CHAPTER IV
ANALYSIS AND DESIGN

4.1 Analysis

Sequential computing is a computational method that uses a single CPU to run a program. That program executes each line of instruction one by one. So there is only one instruction that can be executed at one time by sequential computing.

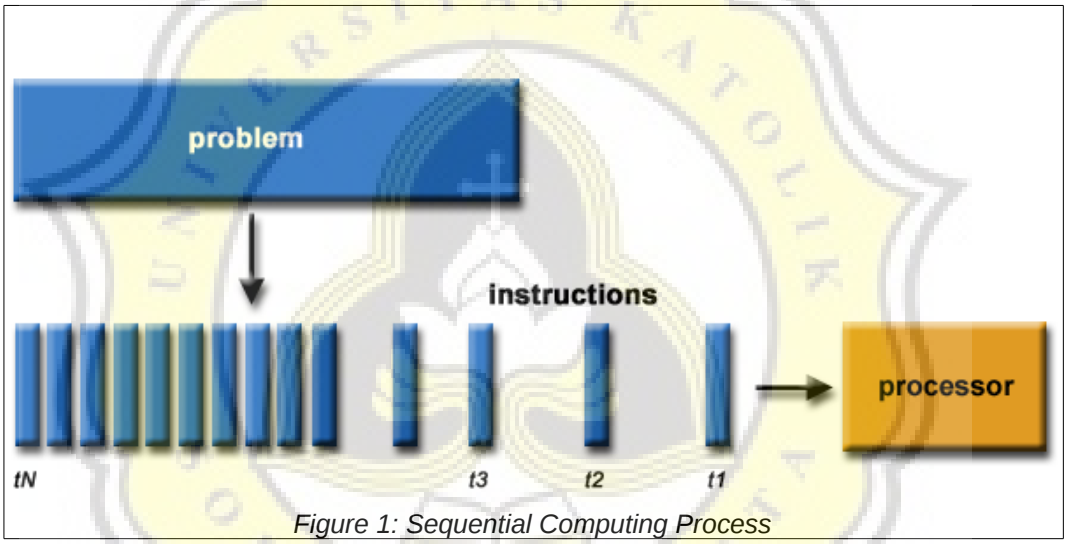

Figure 1: Sequential Computing Process

Different from sequential computing, parallel computing is using many CPUs to run a program. In the execution, a parallel computing split the main problem into separated parts of job, and then executed them by many CPUs simultaneously. So there are a lot of job that can be executed at once by parallel computing compared with sequential computing.

