한국지능시스템학회 논문지 2010, Vol. 20, No. 3, pp. 362-367

영상 감시 시스템을 위한 Nios II 임베디드 프로세서 시스템의 Linux 디바이스 드라이버 구현

An Implementation of Linux Device Drivers of Nios II Embedded Processor System for Image Surveillance System

김동진* · 정용배* · 김태효* · 박영석*

Dong-Jin Kim, Young-Bee Jung, Tae-Hyo Kim, Young-Seak Park

* 경남대학교 정보통신공학과

요 약

본 연구에서는 기존 CCTV 시스템의 고정되어 있는 감시지역과 카메라의 움직임을 수동으로 조작하는 단점을 보완 할 수 있는 영상 감시 시스템을 개발하기 위해 FPGA 기반 Nios II 임베디드 프로세서 시스템과 Linux 디바이스 드라이버를 구현하였다. Altera Nios II 프로세서 8.0부터 메모리를 안정되고 효율적으로 관리할 수 있는 MMU를 지원하고 있다. 각종 응용에 유연하고 적응성이 뛰어난 Altera Nios II 소프트코어 프로세서 시스템을 이용하여 영상감시 관제 하드웨어를 구성하였고, Linux 기반 Nios II 시스템의 카메라 디바이스 드라이버와 VGA 디바이스 드라이버를 구현함으로써 Nios II 시스템을 위한 영상 감시 시스템을 구현할 수 있었다.

키워드 : Nios II 임베디드 시스템, 리눅스, 메모리 관리 유닛, 디바이스 드라이버

Abstract

In this paper, we describe implementation of FPGA-based Nios II embedded processor system and linux device driver for image monitoring system which is supplement weakness for fixed surveillance area of existing CCTV system and by manual operation of the camera's moving. Altera Nios II processor 8.0 is supported MMU which is stable and efficient managed memory. We designed the image monitoring and control system by using Altera Nios II soft-core processor system which is flexible in various application and excellent adaptability. By implementation of camera device driver and VGA decvice driver for Linux-based Nios II system, we implemented image serveillance system for Nios II embedded processor system.

Key Words : Nios II Embedded System, Linux, Memory Management Unit, Device Driver

1. 서 론

사회가 발전하면서 각종 사고와 위험 속에 안전을 위해 하는 요소들이 많아졌다. 점점 지능화 되어가는 범죄, 위험지역에서의 사고, 보안지역 침입 등에 대처하기 위해 CCTV를 적극 활용하고 있으며, 극히 일부에서만 CCTV의 효과로 범인검거, 위험지역에서 구조, 보안 사고에 대한 대처가 될 뿐 CCTV 설치 수에 비교하면 효과가 상당히 낮은 편이다. 기존의 CCTV 시스템은 PC기반이며, 가격이 비싸다. 이러한 기존 CCTV 시스템의 단점을 보완하기 위해 더 많은 인력을 충원하거나 감시 카메라의 수를 늘리기 보다는 지능적으로 판단하고 추적할 수 있는 저가의 임베디드 시스템 기반 영상 감시 시스템의 필요성이 더욱 증대되고 있다[1-2].

본 논문에서는 PC기반 CCTV 시스템의 단점을 보완 할 수 있는 영상 감시 시스템을 위한 Nios II 임베디드 프로세서 시스템의 Linux 디바이스 드라이버를 구현하였다. Altera Nios II 프로세서 8.0부터 메모리를 안정되고 효율적으로 관리할 수 있는 MMU를 지원하고 있다. Altera Nios II 임베디드 시스템은 각종 응용에 유연하고 적응성이 뛰어나지만, 시스템의 설계 구현이 어렵다. 이러한 이유로 Altera Nios II 시스템을 위한 카메라 IP 및 디바이스 드라이버 설계 구현에 대한 연구가 국내·외적으로 거의 보고되고 있지 않다.

Altera Nios II 임베디드 시스템을 이용하여 영상감시 관제 하드웨어를 구성하였으며[3-6], 운영체제로는 저렴한 비용과 많은 장치 드라이버가 제공되며, 소스코드가 공개되어 있는 임베디드 리눅스를 사용하였다.

