Revision 1.1

CONFIDENTIAL

QuickLogic® S3AI™
Sensor & Audio Processing Platform
Data Sheet

Multi-Core, Ultra Low Power Sensor & Audio Processing Platform Enabling Always-On, Always-Aware Application

Multi-Core Design Platform

- An ARM® Cortex® M4F floating point processor for general purpose and AI processing with AI acceleration from on-chip FPGA and Feature Extraction from Ultra-low power µDSP-like Flexible Fusion Engine (FFE) in a single device
- Multiple, concurrent cores enable algorithm partitioning to achieve the most power and computationally efficient sensor processing system-on-a-chip in the market

Cortex M4F Processor

- Up to 80 MHz operating frequency
- Up to 512 KB SRAM with multiple power modes, including deep sleep (128 KB of this memory can be used for sensor batching)
- Ideal for computationally intensive AI analytics algorithms & neural network inference with balanced matrix among performance, memory footprint, power consumption and latency in AI@Endpoint Applications.

AI-Engine Neurons Support

- FPGA Accelerated RBF (Radical Basis Function) soft-neurons
- FFE (µDSP) Feature Extractions support

3rd Gen, Flexible Fusion Engine

- Up to 10 MHz operating frequency
- 50 KB control memory
- 16 KB data memory
- µDSP-like architecture for efficient mathematical computations
- Ideal for AI Feature Extraction as well as always-on, real-time sensor data analytics

Sensor Manager

- 1.5 KB x 18-bit memory
- Completely autonomous (zero load on M4F) initialization and sampling of sensors through hard-wire I²C or configurable I²C/SPI interface
- Dramatically lowers the power consumption associated with sensor data acquisition

Communication Manager

- Communicates with host applications processor through SPI Slave interface of up to 20 MHz

Dedicated Audio Support

- Audio support for PDM or I²S microphones
- Dedicated logic for Pulse Density Modulation (PDM) to Pulse Code Modulation (PCM) conversion
- Dedicated hard logic integration of Sensory Low Power Sound Detect (LPSD) allows system to sleep in quiet environments

On-Chip Programmable Logic

- 2,400 effective logic cells with 64 Kbit RAM available
- Eight RAM FIFO controllers
- Provides capability to add logic functions or augment existing logic functions

OS & Tool Support

- Real-Time Operating System (RTOS): FreeRTOS
- SensiML™ Analytics Toolkits

Preliminary

Table of Contents

Contents

Product Brief....2
Table of Contents....3
1 Introduction....12
2 Functional Overview....13
2.1 EOS S3AI Processing Platform Architecture....13
2.2 M4F Processor....17
2.2.1 M4F Subsystem Overview....17
2.2.2 System-Level Interface....19
2.2.3 Integrated Configurable Debug....19
2.2.4 M4F and Core Peripherals....20
2.2.5 Embedded SRAM....22
2.2.6 Development support (Serial Wire Interface)....22
2.2.6.1 Debugger Configuration....22
2.2.6.2 Debugger Bootstrap Configurations....23
2.2.7 Overview....24
2.2.8 Flexible Fusion Engine....26
2.2.8.1.1 µDSP-Like Processor....27
2.2.8.1.2 Instruction Memory....27
2.2.8.1.3 Data Memory....27
2.2.9 Sensor Manager....28
2.2.9.1 Overview....28
2.2.9.1.1 Microcontroller Unit....29
2.2.9.1.2 Instruction and Data Memory....29
2.2.10 I^{2}C Master....30
2.2.10.1 System Configuration....31
2.2.10.2 I^{2}C Protocol....31
2.2.10.2.1 START Signal....32
2.2.10.2.2 Slave Address Transfer....32
2.2.10.2.3 Data Transfer....33
2.2.10.2.4 STOP Signal....33
2.2.10.2.5 Arbitration....33
2.2.10.3 I^{2}C Core Architecture....33
2.2.10.3.1 Clock Generator....35
2.2.10.3.2 Byte Command Controller....35
2.2.10.3.3 Bit Command Controller....35
2.2.10.3.4 Data I/O Shift Register....36
2.2.11 Serial Peripheral Interface (SPI)....37

Preliminary

2.2.11.1 SPI Master – System Support 37
2.2.11.1.1 Features 38
2.2.11.1.2 Configuration Logic 40
2.2.11.2 SPI Master – Sensor Processing Subsystem Support 41
2.2.11.3 SPI Slave 42
2.2.11.4 SPI Interface Protocol 43
2.2.11.4.1 Basic Read/Write Transfers 44
2.2.11.4.2 Device ID Read 45
2.2.11.4.3 Transfer Types 46
2.2.11.4.3.1 Transfers to TLC Local Registers 46
2.2.11.4.3.2 Transfers from Packet FIFOs 46
2.2.11.4.3.3 Transfers to M4F Memory Address Space 47
2.2.11.4.3.3.1 Basic AHB Transfer Restrictions 47
2.2.11.4.4 SPI Write Cycle 50
2.2.11.4.5 SPI Read Cycle 50
2.2.11.4.6 SPI Multiple Read Cycle 51
2.2.11.4.7 SPI 3 wire configuration 51
2.2.11.4.8 SPI Corner Cases 52
2.2.11.4.9 Transmission Format 53
2.2.11.4.10 Clock Phase and Polarity Controls 54
2.2.11.4.10.1 CPHA = 0 Transfer Format 55
2.2.11.4.10.2 CPHA = 1 Transfer Format 56
2.2.12 AHB Master Bridge 58
2.2.13 Control Registers 58
2.2.14 Packet FIFO 59
2.2.15 On-Chip Programmable Logic 59
2.3 Audio Subsystem 60
2.3.1 PDM Microphone 60
2.3.2 I2S Microphones 61
2.3.3 Low Power Sound Detect (LPSD) Support 61
2.3.4 PDM Slave Port for External Codec 61
2.3.5 DMA and AHB Master Port 61
2.3.6 APB Slave Port 61
2.3.1 I^{2}S Slave Port 62
2.4 Timing 63
2.4.1 I2C Master AC Timing 63
2.4.2 I2S Timing 64
2.4.3 PDM Microphone Timing 65
2.4.4 SPI Timing 66
2.4.4.1 SPI Master 66
2.4.4.2 SPI Slave 67
2.5 On-Chip Programmable Logic 68
2.5.1 Functional Description 68
2.5.1.1 Logic Cell 68

Preliminary

2.5.1.2 RAM/FIFO 69
2.5.1.3 FIFO Controller 70
2.5.1.4 Configurable Input/Output Signals 71
2.5.1.5 Multipliers 72
2.5.2 Interface to the On-chip programmable logic 73
2.5.3 EOS S3AI Platform Interface 73
2.5.3.1 AHB-To-Wishbone Bridge 73
2.5.3.2 SDMA Interface 74
2.5.3.3 Interrupt Interface 75
2.5.3.4 Sensor Processing Subsystem Interface 75
2.5.3.5 Packet FIFO Interface 75
2.6 Power Management 76
2.6.1 Power Supply Modes and Schemes 76
2.6.1.1 SRAM Power domains 77
2.6.1.2 Low Drop-out Regulators 78
2.6.1.2.1 LDO Use Case 1: Dual Voltage Rail Supplied by On-chip LDOs 79
2.6.1.2.2 LDO Use Case 2: Single Voltage Rail Supplied by Single On-chip LDOs 80
2.6.1.2.3 Use Case 3: External Voltage Supplied 81
2.6.2 Power-On Sequence 82
2.6.1 Power-Down Sequence 86
2.6.2 Clocks and Resets 88
2.6.2.1 Clocks 88
2.6.2.2 Resets 91
2.7 Other EOS S3AI Platform Features 92
2.7.1 Multi-Function Inputs/Outputs (IOs) 92
2.7.1 General Purpose Inputs/Outputs (GPIOs) 92
2.7.1 Programmable Logic Inputs/Outputs (GPIOs) 92
2.7.2 Interrupts 93
2.7.2.1 Interrupt Structure 93
2.7.2.2 Interrupt Sources 93
- TOP Interrupts 93
- M4F Subsystem Interrupts 93
- FFE Interrupts 94
- A0 Interrupts 94
- A1 Interrupts 94
- Audio Interrupts 94
- SDMA Interrupts 94
- PKFB Interrupts 94

2.7.2.3 M4F Wake-Up Events 95
2.7.3 Bootstrap Modes 96
2.7.3.1 M4F Serial Wire Debug Port Configuration 96
2.7.3.2 Internal/External HSO Configuration 96
2.7.3.1 SWD Debugger Present Configuration 97
2.7.3.2 AP/Wearable Mode Configuration 97
Preliminary

2.8 Other Peripherals 98
2.8.1 Packet FIFO 98
2.8.1.1 FIFO_8K 100
2.8.1.2 FIFO_0, FIFO_1, FIFO_2 100
2.8.1 System DMA 101
2.8.1.1 Function Description 102
2.8.1.2 SDMA Configurations 103
2.8.2 Analog to digital converter (ADC) 104
2.8.2.1 Overview 104
2.8.2.2 Functional Description 104
2.8.2.3 Electrical Characteristics 104
2.8.2.4 PCB Layout Recommendations 105
2.8.2.5 Example Application 105
2.8.3 Universal Asynchronous Receiver Transmitter (UART) 106
2.8.4 Timer and Counters 107
2.8.4.1 1ms Event Counter 108
2.8.4.2 Error Correction for 1mS Event Counter 108
2.8.4.3 Timeout Event Counter 109
2.8.4.4 30 Bits Counter 109
2.8.4.5 Time Stamp Counters 109
2.8.4.6 PMU and FFE Wakeup 109
3 Device Characteristics 111
3.1 Pinout and Pin Description 111
3.1.1 I/O State 115
3.2 Electrical Specifications 117
3.2.1 DC Characteristics 117
3.2.2 Output Drive Current 119
3.2.3 Clock and Oscillator Characteristics 120
3.2.4 Output Rise/Fall Time 121
3.2.4.1 Output Rise/Fall Time (DVDD = 1.8V) 121
3.2.4.2 Output Rise/Fall Time (DVDD = 2.5V) 122
3.2.4.3 Output Rise/Fall Time (DVDD = 3.3V) 122
3.2.5 Power Consumption 123
4 Application Examples 125
Real-Time Operating System IOT Design 125
5 Package Information 126
5.1 42-Ball WLCSP Package Information 126
5.2 64-Ball BGA Package Information 127
5.3 Soldering Information 128
5.3.1 Reflow Profile (Preliminary) 128
5.4 Package Thermal Characteristics 130
6 Contact Information 131
7 Revision History 132
8 Notice of Disclaimer 133
Preliminary

9 Copyright and Trademark Information 134
Preliminary

List of Figures

Figure 1: EOS S3AI Processing Platform Architecture 13
Figure 2: EOS S3AI Sensor Processing Platform Block Diagram 16
Figure 3: Cortex M4F Schematic Block Diagram 18
Figure 4: Recommended External Debugger Connection for ARM DS Debugger 23
Figure 5: Sensor Processing Subsystem Block Diagram 24
Figure 6: Flexible Fusion Engine Block Diagram 26
Figure 7: Sensor Manager Block Diagram 28
Figure 8: I2C Modules within the EOS S3AI platform 30
Figure 9: I^2C Protocol Example 32
Figure 10: I2C Block Diagram 34
Figure 11: I2C Bit Command Sequences 36
Figure 12: SPI Master - System Support Block Diagram 37
Figure 13: SPI Interface used for Sensor Processing Subsystem Support 41
Figure 14: SPI Slave Block Diagram 43
Figure 15: SPI Slave Protocol Diagram 45
Figure 16: SPI Slave ID Read Protocol Diagram 45
Figure 17: SPI Master Burst Write Sequence 47
Figure 18: Example Burst Read Sequence 49
Figure 19: Basic SPI Write Operation (mode 11) 50
Figure 20: Basic SPI Read Operation (mode 11) 50
Figure 21: SPI Multiple Read Operation (mode 11) 51
Figure 22: 3-Wire Basic SPI Read/Write Sequence (mode 11) 51
Figure 23: muRata Command and 11-bit SPI Acceleration Data Read Sequence 52
Figure 24: AD7091 SPI Transfer Sequence 52
Figure 25: SPI Block Diagram 53
Figure 26: SPI Clock Format 0 (CPHA = 0) 56
Figure 27: SPI Clock Format 1 (CPHA = 1) 57
Figure 28: FFE's AHB Master Bridge Block Diagram 58
Figure 29: Audio Subsystem Block Diagram 60
Figure 30: I^2S Slave Port 62
Figure 31: I^2C Master AC Timing 63
Figure 32: I^2S Timing Waveform 64
Figure 33: PDM Microphone Timing 65
Figure 34: SPI Master AC Timing 66
Figure 35: SPI Slave Timing 67
Figure 36: Logic Cell block diagram 69
Figure 37: On-chip programmable logic Configurable Input/Output 71
Figure 38: On-chip programmable logic Multiplier 72
Figure 39: AHB-To-Wishbone Bridge 74
Figure 40: LDO User Case 1 - Dual Voltage Rail 79
Figure 41: LDO Use Case 2 - Single Voltage Rail 80
Figure 42: LDO Use Case 3 - External Voltage Supplied 81
Figure 43: Power-On Sequencing 82
Figure 44: Current Measurement Scheme with External Power Supplies 84

Preliminary

Figure 45: Power-Down Sequencing 86
Figure 46: Clock Tree 90
Figure 47: Bootstrap Timing 96
Figure 48: PKFB Block Diagram 99
Figure 49: System DMA Block Diagram 101
Figure 50: System DMA Interface 102
Figure 51: ADC Block Diagram 104
Figure 52: Example Voltage Divider Circuit 105
Figure 53: Timer Block Diagram 107
Figure 54: 1ms Count and 1ms Counter Relationship 109
Figure 55: PMU and FFE Timing Waveform 110
Figure 57: Example Real-Time Operating System Wearable Design Block Diagram 125
Figure 58: 42-Ball WLCSP Package Drawing 126
Figure 59: 64-Ball BGA Package Drawing 127
Figure 60: Pb-Free Component Preconditioning Reflow Profile 128

Preliminary

List of Tables

Table 1: EOS S3AI Platform Supported Features ... 14
Table 2: I2C Master AC Timing ... 63
Table 3: I^{2}S Timing ... 64
Table 4: PDM Microphone Timing ... 65
Table 5: SPI Master Timing ... 66
Table 6: SPI Slave Timing ... 67
Table 7: on-chip programmable logic major features ... 68
Table 8: Power Domains ... 77
Table 9: Memory Domains ... 78
Table 10: LDO Regulators ... 78
Table 11: Power-On Sequencing Timing Parameters ... 83
Table 12: Current Measurements for Power-On Sequence[a] ... 85
Table 13: Current Measurements for Power-On Sequence[a] ... 85
Table 14: Power-Down Sequencing Timing Parameters ... 86
Table 15: Current Measurement for Power-Down [a] ... 87
Table 16: LDO Mode Typical Inrush Current ... 87
Table 17: LDO Bypass Mode Typical Inrush Current ... 87
Table 18: Maximum Supply Power Consumption ... 87
Table 19: Clocks Listing ... 88
Table 20: Bootstrap Timing during Power-On Sequence ... 96
Table 21: M4F Serial Wire Debug Port Bootstrap Configuration ... 96
Table 22: Internal/External HSO Configuration ... 96
Table 23: SWD Debugger Present Configuration ... 97
Table 24: AP/Wearable Mode Configuration ... 97
Table 25: Packet FIFO Instances ... 99
Table 26: SDMA channel assignment ... 103
Table 27: ADC Electrical Characteristics ... 104
Table 28: I/O Locations and Functions ... 112
Table 29: I/O State ... 115
Table 30: Absolute Maximum Ratings ... 117
Table 31: Recommended Operating Range ... 117
Table 32: Weak Pull-Up/Pull-Down Characteristics ... 118
Table 33: DC Input and Output Levels[a] ... 118
Table 34: Output Drive Current (DVDD = 1.8V) in mA ... 119
Table 35: Output Drive Current (DVDD = 2.5V) in mA ... 119
Table 36: Output Drive Current (DVDD = 3.3V) in mA ... 120
Table 37: Clock and Oscillator Characteristics[a] ... 120
Table 38: Output Rise/Fall Time (SR = 1, DVDD = 1.8V) ... 121
Table 39: Output Rise/Fall Time (SR = 0, DVDD = 1.8V) ... 121
Table 40: Output Rise/Fall Time (SR = 1, DVDD = 2.5V) ... 122
Table 41: Output Rise/Fall Time (SR = 0, DVDD = 2.5V) ... 122
Table 42: Output Rise/Fall Time (SR = 1, DVDD = 3.3V) ... 122
Table 43: Output Rise/Fall Time (SR = 0, DVDD = 3.3V) ... 123

Preliminary

Table 44: Shutdown Current [a] 123
Table 45: Standby Current [a] 124
Table 46: CoreMark Current Measurement 124
Table 47: EOS S3AI Power Measurements 124
Table 48: Pb-Free Component Preconditioning Reflow Profile[a,b] 129
Table 49: Package Thermal Characteristics 130

Preliminary

1 Introduction

The S3AI platform is a multi-core, ultra-low power AI processing system designed for AI@Endpoint applications such as Internet of Things (IoT) and wearable devices. The core of the QuickLogic S3AI platform is its proprietary µDSP-like Flexible Fusion Engine (FFE). To complement the FFE, the S3AI platform also includes a Cortex M4F subsystem that enables higher level, general purpose processing.

This multi-core architecture offers the unique ability to enable higher performance for FPGA accelerated RBF neurons with low power. With AI-enabled knowledge and know-how from the SensiML Analytics Toolkits and the Acceleration from on-chip eFPGA and Feature Extraction from FFE, S3AI enables not only off-loading feature extraction functions giving more processing power for classifications and algorithms, but also real-time, always-on sensor computational capability. The multi-core approach and multiple power islands allow the S3AI platform to acquire, process sensor data with Smart AI data analytic algorithms, inference and selectively data exchange in the most efficient way with balanced performance, memory footprint, power consumption and latency in AI@Endpoint applications.

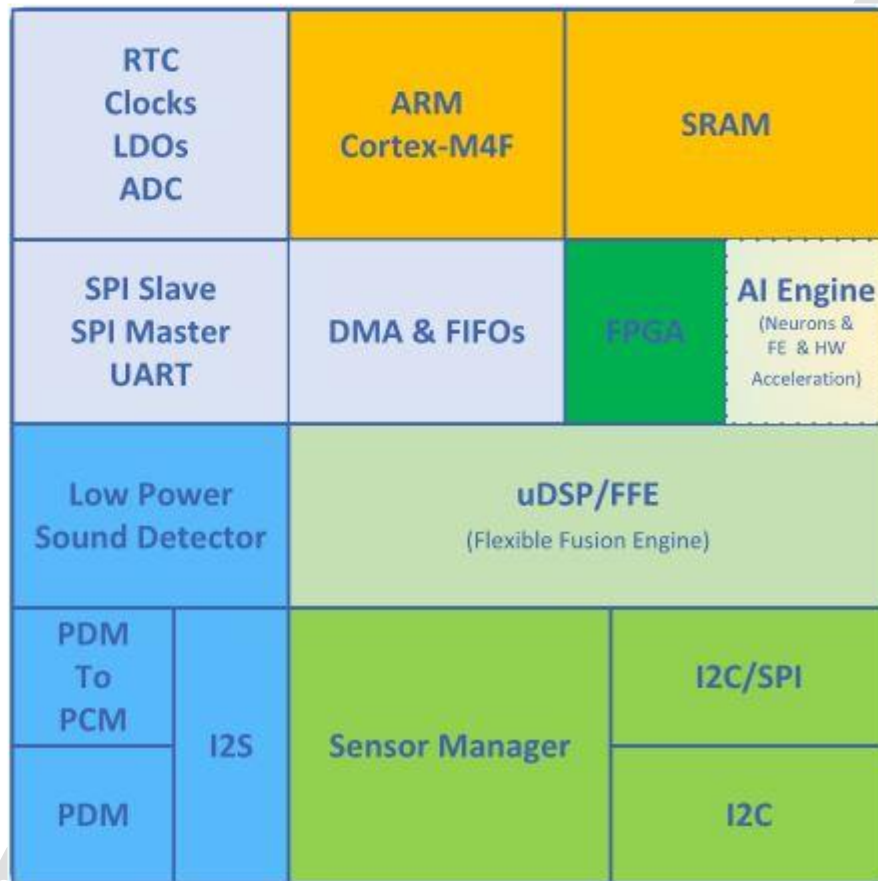
Preliminary

2 Functional Overview

2.1 EOS S3AI Processing Platform Architecture

Figure 1 shows a system level architecture diagram that highlights the major functional blocks of the EOS S3AI processing platform. More detailed block diagrams are provided later in this document.

Figure 1: EOS S3AI Processing Platform Architecture



Preliminary

Table 1 lists the top-level features. This feature set enables the EOS S3AI platform to support use cases in the IoT and wearable device markets.

Table 1: EOS S3AI Platform Supported Features

Feature	Details
M4F Sub-System	• Cortex M4F controller with floating point unit support (M4F) • Embedded SRAM (up to 512 KB) for code and data memory • Vectored interrupt support • Wakeup interrupt controller • 2-pin SWD port
Flexible Fusion Engine	• 50 KB control memory • 16 KB data memory • Single cycle MAC
Digital Microphone Support	• I2S microphone • PDM microphone • On-chip PDM-to-PCM conversion • Integrated Low Power Sound Detector from Sensory, Inc.
Packet FIFOs Batching Memory	• 128 KB of M4F SRAM can be used as sensor batching memory • Multiple packet FIFOs to support FFE to M4F/application processor data transfers: - 8 KB packet FIFO with ring-buffer mode support - 256 x 32 packet FIFO and two 128 x 32 packet FIFOs
Power Management Unit	• Low-power mode with fast wake-up • Programmable power modes (deep sleep, sleep with retention, and active) • Multiple power domains • Power sequencing for sleep and wake-up entry and exit • Software and hardware-initiated sleep entry • Wake-up triggers via internal and external events • Internal LDO support
Programmable On-chip programmable logic	• 2,400 effective logic cells with 64 Kbit of RAM, 8 RAM FIFO controllers and 2 GPIO banks • Supports SPI slave configuration • Supports five clocks
AI-Engine	• FPGA Accelerated RBF Neurons • Support FPGA Hardware Acceleration and FFE (µDSP) based Feature Extractions
32 kHz Oscillator with Real-Time Clock (RTC)	• 32 kHz crystal oscillator (external crystal required) with bypass option • 1 Hz clock generation with compensation register • RTC function with one alarm register • Start time of 350µs

Preliminary

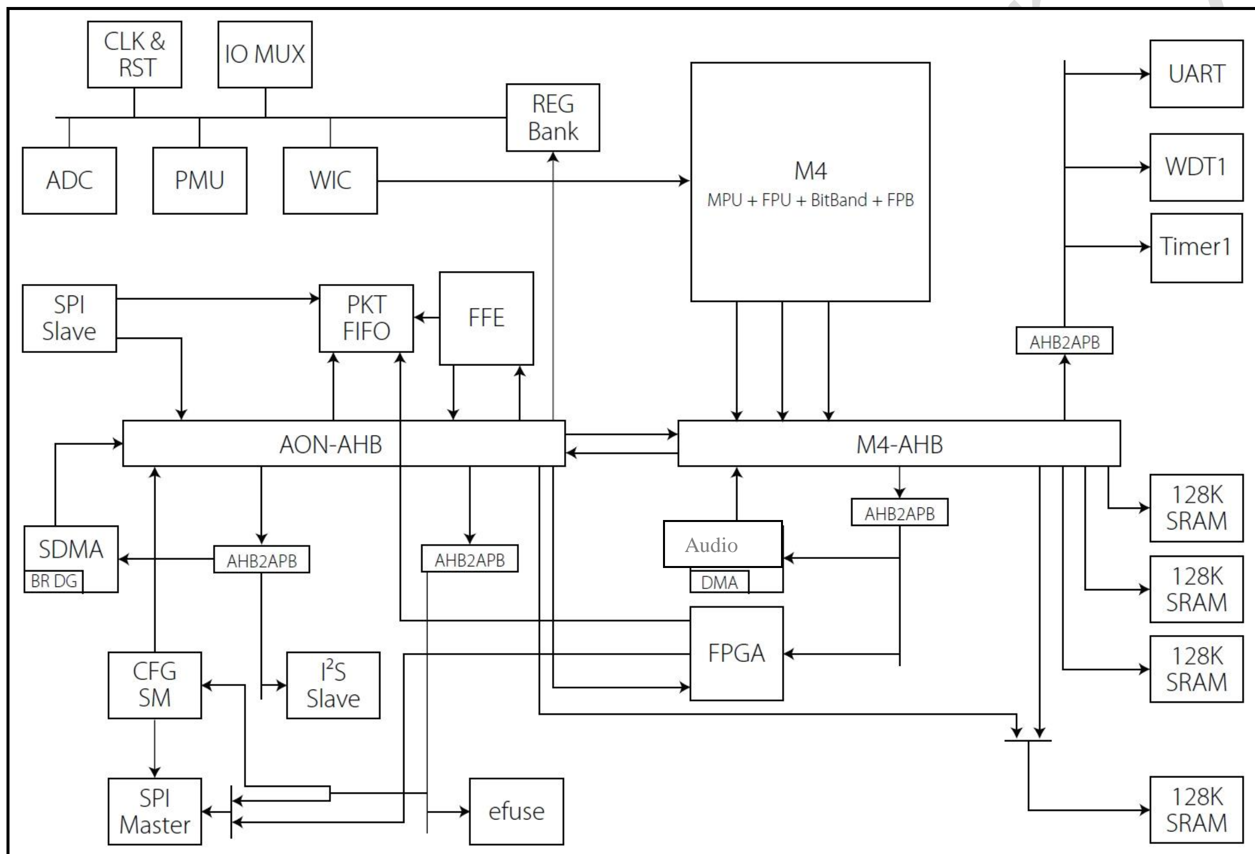
Feature	Details
High Frequency Clock Source	• Programmable frequency (2 MHz to 80 MHz) for better frequency resolution • Calibrated output (using 32 kHz input) • Startup time of 410µs • Clock divider can be programmed in 12 bits
System DMA	• 16 channel DMA allows efficient data movement between processing Elements
SPI Slave	• SPI slave application processor communication of up to 20 MHz
Time Stamping	• Automatic hardware time stamp on every sensor read in the interrupt mode • Up to eight sensor interrupt captured time-stamps (8-bit) • Main time stamp of 24-bits • Resolution of 1ms
I2C Master and Configurable I2C/SPI Interface	• I2C master and SPI master with programmable clock pre-scalar • Option to disable multi-master support and slave-inserted wait for shorter SCL cycles • Configurable for two I2C Masters or one I2C Master and one SPI Master
Other Interfaces	• SPI master for interfacing with serial flash memories and other external SPI-based peripherals of up to 20 MHz • I2S Slave Transmitter for downloading audio samples to Host Application Processor
UART	• Serial support for M4F debug and code development • Communication with UART-based external peripherals
Other Peripherals	• Timers • Watchdogs • GPIO controllers
ADC	• Low sampling rate ΔΣ 12-bit
LDOs	• On-chip LDO for system logic • Separate on-board LDO for memory
eFuse	• On-chip eFuse for storing fuse data for in-chip performance tuning and security key bits
Integrated Software Debug Interface	• 2-pin SWD port for access to the following memory mapped resources: - M4F internal registers and memories - FFE and Sensor Manager memories - FFE control registers - Programmable Logic memories - Programmable Logic designs through generic AHB bus - All memory map peripherals such as timers, WDT, SPI master, etc. - I2C master used for I2C sensor debug - Multiplexed dedicated parallel debug interface

Preliminary

Feature	Details
Packaging Options	• 42-ball WLCSP (2.656 mm x 2.422 mm x 0.564 mm) (27 user I/O, 2 VCCIO banks) • 64-ball BGA (3.5 mm x 3.5 mm x 0.72mm) (46 user I/O, 2 VCCIO banks)

Figure 2 shows a more detailed view of the data paths within the EOS S3AI audio and sensor processing platform.

Figure 2: EOS S3AI Sensor Processing Platform Block Diagram



Preliminary

2.2 M4F Processor

2.2.1 M4F Subsystem Overview

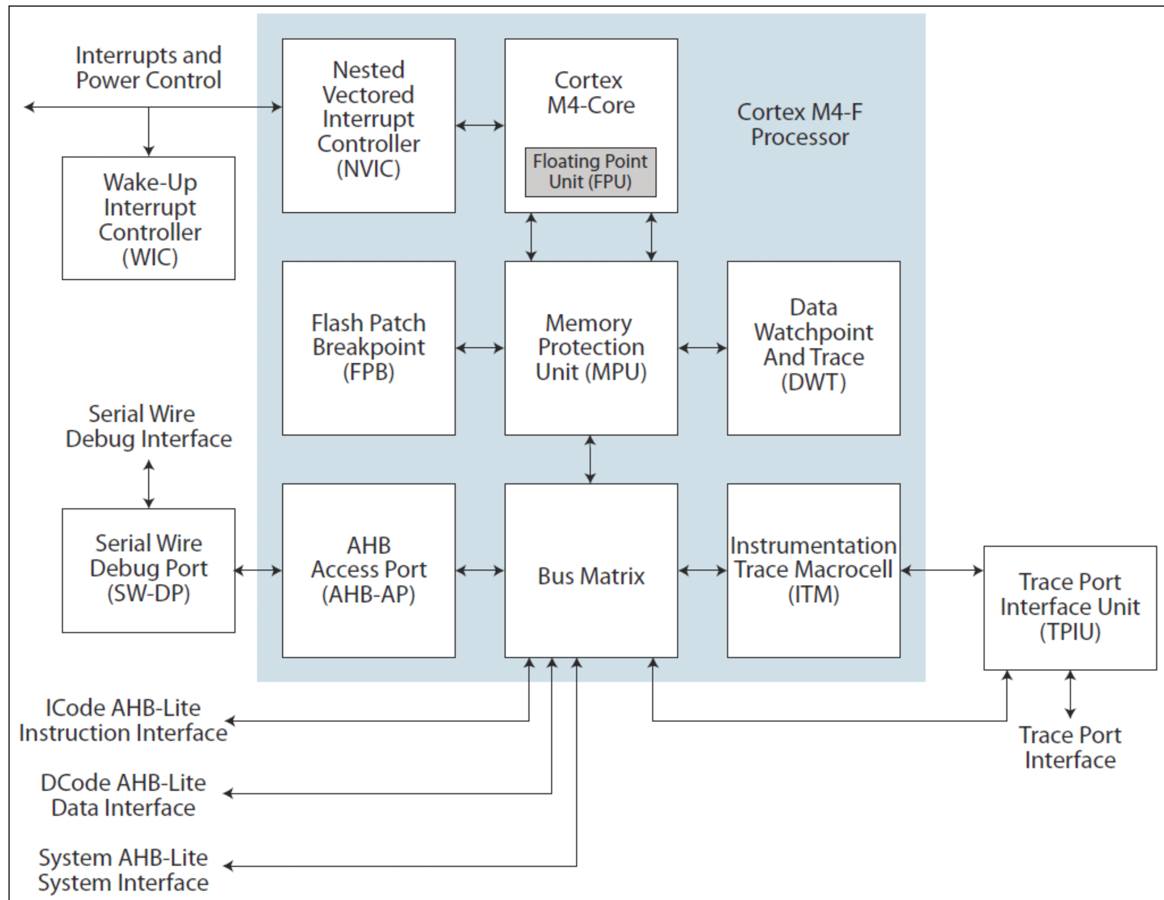
The M4F 32-bit processor subsystem is one of the primary computation blocks of the EOS platform.

The M4F includes:

- Optional features including a Nested Vectored Interrupt Controller (NVIC), flash patch, etc.
- Up to 512 KB SRAM
- AHB bus muxes (AHB bus matrices and AHB slave muxes)
- Multiple asynchronous bridges to interface with the programmable fabric and the FFE subsystem (AHB-to-AHB to the programmable fabric and AHB-to-SRAM to the FFE)
- Peripheral bus incorporating:
  - UART
  - Watchdog Timer
  - Timers

Preliminary

Figure 3: Cortex M4F Schematic Block Diagram



The M4F processor is built on a high-performance processor core, with a 3-stage pipeline Harvard architecture, making it ideal for demanding embedded applications. The processor delivers exceptional power efficiency using an efficient instruction set and extensively optimized design. This combination provides high-end processing hardware that includes optional IEEE754-compliant single-precision floating-point computation, and a range of single-cycle and SIMD multiplication and multiply-with-accumulate capabilities, that ensure saturating arithmetic and dedicated hardware division.

To aid in designing cost-sensitive devices, the M4F processor implements tightly-coupled system components that reduce processor area while significantly improving interrupt handling and system debug capabilities. The M4F processor implements a version of the Thumb®, an instruction set based on Thumb-2 technology, ensuring high code density and reduced program memory requirements. The M4F instruction set provides the exceptional performance expected of a modern 32-bit architecture, with the high code density of 8-bit and 16-bit microcontrollers.

© 2019 QuickLogic Corporation Confidential

The M4F instruction set provides the exceptional performance that is expected of a modern 32-bit architecture, with the high code density of 8-bit and 16-bit microcontrollers. The Cortex M4F processor closely integrates a configurable NVIC, to deliver industry-leading interrupt performance. The NVIC includes a Non-Maskable Interrupt (NMI) that can provide up to 256 interrupt priority levels.

The tight integration of the processor core and NVIC provides fast execution of Interrupt Service Routines (ISRs), dramatically reducing the interrupt latency. This is achieved through the hardware stacking of registers, and the ability to suspend load-multiple and store-multiple operations

Interrupt handlers do not require wrapping in assembler code, removing any code overhead from the ISRs. A tool-chain optimization also significantly reduces the overhead when switching from one ISR to another. To optimize low-power designs, the NVIC integrates with the sleep modes, which includes an optional deep sleep function. This enables the entire device to be rapidly powered down while still retaining program state.

2.2.2 System-Level Interface

The Cortex M4F processor provides multiple interfaces using AMBA® technology to provide high speed, low latency memory accesses. It supports unaligned data accesses and implements atomic bit manipulation that enables faster peripheral controls, system spinlocks and thread-safe Boolean data handling.

The Cortex M4F processor has an MPU that permits control of individual regions in memory, enabling applications to utilize multiple privilege levels, separating and protecting code, data and stack on a task-by-task basis.

2.2.3 Integrated Configurable Debug

The Cortex M4F processor can implement a complete hardware debug solution. This provides high system visibility of the processor and memory through a 2-pin SWD port.

For system trace the processor integrates an Instrumentation Trace Macrocell™ (ITM) alongside data watch points and a profiling unit. To enable simple and cost-effective profiling of the system events these generate, a Serial Wire Viewer (SWV) can export a stream of software-generated messages, data trace, and profiling information through a single pin.

The Breakpoint Unit (FPB) provides up to eight hardware breakpoint comparators that debuggers can use. The comparators in the FPB also provide remap functions of up to eight words of the program in the control memory region.

Preliminary

2.2.4 M4F and Core Peripherals

The Cortex-M4F processor features:

- A low gate count processor core, with low latency interrupt processing that has:
  - A subset of the Thumb instruction set, defined in the ARM®v7-M Architecture Reference Manual.
  - Banked Stack Pointer (SP).
  - Hardware integer divide instructions, SDIV and UDIV.
  - Handler and Thread modes.
  - Thumb and Debug states.
  - Support for interruptible-continued instructions LDM, STM, PUSH, and POP for low interrupt latency.
  - Automatic processor state saving and restoration for low latency Interrupt Service Routine (ISR) entry and exit.
  - Support for ARMv6 big-endian byte-invariant or little-endian accesses.
  - Support for ARMv6 unaligned accesses.

- Floating Point Unit (FPU) providing:
  - IEEE754-compliant operations on single-precision, 32-bit, floating point values.
  - 32-bit instructions for single-precision (C float) data-processing operations.
  - Combined Multiply and Accumulate instructions for increased precision (Fused MAC).
  - Hardware support for conversion, addition, subtraction, multiplication with optional accumulate, division, and square-root.
  - Hardware support for denormals and all IEEE rounding modes.
  - 32 dedicated 32-bit single precision registers, also addressable as 16 double-word registers.
  - Decoupled three stage pipelines.

- Nested Vectored Interrupt Controller (NVIC) closely integrated with the processor core to achieve low latency interrupt processing. Features include:
  - External interrupts, configurable from 1 to 240.
  - Bits of priority, configurable from 3 to 8.
  - Dynamic reprioritization of interrupts.
  - Priority grouping. This enables selection of preempting interrupt levels and non-preempting interrupt levels.
  - Support for tail-chaining and late arrival of interrupts. This enables back-to-back interrupt processing without the overhead of state saving and restoration between interrupts.
  - Processor state automatically saved on interrupt entry, and restored on interrupt exit, with no instruction overhead.
  - Optional Wake-up Interrupt Controller (WIC), providing ultra-low power sleep mode support.

Preliminary

- Memory Protection Unit (MPU). An optional MPU for memory protection, including:
    - Eight memory regions.
    - Sub Region Disable (SRD), enabling efficient use of memory regions.
    - The ability to enable a background region that implements the default memory map attributes.

