МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

**НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ «ЛЬВІВСЬКА ПОЛІТЕХНІКА»**

Інститут комп'ютерних наук та інформаційних технологій

Кафедра інформаційних систем та мереж



ЗВІТ

про виконання лабораторної роботи № 1

«Налаштування робочого середовища»

з дисципліни

«Вбудовані системи»

Виконала:

студентка групи ІТ-31

Щербак Л. В. Прийняв викладач:

Думич С.С.

Львів 2022

Мета роботи: Встановити необхідні пакети та налаштувати робоче середовище.

Хід роботи:

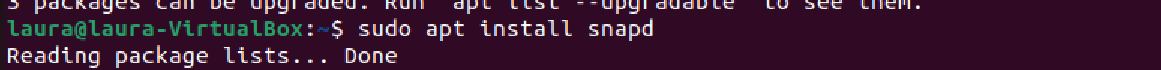
Інсталяція необхідних пакетів для Linux Ubuntu.

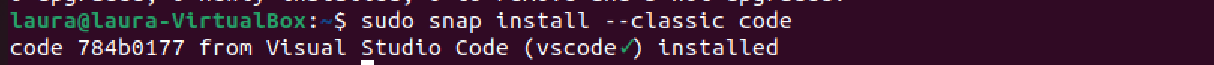
# Інсталяція AVR Toolchain



# Інсталяція VS Code







# Інсталяція додаткових розширень для VS Code





Інсталяція симулятора електронних схем SimulIDE



Створюємо дерево каталогів проекту для Visual Studio Code (VScode) з такою структурою:

/test\_blink

|\_\_ Makefile

|\_\_ main.c

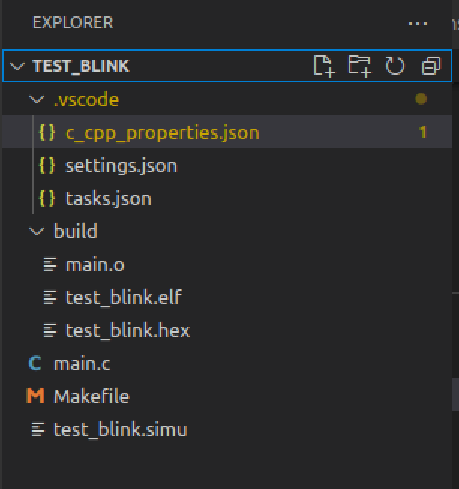
|\_\_ /.vscode

|\_\_ c\_cpp\_properties.json

|\_\_ tasks.json

|\_\_ settings.json

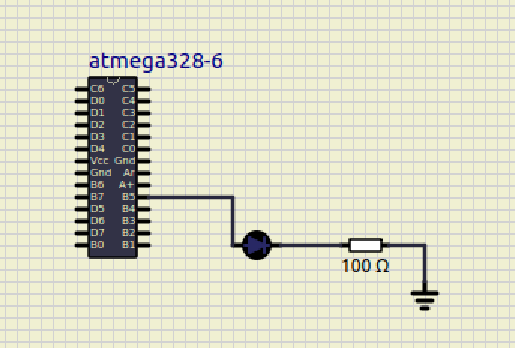
(test\_blink та .vscode є папками (каталогами))

