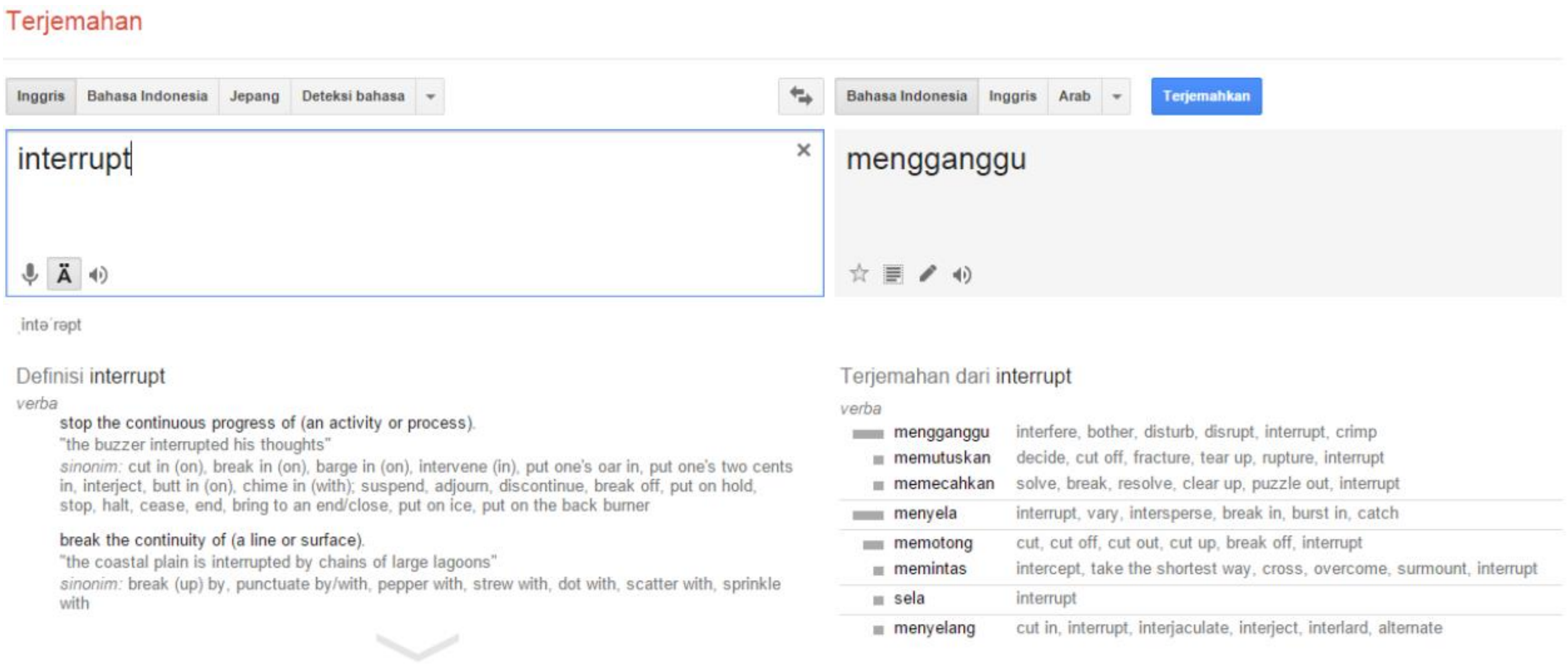
AVR Interrupt

Erdefi Rakun

What is an interrupt ?

Contoh interupsi (interrupt) pada siding paripurna DPR


Terjemahan

Inggris | Bahasa Indonesia | Jepang | Deteksi bahasa

Bahasa Indonesia | Inggris | Arab | Terjemahkan

interrupt

mengganggu

ˌintəˈrəpt

Definisi **interrupt**

verba

stop the continuous progress of (an activity or process).

"the buzzer interrupted his thoughts"

sinonim: cut in (on), break in (on), barge in (on), intervene (in), put one's oar in, put one's two cents in, interject, butt in (on), chime in (with); suspend, adjourn, discontinue, break off, put on hold, stop, halt, cease, end, bring to an end/close, put on ice, put on the back burner

break the continuity of (a line or surface).

"the coastal plain is interrupted by chains of large lagoons"

sinonim: break (up) by, punctuate by/with, pepper with, strew with, dot with, scatter with, sprinkle with

Terjemahan dari **interrupt**

verba

mengganggu	interfere, bother, disturb, disrupt, interrupt, crimp
memutuskan	decide, cut off, fracture, tear up, rupture, interrupt
memecahkan	solve, break, resolve, clear up, puzzle out, interrupt
menyela	interrupt, vary, intersperse, break in, burst in, catch
memotong	cut, cut off, cut out, cut up, break off, interrupt
memintas	intercept, take the shortest way, cross, overcome, surmount, interrupt
sela	interrupt
menyelang	cut in, interrupt, interjaculate, interject, interlard, alternate

What is the meaning of an AVR interrupt ?