II장에서는 하드웨어 설계를 논의하고, III장에서는 디바이스 드라이버 설계 그리고 IV장에서는 실험 및 결과, 마지막으로 V장은 결론을 요약한다.

접수일자 : 2010년 4월 3일
완료일자 : 2010년 5월 17일
감사의 글 :
본 논문은 본 학회 2010년도 춘계학술대회에서 선정된 우수논문입니다.

2. 하드웨어 설계

하드웨어 설계의 주요 도구로는 Quartus II 설계 소프트웨어와 SOPC Builder 시스템 개발도구, Modelsim 시뮬레이션 소프트웨어 그리고 구현회로 검증을 위해서는 SignalTap II 임베디드 논리 해석기를 이용하였다[5-6]. 그림 1은 본 연구가 목적하는 시스템의 하드웨어 구성이다.

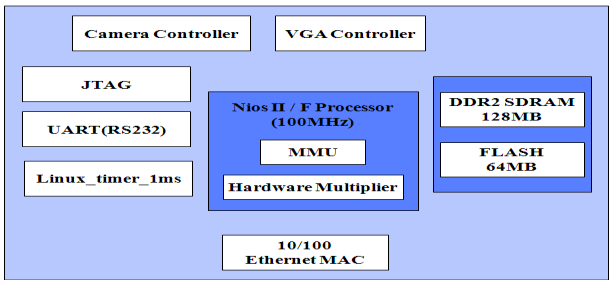

그림 1. 시스템 하드웨어 블록도
Fig. 1. Block diagram of system hardware

2.1 MMU를 포함한 Linux를 위한 하드웨어 최소 사항

Linux를 위한 최소 시스템으로 프로세서의 경우 Nios II f 코어와 MMU, Hardware multiplier가 요구된다. 특히 MMU 설정에서 512 또는 1KB의 On-Chip Memory가 필요하다. MMU를 포함한 Nios II CPU를 디자인하기 위해서 그림 2와 같이 On-Chip Memory를 추가한다[7-8].

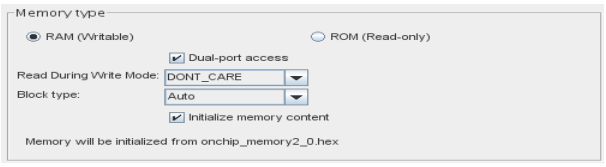

그림 2. On-Chip Memory 설정
Fig. 2. On-Chip Memory Configuration

그림 3은 CPU의 MMU 설정 옵션이다.

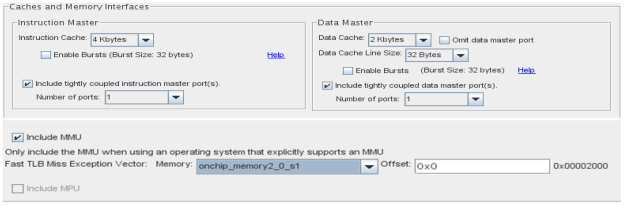

그림 3. MMU 설정
Fig. 3. MMU Configuration

그림 4와 같이 CPU와 On-Chip Memory를 연결한다. CPU와 MMU를 위한 On-Chip Memory의 Base Address 설정은 CPU의 jtag_debug_module은 0x07fff800, MMU를 위한 On-Chip Memory는 0x07fff400으로 설정한다[7-8].

