Міністерство освіти і науки України
Національний університет «Львівська політехніка»
Кафедра «Телекомунікації»

Методичні вказівки
до виконання лабораторної роботи №3
з дисципліни «Вбудовані системи»
на тему «Використання SPI інтерфейсу»

Львів 2021

Мета роботи: Навчитися використовувати SPI інтерфейс для обміну даними між пристроями.

Теоретичні відомості

Послідовний периферійний інтерфейс (Serial Peripheral Interface – **SPI**) є синхронним несиметричним послідовним інтерфейсом, призначеним для передачі даних на невеликі відстані між інтегральними мікросхемами. Він був розроблений фірмою ***Motorola***. Шина SPI організована за принципом “ведучий-ведений”. В якості ведучого шини зазвичай виступає мікроконтролер, але їм також може бути програмована логіка, DSP-контролер або спеціалізована IC (інтегральна схема). В їх ролі виступають різного роду мікросхеми, в т. ч. регістри зсуву, запам'ятовуючі пристрої (EEPROM, Flash-пам'ять, SRAM), годинник реального часу (RTC), АЦП/ЦАП, цифрові потенціометри, спеціалізовані контролери та ін.

На Рис. 1 представлена схема під’єднання МК і периферійних пристроїв з використанням інтерфейсу SPI. У даному прикладі МК є ведучим пристроєм, він може ініціювати передачу інформації між МК і одним з периферійних пристроїв.

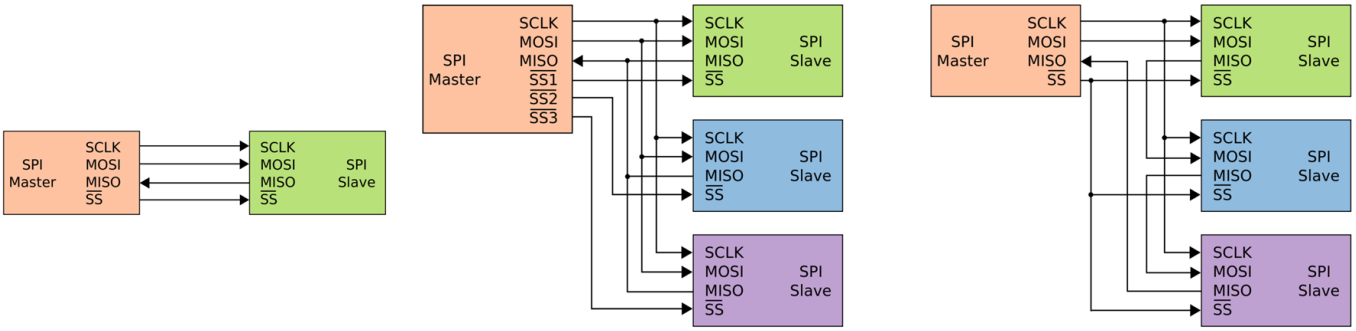


Рис 1. Способи під’єднання пристроїв до шини SPI

Шина SPI використовує чотири лінії зв'язку:
1) MOSI – лінія передачі даних від ведучого приладу до веденого (*Master Output Slave Input*).
2) MISO – лінія передачі даних від веденого до ведучого (*Master Input Slave Output*).
3) SCK – лінія сигналу синхронізації даних.
4) SS (CS) - лінія вибору веденого пристрою.

Утворена на основі інтерфейсу SPI міні-мережа відноситься до класу магістрально-радіальних. Лінії передачі даних і лінія синхронізації є прикладом шинної організації, а лінії вибору веденого пристрою – елементом системи радіального типу.

