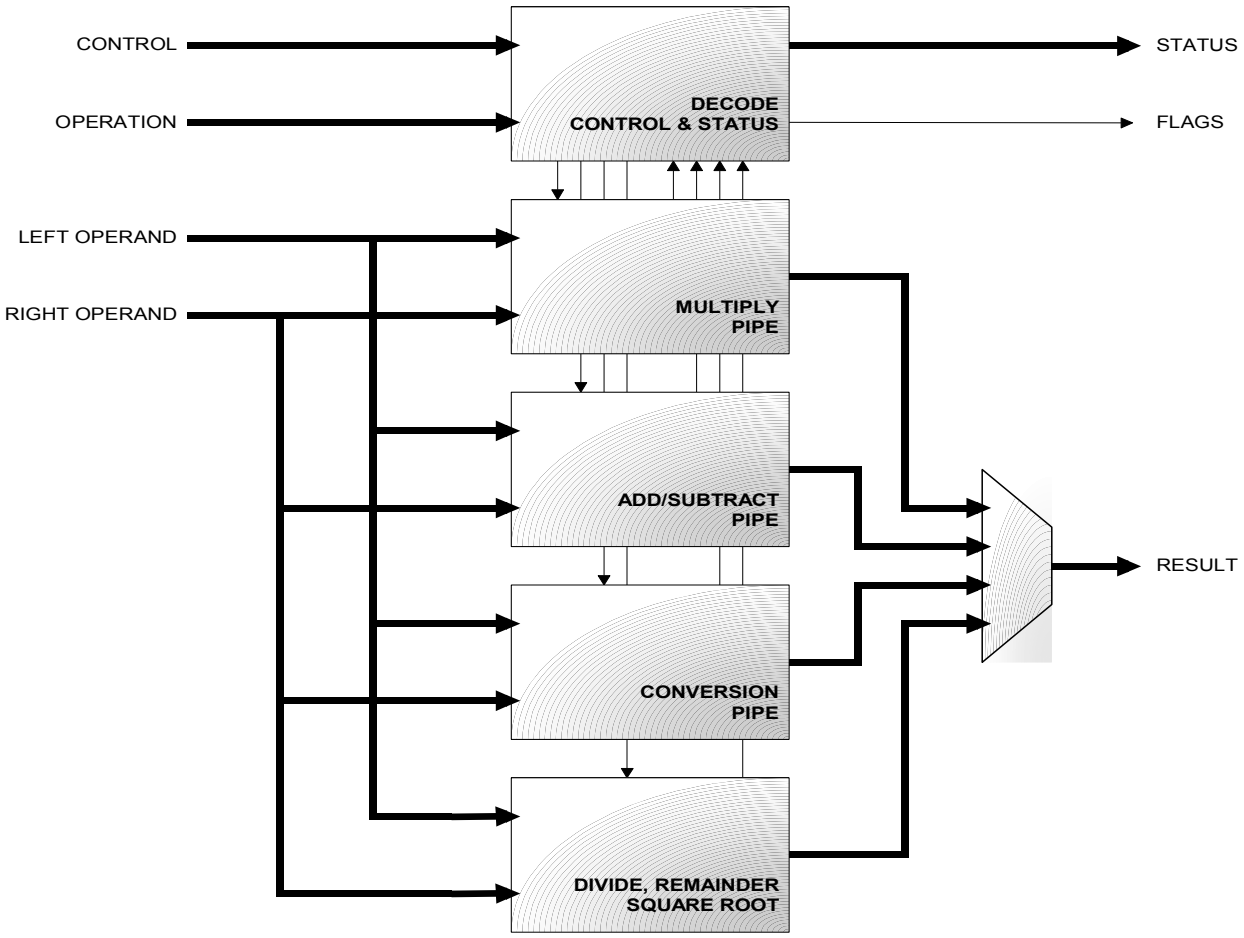


Block Diagram

FEATURES

- Fully Synthesizable RTL - Verilog
- User Selectable Precision
  - Single precision
  - Double precision
- IEEE 754 fast mode compliant
- Supports all IEEE rounding modes
- Extensible to run complex Math Functions
  - Addition of sequencer and ROM allows trigonometry and other complex functions
- C models available
- Fully Synchronous pipelines
  - Three stage pipeline
- Full enhanced IEEE 754 Compliance Suite

OVERVIEW

The A2FD is a fully synthesizable module implemented in Verilog RTL. It is a co-processor unit providing floating-point computation compliant with the ANSI/IEEE Std 754-1985, IEEE Standard for Binary Floating-Point Arithmetic (IEEE Standard). It is designed to provide high performance floating-point computation while minimizing die size and power. Pipelined, single-cycle throughput operation is available for all operations except Divide, Remainder and Square Root operations.

