Developing Soft and Parallel Programming Skills Using Project-Based Learning

Spring 2019

Team HuSTLE

Hwysun Park, Smit Patel, Tommy Lim,
Lucky Phan, Eseosasere Omobude

I. Planning and Scheduling

Work Breakdown Structure

Name	E-mail	Tasks	Duration	Note
Hwysun Park (Coordinator)	hpark44@student.gsu.edu	Setting up Slack and GitHub account; Writing and getting the report ready to submit	5 hours	Slack and GitHub should be created in the first place; Report due Fri, 2/8/19
Smit Patel	spatel261@student.gsu.edu	Putting together the answers to Teamwork Basics questions	3 hours	Teamwork Basics questions due Fri, 2/1/19
Tommy Lim	tlim3@student.gsu.edu	PI Installation and ARM Assembly Programming	5 hours	Set up the date for the Assembly Programming before Wed, 2/6/19
Lucky Phan	lphan7@student.gsu.edu	Creating a YouTube channel; Editing and uploading the presentation video	4 hours	Set up the date for the recording before Wed 2/6/19; Video due Fri, 2/8/19
Eseosasere Omobude	eomobude1@student.gsu.edu	Writing the report to describe the PI Installation and ARM Assembly Programming	2 hours	Set up the date for the Assembly Programming before Wed, 2/6/19

II. Teamwork Basics

1. What to do get the task accomplished and the team members' satisfaction high?
   - To get the tasks done and set the satisfaction of members high, we need to know each other's weaknesses and strengths. This will let us assign tasks easily and finish them before the deadline. Setting some ground rules will give us the best teamwork operation from each member. Using a facilitator is another way the team can become very productive about keeping progress, as he will drive the tasks to reach the goal. Another way is to use Slack app so that every person can talk about his progress. Also, it would be useful for communicating with each other whenever someone encounters a problem, making it possible to come up with a solution together and keep progressing towards the goal.

2. Work Norms: How will work be distributed? Who will set deadlines? What happens if someone doesn't follow through on his/her commitment (for example, misses a deadline)? How will the work be reviewed? What happens if people have different

opinions about the quality of the work? What happens if people have different work habits (e.g., some people like to get assignments done right away; others work better with the pressure of a deadline)?

- All of the assignments are distributed specifically upon the interest and the availability of time. They are given while everyone was on slack being asked about the tasks that he would like to do it. The understanding of our team was so remarkable that no one had difficulties with his tasks or complained about them. The group coordinator will set the deadlines for everyone. If people have different opinions about the quality of work, then they will collaborate to fix the problem. Different work habits are not a big problem unless people can get the assignments done on time. In this group, everyone had agreed upon not procrastinating this project as a last-minute assignment as it would be extremely stressful.

3. Facilitator Norms: Will you use a facilitator? How will the facilitator be chosen? Will you rotate the position? What are the responsibilities of the facilitator?
   - Yes, our group will have a facilitator. The group coordinator will be the facilitator. As the role of group coordinator is going to be rotated for each assignment, every member will have a chance of learning new things as a facilitator. The responsibility of the facilitator is (1) to encourage the members to give their hundred percent of the work to the assignment, (2) help them to figure out an alternative if any problems occur, and (3) make sure they finished the tasks on time.

4. Communication Norms: When should communication takes place and through what medium (e.g., do some people prefer to communicate through e-mail while others would rather talk on the phone)?
   - The communications are being done via Slack app, which is an excellent medium as everyone can see the messages and reply instantly.

5. Meeting Norms: What is everyone's schedule? Should one person be responsible for coordinating meetings? Do people have a preference for when meetings are held? Where is a good place to hold meetings? What happens if people are late to a meeting? What happens if a group member misses a meeting? What if he/she misses several meetings?
   - All the group members have different schedules as some people live on campus while others commute every day. Every person is responsible for deciding the meeting time that works for at least a majority of the group. Everyone has his preference about the meeting time and place. The best place for the meeting is the library study rooms. If any members are late or absent to a meeting, we are going

to post the things discussed during the meeting on the Slack app so that they would not get behind during the assignment.

6. Consideration Norms: Can people eat at meetings? What happens if someone is dominating the discussion? How can norms be changed if someone is not comfortable with what is going on in the team?
   - Most of the time, the meeting will take place at library study rooms, and people might bring some food based on the GSU library policy. As everyone is working in the groups, if one person dominates the discussion, others need to make sure that he listens to others' ideas unless he is making decisions as a coordinator. If someone is not comfortable with the norms, they can be changed by discussing with the team members and coming up with an agreement.

