ECE 560: EMB. SYS. ARCHITECTURES

PROJECT 1 REPORT

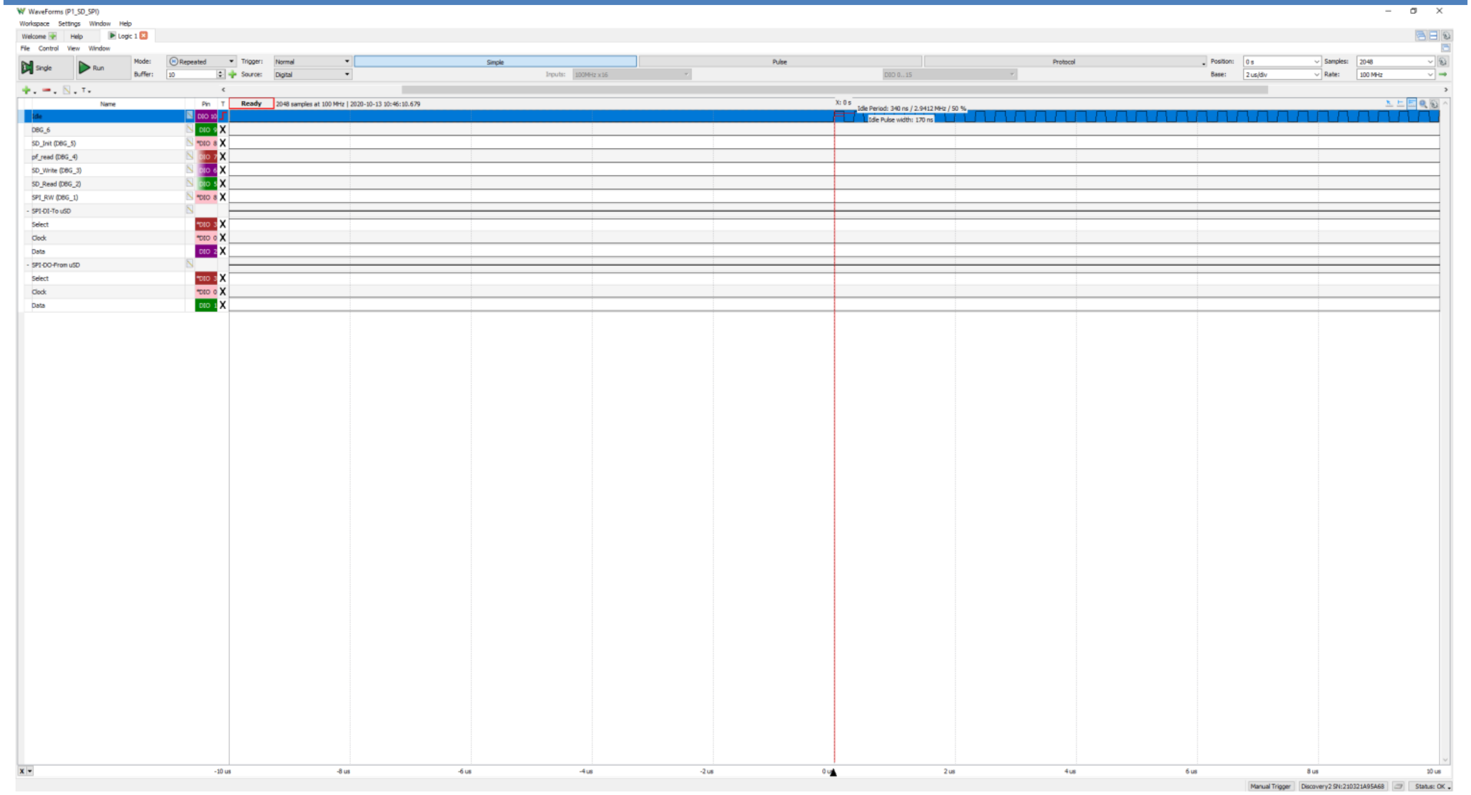
Cristian Hellmer

crhellme

INTRODUCTION

This project involved optimizing a program which displays a series of images on an LCD display and displays stats about the program's execution. To optimize the program, I was responsible for gaining an understanding of the existing system through a series of tests using the logic analyzer. I was then responsible for implementing a way for threads obtain a precise delay in which the idle thread could be utilized. This was then used to optimize SD reading process to take advantage of times where the function is simply waiting for an event.

