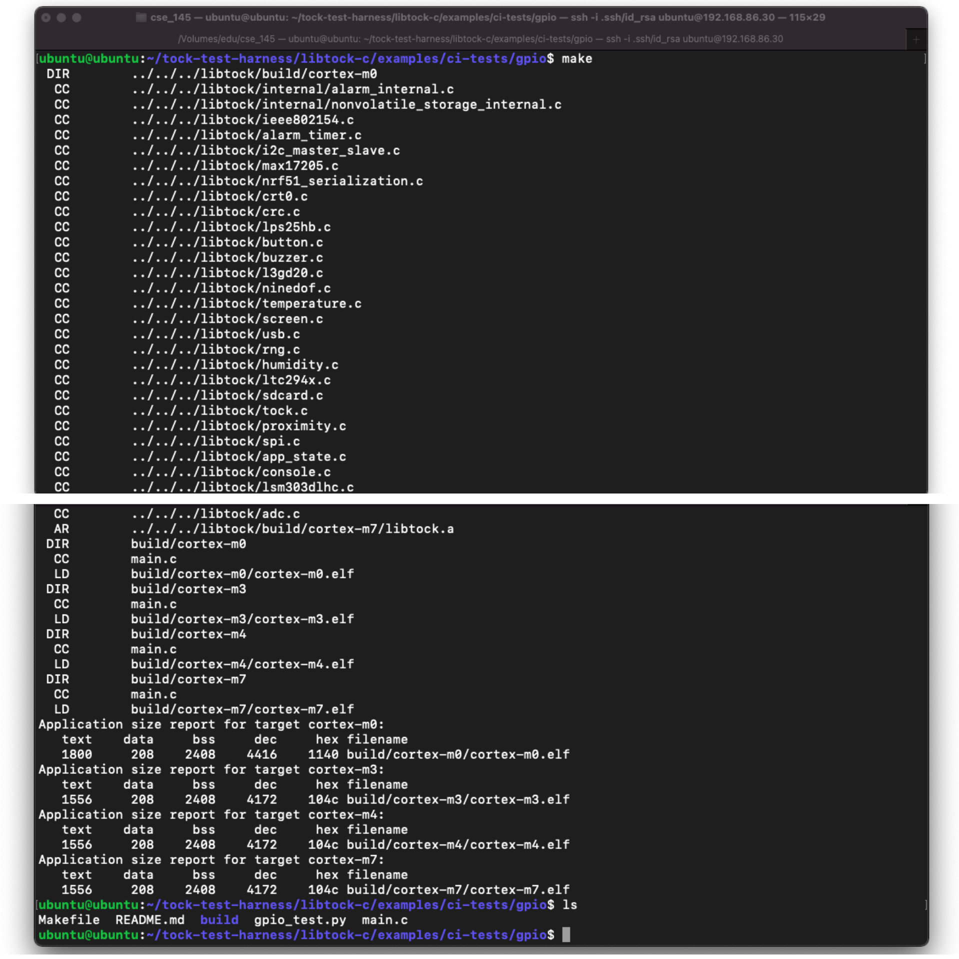
Milestone Report

Tock Hardware CI

Jefferson Chien, Arvin Lin, Jack Sheridan

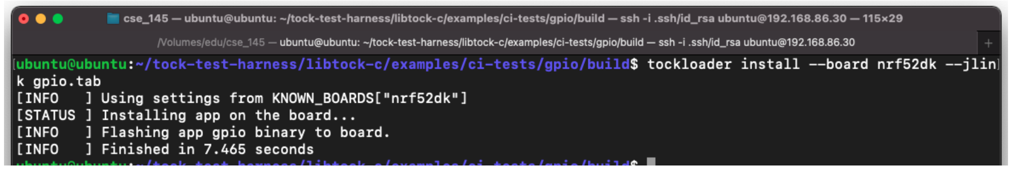
Progress

1. Installation of Tock OS on nRF52840 dev kit



```
cse_145 — ubuntu@ubuntu: ~/tock-test-harness/libtock-c/examples/ci-tests/gpio — ssh -i .ssh/id_rsa ubuntu@192.168.86.30 — 115×29
/Volumes/edu/cse_145 — ubuntu@ubuntu: ~/tock-test-harness/libtock-c/examples/ci-tests/gpio — ssh -i .ssh/id_rsa ubuntu@192.168.86.30
ubuntu@ubuntu:~/tock-test-harness/libtock-c/examples/ci-tests/gpio$ make
 DIR        ../../../libtock/build/cortex-m0
  CC        ../../../libtock/internal/alarm_internal.c
  CC        ../../../libtock/internal/nonvolatile_storage_internal.c
  CC        ../../../libtock/ieee802154.c
  CC        ../../../libtock/alarm_timer.c
  CC        ../../../libtock/i2c_master_slave.c
  CC        ../../../libtock/max17205.c
  CC        ../../../libtock/nrf51_serialization.c
  CC        ../../../libtock/crt0.c
  CC        ../../../libtock/crc.c
  CC        ../../../libtock/lps25hb.c
  CC        ../../../libtock/button.c
  CC        ../../../libtock/buzzer.c
  CC        ../../../libtock/l3gd20.c
  CC        ../../../libtock/ninedof.c
  CC        ../../../libtock/temperature.c
  CC        ../../../libtock/screen.c
  CC        ../../../libtock/usb.c
  CC        ../../../libtock/rng.c
  CC        ../../../libtock/humidity.c
  CC        ../../../libtock/ltc294x.c
  CC        ../../../libtock/sdcard.c
  CC        ../../../libtock/tock.c
