**Task 3: Parallel Programming Skills**

1. Foundation

* Identify the components on the raspberry PI B+.
* Micro USB Connector  
  (power supply)
* DSI Display Connector
* HDMI Connector
* CSI Camera Connector
* RCA Video/Audio Jack
* 10/100 Ethernet Port
* 2 USB 2.0 ports
* Ethernet Controller
* CPU/RAM
* How many cores does the Raspberry Pi’s B+ CPU have?
* Quad-Core (4 Cores) CPU
* List three main differences between X86 (CISC) and ARM Raspberry PI (RISC). Justify your answer and use your own words.
* While X86 processors can take many instructions to its memory, ARM processors can only use registers to operate instructions and have to load/store to access memory.
* X86 processors have larger instruction set with more features, allowing for more operations with less registers. ARM processors have a reduced instruction set with more registers, allowing the instructions to be executed more quickly.
* Unlike X86 processors, almost every instruction in ARM processors can be executed conditionally.
* X86 processors use the little-endian format which addresses, sends, and stores the least significant byte first and the most significant byte last. ARM processors are known as bi-endian which can be operated in either little-endian or big-endian mode.
* What is the difference between sequential and parallel computation and identify the practical significance of each?
* In sequential computation, a problem is broken into a series of instructions that are executed sequentially one after another. In parallel computation, a problem is broken into discrete parts that can be solved concurrently. Then each part is broken down to a series of instructions that execute simultaneously on different processors. Traditionally, software has been written for sequential computation on a single processor. Parallel computing can solve more complex problems in less time with the simultaneous use of multiple compute resources.
* Identify the basic form of data and task parallelism in computational problems.
* Data parallelism refers to a broad category of parallelism in which the same computation is applied to multiple data items, so the amount of available parallelism is proportional to the input size. Task parallelism applies to solutions where parallelism is organized around the functions to be performed rather than around the data.
* Explain the differences between processes and threads.
* A process is the abstraction of a running program. A thread is a lightweight process that allows a single process to be decomposed to smaller, independent parts. Processes do not share memory with each other, whereas all threads share the common memory of the process they belong to. Even though a process can be operated on a single core at a time, threads can be scheduled on separate cores as available.
* What is OpenMP and what is OpenMP pragmas?
* OpenMP is one of the libraries/languages to program multicore architectures. It is the industry standard since late 1990s. It has a native support with GCC compilers and is easier to program than POSIX threads. It uses an implicit multithreading model in which the library handles thread creation and management. OpenMP pragmas are compiler directives that enable the compiler to generate threaded code.
* What applications benefit from multi-core? (list four)
* Database servers
* Web servers
* Compilers
* Multimedia applications
* Scientific applications
* CAD/CAM
* Applications with thread-level parallelism
* Why Multicore? (why not single core, list four)
* An operating system can execute more processes at once with more CPU cores.
* It is difficult to make single-core clock frequencies even higher.
* Deeply pipelined circuits cause heat problems, speed of light problems, design and verification issue, etc.
* Many new applications are multithreaded.
* It is a general trend in computer architecture, shifting toward more parallelism.