Power Matters.

Microsemi

Designing with SmartFusion2 cSoCs
FPGA Fabric

Agenda

- SmartFusion2 Overview
- SmartFusion2 FPGA Fabric
- SmartFusion2 I/Os
- SmartFusion2 Power Management
- SmartFusion2 Development Tools
- SmartFusion2 Programming
- Debugging
- Hands-on Labs

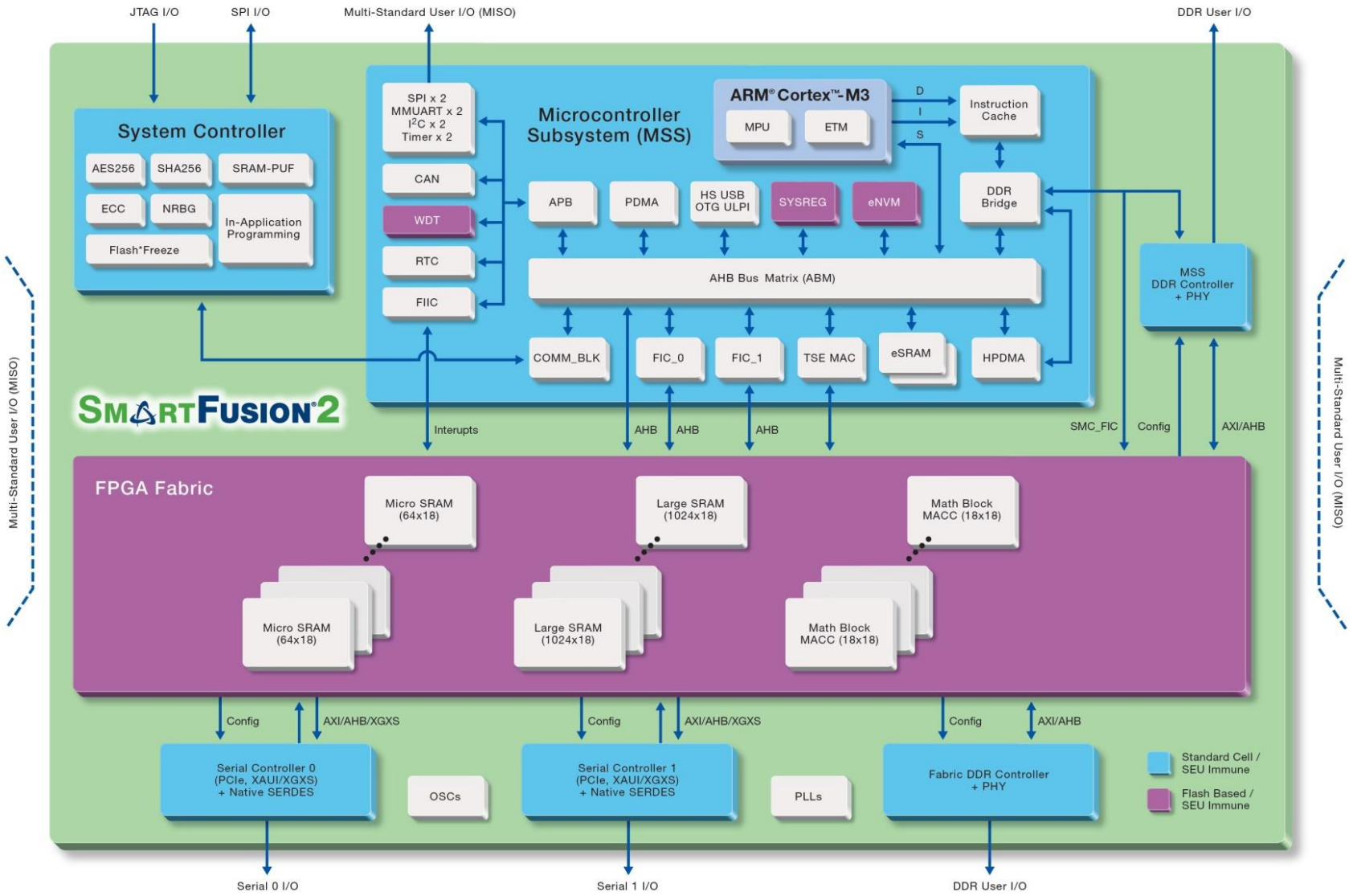
SmartFusion2 Customizable SoC

- 4th Generation Flash FPGA Architecture
  - 65 nm nonvolatile, low power flash process
- Highest Reliability
  - Zero FIT Flash FPGA Configuration
  - SECDED on ASIC SRAM Blocks
  - SEU immune latches on Instruction Cache and Buffers
- Most Secure FPGA
  - Design Security - DPA Hardened
  - Data Security AES256, SHA256
  - Random Number Generator
- Lowest Power FPGA
  - Low static and dynamic power
    - < 1.0 mW in flash-freeze mode
    - 10 mW static power during operation

SmartFusion2 Highlights

- SEU immune FPGA configuration
  - 4 input LUTs with carry chains and DFF
  - Embedded SRAMS 18 Kbit and 1 Kbit
  - Fast Math Blocks (18x18 Multiplier and 44-Bit Accumulator)
- 166 MHz ARM Cortex-M3 microcontroller
  - 8 KB Instruction Cache, On Chip NVM and SRAM
  - Extensive peripherals CAN, TSE, USB
  - Embedded Trace Macrocell for debug
- High speed serial interfaces
  - 5Gbps SERDES, PCIe, XAUI / XGXS+ Native SERDES
- High Speed Memory Interfaces
  - 667 Mbps DDR2/3 controllers with SECDED protection

SmartFusion2 Architecture



SmartFusion2 Family

	Features	M2S005	M2S010	M2S025	M2S050	M2S060	M2S090	M2S150
Logic / DSP	Maximum Logic Elements (4LUT + DFF)*	6,060	12,084	27,696	56,340	56,340	86,316	146,124
	Math Blocks (18x18)	11	22	34	72	72	84	240
	PLLs and CCCs	2		6				8
Security	AES256, SHA256, RNG	1 each						
	ECC, PUF	-				1 each		
MSS	Cortex-M3 + Instruction cache	Yes						
	eNVM (Kbytes)	128	256				512	
	eSRAM (Kbytes)	64						
	eSRAM (Kbytes) Non-SECDED	80						
	TSE, USB, CAN, Watchdog, RTC	1 each						
	SPI, I2C, Multi-Mode UART, Timer	2 each						
Fabric Memory	LSRAM 18K Blocks	10	21	31	69	69	109	236
	uSRAM 1K Blocks	11	22	34	72	72	112	240
	Total RAM (K bits)	191	400	592	1314	1314	2074	4488
High Speed	DDR Controllers (Count x Width)	1x18			2x36	1x18		2x36
	SERDES Lanes	0	4		8	4		16
	PCIe End Points	0	1		2			4
User I/Os	MSIO (3.3V)	115	123	157	139	271	309	292
	MSIOD (2.5V)	28	40	40	62	40	40	106
	DDRIO (2.5V)	66	70	70	176	76	76	176
	Total User I/O	209	233	267	377	387	425	574

* Total logic may vary based on utilization of DSP and memories in the design.

Package Options

I/Os Per Package

	Package Options																			
Package Type	FCS(G)325		VF(G)256		FCS(G)536		VF(G)400		FCV(G)484		TQ(G)144		FG(G)484		FG(G)676		FG(G)896		FC(G)1152	
Pitch (mm)	0.5		0.8		0.5		0.8				0.5		1.0		1.0		1.0		1.0	
Length x Width (mm)	11x11		14x14		16x16		17x17		19x19		20x20		23x23		27x27		31x31		35x35	
Device	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes	I/O	Lanes
M2S005 (S)	–	–	161	–	–	–	171	–	–	–	84	–	209	–	–	–	–	–	–	–
M2S010 (S/T/TS)	–	–	138	2	–	–	195	4	–	–	84	–	233	4	–	–	–	–	–	–
M2S025 (T/TS)	180	2	138	2	–	–	207	4	–	–	–	–	267	4	–	–	–	–	–	–
M2S050 (T/TS)	200	2	–	–	–	–	207	4	–	–	–	–	267	4	–	–	377	8	–	–
M2S060 (T/TS)	200	2	–	–	–	–	207	4	–	–	–	–	267	4	387	4	–	–	–	–
M2S090 (T/TS)	180	4	–	–	–	–	–	–	–	–	–	–	267	4	425	4	–	–	–	–
M2S150 (T/TS)	–	–	–	–	293	4	–	–	248	4	–	–	–	–	–	–	–	–	574	16

Note:

090 FCSG325 is 11x13.5 package dimension

Highlighted devices can migrate vertically in the same package

SmartFusion2/IGLOO2 Product Offering

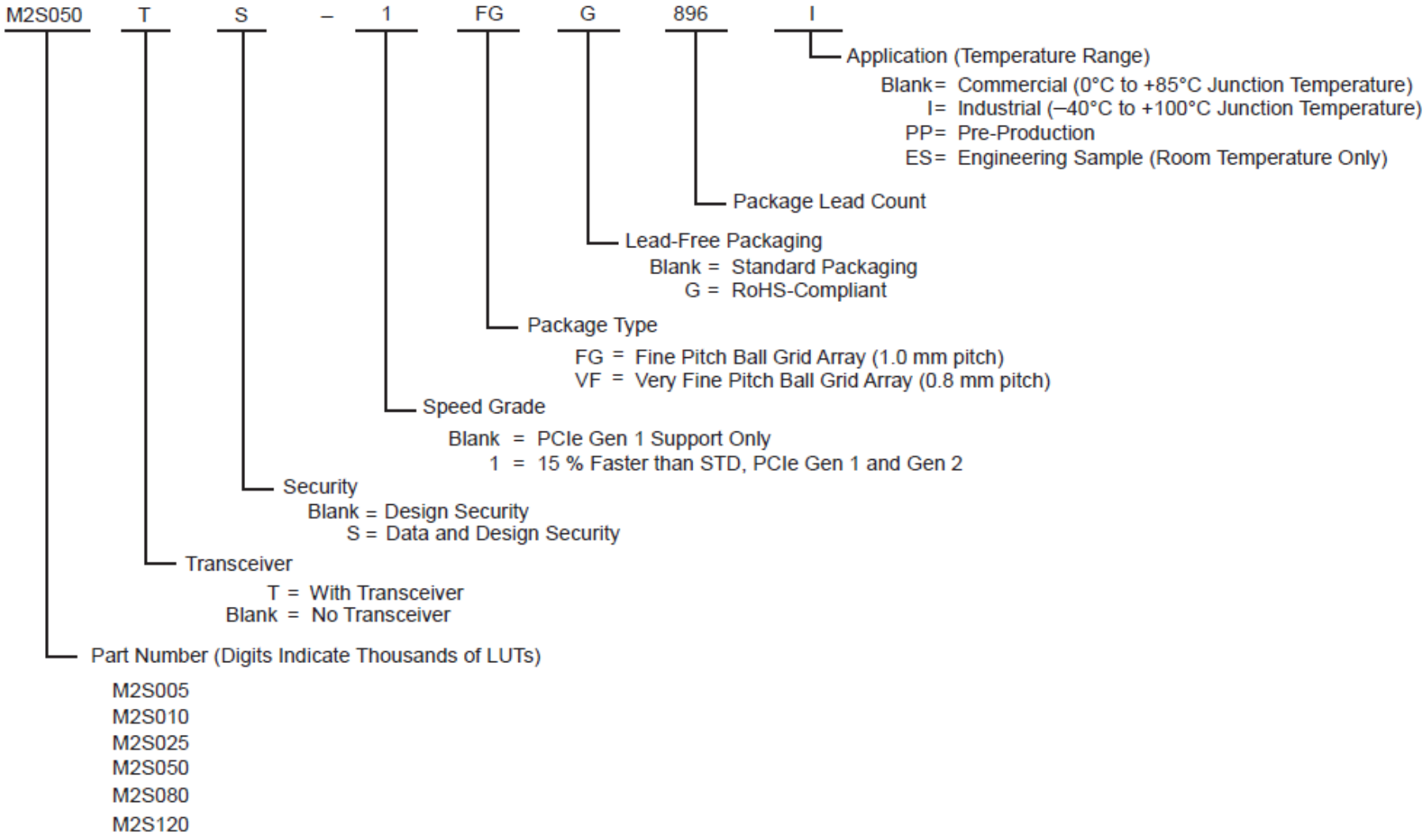
	IGLOO2 Data Security "S" devices offering (all speed grades, C and I temp)									
Density	FCS325	VF256	FCS536	VF400	FCV484	TQ144	FG484	FG676	FG896	FC1152
M2GL005		S		S		S	S			
M2GL010		TS		TS		S	TS			
M2GL025	TS	TS		TS			TS			
M2GL050	TS			TS			TS		TS	
M2GL090	TS						TS	TS		
M2GL150			TS		TS					TS

	SmartFusion2 Data Security "S" devices offering (all speed grades, C and I temp)									
Density	FCS325	VF256	FCS536	VF400	FCV484	TQ144	FG484	FG676	FG896	FC1152
M2S005		S		S		S	S			
M2S010		TS		TS		S	TS			
M2S025	TS	TS		TS			TS			
M2S050	TS			TS			TS		TS	
M2S090	TS						TS	TS		
M2S150			TS		TS					TS

Military Temperature package available

- *Max SERDES speed for Military Temperature devices is 3.125 Gbps.*
- *PCI Express endpoints are limited to Gen1 compliance.*

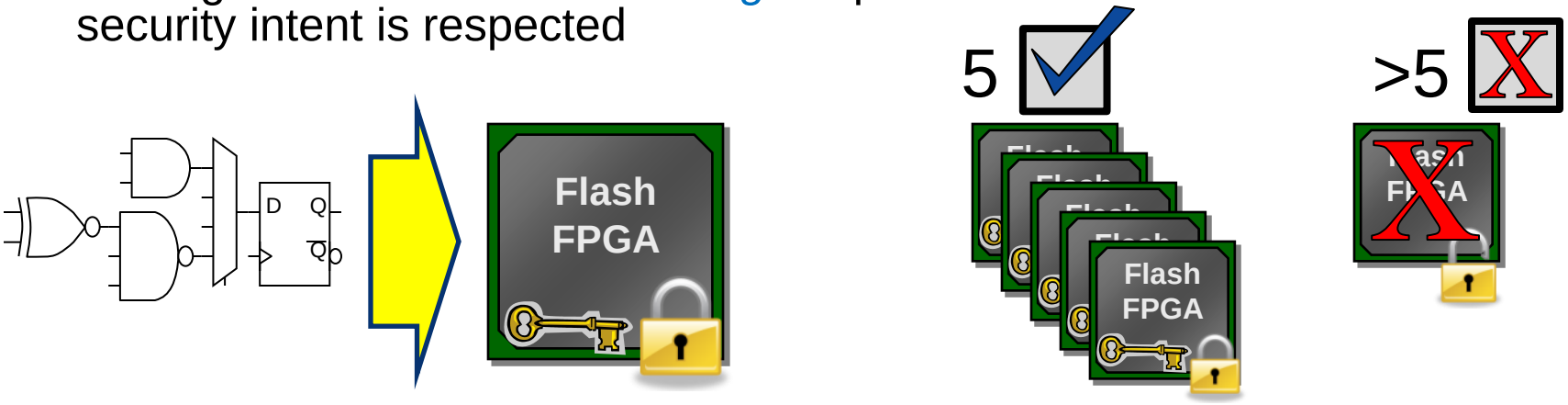
SmartFusion2 Part Numbers



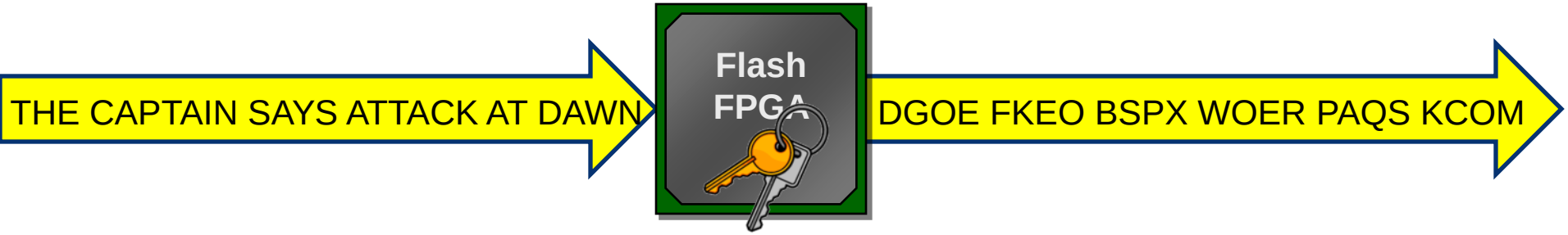
Most Secure, Highest Reliability, Lowest power

Security: Design Security vs. Data Security

- Design Security
  - Making sure that the *FPGA Design* is protected and the IP owner's security intent is respected



- Data Security
  - The *Application* programmed into the device meets its security objectives (authenticity, confidentiality, integrity, etc.)



Security: Why is FPGA Design Security Important?

- Design Security
  - Cloning
    - Someone copies your design without even necessarily having to understand how it works
  - Overbuilding
    - Your contract manufacturer fills your order… then makes a few for himself. After all, he has all the data!
  - Reverse engineering
    - Someone figures out how your design works, then uses or improves on what he learned
  - Counterfeiting
    - Illegal use of your brand name on a work-alike or cloned product
  - Tampering
    - Changing the design for malicious intent
- Data Security
  - The data being managed by the device stays secure
  - Without design/device security, it is virtually impossible to provide good data security

Security: Programmable Design Security

- Built-in Design Security on all Devices
  - Protection against tampering, cloning, overbuilding, reverse engineering and counterfeiting
  - No design or manufacturing overhead to build secure devices
  - Supply-chain assurance with digital Certificate of Conformance
- Design Protection
  - Anti-tamper detection with active zeroization
  - Bitstream is always encrypted with AES-256
  - Secure programming with SHA-256 bitstream authentication
  - On-demand BIST for all non-volatile memories
- Security Key Protection
  - Cryptographic Research Incorporated (CRI) DPA resistant technology
  - Intrinsic-ID physically un-clonable function (PUF) for state of the art key protection
  - Non-deterministic random bit generator (NRBG) for key generation
  - Non-volatile key storage in encrypted form

State of the art security enables root-of-trust applications

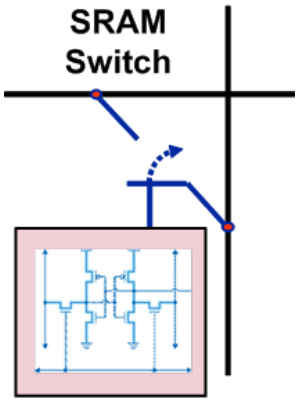
Security: Breakthrough Data Security

- Security Processing Accelerators for Advanced Cryptographic Applications
  - Cryptographic services built into the device accessible to users
  - AES-256, SHA-256, HMAC
  - 384 bit Elliptical Curve Cryptographic (ECC) Engine
  - Pseudo-PUF challenge-response service
  - Non-Deterministic Random Bit Generator with DPA countermeasures
  - CRI-patented key-tree protocol-level construct for DPA-safe designs
  - Hardware firewalls available in the ARM AHB bus matrix

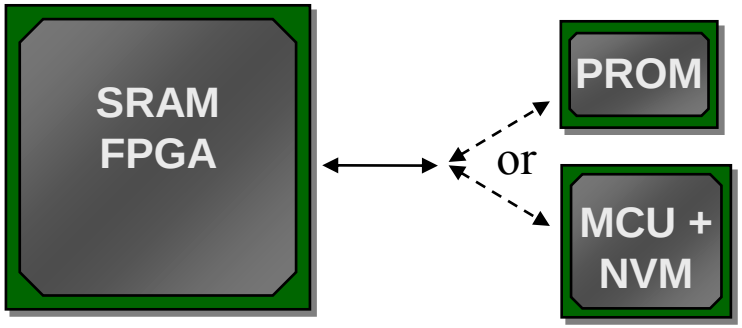
- Protection of Application-level Cryptographic Keys
  - Cryptographic Research Incorporated (CRI) DPA resistant technology
  - Intrinsic-ID's physically unclonable function (PUF) technology

Radically transforms the usefulness of FPGAs in security applications

FPGA Technologies Compared



Reprogrammable and volatile

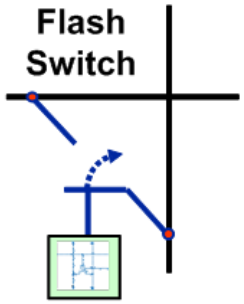


- FPGA reloaded after every power cycle
- Bitstream exposed at board level
- Application keys must be stored externally
- Bitstream keys (if encryption is even available) may also be volatile, requiring a long-term battery to maintain them

Single Chip

Nonvolatile Keys

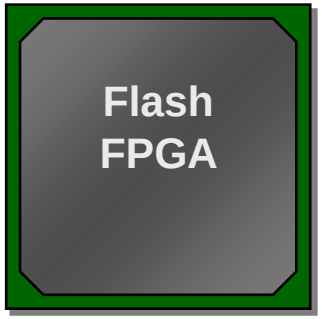
Instant-on



Reprogrammable and nonvolatile

Embedded Nonvolatile Memory

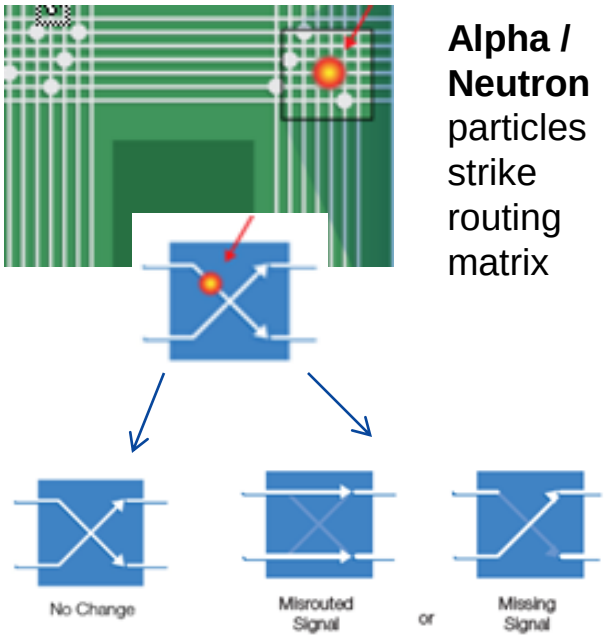
Embedded ARM



- FPGA NOT reloaded after every power cycle
- Bitstream NOT exposed
- All keys, for bitstream AND (optionally) for end application, are NON-volatile and safely stored long-term inside the device

Reliability: The Most Reliable FPGA in the Industry

- Flash FPGA Fabric
  - SEU immune Zero FIT rate configuration
- All SoC Memory SEU Tolerant
  - Thorough built in error detection and correction techniques
    - Cortex-M3 Embedded Scratch Pad Memory, Ethernet, CAN and USB Buffers, PCIe FIFOs
  - Or SEU tolerant implementation
    - DDR Bridges (MSS, MDDR, FDDR), Instruction Cache, MMUART and SPI FIFOs



Alpha / Neutron particles strike routing matrix

Flash FPGA
No Error

SRAM FPGA
Functional Failure

Reliability for safety critical or mission critical systems

Reliability: Flash vs. SRAM Configuration SEU Immunity

FPGA	Technology	Equivalent Functional Failure – FIT Rates per Device			
		Ground-Level Applications		Commercial Aviation	Military Aviation
		Sea Level	5,000 Ft	30,000 Ft	60,000 Ft
Microsemi SmartFusion2	65nm Flash	No Failures Detected	No Failures Detected	No Failures Detected	No Failures Detected
Xilinx XC6SLX25	45nm SRAM	870 FITs	3,000 FITs		
Xilinx XC6SLX45	45nm SRAM	1,600 FIT	5,500 FITs		
Altera EP3C25	65nm SRAM	580 FITs	1,900 FITs		
Altera EP3C55	65nm SRAM	1,500 FITs	5,000 FITs		
Altera EP3C120	65nm SRAM	2,900 FITs	9,700 FITs		
Altera EPCGX50	60nm SRAM	2,500 FITs	8,200 FITs		
Lattice ECP4-50	65nm SRAM	1,400 FITs	4,600 FITs		
Lattice ECP4-130	65nm SRAM	3,700 FITs	12,000 FITs		

Very High SEU Failure Rates at High Altitudes

-- SmartFusion2 based on test chip data
-- Xilinx data – 136 FIT/Mb reported in Xilinx Reliability Data
-- Altera, Lattice data – estimated at 100 FIT/Mb

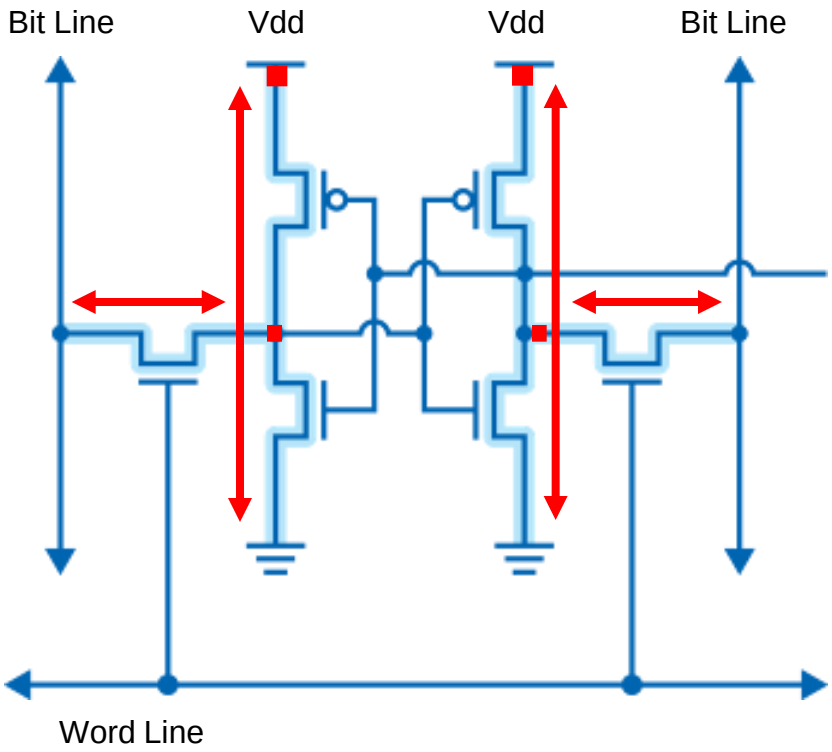
FIT = number of errors in 10^9 hours

*Acceptable FIT rates for commercial applications are **< 100***

*Acceptable FIT rates for high-reliability applications are **< 20***

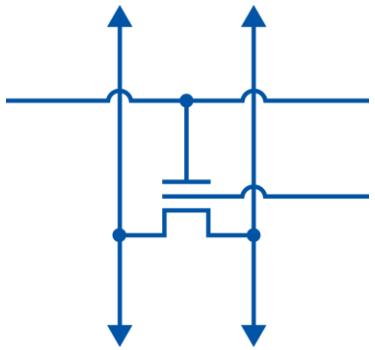
Low Power: Flash FPGA Advantage

Typical Competitors SRAM Cell



- Substantial Leakage per Cell
- Millions of Configuration Cells
- High Static Current

Microsemi Flash Cell



- 1000x lower leakage per cell
- Ultra Low Static Current

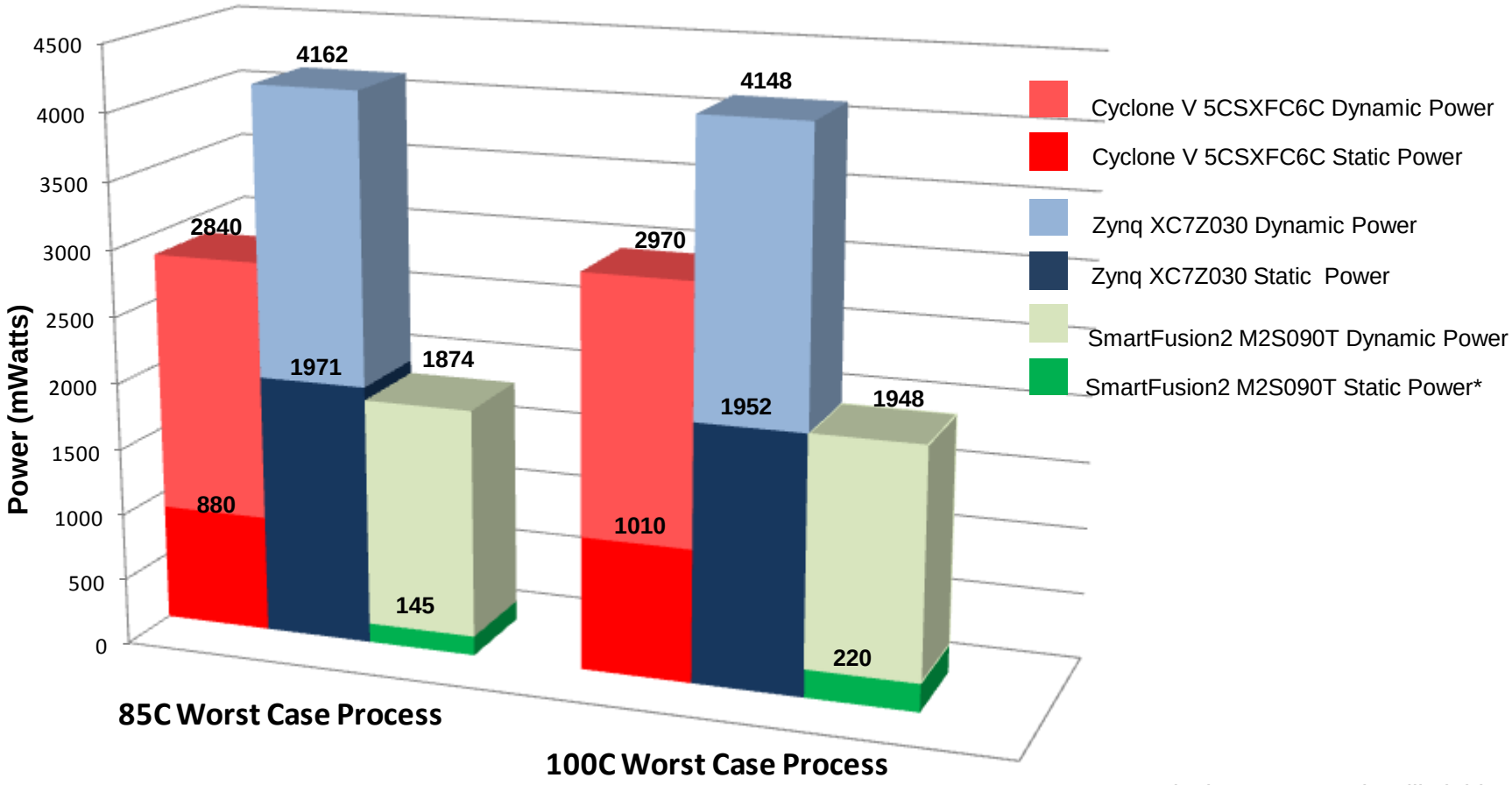
Flash FPGAs inherently lower power

Low Power: Lowest Power FPGA

- Flash FPGAs
  - Lowest power without sacrificing performance

- Flash*Freeze Ultra Low Power Mode 1mW

Results – SoC Devices

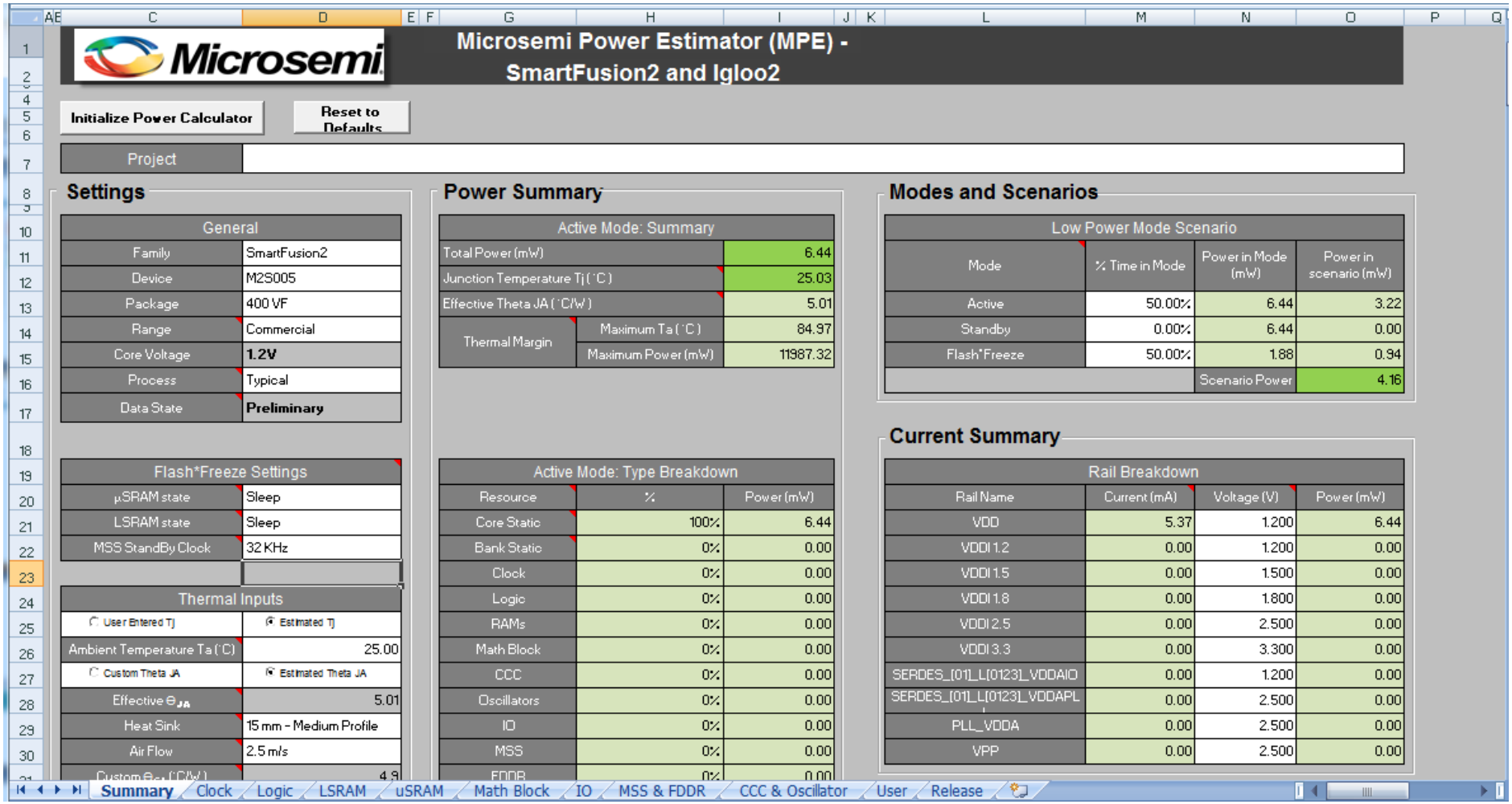


SmartFusion2 Delivers Up To

- ***10X Lower Static Power***
- ***50% Lower Total Power***

Note: Flash*Freeze mode will yield larger differences

Power Estimation Spreadsheet



Microsemi Power Estimator (MPE) - SmartFusion2 and Igloo2

Initialize Power Calculator | Reset to Defaults

Project

Settings

General	
Family	SmartFusion2
Device	M2S005
Package	400 VF
Range	Commercial
Core Voltage	1.2V
Process	Typical
Data State	Preliminary

Flash*Freeze Settings	
µSRAM state	Sleep
LSRAM state	Sleep
MSS StandBy Clock	32 KHz

Thermal Inputs	
User Entered Tj	Estimated Tj
Ambient Temperature Ta (°C)	25.00
Custom Theta JA	Estimated Theta JA
Effective Θ_{JA}	5.01
Heat Sink	15 mm - Medium Profile
Air Flow	2.5 m/s
Custom Θ_{JC} (°C/W)	4.9

Power Summary

Active Mode: Summary		
Total Power (mW)		6.44
Junction Temperature Tj (°C)		25.03
Effective Theta JA (°C/W)		5.01
Thermal Margin	Maximum Ta (°C)	84.97
	Maximum Power (mW)	11987.32

Active Mode: Type Breakdown		
Resource	%	Power (mW)
Core Static	100%	6.44
Bank Static	0%	0.00
Clock	0%	0.00
Logic	0%	0.00
RAMs	0%	0.00
Math Block	0%	0.00
CCC	0%	0.00
Oscillators	0%	0.00
IO	0%	0.00
MSS	0%	0.00
FDDR	0%	0.00

Modes and Scenarios

Low Power Mode Scenario			
Mode	% Time in Mode	Power in Mode (mW)	Power in scenario (mW)
Active	50.00%	6.44	3.22
Standby	0.00%	6.44	0.00
Flash*Freeze	50.00%	1.88	0.94
		Scenario Power	4.16

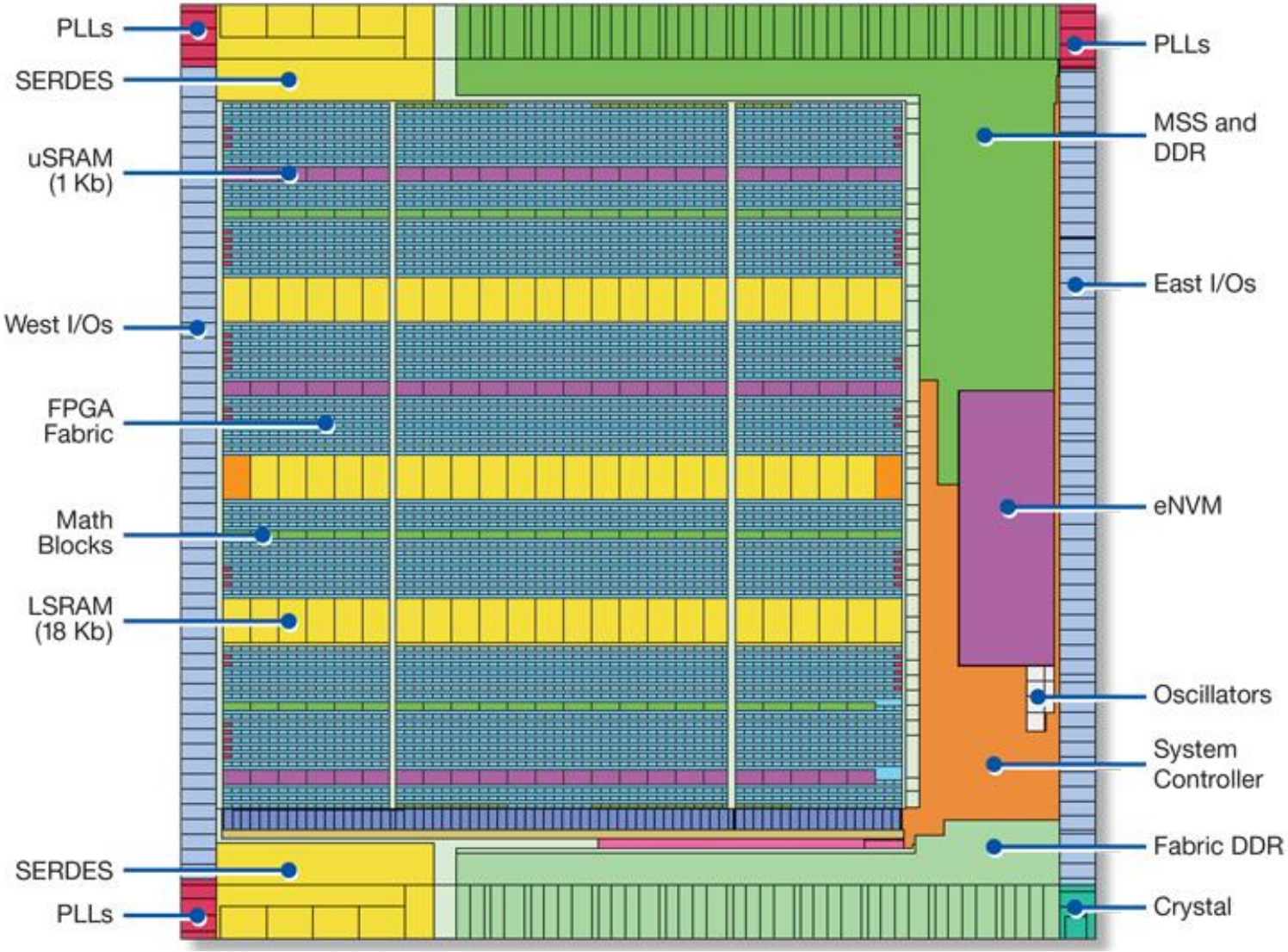
Current Summary

Rail Breakdown			
Rail Name	Current (mA)	Voltage (V)	Power (mW)
VDD	5.37	1.200	6.44
VDDI 1.2	0.00	1.200	0.00
VDDI 1.5	0.00	1.500	0.00
VDDI 1.8	0.00	1.800	0.00
VDDI 2.5	0.00	2.500	0.00
VDDI 3.3	0.00	3.300	0.00
SERDES_[01]_L[0123]_VDDAIO	0.00	1.200	0.00
SERDES_[01]_L[0123]_VDDAPL	0.00	2.500	0.00
PLL_VDDA	0.00	2.500	0.00
VPP	0.00	2.500	0.00

Summary | Clock | Logic | LSRAM | uSRAM | Math Block | IO | MSS & FDDR | CCC & Oscillator | User | Release

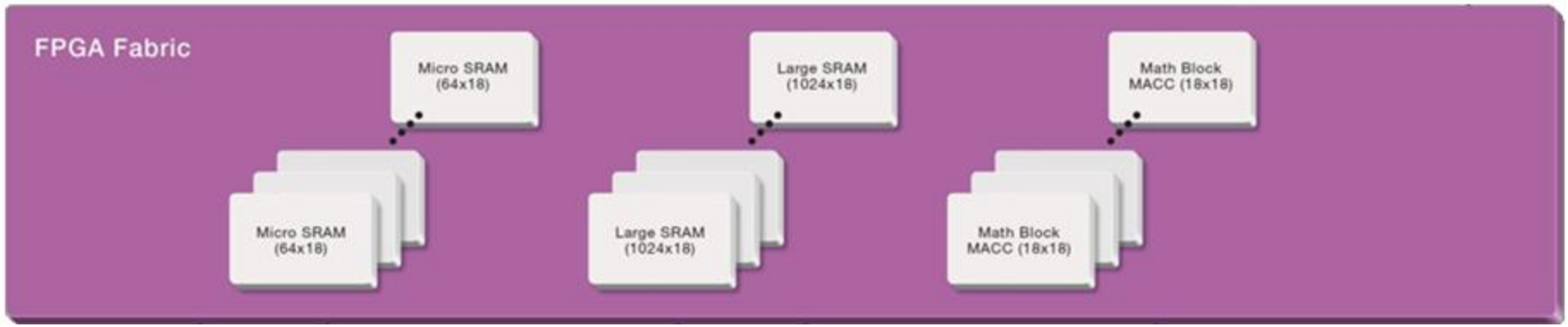
SmartFusion2 Architecture

SmartFusion2 Device Layout



SmartFusion2 FPGA Fabric

FPGA Fabric



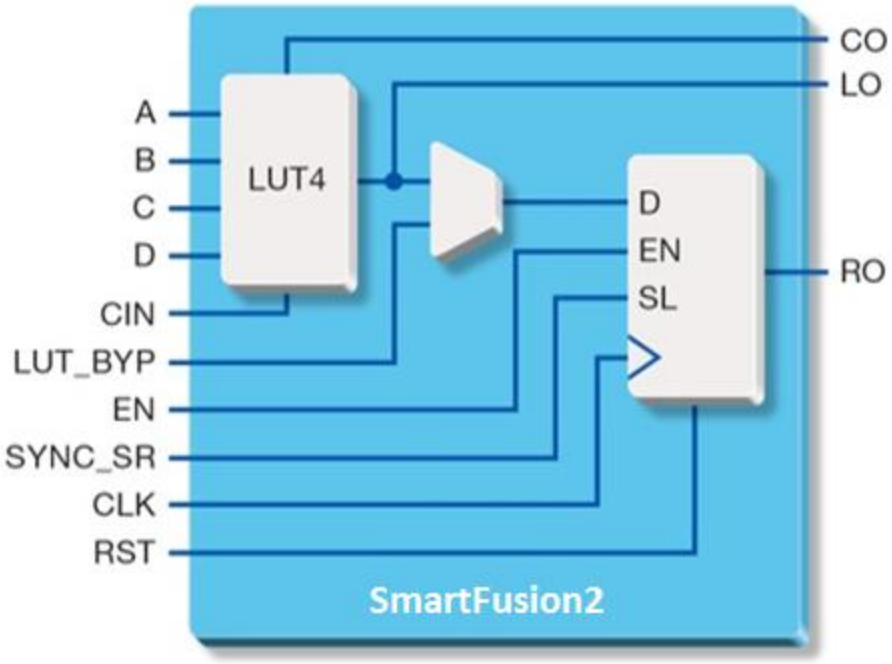
	Features	M2S005	M2S010	M2S025	M2S050	M2S060	M2S090	M2S150
FPGA	Maximum Logic Modules (LUT + DFF)	6,060	12,084	27,696	56,340	56,340	86,316	146,124
	LSRAM 18K Blocks	10	21	31	69	69	109	236
	uSRAM1K Blocks	11	22	34	72	72	112	240
	Total RAM (Kbits)	191	400	592	1,314	1,314	2,074	4,488
	Math Blocks	11	22	34	72	72	84	240
	PLLs and CCCs	2		6				8

Logic Module count includes shadow logic

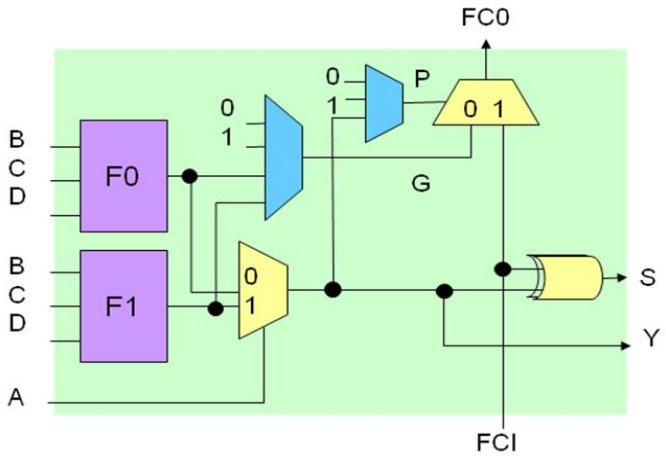
FPGA Fabric Overview

- Efficient 4-Input LUT architecture for lowest power
  - Carry chains for high performance
  - DFF
- Up to 236 Blocks of Dual-Port 18 Kbit LSRAM (Large SRAM)
- Up to 240 Blocks of Three-Port 1 Kbit uSRAM (Micro SRAM)
- High Performance DSP Signal Processing
  - Up to 240 Math Blocks with 18x18 Multiplier / 44-Bit Accumulator

Logic Element

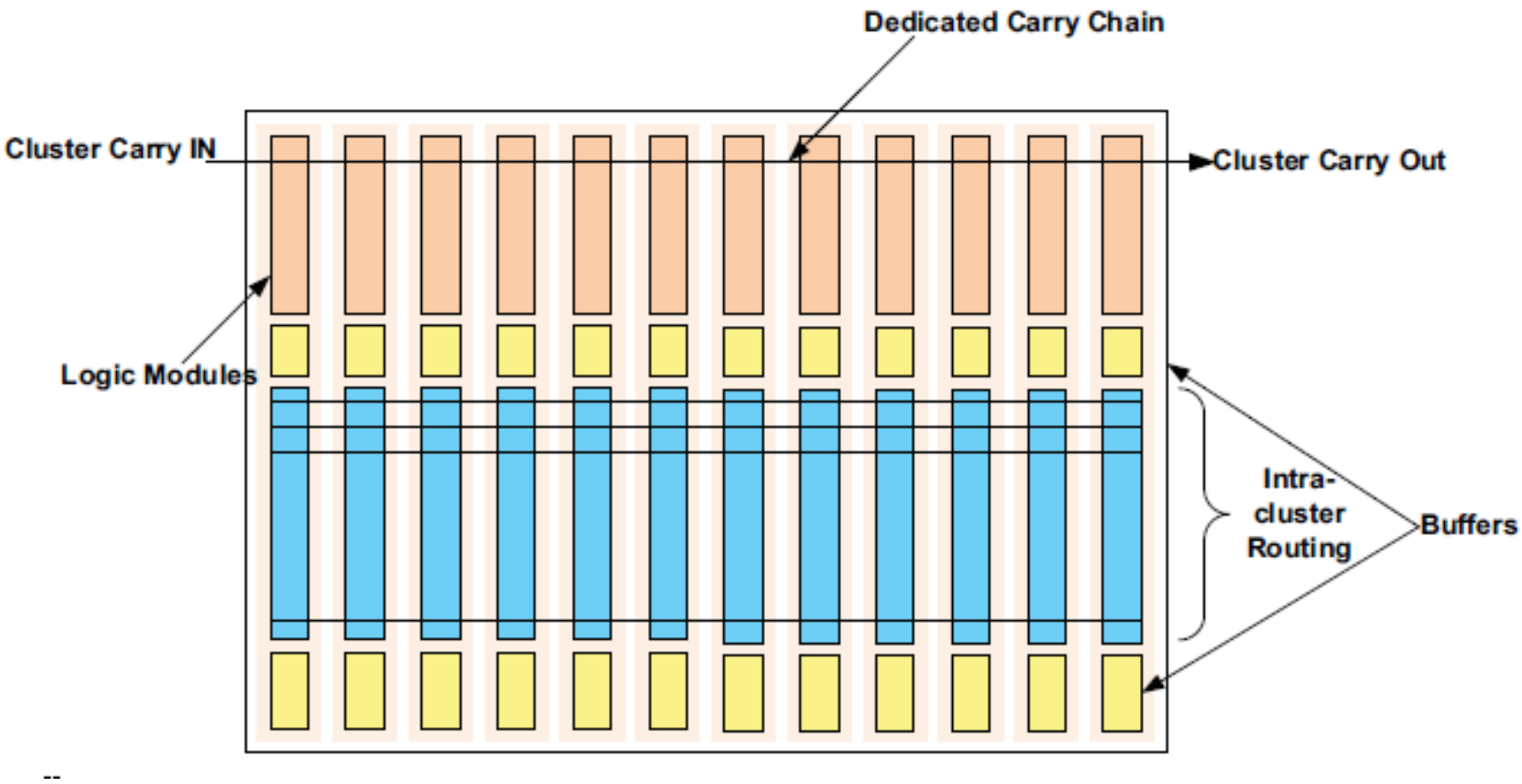


- A fully permutable 4-input LUT
- A separate flip-flop which can be used independently from the LUT
  - Configurable as a register or a latch
  - Asynchronous and synchronous load and clock enable inputs



- A dedicated carry chain based on the carry look-ahead technique

Logic Cluster



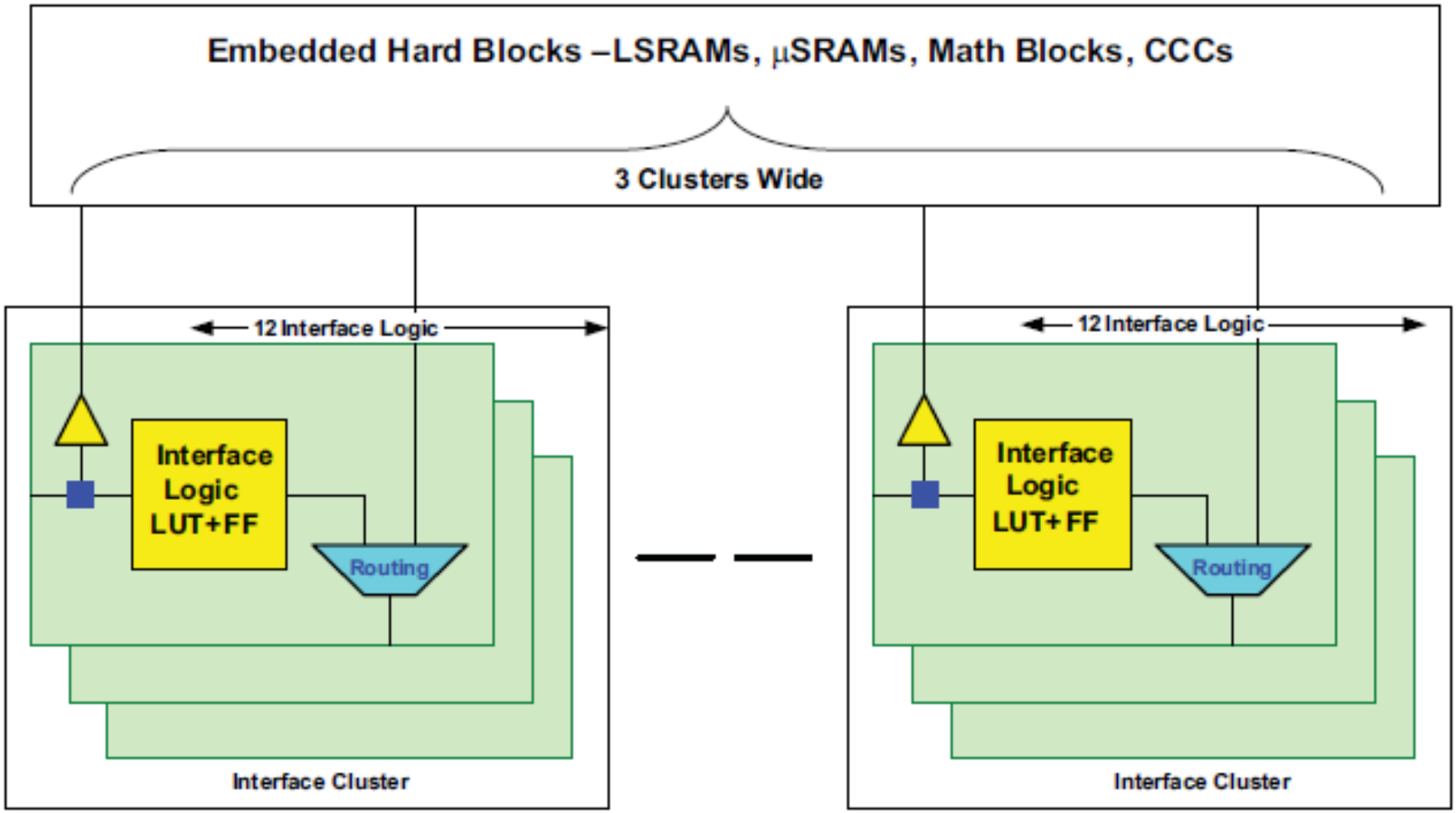
Interface Logic Element - What is it?

