Unit_Test.c program structure

Ismail Yesildirek, Bijan Kianian

In this document, the program flow chart is discussed indicating the functions that are taking effect as the program runs.

Folder arrangement (*Git master branch*):

ECEN5813_SP19_Project2/

Makefile	*READ.ME*	
Unittest/	inc/	src/
Unit_Test.c	ring.h	Project_2
	Uart.h	ring.c

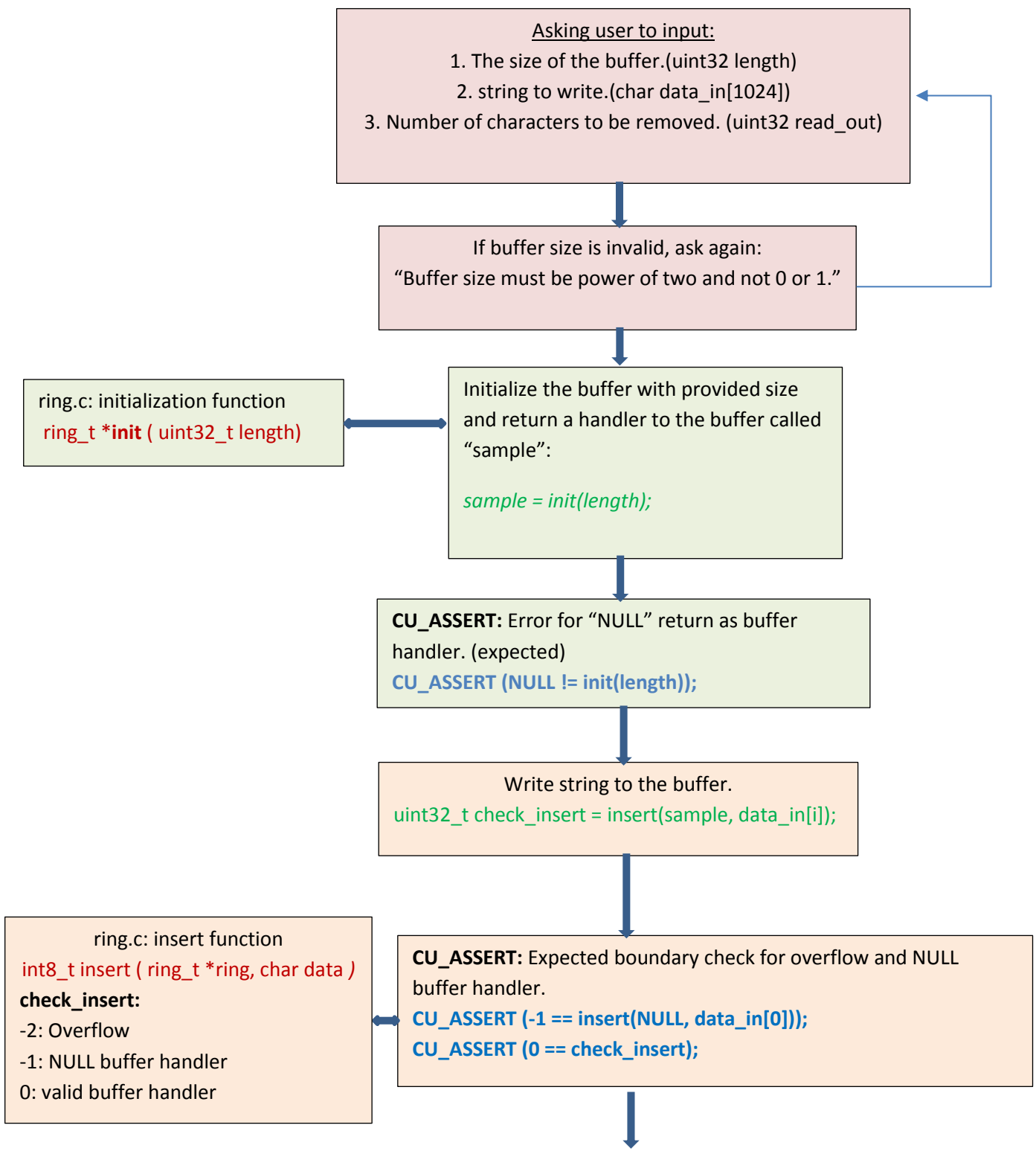
Unit_Test.c

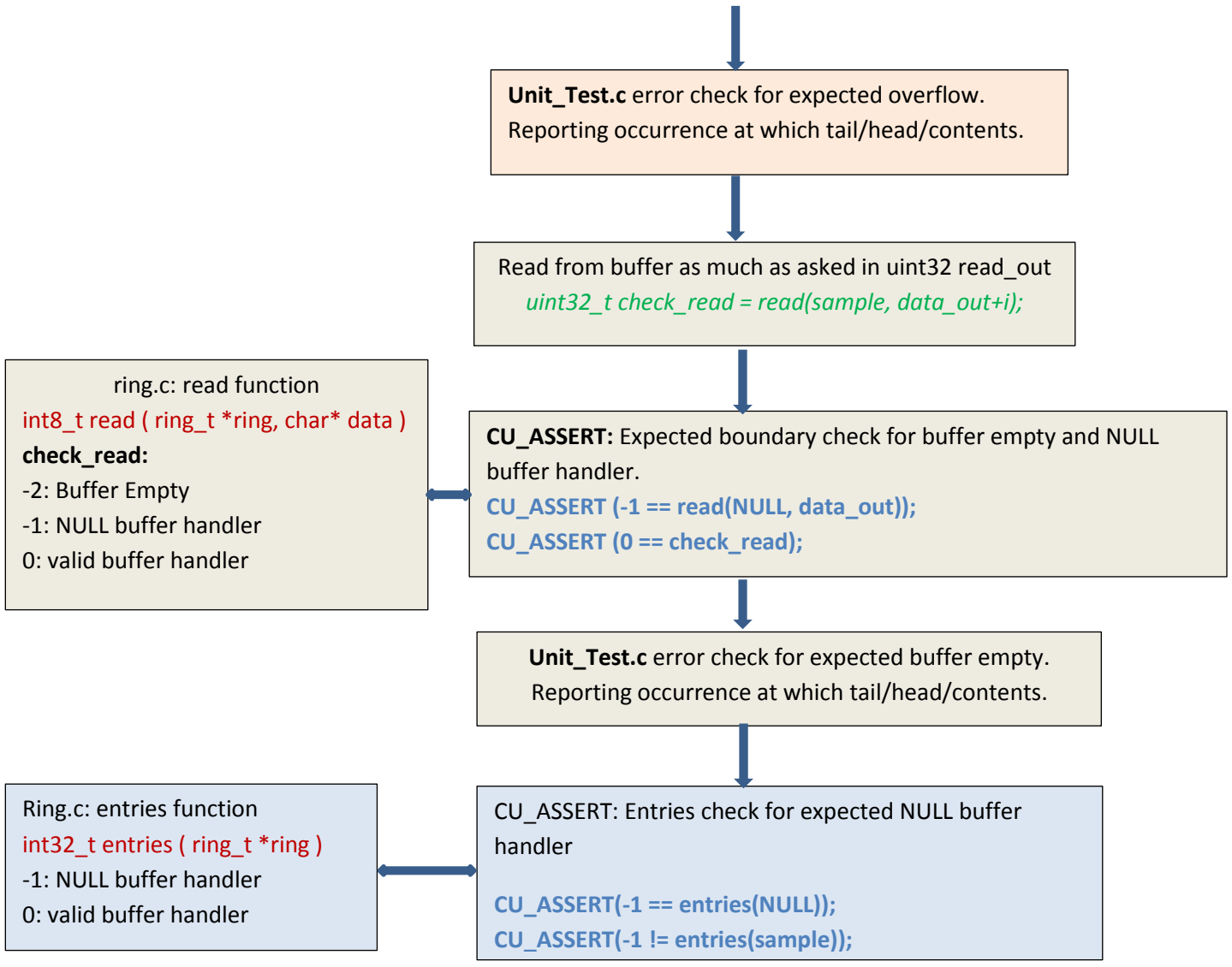
The file is using ring.c and ring.h in Linux environment and in a user interactive fashion, performs multiple tasks and applications in manual and auto mode selected by a compiler switch at the beginning (#define AUTO 0/1) and in the end provides the CUnit test results table.

Executing the program:

To execute the program the make command would be: **make Cunit**

Manual mode (*AUTO = 1*)
CUnit test is implemented with one suite called pSuit1. It includes the following test modules:
1. **int init_suite(void):** Initialization module.
2. **int clean_suite(void):** This module will release all memory allocation once the test is completed at closing.
3. **void test_init(void):** in this module, the initialization function of ring.c is exersized. If Buffer address is not assigned (NULL), an error will be asserted.
4. **void test_insert(void):** This module will test the insert function in ring.c *[int8_t insert (ring_t *ring, char data)].* If the buffer handker is NULL or Buffer_Full event happens, an error will be asserted.
5. **void test_read(void):** In this module, the read function of ring.c *[int8_t read (ring_t *ring, char* data)]* is tested. If there is a NULL buffer handler of Buffer Empty happens, an error will be asserted.
