SECTION 4
BURST BUFFER

The burst buffer module consists of the burst buffer controller (BBC) and the instruction memory protection unit (IMPU).

The BBC delivers the RCPU instruction fetch accesses from the instruction bus onto the U-bus. It utilizes the full U-bus pipeline and a special page access attribute in order to take full advantage of the U-bus bandwidth. It can handle both burstable and non-burstable external memories as well as non-burstable internal memories (flash EEPROM, SRAM).

Code compression features are only available on the MPC556. The MPC556 utilizes a version of code compression / decompression which is called “Phase A”. Phase A code compression / decompression is described in this manual. Future parts may have a different type of code compression. The BBC also contains the functional module which is called the instruction code decompressor unit (ICDU). The ICDU is responsible for on-line (previously compressed) instruction code decompression in the “Decompression-ON” mode. In the “Decompression-OFF” mode, the ICDU is bypassed and the BBC is in normal function.

The IMPU allows the memory to be divided into four regions with different attributes, as well as a default global region (for memory space that is not included in either of the two regions). Each of the two regions can be of size four Kbytes to four Gbytes. Overlap between regions is allowed.

The IMPU includes registers that contain the following information: region base address, region size and the region’s access permissions. For each access (from the processor to the memory), the IMPU finds which region matches the address. If more than one region matches, the region with the lowest index is chosen. If no region is matched, the global region is chosen.

The IMPU compares the attributes of the access from the processor to the attributes of the appropriate region. If the access is allowed, the proper signals are sent to the BBC. If the access is not permitted, an interrupt is sent to the processor.

The IMPU does not support address translation. The effective fetch address issued by the processor is the one that is transferred to the U-bus.

4.1 Burst Buffer Block Diagram

Figure 4-1 is a block diagram of the burst buffer.

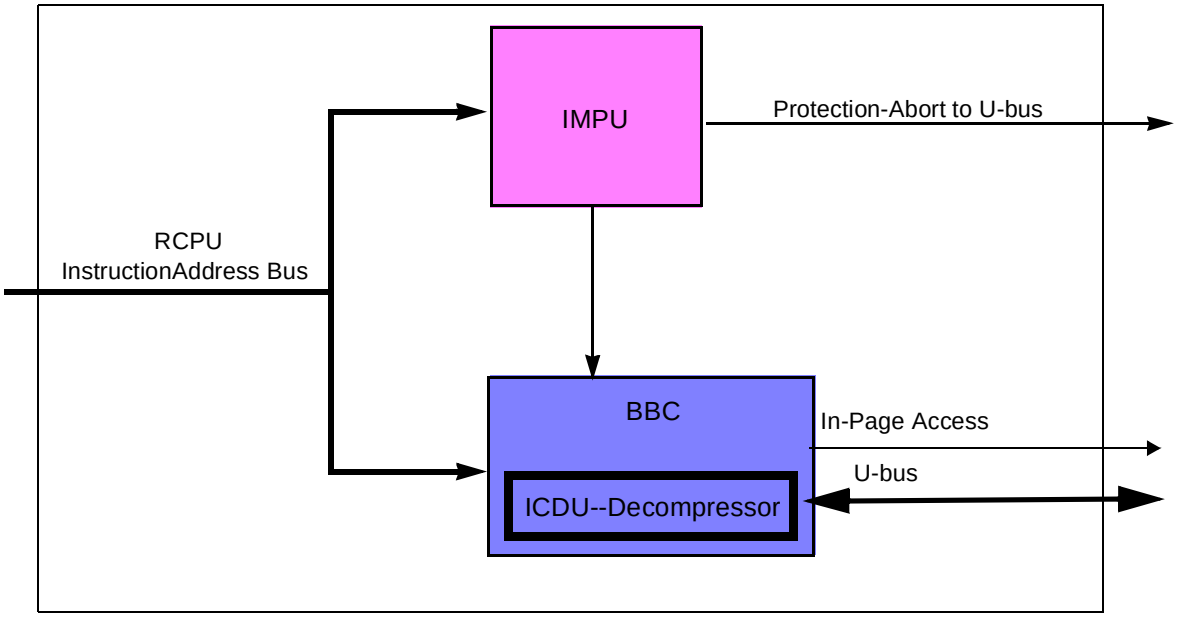

Figure 4-1 Burst Buffer Block Diagram

4.2 Burst Buffer Features

The BBC offers the following features:

- Supports pipelined access to internal memory and burstable access to the external memory.
- Supports the de-coupled interface with the RCPU instruction unit.
- Serves as parked master on the U-bus, resulting in zero clocks delay for RCPU fetch access to cross to the U-bus.
- Full utilization of the U-bus pipeline for fetch accesses.
- Tightly interfaced with L2U interface module, taking advantage of full U-bus bandwidth and back-to-back accesses.
- Supports program trace and show cycle attributes.
- Supports special attribute for debug port fetch accesses.
- Is programmed using the MPC555 / MPC556 **mtspr/mfspr** instructions to/from implementation specific special-purpose registers.
- Designed for minimum power consumption.

The ICDU offers the following features:

- Instruction code on-line decompression based on a fixed vocabulary (bounded Huffman) algorithm.
- No need for address translation between compressed and non-compressed address spaces — ICDU provides “next instruction address” to the RCPU
- Instruction decompression takes one clock cycle.
- Code decompression is pipelined.
- No performance penalty during sequential program flow execution

— Minimal performance penalty due to change of program flow execution
- Two operation modes are available: “Decompression ON” and “Decompression OFF”. Switch between compressed and non-compressed user application software parts is possible.

The IMPU has the following features:

- Four regions in which the base address and size can be programmed.
- Region sizes of four Kbytes up to four Gbytes (in powers of two) can be programmed. (A region must start on the specified region size boundary.)
- Overlap between regions is allowed.
- Each of the four regions supports the following attributes:
  — Access protection (use r/ supervisor fetch or no access).
  — Guarded attribute (causes an interrupt in case of fetch try).
  — On / off option
  — Compressed / non-compressed.
- Global region entry declares the default access protection and guarded attributes for all memory areas not covered by the four regions:
- Interrupt generated upon access violation or fetch from guarded region.
- MPC555 / MPC556 MSR[IR] bit controls MPU protection.
- Programming is done using MPC555 / MPC556 **mtspr/mfspr** instructions to/from implementation specific special purpose registers.
- Designed for minimum power consumption.
- Compressed/non-compressed region with enable/disable option.
- Special reset exception vector for “Decompression ON” mode.

4.3 Instruction VocabularyBased Compression Model Main Principles

4.3.1 Compression Model Features

- Implemented for PowerPC architecture
- Up to 30% code size reduction
- No need for address translation tables
- No changes in the CPU architecture
- Compression is done off line by a special “compressor” tool, using a fixed vocabulary instruction based algorithm, optimized for the PowerPC instruction set.
- Decompression is done at run-time with special hardware.
- Optimized for cache-less systems:
  — Highly effective in system solutions for low cache-hit ratio environment and for systems with fast embedded program memory
  — Deterministic program execution time
  — No performance penalty during sequential program flow execution
  — Minimal performance penalty due to change of program flow execution
- Switch between compressed and non-compressed user application sections is possible. (Compressed subroutine can call non-compressed one and be called from non-compressed portion of user application)

- Slight changes in the core and existing RISC development tools — compilers, simulators, manually coded libraries.
- Compressed address space is up to four Megabytes (4 Mbytes).
- Branch displacement from its target:
  - — Conditional branch displacement is up to two Kbytes (2 Kbytes).
  - — Unconditional branch displacement is up to two Mbytes (2 Mbytes).

NOTE

Branch displacement is hardware limited. The compiler can enlarge the branch scope by creating branch chains.

4.3.2 Model Limitations

No address arithmetic is allowed, because the address map changes during compression and no software tool can identify address arithmetic structures in the code.

4.3.3 Vocabulary Based Instruction Compression Algorithm

The code compression algorithm is based on creating vocabularies of frequently appearing PowerPC RISC instructions or instruction halves and replacing these instructions with pointers to the vocabularies.

Compressed and bypass field lengths may vary. An example of compressed code is shown in **Figure 4-2**.

Compression of the instructions in a vocabulary may be in one of the following modes.

1. Compression of the whole instruction into four vocabulary byte pointers. The four compacted bytes may start on any bit location. Four of the decoded bits and another bit for starting from the left or right side of the address location determine the bit location for the byte start
2. Compression of a combination of the instruction's bytes into vocabulary pointers and bypass of the other byte(s). Bypass is the placing of the field's uncompressed instruction information into the compressed code.
3. Bypass of the whole instruction. No compaction permitted.