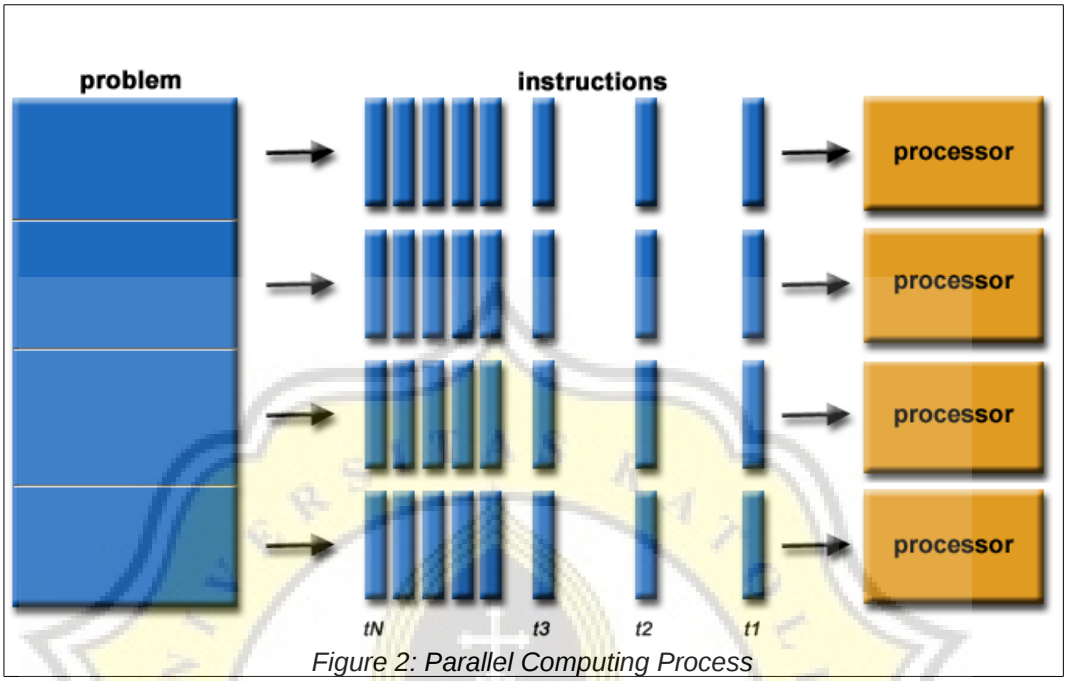

Figure 2: Parallel Computing Process

4.1.1. Bubble Sort Algorithm in Single Processor

There are several steps that required by single processor to resolve bubble sort algorithm with *n* data in ascending:

1. Comparing the data (data *n)* one by one with the data afterward (*data n* +1).
2. Swap the data when the comparison result is bigger (data *n* is greater than data *n*+1).
3. Then do the iteration for steps 1 and 2 until the greatest data in the last position. This step is called a phase.
4. After that, do the iteration of a phase until all data sorted (repeated *n* times).