Передача даних по **SPI** здійснюється наступним чином: Перед початком обміну ведучий пристрій вибирає один ведений пристрій, з яким проводитиметься обмін. Для цього на лінії вибору пристрою ***SS*** (або ***CS***) встановлюється рівень логічного «**0**». Після запису в регістр даних **SPI** (*SPDR для AVR*) ведучого мікроконтролера запускається генератор тактового сигналу модуля, і дані починають побітно видаватися в лінію ***MOSI*** і відповідно поступають на лінію ***MOSI*** веденого пристрою. Після видачі останнього біта поточного байта генератор тактового сигналу зупиняється з одночасним встановленням в «**1**» прапора «*Кінець передачі*». Якщо підтримуються і дозволені переривання від модуля **SPI**, то генерується запит на переривання. Після цього ведучий мікроконтролер може почати передачу наступного байта або, подавши на вхід ***SS*** веденого пристрою рівень логічної «**1**», перевести його в неактивний стан. Одночасно з передачею даних

від ведучого до веденого відбувається передача і в зворотному напрямку, за умови, що на вході ***SS*** веденого присутня напруга низького рівня. Таким чином, в кожному циклі зсуву відбувається обмін даними між пристроями. В кінці кожного циклу прапор переривання встановлюється в «1» як у ведучого, так і у веденого. Прийняті байти зберігаються в буфер прийому для подальшого використання. При прийомі даних прийнятий байт повинен бути прочитаний з регістру даних **SPI** до того, як в регістр зсуву надійде останній біт наступного байта. В іншому випадку перший байт буде втрачено. Вивід ***SS*** призначений для вибору активного веденого пристрою і в режимі веденого завжди є входом. Кожен раз, коли на вивід ***SS*** подається рівень логічної «1», відбувається скидання (*очищення буфера*) модуля **SPI**. Якщо зміна стану цього виводу відбудеться під час передачі даних, і прийом, і передача негайно припиняться, а байт даних який приймався буде втрачено. Приклад обміну інформацією між двома пристроям приведений на рис. 2.

На час відсутності зв'язку (**SS** неактивний) виводи модуля **SPI** переводяться в стан високого імпедансу (налаштовуються на введення). Це дозволяє уникнути конфліктів на шині **SPI**. Інакше декілька виводів **MISO** ведених пристроїв одночасно були б активними, що не дозволило б ведучому пристрою провести прийом достовірної інформації. Тільки один з пристроїв на шині **SPI** в кожен момент часу може працювати у ведучому режимі, інші – тільки у веденому. Ведучий пристрій формує сигнали на виводах **MOSI** і **SCK**, які поступають на однойменні виводи ведених пристроїв. Один з вибраних ведених пристроїв передає дані через вивід **MISO** на вивід **MISO** ведучого пристрою. Вивід **SS** вбудованого модуля **SPI** використовується залежно від того, в якому режимі працює цей пристрій. При роботі у веденому режимі при подачі високого рівня сигналу на вхід **SS** пристрій ігнорує сигнали **SCK** і утримує вивід **MISO** в стані високого імпедансу. Якщо у веденому режимі роботи на вході **SS** встановлений низький логічний рівень, то лінії **MOSI** і **SCK** налаштовуються на введення, лінія **MISO** – на вивід. При роботі у ведучому режимі лінія **SS** може бути використана як звичайна лінія вводу/виводу. Як правило, лінія **SS** мікроконтролера який працює в режимі ведучого, використовується для вибору веденого пристрою.

