



VLSI Design & Consultancy

## **DATASHEET**

**Floating Point AddSub**

**Version 1.0**



## 1 Overview

The Think-Silicon *Floating Point AddSub* is a web configurable generator for standard IEEE 754 single precision floating point Adder/Subtractor. The generated modules support Denormalised numbers and perform rounding to nearest.

## 2 Floating Point numbers

The format of the IEEE 754 single precision numbers is shown in Table 2-1:

Table 2-1 IEEE 754 single precision floating point numbers format

BITS	NAME	DESCRIPTION
31	sign	This is the sign bit. 0 signifies a positive number and a 1 a negative number
30:23	exponent	This number defines the power of 2 that is multiplied by the fractional part. It also defines if the number is normal.
22:0	fraction	These are fractional bits of the number.

The number is defined as follows:

$$number = -1^{sign} \times 2^{(127 - exponent)} \times 1.(fraction)$$

There are also special bit patterns that define +ve and -ve  $\infty$  (infinity) as well as positive and negative  $\emptyset$  (zero). Those patterns as are also supported and handled according to the standard.

## 3 Features

- IEEE-754 single precision floating point compliance
- Denormalized numbers support
- Rounding to the nearest

## 4 Architecture

### 4.1 Block Diagram

The generated fp\_addsub module Block Diagram is show in Figure 3-1. The IP module is fully combinatorial and performs addition or subtraction operations on A and B numbers.

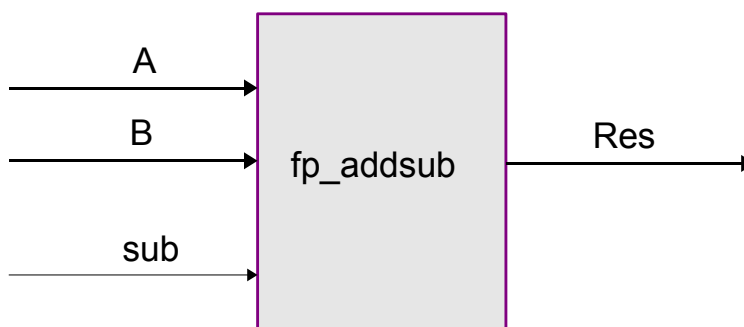


Figure 3-1 fp\_addsub Block Diagram

## 4.2 Port Diagram

The Port Diagram of *fp\_addsub* is shown in Figure 3-2. The input 32-bit ports, *inA* and *inB* comply to the IEEE-754 single precision floating point standard. The result of the addition/subtraction is placed in the *Res* output port and it is also encoded to comply to IEEE-754 single precision floating point standard. The *fp\_addsub* module performs also comparison between the numbers in *inA* and *inB* ports. The result of the comparison is defined by the *aGtB*, *aEqB*, *aLtB* ports.

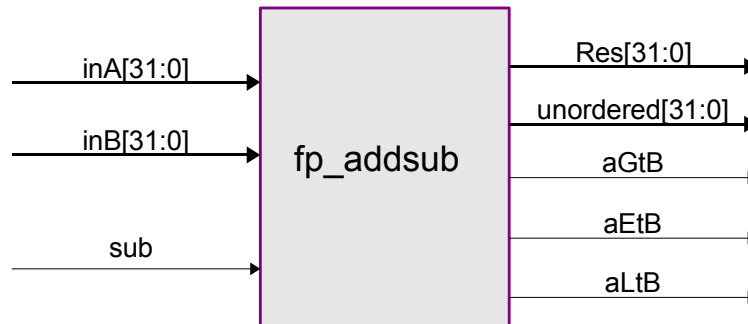


Figure 3-1 *fp\_addsub* Port Diagram

## 4.3 Port Interface

The Port Interface of the *fp\_addsub* module is shown in Table 3-1.

Table 3-1 *fp\_addsub* Port Interface

PORT	TYPE	DESCRIPTION
inA[31:0]	Input	The first addend number
inB[31:0]	Input	The second addend number
sub	Input	Operation control. Should be set to 0 for addition and to 1 for subtraction
Res[31:0]	Output	The result from the addition/subtraction
unordered[31:0]	Output	Unordered addition/subtraction result
aGtB	Output	When 1 A is greater than B
aEtB	Output	When 1 A is equal B
aLtB	Output	When 1 A is less than B

## 5 Interfacing fp\_addsub

The design is fully combinatorial, thus should you require to pipeline the unit, you are advised to do so at the synthesis level, using a balanced registers technique that will provide better results for your target technology.