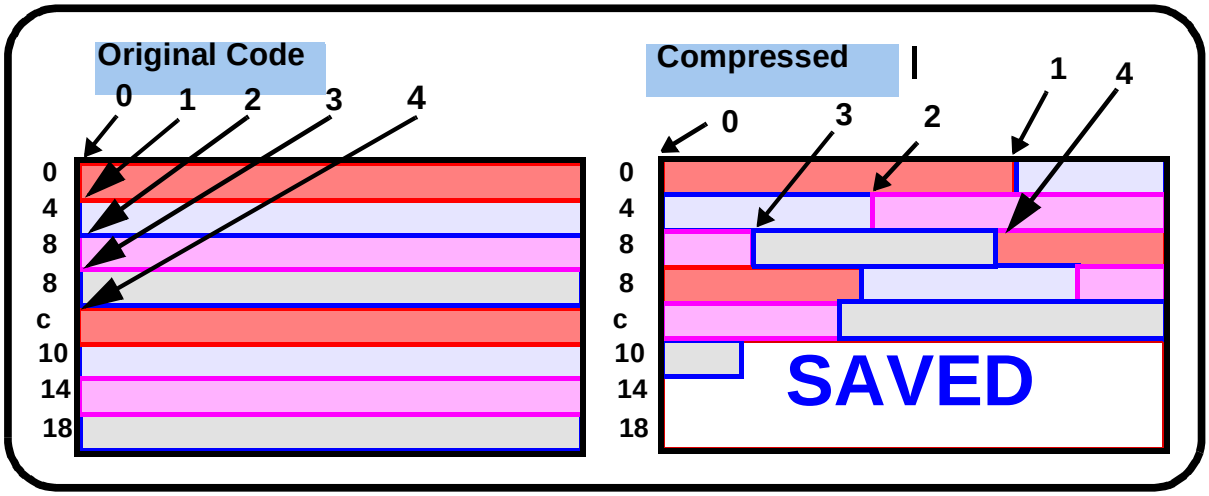

Figure 4-2 Example of Compressed Code

Each instruction is divided to four bytes, marked X1, X2, X3 and X4. For each such byte a separate (Huffman coding) vocabulary is generated, marked Tx1, Tx2, Tx3 and Tx4. Once compressed, each instruction yields four symbols (corresponding to the X1, X2, X3, and X4 input bytes). Therefore, in order to compress a given code, four vocabularies are required. This partitionong produced a better comression ratio.

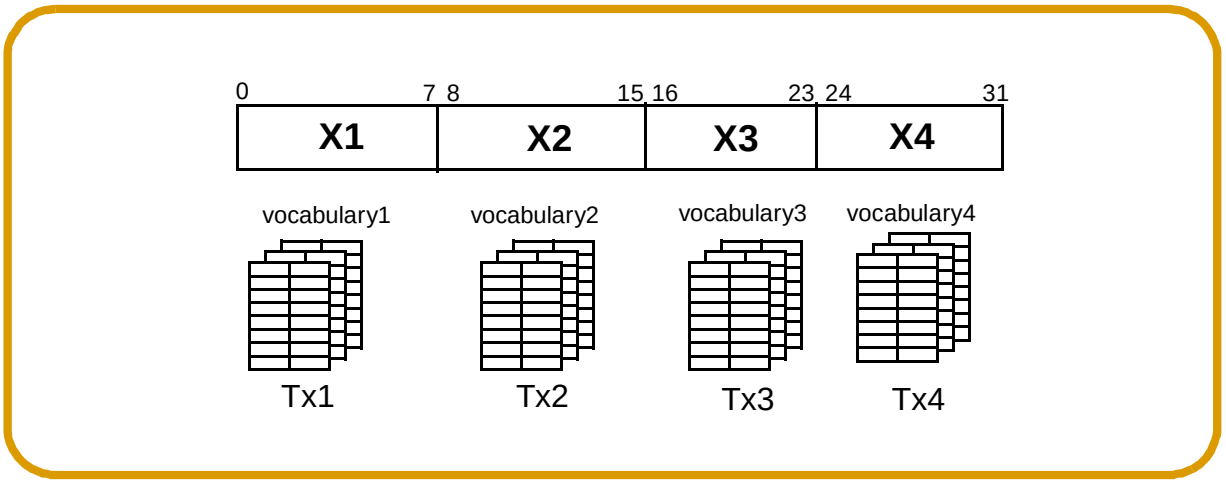

Figure 4-3 Instruction Coding

4.3.4 Memory Organization

In order to enhance performance, the logic is built to decode two halves of an instruction in parallel. The memory is arranged to support this as two streams of compressed symbols: the left stream for the compressed symbols of X1 and X2 bytes, and the right stream for the compressed symbols of X3 and X4 bytes.

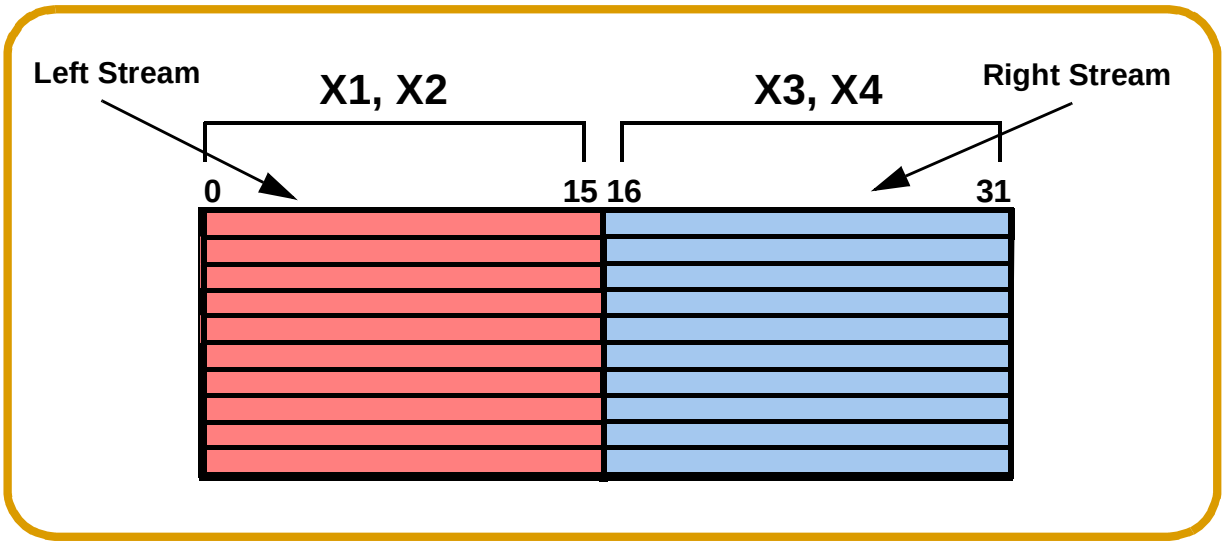

Figure 4-4 Two Streams Memory Organization — Before Compression

In **Figure 4-4**, each left and right stream line includes two original bytes of the instruction. **Figure 4-5**, shows the memory after compressed streams have been put into it.

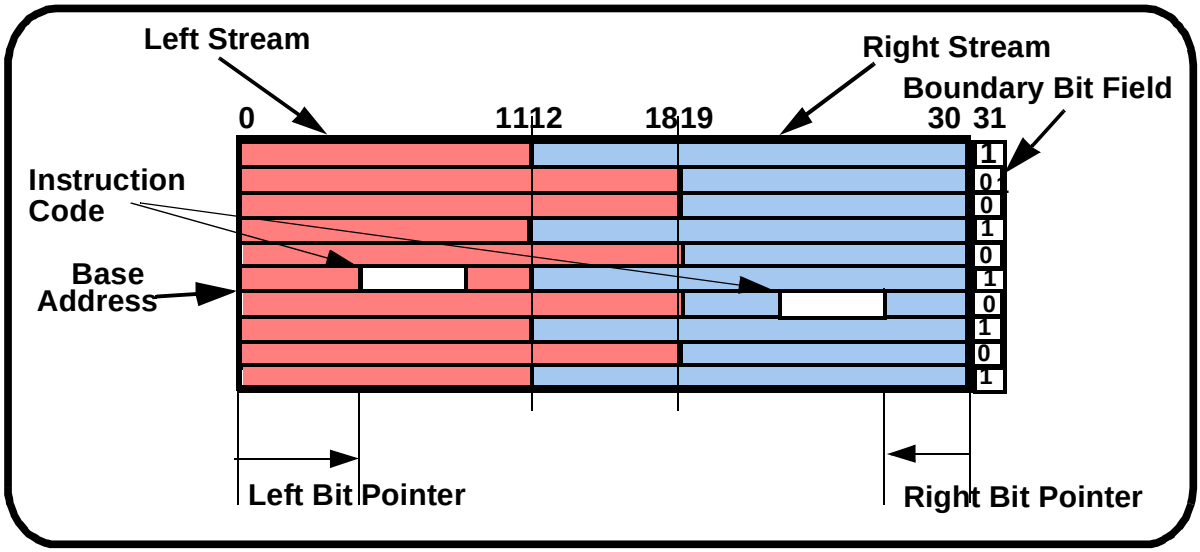

Figure 4-5 Two Streams Memory Organization — After Compression

The compiler will set the left and right stream boundary at either bit 12 or bit 19. This will be determined by the most efficient placement of compressed instruction code. The boundary will be placed between bits 11 and 12 if bit 31 is equal to one. The boundary will be placed between bits 18 and 19 if bit 31 is equal to zero. The original right and left streams may span an adjacent base addresses before or after each other. This fact will also determine the placement of the boundary bit field.

