IMPLEMENTATION OF SOFTWARE-BASED 2X2 MIMO LTE BASE STATION SYSTEM USING GPU

Seunghak Lee (HY-SDR Research Center, Hanyang Univ., Seoul, South Korea; invincible@dsplab.hanyang.ac.kr); Chiyoung Ahn (HY-SDR Research Center, Hanyang Univ., Seoul, South Korea; ahncy@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 demonstrate an implementation of a software-based 2X2 MIMO base station system for LTE mobile communications. The implemented base station system processes baseband signals on a Graphics Processor Unit (GPU). GPU is a high-speed parallel processor which provides very important advantage of using a very powerful C-based programming environment that is Compute Unified Device Architecture (CUDA). From our experimental tests of video stream data of LTE, we have verified that the GPU-based modem is surely capable of real-time processing of all the baseband signal processing algorithms required for LTE.

1. INTRODUCTION

Software defined radio (SDR) enables various wireless multiple access technologies and services to be provided on an open-architecture single hardware platform through software download [1]. In many conventional systems, SDR technology has been implemented using Digital Signal Processor (DSP) and/or Field Programmable Gate Array (FPGA). However, GPU is also attracted attention as SDR device for the implementation. Since GPU utilizes C-based CUDA, programming environment is easy, and its applications are mainly directed towards very high-speed floating-point parallel arithmetic operations [2]. Furthermore, libraries needed for controlling hardware are provided by NVIDIA, a GPU provider, which particularly means that we do not face any restriction on setting up the GPU hardware even when a software module is to be changed [3]. Due to these advantages of GPU, it is also used in the world mobile market to increase the performance of the smart phone.

Thus, GPU is very advantageous for SDR system implementation in which a flexible interface should be guaranteed in between function blocks. In fact, SDR-based platform with the flexible interface is keenly required especially in 4G mobile communications. On this potential, we adopt GPU as a modem platform of LTE in this paper. LTE, 4G mobile communication standard, is based on packet data transmission. LTE provides up to 1Gbps for downlink and 500Mbps for uplink with up to 20MHz bandwidth [4]. In this paper, in order to realize the software-based SDR system, we implement a 2X2 MIMO LTE base station system using GPU. In section 2, we introduce the structure of implemented system. Section 3 explains the detailed implementation of 2X2 MIMO LTE base station system using GPU as its modem platform and USRP2 as its RF transceiver. Section 3 also includes the parallelization of signal processing algorithms required for implementing LTE. Section 4 deals with the system performance while Section 5 concludes this paper.

2. STRUCTURE OF IMPLEMENTED SYSTEM

The parallelization of each signal processing algorithm is significant to maximize the performance of high-speed parallel arithmetic operations. It is a procedure of initially distributing the operations that involve mutually independent data, and later allocating each operation using the independent data to a corresponding thread [2]. In this paper, we present an implementation of a GPU-based LTE MIMO system through the parallelization of the signal processing algorithm.

This section introduces the structure of 2X2 MIMO base station system that has been implemented using GPU. Figure 1 illustrates a block diagram of implemented base station system. All baseband signal processing required for LTE is processed in GPU. In our implementation, we consider downlink only. Note that the transmit and receive part of LTE downlink are shown at the upper and lower part of Figure 1, respectively.

