LN2440SBC 임베디드 시스템의 가상 스크린을 위한 LCD 모듈 초기화 및 패널 디스플레이

LCD Module Initialization and Panel Display for the Virtual Screen of LN2440SBC Embedded Systems

오삼권*, 박근덕*, 김병국*

Sam-Kweon Oh*, Geun-Duk Park* and Byoung-Kuk Kim*

요 약

전원이나 CPU 같은 컴퓨팅 자원의 제약을 받는 임베디드 시스템의 경우, 데이터의 컴퓨터 화면 디스플레이로 인한 오버헤드는 시스템의 성능에 지대한 영향을 줄 수 있다. 본 논문은 ARM9 방식의 S3C2440A 마이크로프로세서가 내장된 LN2440SBC 시스템에서, LCD(liquid crystal display)의 구동을 위해 필요한 ARM 코어, LCD 컨트롤러, SPI(serial peripheral interface) 같은 LCD-구동 요소들의 초기화 방법을 설명한다. 또한 픽셀 디스플레이 함수와 디스플레이로 인한 오버헤드를 줄이기 위한 가상스크린을 이용한 디스플레이 방법을 소개한다. 가상스크린은 한 LCD 화면 디스플레이에 필요한 메모리 용량보다 매우 큰 용량의 메모리 공간으로서, 가상스크린 내의 특정 영역 디스플레이는 그 영역을 뷰포트(view port) 영역으로 지정함으로써 행해진다. 이 같은 가상스크린을 이용한 디스플레이는 임베디드 시스템에서 동시에 수행되는 여러 태스크들이 자신의 수행 결과를 생성하고 그 결과를 각기 화면에 출력하는 경우에 유용하며, 특히 각 태스크의 수행 결과 데이터가 모두 변경되지 않고 일부분만 변경되어 디스플레이 되는 경우에 더욱 그러하다. 이런 임베디드 시스템의 경우, 각 태스크들이 가상스크린의 메모리 영역을 효율적으로 나누어 사용할 경우 디스플레이 오버헤드를 최소화 할 수 있다. 가상스크린을 사용하는 경우와 아닌 경우의 성능 비교를 위해 두 개의 다른 이미지를 번갈아 가며 디스플레이하고 소요 시간을 측정했다. 그 결과 가상 스크린을 사용할 경우가 그렇지 않은 경우에 비해 약 5배의 빠른 출력 속도를 보인다.

Abstract

In case of an embedded system with computing resource restrictions such as system power and cpu, the overhead due to displaying data on the computer screen may have a significant influence on the system performance. This paper describes an initialization method for LCD-driving components such as an ARM Core, an LCD controller, and an SPI(serial peripheral interface). It also introduces a pixel display function and a panel display method using virtual screen for reducing the display overhead for an LN2440SBC system with an ARM9-based S3C2440A microprocessor. A virtual screen is a large space of computer memories allocated much larger than those needed for one-time display of an image. Displaying a specific region of a virtual screen is done by assigning it as a view-port region. Such a display is useful in an embedded system when concurrently running tasks produce and display their respective results on the screen; it is especially so when the execution result of each task is partially modified, instead of being totally modified, on its turn and displayed.

If the tasks running on such a system divide and make efficient use of the region of the virtual screen, the display overhead can be minimized. For the performance comparison with and without using the virtual screen, two different images are displayed in turn and the amount of time consumed for their display is measured. The result shows that the display time of the former is about 5 times faster than that of the latter.

Key words : Embedded Systems, TFT LCD Porting, Display Overhead, Virtual Screen.

