USB를 이용한 고정세 FPD 화질 테스트 시스템 설계 및 구현

조대현, 김근홍, 이동호
경북대학교

Design and Implemente high pitch FPD image quality test system by using USB

Dae-Hyun Cho, Gun-Hong Kim, Dong-Ho Lee
Kyungpook National University

Abstract - 최근에 각광받는 영상표시 장치인 LCD와 PDP는 Progressive 방식으로 영상을 표시하며 화질 또한 SD에서 HD로 바뀌었다. 따라서 대용량의 데이터를 빠른 시간에 FPD 패널로 전송해야 하며 이를 위해 LVDS 전송 방법을 사용한다. 대부분의 화질테스트 장비들은 빠른 데이터 전송을 위해 내부에 저장되어 있는 특정 패턴들만 LVDS 신호로 바꾸어 패널로 전송하게 되며 저장되어 있는 영상을 움직이는 것도 제한적이다. 하지만 LCD 패널의 응답속도, PDP 패널의 DFC등을 테스트 하기위해서는 다양한 패턴이 필요하며 영상의 움직임 또한 자유롭게 조절할 수 있어야 한다. 따라서 본 논문에서는 다양한 패턴을 다운로드 받아서 표시할 수 있는 테스트 장비를 설계하고 USB를 통해 테스트 장치 컨트롤 및 이미지 다운로드를 할 수 있는 프로그램을 설계하여 고정세 FPD의 화질을 테스트 할 수 있는 시스템을 구현하고자 한다.

1. 서 론

기존의 디스플레이 장치인 브라운관은 대부분 영상을 표시할 때 완성된 한 화면을 만들기 위해 홀수 라인과 짝수 라인을 번갈아 표시하는 방식인 Interlace 방식을 사용하였다. 하지만 최근에 각광받는 대표적인 FPD(Flat Panel Display) 영상표시 장치인 LCD와 PDP는 Progressive방식으로 완성된 한 화면을 홀수 라인과 짝수 라인의 구분 없이 한번에 전체 영상을 표시한다. 즉 영상을 60Hz로 표시한다고 가정하였을 경우 브라운관은 30Frame을 표시하지만 PDP와 LCD는 60 Frame을 표시하게 된다. 따라서 FPD에 전송되어야 할 영상데이터의 크기는 브라운관에 비해 두배 커지게 된다. 뿐만 아니라 FPD의 화질 또한 SD에서 HD로 바뀌었기 때문에 FPD에 전송되어야 할 데이터의 양은 기존의 브라운관에 비해 상당히 커지게 된다. 이렇게 커진 영상데이터를 기존의 정해진 시간 이내에 FPD 패널로 모두 전송해야 하기 때문에 고속의 데이터 전송이 필요하며 이를 위해 직렬 전송 방법인 LVDS 전송 방법을 사용한다.

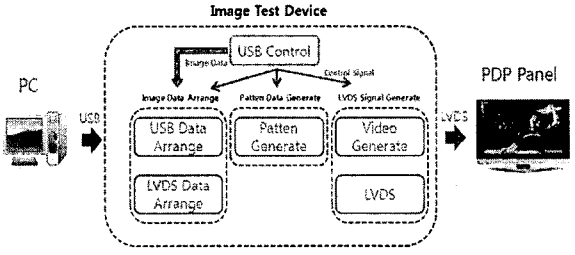
대부분의 화질테스트 장비들은 많은 양의 영상 데이터를 고속으로 전송해야 하기 때문에 내부에 저장되어 있는 특정 패턴들만 LVDS 신호로 바꾸어 패널로 전송하게 되며 저장되어 있는 영상을 움직이는 것도 제한적이다. 하지만 LCD 패널의 응답속도, PDP 패널의 DFC등을 테스트 하기위해서는 다양한 패턴이 필요하며 영상의 움직임 또한 자유롭게 조절할 수 있어야 한다. 뿐만 아니라 효율적인 화질 테스트를 위해서는 테스트 장비와 PC를 쉽게 연결할 수 있어야 하며 테스트 장비의 조작 또한 편리해야 한다.

