

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»

В. В. Шликов

МІКРОПРОЦЕСОРНА ТЕХНІКА

Практикум

*Рекомендовано Методичною радою КПІ ім. Ігоря Сікорського
як навчальний посібник для студентів,
які навчаються за спеціальністю 163 «Біомедична інженерія» та 152 «Метрологія та
інформаційно-вимірвальна техніка»*

Київ
КПІ ім. Ігоря Сікорського
2018

Рецензенти: *Лебедєв В.О., д.т.н., професор*
Соломін А.В., к.ф.-м.н, доцент

Відповідальний
редактор *Зубчук В.І., к.т.н., доцент*

*Гриф надано Методичною радою КПІ ім. Ігоря Сікорського (протокол № 1 від 27.09.2018 р.)
за поданням Вченої ради факультет біомедичної інженерії (протокол № 1 від 07.09.2018 р.)*

Електронне мережне навчальне видання

Шликов Владислав Валентинович, канд. техн. наук, доцент

МІКРОПРОЦЕСОРНА ТЕХНІКА

Мікропроцесорна техніка: Практикум [Електронний ресурс]: навч. посіб. для студ. спеціальності 163 «Біомедична інженерія» та 152 «Метрологія та інформаційно-вимірювальна техніка»/ В.В. Шликов; КПІ ім. Ігоря Сікорського. – Електронні текстові данні (1 файл: 3,1 Мбайт). – Київ : КПІ ім. Ігоря Сікорського, 2018. – 144 с.

Практикум з дисципліни «Мікропроцесорна техніка» містить практичні заняття та лабораторні роботи, що виконуються за допомогою програмного забезпечення в середовищі програмування мікропроцесорів серії Arduino UNO (Arduino Genuino), лабораторних макетів вимірювальної апаратури з використанням програмного та інформаційного забезпечення засобів National Instruments LabVIEW. Навчальне видання рекомендовано для вищих навчальних закладів України з викладанням спеціальностей 163 «Біомедична інженерія» та 152 «Метрологія та інформаційно-вимірювальна техніка».

© В. В. Шликов, 2018
© КПІ ім. Ігоря Сікорського, 2018

ЗМІСТ

Вступ.....	3
1. Загальні відомості з платформи Arduino	4
1.1. Мікроконтролери на платі Arduino UNO.....	6
1.2. Середовище розробки Arduino	13
1.3. Сполучення Arduino з LabVIEW	16
1.4. Структура і особливості програми.....	26
2. Вимірювальні прилади з мікропроцесорним управлінням.....	35
2.1. Цифровий сенсорний датчик	36
2.2. Універсальний звуковий датчик.....	43
2.3. Управління кроковим двигуном 28BYJ-48	48
2.4. Датчик швидкості з цифровим і аналоговим виходами	57
2.5. Датчик вологості і температури HTU21	64
2.6. Цифровий модуль з термістором.....	71
2.7. Ультразвуковий датчик відстані HC-SR04.....	78
2.8. Цифровий датчик перешкоди	84
2.9. Цифровий датчик Холу KY-03	89
2.10. Датчик нахилу на LM393	94
2.11. Датчик алкоголю MQ-3	98
2.12. Цифровий датчик рівня рідини.....	104
2.13. Цифровий датчик вогню.....	112
2.14. Аналогово-цифровий датчик освітленості	118
2.15. Датчик вібрації SW-420.....	122
2.16. Цифровий датчик удару	128
3. Плата для підключення датчиків.....	136
3.1. Багатофункціональний навчальний шилд	136
Література	143

Вступ

Метою практичних занять та лабораторних робіт з дисципліни «Мікропроцесорна техніка», виконуваних за допомогою програмного забезпечення в середовищі програмування мікропроцесорів Arduino UNO (Arduino Genuino), лабораторних макетів вимірювальної апаратури з використанням програмного та інформаційного забезпечення засобів National Instruments LabVIEW 2010, є:

- вивчення принципів побудови електронних компонентів мікропроцесорних систем на основі процесорів Arduino UNO;
- керування мікропроцесорними системами із застосуванням інтерфейсу I2C і візуальних компонентів NI VISA;
- придбання навичок роботи з сенсорами та датчиками з мікропроцесорним управлінням;
- придбання навичок підготовки, проведення та документування результатів вимірювання і функціонування мікропроцесорних систем.

В результаті виконання лабораторних робіт студент повинен вміти формулювати вимоги до параметрів мікропроцесорної системи (МПС). Для проектування МПС необхідно навчитись враховувати наступні основні характеристики:

- можливості мікропроцесорної системи;
- обсяг ПЗУ програм і ОЗУ даних МПС;
- набір команд і способів адресації;
- розрядність і швидкодія;
- вимоги до джерела живлення і споживаної потужності;
- вартість МПС в різних варіантах виконання;
- ефективність засобів програмування і налагодження МПС з використанням програмного та інформаційного забезпечення National Instruments LabVIEW 2010.

Завдання на лабораторні дослідження розраховані на 2 академічні години. Результати проведених досліджень і вимірювань повинні бути задокументовані і в кінці заняття представлені викладачу. За виконаної лабораторної роботи складається звіт, який повинен містити:

1. Титульний аркуш звіту.

2. Теоретичні відомості з теми дослідження.
3. Схеми МПС, на яких проводилися дослідження і вимірювання.
4. Схеми у LabVIEW, на яких проводилися налагодження МПС.
5. Програмний код, що ілюструє досліджені процеси.
6. Висновки за результатами досліджень і вимірів.
7. Відповіді на контрольні питання.

1. Загальні відомості з платформи Arduino

Важливою особливістю систем на платформі Arduino, що відрізняє їх від звичайних комп'ютерів, є те, що в їх роботі ключовим фактором є ефективність. Це означає, що не завжди достатньо просто виконувати потрібну функцію, щоб вирішувати задачу. Можна сказати, що спосіб вирішення повинен бути нестандартним. Наприклад, він повинен працювати швидко, або він повинен працювати з низьким енергоспоживанням, або він повинен бути дешевим. І в цьому насправді полягає велика різниця між розробкою мікропроцесорних систем і, наприклад, традиційним дизайном програмного забезпечення для комп'ютерів.

На рис. 1.1 схематично зображено вигляд загальної схеми. Мікропроцесорна система (МПС) повинна отримувати дані з зовнішнього світу, обробляти їх і потім виводити дані у зовнішній світ. Так що, в першу чергу, вона має набір датчиків для прийому даних. Ці датчики можуть отримувати інформацію про зовнішній світ по-різному, тому існує багато різних типів подібних пристроїв.

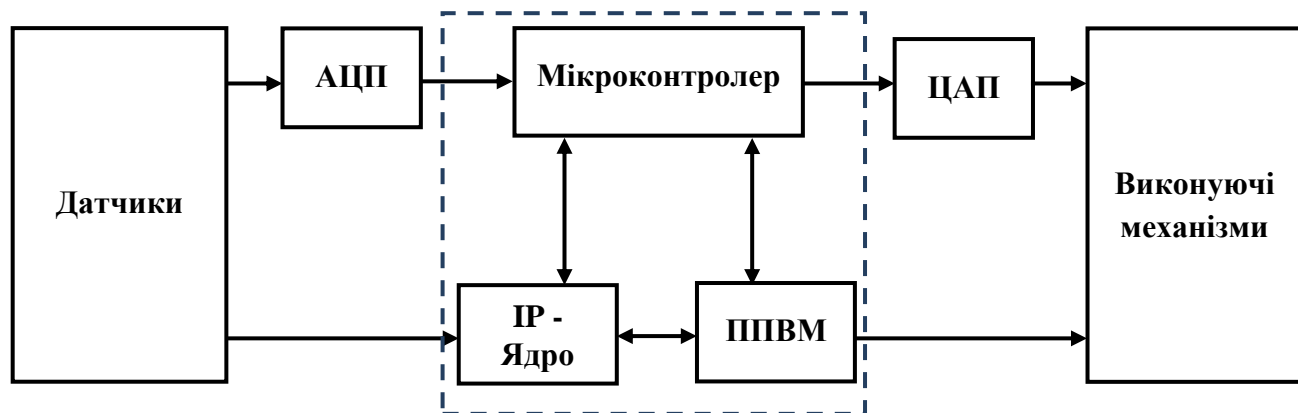


Рис. 1.1. Пристрій вбудованих систем

Найпростіший тип датчика - це кнопка або щось подібне, яка отримує дані в дуже простому вигляді: натиснута або не було натиснуто. Система може отримувати і звукову інформацію, використовуючи мікрофон. Відеокамери – це теж датчики, які отримують інформацію у вигляді зображення. Сенсорний екран дозволяє не тільки виводити, а й отримувати інформацію. І так далі – існує велика безліч найрізноманітніших типів датчиків, через які МПС може отримувати інформацію.

Отже, спочатку інформація надходить в систему і далі вона потрапляє в її ядро, яке займається її обробкою. В кінці цього процесу, коли система вирішила, що робити з інформацією або яке рішення потрібне прийняти на її основі, МПС повинна видати якісь результати. Це робиться за допомогою виконавчих механізмів або приводів, які показані в правій частині схеми на рис. 1.1. Виконавчим механізмом може бути світлодіод, який може горіти або блимати. Наприклад, він може сигналізувати, що відеокамера записує, або що закінчується живлення. Але є інші виконавчі механізми. Наприклад, всередині камери є електромотори, за допомогою яких відбувається управління об'єктивом і наведення різкості. Ці двигуни є виконавчими механізмами, а їх рух – це вихід системи.

Існує безліч різних типів виконавчих механізмів: динаміки відтворюють звук, лампи дозволяють системі виводити світло, TFT екрани. Таким чином, МПС отримує дані від датчиків, обробляє отриману інформацію, а потім посилає відповідні сигнали на виконавчі пристрої, щоб змусити їх щось зробити в реальному часі у відповідь на дані, які вона отримала. У цьому сенсі МПС забезпечує зв'язок між датчиками і виконавчими механізмами.

У центрі МПС знаходиться мікроконтролер. Крім мікроконтролеру можуть бути і інші компоненти. Два з тих, які зустрічаються досить часто, показані на рис. 1.1 – це ІР-ядра і ППВМ. ІР це англійське скорочення від Intellectual Property, тобто інтелектуальний продукт. ІР-ядро являє собою готові блоки для проектування пристроїв, наприклад, це може бути готова мікросхема, яка виконує одну функцію або кілька функцій, що тісно пов'язані між собою, орієнтовані на якусь конкретну задачу. Тобто цей блок не є універсальною програмованою мікросхемою загального призначення. Найчастіше ІР-ядра роблять для поширених завдань – для тих, які виконуються знову і знову і на які є великий попит. Якщо якась задача зустрічається тільки в одній конкретній системі, то для неї не має сенсу робити спеціалізоване ІР-ядро.

Розглянемо, наприклад, контролери Texas Instruments. Такі фірми випускають безліч спеціалізованих мікросхем. Наприклад, якщо потрібно реалізувати стиснення за стандартом MPEG, то у виробника Texas Instruments ви знайдете кілька десятків мікросхем, які його роблять, і вибрати відповідну.

IP-ядра повинні взаємодіяти з мікроконтролером, який є центром всієї системи. Він керує IP-ядрами і командує, коли вони повинні починати роботу, передає їм інформацію і отримує результат. Наприклад, якщо IP-ядро виконує стиснення відео, то мікроконтролер вказує йому, коли починати стискати дані і які дані стискати. Для цього мікроконтролер передає йому певну послідовність сигналів, а коли мікросхема закінчує свою роботу вона посилає певні сигнали мікроконтролеру, повідомляючи, що результат готовий.

Інший вид компонентів, який застосовується у МПС, це програмовані вентиляльні матриці, або ППВМ. Це досить складні апаратні пристрої, а якщо бути точніше – апаратно програмовані мікросхеми. Звичайна програмована мікросхема складається з величезної кількості побудованих на базі транзисторів логічних вентилів. Вентилі, в свою чергу, з'єднані між собою дуже складними численними зв'язками, за якими, власне, і надходять керуючі та інформаційні сигнали. Схема об'єднання цих логічних пристроїв і обумовлює логіку виконуваних мікросхемою операцій.

У свою чергу ППВМ можна образно порівняти з набором декількох десятків транзисторів, резисторів, лампочок, кнопок, проводів, клем, приводів і інших елементів, який повинен виконувати якийсь спеціальне завдання. Таким чином, ППВМ – це складна мікросхема, яка дозволяє змінювати малюнок взаємозв'язків своїх логічних компонент, тобто по суті апаратну конфігурацію пристрою. Тут важливо ще раз підкреслити, що мова йде не про перепрошивку програмної начинки пристрою, а про зміну його апаратної структури.

Таким чином, конфігурація кожної МПС залежить від розв'язуваної задачі, під яку вона розробляється і підбирається найбільш ефективна апаратна конфігурація пристрою.

1.1. Мікроконтролери на платі Arduino UNO

Мікроконтролер є центром МПС. На рис. 1.1.1 представлено одну з плат Arduino. Це не мікроконтролер, це друкована плата з безліччю елементів, серед яких, можна побачити

велику чорну прямокутну мікросхему. Ось ця мікросхема і є мікроконтролер, який виконує програми, що керують пристроєм.

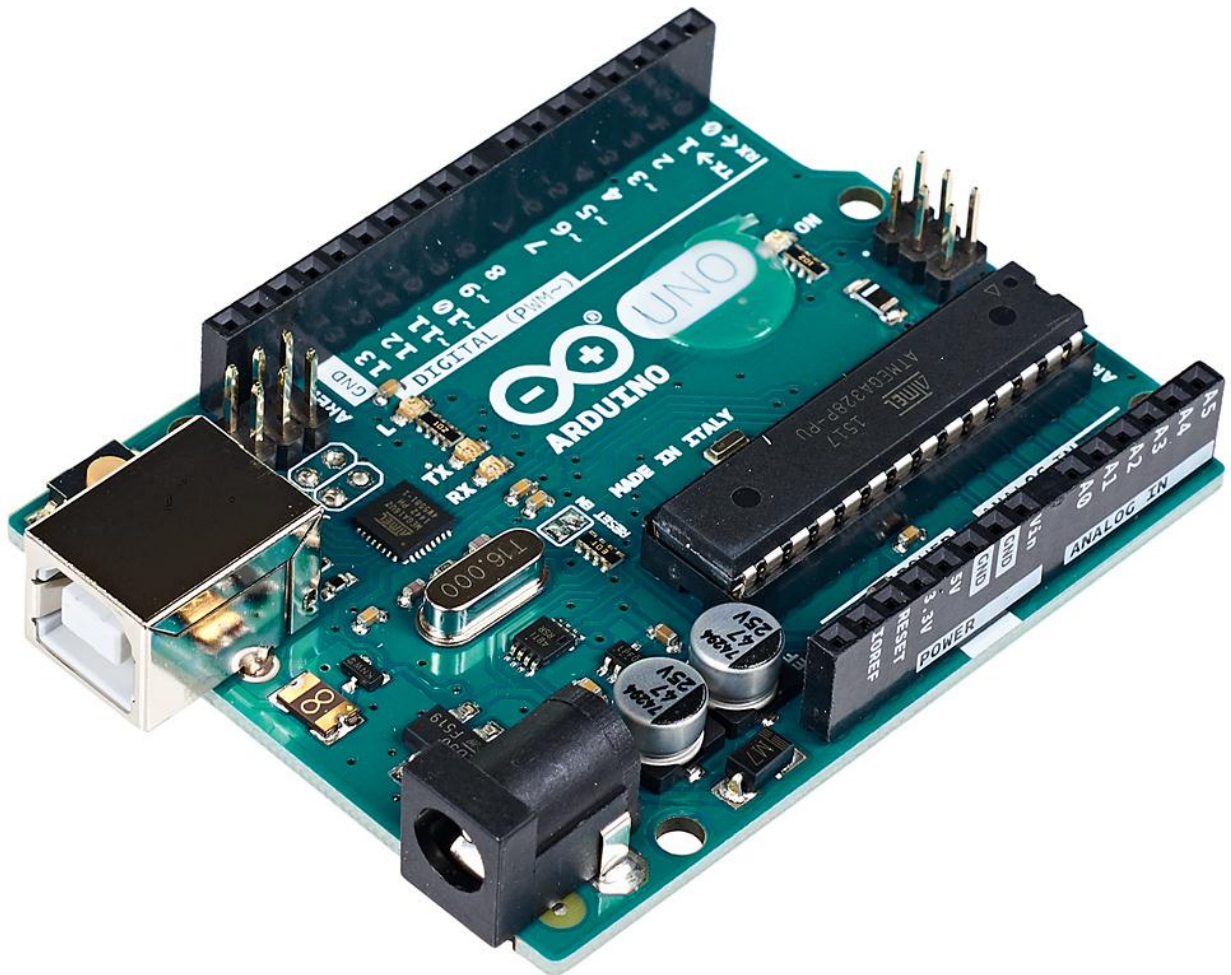


Рис. 1.1.1. Плата Arduino UNO

Отже, робота даної мікросхеми полягає в тому, щоб виконувати код, і це центр всієї системи. Мікроконтролер зчитує дані з інших компонентів, і він також керує іншими компонентами. Всі компоненти МПС повинні бути енергоефективними та дешевими і для них, найчастіше, не потрібна велика продуктивність як у процесорів з 6 ядрами і частотою від 1 ГГц. Якщо потрібно реалізувати конкретну взаємодію з користувачем, то така продуктивність швидше за все не потрібна, тому для МПС зазвичай потрібен дешевий процесор, що споживає мало енергії.

Зараз досить часто в МПС використовується частота від 16 МГц, хоча може бути навіть і менше, наприклад, 8 або навіть 4 МГц. Це майже в кілька сот разів повільніше, ніж процесор звичайного комп'ютера. Звичайно, у МПС можуть застосовувати і більш швидкі мікросхеми – до 500 МГц або навіть до 1 ГГц. Це потрібно, наприклад, для обробки аудіоданих і відеозображень. Інша відмінність МПС від звичайних комп'ютерів полягає в тому, що в звичайних комп'ютерах процесор і пам'ять – це окремі мікросхеми, причому зазвичай пам'ять являє собою кілька мікросхем. Ми можемо змінювати і додавати пам'ять незалежно від процесора. У мікроконтролерах початкового рівня все об'єднано в одну мікросхему, тобто на одній мікросхемі знаходиться і мікропроцесор, і пам'ять, і інші необхідні компоненти.

У більш продуктивних мікроконтролерах використовують зовнішню пам'ять, яка встановлюється окремо. Це менш зручно, але дозволяє вибрати такий обсяг пам'яті, скільки потрібно для роботи конкретної МПС. Зазвичай так роблять, коли мікроконтролеру для роботи потрібно досить великий обсяг пам'яті.

Апаратну архітектуру мікропроцесора міняти не можна, зате в його пам'ять можна завантажити програму у вигляді набору інструкцій, написаних на зрозумілій йому мові. Існує багато мов програмування, на яких можна писати подібні програми. Для програмування мікроконтролера, що встановлений у МПС на платі Arduino UNO (рис. 1.2) використовується мова Сі.

Перш ніж виконати, програму, очевидно, треба зберегти код. Отже, всередині мікроконтролера повинна бути пам'ять для програми. Зараз це, як правило, флеш-пам'ять, такого ж типу, як в USB накопичувачах. Флеш-пам'ять – це незалежна пам'ять, тобто така, з якої дані не стирається після відключення живлення. Коли на МПС подається живлення, мікроконтролер починає виконання програми з самого початку.

Програми для подібних пристроїв зазвичай пишуть на звичайному комп'ютері. Це пов'язано з тим, що мікроконтролери порівняно повільні і слабкі. Тому програму пишуть і компілюють на потужному комп'ютері, а потім остаточна версія компільованої програми записується в пам'ять мікроконтролера. Робиться це за допомогою спеціального пристрою – програматор (рис. 1.1.2).

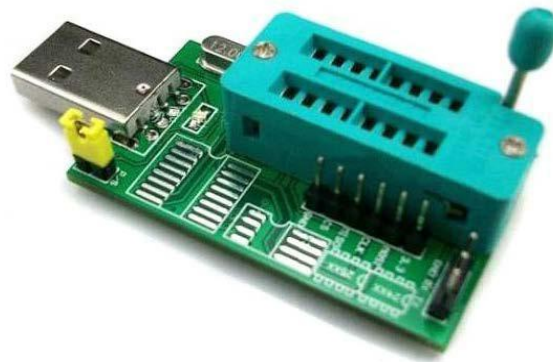


Рис. 1.1.2. Програматор

Програматор – це невелика плата, яка за допомогою порту USB приєднується до комп'ютера, а до розташованого на ній слоту підключається мікроконтролер. Є спеціальні програми на комп'ютері, які зв'язуються з цим пристроєм і записують через нього програму в пам'ять мікроконтролера. Це один із способів запрограмувати мікроконтролер, за яким мікроконтролер програмують окремо за допомогою програматора, а потім він вже встановлюють в МПС, де буде працювати. Іноді буває так, що в МПС вже передбачений спеціальний роз'єм для програмування, і тоді мікроконтролер програмують прямо в системі. Такий спеціальний роз'єм для програмування використовується в платі Arduino UNO. Для неї не потрібен окремий програматор, так як він вже вбудований в плату. Можна просто взяти МПС на основі Arduino і, підключивши її до USB порту комп'ютера (ПК), запрограмувати її безпосередньо в середовищі програмування Arduino UNO (Arduino Genuino).

Загальні відомості з МПС на платі Arduino UNO. Плата Arduino Uno – це пристрій на основі мікроконтролера ATmega328 (рис. 1.1.3 і рис. 1.1.4). У його склад входить все необхідне для зручної роботи з мікроконтролером: 14 цифрових входів / виходів (з них 6 виводів можуть використовуватися в якості ШІМ-виходів), 6 аналогових входів, кварцовий резонатор на 16 МГц, роз'єм USB, роз'єм живлення, роз'єм для внутрішнього схемного програмування (ICSP) і кнопка скидання. Для початку роботи з МПС необхідно подати живлення від AC / DC-адаптера, або підключити його до комп'ютера за допомогою USB-кабелю. Для роботи з платою Arduino Uno в операційній системі Windows необхідно встановити на комп'ютер інтегроване середовище розробки Arduino – Arduino IDE (Integrated Development Environment).

