Міністерство освіти і науки України

Національний університет водного господарства та природокористування

Кафедра автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій

04-03-349М

МЕТОДИЧНІ ВКАЗІВКИ

до виконання практичних робіт з навчальної дисципліни **«Мікропроцесорні системи та програмування мікропроцесорних засобів»** для здобувачів вищої освіти першого (бакалаврського) рівня за освітньо-професійними програмами «Робототехніка та штучний інтелект» та «Автоматизація та комп'ютерно-інтегровані технології» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» та «Електроенергетика, електротехніка та електромеханіка» спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» денної та заочної форм навчання

Рекомендовано науково-методичною радою з якості ННІАКОТ
Протокол №10 від
20 вересня 2022 р.

Рівне – 2022

Методичні вказівки до виконання практичних робіт з навчальної дисципліни «Мікропроцесорні системи та програмування мікропроцесорних засобів» для здобувачів вищої освіти першого (бакалаврського) рівня за освітньо-професійними програмами «Робототехніка та штучний інтелект» та «Автоматизація та комп'ютерно-інтегровані технології» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології» та «Електроенергетика, електротехніка та електромеханіка» спеціальності 141 «Електроенергетика, електротехніка та електромеханіка» денної та заочної форм навчання [Електронне видання] / Реут Д. Т. – Рівне : НУВГП, 2022. – 19 с.

Укладач: Реут Д. Т., к.т.н., доцент кафедри автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій.

Відповідальний за випуск: Древецький В. В., д.т.н., професор, завідувач кафедри автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій.

Керівник освітньої програми «Робототехніка та штучний інтелект» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології»: Сафоник А. П., д.т.н., професор кафедри автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій
Керівник освітньої програми «Автоматизація та комп'ютерно-інтегровані технології» спеціальності 151 «Автоматизація та комп'ютерно-інтегровані технології»: Христюк А. О., к.т.н., доцент кафедри автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій
Керівник освітньої програми «Електроенергетика, електротехніка та електромеханіка» спеціальності 141 «Електроенергетика, електротехніка та електромеханіка»: Василець С. В., д.т.н., професор кафедри автоматизації, електротехнічних та комп'ютерно-інтегрованих технологій

Зміст

Вступ

Практична роботи з навчальної дисципліни «Мікропроцесорні системи та програмування мікропроцесорних засобів» дають змогу освоїти основні прийоми розробки й програмування мікропроцесорних систем і пристроїв на базі мікроконтролерів Microchip ATmega328P і STMicroelectronics STM32F072C8.

У практичних роботах розглянуто:

– використання цифрових портів вводу-виводу, обробка апаратних переривань;

– застосування таймерів для генерування ШІМ-сигналу керування двигуном постійного струму;

– цифрову фільтрацію результатів АЦП;

– використання сторожового таймера та режимів зниженого енергоспоживання мікроконтролера;

– використання таймера розширеного керування для формування сигналів заданої форми;

– прийом та передача даних в мережах RS-485 за допомогою модуля USART.

Практична робота 1. Структури програм для мікроконтролерів. Обробка апаратних переривань INT0/INT1

Мета роботи: ознайомитись з типовими структурами програм для мікроконтролерів; навчитись обробляти апаратні переривання від зовнішніх джерел INT0/INT1.

Теоретичні відомості

Типові структури програм для мікроконтролерів показані на рис. 1-4.

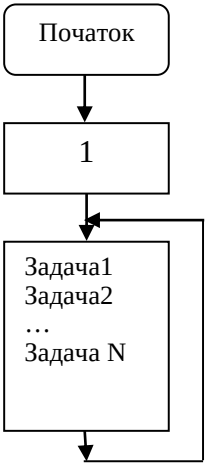

Рис. 1. Циклічна структура програми МК:
1 – настроювання МК та периферійних пристроїв, оголошення змінних.

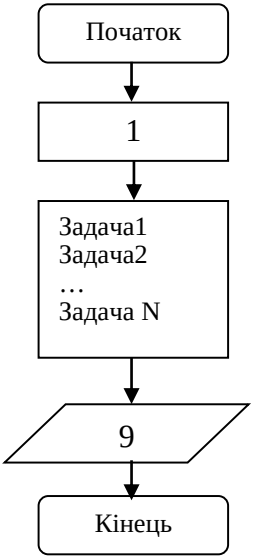

Рис. 2. Структура програми з кінцевим результатом і завершенням роботи програми: 9 - вивід результатів.