7. Handling Difficult Behavior: the person is too quiet.
   - If a person is not performing well or having difficulties communicating with others, asking him about his tasks, offering help he might need, and encouraging him to help others would make him more productive in our group. Another way is to start chatting with the person in Slack, as he may prefer online discussion. Asking questions to break the ice and get familiar can also make the person feel comfortable and be able to adjust to our group environment more easily.

8. Handling Difficult Behavior: the person argues.
   - If a person is rich in ideas and can give feedback to others' work, it can positively affect the performance and productivity of every member as he can be a game-changer for the whole group. However, everyone needs to be careful when criticizing others as it can hurt not only the group as a whole by dragging out the discussion but specifically the one being criticized by making him discouraged and stressful.

9. When making decisions, If the team is having trouble reaching consensus, what should you do?
   - We should try to cooperate with each other, setting a common goal that everyone can agree on. If the group is not able to reach consensus for a certain issue, it will have to make a decision by majority rule unless there is an alternative that can make everyone satisfied.

10. What should you do if a person may reach a decision more quickly than others and pressure people to move on before it is a good idea to do so?
    - As everyone has different but valuable ideas, it would not be a great idea to move on to the decision quickly, putting pressure on others. Understanding is the key factor in making a decision. Everyone should put his proposal in a discussion and make a decision as a whole afterwards.

11. What happens if most people on the team want to get an "A" on the assignment, but another person decides that a "B" will be acceptable?
    - If most of the people are looking forward to getting an A, then we should encourage and convince the person who is happy with B to work harder and make him realize the effort that everyone had put into this assignment.

III. Raspberry PI Installation and ARM Assembly Programming

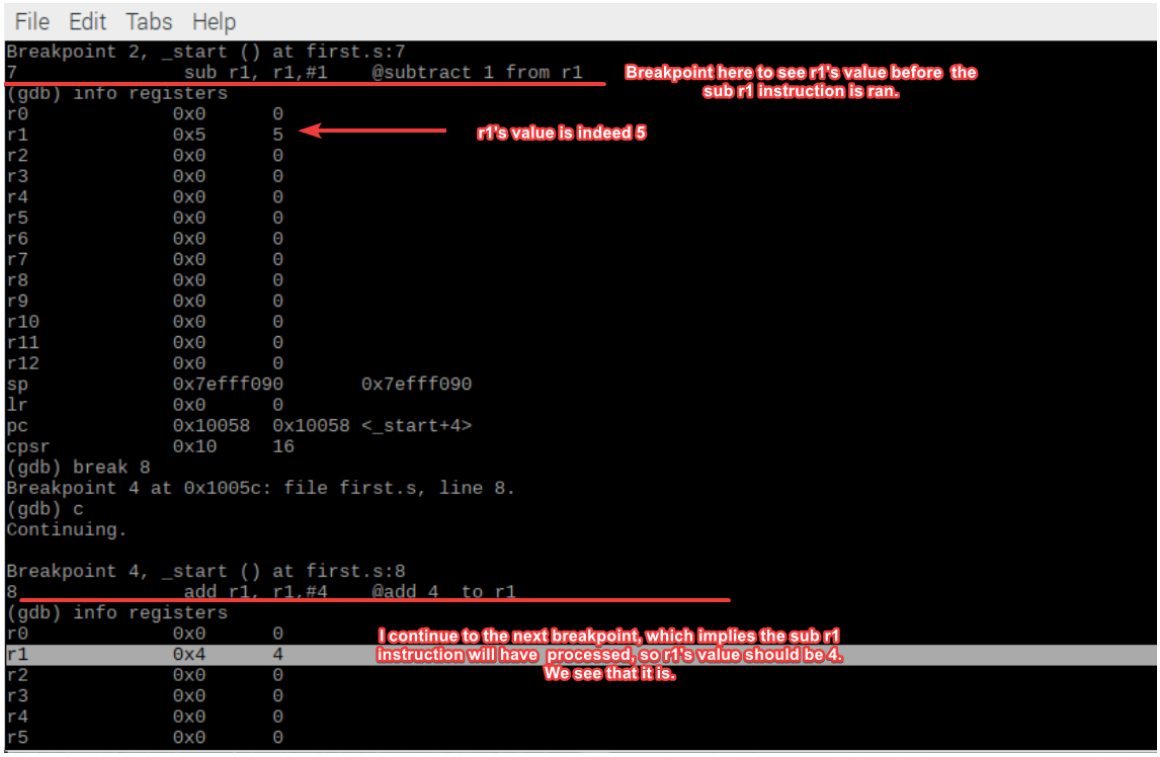
- Part 1
  a. Question: Run your program using ./first. Did you see any output, why?
     - We did not see any output the first time we ran 'first.' The reason is that the program manipulates data between CPU registers and memory and does not involve any input or output. Once we used GDB to debug the program, added a breakpoint, and entered the command 'info registers,' we were able to display and see the output.

