

Ingeniería inversa a pantallas LCD

Alejandro Oquendo

April 21, 2023

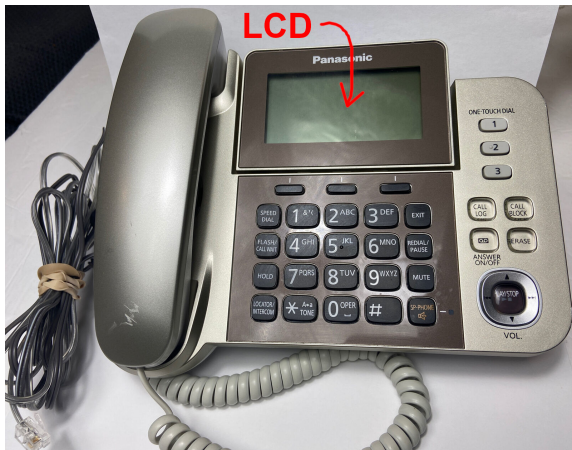
Introducción

- ▶ "La ingeniería inversa es el proceso llevado a cabo con el objetivo de obtener información o un diseño a partir de un producto, con el fin de determinar cuáles son sus componentes y de qué manera interactúan entre sí y cuál fue el proceso de fabricación."
- ▶ "Una pantalla de cristal líquido o LCD (sigla del inglés liquid-crystal display) es una pantalla delgada y plana formada por un número de píxeles en color o monocromos colocados delante de una fuente de luz o reflectora."
- ▶ Esta charla habla acerca del proceso que se hizo para poder controlar un LCD desde una placa compatible con Arduino
- ▶ No se tienen ningún dato acerca de como funciona el LCD, esto es tomado como un reto
- ▶ Esta presentación estuvo a punto de no ser realizada

Propósito

- ▶ Aprender nuevas maneras de ver las cosas
- ▶ Reutilizar componentes electrónicos de alguna manera iban a ser desechados
- ▶ Desarrollar nuevas habilidades
- ▶ Un LCD es un solo un tipo de componente de la infinidad de componentes que se pueden aprovechar

Fuente de componentes



Fuente de componentes

- ▶ Teléfono que no le funcionaba el teclado
- ▶ Pantalla LCD de matriz de puntos

Análisis de placa

- ▶ PCB de teclado roto, se podía reparar pero utilizar la pantalla tiene mas valor en mi caso
- ▶ Pantalla LCD con un conector de 8 pines
- ▶ Algo parecido a un SoC para controlar el funcionamiento del teléfono

Análisis de LCD

- ▶ La pantalla mostraba un modelo, pero despues de buscar no se encontraron ninguna especificación
- ▶ Tiene luz de fondo, pixels oscuros, monocromático
- ▶ Cable flex con 9 pines conectado a la placa principal

Averiguar el pinout

- ▶ Normalmente se tienen 2 pines de alimentación: VCC y GND
- ▶ En este caso se tiene un pin extra para alimentación de backlight
- ▶ El resto de pines eran 5, que debían ser de datos

Averiguar el tipo de bus de datos

- ▶ Las pantallas LCD tienen distintos tipos de buses de datos
- ▶ Estos pueden ser paralelos o seriales
- ▶ En este caso podría ser uno de tipo serial por la cantidad reducida de pines (5)

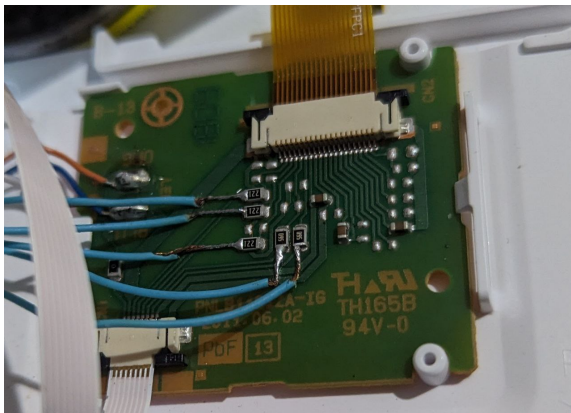
Averiguar el tipo de bus de datos

- ▶ Para confirmar se podría hacer un análisis de como está conectado este bus con los demás componentes
- ▶ Utilizando un multímetro se evidencia que solo existe una conexión directa entre el LCD y el SoC
- ▶ Entonces se podría descartar que use I²C o SPI