Each stream line may include a variable number of compressed symbols, depending on how well the bytes in the original stream were compressed.

The decompressor has to maintain two bit pointers (left and right) in order to have access to the start location of any instruction's half.

The decompressor maintains tracking of the base address, to start fetching from the next address in the memory.

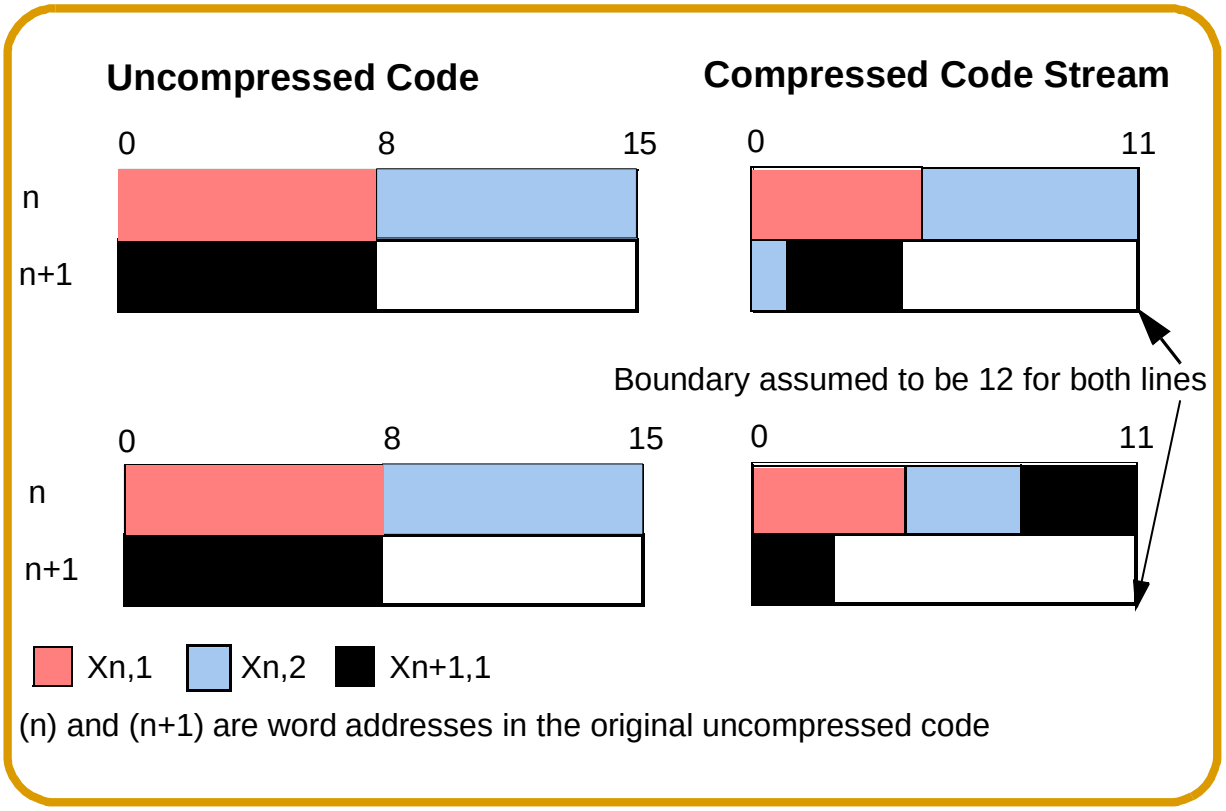

Figure 4-6 Examples of Compressed Symbols Layout

4.3.5 Compressed Code Address Format

The format of the compressed code in memory requires special addressing. The Decompressor module is responsible for generating compressed code addresses. The compressed instruction stream may start on any of the 32 bits. Thus, five bits are needed to locate such instruction inside a memory word. The instruction address in “Decompression ON” mode consists of a 20-bit word pointer for the base address, bits 20 and 21 to show the relation between the left and right streams, and two 5-bit instruction pointers. This is known as the two-pointer address form. See **Figure 4-7**.

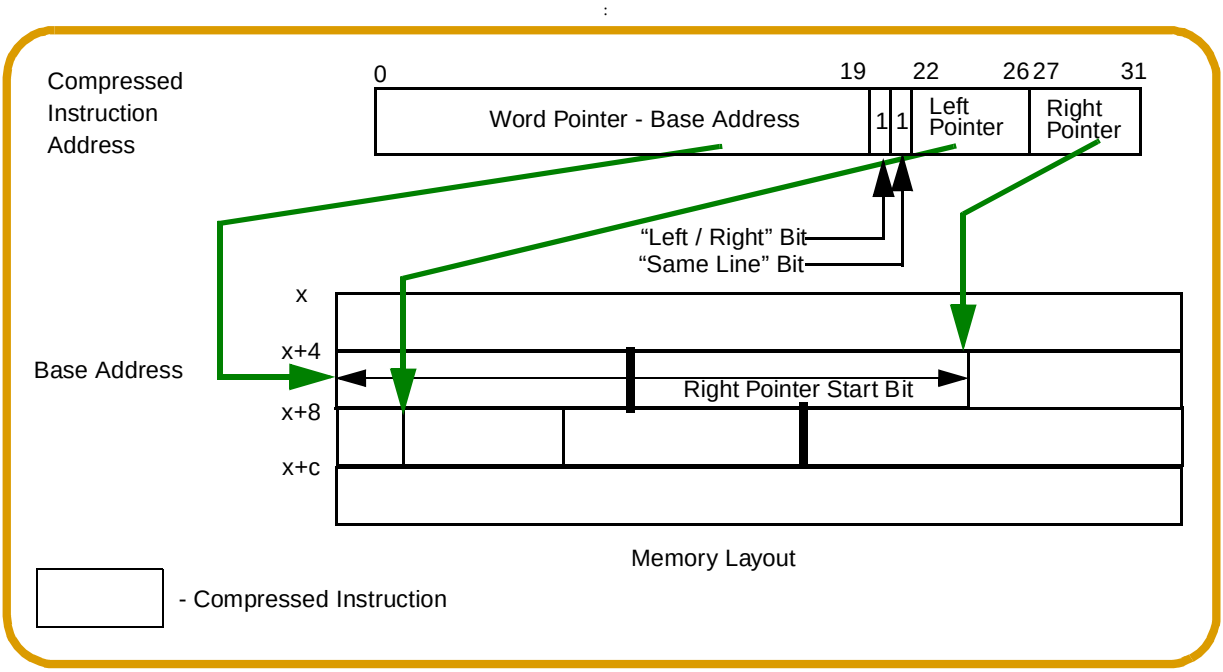

Figure 4-7 Compressed Address Format

The base address contains the lowest word address of physical memory where the instruction resides.

The “left / right” bit, bit number 20, indicates which instruction stream side (left or right) resides in the memory word location being pointed to by the base address. A zero “0” for bit 20 will indicate that the left side is resident in the base address location. A one “1” for bit 20 will indicate that the right side is resident in the base address location. The instruction stream side not pointed to will reside in the following address location.

The “same line” bit, bit 21, reflects the relative location of the two side streams for the instruction. If bit 21 is zero “0”, both left and right streams are located at the base address location. In this case, bit 20 has no meaning and is a “don’t care” value of X. If bit 21 is one “1”, then the two parts of the instruction are located in different address word locations (one at “x” base address, the other at “x+4”).

Figure 4-8 illustrates the three possible cases for bits 20 and 21.

