BROADCOM

Data Sheet

BCM52311

Knowledge-Based Processor

General Description

The BCM52311 Knowledge-Based Processor (KBP) performs high-speed operations on large-rule databases for a wide range of telecommunications applications, including WAN, MAN, and Enterprise switches and routers. It provides network awareness and enables real-time modifications and updates to the routing configuration, making it ideal for packet classification, policy enforcement, and forwarding.

This family of KBPs addresses next-generation classification requirements through high-performance parallel decisions and improved-entry storage capabilities. Up to eight parallel operations allow the device to reach decision speeds of multiple billion decisions per second (BDPS). Embedded error correction circuitry (ECC) improves system testability and operational reliability. The key processing unit (KPU) enables efficient interface transfers with flexible search key construction.

This KBP seamlessly connects to the Jericho BCM88670, Arad Plus BCM88660, and Arad BCM88650.

Features

- Dual host enables two host devices to connect to one KBP.
- Connectivity provided to NPUs that can produce search keys through two independent ports.
- Device available in 2048k/1024k/512k 40b database entries.
- KBP tables are width configurable to 80/160/320/480/640 bits.
- User data array for associated data, width configurable as 32/64/128/256 bits.
- Up to eight parallel searches enabling up to eight results per operation.
- Simultaneous multithreading (SMT) operation.
- NetRoute forwarding solution for longest prefix match (LPM).
- NetACL solution for access control list.
- Logical tables provide support for intelligent database management.
- Key processing unit (KPU) for flexible search key construction.
- Result buffer provides programmability for flexible routing of search results.
- Range matching for efficient storage utilization.
- ECC on user data and database array. Parity protection on all embedded memories.
- Background ECC scan for database entries with provision for 1b auto correction and 2b error detection.

Figure 1: Functional Block Diagram

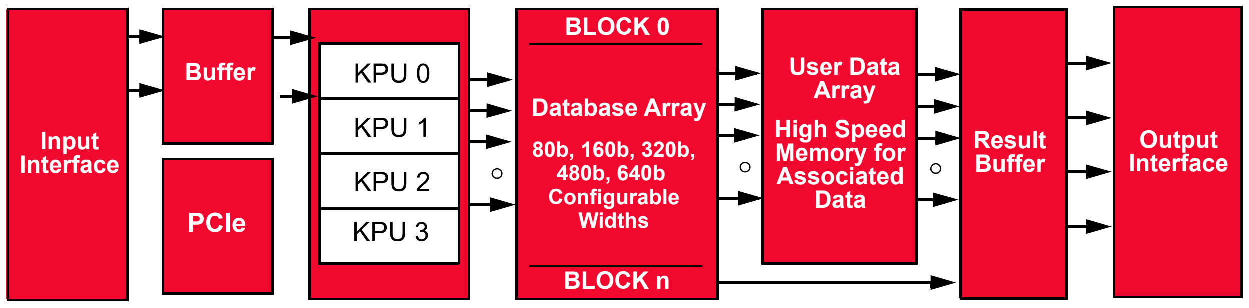


Table of Contents

Chapter 1: Overview

1.1 Purpose and Audience

The KBP performs high-speed operations on large rule databases for a wide range of telecommunications applications, including WAN, MAN, and Enterprise switches and routers. It provides network awareness and enables real-time modifications and updates to the routing configuration, making it ideal for packet classification, policy enforcement, and forwarding.

This document is intended for software, hardware, and application engineers.

1.2 Acronyms and Abbreviations

In most cases, acronyms and abbreviations are defined on first use.

Acronyms and abbreviations in this document are also defined in Appendix A: "Acronyms and Abbreviations," on page 123.

For a comprehensive list of acronyms and other terms used in Broadcom documents, go to: http://www.broadcom.com/press/glossary.php.

1.3 References

The references in this section may be used in conjunction with this document.

For Broadcom documents, replace the "xx" in the document number with the largest number available in the repository to ensure that you have the most current version of the document.

Document (or Item) Name	Number	Source
Broadcom Items		
KBP 16 nm Gen1 Registers Map	KBP_16NM_MP-PR102-R	Broadcom CSP
KBP SDK Reference Manual	KBP-SDK-SWUM1xx-R	Broadcom CSP
16 nm KBP Hardware Design Guide	KBP_16NM_DG-TIxx-R	Broadcom CSP

Chapter 2: Introduction

2.1 Overview

This family of KBPs addresses next-generation classification requirements through high-performance parallel decisions and improved entry storage capabilities. Up to eight parallel operations allow the device to reach decision speeds of multiple Billion Decisions Per Second (BDPS). Embedded Error Correction Circuitry (ECC) improves system testability and operational reliability. The key processing unit (KPU) enables efficient interface transfers with flexible search key construction.

This KBP seamlessly connects to the Jericho BCM88670, Arad Plus BCM88660, and Arad BCM88650.

2.2 Features

2.2.1 General Features

- Dual host enables two host devices to connect to one KBP.
- Connectivity provided to NPUs that can produce search keys through two independent ports.
- Device available in 2048k/1024k/512k 40b database entries.
- KBP tables width configurable as 80/160/320/480/640 bits.
- User Data Array for associated data, width configurable as 32/64/128/256 bits.
- Eight parallel compares enable up to eight results per operation.
- Simultaneous Multithreading (SMT) operation.
- Implements the NetRoute forwarding solution for Longest Prefix Match (LPM).
- NetACL solution for access control list.
- Logical Tables provide support for intelligent database management.
- Key Processing Unit (KPU) for flexible search key construction.
- Result Buffer provides programmable interconnect for flexible routing of search results.
- Range Matching for efficient storage utilization.
- Advanced Low Power Modes.
- ECC on User Data and Database Array. Parity protection on all embedded memories.
- Background ECC scan for database entries.

2.2.2 Electrical Features

- Core clock operating frequency 300–900 MHz.
- 36 Rx and Tx Serial Lanes at 10.315 Gb/s, 12.5 Gb/s, 15.0 Gb/s, 25.78125 Gb/s, 27.34375 Gb/s, and 28.125 Gb/s.
  - In single host dual port, maximum 16 lanes per port with Port 0 = Lanes[15:0] and Port 1 = Lanes[33:18].
  - In dual host quad port, maximum of 8 lanes per port; Port 0 [7:0], Port 1 [15:8], Port 2 [23:16], and Port 3 [31:24].
  - All ports must operate at same SerDes rate.
- Miscellaneous CMOS I/Os at 1.8V.
- Differential I/O 1.8V.
- Core Supply 0.850V (controlled through AVS. Please refer to the *16 nm KBP Hardware Design Guide* for details). AVS connection and operation are required for proper device operation.
- Package 35 mm x 35 mm.

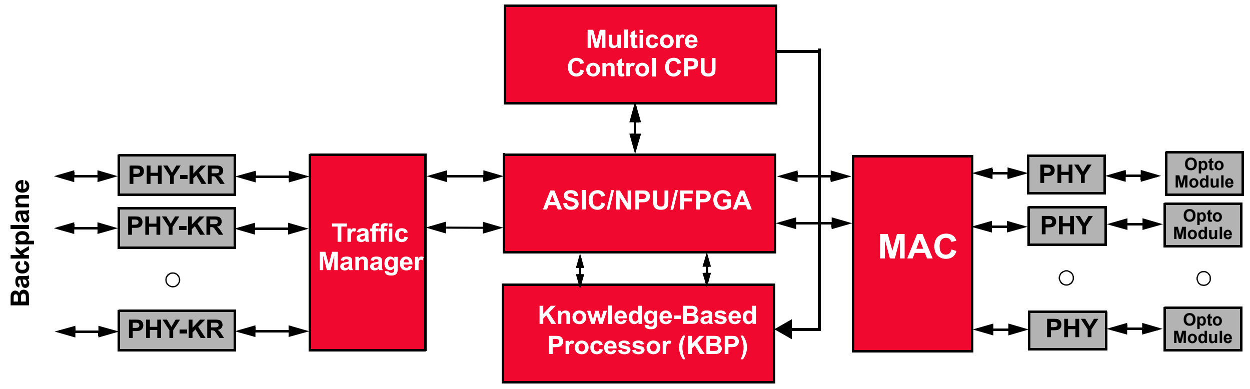
2.3 Applications

The KBP can be used in many applications, including;

- IPv4 and IPv6 Packet Classification
- Access Control Lists
- Policy-based Routing and QoS
- IPv4 and IPv6 Longest Prefix Match
- Flow-based Access Control Lists

Figure 1 shows a typical line-card application.

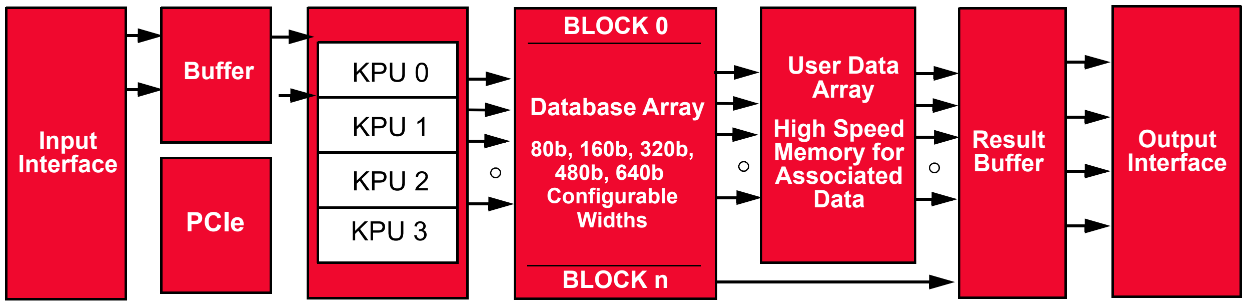
Figure 1: KBP-Based Line Card



2.4 Device Architecture Overview

Figure 2 shows a block diagram of the knowledge-based processor (KBP). The main blocks are described in the following subsections.

Figure 2: KBP Device Block Diagram



2.4.1 Key Processing Unit

The key processing unit provides user flexibility in generating search keys for the various search profiles (logical tables) through key manipulation and preprocessing. The key processing unit offers the ability for users to create unique keys for each search operation issued per a compare instruction.

Refer to Chapter 3, Key Processing Unit for more information.

2.4.2 Database Array

Each database record is composed of 80 X-Y values, one X-Y pair for each bit in the entry. When writing to the processor, users can choose to write in the traditional Data Word/Mask Word format or in the X-Y format.

Refer to Chapter 5, Database Architecture Overview for more information.

2.4.3 User Data Array

The User Data Array (UDA) is a random access memory subsystem. The UDA is a shared storage resource with a total capacity up to 512 Mb. The resource consists of 64 blocks capable of up to eight parallel accesses per clock cycle. ECC protection is provided with 1-bit correct and 2-bit detect.

See Chapter 9, User Data Array for more information.

2.4.4 Result Buffer

The highest priority match response is sent to the Output Interface for transmission. Result Buffer also allows for postprocessing of AD (truncating, shifting and padding).

2.4.5 PCIe Interface

The processor has a PCIe 2.0 port configured for one lane enabling direct connection to control plane. All device functions are supported including device initialization, PIO read/write, table updates and searches.

PCIe must be connected and is required for proper device operation.

See Chapter 11, Device Configuration for more details.

2.4.6 Search Data Flow

The processor is designed to offer maximum flexibility for the user to create and apply search keys, resolve search results from database blocks, access user data memory, and output the results.

The processor provides a buffer for loading and storing a “master search key.” The buffer can store up to 4096x640b contexts. The buffer is followed by the key processing units, which can generate up to eight unique keys, up to 640b in width, from the master search key. The KPU output can be further modified by inserting up to two range information. The range hardware is programmable. The programmable interconnect maps the keys generated by the KPUs to superblocks containing the database entries.

On the result side, a programmable interconnect is available to provide flexible routing and priority encoding of search results. All blocks directed toward the same result output (same parallel search) are used in the determination of the highest priority match (HPM) for that output. The programmable result interconnect allows for the results from the database blocks to be mapped to any one of eight result outputs. This results in a mesh structure between block results to eight output slots. These eight results can then be sent to the result buffer to be output through the interface or can be used to index the user data memory. The search results and user data can then be sent to the result buffer for output.

Chapter 3: Key Processing Unit

3.1 Overview

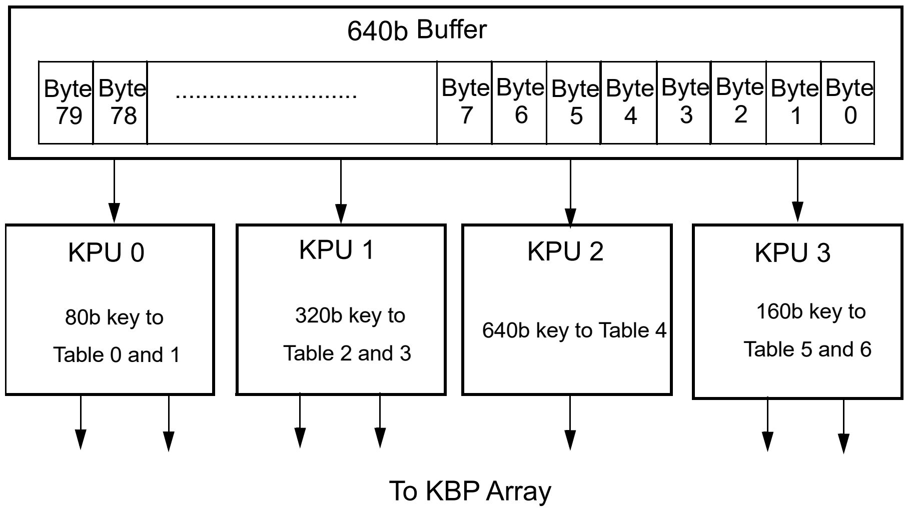
The key processing unit provides user flexibility in generating search keys for the various search profiles (logical tables) through key manipulation and preprocessing. The key processing unit offers the ability for users to create unique keys for each search operation issued per compare instruction. Through register control, the key processing unit assembles four unique keys up to 640b or four unique key pairs, with each key pair sharing, in unit of 160b, up to 640b of data from the buffer. Byte-level parsing is used across the entire 640b of data.

Up to 10 segments can be used to generate search keys for compare operations. Each KPU gets the 640b master key coming from the buffer in single thread mode, and one of the two 640b master keys coming from the buffer in SMT mode. On the master key, a given KPU can produce one key of up to 640b length and one additional key of up to 320b length. However, as stated above, the combined width of the two keys must not exceed 640b.

The KPU has a Fill-with-Zero feature, which, when set, will fill all bytes in a segment with zeros. Enabling Fill-with-Zero for all unused KPUs and KPU segments will reduce power consumption.

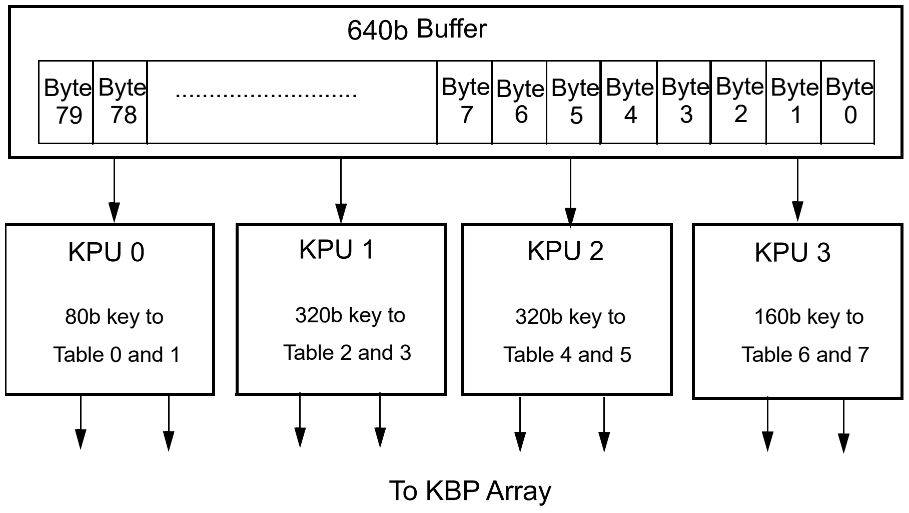
The key processing functionality of the key processing unit is preconfigured using registers, and then activated during compare operations with the referenced logical table register. Figure 3 depicts an example of each key processing unit constructing a unique search key from one distinct context.

Figure 3: Key Processing Unit Diagram—Example 1



Valid Start_Byte values range from 0 to 0x4F. If the Start_Byte fields of all segments are programmed to 0x7F then the search key data is passed through to the database for search operation. Length_Byte values range from one to 16 with 0 = 1 byte, 1 = 2 bytes, … 15 = 16 bytes. If the 10 search key segments are greater than 640b, then only the first 640b is used, and the balance of bits greater than 640b is ignored. If all 10 segments do not complete 640b in length, then the remaining balance up to 640b is copied from the master search key.

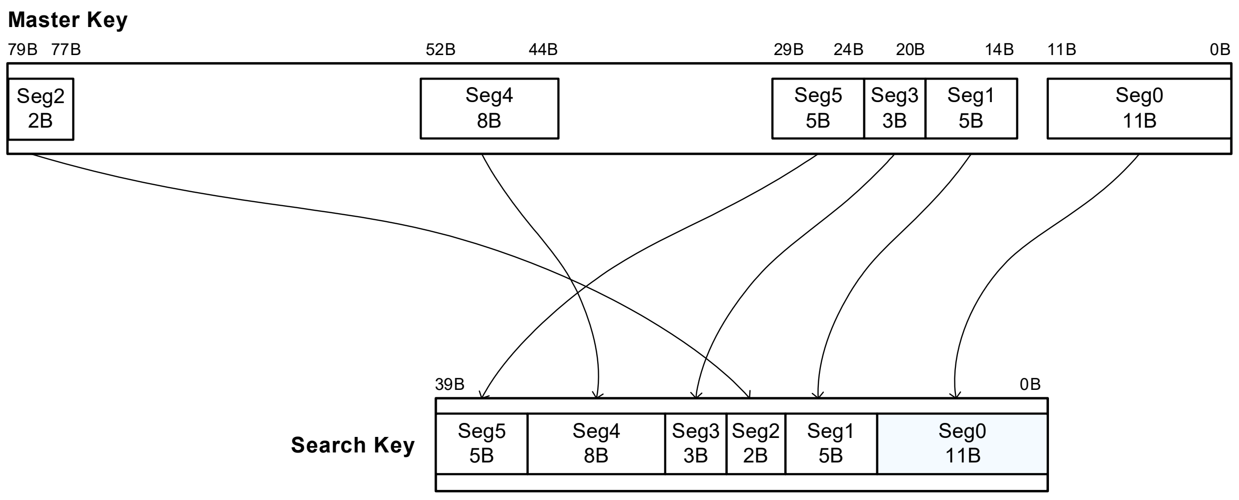
Figure 4: Key Processing Unit Diagram for Eight Search Keys



3.2 Key Processing Unit Byte Parsing

The key processing unit parses search key data from the buffer. The byte-level parsing is defined by register settings in the logical table register. The user specifies the Start_Byte and the Length_Byte of each segment. Up to 10 segments from the buffer can be concatenated to form a search key.

Figure 5: KPU Byte Parsing



3.3 Key Processing Unit Parsing Considerations

Table 1 describes the search key when less than 640b have been parsed from the 640b master search key in the key processing unit for blocks configured to 640b width. The key processing unit has extracted 440b (439:0) from the master search key. Since the key processing unit did not define the upper 200b (639:440), the remaining balance is copied from the master search key. The same parsing rules apply for blocks configured to 320b width.

Table 1: Search Key When Key Processing Unit Parses Less Than 640b

640b master search key	639:560 80b	559:480 80b	479:440 40b	439:400 40b	399:320 80b	319:240 80b	239:160 80b	159:80 80b	79:0 80b
key processing unit parsing less than 640b	Seg_0_CB[10,9], Seg_1_CB[0,9], Seg_2_CB[20,9], Seg_3_CB[40,9], Seg_4_CB [30,9], Seg_5_CB [0,0x7F], Seg_6_CB [0,0x7F], Seg_7_CB [0,0x7F], Seg_8_CB [0,0x7F], and Seg_9_CB [0,0x7F]								
Search Key	639:560 80b	559:480 80b	479:440 40b	439:400 Seg_5 to Seg_9	319:240 Seg_4	399:320 Seg_3	239:160 Seg_2	79:0 Seg_1	159:80 Seg_0

Table 2 describes the search key when more than 640b have been parsed from the 640b master search key in the key processing unit for blocks configured to 640b width. The key processing unit extracts the first 640b from the master search key and ignores any bits greater than 640b. The same parsing rules apply for blocks configured to narrower width.

Table 2: Search Key When Key Processing Unit Parses Greater Than 640b

640b master search key	639:512 128b	511:384 128b	383:256 128b	255:128 128b	127:0 128b
key processing unit parsing more than 640b	Seg_0_CB[0,15], Seg_1_CB[16,15], Seg_2_CB[32,15], Seg_3_CB[48,15], Seg_4_CB [64,15], Seg_5_CB [0,15], Seg_6_CB [16,15], Seg_7_CB [32,15], Seg_8_CB [48,15], and Seg_9_CB [0,0x7F]				
Search Key	639:512 Seg_4	511:384 Seg_3	383:256 Seg_2	255:128 Seg_1	127:0 Seg_0

3.4 Key Processing Unit with Overlapping and Duplication of Data

The key processing unit supports overlapping and duplication of data. In Table 3, master search key data 79:0 is used in segments zero and two, and master search key data 159:80 is used in segments one and three. Segment four consists of master search key data 119:40, which selects data from segments zero, one, two, and three.

Table 3: Key Processing Unit with Overlapping and Duplication of Data

640b master search key	639:560 80b	559:480 80b	479:440 40b	439:400 40b	399:320 80b	319:240 80b	239:160 80b	159:80 80b	79:0 80b

Table 3: Key Processing Unit with Overlapping and Duplication of Data

key processing unit parsing less than 640b	Seg_0_CB[10,9], Seg_1_CB[0,9], Seg_2_CB[10,9], Seg_3_CB[0,9], Seg_4_CB [5,9], Seg_5_CB [0,0x7F], Seg_6_CB [0,0x7F], Seg_7_CB [0,0x7F], Seg_8_CB [0,0x7F], and Seg_9_CB [0,0x7F]								
Search key	639:560 80b	559:480 80b	479:440 40b	439:400 Seg_5 to Seg_9	119:40 Seg_4	79:0 Seg_3	159:80 Seg_2	79:0 Seg_1	159:80 Seg_0

Chapter 4: Range Matching

4.1 Overview

Source and Destination port ranges are often used in Access Control List entries (ACLs) to better identify and process network traffic. However, when the port ranges are translated to ternary values, they often require multiple entries, leading to inefficient use of the database and a reduction in the number of user ACLs that can be stored. The worst case number of entries required to store an n-bit range with prefix encoding is (2^n– 2). This inefficiency is compounded when both source and destination port fields include ranges.

4.2 Range Matching Implementation

The processor implements range encoding by utilizing unused bits within each database entry to minimize the expansion in the number of database entries required. This effectively doubles the number of user ACEs with port ranges. The encoding algorithm in the processor is flexible and selects a combination of encoding schemes, with the control plane software selecting the optimal encoding for each range set.

In the example shown in Table 4, six entries are required to store a single ACL containing the range 1 to 14; with only the range field changing between entries. The same ACL port range 1 to 14, using range expansion requires only three entries as shown in Table 5. The same data is now contained in half the number of entries, effectively doubling the storage density, with part of the unused width being used for range encoding.

Table 4: Port Expansion Without Range Matching for Ports 1 to 14

Destination Address	Source Address	16b Port	Ports
A	B	16’b0000_0000_0000_0001	Port 1
A	B	16’b0000_0000_0000_001*	Ports 2 and 3
A	B	16’b0000_0000_0000_01**	Ports 4 to 7
A	B	16’b0000_0000_0000_10**	Ports 8 to 11
A	B	16’b0000_0000_0000_110*	Ports 12 and 13
A	B	16’b0000_0000_0000_1110	Port 14

Table 5: Port Expansion with Range Matching for Ports 1 to 14 with 24b Encoding

Destination Address	Source Address	Used by Range Matching Algorithm	Ports
A	B	31’b0000_0000_0000_0000_0000_0**1_****_****	Ports 1, 2, and 3
A	B	31’b0000_0000_0000_0000_000*_1***_****_****	Ports 4 to 11
A	B	31’b0000_0000_0000_0000_0011_10**_****_****	Ports 12 to 14

The processor has range encoding implemented in hardware. Range encoding requires both hardware and software support. The software operates both in the control plane and the data plane to implement range operations. Various internal registers are used by the software to implement the range operations.

The range encoding can be dynamically applied per LTR. Up to four range fields per search key can be independently selected for range matching, providing efficient range rule compaction. The search profile in the Logical Table register includes range matching registers. See range registers for further details on insertion locations of encoded range fields into parallel search keys and extraction locations of range fields from the master search key.

4.3 Naming Convention

Throughout this document, the following naming conventions are used for range matching.

Table 6: Naming Conventions

Name	Description
Range Encoding	A form of arithmetic coding; a data compression method
Range Field	16b of the port address requiring range encoding
Encoded Range Field	16b, 24b, or 32b output from the range encoding engine

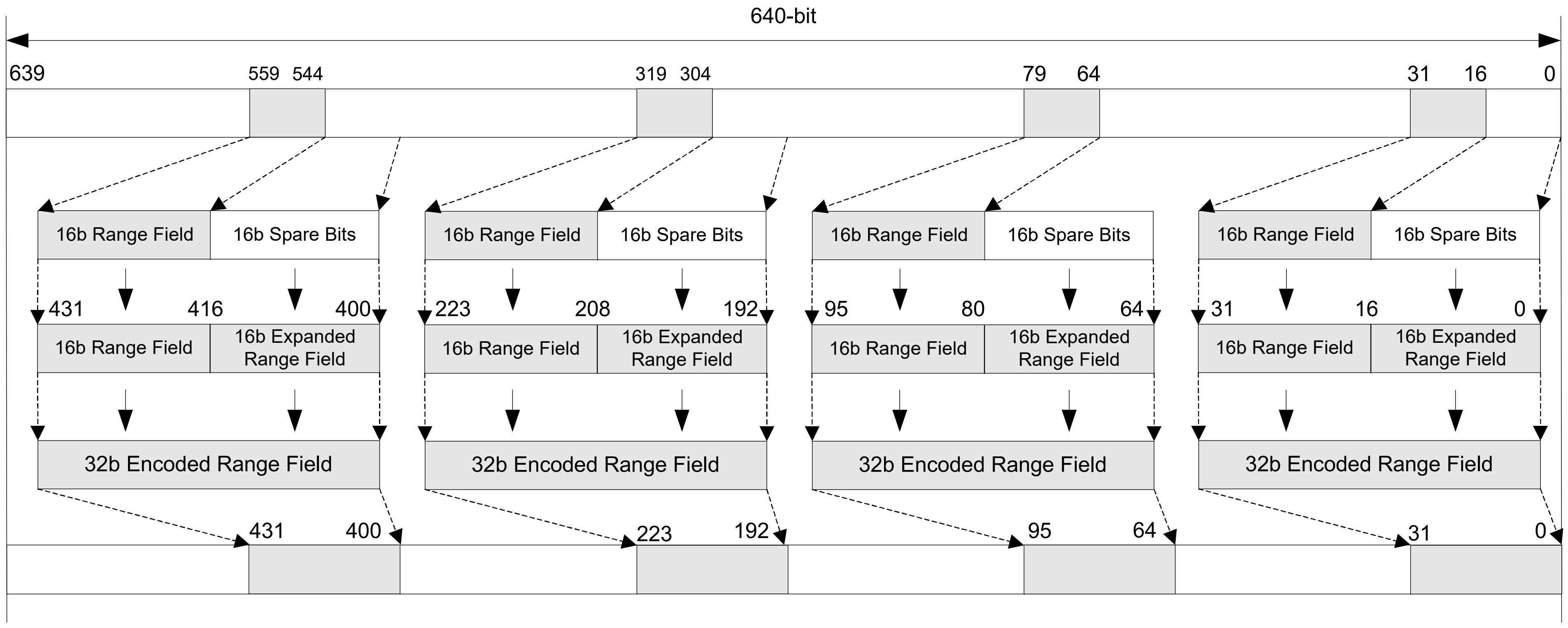
4.4 Considerations

- Range encoding is supported for up to four 16b range fields per 640b search key.
- As shown in "Encoded Range Fields" on page 18, the 16b range fields must start and end on byte boundaries. The device uses all 16-bits of the range field regardless of the number of bits used to classify the port. Potential unused bits in the range field cannot be allocated for other purposes.
- As shown in "Encoded Range Fields", encoded range fields can be inserted at any byte location within the constructed key.
- The user specifies the number of bytes of the encoded range field to insert into the constructed key. Two, three, or four byte insertions are allowed. Optimal compression is achieved when all 32b of the encoded range field is inserted. Even with two-byte insertion, the device performs some level of compression.
- The range function is implemented in parallel with the key processing unit (KPU) function. The user must ensure that there is enough room in the constructed search key to insert the encoded range field.
- There can be at maximum, two ranges (maximum of 32b each) inserted in any key (if 640b contains a key pair, then each key would be able to have two ranges).
- The location of range insertion is programmable. If there are two ranges to be inserted, the second range is inserted after the first one, and the second one occupies higher bit positions than the first one.

