Developing Soft and Parallel Programming Skills Using Project-Based Learning

Fall 2018, Group 5

Austin Nocero, Rickey Clark, Paige Park, Nicholas Economou, Zhiyi Dong

Catalog----

Planning and Scheduling:

Assignee Name	Email	Task	Duration (Hours)	Dependency	Due Date	Note
Nicholas Economou	neconomou1@student.gsu.edu	Editing the video and working on task B	3 hours	Video clips/SD card	9/26/18	Must be ready 24 hours before the due date
Paige Park (Coordinator)	ppark11@student.gsu.edu	Creating Github account/Posting	3 hours	Written Report/ Code	10/4/18	Must be ready on the first day of group work
Zhiyi Dong	zdong4@student.gsu.edu	Task 4C	1 hour	Raspberry Pi Code	9/11/18	Please send everyone links and ask them to log in
Austin Nocero	anocero1@student.gsu.edu	Written Report	3 hours	Github account /Youtube Link	10/3/18	Must be ready 5 hours before deadline
Rickey Clark	rclark39@student.gsu.edu	Task 4A	3 hours	Responses from group members	9/30/18	Must be ready 5 hours before deadline

Parallel Programming Skills:

a) **Foundation**

Identifying the components on the raspberry PI B+ -

Display, Power, CPU/RAM, Ethernet controller, USB, Ethernet, Camera, RCA video , Audio, LEDS, SD card slot

How many cores does the Raspberry Pi's B+ CPU have –

The Raspberry Pi is a Quad-Core Multicore CPU so it has 4 cores.

List four main differences between X86 (CISC) and ARM Raspberry PI (RISC).Justify you answer and use your own words-

CISC and RISC differ in the instruction set and the complexity. CISC processors are more complex than RISC and have more operations and addressing modes. The amount of instruction used by both differ in the creation of code/ software between these two processors. RISC processors have simple instruction sets that operate only on registers and mainly operate faster than most CISC processors. This creates a difference in speed and execution between the two.

What is the difference between sequential and parallel computation and identify the practical significance of each?-

Sequential or Serial Computing is broken into a series of compute instructions and that execute one instruction at one time on a single processor. Parallel Computation is the use of multiple compute resources that is broken down to execute simultaneous on different processors.

Identify the basic form of data and task parallelism in computational problems.-

Data Parallelism is when the same computation is applied to multiple data items and the available parallelism is proportional to input size, leading to tremendous amounts of potential parallelism. The example of counting the 3's exhibit the basic form of data parallelism with the amount of outcomes being accumulated using the tree summation.

Task parallelism is the solutions where parallelism is organized around the functions to be performed rather than around the data. Task parallelism is exhibited in its broadest form with

client server systems that assign tasks the job of making requests and others the job of servicing requests.

Explain the differences between processes and threads.-

The difference of the two is that A process is the abstraction of a running program that does not share memory and increases with the amount of CPUs that an OS has present. While a thread is actually a process itself that is lightweight that is decomposed to smaller independent parts that are scheduled and shared within an OS and common memory.

What is OpenMPand what is OpenMP pragmas?-

OpenMP is a industry standard programming multicore architecture traced back to the late 1990's that uses an implicit multithreading model. It has Native support with GCC compilers and is easier to program than posix threads.

OpenMP pragmas are compiler directives that enable the compiler to generate threaded code. These are used as an implicit multithreading model where the library handles thread creation and management.