## 6 Generator Usage

The *Floating Point AddSub* generator employs a graphical web user interface (GUI) for configuring and

generating the *fp\_addsub* module. In order to use the GUI you must sign-in Think Silicon Ltd web site. If already registered, click on *Sign-in* link in the upper, right side of the web page. Otherwise click on the *Register* link first and follow the instructions. The generator GUI page is shown in Figure 6-1. Press the *Generate* button in order to generate the *fp\_addsub* module.

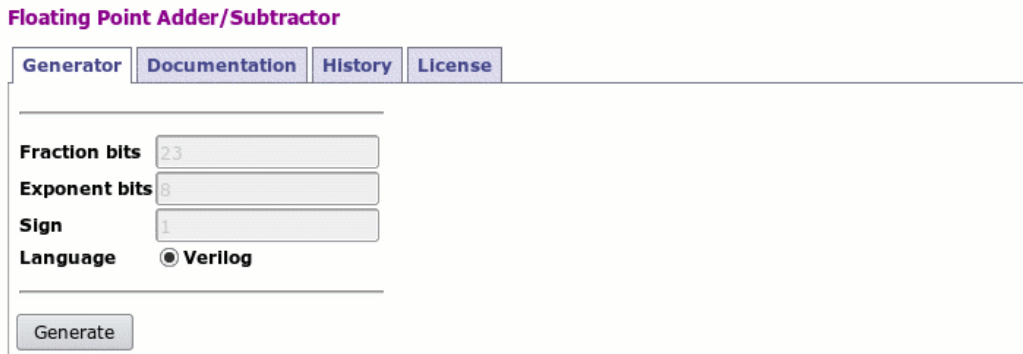


Figure 6-1 The *Floating Point AddSub* GUI page

## 7 Deliverables

The package generated with *Floating Point AddSub* consists of the present document and source code files in Verilog™<sup>1</sup> HDL and C language. The files are listed in Table 7-1.

Table 7-1 *Floating Point AddSub* Deliverables

FILE	DESCRIPTION
./src/fp_addsub.v	Floating Point Adder module
./src/tb_vectors.v	Floating Point Adder testbench
./src/gen_vectors.c	Floating Point Adder Test Vectors generator
./src/Makefile	Makefile with simulation commands
./parameters.txt	Floating Point Adder module configuration parameters
./doc/TSi_fp_addsub.pdf	The present document

## 8 Verification

The Verification Flow Diagram of the *fp\_addsub* core is shown in Figure 8-1.

<sup>1</sup> Verilog is a trademark of Cadence Design Automation. (<http://www.cadence.com>)

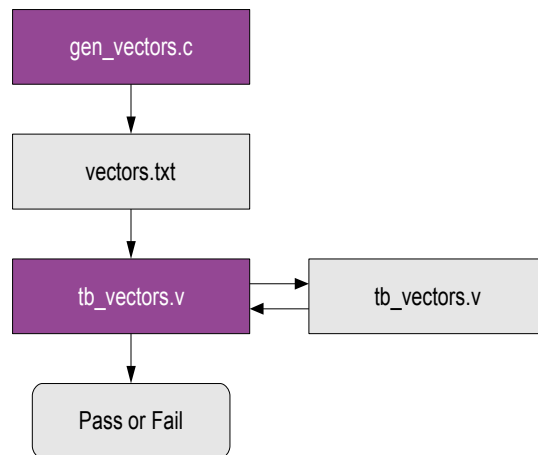


Figure 8-1 Verification Flow Diagram of the fp\_addsub core

The testbench is based on gcc and the Icarus Verilog simulator and we assume these are installed in your system. In order to generate test vectors and run the verification tests for the *fp\_addsub* core, enter the commands shown in Figure 8-2 in a command line shell.

```

> cd ./fp_addsub
> make
  
```

Figure 8-2 Verification commands for *fp\_addsub* core

This should automatically compile the test vector generator and start the simulator. The simulator finishes with a pass or fail. The test vector generator and the netlist are written in standard C and Verilog<sup>TM 2</sup>, so other compilers and Simulators, such as NCVerilog<sup>TM 3</sup> and MTI Modelsim<sup>TM 4</sup> can be used.

2 Verilog is a trademark of Cadence Design Automation, UK (<http://www.cadence.com>)

3 NCVerilog is a trademark of Cadence Design Automation, UK (<http://www.cadence.com>)

4 MTI Modelsim is a trademark of Mentor Graphics (<http://www.mentor.com>)



**Contact:**

Think Silicon Ltd  
Suite B12  
Patras Science Park  
Rion Achaias 26504  
Greece

web: <http://www.think-silicon.com>

email: [info@think-silicon.com](mailto:info@think-silicon.com)

Tel: +30 2610 911543