Section 4. Prefetch Cache

HIGHLIGHTS

This section of the manual contains the following major topics:

Note:	This family reference manual section is meant to serve as a complement to device data sheets. Depending on the device variant, this manual section may not apply to all PIC32 devices. Please consult the note at the beginning of the **"Prefetch Cache"** chapter in the current device data sheet to check whether this document supports the device you are using. Device data sheets and family reference manual sections are available for download from the Microchip Worldwide Web site at: http://www.microchip.com

4.1 INTRODUCTION

This section describes the features and operation of the Prefetch Cache module in the PIC32 device family. Prefetch cache features increase system performance for most applications.

Program Flash Memory (PFM) cache and the Prefetch Cache module increase performance for applications that execute out of the cacheable PFM region by implementing the following features:

- Instruction caching

  The sixteen fully associative lockable 16-byte cache lines supply an instruction every clock, for loops up to 256 bytes long.
- Data caching

  Prefetch cache allows the allocation of up to four cache lines for data storage to provide improved access for PFM-stored constant data.
- Predictive prefetching

  The Prefetch Cache module provides instructions once per clock for linear code even without caching by prefetching ahead of the current program counter, hiding the access time of the PFM.

4.1.1 Additional Prefetch Cache Module Features

The Prefetch Cache module also include the following features:

- Up to four cache lines allocated to data
- Two cache lines with address mask to hold repeated instructions
- Pseudo Least-Recently-Used (LRU) replacement policy
- All cache lines are software-writable
- 16-byte parallel memory fetch
- Predictive instruction prefetch cache

4.2 CACHE OVERVIEW

The Prefetch Cache module is a performance enhancing module included in some processors of the PIC32 family. When running at high-clock rates, Wait states must be inserted into PFM read transactions to meet the access time of the PFM. Wait states can be hidden to the core by prefetching and storing instructions in a temporary holding area that the CPU can access quickly. Although the data path to the CPU is 32 bits wide, the data path to the PFM is 128 bits wide. This wide data path provides the same bandwidth to the CPU as a 32-bit path running at four times the frequency.

There are two main functions that the Prefetch Cache module performs: caching instructions when they are accessed, and prefetching instructions from the PFM before they are needed.

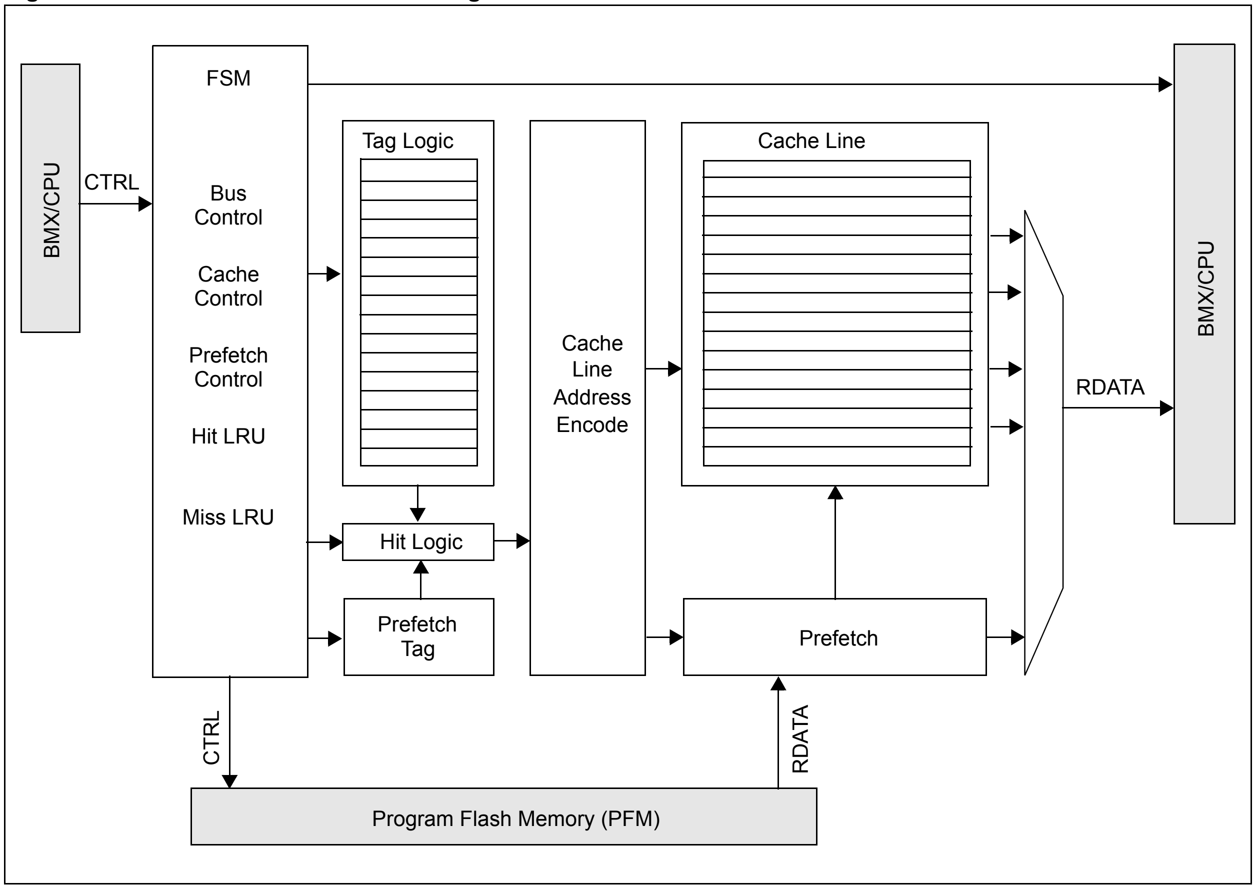
The cache holds a subset of the cacheable memory in temporary holding spaces known as cache lines. Each cache line has a tag describing what it is currently holding, and the address where it is mapped. Normally, the cache lines just hold a copy of what is currently in memory to make data available to the CPU without Wait states.