- Standard Logic Element with no Fast Carry Chain
- Used to interface user logic to embedded IP in the fabric
  - Math Blocks
  - Large RAMs
  - Micro RAMs

  Interface Logic is consumed when these IP's are used
  Math blocks have built in registers

- When any of the above block is not used, associated Shadow Logic can be used as any other normal fabric logic for the design
  - **Hence, LE count in a given device will become effectively more**

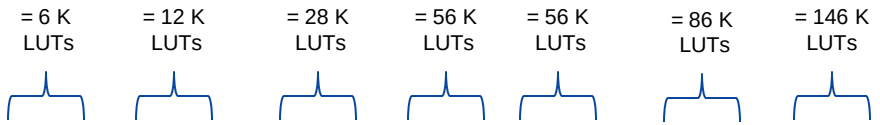
Interface Cluster



3 Interface Logic Clusters used per Embedded Hard Block

SmartFusion2/IGLOO2 Shadow Logic Count

Spare Logic Resource



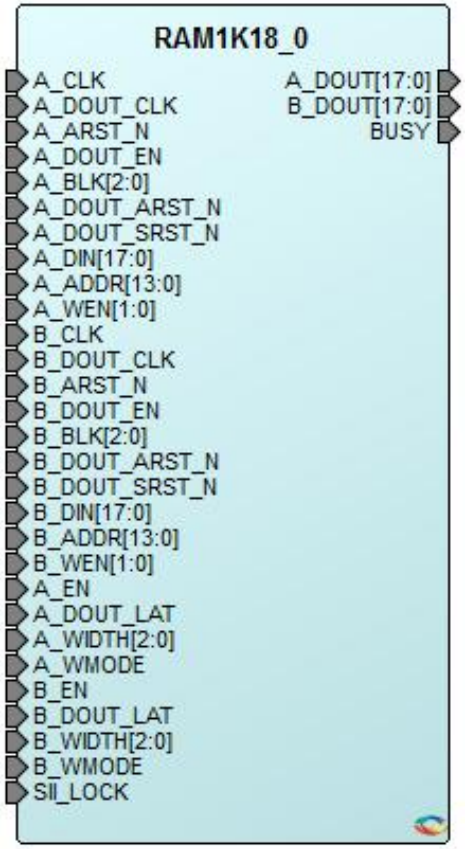
Features	M2S005 M2GL005	M2S010 M2GL010	M2S025 M2GL025	M2S050 M2GL050	M2S060 M2GL060	M2S090 M2GL090	M2S150 M2GL150
Logic Elements (4-Input LUT with DFF and FC)	4,908	9,744	24,132	48,672	48,672	75,336	120,348
Shadow Logic Elements (4-input LUT with DFF)	1,152	2,340	3,564	7,668	7,668	10,980	25,776
Maximum Logic Modules (LUT + DFF)	6,060	12,084	27,696	56,340	56,340	86,316	146,124
Math Blocks (18x18)	11	22	34	72	72	84	240
LSRAM 18K Blocks	10	21	31	69	69	109	236
uSRAM1K Blocks	11	22	34	72	72	112	240

36 Logic Elements Per Hard IP Instance
uRAM, LSRAM, Math

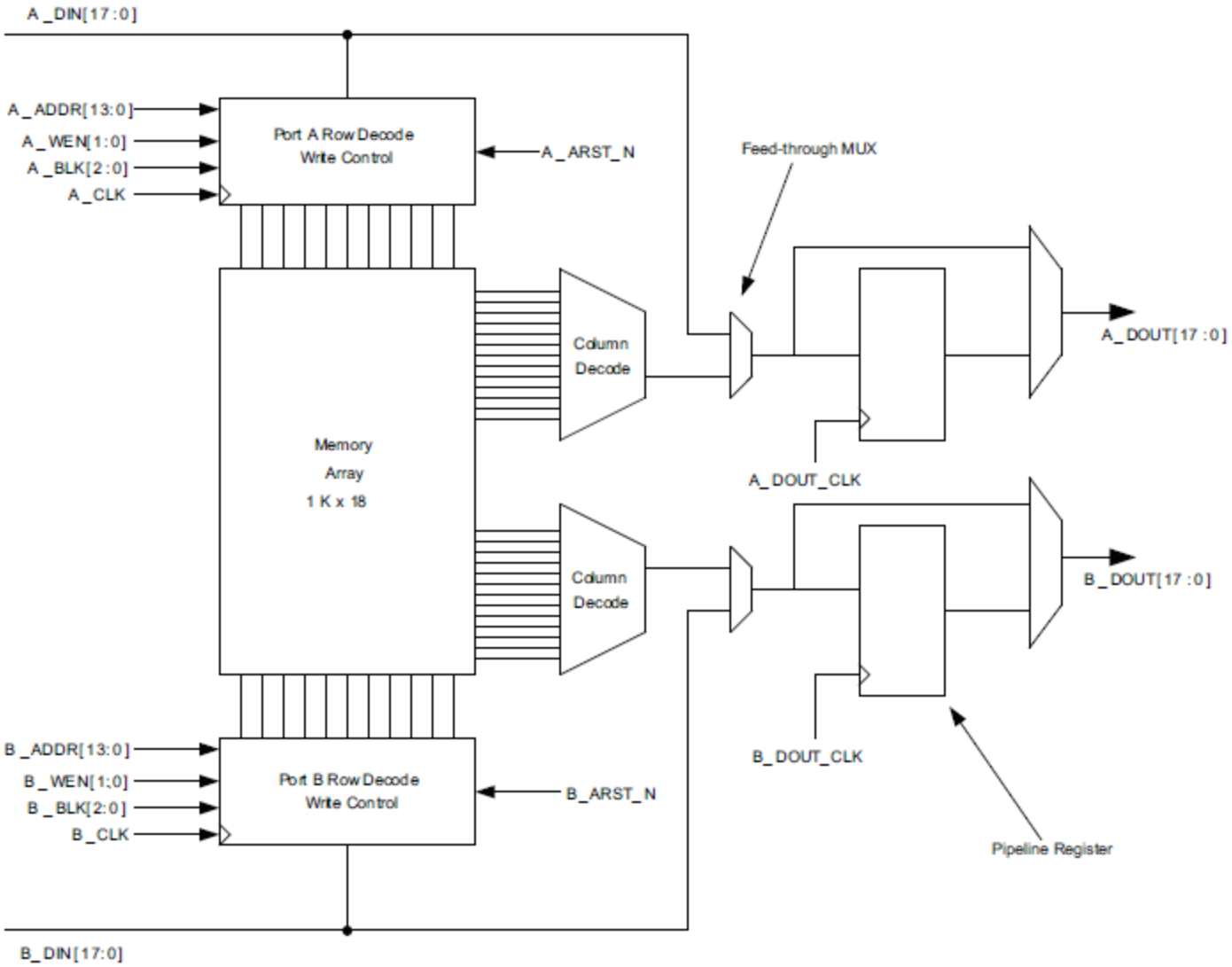
Fabric RAM Blocks

FPGA Memory Blocks – Large SRAM (LSRAM)

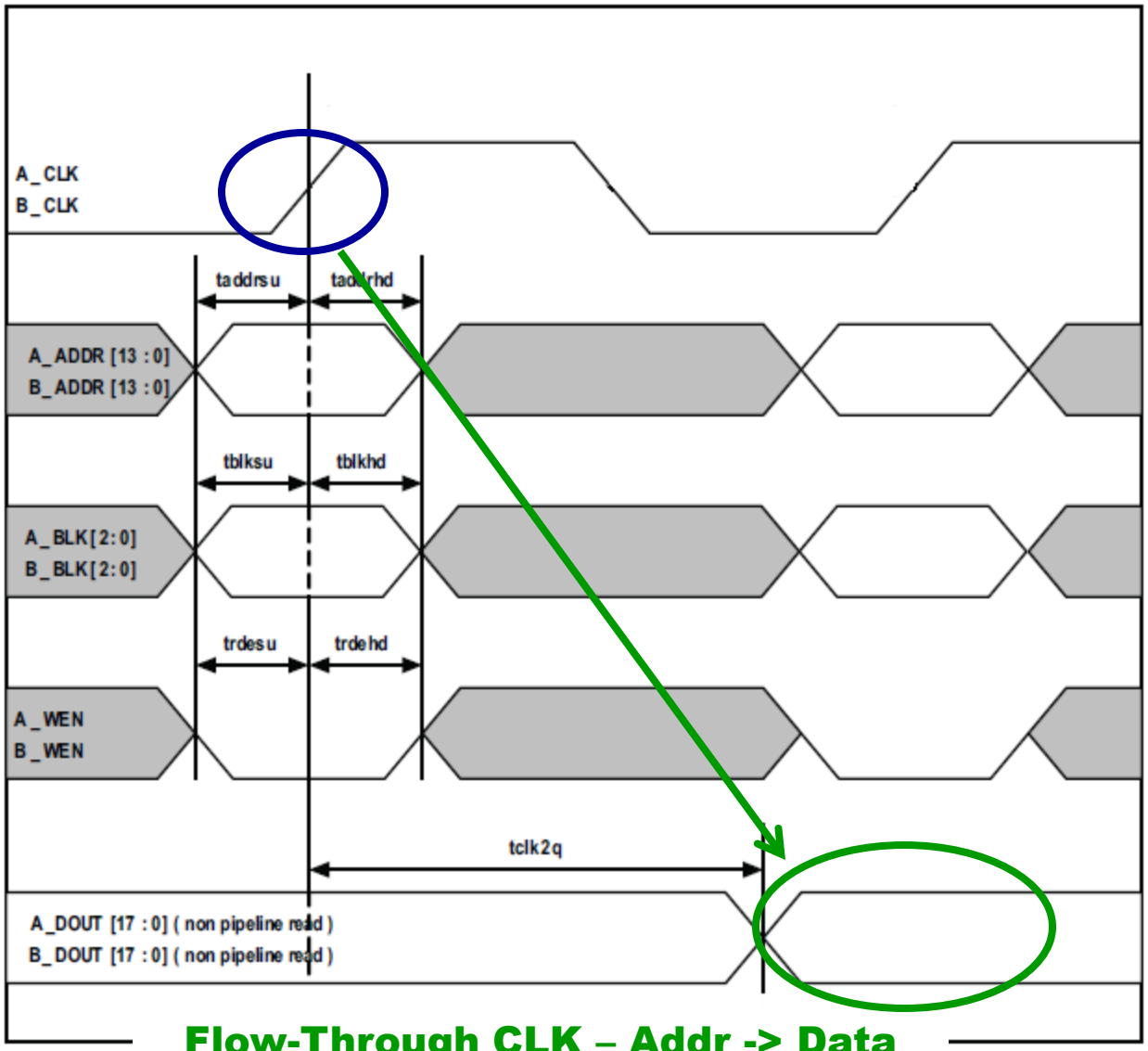
- Each LSRAM block is 18,432 bits
- Dual Port Configuration
  - Two independent data ports
    - Both ports are 18 bits wide or less
  - 1Kx18, 2Kx9, 4Kx4, 8Kx2, 16Kx1
- Two Port Configuration
  - One read port and one write port
  - 512x36, 1Kx18, 2Kx9, 4Kx4, 8Kx2, 16Kx1
- 400 MHz synchronous operation
- Supports two types of read operations:
  - Flow-through read (non-pipelined) and pipelined read
- Ideal for creating larger memories



LSRAM Block Diagram

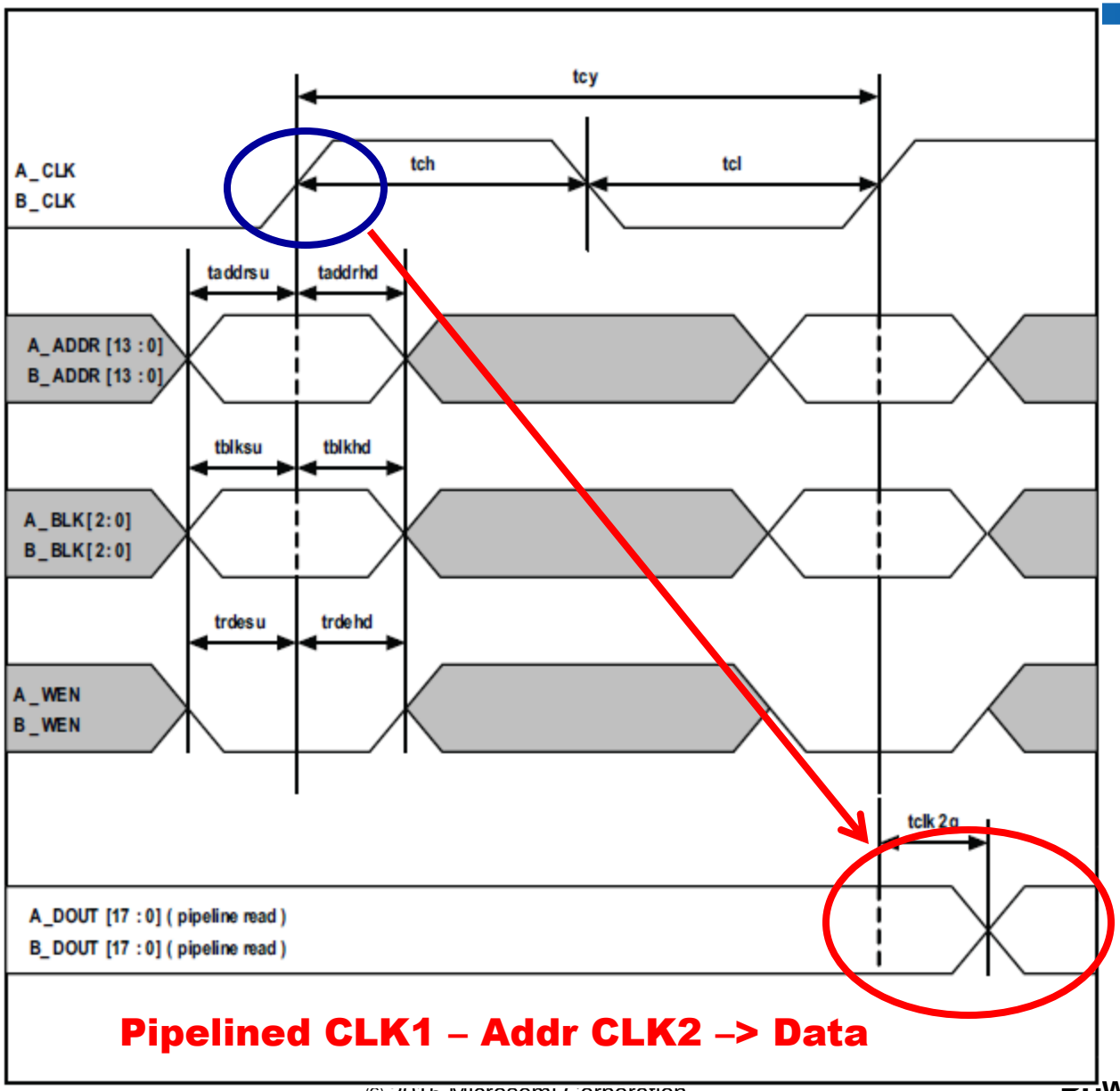


LSRAM: Read Cycle – Flow-Through Output



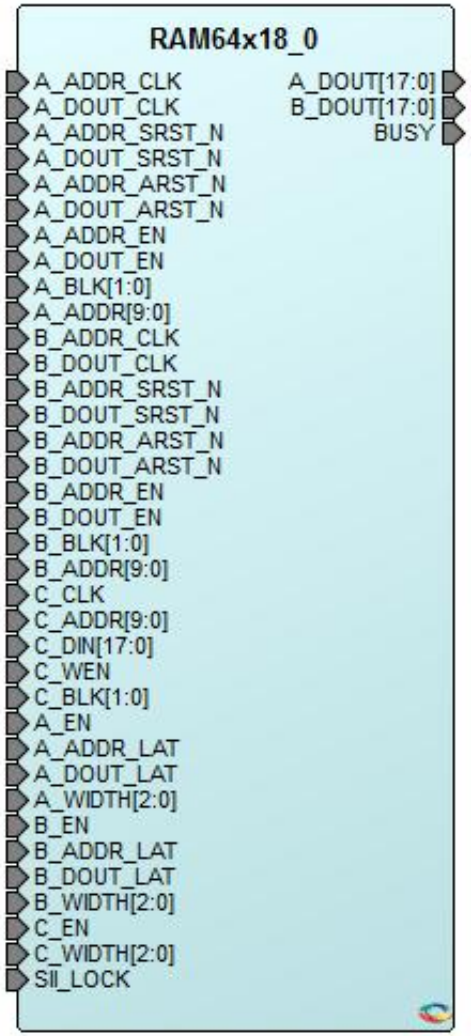
Flow-Through CLK – Addr -> Data

LSRAM: Read Cycle – Pipelined Output

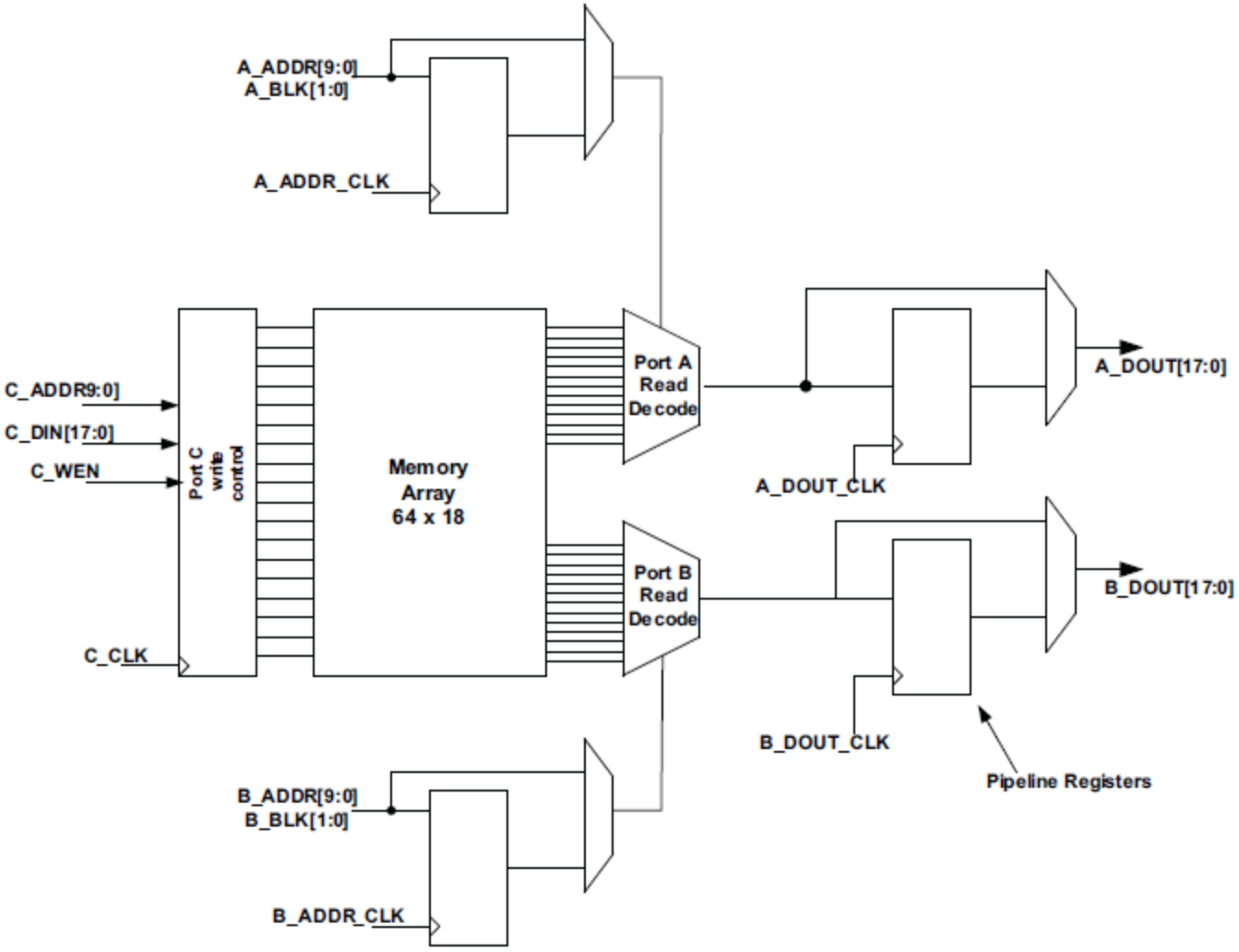


FPGA Memory Blocks – Micro SRAM (uSRAM)

- Each uSRAM block is 1152 bits
- 18-bit Three Port Memory (2 Reads and 1 Write)
  - Read ports are independent; simultaneous read operations can be performed from both ports at the same address
- Synchronous Write Operation
- Asynchronous or Synchronous Read Operation
- 250 MHz Sync Write, Sync Read Operation
- Configurations
  - 64x18, 128x9, 256x4, 512x2, and 1024x1
- Ideal for small memory, small FIFO, register file, and small DSP look-up tables

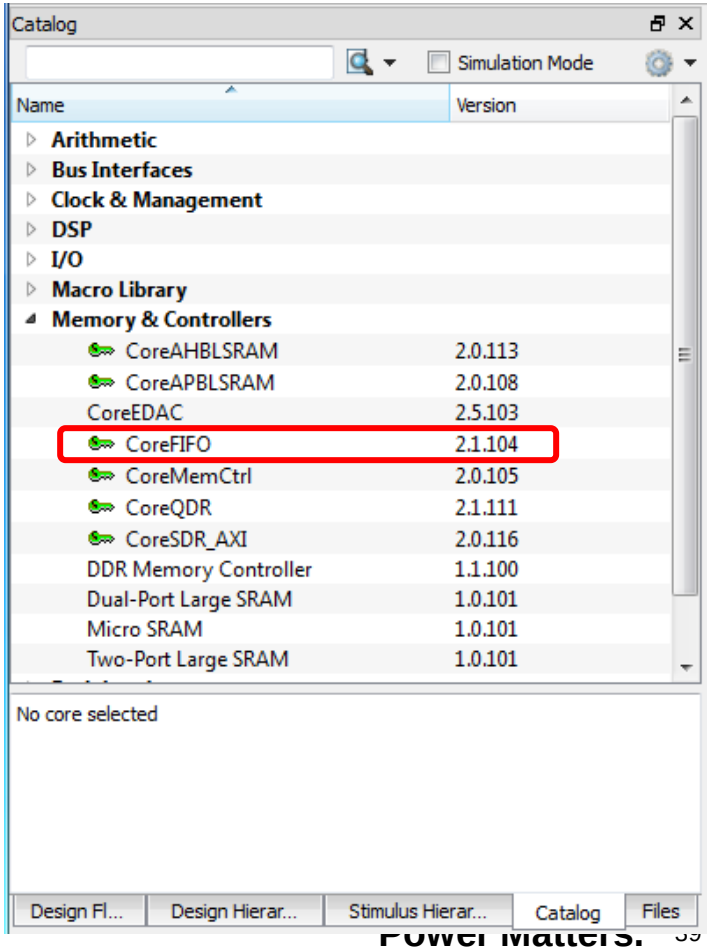


uSRAM Block Diagram



Implementing FIFOs

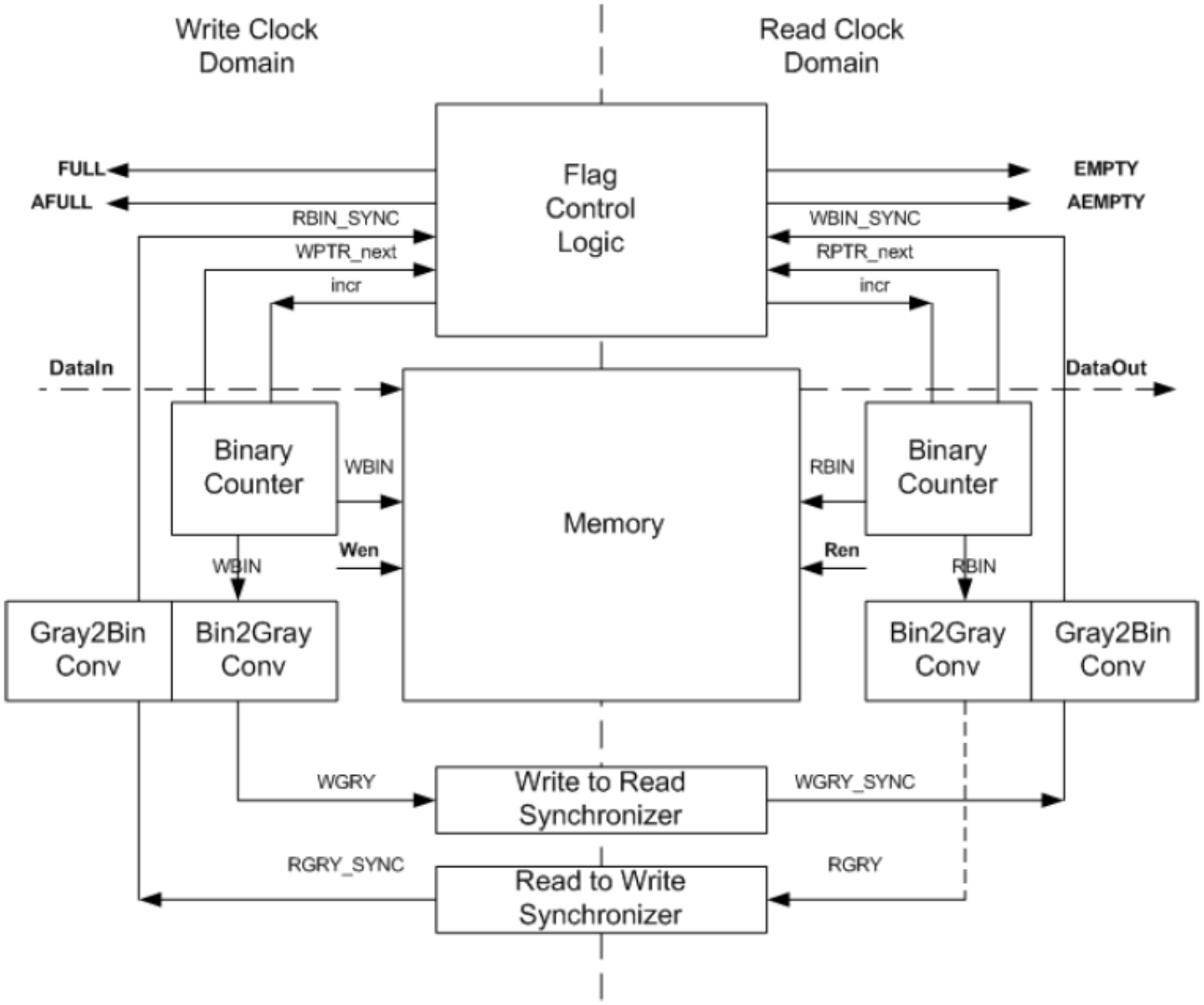
- FIFO Capability is not built into the SmartFusion2/IGLOO2 SRAM block
  - No decoder, FIFO control and flag logic, etc. as in ProASIC3
- Use CoreFIFO to implement FIFOs



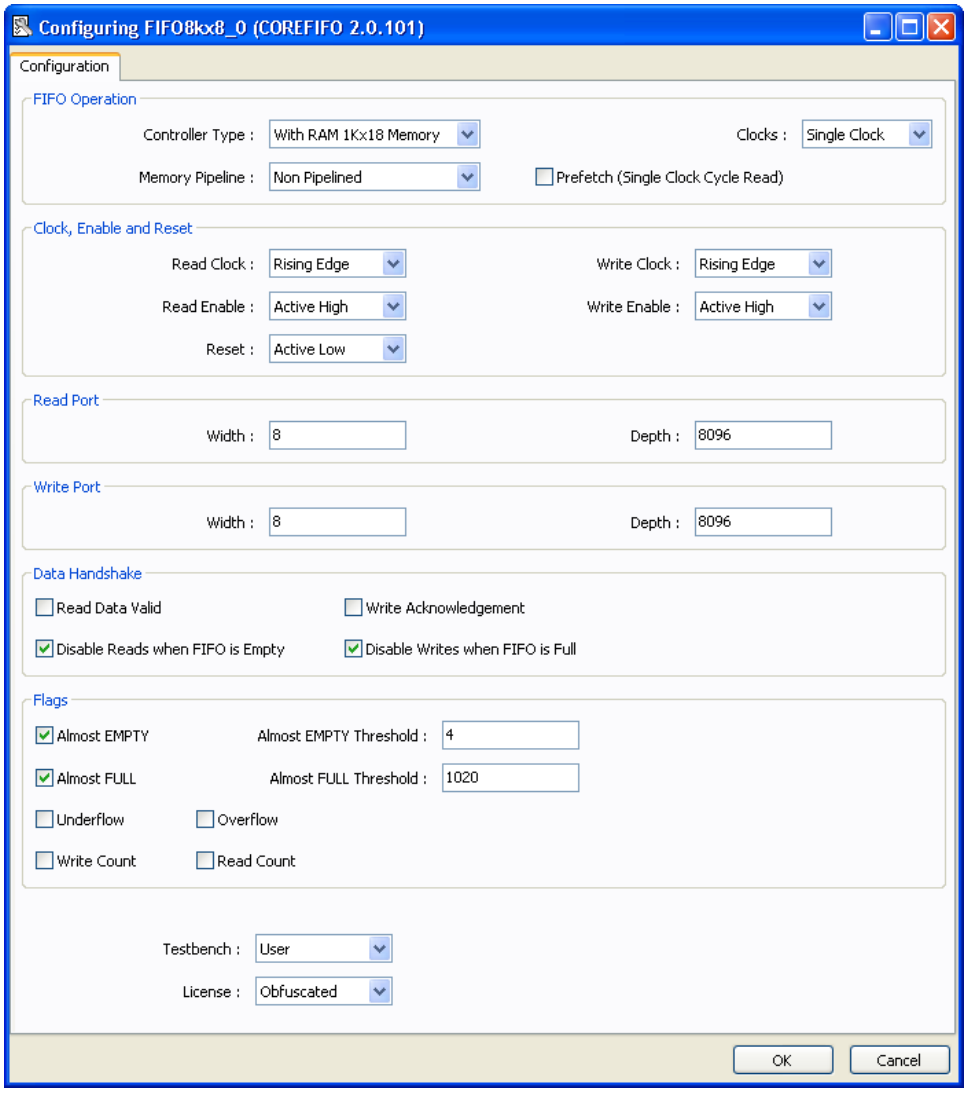
CoreFIFO Overview

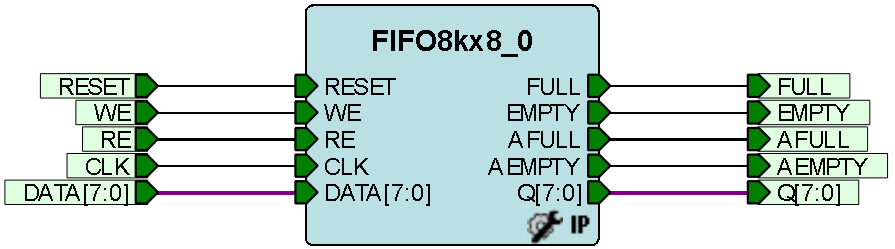
- Implements standard asynchronous FIFO and synchronous FIFO logic
- CoreFIFO Features
  - Dual/single clock operation
    - Positive or Negative Clock edge
  - Flag generation
    - Full/Empty flags
    - Optional Almost Full and Almost Empty flags
      - Almost full/empty single threshold value for assertion
  - Variable aspect ratio (depth/width)
  - Error status generation with overflow and underflow
  - Write acknowledge and read data valid generation
  - Pre-fetch mode option to provide read data in the same clock cycle
  - Supports large RAM (LSRAM) and micro RAM (uSRAM) or controller only option

CoreFIFO Block Diagram



Example: 8k x 8 FIFO





Mathblocks

SmartFusion2 Mathblock
Optimized for DSP Applications

	Features	M2S005	M2S010	M2S025	M2S050	M2S060	M2S090	M2S150
FPGA	Logic Modules (LUT + DFF)	6,060	12,084	27,696	56,340	56,340	86,316	146,124
	LSRAM 18K Blocks	10	21	31	69	69	109	236
	uSRAM1K Blocks	11	22	34	72	72	112	240
	Total RAM (Kbits)	191	400	592	1,314	1,314	2,074	4,488
	Math Blocks	11	22	34	72	72	84	240
	PLLs and CCCs	2		6				8

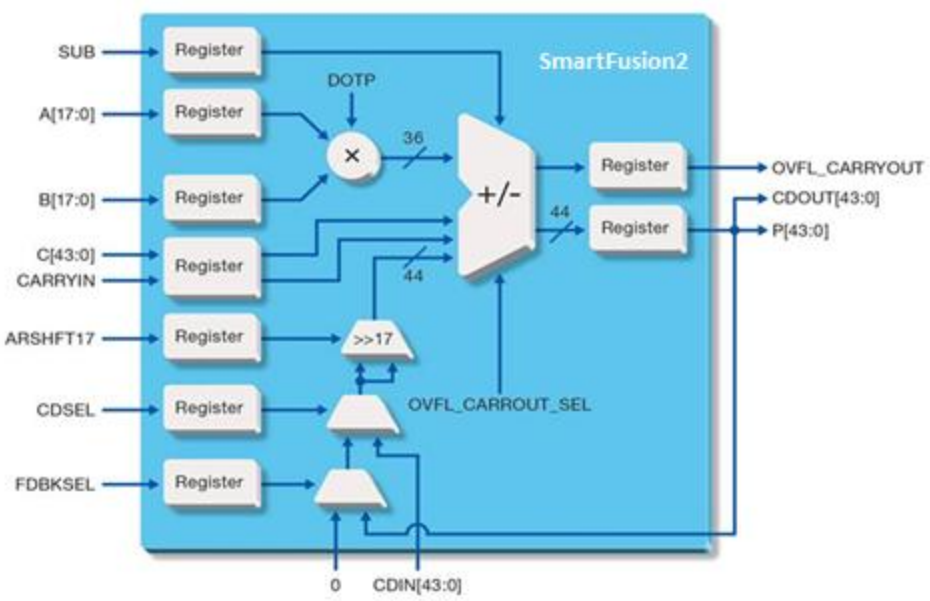
Notes:
Registers are optional on all inputs
A*B+C+D

SYNOPSYS®
Synplify Pro®
Synphony Model Compiler

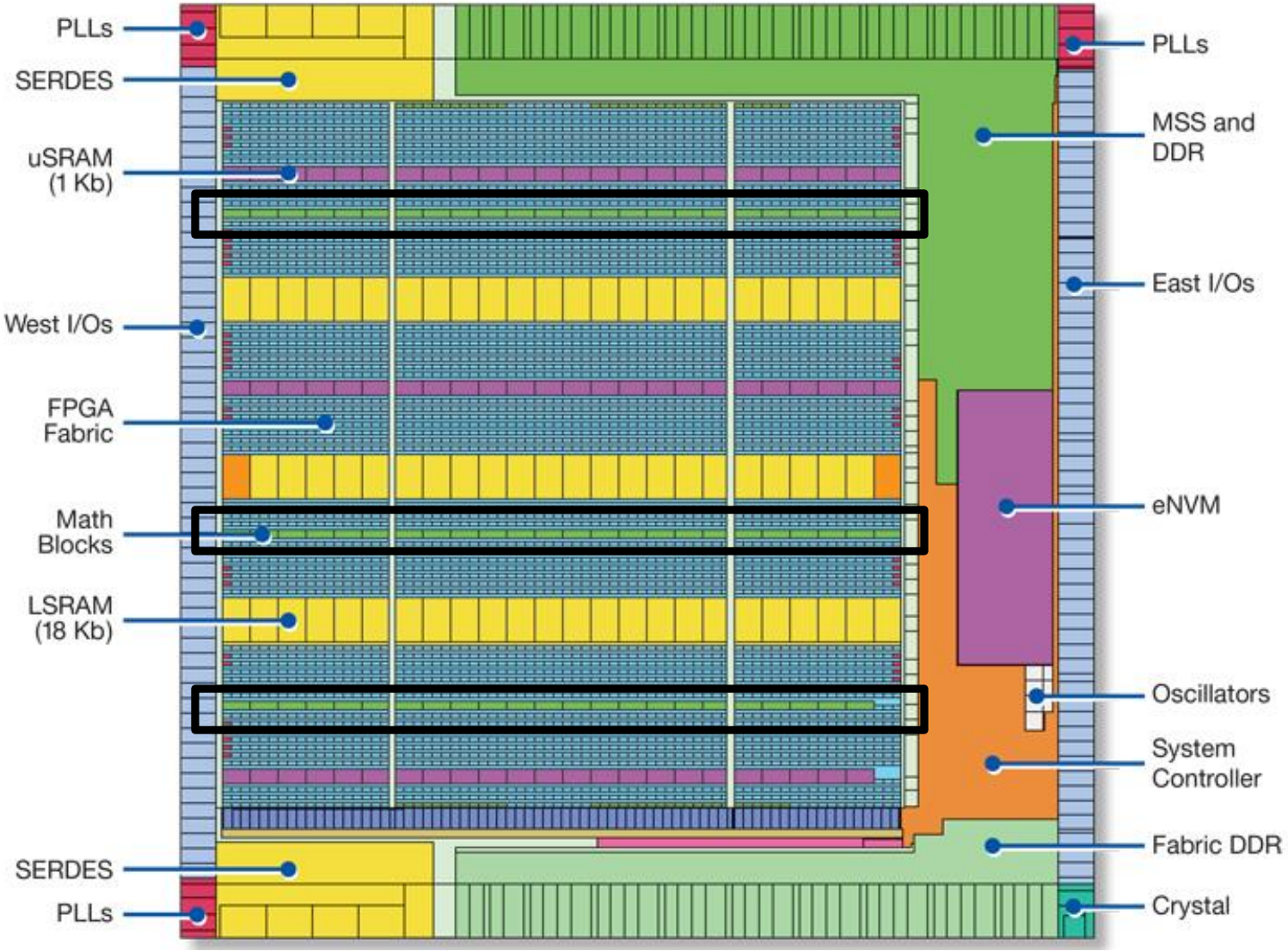
MathWorks®
MATLAB®
Simulink®

MATH Block Overview

- 11 to 240 Mathblocks
- Built-in addition, subtraction, and accumulation units to combine multiplication results efficiently
  - Supports 18 x 18 signed, 17 x 17 unsigned multiplication
  - Adder support: (A x B) + C or (A x B) + D or (A x B) + C + D
  - Supports dot-product (A0[8:0] x B0[8:0] + A1[8:0] x B1[8:0])
- All inputs and outputs can be registered
  - Clock-gated input and output registers for power optimizations
- Independent third input C with data width 44 bits completely registered
- Internal cascade signals (44-bit CDIN and CDOUT) to support larger accumulators/adders/subtractors without extra logic
- Loopback capability to support adaptive filtering

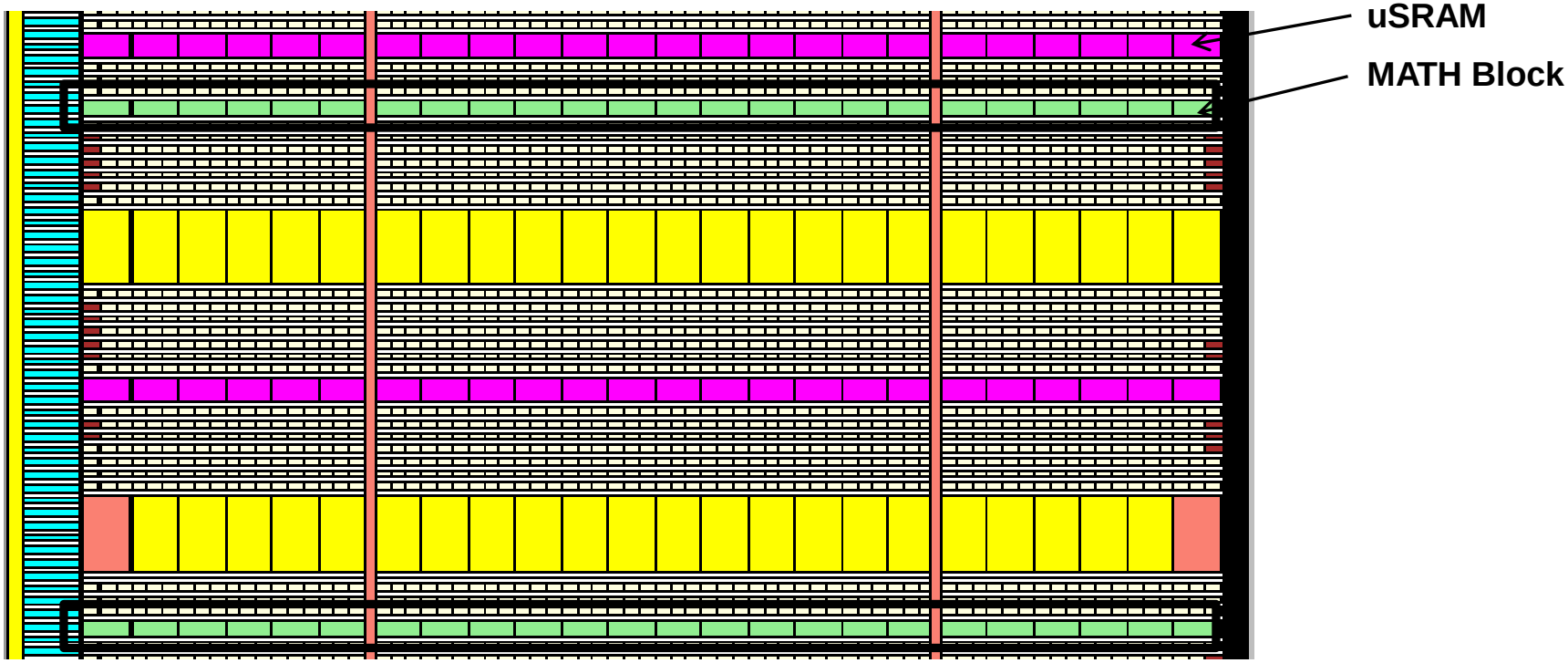


Mathblock Locations



Mathblock Resources per Device

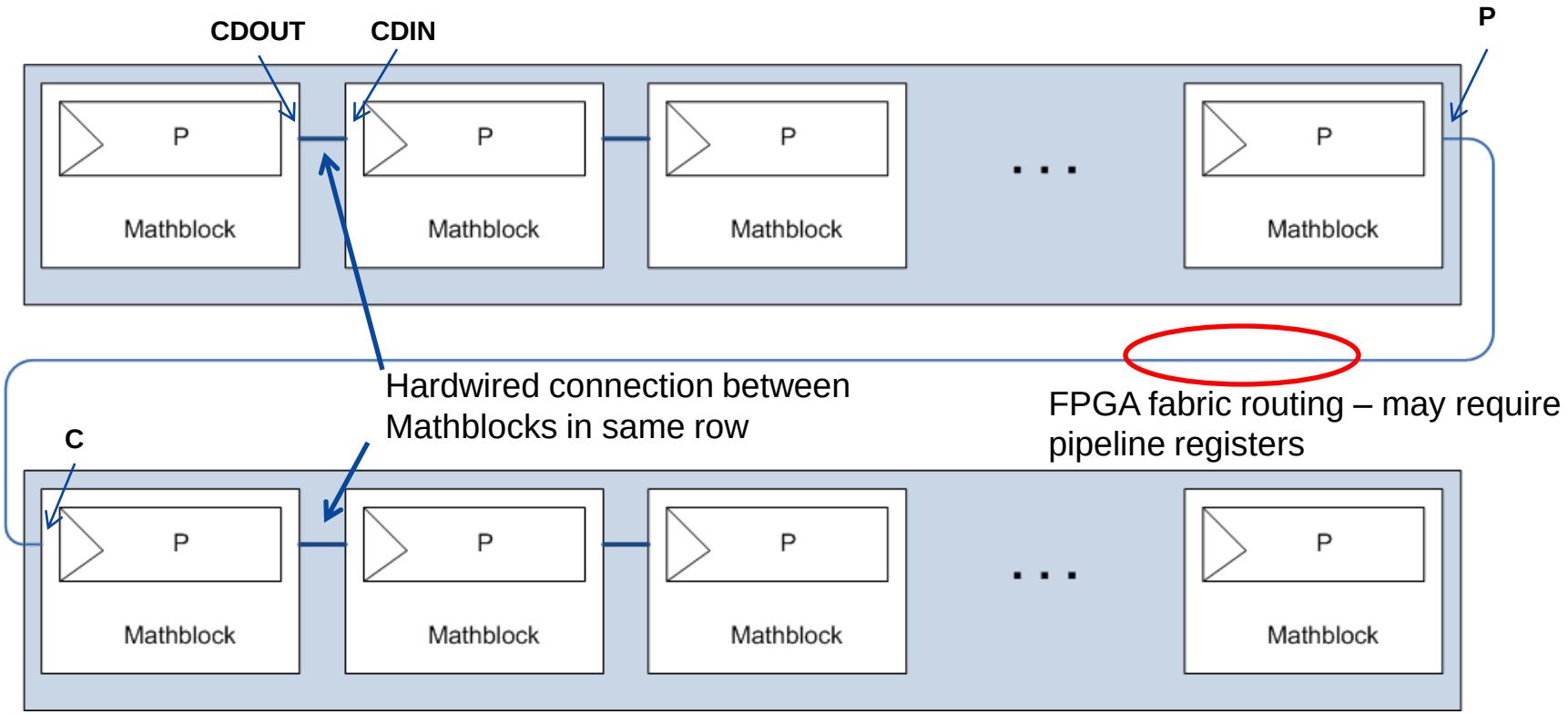
	M2S005 M2GL005	M2S010 M2GL010	M2S025 M2GL025	M2S050 M2GL050	M2S060 M2GL060	M2S090 M2GL090	M2S150 M2GL150
MATH Block rows	1	2	2	3	3	3	6
MATH Blocks	11	22	34	72	72	84	240



- MATH Blocks are located in rows near the uSRAM blocks
  - Useful for coefficient buffering

Mathblock Dedicated Connections

- Dedicated connection between Mathblocks in same row improves performance
  - Connection between rows incurs additional delay



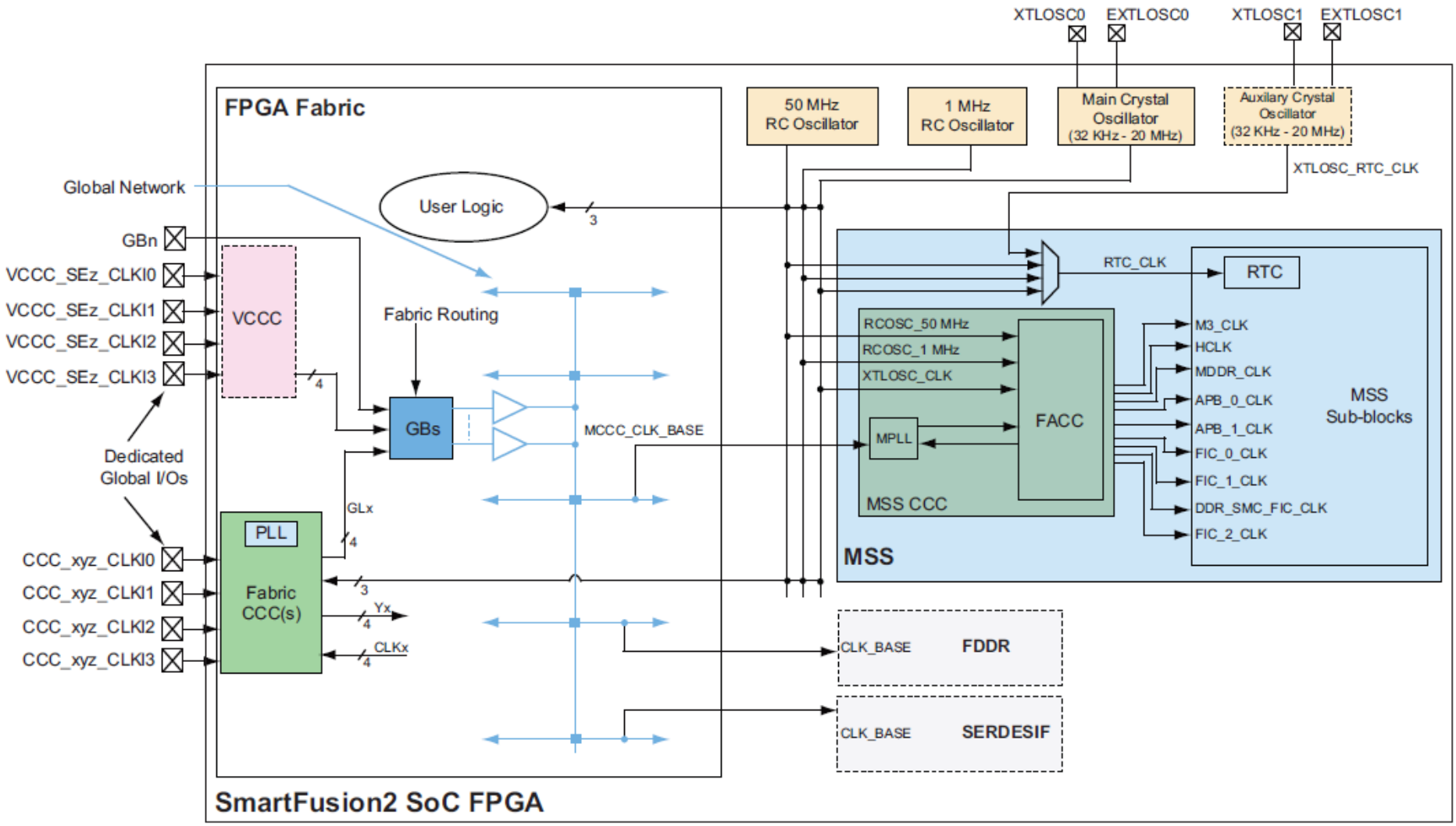
Clocking Resources

SmartFusion2 / IGLOO2
Clocking Resource Summary

		SmartFusion2 / IGLOO2 device						
		M2S005 M2GL005	**M2S010 M2GL010**	**M2S025 M2GL025**	**M2S050 M2GL050**	**M2S060 M2GL060**	**M2S090 M2GL090**	**M2S150 M2GL150**
On-chip Oscillators	1 MHz RC Oscillator	1	1	1	1	1	1	1
	50 MHz RC Oscillator	1	1	1	1	1	1	1
	Main Crystal Oscillator	1	1	1	1	1	1	1
	Auxiliary Oscillator	1	1	1	0	1	1	1
Fabric CCCs[1]		2	2	6	6	6	6	8
MSS CCC[1]		1	1	1	1	1	1	1
Global Buffers		8	8	16	16	16	16	16
Dedicated Global I/Os		16	16	32	32	32	32	32

[1]Each CCC has a dedicated PLL for clock synchronization and clock synthesis

SmartFusion2 Clocking Overview

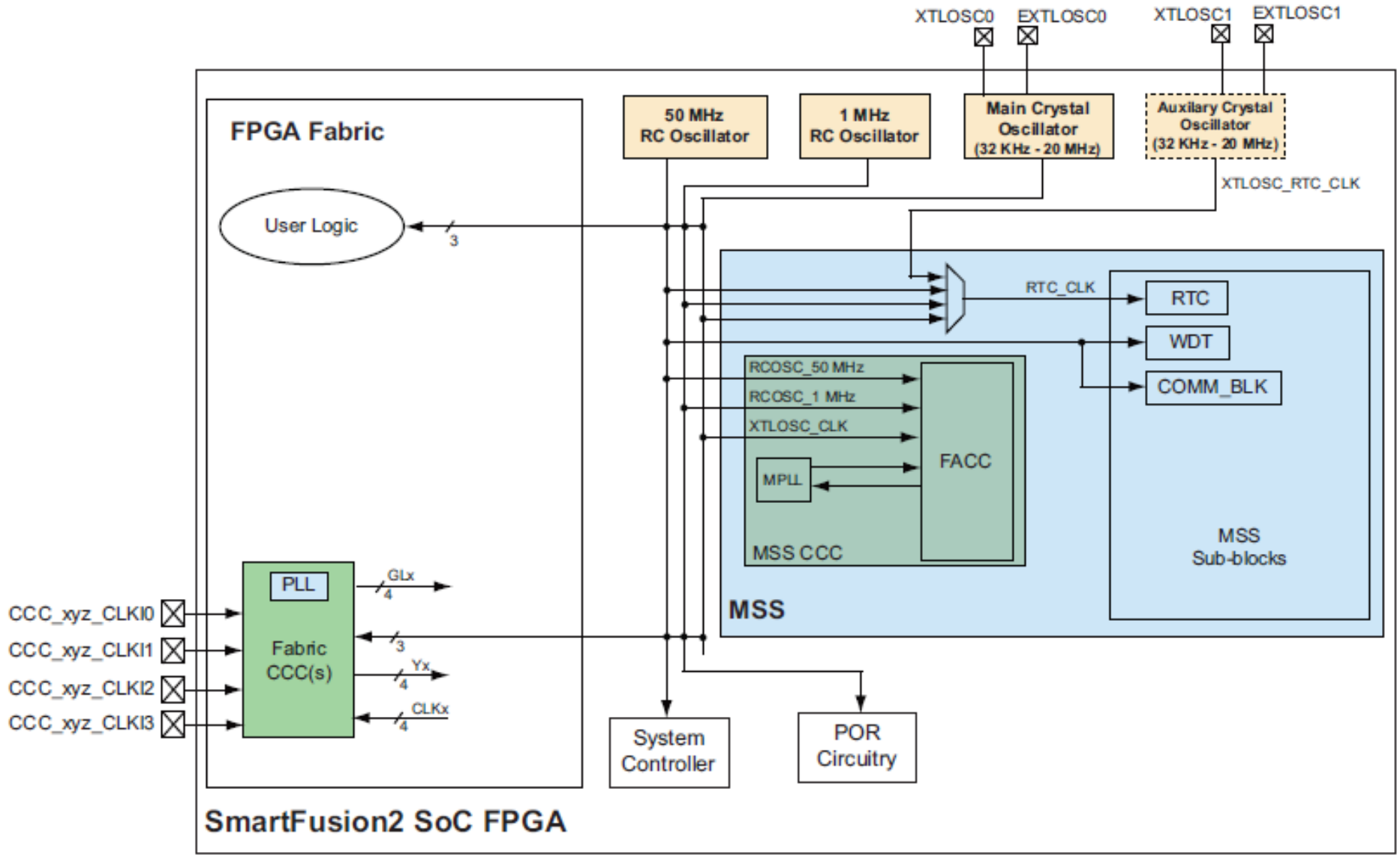


SmartFusion2 On-chip Oscillators

- Available on-chip oscillators:
  - 1 MHz Embedded RC Oscillator, 5% accurate
  - 25/50 MHz Embedded RC Oscillator, 5% accurate
  - Main Crystal Oscillator
  - Auxiliary Crystal Oscillator
    - Dedicated for MSS RTC clocking as an alternative clock source
    - Not available on M2S050 devices
- On-chip oscillators can provide reference clock input to:
  - Fabric CCCs[1]
  - User logic in the FPGA fabric through fabric routing[1]
  - MSS RTC[1]
  - MSS during Flash*Freeze mode
- On-chip oscillators also supply clocks to hard IP blocks such as the system controller and POR circuitry

[1]Not supported with Auxiliary crystal oscillator

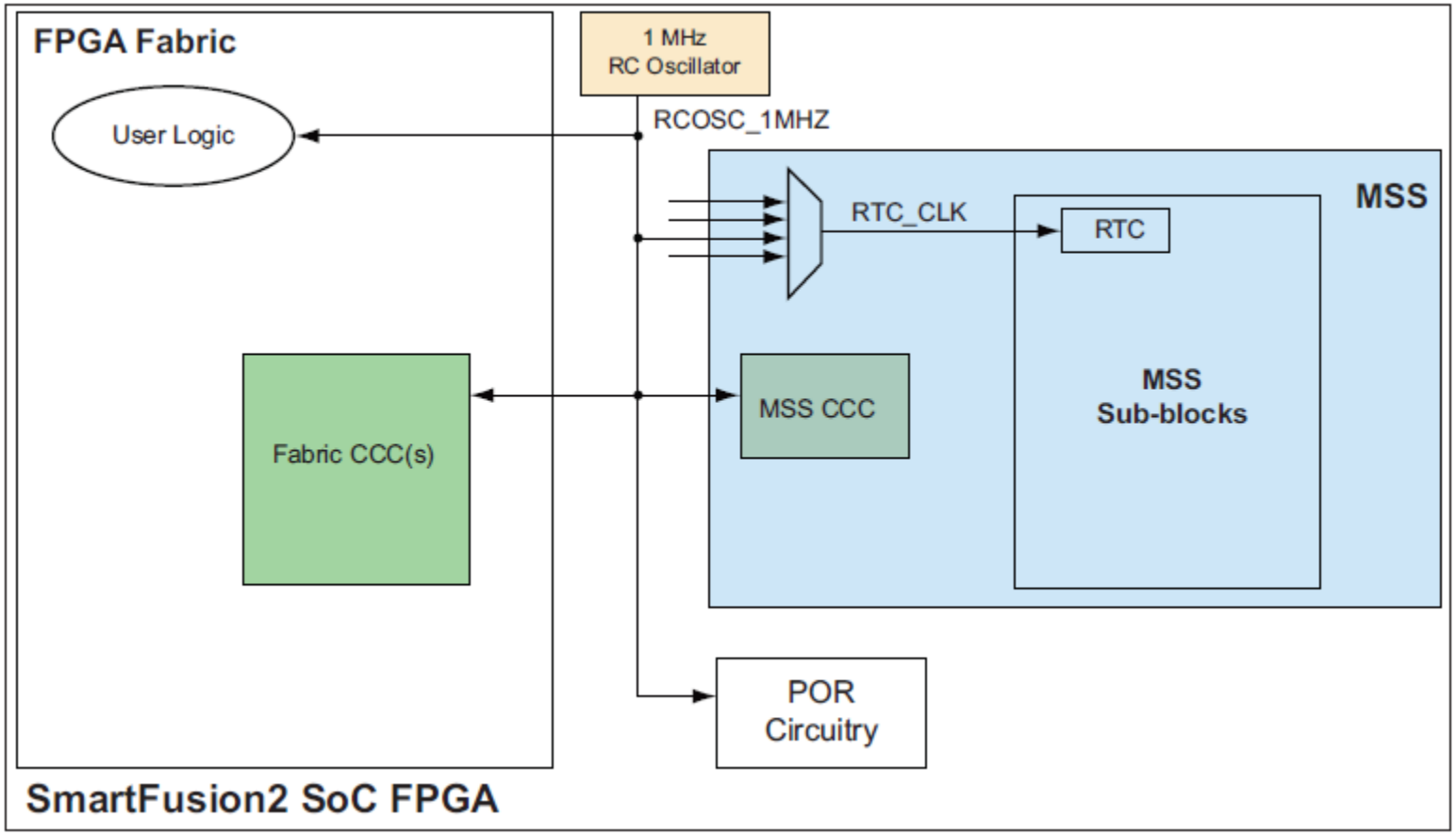
SmartFusion2 On-Chip Oscillators Clock Sourcing Capabilities



