

```
i tried to use interrupt INTO of atmega16..
                            i wnt to disable interrupt once the program enters the INTERRUPT SUBROUTINE and enable the
                            interrupt once the ISR gets over. but for the interruot to start for thr first time #asm("sei") needs to
                            be given in the main()...
                            i want to enavle interrut once and then disable it in the ISR..
                            i have given my code below.
avisekdg
                            plz help..
   New Member
Joined: Jun 4, 2010
                            interrupt [EXT_INT0] void ext_int0_isr(void)
              3
Messages:
Likes:
              0
                            #asm("cli");
                            delay_ms(4.752);
                            for(i=0;i<11;i++)
                            PORTC=PIND&&0x04;
                            PORTA=0x00;
                            delay_us(864);
                            PORTA=0x01;
                            delay_us(864);
                            PORTA=0x00;
                            delay_us(864);
                            PORTA=0x02;
                            delay_us(864);
                            PORTA=0x00;
                            void main(void)
                            PORTA=0x00;
                            DDRA=0xFF;
                            PORTB=0x00;
                            DDRB=0x00;
                            PORTC=0x00;
                            DDRC=0xFF;
                            PORTD=0x00;
                            DDRD=0x00;
                            #asm("sei");
                            while (1)
                            // Place your code here
                            };
```



## wkrug

Active Member

Joined: May 30, 2010

Messages: 280

Likes: 29

Location: Germany

The CLI Command will reset the Interrupt Enable Flag in the SREG Register.

At start of an Interupt routine this Flag will be cleared automaticly by hardware without any user command.

When the interrupt routine ends ( RETI Command in Assembler ) this Bit will be set by Hardware again.

So the command #asm ("CLI"); in an interrupt routine has no effect, because it will be set at the end of the interrupt routine again.

To get your program running declare a global bit variable in declaration section: volatile bit ub\_nointerrupt=0;

In your interrupt routine this bit will be set:

ub\_nointerrupt=1;

And include the CLI command in the main routine - That will work if(ub\_nointerrupt>0){#asm("CLI");};

Thats not a Problem of CodeVision AVR, it appears at every Compiler Software, bacause it will be occoured by Hardware.

To lock an specific interrupt, it will be eaisier to reset the according interrupt enable bits. When enable this bits again write a logical 1 to the according interrupt **flag**.

So the interrupt will not startet immediately at re- enabling the interrupt enable bit.

Last edited: Jun 5, 2010

Like x 1

## n 7, 2010



## avisekdg

New Member

Joined: Jun 4, 2010
Messages: 3
Likes: 0

## wkrug said: ↑

The CLI Command will reset the Interrupt Enable Flag in the SREG Register. At start of an Interupt routine this Flag will be cleared automaticly by hardware without any user command.

Milhan tha interment reciting and a / DETT Command in Assemblar I this Dit will be set by

Click to expand...

thanks a lot for your help sir..

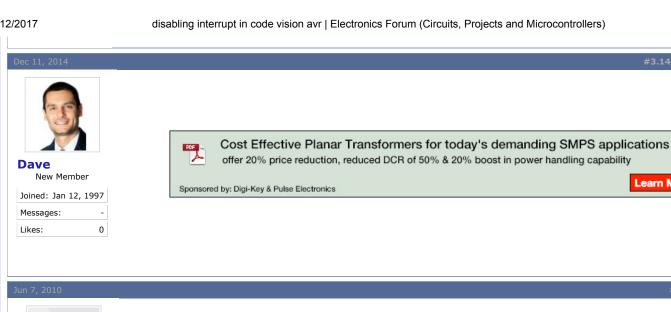
but the code is still not working properly..the interrupt is being enabled again and again.. i am using INTO of atmega16 and it is falling egde triggered..

at the INTO pin the input data is in the form of a bit stream comprising of 1 and 0..so while the subroutine is being executed if a falling edge is encountered at INTO pin then the subroutine is being executed again..

