Accessing Hi-Registers in THUMB State

In THUMB state, registers R8-R15 (the Hi registers) are not part of the standard register set. However, the assembly language programmer has limited access to them, and can use them for fast temporary storage.

A value may be transferred from a register in the range R0-R7 (a Lo register) to a Hi register, and from a Hi register to a Lo register, using special variants of the MOV instruction. Hi register values can also be compared against or added to Lo register values with the CMP and ADD instructions. For more information, refer to Figure 3-34.

THE PROGRAM STATUS REGISTERS

The ARM7TDMI contains a Current Program Status Register (CPSR), plus five Saved Program Status Registers (SPSRs) for use by exception handlers. These register's functions are:

- Hold information about the most recently performed ALU operation
- Control the enabling and disabling of interrupts
- Set the processor operating mode

The arrangement of bits is shown in Figure 2-6.

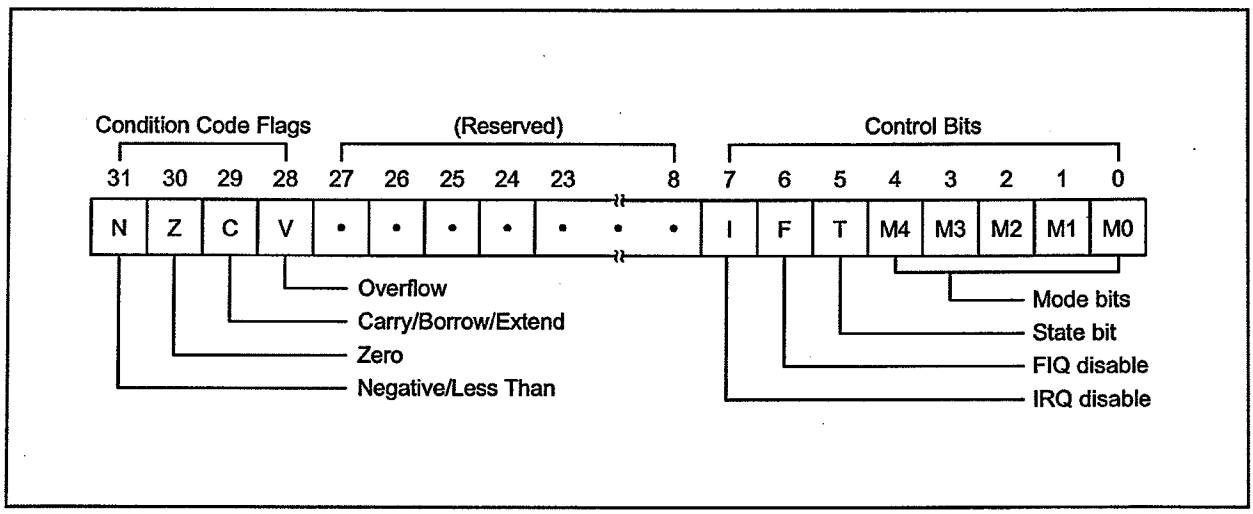

Figure 2-6. Program Status Register Format

The Condition Code Flags

The N, Z, C and V bits are the condition code flags. These may be changed as a result of arithmetic and logical operations, and may be tested to determine whether an instruction should be executed.

In ARM state, all instructions may be executed conditionally: see Table 3-2 for details.

In THUMB state, only the Branch instruction is capable of conditional execution: see Figure 3-46 for details.

The Control Bits

The bottom 8 bits of a PSR (incorporating I, F, T and M[4:0]) are known collectively as the control bits. These will be changed when an exception arises. If the processor is operating in a privileged mode, they can also be manipulated by software.

The T bit	This reflects the operating state. When this bit is set, the processor is executing in THUMB state, otherwise it is executing in ARM state. This is reflected on the **TBIT** external signal. Note that the software must never change the state of the **TBIT** in the CPSR. If this happens, the processor will enter an unpredictable state.
Interrupt disable bits	The I and F bits are the interrupt disable bits. When set, these disable the IRQ and FIQ interrupts respectively.
The mode bits	The M4, M3, M2, M1 and M0 bits (M[4:0]) are the mode bits. These determine the processor's operating mode, as shown in Table 2-1. Not all combinations of the mode bits define a valid processor mode. Only those explicitly described shall be used. The user should be aware that if any illegal value is programmed into the mode bits, M[4:0], then the processor will enter an unrecoverable state. If this occurs, reset should be applied.
Reserved bits	The remaining bits in the PSRs are reserved. When changing a PSR's flag or control bits, you must ensure that these unused bits are not altered. Also, your program should not rely on them containing specific values, since in future processors they may read as one or zero.