I. 서 론

TFT LCD는 작은 부피와 적은 전력 소모라는 장점

* 호서대학교 컴퓨터공학과(Division of Computer Engineering, Hoseo University)
· 제1저자 (First Author) : 오삼권
· 투고일자 : 2010년 5월 6일
· 심사(수정)일자 : 2010년 5월 6일 (수정일자 : 2010년 6월 22일)
· 게재일자 : 2010년 6월 30일

으로 인해 핸드폰, 내비게이션, PMP(portable multimedia player), MP3(MPEG 1 Audio Layer 3) 등 대부분의 휴대용 기기의 출력 장치로 사용 되고 있다. 이러한 프로세서 성능이나 전원 등의 자원 제약을 받는 임베디드 시스템의 경우, 데이터의 화면 디스플레이로 인한 오버헤드는 시스템의 성능에 지대한 영향을 줄 수 있다.

본 논문은 LP35 TFT LCD 모듈이 부착된 고성능의 카 내비게이션 용 테스트 보드인 LN2440SBC 임베디드 시스템의 가상스크린을 위한 LCD 모듈 초기화 및 패널 디스플레이 방법을 제시한다.

ARM 코어, LCD 컨트롤러, LCD 장치와의 통신을 위한 SPI(serial peripheral interface)와 같은 필수적인 LCD 모듈의 초기화를 위해 클럭 속도 설정, GPIO(general purpose input output)을 LCD와 SPI 용으로의 할당. SPI의 마스터/슬레이브 및 보오 레이트 설정, LCD 컨트롤러 레지스터 설정을 통한 LCD 기능 선택. 그리고 SPI를 통한 LCD 장치로의 파워 온(power on) 명령 전달 등을 수행하며, 화면 출력을 위해 기본적으로 제공된 픽셀 디스플레이 함수에 대해 설명한다.

또한 디스플레이 오버헤드를 줄이기 위한 가상스크린을 이용한 디스플레이 방법을 설명 한다. 가상스크린은 한 LCD 화면 디스플레이에 필요한 메모리 용량보다 매우 큰 용량의 메모리 공간으로서, 가상스크린 내의 특정 영역의 디스플레이는 그 영역을 뷰포트(view port) 영역으로 지정함으로써 행해진다[1].

이 같은 가상스크린을 이용한 디스플레이는 임베디드 시스템에서 수행되는 여러 태스크들이 자신의 수행 결과를 생성하고 그 결과를 각기 화면에 출력하는 경우에 유용하며, 특히 각 태스크의 수행 결과 데이터가 모두 변경되지 않고 일부분만 변경되어 디스플레이 되는 경우에 더욱 그러하다. 이런 임베디드 시스템의 경우, 각 태스크들이 가상스크린의 메모리 영역을 효율적으로 나누어 사용할 경우 디스플레이 처리로 인한 오버헤드를 최소화 할 수 있다.

가상스크린을 사용할 경우와 아닌 경우의 성능 비교를 위해 두 개의 다른 이미지를 번갈아 가며 디스플레이하고 소요 시간을 측정한 결과, 가상 스크린을 사용할 경우가 그렇지 않은 경우에 비해 약 5배의 빠른 출력 속도를 보인다. 소요시간의 99%는 이미지 데이터를 가상스크린에 기록하는데 걸린 시간이고 소요시간의 1%는 가상스크린을 통한 화면 전환에 걸리는 시간이다.

본 논문의 구성은 다음과 같다. 2장은 LN2440SBC와 그 안에 내장된 S3C2440A 마이크로프로세서, LP35 TFT LCD 모듈 그리고 가상스크린에 대해 설명한다. 3장은 TFT LCD 구동을 위한 초기화 및 가상스크린 설정 방법을 설명하고 4장은 TFT LCD 디스플레이를 위한 픽셀 디스플레이 함수의 구현을 설명한다. 5장은 가상스크린을 이용한 디스플레이 방법에 대해 설명하며 6장은 가상스크린을 사용한 경우와 사용하지 않은 경우의 성능 평가를 수행한다. 마지막으로 7장에서 결론을 맺는다.

Ⅱ. LN2440SBC 임베디드 시스템

LN2440SBC 임베디드 시스템의 S3C2440A 마이크로프로세서는 ARM920T ARM 코어, TFT/STN LCD 컨트롤러, SPI 등을 내장하고 있다[2]-[4]. 그림 1은 LP35 TFT LCD 모듈이 부착된 LN2440SBC의 S3C2440A 마이크로프로세서를 보여준다.