i am writing my code below..please help..

```
volatile bit ub_nointerrupt=0;
interrupt [EXT_INT0] void ext_int0_isr(void)
int i;
ub_nointerrupt=1;
delay_ms(4.752);
for(i=0;i<11;i++)
PORTC=PIND&&0x04;
PORTA=0x00;
delay_us(864);
PORTA=0x01;
delay_us(864);
PORTA=0x00:
delay_us(864);
PORTA=0x02;
delay_us(864);
PORTA=0x00:
```

```
void main(void)
// Declare your local variables here
// Input/Output Ports initialization
// Port A initialization
// Func7=Out Func6=Out Func5=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
DDRA=0xFF;
// Port B initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTB=0x00;
DDRB=0x00;
// Port C initialization
// Func7=Out Func6=Out Func5=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;
// Port D initialization
// Func7=In Func6=In Func5=In Func4=In Func3=In Func2=In Func1=In Func0=In
// State7=T State6=T State5=T State4=T State3=T State2=T State1=T State0=T
PORTD=0x00;
DDRD=0x00;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00:
TCNT2=0x00;
OCR2 = 0x00;
GICR|=0x40;
MCUCR=0x02;
MCUCSR=0x00;
GIFR=0x40;
TIMSK=0x00;
ACSR=0x80;
SFIOR=0x00;
// Global enable interrupts
#asm("sei");
while (1)
};
#if(ub_nointerrupt>0)
#asm("cli");
};
#endif
}
```







thanks a lot for your help sir..

but the code is still not working properly..the interrupt is being enabled again and again.. i am using INTO of atmega16 and it is falling egde triggered..

at the INTO pin the input data is in the form of a bit stream comprising of 1 and 0..so while the subroutine is being executed if a falling edge is encountered at INTO pin then the subroutine is being executed again..

i am writing my code below..please help..

When the Interrupt follows are so short it would be able to disabler the int0 enable flag in the GICR register in the int0 interrupt routine:

GICR&=0b10111111;

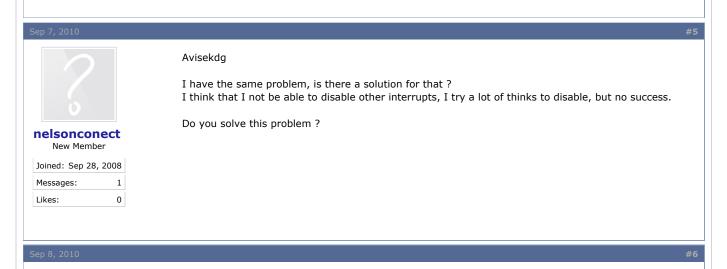
So the Interrupt will be disabled and no furher trigger is allowed.

To avoid a false Trigger signal at re- enabling of the flag in the GICR Register.

Write a logical 1 to GIFR that will clear the according flag:

GIFR|=0b01000000;

GICR|=0b01000000;



The interrupt FLAG always been set when the interrupt condition is true.

The solution if for most Interrupts the same.

before re-enabling the interrupt.

Disable the according interrupt control register and write a logical 1 to the according flag register

But only when the inerrupt CONTROL Flag is active too, an interrupt appears.

Learn More

17/12/2017

disabling interrupt in code vision avr | Electronics Forum (Circuits, Projects and Microcontrollers)

The consequence is, first to delete the FLAG before enabling the interrupt, by the CONTROL bit an the assembler comand "SEI".

To clear an interrupt FLAG write a logical 1 to it.

Some interrupt FLAGS will be set, when the Function is enabled. E.g. The  $\ensuremath{\mathsf{USART}}$ 

(You must log in or sign up to reply here.)



Joined: May 30, 2010

Messages: Likes:

Location:

280

Germany

Tweet

G+ Recommend Be the first of your friends to recommend this.

BNC75 Series RF Connector

Hirose offers their BNC75 series, 75 Ω, 12G-SDI standard compliant, easy-on, easy-off bayonet lock system

Sponsored by: Digi-Key & Hirose

Learn More

Part Search



Discover 28,000+ devices from industry-leading brands, ready to ship today!

Learn More

S News element 14



Discover 28,000+ devices from industry-leading brands, ready to ship today!

Learn More

Seem elementiu

Full Width

Forums

Electronics Forums Microcontrollers

AVR

Privacy Policy. Copyright ©2017 WTWH Media, LLC. All Rights Reserved.

Contact Us Help Home Top

Terms and Rules