  CC        ../../../libtock/proximity.c
  CC        ../../../libtock/spi.c
  CC        ../../../libtock/app_state.c
  CC        ../../../libtock/console.c
  CC        ../../../libtock/lsm303dlhc.c
  CC        ../../../libtock/adc.c
  AR        ../../../libtock/build/cortex-m7/libtock.a
 DIR        build/cortex-m0
  CC        main.c
  LD        build/cortex-m0/cortex-m0.elf
 DIR        build/cortex-m3
  CC        main.c
  LD        build/cortex-m3/cortex-m3.elf
 DIR        build/cortex-m4
  CC        main.c
  LD        build/cortex-m4/cortex-m4.elf
 DIR        build/cortex-m7
  CC        main.c
  LD        build/cortex-m7/cortex-m7.elf
Application size report for target cortex-m0:
   text    data     bss     dec     hex filename
   1800     208    2408    4416    1140 build/cortex-m0/cortex-m0.elf
Application size report for target cortex-m3:
   text    data     bss     dec     hex filename
   1556     208    2408    4172    104c build/cortex-m3/cortex-m3.elf
Application size report for target cortex-m4:
   text    data     bss     dec     hex filename
   1556     208    2408    4172    104c build/cortex-m4/cortex-m4.elf
Application size report for target cortex-m7:
   text    data     bss     dec     hex filename
   1556     208    2408    4172    104c build/cortex-m7/cortex-m7.elf
ubuntu@ubuntu:~/tock-test-harness/libtock-c/examples/ci-tests/gpio$ ls
Makefile  README.md  build  gpio_test.py  main.c
ubuntu@ubuntu:~/tock-test-harness/libtock-c/examples/ci-tests/gpio$
```