ANALYZE IDLE THREAD TIMING


WaveForms (P1_SD_SPI)

Mode: Repeated
Buffer: 10
Trigger: Normal
Source: Digital
Inputs: 100MHz x16
Position: 0 s
Base: 2 us/div
Samples: 2048
Rate: 100 MHz

Ready 2048 samples at 100 MHz | 2020-10-13 10:46:10.679

X: 0 s
Idle Period: 340 ns / 2.9412 MHz / 50 %
Idle Pulse width: 170 ns

Manual Trigger | Discovery2 SN:210321A95A68 | Status: OK

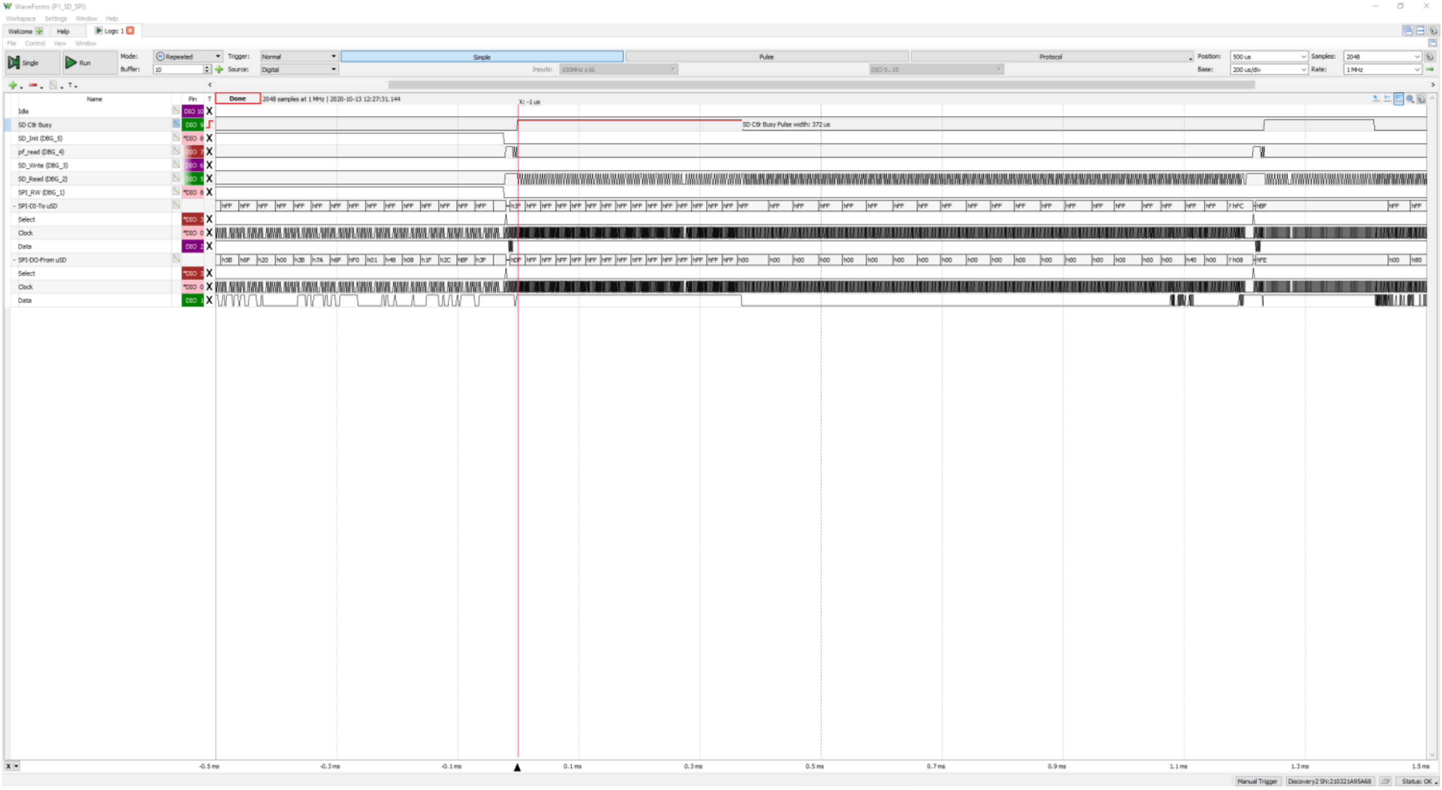
- Answer the following questions. The screenshot must support your answers.
  - How much time does it take for the loop in osRtxIdleThread to execute one iteration?

