Proving the Correctness of a Complete Floating Point Unit on the Gate Level

Christian Jacobi

cj@cs.uni-sb.de

Saarland University

Outline

- Project Description
- Related Work
- Theory of IEEE Rounding
- Hardware Description and Verification

Project Goals

- Verification of a complete CPU featuring
  - MIPS Instruction Set Architecture
  - 5 Stage Pipeline
  - Tomasulo - Scheduler
  - Precise, nested Interrupts
  - Delayed Branch
  - **IEEE compliant Floating Point Unit**

- Verification using Theorem Proving, PVS [CRSS94]
- Automatic Translation to Verilog HDL
- Implementation on Xilinx FPGA / ASIC

Project Goals (cont.)

Features of the FPU:

- Single and double precision
- Add/Sub, Mul/Div and Conversion
- all 4 IEEE rounding modes supported
- Normal/Denormal operands and results
- Exceptions
- Out-of-order control for operation within Tomasulo CPU

Project Status

- Completed [JK00,Krö00,BJK01,MP00,EP00]:
  - Verification of in-order pipeline,
    including interrupts, delayed branch [Kröning 2000]
  - Verification of Theory of IEEE Rounding
  - Verification of Multiplier/Divider, Rounder,
    and Exception Circuits

- Work in progress (expected March 01):
  - FP-Adder Circuits [Berg]
  - FPU Control
  - Translation Tool [Leinenbach]

Related Work

- Designs taken from [HP96,MP95,MP00]
  small changes/bug fixes

- Proofs taken from [MP00,Min95,BJK01]

- Theory of Rounding [EP97,MP00,Min95,Har99]

- Other designs publicly available
  - SUN Microsparc
  - www.opencores.org

Related Work (CPU)

- Verified RISC Pipelines
  Burch, Dill; Henzinger; McMillan

- Verified Tomasulo Schedulers
  Pnueli, Damm; Hosabettu; McMillan

- Verified Processors
  Tamarack [Joyce 88], AVM-1 [Windley 95]

Related Work (FPU)

- Formalization of IEEE Standard 754 [Min95]

- Verified FPUs
  - AMD K6/K7 [Rus 97,98,00]
  - Intel Itanium [Cor98,LZGS99,Har99]

- Verified Components

  Bryant et.al., Clarke et.al.

- Verified Algorithms [ML96, Har97]

IEEE Formalization

- IEEE-Standard 754 informal
- Factorings (s, e, f) [Matula 96, EP97]
  - Sign-Bit $s \in \{0, 1\}$
  - Exponent $e \in \mathbb{Z}$
  - Significant $f \in \mathbb{R}, 0 \leq f < 2$
- Value $[s, e, f] := (-1)^s \cdot 2^e \cdot f$

IEEE Formalization (cont.)

A factoring (s, e, f) is called

- *normal*, if $e \geq e_{min}$ and $1 \leq f < 2$
- *denormal*, if $e = e_{min}$ and $0 \leq f < 1$
- *IEEE factoring*, if normal or denormal
- p*-representable*, if $2^p \cdot f \in \mathbb{N}$

THEOREM: every real $x \neq 0$ has a unique IEEE factoring $\eta(x) = (s, e, f)$ with $[\eta(x)] = x$.

$x \neq 0$ called p-representable, if $\eta(x)$ p-representable

Rounding

- *"... In this mode [round to nearest], the representable value nearest to the infinetly precise result [of the floating point operation] shall be delivered; if the two nearest representable values are equally near, the one with its least significant bit zero shall be delivered ..."* [IEEE]
- Define [Min95]

$$r_{int}(x) := \begin{cases} \lfloor x \rfloor & \text{if } x - \lfloor x \rfloor < \lceil x \rceil - x \\ \lceil x \rceil & \text{if } x - \lfloor x \rfloor > \lceil x \rceil - x \\ x & \text{if } \lfloor x \rfloor = \lceil x \rceil \\ 2 \lfloor \lceil x \rceil / 2 \rfloor & \text{else} \end{cases}$$

- Let $x \in \mathbb{R}, x \neq 0$, and $(s, e, f) = \eta(x)$. Define the rounding function rd as:

$$rd(x) := 2^{e-p} \cdot r_{int}(x \cdot 2^{-(e-p)}).$$

- Similar definitions for other rounding modes.