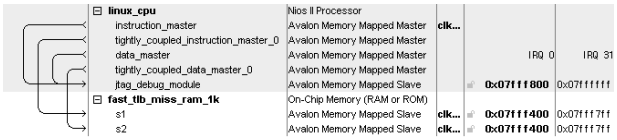

그림 4. Base Address 설정
Fig. 4. Base Address Configuration

사용자 IP나 다른 주변장치를 추가할 경우, 0x0~0x1FFFFFFF의 주소를 가지는 주변장치는 주변장치의 레지스터와 메모리가 직접 맵핑되며, 0x20000000 이상의 주소를 가지는 주변장치의 경우는 MMU에 의해 주변장치의 레지스터와 메모리가 맵핑된다. SDRAM의 경우 최소 8MB의 용량이 요구되며 최대 128MB까지 가능하다. 사용자 콘솔 장치로 JTAG UART 또는 Serial UART가 사용되며, 한 개의 Full Featured Timer가 필요하다. 리눅스의 경우 IRQ0은 Auto-detected를 의미하므로 Timer를 제외한 나머지 디바이스는 절대 IRQ0을 사용할 수 없다[7-8].

2.2 영상감시 시스템을 위한 Nios II 프로세서 및 주변장치 구성

Nios II 프로세서 시스템을 설계하기 위해서는 목적 시스템을 상술하고, 구성하여 시스템을 생성하는 SOPC Builder 개발도구를 사용한다. 이는 Quartus II 설계 소프트웨어 환경하에서 구동되며, SOPC 설계의 전 과정을 수행할 수 있도록 한다.

영상감시 시스템을 위한 Nios II 시스템의 하드웨어 구성은 그림 5와 같이 CPU(Nios II Processor), Camera Controller(TVP5145), VGA Controller(CHRONTEL CH7010B), I2C Controller, JTAG UART, Timer, DDR2_SDRAM 등의 컴포넌트가 필요하다.

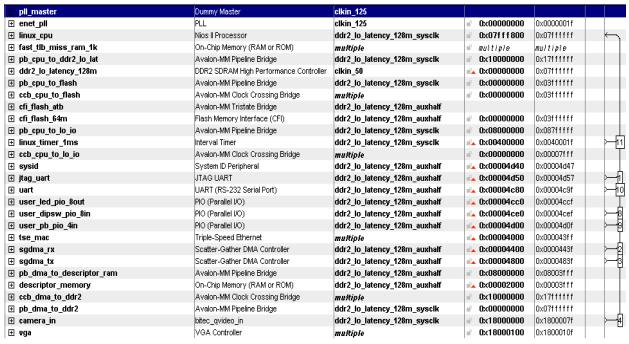

그림 5. SOPC Builder를 이용한 시스템 구성
Fig. 5. Using SOPC Builder system configuration

SOPC Builder를 이용하여 시스템을 구성하고 Generate를 실행함으로써 구성한 시스템이 자동 생성되고, 이러한 하드웨어 정보들은 .sopcinfo 파일에 저장된다. Quartus II에서 Compilation을 실행하여 .sof 파일을 생성함으로써 하드웨어 설계는 완료된다. 그림 6은 Quartus II 소프트웨어로 구성한 시스템 전체 회로도이다.

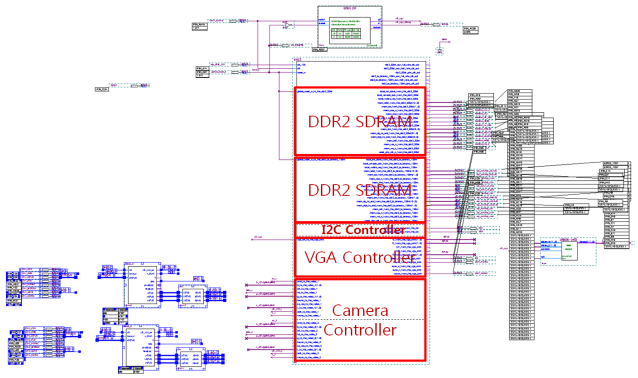

그림 6. 시스템 전체 회로도
Fig. 6. System entire circuit diagram

.sopcinfo 파일을 이용하여 리눅스에서 컴파일시 사용할 헤드파일을 표 1과 같이 생성하고, 리눅스 커널로 복사한다 [7-8].

