ENCM 515 – Lab 1

Group #	16	Date: Jan 30
Student name(s)	Dominic Choi	
	Nathan Ante	
Instructor/TA	Dr. Benjamin Tan	

To submit:

- **Add your lab sheet (as PDF if possible) and your main.c to a zip archive and upload to D2L.**

PART ONE

LEDs

LD3 – colour: ORANGE – I/O Port: PD13

LD4 – colour: GREEN – I/O Port: PD12

LD5 – colour: RED – I/O Port: PD14

LD6 – colour: BLUE – I/O Port: PD15

Push Button

B1 – I/O Port: PA0

B2 – RESET, I/O Port: NRST

Q1: What is the base address of the peripheral space?

0x4000_0000UL

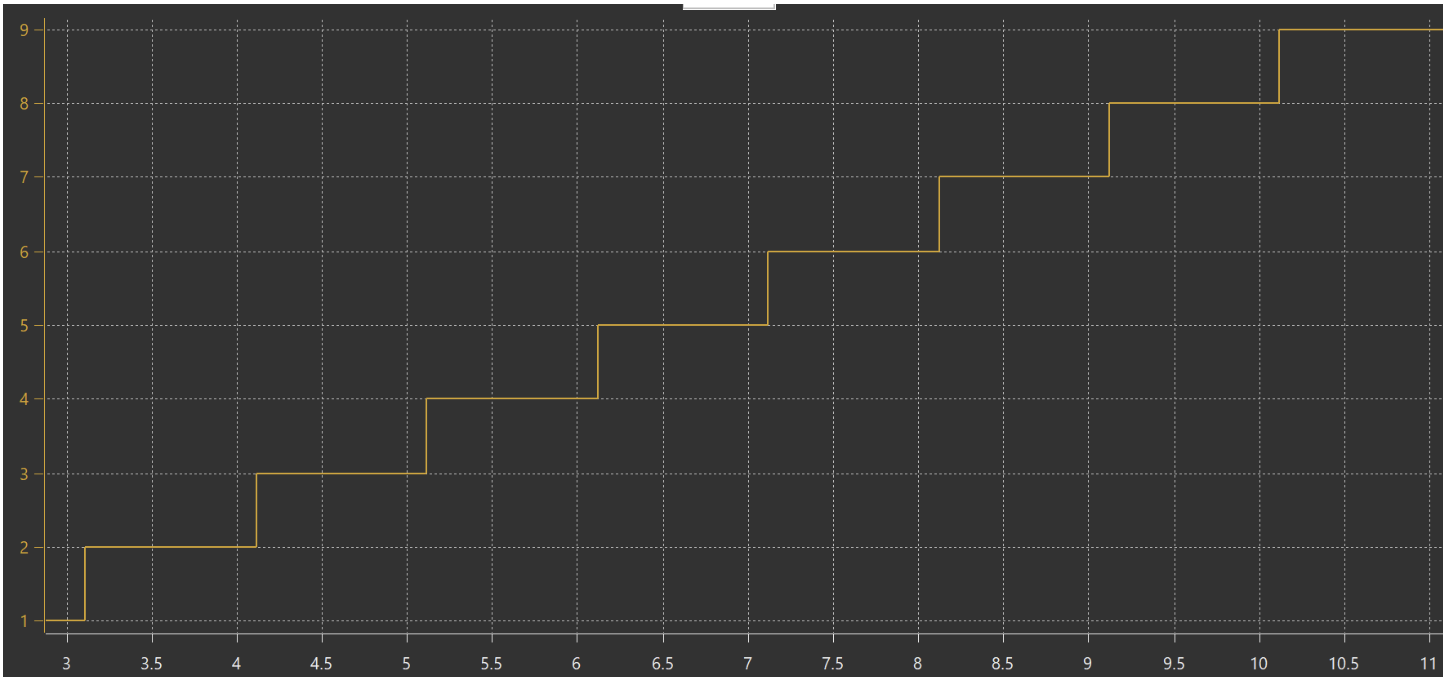
Q2: What are the functions for each of the buttons A – H?

```
A: Terminate and Relaunch
B: Resume
C: Suspend
D: Terminate
E: Disconnect
F: Step Into
G: Step Over
H: Step Return
```

Q3: How much time is elapsed between each WRITE? Is this expected?

```
1.001214 seconds. Yes, this is expected, because we put a delay of 1000 ms, in addition
to the delay due to execute the rest of the code
```

Q4: Take a screenshot of the SWV Data Trace Timeline Graph. You might need to wait some time to get an interesting shape to appear. Alternatively, adjust your delay so that the Loop value is incremented more often.