  b. Instructions: Examine the CPU registers (to see the result). With the program stopped before line 11, the last instruction on line 10 added a literal value 4 to Register 1. We can verify that by using command "**info registers**". The display consists of three columns: the register name, the contents in hexadecimal, and the contents in decimal. The registers holding addresses such as SP or PC will not display decimal values.

(gdb) info registers

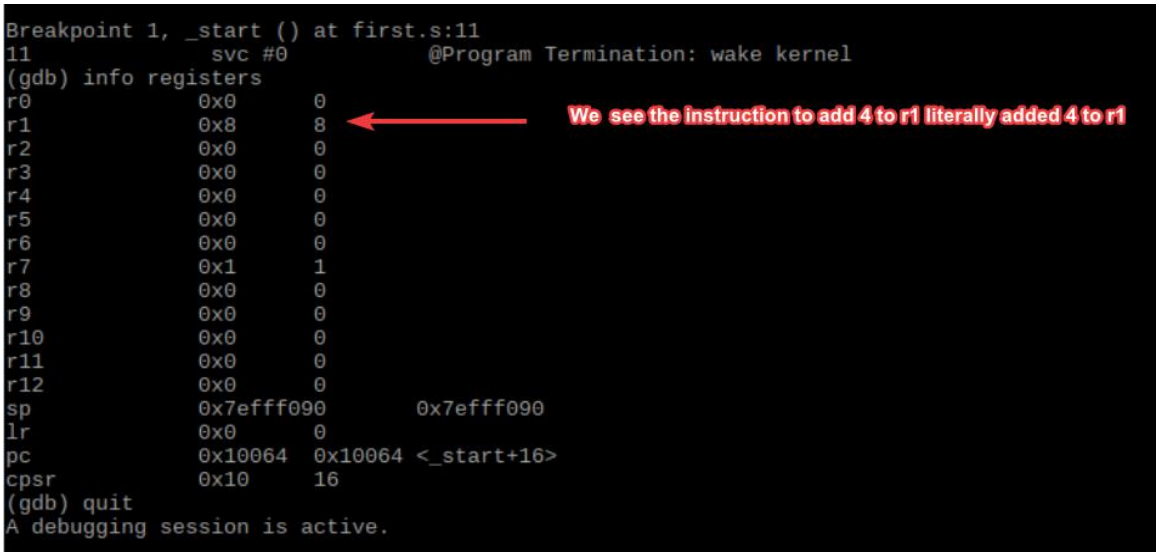
  c. We wanted to confirm and see the whole thing in action, step by step. Therefore, we added some additional breakpoints. These breakpoints showed me the process more in depth.
     - Each register is shown individually on the page. There are notes on the screenshots to show what was happening during each phase.
     - The registers are stored 0x0 except for register r1, sp, pc, and cpsr. Each of these registers have a different assigned value.

- Each of the columns (the register name, hexadecimal and the decimal) are all shown in the screenshot below.



- First break point shows r1 = 5.
- Second break point: r1 = 4, which confirms the subtraction instruction.
- Break point 4 stops the program just before the final instruction "add 4 to r1".
- Each of the steps are shown above.

d. Once the program continues, the final instruction will execute, and r1 should have the value 8 at that point. r1 is 4, and if we add 4 to 4, we get the value 8.



- We observe that r1 has the value 8. The register r7 now has a value of 1. The debugging device also played a part in making sure that the program was readable and ran correctly.