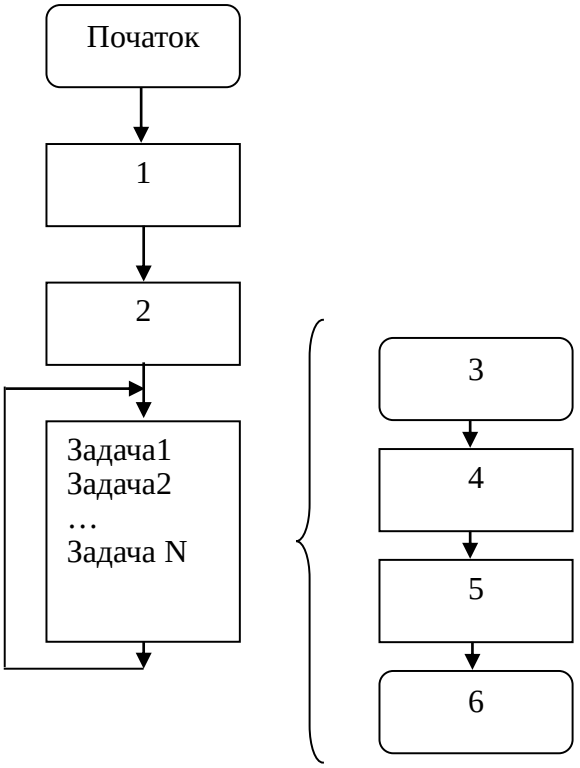

Рис. 3. Циклічна програма з обробкою переривань:
1 - настроювання МК та периферійних пристроїв, оголошення змінних;
2 - настроювання переривань;
3 - початок обробки переривання;
4 - визначення джерела та пріоритету переривання;
5 - здійснення обробки переривання;
6 - вихід з обробки переривання.

Переривання (англ. interrupt) — сигнал, що повідомляє мікроконтролер про настання якої-небудь події, яка потребує невідкладної уваги щодо її обробки.

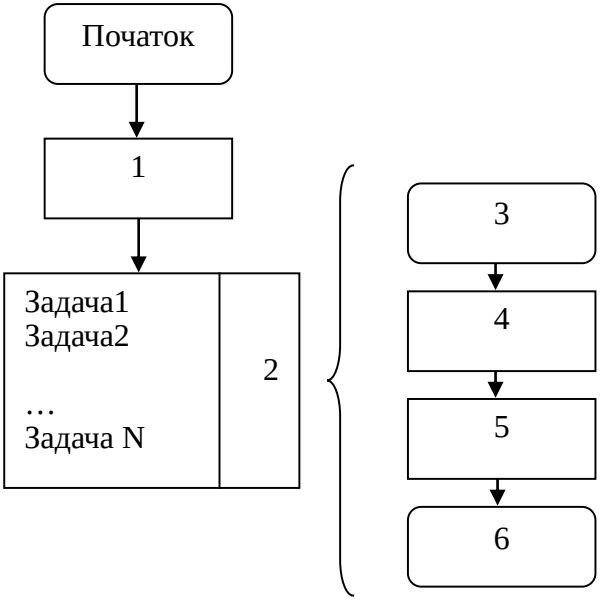

Рис. 4. Багатозадачна структура з використанням ОС:
1 - завантаження ОС;
2 - планувальник задач
3 - початок обробки переривань;
4 - визначення джерела та пріоритету переривання;
5 - алгоритм обробки переривання;
6 - вихід з обробки переривання;

При цьому виконання поточної послідовності команд призупиняється, а керування передається обробнику переривання, який реагує на подію та обслуговує її, після чого повертає керування в перерваний код. Всі джерела переривань описані в документації на мікроконтролер.

Щоб ззовні просигналізувати мікроконтролеру про настання якоїсь події, використовують входи зовнішніх переривань INT0, INT1, … Можна налаштувати

мікроконтролер на генерування переривань по передньому фронту (переходу з 0 в 1), задньому фронту (переходу з 1 в 0), будь-якому фронту, а також по низькому логічному рівні (лог. 0) на відповідному контакті INTx. У мікроконтролері ATmega328P зовнішні переривання INTx налаштовуються в регістрах EICRA та EIMSK (див. п. 12.2 в datasheet).