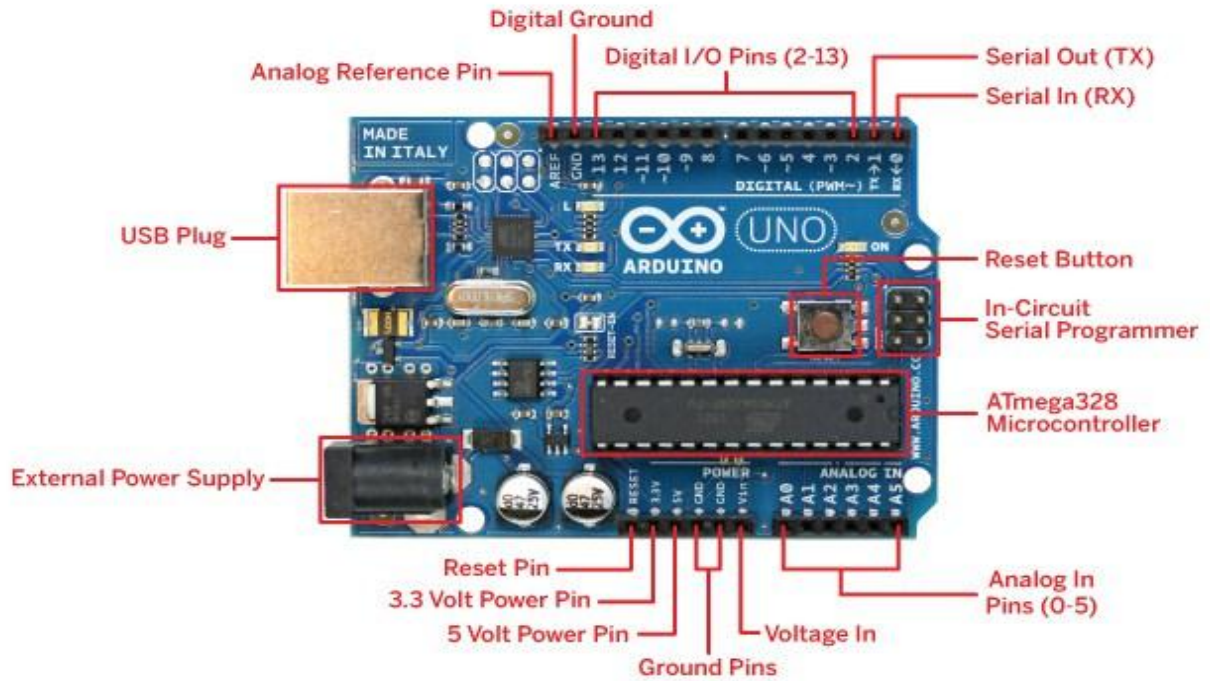


Рис. 1.1.3. Опис елементів плати Arduino UNO

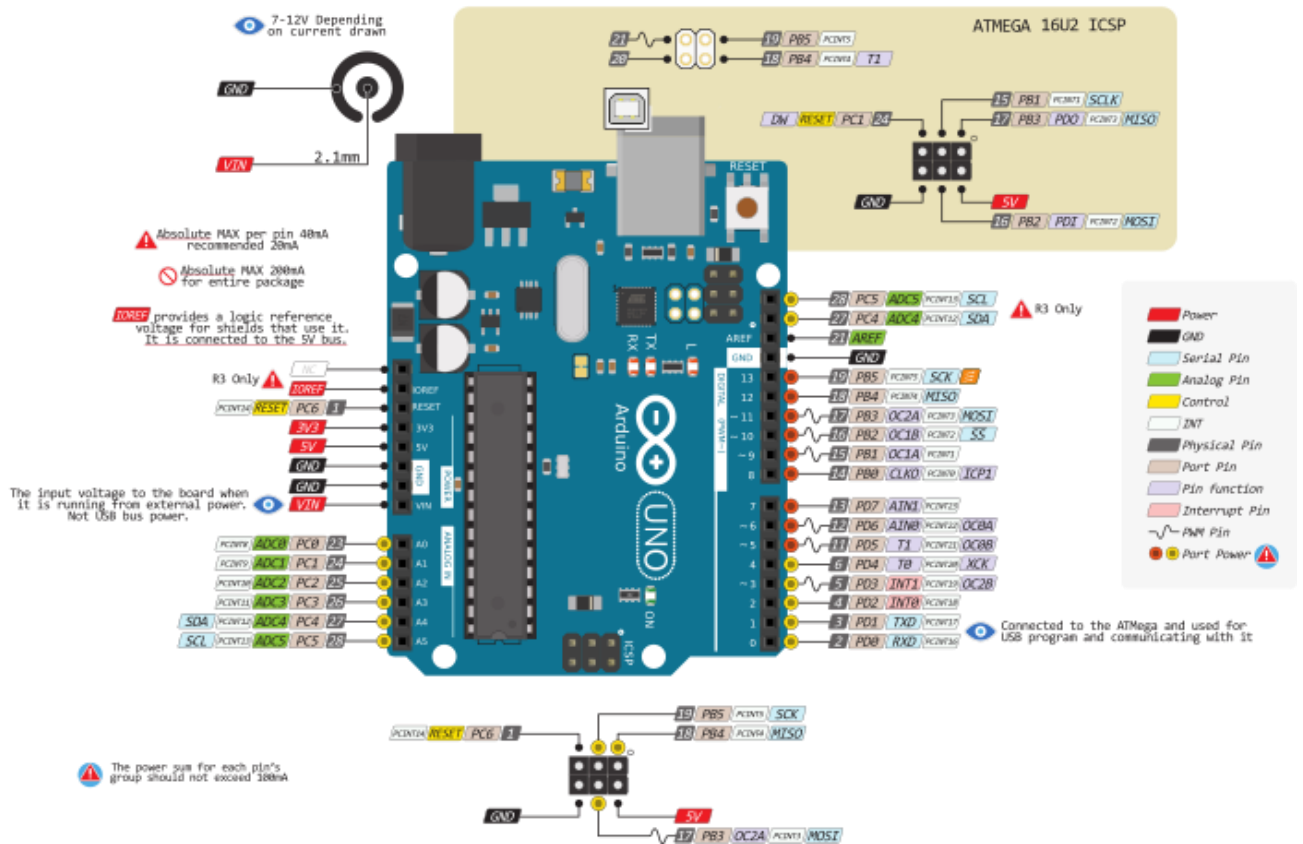


Рис. 1.1.4. Призначення виводів плати Arduino UNO

Характеристики плати Arduino UNO:

Мікроконтролер	ATmega328
Робоча напруга	5В
Напруга живлення (рекомендована)	7-12В
Напруга живлення (гранична)	6-20В
Цифрові входи / виходи	14 (6 у якості ШІМ-виходи)
Аналогові входи	6
Максимальний струм одного виведення	40 мА
Максимальний вихідний струм виводу 3.3В	50 мА
Flash-пам'ять	32 КБ (ATmega328) з яких 0.5 КБ загрузчик
SRAM	2 КБ (ATmega328)
EEPROM	1 КБ (ATmega328)
Тактова частота	16 МГц

Входи і виходи плати Arduino UNO:

– Послідовний інтерфейс: виходи 0 (RX) і 1 (TX), що використовуються для отримання (RX) і передачі (TX) даних по послідовному інтерфейсу. Ці виводи з'єднані з відповідними виводами мікросхеми ATmega8U2 на платі Arduino UNO, яка виконує роль перетворювача USB-UART.

– Зовнішні переривання: виводи 2 і 3, які можуть служити джерелами переривань, що виникають при фронті, спаді або при низькому рівні сигналу на цих виводах.

– ШІМ-виходи 3, 5, 6, 9, 10 і 11 – інтерфейс SPI: висновки 10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK).

– Світлодіод: 13 представляє вбудований світлодіод, приєднаний до висновку 13. При відправці значення HIGH світлодіод включається, при відправці LOW - вимикається.

– 6 аналогових входів (A0 - A5), кожен з яких може уявити аналогову напругу у вигляді 10-бітного числа (1024 різних значення). За замовчуванням, вимір напруги здійснюється щодо діапазону від 0 до 5 В.

На платі Arduino UNO розташовані ще кілька додаткових виводів:

- Reset: формування низького рівня (LOW) на цьому виводу призведе до перезавантаження мікроконтролера. Зазвичай цей висновок служить для функціонування кнопки скидання на платах розширення.

- 3V3: напруга на даному виводі +3.3 В, що генерується вбудованим регулятором на платі. Максимальне споживання струму 50 мА. Від цього виводу можуть живитися деякі апаратні модулі.

- 5V: напруга на даному виводу +5 В, що генерується вбудованим регулятором на платі Arduino UNO.

- GND: земля, або загальний мінус. Цей вивід є другим обов'язковим виводом для будь-якого іншого пристрою.

- Vin: використовується для подачі живлення від зовнішнього джерела в відсутності живлення 5 В від роз'єму USB, наприклад, коли необхідно запустити плату Arduino окремо від комп'ютера.

Крім цього, деякі з аналогових входів мають додаткові функції:

- TWI / I2C: вивід A4 або SDA і вивід A5 або SCL.

- AREF: опорна напруга, на якому працює один з вузлів мікроконтролера – аналогово-цифровий перетворювач, який перетворює напругу в вольтів у числа.

Рівень напруги на виводах обмежений 5В, максимальний струм, який може віддавати або споживати один висновок, становить 40 мА. Всі виводи пов'язані з внутрішніми налаштовуються резисторами номіналом 20-50 кОм.

Зовнішнє живлення (не від USB) може подаватися через роз'єм Vin або роз'єм XP14 плати розширення. Платформа може працювати при зовнішній напрузі від 6 В до 20 В, але рекомендується використовувати напругу в діапазоні 7 - 12 В для запобігання перегріву або нестабільної роботи.

Деякі виводи плати Arduino UNO можуть виконувати додаткові функції:

- З використанням функцій pinMode(), digitalWrite() і digitalRead() кожен з 14 цифрових виходів може працювати в якості входу або виходу.

- За допомогою функції analogWrite() можуть виводитися 8-бітові аналогові значення в вигляді ШІМ-сигналу.

- Опорна напруга 5В для аналогових входів AREF може бути задіяна функцією analogReference().

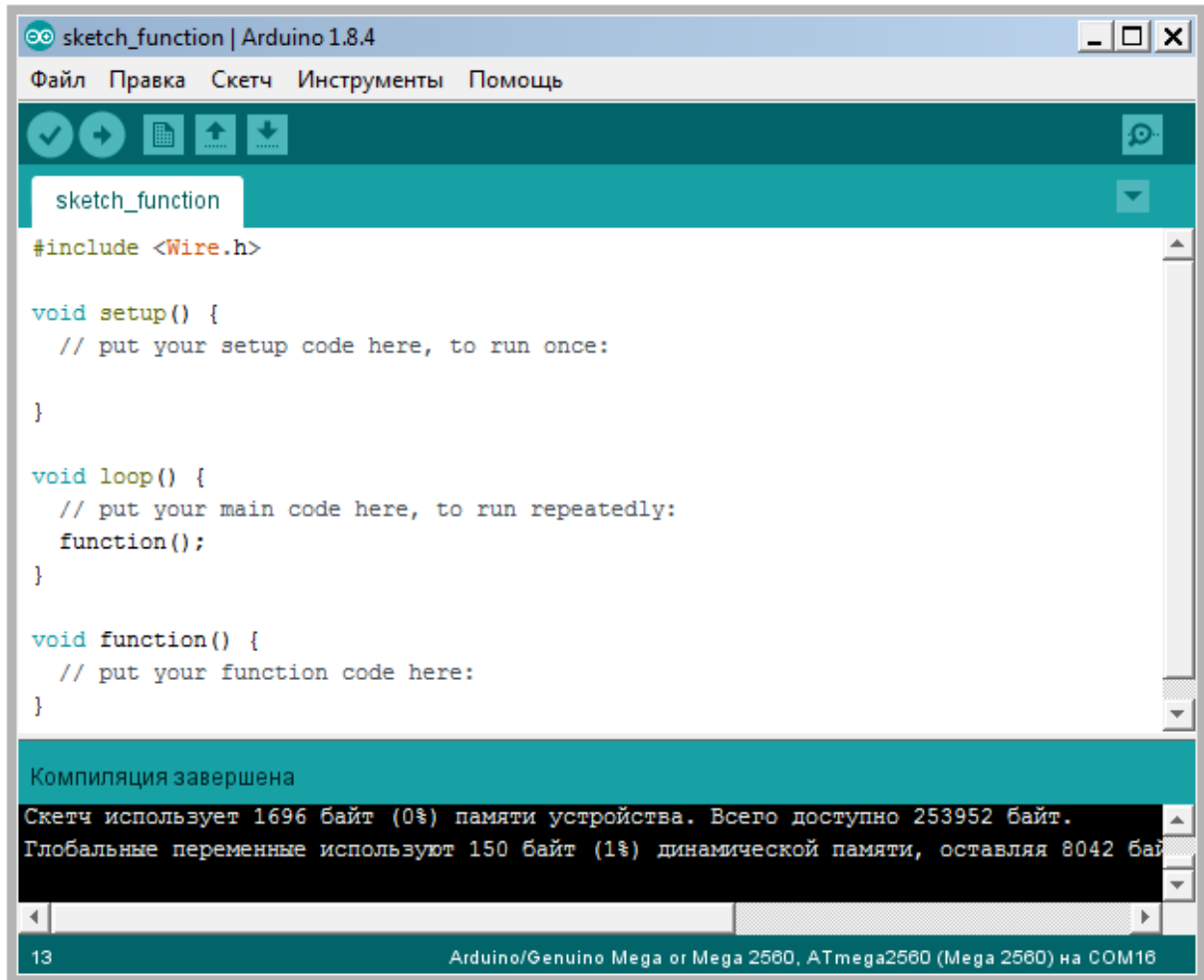


Рис. 1.2.1. Области вікна середовища розробки Arduino

Пункти меню редактора програм Arduino IDE (рис. 1.2.1) включає в себе наступні основні елементи: файл, правка, скетч, сервіс і довідка.

Докладніше кожне з меню складається з пунктів:

1. Меню «Файл» містить команди, що відповідають за створення нової програми, завантаження з диска вже існуючої програми, збереження змін, а також команди для завантаження програми на мікроконтролер:

- «Створити» – створити нову програму (в даному середовищі програми називаються скетчами);
- «Відкрити» – відкрити існуючу програму;
- «Папка зі скетчами» – відкрити програму із заданої папки;
- «Приклади» – відкрити приклад програми;

- «Закрити» – закрити поточне вікно;
- «Зберегти» – зберегти зміни;
- «Зберегти як» – зберегти програму в новому файлі;
- «Завантажити» - завантажити програму в Arduino;
- «Завантажити за допомогою програматора» – завантажити програму за допомогою програматора;
- «Налаштування друку» – настройка принтера;
- «Друк» – висновок на друк коду програми;
- «Налаштування» – настройки редактора;
- «Вихід» – вихід з Arduino IDE.

2. Меню «Правка» містить команди, пов'язані з редагування тексту програми, що включають в себе: копіювання, вставку, настройку відступів і пошук.

3. Меню «Скетч» містить команди для контролю за процесом компіляції програми:

- «Перевірити / Компілювати» – компілювати програму;
- «Показати папку скетчей » – відкриває системну папку з програмами;
- «Додати файл» – додати до проекту файл з даними або програмою;
- «Імпортувати бібліотеку» – підключити до програми бібліотеку зі списку встановлених.







4. Меню «Інструменти» включає в себе допоміжні функції для роботи з самим мікроконтролером:

- «Автоформатування» – автоматична розстановка відступів, переносів рядків і т.п.;
- «Архівувати скетч» – архівація папки з програмою і збереження архіву в вказане місце;
- «Монітор порту» – відкрити вікно для обміну даними з мікроконтролером;
- «Плата» – вибір поточної плати (в нашому випадку Arduino Uno);
- «Послідовний порт» – вибір порту, до якого підключений;
- «Програматор» – вибір програм (не використовується);
- «Записати завантажувач» – запис програми завантажувача в мікроконтролер (також нами не використовується).

5 Меню «Довідка» містить докладний опис всіх функцій самого редактора Arduino IDE, а також всілякі команди і прийоми роботи з платформою.

На панель інструментів винесені у вигляді кнопок найбільш часто використовувані функції редактора програм Arduino IDE. Призначення кнопок на панелі інструментів описано в таблиці 1.2.1.

Таблиця 1.2.1. Кнопки на панелі інструментів

Кнопки	Призначення
	Перевірити / Компілювати програму
	Завантажити програму в Arduino
	Створити нову програму
	Відкрити існуючу програму
	Зберегти програму
	Монітор послідовного порту

Безпосередньо текст програми створюється і редагується у вікні редактора коду. Це вікно є типовим текстовим редактором з підсвічуванням синтаксису програми. У нижній частині вікна Arduino IDE є область, що служить для виведення повідомлень про помилки, що виникають у процесі компіляції програми або під час завантаження програми в мікроконтролер.

1.3. Сполучення Arduino з LabVIEW

МПС на основі плати Arduino UNO підтримує зв'язок мікроконтролера ATmega328P через USB-порт комп'ютера з інтерфейсом у LabVIEW 2010 (рис. 1.3.1).

У розробника National Instruments є офіційні бібліотеки для підключення, налаштування і роботи Arduino UNO з LabVIEW 2010. Для підтримки необхідно встановити бібліотеку VI Package Manager 2010, яку можна завантажити через протокол передачі файлів FTP: NI LabVIEW Interface for Arduino Toolkit.

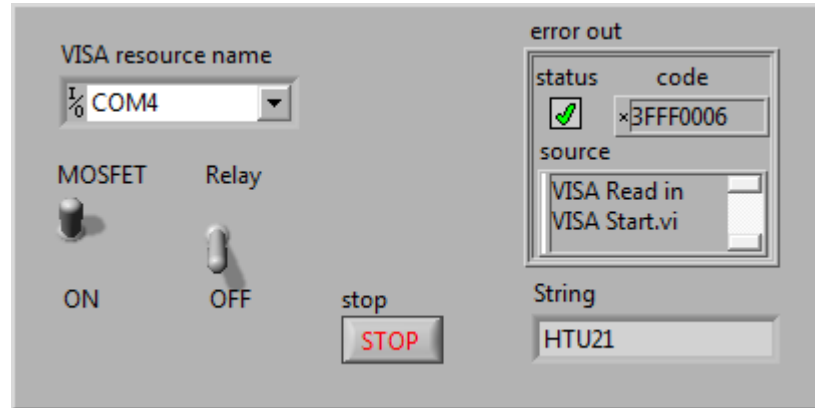


Рис. 1.3.1. Інтерфейс у LabVIEW 2010 для підтримки плати Arduino UNO

Мета цього інтерфейсу в LabVIEW полягає в підтримці послідовного терміналу зв'язку, що дозволяє користувачеві відправити команду читання / запису в контролер Arduino для обміну даними через USB-порт. Ця ситуація зустрічається, коли клієнт повинен мати можливість керувати приладом, не вдаючись до перепрограмування Arduino. Крім того, застосування інтерфейсу буде перешкоджати користувачеві вводити невірні дані, які можуть привести до виникнення небезпечних ситуацій в реальному проектуванні.

Створення каналу зв'язку в LabVIEW для контролера Arduino UNO починається з використання візуального компоненту VI «VISA Configure Serial Port», що призначений для налаштування послідовного порту (рис. 1.3.2).

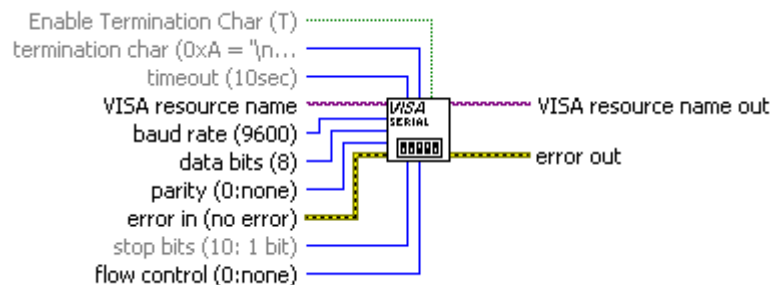


Рис. 1.3.2. Візуальний компонент VI «VISA Configure Serial Port»

Цей візуальний компонент (ВК) отримує ім'я VISA ресурсу, який COM-порт буде займати плата Arduino UNO (наприклад, COM6), що налаштовується у меню компоненту Data Communication>Protocols>Serial Palette.

Для створення ВК необхідно клацнути правою кнопкою миші на ім'я вузла ресурсів і перейти до меню Create> Control. Блок управління повинен мати такий вигляд (рис. 1.3.3):

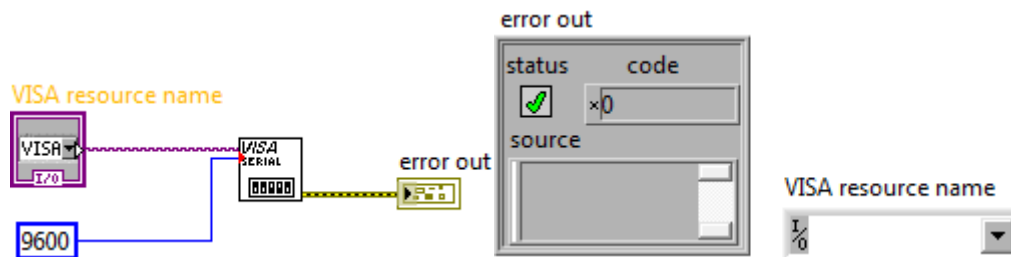


Рис. 1.3.3. Налаштування ВК «VISA Configure Serial Port»

Перейменувавши цей блок в «Serial Communication Port» (послідовний порт зв'язку) можна встановити швидкість передачі даних 9600, клацнувши правою кнопкою миші на вході «baud rate» (швидкість передачі даних), а потім вибравши меню Create> Constant встановити його значення в 9600.

Тепер можна відкрити повідомлення, що надходить через послідовний порт зв'язку, вибравши візуальний компонент VI «VISA Open» (рис. 1.3.4):

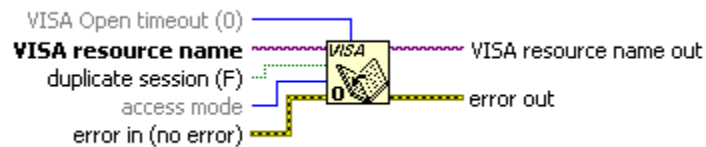


Рис. 1.3.4. Візуальний компонент VI «VISA Open»

Цей ВК можна знайти, натиснувши Ctrl + Space і набравши «VISA Open», або через меню Instrument I/O>VISA> VISA Advanced, як показано нижче (рис. 1.3.5):

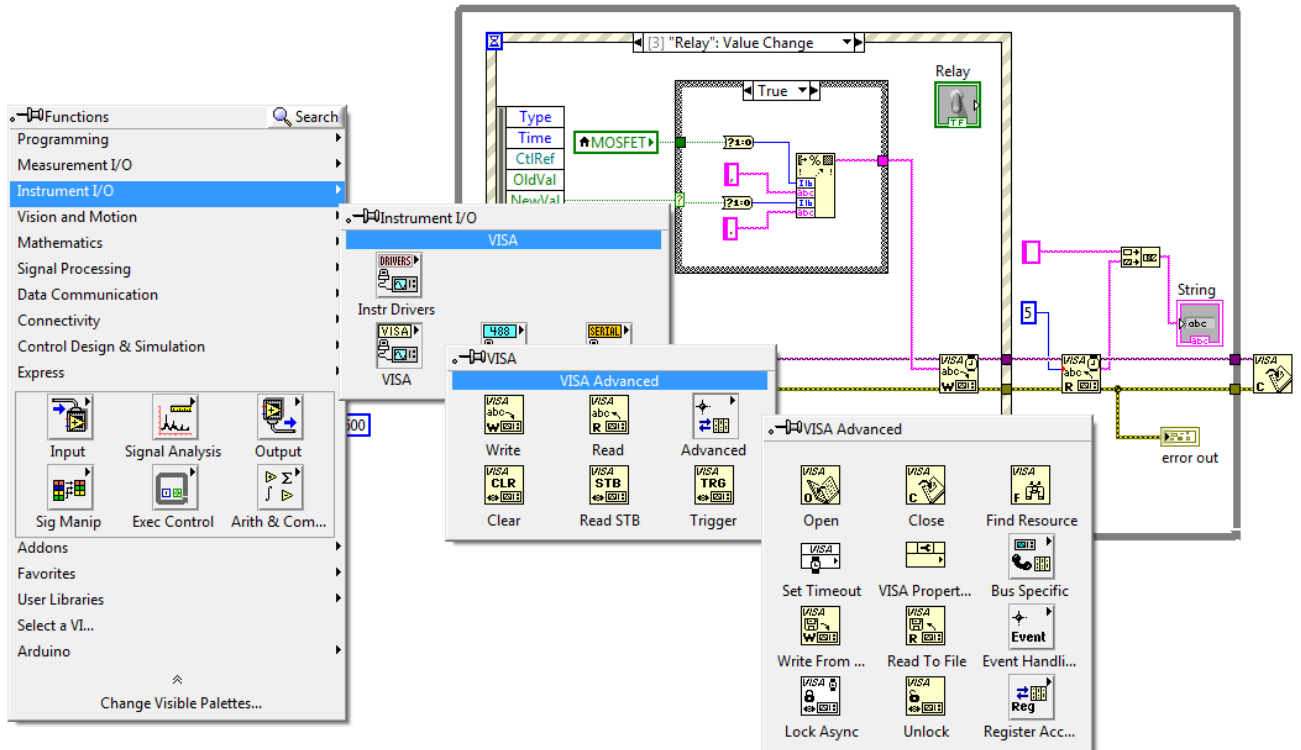


Рис. 1.3.5. Налаштування ВК «VISA Open»

Потім можна створити структуру подій «Event Structure» всередині нового циклу «While Loop» і додати візуальний компонент «VISA Close» VI (рис. 1.3.6), який буде закривати послідовний комунікаційний порт, коли будемо натискати кнопку зупинки «Stop».

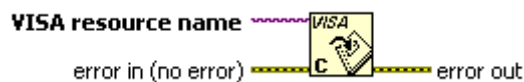


Рис. 1.3.6. Візуальний компонент VI «VISA Close»

На передній панелі, можна додати два тумblers, один з них буде називатися «MOSFET», а інший «Relay», важільні перемикачі можна знайти в палітрі кнопок «Buttons».

Потім у структуру подій «Event Structure» можна додати 3 події:

1. Додайте одну подію для перемикання тумblersа MOSFET: Value change;
2. Додайте одну подію для перемикання тумblersа Relay: Value change;
3. Додайте одну подію для перемикання тумblersа STOP: Value change.

Код всередині блоків подій «Event Structure» буде виконуватися кожного разу, коли її відповідний тумблер або кнопка приводиться в дію:

1. Випадок події «MOSFET: Value change» (рис. 1.3.7)

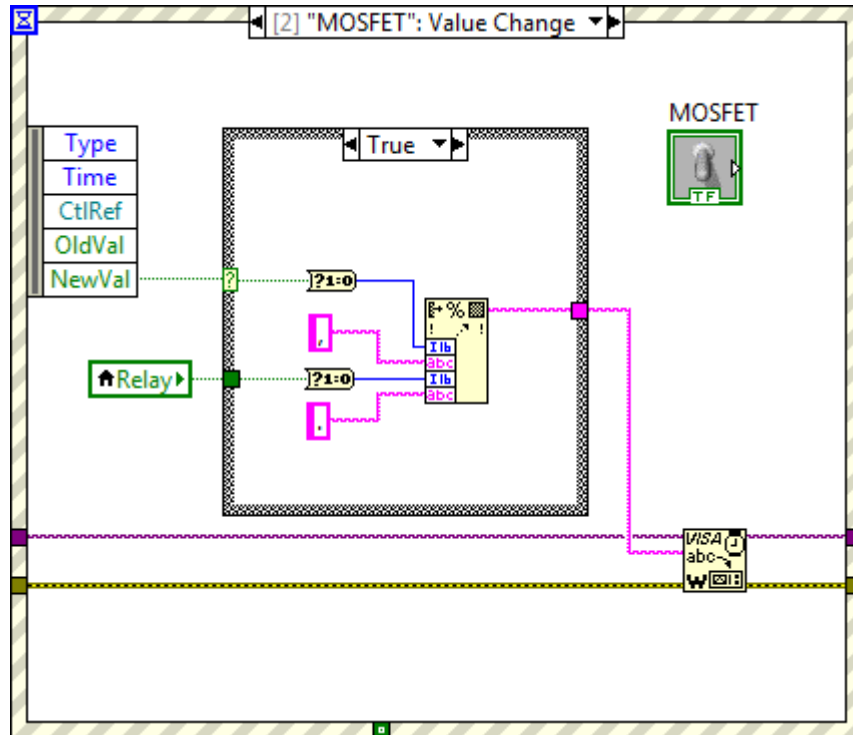


Рис. 1.3.7. Випадок події «MOSFET: Value change»

В випадку події «MOSFET: Value change» передбачається, що до Arduino буде відправлено повідомлення – включити або вимкнути модуль MOSFET.