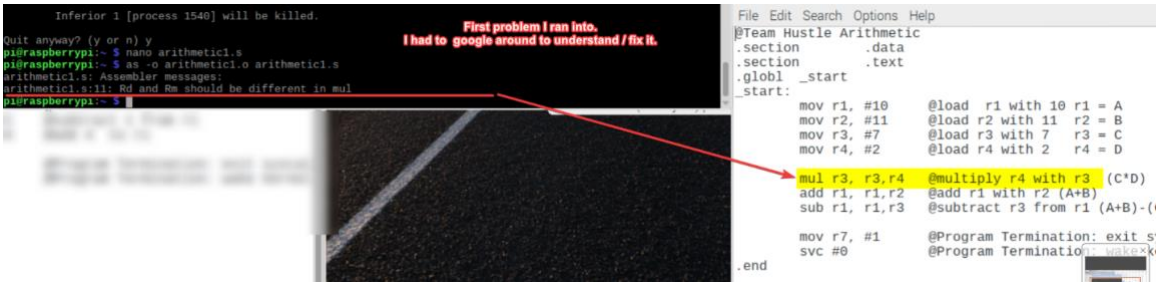
- Part 2
  a. Instructions: Using the first.s program as a reference, and use the same steps to edit, assemble, link, run, and debug the new program.
     - Write a program that calculates the following expression, (do not declare or use memory variables, only use registers):

       A = (A + B) − (C * D),
       where A=10, B=11, C=7, and D=2
       name your program *arithmetic1.s*

     - Assemble, Link, run, and debug the program.

  b. At first, we ran into an assembler error message that said, "Rd and Rm should be different in mul." After doing some research, we found that it was a MUL syntax issue with the original instruction. This was how we decided to fix it:

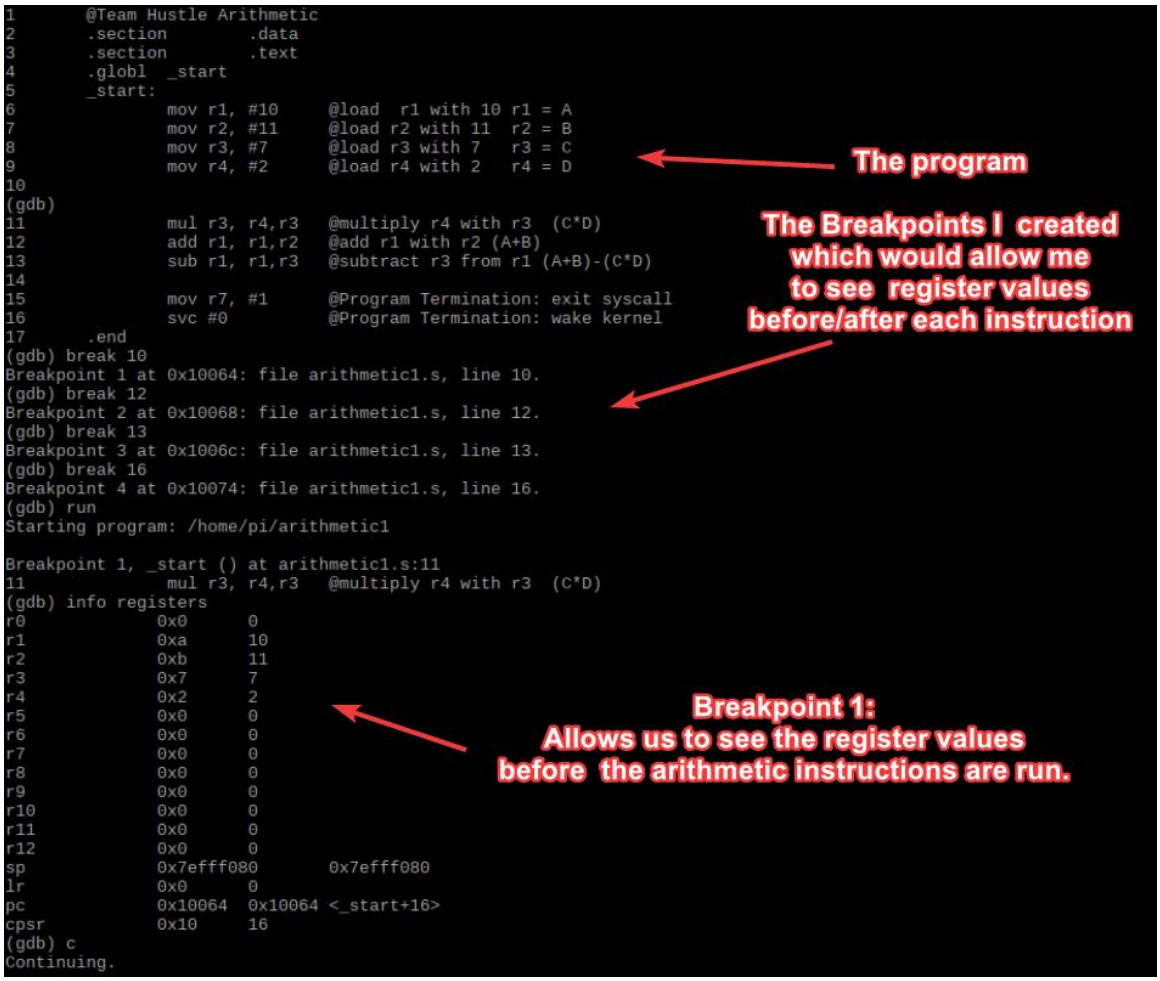
     BEFORE: mul r3, r3, r4
     AFTER: mul r3, r4, r3