표 1. 리눅스 컴파일을 위한 헤더파일 생성
Table. 1. Generation header file for compilation of linux

1. Nios II Command Shell 실행
2. 작업 폴더로 이동
3. sopc-create-header-files --single custom_fpga.h
4. nios2-linux/linux-2.6/arch/nios2/include/asmd으로 복사

3. 디바이스 드라이버 구현

3.1 VGA 디바이스 드라이버

Frame Buffer란 리눅스 시스템에서 그래픽을 표현할 수 있는 하드웨어를 말한다. 즉, VGA Controller가 frame buffer 장치라고 할 수 있으며, 그 하드웨어를 user application이 제어할 수 있도록 만들어진 디바이스 드라이버를 frame buffer driver라고 할 수 있다. VGA Controller에 대한 frame buffer address가 DDR2_SDRAM의 0xd7300000에 할당되어 있다. VGA Controller는 16Bit의 RGB565를 사용하며, 640×480 해상도로 설정되어 있다. 그림 7은 영상출력을 위한 VGA Controller의 레지스터 맵이다.

Register	Address	Description
control	0x00	Control Register
buffer	0x04	Buffer address Register
size	0x08	Pixel size Register

그림 7. VGA Controller의 레지스터 맵
Fig. 7. Register map of VGA Controller

그림 8는 CH7010B를 장착하고 있는 HSMC Quad Video Board의 Video output 회로도이다.

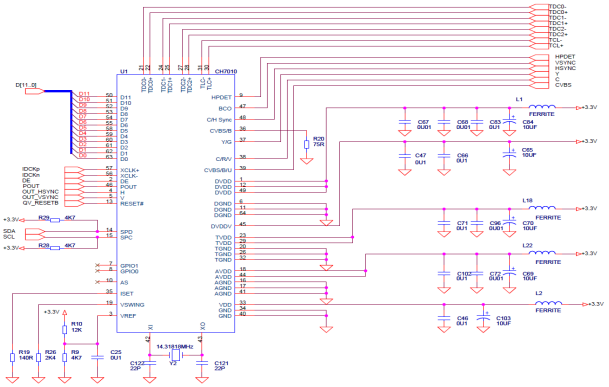

그림 8. Quad Video Board의 Video output 회로도
Fig. 8. Video output Schematic diagram of Quad Video Board

그림 9는 본 연구에서 구현한 Frame Buffer 드라이버의 동작 순서도이다. insmod 또는 리눅스가 부팅되면 frame buffer 드라이버의 초기화 함수 호출과 platform_driver_register() 함수 호출을 통하여 platform_driver의 probe() 함수가 호출된다. probe 함수에는 frame bufer를 640*480*2의 크기로 할당하며, platform_device가 등록될 때 관련된 resource를 획득한다. I/O 공간을 가상 주소에 할당한 후 반환된 주소를 이용하여 VGA Controller의 내부 레지스터를 설정한다. fb_var_screeninfo와 fb_fix_screeninfo 구조체를 이용하여 frame buffer 하드웨어에 대한 정보를 획득하고 설정한다. 또한 저수준 입출력 함수나 mmap 함수가 포함된 file operation 구조체를 등록한 후 register_framebuffer (&fb_info) 함수를 호출하여 커널에 등록함으로써 user application과 연결된다.