Averiguar el tipo de bus de datos

- ▶ Para averiguar mas detalles se necesita analizar los datos que circulan por esos 5 pines
- ▶ Se decidió conectar un analizador lógico al bus (Saleae)
- ▶ Los pines son muy delgados, entonces se suelda en los terminales de unas resistencias SMD

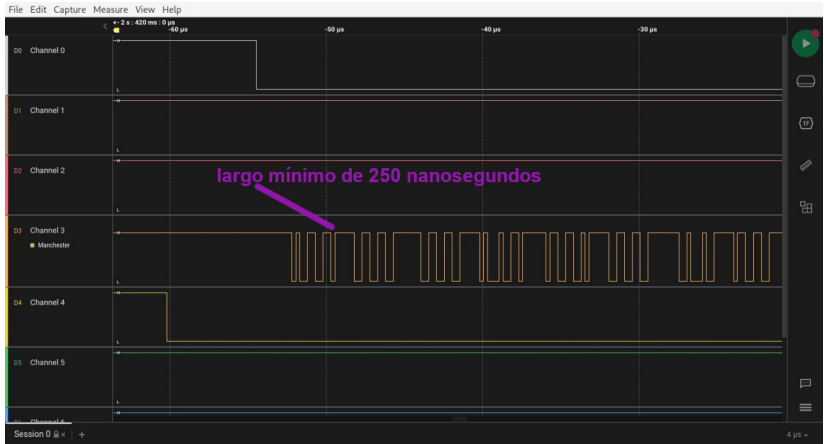
Conexión a bus de datos



Captura de datos

- ▶ Una vez teniendo conectado el bus al analizador lógico, este se conecta al puerto USB
- ▶ Se inicia el software Logic de Saleae con un trigger en uno de los pines para que inicie la captura
- ▶ Se enciende el teléfono para que empiece el proceso

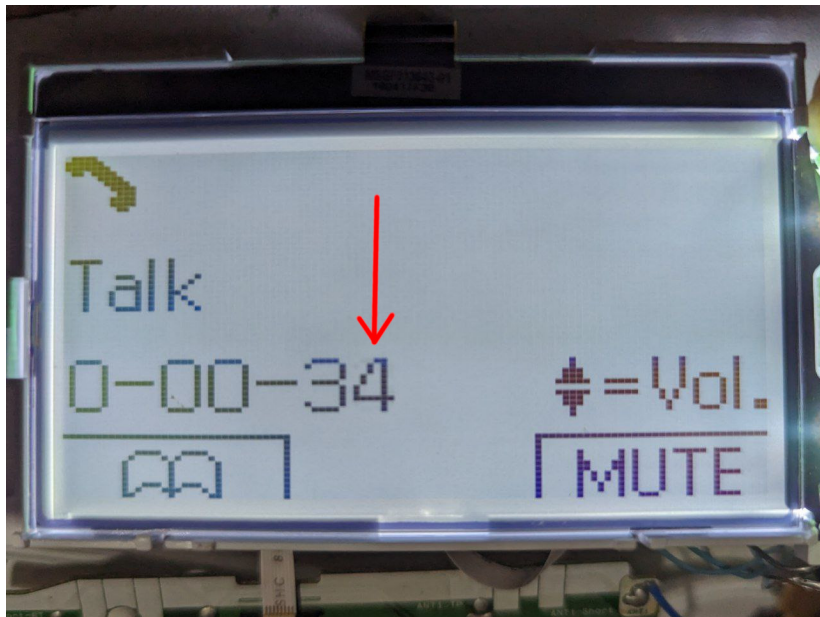
Primera captura en Logic



Análisis de primera captura

- ▶ Se observa un ancho de pulso mínimo de 250 nanosegundos
- ▶ De la captura, de aproximadamente 5 segundos, se observa que al final existe un grupo de datos cada 1 segundo
- ▶ Esto corresponde a un timer que muestra el teléfono cuando esta descolgado

