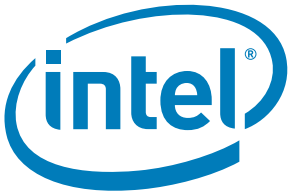


Intel® FM1010

Six-Interface SPI-4.2 Interconnect

Data Sheet

December, 2007 (Revision 2.0)

Legal

Table of Contents

Product Highlights

The FM1010 six-interface System Packet Interface Level 4 Phase 2 (SPI-4.2) interconnect is a high-performance switch-based solution for seamlessly and intelligently interconnecting multiple chips containing the SPI-4.2 interface, enabling designers to deliver sophisticated systems while significantly reducing costs, increasing system flexibility, and reducing time to market. A member of Intel's PivotPoint product family, the FM1010 converts a fixed-configuration half-duplex daisy-chain of devices with SPI-4.2 interfaces (such as NPUs, traffic managers, co-processors, search engines, custom ASICs, and FPGAs) into a dynamically-reconfigurable full-duplex resource pool, enabling more efficient use of the silicon resources, and eliminating the custom glue logic that often accompanies complex system designs.

Applications

The FM1010 can be used at the blade level and for small modular systems (appliances and “pizza boxes”) in high-speed communications and computing applications where the SPI-4.2 interface is used to interconnect the silicon resources in the system, for:

- Ethernet Switches and Routers
- Enterprise Compute Servers
- Enterprise Firewalls and Gateways
- SAN Switches and Gateways
- Multi-Service Platforms
- Enterprise Services Appliances

FM1010 Features

General

- Fully connected non-blocking switch (soft-configurable interfaces and ports)
- Six independent SPI-4.2 interfaces
- 3x internal switch fabric overspeed
- Built-in fine-grained flow control
- 1.2V core, 2.5 V LVDS and 3.3 V LVTTL operation

Test and Control Features

- 16-bit standard CPU interface
- JTAG and boundary scan support
- Support for loopback on each interface

Physical

- Modest and flexible power profile
- 2 W (avg.) per active interface

Interface Features

- Each interface consists of a simplex RX and TX path, each of which is independently clocked
- Each interface can support data rates up to 14.4 Gb/s
- Support for dynamic and static phase alignment
- LVTTL status channel
- Configurable packet buffer (16KB ingress; 16KB egress) per interface

Up to 256 flow controlled ports

- Independent enable/disable control for each interface
- Statistics and error reporting
-

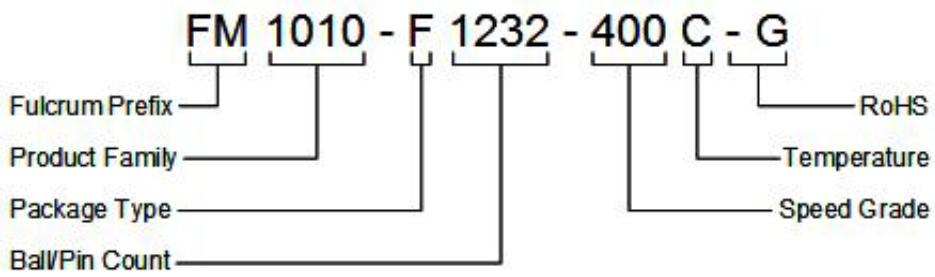
- Power scales linearly on activity
- 130 nm CMOS process technology
- 1232-ball (and 1036-ball) BGA package

Document Revision History

Revision	Date	Notes
2.0	Dec 19, 2007	Details in Section 7.6
1.41	Jan 13, 2006	Details in Section 7.5
1.4	Dec 19, 2005	Details in Section 7.4
1.3	May 20, 2005	Many changes. See Section 7 for details.
1.2	April 28, 2005	IGNORE_DS_N pin changed sense. Made power supply names consistent.
1.1	Feb 11, 2005	Part numbers updated, and new package information added. Several register descriptions updated. Specification updates described in document FM1010-SU-001 incorporated.
1.0	Sept 28, 2004	Characterization and qualification data incorporated. All features and operations have been verified.
0.9	Feb 10, 2004	Advance information datasheet.
0.2	July 7, 2003	Preliminary datasheet. Limited distribution.

Product Applicability

This data sheet documents the features and functionality of all generally-available versions of the PivotPoint FM1010 product family. The PivotPoint FM1010 part number is structured as follows:



Unless specifically noted, the contents of this data sheet apply to all product variants, which include all versions of FM1010-F1036 and FM1010-F1232.

For documentation on the original FM1010 package based on wire-bond package technology (labeled as “FM1010”), refer to the following documents:

- FM1010-DS-1.0: FM1010 Data Sheet, version 1.0
- FM1010-SU-001: FM1010 Specification Update, version 001

- FM10x0-DG-1.0: FM1010 and FM1020 Design Guide, version 1.0

Other Related Documents and Tools

Other documents that may be useful for evaluating and using the FM1010 include:

- FM1010 Specification Update, which contains errata and other specification and documentation changes (document: FM1010-SU)
- FM1010 and FM1020 Design and Layout Guide (document: FM10x0-DG)
- FM1010 and FM1020 Evaluation Platform Data Sheet (document: FM10x0-EP-DS)
- FM1010 Hardware Development Kit CD (order number: FM1010-HDK)

1.0 General Description

1.1 FM1010 Overview

The PivotPoint FM1010 is a configurable six-interface System Packet Interface Level 4 Phase 2 (SPI-4.2) switch chip, which provides configurable connectivity between a variety of chips in a system at full line rate and with minimal additional latency. The switching is accomplished on an interface or port / interface pair basis based on a software-configurable port mapping. No inspection or interpretation of the payload within the SPI-4.2 packet is required, except to the extent required to determine the SPI-4.2 port identification.

Each SPI-4.2 electrical interface operates up to 450 MHz (approximately 14.4 Gb/s throughput). Each SPI-4.2 interface is composed of two simplex interfaces (a transmit and a receive), each of which may operate at an independent frequency from all other interfaces. The calendar status channel for each SPI-4.2 interface is a 1/4 frequency LVTTL implementation. The calendar length, multiplicity, and entries are configured when the SPI-4.2 interface is initialized or reset.

Each SPI-4.2 interface on the FM1010 supports three modes of operation: Clear (where a single port is active on the interface), and multi-port mode (where multiple ports are active, up to 16 ports per interface), and extended multi-port mode (where multiple ports are active, up to 256 ports per interface).

Clear Mode: In Clear mode, the entire SPI-4.2 interface is mapped to a single port and the port ID field in the SPI-4.2 packet is ignored.

Multi-port Mode: Multi-port mode routes all packets with different SPI-4.2 port IDs independently of the others as if the packets were sent over unrelated interfaces. There is no ordering relationship, and thus no blocking between SPI-4.2 packets with different port IDs. All of the flow control information is maintained for each active port, and thus, to be effective, the external SPI-4.2 devices must support multi-port operation. The FM1010 contains up to 16 FIFOs per interface and thus supports up to 16 ports per interface, each of which can be any available port of the 256 total port IDs specified in the OIF SPI-4.2 Implementation Agreement.

Extended Multi-port Mode: Extended Multi-port Mode operates identically to Multi-port mode, with the extended ability to support up to 256 ports per interface, mapped as appropriate to the 16 available FIFOs per interface.

Additionally, the FM1010 can be configured to reassign port IDs as packets flow through the system.

The FM1010 also contains a standard 16-bit address/data processor bus interface that is used to read and write all Control Status Registers that control the chip configuration and operation, and also to obtain status and to debug the chip. This CPU interface can be configured to support a variety of commercial processors including the Motorola MPC8260, and MPC860, IBM PowerPCs with an EBC bus, and various I/O bridge chips (such as the PLX 9030 PCI bridge chip from PLX Technologies). The different modes are supported through pin strapping options. This CPU interface operates up to 100 MHz.

Lastly, the FM1010 implements an industry-standard JTAG controller for test and design debug. The JTAG controller can access boundary scan registers and all internal registers.

A block diagram of the FM1010 is shown in Figure 1.

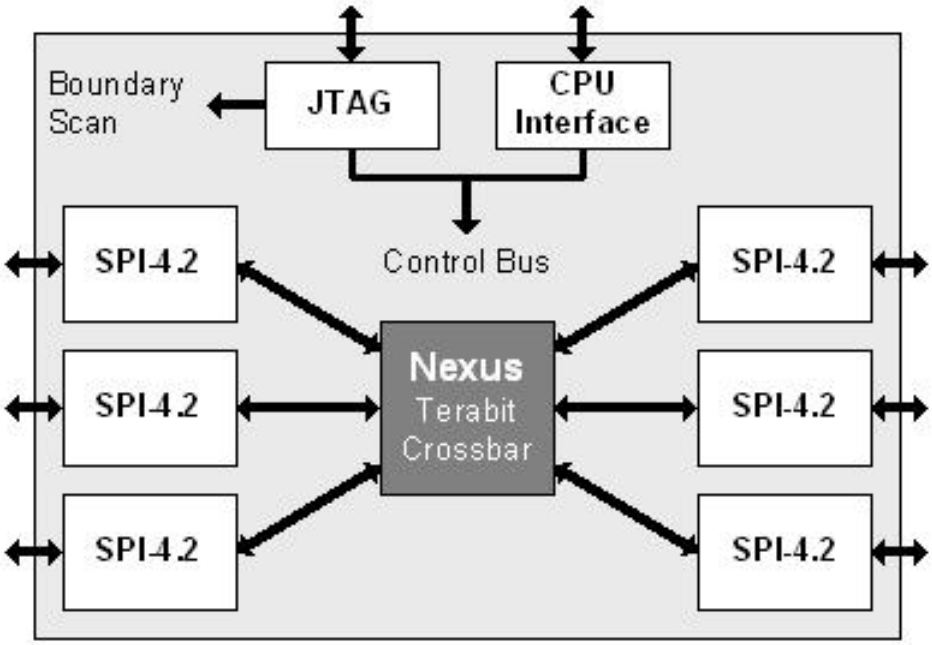


Figure 1. FM1010 Block Diagram

The FM1010 contains six independent SPI-4.2 interfaces connected together through Nexus (Intel's terabit crossbar). An integrated JTAG interface provides standards-based test and debug, and a generic CPU interface provides access to the FM1010 Configuration and Status Registers via a variety of standard processors.

1.2 FM1010 Application Example

The FM1010 enables as many as six SPI-4.2-compliant devices to be interconnected at rates up to 14.4 Gb/s per interface. The FM1010 transparently handles rate mismatch between the different interfaces

with a combination of low-latency buffers and end-to-end flow control. Simple interface and port assignment through a standard CPU interface allows the system to be reconfigured for different applications, allowing the hardware to optimally match the application.

The FM1010 is configured by the Control CPU by writing its control registers through the CPU interface. Each SPI-4.2 interface may operate in clear or multi-port mode independently. Links are established between attached SPI-4.2 devices through port assignment. Multiple ports may be used to support QoS differentiated links between the same two external SPI-4.2 devices provided that the external devices have an ability to assign and support multiple ports within a SPI-4.2 interface. The FM1010 interface and port assignment may be soft reconfigured during device operation.

An FM1010 system-level block diagram of a sample generic “blade” is shown in Figure 2.

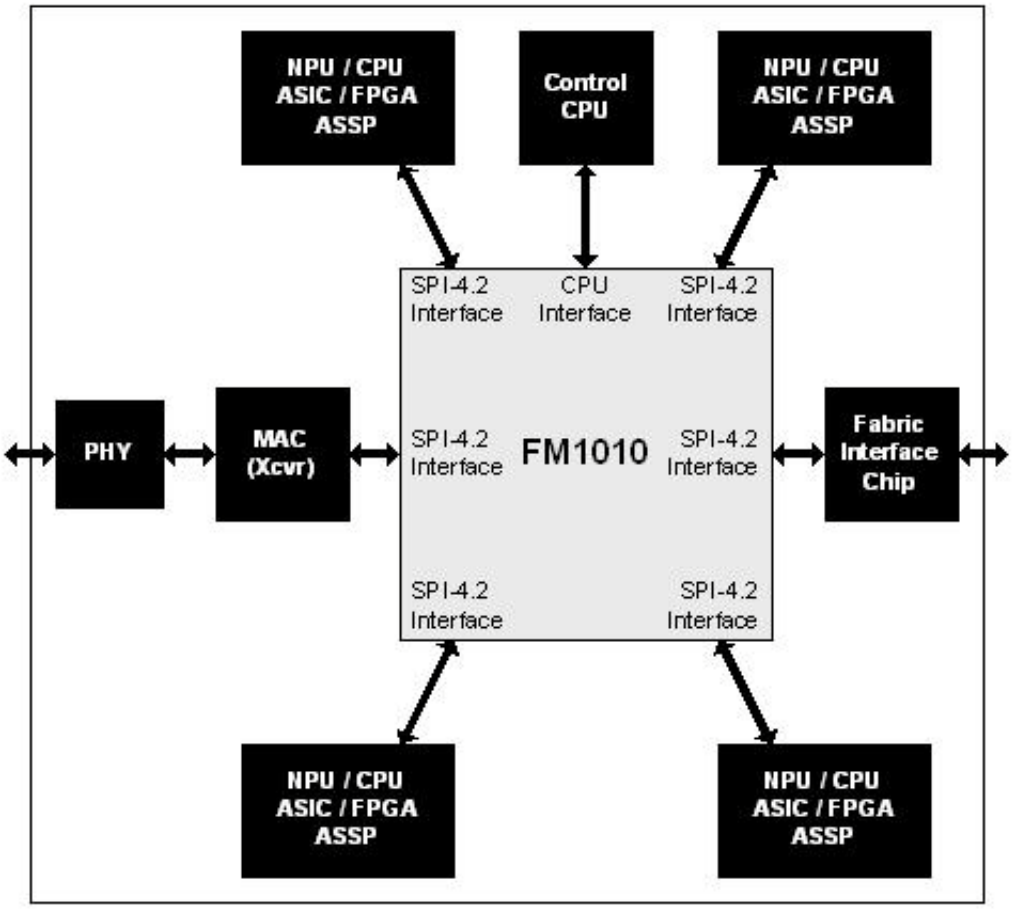


Figure 2. FM1010 System-Level Block Diagram (Sample Generic “Blade”)

The FM1010 can be integrated into a blade or “pizza box” application and convert a fixed hardware configuration into a soft-configurable pool of silicon resources. Low latency, high throughput, and independent interfaces with independent clock rates make the FM1010 a versatile platform for rapidly delivering configurable hardware platforms.

1.3 Definitions

The following are terms that are relevant for the FM1010, and which are used throughout this document to describe the features, functions, configuration, and use of the FM1010.

1.3.1 Terms Defined in the OIF SPI-4.2 Implementation Agreement

Figure 3 depicts the components defined in the Optical Internetworking Forum (OIF) System Packet Interface Level 4 Phase 2 (SPI-4.2) Implementation Agreement. The terminology defined by the OIF is used throughout this document, including the term “port” which refers to a logical partition (or allocated portion) of a physical interface.

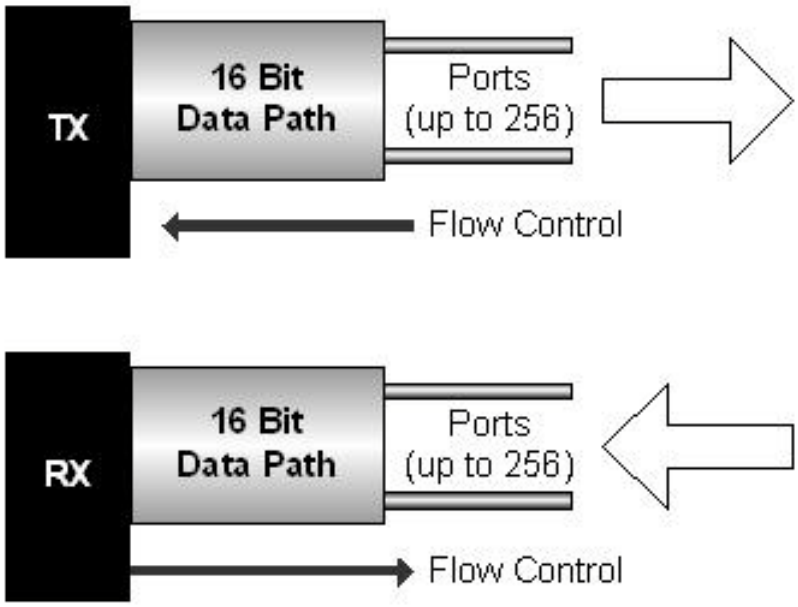


Figure 3. Components of OIF SPI-4.2 Implementation Agreement (Specification)

The Optical Internetworking Forum defined the SPI-4.2 interface originally for interconnecting physical layer devices with link layer devices. Due to the versatility defined into the interface (including multiple ports, each with independent flow control) designers have extended their use of SPI-4.2 to the entire data path in a variety of high-speed streaming data applications.

Interface	A single SPI-4.2 physical implementation containing a transmit and receive data path and a separate status interface for flow control for each direction.
Port	A logical partition (or allocated portion) of a physical interface. A physical SPI-4.2 interface can contain up to 256 logical ports. Generally, status control is supported on a per-port basis so that each interface can support multiple flows without ports blocking each other.
Channel	A commonly used synonym for “Port”. Not defined in the OIF SPI-4.2 Implementation Agreement.
Status Interface	A separate path between sending and receiving SPI-4.2 interfaces used to provide flow control and to report the status of FIFOs in the connected interfaces.
Flow Control	Supported through the Status Interface, as described above.

1.3.2 Other Relevant Terms

CSR (Register)	Control Status Register used for configuration, status reporting, and debug.
Nexus	Intel's Terabit fully-connected non-blocking crossbar; Nexus is the core of PivotPoint.
EOP	End of Packet
SOP	Start of Packet
Queue	A temporary packet storage element in the SPI-4.2 interface (a.k.a., FIFO). In the FM1010, queues are located on the transmit and receive data paths of each SPI-4.2 interface.
Clear	A mode of operation on the SPI-4.2 interface where the Port ID is ignored and all of the data on an interface is transported as a single stream to either a single egress port or an egress interface.
Multi-Port	A mode of operation on the SPI-4.2 interface where each interface can contain multiple active ports which can be mapped independently of each other to any available port on any other interface. In this mode, the Port IDs are used to route data from ingress queue to egress queue through the FM1010, and flow control is enforced on a port-by-port basis. (Extended multi-port mode leverages the 16 FIFOs available per interface to support up to 256 ports per interface.)
Cut Through	A switching mode or architecture where the packet can be delivered to the destination port without being stored, as soon as the destination address can be determined from the packet header.

2.0 Theory of Operation

SPI-4.2 is a point-to-point streaming interface standard defined by the Optical Internetworking Forum for efficiently connecting together high-speed data path devices. The interface operates from 9.9 Gb/s to 16 Gb/s, having a 16-bit LVDS data path and a separate out-of-band LVTTL status channel. Since the data path is dual data rate, the interface operates from 311 to 500 MHz. The protocol supports up to 256 ports per interface, each of which can optionally be independently flow-controlled.

The FM1010 is a transparent SPI-4.2 interconnect. When inserted between two neighboring SPI-4.2 devices, it can switch each port on each interface independently and completely transparent to the neighboring devices. The FM1010 bridges between the flow control domains on its ingress and egress paths using Intel's high-capacity non-blocking crossbar circuit, Nexus.

Each SPI-4.2 interface of the FM1010 implements a transmit interface and a receive interface that operate independently as two simplex interfaces, each with full buffering (ingress and egress queues) and flow control. Packet data is transported efficiently from any interface (and port) to any other interface (and port) through Intel's fully-connected, non-blocking crossbar - Nexus - using an efficient and low-overhead message-passing system. The internal protocol and switching system is transparent to the application. Its operation is described in this document only to the extent that is required for the designer to use the FM1010 in application.

Each SPI-4.2 queue must have the routing configuration -- the pairing of ingress queues to egress queues -- pre-configured. There must be a unique pairing for each active egress queue to an appropriate ingress queue. Each queue has a producer and consumer pointer, which is passed across the Nexus as needed. For each ingress queue, the destination interface and egress queue are configured. Similarly the destination egress queue must be configured to identify where the consumer pointer updates shall be sent. All packet switching within the FM1010 is based on the user-defined queue-to-queue port map, which begins switching once the ingress burst has been placed in an ingress queue.

2.1 Operating Modes

The SPI-4.2 interface supports three modes of operation:

- Clear mode (where an entire interface is connected to a second interface or a single port on an interface with no port relevance)
- Multi-port mode (where packets are routed between multiple SPI-4.2 interfaces based on Port ID, where each interface can support up to 16 active and independent ports)

- Extended Multi-port mode (where packets are routed between multiple SPI-4.2 interfaces based on Port ID, where each interface can support up to 256 active and somewhat independent ports)

Note: The mode of operation can be configured per SPI-4.2 interface, and per transmit and receive block. The FM1010 can support mixing modes among different interfaces, given certain operating constraints - described later in this section.

2.1.1 Clear Mode

In Clear mode, static routing is used (configured on a per-ingress-interface basis) to determine to which egress SPI-4.2 interface (and optionally to which port) the packet is to be delivered, as shown in Figure 4. This mode is constrained to a one-to-one correspondence between an egress interface and an ingress interface (that is no more than one ingress interface may be mapped to a given egress interface). The Port ID information is transported with the packet, but is not used for any routing decisions.

The egress SPI-4.2 interface will send out the packet in configurable-sized SPI-4.2 bursts as credits are available from the SPI-4.2 status channel. The FM1010 is cut-through in its operation. The only restriction is that the complete packet must be transmitted before switching ports, a restriction which is inherently met because the ingress interface receives (and enforces via packet termination if necessary) complete packets per Port ID, and because there is only a one-to-one configured relationship between an egress interface and a single ingress interface.

All incoming SPI-4.2 bursts are stored in a single incoming RX queue in the SPI-4.2 interface. The ingress sends out status for a single queue (FIFO) on the SPI-4.2 status channel. The ingress queue may be configured as a single 1K, 2K, 4K, 8K or 16K Byte queue.

In this mode all port information is ignored.

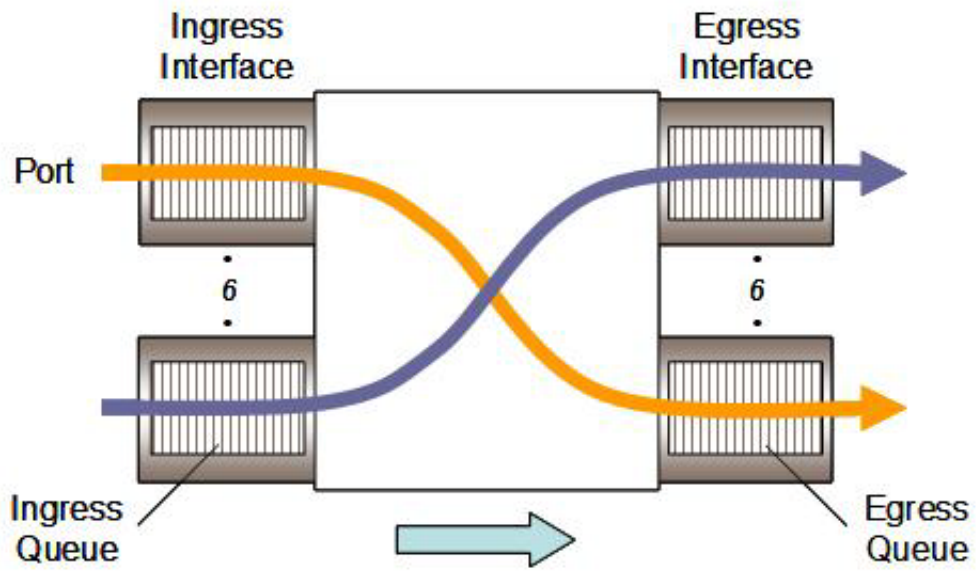


Figure 4. Clear Mode Supported within the FM1010

In this example, all of the interfaces are configured in Clear mode, where a single queue is defined for each interface. In Clear mode, the queue of an ingress interface is mapped to the queue of an egress interface. This mode is the most efficient method for transporting all of the traffic from one chip to another, where both chips are configured with link-level flow control.

2.1.2 Multi-Port Mode

Multi-port mode offers the most flexibility in operation, and is the default mode of the FM1010. In Multi-port mode, each SPI-4.2 transmit and receive interface can support up to 16 fully-flow-controlled SPI-4.2 ports, as shown in Figure 5. The ports can be any 16 of the Port IDs from 0 to 255; they don't need to be contiguous, nor defined in any particular order. Each configured port has a dedicated ingress and egress queue. The size of each queue depends on the number of active ports defined per interface.

The SPI-4.2 status calendar length, multiplicity and entry organization can be set to accommodate ports operating at different rates. A calendar length of up to 255 is supported. SPI-4.2 credit status is distributed on a per-port basis according to the calendar. Packets being sent over the SPI-4.2 interface can be sent as interleaved SPI-4.2 bursts of different ports per the OIF SPI-4.2 specification.

In Multi-port mode, the size of the ingress and egress queues must be set, generally according to the number of active ports being supported per interface. Each SPI-4.2 ingress and egress interface can be configured differently; the interfaces need not be symmetrically

configured. The size may be set to 1K, 2K, 4K, 8K, or 16K Bytes per queue, which corresponds to a maximum of 16, 8, 4, 2, or 1 active ports, respectively. Note the word maximum, as fewer ports may be configured and in use than the maximum. Refer to **RX_FS** (page 46) for correspondence between number of FIFOs and the FIFO size.

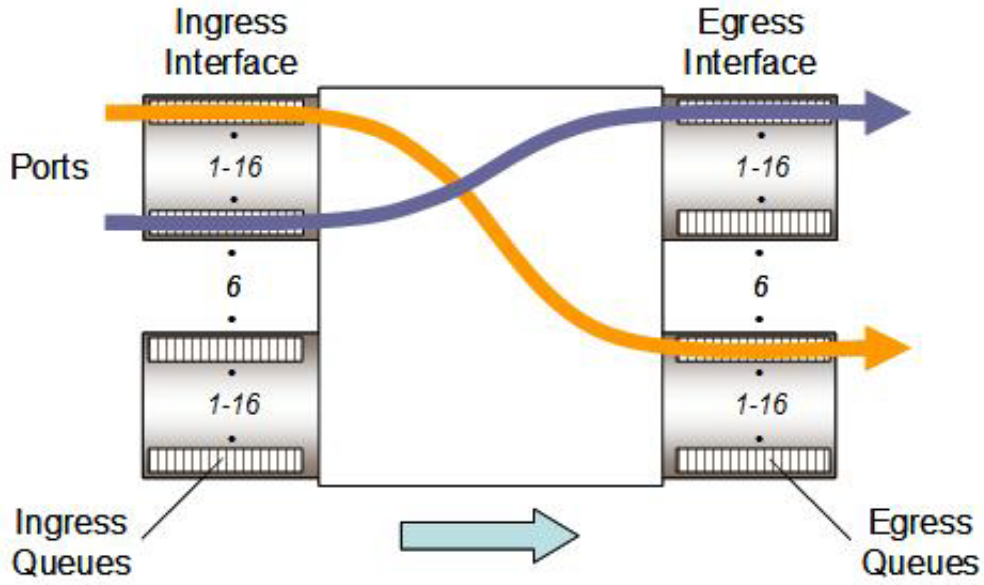


Figure 5. Multi-port Mode Supported Within the FM1010

In this example, all of the interfaces are configured in Multi-port mode, where each ingress interface is configured to support 1, 2, 4, 8, or 16 ports, each egress interface is also configured independently to support 1, 2, 4, 8, or 16 ports, and the queues for each ingress interface are mapped to the appropriate queues for each egress interface. This mode can be used to support source routing between connected chips, where each chip can assign a Port ID to transmitted packets which get switched by the FM1010 and delivered to the appropriate egress interface and port.

2.1.3 Extended Multi-Port Mode

Extended multi-port mode operates identically to multi-port mode, with the additional ability to map multiple ports to a single FIFO, allowing an interface to support up to 256 active ports that are mapped to up to 16 active FIFOs per interface. This mode can be used in applications where a relatively large number of moderately-active ports are supported.

2.2 Mixing Operating Modes

The FM1010 supports mixing modes of operation within certain constraints, as follows:

- An ingress interface configured in Clear mode may be mapped to a single port on an egress interface configured in Multi-port mode.
- A single port on an ingress interface configured in Multi-port mode may be mapped to an egress interface configured in Clear mode.

Figure 6 provides further clarification.

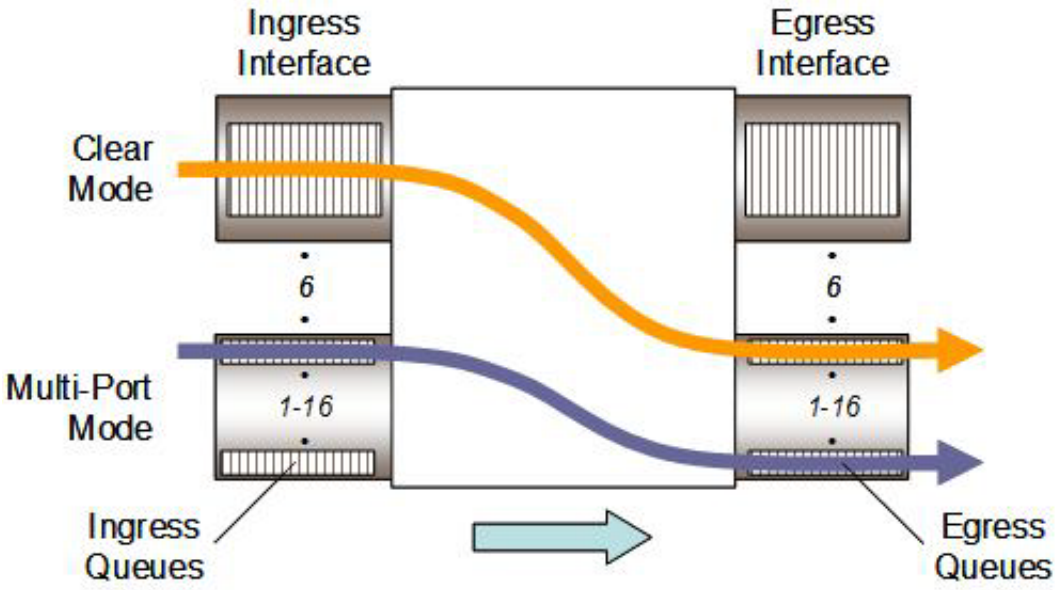


Figure 6. Mapping Between Clear and Multi-port Interfaces

In this example, an ingress interface in Clear mode is connected to an egress interface in Multi-port mode, and vice versa. In the first example, the single queue of the ingress interface in Clear mode is mapped to a single available queue on the egress interface in Multi-port mode. In the second example, a single available queue on the ingress interface in Multi-port mode is mapped to the single queue of the egress interface in Clear mode. This configuration flexibility enables designers to support special applications such as multiplexing streams from multiple devices onto a single interface for delivery to a single connected device. Once a clear interface is mapped to a port of a Multi-port interface, the Port IDs of the stream from the clear interface to the multi-port interface are overwritten with the Port ID of the mapped port of the Multi-port interface. In the other direction, the Port IDs on the clear interface is set to 0.