CPU requested data may or may not be in the cache. A cache-miss occurs if the CPU requests cacheable data that is not in the cache. In this case, a read is performed to the PFM at the correct address, the data is supplied to the cache and to the CPU. A cache-hit occurs if the cache contains the data that the CPU requests. In the case of a cache-hit, data is supplied to the CPU without Wait states.

The second main function of the Prefetch Cache module is to prefetch cache instructions. The module calculates the address of the next cache line and performs a read of the PFM to get the next 16-byte cache line. This line is placed into a 16-byte-wide prefetch cache buffer in anticipation of executing straight-line code.

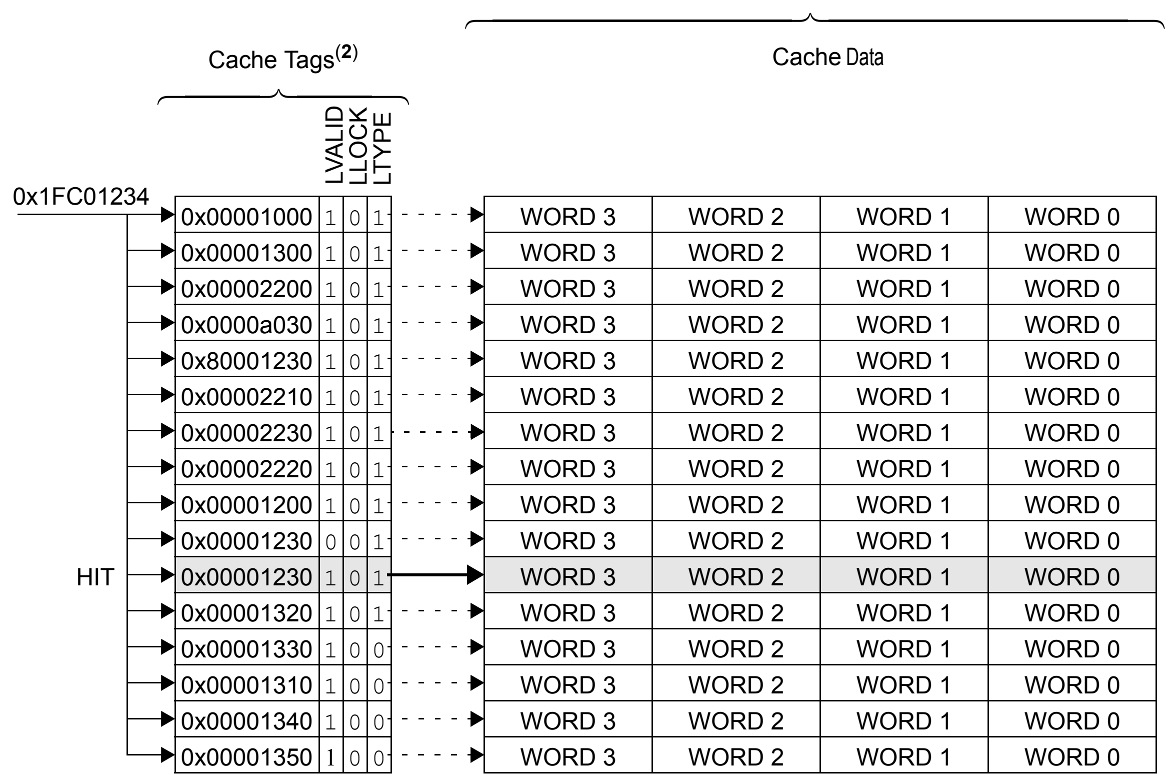
Figure 4-1 shows a block diagram of the Prefetch Cache module. Logically, the Prefetch Cache module fits between the Bus Matrix (BMX) module and the PFM.

Figure 4-1: Prefetch Cache Block Diagram



To illustrate the basic operation of the prefetch cache, Figure 4-2 shows an example of the CPU requesting data from physical address 0x1FC01234. The prefetch cache simultaneously compares this address to all of the tags marked 'valid'. As the shaded entry below has this address, and is marked as valid, this is a cache hit. The proper data word from the data array is then directed to the CPU in a single clock period.

Figure 4-2: Cache Look-up Example[(1)]



Cache Tags	LVALID	LLOCK	LTYPE				
0x00001000	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00001300	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00002200	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x0000a030	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x80001230	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00002210	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00002230	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00002220	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00001200	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00001230	0	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00001230	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00001320	1	0	1	WORD 3	WORD 2	WORD 1	WORD 0
0x00001330	1	0	0	WORD 3	WORD 2	WORD 1	WORD 0
0x00001310	1	0	0	WORD 3	WORD 2	WORD 1	WORD 0
0x00001340	1	0	0	WORD 3	WORD 2	WORD 1	WORD 0
0x00001350	1	0	0	WORD 3	WORD 2	WORD 1	WORD 0

Note 1: Mask bits are not shown and are assumed to be '0'.

2: Bits 3-0 of the address in the Cache Tags register are always implied to be '0'.

4.2.1 Cache Organization

The cache consists of two arrays: data and tag. A data array consists of program instructions or program data. The cache is physically tagged and address matches are based on the physical address, not the virtual address.

Each line in the tag array contains the following information:

- Mask – address mask value
- Tag – tag address to match against
- Valid bit
- Lock bit
- Type – an instruction and/or data type-indicator bit

Each line in the data array contains 16 bytes of program instruction, or program data, depending on the value of the type-indicator bit.

Figure 4-3 illustrates the organization of a line. It should be noted that the LMASK<10:0> bits (CHEMSK<15:5>) and the LTYPE bit (CHETAG<1>) are not programmable for every line. The LTAG<20:0> bits (CHETAG<23:4>) only implement the number of bits needed to fully map to the size of the PFM. For example, if the PFM size is 512 Kbytes, the LTAG<20:0> bits (CHETAG<23:4>) only implement bits 15 through 0 (CHETAG<18:4>).