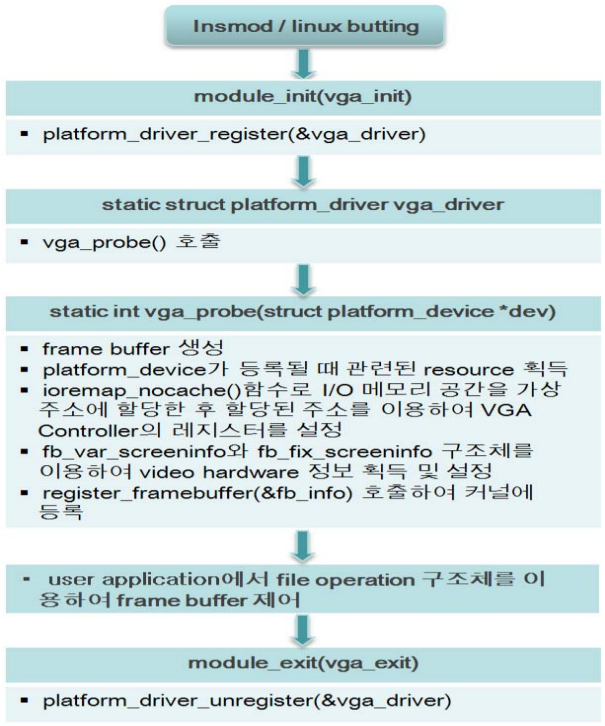

그림 9. Frame buffer driver 등록 순서
Fig. 9. Registration sequence of frame Buffer driver

3.2 카메라 디바이스 드라이버 구현

본 연구에서 사용한 카메라는 4채널 Composit Video Input을 지원하는 TI사의 TVP5154를 사용하였다. 최대 해상도는 720×480이며, ITU-R BT.656 Standard Sampling을 지원한다. 그림 10은 카메라 Controller의 레지스터 맵이다.

Register	Address	Description
Control	0x0	Control Register
Status	0x4	Status Register
Address	0x8	Buffer Address Register
Pane	0xC	Pane Register

그림 10. 카메라 Controller의 레지스터 맵
Fig. 10. Register map of Camera controller

그림 11은 Video Input 모듈의 블록도이다. 4개의 채널을 통해 영상을 입력 받으며, AD변환과 Scaler 블록을 거쳐 YCbCr 포맷으로 출력된다.

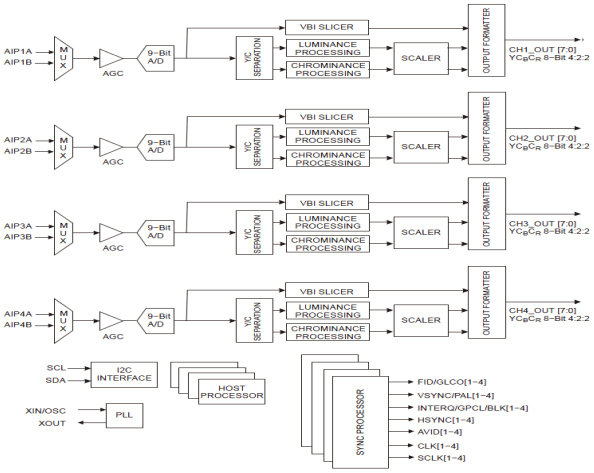

그림 11. Video Input 모듈의 블록도
Fig. 11. Block diagram of Video Input module

I2C 통신으로 Video Input 모듈의 Scaler 레지스터를 설정하여 320×240 해상도의 영상을 Camera Controller로 전달한다. 그림 12는 영상 입력부의 회로도이다. 카메라로부터 받은 영상을 BT656 모듈에서는 YCbCr로 변환하고, CSC 모듈에서는 RGB로 변환하여 Nios II 프로세서로 전송한다. Camera Controller에 대한 image buffer address가 DDR2_SDRAM의 0xd7200000에 할당되어 있다.

그림 13은 본 연구에서 구현한 Camera 디바이스 드라이버의 동작 순서도이다. insmod 또는 리눅스가 부팅되면 tvp5154 카메라 디바이스 드라이버의 초기화 함수와 platform_driver_register() 함수 호출을 통하여 probe()함수를 호출한다. probe() 함수에는 i2c 통신으로 카메라 모듈을 셋팅하고 frame buffer를 640*480*2의 크기로 할당하며, platform_device에 관련된 resource를 획득한다. I/O 메모리 공간을 가상 주소에 할당한 후 반환된 주소를 이용하여 Camera Controller의 내부 레지스터를 설정한다. register_framebuffer(&fb_info)를 호출하여 카메라 디바이스 드라이버의 등록을 완료한다.