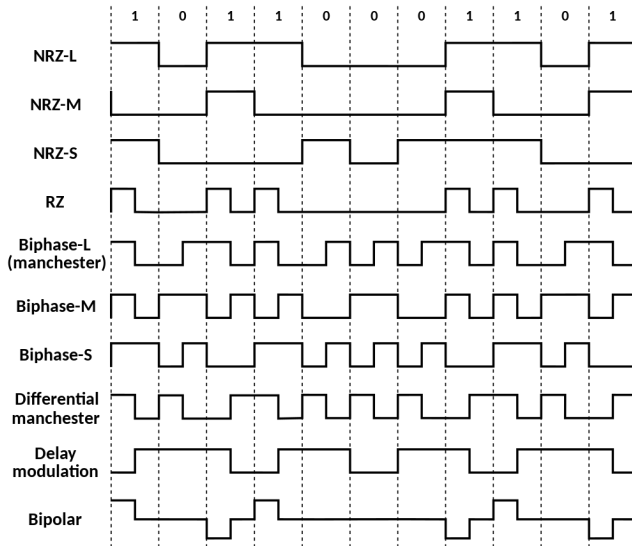
Timer



Análisis de primera captura

- ▶ El canal 3 parece ser de datos
- ▶ Normalmente los primeros datos en ser enviados son los de configuración del LCD
- ▶ Los datos mandados cada segundo son directamente datos de los pixeles, entonces convendría concentrarse en estos
- ▶ El problema con esta captura es que no se tiene un clock, entonces tal vez podría ser el bus no cumple con un estandar conocido
- ▶ Se intentó usar una codificación conocida, la que mas coincidía era la Manchester

Line codes



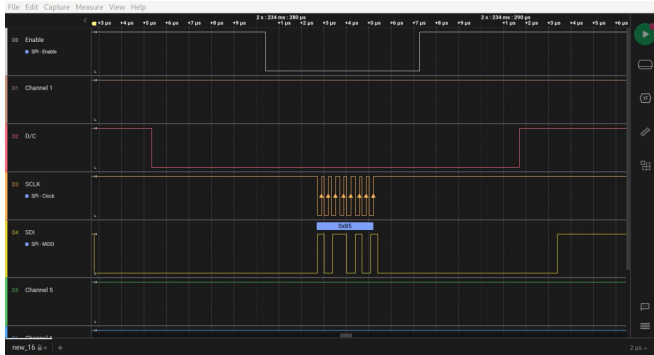
Análisis de primera captura

- ▶ Se exportó los datos a un .csv para comparar los datos de pixeles
- ▶ Se encontraron muchas diferencias, en teoría solo deberían haber diferencias de unos pixeles
- ▶ No había orden establecido, algo estaba fallando en la captura
- ▶ Despues de varios días de revisar la captura, se descubrió que la velocidad de captura no era suficiente

Segunda captura de datos

- ▶ El analizador lógico permite capturar datos a más velocidad, pero menos cantidad de tiempo
- ▶ Lo primero que se nota es que el canal 3 en realidad es el clock
- ▶ El canal 4 posiblemente es de datos
- ▶ Los otros canales son posiblemente de control

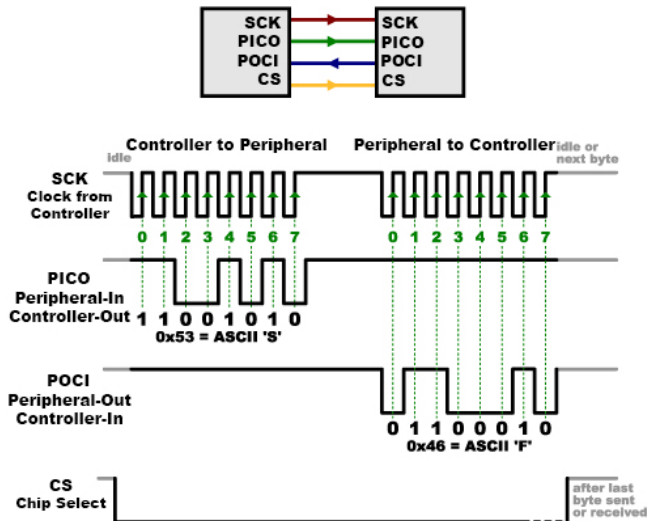
Canales identificados



Análisis de la segunda captura

- ▶ Se nota que el canal 3 y 4 posiblemente cumplen con el protocolo SPI
- ▶ Se hace una revisión de SPI, si corresponden los otros canales
- ▶ Se configura Logic para que decodifique los datos
- ▶ Luego de varios intentos se logra hacer la decodificación y es posible exportar los datos decodificados
- ▶ Se hace una comparación entre los datos de pixeles, se nota que existe cierto orden y algunos datos coinciden