Figure 4-3: Mask Line

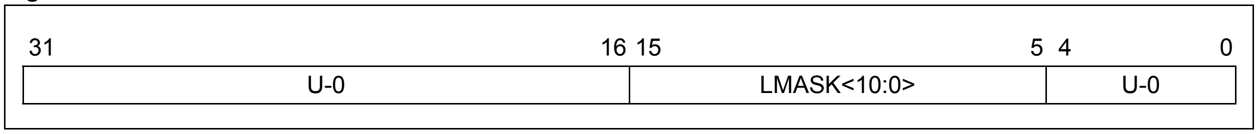


Figure 4-4: Tag Line

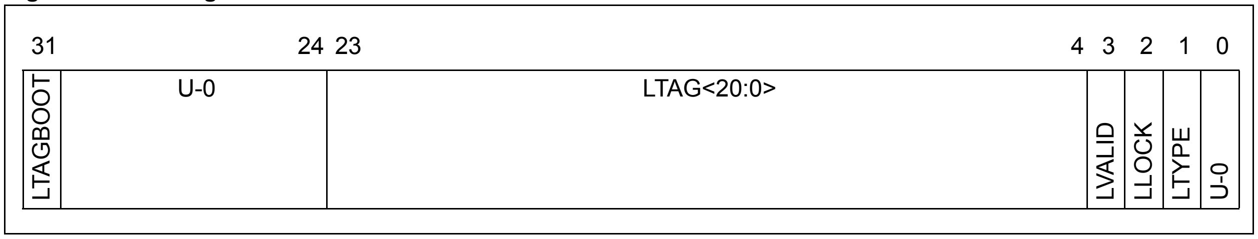
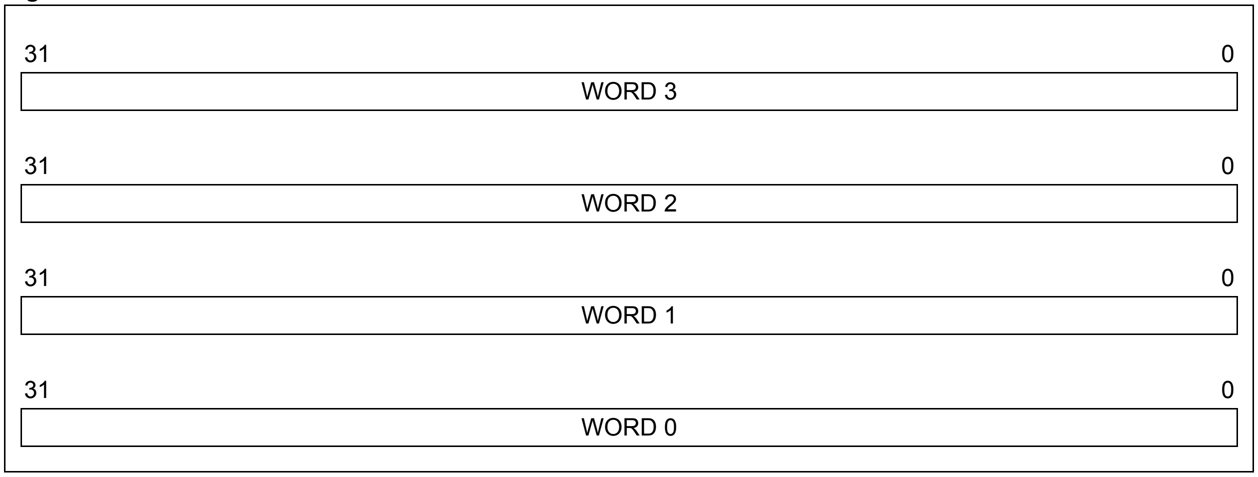


Figure 4-5: Data Line



Cache arrays are shown in Table 4-1. Software can modify the values in both the Tag Line and the Data Line of the cache. The CHEIDX<3:0> bits (CHEACC<3:0>) select a line for access. That line can then be modified via the CHETAG, CHEMSK, CHEW0, CHEW1, CHEW2 and CHEW3 registers.

Table 4-1: Cache Arrays

Line	Tag Array					Data Array[2]			
	Mask	Flags							
0	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
1	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
2	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
3	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
4	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
5	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
6	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
7	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
8	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
9	0x000[1]	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
A	LMASK	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
B	LMASK	LTAG	LVALID	LLOCK	LTYPE[3]	Word 3	Word 2	Word 1	Word 0
C	0x000[1]	LTAG	LVALID	LLOCK	LTYPE	Word 3	Word 2	Word 1	Word 0
D	0x000[1]	LTAG	LVALID	LLOCK	LTYPE	Word 3	Word 2	Word 1	Word 0
E	0x000[1]	LTAG	LVALID	LLOCK	LTYPE	Word 3	Word 2	Word 1	Word 0
F	0x000[1]	LTAG	LVALID	LLOCK	LTYPE	Word 3	Word 2	Word 1	Word 0

Note 1: Read-only bit.
2: Read '0' when device is code-protected; otherwise, read/write.
3: Type is fixed as instruction.

It is recommended that the cache lines be modified while executing from non-cacheable addresses, as the cache controller does not protect against modifying the cache while executing from a cacheable address.

Not all bits are writable. The LMASK<10:0> bits (CHEMSK<15:5>) are only writable for lines A and B, and the LTYPE bit (CHETAG<1>) is fixed to the instruction setting for lines 0 through B.

Note that lines allocated for Lock and Data affect the selection of the line to replace on a miss. However, they do not affect the usage order or pseudo-LRU value.

4.3 CONTROL REGISTERS

Note:	Some devices in the PIC32 family do not contain a Prefetch Cache module. For such devices, all prefetch cache register locations are unimplemented and should not be accessed. Refer to the specific device data sheet to determine its availability.

