IMPLEMENTATION OF MPI-BASED WIMAX BASE STATION SYSTEM FOR SDR

Hyohan Kim (HY-SDR Research Center, Hanyang Univ., Seoul, South Korea; hhkim@dsplab.hanyang.ac.kr); Chiyoung Ahn(HY-SDR Research center, Hanyang Univ., Seoul, South Korea; ahncy@dsplab.hanyang.ac.kr); June Kim (HY-SDR Research Center, Hanyang Univ., Seoul, South Korea; nzneer@dsplab.hanyang.ac.kr); and Seungwon Choi* (corresponding author, HY-SDR Research Center, Hanyang Univ., Seoul, South Korea; choi@ieee.org)

ABSTRACT

In this paper, we propose a new concept of SDR base station system which adopts a parallel processing technology of clustering environment. We implemented a WiMAX system with SDR technology which adopts the method of Message Passing Interface (MPI) mainly for speed-up operations. In order to maximize the efficiency of parallel processing in signal processing, we analyze how the algorithm at each of modules is related to data to be processed. Through the implemented system, we show a drastic improvement in operation time due to parallel processing using the proposed MPI technology. In addition, we demonstrate a feasibility of SDR system for 4G or even beyond-4G as well.

1. INTRODUCTION

Software Defined Radio (SDR) technology is a key technology to deal with rapidly changing mobile communications market [1]. SDR system supports various communication protocols and services through software download on an open-architecture hardware platform. As such, SDR base station is provided a great advantage in flexible adoption of new communication standards only by downloading the necessary software modules without hardware changes. Also, the component expenses are tremendously reduced because many specific hardware components are replaced with software packages, which eventually enable the installation of base station system at geographically adverse environments much easier.

In SDR system, operation time is a key issue because SDR is mainly operated by software packages instead of hardware devices. Recently, there are rapidly growing needs for high-speed data transmission due to the increase of wireless data traffic. The SDR base station has to handle more operations for the increasing data so that it requires a lot faster processing speed. In the parallel computing based on computer cluster, MPI is adopted as a de-facto standard for the parallel processing [4]. MPI can support most High Performance Computing (HPC) platforms. Also, MPI can easily be adapted to systems because of various libraries available for implementing the standard [4].

In this paper, with an intention to improve the operating speed of SDR-based base station, we present how to apply the parallel/distributed technology of MPI in a cluster environment to implement WiMAX system with SDR technology. This paper consists of the following structure. Section 2 explains MPI and parallel/distributed processing. Section 3 shows the system architecture of the implemented system. In Section 4, we present how to implement WiMAX system using the MPI. The result of system performance and evaluation are then described in Section 5. Finally, Section 6 concludes the paper.

2. DESCRIPTION OF MPI

MPI is an Application Program Interface (API) specification that allows multiple nodes to exchange messages with one another [2]. MPI has not been approved as an official standard by any major standardization body. Nevertheless, it is a de-facto standard for parallel programs to be executed on computer clusters or supercomputers. MPI standard includes point-to-point message-passing, collective communications, group and communicator concepts, process creation and management, one-sided communications, etc. There are a variety of open projects for implementing MPI standard. In this paper, we have used an Open MPI to implement our system [3].

Figure 1 shows a general MPI program structure. First, MPI environment has to be initialized after including MPI header file, which defines all parameters and functions of MPI library, as shown in Figure 1. During the initialization, the number of nodes is decided and the unique id is allocated to each node. Then, each node carries out parallel processing by exchanging data with message. Upon completion of the

parallel processing, MPI environment is terminated and returns the resources.