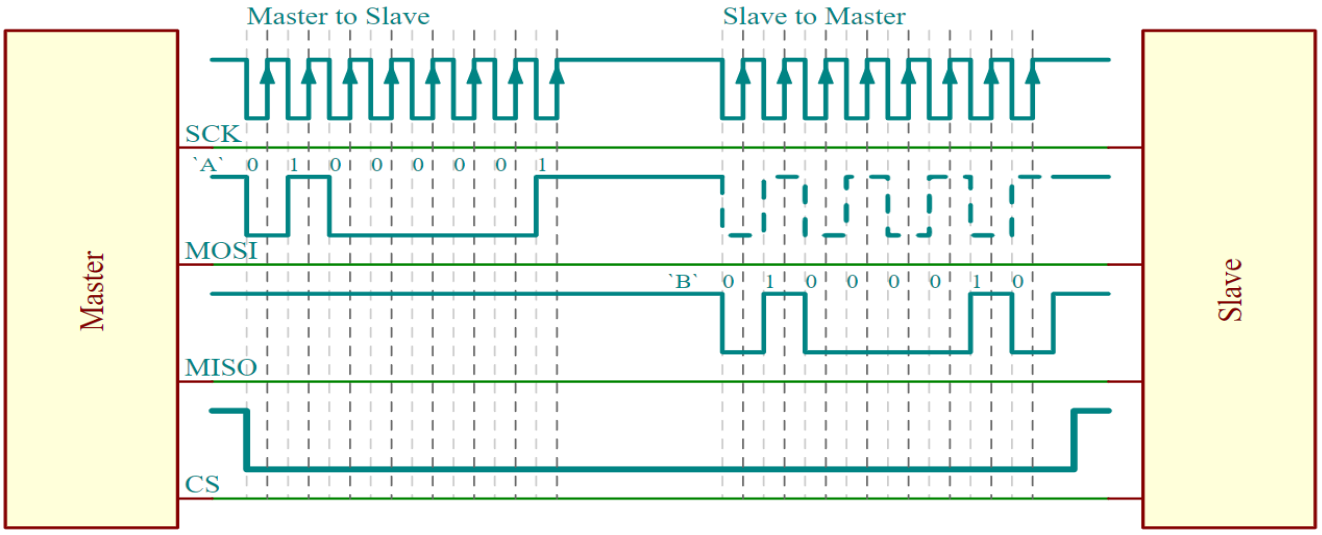
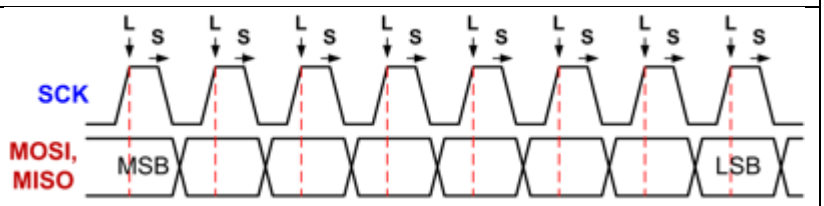
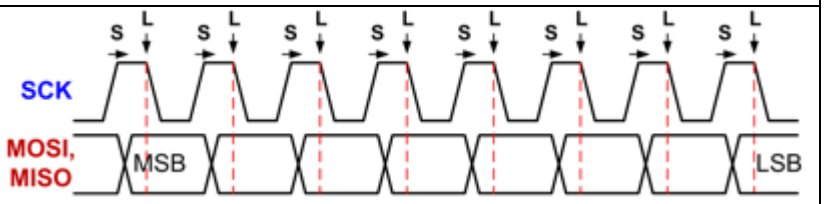
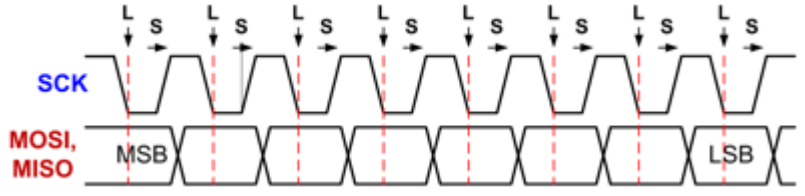
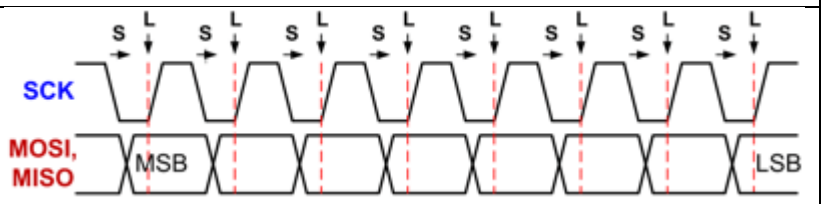


Рис 2. Приклад обміну даними між двома пристроями по SPI.

Схема управління модуля SPI - інтерфейсу дозволяє вибрати один з чотирьох режимів роботи (відповідно до комбінації бітів CPHA і CPOL *(регістра **SPCR** мікроконтролера AVR)*). При роботі у ведучому режимі можна також програмно вибрати швидкість передачі даних.