S3C2440A에 내장된 LCD 컨트롤러는 프레임버퍼와 외부 LCD 드라이버 사이에서 비디오 데이터 전송을 맡아서 처리하는 제어 로직이다. 프레임버퍼에 있는 비디오 데이터는 LCD 전용 DMA(direct memory acess) 컨트롤에 의해 LP35의 LCD 드라이버로 전송되며, 그 결과 LCD 화면에 이미지가 출력된다.

S3C2440A의 LCD 컨트롤러는 단색 LCD부터 컬러 STN LCD, TFT LCD를 지원하며, TFT LCD의 경우 1, 2, 4, 8, 16, 24 bpp(bits per pixel) 컬러 디스플레이를 지원하고 640x480, 320x240, 160x160 등의 다양한 해상도 크기를 지원한다.

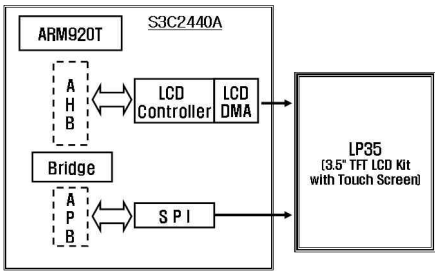


그림 1. LP35를 부착한 LN2440SBC의 S3C2440A
Fig. 1. S3C2440A of LN2440SBC with LP35

S3C2440A의 LCD 컨트롤러는 시스템 메모리의 일부를 프레임버퍼로 사용한다. 프레임버퍼란 화면에 출력 될 각 픽셀의 색상 값이 비트맵 형식으로 저장되어 있는 기억장소로써, 프레임버퍼에 저장된 비트맵 정보는 화면의 픽셀에 그대로 대응하여 출력된다. 4MB의 가상스크린 프레임버퍼를 지원하며, 이는 한 픽셀의 색상 값의 크기가 24bit일 경우, 약 17개의 LCD 화면크기의 이미지를 저장 할 수 있을 만큼 큰 크기이다.

S3C2440A LCD 컨트롤러는 가상스크린 프레임버퍼에 LCD 화면보다 큰 이미지 데이터를 저장한 후, 실제 LCD 화면에는 가상스크린 프레임버퍼의 일부분만을 출력하고 스크롤링 하면서 화면에 보여 지는 내용을 조절할 수 있는 가상스크린 프레임버퍼 수평, 수직 스크롤 기능을 하드웨어적으로 지원 한다. LCD 모듈에 간단한 그림을 그리거나 몇 개의 문자를 표시하는 경우라면 고려할 사항은 아니지만 빠른 애니메이션이 필요하거나 빠른 화면 전환이 필요한 경우에는 반드시 고려해야 할 사항이다. 특히 그림이 그려진 부분에 다른 그림을 그리고 원래 그림을 복구할 필요가 있는 경우 더욱 그러하다.

LP35에 내장된 LTV350QV-F04 LCD는 320*240의 해상도를 가지는 3.5" TFT LCD로 최대 1670만 개의 색상을 표현할 수 있다. 24비트 RGB(red green blue) 인터페이스와 LCD 제어 명령의 수신을 위한 SPI가 내장되어 있으며, LCD 디스플레이에 필요한 신호 및 데이터의 통신 인터페이스로 비동기식의 DE 모드와 동기식의 SYNC 모드를 지원한다. 이중 LP35는 SYNC 모드로 고정되어 있다[5].

ARM920T는 SPI[6]를 통해 LCD 장치로 제어 명령을 전송한다. SPI는 프로세서에 내장된 동기식 직렬 전송을 지원하는 전이중 방식의 직렬 인터페이스이다. 통신하는 한 쌍의 장치는 마스터-슬레이브 관계를 형성하며, 마스터에 의해 생성된 클럭 신호에 의해 동기화 된다. 입출력 핀으로는 SPIMOSI(master out, slave in pin)와 SPIMISO (master in, slave out pin)가 있다.

