Time Measurement of an Interrupt

An Application Note

www.arium.com

Overview

Developers of drivers and embedded systems often need to measure the elapsed time between events to insure adequate system performance. Typically, these are events that can be detected on the bus such as memory accesses, I/O accesses, and interrupts. American Arium's TRC-xx emulators feature a bus analyzer with a bus cycle time stamp having 10 nanosecond resolution that can be used to measure time between events.

Measuring an Interrupt's Interval

The following example will demonstrate how an American Arium TRC emulator can be utilized to measure the time interval between IRQ0 interrupts. On standard PC platforms, IRQ0 is driven by the equivalent of an 8253 Programmable Interval Timer. This timer divides a 1.19318 MHz signal by 2^{16}, yielding an interrupt every 54.925 milliseconds.

Specifying a Qualifier

The first step in measuring the interval between these IRQ0 interrupts is to specify a qualifier. Qualifiers specify a particular type of bus cycle and determine what is captured in trace. A maximum of two qualifiers may be specified (i.e., Q1 and Q2). Since IRQ0 utilizes an interrupt vector of 08h, we need to specify a single qualifier (Q1) on an interrupt acknowledge of vector 08h as shown in Figure 1.

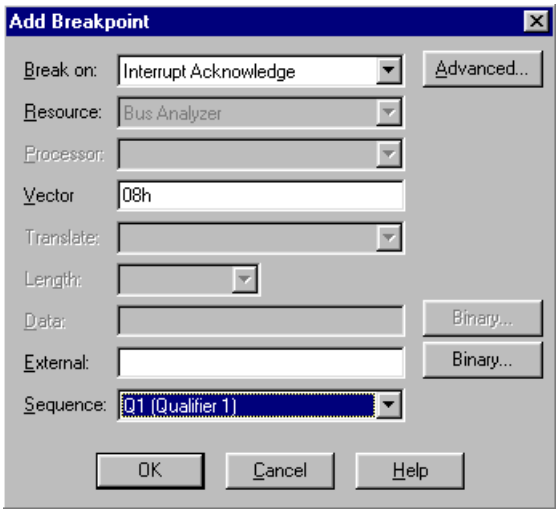

Figure 1: Qualifier 1 (Q1)

Specifying A Qualification Mode To Limit Trace Recording

After defining the qualifier (Q1), the next step to designate how it is to be utilized by specifying the data qualification mode of "record only Q1" as shown in Figure 2. This instructs the bus analyzer to record only the Q1 events previously specified. Keep in mind that regardless of how the bus analyzer records bus activity, it always occurs at full bus speed and never inserts delays of any kind.

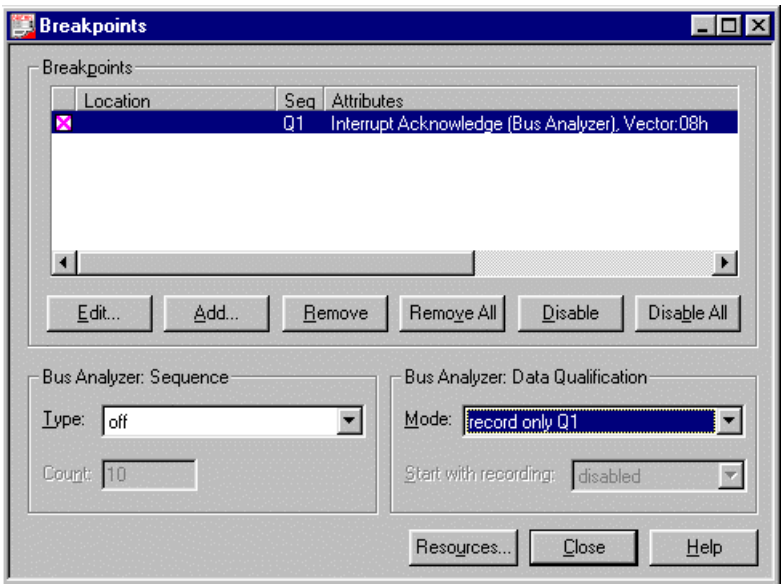

Figure 2: Breakpoints Window

Run, Capture, And View the Interrupt Interval Trace

Finally, we simply need to let the target system run for a few seconds while the bus analyzer captures the Q1 cycles. Figure 3 is a trace window that shows the last seven Q1 cycles that were recorded while the processor was running. Note that in this case, the time stamp on the right side is accumulative and indicates the time elapsed relative to the trigger point (the point at which the processor stopped).

