**Parallel Patternlets**

**CSinParallel Pro ject**

June 10, 2016

**CONTENTS**

1. Parallel Programming Patterns 2
   1. An organization of parallel patterns 2
2. Message Passing Parallel Patternlets 4
   1. Source Code 4
   2. 00. Single Program, Multiple Data 4
   3. 01. The Master-Worker Implementation Strategy Pattern 5
   4. 02. Message passing 1, using Send-Receive of a single value 6
   5. 03. Message passing 2, using Send-Receive of an array of values 7
   6. 04. Message passing 3, using Send-Receive with master-worker pattern 8
   7. 05. Data Decomposition: on *equal-sized chunks* using parallel-for 9
   8. 06. Data Decomposition: on *chunks of size 1* using parallel-for 10
   9. 07. Broadcast: a special form of message passing 11
   10. 08. Broadcast: send data to all processes 12
   11. 09. Collective Communication: Reduction 13
   12. 10. Collective Communication: Reduction 14
   13. 11. Collective communication: Scatter for message-passing data decomposition 16
   14. 12. Collective communication: Gather for message-passing data decomposition 17
   15. 13. The Barrier Coordination Pattern 18
   16. 14. Timing code using the Barrier Coordination Pattern 20
   17. 15. Sequence Numbers 21
3. Shared Memory Parallel Patternlets in OpenMP 24
   1. Source Code 24
   2. Patternlets Grouped By Type 25

**i**

**Parallel Patternlets ,**

**Las t Updated:**2016-06-10

This document contains simple examples of basic elements that are combined to form patterns often used in programs employing parallelism. We call these examples *patternlets* because they are deliberately trivial, small, yet functioning programs that illustrate a basic shell of how a particular parallel pattern is created in a program. They are starting points you can use to create realistic working programs of your own that use the patterns. Before diving into the examples, first there will be some background on parallel programming patterns.

**CONTENTS** **1**