Ⅲ. TFT LCD 구동을 위한 초기화 과정

LP35 모듈의 구동을 위해 ARM사의 C 컴파일러, 디버거, 어셈블러, ARM 에뮬레이터 등을 지원하는 ARM Developer Suite (v1.2)를 사용하여 실행 이미지를 생성하고, 실행 이미지를 Arm Down(v3.8)을 이용하여 임베디드 보드로 로딩 한다.

TFT LCD의 구동을 위한 초기화 과정은 그림 2와 같다.

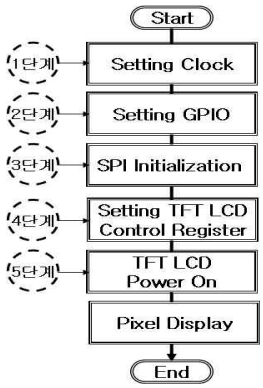


그림 2. TFT LCD 초기화 과정
Fig. 2. sequence of the TFT LCD initialization

1단계에서는 각 장치로 전달되는 클럭 속도를 설정한다. S3C2440A에서는 ARM 코어가 사용하는 클럭과 인터럽트 컨트롤러, DMA, LCD 컨트롤러 등의 고속 장치에서 사용하는 클럭 그리고 ADC(analogue to digital conversion), UART(universal asynchronous receiver transmitter), SPI 등의 저속 주변장치에서 사용하는 클럭들의 클럭비를 조정할 수 있다. S3C2440A 매뉴얼[4]에는 안정화된 여러 개의 클럭비를 제시하고 있으므로, 목적에 따라 적당한 클럭비를 선택한다.

2단계에서는 설정에 따라 여러 용도로 사용할 수

있는 핀인 GPIO를 설정한다. S3C2440A는 총 130핀의 GPIO를 가지고 있으며, 이중 LCD의 구동을 위해 제어 신호용과 데이터 버스용으로 27개의 GPIO 핀들을 할당 하고 ARM 코어와 LCD 장치와의 통신 인터페이스인 SPI의 사용을 위해 SPI 입/출력 핀과 SPI 클럭핀을 할당한다[2],[4],[6].

3단계는 LCD 장치로 제어 명령을 전송하는 SPI를 초기화 하는 과정이다. 프로세서를 마스터로 설정하고 LCD 장치는 슬레이브로 설정하여 SPIMOSI는 출력 핀, SPIMISO는 입력 핀으로 지정하고, 전송 속도인 보오 레이트를 설정한다.

4단계는 TFT LCD 기능을 선택하기 위하여 LCD 컨트롤러 제어 레지스터들을 설정하는 단계이다. 표 1은 본 논문에서 설정한 주요 항목의 값들을 보여준다.

LP35 모듈은 320*240 크기의 해상도를 가지는 TFT LCD로 LCD 디스플레이에 필요한 신호 및 데이터의 통신 인터페이스는 SYNC 모드로 고정되어있다. 따라서 해상도 및 패널 수직, 수평 사이즈와 디스플레이 모드 그리고 LCD 제어 모드를 표 1과 같이 설정 한다.

시스템 메모리의 일부를 프레임버퍼로 사용하기 위해, 프레임버퍼의 시작 주소와 끝 주소를 설정한다. 이것은 그림 3과 같이 뷰포트의 시작 주소와 끝 주소를 의미하며, 뷰포트 영역내의 이미지 데이터들은 화면에 출력된다.

LP35는 최대 24 bpp의 색상을 지원하지만 320*240 크기의 저 해상도에서는 16 bpp와 큰 차이가 없기 때문에 오버헤드를 줄이기 위해 16 bpp를 선택한다. 16 bpp 색상은 RGB 비율이 5:6:5인 형식과 5:5:5:I 형식을 지원하며, I는 밝기 조절에 이용된다. 본 논문에서는 밝기 조절을 제외한 5:6:5 형식을 사용 한다.