- Bus interfaces:
    - Three Advanced High-performance Bus-Lite (AHB-Lite) interfaces: I-Code, D-Code, and System bus interfaces.
    - Private Peripheral Bus (PPB) based on Advanced Peripheral Bus (APB) interface.
    - Bit-band support that includes atomic bit-band write and read operations.
    - Memory access alignment.
    - Write buffer for buffering of write data.
    - Exclusive access transfers for multiprocessor systems.

- Low-cost debug solution that features:
    - Debug access to all memory and registers in the system, including access to memory mapped devices, access to internal core registers when the core is halted, and access to debug control registers even while SYSRESETn is asserted.
    - Serial Wire Debug Port (SW-DP) or Serial Wire JTAG Debug Port (SWJ-DP) debug access.
    - Optional Flash Patch and Breakpoint (FPB) unit for implementing breakpoints and code patches.
    - Optional Data Watchpoint and Trace (DWT) unit for implementing watch points, data tracing, and system profiling.
    - Optional Instrumentation Trace Macrocell (ITM) for support of printf() style debugging.
    - Optional Trace Port Interface Unit (TPIU) for bridging to a Trace Port Analyzer (TPA), including Single Wire Output (SWO) mode.
    - Optional Embedded Trace Macrocell (ETM) for instruction trace.

Preliminary

2.2.5 Embedded SRAM

The M4F processor subsystem has up to 512 KB of embedded SRAM that is divided into 4 sub-blocks of 128KB each. Each sub-block is accessible simultaneously via four independent AHB busses. Each 128KB sub-block is further divided down in 4 32KB segments (16 segments in total).

Three of the 128KB blocks (384 KB) of the SRAM memory reside in the processor power domain. The CPU subsystem must be powered on to access this memory space. Each 128 KB sub-block is addressed as four 32 KB segments (12 segments in total). An interrupt can be triggered when any of the 32 KB memory segments are accessed if the memory is in a lower power state (deep sleep or shut down).

The last 128KB SRAM block resides in the Always-on power domain and can be accessed regardless of the state of CPU subsystem power.

The 512 KB SRAM can be accessed by the following AHB masters:

- M4F system bus
- M4F I-code bus
- M4F D-code bus
- Application Processor (AP) through the SPI Slave Interface via the TLC
- Configuration DMA for SPI Flash Controller Master
- Audio DMA
- System DMA
- Flexible Fusion Engine

The SRAM clocks are dynamically controlled. When there is no activity on the memory, the clocks is gated off to ensure lower power consumption.

2.2.6 Development support (Serial Wire Interface)

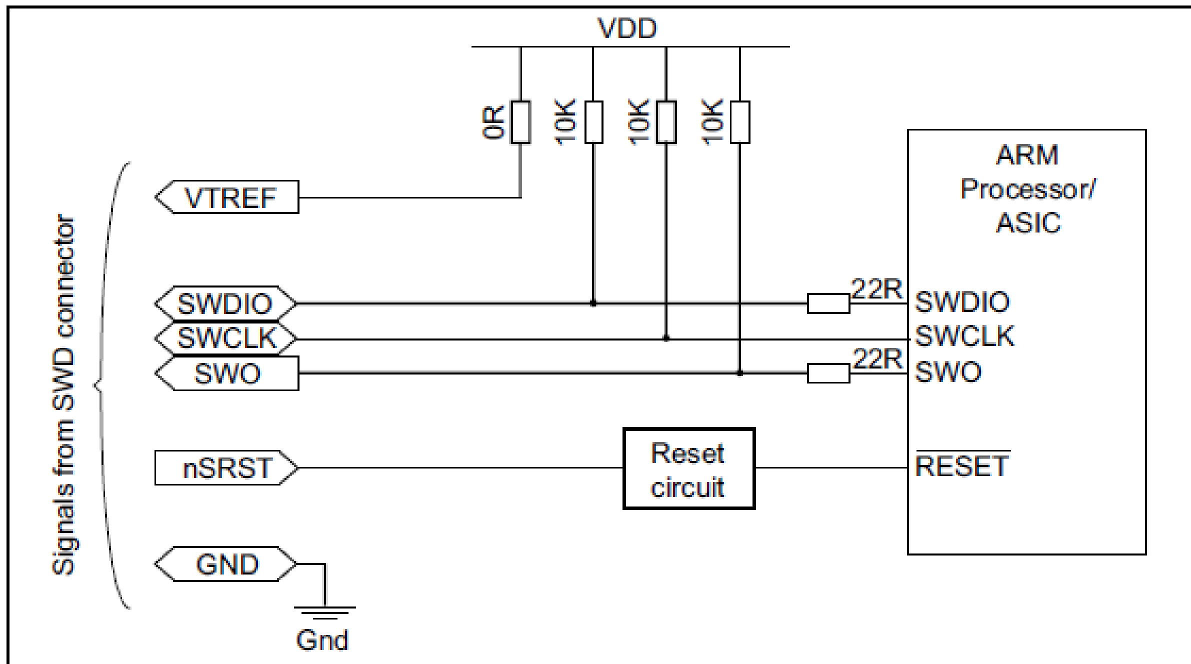
Depending on overall EOS S3AI system setup and requirements, two different Serial Wire interfaces can be selected. The external debugger can be connected to either IO_14/IO_15 or IO_44/IO_45, based on bootstrap pin IO_8 (see Bootstrap Modes section). The two signals are serial wire clock and serial wire data. The optional serial wire viewer can be selected from several different pins.

2.2.6.1 Debugger Configuration

The recommend external configuration if using ARM DS debugger is shown in Figure 4. Other debuggers may have different recommendations.

Preliminary

Figure 4: Recommended External Debugger Connection for ARM DS Debugger



2.2.6.2 Debugger Bootstrap Configurations

Upon cold boot up, the M4F DAP is enabled. The M4F DBGEN is register-enabled by default and can be disabled later if not needed. The M4F DAP will only be reset during cold boot up (it is controlled by the POR from the APC). The release of the M4F reset depends on the state of bootstrap pin IO_19. When it is strapped to high, the M4F reset is released. When strapped to low, the M4F reset release depends on AP `cfg_sm`.

- Standalone Configuration (EOS S3AI platform operating as Host)
  In an IOT design, bootstrap pin IO_19 must always be strapped low to allow M4F operation. Once the boot code is downloaded and the M4F is released from reset, the debugger can be attached to the system.
- Companion Configuration (Application Processor in System)
  In a system with an application processor present, the application processor must drive bootstrap pin IO_19 to indicate whether the Debugger is present. In this configuration, IO_19 is connected to the application processor as part of the SPI interface (the IO_ 19 alternate function is SPIs_MOSI) and it must drive IO_19 during the de-assertion of SYS_RSTn. Driving IO_19 high enables debugger support by releasing the M4- F from system reset immediately. A debugger can take control of the system. Driving IO_19 low allows the system to boot normally, which disallows the debugger access until after M4F is released from reset. Once the M4F is booted, the debugger can be attached.

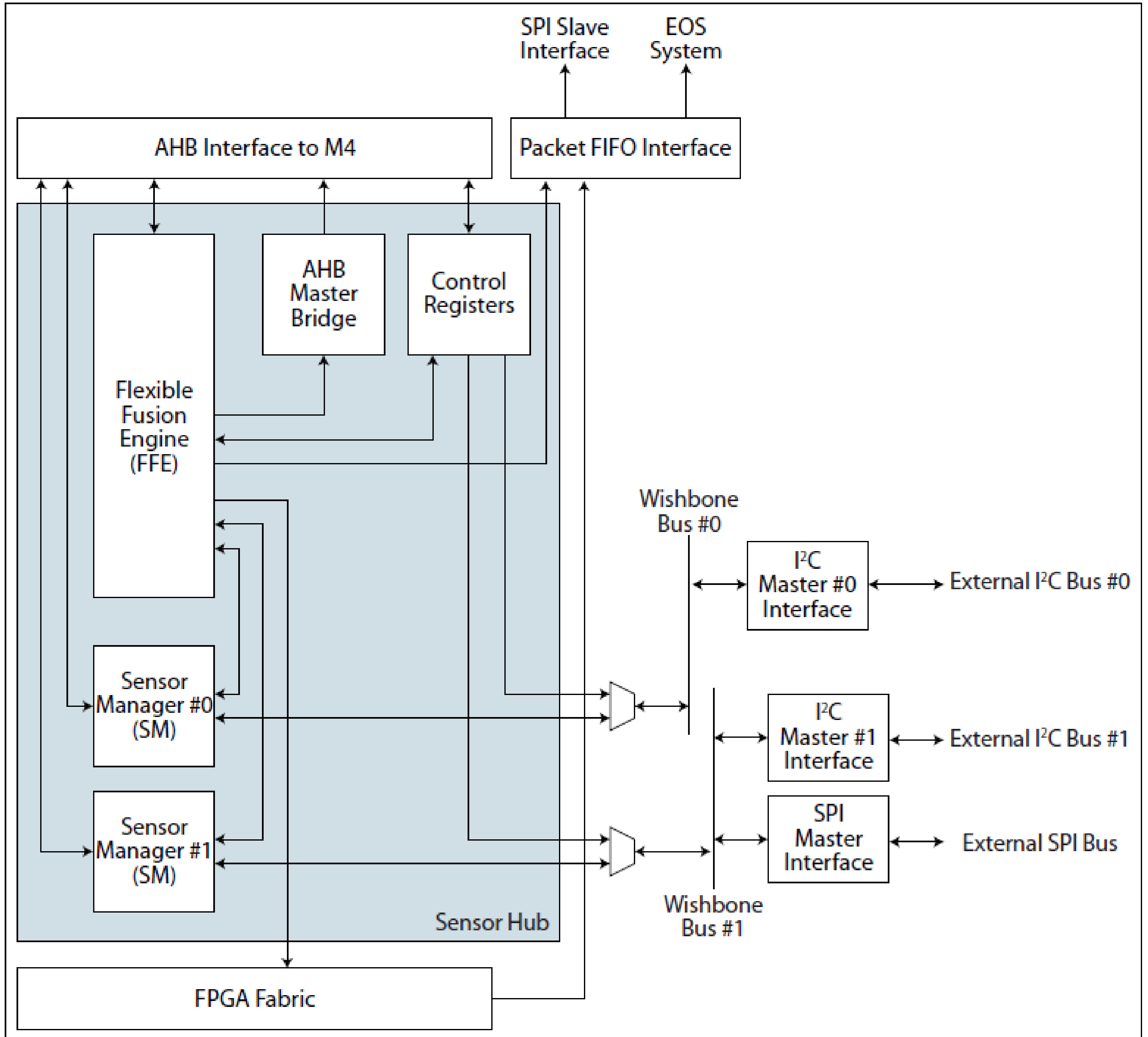
Preliminary

~~2.3~~ Sensor Processing Subsystem

~~2.3.1~~2.2.7 Overview

The Sensor Processing Subsystem provides the EOS device with the ability to perform sensor fusion operations while using low overall power. Figure 5 illustrate the architecture of this module.

Figure 5: Sensor Processing Subsystem Block Diagram



The key to the Sensor Processing Subsystem is the Flexible Fusion Engine (FFE). This block is responsible for coordinating the retrieval of sensor data by each Sensor Manager and using this data for sensor fusion operations.

Preliminary

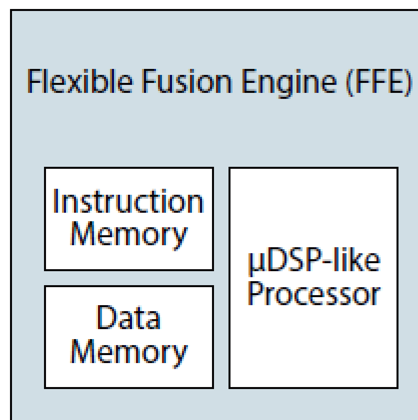
~~2.3.2~~2.2.8 Flexible Fusion Engine

The Flexible Fusion engine is responsible for the following:

- Coordinating the operation of the Sensor Manager(s)
- Retrieval of sensor data retrieved by the Sensor Manager(s)
- Sensor Fusion Calculations
- Transferring the results of the Sensor Fusion Calculations to the EOS S3AI platform
- Coordinating FFE operations with on-chip programmable IP

Figure 6 illustrates the features of the FFE architecture.

Figure 6: Flexible Fusion Engine Block Diagram



The FFE consists of three basic blocks:

- µDSP-Like Processor
- Instruction Memory
- Data Memory

Preliminary

~~2.3.2.1.1~~2.2.8.1.1 µDSP-Like Processor

The µDSP provides the main operation of the FFE. The µDSP retrieves instructions from the Instruction Memory along with data values stored in the Data Memory. In addition, the µDSP performs the following selected operations:

- Waiting for a "Start" signal from the EOS S3AI platform to begin processing
- Receive Mailbox values from the EOS S3AI platform to direct FFE processing
- Writing Mailbox values to Sensor Manager Memory - The Mailbox values determine which sensors are contacted by the Sensor Manager during each sampling period.
- Reading sensor data values from Sensor Manager Memory prior to starting a new Sensor Manager Session.
- Starting each Sensor Manager session to retrieve a new set of sensor data
- In parallel with the Sensor Manager session, performing Sensor Fusion calculations based on the sensor data values retrieved from Sensor Manager Memory.
- Sending the results of the Sensor Fusion calculations to the EOS S3AI platform - The FFE can use either the Packet FIFO interface or the AHB Master Bridge to pass packets of sensor data to the EOS S3AI platform.
- Coordinating FFE operations with On-chip programmable logic-based IP. This IP waits for a Start signal from the EOS S3AI platform prior to beginning processing.

~~2.3.2.1.2~~2.2.8.1.2 Instruction Memory

The Instruction Memory contains the instructions used by the µDSP for performing the Sensor Fusion operation. The EOS S3AI platform loads this memory prior to the beginning of the first FFE session. At the start of each session, µDSP is reading instructions from this memory staring at address "0" and will continue until a "Stop" instruction is read.

The EOS S3AI platform may elect to alter the Sensor Fusion operation on a session-by-session basis by passing Mailbox data from the Control Registers module. If the EOS S3AI platform does this, it does not need to modify the Instruction or Data Memories to support multiple Sensor Fusion processing modes. It is important to note that the FFE cannot write to its own Instruction Memory. Similarly, no other module within the Sensor Processing Subsystem can read or write to the Instruction Memory.

~~2.3.2.1.3~~2.2.8.1.3 Data Memory

The Data Memory contains the data portion of the µDSP' s program execution. The EOS S3AI platform loads this memory prior to the beginning of the first FFE session. The Instruction Memory determines which portions of Data Memory the Microcontroller will read or write thereafter. Values stored in Data Memory can include:

- Constant values
- Variable values
- Sensor data values

Preliminary

It is important to note that both the EOS S3AI platform and the FFE can write to this memory. However, this does not extend to any other module in the Sensor Processing Subsystem.

~~2.3.3~~2.2.9 Sensor Manager

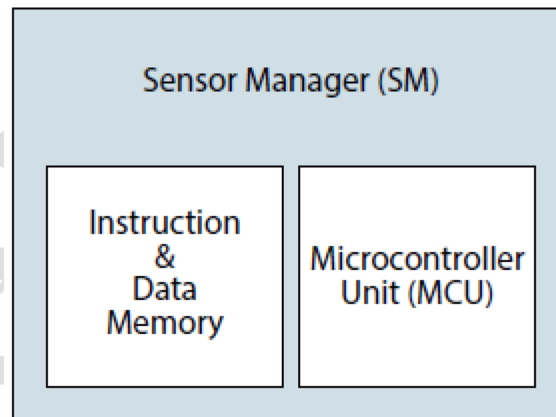
~~2.3.3.1~~2.2.9.1 Overview

The Sensor manager is responsible for the following tasks:

- Coordinating its actions with the FFE.
- Use the Wishbone bus interface to access either I^2C/SPI Master modules.
- Manage external sensors
  - Sensor data retrieval (ex. single values, burst transfers, FIFO transfers, etc.)
  - Sensor configuration
  - Sensor calibration
  - Sensor power state management
- Store sensor data in the proper format for retrieval by the FFE.

Figure 7 illustrates the features of the Sensor Manager Architecture.

Figure 7: Sensor Manager Block Diagram



The Sensor Manager consists of two parts. These are:

- Microcontroller Unit
- Instruction & Data Memory

Preliminary

~~2.3.3.1.1~~2.2.9.1.1 Microcontroller Unit

The Microcontroller Unit (MCU) is responsible for the operation of the Sensor Manager. The MCU retrieves instructions and data from its memory module and performs operations that include:

- Reading the Mailbox data written by the FFE - The Mailbox data defines which sensor handling routines to execute during the current Sensor Manager session
- Executing the selected sensor handling routines
- Accessing the target sensor through the appropriate I^2C Master or SPI Master interface
- Storing sensor data in the appropriate packet format for retrieval by the FFE

It is important to note that each Sensor Manager session is not necessarily targeted at sensor data retrieval. As mentioned earlier, some sessions may be targeted at configuring sensors, or changing power state. The FFE selects which sensor operations is active during each Sensor Manager session. Additionally, the Sensor Manager's focus is on sensor management. As such, it does not participate in Sensor Fusion calculations.

~~2.3.3.1.2~~2.2.9.1.2 Instruction and Data Memory

The Instruction and Data Memory holds the information that the Microcontroller uses for each of its processing sessions. Prior to the first processing session, the EOS S3AI platform (i.e. M4F or Application Processor) loads this memory with a series of sensor management routines. The algorithms loaded into the FFE determine, on a session by session basis, which of these routines the Sensor Manager should use. More specifically, the FFE's algorithms write to a Mailbox data structure in the Sensor Manager's memory. In response, the Sensor Manager only uses those routines enabled for the current session. The purpose for this approach is to enable Sensor Manager to sample each sensor at its own rate.

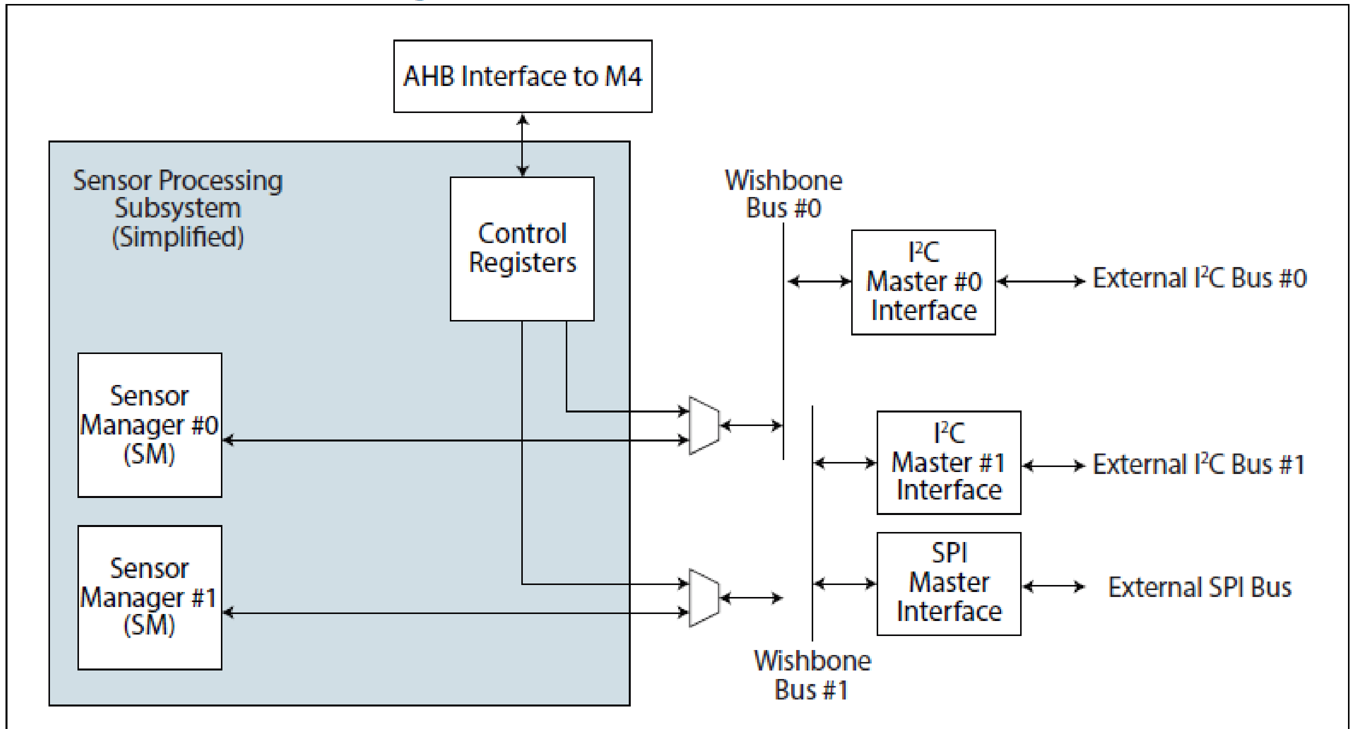
The Sensor Manager stores the retrieved sensor data into its Instruction and Data Memory. The location and format of the resulting data structure helps the FFE to correctly retrieve and process this data. The format of the data structure is not fixed by hardware. Rather, it is left to software to define and implement this structure as needed.

Preliminary

~~2.3.4~~2.2.10 I²C Master

There are two I²C Master modules in the EOS S3AI device; each one is assigned to a Sensor Manager module. In addition, the EOS S3AI also makes both I²C modules directly accessible to the EOS internal bus system. In each case, the I²C Master module provides the means to access devices on the associated I²C bus.

Figure 8: I2C Modules within the EOS S3AI platform



I²C is a two-wire, bi-directional serial bus that provides a simple and efficient method of data exchange between devices. It is most suitable for applications requiring occasional communication over a short distance between many devices. The I²C standard is a true multi-master bus including collision detection and arbitration that prevents data corruption if two or more masters attempt to control the bus simultaneously.

The interface defines 3 transmission speeds:

- Normal: 100Kbps
- Fast: 400Kbps
- High speed: 3.5Mbps

Only 100Kbps and 400Kbps modes are supported.

Preview

The following features are available in the I²C Slave Block:

- Compatible with Philips I²C standard
- Multi Master Operation
- Software programmable clock frequency
- Clock Stretching and Wait state generation
- Software programmable acknowledge bit
- Interrupt or bit-polling driven byte-by-byte data-transfers
- Arbitration lost interrupt, with automatic transfer cancelation
- Start/Stop/Repeated Start/Acknowledge generation
- Start/Stop/Repeated Start detection
- Bus busy detection
- Supports 7bit and 10bit addressing mode
- Operates from a wide range of input clock frequencies
- Static synchronous design
- Fully synthesizable

The following describe the I²C system operations,

~~2.3.4.1~~2.2.10.1 System Configuration

The I²C system uses a serial data line (SDA) and a serial clock line (SCL) for data transfers. All devices connected to these two signals must have open drain or open collector outputs. The logic AND function is exercised on both lines with external pull-up resistors.

Data is transferred between a Master and a Slave synchronously to SCL on the SDA line on a byte-by-byte basis. Each data byte is 8 bits long. There is one SCL clock pulse for each data bit with the MSB being transmitted first. An acknowledge bit follows each transferred byte. Each bit is sampled during the high period of SCL; therefore, the SDA line may be changed only during the low period of SCL and must be held stable during the high period of SCL. A transition on the SDA line while SCL is high is interpreted as a command (see START and STOP signals).

~~2.3.4.2~~2.2.10.2 I²C Protocol

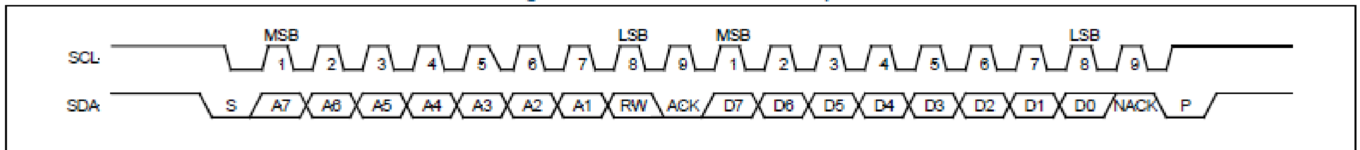
Normally, standard communication consists of four parts:

- START signal generation
- Slave address transfer
- Data transfer
- STOP signal generation

Figure 9 below illustrates an example of I²C protocol.

Preliminary

Figure 9: I²C Protocol Example



~~2.3.4.2.1~~2.2.10.2.1 START Signal

When the bus is free/idle, meaning no master device is engaging the bus (both SCL and SDA lines are high), a master can initiate a transfer by sending a START signal. A START signal, usually referred to as the S-bit, is defined as a high-to-low transition of SDA while SCL is high. The START signal denotes the beginning of a new data transfer.

A Repeated START is a START signal without first generating a STOP signal. The master uses this method to communicate with another slave or the same slave in a different transfer direction (e.g. from writing to a device to reading from a device) without releasing the bus.

The core generates a START signal when the STA-bit in the Command Register is set and the RD or WR bits are set. Depending on the current status of the SCL line, a START or Repeated START is generated.

~~2.3.4.2.2~~2.2.10.2.2 Slave Address Transfer

The first byte of data transferred by the master immediately after the START signal is the slave address. This is a seven-bits calling address followed by a RW bit. The RW bit signals the slave the data transfer direction. No two slaves in the system can have the same address. Only the slave with an address that matches the one transmitted by the master will respond by returning an acknowledge bit by pulling the SDA low at the 9th SCL clock cycle.

Note: The core supports 10bit slave addresses by generating two address transfers. See the Philips I²C specifications for more details.

The core treats a Slave Address Transfer as any other write action. Store the slave device’s address in the Transmit Register and set the WR bit. The core will then transfer the slave address on the bus.

Preliminary

~~2.3.4.2.3~~2.2.10.2.3 Data Transfer

Once successful slave addressing has been achieved, the data transfer can proceed on a byte-by-byte basis in the direction specified by the RW bit sent by the master. Each transferred byte is followed by an acknowledge bit on the 9th SCL clock cycle. If the slave signals a No Acknowledge, the master can generate a STOP signal to abort the data transfer or generate a Repeated START signal and start a new transfer cycle.

If the master, as the receiving device, does not acknowledge the slave, the slave releases the SDA line for the master to generate a STOP or Repeated START signal.

To write data to a slave, store the data to be transmitted in the Transmit Register and set the WR bit. To read data from a slave, set the RD bit. During a transfer the core set the TIP flag, indicating that a Transfer is In Progress. When the transfer is done the TIP flag is reset, the IF flag set and, when enabled, an interrupt generated. The Receive Register contains valid data after the IF flag has been set. The user may issue a new write or read command when the TIP flag is reset.

~~2.3.4.2.4~~2.2.10.2.4 STOP Signal

The master can terminate the communication by generating a STOP signal. A STOP signal, usually referred to as the P-bit, is defined as a low-to-high transition of SDA while SCL is at logical '1'.

~~2.3.4.2.5~~2.2.10.2.5 Arbitration

The I²C Master block supports multi-master arbitration. However, this feature is not supported by other elements of the EOS S3AI platform.

~~2.3.4.3~~2.2.10.3 I²C Core Architecture

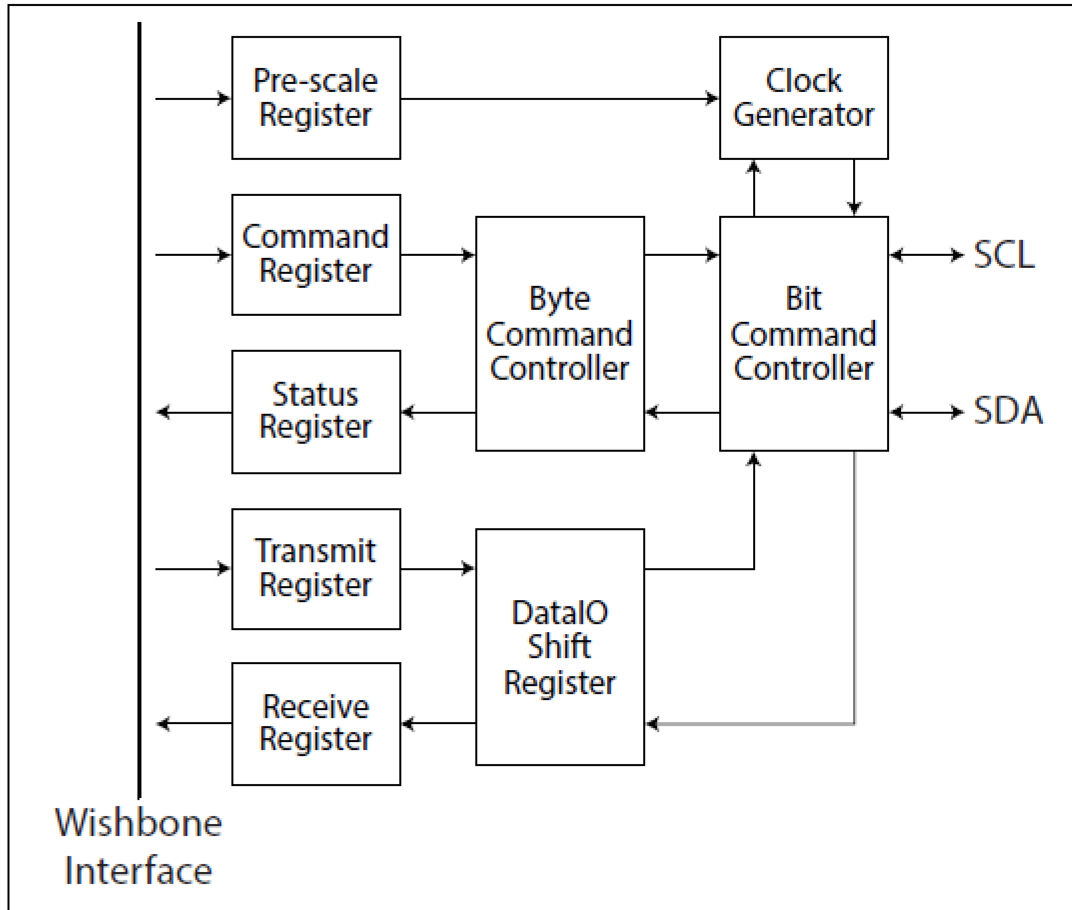
The I²C core is built around four primary blocks:

- Clock Generator
- Byte Command Controller
- Bit Command Controller
- DataIO Shift Register

Note: All other blocks are used for interfacing or for storing temporary values.

Preliminary

Figure 10: I2C Block Diagram



The Sensor Manager uses the Wishbone interface to access the I^2C Master during Sensor data transfers.

~~2.3.4.3.1~~2.2.10.3.1 Clock Generator

The Clock Generator generates an internal 4*Fscl clock enable signal that triggers all synchronous elements in the Bit Command Controller. It also handles clock stretching needed by some slaves.

~~2.3.4.3.2~~2.2.10.3.2 Byte Command Controller

The Byte Command Controller handles I^2C traffic at the byte level. It takes data from the Command Register and translates it into sequences based on the transmission of a single byte. By setting the START, STOP, and READ bit in the Command Register, the Byte Command Controller performs the following sequence:

- A START signal is generated
- The byte is read from the slave device
- A STOP signal is generated

Setting the START, STOP and READ bits and the Byte Command Controller sequence starts a process acts to divide each byte operation into separate bit-operations, which are then sent to the Bit Command Controller.

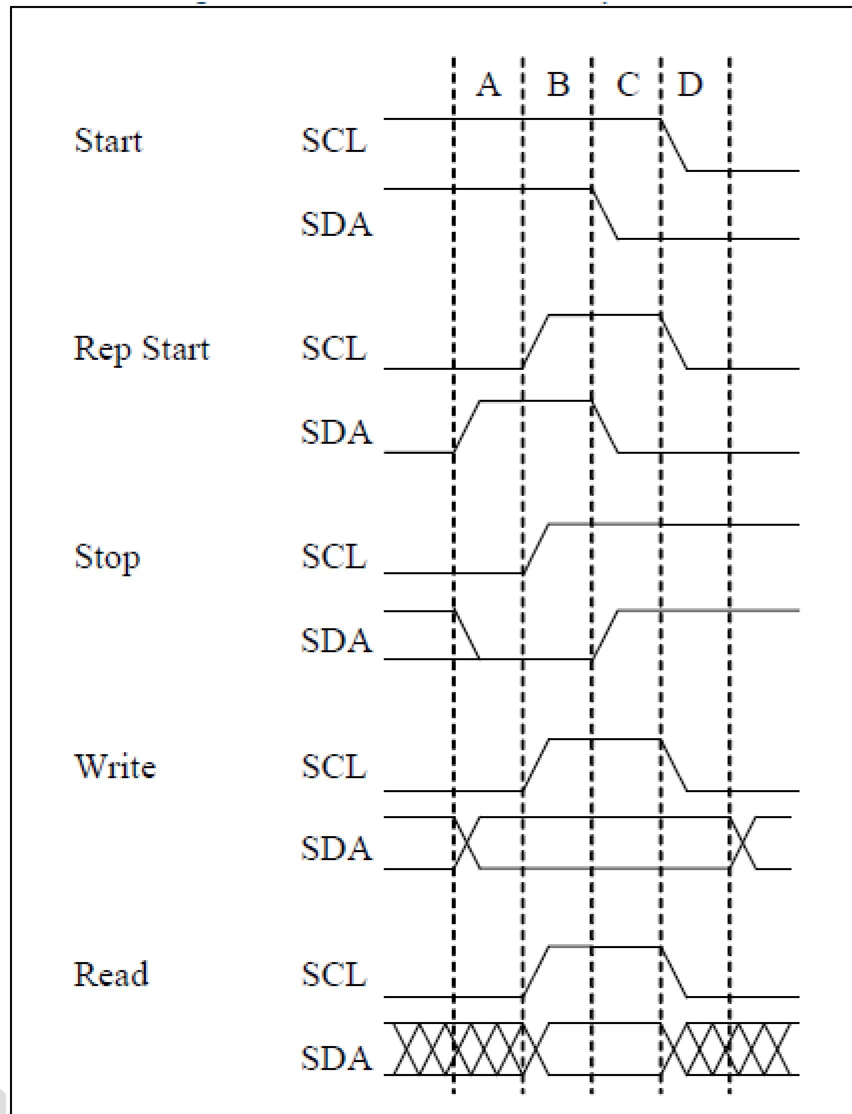
~~2.3.4.3.3~~2.2.10.3.3 Bit Command Controller

The Bit Command Controller handles the actual transmission of data and the generation of the specific levels for START, Repeated START, and STOP signals by controlling the SCL and SDA lines.

The Byte Command Controller tells the Bit Command Controller which operation needs to be performed. For a single-byte read, the Bit Command Controller receives eight separate read commands. Each bit-operation is divided into five smaller pieces (idle and A, B, C, and D), except for a STOP operation which is divided into four smaller pieces (idle and A, B, and C). Figure below illustrates the I2C bit command sequence.

Preliminary

Figure 11: I2C Bit Command Sequences



~~2.3.4.3.4~~2.2.10.3.4 Data I/O Shift Register

The DataIO Shift Register contains the data associated with the current transfer. During a read action, data is shifted in from the SDA line. After a byte has been read the contents are copied into the Receive Register. During a write action, the Transmit Register's contents are copied into the DataIO Shift Register and are then transmitted onto the SDA line.

© 2019 QuickLogic Corporation Confidential

2.3.52.2.11 Serial Peripheral Interface (SPI)

The EOS S3AI platform relies on three separate SPI interfaces.

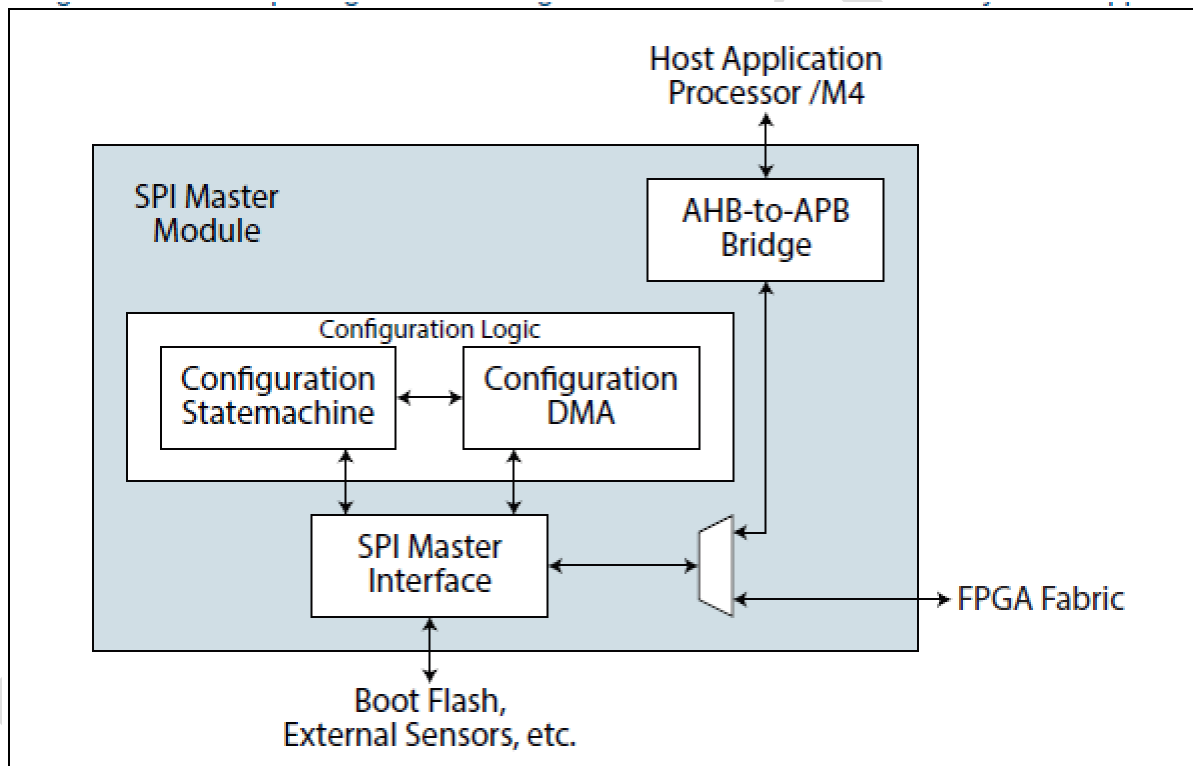
- SPI Master for System Support
- SPI Master for Sensor Subsystem Support
- SPI Slave

The following sections will discuss each of these interfaces.

