논문 2011-48IE-3-4

# WiBro 환경에서 SDR을 위한 GPU 시스템 구현

## ( Implementation of GPU System for SDR in WiBro Environment )

안 성 수*, 이 정 석**

( Sungsoo Ahn and Jungsuk Lee )

요 약

본 논문은 와이브로 통신환경에서 SDR(Software Defined Radio) 시스템을 위한 실행속도 증진 방법을 개발하였다. 본 논문에서는 SDR 기능 구현을 위해 GPU(Graphics Processing Unit)라는 새로운 프로세서를 사용하였다. 일반적으로 통신시스템에서는 DSP(Digital Signalling Processor)나 FPGA(Field Programmable Gate Array)를 이용하여 시스템을 구현한다. 그러나 이러한 프로세서는 장단점이 커서 구현 및 디버깅을 하기 어렵다. GPU는 다수의 프로세서로 구성되어 있어 벡터 처리에 적합하며, 각 프로세서는 thread의 셋으로 구성이 되어 있다. 본 논문에서는 GPU만의 자원뿐만 아니라 CPU 자원 까지 사용하기 위한 Framework 또한 구현하였다. 다양한 실험결과, 본 제안 시스템이 와이브로 환경에서 우수한 성능을 제공함을 확인할 수 있었다.

Abstract

We developed a method of accelerating the operation speed of communication systems for SDR(Software Defined Radio) systems in WiBro environment. In this paper, we propose a new scheme of using GPU(Graphics Processing Unit) for implementing the communication system which perform with the functionality of SDR. In general, communication systems is made by DSP(Digital Signalling Processor) or FPGA(Field Programmable Gate Array). However, in this case, there are exist the problem of implementation and debugging caused by each CPU characteristic. The GPU is optimized for vector processing because it usually consists of multiple processors and each processor in GPU is composed of a set of threads. We also developed Framework to use GPU and CPU resources effectively for reducing the operation time. From the various simulation, it is confirmed that GPU system have good performance in WiBro system.

**Keywords :** GPU, SDR, WiBro, BER, Decoder

## Ⅰ. 서 론

통신 시스템에서는 실시간(Real time) 프로세싱이 가장 중요한 부분 중에 하나이다. 특히 알고리즘이 복잡해지면서 빠른 연산 이라는 것이 큰 화두가 되고 있다. 아무리 좋은 알고리즘이라도 실시간 프로세싱이 불가능하면 통신 시스템에서는 쓸모가 없는 것이 되어 버린다. 그래서 GPU(Graphics Processing Unit) 라는 새로운 프로세서로 어떠한 알고리즘이라도 실시간이 가능하게 하는 시스템을 생각 하게 되었다. GPU 는 3D 연산에 특화된 프로세서이다. 특히 최근 연산량이 점점 늘어나고 있는 3D 게임으로 인해 기하급수 적으로 성능이 개선되고 있다. 최근 SDR을 구현 하는 하드웨어로는 DSP(Digital Signalling Processor) 및 FPGA(Field Programmable Gate Array)를 많이 사용 한다. 하지만 이들은 몇 가지 단점을 가지고 있다. DSP는 코딩 및 탄력적인 개발 환경을 제공하나 성능이 떨어진다는 단점을 가지고 있고 FPGA는 강력한 성능을 가지고 있으나 개발 환경 및 디버깅의 어려움을 가지고 있다. 본 논

* 정회원, 명지전문대학 정보통신과
(Dept. of Information Communication, Myongji College)
** 정회원, 인하공업전문대학 메카트로닉스과
(Dept. of Mechatronics, Inha Technical College)
접수일자: 2011년8월10일, 수정완료일: 2011년9월16일

문에서 적용한 GPU 는 DSP 와 FPGA 의 단점인 성능의 저하 및 개발의 어려움의 중간과정의 프로세서이다. GPU을 생산하는 회사 중 NVDIA 에서는 이 GPU를 이용할 수 있는 각종 툴 및 컴파일러를 제공을 하여서 누구나 손쉽게 코딩을 할 수 있도록 한다.

또한 WiBro 환경[1]에서 GPU 시스템을 활용한 운영시간 감소와 BER(Bit Error Rate)를 확인하도록 하였다. 본 논문의 Ⅱ장에서는 스칼라 및 벡터 채널 모델링을 작성하였고, Ⅲ장에서는 GPU 구조 및 GPU 시스템 구현방법을 언급하고 Ⅳ장에서는 구현된 시스템의 성능에 대해 설명을 한다. 마지막으로 Ⅴ장에서는 결론을 작성하였다.

