Interrupts

This section describes the specifics of the interrupt handling as performed in ATmega8515. For a general explanation of the AVR interrupt handling, refer to "Reset and Interrupt Handling" on page 13.

Interrupt Vectors in ATmega8515

Table 22. Reset and Interrupt Vectors

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog Reset
2	$001	INT0	External Interrupt Request 0
3	$002	INT1	External Interrupt Request 1
4	$003	TIMER1 CAPT	Timer/Counter1 Capture Event
5	$004	TIMER1 COMPA	Timer/Counter1 Compare Match A
6	$005	TIMER1 COMPB	Timer/Counter1 Compare Match B
7	$006	TIMER1 OVF	Timer/Counter1 Overflow
8	$007	TIMER0 OVF	Timer/Counter0 Overflow
9	$008	SPI, STC	Serial Transfer Complete
10	$009	USART, RXC	USART, Rx Complete
11	$00A	USART, UDRE	USART Data Register Empty
12	$00B	USART, TXC	USART, Tx Complete
13	$00C	ANA_COMP	Analog Comparator
14	$00D	INT2	External Interrupt Request 2
15	$00E	TIMER0 COMP	Timer/Counter0 Compare Match
16	$00F	EE_RDY	EEPROM Ready
17	$010	SPM_RDY	Store Program memory Ready

Notes: 1. When the BOOTRST Fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming" on page 166.
2. When the IVSEL bit in GICR is set, Interrupt Vectors will be moved to the start of the Boot Flash section. The address of each Interrupt Vector will then be the address in this table added to the start address of the Boot Flash section.

Table 23 shows Reset and Interrupt Vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the Interrupt Vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the Interrupt Vectors are in the Boot section or vice versa.

Table 23. Reset and Interrupt Vectors Placement[1]

BOOTRST	IVSEL	Reset Address	Interrupt Vectors Start Address
1	0	$0000	$0001
1	1	$0000	Boot Reset Address + $0001
0	0	Boot Reset Address	$0001
0	1	Boot Reset Address	Boot Reset Address + $0001

Note: 1. The Boot Reset Address is shown in Table 78 on page 177. For the BOOTRST Fuse "1" means unprogrammed while "0" means programmed.

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega8515 is:

```
Address  Labels   Code              Comments
$000              rjmp RESET        ; Reset Handler
$001              rjmp EXT_INT0     ; IRQ0 Handler
$002              rjmp EXT_INT1     ; IRQ1 Handler
$003              rjmp TIM1_CAPT    ; Timer1 Capture Handler
$004              rjmp TIM1_COMPA   ; Timer1 Compare A Handler
$005              rjmp TIM1_COMPB   ; Timer1 Compare B Handler
$006              rjmp TIM1_OVF     ; Timer1 Overflow Handler
$007              rjmp TIM0_OVF     ; Timer0 Overflow Handler
$008              rjmp SPI_STC      ; SPI Transfer Complete Handler
$009              rjmp USART_RXC    ; USART RX Complete Handler
$00a              rjmp USART_UDRE   ; UDR0 Empty Handler
$00b              rjmp USART_TXC    ; USART TX Complete Handler
$00c              rjmp ANA_COMP     ; Analog Comparator Handler
$00d              rjmp EXT_INT2     ; IRQ2 Handler
$00e              rjmp TIM0_COMP    ; Timer0 Compare Handler
$00f              rjmp EE_RDY       ; EEPROM Ready Handler
$010              rjmp SPM_RDY      ; Store Program memory Ready
Handler

$011     RESET:   ldi  r16,high(RAMEND); Main program start
$012              out  SPH,r16      ; Set Stack Pointer to top of RAM
$013              ldi  r16,low(RAMEND)
$014              out  SPL,r16
$015              sei               ; Enable interrupts
$016              <instr>  xxx
...      ...      ...
```

When the BOOTRST Fuse is unprogrammed, the Boot section size set to 2K bytes and the IVSEL bit in the GICR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses is:

```
Address  Labels   Code                 Comments
$000     RESET:  ldi  r16,high(RAMEND); Main program start
$001             out  SPH,r16       ; Set Stack Pointer to top of RAM
$002             ldi  r16,low(RAMEND)
$003             out  SPL,r16
$004             sei                ; Enable interrupts
$005             <instr>  xxx
;
.org $C02
$C02             rjmp EXT_INT0      ; IRQ0 Handler
$C04             rjmp EXT_INT1      ; IRQ1 Handler
...      ....    ..                 ;
$C2A             rjmp SPM_RDY       ; Store Program memory Ready
Handler
```