2. Compilation and installation of sample Tock OS testing App on nRF52840 dev kit

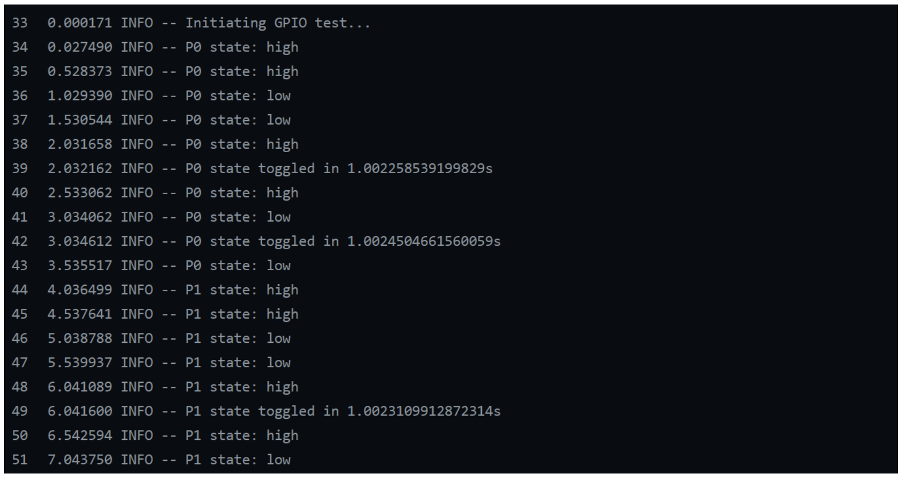


```
cse_145 — ubuntu@ubuntu: ~/tock-test-harness/libtock-c/examples/ci-tests/gpio/build — ssh -i .ssh/id_rsa ubuntu@192.168.86.30 — 115×29
/Volumes/edu/cse_145 — ubuntu@ubuntu: ~/tock-test-harness/libtock-c/examples/ci-tests/gpio/build — ssh -i .ssh/id_rsa ubuntu@192.168.86.30
ubuntu@ubuntu:~/tock-test-harness/libtock-c/examples/ci-tests/gpio/build$ tockloader install --board nrf52dk --jlin
k gpio.tab
[INFO   ] Using settings from KNOWN_BOARDS["nrf52dk"]
[STATUS ] Installing app on the board...
[INFO   ] Flashing app gpio binary to board.
[INFO   ] Finished in 7.465 seconds
ubuntu@ubuntu:~/tock-test-harness/libtock-c/examples/ci-tests/gpio/build$
```

3. Ubuntu server setup on Raspberry Pi
    a. See all pictures of ssh into the pi.

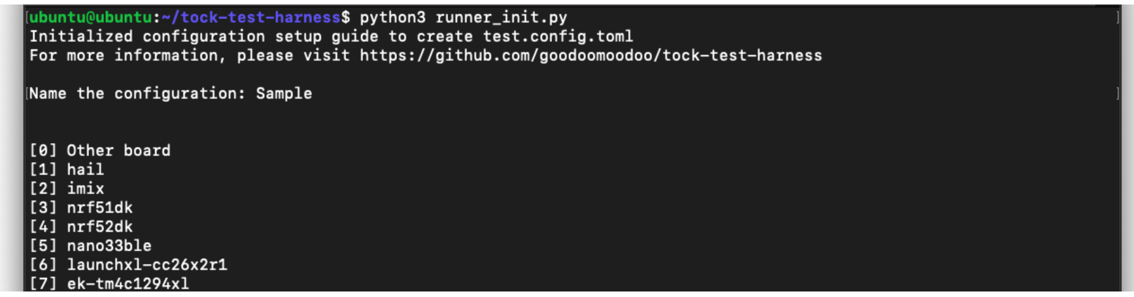
4. Test Github runner on Raspberry Pi and nRF52840 dev kit
    a. [Link to demo]