6. **void test_entries(void):** This module will examine the entries function of ring.c *[int32_t entries (ring_t *ring)]* an error will be asserted if buffer handler is NULL.





Applications:

1. Data validation: It compares input / output strings in a new buffer.
2. Fibonacci sequence: Produces Fibonacci sequence based on user input for third buffer.
3. ASCII table: prints the ascii table using another buffer.

All boundary checks and error warnings are valid in applications, as well.

Long term Autotest *(AUTO 1):*

In the Long-term automatic mode (AUTO 1), the number of test cycles is arbitrarily predefined (here 5). There is no user interaction, instead, all buffer parameters, such as size (between 2 and 1024) string characters (0 to 255 possible ASCII) and string size (between 0 to maximum length 1024) are randomly selected. This method provides the ability to incorporate different buffer sizes and contents each time the new cycle of test begins:

```
seed++;
uint16_t strSize = randomValue(seed, MAX_LENGTH);        // Random # between 0 ~ 1024 for string length

for( uint16_t l = 1; l <= strSize ; l++)                 // data_in holds the generated string with random length
    data_in[l] = randomValue(data_in[l-1], 256);
```

The random number selection also applies to all three extra functions. In Fibonacci, the two initial numbers that were provided by user in manual mod, in auto mode will be defined randomly as well:

```
seed++;
size2 = randomValue (seed, 10);
int32_t sequence[size2];                        // integer array to store calculated series.

sequence[0] = randomValue (seed, MAX_LENGTH);        // Random # between 0 ~ MAX_LENGTH for Fibonacci series
seed++;
sequence[1] = randomValue (seed, MAX_LENGTH);
```