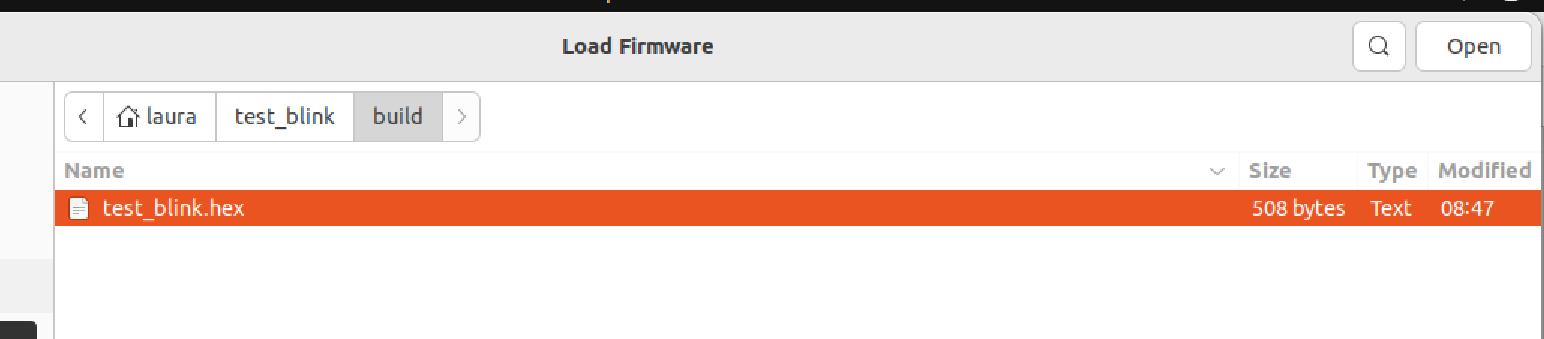


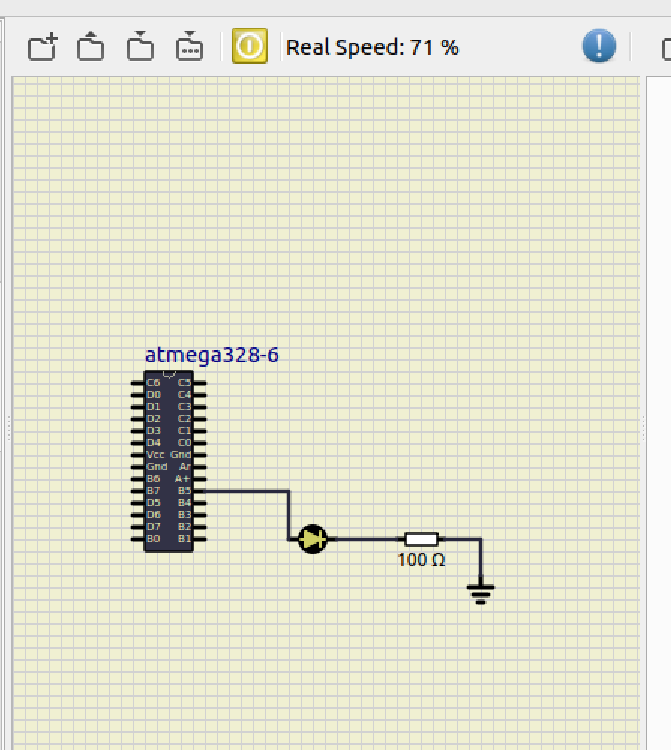
Відкриваємо папку проекту у VSCode вибираємо в меню Terminal -> Run task... -> Build, якщо компіляція пройшла успішно, то у робочому каталозі появиться папка build в якій буде файл з розширенням \*.hex (test\_blink.hex).

Створюємо схему в симуляторі як показано на Рис 1., завантажуємо hex–файл (test\_blink.hex) у модель мікроконтролера Рис 2. та запускаємо моделювання.

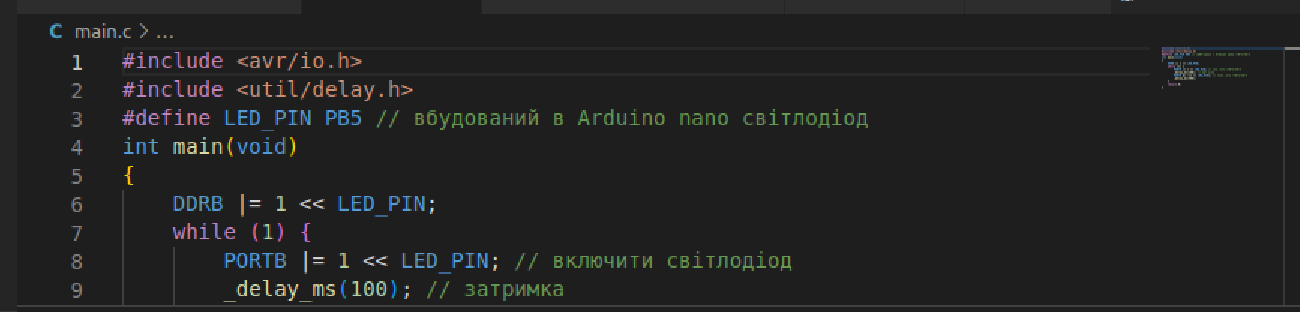
Після перекомпіляції проекту потрібно перезавантажити hex-файл.