Значення перемикача MOSFET при перемиканні тумблера надає доступ до терміналу «NewVal» у структурі подій «Event Structure». З іншого боку, значення перемикача «Relay» надає доступ до локальної змінної «Local Variable» (рис. 1.3.8).



Рис. 1.3.8. Локальна змінна «Local Variable»

Щоб створити локальну змінну «Local Variable», необхідно натиснути Ctrl + Space і виконати пошук локальної змінної (рис. 1.3.9).

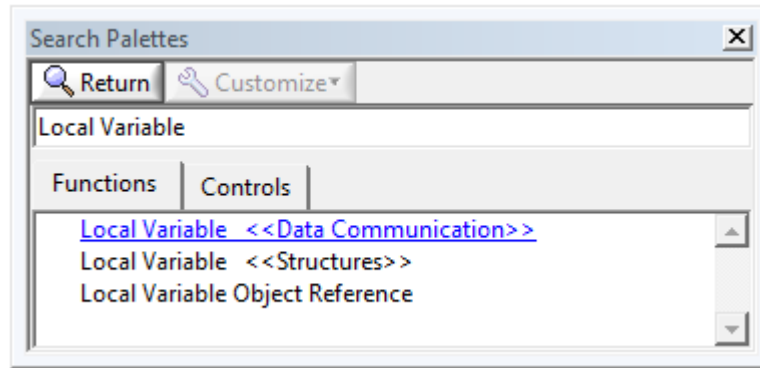



Рис. 1.3.9. Пошук локальної змінної «Local Variable»

Повинен з'явитися блок . Клацніть правою кнопкою миші на цьому блоці та виберіть – змінити наступним чином «Change to Read» і далі побачите, що стрілка рухається вправо на панелі годин «Local Variable». Потім клацніть лівою кнопкою миші на зміненому блоці і виберіть «MOSFET» або «Relay» і локальна змінна буде встановлена на потрібному значенні тумблера (рис. 1.3.10).

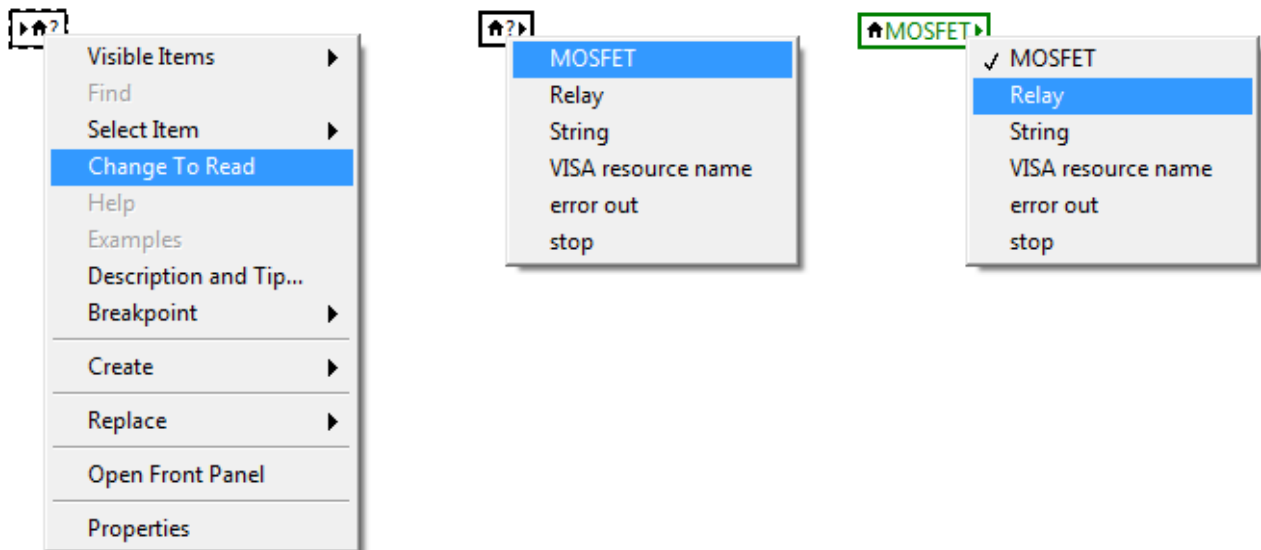


Рис. 1.3.10. Меню «Change to Read»

Так як кнопки і перемикачі в LabVIEW мають логічний тип даних (наприклад, значення TRUE / FALSE), можна використати функцію «Boolean To (0,1)» VI для перетворення логічних значень в цілочисельні значення.

Причому, колір проводів у з'єднанні компонентів змінюється в залежності від типу даних, які вони містять, дроти підключені до булевого введення/виводу «Boolean input/output» завжди буде зеленим. Ці значення потім необхідно форматувати в строку «Format Into String» (рис. 1.3.11).

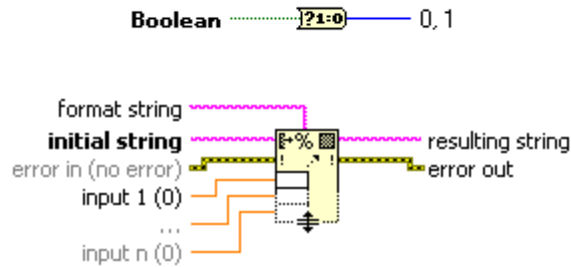


Рис. 1.3.11. Компоненти «Boolean Functions» і «Format Into String»

Отриману строку необхідно далі відправити на мікроконтролер через компонент «VISA Write» VIs (рис. 1.3.12).



Рис. 1.3.12. Візуальний компонент VI «VISA Write»

Код всередині блоків подій «Event Structure» для випадку події «MOSFET: Value change» має виглядати наступним чином (рис. 1.3.13):

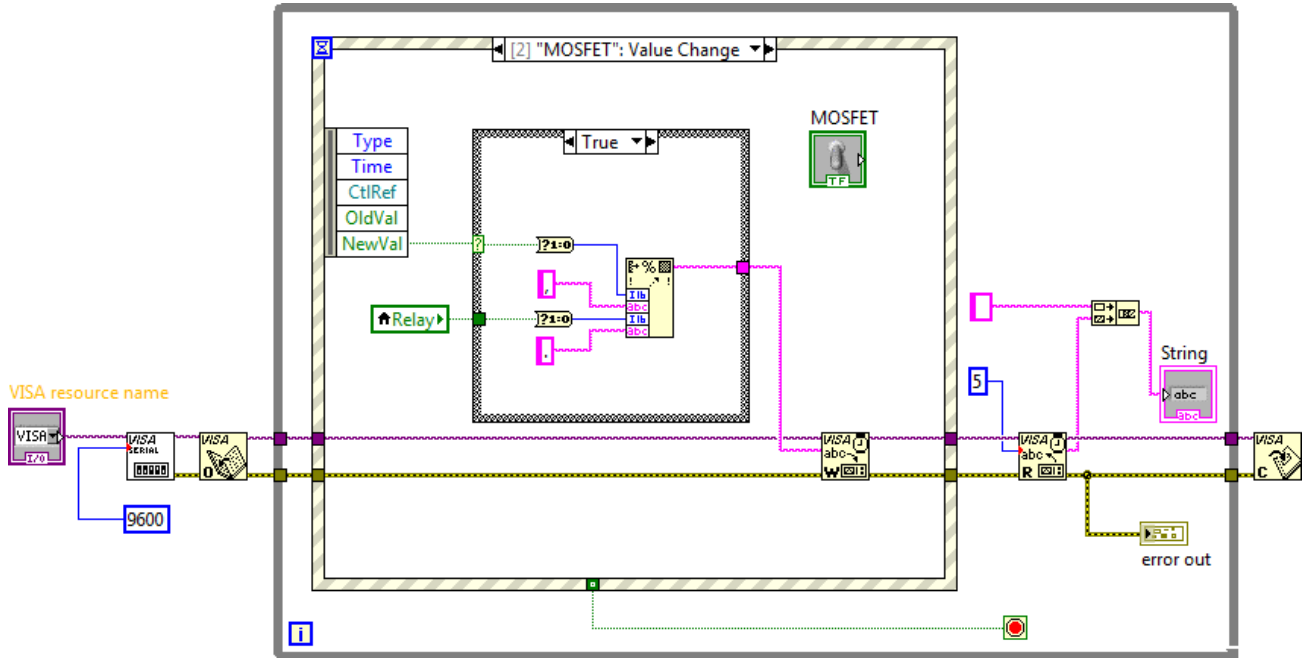


Рис. 1.3.13. Блок-схема для випадку події «MOSFET: Value change»

Для випадку події «Relay: Value change» код має наступний вигляд (рис. 1.3.14):

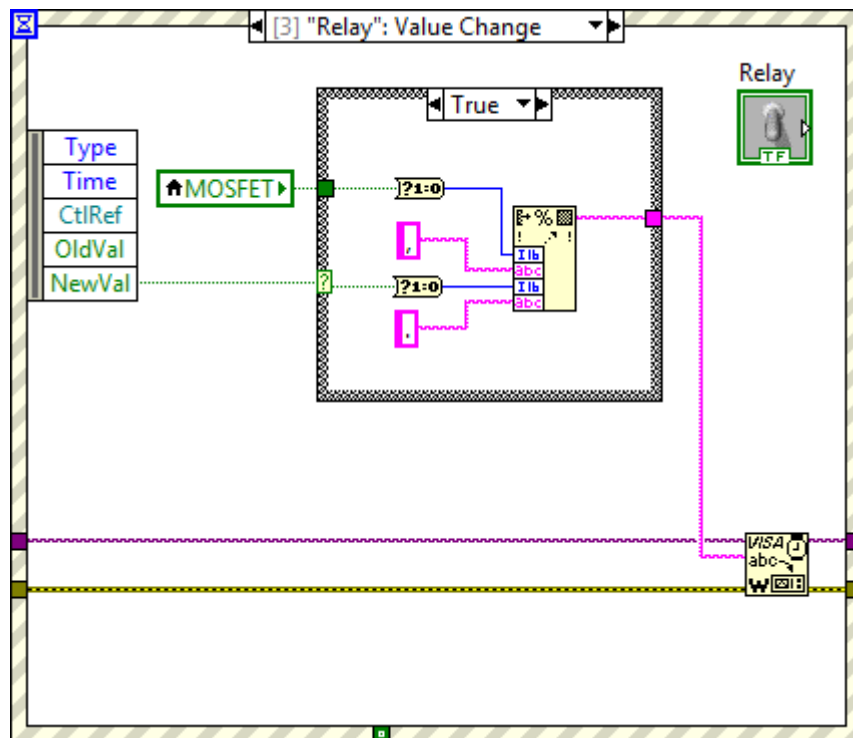


Рис. 1.3.14. Випадок події «Relay: Value change»

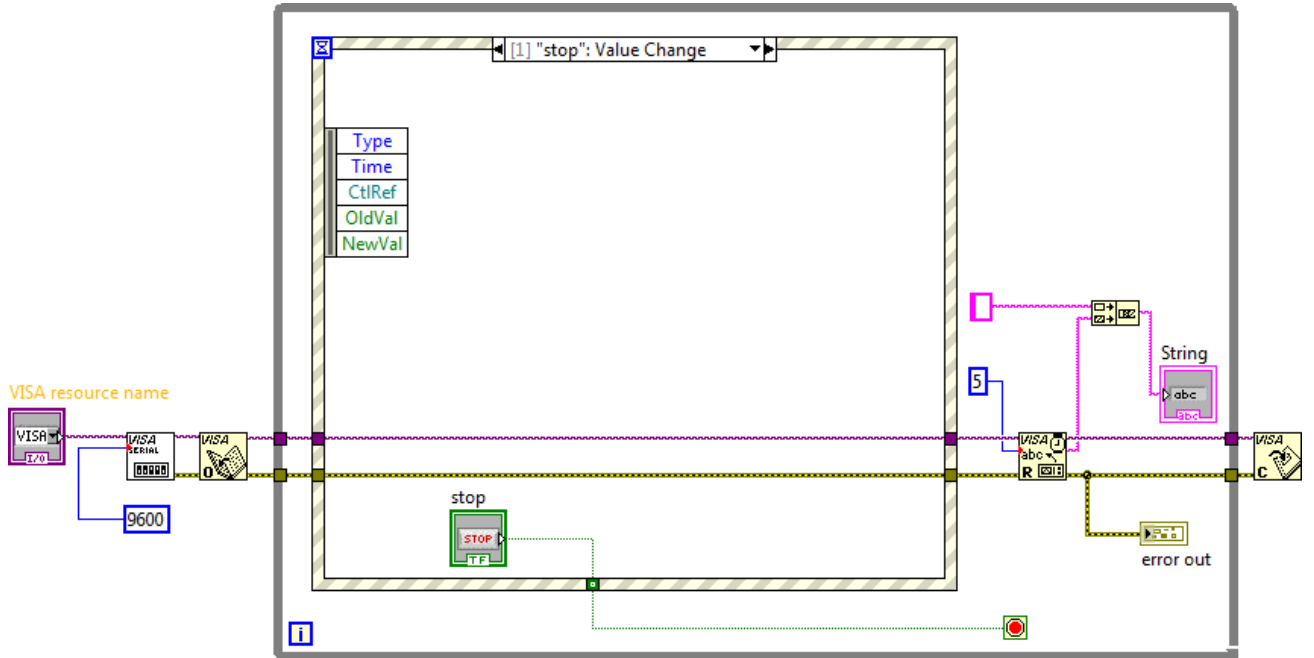


Рис. 1.3.16. Блок-схема для випадку події «STOP: Value change»

Випадок події «Timeout Event» можна залишити з не написаним кодом (рис. 1.3.17). Ця структура може реалізувати затримку в часі при очікуванні повідомлення про подію. Значення терміналу Timeout у верхньому полі структури події «Timeout Event» вказує кількість мілісекунд, яка триває при очікуванні на подію.

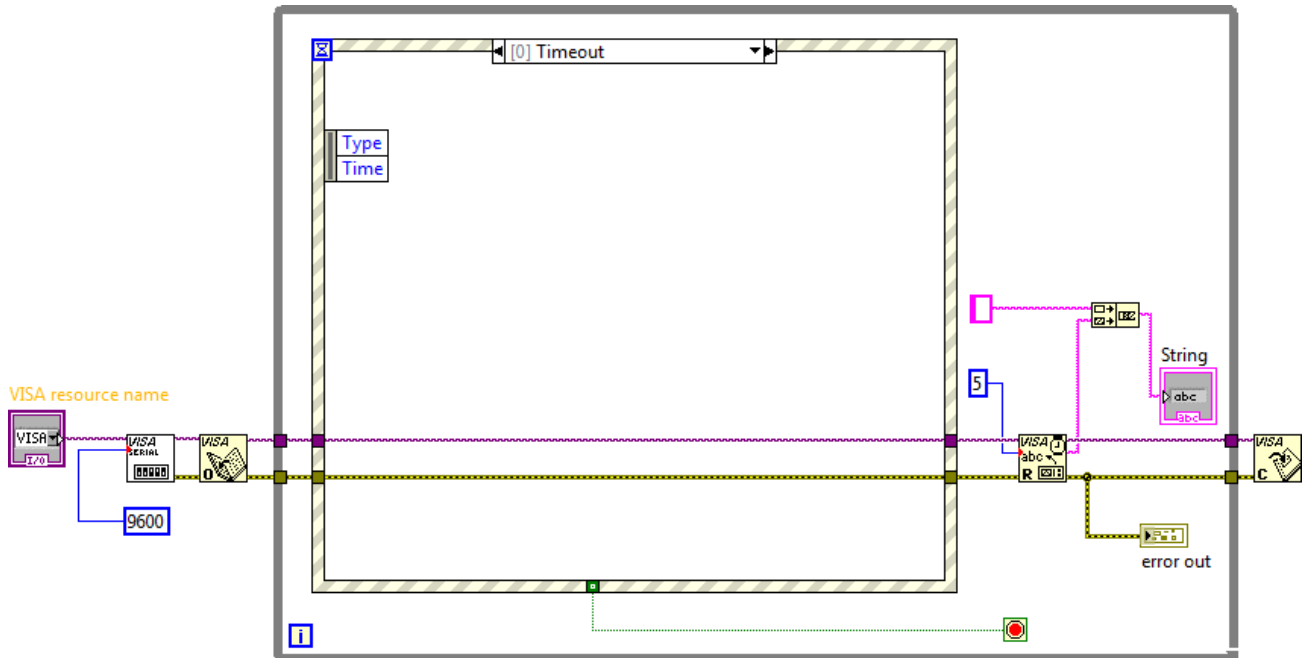


Рис. 1.3.17 Блок-схема для випадку події «Timeout Event»

Типовим значенням Timeout є -1, що вказує на те, що час очікування не встановлено.

При тестуванні програми у LabVIEW якщо всі виводи компонентів підключені правильно, програма повинна запуститися. Необхідно вибрати COM-порт підключення Arduino і включити / вимкнути модулі, натиснувши на тумблери «MOSFET» і «Relay». Зупинити програму, можна натиснувши кнопку STOP.

1.4. Структура і особливості програми

Програмування в середовищі Arduino здійснюється на мові Cі з мінімальним набором елементів C++. Надалі будемо, для стислості, називати мову програмування в середовищі Arduino мовою Cі. Ця мова дуже схожа на мову Cі, але має деякі відмінності. Програма для Arduino повинна обов'язково бути написана в одному файлі, який називають скетч або програма «скетч для Arduino». В структурі скетчу є відмінності від структури звичайної програми на Cі, що пов'язано з відмінністю між роботою програми на звичайному комп'ютері та в мікропроцесорних системах.

При виконанні програми на звичайному комп'ютері, яка написана на мові Cі, за її виконання відповідає функція main(). При запуску програми запускається ця функція, з якої можна викликати інші функції програми. Коли функція main() повертає управління за допомогою return, програма завершує роботу.

Але для програми, що управляє мікроконтролером, така поведінка не має сенсу. Поки на МПС подано живлення, мікроконтролер повинен управляти пристроєм, а значить, повинен постійно виконувати свою програму. Ця програма ніколи не завершується сама, вона перестає працювати, тільки якщо вимкнути мікроконтролер. Тому структура програми для Arduino відрізняється від структури звичайної програми на мові Cі. У програмі для Arduino немає функції main(), Замість неї в мікроконтролері є дві функції: setup() і loop().

Функція setup() викликається один раз у самому початку виконання програми відразу після того, як мікроконтролер Arduino включається. Ця функція налаштовує мікроконтролер, зокрема задає, які його виводи будуть використовуватися як входи, а які – як виходи і так далі. У цій функції можна якимось іншим чином приготувати пристрій до

роботи, наприклад, включити додаткові можливості, або навпаки, переконатися, що нічого зайвого поки не включено.

Функція `loop()` виконується весь час, поки працює мікроконтролер Arduino. Якщо вона завершається, то середовище Arduino запускає її заново. Саме у цій функції потрібно реалізувати алгоритм роботи пристрою. Зазвичай у функції `loop()` опитують підключені до мікроконтролеру сенсори, або самі входи мікроконтролера, аналізують отриману інформацію і видають команди виконавчим пристроям.

Можна припустити, що в бібліотеці Arduino вже є функція `main()`, яка може бути влаштована наступним чином:

```
void main () {
    setup ();
    while (true)
        loop ();
}
```

Така функція `main()` дійсно є в бібліотеці Arduino, але вона влаштована складніше. У середовищі Arduino як і в звичайних програмах, можна створювати будь-які інші функції користувача і викликати їх з функцій `setup()` і `loop()`. Якщо розробляти просту програму, то можна написати весь код просто всередині функцій `setup()` і `loop()`. Проте, якщо робити щось складне, то варто розбити код програми на окремі функції.

В програмі Arduino можна викликати стандартні функції, які є в бібліотеці. Для цього не потрібно додавати директив `#include`. У звичайній мові Сі це потрібно робити навіть для стандартних функцій. Одна з функцій, яка не потребує додавання директив `#include` є функція `delay(time)`, яка виконує затримку на зазначений час.

Приклад функції `delay(time)`, що виконує затримку на зазначений час, що задається в мілісекундах, має вигляд:

```
delay (1000);
```

В програмі Arduino є ще кілька стандартних функцій, які можна відразу писати в програмі. Ці функції керують базовими можливостями мікроконтролера.

Великою перевагою середовища Arduino є велика кількість плат розширення різних виробників, які можна підключати до Arduino і використовувати разом з МПС. Для того щоб такі плати було зручно використовувати, для них написані бібліотеки, що містять зручні

функції для роботи з цими пристроями. Такі бібліотеки представляють окремі компоненти, які можна підключити до плати Arduino.

Щоб використовувати бібліотеку, потрібно на початку програми помістити директиву `#include`:

```
#include <ім'я_бібліотеки.h>
```

Наприклад, бібліотека для роботи з дисплеєм називається `LiquidCrystall`, і щоб її використовувати, треба спочатку написати:

```
#include <LiquidCrystal.h>
```

У мові Сі в середовищі Arduino для роботи з бібліотеками використовують елементи мови С++. Фактично, самі бібліотеки пишуть на С++, але в скетчі можна використовувати тільки дуже прості можливості цієї мови. Мова С++ дозволяє створювати класи. Класи містять призначені для користувача типи даних, що описують набір полів і операції, які можна з ними робити. З допомогою цього механізму в С++ можна зробити, наприклад, клас «`NewLiquidCrystall`», що наслідуює властивості класу «`LiquidCrystall`» і об'єкт, який належить цьому класу. Щоб зробити такий об'єкт, треба його оголосити, як оголошують звичайні змінні. Наприклад, для роботи з дисплеєм треба зробити змінну типу `LiquidCrystall`:

```
LiquidCrystal lcd (4, 5, 6, 7, 8, 9);
```

У цьому прикладі `LiquidCrystall` – це назва типу, яка описана в документації у бібліотеці «`LiquidCrystall`». Змінна `lcd` – це назва об'єкту, який можна використовувати як будь-яку змінну. Цифри – це параметри, які передаються об'єкту при створенні. Про те, які параметри треба вказати, також описано в документації. Для дисплея це, номери виводів мікроконтролера, до яких він підключений.

Для роботи з об'єктами використовують спеціальні функції, які доступні тільки для цієї змінної та представляють методи класу. Викликають методи класу наступним способом:

```
ім'я_змінної . ім'я_функції (параметри);
```

Наприклад, текст на дисплей виводить функція `print()`, яку можна викликати наступним чином:

```
lcd.print ( "Hello!");
```

У прикладі спочатку вказується ім'я змінної, тому функція `print()` викликається саме для цієї змінної. Таким чином, можна підключити кілька дисплеїв і зробити для кожного з них свою змінну.

Розглянемо інші особливості синтаксису мови Сі для Arduino UNO, знання яких необхідно для написання програм.

Кожен вираз на мові Сі закінчується символом крапки з комою. Наприклад вираз:

```
a = b + c;
```

Тіло функцій і складових операторів (if, else, for, while) виділяються фігурними дужками, наприклад:

```
if (a > 0) {
    b = a - c;
}
```

Строкові змінні виділяються подвійними лапками:

```
lcd.print ("some text");
```

Символи виділяються одинарними лапками:

```
symbol = 'a';
```

Підключення бібліотек здійснюється за допомогою конструкції:

```
#include <math.h>
```

Коментарі в програмі починаються двома прямими виду слеш:

```
// це коментар
```

Оголошення змінної в мові Сі здійснюється за допомогою конструкції виду:

```
тип_змінної ім'я_змінної;
```

Наприклад, оголошення змінної має вигляд:

```
int x, y; // оголошені змінні x і y, мають цілий тип
```

У програмах для Arduino можна використовувати такі типи для числових даних:

byte – ціле число від 0 до 255;

int – ціле число від -32768 до +32767;

word – ціле число від 0 до 65535;

long – ціле число від -2147483648 до 2147483647;

float і double – числа з плаваючою точкою, відповідають типу float для звичайних комп'ютерів, тип double еквівалентний float.

Масиви в мові Сі оголошуються конструкцією виду:

```
тип_елемента ім'я_масиву [розмір];
```

Наприклад, оголошення масиву має вигляд:

```
int values [10]; // масив з десяти цілих чисел
```

У мові Сі передбачений тип для зберігання символів, це тип – char. У векторі та масиві, що мають тип char останній символ має код 0, який позначає кінець масиву.