## Ⅱ. 채널 모델링

수신 신호의 Fast fading에 대한 통계적인 모델은 많은 수의 scatter에 대한 물리적은 전파 환경에 따라 결정된다[2]. 송신 신호는

$$x(t) = s(t) \cdot e^{j(2\pi ft + \phi_0)} \tag{1}$$

이고, $s(t)$는 대역폭이 $B$인 복소 기저대역 신호, $f$는 반송파 주파수, $\phi_0$는 임의의 초기 위상이다. 만약 모바일이 $v$의 속도로 움직이고 있고 직접적인 LOS가 없다면 기지국에서 수신 신호는 모든 다중경로의 컴포넌트의 합이 된다.

$$y(t) = A\sum_{i=1}^{L} R_i s(t-\tau_i) e^{j2\pi[(f+f_d\cos\psi_i)t - f\tau_i]} \tag{2}$$

이때 $A$는 거리에 대한 손실과 안테나 이득의 영향을 포함한 값이 된다. $i$번째 다중경로 컴포넌트에 대해, $R_i^2$은 $i$번째 경로의 평균 전력이며, $\tau_i = r_i/c$는 $r_i$가 전파 거리이고 $c$가 빛의 속도라고 하면 다중 경로 지연 시간이 된다. $f_d\cos\psi_i$는 도플러 시프트(Doppler shift)로 $f_d = v/\lambda$는 최대 도플러 시프트(maximum Doppler shift), $\psi_i$는 $i$번째 스캐터의 방향이 된다. 이러한 파라미터는 모두 시간에 의존적이다.

다중경로 지연 스프레드를 다음과 같이 정의한다[3].

$$T = \max_i \tau_i - \max_i \tau_i \tag{3}$$

만약 $T$가 $B^{-1}$보다 훨씬 작다고 하면, 즉 $s(t)$가 협대역 신호일 때 $s(t-\tau_i) \approx s(t-\tau_0)$이다. 단 $\tau_0 \in [\min_i \tau_i, \min_i \tau_0]$이다. 그러면 수신 신호를 다음과 같이 표현할 수 있다.

$$y(t) \approx s(t-\tau_0) \cdot \left(\sum_{i=1}^{L} R_i e^{j\phi_i(t)}\right) \cdot e^{j2\pi ft} \tag{4}$$

여기서 $\phi_i(t) = 2\pi(f_d\cos\psi_i t - f\tau_i)$로 $[0, 2\pi]$ 사이의 값으로 균일한 분포의 랜덤변수로 생성이 된다.

그러므로 equivalent lowpass 수신 신호는 식 (5)와 같다.

$$\tilde{y}(t) \approx s(t-\tau_0) \cdot \left(\sum_{i=1}^{L} R_i e^{j\phi_i(t)}\right) \tag{5}$$

기지국에서 수신한 저역 통과 신호는 단말기와 기지국 사이의 거리에 따른 시간 지연과 괄호 안 복소값에 의해 송신 신호가 왜곡된다. 식 (5)의 괄호 안을 다시 정의하자.

$$\beta(t) = \sum_{i=1}^{L} R_i e^{j\phi_i(t)} = \alpha(t) e^{j\phi(t)} \tag{6}$$

기지국에서의 저역 통과 수신 신호인 $\tilde{y}(t)$는 저역 통과 채널인 $\beta(t)$와 저역 통과 신호인 $s(t)$의 응답으로 표현된다. 일반적인 저역 통과 채널은 다음과 같이 표현 된다.

$$h(t;\tau) = \delta(\tau - \tau_0)\alpha(t) e^{j\phi(t)} \tag{7}$$

그림 1은 채널 응답을 주파수와 시간 영역에서 표현한 그림이다. x축은 시간 축, y축은 주파수 축이며 z축

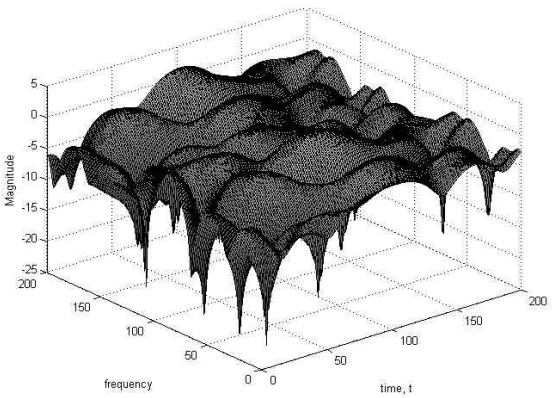


그림 1. 다중 경로 (8 path) 페이딩 환경에서의 채널 응답

Fig. 1. Channel response of multipath fading environment(8 path).