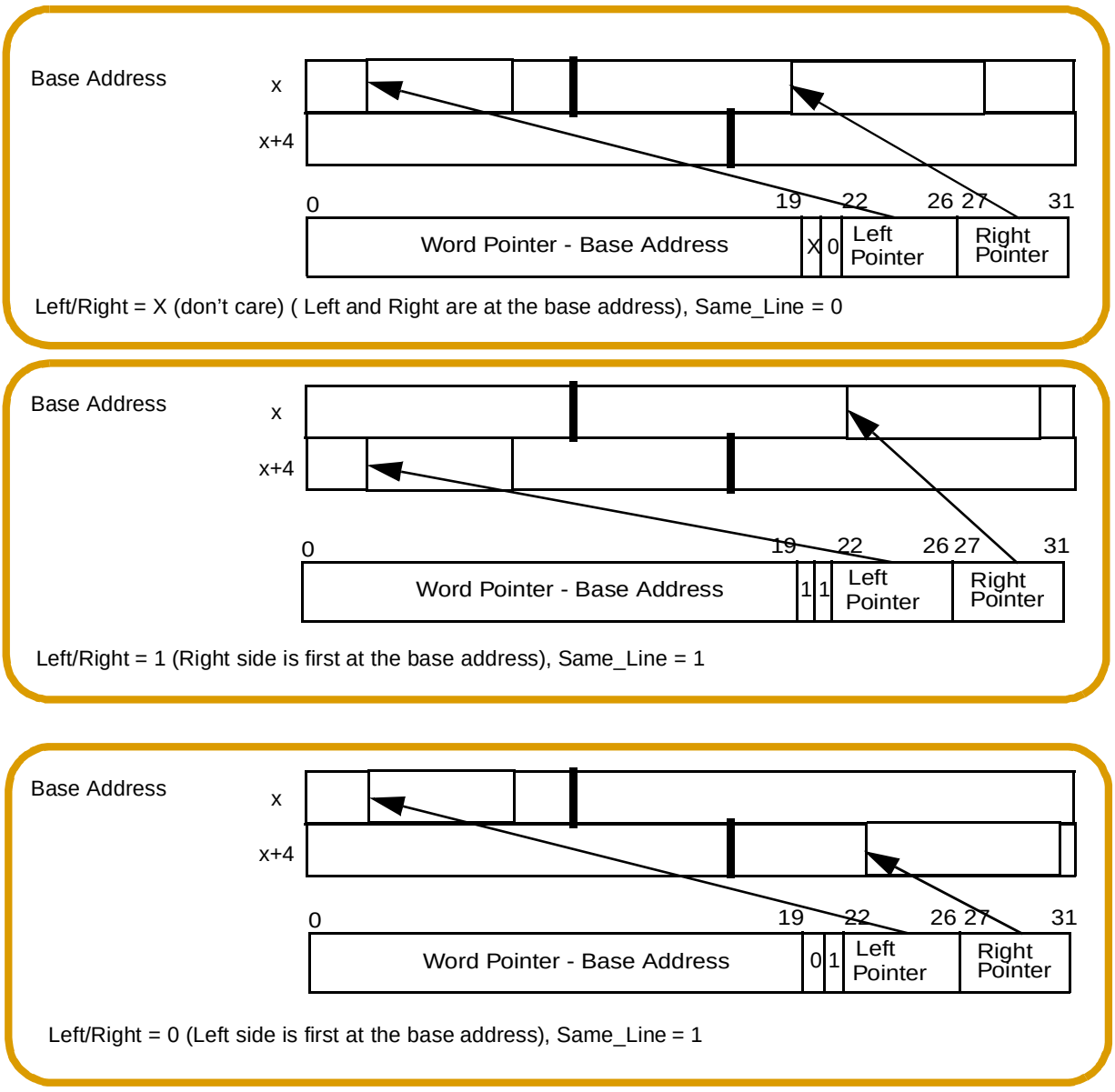

Figure 4-8 Examples of Instruction Layout in Memory

4.3.6 Compressed Address Format – Direct Branches

The one pointer format is used for the conditional and unconditional direct branches. **Figure 4-9** illustrates the one pointer format. The word pointer for the unconditional branch has nineteen bits (the lower two-byte bits are ignored). This will yield an unconditional branch displacement limit of two Mbytes. The word pointer for the conditional

branch has nine bits. This will yield a conditional branch displacement limit of two Kbytes.

When a change of flow occurs, the sequencer of the PPC core will issue the new address in compression mapped format. The address extractor unit of the BBC generates the direct branch address format to internal memory.

Indirect branches use the regular two pointer format described in the previous section.

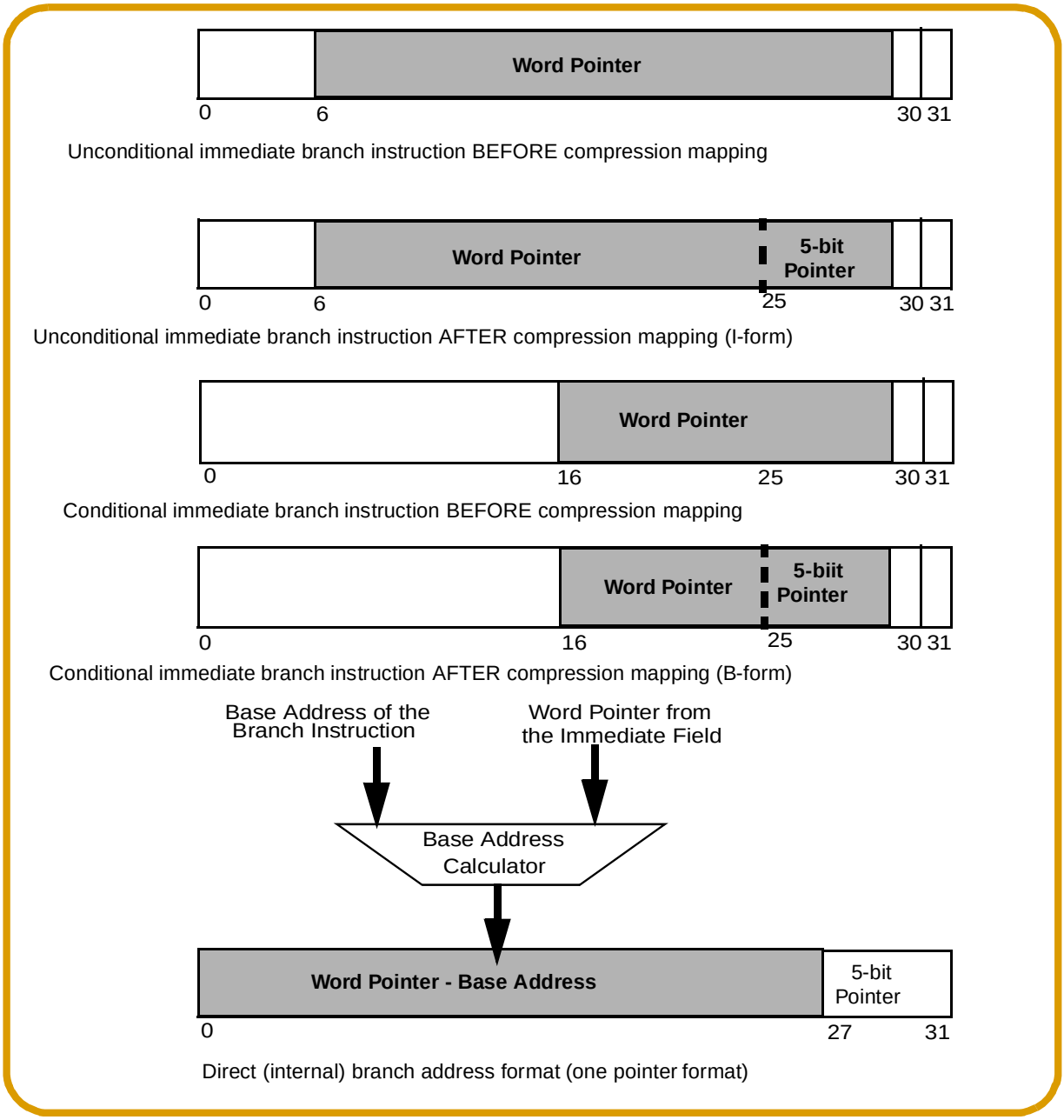

Figure 4-9 Generating Compressed Code Address for PowerPC Direct Branches

An instruction in memory which will serve as the target of a branch will have a label attached. The label provides the needed pointer to the other half of the branch target instruction. The label token will be skipped in normal sequential operation. The label has three parts. First, the label prefix character (which is skipped by the decompressor). Second, a 5-bit pointer to the second half of the instruction. Third, a bit which indicates the location of the second instruction half on the same line or the next line.