Головний файл тестової програми **main.c**



#include <avr/io.h>

#include <util/delay.h>

#define LED\_PIN PB5 *// вбудований в Arduino nano світлодіод*

**int** main(**void**)

{

DDRB |= 1 << LED\_PIN;

**while** (1) {

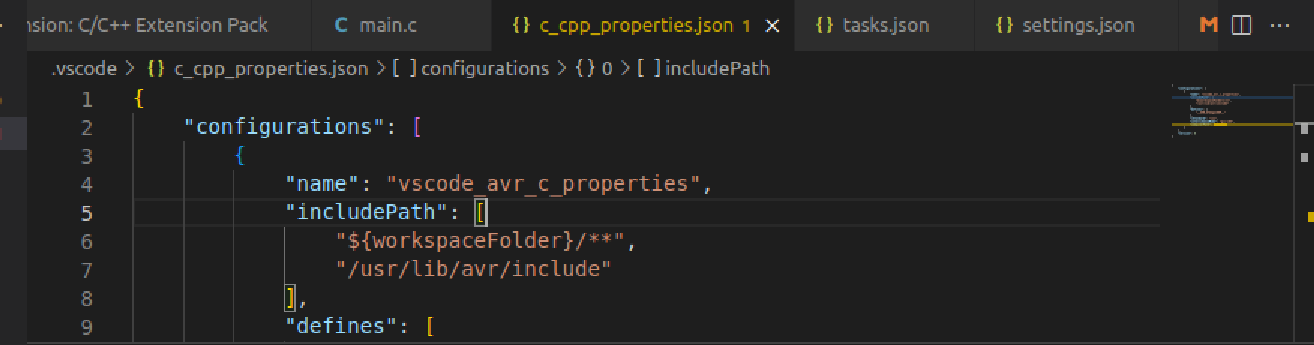
PORTB |= 1 << LED\_PIN; *// включити світлодіод* \_delay\_ms(100); *// затримка* PORTB &= ~(1 << LED\_PIN); *// виключити світлодіод* \_delay\_ms(400);

}

**return** 0;

}

Вміст файлу c\_cpp\_properties.json

{ 

"configurations": [

{

"name": "vscode\_avr\_c\_properties", "includePath": [

"${workspaceFolder}/\*\*",

"/usr/lib/avr/include"

],

"defines": [

"\_\_AVR\_ATmega328P\_\_"

],

"cStandard": "c11",

"intelliSenseMode": "gcc-x64",

"compilerPath": "avr-gcc"

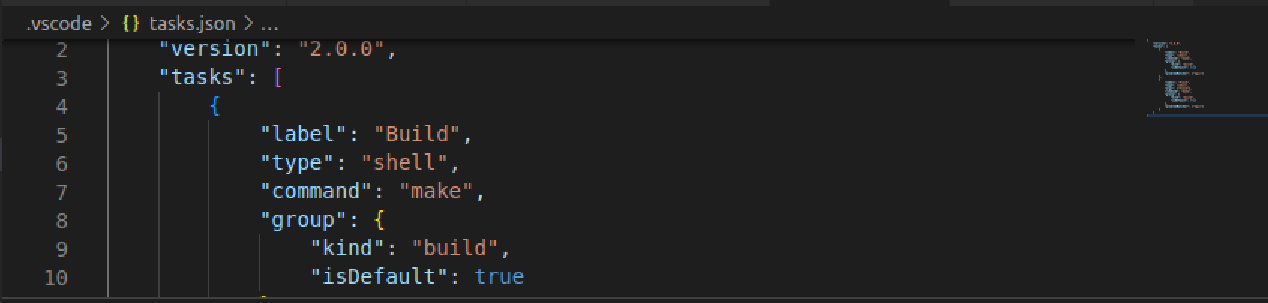
}

],

"version": 4

}

Вміст файлу tasks.json



{

"version": "2.0.0",

"tasks": [

{

"label": "Build",

"type": "shell",

"command": "make",

"group": {

"kind": "build",

"isDefault": true

},

"problemMatcher": ["$gcc"]