Table 2-1. PSR Mode Bit Values

M[4:0]	Mode	Visible THUMB state registers	Visible ARM state registers
10000	User	R7..R0, LR, SP PC, CPSR	R14..R0, PC, CPSR
10001	FIQ	R7..R0, LR_fiq, SP_fiq PC, CPSR, SPSR_fiq	R7..R0, R14_fiq..R8_fiq, PC, CPSR, SPSR_fiq
10010	IRQ	R7..R0, LR_irq, SP_irq PC, CPSR, SPSR_irq	R12..R0, R14_irq, R13_irq, PC, CPSR, SPSR_irq
10011	Supervisor	R7..R0, LR_svc, SP_svc, PC, CPSR, SPSR_svc	R12..R0, R14_svc, R13_svc, PC, CPSR, SPSR_svc
10111	Abort	R7..R0, LR_abt, SP_abt, PC, CPSR, SPSR_abt	R12..R0, R14_abt, R13_abt, PC, CPSR, SPSR_abt
11011	Undefined	R7..R0 LR_und, SP_und, PC, CPSR, SPSR_und	R12..R0, R14_und, R13_und, PC, CPSR
11111	System	R7..R0, LR, SP PC, CPSR	R14..R0, PC, CPSR

Reserved bits — The remaining bits in the PSR's are reserved. When changing a PSR's flag or control bits, you must ensure that these unused bits are not altered. Also, your program should not rely on them containing specific values, since in future processors they may read as one or zero.

IRQ

The IRQ (Interrupt Request) exception is a normal interrupt caused by a LOW level on the **nIRQ** input. IRQ has a lower priority than FIQ and is masked out when a FIQ sequence is entered. It may be disabled at any time by setting the I bit in the CPSR, though this can only be done from a privileged (non-User) mode.

Irrespective of whether the exception was entered from ARM or Thumb state, an IRQ handler should return from the interrupt by executing

```
SUBS    PC,R14_irq,#4
```

Abort

An abort indicates that the current memory access cannot be completed. It can be signalled by the external **ABORT** input. ARM7TDMI checks for the abort exception during memory access cycles.

There are two types of abort:

- *Prefetch abort:* occurs during an instruction prefetch.
- *Data abort:* occurs during a data access.

If a prefetch abort occurs, the prefetched instruction is marked as invalid, but the exception will not be taken until the instruction reaches the head of the pipeline. If the instruction is not executed - for example because a branch occurs while it is in the pipeline - the abort does not take place.

If a data abort occurs, the action taken depends on the instruction type:

- Single data transfer instructions (LDR, STR) write back modified base registers: the Abort handler must be aware of this.
- The swap instruction (SWP) is aborted as though it had not been executed.
- Block data transfer instructions (LDM, STM) complete. If write-back is set, the base is updated. If the instruction would have overwritten the base with data (ie it has the base in the transfer list), the overwriting is prevented. All register overwriting is prevented after an abort is indicated, which means in particular that R15 (always the last register to be transferred) is preserved in an aborted LDM instruction.

The abort mechanism allows the implementation of a demand paged virtual memory system. In such a system the processor is allowed to generate arbitrary addresses. When the data at an address is unavailable, the Memory Management Unit (MMU) signals an abort. The abort handler must then work out the cause of the abort, make the requested data available, and retry the aborted instruction. The application program needs no knowledge of the amount of memory available to it, nor is its state in any way affected by the abort.

After fixing the reason for the abort, the handler should execute the following irrespective of the state (ARM or Thumb):

```
SUBS    PC,R14_abt,#4    ; for a prefetch abort, or
SUBS    PC,R14_abt,#8    ; for a data abort
```

This restores both the PC and the CPSR, and retries the aborted instruction.