3.0 Functional Descriptions

This section describes the high-level functionality, including basic configuration guidelines, for the FM1010.

There are four main tasks associated with configuring the device:

1. Reset the Chip
2. Configure the SPI-4.2 interfaces
3. Map SPI-4.2 ports to FIFOs
4. Map FIFOs to FIFOs

The following sections describe these tasks in detail and in the context of global, mode-independent, and mode-dependent start-up parameters.

Additionally, there are registers used for statistics and interrupts. And finally there is a procedure for link reconfiguration, a feature that allows channel remapping during device operation.

Table 1 shows a list of registers and the basic description of their role in the FM1010. Refer to Chapter 5 for more detailed register information.

Table 1. FM1010 Register Summary

Global Registers		
Name	**Table Ref**	**Description**
VPD	Table 19	Vital Product Data
INTR_DETECT	Table 18	Interrupt Detect
CPU_INTERFACE_IM	Table 16	CPU Interface Interrupt Mask
CPU_INTERFACE_IP	Table 17	CPU Interface Interrupt Pending
SPI4_WATCHDOG	Table 15	SPI.4-2 Watchdog configuration

SPI-4.2 Receive Interface Registers (one set per SPI-4.2 interface)		
Name	**Page Ref**	**Description**
RX_RESET	43	Receive Interface Reset
RX_CAL_LM	43	Receive Calendar Length and Multiplicity
RX_SYNC	44	Receive Interface synchronization status
RX_DESKEW1	44	Static Deskew Configuration #1
RX_DESKEW2	44	Static Deskew Configuration #2
RX_DESKEW3	45	Static Deskew Configuration #3
RX_CALS[j] {0..31}	45	Receive Interface Calendar
RX_FS	46	Receive FIFO Size
RX_OP_MODE	46	Receive Interface Operational Mode
RX_WATERMARK	46	Receive FIFO Watermark
RX_PORT2FIFO[j] {0..31}	47	Receive Port to FIFO Mapping

SPI-4.2 Receive Interface Registers (one set per SPI-4.2 interface)		
Name	**Page Ref**	**Description**
RX_PORT_VALID[j] {0..7}	47	Receive Port Usage
RX_STATUS_OVERRIDE	48	Receive FIFO Flow Control Override
RX_OS	49	Receive Interface Status
RX_LINKCFG1[j] {0..15}	49	Mapping from RX FIFO to TX FIFO
RX_LINKCFG2[j] {0..15}	50	Additional Information in FIFO Map
RX_LINKRESET	50	Receive Link Reset
RX_PKTCNT	51	Receive Packet Count Statistics
RX_PKTERRCNT	51	Receive Error Packet Count Statistics
RX_DATACNT	52	Receive Byte Count Statistics
RX_IP	52	Receive Interrupt Pending
RX_IM	53	Receive Interrupt Mask
RX_DEBUG_STATUS	53	Receive Interrupt Status

SPI-4.2 Transmit Interface Registers (one set per SPI-4.2 interface)		
Name	**Page Ref**	**Description**
TX_RESET	55	Transmit Interface Reset
TX_CAL_LM	55	Transmit Calendar Length and Multiplicity
TX_SYNC0	55	Transmit Synchronization Configuration
TX_SYNC1	56	Transmit Interface's Training Interval
TX_CORE_WATERMARK	56	Reserved
TX_CALS[j] {0..31}	57	Transmit Calendar
TX_FS	57	Transmit FIFO Size
TX_OP_MODE	58	Transmit Operational Mode
TX_SERVICE_LIMIT	58	Transmit Service Limit per FIFO
TX_MAX_BURST[j] {0..15}	59	Transmit Credit Refill per Status
TX_FIFO2PORT[j] {0..15}	59	Transmit FIFO to SPI-4.2 Port Map
TX_FIFO_VALID	59	Transmit FIFO Usage
TX_OS	60	Transmit Interface Operational Status
TX_LINKCFG[j] {0..15}	60	Transmit FIFO to Receive FIFO Map
TX_PKTCNT	61	Transmit Packet Count Statistics
TX_PKTERRCNT	61	Transmit Error Packet Count Statistics
TX_DATACNT	61	Transmit Byte Count Statistics
TX_IP	62	Transmit Interface Interrupt Pending
TX_IM	62	Transmit Interface Interrupt Mask
TX_PLL_CTRL	62	Transmit PLL Control
TX_PLL_STAT	63	Transmit PLL Statistics

3.1 Chip Reset and Configuration

This section provides details on how to reset and configure the FM1010 globally, and for various modes of operation.

3.1.1 Chip Reset Parameters

Table 2 lists a summary of all reset signals associated with the FM1010. Please see the appropriate sections for a detailed description of the functionality of each.

Table 2. FM1010 Reset Signals

Name	Type	Description / Comments
CHIP_RESET_N	Chip Input	Chip reset that resets all logic. Active low.
CPU_BUS_RESET_N	Chip input	Electrical Reset for the CPU bus interface only. Active low. This is a subset of CHIP_RESET_N. Asserting CHIP_RESET_N automatically resets the CPU bus.
TX_PLL_CTRL	Register (1 per SPI-4)	Control register for PLL in the TX SPI-4.2 interface.
TX_PLL_STAT	Register (1 per SPI-4)	Status register for PLL in the TX SPI-4.2 interface.
RX_RESET	Register (1 per SPI-4)	Resets a RX SPI-4.2 interface. Active high. This register defaults to asserted on chip reset.
TX_RESET	Register (1 per SPI-4)	Resets a TX SPI-4.2 interface. Active high. This register defaults to asserted on chip reset.

3.1.2 Reset and Configuration Procedure

The following is the recommended procedure for resetting and configuring the FM1010. The procedure is generally accomplished in three phases: (1) Reset; (2) Mode-independent configuration; and (3) Mode-dependent configuration, as follows:

1. Apply power to the FM1010 and IO pins keeping **CHIP_RESET_N** asserted.
2. Wait a minimum of 100 ns before de-asserting **CHIP_RESET_N**, and then wait at least 4 full CPU Bus clock cycles before activating the CPU Bus functional signals.

Note: If using JTAG, also wait 4 JTAG clock cycles after CHIP_RESET before operating the JTAG interface.

3. Per active SPI-4.2 interface, configure the PLL controls in **TX_PLL_CTRL** (page 62).
4. Check the operation of the PLL by polling the **TX_PLL_STAT** (page 63) to determine whether the lock bit has been set. The lock bit signifies that the PLL has properly locked on the requested frequency.

Notes: (1) One must wait at least 100ms after configuring the PLL control registers before polling the signal **TX_PLL_STAT** to allow it to stabilize.

(2) The Lock Pins can be used instead of polling TX_PLL_STAT to determine whether the PLL has properly locked on the requested frequency. The pins can also be used to drive status LEDs.

5. Configure the desired SPI-4.2 physical layer and SPI-4.2 mode-dependent switching layer registers (page 21), while leaving **TX_RESET** and **RX_RESET** asserted.
6. After the set-up configuration, enable the corresponding SPI-4.2 interfaces by de-asserting the **RX_RESET**s and **TX_RESET**s. This will start training on the SPI-4.2 physical interfaces to gain synchronization.

Notes: (1) There is no order dependency between RX_RESET and TX_RESET; they are independent interfaces, and can be de-asserted in either order.

(2) It is recommended that the TDCLK on the TX interfaces of the devices connected to PivotPoint be present prior to de-asserting the RX_RESET on PivotPoint to ensure that the RDCLK is active. If the RDCLK on an interface is inactive, the watchdog must be disabled when de-asserting RX_RESET. Otherwise, if the RDCLK is active, the watchdog state doesn't matter.

(3) The FM1010 is operational once the training sequences are complete and synchronization has been obtained on the SPI-4.2 interfaces. This can be determined by reading the status registers, **RX_OS** (page 49) and **TX_OS** (page 60).

3.1.3 Mode-Independent Global Start-up Parameters

Note that all interrupt mask registers, including **CPU_INTERFACE_IM** (page 41), **RX_IM** (page 53), and **TX_IM** (page 62) default to masks on, and should only be set to masks off during device operation to avoid spurious interrupts that would occur during start-up.

Set the **SPI4_WATCHDOG** register (page 41) to activate the watchdog on all active SPI-4.2 ports. Some control and status register operations require the SPI-4.2 watchdog to be active if the link is not yet in operation. Activating the watchdog will have no undesirable effects on interrupts during setup as long as the interrupt mask registers have not yet been set to the active interrupt state.

Note: The SPI4_WATCHDOG uses the least significant 6 bits, where bit 0 enables the watchdog for interface 0, bit 1 enables the watchdog for interface 1, and so forth.

3.1.4 Mode-Independent SPI-4.2 Start-up Parameters

The SPI-4.2 interface defaults to dynamic deskew. If it is desired to operate the FM1010 with static deskew on any receive interface, then for that RX SPI-4.2 interface set the registers **RX_DESKEW1** (page 44), **RX_DESKEW2** (page 44), and **RX_DESKEW3** (page 44). The first register configures the manual deskew mode, and all three registers together determine the tap points for data lanes.

SPI-4.2 synchronization is based on correctly receiving DIP4 and DIP2 frames. **MIN_GOOD_PARITY** (MGP) is the number of correct DIP4 frames received before declaring synchronization, and **MAX_BAD_PARITY** (MBP) is the number of incorrect DIP4 frames received before declaring loss of synchronization. The register **RX_SYNC** (page 44) has fields for MGP and MBP on the RX side. The register **TX_SYNC0** (page 55) has MGP and MBP for the TX side as well

as the data training multiplicity. The **TX_SYNC1** (page 56) register contains the **MAX_DATA_T** parameter, which is the number of SPI-4.2 clock cycles between training sequences. Both synchronization registers have reasonable default values. These registers only need to be configured in the event that it is desired to operate the FM1010 outside of the synchronization defaults.

Note: It is not necessary to set any registers on unused SPI-4.2 interfaces.

3.1.5 Mode-Dependent Start-up Parameters

SPI-4.2 is a dual simplex interface. The FM1010 provides six full-duplex SPI-4.2 interfaces, which means there are 12 totally independent simplex interfaces, six of which are TX, and six of which are RX. The FM1010 allows each of its 12 simplex interfaces to be configured independently, including configuring the number of active ports per interface. The FM1010 can have a mixture of Clear and Multi-port modes communicating with each other.

3.1.5.1 Multi-port

There are many ways to configure multi-port mode. The following procedure is a guide to programming multi-port mode in typical configurations.

1. Leave the (M) field **RX_OP_MODE** (page 46) and **TX_OP_MODE** (page 58) unmodified.

   Rationale: The mode fields (M) of **RX_OP_MODE** and **TX_OP_MODE** default to Multi-port after chip reset, and do not need to be set on start-up.

2. Set the calendar length field and optionally set the multiplicity field of **RX_CAL_LM** (page 43) and **TX_CAL_LM** (page 55) and set the calendar in **RX_CALS[0..31]** (page 45) and **TX_CALS[0..31]** (page 57) to reflect the desired SPI-4.2 port configuration.

   Rationale: The default parameter for length is a 16-port interface. The appropriate calendar is system dependent; so it is likely that this parameter needs to be set. The default multiplicity is 4, and should only be changed if the system needs tuning. **RX_CALS[0..31]** and **TX_CALS[0..31]** default to linear 16 ports.

   Background: Flow control status is sent from the RX to the TX side of the SPI-4.2 link on the status channel. The flow control mechanism uses a calendar that must be configured the same on both the RX and TX devices. The RX interface steps through the fields in the calendar, and sends Starving, Hungry, or Satisfied flow control updates per port. The TX receives those credits and adjusts its transmission accordingly. The length of the calendar is defined in the length field of **RX_CAL_LM** and **TX_CAL_LM**. The length is the number of entries in the calendar. As the interfaces step through the calendar, the FIFO IDs in the calendar registers determine which port is currently under status update. In a linear calendar, the length is equal to the number of ports configured on the interface, and the calendar entries appear in ascending order. Sometimes it is desirable to update status on particular ports more frequently or in a different order. This is achieved by adjusting the length field and the calendar entries. The multiplicity field determines how many times the calendar is repeated between DIP2 parity and training pattern frames. The purpose of this is to optimize the efficiency of the status channel bandwidth versus the

status channel overhead. The calendar entries themselves are FIFO numbers from 0-255 in the register **RX_CALS[0..31]** and **TX_CALS[0..31]**.

3. If 8 or fewer FIFOs are to be used, Set the FIFO size value in **RX_FS** (page 46) and **TX_FS** (page 57).

   Rationale: The default is 16 1KB FIFOs. 1KB FIFOs should be enough to cover link partner latency. However, it is often desirable to use all available buffering.

   Background: The FM1010 has 16KB of configurable FIFOs per simplex interface.

4. Set the LOW and HIGH watermarks in **RX_WATERMARK** (page 46) to tune system performance.

   Rationale: These watermarks should be set during start-up because the default values are conservative and have not been optimized for performance. See Table 29 for the recommended values for the watermarks for each FIFO size.

   Background: SPI-4.2 employs a two-watermark credit system for flow control. Below the LOW watermark field in **RX_WATERMARK** (page 46) the status channel update is “Starving,” between the LOW and HIGH watermark, the status update is “Hungry,” and above the HIGH watermark, the status update is “Satisfied.”

5. Leave the registers **TX_MAX_BURST[0..15]** (page 59) and **TX_SERVICE_LIMIT** (page 58) unmodified unless it is required to tune per channel service limits.

   Rationale: The default values in the max burst and service limit registers are typical for a wide range of applications.

   Background: **TX_MAX_BURST[0..15]** specifies the values of **MAXBURST1** and **MAXBURST2** for each transmit port. A status report of “Starving” will reset the transmit port credits to **MAXBURST1** and a status report of “Hungry” will reset the transmit port credits to **MAXBURST2**, provided the credits had fallen below the level of **MAXBURST2**. The register **TX_SERVICE_LIMIT** offers a certain level of traffic shaping. It determines how much data is sent out of an eligible FIFO at a time (an eligible FIFO is one where the FIFO has data and the port has credits).

6. Set **RX_PORT_VALID[0..7]** (page 47, note the encoding) and **TX_FIFO_VALID** (page 59, note the encoding) to indicate the valid ports of the Rx interface and the valid FIFO's of the Tx interface.

   Rationale: All ports default to invalid, so at least one valid port must be set at start-up.

   Background: One error handling feature of the FM1010 is to drop packets with invalid Port IDs and report an error packet. There is a maximum of 16 active ports, which may have any port number, on any interface in multi-port mode.

7. Set **RX_PORT2FIFO[0..31]** (page 47, note encoding) and **TX_FIFO2PORT[0..15]** (page 59) to establish the mapping of ports to FIFOs.

   Rationale: The default values will not likely be appropriate for the target system.

   Background: The default settings map the first 16 SPI-4.2 ports to the 16 per-interface FIFOs, and all other ports to FIFO 0. This register does not determine port validity. On the receive side, the port valid and the port-to-FIFO map registers together establish valid ports and link them to the RX queues. On transmit side, **TX_FIFO_VALID** (page 59) and **TX_FIFO2PORT[0..15]** determine the validity of the queue and map it to an outgoing SPI-4.2 port. There is no filtering function on the transmit side, so **TX_FIFO_VALID** is a check for proper device configuration.

8. Set **RX_LINKCFG1[0..15]** (page 49), **RX_LINKCFG2[0..15]** (page 50), and **TX_LINKCFG[0..15]** (page 60) to configure the binding between ingress FIFOs and egress FIFOs.

   Rationale: It is necessary to define the routing configuration in the FM1010 to reflect the system topology.

Background: **RX_LINKCFG1[0..15]** and **RX_LINKCFG2[0..15]** establish the mapping of ingress queues to egress queues across the internal switch element in the FM1010. **TX_LINKCFG[0..15]** is the reverse mapping of this same information. The register has dependent information, but is set independently for internal hardware efficiency reasons. Consult the detailed sections for register entry equations.

3.1.5.2 Clear Mode

In Clear mode, the flow control of SPI-4.2 applies to the entire interface as if there were only one active port, however any port number may appear in the Port ID of the SPI-4.2 segment control word. This mode enables devices with more than 16 active ports to pass traffic through the FM1010 without error, but sacrifices flow control on a per-port basis, and instead has flow control on a per-interface basis. If a connected device were natively single ported, but an extra tag per SPI-4.2 burst was desired in the system, this mechanism provides that feature.

For the background descriptions on the register-setting instructions for the following procedure in clear mode, please refer to the previous section on configuring multi-port mode.

Note: Clear mode has the restriction that packets must be sent in their entirety before the Port ID can be changed.

1. Set the (M) field **RX_OP_MODE** (page 46) and **TX_OP_MODE** (page 58) to clear mode.

   Rationale: The mode fields (M) of **RX_OP_MODE** and **TX_OP_MODE** default to multi-port mode after chip reset, and thus needs to be changed.

2. Set the calendar length field to 1 and set the multiplicity field according to tuning considerations and to match the partner device in **RX_CAL_LM** (page 43) and **TX_CAL_LM** (page 55). Set the calendar in ***RX_CALS[0..31]*** (page 45) and **TX_CALS[0..31]** (page 57) to FIFO 0.

   Rationale: The default parameter for length is a 16-port interface. n The calendar refers to the FIFOs of which one is in use.

3. Set **RX_FS** (page 46) and **TX_FS** (page 57). to 1 FIFO, 16KB.

   Rationale: The default is 16 1KB FIFOs. 1KB FIFOs should be enough to cover link partner latency. However, it is often desirable to use all available buffering, and this setting allows for 16KB of buffering per interface.

4. Set the LOW and HIGH watermarks in **RX_WATERMARK** (page 46) to tune system performance.

   Rationale: These watermarks should be set during start-up because the default values are conservative and have not been optimized for performance. See Table 29 for the recommended values for the watermarks for each FIFO size.

5. Set the register **TX_MAX_BURST[0..15]** (page 59) to avoid overflow in the partner device buffer and leave **TX_SERVICE_LIMIT** (page 58) unmodified.

   Rationale: **TX_MAX_BURST[0..15]** is only meaningful for multi-port mode.

6. There is no need to overwrite the default values in **RX_PORT_VALID[0..7]** (page 47, note encoding) as they are ignored in clear-port mode. Set **TX_FIFO_VALID** (page 59, note encoding) to valid FIFO 0 only.

   Rationale: In clear mode, **RX_PORT_VALID[0..7]** is ignored. However, **TX_FIFO_VALID** determines the validity of the output queues. All queues

default to off since the SPI-4.2 link defaults to off. Queue 0 must be set to valid.

7. There is no need to set **RX_PORT2FIFO[0..31]** or **TX_FIFO2PORT[0..15]** as these two registers are ignored in clear-port mode..
8. Set **RX_LINKCFG1[0..15]** (page 49), **RX_LINKCFG2[0..15]** (page 50), and **TX_LINKCFG[0..15]** (page 60) to configure the binding between ingress FIFOs and egress FIFOs.

   Rationale: It is necessary to define the routing configuration in the FM1010 to reflect the system topology.

3.2 Chip Operation

3.2.1 Statistics

Basic statistics are available on each transmit and receive SPI-4.2 interface. These statistics are read-only registers.

- **RX_PKTCNT** (page 51) is a 32-bit count of the received packets. It wraps to zero after reaching its maximum value.
- **RX_PKTERRCNT** (page 51) is a 16-bit count of the packets received with errors.
- **RX_DATACNT** (page 52) is the top 32-bits of a 34-bit count of all bytes of data received by the SPI-4.2 interface. It wraps to zero after reaching its maximum value.
- **TX_PKTCNT** (page 61) is a 32-bit count of the transmitted packets. It wraps to zero after reaching its maximum value.
- **TX_PKTERRCNT** (page 61) is a 16-bit count of the packets transmitted with errors.
- **TX_DATACNT** (page 61) is the top 32-bits of a 34-bit count of all bytes of data transmitted by the SPI-4.2 interface. It wraps to zero after reaching its maximum value.

3.2.2 Link/Port Reset Procedure

This procedure can be used to reset a link (queue pairing) within the FM1010. This procedure can only be used after a valid link has been established within the device, and not directly after chip reset. The feature allows the administrator to remap a port during run time if so desired.

To reset one or more ports:

1. Set the corresponding FIFO bit in **RX_STATUS_OVERRIDE** (page 48). This will cause the SPI-4.2 interface to send out satisfied status for the port to be reset.
2. Clear the corresponding bit in **RX_PORT_VALID** (page 47). This will cause any future data that arrives for that port to be discarded at the input.
3. After clearing the **RX_PORT_VALID** bit, wait until the port FIFO's are no longer draining (either empty or, if stalled, no longer draining) before proceeding to step 4. Not doing so will result in unpredictable behavior.
4. Set the corresponding bit in **RX_LINK_RESET** (page 50). This will initiate the reset of the queue pair link (several bits may be set at once to reset several links). This will flush the data from the input and output queue of that link.

5. Poll for the bit in **RX_LINK_RESET** to clear. This indicates that the reset has been successfully completed.
6. Optionally make any link configuration changes.
7. Set the correct bit in **RX_PORT_VALID** to enable the port.
8. Clear the bit in **RX_STATUS_OVERRIDE** to resume sending valid status.

During step 6 of the link reset procedure, it is valid to change the following configuration registers:

- **RX_LINKCFG1**
- **RX_LINKCFG2**
- **RX_PORT2FIFO**
- **TX_LINKCFG**
- **TX_FIFO_VALID**
- **TX_FIFO2PORT**

Note: It is only valid to change the information related to either the link that is being reset and torn down or any replacement link being established.

3.2.3 Interrupt Processing

The FM1010 has 13 possible interrupt sources:

- The CPU bus interface
- Six SPI-4.2 transmit interfaces
- Six SPI-4.2 receive interfaces

Upon detection of an interrupt from the FM1010, the host processor software shall proceed as follow:

1. Read the **INTR_DETECT** register to determine the sources of the interruption. The register contains one bit per possible source.
2. If an interrupt from the CPU interface is detected, read the **CPU_INTERFACE_IP** to determine the exact cause of the interruption. This register is self clearing (this register and the corresponding bit for this source in the **INTR_DETECT** are automatically cleared after being read). The interrupts from the CPU interface may be masked using the **CPU_INTERFACE_IM** register.
3. If an interrupt from the SPI-4.2 transmit or receive interface is detected, read the corresponding **RX_IP** or **TX_IP** register to determine the exact cause of the interruption. These registers are self clearing (these registers and the corresponding bits in the **INTR_DETECT** are automatically cleared after being read). The interrupts from the SPI-4.2 interfaces may be masked using the **RX_IM** and **TX_IM** registers.

3.3 SPI-4.2 Interface

For a general overview of the SPI-4.2 interface specification, please refer to the SPI-4.2 implementation agreement:

- Optical Internetworking Forum System Packet Interface Level 4 (SPI-4) Phase 2: OC-192 System Interface for Link Layer Devices, January 2001

This document is available at http://www.oiforum.com.

3.4 CPU Interface

The CPU Interface in the FM1010 is a 17-bit address, 16-bit data bus used to access the internal FM1010 registers. This bus is the primary access point for configuration and control. The interface is a slave device that uses a handshaking protocol to allow a variable amount of delay to respond to requests. Requests of 32-bit length are accomplished using two sequential 16-bit accesses.

3.4.1 General Description

- Slave-terminated protocol that allows a variable amount of delay to respond to requests
- 16-bit data interface only, supporting single, Big Endian, read/write transactions
- Interrupt generation
- No support for parity and other bus error detection and correction mechanisms
- Reconfiguration of the FM1010 - 10 us max
- Maximum frequency range of 50MHz

3.4.2 Register Read / Write Operations

Reads and writes always act on a word (of 32-bit length) internally. The FM1010 assumed that the processor creates an atomic 32-bit read or write operation that is segmented by the bus protocol into two 16-bit read or write accesses.

Note: It is required that all writes and reads access the upper 16-bit aligned half-word before accessing the lower half-word (big-endian half-word ordering). The High Half Word shall be transferred when ADDR[1] is 0 while the Low Half Word shall be transferred when ADDR[1] is 1.

Figure 7 illustrates the byte ordering assumed by the FM1010.

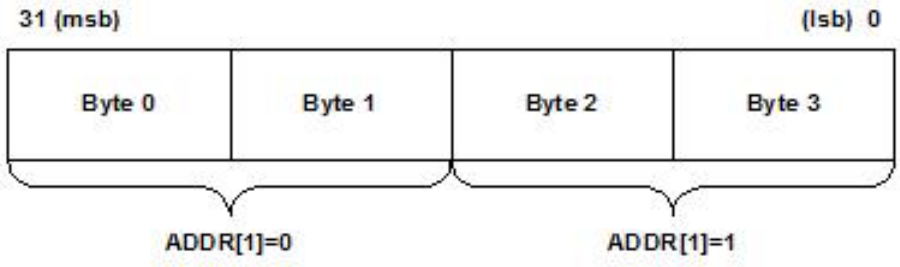


Figure 7. Register Byte Format

3.4.2.1 Read Operation

A read operation must always request the entire 32 bits of an internal register. The host bus controller shall break the 32-bit access into two successive 16-bit accesses:

1. First Access: Read Word Address at address X (Addr[1] not set). The FM1010 performs an internal 32-bit read and returns the most significant 16 bits on the bus while the least significant 16 bits are temporarily cached in the interface logic.
2. Second Access: Read Word Address at address X+2 (Addr[1] is set), the locally stored 16 bits are returned.

Note: If the second read is not to the same word address or Addr[1] is not set, then the first step is repeated for this new word access. In a register read request, it is required that the high Half-Word is read first. A failure to follow this protocol may result in undefined behavior.

3.4.2.2 Write Operation

A write operation must write an entire 32-bit register for the device to work properly. The host bus controller must break the 32-bit access into two successive 16-bit accesses:

1. First Access: Write Word Address X (Addr[1] is not set). The 16 bits of data are stored temporarily in a 16-bit register in the interface logic.
2. Second Access: Write Word Address X+2 (Addr[1] is set). The temporarily stored 16 bits and the new 16 bits of data are combined and written to the appropriate register. Only the second write triggers the register state change.

Note: The FM1010 internal bus interface logic will generate CPU interrupt faults if this protocol is not respected. An overwrite error is generated if Addr[1] is not set and the temporary write register has been initialized with data by a previous transaction. The FM1010 will overwrite the temporary register and set the Overwrite Status bit to indicate the event. A maskable interrupt is automatically generated If this bit is not masked. An underwrite error is generated if Addr[1] is set on a new address (different from previously-accepted address). The FM1010 will use the current content of the 16-bit temporary register, combined with the new data, and proceed with the write, and then set the Underwrite Status bit to indicate the event. A maskable interrupt is automatically generated if this bit is not masked.

3.4.3 CPU Interface Operation

The CPU Bus Interface timing diagram is shown in Figure 8. All input signals are sampled at the rising edge of CLK and all output signals are driven (or tri-stated) at the rising edge of CLK as well.

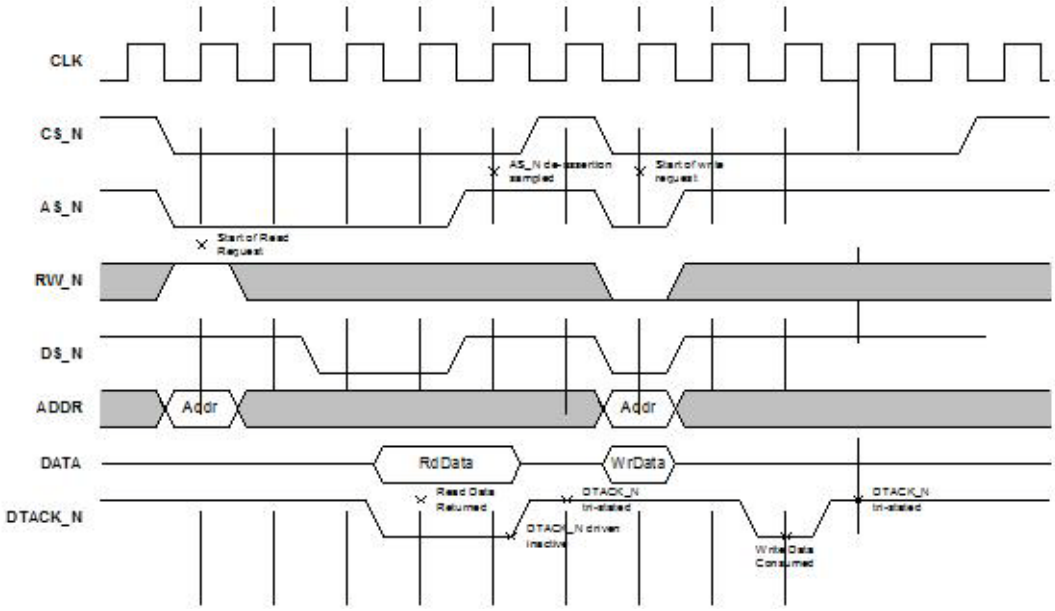


Figure 8. CPU Bus Interface Timing Diagram

Note: Two consecutive 16b read (write) operations are required for a complete 32b read (write) transaction. The diagram of Figure 8 does not represent a complete transaction of either type and is only for illustrative purposes. See Paragraph 3.4.2 for a complete description.