The Prefetch Cache module contains the following Special Functions Registers (SFRs):

- **CHECON: Cache Control Register**[1,2,3]
  This register manages configuration of the prefetch cache and controls Wait states.
- **CHEACC: Cache Access Register**[1,2,3]
  This register points to one of the 16 cache lines to access using the CHETAG, CHEMSK, CHEW0, CHEW1, CHEW2, and CHEW3 registers.
- **CHETAG: Cache TAG Register**[1,2,3,4]
  This register contains the address and type of information stored in a cache line.
- **CHEMSK: Cache TAG Mask Register**[1,2,3,4]
  This register provides a mechanism to ignore the TAG bits in the CHETAG register.
- **CHEW0: Cache Word 0** through **CHEW3: Cache Word 3**[1]
  These four registers provide access to the prefetch cache data array.
- **CHELRU: Cache LRU Register**
  This register indicates the pseudo-LRU state of the cache.
- **CHEHIT: Cache Hit Statistics Register**
  This register contains the number of cache hits.
- **CHEMIS: Cache Miss Statistics Register**
  This register contains the number of missed cache operations.
- **CHEPFABT: Prefetch Cache Abort Statistics Register**
  This register contains the number of aborted prefetch cache operations.

Table 4-2 provides a brief summary of prefetch cache-related registers. Corresponding registers appear after the summary, followed by a detailed description of each register.

Table 4-2: Prefetch Cache SFRs Summary

Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
CHECON[(1,2,3)]	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	CHECOH
	15:8	—	—	—	—	—	—	DCSZ<1:0>	
	7:0	—	—	PREFEN<1:0>		—	PFMWS<2:0>		
CHEACC[(1,2,3)]	31:24	CHEWEN	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	—	—	—	—	—	—	—	—
	7:0	—	—	—	—	CHEIDX<3:0>			
CHETAG[(1,2,3)]	31:24	LTAGBOOT	—	—	—	—	—	—	—
	23:16	LTAG<19:12>							
	15:8	LTAG<11:4>							
	7:0	LTAG<3:0>				LVALID	LLOCK	LTYPE	—
CHEMSK[(1,2,3)]	31:24	—	—	—	—	—	—	—	—
	23:16	—	—	—	—	—	—	—	—
	15:8	LMASK<10:3>							
	7:0	LMASK<2:0>			—	—	—	—	—
CHEW0	31:24	CHEW0<31:24>							
	23:16	CHEW0<23:16>							
	15:8	CHEW0<15:8>							
	7:0	CHEW0<7:0>							
CHEW1	31:24	CHEW1<31:24>							
	23:16	CHEW1<23:16>							
	15:8	CHEW1<15:8>							
	7:0	CHEW1<7:0>							
CHEW2	31:24	CHEW2<31:24>							
	23:16	CHEW2<23:16>							
	15:8	CHEW2<15:8>							
	7:0	CHEW2<7:0>							
CHEW3	31:24	CHEW3<31:24>							
	23:16	CHEW3<23:16>							
	15:8	CHEW3<15:8>							
	7:0	CHEW3<7:0>							
CHELRU	31:24	—	—	—	—	—	—	—	CHELRU<24>
	23:16	CHELRU<23:16>							
	15:8	CHELRU<15:8>							
	7:0	CHELRU<7:0>>							

Legend: — = unimplemented, read as '0'.

Note 1: This register has an associated Clear register at an offset of 0x4 bytes. The Clear register has the same name with CLR appended to the register name (e.g., CHECONCLR). Writing a '1' to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register at an offset of 0x8 bytes. The Set register has the same name with SET appended to the register name (e.g., CHECONSET). Writing a '1' to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register at an offset of 0xC bytes. The Invert register has the same name with INV appended to the register name (e.g., CHECONINV). Writing a '1' to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

Table 4-2: Prefetch Cache SFRs Summary (Continued)

Name		Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
CHEHIT	31:24	CHEHIT<31:24>							
	23:16	CHEHIT<23:16>							
	15:8	CHEHIT<15:8>							
	7:0	CHEHIT<7:0>							
CHEMIS	31:24	CHEMIS<31:24>							
	23:16	CHEMIS<23:16>							
	15:8	CHEMIS<15:8>							
	7:0	CHEMIS<7:0>							
CHEPFABT	31:24	CHEPFABT<31:24>							
	23:16	CHEPFABT<23:16>							
	15:8	CHEPFABT<15:8>							
	7:0	CHEPFABT<7:0>							

Legend: — = unimplemented, read as ‘0’.

Note 1: This register has an associated Clear register at an offset of 0x4 bytes. The Clear register has the same name with CLR appended to the register name (e.g., CHECONCLR). Writing a ‘1’ to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register at an offset of 0x8 bytes. The Set register has the same name with SET appended to the register name (e.g., CHECONSET). Writing a ‘1’ to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register at an offset of 0xC bytes. The Invert register has the same name with INV appended to the register name (e.g., CHECONINV). Writing a ‘1’ to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

Register 4-1: CHECON: Cache Control Register(1,2,3)

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0
	—	—	—	—	—	—	—	CHECOH
15:8	U-0	U-0	U-0	U-0	U-0	U-0	R/W-0	R/W-0
	—	—	—	—	—	—	DCSZ<1:0>	
7:0	U-0	U-0	R/W-0	R/W-0	U-0	R/W-1	R/W-1	R/W-1
	—	—	PREFEN<1:0>		—	PFMWS<2:0>		

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: ('0', '1', x = Unknown)	

bit 31-17 **Unimplemented:** Write '0'; ignore read

bit 16 **CHECOH:** Cache Coherency Setting on a PFM Program Cycle bit
1 = Invalidate all data and instruction lines
0 = Invalidate all data lnes and instruction lines that are not locked

bit 15-10 **Unimplemented:** Write '0'; ignore read

bit 9-8 **DCSZ<1:0>:** Data Cache Size in Lines bits
11 = Enable data caching with a size of 4 Lines
10 = Enable data caching with a size of 2 Lines
01 = Enable data caching with a size of 1 Line
00 = Disable data caching
Changing these bits induce all lines to be reinitialized to the "invalid" state.