mul Rd, Rn, Rm

Rd – destination register
Rn/Rm – registers holding the values to be multiplied

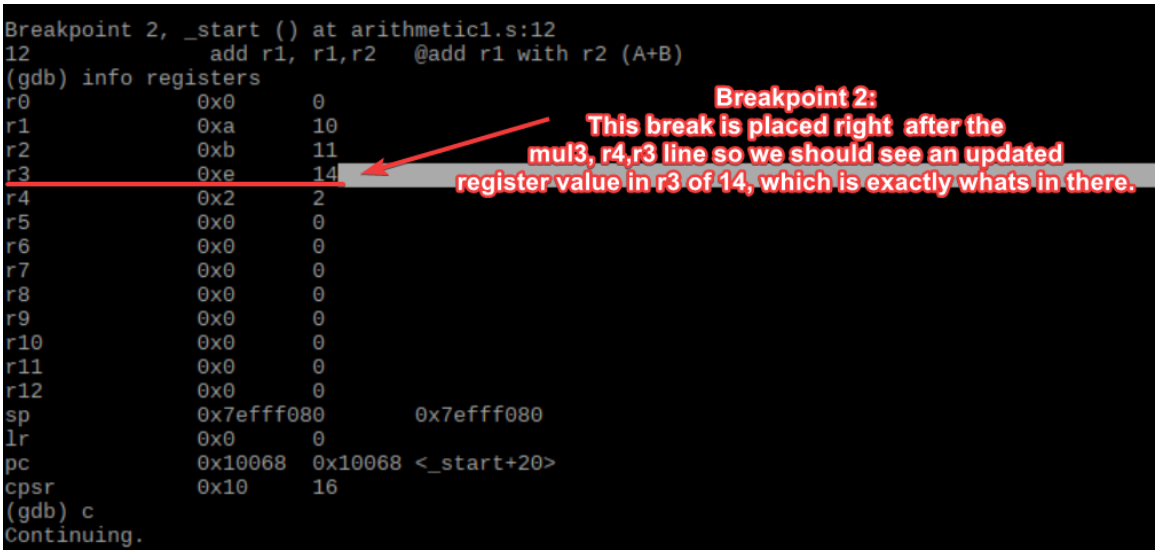
We have noticed this restriction: “Rn must be different from Rd in architectures before ARMv6.” We were not sure why exactly, but we assumed it has to do with the limitations of the instruction set as they take multiple cycles to execute, having to update Rd in each cycle.



