alpha+1} > \\ &\in < 1, \{0,1\}^{p-1}, 0 > \end{aligned} \tag{7.36}$$

Since $MORM\_FPSUM[-1] = 1$, according to equations (7.19-7.20):

$$\begin{aligned} F_{near}[0:p-1] &= MORM\_FPSUM[-1:p-2] \\ &\in < 1, \{0,1\}^{p-1} > \end{aligned} \tag{7.37}$$

$$E_{near} = min\{EA, EB\} - LZP + 1 \geq 1 + 1 > 1 \tag{7.38}$$

which means that $f_{near} \in [1, 2)$ and $e_{near} > e_{min}$ and therefore, they represent a normal, IEEE compliant, number.
We still have to prove that $f_{near} \cdot 2^{e_{near}} = |A + B|$: According to equations (7.18-7.20):

$$\begin{aligned} F_{near} \cdot 2^{E_{near}} &= MORM\_FPSUM \cdot 2^{-1} \cdot 2^{min\{EA,EB\}-LZP+1} && \text{(equations 7.19-7.20)} \\ &= (ABS\_FPSUM \cdot 2^{LZP}) \cdot 2^{min\{EA,EB\}-LZP} && \text{(equation 7.18)} \\ &= ABS\_FPSUM \cdot 2^{min\{EA,EB\}} \end{aligned} \tag{7.39}$$

The rest of the proof is identical to case (1) (equations 7.27-7.28).

Case 4:$LZP' = \alpha + 1$ and $LZP' > min\{EA - 1, EB - 1\}$
According to equation (7.15), in this case:

$$LZP = min\{EA - 1, EB - 1\} \tag{7.40}$$

$$\alpha + 1 > min\{EA, EB\} - 1 \tag{7.41}$$

Therefore, according to equations (7.16) and (7.18):

$$\begin{aligned} MORM\_FPSUM[-1 : p-1] &\in < 0^{\alpha+1}, 1, \{0,1\}^{p-1-\alpha} > \cdot 2^{min\{EA,EB\}-1} \\ &= < 0^{\alpha+1-(min\{EA,EB\}-1)}, 1, \{0,1\}^{p-1-\alpha}, 0^{min\{EA,EB\}-1} > \\ &\in < 0, \{0,1\}^{p} > \end{aligned} \tag{7.42}$$

Since $MORM\_FPSUM[-1] = 0$, according to equations (7.19-7.20):

$$\begin{aligned} F_{near}[0 : p-1] &= MORM\_FPSUM[0 : p-1] \\ &\in < \{0,1\}^{p} > \end{aligned} \tag{7.43}$$

$$\begin{aligned} E_{near} &= min\{EA, EB\} - LZP \\ &= min\{EA, EB\} - (min\{EA, EB\} - 1) \\ &= 1 \end{aligned} \tag{7.44}$$

which means that $f_{near} \in [0, 2)$ and $e_{near} = e_{min}$. Therefore, they can represent de-normalized numbers or (part of the) normalized numbers. In both cases $(e, f)_{near}$ are IEEE compliant since if $e_{near} = e_{min}$ then there are no limitations on f_{near} (can be any value in $[0, 2)$).
The proof that $f_{near} \cdot 2^{e_{near}} = |A + B|$ is identical to case (1) (equations 7.26-7.28).

7.4 Option (2): Leading zeros prediction by modifying the second cycle

In this section we present another approach for supporting de-normalized number. We use the first cycle of SE FP adder without changing its leading zeros estimation, so the output of the first cycle of the N-path is LZP' ($LZP' \in \{\alpha, \alpha + 1\}$ where α is the number of leading zeros in $ABS\_FPSUM[0 : p-1]$). The limitation of LZP' to $min\{EA-1, EB-1\}$ is done