bit 7-6 **Unimplemented:** Write '0'; ignore read

bit 5-4 **PREFEN<1:0>:** Predictive Prefetch Enable bits
11 = Enable predictive prefetch for both cacheable and non-cacheable regions
10 = Enable predictive prefetch for non-cacheable regions only
01 = Enable predictive prefetch for cacheable regions only
00 = Disable predictive prefetch

bit 3 **Unimplemented:** Write '0'; ignore read

Note 1: This register has an associated Clear register (CHECONCLR) at an offset of 0x4 bytes. Writing a '1' to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register (CHECONSET) at an offset of 0x8 bytes. Writing a '1' to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register (CHECONINV) at an offset of 0xC bytes. Writing a '1' to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

Register 4-1: CHECON: Cache Control Register(1,2,3) (Continued)

bit 2-0	**PFMWS<2:0>:** PFM Access Time Defined in Terms of SYSLK Wait States bits `111` = Seven Wait states `110` = Six Wait states `101` = Five Wait states `100` = Four Wait states `011` = Three Wait states `010` = Two Wait states `001` = One Wait state `000` = Zero Wait state

Note 1: This register has an associated Clear register (CHECONCLR) at an offset of 0x4 bytes. Writing a '`1`' to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register (CHECONSET) at an offset of 0x8 bytes. Writing a '`1`' to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register (CHECONINV) at an offset of 0xC bytes. Writing a '`1`' to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

Register 4-2: CHEACC: Cache Access Register(1,2,3)

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	CHEWEN	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
15:8	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
7:0	U-0	U-0	U-0	U-0	R/W-0	R/W-0	R/W-0	R/W-0
	—	—	—	—	CHEIDX<3:0>			

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31 **CHEWEN:** Cache Access Enable bits for registers CHETAG, CHEMSK, CHEW0, CHEW1, CHEW2, and CHEW3
1 = The cache line selected by CHEIDX<3:0> is writeable
0 = The cache line selected by CHEIDX<3:0> is not writeable

bit 30-4 **Unimplemented:** Write ‘0’; ignore read

bit 3-0 **CHEIDX<3:0>:** Cache Line Index bits
The value selects the cache line for reading or writing.

Note 1: This register has an associated Clear register (CHEACCCLR) at an offset of 0x4 bytes. Writing a ‘1’ to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register (CHEACCSET) at an offset of 0x8 bytes. Writing a ‘1’ to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register (CHEACCINV) at an offset of 0xC bytes. Writing a ‘1’ to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

Register 4-3: CHETAG: Cache TAG Register(1,2,3,4)

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	LTAGBOOT	—	—	—	—	—	—	—
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	LTAG<19:12>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	LTAG<11:4>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-0	R/W-0	R/W-1	U-0
	LTAG<3:0>				LVALID	LLOCK	LTYPE	—

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: ('0', '1', x = Unknown)	

bit 31 **LTAGBOOT:** Line TAG Address Boot bit
1 = The line is in the 0x1D000000 (physical) area of memory
0 = The line is in the 0x1FC00000 (physical) area of memory

bit 30-24 **Unimplemented:** Write '0'; ignore read

bit 23-4 **LTAG<19:0>:** Line TAG Address bits
LTAG<19:0> bits are compared against physical address to determine a hit. Because its address range and position of PFM in kernel space and user space, the LTAG PFM address is identical for virtual addresses, (system) physical addresses, and PFM physical addresses.

bit 3 **LVALID:** Line Valid bit
1 = The line is valid and is compared to the physical address for hit detection
0 = The line is not valid and is not compared to the physical address for hit detection

bit 2 **LLOCK:** Line Lock bit
1 = The line is locked and will not be replaced
0 = The line is not locked and can be replaced

bit 1 **LTYPE:** Line Type bit
1 = The line caches instruction words
0 = The line caches data words

bit 0 **Unimplemented:** Write '0'; ignore read

Note 1: This register has an associated Clear register (CHETAGCLR) at an offset of 0x4 bytes. Writing a '1' to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register (CHETAGSET) at an offset of 0x8 bytes. Writing a '1' to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register (CHETAGINV) at an offset of 0xC bytes. Writing a '1' to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

4: The TAG and Status of the Line pointed to by CHEIDX<3:0> bits (CHEACC<3:0>).

Register 4-4: CHEMSK: Cache TAG Mask Register(1,2,3,4)

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
23:16	U-0	U-0	U-0	U-0	U-0	U-0	U-0	U-0
	—	—	—	—	—	—	—	—
15:8	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
	LMASK<10:3>							
7:0	R/W-0	R/W-0	R/W-0	U-0	U-0	U-0	U-0	U-0
	LMASK<2:0>			—	—		—	—

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-16 **Unimplemented:** Write ‘0’; ignore read

bit 15-5 **LMASK<10:0>:** Line Mask bits

1 = Enables mask logic to force a match on the corresponding bit position in LTAG<19:0> bits (CHETAG<23:4>) and the physical address.

0 = Only writeable for values of CHEIDX<3:0> bits (CHEACC<3:0>) equal to 0x0A and 0x0B. Disables mask logic.

bit 4-0 **Unimplemented:** Write ‘0’; ignore read

Note 1: This register has an associated Clear register (CHEMSKCLR) at an offset of 0x4 bytes. Writing a ‘1’ to any bit position in the Clear register will clear valid bits in the associated register. Reads from the Clear register should be ignored.

2: This register has an associated Set register (CHEMSKSET) at an offset of 0x8 bytes. Writing a ‘1’ to any bit position in the Set register will set valid bits in the associated register. Reads from the Set register should be ignored.

3: This register has an associated Invert register (CHEMSKINV) at an offset of 0xC bytes. Writing a ‘1’ to any bit position in the Invert register will invert valid bits in the associated register. Reads from the Invert register should be ignored.