Protocolo SPI



Análisis de los datos

- ▶ Se nota un patrón en los datos, existen 3 bytes que se repiten
- ▶ Estos se repiten cada vez que se envía un lote largo, posiblemente sean de configuración
- ▶ Se realiza una búsqueda de estos 3 bytes y sorprendentemente se encuentra un foro
- ▶ En el post se encuentra un modelo de un LCD
- ▶ Se encuentra el datasheet del LCD

Busqueda de bytes

The screenshot shows a web browser window with a Google search for "0xB4 0x10 0x00 LCD". The search results are displayed on a dark background. The first result is from CCSinfo.com, titled "CCS :: View topic - Graphic LCD", with a snippet of C code: `bargraph_display(0xB1, 0x10, 0x00, 0xB6); bargraph_display(0xB2, 0x10, 0x00, 0x6D); bargraph_display(0xB3, 0x10, 0x00, 0xDB); bargraph_display(0xB4, 0x10 ...`. The second result is from GitHub, titled "M5Stick2Projects/Lcd_Driver.cpp at master", with a snippet: `//LCD Init For 1.44Inch LCD Panel with ST7735R: void Lcd_Init(void) ... Lcd_WriteIndex(0xB4); Lcd_WriteData(0x03); ... Lcd_WriteData(0x00);`. The third result is from Engineers Garage, titled "Mini-Project-1/LCD.c at master", with a snippet: `This is a library for the Adafruit 1.8" SPI display. This library works with the Adafruit 1.8" TFT Breakout w/SD card. ----> http://www ...`. The browser's address bar shows the search URL: `https://www.google.com/search?q=0xB4+0x10+0x00+LCD&client=firefox-b-d&sxsrf=APwXEddeTpb-PXj...`. The browser's tab bar shows several open tabs, including "CFOP Speeds", "Algorithms: 2", "0xB4 0x10", "CCS :: View to...", "untitled - dog...", "#define - Ardu...", "Hacker News", and "102x72 lcd - G...".

Google

0xB4 0x10 0x00 LCD

About 18,000 results (0.36 seconds)

CCSinfo.com
<https://www.ccsinfo.com/forum/viewtopic>

CCS :: View topic - Graphic LCD

`bargraph_display(0xB1, 0x10, 0x00, 0xB6); bargraph_display(0xB2, 0x10, 0x00, 0x6D);
bargraph_display(0xB3, 0x10, 0x00, 0xDB); bargraph_display(0xB4, 0x10 ...`

GitHub
<https://github.com/lirshuqn329/blob/master/src>

M5Stick2Projects/Lcd_Driver.cpp at master

`//LCD Init For 1.44Inch LCD Panel with ST7735R: void Lcd_Init(void) ...
Lcd_WriteIndex(0xB4); Lcd_WriteData(0x03); ... Lcd_WriteData(0x00);`

<https://github.com/Mini-Project-1/blob/LCD.c>

Mini-Project-1/LCD.c at master

This is a library for the Adafruit 1.8" SPI display. This library works with the Adafruit 1.8" TFT Breakout w/SD card. ----> [http://www ...](http://www...)

Engineers Garage
<https://www.engineersgarage.com/displaying-images...>

Displaying Images on Graphical Lcd(JHD12864E) using ...