Rounding (cont.)

Conformance of the rounding function to the standard?

THEOREM [Min95]:

1. $rd(x)$ is p-representable,

2. $\forall x'$, x' p-representable: $|x - x'| \geq |x - rd(x)|$,

3. $|x - rd(x)| \leq 2^{e-p-1}$, and
   $|x - rd(x)| = 2^{e-p-1}$ implies $rd(x)$ even.

Decomposition Theorem

Let $x \in \mathbb{R}, x \neq 0, (s, e, f) = \eta(x)$, and $rd(x) \neq 0$.

$$\begin{aligned} \text{sigrd}(f) &:= 2^{-p} \cdot rd_{int}(f \cdot 2^p) \\ \text{postnorm}(s, e, f) &:= \begin{cases} (s, e+1, 1) & \text{if sigrd}(f) = 2 \\ (s, e, \text{sigrd(f)}) & \text{otherwise} \end{cases} \end{aligned}$$

THEOREM: $\eta(rd(x)) = postnorm(\eta(x))$.

Exceptions

- Overflow ... *shall be signaled whenever the destination format's largest finite number is exceeded in magnitude by what would have been the rounded floating-point result were the exponent range unbounded.*

$$\mathsf{OVF}(x) := |rd(x)| > X_{max} = 2^{e_{max}} \cdot (2 - 2^{1-p})$$

- Tiny before rounding: ... *when a nonzero result computed as though both the exponent range and the precision were unbounded would lie strictly between* $\pm 2^{e_{min}}$.

$$\mathsf{TINY}(x) := x \neq 0 \text{ and } |x| < 2^{e_{min}}$$

- Inexact result: ... *when the delivered result differs from what would have been computed were both exponent range and precision unbounded.*

$$\mathsf{INX}(x) := (x \neq rd(x)) \text{ or } \mathsf{OVF}(x)$$

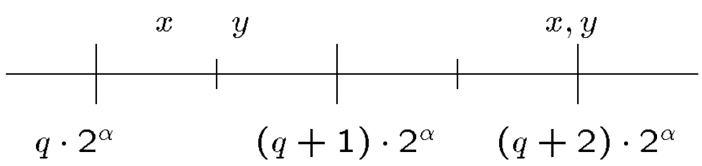
Overflow

Let $(s, e, f) = \eta(x)$. Overflow can be decomposed:

- OVF_before(x) := $e > e_{max}$
- OVF_after(x) := ($e = e_{max}$ and sigrd(f) = 2)

It holds OVF(x) iff (OVF_before(x) or OVF_after(x))

α-equivalence



Two reals x and y are α-equivalent ($x =_\alpha y$) [MP00,EP00] iff

- x=y, or
- $\exists q \in \mathbb{Z} : q \cdot 2^\alpha < x, y < (q+1) \cdot 2^\alpha$

Define α-representatives

$$[x]_\alpha := \begin{cases} x & \text{if } \exists q \in \mathbb{Z} : q \cdot 2^\alpha = x \\ (q+0.5) \cdot 2^\alpha & \text{otherwise} \end{cases}$$

α-equivalence (cont.)

THEOREM: Let $x =_\alpha y$.

1. $=_\alpha$ is an equivalence relation, $[\cdot]_\alpha$ are representatives.

2. $x \cdot 2^k =_{\alpha+k} y \cdot 2^k$ for all $k \in \mathbb{Z}$.

3. $x + q \cdot 2^\alpha =_\alpha y + q \cdot 2^\alpha$ for all $q \in \mathbb{Z}$.

4. $x =_\beta y$ for all $\beta \in \mathbb{Z}, \beta \geq \alpha$.

Similar relations hold for $[\cdot]_\alpha$.

α-equivalence & rounding

THEOREM:
Let $x, x' \in \mathbb{R}, (s, e, f) = \eta(x), (s', e', f') = \eta(x')$, and $x =_{e-p} x'$.

1. $e = e'$, i.e., x and x' are the same order of magnitude.

2. $\eta([x]_{e-p}) = (s, e, [f]_{-p})$

3. $[f]_{-p}$ can be computed by sticky bit computation (OR tree)

4. $rd(x) = rd(x')$

$\Rightarrow$ Paper proof argues geometrically.