Наприклад, оголошення змінної типу char має вигляд:

```
char data_str [10]; // вектор довжиною не більше дев'яти символів
```

Тип boolean використовується для представлення логічних значень true() і false(). Крім того, будь-яке ціле значення може використовуватися як логічне значення, при цьому 0 означає – false(), а будь-яке не нульове значення – true().

Тип const перед оголошенням змінної свідчить про те, що її значення не може бути змінено у процесі виконання програми.

Останній тип даних – це тип void, який використовується при оголошенні функцій, що не мають аргументів або які не повертають результат.

У мові Сі використовується стандартний набір математичних операторів, що включає +, -, * і /, для обчислення залишку від ділення використовується %. Крім того, є їх спеціальні комбінації з оператором присвоювання, що дозволяють скоротити запис:

```
x += 4; // означає x = x + 4
```

Такі форми є для всіх операторів. Щоб збільшити або зменшити задане число на одиницю є інкремент і декремент:

```
x ++; // означає x = x + 1
```

```
y--; // означає y = y - 1
```

Набір операторів порівняння є стандартним і включає оператори <, >, <=, > =. Порівняння записується за допомогою двох знаків рівності (==), а нерівність записується !=. Логічне заперечення записується за допомогою знаку оклику !. Логічні вирази можна записувати за допомогою операторів, що представлені в таблиці 1.4.1

Таблиця 1.4.1. Логічні операції на мові С++

Операція	Визначення	Приклад	Опис складної умови
і	&&	a == 3 && b > 4	істинно, якщо істинні обидві прості умови
або		a == 3 b > 4	істинно, якщо істинна, хоча б одне з простих умов
ні	!	!(a == 3)	істинно, якщо а не дорівнює 3

Умовний оператор має наступний синтаксис:

```
if (умова)
    код, що виконується, якщо умова істинна;
```

Якщо в разі істинності умови необхідно виконати декілька операторів, то вони об'єднуються у один блок за допомогою фігурних дужок:

```
if (умова) {
    код, що виконується, якщо умова істинна;
}
```

Повна форма умовного оператора включає блок else, Код в якому виконується інша умова, якщо істина умова помилкова має вигляд:

```
if (умова) {
    код, що виконується, якщо умова істинна;
}
else {
    код, що виконується, якщо умова помилкова;
}
```

Оператор switch дозволяє здійснити вибір однієї з гілок коду в залежності від значення числової змінної, наприклад:

```
switch (mode) {
    default:
    case 0:
        код для режиму 0;
    break;
    case 1:
        код для режиму 1;
    break;
    case 2:
        код для режиму 2;
    break;
}
```


При цьому значення змінної `mode` визначає, в яку точку коду перейде виконання програми після виконання оператора `switch`. Далі будуть виконуватися всі дії, поки не зустрінеться `break`, який перерве подальше виконання коду всередині `switch`. Якщо значення виразу не відповідає жодній з міток, то виконується код, починаючи з мітки `default`. В даному прикладі через відсутність `break` після `default` управління потрапить в «код для режиму 0».

Цикл `while` дозволяє виконувати програмний код до тих пір, поки умова вірна. Він має синтаксис:

```
while (умова)
    код, що виконується в циклі;
```

Більш складним є цикл `for`:

```
for (ініціалізація; умова; крок)
    код, що виконується в циклі;
```

У цьому циклі вираз, що вказаний в блоці «ініціалізація», виконується один раз перед початком циклу. Зазвичай в ньому задають початкові значення для змінних циклу. Умова перевіряється кожен раз на самому початку чергової ітерації циклу і якщо вона стане помилково, то цикл зупиняється.

Вираз «крок» виконується після виконання тіла циклу і зазвичай збільшує (або зменшує) змінні циклу. У блоках «ініціалізація» і «крок» можна задати кілька змінних, якщо написати відповідні вирази через кому.

Приклад циклу, що обчислює суму арифметичної прогресії:

```
int i, sum;
for (i = 0, sum = 0; i <= 10; i++)
    sum += i;
```

Код програми на мові Сі повинен бути обов'язково розбитий на функції. Для оголошення функції використовується наступний синтаксис:

```
тип_функції ім'я_функції (параметри)
{
    код, що виконується в рамках функції;
    return результат_функції;
}
```

Тип функції описує тип значення, яке буде повертати функція. Наприклад, стандартна функція `cos`, що обчислює косинус, повертає значення типу `float`.

Ім'я функції може мати будь-яку форму, що починається з букви, містить тільки літери, цифри і символи підкреслення. Параметри – це перелік змінних, які приймають значення, передані в функцію. Змінні вказуються через кому, спочатку пишуть тип параметра, а потім – його назву.

Щоб повернути значення з функції використовується оператор `return`. Він негайно перериває виконання функції, а результатом виконання функції стає значення зазначеного після `return` вираження.

Якщо функція нічого не повертає, то тип_функції вказується як `void`. Якщо функція не передається ніяких аргументів, то можна нічого не писати всередині круглих дужок або можна написати там слово `void`.

Наприклад функція, що підсумовує два числа типу `int` має вигляд:

```
int sum (int a, int b) {
    int result;
    result = a + b;
    return result;
}
```

Для виклику функції необхідно написати її ім'я і в круглих дужках вказати через кому значення аргументів, наприклад:

```
r = sum (3, 10);
```

У програмах на мові Сі можна оголосити змінні за межами будь-яких функцій. Такі змінні є глобальними і їх можна використовувати в будь-яких функціях, що розташовані після оголошення змінної. Наприклад, якщо змінна `led` використовується у функціях `setup()` і `loop()` вона описується як глобальна:

```
const int led = 2;
void setup () {
    pinMode (led, OUTPUT);
}
void loop () {
    digitalWrite (led, HIGH);
```

```

    delay (1000);
    digitalWrite (led, LOW);
    delay (1000);
}

```

Одна з найпростіших дій, яку може виконати мікроконтролер, це перемикання стану контактів цифрових входів-виходів на платі Arduino UNO. Будь-який з цифрових виходів мікроконтролера можна програмним способом переключити між високим рівнем (+5 В) і низьким рівнем (0 В).

За допомогою такого перемикання відбувається управління простими пристроями:

1. Перемикання стану контактів цифрових входів-виходів на платі Arduino UNO здійснюється функцією `digitalWrite()`, виклик якої має вигляд:

```
digitalWrite (номер_контакта, рівень_сигналу);
```

2. Перемикання стану контактів аналогових входів-виходів на платі Arduino UNO здійснюється функцією `analogWrite()`, виклик якої має вигляд:

```
analogWrite (номер_контакта, рівень_сигналу);
```

3. Читання стану контактів цифрових входів-виходів на платі Arduino UNO здійснюється функцією `digitalRead()`, виклик якої має вигляд:

```
ім'я_змінної = digitalRead (номер_контакта);
```

4. Читання стану контактів аналогових входів-виходів на платі Arduino UNO здійснюється функцією `analogRead()`, виклик якої має вигляд:

```
ім'я_змінної = analogRead (номер_контакта);
```

Параметр рівня сигналу може приймати два значення: HIGH (високий, +5 В) або LOW (низький, 0 В). Параметр номер_контакта – це число, що відповідає номеру контакту на платі Arduino. Параметр ім'я_змінної – це змінна, до якої записується числовий еквівалент сигналу, що встановлений для заданого номеру контакту на платі Arduino UNO.

Зверніть увагу, що одні й ті ж виводи мікроконтролера можуть виступати як в ролі цифрових виходів, так і в ролі цифрових входів. Тому перед їх використанням необхідно вказати, в якій ролі буде використовуватися контакт: вхід або вихід. Це робиться за допомогою функції `pinMode()`:

```
pinMode (номер_контакта, режим_контакту);
```

Аргумент цієї функції режим_контакту може приймати значення: OUTPUT (вихід) і INPUT (вхід). Таким чином, щоб на цифровому виході (2) встановити спочатку високий рівень сигналу (HIGH), а потім з затримкою (Delay) 1000 мс встановити низький рівень сигналу (LOW), достатньо виконати наступну програму:

```
const int pin = 2;
void setup () {
    pinMode (pin, OUTPUT);
    digitalWrite (pin, HIGH);
}
void loop () {
    digitalWrite (led, HIGH);
    delay (1000);
    digitalWrite (led, LOW);
    delay (1000);
}
```

Згадаймо, що контролер Arduino працює на частоті 16MHz, тому він може включати і вимикати світлодіод, що підключений до виводу (2) тисячі разів у секунду. У сучасній мікроелектроніці застосовуються мініатюрні світлодіоди для поверхневого монтажу. Такі індикатори, наприклад, є на Arduino Uno для інформування користувача про стан системи.

Важливо відзначити, що робоча напруга живлення світлодіода варіюється від 1.85 В до 2.5 В при рекомендованій силі струму не більше 20mA. Щоб сила струму не перевищила це значення, при підключенні світлодіода до виходу мікроконтролера, який видає напругу 5 В, у ланцюг слід додати послідовний обмежувачий резистор з опором від 200 до 500 Ом.

2. Вимірювальні прилади з мікропроцесорним управлінням

Мікропроцесорна система на основі плати Arduino UNO дозволяє реалізувати вимірювальні прилади з мікропроцесорним управлінням, які можуть включати до свого складу електроні двигуни, датчики та сенсори:

- Цифровий сенсорний датчик;
- Універсальний звуковий датчик;

- Управління кроковим двигуном 28BYJ-48;
- Датчик швидкості з цифровим і аналоговим виходами;
- Датчик вологості і температури HTU21;
- Цифровий модуль з термістором;
- Датчик вібрації SW-420;
- Цифровий датчик удару;
- Цифровий датчик Холу KY-03;
- Датчик нахилу на LM393;
- Датчик алкоголю MQ-3;
- Цифровий датчик рівня рідини;
- Цифровий датчик вогню;
- Аналогово-цифровий датчик освітленості;
- Ультразвуковий датчик відстані HC-SR04;
- Цифровий датчик перешкоди;
- Цифровий модуль з термістором.

Вказані вище датчики та сенсори використані в лабораторних макетах приладів з мікропроцесорним управлінням на основі плати Arduino UNO, що використовуються у навчальних цілях при розробці вимірювальної апаратури з використанням програмного середовища Arduino та інформаційного забезпечення засобів National Instruments LabVIEW.

2.1. Цифровий сенсорний датчик

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління сенсорним датчиком торкання TTP223B у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Сенсорний модуль побудований на мікросхемі ємнісного датчика торкання TTP223B (рис. 2.1.1). У нормальному стані на виході модуля SIG низький логічний рівень. При торканні сенсорного майданчика на виході встановлюється високий логічний рівень.

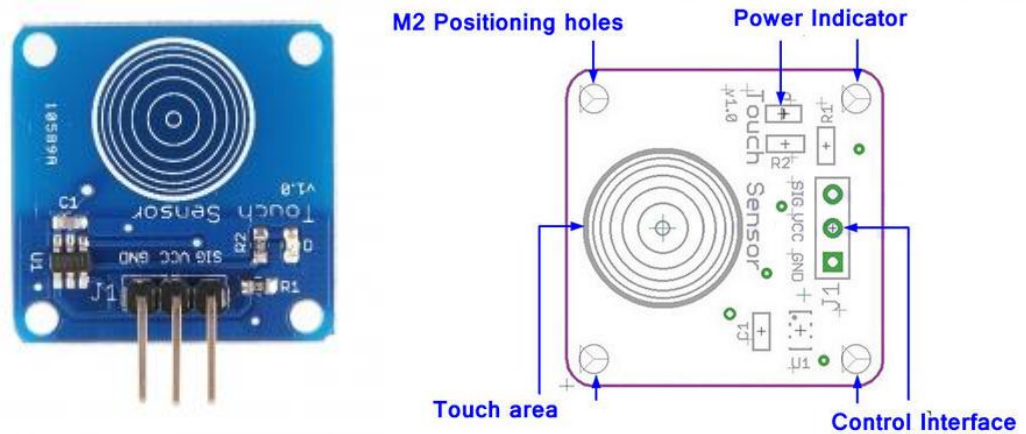


Рис. 2.1.1. Мікросхема ємнісного датчика торкання ТТР223В

Дуже малий струм споживання спільно з режимом зниженого споживання дозволяє використовувати модуль в системах з автономним живленням. Чутливість датчика дозволяє монтувати його за поверхнями невеликої товщини, що дуже зручно для використання його в якості прихованих кнопок. Сенсор спрацьовує на відстані до 4 мм.

Характеристики сенсору ТТР223В:

Параметр	Мін.	Стандартне	Макс.	Одиниця виміру
Робоча напруга живлення VCC	2.0	3	5.5	В
Високий рівень вихідного сигналу VOH	-	0.8VCC	-	В
Низький рівень вихідного сигналу VOL	-	-	0.3VCC	В
Вихідний струм (VCC = 3В, VOL = 0.6В)	-	8	-	мА
Струм на виході (VCC = 3В, VOH = 2.4В)	-	4	-	мА
Час відгуку (Low power mode)	-	-	220	мс
Час відгуку (Touch mode)	-	-	60	мс
Розміри	24x24x7.2			мм

Призначення виводів модуля:

- діапазон напруги живлення: від 2 до 5.5В;
- інтерфейс: GND - загальний, VCC - живлення, SIG (DI) - вихід;
- індикатор живлення: зелений світлодіод, що світиться при подачі напруги живлення;
- область сенсора: позначена область на платі модуля;
- кріпильні отвори: 4 M2 отвори діаметром 2.2мм по кутах модуля.

Підключення мікросхеми ємнісного датчика торкання ТТР223В до МПС на основі Arduino UNO представлено на рис. 2.1.2. Якщо загориться зелений світлодіод на модулі сенсорної кнопки, значить живлення подано коректно.

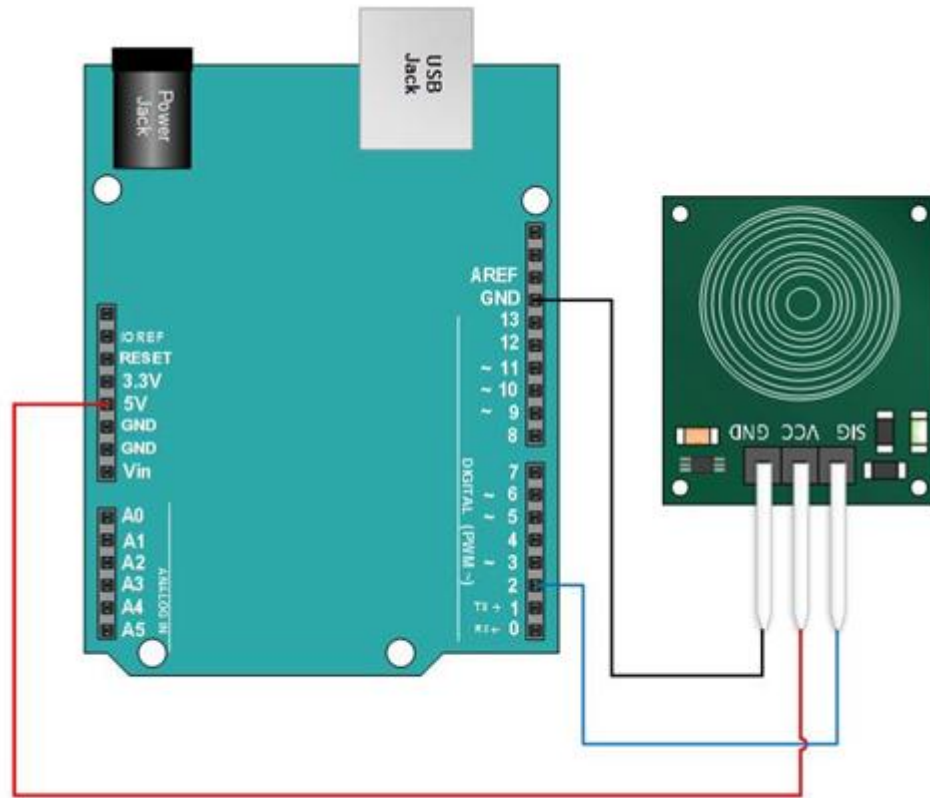


Рис. 2.1.2. Підключення мікросхеми ємнісного датчика торкання ТТР223В до МПС

Для підключення до мікроконтролеру Arduino UNO сенсорний модуль має три контакти: для підключення живлення (VCC), землі (GND) і вихідного сигналу (SIG).

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис. 2.1.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис. 2.1.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

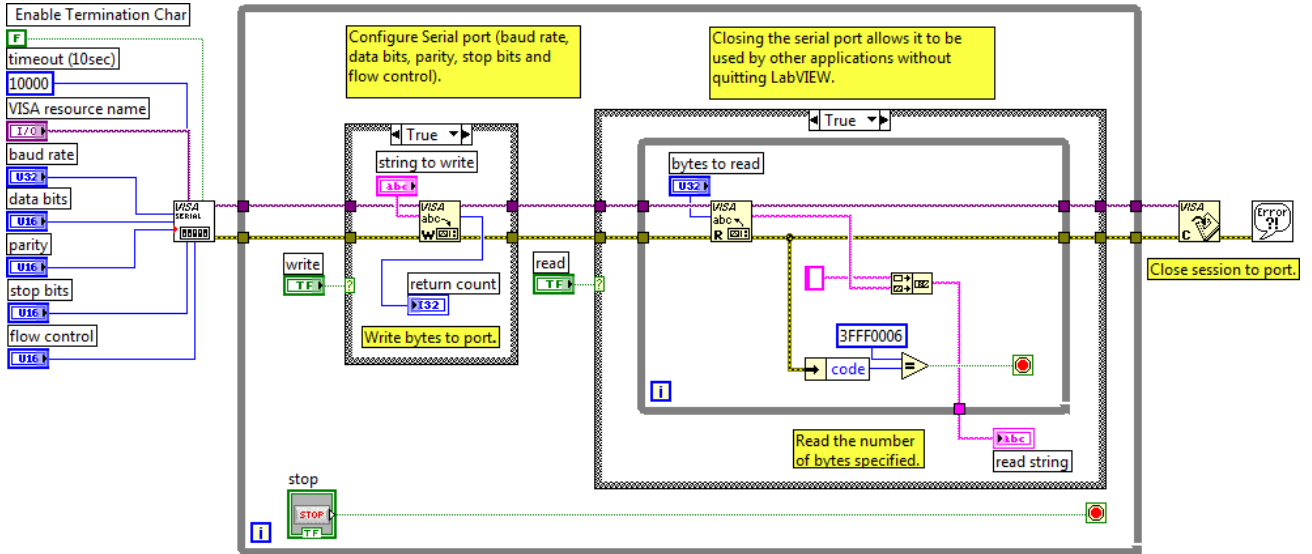


Рис. 2.1.3. Схема приладу в NI LabVIEW 2010 для випадку True

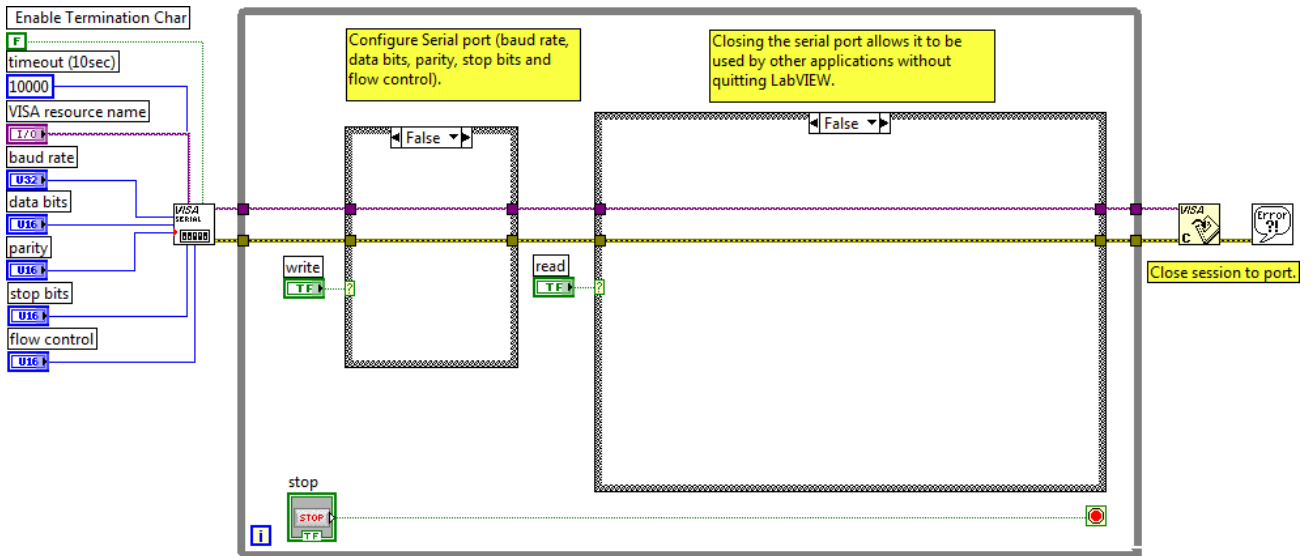


Рис. 2.1.4. Схема приладу в NI LabVIEW 2010 для випадку False

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролера Arduino UNO представлений на рис. 2.1.5.

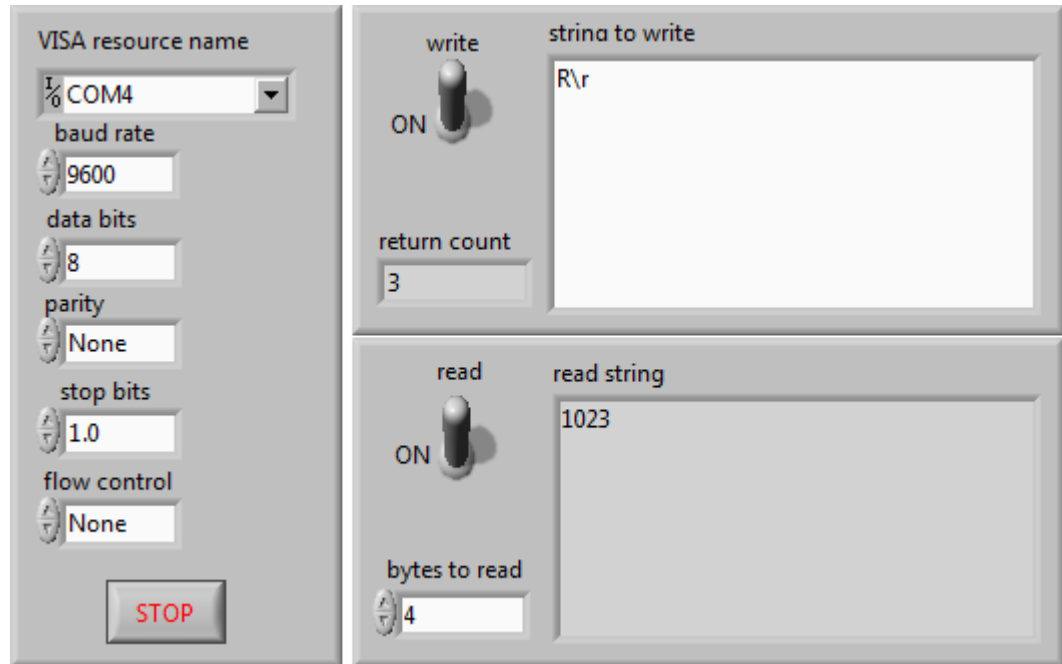


Рис. 2.1.5. Зовнішній вигляд системи керування МПС

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.1.3 і рис. 2.1.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
```

```
#define ctsPin 2 // пін для емнісного датчика торкання
```

```

#define keyPin A0
#define ledPin 5 // пін для світлодіода
void discount(int count, int maxcount); // функція користувача
const int maxcount = 1024;
int keyState = 0;
int count = 0;
int oldstat = 0;
int ctsValue;

```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```

// Налаштування мікроконтролеру
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(ctsPin, INPUT);
  pinMode(keyPin, INPUT);
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```

// Реалізація управління МПС
void loop() {
  ctsValue = digitalRead(ctsPin);
  keyState = analogRead(keyPin);
  if (ctsValue == HIGH){
    // Sig Output in high, сенсор натиснутий
    digitalWrite(ledPin, HIGH);
    Serial.println("TOUCHED");
  } else{
    digitalWrite(ledPin,LOW);
    Serial.println("not touched");
  }
}