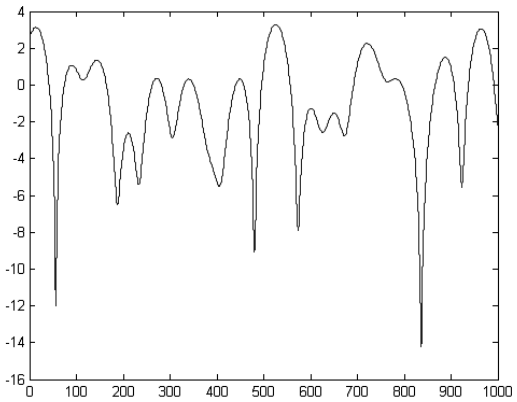


그림 2. 페이딩 채널의 크기: $f_d$=119Hz, $T_s$=66.67$\mu$s
Fig. 2. Magnitude of fading channel: $f_d$=119Hz, $T_s$ =66.67$\mu$s

은 채널의 크기를 나타낸다. 그림에서 볼 수 있듯이 주파수와 시간 영역에서 변화가 매우 다양하다.

또한 그림 2는 페이딩 채널의 크기를 표현하였다. 시간이 변함에 따라 채널 임펄스의 크기가 변하는 것을 알 수 있다. 이러한 변화는 도플러 주파수가 높을수록 빠르게 나타난다[4].

## Ⅲ. GPU 구조 및 구현

### 1. GPU의 하드웨어 구조

GPU 통신 시스템은 다음과 같은 구조를 가진다. 그림 3은 GPU 시스템의 전체 구조를 나타낸다.

그림 3에서 보여 지듯이 GPU system은 크게 GPU 통신 시스템은 다음과 같은 구조를 가진다. 그림 3은 전체 구조를 나타낸다. 그림 3에서 보여 지듯이 GPU system 은 크게 3부분으로 나눌 수 있다. 하나는 무선에서 송수신 되는 데이터를 AD / DA를 하는 RF

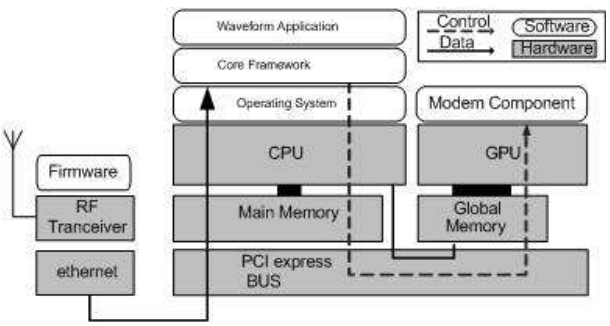


그림 3. GPU System 구조
Fig. 3. Structure of GPU System.

Transceiver, CPU 와 GPU 간의 데이터 통신이 가능한 인터페이스를 제공하는 Framework 마지막으로 Modem 프로그램을 하여 실행이 되는 GPU 부분이다. 이 세 가지가 하나로 묶여 원활한 데이터 송수신이 가능하게 한다. GPU와 CPU간의 데이터 통신은 PCI Express를 통해 이루어지며 DMA(Direct Memory Access)를 통해서 전송이 된다. 그리고 RF Transceiver와 컴퓨터 사이는 100Mbps Ethernet을 통해 이루어진다.

### 2. GPU의 소프트웨어 구조

그림 4에서 보여 지듯이 GPU에서 실행되는 코드는 WiBro[5] 모뎀 코드이다. 크게 Encoder와 Decoder 부분으로 나누어져 구현이 되어있다. Encoder부분은 Randomization, Convolutional Encoder, Permutation, IFFT로 구성이 되어 있으며 Decoder부분은 FFT, Channel Estimation, De-permutation, Compensation, Viterbi Decoder 부분으로 구성이 되어 있다. 각각의 모듈들은 GPU의 kernel과 CUDA에서 제공하는 함수를 이용하였다. 특히 FFT, IFFT는 CUDA에서 제공하는 cufft library를 사용하여 구현하였다.