]

},

{

"label": "Flash",

"type": "shell",

"args": ["flash"],

"command": "make",

"group": {

"kind": "build",

"isDefault": true

},

"problemMatcher": ["$gcc"]

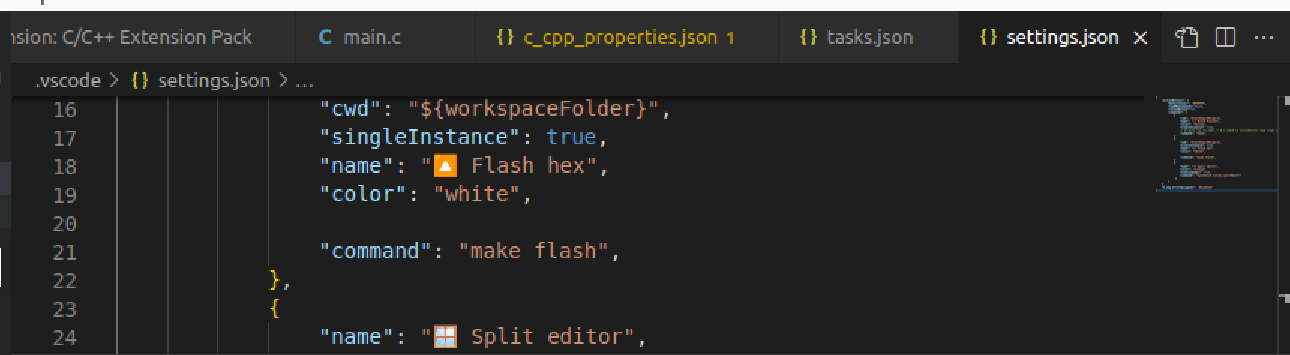
]

}

]

}

Вміст файлу settings.json



{

"actionButtons": {

"defaultColor": "#FFFFFF",

"loadNpmCommands":false,

"reloadButton":"↻",

"commands": [

{

*// Початкова папка терміналу ${workspaceFolder}*

"cwd": "${workspaceFolder}",

"name": "♻️ Build Projekt",

"color": "yellow",

"singleInstance": true,

*// Команда яка виконається при кліку на actionButton* "command": "make",

},

{

"cwd": "${workspaceFolder}",

"singleInstance": true,

"name": "�� Flash hex",

"color": "white",

"command": "make flash",

},

{

"name": "$(split-horizontal) Split editor", "color": "orange",

"useVsCodeApi": true,

"command": "workbench.action.splitEditor" }

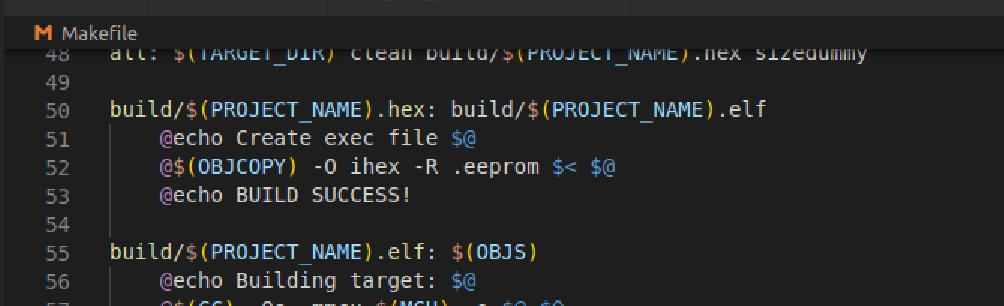
]

},

"C\_Cpp.errorSquiggles": "Disabled"

}

Вміст файлу **Makefile**

****

*# Назва проекту*

PROJECT\_NAME = test\_blink

*# Модель мікроконтролера*

MCU = atmega328p

*# Частота тактового генератора*

F\_CPU = 16000000UL

*# Порт для програмування*

UPLOAD\_PORT=/dev/ttyUSB0

*# Шридкість передачі даних [19200 -(nano atmega168p), 57600 -(nano atmega328p), 115200 -(uno)]*

UPLOAD\_PORT\_BAUD=57600

*# Sources files needed for building the application #SRC = $(wildcard \*.c)*

SRC = main.c

SRC +=

*# The headers files needed for building the application* INCLUDE = -I.