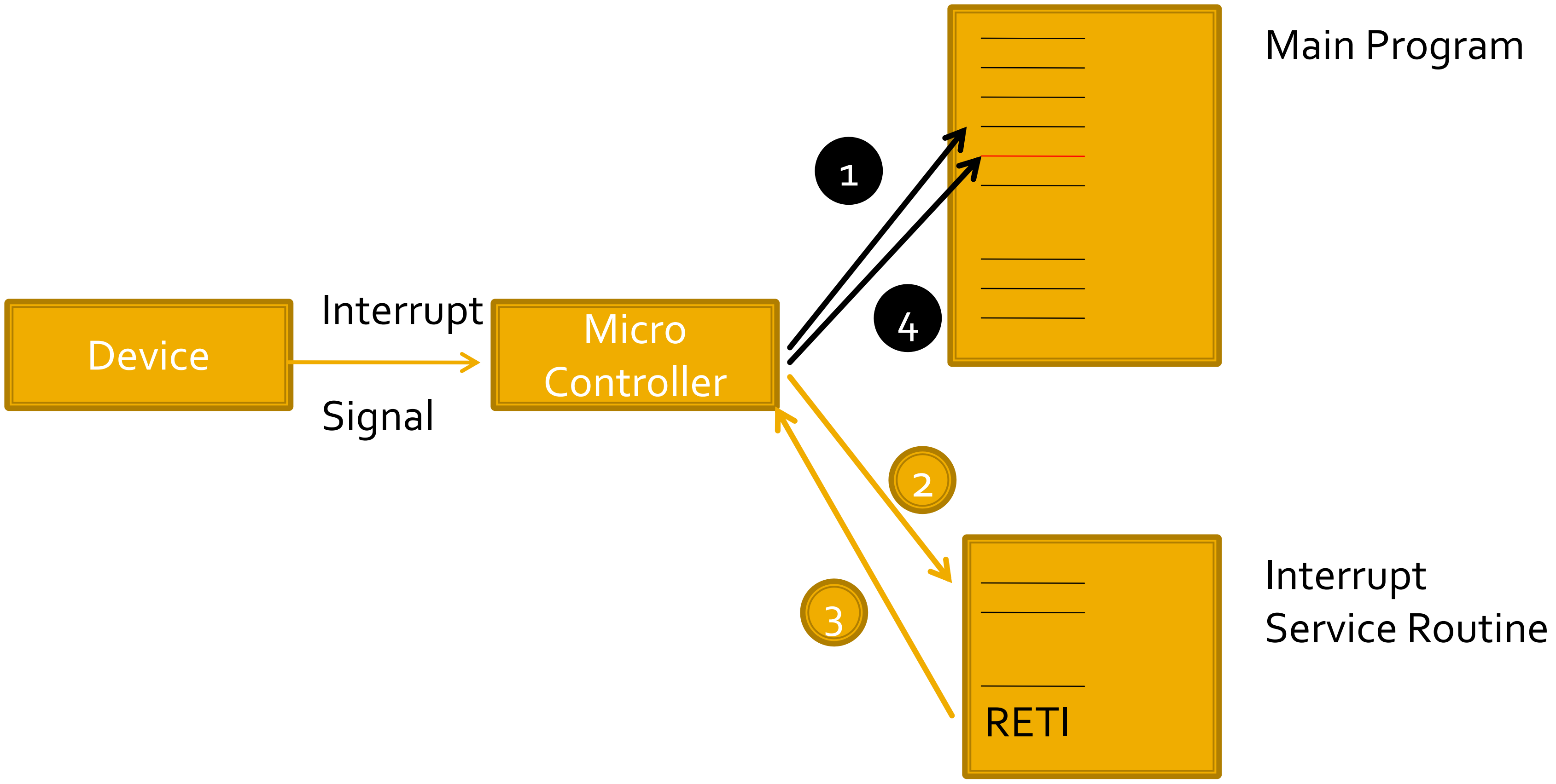
Interrupt in AVR

- An event when any device needs the microcontroller's service, the device notifies it by sending an **interrupt signal**.
- Upon receiving an interrupt signal, the microcontroller stops whatever it is doing and serves the device.
- The program associated with the interrupt is called the **Interrupt Service Routine** (ISR) or **interrupt handler.**
- There are **external** and **internal** interrupts.