170ns

  - How many iterations would happen in one millisecond if only the idle thread ran, and nothing else?

5882

ANALYZE SD TIMING


WaveForms (P1_SD_SPI)

Mode: Repeated
Buffer: 10
Trigger: Normal
Source: Digital
Position: 500 us
Base: 200 us/div
Samples: 2048
Rate: 1 MHz

Done 2048 samples at 1 MHz | 2020-10-13 12:27:31.144

SD Ctlr Busy Pulse width: 372 us

- Logic analyzer screenshot showing a full read transaction (sending command, waiting for controller to read block, reading data back over SPI). Trigger on the rising edge of SD Ctlr Busy. Measure how long SD Ctlr Busy is a 1. Include this number and the screenshot in your report.

372us

- What are the values of the statistics reported on the display? What do they indicate? How do they relate to the length of SD Ctlr Busy?

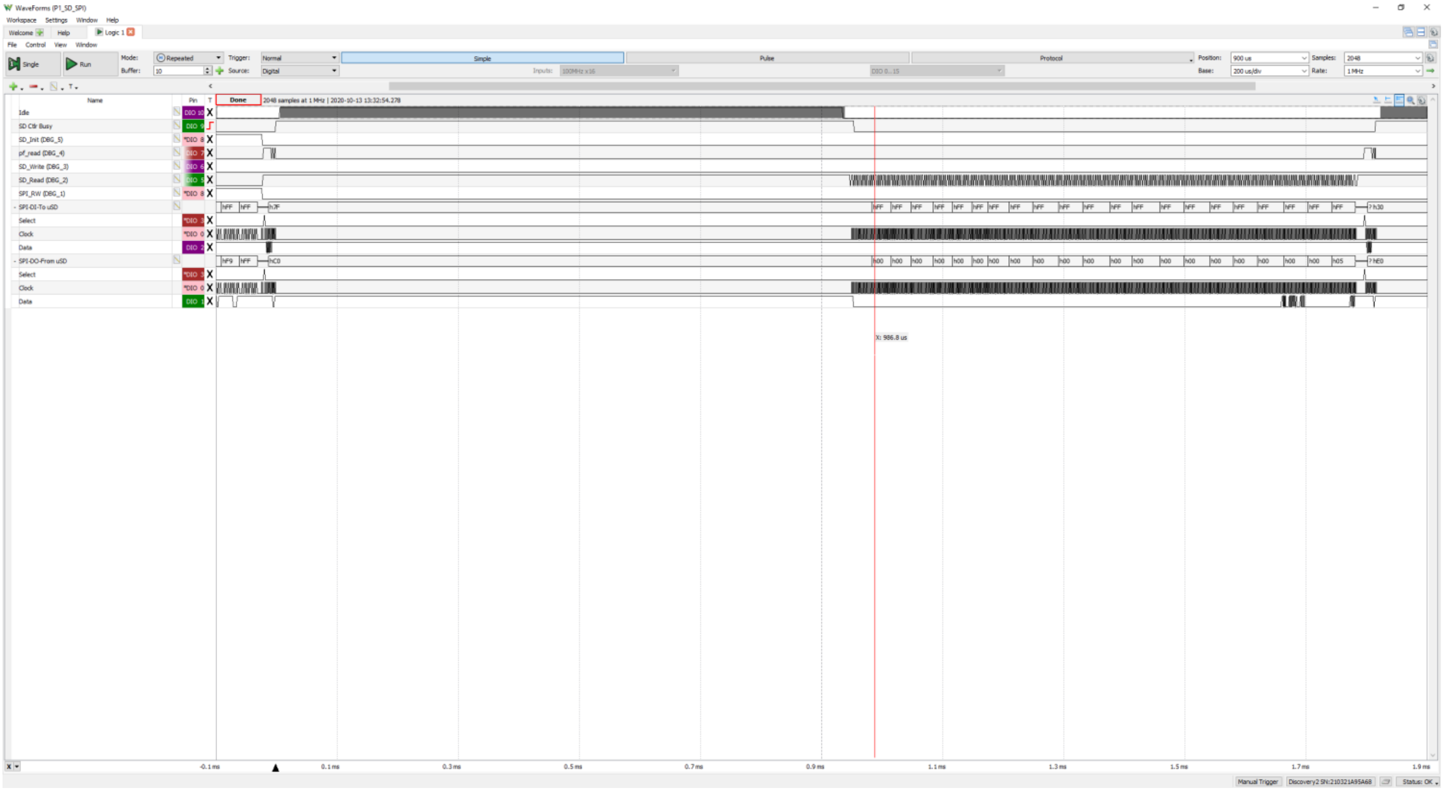
Blocks read	1149
Total time	11364
Idle loops	0
Idle time	0