1 MHz RC Oscillator

- 1 MHz RC oscillator generates a calibrated 1 MHz clock signal
  - Powered by the device core supply (VDD)
  - No external components required
- Can be configured as a clock source to:
  - Fabric CCCs
  - User logic in the FPGA fabric through fabric routing
  - MSS RTC
  - MSS during Flash*Freeze mode
- Provides the clock to the POR circuitry to determine the duration of POR signal assertion during device power-up

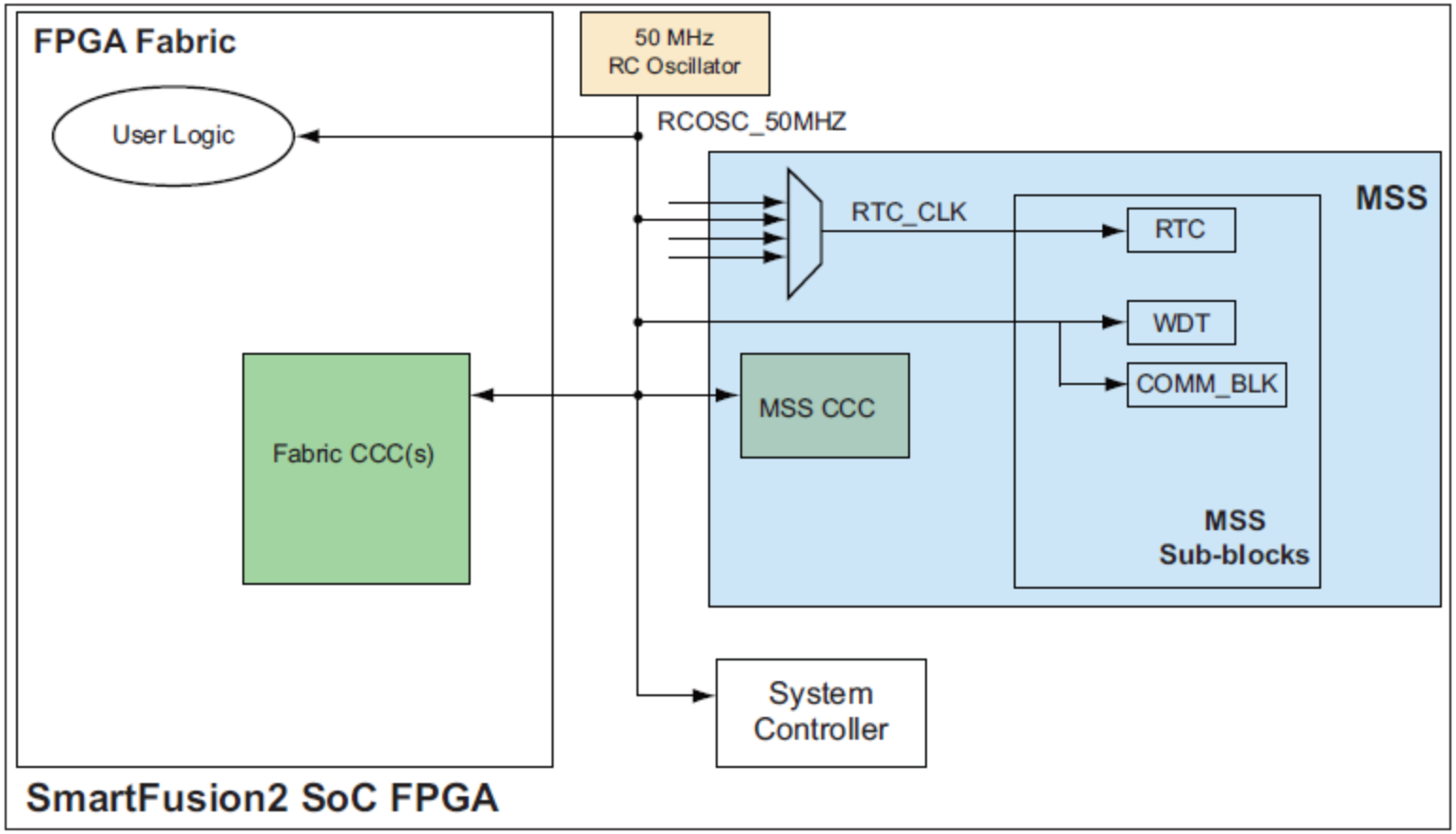
1 MHz RCO Clock Sourcing Capabilities



25/50 MHz RC Oscillator

- The 25/50 MHz RC oscillator generates a calibrated 50 MHz clock signal
  - Powered by the device core supply (VDD)
  - No external components required
- Can be configured as a clock source to:
  - Fabric CCCs
  - User logic in the FPGA fabric through fabric routing
  - MSS RTC
  - MSS during Flash*Freeze mode
- Supplies a clock to the following hard IP blocks:
  - System controller
  - MSS after power-on reset for system initialization
  - MSS Watchdog 32-bit counter
  - Communication block (COMM_BLK) core logic

25/50 MHz RCO Clock Sourcing Capabilities



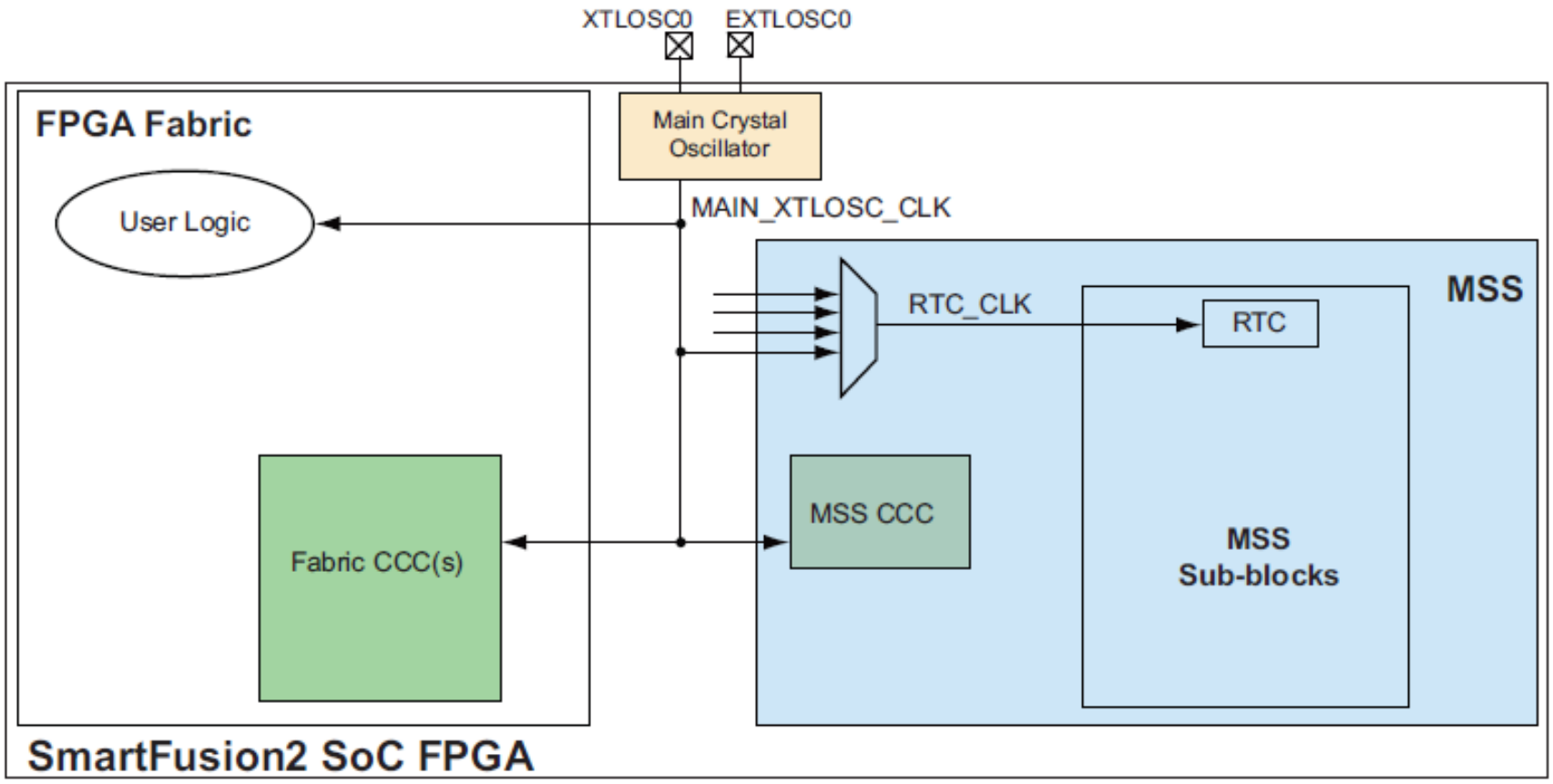
Main Crystal Oscillator

- The main crystal oscillator works with an external crystal, ceramic resonator, or a resistor-capacitor (RC) network to generate a high-precision clock in the range of 32 KHz to 20 MHz
- Can be configured as a clock source to:
  - Fabric CCCs
  - User logic in the FPGA fabric through fabric routing
  - MSS RTC
  - MSS during Flash*Freeze mode[1]

  [1] Not available for M2S050
- Output Frequency Range

Source	Output Frequency Range
Crystal	32 KHz to 20 MHz
RC Circuit	32 KHz to 4 MHz
Ceramic Resonator	500 KHz to 4 MHz

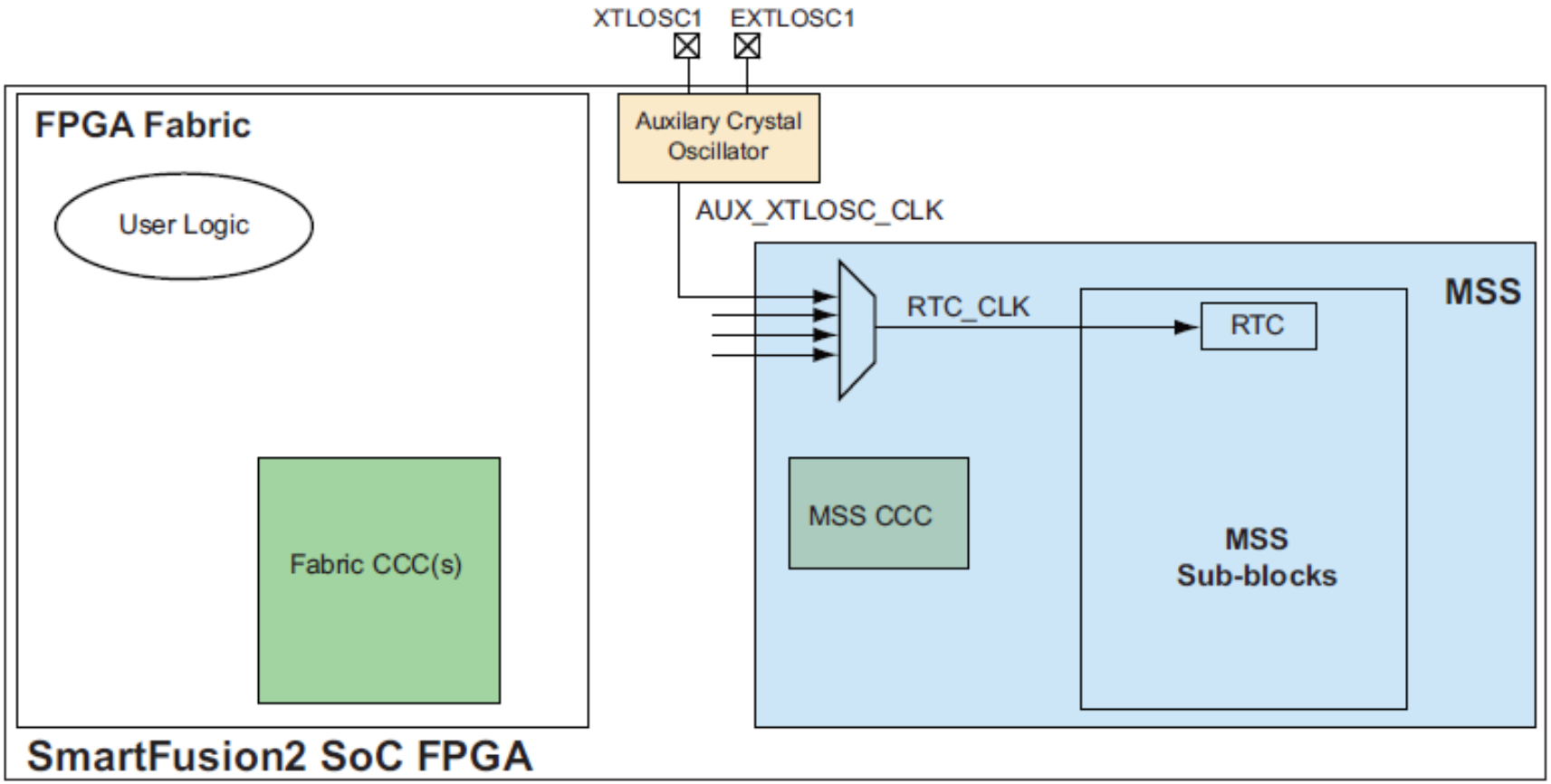
Main Crystal Oscillator Clock Sourcing Capabilities



Auxiliary Crystal Oscillator

- SmartFusion2 devices have an auxiliary crystal oscillator dedicated to MSS RTC clocking as an alternative clock source
  - Not available in M2S050
- The Auxiliary crystal oscillator works with an external crystal, ceramic resonator, or an RC circuit to generate a high-precision clock in the range of 32 KHz to 20 MHz
  - Output frequency range, operating modes, and characteristics are the same as the Main Crystal Oscillator

Auxiliary Crystal Oscillator Clock Sourcing Capabilities



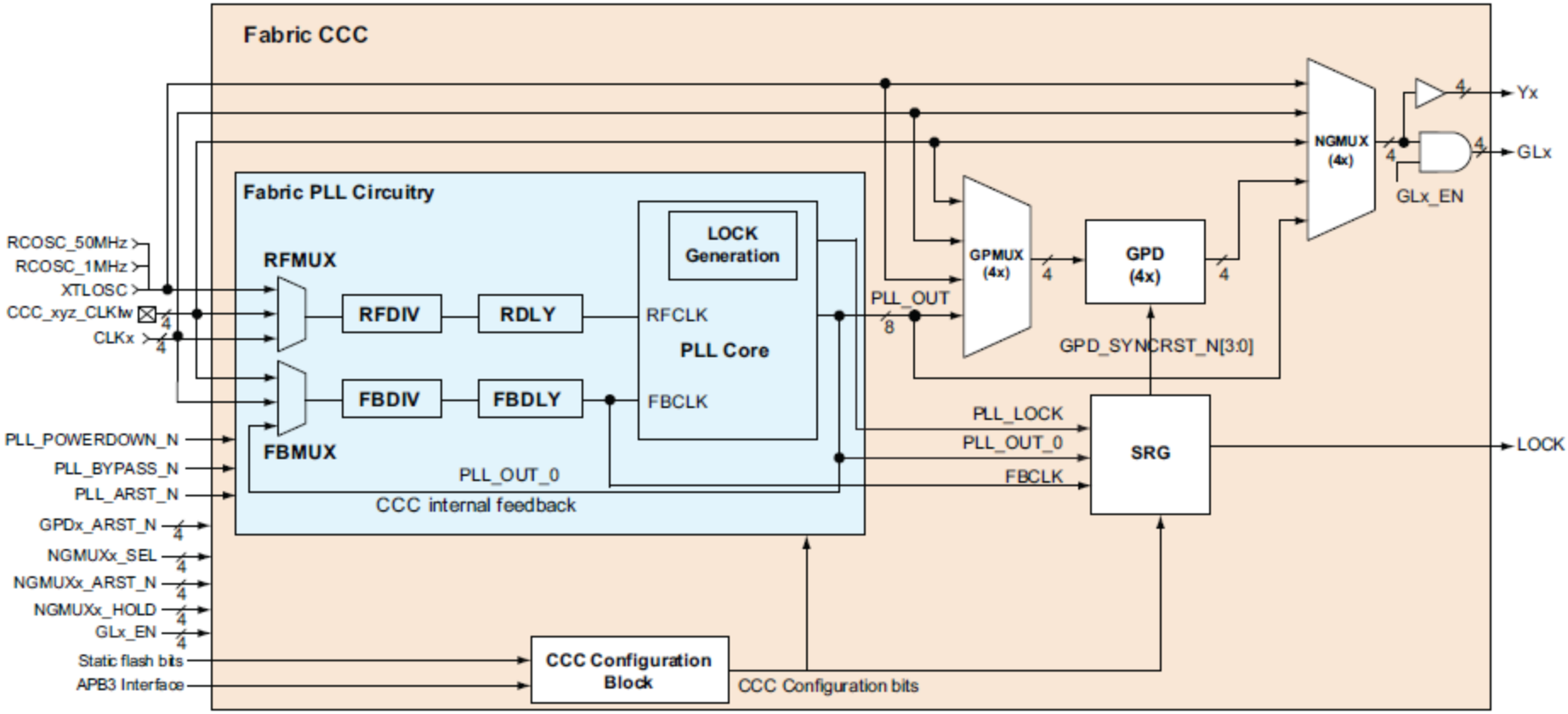
Fabric Clock Conditioning Circuitry

- SmartFusion2 devices have two, six, or eight fabric CCCs
  - Fabric CCCs enable flexible clocking schemes for FPGA fabric logic
  - Can provide the base clock for on-chip hard IP blocks - MSS, FDDR, and SERDESIF
- Each fabric CCC includes a dedicated PLL and generates clock signals of different frequency and phase
  - The associated PLL can be bypassed if it is not required

Fabric CCC Features

- Each fabric CCC supports:
  - Internal and external clock sources
  - Generation of four different clocks with a maximum frequency up to 400 MHz
    - Drives global network and/or the local routing network
- CCC Features
  - Automatic output resynchronization after PLL lock
  - Configurable phase error window for PLL lock assertion
  - Configurable PLL lock delay
  - Spread spectrum clock generation (SSCG)
  - Dynamic clock switching without any glitches (glitchless mux)
  - Clock delay or clock advancement
  - Clock phase shifting
  - Clock inversion
  - Clock gating

Fabric CCC Block Diagram



Fabric CCC Clock Sources

- The following clock sources can be used to drive the fabric CCC clock input and the fabric CCC outputs directly:
  - 1 MHz RC Oscillator
  - 25/50 MHz RC Oscillator
  - Main Crystal Oscillator
  - Dedicated Global I/Os
  - FPGA Fabric Clock Sources
- The input clock frequency range for the fabric CCCs depends on the PLL usage
  - PLL used: PLL reference clock frequency must be between 1 MHz and 200 MHz
    - CCC-NE1 has 32 KHz reference clock support
  - PLL is bypassed: the fabric CCC input clock frequency can be up to 400 MHz

Fabric CCC Outputs

- Each fabric CCC generates up to four different global clocks and four core clocks
  - Global clocks drive the FPGA fabric global networks
  - Core clock outputs can be used to drive internal logic without using global network resources
    - The frequencies and phases of the core clocks are the same as those of the associated global clocks
- The fabric CCCs can be connected to each other or cascaded through the FPGA fabric

Fabric CCC PLL Core

- Input frequency range 1 MHz to 200 MHz
  - CCC-NE1 has 32 KHz reference clock support
- Output frequency range 20 MHz to 400 MHz
- 8 Output Phases and 45° Phase Difference
- A lock signal indicates that the PLL has locked onto the incoming signal
  - Asserts High to indicate that the frequency and phase lock is achieved
  - The precision of the lock discrimination can be adjusted using the lock window controls
    - Lock window can be adjusted between 500 parts per million (ppm) and 64,000 ppm in powers of 2
- A reset control signal is provided to power-down the PLL core and asynchronously reset all its internal digital blocks
- Fabric PLL is configured using static flash bits

CCC PLL Delay Lines and Dividers

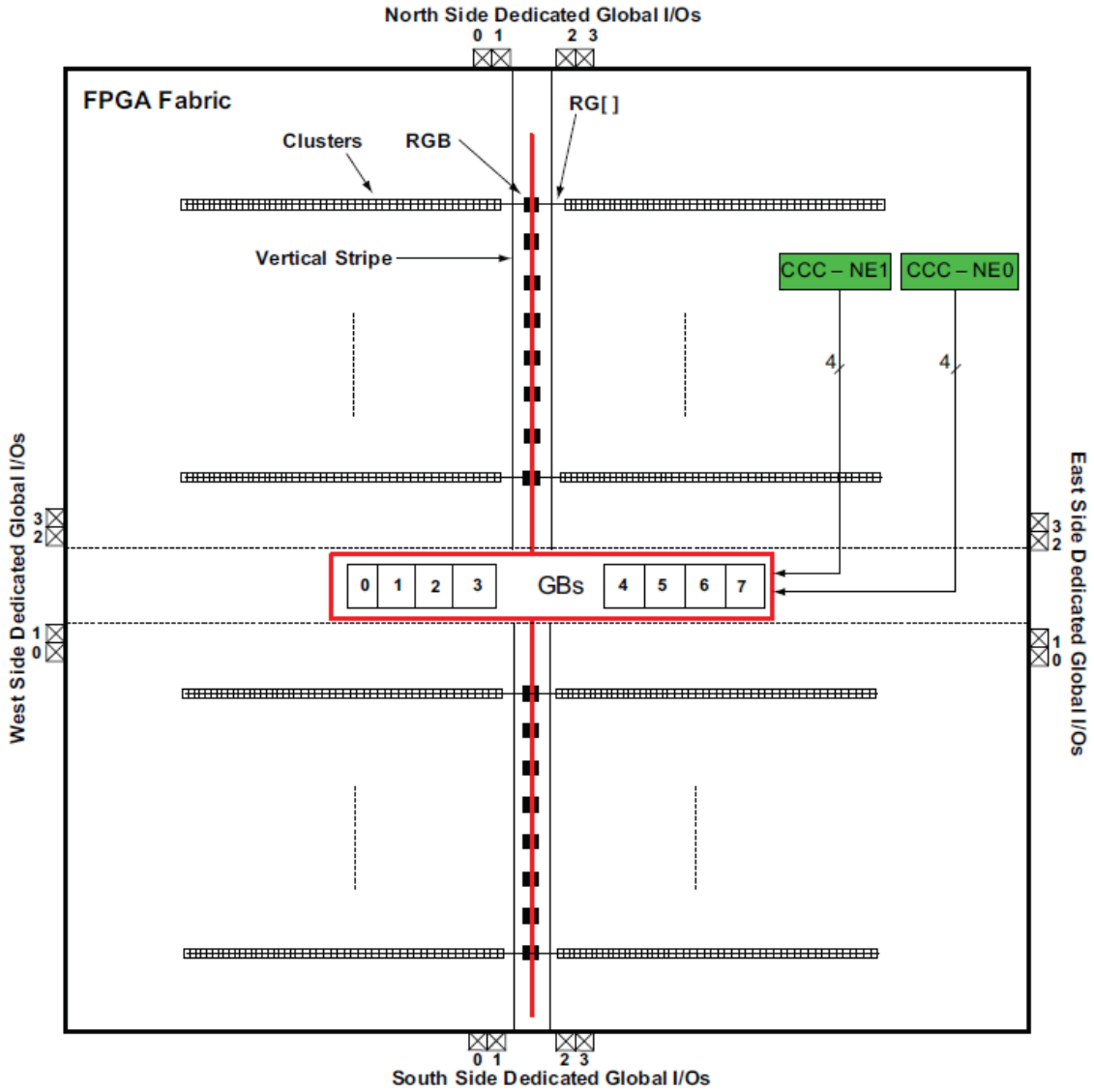
- Delay lines for PLL inputs:
  - Max delay = 7ns
  - Resolution = 100ps
  - 1-200 MHz input frequency
- Each fabric CCC has four General purpose mux-dividers (GPDs)
  - Divider range = 1 to 255
  - Input clock duty cycle is maintained
- The output of the GPDs can be used as the source for any of the four fabric CCC outputs

SmartFusion2 Global Networks

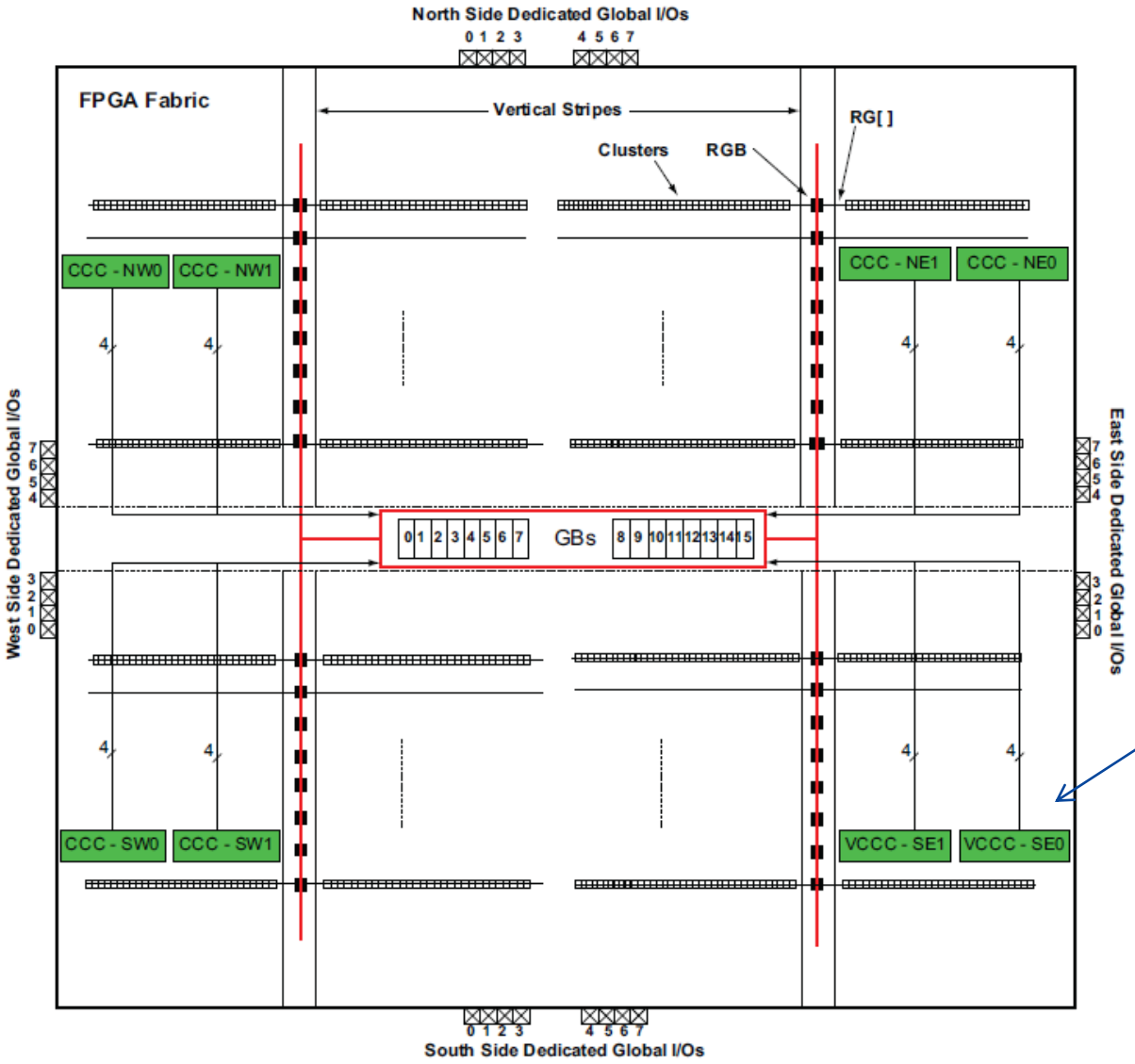
Global Signal Routing

- SmartFusion2 offers a dedicated low skew Global Network (GN) coupled to CCCs, PLLs, and a large number of dedicated IOs
- GN is designed to have very low skew corner to corner anywhere on the FPGA fabric
  - Skew < 100 ps worst case
  - We guarantee no hold violation when clocks are routed on GN
- GN can also be used to distribute very high fanout uncritical nets (control and/or logic both)
- 16 Global networks on M2S025 and above

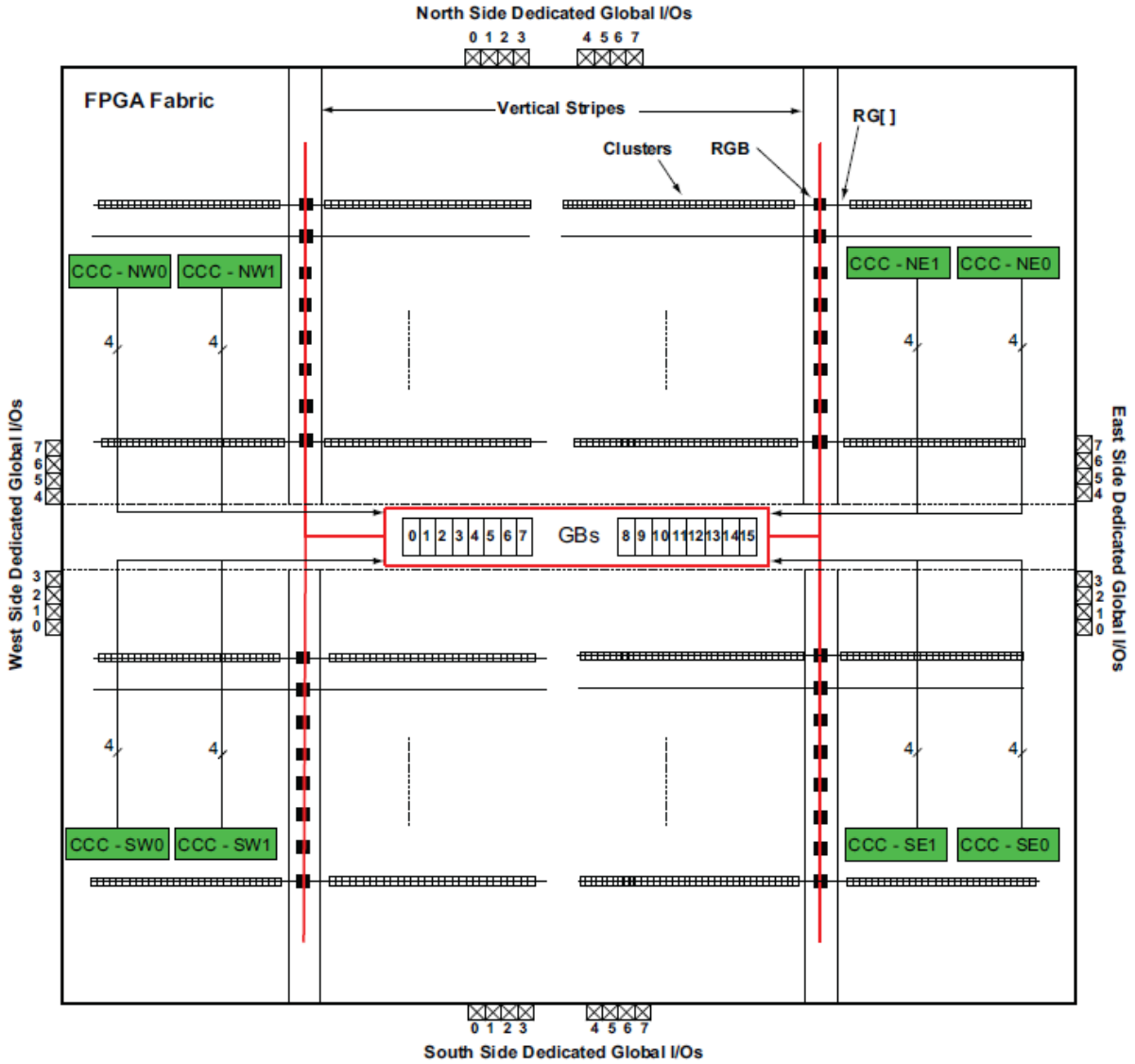
Global Network Architecture
M2S005 and M2S010



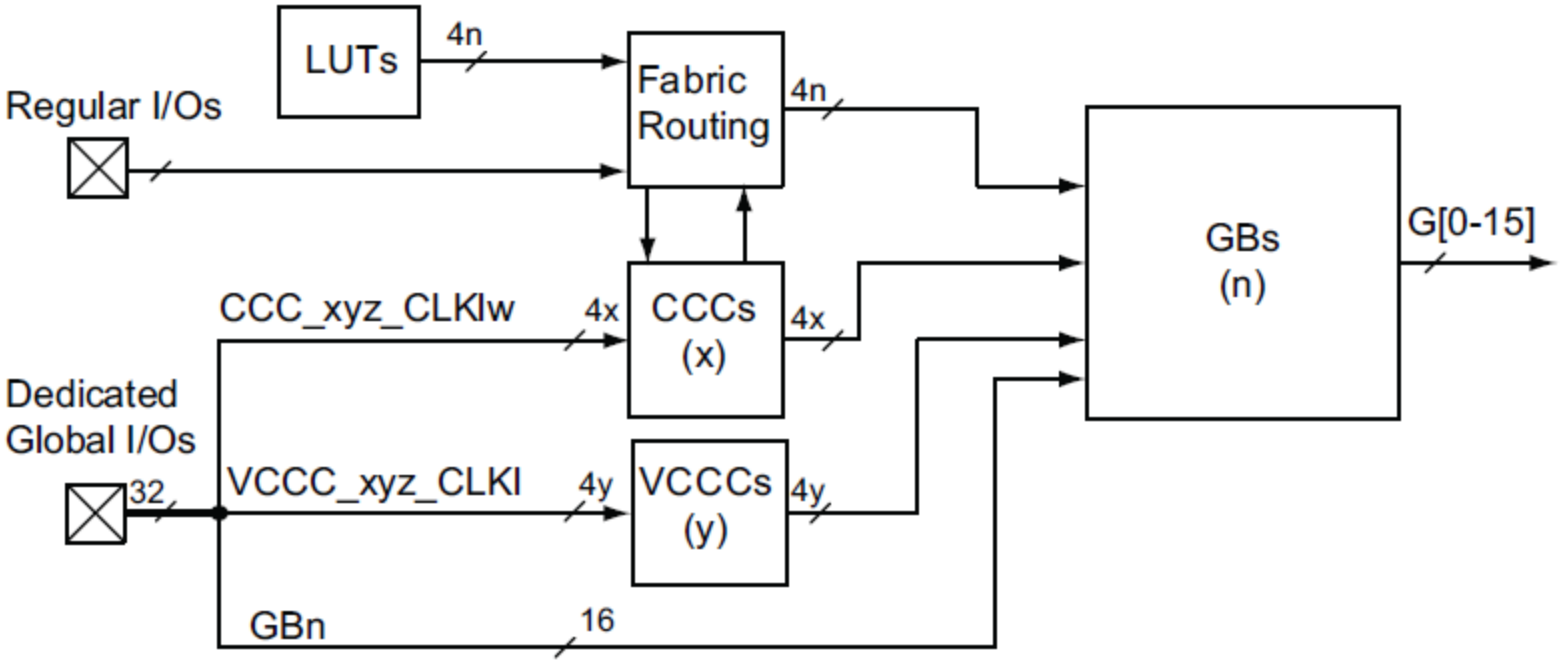
Global Network Architecture
M2S025, M2S050, M2S060 and M2S090



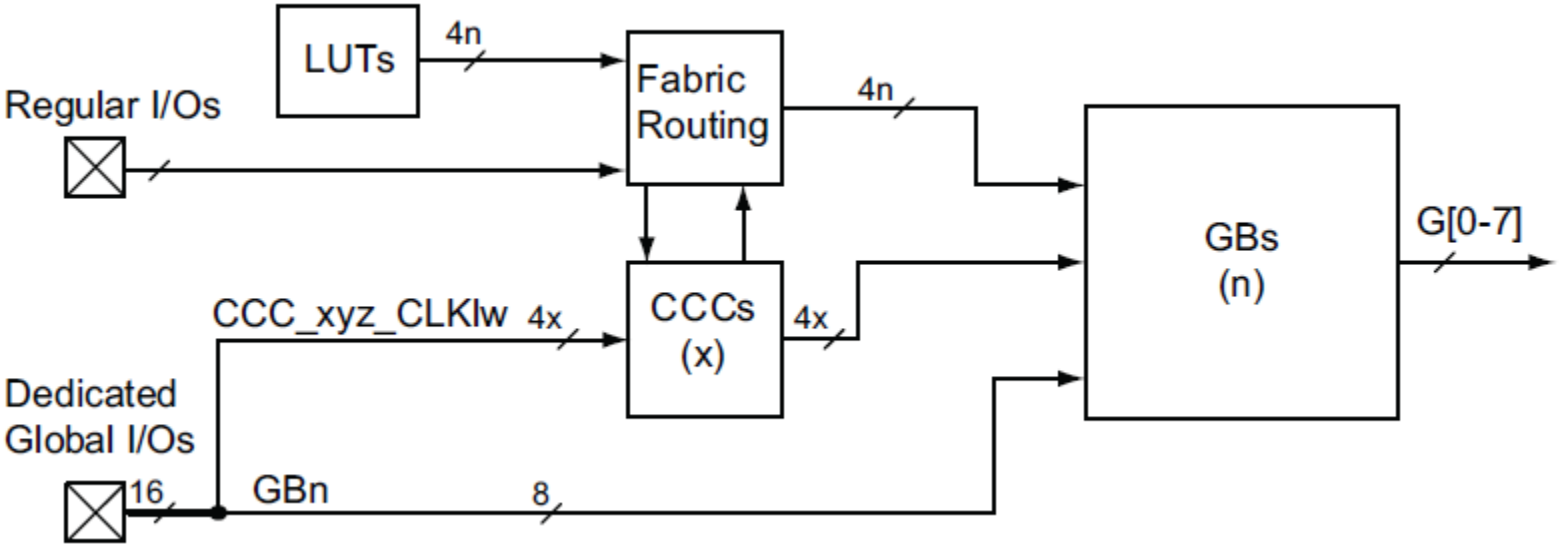
Global Network Architecture
M2S150



Multiple Sources Feed Global Blocks
Devices with Two Vertical Stripes



Multiple Sources Feed Global Blocks
Devices with One Vertical Stripe



Global Buffer (GB)

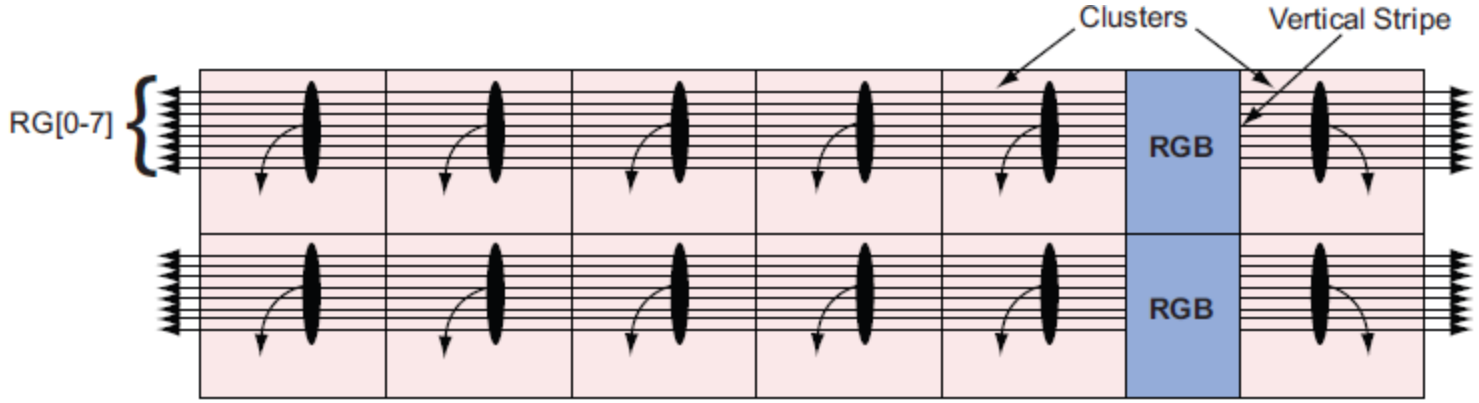
- GB is a multiplexer that generates an independent global signal
  - GB generates16 global signals for devices with two vertical stripes
  - GB generates 8 global signals for devices with one vertical stripe
  - These signals can go everywhere on the chip
  - They can also go to an arbitrary subset of all quarter (half) rows on the chip
    - This subset need not be a fixed size, like a quadrant
    - It does not even have to be rectangular, or simply connected, …
- GBs can be accessed from
  - Dedicated global IOs, CCC/VCCC global outputs, and the FPGA fabric routing
- Clocks coming from regular IOs can reach GBs or RGBs through FPGA fabric routing
- Signals from dedicated IOs and CCCs must go to GB first; They cannot go to RGs directly

Dedicated Global IOs

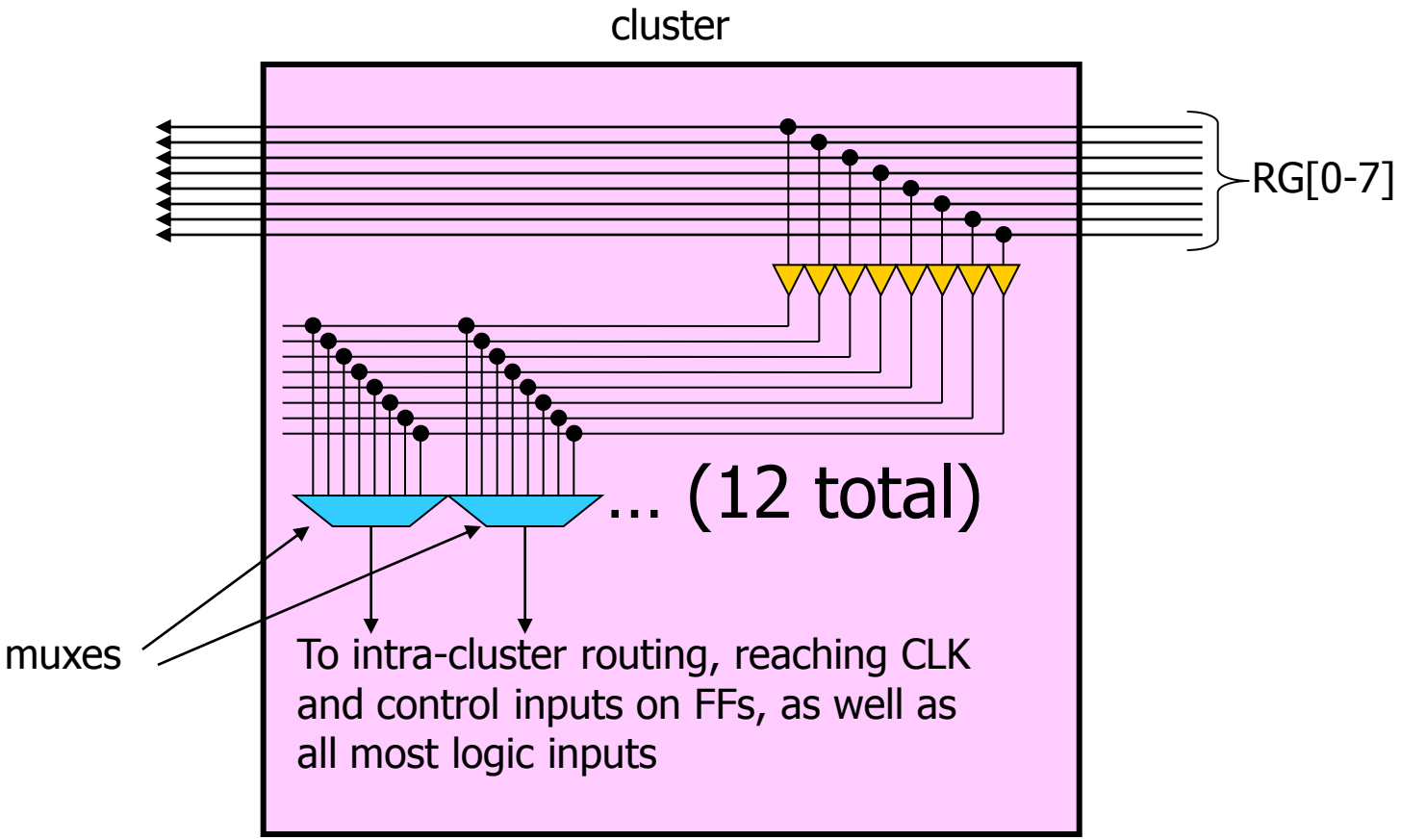
- Dedicated global IOs, are dual-use IOs which are capable of driving the global routing network or local routing network
  - Dedicated global IOs can be used to bring in external clock signals as inputs to the FPGA fabric
  - Dedicated global IOs can be used as regular IOs, as either input or output for any design signal
  - Dedicated global IOs can be configured as single-ended or differential
  - Dedicated global I/Os are located on each of the four sides of the FPGA fabric
  - The dedicated global IOs connect to fabric CCCs, VCCCs, and GBs through a hardwired connection
- Some of the dedicated global IOs have direct access to GBs, whereas others have to go through either VCCCs or CCCs to reach GBs
- Each fabric CCC has four dedicated global IOs as inputs and can drive up to four GBs.

Row Global Buffers (RGB)

- RGBs are located on the vertical stripes inside the FPGA fabric
- The global signals from the GBs are routed to RGBs which are then fed into the clusters through RGs
- Each GB has access to all RGBs available on the vertical stripes since the global network is segmented
- Each RGB is independent and can be driven by fabric routing in addition to being driven by GBs
  - This facilitates the use of RGBs to drive regional clocks spanning a small fabric area



RG[ ] wires to clusters



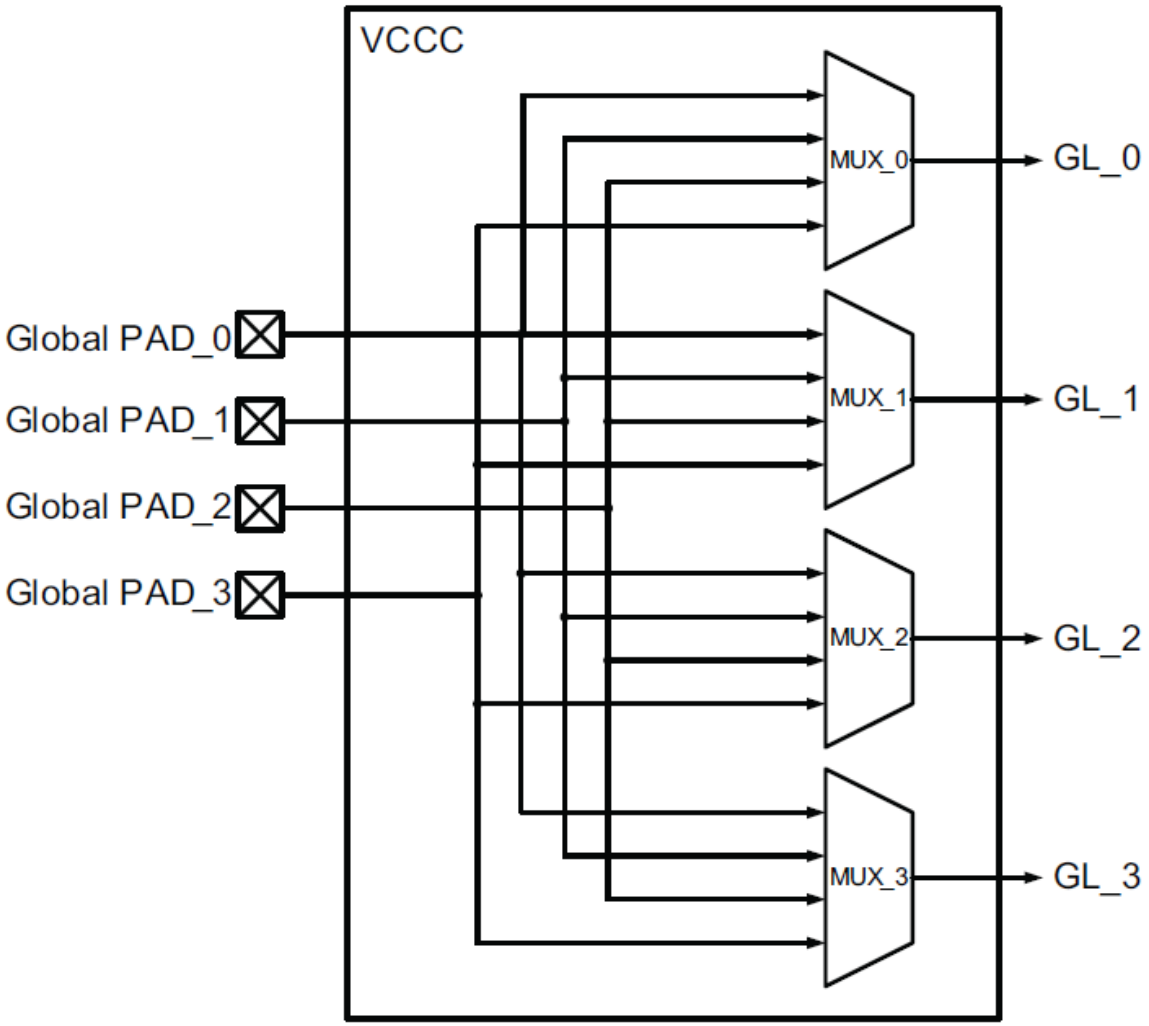
RGs and Local Clocks

- Each RG[ ] is independent:
  - It can be driven from any of the global signals.
  - It can also be driven locally from the fabric.
  - It can be tied off for power saving if not used.
- This feature allows RGs to be used for small local low skew clocks.
  - But not low latency!
- Larger (but still local) clocks may use several parallel RGs in adjacent cluster rows.

VCCCs

- VCCC = Virtual CCC
  - Number of (VCCC+CCC) = 8 for all family members with two stripes of globals
  - Number of CCC = Number of PLLs
- However the M2S025, M2S050 and M2S090 have 6 CCCs, so it must have two VCCCs
- **Why do we need VCCCs?**
  - In order not to create a new and different global signal architecture for every chip having different numbers of PLLs and CCCs
  - A VCCC is nothing more than a few muxes emulating the connectivity of CCC, but not its functionality
- **How do I use a VCCC?**
  - You don't explicitly choose to use one. The Designer SW will use it when necessary. Otherwise ignore them!

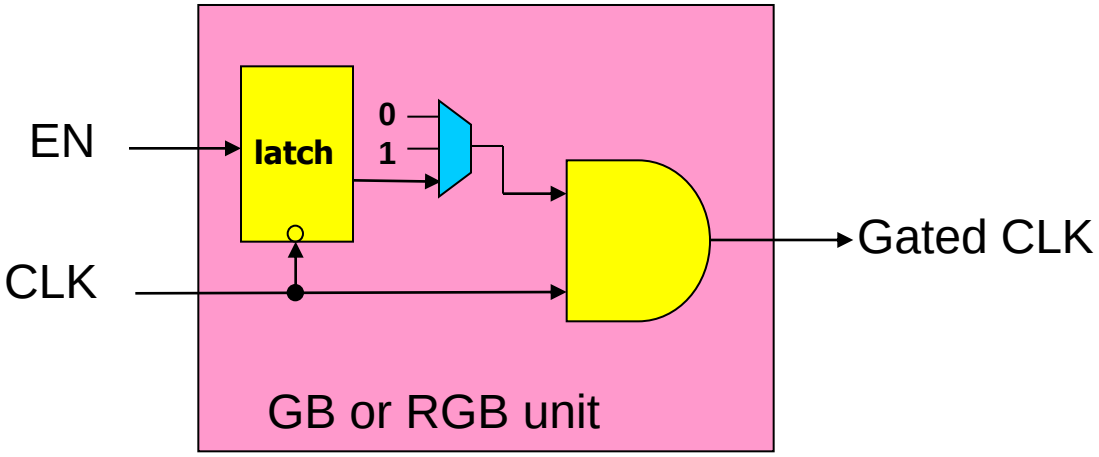
VCCC Block Diagram



The outputs of the VCCC blocks drive the global blocks of the FPGA fabric

Global Network Clock Gating

- The global network has built-in clock gating capability at both GB and RGB level to save power
- The clock gating can be enabled by instantiating a global clock buffer macro

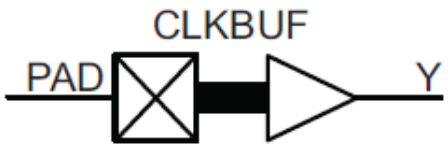
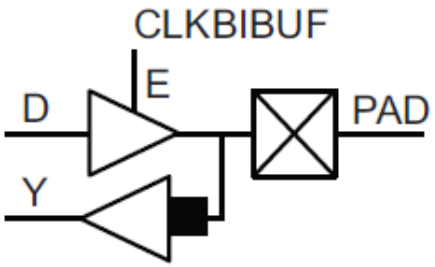
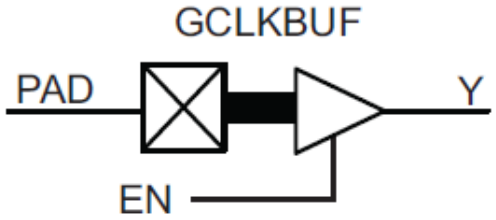


Latch is transparent when clock is low

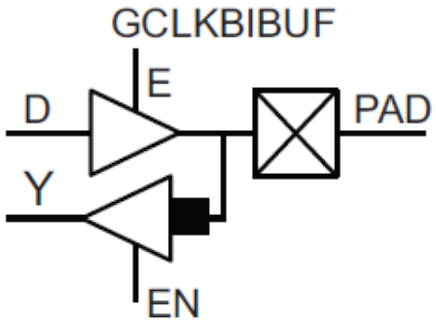
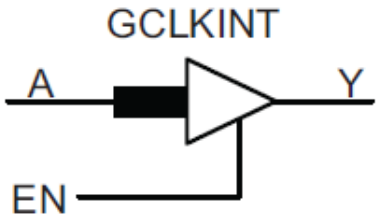
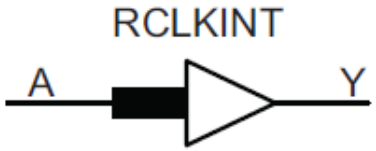
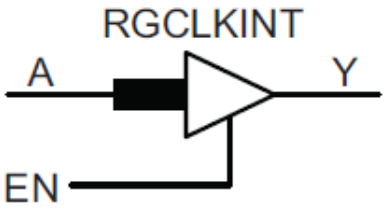
Global Macros

- Global macros can be used for assigning signals to the global network
- The CLKBUF, CLKBIBUF, GCLKBUF, and GCLKBIBUF macros route the clock coming from the dedicated global I/Os to GBs
  - Dedicated global I/Os which have direct access to GBs or through VCCCs to GBs are available as input pads for these macros
- The CLKINT and GCLKINT macros route the internal clock signals to GBs through FPGA fabric routing
- The RCLKINT and RGCLKINT macros route the internal clock signals to RGBs through FPGA fabric routing
- The gated clock buffer macros have an FPGA fabric routed control input (EN) to gate off the global clock

Global Macros (1)

Macro Name	Description	Functional Symbol
CLKBUF	Macro to drive a clock signal coming from input pad to a global buffer	
CLKBIBUF	Macro to drive a clock signal coming from bidirectional I/O pad to a global buffer	
CLKINT	Macro to drive a global buffer from FPGA fabric routing	
GCLKBUF	Input macro with control input (EN) to drive or gate off the input clock to a chip-level global net	

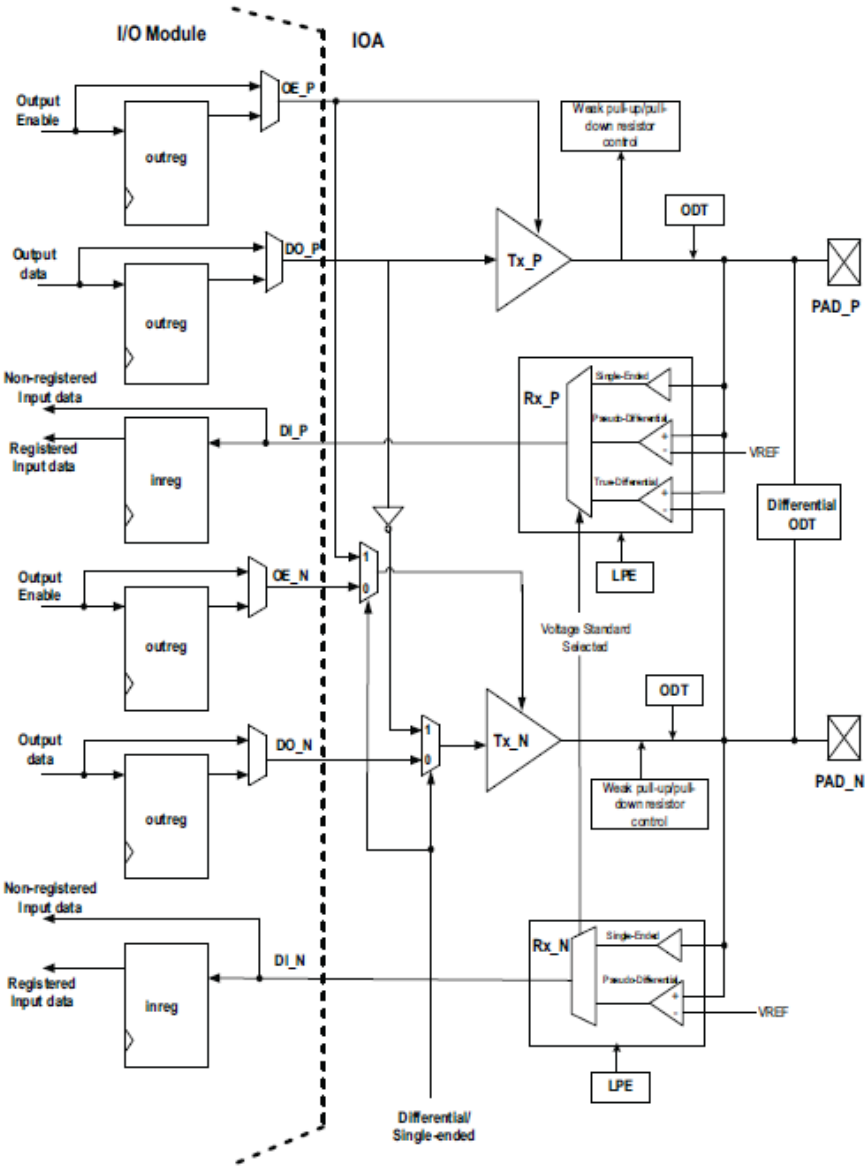
Global Macros (2)

Macro Name	Description	Functional Symbol
GCLKBIBUF	Bidirectional macro with control input (EN) to drive or gate off the input clock to a chip-level global net	
GCLKINT	Macro with control input (EN) to drive or gate off the input clock to a chip-level global net; input clock is routed through FPGA fabric.	
RCLKINT	Macro to drive a row global net from FPGA fabric.	
RGCLKINT	Macro with control input (EN) to drive or gate off the input clock to a row global net; input clock is routed through FPGA fabric.	