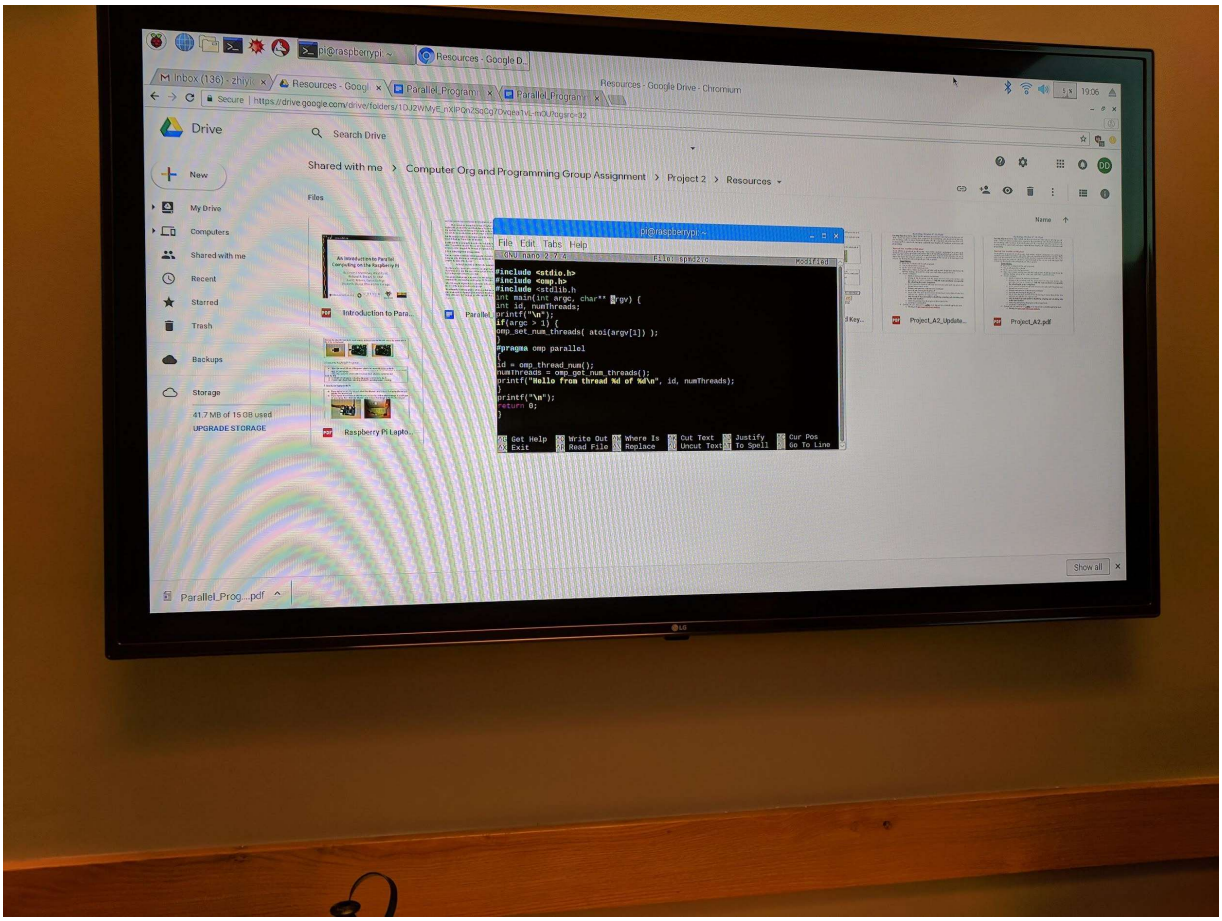
What applications benefit from multi-core(list four)?-

Why Multicore? (why not single core, list four)-

1. Many applications are multithreaded
2. It is difficult to make single-core clock frequencies higher
3. Deeply pipelined circuits
4. It is the current trend as the current shift is towards more parallelism.

c) Parallel Programming Basics

Raspberry pi Code



Here is the copied code that was given from the project description. In this code we're supposed to exam "spmd2.c". It is a C program that uses special additions to the language that make it easy to run a portion of your program on multiple threads on the different cores of a multicore machine. We complete the task with the following actions.