Blocks read indicates the number of blocks of data which were read during the program's execution. Total time indicates the number of milliseconds the program took to run. Idle loops indicated how many times the idle thread ran. Idle time indicates the number of milliseconds the idle thread ran for. The total time relates to the length of SD Ctlr Busy because SD Ctlr Busy shows how much time the program waits for a block of data from the SD card.

ANALYZE OSDELAY TIMING

- What range of delay times do you see, and how do they compare with the requested delay?

The delay times range anywhere from 0ms to 1ms. This compares to the requested delay in they the times range anywhere between the requested delay and the requested delay – 1.



WaveForms (P1_SD_SPI)

Mode: Repeated Buffer: 10 Trigger: Normal Source: Digital Inputs: 100MHz x16 DIO 0...15

Position: 900 us Base: 200 us/div Samples: 2048 Rate: 1 MHz

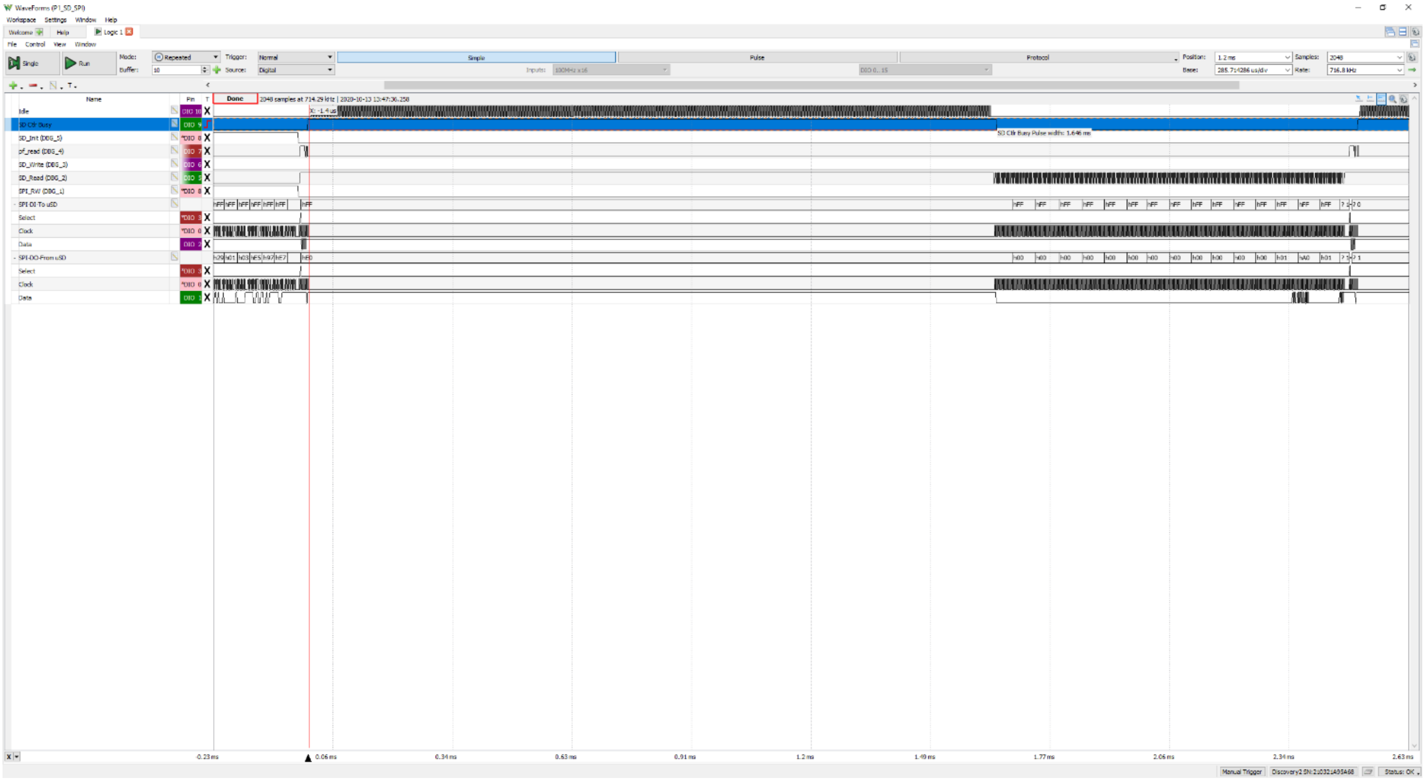
Done 2048 samples at 1 MHz | 2020-10-13 13:32:54.278