2.3.5.12.2.11.1 SPI Master – System Support

IP within the on-chip programmable logic can directly access the SPI Master for system support interface. Figure below shows the connection.

Figure 12: SPI Master - System Support Block Diagram



The ability to directly access the SPI Master provides an IP designer with the option to create on-chip programmable logic IP that can directly access external devices such as Flash Memory or Sensors. In the latter case, the Sensor data values can be used for additional Sensor Fusion operations in parallel with the Sensor Processing Subsystem. In such cases, the on-chip programmable logic IP can use either the Packet FIFO Interface or use the SDMA to move the processed data into the EOS S3AI platform for further evaluation or processing.

During the initial boot operation, the SPI Master is exclusively accessed by the Configuration Logic. Once the Configuration Logic completes the initial boot operation, the SPI Master becomes available to the M4F for additional data retrieval from the external flash device. More specifically, the initial boot code provides a boot loader to the M4F for the M4F to complete its retrieval of M4F code.

Once the boot process completes, either the M4F or on-chip programmable logic can access the SPI Master Interface and use this to access any device on the SPI bus. This can be additional external flash devices, sensors, or system support devices such as Power Management devices

~~2.3.5.1.1~~2.2.11.1.1 Features

The SPI Master interface supports the following operations:

- Single SPI transfers
- DMA transfers of SPI data retrieved from an external flash device

The following features show the operation of this module.

SPI Master Interface provides the following:

- Operate as a Master only
- Support up to 3 slaves
- Default mode is mode 0 (this can be reprogrammed by M4)
- Default frame size of 8 (this can be reprogrammed by M4)
- Maximum transfer size of 64K frames.
- Support little Endian data ordering
- Shift out data most significant bit first.
- Support DMA transfers
- Support standard SPI protocol

SPI Master Interface will not provide the following:

- Support multiple SPI masters
- Support other serial protocols like SSP or Microwire
- Support for protocols that include DDR, dual, and quad transfers.

Preliminary

SPI Master Interface that is accessible by:

- Host Application Processor
- Configuration Logic
- Programmable On-chip programmable logic

Preliminary

Configuration Logic is responsible for:

a. Reading the external Flash device.
b. Using data stored within the external Flash device to configure its SPI transfer parameters.
c. Using the values stored within the external Flash to confirm that the boot code is compatible with the EOS device.
d. Loading the boot code into M4F memory and enabling the M4F execution once the boot code transfer has completed.
e. Minimizing the elapsed time for booting the M4F by using DMA transfers of boot code from the external Flash.
f. Posting status bits to the M4F that help to diagnose the state of the boot process.

~~2.3.5.1.2~~2.2.11.1.2 Configuration Logic

This logic consists of the following two parts:

- Configuration State machine
- Configuration DMA logic.

The Configuration State machine provides control over the retrieval of boot code from an external Flash device. To do so, it must configure and control the Configuration DMA and SPI Master Interface

The Configuration State machine programs the Configuration DMA and SPI Master Interface to access an external flash, by performing the following operations:

- Setting the correct SPI clock rate
- Awakened the external Flash device from a low power state (ex. Deep Sleep modes)
- Examining the boot code parameters to optimize SPI Master transfers
- Initiating DMA loading of the M4F boot code into M4F memory
- Enabling M4F operation.

As a part of this process, the Configuration State machine examines the EOS S3AI device ID in the boot Flash data. If this device ID is incorrect, the Configuration State machine will halt the boot process. This boot process can only be re-started by asserting reset.

Preliminary

~~2.3.5.2~~2.2.11.2 SPI Master – Sensor Processing Subsystem Support

The EOS SPI Host controller can communicate with up to eight SPI sensor devices with a WISHBONE Classic interface.

SPI Master Features:

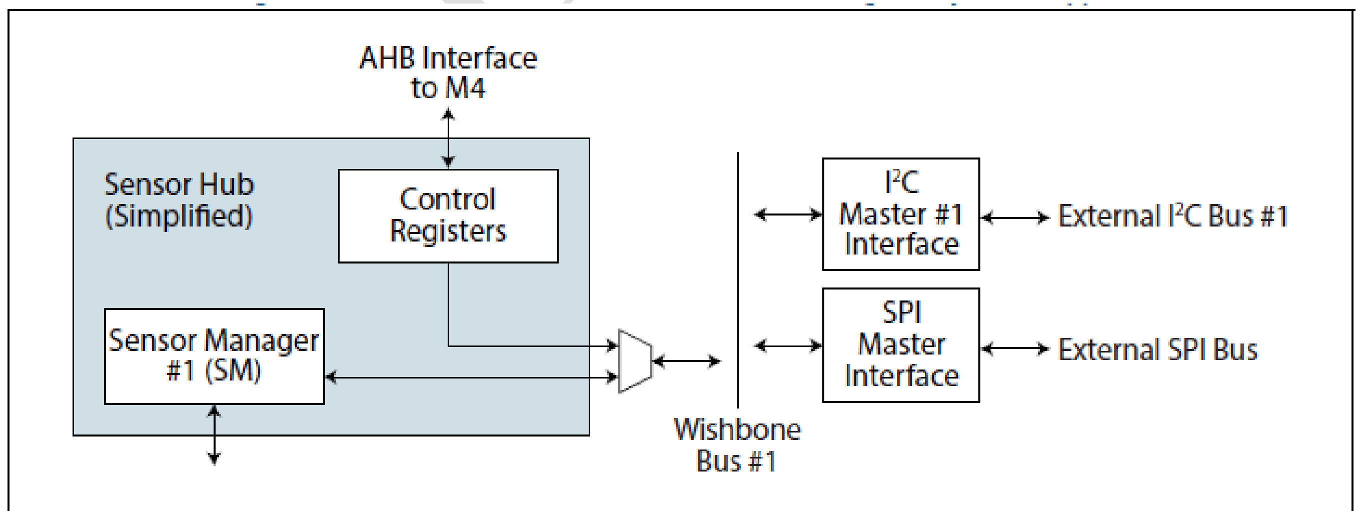
- Supports Master configuration (Multi-Master configuration is not supported).
- Capable of being connected to eight SPI slaves with individual slave select lines.
- Interrupt generation capability
- Serial clock with programmable phase and polarity.
- Four programmable transfer formats supported (controlled by CPOL and CPHA).
- Support all current SM instructions
- Support all operations as with I^{2}C Master Controller
- 8-Bit Wishbone interface same as the I^{2}C Master Controller

The SPI Master module used for Sensor Processing Subsystem Support resides on the same Wishbone bus as the I^{2}C Master module described in the Sensor Manager and Control Registers. Like the I^{2}C Master module, the SPI Master module enables the Sensor Processing Subsystem to communicate with external devices.

The SPI module is accessible to the M4F processor but is not used for operation such as retrieval of boot code from an external Flash storage device. This is left to the SPI Master used for System Support.

Figure below shows basic connections to the SPI Interface that supports the Sensor Processing Subsystem.

Figure 13: SPI Interface used for Sensor Processing Subsystem Support



Preliminary

~~2.3.5.3~~2.2.11.3 SPI Slave

The SPI Slave module provides the means to communicate between a host system and the EOS S3AI platform. This block consists of the SPI Interface and the Top-Level Controller (i.e. TLC) module.

The SPI Slave module has two roles:

- Setting up and debugging the EOS S3AI platform
- Retrieve run-time data

Both of these operations are possible during either “Companion” or “Standalone” EOS modes. However, in a typical IOT application, there is no local host to use the SPI Slave interface. Therefore, the following discussion is primarily concerned with the “Companion” mode of operation.

During EOS S3AI platform non-debug setup operations, the host system uses the SPI Slave module to write a pre-compiled binary file to the EOS device’s M4F memory. In addition, the host also writes pre-compiled binary files to other memories such as those used by the FFE (i.e. FFE) and the Sensor Manager (i.e. SM). Once complete, the host system enables the EOS’s M4F processor to execute the newly written binary code.

During normal EOS S3AI platform processing, the host uses the SPI Slave interface to retrieve the results of the EOS S3AI platform’s sensor fusion processing and passes this data to the corresponding application. How the host knows when to do this retrieval is beyond the scope of this section.

During EOS S3AI platform debugging operations, the host uses the SPI Slave interface in ways similar to the “setup” and “normal” processing operations outlined above. Specifically, the host uses the SPI Slave interface to perform the following tasks:

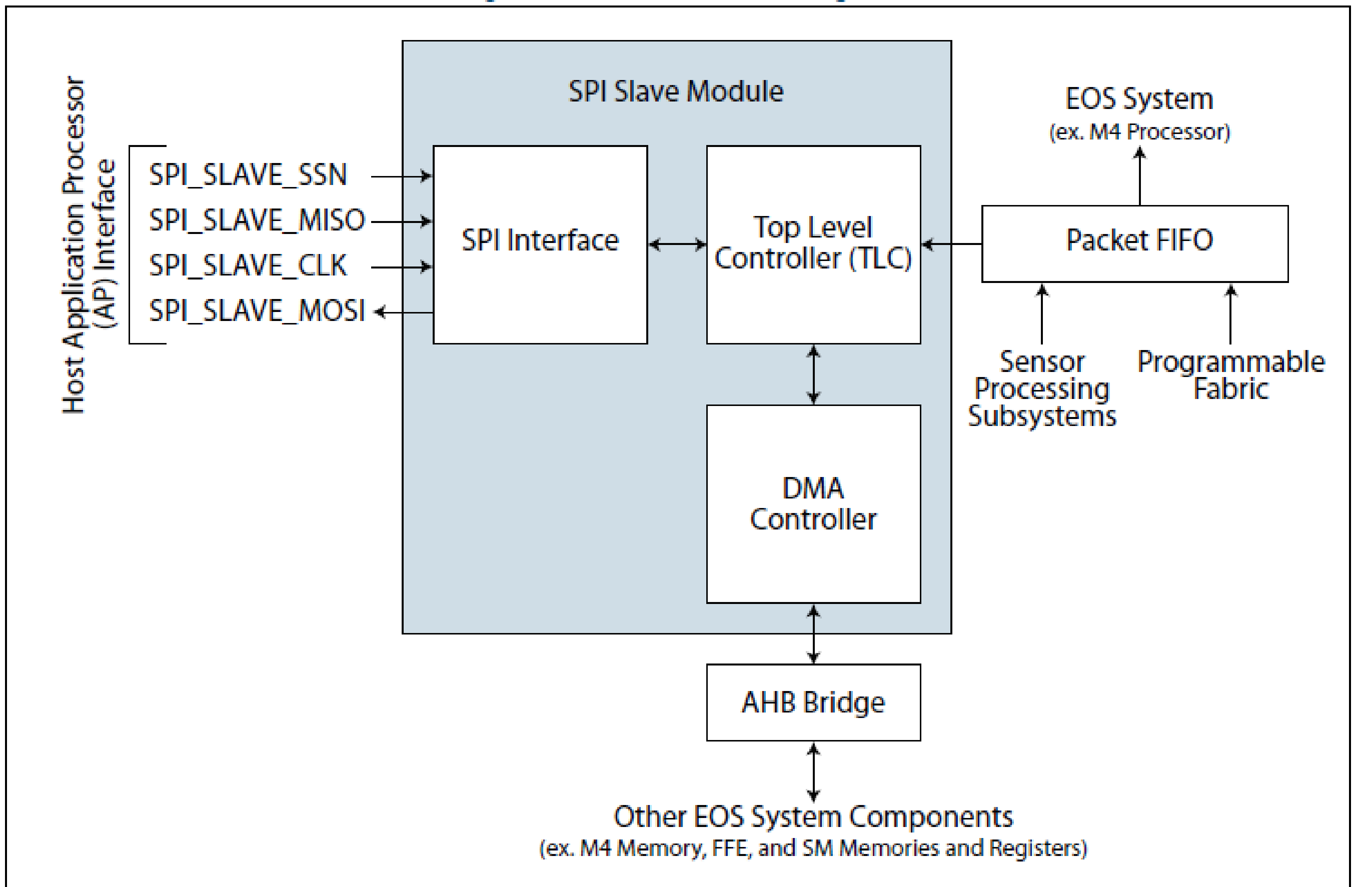
- Load debug code
- Enable the M4F to execute this code
- Retrieve the result

In addition, the host can elect to only enable certain subsystems to diagnose. For example, the host can configure the FFE and Sensor Manager with diagnostic code, enable these subsystems to do sensor fusion processing, and retrieve the result of this processing all without enabling the EOS’s M4F processor.

Preliminary

Figure 14 illustrates the SPI Slave interface.

Figure 14: SPI Slave Block Diagram



It is important to note that the SPI Slave module does not communicate directly to other functional blocks. Rather, it must perform transfers through one or more AHB bridges in order to access any register or memory within the EOS S3AI platform. For optimum performance of this interface during the "Companion" mode, the EOS S3AI platform should avoid using common paths through the AHB infrastructure that are used by the SPI Slave interface.

~~2.3.5.4~~2.2.11.4 SPI Interface Protocol

The SPI Interface block supports SPI Mode 0 only:

CPOL = 0, the base value (idle state) of the clock is 0.
CPHA = 0, data is captured on the rising edge of the clock, and data is driven on the falling edge of the clock.

A transaction consists of SPI_SS being driven low (active) by the SPI Master, and then driven high after all of the desired bytes have been transferred. The SPI Interfaces assumes that all transfers will consist of complete bytes. Any incomplete bytes at the end of a transaction are ignored by the hardware.

The EOS's SPI Interface protocol supports several different operations. The following sections will outline these operations.

~~2.3.5.4.1~~2.2.11.4.1 Basic Read/Write Transfers

For Basic Read/Write transfers to the TLC's registers, the EOS S3AI platform SPI Interface protocol requires that the Address Byte be transmitted first. The Address Byte includes a single Direction Bit representing the direction for the transfer (write vs. read). As per SPI protocol, SPI_SLAVE_SSn is also required to be "LOW" when active SPI transactions are in progress. The rising edge of SPI_SLAVE_CLK captures the data bits on SPI_MSLAVE_MOSI and SLPI_SLAVE_MISO pins.

The Direction Bit is positioned in the most-significant bit of the Address Byte. The value of the Direction Bit is 1 for write transactions, and 0 for read transactions. The remaining 7 bits represent the register address within the TLC module. This address is unique to the TLC module and should not be confused with M4F addresses.

Examples:

Read starting from TCL address 0x05 → Address Byte = 0x05.
Write starting to TLC address 0x03 → Address Byte = 0x83.

During write transactions, the first byte received on the Master Out Slave In (SPI_SLAVE_MOSI) pin corresponds to the Address Byte. The next bytes received correspond to Data Bytes. No valid data is driven on the Master In Slave Out (SPI_SLAVE_MISO) pin. If the SPI clock is not free-running (for example, it does not toggle while SPI_SLAVE_SSN is inactive), the hardware requires that there be at least two extra rising clock edges on the SPI clock following the completion of a write transfer.

This ensures that the write data reaches its destination. The simplest way to generate these extra clock cycles is to perform a read from the TLC's "Scratch Register" (refer to the TLC register listing in later sections).

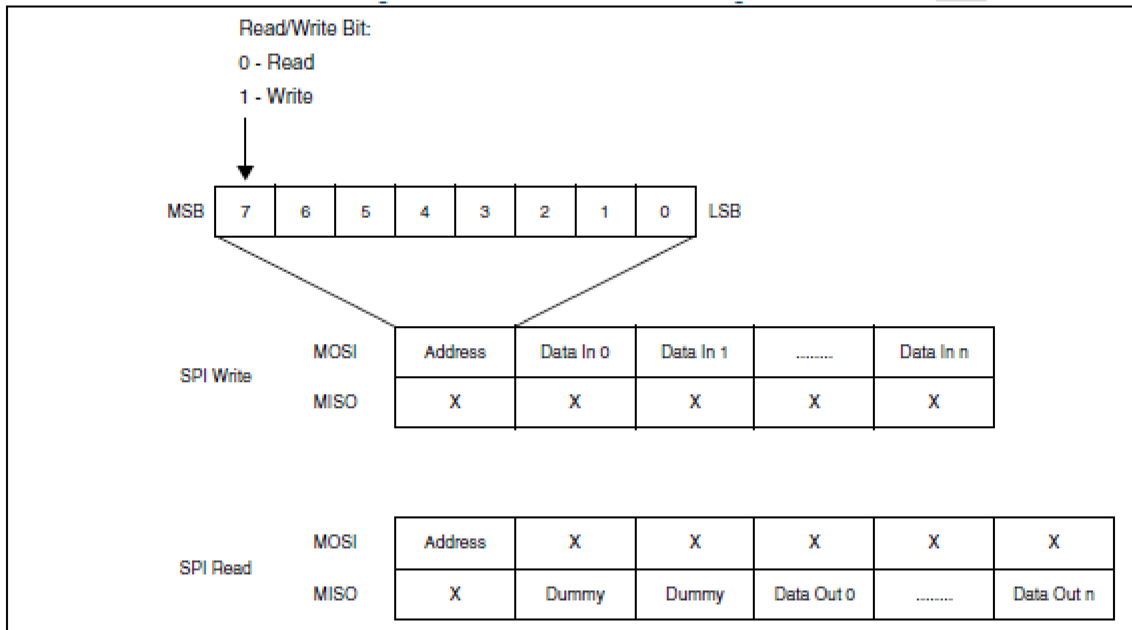
During read transactions, the first byte received on the MISO pin corresponds to the Address Byte. The SPI Interface ignores all subsequent data bits on the MISO pin. Instead, following the Address Byte, the SPI Interface begins to drive data on the MOSI pin.

Preliminary

This begins by outputting two dummy bytes followed by the first Data Byte. Thereafter, each byte transmitted on the MOSI pin will correspond to valid Data Bytes. Unlike write transfers, read transfers do not require extra SPI clock cycles after each transfer.

Figure 15 shows an example of the SPI Interface protocol. In that use case, the MSB bit is shifted out first.

Figure 15: SPI Slave Protocol Diagram



~~2.3.5.4.2~~2.2.11.4.2 Device ID Read

The Device ID transfer cycle is a special protocol cycle for identifying the EOS device to the Host SPI controller. Unlike the write transfer cycle described earlier, the address value of 0xFF will indicate to the SPI Interface that an ID read cycle is underway. In response, the SPI Interface will return an ID value 0x21 on the MISO pin one byte after the address phase.

Figure 16 shows an example of the SPI Slave Device ID read protocol.

Figure 16: SPI Slave ID Read Protocol Diagram

MOSI	0xFF	X	X
MISO	X	Dummy	0x21

~~2.3.5.4.3~~2.2.11.4.3 Transfer Types

There are three basic transfer types. These are:

- Transfers to TLC local registers
    - These transfers access TLC local registers alone and do not produce any activity on the AHB interface.
- Transfers from Packet FIFOs.
    - These transfers also access the TLC local registers. However, the goal of these accesses is to read one or more bytes from the Packet FIFOs.
    - These read transactions can be conducted as a single or burst transfer.
- Transfers to resources in the M4F Memory address space.
    - These transfers also access the TLC local registers. However, the goal of these accesses is to conduct a transfer using the AHB interface.
    - These transfers provide for the following operations:
        - Burst write transfers to the M4's Memory space.
        - Burst read transfers from the M4's Memory space.

The following sections will discuss each of these transfer types.

~~2.3.5.4.3.1~~2.2.11.4.3.1 Transfers to TLC Local Registers

This transfer type depends upon the type of TLC register being accessed. The default TLC response is to increment automatically the TLC register address (not the M4F Memory space address) after each byte accessed (for example, as in a read from the M4F Memory Address registers). For details see Figure TBD, where the SPI protocol address phase uses the address for the Memory Address Byte 0 register.

Once this transfer has completed, the TLC automatically increment its register address to "Memory Address Byte 1". Similarly, once this transfer has completed, the TLC automatically increment its register address to "Memory Address Byte 2". This sequence will repeat until all bytes have been transferred or a TLC register address is reached that prevents this auto-incrementing operation. For details about TLC register types that prevent auto-incrementing, see Transfers to M4F Memory Address Space section.

Note: This auto-incrementing operation does not prevent any special features in these registers from being triggered. For exceptions, see Basic AHB Transfer Restrictions section.

~~2.3.5.4.3.2~~2.2.11.4.3.2 Transfers from Packet FIFOs

Packet FIFOs accessible from within TLC can only be read and not written. Therefore, writes to the Packet FIFOs are ignored. Conversely, since these are FIFOs, a burst read from these FIFOs requires that the same TLC register address be accessed multiple times. For this reason, the TLC does not increment its register address when accessing any Packet FIFO address.

Preliminary

~~2.3.5.4.3.3~~2.2.11.4.3.3 Transfers to M4F Memory Address Space

The following sections will outline the transfer types and restrictions when accessing the M4's Memory Address space through the AHB Bridge interface.

~~2.3.5.4.3.3.1~~2.2.11.4.3.3.1 Basic AHB Transfer Restrictions

The TLC restricts AHB Memory transfers to 4 bytes per transfer cycle. No other transfer size is currently supported. The following sections will expand on additional transfer specific restrictions.

- AHB Memory Burst Write

  All AHB write operations through the TLC registers. A single write transfer is treated as a burst of one 32-bit word. To setup an AHB write operation, the Host needs to write the TLC AHB Access Control to 0x3.

  Set up the target AHB address by writing to the TLC Memory Address Byte 3 – 0. Keep the TLC Memory Address Byte 0 as bits[1:0] to 0x3. For example, writing to address 0x20001040 means writing:

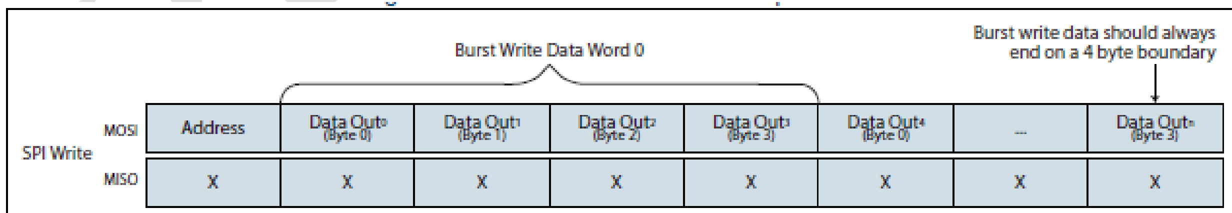
  - TLC Memory Address Byte 0 to 0x43
  - TLC Memory Address Byte 1 to 0x10
  - TLC Memory Address Byte 2 to 0x00
  - TLC Memory Address Byte 3 to 0x20

  This is followed by writing the data value into TLC Memory Data Byte 0 – 3. Upon writing to the Memory Data Byte 3 register, the Memory Address Byte 0 – 1 registers are automatically incremented.

  In addition, the TLC registers address loops back to point to the first data byte register, Memory Data Byte 0. By doing this, a block of data may be written from a single SPI data stream (for example, one address phase followed by a series of data phases representing the burst data).

  Figure 17 shows an example of a burst write sequence.

Figure 17: SPI Master Burst Write Sequence



Preliminary

AHB Memory write transfer block sizes are currently limited to 64K bytes (for example, 16K, 32-bit words). In addition, access restrictions through the AHB Bridge require each transfer cannot consist of more than four bytes. Consequently, writes to the TLC Memory Data Byte 3 register automatically trigger an increment by four in the TLC Memory Address Byte 0 – 1 registers, as well as triggering as data write through the AHB interface. The TLC Memory Address Bytes 2 – 3 registers are not affected by writes to Memory Data Byte 3. Therefore, burst transfers that exceed the 64K byte boundary automatically wrap back to the beginning.

- AHB Memory Burst Read

To complement the Burst Write transfer of the last section, the EOS device provides a Burst Read transfer operation. Unlike the Burst Write, the EOS S3AI device implements its Burst Read transfer as a DMA operation.

Prior to initiating Burst Read transfers, software must first check the following:

- There is no Burst Read transfer underway. All of the data from the previous Burst Read transfer must be read out through the SPI Interface prior to initiating another Burst Read transfer.
- There is data in the Burst Read transfer FIFO. This can be checked by reading the "Burst FIFO Status" register.

Once both conditions are true, the Host can configure the Burst Read transfer by writing the target address into the TLC Burst Read AHB Byte Address Byte 0 – 3 registers. This is followed by writing the data value into the TLC Burst Size Byte 0 register. The Burst Read begins with writing to the Burst Size Byte 1 register, and there needs to be two *dummy* byte cycles following the write to the Burst Size Byte 1 register.

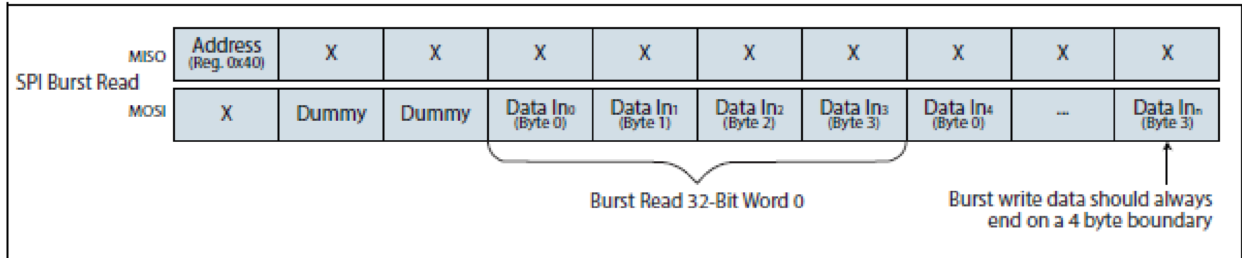
The simplest way to do this is by reading the Burst FIFO Status register, where this status is used to determine that there is a minimum of one byte available to be read from the Burst Read Data register. If the minimum (one Burst Read data byte) is available, the burst read operation begins by reading from the Burst Read Data register.

The TLC supports the burst transfer operation by not incrementing its register address pointer when it reads from the Burst Read Data register. If a single word needs to be read, set both the Burst Size Byte 0 and 1 registers to zero.

Figure 18 shows an example of the sequence.

Preliminary

Figure 18: Example Burst Read Sequence

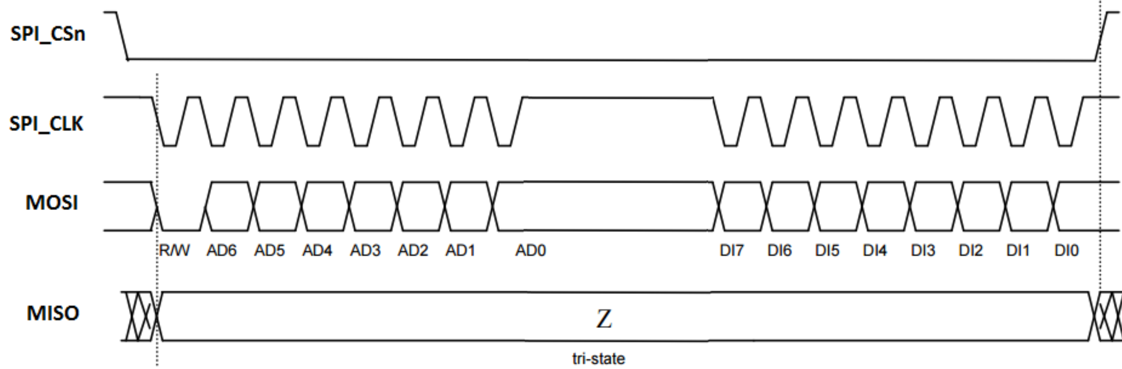


Preliminary

~~2.3.5.4.4~~2.2.11.4.4 SPI Write Cycle

Basic SPI write operation with mode 11 (CPOL = 1, CPHA = 1)

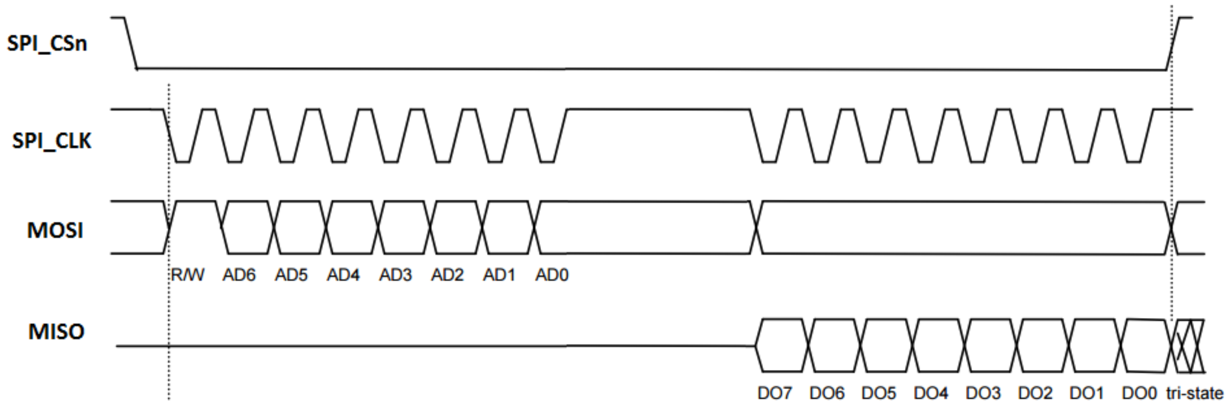
Figure 19: Basic SPI Write Operation (mode 11)



~~2.3.5.4.5~~2.2.11.4.5 SPI Read Cycle

Basic SPI read operation with mode 11 (CPOL = 1, CPHA = 1)

Figure 20: Basic SPI Read Operation (mode 11)

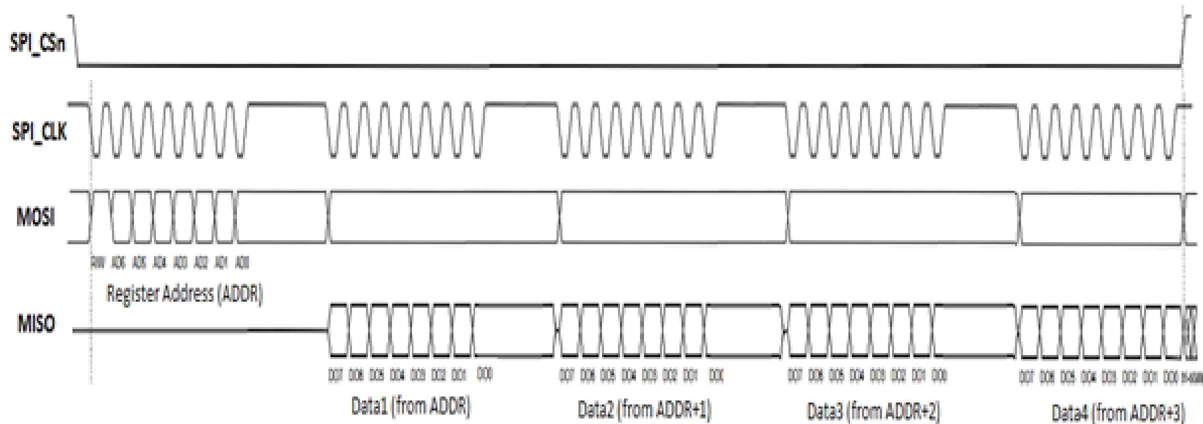


~~2.3.5.4.6~~2.2.11.4.6 SPI Multiple Read Cycle

SPI Multiple read operation with mode 11 (CPOL = 1, CPHA = 1).

Multiple read operations are possible by keeping the SPI_CSn low and continuing the data transfer. Only the first register address needs to be written. Addresses are automatically incremented after each read as long as the SPI_CSn line is held low.

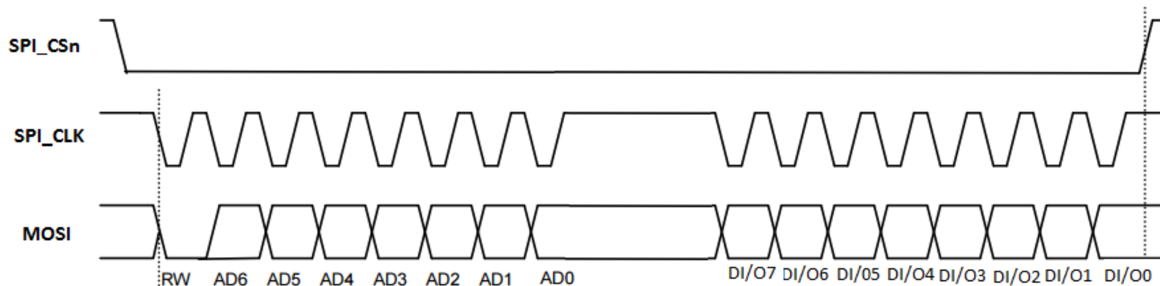
Figure 21: SPI Multiple Read Operation (mode 11)



~~2.3.5.4.7~~2.2.11.4.7 SPI 3 wire configuration

Basic SPI read / write operation with mode 11 (CPOL = 1, CPHA = 1) in 3 wire configuration.

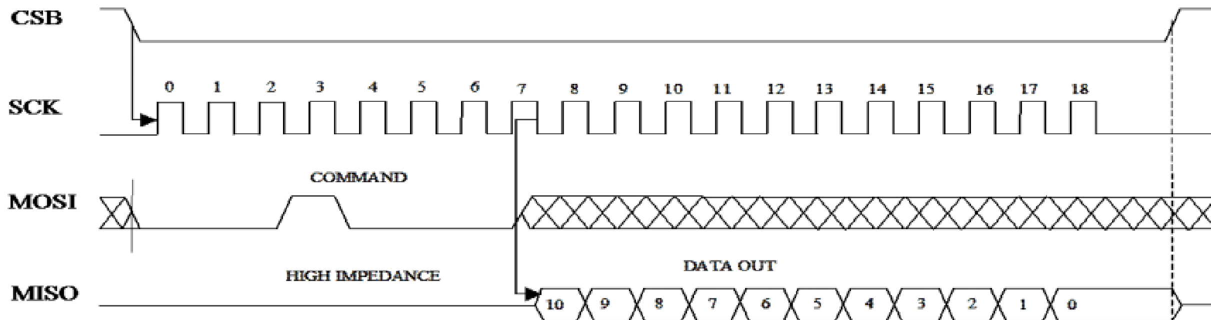
Figure 22: 3-Wire Basic SPI Read/Write Sequence (mode 11)



2.3.5.4.82.2.11.4.8 SPI Corner Cases

There are cases where certain sensors do not support 8-bit data alignment. In the following example, transfer sequence comes from a muRata™ SCA100T-D07 2-Axis High Performance Analog Accelerometer.

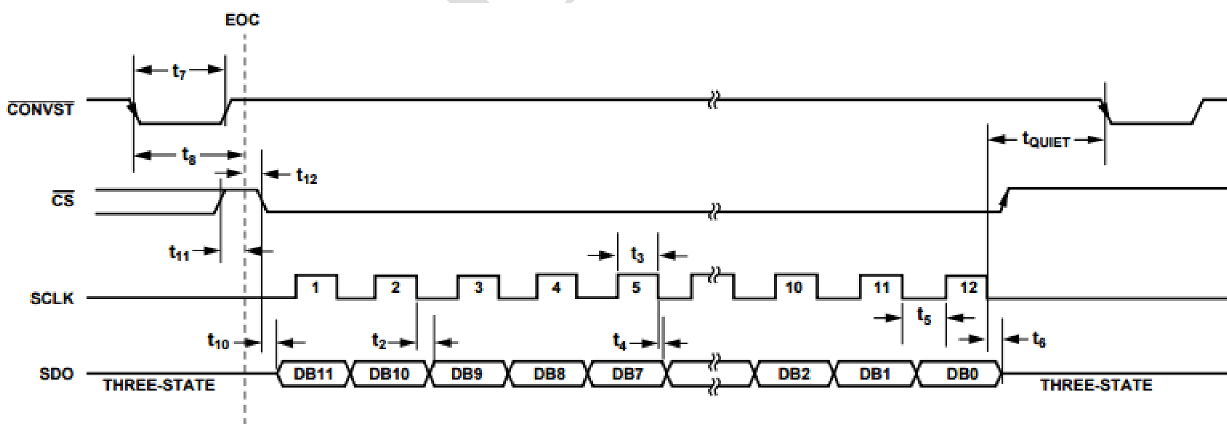
Figure 23: muRata Command and 11-bit SPI Acceleration Data Read Sequence



If the SPI interface is to support additional devices such as SPI based Analog-to-Digital converters (ex. to measure the output of additional sensor types) then the number of potential non-byte aligned shift sequences increases significantly.