4: The TAG Mask of the line pointed to by CHEIDX<3:0> bits (CHEACC<3:0>).

Register 4-5: CHEW0: Cache Word 0

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW0<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW0<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW0<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW0<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0 **CHEW0<31:0>:** Word 0 of the cache line selected by CHEIDX<3:0> bits (CHEACC<3:0>)
Readable only if the device is not code-protected.

Register 4-6: CHEW1: Cache Word 1

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW1<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW1<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW1<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW1<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0	**CHEW1<31:0>:** Word 1 of the cache line selected by CHEIDX<3:0> bits (CHEACC<3:0>) Readable only if the device is not code-protected.

Register 4-7: CHEW2: Cache Word 2

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW2<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW2<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW2<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW2<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0	**CHEW2<31:0>:** Word 2 of the cache line selected by CHEIDX<3:0> bits (CHEACC<3:0>) Readable only if the device is not code-protected.

Register 4-8: CHEW3: Cache Word 3[1]

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW3<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW3<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW3<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEW3<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0 **CHEW3<31:0>:** Word 3 of the cache line selected by CHEIDX<3:0> bits (CHEACC<3:0>)
Readable only if the device is not code-protected.

Note 1: This register is a window into the cache data array and is readable only if the device is not code-protected.

Register 4-9: CHELRU: Cache LRU Register

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	U-0	U-0	U-0	U-0	U-0	U-0	U-0	R-0
	—	—	—	—	—	—	—	CHELRU<24>
23:16	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	CHELRU<23:16>							
15:8	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	CHELRU<15:8>							
7:0	R-0	R-0	R-0	R-0	R-0	R-0	R-0	R-0
	CHELRU<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-25 **Unimplemented:** Write ‘0’; ignore read

bit 24-0 **CHELRU<24:0>:** Cache Least Recently Used State Encoding bits
Indicates the pseudo-LRU state of the cache.

Register 4-10: CHEHIT: Cache Hit Statistics Register

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEHIT<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEHIT<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEHIT<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEHIT<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0	**CHEHIT<31:0>:** Cache Hit Count bits Incremented each time the processor issues an instruction fetch or load that hits the prefetch cache from a cacheable region. Non-cacheable accesses do not modify this value.

Register 4-11: CHEMIS: Cache Miss Statistics Register

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEMIS<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEMIS<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEMIS<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEMIS<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0 **CHEMIS<31:0>:** Cache Miss Count bits

Incremented each time the processor issues an instruction fetch from a cacheable region that misses the prefetch cache. Non-cacheable accesses do not modify this value.

Register 4-12: CHEPFABT: Prefetch Cache Abort Statistics Register

Bit Range	Bit 31/23/15/7	Bit 30/22/14/6	Bit 29/21/13/5	Bit 28/20/12/4	Bit 27/19/11/3	Bit 26/18/10/2	Bit 25/17/9/1	Bit 24/16/8/0
31:24	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEPFABT<31:24>							
23:16	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEPFABT<23:16>							
15:8	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEPFABT<15:8>							
7:0	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x	R/W-x
	CHEPFABT<7:0>							

Legend:		
R = Readable bit	W = Writable bit	P =Programmable bit
U = Unimplemented bit	-n = Bit value at POR: (‘0’, ‘1’, x = Unknown)	

bit 31-0 **CHEPFABT<31:0>:** Prefab Abort Count bits
Incremented each time an automatic prefetch cache is aborted due to a non-sequential instruction fetch, load or store.

4.4 CACHE OPERATION

The PFM cache and the Prefetch Cache module implement a fully associative 16-line cache. Each line consists of 128 bits (16 bytes). The PFM cache and the Prefetch Cache module request only 16-byte aligned instruction data from the PFM. If the CPU requested address is not aligned to a 16-byte boundary, the module will align the address by dropping address bits 3 through 0. When configured only as a cache, the module loads multiple instructions into a line on a miss. It uses the pseudo-LRU algorithm to select which line receives the new set of instructions. The cache controller uses the Wait states values from PFMWS<2:0> bits (CHECON<2:0>) to determine how long it must wait for a PFM access when it detects a miss. On a hit, the cache returns data in zero Wait states. If the code is 100% linear, the Cache-only mode will provide instructions back to the CPU with Wait states only on the first instruction of a cache line. For 32-bit linear code, Wait states are seen every four instructions. For 16-bit linear code, Wait states occur only once for every eight instructions executed.

4.5 CACHE CONFIGURATIONS

The CHECON register controls the configurations available for instruction and data caching of the PFM. Two parameters control the allocation of cache lines to specific features.

The DCSZ<1:0> bits (CHECON<9:8>) control the number of lines allocated to program data caching. Table 4-3 shows the cache line relationship for the values of the DCSZ<1:0> bits (CHECON<9:8>). The data caching capability is for read-only data, for example, constants, parameters, table data, and so on, that are not modified.

Table 4-3: Program Data Cache

DCSZ<1:0> (CHECON<9:8>)	Lines Allocated to Program Data
`11`	Cache Lines Number 12 through 15
`10`	Cache Lines Number 14 and 15
`01`	Cache Line Number 15
`00`	None

The PREFEN<1:0> bits (CHECON<5:4>) control predictive prefetching, which allows the cache controller to speculatively fetch the next 16-byte aligned set of instructions.

4.5.1 Line Locking

Each line in the cache can be locked to hold its contents. A line is locked if both the LVALID bit (CHETAG<3>) = `1` and the LLOCK bit (CHETAG<2>) = `1`. If LVALID = `0` and LLOCK = `1`, the cache controller issues a preload request (see **4.5.3 "Preload Behavior"**). Locking cache lines may reduce the performance of general program flow. However, if one or two function calls consume a significant percent of overall processing, locking their addresses can provide improved performance.