본 논문에서는 다양한 패턴을 USB를 통해 다운로드 받아서 표시할 수 있는 화질 테스트 장비를 설계하고 PC에서 USB를 통해 테스트 장치의 컨트롤 및 이미지 다운로드를 할 수 있는 프로그램을 설계하여 고정세 FPD의 화질을 테스트 할 수 있는 시스템을 구현하였다.

2. 본 론

2.1 테스트 시스템 구성

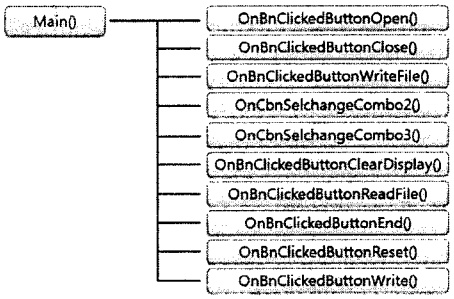
전체 시스템 구성은 USB를 컨트롤하고 이미지를 전송하는 PC와 전송 받은 이미지를 LVDS 신호로 바꾸어서 각종 Sync Signal과 영상 데이터를 함께 전송하는 화질 테스트 장비 그리고 FPD 이루어져 있다.



<그림 1> 전체 시스템 구성도

2.2 USB 컨트롤 S/W

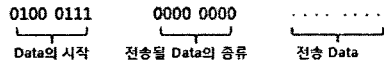
PC와 테스트 장비의 USB 인터페이스를 구축하기 위해 FT245BM이라는 USB 컨트롤러를 사용하였으며 칩 제조사에서 제공하는 USB 설치 드라이버를 사용하였다. USB 인터페이스를 이용하여 테스트 장비를 컨트롤하며 이미지를 다운로드 시키는 프로그램은 Visual C++ .NET으로 작성하였으며 프로그램의 전체적인 구조는 그림 2와 같다.



<그림 2> USB 컨트롤 프로그램 구조

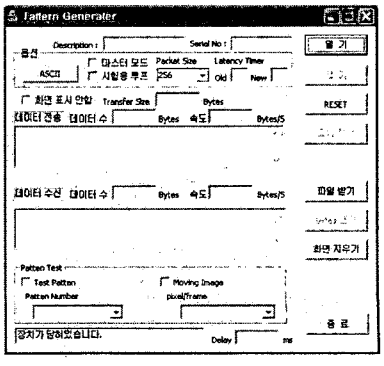
이 프로그램의 가장 주된 Class는 USB 컨트롤러와 PC를 연결하는 Open() Class, 연결을 닫는 Close() Class, 그림은 전송하는 WriteFile() Class, 패턴 명령을 전송하는 Conbo2() Class, 그림의 이동 명령을 전송하는 Combo3() Class이다. USB의 상태를 나타내는 함수들은 칩 제조사에서 제공하는 FTD2XX.h에 선언되어 있으며 프로그램 작성시 Include 하여 사용하면 된다.

파일 전송은 8bit 단위로 하게 되며 가장 처음 보내는 2byte가 다음에 올 데이터의 정보를 나타내는 Head 역할을 하게 된다. 테스트 환경 구축시 디스플레이 장치로 42인치 HD PDP를 사용하였기 때문에 모든 패턴의 크기는 여기에 맞추어져 있으며 Data의 구성은 아래 그림과 같다.



<그림 3> 전송 데이터 구성

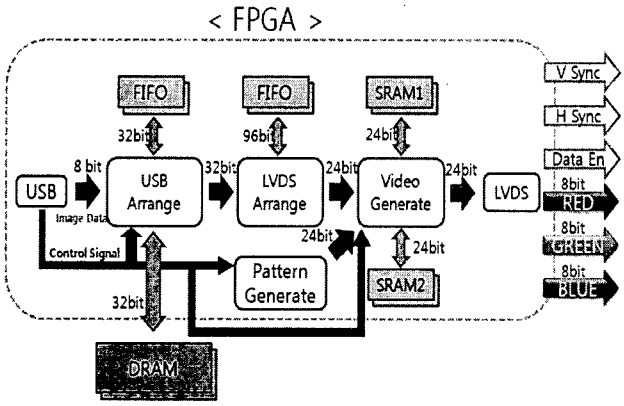
0x47은 Data의 시작을 알리는 것으로 Data를 전송할 때 가장 앞에 오는 값이다. 이 값 다음에 전송되는 1byte는 전송될 Data가 무엇인지는 나타내는 것으로써 3가지 종류가 있다. 0x00은 다음에 전송될 Data가 Image라는 것을 나타내며 1024X768X3byte의 영상 Data가 전송되게 된다. 0x02이면 다음에 전송될 Data는 테스트 장비 내부에서 만들어지는 패턴의 번호를 의미하는 것이고 0x40 다음에 전송될 Data는 패턴이 프레임당 몇 픽셀이 움직일 것인지를 나타낸다. 작성된 프로그램의 UI설계는 아래와 같다.



