**Vectored Interrupt Controller**

* 32 interrupt request input
* 16 vectored IRQ interrupt
* 16 priority levels
* Software interrupt generation
* takes 32 interrupt request and programmably assigns them into three category
  + FIQ
  + vectored IRQ
  + non-vectored IRQ
* **FIQ(Fast Interrupt reQuest)** 
  + highest priority
  + if more than one request is assigned to FIQ, the VIC ORs the request to produce FIQ signal to ARM.
  + if more than one request is assigned to FIQ, the FIQ service routine can read a word from VIC that identifies the FIQ sources.
* **Vectored IRQ**
  + middle priority
  + only 16 request can be assigned to this category – only 16 vectored IRQ sources available
  + But any of the 32 request can be assigned to one of the 16 vectored IRQ sources
  + **slot0** has highest and **slot15** has lowest
* **Non-Vectored IRQs**
  + lowest priority

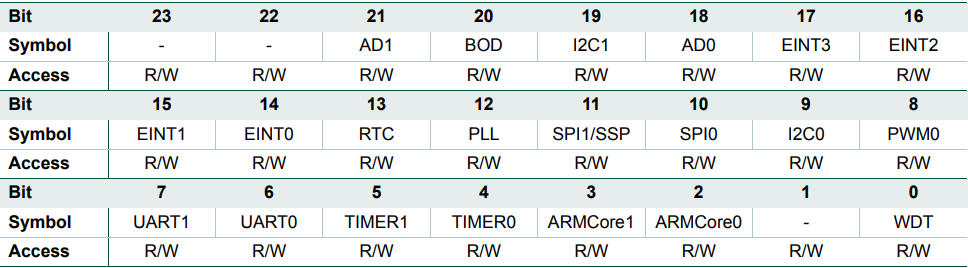
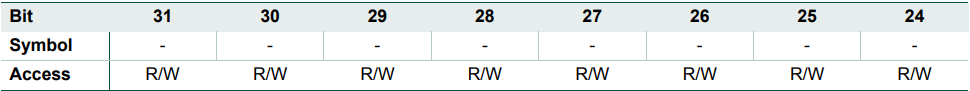
**Note:**

* The VIC ORs the request of all vectored and non-vectored IRQs to produce the IRQ signal to ARM
* The IRQ service routine reads a word register from VIC and jumps there
* VIC provides the highest-priority requesting IRQs service routine else provides default routine

**Register Description**

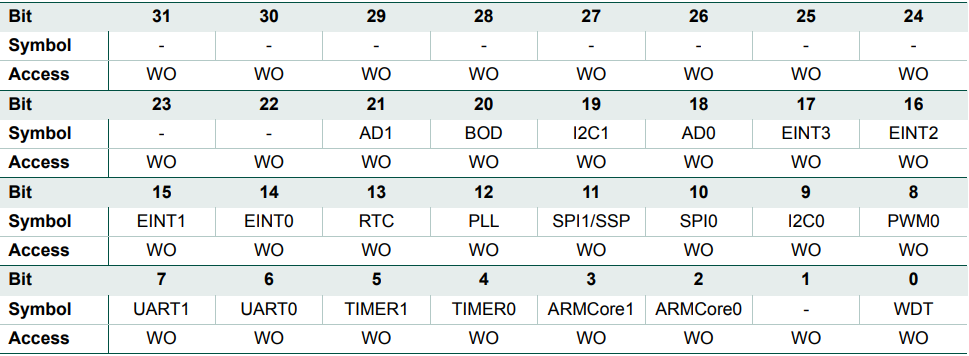
**Software Interrupt register (VICSoftInt - 0xFFFF F018)**

* content of this register are ORed with 32 interrupt request from various peripherals
* Forces the interrupt request with this bit
* Don’t’ do an anything here



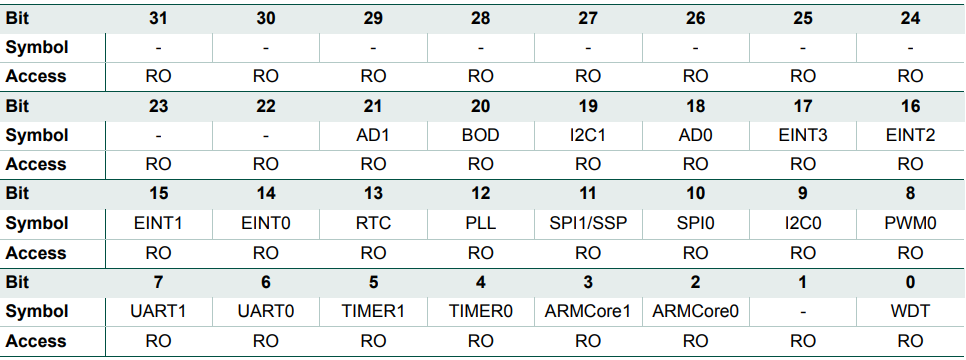
**Software Interrupt Clear register (VICSoftIntClear - 0xFFFF F01C)**