Though any number of lines can be locked, the cache works more efficiently when locking either 1 or 4 lines. If locking 4 lines, choose those lines in which the line numbers, when divided by 4, have the same quotient. This locks an entire LRU group, which benefits the LRU algorithm. For example, lines 8, 9, A, and B each have a quotient of 2 when divided by 4.

4.5.2 Address Mask

Cache lines 10 and 11 allow masking of the CPU address, and the tag address, to force a match on corresponding bits. The LMASK<10:0> bits (CHEMSK<15:5>) is set up to complement the interrupt vector spacing field in the CPU. This feature allows boot code to lock the first four instructions of a vector in the cache. If all vectors contain identical instructions in their first four locations, setting the LMASK<10:0> bits (CHEMSK<15:5>) to match the vector spacing, and the LTAG<19:0> bits (CHETAG<23:4>) to match the vector base address, causes all of the vector addresses to hit the cache. The cache responds with zero Wait states and immediately initiates a fetch of the next set of four instructions for the requesting vector if the prefetch cache is enabled.

Using the LMASK<10:0> bits (CHEMSK<15:5>) is restricted to aligned address ranges. Its size allows for a maximum range of 32 Kbytes and a minimum spacing of 32 bytes. Using the two lines in conjunction provides the ability to have different ranges and different spacing.

Setting up the address mask such that more than one line will match an address causes undefined results. Therefore, it is highly recommended that masking is set up before entering the cacheable code.

4.5.3 Preload Behavior

Application code can direct the cache controller to preform a preload of a cache line and lock it with instructions or data from the PFM. The preload function uses the CHEIDX<3:0> bits (CHEACC<3:0>) to select the cache line into which the load is directed. Setting the CHEWEN bit (CHEACC<31>) to ‘1’ enables writes to the CHETAG register.

Writing the LVALID bit (CHETAG<3>) = 0 and the LLOCK bit (CHETAG<2>) = 1 causes a preload request to the cache controller. The controller acknowledges the request in the cycle after the write and, if possible, stops any outstanding PFM access, and stalls any CPU load from the cache or PFM.

When the controller has finished or stalled the previous transaction, it initiates a PFM read to fetch the instructions, or data, requested using the address in LTAG<19:0> bits (CHETAG<23:4>). After the programmed number of Wait states, as defined by the PFMWS<2:0> bits (CHECON<2:0>), the controller updates the data array with the values read from the PFM. On the update, it sets the LVALID bit (CHETAG<3>) = 1. The LRU state of the line is not affected.

After the controller finishes updating the cache, it allows CPU requests to complete. If this request misses the cache, the controller initiates a PFM read, which incurs the full PFM access time.

4.5.4 Bypass Behavior

Processor accesses in which cache coherency attributes indicate uncacheable addresses bypass the cache. In bypass, the module accesses the PFM for every instruction, incurring the PFM access time as defined by the PFMWS<2:0> bits (CHECON<2:0>).

4.5.5 Predictive Prefetch Cache Behavior

When configured for predictive prefetch cache on cacheable addresses, the Prefetch Cache module predicts the next line address and returns it into the pseudo-LRU line of the cache. If enabled, the prefetch cache function starts predicting based on the first CPU instruction fetch. When the first line is placed in the cache, the module simply increments the address to the next 16-byte aligned address and starts a PFM access. When running linear code (i.e. no jumps), the PFM returns the next set of instructions into the prefetch cache buffer on or before all instructions can be executed from the previous line.

If, at any time during a predicted PFM access, a new CPU address does not match the predicted one, the PFM access will be changed to the correct address. This behavior does not cause the CPU access to take any longer than it does without prediction.

If an access that misses the cache hits the prefetch cache buffer, the instructions are placed in the pseudo-LRU line, along with its address tag. The pseudo-LRU value is marked as the most recently used line, and other lines are updated accordingly. If an access misses both the cache and the prefetch cache buffer, the access passes to the PFM, and those returning instructions are placed in the pseudo-LRU line.

When configured for predictive prefetch cache on non-cacheable addresses, the controller uses only the prefetch cache buffer. The LRU cache line is not updated for hits or fills, so the cache remains intact. For linear code, enabling predictive prefetch cache for non-cacheable addresses allows the CPU to fetch instructions in zero Wait states.

It is not useful to use non-cacheable predictive prefetching when accesses to the PFM are set for zero Wait states. The controller holds prefetched instructions on the output of the PFM for up to three clock cycles (while the CPU is fetching from the buffer). This consumes more power, without any benefit for zero Wait state PFM accesses.

Predictive data prefetching is *not* supported. However, a data access in the middle of a predictive instruction fetch causes the cache controller to stop the PFM access for the instruction fetch, and to start the data load from PFM. The predictive prefetch cache does not resume, but instead, waits for another instruction fetch. When this occurs, it either fills the buffer because of a miss, or starts a prefetch cache because of a hit.

4.5.6 Cache Replacement Policy

The cache controller uses a pseudo-LRU replacement policy for cache line fills that are caused by a read miss. The policy allows any line in the last quarter of least recently used lines to be replaced. Enabling locking and data caching affect the line to be replaced, but not the actual value of the pseudo-LRU.

4.6 COHERENCY SUPPORT

It is not possible to execute out of cache while programming the PFM. The PFM controller stalls the cache during the programming sequence. Therefore, user code that initiates a programming sequence should not be located in a cacheable address region.

During a programming operation, the prefetch cache is flushed by invalidating either all, or some of the cache lines.

If the CHECOH bit (CHECON<16>) is set, every cache line is invalidated and unlocked during a PFM write operation. The cache tags and masks are also cleared for all lines.

If CHECOH is not set, only lines that are not locked are forced invalid. Lines that are locked are retained.

Note:	The user application should switch to a non-cacheable region before invalidating the cache lines.