The Multiply and Add/Subtract Pipes maybe optionally be replaced with a full Fused-Multiply-Add Pipe that provides improved precision for complex operations.

The A2FD implements most of the formats and operations specified in the IEEE 754 standard directly in its hardware pipelines. The following operations are not supported directly by the A2FD, but can be supported with software support:

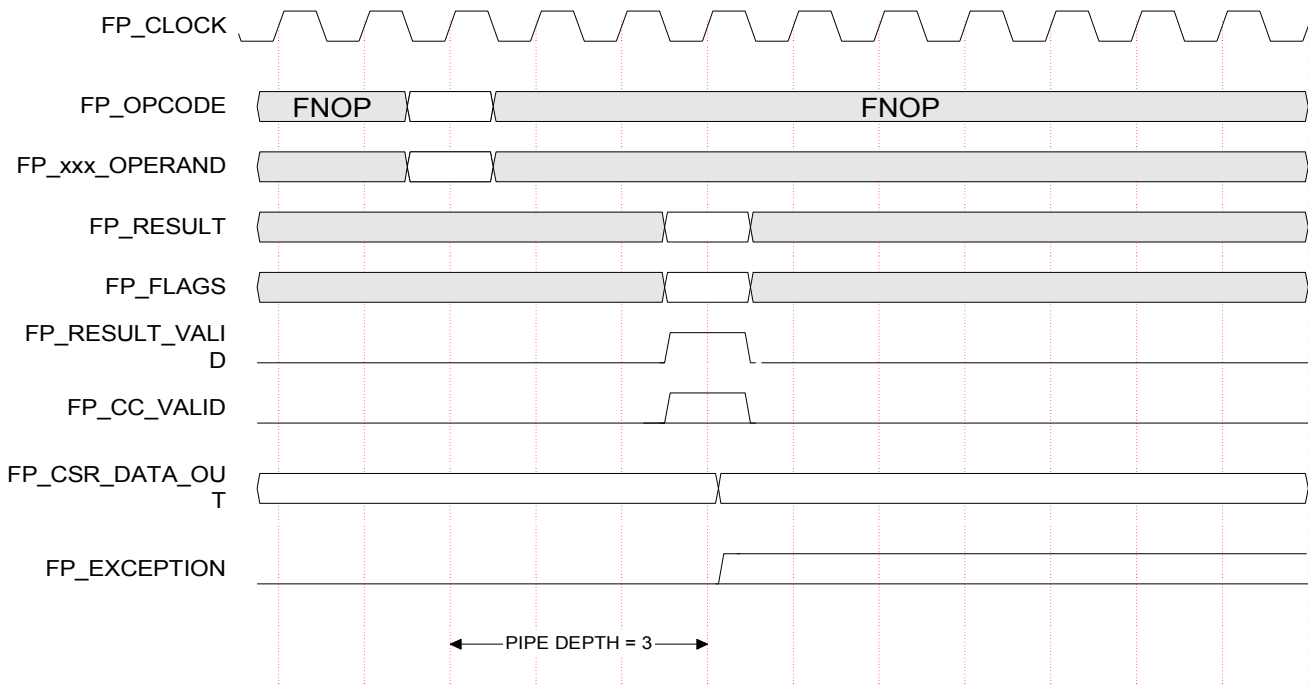
- Denormal numbers
- Underflow to denormal numbers
- Full NaN support
- BCD conversions

Control Register bits determine how the A2FD treats denormals and NaN operands, and underflow conditions. In Fast Mode, input denormal operands are treated as equivalent to positive zero. When an arithmetic operation generates an underflow result, the A2FD returns a zero instead (flush-to-zero mode). When an arithmetic instruction generates a NaN operand, the default NaN value is always generated. The A2FD does not propagate the lower mantissa bits of NaNs. In Compliance Mode, the A2FD will generate a software trap when it encounters a denormal input, generates an underflow value, or needs to generate a NaN result. This allows support software to complete the trapped operation in a way that is fully compliant with the IEEE standard.