* allows software to clear Software Interrupt register(**VICSoftInt**)
* clears the interrupt of 32 request as needed by writing 1



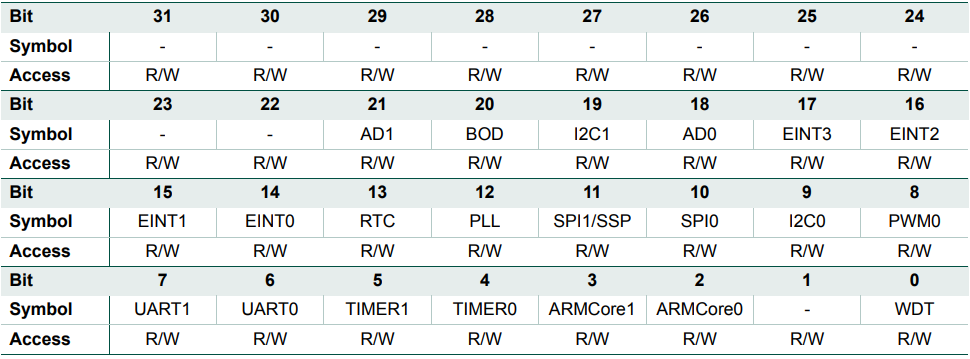
**Raw Interrupt status register (VICRawIntr - 0xFFFF F008)**

* reads the state of 32 interrupt requests and software interrupt
* Do not write



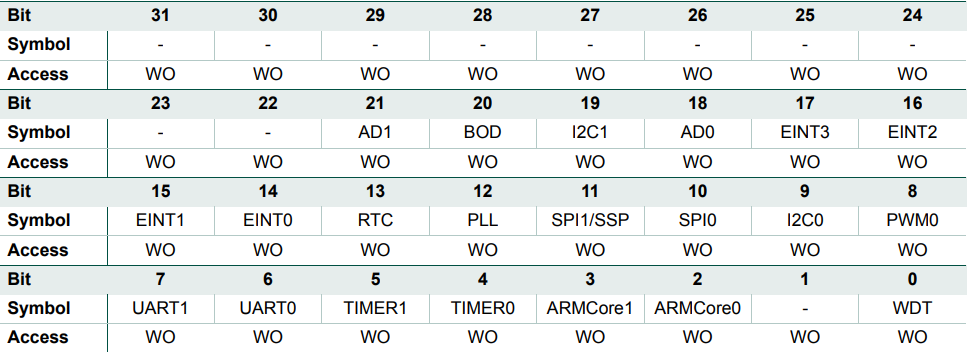
**Interrupt Enable register (VICIntEnable - 0xFFFF F010)**

* Enable the corresponding 32 interrupt request and software interrupts either FIQ or IRQ



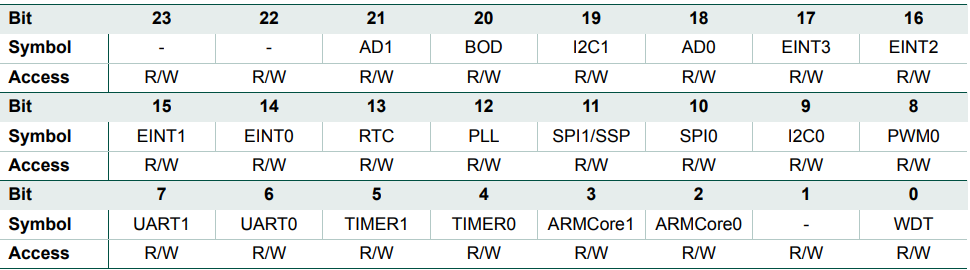
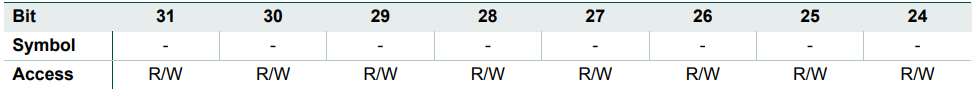
**Interrupt Enable Clear register (VICIntEnClear - 0xFFFF F014)**

* allows software to clear Interrupt enable register(**VICIntEnable**)



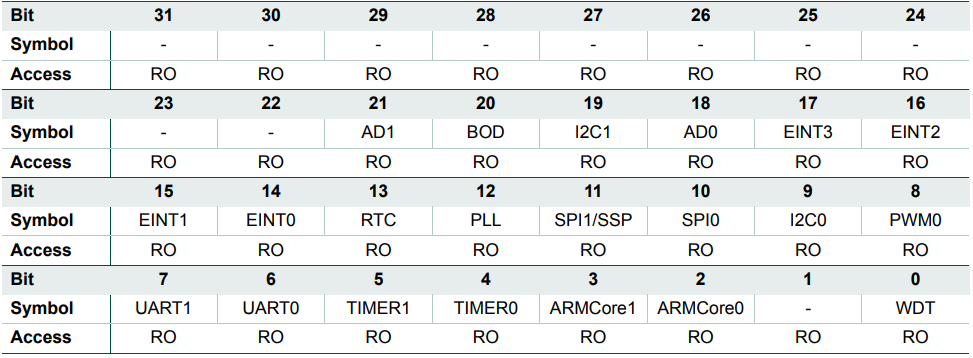
**Interrupt Select register (VICIntSelect - 0xFFFF F00C)**