```

```

    if (keyState != oldstat) {
        count=0;
        oldstat = keyState;
    }
    discount(count, maxcount);
    delay(500);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

void discount(int count, int maxcount) {
    if (count < maxcount){
        digitalWrite(ledPin, HIGH);
        Serial.println("level Count: ");
        Serial.println(keyState); // display
    }
    else {
        Serial.println("max Count: ");
        Serial.println(keyState); // display
        digitalWrite(ledPin, LOW);
    }
    if (count<=maxcount+1) {
        ++count;
    }
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).

2.2. Універсальний звуковий датчик

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління універсальним звуковим сенсором у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Універсальний звуковий датчик призначений для виявлення звуку і визначення порогового значення звуку (рис. 2.2.1). Звуковий сенсор має наступні модулі: чутливий мікрофон, вбудований компаратор напруги, аналоговий і цифровий вихід.

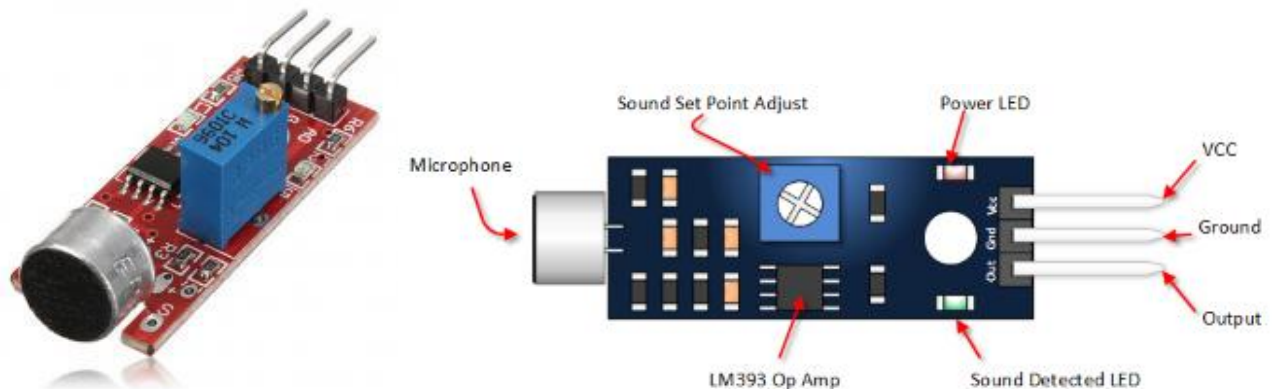


Рис. 2.2.1. Універсальний звуковий датчик

Вбудовані у плату модулі роблять цей датчик зручним для застосування в МПС на основі Arduino UNO і робототехніці. Поріг спрацьовування компаратора LM393 регулюється потенціометром.

Характеристики датчика:

- аналоговий вихід напруги з мікрофона;
- цифровий вихід порогового компаратора;
- компаратор напруги: LM393;
- індикатор живлення;
- індикатор стану цифрового виходу;
- робоча напруга: 4-6В;
- кріпильне отвір 3 мм;
- розміри: 32x17x8мм;

Призначення виводів модуля:

- VCC: живлення 5В постійного струму;
- Ground: земля з плати Arduino UNO;
- Out: передача даних, підключаємо до цифрового входу Arduino;
- Power LED: індикатор живлення, що світиться при подачі живлення;
- Sound Detection LED: світлодіод, що світиться при виявленні звуку;
- Sound Set Point Adjust: CW - збільшення чутливості; CCW - зменшення чутливості.

Підключення мікросхеми універсального датчика звуку до МПС на основі Arduino UNO представлено на рис. 2.2.2.

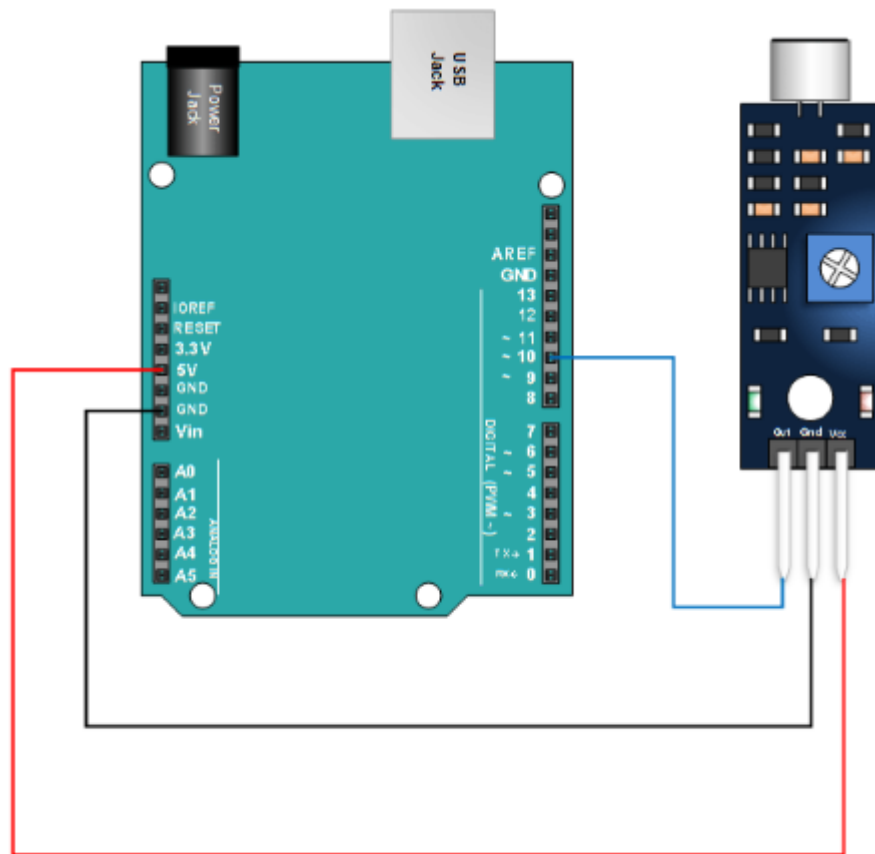


Рис. 2.2.2. Підключення мікросхеми універсального датчика звуку до МПС

Коли рівень звуку перевищує задане значення, на датчику загоряється світлодіод, а вихід переходить у низький рівень (LOW).

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку

True у структурі «Case Structure» представлена на рис 2.2.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.2.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

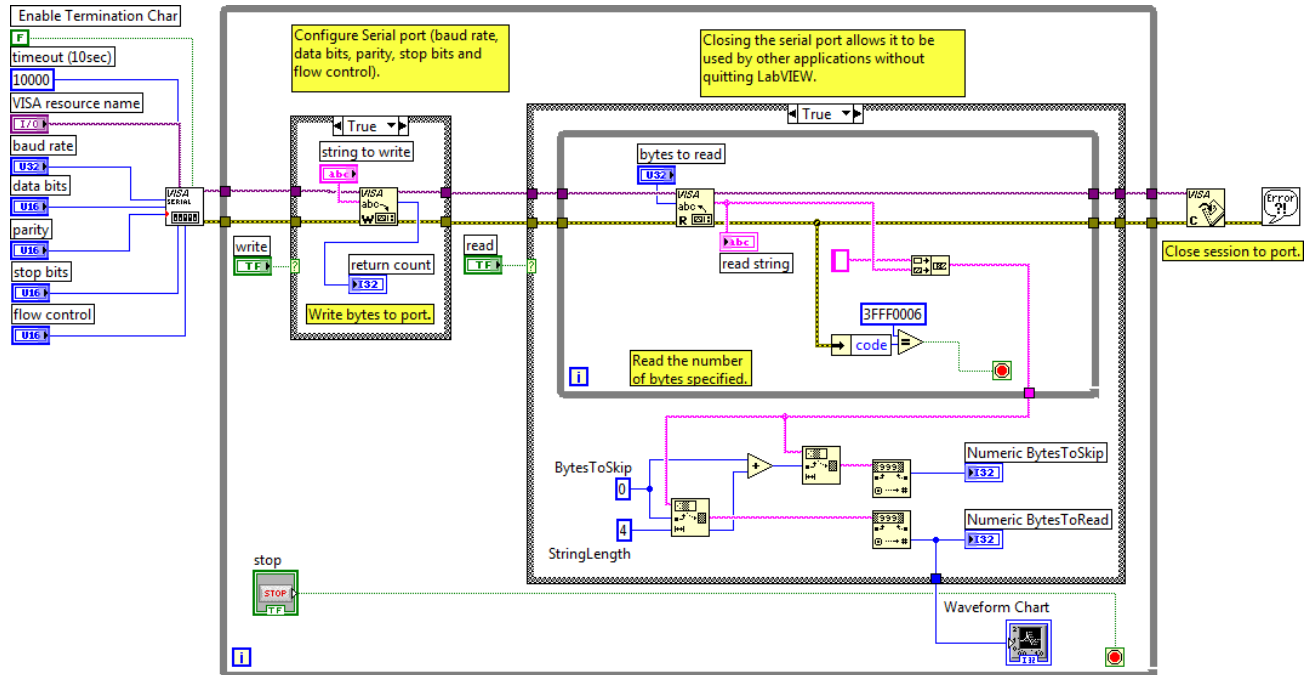


Рис. 2.2.3. Схема приладу в NI LabVIEW 2010 для випадку True

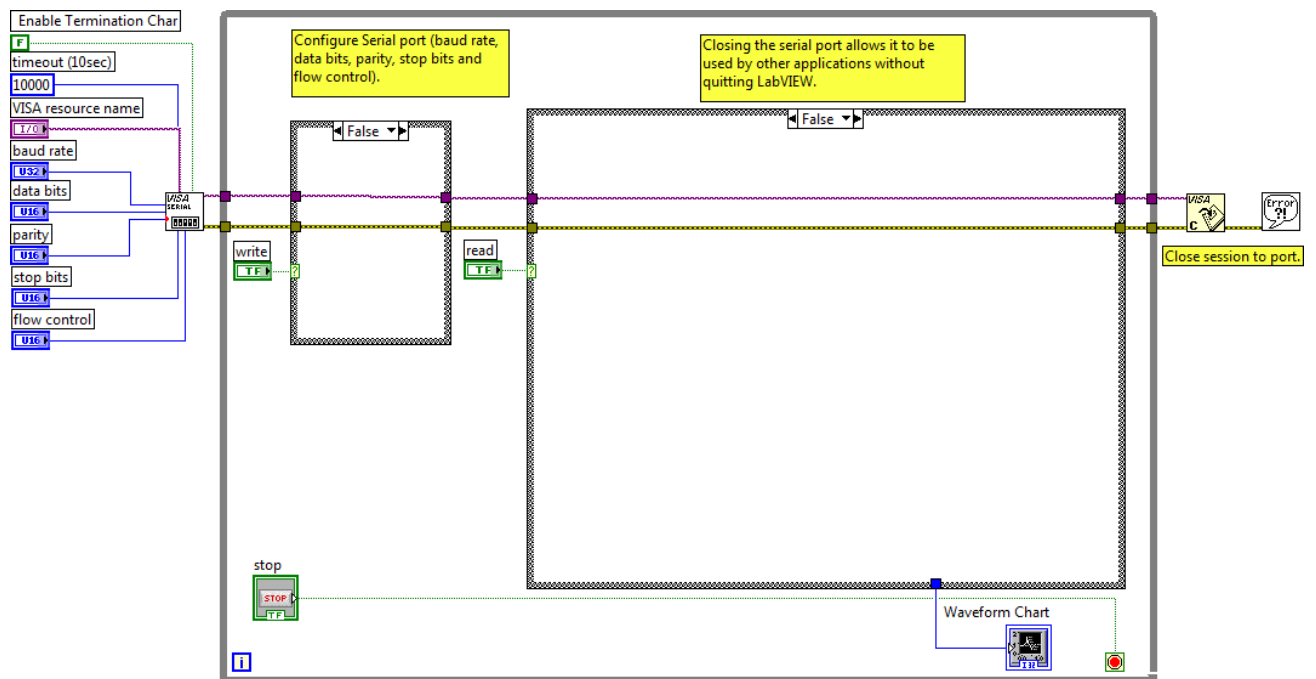


Рис. 2.2.4. Схема приладу в NI LabVIEW 2010 для випадку False

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.2.5.

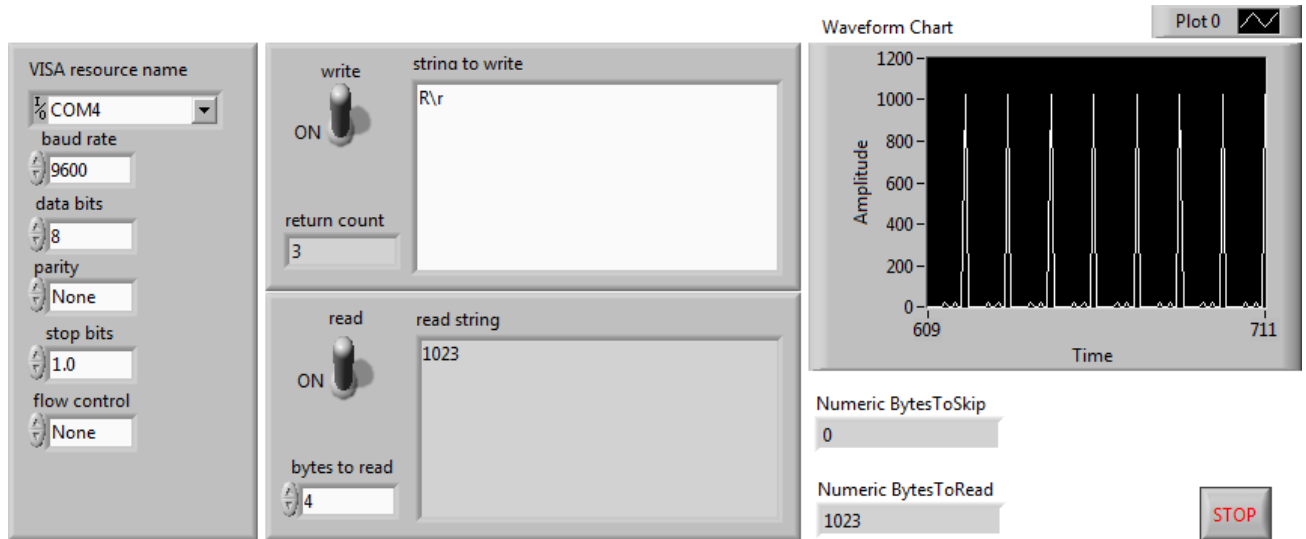


Рис. 2.2.5. Зовнішній вигляд системи керування МПС

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.2.3 і рис. 2.2.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

// Ініціалізація даних і змінних

```

#define ctsPin 10      // пін звукового датчика
int pinRead(int levPin); // функція користувача
int levValue;         // max Value = 1024;
boolean bAlarm = false;
unsigned long lastSoundDetectTime; // Record the time that we measured a sound
int soundAlarmTime = 500;         // Number of seconds to keep the sound alarm high

```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

// Налаштування мікроконтролера

```

void setup() {
  Serial.begin(9600);
  pinMode(ctsPin, INPUT);
  digitalWrite(ctsPin, HIGH);
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

// Реалізація управління МПС

```

void loop() {
  levValue = pinRead(ctsPin);
  if (levValue == LOW){
    lastSoundDetectTime = millis(); // record the time of the sound alarm
    // The following is so you don't scroll on the output screen
    if (!bAlarm){
      // Sig Output Level
      Serial.println("Level = ");
      Serial.println(levValue);
      bAlarm = true;
    }
  } else{
    if( (millis()-lastSoundDetectTime) > soundAlarmTime && bAlarm) {
      Serial.println("not sound");
    }
  }
}

```



```

        bAlarm = false;
    }
}
delay(500);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

int pinRead(int levPin) {
    return digitalRead(levPin);
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи звукового датчику з компаратором LM393.

2.3. Управління кроковим двигуном 28BYJ-48

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління кроковий двигун 28BYJ-48 5В у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

П'ятивольтовий кроковий двигун 28BYJ-48 5В (рис. 2.3.1), використовується в робототехніці, ДІУ-пристроях, поворотних жалюзі кондиціонерів, невеликих вентиляторах і т.п. Всі технічні параметри двигуна відповідають національному електронному стандарту (США) SJ / T10689-95.



Рис. 2.3.1. П'ятивольтовий кроковий двигун 28BYJ-48 5В

Характеристики крокового двигуна:

- номінальна напруга живлення: 5 В (постійний струм)
- кількість фаз: 4
- кількість кроків: 64
- кількість мікрошагів: 4096
- крок: 5.625 Градусів
- номінальна частота: 100 Гц
- номінальний опір обмоток (при 25°C): 50 Ом
- частота холостого ходу (за годинниковою стрілкою): 600 Гц
- частота холостого ходу (проти годинникової стрілки): 1000 Гц
- крутний момент (за годинниковою стрілкою, при частоті 120 Гц): 34,3 Н / м
- крутний момент: 34,3 Н / м
- момент тертя (опір обертанню): 600-1200 г / см
- номінальна тяга: 3500 г / см
- клас електробезпеки: А
- рівень шуму <40dB

Для апаратного управління кроковим двигуном 28BYJ-48 застосовується драйвер крокового двигуна ULN2003, який дозволяє підключати до крокового двигуна зовнішнє живлення 5 - 12 В (рис. 2.3.2).



Рис. 2.3.2. Драйвер крокової двигуна ULN2003

Драйвер крокової двигуна (підходить по роз'єму до крокового двигуна 28BYJ-48 5В), дозволяє працювати як з 5В так і з 12В моторами. Сумісний зі стандартною бібліотекою Arduino Stepper, має 4 світлодіода для індикації роботи (по світлодіоду на кожен канал).

Приклад підключення п'ятивольтового крокового двигуна 28BYJ-48 5В і драйверу крокового двигуна ULN2003 до плати Arduino UNO представлений на рис. 2.3.3.

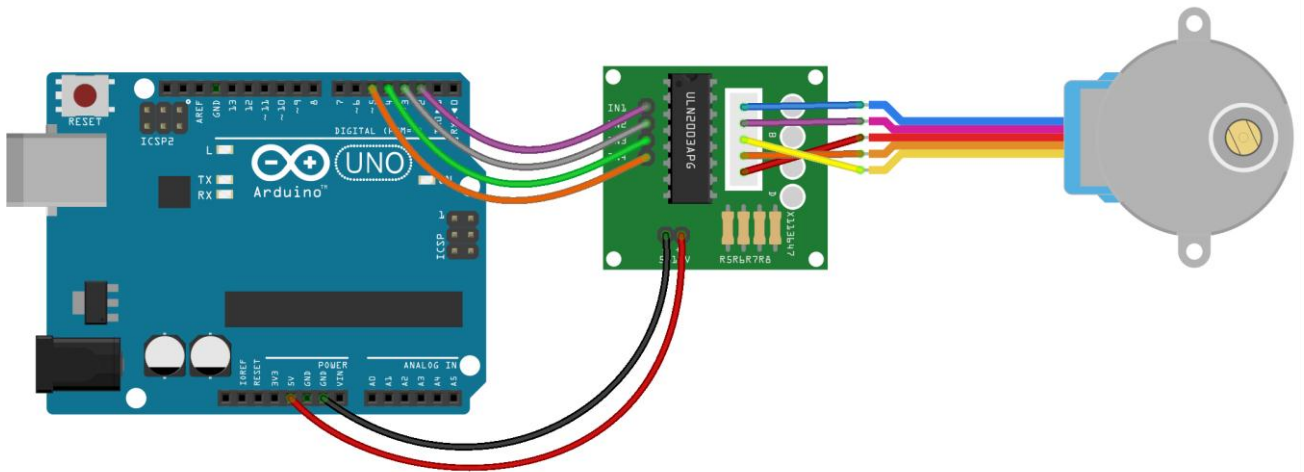


Рис. 2.3.3. Підключення приладів до плати Arduino UNO

Щоб змусити двигун рухатися швидше або повільніше, необхідно змінювати тривалість паузи між комутаціями виводів двигуна. Якщо збільшувати паузу – двигун буде обертатися повільніше, зменшувати паузу – двигун буде крутитися швидше.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.3.4, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.3.5, що відповідає зупинці МПС при натисненні кнопки «Stop».

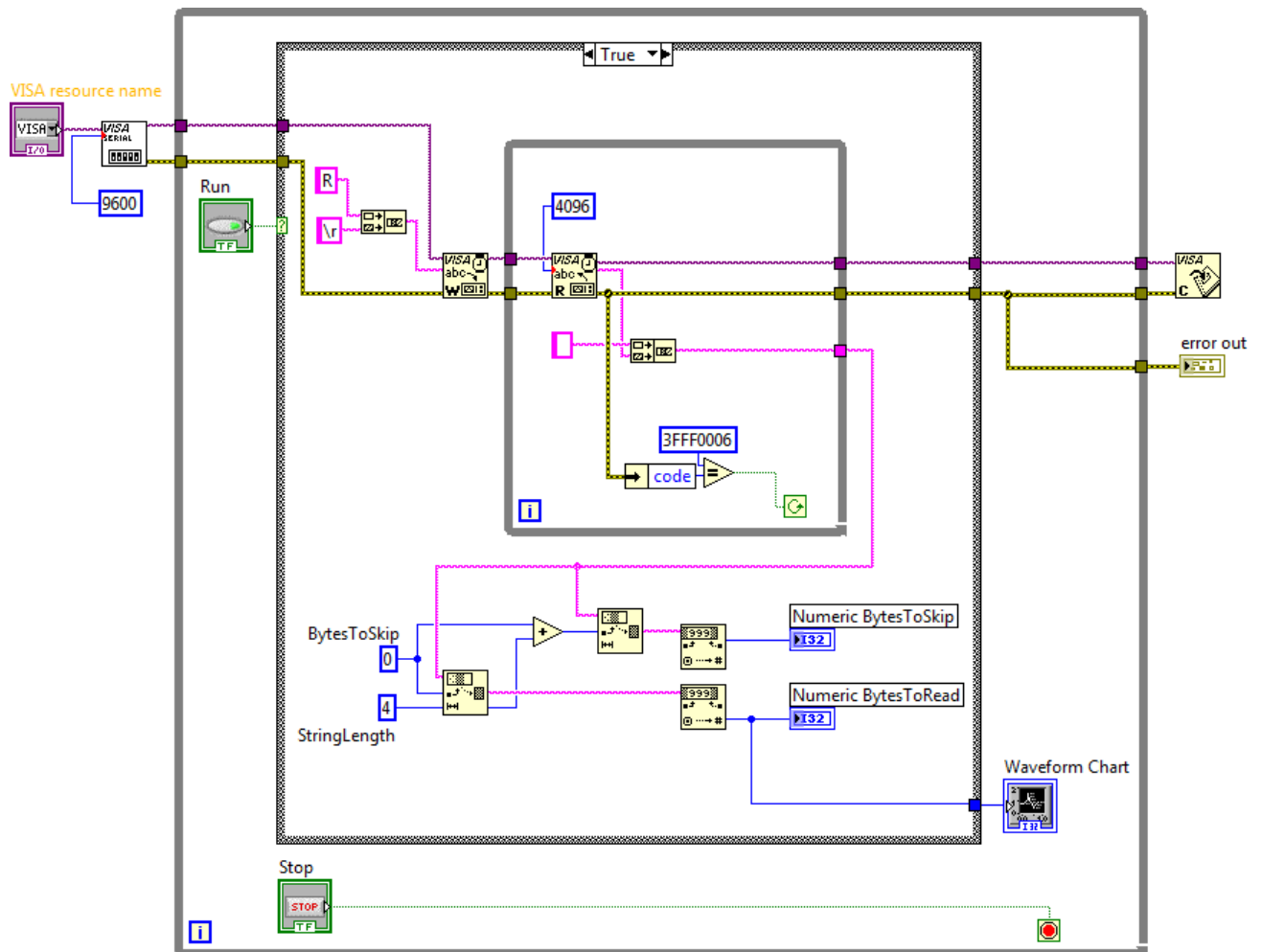


Рис. 2.3.4 Схема приладу в NI LabVIEW 2010 для випадку True

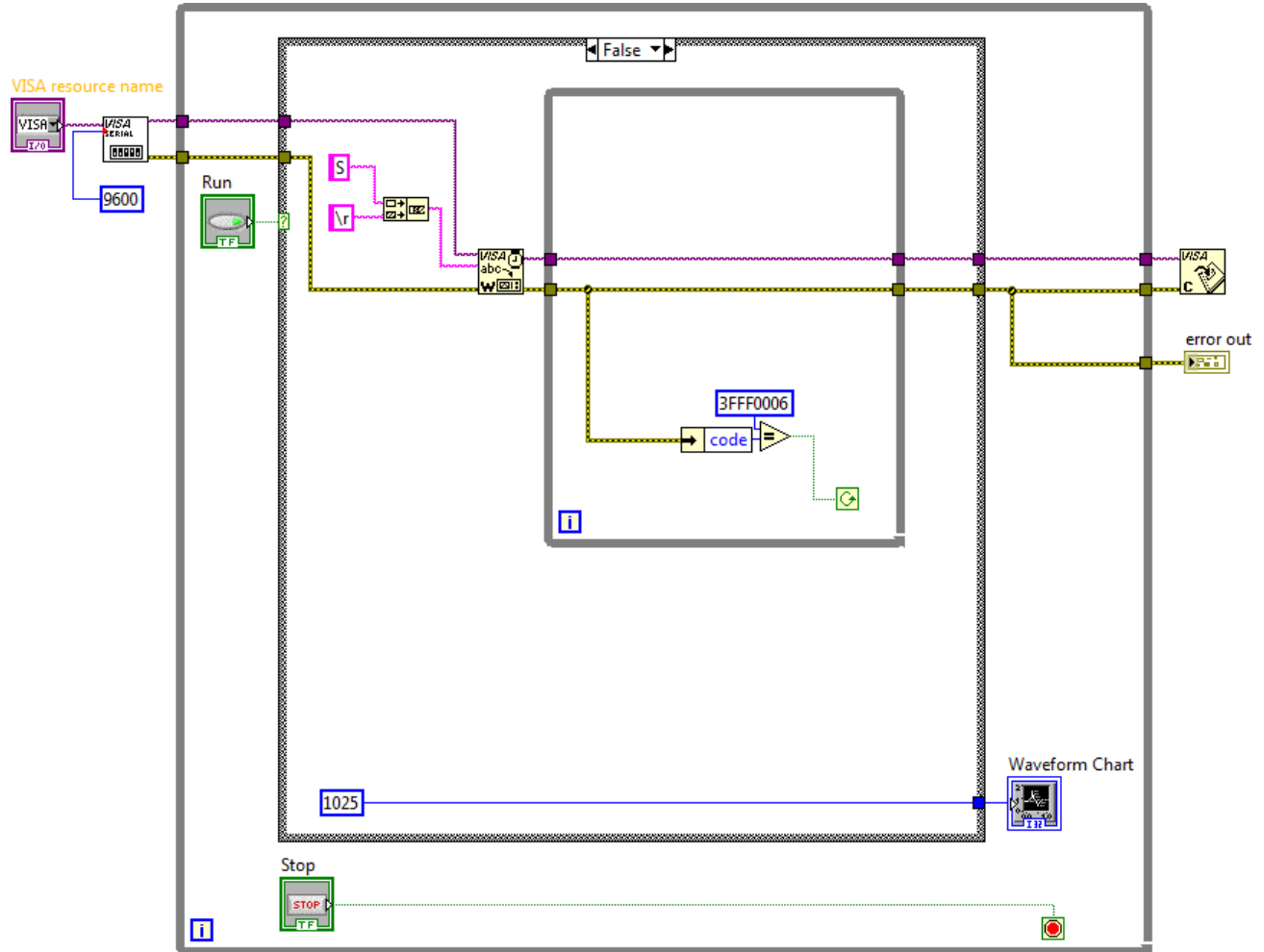


Рис. 2.3.5 Схема приладу в NI LabVIEW 2010 для випадку False

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролера Arduino UNO представлений на рис. 2.3.6.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

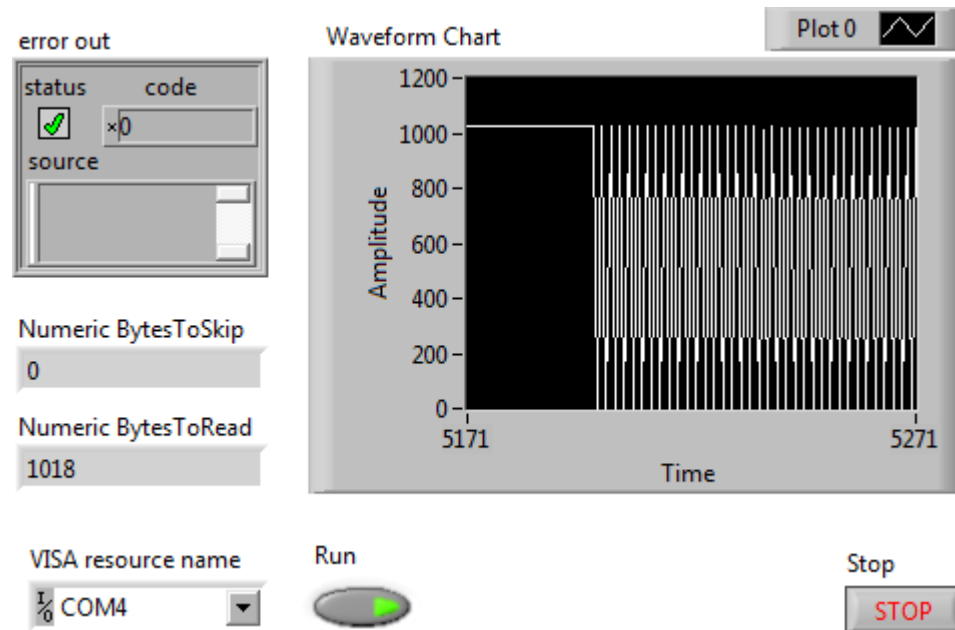


Рис. 2.3.6 Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.3.4 і рис. 2.3.5. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define IN1 8
#define IN2 9
#define IN3 10
#define IN4 11
#define HOLES_DISC 2
unsigned long timeOld;
int Steps = 0;
boolean Direction = true;
unsigned long last_time;
```

```

unsigned long currentMillis ;
int steps_left=24095;
long time;