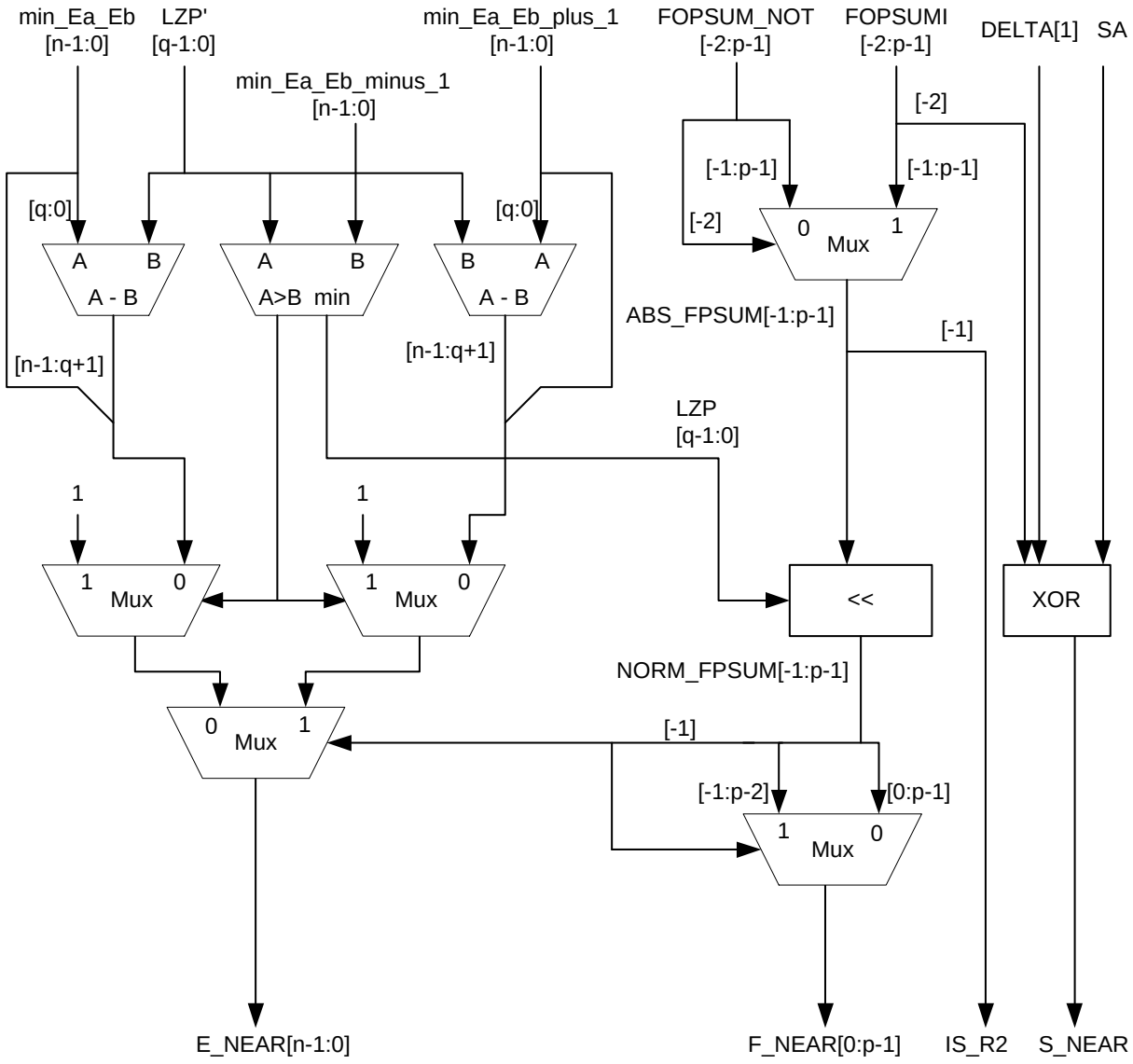

Figure 7.3: The second cycle of N-path in option (2).

by a "minimum" module in the second cycle of the N-path as depicted in figure 7.3. Its obvious that:

$$LZP = min\{EA - 1, EB - 1, LZP'\} \tag{7.45}$$

and all the outputs except E_{near} are identical to option (1). In order to save time, E_{near} in this option is calculated differently then in option (1). In option (1) E_{near} was one of 2 subtractors outputs that were feed by LZP, but LZP was without any delay and thus the subtractors delay is bearable in option (1). In option (2) LZP suffers from the delay of the "minimum" module and thus feeding it into additional substraction is unbearable.
We prove the correctness of E_{near} by verifying that it is equal to E_{near} in option (1) in all the cases:

1. $LZP' = \alpha$ and $LZP' \leq min\{EA-1, EB-1\}$
2. $LZP' = \alpha$ and $LZP' > min\{EA-1, EB-1\}$
3. $LZP' = \alpha+1$ and $LZP' \leq min\{EA-1, EB-1\}$
4. $LZP' = \alpha+1$ and $LZP' > min\{EA-1, EB-1\}$

The equation of E_{near} in option (2) is:

$$E_{near} = \begin{cases} min\{EA, EB\} - LZP' & \text{if } (NORM\_FPSUM[-1] = 0) \ \& \\ & (LZP' \leq min\{EA, EB\} - 1) \\ 1 & \text{if } (NORM\_FPSUM[-1] = 0) \ \& \\ & (LZP' > min\{EA, EB\} - 1) \\ min\{EA, EB\} - LZP' + 1 & \text{if } (NORM\_FPSUM[-1] = 1) \ \& \\ & (LZP' \leq min\{EA, EB\} - 1) \\ 1 & \text{if } (NORM\_FPSUM[-1] = 1) \ \& \\ & (LZP' > min\{EA, EB\} - 1) \end{cases} \tag{7.46}$$

Case 1:$LZP' = \alpha$ and $LZP' \leq min\{EA-1, EB-1\}$

According to equation (7.23) in option (1):

$$MORM\_FPSUM[-1] = 0 \tag{7.47}$$

In this case $LZP = LZP'$, therefore according to equations (7.46-7.47):

$$\begin{aligned} E_{near} &= min\{EA, EB\} - LZP' \\ &= min\{EA, EB\} - LZP \end{aligned} \tag{7.48}$$

as required according to equation (7.25) in option (1).

Case 2:$LZP' = \alpha$ and $LZP' > min\{EA-1, EB-1\}$

According to equation (7.46), in this case:

$$E_{near} = 1 \tag{7.49}$$

as required according to equation (7.33) in option (1).

Case 3:$LZP' = \alpha+1$ and $LZP' \leq min\{EA-1, EB-1\}$

According to equation (7.36) in option (1):

$$MORM\_FPSUM[-1] = 1 \tag{7.50}$$

In this case $LZP = LZP'$, therefore according to equations (7.46) and (7.50)

$$\begin{array}{rl} E_{near} & = min\{EA, EB\} - LZP' + 1 \\ & = min\{EA, EB\} - LZP + 1 \end{array} \tag{7.51}$$

as required according to equation (7.38) in option (1).

Case 4:$LZP' = \alpha + 1$ and LZP'>$min\{EA - 1, EB - 1\}$

According to equation (7.46), in this case:

$$E_{near} \quad = 1 \tag{7.52}$$

as required according to equation (7.44) in option (1).