Режим SPI	**Часова діаграма**
Режим 0 (CPHA=0, CPOL = 0), Активний рівень імпульсу синхронізації - високий, спочатку зчитування, потім зсув	
Режим 1 (CPHA=0, CPOL = 1), Активний рівень імпульсу синхронізації - високий, спочатку зсув, потім зчитування	
Режим 2 (CPHA=1, CPOL = 0), Активний рівень імпульсу синхронізації - низький, спочатку зчитування, потім зсув	
Режим 3 (CPHA=1, CPOL = 1) , Активний рівень імпульсу синхронізації - низький, спочатку зсув, потім зчитування	

*(MSB - старший біт, LSB - молодший біт)

Завдання до роботи:

1. Написати програму на мові Сі згідно з варіантом завдання.
2. Створити схему (згідно з варіантом завдання) в програмі для моделювання (SimulIDE або Proteus 8). Провести моделювання написаної програми.
3. При наявності деталей зібрати схему на макетній платі та запрограмувати мікроконтролер.

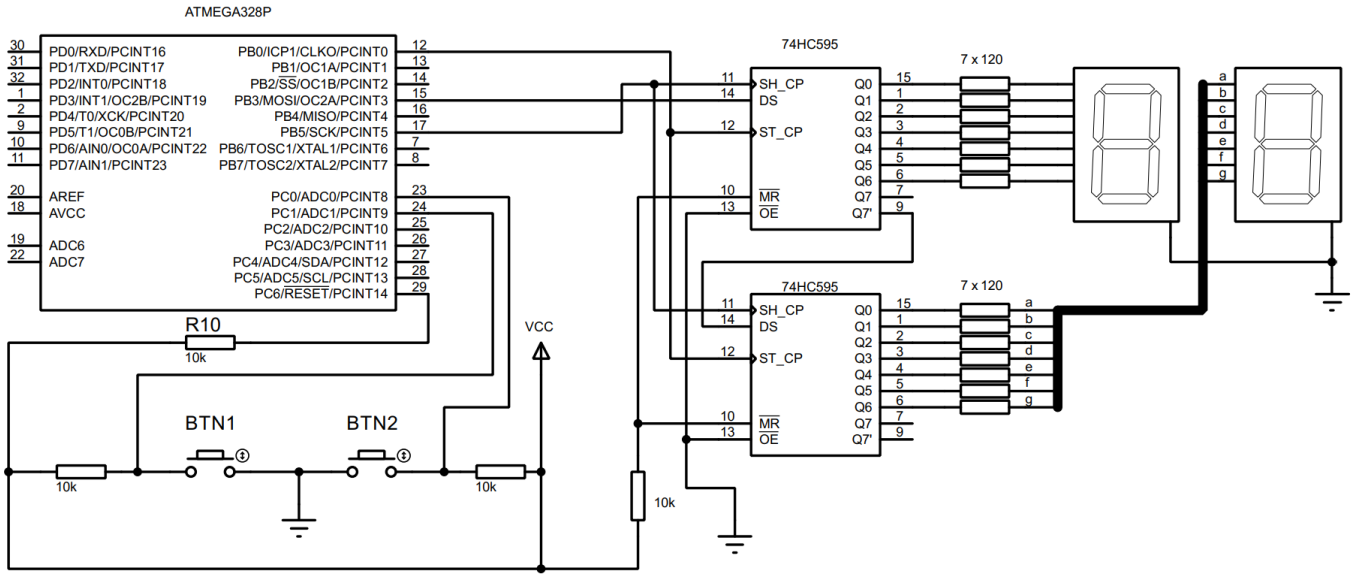


Рис 3. Схема підключення 7-сегментних індикаторів до мікроконтролера, через регістри зсуву.

