

Known errata in *Mueller/Paul: Computer Architecture, Complexity and Correctness*

Maintained by Dirk Leinenbach
dirkl@cs.uni-sb.de

June 30, 2003

1. Reported by Jrg Fischer, Saarland University
Page 8, table 2.1: the delay of AND,OR is 2.
2. Reported by Jrg Fischer, Saarland University
Page 22: the brackets $\langle \cdot \rangle$ are missing in the equation-chain $z = \dots$
3. Reported by Warren E. Ferguson, Intel
Page 44: since $\langle a \rangle \in \{0, \dots, 2^n - 1\}$ and $B_{2j} \in \{-2, \dots, 2\}$, it holds

$$C_{2j} \in \{-2^{n+1} + 2, \dots, 2^{n+1} - 2\},$$
$$D_{2j} \in \{0, \dots, 2^{n+1} - 2\}.$$

Since D_{2j} has a $n + 1$ -bit representation d_{2j} (which was not the case for the faulty range $D_{2j} \in \{0, \dots, 2^{n+1}\}$), this does not affect the further arguments.

4. Reported by Jrg Fischer, Saarland University
Page 46: The definition of $S'_{2j,2k}$ must be

$$S'_{2j,2k} := \sum_{t=j}^{j+k-1} \langle g_{2k} \rangle \cdot 4^{t-1}.$$

5. Reported by Jrg Fischer, Saarland University
Page 47, Lemma 2.7, induction base: $\dots < 2^{n+6} \cdot 2^{2j-2} = \dots$
Induction step: the right bracket \rangle is in the wrong place.
6. Reported by Jochen Preiss, Saarland University
Page 47, Lemma 2.7: the second line of the inequality-chain must be

$$\dots < 2^{n+2j+2k} + 2^{n+5} \cdot 2^{2j+2k-4}.$$

7. Reported by Jochen Preiss, Saarland University
Page 67, line 1: replace `begz` by `sgri`.
8. Reported by Dirk Leinenbach, Saarland University
Page 322, property 3 of representable numbers: $(-2^{-e_{min}}, 0]$ must be $(-2^{e_{min}}, 0]$
9. Reported by Christoph Berg, Saarland University
Page 323, 4th paragraph: $e = \llbracket e[n-1] \rrbracket_{bias}$ must be $e = \llbracket e[n-1:0] \rrbracket_{bias}$
10. Reported by Dirk Leinenbach, Saarland University
Page 329, end of last line of proof of lemma 7.1: it should be $= r(x')$ instead of $= r'(x)$.

11. Reported by Chris Jacobi, Saarland University

In the `unpack`-circuit (pg. 355), sub-circuits `lzero(53)` and `CLS(53)` are used, even though these circuits were only designed for powers of two in chapter 2.

One can derive `lzero(53)` from `lzero(64)` by padding the input with 1's, since $lz(x) = lz(x \cdot 1^k)$.

The cyclic shifter `CLS(53)` in the `unpack`-circuits can be replaced by a logic right shifter `LRS(53)`. It is easy to design logic right shifters for non-power-of-two inputs.

12. Reported by Chris Jacobi, Saarland University

Page 383. In the circuit `Sign/ExpMD`, a carry-in is fed into the 4/2-adder, although 4/2-adders do not feature a carry-in. To fix this, add a 3/2-adder(7) which adds `lza`, `lzb` (or the inverse) and a constant 1. The output of the new 3/2-adder is sign-extended and fed into the 4/2 adder, instead of the inputs `lza` and `lzb`.

13. Reported by Chris Jacobi, Saarland University

Page 392, 5th item: the input factoring shall not satisfy $f_r[-1 : 0] = 00 \Rightarrow OVF = 0$, but

$$f_r[-1 : 0] = 00 \Rightarrow e \leq e_{max}.$$

Otherwise, the correctness argument for the `OVF1`-computation in circuit `Flags` is wrong (e.g. $e_r = e_{max} + 1, f_r = 0.5$, i.e. $f_r[-1 : 0] = 00$; no overflow occurs, although `OVF1` is asserted).

The new condition for the input factoring is satisfied

- by the adder, since the delivered exponent is the maximum of the input exponents, and hence $e_r \leq e_{max}$.
- by the multiplier, since it delivers $f_r < 1$ only if one of the operands is denormal, and hence the sum of the input exponents $e_r \leq e_{max}$.

14. Reported by Chris Jacobi, Saarland University

In circuits `ExpNorm` (pg. 400), a carry-in is fed into the compound-adder, although compound adders do not feature a carry-in (cf. chapter 2). To fix this, incorporate the constant increment by 1 into the constant in the 3/2-adder.

15. Reported by Chris Jacobi, Saarland University

Page 407: the circuit for the rounding decision (fig 8.34) does not conform with table 8.5 (e.g., $s = 0, r = st = 1$, mode r_u). Replace the `XOR` gate by an `XNOR` gate.