Steps in executing an interrupt

- AVR **finishes the instruction** it is **currently** executing and saves the address of the next instruction on the stack
- **Jump to** the address of the interrupt service routine (**ISR**) listed in interrupt vector table
- AVR **starts to execute** the interrupt service subroutine until it reaches the **last instruction** (RETI)
- Upon receiving **RETI**, AVR **returns** to the place where it was interrupted.

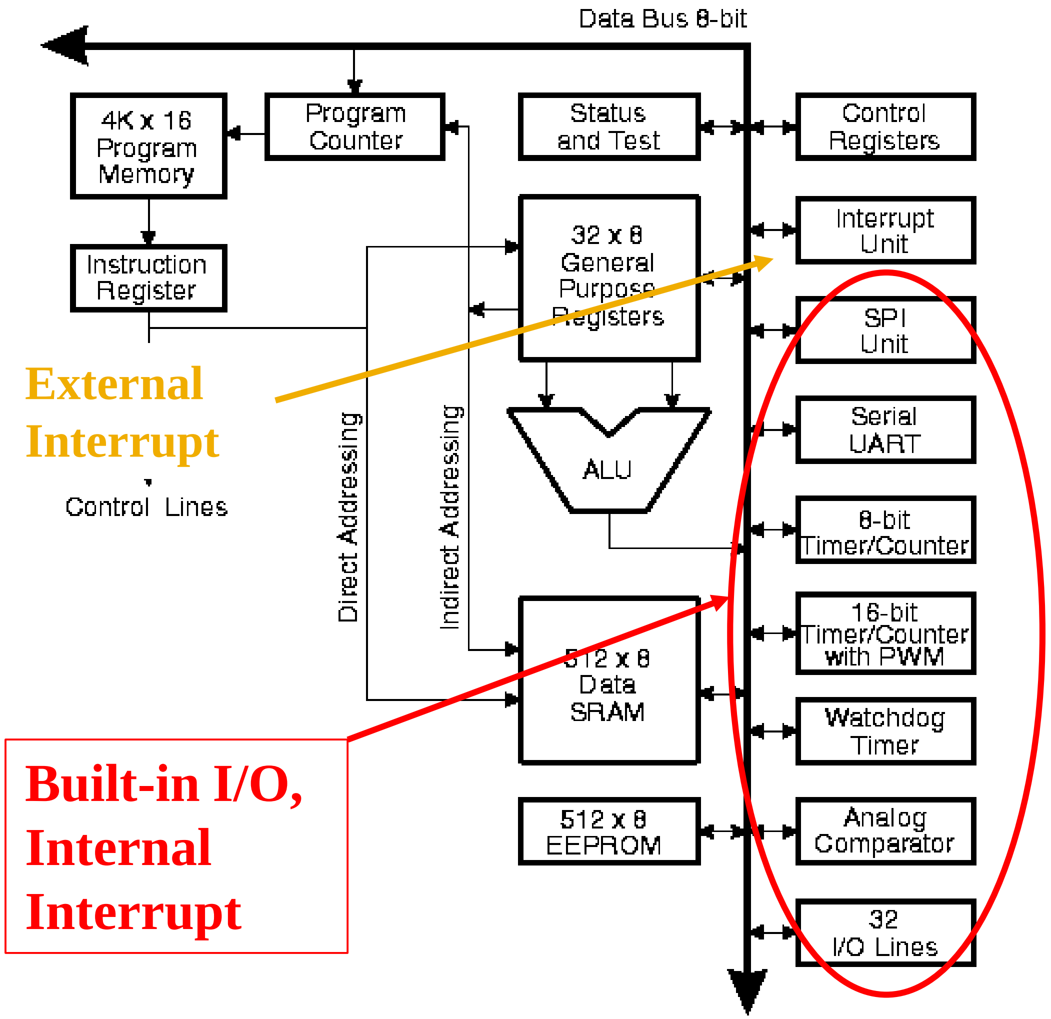
Steps in executing an interrupt



Sources of interrupts in AVR

Covered by ICO:

1. External Interrupt
2. Internal Interrupt / Timer Interrupt



I/O registers

Address	Hex Name	Function
$3F ($5F)	SREG	Status Register
$3E ($5E)	SPH	Stack Pointer High
$3D ($5D)	SPL	Stack Pointer Low
$3B ($5B)	GIMSK	General Interrupt Mask Register
$3A ($5A)	GIFR	General Interrupt Flag Register
$39 ($59)	TIMSK	Timer/Counter Interrupt Mask Register
$38 ($58)	TIFR	Timer/Counter Interrupt Flag Register
$35 ($55)	MCUCR	MCU general Control Register
$33 ($53)	TCCR0	Timer/Counter 0 Control Register
$32 ($52)	TCNT0	Timer/Counter 0 (8-bit)
$2F ($4F)	TCCR1A	Timer/Counter 1 Control Register A
$2E ($4E)	TCCR1B	Timer/Counter 1 Control Register B
$2D ($4D)	TCNT1H	Timer/Counter 1 High Byte
$2C ($4C)	TCNT1L	Timer/Counter 1 Low Byte
$2B ($4B)	OCR1AH	Output Compare Register A High Byte
$2A ($4A)	OCR1AL	Output Compare Register A Low Byte
$29 ($49)	OCR1AH	Output Compare Register B High Byte
$28 ($48)	OCR1AL	Output Compare Register B Low Byte
$25 ($45)	ICR1H	T/C 1 Input Capture Register High Byte
$24 ($44)	ICR1L	T/C 1 Input Capture Register Low Byte
$21 ($41)	WDTCR	Watchdog Timer Control Register

I/O registers (cont.)

Address	Hex Name	Function
$1B ($38)	PORTA	Data Register, Port A
$1A ($3A)	DDRA	Data Direction Register, Port A
$19 ($39)	PINA	Input Pins, Port A
$18 ($38)	PORTB	Data Register, Port B
$17 ($37)	DDRB	Data Direction Register, Port B
$16 ($36)	PINB	Input Pins, Port B
$15 ($35)	PORTC	Data Register, Port C
$14 ($34)	DDRC	Data Direction Register, Port C
$13 ($33)	PINC	Input Pins, Port C
$12 ($32)	PORTD	Data Register, Port D
$11 ($31)	DDRD	Data Direction Register, Port D
$10 ($30)	PIND	Input Pins, Port D
$0F ($2F)	SPDR	SPI I/O Data Register
$0E ($2E)	SPSR	SPI I/O Status Register
$0D ($2D)	SPCR	SPI I/O Control Register
$0C ($2C)	UDR	UART I/O Data Register
$0B ($2B)	USR	UART Status Register
$0A ($2A)	UCR	UART Control Register
$09 ($29)	UBRR	UART Baud Rate Register
$08 ($28)	ACSR	Analog Comparator Control and Status Register

Interrupt Priority

- If two interrupt are activated **at the same time**, the interrupt with the **higher priority** is served first.
- The priority of each interrupt is related to the **address** of the interrupt in the **interrupt vector**.
- To avoid interrupt inside an interrupt, AVR will **disable the I bit** of the SREG when it begins to execute an ISR

Priority and fixed location in memory that holds the addresses of ISRs in ATMega8515

Prioritas Tertinggi

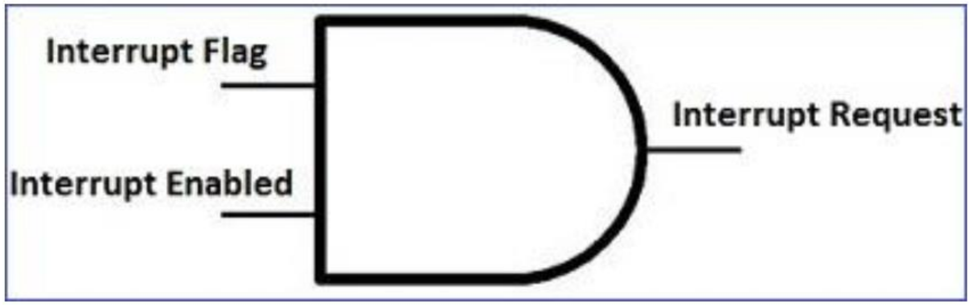
Vec No	Prg Adr	Source	Interrupt Definition
1	$000	RESET	Hardware Pin, Power-on Reset and Watchdog Reset
2	$001	INT0	External Interrupt Request 0
3	$002	INT1	External Interrupt Request 1
4	$003	TIMER1 CAPT	Timer/Counter1 Capture Event
5	$004	TIMER1 COMPA	Timer/Counter1 Compare Match A
6	$005	TIMER1 COMPB	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	UART, RX	UART, RX Complete
11	$00A	UART, UDRE	UART Data Register Empty
12	$00B	UART, TX	UART, TX Complete
13	$00C	ANA_COMP	Analog Comparator

Interrupt Vector table

Address	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 CompareA Handler
$005	rjmp TIM1_COMPB	; Timer1 CompareB 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 UART_RXC	; UART RX Complete Handler
$00a	rjmp UART_DRE	; UDR Empty Handler
$00b	rjmp UART_TXC	; UART TX Complete Handler
$00c	rjmp ANA_COMP	; Analog Comparator Handler