7.5 Computation of S_{near} in the modified SE FP adder

The equation of S_{near}:

$$S_{near} = xor(SA, FOPSUMI[-2], \Delta[1]) \tag{7.53}$$

¿From the specifications of the first cycle of N-path:

$$FOPSUMI[-2] = f'_{prenorm} < 0 \tag{7.54}$$

and:

$$\Delta[1] = (\delta < 0) \tag{7.55}$$

¿From equations (160-162):

$$(-1)^{s_{near}} = (-1)^{sa} \cdot sign(f'_{prenorm}) \cdot sign(\delta) \tag{7.56}$$

where:

$$sign(x) = \begin{cases} 1 & if\ x \geq 0 \\ -1 & \text{otherwise} \end{cases} \tag{7.57}$$

We check two cases of δ:

1. $\delta \geq 0$, and
2. $\delta < 0$.

$\underline{\delta \geq 0}$
If $\delta \geq 0$ then $(fl, fs) = (fa, fb)$ and $sign(\delta) = 1$
therefore, according to equation (7.56):

$$\begin{aligned}(-1)^{s_{near}} &= (-1)^{sa} \cdot sign(f'_{prenorm}) \\ &= (-1)^{sa} \cdot sign(fl \cdot 2^{|\delta|} - fs) \\ &= (-1)^{sa} \cdot sign(fa \cdot 2^{ea} - fb \cdot 2^{eb}) \\ &= sign[(-1)^{sa} \cdot fa \cdot 2^{ea} - (-1)^{sa} \cdot fb \cdot 2^{eb}] \\ &= sign[(-1)^{sa} \cdot fa \cdot 2^{ea} + (-1)^{sb} \cdot fb \cdot 2^{eb}] \quad (SA = \overline{SB}) \\ &= sign(A + B)\end{aligned} \tag{7.58}$$

as required.

$\underline{\delta < 0}$
If $\delta < 0$ then $(fl, fs) = (fb, fa)$ and $sign(\delta) = -1$
therefore, according to equation (7.56):

$$\begin{aligned}(-1)^{s_{near}} &= -(-1)^{sa} \cdot sign(f'_{prenorm}) \\ &= -(-1)^{sa} \cdot sign(fl \cdot 2^{|\delta|} - fs) \\ &= -(-1)^{sa} \cdot sign(fb \cdot 2^{eb} - fa \cdot 2^{ea}) \\ &= (-1)^{sa} \cdot sign(fa \cdot 2^{ea} - fb \cdot 2^{eb}) \\ &= sign[(-1)^{sa} \cdot fa \cdot 2^{ea} - (-1)^{sa} \cdot fb \cdot 2^{eb}] \\ &= sign[(-1)^{sa} \cdot fa \cdot 2^{ea} + (-1)^{sb} \cdot fb \cdot 2^{eb}] \quad (SA = \overline{SB}) \\ &= sign(A + B)\end{aligned} \tag{7.59}$$

as required.

7.6 Modifications of the computation of $fsopa$ and flp in the N-path

In The first cycle of N-path the values $fsopa$ and flp are calculated. Their values in the SE FP adder are :

$$flp = 2 \cdot fl \tag{7.60}$$

$$fsopa = -2 \cdot fs \cdot 2^{-|\delta|} \tag{7.61}$$

This requires compensation of δ in the exponent computation. In order to simplify the exponent computation, we changed their values to:

$$flp = fl \cdot 2^{|\delta|} \tag{7.62}$$

$$fsopa = -fs \tag{7.63}$$

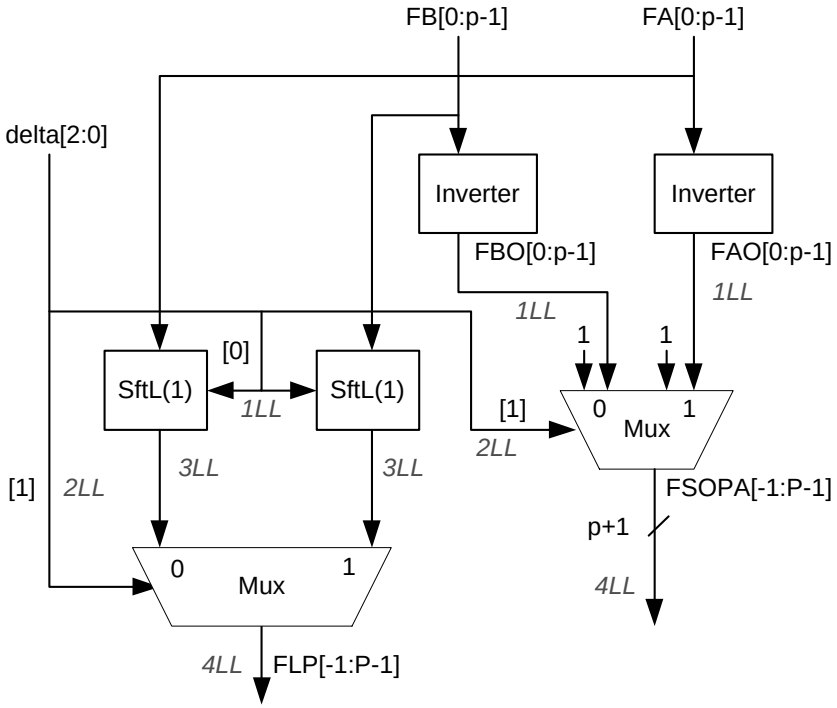

Figure 7.4: The computation of flp and $fsopa$ in the N-path of the modified SE FP adder.