4.5 Encoded Range Fields

Figure 6 shows the format of the encoded range fields.

Figure 6: Encoded Range Fields



4.6 Range Encoding Data Flow

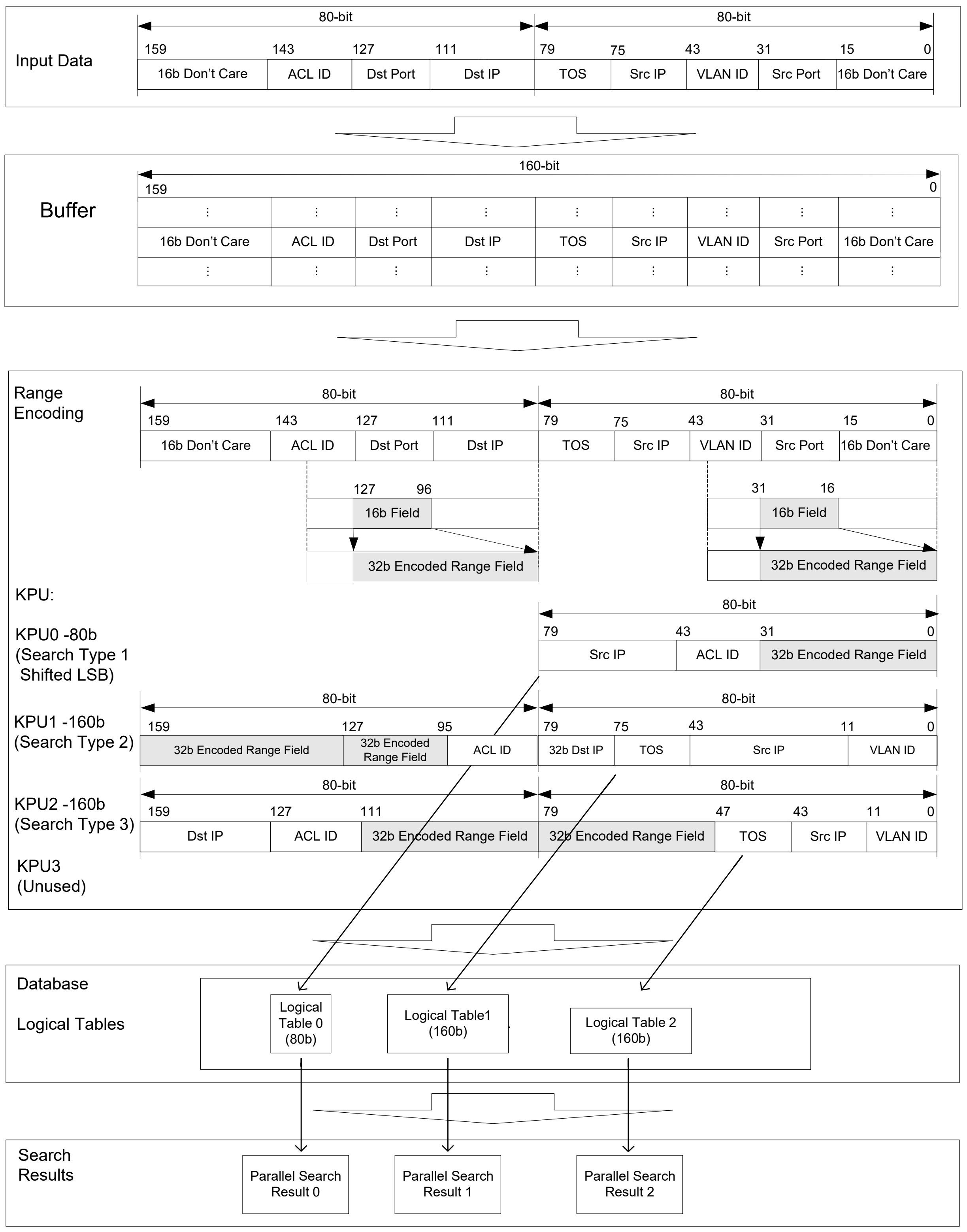
Range Encoding requires both hardware and software support. The software operates both in the control plane and the data plane to implement range operations. Various internal registers are used by the software to implement the range operations.

In the control plane, the Broadcom software provides a set of Range APIs. The range API has several function calls, including a function call that encodes the 16b Range Field into a 16b, 24b, or 32b encoded range field. The user passes the port ranges to this function, which determines the encoding for most efficient storage. The function returns the range encoded fields to the user to be included in the user's databases.

In the data plane, the user writes the input data, (that is, unencoded search key) to the buffer. The buffer transfers the data to the KPU and range function. The KPU and range function operate in parallel. The device, using the range function, encodes the selected range field(s) and inserts up to 32b of the encoded range field back into the search key.

Figure 7 on page 20 shows a typical sequence where an instruction writes 160b of input data into the buffer. An instruction such as Write buffer and Compare1 transfers 640b of data from the buffer to the KPU and range function. In this example only the least significant 160b of the transfer data is used, the most significant 160b of transfer data is not used, and the database blocks targeted by this instruction must be configured to 160b or 80b.

Figure 7: Search Data Flow with Range Encoding



Chapter 5: Database Architecture Overview

5.1 Database Entry

Each database record is composed of 80 X-Y values, one X-Y pair for each bit in the entry. When writing to the processor, users can choose to write in the traditional Data Word/Mask Word format or in the X-Y format. Internally, the processor always stores in the encoded, X-Y format. Consequently, database read operations return data in the X-Y format only (a single read operation returns an 'X' value or a 'Y' value, as dictated by the instruction encoding). Table 7 shows the two encoding formats:

Table 7: Data/Mask Encoding

Data	Mask	Data/Mask Description
0	0	Bit matches a '0' only.
0	1	Always hit–bit matches either a '0' or a '1'. Represents an 'X' (don't care).
1	0	Bit matches a '1' only.
1	1	Always hit–bit matches either a '0' or a '1'. Represents an 'X' (don't care).

Table 8: X-Y Encoding

X	Y	X-Y Description
0	0	Always hit–bit matches either a '0' or a '1'. Represents an 'X' (don't care).
0	1	Bit matches a '0' only.
1	0	Bit matches a '1' only.
1	1	Always miss–forces a mismatch for the compare, unless BMR masks off corresponding bit.
Y = D_b AND M_b		
X = D AND M_b		

ECC scan will detect corruption of up to four consecutive bits and two bits anywhere in the 80b word and correct up to 1b error at the time of the ECC scan. The ECC code is extended to 11 bits, by adding three parity bits to the 8b regular ECC code. The three parity bits are defined as follows:

- P0 (Code[8]) = XOR {D0, D3,...,D78}
- P1 (code[9]) = XOR {D1,D4,..., D79}
- P2(Code[10]) = XOR {D2, D5,..., D80}

Note that D80 is a valid bit, VBIT.

5.2 Block Description

The database is divided into a number of internal blocks, dependent on the processor capacity. In case of 80b, 160b, and 320b configurations, each array block works independently. In case of wider widths of 480b and 640b, two array blocks are paired. The valid 480b and 640b block pairs are 0 and 4, 1 and 5, 2 and 6, and 3 and 7. Each of these blocks is configurable into the following organizations: 4k x 80b, 2k x 160b, 1k x 320b, and 512 x 640b.

Two neighboring blocks in a super block can be combined to create two blocks of one of the following combinations:

- One block 1k x 480b and one block of 1k x 0b,
- One block 1k x 480b and one block 1k x 80b
- One block 1k x 480b and one block 1k x 160b
- Two blocks of 512 x 640b

Additionally, the blocks can be configured to store other widths (example: 8k x 40b) through the use of block masks.

To program a 640b entry into an Array Block (AB), the 320 LSB bits are programmed in the first AB of the block pair and then 320 MSB bits are programmed in the second AB of the block pair. Array Block numbers are generated in the final result for 640b compares. For entries 0-511, the AB number in the result is the same as the first AB in the block pair. For entries 512 to 1023, the AB number in the result is the same as the second AB in the block pair.

When database ECC scan is enabled and the block is enabled, then all database records in that block take part in the scanning process irrespective of their VBIT status, valid or empty.

Table 9: Block Organization

Block	Block Mask Register [319:0]	MSB	Block Mask Register 0			LSB
		MSB	Block Mask Register 1			LSB
		MSB	Block Mask Register 2			LSB
		MSB	Block Mask Register 3			LSB
	Database Block	12'h000	MSB	X	LSB	VBIT
			MSB	Y	LSB	
		12'h001	MSB	X	LSB	VBIT
			MSB	Y	LSB	
						⋮
		12'hFFE	MSB	X	LSB	VBIT
			MSB	Y	LSB	
		12'hFFF	MSB	X	LSB	VBIT
			MSB	Y	LSB	

Each 80-bit database record has a Validity Bit, VBIT. The status of the VBIT, valid, '1' or invalid, '0,' determines if the record takes part in compare operations. When a VBIT is set to valid '1,' the database record takes part in a compare operation. When the VBIT is cleared to '0,' that record does not take part in a compare operation.

To invalidate or delete a database entry, clear the VBIT to '0.' Only one VBIT must be cleared to '0' for 320b, 480b, and 640b records.

Table 10: Invalidate or Delete Database Record

Database Entry	A to H Represent 80-bit Segments A = [79:0], B = [159:80], C = [239:160], to H = [639:560]
80-bit Entry	Must clear VBIT of {A} to ‘0’
160-bit Entry	Must clear one location of VBIT in set {A, B} to ‘0’
320-bit Entry	Must clear one location of VBIT in set {A, B, C, D} to ‘0’
480-bit Entry	Must clear one location of VBIT in set {A, B, C,..., F} to ‘0’
640-bit Entry	Must clear one location of VBIT in set {A, B, C,..., H} to ‘0’

5.2.1 Superblock

The blocks are grouped into superblocks, each superblock is composed of a set of blocks. For each compare cycle, each superblock can offer one search key to its member blocks. All blocks in a single superblock must share the same search key on a given search operation. However, individual blocks within a superblock can have different block masks. Table 11 details the superblock assignments for a processor with 2048k 40b or 1024k 80b records.

Table 11: Superblock Assignments for 2048k 40b or 1024k 80b Records

Superblock	Blocks
0	0, 1, 2, and 3
1	4, 5, 6, and 7
:	:
62	248, 249, 250, and 251
63	252, 253, 254, and 255

5.2.2 Block and Superblock Relationship

For each device the associated blocks and superblocks are listed in Table 12.

Table 12: Device Block and Superblock Details

40b Records	Number of Blocks	Block Configuration	Number of Superblocks
2048k	256	4k x 80b, configurable	64
1024k	128	4k x 80b, configurable	32
512k	64	4k x 80b, configurable	16

5.2.2.1 Number of Records per Width

For each device, the associated number of records based on block width is listed in Table 13. The native record widths are 80b, 160b, 320b, 480b, or 640b. Other sized records may be stored in a native record width and accessed by the use of block masks. For example, two 40b records may be stored in one 80b record.

Table 13: Records Details

40-bit Records	80-bit Records	160-bit Records	320-bit Records	480-bit Records	640-bit Records
2048k	1024k	512k	256k	128k	128k
1024k	512k	256k	128k	64k	64k
512k	256k	128k	64k	32k	32k

Table 13: Records Details (Continued)

40-bit Records	80-bit Records	160-bit Records	320-bit Records	480-bit Records	640-bit Records
256k	128k	64k	32k	16k	16k

Table 14: 480b Records Details

480b	480b + 80b	480b + 160b
128k	128k/256k	128k/128k

5.3 Error Detection and Correction

This section describes the following types of error detection for the KBP device.

- Database array soft error detection and correction
- Buffer soft error detection
- User data array soft error detection and correction
- Logical table register soft error detection

5.3.1 Database Soft Error Detection and Correction

This device implements ECC protection for database soft errors. ECC is calculated and stored automatically by the processor during write operations to database locations. Optionally, GIO_L can be asserted when any database soft error is detected. Additionally, Database Soft Error bit in the Error Status Register is asserted.

Table 15 outlines all of the different combinations of Database Soft Error Scan that are possible, and how to select each configuration using Device Configuration Register (DCR) options.

Table 15: Database Scan Options

Database Scan Type	DCR[6] Soft Error Scan Enable	DCR[7] Correction Invalidation Enable	DCR[8] Auto Correction Enable
Database Scan Enabled	1	0: Detect only	0: no correction
		1: Detect and invalidate	1: correction
Database Scan Disabled	0	X	X

5.3.2 Corrective Action Example for Soft Errors

When GIO_L is asserted or an error reply is generated, the Error Status Register should be read to determine the cause of the interrupt. GIO_L is asserted for database, buffer, or algorithmic engine parity error.

Chapter 6: Low-Power Mode

The Low-Power (LP) mode can significantly reduce total active power consumption of the processor. This mode of operation, configured through the *Low-Power Mode Control Register*, is programmed on a per array column (two super-block) basis as shown in Table 16.

Table 16: Low-Power Mode Control Register

Bit Location	Name	Read/ Write	Initial Value	Description
79:32	Reserved	R	48'b0	Reserved
31:0	LPT Control per Superblock pair	R/W	32'b0	Low-Power (LP) mode control per superblock pair. Bit 0 controls LP setting for Superblock 0 and 1. Bit 1 controls LP setting for Superblock 2 and 3, and so on. 1'b0 = Low Power Mode disabled 1'b1 = Low Power Mode enabled When LP mode is enabled, Block Mask Registers must be programmed so that two consecutive bits starting on even boundaries (0:1, 2:3, and so on) have the same value.

This mode requires that the block mask registers be programmed on 2-bit boundaries (pairs of bits must be the same mask value) as shown in Table 17.

Table 17: Block Mask Configuration for Low-Power Mode

Block Mask Bits [79:78]			Block Mask Bits [5:4]		Block Mask Bits [3:2]		Block Mask Bits [1:0]	
0	0	…	1	1	1	1	0	0

6.1 Data/Mask to LP X-Y Translation During Database Write

The user has the option to write to the device database in either data/mask format or X-Y format using the WRMODE bit. When WRMODE = 0, the device interprets the data coming into the device as data/mask format. In LP mode, when data is written in data/mask format, the device internally converts data and mask bits to LP X-Y bits. Table 18 shows the conversion from data/mask-to-LP X-Y encoding performed internally by the processor.

Table 18: Data/Mask-to-LP X-Y Encoding

Data/Mask				**LP X-Y**			
D1	M1	D0	M0	LPX1	LPY1	LPX0	LPY0
0	0	0	0	1	1	0	1
0	0	1	0	0	1	1	1
1	0	0	0	1	0	1	1
1	0	1	0	1	1	1	0
0	0	X	1	0	1	0	1
1	0	X	1	1	0	1	0
X	1	0	0	1	0	0	1
X	1	1	0	0	1	1	0
X	1	X	1	0	0	0	0

The corresponding Boolean equations for translation from data/mask-to-LP X-Y encoding are given below:

- LPX1 = D1M1' + D0'M0'
- LPY1 = D1'M1' + D0M0'
- LPX0 = D1M1' + D0M0'
- LPY0 = D1'M1' + D0'M0'

6.2 X-Y-to-LP X-Y Translation During Database Write

Database entries can be written into the KBP in X-Y mode by setting the WRMODE bit to 1. When LP mode is enabled, the device converts X-Y format bits to LP X-Y format. In LP mode, when data is written in X-Y mode, the device internally converts X-Y bits to LP X-Y bits. Table 19 shows the conversion from X-Y encoding to LP X-Y encoding.

Table 19: X-Y-to-LP X-Y Encoding

X-Y				LP X-Y			
X1	Y1	X0	Y0	LPX1	LPY1	LPX0	LPY0
0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	1
0	0	1	0	0	1	1	0
0	0	1	1	1	1	1	1
0	1	0	0	0	1	0	1
0	1	0	1	1	1	0	1
0	1	1	0	0	1	1	1
0	1	1	1	1	1	1	1
1	0	0	0	1	0	1	0
1	0	0	1	1	0	1	1
1	0	1	0	1	1	1	0
1	0	1	1	1	1	1	1
1	1	0	0	1	1	1	1
1	1	0	1	1	1	1	1
1	1	1	0	1	1	1	1
1	1	1	1	1	1	1	1

The corresponding Boolean equations for translation from X-Y to LP X-Y encoding are given below:

- LPX1 = X1 + Y0
- LPY1 = X0 + Y1
- LPX0 = X1 + X0
- LPY0 = Y1 + Y0

The KBP outputs the LPX-Y format data during a read operation.

6.3 LP X-Y-to-Data/Mask Translation During Database Read

When a translation from data/mask-to-LP X-Y encoding is carried out, the original content is lost due to the encoding. This can be seen in Table 18 where some combinations of LP X-Y correspond to multiple data values. In the reverse translation, the data value is chosen to be 0 for each masked bit (Mask = 1). Using this rule, the LP X-Y-to-data/mask translation is shown in Table 20. Note that some rows are NA because these correspond to LP X-Y bit combinations that do not occur.

Table 20: LP X-Y-to-Data/Mask Format

LP X-Y				**Data/Mask**			
LPX1	LPY1	LPX0	LPY0	D1	M1	D0	M0
0	0	0	0	0	1	0	1
0	0	0	1	NA	NA	NA	NA
0	0	1	0	NA	NA	NA	NA
0	0	1	1	NA	NA	NA	NA
0	1	0	0	NA	NA	NA	NA
0	1	0	1	0	0	0	1
0	1	1	0	0	1	1	0
0	1	1	1	0	0	1	0
1	0	0	0	NA	NA	NA	NA
1	0	0	1	0	1	0	0
1	0	1	0	1	0	0	1
1	0	1	1	1	0	0	0
1	1	0	0	NA	NA	NA	NA
1	1	0	1	0	0	0	0
1	1	1	0	1	0	1	0
1	1	1	1	NA	NA	NA	NA

The corresponding Boolean equations for translation from LP X-Y to data/mask format are given below:

- D1 = (LPX1)(LPX0)
- M1 = (LPY1')(LPX0') + (LPX1')(LPY0')
- D0 = (LPY1)(LPX0)
- M0 = (LPX1')(LPX0') + (LPY1')(LPY0')

6.4 LP X-Y-to-X-Y Translation During Database Read

As indicated above, when a write to database entries is carried out with LP mode enabled, the KBP internally converts the write data into LP X-Y format before storing in the internal database. Thus the KBP outputs the LP X-Y format data during a read operation. The user must convert the LP X-Y encoded bits back to X-Y format to recreate the values that were written into the device. The table for conversion from LP X-Y to X-Y is given below. Note that some rows are NA because these correspond to LP X-Y bit combinations that do not occur.

Table 21: LP X-Y to X-Y Encoding

LP X-Y				X-Y			
LPX1	LPY1	LPX0	LPY0	X1	Y1	X0	Y0
0	0	0	0	0	0	0	0
0	0	0	1	NA	NA	NA	NA
0	0	1	0	NA	NA	NA	NA
0	0	1	1	NA	NA	NA	NA
0	1	0	0	NA	NA	NA	NA
0	1	0	1	0	1	0	0
0	1	1	0	0	0	1	0
0	1	1	1	0	1	1	0
1	0	0	0	NA	NA	NA	NA
1	0	0	1	0	0	0	1
1	0	1	0	1	0	0	0
1	0	1	1	1	0	0	1
1	1	0	0	NA	NA	NA	NA
1	1	0	1	0	1	0	1
1	1	1	0	1	0	1	0
1	1	1	1	NA	NA	NA	NA

The corresponding Boolean equations for translation from LP X-Y to X-Y format are given below:

- X1 = (LPX1)(LPX0)
- Y1 = (LPY1)(LPY0)
- X0 = (LPY1)(LPX0)
- Y0 = (LPX1)(LPY0)

6.5 Block Mask Configuration

When LP mode is enabled, the granularity of the block masks changes from 1 bit to 2 bits. The block masks must be programmed on 2-bit boundaries with each pair of adjacent bits having the same value. Table 22 illustrates this requirement.

Table 22: Example of Block Mask Setting in LP Mode

Block Mask Bits [79:78]	...	Block Mask Bits [5:4]	Block Mask Bits [3:2]	Block Mask Bits [1:0]
00	…	00	11	00

6.6 Recommended Sequence of Steps

The following details the sequence of operations that should be carried out when the user intends to perform a database write operation followed by a read that validates the write operation.

6.6.1 Using Data/Mask Writes (WRMODE = 0)

When LP mode is enabled and database entries are written in data/mask format, the KBP internally encodes the data in LP X-Y format. When database entries are read back, the encoded LP X-Y values are returned. The sequence of operations to validate a write operation is as follows:

1. Enable LP mode.
2. Write D word and M word with WRMODE = 0.
3. Read back LP X.
4. Read back LP Y.
5. CPU converts LP X-Y read bits to corresponding D_{new} and M_{new} using equations in "LP X-Y-to-Data/Mask Translation During Database Read" on page 28.
6. Check if $D = D_{new}$ and $M = M_{new}$.

6.6.2 Using X-Y Writes (WRMODE = 1)

When LP mode is enabled and database entries are written in X-Y format, the KBP internally encodes the data in LP X-Y format. When database entries are read back, the encoded LP X-Y values are returned. The sequence of operations to validate a write operation is as follows:

1. Enable LP mode.
2. Write X word and Y word with WRMODE = 1.
3. Read back LP X.
4. Read back LP Y.
5. CPU converts LP X-Y read bits to corresponding X_{new} word and Y_{new} word using equations in "LP X-Y-to-X-Y Translation During Database Read" on page 29.
6. Check if $X = X_{new}$ and $Y = Y_{new}$.

Chapter 7: Enhanced Power Control

This section highlights the Enhanced Power Control (EPC) options.

7.1 Overview

This feature limits the maximum dynamic power consumed by the device while maintaining the maximum number of database records. This is accomplished by limiting the maximum number of blocks that are simultaneously accessed during a search operation.

7.2 Usage

The Enhanced Power Control (EPC) searches are enabled in the LTR on a per-result basis. As such, it is possible to perform EPC searches in parallel with non-EPC searches. The enhanced power control level offers further power savings with one restriction, a maximum of 128 array blocks of 80b width and a maximum of 256 array blocks of 160b or greater width in the DBA can be enabled for enhanced power control compares. The SDK is required in order to use this feature.

The searches done at the regular power control level behave exactly like the non-EPC searches. There are no additional restrictions.

Compared to the previous generation of KBP devices, this generation of power control provides up to four times improvement in enhanced power control. The previous KBP device split a search into two *trees*, reducing the number of DBA blocks to be searched by almost half. This device can split a single search into eight *trees*, potentially reducing the number of DBA blocks to be searched by one eighth.

For specific details on how to enable this feature using the SDK, contact your local Broadcom representative.

Chapter 8: Simultaneous Multithreading (SMT)

8.1 Overview

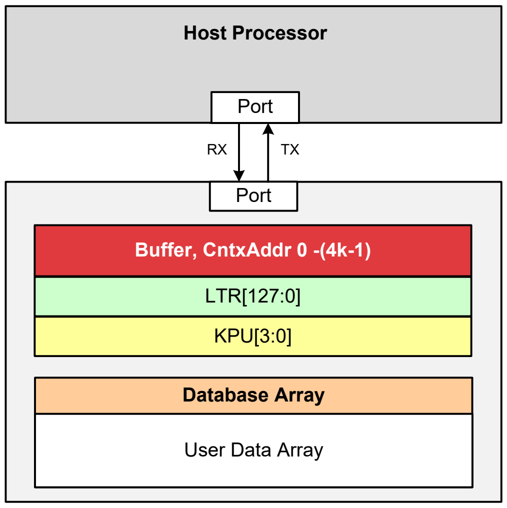
The KBP device, by default, is enabled for simultaneous multithreading (SMT) as long as compares use LTR from two groups (that is, 0–63 and 64–127). SMT mode allows for higher rates of compare instructions as compared to single-thread mode. The device, on reset, gets configured to accept SMT instructions belonging to LTR groups 0–63 and 64–127. The SDK can configure the device in other SMT ways.

This section discusses the various aspects of single-thread and multithread modes.

8.2 Single-Thread Mode

In this mode, the host controller must combine searches into a single instruction from a single master key. Parallel searches can be executed from this master key. Each search requires a unique key constructed from the master key and must reference a unique database partition. Figure 8 shows a diagram of a single-thread configuration.

Figure 8: Single-Thread Mode

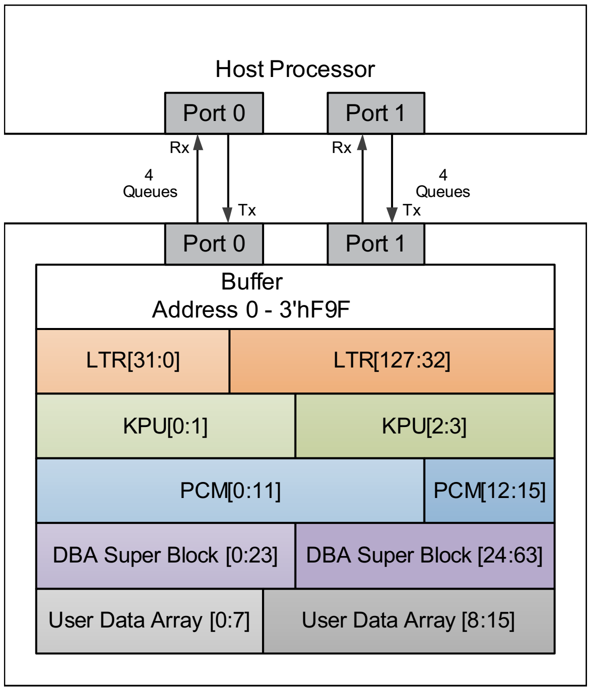


8.3 Simultaneous Multithread Mode

In SMT mode, the KBP device allows for the concurrent processing of two master keys. KBP selects the two master keys based on resource conflicts between the two selected top-of-queue instructions.

KBP implements four queues per port. These four queues are based on top two bits of LTR index [6:5]. Thus, the two instructions are selected out of four (one-port usage) or eight (two-port usage) instructions. The various resources (LTR, KPU, PCM, Database Array, and UDA) as shown in Figure 9 can be partitioned in two parts in non-symmetric ways. For example, KPUs could be divided as two KPUs per thread, PCM could be divided on 1/3 – 2/3 basis and DBA/UDA could be divided on 1/4 – 3/4 basis. The two partitions are called Banks. Thus, Thread 0 maps to Bank 0 resources and Thread 1 maps to Bank 1 resources.