For example, the SPI transfer sequence corresponding to the AD7091 Low Power, 12-bit ADC is shown in Figure below.

Figure 24: AD7091 SPI Transfer Sequence



Both Figure 23 and Figure 24 show that the data portion of the transfer consists of values that are not a direct multiple of 8-bits. Therefore, the SPI transfer would need to first store the MSB bits (ex. DB[11:8] for the AD7091) using a shift sequence of 4 SPI clock cycles prior to storing the LSB byte.

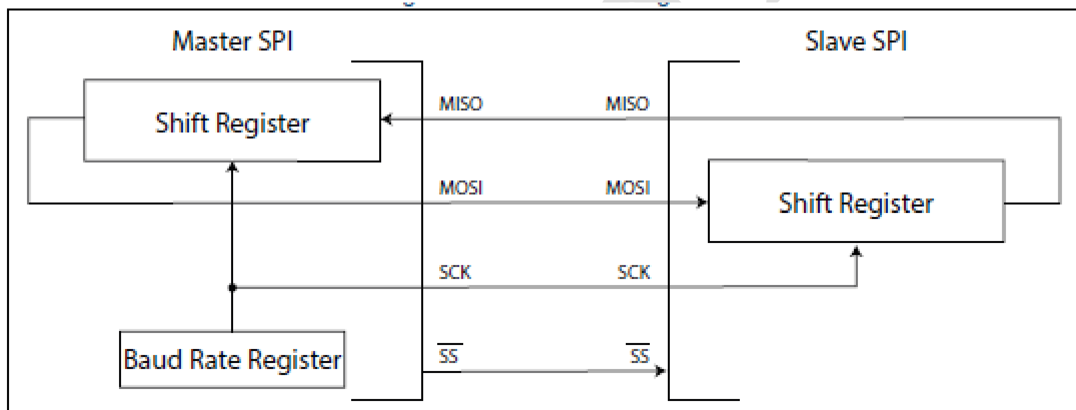
These are just a few of the potential corner cases that SPI based devices may present. To support such cases, the following capabilities have been added:

- Shift data based on 1 to 8 shift clock cycles.
- Keep the Chip Select signal low between SPI transfer cycles.
- Allow for running the SPI shift clock for a preset number of cycles after Chip Select has been de-asserted. Some devices require additional shift clock cycles after Chip Select is removed in order to achieve, for example, low power states or to complete the requested operation.

~~2.3.5.4.9~~2.2.11.4.9 Transmission Format

During an SPI transmission, data is transmitted (shifted out serially) and received (shifted in serially) simultaneously. The serial clock (SCK) synchronizes shifting and sampling of the information on the two serial data lines. A slave select line allows selection of an individual slave SPI device, slave devices that are not selected do not interfere with SPI bus activities.

Figure 25: SPI Block Diagram



Preliminary

~~2.3.5.4.10~~2.2.11.4.10Clock Phase and Polarity Controls

Using two bits in the SPI Control Register1, software selects one of four combinations of serial clock phase and polarity.

- The CPOL clock polarity control bit specifies an active high or low clock and has no significant effects on the transmission format.
- The CPHA clock phase control bit selects one of two fundamentally different transmission formats.

Clock phase and polarity should be identical for the master SPI device and the communicating slave device. In some cases, the phase and polarity may be changed between transmissions to allow a master device to communicate with peripheral slaves that have different requirements.

Preliminary

~~2.3.5.4.10.1~~2.2.11.4.10.1 CPHA = 0 Transfer Format

The first edge on the SCK line clocks the first data bit of the slave into the master, and the first data bit of the master into the slave. In some peripherals, the first bit of the slave data is available at the slave data out pin as soon as the slave is selected. In this format, the first SCK edge is issued a half cycle after SS has become low.

A half SCK cycle later, the second edge appears on the SCK line. When this second edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the shift register, depending on Least Significant Bit First Enable (LSBFE).

After this second edge, the next bit of the SPI Master data is transmitted out of the serial data output pin of the master to the serial input pin on the slave. This process continues for 16 edges on the SCK line, with data being latched on odd numbered edges, and shifted on even numbered edges.

Data reception is double buffered. Data is shifted serially into the SPI shift register during the transfer, and is transferred to the parallel SPI data register after the last bit is shifted in. After the sixteenth (the last of the edges) on the SCK line:

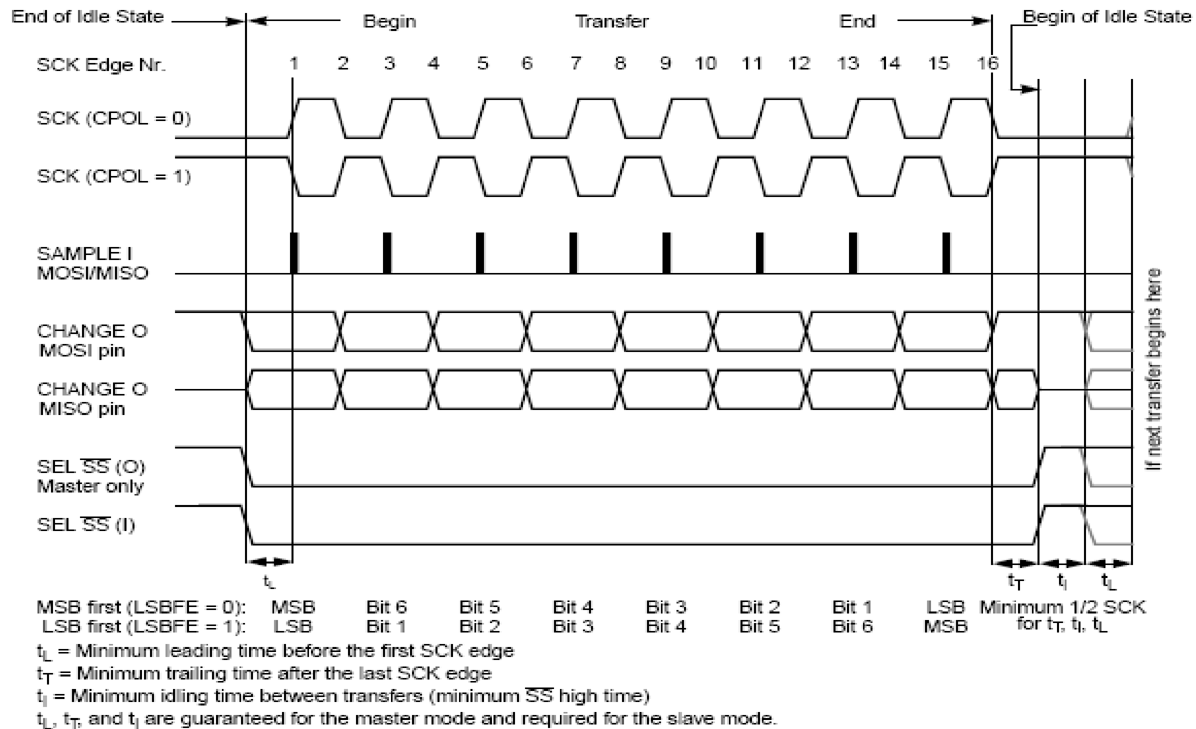
- Data that was previously in the master SPI Data Register should now be in the slave data register and the data that was in the slave data register should be in the master.
- The SPIF flag in the SPI Status Register is set indicating that the transfer is complete.

Figure 26 shows a timing diagram of an SPI transfer where CPHA = 0, with SCK waveforms are shown for CPOL = 0 and CPOL = 1. The diagram may be interpreted as a master or slave timing diagram since the SCK, MISO, and MOSI pins are connected directly between the master and the slave.

The MISO signal is the output from the slave and the MOSI signal is the output from the master. The SS pin of the master must be either high or reconfigured as a general-purpose output not affecting the SPI.

Preliminary

Figure 26: SPI Clock Format 0 (CPHA = 0)



~~2.3.5.4.10.2~~2.2.11.4.10.2 CPHA = 1 Transfer Format

Some peripherals require the first SCK edge before the first data bit becomes available at the data out pin, and the second edge clocks data into the system. In this format, the first SCK edge is issued by setting the CPHA bit at the beginning of the 8-cycle transfer operation.

The first edge of SCK occurs immediately after the half SCK clock cycle synchronization delay. This first edge commands the slave to transfer its first data bit to the serial data input pin of the master. A half SCK cycle later, the second edge appears on the SCK pin, which is the latching edge for both the master and slave.

When the third edge occurs, the value previously latched from the serial data input pin is shifted into the LSB or MSB of the SPI shift register, depending on LSBFE bit. After this edge, the next bit of the master data is coupled out of the serial data output pin of the master to the serial input pin on the slave.

This process continues for a total of 16 edges on the SCK line with data being latched on even numbered edges and shifting taking place on odd numbered edges. Data reception is double-buffered;

data is serially shifted into the SPI shift register during the transfer and is transferred to the parallel SPI data register after the last bit is shifted in.

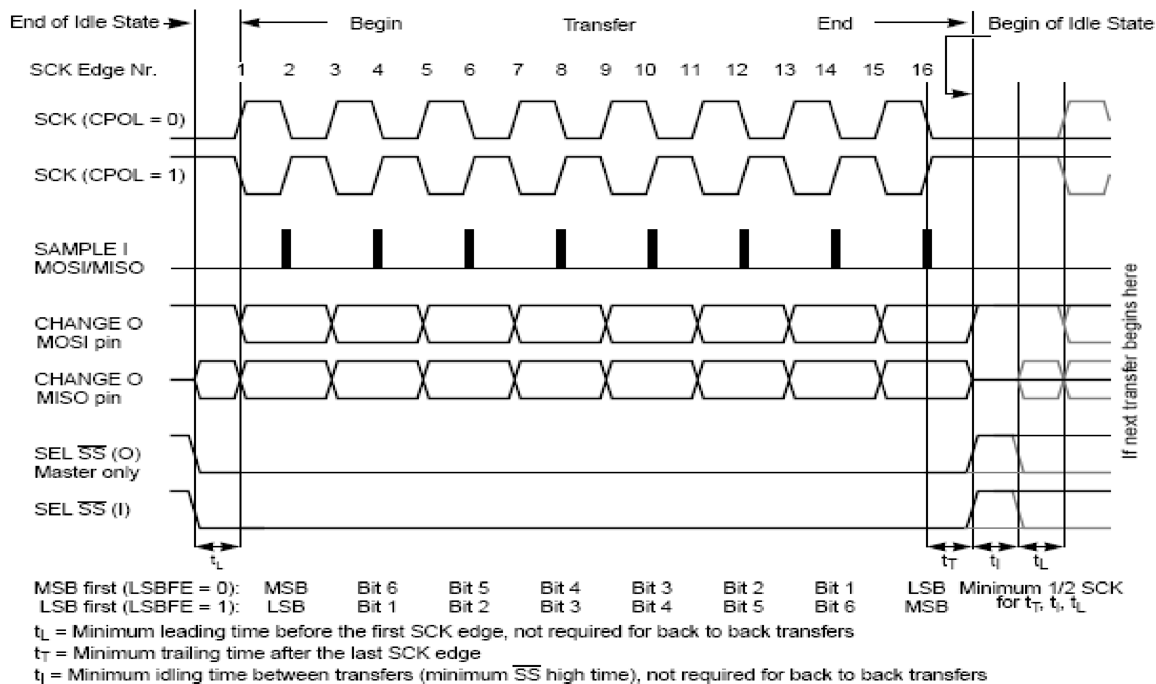
After the 16th SCK edge:

- Data that was previously in the SPI Data Register of the master is now in the data register of the slave, and data that was in the data register of the slave is in the master.
- The SPIF flag bit in SPISR is set indicating that the transfer is complete.

Figure 27 shows two clocking variations for CPHA = 1. The diagram may be interpreted as a master or slave timing diagram since the SCK, MISO, and MOSI pins are connected directly between the master and the slave.

The MISO signal is the output from the slave, and the MOSI signal is the output from the master. The SS line is the slave select input to the slave. The SS pin of the master must be either high or reconfigured as a general-purpose output that does not affect the SPI.

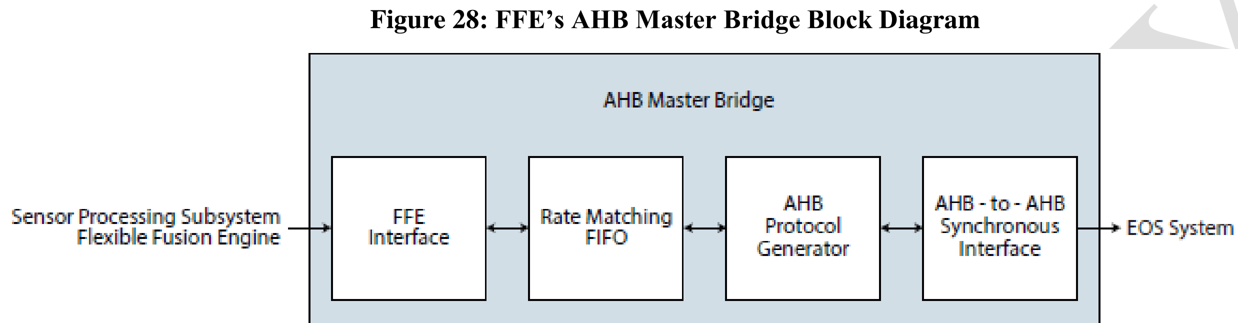
Figure 27: SPI Clock Format 1 (CPHA = 1)



~~2.3.6~~2.2.12 AHB Master Bridge

The FFE AHB Master Bridge gives the FFE of the Sensor Processing Subsystem the ability to write directly to some of the EOS S3AI platform's resources. Figure 28 shows that this interface is composed of four functional units.

Figure 28: FFE's AHB Master Bridge Block Diagram



The primary purpose for the AHB Master Bridge is to enable the Sensor Processing Subsystem's FFE to conduct the following operations:

a. Enable the Sensor Processing Subsystem's FFE to initiate updates of the FFE's memories using the M4F DMA controller. This requires the M4F to configure the DMA controller in advance of the FFE DMA request.
b. To use sections of the M4F memories as a large FIFO. This enables the storage of a larger amount of processed sensor data than is available via other hardware paths.

~~2.3.7~~2.2.13 Control Registers

The Control Registers block contains a series of register used for accessing the operations of the Sensor Processing Subsystem. These registers include the following:

- Wishbone Bus access to multiple I^{2}C and SPI Interfaces
- Access to the FFE and Sensor Manager (SM) memories
- Debug resources for both the FFE and SM
- Execution control and status for both the FFE and SM
- Interrupt resources for the Sensor Processing Subsystem

Preliminary

~~2.3.8~~2.2.14 Packet FIFO

The Packet FIFO interface enables the FFE to pass sensor data in the form of packets to the EOS S3AI platform. These packets can contain either data resulting from Sensor Fusion processing or can contain unprocessed sensor data. The format and content of each packet is entirely determined by the algorithm running on the FFE.

~~2.3.9~~2.2.15 On-Chip Programmable Logic

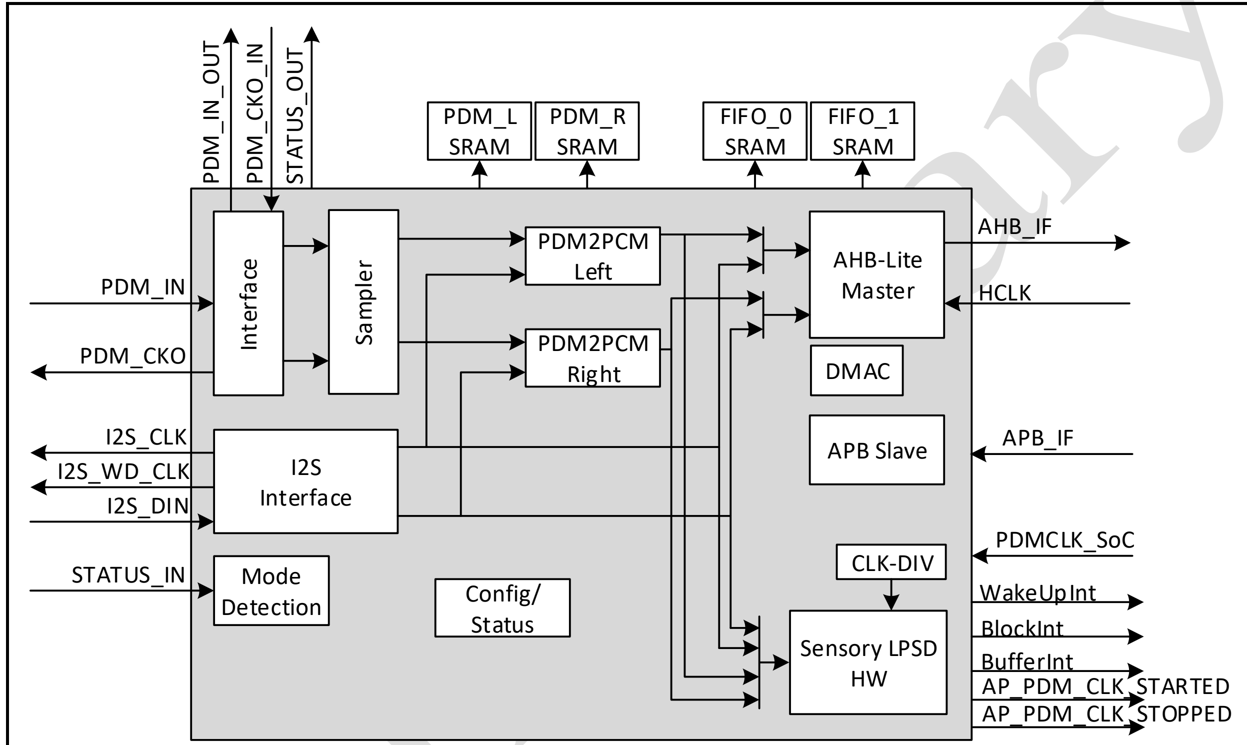
The FFE has the capability to pass a Start signal to an IP in the on-chip programmable logic. Similarly, the IP in the on-chip programmable logic can pass a Busy signal to the Sensor Processing Subsystem. The objective is to extend the coordination of Sensor Fusion processing to IP in the on-chip programmable logic.

Preliminary

~~2.4~~2.3 Audio Subsystem

The integrated Audio Subsystem shown in Figure 29 is designed to support always-on Audio capability. The EOS S3AI platform supports two types of digital microphones. Both types of microphones are supported in mono and stereo configuration.

Figure 29: Audio Subsystem Block Diagram



~~2.4.1~~2.3.1 PDM Microphone

The EOS S3AI platform supports PDM microphones in a mono or stereo configuration. As shown in Figure 29 the incoming PDM data is sampled using the PDM sampler. In case of a mono microphone, software can enable the Left or Right PDM-to-PCM (PDM2PCM) converter. When two microphones are used in stereo configuration, both Left and Right PDM2PCM converters are enabled. The output of the PDM2PCM converters are 16-bit PCM samples at a default 16-kHz sample rate.

Preliminary

~~2.4.2~~2.3.2 I2S Microphones

The EOS S3AI platform also supports mono or stereo I2S microphones, as shown in Figure 29. The I2S interface inside the EOS S3AI platform provides signals needed for interfacing to the microphones, and the outputs are 16-bit PCM samples. The output can be used as is or it can be multiplied by a factor of 2, 4, or 8. In the case where down sampling of the PCM sample is required, the PDM2PCM block can be used to down sample from a 32-kHz sample rate to a 16-kHz sample rate. In situations where incoming PCM samples lack acoustic fidelity, digital gain of up to 34.5 dB can be applied to the PCM samples by PDM2PCM block.

~~2.4.3~~2.3.3 Low Power Sound Detect (LPSD) Support

In order to minimize power associated with always on audio processing, the EOS S3AI platform supports acoustic activity detection using LPSD hardware. This allows normal PDM or I2S microphones from any vendor to get power savings associated with the LPSD hardware. When enabled, the EOS S3AI can send PCM samples from left or right microphone to the LPSD hardware. The logic inside the LPSD hardware is designed to detect a change in audio characteristics and wakeup the rest of the EOS S3AI device. The LPSD hardware is best used with PDM or I2S microphones and should not be enabled when using microphones that have dedicated acoustic detection capability.

~~2.4.4~~2.3.4 PDM Slave Port for External Codec

As shown in Figure 29, the EOS S3AI platform provides a PDM slave port that allows an external Application Processor (AP) to interface to the PDM microphones connected to the EOS S3AI device. When this port is enabled, the EOS S3AI essentially behaves transparently. The PDM clock driven by AP, received by the EOS S3AI on its PDM_CKO_IN input pin is driven out to PDM_CKO output pin of the EOS S3AI device. Similarly, PDM data received by the EOS S3AI device on its PDM_IN pin from PDM microphones is driven out on the EOS S3AI device PDM_IN_OUT pin without modification. There are several software-controlled modes, associated with this port. Depending on the mode and state of clock on PDM_CKO_IN pin, PDM_CKO can be driven by the EOS S3AI and PDM data can be consumed by the EOS S3AI.

~~2.4.5~~2.3.5 DMA and AHB Master Port

PCM samples from the microphones are stored in FIFOs as shown in Figure 29. There is a separate FIFO for each microphone. The DMA Controller (DMAC) inside the EOS S3AI platform is responsible for transferring the PCM samples from the FIFOs to M4F SRAM using EOS S3AI platform AHB-Lite master port.

~~2.4.6~~2.3.6 APB Slave Port

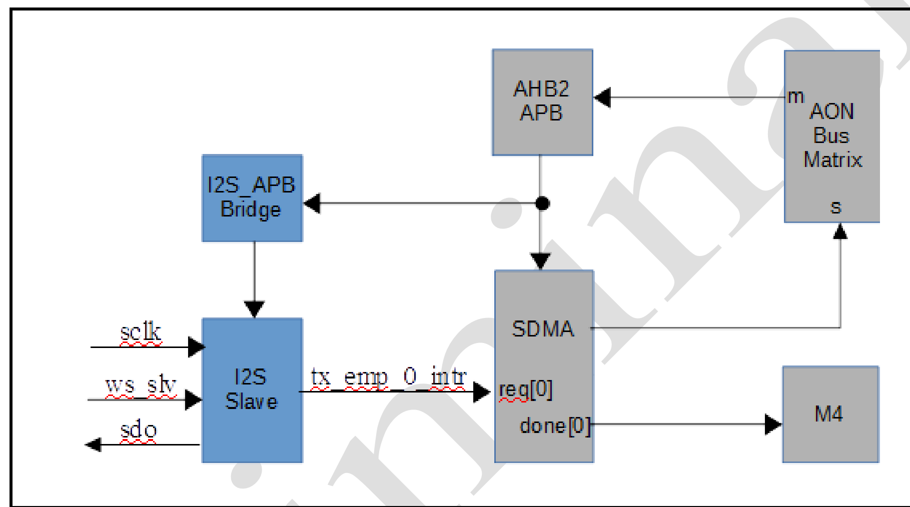
As shown in Figure 29, the EOS S3AI platform supports an APB slave port. The M4F can use this port to access state registers inside the EOS S3AI platform.

Preliminary

~~2.4.1~~2.3.1 I^{2}S Slave Port

The EOS S3 platform implements an I2S Slave port, allowing an external I2S Master to interface to EOS S3 platform, as shown in Figure 29. This port can transfer audio PCM samples from M4F SRAM to an external I2S Master, such as an Application Processor or a Codec. Channel 0 of System DMA is allocated for transferring PCM samples from M4F SRAM to I2S Slave port. Figure 30 illustrates connections between external I2S Master and EOS S3 platform I2S Slave port. It also shows connections between I2S Slave port, System DMA (SDMA) and other blocks inside EOS S3 platform.

Figure 30: I^{2}S Slave Port



Preliminary

~~2.5~~2.4 Timing

~~2.5.1~~2.4.1 I2C Master AC Timing

Figure 31 shows the I²C Master AC timing.

Figure 31: I²C Master AC Timing

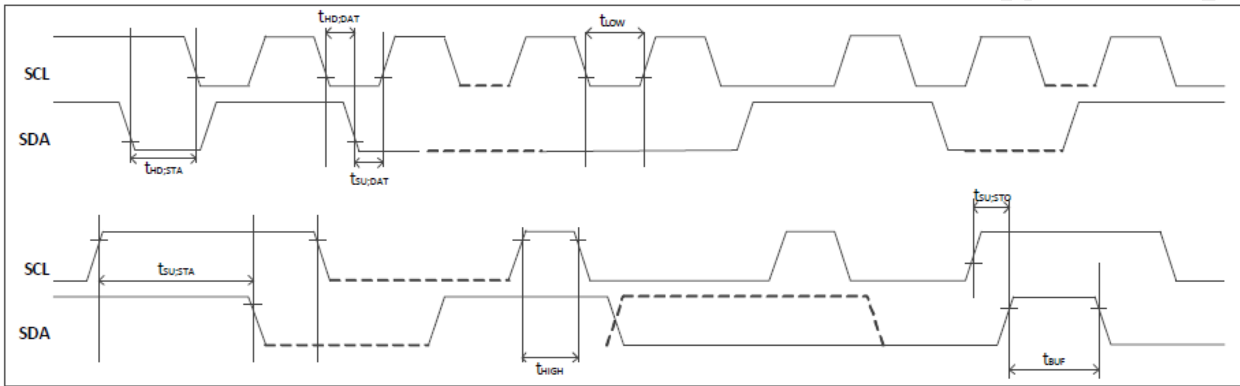


Table 2 describes the I²C Master AC timing parameters.

Table 2: I2C Master AC Timing

Symbol	Description	Standard Mode		Fast Mode		Units
		Min.	Max.	Min.	Max.	
f_{SCL}	Operating frequency.	-	100	-	400	kHz
t_{LOW}	Clock low period.	4.7	-	1.30	-	µS
t_{HIGH}	Clock high period.	4.0	-	0.60	-	µS
$t_{HD;STA}$	Hold time for repeated START condition.	3.10	-	0.60	-	µS
$t_{SU;STA}$	Setup time for repeated START condition.	4.19	-	0.60	-	µS
t_{BUF}	Bus free time between STOP and START condition.	4.7	-	1.3	-	µS
$t_{HD;DAT}$[a]	Data hold time.	0	-	0	-	µS
$t_{SU;DAT}$	Data setup time.	0.25	-	0.10	-	µS
$t_{SU;STO}$	Setup time for STOP.	4.0	-	0.6	-	µS

a. The receiving device must provide an internal delay of 300 nS for the SDA signal with respect to the SCL signal to bridge the undefined region of the falling edge of SCL.

~~2.5.2~~2.4.2 I2S Timing

Figure 32 and Table 3 describes the I^2S timing.

Figure 32: I^2S Timing Waveform

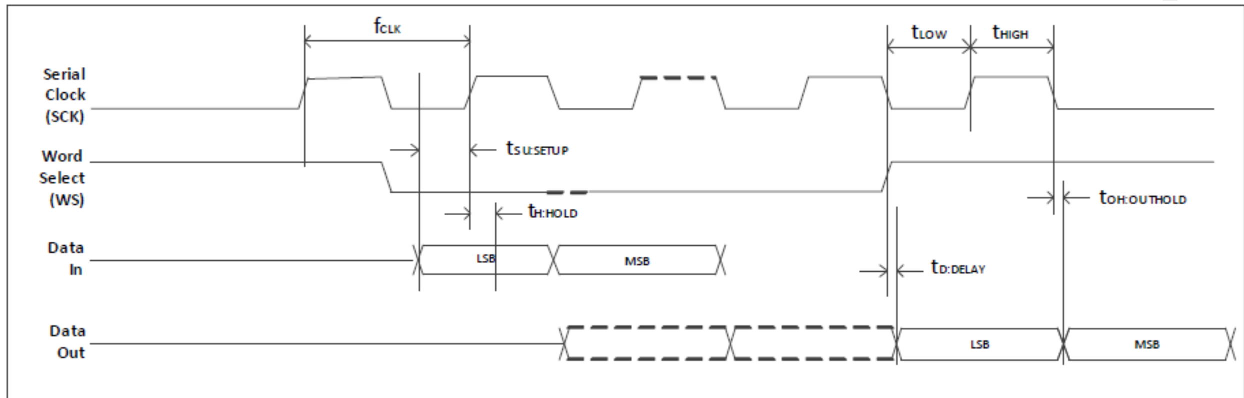


Table 3: I^2S Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
f_{CLK}	I^2S Clock Frequency	-	-	10	MHz
t_{LOW}	Clock High Period	45	-	-	ns
t_{HIGH}	Clock Low Period	45	-	-	ns
t_{SU}	Data Input Set Up Time	10	-	-	ns
t_H	Data Input Hold Time	1	-	-	ns
t_D	Clock to Data Out Delay	3	-	35	ns
t_{OH}	Clock to Out Hold	2	-	-	ns

Preliminary

~~2.5.3~~2.4.3 PDM Microphone Timing

Figure 33 and Table 4 describe the PDM microphone timing.

Figure 33: PDM Microphone Timing

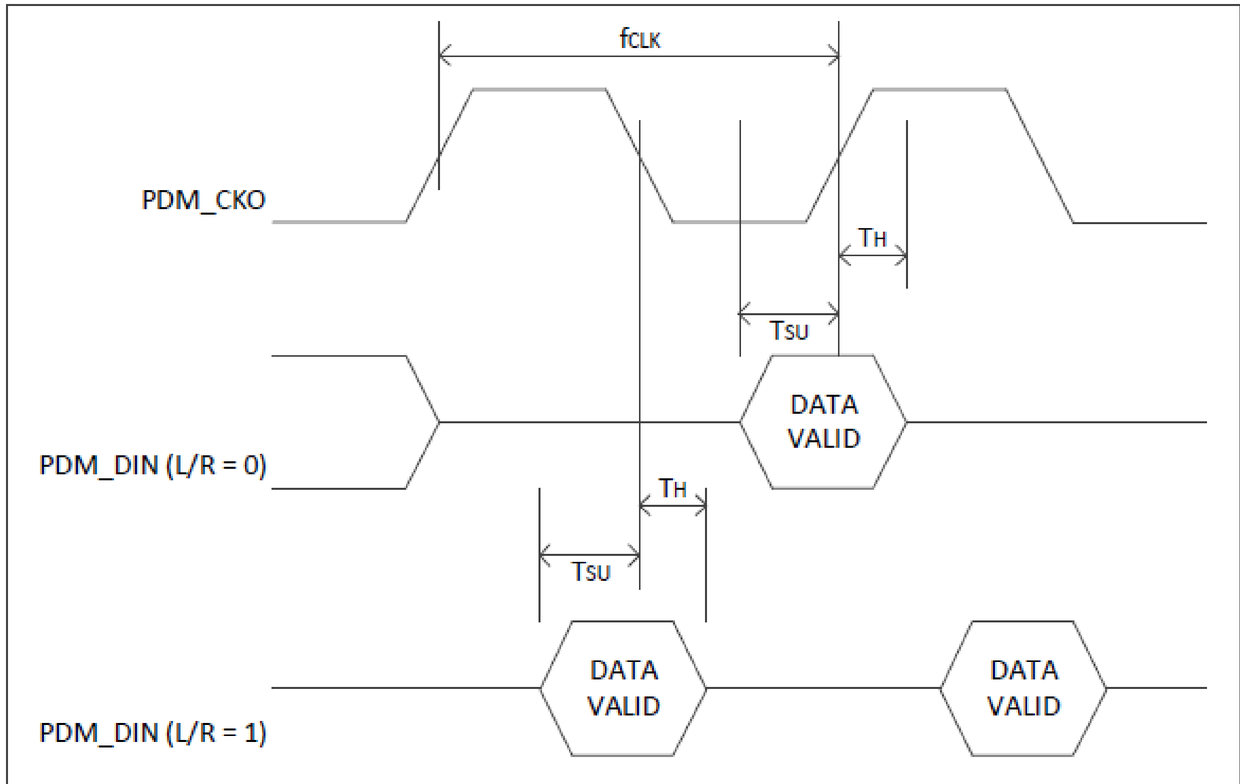


Table 4: PDM Microphone Timing

Symbol	Parameter	Min.	Typ.	Max.	Units
f_{CLK}	PDM Frequency	-	-	10	MHz
t_{SU}	Data Input Set Up Time	10	-	-	ns
t_H	Data Input Hold Time	1	-	-	ns
	f_{CLK} Duty Cycle	48	50	52	%

Prel

~~2.5.4~~2.4.4 SPI Timing

~~2.5.4.1~~2.4.4.1 SPI Master

Figure 34 illustrates the SPI Master timing.

Figure 34: SPI Master AC Timing

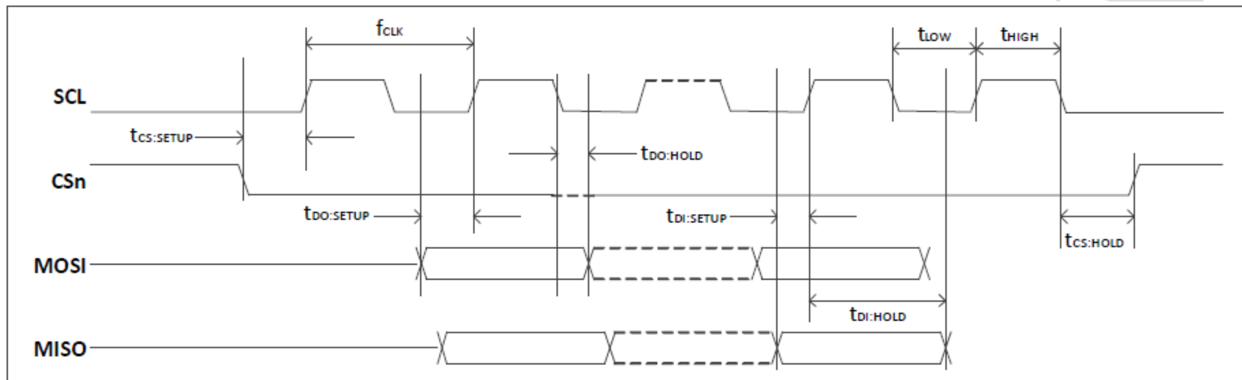


Table 5: SPI Master Timing

Symbol	Parameter	Min.	Max.	Units
f_{CLK}	SPI Clock Frequency	-	20	MHz
t_{LOW}	SPI CLK Low Time	24	-	nS
t_{HIGH}	SPI CLK High Time	24	-	nS
$t_{DO:SETUP}$	Data Output Setup Time to slave device	$(2/ f_{CLK})-5$	-	nS
$t_{DO;HOLD}$	Data Output Hold Time	1.1	4.8	nS
$t_{DI:SETUP}$	Data Input Setup Time	8	-	nS
$t_{DI;HOLD}$	Data Input Hold Time	1	-	nS
$t_{CS:SETUP}$	CS Input Setup Time	50	-	nS
$t_{CS;HOLD}$	CS Input Hold Time	50	-	nS

Preliminary

2.5.4.22.4.4.2 SPI Slave

Figure 35 illustrates the SPI Slave timing.

Figure 35: SPI Slave Timing

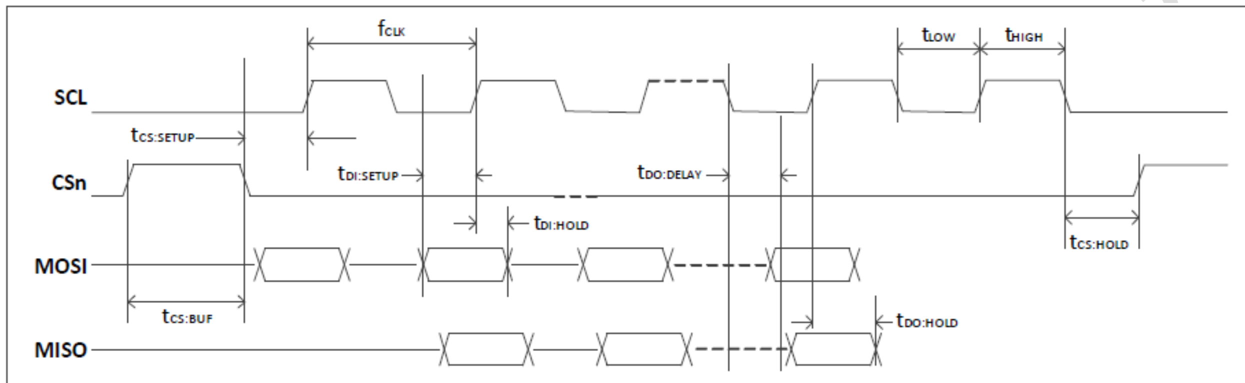


Table 6: SPI Slave Timing

Symbol	Parameter	Min.	Max.	Units
f_{CLK}	SPI Clock	-	20	MHz
t_{LOW}	SPI Clock Low Time	22.5	-	nS
t_{HIGH}	SPI Clock High Time	22.5	-	nS
$t_{DO:SETUP}$	MISO Output Delay from SPI Clock Driving Edge	-	16	nS
$t_{DO;HOLD}$	MISO Output Delay Hold Time	2	-	nS
$t_{DI:SETUP}$	MOSI Input Setup Time	4	-	nS
$t_{DI;HOLD}$	MOSI Input Hold Time	4	-	nS
$t_{CS:SETUP}$	CS Input Setup Time	4	-	nS
$t_{CS;HOLD}$	CS Input Hold Time	4	-	nS
$t_{CS;BUFF}$	CS High Time	50	-	nS

~~2.6~~2.5 On-Chip Programmable Logic

The on-chip programmable fabric provides flexibility to the EOS S3AI platform for implementing additional functions. The on-chip programmable logic consists of multiplexor based logic cells, built-in RAM modules and FIFO controllers, built-in multipliers, as well as interfaces with I/O drivers of the EOS S3AI device. The major features of the on-chip programmable logic are listed in Table 7.

