

Re: Interrupt driven UART

Source: <http://coding.derkeiler.com/Archive/General/comp.arch.embedded/2006-10/msg00762.html>

- *From:* Ico <usenet@xxxxxxx>
 - *Date:* 12 Oct 2006 06:36:06 GMT
-

goister@xxxxxxxx <goister@xxxxxxxx> wrote:

Hi,

I'm working with a Toshiba TMP91 series MCU that doesn't seem to have any UART control/status bits to check for empty data register, rx/tx ready, etc, but does have interrupt vectors for serial tx and rx, which is why I think I have to use interrupt driven UART rather than polling it.

I have functions that expect to receive and send a single byte by calling receivebyte and sendbyte functions. However, since it's interrupt based, receivebyte seems to be pretty redundant. What I have now is a UART receive ISR that first does error checking, then copies the rx buffer to a global rx byte variable, and setting a global rx_ready flag to 1. Then my receivebyte function does nothing until the rx ready flag is set, after which it just clears it. Does this make sense?

This might work, but imagine what would happen if a byte is received on the uart while your code happens to be doing something else then calling the uart_receive function ? Instead of having only one 'global rx variable', consider using a ringbuffer aka 'circular buffer' aka fifo for this. The RX interrupt stores incoming bytes into the buffer and updates the head and tail pointers, while your uart_receive function reads one byte from the buffer, or – if the buffer is empty – waits until new data comes available.

—
:wq
^X^Cy^K^X^C^C^C
.