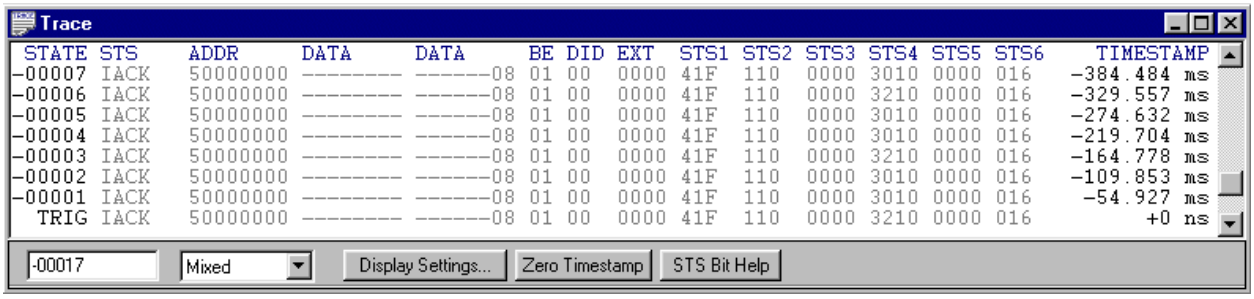

Figure 3: Trace Window with an Accumulative Time Stamp

By clicking on the Display Settings button at the bottom of the window, the time stamp can be changed to the delta mode. In delta mode the time stamp indicates the elapsed time between recorded cycles. The time stamp mode can be changed at any time, even after the trace has been captured. Figure 4 shows the same window with the time stamp changed to the delta mode.

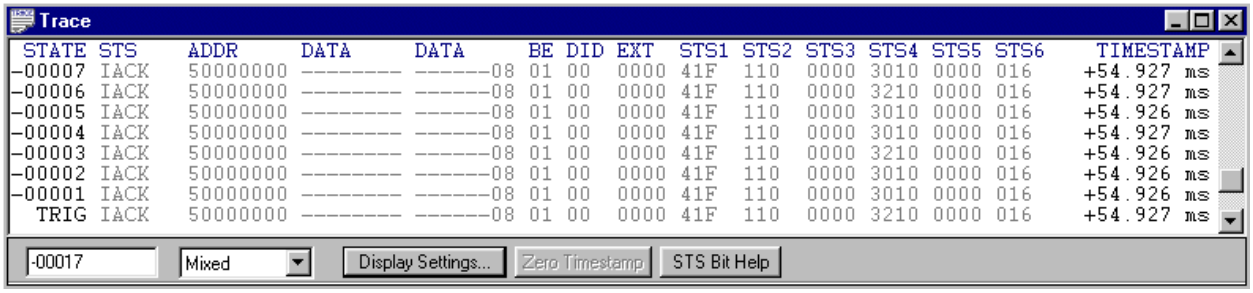

Figure 4: Trace Window with A Delta Time Stamp

Measuring an Interrupt's Duration

In the previous example we measured how often the IRQ0 interrupt occurs. In the next example we will measure the time spent inside the IRQ0 interrupt service routine. We already know that the interrupt starts with an interrupt acknowledge and a little bit of investigation reveals that in this case, it ends with an IRET instruction preceded by a write of 20h to the programmable interrupt controller at I/O port 20h. Therefore, we want to trace from the interrupt acknowledge to the interrupt return instruction and then stop. We will accomplish this by using an additional qualifier (Q2) with a qualification mode of "record on Q1, stop on Q2".

Specifying a Second Qualifier

We will retain the Q1 qualifier from the previous example (Figure 1) which was specified as a vector 08h interrupt acknowledge. Next, we will define a Q2 qualifier that specifies a write of 20h to I/O port 20h as shown in Figure 5. The I/O write to the controller is a better choice than the IRET because the I/O write always appears on the bus while the IRET might become cached and not appear on the bus.