The FM1010 Bus Interface operates under the following rules:

1. The assertion of **AS_N, ADDR, RW_N** can occur at an arbitrary number of cycles after **CS_N** is asserted. **ADDR and RW_N** are only sampled after the first clock edge in which **AS_N** is asserted.
2. The assertion of **AS_N** can be as small as 1 cycle. This will allow **DTACK_N** and **DATA** to be asserted as soon as possible (for the minimum of 1 clock period). This is shown in the Write request example. Otherwise a handshake occurs between **AS_N** and **DTACK_N** which forces **DTACK_N** and **DATA** to be driven for a minimum of 2 clock cycles. This is shown in the Read request example.
3. C**S_N** can be de-asserted 1 clock earlier on the read request but the **DTACK_N** will not drive itself inactive and will instead be tri-stated on the clock edge after **CS_N** is de-asserted.
4. **CS_N** must be de-asserted for at least 1 cycle between transactions.
5. **DS_N** is an optional control input (the **IGNORE_DS_N** strapping option can be set to force the bus interface to ignore this signal). This signal can be used to specify when the read data may be placed on the data bus and when write data is available independently of **AS_N** assertion.
6. **DTACK_N** and **RW_N** polarities can be inverted using the **DTACK_INV** and **RW_INV** strapping options.

The **SYNC_MODE** strapping option is not currently supported and should always be asserted high.

3.4.4 Bus Error Condition Handling

The FM1010 Bus Interface behavior under abnormal circumstances is detailed in the following paragraphs. It is recommend to design the bus interface to the FM1010 to avoid the abnormal circumstances described here.

1. The Write Over/Underflow error conditions create a maskable interrupt.
   - **WR OVERWRITE STATUS**: Indicates that a Bus Interface write was started but was then superseded before it was completed internally (by another High Half-Word write to a different address).
   - **WR UNDERWRITE STATUS**: Indicates that the Bus Interface has received the Low Half-Word for a write without first receiving the High Half-Word (last High Half-Word write was to a different address or a Serial Tree write occurred immediately before). The write will still occur but the High Half-Word will receive the contents of the **WRDATA** register.
2. A burst request (multiple **AS** or **DS** signals within one **CS** envelope) is attempted on the interface. This is unsupported and therefore should NOT be attempted. The Bus Interface only returns one piece of data and no more. The bus deadlocks until the processor times out and terminates the transaction. At this point the Bus Interface is ready to respond to the next access with no additional clean-up required (in fact the Bus Interface was unaware and completely unaffected by the deadlock).
3. **AS_N** and **DS_N** get out of sync. This can occur if the processor fails to properly follow an **AS_N** phase by a **DS_N** phase (for instance starting **AS_N** but then terminating the request and restarting another request). In this case the Bus Interface will always use the n-1 **AS_N** address to service the current request. For processors that set the **IGNORE_DS_N** pin strap this can never be a problem. For all other processors they must guarantee that they do NOT terminate a request that has been started. If they do, then their Address and Data Tokens will become misaligned.
4. **AS_N** re-asserted before **DTACK_N** phase completed. This does NOT create an internal problem within the Bus Interface but may cause problems to the driving processor since the **DTACK_N** may become misaligned with the ongoing request.
5. Once the **AS_N** and **DS_N** transitions have been seen then the Bus Interface responds with **DTACK_N** and **DATA** regardless of what occurs on all other input pins (for example **CS_N**). There is no method to early-terminate an ongoing transaction once it has been started. But early termination of **CS_N** will tri-state **DTACK_N** and **DATA** on the next clock edge, which will appear to the outside system as a terminated transaction. Care must be taken not to leave the Bus Interface waiting for data phase (Phase 2) to complete -- this would create an error 3 condition (as described above).
6. The Bus Interface does NOT have a dedicated watchdog timer. The processor must respect the normal bus protocol of asserting **AS_N** and **DS_N** and then waiting for **DTACK_N** assertion before proceeding to the next transaction with **CS_N** enveloping the entire transaction.

3.5 JTAG Interface

For a general overview of JTAG, please review:

- IEEE 1149.1 Standard Test Access Port and Boundary-Scan Architecture, 2001, available from http://www.ieee.org.

3.6 Clocks

The FM1010 has SPI-4.2 data path input and output clocks (per interface), CPU interface clock, and JTAG input clock. The unique requirements of each clock source are detailed in this section.

3.6.1 SPI-4.2 Receive and Transmit Data Path Clocks

The SPI-4.2 data path clocks are compliant with the OIF specification. The FM1010 has the following requirements on the transmit data path:

- 2.5 V LVDS drive
- Maximum duty cycle distortion 45/55
- Maximum peak-to-peak jitter (low and high frequency) of 125 pS
- Stable (frequency and level) when reset is removed or when sourced
- The FM1010 meets the following specifications on the receive data path:
- 2.5 V LVDS drive
- Maximum duty cycle distortion 45/55
- Maximum peak-to-peak jitter (low and high frequency) of 0.1UI
- Stable when sourced
- Frequency range between 311 MHz and 450 MHz

The FM1010 status path meets the following requirements:

- 3.3V LVTTL
- 1/4 frequency of the SPI-4.2 data path frequency
- Maximum duty cycle distortion 45/55

3.6.2 CPU Interface Clock

The clock source for the CPU interface on the FM1010 must meet the following requirements:

- 3.3 V CMOS drive
- Maximum frequency of 50 MHz

3.6.3 JTAG Interface Clock

The FM1010 supports JTAG. The clock source must meet the 3.3 V LVTTL specification, with a maximum clock input frequency of 40 MHz and a maximum duty cycle distortion of 40/60.

4.0 Electrical Specifications

Table 3 through Table 10 provide recommended operating conditions for the FM1010.

4.1 Absolute Maximum Ratings

Table 3. Absolute Maximum Ratings

Parameter	Symbol	Min	Max	Units
LVDS Power Supply	V_{DD25}	2.375	2.625	Volts
LVDS Output Reference Voltage	V_{REF}	-0.3	2.8	Volts
LVTTL Power Supply	V_{DD3}	-0.3	3.9	Volts
PLL Analog power supply	V_{DDA33_PLL}	-0.3	3.75	Volts
Core Voltage	V_{DD}	-0.3	1.5	Volts
Operating Temp (Case)		-	+130	°C
Storage Temp		-65	+150	°C

4.2 Recommended Operating Conditions

Table 4. Recommended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Units
LVDS Power Supply	V_{DD25}	2.375	2.500	2.625	Volts
LVDS Output Reference Voltage	V_{REF}	1.140	1.200	1.55	Volts
LVTTL Power Supply	V_{DD33}	3.000	3.300	3.600	Volts
PLL Analog power supply	V_{DDA33_PLL}	3.000	3.300	3.600	Volts
Core Voltage	V_{DD}	1.140	1.200	1.55	Volts
		.95[3]	1.0[3]	1.05[3]	
		1.05[4]	1.1[4]	1.15[4]	
Operating Temp (Case)					
Commercial[1]		0	+40	+70	°C
Extended[2]		-20	+50	+85	°C

(1) Commercial grade version of the device (PN: FM1010-F1232-400C).

(2) Extended grade version of the device (PN: FM1010-F1232-400E).

(3) Core voltage of VDD = 1.0V may be used in applications up to 400 MHz and in 1232-ball package to reduce current.

(4) Core voltage of VDD = 1.0V may be used in applications up to 450 MHz and in 1232-ball package to reduce current.

Note: For information on calculating the power budget for a particular application, refer to page 68.

Table 5. DC Characteristics of LVTTL 4mA Outputs

Parameter	Symbol	Test Conditions	Min	Type	Max	Units
HIGH Force Tri-State output leakage	I_{OZH}	V_{DD}= Max V_o=V_{DD}	-1	-	+1	µA
LOW Force Tri-State output leakage	I_{OZL}	V_{DD}=Max V_o=GND	-1	-	+1	µA
Output HIGH Current	I_{ODH}	V_{DD}=1.2 V, V_{DD33}=3.3 V, V_O = 1.5 V	-	-17	-	mA
Output LOW Current	I_{ODL}	V_{DD}=1.2 V, V_{DDO}=3.3 V, V_O = 1.5 V	-	-20	-	mA
Output HIGH Voltage	V_{OH}	V_{DD}= Min V_{DD33}=Min I_{OH}= -0.4 mA	V_{DD33} - 0.2	-	-	V
Output HIGH Voltage	V_{OH}	V_{DD}= Min V_{DD33}=Min I_{OH}= -4.0 mA	V_{DD33} - 0.5	-	-	V
Output LOW Voltage	V_{OL}	V_{DD}= Min V_{DD33}=Min I_{OL}= -0.4 mA	-	-	0.2	V
Output LOW Voltage	V_{OL}	V_{DD}= Min V_{DD33}=Min I_{OL}= -4.0 mA	-	0.2	0.4	V
Short Circuit Current	I_{OS}	V_{DD}=MAX V_o=GND			-32	mA
Power Supply Quiescent Current	I_{AA}	V_{DD}= Max V_{DD33}=Max			74	µA
Power Supply Quiescent Current	I_{AA}	Tri-stated			-1	µA

Table 6. DC Characteristics of LVTTL 8mA Outputs

Parameter	Symbol	Test Conditions	Min	Typical	Max	Units
HIGH Force Tri-State output leakage	I_{OZH}	V_{DD}= Max V_o=V_{DD}	-1	-	+1	µA
LOW Force Tri-State output leakage	I_{OZL}	V_{DD}=Max V_o=GND	-1	-	+1	µA
Output HIGH Current	I_{ODH}	V_{DD}=1.2 V, V_{DD33}=3.3 V, V_O = 1.5 V	-	-35	-	mA
Output LOW Current	I_{ODL}	V_{DD}=1.2 V, V_{DD33}=3.3 V, V_O = 1.5 V	-	-40	-	mA
Output HIGH Voltage	V_{OH}	V_{DD}= Min V_{DD33}=Min I_{OH}= -0.8 mA	V_{DD33} - 0.2	-	-	V
Output HIGH Voltage	V_{OH}	V_{DD}= Min V_{DD33}=Min I_{OH}= -8.0 mA	V_{DD33} - 0.5	-	-	V
Output LOW Voltage	V_{OL}	V_{DD}= Min V_{DD33}=Min I_{OL}= -0.8 mA	-	-	0.2	V
Output LOW Voltage	V_{OL}	V_{DD}= Min V_{DD33}=Min I_{OL}= -8.0 mA	-	0.2	0.4	V

Table 6. DC Characteristics of LVTTL 8mA Outputs (Continued)

Parameter	Symbol	Test Conditions	Min	Typical	Max	Units
Short Circuit Current	I_{OS}	V_{DD}=MAX V $_o$=GND			-64	mA
Power Supply Quiescent Current	I_{AA}	V_{DD}= Max V_{DD33}=Max			74	µA
Power Supply Quiescent Current	I_{AA}	Tri-stated			-1	µA

Table 7. DC Characteristics of LVTTL Inputs

Parameter	Symbol	Test Conditions	Min	Typical	Max	Units
Input HIGH Level (Input and I/O pins)	V_{IH}	Guaranteed Logic HIGH Level	2	-	V_{DD33} +0.5	V
Input LOW Level (Input and I/O pins)	V_{IL}	Guaranteed Logic LOW Level	-0.3	-	0.8	V
Input Hysterisis	V_H	it0		5		mV
Input Hysterisis	V_H	it2		200		mV
Input HIGH Current (Input pins)	I_{IH}	V_{DD}=Max, V_I=V_{IH}(Max)			+-1	µA
Input HIGH Current (I/O pins)	I_{IH}	V_{DD}=Max, V_I=V_{DD33}			+-1	µA
Input LOW Current (Input pins)	I_{IL}	V_{DD}=Max, V_I=GND			+-1	µA
Input LOW Current (I/O pins)	I_{IL}	V_{DD}=Max, V_I=GND			+-1	µA
Clamp Diode Voltage	V_{IK}	V_{DD}=Min, I_{IN}=-18mA		-0.7	-1.2	V
Quiescent Power Supply Current	I_{DD33}	V_{DD}=Max, V_{DD33}=Max, V_{IN}=GND		0.1	10	µA
Quiescent Power Supply Current	I_{DD33}	V_{DD}=Max, V_{DD33}=Max, V_{IN}=V_{DD}		0.1	10	µA

Table 8. DC Characteristics of LVDS Outputs

Parameter	Symbol	Test Conditions	Min	Typical	Max	Units
Output high level, V_{IA} or V_{OB}	V_{OH}	R_{load}=100 ohms	-	-	1475	MV
Output low level, V_{IA} or V_{OB}	V_{OL}	R_{load}=100 ohms	925	-	-	MV
Output Voltage Differential	$\|V_{OD}\|$	R_{load}=100 ohms	250	330	450	MV
VREF supply voltage	V_{VREF}	R_{load}=100 ohms	1140	1200	1260	MV
Output common mode offset	V_{OS}	R_{load}=100 ohms	1125	1200	1275	MV
Between “0” and “1”	|Delta V_{OD}|	R_{load}=100 ohms	-	-	25	mV
Between “0” and “1”	|Delta V_{OS}|	R_{load}=100 ohms	-	-	25	mV
Short circuit output current		Drivers shorted to ground	-	-	40	mA

Table 8. DC Characteristics of LVDS Outputs (Continued)

Parameter	Symbol	Test Conditions	Min	Typical	Max	Units
Short circuit output current		Drivers shorted together	-	-	12	mA
Power off output leakage	$\|IX_A\|$, $\|IX_B\|$	Power off			10	mA
Reference Leakage Current	$\|I_{VREF}\|$		-1.0	0	1.0	mA

Table 9. DC Characteristics of LVDS Inputs

Parameter	Symbol	Test Conditions	Min	Typical	Max	Units
Valid Input Voltage	V_{IA} or V_{IB}	$\|V_{GPD}\|$ < 125 mV (1)	0	-	2400	mV
Differential Input Voltage Range	$\|V_{ID}\|$	$\|V_{GPD}\|$ < 125 mV (1)	100	-	1200	mV
Hysteresis	V_{HYST}	V_{idthh}-V_{idthl}	5	-	-	mV
Internal Differential Termination	R_{IN}	-	80	98	120	Ohms
(1) VGPD: Ground Potential Difference						

4.3 AC Timing Specifications

4.3.1 SPI-4 Interface

Please refer to the OIF SPI4-02.1 Implementation Agreement for SPI-4.2 interface timing information. Section 6.4.1 of the agreement describes the AC timing parameters for the data path of the SPI-4.2 interface. Section 6.4.2 describes the AC timing parameters for the LVTTL Status Channel.

4.3.2 CPU Interface, General Timing Requirements

General timing requirements for the CPU interface are shown in Figure 9.

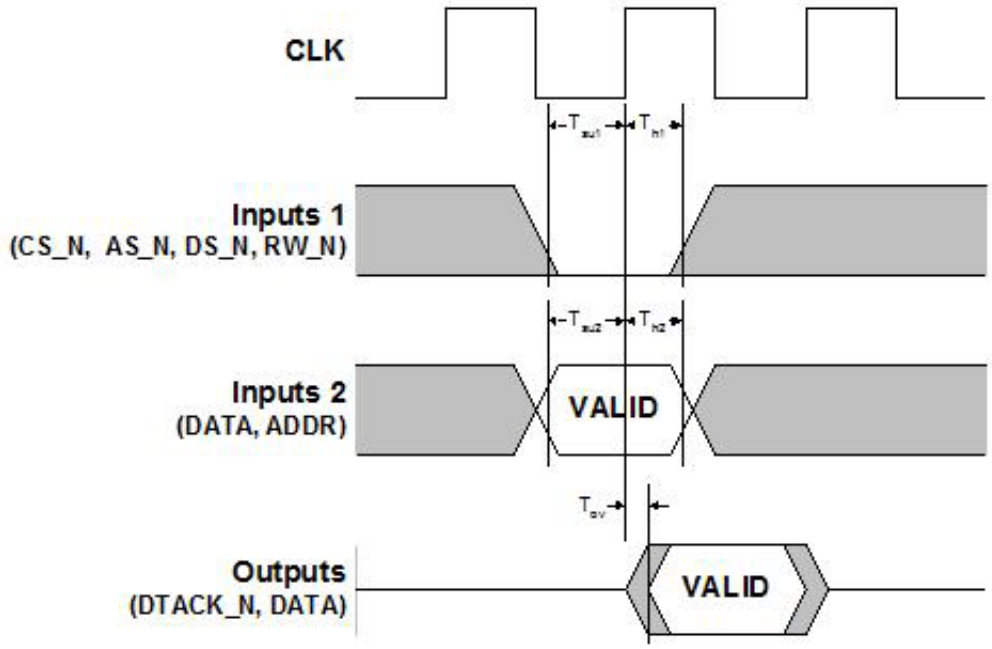


Figure 9. CPU Signal Timing

Table 10. CPU Interface Timing Constraints

Parameter	Symbol	Min	Typ	Max	Units	Test Conditions
Setup time for CS_N, AS_N, DS_N and RW_N, to rising edge of clock	T_{su1}	4.0	-	-	ns	-
Hold time for CS_N, AS_N, DS_N and RW_N, to rising edge of clock	T_{h1}	0.5	-	-	ns	-
Setup time for ADDR and DATA(in) to rising edge of clock	T_{su2}	4.5	-	-	ns	-
Hold time for ADDR and DATA(in) to rising edge of clock	T_{h2}	0.5	-	-	ns	-
Output valid for DSTACK_N and DATA(out) to rising edge of clock	T_{ov}	0	-	5.7	ns	

Notes:
- DTACK_INV, RW_N_INV, IGNORE_DS_N, SYNC_MODE are static signals. They must be stable before RESET_N is de-asserted.
- BUSIF_RESET and INTR are asynchronous signals.
- Typical latency to access an internal 32-bit register is in the range of 100-150ns

4.3.3 JTAG Interface

The JTAG interface follows standard timing as defined in the IEEE 1149.1 Standard Test Access Port and Boundary-Scan Architecture, 2001.

Note: When not using the JTAG interface, either drive the TCK pin with an external clock, or drive the TRST_N pin low. Conversely, when using the JTAG interface assert TRST_N along with chip reset to ensure proper reset of the JTAG interface prior to use.

5.0 Register Definitions

Table 11 through Table 14 provide information on the location and functionality of the control and status registers contained in the FM1010.

5.1 Memory Map

This section provides the FM1010 memory maps. A number of global control and status registers are used to configure or report on all interfaces, and some registers are replicated on a per-interface basis.

Table 11. FM1010 Register Memory Map

Register	Base Address
CPU Interfaces registers	0x0000
SPI4 Interface #0	0x8000
SPI4 Interface #1	0x9000
SPI4 Interface #2	0xA000
SPI4 Interface #3	0xB000
SPI4 Interface #4	0xC000
SPI4 Interface #5	0xD000

Table 12. FM1010 CPU Interface Memory Map

Register	Size	Type	Ref Page	Offset
CPU_INTERFACE_IM	32	RW		0x0028
CPU_INTERFACE_IP	32	CR		0x002c
INTR_DETECT	32	RO		0x0044
SPI4_WATCHDOG	32	RW		0x0048
VPD	32	RO		0x0040

Table 13. Register Offsets in the SPI-4.2 Interface (RX)

Register	Size	Type	Ref Page	Offset
RX_RESET	32	RW	43	0x0c00
RX_CAL_LM	32	RW*	43	0x0c0c
RX_SYNC	32	RW*	44	0x0c10
RX_DESKEW1	32	RW*	44	0x0c14
RX_DESKEW2	32	RW*	44	0x0c18
RX_DESKEW3	32	RW*	45	0x0c1c
RX_CALS[i] {i: 0..31}	32x32	RW*	45	0x0c80 + 0x0004*i
RX_FS	32	RW*	46	0x0c08
RX_OP_MODE	32	RW*	46	0x0f00
RX_WATERMARK	32	RW*	46	0x0d00

Table 13. Register Offsets in the SPI-4.2 Interface (RX) (Continued)

Register	Size	Type	Ref Page	Offset
RX_PORT2FIFO[i] {i: 0..31}	32x32	RW	47	0x0f80 + 0x0004*i
RX_PORT_VALID[i] {i: 0..7}	32x32	RW	47	0x0f20 + 0x0004*i
RX_STATUS_OVERRIDE	32	RW	48	0x0d04
RX_OS	32	RO	49	0x0e18
RX_LINKCFG1[i] {i: 0..15}	32x32	RW	49	0x0900 + 0x0004*i
RX_LINKCFG2[i] {i: 0..15}	32x32	RW	50	0x0800 + 0x0004*i
RX_LINK_RESET	32	RW	50	0x0e20
RX_PKTCNT	32	RO	51	0x0f18
RX_PKTERRCNT	32	RO	51	0x0f1c
RX_DATACNT	32	RO	52	0x0f10
RX_IP	32	CR	52	0x0e04
RX_IM	32	RW	53	0x0e00
RX_DEBUG_STATUS	32	RO	53	0x0e08
RW- Read/Write RO – Read Only CR – Clear on Read (*) Should only be written while RX_RESET=1 (**) For Intel® debug purposes only. DO NOT WRITE				

Table 14. Register Offsets in the SPI-4.2 Interface (TX)

Register	Size	Type	Ref Page	Offset
TX_RESET	32	RW	55	0x0040
TX_CAL_LM	32	RW*	55	0x004c
TX_SYNC0	32	RW*	55	0x0050
TX_SYNC1	32	RW*	56	0x0054
TX_CORE_WATERMARK	32	RW*	56	0x0058
TX_CALS[i] {i: 0..31}	32x32	RW*	57	0x0780 + 0x0004*i
TX_FS	32	RW*	57	0x0048
TX_OP_MODE	32	RW*	58	0x0600
TX_SERVICE_LIMIT	32	RW	58	0x0400
TX_MAX_BURST[i] {i: 0..15}	16x32	RW	59	0x0700+0x0004*i
TX_FIFO2PORT[i] {i: 0..15}	16x32	RW	59	0x0640 + 0x0004*i
TX_FIFO_VALID	32	RW	59	0x00c0
TX_OS	32	RO	60	0x0070
TX_LINKCFG[i] {i: 0..15}	32x32	RW	60	0x0080+0x0004*i
TX_PKTCNT	32	RO	61	0x0008
TX_PKTERRCNT	32	RO	61	0x000c
TX_DATACNT	32	RO	61	0x0000
TX_IP	32	CR	62	0x007c
TX_IM	32	RW	62	0x0078

Table 14. Register Offsets in the SPI-4.2 Interface (TX) (Continued)

Register	Size	Type	Ref Page	Offset
TX_PLL_CTRL	32	RW	62	0x0060
TX_PLL_STAT	32	RO	63	0x0064
RW- Read/Write RO – Read Only CR – Clear on Read (*) Should only be written while RX_RESET=1 (**) For Intel® debug purposes only; not protected against writing. DO NOT WRITE.				

5.2 FM1010 Global and CPU Interface Register Definitions

Table 15 through Table 19 provide information pertaining register definitions.

Table 15. SPI4_WATCHDOG Register

Name	Bit	Description	Type	Default
Interface	5:0	Activates the watchdog on the SPI4 ports, as follows: 0 = SPI4 Interface #0 1 = SPI4 Interface #1 2 = SPI4 Interface #2 3 = SPI4 Interface #3 4 = SPI4 Interface #4 5 = SPI4 Interface #5	RW	0
RSVD	31:6	Reserved. Set to 0.	RV	0

Table 16. CPU_INTERFACE_IM Register

Name	Bit	Description	Type	Default
OVERWRITE	0	Mask for OVERWRITE interrupt status. This bit is “AND-ed” with the corresponding bit of the CPI_INTERFACE_IP register to create the CPU interrupt signal. 0 = Mask 1 = Do NOT MASK	RW	0
UNDERWRITE	1	Mask for UNDERWRITE interrupt status. This bit is “AND-ed” with the corresponding bit of the CPI_INTERFACE_IP register to create the CPU interrupt signal. 0 = Mask 1 = Do NOT MASK	RW	0
RSVD	31:2	Reserved. Set to 0.	RV	0

Table 17. CPU_INTERFACE_IP Register

Name	Bit	Description	Type	Default
OVERWRITE	0	Indicates that a CPU bus IF write was started but was then superseded before it was completed internally. This register clears itself on read. Writing has no effect.	RC	0
UNDERWRITE	1	Indicates that the CPU bus IF has received the lower 16 bits for a write (Addr[1]=1) without first receiving the upper 16 bits. This register clears itself on read. Writing has no effect.	RC	0
RSVD	31:2	Reserved. Set to 0.	RV	0

Table 18. INTR_DETECT Register

Name	Bit	Description	Type	Default
SPI4_0_RXINTR	0	Active RX interrupt outstanding from SPI4, port 0.	RO	0
SPI4_0_TXINTR	1	Active TX interrupt outstanding from SPI-4, port 0.	RO	0
SPI4_1_RXINTR	2	Active RX interrupt outstanding from SPI4, port 1.	RO	0
SPI4_1_TXINTR	3	Active TX interrupt outstanding from SPI4, port 1.	RO	0
SPI4_2_RXINTR	4	Active RX interrupt outstanding from SPI4, port 2.	RO	0
SPI4_2_TXINTR	5	Active TX interrupt outstanding from SPI4, port 2.	RO	0
SPI4_3_RXINTR	6	Active RX interrupt outstanding from SPI4, port 3.	RO	0
SPI4_3_TXINTR	7	Active TX interrupt outstanding from SPI4, port 3.	RO	0
SPI4_4_RXINTR	8	Active RX interrupt outstanding from SPI4, port 4	RO	0
SPI4_4_TXINTR	9	Active TX interrupt outstanding from SPI4, port 4.	RO	0
SPI4_5_RXINTR	10	Active RX interrupt outstanding from SPI4, port 5.	RO	0
SPI4_5_TXINTR	11	Active TX interrupt outstanding from SPI4, port 5.	RO	0
RSVD	12:15	Reserved	RO	0
CPU_IF_INTR	16	An illegal sequence has occurred on the CPU Bus.	RO	0
RSVD	31:17	Reserved	RO	0
Note: These 17 Interrupt Status bits are “O-ed” to assert INTR_N.				

Table 19. VPD - Vital Product Data

Name	Bit	Description	Type	Default
CONST	0	1 bit constant alignment field	RO	1
JTAG ID	11:1	JEDEC Manufacturer's ID for Intel® (4 bytes of continuation code and ID of 7'h15)	RO	0x215
PART NUM	27:12	Part Number – Intel® specific	RO	0x00A8
VERSION	31:28	Version	RO	0x01

5.3 SPI-4.2 Interface Register Descriptions

5.3.1 RX_RESET

The **RX_RESET** register, provided in Table 20, controls the SPI-4.2 RX interface reset status. After hardware reset, this register has a value of 1. When the value of this register is changed from 1 to 0 for the first time after hardware reset, the device reads and locks the value of RX_FS. Further writes to the register will have no effect.

An RX SPI-4.2 interface performs no other operation before this register is cleared for the first time. All start-up register configurations should be set before clearing this register.

Table 20. RX_RESET Register

Name	Bit	Description	Type	Default
RSVD	31:1	Reserved. Set to 0.	RO	0
RESET	0	Place the SPI4 receiver in reset mode.	RW	1

5.3.2 RX_CAL_LM

This register specifies the length and multiplicity of the calendar. The calendar is the sequence of per "port" status updates on the status channel. Table 21 lists the fields of the registers:

Table 21. Calendar Length and Multiplicity Fields (RX)

Field	Bits	Description	Type	Default
RSVD	31:17	Reserved. Set to 0.	RO	0
E	16	Clock edge used by the status channel. 1 is rising edge; 0 is falling edge.	RW	0x1
MULT	15:8	Calendar multiplicity, i.e., number of times calendar sequence is repeated between framing patterns.	RW	0x04
LEN	7:0	Calendar length, i.e., number of calendar entries. A value of zero is illegal and results in undefined behavior. Zero does not map to 256, and the FM1010 supports up to 255 ports.	RW	0x10

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., when the value of **RX_RESET** is 1). Writes at other times may cause errors in the SPI-4.2

interface. The calendar of the transmit and receive interfaces must be identical on both sides of a SPI-4.2 link.

5.3.3 RX_SYNC

RX_SYNC specifies the conditions for the SPI-4.2 interface block to declare acquisition and loss of data path synchronization, as provided in Table 22.

Table 22. Declaring Loss of Synchronization (RX)

Field	Bits	Description	Type	Default
RSVD	31:8	Reserved. Set to 0.	RO	0
MBP	7:4	MAX_BAD_PARITY, maximum number of consecutive bad DIP4 frames received on the data path before declaring loss of synchronization. The valid range is 0 to 15.	RW	0x2
MGP	3:0	MIN_GOOD_PARITY, minimum number of consecutive correct DIP4 frames received on data path before declaring synchronization. The valid range is 1 to 15.	RW	0x5

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., the value of **RX_RESET** is 1). Writes at other times will cause errors.

5.3.4 RX_DESKEW1

This register is in conjunction with the RX_DESKEW2 and RX_DESKEW3 registers, as listed in Table 23. It specifies the data path deskew mode for the SPI-4.2 interface block.

Table 23. Data Path Deskew Mode

Field	Bits	Description	Type	Default
ART	31:28	Average response time to jitter.	RW	0x6
ERT	27	Enable real-time tuning. 0 = Disabled 1 = Enabled.	RW	1
ESD	26	Enable static deskew. 0 = Disabled 1 = Enabled.	RW	0
EMD	25	Enable manual deskew; 0 –disabled, 1 – enabled.	RW	0
TAP	24:0	Tap points for data path lanes 0..4 in manual deskew mode.	RW	0x0

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., the value of **RX_RESET** is 1). Writes at other times will cause errors.