<그림 4> Data 전송 프로그램 UI

2.3 H/W 구조

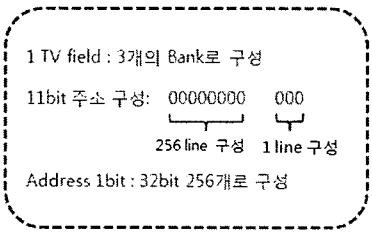
USB Controller로부터 8bit Data를 입력받아 처리하는 부분은 FPGA로 설계하였으며 사용된 FPGA는 Altera사의 APEX 20ke 모델이다. FPGA 내부 구조는 그림 5와 같다.



<그림 5> H/W 구조

USB module 은 외부에 연결되어 있는 USB controller을 제어한다. rxf, txe, rd, wr로 이루어진 제어신호를 생성함으로써 외부 USB controller을 제어하게 되며 Data Bus 폭은 8bit로 이루어져 있다. 그리고 전송된 Data의 첫 부분을 찾아 Data의 내용을 판단한 후에 해당하는 제어 신호를 만들어 USB Arrange module과 Patten Generate module, Video Generate module을 제어한다. 이때 전송받은 Data가 그림일 경우 그림에 해당하는 Data는 USB Arrange module로 넘어간다.

USB Arrange Module은 USB module을 통해 입력받은 8bit Data를 32bit단위로 정렬하여 FIFO에 저장하게 된다. USB를 통해 전송된 영상 Data는 24bit의 RGB 영상으로 32bit의 Data Bus를 가지고 있는 SDRAM에 바로 저장할 경우 1 Pixel 마다 8bit씩을 낭비하게 된다. 이렇게 되면 엄청난 양의 저장 공간을 낭비하게 되므로 이것을 방지하기 위해 전송된 Data를 32bit로 재 정렬할 필요가 있다. 영상 Data의 재정렬을 위해 FIFO를 사용하며 FIFO에 256개의 entry가 차게 되면 SDram controller module을 통해 SDram에 저장하게 된다. 이때 사용되는 SDram controller module은 메모리의 초기화와 입출력을 담당하는 모듈로서 FPGA 외부에 연결되어 있는 SDram의 Cs, Ras, Cas, We, DQM, Addr, Dqs등과 같은 컨트롤 신호를 제어한다. SDram에 저장되는 Data의 주소 구성은 다음과 같다.



<그림 6> Address 구성

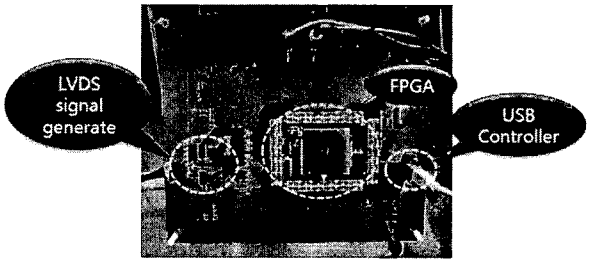
1TV field 영상의 저장이 끝나면 USB Arrange Module은 SDRam에 저장된 영상을 읽어 LVDS Arrange module로 Data를 보내게 된다.

LVDS Arrange module은 USB Arrange module로부터 입력받은 32bit 영상을 FIFO를 이용하여 다시 24bit 단위로 바꾸는 역할을 한다. 32와 24의 최소 공배수가 96이기 때문에 96bit의 FIFO를 사용하여 Data를 24bit로 재 정렬 하고 이것을 Video generate module로 전송하게 된다.

