Brian Burton

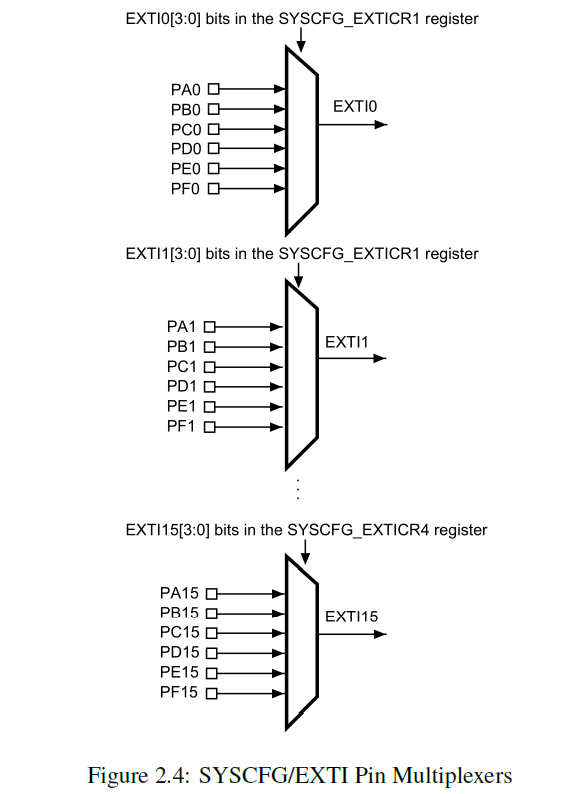
ECE 5780

2/16/2022

**Post Lab 02 (Intro/GPIO)**

1. Why can’t you use both pins PA0 and PC0 for external interrupts at the same time?

The reason that you cannot use pins PA0 and PC0 for external interrupts at the same time is because the hardware denies it. Multiplexers only enables one input at a time. Since PA0 and PC0 are inputs for the same multiplexer, they cannot run simultaneously.



1. What software priority level gives the highest priority? What level gives the lowest?

There are for priority level options, zero, one, two, and three. Therefore zero would the highest priority level, which in return means that three gives the lowest priority level.

Because the NVIC within the STM32F0 has two configuration bits for each interrupt’s priority, four

software priority levels are available. The CMSIS library functions accept a numeric value in the range

of [0-3] as allowed priority levels. The lower the numerical value given, the higher the precedence

assigned to the interrupt by the NVIC.

1. How many bits does the NVIC have reserved in its priority (IPR) registers for each interrupt

(including non-implemented bits)? Which bits in the group are implemented?

The NVIC has four 8-bit regions reserved in it’s priority (IPR) registers for each interrupt. The uppermost two bits from the regions are implemented.

Each IPR register contains four 8-bit regions to set the priority of an interrupt; the NVIC

within the STM32F0 only has the uppermost two bits from these regions implemented,

giving four possible configurable priority levels (0-3). More bits are available on other

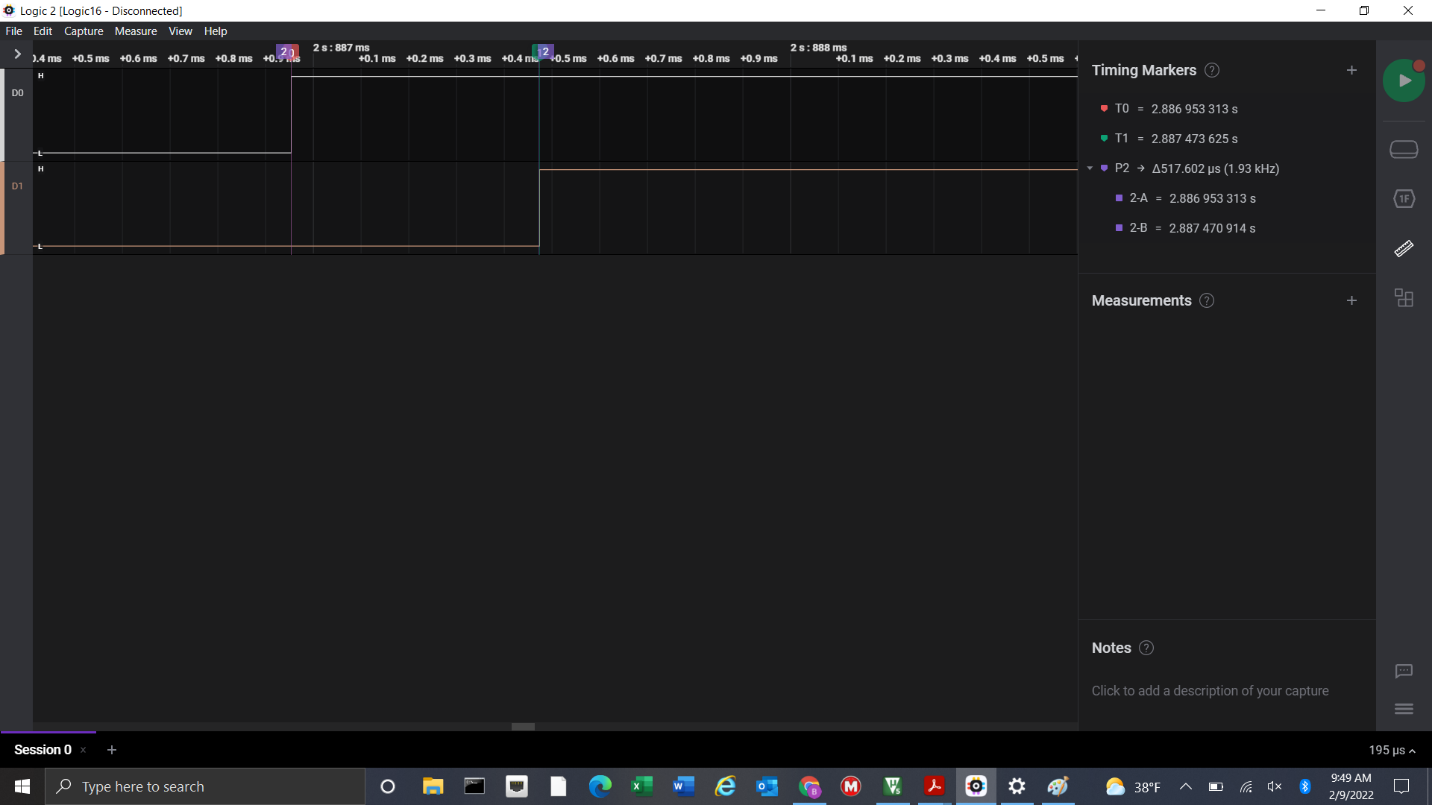
chipsets.

1. What was the latency between pushing the Discovery board button and the LED change

(interrupt handler start) that you measured with the logic analyzer? Make sure to include a

screenshot in the post-lab submission.

The latency between pushing the Discovery board button and the LED change was 517.602µs.



1. Why do you need to clear status flag bits in peripherals when servicing their interrupts?

You need to clear status flag bits in peripherals when servicing their interrupts because if do don’t, the interrupt will repeat continuously. Once the flag is cleared, the program then knows that the peripheral has completed.

Most peripherals have a status register containing flag bits for pending interrupt requests; however,

even in those without dedicated registers, most interrupts set status flags within their peripheral. These

flags are necessary to generate interrupt requests. Typically you will need to clear the matching status

bit manually for the interrupt condition that you are handling; otherwise, the interrupt will repeat

continuously because the request never acknowledges as complete.