α-equivalence & exceptions

THEOREM:
Let $x, x' \in \mathbb{R}, (s, e, f) = \eta(x)$, and $x =_{e-p} x'$.

1. $\text{OVF\_before}(x) = \text{OVF\_before}(x'), \text{OVF\_after}(x) = \text{OVF\_after}(x')$

2. $\text{OVF(x)} = \text{OVF(x}')$

3. $\text{TINY(x)} = \text{TINY(x}')$

4. $\text{INX(x)} = \text{INX(x}')$

$\Rightarrow$ one direction missed in paper proof

Addition

$$
\begin{aligned}
a &= [s_a, e_a, f_a] = (-1)^{s_a} \cdot 2^{e_a} \cdot f_a \\
b &= [s_b, e_b, f_b] = (-1)^{s_b} \cdot 2^{e_b} \cdot f_b
\end{aligned}
$$

$$
\begin{aligned}
\delta &:= e_a - e_b \geq 0 \text{ (w.l.o.g.)} \\
f' &:= [2^{-\delta} \cdot f_b]_{p+1}
\end{aligned}
$$

THEOREM:

$$a + b =_{e-p} 2^{e_a} \left((-1)^{s_a} \cdot f_a + (-1)^{s_b} \cdot f' \right)$$

Classical Newton/Raphson Iteration

Define a sequence x_i by

$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}}.$$

Solving for x_{i+1} yields

$$x_{i+1} = x_i - f(x_i)/f'(x_i).$$

With $f(x) = 1/x - f$ this yields

$$x_{i+1} = x_i \cdot (2 - f \cdot x_i),$$

and

$$|1/f - x_{i+1}| \leq 2 \cdot |1/f - x_i|^2.$$

Newton/Raphson Iteration with Finite Precision

$$
\begin{aligned}
z_i &:= \lfloor f \cdot x_i \rfloor_\sigma \\
A_i &:= 2 - z_i - 2^{-\sigma} = \overline{z_i[0:\sigma]} \\
x_{i+1} &:= \lfloor A_i \cdot x_i \rfloor_\sigma \\
\delta_i &:= 1/f - x_i
\end{aligned}
$$

This yields $\delta_{i+1} < 2 \cdot \delta_i^2 + 2^{-\sigma+1}$.

With appropriate x_0 (2KBit lookup), and $\sigma = 57$, we have

$$\delta_2 < 11/10 \cdot 2^{-29}, \;\; \delta_3 < 2^{-55}.$$

Computing the p-Representative

With $E := \left\lfloor f_a \cdot x_{2/3} \right\rfloor_{p+1}$ we have an approximation of the quotient f_a/f (p=24/53).

$$[f_a/f]_{p+1} = \begin{cases} E + 2^{-(p+2)} & \text{if } f_a < f \cdot (E + 2^{-(p+1)}) \\ E + 2^{-(p+1)} & \text{if } f_a = f \cdot (E + 2^{-(p+1)}) \\ E + 3 \cdot 2^{-(p+2)} & \text{if } f_a > f \cdot (E + 2^{-(p+1)}) \end{cases}$$

Implementation of the Rounder

Inputs:

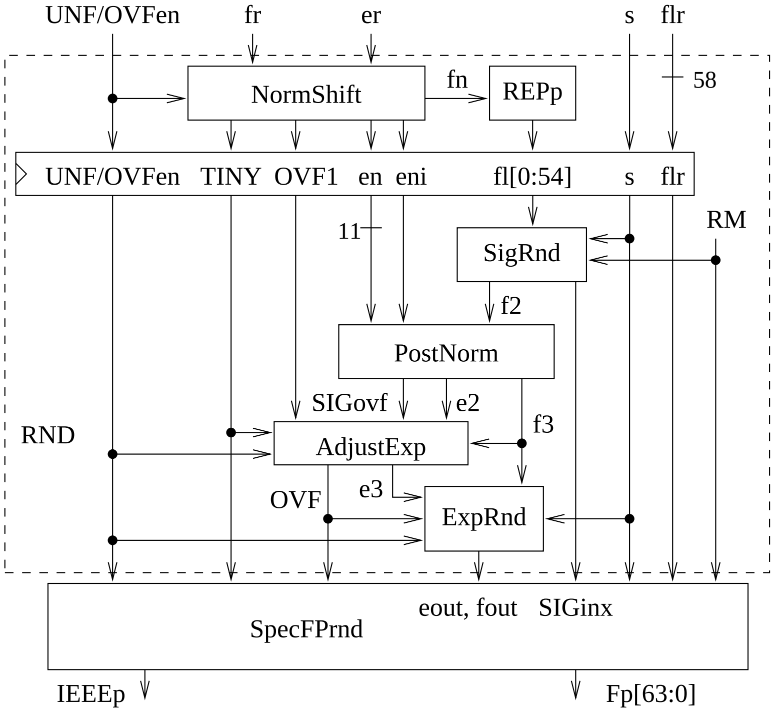
- $s_r \in \{0,1\}, e_r \in \{0,1\}^{13}, f_r \in \{0,1\}^{57}$;
  Result $x =_{e-p} [s_r, twoc(e_r), bin(f_r) \cdot 2^{-56}]$

- $(s_r, twoc(e_r), bin(f_r) \cdot 2^{-56})$ need not be normalized

- $f_r[56:55] = 00$ implies $twoc(e_r) \leq e_{max}$

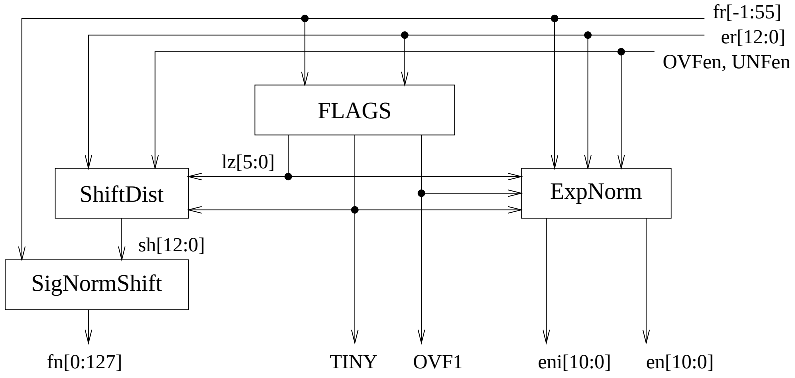
Output $rd(x)$ in IEEE format.