INCLUDE +=

*#--------------------------------------------------------------* PROJECT\_NAME := $(strip $(PROJECT\_NAME))

THIS\_DIR := $(dir $(abspath $(firstword $(MAKEFILE\_LIST))))

*#OBJS:=$(SRC: .c=.o)*

OBJS:= $(addsuffix .o,$(basename $(SRC)))

OBJS:= $(addprefix build/,$(OBJS))

CC = avr-gcc

OBJCOPY = avr-objcopy

AVRDUDE = avrdude

AVRDUDE\_CONF = /etc/avrdude.conf

TARGET\_DIR = build

RM = rm -rf

CFLAGS =-Os

CFLAGS +=-Wall

CFLAGS +=-fpack-struct

CFLAGS +=-fshort-enums

CFLAGS +=-ffunction-sections

CFLAGS +=-fdata-sections

CFLAGS +=-std=gnu99

CFLAGS +=-funsigned-char

CFLAGS +=-funsigned-bitfields

.PHONY: all

all: $(TARGET\_DIR) clean build/$(PROJECT\_NAME).hex sizedummy

build/$(PROJECT\_NAME).hex: build/$(PROJECT\_NAME).elf @echo Create exec file $@

@$(OBJCOPY) -O ihex -R .eeprom $< $@

@echo BUILD SUCCESS!

build/$(PROJECT\_NAME).elf: $(OBJS)

@echo Building target: $@

@$(CC) -Os -mmcu=$(MCU) -o $@ $^

build/%.o: %.c

@echo Compile file: $<

@$(CC) $(CFLAGS) -mmcu=$(MCU) -DF\_CPU=$(F\_CPU) -o $@ -c $<

$(TARGET\_DIR):

mkdir $(TARGET\_DIR)

.PHONY: clean sizedummy

sizedummy:

@echo

@avr-size --format=avr --mcu=$(MCU) build/$(PROJECT\_NAME).elf

flash:

@$(AVRDUDE) -C$(AVRDUDE\_CONF) -F -v -p$(MCU) -carduino -P$(UPLOAD\_PORT) -b$(UPLOAD\_PORT\_BAUD) -D -Uflash:w:build/$(PROJECT\_NAME).hex:i

clean:

**Додаткове завдання.**

Написати програму яка з періодом 1 секунда показує сигнал SOS ( . . . - - - . . . ) на світлодіоді (моргає світлодіодом короткі і довгі імпульси).

#include <avr/io.h>

#include <util/delay.h>

#define LED\_PIN PB5 // вбудований в Arduino nano світлодіод

static const char symbol\_table[2][5] = {

{'.', '.', '.', 0, 0}, {'-', '-', '-', 0, 0}

};

void morzo\_sent\_symbol(char sym)

{

uint8\_t index, i;

if (sym == 's'){

index = 0;

} else {

index = 1;

}

i = 0;

while(symbol\_table[index][i] != 0) {

PORTB |= 1 << LED\_PIN; // включити світлодіод

if (symbol\_table[index][i] == '.') {

\_delay\_ms(200);

} else {

\_delay\_ms(600);

}

PORTB &= ~(1 << LED\_PIN); // виключити світлодіод

\_delay\_ms(500);

i++;

}

}

int main(void)

{

DDRB |= 1 << LED\_PIN;

while (1) {

/\* for (int i=0; i<3; i++) {

PORTB |= 1 << LED\_PIN; // включити світлодіод

\_delay\_ms(200); // затримка

PORTB &= ~(1 << LED\_PIN); // виключити світлодіод

\_delay\_ms(500);

}

for (int i=0; i<3; i++) {

PORTB |= 1 << LED\_PIN; // включити світлодіод

\_delay\_ms(600); // затримка

PORTB &= ~(1 << LED\_PIN); // виключити світлодіод

\_delay\_ms(500);

}

for (int i=0; i<3; i++) {

PORTB |= 1 << LED\_PIN; // включити світлодіод

\_delay\_ms(200); // затримка

PORTB &= ~(1 << LED\_PIN); // виключити світлодіод

\_delay\_ms(500);

