임베디드 시스템

<DE1-SOC와 VGA를 이용한 비디오 시스템과 캐치 게임>

2012253066 김기현

2012252044 김성찬

1. 초기 목표

초기 목표는 VGA 케이블과 모니터의 연결을 통한 비디오 모니터를 띄우는 것이었고, 그 다음은 여러 인터럽트(ex. Switch, Button)를 통하여 동작들을 만들어주는 것으로 하였다.

2. 프로그램 설명

먼저 모니터에 백그라운드 배경이 될 수 있는 color를 전체적으로 채워준다. 그 후에, 간단한 게임 제작자, 프로그램 이름, 게임 설명 등이 화면에 뜬 후, '프로그램을 실행하려면 아무 버튼이나 누르세요'라는 문구가 뜬다. 아무 버튼이나 누를 시 게임이 시작되며 target인 직사각형 모양이 랜덤으로 위치배정이 되고 user인 또 다른 같은 크기의 직사각형이 화면 정가운데에 배치된다. button을 이용하여 좌,우,상,하로 움직일 수 있으며 switch를 이용하여 이동거리를 늘릴 수 있다(2,5,10). 정확히 같은 위치에 직사각형을 옮겼을 시 clear화면이 뜬다. 타이머를 이용하여 제한시간 30초가 걸려져있으며 이 시간이 지나면 game over 문구가 뜨고 프로그램이 종료된다.

3. Flow Chart

A. 초기 설정(헤더 파일 설정, define 정의, 전역변수 설정)

B. Open함수를 이용한 /dev/mem 디바이스 장치 오픈

C. Mmap함수를 이용하여 가상 주소 공간에 물리 주소를 매핑

D. 함수 정의(VGA_text, VGA_text_clear, VGA box)

E. 초기 게임 배경 화면 구현 및 프로그램 설명

F. 제한 시간을 위해 타이머 설정(MPcore Private Timer)

G. Button과 switch를 이용한 게임 구현

4. 중요 코드 설명

```
    // Open /dev/mem
        if( ( fd = open( "/dev/mem", ( O_RDWR | O_SYNC ) ) ) == -1 )    {
                printf( "ERROR: could not open "/dev/mem"...\n" );
                return( 1 );
        }

    // get virtual addr that maps to physical
        h2p_lw_virtual_base = mmap( NULL, HW_REGS_SPAN, ( PROT_READ | PROT_WRITE
), MAP_SHARED, fd, HW_REGS_BASE );
        if( h2p_lw_virtual_base == MAP_FAILED ) {
                printf( "ERROR: mmap1() failed...\n" );
                close( fd );
                return(1);
        }

        //get timer virtual addr that maps to physical
        timer_virtual_base = mmap(NULL, MPcore_REGS_SPAN, ( PROT_READ | PROT_WRI
TE), MAP_SHARED, fd, MPcore_REGS_BASE);
        if( timer_virtual_base == MAP_FAILED ) {
                printf( "ERROR: mmap1() failed...\n" );
                close( fd );
                return(1);
        }
```

```
        // get virtual addr that maps to physical
        vga_char_virtual_base = mmap( NULL, FPGA_CHAR_SPAN, (   PROT_READ | PROT
_WRITE ), MAP_SHARED, fd, FPGA_CHAR_BASE );
        if( vga_char_virtual_base == MAP_FAILED ) {
                printf( "ERROR: mmap2() failed...\n" );
                close( fd );
                return(1);
        }
```

/dev/mem 디바이스 장치를 열고 mmap을 사용하여 가상 메모리 주소 공간에 /dev/mem에 있는 물리 주소를 매핑하는 코드이다. h2p_lw_virtual_base는 switch와 button을 위한 bridge이고 timer_virtual_base는 MPcore_prviate_timer를 위한 bridge이며 마지막으로 vga_char_virtual_base는 vga_char를 위한 bridge이다.

```
srand(time(NULL));      //seed
x=0;
y=0;
x=(rand()%580)+0;       //0~580
y=(rand()%190)+0;       //0~190
//start poing is always middle
x2=290;
y2=100;
// clear the screen
VGA_box (0, 0, 639, 479, gray);
// clear the text
VGA_text_clear();
// write header text
VGA_text (1, 5, text_top_row);
VGA_text (1, 10, text_next);
VGA_text (1, 15, text_manual);
VGA_text (15,20, start);
while(begin==0){
VGA_text_clear();
if(*(KEY_ptr+3)!=0){
begin=1;
// text box
VGA_box(x, y, x+40, y+40, 0xe0); //target box(red) size is 40x40
VGA_box(x2, y2, x2+40, y2+40,0x04 ); //start box(green)
//it always is built in middle
```