Foro con datos de LCD

CFOF Speeds Algorithms: 2 Algorithms: 2 CCS: View x untitled - dog #define - Arduino Hacking News 10x7d lcd - C + +

[←](#) [→](#) [↺](#) [https://www.ccsinfo.com/forum/viewtopic.php?t=56396&postdays=D&postorder=asc&start=0](#)

[new topic](#) [post reply](#) CCS Forum Index → General C/C Discussion

[View previous topic :: View next topic](#)

| Author | Message |
|---------|--|
| hemnath | <p>Graphic LCD D Posted: Tue Jul 25, 2017 2:36 am</p> <p>I'm using PIC18F2520, Internal oscillator: 1MHz trying to interface with Graphic LCD: EA0GSM12W-K0 (128 x 64 DOTS).</p> <p>Below is the sample program:</p> <pre>Code: #include "18F2520.h" #fuses INTOSC_10, NOBODT, BROWNOUT, PUT, NOWLP // Internal oscillator #use delay(clock=1000000) const unsigned char numbers_22[10][48] = { (0x00, 0x00, 0x00, 0xC0, 0xFF, 0x00, 0xF0, 0xFF, 0x03, 0xFC, 0xFF, 0x0F, 0xFC, 0xFF, 0x3E, 0x00, 0x1F, 0x0E, 0x00, 0x1C, 0x0E, 0x00, 0x1C, 0x0E, 0x00, 0x1C, 0x3E, 0x00, 0x1F, 0xFC, 0xFF, 0x0F, 0xFC, 0xFF, 0x0F, 0xF0, 0xFF, 0x3E, 0xC0, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00), // 0 (0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xC0, 0x03, 0x00, 0xE0, 0x01, 0x00, 0xE0, 0x01, 0x00, 0xF0, 0x00, 0x00, 0x78, 0x00, 0x00, 0xFC, 0xFF, 0x1F, 0xFE, 0xFF, 0x1F, 0xFE, 0xFF, 0x1F, 0xFE, 0xFF, 0x1F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00), // 1 (0x00, 0x00, 0x00, 0x00, 0x00, 0x18, 0x78, 0x00, 0x1E, 0x7C, 0x80, 0x1F, 0x7C, 0xC0, 0x1F, 0x1E, 0xE0, 0x1F, 0x0E, 0xF0, 0x1D, 0x0E, 0xF8, 0x1C, 0x0E, 0x7C, 0x1C, 0x1E, 0x3E, 0x1C, 0xFE, 0x1F, 0x1C, 0xFC, 0x0F, 0x1C, 0xF8, 0x0F, 0x1C, 0xF8, 0x0F, 0x1C, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00), // 2 (0x00, 0x00, 0x00, 0x30, 0x00, 0x01, 0x38, 0x80, 0x07, 0x3C, 0x88, 0x0F, 0x3E, 0x80, 0x1C, 0x1E, 0x00, 0x1E, 0x0E, 0x07, 0x1C, 0x0E, 0x07, 0x1C, 0x9E, 0x07, 0x1C, 0xFE, 0x0F, 0x1E, 0xFC, 0xFF, 0x1F, 0xF8, 0xF0, 0x0F, 0xF0, 0xF8, 0x07, 0x00, 0xF0, 0x03, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00), // 3 (0x00, 0xF8, 0x01, 0x00, 0xFC, 0x01, 0x00, 0x0E, 0x01, 0x00, 0xCF, 0x01, 0x80, 0xC7, 0x01, 0xC0, 0xC3, 0x01, 0xE0, 0xC1, 0x01, 0xF0, 0xC0, 0x01, 0xF8, 0xFF, 0x1F, 0xFC, 0xFF, 0x1F, 0xFE, 0xFF, 0x1F, 0xFE, 0xFF, 0x1F, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00), // 4 }</pre> |

Joined: 03 Oct 2012
Posts: 226
Location: chennai

[profile](#) [pm](#)

Análisis de datasheet

- ▶ El LCD esta dividido en paginas, al parecer 8 y según los datos capturados hay 9
- ▶ Se confirma que hay un pin que sirve para diferenciar los datos entre comandos y datos de pixels
- ▶ Existen comandos para configurar el LCD, como ser el contraste
- ▶ Se puede concluir que el datasheet no coincide a la totalidad con el LCD