5.3.5 RX_DESKEW2

A 30-bit Read/Write register that specifies the tap points for data path lanes 5 to 10 in the manual deskew mode, as listed in Table 24.

Table 24. Tap Points in Manual Deskew Mode (lanes 5 to 10)

Field	Bits	Description	Type	Default
RSVD	31:30	Reserved. Set to 0.	RO	0
TAP	29 – 0	Tap points for data path lanes 5..10 in manual deskew mode.	RW	0x0

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., the value of **RX_RESET** is 1). Writes at other times will cause errors.

5.3.6 RX_DESKEW3

A 30-bit Read/Write register that specifies the tap points for data path lanes 11 to 16 in the manual deskew mode, as listed in Table 25.

Table 25. Tap Points in Manual Deskew Mode (lanes 11 to 16)

Field	Bits	Description	Type	Default
RSVD	31:30	Reserved. Set to 0.	RO	0
TAP	29 – 0	Tap points for data path lanes 11..16 in manual deskew mode.	RW	0x0

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., the value of **RX_RESET** is 1). Writes at other times will cause errors.

5.3.7 RX_CALS

There are 32 calendar registers each of which can store up to 8 calendar entries for a total of 256 entries (the last entry is not usable). The *n*th such register (*n* = 0, …, 31) stores the calendar entries 8**n* to 8**n*+7. Each entry shall be programmed with a FIFO (queue) number assigned to this entry.

Table 26 lists the fields of the registers.

Table 26. Calendar Registers (RX)

Field	Bits	Description	Default		
			n=0	n=1	n>1
ENTRY7	31:28	Calendar[8**n*+7]	7	15	0
ENTRY6	27:24	Calendar[8**n*+6]	6	14	0
ENTRY5	23:20	Calendar[8**n*+5]	5	13	0
ENTRY4	19:16	Calendar[8**n*+4]	4	12	0
ENTRY3	15:12	Calendar[8**n*+3]	3	11	0
ENTRY2	11:8	Calendar[8**n*+2]	2	10	0
ENTRY1	7:4	Calendar[8**n*+1]	1	9	0
ENTRY0	3:0	Calendar[8**n*+0]	0	8	0

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., the value of **RX_RESET** is 1). Writes at other times will cause errors.

5.3.8 RX_FS

This register specifies the size of the FIFOs in this RX interface, as provided in Table 27. The device reads the **RX_FS** when the value of **RX_RESET** changes from 1 to 0 for the first time after chip reset. Any subsequent writes to this register have no effect.

Table 27. RX_FS Register

Field	Bits	Description	Type	Default
RSVD	31:4	Reserved. Set to 0.	RO	0
FIFO_SIZE	3:0	Size of the FIFO FS / FIFO sizes (KB) / MAX number of FIFOS 0 (default / 1 / 16 1 / 2 / 8 2 / 4 / 4 3 / 8 / 2 4 / 16 / 1 5 - 15 / Reserved / N/A	RW	0

5.3.9 RX_OP_MODE

A Read/Write register that specifies the device operation mode, as provided in Table 28. Setting bit 1 to drop all packets supercedes the settings in **RX_PORT_VALID** for the multi-port mode. Note also that the value of the operating mode bit (field M) should not be changed once the **RX_RESET** value changes from 1 to 0 the first time after chip reset.

Changes to this register take place immediately without regard to packet boundaries once modified. Because changes don't take effect on packet boundaries, packets may be split into two pieces, possibly resulting in one or two bad packets per FIFO. This register is intended to be modified as part of a link (FIFO) reset procedure where this will not happen.

Table 28. RX Interface Operating Mode

Field	Bit	Description	Type	Default
RSVD	31:2	Reserved. Set to 0.	RO	0
D	1	Drop all incoming data 0 = do not drop 1 = drop data	RW	0
M	0	Operating mode. 0 – Multi-port 1 - Clear.	RW	0

5.3.10 RX_WATERMARK

This is a 32-bit Read/Write register that specifies the high (almost full) and low (almost empty) watermarks of the FIFOs used in the SPI-4.2 flow control.

Note that the default values are set conservatively and may not result in full performance. Table 29 lists the fields of the registers:

Table 29. Flow Control High/Low Watermarks

Field	Bits	Description	Type	Default
HIGH	31:16	Data level above which FIFO enters SATISFIED status (in bytes).	RW	0x0100
LOW	15:0	Data level at or below which FIFO enters STARVING status (in bytes).	RW	0x0080

5.3.11 RX_PORT2FIFO[0..31]

These registers specify the mapping from the Port IDs of incoming data to the RX queues. Each may store up to 8 4-bit FIFO IDs.

This register is used only in the Multi-port mode (see **RX_OP_MODE**) and otherwise is ignored. Note that in Clear mode, all ports are implicitly mapped to FIFO 0.

RX_PORT2FIFO[*n*] maps Port IDs (8*n) to (8*n + 7) to RX queues. Writes to these registers take effect throughout the normal device operation. Table 30 lists the fields of the registers.

Table 30. Port ID Mapping (RX Queues)

Field	Bits	Description	Default		
			n=0	n=1	n>1
PORT7	31:28	FIFO ID mapped from port 8*n+7.	7	15	0
PORT 6	27:24	FIFO ID mapped from port 8*n+6.	6	14	0
PORT 5	23:20	FIFO ID mapped from port 8*n+5.	5	13	0
PORT 4	19:16	FIFO ID mapped from port 8*n+4.	4	12	0
PORT 3	15:12	FIFO ID mapped from port 8*n+3.	3	11	0
PORT 2	11:8	FIFO ID mapped from port 8*n+2.	2	10	0
PORT 1	7:4	FIFO ID mapped from port 8*n+1.	1	9	0
PORT 0	3:0	FIFO ID mapped from port 8*n+0.	0	8	0

5.3.12 RX_PORT_VALID[0..7]

These registers indicate which ports are valid. To indicate that a port is valid, i.e., that the corresponding entry in **RX_PORT2FIFO** is valid, a corresponding bit must be set in one of the registers. This register is used only in Multi-port mode (see **RX_OP_MODE**) and otherwise is ignored.

If any data is received for an invalid port, the data is discarded on receipt.

Writes to the register take effect throughout the normal device operation.

Note: Changes to this register take place immediately without regard to packet boundaries once modified. Because changes don't take effect on packet boundaries, packets may be split into two pieces, possibly resulting in one or two bad packets per FIFO. This register is intended to be modified as part of a link (FIFO) reset procedure where this will not happen.

RX_PORT_VALID[*n*] indicates whether ports 32**n* to 32**n*+31 are valid, as shown in Table 31.

Table 31. Valid Port Identification

Field	Bit	Description	Type	Default
PORT31	31	Port 32**n*+31 maps to a FIFO. 0 = invalid 1 = valid	RW	0
PORT30	30	Port 32**n*+30 maps to a FIFO.	RW	0
...	...	...	...	...
PORT0	0	Port 32**n* maps to a FIFO.	RW	0

5.3.13 RX_STATUS_OVERRIDE

This register has one position for each FIFO in the calendar sequence, shown in Table 32. Setting a bit to 1 will cause the normal status for a given FIFO to be overridden with a “satisfied” setting. Bit 0 corresponds to FIFO 0 and bit 15 to FIFO 15. The register defaults upon reset to 0 for all FIFOs.

This register can be used to help temporarily stall a port or SPI-4.2 interface. Note that credits previously advertised but unused by the remote sender are still valid until used.

Writes to the register take effect throughout the normal device operation.

Note: Changes to this register take place immediately without regard to packet boundaries once modified. Because changes don't take effect on packet boundaries, packets may be split into two pieces, possibly resulting in one or two bad packets per FIFO. This register is intended to be modified as part of a link (FIFO) reset procedure where this will not happen.

Table 32. Valid Port Identification

Field	Bit	Description	Type	Default
RSVD	31:16	Reserved. Set to 0.	RO	0
FIFO15	15	Force FIFO status to satisfied or not. 0 = normal operation 1 = force “satisfied”	RW	0
FIFO14	14		RW	0
...	...	...	...	...
FIFO0	0		RW	0

5.3.14 RX_OS

RX_OS is a read-only register that indicates RX device operation status according to Table 33.

Table 33. RX Interface Operation Status

Field	Bit	Description	Type	Default
RSVD	31-3	Reserved. Set to 0.	RO	0
T	2	Indicates whether the RX interface is receiving training patterns on the data lines. Reliable only if the RX interface is in sync. 0 = not in training 1 = training is active	RO	0*
L	1	Loss of synchronization 0 = in sync 1 = synchronization lost	RO	0*
F	0	SPI-4.2 RX interface has experienced a fatal error 0 = no fatal error 1 = fatal error (Should never occur. Will require a chip reset if it occurs.)	RO	0

Notes: (1) Bit 1 may indicate loss of synchronization (field L has value 1 when read) either because initial training on the data path has not been completed or because it received a sufficient number of DIP4 values on the data lines to conclude that it is out of sync - until another successful training occurs.

(2) * - These values change immediately to 1 when the clock becomes present and return to 0 when the training sequence completes.

5.3.15 RX_LINKCFG1[0..15]

This Read/Write register establishes the TX module and queue control to which the RX module sends the producer pointers. There are 16 32-bit per-FIFO queue-pair registers. For RX queue *n*, **RX_LINKCFG1[*n*]** specifies the internal route to its remote TX SPI-4.2 module control address. Writes to the registers take effect throughout the normal device operation.

Table 34 lists the fields of the registers.

Table 34. Producer Pointer Configuration

Field	Bits	Description	Type	Default
RSVD	31:20	Reserved. Set to 0.	RW	0x0
PORT	19:16	Port ID to target. 0 = SPI4 Interface #0 1 = SPI4 Interface #1 2 = SPI4 Interface #2 3 = SPI4 Interface #3 4 = SPI4 Interface #4 5 = SPI4 Interface #5	RW	0xF
CTRLADRR	15:0	Control address of the queue pair in the TX module. The least significant 3 bits are reserved and must be set to 0. Value must be (TX Q #)*8 I.e., TX queue #15 = 0x0078	RW	0x0

5.3.16 RX_LINKCFG2[0..15]

This register provides the base address and wrap mask of the TX queue. There are 16 32-bit per-FIFO queue-pair registers. For RX queue *n*, register **RX_LINKCFG2[*n*]** specifies the base address and address mask of its TX target queue.

Table 35 lists the fields of the registers.

Table 35. TX Queue Configuration

Field	Bits	Description	Type	Default
MASK	31:16	Mask of address of remote TX FIFO, used to properly wrap the local copy of remote producer pointer at the FIFO boundary. Set this value to the (2^FS*1K)-1. Where FS is the FIFO size set in TX_FS of the target TX SPI4.	RW	0x0
BASE	15:0	Base address of the FIFO within the remote TX SPI4. Set this according to the equation: Value = ((TX Q #)*(2^FS)*1K)+ 16K Where FS is the FIFO size set in TX_FS of the target TX SPI4. I.e., if FS=2, which is a 4K queue size, then the value for TX queue 3 equals (3*4*1K)+0x4000 = 12K+16K= 28K = 0x7000	RW	0x0
Writes to this register take effect throughout the normal device operation.				

5.3.17 RX_LINK_RESET

This is a 16-bit Read/Write register that has a bit for each RX queue, as listed in Table 36. Setting a bit to 1 will cause the RX queue and the linked queue on the TX side to be reset. This reset is triggered by this register on the RX side, and the entire data and control path on the RX

and linked queue on the TX side (as configured by **RX_LINKCFG1**, **RX_LINKCFG2** and **TX_LINKCFG**) will be reset. After the reset has completed, the bit will be cleared. Configuration software should poll on this event before continuing with any configuration changes.

Table 36. RX Queue Reset

Field	Bit	Description	Type	Default
Reserved	31:16	Reserved. Set to 0.	RO	0
Queue15	15	Reset for queue 15 link	RW	0
...	...	...		0
Queue0	0	Reset for queue 0 link	RW	0

Note: The per-link reset feature works only on a properly configured link. This reset is not used to reset a mis-configured link. Chip reset should be used in that event

5.3.18 RX_PKTCNT

This is a 32-bit statistics register that accumulates the number of packets received (both good with errors) by the SPI-4.2 interface.

Packets sent on non-provisioned ports, and those sent to a FIFO in an overflow condition, are not counted. Packets that cause a FIFO overflow may be counted even though they are not accepted by the interface.

The count rolls over after reaching its maximum value. Upon reset, it defaults to a value of 0.

5.3.19 RX_PKTERRCNT

This is a 16-bit statistics register that accumulates the number of packets received with errors by the SPI-4.2 interface.

RX increments **RX_PKTERRCNT** after receiving a packet that meets any of the following conditions:

1. AEOP
2. DIP4 error
3. Missing EOP
4. Missing SOP

Packets sent on non-provisioned ports, and those sent to a FIFO in an overflow condition, are not counted. Packets that cause a FIFO to overflow may be counted even though the packet is not accepted by the interface.

The count rolls over after reaching its maximum value. Upon reset, it defaults to a value of 0.

5.3.20 RX_DATACNT

This is a 32-bit statistics register that returns the most significant 32 bits of a 34-bit count of each byte received by the SPI-4.2 interface.

Data bytes sent on non-provisioned ports, and those sent to a FIFO in an overflow condition, are not counted. Data that causes a FIFO to overflow may be counted even though it is not accepted by the interface.

The count rolls over after reaching its maximum value. Upon reset, it defaults to a value of 0.

5.3.21 RX_ IP

The Interrupt Pending register is Clear On Read and it signals the interrupts listed in Table 37.

Note: Even if the corresponding bit in the interrupt mask register (**RX_IM**) is clear, an interrupt state bit may still be set but an external interrupt will not be generated.

Table 37. Interrupt Pending

Field	Bit	Description (0 – False, 1 – Interrupt True)	Type	Default
RSVD	31:8	Reserved.	RO	0
W	7	Clock Watchdog timeout has occurred (During configuration, this is normal. During operation, this implies that the interface of the link partner has stopped or become disconnected.)	RC	0
D	6	Data received for non-provisioned port	RC	0
E	5	Parity error on external SPI4 interface (See RX_DEBUG_STATUS for details) (This indicates the detection of a DIP4 error. If this happens frequently, check signal integrity or enable periodic training.)	RC	0
I	4	Parity error in internal memory	RC	0
C	3	S2A conversion failure (See RX_DEBUG_STATUS for details)	RC	0
O	2	FIFO overflow (See RX_DEBUG_STATUS for details)	RC	0
F	1	SPI-4.2 RX interface has experienced a fatal error 0 = no fatal error 1 = fatal error (Should never occur. Will require a chip reset if it occurs.)	RC	0
L	0	Loss of synchronization (The result of multiple consecutive DIP4 errors. The RX interface will initiate a training sequence with the link partner's TX interface, which should resolve the problem.)	RC	0

5.3.22 RX_IM

This is an interrupt mask register that enables or disables the generation of an external interrupt for the individual conditions reflected in the **RX_IP**. Its default value is 0 (all interrupts disabled).

5.3.23 RX_DEBUG_STATUS

This is a Clear On Read register that provides further details of interrupt conditions, listed in Table 38, for error handling and debug.

Table 38. Interrupt Conditions

Field	Bit	Description (0 – false, 1 – true)	Type	Default
RSVD	31:24	Reserved.	RO	0
	23	RX FIFO #15 Overflow	RC	0
	22	RX FIFO #14 Overflow	RC	0
	21	RX FIFO #13 Overflow	RC	0
	20	RX FIFO #12 Overflow	RC	0
	19	RX FIFO #11 Overflow	RC	0
	18	RX FIFO #10 Overflow	RC	0
	17	RX FIFO #9 Overflow	RC	0
	16	RX FIFO #8 Overflow	RC	0
	15	RX FIFO #7 Overflow	RC	0
	14	RX FIFO #6 Overflow	RC	0
	13	RX FIFO #5 Overflow	RC	0
	12	RX FIFO #4 Overflow	RC	0
	11	RX FIFO #3 Overflow	RC	0
	10	RX FIFO #2 Overflow	RC	0
	9	RX FIFO #1 Overflow	RC	0
	8	RX FIFO #0 Overflow	RC	0
	7	RX Alt SOP Address Parity Error	RC	0
	6	RX Alt data path Parity Error	RC	0
	5	RX Main SOP Path Parity Error	RC	0
	4	RX Main data Path Parity Error	RC	0
	3	RX Calendar S2A conversion error occurred	RC	0
	2	RX Alt Data S2A conversion error occurred	RC	0
	1	RX Main Data S2A conversion error occurred	RC	0
	0	RX Control S2A conversion error occurred	RC	0

5.3.23.1 Note on Overflow

Overflow is detected when a data burst could not be stored in the RX FIFO because there is no more room left in the RX FIFO. Upon detection of an overflow, the corresponding overflow bit is set in **RX_DEBUG_STATUS** and the RX FIFO enters into an overflow state

where all future bursts are discarded until the RX FIFO is reset, regardless of whether room has been gained due to draining or if a new burst is small enough to fit in the remaining space in the FIFO.

While the RX FIFO is in an overflow state, the data that was actually stored in the RX FIFO before the overflow occurred continues to be drained and will eventually be transmitted on the transmit port.

Once the RX FIFO is reset, the normal operation is resumed and incoming bursts are stored in the RX FIFO. If the first burst is not an SOP, then the new burst is stored and labeled “missing SOP” and the incomplete packet will be transmitted as an EOP abort on the outbound interface.

Depending when the overflow occurs and how long it takes for the host processor to reset the FIFO, it is possible on the outbound to have an incomplete packet (end of packet has been flushed) followed by a pause (while the host processor is resetting the **RX_FIFO**) followed by a new packet which could potentially be an ABORT EOP if the beginning of this new packet has been flushed as well.

The recommended procedure in the interrupt handler is the following:

1. Receive the interrupt
2. Read **RX_IP** and discover that the FIFO has overflowed
3. Mask off the interrupt in **RX_IM**
4. Read **RX_DEBUG_STATUS** to see which port / FIFO(s) overflowed
5. Override status for those ports to be reset
6. Reset those ports
7. Wait for link reset to complete
8. Read the debug register to see if any new FIFO(s) overflowed than the ones that were reset
9. Unmask the FIFO overflow in the IM
10. Turn off status override for those ports

If a fifo overflows and data destined for that fifo continues toflow into the chip, repeated fifo overflow interrupts will occur. That is, fifo overflow is not reported only the first time the fifo overflows. This can lead to the situation shown in Table 39.

Table 39. Continued FIFO Overflow Behavior

Chip action	Software Action
Overflow Int #0	Int Handler #0: Read IP, shows OVFL
Overflow Int #1	Int Handler #0: Read DS, shows OVFL fifo 1
	Int Handler #1: Read IP, shows OVFL
	Int Handler #1: Read DS, show 0 because it was cleared by the above read

Note: DS refers to RX_DEBUG_STATUS and IP refers to RX_IP.

This is not a problem. If the RX_DEBUG_STATUS register reads as 0 when handling a fifo overflow interrupt, the interrupt can be ignored because the overflow has already been dealt with.

5.3.24 TX_RESET

This register is used to reset the SPI-4.2 TX interface. After chip reset, this register has a default value of 3. When this register changes to 0 for the first time after hardware reset, the device reads and locks the values of **TX_FS**, **TX_OP_MODE** and **TX_FREE_THRESHOLD**. These three registers must be written only once.

A TX SPI-4.2 interface performs no other operation before this register is cleared for the first time after chip reset, except for PLL synchronization.

Field	Bit	Description	Type	Default
	15:0	This register is used to set the SPI-4.2 TX interface. When this register changes to 0 for the first time after hardware reset, the device reads and locks the values of TX_FS, TX_OP_MODE and TX_FREE_THRESHOLD.	RW	0X3

Notes: (1) All start-up register configurations should be set before clearing this register.

(2) De-asserting TX_RESET starts FIFO training.

5.3.25 TX_CAL_LM

This register specifies the length and multiplicity of the calendar, as listed in Table 40.

Table 40. Calendar Length and Multiplicity (TX)

Field	Bits	Description	Type	Default
RSVD	31:17	Reserved. Set to 0.	RO	0
E	16	Clock edge used by the status channel. 1 for rising clock edge. 0 for falling clock edge.	RW	0x1
MULT	15:8	Calendar multiplicity, i.e., number of times calendar sequence is repeated between framing patterns.	RW	0x04
LEN	7:0	Calendar length, i.e., number of calendar entries.	RW	0x10

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., when the value of **TX_RESET** is 1). Writes at other times may cause errors in the SPI-4.2 Interface.

5.3.26 TX_SYNC0

A Read/Write register that specifies the conditions for the external core to declare acquisition and loss of status synchronization. It also specifies the data training sequence multiplicity.

Table 41 lists the fields of the registers.

Table 41. Declaring Loss of Synchronization (TX)

Field	Bits	Description	Type	Default
RSVD	31:16	Reserved. Set to 0.	RO	0x00
ALPHA	15:8	Data training sequence multiplicity, i.e., number of times the 20-word data training pattern is repeated during the training phase. The SPI-4.2 Interface repeats the data training pattern 256 times if ALPHA=0.	RW	0x02
MBP	7:4	MAX_BAD_PARITY, i.e., maximum number of consecutive bad parities to receive on status channel before declaring loss of synchronization. In other words, the device tolerates up to MBP consecutive bad DIP2 frames while remaining in the good sync state. The valid range is 0 to 15.	RW	0x2
MGP	3:0	MIN_GOOD_PARITY, i.e., minimum number of consecutive good DIP2 frames received on status channel before declaring synchronization. The valid range is 1 to 15.	RW	0x5

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., when the value of **TX_RESET** is 1). Writes at other times may cause errors in the SPI-4.2 Interface.

5.3.27 TX_SYNC1

A 32-bit Read/Write register that specifies the maximum interval in cycles between training sequences on the data path interface, as specified in Table 42.

Table 42. Interval Between Training Sequences

Field	Bits	Description	Type	Default
T	31:0	Maximum interval in cycles between scheduling of training sequences on data path interface.	RW	0x00002800

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., when the value of **TX_RESET** is 1). Writes at other times may cause errors in the SPI-4.2 Interface.

The actual implementation is such that the interval between training may exceed **MAX_DATA_T** cycles by 7. The training must start on eight byte (4 clock cycle) boundary. Therefore set the least significant two bits to 0.

Setting the value of **MAX_DATA_T** to 0 will disable the sending of periodic training for the TX device.

5.3.28 TX_CORE_WATERMARK

In the SPI-4.2 TX interface there is a small elasticity FIFO. This register contains watermarks for this FIFO. These watermarks are not related to the SPI-4.2 credits. This register has a reasonable default value that is not required to be changed in the regular process of device configuration.

TX_CORE_WATERMARK is a 16-bit Read/Write register that specifies the high and low watermarks for the TX SPI-4.2 interface to assert/de-assert the almost-full signal of its data FIFO, as shown in Table 43.

Table 43. Link Layer FIFO Configuration

Field	Bits	Description	Type	Default
RSVD	31:16	Reserved. Set to 0.	RO	0x00
LOW	15:8	Watermark of FIFO occupancy below which SPI-4.2 TX interface de-asserts FIFO almost full signal. Valid range is 1 to 8-HIGH. Recommended value is 2.	RW	0x02
HIGH	7:0	Number of empty entries left below which SPI-4.2 TX interface asserts FIFO almost full signal. Valid range is 1 to 7.	RW	0x05

Note: This register should be written only when the SPI-4.2 interface is in reset (i.e., when the value of **TX_RESET** is 1). Writes at other times will cause errors in the SPI-4.2 Interface.

5.3.29 TX_CALS

There are 32 calendar registers each of which can store up to 8 calendar entries. They support a maximum of 256 calendar entries. The *n*th such register (*n* = 0: 31) stores the calendar entries 8**n* to 8**n*+7 as specified in Table 44.

Table 44. Calendar Registers (TX)

Field	Bits	Description	Default		
			n=0	n=1	n>1
ENTRY7	31:28	Calendar[8**n*+7]	7	15	0
ENTRY6	27:24	Calendar[8**n*+6]	6	14	0
ENTRY5	23:20	Calendar[8**n*+5]	5	13	0
ENTRY4	19:16	Calendar[8**n*+4]	4	12	0
ENTRY3	15:12	Calendar[8**n*+3]	3	11	0
ENTRY2	11:8	Calendar[8**n*+2]	2	10	0
ENTRY1	7:4	Calendar[8**n*+1]	1	9	0
ENTRY0	3:0	Calendar[8**n*+0]	0	8	0

Note: This register should be written only when the SPI-4.2 Interface is in reset (i.e., when the value of **TX_RESET** is 1). Writes at other times will cause errors in the SPI-4.2 Interface.

5.3.30 TX_FS

This register specifies the size of the FIFOs in this TX interface, as given in Table 45. The device reads the **TX_FS** when the value of **TX_RESET** changes from 1 to 0 for the first time after chip reset. Any subsequent writes to this register have no effect.

Table 45. TX_FS Register

Field	Bits	Description	Type	Default
RSVD	31:4	Reserved. Set to 0.	RO	0
FIFO_SIZE	3:0	Size of the FIFO FS / FIFO sizes (KB) / Max number of FIFOs 0 (default) / 1 / 16 2 / 4 / 8 3 / 4 / 4 4 / 8 / 2 5 - 15 / 16 / 1 Reserved / N/A	RW	0

5.3.31 TX_OP_MODE

A Read/Write register that specifies the device operation mode, as given in Table 46. The device reads the register once only when **TX_RESET** transitions from 1 to 0 for the first time after chip reset, and all further writes are ignored.

Table 46. TX Interface Operating Mode

Field	Bit	Description (0 – false, 1 – true)	Type	Default
Rsvd	31:1	Reserved	RO	0
M	0	Operating mode. 0 – Multi-port 1 - Clear.	RW	0

5.3.32 TX_SERVICE_LIMIT

This register specifies the service limits in SPI-4.2 credits beyond which the TX device stops servicing the next eligible FIFO even if the FIFO currently being serviced is still eligible (has credits and data available). There are two service limits, SL1 for the case where the port is STARVING and SL2 for all other cases, e.g., the status is HUNGRY.

This register may be changed on the fly during operation with no side effects.

Table 47 lists the fields of the registers.

Table 47. TX Service Limit

Field	Bits	Description	Type	Default
RSVD	31:16	Reserved. Set to 0.	RO	0x00
SL1	15:8	Service limit when STARVING (in SPI-4.2 credits)	RW	0x8
SL2	7:0	Service limit otherwise (in SPI-4.2 credits)	RW	0x4

5.3.33 TX_MAX_BURST[0..15]

There are 16-per-FIFO Read/Write registers that specify the credit refill levels.

Table 48 lists the fields of the registers.

Table 48. Credit Refill Levels

Field	Bits	Description	Type	Default
RSVD	31:16	Reserved. Set to 0.	RO	0x00
MAXBURST1	15:8	Credit refill level when receiving a STARVING status update.	RW	0x08
MAXBURST2	7:0	Credit refill level when receiving a HUNGRY status update.	RW	0x04

Writes to the register take effect throughout the normal device operation.

5.3.34 TX_FIFO2PORT[0..15]

These Read/Write registers specify the mapping from FIFO IDs to Port IDs in the Multi-port mode. This register is ignored when the SPI-4.2 interface is in Clear mode. Each register stores one 8-bit Port ID for each FIFO, and the remaining 8 bits are reserved.

Table 49 lists the fields of the registers.

Table 49. Port ID Mapping (TX Queues)

Field	Bits	Description	Type	Default
RSVD	32:8	Reserved (set to 0)	RO	0
PORT	7:0	Port ID mapped from FIFO *n*	RW	n

Writes to these registers take effect throughout the normal device operation.

5.3.35 TX_FIFO_VALID

This Read/Write register specifies which FIFOs are properly configured and expected to receive data, as specified in Table 50.

Note: This register must be set to non-default values if the SPI-4.2 interface is in use.

Table 50. Properly-Configured FIFOs

Field	Bit	Description (0 – false, 1 – true)	Type	Default
RSVD	31:16	Reserved. Set to 0.	RO	0
FIFO15	15	FIFO 15 is valid.	RW	0

Table 50. Properly-Configured FIFOs (Continued)

FIFO14	14	FIFO 14 is valid.	RW	0
...	...	...	RW	...
FIFO0	0	FIFO 0 is valid	RW	0

Incoming data addressed to an invalid FIFO is ignored.

Note: When running in Clear mode, FIFO 0 is always used and the other FIFOs are not, so this register should be set to 0x0001 for these modes.

Writes to the register take effect throughout the normal device operation.

5.3.36 TX_OS

This register indicates the SPI-4.2 TX interface operation status, as provided in Table 51.

Table 51. TX Interface Operation Status

Field	Bit	Description (0 – false, 1 – true)	Type	Default
Rsvd	31-5	Reserved.	RO	0
L*	4	TX interface loses synchronization with external peer device. Loss of synchronization 0 = in sync 1 = synchronization lost (This occurs when the status line on the interface receives consecutive DIP2 errors on its status lines or if it receives continuous framing from its link partner.)	RO	0*
F	3	SPI-4.2 TX interface has experienced a fatal error 0 = no fatal error 1 = fatal error (Should never occur. Will require a chip reset if it occurs.)	RO	0
T	2	TX interface is sending training sequence on data path.	RO	0
P	1	TX interface finds parity on status path is bad.	RO	0
S	0	TX interface data FIFO asserts stall signal.	RO	0

Notes: (1) Frequent DIP2 errors is often a sign of significant skew between the status clock and data lines, or not using opposite status clock edges on the TX and RX interfaces.

(2)* - Value changes immediately to 1 when the clock becomes present and returns to 0 when the training sequence completes.