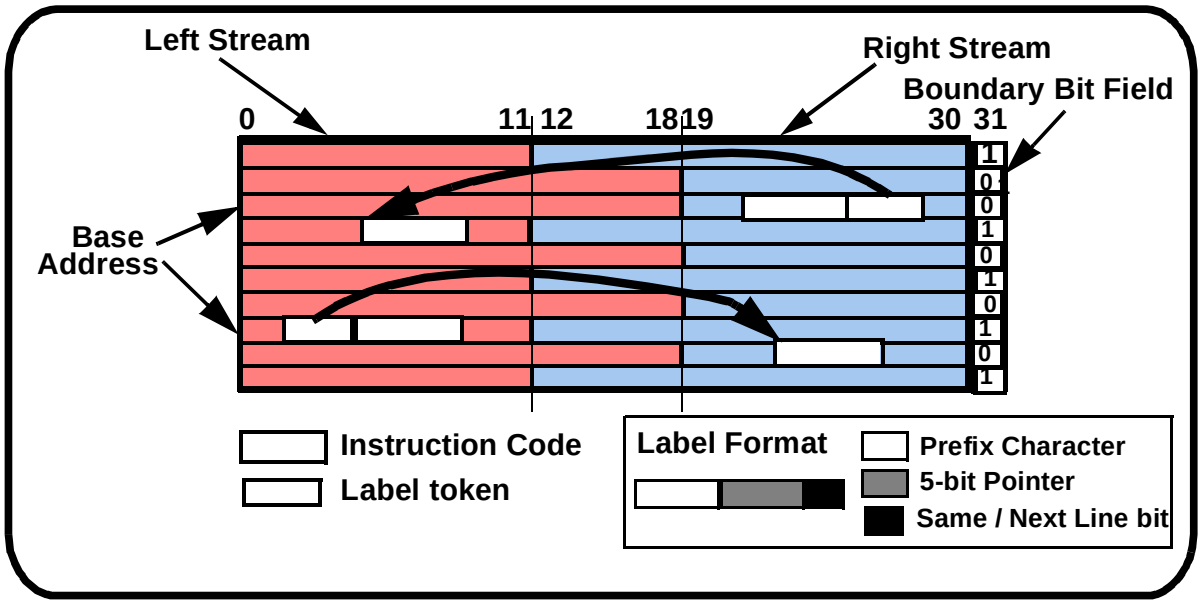

Figure 4-10 Extracting Direct Branch Target Address in the Decompressor

4.3.7 Compressed Address Format – Indirect Branches

Indirect branches use the regular two pointer format described above. The indirect branch destination address is copied without any change from one of the following registers:

- LR
- CTR
- SRR0

See the PowerPC™ ***RCPU User's Manual*, RCPURM/AD**, for more details.

4.3.8 Compression Process

The compression process is implemented by the following steps (See **Figure 4-11**):

- User code compilation/linking
- User application code compression by software compression tool.

The compiler will add a few simple "hooks" to the compiled code which will make compression possible. Compiled code will be generated in the "elf" format for code

compression purposes. The resulting uncompressed elf code (with compression hooks) will load and run like any other elf code.

The software compression tool compresses the elf code (x.elf) and produces a compressed elf code (x.elf.sqz). The system sees the compressed elf code as regular elf formatted code for purposes of loading into hardware (when programming the flash). However, the decompression module must now be used to run the compressed elf code.

Figure 4-11 illustrates the several steps for generating compressed executable code. Once generated, this code can be loaded into flash or SRAM (internal or external) .

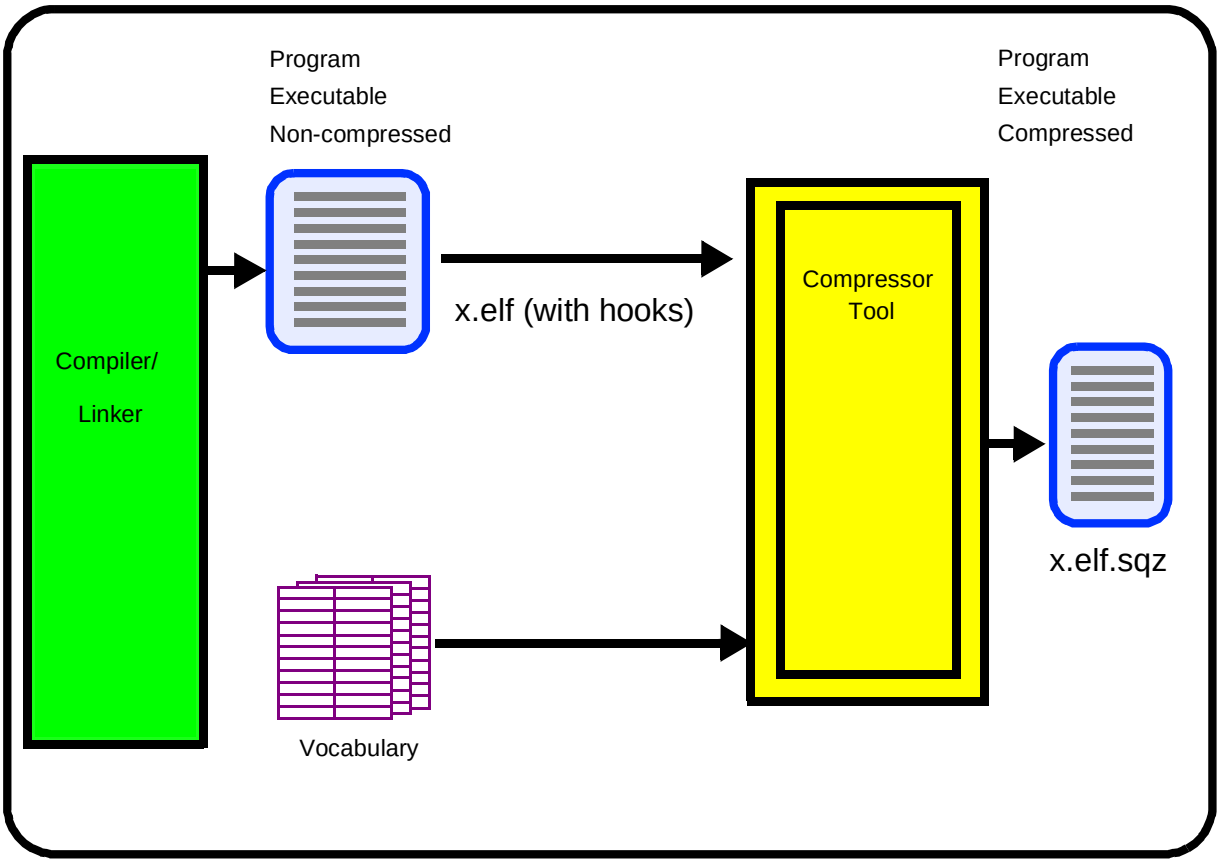

Figure 4-11 Code Compression Process (Phase A)

The compression tool replaces regular PowerPC instructions by their “compressed” representation which contain fewer data bits. The compressed data bit representation is contained in the vocabulary. The vocabulary is structured into a binary bounded Huffman code tree. This method has the result of the first instructions being represented by fewer bits. Further instructions require more bits for unique decoding. Therefore, the instructions that occur most in code should be represented earlier in the vocabulary structure. This would produce the most condensed code. A statistical study was made of typical application code. The existing vocabulary is fixed for Phase A

code compression, and is a result of the statistical study. **Figure 4-12** illustrates the binary decode tree for specific instructions.

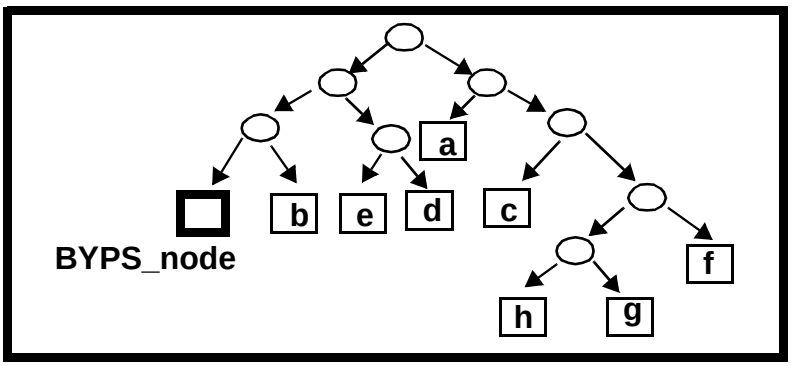

Figure 4-12 Bounded Huffman Code Tree

In **Figure 4-12**, instruction “a” would require two bits. The bypass node would require four bits. The bounded form of the Huffman code tree is limited in size for implementation into hardware. The largest compressed instruction is 36 bits — four bits for the bypass mode plus the normal uncompressed 32-bit instruction.