Table 7: on-chip programmable logic major features

Feature	
Logic Cells	891
8K RAM Modules (9216 bits)	8
FIFO Controllers	8
RAM bits	73,728
Configurable Interface	32
Multiplier	2x 32 x 32
	4x 16 x 16

~~2.6.1~~2.5.1 Functional Description

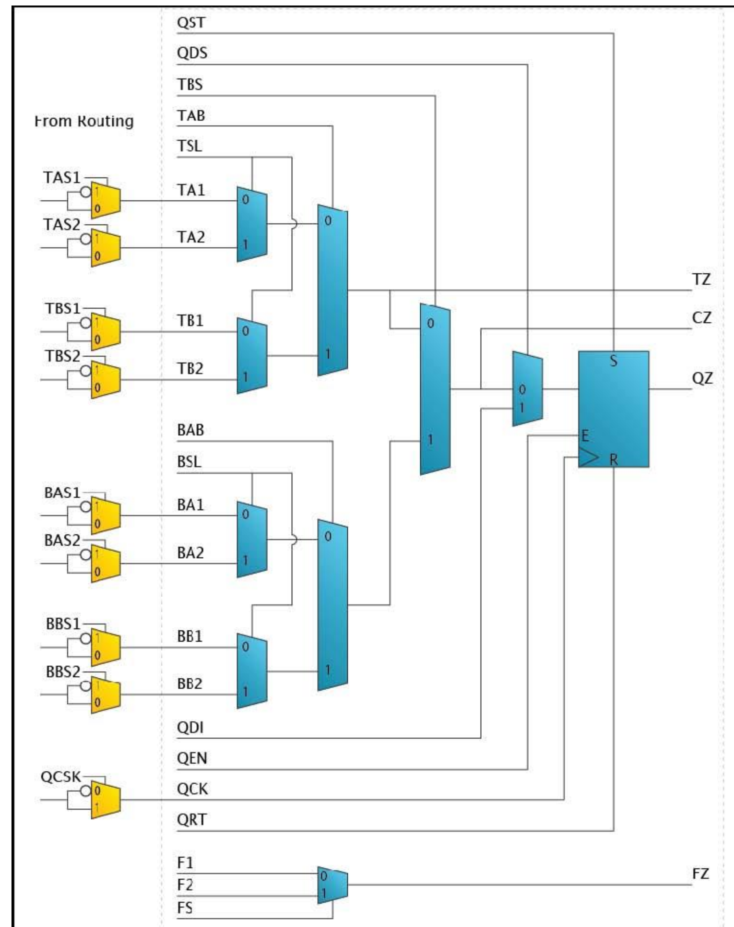
~~2.6.1.1~~2.5.1.1 Logic Cell

Each logic cell is a multiplexer-based single register. The cell has a high fan-in, and fits a wide range of functions with up to 22 simultaneous inputs (including register control lines), and four outputs (three combinatorial and one registered). Figure 36 illustrates the logic block structure. The high logic capacity and fan-in of the logic cell accommodates many user functions with a single level of logic delay. The logic cell is capable to implement the following functions:

- Two independent 3-input functions
- Any 4-input function
- 8 to 1 mux function
- Independent 2 to 1 mux function
- Inverted or non-inverted clock signal to flip-flop
- Single dedicated register with active high clock enable, set and reset signals
- Direct input selection to the register, which allows combinatorial and register logic to be used separately
- Combinatorial logic can also be configured as an edge-triggered master-slave D flip-flop

Preliminary

Figure 36: Logic Cell block diagram



~~2.6.1.2~~2.5.1.2 RAM/FIFO

On-chip programmable logic also includes up to eight 8 kilobits (9,216 bits) dual-port RAM modules for implementing RAM and FIFO functions.

RAM features include:

- Independently configurable read and write data bus widths
- Independent read and write clocks
- Inverted or non-inverted clock signals to read and write clock inputs
- Horizontal and vertical concatenation
- Write byte enables
- Selectable pipelined or non-pipelined read data

Preliminary

2.6.1.3 2.5.1.3 FIFO Controller

Every 8K RAM block can also be implemented as a synchronous or asynchronous FIFO. There are built-in FIFO controllers that allow for varying depths and widths without requiring on-chip programmable logic resources. During asynchronous operation, the FIFO works in a half-duplex fashion such that PUSH is on one clock domain and POP is on another clock domain. The DIR signal allows the FIFO PUSH and POP signal directions to reverse.

FIFO controller features include:

- x9, x18 and x36 data bus widths
- Independent PUSH and POP clocks
- Independent programmable data width on PUSH and POP sides
- Configurable synchronous or asynchronous FIFO operation
- 4-bit PUSH and POP level indicators to provide FIFO status outputs for each port
- Pipelined read data to improve timing
- Option for inverted or non-inverted asynchronous flush input

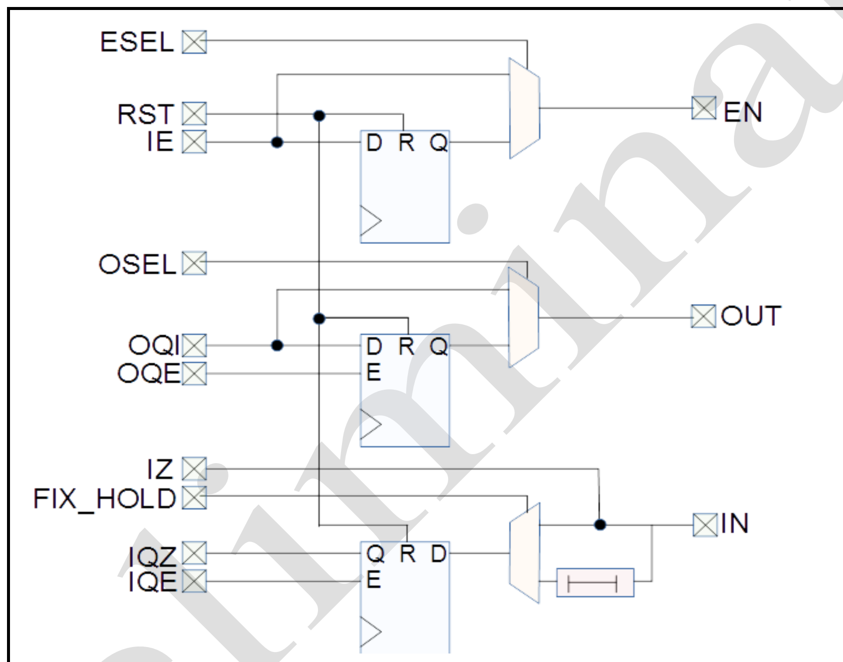
Preliminary

~~2.6.1.4~~2.5.1.4 Configurable Input/Output Signals

Configurable I/O is used to provide additional functionality prior to driving the actual GPIO. The additional functionality is described in the following:

- Register path versus non-register path for IN, OUT, and EN
- FIX_HOLD feature to improve hold time

Figure 37: On-chip programmable logic Configurable Input/Output



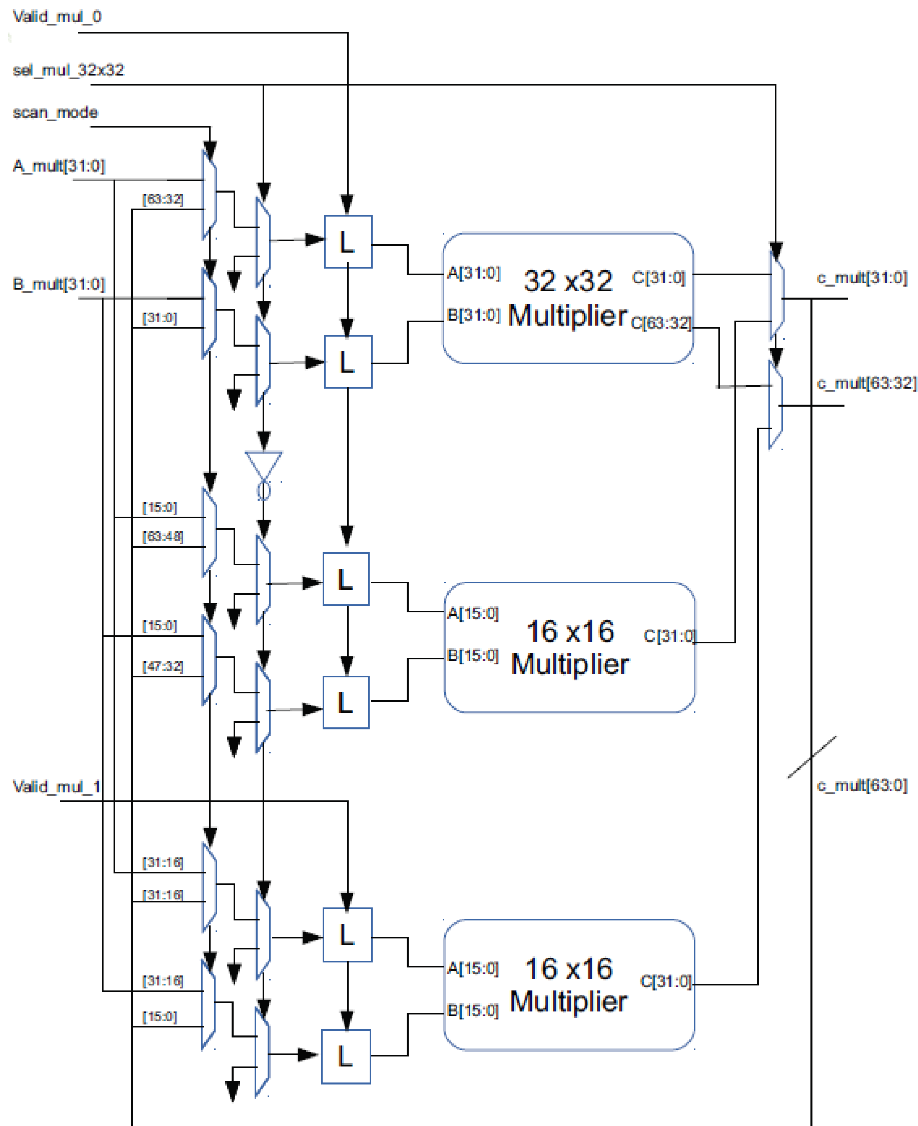
Preliminary

~~2.6.1.5~~2.5.1.5 Multipliers

Built-in signed multipliers are also available in the On-chip programmable logic. The multiplier relieves the use of logic to implement such functions. There are total of 2 such instances embedded in the On-chip programmable logic. The multiplier can be configured as one 32x32 bit multiplier or two 16x16 bit multipliers.

The block diagram of the multiplier is shown in Figure 38.

Figure 38: On-chip programmable logic Multiplier



~~2.6.2~~2.5.2 Interface to the On-chip programmable logic

IP within the On-chip programmable logic can use the following interfaces to communicate with resources outside of the fabric. These include:

- EOS S3AI platform
- SPI Master Interface for System Support
- Sensor Processing Subsystem
- Packet FIFO

These resources help the IP to coordinate its activities with other modules in the EOS S3AI platform. Additionally, the IP can call upon external resources to support its processing activities. The following sections will cover what resources are available to IP within the On-chip programmable logic.

It is important to note that it is not essential that the IP within the On-chip programmable logic use these interfaces.

~~2.6.3~~2.5.3 EOS S3AI Platform Interface

The interface between IP in the Programmable On-chip programmable logic and the EOS S3AI platform consists of the following:

- Data transfer interface via an AHB-To-Wishbone Bridge
- SDMA Interface
- Interrupt Interface

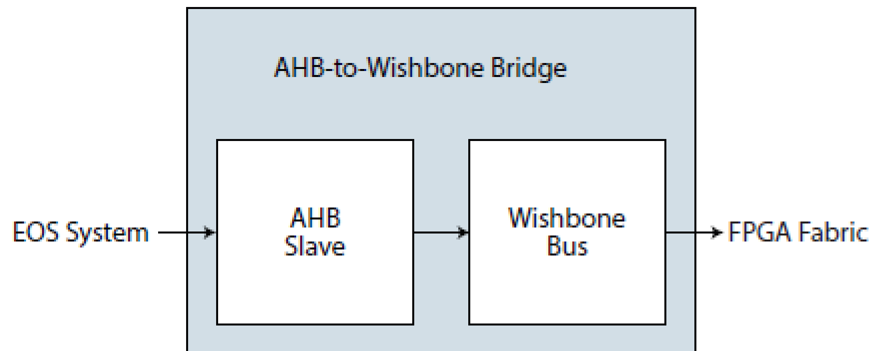
Each of these interfaces is discussed in the following sections.

~~2.6.3.1~~2.5.3.1 AHB-To-Wishbone Bridge

The AHB-To-Wishbone Bridge provides the means for the EOS S3AI platform (ex. M4F or AP) to access IP within the On-chip programmable logic. Specifically, this interface takes a 32-bit address and data on its AHB port and passes these values to the Wishbone bus.

Preliminary

Figure 39: AHB-To-Wishbone Bridge



The connection between the AHB Slave and Wishbone Bus supports asynchronous transfers. This allows the On-chip programmable logic based IP to use a clock frequency appropriate to its operation without the EOS S3AI platform losing its ability to communicate with the resources of the IP. For example, most IP's require the use of some type of "Start" or "Stop" register bit to control their operations. The asynchronous interface insures that the EOS S3AI platform can still access these registers.

It is important to note that the On-chip programmable logic based IP does not have the ability to initiate direct transfers to the EOS S3AI platform. This is done for two reasons: first, The he Wishbone bus cannot support multiple masters (the Wishbone interface only supports Wishbone clients) and second, the AHB Slave interface cannot master the AHB bus.

~~2.6.3.2~~2.5.3.2 SDMA Interface

The EOS S3AI platform provides a System DMA (SDMA) module for use by various components. The purpose for the SDMA function is to both avoid loading the Host processor with simple data movement operations, and to conduct data movement during periods when the Host Processor is in a low power state.

For the IP in the On-chip programmable logic, the SDMA interface provides the means to process a block of data and transfer the results to M4F memory. Examples of this include:

- IP uses SDMA to move data from M4F memory, processes the data, and initiates an SDMA move of the finished result back to M4F memory.
- IP reads data from an external sensor (e.g. through the SPI Master for System Support), processing the data, and using an SDMA operation to save the results to M4F memory.

Once the SDMA operation completes, an interrupt from the SDMA can signal to the Host that the operation has completed. Alternatively, the IP in the On-chip programmable logic can also issue its own interrupt to the EOS S3AI platform to signal the completion of its processing operation.

Preliminary

It is important to note that the SDMA module resides at the EOS S3AI platform level. Therefore, it can access both the M4F Memory modules and the On-chip programmable logic IP. The IP does not have the ability to directly access the M4F Memory modules.

~~2.6.3.3~~2.5.3.3 Interrupt Interface

The EOS S3AI platform provides a set of interrupt signals to the On-chip programmable logic IP. These interrupt signal enable the IP to signal to the EOS S3AI platform that it needs attention. The nature of this attention will depend upon the IP and the nature of the required processing.

~~2.6.3.4~~2.5.3.4 Sensor Processing Subsystem Interface

The Sensor Processing Subsystem has the ability to drive IP residing in the On-chip programmable logic with a "Start" signal and sample a "Busy" status signal. These signals enable the IP to coordinate its processing with that of the Sensor Processing Subsystem. In doing, so both modules can provide the processed data to the EOS S3AI platform for further evaluation or Sensor Fusion processing.

It is important to note that there is no direct connection from the Sensor Processing Subsystem to the On-chip programmable logic for the purpose of passing data that either processed or is raw. Instead, the data must first be passed to the EOS S3AI platform (ex. M4F Memory) before it can be passed to either the Sensor Processing Subsystem (ex. from the On-chip programmable logic IP) or to the On-chip programmable logic IP (e.g. from the Sensor Processing Subsystem).

~~2.6.3.5~~2.5.3.5 Packet FIFO Interface

IP within the Programmable On-chip programmable logic has the ability to pass data to the EOS S3AI platform via the Packet FIFO interface. Like the Sensor Processing Subsystem, the IP logic can write raw or processed data into the Packet FIFO in a software-defined packet format.

It is important to note that the Sensor Processing Subsystem and On-chip programmable logic based IP do not share the same connection to the Packet FIFO. Rather, each module uses a separate port to the Packet FIFO. However, within the Packet FIFO, each port must be assigned by software to a separate internal FIFO (ex. the Packet FIFO is composed of sub-FIFO 8K, 0, 1, and 2.).

Preliminary

~~2.7~~2.6 Power Management

~~2.7.1~~2.6.1 Power Supply Modes and Schemes

In the EOS S3AI platform, there are 31 power islands including the always-on domain. All of the power domains can be independently powered on and off by power management logic, with the exception of the always-on power domain. This allows for additional power savings using software and hardware control through Power Management Unit (PMU) registers.

Out of reset, the following domains are powered ON by default:

- M4F Subsystem
- Configuration DMA / SPI Flash controller (A1)
- 16 M4F SRAM (32x8K) domains
- Always-on domain (Cannot be powered off)

The other domains are set to OFF by default after reset release. Refer to Table 8 for a listing of switchable power domains. Additionally, some power domains also have a retention power rail, allowing for retention of state within that domain. This capability allows the main power supply to be turned off to that domain, while still keeping the retention power supply on. In addition to an on/off power state, some power domains my support a retention mode. When in the retention mode, the power domains can retain the state information, but may not be accessible for read/write operations.

Table 8 shows the list of power domains, default power states and retention modes.

Preliminary

Table 8: Power Domains

No.	Power Domain	Default Power State	On/Off Mode	Retention Mode
1	On-Chip Programmable Logic	Off	Yes	Yes
2	M4F Subsystem (M4)	On	Yes	No
3	Packet FIFO Bank (PKFB)	Off	Yes	Yes
4	Configuration DMA / SPI Flash (A1)	On	Yes	Yes
5	System DMA (SDMA)	Off	Yes	Yes
6	Flexible Fusion Engine (FFE)	Off	Yes	Yes, Partial
7	I^2S Slave	Off	Yes	No
8	32x8K SRAM Instance 0	On	Yes	Yes
9	32x8K SRAM Instance 1	On	Yes	Yes
10	32x8K SRAM Instance 2	On	Yes	Yes
11	32x8K SRAM Instance 3	On	Yes	Yes
12	32x8K SRAM Instance 4	On	Yes	Yes
13	32x8K SRAM Instance 5	On	Yes	Yes
14	32x8K SRAM Instance 6	On	Yes	Yes
15	32x8K SRAM Instance 7	On	Yes	Yes
16	32x8K SRAM Instance 8	On	Yes	Yes
17	32x8K SRAM Instance 9	On	Yes	Yes
18	32x8K SRAM Instance 10	On	Yes	Yes
19	32x8K SRAM Instance 11	On	Yes	Yes
20	32x8K SRAM Instance 12	On	Yes	Yes
21	32x8K SRAM Instance 13	On	Yes	Yes
22	32x8K SRAM Instance 14	On	Yes	Yes
23	32x8K SRAM Instance 15	On	Yes	Yes
24	Audio – DMA (AD0)	Off	Yes	No
25	Audio – PDM Left (AD1)	Off	Yes	No
26	Audio – PDM Right (AD2)	Off	Yes	No
27	Audio – LPSD (AD3)	Off	Yes	No
28	Audio – I^2S (AD4)	Off	Yes	No
29	Audio – TOP (AD5)	Off	Yes	No
30	Always-On	On	No	No
31	eFuse	On	Yes	No

2.7.1.12.6.1.1 SRAM Power domains

There are 16 32KByte SRAM instances and the power domain of each instance can be controlled independently. Other SRAM instances do not have independently controlled power domains; their power domain is bonded with logic block they reside. Additionally, the 16 32KByte SRAM instances utilize auto-wakeup logic, which allows the SRAM to been in a shutdown or deep sleep state when not in use.

NOTE: The EOS S3AI platform software puts the SRAM into sleep when it is not in use.

The SRAM instances on FFE and Packet FIFO Bank (PKFB) are on the corresponding retention power supply/domain. For other SRAM instances (such as Audio), they are on the corresponding ON/OFF domain. Please refer to Table 9. Each SRAM memory instance itself can also support deep sleep and shut down modes for additional power savings.

Preliminary

Table 9: Memory Domains

SRAM Power Domain	Size of SRAM Macro	Power Supply Net
PKFB SRAM	0.25Kx32 (Instance 0)	Retention Power for PKFB
	0.25Kx32 (Instance 1)	
	0.5Kx32 (Instance 0)	
	4Kx17 (Instance 0)	
FFE SRAM	CM 10Kx40	Retention Power for FFE
	DM 4Kx32	
	SM0 1Kx18	
	SM1 0.5Kx18	
PDM Left SRAM	L0 2Kx32	Main Power for Audio AD1
	L1 128x32	
	L2 256x16	
PDM Right SRAM	R0 2Kx32	Main Power for Audio AD2
	R1 128x32	
	R2 256x16	

~~2.7.1.2~~2.6.1.2 Low Drop-out Regulators

EOS S3AI platform has two on-chip low drop-out regulators (LDOs). One LDO is for SRAM, and the other LDO is for digital logic. By having a separate regulator for the SRAM, the SRAM voltage can be further reduced to save power consumption.

Table 10: LDO Regulators

LDO	Max Current	Output Voltage Range
LDO-1 for SRAM	50mA	0.1V ~ 1.20V
LDO-2 for Digital Logic	30mA	0.90V ~ 1.20V

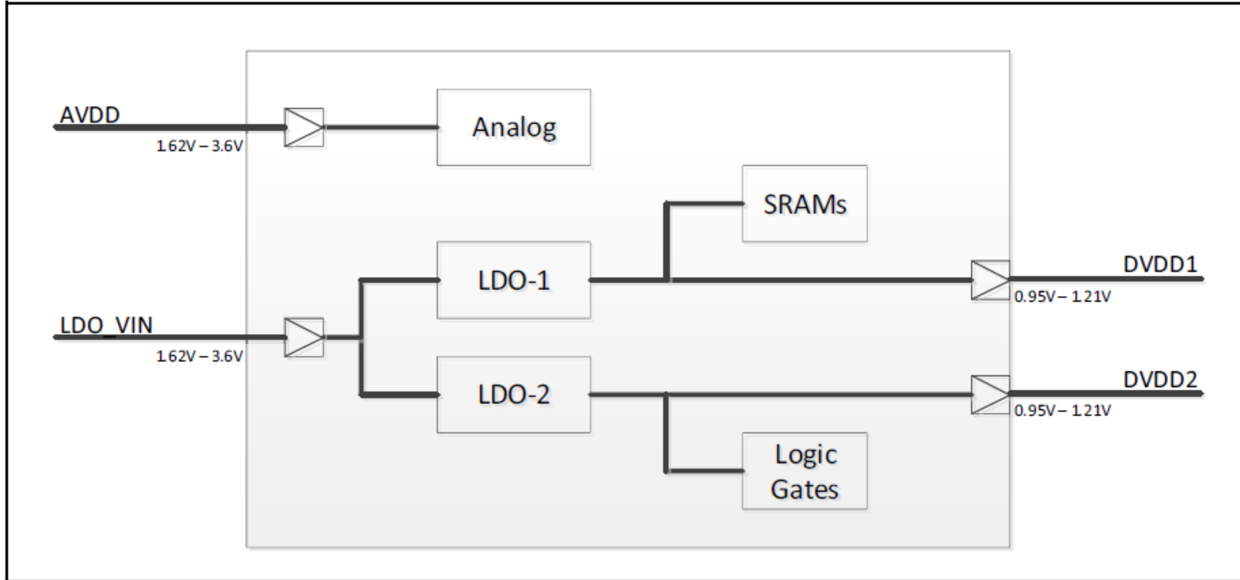
There are three possible LDO use cases for EOS S3AI platform, which allows customer flexibility in configuration. Each is illustrated in Figure 40 to Figure 42.

Preliminary

~~2.7.1.2.1~~2.6.1.2.1 LDO Use Case 1: Dual Voltage Rail Supplied by On-chip LDOs

In this example, all SRAMs is supplied by LDO-1 and Logic Gates is provided by LDO-2.

Figure 40: LDO User Case 1 - Dual Voltage Rail



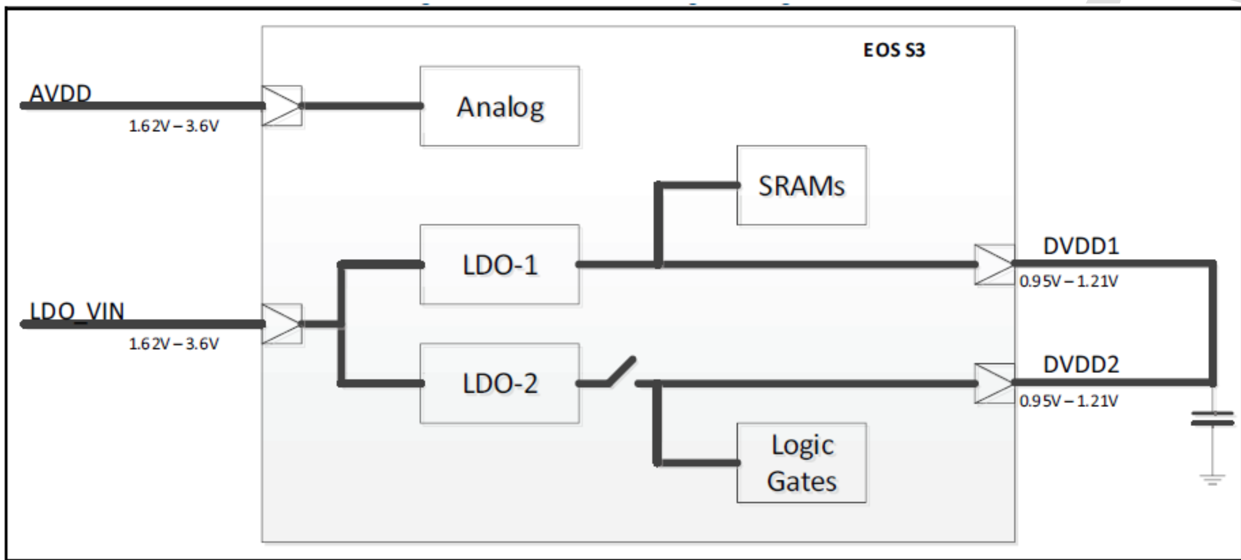
Preliminary

2.7.1.2.22.6.1.2.2 LDO Use Case 2: Single Voltage Rail Supplied by Single On-chip LDOs

In this example, the SRAMs and Logic Gates are provided by LDO-1. The DVDD2 and DVDD1 pads are tied together on the board. After voltage is applied to LDO_VIN, both LDO-1 and LDO-2 are on. To further reduce the device core power consumption, turn off LDO-2 with firmware.

NOTE: LDO-1 supports up to 50mA loading.

Figure 41: LDO Use Case 2 - Single Voltage Rail



Preliminary

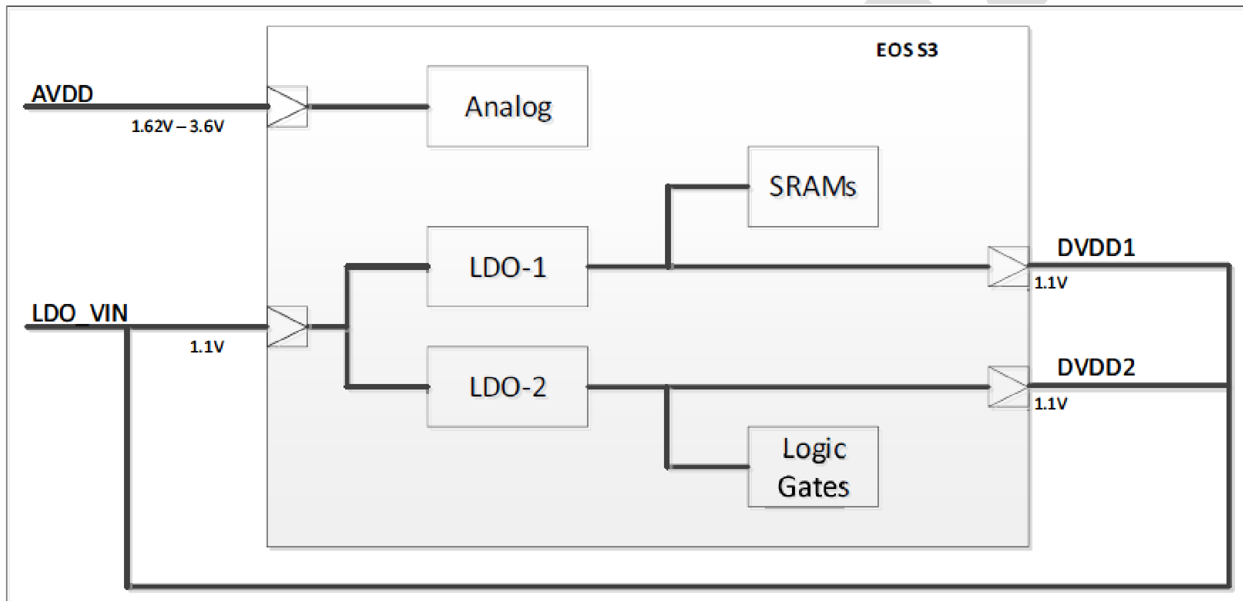
~~2.7.1.2.3~~2.6.1.2.3 Use Case 3: External Voltage Supplied

In this example, the EOS S3AI platform LDOs are bypassed and SRAMs and Logic Gates power are externally supplied through DVDD1 and DVDD2 pads. The LDO_VIN, DVDD2 and DVDD1 pads are tied together on the board and hook up to the supply voltage.

NOTE: In this configuration, an external voltage of 1.1V±50mV is required to be applied to LDO_VIN, DVDD1 and DVDD2. The device requires 1.05V minimum to come out of Power-On reset. See DC Specification section for more details.

For bypass configuration, external power supplies are used. Therefore, turn off the internal LDOs with firmware to further reduce the device core power consumption.

Figure 42: LDO Use Case 3 - External Voltage Supplied



Preliminary

~~2.7.2~~2.6.2 Power-On Sequence

The recommend power-on sequence for EOS S3AI platform is shown in **Error! Reference source not found.**.

Figure 43: Power-On Sequencing

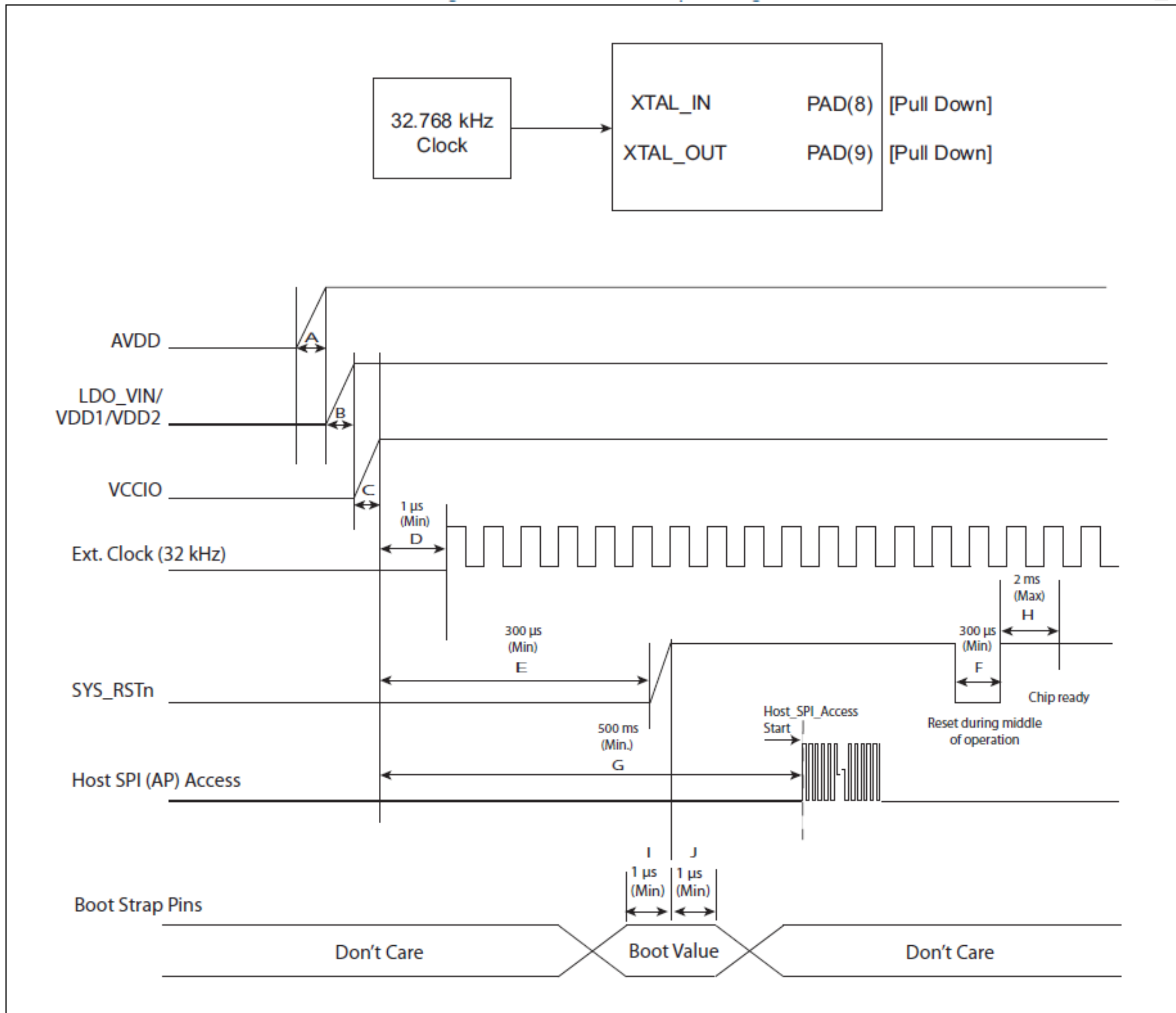


Table 11 shows the Power-On sequence timing.

Table 11: Power-On Sequencing Timing Parameters

Letter	Parameter	Condition	Min.	Typ.	Max.	Unit
A	AVDD Voltage Rising Time	From 0V to Target operating voltage	See Note	See Note	See Note	ms
B	LDO_VIN/VDD1/VDD2 Rising Time	From 0V to Target operating voltage	See Note	See Note	See Note	ms
C	VCCIOA/VCCIOB Rising Time	From 0V to Target operating voltage	See Note	See Note	See Note	μs
D	Voltage Ready to XTAL_IN (CMOS CLK)	All voltage rails at 90% voltage to CMOS CLK	1	-	-	μs
E	Voltage Ready to SYS_RSTn Release	All voltage rails at 90% voltage to SYS_RSTn is released	300	-	-	μs
F	SYS_RSTn Middle of the Operation	Host SPI Access is available 2 ms after SYS_RSTn is released	300	-	-	μs
G	Voltage Ready to HOST_SPI_Access Start	Conditions A through F must be met	500	-	-	ms
H	System Reset Release to Chip Exits Reset State	Chip ready 2 ms after SYS_RSTn release	-	-	2	ms
I	Bootstrap pins setup time	Setup time with respect to SYS_RSTn	1	-	-	μs
J	Bootstrap pins hold time	Hold time with respect to SYS_RSTn	1	-		μs
	CMOS CLOCK is active after AVDD is powered down[a]	CMOS CLOCK is required to be active for a minimum time if AVDD is powered down	2.0	-		ms

a. This is only required for a system that has intermittent power-down interruption. It is not required for AVDD always-on or CMOS CLOCK always-on.

NOTE**:** There is no special power sequence and power rails ramp time requirement. However, avoid power supply sequencing VCCIO supply voltage before VDD1/VDD2 core voltage to prevent contention with other signals in the system (including SYS_RSTn signal), refer to **Table 29** for more information.

NOTE**:** Device initiation can begin after SYS_RSTn timing is met. SYS_RSTn signal is held low before power rails reach 90%. Initialization before SYS_RSTn timing is met can result in bootup failure. There is a weak pull-up inside SYS_RSTn, see **Table 32** for the resistance value.

NOTE**:** To avoid high current during power-on sequence, VDD must not be powered on before AVDD. The recommended sequence is AVDD–>VDD–>VCCIO.