* classifies each of the 32 interrupt request to either FIQ or IRQ
  + 0 – IRQ
  + 1 – FIQ



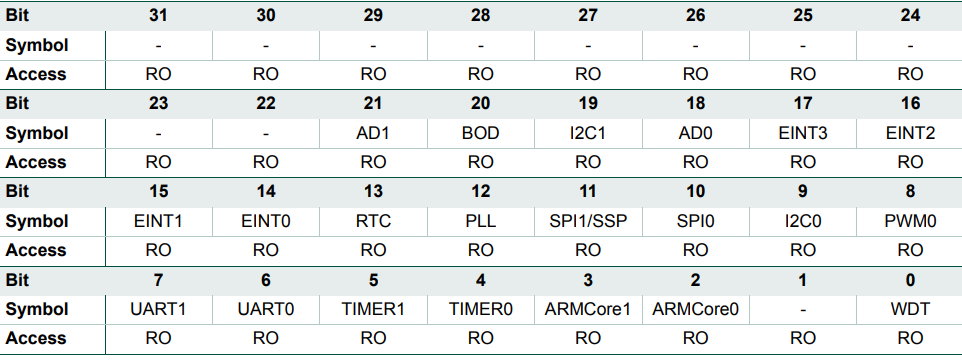
**IRQ Status register (VICIRQStatus - 0xFFFF F000)**

* reads the state of 32 interrupt request that are enabled and classified as IRQ. can be either vectored IRQ or non-vectored IRQ
* 1 indicates that the corresponding interrupt request has been enabled and classified as IRQ and asserted



**FIQ Status register (VICFIQStatus - 0xFFFF F004)**

* reads the state of 32 interrupt request that are enabled and classified as FIQ.
* 1 indicates that the corresponding interrupt request has been enabled and classified as FIQ and asserted



**Vector Control registers 0-15 (VICvectCntl0-15 - 0xFFFF F200-23C)**

* **VICvectCntl0, VICvectCntl1, … ,VICvectCntl14 ,VICvectCntl15**
* controls one of the 16 vectored IRQ slots

|  |  |
| --- | --- |
| **Bits** | **Description** |
| **31:6** | Reserved |
| **5** | IRQslot\_en |
| **4:0** | int\_request/ sw\_int\_assign |

* **IRQslot\_en –** to enable that particular IRQ slot
* **int\_request**
  + the number of the interrupt request assigned to this vectored IRQ slot
  + connect the Interrupt sources to IRQ slots

**eg) to assign SPI0 IRQ interrupt to slot7**

VICVectCntl7 = (1<<5) | 0xA; // enabling and assign SPI0 (interrupt number 10)

**eg) to assign EINT3 IRQ interrupt to slot0**

VICVectCntl0 = (1<<5) | 17; // enabling and assign EINT3 (interrupt number 17)

**Vector Address registers 0-15 (VICVectAddr0-15 - 0xFFFF F100-13C)**

* **VICVectAddr0, VICVectAddr1, …, VICVectAddr15**
* holds address of ISR for the 16 vectored IRQ slots

**Default Vector Address register (VICDefVectAddr - 0xFFFF F034)**

* holds address of ISR for non-vectored IRQs

**Vector Address register (VICVectAddr - 0xFFFF F030)**

* the IRQ service routine can read this register and jump to the value read.

**Protection Enable register (VICProtection - 0xFFFF F020)**

* bit 0
  + 1 – VIC registers can be accessed in privileged mode
  + 0 – VIC registers can be accessed in User mode

**VIC Usage**

1. **FIQ**

* multiple sources can be selected to generate FIQ request(by using VICIntSelect register)
* But, only one service routine should be dedicated for all FIQ’s
* we check the VICFIQStatus to decide which caused interrupt and handle
* after completion of desired ISR, interrupt flag is cleared on peripheral level by writing in VICVectAddr register
* to disbale interrupt, clear the corresponding bit in VICIntEnClr register

1. **IRQ**

* multiple sources can be selected to generate IRQ request(by using VICIntSelect register)
* we can then assign slots to each request by **VICvectCntlx** register
* then, we assign **VICVectAddrx** the vectored-IRQ ISR location or **VICDefVectAddr** for non-vectored-IRQ ISR location
* after completion of desired ISR, interrupt flag is cleared on peripheral level by writing in VICVectAddr register
* to disable interrupt, clear the corresponding bit in VICIntEnClr register