The delays to calculate the modified $fsopa$ and flp are 4 logic levels, as specified in figure 7.4. We prove the correctness of $fsopa$ and flp as specified in (3) and (4).
The equations of FAO, FBO, FLP and $FSOPA$ are:

$$FAO[0:p-1] = \overline{FA[0:p-1]} \quad (FAO = 2 - 2^{-(p-1)} - FA) \tag{7.64}$$

$$FBO[0:p-1] = \overline{FB[0:p-1]} \quad (FBO = 2 - 2^{-(p-1)} - FB) \tag{7.65}$$

$$FSOPA[0:p-1] = \begin{cases} <1, FAO[0:p-1]> & if \quad \delta = -1 \\ <1, FBO[0:p-1]> & if \quad \delta \in \{0,1\} \end{cases} \tag{7.66}$$

$$FLP[-1:p-1] = \begin{cases} <0, FA[0:p-1]> & if \quad \delta = 0 \\ <FB[0:p-1], 0> & if \quad \delta = -1 \\ <FA[0:p-1], 0> & if \quad \delta = 1 \end{cases} \tag{7.67}$$

Equation (7.67) is equivalent to:

$$FLP = \begin{cases} FA & if\ \delta = 0 \\ 2{\cdot}FB & if\ \delta = -1 \\ 2{\cdot}FA & if\ \delta = 1 \end{cases} \tag{7.68}$$

The definitions of FS and FL are:

$$FL = \begin{cases} FA & if \quad \delta \geq 0 \\ FB & otherwise \end{cases} \tag{7.69}$$

$$FS = \begin{cases} FB & if \quad \delta \geq 0 \\ FA & otherwise \end{cases} \tag{7.70}$$

According to equations (7.64-7.70):

$$FLP = FL \cdot 2^{|\delta|} \tag{7.71}$$

$$FSOPA = 4 - 2^{-(p-1)} - FS \tag{7.72}$$

Which is equivalent to:

$$flp = fl \cdot 2^{|\delta|} \tag{7.73}$$

$$fsopa = -fs \tag{7.74}$$

as required.

7.7 Computation of is_r2

The computation of is_r2 in the SE FP adder is based on the values of flp and $fsopa$. Since we changed the computation of flp and $fsopa$, we had to change properly the computation of is_r2.
The expected value of is_r2 is: $is\_r2 = |f'_{prenorm}| \geq 2$. The correctness of our implementation is proved in subsection 5.5.

7.8 Timing Analysis of the modifications

The modifications that we did and are critical to the timing are:

1. The calculation of flp and $fsopa$ in the N-path,
2. In option (1):The computation of LZP output of the first cycle of N-path, and
3. In option (2):The insertion of the "minimum" module in the second cycle of the N-path for calculation of the normalizing shift.

The first modification do not change the timing, since $fsopa$ and flp in the N-path are achieved after 4 logic level both in SE FP-Adder and in the modified calculation, as can be seen in figure 7.4. In the modifications of option (1), we added one logic level (array of "or" gates) to the first cycle of N-path. Therefore, we increased the delay of the adder from 12 to 13 logic levels.
In order to analyze the delay of option (2), we suggest implementations to the normalizing shifter and the minimum module, so that the delay they add will be minimal.
Figures 7.4, 7.6 and 7.7 depict our suggestion for the implementations of the normalizing

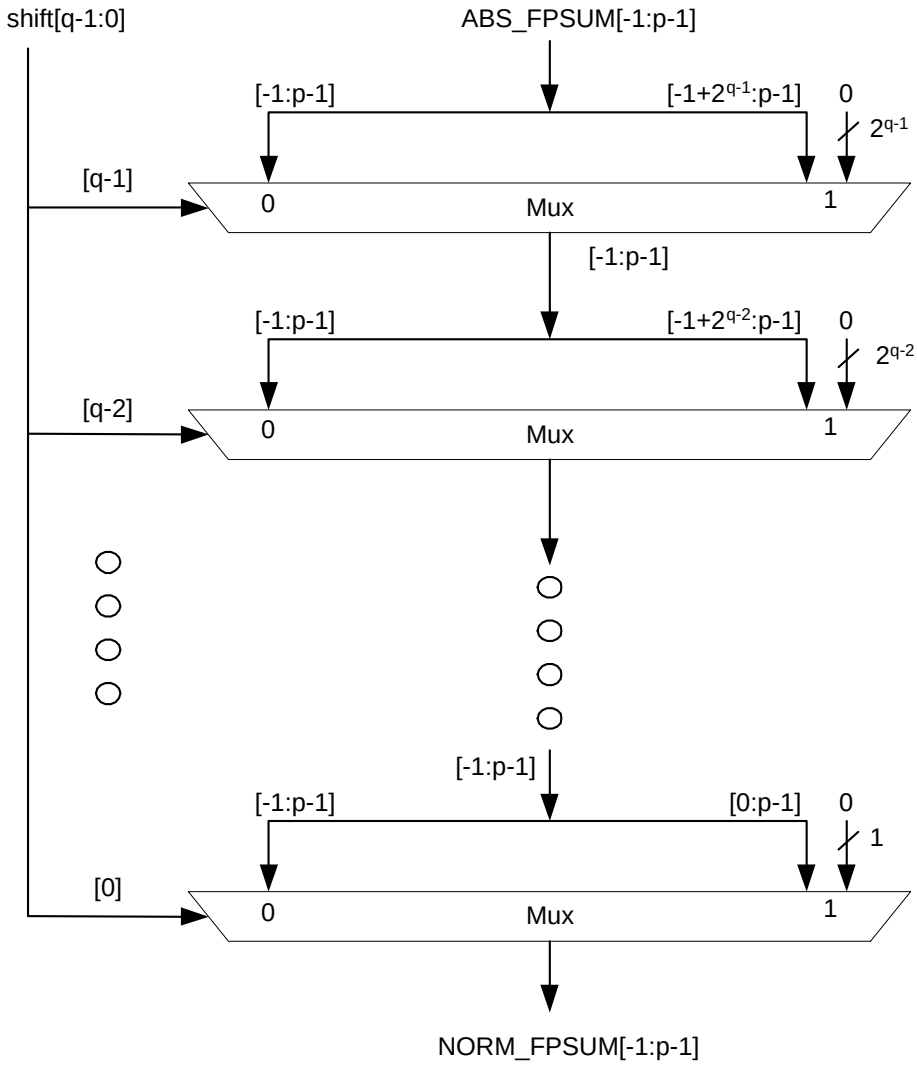

Figure 7.5: N-path normalization shifter for option (2)