Primary Functions	Secondary Functions	Conversion Functions
Add	Divide	Integer to Float
Subtract	Remainder	Float to Integer
Compare	Square Root	Round to Integer
Multiply	Modulus (Java)	Round to Nearest
Change Sign	Minimum	Round Up
Absolute Value	Maximum	Round Down
Absolute Difference	Clip to Magnitude	Round toward Zero

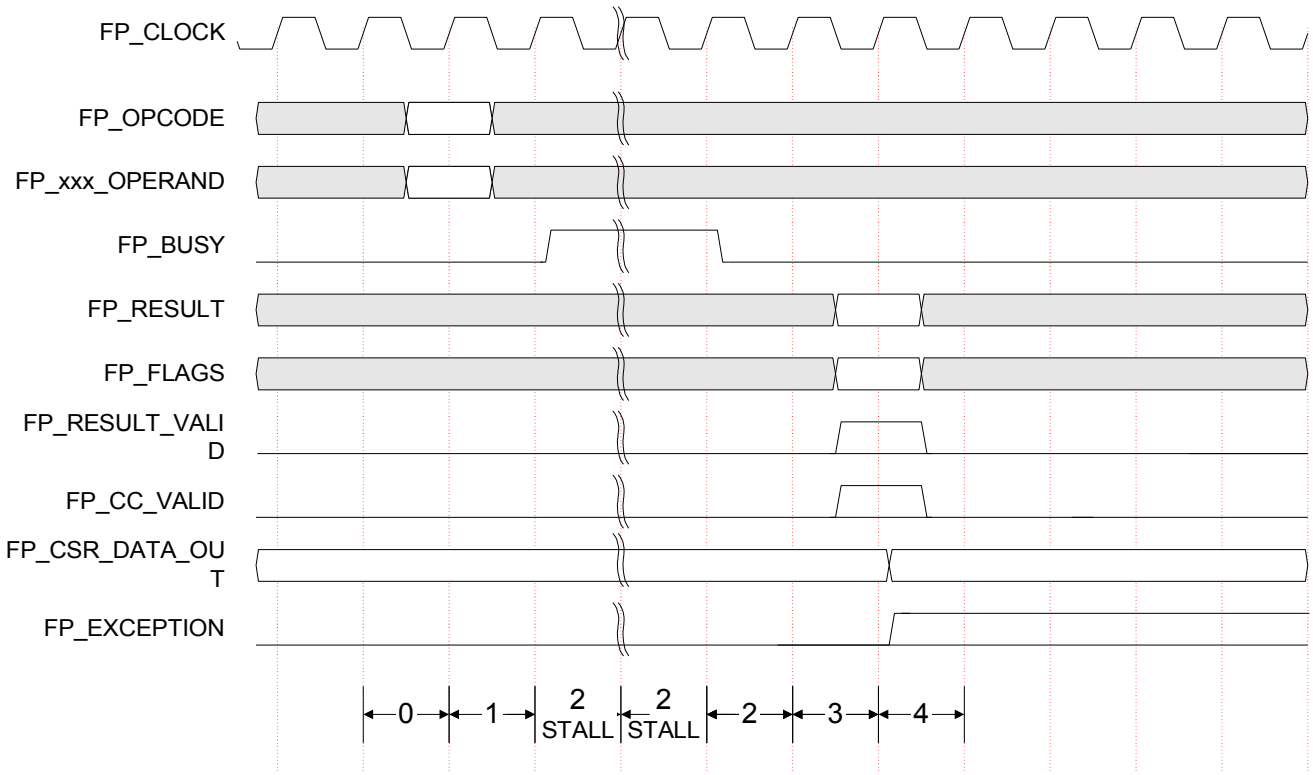
SIGNAL	I/O	SIZE	DESCRIPTION
CLK	I	1	System Clock
RESET	I	1	System Reset – configurable active level and synchronization
FP_LEFT_OPERAND	I	64	Left hand operand to FP binary operations
FP_RIGHT_OPERAND	I	64	Right hand operand to FP binary operations, operand for unary operations.
FP_OPCODE	I	6	Floating Point operation code. Ignored with either FP_STALL or FP_BUSY is asserted.
FP_STALL	I	1	Floating Point pipeline stall. When asserted, A2FD stalls its internal pipeline stalls and ignores FP_OPCODE.
FP_RESULT_VALID	O	1	Floating Point result is valid
FP_CC_VALID	O	1	Floating Point compare result is valid
FP_RESULT	O	64	Floating Point operation result. Valid when FP_RESULT_VALID is asserted and FP_CC_VALID is not asserted. Invalid and indeterminate otherwise.
FP_IF	O	1	Invalid Operation Flag. Indicates that the requested operation is invalid for the input operands. Valid when FP_RESULT_VALID is asserted.
FP_ZF	O	1	Divide by Zero Flag. Indicates that the instruction attempted a divide by 0. Valid when FP_RESULT_VALID is asserted.
FP_OF	O	1	Overflow Flag. Indicates that the result exceeds the exponent range of the destination format. Valid when FP_RESULT_VALID is asserted.
FP_UF	O	1	Underflow Flag. Indicates that the result is inexact and too small to be represented in the destination format. Valid when FP_RESULT_VALID is asserted.