Preliminary

Figure 44: Current Measurement Scheme with External Power Supplies

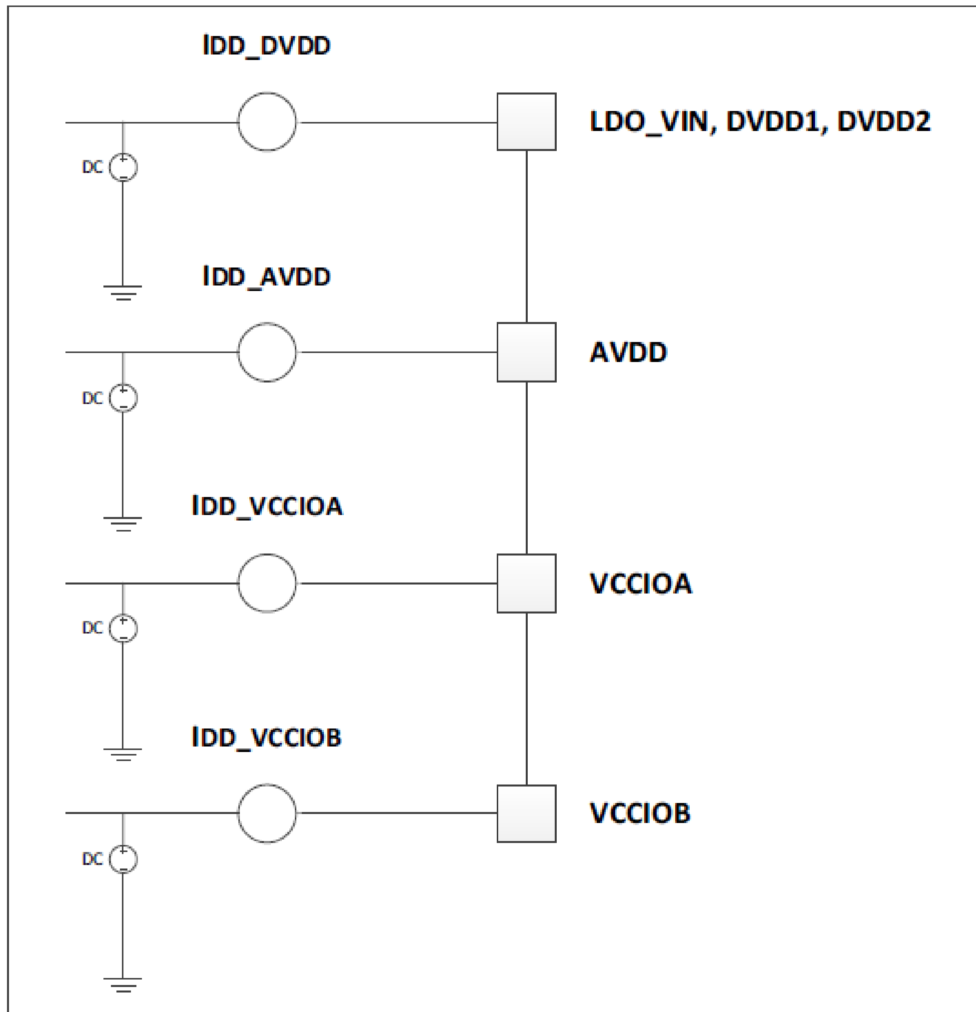


Table 12 and Table 13 show the current measurement on each supply rail for power-on sequence in LDO bypass mode (external voltage supply). All I/Os and dedicated pins are tri-stated, and all external caps are removed.

Preliminary

Table 12: Current Measurements for Power-On Sequence[a]

Power Sequence	1 DVDD (1.1V)	2 AVDD (1.8V)	3 VCCIOA (1.8V)	4 VCCIOB (1.8)
IDD DVDD	275 µA	147 µA	147 µA	183 µA
IDD VCCIO A	0.01 µA	0.01 µA	0.05 uA	0.03 uA
IDD VCCIO B	0.01 µA	0.01 µA	0.01 uA	0.3 uA
IDD AVDD	-60 µA	19.8 µA	19.8 µA	19.8 µA

a. Typical values are based on 25°C and nominal voltage (VDD1=VDD2=1.1V, VCCIO=1.8V)

Table 13: Current Measurements for Power-On Sequence[a]

Power Sequence	1 AVDD (1.8V)	2 DVDD (1.1V)	3 VCCIOB (1.8V)	4 VCCIOA @ 1.8V
IDD DVDD	-0.08 µA	147 µA	147 µA	183 uA
IDD VCCIO A	0.01 µA	0.01 µA	0.05 uA	0.03 uA
IDD VCCIO B	0.01 µA	0.01 µA	0.01 uA	0.3 uA
IDD AVDD	20.0 µA	19.8 uA	19.9 uA	19.8 uA

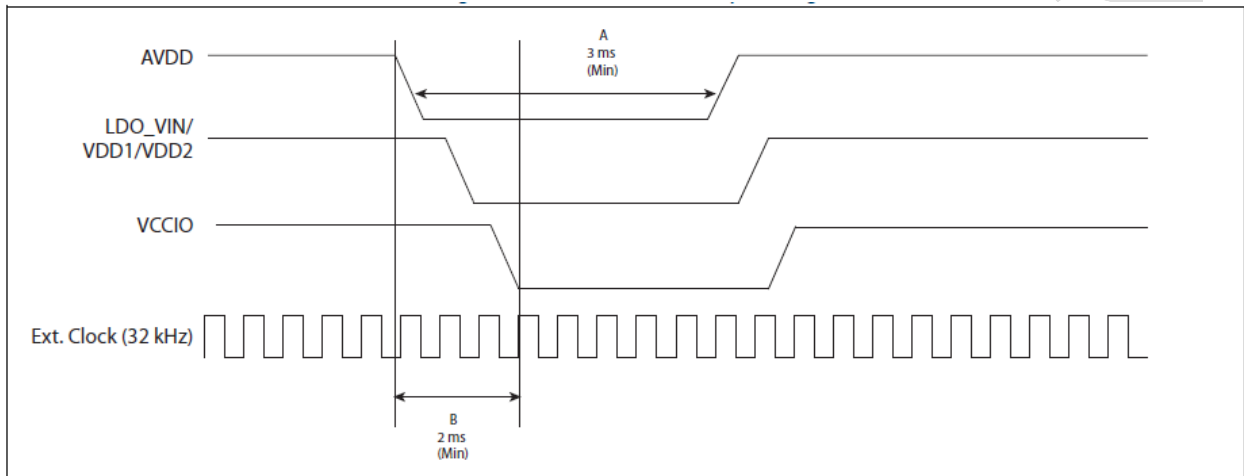
a. Typical values are based on 25°C and nominal voltage (VDD1=VDD2=1.1V, VCCIO=1.8V).

Preliminary

~~2.7.1~~2.6.1 Power-Down Sequence

The recommend power-down sequence for the EOS S3AI platform is shown in Figure 45. This is only required for a system that has intermittent power-down interruption for battery savings. It is not required for AVDD always-on or CMOS CLOCK always-on.

Figure 45: Power-Down Sequencing



Note: There is no special power down sequence and power rails ramp time requirement with the exception of AVDD, refer to table below for AVDD.

Table 14 shows the power-down sequencing timing parameters.

Table 14: Power-Down Sequencing Timing Parameters

Letter	Parameter	Condition	Min.	Typ.	Max.	Unit
A	AVDD Voltage Power-Down Duration Time	AVDD Voltage Power-Down Duration Time	3.0	-	-	ms
B	CMOS CLOCK is active after AVDD is powered down[a]	CMOS CLOCK is required to be active for a minimum time if AVDD is powered down	2.0	-	-	ms

a. The requirement applies for a system that has intermittent power-down interruption. It is not requried for AVDD always-on or CMOS CLOCK always-on.

Preliminary

Table 15 shows the current measurement of voltage rail after power-down with active CMOS clock.

Table 15: Current Measurement for Power-Down [a]

Power Sequence	AVDD 1.8V	VDD 1.1V	VCCIOB 1.1V	VCCIOA 1.1V
IDD with CMOS clock active	-2.8 mA	0	0	0

a. Typical values are based on 25◦C and nominal voltage (VDD1 = VDD2 = 1.1V, VCCIO = 1.8V) Table 16 and Table 17 shows the typical inrush current for each mode with one voltage rail power up.

Table 16: LDO Mode Typical Inrush Current

Mode	Data (mA)
AVDD @ 1.8V	0.013
VCCIOA @ 1.8V	3.662
VCCIOA @ 3.3V	65.640
VCCIOB @ 1.8V	51.630
VCCIOB @ 3.3V	114.000
LDOVIN @ 1.1V	4.000

a. All IOs are tri-stated or not driven, including SYS_RSTn

Table 17: LDO Bypass Mode Typical Inrush Current

Mode	Data (mA)
AVDD @ 1.8V	0.020
AVDD @ 3.3V	0.023
VCCIOA @ 1.8V	7.598
VCCIOA @ 3.3V	77.610
VCCIOB @ 1.8V	42.550
VCCIOB @ 3.3V	269.200
LDOVIN @ 1.1V	4.144

Table 18: Maximum Supply Power Consumption

Mode	Data (mA)
LDO Mode @ 1.8V	80
LDO Bypass Mode @ 1.1V	80

Preliminary

~~2.7.2~~2.6.2 Clocks and Resets

~~2.7.2.1~~2.6.2.1 Clocks

The EOS S3 platform contains 19 clock domains, and most clock domains have their own register-controlled divider. Each clock domain has one or more clock paths that it supports. Clock paths can be individually gated. See Table 19 for a full listing of the clock domains.

Table 19: Clocks Listing

Block Name(s)	Maximum Frequency[a]	Clock Name	Clock Path	Notes
Always-On Domain				
AP to SPI_Slave, SPI_Slave to TLC, TLC to PKT_FIFO clock	20 MHz	C00	P0_A0	
TLC clock to AHB Switch (AHB clock)	10 MHz	C01	P0_A0	The AHB clock must be greater than or equal to one half of the SPI Slave clock (in line above).
AHB Switch, Reg Bank, other blocks connected to the switch on the Always-on power domain	10 MHz	C01	P0_A0	
A1 Domain				
CfgSM, CfgDMA, SPI_Master (APB clock)	40 MHz	C02	P0_A1	
CfgDMA (AHB clock)	10 MHz	C01	P4_A1	
SPI_Master serial data clock	20 MHz	C02	n/a	The SPI_Master serial clock frequency is one half of the C02 clock frequency.
I²S Domain				
I²S Slave (DA pin)	10 MHz	C32	P0_I2S	
I²S Slave APB interface	10 MHz	C01	P5_I2S	
SDMA Domain				
AHB2APB, SDMA (AHB clock)	10 MHz	C01	P6_SDMA	
SDMA SRAM Domain				
SDMA SRAM	10 MHz	C01	P1_A0	
FFE Domain				
AHB switch	10 MHz	C01	P3_FFE	
X1 clk	10 MHz	C08	X1_P0_FFE	
X4 clk	40 MHz	C08	X4_P0_FFE	
For A0	10 MHz	C08	X1_P2_A0	
For PKT FIFO	10 MHz	C08	X1_P3_PF	

Preliminary

Table 19: Clocks Listing (*continued*)

Block Name(s)	Maximum Frequency[a]	Clock Name	Clock Path	Notes
Packet FIFO Domain				
PKT FIFO (AHB clock)	10 MHz	C01	P2_PF	
PKT FIFO (TLC clock)	20 MHz	C00	P0_A0	
PKT FIFO (FPGA clock)	10 MHz	C41	n/a	
PKT FIFO (FFE clk)	10 MHz	C08	X1_P3_PF	
M4F Subsystem Domain				
M4-Complex: M4F subsystem, M4-AHB switch (AHB clock)	80 MHz	C10	HCLK_P0_M4 FCLK_P0_M4	
M4-Complex: UART, WDT1, Timer1 (APB clock 0)	10 MHz	C11	P0_M4	
M4-Complex: to Audio SS and CFG_CTL (APB clock 1)	80 MHz	C10	FCLK_PS_AD0	
M4F CFG_CTL to FPGA (APB clock)	10 MHz	C09	P2_FB	
A0 (AHB clock M4)	80 MHz	C10	FCLK_P6_A0	
M4F SWD (DA pin)	20 MHz	CS	P0_M4	
SRAM Domain				
SRAM 128 Kbyte instance 0 (AHB clock)	80 MHz	C10	FCLK_P1_MS0	
SRAM 128 Kbyte instance 1 (AHB clock)	80 MHz	C10	FCLK_P2_MS1	
SRAM 128 Kbyte instance 2 (AHB clock)	80 MHz	C10	FCLK_P3_MS2	
SRAM 128 Kbyte instance 3 (AHB clock)	80 MHz	C10	FCLK_P4_MS3	
Audio Subsystem Domain				
Audio SS (AHB clock)	80 MHz	C10	FCLK_P5_AD0	The ratio between the Audio SS AHB and APB clock must be an integer ratio, such as 1-1, 1-2, 1-4.
Audio SS (APB clock)	10 MHz	C09	P0_AD5	
PDM left clock	5 MHz	C30	P0_AD1	
PDM right clock	5 MHz	C30	P1_AD2	
I^2S clock	5 MHz	C30	P2_AD4	
LPSD clock	1 MHz	C31	P3_AD3	
FPGA Domain				
FPGA	10 MHz	C16	P0_FB	
FPGA	10 MHz	C21	P0_FB	
AHB2WB for FPGA clock	10 MHz	C40	P0_FB	
FPGA to Packet FIFO clock	10 MHz	C41	P0_FB	

a. Max Frequency is with DVDD = 1.1V ±10%.

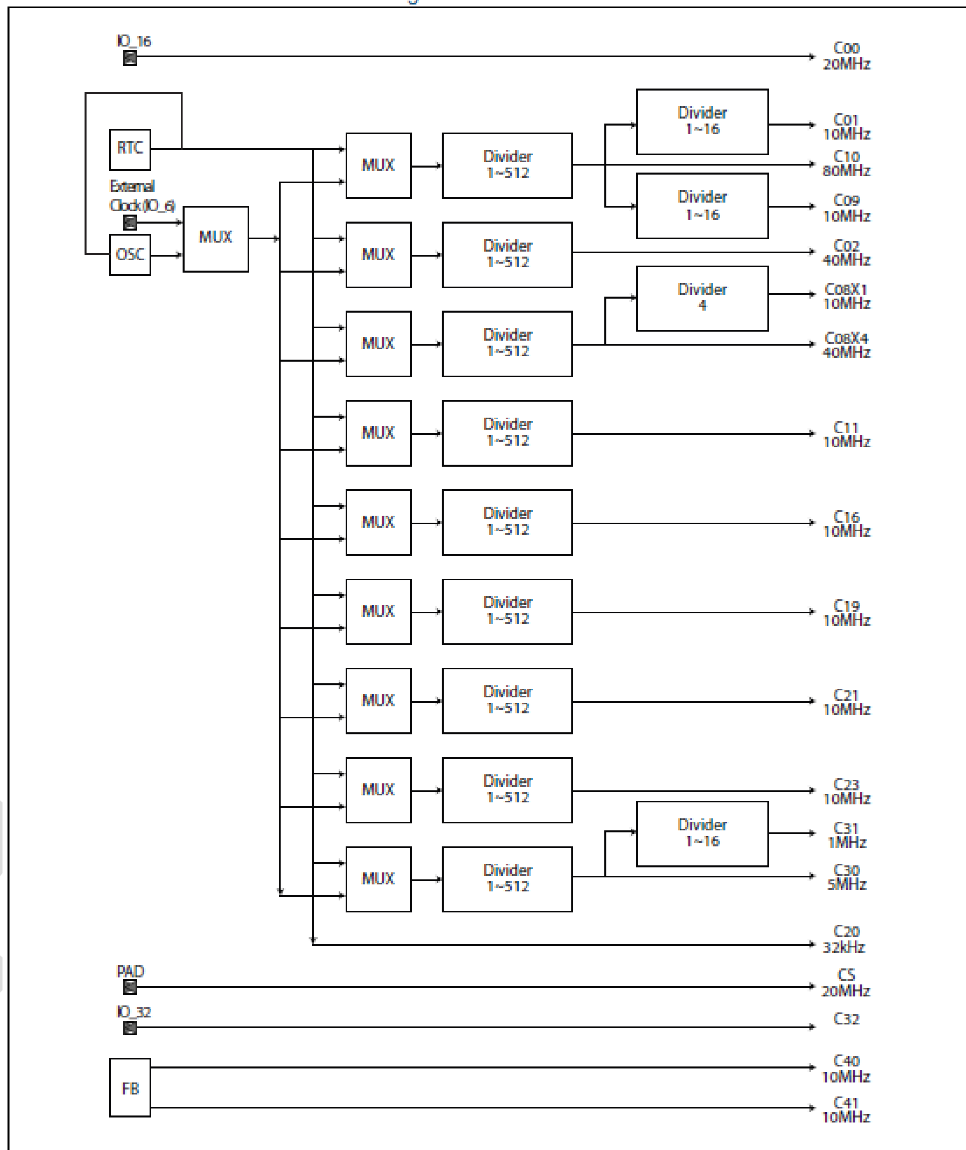
Clock domains are numbered in sequence (e.g., C00, C01, C02, and so on). Individual clock paths in a single clock domain are numbered (e.g., P0_A0, P1_A0, and so on). There can be similar clock path names, but they must be in different clock domains (e.g., C00 P0_A0, C01 P0_A0).

For most of the clock domains, there are three possible sources:

- Fast Clock Driving from IO_6 - FCLK
- Real-Time Clock (RTC - 32 kHz)
- Oscillator Clock (OSC - Max. is 80 MHz)

See Figure 46 for details.

Figure 46: Clock Tree



For Clock domains 01, 09 and 10, the clock phase is locked, but frequency may be different, same as Clock domain C30 and C31. For Clock C08X4 and C08X1, their clock phase is locked and frequency of C08X1 is always one-fourth of C08X4 clock frequency.

Most of Clock Paths can be gated by software independently, except Clock C00_P0, C01_P0, C10_HCLK_P0, C10_FCLK_P0, C20_P0 and C23_P0. Software can also gate off the clock domain individually. For Clock domains C40 and C41, the clock gating scheme is depends on the design inside the On-chip programmable logic.

~~2.7.2.2~~2.6.2.2 Resets

Each Clock path has its corresponding Reset Path. Most of them are asynchronous asserted and synchronous release except the Reset path for non-free running clock domain (e.g. clock from PAD, like C01, C40, C41, CS and C32). The reset for M4F core will not be de-asserted until clock is toggling for at least four 4 cycles.

There are two possible global reset sources

- Power-ON-Reset
- SYS_RSTn (System Reset)

After booting up, software can program a PMU register to block the SYS_RSTn (System Reset) and treat it as one of the interrupt sources. The software can also reset some of the modules such as Audio Support by programming register bits.

Preliminary

~~2.8~~2.7 Other EOS S3AI Platform Features

~~2.8.1~~2.7.1 Multi-Function Inputs/Outputs (IOs)

There are 46 I/Os for BGA package and 27 I/Os for WLCSP package that can be muxed for various functions. Each I/O output can have up to 4 different functional outputs. Each functional input can be selected from up to 8 different I/Os. The controls for I/Os (such as output enable, drive strength, etc.) can be controlled from three different sources; the A0 registers, the on-chip programmable logic and other sources (such as M4F, FFE, etc.). Refer to Table 28 for more IO options. Complete programming examples can be found in the QuickLogic *EOS S3AI Sensor Processing Platform Input Output Multiplexor User Guide*.

~~2.8.1~~2.7.1 General Purpose Inputs/Outputs (GPIOs)

Of the 46 multi-functional IOs, only 8 can be used as GPIOs by M4F to drive or sample from registers. Each of the 8 GPIOs can be assigned to 2 different IOs. Following are possible IO assignments for each of the 8 GPIOs. Refer to the QuickLogic *EOS S3AI Sensor Processing Platform Input Output Multiplexor User Guide* for programming details.

- IO_6 or IO_24 can be GPIO 0
- IO_9 or IO_26 can be GPIO 1
- IO_11 or IO_28 can be GPIO 2
- IO_14 or IO_30 can be GPIO 3
- IO_18 or IO_31 can be GPIO 4
- IO_21 or IO_36 can be GPIO 5
- IO_22 or IO_38 can be GPIO 6
- IO_23 or IO_45 can be GPIO 7

IMPORTANT: When doing system design, not all IOs can be used as M4F controllable GPIOs.

~~2.8.1~~2.7.1 Programmable Logic Inputs/Outputs (GPIOs)

Alternately, the 46 multi-functional IOs can be driven by on-chip programmable logic. This is listed as FBIO(x) in Table 28 in the Alternate Function column. Each IO can be driven by on-chip programmable logic as FBIO. For example, IO_0 is FBIO_0, IO_1 is FBIO_1, and so on. Refer to the QuickLogic *EOS S3 Sensor Processing Platform Input Output Multiplexor User Guide* for programming details.

IMPORTANT: While this option gives more flexibility for system design, it does consume more current as the on-chip programmable logic power island must be on and configured to utilize this feature.

Preliminary

~~2.8.2~~2.7.2 Interrupts

Interrupts that are generated by EOS S3AI device subsystem events can be routed to the Application Processor (AP) or the M4F.

~~2.8.2.1~~2.7.2.1 Interrupt Structure

Interrupts in the system can be routed to 2 different destinations.

- M4F - All interrupts to M4F is connected to M4's NVIC. There are 2 levels of interrupt masking and clearing. One is at the source of interrupts, and the other at the top level interrupt controller.
- AP - The interrupt mechanism for the AP is the same as M4's but it has a different mask. All interrupt sources are muxed to single combined interrupt before being sent to the AP.

~~2.8.2.2~~2.7.2.2 Interrupt Sources

Interrupts sourced from each subsystem functional blocks get combined into one interrupt for each subsystem.

- TOP Interrupts
  - Sensor/GPIO Interrupts - Eight pins can be used for either sensor interrupts or generic GPIO interrupts depending on system requirements. Each interrupt can be configured to be either *edge* detection (configurable to positive or negative edge) or *level* detection (configurable to high or low level setting).

- M4F Subsystem Interrupts
  - FPU - The Floating Point Unit can generate interrupts on floating point events.
  - Bus Timeout - There are bus timeout monitors that prevent AHB/APB slaves from locking up a system. If there is no response after 1024 clock cycles, an interrupt can be generated.
  - UART - The UART can generate transmit FIFO, receive FIFO, receive timeout, modem status, and error condition interrupts.
  - Timer - An interrupt can be triggered when the timer counts down to zero.
  - Watchdog Timer - Software can enable the watchdog timer, and upon counting down to zero, an interrupt can be triggered. If the interrupt is not cleared, a reset is triggered.
  - SRAM - An interrupt can be triggered when any segment of the 512KB (16 instances of 32KB) M4F memory is accessed when the memory is in a lower power state (e.g. deep sleep or shut down).

Preliminary

- FFE Interrupts
  - FFE Message - There are a total of 8 interrupt messages that can be used by FFE for various purposes.
  - FFE Subsystem - There are a total of 16 interrupts that are generated from the FFE subsystem, and they are combined into a single interrupt source.

- **A0 Interrupts**
  - AP Re-Boot - This interrupt is asserted when there is a need for rebooting. This occurs when all the M4F SRAMs are shut down for power savings, and upon wake up, rebooting is necessary.
  - Reset Interrupt - The SYS_RSTn pin can be used to generate an interrupt.
  - ADC Interrupt - Interrupt generated upon completion of analog to digital conversion.
  - PMU Timer - This 6 bit timer (with 32 KHz clock source) can be used to wake up FFE from low power mode before the FFE kickoff timer expires.
  - Software Interrupts - Two software interrupts can be triggered by software for handshaking between AP and the M4.
  - LDO Power Good - Independent interrupts from LDO30 and/or LDO50 is triggered when the voltage falls below threshold value.

- A1 Interrupts
  - Configuration DMA - In wearable mode, an interrupt is asserted after the DMA download from flash to M4F memories is completed.
  - SPI Master Interrupt - The configuration block can generate a combined interrupt. This will include interrupts from the SPI Master.

- Audio Interrupts
  - LPSD Audio Detect – Interrupt is triggered when audio is detected by LPSD HW
  - DMIC Audio Detect – Interrupt is triggered when audio is detected by DMIC
  - DMIC Audio Off – Interrupt is triggered when HW Loss of Audio detection by DMIC
  - LPSD Audio Off - Interrupt is triggered when Loss of Audio detection by LPSD
  - DMAC0 Block Done – DMAC0 finish transfer of a block size of data.
  - DMAC0 Buffer Done – DMAC0 finish transfer of a buffer size of data.
  - DMAC1 Block Done – DMAC1 finish transfer of a block size of data.
  - DMAC1 Buffer Done – DMAC1 finish transfer of a buffer size of data.
  - AP PDM Clock ON – AP PDM Clock is detected.
  - AP PDM Clock OFF – Loss of AP PDM Clock.

- SDMA Interrupts
  - SDMA Done – Interrupt per each channel of DMA (0-11) when DMA is completed.
  - SDMA Error – Interrupt for SDMA Error

- PKFB Interrupts
  - The four packet FIFOs can generated a combined interrupt for the following exception events: overflow, underflow, count threshold, access during sleep, and collision.

Preliminary

- On-chip programmable logic Interrupts
  - Four outputs from the fabric can be used as messages. Each interrupt message can be selected to be either *edge* or *level* detection. For edge detection, it can be configured to be positive or negative edge; similarly for level detection, it can be configured to be level high or low.

~~2.8.2.3~~2.7.2.3 M4F Wake-Up Events

The following interrupts sources can be used as wake-up events to the M4F. The M4F can configure the system to wake-up after a certain event. M4F can power itself down in the interim to conserve power. The PMU will wake up the M4F when the following interrupts are detected.

- Software Interrupts
- FFE Interrupts
- On-chip programmable logic Interrupts
- Sensor/GPIO Interrupts
- M4F SRAM Sleep interrupt
- UART
- TIMER
- WDOG Interrupt/Reset
- Bus Timeout
- FPU
- PKFB
- I2S
- Audio
- Configuration DMA
- Configuration SPI Master
- PMU Timer
- ADC Done
- RTC Alarm
- Reset Interrupt
- FFE Message
- FFE Combined
- AP Boot
- LDOs PG Interrupts
- SRAM Timeout
- LPSD Audio Detect
- DMIC Audio Detect
- SDMA DONE Channel 1–11

Note: SDMA Channel 0 (I^2S Slave) does not wake the M4F

Preliminary

2.8.32.7.3Bootstrap Modes

The EOS S3AI device I/O configuration options are selected by pulling special bootstrap pins high or low, which are latched upon de-assertion of the SYS_RSTn pin.

Figure 47: Bootstrap Timing

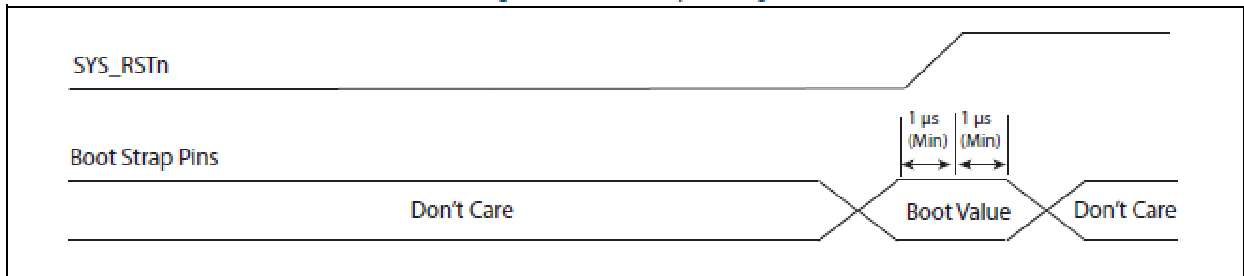


Table 20 shows the bootstrap timing during power-on sequence.

Table 20: Bootstrap Timing during Power-On Sequence

Symbol	Parameter	Min.	Typ.	Max.	Unit
T_{BSU}	Bootstrap pins setup time	1	-	-	µs
T_{BH}	Bootstrap pins hold time	1	-		µs

2.8.3.12.7.3.1 M4F Serial Wire Debug Port Configuration

IO_19 assert high is used for the Serial Wire Debug with pin IO_8 which configures I/Os used for the M4F Serial Wire Debug.

Table 21: M4F Serial Wire Debug Port Bootstrap Configuration

Serial Wire Debug Port Signal	IO_8 Pulled Down (default)	IO_8 Pulled Up
SW_CLK	IO_14	IO_45
SW_IO	IO_15	IO_44

2.8.3.22.7.3.2 Internal/External HSO Configuration

The configuration of internal or external High Speed Oscillator (HSO) is configured by bootstrap pins IO_8 and IO_9. When selecting the External HSO, the external clock is provided on IO_6.

Table 22: Internal/External HSO Configuration

IO_8	IO_9	Clock Source	HSO Configuration
Pulled Down (default)	Pulled Down (default)	Crystal or 32 kHz CMOS Clock at XTAL_IN	Internal HSO
Pulled Down (default)	Pulled Up	Crystal or 32 kHz CMOS Clock at XTAL_IN	Internal HSO
Pulled Up	Pulled Down (default)	32 kHz CMOS Clock at XTAL_IN	Internal HSO
Pulled Up	Pulled Up	IO_6	External HSO

~~2.8.3.1~~2.7.3.1 SWD Debugger Present Configuration

The state of bootstrap pin IO_19 configures whether the SWD debugger is present.

Note: This setting only applies in AP configuration. In the Wearable configuration, bootstrap pin IO_19 must be pulled down to allow operation of the SPI flash boot.

Table 23: SWD Debugger Present Configuration

IO_19	Debugger State
Pulled Down	Debugger is not available until after the M4F CPU core reset is released.
Pulled Up	Debugger access is allowed, as M4F CPU core reset is released immediately.

~~2.8.3.2~~2.7.3.2 AP/Wearable Mode Configuration

The state of bootstrap pin IO_20 configures whether the device is in AP mode (SPI slave) or Wearable mode (SPI master).

Table 24: AP/Wearable Mode Configuration

GPIO20	Mode	SPI Function	SPI I/Os Used
Pulled Down	Wearable	Master	GPIO34 - SCLK GPIO36 - MISO GPIO38 - MOSI GPIO39 – SS1
Pulled Up	AP	Slave	GPIO16 - SCLK GPIO17 - MISO GPIO19 - MOSI GPIO20 - CS

If in Standalone mode, the M4F CPU core reset is de-asserted automatically once boot code is downloaded by Configuration DMA.

In Companion mode, the Application Processor controls the release the M4F CPU core reset through register settings.

Preliminary

2.92.8 Other Peripherals

2.9.12.8.1 Packet FIFO

The packet FIFO bank provides data buffering for data transfers between FFE and/or on-chip programmable logic and/or M4F to AP and/or M4F. It is composed of four packet FIFOs of differing sizes. A typical use case may have the FFE push sensor data as packets into the Packet FIFO. When a specific threshold is reached, the programmable interrupt signals to the M4F or AP to pop off the data for additional processing. The M4F can also write data into the Packet FIFO and pop off the data. The on-chip programmable logic can push data packets into the Packet FIFOs, depending on the on-chip programmable logic configuration.

Note: The FFE and On-chip programmable logic cannot pop data from Packet FIFOs to perform additional processing.

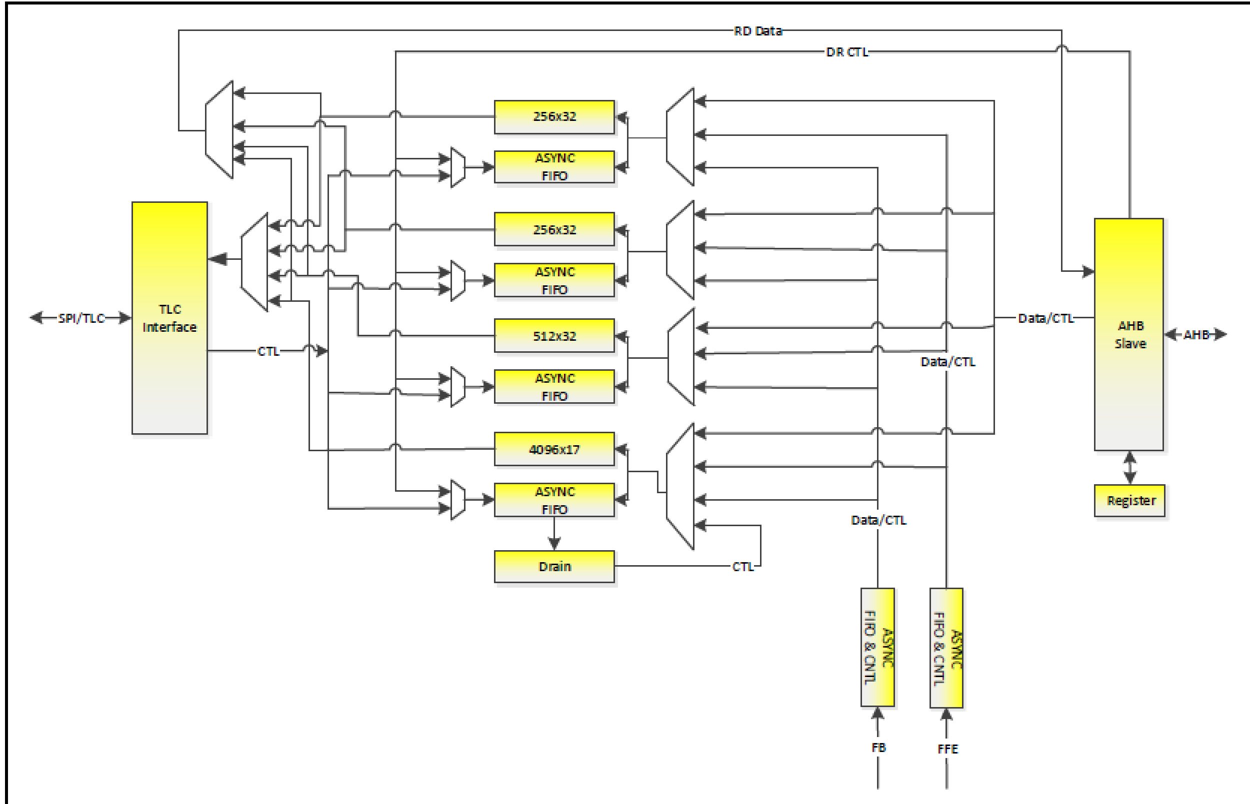
Packet FIFOs are designed to support two operating modes:

- Packet mode – this FIFO mode differs from a normal FIFO in its generation of the pop side flags (empty and pop word count). It only considers data pushed up to end-of-packet (EOP) boundaries.
- Normal FIFO mode – this mode behaves like a normal FIFO with pop side flags updated for every pushed data.

Preliminary

Figure 48 shows a block diagram for Packet FIFO Bank (PKFB).

Figure 48: PKFB Block Diagram



The FIFO instances are listed in the table below.

Table 25: Packet FIFO Instances

Instance	Depth	Width	Description
FIFO_8K	4096	17	• Supports normal or packet FIFO modes • Supports ring buffer mode support (16-bit data, 1-bit SOP) • Drainer logic with programmable threshold to implement ring buffer function • PUSH source: FFEs or M4F • POP destination: AP or M4F
FIFO_0	256	32	• Supports normal FIFO modes • PUSH source: FFEs or M4F • POP destination: AP or M4F
FIFO_1	128	32	
FIFO_2	128	32	

~~2.9.1.1~~2.8.1.1 FIFO_8K

The FIFO_8K is a 4096x17 packet FIFO that has the following configurable options:

- Packet FIFO mode - when enabled, the FIFO behaves as a packet FIFO, otherwise, it behaves like a generic FIFO.
- Ring buffer mode - When enabled, a small drainer block on the pop side of the FIFO will be enabled. This drainer logic is triggered once the pop word count reaches a programmable threshold, which causes it to pop a packet off the top of the FIFO. Once the final pop agent (AP or M4F) is triggered by an external event to start popping the FIFO, it will disable the drainer logic to start reading the FIFO. Care must be taken when disabling the drainer logic to avoid any type of race condition that can cause coherency issues. One possible way of doing this is for the AP/M4F to disable the drainer logic before it polls the drainer logic BUSY status. The drainer is designed only to check the enable bit at the start of the pop transactions. Controls for the muxes should consider this to prevent going off-sync (such as changing mux control before the logic drainer is done).

NOTE: The Ring buffer mode can only be used with packet FIFO mode. Software must ensure that it does not enable ring buffer mode for non-packet FIFO operation.

- Threshold - this register determines the FIFO threshold that will trigger either the drainer logic, when used in ring mode, or an interrupt (when used as a normal FIFO). Both these are for the purpose of avoiding a FIFO overrun condition.

The packet FIFO provides FIFO word count on the pop side. When used in packet FIFO mode, this indicates the exact number of words in the FIFO that represent full packets. When used as a normal FIFO, the FIFO word count specifies the exact number of words in the FIFO regardless of packet boundaries. The AP or M4F is expected to read the FIFO word count and pop no more data than allowed by the count. In this way, the empty flag signal is never used to throttle the pop of data.

In ring buffer mode, the start of packet signal (SOP) is pushed into the FIFO as data bit[16] to alert the drainer logic to identify the SOP on the pop side so it is able to pop packets when the threshold is reached.

~~2.9.1.2~~2.8.1.2 FIFO_0, FIFO_1, FIFO_2

All three FIFOs are designed to be generic FIFOs that have the following firmware configurable option:

- Threshold - this register determines the FIFO threshold that will trigger an interrupt when reached. This is for the purpose of avoiding a FIFO overrun condition.
- Sizes – FIFO_0 is 256x32 bits, and both FIFO_1 and FIFO_2 are 128x32 bits.