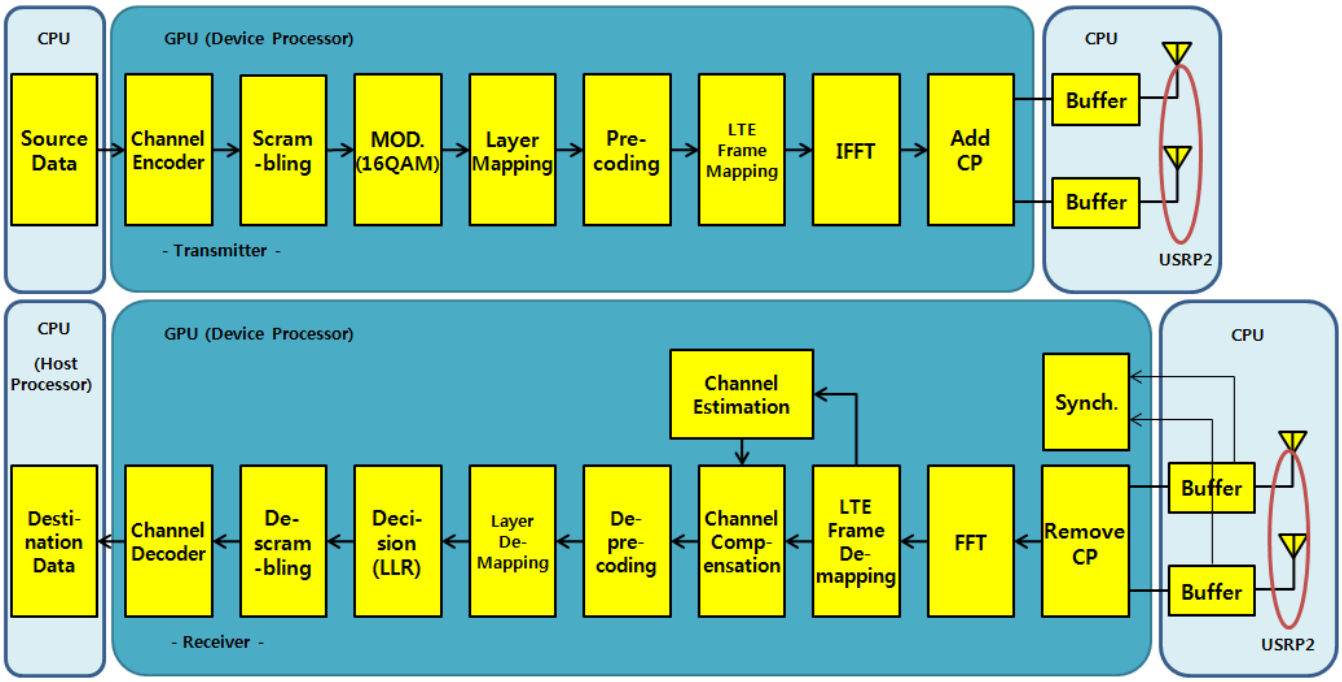


Figure 1. Block diagram of 2X2 MIMO LTE Baseband

The implemented system adopts Central Processing Unit (CPU) as its host processor and GPU as its device processor. Baseband signal processing required in LTE is provided by the device processor while the interface control of video data between RF transceiver and baseband modem is provided by the host processor. Figure 1 clearly shows which parts are processed by the device processor and which parts are done by the host processor.

We first consider the procedure of encoding the transmit video data at the base station site, which is shown at the upper part of Figure 1. Firstly, video data should be copied into the device memory to be processed in GPU from the CPU memory. Then, the video data are stored in the host memory buffer after channel coding and other necessary signal processing required in LTE. The video data which have been encoded according to the LTE standard are modulated in USRP2 and transmitted through a wireless channel as shown at the right-hand side of upper part of Figure 1.

Now, we consider the procedure of decoding the received video data at the handset site, which is shown at the lower part of Figure 1. The received data are processed in GPU using synchronization and tracking algorithm. Figure 2 illustrates the frame structure of LTE downlink implemented in our system. Horizontal and vertical axes denote time and frequency, respectively.

As shown in Figure 2, each frame consists of 10 sub-frames. Since there are 14 Orthogonal Frequency Division Multiplexing (OFDM) symbols assigned at each sub-frame,

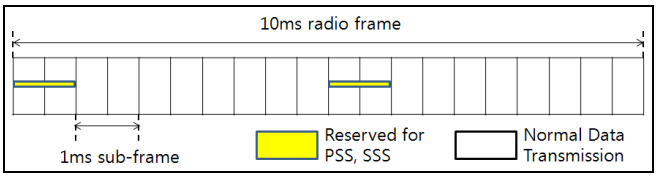


Figure 2. LTE Downlink frame structure

there are 140 OFDM symbols occupied at each frame. It is shown in Figure 2 that the 1st and 6th sub-frame contain Primary Synchronization Signal (PSS) and Secondary Synchronization Signal (SSS) needed for synchronization [5] as well as normal data symbols. More details about PSS and SSS are shown later in Section 3.2.