SmartFusion2 I/Os

I/O Module Block Diagram

Optional Input, Output and Output Enable registers



SmartFusion2 I/O Types

- Multi Standard I/Os (MSIO)
  - LVTTL/LVCMOS 3.3V
  - LVCMOS 1.2V, 1.5V, 1.8V, 2.5V
  - LVDS, MLVDS, mini-LVDS, RSDS differential standards
  - PCI
  - LVPECL (Receiver Only)
- MSIOD
  - Same as MSIO without PCI, 2.5V Max, no LVPECL
- DDR I/Os (DDRIO)
  - DDR, DDR2, DDR3, LPDDR, SSTL2, SSTL18, HSTL
  - LVCMOS 1.2V – 2.5V

MSIO Features

- Single-Ended, Pseudo-Differential Driver, True-Differential (LVDS) Driver
- Single Ended Receiver, Schmitt Trigger, True-Differential Receiver
- Programmable On-Die Termination
  - 300, 150, 100, 75, 60, 50 ohm ODT for LVDS
- Programmable Driver Strength
- Hot insertion capable
- 8 programmable drive strengths in LVTTL/LVCMOS operation
- Weak pull-up/down, bus keeper
- I/O low power signature mode and activity mode
- Performance
  - Up to 530 Mbps (LVDS)

MSIOD

- Standards Supported
  - LVCMOS 1.2V – 2.5V
  - LVDS, RSDS, Mini-LVDS
- Features
  - Single-Ended, Pseudo-Differential Driver, True-Differential (LVDS) Driver
  - Single Ended Receiver, Schmitt Trigger, True-Differential Receiver
  - Programmable On-Die Termination
  - Programmable Driver Strength
  - Pre-Emphasis for LVDS
- Performance
  - Up to 700Mbps (LVDS)

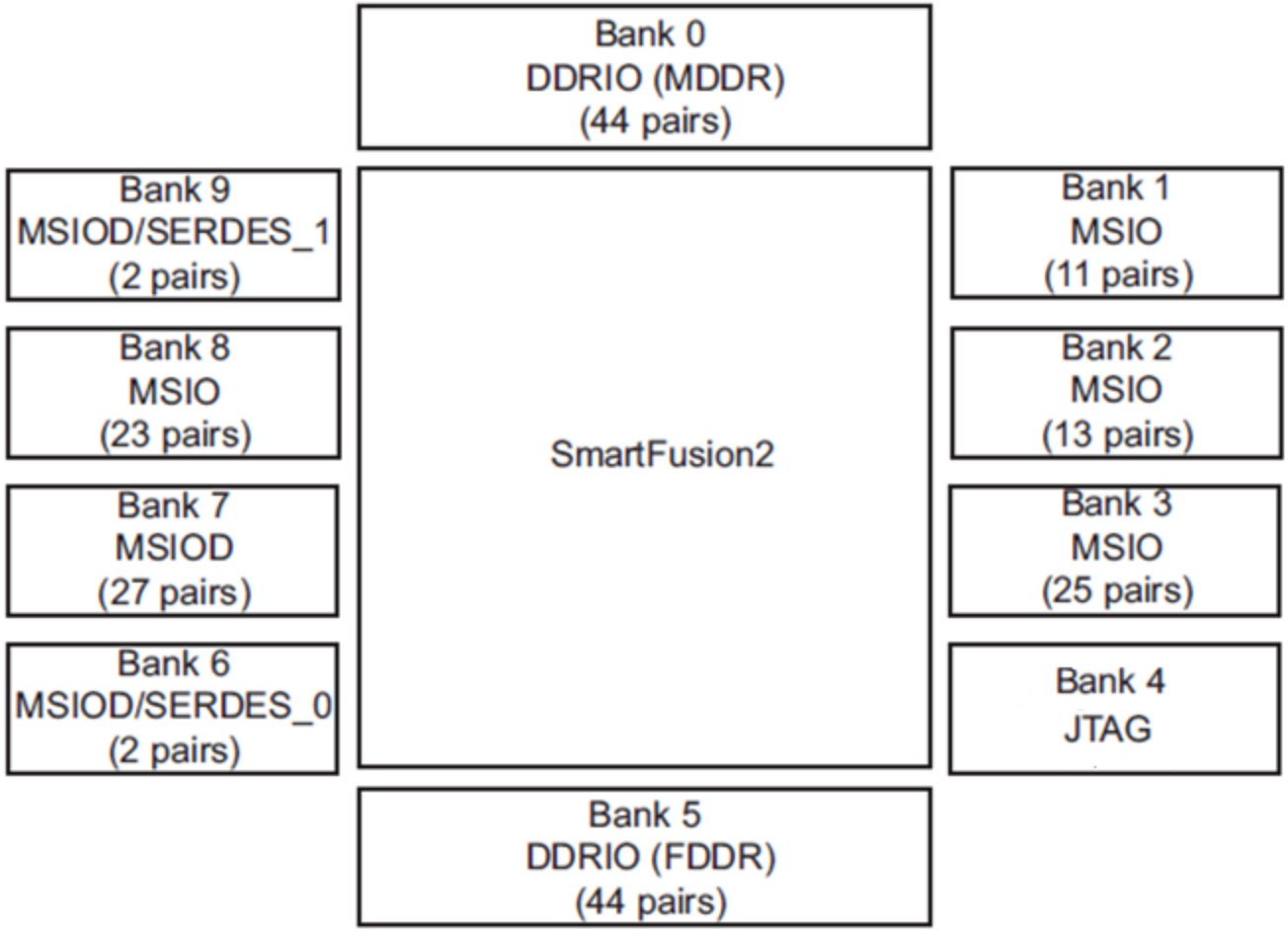
DDRIO Features

- 1.2 – 2.5V (Nominal) Operation
- External Reference Impedance Control
- Driver Resistor Linearization
- 1.5V and 1.8V On-Die Termination
- True-Differential, Pseudo-Differential, Single-Ended, and Schmitt Input Options
- Low power signature mode and activity mode

I/O Type Summary

I/O Standards	I/O Types		
	MSIO	MSIOD	DDRIO
Single-Ended I/O			
LVTTL 3.3V	Yes	-	-
LVCMOS 3.3V	Yes	-	-
PCI	Yes	-	-
LVCMOS 1.2V	Yes	Yes	Yes
LVCMOS 1.5V	Yes	Yes	Yes
LVCMOS 1.8V	Yes	Yes	Yes
LVCMOS 2.5V	Yes	Yes	Yes
Voltage-Referenced I/O			
HSTL 1.5V	-	-	Yes
SSTL 1.8V	-	-	Yes
SSTL 2.5V	Yes	Yes	Yes
SSTL 2.5V (DDR1)	Yes	Yes	Yes
SSTL 1.8V (DDR2)	-	-	Yes
SSTL 1.5V (DDR3)	-	-	Yes
Differential I/O			
LVPECL (input only)	Yes	-	-
LVDS 3.3V	Yes	-	-
LVDS 2.5V	Yes	Yes	-
RSDS	Yes	Yes	-
BLVDS	Yes	Yes	-
MLVDS	Yes	Yes	-

M2S050 I/O Bank Location



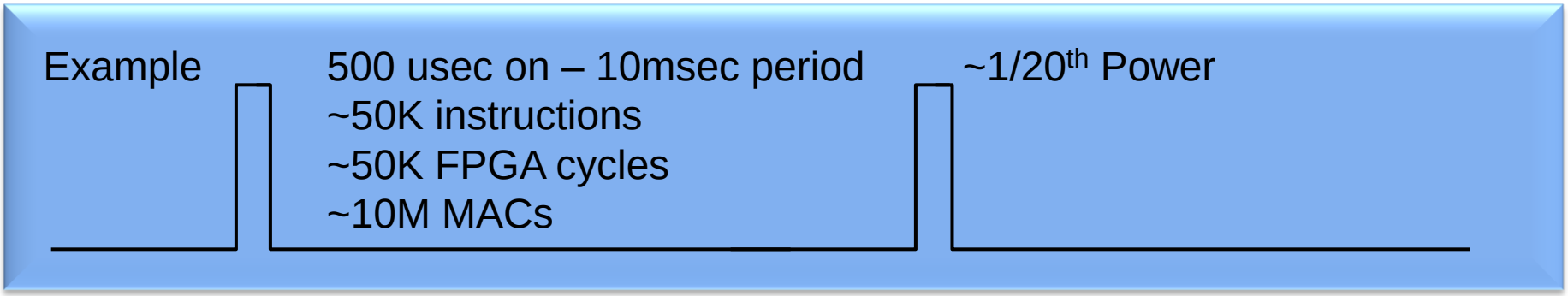
SmartFusion2 Power Management

Power Saving Options

- Flash*Freeze Mode
  - Initiated by user (to put fabric and IOs into low power state)
  - Initiated from fabric or from MSS
- MSS Power Modes
  - Sleep
  - Peripheral Soft Reset
- SERDES Block Power Modes:
  - L1 – Transmitter Disabled
  - L2 – SERDES Disabled

Flash*Freeze Mode

- Simple command to initiate low power mode
  - Entry to Flash*Freeze 100 µs
    - Retains all register and SRAM state in Flash*Freeze mode
    - Set I/O state during Flash*Freeze mode
    - MSS can be operational during Flash*Freeze with low frequency clock
      - I/O's associated with MSS peripherals can be operational
  - Exit Flash*Freeze mode in 100 µs
  - 1 mW static power consumption in Flash*Freeze mode



During Flash*Freeze

- Fabric
  - All state information stored in suspend latches (which remain powered)
  - RAM blocks can be put into a Suspend Mode or a Sleep Mode
    - SRAM rows can be configured for Suspend or Sleep mode independently
    - Suspend mode: RAM contents are retained
    - Sleep mode: RAM contents are not retained
  - Remainder of fabric un-powered

- Digital IO
  - IO configuration information saved
  - IOs may be set to monitor for incoming activity or signature match
  - User designated IOs in MSS can remain unaffected by Flash Freeze
    - MSS is operational in FF Mode

M2S050 Power Numbers

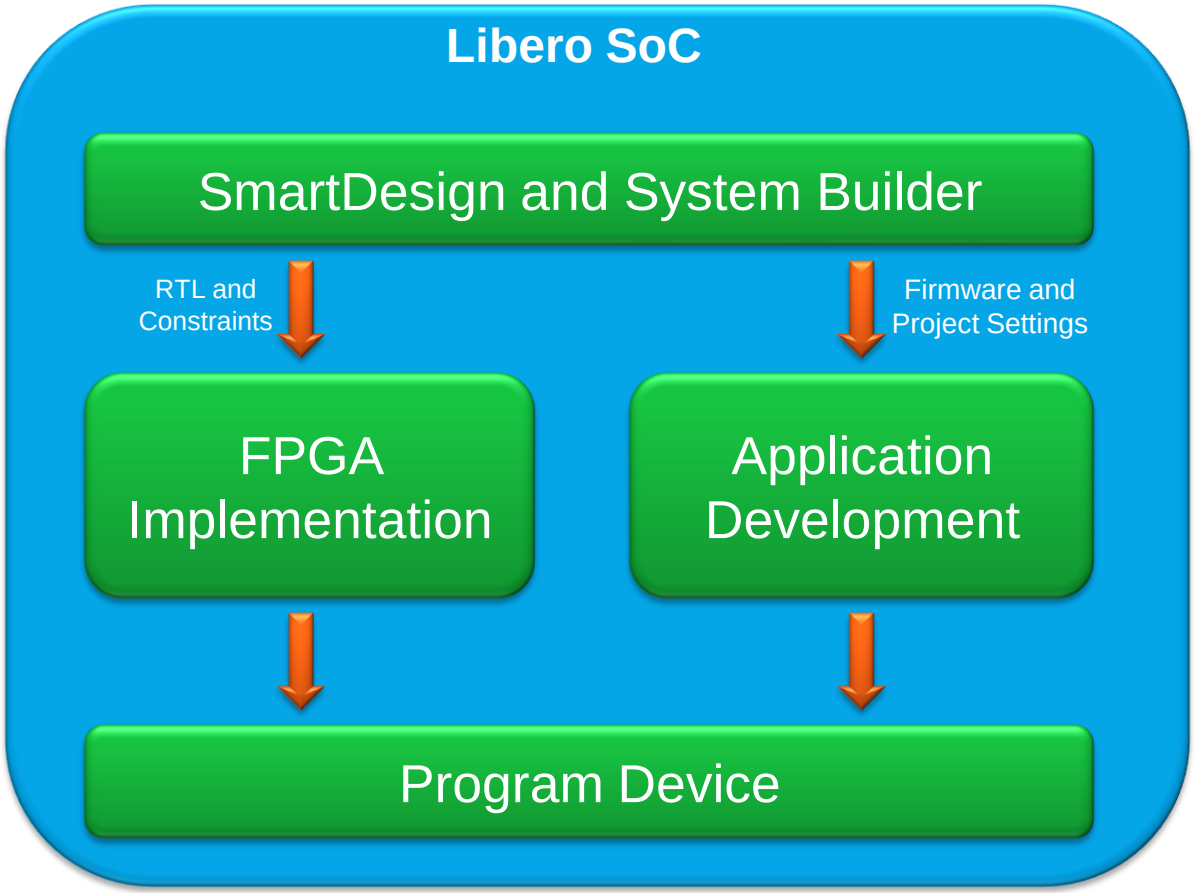
- Static
  - 10 mW
- Flash*Freeze
  - < 1 mW

Estimates, Room Temp, MSS running at 32KHz, No I/O power

Design Tools and Development Boards

SmartFusion2 Design Creation

- Libero SoC v11.5
  - Integrated environment for System-on-Chip Design

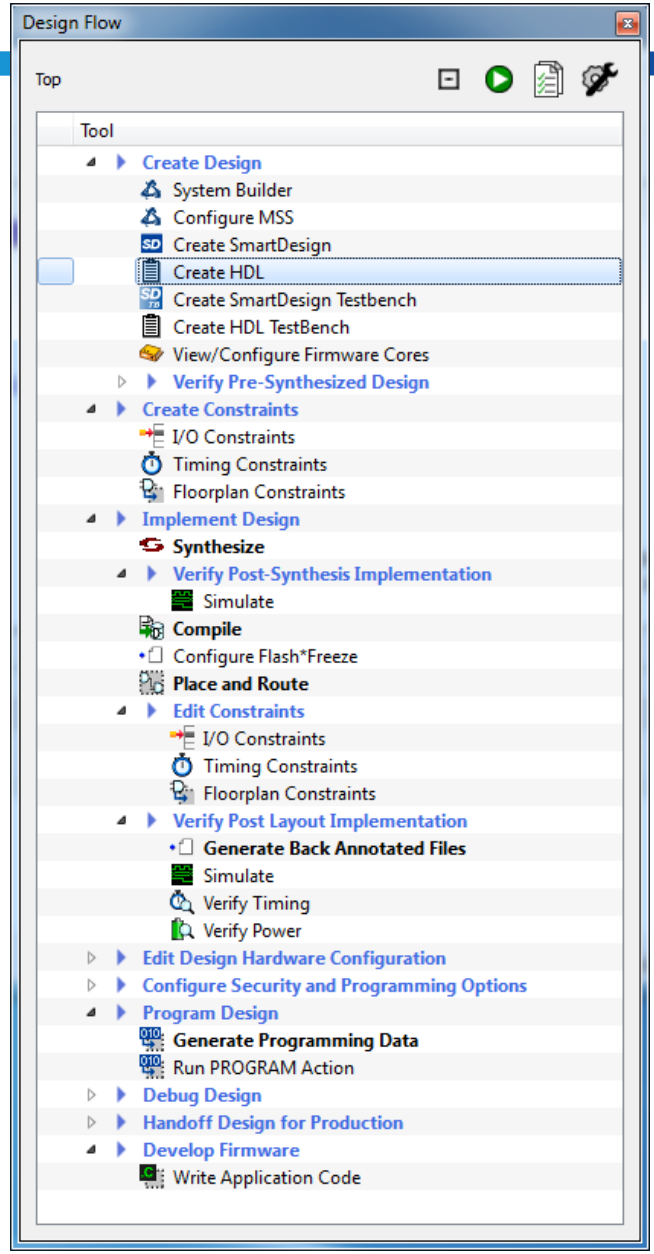


Libero SoC v11.5 Content

- Libero SoC v11.5 includes the following:
  - Microsemi Firmware Catalog 11.5
  - Microsemi FlashPro 11.5
  - ModelSim ME 10.3c
  - Synopsys Synplify Pro ME I-2014.03M-SP1
  - IP Catalog with approximately 50 IP Cores

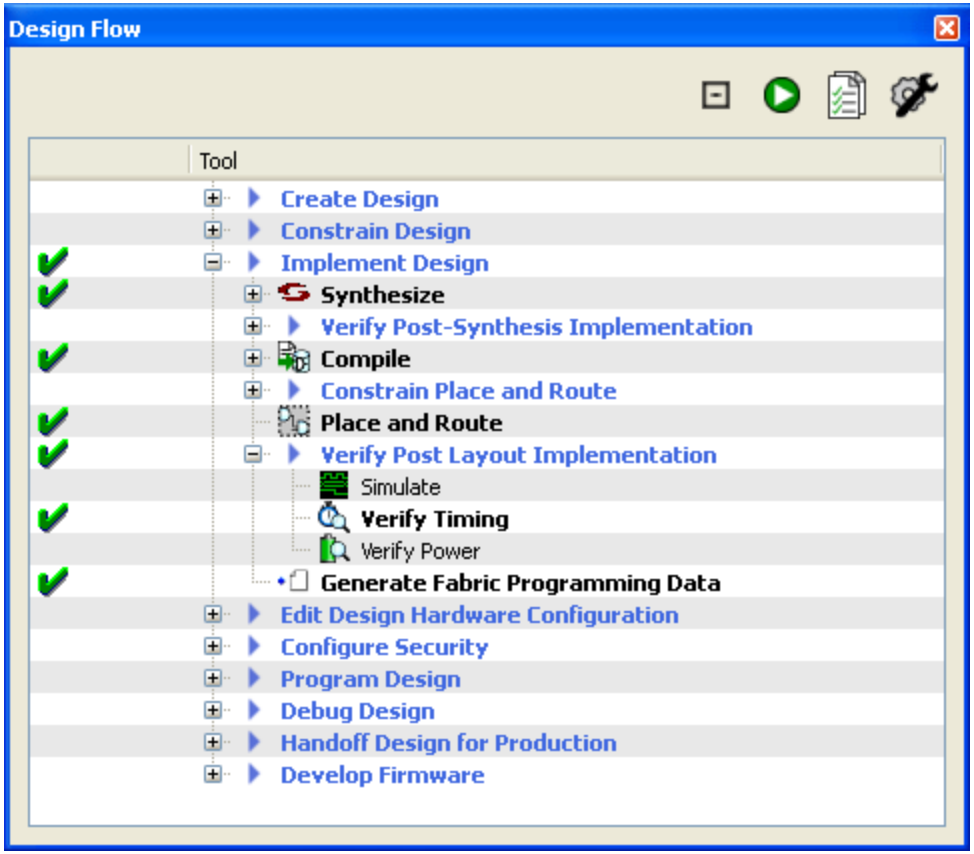
Libero SoC Design Flow Window

- Single Window Provides Access to tools and shows Design Status
- Tools Can be Run in Background or Launched from Design Flow Window or Menus
- Project Flow Window Displays:
  - Tools
    - Design Creation
      - Includes option to import files into the project
    - Stimulus Creation
    - Processing
      - Simulation
      - Synthesis
      - Place and Route
  - Transitions
  - Current State
  - Tool Tips



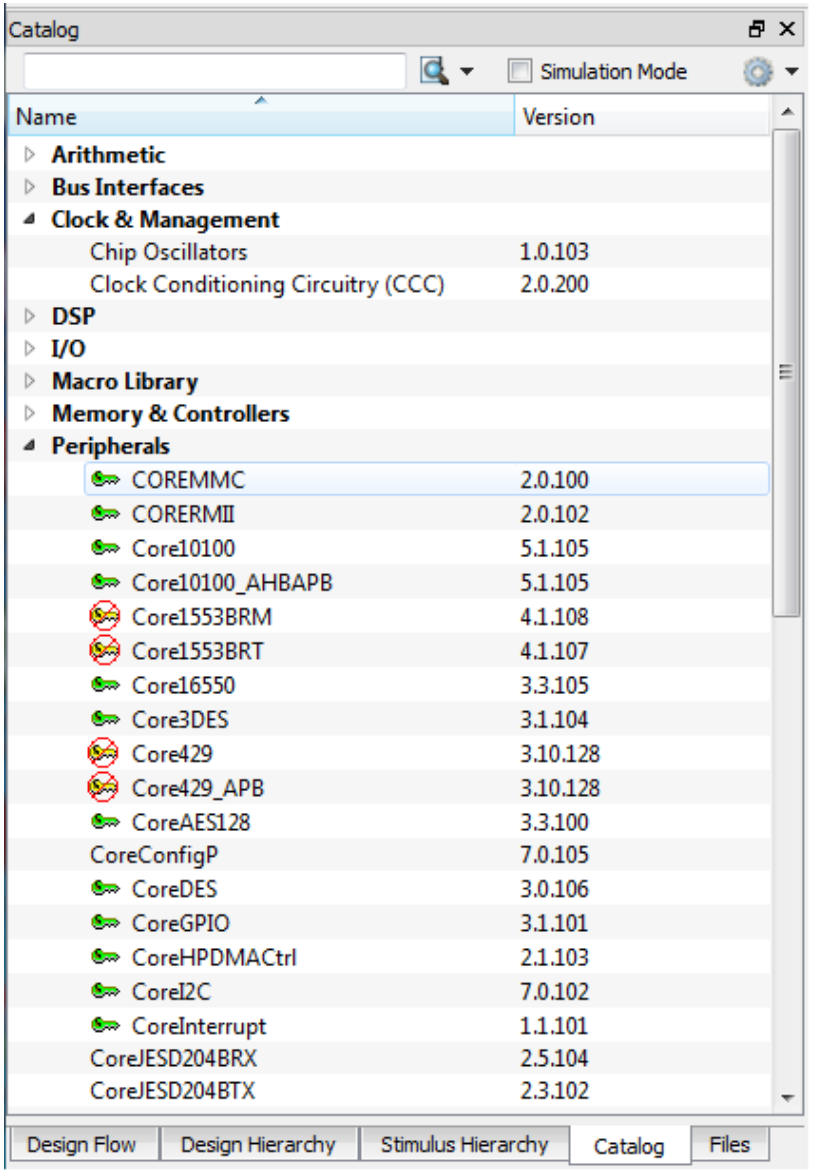
Design Flow Window: Design Status

- Check marks indicate completed steps in Design Flow



Libero SoC IP Catalog

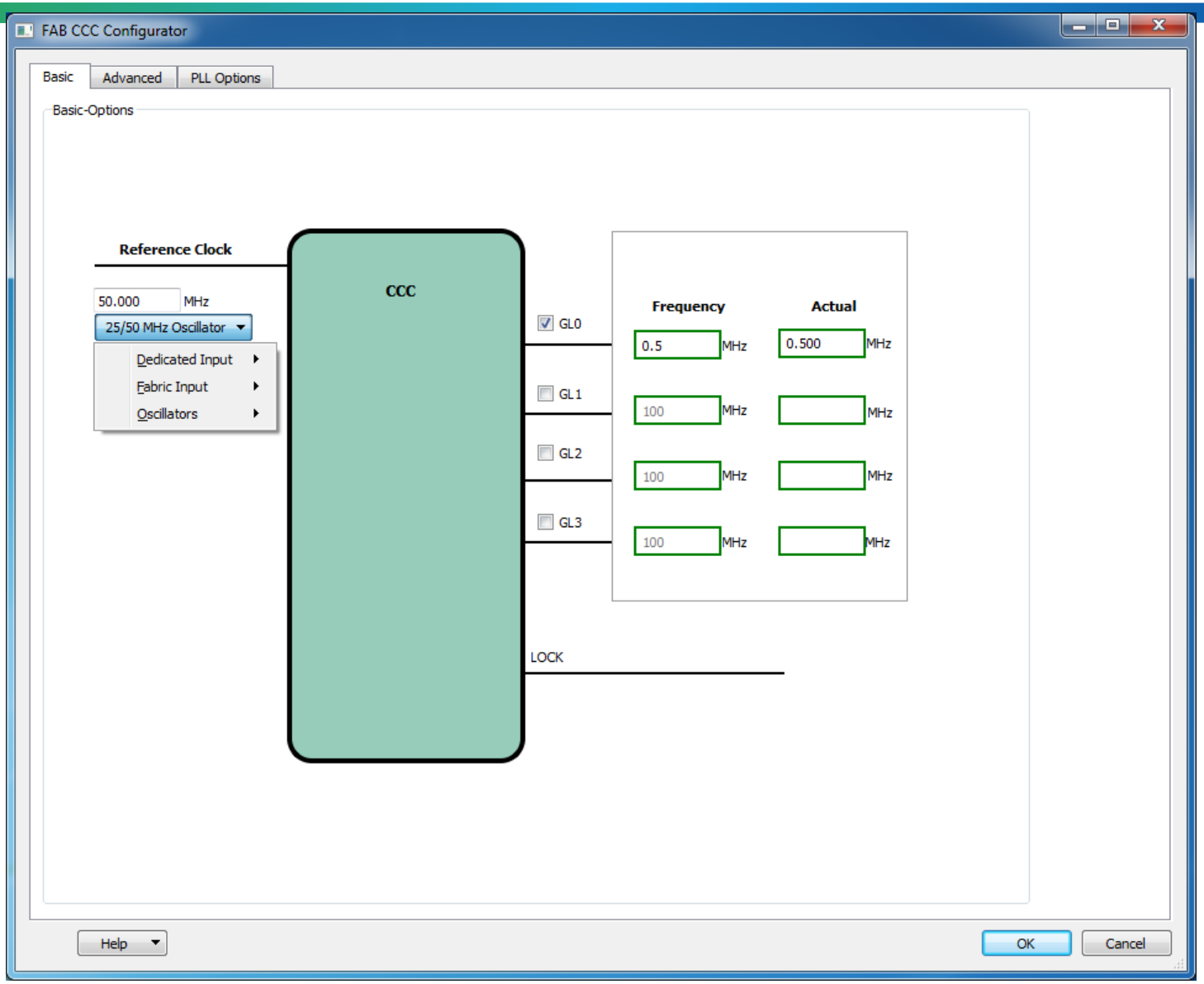
- Library of Proven Configurable Cores
- IP Based Design Flow
  - Access to 50+ IP Cores
- Configure and Instantiate Directly from Libero
  - No need to launch separate tools
  - No need to re-enter common information
  - No need to import generated cores
- Modify Generated Cores Directly from Libero
  - Change Configurations
  - Change Versions



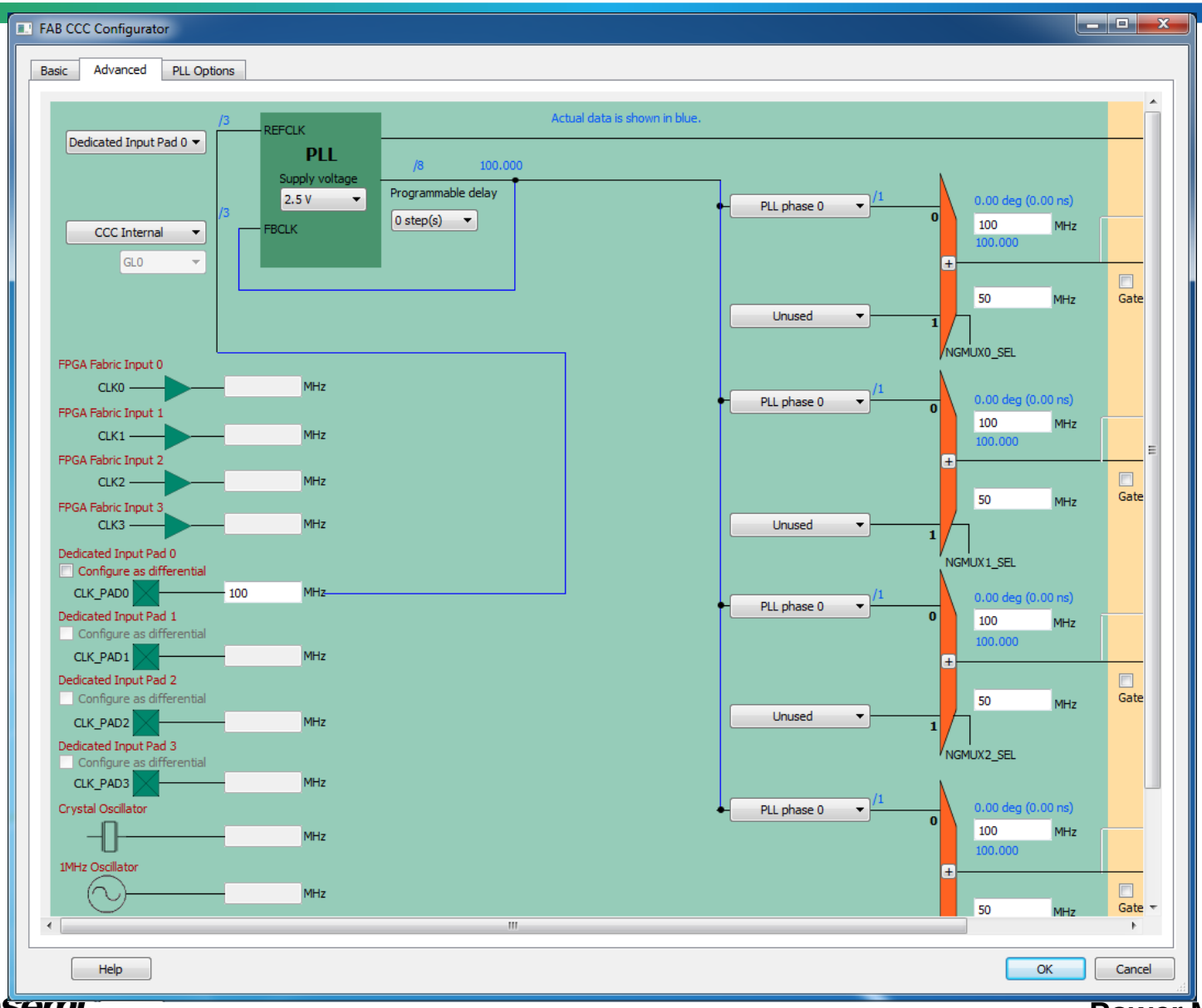
Configurators

Fabric CCC Configurator Basic Options

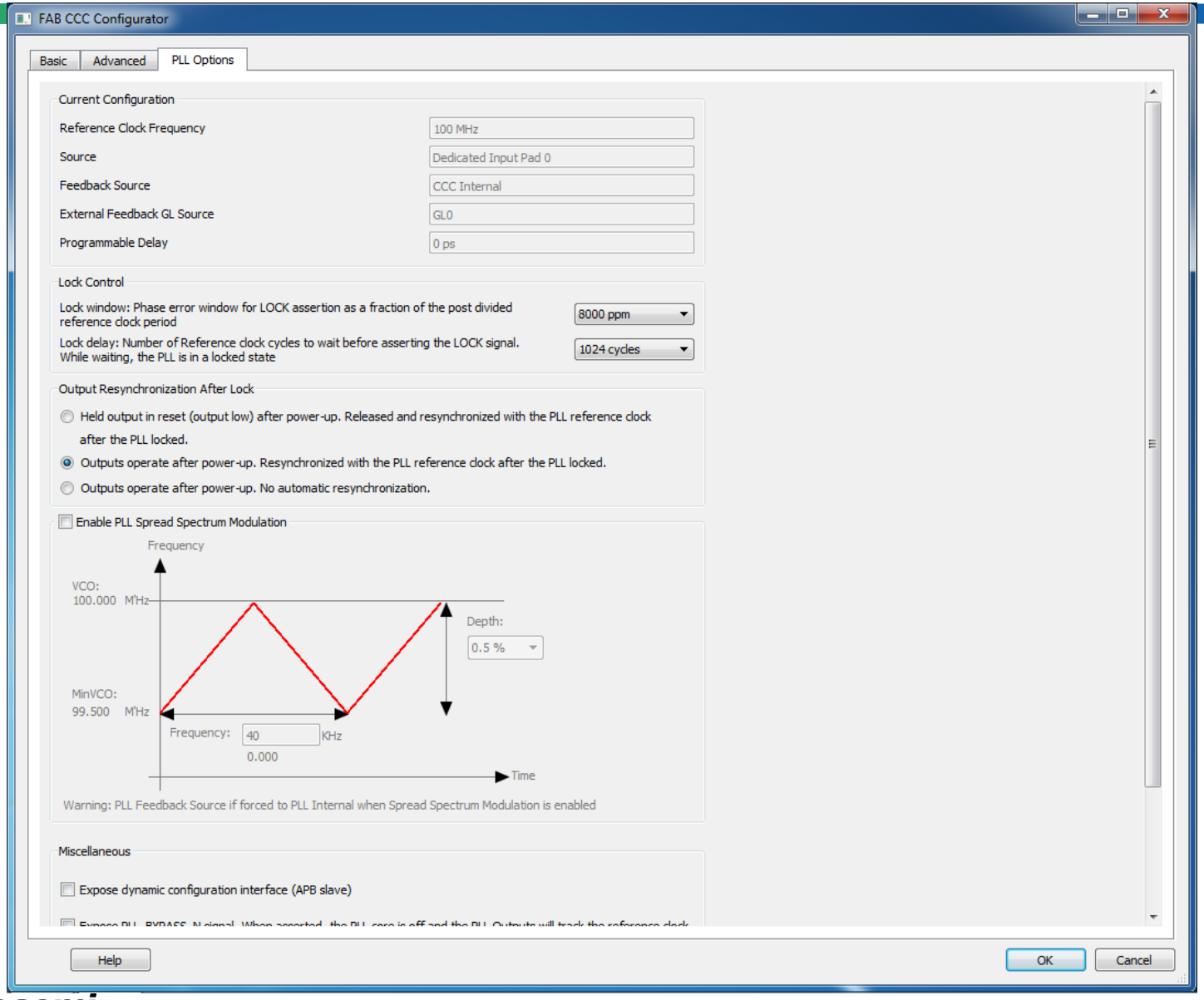
CCC = Clock Conditioning Circuit



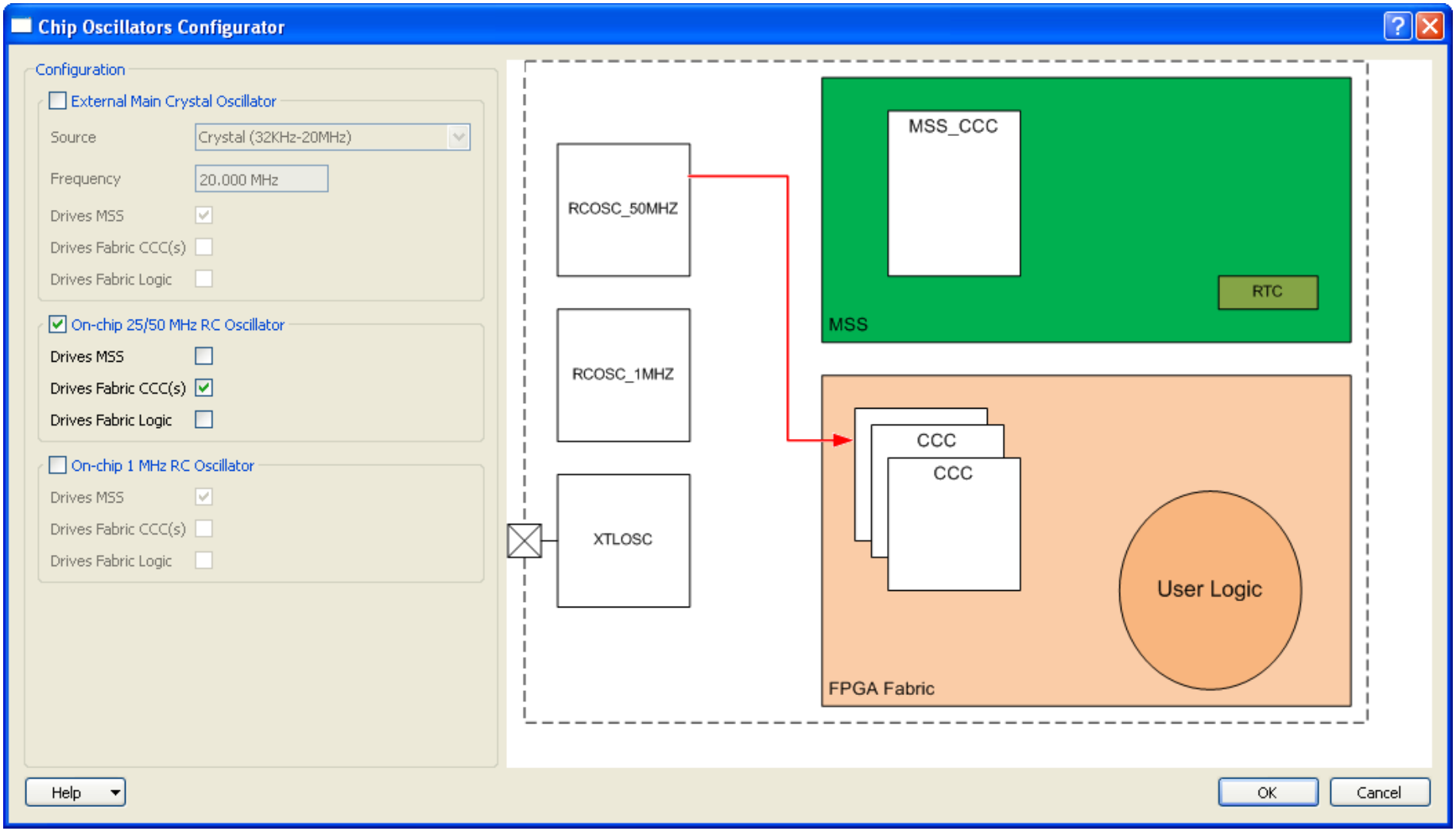
Fabric CCC Configurator Advanced Options



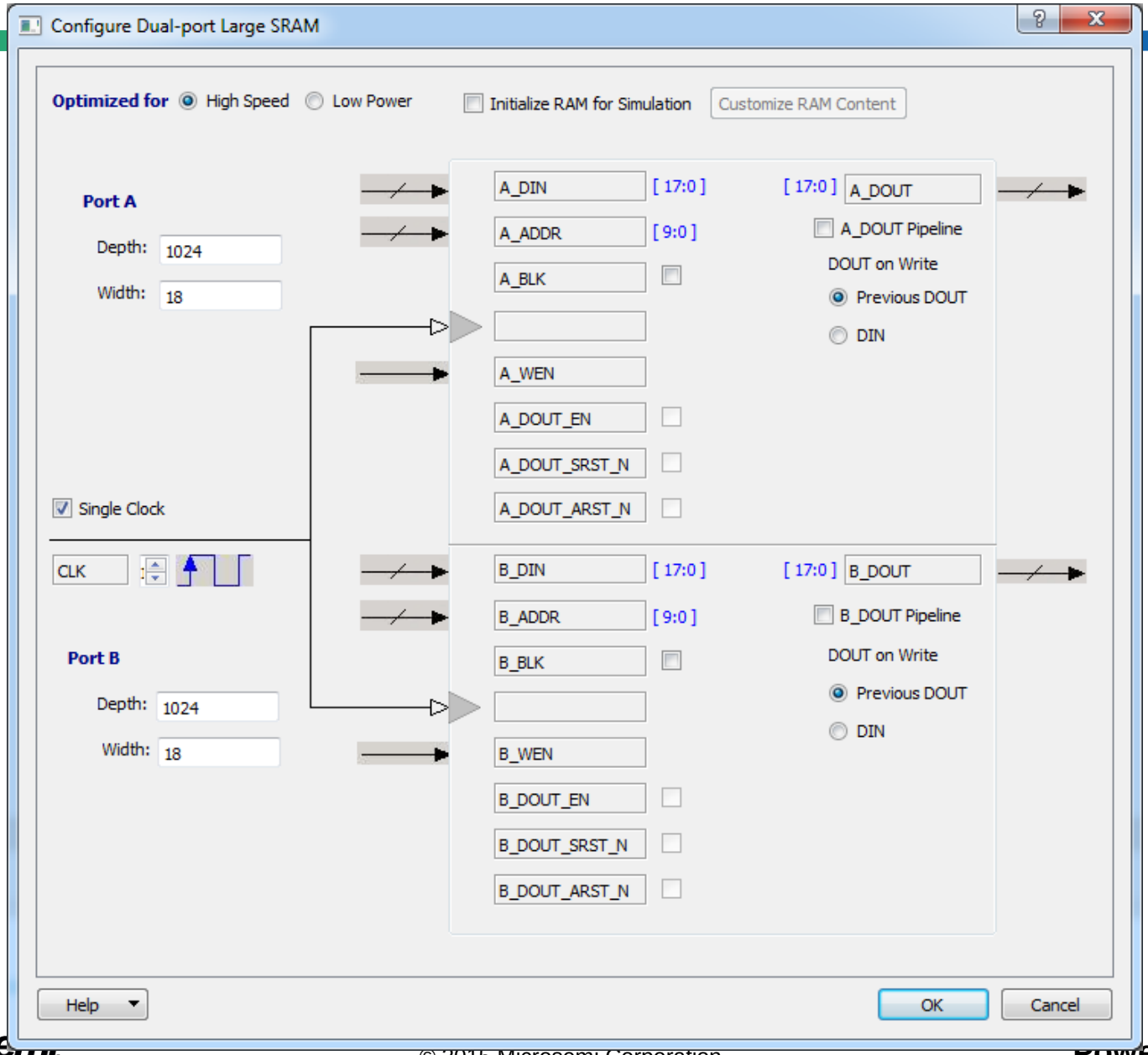
Fabric CCC Configurator PLL Options



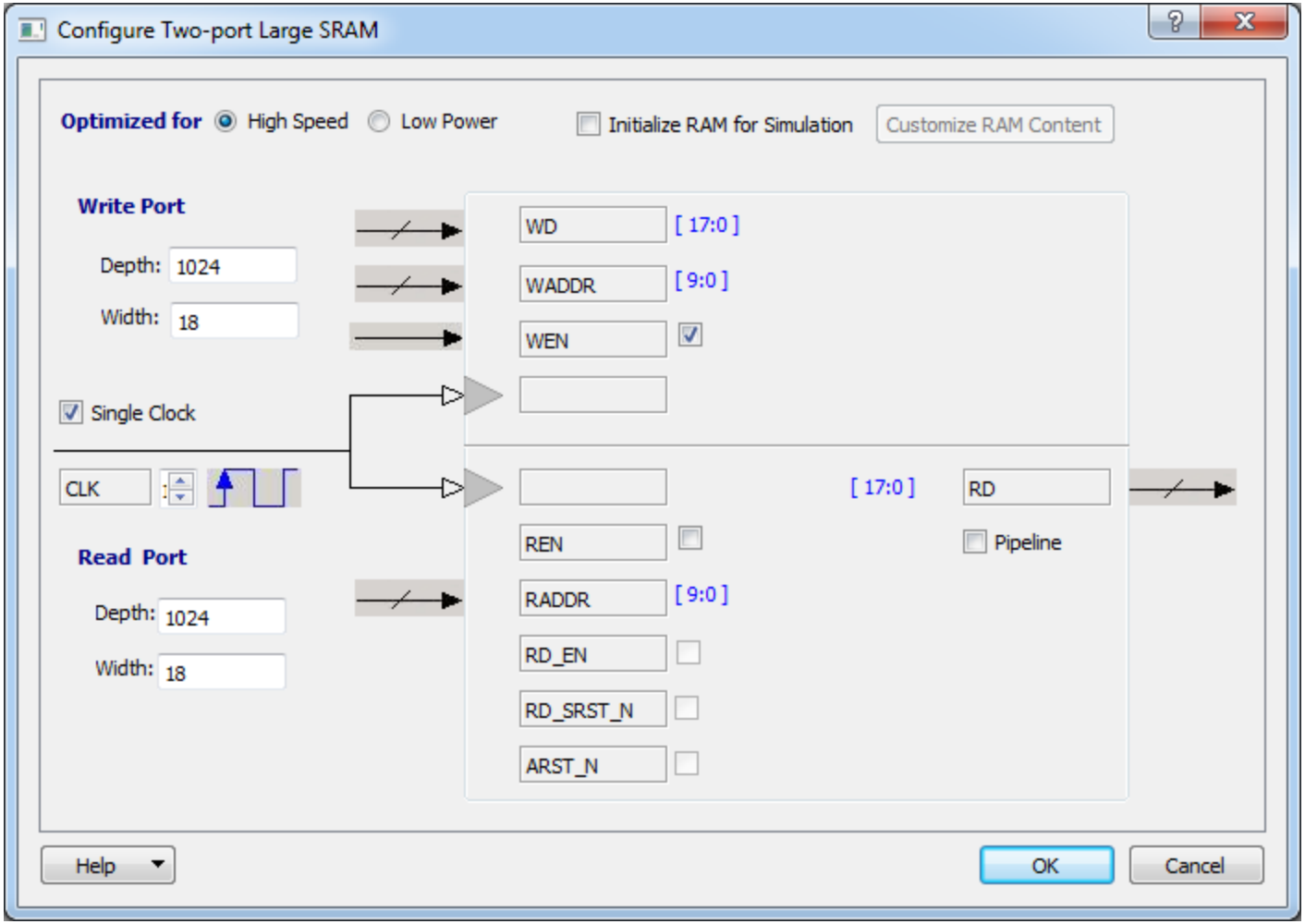
Chip Oscillators Configuration



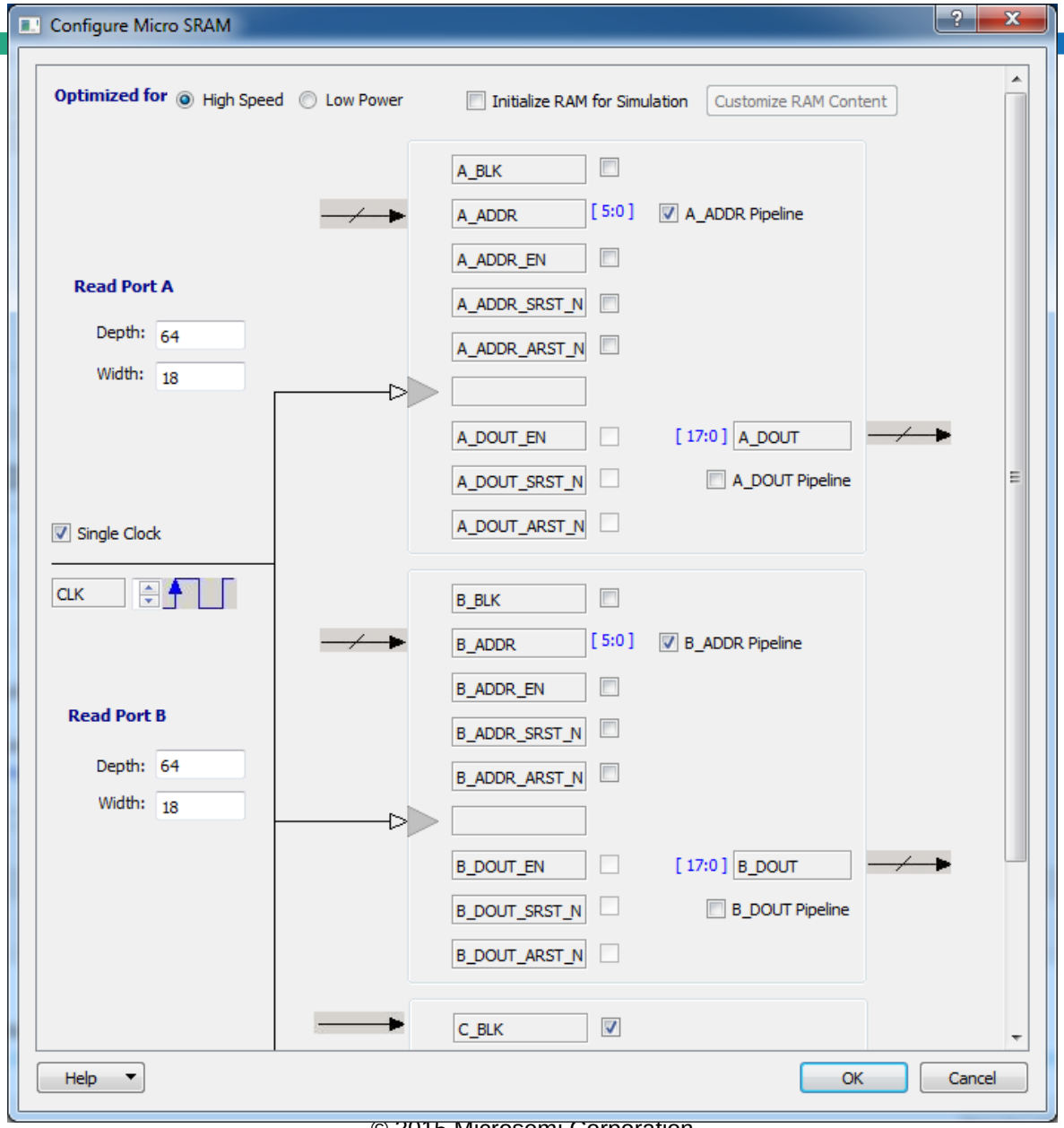
LSRAM Dual-Port Configurator



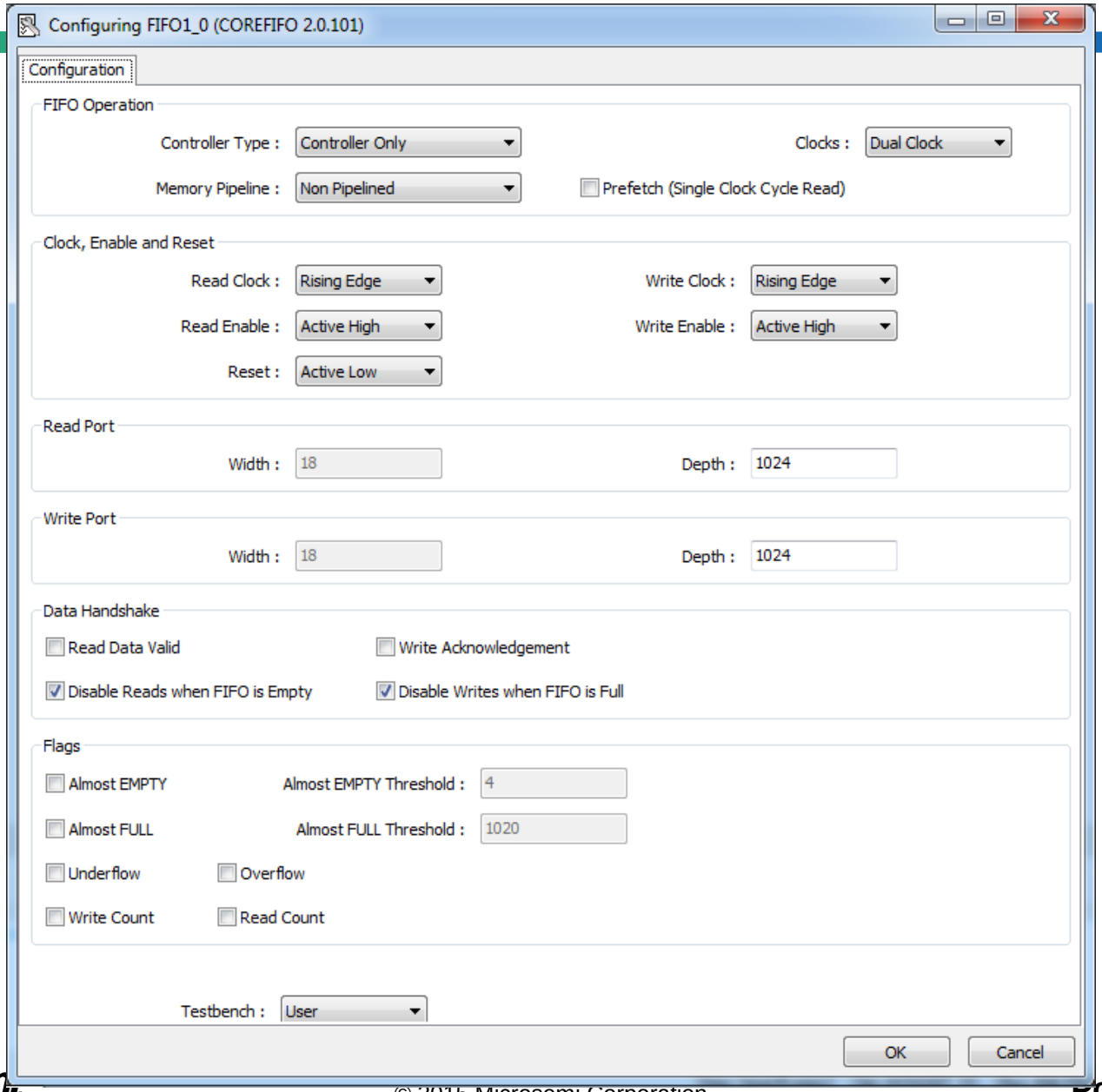
LSRAM Two-Port Configurator



uSRAM Configurator



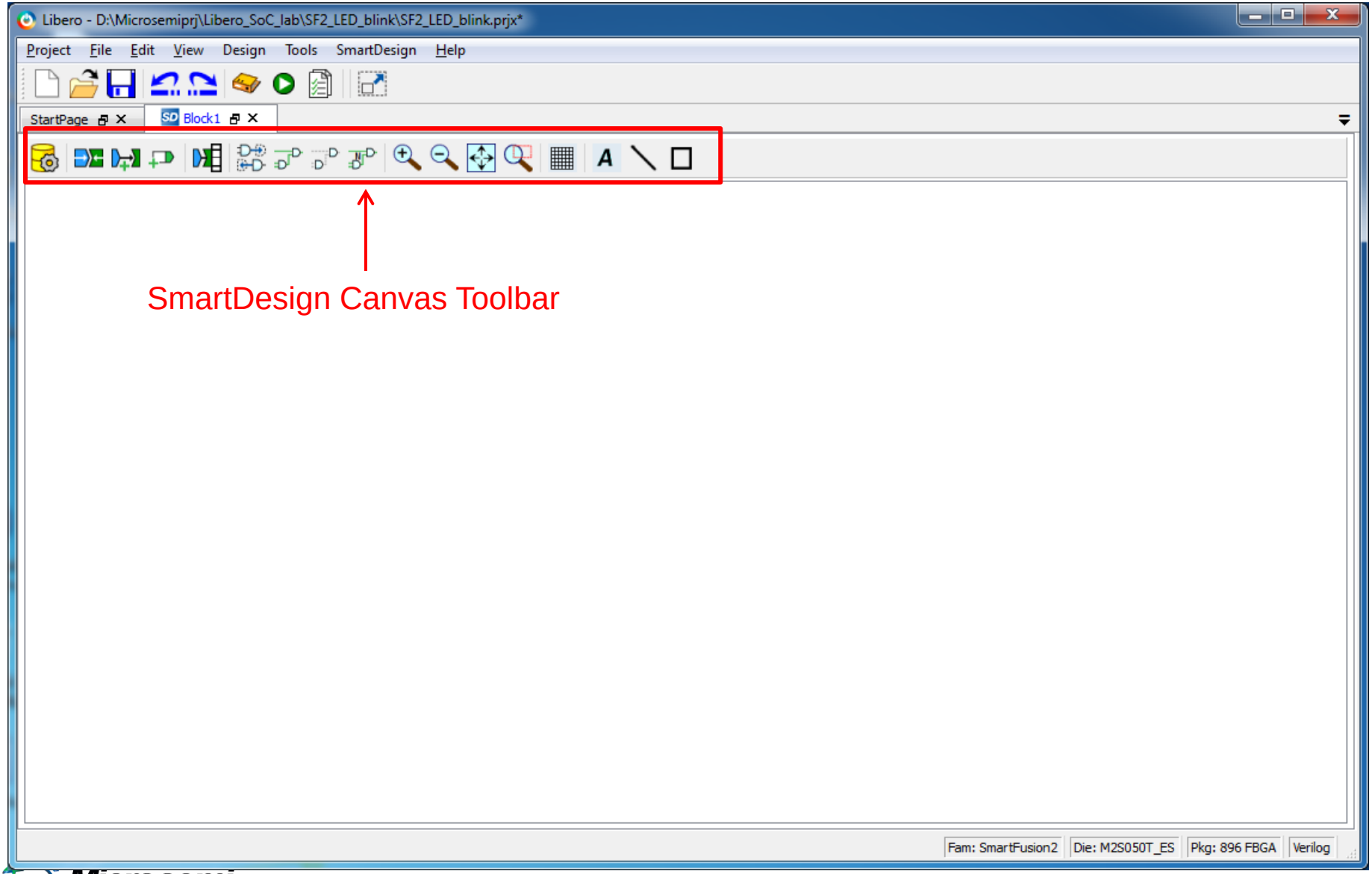
CoreFIFO Configurator



What is SmartDesign?