Preliminary

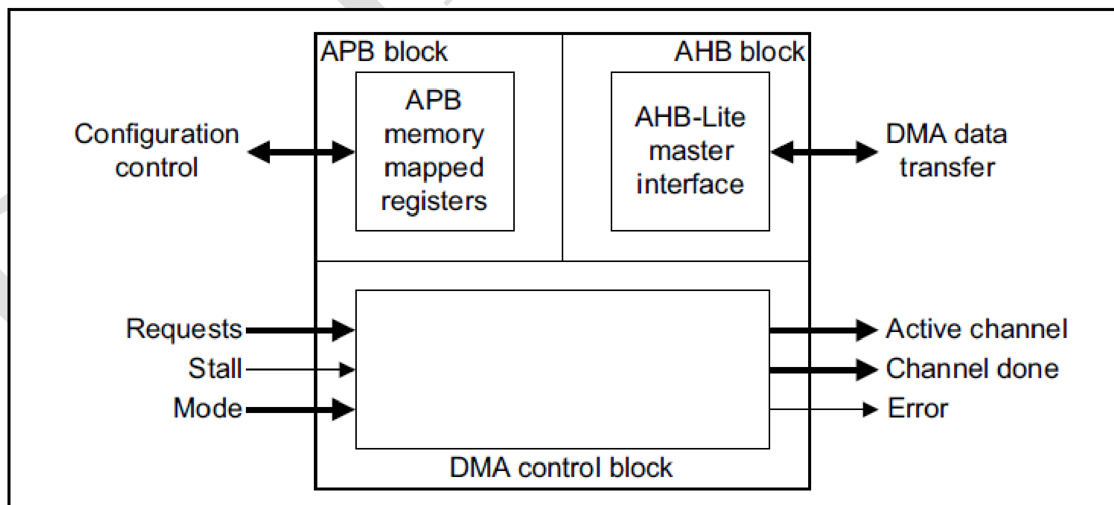
As with FIFO_8K, these FIFOs provide FIFO word count on the pop side. FIFO word counts specify the exact number of words in the FIFO regardless of packet boundaries. The AP or M4F is expected to read the FIFO word count and pop no more than the count. In this way, the empty flag signal is never used to throttling the pop of data.

~~2.9.1~~2.8.1 System DMA

The principal features of SDMA including the following (and are illustrated in Table 26):

- Uses AHB-Lite for the DMA transfers
- Uses APB for programming the registers
- Single AHB-Lite master for transferring data using a 32-bit address bus and 32-bit data bus
- Supports up to 16 DMA channels
- Dedicated handshake signals on each DMA channel
- Programmable priority level on each DMA channel
- Each priority level arbitrates using a fixed priority that is determined by the DMA channel number
- Supports multiple transfer types:
    - Memory-to-Memory
    - Memory-to-Peripheral
    - Peripheral-to-Memory
- Supports multiple DMA cycle types
- Supports multiple DMA transfer data widths
- Each DMA channel can access a primary and an alternate channel control data structure
- All channel control data is stored in system memory using the little-endian format
- Performs all DMA transfers using the SINGLE AHB-Lite burst type

Figure 49: System DMA Block Diagram

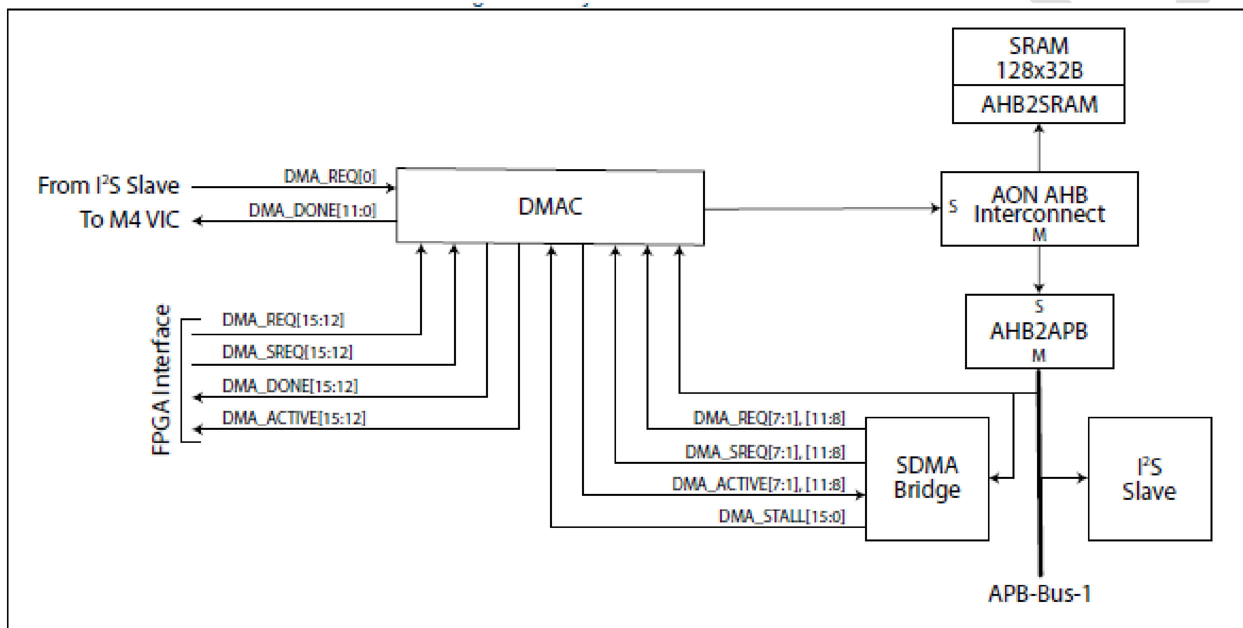


Preliminary

~~2.9.1.1~~2.8.1.1 Function Description

The SDMA aids in offloading data move tasks from the M4F CPU, and allows M4F to sleep as long as possible to save power. The SDMA can support a hardware/software request, and DMA requests can be from a peripheral or initiated by software. The SDMA uses an AHB-Lite Master port for reading DMA descriptors from 128x32 bits SRAM and transferring data from source to destination and the SDMA AHB Master port is connected to the Always-On AHB bus matrix so that it can transfer data even when M4F power domain is in deep sleep or shut down. The APB bus is the register interface of the SDMA and SDMA Bridge.

Figure 50: System DMA Interface



The blocks that can initiate transfer using SDMA are:

1. I2S slave port
2. M4F processor
3. FFE
4. On-chip programmable logic

System DMA supports up to 16 DMA channels. Channel assignment is shown in Table 26. Each channel has a primary and an alternate descriptor associated with it, and each description consists of four words. These descriptors are hosted in a 128x32b SRAM connected to Always-On AHB bus matrix. SDMA bridge can generate single/burst DMA request to SDMA, which is software controlled.

Table 26: SDMA channel assignment

Channel Number	Primary Channel Programming	Alternate Channel Programming	Channel Trigger	Channel Acknowledge	Comments
0	M4	FFE	I2S	M4	Masking under SW control possible
1-7	M4	FFE	SDMA Bridge	SDMA Bridge & M4	These channels are under direct control of M4
8-11	FFE	M4	SDMA Bridge	SDMA Bridge & M4	These channels can be controlled by FFE or M4. DMA_DONE signals associated with these channels are connected to M4F VIC
12-15	M4	FFE	FPGA	FPGA	

The System DMA can perform the following tasks:

- Transfer Audio data from M4F SRAM to I2S slave.
- Transfer data from M4F SRAM to FFE CM (swapping).
- Transfer data from M4F SRAM to FFE DM.
- Transfer data from FFE DM to M4F SRAM.
- Transfer data from M4F SRAM to On-chip programmable logic.
- Transfer data from On-chip programmable logic to M4F SRAM.

~~2.9.1.2~~2.8.1.2 SDMA Configurations

The configuration for SDMA transfers includes the following three elements:

- DMA descriptors - Each DMA channel has two associated channel descriptor structures which are normally located in SDMA SRAM. These include the source and destination address for the channel as well as information on number of elements to transfer, data size, transfer type, etc.

  Note: When a channel is triggered, the DMA reads the associated descriptor from SRAM, which includes the instructions on what actions the DMA should take. When the channel has finished the defined actions, the updated descriptors are written back to the SRAM.

- DMA registers - Common configurations for the DMA, as well as DMA-generated interrupts and trigger sources for the various channels are configured in the DMA registers. The location of DMA descriptors in SRAM is also configured here.
- Registers in trigger peripheral – The DMA request signals from the peripherals get generated by various events in the peripherals; hence, it is important to configure the peripherals correctly to generate the desired DMA requests.

Preliminary

2.9.22.8.2 Analog to digital converter (ADC)

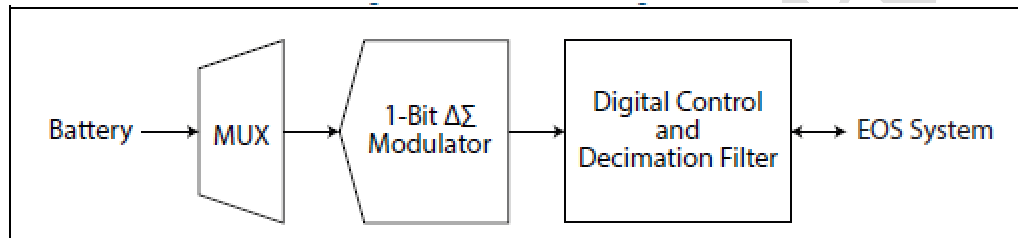
2.9.2.12.8.2.1 Overview

The ADC is an accurate high resolution ADC for voltage monitoring. To achieve excellent repeatability and high PSRR, the ADC uses a 12-bit advanced fully differential Delta-Sigma ADC.

The ADC is specified from T_J = -40°C to +125°C and it is designed to achieve 2.0% overall accuracy.

Figure 51 shows a general block diagram of the ADC.

Figure 51: ADC Block Diagram



2.9.2.22.8.2.2 Functional Description

In brief, the ADC voltage measurement core includes an input multiplexer, a delta-sigma modulator, and a digital control core. The digital control core controls the analog blocks power up/down sequences, generates the delta-sigma control signals and implements the output decimation.

2.9.2.32.8.2.3 Electrical Characteristics

Table 27 shows key electrical characteristics for the ADC module.

A_{GND} = 0V DC, Internal V_{REF}= 1.200V DC, F_{CLK} = 500kHz, T_J = -25°C to +125°C, unless otherwise specified

Table 27: ADC Electrical Characteristics

Symbol	Description	Min.	Typ.	Max.	Units
V_{AVDD}	Analog Supply Voltage	1.62	1.80	3.63	V
V_{DAC}	Input Voltage Conversion Range	0	-	1.4	V
F_{CLK}	Delta-Sigma Clock Frequency	200	1000	2000	kHz
Temp	Temperature Range	-20	-	85	°C
I_{AVDD}	Analog Current	0.1	0.2	0.3	mA
IQ_{PD}	Total Power Down Current Consumption	-	-	1	µA
ADC	ADC Resolution Voltage Per Step	-	0.34	-	µV
T_{CONV}	ADC Conversion Time[a]	2.5	12.5	25	ms

a. ADC conversion time is ~5,000 F_{CLK} cycles.

Preliminary

~~2.9.2.4~~2.8.2.4 PCB Layout Recommendations

When using ADC, the PCB layout recommendation includes:

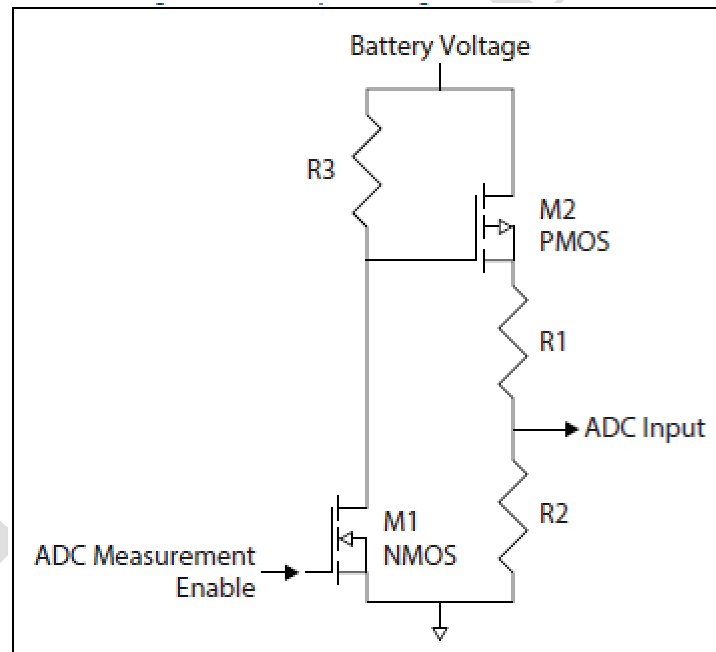
- Minimize the distances between the ADC input and the location where the voltage being measured. Avoid routing where it is close to noise sources such as clock generators, DC/DC converters and data/address buses.
- Minimize inductance and reduce series resistance by using wide tracks. Use grounded guard traces when possible. A trace with a 10 mil minimum width and spacing is recommended.

~~2.9.2.5~~2.8.2.5 Example Application

The ADC input voltage conversion range is 0 V to 1.4 V. Most rechargeable battery outputs are higher than the ADC input voltage. To support higher input voltage range, a maximum of up to 4 VDC, use an external analog level shifter to scale the battery voltage to ensure it is compatible with the ADC voltage range.

Figure 52 shows an example.

Figure 52: Example Voltage Divider Circuit



The voltage divider resistors, R1 and R2 should be chosen according to the amount of voltage scaling required. It is left to software to scale the ADC values to best determine the proper corresponding battery voltage level. In addition, the ADC provides an enable output for the specific purpose of controlling an external voltage divider.

Preliminary

In this example, this enables control components M1, M2, and R3, which allow the ADC to disable the voltage divider between ADC measurements. Not doing so results in a constant current draw on the battery. Any constant current draw causes reduced battery life. Besides extending battery life, an additional benefit is that these components disable the voltage divider when the EOS S3 device is in a low power state.

~~2.9.3~~2.8.3 Universal Asynchronous Receiver Transmitter (UART)

The UART provides a serial data connection that can be used for communications and trace. The main features of UART include:

- Programmable use as UART or IrDA SIR input/output
- Separate 32×8 transmit and 32×12 receive FIFO memory buffers to reduce CPU interrupts
- Programmable FIFO disabling for 1-byte depth
- Programmable baud rate generator
- Standard asynchronous communication bits (start, stop and parity)
- Independent masking of transmit FIFO, receive FIFO, receive timeout, modem status, and error condition interrupts
- Support for Direct Memory Access (DMA)
- False start bit detection
- Line break generation and detection
- Support of the modem control functions CTS, RTS
- Programmable hardware flow control
- Fully-programmable serial interface characteristics:
    - data can be 5, 6, 7, or 8 bits
    - even, odd, stick, or no-parity bit generation and detection
    - 1 or 2 stop bit generation
    - baud rate generation, dc up to UARTCLK/16
- IrDA SIR ENDEC block which supports:
    - programmable use of IrDA SIR or UART input/output
    - support of IrDA SIR ENDEC functions for data rates up to 115200 bps half-duplex
    - support of normal 3/16 and low-power (1.41-2.23µs) bit durations
    - Programmable division of the reference clock to generate the appropriate bit duration for low-power IrDA mode.

Preliminary

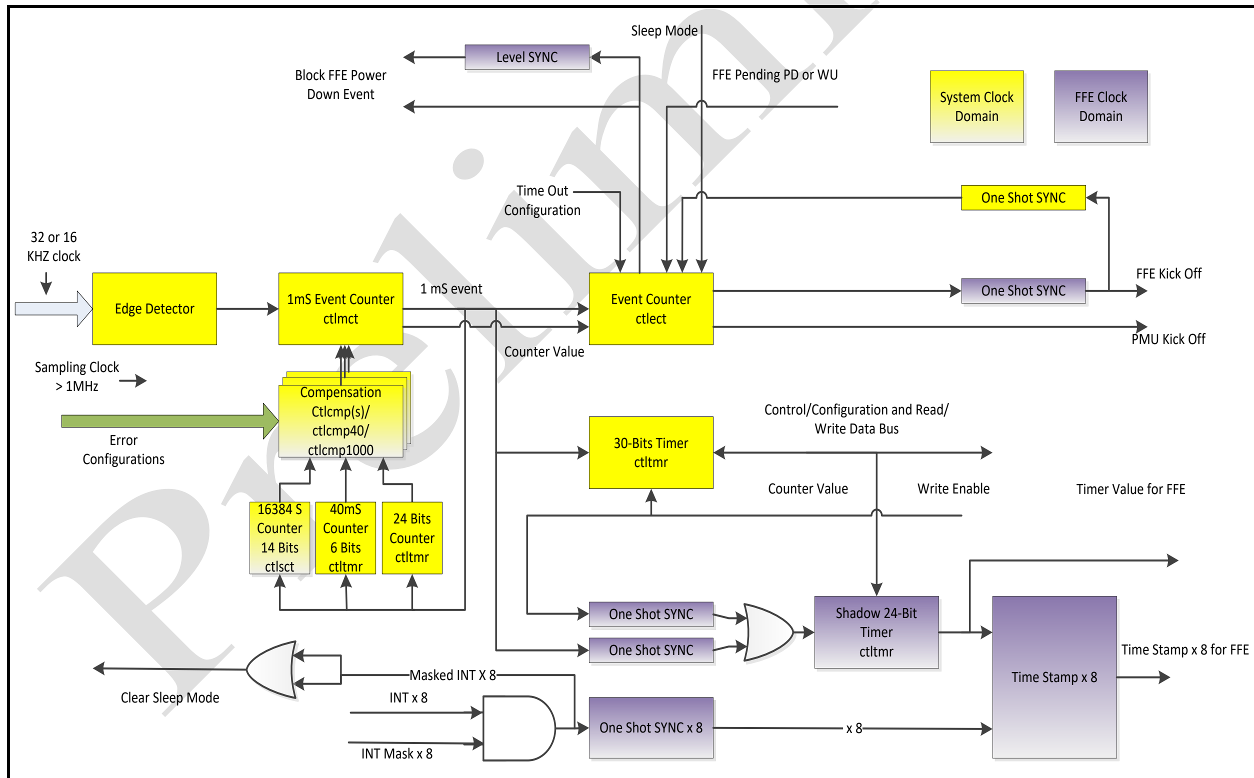
~~2.9.4~~2.8.4 Timer and Counters

The EOS S3AI platform supports several counters that count the clock event to generate the 1ms event as well as time out events for waking up FFE and FFE power domain. It also provides the 24-bit timers and eight time stamps for FFE and 30-bit timer for software use.

Clock events are generated on both edges of reference clock. The 1mS time out event can be adjusted by configuring the trim bits.

- Clock Event Generator: Generate the Clock Event base on the edge of reference clock. Reference Clock could be either 16 KHz or 32 KHz with certain PPM error.
- 1ms event Counter: The resolution is 1 Clock Event.
- 30/24 Bits Timer: The resolution is 1mS (for details, see Figure TBD; the 30-boit timer associated with the 1mS event counter, the shadow 24-bit timer associated with the FFE).
- Time Stamp: Eight total, timers LSB (Lower 16 bits of timers) is latched once the corresponding Interrupt triggers.
- 5mS time-out event with less than 1% error in two hours period.

Figure 53: Timer Block Diagram



Preliminary

~~2.9.4.1~~2.8.4.1 1ms Event Counter

For 32 KHz Reference Clock

- 1ms event is alternating between 65 and 66 counts of clock event in the following sequence
    - 65 → 66 → 65 → 66 → 65 → 66 → 65…….

For 16 KHz Reference Clock

- 1ms event is alternating between 32 and 33 counts of clock event in the following sequence
    - 32 → 33 → 33 → 33 → 32 → 33 → 33 → 33 → 32 → 33→ 33….

~~2.9.4.2~~2.8.4.2 Error Correction for 1mS Event Counter

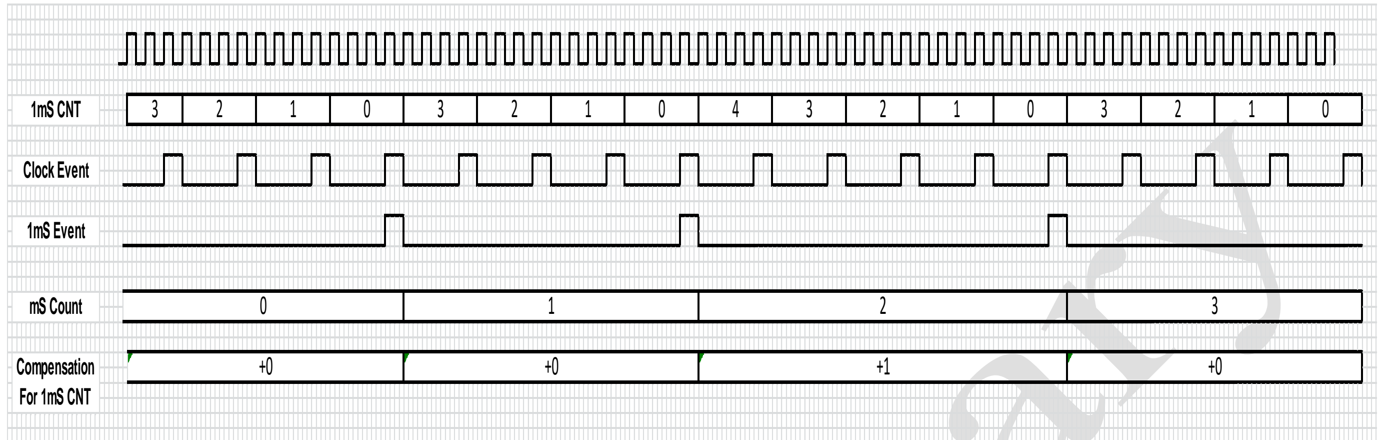
The 1ms event counter implementation will accumulate around -550us error for 1 Second period without error compensation. As a result, an Error Correction circuit is implemented to compensate the error of design and reference clock.

The following list represents multiple layer error correction compensation schemes in use:

- Compensation every 40ms – Increase 1 or not
- Compensation every 1 Second – Increase or Decrease 1
- Compensation every 2 Second – Increase or Decrease 1
- Compensation every 4 Seconds – Increase or Decrease 1
- Compensation every 8 Seconds – Increase or Decrease 1
- Compensation every 16 Seconds – Increase or Decrease 1
- Compensation every 32 Seconds – Increase or Decrease 1
- Compensation every 64 Seconds – Increase or Decrease 1
- Compensation every 128 Seconds – Increase or Decrease 1
- Compensation every 256 Seconds – Increase or Decrease 1
- Compensation every 512 Seconds – Increase or Decrease 1
- Compensation every 1024 Seconds – Increase or Decrease 1
- Compensation every 2048 Seconds – Increase or Decrease 1
- Compensation every 4096 Seconds – Increase or Decrease 1
- Compensation every 8192 Seconds – Increase or Decrease 1
- Compensation every 16384 Seconds – Increase or Decrease 1

Preliminary

Figure 54: 1ms Count and 1ms Counter Relationship



~~2.9.4.3~~2.8.4.3 Timeout Event Counter

The timeout event counter counts the 1ms events and the time out period is (from 1ms to 255ms) based on configured value.

~~2.9.4.4~~2.8.4.4 30 Bits Counter

The 30-bit counter count the 1ms event (in 1ms resolution) and allows the software read/write the timer value through Registers space.

~~2.9.4.5~~2.8.4.5 Time Stamp Counters

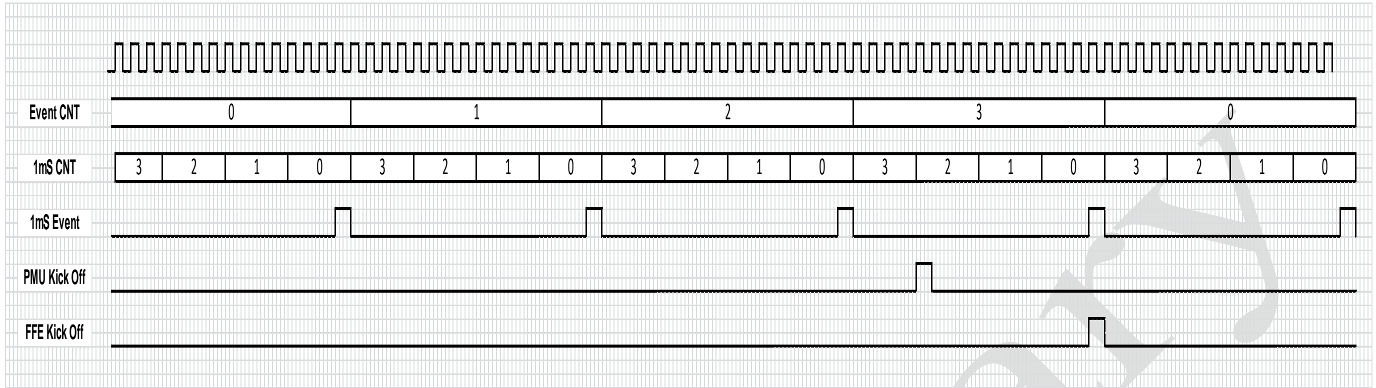
There are eight time stamps. Each time stamp has 16-bits and the corresponding interrupt source that could be individual mask out. If the interrupt triggers, the lower 16 bits of the 24 bits timer loads with the corresponding time stamp.

~~2.9.4.6~~2.8.4.6 PMU and FFE Wakeup

This timer allows the PMU to power up the sensor processing subsystem FFE. Once the FFE power domain is ON, a kick-off event is sent to the FFE. There are register controls that control the timing relationship between the time that FFE is powered up and the kick-off event is sent to the FFE, which provides for some timing flexibility for the software.

Preliminary

Figure 55: PMU and FFE Timing Waveform



Preliminary

3 Device Characteristics

3.1 Pinout and Pin Description

Table 28 lists the input and output (I/O) location and functions for each of the IO pins in EOS S3. It is important to note two things:

- There is at least one default function assigned to each IO (and its default = 0)
- The EOS S3AI software does not configure the IO if it is being used as a default

Note: Table below lists the default functions in **bold** for all EOS S3 IO pins in the Alternate Functions column. This convention visually indicates the default function for each IO. There are also cases indicated where end-users may choose to bootstrap IO pins to suit specific product-based requirements and these are noted (for example, see IO_14).

Preliminary

Table 28: I/O Locations and Functions

Signal Name	Signal Type	I/O Bank	WCLSP Ball	BGA Ball	Description	Alternate functions
ADC0	ANALOG	ANALOG	B2	A7	ADC 0 Input	
ADC1	ANALOG	ANALOG	--	C7	ADC 1 Input	
AGND	ANA		C3	A8	Analog Ground	
AVDD	ANA		C2	D8	Analog Power	
FSOURCE	VPP		E5	G3	Ground (Set to 1.8V for eFuse programming)	
GND	GND		C4,C5	D3,D4,D5	Ground	
LDO_VIN	POWER		B3	A6	Internal LDO Power Input	
XTAL_I	ANALOG	ANALOG	A1	B8	32kHz Crystal	32kHz Clock Input
XTAL_O	ANALOG	ANALOG	B1	C8	32kHz Crystal	
IO_0	IO	VCCIOA	A7	B1		FBIO_0, **SCL_0**
IO_1	IO	VCCIOA	B7	C1		FBIO_1, **SDA_0**
IO_2	IO	VCCIOA	--	A1		**FBIO_2**, SPI_SENSOR_SS2n, DEBUG_MON_0, BATT_MON, SENSOR_INT_1
IO_3	IO	VCCIOA	C7	A2		FBIO_3, **SENSOR_INT_0**
IO_4	IO	VCCIOA	--	B2		**FBIO_4**, SPI_SENSOR_SS3n, DEBUG_MON_1, SDA_1_DPU, SENSOR_INT_2
IO_5	IO	VCCIOA	--	C3		**FBIO_5**, SPI_SENSOR_SS4n, DEBUG_MON_2, **SDA_0_DPU**, SENSOR_INT_3
IO_6	IO	VCCIOA	A6	B3		**FBIO_6**, SPI_SENSOR_MOSI, DEBUG_MON_3, **FCLK**, GPIO(0), IrDA_SIRIN, SENSOR_INT_1 Note: FBIO_6 (when bootstrap IO_9 = 0 and choose FCLK when IO_9 = 1).
IO_7	IO	VCCIOA	--	A3		**FBIO_7**, SPI_SENSOR_SS5n, DEBUG_MON_4, SWV, SENSOR_INT_4
IO_8	IO	VCCIOA	B6	C4		**FBIO_8**, PDM_CLK_O, I2S_CLK_O, IrDA_SIROUT, SENSOR_INT_2
IO_9	IO	VCCIOA	A5	B4		**FBIO_9**, SPI_SENSOR_SS1n, I2S_WD_CLK_O, GPIO(1),PDM_STAT_I, SENSOR_INT_3
IO_10	IO	VCCIOA	B5	A4		**FBIO_10**, SPI_SENSOR_CLK, SWV, SENSOR_INT_4, I2S_DIN, PDM_DIN
IO_11	IO	VCCIOA	--	C5		**FBIO_11**, SPI_SENSOR_SS6n, DEBUG_MON_5, GPIO(2), SENSOR_INT_5
IO_12	IO	VCCIOA	--	B5		**FBIO_12**, SPI_SENSOR_SS7n, DEBUG_MON_6, IrDA_SIROUT, SENSOR_INT_6
IO_13	IO	VCCIOA	--	D6		**FBIO_13**, SPI_SENSOR_SS8n, DEBUG_MON_7, SWV, SENSOR_INT_7
IO_14	IO	VCCIOA	A4	A5		**FBIO_14**, **SW_DP_CLK**, IrDA_SIROUT, SCL_1, GPIO(3), UART_RXD, SENSOR_INT_5 Note: FBIO_14 when bootstrap IO_8 = 1 and choose SW_DP_CLK when IO_8 = 0.
IO_15	IO	VCCIOA	B4	C6		**FBIO_15**, **SW_DP_IO**, IrDA_SIRIN, SDA_1, UART_TXD, SENSOR_INT_6 Note: FBIO_15 when bootstrap IO_8 = 1 and choose SW_DP_IO when IO_8 = 0.
IO_16	IO	VCCIOB	E1	E7		**FBIO_16**, **SPI_SLAVE_CLK**, UART_RXD Note: FBIO_16 when bootstrap IO_20 = 0 and choose SPI_SLAVE_CLK when IO_20 = 1.

Preliminary

Signal Name	Signal Type	I/O Bank	WCLSP Ball	BGA Ball	Description	Alternate functions
IO_17	IO	VCCIOB	D1	D7		**FBIO_17**, **SPI_SLAVE_MISO**, UART_CTS Note: FBIO_17 when bootstrap IO_20 = 0 and choose SPI_SLAVE_MISO when IO_20 = 1.
IO_18	IO	VCCIOB	--	E8		**FBIO_18**, SWV, DEBUG_MON_0, GPIO(4), SENSOR_INT_1
IO_19	IO	VCCIOB	C1	H8		**FBIO_19**, **SPI_SLAVE_MOSI**, UART_RTS Note: IO_19 is served as bootstrap for debugger mode as an M4F reset release mechanism. Note: FBIO_19 when bootstrap IO_20 = 0 and choose SPI_SLAVE_MOSI when IO_20 = 1.
IO_20	IO	VCCIOB	F2	G8		**FBIO_20**, **SPI_SLAVE_SSn**, UART_TXD Note: FBIO_20 when bootstrap IO_20 = 0 and choose SPI_SLAVE_SSn when IO_20 = 1.
IO_21	IO	VCCIOB	--	H7		**FBIO_21**, DEBUG_MON_1, IrDA_SIRIN, GPIO(5), UART_RTS, SENSOR_INT_2
IO_22	IO	VCCIOB	--	G7		**FBIO_22**, DEBUG_MON_2, IrDA_SIROUT, GPIO(6), UART_CTS, SENSOR_INT_3
IO_23	IO	VCCIOB	E2	H6		**FBIO_23**, SPI_MASTER_SS2n, SWV, GPIO(7), AP_I2S_WD_CLK_IN, SENSOR_INT_7
IO_24	IO	VCCIOB	D2	G6		**FBIO_24**, AP_I2S_DOUT, IrDA_SIRIN, GPIO(0), UART_TXD, SENSOR_INT_1
IO_25	IO	VCCIOB	D3	F7		**FBIO_25**, SPI_MASTER_SS3n, SWV, IrDA_SIROUT, UART_RXD, SENSOR_INT_2
IO_26	IO	VCCIOB	--	F6		**FBIO_26**, SPI_SENSOR_SS3n, DEBUG_MON_3, GPIO(3), SENSOR_INT_4
IO_27	IO	VCCIOB	--	H5		**FBIO_27**, SPI_MASTER_SS2n, SPI_SENSOR_SS4n, DEBUG_MON_4, SENSOR_INT_5
IO_28	IO	VCCIOB	F3	G5		**FBIO_28**, SPI_SENSOR_MOSI, DEBUG_MON_5, GPIO(2), I2S_DIN, PDM_DIN, IrDA_SIRIN, SENSOR_INT_3
IO_29	IO	VCCIOB	E3	F5		**FBIO_29**, SPI_SENSOR_MISO, I2S_CLK_O, PDM_CLK_O, IrDA_SIROUT, SENSOR_INT_4
IO_30	IO	VCCIOB	F4	F4		**FBIO_30**, SPI_SENSOR_SS1n, GPIO(3), I2S_WD_CLK_O, PDM_STAT_I, SENSOR_INT_5
IO_31	IO	VCCIOB	E4	G4		**FBIO_31**, SPI_SENSOR_CLK, GPIO(4), AP_I2S_CLK_IN, SENSOR_INT_6
IO_32	IO	VCCIOB	--	H4		**FBIO_32**, SPI_SENSOR_SS5n, DEBUG_MON_6, SDA_1, SENSOR_INT_6
IO_33	IO	VCCIOB	--	E3		**FBIO_33**, SPI_SENSOR_SS6n, DEBUG_MON_7, SCL_1, SENSOR_INT_7
IO_34	IO	VCCIOB	D5	F3		**FBIO_34**, **SPI_MASTER_CLK**, DEBUG_MON_0, AP_PDM_STAT_O, SENSOR_INT_7 Note: FBIO_34 when bootstrap IO_20 = 1 and choose SPI_MASTER_CLK when IO_20 = 0.
IO_35	IO	VCCIOB	--	F2		**FBIO_35**, SPI_MASTER_SS3n, SPI_SENSOR_SS7n, DEBUG_MON_1, SENSOR_INT_1
IO_36	IO	VCCIOB	F5	H3		**FBIO_36**, **SPI_MASTER_MISO**, SWV, SPI_SENSOR_SS2n, GPIO(5), SENSOR_INT_1 Note: FBIO_36 when bootstrap IO_20 = 1 and choose SPI_MASTER_MISO when IO_20 = 0.
IO_37	IO	VCCIOB	--	G2		**FBIO_37**, SPI_SENSOR_SS8n, DEBUG_MON_2, SDA_2_DPU, SENSOR_INT_2

Preliminary

Signal Name	Signal Type	I/O Bank	WCLSP Ball	BGA Ball	Description	Alternate functions
IO_38	IO	VCCIOB	E6	E2		**FBIO_38**, **SPI_MASTER_MOSI**, DEBUG_MON_3, GPIO(6), AP_PDM_CLK_IN, SENSOR_INT_2 Note: FBIO_38 when bootstrap IO_20 = 1 and choose SPI_MASTER_MOSI when IO_20 = 0.
IO_39	IO	VCCIOB	F6	H2		**FBIO_39**, **SPI_MASTER_SS1n**, DEBUG_MON_4, AP_PDM_IO, SENSOR_INT_3 Note: FBIO_39 when bootstrap IO_20 = 1 and choose SPI_MASTER_SS1n when IO_20 = 0.
IO_40	IO	VCCIOB	--	D2		**FBIO_40**, SCL_2, DEBUG_MON_5, IrDA_SIRIN, SENSOR_INT_3
IO_41	IO	VCCIOB	--	F1		**FBIO_41**, SDA_2, DEBUG_MON_6, IrDA_SIROUT, SENSOR_INT_6
IO_42	IO	VCCIOB	--	H1		**FBIO_42**, SDA_1_DPU, DEBUG_MON_7, SWV, SENSOR_INT_7
IO_43	IO	VCCIOB	D7	D1		FBIO_43, **AP_INTERRUPT**
IO_44	IO	VCCIOB	E7	E1		**FBIO_44**, **SW_DP_IO**, SDA_1, UART_TXD, IrDA_SIRIN, SENSOR_INT_4 Note: FBIO_44 when bootstrap IO_8 = 0 and choose SW_DP_IO when IO_8 = 1.
IO_45	IO	VCCIOB	F7	G1		**FBIO_45**, **SW_DP_CLK**, SCL_1, UART_RXD, IrDA_SIROUT, GPIO(7), SENSOR_INT_5 Note: FBIO_45 when bootstrap IO_8 = 0 and choose SW_DP_CLK when IO_8 = 1.
STM	INPUT	VCCIOB	D4	E5	Ground (Set to VCCIOB for eFuse programming)	
SYS_RSTn	INPUT	VCCIOB	F1	F8	System Reset Input	
VCCIOA	POWER		C6	C2	Bank A VCC In	
VCCIOB	POWER		D6	E4,E6	Bank B VCC In	
VDD1	POWER		A2	B7	LDO 1 Output	
VDD2	POWER		A3	B6	LDO 2 Output	

NOTE:

a. Each of the 46 multi-function IOs can be used as on-chip programmable logic (FB) IOs. Each IO is assigned a corresponding FBIO function. Refer to subsection **Fabric Inputs/Outputs (FBIOs)** for more details.

b. M4F can only control total of 8 IO pads as GPIOs out of the 46 multi-function IOs. For example, M4F can control GPIO bit 0 on either IO_6 or IO_24. M4F can control GPIO bit 1 on either IO_9 or IO_26, etc. Only 1 IO can be selected for each GPIO. Both IOs cannot be selected at the same time. Look at the Alternate Function column to see which IOs can be used as GPIOs. A complete explanation and listing can be found in the subsection **General Purpose Inputs/Outputs (GPIOs)**.