5.3.37 TX_LINKCFG[0..15]

There are per-FIFO Read/Write registers that specify the queue pair configuration from RX SPI-4.2 interface and queue to TX SPI-4.2 interface and queue. **TX_LINKCFG[*n*]** specifies the configuration for TX FIFO *n* as given in Table 52.

Table 52. Queue Pair Configuration

Field	Bits	Description	Default
Reserved	31 – 20	Reserved. Set to 0.	0x000
PORT	19 – 16	Port ID to source. 0 = SPI4 Interface #0 1 = SPI4 Interface #1 2 = SPI4 Interface #2 3 = SPI4 Interface #3 4 = SPI4 Interface #4 5 = SPI4 Interface #5	0xF
CTRLADRR	15 – 0	Control address of RX FIFO within a given SPI4 RX module. The least significant 3 bits are ignored and should be set to 0. Value = ((RX Q#)*8)+0x0080 I.e., For RX queue 15, use 0x00F8.	0x000

Writes to this register take effect throughout the normal device operation.

5.3.38 TX_PKTCNT

This is a 32-bit statistics register that accumulates the number of packets transmitted (both good and in error) by this SPI-4.2 interface.

The count rolls over after reaching its maximum value. Upon reset, it defaults to a value of 0.

5.3.39 TX_PKTERRCNT

This is a 16-bit statistics register that accumulates the number of packets transmitted with errors by this SPI-4.2 interface.

TX increments **TX_PKTERRCNT** after sending a packet which meets any of the following conditions:

1. AEOP received by DUT RX
2. DIP4 error found on the packet by the DUT RX
3. Missing EOP
4. Missing SOP
5. Bad parity found on the packet in the DUT TX local memory

The count rolls over after reaching its maximum value. Upon reset, it defaults to a value of 0.

5.3.40 TX_DATACNT

This is a 32-bit statistics register that returns the most significant 32 bits of a 34-bit accumulator that counts each byte transmitted by this SPI-4.2 interface.

The count rolls over after reaching its maximum value. Upon reset, it defaults to a value of 0.

5.3.41 TX_IP

This is an interrupt status register that indicates various errors, as provided in Table 53. The register is Cleared On Read.

Note: Even if the corresponding bit in the interrupt mask register (**RX_IM**) is clear, an interrupt pending bit may still be set, but an external interrupt will not be posted.

Table 53. Interrupt Pending

Field	Bit	Description (0 – false, 1 – true)	Default
I	3	Clock Domain Converter failed (Should never occur. Would require a chip reset.)	0
L	2	Loss of synchronization (The result of too many consecutive DIP2 errors. Likely caused by the link partner's RX interface losing sync and requesting retraining.)	0*
F	1	SPI-4.2 TX interface has experienced a fatal error 0 = no fatal error 1 = fatal error (Should never occur. Will require a chip reset if it occurs.)	0
S	0	Bad parity found on data from local memory	0

Note: * - As soon as a clock is present, this value will transition to 1 indicating a loss of synchronization. It must be masked or cleared on interface synchronization.

5.3.42 TX_IM

The interrupt mask register enables or disables external interrupts from being generated by the conditions reflected in **TX_IP**, as provided in Table 54. There is a one-to-one correspondence between the bits in these two registers. Its default value is 0 (all interrupts disabled).

Table 54. Interrupt Mask

Field	Bit	Description (0 – false, 1 – true)	Default
I	3	Clock Domain Converter failed	0
S	2	Bad parity found on data from local memory	0
F	1	Core experienced a fatal error	0
L	0	Loss of synchronization	0*

5.3.43 TX_PLL_CTRL

This is an 8-bit Read/Write register that controls the PLL in the TX SPI-4.2 interface, as given in Table 55.

Table 55. PLL Tuning

Field	Bits	Description	Default
BYPASS_EN	7	0 – PLL normal operation, 1 – bypass external signal, BYPASS_VCO, to the clock outputs.	0
M	3 – 0	Internal VCO output divisor.	1
N	6 – 4	Input clock divisor.	5

The formula for the output frequency when the PLL is in normal operation is equal to:

```
Output Frequency = (Input Frequency * M) / N
```

5.3.44 TX_PLL_STAT

This is a 2-bit register that that reports status on whether the PLL is phase locked. On each serial read, bit 0 samples the LOCK signal from the PLL device and returns it to the reader. Bit 1 is reserved.

5.4 Watermark Recommendation

Optimal watermark settings depend on the FIFO size and the round trip latency of the SPI-4.2 data and SPI-4.2 status flow control of the FM1010 and its link partner. As there is a rich combination of devices, generally optimal values must be determined by the system designer during system development. It is generally desirable to make the available space in the FIFO as large as possible while ensuring data safety. Table 56 gives recommended watermark levels for the different FIFO sizes assuming that the timing is similar between the FM1010 and its link partner.

Table 56. Recommended Watermark Levels

FIFO Size (K)	High Watermark Default 0x0100	Low Watermark Default 0x0080
1K	256	128
2K	512	256
4K	1024	512
8K	2048	1024
16K	4096	2048

The high watermark is necessary to ensure system correctness. This data is a recommendation based on system modeling. It is anticipated that this data will be revised in future revisions of the data sheet, based on actual application experience. Please contact Intel® for updated information.

The low watermark is not necessary for system correctness. It is used to optimize scheduling across multiple ports.

5.5 Memory Parity Errors

The internal memory of the FIFOs is parity error protected. The memory stores both segment data and segment headers, and it is possible for a parity error to occur on either the segment header or the segment data.

If a parity error occurs on the segment header, then an interrupt **PARITY_ERROR** is posted in the **RX_IP** or **TX_IP** (depending on where the error occurred, on the receive or transmit FIFO) and the data path on the particular SPI-4.2 interface is locked until the chip is reset.

If a parity error occurs on the segment data while reading from the **TX_FIFO**, then an interrupt is posted on the **TX_IP** and the data packet will continue to be transmitted but will be terminated with a bad DIP4 CRC and an EOP abort.

If a parity error occurs on the segment data while reading from the **RX_FIFO**, then an interrupt is posted on the **RX_IP** and the data packet will be forwarded to the transmit FIFO with a segment header modified to indicate that the data is corrupted. When the segment is retrieved from the **TX_FIFO**, the transmitter will notice that this segment contains corrupted data and will thus force a bad DIP4 CRC at the termination of the packet.

Notes: (1) In the case where the parity error occurs in the RX FIFO, spurious data will be sent to the TX FIFO, which will also log an error. In the abstract, this may be misleading. But, by checking whether an RX FIFO error occurred at the same time, one can determine whether the error was a parity error in the RX FIFO or the TX FIFO. In any case, in a properly-functioning system, a parity error is extremely rare. When a parity error is detected, it is recommended that the device be reset.

(2) If the TX interface's downstream link partner is configured to restart synchronization on a single DIP4 error, a parity error will trigger resynchronization.

6.0 Signal, Ball, and Package Descriptions

6.1 Package Overview

The FM1010 uses the following package:

- Overall package dimensions of 45mm x 45mm
- Flip-chip-based BGA package, with attached heat spreader
- Two pinout variants are offered:
  - Improved pinout (1232 balls):44 balls on a side, seven rows deep, with an internal power and ground ring (ball pitch of 1.0mm)
  - Original pinout (1036 balls): 44 balls on a side, seven rows deep (ball pitch of 1.0mm)

Note: The improved pinout version of the device (with the part number prefix FM1010-F1232) offers improved performance/power over the original pinout configuration, enabling either higher performance for a given input voltage or lower power consumption for a given level of performance, or both. Both pinout variants are described in the sections that follow.

6.2 Power Mapping

Figure 10 shows a visual mapping of the power pins for the improved pinout version of the device (FM1010-F1232):

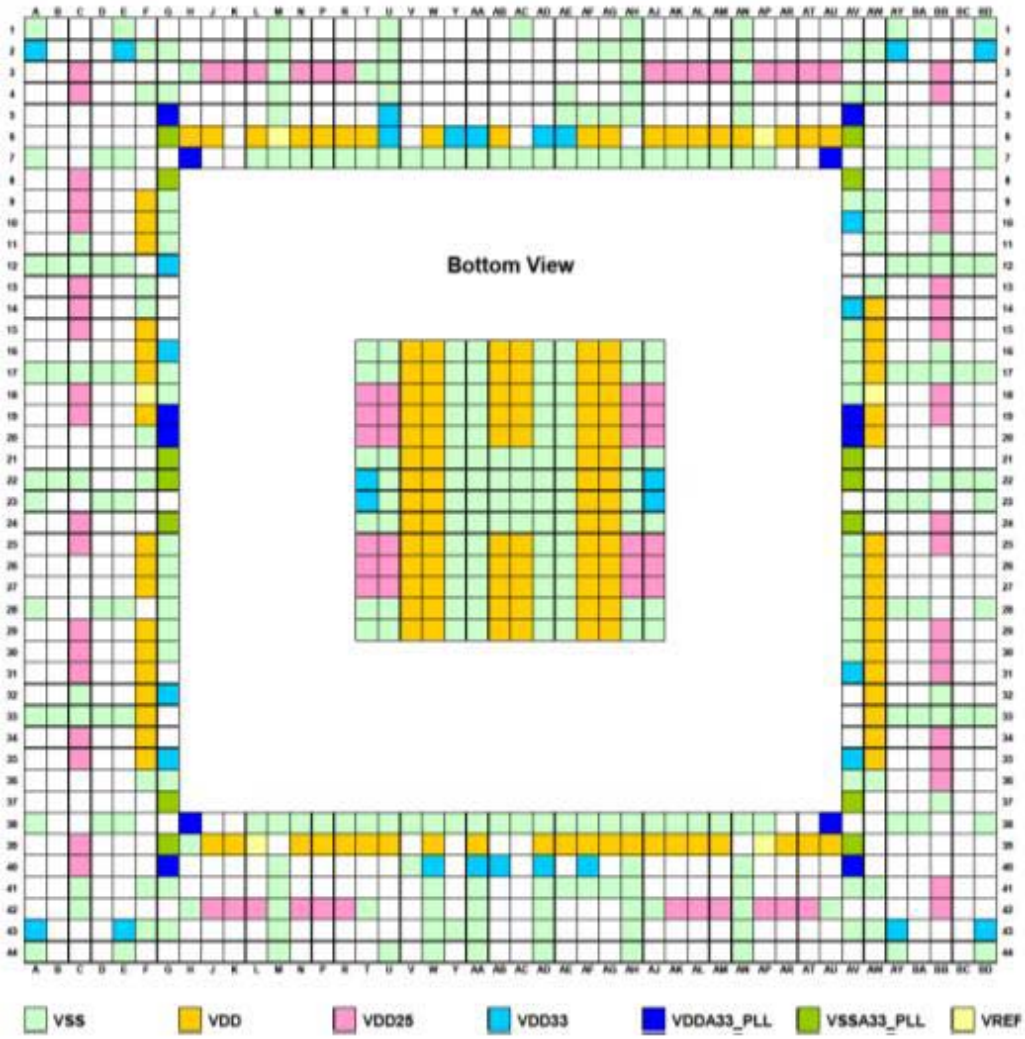


Figure 10. Power Mapping for the FM1010 1232-ball BGA Package

Notes:

(1) Consult the FM1010 Design and Layout Guide (Intel® document number: FM1010-DG) for specific information on filtering strategies.

(2) The visual mapping is identical for the original pinout version of the device (the FM1010-F1036) with the exception that the center power and ground section is removed.

6.3 Interface Mapping

Figure 11 shows a visual mapping of the interface pins for both the 1232-ball and 1036-ball versions of the device:

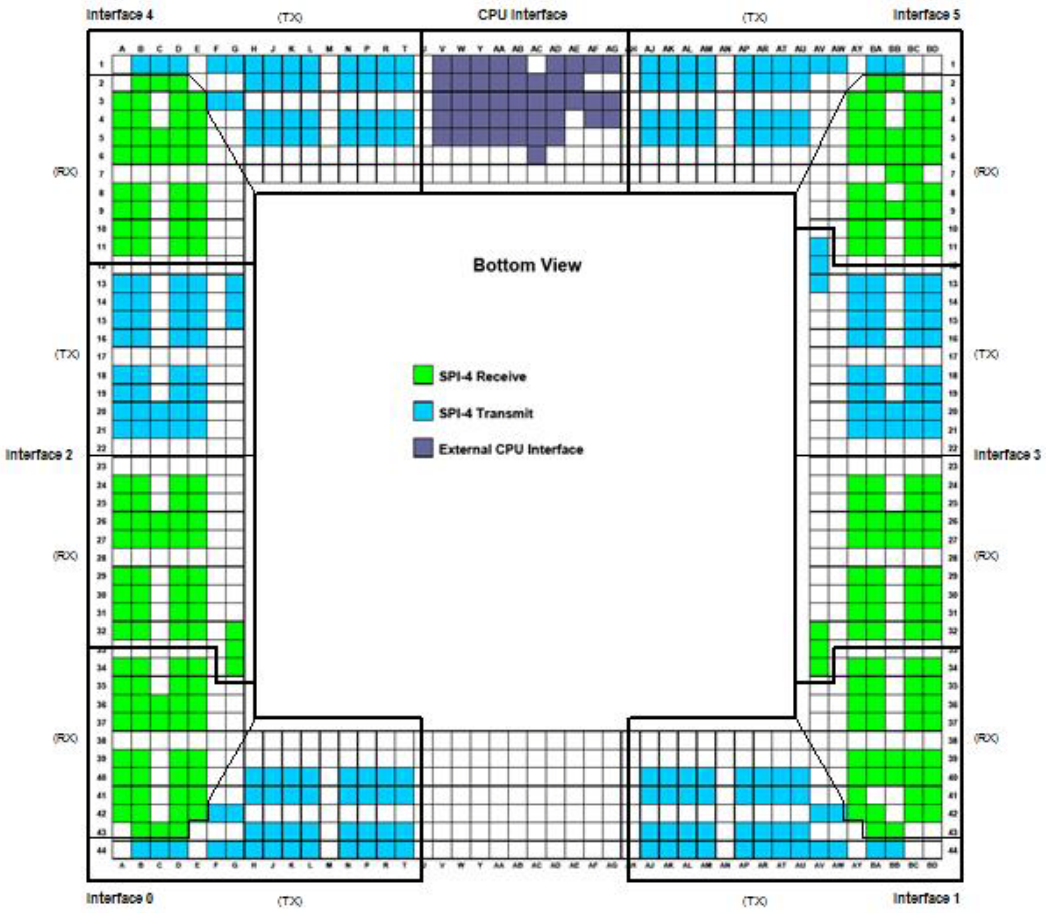


Figure 11. Interface Mapping

6.4 Signal Descriptions

This section describes the signals for both the 1232-ball and 1036-ball versions of the FM1010, providing details on the name, ball assignment, type, and use of each signal, as given in Table 57 through Table 66.

6.4.1 FM1010 Signals

Table 57. FM1010 SPI-4.2 Interface Signal Pins

Signal Name	I/O	Type	Description
TDCLKP [0:5]	Output	LVDS	SPI-4.2 TX Clock used as source synchronous strobe, +
TDCLKN [0:5]	Output	LVDS	SPI-4.2 TX Clock used as source synchronous strobe, -
TDATP [0:5] <0:15>	Output	LVDS	SPI-4.2 TX Data, +

Table 57. FM1010 SPI-4.2 Interface Signal Pins (Continued)

TDATN [0:5] <0:15>	Output	LVDS	SPI-4.2 TX Data, -
TCTLP [0:5]	Output	LVDS	SPI-4.2 TX Control bit indicating whether data is packet data or SPI-4 control word, +
TCTLN [0:5]	Output	LVDS	SPI-4.2 TX Control bit indicating whether data is packet data or SPI-4 control word, -
TSCLK [0:5]	Input	LVTTL	SPI-4.2 Status channel clock
TSTAT [0:5] <0:1>	Input	LVTTL	SPI-4.2 TX Status Channel
PWR_DN_ STRAP[0:5]	Input	LVTTL	Driver Power Down Pin per SPI-4.2 Port, active HIGH
RDCLKP [0:5]	Input	LVDS	SPI-4.2 Receive Data, Clock +
RDCLKN [0:5]	Input	LVDS	SPI-4.2 Receive Data, Clock -
RDATP [0:5] <0:15>	Input	LVDS	SPI-4.2 Receive Data, +
RDATN [0:5] <0:15>	Input	LVDS	SPI-4.2 Receive Data, -
RCTLP [0:5]	Input	LVDS	SPI-4.2 RX Control bit, +
RCTLN [0:5]	Input	LVDS	SPI-4.2 RX Control bit, -
RSCLK [0:5]	Output	LVTTL	SPI-4.2 Status Channel Clock
RSTAT [0:5] <0:1>	Output	LVTTL	SPI-4.2 RX Status Channel

Note: There are six SPI-4.2 interfaces in total.

Table 58. FM1010 CPU Interface Signal Pins

Signal Name	I/O	Type	Description
CLK	Input	LVTTL	Clock for Bus Interface
ADDR[17:1]	Input	LVTTL	Address Bus, (no single byte access)
DATA[15:0]	Bus	LVTTL	Data Bus
AS_N	Input	LVTTL	Address Strobe. Active Low
RW_N	Input	LVTTL	Read/Write – WRITE is Active Low
RW_INV	Input	LVTTL	Inverts Sense of RW_N pin. If connected to VDD33, then WRITE is active high while READ is active low. If connected to ground, then WRITE is active low while READ is active high.
DS_N	Input	LVTTL	Data Strobe. Active Low
CS_N	Input	LVTTL	Chip select. Active Low
DTACK_INV	Input	LVTTL	Strap pin. Inverts sense of DTACK_N. If connected to ground, then DTACK_N is active low. If connected to VDD33, then DTACK_N is active high.
BUSIF_RESET_N	Input	LVTTL	Reset for Bus Interface. Active Low. Internally pulled up. Can be not connected.
IGNORE_DS_N	Input	LVTTL	Strap pin. Ignore Data Strobe when HIGH. AS_N is used in this case.
SYNC_MODE	Input	LVTTL	Strap pin. Connect to VCC.
DTACK_N	Output	LVTTL	Data Acknowledge/Valid
INTR_N	Output	SE, Open Drain	Interrupt, Active Low. Connect to an external pull-up resistor.

Table 59. FM1010 JTAG Interface Signal Pins

Signal Name	I/O	Type	Description
TDI	Input	LVTTL	JTAG Input Data. Internally pulled up.
TCK	Input	LVTTL	JTAG Clock
TMS	Input	LVTTL	JTAG Test Mode. Internally pulled up.
TRST_N	Input	LVTTL	JTAG Reset Pin. Internally pulled up.
TDO	Output	LVTTL	JTAG Data Out

Note: When not using the JTAG interface, either drive the TCK pin with an external clock, or drive the TRST_N pin low. Conversely, when using the JTAG interface assert TRST_N along with chip reset to ensure proper reset of the JTAG interface prior to use.

These signal pins are reserved to Intel®.

Table 60. FM1010 Asynchronous I/F Signal Pins

Signal Name	I/O	Type	Description
ASII0	Input	LVTTL	Reserved. Leave unconnected.
ASII1	Input	LVTTL	Reserved. Leave unconnected.
ASII2	Input	LVTTL	Reserved. Leave unconnected.
ASII_E	Output	LVTTL	Reserved. Leave unconnected.
ASIO0	Output	LVTTL	Reserved. Leave unconnected.
ASIO1	Output	LVTTL	Reserved. Leave unconnected.
ASIO2	Output	LVTTL	Reserved. Leave unconnected.
ASIO_E	Input	LVTTL	Reserved. Leave unconnected.

Note: The asynchronous I/F signal lines are internally pulled down and do not need to be connected in normal operation.

Table 61. FM1010 PLL Signal Pins

Signal Name	I/O	Type	Description
PLL_LOCK [0:5]	Output	LVTTL	PLL Lock Indicator
Bypass_VCO [0:5]	Input	LVTTL	Reserved. PLL Bypass Input
Bypass_EN [0:5]	Input	LVTTL	Reserved. PLL Bypass Enable. Active High. Internally pulled down.
PLL_REFCLOCK[0:5]	Input	LVTTL	PLL Reference Clock (33 MHz -133 MHz)

Notes: (1) There is one PLL per SPI-4.2 TX block.

(2) Bypass_VCO and Bypass_EN are for test purposes, and should be left un connected for normal operation.

6.4.2 Power Supply Pins and Recommendations

Table 62. FM1010 Power Supply Signal Descriptions

Signal Name	Quantity	Type	Description
VSS	332 (249)	Power	Ground, for Core and I/O
VDD	150 (74)	Power	Core VDD (1.2 V)
VDD25	89 (65)	Power	I/O VDD (2.5 V), for LVDS
VDD33	31 (27)	Power	I/O VDD (3.3 V), for LVTTL
VSSA33_PLL	12 (12)	Power	Ground for PLL, isolated on die and in package. Connect on board.
VDDA33_PLL	14 (14)	Power	VDD (3.3 V) for PLL, isolated n die and in package. Connect on board.
VREF	6 (6)	Power	LVDS driver reference supply to set common mode

Note: The signal quantity in parentheses represent the original 1036-ball version of the device; the other numbers represent the improved 1232-ball version.

6.4.2.1 Recommended Connections

Ideally the following power supplies should be on the board containing the FM1010:

- A single 1.2 V source to supply the core VDD
- An isolated 2.5 V source to supply the LVDS I/O
- A 3.3 V supply for the LVTTL I/O signals
- A 3.3 V noise minimized source to supply the PLL (VDDA33_PLL)

6.4.2.2 Recommended Filtering

The power supply should be filtered both at the source of the power supply and local to the power supply balls on the FM1010. The power balls have been designed to take advantage of the space on the inside of the signal pins on the back side of the board for this purpose.

Note: Consult the FM1010 Design and Layout Guide (Intel® document number: FM1010 DG) for specific information on filtering strategies.

6.4.2.3 Power Budget for Improved 1232-Ball Package

- Multiple factors contribute to the power consumption of the FM1010 in operation, including:
- Number of enabled interfaces
- Core voltage
- Operating frequency of each interface
- Total cross-sectional bandwidth required for the application

Table 63 and Figure 12 provide a simple method for establishing the appropriate core voltage to support the required performance, and the resulting power budget.

To calculate the power budget for the application, follow these steps:

1. Determine the appropriate core voltage, as follows:
   - Determine the maximum frequency of the highest-speed SPI-4.2 interface in the application
   - Look up that frequency in Table 63 to determine the appropriate core voltage
2. Establish the power budget based on core voltage, as follows:
   - Estimate the total cross-sectional bandwidth expected in the application, with all devices connected to PivotPoint and transporting data as architected
   - Find the appropriate power envelope in Figure 12 for the core voltage established in step 1 above
   - Look up the total cross-sectional bandwidth for the application, along the X axis
   - That will tell you the maximum power consumption for the chosen core voltage, along the Y axis

Table 63. FM1010-F1232 Core Voltage Calculator

	Core Voltage (-5%, +10%)				
Maximum Interface Frequency	**1.1V**	**1.15V**	**1.2V**	**1.25V**	**1.3V**
<12.8Gbps (400MHz)	✔	✔	✔	✔	✔
12.8Gbps (400MHz)	✔	✔	✔	✔	✔
13.6Gbps (425MHz)		✔	✔	✔	✔
14.4Gbps (450MHz)			✔	✔	✔

Note: To support interface frequencies greater than 400MHz, order the FM1010 with the 450MHz speed grade, as follows: FM1010-F1232-450C. ✔

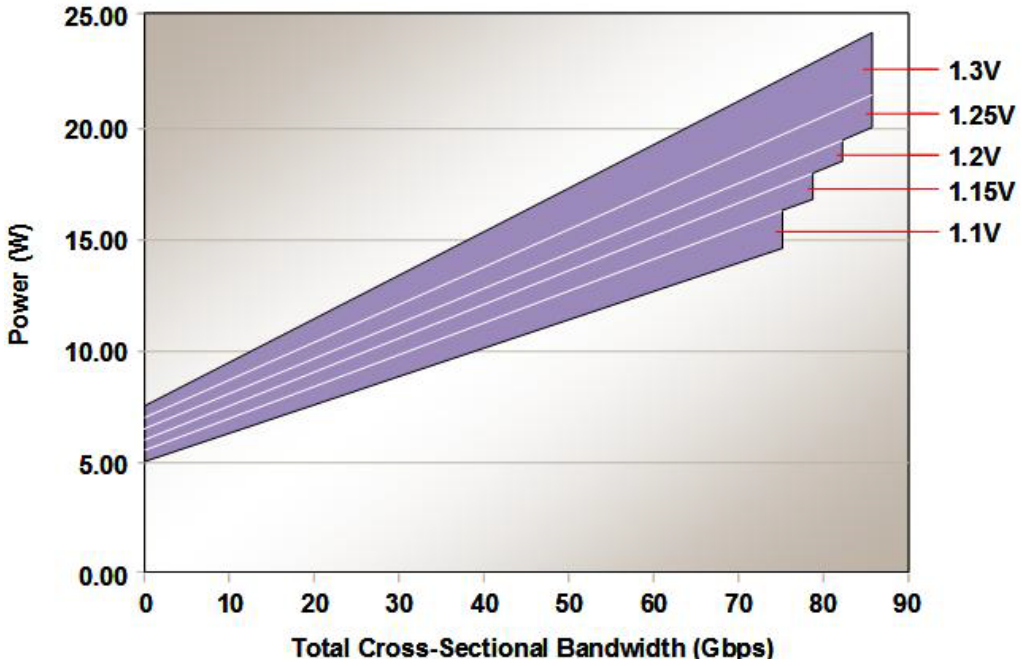


Figure 12. FM1010-F1232 Power Envelope

6.4.2.4 Power Budget for Original 1036-Ball Package

Multiple factors contribute to the power consumption of the FM1010 in operation, including:

- Number of enabled interfaces
- Core voltage
- Operating frequency of each interface
- Total cross-sectional bandwidth required for the application

Table 64 and Figure 13 provide a simple method for establishing the appropriate core voltage to support the required performance, and the resulting power budget.

To calculate the power budget for the application, follow these steps:

3. Determine the appropriate core voltage, as follows:
   — Determine the maximum frequency of the highest-speed SPI-4.2 interface in the application
   — Look up that frequency in Table 64 to determine the appropriate core voltage
4. Establish the power budget based on core voltage, as follows:
   — Estimate the total cross-sectional bandwidth expected in the application, with all devices connected to PivotPoint and transporting data as architected
   — Find the appropriate power envelope in Figure 13 for the core voltage established in step 1 above
   — Look up the total cross-sectional bandwidth for the application, along the X axis
   — That will tell you the maximum power consumption for the chosen core voltage, along the Y axis

Table 64. FM1010-F1036 Core Voltage Calculator

	Core Voltage (-5%, +10%)				
Maximum Interface Frequency	**1.2V**	**1.25V**	**1.3V**	**1.35V**	**1.4V**
<12.8Gbps (400MHz)	✔	✔	✔	✔	✔
12.8Gbps (400MHz)		✔	✔	✔	✔
13.6Gbps (425MHz)			✔	✔	✔
14.4Gbps (450MHz)				✔	✔

Note: To support interface frequencies greater than 400MHz, order the FM1010 with the 450MHz speed grade, as follows: FM1010-F1036-450C

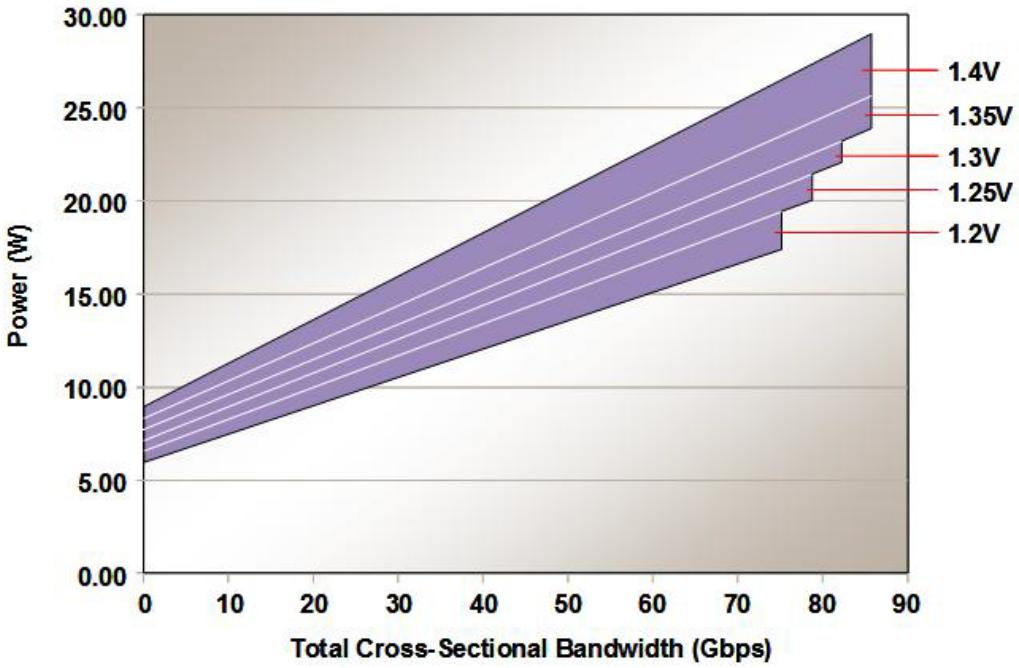


Figure 13. FM1010-F1036 Power Envelope

6.4.3 Ball Assignment

Note: For the original 1036-ball package, the center balls from T16 to AJ29 are not populated.