이로써 LCD 기능의 주요 설정을 마친다. 상황에 따라 다른 설정 값을 선택할 수 있으며 자세한 내용은 S3C2440A 매뉴얼을 참조하면 알 수 있다[4].

표 1. LCD 컨트롤러 설정 항목
Table 1. configuration items of the LCD controller

설정항목	설정값
출력 해상도	320*240
패널 수직 사이즈	240-1
패널 수평 사이즈	320-1
디스플레이 모드	TFT LCD Panel
DE/Sync 모드	Sync 모드
프레임 버퍼 시작 메모리 주소	0x30800000
프레임 버퍼 끝 메모리 주소	프레임버퍼 시작메모리주소+ 가상스크린 수평크기*LCD 패널 수직크기*2
BPP 선택	16 bpp for TFT
16 bpp 비디오 포맷	5:6:5 포맷

마지막 단계인 파워-온(power-on)에서는 SPI를 통해 파워-온 명령을 LCD 장치로 전송함으로써, LCD 장치의 구동준비를 완료한다. LP35에 내장된 LTV350QV-F04 LCD는 파워-온과 파워-오프 순서를 제공하며, 해당 명령어들을 매뉴얼에서 제시하는 지연 시간과 순서를 지켜 SPI를 통해 전송해야 한다. 이로써 TFT LCD를 구동하기 위한 초기 설정이 완료된다[5].

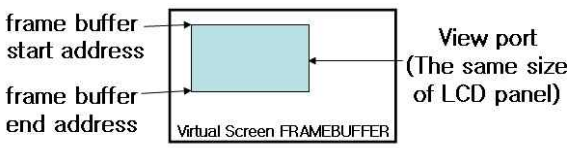


그림 3. 가상스크린 프레임버퍼의 뷰포트 영역
Fig. 3. view-port region of the virtual screen frame buffer

Ⅳ. 픽셀 출력 함수 구현

LCD 출력을 위해서는 디지털 화면을 구성하는 최소단위인 픽셀을 출력하는 함수가 필요하다. 본 논문에서 구현한 TFT_PutPixel 함수는 출력할 픽셀좌표(x, y)와 색상 값을 인자로 받아 해당 좌표에 지정된 색상의 픽셀을 출력하는 함수이다. 다음은 TFT_PutPixel 함수의 의사코드를 보여준다.

```
function TFT_PutPixel(x, y, color)
begin
  address = FRAMEBUFFER_Start_Address + (x+y*
            Virtual_Screen_Horizontal_Size)*2      (1)
  memory of address = color
end
```

프레임버퍼와 LCD 패널의 좌표체계는 서로 다르기 때문에 인자로 받은 LCD 패널의 X/Y 직교좌표를 LCD 패널에 대응되는 프레임버퍼의 선형적인 주소의 좌표로 변환해 주어야 한다.

가상스크린 프레임버퍼는 수평, 수직의 크기 설정에 따라 수평크기*수직크기 개의 픽셀을 저장할 수 있으며, X/Y 직교좌표의 픽셀은 식 1의 x+y*Virtual_Screen_Horizontal_Size와 같은 공식으로 가상스크린 프레임버퍼에 몇 번째로 적재될 픽셀인지 알 수 있고, 한 픽셀 당 2 바이트 크기의 색상 값을 가지므로 2를 곱하면 픽셀좌표에 대응되는 가상스크린 프레임버퍼의 주소가 계산된다. 이 주소에 인자로 입력받은 색상 값을 저장하면, 해당 위치에 대응되는 픽셀이 출력된다[7],[8].