Screen을 한 색으로 모두 채운 후(모니터 clear) VGA_TEXT함수를 이용하여 적절한 위치에 위에 선언해 둔 문장들을 모니터에 출력해준다. 그 후, 버튼이 눌릴 때까지 계속 루프를 돌고, 눌렸을 시, VGA_box함수를 이용해 랜덤함수들을 이용하여 x와 y의 값이 정해진 target(크기는 40)과 정중앙에 표시할 user 직사각형을 출력한다.

```
//set timer
unsigned long counter = 30UL * 200000000UL;    //counter seconds
*(MPcore_private_timer_ptr) = counter;
*(MPcore_private_timer_ptr + 2) = 0b011;


*(KEY_ptr+3) = 0xf;

while(catch==0){
    if(*(MPcore_private_timer_ptr + 3) == 1){
        VGA_text(25,15, fail);
        break;
        }
        if(*(SW_ptr)==0x1)
                level=2;
        else if(*(SW_ptr)==0x2)
                level=5;
        else if(*(SW_ptr)==0x4)
                level=10;
        else{
                level=1;
        }
```

타이머의 설정과 타이머가 끝났을 시, fail이라는 문장이 화면에 표시되고 프로그램이 끝나는 코드와 Switch를 이용하여 이동거리를 바꾸는 코드이다. 0번째 스위치는 2, 1번째는 5, 2번째는 10, 그 외에는 1로 이동거리가 설정된다.

```
while(*(KEY_ptr+3)!=0){//when you push the button
        if(*(KEY_ptr+3)==0x1){//pressed key0
                if(x2<=0)
                        break;
                x2+=level;
        VGA_box (0, 0, 639, 479, gray);//clear the page
            VGA_box (x, y, x+40, y+40, 0xe0);//restore the page
        VGA_box (x2, y2, x2+40, y2+40, 0x04);//update present st

        }else if(*(KEY_ptr+3)==0x2){//pressed key1
                if(y2<=0)
                        break;
                y2-=level;
        VGA_box (0, 0, 639, 479, gray);//clear the page
            VGA_box (x, y, x+40, y+40, 0xe0);//restore the page
        VGA_box (x2, y2, x2+40, y2+40, 0x04);//update present st

        }else if(*(KEY_ptr+3)==0x4){//pressed key2
               if(y2+40>=240)
                        break;
                y2+=level;
        VGA_box (0, 0, 639, 479, gray);//clear the page
            VGA_box (x, y, x+40, y+40, 0xe0);//restore the page
        VGA_box (x2, y2, x2+40, y2+40, 0x04);//update present st

        }else{  //pressed key3
                if(x2+40>=630)
                        break;
               x2-=level;
       VGA_box (0, 0, 639, 479, gray);//clear the page
            VGA_box (x, y, x+40, y+40, 0xe0);//restore the page
        VGA_box (x2, y2, x2+40, y2+40, 0x04);//update present st

        }
        *(KEY_ptr+3)=0xF;  //initialize edge capture register
}
if(x==x2 && y==y2){
        VGA_box (0, 0, 639, 479, gray);//clear the page
        VGA_text(25,15, success);
        catch = 1;
}
```

좌, 우, 상, 하로 버튼을 이용하여 이동하는 코드이다. 직사각형이 모니터 화면을 벗어나지 못하게 조건식이 달려있고, 이동하면 일단 모든 페이지를 원래 기본 화면(아무 모양도 있지 않은)으로 바꾼 후, target과 이동한 user를 출력해준다. 그리고 꼭 동작이 끝난 후, edge capture register를 초기화 해주어 버튼 상태를 되돌린다. 만약 target의 x, y와 user의 x2, y2가 일치할 시에는 성공 메시지를 출력하고 프로그램을 종료한다.

```
void VGA_text(int x, int y, char * text_ptr)
{
        volatile char * character_buffer = (char *) vga_char_ptr ;      // VGA c
haracter buffer
        int offset;
        offset = (y << 7) + x;
        while ( *(text_ptr) )
        {
                // write to the character buffer
                *(character_buffer + offset) = *(text_ptr);
                ++text_ptr;
               ++offset;
        }
}
```

모니터에 텍스트를 출력하는 함수이다. 응용 프로그램에서 x, y와 출력할 문장을 받아오고 x, y로 offset을 설정 후, while문을 이용하여 한 글자 씩 출력해준다.

```
void VGA_text_clear()
{
        volatile char * character_buffer = (char *) vga_char_ptr ;      // VGA c
haracter buffer
        int offset, x, y;
        for (x=0; x<79; x++){
                for (y=0; y<59; y++){
        /* assume that the text string fits on one line */
                        offset = (y << 7) + x;
                        // write to the character buffer
                        *(character_buffer + offset) = ' ';
                }
        }
}
```