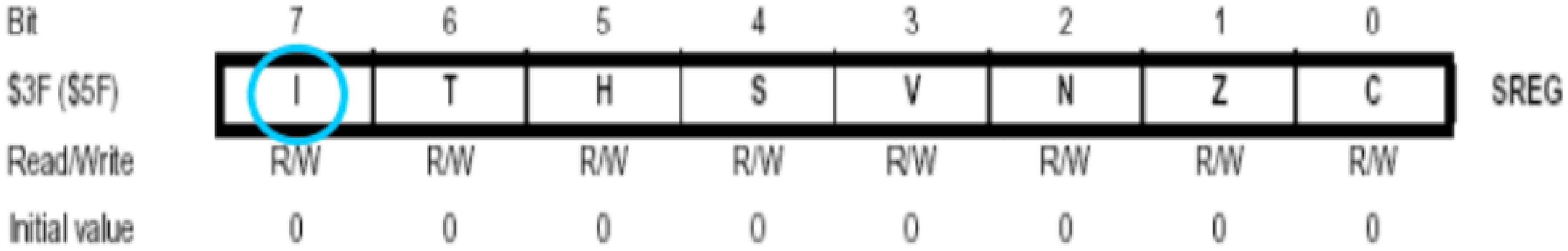
Interrupt Flag and Enable Bit

- **Interrupt flag** is used as a sign that an interrupt has occurred. Interrupt flag = 1 means that there is an interrupt, while interrupt flag = 0 means there is no interrupt. **For example :** Interrupt flag will be set if the button connected to the pin is active (pushed)
- **Interrupts** must be **enabled** by software in order for the **microcontroller to respond to them.**



Steps in enabling an interrupt

- Bit D7 of SREG is the Global Interrupt Enable bit
- Bit D7 of SREG must be set HIGH to allow the interrupts to happen.
- SEI: instruction to activate the interrupt
- CLI: instruction to inactive the interrupt



Bit	7	6	5	4	3	2	1	0	
\$3F (\$5F)	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial value	0	0	0	0	0	0	0	0	

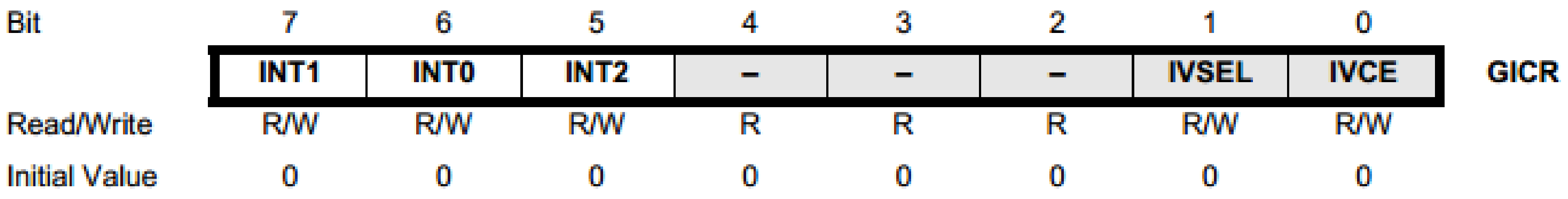
- When an interrupt occurs, Global Interrupt Enable Bit will automatically set to zero.

Steps in enabling an interrupt (cont.)

- There are many sources of interrupts in the AVR. Use the following registers to choose type of interrupts:
- **GICR** (General Interrupt Control Register) is used to choose type of external interrupts.
- **MCUCR** (MCU Control Register) is used to set interrupt activation mode.
- **TIMSK**(Timer/Counter Interrupt Mask Register) is used to choose timer/counter interrupt types.
  (If you use Timer Interrupt)
- After configuring the above registers, then use the **SEI** instruction to enable the interrupt.

Steps in enabling an interrupt (cont.)

- **GICR** (General Interrupt Control Register)



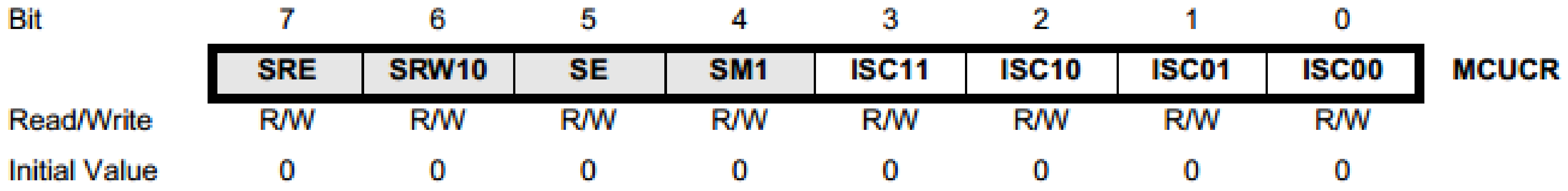
Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	INT2	–	–	–	IVSEL	IVCE	GICR
Read/Write	R/W	R/W	R/W	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – INT1: External Interrupt Request 1 Enable**
- **Bit 6 – INT0: External Interrupt Request 0 Enable**
- **Bit 5 – INT2: External Interrupt Request 2 Enable**
- **IVSEL : Interrupt Vector Select**

  0 → Interrupt vectors are placed at the start of flash memory,
  1 →interrupt vectors are moved to the beginning of the Boot Loader
- **IVCE: Interrupt Vector Change enable**

Steps in enabling an interrupt (cont.)