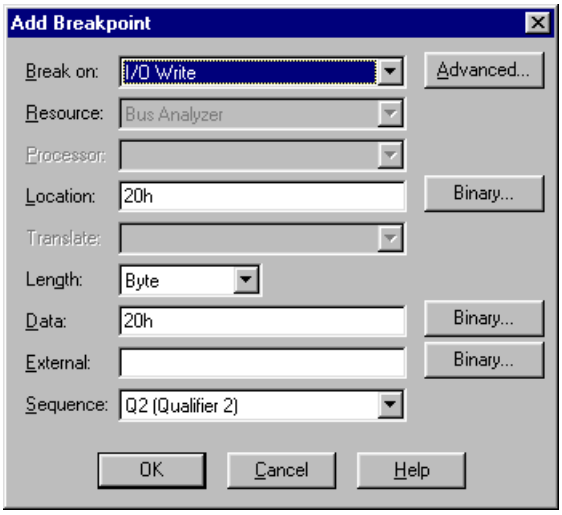

Figure 5: Qualifier 2 (Q2)

Specifying a Breakpoint

Next we will create a breakpoint term to stop the processor on the write to the interrupt controller at I/O port 20h occurs. Terms are similar to qualifiers in that they specify a particular type of bus cycle. They differ from qualifiers in that they are used to stop the processor rather than to control trace recording. Figure 6 shows a Term 1 (T1) bus analyzer breakpoint on this event.

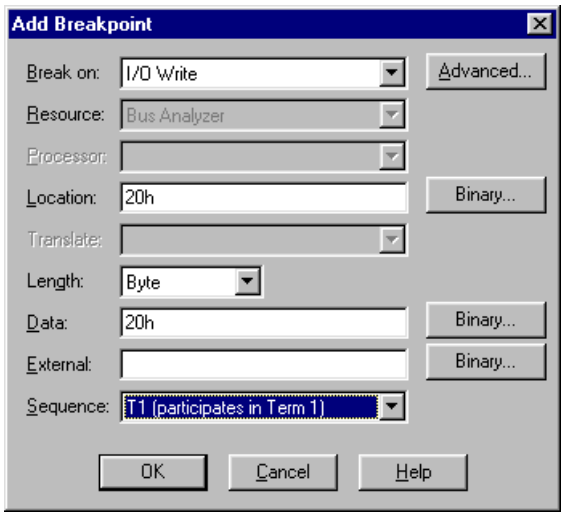

Figure 6: Processor Breakpoint

Specifying a Qualification Mode to Selectively Record Trace

After defining the breakpoint (T1) and both qualifiers (Q1 and Q2), the next step is to designate how the qualifiers are to be utilized by selecting a data qualification mode of "record on Q1, stop on Q2" as shown in Figure 7. This instructs the bus analyzer to begin recording when a match is encountered for the Q1 qualifier and stop recording when a match is encountered for the Q2 qualifier.

We have also specified a bus analyzer sequence of "count of T1" that breaks on the 10th occurrence of the T1 breakpoint. Combined with the qualification mode, this will allow us to capture ten iterations of the interrupt service routine in trace.

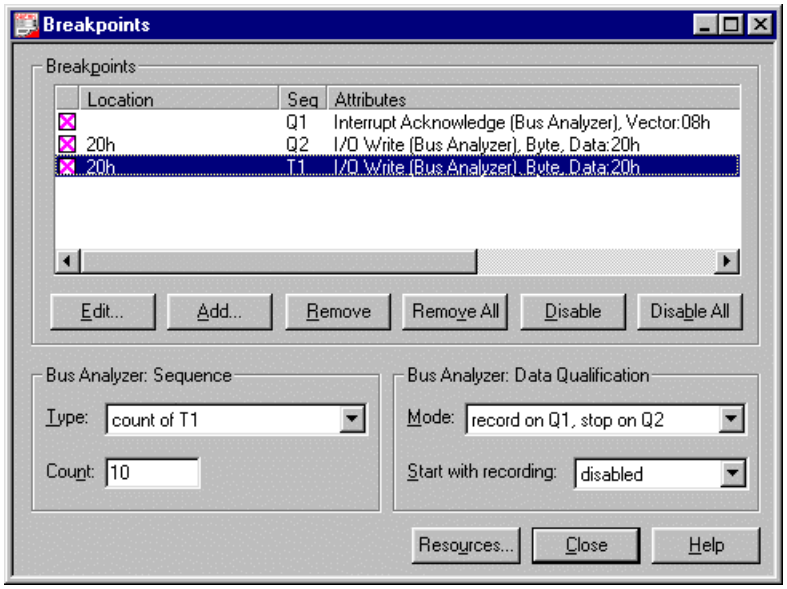

Figure 7: Breakpoints Window

Run, Capture, And View the Interrupt Duration Trace

Finally, we simply need to let the target system run while the bus analyzer repeatedly captures everything that occurs between the Q1 and Q2 qualifiers. Figure 8 is a trace window that shows the last of ten captured iterations of the interrupt service routine. Since the window currently shows the last iteration of the interrupt service routine, you will note that the code in now completely cached and no fetches occurred. As you can easily see, the entire time spent within the service routine was 4.070 µs.