Figure 9: Simultaneous Multithread Resource Example



8.3.1 Assignment of Resources in SMT Mode

In SMT mode, the KBP device resources can be assigned to one of the two threads as shown in Table 23.

Table 23: SMT Resource Assignment

Resource	Thread	Min. No.	Max. No.	Unit	Total Range
LTR	0	32	96	32	LTR [0:95]
	1	32	96	32	LTR [32:127]
KPU	0	1	3	1	KPU [0:2]
	1	1	3	1	KPU [1:3]
PCM	0	1	15	1	PCM [0:14]
	1	1	15	1	PCM [1:15]

Table 23: SMT Resource Assignment (Continued)

Resource	Thread	Min. No.	Max. No.	Unit	Total Range
DBA Superblock (SB)	0	1	62	1	SB [0:61]
	1	1	62	1	SB [2:63]
UDA	0	1	63	1	UDA [0:62]
	1	1	63	1	UDA [1:63]

8.3.2 Database Array Bank Boundaries

In SMT mode, the user sets the bank boundary by defining a set of super-blocks that can only be searched by a given thread. However, for keys that are 480b and wider, pairs of super blocks must be assigned to the same bank. Read/Write access is not restricted.

When accessing database array banks:

- Search operations in each bank are issued simultaneously.
- Search operations must not cross database array bank boundary.
- Search operations may enable any number of database array blocks as in normal operation.

8.3.3 UDA Bank Boundaries

In SMT mode, the user sets the user data array bank boundary by defining a set of User Data Memories (UDM) where each 8 Mb can only be searched by a given thread. However, for keys that are 320b and wider, pairs of UDM must be assigned to the same bank. Read/Write access is not restricted.

When accessing UDA banks:

- Search operations in each bank are issued simultaneously.
- Search operations that access UDMs must not cross bank boundary. If search key more than 320b, UDM pairs must not cross bank boundary.
- Search operations may enable any number of UDMs as in normal operation.

Chapter 9: User Data Array

9.1 Overview

The User Data Array (UDA) is a random access memory resource within the processor that can be used for associated data storage as well as other applications. The size of the UDA is device dependent and can be up to 512 Mbits (536, 870, 912 bits). Associated data storage for database search results is the application envisioned for the UDA. The UDA supports flexible access widths, concurrent accesses from multiple request engines, and single-cycle accesses up to 1024 bits. If the search results in a miss, then 0s are returned in the specified AD width. In AD-Only mode, this is the only indication that the search missed. As such, it is not recommended to have all 0s as valid associated data.

9.2 Features

The UDA provides the following features:

- Total capacity: 512 Mb/256 Mb, with option to power-off any segment of memory in unit of 8 Mb.
- UDA blocks: 256/128.
- Block width configurable: 32b, 64b, 128b, 256b.
  – Each block is independently configurable.
- Parallelism: Up to eight associated data per clock.
  – Each associated data can be a different size.
- In single-bank mode, maximum associated data is 1024b for a compare instruction.
- In dual-bank mode, maximum associated data is 1024b for a compare instruction.
- ECC protection.
- ECC scheme: 1-bit detect and correct, 2-bit detect.

Table 24: Database Array and User Data Array Capacity

Records	**Database Array (Mb)**	**User Data Array (Mb)**
2048k 40b records	80 Mb	512 Mb
1024k 40b records	40 Mb	512 Mb
512k 40b records	40 Mb	256 Mb

9.3 UDA Read/Write Accesses

Each address location in the UDA memory is accessed as a 32b value. As a result, all write and read operations are done on 32 bit locations. The read/write address range of the device is given below.

The UDA also supports 64b read/write access. However, addressing is still at 32b word level. In other words, while doing 64b read or write, the LSb of the UDA address is expected to be 0.

Table 25: UDA Address Range Based on Device Size

Device Size	Address Range for 32b Access
512 Mbit	0x0–0xFF_FFFF
256 Mbit	0x0–0x7F_FFFF

Note that the Associated Data width configuration for associated data referencing is done on a per- LTR basis and/or specified by fields embedded in the search data itself.

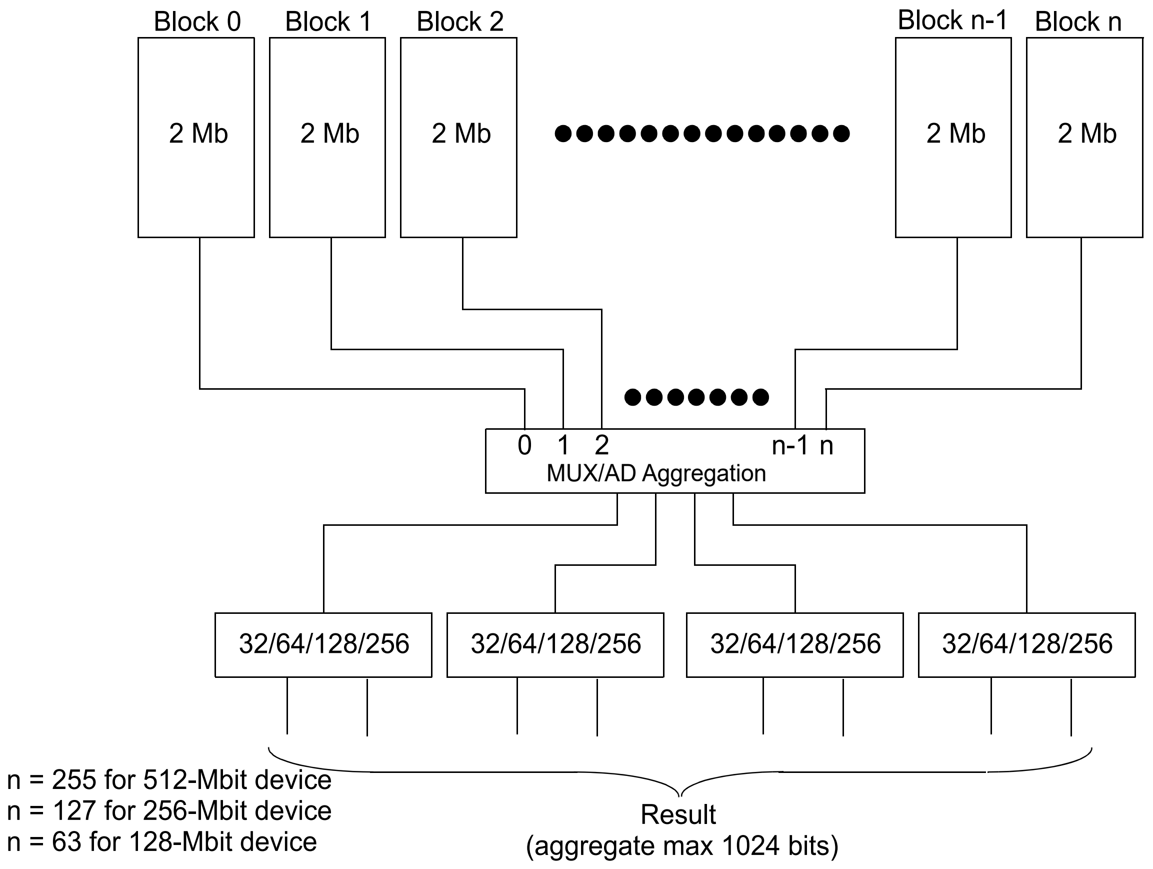
9.4 UDA Organization for Associated Data

The user data array is organized into 256 blocks (for a 512 Mb device). Each block has 2 Mb of capacity and can be configured in widths of 32, 64, 128, or 256 bits. Memory update accesses are always 32 or 64 bits. Each block is read-accessible independent of the other blocks. A total of eight accesses are allowed at the same time. Only two parallel accesses are allowed per 32 Mb and the two parallel access cannot be the same 2 Mb block. Each parallel access can have a different width, as long as the aggregate data size does not exceed 1024 bits.

For power savings, UDA blocks may be powered down in units of 16 consecutive blocks. See UDA configurations registers for further details.

If there are more than four associated data accesses, then the max AD size for result 0 and 4 each is 128b or less. The same rule applies for result pairs 1 and 5, 2 and 6, and 3 and 7.

Figure 10: UDA Organization for Associated Data Lookup



Each UDA block can be dynamically accessed in multiple bit widths ranging from 32b to 256b. This allows the user to assign associated data of various widths on a per Database Array block basis. This gives flexibility to the table management software to reassign blocks to different tables as the tables grow or shrink in real-time. As a result, logical tables may be mapped onto noncontiguous blocks within the UDA blocks.

9.5 Database to UDA Referencing

The UDA address generation is controlled by the application using registers, where the base addresses and shift parameters are configured on a per Database Array block basis.

The translation from search index results to UDA memory is done on a per Database block basis. In particular, each Database Array block has a user-configured base address (BA) as well as shift value and shift direction, as shown below. The BA represents the UDA address corresponding to the first entry in the searched Database Array block. The shift represents the binary division of the DBA address so as to normalize it into continuously increasing 32b AD words. The correspondence between database array block widths and UDA data-widths is shown below, along with the with the shift parameters. The shift index direction and shift index value must be programmed by the user.

In the examples below, the 15b base address (BA) represents the UDA address where the logical table to which the current database block belongs is referenced from. One base address is available for each Database Array block. The total number of 32b locations in a 512-Mbit UDA is 16M, which requires a 24b address. The total number of 640b address locations in a database array block is 512, thereby, requiring a 9b address. Thus, the BA bits required for this case are 15b (24b – 9b).

The address translation is done for each search result after the hit index is derived. Both the index and the derived UDA data are output to support different applications that may require either or both outputs.

9.5.1 UDA Base Address Translation

The following are the equations to derive the base address:

BA (base address) for AB (as defined in equation): (FTA >> 9) >> n, where (2^n = (ADwidth/32) (1)

BA for the next AB: (ADwidthPrevBlock/ADwidthNextBlock)* (BA for previous AB+ (640/ABwidth)) (2)

FTA (final translated address)—physical UDA address derived from the DBA index.

A representation of the base address, in terms of FTA is given below:

Base address (BA) calculation:

FTA = ({BA, 9’b0} + ShiftIndex) << 0, if AD = 32b
FTA = ({BA, 9’b0} + ShiftIndex) << 1, if AD = 64b
FTA = ({BA, 9’b0} + ShiftIndex) << 2, if AD = 128b
FTA = ({BA, 9’b0} + ShiftIndex) << 3, if AD = 256b

The following subsection provides an example of the base address calculation.

Two database array blocks are configured to be 320b width. Assume these blocks are called 0x15 and 0x16. The database address (in unit of 80b words) range of the entries in these blocks is 0x15000 to 0x16FFF. The total number of database entries in the database for these two blocks is 2K 320 bit entries. Thus, 2K entries are required in the UDA to correspond to the 2K database entries. Assume the associated data for these database entries is 64b. The translated address per block is generated by the following:

Since the DBA width is 320, there are 1024 entries per DBA block.

In this example, we are considering that the 1st DBA entry corresponds to UDM8 (memory is at 64 Mb), this means that the address of the UDA entry in unit of 32b words is 0x200000. The table represents the address of the UDA entry in terms of UDA size.

From Table 26 on page 40, FTA = 0x200000, FTA[21] = 1b1, BA[11] = 1b1 or BA = 0x800.

- Pre-shift Index of the first entry in 320b database block 0x15 = 0x000.
- Pre-shift Index of the last entry in 320b database block 0x15 = 0xFFC.
- Shifted Index for the first entry in database block 0x15 = 0x000.
- Shifted Index for the last entry in database block 0x15 = 0x3FF (shifted right by 2, from table) – DB[11:2].
- FTA, start = 0x200000 in terms of AD size.
- FTA, end = 0x2007FE in terms of AD size.

For the next DB, since contiguous addressing is considered, the base address is 0x802.

Use equation 2—(64/64) × (0x800 + 640/320) = 0x802 or incrementing BA[1]/FTA[11] derives the same result:

- Pre-shift Index for the first entry in 320b database block 0x16 = 0x000.
- Pre-shift Index for the last entry in 320b database block 0x16 = 0xFFC.

- Shifted Index for the first entry in database block 0x16 = 0x000.
- Shifted Index for the last entry in database block 0x16 = 0x3FF (shifted right by 2, from table) – DB[11:2].
- FTA, start = 0x200800 in terms of AD size.
- FTA, end = 0x200FFE in terms of AD size.

Table 26: Database Address to Final Translation Address (FTA)

AB Width	AD Size	AB shift dir	FTA23	FTA22	FTA21	FTA20	FTA19	FTA18	FTA17	FTA16	FTA15	FTA14	FTA13	FTA12	FTA11	FTA10	FTA9	FTA8	FTA7	FTA6	FTA5	FTA4	FTA3	FTA2	FTA1	FTA0
640	32	R3	BA14	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3
320	32	R2	BA14	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2
160	32	R1	BA14	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1
80	32	R0	BA14	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
640	64	R3	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	1'b0
320	64	R2	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	1'b0
160	64	R1	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	1'b0
80	64	R0	BA13	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	1'b0
640	128	R3	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	1'b0	1'b0
320	128	R2	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	1'b0	1'b0
160	128	R1	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	1'b0	1'b0
80	128	R0	BA12	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	1'b0	1'b0
640	256	R3	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	BA0	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	1'b0	1'b0	1'b0
320	256	R2	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	BA1	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	1'b0	1'b0	1'b0
160	256	R1	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	BA2	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	1'b0	1'b0	1'b0
80	256	R0	BA11	BA10	BA9	BA8	BA7	BA6	BA5	BA4	BA3	DB11	DB10	DB9	DB8	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	1'b0	1'b0	1'b0

9.5.2 Associated Data Response Formats

Associated Data is returned in one of two formats. In default operation, the Index and the Associated Data are returned for each search. If the search result is a miss, then zeroes are returned in the specified AD width instead.

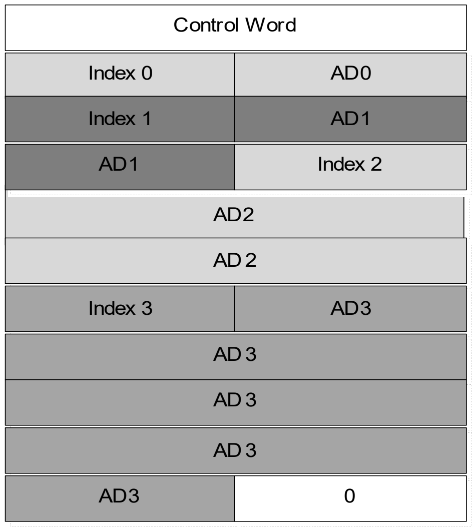
Alternatively, the device can be placed in ADorIndex mode by setting bits 47:40 of AD (Associated Data) Control Logical Table Register. Bit 40 works for Result0 and Bit 47 works for Result7. If the bit is set to 1’b1, either index or AD but not both are sent as part of result. In other words, if AD is enabled for a result, index (and match flags) will not be sent as part of that result but if AD is not enabled, index (and match flags) would be sent. Also, if only AD is being returned and result is a miss, then zeroes in the specified AD width are retuned instead. In this case, all zero should not be a valid AD value since all 0 would be indication of a miss.

9.5.2.1 Example 1a

This scenario has the following parameters:

- HPM index + associated data
- Four results:
  - Result 0 has 32b AD
  - Result 1 has 64b AD
  - Result 2 has 128b AD
  - Result 3 has 256b AD

Figure 11: Example 1a

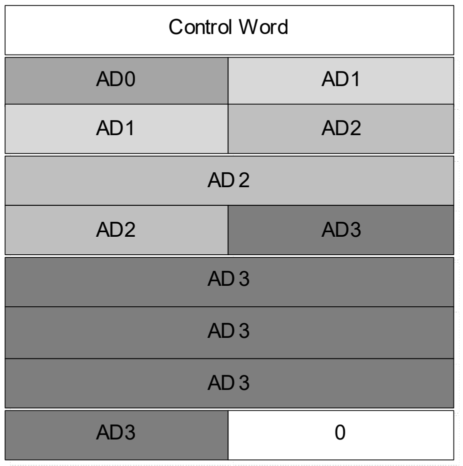


9.5.2.2 Example 1b

This is the same as Example 1a with the exception that all the four results are programmed in ADorIndex mode.

- Associated data only
- Four results:
  - Result 0 has 32b AD
  - Result 1 has 64b AD
  - Result 2 has 128b AD
  - Result 3 has 256b AD

Figure 12: Example 1b

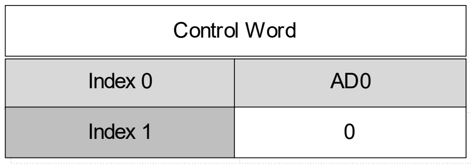


9.5.2.3 Example 2a

This example has the following parameters:

- HPM index + associated data
- Two results
  - Result 0 has 32b AD
  - Result 1 does not have AD

Figure 13: Example 2a

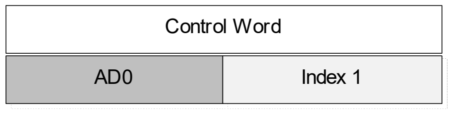


9.5.3 Example 2b

This is the same as Example 2a with the exception that both results are programmed in ADorIndex mode.

- HPM index or associated data
- Two results
  - Result 0 has 32b AD
  - Result 1 does not have AD

Figure 14: Example 2b



NOTE: If Burst Short is 16 bytes, then an extra data word is included in the response packet.

Chapter 10: Pin Descriptions

10.1 Overview

Providing power to the ESD diode can potentially damage the chip. To prevent turning on the ESD diode, do not apply voltage to all signal pins before the power supplies have ramped.

The following tables list the processor pins and their definitions.

- Table 27, Clock, Configuration, and Reset Pin Descriptions
- Table 28, Interface Transmit, Receive, Clock, and Control Pin Descriptions
- Table 29, JTAG Pin Descriptions
- Table 30, Miscellaneous Pin Descriptions
- Table 31, Test, Power, and Ground Pin Descriptions

For all static low and high inputs, an external 4.7 kΩ pull-up or pull-down resistor is recommended. Do not directly connect any inputs to ground or VDDQ.

10.2 Pin Descriptions

Table 27: Clock, Configuration, and Reset Pin Descriptions

Parameter	Symbol	Type	No. Pins	Description
Core PLL Select	CPSEL[4:0]	I	5	Determines core frequency operating mode of PLL. These are static signals, which are used to select the core operating frequency mode for the on-chip PLL. This field is encoded as follows: CPSEL[4:0] = 5'b00000 Reserved CPSEL[4:0] = 5'b00XXX Reserved CPSEL[4:0] = 5'b01XXX Reserved CPSEL[4:0] = 5'b10000 Reference = 156.25 CCLK=898.4 MHz CPSEL[4:0] = 5'b10001 Reference = 156.25 CCLK=833.3 MHz CPSEL[4:0] = 5'b10010 Reference = 156.25 CCLK=718.8 MHz CPSEL[4:0] = 5'b10011 Reference = 156.25 CCLK=600.6 MHz CPSEL[4:0] = 5'b10100 Reference = 156.25 CCLK=500.0 MHz CPSEL[4:0] = 5'b10101 Reference = 156.25 CCLK=400.4 MHz CPSEL[4:0] = 5'b10110 Reference = 156.25 CCLK=302.7 MHz CPSEL[4:0] = 5'b10111 Reference = 156.25 CCLK=249.0 MHz CPSEL[4:0] = 5'b11000 Reference = 100 CCLK=900 MHz CPSEL[4:0] = 5'b11001 Reference = 100 CCLK=833.3 MHz CPSEL[4:0] = 5'b11010 Reference = 100 CCLK=720.0 MHz CPSEL[4:0] = 5'b11011 Reference = 100 CCLK=600.0 MHz CPSEL[4:0] = 5'b11100 Reference = 100 CCLK=500.0 MHz CPSEL[4:0] = 5'b11101 Reference = 100 CCLK=400.0 MHz CPSEL[4:0] = 5'b11110 Reference = 100 CCLK=300.0 MHz CPSEL[4:0] = 5'b11111 Reference = 100 CCLK=250.0 MHz For frequencies (example: 750 MHz) not listed above, consult the KBP SDK for programming values. Connection to these input pins is required. These pins do not have On Die Termination (ODT).
Core Logic Reset	CRST_L	I	1	CRST_L is an active low asynchronous input that resets the core logic of the processor. The reset operation initializes the control logic. Connection to this input pin is required. This pin does not have ODT.
Management Data IO Clock	MDC	I	1	The Management Data IO clock input is the clock for the MDIO port, compatible with IEEE 802.3ae, only Clause 45, not Clause 22. Connection to this input pin is required. This pin does not have ODT.
Management Data Input Output	MDIO	IO	1	The Management Data Input Output pin is compatible with IEEE 802.3ae, Clause 45. This pin is open drain. Externally pull-up this pin to 1.8V via a 1k Ω resistor. Connection to this input/output pin is required. This pin does not have ODT.
Management Port ID	MPID[4:0]	I	5	The Management Port ID input pins set the management port address for the device. The Management Port ID must match the 5-bit management port address to select this device. Connection to these input pins is required. These pins do not have ODT.

Table 27: Clock, Configuration, and Reset Pin Descriptions (Continued)

Parameter	Symbol	Type	No. Pins	Description
PCIe Reset	PERST_L	I	1	The PCIe Reset is an asynchronous active low input that provides the hardware reset used to initialize the PCIe port. Connection to this input pin is required. This pin does not have ODT. The PCIe interface must be connected and is required for proper device operation.
I2C Data Line	SDA	Open Drain	1	Unused I^{2}C interface pin. One external 1 kΩ resistor to VDD18 is required for this signal.
I2C Clock Line	SCL	I	1	Unused I^{2}C interface pin. One external 1 kΩ resistor to VDD18 is required for this signal.
System Power On Reset	SRST_L	I	1	The System Power on Reset is an asynchronous active low input that provides the hardware reset used to initialize the processor. Connection to this input pin is required. This pin does not have ODT.

Table 28: Interface Transmit, Receive, Clock, and Control Pin Descriptions

Parameter	Symbol	Type	No. Pins	Description
SerDes Reference Clock	SREFCLKP SREFCLKN	I	2	The SerDes Reference Clock differential pair is used to drive the Rx and Tx serial lanes. SREFCLKP and SREFCLKN are typically driven by a clock generator. These pins contain 100Ω differential on-die termination to generate its own bias voltage. These pins require AC coupling. Connection to these input pins is required.
PCIe Reference Clock	PCREFCLKP PCREFCLKN	I	2	The PCIe Reference Clock operates at 100 MHz and are typically driven by a clock generator. These pins contain 100Ω differential on-die termination to generate its own bias voltage. If PCIe is being used then connection to these input pins is required. The PCIe interface must be connected and is required for proper device operation.
Core Reference Clock	CREFCLKP CREFCLKN	I	2	The Core Reference Clock differential pair is used as a reference for the database core. CREFCLKP and CREFCLKN are typically driven by a clock generator. These pins contain 100Ω differential on-die termination to generate its own bias voltage. These pins require AC coupling. Connection to these input pins is required.
PCIe Receive Data	PCRxDATAP, PCRxDATAN	I/O	2	P/N differential pairs for the PCIe interface. The PCIe interface must be connected and is required for proper device operation. Each receiver pair has a 100Ω differential ODT.
PCIe Transmit Data	PCTxDATAP, PCTxDATAN	I/O	2	P/N differential pairs for the PCIe interface. The PCIe interface must be connected and is required for proper device operation.
Receive Data	RxDATA_P [35:0] RxDATA_N [35:0]	I	72	Primary Receive Data are P/N differential pairs for the input serial receivers. Unused signal pairs must be left unconnected and configured to be turned off. RxDATA_P and RxDATA_N must be connected to the host processor's transmitters. Each receiver pair has a 100Ω differential ODT.

Table 28: Interface Transmit, Receive, Clock, and Control Pin Descriptions (Continued)

Parameter	Symbol	Type	No. Pins	Description
Transmit Data	TxDATA_P [35:0] TxDATA_N [35:0]	O	72	Primary Transmit Data are P/N differential pairs for the output serial transmitters. Unused signal pairs must be left unconnected and configured to be turned off. TxDATA_P and TxDATA_N must be connected to the host processor's receivers.

Table 29: JTAG Pin Descriptions

Parameter	Symbol	Type	No. Pins	Description
Test Clock Input	TCK	I	1	The JTAG test clock TCK pin is input only and provides the clock for all JTAG operations. The TCK pin is independent of the processor and interface clocks. TCK has no internal bias; when not active this pin must be externally biased to VDDQ or VSS to prevent floating inputs. If JTAG is not active, provide bias to VDDQ or ground through a resistor. This pin does not have ODT.
Test Data Input	TDI	I	1	The Test Data Input TDI signal is input only and is sampled on the rising edge of the Test Clock. Data and test instructions are received serially by the test logic through TDI. Leave unconnected if not used. Contains a weak pull-up. This pin does not have ODT.
Test Data Output	TDO	O	1	The Test Data Output TDO signal is output only and available on the falling edge of the Test Clock. The Test Data Output driver always drives even when not passing test data.
Test Mode Select	TMS	I	1	The Test Mode Select TMS signal is input only and sampled on the rising edge of the Test Clock. The Test Mode Select is the command input for the TAP controller. Leave unconnected if not used. Contains a weak pull-up. This pin does not have ODT.
Test Reset	TRST_L	I	1	The Test Reset Input TRST_L is an active low reset which provides for asynchronous initialization and drives the TAP controller into the Test-Logic-Reset state. TRST_L must be held low during normal operation. This signal contains a weak internal pull-up. Connection to this input pin is required. This pin does not have ODT.

Table 30: Miscellaneous Pin Descriptions

Parameter	Symbol	Type	No. Pins	Description
General Interrupt Output	GIO_L[1:0]	O	2	General Interrupt Outputs are asynchronous, active-low, open-drain outputs that flags the occurrence of error conditions, which includes database parity errors. These signals should be routed back to the host device as an interrupt signal. GIO_L[0] is asserted when Packet Errors, Interface Errors or Device Errors are detected on host 0. When GIO_L[0] is asserted, the host device should read the Error Status Register of host 0 to determine the cause of the error. GIO_L[0] remains asserted until the corresponding error bit is cleared in the Error Status Register. GIO_L[1] is asserted when Packet Errors, Interface Errors or Device Errors are detected on host 1. When GIO_L[1] is asserted, the host device should read the Error Status Register of host 1 to determine to cause of the error. GIO_L[1] remains asserted until the corresponding error bit is cleared in the Error Status Register.

Table 30: Miscellaneous Pin Descriptions (Continued)

Parameter	Symbol	Type	No. Pins	Description
Mode Select	MSEL[1:0]	I	2	Mode Select are static signals which are used to select device operation modes: MSEL[1:0] = 2’b00 = Standard mode of operation MSEL[1:0] = 2’b01 = Reserved MSEL[1:0] = 2’b10 = Reserved MSEL[1:0] = 2’b11 = Reserved These pins must be pulled low with a 4.7 kΩ resistor to VSS on the board. Connection to these input pins is required. These pins do not have ODT.
Resistor Calibration 0	RESCAL0	I	1	External Bias Resistor 0. One external 4.53 kΩ resistor must be connected between this pin and VSS.
Resistor Calibration 1	RESCAL1	I	1	External Bias Resistor 1. One external 4.53 kΩ resistor must be connected between this pin and VSS.
AVS Output Voltage	AVSA	Sig or other	1	Automatic Voltage Scaling A Pin. AVS connection and operation are required for proper device operation. Pin is used for automatic voltage scaling. See the Board Design Guidelines for proper pin connection.