4.3.9 Decompression

- The instruction code is stored in the memory in the compressed form
- The decode vocabulary is stored in the burst buffer controller (BBC).
- The decompression is done on-line by the dedicated decompressor unit in the BBC.
- Decompression flow: (See **Figure 4-11**)
  - — RCPU provides a **“bit aligned COF[1] address”** to the BBC.
  - — ICDU:
    - Converts COF address to **“word aligned physical address”** to access the memory
    - Fetches the compressed instruction code data from the memory, decompresses it and delivers **“non-compressed instruction code”** together with the bit aligned **“next instruction address”** to the RCPU, that uses it for subroutine and exceptions handling.
    - When instructions are running without a COF, the next instruction is prefetched and decoded in the current cycle. This eliminates any delays from code compression during regular sequential (non-COF) operation.

1. COF = Change of Flow

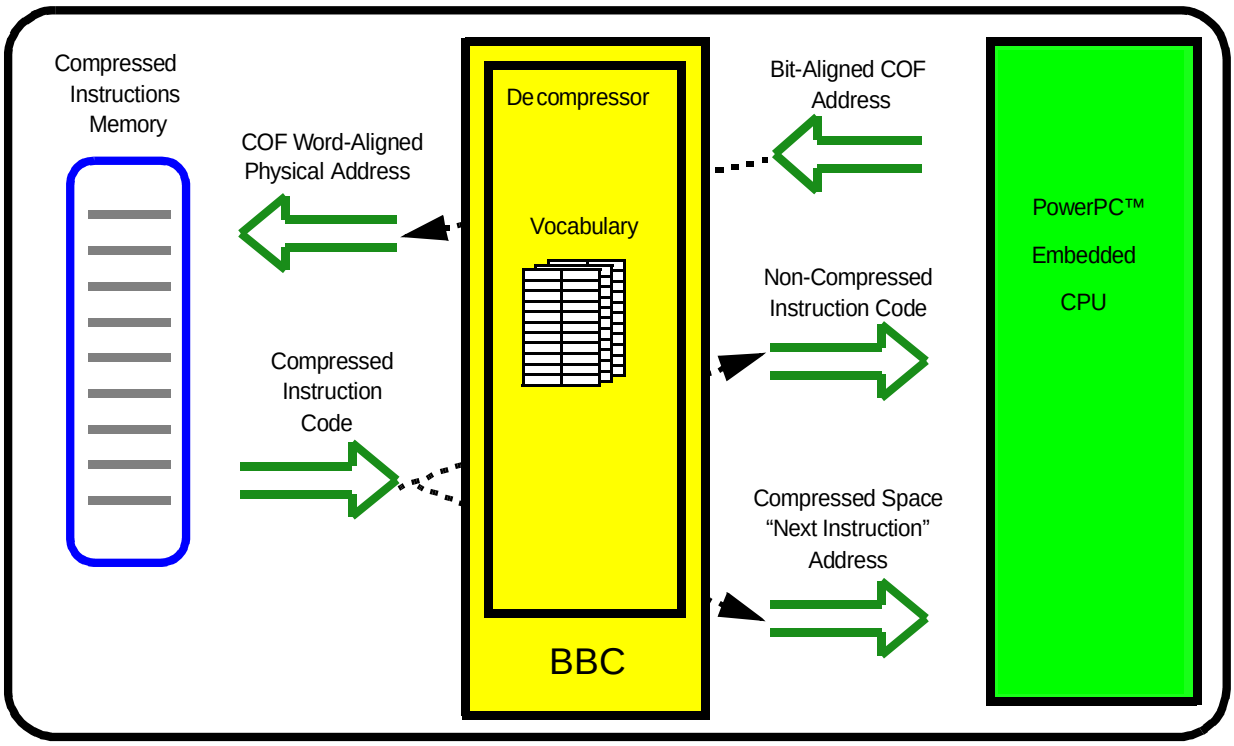

Figure 4-13 Code Decompression Process

4.3.10 Compression Environment Initialization

At power on reset (POR) or with a hard reset, the default settings will be activated unless the configuration word inputs override these defaults. The compression mode configuration data to be programmed is supplied by the user software in the flash.

The hard reset configuration word (described in **SECTION 7 RESET**) has two bits which control the code compression mode. Bit 21 enables code compression when equal to "1" and disables code compression when equal to "0". Bit 22 defines the exception table code as either compressed with a value of "1" or non-compressed with a value of "0".

4.4 Modes Of Operation

The burst buffer module can operate in the following modes:

- Normal
- Slave
- Reset
- Debug
- Standby
- Burst

The modes of operation are described in the following paragraphs.

4.4.1 Normal Operation

During normal operation, the burst buffer module transfers fetch accesses from the CPU to the U-bus. When a new access is issued by the CPU, it is transferred in parallel to both the IMPU and the BBC. The IMPU compares the address of the access to its region programming. The BBC determines whether the access can be immediately transferred to the U-bus. If not, it requests the U-bus for the next clock.

Each new BBC U-bus access is accompanying by the burst request attribute. If burstable access is enabled, the BBC performs a burst access; otherwise, it performs a single access.

If the IMPU detects an access violation, it does the following:

- Cancels the request that was forwarded to the BBC
- Informs the RCPU core that the requested address generated an exception

If the required address contains show cycle or program trace attributes, the BBC delivers the access to the U-bus even if the request is cancelled (due to the exception it caused).

The BBC forwards show cycle, program trace and debug port access attributes accompanying the CPU access along with the U-bus access.

4.4.2 Slave Operation.

The burst buffer module is operating as a U-bus slave module when the instruction memory protection unit (IMPU) registers are accessed by the user in order to be programmed. This programming is done using the **mtspr /mfspr** instructions.

4.4.3 Reset Operation

On reset the BBC goes to an idle state, and all pending U-bus accesses are ignored. The IMPU goes to a disabled state in which all memory space is accessible to both user and supervisor.

4.4.4 Debug Mode Operation

When the CPU is in debug mode, fetch accesses are attached with a special attribute. If this attribute is asserted, the BBC must initiate not-burstable accesses to the debug port.

4.4.5 Standby Mode Operation

In this low-power mode the CPU stops issuing further accesses. The BBC clocks are turned off, and the BBC enters a power-save state. When the low-power mode is exited, clocks are activated and a new access from the CPU will activate the BBC.

4.4.6 Burst Operation

The BBC can run burst accesses on the U-bus. Such burst cycles, if forwarded to external memory, are then exported to the EBI as burst cycles (if bursts are enabled by the USIU).

The BE bit defined in **4.6.4 BBC Module Configuration Register (BBCMCR)** determines whether the BBC operates burst cycles or not. Burst requests are enabled only when the BE bit is set.

NOTE

The negated state of the BE bit is useful mainly when the RCPU core runs in serialized mode. (Refer to **21.7.6 I-Bus Support Control Register** for the ICTRL register.)

4.4.7 Error Detection

If the IMPU detects access violation, the following actions must be taken:

1. Cancel the request that was forwarded to the burst buffer controller
2. Inform the RCPU core that the requested address generated an exception

If the required address contains show cycle or program trace attributes, than the BBC delivers the access onto the U-bus even if the request is cancelled (due to the exception it caused).

The way the IMPU notifies the RCPU core for an interrupt is by feeding error information into four bits (1, 3, 4 and 10) in the SRR1 reg ister in the core. Only one bit is set at a time. The exception vector (address) that the core issues for this event is 0xnnn0-1300. The encoding of the status bits is as follows:

- SRR1 = 0
- SRR3 = Guarded storage.
- SRR4 = Protected storage.
- SRR10 = 0

4.5 Exception Table Relocation

The BBC has the ability to relocate the exception table. Exception table relocation is a feature to save memory space in the exception table. See **3.11.5 Exception Vector Table** for normal operation of the exception vector table. This is done by mapping exceptions to be separated by eight bytes instead of 256 bytes (see **Table 4-1**). The relocation feature maps the exception table into the internal memory space of the MPC555 / MPC556 and requries MSR[IP] = 1. This feature is important in multi-MPC555 / MPC556 systems, where more than one MCU can have internal exception tables with the same exception addresses issued by the RCPU.

The relocation feature also saves the wasted space between exception table entries when each exception entry contains only a branch instruction to the exception routine, which is located elsewhere.

If exception relocation is enabled (ETRE bit is set in the BBCMCR), all exception routines (except the reset exception routine) can be controlled to either remain in the lower addresses of the memory (base address + exception offset) BBCMCR[OERC] = 0 or to be relocated to memory (base address + 32 Kbytes) by setting BBCMCR[OERC] = 1. The reset exception routine location is fixed in memory (base address + the reset exception offset) and can not be relocated.