When the BOOTRST Fuse is programmed and the Boot section size set to 2K bytes, the most typical and general program setup for the Reset and Interrupt Vector Addresses is:

```
Address  Labels   Code               Comments
.org $002
$001             rjmp EXT_INT0      ; IRQ0 Handler
$002             rjmp EXT_INT1      ; IRQ1 Handler
...      ....    ..                 ;
$010             rjmp SPM_RDY       ; Store Program memory Ready
Handler
;
.org $C00
$C00     RESET:  ldi  r16,high(RAMEND); Main program start
$C01             out  SPH,r16       ; Set Stack Pointer to top of RAM
$C02             ldi  r16,low(RAMEND)
$C03             out  SPL,r16
$C04             sei                ; Enable interrupts
$C05             <instr>  xxx
```

When the BOOTRST Fuse is programmed, the Boot section size set to 2K bytes and the IVSEL bit in the GICR Register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses is:

```
Address  Labels   Code                 Comments
.org $C00
$C00             rjmp RESET         ; Reset handler
$C01             rjmp EXT_INT0      ; IRQ0 Handler
$C02             rjmp EXT_INT1      ; IRQ1 Handler
...      ....    ..                 ;
$C10             rjmp SPM_RDY       ; Store Program memory Ready
Handler
;
$C11     RESET:  ldi  r16,high(RAMEND); Main program start
```

```
$C12            out   SPH,r16      ; Set Stack Pointer to top of RAM
$C13            ldi   r16,low(RAMEND)
$C14            out   SPL,r16
$C15            sei                ; Enable interrupts
$C16            <instr>  xxx
```

Moving Interrupts between Application and Boot Space

The General Interrupt Control Register controls the placement of the Interrupt Vector table.

General Interrupt Control Register – GICR

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	INT2	–	–	–	IVSEL	IVCE	GICR
Read/Write	R/W	R/W	R/W	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 1 – IVSEL: Interrupt Vector Select**

When the IVSEL bit is cleared (zero), the Interrupt Vectors are placed at the start of the Flash memory. When this bit is set (one), the Interrupt Vectors are moved to the beginning of the Boot Loader section of the Flash. The actual address of the start of the Boot Flash section is determined by the BOOTSZ Fuses. Refer to the section “Boot Loader Support – Read-While-Write Self-Programming” on page 166 for details. To avoid unintentional changes of Interrupt Vector tables, a special write procedure must be followed to change the IVSEL bit:

1. Write the Interrupt Vector Change Enable (IVCE) bit to one.
2. Within four cycles, write the desired value to IVSEL while writing a zero to IVCE.

Interrupts will automatically be disabled while this sequence is executed. Interrupts are disabled in the cycle IVCE is set, and they remain disabled until after the instruction following the write to IVSEL. If IVSEL is not written, interrupts remain disabled for four cycles. The I-bit in the Status Register is unaffected by the automatic disabling.

Note: If Interrupt Vectors are placed in the Boot Loader section and Boot Lock bit BLB02 is programmed, interrupts are disabled while executing from the Application section. If Interrupt Vectors are placed in the Application section and Boot Lock bit BLB12 is programed, interrupts are disabled while executing from the Boot Loader section. Refer to the section “Boot Loader Support – Read-While-Write Self-Programming” on page 166 for details on Boot Lock bits.

- **Bit 0 – IVCE: Interrupt Vector Change Enable**

The IVCE bit must be written to logic one to enable change of the IVSEL bit. IVCE is cleared by hardware four cycles after it is written or when IVSEL is written. Setting the IVCE bit will disable interrupts, as explained in the IVSEL description above. See Code Example below.

Assembly Code Example

```
Move_interrupts:
  ; Enable change of interrupt vectors
  ldi  r16, (1<<IVCE)
  out  GICR, r16
  ; Move interrupts to boot flash section
  ldi  r16, (1<<IVSEL)
  out  GICR, r16
  ret
```

C Code Example

```
void Move_interrupts(void)
{
  /* Enable change of interrupt vectors */
  GICR = (1<<IVCE);
  /* Move interrupts to boot flash section */
  GICR = (1<<IVSEL);
}
```