3. IMPLEMENTATION OF LTE SYSTEM USING GPU

One of the most important key issues in implementing a base station system using GPU is to parallelize the baseband signal processing algorithms in accordance with GPU's Single Instruction Multiple Data (SIMD) architecture [6]. Thus, we first represent each block shown in Figure 1 in terms of function such that each function block can be processed through corresponding signal processing algorithm using as many threads as possible inside GPU. Amongst the function blocks shown in Figure 1, some function blocks, to which the parallelization can be applied especially efficiently, are discussed in this section.

3.1 Scrambling

In 3GPP LTE, in order to avoid burst error that can happen during the procedure of data transmission, bit scrambling had been adopted. The bit scrambling includes modulo-2 operation between 31-bit long Pseudo-random sequence and transmit sequence [5] as follows.

$$\tilde{b}(i) = (b(i) + c(i)) \bmod 2 \qquad (1)$$

As shown in (1), transmit bit, $b(i)$, and scrambling bit, $c(i)$, are processed. The operation result $\tilde{b}(i)$ is processed for 16QAM modulation. Note that the operation shown in (1) is performed for every sub-frame. In our implementation, we assign 10 threads for the 10 sub-frames such that 10 operations shown in (1) can be performed with a single instruction.

3.2 Synchronization signals (PSS / SSS)

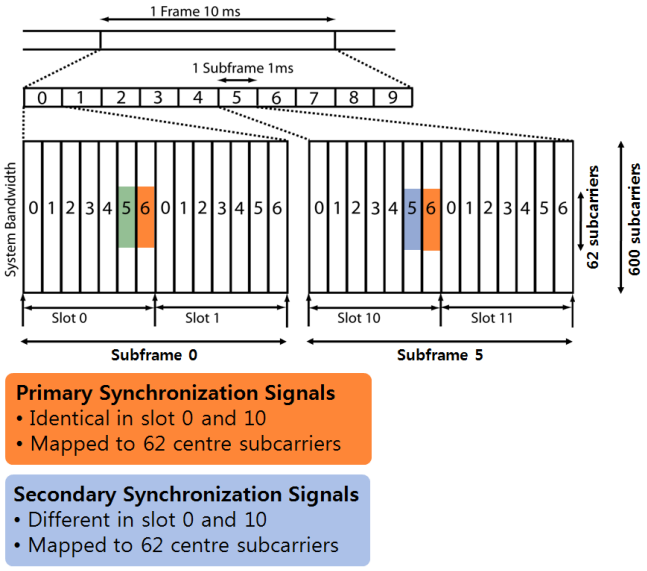


Figure 3. LTE Downlink Synchronization signals

In order to decode the received data captured from a wireless channel, we first have to perform a frame synchronization. PSS and SSS have been defined for such objective in 3GPP LTE. Figure 3 illustrates how the PSS and SSS are mapped at each sub-frame. As shown in the figure, PSS is mapped into 7th OFDM symbol of the 1st and 6th sub-frame, while SSS is mapped into the previous symbol to that.

Note that PSS detects the half-frame timing while SSS detects the cell ID and frame boundary. It is also noteworthy that PSS and SSS occupy just 1,080kHz out of the entire symbol frequency band, i.e., 10MHz in our implementation. It particularly means that the other frequency band is occupied by the normal transmit data. We must be very careful that the normal transmit data allocated near the synchronization signal act as interferences to the synchronization operation. In order to minimize the interference effect from the transmit data, synchronization algorithm that utilizes the partial correlation has been adopted [7].

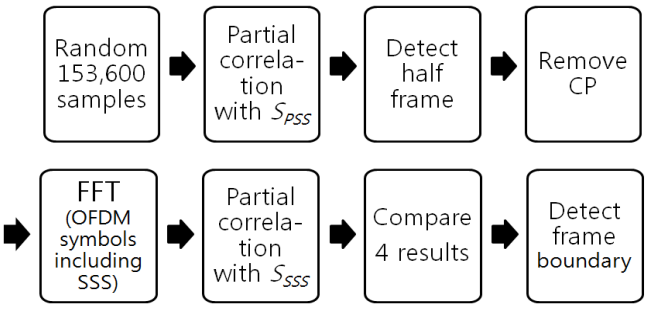


Figure 4. Block diagram of LTE Synchronization Algorithm

Figure 4 illustrates a block diagram of LTE synchronization algorithm that uses partial correlation. As shown in the figure, signal is sampled for the first 10ms with a sampling rate of 15.36MHz. Then, since the 153,600 sampled data correspond to a single radio frame length, the starting point of radio frame must be included in that 153,600 samples. At receiver, received data are correlated with the signal obtained by taking 1024-point FFT of the PSS.