See **4.6.4 BBC Module Configuration Register (BBCMCR)** for programming details.

4.5.1 Exception Table Relocation Operation

When an exception is requested, the CPU initiates a fetch cycle that branches to the exception routine associated with the exception that caused the fetch. The exception addresses are fixed within the RCPU architecture and are 0x100 bytes apart from each other, starting at address 0x0000_0100 or 0xFFF0_0100, depending on the value of the MSR[IP] bit.

If the relocation feature is disabled, the BBC transfers the exception fetch address to the internal bus of the MPC555 / MPC556 with no interference.

In order to activate exception table relocation, the following steps are required:

1. Set the MSR[IP] bit. To set this bit out of reset, set the appropriate bit in the reset configuration word.
2. Set the ETRE bit in BBCMCR register. See **4.6.4 BBC Module Configuration Register (BBCMCR)** for programming details.
3. Program absolute branch instructions at the locations indicated in **Table 4-1** pointing to the desired exception handler routines.

If the relocation feature is enabled, the BBC translates the starting address of the exception routine into the address located at the lowest portion of the internal memory. At that location, the user must insert a series (table) of consecutive branch instructions that point to the appropriate exception routines.

NOTE

These branch instructions must utilize absolute addressing modes of the RCPU (relative branches can not be used).

Thus, the CPU branches twice to reach the appropriate exception routine.

NOTE 1

The eight Kbytes allocated for the exception table can be almost fully utilized. This is possible if the MPC555 / MPC556's address space is *not* mapped to the exception address space — that is, if addresses 0xFFF0_0000 to 0xFFF0_1FFF are not part of the MPC555 / MPC556 address space. In this case, these eight Kbytes can be fully utilized by the compiler, except for the lower 64 words (256 bytes), which are reserved for the exception pointers.

NOTE 2

If the CPU issues any address that falls between two successive exception entries (e.g., 0xFFF0_0104), then an exception is generated to the CPU if exception relocation is enabled. See **4.6.4 BBC Module Configuration Register (BBCMCR)**.

Table 4-1 Exception Addresses Mapping by BBC[1]

Name of Exception	Address Issued by CPU (MSR[IP] = 1)	Mapped Address by Exception Table Relocation Logic (BBCMCR[ETRE] = 1)	
		BBCMCR[OERC] = 0[2]	BBCMCR[OERC] = 1
Reserved	0xFFF0_0000	0x0000	0x8000
System Reset	0xFFF0_0100	0x0008	0x0008[3]
Machine Check	0xFFF0_0200	0x0010	0x8010
Data Storage	0xFFF0_0300	0x0018	0x8018
Instruction Storage	0xFFF0_0400	0x0020	0x8020
External Interrupt	0xFFF0_0500	0x0028	0x8028
Alignment	0xFFF0_0600	0x0030	0x8030
Program	0xFFF0_0700	0x0038	0x8038
Floating Point unavailable	0xFFF0_0800	0x0040	0x8040
Decrementer	0xFFF0_0900	0x0048	0x8048
Reserved	0xFFF0_0A00	0x0050	0x8050
Reserved	0xFFF0_0B00	0x0058	0x8058
System Call	0xFFF0_0C00	0x0060	0x8060
Trace	0xFFF0_0D00	0x0068	0x8068
Floating Point Assist	0xFFF0_0E00	0x0070	0x8070
Implementation Dependant Software Emulation	0xFFF0_1000	0x0080	0x8080
Implementation Dependant Storage Error	0xFFF0_1300	0x0098	0x8098
Implementation Dependant Data Breakpoint	0xFFF0_1C00	0x00E0	0x80E0
Implementation Dependant Instruction Breakpoint	0xFFF0_1D00	0x00E8	0x80E8
Implementation Dependant Maskable External Breakpoint	0xFFF0_1E00	0x00F0	0x80F0
Non-Maskable External Breakpoint	0xFFF0_1F00	0x00F8	0x80F8

NOTES:
1. See **Table 3-21** and **3.11.5 Exception Vector Table** for Exception Relocation Table with ETRE = 0.
2. See **4.6.4 BBC Module Configuration Register (BBCMCR)**
3. The reset exception is NOT affected by OERC.

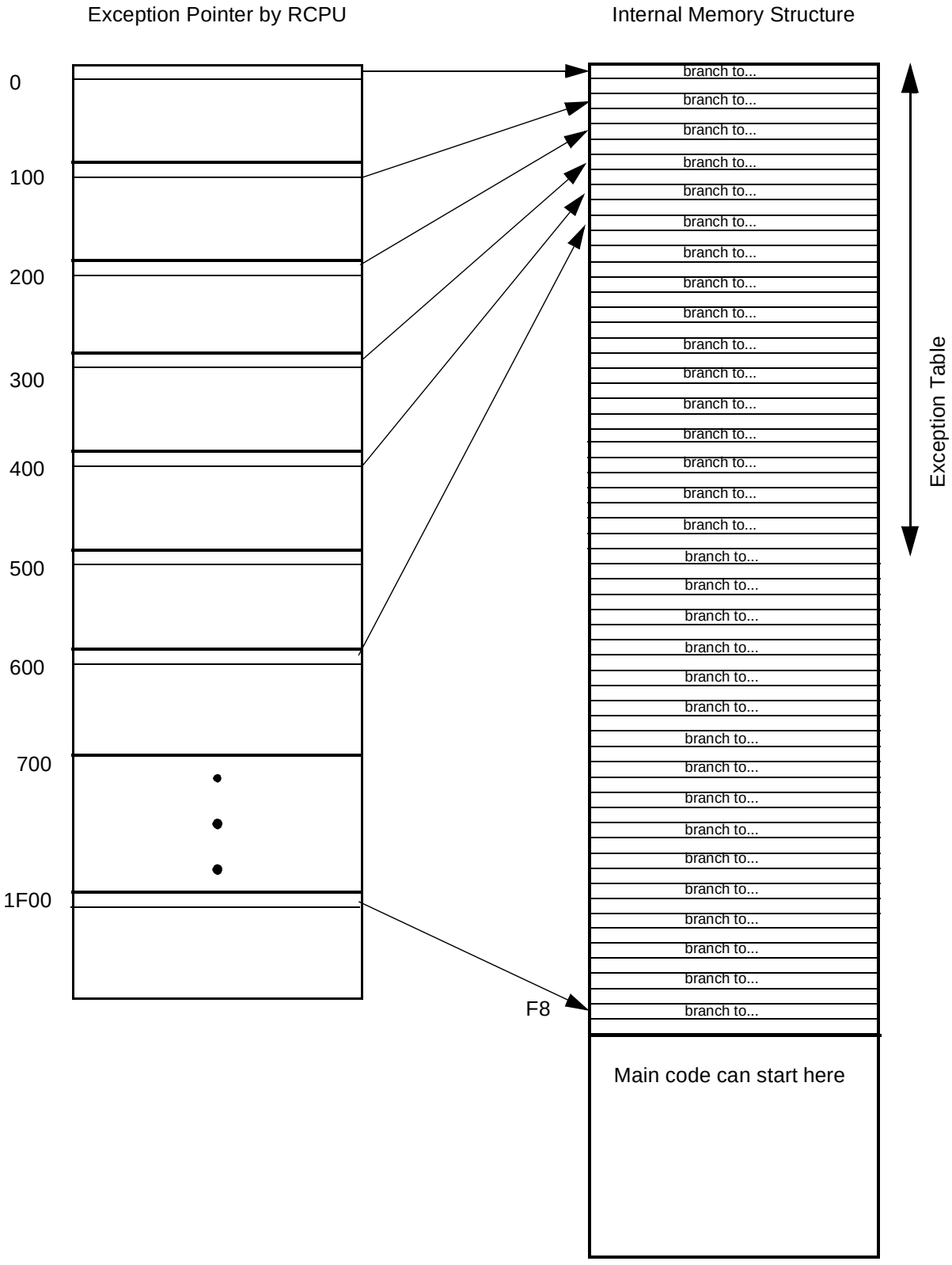

Figure 4-14 Exception Table Entries Mapping

4.6 Burst Buffer Programming Model

The BBC and IMPU module configuration registers are MPC555 / MPC556 special-purpose registers (SPRs). They are programmed with the MPC555 / MPC556 **mtspr/ mfspr** instructions.

All the registers can be accessed in supervisor mode only. The processor generates an exception internally if an attempt is made to access the registers from user mode.

The following 32-bit registers contain the starting address and the size of the region. There is one register for each region.