Encoder와 Decoder에서 만들어지는 데이터는 CPU로 메모리 카피가 이루어 져야 한다. 이는 CUDA에서 제공하는 함수인 cudaMemcpy를 통해서 CPU와 GPU 간의 메모리 접근이 가능하게 한다. GPU 양쪽의 CPU는 소켓 및 RF 모듈의 인터페이스 부분이다.

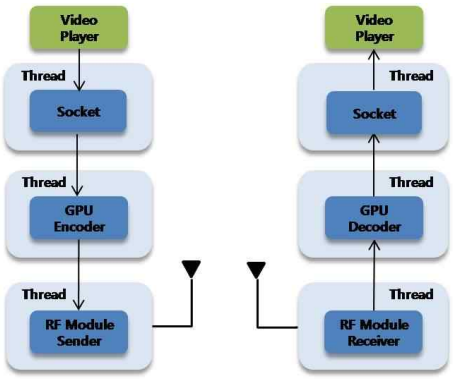


그림 4. GPU System의 소프트웨어 구조
Fig. 4. Software structure of GPU System.

### 3. Encoder 구현

Encoder는 크게 3개의 component 로 구성이 되어 있다. 그림 4에서도 볼 수 있듯이 FEC, Permutation, IFFT 로 구성이 되어 있다. Permutation은 parallel하게

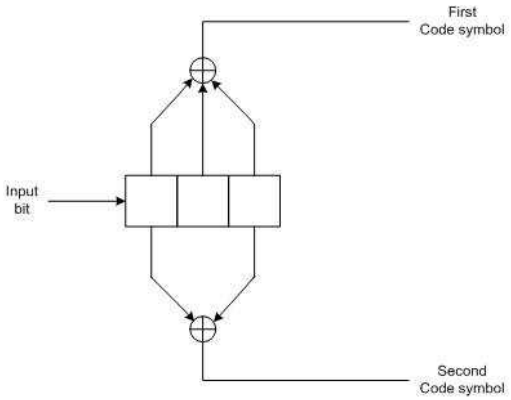


그림 5. Convolutional Encoder 구조
Fig. 5. Structure of convolutional Encoder.

알고리즘을 처리가 가능 한 구조이나 FEC는 그러지 않다. 본 논문에서는 FEC중 많이 사용되는 Convolutional code를 사용 하였다[6~7].

### 4. Viterbi Decoder 구현

Convolutional code를 디코딩하는 알고리즘은 여러 가지가 있지만 그 중에서도 MLSE(minimize the sequence error probability) 디코딩 방법인 viterbi 알고리즘이 가장 널리 사용된다. Viterbi 알고리즘은 그림 6과 같이 세로축이 스테이트를 나타내고 가로축이 시간을 나타내는 Trellis diagram으로 설명이 용이하다.

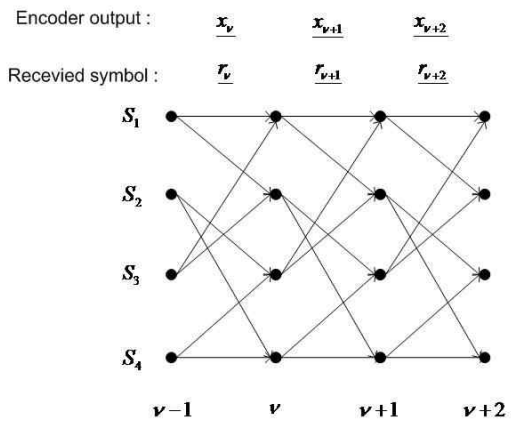


그림 6. Convolutional code의 Trellis 다이어그램
Fig. 6. Trellis diagram of Convolutional code.

## Ⅳ. 성능분석

GPU는 NVIDIA에서 만든 9800 GTX+를 이용하여 구현하였다[8]. 그리고 성능의 비교를 위해 사용한 CPU는 Intel에서 만든 Pentium 4 3.2 GHz를 사용하여 비교하였다. 시스템 파라미터는 다음과 같다.

(1). Modulation : QPSK
(2). FFT Size : 1024 points
(3). Coding Rate : 1/2
(4). Channel Estimation : Linear Interpolation

위와 같은 시스템 파라미터를 가지고 그림 7과 같은 GPU 모뎀 시스템을 구현 하였다.

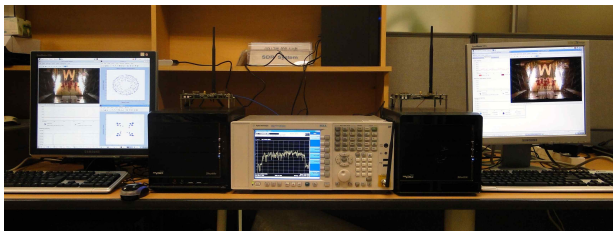

그림 7. 구현된 시스템
Fig. 7. Implementation system.