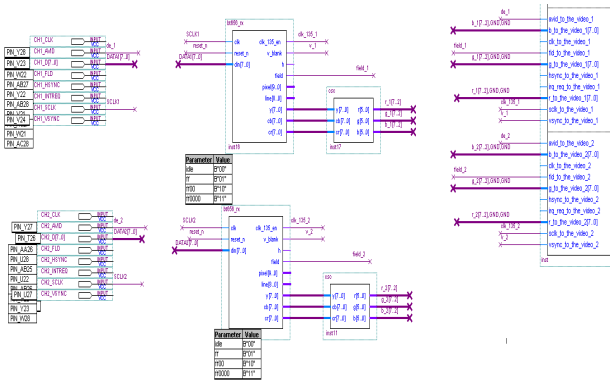

그림 12. 영상 입력부의 시스템 회로도
Fig. 12. System circuit diagram of Image Input part

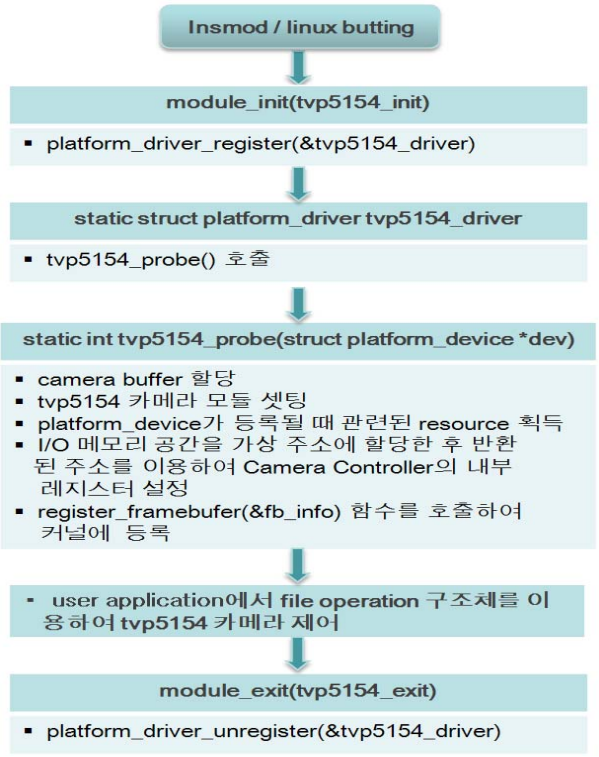

그림 13. 카메라 디바이스 드라이버 등록 순서
Fig. 13. Registration sequence of camera device driver

4. 실험 및 결과

4.1 실험 환경

본 논문의 실험은 Altera Cyclone III 디바이스가 장착된 Bitec Video Development Kit와 TVP5154 카메라 모듈과 CH7010B VGA 모듈을 사용하여 시스템을 구성하였다. 하드웨어 개발에는 Altera Design Suite 9.1을 사용하였으며, 소프트웨어 개발은 Fedora 7기반 Linux Kernel 2.6.23을 사용하였다.

표 2. 실험 환경
Table 2. Experiment environment

▣ Host PC
▶ 하드웨어 개발 : 윈도우 기반 Quartus II 9.1 ▶ 소프트웨어 개발 : RED Hat Fedora 7 기반 linux Kernel 2.6.30 ▶ GCC tool chain version : 4.1.2
▣ Embedded system
▶ 프로세서 : Nios II / Full featured (100MHz), MMU ▶ Flash memory : 64MB ▶ DDR2 SDRAM : 128MB ▶ Camera : TVP5154 ▶ VGA : CH7010B

4.2 실험 결과 및 성능 비교

2장에서 설계한 하드웨어를 개발보드의 FPGA에 실장한다. 3장에서 구현한 디바이스 드라이버를 커널에 추가하여 리눅스 이미지를 생성한다. 생성된 이미지를 타켓 시스템의 SDRAM에 다운로딩하면 JTAG 포트를 통해 Linux 부팅 메시지가 출력된다. 그림 14는 리눅스 부팅 메시지이다.