본 논문에서는 픽셀 색상 값으로 16 bpp (5:6:5)을 사용한다. 따라서 일반적으로 사용되는 24 bpp (R(8):G(8):B(8)) 색상 값을 16 bpp (5:6:5 형식)의 색상 값으로 변환하여 사용해야 하며, 그 방식은 표 2와 같이 붉은색은 상위 5 비트, 녹색은 상위 6 비트, 푸른색은 상위 5 비트만 추출하여 5:6:5 형식의 16비트 색상 값을 만든다.

표 2. 16 bpp(5:6:5 형식) 색상
Table 2. 16bpp(5:6:5 format) color

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
R7	R6	R5	R4	R3	G7	G6	G5	G4	G3	G2	B7	B6	B5	B4	B3

V. 가상스크린을 이용한 디스플레이

다음은 가상스크린 초기화 함수의 의사코드이다. 가상스크린의 넓이는 그림 4와 같이 PAGEWIDTH와 OFFSIZE로 구성되며, 이 값들을 설정함으로서, 가상스크린의 전체크기를 설정할 수 있다.

```
VirtualScreenInit(virtual_screen_width, lcd_width)
begin
  OFFSIZE = virtual_screen_width - lcd_width
  PAGEWIDTH = lcd_width
end
```

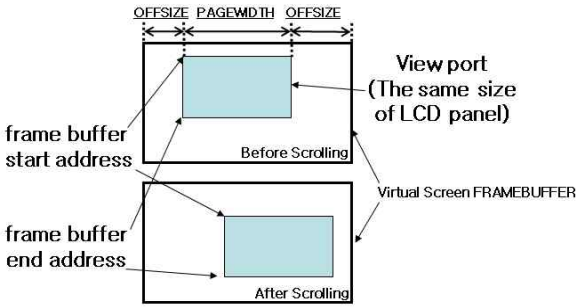


그림 4. 가상스크린의 수평, 수직 스크롤 기능
Fig. 4. horizontal or vertical scrolling of the virtual screen

S3C2440A LCD 컨트롤러는 가상스크린의 수평, 수직 스크롤 기능을 하드웨어적으로 지원한다. 가상스크린 내의 특정영역의 디스플레이는 그 영역을 뷰포트 영역으로 지정함으로써 행해진다. 뷰포트의 시작 주소와 끝 주소는 프레임버퍼의 시작 주소와 끝 주소를 의미하며, 실제 LCD 화면보다 큰 이미지를 가상스크린 프레임버퍼에 저장하고 뷰포트의 시작과 끝 주소를 연속으로 변화시킴으로, 이미지를 스크롤하며 출력시킬 수 있다.

다음은 뷰포트 영역을 지정하는 함수의 의사코드이다. 뷰포트의 전체 크기는 실제 LCD 화면의 크기와 동일함으로, 항상 고정 되어 있다. 따라서, 뷰포트의 끝 주소는 식 2를 이용하여 구할 수 있다.

```
ViewPortSetting(framebuffer_start_address)
begin
    framebuffer_end_address =
    framebuffer_start_address + (PAGEWIDTH +
    OFFSIZE) x lcd_height                          (2)
end
```

또한, 이 기능을 사용함으로써 그림 5와 같이 임베디드 시스템에서 수행되는 여러 태스크들이 자신의 수행 결과를 생성하고 그 결과를 각기 화면에 출

력해야하는 경우 각 태스크들은 가상스크린 프레임 버퍼 중 자신의 영역에 수행 결과를 저장하고, 필요시 해당 영역을 출력함으로써 유동적인 화면 디스플레이가 가능 하다.

특히 각 태스크의 수행 결과 데이터가 모두 변경되지 않고 일부분만을 변경되어 디스플레이 되는 경우에 더욱 유용하다. 이런 임베디드 시스템의 경우, 각 태스크들이 가상스크린의 메모리 영역을 효율적으로 나누어 사용할 경우 디스플레이 처리로 인한 오버헤드를 최소화 할 수 있다.

