DESIGN NOTE #027

AUTHOR: BJØRN BIRKELAND, (BJORN@SCANMATIC.NO)

KEYWORDS: OCR, FSK, ICP

A Simple FSK-modem, Frequence Measuring and Frequence Generation Technique

This design note shows how to utilize the Input Capture and Output Compare capabilities of the 16-bit Timer T1 to implement a FSK-modem. By hard-wiring the TXD and RXD pins of the internal UART together with the FSK-modem you get a inexspensive and simple modem communication engine.

Basically this design is based on the two main functions: Frequence measuring and frequence generation.

The design has been tested and verified on an ATmega8 running at 11.0592 MHz.

Principle, Demodulator

Measure the frequence of the received FSK-signal on an input pin, (ICP) and if the frequence mathches the “space frequence” set demodulator output to space-level. If the measured frequence mathces the mark frequence’, set demodulator output to mark-level. If the measured frequence do not match either mark or space, then assume it is a noise triggered interrupt and let the de-modulator output level stay at current level.

Implmentation, Demodulator

Set the 16-bit Timer T1 in Free Running mode and activate the Input Capture Interrupt function. Wether you select falling or rising edge capture depends on which edges in the signal that gives the “cleanest edge”. Also activation of Input Capture Noise Canceller should normally be a good choice here.

Now at each capture interrupt, read the captured value and subtract the value with the previous captured value. Now you have a “count” proportional to the period of the receiving signal. If this period matches the period of the mark or space frequence than set the demodulator output to the corresponding level. Else if no match leave the demodulator output at current level. Remember to save the captured value in a “previous_capture” before leaving the interrupt routine.

Demodulator Code Example

```
/* FSK-demodulator */

#define MARGIN_CNT 15

SIGNAL (SIG_INPUT_CAPTURE1)
{
        static W16 prev_capture = 0;
        W16        capture;
        W16        delta;

        capture = __inw(ICR1L);
        delta = capture - prev_capture;
        if ( ( delta >= (RX_FREQUENCE_MARK_CNT - MARGIN_CNT) ) &&
             ( delta <= (RX_FREQUENCE_MARK_CNT + MARGIN_CNT) ) ){
                mark();
        } else if ( ( delta >= (RX_FREQUENCE_SPACE_CNT - MARGIN_CNT) ) &&
                    ( delta <= (RX_FREQUENCE_SPACE_CNT + MARGIN_CNT) ) ){
                space();
        }
        prev_capture = capture;
}
```

Principle, Modulator

If modulator input is at space level, then generate space-frequence on the output pin (OCR1A). Else generate mark-frequence at output pin.

Implmentation, Modulator

Set the 16-bit Timer T1 in Free Running mode (already done if demodulator is active) and activate the “output-toggeling at compare match” function. Also enable the Compare Match interrupt. Now in the Compare Match interrupt service routine one has to pre-load the Compare Match value with a new value which goes “n” clocks a head so that the output is toggled at the right time to generate the mark or space frequence. “n” clocks will thus have one value for space frequence and another value for the mark frequence generation.

new compare match = old compare match + n

Sampling an input pin (modulator input) decides which frequence to produce.

Modulator Code Example

```
/* FSK-modulator */
SIGNAL( SIG_OUTPUT_COMPARE1A )
{
        if ( inp(PINB) & BIT_5) {
                __outw(__inw(OCR1AL) + TX_FREQUENCE_MARK_CNT, OCR1AL);
        } else {
                __outw(__inw(OCR1AL) + TX_FREQUENCE_SPACE_CNT, OCR1AL);
        }
}
```

Connecting the Internal UART/ USART to the FSK Modem

By wiring the UART TXD pin to the input pin of the modulator and wiring the output pin of the demodulator to the UART RXD pin you have a simple UART/USART based FSK Modem. For writing UART driver routines I would recommend ATMELS application note “AVR306: Using the AVR UART in C”.

I have not tried it out, but maybe it is possible to sample the UART TXD pin directly without the need for wiring the output to another input. Similar there might exist a way to feed the UART directly without the need for an external wiring from the demodulator output to the RXD input. If that's possible we could free up two IO-pins.

Another solution would be to write a simple SW-based UART. Thats no problem especially since the baudrate would be relatively low in this system.

Other Hint Tips

Since this design requires the 16-bit Timer in Free Running mode, its Overflow Interrupt could be used as a system-tick source. Also note that the 16-bit Timer has two Output Compare Registers. The FSK Modem only use one of them. Thus the other one is still free and can be used for general frequnce generation tasks.