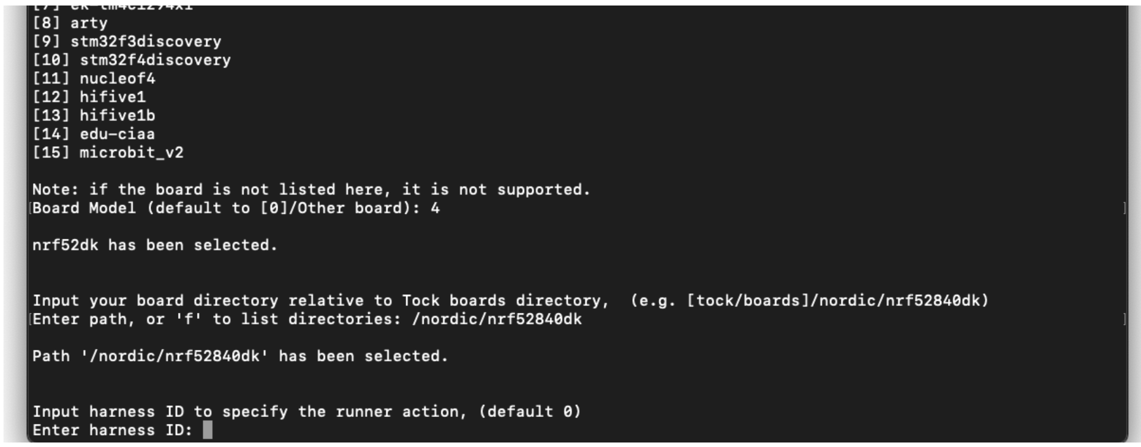
5. Setup GPIO testing software



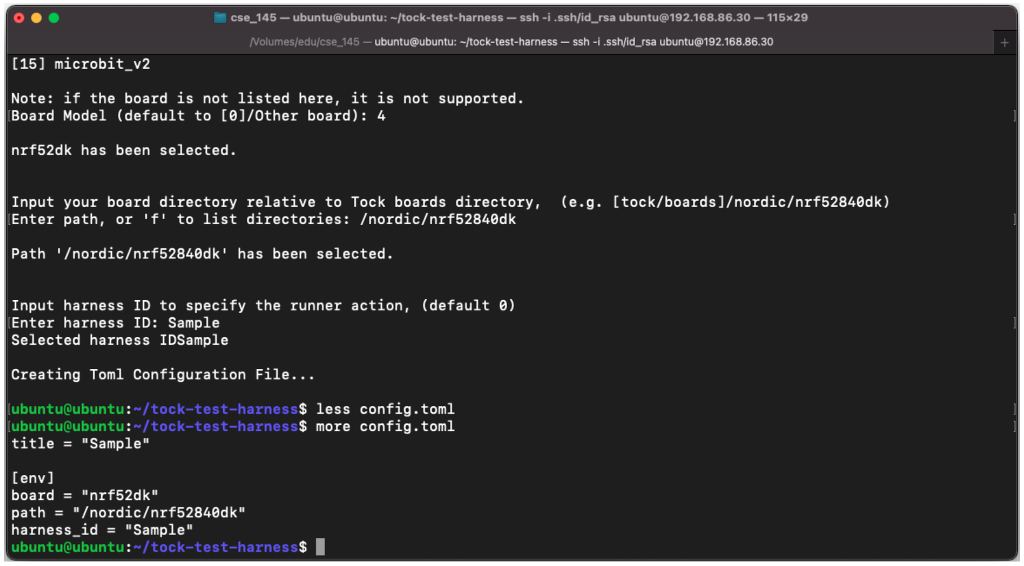
```
0.000171 INFO -- Initiating GPIO test...
0.027490 INFO -- P0 state: high
0.528373 INFO -- P0 state: high
1.029390 INFO -- P0 state: low
1.530544 INFO -- P0 state: low
2.031658 INFO -- P0 state: high
2.032162 INFO -- P0 state toggled in 1.002258539199829s
2.533062 INFO -- P0 state: high
3.034062 INFO -- P0 state: low
3.034612 INFO -- P0 state toggled in 1.0024504661560059s
3.535517 INFO -- P0 state: low
4.036499 INFO -- P1 state: high
4.537641 INFO -- P1 state: high
5.038788 INFO -- P1 state: low
5.539937 INFO -- P1 state: low
6.041089 INFO -- P1 state: high
6.041600 INFO -- P1 state toggled in 1.0023109912872314s
6.542594 INFO -- P1 state: high
7.043750 INFO -- P1 state: low
```

6. Setup mechanism to run testing Apps on the embedded device
    a. Generate configuration files for test specification





```
ubuntu@ubuntu:~/tock-test-harness$ python3 runner_init.py
Initialized configuration setup guide to create test.config.toml
For more information, please visit https://github.com/goodoomoodoo/tock-test-harness

Name the configuration: Sample


[0] Other board
[1] hail
[2] imix
[3] nrf51dk
[4] nrf52dk
[5] nano33ble
[6] launchxl-cc26x2r1
[7] ek-tm4c1294xl
[8] arty
[9] stm32f3discovery
[10] stm32f4discovery
[11] nucleof4
[12] hifive1
[13] hifive1b
[14] edu-ciaa
[15] microbit_v2

Note: if the board is not listed here, it is not supported.
Board Model (default to [0]/Other board): 4

nrf52dk has been selected.


Input your board directory relative to Tock boards directory,  (e.g. [tock/boards]/nordic/nrf52840dk)
Enter path, or 'f' to list directories: /nordic/nrf52840dk

Path '/nordic/nrf52840dk' has been selected.


Input harness ID to specify the runner action, (default 0)
Enter harness ID: 
```