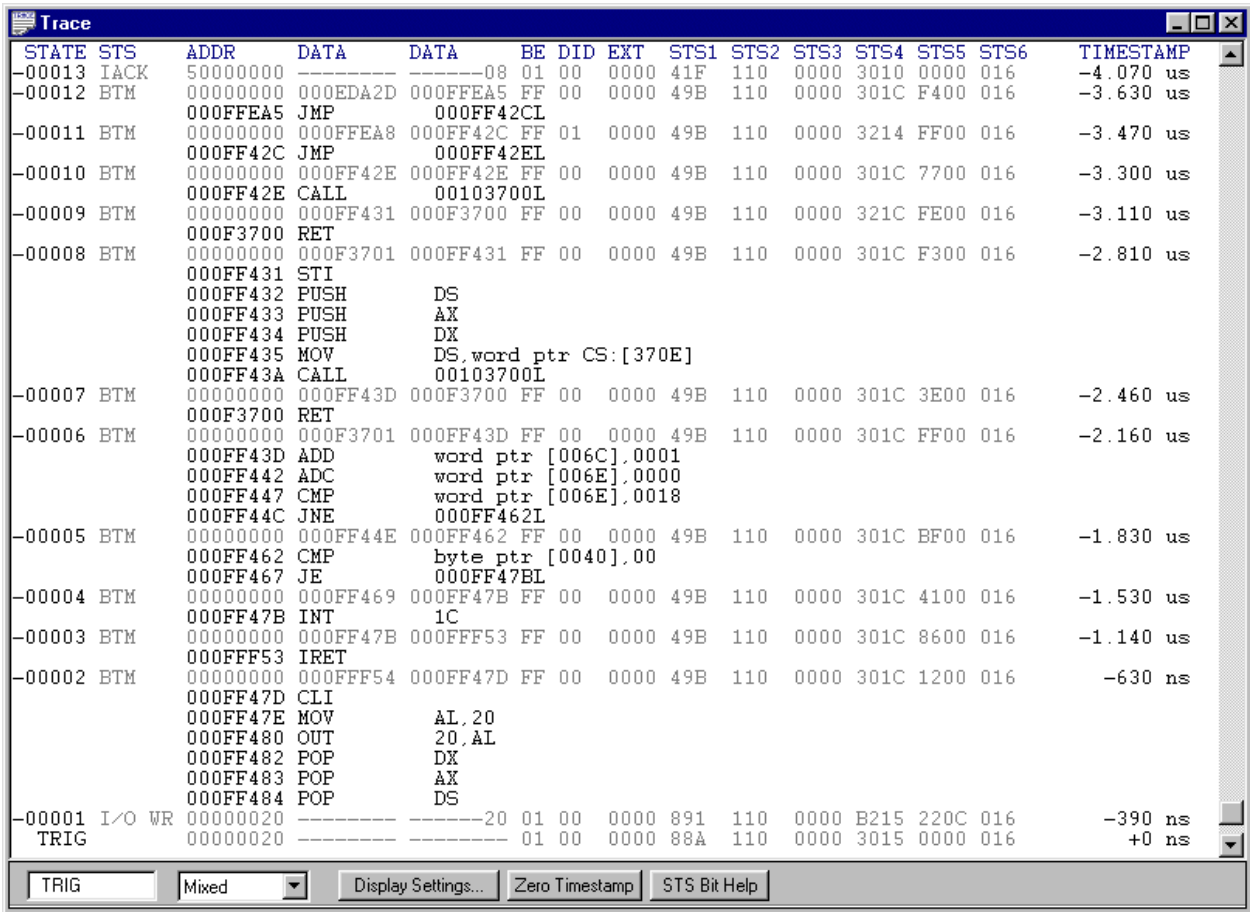

Figure 8: Trace With an Accumulative Time Stamp Showing the Interrupt's Duration

14281 Chambers Road
Tustin, CA 92780
Voice: 714-731-1661
Fax: 714-731-6344
Web: www.arium.com
E-mail: info@arium.com