Завдання до виконання практичної роботи

1. Написати програму з нескінченим циклом, яка блимає світлодіодом з частотою 2 Гц.

2. Налаштувати контакт мікроконтролера INT1 на вхід. Підключити кнопку до контакту мікроконтролера INT1 так, щоб при натиснутій кнопці на вході мікроконтролера була напруга низького логічного рівня (лог. 0), а відпущеній – високого (лог. 1). Дозволити переривання глобально викликом *sei()* та налаштувати генерування переривання INT1 по низькому логічному рівню на відповідному вході. В обробнику переривання (вектор *INT1_vect*) вмикати світлодіод, а нескінчений цикл залишити без змін. Перевірити реакцію складеної схеми на натискання кнопки.

3. Змінити затримки в нескінченому циклі так, щоб світлодіод світився 500 мс, а був погашений 4500 мс. Змінити налаштування переривань так, щоб світлодіод засвічувався лише в момент спадання напруги від високого до низького логічного рівня, що відповідає натисканню кнопки. Оцінити зміни в реакції складеної схеми на натискання кнопки.

4. Видалити оператор циклу в програмі, залишивши в main() лише кілька послідовних блимань світлодіода. Оцінити зміни в роботі програми.

Практична робота 2. Керування двигуном постійного струму з використанням таймерів-лічильників

Мета роботи: навчитись використовувати таймери-лічильники AVR-мікроконтролера для генерування ШІМ-сигналів керування двигуном постійного струму.

Теоретичні відомості

Мікроконтролер ATmega328P містить два 8-розрядні таймери та один 16-розрядний. Всі таймери мають регістри OCRxy, значення з яких може використовуватись для задання тривалості ШІМ-сигналу. Таймер 1 має додатково регістр захоплення OCR1, куди записується вміст таймера TCNT1 по події захоплення - зміні сигналу на вході ICP1 або виході аналогового компаратора. Вибір режиму роботи здійснюється у регістрах TCCR1A та TCCR1B (див. п. 15.11 в datasheet).

В ШІМ-режимі таймер використовується для генерування ШІМ-сигналу на відповідних виводах мікроконтролера. Відповідний біт настроювання виводу мікроконтролера DDRxy повинен бути встановлений в 1 для настроювання виводу на вихід.

В режимі Fast PWM значення в лічильному регістрі таймера збільшується й при рівності вмісту OCRx вихід OCx буде скинутий в 0, а значення продовжить рости до моменту, коли таймер переповниться. При цьому вихід OCx буде встановлений в 1 та рахунок почнеться спочатку. Таким чином передподільник таймера задає період ШІМ, а вміст OCRx - тривалість імпульсу (рис. 5).

Завдання до виконання практичної роботи

1. Написати програму з нескінченим циклом, яка впродовж 10 с збільшує змінну від 0 до 255. Кожні наступні 10 с значення змінної знову починає зростати від

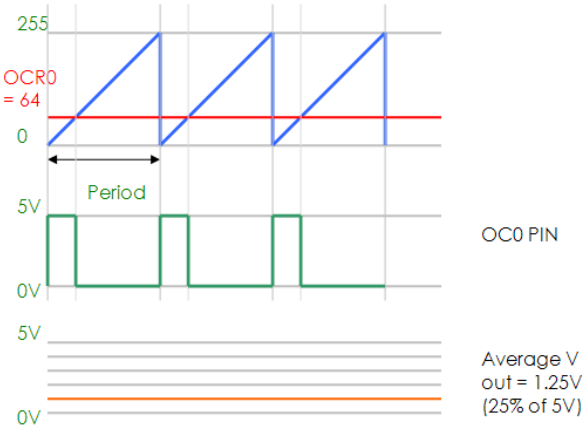

Рис. 5. Генерування ШІМ-напруги таймером

нуля. Надалі ця змінна буде використана як тривалість імпульсу.

2. Налаштувати таймер 1 на роботу в режимі швидкого ШІМ з частотою не менше 5 кГц. Підключити світлодіод до відповідного виводу мікроконтролера, що генерує ШІМ сигнал, і переконатись в циклічній зміні яскравості.

3. Підключити до мікроконтролера двигун постійного струму через польовий транзистор, що виконуватиме роль силового ключа. Переконатись у зміні швидкості обертання двигуна постійного струму.