shifter and the minimum module. The minimum module calculates $LPZ[q-1:0]$ from the M.S.B. to the L.S.B., which is what the shifter requires. The delay for calculating bit $LZP[i]$ is $2(q-i)-1$ logic levels in the minimum unit and there is additional delay of 2 logic levels in the shifter, so the total delay for the minimum and the shifter is the delay of bit $LZP[0]$ and additional delay of 2 logic levels in the shifter, which is : $2q+1$. In the double-precision adder $q=6$, therefore the delay of the shifter result is 13 logic levels. which is far more the the delay of the shifter in the SE FP-adder, and therefore we have chosen in option (1).

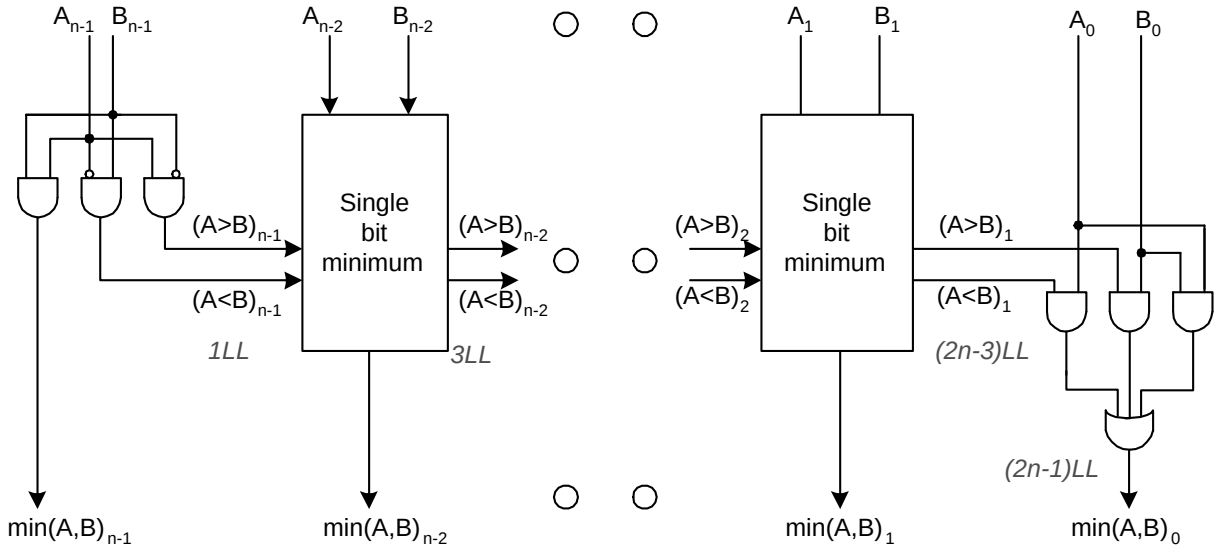

Figure 7.6: N-path minimum module for option (2)

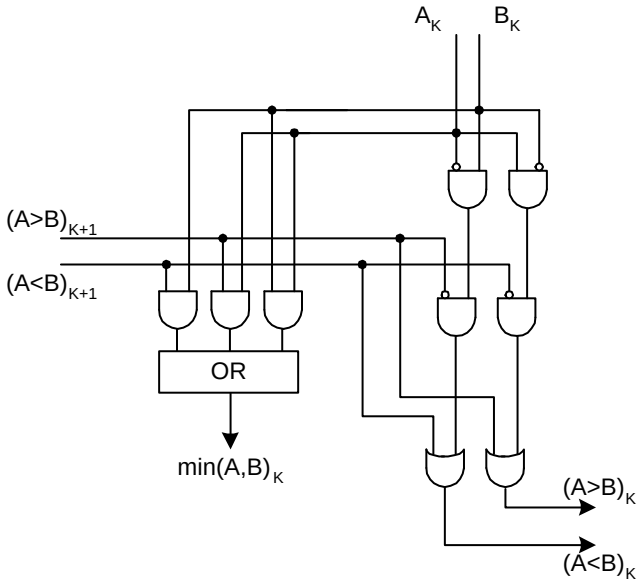

Figure 7.7: Single bit minimum for the minimum module

Chapter 8

The Test Environment

We implemented the modified SE FP-Adder and the testing environment using the "MATLAB" software. Each signal in the design was represented by an integer variable that holds the unsigned value of the signal.
Examples:

1. Significand $F[0:p-1]$ is a signal with p bits where bit [p-1] is the L.S.B and bit [0] is the M.S.B, therefore its value is represented by an integer that equals $\sum_{i=0}^{p} F[i] \cdot 2^{p-1-i}$
2. Exponent $E[n-1:0]$ is a signal with n bits where bit [n-1] is the M.S.B and bit [0] is the L.S.B, therefore its value is represented by an integer that equals $\sum_{i=0}^{n-1} E[i] \cdot 2^{i}$

Based on this concept, we built a level of basic logic-operation function. Examples:

1. $Bitwise\_or(A, B, width)$,
2. $Bitwise\_not(A, width)$,
3. $Concatenate(A, B, width\ of\ B)$, and
4. $Get\_bits(A, width\ of\ A, L.S.B\ from\ A, M.S.B\ from\ A)$.