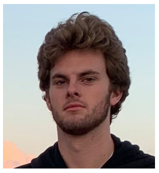
```
cse_145 — ubuntu@ubuntu: ~/tock-test-harness — ssh -i .ssh/id_rsa ubuntu@192.168.86.30 — 115×29
/Volumes/edu/cse_145 — ubuntu@ubuntu: ~/tock-test-harness — ssh -i .ssh/id_rsa ubuntu@192.168.86.30
[15] microbit_v2

Note: if the board is not listed here, it is not supported.
Board Model (default to [0]/Other board): 4

nrf52dk has been selected.


Input your board directory relative to Tock boards directory,  (e.g. [tock/boards]/nordic/nrf52840dk)
Enter path, or 'f' to list directories: /nordic/nrf52840dk

Path '/nordic/nrf52840dk' has been selected.


Input harness ID to specify the runner action, (default 0)
Enter harness ID: Sample
Selected harness IDSample

Creating Toml Configuration File...

ubuntu@ubuntu:~/tock-test-harness$ less config.toml
ubuntu@ubuntu:~/tock-test-harness$ more config.toml
title = "Sample"

[env]
board = "nrf52dk"
path = "/nordic/nrf52840dk"
harness_id = "Sample"
ubuntu@ubuntu:~/tock-test-harness$ 
```

   b. Create and design workflow to integrate config files into custom tests

7. ~~Record behaviors to generate logs which is then send back to Raspberry Pi~~
8. ~~Compile all logs, including compilation results and messages on GPIO, into a single compressive report on Raspberry Pi.~~
9. ~~Automatically upload CI results onto the Github repository under the CI event in a reasonable format.~~
10. Construct new tests or adapt more existing Tock OS test
    a. Custom UART test
    b. Custom BLE test
11. (Optional) Add colors to log and add different level to log: info, warning, error
12. (Optional) Setup embedded system boot controller
13. Design documentation and Installation guide

Arvin Lin
Completed: 1, 2, 3, 4, 5
Future: 6b, 7

Jefferson Chien
Completed: 1, 2, 3, 4, 6a
Future: 10a, Modularize 6a wizard



Jack Sheridan
Completed: 1, 2, 3, 4
Future: 10b, 13

Deliverable 2 / MVP

MVP Demo

The logs and the test result shown in the GitHub Actions are returned from Raspberry Pi. The error pertaining to the log during the demo is a minor connection error between GitHub Actions and the Raspberry Pi, and it has no impact on the actual test workflow. The log will show up again if you refresh the page.

Updates / Changes

Some changes had to be made to our milestones now that we have researched Github's continuous integration as well as designed our system less for “hard-coding” and more for dynamic semi-scalability.

Milestone 7: Since the system monitor cannot be done inside the target board, the Raspberry Pi shall monitor the status of the target board externally. The target board should not send any logs to Raspberry Pi, instead, the Raspberry Pi shall monitor the target board and generate the logs.

Milestone 8 and 9: It turns out that the Github Action runner does not need to be passed a “report” or file at all, but instead passes all data written to standard out to Github. Due to efficiency and to keep specificity of each test harness, we decided not to specify each and every test in the top level yaml file that configures all test harnesses, but instead in a test configuration TOML file in each board folder. Therefore, these two milestones can be deleted.

Milestone 10: This milestone is not necessarily changed, but we would like to specify it a little more so that it is not as broad. We have decided to create a new directory called ci-test in the example folder of the libtock-c repository. This directory will have many unique directories which can be specified in each test harness’ TOML configuration file. Each of these folders will hold ‘test.py’ and ‘main.c’ files for the embedded app as well as it’s respective python test to run on the Pi.