- Powerful Block-based Visual Design Creation Tool
  - Instantiate blocks from a variety of sources
    - Configurators, DirectCore IP, User HDL, Companion Cores, Microsemi library cells, etc.
  - Supported for all platforms
- Simple and Intuitive Design Creation
  - Auto Connect
  - Fast manual connectivity between blocks
  - Hierarchical design support
- Design Rule Checking (DRC)
  - Checks rules to guarantee correct by construction design
    - Connectivity errors
    - Configuration errors
    - Special silicon rules
- SoC Features
  - Auto Connect
    - Clocks and resets for processors and peripherals
    - Other known DirectCore connections
  - Memory Map Configuration Dialog
  - Testbench and Bus Functional Model (BFM) script generation

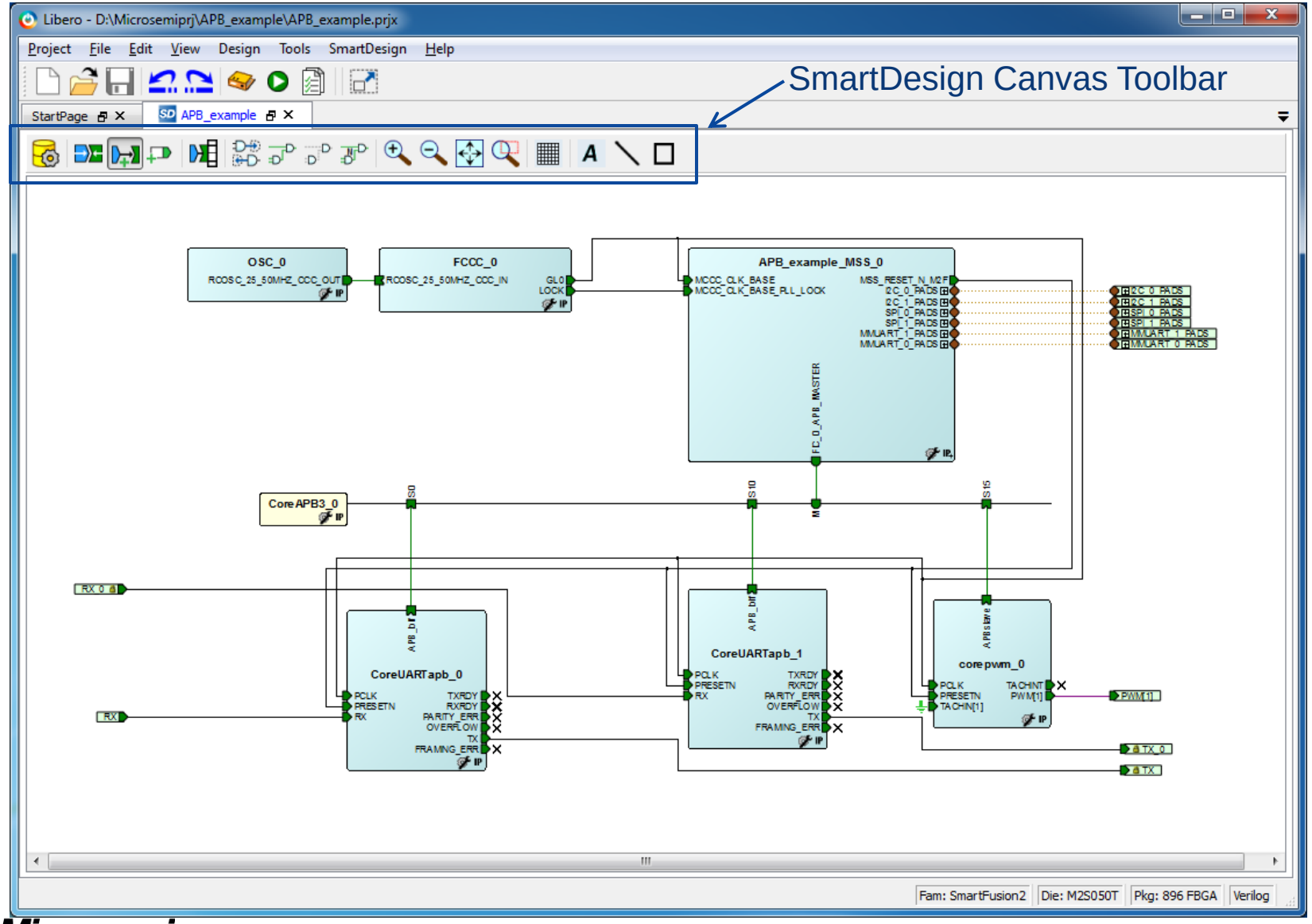
SmartDesign Canvas



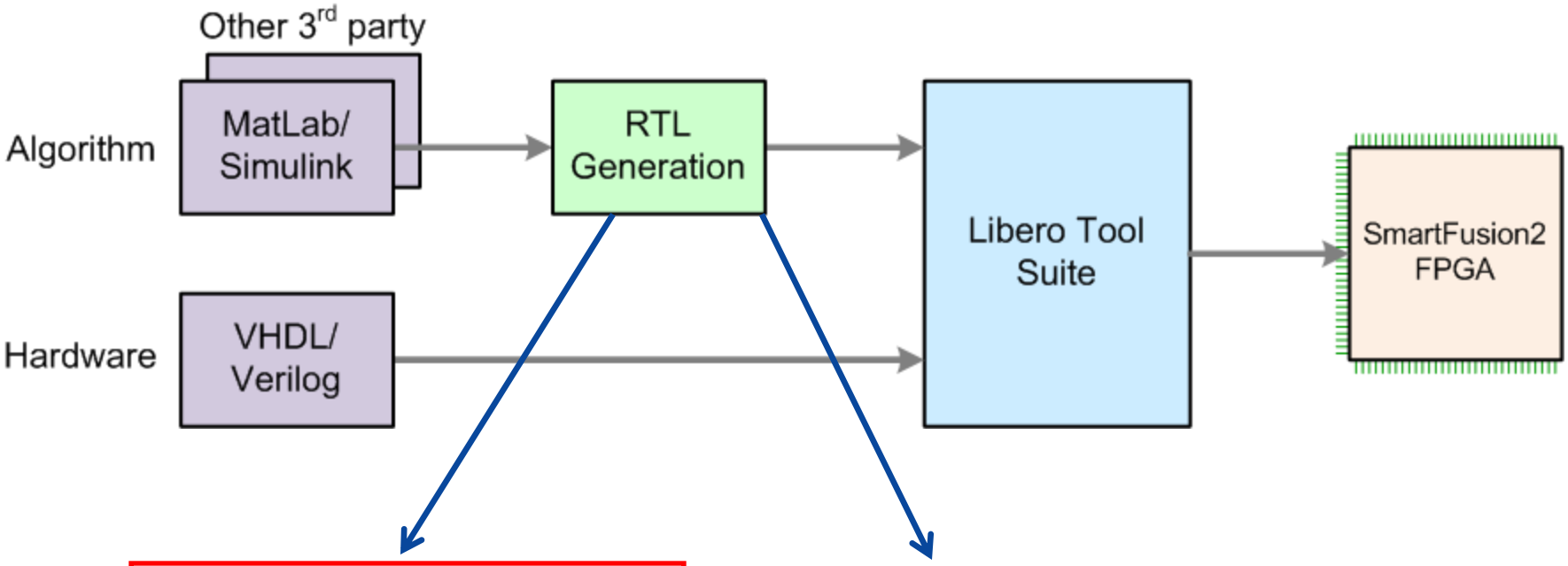
SmartDesign Canvas Features

- Instance pins are displayed on canvas.
- Connections are shown using nets
  - Displaying of Nets is optional
  - Selective enabling / disabling of showing nets
- Drag and Drop directly from the Catalog into the Canvas
- All Design Operations Available in the Canvas
  - Connect / Disconnect
  - Promote To Top
  - Tie Low / Tie High / Tie Constant / Inversion
  - Float
  - Split (if bus)
  - Group

Complete Design



DSP Design Flows



- IP Core RTL Generators
  - CoreFIR
  - CoreFFT
  - CoreRSDEC
  - CoreRSENC

- Synphony MC (formerly Synplify® DSP)
  - CoreFIR
  - CoreFFT

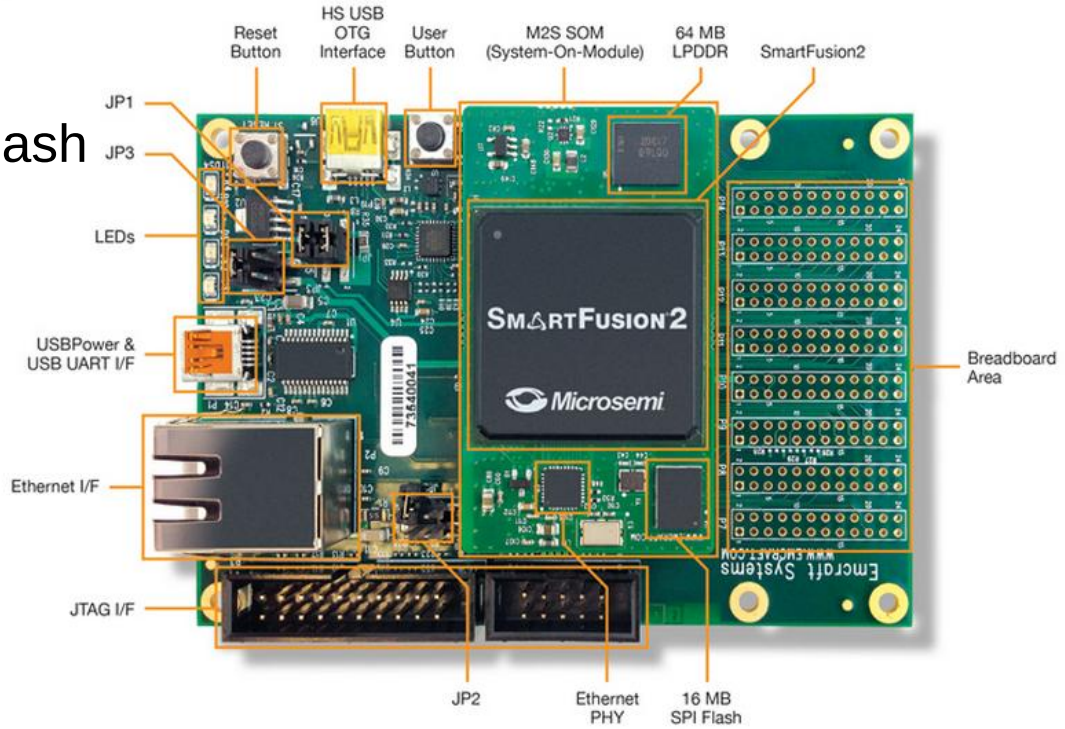
Development Boards

Overview

Development Kit/Boards	Device	Description
SmartFusion2 Starter Kit (SF2-484-STARTER-KIT or SF2-STARTER-KIT)	M2S050-FGG484 or M2S010-FGG484	uClinux based Low cost, Embedded design platform
SmartFusion2 Evaluation Kit (M2S-EVAL-KIT)	M2S025T-1FGG484	Low cost, SERDES, PCIe system development platform
SmartFusion2 Security Evaluation Kit (M2S090TS-EVAL-KIT)	M2S090TS-1FGG484	Low cost, Security, SERDES, PCIe system development platform
SmartFusion2 Advanced Development Kit (M2S150-ADV-DEV-KIT-ES)	M2S150TS-1FCG1152ES	Development platform for 150K LE devices focused on Security and FMC Expansion
IGLOO2 Evaluation Kit (M2GL-EVAL-KIT)	M2GL010T-1FGG484	Low cost, SERDES, PCIe system development platform

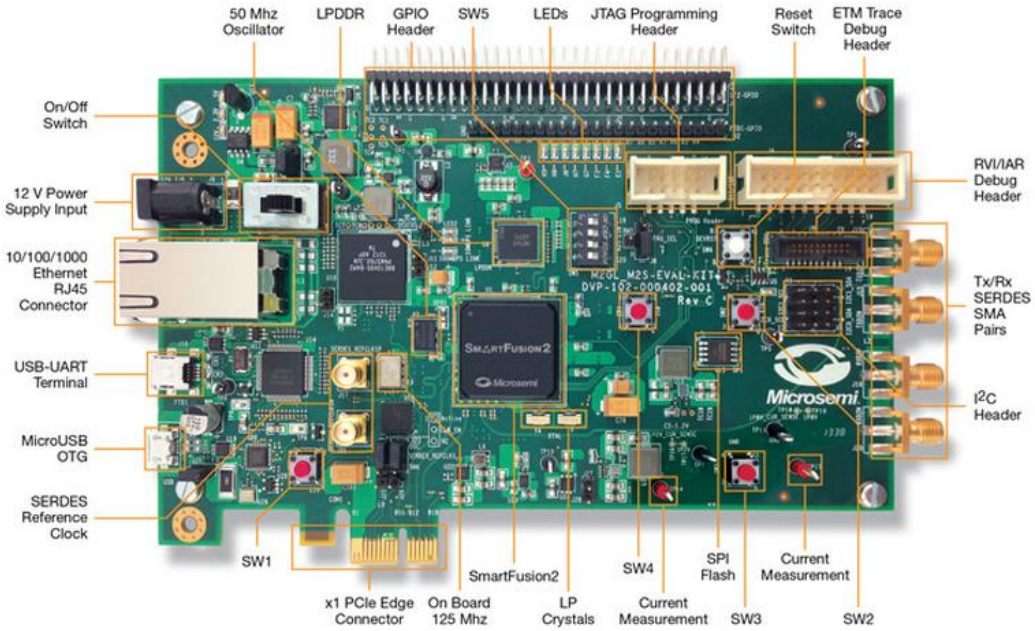
SmartFusion2 Starter Kit

- Two variants –
  - SF2-STARTER-KIT with **M2S050-FGG484**
  - SF2-484-STARTER-KIT with **M2S010-FGG484**
- Features –
  - 10/100 Ethernet
  - USB OTG interface
  - 64MB LPDDR, 16MB SPI flash
  - USB based Wi-Fi module
  - Breadboard expansion
- Price
  - List - $299
- Part number
  - SF2-STARTER-KIT
  - SF2-484-STARTER-KIT



SmartFusion2 Evaluation Kit

- Applications
  - Develop and test PCI Express Gen2 x1
  - Evaluate SerDes transceiver using SMA Pairs
  - Power measurement of the SmartFusion2 SoC FPGA
  - Create a working PCIe link quickly with the PCIe Control Plane Demo
- Features
  - 512MB LPDDR, 64MB SPI flash
  - x1 PCIe Edge connector
  - Four SMA connector
  - 10/100/1000 Ethernet
  - GPIO expansion
- Price
  - List - $399
- Part number
  - M2S-EVAL-KIT

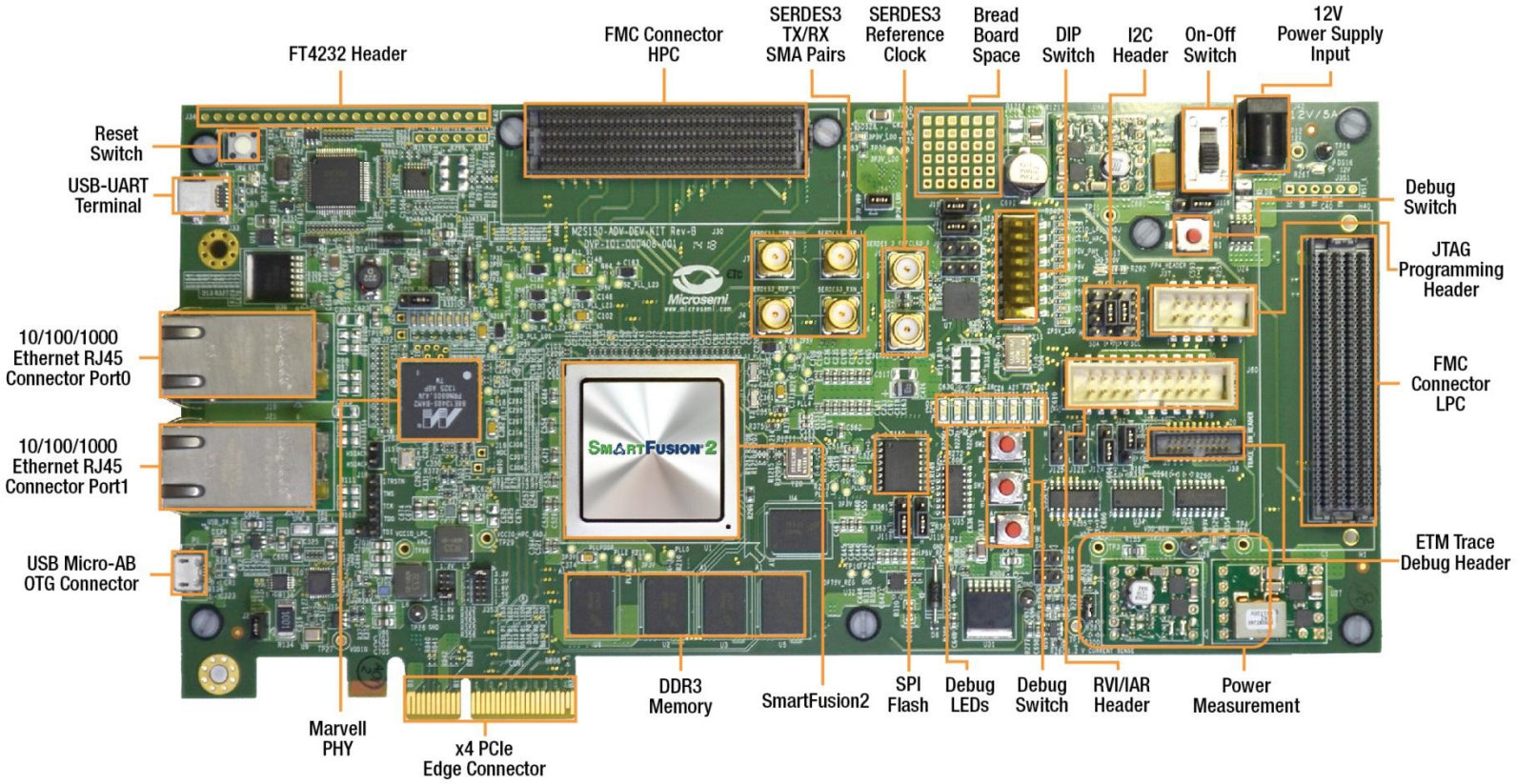


SmartFusion2 Advanced Development Kit

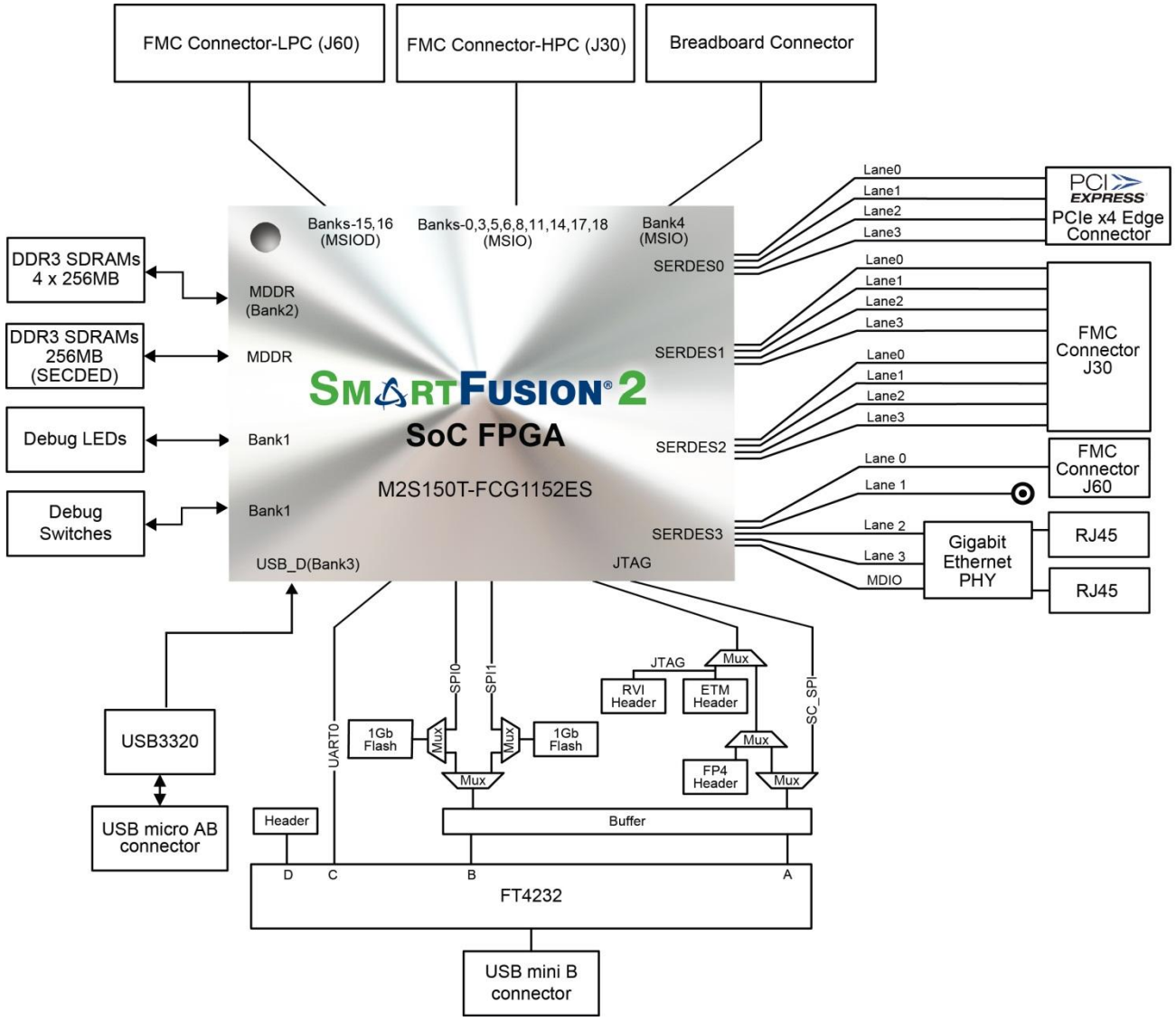
- Applications
  - Embedded ARM® Cortex™-M3 processor based systems
  - PCIe endpoint
  - Motor control
  - Industrial automation
  - Power measurement
  - Security
  - FMC expansion
  - High speed I/O
  - Universal serial bus (USB) applications (OTG support)
- Features
  - Dual Gigabit Ethernet, USB 2.0, SPI, I2Cs, UARTs
  - 2 Giga Bytes (GB) SPI flash - 1GB connected to MSS and other 1GB connected to FPGA fabric
  - Two FMC connector with HPC/LPC pinout for expansion
  - x4 PCIe edge connector
  - One pair SMA connector
  - Core current measurement test points

SmartFusion2 Advanced Development Kit

- Part number - M2S150-ADV-DEV-KIT-ES
- List Price – $999



SmartFusion2 Advanced Development Kit



Hands-on Lab

FPGA Fabric Lab

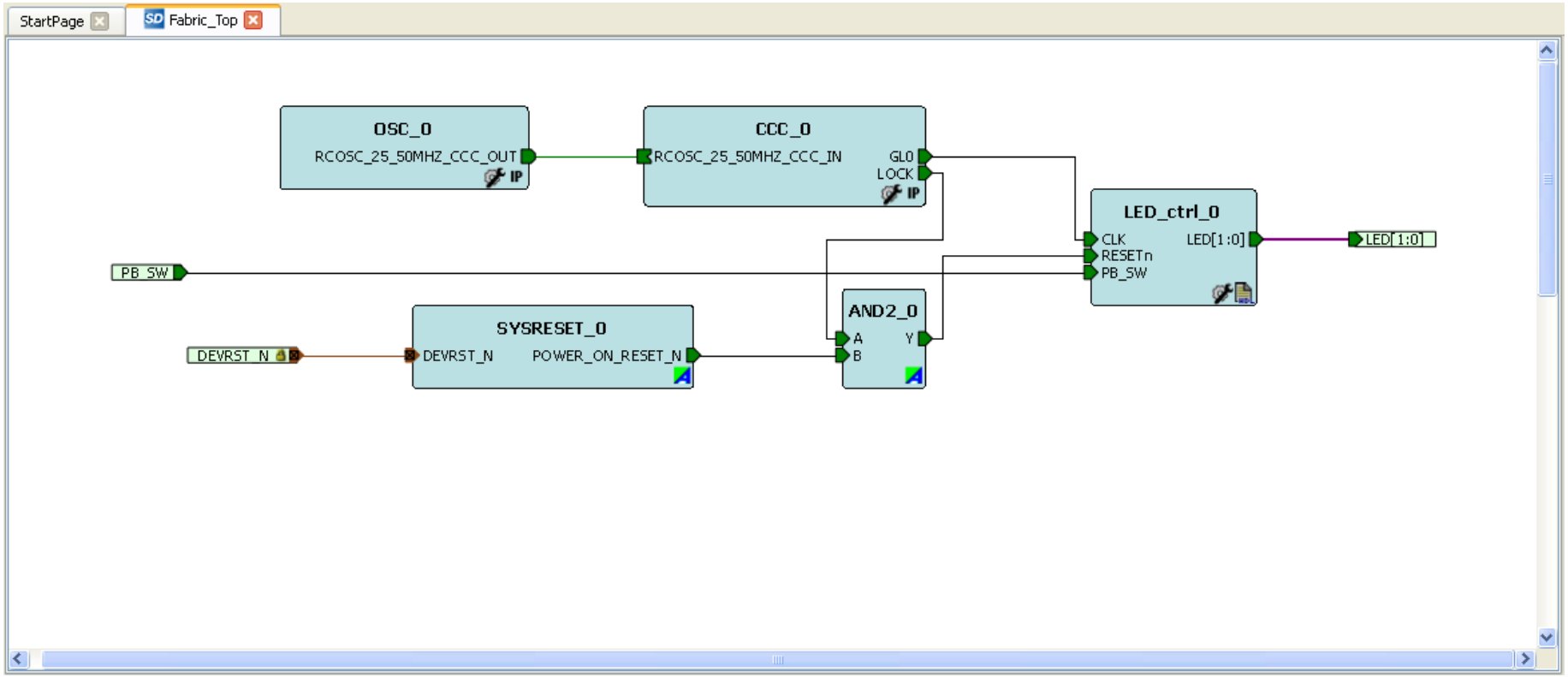
- This lab demonstrates how to implement a basic SmartFusion2 FPGA fabric design using SmartDesign.
  - The design drives the LEDs on SmartFusion2 Kits with different patterns based on the state of switch S2 as shown in the table below.

Reset Switch	User Switch	LED Behavior
Depressed	don't care	LEDs off
Released	Released	LEDs toggle
Released	Depressed	Both LEDs blinking

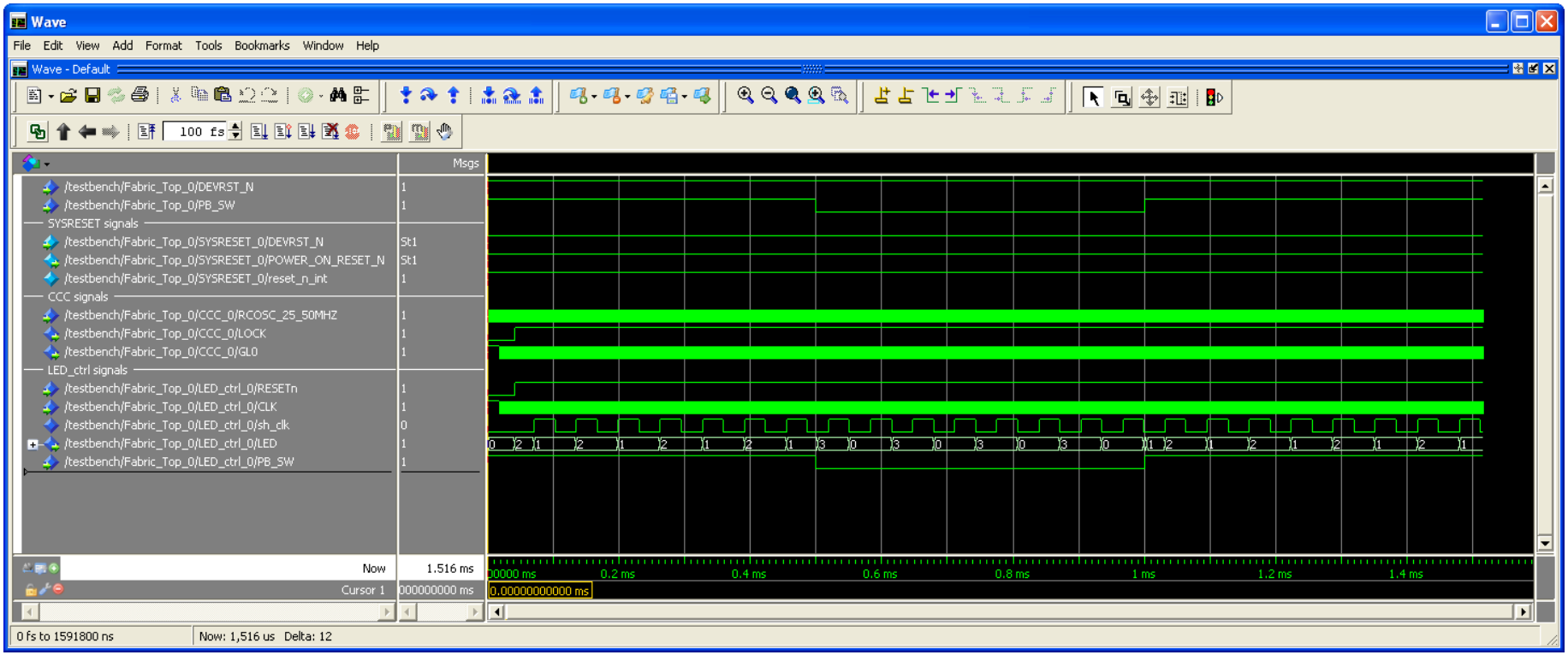
- After completing this lab you will be familiar with the following:
  - Creating a Libero SoC project
  - Implementing a SmartFusion2 fabric design with SmartDesign
  - Simulating the design, running layout and programming the SmartFusion2 silicon

- Components of SmartFusion2 Device Used
  - SmartFusion2 FPGA fabric, the on-chip 25/50 MHz RC oscillator and the Fabric CCC.

SmartDesign Canvas



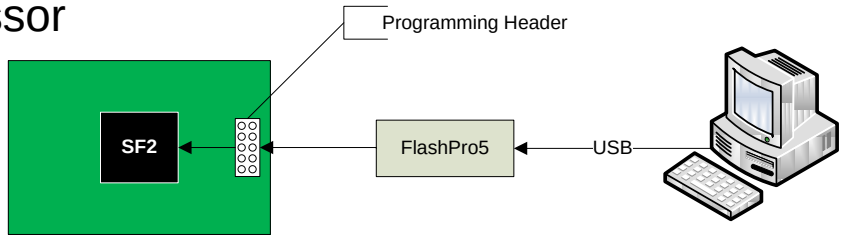
Simulation



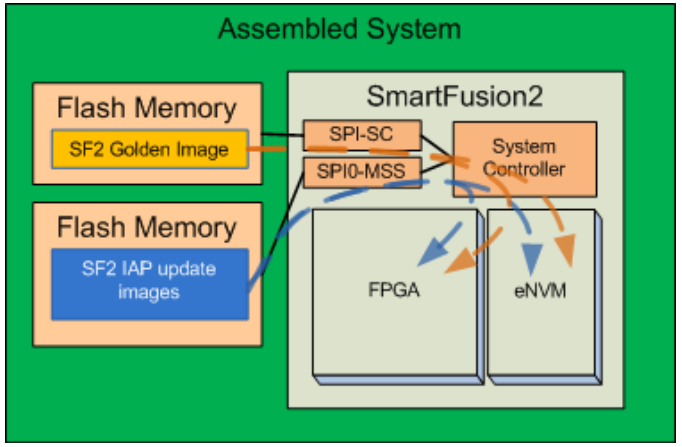
Programming

SF2 Programming Terminology

- In-System Programming (ISP)
  - **External intelligent** source acting as master
    - FlashPro, Libero or embedded processor
  - Programming interfaces
    - JTAG, SPI-Slave



- In-Application-Programming (IAP)
  - **Internal intelligent** source acting as master
    - User Application (in FPGA or Cortex®M3)
    - Programming data received from USB, Ethernet, etc.
      Programming Interface
    - SPI0-MSS, as SPI-Master



- Auto-programming with Golden Image
  - System Controller acting as master

Programming Support Matrix

Programming Mode	JTAG	SPI Slave	Auto Programming	Auto Update	2 Step IAP	Programming Recovery	M3 ISP
Programming Interface	JTAG	SPI_SC	SPI_0	SPI_0	SPI_0	SPI_0	-
M2S005(S)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
M2S010(T),(TS)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
M2S025(T),(TS)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
M2S050(T),(TS)*	Yes	Yes	YES/SC_SPI	N/A	No***/SC_SPI	N/A	Yes
M2S090(T),(TS)	Yes	Yes	Yes	Yes	Yes	Yes	Yes
M2S150(T),(TS)	Yes	Yes	Yes	Yes	Yes	Yes	Yes

*050 System Controller always uses SC SPI. No support for recovery/auto update

*** Next die rev (rev 2) will support IAP in 050

Programming Interfaces

- SPI_0
  - SPI peripheral 0 in the MSS (SF2) or HPMS (IGLOO2)
  - Use: Auto Programming, 2 Step IAP, Programming Recovery
  - I/O Type: MSIO (3.3V max), internal pull ups
- SC_SPI
  - System Controller dedicated SPI port
  - Use: SPI-Slave programming
  - I/O Type: MSIO (3.3V max), internal pull ups
- JTAG
  - System Controller dedicated port
  - Use: JTAG programming
  - I/O Type: dedicated l I/O – supports multiple voltage level (1.2/1.5/1.8/2.5/3.3V), internal pull ups

Other Dedicated Pins Used In Programming

- FLASH_GOLDEN_N
  - Dedicated input Pin
    - Assertion on power-up commands the System Controller to read the golden image bitstream from a SPI flash on SPI_0 port and program device with it
    - Active low pin
    - Under normal operating condition it must be connected to bank voltage with 10k pull up

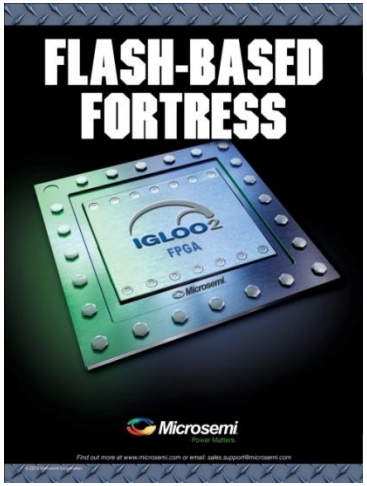
- DEVRST_N
  - Dedicated input pin with Schmitt Trigger
    - Device reset; active low and powered by VPP
    - In unused condition, pull up to VPP through 10 K resistor
    - Minimum pulse width 1us to put the device into reset

Programming Interface Availability

Package	Devices	Programming interfaces			Dedicated pins for programming	
		JTAG	SPI_0	System Controller SPI Port (SC-SPI)	Flash_GOLDEN_N	DEVRST_N
VQ144	M2S/M2GL005/S	Yes	Yes	No	No	Yes
	M2S/M2GL010/S	Yes	Yes	No	No	Yes
VF256	M2S/M2GL010 (T/TS)	Yes	Yes	No	No	Yes
	M2S/M2GL025 (T/TS)	Yes	Yes	No	No	Yes
FCS325	M2S/M2GL025 (T/TS)	Yes	Yes	No	No	Yes
	M2S/M2GL050 (T/TS)	Yes	Yes	No	No	Yes
	M2S/M2GL090 (T/TS)	Yes	Yes	No	No	Yes
VF400	M2S/M2GL005/S	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL010 (T/TS)	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL025 (T/TS)	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL050 (T/TS)	Yes	Yes	Yes	Yes	Yes
FCV484	M2S/M2GL150 (T/TS)	Yes	Yes	Yes	Yes	Yes
FG484	M2S/M2GL005/S	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL010 (T/TS)	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL025 (T/TS)	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL050 (T/TS)	Yes	Yes	Yes	Yes	Yes
	M2S/M2GL090 (T/TS)	Yes	Yes	Yes	Yes	Yes
FG676	M2S/M2GL090 (T/TS)	Yes	Yes	Yes	Yes	Yes
FG896	M2S/M2GL050 (T/TS)	Yes	Yes	Yes	Yes	Yes
FC1152	M2S/M2GL150 (T/TS)	Yes	Yes	Yes	Yes	Yes

Programming Bitstream

- Bitstreams can contain:
  - FPGA only content
  - eNVM only content or a portion of eNVM
  - All of the above



- Bitstreams are ALWAYS encrypted
  - That means eNVM as well
  - Secure encrypted firmware updates
    - This is one of the reasons why SF2 boots securely!
  - Additional security (user security) can be added from SPM (security policy manager)

- Bitstream Protocol is DPA resistant

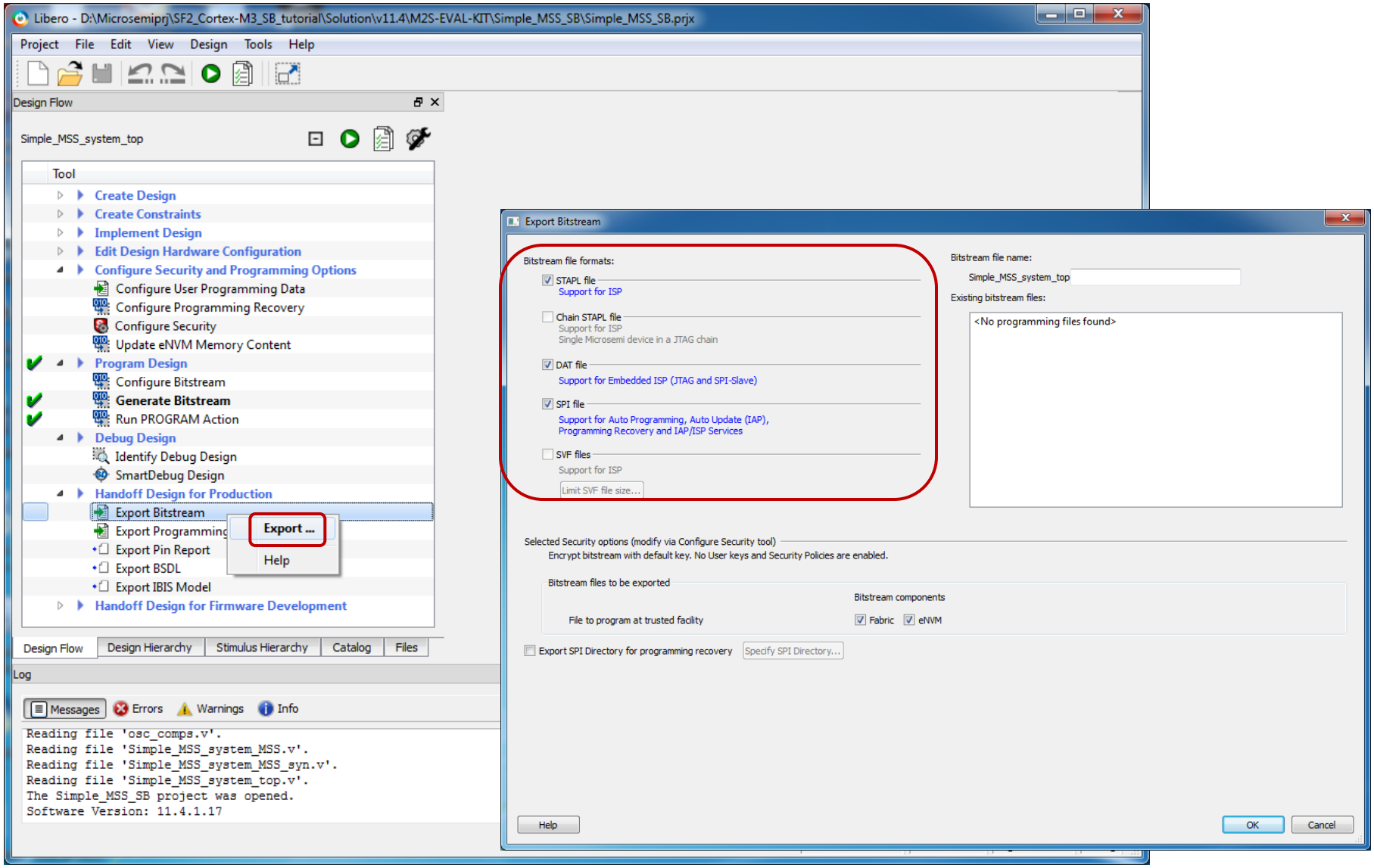
LICENSED
DPA
COUNTERMEASURES
™

Programming Bitstream (cont.)

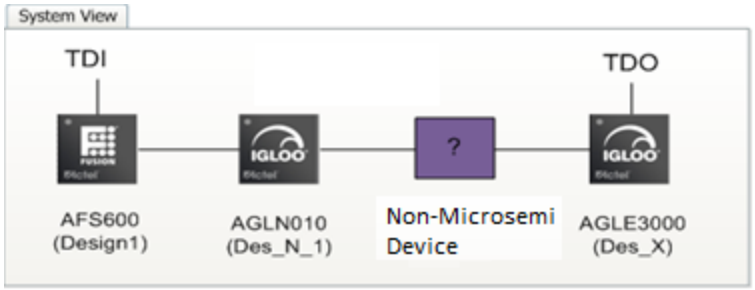
- Bitstreams format supported:
  - STAPL- JTAG Programming
  - DAT – JTAG, SPI-Slave
  - SPI – Auto Programming, MSS ISP, 2 step IAP
- Bitstreams size (KB)

Device	File Type	Fabric	eNVM Full	Fabric/eNVM Full	Fabric/eNVM Full/Security
M2S010 (T/S/TS)					
	STAPL	925	473	1347	2646
	SPI	556	270	823	N/A
	DAT	557	272	825	1648
M2S025 (T/S/TS)					
	STAPL	1937	473	2359	4670
	SPI	1195	270	1463	N/A
	DAT	1197	272	1464	2926
M2S050 (T/S/TS)					
	STAPL	3784	473	4206	8363
	SPI	2363	270	2630	N/A
	DAT	2364	272	2632	5261
M2S090 (T/S/TS)					
	STAPL	5680	889	6517	12985
	SPI	3561	534	4092	N/A
	DAT	3562	535	4093	8185
M2S150 (T/S/TS)					
	STAPL	9530	889	10367	20685
	SPI	5996	534	6527	N/A
	DAT	5997	535	6528	13054

Bitstream Export



Programming Methods – JTAG Programming



JTAG Programming (M2S and M2GL)

- Programming Interface:
  - System controller's dedicated JTAG Port
- Programming Master:
  - External Programmer such as FlashPro4
  - External Microprocessor running DirectC
  - External standalone programmer such as Silicon Sculptor 3
- Programming software:
  - Libero SoC (default programming flow)
  - FlashPro (stand alone programming)
- Bitstream format: .stp, .dat, .pdb (Libero default flow, not exportable)

JTAG Programming (cont.)

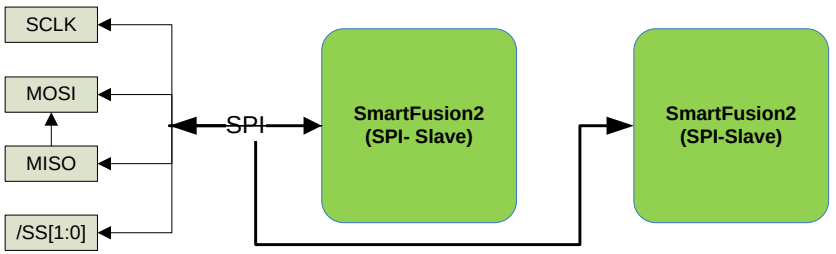
- Programming voltage:
  - Power up the bank containing JTAG pins
  - VPP*, VNVM

VPP/VPPNVM M2S005, M2S010, M2S025, M2S050	Power supply for charge pumps (for normal operation and programming)	2.5 V range	2.375	2.5	2.625	V	
		3.3 V range	3.15	3.3	3.45	V	
VPP/VPPNVM M2S090, M2S150	Power supply for charge pumps (for normal operation and programming)	3.3 V range	3.15	3.3	3.45	V	

  - Core voltage

*When VPP = 2.5 V is used, leave Vpump pin of FP4 JTAG header floating (no connect) and select "Drive VPUMP" option FP4 programmer settings.

Programming Methods – SPI-Slave



System Controller SPI Slave Programming
M2S and M2GL

- Programming Interface:
  - System controller's dedicated SPI Port (SC_SPI)
  - SC_SPI port default is slave mode
  - SPI mode: 3 (SPO/CPOL= SPH/CPHA =1)

Name	Type	Description
SC_SPI_SS	In	SPI Slave select
SC_SPI_SDO	Out	SPI data output
SC_SPI_SDI	In	SPI data input
SC_SPI_CLK	In	SPI clock

- Programming Master:
  - An external Microprocessor with at least 1200 bytes of RAM
  - External Programmer such as FP5 (ISP), SS3 (standalone)
  - FPGA – embedded microprocessor such as CM3 (SF2) or soft microprocessor such as CoreABC (M2GL) can be used
- Programming software:
  - End User Application such as SPI DirectC code
- Bitstream format: .dat

SPI-Slave (cont.)

- Programming voltage:
  - Power up the bank containing SC_SPI pins
  - VPP, VNVM

VPP/VPPNVM M2S005, M2S010, M2S025, M2S050	Power supply for charge pumps (for normal operation and programming)	2.5 V range	2.375	2.5	2.625	V	
		3.3 V range	3.15	3.3	3.45	V	
VPP/VPPNVM M2S090, M2S100, M2S150	Power supply for charge pumps (for normal operation and programming)	3.3 V range	3.15	3.3	3.45	V	

  - Core voltage

- Other requirement:
  - Access to the data file containing the programming data
  - Memory to store and run SPI DirectC code
  - Example:

Code Memory Requirements- SPI DirectC Code Size on M3 16-Bit Mode

Text in Bytes	Data in Bytes	BSS in Bytes
4680	1169	60

Text - This is the compiled code size memory requirements.

Data - This is the run time memory requirement, i.e. the free data memory space required to execute the code.

BSS - This is the Block Started by Symbol allocation for variables that do not yet have values, i.e. uninitialized data. It is part of the overall Data size.

SPI-Slave (cont.)

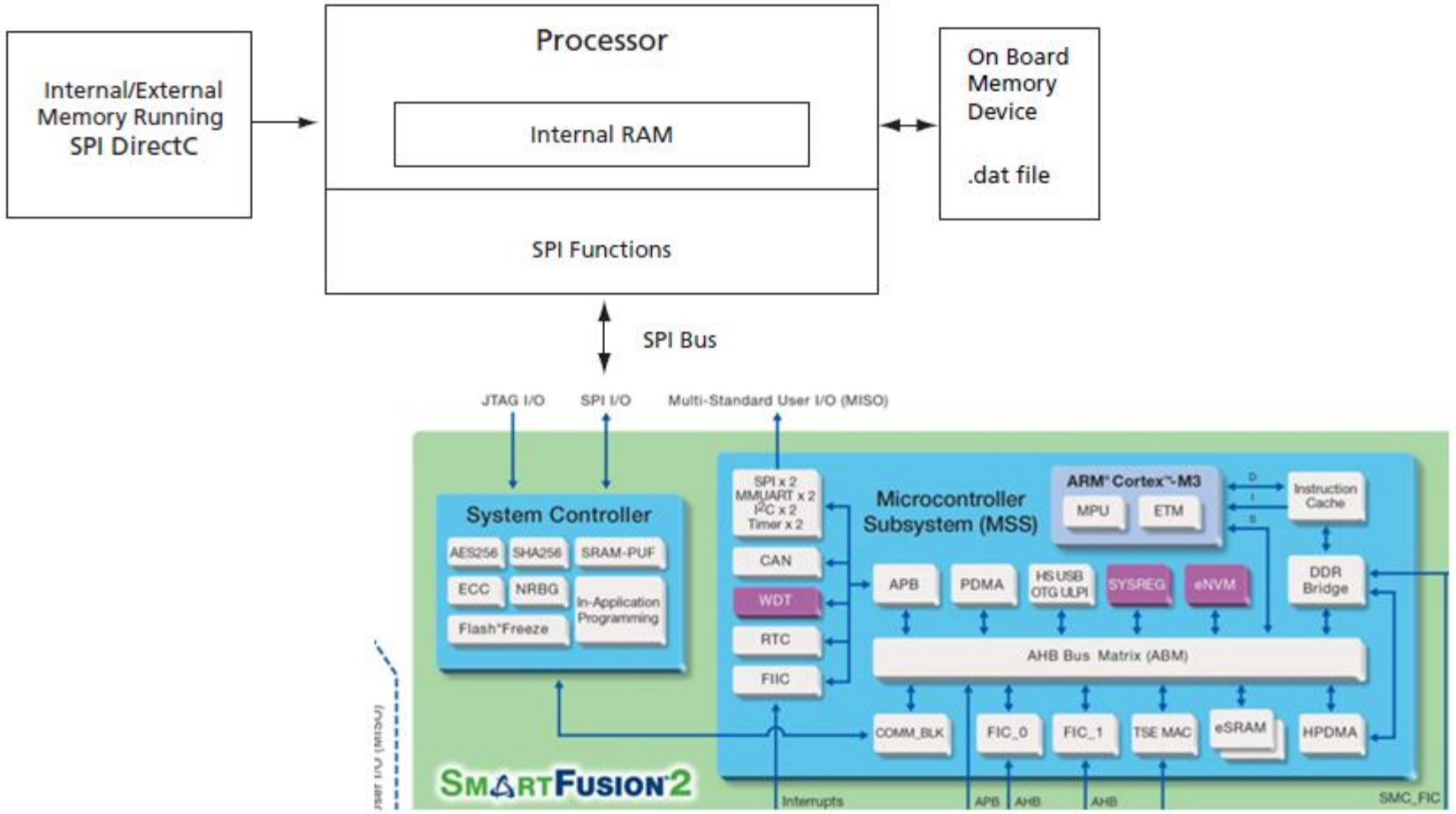
- Other Requirements
  - Data Storage Memory Requirements (DAT file size) -See slide # 136
  - System Controller supports any serial flash supporting the standard 0B read command
  - System controller expects bitstream in address zero of SPI Flash
  - Refer to System Controller SPI Characteristics section in the data sheet for details on SPI clock frequency and timing info

SPI-Slave Use Models

- Two use models:
  - Direct Access to Memory
  - Indirect Access to Memory
    - By paging support data can be accessed remotely (i.e. no direct access to entire memory)
    - Need modifications to some communication functions (See SPI-DirectC UG for details)

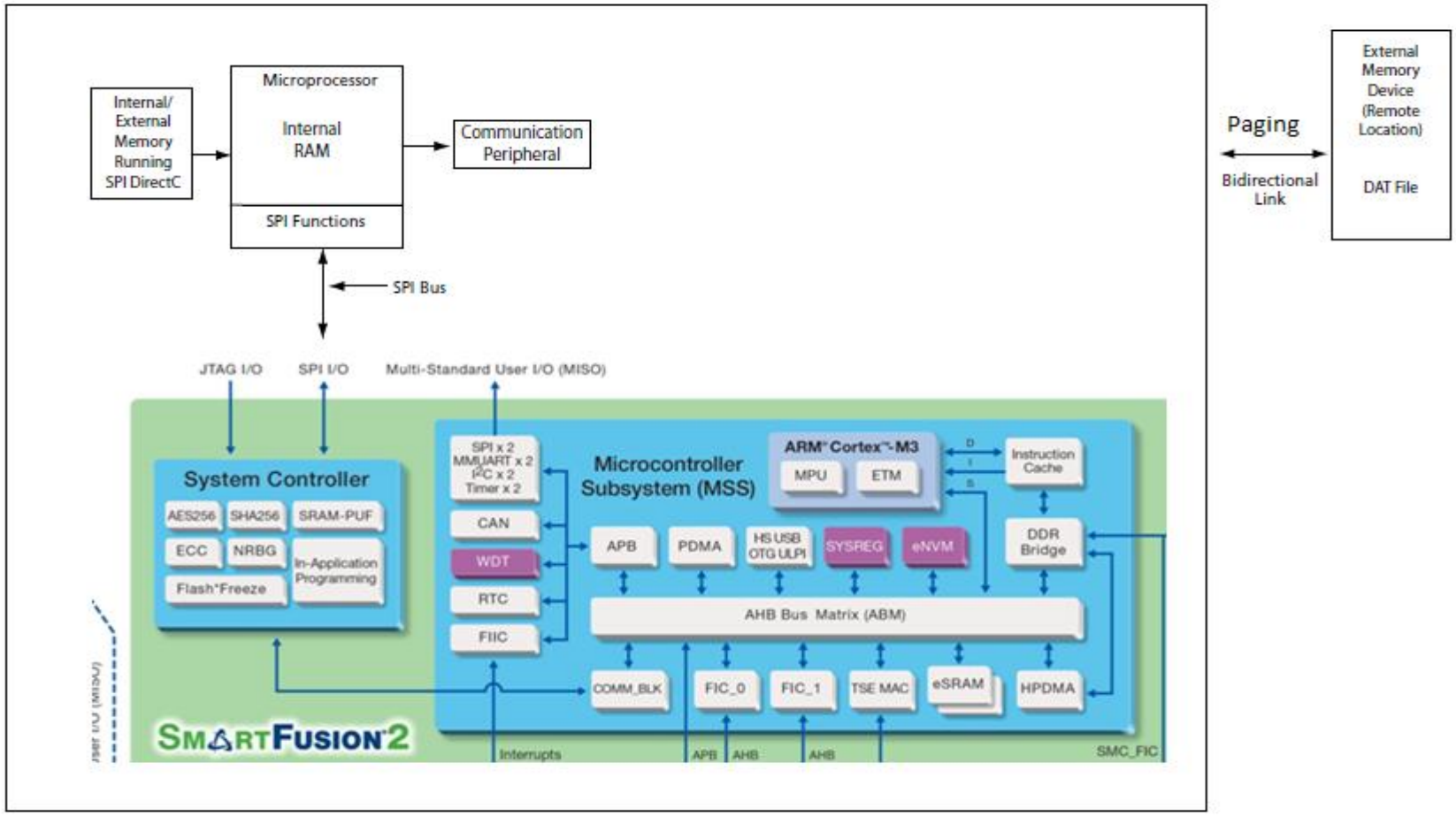
SPI-Slave Use Models
Direct Access to Memory

- Direct Access to Memory



SPI-Slave Use Models
Indirect Access to Memory

- Indirect Access to Memory

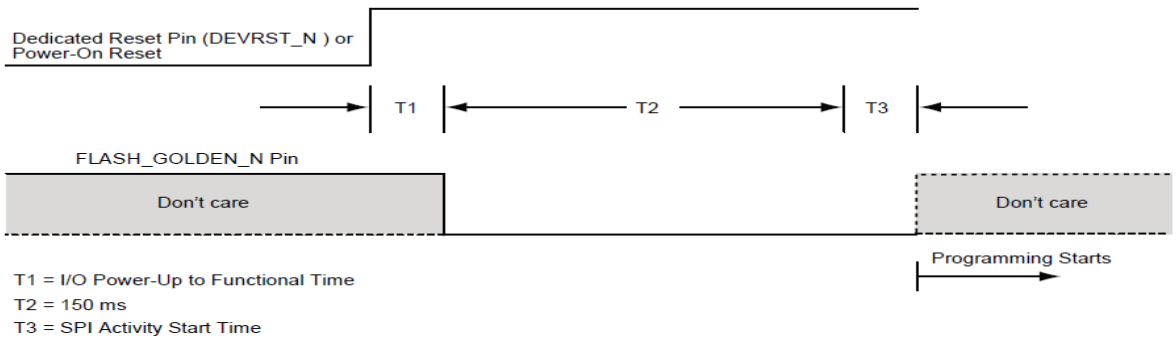


Auto Programming (M2S and M2GL)

- Programming Interface:
  - MSS SPI_0
  - SPI Master mode
    - Mode 3 as default in Blank device
    - Other modes can be set also during first time programming

Name	Type	Description
SPI_0_SS	Out	SPI Slave select
SPI_0_SDO	In	SPI data output
SPI_0_SDI	Out	SPI data input
SPI_0_CLK	Out	SPI clock

- Programming Master:
  - System Controller (SPI-Master)
- Programming software:
  - End User Application which triggers Auto programming
    - FLASH_GOLDEN_N needs to be pulled low during power-up or DEVRST_N assertion



Auto Programming (cont.)