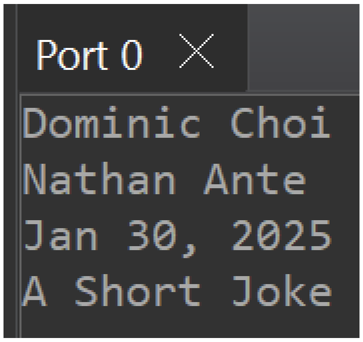


Q5: Copy the listing for the `test_assembly` function and annotate it. Mark which lines of assembly correspond to what you've written and what corresponds to the compiler-inserted code for the Calling Convention. Explain what's going on (e.g., here we can see the registers that will be overwritten are being saved on the stack…. Now we see them restored…)

```
int test_assembly(int a, int b) {
      int res = 0;
      __asm volatile ("ADD %[result], %[input_i], %[input_b]"    // ADD dest, arg1, arg2
      : [result] "=r" (res)                                      // destination res
      : [input_i] "r" (a), [input_b] "r" (b)                     // add values stored in a and b
      );
 return res;
}
```

PART TWO

Q1> Screenshot of the console output (name, date, joke)



Q2> Time stamps and cycles for the two ITM Port 31 entries (screenshot should suffice).

How many clock cycles are there between the two writes to the ITM Port 31?

`24000024 clock cycles`

How much time has elapsed?

`250.00025ms`

What is the average amount of time taken for each loop iteration?

`250.00025ms / 2000000 iterations = 0.000125ms = 125 ns`

Q3 > > How many clock cycles are taken to perform the floating point function?

Test	Value of A	Value of B	Clock Cycles needed
1	0.5	0.125	43
2	0.51	0.1251	45
3	0.321321	0.123123	45
4	0.43567	0.89261	45
5	5.423178	7.820365	45

Q4 > How many clock cycles are taken to perform the floating point function now? Include a screenshot of the SWV Trace Log. 0.

It takes 45 clock cycles for “non-clean”

while “clean” floating point numbers took 43 clock cycles

Index	Type	Data	Cycles	Time(s)	Extra info
0	Sync	56	54987 ?	?	No timestamp received for p...
1	Overflow		54987 ?	?	No timestamp received for p...
2	ITM Port 31	1	54987	572.781250 µs	
3	Sync	48	55032 ?	?	No timestamp received for p...
4	Sync	48	55032 ?	?	No timestamp received for p...
5	ITM Port 31	2	55032	573.250000 µs	
6	ITM Port 31	1	55038	573.312500 µs	

Q5 > Fix the code snippet so that the correct values are loaded in a and b. Paste your completed function here.

```
void FixedExperiment(void){
	int16_t a = 16384; // 0.5 Q.15 in int16_t representation
	int16_t b = 4096;	// 0.125 Q.15 in int16_t representation
	int16_t c = a * b;
}
```

This calculation took only 39 clock cycles and the variable c had a value of 0.

Q6> Take a screenshot of the Variables window after calling these functions. What do you observe?

Name	Type	Value
a	float [10]	0x2001ffc0
a[0]	float	-0.109999999
a[1]	float	-0.119999997
a[2]	float	-0.129999995
a[3]	float	-0.140000001
a[4]	float	0.150000006
a[5]	float	0.159999996
a[6]	float	0.170000002
a[7]	float	0.180000007
a[8]	float	0.189999998
a[9]	float	1.20000005
b	q15_t [10]	0x2001ffac
b[0]	q15_t	-3604
b[1]	q15_t	-3932
b[2]	q15_t	-4259
b[3]	q15_t	-4587
b[4]	q15_t	4915
b[5]	q15_t	5242
b[6]	q15_t	5570
b[7]	q15_t	5898
b[8]	q15_t	6225
b[9]	q15_t	32767
c	float32_t [10]	0x2001ff84
c[0]	float32_t	-0.109985352
c[1]	float32_t	-0.119995117
c[2]	float32_t	-0.129974365
c[3]	float32_t	-0.139984131
c[4]	float32_t	0.149993896
c[5]	float32_t	0.159973145
c[6]	float32_t	0.16998291
c[7]	float32_t	0.179992676
c[8]	float32_t	0.189971924
c[9]	float32_t	0.999969482

We observe that the original float is not exactly as defined.
When converting back to float32_t from q15, most numbers differed from the original around the 5th decimal place.

Q7> Write a reflection of what you have observed and learned in this lab [approx.. 200 words max.]. Which lecture concepts did you encounter here?

In this lab, we have learned how to use STM32's development environment with STM32CubeIDE, and some of its useful features. We have played around with STM32s in the past so we are slightly accustomed to it, but one useful IDE feature that we learned to use was the SWV Data Tracer. With that feature, we were able to accurately monitor the system's behavior such as the time elapsed as well as the number of clock cycles between lines of code. With SWV data tracer, we were also able to see how long it took the ARM Cortex-M4 based STM32F4 microcontroller to perform multiplications of differing variable types. We also were introduced to using the CMSIS DSP software library which offered floating-point variable types and some functions to manipulate them. We have learned quite a bit from this lab. Some lecture concepts we encountered were discrete time signals as well as number systems like floating, fixed, and binary point numbers.