EINTCON	Bit	Description	Initial State
EINT0	[0]	Setting external interrupt enable of EINT0 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT1	[1]	Setting external interrupt enable of EINT1 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT2	[2]	Setting external interrupt enable of EINT2 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT3	[3]	Setting external interrupt enable of EINT3 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT4	[4]	Setting external interrupt enable of EINT4 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT5	[5]	Setting external interrupt enable of EINT5 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT6	[6]	Setting external interrupt enable of EINT6 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT7	[7]	Setting external interrupt enable of EINT7 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT8	[8]	Setting external interrupt enable of EINT8 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT9	[9]	Setting external interrupt enable of EINT9 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT10	[10]	Setting external interrupt enable of EINT10 0 = Disable external interrupt 1 = Enable external interrupt	0
EINT11	[11]	Setting external interrupt enable of EINT11 0 = Disable external interrupt 1 = Enable external interrupt	0

EINTMOD	Bit	Description	Initial State
EINT0	[2:0]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT1	[5:3]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT2	[8:6]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT3	[11:9]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT4	[14:12]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT5	[17:15]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT6	[20:18]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT7	[23:21]	000 = Falling edge triggered 001 = Rising edge triggered 010 = High level interrupt 011 = Low level interrupt 100 = Both edge triggered	000
EINT8	[25:24]	00 = Falling edge triggered 01 = Rising edge triggered 10 = High level interrupt 11 = Low level interrupt	00
EINT9	[27:26]	00 = Falling edge triggered 01 = Rising edge triggered 10 = High level interrupt 11 = Low level interrupt	00
EINT10	[29:28]	00 = Falling edge triggered 01 = Rising edge triggered 10 = High level interrupt 11 = Low level interrupt	00
EINT11	[31:30]	00 = Falling edge triggered 01 = Rising edge triggered 10 = High level interrupt 11 = Low level interrupt	00

NOTES:

1. Because each external interrupt pins has a 200ns noise filter
2. Because EINTPNDx bits are not cleared automatically, you have to clear this bit by writing "0". (Although these bits are not cleared, the interrupt triggering will operate.)

OPERATION DESCRIPTION

16-BIT TIMERS (TIMER0, TIMER1 AND TIMER2)

Interval Mode Operation

In interval mode, a match signal should be generated when the counter value is identical to the value written to the timer data register, TDAT0, TDAT1 and TDAT2. The match signal can generate a timer 0, 1, or 2 match interrupt and clear the counter value. When a match condition happens, the timer output(TOUT0/1/2) will be toggled.

Capture Mode Operation

In capture mode, the timer can perform the capturing operation, which is that the counter value is transferred into the capture register(Timer Data Register) in synchronization with an external trigger. The external triggering signal for capturing operation is a pre-defined valid edge on the capture input pin. When this valid signal happens, the counter value in process should be moved into the capture register(Timer Data Register). By using the capturing function, users can measure the time difference between external events. If a valid trigger signal on the pin does not happen before the overflow, an overflow interrupt will be generated and the counter value will be counted from 0000h, again.

Match & Overflow Mode Operation

In this mode, a match signal can be generated when the counter value is identical to the value written to the timer data register. However, the match signal does not clear the counter even if it can generate a match interrupt as same as the interval mode. Because it does not clear the counter value, the timer can run up to the overflow of counter value and generate an overflow interrupt, also. After the overflow of counter value, the counter value will be counted from 0000h, again.

DMA Mode Operation (Timer 1 Only)

Users can use the DMA to support the Timer 1. The DMA can transfer the data in memory to the TDAT1(Timer Data Register). When the match interrupt happens, the Timer 1 can request the DMA service to transfer the data into the TDAT1 register, again. Before the DMA-based operation, users should configure the control information on DMA, such as TCON1[5:3] to "010", TDAT1, destination address, source address, and so on. This kind of DMA-based timer operation is very helpful to generate the pre-defined timing event.