Table 4-2 Region Base Address Registers RBA[0:1]

Register Name	Address (Decimal)	ub_addr[18:27] (hex)
MI_RBA[0]	784	0x2180
MI_RBA[1]	785	0x2380
MI_RBA[2]	786	0x2580
MI_RBA[3]	787	0x2780

The following registers hold the attributes of the corresponding regions and of the default region. Each of the four MI_RAx registers contains access permission attributes. The MI_GRA (global region attribute) register contains two additional bits to enable each of the MI_RBAx registers.

Table 4-3 Region Attributes Registers

Register Name	Address (Decimal)	ub_addr [18:27] (Hex)
MI_RA[0]	816	0x2190
MI_RA[1]	817	0x2390
MI_RA[2]	818	0x2590
MI_RA[3]	818	0x2790
MI_GRA	528	0x2100

The BBC holds only one register, the BBC module configuration register (BBCMCR).

Table 4-4 BBC Module Configuration Register

Register name	Addr (Decimal)	ub_addr [0:31] (Hex)
BBCMCR	560	0x2110

4.6.1 Region Base Address Registers

MI_RBA[0:3] — Region Base Address Register **SPR 784 – 787**

MSB 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RA															
RESET:															
Unaffected by Reset															

16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	LSB 31
RA				Reserved											
RESET:															
Unaffected by Reset															

Table 4-5 MI_RBA[0:3] Bit Descriptions

Bit(s)	Name	Description
0:19	RA	Region address. This field defines the base address (most significant 20 bits) for the region.
20:31	—	Reserved

4.6.2 Region Attribute Registers MI_RA[0:3] Description

MI_RA[0:3] — Region Attribute Register **SPR 816 – 819**

MSB 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RS															
HRESET															
U	U	U	U	U	U	U	U	U	U	U	U	U	U	U	U

16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	LSB 31
RS				PP		RESERVED			G	CMPR[1]		RESERVED			
HRESET															
U	U	U	U	U	U	0	0	0	U	U	U	U	0	0	0

NOTES:
1. Available only on the MPC556.

Table 4-6 MI_RA[0:3] Registers Bits Description

Bit(s)	Name	Description
0:19	RS	Region size. The region size is a power of two, determined as follows: 0000_0000_0000_0000_0000 — 4 Kbytes 0000_0000_0000_0000_0001 — 8 Kbytes 0000_0000_0000_0000_0011 — 16 Kbytes 0000_0000_0000_0000_0111 — 32 Kbytes 0000_0000_0000_0000_1111 — 64 Kbytes 0000_0000_0000_0001_1111 — 128 Kbytes 0000_0000_0000_0011_1111 — 256 Kbytes 0000_0000_0000_0111_1111 — 512 Kbytes 0000_0000_0000_1111_1111 — 1 Mbyte 0000_0000_0001_1111_1111 — 2 Mbytes 0000_0000_0011_1111_1111 — 4 Mbytes 0000_0000_0111_1111_1111 — 8 Mbytes 0000_0000_1111_1111_1111 — 16 Mbytes 0000_0001_1111_1111_1111 — 32 Mbytes 0000_0011_1111_1111_1111 — 64 Mbytes 0000_0111_1111_1111_1111 — 128 Mbytes 0000_1111_1111_1111_1111 — 256 Mbytes 0001_1111_1111_1111_1111 — 512 Mbytes 0011_1111_1111_1111_1111 — 1 Gbyte 0111_1111_1111_1111_1111 — 2 Gbytes 1111_1111_1111_1111_1111 — 4 Gbytes
20:31	PP[1]	Protection bits: 00: Supervisor — No Access, User — No Access. 01: Supervisor — Fetch, User — No Access. 1x: Supervisor — Fetch, User — Fetch.
22:24	—	Reserved
25	G	Guard attribute for region 0 = Speculative fetch is not prohibited from region. Region is not guarded. 1 = Speculative fetch is prohibited from guarded region. An exception will occur under such attempt.
26:27	CMPR[2]	Compressed Region. x0 = The region in not restricted. 01 = Region is considered a non-compressed code region. Access to the region is allowed only in “Decompression Off” mode. 11 = Region is considered a compressed code region. Access to the region is allowed only in “Decompression On” mode.
28:30	—	Reserved

NOTES:

1. G and PP attributes perform similar protection activities on a region. The more protective attribute will be implied on the region if the attributes programming oppose each other.
2. This bit is available only on the MPC556.

4.6.3 Global Region Attribute Register Description (MI_GRA)

MI_GRA — Global Region Attribute Register **SPR 528**

MSB 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
ENR0	ENR1	ENR2	ENR3	RESERVED											
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	LSB 31
RESERVED				PP		RESERVED			G	CMPR[1]		RESERVED			
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

NOTES:
1. Available only on the MPC556.

Table 4-7 MI_GRA Bit Descriptions

Bit(s)	Name	Description
0	ENR0	Enable region 0 of IMPU 0 = Region 0 is off 1 = Region 0 is on
1	ENR1	Enable region 1 of IMPU 0 = Region 1 is off 1 = Region 1 is on
2	ENR2	Enable region 2 of IMPU 0 = Region 2 is off 1 = Region 2 is on
3	ENR3	Enable region 3 of IMPU 0 = Region 3 is off 1 = Region 3 is on
4:19	—	Reserved
20:21	PP	Protection bits 00 = No supervisor access, no user access 01 = Supervisor fetch access, no user access 10 = Supervisor fetch access, user fetch access 11 = Supervisor fetch access, user fetch access
22:24	—	Reserved
25	G	Guarded attribute for region 0 = Fetch is allowed from guarded region. 1 = Fetch is prohibited from guarded region. An attempted fetch will generate an exception.
26:27	CMPR[1]	Compressed region x0 = The region in not restricted 01 = Region is considered a non-compressed code region. Access to the region is allowed only in "Decompression Off" mode 11 = Region is considered a compressed code region. Access to the region is allowed only in "De-compression On" mode
28:31	—	Reserved

NOTES:
1. Available only on the MPC556.

4.6.4 BBC Module Configuration Register (BBCMCR)

BBCMCR — BBC Module Configuration Register **SPR 560**

MSB 0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
RESERVED															
RESET:															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	LSB 31
RESERVED		BE	ETRE	OERC	EN_ COMP[2]	EXC_ COMP[2]	DECOMP _SC_EN[2]	RESERVED							
RE-SET:															
0	0	0	ID[19][1]	0	0	0	0	0	0	0	0	0	0	0	0

NOTES:
1. Reset value is taken from the indicated bit of the reset configuration word.
2. Available only on the MPC556.

Table 4-8 BBCMCR Bit Descriptions

Bit(s)	Name	Description
16:17	—	Reserved
18	BE	Burst enable 0 = BBC does not request burst accesses 1 = BBC requests burst accesses
19	ETRE	Exception table relocation enable 0 = Exception table relocation is off — the BBC does *not* map exception addresses 1 = Exception table relocation is on — the BBC maps exception addresses to a branch instruction table. Refer to **4.5 Exception Table Relocation**.
20	OERC	Other exceptions relocation control. 0 = All exceptions except reset are mapped to the internal memory base address. 1 = All exceptions except reset are mapped to the internal memory base address + 32 Kbytes.
21	EN_COMP	Enable COMPression – This bit enables the operation of the MPC556 in "Compression ON" mode. The default state is disabled. This bit is read only. 0 = "Decompression ON" mode is disabled. The MPC556 operates only in "Decompression OFF" mode. 1 = "Decompression ON" mode is enabled. The MPC556 may operates with both "Decompression ON" and "Decompression OFF" modes. The bit value is determined by reset configuration word, bit #21.
22	EXC_ COMP	Exception Compression – This bit determines the operation of the MPC556 with exceptions. If this bit is set, the MPC556 assumes that the all exception routines code is compressed; otherwise it is assumed that all exception routines code is not compressed. The reset value is determined by reset configuration word bit #22. 0 = The MPC556 assumes that exception routines are non-compressed 1 = The MPC556 assumes that ALL exception routines are compressed. This bit effects only when EN_COMP bit is set.

Table 4-8 BBCMCR Bit Descriptions (Continued)

Bit(s)	Name	Description
23	DECOMP_SC_EN	DECOMPression Show Cycle ENable - This bit determines the way the MPC556 executes instruction show-cycle. The reset value is determined by configuration word bit #21. For further details regarding show cycles execution in "Decompression ON" mode see **4.3.9 Decompression**. 0 = Decompression Show Cycle does not include the bit pointer. 1 = Decompression Show Cycles includes the bit pointer information on the data bus.
24:31	—	Reserved

NOTE

An ISYNC instruction is required immediately following any write to the BBCMCR.