Практична робота 3. Фільтрація аналогових сигналів мікроконтролером

Мета роботи: навчитись використовувати АЦП AVR-мікроконтролера для зчитування аналогових сигналів;

навчитись застосовувати цифрові фільтри до результатів аналогово-цифрового перетворення.

Теоретичні відомості

Аналогово-цифровий перетворювач перетворює вхідний аналоговий сигнал в цифровий код (цифровий сигнал). У мікроконтролері ATmega328P є 10-розрядний АЦП. Вхідна напруга конвертується в 10-розрядне двійкове значення. Мінімальне значення відповідає 0 В, а максимальне - опорній напрузі V_{ref}. В даному мікроконтролері є 3 варіанти опорних напруг (напруга живлення, зовнішня опорна напруга, внутрішнє джерело 1,1 В), які перемикаються парою бітів при налаштуванні АЦП. Отриманий результат перетворення записується в 2 регістра: ADCH і ADCL. Результат перетворення можна розрахувати за формулою

$$ADC = \frac{V_{in} \cdot 1023}{V_{ref}}$$

Налаштування АЦП зводиться до наступних дій:

- налаштування регістра ADMUX (регістр настройки мультиплексора АЦП),
- налаштування регістра ADCSRA (регістр статусу і контролю А),
- за потреби, налаштування ADCSRB (регістр статусу і контролю В).

Результат перетворення часто містить шуми, для усунення яких можуть використовуватись апаратні або програмні фільтри. Найчастіше використовують експоненційний фільтр, ковзне середнє та середнє по черговій групі результатів.

Щоб реалізувати останнє, кожен результат АЦП додається до загальної суми. Після надходження n-го результату сума ділиться на n і це буде вихідним значенням фільтра. Після обчислення виходу змінна суми обнуляється

і починається обчислення середнього наступної групи результатів.

Для реалізації ковзного середнього в масиві зберігають останні *n* результатів аналогово-цифрового перетворення, при одержанні нового результату він додається в масив, масив зміщається на один елемент, найстарший результат видаляється з нього. З надходженням кожного результату обчислюється середнє елементів масиву – вихід фільтра.

Експоненційний фільтр першого порядку (аперіодична ланка) програмно може бути реалізований так:

$$y(n)=a*x(n)+(1-a)*y(n-1),$$

де $0<a<1$ — коефіцієнт фільтрації, x – вхід, y – вихід.

Завдання до виконання практичної роботи

1. Написати програму, що зчитує напругу з каналу ADC0, застосовує експоненційний фільтр першого порядку й виводить отримане значення на дисплей, використаний в лабораторній роботі

2. Подати на вхід стрибкоподібну зміну напруги й відстежити зміну вихідного значення при різних значеннях коефіцієнта фільтрації *a*.

3. Замінити експоненційну фільтрацію усередненням і оцінити відмінності в отриманих результатах.

Практична робота 4. Використання режимів зниженого енергоспоживання та сторожового таймера

Мета роботи: навчитись використовувати режими зниженого енергоспоживання та сторожовий таймер AVR-мікроконтролера.

Теоретичні відомості

Сторожовий таймер (WDT) - незалежний таймер, що тактується від окремого генератора й при переповненні може викликати скидання мікроконтролера або переривання. Час переповнення — від 15 мс до 8 с.

Заголовний файл для роботи з watchdog timer

#include <avr/wdt.h>

Увімкнення сторожового таймера здійснюється функцією

wdt_enable(WDTO_x);

де WDTO_x: WDTO_15MS, WDTO_30MS, WDTO_60MS, WDTO_120MS, WDTO_250MS, WDTO_500MS, WDTO_1S, WDTO_2S, WDTO_4S, WDTO_8S

Щоб сторожовий таймер не переповнився і не перезавантажив мікроконтролер, його потрібно періодично скидати в 0 функцією

wdt_reset();

Обробник переривання від сторожового таймера реалізується у функції

ISR(WDT_vect) { … }

Режими зниженого енергоспоживання (режими сну, sleep modes) дозволяють зменшити споживання струму мікроконтролером зупинкою тактування або відключенням частин мікросхеми. В цих режимах тактування обчислювального ядра та пам’яті програм відключено, відповідно після виконання інструкції sleep наступна інструкція в програмі виконається лише після пробудження мікроконтролера. Вивести мікроконтролер з режиму сну можуть переривання, вказані в документації.