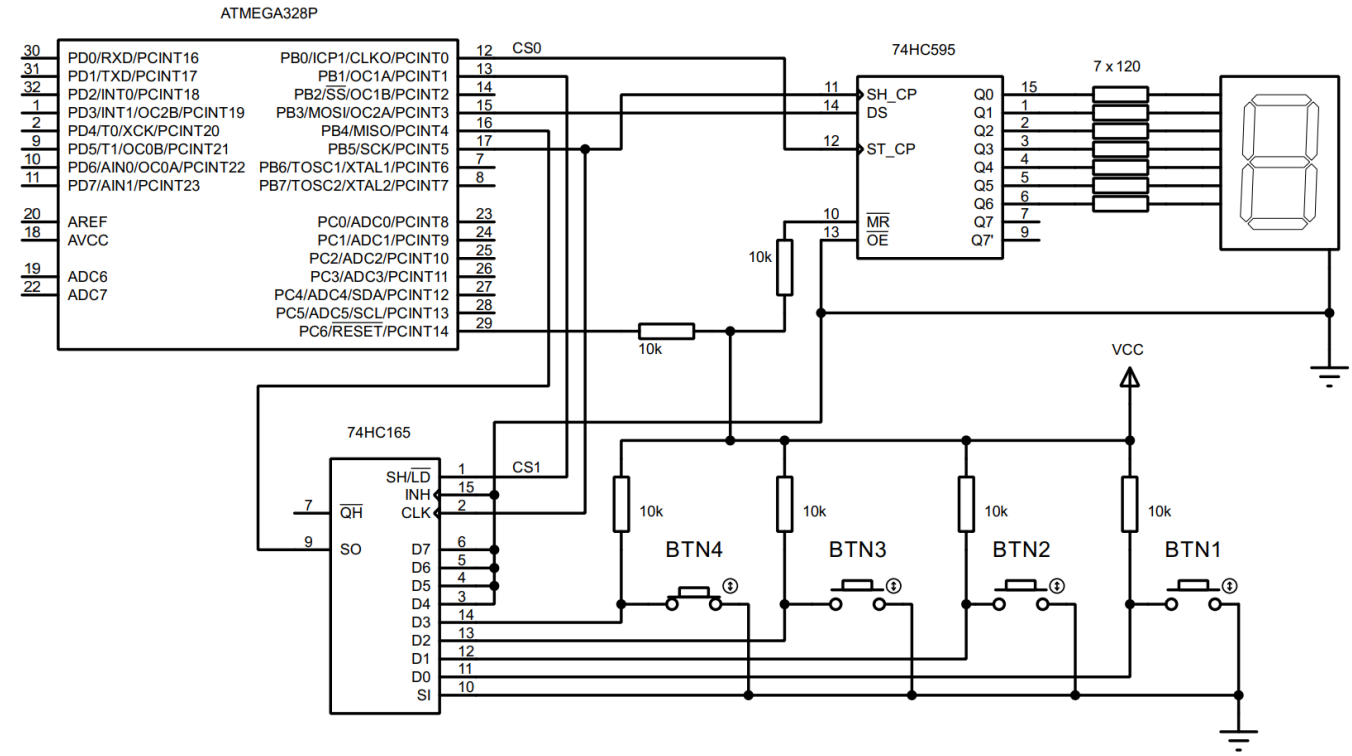


Рис 4. Схема підключення вхідного, та вихідного регістрів зсуву до мікроконтролера по SPI.