FP_XF	O	1	Inexact Flag. Indicates that the rounded FP_RESULT is not exact. Valid when FP_RESULT_VALID is asserted.
FP_DF	O	1	Denormal Flag. Indicates that at least one of the input operands was a denormal number. Valid when FP_RESULT_VALID is asserted.
FP_UO	O	1	Compare result: unordered. Indicates that at least one of the input operands was a NAN. Valid when FP_CC_VALID is asserted.
FP_GT	O	1	Compare result: greater than. Indicates that neither operand was a NAN, and the LEFT operand was greater than the RIGHT operand. Valid when FP_CC_VALID is asserted.
FP_LT	O	1	Compare result: less than. Indicates that neither operand was a NAN, the LEFT operand was less than the RIGHT operand. Valid when FP_CC_VALID is asserted.
FP_BUSY	O	1	Floating Point busy status. Asserted when A2FD has internally stalled its pipeline during the execution of an instruction. FP_OPCODE is ignored when FP_BUSY is asserted.
FP_EXCEPTION	O	1	Floating Point exception flag. Asserted when any of the exception flags in the Status Register is unmasked by the exception mask in the Control Register.
FP_LOAD_CSR	I	1	Load enable for Control/Status Register. When asserted, FP_CSR_DATA_IN is loaded into the Control/Status register on the rising edge of CLK. FP_LOAD_CSR is not affected by FP_STALL or FP_BUSY. When FP_LOAD_CSR is asserted on the same cycle as a valid FP_OPCODE, then the data in FP_CSR_DATA_IN determines the rounding and precision modes of the instruction, and the modes in the current Control Register are ignored.
FP_CSR_DATA_IN	I	32	Control/Status Register load data.
FP_CSR_DATA_OUT	O	32	Control/Status Register contents.