Video generate module은 영상을 1 line 단위로 두 개의 Sram에 저장한 후에 다시 읽을 때 Address의 시작점을 조정함으로써 움직이는 영상을 만들어 낸다. 영상을 실시간으로 저장하고 읽어야 하기 때문에 double buffering이 필요하고 따라서 SRAM도 두 개가 사용되었다. Sram Read 시 Address를 조정함으로써 영상의 움직임 방향, 속도를 모두 조절 할 수 있으며 fps(frame per sec)가 초당 60 혹은 30을 마음대로 조절할 수 있다. Video generate module에 입력되는 영상 Data는 LVDS Arrange module로부터 오는 영상 Data와 Pattern generate module로부터 오는 영상 Data로 구성되어 있으며 USB module에 의해 선택되어진 Display mode에 따라서 선택적으로 입력 받게 된다.

Pattern Generate module은 USB module의 Control signal에 의해서 제어되며 입력된 8bit 값에 해당하는 패턴은 만들어서 Video Generate module로 전송하게 된다.

LVDS module은 V_sync, H_sync, Data enable 신호를 생성하며 Video generate module로부터 Data를 입력받아 Sync Signal에 따라 Data를 외부에 있는 LVDS transfer chip으로 출력하게 된다. 외부로 출력되는 Signal의 사양은 1024X768, 70Mhz이다.



<그림 7> 제작된 영상 테스트 장비

3. 시스템 구축 및 실험

<그림 8> 구축된 영상 테스트 시스템

그림8은 앞에서 소개한 테스트 시스템을 구축한 것이다. 다운로드한 파일은 Raw파일 형태의 이미지파일을 사용하였다. FPGA는 Altera사의 EP20K600EBC 칩을 사용하였으며 LVDS 신호는 Thine사의 THC63LVDM83R 두 개를 사용하여 만들었다. 메모리는 64Mbyte(2M×32)의 삼성 SDRAM을 사용하였고 USB Controller로는 FT245 Module을 사용하였다. FPGA의 외부 클럭은 5MHz를 사용하였으며 FPGA 내부의 PLL을 사용하여 70MHz 만들어 이것을 시스템 클럭으로 사용하였다.

S/W 제작 툴로는 Visual C++ .NET을 사용하였고 H/W 시뮬레이션 툴로는 Modelsim 6.2를 사용하였며 H/W 합성툴로는 Qurtus 5.1을 사용하였다. 합성 결과는 표1과 같다.

<표 1> H/W 합성 결과

Device	EP20K600EBC652-1x
Logic element	1,181/24,320(5%)
Total Pins	105/488(22%)
Total memory bits	81,920/311,296(26%)
Max clock	75.76 MHz

디스플레이 장치로는 42인치 PDP를 사용하였으며 패턴에 따른 DFC(Dynamic False Contour)의 발생정도, 계조 표현력 , 움직이는 stripe 패턴의 인식정도를 측정하였다. 특히 DFC의 경우는 피부색과 같은 특정한 패턴에 주로 많이 나타나기 때문에 DFC의 발생정도를 확인하기 위해서 DFC가 많이 일어날 것 같은 영상을 선별하여 테스트 하였다.

4. 결론

본 논문에서는 고정세 FPD의 화질 테스트 시스템 설계 구조를 제시하고 이를 바탕으로 실제 시스템을 제작하여 42인치 PDP 패널의 화질 테스트를 시행하였다. 원하는 패턴을 마음대로 움직여 볼 수 있기 때문에 FPD패널의 화질 테스트를 보다 정확하게 할 수 있었으며 USB 인터페이스를 사용함으로써 손쉽게 할 수 있었다.

[참 고 문 헌]

[1] 박시찬, "비디오 스트리밍을 위한 USB 디바이스 소프트웨어 구현", 경북대학교·졸업논문, 2004

[2] 양성규, "PDP 테스트-베드 모듈 구현을 위한 HDL 소프트웨어 구조", CICS'06, 381-384, 2006