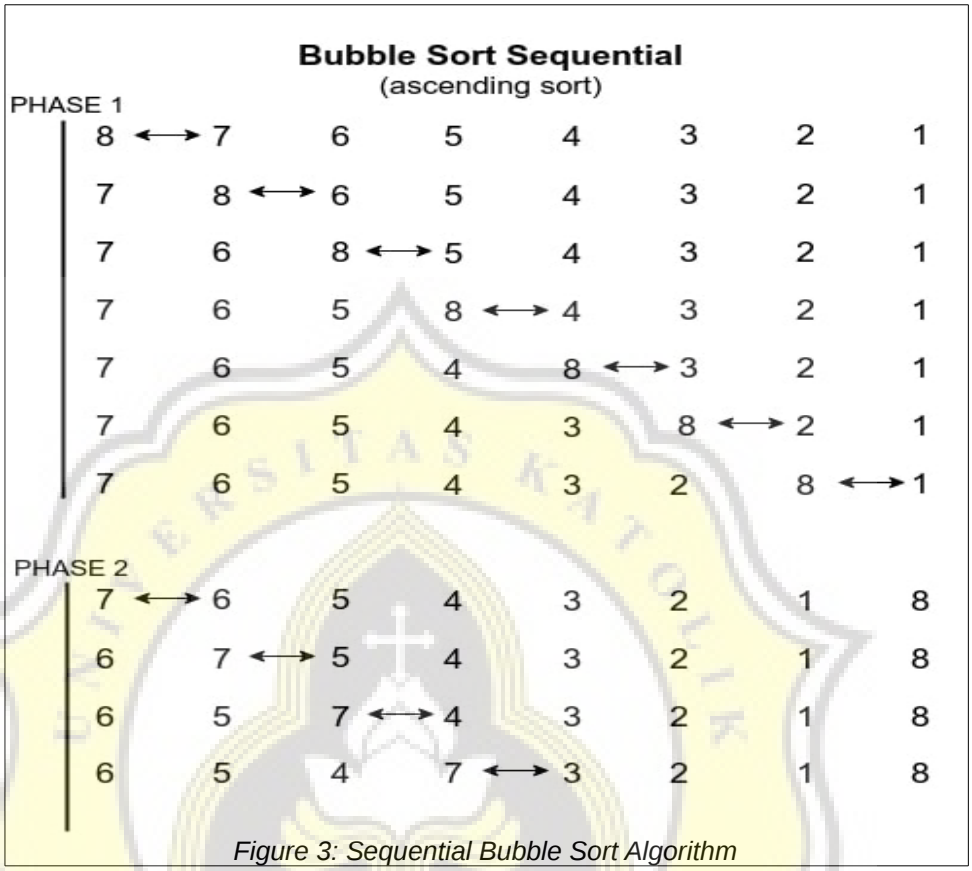

Figure 3: Sequential Bubble Sort Algorithm

On this bubble sort, each step of bubbling (data exchange) solved with a multilevel of iteration, which for the worst case makes this bubble sort has a Big O level of time cost O(n2). The calculation formula for ascending bubble sort are using 8 unsorted data, a single processor requires 7 steps to complete phase 1, to complete phase 2 takes 6 steps, then it takes 5 steps to complete phase 3 and so on until the entire data are in ordered (repeated *n-1* times). So if this steps calculated entirely, a single processor requires 28 steps (7 + 6 + 5 + 4 + 3 + 2 + 1 + 0) to be able to complete this job.

4.1.2. Bubble Sort Algorithm in Parallel Computing

The implementation of single computer algorithm in parallel has a lot of techniques. One of them is pipeline. On pipeline, the main problem is divided into a set of tasks. That can be completed by a separate process, where each process does not interfere with other processes.

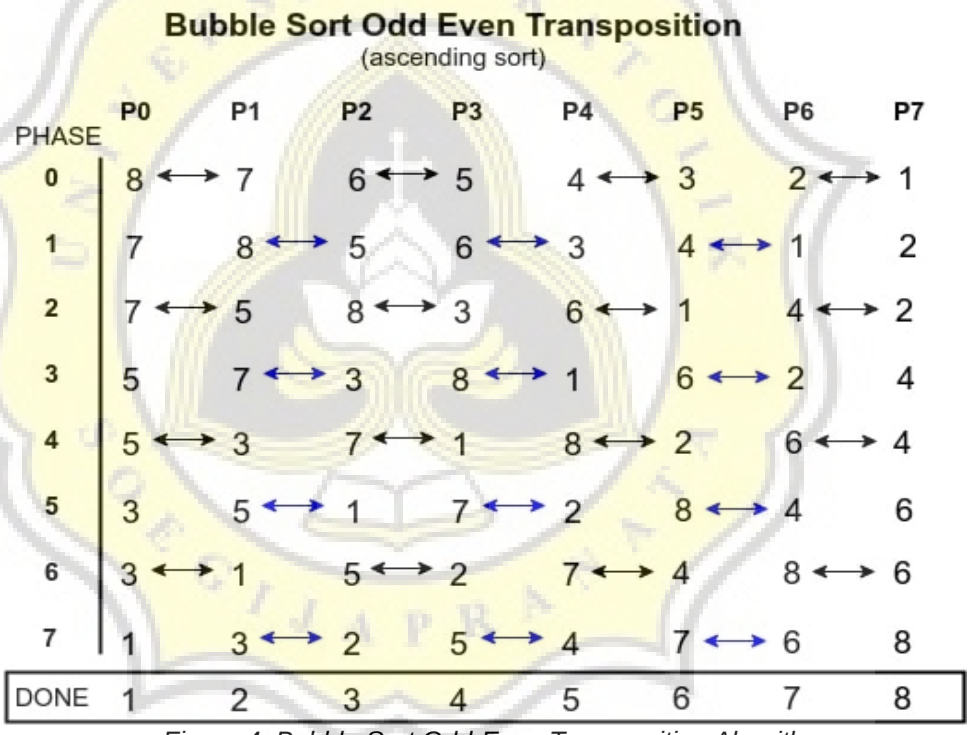

Figure 4: Bubble Sort Odd-Even Transposition Algorith

The bubble sort algorithm with pipeline technique is called Bubble Sort Odd-Even Transposition. This technique will generate an odd and even pattern so it is called odd-even transposition.

At the execution, every data is mapped in a process (*P*) and to be done in several phase. In an even-phase, even-numbered process will comparing the data with the next odd-process. If the

data of even-numbered process is bigger then the next odd-process, it will swap the data. Likewise in the odd-phase, odd-numbered process will comparing the data with the next even-process. If the data of odd-numbered process is bigger then the next even-process, it will swap the data.

The result of this algorithm is only required 8 steps or it has a Big O level of time cost O(n). That mean its possible if bubble sort algorithm with parallel aproach can be more efficient than the bubble sort with single processor .