TIMER SPECIAL FUNCTION REGISTER

TIMER CONTROL REGISTERS

Register	Offset Address	R/W	Description	Reset Value
TCON0	0x9003	R/W	Timer 0 control register	0x00
TCON1	0x9013	R/W	Timer 1 control register	0x00
TCON2	0x9023	R/W	Timer 2 control register	0x00
TCON3	0x9033	R/W	Timer 3 control register	0x00
TCON4	0x9043	R/W	Timer 4 control register	0x00

TCON0, 1, 2	Bit	Description	Initial State
Reserved	[1:0]	Reserved	00
ICS	[2]	**Timer Input Clock Selection:** This bit can determine the input clock source of Timer. 0 = Internal Clock 1 = External Clock	0
OMS	[5:3]	**Timer Operating Mode Selection:** This field can determine the operating mode of Timer. 000 = Interval mode operation 001 = Match & overflow mode operation 010 = Match & DMA mode operation (Timer1 Only) 100 = Capture on falling edge of TCAP0, 1, and 2 101 = Capture on rising edge of TCAP0, 1, and 2 110 = Capture on rising or falling edges of TCAP0, 1, and 2	000
CL	[6]	**Timer Counter Clear:** This bit can clear the content of timer counter register. 0 = No effect 1 = Clearing the counter register	0
TEN	[7]	**Timer Enable:** This bit can enable or disable of Timer functionality. 0 = Disable (Stop) 1 = Enable (Start)	0

TIMER PRESCALER REGISTERS

Register	Offset Address	R/W	Description	Reset Value
TPRE0	0x9002	R/W	Timer 0 pre-scale register	0xff
TPRE1	0x9012	R/W	Timer 1 pre-scale register	0xff
TPRE2	0x9022	R/W	Timer 2 pre-scale register	0xff
TPRE3	0x9032	R/W	Timer 3 pre-scale register	0xff
TPRE4	0x9042	R/W	Timer 4 pre-scale register	0xff

TPREx	Bit	Description	Initial State
Pre-scale	[7:0]	This field can determines pre-scale value for Timer 0,1,2,3, and 4	0xff

TIMER 0, 1, AND 2 DATA REGISTERS

Register	Offset Address	R/W	Description	Reset Value
TDAT0	0x9000	R/W	Timer 0 data register	0xffff
TDAT1	0x9010	R/W	Timer 1 data register	0xffff
TDAT2	0x9020	R/W	Timer 2 data register	0xffff

TDAT0,1,2	Bit	Description	Initial State
Data	[15:0]	This field can determine the data value for Timer 0,1, and 2	0xffff

TIMER 0, 1, AND 2 COUNT REGISTERS

Register	Offset Address	R/W	Description	Reset Value
TCNT0	0x9006	R	Timer 0 count register	0x0
TCNT1	0x9016	R	Timer 1 count register	0x0
TCNT2	0x9026	R	Timer 2 count register	0x0

TDAT0,1,2	Bit	Description	Initial State
CV	[15:0]	This field contains the current timer's count value during the normal operation	0x0

TIMER 3 AND 4 COUNT REGISTERS

Register	Offset Address	R/W	Description	Reset Value
TCNT3	0x9037	R	Timer 3 count register	0x0
TCNT4	0x9047	R	Timer 4 count register	0x0

TDAT3, 4	Bit Size	Description	Initial State
CV	[7:0]	This field contains the current timer's count value during the normal operation	0x0

INTERRUPT SOURCE

In S3C3410X, there are 35 interrupt sources. Among them, 23 interrupt sources are coming from internal peripheral devices like the DMA controller, UART, SIO, etc. Other 8 interrupt sources are coming from external interrupt request pins like EINT0, EINT1, EINT2, EINT3, EINT8, EINT9, EINT10, and EINT11. The other 4 are coming from external interrupt request pins like EINT4, EINT5, EINT6, and EINT7. Because these 4 external interrupt requests should be OR-ed internally, we consider these external interrupt request sources as one interrupt request source to CPU. In other word, the total interrupt request sources to CPU is 32, not 35.