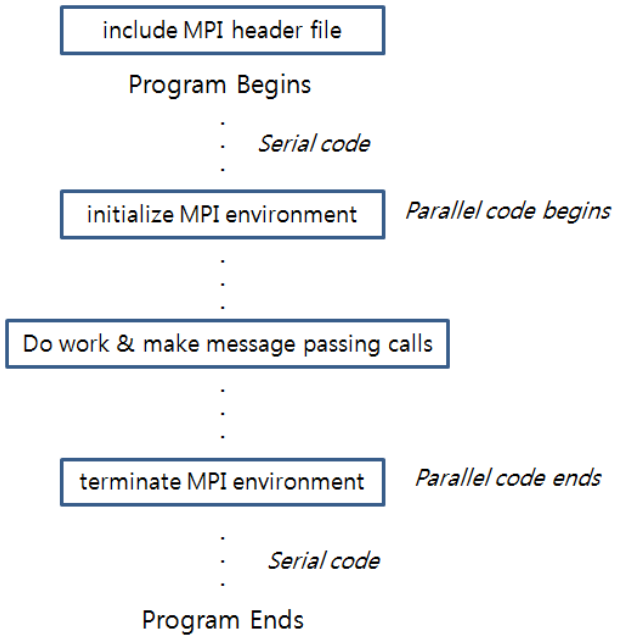


Figure 1. General MPI program structure

Motivations of using MPI are as follows:
First, MPI is the only message passing library which can be considered as a standard. Second, MPI supports almost all HPC platforms. Therefore, there is no need to modify the source code in order to port the application to another hardware platform, once it supports the MPI standard. Also, over 115 routines have been defined in MPI standard and a variety of implementations are available.

3. SYSTEM ARCHITECTURE

3.1. Hardware Components

The WiMAX system implemented in this paper performs all baseband signals processing within a general-purpose PC without using any extra hardware resources. In Figure 2, system components and architecture are illustrated. Each node is a general-purpose computer and they are connected with each other like a cluster transmitting/receiving the data through the Gigabit Ethernet. USRP2 is used as RF transceiver which conducts the analog-to-digital (AD) / digital-to-analog (DA) conversion and decimation / interpolation. USRP2 and PC are connected through Ethernet.

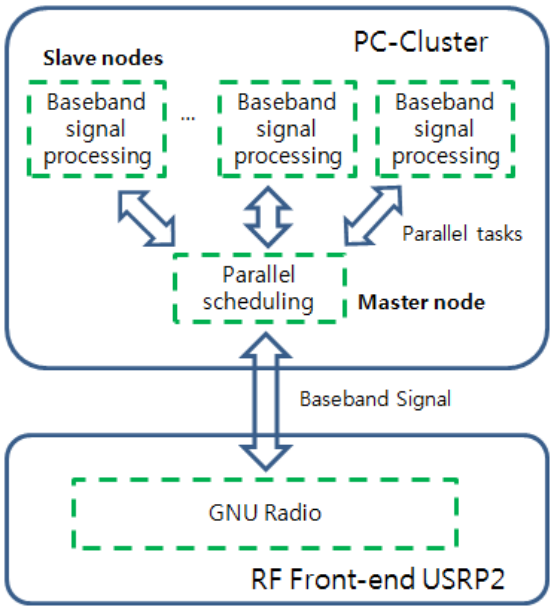


Figure 2. System components and architecture

3.2. Software Components

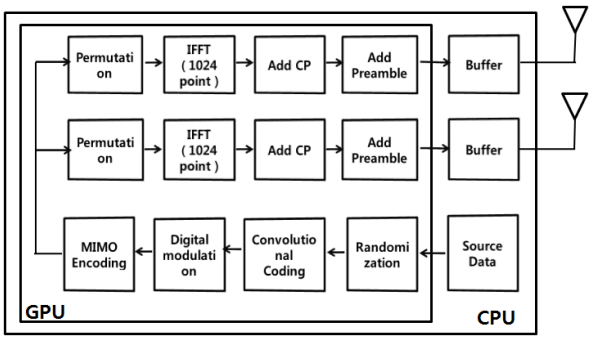
From Figure 2, it can be observed that software components consist of 3 parts: Parallel scheduling, Baseband signal processing and GNU Radio. In PC-cluster, a lot of nodes are clustered. The master node conducts the parallel scheduling. In other words, the master node distributes or gathers the baseband signal to or from the slave node, respectively. Slave node processes baseband signals which are allocated from the master node and delivers the result of processing to the master node. GNU radio is a driver needed for using USRP2, RF transceiver.

3.3. Two-level Parallel Processing

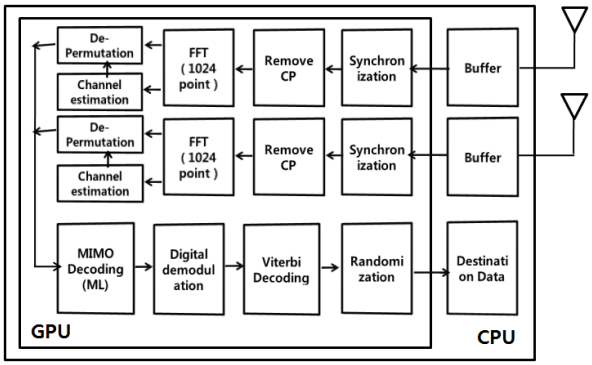
In the implemented system, the slave node is realized using Graphics Processing Unit (GPU) in order to process the baseband signals in parallel processing. To maximize the number of ALU available in the GPU which is used to process the baseband signals in the slave node, baseband signal processing algorithm should be parallelized in accordance with the multi-thread architecture of the GPU. The parallelization of the baseband signal processing algorithm is explained in more details later in Section 4.