4.2 Design

Based on the analysis of bubble sort algorithm with parallel aproach, a program will be created with C languange and PVM library. PVM is a master-slave programming, which are divided into two parts: master and slave program. The master program is executed at the first time and have responsibility to initiate the slave program, to manage the process, and to perform data collection. Besides, slave program computes the tasks assigned by the master program.

4.2.1. Master Program

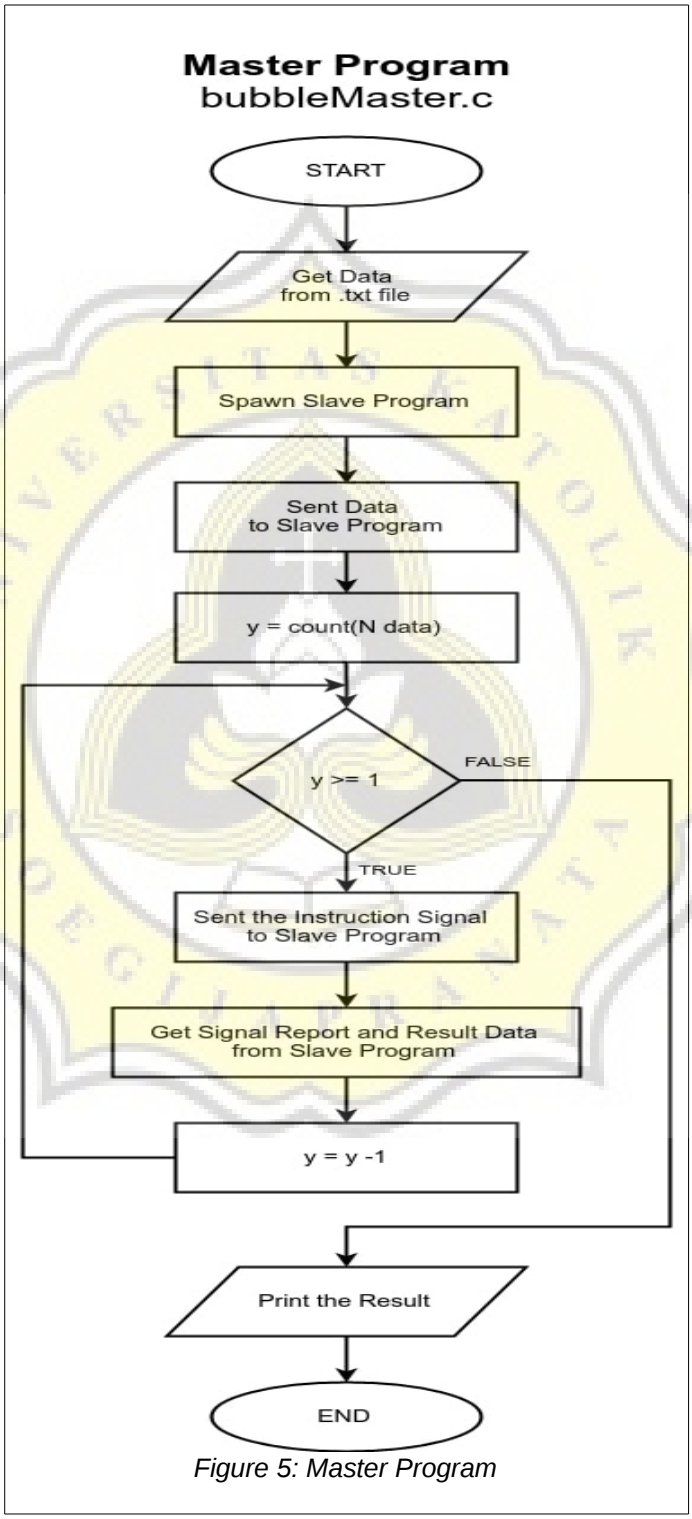

Figure 5: Master Program

From this flowchart, first master program will get the data by reading from a text file. After all data is obtained, master program will spawn the process (slave program). After the slave programs spawned, the master program will divide the data and send it to slave program.