```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
```

```

void setup() {
  Serial.begin(9600);
  pinMode(IN1, OUTPUT);
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
```

```

void loop() {
  while(steps_left>0) {
    currentMillis = micros();
    if(currentMillis-last_time>=1000) {
      stepper(1);
      time=time+micros()-last_time;
      last_time=micros();
      steps_left--;
    }
    if (millis() - timeOld >= 1000) {
      timeOld = millis();
    }
  }
  Serial.println(time);
  Serial.println("Wait...!");
  delay(2000);
  Direction = Direction;
}

```

```

        steps_left=24095;
    }

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

// Функції користувача
void stepper(int xw){
    for (int x=0; x<xw; x++){
        switch(Steps){
            case 0:
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, LOW);
                digitalWrite(IN3, LOW);
                digitalWrite(IN4, HIGH);
            break;
            case 1:
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, LOW);
                digitalWrite(IN3, HIGH);
                digitalWrite(IN4, HIGH);
            break;
            case 2:
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, LOW);
                digitalWrite(IN3, HIGH);
                digitalWrite(IN4, LOW);
            break;
            case 3:
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, HIGH);
                digitalWrite(IN3, HIGH);
                digitalWrite(IN4, LOW);
            break;
            case 4:
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, HIGH);
                digitalWrite(IN3, LOW);
                digitalWrite(IN4, LOW);
            break;

```



```

    case 5:
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, HIGH);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, LOW);
    break;
    case 6:
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, LOW);
    break;
    case 7:
        digitalWrite(IN1, HIGH);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, HIGH);
    break;
    default:
        digitalWrite(IN1, LOW);
        digitalWrite(IN2, LOW);
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, LOW);
    break;
}
SetDirection();
}
}
void SetDirection(){
    if(Direction==1){ Steps++;}
    if(Direction==0){ Steps--; }
    if(Steps>7){Steps=0;}
    if(Steps<0){Steps=7; }
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролеру Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип комутації виводів крокового двигуна 28BYJ-48.

2.4. Датчик швидкості з цифровим і аналоговим виходами

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком швидкості на мікросхемі LM393 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Датчик швидкості призначений для вимірювання швидкості обертання за допомогою диска, що має прорізи (шторки). Принцип дії полягає у реєстрації переривань світлового випромінювання при його проходженні крізь прорізи (шторки) у об'єкті (рис. 2.4.1).

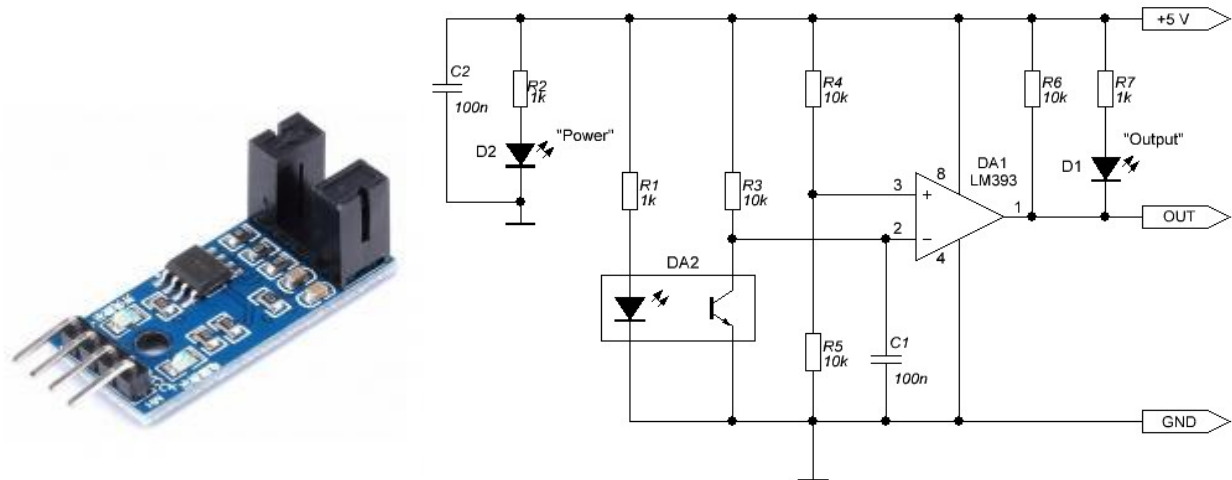


Рис. 2.4.1. Датчик швидкості з цифровим і аналоговим виходами

Цей модуль можна використовувати в якості кінцевого датчика для визначення положення об'єкту (ENDSTOP). Завдяки конструкції датчика відстеження переміщення об'єкту, що є перешкодою для проходження світла перекриває визначається дуже точно.

Характеристики датчика швидкості:

- напруга живлення: від 3,6 до 5В;

- тип виходу: аналоговий і цифровий;
- поточний компаратор: LM393;
- індикатори: живлення, стану виходу;

Призначення виводів модуля:

- VCC: живлення +5В;
- GND: загальна земля;
- DO: цифровий вихід;
- AO: аналоговий вихід.

Підключення датчика швидкості на мікросхемі LM393 до МПС на основі Arduino UNO представлено на рис. 2.4.2.

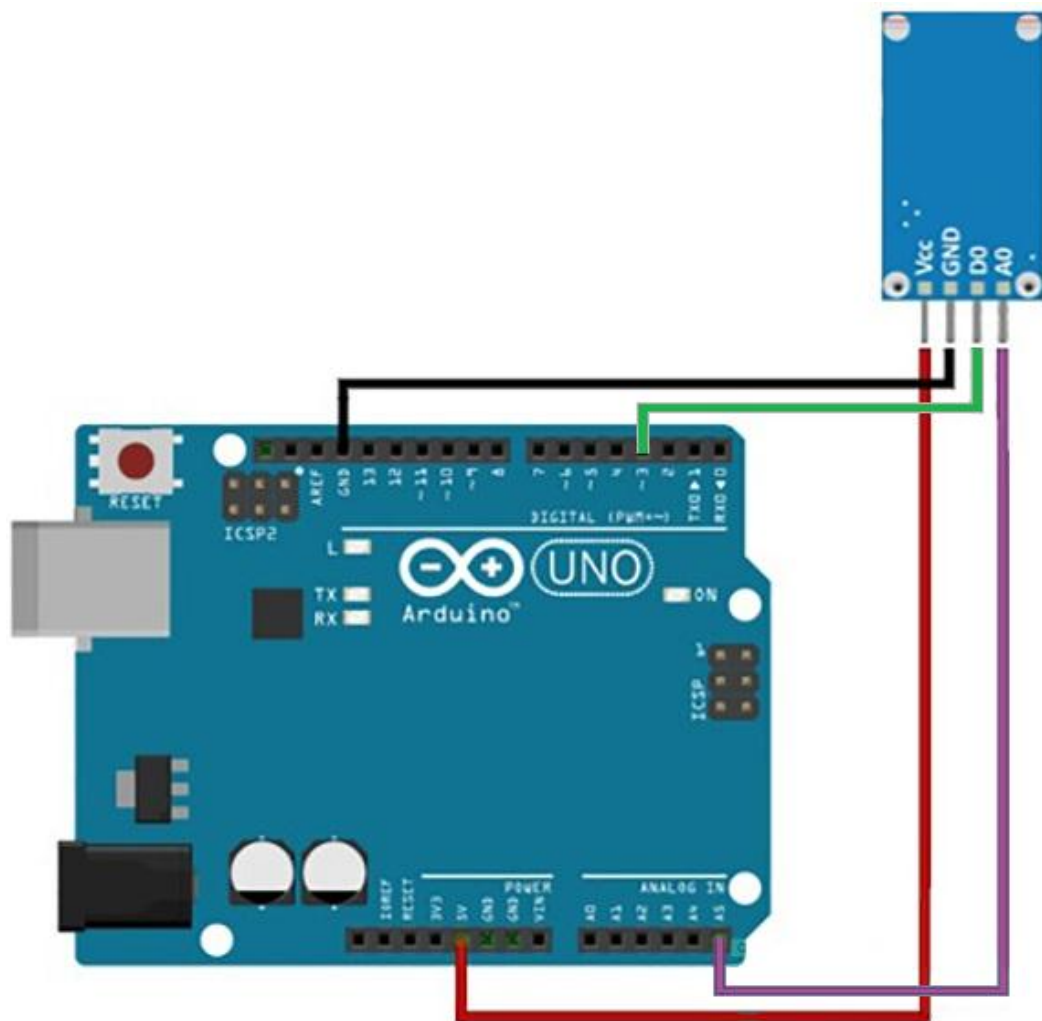


Рис. 2.4.2. Підключення мікросхеми датчика швидкості до МПС

Для підключення плати використовується 4-х контактний роз'єм – напруга живлення + 5В, земля GND, вихід цифровий DO, і вихід аналоговий AT. Працює датчик швидкості зі спеціальними дисками з отворами, які називають щілинними дисками.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.3.4, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.3.5, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.3.6.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.3.4 і рис. 2.3.5. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define IN1 8 // Установка контактів двигуна
#define IN2 9
#define IN3 10
#define IN4 11
```

```

#define HOLES_DISC 2
#define IN_DO 3 // Установка контактів датчика швидкості
float rpm;      // Змінна швидкості
unsigned long timeOld;
volatile unsigned int pulses; // Змінна числа обертів
int Steps = 0;  // Змінна числа кроків
boolean Direction = true;
unsigned long last_time;
unsigned long currentMillis;
int steps_left=24095;
long time;

```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```

// Налаштування мікроконтролера
void setup() {
  Serial.begin(9600);
  pinMode(IN1, OUTPUT); // Установка контактів двигуна
  pinMode(IN2, OUTPUT);
  pinMode(IN3, OUTPUT);
  pinMode(IN4, OUTPUT);
  pinMode(IN_DO, INPUT); // Установка контактів датчика швидкості
  pulses = 0;
  timeOld = 0;
  attachInterrupt(digitalPinToInterrupt(IN_DO), counter, FALLING);
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```

// Реалізація управління МПС
void loop() {
  while(steps_left>0){
    currentMillis = micros();

```

```

// Програмування таймеру
if(currentMillis-last_time>=1000){
    stepper(1);
    time=time+micros()-last_time;
    last_time=micros();
    steps_left--;
}
// Визначення швидкості обертання
if (millis() - timeOld >= 1000) {
    detachInterrupt(digitalPinToInterrupt(IN_DO));
    rpm = (pulses * 60) / (HOLES_DISC);
    Serial.println(rpm); // Вивід швидкості обертання
    pulses = 0;          // Число обертів
    timeOld = millis();
    attachInterrupt(digitalPinToInterrupt(IN_DO), counter, FALLING);
}
}
Serial.println(time);
Serial.println("Wait...!");
delay(2000);
// Зміна напрямку обертання
Direction = Direction;
steps_left=24095;
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

// Функції керування двигуном
void stepper(int xw) {
    for (int x=0; x<xw; x++) {
        switch(Steps) {
            case 0:
                digitalWrite(IN1, LOW);
                digitalWrite(IN2, LOW);

```

```
        digitalWrite(IN3, LOW);
        digitalWrite(IN4, HIGH);
break;
case 1:
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, HIGH);
break;
case 2:
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
break;
case 3:
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    digitalWrite(IN3, HIGH);
    digitalWrite(IN4, LOW);
break;
case 4:
    digitalWrite(IN1, LOW);
    digitalWrite(IN2, HIGH);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
break;
case 5:
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, HIGH);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
break;
case 6:
    digitalWrite(IN1, HIGH);
    digitalWrite(IN2, LOW);
    digitalWrite(IN3, LOW);
    digitalWrite(IN4, LOW);
break;
```

```

        case 7:
            digitalWrite(IN1, HIGH);
            digitalWrite(IN2, LOW);
            digitalWrite(IN3, LOW);
            digitalWrite(IN4, HIGH);
        break;
        default:
            digitalWrite(IN1, LOW);
            digitalWrite(IN2, LOW);
            digitalWrite(IN3, LOW);
            digitalWrite(IN4, LOW);
        break;
    }
    SetDirection(); // Зміна напрямку обертання
}
}
void SetDirection() {
    // Підрахунок числа кроків
    if(Direction==1){ Steps++;}
    if(Direction==0){ Steps--; }
    if(Steps>7){Steps=0;}
    if(Steps<0){Steps=7; }
}
void counter() {
    pulses++; // Число обертів
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).

2.5. Датчик вологості і температури HTU21

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком вологості та температури HTU21 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Датчик вологості та температури HTU21 призначений для точного вимірювання вологості і температури. У датчику застосована мікросхема HTU21 з інтерфейсом I2C, що забезпечує високу точність вимірювань температури $\pm 0,05^{\circ}\text{C}$ (рис. 2.5.1).

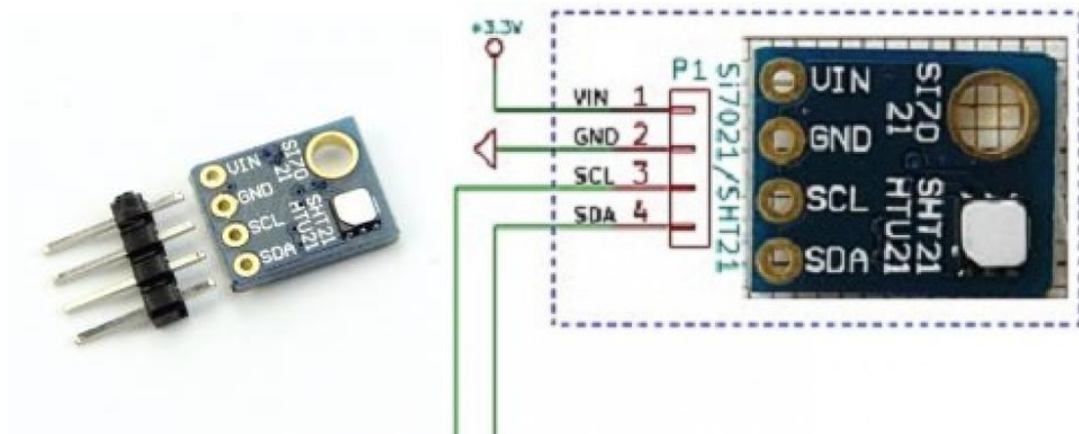


Рис. 2.5.1. Датчик вологості та температури HTU21

Запатентована технологія промислового стандарту, з використанням полімерних діелектриків для зондування вологості, дозволяє створювати CMOS датчики з малим дрейфом, гістерезисом і довгостроковою стабільністю показань. На кристалі розміщені: аналого-цифровий модуль обробки сигналу, дані калібрування і I2C інтерфейс, сенсорний елемент у вигляді монолітного CMOS датчика. Модуль має систему управління з низьким дрейфом і наднизьке споживання енергії.

Характеристики датчика вологості та температури HTU21:

- відносна точність датчика вологості: $\pm 3\%$ RH (макс.) у діапазоні 0 - 80% RH;
- точність датчика температури: $\pm 0,4^{\circ}\text{C}$ (макс) у діапазоні від -10 до 85°C ;

- робочий діапазон: 0 до 100% RH;
- робочий діапазон температур: -40 до + 125°C;
- діапазон робочої напруги датчика: 1.9 - 3.6В;
- напруга живлення модуля: 5 - 6В;
- струм в активному режимі: 150 мкА;
- струм в режимі очікування: 60 пА;
- інтерфейс: I2C;
- адреса I2C пристрої: 0x40;
- нагрівач: інтегрований на платі модуля.
- розміри: 3x3 мм DFN корпус.

Підключення датчика вологості та температури HTU21 до МПС на основі Arduino UNO представлено на рис. 2.5.2.

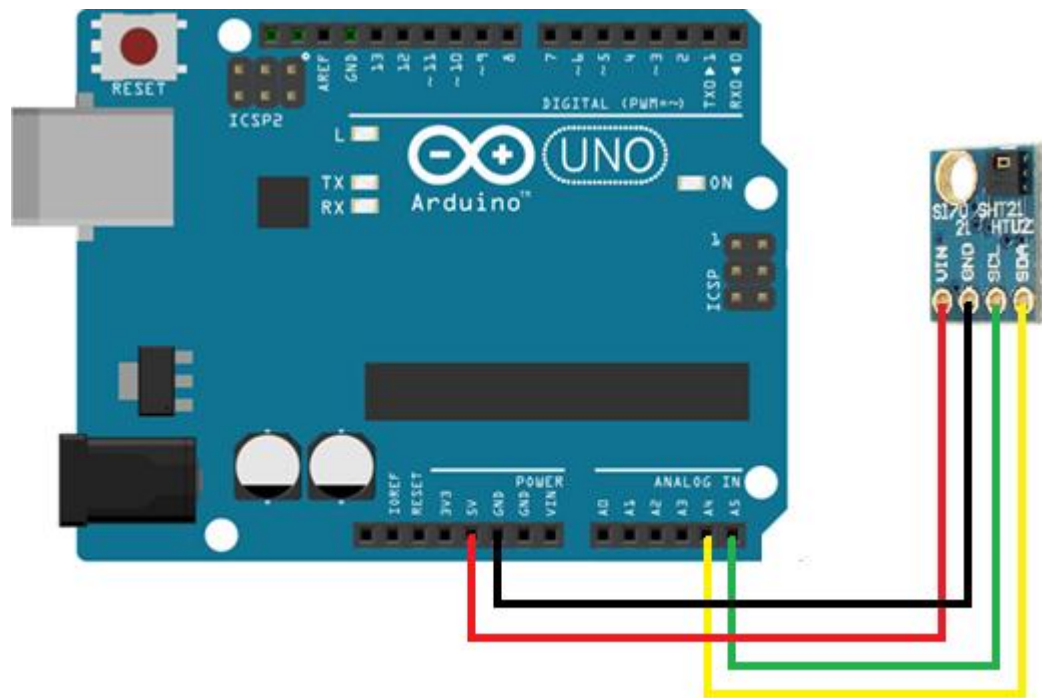


Рис. 2.5.2 Підключення датчика вологості та температури HTU21 до МПС

Дані калібрування записані на заводі-виробнику і зберігаються в незалежній пам'яті. Це гарантує, що датчики повністю взаємозамінні, без калібрування або зміни програмного забезпечення.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку Numeric = 1 у структурі «Case Structure» представлена на рис 2.5.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку Numeric = 0 у структурі «Case Structure» представлена на рис 2.5.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

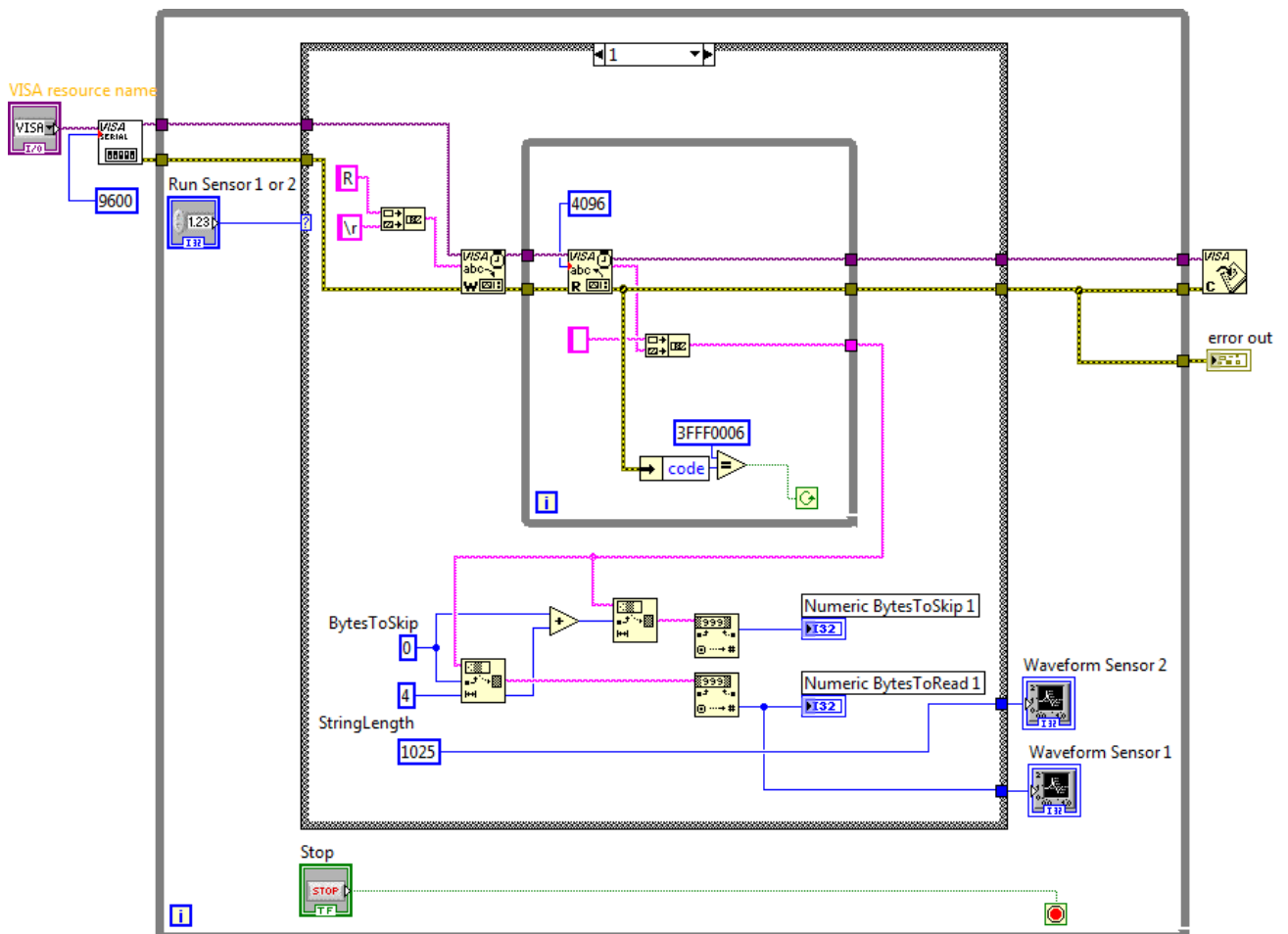


Рис. 2.5.3. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 1

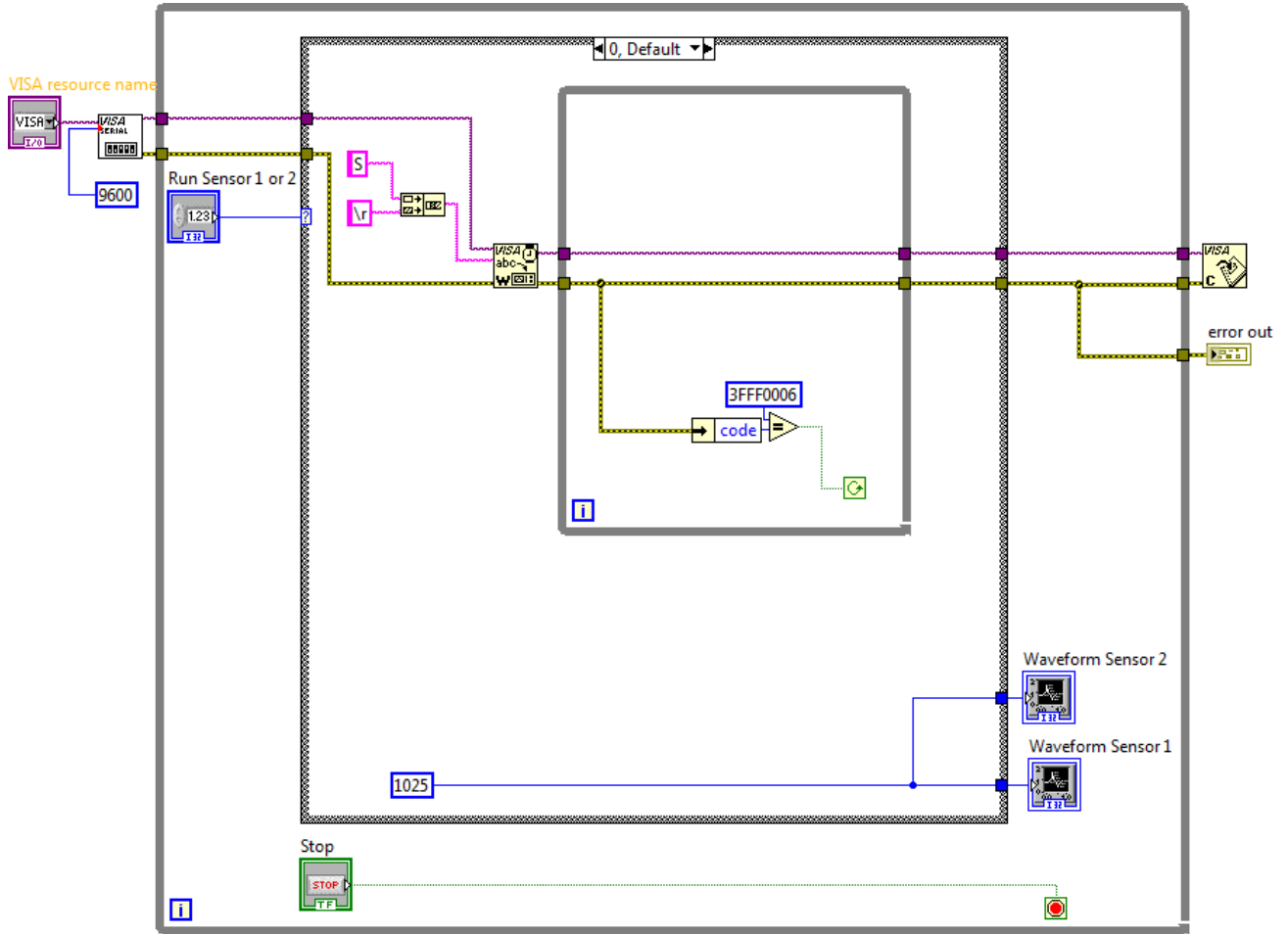


Рис. 2.5.4. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 0

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролера Arduino UNO представлений на рис. 2.5.5.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

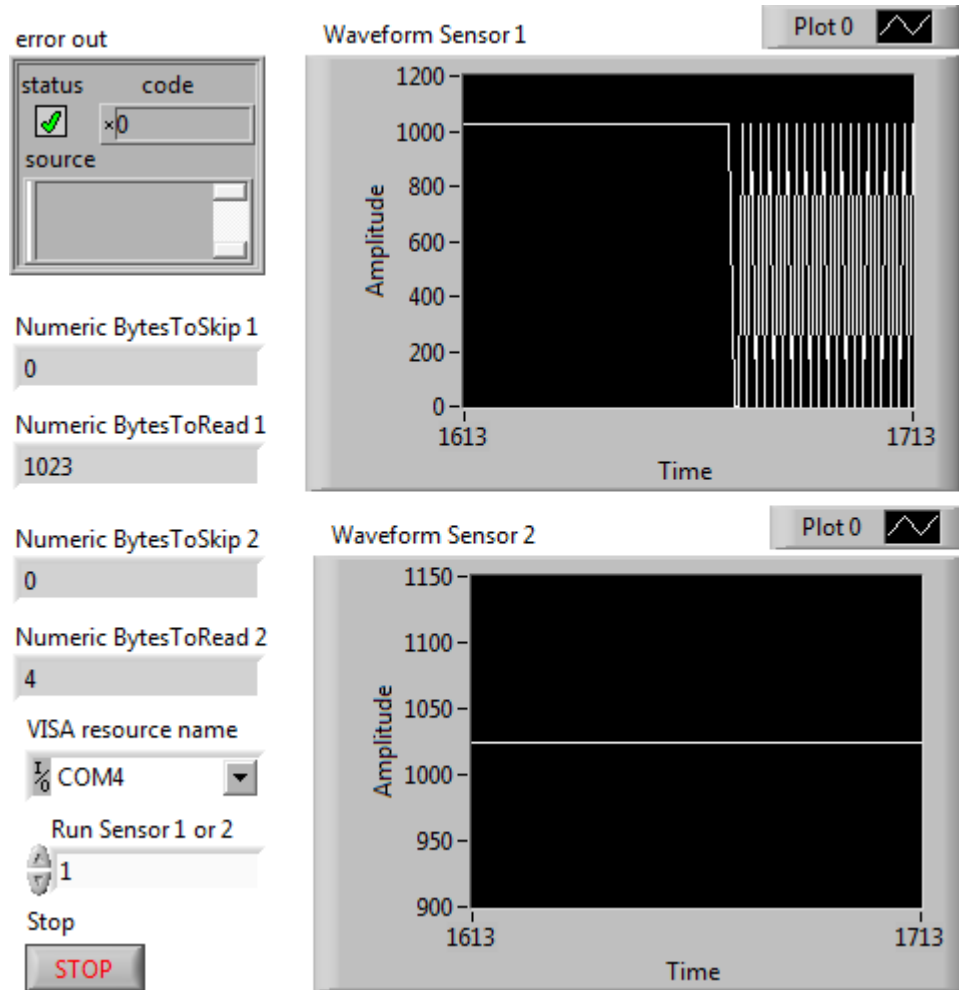


Рис. 2.5.5 Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.5.3 і рис. 2.5.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

// Ініціалізація даних і змінних

/* Hardware Connections (Break outboard to Arduino):

VCC = 3.3V