Preliminary

3.1.1 I/O State

Table 29 shows the default state of the I/Os before and after SYS_RST_N release when all supply rails have reached 90% of level. The I/O states are driven as output if the VCCIO supply is powered up before the VDD core supply.

Table 29: I/O State

IO	VCCIO Bank	WLCSP Ball	BGA Ball	SYS_RST_N = 0	SYS_RST_N = 1
IO<0>	VCCIO<A>	A7	B1	PU	PU
IO<1>	VCCIO<A>	B7	C1	PU	PU
IO<2>	VCCIO<A>	NC	A1	Z	Z
IO<3>	VCCIO<A>	C7	A2	Z	Z
IO<4>	VCCIO<A>	NC	B2	Z	Z
IO<5>	VCCIO<A>	NC	C3	Z	Z
IO<6>	VCCIO<A>	A6	B3	Z	Z
IO<7>	VCCIO<A>	NC	A3	Z	Z
IO<8>	VCCIO<A>	B6	C4	PD	PD
IO<9>	VCCIO<A>	A5	B4	PD	PD
IO<10>	VCCIO<A>	B5	A4	Z	Z
IO<11>	VCCIO<A>	NC	C5	Z	Z
IO<12>	VCCIO<A>	NC	B5	Z	Z
IO<13>	VCCIO<A>	NC	D6	Z	Z
IO<14>	VCCIO<A>	A4	A5	PU	IO<19>=1 & IO<8>=0; PU IO<19>=1 & IO<8>=1; Z
IO<15>	VCCIO<A>	B4	C6	PU	IO<19>=1 & IO<8>=0; PU IO<19>=1 & IO<8>=1; Z
IO<16>	VCCIO<B>	E1	E7	Z	Z
IO<17>	VCCIO<B>	D1	D7	IO<20> = 0; 0 IO<20> = 1; Z	Z
IO<18>	VCCIO<B>	NC	E8	Z	Z
IO<19>	VCCIO<B>	C1	H8	PD	PD
IO<20>	VCCIO<B>	F2	G8	PD	PD
IO<21>	VCCIO<B>	NC	H7	Z	Z
IO<22>	VCCIO<B>	NC	G7	Z	Z
IO<23>	VCCIO<B>	E2	H6	Z	Z
IO<24>	VCCIO<B>	D2	G6	Z	Z

IO	VCCIO Bank	WLCSP Ball	BGA Ball	SYS_RST_N = 0	SYS_RST_N = 1
IO<25>	VCCIO<B>	D3	F7	Z	Z
IO<26>	VCCIO<B>	NC	F6	Z	Z
IO<27>	VCCIO<B>	NC	H5	Z	Z
IO<28>	VCCIO<B>	F3	G5	Z	Z
IO<29>	VCCIO<B>	E3	F5	Z	Z
IO<30>	VCCIO<B>	F4	F4	Z	Z
IO<31>	VCCIO<B>	E4	G4	Z	Z
IO<32>	VCCIO<B>	NC	H4	Z	Z
IO<33>	VCCIO<B>	NC	E3	Z	Z
IO<34>	VCCIO<B>	D5	F3	Z	Z
IO<35>	VCCIO<B>	NC	F2	Z	Z
IO<36>	VCCIO<B>	F5	H3	Z	Z
IO<37>	VCCIO<B>	NC	G2	Z	Z
IO<38>	VCCIO<B>	E6	E2	Z	Z
IO<39>	VCCIO<B>	F6	H2	Z	Z
IO<40>	VCCIO<B>	NC	D2	Z	Z
IO<41>	VCCIO<B>	NC	F1	Z	Z
IO<42>	VCCIO<B>	NC	H1	Z	Z
IO<43>	VCCIO<B>	D7	D1	Z	Z
IO<44>	VCCIO<B>	E7	E1	Z	IO<8>=1; IO<8>=0
IO<45>	VCCIO<B>	F7	G1	Z	IO<8>=1; IO<8>=0

Preliminary

3.2 Electrical Specifications

3.2.1 DC Characteristics

The DC specifications are provided in Table 30 thru Table 33.

Table 30: Absolute Maximum Ratings

Parameter	Value	Parameter	Value
LDO Input Voltage	-0.5 V to 3.6 V	ESD Pad Protection	2 kV
VDD Voltage	-0.5 V to 1.26 V	Laminate Package (BGA) Storage Temperature	-55°C to + 125°C
AVDD/VDDIO Voltage	-0.5 V to 3.6 V		
Input Voltage	-0.5 V to 3.6 V	Latch-up Immunity	±100 mA

Warning: The absolute maximum ratings may cause permanent damage due to EOS S3AI platform. Functional operation of the device should follow the recommended operating range in Table 31.

Table 31: Recommended Operating Range

Symbol	Parameter[a,b,c,d,e]	Min.	Typ.	Max.	Unit
LDO1_VIN	LDO1 input voltage	1.62		3.6	V
LDO1_I	LDO1 analog current consumption – maximum output current set to: 50 mA 8 mA 1 mA		30 10 6		µA
LDO2_VIN	LDO2 input voltage	1.62		3.6	V
LDO2_I	LDO2 analog current consumption – maximum output current set to: 30 mA 8 mA 1 mA		20 10 6		µA
VDD1 Memory (LDO1_OUT)	Supply voltage during active mode	0.95	1.1	1.21	V
	Supply voltage during initialization	1.05	1.1	1.21	V
VDD2 Logic (LDO2_OUT)	Supply voltage during active mode	0.95	1.1	1.21	V
	Supply voltage during initialization	1.05	1.1	1.21	V
VCCIO	Input tolerance voltage	1.71	-	3.6	V
T_J	Ambient temperature	-20	25	85	°C
AVDD	Analog voltage	1.71	-	3.6	V
XTAL_IN	Crystal input	-	32.768	-	kHz
XTAL_IN Low Level	CMOS input low level	-0.3	-	0.35	V

Preliminary

XTAL_IN High Level	CMOS input high level[f]	0.80	-	3.4[g]	V
HSOSC	High speed oscillator frequency	2	20	80	MHz
CMOS Clock Duty Cycle	CMOS clock duty cycle	40	50	60	Percent
CMOS Clock Input Jitter	CMOS clock input jitter	-	-	280	ns

a. Refer to **Low Dropout Regulators** for an explanation of the different LDO configurations.
b. Except where indicated, Min and Max values are tested on 100% of the device at 25°C.
c. Typical values are based on 25°C and nominal voltage (VDD1=VDD2=1.1V, VCCIO=1.8V).
d. Device bootup and initialization should be at 1.1V, and minimum of 1.05V to come out of Power-On reset in LDO Bypass mode. e. LDO1_VIN and LDO2_VIN must be the same voltage.
e. Special analog pad with CMOS tolerant input.
f. The OSCin/out pads are connected to an AVDD supply. Additional current consumption is drawn through the pin when OSCin high level is higher than AVDD.

Table 32: Weak Pull-Up/Pull-Down Characteristics

Parameter	Symbol	Condition	Min.	Typ.	Max.	Unit
Weak Pull-Up Current	I_{PU}	VDDIO = 3.3V VDDIO = 2.5V VDDIO = 1.8V	37 19 16	64 35 32	1 59 58	µA
Weak Pull-Down Current	I_{PD}	VDDIO = 3.3V VDDIO = 2.5V VDDIO = 1.8V	29 14 15	59 31 31	105 59 56	µA
Input Leakage	I_{IH}/I_{IL}		-	-	<1	µA
Short Circuit Current	I_{OSH}	VDDIO = 3.3V VDDIO = 2.5V VDDIO = 1.8V	-	116 72 69	-	mA
Short Circuit Current	I_{OSL}	VDDIO = 3.3V VDDIO = 2.5V VDDIO = 1.8V	-	109 74 68	-	mA

Table 33: DC Input and Output Levels[a]

Symbol	V_{IL}		V_{IH}		V_{OL}	V_{OH}	I_{OL}	I_{OH}
	V_{MIN}	V_{MAX}	V_{MIN}	V_{MAX}	V_{MAX}	V_{MIN}	mA	mA
LVTTL	-0.3	0.8	2.2	VDDIO + 0.3	0.4	2.4	2.0	-2.0
LVCMOS25	-0.3	0.7	1.7	VDDIO + 0.3	0.4	1.8	2.0	-2.0
LVCMOS18	-0.3	0.63	1.17	VDDIO + 0.3	0.45	VCCIO - 0.45	2.0	-2.0

a. The data in this table represents JEDEC specifications. QuickLogic devices either meet or exceed these requirements. Based on weak pull-down I/O termination disabled.

Preliminary

3.2.2 Output Drive Current

Note: The multi-functional IOs have four programmable drive strength states D[1:0]: D00 = 2mA, D01 = 4mA, D10 = 8mA, D11=12mA. The drive strength can be set by programming A0 registers.

Table 34: Output Drive Current (DVDD = 1.8V) in mA

Parameter	Condition	D[1]	D[0]	Min.	Typ.	Max.
I_{OH}	V_{OH} = DVDD – 0.4	0	0	3.42	5.83	9.16
		0	1	6.84	11.7	18.3
		1	0	9.12	15.5	24.4
		1	1	12.5	21.4	33.6
I_{OL}	V_{OL} = 0.4	0	0	4.56	7.86	12.4
		0	1	9.14	15.7	24.8
		1	0	10.1	17.3	27.3
		1	1	14.6	25.2	39.7

Table 35: Output Drive Current (DVDD = 2.5V) in mA

Parameter	Condition	D[1]	D[0]	Min.	Typ.	Max.
I_{OH}	V_{OH} = DVDD – 0.4	0	0	3.60	5.68	8.28
		0	1	5.40	8.53	12.4
		1	0	9.01	14.2	20.7
		1	1	10.8	17.1	24.9
I_{OL}	V_{OL} = 0.4	0	0	4.07	6.65	9.78
		0	1	6.79	11.1	16.3
		1	0	10.9	17.8	26.2
		1	1	13.6	23.3	32.7

Preliminary

Table 36: Output Drive Current (DVDD = 3.3V) in mA

Parameter	Condition	D[1]	D[0]	Min.	Typ.	Max.
I_{OH}	V_{OH} = DVDD – 0.4	0	0	4.94	7.25	9.90
		0	1	7.42	10.9	14.9
		1	0	12.4	18.2	24.8
		1	1	14.8	21.8	29.7
I_{OL}	V_{OL} = 0.4	0	0	5.08	7.91	11.0
		0	1	8.48	13.2	18.3
		1	0	13.6	21.1	29.3
		1	1	17.0	26.4	36.6

3.2.3 Clock and Oscillator Characteristics

Table 37: Clock and Oscillator Characteristics[a]

Symbol	Min.	Typ.	Max.	Unit
XTAL_IN	16	32.768	-	kHz
HOSC	2	20	80	MHz
SPI Master CLK	2	10	20	MHz
SPI Slave CLK	2	10	20	MHz
SWD CLK	2	5	10	MHz
Audio SS (APB Clock)	-	-	10	MHz
PDM Left Clock	-	-	5	MHz
PDM Right Clock	-	-	5	MHz
I^2S Clock	-	-	5	MHz
LPSD Clock	-	-	1	MHz
FPGA Clock	-	-	10	MHz

a. Maximum frequency is with VDD at 1.1V, ±10%.

Preliminary

3.2.4 Output Rise/Fall Time

NOTE: The multi-functional IOs also have programmable slew rates (SRs). The two states are SR = 0 (slow) or SR = 1 (fast) and can be programmed from A0 registers.

3.2.4.1 Output Rise/Fall Time (DVDD = 1.8V)

Table 38: Output Rise/Fall Time (SR = 1, DVDD = 1.8V)

Transition	D[1]	D[0]	C_{LOAD}	Min.	Typ.	Max.	Units
Rise Time PAD↑ (10% to 90%)	0	0	2pF	0.71	1.27	2.23	ns
	0	1	5pF	0.65	1.18	2.16	ns
	1	0	10pF	0.67	1.24	2.30	ns
	1	1	20pF	0.84	1.51	2.81	ns
Fall Time, PAD↓ (90% to 10%)	0	0	2pF	0.60	1.02	1.80	ns
	0	1	5pF	0.57	0.96	1.80	ns
	1	0	10pF	0.70	1.18	2.14	ns
	1	1	20pF	0.84	1.39	2.51	ns

Table 39: Output Rise/Fall Time (SR = 0, DVDD = 1.8V)

Transition	D[1]	D[0]	C_{LOAD}	Min.	Typ.	Max.	Units
Rise Time PAD↑ (10% to 90%)	0	0	2pF	0.74	1.30	2.28	ns
	0	1	5pF	0.70	1.24	2.26	ns
	1	0	10pF	0.77	1.36	2.50	ns
	1	1	20pF	0.92	1.63	2.97	ns
Fall Time, PAD↓ (90% to 10%)	0	0	2pF	0.65	1.13	1.97	ns
	0	1	5pF	0.71	1.20	2.13	ns
	1	0	10pF	0.84	1.41	2.47	ns
	1	1	20pF	1.00	1.66	2.90	ns

Preliminary

3.2.4.2 Output Rise/Fall Time (DVDD = 2.5V)

Table 40: Output Rise/Fall Time (SR = 1, DVDD = 2.5V)

Transition	D[1]	D[0]	C_{LOAD}	Min.	Typ.	Max.	Units
Rise Time PAD↑ (10% to 90%)	0	0	2pF	0.73	1.19	1.96	ns
	0	1	5pF	0.78	1.27	2.15	ns
	1	0	10pF	0.85	1.40	2.42	ns
	1	1	20pF	1.11	1.83	3.14	ns
Fall Time, PAD↓ (90% to 10%)	0	0	2pF	0.65	1.03	1.71	ns
	0	1	5pF	0.68	1.02	1.79	ns
	1	0	10pF	0.77	1.40	2.03	ns
	0	0	20pF	0.73	1.19	1.96	ns

Table 41: Output Rise/Fall Time (SR = 0, DVDD = 2.5V)

Transition	D[1]	D[0]	C_{LOAD}	Min.	Typ.	Max.	Units
Rise Time PAD↑ (10% to 90%)	0	0	2pF	0.80	1.31	2.14	ns
	0	1	5pF	0.90	1.45	2.47	ns
	1	0	10pF	1.13	1.82	3.12	ns
	1	1	20pF	1.43	2.29	3.84	ns
Fall Time, PAD↓ (90% to 10%)	0	0	2pF	0.72	1.16	1.88	ns
	0	1	5pF	0.82	1.29	2.11	ns
	1	0	10pF	1.08	1.69	2.78	ns
	1	1	20pF	1.36	2.08	3.40	ns

3.2.4.3 Output Rise/Fall Time (DVDD = 3.3V)

Table 42: Output Rise/Fall Time (SR = 1, DVDD = 3.3V)

Transition	D[1]	D[0]	C_{LOAD}	Min.	Typ.	Max.	Units
Rise Time PAD↑ (10% to 90%)	0	0	2pF	0.63	0.86	1.42	ns
	0	1	5pF	0.57	0.93	1.57	ns
	1	0	10pF	0.64	1.04	1.76	ns
	1	1	20pF	0.88	1.36	2.25	ns
Fall Time, PAD↓ (90% to 10%)	0	0	2pF	0.59	0.78	1.32	ns
	0	1	5pF	0.52	0.81	1.37	ns
	1	0	10pF	0.61	0.92	1.53	ns
	1	1	20pF	0.82	1.19	1.94	ns

Preliminary

Table 43: Output Rise/Fall Time (SR = 0, DVDD = 3.3V)

Transition	D[1]	D[0]	C_{LOAD}	Min.	Typ.	Max.	Units
Rise Time PAD↑ (10% to 90%)	0	0	2pF	0.66	0.97	1.59	ns
	0	1	5pF	0.72	1.08	1.81	ns
	1	0	10pF	0.90	1.37	2.23	ns
	1	1	20pF	1.14	1.74	2.83	ns
Fall Time, PAD↓ (90% to 10%)	0	0	2pF	0.62	0.91	1.48	ns
	0	1	5pF	0.68	0.96	1.60	ns
	1	0	10pF	0.88	1.52	2.05	ns
	1	1	20pF	1.10	1.61	2.53	ns

3.2.5 Power Consumption

Table 44 to Table 47 shows show power consumption measurements performed on the EOS S3AI reference design board. Developers can expect to measure similar current values at 25°C. Refer to the power optimized schematic available from QuickLogic.

Table 44: Shutdown Current [a]

VDD	1.1V	1.0V	Units
Shutdown LDO Bypass (with external voltage supplied)	12.1	10.5	μA
Shutdown with on-chip LDO Mode[b]	19.0	16.0	

a. Device bootup and initialization should be at 1.1V.
b. LDO mode with maximum output current at 1mA

Preliminary

Table 45 shows Standby Current for LDO Bypass with external voltage supplied.

Table 45: Standby Current [a]

Power Mode	1.1V	1.0V	Units
Standby 32K DS	20.683	17.52	µA
Standby 64K DS	23.967	19.69	
Standby 128K DS	28.833	23.83	
Standby 512K DS	59.303	49.0	
FPGA	60.0	42.0	

a. Standby with SRAM blocks in Deep Sleep (DS). Programmable Logic is in low power mode with retention.

Table 46 shows the CoreMark current consumption at varying frequencies.

Table 46: CoreMark Current Measurement

Frequency in MHz	VDD 1.1V	VDD 1.0V	Units
80	84	75	µA/MHz
40	86	77	

Table 47 shows the EOS S3AI power consumption under various conditions.

Table 47: EOS S3AI Power Measurements

Mode	EOS S3AI Power VDD at 1.0 V	EOS S3AI Power AVDD at 1.8V	Comment
Cortex-M4 with FPU	75 µW/MHz	31µW	Running CoreMark (HSOSC@20 MHz)
Flexible fusion engine	30 µW/MHz	31µW	Running the QuickLogic PCG algorithm (HSOSC@20 MHz)
Always-on audio listening Mode	97 µW	23µW	PDM microphone interface and LPSD Active and Listening (HSOSC@2.1 MHz)
Always-on audio running a fixed trigger	387 µW	31µW	Assumes 12 MIPS on M4, LPSD Active 100% of the time, M4F wakes up 30% of the time to check for a fixed trigger (HSOSC@20 MHz)

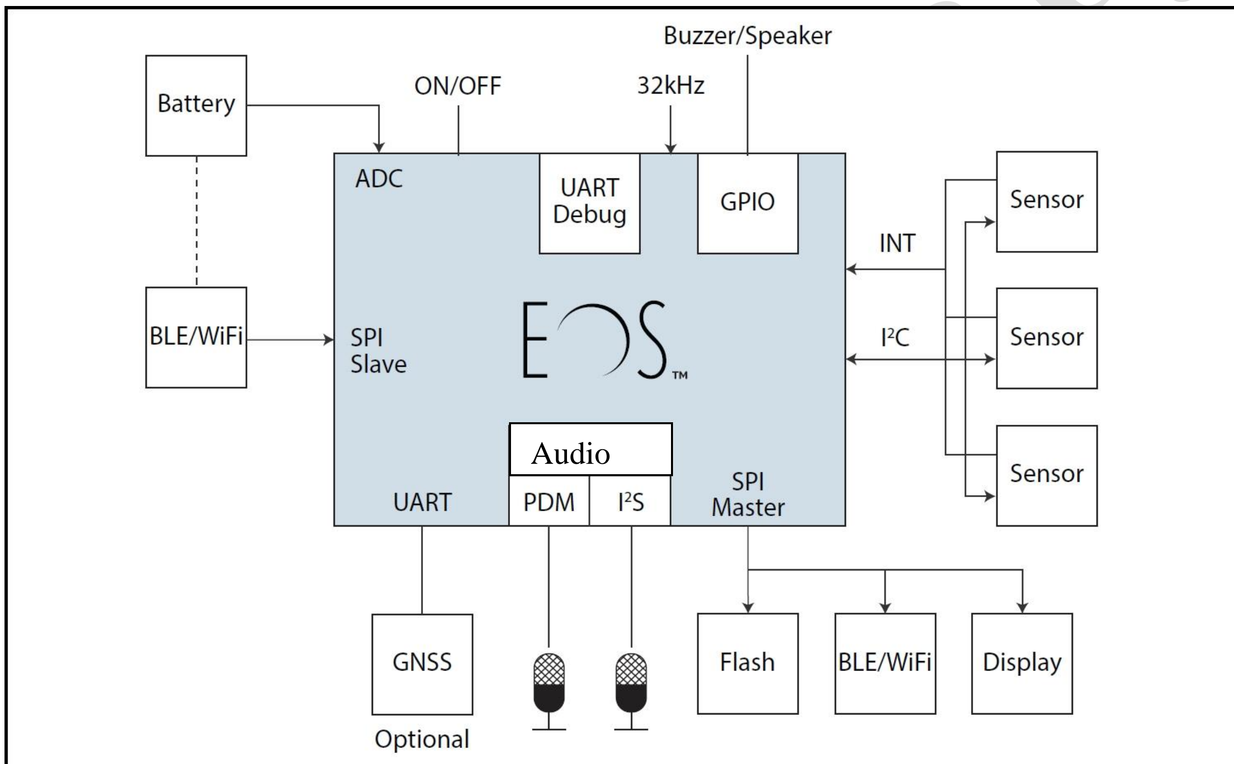
Preliminary

4 Application Examples

Real-Time Operating System IOT Design

Figure 56 illustrates the EOS S3AI platform as a true SoC in a RTOS-based IoT or wearable device. In this use case, the EOS S3AI platform acts as the host processor running the operating system, the always-on, real-time sensor processing, and the interface to the connectivity device(s) in the system.

Figure 56: Example Real-Time Operating System Wearable Design Block Diagram

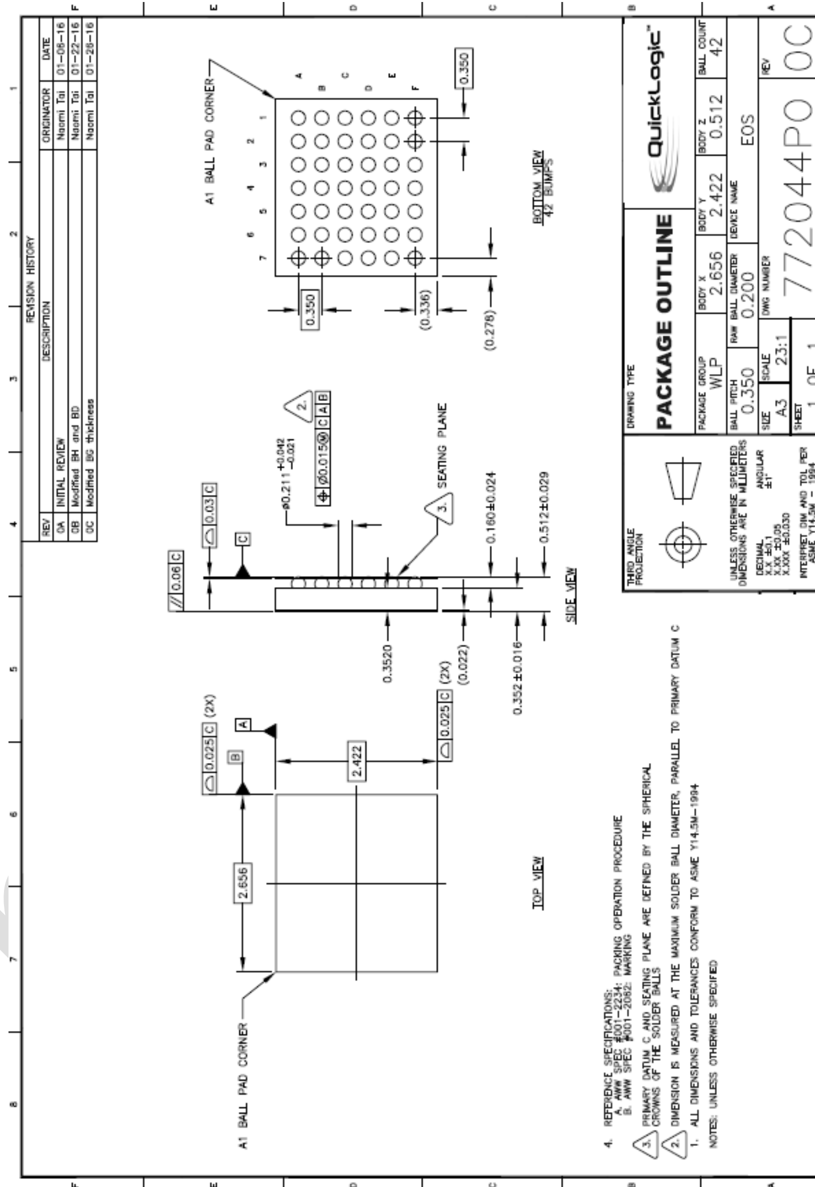


Preliminary

5 Package Information

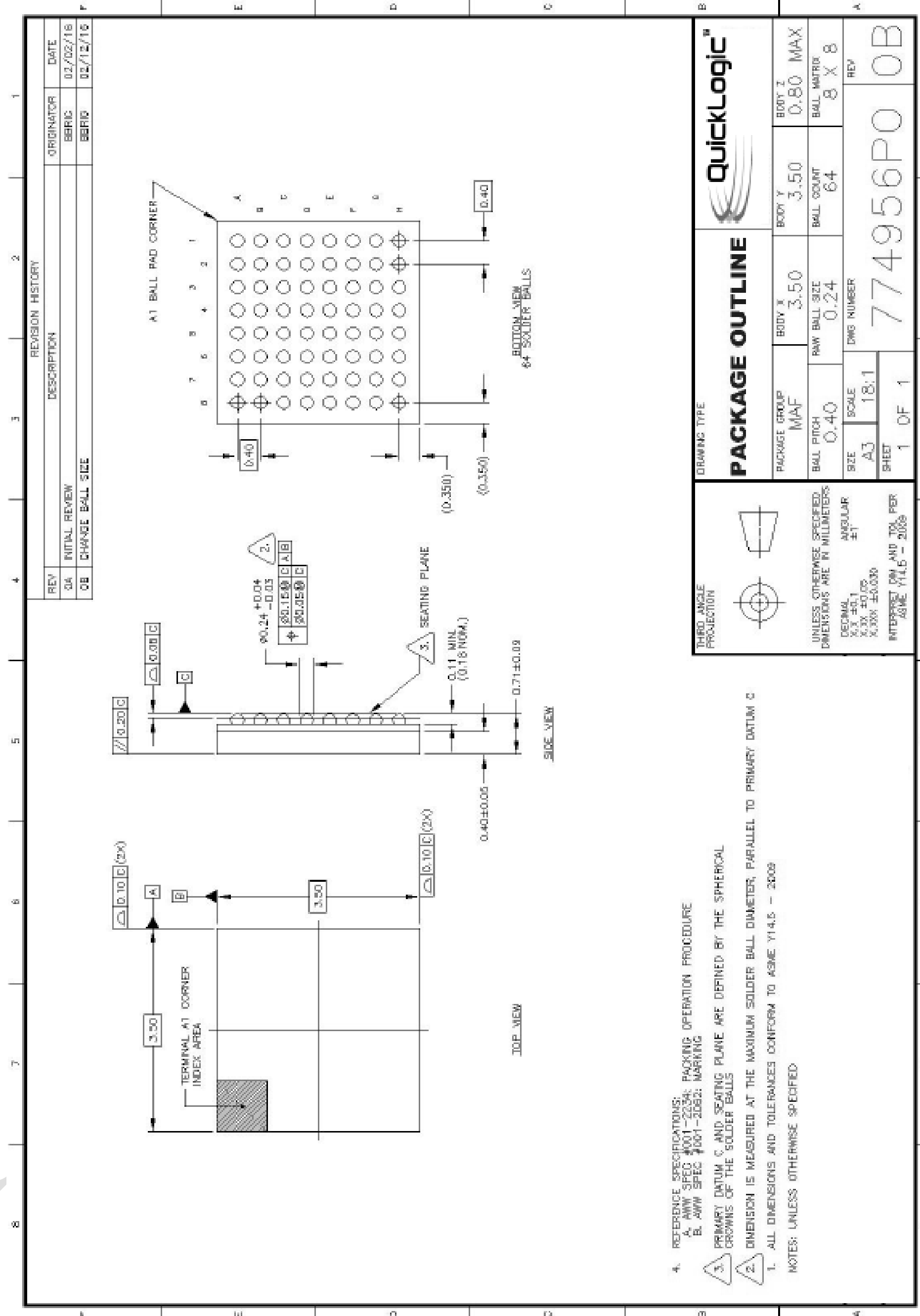
5.1 42-Ball WLCSP Package Information

Figure 57: 42-Ball WLCSP Package Drawing



5.2 64-Ball BGA Package Information

Figure 58: 64-Ball BGA Package Drawing

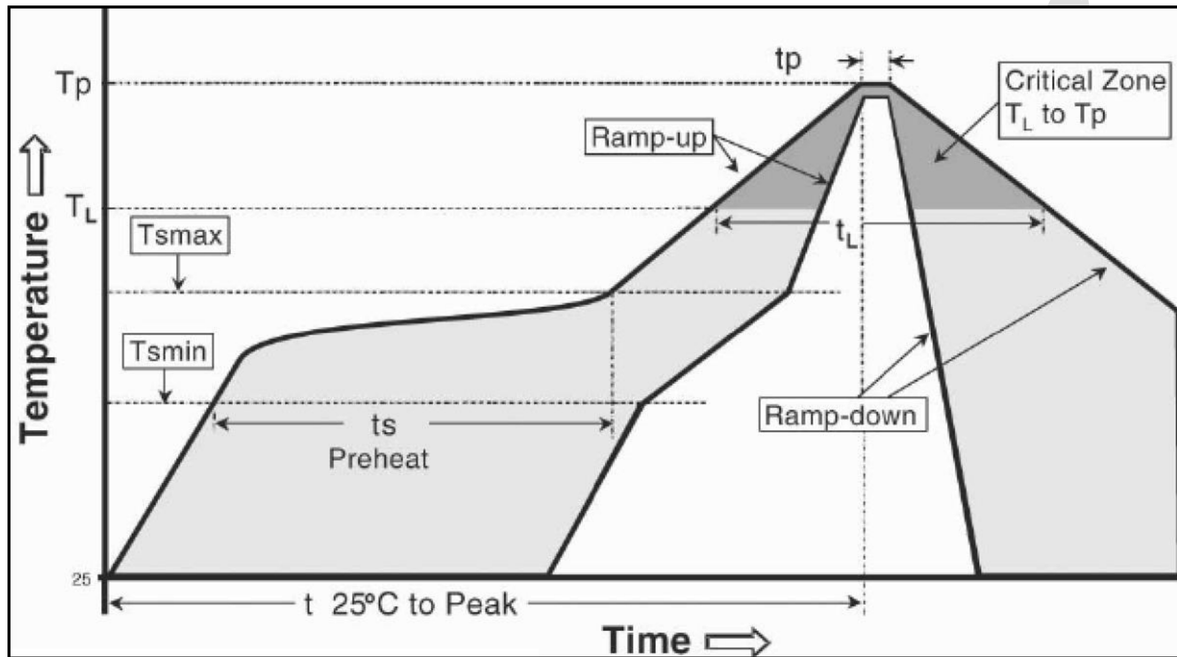


5.3 Soldering Information

5.3.1 Reflow Profile (Preliminary)

QuickLogic follows IPC/JEDEC J-STD-020 specification for lead-free devices. Figure 59 shows the Pb-free component preconditioning reflow profile.

Figure 59: Pb-Free Component Preconditioning Reflow Profile



Preliminary

Table 48 shows the Pb-free component preconditioning reflow profile.

Table 48: Pb-Free Component Preconditioning Reflow Profile[a,b]

Profile Feature	Profile Conditions
Average ramp-up rate (T_{Smax}) to T_P)	3° C per sec. max.
Preheat: - Temperature Min (T_{Smin}) - Temperature Max (T_{Smax}) - Time (T_{Smin} to T_{Smax}) (t_S)	 150°C 200°C 60 sec. to 120 sec.
Time maintained above: - Temperature (T_L) - Time (t_L)	 217°C 60 sec. to 150 sec.
Peak Temperature (T_P)	260°C
Time within 5°C of actual Peak Temperature (260°C)	20 sec. to 40 sec.
Ramp-down Rate	6°C per sec. max.
Time 25°C to Peak Temperature	8 min. max.

[a] The above conditions are used for component qualifications. This should not be interpreted as the recommended profile for board mounting. Customers should optimize their board mounting reflow profile based on their specific conditions such as board design, solder paste, etc.
[b] All temperatures are measured on the package body surface.

Preliminary

5.4 Package Thermal Characteristics

The EOS S3AI sensor processing platform is available for Commercial (-20°C to 85°C) junction temperature ranges.

Thermal Resistance Equations:

$\theta_{JC} = (T_J - T_C)/P$

$\theta_{JA} = (T_J - T_A)/P$

$P_{MAX} = (T_{JMAX} - T_{AMAX})/\theta_{JA}$

Parameter Description:

θ_{JC}: Junction-to-case thermal resistance
θ_{JA}: Junction-to-ambient thermal resistance
T_J: Junction temperature
T_A: Ambient temperature
P: Power dissipated by the device while operating
P_{MAX}: The maximum power dissipation for the device
T_{JMAX}: Maximum junction temperature
T_{AMAX}: Maximum ambient temperature

NOTE: Maximum junction temperature (T_{JMAX}) is 125°C. To calculate the maximum power dissipation for a device package look up θ_{JA} from Table 49, pick an appropriate T_{AMAX} and use:

$P_{MAX} = (125°C - T_{AMAX}) / \theta_{JA}$.

Table 49: Package Thermal Characteristics

Package Description		θ_{JC} (°C/W)	Air Flow (m/sec)	θ_{JC} (°C/W)
Package Type	Pin Count			
WLCSP	42	5.3	0.0	71.5
			0.5	66.4
			1.0	65.0
			1.5	64.1
VFBGA	64	36.3	0.0	69.9
			0.5	67.6
			1.0	66.7
			1.5	66.1

Preliminary

6 Contact Information

Phone: (408) 990-4000 (US)
+(44) 1932-21-3160 (Europe)
+(886) 26-603-8948 (Taiwan)
+(86) 139-0517-5302 (China)
+(81) 3-5875-0547 (Japan)
+(82) 31-601-4225 (Korea)
E-mail: info@quicklogic.com
Sales: America-sales@quicklogic.com
Europe-sales@quicklogic.com
Asia-sales@quicklogic.com
Japan-sales@quicklogic.com
Korea-sales@quicklogic.com
Support: www.quicklogic.com/support
Internet: www.quicklogic.com

Preliminary

7 Revision History

Revision	Date	Originator and Comments
1.0	January 3, 2019	Initial Release for Review
1.1	February 14,2019	Update Audio contents and descriptions

Preliminary

8 Notice of Disclaimer

QuickLogic is providing this design, product or intellectual property "as is." By providing the design, product or intellectual property as one possible implementation of your desired system-level feature, application, or standard, QuickLogic makes no representation that this implementation is free from any claims of infringement and any implied warranties of merchantability or fitness for a particular purpose. You are responsible for obtaining any rights you may require for your system implementation. QuickLogic shall not be liable for any damages arising out of or in connection with the use of the design, product or intellectual property including liability for lost profit, business interruption, or any other damages whatsoever. QuickLogic products are not designed for use in life-support equipment or applications that would cause a life-threatening situation if any such products failed. Do not use QuickLogic products in these types of equipment or applications.

QuickLogic does not assume any liability for errors which may appear in this document. However, QuickLogic attempts to notify customers of such errors. QuickLogic retains the right to make changes to the documentation, specification, or product without notice. Verify with QuickLogic that you have the latest specifications before finalizing a product design.

Preliminary

9 Copyright and Trademark Information

Copyright © 2019 QuickLogic Corporation. All Rights Reserved.

The information contained in this document is protected by copyright. All rights are reserved by QuickLogic Corporation. QuickLogic Corporation reserves the right to modify this document without any obligation to notify any person or entity of such revision. Copying, duplicating, selling, or otherwise distributing any part of this product without the prior written consent of an authorized representative of QuickLogic is prohibited.

QuickLogic is a registered trademark. EOS, the EOS design, and the QuickLogic logo are trademarks of QuickLogic. Other trademarks are the property of their respective companies.

Preliminary