- **MCUCR** (MCU Control Register)



Bit	7	6	5	4	3	2	1	0	
	SRE	SRW10	SE	SM1	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Pay attention to Bit 3-0

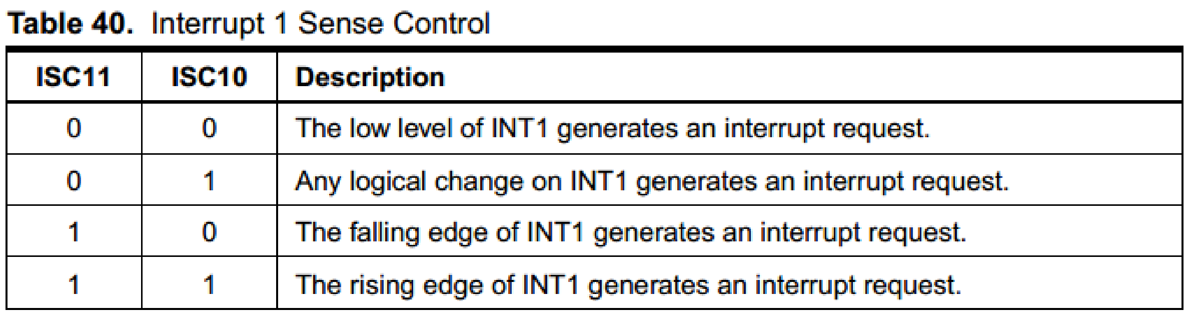


Table 40. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

Bit 3&2
For INT1

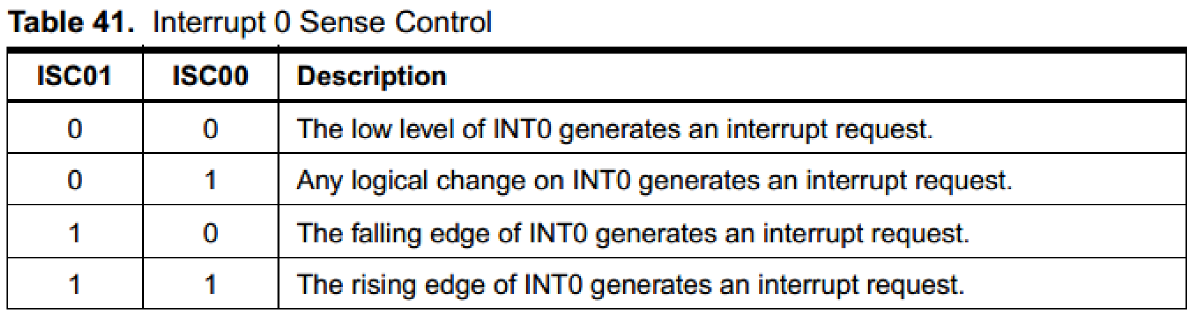


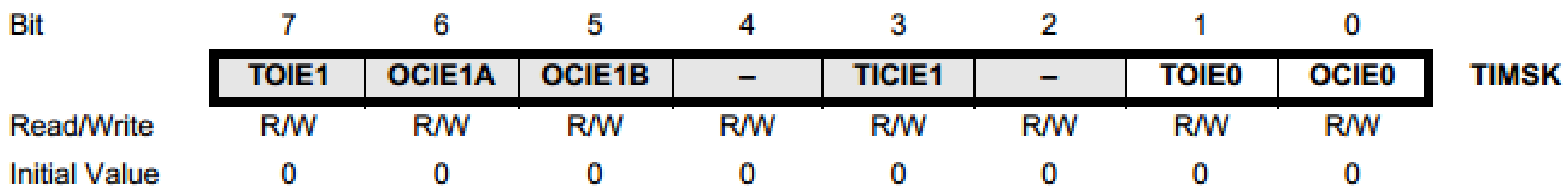
Table 41. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