Table 31: Test, Power, and Ground Pin Descriptions

Parameter	Symbol	Type	No. Pins	Description
Do Not Connect	DNC	I/O	94	Do not connect.
Do Not Populate	DNP	–	4	Do not populate.
Core Power	VDD	PWR	212	Programmable 0.850V nominal core voltage supply.
IO Power	VDD18	PWR	14	1.8V I/O voltage supply.
Memory Power	VDDM	PWR	23	0.85V Memory voltage supply.
SerDes PLL Power	VDDSPLL1 to VDDSPLL9	PWR	10	0.8V SerDes PLL Power1 to 9. Connect to PLL power supply filter
PCIe PLL Power	VDDPCPLL	PWR	1	0.8V PCIe PLL supply. Connect to PLL power supply filter.
Core PLL Power	VDDCPLL	PWR	1	1.8V Core PLL supply. Connect to PLL power supply filter.
AVS Output Voltage	AVSD	PWR	1	Analog Voltage Scaling D Pin. AVS connection and operation are required for proper device operation. Pin is used for automatic voltage scaling. See the Board Design Guidelines for proper pin connection.
Voltage Sense Negative	VSENSEN	GND	1	Negative Voltage Sense. This pin is shorted to the on-die VSS plane. It is to be used for regulator sensing and cannot carry large current. If unused, this pin must be shorted to VSS.
Voltage Sense Positive	VSENSEP	PWR	1	Positive Voltage Sense. This pin is shorted to the on-die VDD plane. It is to be used for regulator sensing and cannot carry large current. If unused, this pin must be shorted to VDD.
Core Ground	VSS	GND	534	Core Ground.

Table 31: Test, Power, and Ground Pin Descriptions (Continued)

Parameter	Symbol	Type	No. Pins	Description
Core PLL Ground	VSSCPLL	GND	1	Connect to PLL power supply filter GND for VDDCPLL.
PCIe Power	VDDPC	PWR	2	0.8V PCIe power supply.
SerDes Power	VDDS1 to 9	PWR	56	0.8V SerDes power supply.
SerDes TX Termination Power	VDD12	PWR	18	1.2V SerDes transmit termination power supply.

Chapter 11: Device Configuration

11.1 Overview

The KBP device uses the MDIO interface for communication with the host via a 2-wire MDIO port. This section provides a brief overview the MDIO physical interface and also provides configuration information, registers definitions, and register programming.

11.2 MDIO Physical Interface

The Management Data Input Output (MDIO) port is used to configure the KBP device, to exercise certain test modes, and to check the status of the device. This interface is consistent with IEEE 802.3ae Clause 45. For additional information on the MDIO interface, refer to IEEE 802.3ae, Clause 45.

The MDIO interface has two signals: Management Data Clock (MDC) and Management Data Input/Output (MDIO). MDIO has specific terminology to define various devices on the bus. The device driving the MDIO bus is identified as the Station Management Entity (STA). The target devices that are managed by the MDC are referred to as MDIO Manageable Devices (MMD).

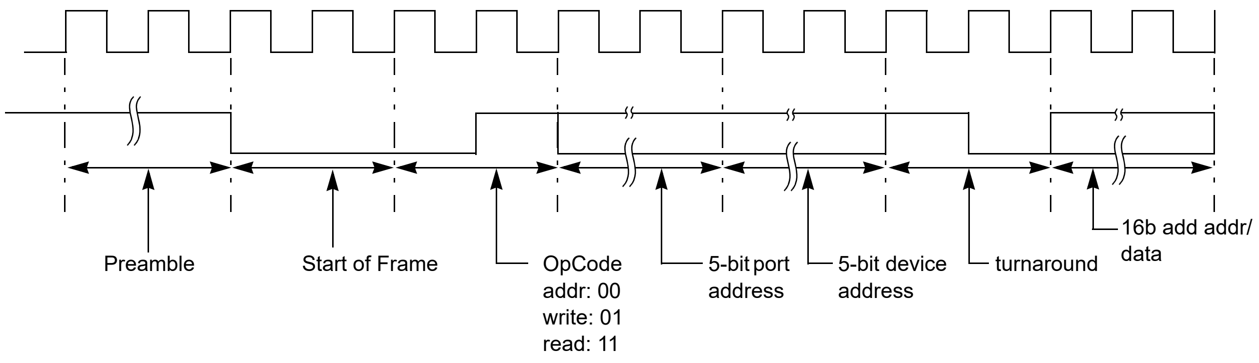
The STA initiates all communication in MDIO and is responsible for driving the clock on MDC. MDC is specified to have a frequency of up to 2.5 MHz.

Three transactions are supported:

- Address
- Write Data
- Read Data

Figure 15 shows a representative MDIO transaction. Each transaction is 64 MDC clock cycles long and contains 32 cycles preamble, 2 cycles start-of-frame, 2 cycles OpCode, 5 cycles port address, 5 cycles device address, 2 cycles turnaround, and 16 cycles of address/data.

Figure 15: MDIO Transaction Format



11.2.1 MDIO Address Transaction

During every address transaction the register address is stored along with OpCode, port address, and device address. The MDIO address is set equal to the value on the incoming MDIO pin. The device supports MDIO device addresses 1–11, 16, 17, 18, and 20.

11.2.2 MDIO Write Transaction

During a write MDIO transaction, as soon as the write data is received and stored into the 16 bits of the status register, a one-shot transaction follows where the data is written to the address specified in the address register.

11.2.3 MDIO Read Transaction

During a read MDIO transaction, after the port address is received, the MDIO logic does a one-shot transaction where it reads the data from the location specified in the address register. This falls into the Hi-Z part of the turnaround cycles. After the turnaround is complete, the MDIO logic drives the MDIO bus with data from the status register.

Chapter 12: PCIe Interface

12.1 Introduction

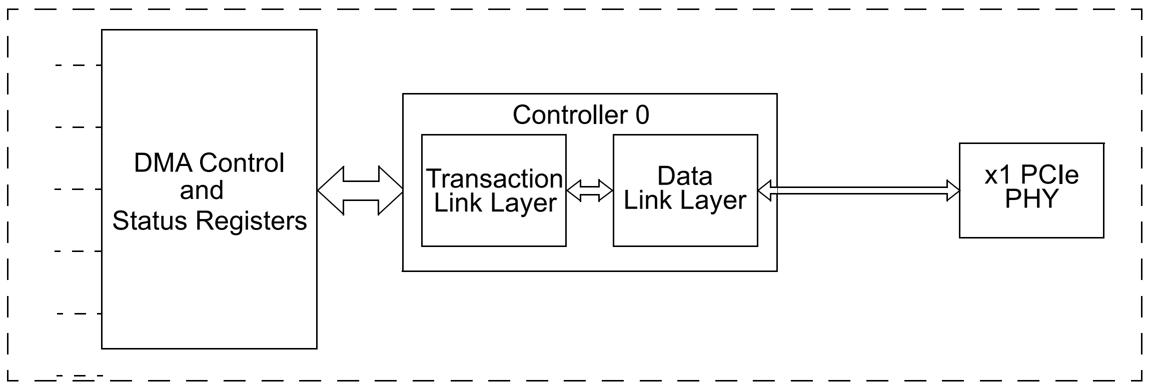
This section describes the functional description, relevant interface capabilities, and programming requirements for the PCIe controller implementation, which uses one controller to support one individual PCI Express (PCIe) lane that can be organized into x1 links.

The PCIe interface must be connected and is required for proper device operation.

12.2 PCIe Features

- Compliant to PCI Express Base Specification revision 2.0
- 5 Gigabits/second/direction of raw bandwidth
- Type-0 configuration space in Endpoint (EP) mode
- Maximum payload size of 128 bytes
- Maximum request size of 512B
- Polarity inversion on receive
- Supports MSI

Figure 16: PCIe Topology Block Diagram for a 1x1 PCIe Configuration



12.3 Theory of Operation

The PCIe interface connects internally through the DMA controller(s).

The primary transaction type is a DMA read operation from the Host system memory initiated by the KBP as a master and a DMA write operation to the Host system memory. Individual writes and reads to and from all KBP registers and memories are also supported.

The PCIe external side interface complies with PCI Local Bus Specification Revision 2.2 for PCI-compatible operation, and PCI Express 2.0 Base specification with regard to signals and behavior.

The PCIe interface must be connected and is required for proper device operation.

12.3.1 PCIe Configuration Space Access

The PCIe Controller can be set up to use the original configuration space or the extended configuration space.

Upon access to this memory address space, a 40-bit address (translated from 64 bits by the MMU) is presented to the PCIe controller, which in turn is translated to the PCIe configuration address as 39:24 Base, 23:16 Bus Number, 15:11 Device Number, 10:8 Function Number, and 7:0 Register Number.

Extended register configuration space access is as below:

39:28	27:20	19:15	14:12	11:2
Base	Bus Number	Device Number	Function Number	Register Number

Providing the Bus Number and Device Number that identifies the PCIe Controller's own configuration space results in an internal configuration register access. The required read or write command effectively reads or writes the on-chip configuration registers.

12.3.2 DMA Descriptor Structures

The Request DMA Descriptors are set-up by the Host-resident software in a system memory request queue.

These descriptors are up to 512B in length and 64b granular. A descriptor is comprised of a 64b header followed by payload.

The header contains the same user-data fields found in the ILA Control Word as well as a few control bits and the payload length.

The KBP's Request DMA Controller pulls out (Master DMA over the PCIe interface) each Descriptor on a demand basis and transfers it to the targeted subsystem for execution.

Similarly, responses are derived by the KBP subsystem that executed the request instructions within the Request Descriptors and are handed over to the Response DMA Controller.

The Response DMA Controller formats the Response Descriptors of the same structure as the Request Descriptors described above. It then places each Response Descriptor into the system memory response queue by Master DMA'ing over the PCIe interface.

These request and response transactions over the PCIe interface occur simultaneously for different instructions. This in effect allows the pipelining of the request-execution-response phases. The DMA Controllers have their own FIFO's in each direction in order to decouple the rate differences in the three phases.

12.4 PCIe Interrupts

The PCIe controller reports interrupts through the "Virtual Wire" concept of the PCIe, that is - in-band messages carry Interrupt information.

12.4.1 Interrupt Setup

The PCIe Controller detects multiple sources of interrupts and reports them to the Programmable Interrupt Controller (PIC).

The MSI interrupt supports 1 entry.

12.4.2 Configuring PCIe

The following must be configured and setup for startup and initialization:

- PCI Configuration Header registers as required.
- PCIe Interface Capability Registers as required.
- PCIe Interface's Extended Capability Registers as required.

12.5 Addressing Through PCIe

PCIe access to the device resources is through BAR0 (Base Address Register 0) in the PCIe configuration space.

There are two methods, one for DMA and another for non-DMA accesses as follows:

- For DMA accesses, whereby the device acts as a Master, the application running on the Host (Root Complex in the PCIe hierarchy) sets up the device's DMA Controller registers at:

  BAR0 + [000h through 3ffh]

- For non-DMA accesses, whereby the device is a slave to the Host, only atomic (as opposed to longer bursts in the DMA method) register and memory reads and writes are allowed at:

  BAR0 + [400h] for Request writes

  BAR0 + [500h] for Response writes

Within the non-DMA accesses, there are two address spaces as follows:

- For Satbus transactions over the internal Satellite Bus, which provides accesses to all device registers and memories over an internal (21b) address and (16b) data bus. Each Satbus access requires a write to the Satbus address register at [BAR0 + 600h] then followed by a write or read to the Satbus data register at [BAR0 + 608h].
- For PIO transactions, all of which are 64b reads and writes targeting any register and memory in the device.

These transactions address the DMA Controller's Request FIFO (for writes) to the address [BAR0 + 400h] and Response FIFO (for reads) from address [BAR0 + 500h]. The individual register or memory device addresses targeted by these transactions are embedded within the payload of the PCIe packet.

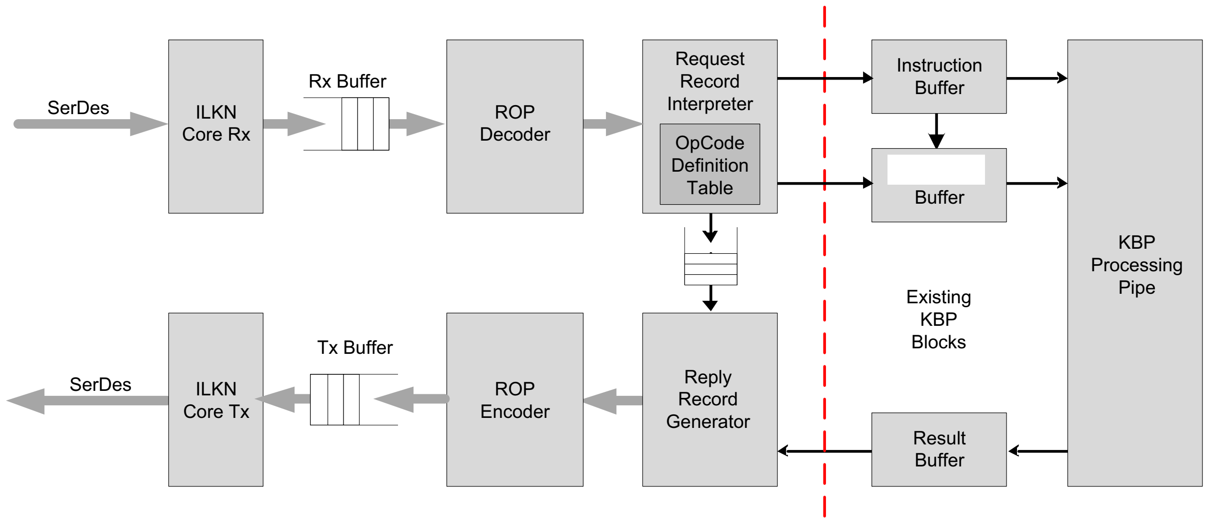
Chapter 13: ROP Layer and Interlaken Controller

13.1 General Overview

The Interlaken controller block diagram describes, at a high level, the Records-Over-Packet (ROP) layer implementation with the Interlaken PCS layer.

The Interlaken (ILKN) Core Rx block with the ROP Decoder block extracts the OpCode, sequence number, and packet data. The Request Record Interpreter determines what operations to perform on each packet. After leaving the Request Record Interpreter, the packet is passed to the core KBP blocks for read, write, or search operations. Upon leaving the KBP core Result Buffer, the Reply Record Generator and ROP Encoder blocks correctly format packets for transmission via the Interlaken (ILKN) Core Tx block.

Figure 17: Interlaken Controller/ROP Functional Diagram



13.2 Interlaken Controller Overview

The Interlaken controller features and functions are described in this section.

13.2.1 Interlaken Controller Feature Set

The interlaken controller features are as follows:

- The interlaken device supports two host processors with two ports per host
- Lane configuration options single port:
  – 4x Rx/Tx, lanes 3:0 only
  – 8x Rx/Tx, lanes 7:0 only
  – 12x Rx/Tx, lanes 11:0 only
  – 16x Rx/Tx, lanes 15:0 only
- Lane configuration options dual port
  – 4x Rx/Tx per port, lanes 21:18 for port 1 and lanes 3:0 for port 0
  – 8x Rx/Tx per port, lanes 25:18 for port 1 and lanes 7:0 for port 0
  – 12x Rx/Tx per port, lanes 29:18 for port 1 and lanes 11:0 for port 0
  – 16x Rx/Tx per port, lanes 33:18 for port 1 and lanes 15:0 for port 0

  Port 0 and Port 1 lane configuration can be different
- 4 lane Increments
- Rx and Tx lanes can be configured independently
- Lane reversal and polarity inversion supported
- Traffic sent only on Channel 0

SerDes Rate

- Minimum SerDes rate is 12.5 Gb/s
- Maximum SerDes rate is 28.125 Gb/s
- In dual port mode, both ports must operate at same SerDes rate.

Packet

- BurstShort = 32B per Interlaken specification.
- BurstMax = 256B per Interlaken specification.
  Rx Interlaken core discards any packet received that violates the BurstMax value
- All packets must be sent in a single Burst
- Rx Interlaken checks that EOP and SOP fields of a burst are both asserted, and if not, the packet is considered in error (contents are discarded even if the CRC-24 is correct)
- Tx Interlaken guarantees the SOP or EOP or both SOP and EOP fields in the burst control word are set wherever applicable
- Every packet is a single burst, BurstMin applies that is, 32B

Channel and Backpressure

- Single Channel supported
  For example, channel 0 represented in Burst Control Word by Channel Number field set to all zeros. Any Burst Control Word received with Channel Number set to nonzero value is discarded by the receiver.
- Tx Interlaken guarantees the SOP or EOP or both SOP and EOP fields in the burst control word are set wherever applicable
- Tx asserts reset calendar bit in all transmitted control word
- Tx asserts Xoff for all other channels (Channel 1 to 15)

- Tx controller assert Xoff to host when Rx FIFO full condition or after power-up when the core is not ready
- KBP receiver, channel 0 Xoff bit is detected when “reset calendar bit is set”
- Multiple-use field must not be used to extend either Flow Control or Channel Number information and thus channel number field and in-band flow control field are both 8-bit values
- KBP receiver, Xoff is extracted from bit 55 of IL Control Word. On Transmit side, Flow Control information (or Xoff) is sent on IL Control Word bit 55. Xoff and channel information sent on bits [54:40] will be ignored at the Rx side.

Metaframe

- Maximum Metaframe Length = 8K words
- Default Metaframe Length is 2K words and identical for both Receive and Transmit
- Tx Interlaken only send a single SKIP word
- Rx Interlaken supports the reception of additional SKIP word and silently discards them. The minimum gap between Skip Words within a MetaFrame must be ≥ 16 Data Words.

Clocking, PPM, and Skew

- Interlaken supports the independent clocking scheme between Host and KBP with the PPM limit specified in Interlaken Specification
- Rx Interlaken controller supports up to 214 UI, value recommended by Interlaken interop specification for 25.78125 Gb/s speed (CEI-28G-SR/MR)
- Tx Interlaken controller skew between lanes is guaranteed to be lower than 40 UI (lower than 67 UI budget for PMA Tx)

  The total UI for the KBP chip output is 160. (The budget is 214 UI for speed > 5G). The internal UI distribution is shown in Table 32.

Table 32: UI Skew

SerDes	QHMF	CLK to QHMF	Final UI
1 lane	1 cycle	2.45 ns	
40 UI	40 UI	80 UI	160 UI

Features Not Supported

- Tx Interlaken does not implement the Rate Limiter feature
- Interlaken controller does not implement retransmit
- KBP does not implement sequential instructions
- KBP does not implement re-entrant instructions
- Interlaken controller does not implement EEI (Energy Efficient Interlaken)
- Multiple-use field of Interlaken Burst Control Word is ignored on Rx
- The diagnostic Word status bits[33:32] is driven to all 1s by default on the transmit side
- Diagnostic word status bits are as per IL specification. Lane and Link health are transmitted.
- No padding between records within a ROP packet. As soon as a pad is seen, the record extraction will stop and the remaining ROP packet will be discarded.
- No padding at the start of the packet. If so, that ROP packet will be discarded.
- Padding is only allowed towards the end of the ROP packet.
- No Support for INFO records.
- No support for Copy Data.
- No support for Key_cfg.
- Two modes are supported.
  - Mode 1: With 16b ContextId in the payload.
  - Mode 0: Without ContextId.

Link-up Sequence

- Interlaken Controller does not power up automatically and requires control plane software accessing the KBP interface controller through MDIO/PCIe to configure the SerDes as well as the Interlaken Controller before enabling the Interlaken training.
- The Interlaken controller implements a complete register space accessible through the MDIO/PCIe registers for link-up, debug, statistics, and more.

13.2.1.1 Interlaken Controller Performance

This section highlights the Interlaken-ROP layer performance, latency, and potential limitations.

On the receive side, the Interlaken ROP is able to sustain the full bandwidth of Interlaken protocol without stalling the Host (assuming that the KBP Core logic does not stall the Interface receiver). We support one record extraction per core clock per port and are able to sustain single cycle extraction at 900 MPPS.

13.2.1.2 Interlaken Descriptor Formats

Within a packet, Control Words are transmitted first followed by an optional burst of Data Words (DW). All bursts of Data Words must be transmitted in order with DW0 first, and the number of Data Words is variable per the Interlaken specifications with a minimum of one Data Word on Rx and four Data Words on Tx per packet.

The Control Word for each instruction contains fields defined by the Interlaken specification. These include bits 66:57, bits 40:39, and bit 32. For more information about these bits refer to Table 33 on page 60.

13.2.1.2.1 Example of Interlaken Instruction

13.2.1.2.2 Control Word

66	65:64	63	0
INV	Framing	Control Word	

13.2.1.2.3 Data Words

66	65:64	63	32	31	0
Data Word 0					
INV	Framing	Valid Data (LSB)			
Data Word 1					
INV	Framing	Valid Data			
Data Word 2					
INV	Framing	Valid Data (MSB)		32'b0	

13.2.1.2.4 Interlaken Control Word Format for Request and Response Packets

The Interlaken Control Word for device request and response packets is described in this section. Interlaken-specific fields are described in detail in Table 33 on page 60. Device specific fields are described in detail in the following sections for each instruction.

Table 33: Interlaken Control Word Format for Request and Response Packets

Bit	Field Name	Description
66	Inversion	Interlaken-specified field Used to indicate whether bits [63:0] have been inverted to limit the running disparity 1'b1 = Inverted 1'b0 = Not inverted

Table 33: Interlaken Control Word Format for Request and Response Packets (Continued)

Bit	Field Name	Description
65:64	Framing	Interlaken-specified field 64b/67b mechanism to distinguish control and data words Must be written with 10 for control words
63	Control	Interlaken-specified field 1’b1 = This is an idle or burst control word 1’b0 = This is a framing layer control word
62	Type	Interlaken-specified field 1’b1 = The channel number and SOP fields are valid and a data burst follows this control word (a ‘Burst Control Word’) 1’b0 = The channel number field and SOP fields are invalid and no data follows this control word (an ‘Idle Control Word’)
61	SOP	Interlaken-specified field Start of Packet 1’b1 = The data burst following this control word represents the start of a data packet 1’b0 = A data burst that follows this control word is either the middle or end of a packet
60:57	EOP[3:0]	Interlaken-specified field This field refers to the data burst preceding this control word 1xxx - End-of-Packet, with bits [59:57] defining the number of valid bytes in the last 8-byte word in the burst. Bits [59:57] are encoded such that 3’b000 means 8 bytes valid, 3’b001 means 1 byte valid, etc., with 3’b111 meaning 7 bytes valid. The valid bytes start with bit position [63:56]. 1’b0000 = No End-of-Packet, no ERR 1’b0001 = Error and End-of-Packet All other combinations are undefined
56	Reset Calender	Reset Calender
55	Xon/Xoff	Xon/Xoff indication
54:24	Reserved	Reserved
23:0	CRC24	Interlaken-specified field. A CRC error check that covers the previous data burst, if any, and this control word.

Interlaken Data Word Format for Request and Response Packets

The Interlaken data word for request and response packets is described in this section. Device specific fields are described in detail in the following sections for each instruction.

Table 34: Interlaken Data Word Format for Request and Response Packets

Bit	Field Name	Description
66	Inversion	Interlaken-specified field. Used to indicate whether bits [63:0] have been inverted to limit the running disparity. 1’b1 = Inverted 1’b0 = Not inverted
65:64	Control/Data	Interlaken-specified field. 64b/67b mechanism to distinguish between control and data words. This field must be written with 01 for data words.
63:0	Data/Reserved	Refer to each instruction section for details

13.3 ROP Layer Overview

The ROP protocol is used to facilitate efficient exchange of small records over relatively large packets, synchronize replies to requests, and protect against packet/record loss.

OpCode	Sequence Number	Data

8-bit OpCode

An 8-bit OpCode points to a Lookup Table (LUT) descriptor that contains all the information a sender or recipient requires about the record. The information includes the record type and the size of the data portion. OpCodes are synchronized between both sides of Rx and Tx. OpCode '0' is reserved to designate an empty record having no sequence number and no data. Thus, inserting '0' padding before or between a record within a ROP packet is illegal (called padding). Inserting '0' padding after all records in a packet is legal.

In reply records, the OpCode is copied from the corresponding Request record.

The ROP layer Lookup Table (LUT) contains a 19-bit LUT Descriptor field.

16-bit Sequence Number

The 16-bit Sequence Number is a field that is present following the OpCode of the first Request record and the first Reply record within a given packet, which is used to protect against record loss.

All Request records are assigned an incrementing sequence number. This number is explicitly inserted in packets for the first Request record.

All Reply records have a sequence number that is equal to the corresponding Request record's number. Replies within one packet have an incrementing sequence number, thus only the first Reply record's sequence number is explicitly inserted. In the event of a lost Request/Reply, a packet may have to be truncated (and possibly padded) to insure that all records within a packet have incrementing Sequence-Number.

- A field following the OpCode of the first Request record and the first Reply record within a given packet.
- Used to protect against records loss.
- All Request records are assigned an incrementing sequence number. This number is inserted in packets for the first Request record (the remainder can be deduced).
- All Reply records have a sequence number that is equal to the corresponding Request record's number. Replies within one packet have an incrementing sequence number, thus only the first Reply record's sequence number is explicitly inserted. In the event of a lost Request/Reply, a packet may have to be truncated (and possibly padded) to ensure that all records within a packet have incrementing Sequence-Number.
- If a record is lost, there is a discontinuity in the Sequence-Number. If the discontinuity falls in the middle of a packet, the packet must be truncated and possibly padded, and a new packet started. The new packet has the sequence number of the record after the lost record.

Data

Data is variable in size, and contains additional information according to the OpCode. For example, in a Request, the data field contains the lookup keys information. While in a Reply, the data field contains the lookup result.

Lookup Reply Record Data is flexibly constructed from:

- Match status
- Match Index Associated Data

Write Register Request Record contains:

- Write address
- Write data

Table 35 is an example format of a ROP Request Packet with Various Search Key Sizes. All padding has a value of '0', consistent with the interpretation of the '0' (NULL) OpCode. The above example shows the padding between the Blue record and the Green record and at the end of packet.