Datasheet de LCD

EA DOGM128-6 GRAPHIC



TABLE OF PROGRAMMING COMMANDS

| Command | Command Code | | | | | | | | | | | Function |
|---|--------------|-----|-----|------------|----|-----------------------|----|----------------------------------|----------------|----|------|--|
| | A0 | /RD | /WR | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| (1) Display ON/OFF | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | LCD display ON/OFF 0: OFF, 1: ON |
| (2) Display start line set | 0 | 1 | 0 | 0 | 1 | Display start address | | | | | | Sets the display RAM display start line address |
| (3) Page address set | 0 | 1 | 0 | 1 | 0 | 1 | 1 | Page address | | | | Sets the display RAM page address |
| (4) Column address set upper bit Column address set lower bit | 0 | 1 | 0 | 0 | 0 | 0 | 1 | Most significant column address | | | | Sets the most significant 4 bits of the display RAM column address. |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Least significant column address | | | | Sets the least significant 4 bits of the display RAM column address. |
| (5) Status read | 0 | 0 | 1 | Status | | | 0 | 0 | 0 | 0 | 0 | Reads the status data |
| (6) Display data write | 1 | 1 | 0 | Write data | | | | | | | | Writes to the display RAM |
| (7) Display data read | 1 | 0 | 1 | Read data | | | | | | | | Reads from the display RAM |
| (8) ADC select | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | Sets the display RAM address SEQ output correspondence 0: normal, 1: reverse |
| (9) Display normal/reverse | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 1 | 0 | Sets the LCD display normal/reverse 0: normal, 1: reverse |
| (10) Display all points ON/OFF | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | Display all points 0: normal display 1: all points ON |
| (11) LCD bias set | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | Sets the LCD drive voltage bias ratio 0: 1/9 bias, 1: 1/7 bias (ST7565) |
| (12) Read/modify/write | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | Column address increment At write: +1 At read: 0 |
| (13) End | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 0 | Clear read/modify/write |
| (14) Reset | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 0 | Internal reset |
| (15) Common output mode select | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0 | - | - | - | Select COM output scan direction 0: normal direction 1: reverse direction |
| (16) Power control set | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 1 | Operating mode | | | Select internal power supply operating mode |
| (17) Vo voltage regulator internal resistor ratio set | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 | Resistor ratio | | | Select internal resistor ratio(Rb/Ra) mode |
| (18) Electronic volume mode set Electronic volume register set | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | Set the Vo output voltage electronic volume register |
| (19) Static indicator ON/OFF Static indicator register set | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 | 1 | 0 | 0 | 0: OFF, 1: ON |
| | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | Mode | Set the flashing mode |
| (20) Booster ratio set | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | select booster ratio 00: 2x, 4x 01: 5x 11: 6x |
| (21) Power saver | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 1 | 1 | Display OFF and display all points ON compound command |
| (22) NCP | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | Command for non-operation |
| (23) Test | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | - | - | - | Command for IC test. Do not use this command |

0-----Column address-----127

| | |
|----------|--------|
| D0 D7 | Page 0 |
| D0 D7 | Page 1 |
| D0 D7 | Page 2 |
| D0 D7 | Page 3 |
| D0 D7 | Page 4 |
| D0 D7 | Page 5 |
| D0 D7 | Page 6 |
| D0 D7 | Page 7 |

Implementación de una biblioteca para Arduino

- ▶ Ya sabiendo el modelo del LCD, se nota que existe una biblioteca (library) para Arduino
- ▶ Se decide implementar una biblioteca para entender a profundidad el funcionamiento
- ▶ Se implementa un constructor que inicializa el LCD, se utiliza los datos exportados de la captura
- ▶ La placa de prueba tiene soporte de hardware para SPI, se utiliza ese mismo mas los pines extra

Implementación de una biblioteca para Arduino

- ▶ Se implementan `init_lcd()`, `clean_screen()`, `set_pixel()` y `refresh_screen()`
- ▶ Se añade una matriz que representa la pantalla
- ▶ Esta matriz puede ser manipulada indirectamente con `set_pixel(x,y,value)`
- ▶ Con `refresh_screen()` se vuelca la información de la matriz directamente al LCD
- ▶ Dentro de `refresh_screen()` hay una llamada a `SPI.transfer(display[i][j])`

Ya se puede controlar el LCD



Mostrando pixels aleatorios



Destino del LCD

- ▶ Ya sabiendo como funciona y teniendo implementada una biblioteca, posiblemente será destinado a un proyecto que implemente el juego de la vida de Conway

Gracias!