We first created the file by typing the following codes in a terminal window:

nano spmd2.c

We then wrote the file with the following codes:

1.#include <stdio.h>
2.#include <omp.h>
3.#include <stdlib.h>
4.int main(int argc, char argv) {**
5.int id, numThreads;
6.printf("\n");
7.if (argc > 1) {

```
8.omp_set_num_threads( atoi(argv[1]) );
9.}
10.#pragma omp parallel
11.{
12.id = omp_get_thread_num();
13.numThreads = omp_get_num_threads();
14.printf("Hello from thread %d of %d\n", id, numThreads);
15.}
16.printf("\n");
17.return 0;
18.}
```

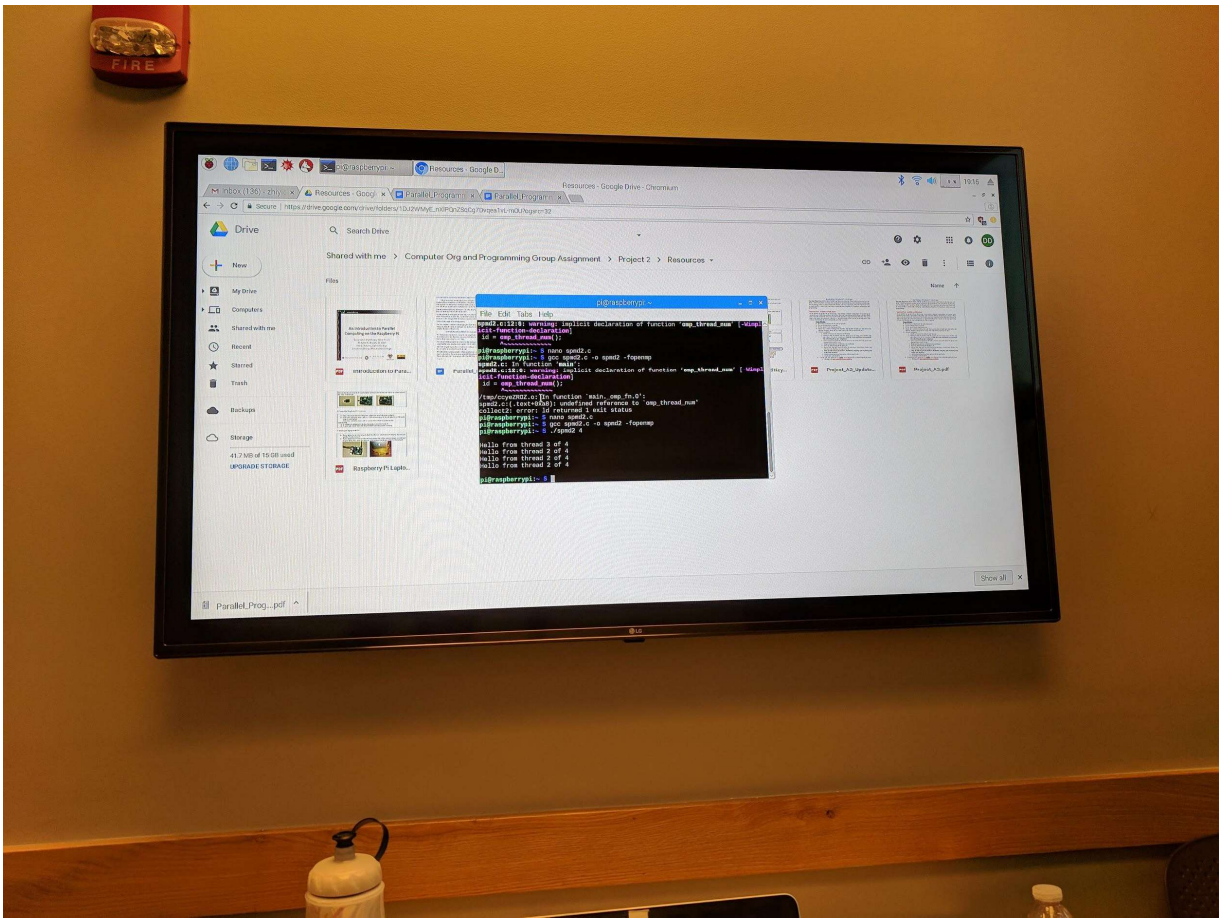
After saving and exiting from nano, we made the executable program by typing:

```
gcc  spmd2.c -o spmd2 -fopenmp
```

And this should create a file called spmd2, which is the executable program. To run the program, we typed:

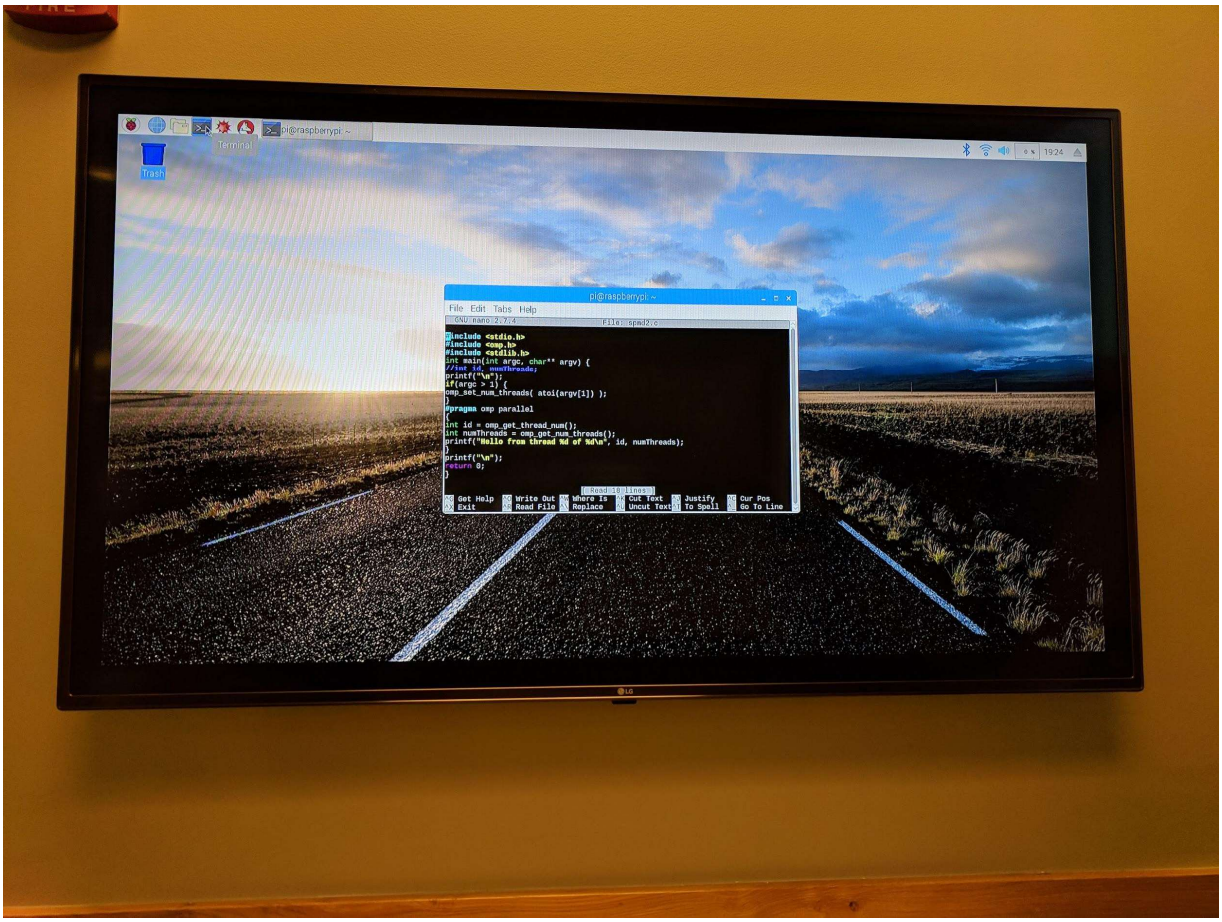
```
./spmd2 4
```

Dasdas



The 4 is called a command-line argument that indicates how many threads to fork. Since Raspberry Pi has a 4-core processor, it is natural to try 4 threads. It is fine if the thread is shown not in a natural order (When the thread finishes can not be guaranteed), but the output should be different threads shown. As as shown in the picture above, some threads appeared multiple times.

