

Re: 16F628 Interrupt problem – help!

Source: <http://sci.tech–archive.net/Archive/sci.electronics.design/2005–01/2360.html>

From: Mark Jones (*abuse_at_127.0.0.1*)

Date: 01/08/05

Date: Sat, 08 Jan 2005 12:26:05 -0500

Anthony Fremont wrote:

> "Rob" <rdsfal@yahoo.com.au> wrote in message
> news:41df92f8\$0\$29408\$5a62ac22@per–qv1–newsreader–01.iinet.net.au...
>
>
>> Snipped original post content
>>
>>
>>> ~ Your LED2_PULSE and LED1_OFF, what do they do?
>>>
>>> ~ The optical encoder, is it measuring something fast? Precise?
>>> Or is it just being used as a knob? (100ms sample rate is much
>>> easier to deal with than 2kHz.)
>>>
>>> ~ The general rule of thumb regarding ISR's & PICs is A) 95% of
>>> the time they are not needed, B) they should be very short,
>>> and C) all port activity has to be double–buffered. (Think,
>>> what happens if an interrupt occurs while a port is being
>>> modified?)
>>>
>>>
>>> ~ That said, if the encoder is suppling a very fast interrupt,
>>> what happens if another interrupt occurs before the line ~ BCF
>>> INTCON,INTE ;DISABLE RBO INTERRUPT ~ ?
>>>
>>
>>
>> Thanks for the reply Mark.
>>
>> The LEDs are just being turned on/off for test purposes. The
>> encoder runs very slowly – it is to be used to measure a slowly
>> changing mechanical position, << 500 Hz encoder pulse rate (500
>> slot encoder wheel). The application is battery powered hence the
>> need for the extra glue logic to power up and sample the encoder
>> periodically.
>>
>> Re point C, can you please elaborate on "double buffering" port
>> read/writes – I don't understand what you mean by this.

- >
- >
- > *He was referring to keeping a "shadow" ram location that represents*
- > *the value of an output port. You then set and reset individual*
- > *bits within the shadow register and move the whole shadow register*
- > *to the output port, instead of flipping bits directly on the output*
- > *port. This avoids potential Read/Modify/Write issues having to do*
- > *with the way the PIC ports are designed. These issues are*
- > *generally associated with highly loaded output pins and/or fast*
- > *clock speeds.*
- >
- > *When you flip an output bit on a PIC, the whole port is read, a*
- > *single bit is modified, and the whole port is re-written. With*
- > *heavily loaded pins, or in the case of not allowing enough settling*
- > *time after changing a pins state, the pin state can be misread. It*
- > *will consequently be re-written with the incorrect value resulting*
- > *in the appearance of output pins mysteriously changing themselves*
- > *when other pins on the same port are changed.*
- >
- >
- >> *I was of the understanding that as soon as an interrupt occurs*
- >> *the GIE bit was cleared – preventing further interrupts being*
- >> *registered, so the problem of an interrupt occurring before the*
- >> *line "BCF INTCON,INTE" would not be an issue – am I incorrect on*
- >> *this?*
- >
- >
- > *Your interpretation is correct, Mark appears to be somewhat*
- > *unfamiliar with PICs.*

No, it's just that I avoid ISR's because of all the inherent trouble with them. It's been so long, I'd forgotten GIE was automatically cleared. It's just as easy most of the time to check for things in the main program loop anyways.

Also check out JAL, it's a friendly pascal-like language for PICs (which will do assembler also if you like.)
<http://groups.yahoo.com/group/jallist/> – binary from
sourceforge.net/projects/jal/

- > *In fact your clearing and setting of INTE in*
- > *the ISR is not really necessary, since no other interrupts will be*
- > *serviced until GIE is set again (by retfie).*
- >
- > *Be aware that the PIC's stack is quite small. Your ISR is using*
- > *two of the eight locations available.*
- >

Also sorry about the strange message formatting... was tinkering with PGP stuff.

sci.electronics.design: Re: 16F628 Interrupt problem – help!

— "The one truth of the universe is: infinite recursion. That is all."