- Bitstream format: .spi
- Programming voltage:
  - Power up the bank containing MSS SPI_0 pins
  - VPP, VNVM

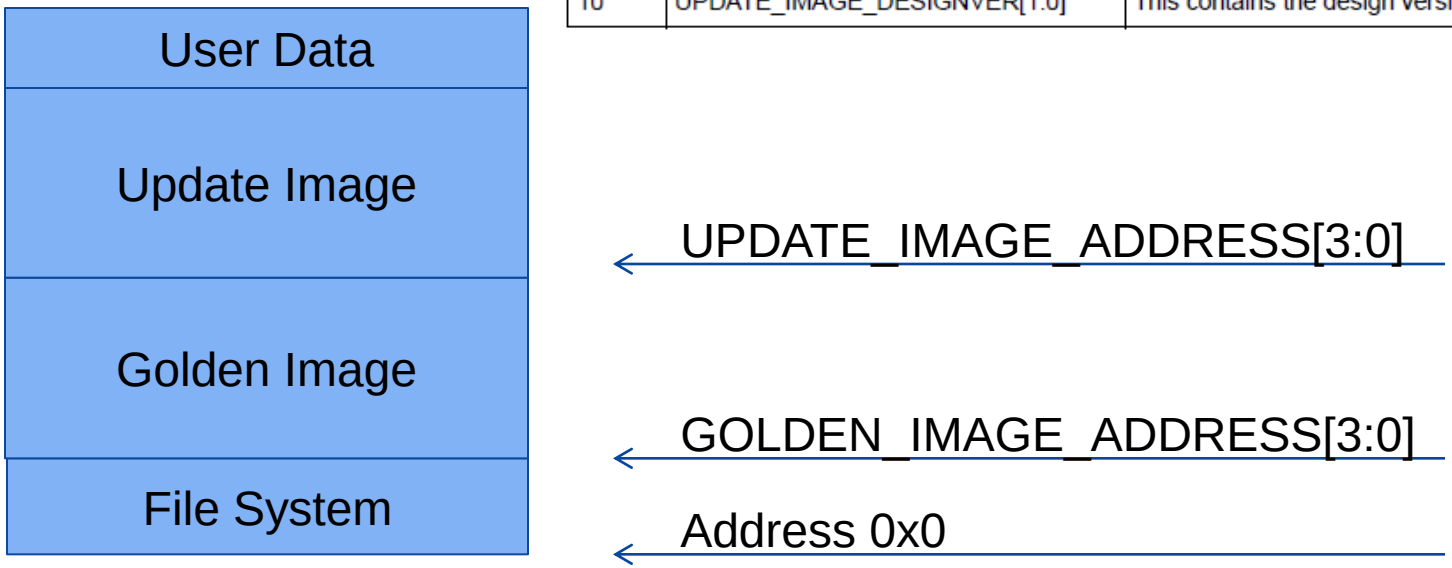
VPP/VPPNVM M2S005, M2S010, M2S025, M2S050	Power supply for charge pumps (for normal operation and programming)	2.5 V range	2.375	2.5	2.625	V	
		3.3 V range	3.15	3.3	3.45	V	
VPP/VPPNVM M2S090, M2S100, M2S150	Power supply for charge pumps (for normal operation and programming)	3.3 V range	3.15	3.3	3.45	V	

  - Core voltage
- Other Requirements
  - SPI flash from the following manufacturers are supported
    - Micron, Spansion, Atmel, Winbond
    - 3 and 4 byte SPI addressing supported (serial flash devices supporting the RDID (9F), 24-bit read (0B) and 32-bit read (0C). If RDID indicates a size greater than 2^{24} the firmware switches to using the 0C command (otherwise it uses 0B)
  - System Controller reads the SPI flash Device ID and Device Density from the SPI flash to determine the read algorithm
  - Refer to data sheet for SPI_0 timing

Auto Programming (cont.)

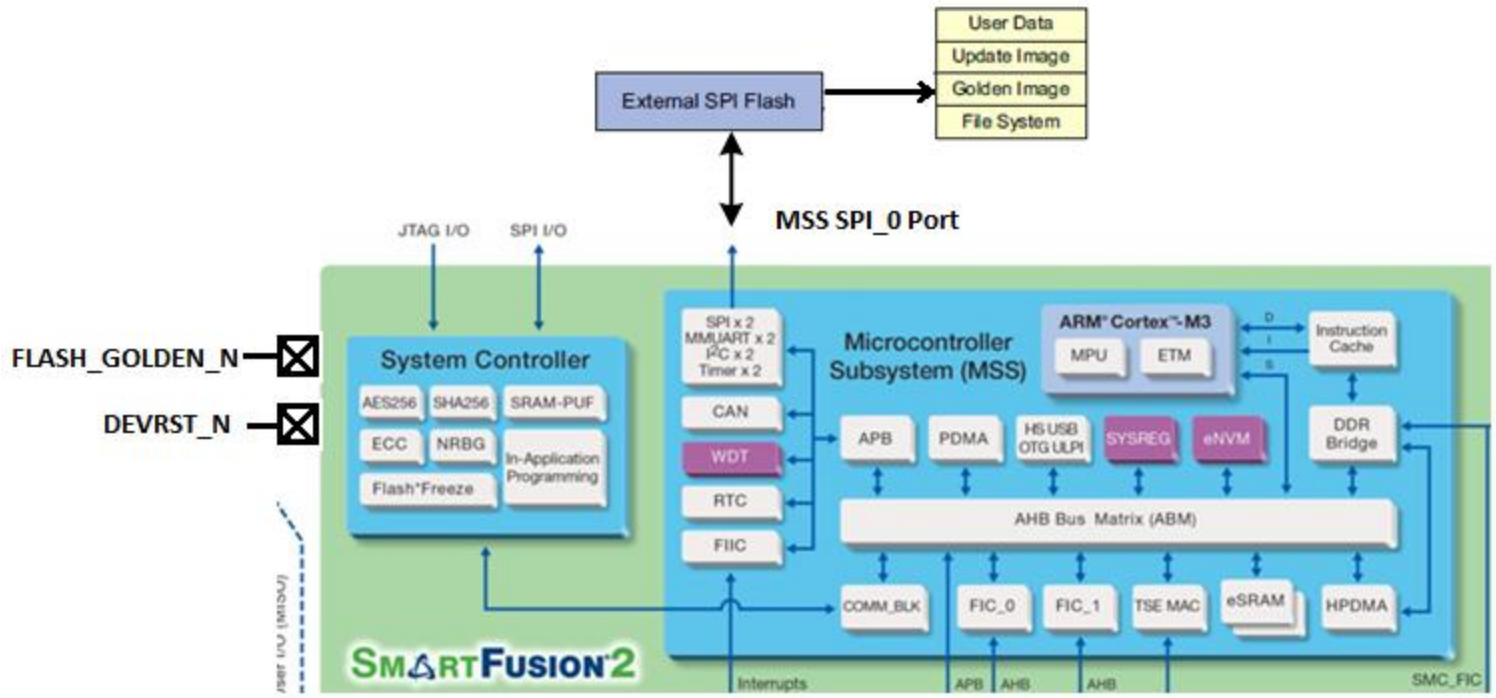
- Other Requirements (contd.)
  - User needs to authenticate the bitstream, otherwise the FPGA might be partially programmed (inoperable)
  - SPI flash contents & mapping – SPI_0 only

Offset	Name	Description
0	GOLDEN_IMAGE_ADDRESS[3:0]	This contains the address where the golden image starts.
4	GOLDEN_IMAGE_DESIGNVER[1:0]	This contains the design version of the golden image.
6	UPDATE_IMAGE_ADDRESS[3:0]	This contains the address where the update image starts.
10	UPDATE_IMAGE_DESIGNVER[1:0]	This contains the design version of the update image.



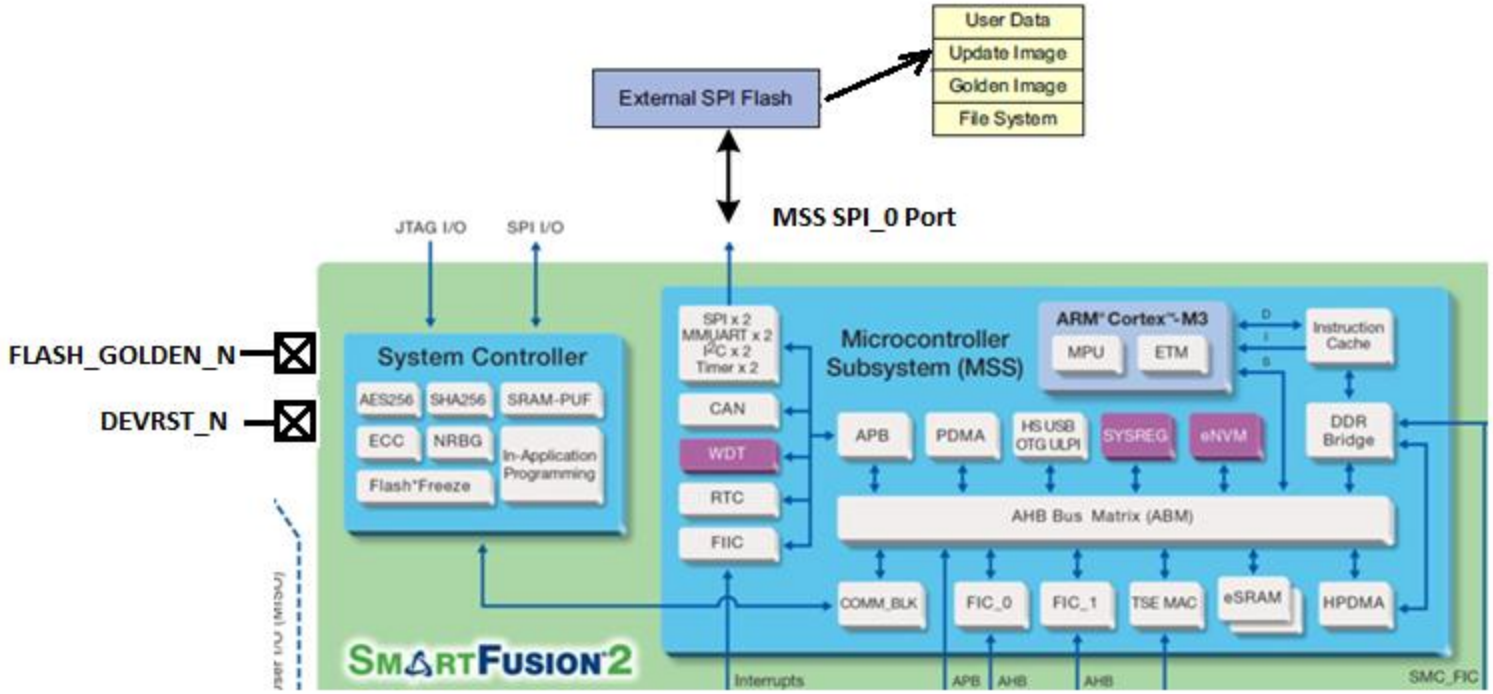
Auto Programming Use Models

- Blank device:
  - Manual trigger (or external logic) of FLASH_GOLDEN_N upon power up
  - System Controllers programs Golden Image from external SPI flash
    - Golden image needs to be programmed at address OFFSET '0'



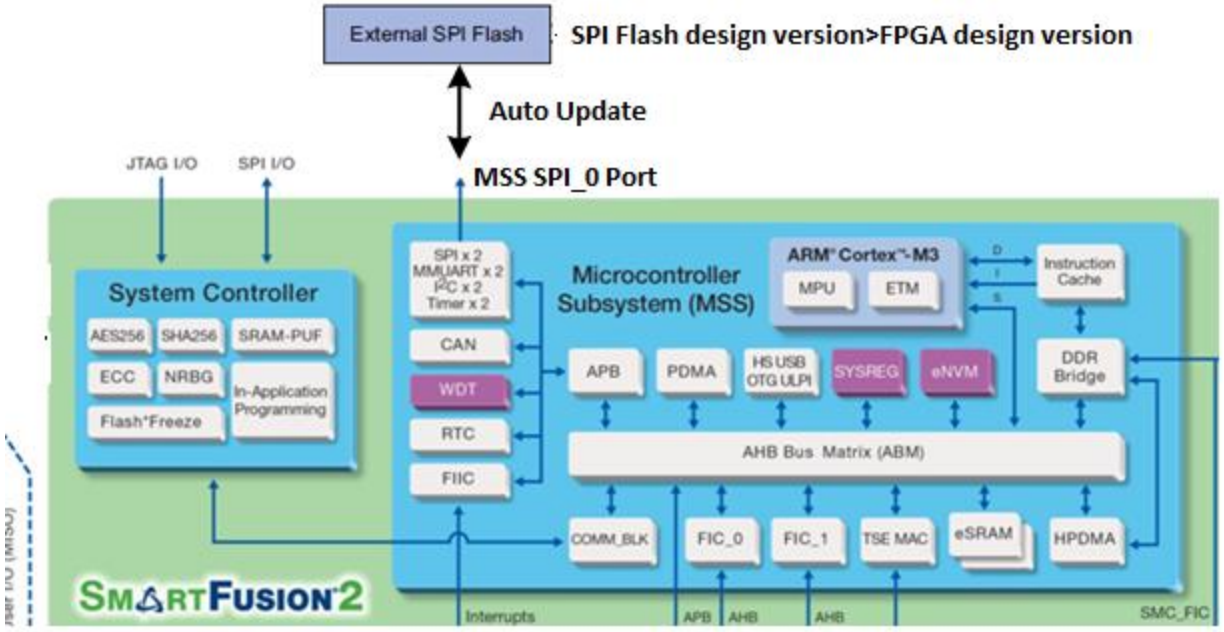
Auto Programming Use Models (cont.)

- Programmed device:
  - User logic triggers FLASH_GOLDEN_N upon power up or DEVRST_N
    - System Controllers programs Update Image from external SPI flash
      - Designer version of Update Image needs to greater than that of Golden Image



Auto Programming Use Models (cont.)

- Auto Update:
  - Used where Flash Golden pin is not bonded out
  - Factory programs AUTO UPDATE bits during manufacturing flow
  - During power-up System Controllers programs known good Image from external SPI flash
    - Design version of SPI Flash Image needs to greater than that of programmed FPGA



MSS ISP Service (SmartFusion2 Only)

- Programming Interface:
  - MSS Peripherals: MMUART, SPI, I2C, USB
- Programming Master:
  - Embedded Cortex-M3 (CM3)
    - Application code needs to be pre-programmed in the internal memories (i.e. eNVM, DDR/MDDR
- Programming software:
  - System Controller's ISP service : It has three modes: authentication, programming, and verification
- Bitstream format: .spi

MSS ISP Service (cont.)

- Programming voltage:
  - Power up the bank containing programming interface pins
  - VPP, VNVM

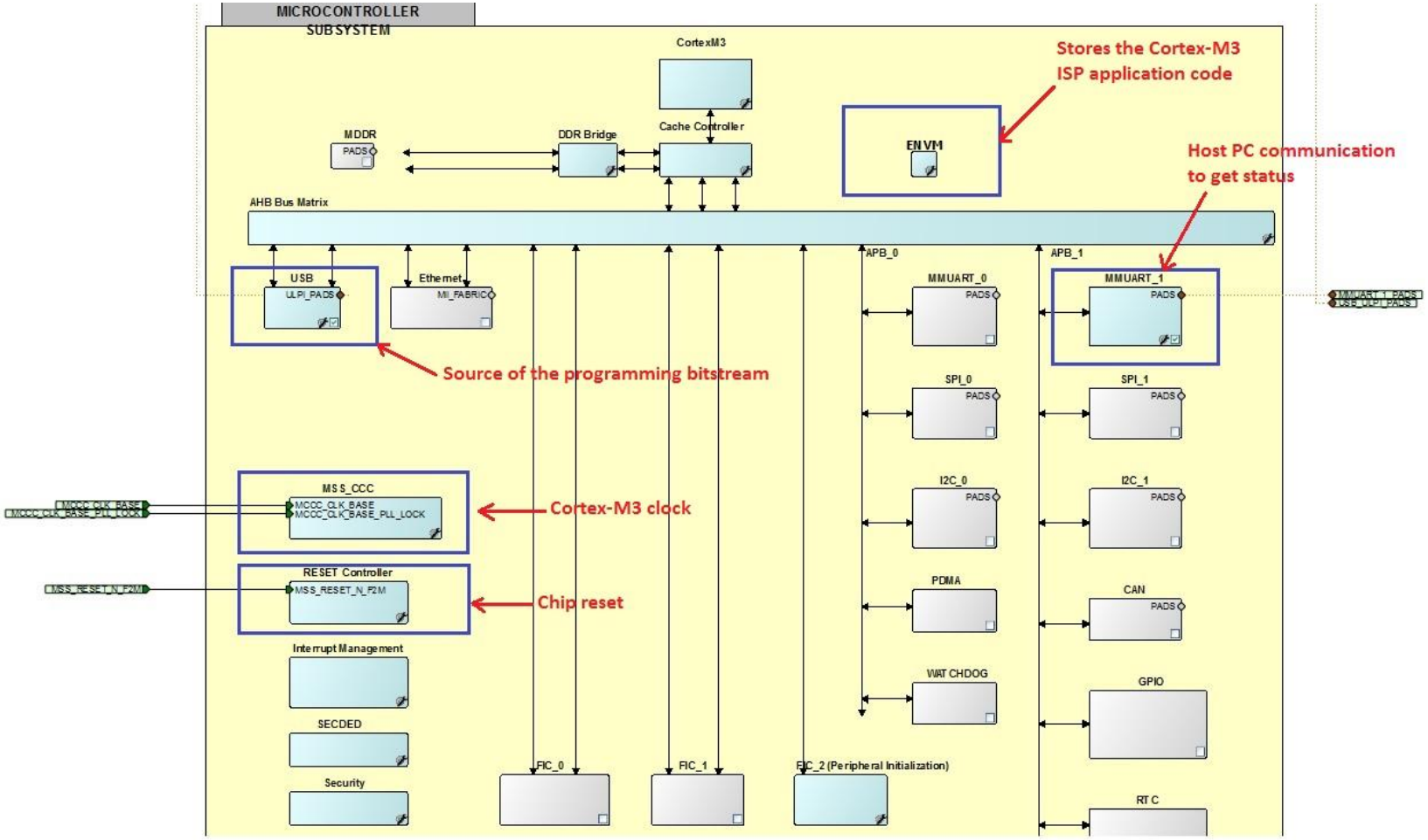
VPP/VPPNVM M2S005, M2S010, M2S025, M2S050	Power supply for charge pumps (for normal operation and programming)	2.5 V range	2.375	2.5	2.625	V	
		3.3 V range	3.15	3.3	3.45	V	
VPP/VPPNVM M2S090, M2S100, M2S150	Power supply for charge pumps (for normal operation and programming)	3.3 V range	3.15	3.3	3.45	V	

  - Core voltage

- Pre-programming M2S for MSS ISP:
  - Configure source of application image: eNVM, eSRAM, or DDR/MDDR
  - Configure source of programming bitstream: MSS Peripherals
  - Configure MSS clock source
  - Configure the firmware

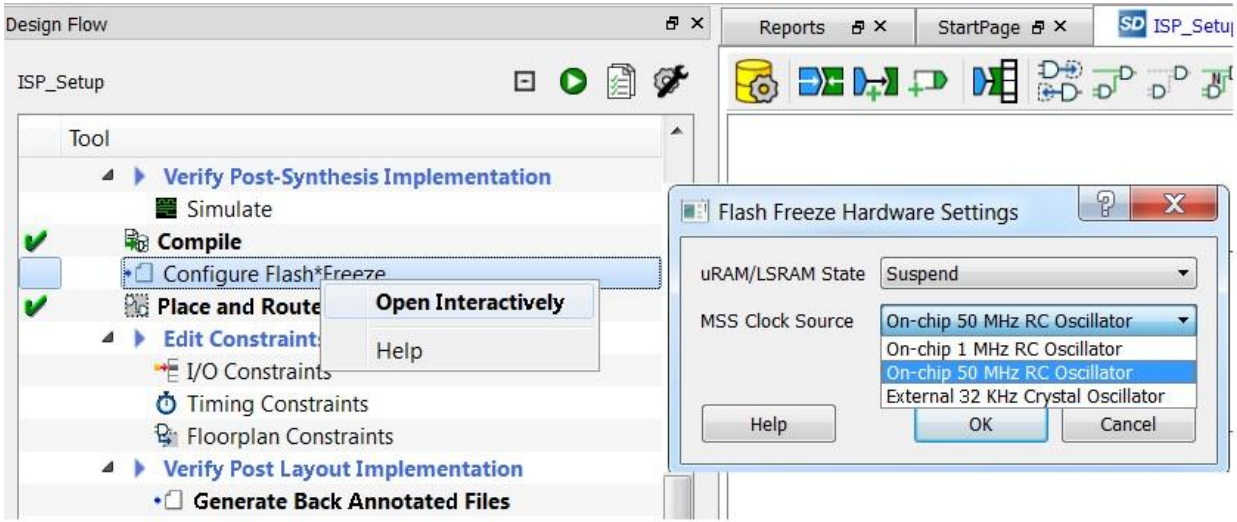
MSS ISP Service (cont.)

- Configuring MSS and peripherals

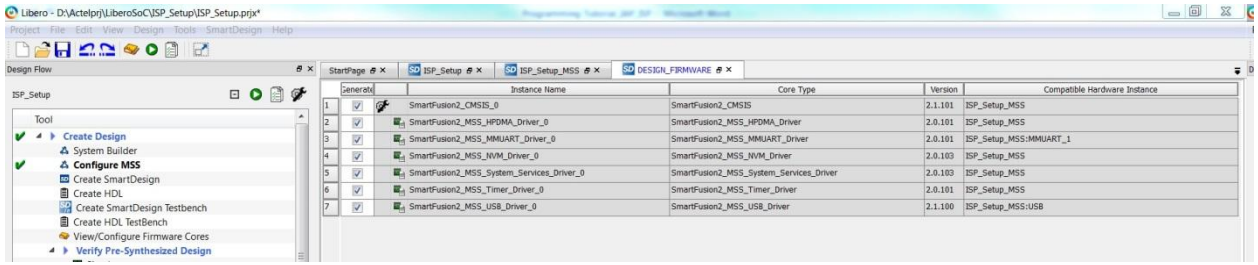


MSS ISP Service (cont.)

- Configuring MSS clock



- Configuring firmware

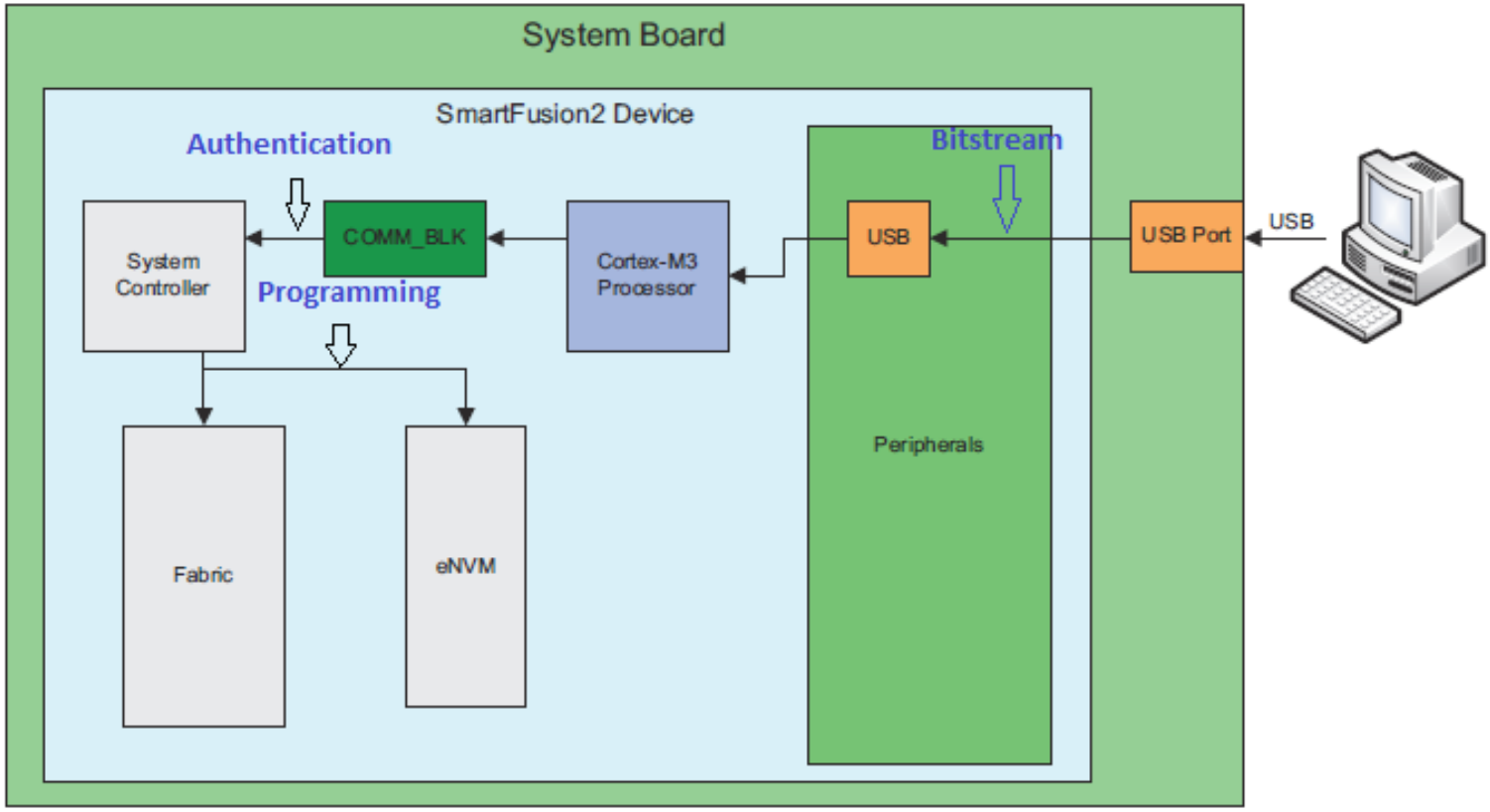


MSS ISP Service (cont.)

- Other Requirements
  - CM3 must be operational during the entire process of programming
  - If power failure is expected refer to Programming Recovery slides
  - Configuration can be pre-programmed to avoid putting the device into nonoperational state
  - If eNVM is source of ISP application code while the eNVM is being programmed the Cortex-M3 processor must execute the ISP application code from eSRAM. In order to execute the application code from eSRAM, the Cortex-M3 processor copies the ISP application code (stored in eNVM data client) to the eSRAM and remaps the eSRAM to the Cortex-M3 processor code region. See AC390 for details on remapping eSRAM
  - For Verify and Authenticate operation modes, the application code can be executed from either eNVM or eSRAM since the eNVM programming is not taking place

MSS ISP Service Use Model
(SmartFusion2 Only)

- Programming through USB



Programming Methods – 2 Step IAP

2 Step IAP Service
SmartFusion2 / IGLOO2

- Programming Interface:
  - MSS peripherals : MMUART, SPI, I2C, USB
  - MSS/HPMS SPI_0 port
- Programming Master:
  - User Application
    - It can be operated by embedded CM3 (M2S) or from FGPA fabric (M2GL)
- Programming software:
  - System Controller's IAP service : It has three modes: authentication, programming, and verification
- Bitstream format: .spi

2 Step IAP Service (cont.)

- Programming voltage:
  - Power up the bank containing programming interface pins
  - VPP, VNVM

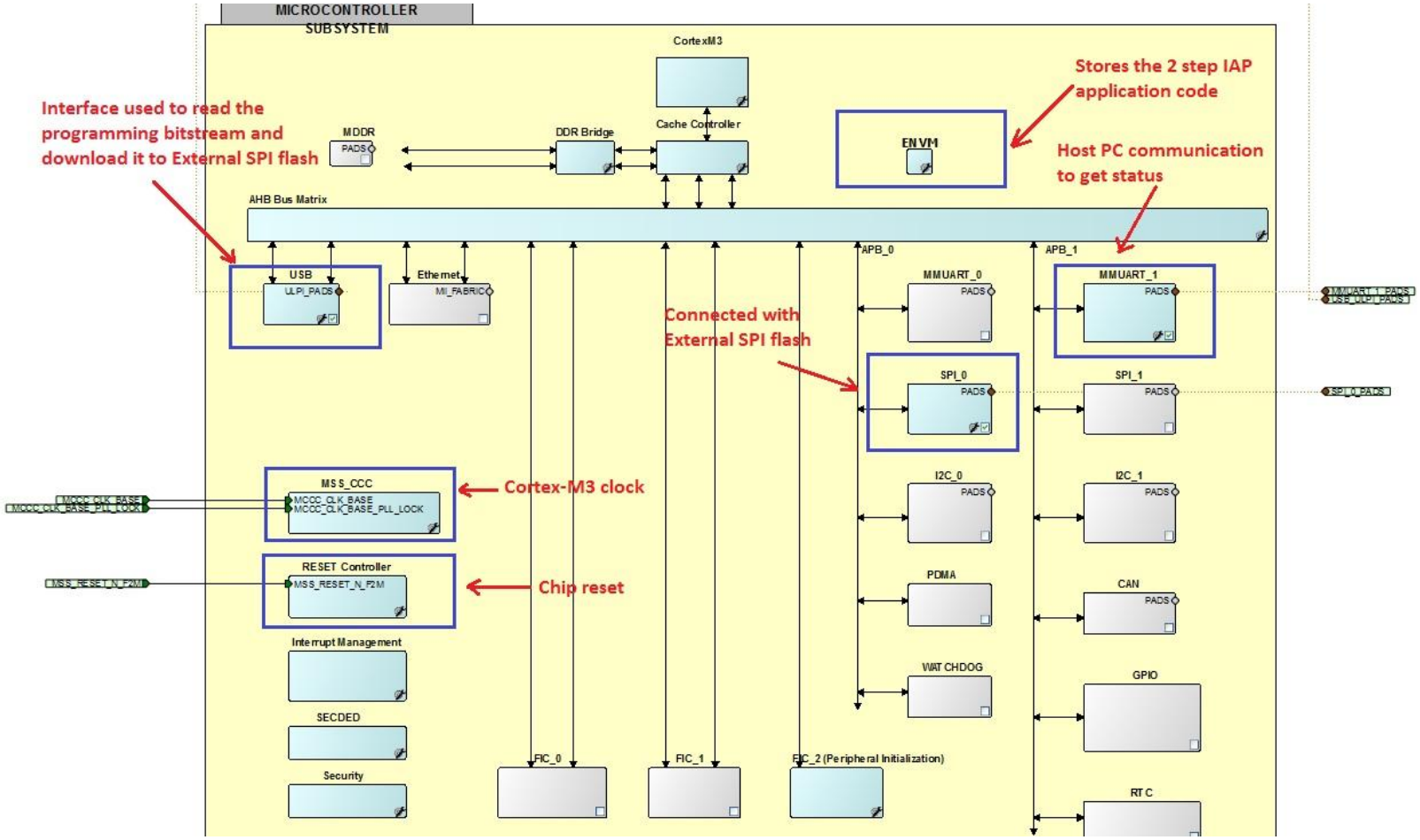
VPP/VPPNVM M2S005, M2S010, M2S025, M2S050	Power supply for charge pumps (for normal operation and programming)	2.5 V range	2.375	2.5	2.625	V	
		3.3 V range	3.15	3.3	3.45	V	
VPP/VPPNVM M2S090, M2S100, M2S150	Power supply for charge pumps (for normal operation and programming)	3.3 V range	3.15	3.3	3.45	V	

  - Core voltage

- Pre-programming the FPGA for 2 Step IAP:
  - Configure source of application image: eNVM, eSRAM, DDR/MDDR or FPGA fabric (M2GL)
  - Configure source of programming bitstream: MSS Peripherals including SPI_0 port
  - Configure MSS/HPMS clock source
  - Configure the firmware

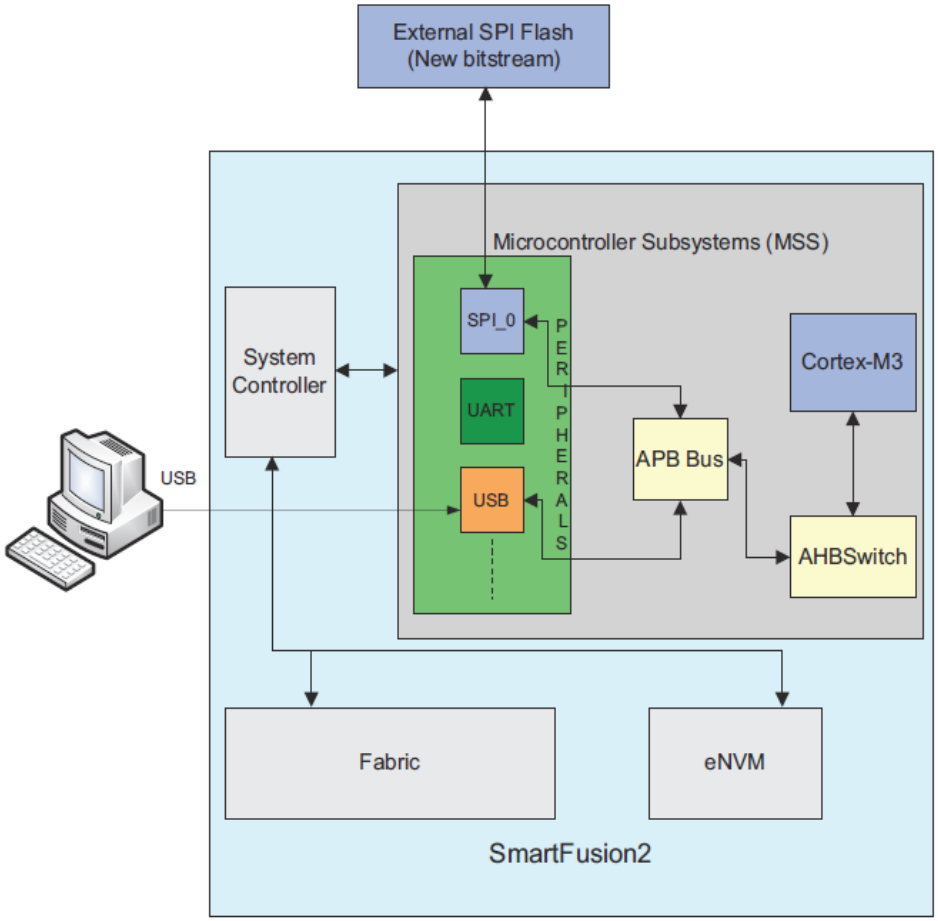
2 Step IAP Service (cont.)

- Configuring the FPGA for 2 step IAP



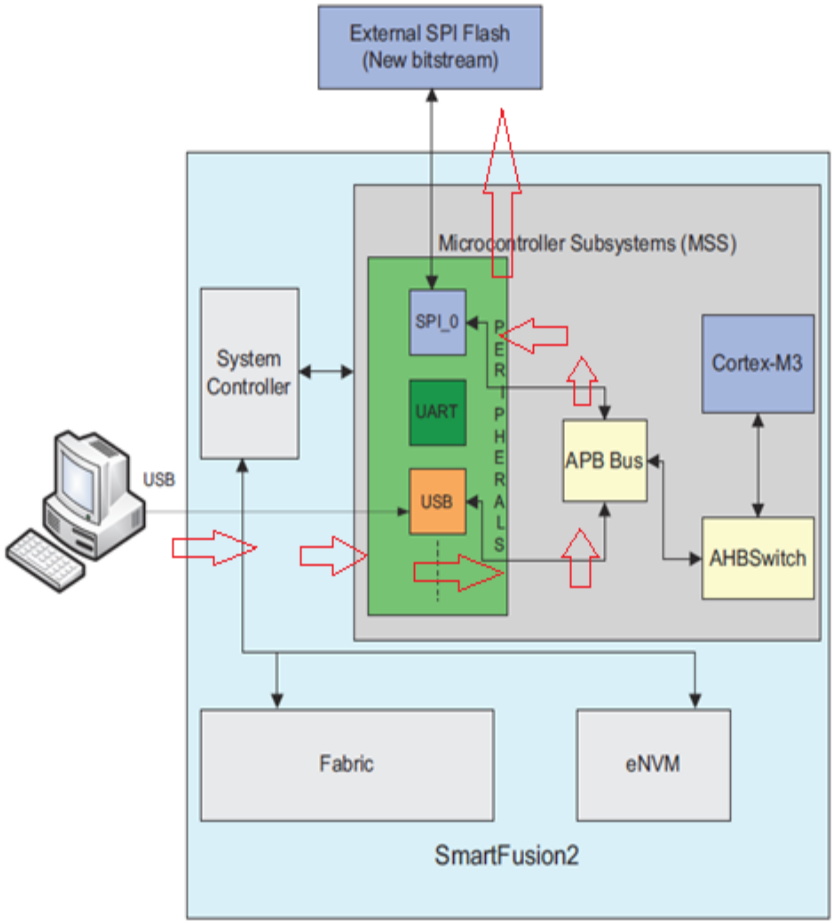
2 Step IAP Service Use Model (SmartFusion2)

- Always a 2 Step process
  - Step 1
    - User programs bitstream into external SPI flash connected to SPI_0
    - User should authenticate the bitstream themselves and have the System Controller authenticate the bitstream before proceeding to step 2
  - Step 2
    - User calls IAP System Service
    - System Controller will program the device based on the contents of the bitstream that the service call points to



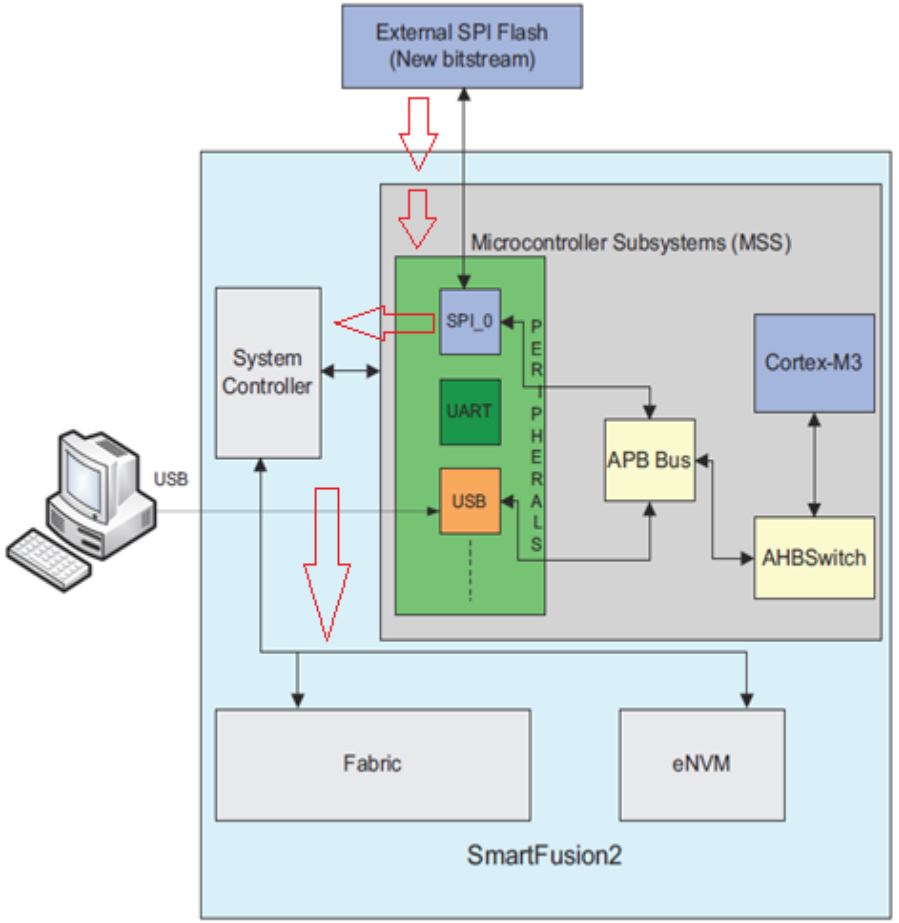
2 Step IAP: Step 1

- Program external SPI Flash
  - Device is fully operational during this stage
- Verify SPI flash programmed properly
  - Highly Recommend use the system controller to authenticate the bitstream



2 Step IAP: Step 2

- User calls IAP System Service to program the FPGA
  - User must allow System Controller exclusive access to SPI_0



I/O During Programming

- ISP
  - JTAG – I/O controlled by Boundary Scan Register (BSR)
  - SPI-Slave
    - Tri-state (if device is blank)
    - Bus hold, using user F*F configuration (if device is pre-programmed with the configuration)
- 2 step IAP or Cortex®M3 ISP
  - I/O depends on user option
    - Tri-state, or
    - Bus hold, using user F*F configuration (if device is pre-programmed with the configuration
- Auto-Programming
  - Tri-state

Peripherals During Programming

- JTAG & SPI-Slave (ISP)
  - MSS are held in reset, and the peripheral reverts to its hardware default
  - Cortex®M3 are held in reset
- 2 Step IAP
  - The peripheral initialized data will persist - until the new configurations are loaded into the peripheral registers when the Fabric powers up
- Cortex®M3 ISP
  - Peripherals are active during programming

Programming Recovery

Programming Recovery

- What it does:
  - If there is a power failure during programming update when the power is back on (power-up) the FPGA automatically begins programming and restore it with desired image (i.e. programs the FPGA with Golden Bitstream)

- What it does not:
  - Programming does not continue if there is –
    - SPI FLASH error
    - Authentication error of programming bitstream
    - Loss of communications link during ISP
  - ➢ The system controller does not time out, but keeps waiting for more data forever. The user needs to allow clean re-establishment of the communications link or correct the error (power cycle the FPGA)

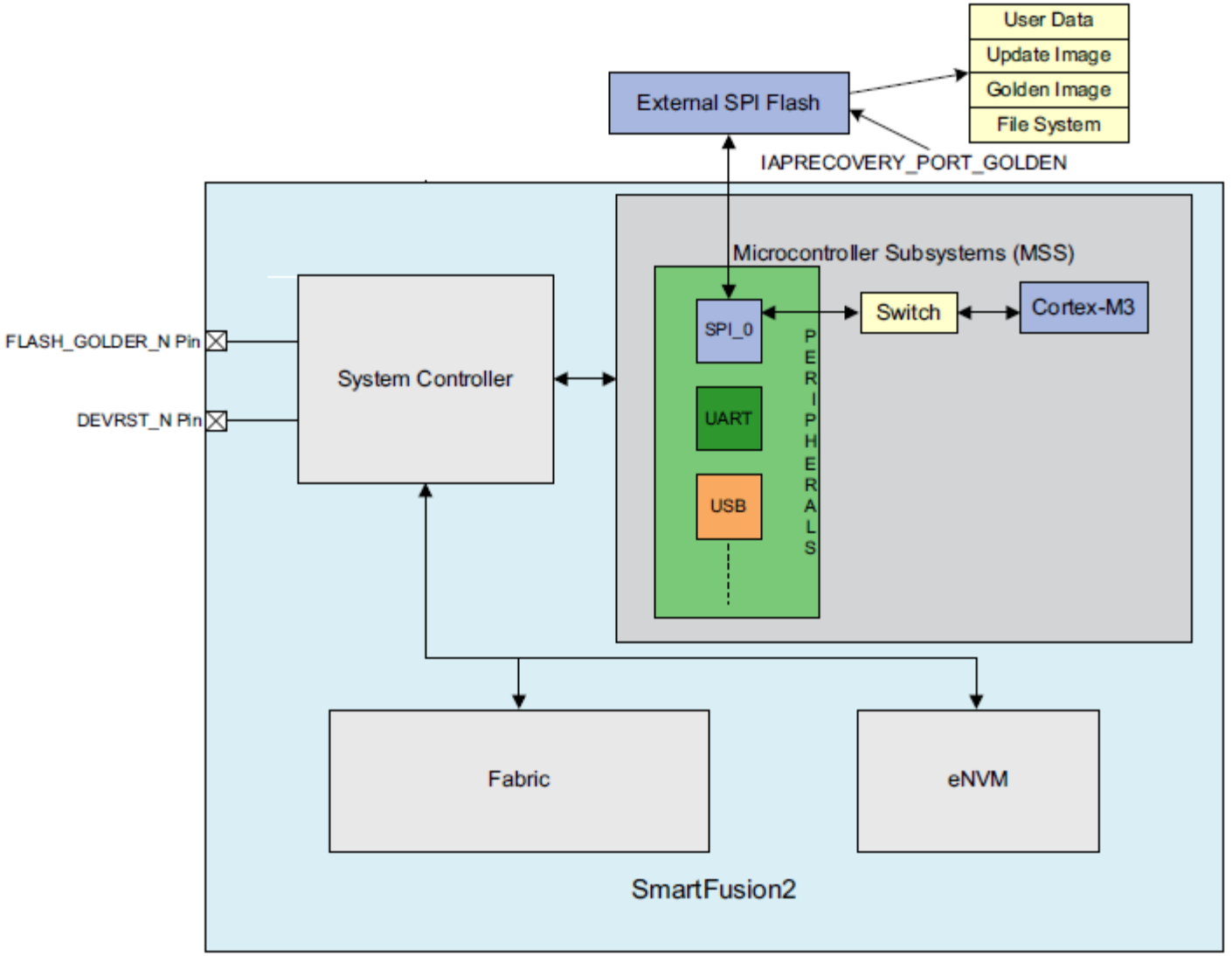
Programming Recovery (cont.)

- Programming Recovery setting (defined in Libero SoC) has to be programmed in to recovery to work during power failure
- For recovery MSS/HPMS SPI_0 port is used
- If power failed recovery takes place during power up where neither FPGA fabric nor Cortex-M3 is active
- Once recovery is successful device is enabled automatically
- If recovery fails, the device needs to be powered up again
- Golden image can’t be corrupted!
  - Recovery will fail and the device will be left in idle state

Programming Recovery (cont.)

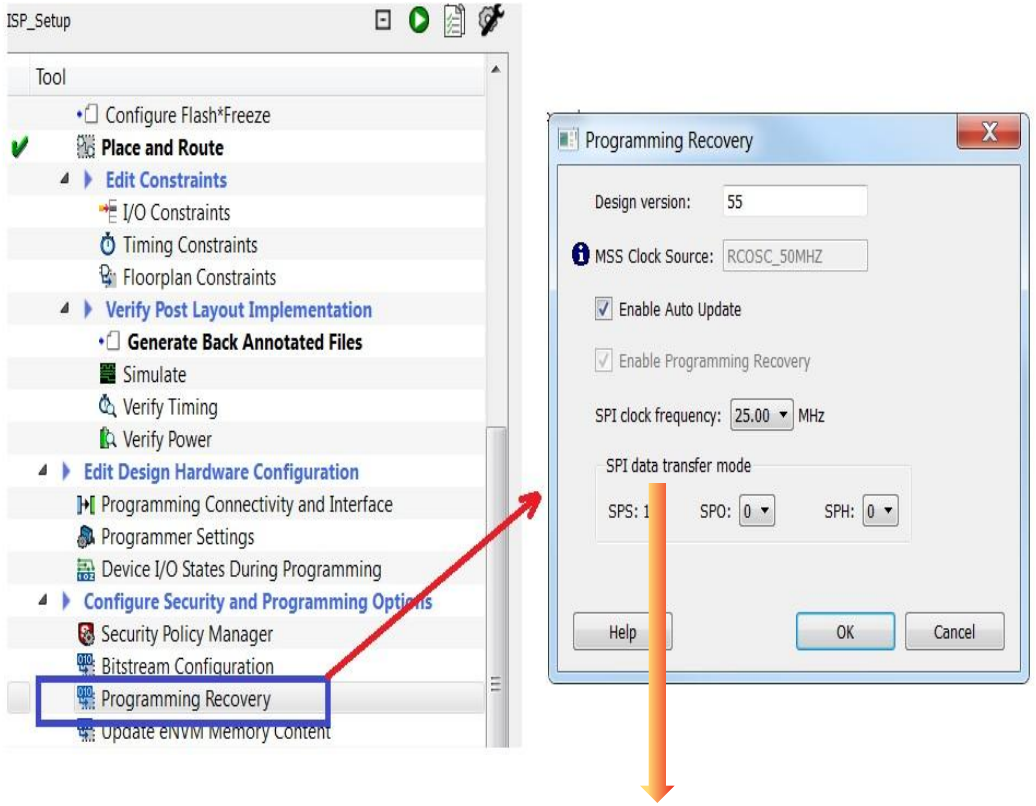
- For remote updates program the device with recovery during production
- Do not change the configuration of the Recovery Settings remotely
  - The recovery configuration settings may be corrupted
  - Program/Erase of the security are disabled to prevent the corruption of the recovery setting in the event of programming interruption which could result the recovery to fail
- Bitstreams need to be created as remotely updateable
- The Golden Image can be overwritten – no restriction
  - This allows the golden image to be updated over time

Programming Recovery (cont.)



Programming Recovery Settings

- User Configuration Settings, set at design time
  - IAPRECOVERY_ENABLE
  - IAPRECOVERY_PORT_GOLDEN
    - Defines which SPI port contains the Golden Image
      - SPI_0 – user write, SC read
  - IAPRECOVERY_UPDATE
    - On Power-up and IAPRECOVERY_ENABLE = 0
      - If update image design Version number is > Design Version Number in Fabric
      - Program the Device with new “update” bitstream

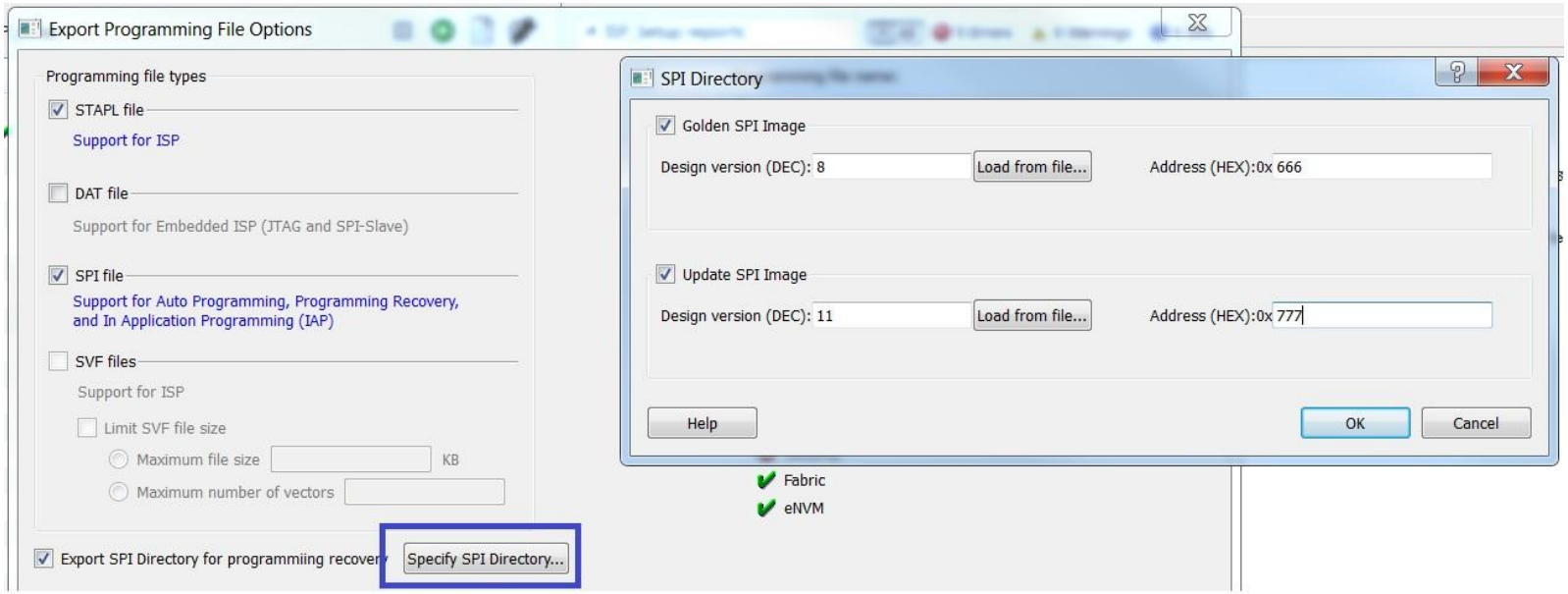


SPI Data Transfer Mode (0,1,2,3)

SPO	SPH	Clock in Idle	Sample Edge	Shift Edge	Select in Idle	Select between frames
0	0	Low	Rising	Falling	High	Stays active until all the frames set by frame counter have been transmitted
0	1	Low	Falling	Rising	High	
1	0	High	Falling	Rising	High	
1	1	High	Rising	Falling	High	

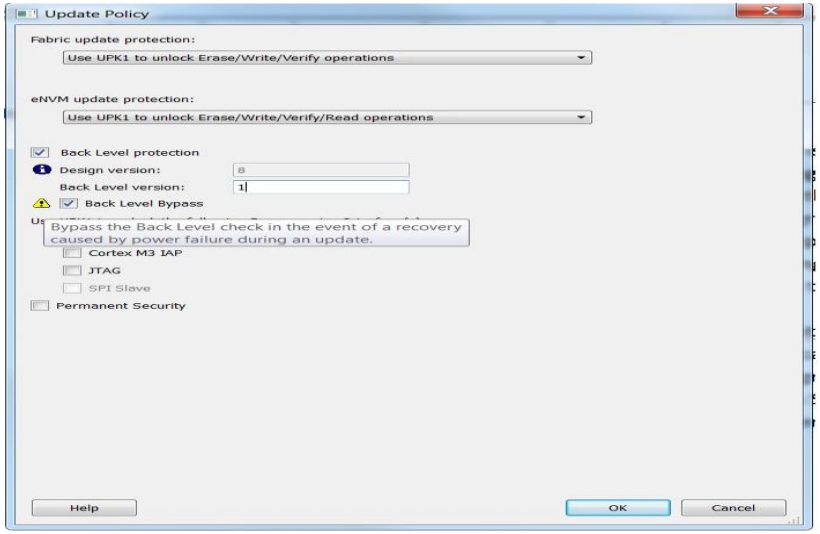
Programming Recovery Settings (cont.)