Sources	Description	Number
EINT0	External interrupt 0	0
EINT1	External interrupt 1	1
INT_URX	UART receive interrupt	2
INT_UTX	UART transmit interrupt	3
INT_UERR	UART error interrupt	4
INT_DMA0	DMA0 interrupt	5
INT_DMA1	DMA1 interrupt	6
INT_TOF0	Timer 0 overflow interrupt	7
INT_TMC0	Timer 0 match/capture interrupt	8
INT_TOF1	Timer 1 overflow interrupt	9
INT_TMC1	Timer 1 match/capture interrupt	10
INT_TOF2	Timer 2 overflow interrupt	11
INT_TMC2	Timer 2 match/capture interrupt	12
INT_TOF3	Timer 3 overflow interrupt	13
INT_TMC3	Timer 3 match/capture interrupt	14
INT_TOF4	Timer 4 overflow interrupt	15
INT_TMC4	Timer 4 match/capture interrupt	16
INT_BT	Basic Timer interrupt	17
INT_SIO0	SIO 0 interrupt	18
INT_SIO1	SIO 1 interrupt	19
INT_IIC	IIC interrupt	20
INT_RTCA	RTC alarm interrupt	21
INT_RTCT	RTC time interrupt(SEC/MIN/HOUR)	22
INT_TF	Timer4 FIFO interrupt	23
EINT2	External interrupt 2	24
EINT3	External interrupt 3	25
EINT4/5/6/7	External interrupt 4/5/6/7	26
INT_ADC	ADC interrupt	27
EINT8	External interrupt 8	28
EINT9	External interrupt 9	29
EINT10	External interrupt 10	30
EINT11	External interrupt 11	31

NOTE: EINT4, EINT5, EINT6 and EINT7 are sharing the same interrupt request line. So, the ISR(Interrupt Service Routine) can discriminate the interrupt request source by reading the EINTPND register because it has 4-bit for the interrupt source of EINT4, EINT5, EINT6, and EINT7. The EINTPND has to be cleared by writing "0" in ISR

INTERRUPT CONTROLLER SPECIAL FUNCTION REGISTERS

INTERRUPT MODE REGISTER (INTMOD)

Each bit in INTMOD register can determine the interrupt mode of each interrupt request. In case of FIQ mode, this bit should be "1". Otherwise, it means the IRQ mode interrupt. The FIQ mode has higher priority than IRQ mode. During the service of IRQ, the FIQ mode interrupt can occupy the CPU for its service.

Register	Offset Address	R/W	Description	Reset Value
INTMOD	0xc000	R/W	Interrupt mode register 0 = IRQ mode 1 = FIQ mode	0x0

INTMOD	Bit	Description		Initial State
EINT0	[0]	0 = IRQ mode	1 = FIQ mode	0
EINT1	[1]	0 = IRQ mode	1 = FIQ mode	0
INT_URX	[2]	0 = IRQ mode	1 = FIQ mode	0
INT_UTX	[3]	0 = IRQ mode	1 = FIQ mode	0
INT_UERR	[4]	0 = IRQ mode	1 = FIQ mode	0
INT_DMA0	[5]	0 = IRQ mode	1 = FIQ mode	0
INT_DMA1	[6]	0 = IRQ mode	1 = FIQ mode	0
INT_TOF0	[7]	0 = IRQ mode	1 = FIQ mode	0
INT_TMC0	[8]	0 = IRQ mode	1 = FIQ mode	0
INT_TOF1	[9]	0 = IRQ mode	1 = FIQ mode	0
INT_TMC1	[10]	0 = IRQ mode	1 = FIQ mode	0
INT_TOF2	[11]	0 = IRQ mode	1 = FIQ mode	0
INT_TMC2	[12]	0 = IRQ mode	1 = FIQ mode	0
INT_TOF3	[13]	0 = IRQ mode	1 = FIQ mode	0
INT_TMC3	[14]	0 = IRQ mode	1 = FIQ mode	0
INT_TOF4	[15]	0 = IRQ mode	1 = FIQ mode	0
INT_TMC4	[16]	0 = IRQ mode	1 = FIQ mode	0
INT_BT	[17]	0 = IRQ mode	1 = FIQ mode	0
INT_SIO0	[18]	0 = IRQ mode	1 = FIQ mode	0
INT_SIO1	[19]	0 = IRQ mode	1 = FIQ mode	0
INT_IIC	[20]	0 = IRQ mode	1 = FIQ mode	0
INT_RTCA	[21]	0 = IRQ mode	1 = FIQ mode	0