Implementation of the Rounder



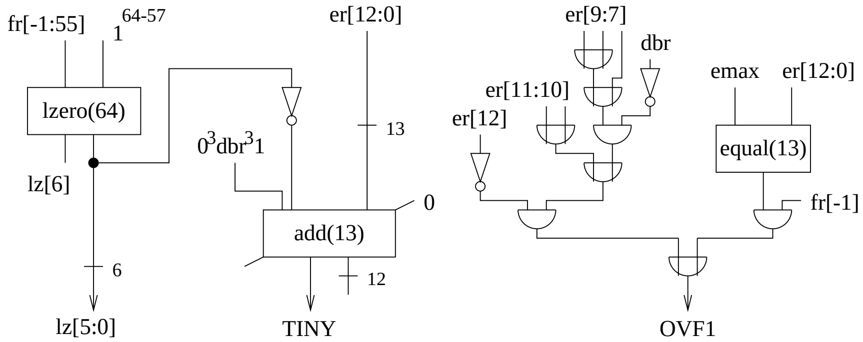
$$\eta(rd(x)) = postnorm(\eta(x)).$$

Normalization Shift



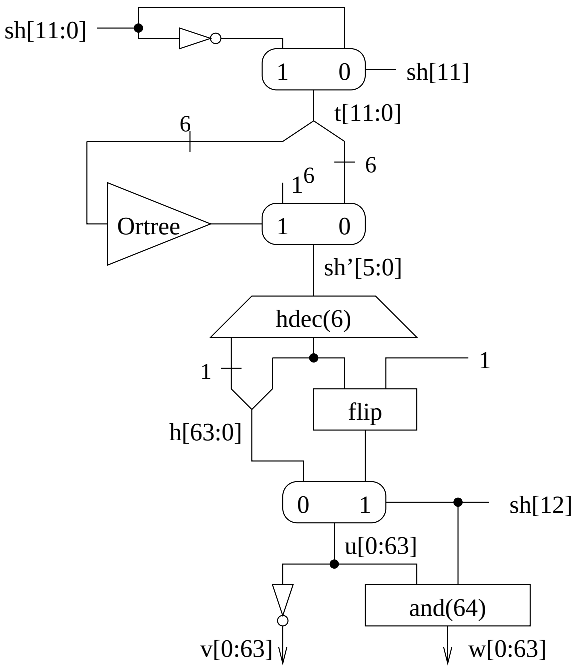
Correctness:
- $[s, e_n, f_n] =_{e_n - p} x$
- (s, e_n, f_n) IEEE factoring
- OVF1 iff OVF_before(x)
- TINY iff TINY(x)

Computation of TINY/OVF1



- TINY: compare exponent before rounding $(e_r + lz)$ with e_{min}
- OVF1: evaluate high-order bit of e_r to check, if $e_r > e_{max}$, and compare e_r with e_{max}

Computation of the Shifter Mask



Specification of Hardware in PVS

```
zero_test(n: nat, b: bvec[n]):
    Recursive bit =
    If n=1 Then
        NOT b^0
    Else
    Let a=floor(n/2) In
        zero_test(n-a, b^(n-1,a)) AND
        zero_test(a, b^(a-1,0))
    Endif
```

Summary & Future Work

- Verification of a complex CPU & FPU
- Verification of Theory of IEEE Rounding
- Verification of FPU datapaths

- Completion of CPU/FPU specification and verification
- Translation to Verilog HDL
- Implementation on FPGA / ASIC

Literature

BJK01 Berg, Jacobi, Kröning: Formal Verification of a Basic Circuits Library. Accepted for IASTED Applied Informatics, 2001

Cor98 Cornea-Hasegan: Proving the IEEE Correctness of Iterative Floating-Point Square Root, Divide, and Remainder Algorithms, Intel Technology Journal Q2, 1998

CRSS94 Cyrluk, Rajan, Shankar, Srivas: Effective theorem proving for hardware verification. In 2nd Int Conf on Theorem Proving in Circuit Design, 1994

EP97 Even, Paul: On the Design of IEEE Compliant Floating Point Units, International Symposium Computer Arithmetic 97

EP00 Even, Paul: On the Design of IEEE Compliant Floating Point Units, IEEE Trans Comp 49:5 2000

Har97 Harrison: Floating-Point Verification in HOL Light: The Exponential Function, TR 428, University Cambridge Computer Lab, UK, 1997

Har99 Harrison: A Machine-Checked Theory of Floating Point Arithmetic, TPHOL 1999

HP96 Hennessy, Patterson: Computer Architecture a Quantitive Approach, Morgan Kaufman, 1996

JK00 Jacobi, Kröning: Proving the Correctness of a Complete Microprocessor. In Informatik 2000, Proc. of the 30. Jahrestagung der Gesellschaft für Informatik, 2000

Krö00 Kröning: Formal Verification of Pipelined Microprocessors, PhD Thesis (Draft), Saarland University, 2000

LGZS99 O'Leary et.al.: Formally Verifying IEEE Compliance of Floating-Point Hardware, Intel Technology Journal Q2, 1999

Min95 Miner: Defining the IEEE-854 floating-point standard in PVS. Technical memorandum 110167, NASA Langley Research Center, Hampton, VA 23681-0001, USA, 1995

ML96 Miner, Leathrum: Verification of IEEE Compliant Subtractive Division Algorithms, FMCAD 96

MP95 Müller, Paul: The Complexity of Simple Computer Architectures, Springer 1995

MP00 Müller, Paul: Computer Architecture - Complexity and Correctness, Springer 2000

Rus97 Russinoff: Proof of IEEE Compliance of the AMD-K5 Floating Point Square Root Microcode, Formal Methods in System Design, 14:1, 1999

Rus98 Russinoff: A Mechanically Checked Proof of IEEE Compliance of the Floating Point Multiplication, Division, and Square Root Algorithms of the AMD-K7 TM Processor, LMS Journal of Computations and Mathematics, 1998

Rus00 Russinoff: Mechanical Verification of RTL: The Athlon Floating-Point Adder, FMCAD 00