Bit 1&0
For INT0

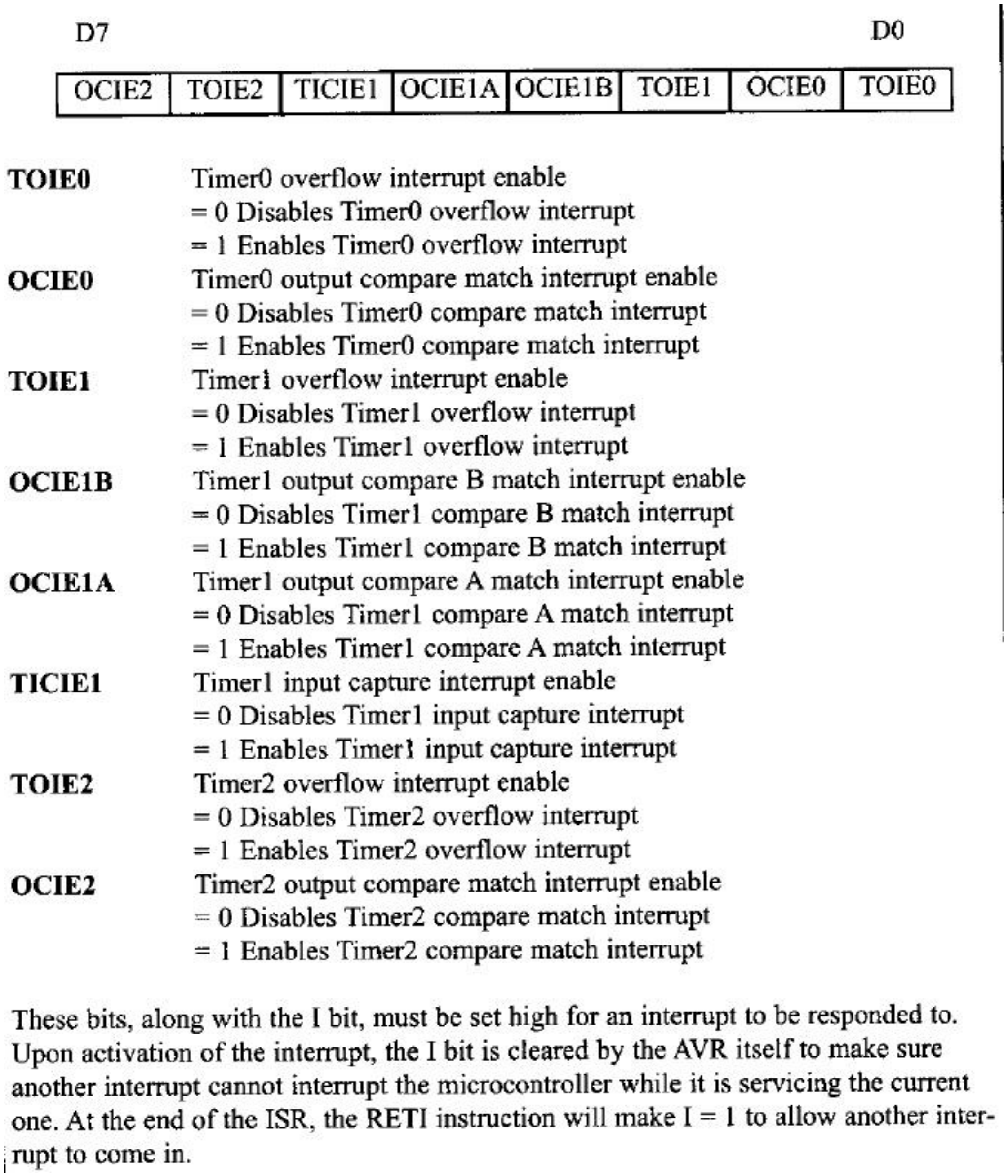
Steps in enabling an interrupt (cont.)

- **TIMSK**(Timer/Counter Interrupt Mask Register)



Bit	7	6	5	4	3	2	1	0	
	TOIE1	OCIE1A	OCIE1B	–	TICIE1	–	TOIE0	OCIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **TOIEx : Timer Counter Overflow x**
- **TOCIEx : Timer Counter Output Compare x**
- **Set the bit(s) if you use timer interrupt**
- **If you only use external interrupt, then you don't need to set this register**