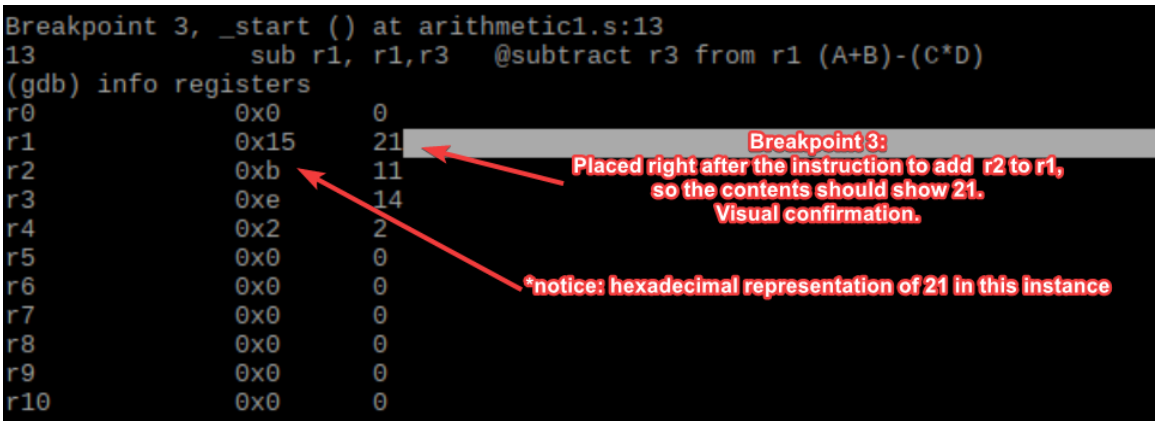

```
1       @Team Hustle Arithmetic
2       .section        .data
3       .section        .text
4       .globl  _start
5       _start:
6               mov r1, #10     @load  r1 with 10 r1 = A
7               mov r2, #11     @load r2 with 11  r2 = B
8               mov r3, #7      @load r3 with 7   r3 = C
9               mov r4, #2      @load r4 with 2   r4 = D
10
(gdb)
11              mul r3, r4,r3   @multiply r4 with r3  (C*D)
12              add r1, r1,r2   @add r1 with r2 (A+B)
13              sub r1, r1,r3   @subtract r3 from r1 (A+B)-(C*D)
14
15              mov r7, #1      @Program Termination: exit syscall
16              svc #0          @Program Termination: wake kernel
17      .end
(gdb) break 10
Breakpoint 1 at 0x10064: file arithmetic1.s, line 10.
(gdb) break 12
Breakpoint 2 at 0x10068: file arithmetic1.s, line 12.
(gdb) break 13
Breakpoint 3 at 0x1006c: file arithmetic1.s, line 13.
(gdb) break 16
Breakpoint 4 at 0x10074: file arithmetic1.s, line 16.
(gdb) run
Starting program: /home/pi/arithmetic1

Breakpoint 1, _start () at arithmetic1.s:11
11              mul r3, r4,r3   @multiply r4 with r3  (C*D)
(gdb) info registers
r0             0x0      0
r1             0xa      10
r2             0xb      11
r3             0x7      7
r4             0x2      2
r5             0x0      0
r6             0x0      0
r7             0x0      0
r8             0x0      0
r9             0x0      0
r10            0x0      0
r11            0x0      0
r12            0x0      0
sp             0x7efff080       0x7efff080
lr             0x0      0
pc             0x10064  0x10064 <_start+16>
cpsr           0x10     16
(gdb) c
Continuing.
```

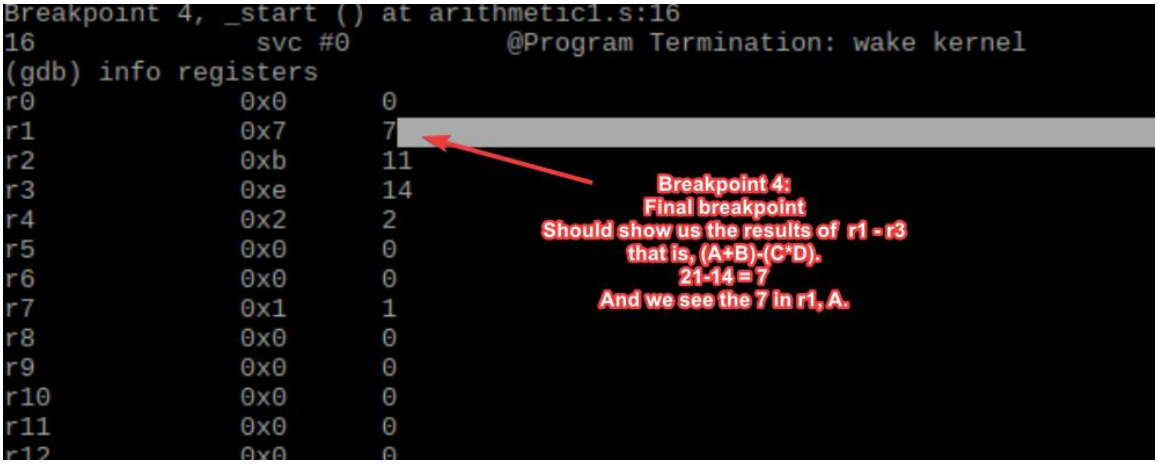
c. Breakpoint 1 Observations:
   - The second column displays the information of the registers.
   - r1: 0 x a – hexadecimal representation for the decimal value 10
   - There were many lines that explained the arithmetic parts of the program. Each breakpoint was mentioned in the program.
   - r1 = 10 / r2 = 11 / r3 = 7 / r4 = 2



d. Breakpoint 2 Observations:
   - After the calculation is done, r3 has now changed from 7 to 14.