-

  - What are the values of the statistics reported on the LCD? What is the fraction of time the idle thread executes? What do they indicate has changed from the previous case?

Blocks read	1149
Total time	11732
Idle loops	3464210
Idle time	577

The idle thread executes for 4.92% of the time. These values indicate that the total time increased and the idle thread was able to execute during the execution of the program.


WaveForms (P1_SD_SPI)

Mode: Repeated
Buffer: 10
Trigger: Normal
Source: Digital
Position: 1.2 ms
Base: 285.714286 us/div
Samples: 2048
Rate: 716.8 kHz

Done 2048 samples at 714.29 kHz | 2020-10-13 13:47:36.258

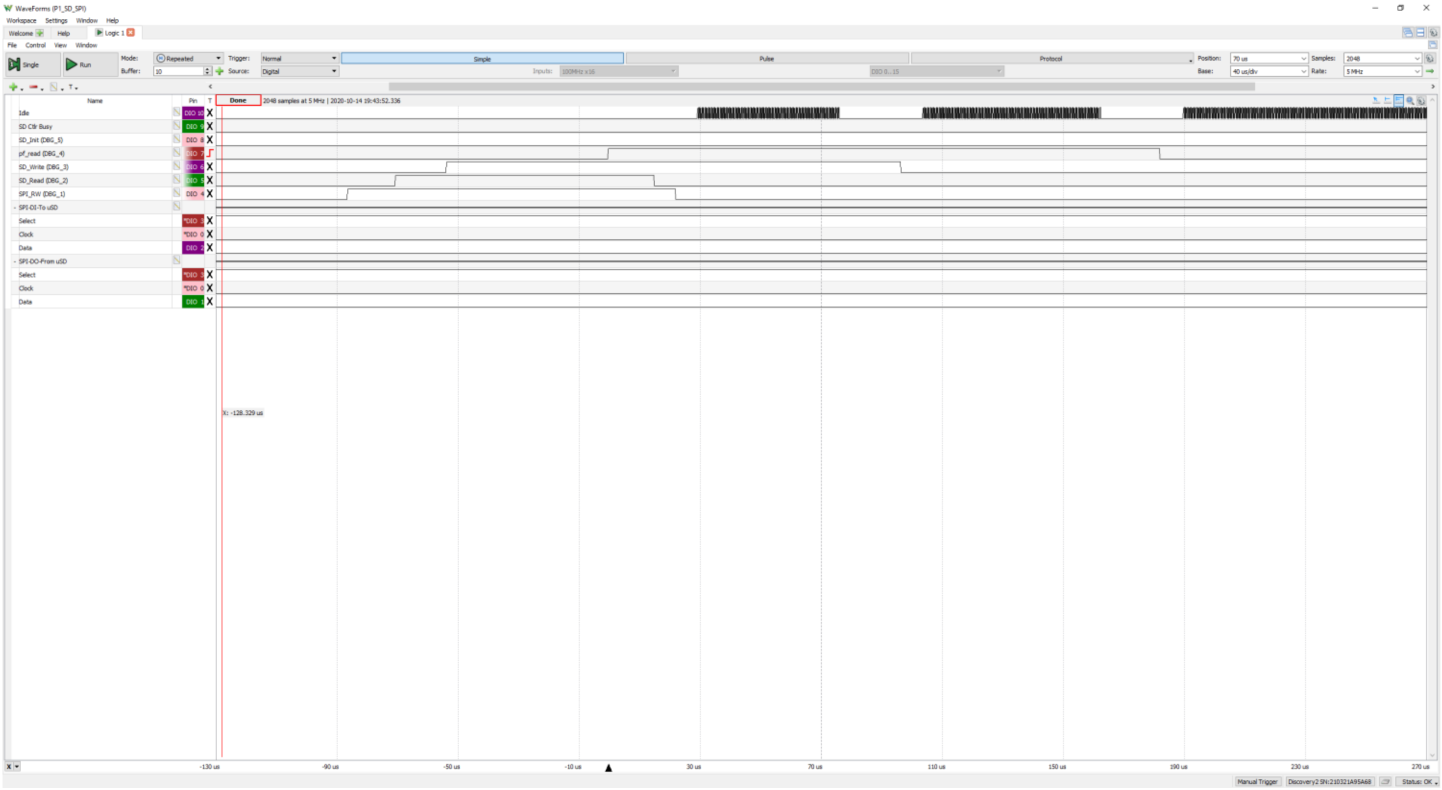
SD Ctlr Busy Pulse width: 1.646 ms

The delay times range anywhere from 1ms to 2ms. This compares to the requested delay in they the times range anywhere between the requested delay and the requested delay – 1.

Blocks read	1149
Total time	12845
Idle loops	10267424
Idle time	1711

The idle thread executes for 13.32% of the time. These values indicate that the total time increased and the idle thread was able to execute for a greater period of time during the execution of the program.

ECE 560: EXPLAIN PRECISION DELAY SOFTWARE DESIGN AND ANALYZE TIMING



Mode: Repeated
Buffer: 10
Trigger: Normal
Source: Digital
Inputs: DIO 0...15
Position: 70 us
Base: 40 us/div
Samples: 2048
Rate: 5 MHz

Done 2048 samples at 5 MHz | 2020-10-14 19:43:52.336

- Evaluate the timing accuracy of your solution.
  - How much time overhead does your precision delay mechanism add to the requested delay?

For T1, my precision delay mechanism added 80.4us. For T2, my precision delay mechanism added 75.6us. For T3, my precision delay mechanism added 64.2us. For T4, my precision delay mechanism added 3.2us.