Table 65. Pin List in Numerical Order

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
A1	VSS	R6	VDD	AK41	TDATP_1[12]
A2	VDD33	R7	VSS	AK42	VDD25
A3	RDATP_4[15]	R38	VSS	AK43	TDATN_1[13]
A4	RDATP_4[13]	R39	VDD	AK44	TDATP_1[13]
A5	RDCLKP_4	R40	TDATN_0[12]	AL1	TDATP_5[11]
A6	RDCLKN_4	R41	TDATP_0[12]	AL2	TDATN_5[11]
A7	VSS	R42	VDD25	AL3	VDD25
A8	RDATP_4[7]	R43	TDATN_0[13]	AL4	TDATP_5[10]
A9	RDATP_4[5]	R44	TDATP_0[13]	AL5	TDATN_5[10]
A10	RDATP_4[3]	t1	TDATP_4[15]	AL6	VDD
A11	RDATP_4[1]	T2	TDATN_4[15]	AL7	VSS
A12	VSS	T3	VSS	AL38	VSS
A13	TDATP_2[15]	T4	TDATP_4[14]	AL39	VDD
A14	TDATP_2[13]	T5	TDATN_4[14]	AL40	TDATN_1[10]
A15	TDATP_2[11]	T6	VDD	AL41	TDATP_1[10]

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
A16	TDATP_2[9]	T7	VSS	AL42	VDD25
A17	VSS	T16	VSS	AL43	TDATN_1[11]
A18	TDATP_2[7]	T17	VSS	AL44	TDATP_1[11]
A19	TDATP_2[5]	T18	VDD25	AM1	TDATP_5[9]
A20	TDCLKP_2	T19	VDD25	AM2	TDATN_5[9]
A21	TDCLKN_2	T20	VDD25	AM3	VDD25
A22	VSS	T21	VSS	AM4	TDATP_5[8]
A23	VSS	T22	VDD33	AM5	TDATN_5[8]
A24	RDATP_2[15]	T23	VDD33	AM6	VDD
A25	RDATP_2[13]	T24	VSS	AM7	VSS
A26	RDCLKP_2	T25	VDD25	AM38	VSS
A27	RDCLKN_2	T26	VDD25	AM39	VDD
A28	VSS	T27	VDD25	AM40	TDATN_1[8]
A29	RDATP_2[7]	T28	VSS	AM41	TDATP_1[8]
A30	RDATP_2[5]	T29	VSS	AM42	VDD25
A31	RDATP_2[3]	T38	VSS	AM43	TDATN_1[9]
A32	RDATP_2[1]	T39	VDD	AM44	TDATP_1[9]
A33	VSS	T40	TDATN_0[14]	AN1	VSS
A34	RDATP_0[1]	T41	TDATP_0[14]	AN2	VSS
A35	RDATP_0[3]	T42	VSS	AN3	VSS
A36	RDCLKP_0	T43	TDATN_0[15]	AN4	VSS
A37	RDCLKN_0	T44	TDATP_0[15]	AN5	VSS
A38	VSS	U1	VSS	AN6	VDD
A39	RDATP_0[9]	U2	VSS	AN7	VSS
A40	RDATP_0[11]	U3	VSS	AN38	VSS
A41	RDATP_0[13]	U4	VSS	AN39	Reserved
A42	RDATP_0[15]	U5	VDD33	AN40	VSS
A43	VDD33	U6	VDD33	AN41	VSS
A44	VSS	U7	VSS	AN42	VSS
B1	TSCLK_4	U16	VSS	AN43	VSS
B2	RSCLK_4	U17	VSS	AN44	VSS
B3	RDATN_4[15]	U18	VDD25	AP1	TDATP_5[7]
B4	RDATN_4[13]	U19	VDD25	AP2	TDATN_5[7]
B5	RDATP_4[11]	U20	VDD25	AP3	VDD25
B6	RDATP_4[9]	U21	VSS	AP4	TDATP_5[6]
B7	RCTLP_4	U22	VSS	AP5	TDATN_5[6]
B8	RDATN_4[7]	U23	VSS	AP6	VREF_5
B9	RDATN_4[5]	U24	VSS	AP7	VSS
B10	RDATN_4[3]	U25	VDD25	AP38	VSS
B11	RDATN_4[1]	U26	VDD25	AP39	VREF_1

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
B12	VSS	U27	VDD25	AP40	TDATN_1[6]
B13	TDATN_2[15]	U28	VSS	AP41	TDATP_1[6]
B14	TDATN_2[13]	U29	VSS	AP42	VDD25
B15	TDATN_2[11]	U38	VSS	AP43	TDATN_1[7]
B16	TDATN_2[9]	U39	VDD	AP44	TDATP_1[7]
B17	VSS	U40	Reserved	AR1	TDATP_5[5]
B18	TDATN_2[7]	U41	Reserved	AR2	TDATN_5[5]
B19	TDATN_2[5]	U42	Reserved	AR3	VDD25
B20	TDATP_2[3]	U43	Reserved	AR4	TDATP_5[4]
B21	TDATP_2[1]	U44	VSS	AR5	TDATN_5[4]
B22	VSS	V1	INTR_N	AR6	VDD
B23	TCTLP_2	V2	SYNC_MODE	AR7	Reserved
B24	RDATN_2[15]	V3	IGNORE_DS_N	AR38	Reserved (no connect)
B25	RDATN_2[13]	V4	DTACK_INV	AR39	VDD
B26	RDATP_2[11]	V5	DTACK_N	AR40	TDATN_1[4]
B27	RDATP_2[9]	V6	CS_N	AR41	TDATP_1[4]
B28	RCTLN_2	V7	VSS	AR42	VDD25
B29	RDATN_2[7]	V16	VDD	AR43	TDATN_1[5]
B30	RDATN_2[5]	V17	VDD	AR44	TDATP_1[5]
B31	RDATN_2[3]	V18	VDD	AT1	TDATP_5[3]
B32	RDATN_2[1]	V19	VDD	AT2	TDATN_5[3]
B33	VSS	V20	VDD	AT3	VDD25
B34	RDATN_0[1]	V21	VDD	AT4	TDATP_5[2]
B35	RDATN_0[3]	V22	VDD	AT5	TDATN_5[2]
B36	RDATP_0[5]	V23	VDD	AT6	VDD
B37	RDATP_0[7]	V24	VDD	AT7	Reserved
B38	RCTLN_0	V25	VDD	AT38	Reserved
B39	RDATN_0[9]	V26	VDD	AT39	VDD
B40	RDATN_0[11]	V27	VDD	AT40	TDATN_1[2]
B41	RDATN_0[13]	V28	VDD	AT41	TDATP_1[2]
B42	RDATN_0[15]	V29	VDD	AT42	VDD25
B43	RSTAT_0[1]	V38	VSS	AT43	TDATN_1[3]
b44	TSTAT_0[1]	V39	TDO	AT44	TDATP_1[3]
C1	TSTAT_4[0]	V40	VSS	AU1	TDATP_5[1]
C2	RSTAT_4[0]	V41	TRST_N	AU2	TDATN_5[1]
C3	VDD25	V42	TMS	AU3	VDD25
C4	VDD25	V43	TDI	AU4	TDATP_5[0]
C5	RDATN_4[11]	V44	TCK	AU5	TDATN_5[0]
C6	RDATN_4[9]	W1	DS_N	AU6	VDD
C7	RCTLN_4	W2	AS_N	AU7	VDDA33_PLL

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
C8	VDD25	W3	RW_N	AU38	VDDA33_PLL
C9	VDD25	W4	RW_N_INV	AU39	VDD
C10	VDD25	W5	DATA[15]	AU40	TDATN_1[0]
C11	VSS	W6	VDD	AU41	TDATP_1[0]
C12	VSS	W7	VSS	AU42	VSS
C13	VDD25	W16	VDD	AU43	TDATN_1[1]
C14	VDD25	W17	VDD	AU44	TDATP_1[1]
C15	VDD25	W18	VDD	AV1	TDCLKP_5
C16	VSS	W19	VDD	AV2	VSS
C17	VSS	W20	VDD	AV3	TCTLP_5
C18	VDD25	W21	VDD	AV4	VSS
C19	VDD25	W22	VDD	AV5	VDDA33_PLL
C20	TDATN_2[3]	W23	VDD	AV6	VSSA33_PLL
C21	TDATN_2[1]	W24	VDD	AV7	PLL_REFCLK_5
C22	VSS	W25	VDD	AV8	VSSA33_PLL
C23	TCTLN_2	W26	VDD	AV9	VSS
C24	VDD25	W27	VDD	AV10	VDD33
C25	VDD25	W28	VDD	AV11	TSTAT_3[1]
C26	RDATN_2[11]	W29	VDD	AV12	TSTAT_3[0]
C27	RDATN_2[9]	W38	VSS	AV13	TSCLK_3
C28	RCTLP_2	W39	VDD	AV14	VDD33
C29	VDD25	W40	VDD33	AV15	VSS
C30	VDD25	W41	VSS	AV16	VSS
C31	VDD25	W42	VSS	AV17	VSS
C32	VSS	W43	VSS	AV18	VSS
C33	VSS	W44	VSS	AV19	VDDA33_PLL
C34	VDD25	Y1	DATA[14]	AV20	VDDA33_PLL
C35	VDD25	Y2	DATA[13]	AV21	VSSA33_PLL
C36	RDATN_0[5]	Y3	DATA[12]	AV22	VSSA33_PLL
C37	RDATN_0[7]	Y4	DATA[11]	AV23	PLL_REFCLK_3
C38	RCTLP_0	Y5	DATA[10]	AV24	VSSA33_PLL
C39	VDD25	Y6	VDD33	AV25	VSS
C40	VDD25	Y7	VSS	AV26	VSS
C41	VSS	Y16	VSS	AV27	VSS
C42	VSS	Y17	VSS	AV28	VSS
C43	RSTAT_0[0]	Y18	VSS	AV29	VSS
C44	TSTAT_0[0]	Y19	VSS	AV30	VSS
D1	TSTAT_4[1]	Y20	VSS	AV31	VDD33
D2	RSTAT_4[1]	Y21	VSS	AV32	RSTAT_3[1]
D3	RDATP_4[14]	Y22	VSS	AV33	RSTAT_3[0]

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
D4	RDATP_4[12]	Y23	VSS	AV34	RSCLK_3
D5	RDATP_4[10]	Y24	VSS	AV35	VDD33
D6	RDATP_4[8]	Y25	VSS	AV36	VSS
D7	VSS	Y26	VSS	AV37	VSSA33_PLL
D8	RDATP_4[6]	Y27	VSS	AV38	PLL_REFCLK_1
D9	RDATP_4[4]	Y28	VSS	AV39	VSSA33_PLL
D10	RDATP_4[2]	Y29	VSS	AV40	VDDA33_PLL
D11	RDATP_4[0]	Y38	VSS	AV41	VSS
D12	VSS	Y39	Reserved	AV42	TCTLP_1
D13	TDATP_2[14]	Y40	Reserved	AV43	VSS
D14	TDATP_2[12]	Y41	Reserved	AV44	TDCLKP_1
D15	TDATP_2[10]	Y42	Reserved	AW1	TDCLKN_5
D16	TDATP_2[8]	Y43	VSS	AW2	VSS
D17	VSS	Y44	CHIP_RESET_N	AW3	TCTLN_5
D18	TDATP_2[6]	AA1	DATA[9]	AW4	VSS
D19	TDATP_2[4]	AA2	DATA[8]	AW5	PWR_DN_STRAP_5
D20	TDATP_2[2]	AA3	DATA[7]	AW6	PLL_LOCK_5
D21	TDATP_2[0]	AA4	DATA[6]	AW7	Reserved
D22	Reserved	AA5	DATA[5]	AW8	Reserved
D23	VSS	AA6	VDD33	AW9	VSS
D24	RDATP_2[14]	AA7	VSS	AW10	VSS
D25	RDATP_2[12]	AA16	VSS	AW11	VSS
D26	RDATP_2[10]	AA17	VSS	AW12	PWR_DN_STRAP_3
D27	RDATP_2[8]	AA18	VSS	AW13	VSS
D28	VSS	AA19	VSS	AW14	VDD
D29	RDATP_2[6]	AA20	VSS	AW15	VDD
D30	RDATP_2[4]	AA21	VSS	AW16	VDD
D31	RDATP_2[2]	AA22	VSS	AW17	VDD
D32	RDATP_2[0]	AA23	VSS	AW18	VREF_3
D33	VSS	AA24	VSS	AW19	VDD
D34	RDATP_0[0]	AA25	VSS	AW20	VDD
D35	RDATP_0[2]	AA26	VSS	AW21	Reserved
D36	RDATP_0[4]	AA27	VSS	AW22	Reserved
D37	RDATP_0[6]	AA28	VSS	AW23	Reserved
D38	VSS	AA29	VSS	AW24	PLL_LOCK_3
D39	RDATP_0[8]	AA38	VSS	AW25	VDD
D40	RDATP_0[10]	AA39	VDD	AW26	VDD
D41	RDATP_0[12]	AA40	VDD33	AW27	VDD
D42	RDATP_0[14]	AA41	VSS	AW28	VDD
D43	RSCLK_0	AA42	VSS	AW29	VDD

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
D44	TSCLK_0	AA43	VSS	AW30	VDD
E1	VSS	AA44	VSS	AW31	VDD
E2	VDD33	AB1	DATA[4]	AW32	VDD
E3	RDATN_4[14]	AB2	DATA[3]	AW33	VDD
E4	RDATN_4[12]	AB3	DATA[2]	AW34	VDD
E5	RDATN_4[10]	AB4	DATA[1]	AW35	VDD
E6	RDATN_4[8]	AB5	DATA[0]	AW36	VSS
E7	VSS	AB6	VDD	AW37	Reserved
E8	RDATN_4[6]	AB7	VSS	AW38	Reserved
E9	RDATN_4[4]	AB16	VDD	AW39	PLL_LOCK_1
E10	RDATN_4[2]	AB17	VDD	AW40	PWR_DN_STRAP_1
E11	RDATN_4[0]	AB18	VDD	AW41	VSS
E12	VSS	AB19	VDD	AW42	TCTLN_1
E13	TDATN_2[14]	AB20	VDD	AW43	VSS
E14	TDATN_2[12]	AB21	VSS	AW44	TDCLKN_1
E15	TDATN_2[10]	AB22	VSS	AY1	VSS
E16	TDATN_2[8]	AB23	VSS	AY2	VDD33
E17	VSS	AB24	VSS	AY3	RDATN_5[14]
E18	TDATN_2[6]	AB25	VDD	AY4	RDATN_5[12]
E19	TDATN_2[4]	AB26	VDD	AY5	RDATN_5[10]
E20	TDATN_2[2]	AB27	VDD	AY6	RDATN_5[8]
E21	TDATN_2[0]	AB28	VDD	AY7	VSS
E22	Reserved	AB29	VDD	AY8	RDATN_5[6]
E23	VSS	AB38	VSS	AY9	RDATN_5[4]
E24	RDATN_2[14]	AB39	Reserved	AY10	RDATN_5[2]
E25	RDATN_2[12]	AB40	VDD33	AY11	RDATN_5[0]
E26	RDATN_2[10]	AB41	Reserved	AY12	VSS
E27	RDATN_2[8]	AB42	Reserved	AY13	TDATN_3[14]
E28	VSS	AB43	Reserved	AY14	TDATN_3[12]
E29	RDATN_2[6]	AB44	Reserved	AY15	TDATN_3[10]
E30	RDATN_2[4]	AC1	VSS	AY16	TDATN_3[8]
E31	RDATN_2[2]	AC2	ADDR[17]	AY17	VSS
E32	RDATN_2[0]	AC3	ADDR[16]	AY18	TDATN_3[6]
E33	VSS	AC4	ADDR[15]	AY19	TDATN_3[4]
E34	RDATN_0[0]	AC5	ADDR[14]	AY20	TDATN_3[2]
E35	RDATN_0[2]	AC6	ADDR[13]	AY21	TDATN_3[0]
E36	RDATN_0[4]	AC7	VSS	AY22	Reserved (no connect)
E37	RDATN_0[6]	AC16	VDD	AY23	VSS
E38	VSS	AC17	VDD	AY24	RDATN_3[14]
E39	RDATN_0[8]	AC18	VDD	AY25	RDATN_3[12]

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
E40	RDATN_0[10]	AC19	VDD	AY26	RDATN_3[10]
E41	RDATN_0[12]	AC20	VDD	AY27	RDATN_3[8]
E42	RDATN_0[14]	AC21	VSS	AY28	VSS
E43	VDD33	AC22	VSS	AY29	RDATN_3[6]
E44	VSS	AC23	VSS	AY30	RDATN_3[4]
F1	TDCLKP_4	AC24	VSS	AY31	RDATN_3[2]
F2	VSS	AC25	VDD	AY32	RDATN_3[0]
F3	TCTLP_4	AC26	VDD	AY33	VSS
F4	VSS	AC27	VDD	AY34	RDATN_1[0]
F5	PWR_DN_STRAP_4	AC28	VDD	AY35	RDATN_1[2]
F6	PLL_LOCK_4	AC29	VDD	AY36	RDATN_1[4]
F7	Reserved	AC38	VSS	AY37	RDATN_1[6]
F8	Reserved	AC39	Reserved	AY38	VSS
F9	VDD	AC40	Reserved	AY39	RDATN_1[8]
F10	VDD	AC41	Reserved	AY40	RDATN_1[10]
F11	VDD	AC42	Reserved	AY41	RDATN_1[12]
F12	PWR_DN_STRAP_2	AC43	Reserved	AY42	RDATN_1[14]
F13	VSS	AC44	Reserved	AY43	VDD33
F14	VSS	AD1	ADDR[12]	AY44	VSS
F15	VDD	AD2	ADDR[11]	BA1	TSTAT_5[1]
F16	VDD	AD3	ADDR[10]	BA2	RSTAT_5[1]
F17	VDD	AD4	ADDR[9]	BA3	RDATP_5[14]
F18	VREF_2	AD5	ADDR[8]	BA4	RDATP_5[12]
F19	VDD	AD6	VDD33	BA5	RDATP_5[10]
F20	VSS	AD7	VSS	BA6	RDATP_5[8]
F21	PLL_LOCK_2	AD16	VSS	BA7	VSS
F22	VSS	AD17	VSS	BA8	RDATP_5[6]
F23	Reserved	AD18	VSS	BA9	RDATP_5[4]
F24	Reserved	AD19	VSS	BA10	RDATP_5[2]
F25	VDD	AD20	VSS	BA11	RDATP_5[0]
F26	VDD	AD21	VSS	BA12	VSS
F27	VDD	AD22	VSS	BA13	TDATP_3[14]
F28	Reserved	AD23	VSS	BA14	TDATP_3[12]
F29	VDD	AD24	VSS	BA15	TDATP_3[10]
F30	VDD	AD25	VSS	BA16	TDATP_3[8]
F31	VDD	AD26	VSS	BA17	VSS
F32	VDD	AD27	VSS	BA18	TDATP_3[6]
F33	VDD	AD28	VSS	BA19	TDATP_3[4]
F34	VDD	AD29	VSS	BA20	TDATP_3[2]
F35	VDD	AD38	VSS	BA21	TDATP_3[0]

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
F36	VSS	AD39	VDD	BA22	Reserved
F37	PLL_LOCK_0	AD40	VDD33	BA23	VSS
F38	Reserved	AD41	VSS	BA24	RDATP_3[14]
F39	Reserved	AD42	VSS	BA25	RDATP_3[12]
F40	PWR_DN_STRAP_0	AD43	VSS	BA26	RDATP_3[10]
F41	VSS	AD44	VSS	BA27	RDATP_3[8]
F42	TCTLP_0	AE1	ADDR[7]	BA28	VSS
F43	VSS	AE2	ADDR[6]	BA29	RDATP_3[6]
F44	TDCLKP_0	AE3	ADDR[5]	BA30	RDATP_3[4]
G1	TDCLKN_4	AE4	VSS	BA31	RDATP_3[2]
G2	VSS	AE5	VSS	BA32	RDATP_3[0]
G3	TCTLN_4	AE6	VDD33	BA33	VSS
G4	VSS	AE7	VSS	BA34	RDATP_1[0]
G5	VDDA33_PLL	AE16	VSS	BA35	RDATP_1[2]
G6	VSSA33_PLL	AE17	VSS	BA36	RDATP_1[4]
G7	PLL_REFCLK_4	AE18	VSS	BA37	RDATP_1[6]
G8	VSSA33_PLL	AE19	VSS	BA38	VSS
G9	VSS	AE20	VSS	BA39	RDATP_1[8]
G10	VSS	AE21	VSS	BA40	RDATP_1[10]
G11	VSS	AE22	VSS	BA41	RDATP_1[12]
G12	VDD33	AE23	VSS	BA42	RDATP_1[14]
G13	TSTAT_2[1]	AE24	VSS	BA43	RSTAT_1[1]
G14	TSTAT_2[0]	AE25	VSS	BA44	TSTAT_1[1]
G15	TSCLK_2	AE26	VSS	BB1	TSTAT_5[0]
G16	VDD33	AE27	VSS	BB2	RSTAT_5[0]
G17	VSS	AE28	VSS	BB3	VDD25
G18	VSS	AE29	VSS	BB4	VDD25
G19	VDDA33_PLL	AE38	VSS	BB5	RDATN_5[11]
G20	VDDA33_PLL	AE39	VDD	BB6	RDATN_5[9]
G21	VSSA33_PLL	AE40	Reserved	BB7	RCTLN_5
G22	VSSA33_PLL	AE41	VSS	BB8	VDD25
G23	PLL_REFCLK_2	AE42	Reserved	BB9	VDD25
G24	VSSA33_PLL	AE43	Reserved	BB10	VDD25
G25	VSS	AE44	Reserved	BB11	VSS
G26	VSS	AF1	Reserved	BB12	VSS
G27	VSS	AF2	VSS	BB13	VDD25
G28	VSS	AF3	ADDR[4]	BB14	VDD25
G29	VSS	AF4	ADDR[3]	BB15	VDD25
G30	VSS	AF5	VSS	BB16	VSS
G31	VDD33	AF6	VDD	BB17	VSS

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
G32	RSTAT_2[1]	AF7	VSS	BB18	VDD25
G33	RSTAT_2[0]	AF16	VDD	BB19	VDD25
G34	RSCLK_2	AF17	VDD	BB20	TDATN_3[3]
G35	VDD33	AF18	VDD	BB21	TDATN_3[1]
G36	VSS	AF19	VDD	BB22	VSS
G37	VSSA33_PLL	AF20	VDD	BB23	TCTLN_3
G38	PLL_REFCLK_0	AF21	VDD	BB24	VDD25
G39	VSSA33_PLL	AF22	VDD	BB25	VDD25
G40	VDDA33_PLL	AF23	VDD	BB26	RDATN_3[11]
G41	VSS	AF24	VDD	BB27	RDATN_3[9]
G42	TCTLN_0	AF25	VDD	BB28	RCTLN_3
G43	VSS	AF26	VDD	BB29	VDD25
G44	TDCLKN_0	AF27	VDD	BB30	VDD25
H1	TDATP_4[1]	AF28	VDD	BB31	VDD25
H2	TDATN_4[1]	AF29	VDD	BB32	VSS
H3	VSS	AF38	VSS	BB33	VSS
H4	TDATP_4[0]	AF39	VDD	BB34	VDD25
H5	TDATN_4[0]	AF40	VDD33	BB35	VDD25
H6	VDD	AF41	VSS	BB36	VDD25
H7	VDDA33_PLL	AF42	Reserved	BB37	VSS
H38	VDDA33_PLL	AF43	Reserved	BB38	RCTLN_1
H39	VSS	AF44	VSS	BB39	RDATN_1[9]
H40	TDATN_0[0]	AG1	BUSIF_CLK	BB40	RDATN_1[11]
H41	TDATP_0[0]	AG2	VSS	BB41	VDD25
H42	VSS	AG3	ADDR[2]	BB42	VDD25
H43	TDATN_0[1]	AG4	ADDR[1]	BB43	RSTAT_1[0]
H44	TDATP_0[1]	AG5	VSS	BB44	TSTAT_1[0]
J1	TDATP_4[3]	AG6	VDD	BC1	TSCLK_5
J2	TDATN_4[3]	AG7	VSS	BC2	RSCLK_5
J3	VDD25	AG16	VDD	BC3	RDATN_5[15]
J4	TDATP_4[2]	AG17	VDD	BC4	RDATN_5[13]
J5	TDATN_4[2]	AG18	VDD	BC5	RDATP_5[11]
J6	VDD	AG19	VDD	BC6	RDATP_5[9]
J7	Reserved	AG20	VDD	BC7	RCTLP_5
J38	Reserved	AG21	VDD	BC8	RDATN_5[7]
J39	VDD	AG22	VDD	BC9	RDATN_5[5]
J40	TDATN_0[2]	AG23	VDD	BC10	RDATN_5[3]
J41	TDATP_0[2]	AG24	VDD	BC11	RDATN_5[1]
J42	VDD25	AG25	VDD	BC12	VSS
J43	TDATN_0[3]	AG26	VDD	BC13	TDATN_3[15]

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
J44	TDATP_0[3]	AG27	VDD	BC14	TDATN_3[13]
K1	TDATP_4[5]	AG28	VDD	BC15	TDATN_3[11]
K2	TDATN_4[5]	AG29	VDD	BC16	TDATN_3[9]
K3	VDD25	AG38	VSS	BC17	VSS
K4	TDATP_4[4]	AG39	VDD	BC18	TDATN_3[7]
K5	TDATN_4[4]	AG40	Reserved	BC19	TDATN_3[5]
K6	VREF_4	AG41	VSS	BC20	TDATP_3[3]
K7	Reserved	AG42	Reserved	BC21	TDATP_3[1]
K38	Reserved	AG43	Reserved	BC22	VSS
K39	VDD	AG44	Reserved	BC23	TCTLP_3
K40	TDATN_0[4]	AH1	VSS	BC24	RDATN_3[15]
K41	TDATP_0[4]	AH2	VSS	BC25	RDATN_3[13]
K42	VDD25	AH3	VSS	BC26	RDATP_3[11]
K43	TDATN_0[5]	AH4	VSS	BC27	RDATP_3[9]
K44	TDATP_0[5]	AH5	VSS	BC28	RCTLP_3
L1	TDATP_4[7]	AH6	Reserved	BC29	RDATN_3[7]
L2	TDATN_4[7]	AH7	VSS	BC30	RDATN_3[5]
L3	VDD25	AH16	VSS	BC31	RDATN_3[3]
L4	TDATP_4[6]	AH17	VSS	BC32	RDATN_3[1]
L5	TDATN_4[6]	AH18	VDD25	BC33	VSS
L6	VDD	AH19	VDD25	BC34	RDATN_1[1]
L7	VSS	AH20	VDD25	BC35	RDATN_1[3]
L38	VSS	AH21	VSS	BC36	RDATN_1[5]
L39	VREF_0	AH22	VSS	BC37	RDATN_1[7]
L40	TDATN_0[6]	AH23	VSS	BC38	RCTLP_1
L41	TDATP_0[6]	AH24	VSS	BC39	RDATP_1[9]
L42	VDD25	AH25	VDD25	BC40	RDATP_1[11]
L43	TDATN_0[7]	AH26	VDD25	BC41	RDATN_1[13]
L44	TDATP_0[7]	AH27	VDD25	BC42	RDATN_1[15]
M1	VSS	AH28	VSS	BC43	RSCLK_1
M2	VSS	AH29	VSS	BC44	TSCLK_1
M3	VSS	AH38	VSS	BD1	VSS
M4	VSS	AH39	VDD	BD2	VDD33
M5	VSS	AH40	VSS	BD3	RDATP_5[15]
M6	Reserved	AH41	VSS	BD4	RDATP_5[13]
M7	VSS	AH42	VSS	BD5	RDCLKN_5
M38	VSS	AH43	VSS	BD6	RDCLKP_5
M39	Reserved	AH44	VSS	BD7	VSS
M40	VSS	AJ1	TDATP_5[15]	BD8	RDATP_5[7]
M41	VSS	AJ2	TDATN_5[15]	BD9	RDATP_5[5]

Table 65. Pin List in Numerical Order (Continued)

Pkg Ball	Signal Name	Pkg Ball	Signal Name	Pkg Ball	Signal Name
M42	VSS	AJ3	VDD25	BD10	RDATP_5[3]
M43	VSS	AJ4	TDATP_5[14]	BD11	RDATP_5[1]
M44	VSS	AJ5	TDATN_5[14]	BD12	VSS
N1	TDATP_4[9]	AJ6	VDD	BD13	TDATP_3[15]
N2	TDATN_4[9]	AJ7	VSS	BD14	TDATP_3[13]
N3	VDD25	AJ16	VSS	BD15	TDATP_3[11]
N4	TDATP_4[8]	AJ17	VSS	BD16	TDATP_3[9]
N5	TDATN_4[8]	AJ18	VDD25	BD17	VSS
N6	VDD	AJ19	VDD25	BD18	TDATP_3[7]
N7	VSS	AJ20	VDD25	BD19	TDATP_3[5]
N38	VSS	AJ21	VSS	BD20	TDCLKN_3
N39	VDD	AJ22	VDD33	BD21	TDCLKP_3
N40	TDATN_0[8]	AJ23	VDD33	BD22	VSS
N41	TDATP_0[8]	AJ24	VSS	BD23	VSS
N42	VDD25	AJ25	VDD25	BD24	RDATP_3[15]
N43	TDATN_0[9]	AJ26	VDD25	BD25	RDATP_3[13]
N44	TDATP_0[9]	AJ27	VDD25	BD26	RDCLKN_3
P1	TDATP_4[11]	AJ28	VSS	BD27	RDCLKP_3
P2	TDATN_4[11]	AJ29	VSS	BD28	VSS
P3	VDD25	AJ38	VSS	BD29	RDATP_3[7]
P4	TDATP_4[10]	AJ39	VDD	BD30	RDATP_3[5]
P5	TDATN_4[10]	AJ40	TDATN_1[14]	BD31	RDATP_3[3]
P6	VDD	AJ41	TDATP_1[14]	BD32	RDATP_3[1]
P7	VSS	AJ42	VSS	BD33	VSS
P38	VSS	AJ43	TDATN_1[15]	BD34	RDATP_1[1]
P39	VDD	AJ44	TDATP_1[15]	BD35	RDATP_1[3]
P40	TDATN_0[10]	AK1	TDATP_5[13]	BD36	RDATP_1[5]
P41	TDATP_0[10]	AK2	TDATN_5[13]	BD37	RDATP_1[7]
P42	VDD25	AK3	VDD25	BD38	VSS
P43	TDATN_0[11]	AK4	TDATP_5[12]	BD39	RDCLKN_1
P44	TDATP_0[11]	AK5	TDATN_5[12]	BD40	RDCLKP_1
R1	TDATP_4[13]	AK6	VDD	BD41	RDATP_1[13]
R2	TDATN_4[13]	AK7	VSS	BD42	RDATP_1[15]
R3	VDD25	AK38	VSS	BD43	VDD33
R4	TDATP_4[12]	AK39	VDD	BD44	VSS
R5	TDATN_4[12]	AK40	TDATN_1[12]		

Note: For the original 1036-ball package, the center balls from Tt16 to AJ29 are not populated.

