# Study Guide

## Week 7 February 22nd, 2016 – February 28th, 2016

Author B. Wilkinson Modification date February 21st, 2016

## Study Materials on Moodle

- *PowerPoint Slides*
    - Paraguin-1
    - Paraguin-2
    - Paraguin-3
- *Video*
    - Lecture 11 video: 75-minute video of Lecture 11 in Fall 2014 on Assignment 4 and the Paraguin compiler directive approach.
    - Lecture 12 video: 75-minute video of Lecture 12 in Fall 2014 on Paraguin.
- *Sample Quiz Questions*
    - Paraguin

## Tasks

- **Mini-Quiz:** Answer the short posted quiz before 11:55 pm Sunday February 28th, 2016.
- **Continue working on Assignment 3** MPI tutorial, using command line and Eclipse-PTP
    - Assignment 3 Due: *Sunday February 28th 2016* (Week 7)
- **Assignment 4 Second MPI Assignment - Monte Carlo π workpool**
    - Assignment 4 Due: *Sunday March 13th, 2016 (Week 9)*

## Moodle Saba meeting – *Friday February 26th, 2016, 5 pm*

*Paraguin–1* describes the Paraguin compiler, which was developed at UNC-Wilmington. Paraguin uses "pragma" compiler directives to create MPI code to avoid the programmer having to write in MPI directly. The Paraguin compiler takes a C program with Paraguin directives and converts the program into MPI source code. This MPI source code then has to be compiled with an MPI compiler to create executable code. As such, Paraguin is a "source-to-source" compiler. Rather than installing the Paraguin compiler locally, a web page is provided at UNC-Wilmington that enables one to upload a C program with Paraguin directives, compile it, and download the resulting MPI program, which can then be compiled locally as an MPI program. The basic Paraguin directives constructs are similar, but not identical, to those in OpenMP except they create MPI code. Just as in OpenMP, a parallel region has to be created within which other parallel constructs are formed, such as the parallel for construct (called forall).

*Paraguin-2* continues the description of Paraguin. Paraguin offers pre-implemented message passing patterns not found in OpenMP, including scatter, gather, and stencil.

*Paraguin-3* shows how to implement matrix addition, the travelling salesman problem, and the Sobel edge detection algorithm using Paraguin.

You are expected to know the syntax of Paraguin directives and be able to write programs using Paraguin directives. However we will not have an assignment of Paraguin that was previously done in Fall 2014.

Instead we will explore another tool for writing pattern-based MPI code developed at UNC-Charlotte in an assignment later in the course.

*Assignment 4 Instructions:* Assignment 4 is the second assignment on MPI. Whereas the first MPI assignment was basically a tutorial, this time you are to write your own MPI program without any sample code - to compute π by the Monte Carlo method using a workpool pattern. As usual, first you are to write sequential code and then MPI code. The code is executed on your computer and on the cluster. You have two weeks to do this assignment but start early.

***Some things to note about writing an MPI program:***

An **MPI_Send()** routine returns after its local actions are complete and generally *before the message has been received.* If you want the source process to wait for the message to arrive at the destination, use **MPI_Ssend()**. In fact, the code must work with **MPI_Ssend(),** as **MPI_Send()** can degenerate into **MPI_Ssend()** if system buffer space becomes exhausted.

The **MPI_Recv()** routine also returns after its local actions are complete but this means *when the message has been received and it will wait for the message*. So it is easy to deadlock MPI programs! If an **MPI_Recv()** does not get the message it is waiting for, it will simply carry on waiting.

To find information about a received message, read the status structure. **status.MPI_TAG** holds the tag of the received message. **status.MPI_SOURCE** holds the rank of the source of the message.