Заголовний файл для роботи з режимами сну

#include <avr/sleep.h>

Вибір режиму сну можна виконати функцією

set_sleep_mode(mode);

де mode: SLEEP_MODE_IDLE, SLEEP_MODE_ADC, SLEEP_MODE_PWR_DOWN, SLEEP_MODE_PWR_SAVE, SLEEP_MODE_STANDBY, SLEEP_MODE_EXT_STANDBY

Дозволити перехід в режим сну можна функцією

sleep_enable();

Виконати перехід в режим сну (інструкцію sleep) -

sleep_cpu();

Заборонити перехід у режим сну -

sleep_disable();

Завдання до виконання практичної роботи

1. Написати програму, яка налаштовує сторожовий таймер в режим перезапуску програми при спрацюванні. Час спрацювання – 8 с. Головний цикл оформити так, щоб при відпущеній кнопці світлодіод блимав із зростаючим періодом, в кінці циклу виконувалось скидання сторожового таймера, а при натиснутій – світлодіод горів постійно й скидання таймера не виконувалось. Перед початком нескінченого циклу двічі блимнути світлодіодом. Скласти схему й прошити програму. Впевнитись у перепуску програми сторожовим таймером при натисканні кнопки на тривалий час.

2. Написати програму, яка більшу частину часу триматиме мікроконтролер в режимі сну з найбільшим енергозбереженням, а по натисканні кнопки вмикатиме навантаження на 10 с, після чого вимикатиме та знову переходитиме в режим сну. Скласти схему, прошити програму й виміряти мультиметром струм споживання.

3. Написати програму, яка з періодом 0,5 мс перевірятиме, чи стан виводів PB3=1, PC1=0, PD5=1 й вмикатиме навантаження на 5 с. В протилежному випадку переходитиме в режим сну. Скласти схему, прошити програму й виміряти мультиметром струм споживання.

Практична робота 5. Використання таймера TIM1 для генерування трифазної системи напруг

Мета роботи: навчитись використовувати розширений таймер TIM1 мікроконтролера сімейства STM32 для генерування трифазної системи напруг.

Теоретичні відомості

Таймер TIM1 в мікроконтролерах STM32 – це таймер розширеного керування. Його можна використовувати для різноманітних цілей, у тому числі для вимірювання тривалості імпульсів вхідного сигналу (захоплення) або генерування вихідних сигналів (порівняння, ШІМ, комплементарний ШІМ із вставкою «мертвого» часу). Довжину імпульсу та період можна змінювати від кількох мікросекунд до кількох мілісекунд з використанням передподільника таймера. Таймер має комплементарні виходи, що дозволяє одному каналу керувати одразу двома силовими ключами (один – підключення до позитивного полюса живлення, інший – до негативного). Трьох каналів ШІМ з комплементарними виходами достатньо, щоб керувати 6-ма ключами інвертора й генерувати трифазну напругу (рис. 6).

Якщо обрано режим рахунку вгору, вміст таймера збільшується до значення, вказаного в регістрі ARR, після чого таймер скидається в 0, а якщо режим рахунку вниз –

зменшується до 0, після чого таймер завантажується вмістом регістра ARR.

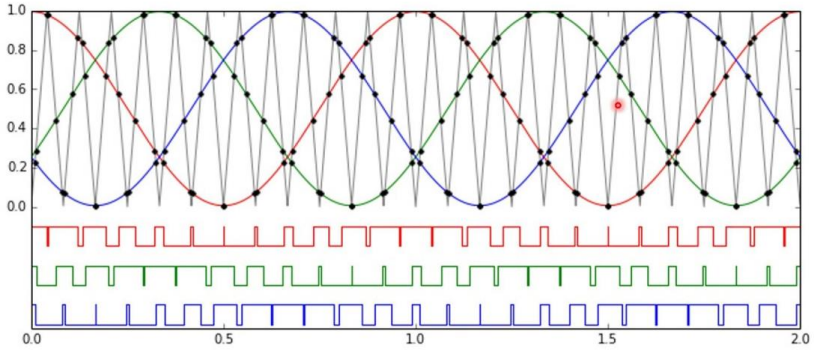

Рис. 6. Відповідність вмісту таймера (чорна лінія), синусоїд трьох фаз (вгорі) та ШІМ-напруги (внизу)

Завдання до виконання практичної роботи

1. Налаштувати в STM32CubeMX період ШІМ (вирівнювання по центру), використовуючи передподільник таймера TIM1 і регістр ARR, так, щоб частота ШІМ дорівнювала 5 кГц. Задати сталу ненульову тривалість імпульсу для всіх трьох каналів. Перевірити наявність сигналу на відповідних трьох виходах.