The A2FD module operates on a single or multi-stage execution pipeline. Figure 2 shows the timing relationships between the main A2FD interface signals during the execution of a single instruction. In cycle 0, an instruction is issued to A2FD by asserting an opcode and operands to the A2FD inputs. The inputs are registered on the rising edge of CLK and the execution pipeline stages begin. The instruction completes "Pipe-Depth" cycles later when the A2FD module asserts FP_RESULT_VALID, drives the data result to FP_RESULT, and drives the flag results to the corresponding flag outputs (e.g. FP_XF). For compare instructions, A2FD also asserts FP_CC_VALID, signifying that the compare flags are valid (e.g. FP_LT). On the rising edge of CLK at the end of the same cycle, the Status Register is updated with the results of the completing instruction. The updated Status Register appears on FP_CSR_DATA_OUT in cycle ("Pipe-Depth+1). If the instruction sets an unmasked exception flag, then FP_EXCEPTION is also asserted in cycle ("Pipe-Depth+1).



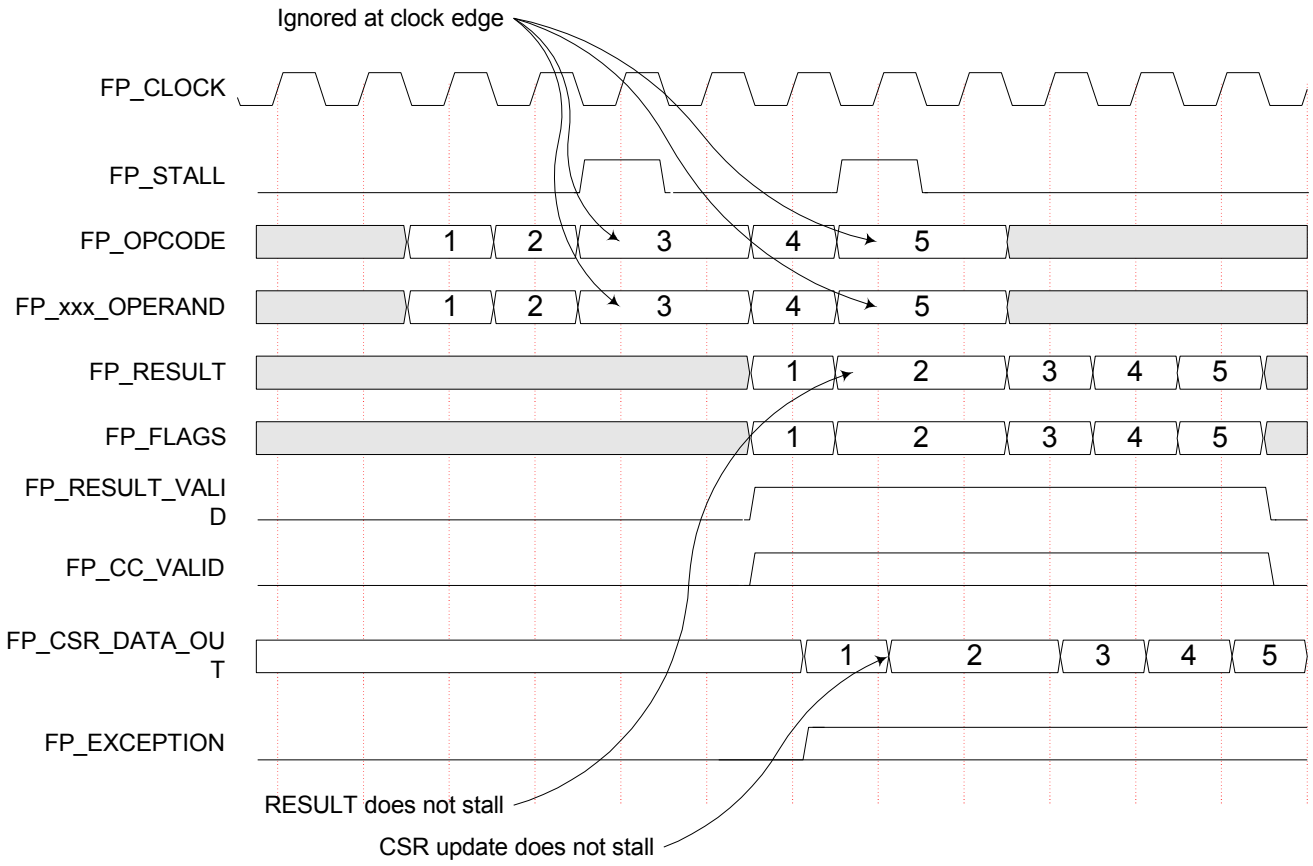
Basic Pipeline Operation

For certain instructions, the A2FD must stall the pipeline in order to complete a lengthy computation. When required, A2FD stalls the second stage of the pipeline by asserting the output signal FP_BUSY. When the computation is complete, FP_BUSY is de-asserted and the pipeline continues with stage 2 as shown in Figure 3 below. When FP_BUSY is asserted, A2FD ignores the FP_OPCODE input and does not start any new instructions.



Pipeline Stall for FP_BUSY

The A2FD pipeline can be stalled by either external or internal conditions. External logic can force A2FD to stall its pipeline by asserting FP_STALL at any clock boundary. Internally, A2FD stalls its pipeline when performing lengthy divide loop computations by asserting the output signal FP_BUSY. The two signals have the same effect on the A2FD pipeline. When either signal is asserted, A2FD ignores the FP_OPCODE input and does not start new instructions. Instructions already in progress are suspended at their current execution stage until the stall is removed. However, asserting FP_STALL does stop the divide hardware iterations that cause FP_BUSY to be asserted. If the iterations complete while FP_STALL is still asserted, then the A2FD maintains the results and continues the external stall.



Pipeline Stall for FP_STALL