МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ "ЛЬВІВСЬКА ПОЛІТЕХНІКА"

Інститут **ІКНІ**
Кафедра **ПЗ**

ЗВІТ

До лабораторної роботи № 4
На тему: *“Дослідження протоколу SPI на прикладі акселерометра у складі STM32F4DISCOVERY та LCD-дисплею”*
З дисципліни: *“Програмування мікроконтролерів”*

Лектор:
старший викладач
Марусенкова Т.А.

Виконав:
студ. групи ПЗ-31з
Качмар Роман

Прийняв:
асистент
Чопей Р.С.

«___» _____ 2019 р.

∑ = ____ _____

Львів – 2019

Тема роботи: Дослідження протоколу SPI на прикладі акселерометра у складі STM32F4DISCOVERY та LCD-дисплею.

Мета роботи: Засвоїти принципи роботи інтерфейсу SPI та здобути практичні навички організації взаємодії мікроконтролера з периферійними пристроями через SPI за допомогою бібліотек CMSIS, SPL та HAL, акселерометра LIS302DL у складі оціночної плати stm32f4Discovery та дисплею моделі ET-NOKIA LCD 5110.

Теоретичні відомості

1. Яким є інтерфейс SPI: послідовним чи паралельним? *Послідовний.*
2. Яким є інтерфейс SPI: синхронним чи асинхронним? *Синхронний.*
3. Скільки ліній потрібно для забезпечення повнодуплексного режиму роботи SPI?

"Повнодуплексний" вказує не можливість передачі даних одночасно у двох напрямах — від ведучого до веденого і навпаки. Утім, можна налаштувати SPI і на напівдуплексний режим (при цьому можна використовувати на одну лінію менше).

Типово для спілкування ведучого пристрою з веденим застосовуються 4 лінії: SCLK, MOSI, MISO та SS.

***SCLK** – шина тактування, саме по ній передається синхросигнал. Варіант назви: DCLOCK, CLK, SCK.*

***MOSI** (Master Out Slave In) – лінія, по якій ведучий пристрій передає дані веденому. Варіанти назви: DOUT, DO, SDO для ведучого, DIN, DI, SDI – для веденого.*

***MISO** (Master In Slave Out) – лінія, по якій ведучий пристрій приймає дані від веденого. Варіанти назви: DIN, DI, SDI для ведучого, DOUT, DO, SDO – для веденого.*

***SS** (Slave Select) – лінія для вибору конкретного веденого пристрою. Варіант назви: CS (Chip Select). Зазвичай, щоб зробити один ведений пристрій активним, встановлюють CS = 0, для інших підключених ведених пристроїв – CS = 1.*

Формулювання завдання

Завдання:

1. *Написати функцію ініціалізації GPIO та SPI для обміну даними з акселерометром за допомогою **CMSIS SPI Driver.***
2. *Розробити функції для обміну даними (читання та запису) за допомогою CMSIS SPI Driver.*
3. *Зчитати модель акселерометра. За допомогою вікна Watch у середовищі Keil uVision перевірити покази акселерометра. Злегка обертаючи плату, приблизно оцінити правильність показів. Продемонструвати результати роботи.*
4. *Додати у проект функції для ініціалізації дисплею ET-NOKIA LCD 5110. Визначити за наданими кодами, які з виводів GPIO задіяні для обміну даними з дисплеєм і за що відповідає кожен вивід.*
5. *Підключити макетну плату з дисплеєм до визначених виводів (усі виводи дисплею промарковані).*
6. *Написати додаткові функції для роботи з дисплеєм у відповідності до індивідуального завдання.*

Хід виконання завдання

Завдання 1: *Написати функцію для очищення дисплею.*

Код програми (**main.c**):

```
#include <stm32f4xx.h>
#include <stm32f4xx_hal_rcc.h>
#include <stm32f4xx_hal_rcc_ex.h>
#include <stm32f4xx_hal_gpio.h>
#include "main.h"
#include "Driver_SPI.h"

GPIO_InitTypeDef GPIO_Init_LED;
SPI_InitTypeDef spi;

SPI_Master_init();

int main(void)
{

  SPI_Master_init();
  while(1) {
    SPI_I2S_SendData(SPI1, 0xFFFF);
    SPI1->DR = 0x11;
    SPI_I2S_SendData(SPI2,0xFFFF);
    SPI2->DR = 0x11;
  }

}

void SPI_Master_init(void) {

GPIO_Init_LED.GPIO_Pin = GPIO_Pin_10|GPIO_Pin_11|GPIO_Pin_12;
GPIO_Init_LED.GPIO_Mode = GPIO_Mode_AF;
GPIO_Init_LED.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_Init_LED.GPIO_OType = GPIO_OType_PP;
GPIO_Init_LED.GPIO_PuPd = GPIO_PuPd_NOPULL;
GPIO_Init(GPIOC, &amp;amp;GPIO_Init_LED);
GPIO_PinAFConfig(GPIOC, GPIO_PinSource10, GPIO_AF_SPI3);
GPIO_PinAFConfig(GPIOC, GPIO_PinSource11, GPIO_AF_SPI3);
GPIO_PinAFConfig(GPIOC, GPIO_PinSource12, GPIO_AF_SPI3);
RCC_AHB1PeriphClockCmd(RCC_AHB1Periph_GPIOE, ENABLE);
GPIO_Init_LED.GPIO_Pin = GPIO_Pin_3;
GPIO_Init_LED.GPIO_Mode = GPIO_Mode_OUT;
GPIO_Init_LED.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_Init_LED.GPIO_OType = GPIO_OType_PP;
GPIO_Init_LED.GPIO_PuPd = GPIO_PuPd_NOPULL;
GPIO_Init(GPIOD, &amp;amp;GPIO_Init_LED);


  //ENABLE THE CLOCK TO SPI1 PERIPHERAL
RCC_APB2PeriphClockCmd(RCC_APB2Periph_SPI1, ENABLE);

//SET SPI SPEED TO LOW SPEED MODE (~282KHZ)
spi.SPI_BaudRatePrescaler = SPI_BaudRatePrescaler_256;
spi.SPI_CPHA = SPI_CPHA_1Edge;
spi.SPI_CPOL = SPI_CPOL_Low;
spi.SPI_DataSize = SPI_DataSize_8b;
spi.SPI_Direction = SPI_Direction_2Lines_FullDuplex;
spi.SPI_FirstBit = SPI_FirstBit_MSB;
spi.SPI_Mode = SPI_Mode_Master;
spi.SPI_NSS = SPI_NSS_Soft;

SPI_Init(SPI1, &spi);
```

```
SPI_Cmd(SPI1, ENABLE);
}
```

Висновок

Виконуючи дану лабораторну роботу, я засвоїв принципи роботи інтерфейсу SPI та здобув практичні навички організації взаємодії мікроконтролера з периферійними пристроями з використанням бібліотеки SPL.