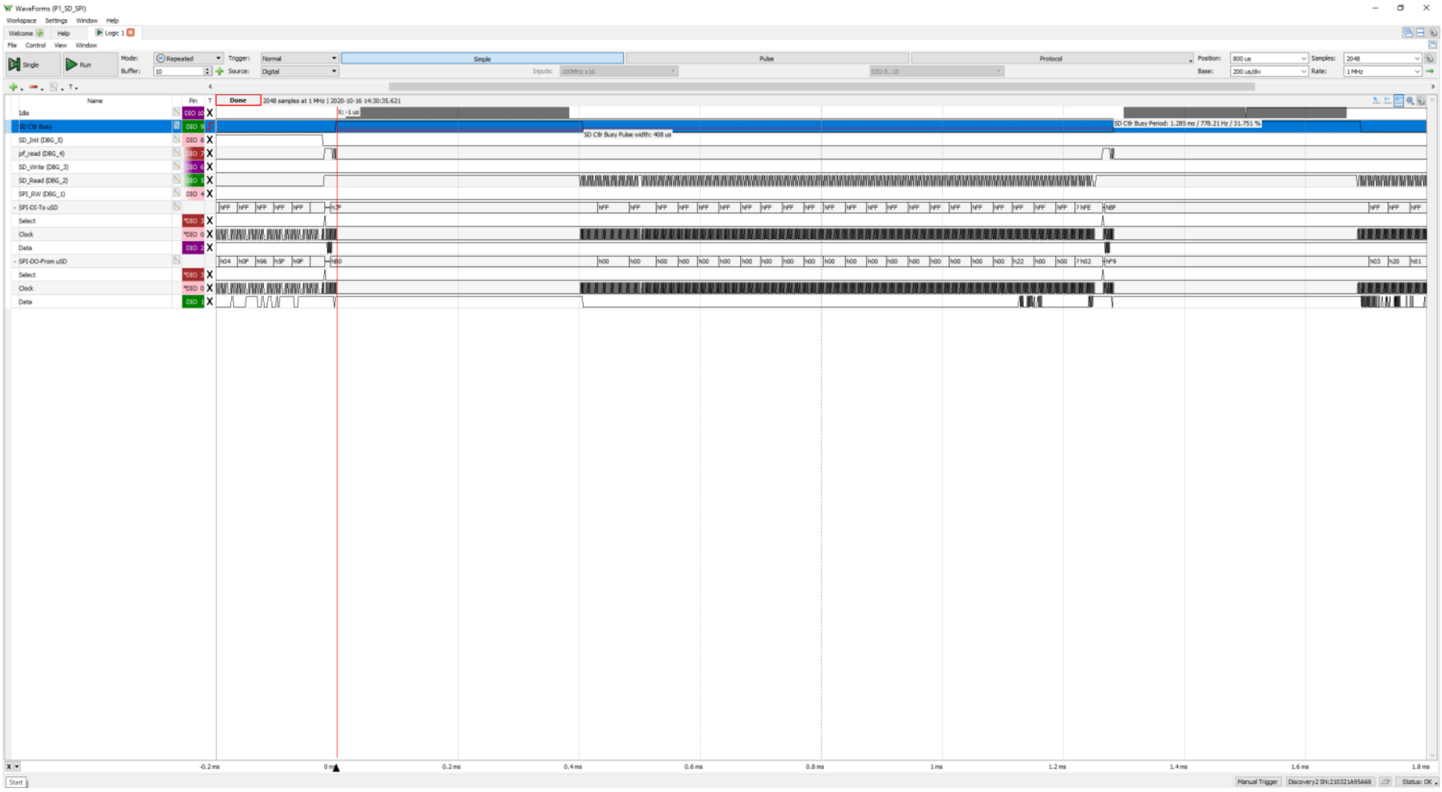
- Does the overhead vary with the number of times a precision delay is interrupted by another channel? If so, explain why.

The overhead does seem to vary with the number of times a precision delay is interrupted by another channel. I believe this is due to the fact that when the precision delay is called when another is virtual channel is being used, the other channels are updated after some calculations are made. These calculations could introduce overhead for every successive call of precision delay.

- How consistent are the time delays? Use your logic analyzer or oscilloscope to find the minimum and maximum timing errors.

The time delays are very consistent from run to run. For T1, the minimum timing error was 79.6us and the maximum timing error was 82.0us. For T2 the minimum timing error was 73.9us and the maximum timing error was 76.2us. For T3, the minimum timing error was 64.0us and the maximum timing error was 66.5us. For T4, the minimum timing error was 3.1us and the maximum timing error was 4.0us.

USE PRECISION DELAY FOR SD READ COMMAND



WaveForms (P1_SD_SPI)

Mode: Repeated
Buffer: 10
Trigger: Normal
Source: Digital
Inputs: 100MHz x16
Position: 800 us
Base: 200 us/div
Samples: 2048
Rate: 1 MHz

Done 2048 samples at 1 MHz | 2020-10-16 14:30:35.621

SD Ctlr Busy Pulse width: 408 us

SD Ctlr Busy Period: 1.285 ms / 778.21 Hz / 31.751 %

Manual Trigger | Discovery2 SN:210321A95A68 | Status: OK

- What are the values of the statistics reported on the LCD? What is the fraction of time the idle thread executes? What do the statistics indicate has changed from the previous case?

Blocks read	1149
Total time	11584
Idle loops	2512484
Idle time	418

The idle thread executes for 3.61% of the time. The statistics indicate that the sd_read thread is able to more precisely use the idle thread to take advantage of time where it is waiting for data from the SD card.

RETROSPECTIVE

- Lessons learned in this project, and how you might do things differently next time
  - Technical issues (processor, peripherals, compiler, tools, assembly code, etc.)

    The SD card could not be formatted easily due to its size. The project code became too big after implementing my precision delay code so I needed a new version of uVision.
  - Changes to your own development process

    Next time I plan to better utilize careful algorithm planning before coding and to make better use of the debugging tools such as the watch window.