Restriction

Contents

Notes on Hardware

EE and GS chips have the following restrictions. Take note for the use of the following features.

Restrictions on EE

(1) Loop

Do not create a short loop which is 6 steps or less including a branch instruction.Such a loop may end after executing once or twice, under special conditions.

(2) Arrangement of Program Code and Data

When arranging program code and data in adjoining addresses, put 5 or more NOP instructions, or a combination of SYNC.P and NOP instructions on the boundary between them. When the data arranged next to the program code has a specific bit pattern, it is regarded as a CACHE instruction, and may fetch a wrong instruction, destroy the data cache, or affect floating point divide of COP1.

(3) Microprogram and VCALLMS/VCALLMSR Instruction

It is necessary to prohibit VU microsubroutines from generating resource hazard, data hazard, or interlock due to a SYNC instruction at the last microinstruction (the instruction following Ebit=1). If there is a possibility of causing an interlock, avoid placing VCALLMS / VCALLMSR successively in the EE Core program by inserting another macro instruction or a move instruction with interlock (CFC2 / CTC2 / LQC2 / SQC2) in between. If this restriction is not followed, a microinstruction may not be executed at the occurrence of an exception.

(4) DMAtag Mismatch Detection Feature

Set VIFn_ERR.ME0 to 1 to disable DMAtag mismatch error detection.If this restriction is not followed, a DMAtag mismatch error interrupt may occur to a normal DMA packet.

(5) DMAtag when Using MFIFO

When using MFIFO, avoid placing ref/refs/refe tag at the end of the DMA packet which is sent from the source channel (ch-8), by taking the following steps.

[End of DMA Packet]

Eref/refs tag: Add cnt tag (QWC=0)

Erefe tag: Substitute ref tag + end tag

If this restriction is not followed, the drain channel may stall due to a misunderstanding that MFIFO has become empty during data transfer.

(6) Interrupt Handler

Return from the interrupt hander after executing the EI instruction.If this restriction is not followed, an inconvenience may happen when an interrupt occurs immediately after executing the DI instruction.

(7) Undefined Instruction Exception Handler

In the reserved instruction (RI) exception handler routine, do not execute a COP1 divide instruction (DIV.S, SQRT.S, or RSQRT.S) in the first 18 instructions. If this restriction is not followed, the calculation result by the floating-point divider (FDIV) may cling to a certain value.

(8) Restriction on Data Value Breakpoint

When using a data value breakpoint (BPC.DRE=1, BPC.DVE=1), specify blocking mode (Status.NBL=0).

The breakpoint conditions for the data value breakpoint are established when the specified data value is read from the specified address. In non-blocking mode, however, the breakpoint conditions are considered to be established even when executing an instruction to load a non-specified data value from the specified address and an instruction to load the specified data value from a non-specified address concurrently.

(9) Note on COP1 Divide Instructions

When executing a COP1 divide instruction (DIV.S / SQRT.S / RSQRT.S) at the D stage, if the pipeline stalls for any reason, the stall continues until the divide instruction ends.

(10) DI Instruction

Execute the following code or the like when disabling an interrupt by the DI instruction.

```
#define DI()
{
  u_int stat;
  do {
    asm volatile ("di");
    asm volatile ("sync.p");
    asm volatile ("mfc0     %0, $12" : "=r"(stat):);
  } while (stat & 0x00010000);
}
```

If this method is not applied, when an interrupt occurs immediately after the DI instruction, the EIE bit becomes 0 (interrupt disable) in the interrupt handler.Depending on OS implementation, thread switching etc. generated in the above state may cause an inconvenience with the interrupt kept disabled.

(11) CACHE/TLB Instruction

Instructions which operate the cache or TLB have restrictions; they must be directly preceded and followed by a SYNC instruction, for example. Detailed information is given to the respective instructions.If this restriction is not followed, an inconvenience may be caused when a COP0 Unusable exception occurs.

(12) CACHE Instructions

In the following instructions which operate the instruction cache or BTAC, specify a virtual address whose MSB (VA[31]) is 0 to the argument.Specifying the virtual address whose MSB is 1 may result in a malfunction.

CACHE IXIN	I$ index invalidate
CACHE IXLTG	I$ index load tag
CACHE IXSTG	I$ index store tag
CACHE IXLDT	I$ index load data
CACHE IXSDT	I$ index store data
CACHE BXLBT	index load BTAC
CACHE BXSBT	index store BTAC
CACHE BFH	BTAC flush
CACHE BHINBT	hit invalidate BTAC

(13) CACHE IFL Instruction

If an address specified by the CACHE IFL instruction already exists in the instruction cache, a malfunction such as an unpredictable TLB miss exception may occur.