Table 66. Pin List per Functional Group

CPU Bus Interface			
Y44	CHIP_RESET_N	W3	RW_N
AF1	BUSIF_RESET_N	W4	RW_N_INV
AG1	BUSIF_CLK	V5	DTACK_N
V6	CS_N	V1	INTR_N
W2	AS_N	V4	DTACK_INV
W1	DS_N	V3	IGNORE_DS_N
V2	SYNC_MODE	AC2	ADDR[17]
W5	DATA[15]	AC3	ADDR[16]
Y1	DATA[14]	AC4	ADDR[15]
Y2	DATA[13]	AC5	ADDR[14]
Y3	DATA[12]	AC6	ADDR[13]
Y4	DATA[11]	AD1	ADDR[12]
Y5	DATA[10]	AD2	ADDR[11]
AA1	DATA[9]	AD3	ADDR[10]
AA2	DATA[8]	AD4	ADDR[9]
AA3	DATA[7]	AD5	ADDR[8]
AA4	DATA[6]	AE1	ADDR[7]
AA5	DATA[5]	AE2	ADDR[6]
AB1	DATA[4]	AE3	ADDR[5]
AB2	DATA[3]	AF3	ADDR[4]
AB3	DATA[2]	AF4	ADDR[3]
AB4	DATA[1]	AG3	ADDR[2]
AB5	DATA[0]	AG4	ADDR[1]

SPI-4.2 INTERFACE #0			
B44	TSTAT_0[1]	B43	RSTAT_0[1]
C44	TSTAT_0[0]	C43	RSTAT_0[0]
D44	TSCLK_0	D43	RSCLK_0
F44	TDCLKP_0	A36	RDCLKP_0
G44	TDCLKN_0	A37	RDCLKN_0
T44	TDATP_0[15]	A39	RDATP_0[9]
T43	TDATN_0[15]	B39	RDATN_0[9]
T41	TDATP_0[14]	D39	RDATP_0[8]
T40	TDATN_0[14]	E39	RDATN_0[8]
R44	TDATP_0[13]	B37	RDATP_0[7]
R43	TDATN_0[13]	C37	RDATN_0[7]
R41	TDATP_0[12]	D37	RDATP_0[6]
R40	TDATN_0[12]	E37	RDATN_0[6]
P44	TDATP_0[11]	B36	RDATP_0[5]

P43	TDATN_0[11]	C36	RDATN_0[5]
P41	TDATP_0[10]	D36	RDATP_0[4]
P40	TDATN_0[10]	E36	RDATN_0[4]
N44	TDATP_0[9]	A35	RDATP_0[3]
N43	TDATN_0[9]	B35	RDATN_0[3]
N41	TDATP_0[8]	D35	RDATP_0[2]
N40	TDATN_0[8]	E35	RDATN_0[2]
L44	TDATP_0[7]	A34	RDATP_0[1]
L43	TDATN_0[7]	B34	RDATN_0[1]
L41	TDATP_0[6]	A42	RDATP_0[15]
L40	TDATN_0[6]	B42	RDATN_0[15]
K44	TDATP_0[5]	D42	RDATP_0[14]
K43	TDATN_0[5]	E42	RDATN_0[14]
K41	TDATP_0[4]	A41	RDATP_0[13]
K40	TDATN_0[4]	B41	RDATN_0[13]
J44	TDATP_0[3]	D41	RDATP_0[12]
J43	TDATN_0[3]	E41	RDATN_0[12]
J41	TDATP_0[2]	A40	RDATP_0[11]
J40	TDATN_0[2]	B40	RDATN_0[11]
H44	TDATP_0[1]	D40	RDATP_0[10]
H43	TDATN_0[1]	E40	RDATN_0[10]
H41	TDATP_0[0]	D34	RDATP_0[0]
H40	TDATN_0[0]	E34	RDATN_0[0]
F42	TCTLP_0	C38	RCTLP_0
G42	TCTLN_0	B38	RCTLN_0
F40	PWR_DN_STRAP_0		
G38	PLL_REFCLK_0		
F37	LL_LOCK_0		
F38	BYPASS_VCO_0		
F39	BYPASS_EN_0		

SPI-4.2 INTERFACE #1			
BA44	TSTAT_1[1]	BA43	RSTAT_1[1]
BB44	TSTAT_1[0]	BB43	RSTAT_1[0]
BC44	TSCLK_1	BC43	RSCLK_1
AV44	TDCLKP_1	BD40	RDCLKP_1
AW44	TDCLKN_1	BD39	RDCLKN_1
AJ44	TDATP_1[15]	BD42	RDATP_1[15]
AJ43	TDATN_1[15]	BC42	RDATN_1[15]
AJ41	TDATP_1[14]	BA42	RDATP_1[14]
AJ40	TDATN_1[14]	AY42	RDATN_1[14]
AK44	TDATP_1[13]	BD41	RDATP_1[13]

AK43	TDATN_1[13]	BC41	RDATN_1[13]
AK41	TDATP_1[12]	BA41	RDATP_1[12]
AK40	TDATN_1[12]	AY41	RDATN_1[12]
AL44	TDATP_1[11]	BC40	RDATP_1[11]
AL43	TDATN_1[11]	BB40	RDATN_1[11]
AL41	TDATP_1[10]	BA40	RDATP_1[10]
AL40	TDATN_1[10]	AY40	RDATN_1[10]
AM44	TDATP_1[9]	BC39	RDATP_1[9]
AM43	TDATN_1[9]	BB39	RDATN_1[9]
AM41	TDATP_1[8]	BA39	RDATP_1[8]
AM40	TDATN_1[8]	AY39	RDATN_1[8]
AP44	TDATP_1[7]	BD37	RDATP_1[7]
AP43	TDATN_1[7]	BC37	RDATN_1[7]
AP41	TDATP_1[6]	BA37	RDATP_1[6]
AP40	TDATN_1[6]	AY37	RDATN_1[6]
AR44	TDATP_1[5]	BD36	RDATP_1[5]
AR43	TDATN_1[5]	BC36	RDATN_1[5]
AR41	TDATP_1[4]	BA36	RDATP_1[4]
AR40	TDATN_1[4]	AY36	RDATN_1[4]
AT44	TDATP_1[3]	BD35	RDATP_1[3]
AT43	TDATN_1[3]	BC35	RDATN_1[3]
AT41	TDATP_1[2]	BA35	RDATP_1[2]
AT40	TDATN_1[2]	AY35	RDATN_1[2]
AU44	TDATP_1[1]	BD34	RDATP_1[1]
AU43	TDATN_1[1]	BC34	RDATN_1[1]
AU41	TDATP_1[0]	BA34	RDATP_1[0]
AU40	TDATN_1[0]	AY34	RDATN_1[0]
AV42	TCTLP_1	BC38	RCTLP_1
AW42	TCTLN_1	BB38	RCTLN_1
AW40	PWR_DN_STRAP_1		
AV38	PLL_REFCLK_1		
AW39	PLL_LOCK_1		
AW38	BYPASS_VCO_1		
AW37	BYPASS_EN_1		

SPI-4.2 INTERFACE #2			
G13	TSTAT_2[1]	G32	RSTAT_2[1]
G14	TSTAT_2[0]	G33	RSTAT_2[0]
G15	TSCLK_2	G34	RSCLK_2
A20	TDCLKP_2	A26	RDCLKP_2
A21	TDCLKN_2	A27	RDCLKN_2
A13	TDATP_2[15]	A24	RDATP_2[15]

B13	TDATN_2[15]	B24	RDATN_2[15]
D13	TDATP_2[14]	D24	RDATP_2[14]
E13	TDATN_2[14]	E24	RDATN_2[14]
A14	TDATP_2[13]	A25	RDATP_2[13]
B14	TDATN_2[13]	B25	RDATN_2[13]
D14	TDATP_2[12]	D25	RDATP_2[12]
E14	TDATN_2[12]	E25	RDATN_2[12]
A15	TDATP_2[11]	B26	RDATP_2[11]
B15	TDATN_2[11]	C26	RDATN_2[11]
D15	TDATP_2[10]	D26	RDATP_2[10]
E15	TDATN_2[10]	E26	RDATN_2[10]
A16	TDATP_2[9]	B27	RDATP_2[9]
B16	TDATN_2[9]	C27	RDATN_2[9]
D16	TDATP_2[8]	D27	RDATP_2[8]
E16	TDATN_2[8]	E27	RDATN_2[8]
A18	TDATP_2[7]	A29	RDATP_2[7]
B18	TDATN_2[7]	B29	RDATN_2[7]
D18	TDATP_2[6]	D29	RDATP_2[6]
E18	TDATN_2[6]	E29	RDATN_2[6]
A19	TDATP_2[5]	A30	RDATP_2[5]
B19	TDATN_2[5]	B30	RDATN_2[5]
D19	TDATP_2[4]	D30	RDATP_2[4]
E19	TDATN_2[4]	E30	RDATN_2[4]
B20	TDATP_2[3]	A31	RDATP_2[3]
C20	TDATN_2[3]	B31	RDATN_2[3]
D20	TDATP_2[2]	D31	RDATP_2[2]
E20	TDATN_2[2]	E31	RDATN_2[2]
B21	TDATP_2[1]	A32	RDATP_2[1]
C21	TDATN_2[1]	B32	RDATN_2[1]
D21	TDATP_2[0]	D32	RDATP_2[0]
E21	TDATN_2[0]	E32	RDATN_2[0]
B23	TCTLP_2	C28	RCTLP_2
C23	TCTLN_2	B28	RCTLN_2
F12	PWR_DN_STRAP_2		
G23	PLL_REFCLK_2		
F21	PLL_LOCK_2		
F23	BYPASS_VCO_2		
F24	BYPASS_EN_2		

SPI-4.2 INTERFACE #3			
AV11	TSTAT_3[1]	**AV32**	RSTAT_3[1]
AV12	TSTAT_3[0]	AV33	RSTAT_3[0]

AV13	TSCLK_3	AV34	RSCLK_3
BD21	TDCLKP_3	BD27	RDCLKP_3
BD20	TDCLKN_3	BD26	RDCLKN_3
BD13	TDATP_3[15]	BD24	RDATP_3[15]
BC13	TDATN_3[15]	BC24	RDATN_3[15]
BA13	TDATP_3[14]	BA24	RDATP_3[14]
AY13	TDATN_3[14]	AY24	RDATN_3[14]
BD14	TDATP_3[13]	BD25	RDATP_3[13]
BC14	TDATN_3[13]	BC25	RDATN_3[13]
BA14	TDATP_3[12]	BA25	RDATP_3[12]
AY14	TDATN_3[12]	AY25	RDATN_3[12]
BD15	TDATP_3[11]	BC26	RDATP_3[11]
BC15	TDATN_3[11]	BB26	RDATN_3[11]
BA15	TDATP_3[10]	BA26	RDATP_3[10]
AY15	TDATN_3[10]	AY26	RDATN_3[10]
BD16	TDATP_3[9]	BC27	RDATP_3[9]
BC16	TDATN_3[9]	BB27	RDATN_3[9]
BA16	TDATP_3[8]	BA27	RDATP_3[8]
AY16	TDATN_3[8]	AY27	RDATN_3[8]
BD18	TDATP_3[7]	BD29	RDATP_3[7]
BC18	TDATN_3[7]	BC29	RDATN_3[7]
BA18	TDATP_3[6]	BA29	RDATP_3[6]
AY18	TDATN_3[6]	AY29	RDATN_3[6]
BD19	TDATP_3[5]	BD30	RDATP_3[5]
BC19	TDATN_3[5]	BC30	RDATN_3[5]
BA19	TDATP_3[4]	BA30	RDATP_3[4]
AY19	TDATN_3[4]	AY30	RDATN_3[4]
BC20	TDATP_3[3]	BD31	RDATP_3[3]
BB20	TDATN_3[3]	BC31	RDATN_3[3]
BA20	TDATP_3[2]	BA31	RDATP_3[2]
AY20	TDATN_3[2]	AY31	RDATN_3[2]
BC21	TDATP_3[1]	BD32	RDATP_3[1]
BB21	TDATN_3[1]	BC32	RDATN_3[1]
BA21	TDATP_3[0]	BA32	RDATP_3[0]
AY21	TDATN_3[0]	AY32	RDATN_3[0]
BC23	TCTLP_3	BC28	RCTLP_3
BB23	TCTLN_3	BB28	RCTLN_3
AW12	PWR_DN_STRAP_3		
AV23	PLL_REFCLK_3		
AW24	PLL_LOCK_3		
AW23	BYPASS_VCO_3		
AW21	BYPASS_EN_3		

SPI-4.2 INTERFACE #4			
D1	TSTAT_4[1]	D2	RSTAT_4[1]
C1	TSTAT_4[0]	C2	RSTAT_4[0]
B1	TSCLK_4	B2	RSCLK_4
F1	TDCLKP_4	A5	RDCLKP_4
G1	TDCLKN_4	A6	RDCLKN_4
T1	TDATP_4[15]	A3	RDATP_4[15]
T2	TDATN_4[15]	B3	RDATN_4[15]
T4	TDATP_4[14]	D3	RDATP_4[14]
T5	TDATN_4[14]	E3	RDATN_4[14]
R1	TDATP_4[13]	A4	RDATP_4[13]
R2	TDATN_4[13]	B4	RDATN_4[13]
R4	TDATP_4[12]	D4	RDATP_4[12]
R5	TDATN_4[12]	E4	RDATN_4[12]
P1	TDATP_4[11]	B5	RDATP_4[11]
P2	TDATN_4[11]	C5	RDATN_4[11]
P4	TDATP_4[10]	D5	RDATP_4[10]
P5	TDATN_4[10]	E5	RDATN_4[10]
N1	TDATP_4[9]	B6	RDATP_4[9]
N2	TDATN_4[9]	C6	RDATN_4[9]
N4	TDATP_4[8]	D6	RDATP_4[8]
N5	TDATN_4[8]	E6	RDATN_4[8]
L1	TDATP_4[7]	A8	RDATP_4[7]
L2	TDATN_4[7]	B8	RDATN_4[7]
L4	TDATP_4[6]	D8	RDATP_4[6]
L5	TDATN_4[6]	E8	RDATN_4[6]
K1	TDATP_4[5]	A9	RDATP_4[5]
K2	TDATN_4[5]	B9	RDATN_4[5]
K4	TDATP_4[4]	D9	RDATP_4[4]
K5	TDATN_4[4]	E9	RDATN_4[4]
J1	TDATP_4[3]	A10	RDATP_4[3]
J2	TDATN_4[3]	B10	RDATN_4[3]
J4	TDATP_4[2]	D10	RDATP_4[2]
J5	TDATN_4[2]	E10	RDATN_4[2]
H1	TDATP_4[1]	A11	RDATP_4[1]
H2	TDATN_4[1]	B11	RDATN_4[1]
H4	TDATP_4[0]	D11	RDATP_4[0]
H5	TDATN_4[0]	E11	RDATN_4[0]
F3	TCTLP_4	B7	RCTLP_4
G3	TCTLN_4	C7	RCTLN_4
F5	PWR_DN_STRAP_4		
G7	PLL_REFCLK_4		

F6	PLL_LOCK_4		
F7	BYPASS_VCO_4		
F8	BYPASS_EN_4		

SPI-4.2 INTERFACE #5			
BA1	TSTAT_5[1]	BA2	RSTAT_5[1]
BB1	TSTAT_5[0]	BB2	RSTAT_5[0]
BC1	TSCLK_5	BC2	RSCLK_5
AV1	TDCLKP_5	BD6	RDCLKP_5
AW1	TDCLKN_5	BD5	RDCLKN_5
AJ1	TDATP_5[15]	BD3	RDATP_5[15]
AJ2	TDATN_5[15]	BC3	RDATN_5[15]
AJ4	TDATP_5[14]	BA3	RDATP_5[14]
AJ5	TDATN_5[14]	AY3	RDATN_5[14]
AK1	TDATP_5[13]	BD4	RDATP_5[13]
AK2	TDATN_5[13]	BC4	RDATN_5[13]
AK4	TDATP_5[12]	BA4	RDATP_5[12]
AK5	TDATN_5[12]	AY4	RDATN_5[12]
AL1	TDATP_5[11]	BC5	RDATP_5[11]
AL2	TDATN_5[11]	BB5	RDATN_5[11]
AL4	TDATP_5[10]	BA5	RDATP_5[10]
AL5	TDATN_5[10]	AY5	RDATN_5[10]
AM1	TDATP_5[9]	BC6	RDATP_5[9]
AM2	TDATN_5[9]	BB6	RDATN_5[9]
AM4	TDATP_5[8]	BA6	RDATP_5[8]
AM5	TDATN_5[8]	AY6	RDATN_5[8]
AP1	TDATP_5[7]	BD8	RDATP_5[7]
AP2	TDATN_5[7]	BC8	RDATN_5[7]
AP4	TDATP_5[6]	BA8	RDATP_5[6]
AP5	TDATN_5[6]	AY8	RDATN_5[6]
AR1	TDATP_5[5]	BD9	RDATP_5[5]
AR2	TDATN_5[5]	BC9	RDATN_5[5]
AR4	TDATP_5[4]	BA9	RDATP_5[4]
AR5	TDATN_5[4]	AY9	RDATN_5[4]
AT1	TDATP_5[3]	BD10	RDATP_5[3]
AT2	TDATN_5[3]	BC10	RDATN_5[3]
AT4	TDATP_5[2]	BA10	RDATP_5[2]
AT5	TDATN_5[2]	AY10	RDATN_5[2]
AU1	TDATP_5[1]	BD11	RDATP_5[1]
AU2	TDATN_5[1]	BC11	RDATN_5[1]
AU4	TDATP_5[0]	BA11	RDATP_5[0]
AU5	TDATN_5[0]	AY11	RDATN_5[0]

AV3	TCTLP_5	BC7	RCTLP_5
AW3	TCTLN_5	BB7	RCTLN_5
AW5	PWR_DN_STRAP_5		
AV7	PLL_REFCLK_5		
AW6	PLL_LOCK_5		
AW7	BYPASS_VCO_5		
AW8	BYPASS_EN_5		

POWER PINS			
F9	VDD	AW16	VDD
F10	VDD	AW17	VDD
F11	VDD	AW19	VDD
F15	VDD	AW20	VDD
F16	VDD	AW25	VDD
F17	VDD	AW26	VDD
F19	VDD	AW27	VDD
F25	VDD	AW28	VDD
F26	VDD	AW29	VDD
F27	VDD	AW30	VDD
F29	VDD	AW31	VDD
F30	VDD	AW32	VDD
F31	VDD	AW33	VDD
F32	VDD	AW34	VDD
F33	VDD	AW35	VDD
F34	VDD	C3	VDD25
F35	VDD	C4	VDD25
H6	VDD	C8	VDD25
J6	VDD	C9	VDD25
J39	VDD	C10	VDD25
K39	VDD	C13	VDD25
L6	VDD	C14	VDD25
N6	VDD	C15	VDD25
N39	VDD	C18	VDD25
P6	VDD	C19	VDD25
P39	VDD	C24	VDD25
R6	VDD	C25	VDD25
R39	VDD	C29	VDD25
T6	VDD	C30	VDD25
T39	VDD	C31	VDD25
U39	VDD	C34	VDD25
V16	VDD	C35	VDD25
V17	VDD	C39	VDD25

V18	VDD	C40	VDD25
V19	VDD	J3	VDD25
V20	VDD	J42	VDD25
V21	VDD	K3	VDD25
V22	VDD	K42	VDD25
V23	VDD	L3	VDD25
V24	VDD	L42	VDD25
V25	VDD	N3	VDD25
V26	VDD	N42	VDD25
V27	VDD	P3	VDD25
V28	VDD	P42	VDD25
V29	VDD	R3	VDD25
W6	VDD	R42	VDD25
W16	VDD	T18	VDD25
W17	VDD	T19	VDD25
W18	VDD	T20	VDD25
W19	VDD	T25	VDD25
W20	VDD	T26	VDD25
W21	VDD	T27	VDD25
W22	VDD	U18	VDD25
W23	VDD	U19	VDD25
W24	VDD	U20	VDD25
W25	VDD	U25	VDD25
W26	VDD	U26	VDD25
W27	VDD	U27	VDD25
W28	VDD	AH18	VDD25
W29	VDD	AH19	VDD25
W39	VDD	AH20	VDD25
AA39	VDD	AH25	VDD25
AB6	VDD	AH26	VDD25
AB16	VDD	AH27	VDD25
AB17	VDD	AJ3	VDD25
AB18	VDD	AJ18	VDD25
AB19	VDD	AJ19	VDD25
AB20	VDD	AJ20	VDD25
AB25	VDD	AJ25	VDD25
AB26	VDD	AJ26	VDD25
AB27	VDD	AJ27	VDD25
AB28	VDD	AK3	VDD25
AB29	VDD	AK42	VDD25
AC16	VDD	AL3	VDD25
AC17	VDD	AL42	VDD25
AC18	VDD	AM3	VDD25

AC19	VDD	AM42	VDD25
AC20	VDD	AP3	VDD25
AC25	VDD	AP42	VDD25
AC26	VDD	AR3	VDD25
AC27	VDD	AR42	VDD25
AC28	VDD	AT3	VDD25
AC29	VDD	AT42	VDD25
AD39	VDD	AU3	VDD25
AE39	VDD	BB3	VDD25
AF6	VDD	BB4	VDD25
AF16	VDD	BB8	VDD25
AF17	VDD	BB9	VDD25
AF18	VDD	BB10	VDD25
AF19	VDD	BB13	VDD25
AF20	VDD	BB14	VDD25
AF21	VDD	BB15	VDD25
AF22	VDD	BB18	VDD25
AF23	VDD	BB19	VDD25
AF24	VDD	BB24	VDD25
AF25	VDD	BB25	VDD25
AF26	VDD	BB29	VDD25
AF27	VDD	BB30	VDD25
AF28	VDD	BB31	VDD25
AF29	VDD	BB34	VDD25
AF39	VDD	BB35	VDD25
AG6	VDD	BB36	VDD25
AG16	VDD	BB41	VDD25
AG17	VDD	BB42	VDD25
AG18	VDD	A2	VDD33
AG19	VDD	A43	VDD33
AG20	VDD	E2	VDD33
AG21	VDD	E43	VDD33
AG22	VDD	G12	VDD33
AG23	VDD	G16	VDD33
AG24	VDD	G31	VDD33
AG25	VDD	G35	VDD33
AG26	VDD	T22	VDD33
AG27	VDD	T23	VDD33
AG28	VDD	U5	VDD33
AG29	VDD	U6	VDD33
AG39	VDD	W40	VDD33
AH39	VDD	Y6	VDD33
AJ6	VDD	AA6	VDD33

AJ39	VDD	AA40	VDD33
AK6	VDD	AB40	VDD33
AK39	VDD	AD6	VDD33
AL6	VDD	AD40	VDD33
AL39	VDD	AE6	VDD33
AM6	VDD	AF40	VDD33
AM39	VDD	AJ22	VDD33
AN6	VDD	AJ23	VDD33
AR6	VDD	AV10	VDD33
AR39	VDD	AV14	VDD33
AT6	VDD	AV31	VDD33
AT39	VDD	AV35	VDD33
AU6	VDD	AY2	VDD33
AU39	VDD	AY43	VDD33
AW14	VDD	BD2	VDD33
AW15	VDD	BD43	VDD33

Note: For the original 1036-ball package, the center balls from t16 to aj29 are not populated.

LVDS REFERENCE VOLTAGE			
L39	VREF_0	AW18	VREF_3
AP39	VREF_1	K6	VREF_4
F18	VREF_2	AP6	VREF_5

GROUND PINS			
A1	VSS	AC22	VSS
A7	VSS	AC23	VSS
A12	VSS	AC24	VSS
A17	VSS	AC38	VSS
A22	VSS	AD7	VSS
A23	VSS	AD16	VSS
A28	VSS	AD17	VSS
A33	VSS	AD18	VSS
A38	VSS	AD19	VSS
A44	VSS	AD20	VSS
B12	VSS	AD21	VSS
B17	VSS	AD22	VSS
B22	VSS	AD23	VSS
B33	VSS	AD24	VSS
C11	VSS	AD25	VSS
C12	VSS	AD26	VSS
C16	VSS	AD27	VSS

C17	VSS	AD28	VSS
C22	VSS	AD29	VSS
C32	VSS	AD38	VSS
C33	VSS	AD41	VSS
C41	VSS	AD42	VSS
C42	VSS	AD43	VSS
D7	VSS	AD44	VSS
D12	VSS	AE4	VSS
D17	VSS	AE5	VSS
D23	VSS	AE7	VSS
D28	VSS	AE16	VSS
D33	VSS	AE17	VSS
D38	VSS	AE18	VSS
E1	VSS	AE19	VSS
E7	VSS	AE20	VSS
E12	VSS	AE21	VSS
E17	VSS	AE22	VSS
E23	VSS	AE23	VSS
E28	VSS	AE24	VSS
E33	VSS	AE25	VSS
E38	VSS	AE26	VSS
E44	VSS	AE27	VSS
F2	VSS	AE28	VSS
F4	VSS	AE29	VSS
F13	VSS	AE38	VSS
F14	VSS	AE41	VSS
F20	VSS	AF2	VSS
F22	VSS	AF5	VSS
F36	VSS	AF7	VSS
F41	VSS	AF38	VSS
F43	VSS	AF41	VSS
G2	VSS	AF44	VSS
G4	VSS	AG2	VSS
G9	VSS	AG5	VSS
G10	VSS	AG7	VSS
G11	VSS	AG38	VSS
G17	VSS	AG41	VSS
G18	VSS	AH1	VSS
G25	VSS	AH2	VSS
G26	VSS	AH3	VSS
G27	VSS	AH4	VSS
G28	VSS	AH5	VSS
G29	VSS	AH7	VSS

G30	VSS	AH16	VSS
G36	VSS	AH17	VSS
G41	VSS	AH21	VSS
G43	VSS	AH22	VSS
H3	VSS	AH23	VSS
H39	VSS	AH24	VSS
H42	VSS	AH28	VSS
L7	VSS	AH29	VSS
L38	VSS	AH38	VSS
M1	VSS	AH40	VSS
M2	VSS	AH41	VSS
M3	VSS	AH42	VSS
M4	VSS	AH43	VSS
M5	VSS	AH44	VSS
M7	VSS	AJ7	VSS
M38	VSS	AJ16	VSS
M40	VSS	AJ17	VSS
M41	VSS	AJ21	VSS
M42	VSS	AJ24	VSS
M43	VSS	AJ28	VSS
M44	VSS	AJ29	VSS
N7	VSS	AJ38	VSS
N38	VSS	AJ42	VSS
P7	VSS	AK7	VSS
P38	VSS	AK38	VSS
R7	VSS	AL7	VSS
R38	VSS	AL38	VSS
T3	VSS	AM7	VSS
T7	VSS	AM38	VSS
T16	VSS	AN1	VSS
T17	VSS	AN2	VSS
T21	VSS	AN3	VSS
T24	VSS	AN4	VSS
T28	VSS	AN5	VSS
T29	VSS	AN7	VSS
T38	VSS	AN38	VSS
T42	VSS	AN40	VSS
U1	VSS	AN41	VSS
U2	VSS	AN42	VSS
U3	VSS	AN43	VSS
U4	VSS	AN44	VSS
U7	VSS	AP7	VSS
U16	VSS	AP38	VSS

U17	VSS	AU42	VSS
U21	VSS	AV2	VSS
U22	VSS	AV4	VSS
U23	VSS	AV9	VSS
U24	VSS	AV15	VSS
U28	VSS	AV16	VSS
U29	VSS	AV17	VSS
U38	VSS	AV18	VSS
U44	VSS	AV25	VSS
V7	VSS	AV26	VSS
V38	VSS	AV27	VSS
V40	VSS	AV28	VSS
W7	VSS	AV29	VSS
W38	VSS	AV30	VSS
W41	VSS	AV36	VSS
W42	VSS	AV41	VSS
W43	VSS	AV43	VSS
W44	VSS	AW2	VSS
Y7	VSS	AW4	VSS
Y16	VSS	AW9	VSS
Y17	VSS	AW10	VSS
Y18	VSS	AW11	VSS
Y19	VSS	AW13	VSS
Y20	VSS	AW36	VSS
Y21	VSS	AW41	VSS
Y22	VSS	AW43	VSS
Y23	VSS	AY1	VSS
Y24	VSS	AY7	VSS
Y25	VSS	AY12	VSS
Y26	VSS	AY17	VSS
Y27	VSS	AY23	VSS
Y28	VSS	AY28	VSS
Y29	VSS	AY33	VSS
Y38	VSS	AY38	VSS
Y43	VSS	AY44	VSS
AA7	VSS	BA7	VSS
AA16	VSS	BA12	VSS
AA17	VSS	BA17	VSS
AA18	VSS	BA23	VSS
AA19	VSS	BA28	VSS
AA20	VSS	BA33	VSS
AA21	VSS	BA38	VSS
AA22	VSS	BB11	VSS

AA23	VSS	BB12	VSS
AA24	VSS	BB16	VSS
AA25	VSS	BB17	VSS
AA26	VSS	BB22	VSS
AA27	VSS	BB32	VSS
AA28	VSS	BB33	VSS
AA29	VSS	BB37	VSS
AA38	VSS	BC12	VSS
AA41	VSS	BC17	VSS
AA42	VSS	BC22	VSS
AA43	VSS	BC33	VSS
AA44	VSS	BD1	VSS
AB7	VSS	BD7	VSS
AB21	VSS	BD12	VSS
AB22	VSS	BD17	VSS
AB23	VSS	BD22	VSS
AB24	VSS	BD23	VSS
AB38	VSS	BD28	VSS
AC1	VSS	BD33	VSS
AC7	VSS	BD38	VSS
AC21	VSS	BD44	VSS

Note: For the original 1036-ball package, the center balls from t16 to aj29 are not populated.