4. IMPLEMENTATION OF WIMAX SYSTEM

4.1. Structure of entire system



(a) transmitter



(b) receiver

Figure 3. Block diagram of 2x2 SM MIMO WiMAX system

In Figure 3, a block diagram of 2x2 Spatial Multiplexing (SM) MIMO WiMAX system is illustrated [5]. The CPU conducts as a buffer which connects USRP2 to PC. It also performs encoding of the source data and decoding of the destination data. All other processing of baseband signals are conducted by GPU.

In Transmitter, as shown in Figure 3(a), CPU encodes the source data and then delivers it to GPU. GPU generates the WiMAX frame and delivers it to CPU. CPU transmits the WiMAX frame through the RF transceiver, USRP2. Above procedures are conducted in the master node.

In Receiver, as shown in Figure 3(b), the received data which are received through USRP2 are stored in the buffer of CPU. In Figure 3, GPU attains the destination data by processing the decoding functions successively.

4.2. Parallelization of baseband signal processing

In Figure 4, computation time of each of WiMAX function blocks is illustrated. The computation time is a processing time taken for encoding/decoding a frame data at one node which has been implemented on GPU. It can be observed that Viterbi Decoder block and Maximum Likelihood (ML) block take relatively a lot more time compared to the other function blocks as shown in Figure 4.

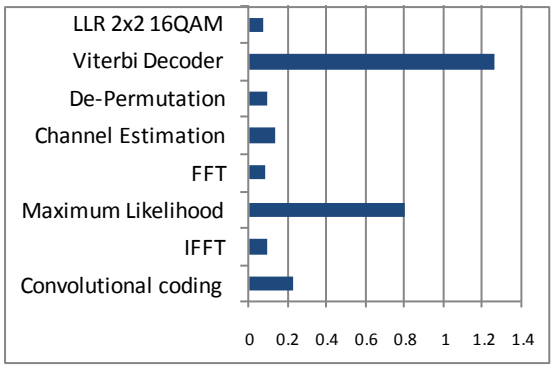


Figure 4. Computation time(ms) of 2x2 SM MIMO WiMAX system

To reduce the computation time of the entire system, it is necessary that MPI is used to design Viterbi decoder block and ML block in such a way that the parallel processing can be exploited among the nodes. It can also be observed that synchronization block for finding the frame start also takes pretty a lot of processing time, i.e., about 7.8ms. The reason is that the synchronization block has a lot of computational load for calculating correlations of all received signals. Let's observe the parallelization of Viterbi Decoder block, ML block and synchronization block in more details in the following subsections.

4.2.1. Viterbi decoding

Amongst a lot of algorithms for decoding the convolutional code, Viterbi algorithm has been used most widely. In Figure 5 illustrates the Trellis diagram for decoding the convolutional code.

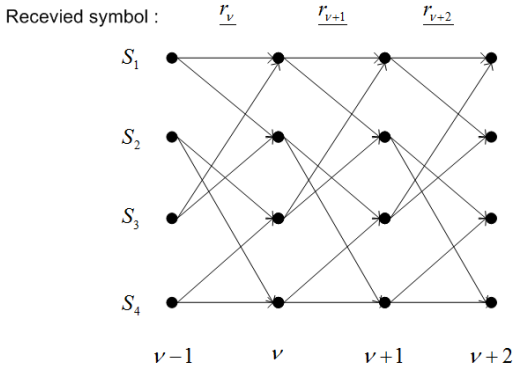


Figure 5. Trellis diagram of [2,1,3] Convolutional code

Viterbi algorithm receives the signal $r=(r_1,r_2,\Lambda\ ,r_v,\Lambda\ )$ which is delivered through the channel and finds the optimum path for all states s_i at every time duration v. After finding the optimum path for every received symbol, it determines an information bits through the processing of trace back. A complexity of Viterbi algorithm depends on

the number of states and the length of convolutional code symbols. It is important that each received symbol set can be operated independently. Therefore, if the received symbol set, for which the Viterbi decoding is to be performed, is implemented independently of the slave nodes, the processing time for decoding can be much reduced. In Figure 6, a procedure of MPI-based Viterbi decoding is illustrated.