Table 35: ROP Packet Format Example

	66:64	**63:56**	**55:48**	**47:40**	**39:32**	**31:24**	**23:16**	**15:8**	**7:0**
CW	IL Synch	Interlaken: Control, Type, SOP, EOP, CRC24							
DW0	IL Synch	OpCode	Seq 15:8	Seq 7:0	Data 159:152	Data 151:144	Data 143:136	Data 135:128	Data 127:120
DW1	IL Synch	Data 119:112	Data 111:104	Data 103:96	Data 95:88	Data 87:80	Data 79:72	Data 71:64	Data 63:56
DW2	IL Synch	Data 55:48	Data 47:40	Data 39:32	Data 31:24	Data 23:16	Data 15:8	Data 7:0	OpCode
DW3	IL Synch	Data 79:72	Data 71:64	Data 63:56	Data 55:48	Data 47:40	Data 39:32	Data 31:24	Data 23:16
DW4	IL Synch	Data 15:8	Data 7:0	OpCode	Data 639:632	Data 631:624	Data 623:616	Data 615:608	Data 607:600
DW5	IL Synch	Data 599:592	Data 591:584	Data 583:576	Data 575:568	Data 567:560	Data 559:552	Data 551:544	Data 543:536
DW6	IL Synch	Data 535:528	Data 527:520	Data 519:512	Data 511:404	Data 503:496	Data 495:488	Data 487:480	Data 479:472
DW7	IL Synch	Data 471:464	Data 463:456	Data 455:448	Data 447:440	Data 439:432	Data 431:424	Data 423:416	Data 415:408
DW8	IL Synch	Data 407:400	Data 399:392	Data 391:384	Data 383:376	Data 375:368	Data 367:360	Data 359:352	Data 351:344
DW9	IL Synch	Data 343:336	Data 335:328	Data 327:320	Data 319:312	Data 311:304	Data 303:296	Data 295:288	Data 287:280
DW10	IL Synch	Data 279:272	Data 271:264	Data 263:256	Data 255:248	Data 247:240	Data 239:232	Data 231:224	Data 223:216
DW11	IL Synch	Data 215:208	Data 207:200	Data 199:192	Data 191:184	Data 183:176	Data 175:168	Data 167:160	Data 159:152
DW12	IL Synch	Data 151:144	Data 143:136	Data 135:128	Data 127:120	Data 119:112	Data 111:104	Data 103:96	Data 95:88
DW13	IL Synch	Data 87:80	Data 79:72	Data 71:64	Data 63:56	Data 55:48	Data 47:40	Data 39:32	Data 31:24
DW14	IL Synch	Data 23:16	Data 15:8	Data 7:0	OpCode	Data 79:72	Data 71:64	Data 63:56	Data 55:48
...									
DW30	IL Synch	Data 95:88	Data 87:80	Data 79:72	Data 71:64	Data 63:56	Data 55:48	Data 47:40	Data 39:32

Table 35: ROP Packet Format Example

	66:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
DW31	IL Synch	Data 31:24	Data 23:16	Data 15:8	Data 7:0	Pad 0s	Pad 0s	Pad 0s	Pad 0s

13.3.1 ROP Layer Performance

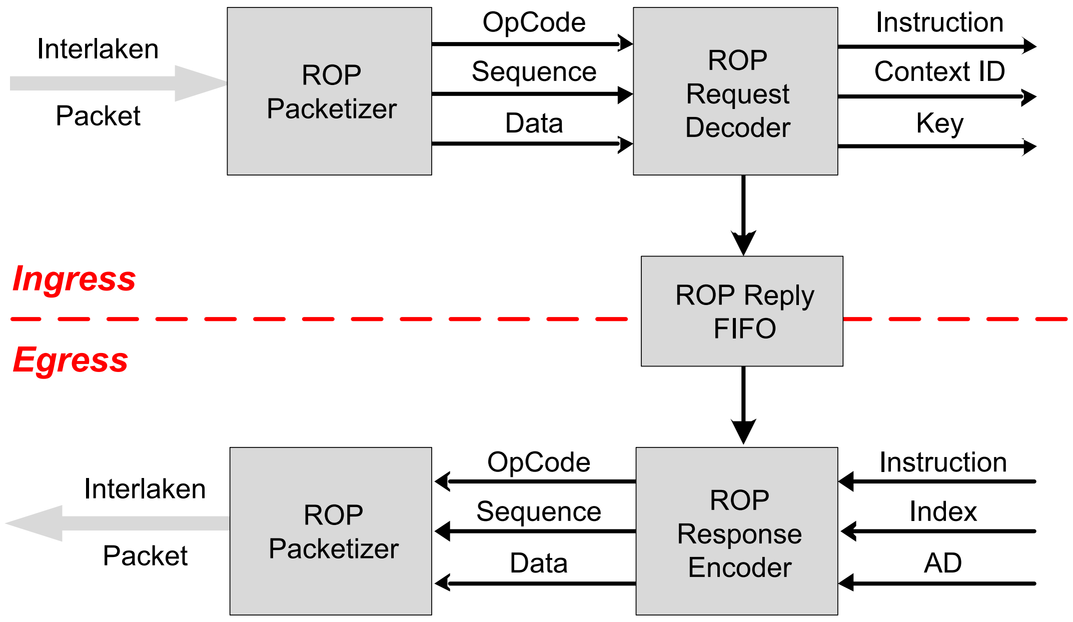
The ROP Extractor layer can extract one record every clock cycle at Core clock up to 640 bits from the Interlaken Rx layer

The following restrictions apply.

- ROP Layer LUT is programmed using LUT WR instruction. The Host must wait for LUT WR Response before start using the newly programmed entry.
- The search key is always MSB aligned on 80b boundary if the key is not a multiple of 80b.

13.3.2 ROP Ingress and Egress Flow

Figure 18: ROP Ingress and Egress Flow



For ROP ingress, the ROP Packetizer extracts (unpacks) records received from the Interlaken Rx and provides the OpCode, sequence number, and data to the ROP Request Decoder, that passes the instruction context ID and search key to the KBP core.

For ROP egress, the KBP core passes the instruction, index, and/or associated data to the ROP Response Encoder. LTR programming determines if match index or associated data is sent to the host. LTR programming also determines how responses are packed.

When a response comes from the KBP core, the SeqNum from the core and the SeqNum stored in the ROP reply FIFO are checked. If the ROP Reply FIFO contains no parity error and the SeqNums match, replies are constructed according to the core information. If the sequence numbers mismatch, the current ROP packet accumulation is closed and sent to the host. If parity error is detected, an ERROR record is returned to the host.

13.3.2.1 ROP Request Decoder Lookup Table (LUT)

The ROP Request Decoder LUT gives the size of the record (S) and record valid state. The ROP Request Decoder LUT also has a 1b Mode field with the Instruction. In Mode 0, the 10b Instruction is taken directly from the LUT entry. The 15b ContextId is fixed. The Record Data (RecData) contains up to 640b of Search Key data and must be byte aligned. Mode field is programmed in LUT Table.

Up to 640b Search Key is constructed based on the LUT programming Record Data[S-1:0] where S = maximum size of a record.

13.3.2.2 ROP OpCodes

This section describes the OpCodes that are supported by ROP Layer:

Table 36: Complete Instruction List Supported for ROP

Fields	Request Size	Response Size
Hard-Coded OpCodes		
LUT_WR (255)	OpCode: 1B Record Payload: 4B	OpCode: 1B Record Payload: 1B of data
LUT_RD (254)	OpCode: 1B Record Payload: 1B	OpCode: 1B Record Payload: 4B of data
PIO Write (253)	OpCode: 1B Record Payload: 24B of data	OpCode: 1B Record Payload: 1B of data
PIORDX and PIORDY (252, 251)	OpCode: 1B Record Payload: 4B of data	OpCode: 1B Record Payload: 11B of data
ERROR (250)	N/A	OpCode: 1B Record Payload: 8B of error status
Variable OpCodes		
NOP	OpCode: 1B Record Payload: 16B of 0	OpCode: 1B Record Payload: 1B of data
Buffer Write (applicable in Mode1 only)	OpCode: 1B Record Payload: Up to 80B of data + 2B of ContextId	OpCode: 1B Record Payload: 1B of data
Buffer Write and Compare, Up To Eight Results	OpCode: 1B Record Payload: Up to 80B of key + 2B of ContextId	OpCode: 1B Record Payload: 2B (for 1 Result, No Index, 1B AD) to 128B (for 8 results and 128b of AD each)

Table 37: ROP LUT Entry

Fields	Range	Definition
RecSize	6:0	Record Size, up to 640b data.
RecValid	7	Record is Valid. 1’b1 = Valid 1’b0 = Invalid
Mode	8	Used to indicate if the ContextId is present or not for Buffer Write and for Buffer Write and Compare instructions. Indicate active values. Mode == 1 Context ID present Mode == 0 No Context ID
Instruction	18:9	Instruction bits.

13.3.2.3 Instruction Set Definitions

The following table contains the Instruction Set definitions.

Table 38: Instruction Set Definitions

Instr[9:0] {Instr[9], Instr[8:6], Instr[5:0]}			Instruction	Description
0	3’b000	6’b000000	NOP	No operation is performed
		6’b000001	PIO Write	Performs write operation to register, database, or user data array
		6’b000010	PIORDX	Performs read operation from register, database, or user data array
		6’b000011	PIORDY	Performs read operation from database Y-value
LTR[6]	3’b010	LTR[5:0]	Buffer Write and Compare	Writes data to buffer and executes a compare up to 640b wide
	3’b100	6’b000000	Buffer Write	Performs write operation to buffer location specified by context address

13.3.3 Programming Example

Example 1:Initiate a non-CMP instruction (for example, LUT) in Mode = 0. Use opcode “X”.

7. Use LUT_WR to program the Request LUT.
8. Send record with opcode = 255 and record size 4B (the record size contains the address of the LUT and the data that must be populated in the LUTs).
   a. REQ_LUT: address “X”. Data = LDATA[18:0].
      –Program recValid = 1, recSize = 7’b1010, mode = 0, Instr = 10’h100.
9. When a record with opcode = “X” is decoded by the Ingress ROP layer:
   a. REQ_LUT will indicate to use mode = 0, size = 4B and Valid. Mode = 0 implies that there is no ContextId in the payload.

Example 2: Initiate a CMP instruction (eg CBWC1 using LTR = ”Z”) in Mode = 0. Use opcode “Y”.

1. Use LUT_WR to program the LUT’s.
2. Send record with opcode = 255 and record size 10B (the record size contains the address of the LUT and the data that must be populated in the LUTs).
   a. REQ_LUT: address “Y”. Data = LDATA[18:0].
      –Program recValid = 1, recSize = <random value from 1B to 80B>, mode = 0, Instr = {3'h1,7'hZ}, where Z is the LTR number and MSB represents the CBWC1 opcode.
3. When a record with opcode = “Y” is decoded by the Ingress ROP layer:
   a. REQ_LUT will indicate to use mode = 0, size of key and Valid. Mode = 0 implies that there is no ContextId in the payload.
   b. Payload of “recSize” is extracted from the record.
   c. Final key to core = Payload extracted from record.

Example 3:Initiate a CMP instruction (for example, CBWC1 using LTR = ”Z”) in Mode = 1. Use opcode “Y1”.

1. Use LUT_WR to program the LUT’s.
2. Send record with opcode = 255 and record size 4B (the record size contains the address of the LUT and the data that must be populated in the LUTs).
   a. REQ_LUT: address “Y1”. Data = LDATA[18:0].
      –Program recValid = 1, recSize = <random value from 3B to 82B>, mode = 1, Instr = 10’h4Z1.
3. When a record with opcode = “Y1” is decoded by the Ingress ROP layer:
   a. REQ_LUT will indicate to use mode = 1, size of key and Valid. This implies that there is ContextId in the payload.
   b. Payload of “recSize” is extracted from the record (= record_payload).
   c. In mode = 1, the ContextId and Key should be extracted from the record payload.
      –Instruction = LUT entry
      –ContextId = record_payload[msb: msb-15]]
      –Final Key to core = record_payload[msb-16:0].

13.4 Instruction Formats

This section provides request and response formatting for each instruction.

13.4.1 LUT_WR—Lookup Table Write

This instruction is used to program the LUT with programmable OpCodes.

13.4.1.1 LUT_WR Request Format (OpCode = 255)

The record size is 4B.

66 64	63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
IL-Sync	OpCode [7:0]= 255	SeqNum [15:8]	SeqNum [7:0]	Addr [7:0]	5’b0, LData [18:16]	LData [15:8]	LData [7:0]	NxtRec OpCode

Table 39: LUT_WR Request Fields

Field	Description
IL-Sync	Interlaken sync bits
OpCode[7:0]	255
SeqNum[15:0]	Request sequence number
Addr[7:0]	Lookup table address
LData[18:0]	LUT[18:0]
NxtRec OpCode	Start of the next record OpCode

13.4.1.2 LUT_WR Response Format

The record size is 1B.

66 64	63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
IL-Sync	OpCode [7:0]	SeqNum [15:8]	SeqNum [7:0]	Data [7:0]=STS	NxtRec OpCode	–	–	–

Table 40: LUT_WR Response Fields

Field	Description
IL-Sync	Interlaken sync bits
OpCode[7:0]	255
SeqNum[15:0]	Response sequence number that matches the corresponding Request sequence number
Data[7:0]	STS = 7’b0, ERR ERR: Error – indicates if the instruction has error ■ If ERR = 1’b1, then the instruction has error. ■ If ERR = 1’b0, then the instruction has no error.
NxtRec OpCode	Start of the next response OpCode

13.4.2 LUT_RD—Lookup Table Read

This section discusses the LUT read instruction at a specified address.

13.4.2.1 LUT_RD Request Format (OpCode = 254)

The record size is 1B.

66 64	**63 56**	**55 48**	**47 40**	**39 32**	**31 24**	**23 16**	**15 8**	**7 0**
IL-Sync	OpCode [7:0]= 254	SeqNum [15:8]	SeqNum [7:0]	Addr [7:0]	NxtRec OpCode	–	–	–

Table 41: LUT_RD Request Fields

Field	**Description**
IL-Sync	Interlaken sync bits
OpCode[7:0]	254
SeqNum[15:0]	Request sequence number
Addr[7:0]	Lookup table address
NxtRec OpCode	Start of the next record OpCode

13.4.2.2 LUT_RD Response Format

The record size is 4B.

66 64	**63 56**	**55 48**	**47 40**	**39 32**	**31 24**	**23 16**	**15 8**	**7 0**
IL-Sync	OpCode [7:0]	SeqNum [15:8]	SeqNum [7:0]	Rd Sts [7:0]	5'b0, LData [18:16]	LData [15:8]	LData [7:0]	NxtRec OpCode

Table 42: LUT_RD Response Fields

Field	**Description**
IL-Sync	Interlaken sync bits
OpCode[7:0]	254
SeqNum[15:0]	Response sequence number that matches the corresponding Request sequence number
Rd Sts[7:0]	7'b0, ERR. ERR: Error – indicates if the instruction has error ■ If ERR = 1'b1, then the instruction has error. ■ If ERR = 1'b0, then the instruction has no error.
LData[18:0]	LUT[18:0] data
NxtRec OpCode	Start of the next response OpCode

13.4.3 PIOWR—PIO Write

The record size is 24B.

The database entry data or mask is identified as follows:

- D/X = Data
- M/Y = Mask

13.4.3.1 PIO Write Request Format (OpCode = 253)

66 64	63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
IL-Sync	OpCode [7:0]= 253	SeqNum [15:8]	SeqNum [7:0]	Addr [31:24]	Addr [23:16]	Addr [15:8]	Addr [7:0]	D/X [79:72]
IL-Sync	D/X [71:64]	D/X [63:56]	D/X [55:48]	D/X [47:40]	D/X [39:32]	D/X [31:24]	D/X [23:16]	D/X [15:8]
IL-Sync	D/X [7:0]	M/Y [79:72]	M/Y [71:64]	M/Y [63:56]	M/Y [55:48]	M/Y [47:40]	M/Y [39:32]	M/Y [31:24]
IL-Sync	M/Y [23:16]	M/Y [15:8]	M/Y [7:0]	NxtRec OpCode	–	–	–	–

Table 43: PIO Write Request Fields

Field	Description
IL-Sync	Interlaken sync bits
OpCode[7:0]	253
SeqNum[15:0]	Request sequence number
Addr[31:0]	ADDR[31:0] = WRMODE, VBIT, 3’b0, ADDR[26:0] WRMODE: Write Mode Select Bit – selects between the following values ■ If WRMODE = 1’b0, then the device will interpret database writes in data/mask format. For 32b UDA accesses, WRMODE = 1'b0 and data is LSB aligned. ■ If WRMODE = 1’b1, then the device will interpret database writes in X/Y format. For 64b UDA accesses, WRMODE = 1'b1 and data is LSB aligned. For register writes, the value of this bit must be set to zero (WRMODE = 1’b0). VBIT: Valid Bit ■ Set VBIT =1’b1 for valid database entry. ■ Set VBIT = 1’b0 for invalid or empty database entry. For writes to any other resources, the value of this bit must be set to zero (VBIT = 1’b0). ADDR[26:0]: Addresses Database or Register Location ■ ADDR[26:25] = 10 for UDA access. ■ ADDR[26:25] = 01 for DBA access. ■ ADDR[26:25] = 00 for Register access. ■ ADDR[24:23] = 00 reserved.

Table 43: PIO Write Request Fields (Continued)

Field	Description
Data or X[79:0]	Database write: Data [79:0] - value to be written to database entry data. Register write: Data[79:0] - value to be written to register User Data Array write [31:0] - value to be written to user data array ■ If WRMODE = 1'b0, then the processor accepts data in data/mask format for DBA access. In case of UDA accesses, it implies 32b data size LSB aligned. ■ If WRMODE = 1'b1, then processor accepts data as "X" value for DBA access. In case of UDA accesses, it implies 64b data size.
Mask or Y[79:0]	Database write: Mask [79:0] - value to be written to database entry local mask. ■ If WRMODE = 1'b0, then the processor interprets mask field as in data/mask. ■ If WRMODE = 1'b1, then processor accepts data as "Y" value. This field is ignored for register and user data write operations.
NxtRec OpCode	Start of the next record OpCode

13.4.3.2 PIO Write Response Format

The record size is 1B.

66 64	**63 56**	**55 48**	**47 40**	**39 32**	**31 24**	**23 16**	**15 8**	**7 0**
IL-Sync	OpCode [7:0]	SeqNum [15:8]	SeqNum [7:0]	Data [7:0]=STS	NxtRec OpCode	–	–	–

Table 44: PIO Write Response Fields

Field	**Description**
IL-Sync	Interlaken sync bits
OpCode[7:0]	253
SeqNum[15:0]	Response sequence number that matches the corresponding Request sequence number
Data[7:0]	STS = 7'b0, ERR ERR: Error – indicates if the instruction has error ■ If ERR = 1'b1, then the instruction has error. ■ If ERR = 1'b0, then the instruction has no error.
NxtRec OpCode	Start of the next response OpCode

13.4.4 PIO Read X and Y

The record size is 4B.

13.4.4.1 PIO Read X and Y

66 64	63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
IL-Sync	OpCode [7:0]= 252/251	SeqNum [15:8]	SeqNum [7:0]	Addr [31:24]	Addr [23:16]	Addr [15:8]	Addr [7:0]	NxtRec OpCode

Table 45: PIO Read X and Y Request Fields

Field	**Description**
IL-Sync	Interlaken sync bits
OpCode[7:0]	251 = Read Database-Y 252 = Read Database-X, Register, and User Data Array.
SeqNum[15:0]	Request sequence number
Addr[31:0]	ADDR[31:0] = WRMODE, VBIT, 3’b0, ADDR[26:0] WRMODE: Write Mode Select Bit Set to 1'b0. VBIT: Valid Bit Set to 1'b0. ADDR[26:0]: Addresses Database or Register Location ■ ADDR[26:25] = 10 for UDA access ■ ADDR[26:25] = 01 for DBA access ■ ADDR[26:25] = 00 for Register access ■ ADDR[24:23] = 00 reserved
NxtRec OpCode	Start of the next record OpCode

13.4.4.2 PIO Read X and Y Response Format

The record size is 11B.

66 64	**63 56**	**55 48**	**47 40**	**39 32**	**31 24**	**23 16**	**15 8**	**7 0**
IL-Sync	OpCode [7:0] = 251/ 252	SeqNum [15:8]	SeqNum [7:0]	Rd Sts [7:0]	Data [79:72]	Data [71:64]	Data [63:56]	Data [55:48]
IL-Sync	Data [47:40]	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]	NxtRec OpCode	–

Table 46: PIO Read X and Y Response Fields

Field	**Description**
IL-Sync	Interlaken sync bits
OpCode[7:0]	251 = Read Database-Y 252 = Read Database-X, Register, and User Data Array.
SeqNum[15:0]	Response sequence number that matches the corresponding Request sequence number
Rd Sts[7:0]	STS = VBIT, UCCER, 5'b0, ERR VBIT: Valid Bit – valid bit for database entry. ▪ If VBIT =1’b1, then valid database entry. ▪ If VBIT = 1’b0, then invalid. For register reads and PIORDY, the value of this bit is zero (VBIT = 1’b0). UCCER: Uncorrectable ECC Error The database is ECC protected. If there is an uncorrectable ECC error, it will be indicated via this bit. ▪ If UCCER = 1’b1, then the instruction has an uncorrectable ECC error. ▪ If UCCER = 1’b0, then the instruction has no uncorrectable ECC error. ERR: Error – indicates if the instruction has error ▪ If ERR = 1’b1, then the instruction has error. ▪ If ERR = 1’b0, then the instruction has no error.
Data[79:0]	Database read - Data [79:0] - database X-entry data Register read - Data[79:0] - register User Data Array read - Data[63:0] - user data array
NxtRec OpCode	Start of the next response OpCode

13.4.5 Instruction Formats

This section provides request and response formatting for each compare instruction.

13.4.5.1 CMP Instruction

The format of the CMP instruction is as follows:

NOTE: The maximum Key size (Data) is 80B.

The following example shows a 10B Instruction:

66:64	**63:56**	**55:48**	**47:40**	**39:32**	**31:24**	**23:16**	**15:8**	**7:0**
IL-Sync Bits	OpCode [7:0]	SeqNum [15:8]	SeqNum [7:0]	Data [79:72]	Data [71:64]	Data [63:56]	Data [55:48]	Data [47:40]
IL-Sync Bits	Data [39:32]	Data [31:24]	Data [23:16]	Data [15:8]	Data [7:0]	NxtRec OpCode	–	–

Table 47: CMP Request Fields

Field	**Description**
IL-Sync	Interlaken sync bits
OpCode[7:0]	Compare OpCode (variable) as set up in the LUT table
SeqNum[15:0]	Request sequence number
Data[79:0]	Data to be compared (10B in the example – max 80B)
NxtRec OpCode	Start of the next record OpCode

13.4.5.2 CMP Response Format

The format of the CMP Response is as follows:

The response size can be up to 113B for eight results in a search. The following example shows a reply with seven results. R0, R2, R4, R6 are configured to send 24b Index and R1, R3, R5 are configured to send 128b AD.

	66:64	**63:56**	**55:48**	**47:40**	**39:32**	**31:24**	**23:16**	**15:8**	**7:0**
DW0	IL-Sync	OpCode [7:0]	SeqNum [15:8]	SeqNum [7:0]	MF Status [7:0] =11111110	IDX0 [23:16]	IDX0 [15:8]	IDX0 [7:0]	AD1 [127:120]
DW1	IL-Sync	AD1 [119:112]	AD1 [111:104]	AD1 [103:96]	AD1 [95:88]	AD1 [87:80]	AD1 [79:72]	AD1 [71:64]	AD1 [63:56]
DW2	IL-Sync	AD1 [55:48]	AD1 [47:40]	AD1 [39:32]	AD1 [31:24]	AD1 [23:16]	AD1 [15:8]	AD1 [7:0]	IDX2 [23:16]
DW3	IL-Sync	IDX2 [15:8]	IDX2 [7:0]	AD3 [127:120]	AD3 [119:112]	AD3 [111:104]	AD3 [103:96]	AD3 [95:88]	AD3 [87:80]
DW4	IL-Sync	AD3 [79:72]	AD3 [71:64]	AD3 [63:56]	AD3 [55:48]	AD3 [47:40]	AD3 [39:32]	AD3 [31:24]	AD3 [23:16]
DW5	IL-Sync	AD3 [15:8]	AD3 [7:0]	IDX4 [23:16]	IDX4 [15:8]	IDX4 [7:0]	AD5 [127:120]	AD5 [119:112]	AD5 [111:104]
DW6	IL-Sync	AD5 [103:96]	AD5 [95:88]	AD5 [87:80]	AD5 [79:72]	AD5 [71:64]	AD5 [63:56]	AD5 [55:48]	AD5 [47:40]
DW7	IL-Sync	AD5 [39:32]	AD5 [31:24]	AD5 [23:16]	AD5 [15:8]	AD5 [7:0]	IDX6 [23:16]	IDX6 [15:8]	IDX6 [7:0]
DW8	IL-Sync	NxtRec OpCode	–	–	–	–	–	–	–

Table 48: CMP Response Fields

Response Field	**Definition**
IL-Sync	Interlaken sync bits.
Opcode[7:0]	Opcode of Request Record.
SeqNum[15:0]	Response sequence number that matches the corresponding Request sequence number.
MF Status[7:0]	Match Flag 0= Match was not found. 1= Match was found and Index and AD are Valid.
IDX[23:0]	Match Index is valid only if MF bit is ‘1’.
AD[nn:0]	Associated data is valid only if MF is ‘1’. Size of AD depends on Result size requested.
NxtRec OpCode	Start of the next response OpCode.

13.4.6 ERROR Record

The record size is 10B.

The Error Record is sent if eight_rslt_en is set to 1 and any compare instruction encounters an error.

Whenever an error is encountered in the ROP layer during the processing of an ROP record or if the record encounters any kind of error in the KBP Core, the following error record (fixed opcode = 250) is sent back to the Host. The response payload portion contains the error status information. The request opcode that encountered this error will be sent as part of the Error response payload.

Table 49: When ERROR Record is the Only Record in a Packet

	66:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
DW0	IL-Sync	OpCode = 250	SeqNum [15:8]	SeqNum [7:0]	InSeqNum [15:8]	InSeqNum [7:0]	InOpCode [7:0]	Err Sts [55:48]	Err Sts [47:40]
DW1	IL-Sync	Err Sts [39:32]	Err Sts [31:24]	Err Sts [23:16]	Err Sts [15:8]	Err Sts [7:0]	NxtRec OpCode	–	–

Table 50: When ERROR Record is the Last Record in a Packet

	66:64	63:56	55:48	47:40	39:32	31:24	23:16	15:8	7:0
DW0	IL-Sync	OpCode = 250	InSeqNum [15:8]	InSeqNum [7:0]	InOpCode [7:0]	Err Sts [55:48]	Err Sts [47:40]	Err Sts [39:32]	Err Sts [31:24]
DW1	IL-Sync	Err Sts [23:16]	Err Sts [15:8]	Err Sts [7:0]	NxtRec OpCode	–	–	–	–

Table 51: ERROR Response Fields

Response Field	Definition
IL-Sync	Interlaken sync bits
OpCode[7:0]	250
SeqNum[15:0]	Response sequence number.
InSeqNum[15:0]	Incoming packet sequence number that generated the error.
InOpCode[7:0]	Incoming OpCode that generated the error.
Err Sts[55:48]	Bits[7:1]: Always 0 Bit[0]: Reply FIFO Error
Err Sts[47:0]	This bits can also be retrieved from Error Status Register. A summary on the Interface related errors is provided in Table 52. For additional details, please refer to the Registers document or contact Broadcom application team.
NxtRec OpCode	Start of the next response OpCode.

Table 52: Summary on Interface related errors (ERROR Response)

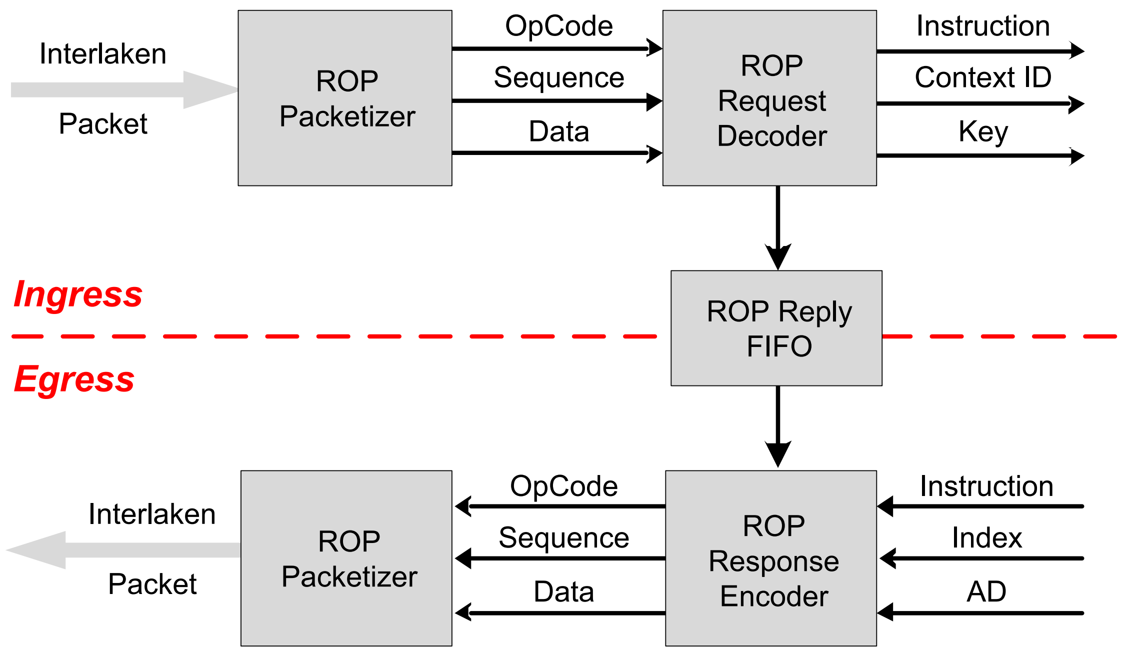
Err Sts[17]	Host Transmit Interface Error
Err Sts[16]	Host Receive Interface Error
Err Sts[15:12]	Reserved

Table 52: Summary on Interface related errors (ERROR Response) (Continued)

Err Sts[11]	Instruction Length Error on Receive Interface
Err Sts[10]	Illegal Instruction Error on Receive Interface.
Err Sts[09]	“Illegal End Of Packet (EOP) setting in the Control Word” Error on Receive Interface.
Err Sts[08]	Burst Max Error on Receive Interface.
Err Sts[07]	Burst Short Error on Receive Interface.
Err Sts[06]	Protocol Error on Receive Interface.
Err Sts[05]	“Missing Start of Packet (SOP) in Burst Control Word” Error on Receive Interface.
Err Sts[04]	MAC FIFO Parity Error on Receive Interface.
Err Sts[03]	“Missing End of Packet (EOP) in Burst Control Word” Error on Receive Interface.
Err Sts[02]	Framing (ILA-Sync) Error on Receive Interface.
Err Sts[01]	CRC24 Error on Receive Interface.
Err Sts[00]	Reserved

13.4.7 ROP Egress Flow

Figure 19: ROP Egress Flow



On egress, the KBP core passes the instruction, index, and/or associated data to the ROP Response Encoder. LTR programming determines if match index and/or associated data are sent to the host. LTR programming also determines how responses are packed.