- User Configuration Settings
  - IAPRECOVERY_IMAGE
    - Defines which image to use when in recovery mode (Update or Golden)
  - SPI directory (part of bitstream)
    - User needs to program it at address zero of SPI flash



Programming Recovery Settings (cont.)

- Back level protection
  - BACKLEVEL value limits the design versions that the device can update
  - Bitstreams with DESIGNVER > BACKLEVEL are allowed for programming
    - Golden Image DESIGNVER > the existing image's BACKLEVEL version
  - Enable it in SPM (security policy manager)>Update Policy
    - Back level bypass is enabled by default



Recovery Truth Table

IAPRECOVERY_ENABLE	IAPRECOVERY_UPDATE	IAPRECOVERY_PORT_GOLDEN	IAPRECOVERY_IMAGE	Events During Assertion of FLASH_GOLDEN_N	Events During Recovery	Events During Power-Up
				At power-up, first check the zeroization status. If the device is not zeroizing and Auto Programming is selected in SPM, and FLASH_GOLDEN_N is asserted	The device is not programmed at startup, and that the device is not zeroizing or virgin and FLASH_GOLDEN_N did not assert	The device is programmed and ready to be used and was not programmed at power-up through recovery or auto programming. And the design version of the image is higher than the design version in the device
0	0	1	0	auto program golden image from SPI-0	auto program update image from SPI-0	auto program update image from SPI-0
0	0	1	1	auto program golden image from SPI-0	auto program golden image from SPI-0	auto program update image from SPI-0
0	1	1	0	auto program golden image from SPI-0	auto program update image from SPI-0	do nothing
0	1	1	1	auto program golden image from SPI-0	auto program golden image from SPI-0	do nothing
1	x	1	x	auto program golden image from SPI-0	do nothing	do nothing

Design Version

- A System service call fetches the design version
  - 16-bit data
  - The service request includes a service command and a pointer to a buffer in MSS/HPMS memory space to receive the result
  - User can specify buffer address in service request
  - Memory buffer can be located in eSRAM, DDR DRAM, or FPGA fabric SRAM
  - JTAG command can read back the design version

***Table 2-9* • Design Version Service Request**

Offset	Length (bytes)	Field	Description
0	1	CMD = 5	Command
1	4	DESIGNVERPTR	Pointer to 2-byte buffer to receive the 16-bit design version

***Table 2-10* • Design Version Service Response**

Offset	Length (bytes)	Field	Description
0	1	CMD = 5	Command
1	1	STATUS	Command status
2	4	DESIGNVERPTR	Pointer to original buffer from request

***Table 2-2* • Device and Design Information Services Status**

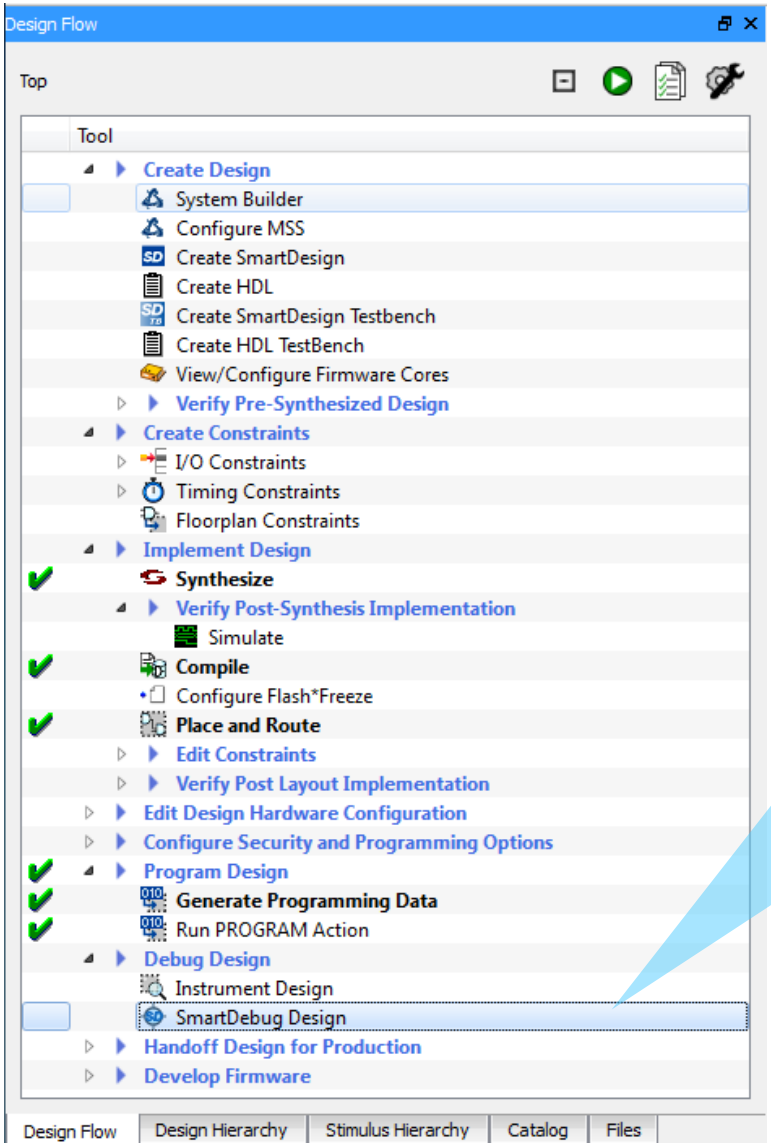
Status	Description
0	Success
127	MSS/HPMS memory access error (HRESP)

Debugging

Smart Debug

SmartFusion2 / IGLOO2

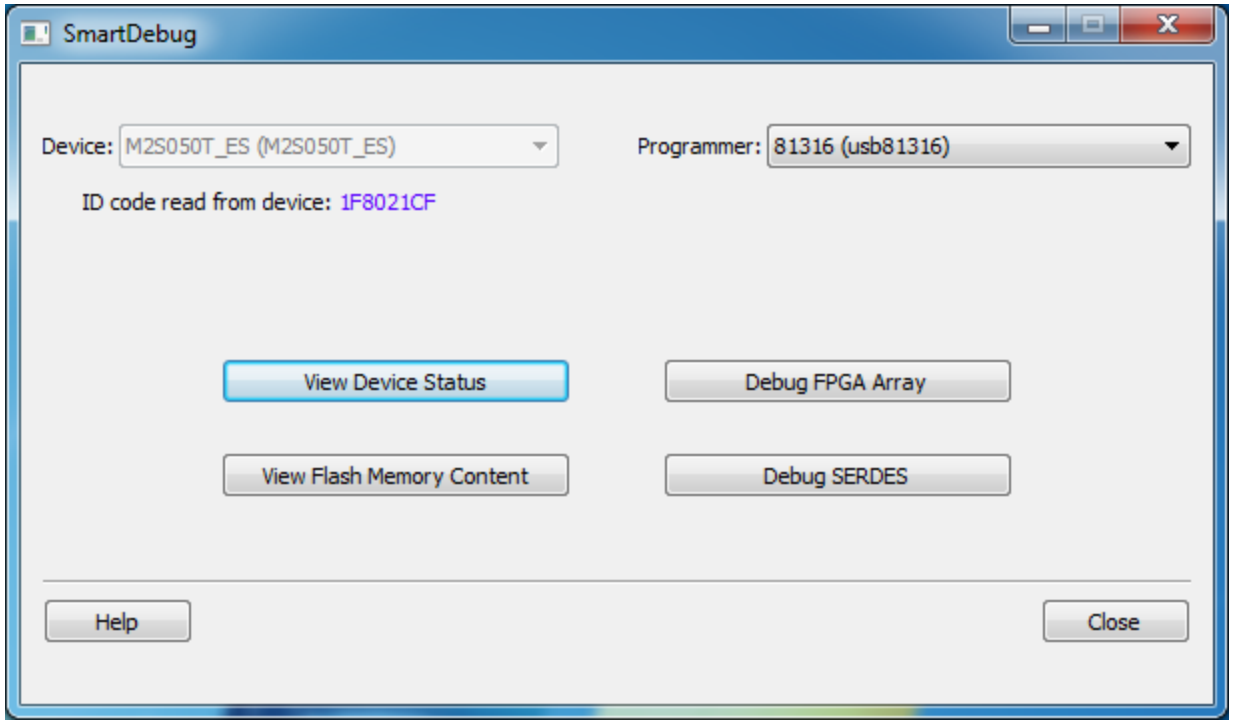
Launching SmartDebug



- SmartFusion2 and IGLOO2 only
- SmartDebug
  - Live Probe
  - Active Probe
  - Memory debug
  - SERDES debug

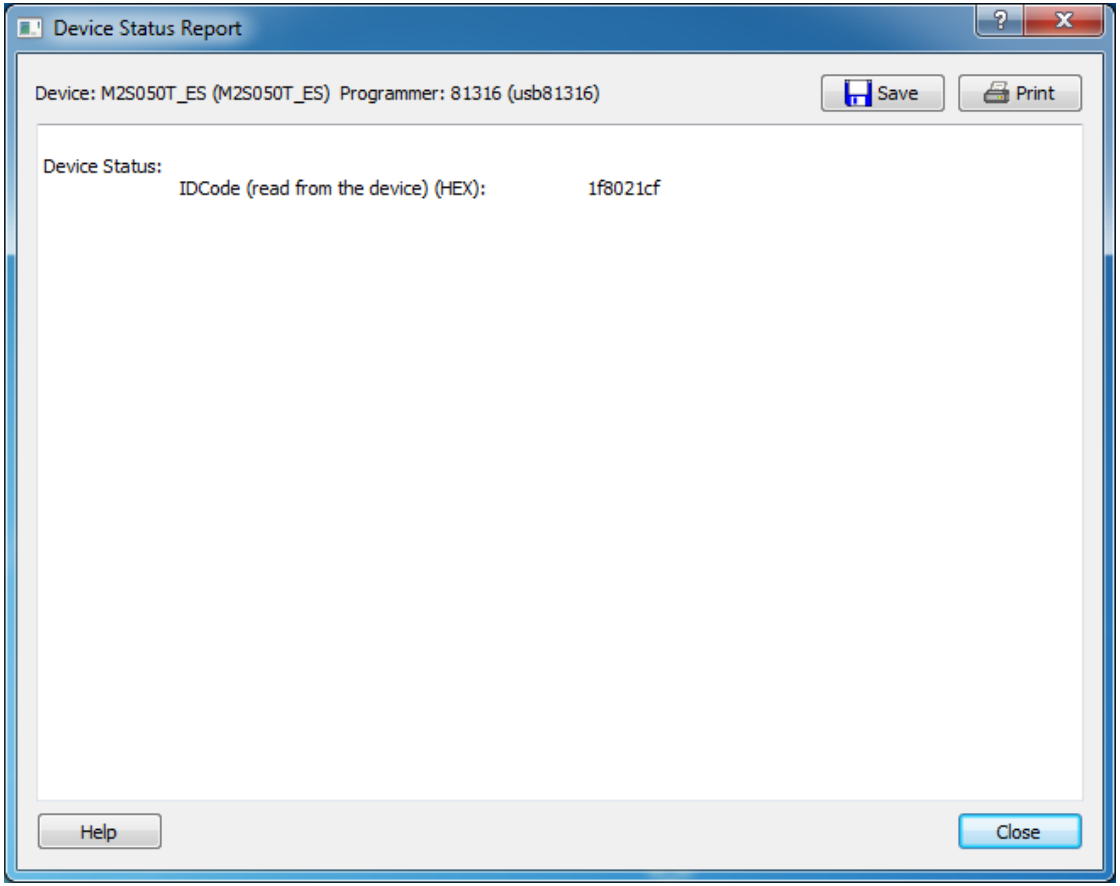
SmartDebug Options

- SmartDebug is an industry leading on-chip FPGA debug tool that gives unparalleled silicon access
  - View Device Status
  - Debug FPGA Array
  - View Flash Memory Content
  - Debug SERDES



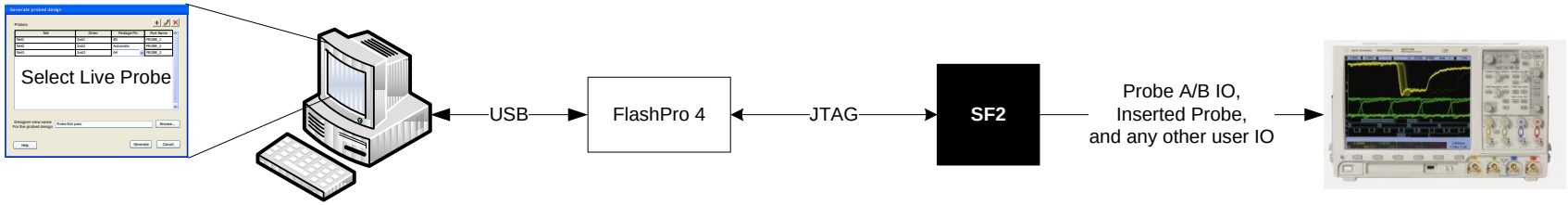
SmartDebug: View Device Status

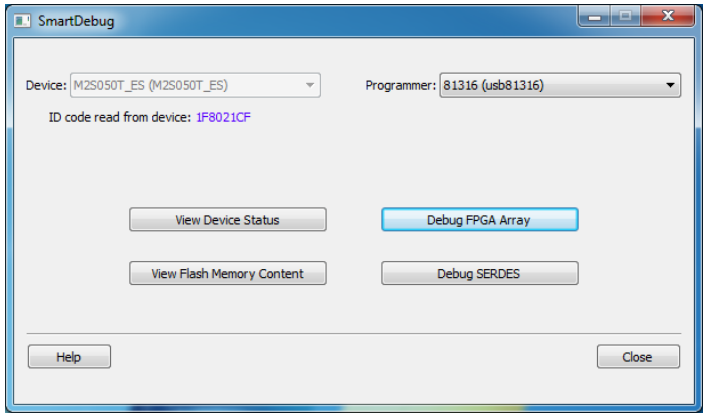
- Read Device IDCode



Debug FPGA Array (Live Probe)

- Dedicated ProbeA and ProbeB pins (PRA and PRB)

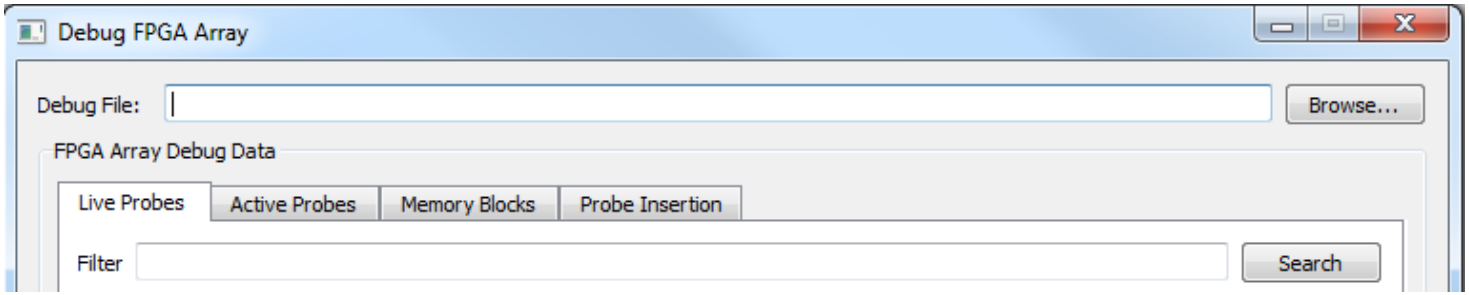




- Load Probe and u/LSRAM information from debug file generated
- Select the net name and assign either to ProbeA or ProbeB
- Set probe and SmartDebug will configure SmartFusion2 / IGLOO2 to send targeted signal to the Probe Pins for observation
- Limited to Flip-Flop only
- Post Route Probe insertion needed if user want to observe more then two internal signals

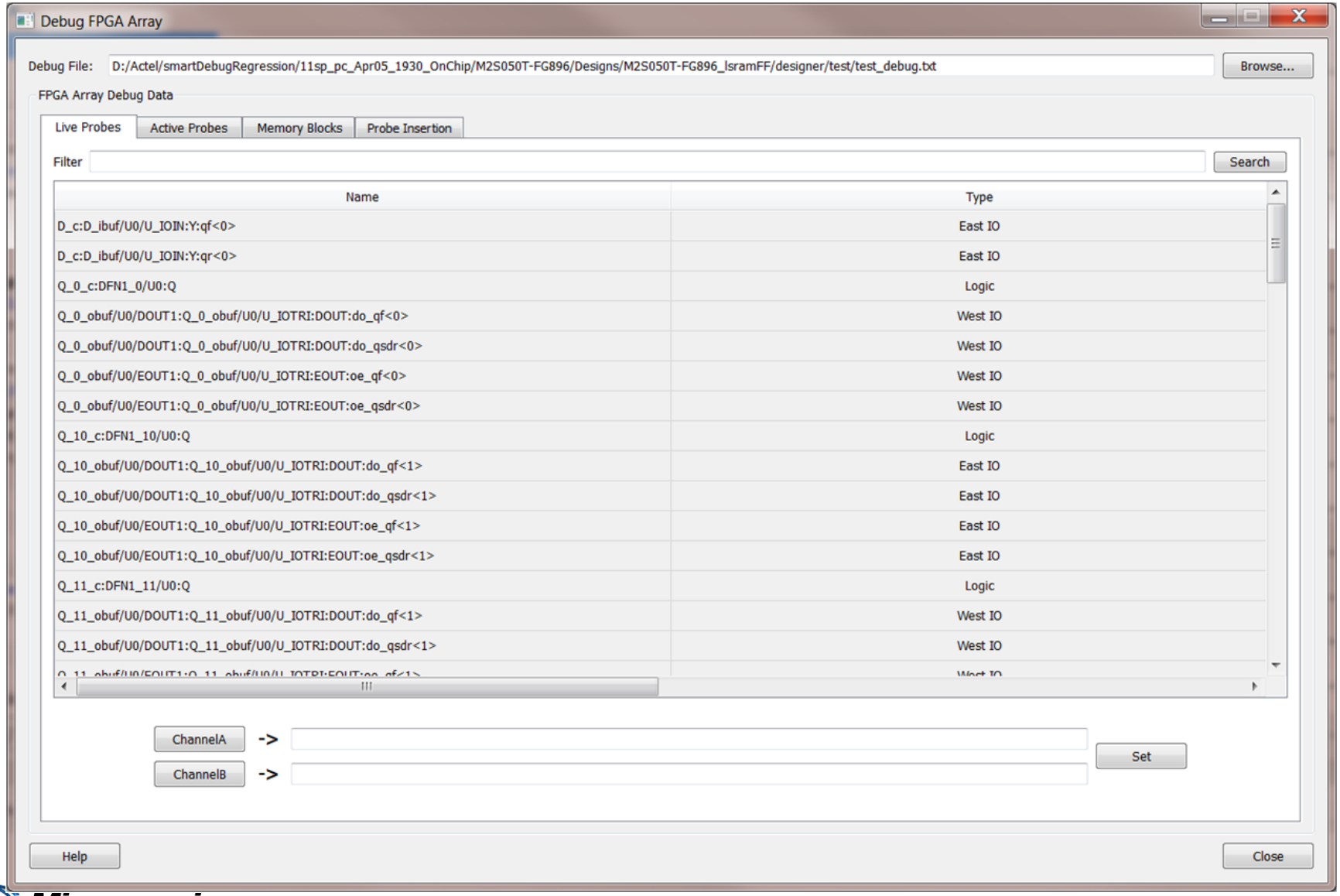
Debug FPGA Array: Loading Debug File

- Libero SoC generates the Debug File, *<projectName>_debug.txt, during Place and Route*
  - File saved into the *<project path>\designer folder*
  - The Debug File contains information used by SmartDebug mainly for mapping the user design names to their respective physical addresses on the device
- Pre Libero v11.5,location of Debug File must be specified
  - **Browse** and then select the ***<projectName>_debug.txt*** file
  - *Debug File = <project root>\designer\<projectName>_debug.txt*

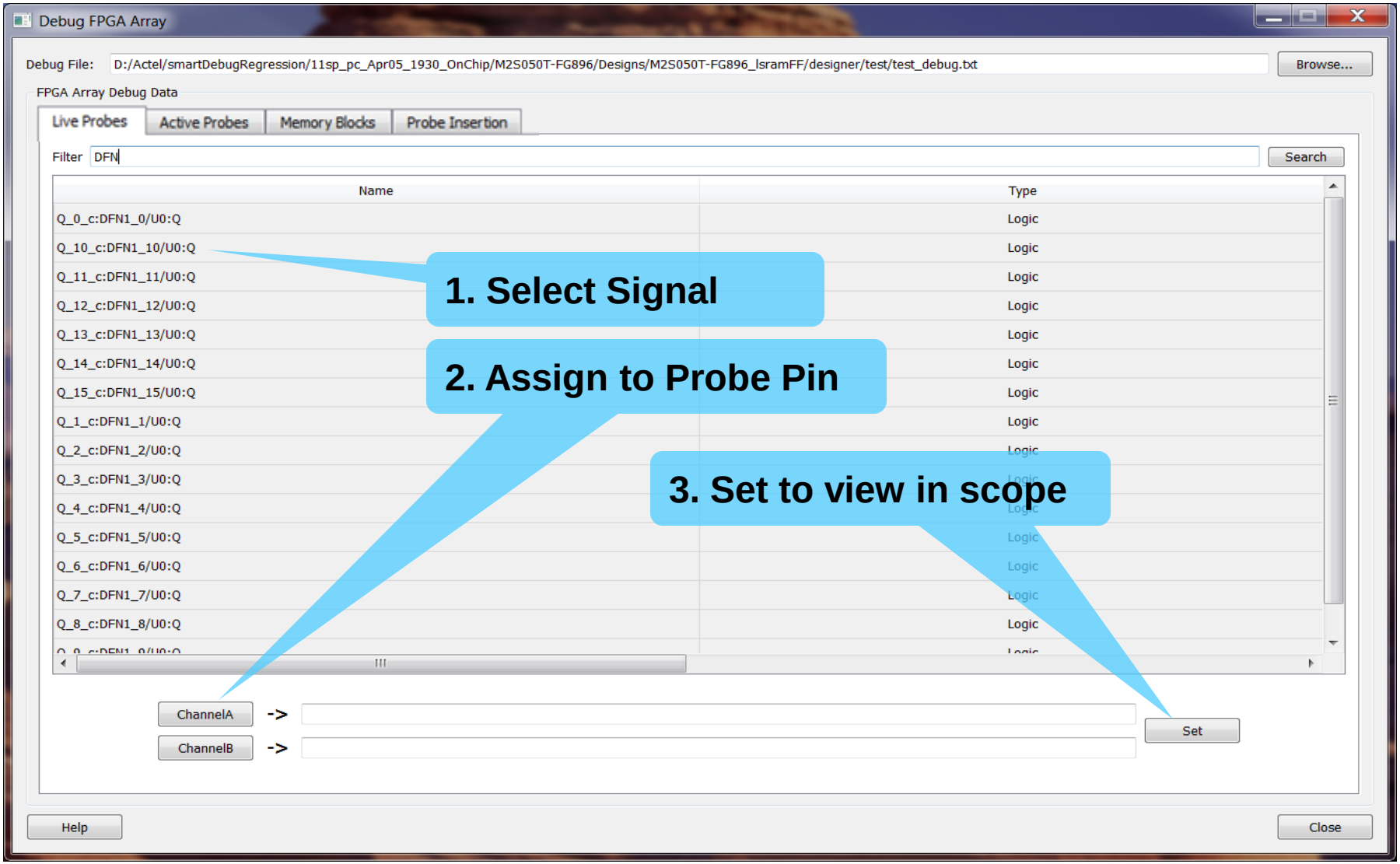


- Starting with Libero 11.5, Debug file is automatically selected

Live Probe: Pre-filtered List

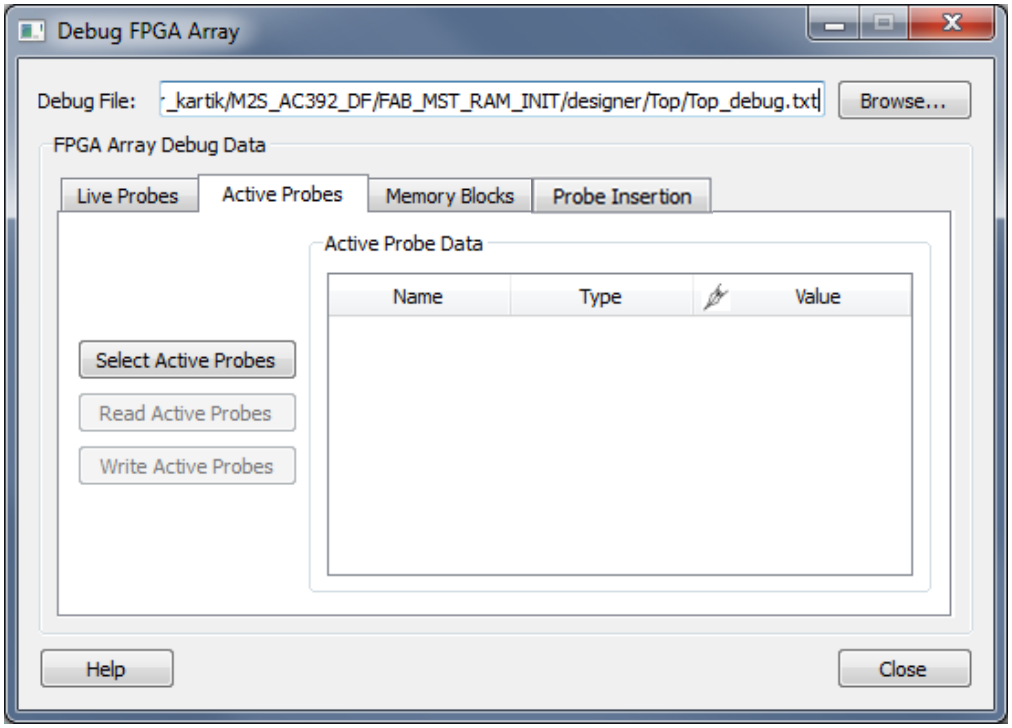


Live Probe: Filtered List

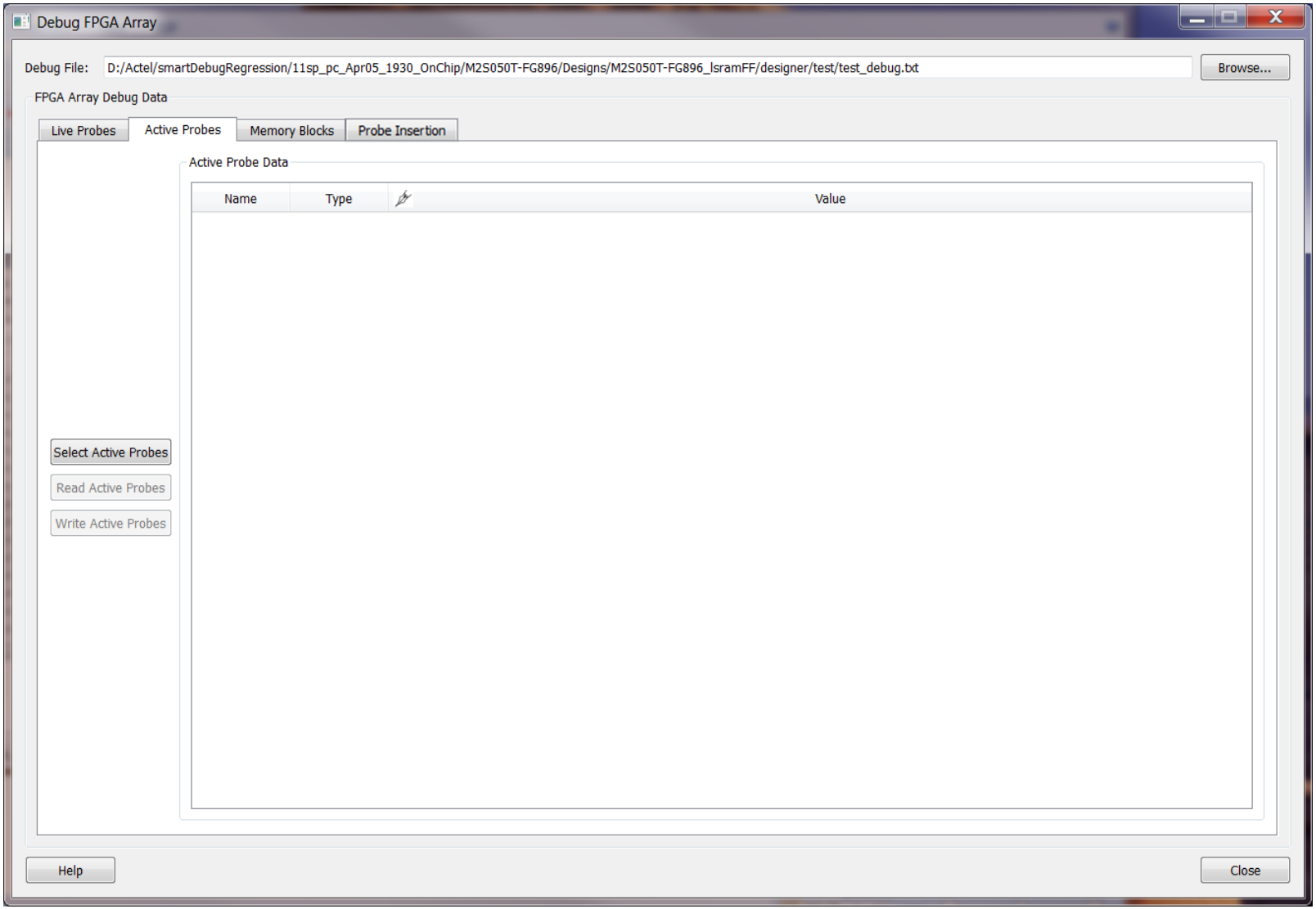


Active Probes: Asynchronous

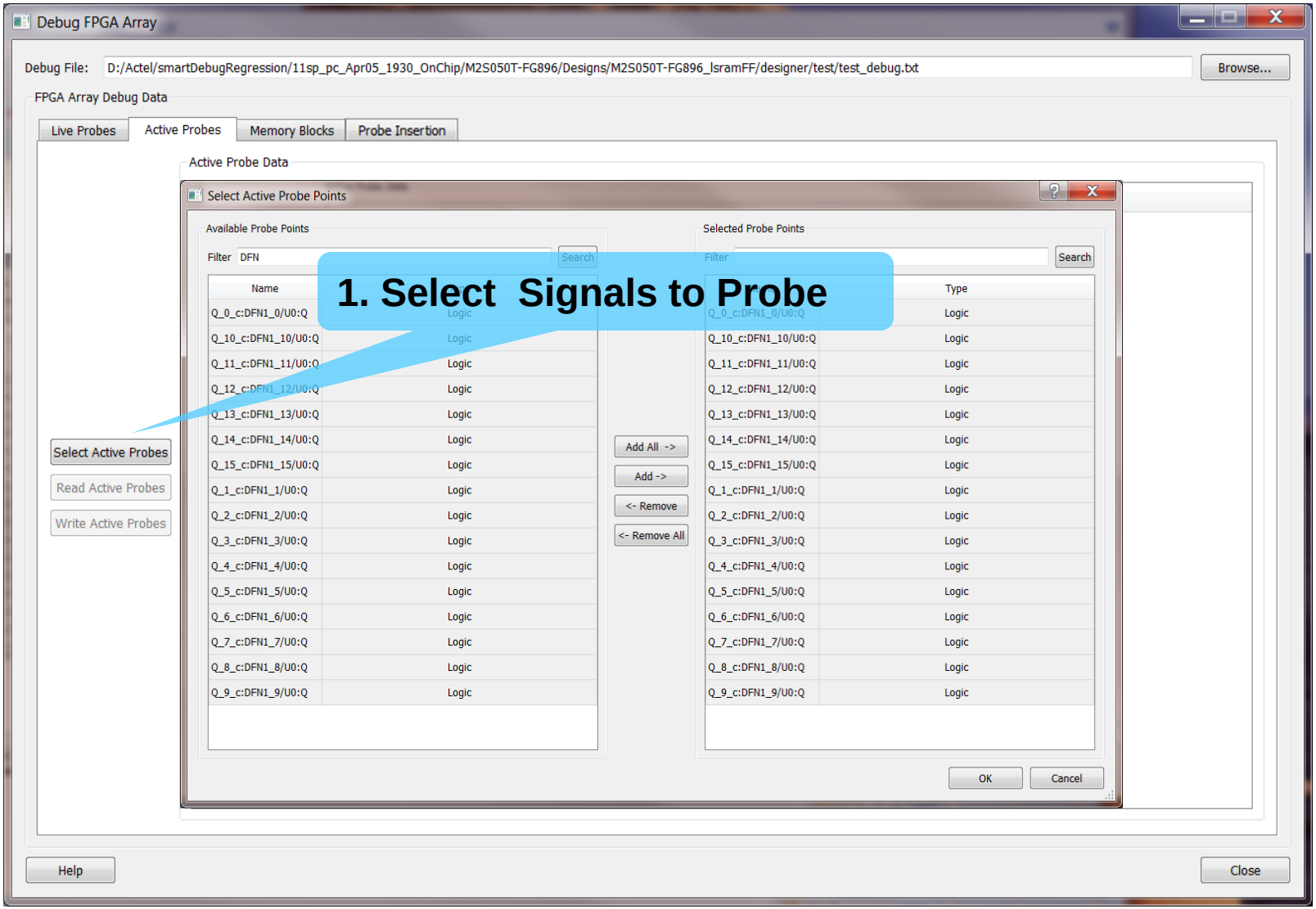
- Dynamic asynchronous read and write to a probe point
- Capabilities
  - Quick observation if internal signal is live
  - Quick experiment to affect the logic by writing to probe point
    - The value written will persist for 1 user clock cycle



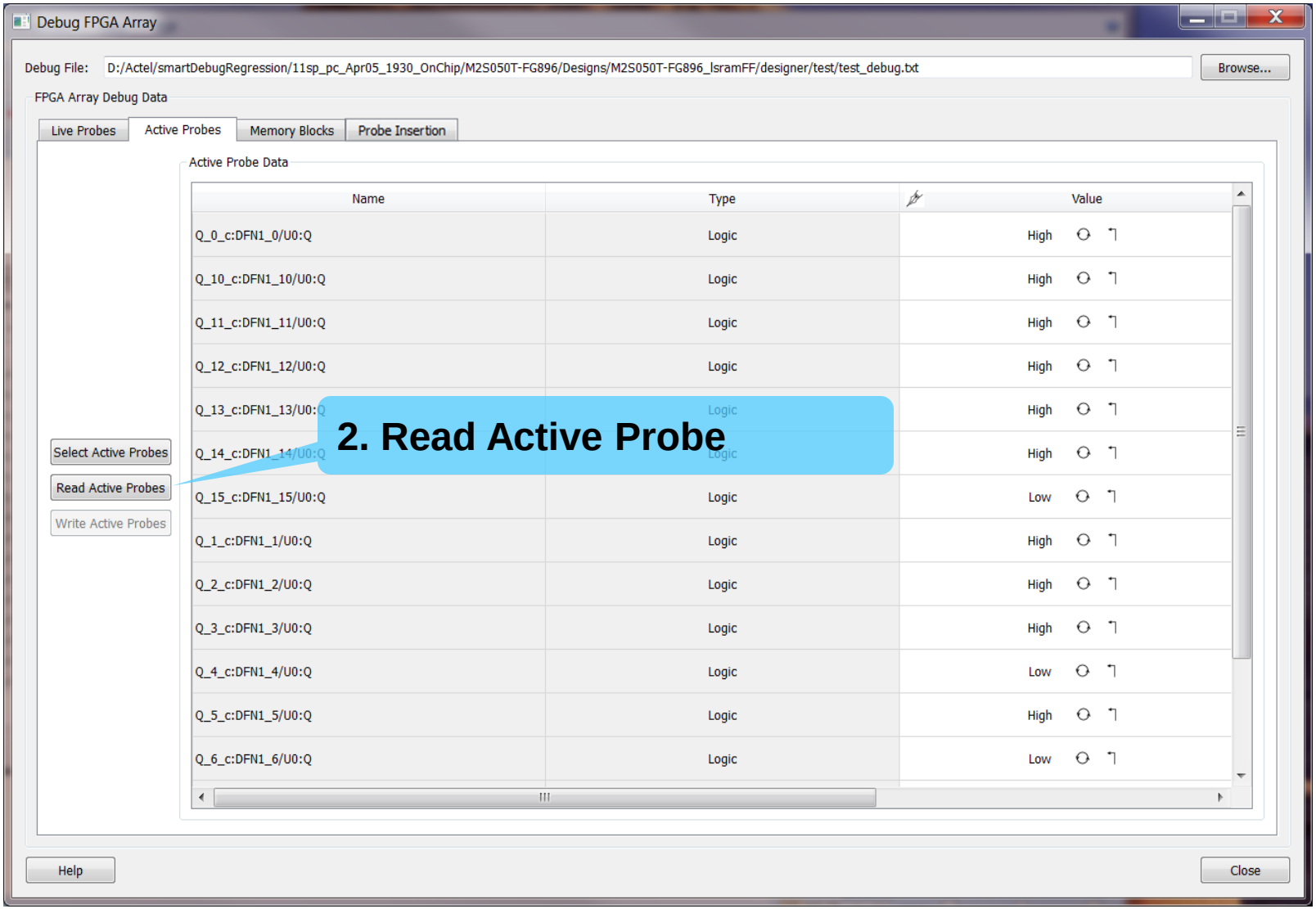
Active Probe



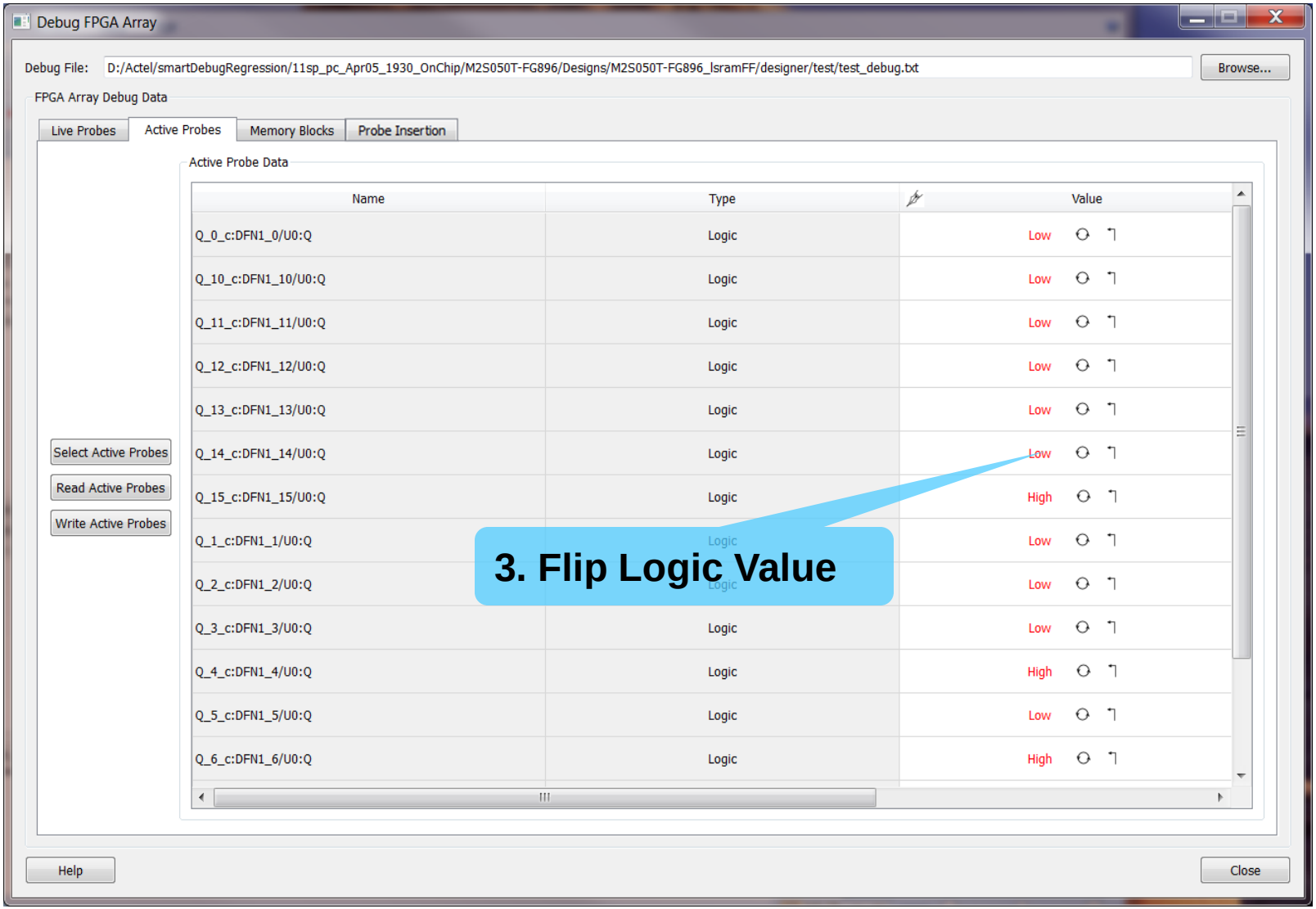
Active Probe



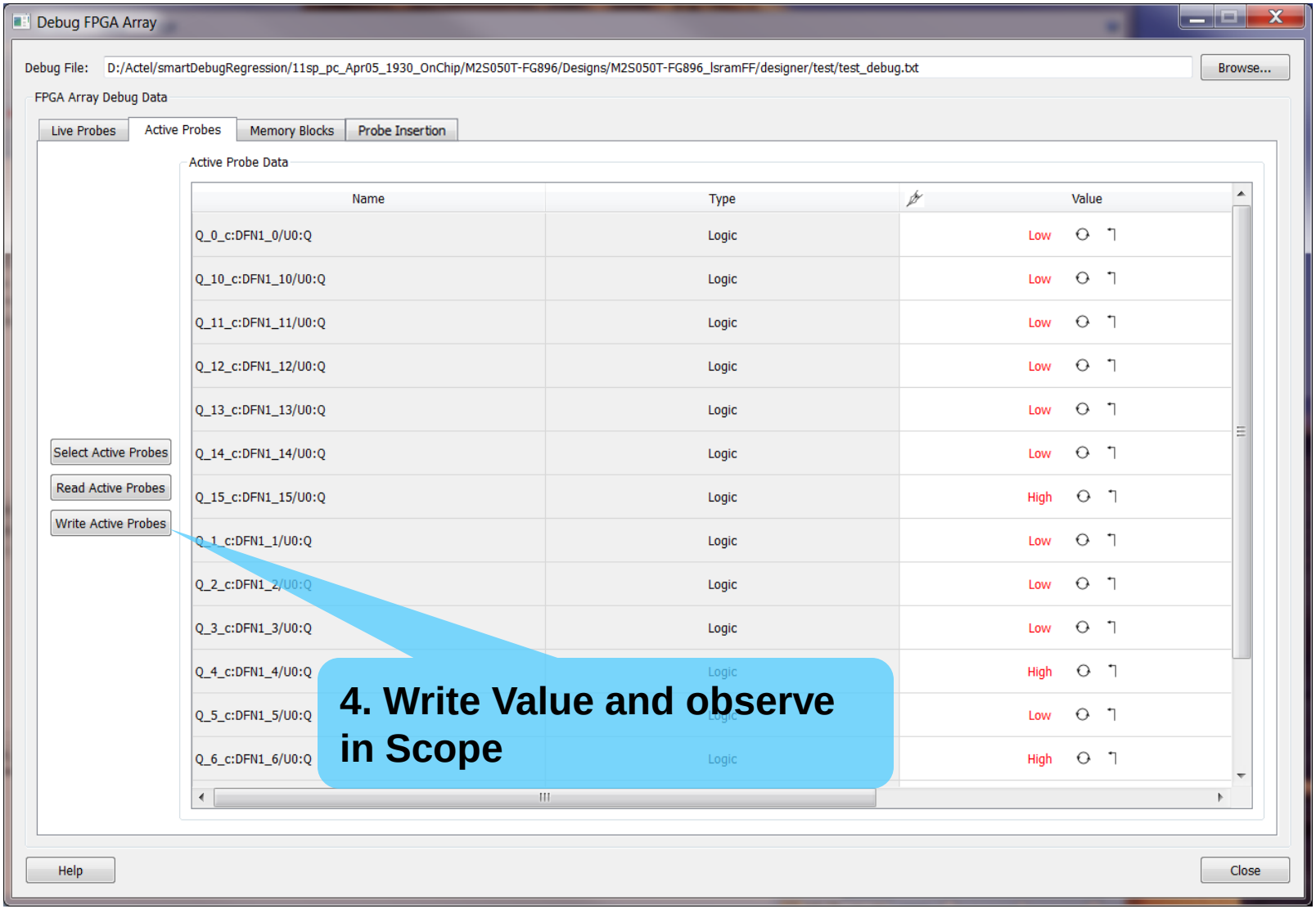
Active Probe



Active Probe

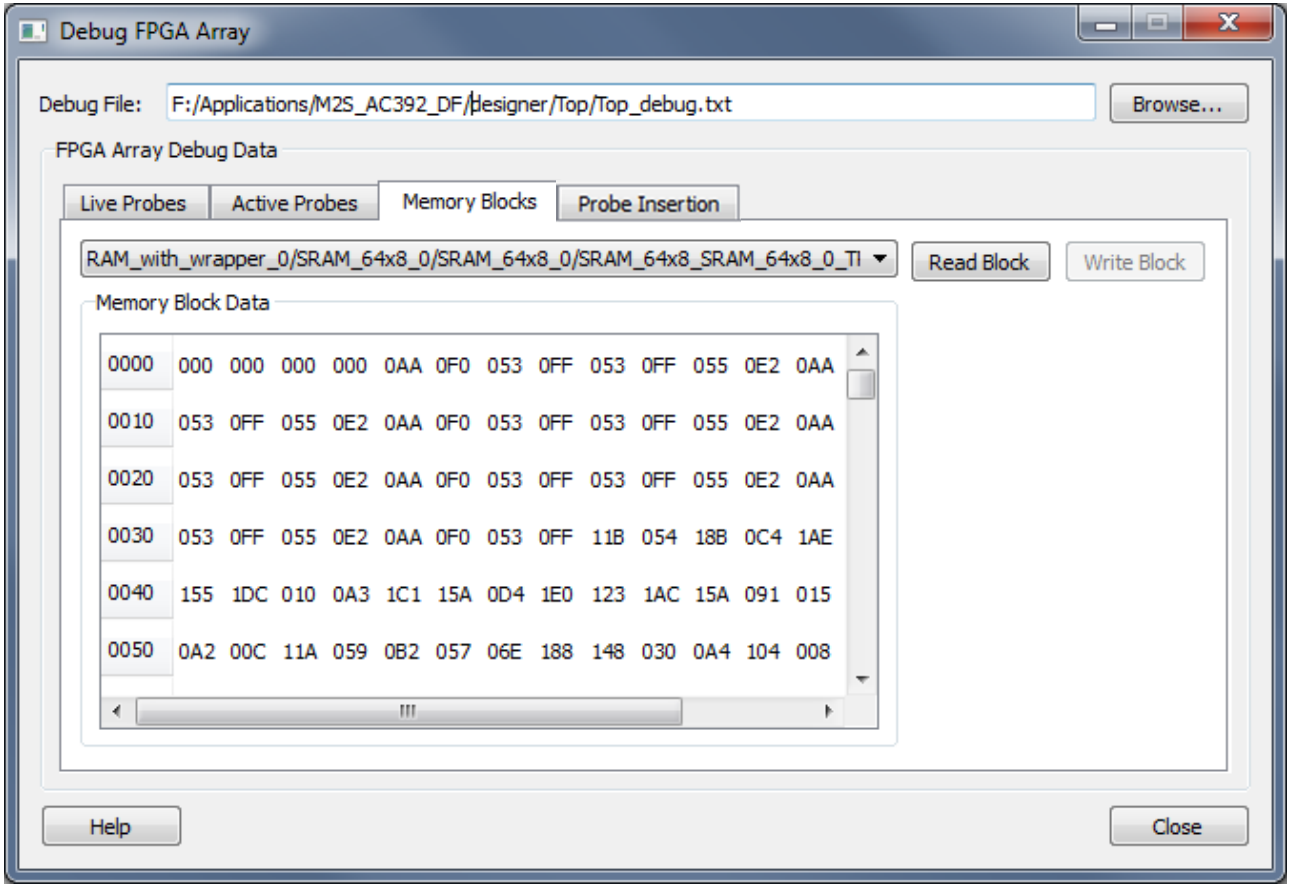


Active Probe

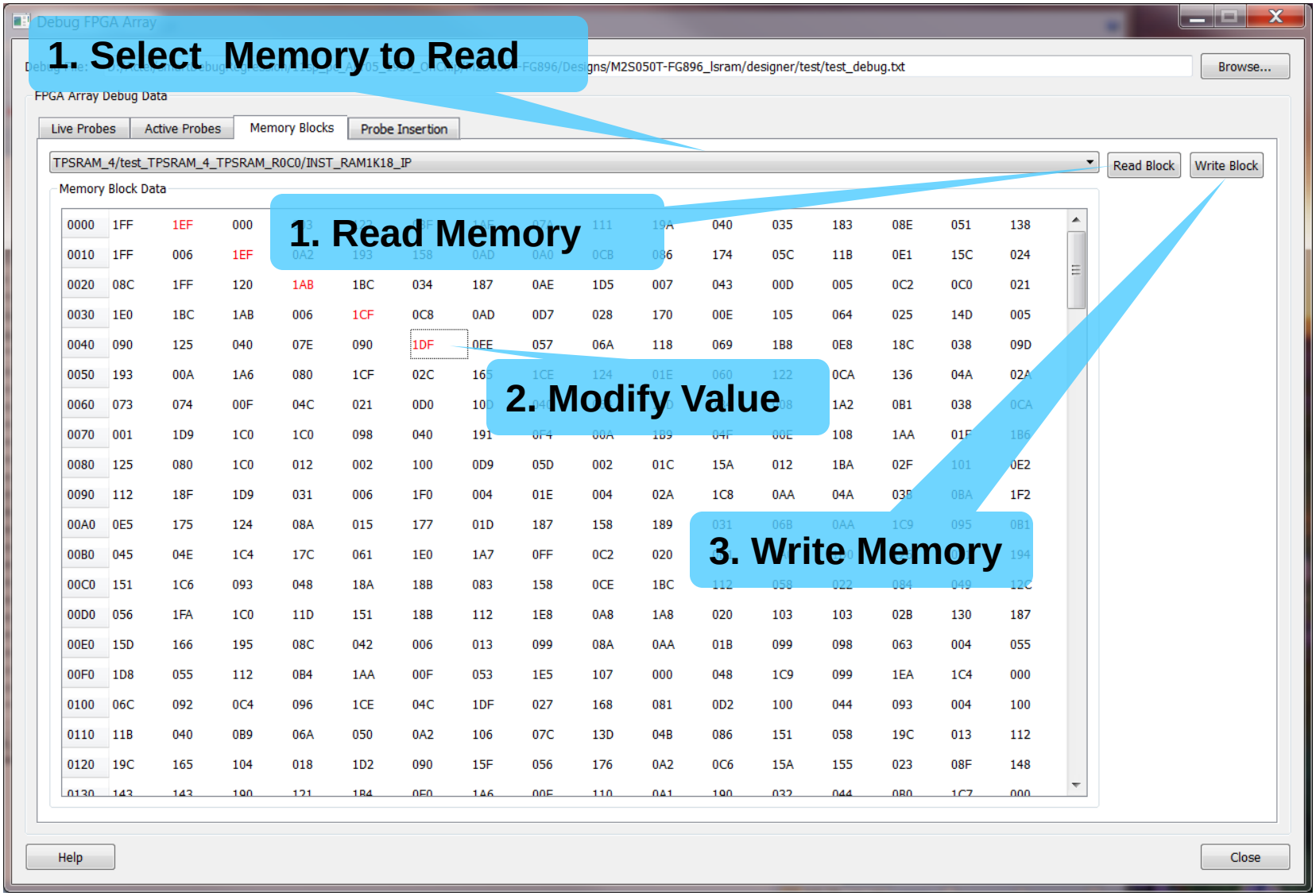


Memory Blocks Debug

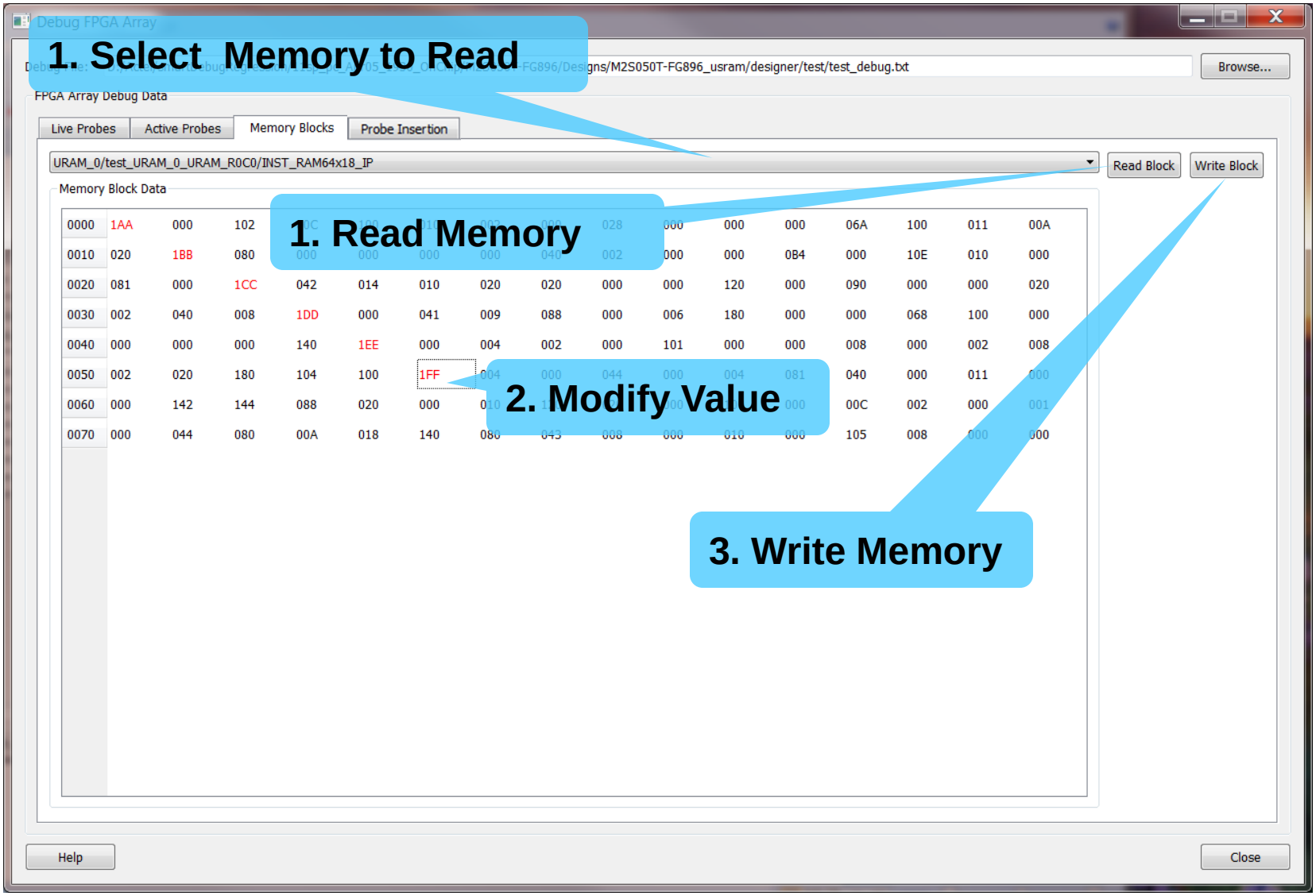
- Dynamic asynchronous read to a u/LSRAM block
- Capability:
  - Quick verification if content of u/LSRAM changing as expected