Above these functions we implemented the modified SE FP-Adder and the testing environment that is depicted figure 8.1. The goal of the testing environment is to invoke an exhaustive tests to a reduced precision floating point adder. In order to do it, the parameters **p** (the width of the significand) and **n** (the width of the exponent have to be initialized.In addition, There is a **test type** parameter for testing only a subset of the inputs space and/or a subset of the rounding modes, for example only the inputs that require the R-Path with "round towards zero" rounding.
The testing environment includes the following elements:

- Addends generator -
  This element creates a sequence of the addends A and B according the test configuration.
- The modified SE FP-Adder -
  The implementation of the modified SE FP-Adder.

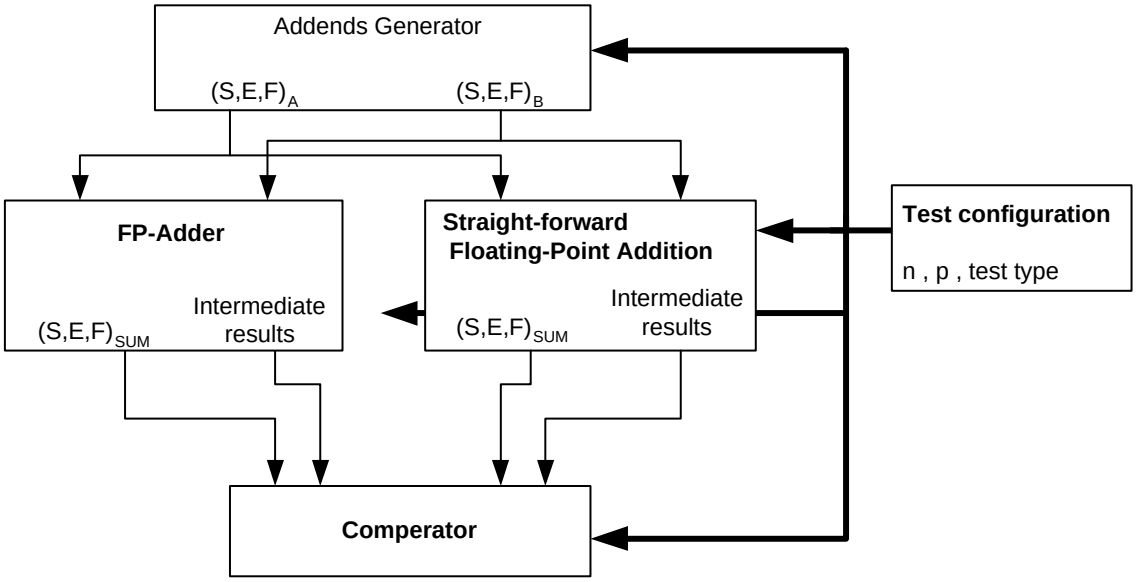

Figure 8.1: The testing environment

- Straight-Forward FP-Addition -
  This is an algorithm that implements the FP-Addition as it is defined by the IEEE standard. It includes the following stages:

  1. Conversion of the addends from (S,E,F) format to real values,
  2. Addition of the real values which produces real value,
  3. Rounding of the result, and
  4. Conversion of the rounded sum from real value to (S,E,F) format.

  In addition the sum, many intermediate results of the modified SE FP-Adder are calculated according to the specifications of the sub-modules of the adder.

- Comperator -
  This module compares both the result of the addition and the intermediate results.

Chapter 9

Summary and Discussion

The SE FP adder is modified to support de-normal numbers and to produce the sign and the exponent of the result in addition to the significand. The modified SE-FP adder supports all four rounding modes, and outputs the rounded sum provided that exceptions do not occur. An analysis of the modifications shows that the additional delay due to the modifications, is only one logic level. Therefore, its latency is roughly 25 logic levels (excluding the latches between pipeline stages).

We suggested two alternatives for supporting de-normals, for each alternative we analyed the timing and decided on alternative (1).

In order to verify the correctness of the modified SE-FP addition algorithm, two approaches were taken:

1. Correctness proofs of some of the sub-modules are given.
2. Parameterization of the algorithm's data width enabled exhaustive testing of the algorithm with reduced data width.
   This approach described in detail in reference [BL].

The paper gives a complete description of the modified SE FP adder, including:

- The modifications for de-normal numbers,
- Integration of the ES-rounding algorithm (described in reference [ES]) into the SE-FP adder, and
- Computation of the exponent and the sign.

Bibliography

[SE] Peter Seidel and Guy Even, "On the Design of Fast IEEE Floating-Point Adders", Proceedings of the 15th IEEE Symposium on Computer Arithmetic, 2001.

[IEEE] IEEE standard for binary floating point arithmetic. ANSI/IEEE754-1985, New York, 1985.

[ES] Guy Even and Peter Seidel, "A comparison of three rounding algorithms for IEEE Floating-Point multiplication", Technical Report EES1998-8,Dept. of Elec. Eng. Systems,Tel-Aviv Univ.,1998. http://www.eng.tau.ac.il/Utils/reportlist/repfarm.html.

[BL] Shahar Bar-Or and Yariv Levin, "Test pattern generation for parametric designs: from low precision to high precision"

[DEM] Mark Daumas Guy Even and David W. Matula, "Partial Compression and Recoding: an arithmetic unit design tool"

Appendix A

Additional Required Designs

In this appendix we depict few sub-modules that are used in the SE FP-Adder. They complete the description of the FP-Adder.

A.1 Recoding Modules

Figures A.1 and A.2 depict the recoding modules that are used in the first cycle of N-path for the estimation of leading zeros of $|f_{prenorm}|$.