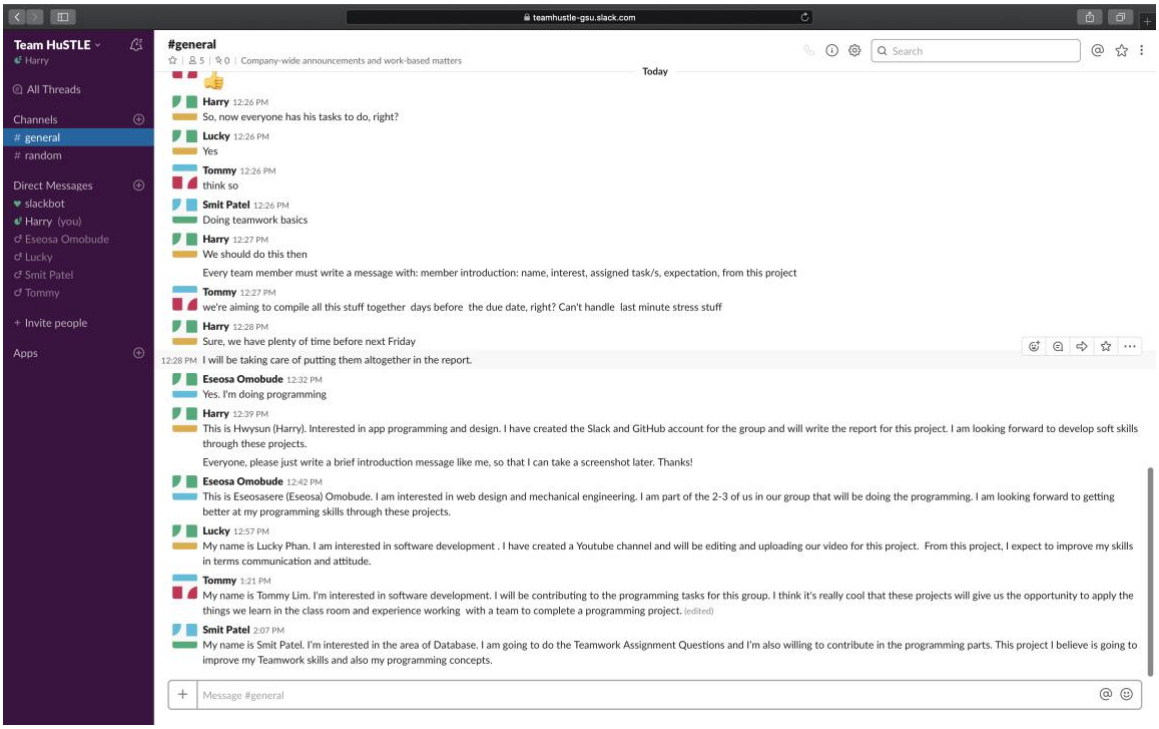
e. Breakpoint 3 Observations:
   - A + B = 10 + 11 = 21 – in hexadecimal, it is being represented as 0x15 here.