2. Сформувати масиви значень регістрів CCR1, CCR2, CCR3 для трьох фаз так, щоб максимум синусоїди відповідав значенню в регістрі ARR, а мінімум – нулю. Розмір масивів – 100 елементів. Передбачити зсув фаз на 120^0.

3. Налаштувати DMA так, щоб значення з масивів циклічно записувались в регістри CCR1, CCR2, CCR3 таймера TIM1. Для цього у налаштуваннях таймера увімкнути функцію auto-reload preload, обрати кільцевий режим DMA, розмір даних – word. Для запуску ШІМ з використанням DMA використати функцію

*HAL_TIM_PWM_Start_DMA(TIM_HandleTypeDef *htim, uint32_t Channel, uint32_t *pData, uint16_t Length).* Перевірити, чи змінюється сигнал на 3 виходах.

4. Передбачити в програмі можливість зміни частоти трифазної системи напруг кнопками «збільшити», «зменшити». Перевірити реакцію на натискання кнопок і за потреби налагодити програму.

Практична робота 6. Обмін даними за допомогою трансивера RS-485 та модуля USART

Мета роботи: навчитись використовувати модуль USART мікроконтролера для прийому та передачі даних в мережах RS-485.

Теоретичні відомості

RS-485 – напівдуплексний інтерфейс, відповідно в один момент часу дані можуть передаватись лише в одному напрямку. Для задання напрямку роботи трансивера (прийом або передача) RS-485 багато з них мають окремі виводи RE (Receiver enable, ввімкнення приймача) і DE (Driver enable, ввімкнення передавача) (рис. 7).

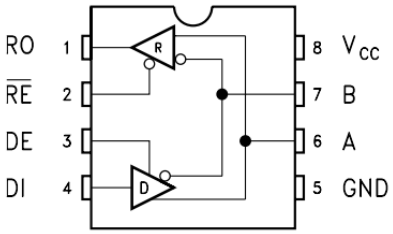

Рис. 7. Виводи трансивера ST485B

Оскільки активний рівень сигналів DE та RE протилежний, можна використовувати один вихід мікроконтролера для керування напрямком передачі:

низький рівень активує приймач, а високий – передавач. Відповідно мікроконтролер під час передачі даних повинен утримувати логічну одиницю на цьому контакті, а решту часу утримувати логічний нуль, працюючи на прийом. Реалізувати це можна програмно, виставляючи лог. 1 перед заповненням буфера передачі й лог. 0 по події закінчення передачі, або використати контакт RTS апаратного керування потоком.

Завдання до виконання практичної роботи

1. Налаштувати модуль USART на швидкість передачі 115200 біт/с, 8 біт даних, доповнення до парності, один стоп-біт в асинхронному режимі, переривання по прийому символа. Налаштувати керування сигналами DE та RE трансивера інтерфейсу RS-485.

2. Реалізувати передачу рядка “Test RS-485” на комп’ютер (використати програму-монітор порта).

3. В обробнику переривання по прийому передбачити запис прийнятого байту в змінну, яка задає яскравість світлодіода. Перевірити, чи змінюється яскравість світлодіода при надсиланні різних значень з комп’ютера.

Інформаційні ресурси

1. ATmega328/P AVR MCU with picoPower Technology Data Sheet. URL: http://ww1.microchip.com/downloads/en/DeviceDoc/ATmega328_P%20AVR%20MCU%20with%20picoPower%20Technology%20Data%20Sheet%2040001984A.pdf .

2. RM0091 Reference manual. URL: https://www.st.com/resource/en/reference_manual/dm0003193

6-stm32f0x1stm32f0x2stm32f0x8-advanced-armbased-32bit-mcus-stmicroelectronics.pdf .

3. AVR Libc Home Page. URL: http://www.nongnu.org/avr-libc/.

4. Custom Signal generation using PWM and DMA. URL: https://community.st.com/s/article/Custom-signal-generation-using-pwm-and-dma .