Algorithm for zero-crossing detector frequency determination

David R. Andersen

November 11, 2005

Here's a quick summary of the best zero-crossing algorithm that I've seen for decoding NRZI-encoded Bell 202 data. It uses TMR0 to determine the real-time state of the incoming frequency in the ISR. The frequency of the incoming wave is updated by the ISR at each zero-crossing.

Independently in the main code, TMR1 is used to time a bit-determination period. The frequency information is sampled once per bit period (although not always at the same point in the bit period), and NRZI bits are rolled into your packet buffer based on the TMR1 loop samples.

ISR Code:

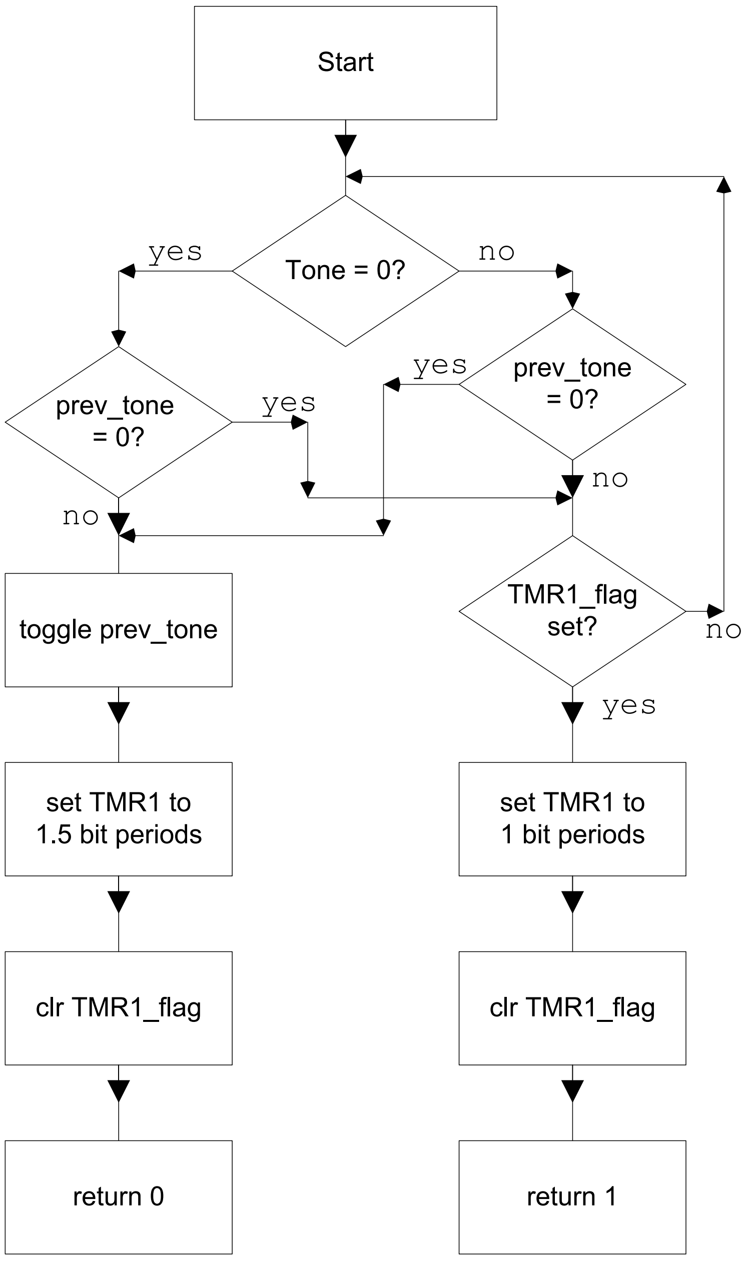
Use TMR0 to count time between zero crossings. Use the RBIE interrupt facility to determine every state change on the input pin. The input pin can be any of pins PORTB, 4 or 5 or 6 or 7. Assume you use PORTB, 4. Thus, the ISR would look something like this:

> Watch for change on *e.g.* PORTB,4 – when change occurs, the RBIE fires and you enter the ISR – which executes as follows:
>
> 1. On change, compare TMR0 count to the count corresponding to 1700 Hz.
>   a. If the count is greater, set TONE=LOW
>   b. If the count is less, set TONE=HIGH
> 2. Clear TMR0
> 3. Return from ISR
>
> What this gives you is a register TONE that has the current real-time frequency information in it. TONE will be updated at each zero-crossing, independent of the bit period. The ISR should be written in assembly language for fastest execution.

Main Code:

Once each bit period, check TONE to see if it has changed state from the last bit period. If it has, roll a 0 into your buffer. If it has not, roll a 1 into your buffer. The attached flow chart shows one algorithm that can be used to determine the bit stream coming from your input signal. It uses TMR1 to create a time window in which to determine whether the input frequency has changed or not. If the time window runs out without a change in frequency, a 1 is returned, and if a change in frequency occurs before timeout, a 0 is recorded (NRZI).

Then go ahead and work the HDLC decode magic on your buffer, keeping track of bit-stuffing, FCS, etc.



Bell 202 zero-crossing detector. There are three flags used. Tone is the instantaneous tone, prev_tone is the tone from the previous bit period, and TMR1_flag is the overflow flag for TMR1. TMR1 is a counter set to overflow in 1 or 1.5 bit periods as shown in the flowchart.

The flag Tone is assumed to be instantaneously updated (presumably by an ISR) at each zero-crossing with the frequency of the tone segment that was present immediately prior to that zero-crossing.