Memory Debug: LSRAM

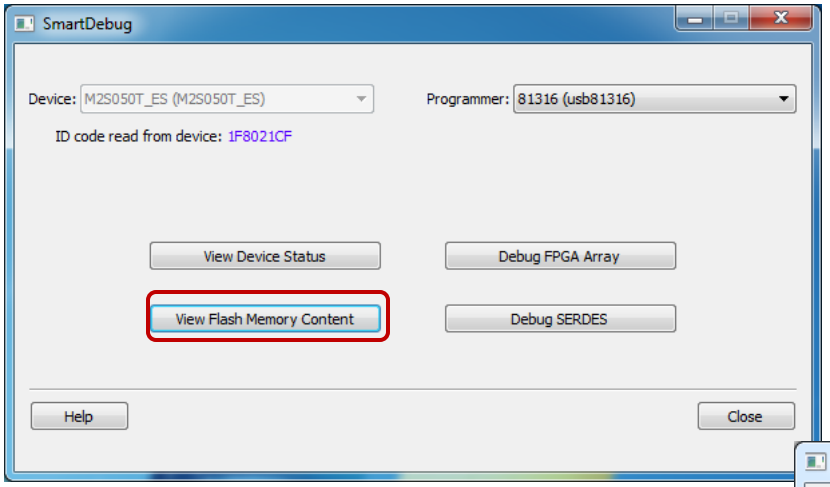


Memory Debug: uSRAM



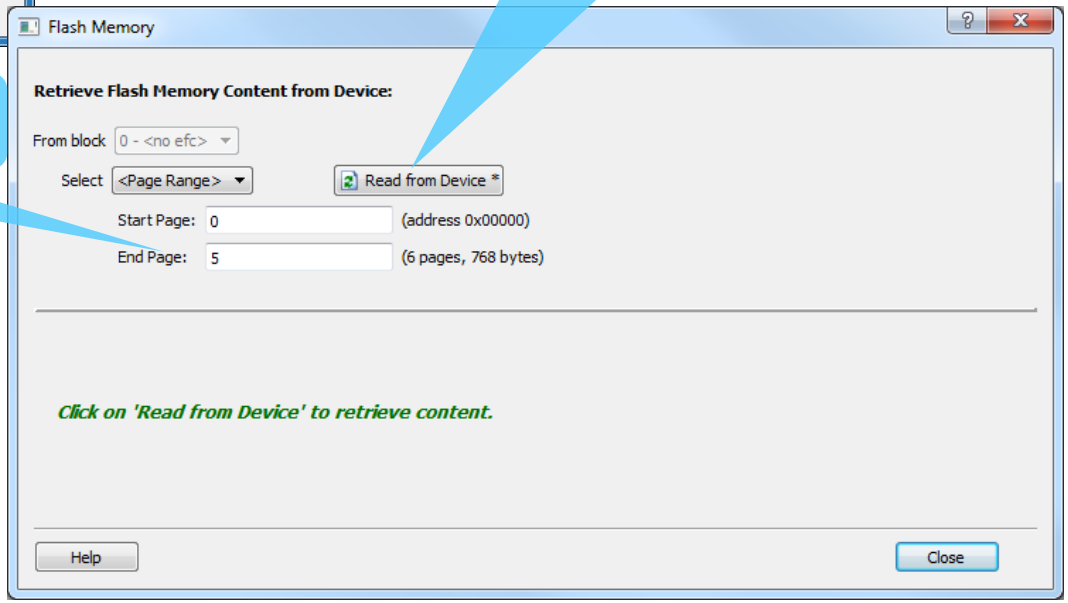
SmartDebug: View Flash Memory Content

- Read Flash Memory contents

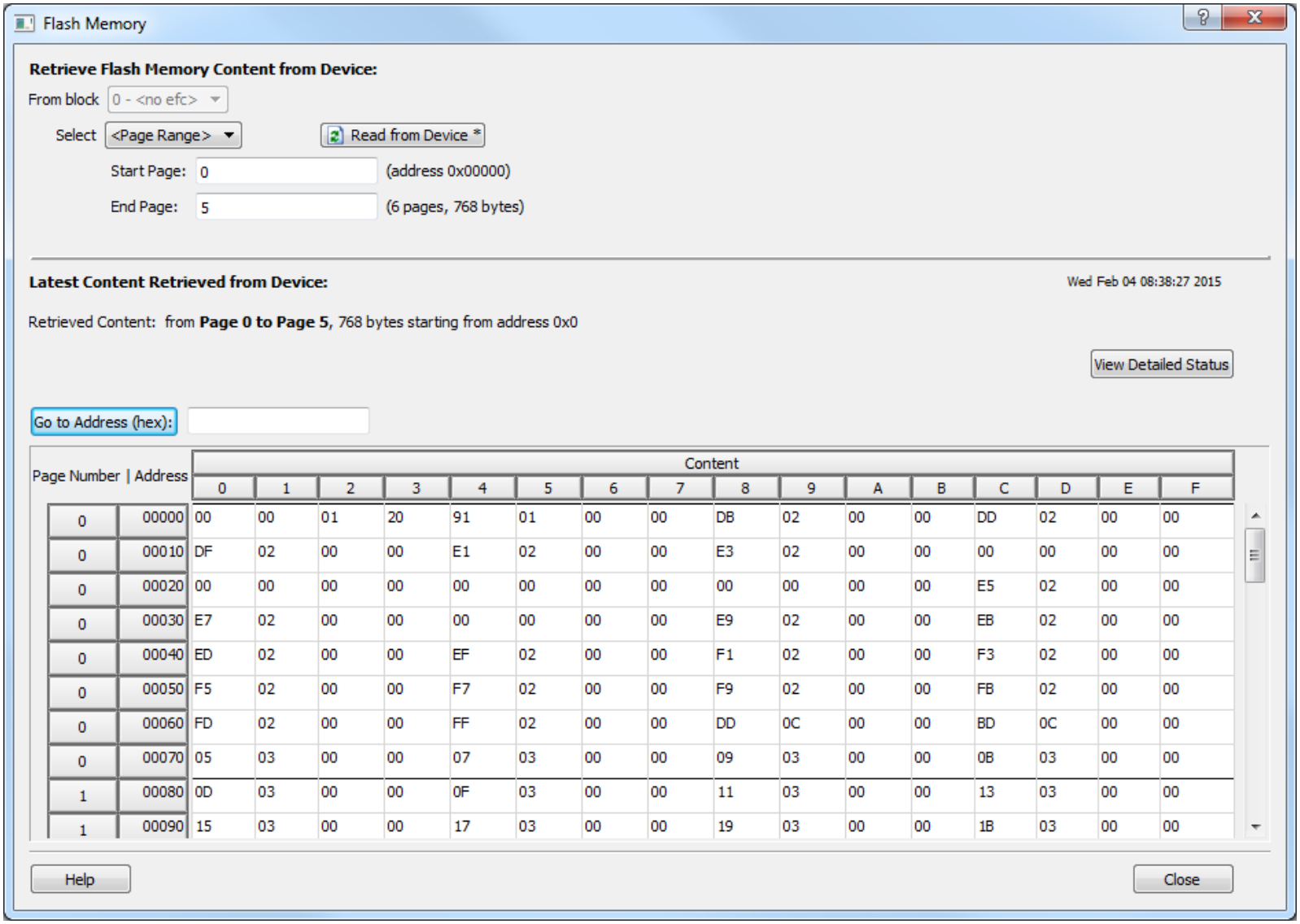


1. Enter Pages to Read

2. Read Memory



Flash Memory Content



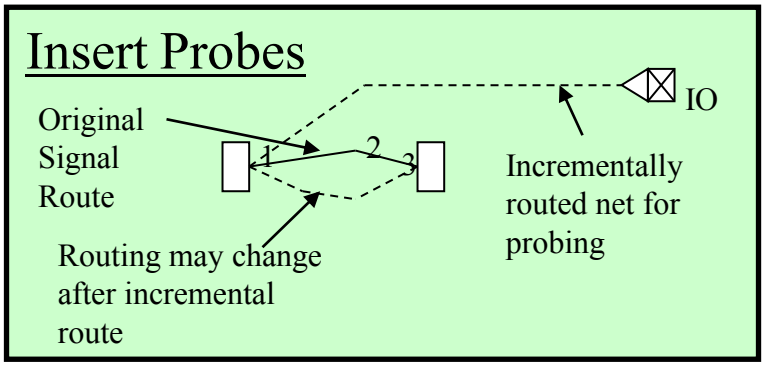
Page Number	Address	Content															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00000	00	00	01	20	91	01	00	00	DB	02	00	00	DD	02	00	00
0	00010	DF	02	00	00	E1	02	00	00	E3	02	00	00	00	00	00	00
0	00020	00	00	00	00	00	00	00	00	00	00	00	00	E5	02	00	00
0	00030	E7	02	00	00	00	00	00	00	E9	02	00	00	EB	02	00	00
0	00040	ED	02	00	00	EF	02	00	00	F1	02	00	00	F3	02	00	00
0	00050	F5	02	00	00	F7	02	00	00	F9	02	00	00	FB	02	00	00
0	00060	FD	02	00	00	FF	02	00	00	DD	0C	00	00	BD	0C	00	00
0	00070	05	03	00	00	07	03	00	00	09	03	00	00	0B	03	00	00
1	00080	0D	03	00	00	0F	03	00	00	11	03	00	00	13	03	00	00
1	00090	15	03	00	00	17	03	00	00	19	03	00	00	1B	03	00	00

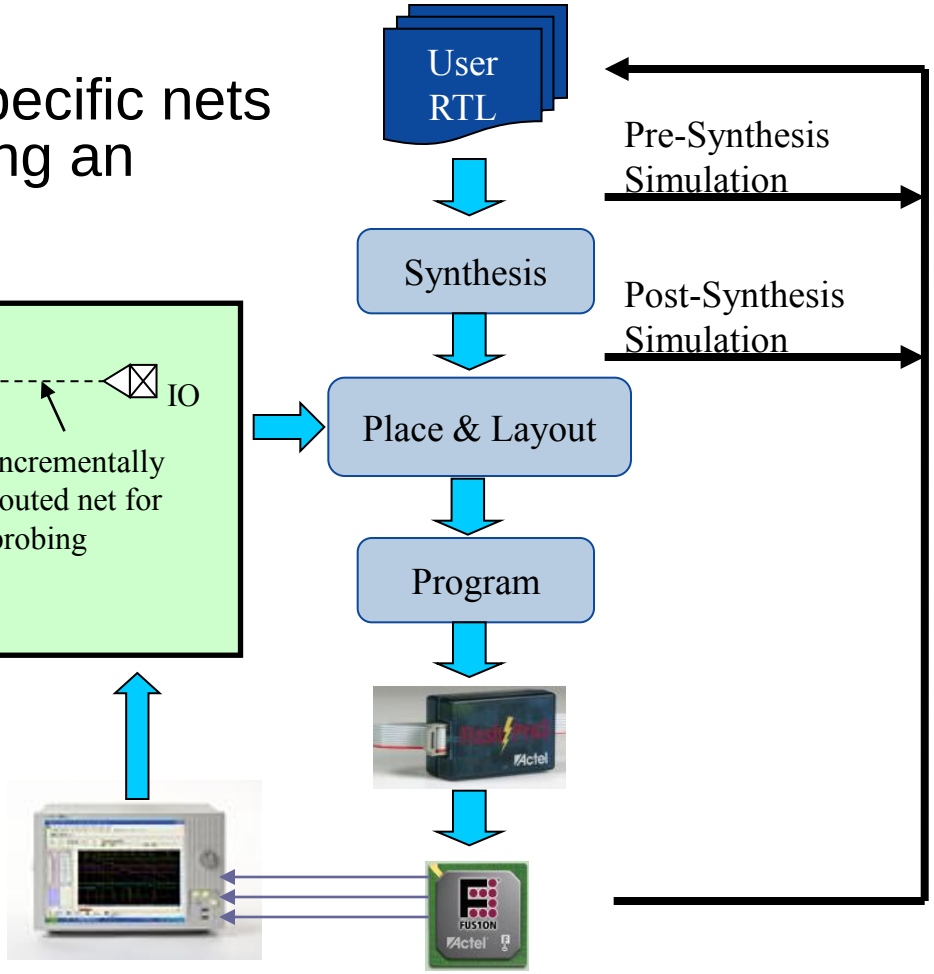
Post-route Probe Insertion

Post-route Probe Insertion Flow

Feature Overview

- Allows user to quickly probe specific nets at selected I/Os without inserting an Internal Logic Analyzer (ILA)





Key Requirements

- Multiple signal probing
- Preserve all existing routing
- Support used and unused I/O for probing

Generating a Probed Design Methodology

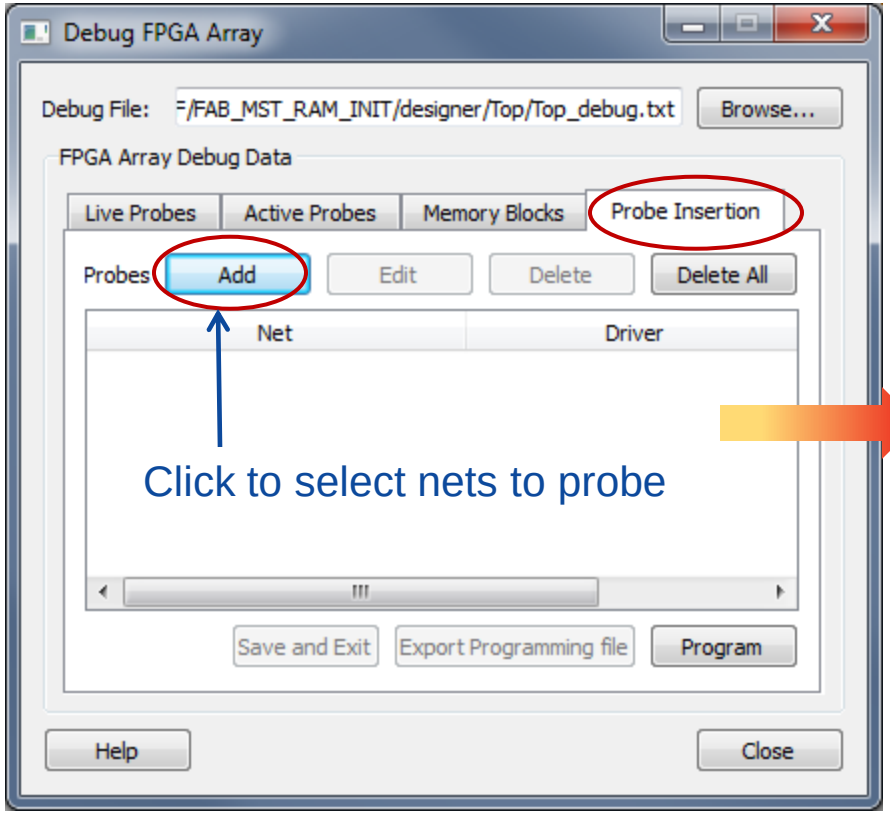
- You can ‘probe’ an internal net of your design by routing that signal to an external I/O pin
  - Pin can be unused or another pin used in the design
- A probed design exposes one or more nets within the design and makes that net available to an off-chip diagnostic tool
  - No need to instantiate an internal logic analyzer
- Supported Families:
  - SmartFusion2, IGLOO2, IGLOO, ProASIC3, SmartFusion and Fusion

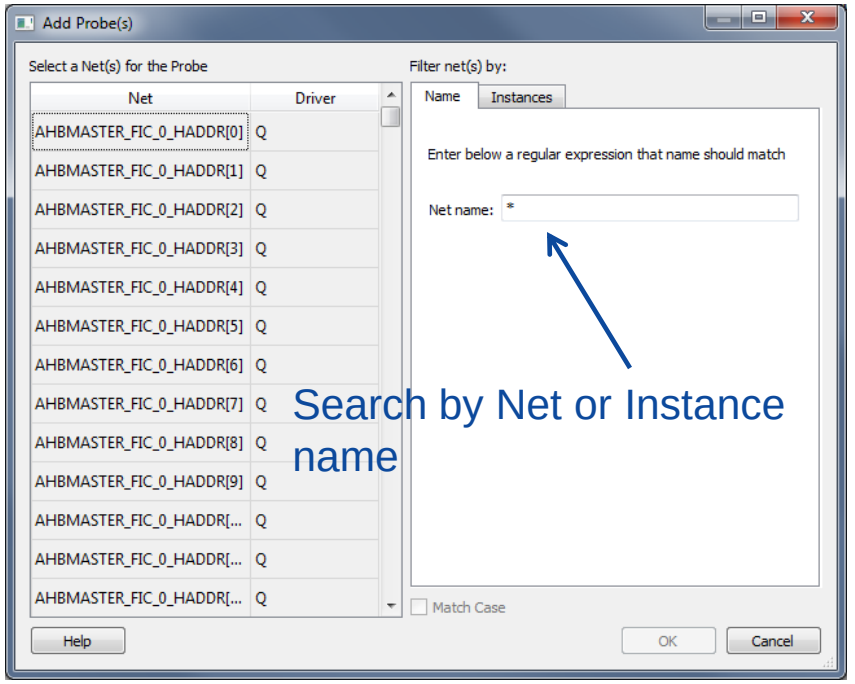
Generating a Probed Design: Steps
SmartFusion2 and IGLOO2

- Capture and synthesize the design
- Run Place & Route
- Specify nets to probe using SmartDebug
- Program the device

Selecting Probe Nets

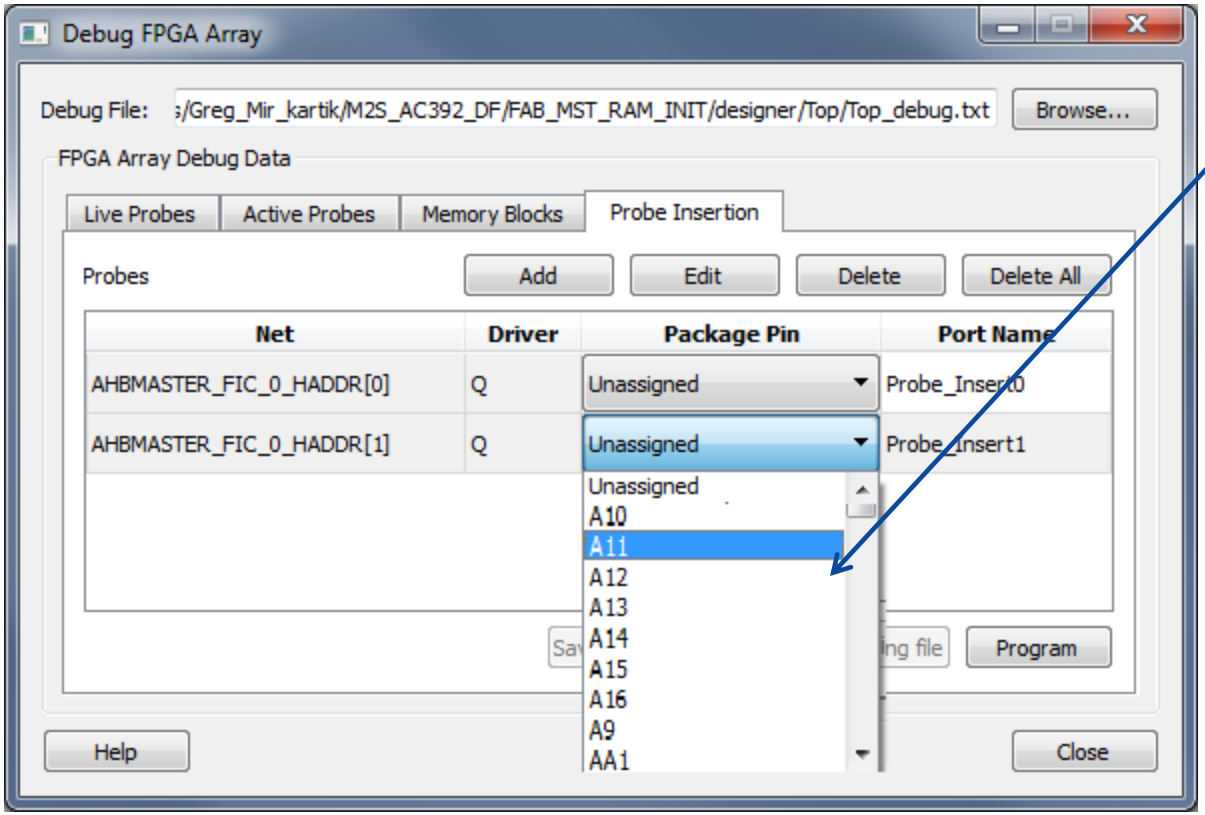
- SmartDebug > Debug FPGA Array > Probe Insertion Tab





Assigning Probe Nets to Package Pin

- Select the package pin to assign a net to and select “Program”
- Place & Route runs incrementally and automatically program the device with the assigned probes

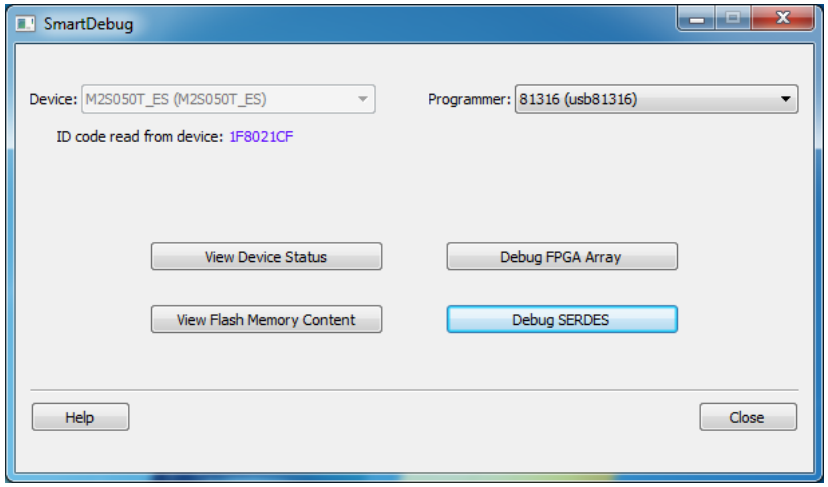


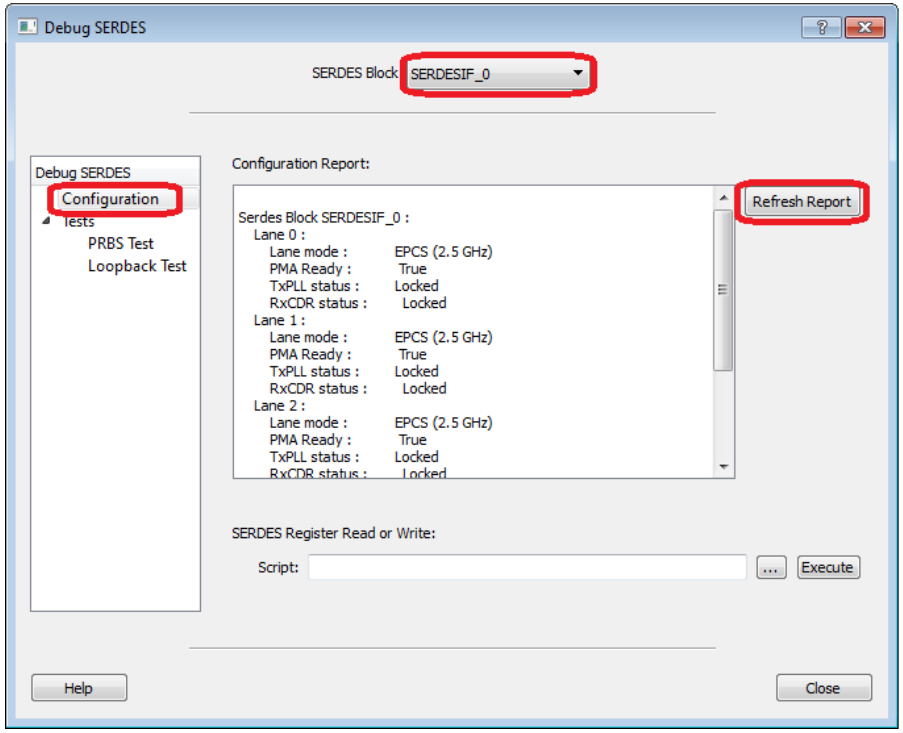
Assign probe net to any Package Pin from the drop down list

SmartDebug: Debug SERDES

- SERDES utility assists FPGA and board designers without any design modifications (in real-time) using FlashPro4 cable to:
  - Validate signal integrity of high speed serial links in a system
  - Improve board bring-up time
  - Adjustment and tune the PMA analog settings for optimal link performance to match the design to the system
  - Real-time access to SERDESIF Block control and status registers
  - Provide testing functions with pseudo-random binary sequence (PRBS) or constant pattern generators and checkers
  - Run link tests with various loop back options

SmartDebug: Debug SERDES

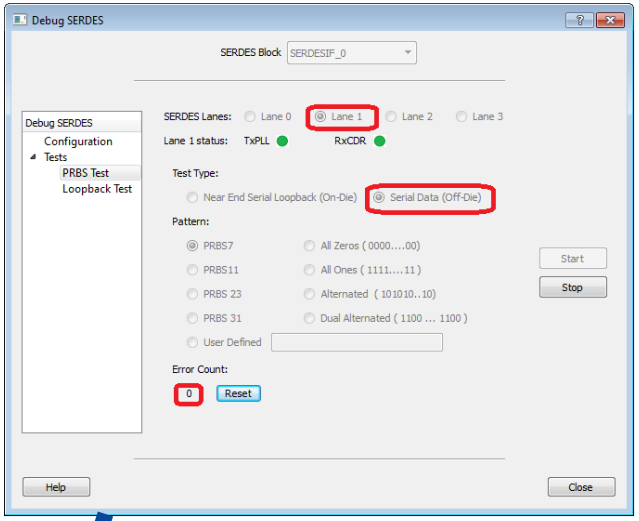


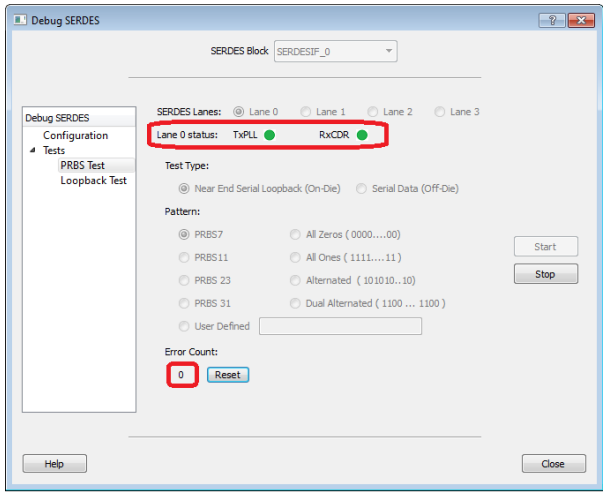


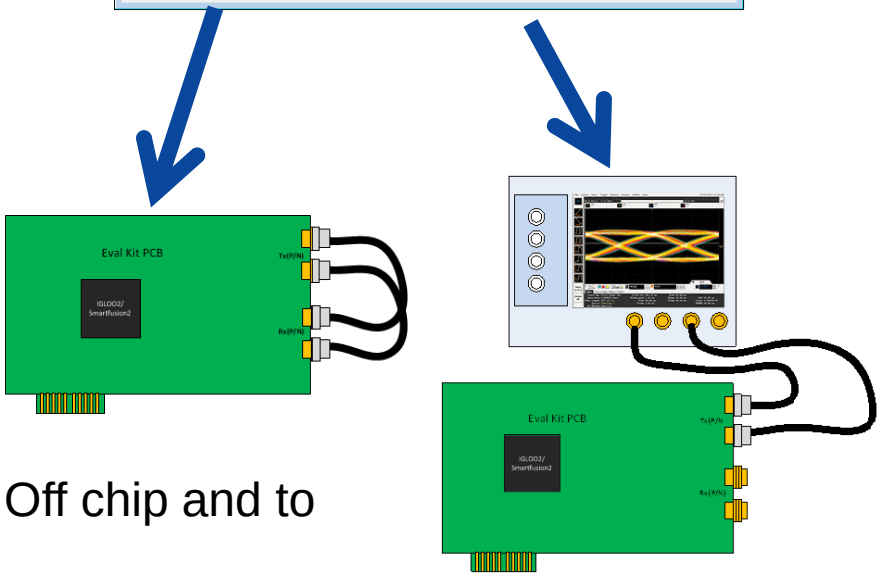
- Configuration tab will:
  - Auto-identify and populate which SERDESIF is used in the design
  - Show the lanes and how they are programmed and powered-up
  - Show the status of each lane as well as the programmed lane mode

SmartDebug: Debug SERDES

- Test SERDES Block features on-die or off-die features
- Control Embedded Test Pattern Generator and Checkers



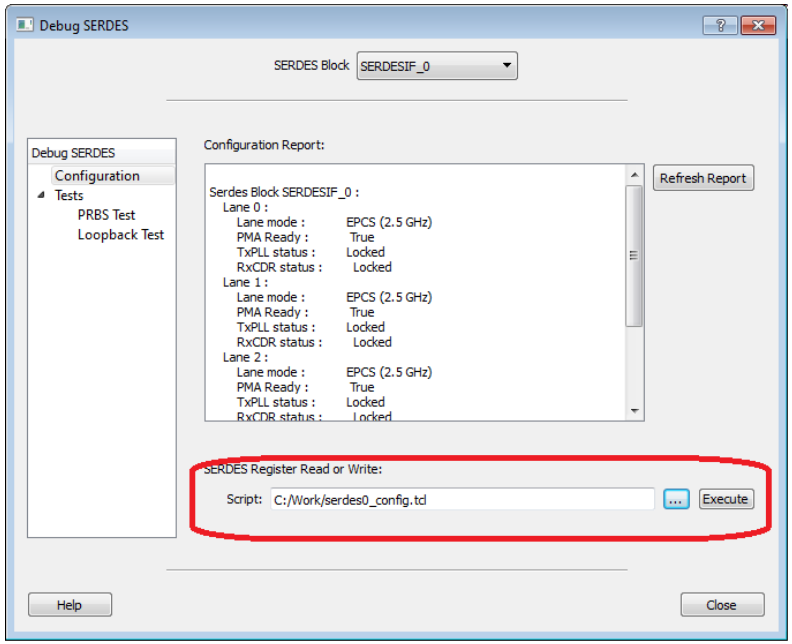


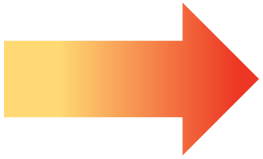


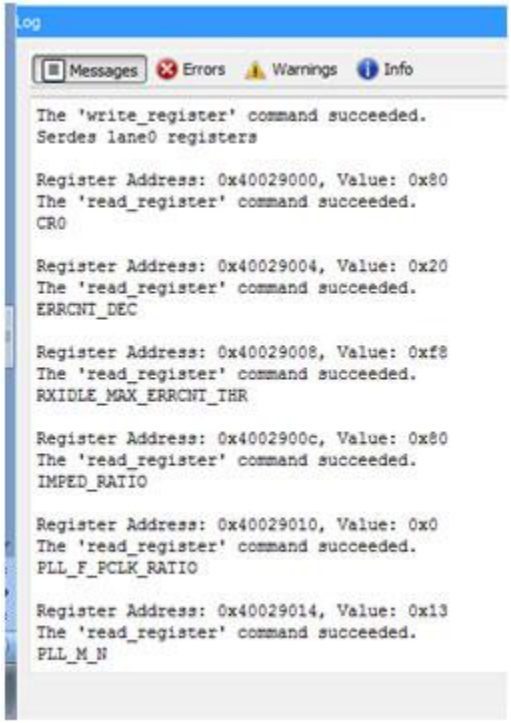
Send Test Patterns Off chip and to Test Equipment

SmartDebug: Debug SERDES

- The SERDES Debug tool set permits execution of Tcl scripts. This scripting capability allows customized writes and reads of the entire SERDES core register base
- Tcl can be used to update or check status of the SERDES system, PCIe system, and SERDES lane registers







Microsemi SmartFusion2 SoC FPGA

SmartFusion2 SoC FPGAs extend our leadership in security, reliability and low power into mainstream applications

- Leadership in Low Power FPGAs
  - 100X lower static power with same performance
- Leadership in Secure FPGAs
  - State of the art security enables root-of-trust applications
  - Radically transforms the usefulness of FPGAs in security applications
- Leadership in Reliable FPGAs
  - Only SoC FPGA with SEU hardened fabric and processor
  - Reliability designed for safety critical and mission critical systems
- Leadership in Real-Time FPGAs
  - ARM® Cortex™-M3 real-time microcontroller
  - Flash*Freeze real-time power management
  - Instant on real-time availability

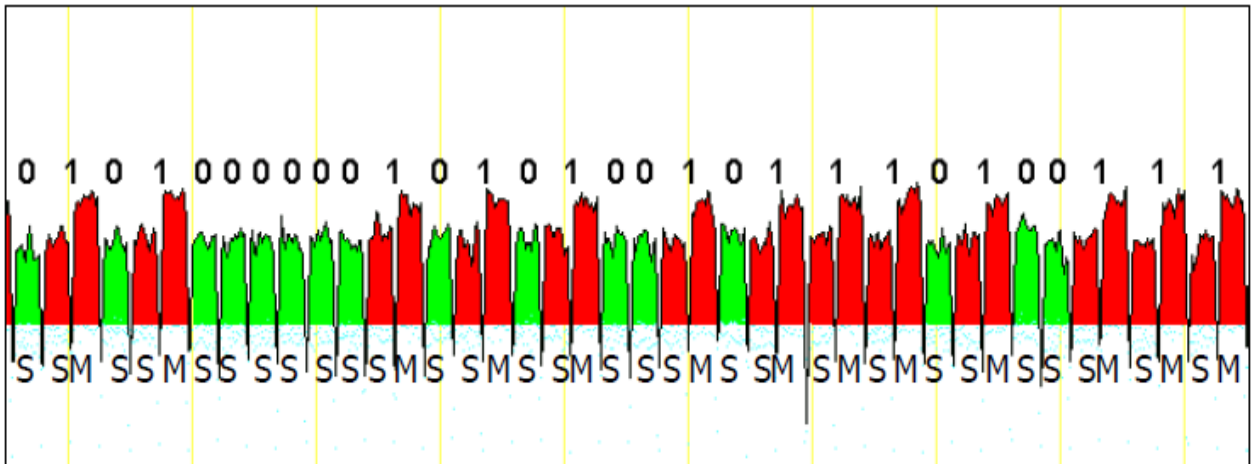
SmartFusion® 2

Highest Reliability, Most Secure, Lowest Power

Back-up Slides

Simple Power Analysis / Differential Power Analysis

- Extraction of encryption keys through measuring power of cryptographic operations
- IC power consumption depends on activity of transistors
- Measurements of device operation can directly reveal keys and other secrets



Secret Exponent
{0 or 1}

S = Square
M = Multiply

Crypto algorithms can be functionally correct but susceptible to attack!

They need to be SPA / DPA resistant!

Reliability: Our Devices Deployed at the Highest Levels

Example: Safety Integrity Level (SIL) - IEC 61508

SIL Level	Availability	Probability of Failure	FIT Rate	Consequence	Application Example
4	>99.99%	1 failure in 110,000 yrs	~1	Potential for fatalities in the community	Nuclear Power Plant Control
3	99.9%	1 failure in 11,100 yrs	~10	Potential for multiple on-site fatalities	Hazardous area laser curtain sensors
2	99-99%	1 failure in 1,100 yrs	~100	Potential for major on-site injuries or fatalities	Hazardous liquid flow meter
1	90-90%	1 failure in 110 yrs	~1000	Potential for minor on-site injuries	Thermal Meter

Example: Design Assurance Level (DAL) – DO 254

DAL Level	Occurrence/hour	Classification	Mitigation	Application Examples
A	10^{-9} per flight hour	Catastrophic	Need Redundancy and Dissimilar Technologies	Primary flight controls, navigation
B	10^{-7} per flight hour	Hazardous	Need Redundancy	Secondary flight controls
C	10^{-5} per flight hour	Major	May Need Redundancy	Backup systems
D	10^{-3} per flight hour	Minor	No Redundancy	Announcement systems, maps
E	N/A	No Effect	No Redundancy	In flight entertainment

Reliability:
Safety Standards Drive New Requirements

Industrial (Generic)	• IEC61508 • Functional safety of electrical, electronic, programmable electronic safety related systems
Process Industry	• IEC 61511 • Safety instrumented systems
Medical	• IEC 62304 • Software for medical devices
Machines	• ISO 13849/ IEC 62601 • Safety of machines
Aviation	• DO254, DO178B • Software considerations in airborne systems
Railway	• EN 50128 • Railway applications software
Nuclear Power	• IEC 60880 • Control technology, software aspects
Gas Measure Techniques	• EN50271/ EN 50402 • Functional safety gas warning systems

- Sample Applications
  - Emergency shut-down systems
  - Fire and gas systems
  - Turbine control
  - Defibrillator
  - Railway signaling systems

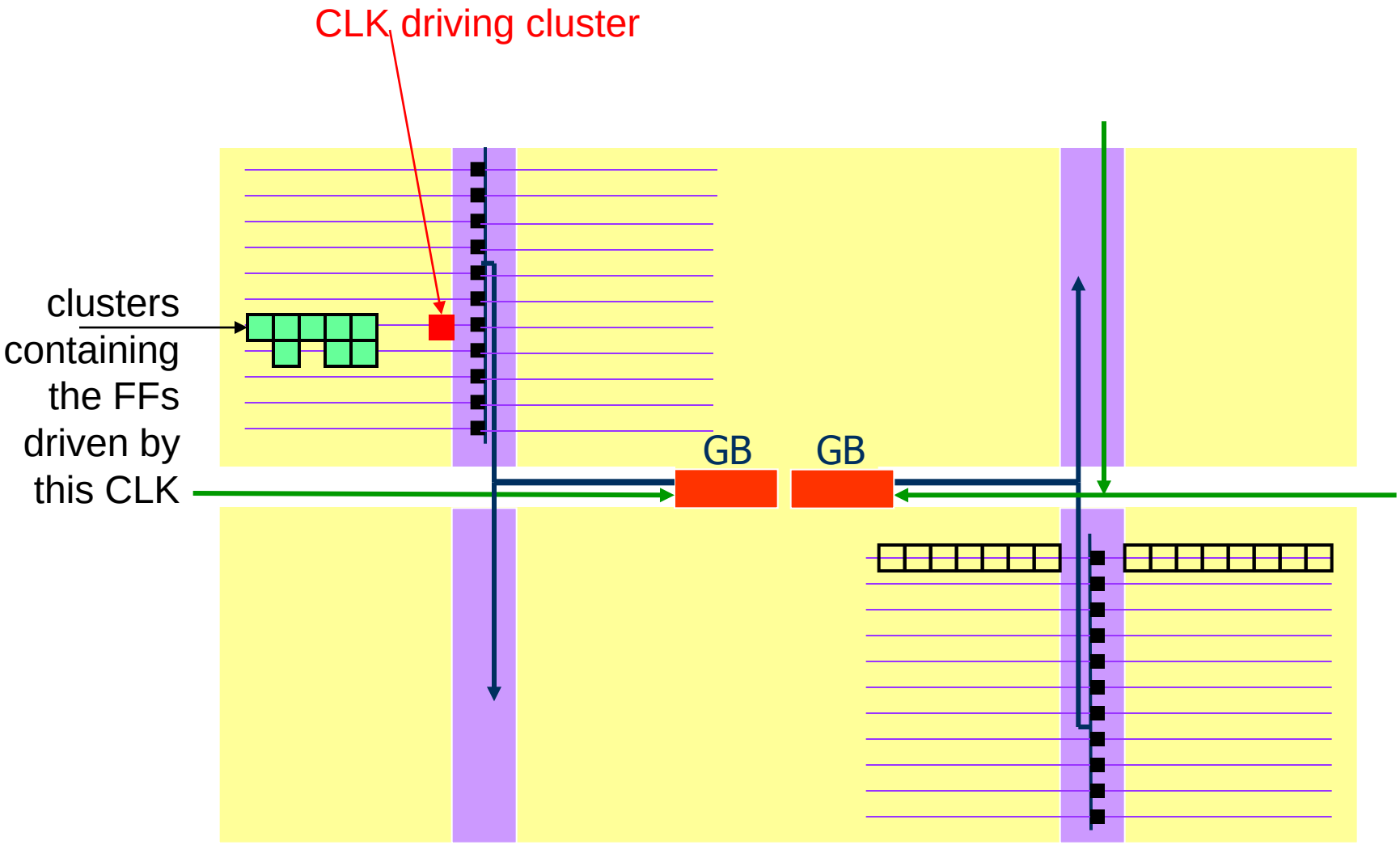




Q&A on Small Local Clocks (1):

- What is the best routing option for a small clock (say fanout=24 only)?
  - If this CLK is generated in the FPGA, place the generating BLE as close to a vertical stripe.
  - Place all its fanout (the FFs it drives) on a quarter-row.
    - If the fanout is higher (say 200) you can use multiple quarter rows.

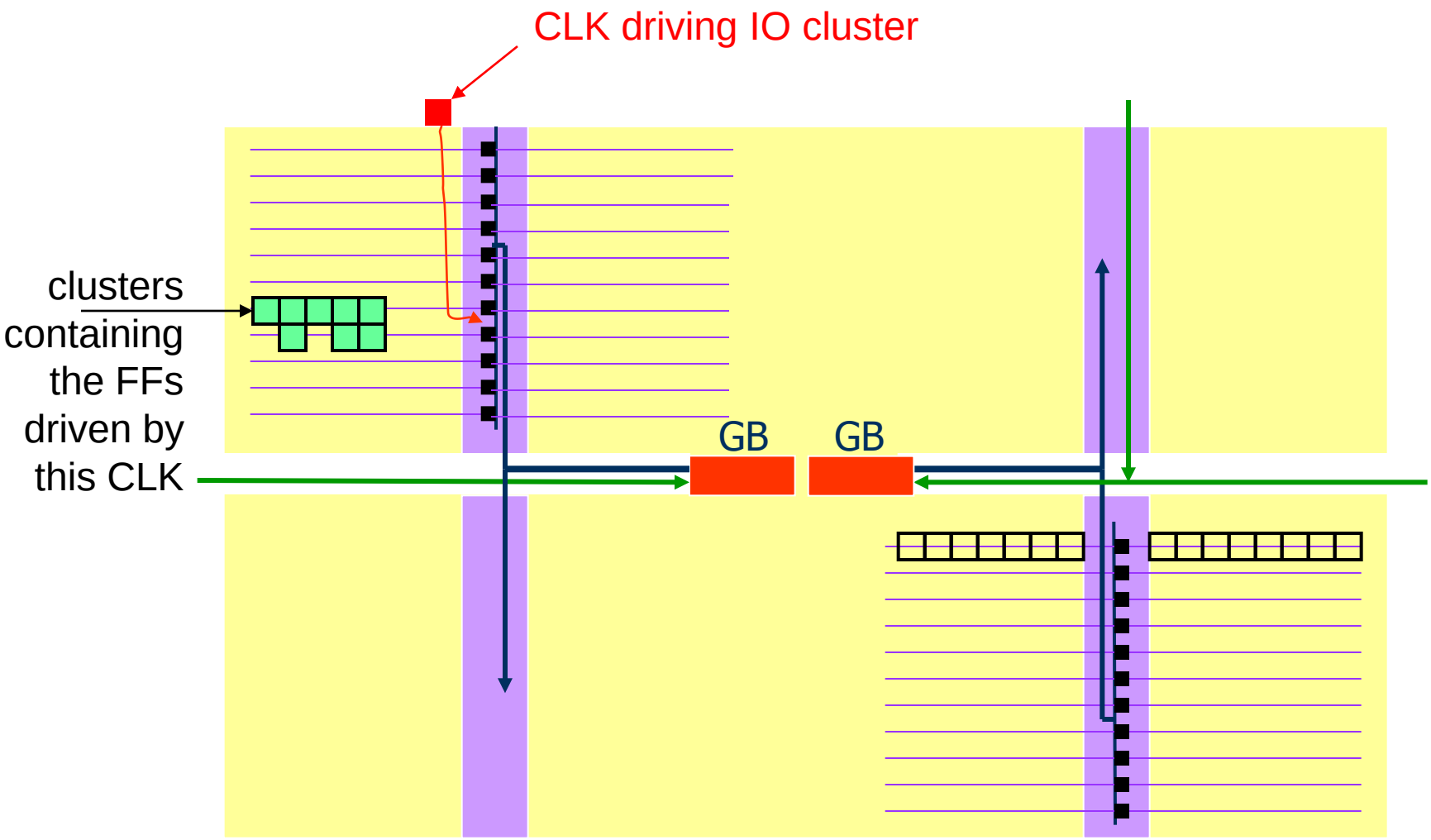
Placing a Locally Generated Small Low-skew CLK:



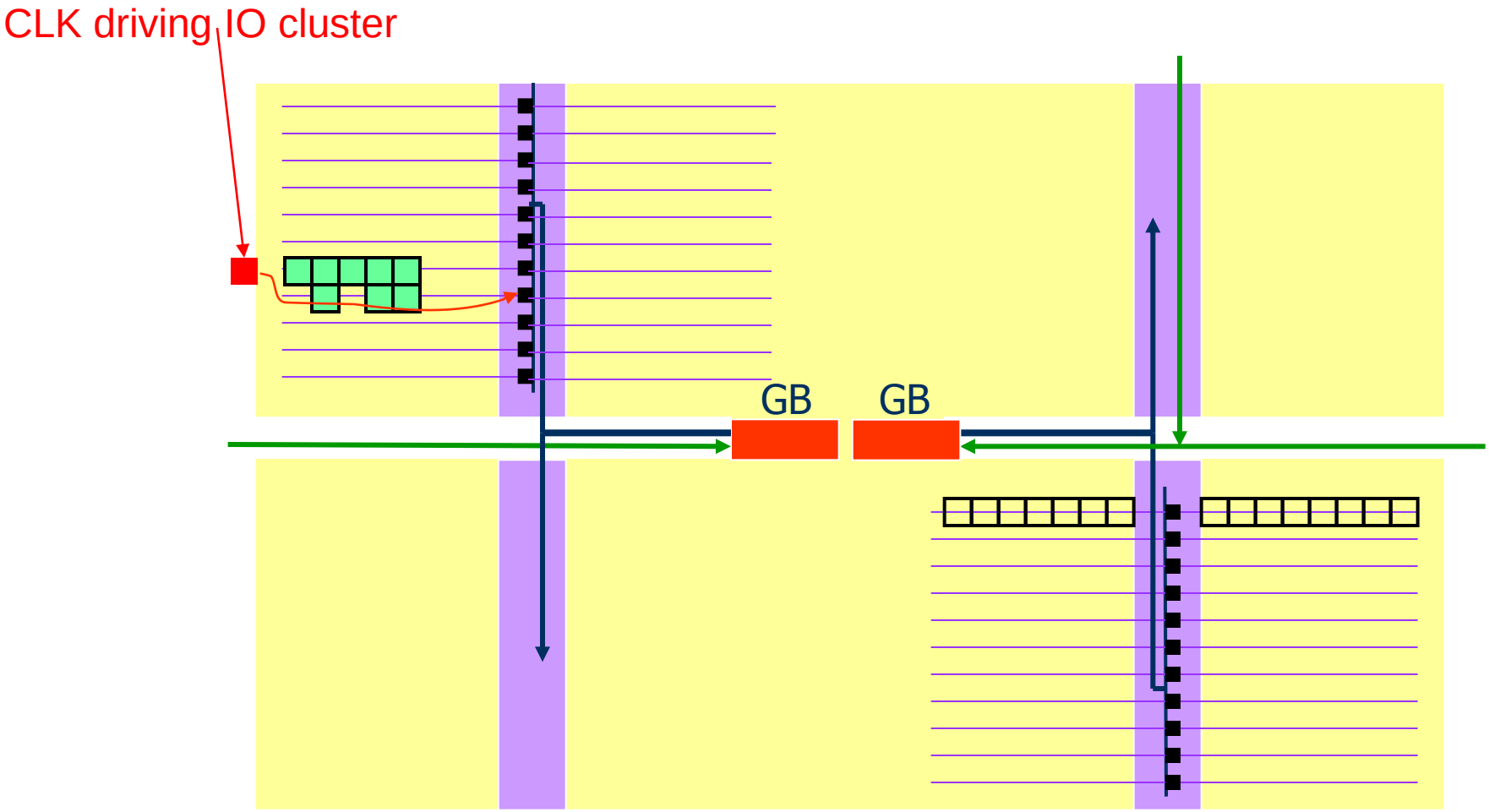
Q&A on Small Local Clocks (2):

- What if this small clock is not generated on chip, but it comes directly from a PAD?
  - In this case choose a PAD location near a stripe at the top or bottom.
  - The router will route the signal to the quarter row global in the best possible way.

Placing an external small low-skew CLK:



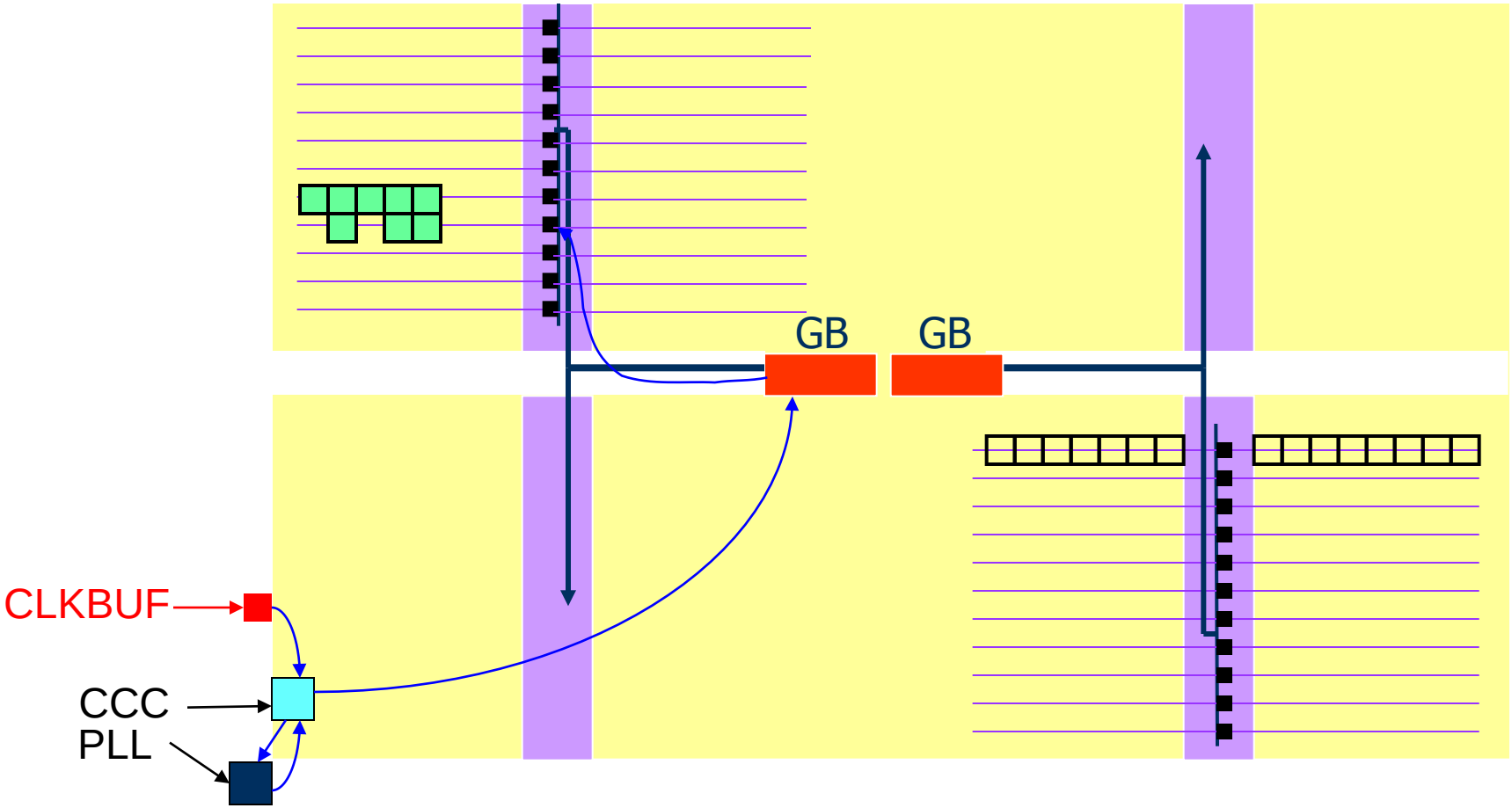
An alternate placement for the IO:



Q&A on Small Local Clocks (3):

- **What if this small clock needs a PLL?**
- Any Signal that goes through a PLL must go through GB. It cannot be redirected to an RGB bypassing GB.
  - In this case the solution may become expensive whether or not the signal comes from an IO.
  - We now have to burn one of the 16 global signals for this CLK.
  - Assuming this is not a problem (meaning we have plenty of global signals for our needs) then all is well. Otherwise we may have to move some other (larger) CLK off the global network.

Driving a small CLK needing a PLL:

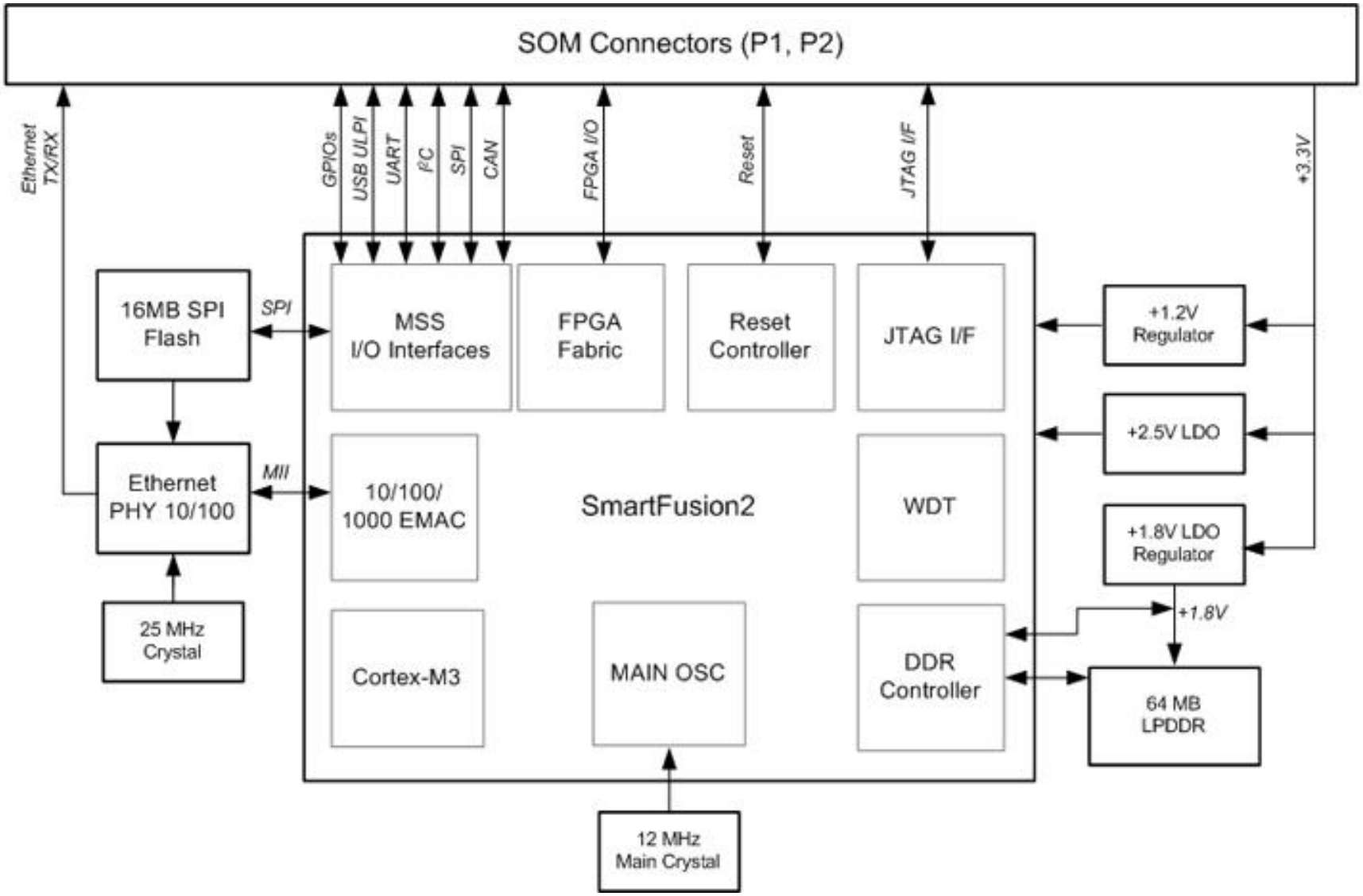


Q&A on Non-Clk signals:

- What is the penalty for using the global network for other signals like ENABLE or RESET, or even a large combinatorial net?
- There is no penalty if:
  1. The signal in question is not critical.
  2. We have plenty of global resources for all CLKs (at least for all CLKs that need PLLs.)
- In fact it is advantageous to route non-critical signals with very high fanout on the global network: this decreases congestion for the entire design and improves performance and routability both.

SF2-Starter-Kit

SmartFusion2 System-On-Module (SOM) Block Diagram



SmartFusion2 Starter Kit
SOM Baseboard Block Diagram