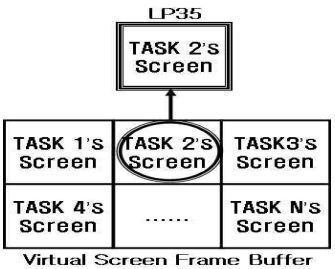


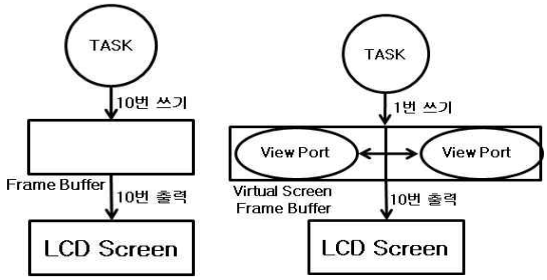
그림 5. 가상스크린을 이용한 디스플레이
Fig. 5. display using the virtual screen

VI. 성 능 평 가

성능 평가는 터치스크린 모듈인 LP35가 부착된 LN2440SBC 임베디드 보드에서 측정한다. 코어의 속도는 최고속도인 405MHz로 설정하고, 측정은 PWM(pulse width modulation) 타이머를 이용하며, 타이머 클럭을 약 130.9KHz로 설정하여, 1틱 당 7.585us의 시간이 소요 되도록 하였다.

측정은 현재 화면과 전 화면이 번갈아 가며 출력된다는 상황을 가정하여, 녹색 화면과 빨간색 화면을 번갈아 가며 출력시켜 총 10번의 화면전환이 일어나도록 한 후, 각각의 출력 소요시간을 측정하였다.

가상스크린을 사용하지 않는 경우, 그림 6의 (a)와 같이 화면전환을 할 때마다 프레임버퍼에 데이터를 적재해야 하므로 총 10번의 프레임버퍼의 쓰기 작업을 실행하고, 가상스크린을 사용하는 경우, 그림 6의 (b)와 같이 LCD 화면의 2배 크기인 640*240 크기의 이미지를 미리 가상스크린 프레임버퍼에 저장한 후, 뷰포트를 이동하여 10번의 화면 전환이 이루어지도록 하였다.



(a) 가상스크린 미사용 (b) 가상스크린 사용
그림 6. 성능평가 방법
Fig. 6. method of the performance estimation

그 결과, 두 경우의 출력 소요시간은 그림 7과 같이 가상스크린을 사용한 경우 약 5배의 출력속도의 증가를 보였다. 이 중 출력에 소요되는 시간의 99%는 데이터를 가상스크린 프레임버퍼에 기록하는 시간이고, 뷰포트를 이동하여 화면 전환을 수행하는 시간은 1%에 불과 하였다. 이러한 출력 소요시간의 감소는 표 3에서 보이는 바와 같이, 프레임 버퍼에 적재된 데이터의 총 크기를 감소시킬 수 있기 때문이다.

표 3. 프레임 버퍼의 데이터 적재 횟수 및 메모리 크기 비교
Table 3. comparison of the frequency and memory size for loading data to frame buffer

구 분	프레임버퍼의 데이터 적재 회수 및 총 크기	뷰포트 이동 회수
가상스크린 사용	10 번(1.536MB)	0 번
가상스크린 미사용	1 번(0.3072MB)	9 번

따라서 이것은 여러 태스크들이 자신의 수행 결과를 각기 화면에 출력하는 경우, 각 태스크들이 가상스크린의 메모리 영역을 효율적으로 나누어 유휴시간에 자신의 영역에 이미지를 저장하고, 뷰포트를 자신의 영역으로 이동시킴으로써 출력속도를 증가시킬 수 있다.

또한, 각 태스크의 수행 결과 데이터가 모두 변경되지 않고 일부분만 변경되어 디스플레이 되는 경우나 전 화면을 다시 디스플레이 하는 경우와 같이 중복된 화면 출력이 빈번한 경우, 출력 소요시간을 상당히 줄여, 디스플레이 처리로 인한 오버헤드를 최소화 할 수 있음을 보인다.