```

GND = GND
SDA = A4 (use inline 10k resistor if your board is 5V)
SCL = A5 (use inline 10k resistor if your board is 5V) */
#include <Wire.h>
#include "HTU21D.h" // Бібліотека датчика температури HTU21
int count = 0;      // Контроль включення сенсору
int oldStat = 0;
HTU21D myHumidity; // Create an instance of the object
float ReadTemp();  // Функція користувача для температури
float ReadHumd();  // Функція користувача для вологості

```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```

// Налаштування мікроконтролера
void setup() {
  Serial.begin(9600);
  Serial.println("HTU21D Sensor!");
  myHumidity.begin();
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```

// Реалізація управління МПС
void loop() {
  count=Serial.read();
  if (count<0) {
    delay(100);
    count=oldStat;
  }
  if(count=='R') {
    float humd, temp;
    humd = ReadHumd();
    temp = ReadTemp();
  }
}

```

```

        Serial.print("Time:");          // Вивід часу
        Serial.print(millis());
        Serial.print(" Temperature:"); // Вивід температури
        Serial.print(temp, 1);
        Serial.print("C");
        Serial.print(" Humidity:");    // Вивід вологості
        Serial.print(humd, 1);
        Serial.print("%");
        Serial.println();
        delay(1000);
        count = 'R';
    }
    if (count=='S') {
        delay (100);
        count='S';
    }
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

// Вимірювання температури
float ReadTemp() {
    return myHumidity.readTemperature();
}

// Вимірювання вологості
float ReadHumd() {
    return myHumidity.readHumidity();
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть застосування CMOS датчиків для вимірювання температури.

2.6. Цифровий модуль з термістором

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління цифровим модулем з термістором у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Цифровим модулем з термістором призначений для вимірювання температури навколишнього повітря або закритого простору (рис. 2.6.1).

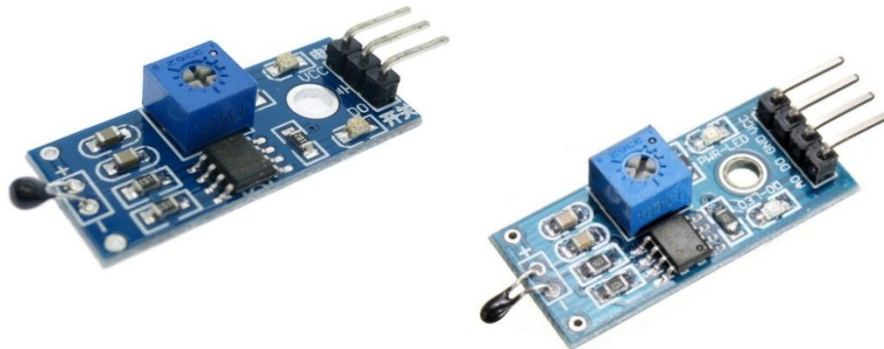


Рис. 2.6.1. Цифровим модулем з термістором

Висока точність і можливість безпосередньо підключати його до контролера Arduino або до модулю реле дозволяють широко використовувати його в конструкціях, де потрібно точно і швидко вимірювати температуру: термометри, термостати, системи вентиляції та кондиціонування повітря.

Характеристики цифрового модуля:

- тип датчика: NTC термістор;
- діапазон вимірюваної температури: 20 - 80°C;

- макс. вихідний струм: 15 мА;
- цифровий компаратор: LM393;
- поріг спрацьовування виходу: налаштовується;
- вбудовані індикатори: напруги живлення і стану виходу;
- робоча напруга: 3 - 5В;
- розмір: 3.2 x 1.4 см.

Підключення цифрового модуля з термістором до МПС на основі Arduino UNO представлено на рис. 2.6.2.

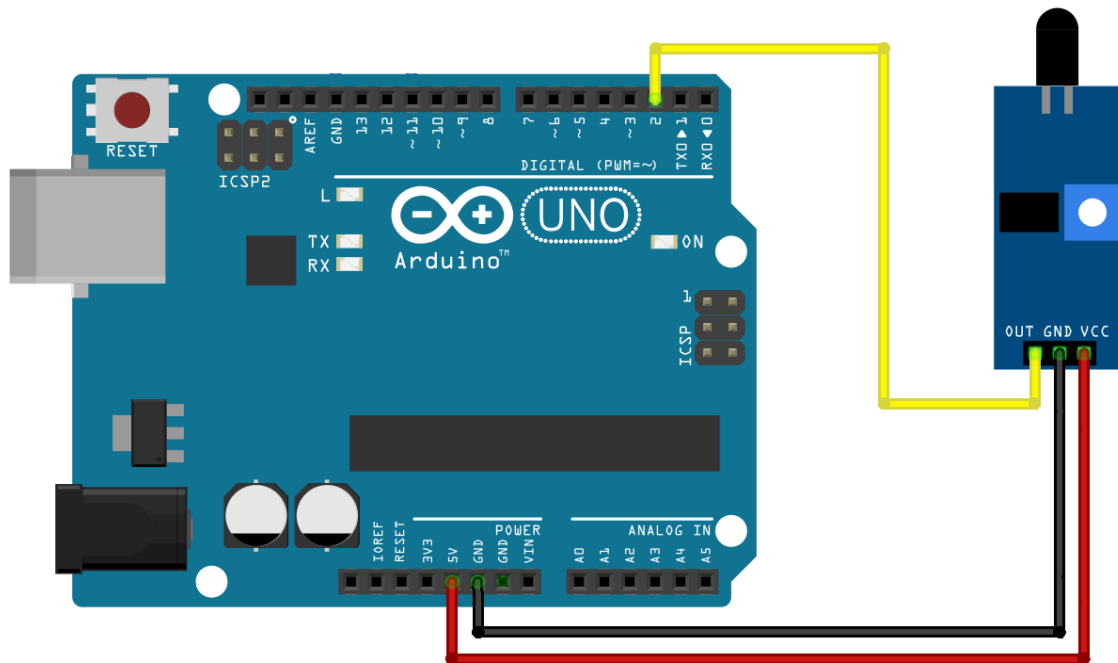


Рис. 2.6.2. Підключення цифрового модуля з термістором до МПС

Використаний цифровий модуль з термістором вимірює температуру навколишнього середовища з точністю $\pm 0,5^{\circ}\text{C}$.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку Numeric = 2 у структурі «Case Structure» представлена на рис 2.6.3, що відповідає запуску

МПС. Блок-схема віртуального приладу для випадку Numeric = 0 у структурі «Case Structure» представлена на рис 2.6.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

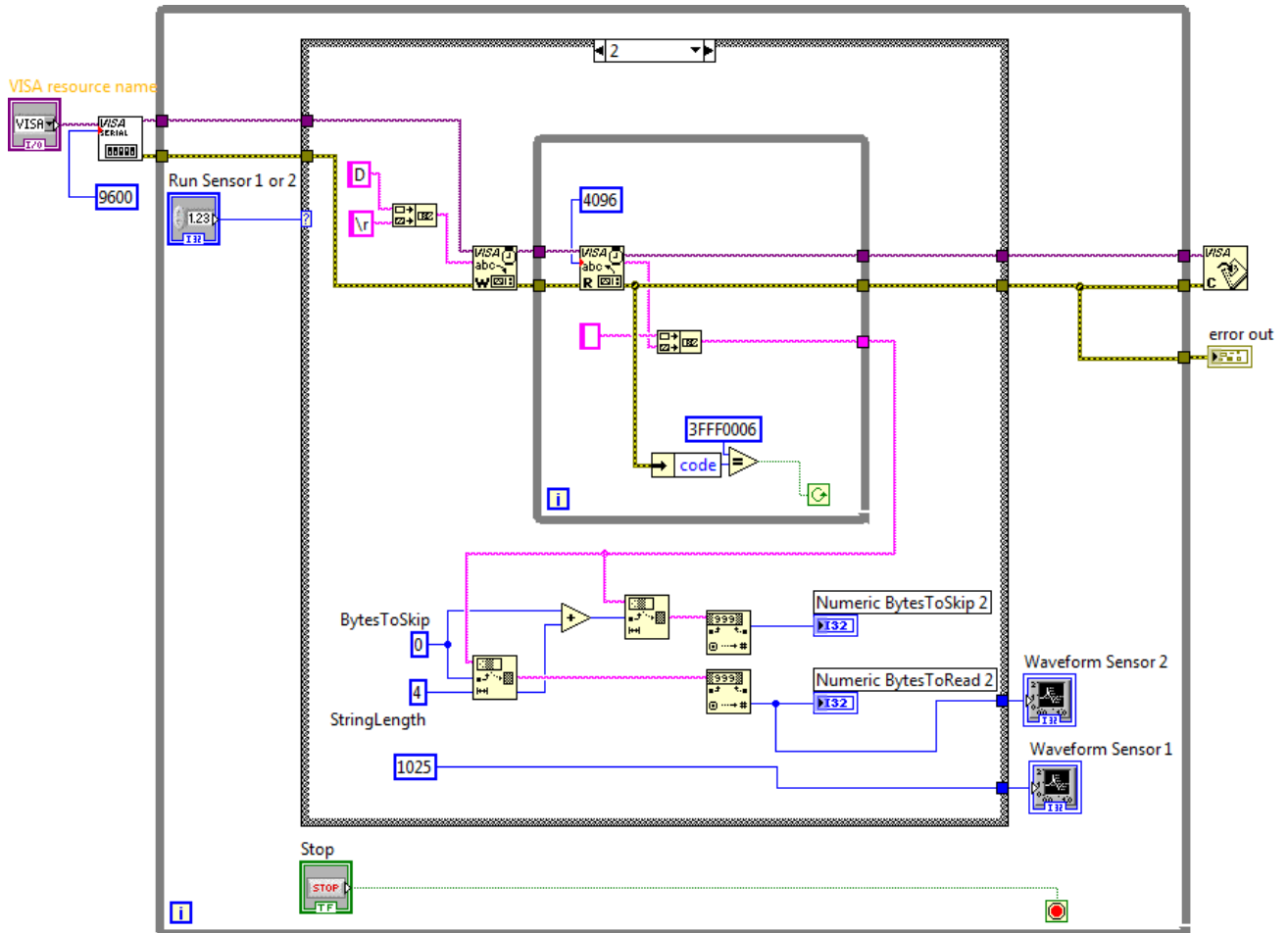


Рис. 2.6.3. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 2

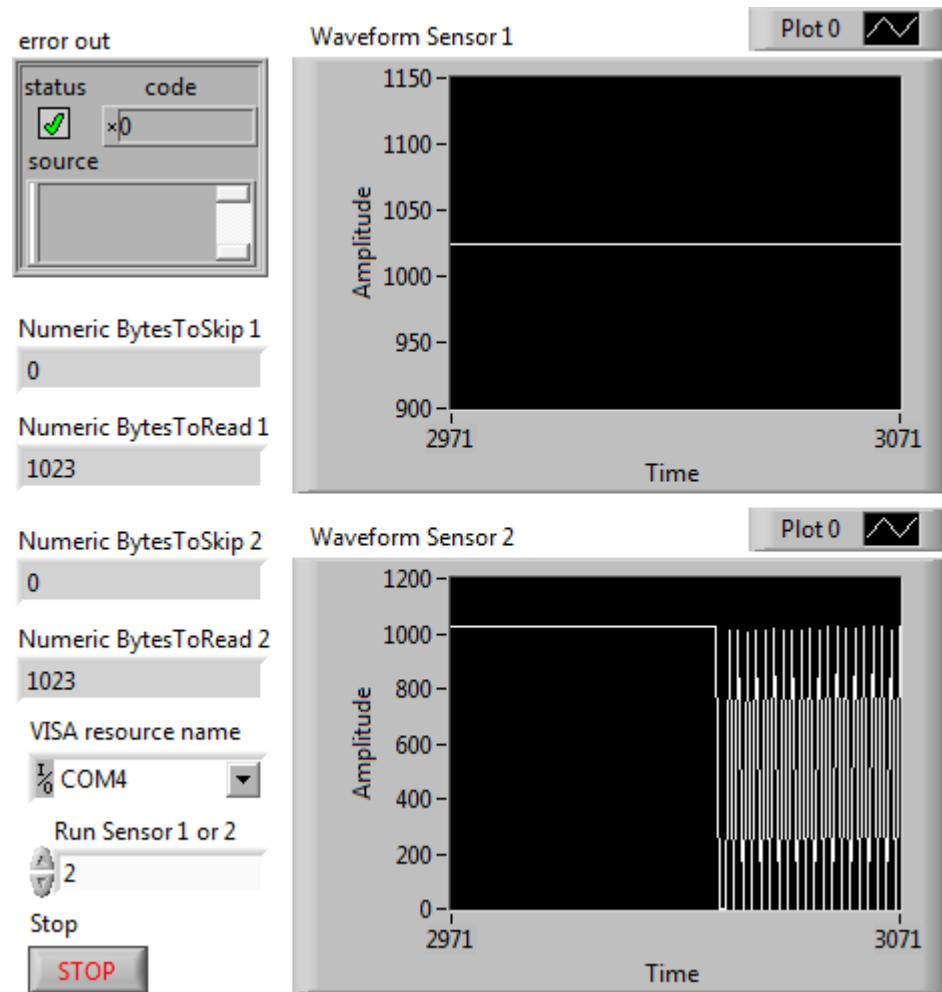


Рис. 2.6.5. Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.6.3 і рис. 2.6.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
```

```
#define sensorPin A2 // Select the input pin for the potentiometer;
```

```
int count = 0;           // Контроль включення сенсору;
int oldStat = 0;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
```

```
void setup() {
  Serial.begin(9600);
  pinMode (sensorPin, INPUT);
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
```

```
void loop() {
  count=Serial.read();
  if (count<0) {
    delay(100);
    count=oldStat;
  }
  if(count=='D') {
    int readVal=analogRead(sensorPin); // Читання цифрових даних
    double temp = Thermistor(readVal); // Функція користувача
    if (readVal != 0) {
      Serial.println("Convert Kelvin to Celsius:");
      Serial.println(temp); // display temperature
      Serial.println("Temperature of Thermistor:");
      Serial.println(readVal); // display temperature
    }
  }
  else {
    pinMode (sensorPin, LOW);
    Serial.println("not measurement");
  }
}
```

```

        delay(500);
        count = 'D';
    }
    if (count=='S') {
        delay (100);
        count='S';
    }
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

double Thermistor(int RawADC) {
    double Temp;
    Temp = log(10000.0*((512.0/RawADC-1)));
    Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp * Temp ))*
Temp );
    Temp = 273.15 - Temp;           // Convert Kelvin to Celsius
    // Temp = (Temp * 9.0)/ 5.0 + 32.0; // Convert Celsius to Fahrenheit
    return Temp;
}

```

Завантажити програмне забезпечення для мікроконтролеру Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролеру Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип застосування термісторів для вимірювання температури.

2.7. Ультразвуковий датчик відстані HC-SR04

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління ультразвуковим датчиком відстані HC-SR04 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Ультразвуковий датчик HC-SR04 – це стабільний і точний сенсор (Ultrasonic Sonar) для вимірювання відстані, який не має "сліпих зон". Може вимірювати відстань до об'єкту від 0 мм до 1500мм з точністю до 3 мм (рис. 2.7.1).



Рис. 2.7.1. Ультразвуковий датчик відстані HC-SR04

Принцип роботи ультразвукового датчика HC-SR04 наступний:

1. На вихід Trig (тригер) посилається високий рівень протягом як мінімум 10мкс;
2. Модуль починає посылати ультразвукові імпульси з частотою 40 кГц і приймає їх назад, якщо в зоні видимості є будь-які перешкоди;

3. Якщо сигнал повертається, модуль встановлює низький рівень на виході Echo на 150мс. За часом, який минув з встановлення на виході Trig високого рівня сигналу до низького рівня на виході Echo можна розрахувати відстань до перешкоди за формулою:

$$\text{Distance} = (\text{time} * \text{sound velocity}) / 2,$$

де time – виміряний час імпульсу,

sound velocity – швидкість звуку (340 м/с).

Характеристики ультразвукового датчика HC-SR04:

– робоча напруга: 3.8 - 5.5В;

- тип сенсору: HC-SR04;
- струм: 8 мА;
- частота: 40 кГц;
- максимальна дистанція: 1500 мм;
- мінімальна дистанція: 0 см;
- роздільна здатність: 3 мм;
- ширина імпульсів: 10 мкс;
- кут: 15 градусів;
- зовнішні габарити: 37х20х15 мм.

Підключення ультразвукового датчика HC-SR04 до МПС на основі Arduino UNO представлено на рис. 2.7.2.