INTMOD	Bit	Description		Initial State
INT_RTCT	[22]	0 = IRQ mode	1 = FIQ mode	0
INT_TF	[23]	0 = IRQ mode	1 = FIQ mode	0
EINT2	[24]	0 = IRQ mode	1 = FIQ mode	0
EINT3	[25]	0 = IRQ mode	1 = FIQ mode	0
EINT4/5/6/7	[26]	0 = IRQ mode	1 = FIQ mode	0
INT_ADC	[27]	0 = IRQ mode	1 = FIQ mode	0
EINT8	[28]	0 = IRQ mode	1 = FIQ mode	0
EINT9	[29]	0 = IRQ mode	1 = FIQ mode	0
EINT10	[30]	0 = IRQ mode	1 = FIQ mode	0
EINT11	[31]	0 = IRQ mode	1 = FIQ mode	0

INTERRUPT PENDING REGISTER (INTPND)

In CPU core, there is PSR(Processor Status Register) register, which has several field including the interrupt relating I-Flag and F-Flag. As mentioned above, the CPU can accept two kinds of interrupt even if there are many interrupt sources in S3C3410X. That is why all interrupt sources in S3C3410X should be categorized into two mode, which is IFQ mode and FIQ mode. In this case, if CPU is running the service for certain interrupt, and if this interrupt has IRQ mode, the other interrupt sources with IRQ mode can not be serviced until the completion of current service. These interrupt should be pending in IPR(Interrupt Pending Register). In case of FIQ mode, other FIQ interrupt request can not take CPU while the current FIQ service is running as same as IRQ case. Therefore, the FIQ interrupt request should be pending in IPR as same as IRQ. If IRQ interrupt service is running, the FIQ interrupt can take the CPU for service because FIQ has higher priority than IRQ. In other word, ARM CPU can support two level interrupt architecture. The pending interrupt service can start whenever the I-Flag or F-Flag should be cleared to "0". The service routine should clear the pending bit, also.

Register	Offset Address	R/W	Description	Reset Value
INTPND	0xc004	R/W	Interrupt pending register. Indicates the interrupt request status of each source. 0 = The interrupt has not been requested (when reading) *0 = Clear pending bit (when writing)* 1 = The interrupt source has asserted the interrupt request (when reading) *1 = No effect, keeping current status, '0' or '1'. (when writing)*	0x0

INTPND	Bit	Description		Initial State
EINT0	[0]	0 = Not requested	1 = Requested	0
EINT1	[1]	0 = Not requested	1 = Requested	0
INT_URX	[2]	0 = Not requested	1 = Requested	0
INT_UTX	[3]	0 = Not requested	1 = Requested	0
INT_UERR	[4]	0 = Not requested	1 = Requested	0
INT_DMA0	[5]	0 = Not requested	1 = Requested	0
INT_DMA1	[6]	0 = Not requested	1 = Requested	0
INT_TOF0	[7]	0 = Not requested	1 = Requested	0
INT_TMC0	[8]	0 = Not requested	1 = Requested	0
INT_TOF1	[9]	0 = Not requested	1 = Requested	0
INT_TMC1	[10]	0 = Not requested	1 = Requested	0
INT_TOF2	[11]	0 = Not requested	1 = Requested	0
INT_TMC2	[12]	0 = Not requested	1 = Requested	0
INT_TOF3	[13]	0 = Not requested	1 = Requested	0
INT_TMC3	[14]	0 = Not requested	1 = Requested	0
INT_TOF4	[15]	0 = Not requested	1 = Requested	0
INT_TMC4	[16]	0 = Not requested	1 = Requested	0
INT_BT	[17]	0 = Not requested	1 = Requested	0

INTPND	Bit	Description		Initial State
INT_SIO0	[18]	0 = Not requested	1 = Requested	0
INT_SIO1	[19]	0 = Not requested	1 = Requested	0
INT_IIC	[20]	0 = Not requested	1 = Requested	0
INT_RTCA	[21]	0 = Not requested	1 = Requested	0
INT_RTCT	[22]	0 = Not requested	1 = Requested	0
INT_TF	[23]	0 = Not requested	1 = Requested	0
EINT2	[24]	0 = Not requested	1 = Requested	0
EINT3	[25]	0 = Not requested	1 = Requested	0
EINT4/5/6/7	[26]	0 = Not requested	1 = Requested	0
INT_ADC	[27]	0 = Not requested	1 = Requested	0
EINT8	[28]	0 = Not requested	1 = Requested	0
EINT9	[29]	0 = Not requested	1 = Requested	0
EINT10	[30]	0 = Not requested	1 = Requested	0
EINT11	[31]	0 = Not requested	1 = Requested	0

INTERRUPT MASK REGISTER (INTMSK)

The interrupt mask register has interrupt mask bits for all interrupt source. When an interrupt source mask bit is "0", the corresponding interrupt can not be serviced by the CPU when the corresponding interrupt request is generated. If the mask bit is "1", the interrupt service can be done.