그림 7은 구현된 시스템의 사진이다. 오른쪽에 있는 컴퓨터는 Encoder이며 왼쪽의 컴퓨터는 Decoder이다. 구현된 시스템은 무선으로 통신을 할 수 있도록 구현이 되어 있다.

### 1. 시간성능

시간 성능을 확인하기 위해 NVIDIA에서 제공하는 profiler를 이용하였다. 그림 8과 9는 CPU와 GPU상에서의 시간 성능을 분석한 그래프이다.

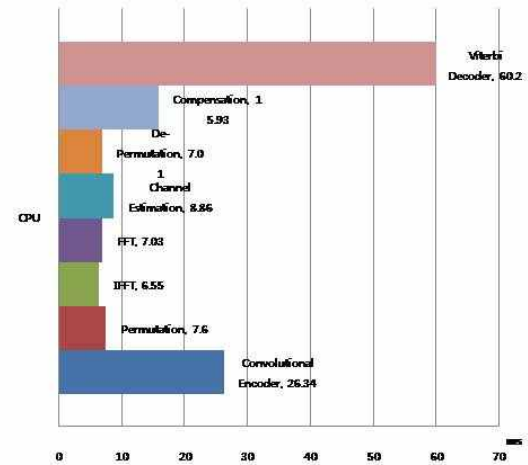


그림 8. GPU 상에서 component 들의 시간 성능
Fig. 8. Time performance of GPU component.

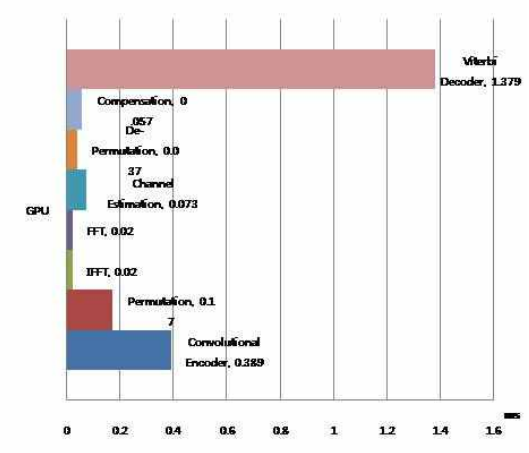


그림 9. CPU 상에서 component 들의 시간 성능
Fig. 9. Time performance of CPU component.

그림 8과 9에서 보여 지듯이 GPU와 CPU에서의 성능은 많은 차이를 보인다. 각각의 component는 CPU(1,423.7)에 비해 GPU(131.5)가 약 10배 정도의 성능의 이득을 보인다.

그리고 만일 병렬화 구현이 어려운 Viterbi Decoder와 Convolutional Code의 완벽한 병렬화를 이룰 수 있다면 더욱 많은 시간적 성능의 이득을 이룰 수 있다.

### 2. BER(Bit Error Rate) 성능

그림 10은 GPU 시스템의 성능을 컴퓨터 시뮬레이션과 비교를 한 그래프이다. 컴퓨터 시뮬레이션은 MATLAB을 이용하여 확인 하였다.

그림 10에서도 볼 수 있듯이 시뮬레이션과 GPU 시스템간의 성능은 비슷하게 나온다. 그러므로 GPU 시스

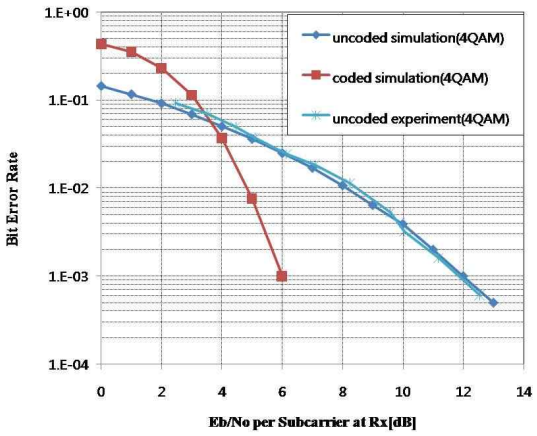


그림 10. GPU 시스템과 시뮬레이션의 BER 성능 비교
Fig. 10. Comparison of BER performance of GPU system and simulation.