Np	1	2	4	8
Probability (%)	81.43	93.71	90.08	83.37
Processing time (ms)	21.1	19.4	17.9	16.2

Table 1. Probability / Processing time of PSS

Np2	1	2	4	8
Probability (%)	99.54	99.88	99.82	99.25
Processing time (μs)	55	57	58	59

Table 2. Probability / Processing time of SSS

Table 1 and 2 show the probability of finding the frame start and processing time required for the correlation when the received signal is divided into 1, 2, 4, and 8 parts for PSS and SSS, respectively.

In the case of PSS, the success probability increases as the PSS sequence is divided into 2. The reason is that the number of transmit data which react as interferences to the synchronization is reduced as the PSS sequence is divided into 2 parts. However, as the PSS sequence is divided into more than 2 pieces, the success probability is rather worsened. The reason is that, as the PSS sequence is divided

into too many pieces, the correlation characteristic is degraded due to too small number of correlation data. In Table 1, it can also be observed that the operation time is saved as the sequence is divided more. In this case, however, success probability is a lot more important than saving the processing time because the frame start is to be found only once at the very beginning stage. Consequently, we have selected Np=2 for partial correlation of the PSS sequence.

Note that PSS provides just the half-frame start because PSS sequence mapped into 1st and 6th sub-frame are identical. On the contrary, the SSS sequence mapped into 1st and 6th sub-frame are distinct to each other, the boundary detection of radio frame can be found using SSS.

For the SSS detection, we use the FFT of received signal that finds the half-frame start through PSS. The radio frame start can be found using the SSS sequence of which the length is 62-symbol period because the SSS sequence is mapped, as a result of FFT, into the 62 resource elements not spread into the entire frequency band, i.e., 10MHz in our implementation.

As shown in Table 2, to divide the SSS sequence into 2, 4, or 8 pieces does not affect the performance of SSS detection. Nevertheless, in our implementation, we have selected Np2 = 2 for the partial correlation because it provides relatively better success probability compared to the other choices although the difference is not very conspicuous.

3.3 Channel Estimation

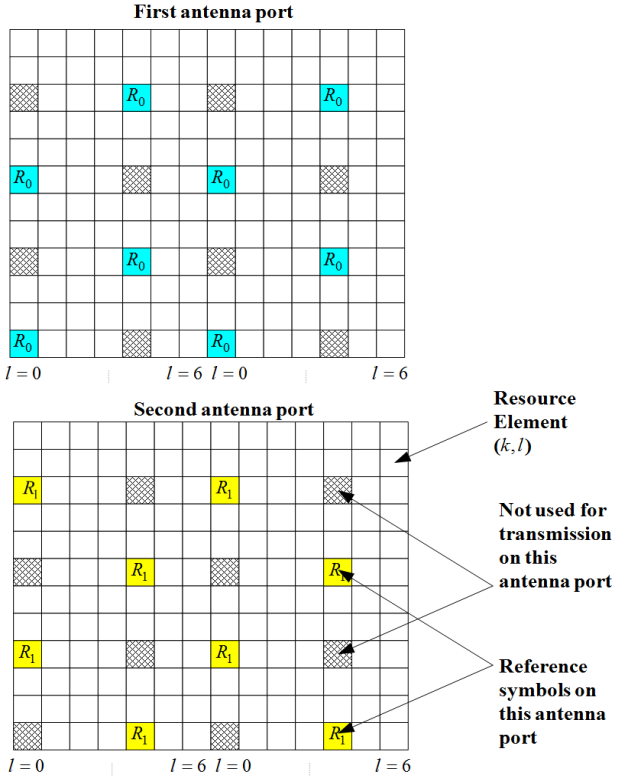


Figure 5. Mapping of downlink reference signals

This section introduces the parallelization technique for the channel estimation. We have implemented the channel estimation blocks in such a way that all the channel of entire radio frame can be estimated by applying the channel estimation algorithm to 2 resource blocks while there are totally 1,000 resource blocks at each radio frame. Figure 5 illustrates the structure of resource blocks of LTE downlink. Since we consider 2X2 MIMO system, the data structure for the first antenna and second antenna are shown at the upper and lower part of Figure 5, respectively. As shown in the figure, there are 4 reference signals at a single resource block. In our implementation, we have adopted 2-dimensional linear interpolation algorithm [2] using 8 reference signals existing in 2 resource blocks.