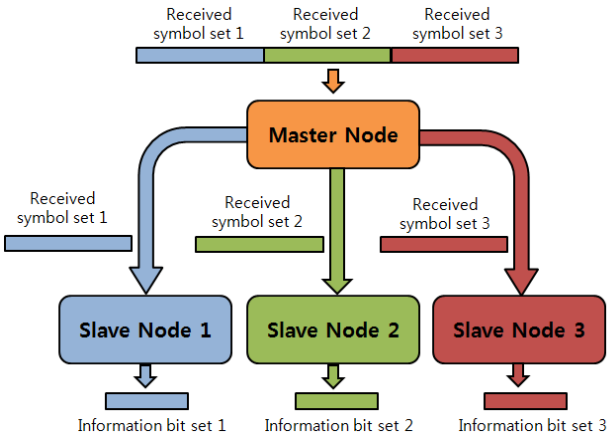


Figure 6. MPI-based Viterbi decoding

4.2.2. ML decoding

It is well know that ML decoding algorithm offers the optimum performance, but, it suffers from a tremendous computational load. To operate the ML decoding efficiently, the master node should first partition the received data into N pieces where N is the number of slave nodes and distribute each partition of the received data to each of the corresponding slave nodes. The total computational time is reduced by doing so because the amount of each data to be processed at each slave node is reduced. Each data are decoded in parallel at each of the slave nodes.

4.2.3. Synchronization

In WiMAX, the sampling rate is 10MHz and the frame length 5ms. The frame start can be found through a cross correlation between received samples and preamble sequences. Since the cross-correlation needs a lot of computations for 50,000 samples, the distributed processing using MPI should be adopted.

In Figure 7, the MPI–based synchronization algorithm is illustrated.

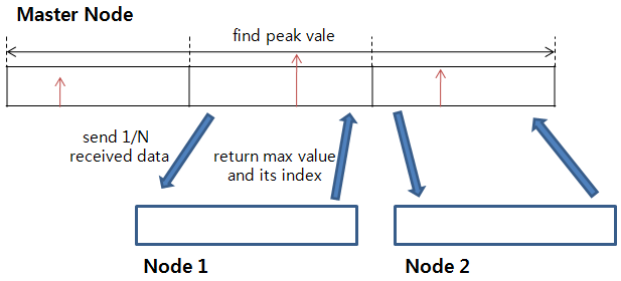


Figure 7. MPI-based Synchronization algorithm

First, the Master node divides the received signals into N pieces where N denotes the number of slave nodes to deliver each piece to each of the slave nodes. Each slave node calculates the cross correlation and delivers the master node the largest value and the location of it. Then, the master node determines the location of the frame start by comparing values of cross correlations delivered from slave nodes.

5. EXPERIMENTAL RESULTS

In this paper, we use 3 PCs as a cluster, which means that there are 2 slave nodes as shown in Figure 8.

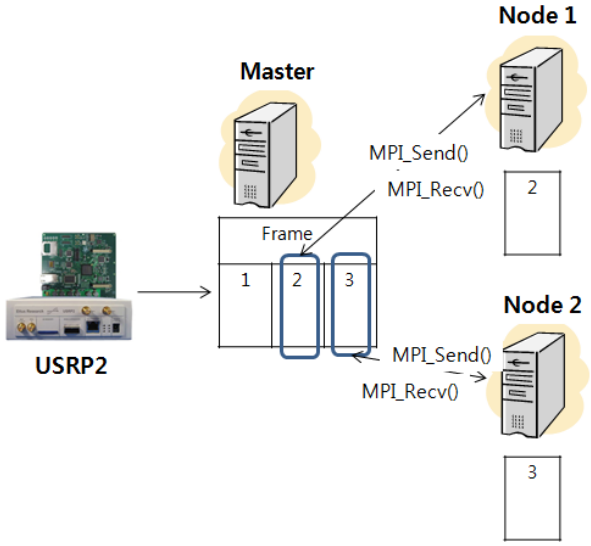


Figure 8. Inter-node data transmission using MPI

As shown in Figure 8, a PC as the master node, exchanges baseband signals with USRP2. It processes baseband signal processing blocks for which the computation time does not take much. The other two PCs, acting as slave nodes, perform the other signal processing parts for which the computation time takes relatively a lot in three steps using MPI: (1)to receive assigned data from the master node, (2)to process the signals, (3)to return the processed data to the

master node. Master node also participates in the baseband signal processing which takes an excessive amount of time together with the two slave nodes, which means that effectively three PCs simultaneously carry out the distributed work together.