4.7 EFFECTS OF RESET

4.7.1 On Reset

- All cache lines are invalidated
- All cache lines revert to instruction
- All cache lines are unlocked
- The LRU order is sequential, with line 0 being the least recently used
- All mask bits are cleared
- All registers revert to their reset state

4.7.2 After Reset

- The module operates as per the values in the CHECON register
- The cache obeys the core's cache coherency attributes

4.8 OPERATION IN POWER-SAVING MODES

4.8.1 Sleep Mode

When the device enters Sleep mode, the prefetch cache is disabled and placed into a low-power state where no clocking occurs in the Prefetch Cache module.

4.8.2 Idle Mode

When the device enters Idle mode, the cache and prefetch cache clock source remains functional and the CPU stops executing code. Any outstanding prefetch cache completes before the Prefetch Cache module stops its clock via automatic clock gating.

4.8.3 Debug Mode

The behavior of the prefetch cache is unaltered in Debug mode. Care must be taken to make sure the cache remains coherent during Debug mode execution when using software breakpoints. If a debugger places a software break instruction in the cache, the line should be locked before returning control to the application. When a locked software breakpoint is removed, the line should be unlocked and invalidated, causing the original instructions to be reloaded from the PFM upon execution.

4.9 DESIGN TIP

Even while running at clock frequencies allowing for zero Wait state operation, the cache function proves useful as a power-saving technique. Accesses to the PFM consume more power than accesses to the cache.

4.10 RELATED APPLICATION NOTES

This section lists application notes that are related to this section of the manual. These application notes may not be written specifically for the PIC32 device family, but the concepts are pertinent and could be used with modification and possible limitations. The current application notes related to the Prefetch Cache module are:

Title	**Application Note #**
No related application notes at this time.	N/A

Note:	Please visit the Microchip web site (www.microchip.com) for additional application notes and code examples for the PIC32 family of devices.

4.11 REVISION HISTORY

Revision A (October 2007)

This is the initial released version of this document.

Revision B (October 2007)

Updated document to remove Confidential status.

Revision C (April 2008)

Revised status to Preliminary; Revise U-0 to r-x.

Revision D (June 2008)

Change Reserved bits from “Maintain as” to “Write”.

Revision E (February 2011)

This revision includes the following updates:

- Changed the document running header from PIC32MX Family Reference Manual to PIC32 Family Reference Manual
- Changed the section name from Prefetch Cache Module to Prefetch Cache
- Changed all occurrences of PIC32MX to PIC32
- Updated the note describing the availability of the Prefetch Cache module in **4.3 “Control Registers”**
- Updated the Cache Arrays table (see Table 4-1)
- Changed the PREFEN<1:0> bits definition from Predictive Prefetch Cache Enable bits to Predictive Prefetch Enable bits in Register 4-1
- Renamed the bit, PFABT to CHEPFABT in Register 4-12
- Removed the following Clear, Set and Invert registers:
  - CHECONCLR: CHECON Clear Register
  - CHECONSET: CHECON Set Register
  - CHECONINV: CHECON Invert Register
  - CHEACCCLR: CHEACC Clear Register
  - CHEACCSET: CHEACC Set Register
  - CHEACCINV: CHEACC Invert Register
  - CHETAGCLR: CHETAG Clear Register
  - CHETAGSET: CHETAG Set Register
  - CHETAGINV: CHETAG Invert Register
  - CHEMSKCLR: CHEMSK Clear Register
  - CHEMSKSET: CHEMSK Set Register
  - CHEMSKINV: CHEMSK Invert Register
- Added notes to Register 4-1 through Register 4-4 describing the corresponding Set, Clear and Invert registers.
- Changed all occurrences of r-0 and r-x to U-0 and updated the corresponding bit descriptions
- Added a note in **4.6 “Coherency Support”**
- Removed the note describing the distinction between power modes of the specific module and the power modes of the device in **4.8 “Operation in Power-Saving Modes”**
- Minor changes to the text and formatting have been incorporated throughout the document

ISBN: 978-1-60932-908-2

MICROCHIP

Worldwide Sales and Service

AMERICAS

Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 480-792-7200
Fax: 480-792-7277
Technical Support:
http://www.microchip.com/support
Web Address:
www.microchip.com

Atlanta
Duluth, GA
Tel: 678-957-9614
Fax: 678-957-1455

Boston
Westborough, MA
Tel: 774-760-0087
Fax: 774-760-0088

Chicago
Itasca, IL
Tel: 630-285-0071
Fax: 630-285-0075

Cleveland
Independence, OH
Tel: 216-447-0464
Fax: 216-447-0643

Dallas
Addison, TX
Tel: 972-818-7423
Fax: 972-818-2924

Detroit
Farmington Hills, MI
Tel: 248-538-2250
Fax: 248-538-2260

Indianapolis
Noblesville, IN
Tel: 317-773-8323
Fax: 317-773-5453

Los Angeles
Mission Viejo, CA
Tel: 949-462-9523
Fax: 949-462-9608

Santa Clara
Santa Clara, CA
Tel: 408-961-6444
Fax: 408-961-6445

Toronto
Mississauga, Ontario,
Canada
Tel: 905-673-0699
Fax: 905-673-6509

ASIA/PACIFIC

Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

China - Beijing
Tel: 86-10-8528-2100
Fax: 86-10-8528-2104

China - Chengdu
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

China - Chongqing
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

China - Hong Kong SAR
Tel: 852-2401-1200
Fax: 852-2401-3431

China - Nanjing
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

China - Qingdao
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

China - Shanghai
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

China - Shenyang
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

China - Shenzhen
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

China - Wuhan
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

China - Xian
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

China - Xiamen
Tel: 86-592-2388138
Fax: 86-592-2388130

China - Zhuhai
Tel: 86-756-3210040
Fax: 86-756-3210049

ASIA/PACIFIC

India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 or
82-2-558-5934

Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

Taiwan - Hsin Chu
Tel: 886-3-6578-300
Fax: 886-3-6578-370

Taiwan - Kaohsiung
Tel: 886-7-213-7830
Fax: 886-7-330-9305

Taiwan - Taipei
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

EUROPE

Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

Denmark - Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

02/18/11