모니터에 표시된 텍스트를 모두 삭제하는 함수이다. x와 y를 처음부터 끝까지 for 문으로 탐색하면서 모두 ' '로 채우는 함수이다.

```
#define SWAP(X,Y) do{int temp=X; X=Y; Y=temp;}while(0)

void VGA_box(int x1, int y1, int x2, int y2, short pixel_color)
{
        char  *pixel_ptr ;
        int row, col;

        /* check and fix box coordinates to be valid */
        if (x1>639) x1 = 639;
        if (y1>479) y1 = 479;
        if (x2>639) x2 = 639;
        if (y2>479) y2 = 479;
        if (x1<0) x1 = 0;
        if (y1<0) y1 = 0;
        if (x2<0) x2 = 0;
        if (y2<0) y2 = 0;
        if (x1>x2) SWAP(x1,x2);
        if (y1>y2) SWAP(y1,y2);
        for (row = y1; row <= y2; row++)
                for (col = x1; col <= x2; ++col)
                {
                        //640x480
                        pixel_ptr = (char *)vga_pixel_ptr + (row<<10)    + col ;
                        // set pixel color
                        *(char *)pixel_ptr = pixel_color;
                }
}
```

Define을 이용하여 swap함수를 설정하였고, 직사각형 모양을 출력해주는 함수이다. 직사각형이 화면을 밖으로 넘어갈 경우, 적절한 값으로 바꾸게 하였으며 x2, y2가 x1, y1보다 클 경우 서로의 값을 switch하게 만들었다(식을 간단한게 하기 위해). 행을 기준으로 열을 color로 채워가며 결국에는 한 줄이 색으로 채워지게 되며 이것을 행마다 반복해 color가 채워진 직사각형 모양을 출력하는 것이다.

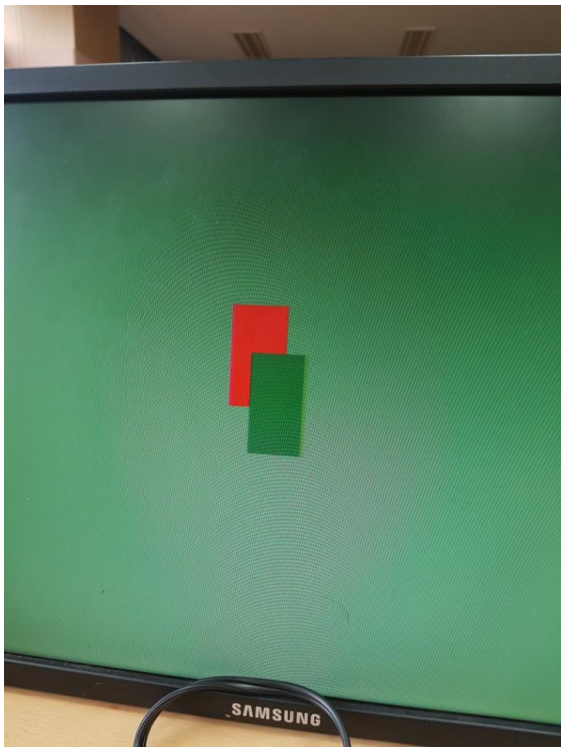
5. 개선 가능성

1)인터럽트를 이용하여 button을 누른 채로 있으면 계속 모형이 움직이는 동작이 가능할 것이다.

2)레벨을 만들어 단계별로 target의 수 증가, 시간제한 조절을 통해 난이도 조절을 할 수 있을 것이다.

3)조건식을 이용하여 target을 잡을 때마다 자신의 크기를 늘리고 자신보다 작은 target만 잡을 수 있고 큰 target은 닿을 시 게임오버가 되는 방향의 게임 접근이 가능할 것이다.

6. 결과



7. Discussion

처음에는 DE1-SOC를 이용한 비디오 프로그램을 만들어본 적이 없어서 많이 헤매었다. 교수님이 알려주신 altera 사이트에 있는 예제 프로그램을 먼저 참고해보았는데 몇 몇 코드가 빠져있어서 이해하기가 조금 어려웠다. 그래서 인터넷에 예제 프로그램을 찾아보고 vga 비디오 시스템에 대해서도 알아보았다. 결국 한 예제 프로그램을 찾아내었고 그 프로그램을 참고해가면 만들게 되었다. 다 만들고 나서 처음 시행을 하였을 때, 모니터에 뜨긴 뜨는데 온통 tv가 망가졌을 때 나오는 지지직 같은 모양으로 화면이 깨져서 나왔다. 처음에는 프로그램이 잘못되어서 그런 줄 알았는데 모니터를 바꾸어서 실행해 보니 옆 친구 모니터가 어떤 이유로 인해 rgb표현이 제대로 되지 않는 모양이었다. 또 처음 버튼으로 이동하는 동작을 구현하였을 때, 버튼을 한 번만 눌러도 그 방향으로 계속 움직여버리게 오류가 났는데 이것은 edge capture register를 초기화하는 것을 하지 않아서 생긴 오류였다. 초기화를 추가해줌으로써 해결하였다. 초기 비디오 구현에 시간을 너무 많이 소비하지 않았더라면 level 구현과 새로운 동작들을 더 추가할 수 있었을 텐데라는 안타까움이 든다.