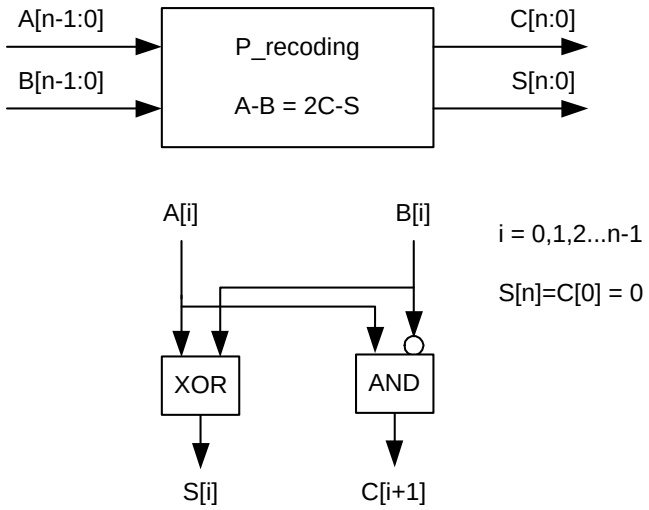

Figure A.1: P-recoding

A.2 Priority Encoding

In addition to recoding, the leading zeros estimation requires two instances of the priority encoders. Figure A.3 depicts the I/O of the priority encoder.

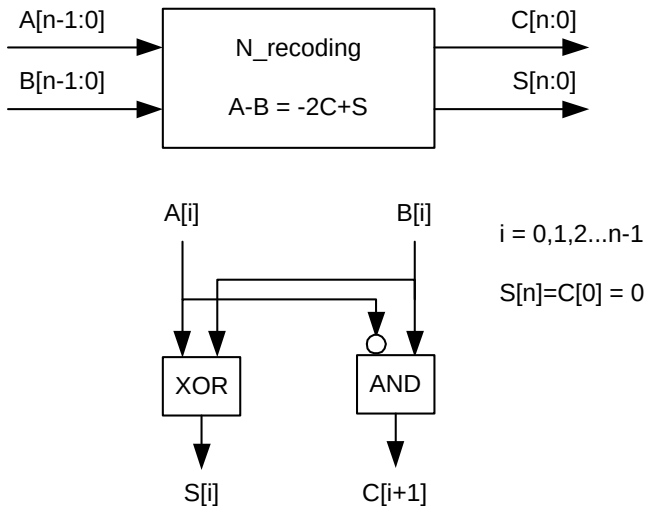

Figure A.2: N-recoding

Figure A.3: The priority encoder I/O

The priority encoder searches its input from the M.S.B to the L.S.B for the first bit that is one and produces in its output the number of zeros it searched before finding the bit with value of "1". The output of the priority encoder respects the equation:
$In[2^n - 1 : 0] =< 0^{Out}, 1, \{0,1\}^{2^n-1-Out} >$

where:

- $Out \in [0, 2^n - 1]$
- $In[2^n - 1 : 0] = 0$ is illegal input to the priority encoder.

A.3 Decoding

In order to support de-normalized addition we needed to limit the leading zeros estimation of $|f_{prenorm}|$ to $min\{EA, EB\}$. We implemented this limitation with a decoding of $min\{EA, EB\}$.
The I/O and an optimized implementation of the decoder is depicted in figures A.4 and A.5 respectively.

The decoder's output is: $Out[2^n - 1 : 0] =< 0^{2^n-1-In}, 1, 0^{In} >$ where:

- $In \in [0, 2^n - 1]$

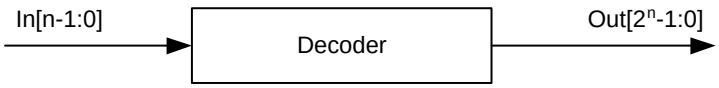

Figure A.4: The decoder I/O

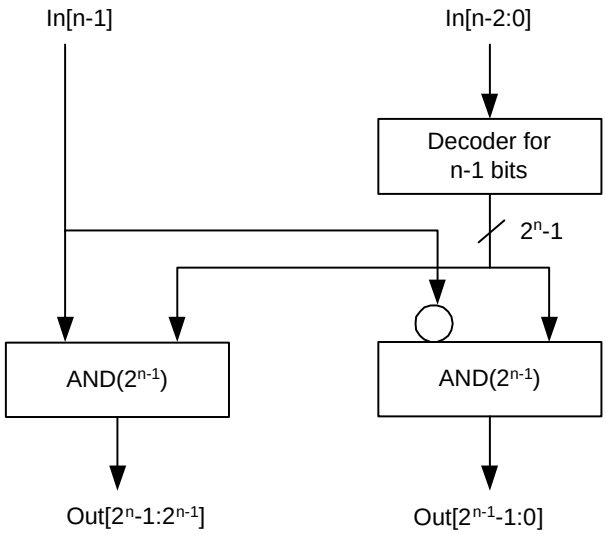

Figure A.5: Optimized decoder implementation

A.4 Rounding decision

The second cycle of the R-path includes a "rounding decision" module for decision if the rounding of the sum's significand is towards zero or towards infinity.
Figure A.6 depicts the "rounding decision" module.

A.5 Compound Addition

The compound addition is calculation of $A + B$, and in only one additional logic level - calculation of $A + B + 1$. The addends are assumed to be positive and represented by unsigned representation.
Both in the R-path and in the N-path there is a usage of compound addition. In the R-path there is a need to calculate two candidates for the significand f_{far} (the "rounding decision" module selects between them), and in the N-path the compound addition is for conversion of $fopsum$ that is represented in ones complement to its absolute value. Figure A.7 depicts a compound adder.