For the signals received at the first antenna, linear interpolation is performed along the time axis using the reference signals, denoted as "R_0" as shown at the upper part of Figure 5. Similarly, for the signals received at the second antenna, linear interpolation is performed using the reference signals denoted as "R_1".

After the interpolation along the time axis, exactly the same procedure is performed along the frequency axis using the channel estimation obtained by the time-axis interpolation.

The channel estimation procedure based on the 2 reference resource blocks as described above is performed simultaneously through 500 parallel operations which consists of 10 CUDA Blocks along the horizontal axis and 50 CUDA Blocks along the vertical axis.

4. PERFORMANCE EVALUATION

Figure 6 illustrates the software-based LTE base station that has been implemented in our lab using GPU as a modem platform. The implemented system is a 2X2 MIMO base station system adopting USRP2 as its RF transceiver and GPU as its modem. Note that a normal PC itself that is equipped with a GPU as its graphic card is used as base station and handset as shown in Figure 6. The PC and USRP2 are connected via Ethernet.

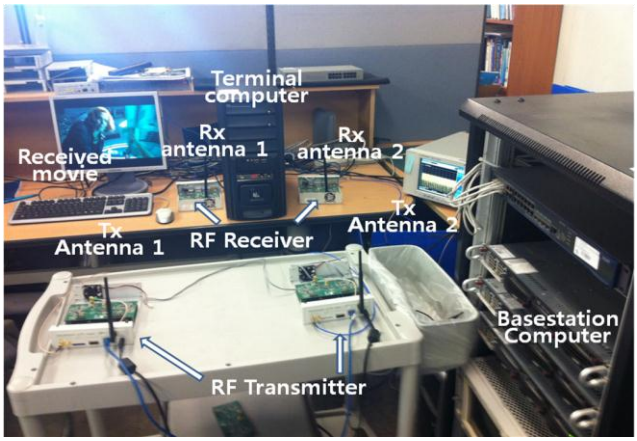
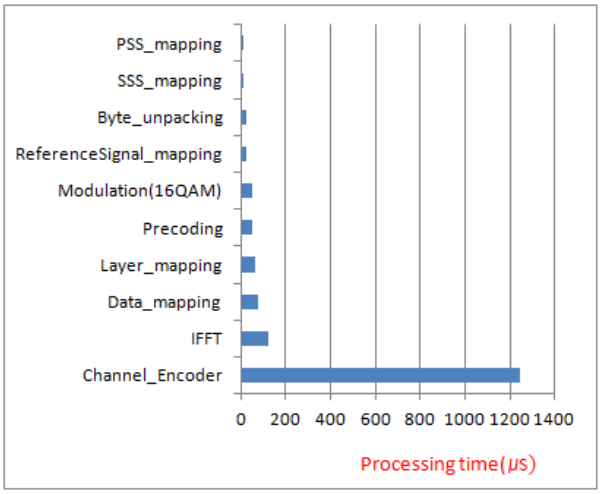


Figure 6. 2x2 SM MIMO LTE system

Performance evaluation can be summarized as follows. Video stream data are transmitted from the transmitting antennas through the RF transceiver after a proper encoding according to the LTE standard at GPU of base station computer shown at the right-hand side of Figure 6.
The receive antennas capture the wireless signals and RF transceiver passes the captured signal to the GPU of the terminal computer such that GPU modem decodes the received data into corresponding video stream. The decoded data are sent to the monitor of the receive computer to show the real-time monitoring of the video stream.
We have used the profiler provided by the GPU manufacturer in order to measure the processing time taken at each physical layer component. Figure 7 illustrates the operation time taken at each of the physical layer components measured by the profiler.