Chapter 14: Device Initialization

This section outlines the specific power-up, initialization, and reset sequence required by the processor. Contact your local Broadcom field applications engineer representative for further information on device initialization.

14.1 Initialization Flow

Figure 20: High-Level KBP Initialization Flow

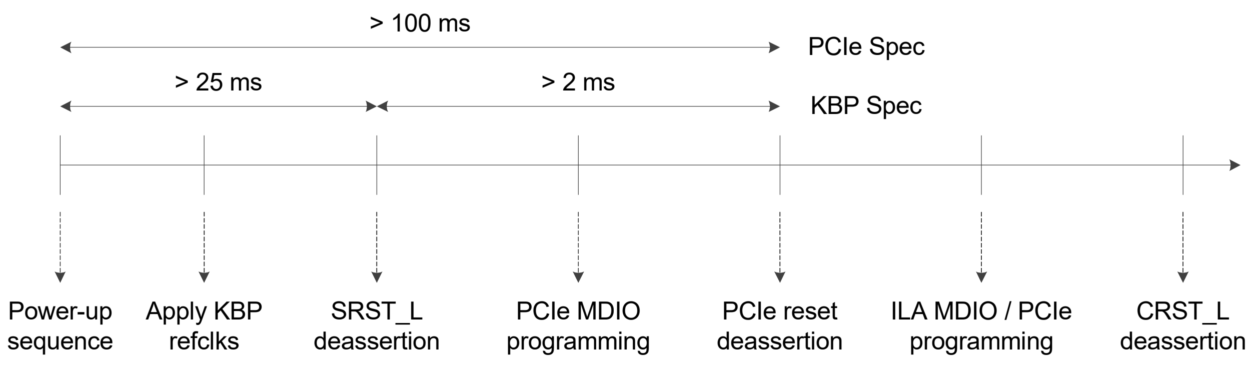


Figure 20 describes the high-level steps required in initializing the KBP. The steps are:

1. Power up.
2. Apply KBP reference clocks (clocks must not be applied while KBP power supplies are down).
3. SRST_L deassertion.
4. MDIO programming to configure PCIe interface.
5. PERST_L deassertion.
6. MDIO and/or PCIe programming for device configuration (ILA, device modes, etc.).
7. CRST_L deassertion.

Designers must ensure that the minimum timing requirements as shown in Figure 20 are met.

14.2 Power-Up Sequence

Refer to the *16 nm KBP Hardware Design Guide* ("References").

14.3 Power-Down Sequence

Refer to the *16 nm KBP Hardware Design Guide* ("References").

14.4 PCIe Considerations

A PCIe interface is highly recommended in order to maximize control plane performance. Customers who are designing with PCIe-use in mind should pay careful attention to the following list of considerations since they are new to this device. For some customers, changes may be required to the KBP power supply and reset controls.

- KBP power-up sequence must be completed prior to PCIe reset deassertion and enumeration.
- SRST_L must be deasserted prior to PCIe reset deassertion and enumeration.
- MDIO programming may be required prior to PCIe reset deassertion and enumeration. Some designs may require MDIO to be controlled during system boot while others may require the ability to re-enumerate PCIe components. Note that additional MDIO programming may be required for configuring the device later in the initialization sequence.
- After the device and the system have been fully initialized, any time SRST_L is asserted, the device will be in full reset including the PCIe interface. Consider the effects of bringing down the KBP PCIe link and what is required to re-establish the link once the KBP is brought back up out of reset.

Chapter 15: Device Addressing

15.1 Combined Resource Access

The processor is composed of user data, register and database entry locations.

The following address format is used for all device register and database write/read accesses.

Table 53: Combined Resource Address Map

Address Fields A[26:0]		**Description**
A[26:25]	**A[24:0]**	
Registers A[26:25] = 2'b00	25'h000_0000–25'h03F_FFFF	Registers
Database Array A[26:25] = 2'b01	25'h000_0000–25'h00F_FFFF	Up to 1M 80-bit words
User Data Array (UDA) A[26:25] = 2'b10	25'h000_0000–25'h0FF_FFFF	Up to 16M 32-bit words

Chapter 16: Dual Host

16.1 Overview

In a dual host implementation, the KBP device is connected to two independent ports. These ports can be either two separate host devices, or a single host device with two ports. Both ports must operate at same SerDes rate.

In dual host mode, the external SerDes channels are divided into two groups. The number of SerDes channels in each group depends on the system configuration.

16.2 Lane Assignments in Dual Host Mode

The KBP provides the following types of lane configurations for connection to the host device.

Table 54: Lane Assignment for Single Host Mode

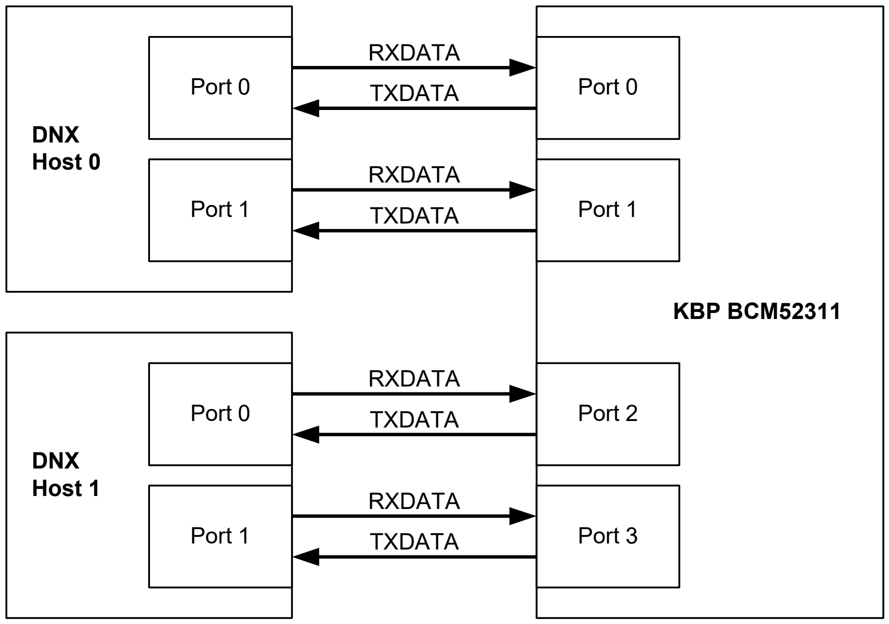
	Host 0	
Lanes Per Host	**Port 0**	**Port 1**
4 lanes	3:0	21:18
6 lanes	5:0	23:18
8 lanes	7:0	25:18
10 lanes	9:0	27:18
12 lanes	11:0	29:18
16 lanes	15:0	33:18

Table 55: Lane Assignments in Dual Host Mode

	Host 0		**Host 1**	
Lanes Per Host	**Port 0**	**Port 1**	**Port 0**	**Port 1**
4 lanes	1:0	9:8	17:16	25:24
6 lanes	2:0	10:8	18:16	26:24
8 lanes	3:0	11:8	19:16	27:24
10 lanes	4:0	12:8	20:16	28:24
12 lanes	5:0	13:8	21:16	29:24
16 lanes	7:0	15:8	23:16	31:24

Figure 21 shows the lane configuration.

Figure 21: Dual Host Mode



The number of lanes per port is set by accessing the registers via the MDIO interface.

16.3 Dual Host Feature Summary

Table 56: Dual Host Feature Summary

Features	**Description**
Number of Lanes	*Dual Port*: Maximum of 16 lanes per port. *Quad Port:* Maximum of 8 lanes per port.
Lane Swapping	M and N values are a subset of the number of lane configurations mentioned. Single Port (16 lanes maximum) Lane [N] becomes lane [15–N], where N = 0–15. Dual Host (16 lanes each maximum) Port0: Lane [N] becomes lane [15–N], where N = 0–15 Port1: Lane [M] becomes lane [33–M], where M = 18–33 The Tx and Rx swapping have independent control.
Skew (Lanes)	Transmit skew between all enabled lanes is guaranteed to be lower than 40 UI. Rx skew between all enabled lanes is tolerable to maximum of 900 UI (standard requirement is 204 UI for 28G).
Speeds	28.125 Gb/s 27.34375 Gb/s 25.78125 Gb/s 15.0 Gb/s 12.5 Gb/s 10.3125 Gb/s
Protocol	■ ILA ■ Burst Short of 8 or 16 (default) bytes (Tx) ■ BurstMax: 80B (Rx) ■ Only 1 channel.(Channel number 0) ■ EOP byte count transmitted in the ILA control word has byte granularity ■ 1 skip word sent on Tx ■ 1 (and only 1) skip word expected on Rx per meta frame ■ Programmable MetaFrameLength up to 8K words ■ Burst Short: 32B ■ BurstMax: 256B ■ Maximum packet Size = 2Kb. Register programmable. ■ Minimum packet size= 1050b ■ EOP byte count transmitted in the IL control word has byte granularity ■ Allows for packetization of multiple response into one packet ■ 1 skip word sent on Tx ■ 1 N skip word expected on Rx per meta frame ■ Programmable MetaFrameLength up to 8K words References: ■ Interlaken Protocol Definition (v1.2) ■ Interlaken Look-Aside Protocol definition (v1.1) ■ Interlaken Clarification (v1.1)
ROP Attributes	2 Ports, no SMT with same SerDes attributes.
Response Related	Zeroes out the 24b Index and 1B–32B AD for a Valid Miss Result. The rest of the fields in the Index Status Byte remain untouched (if only IndexOrAD is used, the AD/Index field is still zeroed out).

Table 56: Dual Host Feature Summary (Continued)

Features	Description
Ordering	All responses are in strict order irrespective of interface mode.
Ports	2 Ports: Both ports must have same speed.
Banks (SMTs)	2 Banks.
Max Results	8 Results Max. In case of 8 results with AD, max size of AD per result becomes 128b.
Configurability	SerDes lanes, see Number of Lane feature above
	Metaframe A programmable MetaFrame to guarantee lane alignment and scrambler synchronization Programmable MetaFrameLength. (up to 8K IL/ILA words, each word being 67b)
	MDIO interface for configuration and status reporting
Robustness	64b/67b data decoding and descrambling CRC32 check for lane integrity respectively The Transmit and Receive MAC reply FIFO are Parity protected
Flow Control	In-band flow control

16.4 Dual Host Mode Register Settings

To program the KBP device in dual host mode, reference the following registers in the *KBP 16 nm Gen1 Registers Map* (see "References"):

- HST_RX_REGS_HST_CONFIG_EN - HST_CONFIG_EN
- HST_RX_REGS_PORT0_1_REFCLK_SEL - REFCLK_SEL

Chapter 17: LMax

17.1 Overview

KBP issues Xoff (transmit off) to the source device as an indication that the KBP's input FIFOs have reached a threshold. LMax is defined as the maximum response time, from the time the input FIFO threshold has been detected until the source of data (NPU/ASIC) stops sending data to the response to that Xoff is observed on the data path. The Xoff LMax can also be defined as how much additional data can be received after input FIFO threshold has been reached without data loss.

The device input FIFO is used to buffer packets received. When the packets in the FIFO reach a threshold value (Xoff threshold), the device sends an Xoff in all outgoing Interlaken-LA Control Words of that port. Upon receiving the Xoff, the ASIC/NPU stops packet transmission to this device. After the packets stored in the FIFO fall below a threshold (Xon threshold), the device sends an Xon in all outgoing Interlaken-LA Control Words, and upon receiving this information, the ASIC or NPU device resumes packet transmission of that port.

17.2 Input FIFO Architecture

Incoming instruction packets generate information that is stored in the input FIFO. The KBP issues Xoff when the FIFO reaches its Xoff threshold, and reissues Xon when the Xon threshold is reached.

Xoff is declared when the FIFO threshold ≥ the programmed value in MAC_REGS_RXMAC_THRESHOLD_PT*_HI.

Xon is declared when the FIFO threshold ≤ the programmed value in MAC_REGS_RXMAC_THRESHOLD_PT*_LO.

The input FIFO total depth is 12288 entries, where the entry width is 8B, that is, 64b. For each instruction packet, one input FIFO entry is used to store the relevant information from the ILA control word and additional input FIFO entries are used to store the information from the ILA data words. For example, a Compare1 search with 80b search key has one ILA control word and two ILA data words; therefore, this instruction will consume three input FIFO entries.

Bank configuration does not affect the FIFO depth.

Table 57: Total Number of Input FIFO Entries for Different Port Combinations

Port Combination	Number of Input FIFO Entries
Single Port	12,288
Dual Host	6,144

17.3 How to Program Xoff/Xon Threshold Registers

The threshold values that are written into MAC_REGS_RXMAC_THRESHOLD_PT*_HI and MAC_REGS_RXMAC_THRESHOLD_PT*_LO registers described in *KBP 16 nm Gen1 Registers Map* (see “References”) must be calculated using the following formula.

Register Value = (Desired Threshold in Number of Entries)/(Division Factor)

The Division Factor is listed in Table 58.

Table 58: FIFO Division Factors

Port Combination	Input FIFO Division Factor
Single Port	24
Dual Host	12

For example, in single port configuration, the default desired Xoff threshold for the input FIFO is 4800. The Division Factor for the input FIFO in single port configuration is 24. Therefore, the default Xoff threshold programmed into MAC_REGS_RXMAC_THRESHOLD_PT*_HI and MAC_REGS_RXMAC_THRESHOLD_PT*_LO registers is 4800/24 = 200.

Table 59 shows how the default threshold values translate to number of input FIFO entries and other relevant metrics.

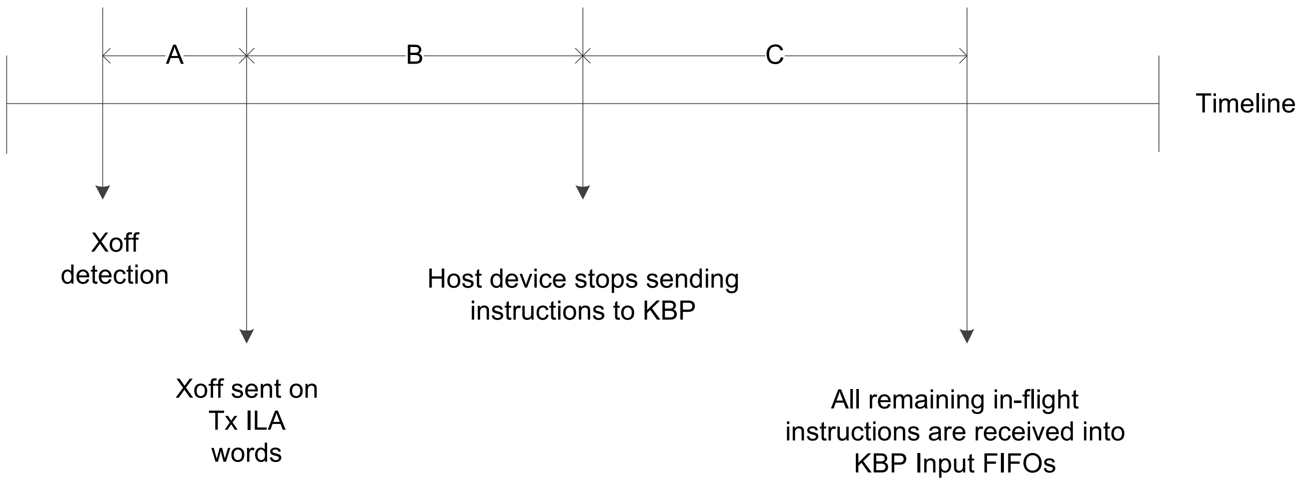
Table 59: Default Threshold for Data FIFOs

Port Combination	Default Register Value (Xoff/Xon)	Number of Entries or ILA Words (Xoff/Xon	Number of Bytes (Xoff/Xon)
Single Port	200/192	4,800/4,608	38,400B/36,864B
Dual Host		2,400/2,304	19,200B/18,432B

17.4 Calculating System LMax Requirements

Xoff threshold value must be carefully selected in order to avoid input FIFO overflow. Figure 22 shows the timeline of events that occur when Xoff is detected in the input FIFO.

Figure 22: System Latency


A
B
C
Timeline
Xoff detection
Host device stops sending instructions to KBP
Xoff sent on Tx ILA words
All remaining in-flight instructions are received into KBP Input FIFOs

When Xoff is detected, there are three delays until there are no more instructions being inserted into the KBP's input FIFO.

- Delay A: This delay occurs from Xoff threshold detection until Xoff transmission on KBP transmit ILA control words.
- Delay B: This delay occurs from the transmission of Xoff until the host depletes all in-flight KBP instructions to the pin of the host device. This time includes the time it takes for host to detect the Xoff through the ILA controller, to stop the transmission of instruction packets, and to deplete the in-flight instructions within the host device's ILA output pipeline.
- Delay C: This delay occurs while depleting the remaining in-flight instructions within the KBP input pipeline into the input FIFO.

The host controller designer must select the input FIFO threshold such that when Xoff threshold is detected, the remaining buffer in the FIFO can absorb the maximum number of instructions that will continue to be issued to the KBP during the three delays described above.

Delay A and C are within the KBP during which 43 ILA words will be transmitted per lane. Across 18 lanes, this means that a total of 774 ILA words can be transmitted during delays A and C. In the worst case, all of these words can be valid data that must be stored into the input FIFO.

Delay B depends on the host device ILA controller and the designer should make every effort to minimize this Delay.

For reference, Table 60 and Table 61 show the available input FIFO entries for different lane configurations for single port and dual host configurations respectively.

The available entries indicate the buffer available for host controller delay to Xoff event (delay B). All the data in the tables has been rounded up to the nearest entry.

NOTE: The rate at which the data FIFO fills up depends on the number of enabled lanes and the SerDes speed, whereas the rate at which the control FIFO fills up depends on both of these factors as well as the instruction packet size.

Table 60: Available Input FIFO Entries in Single-Port Mode

Port Combination	24 Lanes	16 Lanes	12 Lanes	8 Lanes	4 Lanes
Single Port	(12,288 – 4,800 – (43 x NumberOfLanes)) = 6,456	6,800	6,972	7,144	7,316

Table 61: Available Input FIFO Entries in Dual-Host Mode

Port Combination	16 Lanes	12 Lanes	8 Lanes	4 Lanes
Dual Host	(6,144 – 2,400 – (43 x NumberOfLanes)) = 3,056	3,228	3,400	3,572

Chapter 18: Latency

The processor uses a fully pipelined architecture for all operations. The latency of the device is defined as the time from when the device detects the request on the Rx bus pins to the time the response is sent on the Tx bus.

Table 62: High-Frequency Latency Table

SerDes Speed (Gb/s)	750 MHz		900 MHz	
	Min. (ns)	Max. (ns)	Min. (ns)	Max. (ns)
10.3125	531	597	465	531
12.5	496	550	430	484
15.0	468	513	402	447
25.78125	412	440	347	374
27.34375	391	429	338	363
28.125	403	428	338	362

Chapter 19: NetRoute and NetACL

19.1 Overview

This device is a “heterogeneous” knowledge-based processor (KBP) featuring the convergence of massively parallel circuit technology with the superior low-power efficiency and flexibility of NetACL and NetRoute technologies. The processor enables multiple parallel searches and assemble all outputs into a single result based on a user-defined priority.

The processor employs advanced processing engines and intelligent load balancing to provide a dramatic increase in forwarding and ACL capacity, reduced power consumption, a single device with a fixed latency, and the ability to dynamically configure application tables for forwarding and Access Control Lists (ACL) based on user scenarios.

The processor has fully deterministic longest-prefix match (LPM), ACL, and exact match functionality, and serves a broad range of forwarding applications including IPv4 unicast/multicast, IPv6 unicast/multicast, and MPLS-VPN.

NetRoute has the capability to perform two LPM in parallel and select one of the results, which are referred to as “public” and “private”. If “private” matches, then we select private; otherwise “public” is selected. An additional feature for default routes enables the ability to support private, private or public, and public.

The processor provides the largest database in the industry, supporting up to 12M IPv4 routes or 8M IPv6 routes and 4M 160-bit ACL entries on a single chip. The processor operates at line rate with no impact to the maximum number of decisions per second with a low fixed latency for all key sizes.

19.2 Features

- Support for 10M IPv4 route entries with VPN
- Support for 6M IPv6 route entries with VPN
- Support for 3M x 160b ACLs
- Capacity scales approximately linearly with entry width
- Support for associated data and indirection tables
- 8x parallel searching without restrictions
- Up to 100k table updates per second
- Low latency for both forwarding and ACL applications
- Flexible database configuration to support both traditional KBP, NetRoute, and NetACL applications

19.3 NetRoute Type Instructions for Non-Network Byte Ordering within Data Words

19.3.1 NetRoute Type1

NetRoute Type1 instruction is used by Broadcom SDK.

19.3.1.1 NetRoute Type1 Request Format

19.3.1.1.1 Control Word

66 57	56 42	41	40 39	38 33	32	31 28	27	26 24	23 0
I-LA	Context_Addr[14:0]	Rsvd	I-LA	OpCode[5:0] = 6'b001100	I-LA	Rsvd	LTR[6]/ BankSel	OpCode[8:6] = 3'b000	CRC24

19.3.1.1.2 Data Words

66 64	63 56	55 48	47 40	39 32	31 24	23 16	15 8	7 0
Data Word 0								
I-LA	m_data_p[0]	m_data_p[1]	m_data_p[2]	m_data_p[3]	m_data_p[4]	m_data_p[5]	m_data_p[6]	m_data_p[7]
Data Word 1								
I-LA	m_data_p[8] to m_data_p[15]							
Data Word 2								
I-LA	m_data_p[23] to m_data_p[16]							
Data Word 3								
I-LA	m_data_p[31] to m_data_p[24]							
Data Word 4								
I-LA	m_data_p[39] to m_data_p[32]							

Table 63: NetRoute Type1 Request Fields

Field	Description
OpCode[8:0]	9'b000_001100
LTR[6]/BankSel	1'b0
m_data_p[63:0]	uint8_t pointer from KBP SDK that contains data to be transmitted as data words in the instruction. Refer to *KBP SDK Reference Manual* for details (see "References".)

19.3.1.2 NetRoute Type1 Response Format

19.3.1.2.1 Control Word 0

66 57	56 42	41	40 39	38 33	32	31 30	29 28	27	26 24	23 0
I-LA	Context_Addr[14:0]	Rsvd	I-LA	OpCode[5:0] = 6'b001100	I-LA	Error Status	Rsvd	LTR[6]/ BankSel = Same as Request	OpCode[8:6] = 3'b000	CRC24

19.3.1.2.2 Data Words

66 64	63 0
Data Word 0	
I-LA	64'b0
Data Word 1	
I-LA	64'b0

Table 64: NetRoute Type1 Response Fields

Field	Description
DEVID[1:0]	Device ID. Device ID of instruction target device.

19.3.2 NetRoute Type2

NetRoute Type2 instruction is used by Broadcom SDK.

19.3.2.1 NetRoute Type2 Request Format

19.3.2.1.1 Control Word

66 *57*	**56** *42*	*41*	**40** *39*	**38** *33*	*32*	**31** *28*	*27*	**26** *24*	**23** *0*
I-LA	Context_Addr[14:0]	Rsvd	I-LA	OpCode[5:0] = 6’b001101	I-LA	Rsvd	LTR[6]/ BankSel	OpCode[8:6] = 3’b000	CRC24

19.3.2.1.2 Data Words

66 64	**63 56**	**55 48**	**47 40**	**39 32**	**31 24**	**23 16**	**15 8**	**7 0**
Data Word 0								
I-LA	m_data_p[0]	m_data_p[1]	m_data_p[2]	m_data_p[3]	m_data_p[4]	m_data_p[5]	m_data_p[6]	m_data_p[7]
Data Word 1								
I-LA	64’b0							

Table 65: NetRoute Type2 Request Fields

Field	Description
OpCode[8:0]	9’b000_001101
LTR[6]/BankSel	1'b0
m_data_p[7:0]	uint8_t pointer from KBP SDK that contains data to be transmitted as data words in the instruction. Refer to *KBP SDK Reference Manual* for details (see “References”.)

19.3.2.2 NetRoute Type2 Response Format

19.3.2.2.1 Control Word 0

66 57	56 42	41	40 39	38 33	32	31 30	29 28	27	26 24	23 0
I-LA	Context_Addr[14:0]	Rsvd	I-LA	OpCode[5:0] = 6’b001101	I-LA	Error Status	Rsvd	LTR[6]/ BankSel = Same as Request	OpCode[8:6] = 3’b000	CRC24

19.3.2.2.2 Data Words

66 64	63 0
Data Word 0	
I-LA	64’b0
Data Word 1	
I-LA	64’b0

Table 66: NetRoute Type2 Response Fields

Field	Description
DEVID[1:0]	Device ID. Device ID of instruction target device.

19.3.3 NetRoute Type3

NetRoute Type3 instruction is used by Broadcom SDK.

19.3.3.1 NetRoute Type3 Request Format

19.3.3.1.1 Control Word

66 *57*	56 *42*	*41*	40 *39*	38 *33*	*32*	31 *28*	*27*	26 *24*	23 *0*
I-LA	Context_Addr[14:0]	Rsvd	I-LA	OpCode[5:0] = 6’b001110	I-LA	Rsvd	LTR[6]/ BankSel	OpCode[8:6] = 3’b000	CRC24

19.3.3.1.2 Data Words

66 *64*	63 *56*	55 *48*	47 *40*	39 *32*	31 *24*	23 *16*	15 *8*	7 *0*
Data Word 0								
I-LA	m_data_p[0]	m_data_p[1]	m_data_p[2]	m_data_p[3]	m_data_p[4]	m_data_p[5]	m_data_p[6]	m_data_p[7]
Data Word 1								
I-LA	64’b0							

Table 67: NetRoute Type3 Request Fields

Field	Description
OpCode[8:0]	9’b000_001110
LTR[6]/BankSel	1'b0
m_data_p[7:0]	uint8_t pointer from KBP SDK that contains data to be transmitted as data words in the instruction. Refer to *KBP SDK Reference Manual* for details (see “References”.)