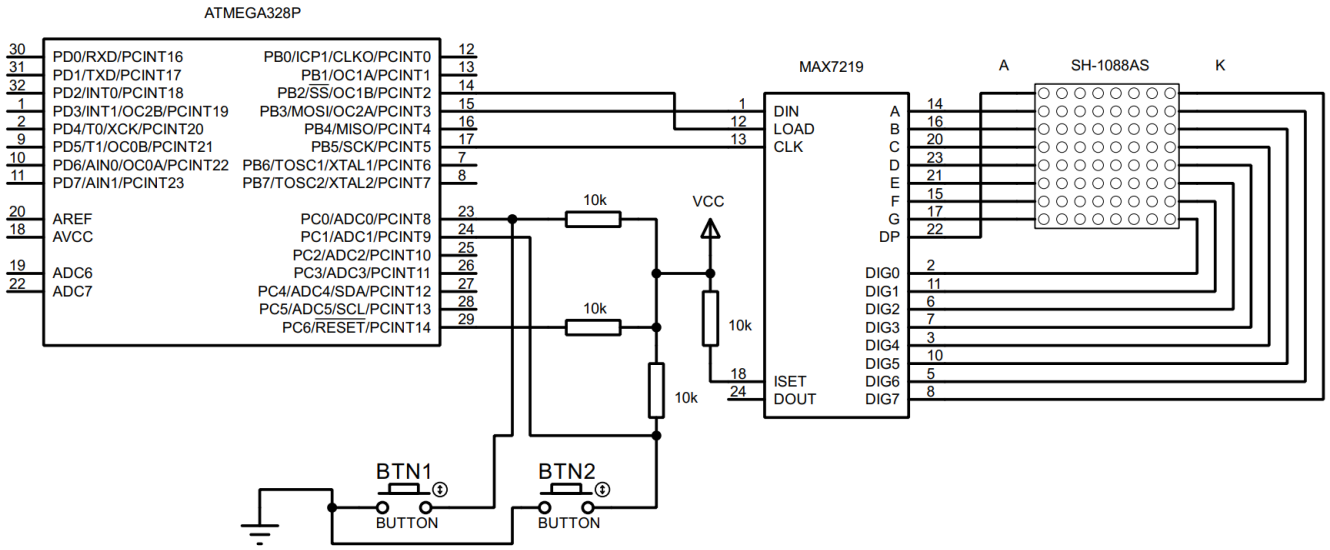


Рис 5. Схема підключення матричного світлодіодного індикатора до мікроконтролера, з використанням мікросхеми MAX7219.

Демонстраційна програма :

```
#include <avr/io.h>
#include <util/delay.h>

#define OUT_REG_CS PB0

void SPI_MasterInit (void)
{
  // Налаштувати піни MOSI (PB3) і SCK (PB5) як виходи
  DDRB |= (1<<PB3)|(1<<PB5);
  // Включити SPI, задати режим роботи 0 Master,
  // встановити швидкість fck/16
  SPCR = (1<<SPE)|(1<<MSTR)|(1<<SPR0);
}

void SPI_MasterTransmit(uint8_t cData)
{
  // Початок передачі
  SPDR = cData;
  // Чекати завершення передачі
  while (!(SPSR & (1<<SPIF)));
}

//-----------------------------------------------------------
int main(void)
{
 uint8_t d = 1;

  SPI_MasterInit();
  DDRB |= 1 << OUT_REG_CS; // setup CS
```

```
  for(;;)
  {

    PORTB |= 1 << OUT_REG_CS;
    SPI_MasterTransmit(d);
    PORTB &= ~(1 << OUT_REG_CS);
    d <<= 1;
    if (d > 32) d = 1;
    _delay_ms(200);
  }
 return 0;
}
```

Дана програма на схемі Рис 4. «по кругу» засвічує сегменти на індикаторі підключеному до регістра зсуву.

Варіанти завдань

1. Використовуючи схему Рис 3. реалізувати секундомір з виводом результату на 7-сегментні індикатори. Кнопка BTN1 - старт/стоп, BTN2 - занулює результат. Для реалізації зв'язку мікроконтролера використовувати апаратну реалізацію SPI.
2. Реалізувати завдання 1. Використовуючи програмну реалізацію SPI.
3. Реалізувати на схемі Рис 3. лічильник натискань кнопок BTN1 і BTN2 в BCD форматі. Кнопка BTN1 – збільшує значення лічильника на 1, BTN2 – зменшує значення на 1.
4. Написати програму для схеми Рис 4. яка реалізує лічильник в 16-ковому коді (0…F) з виводом на 7-сегментний індикатор. Кнопка BTN1 – збільшує значення на 1, BTN2 – зменшує значення на 1, BTN3 – занулює результат. *Для отримання даних з 74HC165 фіксуємо стан входів низьким лог. рівнем на виводі CS1 (1 -> 0 -> 1) а потім зчитуємо байт даних.*
5. Розробити програму яка виводить на індикатор номер натиснутої кнопки. Використовувати схему на Рис 4. та апаратну реалізацію SPI. *Для отримання даних з 74HC165 фіксуємо стан входів низьким лог. рівнем на виводі CS1 (1 -> 0 -> 1) а потім зчитуємо байт даних.*
6. (*****) Написати програму яка виводить біжучу стрічку (з зсувам на один піксель) на матричному індикаторі (схема Рис 5).
7. (*****) Використовуючи схему Рис 5. написати просту ігру (pong, google Dino, brakeout, …).