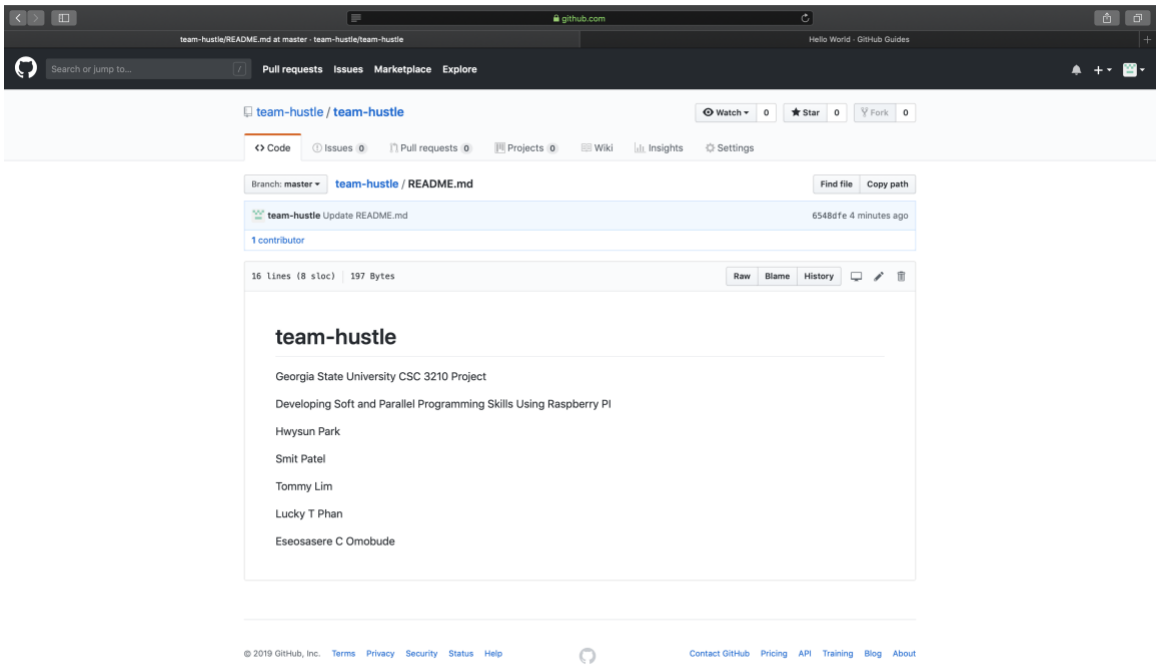
f. Breakpoint 4 Observations:
   - This final breakpoint is the result of the mathematical formula. The new result for r1 is 7.

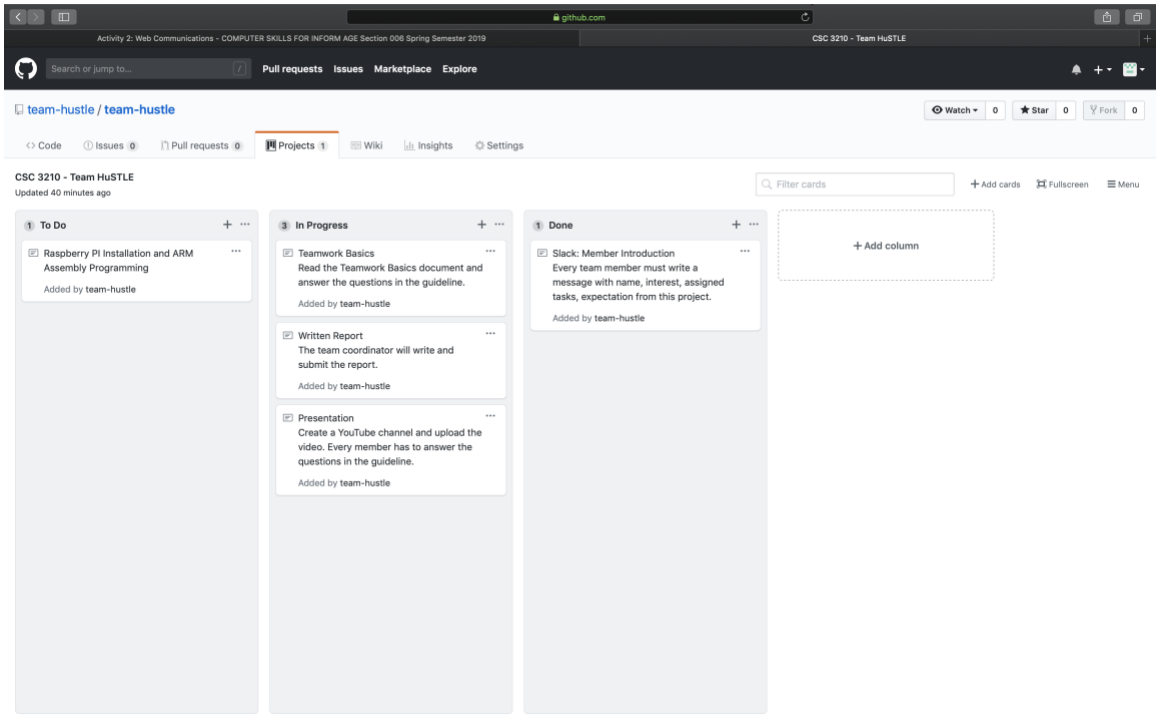
IV. Appendix

- Slack – https://teamhustle-gsu.slack.com/



- GitHub – https://github.com/team-hustle/team-hustle





- YouTube – https://www.youtube.com/channel/UCfW_InzE5vaBrMkOPuKUd7g