Chapter 20: Electrical Specifications

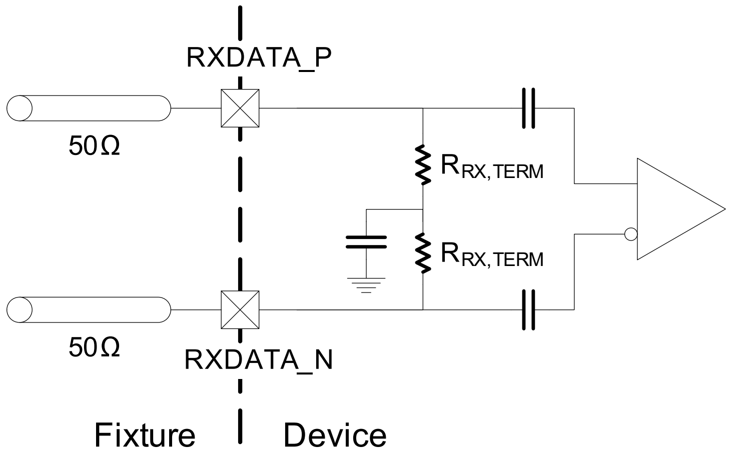
20.1 Overview

This section describes the electrical specifications for the KBP device.

20.1.1 Receiver

The RXDATA inputs' Equivalent circuit is shown in Figure 23. RXDATA_P and RXDATA_N are terminated differentially to one another on-chip, with a nominal differential resistance of 100Ω. AC coupling capacitors are implemented past the termination. The data must be DC-balanced.

Figure 23: Equivalent Input Circuit for the SerDes Receiver

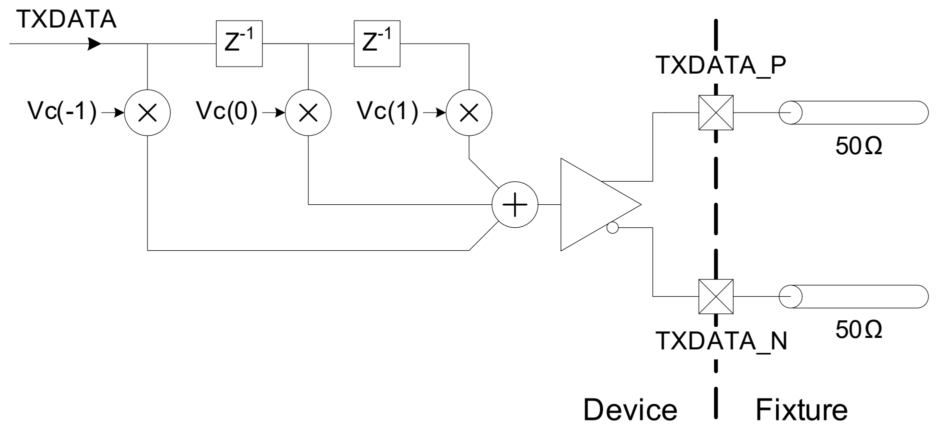


The far-end transmitter devices' total output jitter and the jitter induced by the channel cannot exceed the processor's receiver jitter tolerance, $J_{RX,TJ}$. In addition, the deterministic jitter $J_{RX,DJ}$ specification must be met.

20.1.2 Transmitter

The processor provides parallel data on the TXDATA pins. A simplified version of the output circuit and test fixture is shown in Figure 24.

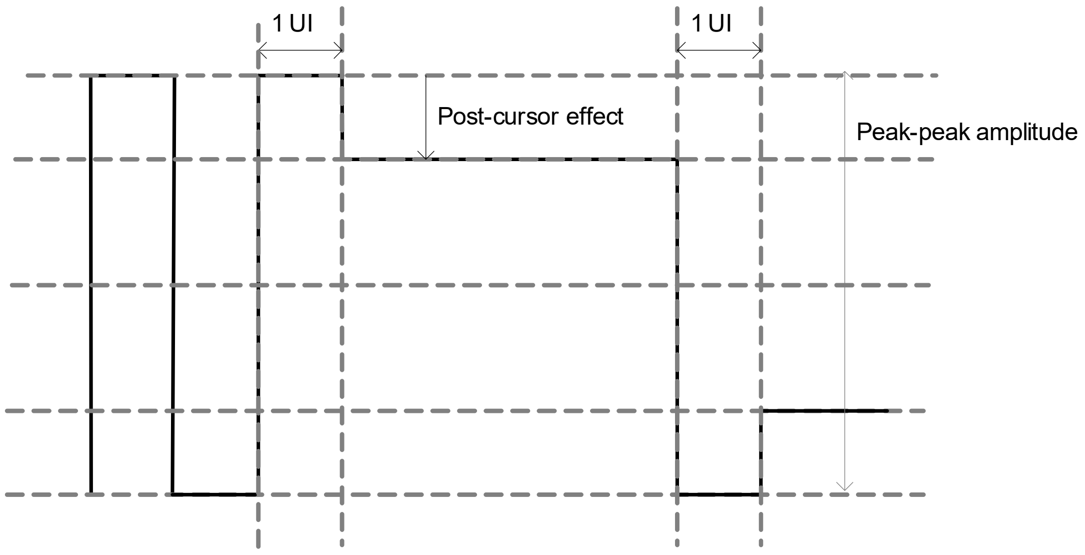
Figure 24: SerDes Equivalent Output Circuit



The drive current flowing in the output circuit has two different components: half the current is inversely proportional to the sheet resistance, and the other half has a constant value.

For Interlaken-LA applications, the processor includes a three-tap feed-forward equalizer that is useful in driving longer printed circuit traces. This equalizer preemphasizes the output signal whenever there is a data transition. The amount of preemphasis can vary between 0 and 50% and is programmed using an internal register. Figure 25 shows the effect of different register settings. Note that preemphasis does not increase the overall swing, but instead reduces the output amplitude when there is no transition. In Figure 24 on page 103, Vc(-1), Vc(0), and Vc(1) refer to pre-, main, and post-cursor, respectively.

Figure 25: Effects of Transmit Preemphasis



20.2 Electrical Specifications

Table 68: Performance Specifications

Parameter	Min.	Typical	Max.	Unit	Notes
Baud, Symbol Rate	10.3125	–	28.125	Gbd	–
Unit Interval	35.55	–	96.97	ps	–
Reference Clock Frequency	125	156.25	–	MHz	–
BER	–	–	10^{-12}	S^{-1}	Under worst conditions
RX Receiver					
Input Voltage—Differential	85	–	1600	mV	–
Input Voltage—Common Mode	–	750	900	mV	With respect to SerDes ground VSSS
Input Impedance	–	100	–	Ω	DC, differential, integrated on-chip
Jitter Tolerance—Total	–	–	0.65	UI	Peak-peak
Jitter Tolerance—Deterministic	–	–	0.37	UI	Peak-peak
TX Transmitter					
Output Voltage—Differential	0	1	1.05	V	Given VDD12 = 1.2V
Output Voltage—Common Mode	–	0.45	–	V	When terminated with 100Ω differential impedance
Output Impedance	–	100	–	Ω	DC, differential, integrated on-chip
Output Differential Skew	–	–	2	ps	50% rising/falling vs. 50% falling/rising
Output Jitter—Random	–	8	10	mUI	Wideband, RMS value
Output Jitter—Deterministic	–	0.05	–	UI	Peak-peak
Output Jitter – total	–	0.15	0.28	UI	Peak-peak

Table 69: SerDes Data Rates

Refclk (MHz)	SerDes Rate Gb/s
156.25	10.3125
156.25	12.5
156.25	25.78125
156.25	27.34375
156.25	28.125
125	15

20.3 Absolute Maximum Ratings

Stresses and conditions greater than those rated below will cause permanent damage to the processor, resulting in failures.

Table 70: Absolute Maximum Ratings

Parameter	Range Min.	Range Max.
VDD	–0.5V	+1.05V
VDD18	–0.5V	+1.98V
VTT	–0.5V	+1.32V
VDDM	–0.5V	+0.935V
VDDS1-9	–0.5V	+0.88V
VDDSPLL1-9	–0.5V	+0.88V
VDDPC	–0.5V	+0.88V
VDDPCPLL	–0.5V	+0.88V
VDDCPLL	–0.5V	+1.98V
Storage Temperature	–55°C	+150°C

Table 71: Absolute Maximum Ratings—ESD Protection

Parameter	Min.	Nom.	Max.	Revision
ESD Protection (non-SerDes pins)				
Human Body Model (HBM) per EIA/JESD22-A114-E	±1000V	–	–	Ax
Charge Device Model (CDM) per EIA/JESD22-C101C-E	±250V	–	–	Ax
ESD Protection (SerDes pins)				
Human Body Model (HBM) per EIA/JESD22-A114-E	±1000V	–	–	Ax
Charge Device Model (CDM) per EIA/JESD22-C101C-E	±200V	–	–	Ax

20.4 JTAG Timing Diagram and AC Timing Parameters

Figure 26 shows the timing diagram for the JTAG signals.

Figure 26: JTAG Timing Diagram

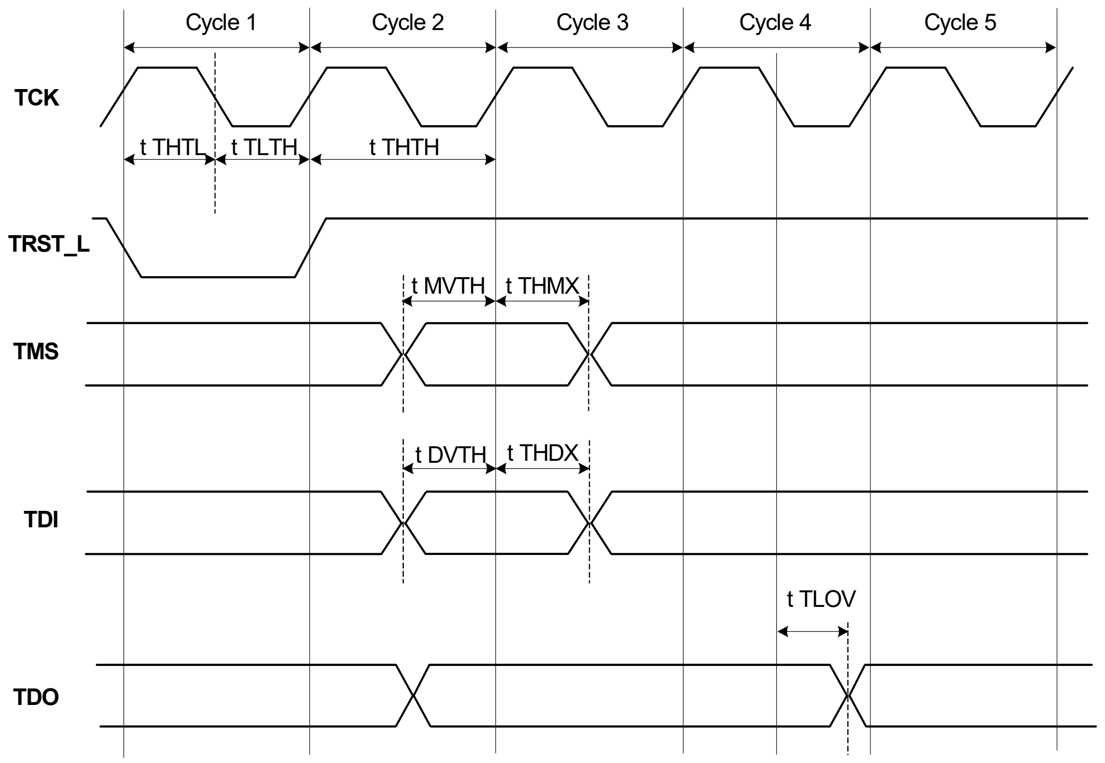


Table 72: JTAG AC Timing Parameters

Parameter	Description	Min.	Max.	Unit
t THTH	TCK Cycle Time	100	–	ns
t THTL	TCK High Pulse Width	40	–	ns
t TLTH	TCK Low Pulse Width	40	–	ns
t MVTH	TMS Setup	10	–	ns
t THMX	TMS Hold	10	–	ns
t DVTH	TDI Setup	10	–	ns
t THDX	TDI Hold	10	–	ns
t TLOV	TCK Low to Data Valid	–	25	ns

20.5 Recommended Operating Conditions

Table 73: Recommended Operating Conditions

Parameter	Condition	Description
Operating Junction Temperature – T_J	Commercial	0°C to 85°C
Operating Junction Temperature – T_J	Industrial	0°C to 95°C
Operating Junction Temperature – T_J	High	0°C to 105°C
Operating Junction Temperature – T_J	Extended	–40°C to 105°C

20.6 DC Electrical Specifications

Table 74: DC Electrical Specifications

Parameter	Symbol	Test Conditions	Min.	Nominal	Max.	Unit
Core Supply Voltage [a]	VDD	–	X[b]	0.850	X[b]	V
I/O Supply Voltage	VDD18	–	1.710	1.800	1.890	V
SerDes Termination	VDD12	–	1.165	1.200	1.235	V
SRAM Supply Voltage	VDDM	–	0.825	0.850	0.875	V
Analog SerDes	VDDS1–9	--	0.775	0.800	0.825	V
SerDes PLL	VDDSPLL1–9	--	0.775	0.800	0.825	V
Analog PCIe	VDDPC	--	0.775	0.800	0.825	V
PCIe PLL	VDDPCPLL	--	0.775	0.800	0.825	V
Core PLL Supply	VDDCPLL	–	1.745	1.800	1.855	V
Input HIGH Voltage[c]	VIH(DC)	Logic HIGH for slow speed input signals	1.145	1.800	1.890	V
Input LOW Voltage[c]	VIL(DC)	Logic LOW for slow speed input signals	–0.090	0	+0.745	V

a. The VDD spec must be met at the balls of the device. The regulator used to power the VDD rail must have the sense pins connected to the VDD and VSS balls of the device.

b. Managed by the AVS system. AVS connection and operation are required for proper device operation. For details of the regulator design, see the Board Design Guidelines.

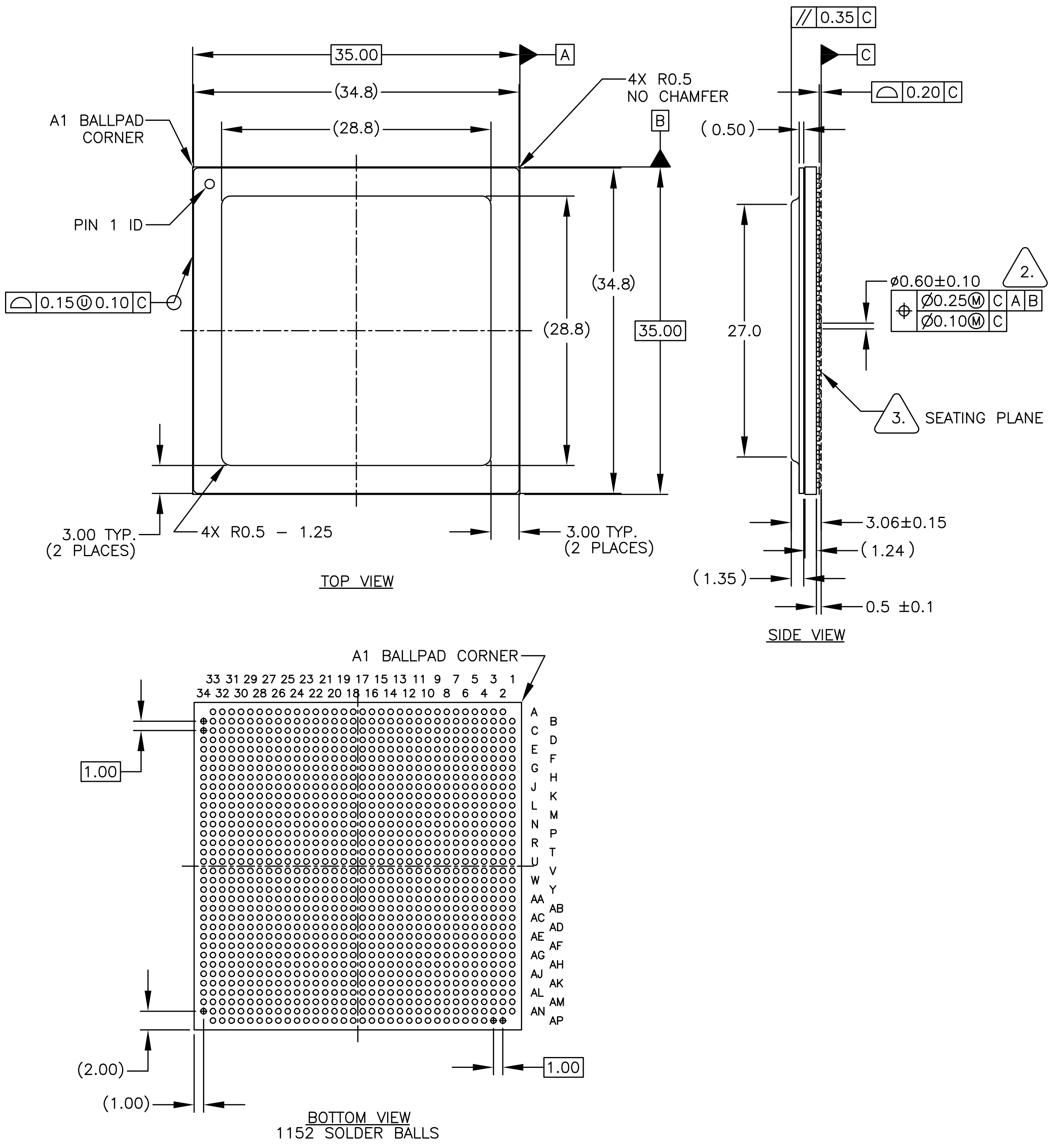
c. VIH and VIL spec also applies to MDIO and JTAG signals.

Chapter 21: Mechanical Specifications

21.1 Package Specifications

Figure 27 provides the package specifications for the KBP device.

Figure 27: Package Drawing



Package Notes:

- Controlling Dimension: Millimeter.
- Primary datum C and seating plane are defined by the spherical crowns of the solder balls.
- Dimension 2 is measured at the maximum solder ball diameter, parallel to primary datum C.
- The fiducial marker for pin 1 is for reference only.
- The maximum allowed force during heat sink attachment at room temperature is 45 lb/20.4 kgf for a maximum of 30 seconds.
- The maximum allowed sustained force during device lifespan is 28.8 lb/13.1 kgf.
- With heat sink attached, force should be uniformly applied on device surface. Concentrated force on device surface can cause die or package damage.
- An external heat sink is not allowed to tilt when in contact with the device top surface.

21.2 Package Thermal Characteristics

Customer system designs, materials and processes used for the assembly of our products (for example, selection of ther mal interface materials, methods of applying heat sinks, and so on), power dissipation levels of adjacent components on t he customer boards, and various environmental and customer board manufacture variables can greatly affect the selectio n of thermal solution required. Since these factors are unlikely within
the customer’s knowledge and control, it is essential that the customers evaluate our products and their associated solutions (for example,
recommended heat sinks) to determine whether they are fit for a particular purpose and suitable for the customer’s metho d of application while maintaining our product’s junction temperature below our maximum allowed specification.

Table 75: 50 mm x 50 mm x 35 mm Heat Sink Characteristics and Constants

Air Velocity		T_J (°C)	θ_{JA}	Constants	
m/s	ft/min			Symbol	Value
0	0	191	1.50	T_A (°C)	70
3	600	108	0.47	Power (W)	81
				θ_{JB}	1.05
				θ_{JC}	0.16

Chapter 22: Ball Assignment by Location

Signal Name	Ball
DNP	A01
VDDS9	B01
TXDATA_N<33>	C01
VSS	D01
TXDATA_P<34>	E01
VDD12	F01
TXDATA_N<35>	G01
VSS	H01
RXDATA_N<32>	J01
VDDS9	K01
RXDATA_P<33>	L01
VSS	M01
RXDATA_P<34>	N01
VDDS9	P01
RXDATA_N<35>	R01
VSS	T01
VSS	U01
PCTXDATA_N	V01
VDDPC	W01
PCRXDATA_N	Y01
VSS	AA01
PCREFCLKP	AB01
PCREFCLKN	AC01
VSS	AD01
DNC	AE01
DNC	AF01
VSS	AG01
DNC	AH01
DNC	AJ01
VDD18	AK01
DNC	AL01
DNC	AM01
VSS	AN01
DNP	AP01
VSS	A02
VDDS9	B02
TXDATA_P<33>	C02
VSS	D02
TXDATA_N<34>	E02
VDD12	F02
TXDATA_P<35>	G02
VSS	H02
RXDATA_P<32>	J02
VDDS9	K02
RXDATA_N<33>	L02
VSS	M02
RXDATA_N<34>	N02
VDDS9	P02
RXDATA_P<35>	R02
VSS	T02
VSS	U02
PCTXDATA_P	V02
VDDPC	W02
PCRXDATA_P	Y02
VSS	AA02
VDDPCPLL	AB02
VSS	AC02
RESCAL<1>	AD02
VDD18	AE02
DNC	AF02
DNC	AG02
VDD	AH02
DNC	AJ02
DNC	AK02
VSS	AL02
DNC	AM02
DNC	AN02
VSS	AP02
TXDATA_N<32>	A03
TXDATA_P<32>	B03
VSS	C03
VSS	D03
VSS	E03
VSS	F03
VSS	G03
VSS	H03
VSS	J03
VSS	K03
VSS	L03
VSS	M03
VSS	N03
VSS	P03
VSS	R03
VSS	T03
VSS	U03
VSS	V03
VSS	W03
VSS	Y03
DNC	AA03
AVSA	AB03
AVSD	AC03
VSS	AD03
VDD18	AE03
VDD	AF03
DNC	AG03
DNC	AH03
VSS	AJ03
DNC	AK03
DNC	AL03
VDD	AM03
DNC	AN03
DNC	AP03
VSS	A04
VSS	B04
TXDATA_P<31>	C04
TXDATA_N<31>	D04
VSS	E04
VSS	F04
VSS	G04
VSS	H04
RXDATA_N<31>	J04
RXDATA_P<31>	K04
VSS	L04
VDDSPLL8	M04
DNC	N04
DNC	P04
VDDSPLL9	R04
VDDSPLL9	T04
VDDS9	U04
VDDS9	V04
VSS	W04
VSS	Y04
DNC	AA04
VSS	AB04
VSS	AC04
VSS	AD04
VSS	AE04
VSS	AF04
VSS	AG04
DNC	AH04
DNC	AJ04
VDD	AK04
DNC	AL04
DNC	AM04
VSS	AN04
DNC	AP04
TXDATA_N<30>	A05
TXDATA_P<30>	B05
VDD12	C05
VDD12	D05
VSS	E05
RXDATA_P<30>	F05
RXDATA_N<30>	G05
VSS	H05
VDDS8	J05
VDDS8	K05
VSS	L05
VDD18	M05
VDD18	N05
VSS	P05
VDD18	R05
VDD18	T05
VSS	U05
VSS	V05
VSS	W05
VSS	Y05
VSS	AA05

Signal Name	Ball
VSS	AB05
VSS	AC05
VSS	AD05
VSS	AE05
VSS	AF05
VSS	AG05
VDD	AH05
DNC	AJ05
DNC	AK05
VSS	AL05
DNC	AM05
DNC	AN05
VDD	AP05
VDDS8	A06
VDDS8	B06
TXDATA_P<29>	C06
TXDATA_N<29>	D06
VSS	E06
VDDS8	F06
VDDS8	G06
VSS	H06
RXDATA_N<29>	J06
RXDATA_P<29>	K06
VSS	L06
VDDSPLL7	M06
VSS	N06
VDD	P06
VSS	R06
VDD	T06
VSS	U06
VSS	V06
VSS	W06
VSS	Y06
VSS	AA06
VSS	AB06
VSS	AC06
VSS	AD06
VSS	AE06
VSS	AF06
VSS	AG06
VSS	AH06
VSS	AJ06
DNC	AK06

Signal Name	Ball
DNC	AL06
VDD	AM06
DNC	AN06
DNC	AP06
TXDATA_N<28>	A07
TXDATA_P<28>	B07
VSS	C07
VSS	D07
VSS	E07
RXDATA_P<28>	F07
RXDATA_N<28>	G07
VSS	H07
VSS	J07
VSS	K07
VSS	L07
DNC	M07
VSS	N07
VSS	P07
VDD	R07
VSS	T07
VDD	U07
VSS	V07
VDD	W07
VSS	Y07
VDD	AA07
VSS	AB07
VDD	AC07
VSS	AD07
VDD	AE07
VSS	AF07
VDDM	AG07
VSS	AH07
VDD	AJ07
VDD	AK07
DNC	AL07
DNC	AM07
VSS	AN07
DNC	AP07
VSS	A08
VSS	B08
TXDATA_P<27>	C08
TXDATA_N<27>	D08
VSS	E08

Signal Name	Ball
VSS	F08
VSS	G08
VSS	H08
RXDATA_N<27>	J08
RXDATA_P<27>	K08
VSS	L08
DNC	M08
VSS	N08
VDD	P08
VSS	R08
VDD	T08
VSS	U08
VDD	V08
VSS	W08
VDD	Y08
VSS	AA08
VDD	AB08
VSS	AC08
VDD	AD08
VSS	AE08
VDDM	AF08
VSS	AG08
VDD	AH08
VSS	AJ08
VDD	AK08
VSS	AL08
DNC	AM08
DNC	AN08
VDD	AP08
TXDATA_N<26>	A09
TXDATA_P<26>	B09
VDD12	C09
VDD12	D09
VSS	E09
RXDATA_P<26>	F09
RXDATA_N<26>	G09
VSS	H09
VDDS7	J09
VDDS7	K09
VSS	L09
VDDSPLL6	M09
VSS	N09
VSS	P09

Signal Name	Ball
VDD	R09
VSS	T09
VDD	U09
VSS	V09
VDD	W09
VSS	Y09
VDD	AA09
VSS	AB09
VDD	AC09
VSS	AD09
VDD	AE09
VSS	AF09
VDDM	AG09
VSS	AH09
VDD	AJ09
DNC	AK09
DNC	AL09
VDD	AM09
DNC	AN09
DNC	AP09
VDDS7	A10
VDDS7	B10
TXDATA_P<25>	C10
TXDATA_N<25>	D10
VSS	E10
VDDS7	F10
VDDS7	G10
VSS	H10
RXDATA_N<25>	J10
RXDATA_P<25>	K10
VSS	L10
VDD18	M10
VSS	N10
VDD	P10
VSS	R10
VDD	T10
VSS	U10
VDD	V10
VSS	W10
VDD	Y10
VSS	AA10
VDD	AB10
VSS	AC10

Signal Name	Ball
VDD	AD10
VSS	AE10
VDDM	AF10
VSS	AG10
VDD	AH10
VSS	AJ10
VDD	AK10
DNC	AL10
DNC	AM10
VSS	AN10
DNC	AP10
TXDATA_N<24>	A11
TXDATA_P<24>	B11
VSS	C11
VSS	D11
VSS	E11
RXDATA_P<24>	F11
RXDATA_N<24>	G11
VSS	H11
VSS	J11
VSS	K11
VSS	L11
DNC	M11
VSS	N11
VSS	P11
VDD	R11
VSS	T11
VDD	U11
VSS	V11
VDD	W11
VSS	Y11
VDD	AA11
VSS	AB11
VDD	AC11
VSS	AD11
VDD	AE11
VSS	AF11
VDDM	AG11
VSS	AH11
VDD	AJ11
VSS	AK11
VSS	AL11
DNC	AM11

Signal Name	Ball
DNC	AN11
VDD	AP11
VSS	A12
VSS	B12
TXDATA_P<23>	C12
TXDATA_N<23>	D12
VSS	E12
VSS	F12
VSS	G12
VSS	H12
RXDATA_N<23>	J12
RXDATA_P<23>	K12
VSS	L12
DNC	M12
VSS	N12
VDD	P12
VSS	R12
VDD	T12
VSS	U12
VDD	V12
VSS	W12
VDD	Y12
VSS	AA12
VDD	AB12
VSS	AC12
VDD	AD12
VSS	AE12
VDDM	AF12
VSS	AG12
VDD	AH12
VSS	AJ12
VDD	AK12
VSS	AL12
VDD	AM12
DNC	AN12
DNC	AP12
TXDATA_N<22>	A13
TXDATA_P<22>	B13
VDD12	C13
VDD12	D13
VSS	E13
RXDATA_N<22>	F13
RXDATA_P<22>	G13