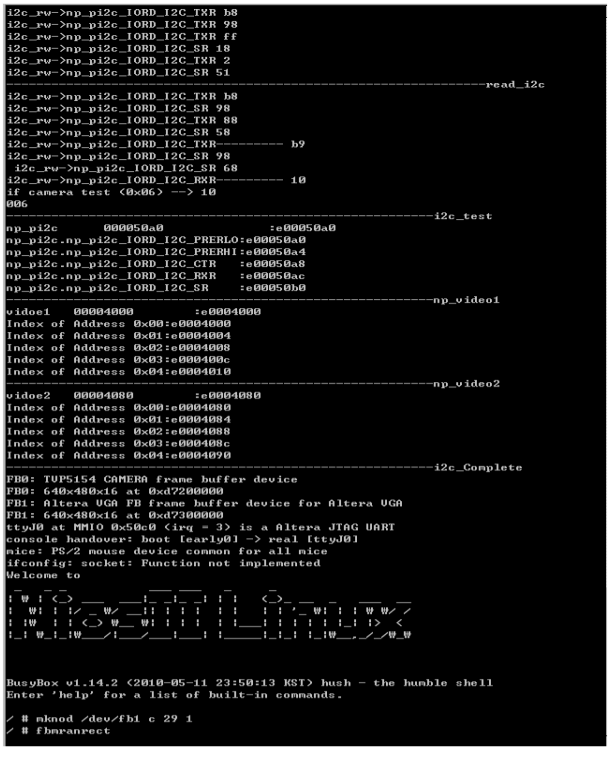

그림 14. 리눅스 부팅 메시지
Fig. 14. linux booting message

그림 15는 카메라 디바이스 드라이버와 VGA 디바이스 드라이버를 실행한 결과 화면이다. 카메라로부터 입력 받은 영상을 320*240 해상도로 스케일링하여 VGA 모니터에 출력됨을 확인할 수 있다.

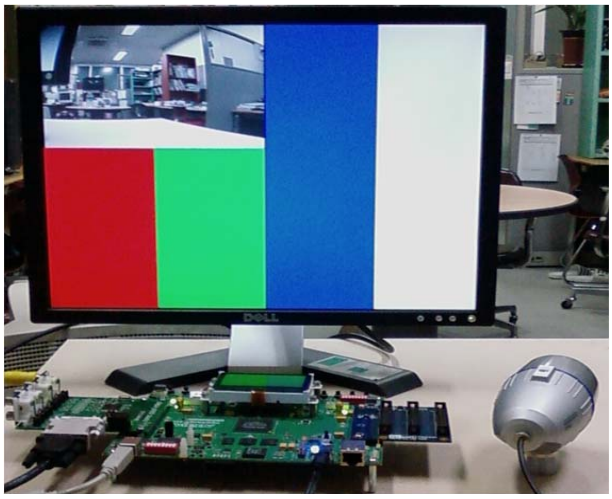

그림 15. 실험 화면
Fig. 15. Experiment screen

본 연구에서 구현한 시스템의 성능을 평가하기 위해 PC 기반의 CCTV 시스템과 비교하였다. PC 기반의 CCTV 시스템의 사양은 Intel Core 2 Quad CPU 2.33GHz, 3GB RAM의 PC를 사용하였으며, 비디오 Grabber Board는 NTSC 비디오 입력, 640*480 해상도를 사용하였다. 표 4는 PC 기반 CCTV 시스템과 본 연구의 시스템과 성능 평가의 결과로써, PC 기반 CCTV 시스템의 성능에는 미치지 못하지만 초당 15 프레임의 처리속도를 확인 할 수 있었다.

표 3. PC 기반의 CCTV 시스템과 성능 비교
Table 3. Performance compare with PC Based CCTV system

	처리 속도	영상 사이즈
PC 기반 CCTV 시스템	29.97 frame/sec	640*480
본 시스템	14.6 frame/sec	640*480

5. 결 론

본 논문에서는 PC기반 CCTV 시스템의 단점을 보완 할 수 있는 영상 감시 시스템을 위한 Nios II 임베디드 프로세서 시스템의 Linux 디바이스 드라이버를 구현하였다.