Register	Offset Address	R/W	Description	Reset Value
INTMSK	0xc008	R/W	Interrupt mask register. Each bit can disable or enable the corresponding interrupt request. 0 = Interrupt service is masked or disabled. 1 = Interrupt service is available	0x0

INTMSK	Bit	Description		Initial State
EINT0	[0]	0 = Masked	1 = Service available	0
EINT1	[1]	0 = Masked	1 = Service available	0
INT_URX	[2]	0 = Masked	1 = Service available	0
INT_UTX	[3]	0 = Masked	1 = Service available	0
INT_UERR	[4]	0 = Masked	1 = Service available	0
INT_DMA0	[5]	0 = Masked	1 = Service available	0
INT_DMA1	[6]	0 = Masked	1 = Service available	0
INT_TOF0	[7]	0 = Masked	1 = Service available	0
INT_TMC0	[8]	0 = Masked	1 = Service available	0
INT_TOF1	[9]	0 = Masked	1 = Service available	0
INT_TMC1	[10]	0 = Masked	1 = Service available	0
INT_TOF2	[11]	0 = Masked	1 = Service available	0
INT_TMC2	[12]	0 = Masked	1 = Service available	0
INT_TOF3	[13]	0 = Masked	1 = Service available	0
INT_TMC3	[14]	0 = Masked	1 = Service available	0
INT_TOF4	[15]	0 = Masked	1 = Service available	0
INT_TMC4	[16]	0 = Masked	1 = Service available	0
INT_BT	[17]	0 = Masked	1 = Service available	0
INT_SIO0	[18]	0 = Masked	1 = Service available	0
INT_SIO1	[19]	0 = Masked	1 = Service available	0
INT_IIC	[20]	0 = Masked	1 = Service available	0
INT_RTCA	[21]	0 = Masked	1 = Service available	0

INTMSK	Bit	Description		Initial State
INT_RTCT	[22]	0 = Masked	1 = Service available	0
INT_TF	[23]	0 = Masked	1 = Service available	0
EINT2	[24]	0 = Masked	1 = Service available	0
EINT3	[25]	0 = Masked	1 = Service available	0
EINT4/5/6/7	[26]	0 = Masked	1 = Service available	0
INT_ADC	[27]	0 = Masked	1 = Service available	0
EINT8	[28]	0 = Masked	1 = Service available	0
EINT9	[29]	0 = Masked	1 = Service available	0
EINT10	[30]	0 = Masked	1 = Service available	0
EINT11	[31]	0 = Masked	1 = Service available	0

POWER MANAGEMENT SPECIAL FUNCTION REGISTERS

SYSTEM CONTROL REGISTER (SYSCON)

The system control register is used to control the system operation of the chip.

Register	Offset Address	R/W	Description	Reset Value
SYSCON	0xd003	R/W	System control register	0x0

SYSCON	Bit	Description	Initial State
STOP	[0]	**STOP Control:** This bit value determines whether STOP mode is enabled or disabled. 0 = Normal operation 1 = Entering STOP mode	0
IDLE	[1]	**IDLE Control:** This bit value determines whether IDLE mode is enabled or disabled. 0 = Normal operation 1 = Entering IDLE mode	0
DMA_IDLE	[2]	**DMA IDLE Control:** This bit value determines whether DMA_IDLE mode is enabled or disable. 0 = Normal operation 1 = Entering DMA_IDLE mode	0
CS	[5:3]	**Clock Select:** This field determines frequency of system clock. 000 = MCLK / 16 001 = MCLK / 8 010 = MCLK / 2 011 = MCLK 100 = MCLK / 1024	000
GIE	[6]	**Global Interrupt Enable:** This bit control to enable or disable the interrupt 0 = No requested 1 = Requested	0