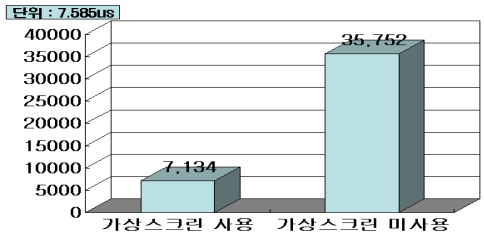


그림 7. 전체 출력 시간 비교
Fig. 7. comparison of the total display time

VII. 결론 및 향후계획

본 논문에서는 LN2440SBC 임베디드 시스템에 터치스크린 기능을 가진 LP35 TFT LCD 모듈을 부착하여, 현재 휴대용 기기들의 입출력 장치로 각광 받는 TFT LCD의 초기화 방법과 LCD 출력을 위한 픽셀 디스플레이 함수 그리고 가상스크린을 이용한 LCD 디스플레이 방식에 대해 설명 했다. 또한 가상스크린을 이용한 디스플레이 방식과 사용하지 않은 방식의 출력 속도를 비교한 결과 중복된 화면 출력이 빈번한 경우 출력 소요시간을 비약적으로 줄여, 디스플레이 처리로 인한 오버헤드를 최소화 할 수 있음을 보였다.

향후에는 TFT LCD와 함께 입력 장치로 각광받고 있는 터치스크린과 이미지, 선, 도형들을 그리는 기본적인 2D 그래픽스 라이브러리 함수[8]를 추가하여 보다. 그래픽 적으로 보여 질 수 있는 사용자 인터페이스를 구현할 예정이다.

감사의 글

본 논문은 2009년도 호서대학교 재원으로 학술연구비 지원을 받아 수행된 연구임.

참 고 문 헌

[1] 이희문, PIC를 이용한 LCD 모듈 활용, *성안당* pp. 295-315, 2006.
[2] 홍건표, 하동호, ARM920T S3C24101, S3C2440A를 이용한 개발자들을 위한 ARM 프로세서, OHM, pp. 116-489, 2006.
[3] 오삼권, "uC/OS-II를 탑재한 S3C2440A 싱글 보드 컴퓨터를 위한 모니터링 시스템의 구현", *호서대학교 논문집, 제26권*, pp. 63-69, 2007.
[4] Samsung Electronics, S3C2440A 32-BIT CMOS Microcontroller User's Manual Revision 1, Samsung Electronics, pp. 30-509, 2004.
[5] Samsung Electronics, Product Information LTV350QV-F04, Samsung Electronics, pp. 1-30, 2005.
[6] John Catsoulis, Designing Embedded Hardware, 2nd ED의 역서, 한빛미디어, pp. 97-212, 2007.
[7] 오삼권, 박근덕, 김병국, "임베디드 시스템을 위한 한글 포팅 및 출력 성능 비교", *한국 디지털 콘텐츠 학회 논문지, 제10권 제4호*, pp. 493-499, 2009. 12.
[8] Donald Hearn, M.Pauline Baker, Computer Graphics의 역서, 아진, pp. 85-159, 2002.

오 삼 권 (吳三權)

1994년 : Queen's University (Ca nada)(컴퓨터과학박사)
1995년 3월 ~ 현재 : 호서대학교 컴퓨터공학부 교수
관심분야 : Embedded Real-Time Systems, OS, Communication Protocol, Fault Tolerance

박 근 덕 (朴根德)

2005년 8월 : 서울대학교 전기컴퓨터공학부 (공학박사)
2006년 3월 ~ 현재 : 호서대학교 컴퓨터공학부 조교수
관심분야 : 임베디드소프트웨어공학, 서비스 지향 컴퓨팅, XML 응용

김 병 국 (金炳椆)

2009년 2월 : 호서대학교 컴퓨터공학과(공학사)
2009년 3월~현재 : 호서대학교 일반대학원 컴퓨터공학과 (컴퓨터공학 석사과정)
관심분야 : Embedded System, Real Time Operating System