Altera Nios II 임베디드 시스템은 각종 응용에 유연하고 적응성이 뛰어나지만, 시스템의 설계 구현이 어렵다. 이러한 이유로 Nios II 시스템을 위한 카메라 IP 및 디바이스 드라이버 설계 구현에 대한 연구가 국내·외적으로 거의 보고되고 있지 않다.

본 연구의 결과는 임베디드 리눅스를 Nios II 시스템에 포팅하였으며, Linux 기반 Nios II 시스템의 카메라 디바이스 드라이버와 VGA 디바이스 드라이버를 구현하여 임베디드 시스템 기반 영상 감시 시스템을 구현하였다.

향후 연구 과제로 추적 및 인식 기술을 이용하여 지능적

무인 감시 시스템을 구현할 필요가 있다.

참 고 문 헌

[1] 정보통신연구진흥원, "지능형 CCTV 기술 및 시장 동향", *주간기술동향*, 통권, 1361호 pp. 13-27, 2008.

[2] ETRI, "지능형 영상보안 기술현황 및 동향", *전자통신동향분석*, 23권, 4호, pp. 80-88, 2008.

[3] 박영석, *PLD를 이용한 디지털 시스템의 설계*, 경남대 지능형홈 인력양성사업팀, pp. 1-480, 2005

[4] Altera Corp, *Nios II Hardware Development Tutorial*, V2.5, pp. 1-41, 2007

[5] Altera Corp, *Quartus II Handbook*, V9.1, 1-2454, 2009

[6] Altera Corp, *Embedded Design Handbook*, V2.2, pp. 1-306, 2009

[7] Nios Community Wiki, *Running Linux on the Altera Nios II Processor USER GUIDE*, V1.0, pp. 1-45, 2010

[8] Nios Community Wiki, www.nioswiki.com

[9] 다니엘 보베이, 마르코 체사티, *리눅스 커널의 이해*, 한빛미디어, pp. 29-846, 2006

[10] 유영창, *리눅스 디바이스 드라이버*, 한빛미디어, pp. 28-910, 2007

저 자 소 개

김동진 (Dong-Jin Kim)
2007년 : 경남대 정보통신공학과 학사 졸업.
2009년 : 경남대 정보통신공학과 석사 졸업
2009～현재 : 경남대 정보통신공학과 박사 과정

관심분야 : FPGA설계, 임베디드 시스템
Phone : 010-9331-6344
Fax : 05059992163
E-mail : rivaldo2000@nate.com

정용배 (Young-Bee Jung)
2004년 : 경남대 정보통신공학과 학사 졸업.
2005년 : 경남대 정보통신공학과 석사 졸업
2010년 : 경남대 정보통신공학과 박사 졸업

관심분야 : 영상처리, FPGA설계, 임베디드 시스템
Phone : 010-5875-3525
Fax : 05059992163
E-mail : wjddydqo1@nate.com

김태효 (Young-Bee Jung)
1988년 : 영남대 전자공학과 박사 졸업.
1979년～현재 : 경남대 정보통신공학과 교수
1990년～1991년 : 펜실베니아대학 Post Doc.

관심분야 : 영상처리, 컴퓨터비전, 영상계측
Phone : 055-249-2645
Fax : 05059992163
E-mail : hyo@nate.com

박영석 (Young-Seak Park)
1979년 : 영남대 전자공학과 학사 졸업.
1981년 : 한양대 전자공학과 석사 졸업
1985년 : 한양대 전자공학과 박사 졸업
1990～1991년 : 일본 우정성 통신총합연구소(관서선단연구센터) 초빙과학자
1990～1991년 : 일본 긴끼이동통신센터 객원연구원
2001년～2002년 : 미국 North Carolina 주립대학(NCSU) 교환교수
2001년～현재 : 경남대 정보통신공학과 교수

관심분야 : Software Engineering, Web-based Software Design & Development, Pattern Recognition, Image Processing, Computer Network & Network Computing, Embedded Processor System HW/SW
Phone : 055-249-2644
Fax : 05059992163
E-mail : yspark@kyungnam.ac.kr