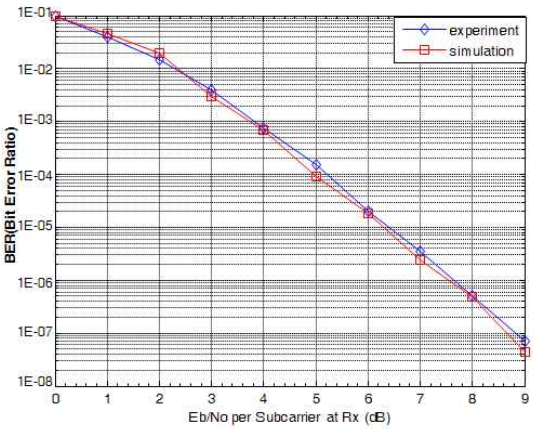


그림 11. Viterbi Decoder 에서의 성능
Fig. 11. Performance of Viterbi Decoder.

템의 성능은 우수하다고 볼 수 있다. 그리고 그림 11은 실제 GPU 상에서 Viterbi Decoder의 성능을 나타내는 그래프이다.

그림 11은 Viterbi Decoder의 Bit Error Rate 그래프이다. 이 그래프에서도 볼 수 있듯이 시뮬레이션 값과 실제GPU에서 실행한 값이 비슷하다는 것을 볼 수 있다.

## V. 결 론

기존 시스템인 HDR(Hardware Defined Radio)은 성능은 우수하나 유지보수 및 업그레이드의 어려움으로 최근 SDR을 이용한 기술들이 많이 발전되어 지고 사용되고 있어, 본 논문에서는 SDR의 장점을 살리면서 우수한 통신 시스템을 구현하기 위해 GPU를 이용한 통신 시스템을 구현하였다. 따라서 본 논문에서 모뎀 시스템을 구현하기 위해 많이 사용하는 DSP, FPGA가 아닌 새로운 프로세서인 GPU로 구현함으로써 컴퓨터에서도 모뎀을 구현 할 수 있다는 것을 보여주었다. WiBro 환경에서 GPU 시스템을 구현한 결과 우수한 성능을 제시함을 확인할 수 있었다.

## 참 고 문 헌

[1] 우상희, “2.3-2.7GHz WiMAX용 TDD 중계기의 송숫긴 안테나 제어를 위한 동기신호 생성 모듈 설계 및 구현”, 제46권, TC편, 제 1호, pp.60~63, 2009.
[2] IEEE Standards Activities Department, IEEEP 802.16 Rev2 / D2 ,DRAFT Standard for Local and metropolitan area networks, AirInterface for

Broadband Wireless Access Systems, December 2007.
[3] L.HANZO 외, OFDM and MC-CDMA for Broadband Multi-User Communications, WLANs and Broadcasting, Wiley, 2003.
[4] Alan V.Oppenheim, Ronald W.Schafer, DISCRETE - TIME SIGNAL PROCESSING Second Edition, PrenticeHall, 1999.
[5] Timothy M. Schmidl and Donald C. Cox, "Robust Frequency and Timing Synchronization for OFDM", IEEE Trans .Commun., Vol. 45, No. 12, December 1997.
[6] P. Moose, "A technique for orthogonal frequency division multiplexing frequency offset correction," IEEE Trans. Commun., vol.42 October 2006.
[7] Hyeong-Sook Park, "Design of synchronization in OFDMA/TDD based WiBro system", IEEE international symposium on personal ,Indoor and Mobile Radio Commun., 2007.
[8] NVIDIA CUDA Compute Unified Device Architecture Programming Guide, 2007.

## 저 자 소 개

안 성 수(정회원)
1987년 한양대학교 전자공학과 학사 졸업.
1990년 한양대학교 전자공학과 석사 졸업.
2001년 한양대학교 전자공학과 박사 졸업.
1990년~1997년 국방과학연구소 연구원
2002년~현재 명지전문대학 정보통신과 교수
<주관심분야 : 스마트 안테나, DSP 신호처리, 이동통신>

이 정 석(정회원)
1985년 2월 광운대학교 전기공학과 학사
1990년 8월 광운대학교 전기공학과 석사
2001년 8월 광운대학교 제어계측공학과 박사
1990년 10월~1997년 2월 국방과학연구소 항공전자실 선임연구원
2002년 9월~2004년 현재 인하공업전문대학 메카트로닉스과 부교수
<주관심 분야: 제어계측, 마이크로프로세서, RFID 응용>