Signal Name	Ball
VSS	H13
VDDS6	J13
VDDS6	K13
VSS	L13
VDDSPLL5	M13
VSS	N13
VSS	P13
VDD	R13
VSS	T13
VDD	U13
VSS	V13
VDD	W13
VSS	Y13
VDD	AA13
VSS	AB13
VDD	AC13
VSS	AD13
VDD	AE13
VSS	AF13
VDDM	AG13
VSS	AH13
VDD	AJ13
VSS	AK13
VDD	AL13
VSS	AM13
VSS	AN13
DNC	AP13
VDDS6	A14
VDDS6	B14
TXDATA_N<21>	C14
TXDATA_P<21>	D14
VSS	E14
VDDS6	F14
VDDS6	G14
VSS	H14
RXDATA_P<21>	J14
RXDATA_N<21>	K14
VSS	L14
VSS	M14
VSS	N14
VDD	P14
VSS	R14
VDD	T14

Signal Name	Ball
VSS	U14
VDD	V14
VSS	W14
VDD	Y14
VSS	AA14
VDD	AB14
VSS	AC14
VDD	AD14
VSS	AE14
VDDM	AF14
VSS	AG14
VDD	AH14
VSS	AJ14
VDD	AK14
VSS	AL14
VDD	AM14
VSS	AN14
VDD	AP14
TXDATA_P<20>	A15
TXDATA_N<20>	B15
VSS	C15
VSS	D15
VSS	E15
RXDATA_N<20>	F15
RXDATA_P<20>	G15
VSS	H15
VSS	J15
VSS	K15
VSS	L15
CREFCLKN	M15
VSS	N15
VSS	P15
VDD	R15
VSS	T15
VDD	U15
VSS	V15
VDD	W15
VSS	Y15
VDD	AA15
VSS	AB15
VDD	AC15
VSS	AD15
VDD	AE15

Signal Name	Ball
VSS	AF15
VDDM	AG15
VSS	AH15
VDD	AJ15
VSS	AK15
VDD	AL15
VSS	AM15
VDD	AN15
VSS	AP15
VSS	A16
VSS	B16
TXDATA_N<19>	C16
TXDATA_P<19>	D16
VSS	E16
VSS	F16
VSS	G16
VSS	H16
RXDATA_P<19>	J16
RXDATA_N<19>	K16
VSS	L16
CREFCLKP	M16
VSS	N16
VDD	P16
VSS	R16
VDD	T16
VSS	U16
VDD	V16
VSS	W16
VDD	Y16
VSS	AA16
VDD	AB16
VSS	AC16
VDD	AD16
VSS	AE16
VDDM	AF16
VSS	AG16
VDD	AH16
VSS	AJ16
VDD	AK16
VSS	AL16
VDD	AM16
VSS	AN16
VDD	AP16

Signal Name	Ball
TXDATA_P<18>	A17
TXDATA_N<18>	B17
VDD12	C17
VDD12	D17
VSS	E17
RXDATA_N<18>	F17
RXDATA_P<18>	G17
VSS	H17
VDDS5	J17
VDDS5	K17
VSS	L17
VSSCPLL	M17
VSS	N17
VSS	P17
VDD	R17
VSS	T17
VDD	U17
VSS	V17
VDD	W17
VSS	Y17
VDD	AA17
VSS	AB17
VDD	AC17
VSS	AD17
VDD	AE17
VSS	AF17
VDDM	AG17
VSS	AH17
VDD	AJ17
VSS	AK17
VDD	AL17
VSS	AM17
VDD	AN17
VSS	AP17
VDDS5	A18
VDDS5	B18
TXDATA_N<17>	C18
TXDATA_P<17>	D18
VSS	E18
VDDS5	F18
VDDS5	G18
VSS	H18
RXDATA_P<17>	J18

Signal Name	Ball
RXDATA_N<17>	K18
VSS	L18
VDDCPLL	M18
VSS	N18
VDD	P18
VSS	R18
VDD	T18
VSS	U18
VDD	V18
VSS	W18
VDD	Y18
VSS	AA18
VDD	AB18
VSS	AC18
VDD	AD18
VSS	AE18
VDDM	AF18
VSS	AG18
VDD	AH18
VSS	AJ18
VDD	AK18
VSS	AL18
VDD	AM18
VSS	AN18
VDD	AP18
TXDATA_P<16>	A19
TXDATA_N<16>	B19
VSS	C19
VSS	D19
VSS	E19
RXDATA_N<16>	F19
RXDATA_P<16>	G19
VSS	H19
VSS	J19
VSS	K19
VSS	L19
VSS	M19
VSS	N19
VSS	P19
VDD	R19
VSS	T19
VDD	U19
VSS	V19

Signal Name	Ball
VDD	W19
VSS	Y19
VDD	AA19
VSS	AB19
VDD	AC19
VSS	AD19
VDD	AE19
VSS	AF19
VDDM	AG19
VSS	AH19
VDD	AJ19
VSS	AK19
VDD	AL19
VSS	AM19
VDD	AN19
VSS	AP19
VSS	A20
VSS	B20
TXDATA_P<12>	C20
TXDATA_N<12>	D20
VSS	E20
VSS	F20
VSS	G20
VSS	H20
RXDATA_N<12>	J20
RXDATA_P<12>	K20
VSS	L20
SREFCLKP	M20
VSS	N20
VDD	P20
VSS	R20
VDD	T20
VSS	U20
VDD	V20
VSS	W20
VDD	Y20
VSS	AA20
VDD	AB20
VSS	AC20
VDD	AD20
VSS	AE20
VDDM	AF20
VSS	AG20

Signal Name	Ball	Signal Name	Ball	Signal Name	Ball	Signal Name	Ball
VDD	AH20	TXDATA_P<14>	C22	VDDSPLL4	M23	VSS	AA24
VSS	AJ20	TXDATA_N<14>	D22	VSS	N23	VDD	AB24
VDD	AK20	VSS	E22	VSS	P23	VSS	AC24
VSS	AL20	VDDS4	F22	VDD	R23	VDD	AD24
VDD	AM20	VDDS4	G22	VSS	T23	VSS	AE24
VSS	AN20	VSS	H22	VDD	U23	VDDM	AF24
VDD	AP20	RXDATA_P<14>	J22	VSS	V23	VSS	AG24
TXDATA_N<13>	A21	RXDATA_N<14>	K22	VDD	W23	VDD	AH24
TXDATA_P<13>	B21	VSS	L22	VSS	Y23	VSS	AJ24
VDD12	C21	VSS	M22	VDD	AA23	VDD	AK24
VDD12	D21	VSS	N22	VSS	AB23	VDD	AL24
VSS	E21	VDD	P22	VDD	AC23	DNC	AM24
RXDATA_P<13>	F21	VSS	R22	VSS	AD23	DNC	AN24
RXDATA_N<13>	G21	VDD	T22	VDD	AE23	VDD	AP24
VSS	H21	VSS	U22	VSS	AF23	TXDATA_P<9>	A25
VDDS4	J21	VDD	V22	VDDM	AG23	TXDATA_N<9>	B25
VDDS4	K21	VSS	W22	VSS	AH23	VDD12	C25
VSS	L21	VDD	Y22	VDD	AJ23	VDD12	D25
SREFCLKN	M21	VSS	AA22	VSS	AK23	VSS	E25
VSS	N21	VDD	AB22	VDD	AL23	RXDATA_N<9>	F25
VSS	P21	VSS	AC22	VSS	AM23	RXDATA_P<9>	G25
VDD	R21	VDD	AD22	DNC	AN23	VSS	H25
VSS	T21	VSS	AE22	DNC	AP23	VDDS3	J25
VDD	U21	VDDM	AF22	VSS	A24	VDDS3	K25
VSS	V21	VSS	AG22	VSS	B24	VSS	L25
VDD	W21	VDD	AH22	TXDATA_N<8>	C24	DNC	M25
VSS	Y21	VSS	AJ22	TXDATA_P<8>	D24	VSS	N25
VDD	AA21	VDD	AK22	VSS	E24	VSS	P25
VSS	AB21	VSS	AL22	VSS	F24	VDD	R25
VDD	AC21	VDD	AM22	VSS	G24	VSS	T25
VSS	AD21	VDD	AN22	VSS	H24	VDD	U25
VDD	AE21	DNC	AP22	RXDATA_P<8>	J24	VSS	V25
VSS	AF21	TXDATA_N<15>	A23	RXDATA_N<8>	K24	VDD	W25
VDDM	AG21	TXDATA_P<15>	B23	VSS	L24	VSS	Y25
VSS	AH21	VSS	C23	DNC	M24	VDD	AA25
VDD	AJ21	VSS	D23	VSS	N24	VSS	AB25
VSS	AK21	VSS	E23	VDD	P24	VDD	AC25
VDD	AL21	RXDATA_N<15>	F23	VSS	R24	VSS	AD25
VSS	AM21	RXDATA_P<15>	G23	VDD	T24	VDD	AE25
VDD	AN21	VSS	H23	VSS	U24	VSS	AF25
VSS	AP21	VSS	J23	VDD	V24	VDDM	AG25
VDDS4	A22	VSS	K23	VSS	W24	VSS	AH25
VDDS4	B22	VSS	L23	VDD	Y24	VDD	AJ25

Signal Name	Ball
VSS	AK25
DNC	AL25
DNC	AM25
VSS	AN25
DNC	AP25
VDDS3	A26
VDDS3	B26
TXDATA_N<10>	C26
TXDATA_P<10>	D26
VSS	E26
VDDS3	F26
VDDS3	G26
VSS	H26
RXDATA_P<10>	J26
RXDATA_N<10>	K26
VSS	L26
VDD18	M26
VSS	N26
VDD	P26
VSS	R26
VDD	T26
VSS	U26
VDD	V26
VSS	W26
VDD	Y26
VSS	AA26
VDD	AB26
VSS	AC26
VDD	AD26
VSS	AE26
VDDM	AF26
VSS	AG26
VDD	AH26
VDD	AJ26
DNC	AK26
DNC	AL26
VDD	AM26
DNC	AN26
DNC	AP26
TXDATA_P<11>	A27
TXDATA_N<11>	B27
VSS	C27
VSS	D27

Signal Name	Ball
VSS	E27
RXDATA_N<11>	F27
RXDATA_P<11>	G27
VSS	H27
VSS	J27
VSS	K27
VSS	L27
VDDSPLL3	M27
VSS	N27
VSS	P27
VDD	R27
VSS	T27
VDD	U27
VSS	V27
VDD	W27
VSS	Y27
VDD	AA27
VSS	AB27
VDD	AC27
VSS	AD27
VDD	AE27
VSS	AF27
VDDM	AG27
VSS	AH27
DNC	AJ27
DNC	AK27
VSS	AL27
DNC	AM27
DNC	AN27
VSS	AP27
VSS	A28
VSS	B28
TXDATA_N<4>	C28
TXDATA_P<4>	D28
VSS	E28
VSS	F28
VSS	G28
VSS	H28
RXDATA_P<4>	J28
RXDATA_N<4>	K28
VSS	L28
VDD18	M28
VSS	N28

Signal Name	Ball
VDD	P28
VSS	R28
VDD	T28
VSS	U28
VDD	V28
VSS	W28
VDD	Y28
VSS	AA28
VDD	AB28
VSS	AC28
VDD	AD28
VSS	AE28
VDDM	AF28
VSS	AG28
VDD	AH28
DNC	AJ28
VDD	AK28
DNC	AL28
DNC	AM28
VDD	AN28
CPSEL<4>	AP28
TXDATA_P<5>	A29
TXDATA_N<5>	B29
VDD12	C29
VDD12	D29
VSS	E29
RXDATA_N<5>	F29
RXDATA_P<5>	G29
VSS	H29
VDDS2	J29
VDDS2	K29
VSS	L29
DNC	M29
VSS	N29
VSS	P29
VDD	R29
VSS	T29
VDD	U29
VSS	V29
VSS	W29
VSS	Y29
VSS	AA29
VSS	AB29

Signal Name	Ball
VSS	AC29
VSS	AD29
VSENSEP	AE29
VSENSEN	AF29
VDDM	AG29
VSS	AH29
VSS	AJ29
DNC	AK29
DNC	AL29
VSS	AM29
GIO_L<1>	AN29
CPSEL<3>	AP29
VDDS2	A30
VDDS2	B30
TXDATA_N<6>	C30
TXDATA_P<6>	D30
VSS	E30
VDDS2	F30
VDDS2	G30
VSS	H30
RXDATA_P<6>	J30
RXDATA_N<6>	K30
VSS	L30
DNC	M30
VSS	N30
VSS	P30
VSS	R30
VSS	T30
VSS	U30
VSS	V30
VSS	W30
VSS	Y30
VSS	AA30
VSS	AB30
VSS	AC30
VSS	AD30
VSS	AE30
VSS	AF30
VSS	AG30
VDD	AH30
DNC	AJ30
DNC	AK30
VDD	AL30

Signal Name	Ball
MSEL<1>	AM30
GIO_L<0>	AN30
VSS	AP30
TXDATA_P<7>	A31
TXDATA_N<7>	B31
VSS	C31
VSS	D31
VSS	E31
RXDATA_N<7>	F31
RXDATA_P<7>	G31
VSS	H31
VSS	J31
VSS	K31
VSS	L31
VDD18	M31
VDDSPLL2	N31
VDD18	P31
VDDSPLL1	R31
VSS	T31
VSS	U31
VSS	V31
VSS	W31
VSS	Y31
VSS	AA31
VSS	AB31
VSS	AC31
VSS	AD31
VSS	AE31
VSS	AF31
VSS	AG31
DNC	AH31
DNC	AJ31
VSS	AK31
MDIO	AL31
MSEL<0>	AM31
VDD	AN31
SRST_L	AP31
VSS	A32
VSS	B32
TXDATA_N<0>	C32
TXDATA_P<0>	D32
VSS	E32
VSS	F32

Signal Name	Ball
VSS	G32
VSS	H32
VSS	J32
VSS	K32
VSS	L32
VSS	M32
VSS	N32
VSS	P32
VSS	R32
VSS	T32
VSS	U32
VSS	V32
VSS	W32
VSS	Y32
VSS	AA32
VSS	AB32
VSS	AC32
SDA	AD32
VSS	AE32
VDD	AF32
DNC	AG32
DNC	AH32
VDD	AJ32
MDC	AK32
MPID<4>	AL32
VSS	AM32
CPSEL<2>	AN32
CRST_L	AP32
TXDATA_P<1>	A33
TXDATA_N<1>	B33
VDDS1	C33
VDDS1	D33
VSS	E33
VDD12	F33
VDD12	G33
VSS	H33
RXDATA_P<0>	J33
VDDS1	K33
RXDATA_N<1>	L33
VSS	M33
RXDATA_N<2>	N33
VDDS1	P33
RXDATA_P<3>	R33

Signal Name	Ball
VSS	T33
VSS	U33
DNC	V33
DNC	W33
VSS	Y33
VDD	AA33
VSS	AB33
PERST_L	AC33
SCL	AD33
VDD	AE33
TDO	AF33
DNC	AG33
VSS	AH33
MPID<2>	AJ33
MPID<3>	AK33
VDD	AL33
CPSEL<0>	AM33
CPSEL<1>	AN33
VSS	AP33
DNP	A34
VSS	B34
TXDATA_N<2>	C34
TXDATA_P<2>	D34
VSS	E34
TXDATA_P<3>	F34
TXDATA_N<3>	G34
VSS	H34
RXDATA_N<0>	J34
VDDS1	K34
RXDATA_P<1>	L34
VSS	M34
RXDATA_P<2>	N34
VDDS1	P34
RXDATA_N<3>	R34
VSS	T34
VSS	U34
DNC	V34
DNC	W34
RESCAL<0>	Y34
VDD18	AA34
DNC	AB34
DNC	AC34
VSS	AD34

Signal Name	Ball
TCK	AE34
TDI	AF34
VDD18	AG34
MPID<0>	AH34
MPID<1>	AJ34
VSS	AK34
TMS	AL34
TRST_L	AM34
VSS	AN34
DNP	AP34

Chapter 23: Summary of NLA88650 and BCM52311 Differences

23.1 Summary of Differences

This section summarizes the differences between this KBP device and the previous generation KBP device.

Table 76: Summary of Differences

NLA88650	BCM52311
Max. core frequency: 600 MHz.	Max. core frequency: 900 MHz.
Four KPUs, where each KPU generates one key.	Four KPUs, where each KPU generates one key or a key pair.
Maximum 4x parallel search per cycle.	Maximum 8x parallel search per cycle.
Each Array Block independently supports all block widths: 80b, 160b, 320b, and 640b.	Each Array Block supports the following block widths: 80b, 160b, and 320b. Blocks must be assigned in pairs for 480b and 640b widths.
Database Superblock: Eight blocks per superblock for 80M device, four blocks per superblock for 40M device.	Database Superblock: Eight blocks per superblock for all device densities.
480b block width not supported.	480b block width supported.
Database parity scan only, no ECC support.	Database ECC and parity scan supported.
In SMT mode, LTRs and KPUs are split evenly between two threads.	In SMT mode, LTRs and KPUs can be assigned asymmetrically between two threads.
128 Mb User Data Array.	512 Mb User Data Array.
Maximum 512b User Data Array accessible per cycle with restrictions.	Maximum 1024b User Data Array accessible per cycle with no restrictions.
User Data Array for associated data supported for non-algorithmic databases only.	User Data Array for associated data supported for all types of databases.
Limited algorithmic ACL support for power control mode.	Algorithmic ACL fully supported.
–	Register address map for device, logical tables and blocks significantly different.
–	Additional instructions: ■ Buffer No-Write and Compare. ■ Buffer Skip and Compare.
Compare1 instruction for ≤ 320b searches, one cycle searches. Compare2 instruction for > 320b searches, two cycle searches.	Compare instruction for all search key widths in one cycle.
Compare3 instruction supported.	Compare3 instruction not supported. Equivalent capability available through Compare instruction.
Cascade supported up to four devices.	Cascade not supported.
PCIe not supported.	PCIe Gen2 port.
–	A portion of KBP Initialization must be done prior to the system's PCIe enumeration or re-enumeration.
Serial Lanes: 24 RX/TX lanes up to 12.5G.	Serial Lanes: 36 RX/TX lanes up to 28.125G. Max 16 lanes for single host.
Input SerDes Reference Clock RCLK.	Signal name change to SREFCLK.
Core Reference Clock CORECLK.	Signal name change to CREFCLK.

Table 76: Summary of Differences (Continued)

NLA88650	**BCM52311**
AVS for voltage regulation not supported.	AVS for voltage regulation required.
List of power rails: VDD, VDDS, VDDA, VDDV, VTT, VDD18	List of power rails: VDD, VDDS, VDDSPLL, VDD18, VDDSRX, VDDSTX, VDDSTX12, VDDS18, VDDCPLL, VDDPC

Chapter 24: Errata

24.1 Potential Device Instability When Core Clock is Greater Than 600 MHz

Issue	Potential device instability when Core Clock (CCLK) frequency is greater than 600 MHz with SerDes operating at less than 25 Gb/s.
Workaround	Limit core frequency to 600 MHz or less when the SerDes operates at a rate of less than 25 Gb/s.
Silicon Revisions Affected	A0.
Resolution	Fixed in Revision A1.

24.2 PCIe Bus May Not Be Detected When Powered On

Issue	On some devices, the PCIe bus may not be detectable.
Workaround	MDIO writes are required between SRST_L deassertion and PERST_L deassertion. Please contact Broadcom support for the required sequence. In some cases, the receiver detect may need to be masked or disabled on the PCIe host that is connected to the KBP.
Silicon Revisions Affected	A0 and A1.
Resolution	There is no plan to fix this issue.

24.3 NPU RX May Report Link Down

Issue	After the SerDes initialization of the KBP using the KBP SDK APIs, the NPU RX may report a link down.
Workaround	Apply the following MDIO workaround to reset the KBP TX PCS until the NPU RX link comes up:

```
while (NPU RX link down for port N) {
   uint16_t value;
   if (port N) {
   mdio_write (device_id=2, addr=0xF164, 0x3037);
   mdio_write (device_id=2, addr=0xF165, 0x3037);
   mdio_write (device_id=2, addr=0xF166, 0x3037);
   mdio_write (device_id=2, addr=0xF167, 0x3037);
   mdio_write (device_id=2, addr=0xF168, 0x3037);
   usleep(10000);
   mdio_write (device_id=2, addr=0xF164, 0x3035);
   mdio_write (device_id=2, addr=0xF165, 0x3035);
   mdio_write (device_id=2, addr=0xF166, 0x3035);
   mdio_write (device_id=2, addr=0xF167, 0x3035);
   mdio_write (device_id=2, addr=0xF168, 0x3035);
   } else {
   mdio_write (device_id=2, addr=0xF160, 0x3037);
   mdio_write (device_id=2, addr=0xF161, 0x3037);
   mdio_write (device_id=2, addr=0xF162, 0x3037);
   mdio_write (device_id=2, addr=0xF163, 0x3037);
   usleep(10000);
   mdio_write (device_id=2, addr=0xF160, 0x3035);
   mdio_write (device_id=2, addr=0xF161, 0x3035);
   mdio_write (device_id=2, addr=0xF162, 0x3035);
   mdio_write (device_id=2, addr=0xF163, 0x3035);
   }
   mdio_read (device_id=2, addr=0xF00F, &value);
   mdio_write(device_id=2, addr=0xF00F, value & ~(1 << port N));
   mdio_write(device_id=2, addr=0xF00F, value);
}
```

Typically, only one or two applications of the above loop may be required. If the link issue only occurs on port N of the KBP, toggle only the TX PCS of the specific port. Bit N of value controls port N KBP.

The workaround above works for single port applications up to 16 lanes and dual-port applications with 16 lanes per port. For other configurations, please contact Broadcom support.

Silicon Revisions Affected	All
Resolution	There is no plan to fix this issue.

24.4 AC-Coupled PCIe Reference Clock Requires Common-Mode Voltage

Issue	When the PCIe reference clock is AC-coupled, a MDIO register write is necessary to turn on common-mode voltage on the KBP.
Workaround	Issue the following MDIO write prior to the PCIe reset deassertion: `mdio_write (device_id=0x11, addr=0x2505, value=0x2400);`
Silicon Revisions Affected	All
Resolution	There is no plan to fix this issue.

24.5 Switching MDIO Access from Pins to PCIe

Issue	When MDIO access is switched from external MDIO pins to MDIO through PCIe, an error may occur. If the last access through external MDIO pins is a write, any subsequent register access through PCIe times out and is unsuccessful.
Workaround	Always close the MDIO transaction through external pins with an MDIO read. Any register can be read.
Silicon Revisions Affected	All
Resolution	There is no plan to fix this issue in silicon. Use SDK 1.4.14 or later.

Chapter 25: Ordering Information

25.1 Part Number Decoding

Table 77 shows how the ordering part number is decoded. Contact your sales representative for device availability prior to ordering.
The only available part numbers are listed in Table 78.

Table 77: Ordering Part Number Constituents

Parameters	Constituents
Part Numbers	BCM52311
Revision	A1

Temperature	E = Extended –40°C ambient (T_a) to +105°C junction temperature (T_j)
	H = High 0°C ambient (T_a) to +105°C junction temperature (T_j)

RoHS	H = RoHS 6
Package	0 = 35 mm; 1 = TBD
Mode	H = Heterogeneous
Capacity	1 = 40 Mb/256 Mb 2 = 80 Mb/512 Mb 3 = 40 Mb/512 Mb 6 = 10 Mb/256 Mb 7 = 20 Mb/512 Mb
SerDes	2 = 12.5 Gb/s 7 = 27.34375 Gb/s
Frequency	4 = 750 MHz 5 = 900 MHz
Hosts	S = Single; D = Dual

NOTE: The only available part numbers are listed in Table 78.

Table 78: New Ordering Part Numbers

Part Numbers	Rev	Temperature	RoHS	Package Size	Mode	Capacity	SerDes (Gb/s)	Frequency	Hosts
BCM52311A1EH0H275S	A1	E	H	0	H	2	7	5	S
BCM52311A1EH0H774S	A1	E	H	0	H	7	7	4	S
BCM52311A1HH0H174S	A1	H	H	0	H	1	7	4	S
BCM52311A1HH0H175S	A1	H	H	0	H	1	7	5	S
BCM52311A1HH0H274D	A1	H	H	0	H	2	7	4	D
BCM52311A1HH0H274S	A1	H	H	0	H	2	7	4	S
BCM52311A1HH0H275D	A1	H	H	0	H	2	7	5	D
BCM52311A1HH0H275S	A1	H	H	0	H	2	7	5	S
BCM52311A1HH0H324S	A1	H	H	0	H	3	2	4	S
BCM52311A1HH0H325D	A1	H	H	0	H	3	2	5	D
BCM52311A1HH0H375D	A1	H	H	0	H	3	7	5	D
BCM52311A1HH0H375S	A1	H	H	0	H	3	7	5	S
BCM52311A1HH0H625D	A1	H	H	0	H	6	2	5	D
BCM52311A1HH0H674S	A1	H	H	0	H	6	7	4	S
BCM52311A1HH0H675S	A1	H	H	0	H	6	7	5	S
BCM52311A1HH0H774S	A1	H	H	0	H	7	7	4	S

Appendix A: Acronyms and Abbreviations

Table 79 lists the acronyms and abbreviations used in this document.

For a more complete list of acronyms and other terms used in Broadcom documents, go to: http://www.broadcom.com/press/glossary.php.

Table 79: Acronyms and Abbreviations

Term	Description
ACE	Access Control Entry
ACL	Access Control List
BA	Base Address
BCR	Block Configuration Register
BDPS	Billion Decisions Per Second
BMR	Block Mask Register
CB	Context Buffer
DCR	Device Configuration Register
ECC	Embedded Error Correction Circuitry
EEI	Energy Efficient Interlaken
HPM	Highest Priority Match
ILKN	Interlaken
KPU	Key Processing Unit
LP	Low Power Mode
LTR	Logical Table Register
LUT	Lookup Table
MDC	Management Data Clock
MMD	MDIO Manageable Devices
MSB	Most Significant Bit
ODT	On Die Termination
ROP	Records-Over-Packet
STA	Station Management Entity
UDA	User Data Array
UI	Unit Interval

Revision History

52311-DS129; January 15, 2020

Updated:

- Ordering Information – Updated ordering part number information.

52311-DS128; January 15, 2019

Updated:

- Table 70, Absolute Maximum Ratings

52311-DS127; June 21, 2018

Updated:

- ERROR Record

Added:

- CMP Instruction
- CMP Response Format

52311-DS126; March 8, 2018

Updated:

- Errata

52311-DS125; November 6, 2017

Updated:

- Ordering Information

52311-DS124; August 31, 2017

Added:

- Errata

52311-DS123; August 21, 2017

Updated:

- “Electrical Features” on page 12
- Table 33: “Miscellaneous Pin Descriptions,” on page 66
- Table 34: “Test, Power, and Ground Pin Descriptions,” on page 67
- Table 133: “DC Electrical Specifications,” on page 193

52311-DS122; July 31, 2017

Updated:

- Table 67: “JTAG AC Timing Parameters,” on page 108

52311-DS121-R; July 27, 2017

Updated:

- Table 66: “Absolute Maximum Ratings—ESD Protection,” on page 107

52311-DS120-R; June 07, 2017

Updated:

- Table 138: “New Ordering Part Numbers,” on page 211

52311-DS119-R; May 31, 2017

Updated:

- Section 25: “Ordering Information,” on page 210

52311-DS118-R; February 16, 2017

Updated:

- Table 137: “Ordering Part Number Constituents,” on page 209

52311-DS117-R; December 21, 2016

Updated:

- Context buffer information

52311-DS116-R; December 09, 2016

Updated:

- Maximum Pressure Specification

52311-DS115-R; November 16, 2016

Updated:

- Table 111: “Ordering Part Number Constituents,” on page 171