The equations of the compound adder are:

- $P[i] = xor(A[i], B[i])$ $\qquad i \in [0, n-1]$

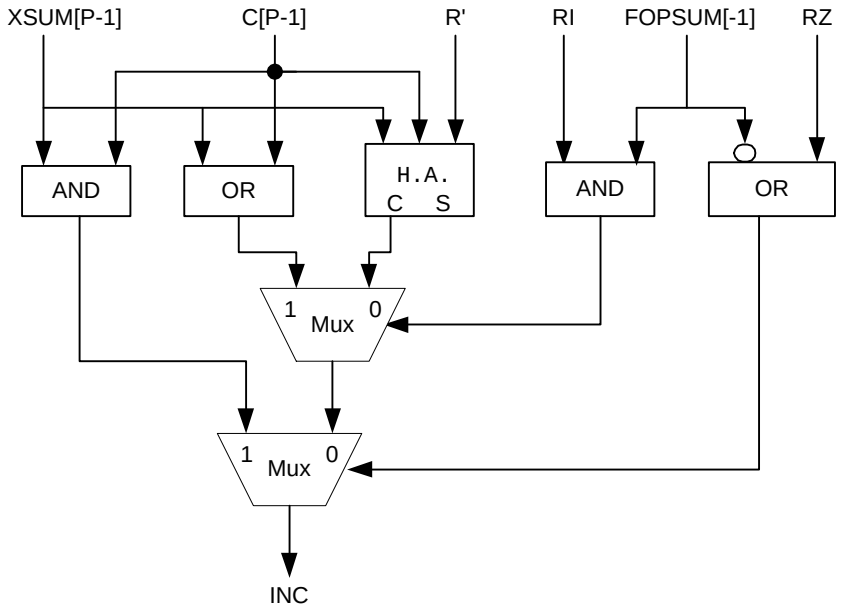

Figure A.6: The rounding decision implementation

- $\sigma[i] = A[i] + B[i]$ $\sigma \in \{0,1,2\}^n$
- $\pi[i] = \sigma[i] \otimes \sigma[i-1] \otimes ... \otimes \sigma[0]$ $\pi \in \{0,1,2\}^n$
- $GC[j] = \begin{cases} 1 & if\ \pi[j-1] = 2 \\ 0 & \text{otherwise} \end{cases}$ $j \in [1,n]$
- $PCI[j] = \begin{cases} 1 & if\ \pi[j-1] = 1 \\ 0 & \text{otherwise} \end{cases}$
- $GCI[j] = or(PCI[j], GC[j])$
- $S[k] = \begin{cases} P[0] & if\ k = 0 \\ GC[n] & if\ k = n \\ xor(P[k], GC[k]) & \text{otherwise} \end{cases}$ $k \in [0,n]$
- $SI[k] = \begin{cases} \overline{P[0]} & if\ k = 0 \\ GCI[n] & if\ k = n \\ xor(P[k], GCI[k]) & \text{otherwise} \end{cases}$

where:

- $X \otimes Y = \begin{cases} 0 & if\ X = 0 \\ Y & if\ X = 1 \\ 2 & if\ X = 2 \end{cases}$ $X, Y \in \{0,1,2\}$

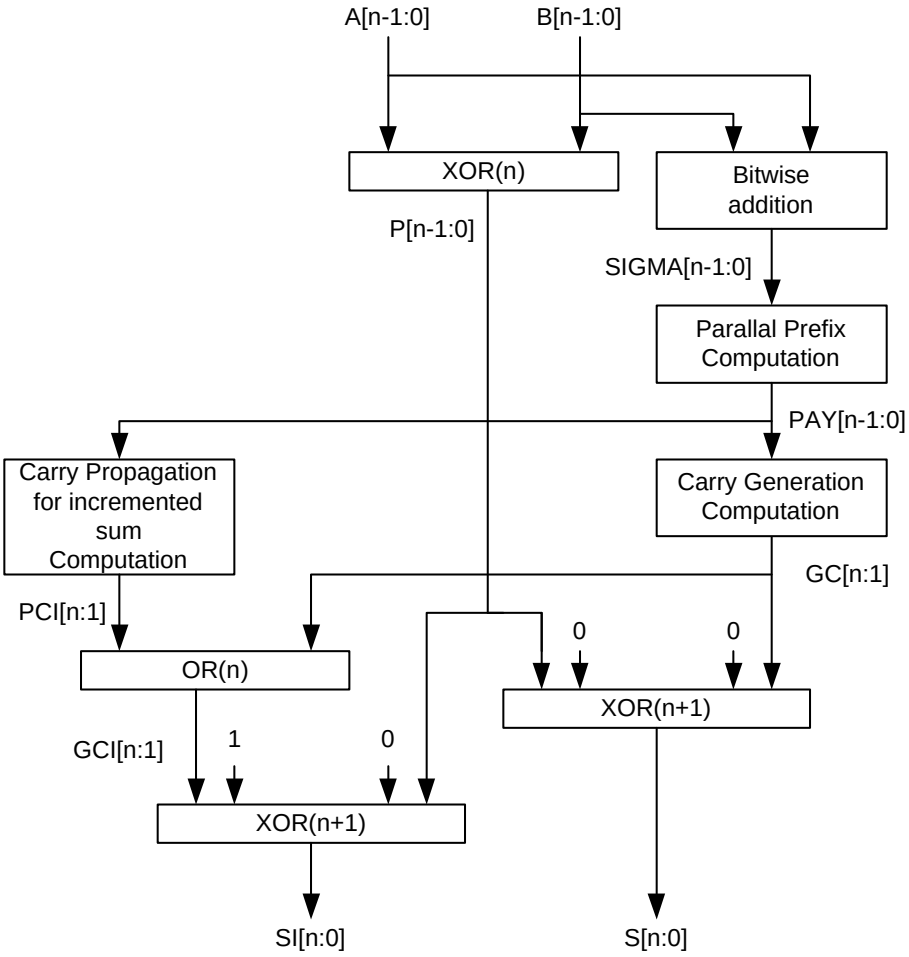

Figure A.7: Compound Adder