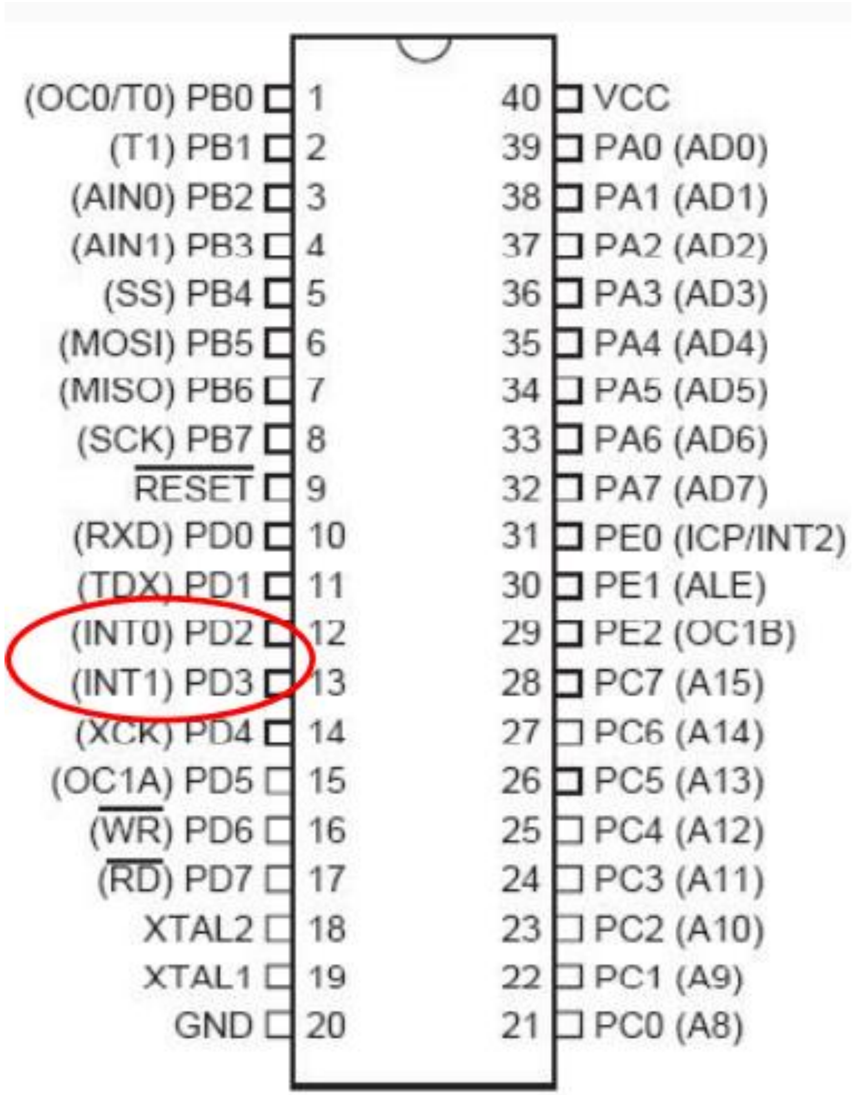
D7							D0
OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0

TOIE0	Timer0 overflow interrupt enable = 0 Disables Timer0 overflow interrupt = 1 Enables Timer0 overflow interrupt
OCIE0	Timer0 output compare match interrupt enable = 0 Disables Timer0 compare match interrupt = 1 Enables Timer0 compare match interrupt
TOIE1	Timer1 overflow interrupt enable = 0 Disables Timer1 overflow interrupt = 1 Enables Timer1 overflow interrupt
OCIE1B	Timer1 output compare B match interrupt enable = 0 Disables Timer1 compare B match interrupt = 1 Enables Timer1 compare B match interrupt
OCIE1A	Timer1 output compare A match interrupt enable = 0 Disables Timer1 compare A match interrupt = 1 Enables Timer1 compare A match interrupt
TICIE1	Timer1 input capture interrupt enable = 0 Disables Timer1 input capture interrupt = 1 Enables Timer1 input capture interrupt
TOIE2	Timer2 overflow interrupt enable = 0 Disables Timer2 overflow interrupt = 1 Enables Timer2 overflow interrupt
OCIE2	Timer2 output compare match interrupt enable = 0 Disables Timer2 compare match interrupt = 1 Enables Timer2 compare match interrupt

These bits, along with the I bit, must be set high for an interrupt to be responded to. Upon activation of the interrupt, the I bit is cleared by the AVR itself to make sure another interrupt cannot interrupt the microcontroller while it is servicing the current one. At the end of the ISR, the RETI instruction will make I = 1 to allow another interrupt to come in.

Pins for External Interrupt in AVR ATMega 8515

Usually connected to Button



Example: Setting External Interrupt

```
ldi r17,0b11000000
out GICR,r17
ldi r17,0b00001010
out MCUCR,r17
sei
```

AVR uses external interrupt1 (INT1) and external interrupt0 (INT0). INT1 and INT0 active at the falling edge

	1	1	0	0	0	0	0	0	
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	

	0	0	0	0	1	0	1	0	
Bit	7	6	5	4	3	2	1	0	
	SRE	SRW10	SE	SM1	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	