}

\*/

morzo\_sent\_symbol('s');

morzo\_sent\_symbol('o');

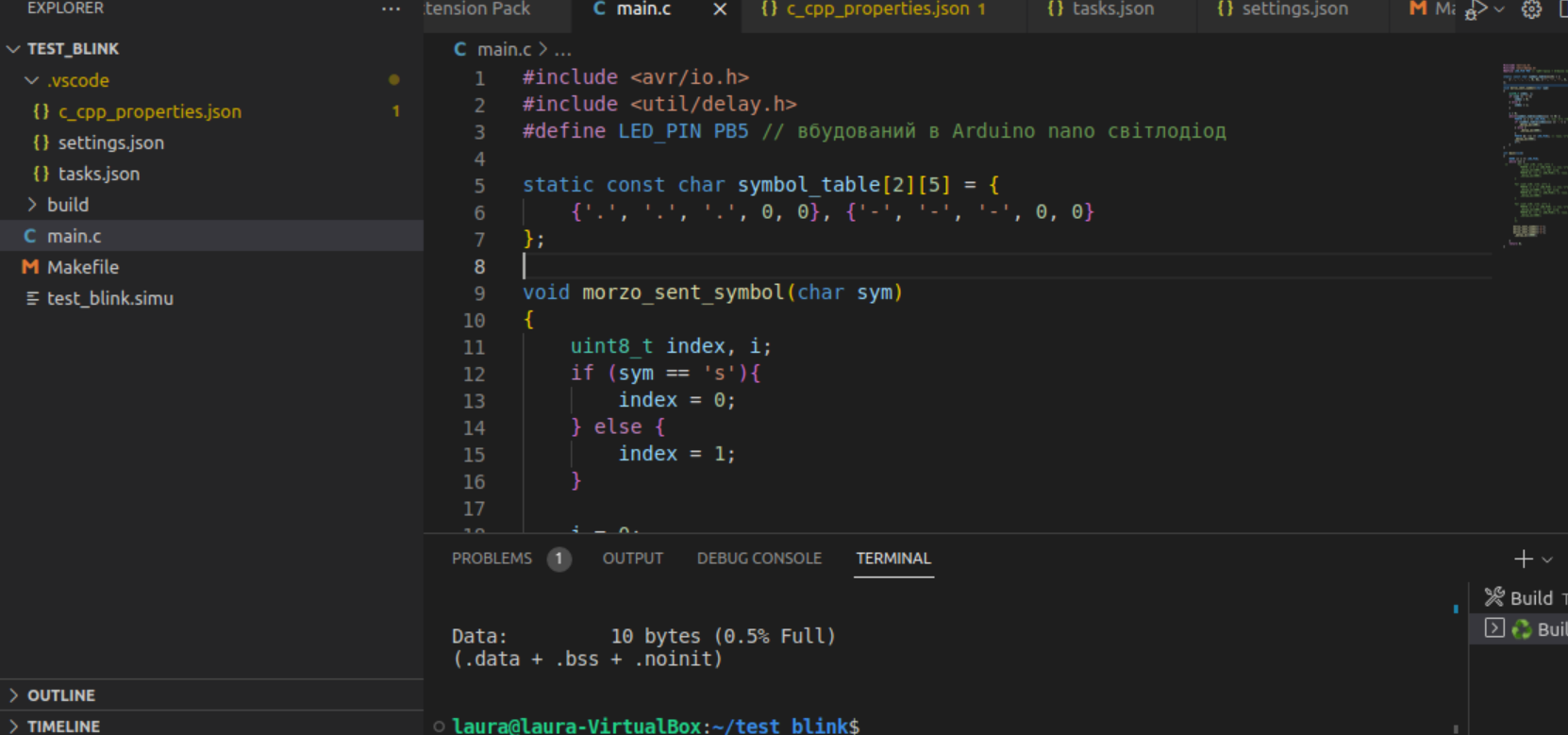
morzo\_sent\_symbol('s');

\_delay\_ms(2000);

}

return 0;

}



Висновок: Встановила необхідні пакети та налаштувала робоче середовище.