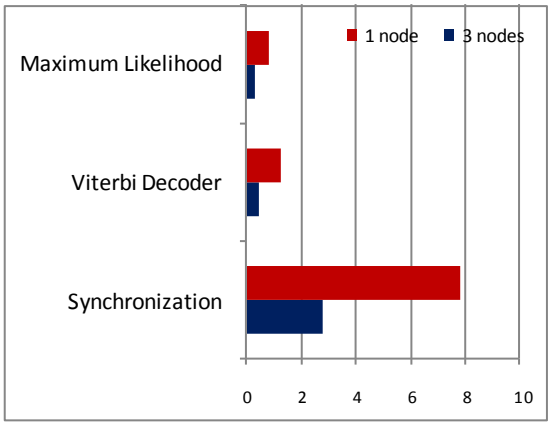


Figure 9. Algorithm processing time(ms) according to the number of nodes

Figure 9 illustrates the operation time of ML decoder, Viterbi decoder, and frame synchronization according to the number of nodes. As shown in the Figure 9, it takes much less time when 3 nodes perform together the distributed job compared to the case when only one node performs all processes.

Table 1. Processing time for 1 frame according to number of nodes

	1 node	3 nodes
Synchronization	7.8 ms	2.81 ms
Baseband signal processing	4.554 ms	1.715 ms

Table 1 shows the computation time according to the number of nodes required for processing one WiMAX frame. As shown in table 1, synchronization can be performed a lot faster when 3 nodes are activated by about 2.78 times. Also, we have found that the entire baseband signal processing is enhanced by about 2.65 times. The reason why the result of using 3 nodes is not improved exactly by 3 times is that the processing time includes the time of transferring data between the master node and each of the slave nodes. In particular, since data transfer between nodes for ML decoding and Viterbi decoding is needed twice, the baseband signal processing becomes less efficient compared to the case of synchronization. However, we expect the efficiency of distributed processing to be improved by increasing the number of nodes and by enhancing the network performance used for data exchange between nodes.

6. CONCLUSION

We have shown a drastic improvement in operation time by applying the MPI-based parallel/distributed processing to WiMAX system. SDR base station has been hailed as an appropriate technology to 4G environment which aims at a convergence of various kinds of communication standards. There is a limit, however, on the amount of operations that can be supported by a single base station due to physical constraints at each resource. To overcome that problem, we recommend applying the MPI-based distributed processing in a cluster environment to the SDR base station, which brings about much higher processing speed nearly proportionally to the number of slave nodes. Applying the MPI-based parallel/distributed processing technology to the SDR base station for 4G or even beyond-4G, we will be able to achieve extremely high operation speed.

7. REFERENCES

[1] The Wireless Innovation Forum http://www.sdrforum.org/

[2] Message Passing Interface Forum, “MPI: A Message-Passing Interface Standard version 2.2”, September 2009

[3] R.L. Graham, G.M. Shipman, B.W. Barrett, R.H. Castain, G. Bosilca, A. Lumsdaine, “Open MPI: A High-Performance, Heterogeneous MPI”, HeteroPar '06, September 2006, in Barcelona, Spain.

[4] A. Vishnu, G. Santhanaraman, W. Huang, H. Jin, and D.K. Panda, “Supporting MPI-2 one sided communication on multi-rail infiniband clusters: Design challenges and performance benefits”, in High Performance Computing – HiPC, 12th International Conference, Goa, India, December 18-21, Proceedings, 2005, pp. 137-147

[5] J. Ju, C. Ahn, J. Kim, S. Hyeon, S. Choi, “Implementation of an SDR Platform using GPU and its Application to 2x2 MIMO WiMAX System”, Proceedings of the SDR ’10 Technical Conference and Product Exposition, November 2010

[6] NVIDIA CUDA C Programming Guide Version 3.2 for Linux.

[7] M. Ergen, *Mobile Broadband - Including WiMAX and LTE*, Springer, NY, 2009

[8] J. Sanders, E.Kandrot, *CUDA by Example: An Introduction to General-Purpose GPU Programming*, Addison-Wesley, 2010.