To begin the computation of slave program, Master program sends an instruction signal and wait for feedback signals from all slave to transmit the next instruction. Finally, after all the instruction of the master program is completed, the master program will collect all the data from each slave and print the result.

4.2.1. Slave Program

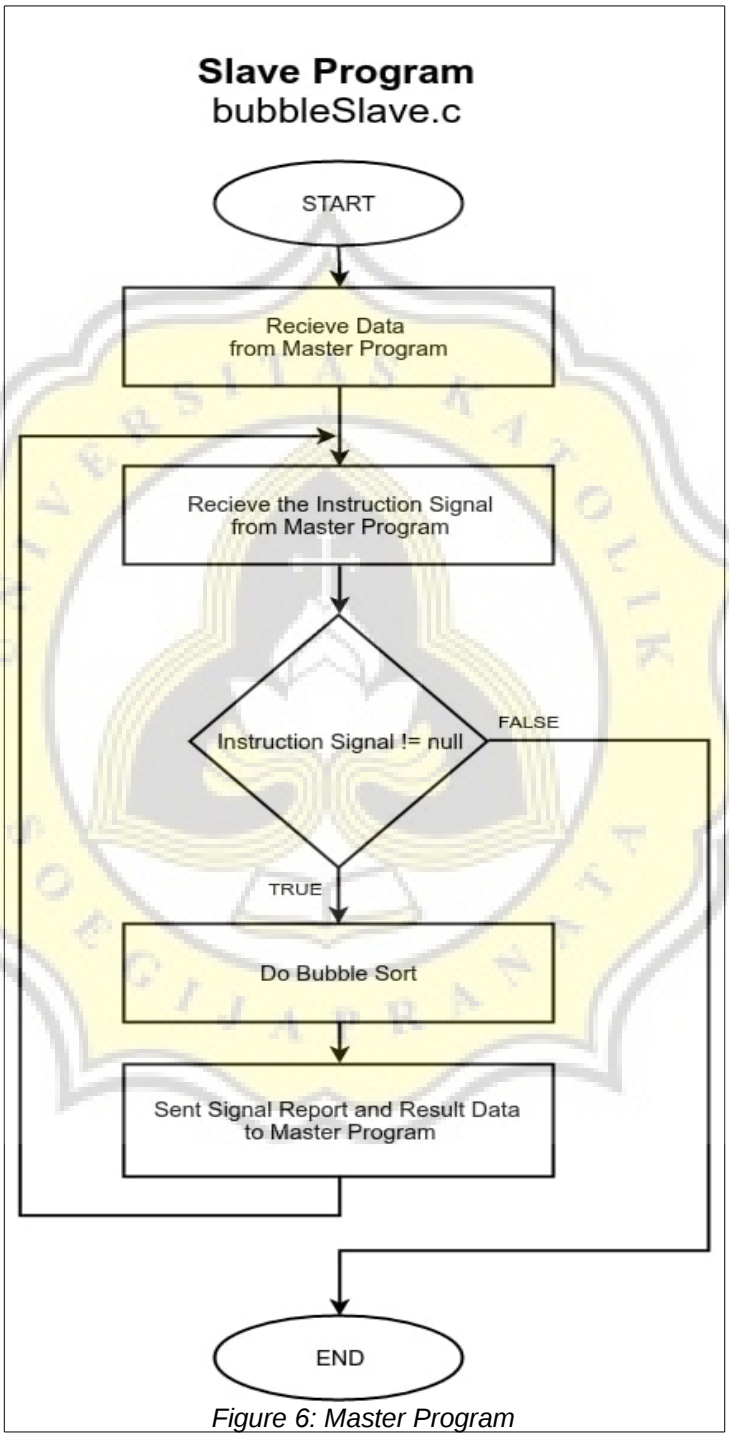

Figure 6: Master Program

Slave programs will be started when it are spawned by the master program. Then slave programs will wait the data provided by master program and signal instruction to carry out their duties. Each task has been completed, the slave will send the feedback signal to the master program as a sign that the jobs has been completed. Slave programs will terminated when all the instruction of master program have been completed.