Take steps (e.g. execute the CACHE IHIN instruction in advance) to avoid the specified address from existing in the instruction cache.

(14) TLBR Instruction

The TLBR instruction must not be immediately followed by a jump/branch instruction. Four instructions or more are required between them excluding the SYNC.P instruction next to the TLBR instruction. The TLBR instruction must not be placed at the end of a page, either. Six instructions or more from the end of the page are required for the TLBR instruction.
If this restriction is not followed, when an ITLB miss occurs immediately after the TLBR instruction cancellation due to an exception etc., an inconvenience may be caused.

(15) Write Operations to EPC/ErrorEPC Register

When writing to the EPC/ErrorEPC register with the MTC0 instruction, put any of the SYNC / COP0 / Load / Store instructions that is executed only in I1 Pipe immediately after the MTC0 instruction. An exception immediately after the write operation with the MTC0 instruction may result in a loss of data written with the MTC0 instruction.This is because the process to write the PC value to the EPC/ErrorEPC register is performed due to the occurrence of an exception.

(16) Undefined Instructions (1)

Undefined instructions with specific bit patterns may be misunderstood in the following cases and cause an inconvenience.

a) The next undefined instruction is in the delay slot of the Branch-Likely instruction.

Inst[31:26] = 010000 &&

Inst[25:21] = 1**** &&

Inst[5: 0] = 01****

If a branch does not occur, instructions in BDS+1 and BDS+2 may not execute.

B) The conditional branch instruction is directly preceded or followed by the next undefined instruction.

Inst[31:26] = 010011 &&

Inst[25:21] = 01000 &&

Inst[18:16] = 000

The branch target address may be mistaken with the above or succeeding branch instructions.

(17) Undefined Instruction (2)

Do not execute the following undefined instructions with specific bit pattern, since they interfere with the operation.

a) Undefined instructions which interfere with floating-point calculations

Inst[31:26]== 010001 &&
Inst[25:23]== 1*0 &&
(Inst[5: 0]== 010**1 | | Inst[5:0]==*1*011)

Floating-point calculation results may cling to a certain value. This problem also occurs when this bit pattern exists in the data area next to the program code. Therefore, it is necessary to put 5 or more NOP instructions, or a combination of SYNC.P and NOP instructions on the boundary between the program code and data.

b) Undefined instructions which affect the data cache

Inst[31:26]==101111 &&
(Inst[20:16]== 10101 | | 10111 | | 11001 | | 11011 | |
11101 | | 11110 | | 11111)

The data cache may be destroyed. An undefined instruction exception does not occcur.

c) Undefined instructions which affect TLB entries

Inst[31:26]==010000 &&
Inst[25:21]== 1**** &&
(Inst[5: 0]==000*** | | 0****1 | | *01*** | | ****1*)

TLB entries may be destroyed.

Restrictions on GS

(1) Depth Test

When the ZTE which controls the depth test is ON/OFF switched,the test may not be performed properly in some conditions.Therefore, if you are not executing the depth test nor using the Z buffer, do not set as ZTE=0 but set as ZTE=1 and ZTST=01(Always).At the same time, set the ZBUF register as ZMSK=1.The GS internal operation in this state is equivalent to the ZTE=0state.

(2) GS Interrupt Handler

The GS interrupt mask register is to be in the masked state while the interrupt from the GS to the EE (SIGNAL/FINISH/HSync/VSync) is being handled.Otherwise, while the interrupt handler is in the process for the interrupt from the GS, if additional interrupt from the GS occurs, the EE may not be able to detect it.The above process is unnecessary for the interrupts of start and termination of V-Blank since they are different interrupts from the GS interrupt described above.

(3) Texture Width

The width of the textures transferred to GS local memory has the following restriction according the texel format in use.

Texel Format	Texture Width
PSMCT32	Multiple of 2 Pixels
PSMCT24	Multiple of 8 Pixels
PSMCT16, 16S	Multiple of 4 Pixels
PSMT8, 8H, PSMT4, 4HH, 4HL	Multiple of 8 Pixels

If this restriction is not followed, a part of the data (between 1 to several pixels, in general) may not be written correctly to local memory in some rare cases (the value that had been previously written remains in the PIXEL where the data writing failed.) There will be no offset of the transfer position due to the writing failure.