Dasd



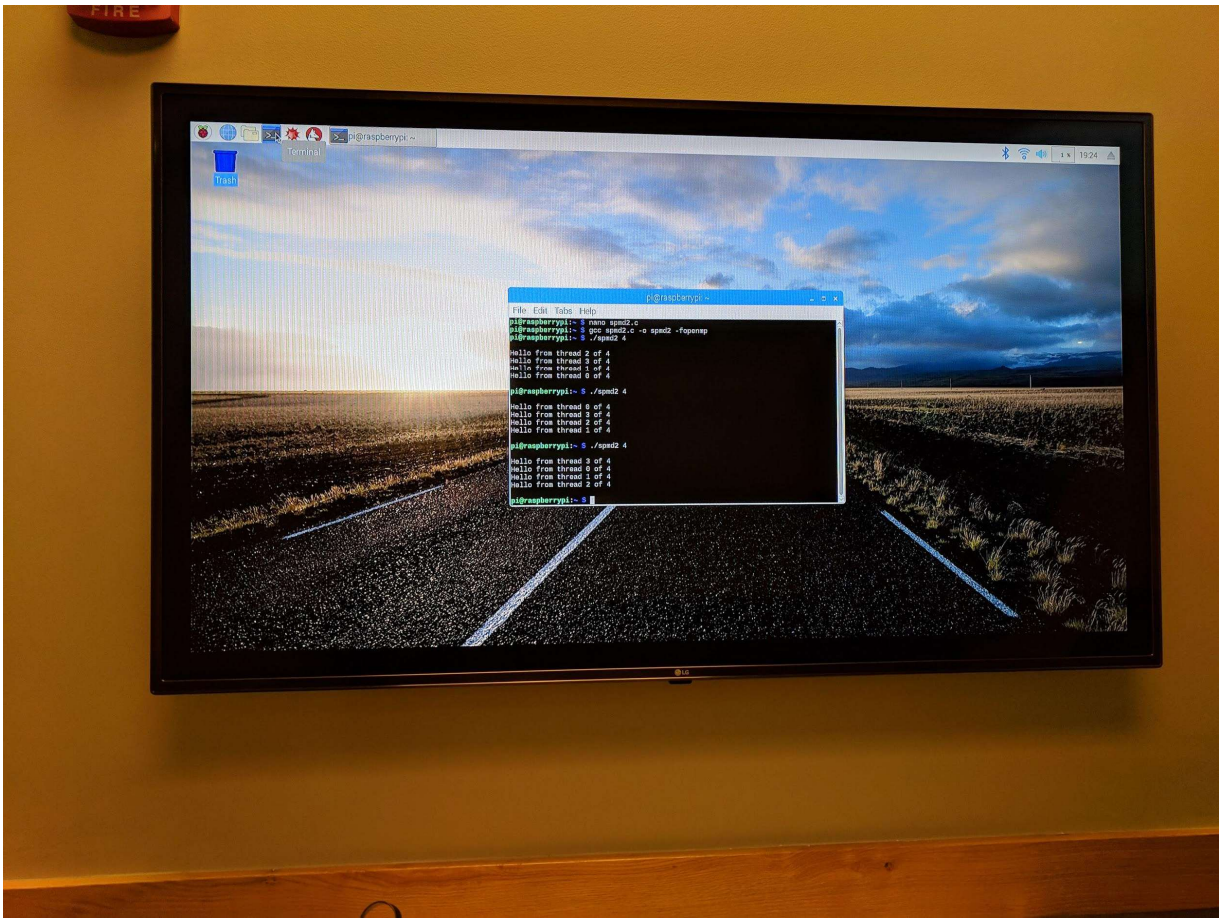
So we did the fix with the following codes:

5.//int id, numThreads;

12.int id = omp_get_thread_num();

13.int numThreads = omp_get_num_threads();

Asdasd



And, the result is corrected. The fix worked by saying in the code that each thread will now have its own private copy of the variables named “id” and “numThread” (line 12&13).

Appendix: Contact Links

Slack: https://csc3210group5.slack.com/archives/CCLF40G93/p1536007126000200

YouTube channel: https://www.youtube.com/channel/UCOP-InfUQx_cPpknqOH4cZA

Github: https://github.com/Csc3210GroupFive/Group5