a) Transmitter



b) Receiver

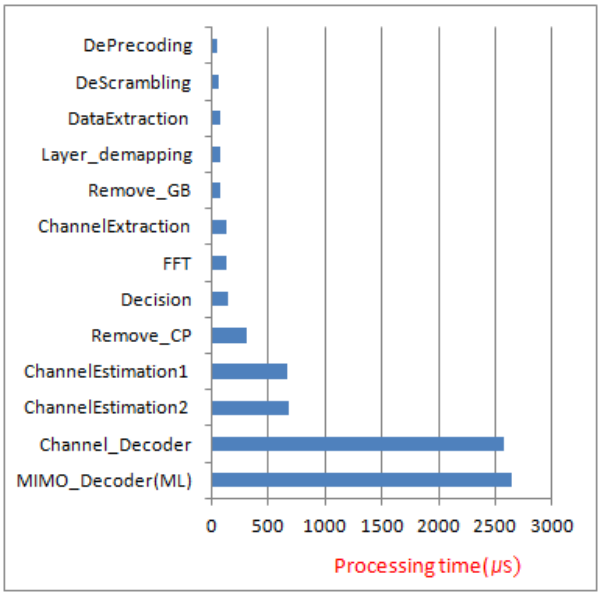


Figure 7. Computation time of LTE system

Figure 7 summarizes the processing time of each function at both transmit and receive. As shown in the figure, most operations are performed within 0.1ms except Channel Encoder / Decoder, channel estimation and MIMO decoder which are relatively more complicated. In the transmitter, channel encoder takes the longest time, 1.27ms. The entire processing time for generating transmit data frame has been found to be 4.20ms. In the receiver, MIMO decoder and channel decoder have taken 2.64ms and 2.58ms, respectively, while channel estimation takes about 1.37ms.
Channel Encoder / Decoder take much longer operation time than other function blocks because of its unique algorithm [2]. The algorithm divides data into several slots and operates them sequentially in each slot. The processing of each slot is independent so that we take advantage of parallel operations of GPU to implement these blocks. However, operations in each slot cannot be parallelized because they are sequential algorithm.

	GPU Processing time
Transmitter	4199.332 μs
Receiver	7617.342 μs

Table 3. GPU Processing time for 1frame

Table 3 shows the processing time required for the implemented system to encode and decode a single radio frame. As shown in the table, the processing time for the transmitter and receiver turned out to be 4,199.332 μs and 7,617.342 μs, respectively. Recalling that the frame time of LTE is 10ms, we can conclude that the implemented system can process the video stream data in real-time with a reasonable margin.

5. CONCLUSION

We have implemented a 2X2 MIMO LTE base station system and corresponding terminal system using GPU modem. Using the implemented systems, we performed downlink of LTE video stream data to verify the real-time processing capability. Signal processing algorithms required for implementing LTE data have been parallelized as much as possible in order to minimize the processing time for the real-time processing. We have measured the total processing time for both transmit and received data to find that it takes about 4.20ms and 7.62ms for transmit and receive, respectively, which confirms a good real-time capability for LTE data of which the frame length is 10ms. From the various tests using the implemented system, we conclude that the GPU modem is appropriate for the software-based base station system while the terminal modem should resolve the problem of power consumption which has not been considered in our work.

6. REFERENCES

[1] W. Tuttlebee, *The Software Defined Radio: Enabling Technologies*, John Wiley & Sons, 2002.

[2] Jaehyuk Ju, Chiyoung Ahn, June Kim, Seungwon Choi, "Implementation of an SDR platform using GPU and its Application to 2x2 MIMO WiMAX System," Wireless Innovation Forum, 2010

[3] NVIDIA Corporation, *CUDA Programming Guide, NVIDIACorporation*, Apr. 2009.

[4] Anthony Lo, Ignas Niemegeers, "Multi-hop Relay Architectures for 3GPP LTE-Advanced," IEEE 9th Malaysia International Conference on Communications, 2009

[5] 3G Generation Partnership Project(3GPP); Technical Specification Group Radio Access Network; **Evolved** Universal Terrestrial Radio Access (E-UTRA); Physical Channels and Modulation(Release 9), http://3gpp.org/ftp/specs/html-info/36211.htm

[6] JD Owens, M Houston, D Luebke, S Green, JE Stone, JC Phillips, "GPU Computing", Proceedings of the IEEE, vol. 96, May 2008.

[7] Jung-In Kim, Jung-Su Han, Hee-Jin Roh, and Hyung-Jin Choi, "SSS Detection Method for Initial Cell Search in 3GPP LTE FDD/TDD Dual Mode Receiver," Communications and Information Technology, ISCIT, 2009.