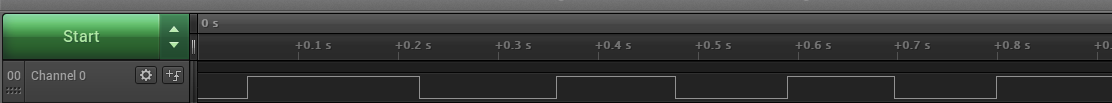
Lab Results:

Submit the following on the project assignment on I-learn.

1. Create a new project in the Code Composer IDE, an empty project with main.c. Add the code for the simple switch loop with the LED. Build the code, and then download the code. Use the logic analyzer on the LED output signal, and submit a screen shot of the Logic Analyzer capturing a transition based on a button press.





2) The gpio\_input\_interrupt\_MSP\_EXP432P401R\_nortos\_ccs example project

/\* DriverLib Includes \*/

#include<C:\Users\drtch\OneDrive\Documents\Kyle\School\ECEN361\msp432\_driverlib\_3\_21\_00\_05\driverlib\MSP432P4xx\driverlib.h>

/\* Standard Includes \*/

**#include** <stdint.h>

**#include** <stdbool.h>

**int main**(**void**)

{

MAP\_WDT\_A\_holdTimer();

MAP\_GPIO\_setAsOutputPin(GPIO\_PORT\_P1, GPIO\_PIN0);

MAP\_GPIO\_setAsInputPinWithPullUpResistor(GPIO\_PORT\_P1, GPIO\_PIN1);

MAP\_GPIO\_clearInterruptFlag(GPIO\_PORT\_P1, GPIO\_PIN1);

MAP\_GPIO\_enableInterrupt(GPIO\_PORT\_P1, GPIO\_PIN1);

MAP\_Interrupt\_enableInterrupt(INT\_PORT1);

MAP\_SysCtl\_enableSRAMBankRetention(SYSCTL\_SRAM\_BANK1);

MAP\_Interrupt\_enableMaster();

**while** (1) { MAP\_PCM\_gotoLPM3(); }}

**void PORT1\_IRQHandler**(**void**)

{

uint32\_t status;

status = MAP\_GPIO\_getEnabledInterruptStatus(GPIO\_PORT\_P1);

MAP\_GPIO\_clearInterruptFlag(GPIO\_PORT\_P1, status);

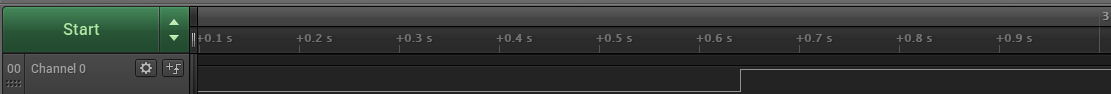
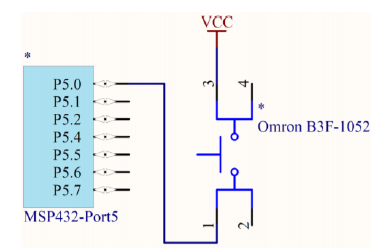
**if**(status & GPIO\_PIN1) { MAP\_GPIO\_toggleOutputOnPin(GPIO\_PORT\_P1, GPIO\_PIN0); }}

Compile and Run the Code. Use the Logic Analyzer to capture a transition based on a button press.

Code doesn’t load on to the board.

3) Use your parts to create your own switch and LED interface. Use these to create an external version of the internal version you created in step 1) above. Capture a Logic Analyzer trace of the functionality.

Here is how to connect the switch:



And here is how to connect the LED:

