НАВЧАЛЬНО-МЕТОДИЧНА ЛІТЕРАТУРА

Тернопільський національний технічний університет імені Івана Пулюя

Кафедра автоматизації технологічних процесів і виробництв

# Лабораторна робота № R02

з курсу

”Мікропроцесорні та програмні засоби автоматизації”

# Робота з портами GPIO за допомогою бібліотеки WiringPi

Тернопіль 2021

Методичні вказівки до лабораторної роботи № R02 “Робота з портами GPIO за допомогою бібліотеки WiringPi” з курсу "Мікропроцесорні та програмні засоби автоматизації". Марущак П.О., Медвідь В.Р., Пісьціо В.П., Тернопіль: ТНТУ, 2021 - 12 с.

Для студентів напряму: 151 "Автоматизація та комп'ютерні технології"

Автори: Марущак П.О., Медвідь В.Р., Пісьціо В.П..

**Тема роботи**

Робота з портами GPIO за допомогою бібліотеки WiringPi

**Мета роботи**

Ознайомитись з функціональними можливостями та роботою із GPIO Raspberry Pi за допомогою бібліотеки WiringPi.

**Теоретичні відомості**

**Робота з портами**

Для простого вводу-виводу даних у Raspberry Pi можна використовувати інтерфейс GPIO. GPIO (General Purpose Input Output) - це низькорівневий інтерфейс вводу-виводу прямого управління, для цього на платі Raspberry Pi 3 присутній 40 контактний роз'єм GPIO. Через котрий Raspberry може приймати та отримувати сигнали стану логічного 0 та 1. Так як інтерфейс низькорівневий, обмінюватися сигналами Raspberry може з будь-якими іншими пристроями - від світлодіоду до складних цифрових приладів та датчиків. Схема розташування та призначення виводів приведені на рис. 2.1.

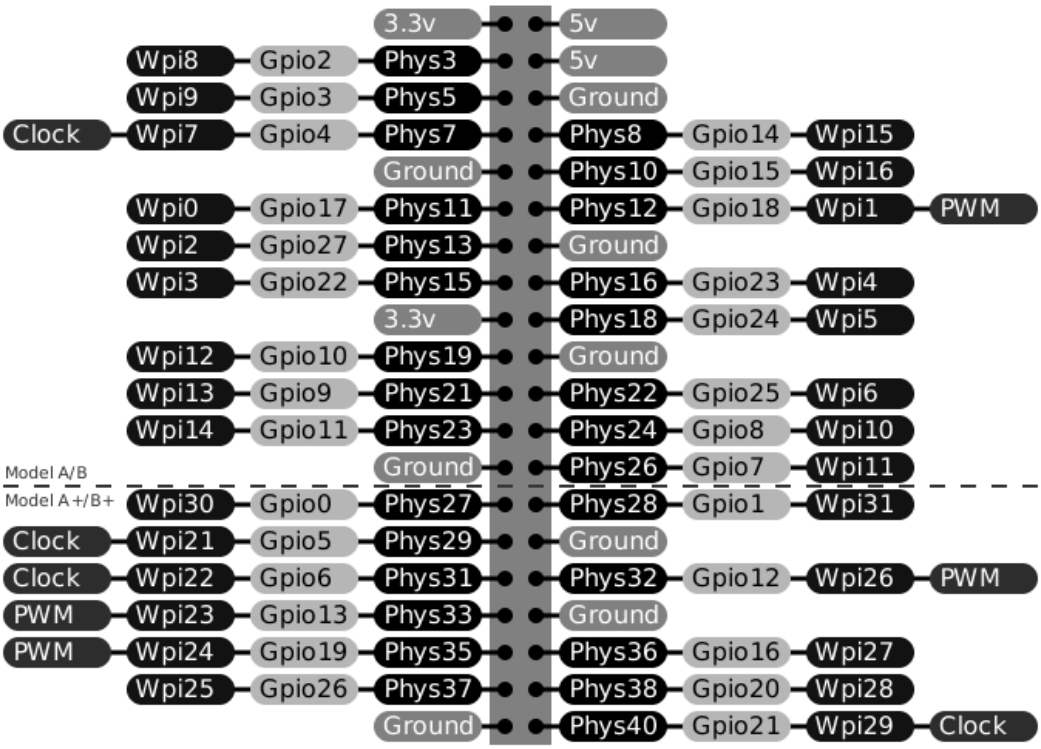



**Рис. 1. Контакти GPIO**

Як видно із рисунку, деякі сигнали крім стандартного простого вводу-виводу можуть виконувати свою роль, котру назвають альтернативною функцією контакта. Крім того як видно, на контакти роз'єма виведені також сигнали деяких поширених цифрових інтерфейсів, а також напруги живлення та заземлення. Всі контакти GPIO підведені безпосередньо у схему, а сигнали GPIO заведені напряму у процесор, тому слід бути уважним і не допускати перевантаження сигналів чи подачи на лінії сторонніх напруг більше 3.3 В чи менше 0 В.

Сигнали GPIO підтримують режими: низькорівневе вводу; низькорівневого виводу та 6 спеціальних режимів роботи. Також деякі сигнали GPIO можна використовувати для відправки переривань на процесор.

**Бібліотека інтерфейсів GPIO для Raspberry Pi WiringPi**

WiringPi - це бібліотека доступу до контактів вводу-виводу GPIO на основі, написана на C для пристроїв SoC BCM2835, BCM2836 та BCM2837, використовуваних у всіх версій Raspberry Pi. Вона випущена за ліцензією GNU LGPLv3 і може використовуватися з C, C ++ та RTB (BASIC), а також багатьма іншими мовами. І призначена для ознайомлення людьми, які використовували систему Wiring Arduino та призначена для використання досвідченими програмістами C / C ++.

WiringPi можна завантажити тут

**https://github.com/WiringPi/WiringPi**

WiringPi розроблена безпосередньо на Raspberry Pi з 32-розрядною програмою Raspbian. Вона не підтримує будь-яку іншу платформу, крос-компіляцію або операційні системи. Бібліотека була перенесений на інші платформи, інші операційні системи, а деякі - перехресні компіляції, однак автор не підтримує ці системи. Якщо ви намагаєтеся використовувати wiringPi на інших платформі, ніж Raspberry Pi з Raspbian, тоді ви повинні зв'язатися з людиною, яка зробила порт.

Оригінальна версія Raspberry Pi Модель A та B була версією B1 з 26-контактним роз'ємом вводу/виводу загального призначення (GPIO), який містить набір сигналів та шини. Вона має 8 цифрових ліній вводу / виводу загального призначення - що можуть бути запрограмовані як цифрові виходи, так і входи. Дві з цих ліній на 40-контактних Pi, ( і лише одна на 26-контактних Pi) можуть бути призначені і для апаратного виходу ШІМ. Додатково є 2-провідний інтерфейс I2C та 4-провідний інтерфейс SPI (з другою лінією вибору, що робить його 5 контактним) а також послідовний порт UART з ще двома контактами.

Протягом останніх років були випущені:

- Оригінальна модель B з 26-контактним роз'ємом GPIO.
- Модель B, версія 1.1 Raspberry Pi що має додаткові 4 лінії GPIO на окремому роз'ємі, який потрібно припаяти до плати.
- Модель A, яка по суті така ж, як і модель B v1.1, але без концентратора USB та роз'єму Ethernet.
- Модель A + та B + Raspberry Pi має єдиний 40-контактний роз'єм GPIO з 28 лініями GPIO та 4 розетками USB. (Немає USB або Ethernet на A +)
- Модель B v2 оснащена чотирьохядерним процесором Arm A7 з 1 Гб оперативної пам'яті. Кількість ліній GPIO не змінилась.
- Модель Zero - це скороченна версія Pi A + з 40-контактним роз'ємом GPIO.
- Модель B v3 оснащена чотирьохядерним процесором Arm A8 (64 біт) з тими ж обсягом оперативної пам'яті та GPIO, що і модель 2, однак має також вбудований Wi-Fi та Bluetooth.
- Модель Zero-W додає вбудовану Wi-Fi, Bluetooth та роз'єм камери Pi до існуючої моделі Zero.

Інтерфейси I2C, SPI та UART також можуть використовуватися як контакти вводу/виводу загального призначення, коли вони не використовуються в режимах відповідних шини, даючи додатково 8 + 2 + 5 + 2 = 17 контактів вводу / виводу на роз'ємі P1 (плюс ще 4 на роз'ємі P5 на версії 2 Pi) та 28 контактів вводу-виводу на платах B + та версіях 2 та 3 (хоча 2 лінії зарезервовані для інтерфейсу I2C, але вони можуть використовуватися як звичайні GPIO, якщо він не використовується.)

WiringPi включає у себе утиліту командного рядка GPIO яка може бути використана для програмування і установки ліній GPIO. Можна використовувати її для читання та запису ліній і навіть використовувати її для керування лініями зі скриптів оболонки.

WiringPi розширюється, за допомогою модулів для використання аналогових інтерфейсних пристроїв на Gertboard, а також для використання популярних мікросхем розширення MCP23x17 / MCP23x08 GPIO та інших. Також можна використати модуль що розширює кількість ліній вводу-виводу на додаткові 32 лінії за допомогою 4 зсувних регістрів 74x595. Існує модуль розширення, що дозволяє використовувати ATmega (наприклад, Arduino або Gertboard) як додаткове розширення GPIO.

Крім того, можна легко написати власні модулі розширення, щоб інтегрувати свої власні периферійні пристрої з wiringPi за потребою. WiringPi підтримує аналогове читання та запис, і хоча за замовчуванням немає вбудованого аналогового обладнання на Pi, передбачені модулі для підтримки аналогових мікросхем Gertboards та інших ЦАП та АЦП.

WiringPi devLib являє собою набір бібліотечних підпрограм , реалізованих з використанням wiringPi, щоб дати вам легкий доступ до деякої популярної периферії. Підтримувані пристрої включають символьні РК-дисплеї (на основі чіпів Hitachi HD44780U) та графічні - наприклад, звичайні дисплеї 128 × 64 пікселів із загальним чіп-драйвером 12864H.

Тактовий чіп DS1302 RTC, датчики на основі мікросхем Maxdetect (наприклад, RHT003), інтерфейсні плати Gertboard та PiFace тощо.

PiFace та Gertboard. WiringPi також повністю підтримує плату PiFace і Gertboard. Завдяки Gadgetoid зараз є обгортки для Ruby, Python та Perl, завдяки Jeroen Kransen є обгортки для Java, які можна, завдяки Дейву Бултону за створення TCL обгортки, існує також обгортка Pi4J - ще один проект Java, який використовує WiringPi.

**Ініціалізація WiringPi.**

Перед використанням функцій бібліотеки її слід ініціалізувати. Існує чотири способи ініціалізації WiringPi.

```
int wiringPiSetup (void);
int wiringPiSetupGpio (void);
int wiringPiSetupPhys (void);
int wiringPiSetupSys (void);
```

WiringPi версії 1 повертає код помилки, якщо ці функції не вдалися з будь-якої причини. Версія 2 завжди повертає нуль. Після обговорення та перевірки багатьох програм, написаних користувачами wiringPi, і помітивши, що багато людей не турбуються перевіряти код повернення, я висловив позицію, що якщо одна з функцій налаштування wiringPi не вдасться, тоді це вважатиметься фатальною помилкою програми та виконання програми буде припинено в цей момент повідомленням про помилку, надрукованим на терміналі.

Якщо є потреба відновити поведінку з початкової версії, тоді потрібно встановити змінну середовища: WIRINGPI_CODES (значення не важливе, сама змінна лише має існувати).

Відмінності між функціями налаштування полягають у наступному:

- wiringPiSetup (void); ініціалізує wiringPi і передбачає, що програма, що викликає функції бібіліотеки, буде використовувати схему нумерації контактів wiringPi . Це спрощена схема нумерації, яка забезпечує відображення віртуальних контактів від 0 до 16 до реальних базових номерів контактів Broadcom GPIO.
- wiringPiSetupGpio (void); ідентично вищеописаній, проте дозволяє програмам, що викликають, безпосередньо використовувати номери контактів Broadcom GPIO без повторного відображення. Як зазначено вище, цю функцію потрібно викликати з правами root, і зауважте, що деякі контакти відрізняються від версії 1 до версії 2 плати.
- wiringPiSetupPhys (void); Ідентично вищеописаній, проте дозволяє програмам, що викликають, використовувати лише фізичні номери контактів на роз'ємі P1.
- wiringPiSetupSys (void); ініціалізує wiringPi, але використовує інтерфейс /sys/class/gpio, а не прямий доступ до обладнання. Це може дозволити роботу програми із без прав доступу root за умови попереднього експорту контактів GPIO за допомогою програми gpio. Нумерація контактів у цьому режимі - це рідні GPIO-номери Broadcom - такі самі, як wiringPiSetupGpio(), наведені вище, тому слід пам’ятати про відмінності між платами Rev 1 та Rev 2. У цьому режимі ви можете використовувати лише контакти, які були експортовані через інтерфейс /sys/class/gpio перед запуском програми. Ви можете зробити це в окремому скрипті оболонки або за допомогою функції system () зсередини програми для виклику програми gpio. Також зауважте, що деякі функції не мають ніякого ефекту при використанні цього режиму, оскільки в даний час їх неможливо діяти, якщо не викликається правами root. (хоча ви можете використовувати system () для виклику gpio для встановлення / зміни режимів, якщо потрібно)

**Основні функції**

Ці функції працюють безпосередньо на Raspberry Pi, а також із зовнішніми модулями GPIO, такими як розширювачі GPIO тощо, хоча не всі модулі підтримують усі функції - наприклад, PiFace попередньо налаштований для фіксованих входів і виходів, а Raspberry Pi не має бортове аналогове обладнання.

- void pinMode (int pin, mode int) встановлює режим лінії або INPUT, OUTPUT , PWM_OUTPUT абоGPIO_CLOCK . Зверніть увагу , що тільки wiringPi контакт 1

(BCM_GPIO 18) вихід підтримує ШІМ і тільки wiringPi контакт 7 (BCM_GPIO 4) підтримує режими виведення такотового сигналу.

- void pullUpDnControl (int pin, int pud) встановлює режим контакта, що слід встановити як вхід. На відміну від Arduino, BCM2835 має як підтягуючі внутрішні резистори і до високого і до низького рівня. Параметр pud може бути; PUD_OFF, (без підтяжки),PUD_DOWN (підтяжка на землю) або PUD_UP (підтяжка до 3,3 В) Внутрішні підтягуючі резистори мають опір у Raspberry Pi приблизно 50 кОм. Ця функція не впливає на GPIO-лінії Raspberry Pi в режимі Sys. Якщо вам потрібно активувати підтяжку, ви можете зробити це за допомогою програми gpio в сценарії перед запуском програми.
- void digitalWrite (int pin, int value ); видає значення HIGH та LOW (1 або 0) на дану лінію, якщо лінія працює як вихід. У WiringPi будь-ненульове значення параметра value вважається високим, а нульове значення параметра value видає на вивід 0 (LOW).
- int digitalRead (int pin); функція повертає значення, прочитане з даного контакту. Повернуте значення буде 1 (HIGH) або 0 (LOW) залежно від логічного рівня на контакті.
- analogRead (int pin); повертає значення, прочитане на вхідному аналоговому вводі. Програма взаємодіє з додатковими аналоговими модулями, щоб увімкнути цю функцію для таких пристроїв, як Gertboard, аналогова плата quick2Wire тощо.
- analogWrite (int pin, int значення); записує задане значення на заданий аналоговий контакт. Вам потрібно буде зареєструвати додаткові аналогові модулі, щоб увімкнути цю функцію для таких пристроїв, як Gertboard.

**Спеціальні функції контактів**

WiringPi може керувати всіма контактами GPIO, але деякі з них можуть потенційно спричинити проблеми з іншими частинами системи Raspberry Pi Linux.

Контакти від 0 до 6 безпечні у використанні в будь-який час і можуть бути встановлені для вводу або виводу за допомогою або без включених внутрішніх підтягувальних або знижувальних резисторів.

Контакт 7 (BCM_GPIO 4) можно нормально використовувати, однак він використовується драйвером ядра 1-Wire і він, за бажанням, може бути підключений до джерела GPIO тактового сигналу. ШІМ. Ви можете змінити функцію контакта на вихід ШІМ, однак якщо ви в даний час відтворюєте музику або використовуєте аудіосистему через роз'єм 3,5 мм, тоді ви знайдете один канал аудіо ШІМ, що проходить через контакт. Якщо ви взагалі не використовуєте аудіо (або аудіо йде через кабель HDMI), цей контакт можна легко використовувати в режимі ШІМ.

Контакти 8 і 9 : - контакти шини I2C. їх можна використовувати для цифрового вводу-виводу, якщо не використовується жодних драйверів I2C, які використовують ці контакти, однак у них є внутрішні резистори 1,8 кОм, що подають сигнали на 3v3-джерело. Ця функція робить їх зручними для входів комутатора, коли комутатор просто закорочує контакт на землю, не вмикаючи внутрішні підтягуючі резистори

Контакти 10, 11, 12, 13 і 14 використовуються для інтерфейсу SPI. Як і інтерфейс I2C, якщо ви його не використовуєте, то ви можете вільно використовувати їх у своїх цілях. На відміну від I2C, ці контакти не мають зовнішніх резисторів, що підтягуються.

Контакти 15 і 16 використовуються UART відповідно для Tx і Rx. Якщо їх використовувати як контакти загального призначення, то слід переконатися, що системна послідовна консоль відключена.

**Використання плати Raspberry Pi 3 із платами розширення**

При роботі із платами розширення ARPI600 та Multifunction Shield деякі контакти GPIO стають під'єднаними до периферійних пристроїв плати Multifunction Shield. Фрагмент схеми відповідного включення показаний на наступному рисунку (Рис. 2).

Із схеми видно кнопки підключені до контактів GPIO 21-23, світлодіоди - до контактів GPIO 14-12,10, а динамік до контакту 1.

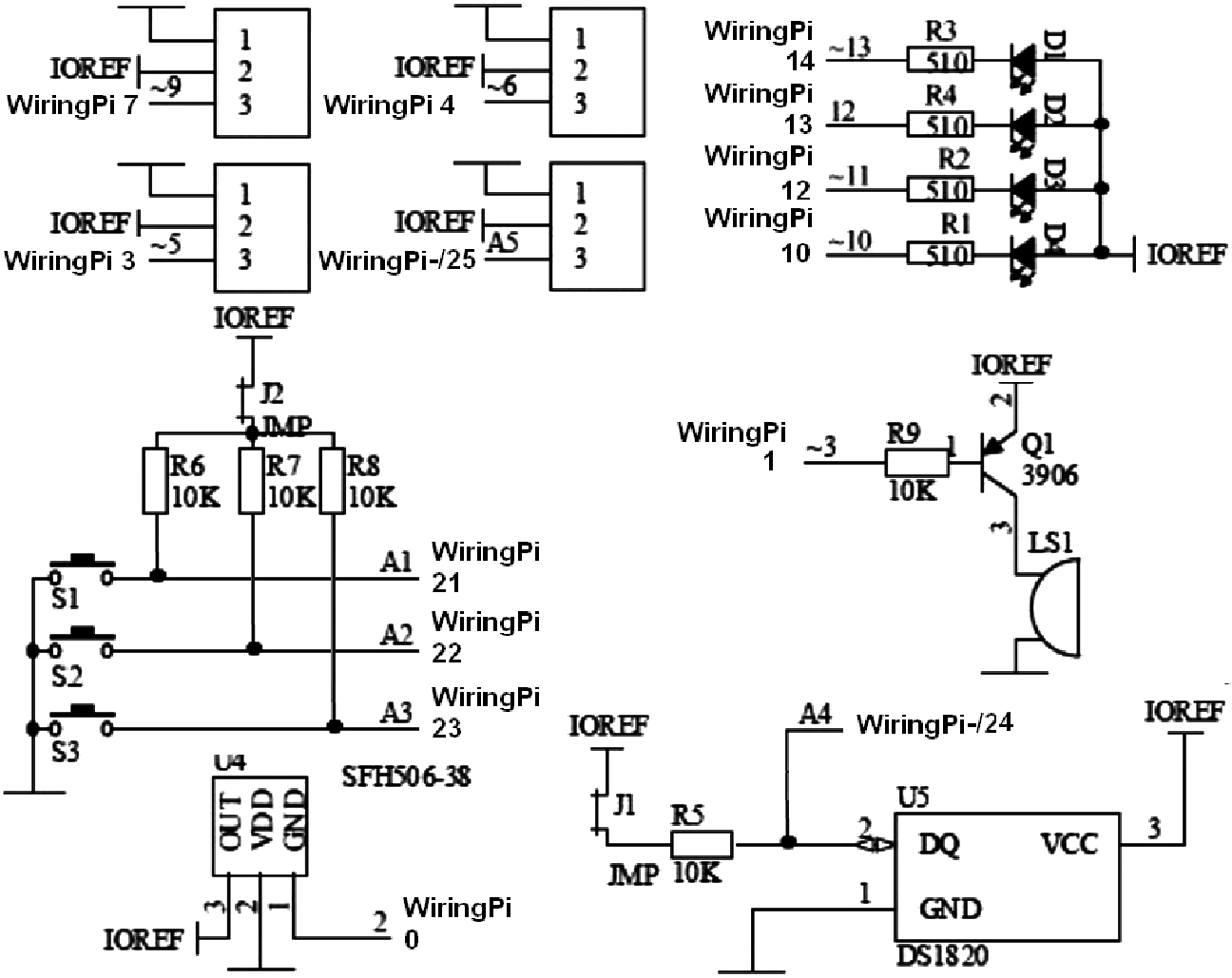



**Рис. 2. Фрагмент схеми елементів простого вводу-виводу на платі Multifunction Shield**

**Приклад простої програми**

Програма мигає світлодіодом D1 на платі.

```
/* * blink.c: *     Standard "blink" program in wiringPi. Blinks
an LED connected to the D1 *
*********************************/
#include <stdio.h>
#include <wiringPi.h>
// LED Pin - wiringPi pin 0 is BCM_GPIO 17.
#define LED 14
int main (void)
{ printf ("Raspberry Pi blink\n") ;
  wiringPiSetup () ;
  pinMode (0,INPUT );
  pinMode (LED, OUTPUT) ;
  for (;;)
  {
    digitalWrite (LED, HIGH) ;     // On
    delay (500) ;         // mS
    printf("On\n");
    digitalWrite (LED, LOW) ; // Off
    printf("Off\n");
    delay (500) ;
  }
  return 0 ;
```

```
}
```

**Більш складна програма виводу даних**

Програма перемикає контактЮ що заданий користувачем на режим виводу і видає на нього 0 чи 1. Вихід з програми здійснюється коли користувач введе у якості номера контакта -1.

```
#include <stdio.h>
#include <wiringPi.h>
 int PINS=0;
 int ST  =0;
 char x[10];
int main (void)
{  printf ("Raspberry test \n") ;
  wiringPiSetup () ;
  for(;;){
    printf("Enter pin tumber (-1- exit) \n");
    scanf("%i",&PINS);
    if(PINS==-1) break;
    printf("Enter state 1/0\n");
    scanf("%i",&ST);
    ST=ST & 01;
    pinMode (PINS, OUTPUT) ;
    pullUpDnControl(PINS,PUD_OFF);
    digitalWrite (PINS, ST);
    }
  digitalWrite (1, 1);
 return 0 ;
}
```

**Програма вводу даних з перемикачів**

Програма читає стан кнопок S1-S3 та виводить його на дисплей та на світлодіоди

```
#include <stdio.h>
#include <stdint.h>
#include <unistd.h>
#include <stdlib.h>
#include <wiringPi.h>
// uint8_t  - 8 бітовий беззнаковий int з файла stdint.h
const uint8_t IN[3]  ={21,22,23};       //опис входів
const uint8_t OUT[4] ={14,13,12,10};//опис виходів
uint8_t State[3];                   //стан кнопок
void IOStart()
{ for(uint8_t i=0;i<3;i++)
  {  pinMode (IN[i],   INPUT);
     pullUpDnControl(IN[i],PUD_OFF);}
  for(uint8_t i=0;i<4;i++) {  pinMode (OUT[i],   OUTPUT);}
};
int main (void)
{ wiringPiSetup();  IOStart();
  printf("\r\nExit:Cntl-C\r\n State of input pin is...\n\r");
  while(1) {
    for(uint8_t i=0;i<3;i++)
    {State[i]=digitalRead(IN[i]);digitalWrite(OUT[i],State[i]);
     printf("%d ",State[i]);}
    printf("\r"); //повернутись на початок стрічки
    usleep(10000);//заснути на 10 мс
  }
  return 0 ;}
```

**Завдання на лабораторну роботу**

1.1 Завантажити та зібрати програму blink.c на Raspberry Pi.

1.2 Модифікувати програму тамим чином, щоб вихід з неї відбувався за кнопкою, що підключена до порта чи за кнопкою клавіатури, а частота мигання задавалась при запуску програми.

1.3 Модифікувати програму так, щоб мигало всі 4 світлодіоди.

1.4 Скласти звіт по виконаній роботі.

1.5 Відповісти на контрольні запитання.

**Контрольні запитання**

1. Що таке бібліотека WiringPi та як її підключити до проекту?
2. Як налаштувавати режими GPIO ?
3. Які часові функції можна використовувати при роботі з WiringPi?

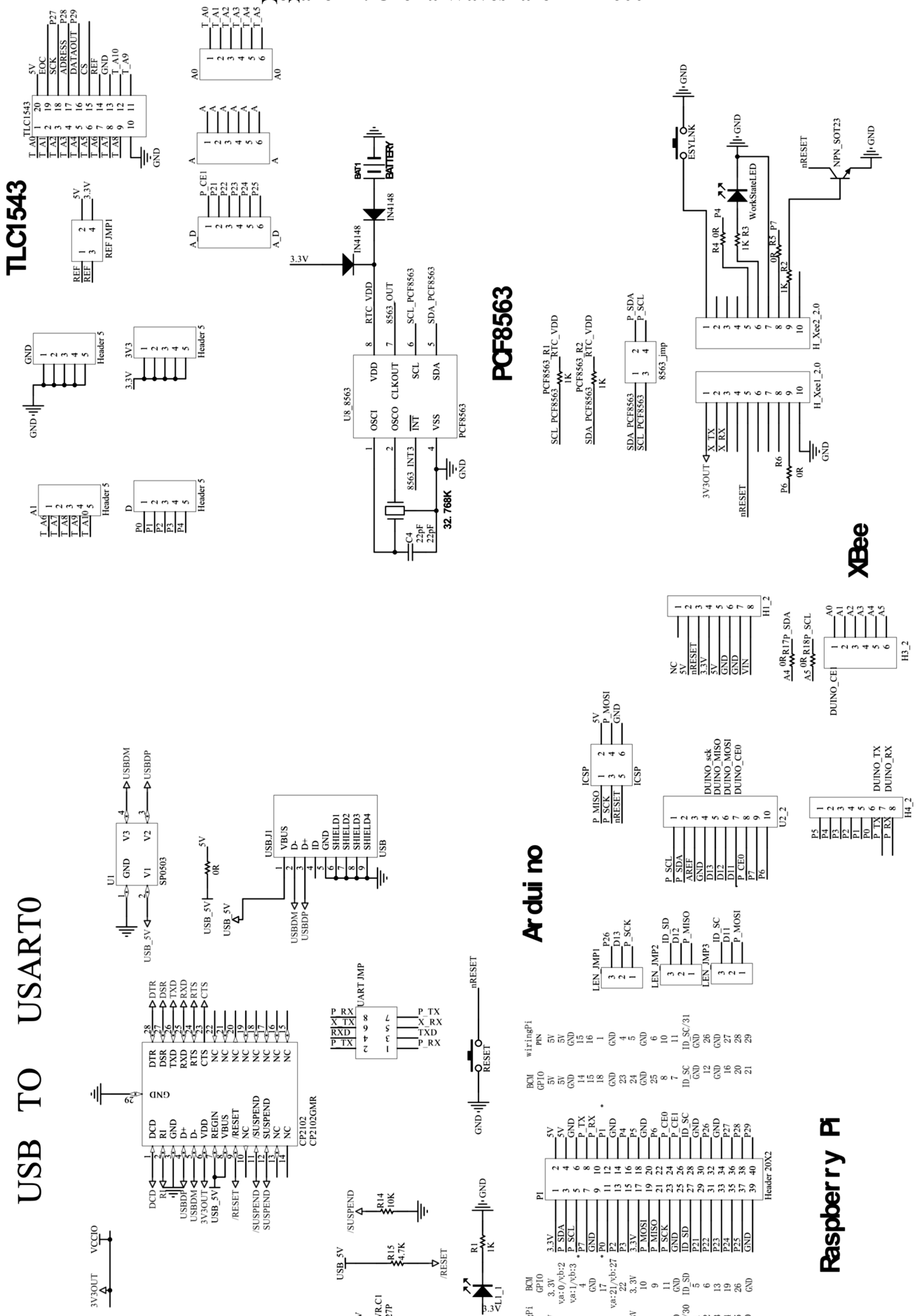
**Література**

1. Иго Т. Arduino, датчики и сети для связи устройств: Пер. с англ. -СПб.: БХВ-Петербург, 2016. - 544 с.

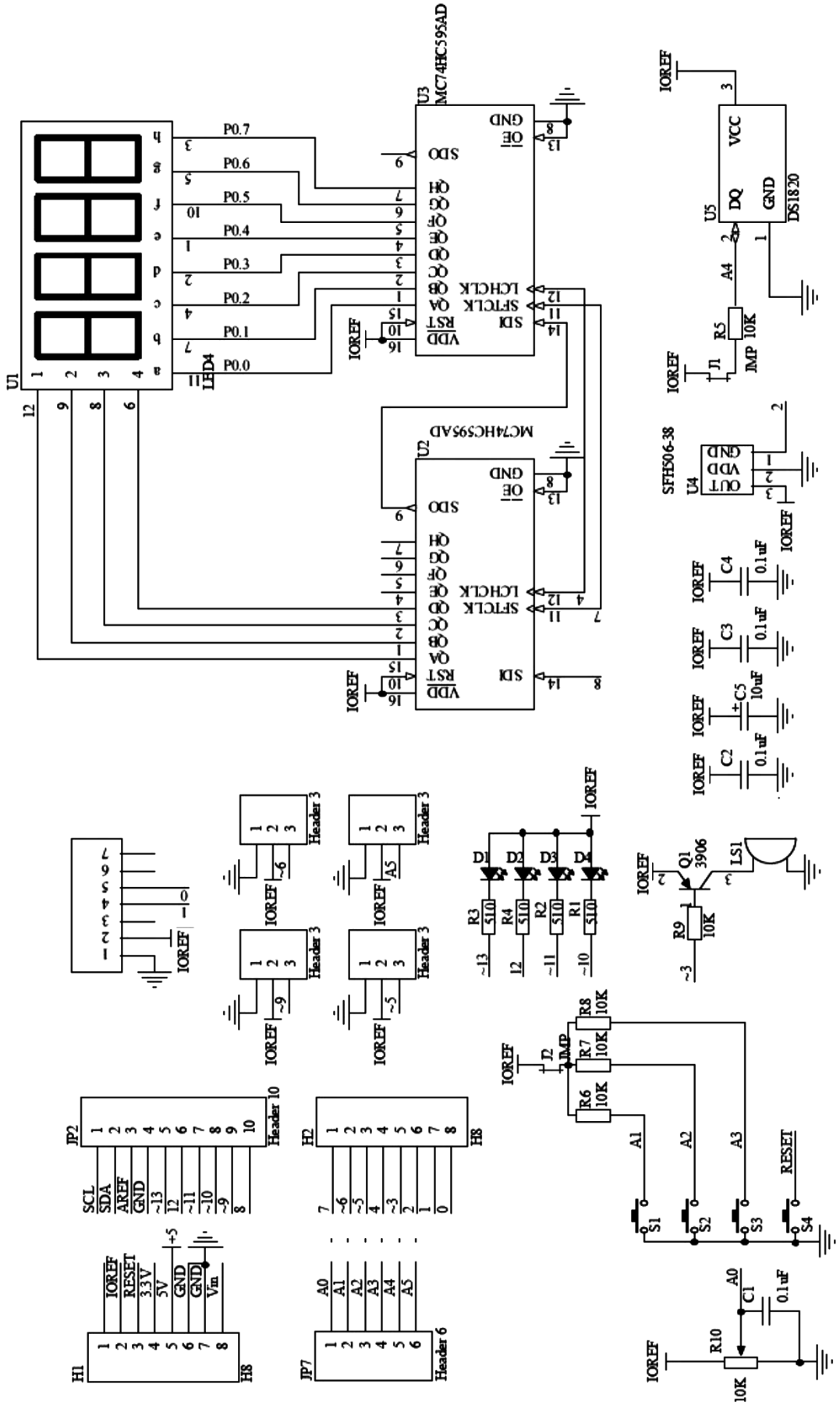
2. Петин В.А. Arduino и Raspberry Pi в проектах Internet of Things. -СПб.: БХВ-Петербург, 2016.-464 с.

3. Петин В.А. Микрокомпьютеры Raspberry Pi.Практическое руководство. СПб.: БХВ-Петербург, 2015. -240 с.

**Додаток 1. Схема Waveshare ARPI600**



**Додаток 2. Схема Multifunction Shield**



**Зміст**