

GameBoy-Periféricos LCD y GPIO

Sebastián Cortés, Luis Carlos Díaz

February 2019

1 Hardware

1.1 LCD parallel

La función de este modulo es escribir en los registros de la pantalla, siguiendo el diagrama de tiempos mostrado en la figura 1.

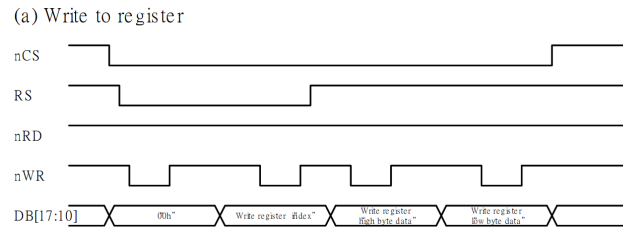


Figure 1: Diagrama de tiempos para la escritura en los registros de la LCD.

Para acceder a la Ram de gráficos se sigue otro diagrama de tiempos, mostrado en la figura 2, este secuencia se identifica con el comando 0x3c, y recibe N argumentos, correspondientes a el color de cada pixel.

En las figuras 3 y 4 se muestra el funcionamiento del modulo Lcd parallel en el analizador lógico

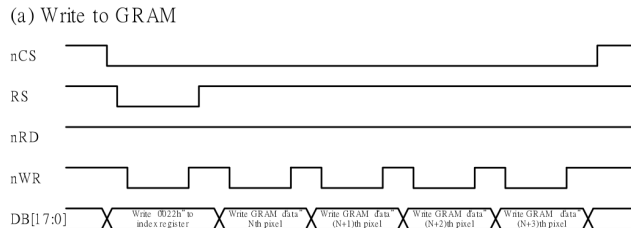


Figure 2: Diagrama de tiempos para la escritura en la Ram de gráficos.

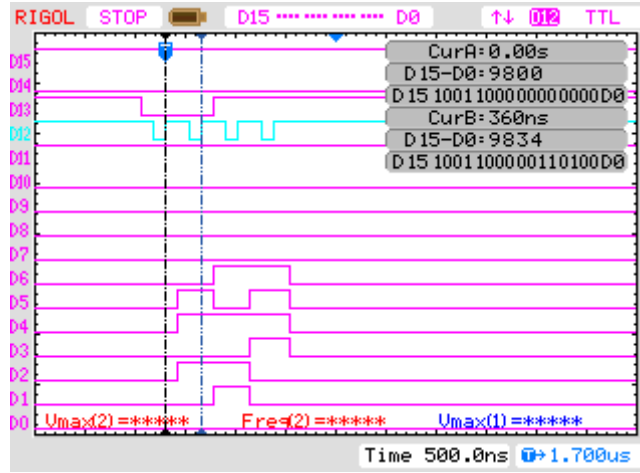


Figure 3: Escritura de datos.

En la tabla 1 se muestra el mapa de memoria del periférico encargado de comunicarse con la pantalla LCD.

Registro	Dirección	Tamaño	Tipo
lcd_test_DATA	0xe0005800	16	rw
lcd_test_NUMBER	0xe0005808	32	rw
lcd_test_ADDR	0xe0005818	8	rw
lcd_test_BUSY	0xe000581c	1	ro
lcd_test_START	0xe0005820	2	rw
lcd_test_OPTION	0xe0005824	3	rw
lcd_test_CS_	0xe0005828	1	rw

Table 1: Mapa de memoria de lcd_core.

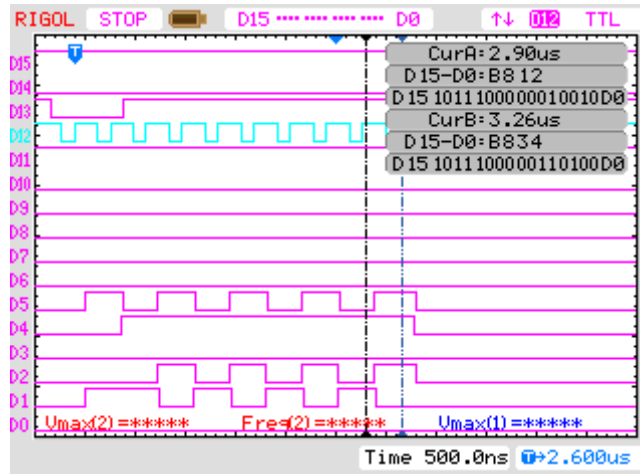


Figure 4: Escritura en la ram de gráficos

1.2 GPIO

Para leer el estado de los botones se uso un periférico GPIO que almacena el estado de los botones en un registro, el mapa de memoria se muestra en la ba

Registro	Dirección	Tamaño	Tipo
buttons_in	0xe0006800	1	ro

Table 2: Mapa de memoria del GPIO

2 Software

2.1 LCD

Para escribir en los registros de la pantalla haciendo uso de los registros de control se usa la función *lcd_write_reg*.

```
static void lcd_write_reg(unsigned char address,unsigned short int value,
unsigned char option){
    lcd_test_ADDR_write(address);    //Escritura de la direccion
    lcd_test_DATA_write(value);      //Escritura del dato
    lcd_test_SENDONE_write(option);  //Escritura de la direccion
    lcd_test_START_write(1);          //Inicio del ciclo de escritura
    lcd_test_START_write(0);
    while(lcd_test_BUSY_read());      //Esperar al fin del proceso
}
```

Haciendo uso de esta función se hace la secuencia de inicialización que consiste en escribir los registros mostrados a continuación.

Reg	Data	Descripcion
0x01	0x0000	Software rest
0x28	0x0000	Display off
0xc0	0x0023	Power control 1
0xc1	0x0010	Power control 2
0xc5	0x2b2b	Vcom Control
0xc7	0x00c0	Vcom Control 2
0x36	0x0088	Memory access control
0x3a	0x0055	Pixel format set
0xb1	0x001b	Frame control
0xb7	0x0007	Entry mode set
0x11	0x0000	Sleep out
0x29	0x0000	Display on

Table 3: Secuencia de inicialización.

Luego de la inicialización se procede a escribir en cada posición de la memoria de gráficos para esto se implementó la función *lcd_write_ntimes*, esta función ejecuta el comando 0x3e y envía el dato n veces.

```
static void lcd_write_ntimes(unsigned short int value ,
unsigned int times){
    lcd_test_ADDR_write(0x3c);
    lcd_test_DATA_write(value);
    lcd_test_NUMBER_write(times);
    lcd_test_SENDONE_write(0);
    lcd_test_START_write(1);
    lcd_test_START_write(0);
    while(lcd_test_BUSY_read());
}
```

2.2 GPIO

Para leer los registros correspondientes a los botones se uso la función *buttons_in_read*, generada por Migen

```
static inline unsigned char buttons_in_read(void) {
    unsigned char r = csr_readl(0xe0006800);
    return r;
}
```

References

- [1] Ilitek, "TFT LCD Single Chip Driver", ili9341 datasheet, [Revisado Feb 2019]