PLL POWER & GROUND PINS			
G5	VDDA33_PLL	G6	VSSA33_PLL
G19	VDDA33_PLL	G8	VSSA33_PLL
G20	VDDA33_PLL	G21	VSSA33_PLL
G40	VDDA33_PLL	G22	VSSA33_PLL
H7	VDDA33_PLL	G24	VSSA33_PLL
H38	VDDA33_PLL	G37	VSSA33_PLL
AU7	VDDA33_PLL	G39	VSSA33_PLL
AU38	VDDA33_PLL	AV6	VSSA33_PLL
AV5	VDDA33_PLL	AV8	VSSA33_PLL
AV19	VDDA33_PLL	AV21	VSSA33_PLL
AV20	VDDA33_PLL	AV22	VSSA33_PLL
AV40	VDDA33_PLL	AV24	VSSA33_PLL
		AV37	VSSA33_PLL
		AV39	VSSA33_PLL

JTAG PINS			
V43	TDI	V41	TRST_N
V39	TDO	V44	TCK
V42	TMS		

Intel® RESERVED PINS			
AB39	SCAN_ENABLE (Do not connect)	AW23	BYP_VCO_3, pull down to Vss through 1KΩ
AB41	Reserved (Do not connectt)	AW37	Reserved (Do not connectt)
AB42	Reserved (Do not connect)	AW38	BYP_VCO_1, pull down to Vss through 1KΩ
AB43	Reserved (Do not connect)	AW7	BYP_VCO_5, pull down to Vss through 1KΩ
AB44	Reserved (Do not connect)	AW8	Reserved (Do not connect)
AC39	TEST_MODE (Do not connect)	AY22	Reserved (Do not connect)
AC40	SOSC_TEST_PAD[0] (no connect)	BA22	Reserved (Do not connect)
AC41	SOSC_TEST_PAD[1] (no connect)	D22	Reserved (Do not connect)
AC42	SOSC_TEST_PAD[2] (no connect)	E22	Reserved (Do not connect)
AC43	SOSC_TEST_PAD[3] (no connect)	F23	BYP_VCO_2, pull down to Vss through 1KΩ
AC44	SOSC_TEST_PAD[4] (no connect)	F24	Reserved (Do not connect)
AE40	ASII_E (Do not connect)	F28	Reserved pull down to Vss through 1KΩ
AE42	ASII_2 (Do not connect)	F38	BYP_VCO_0, pull down to Vss through 1KΩ
AE43	ASII_1 (Do not connect)	F39	Reserved (Do not connect)
AE44	ASII_0 (Do not connect)	F7	BYP_VCO_4, pull down to Vss through 1KΩ
AF1	BUSIF_RESET_N (Do not connect)	F8	Reserved (Do not connect)
AF42	Reserved (Do not connect)	J38	Reserved (Do not connect)
AF43	Reserved (Do not connect)	J7	Reserved (Do not connect)
AG40	ASIO_E (Do not connect)	K38	Reserved (Do not connect)
AG42	ASIO_2 (Do not connect)	K7	Reserved (Do not connect)
AG43	ASIO_1 (Do not connect)	M39	Reserved (Do not connect)
AG44	ASIO_0 (Do not connect)	M6	Reserved (Do not connect)
AH6	Reserved (Do not connect)	U40	PAD_OBS_SPI4_IP_VDD (no connect)
AN39	Reserved (Do not connect)	U41	PAD_OBS_SPI4_IP_GND (no connect)
AR38	Reserved (Do not connect)	U42	PAD_OBS_SPI4_ASYNC_VDD (no connect)
AR7	Reserved (Do not connect)	U43	PAD_OBS_SPI4_ASYNC_GND (no connect)
AT38	Reserved (Do not connect)	Y39	Reserved (Do not connect)
AT7	Reserved (Do not connect)	Y40	Reserved (Do not connect)
AW21	Reserved (Do not connect)	Y41	Reserved (Do not connect)
AW22	Reserved (Do not connect)	Y42	Reserved (Do not connect)

6.5 Package Dimensions

Example package dimension views are provided in Figure 14 through Figure 23. Dimensional references are given in Table 67 and Table 68.

6.5.1 1232-Ball Version (FM1010-F1232)

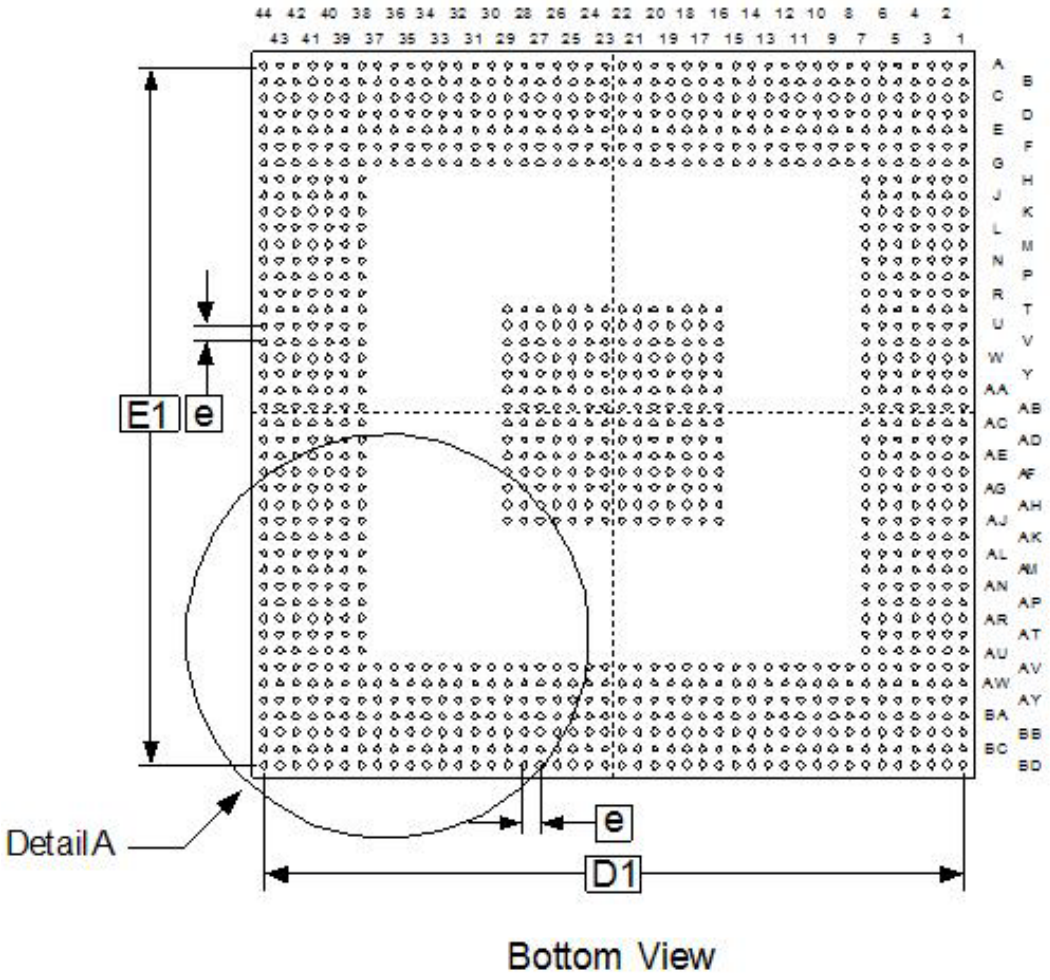


Figure 14. 1232-Ball Ball Grid Array

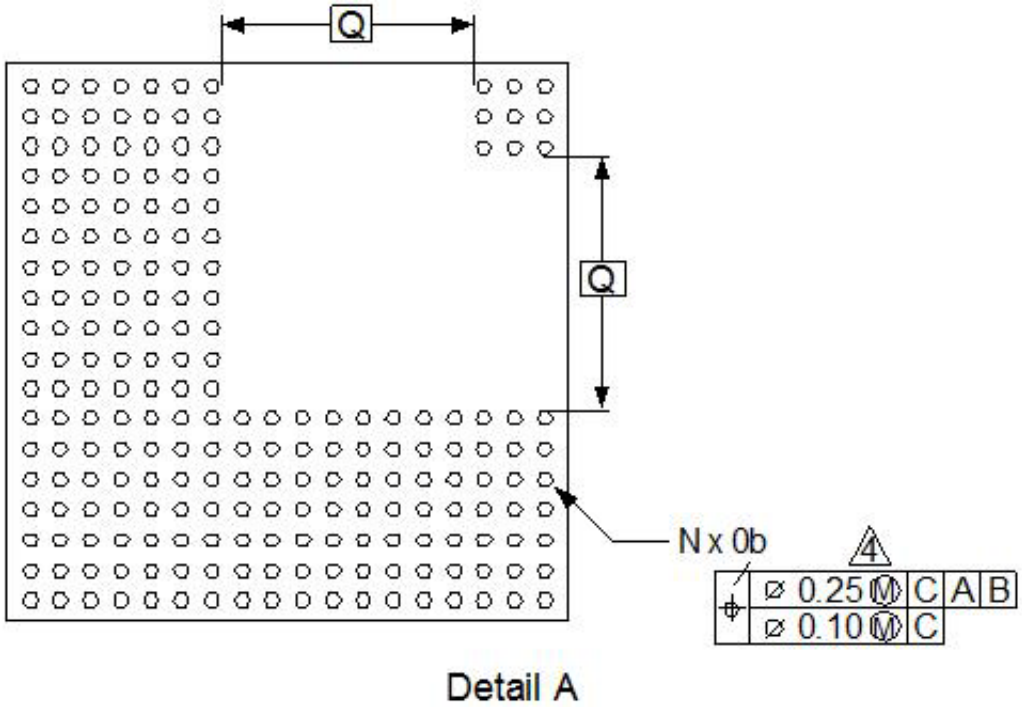


Figure 15. Expanded Detail A of Bottom View

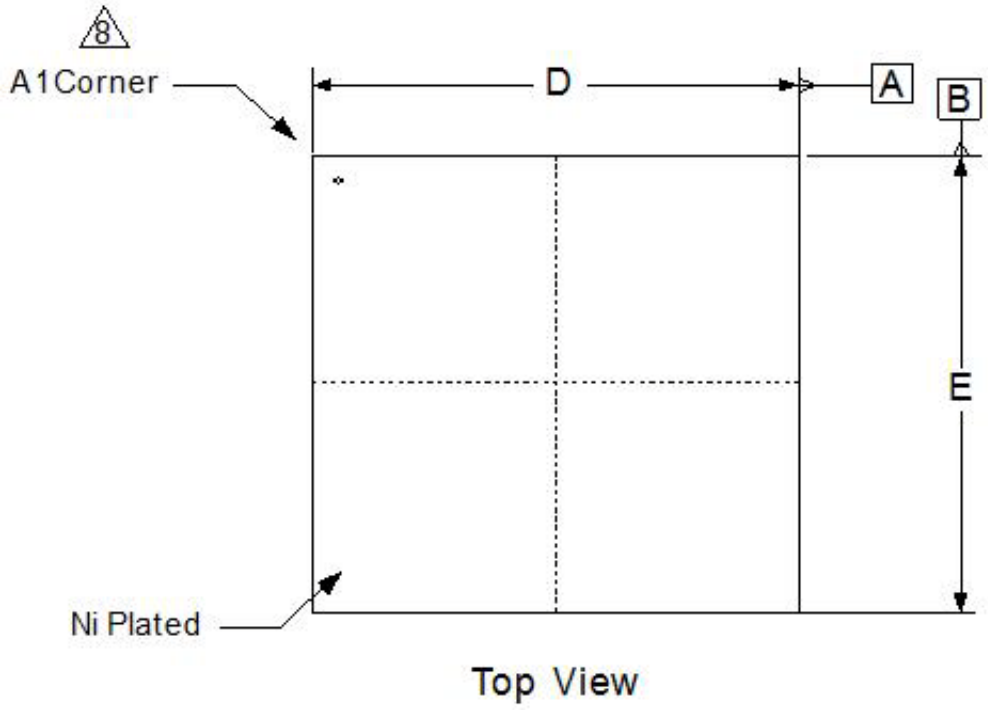


Figure 16. FM1010 Package Top View

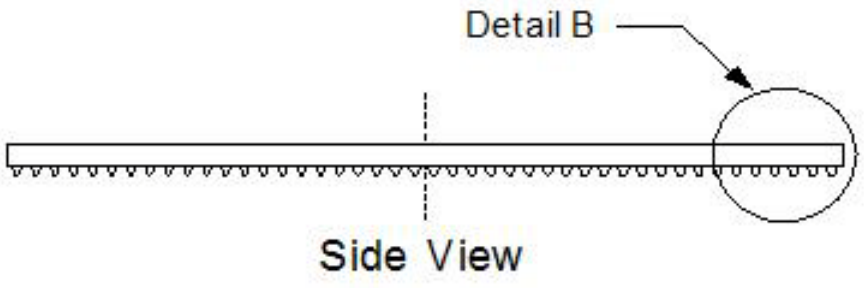


Figure 17. FM1010 Package Side View

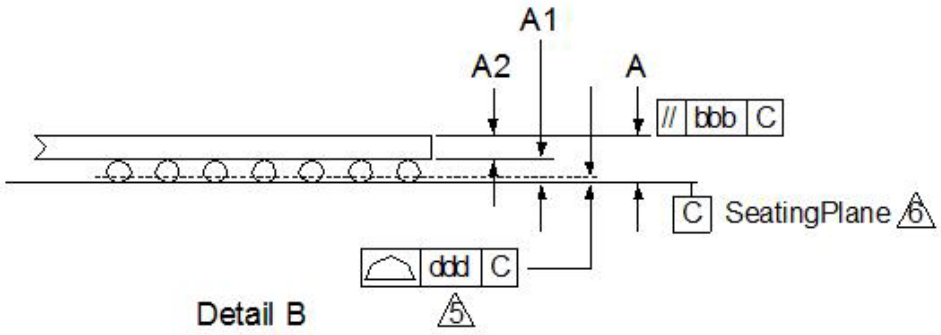


Figure 18. Expanded Detail B of Side View

6.5.2 1036-Ball Version (FM1010-F1036)

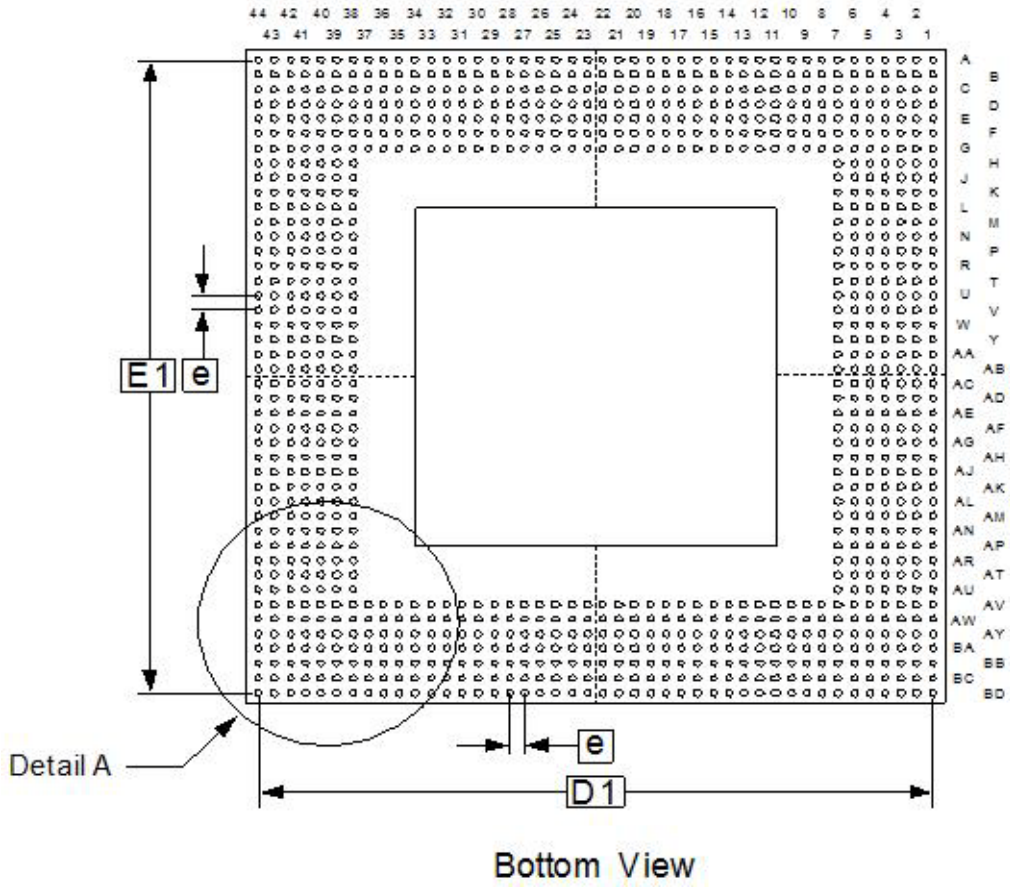


Figure 19. 1036-Ball Ball Grid Array

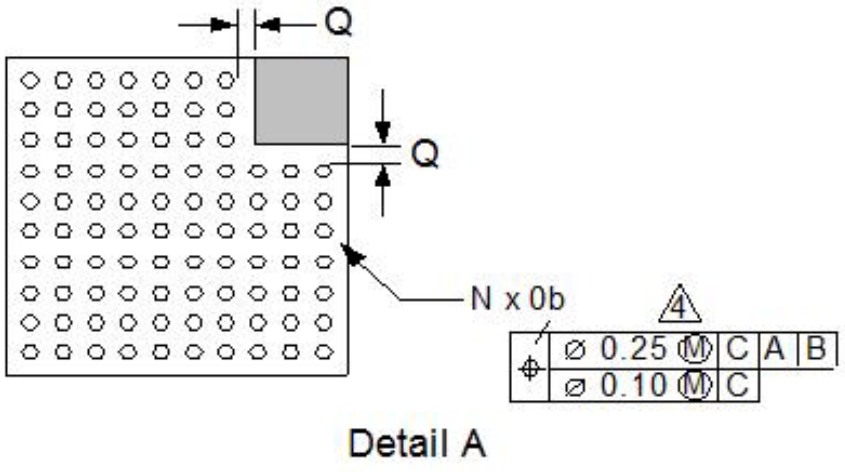


Figure 20. Expanded Detail A of Bottom View

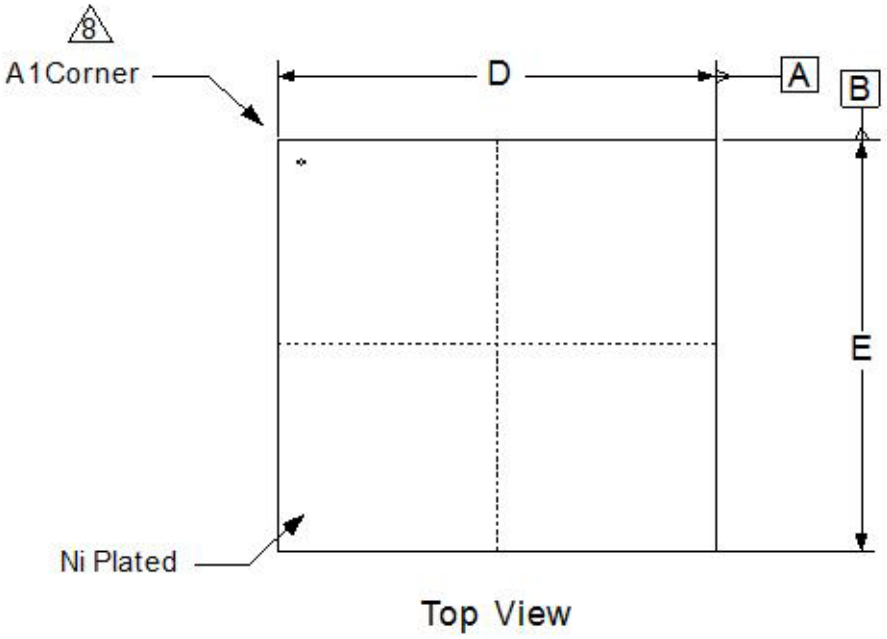


Figure 21. FM1010 Package Top View

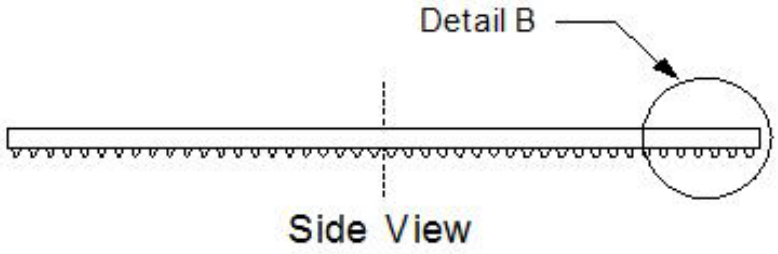


Figure 22. FM1010 Package Side View

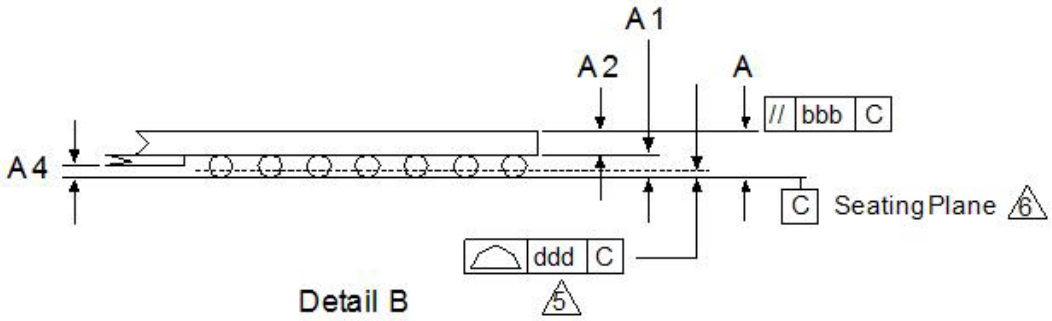


Figure 23. Expanded Detail B of Side View

Table 67. Dimensions Used in Figures, 1232-Ball Version

Dimensional References			
Reference	**Min**	**Nom**	**Max**
A	3.14	3.47	3.80
A1	0.40	0.50	0.60
A2	2.74	2.97	3.20
A4	0.075		
D	44.80	45.00	45.20
D1		43.0 BSC	
E	44.80	45.00	45.20
E1		43.0 BSC	
b	0.53	0.63	0.73
M		44	
N		1036	
bbb			0.25
ddd			0.20
e		1.00 BSC	
Q	0.35		
Ref.: JEDEC MS-034 B			

Notes:

(1) All dimensions are in millimeters.

(2) “e” represents the basic solder ball grid pitch.

(3) “M” represents the basic solder ball matrix size, and symbol “N” is the maximum allowable number of balls after depopulating.

(4) Dimension “b” is measured at the maximum solder ball diameter parallel to primary datum C.

(5) Dimension “ddd” is measured parallel to primary datum C.

(6) Primary datum C and Seating Plane are defined by the spherical crowns of the solder balls.

(7) Package surface shall be Ni plated.

(8) Black spot (or circular etch) for pin 1 identification.

(9) Dimensioning and tolerancing per ASME Y14.5M 1994

Table 68. Dimensions Used in Figures, 1036-Ball Version

Dimensional References			
Reference	**Min**	**Nom**	**Max**
A	1.25	1.45	1.65
A1	0.40	0.50	0.60
A2	0.85	0.95	1.05

Table 68. Dimensions Used in Figures, 1036-Ball Version (Continued)

D	44.80	45.00	45.20
D1	43.0 BSC		
E	44.80	45.00	45.20
E1	43.0 BSC		
b	0.53	0.63	0.73
M		44	
N	1232		
bbb			0.25
ddd			0.20
e	1.00 BSC		
Q	7.90	8.00	8.10
Ref.: JEDEC MS-034 B			

Notes: (1) All dimensions are in millimeters.

(2) “e” represents the basic solder ball grid pitch.

(3) “M” represents the basic solder ball matrix size, and symbol “N” is the maximum allowable number of balls after depopulating.

(4) Dimension “b” is measured at the maximum solder ball diameter parallel to primary datum C.

(5) Dimension “ddd” is measured parallel to primary datum C.

(6) Primary datum C and Seating Plane are defined by the spherical crowns of the solder balls.

(7) Package surface shall be Ni plated.

(8) Black spot (or circular etch) for pin 1 identification.

(9) Dimensioning and tolerancing per ASME Y14.5M 1994

6.6 Recommended Heat Sink Vendors

It is anticipated that a heat sink will be required for many applications. Intel® has qualified a list of heat sink vendors for the FM1010's BGA package.

Recommended airflow is 200f/min, with a heat sink measuring 45mm (w) x 45mm (h).

Table 69 provides a list of heat sink vendors.

Table 69. Alphabetical Listing of BGA Heat Sink Vendors

Vendors	Location	Phone
Aavid Thermal Technology	Laconia, NH	(603) 527-2152
Chip Coolers	Warwick, RI	(800) 227-0254

Table 69. Alphabetical Listing of BGA Heat Sink Vendors (Continued)

IERC	Burbank, CA	(818) 842-7277
R-Theta	Buffalo, NY	(800) 388-5428
Sanyo Denki	Torrance, CA	(310) 783-5400
Thermalloy	Dallas, TX	(214) 243-4321
Wakefield Engineering	Wakefield, MA	(617) 246-0874

7.0 Document Revision Information

The following table lists the changes made to the FM1010 Datasheet resulting in the publication of a new revision.

7.1 Nomenclature

Document revisions are placed in either of two categories to allow the user to quickly focus on changes of a substantive nature (Category 1), that is, changes that may have an impact on system or board level design.

Category 1 changes are specification clarifications or changes and include modifications to the current published specifications, or describe a specification in greater detail or further highlight a specification's impact to a complex design situation. These clarifications and changes will be incorporated in any new release of the specification or other affected document.

Category 2 changes are documentation changes and include corrections for typos, errors, or omissions from the current published specifications. These will be incorporated in any new release of the specification.

7.2 Rev 1.1 to 1.2 Changes

	Page	Category		Description
		1	2	
1	68	X		Correct the definition of IGNORE_DS_N pin. Was : Ignore data strobe when LOW. Is: Ignore data strobe when HIGH.
2	31-35		X	Change to consistent naming for power supplies: LVDS Power Supply: V_{DD25} LVTTL Power Supply: V_{DD33} PLL Analog Power Supply: V_{DDA33_PLL}

7.3 Rev 1.2 to 1.3 Changes

	Page	Category		Description
		1	2	
1	28		X	Figure 8 title mis-labeled. Change to, “CPU Bus Interface Timing Diagram”.
2	28		X	Add note to Figure 8 describing the requirement for dual 16b read/write operations to form a complete 32b transaction.
3	95		X	Table 65. Clarify “no connect” status of a number of pins in this table that had no instructions for connection during normal operation.
4	95	X		Table 65. Pins aw23, aw38, aw37, f23, f38 and f7 changed from “no connect” to “connect to ground”.
5	68	X		Table 57. Reverse the sense of the RW_N pin. In actual operation, a WRITE (not READ) operation is signaled by setting this pin LOW
6	68	X		Table 57. Change the definition of the RW_INV pin to be consistent with the RW_N pin (change #5).
7	69		X	Table 59. Pins ASII0, ASII1, ASII2, ASIOE relabeled as, “no connect”
8	69		X	Table 60. BYPASS_VCO[0:5] and BYPASS_EN[0:5] labeled as, “reserved, no connect”
9	Many		X	Alphanumeric ball assignments changed to uppercase letters only. Throughout document.
10	103		X	Add section 7, a detailed document revision description

7.4 Rev 1.3 to 1.4 Changes

	Page	Category		Description
		1	2	
1	68		X	Table 57: Invert the sense of the RW_INV pin
2	28		X	Add note to Figure 8 describing the requirement for dual 16b read/write operations to form a complete 32b transaction.
3	13	X		Eliminate reference to LVDS status channels
4	54		X	Clarify overflow interrupt behavior
5	101	X		Increase package thickness for 1232-ball package

7.5 Rev 1.4 to 1.41 Changes

	Page	Category		Description
		1	2	
1	101		X	Table 67: Correct Dimension “A” in table.

7.6 Rev 1.41 to 2.0 Changes

	Page	Category		Description
		1	2	
1	61		X	Table 63: Swap definitions for bits 2 and 0. 2 is Loss of Sync, and 0 is Bad parity.
2	95		X	Intel® reserved pin table: Changed “no connect” to “Do not connect”, and changed AW23, AW38, AW7, F23, F38 and F7 to “pull down through 1K ohm.
3	21		X	3.1.5.1 #2: Change Port ID’s to FIFO ID’s (correction)
4	23		X	3.1.5.2: add clarifying text
5	31		X	Table 4: Added more flexible core voltage requirements.
6	29		X	3.6.1: Rx jitter tolerance changed from 125 ps to 0.1 UI
7	30		X	RSTAT jitter spec eliminated.
8	46		X	Add notes to RX_PORT_VALID, RX_OP_MODE and STATUS_OVERRIDE describing the possible effects of changing theses registers while not part of link RESET.
9	25		X	Section 3.2.2: added step 3 to avoid the possibility of unpredicted behavior on link reset.

§ §

NOTE: ***This page intentionally left blank.***