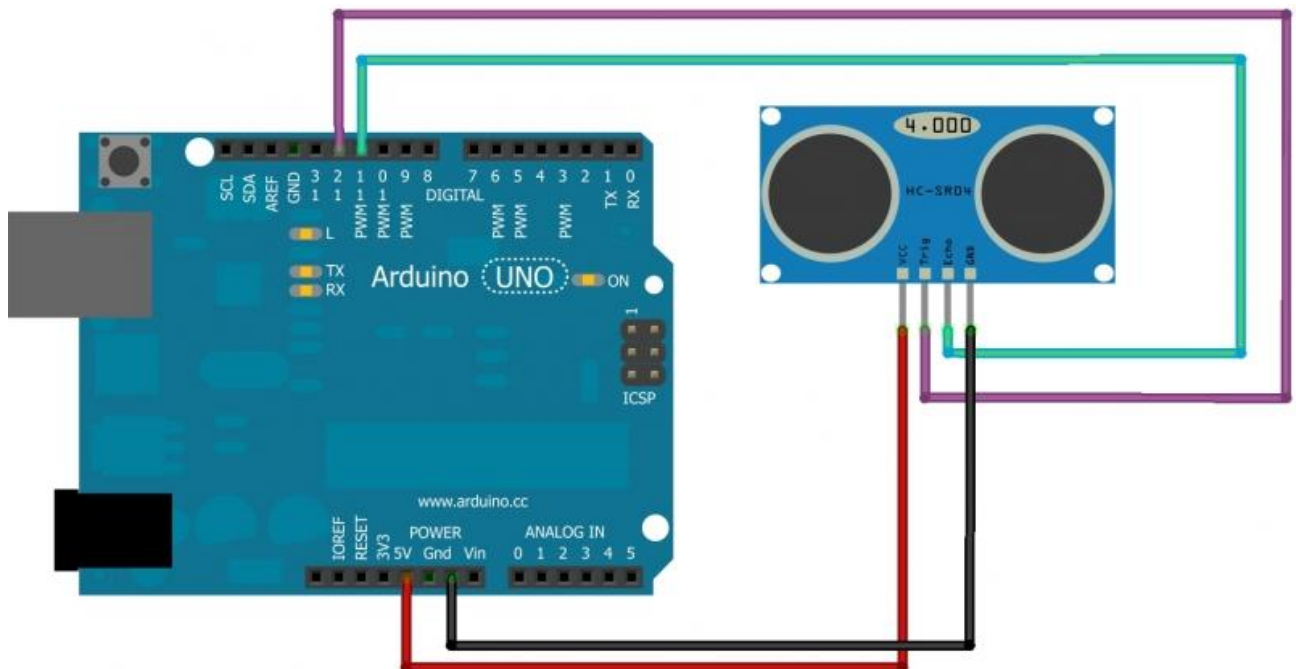


Рис. 2.7.2. Підключення ультразвукового датчика HC-SR04 до МПС

Більшість датчиків відстаней, що підключаються до контролера Arduino UNO мають 4 - 5 виводів, яких достатньо для нормальної роботи з МПС: живлення, земля, тригер, вихід.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino

UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.7.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.7.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

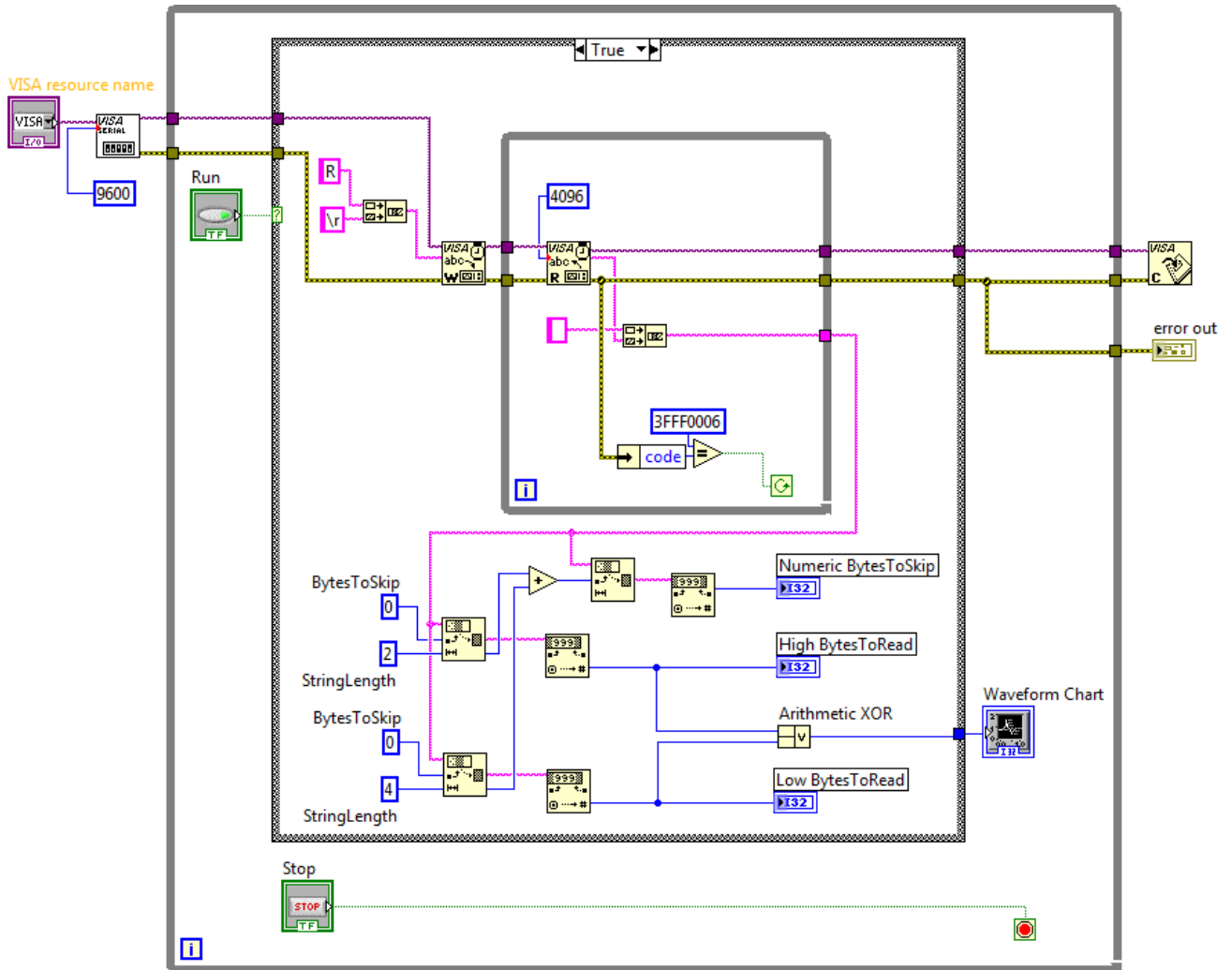


Рис. 2.7.3. Схема приладу в NI LabVIEW 2010 для випадку True

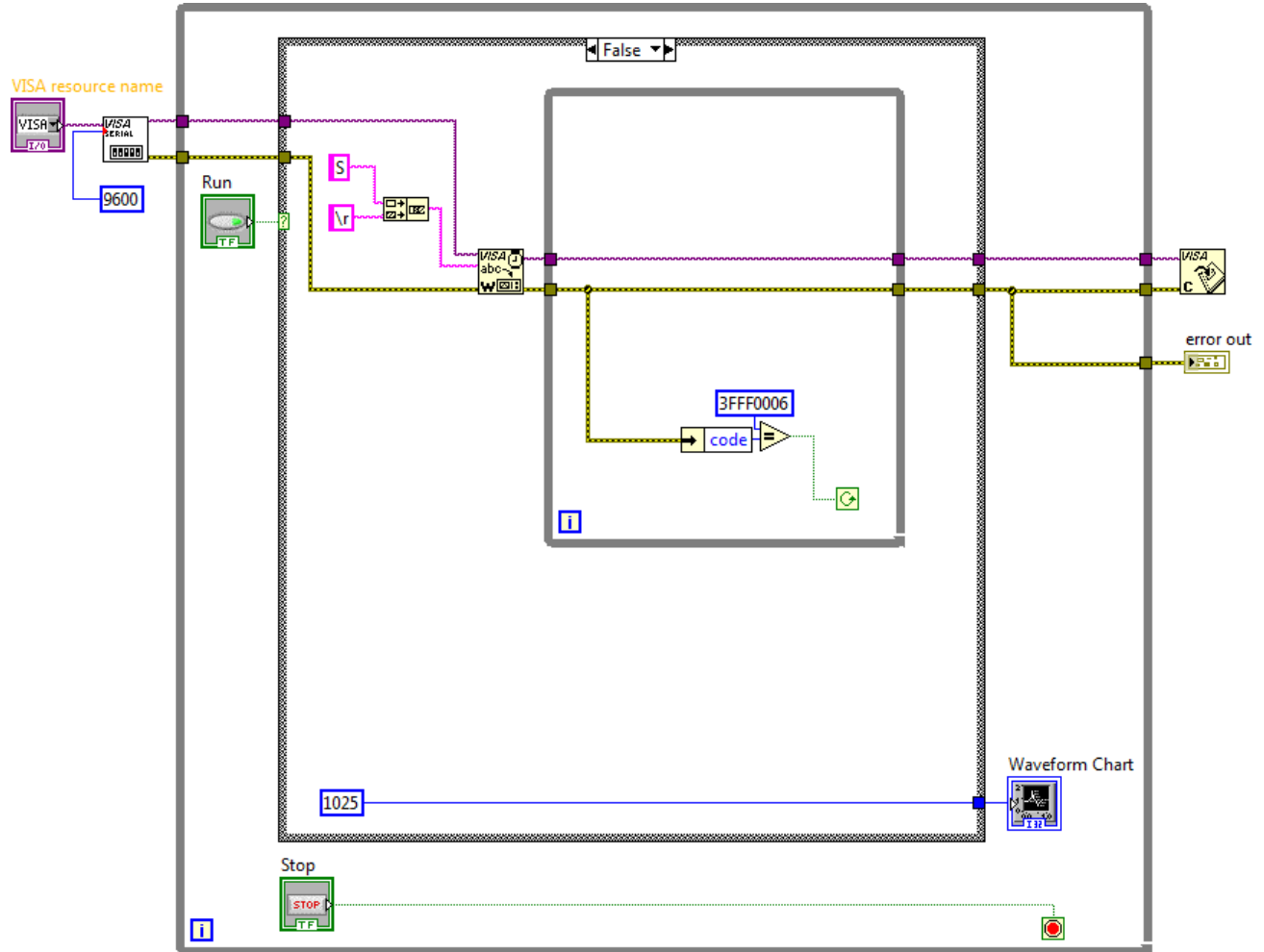


Рис. 2.7.4 Схема приладу в NI LabVIEW 2010 для випадку False

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.7.5.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

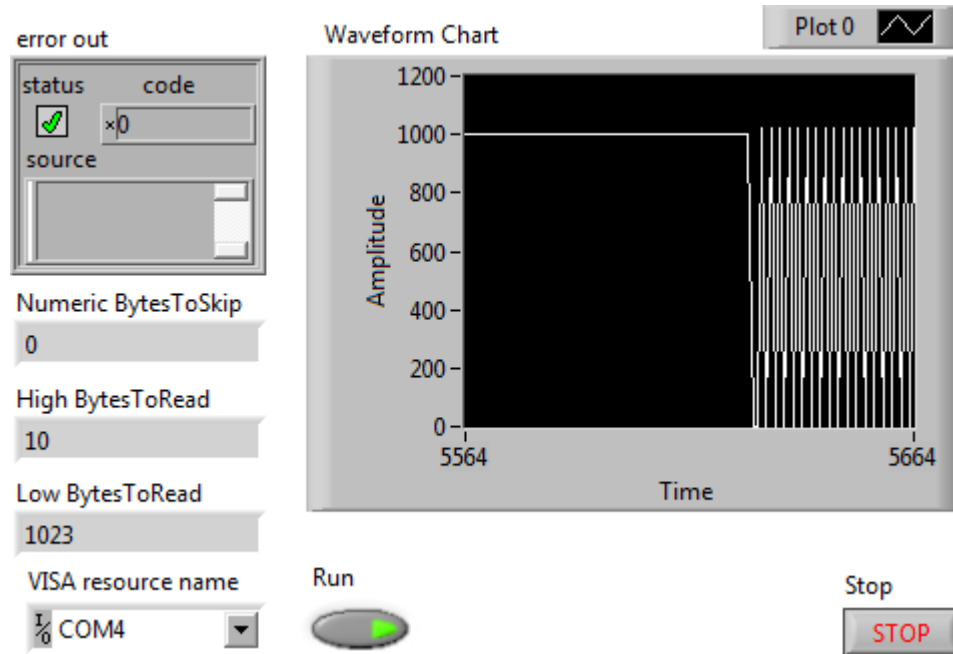


Рис. 2.7.5. Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.7.3 і рис. 2.7.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define echoPin 4 // Echo Pin
#define trigPin 5 // Trigger Pin
int maximumRange = 200; // Maximum range needed
int minimumRange = 0; // Minimum range needed
long duration, distance; // Duration used to calculate distance;
long ReadSensor(int setPin); // Measurement of distance
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
```

```
void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
};
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
```

```
void loop() {
  /* The following trigPin/echoPin cycle is used to determine the
  distance of the nearest object by bouncing sound waves off of it. */
  // Turn Ultrasonic Sensor:
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  //Calculate the distance (in cm) based on the speed of sound:
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  duration = ReadSensor(echoPin); // Measurement of distance
  distance = duration/58.2;
  if (distance >= maximumRange || distance <= minimumRange) {
    /* Send a negative number to computer and Turn LED ON
    to indicate "out of range" */
    Serial.println("-1");
    digitalWrite(trigPin, HIGH);
  }
  else {
    /* Send the distance to the computer using Serial protocol, and
    turn LED OFF to indicate successful reading. */
    Serial.println("Distance = ");
```

```

        Serial.println(distance);
        digitalWrite(trigPin, LOW);
    }
    //Delay 50ms before next reading.
    delay(500);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

// Функція зчитування відстані
long ReadSensor(int setPin) {
    return pulseIn(setPin, HIGH);
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Які змінні входять до формули розрахунку відстані до перешкоди?
5. Поясніть принцип роботи ультразвукового датчика відстані HC-SR04.

2.8. Цифровий датчик перешкоди

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком перешкоди у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Датчик перешкод призначений для визначення наявності оточуючих об'єктів. Цифровий вихід датчика можна безпосередньо підключати до Arduino або до інших мікроконтролерів (рис. 2.8.1).

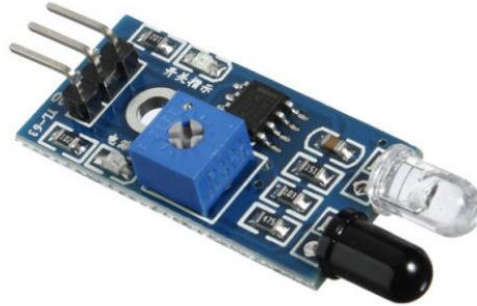


Рис. 2.8.1. Цифровий датчик перешкод

Характеристики цифрового датчика перешкод:

- відстань виявлення перешкоди: від 2 до 30 см;
- робоча напруга від 3.3 до 5 В;
- компаратор напруги: LM393;
- кріпильні отвори діаметром 3 мм;
- розміри модуля: 3.2 x 1.4 см;

Опис виводів датчика:

- VCC: 3.3 - 5В підключення + харчування
- GND: загальна земля;
- OUT: цифровий вихід;

Підключення цифрового датчика перешкод до МПС на основі Arduino UNO представлено на рис. 2.8.2.

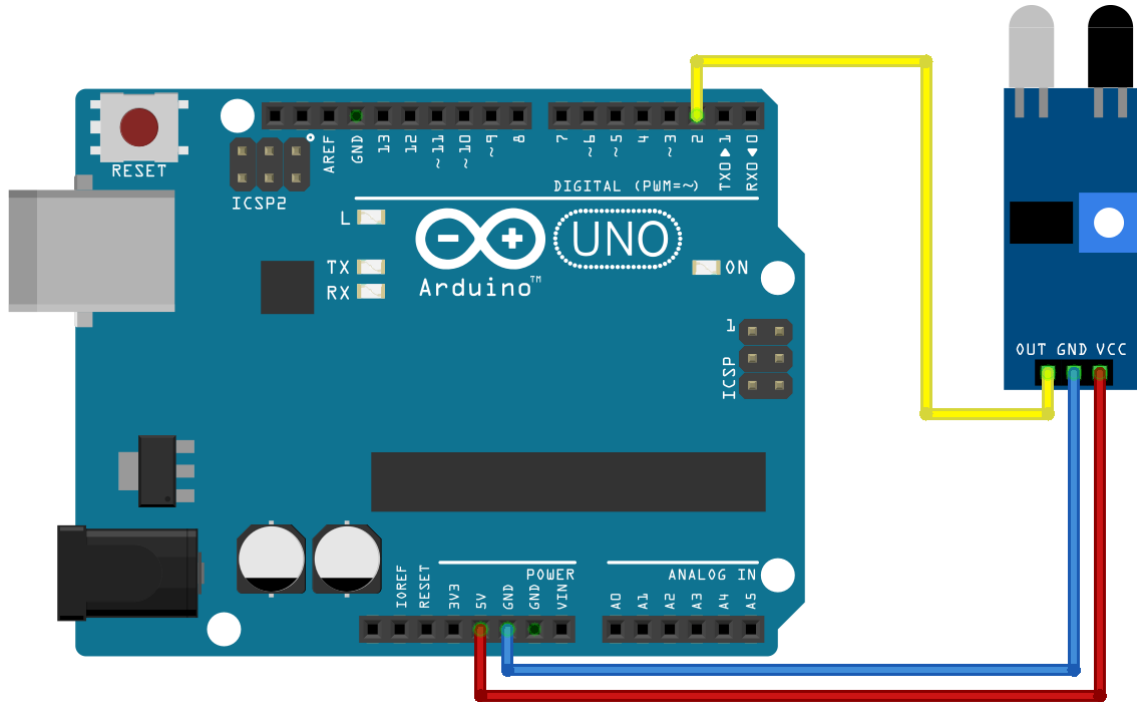


Рис. 2.8.2. Підключення цифрового датчика перешкод до МПС

Цифровий датчик для визначення перешкод застосовується у роботобудуванні, що обумовлюється малими розмірами сенсора.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.7.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.7.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.7.5.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування

Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.7.3 і рис. 2.7.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define echoPin 4 // Echo Pin
#define trigPin 5 // Trigger Pin
#define ledPin 2 // Hindrance Pin
int maximumRange = 200; // Maximum range needed
int minimumRange = 0; // Minimum range needed
long duration, distance; // Duration used to calculate distance
long duration_duo, distance_duo; // Duration used to calculate hindrance
long ReadSensor(int setPin); // User function
long ReadHindrance(int setPin); // User function
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
void setup() {
  Serial.begin (9600);
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT); // Port for distance
  pinMode(ledPin, INPUT); // Port for hindrance
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
```



```

void loop() {
    /* The following trigPin/echoPin cycle is used to determine the
    distance of the nearest object by bouncing sound waves off of it. */
    //Calculate the hindrance based on the speed of sound:
    duration_duo = ReadHindrance(ledPin); // User function for distance
    distance_duo = duration_duo;
    // Turn Ultrasonic Sensor:
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    //Calculate the distance (in cm) based on the speed of sound:
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    duration = ReadSensor(echoPin); // User function for hindrance
    distance = duration/58.2;
    if (distance >= maximumRange || distance <= minimumRange){
        /* Send a negative number to computer and Turn LED ON
        to indicate "out of range" */
        Serial.println("-1");
        digitalWrite(trigPin, HIGH);
    }
    else {
        /* Send the distance to the computer using Serial protocol, and
        turn LED OFF to indicate successful reading. */
        Serial.println("Distance = ");
        Serial.println(distance);
        digitalWrite(trigPin, LOW);
        // The hindrance based on the speed of sound:
        Serial.println(" Hindrance= ");
        Serial.println(distance_duo);
    }
}

```

```

//Delay 50ms before next reading.
delay(500);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

long ReadSensor(int setPin) {
    return pulseIn(setPin, HIGH); // User function for distance
}
long ReadHindrance(int setPin) {
    return digitalRead(setPin); // User function for hindrance
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Які змінні входять до формули розрахунку відстані до перешкоди?

2.9. Цифровий датчик Хола KY-024

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком Хола KY-024 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Магнітний датчик Хола KY-024 є лінійним датчиком з цифровим інтерфейсом. Датчик має і цифровий та аналоговий виходи. Використовується для визначення магнітного поля поруч з датчиком (рис. 2.9.1).

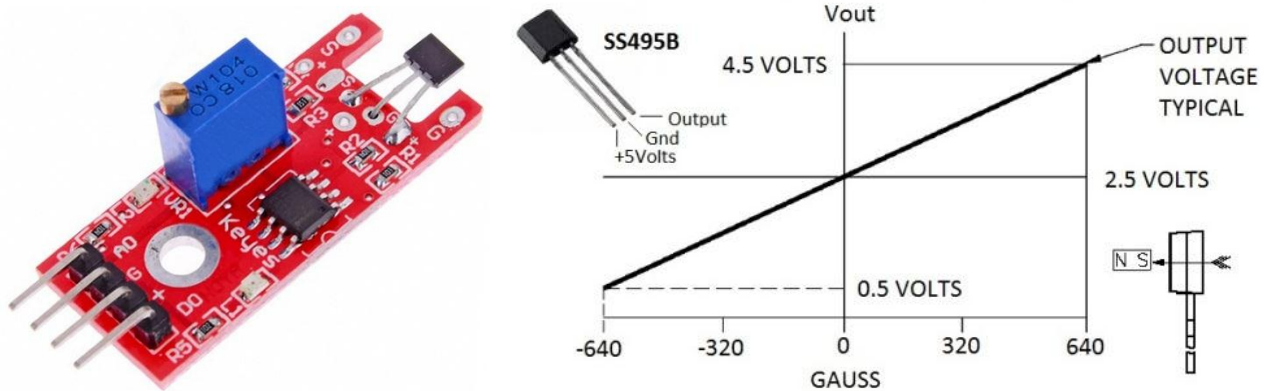


Рис. 2.9.1. Магнітний датчик Хола KY-024

Датчик Хола KY-024 використовується для виявлення магнітного поля, може працювати в якості кінцевого вимикача, датчика швидкості обертання, енодера і ін. На відміну від герконовий датчиків не має рухомих частин, тому є набагато довговічним.

Характеристики магнітного датчика:

- тип: датчик Хола (SS49E);
- подвійний компаратор: LM393;
- робоча напруга: DC 5В;
- чутливість: регульована потенціометром;
- розмір: 32x11x20 мм.

Виводи магнітного лінійного датчика Хола:

- A0: вихідна напруга в реальному часі (аналоговий вихід);
- D0: цифровий вихід, порогове напруга регулюється потенціометром;
- G: GND (загальний);
- живлення: "+" 3,3 - 5 В.

Підключення датчика Хола KY-024 до МПС здійснюється аналогічно підключенню датчика швидкості на мікросхемі LM393 до МПС (рис. 2.4.2).

На платі KY-024 магнітного датчика є світлодіод, який світиться при детектуванні магнітного поля.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.9.4.

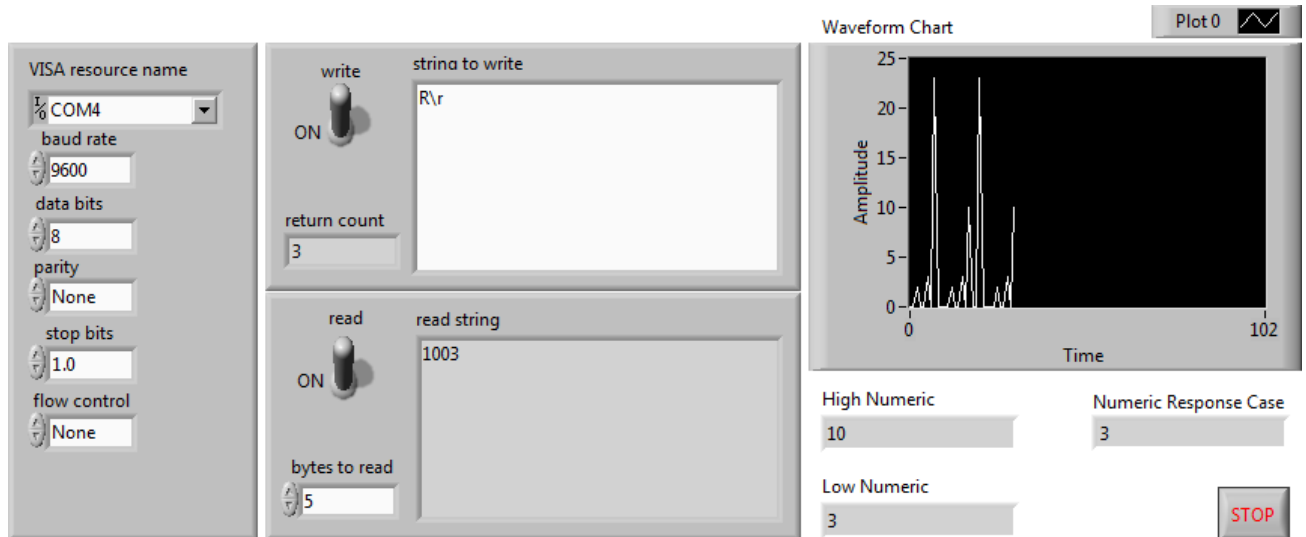


Рис. 2.9.4. Зовнішній вигляд системи керування МПС

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.9.2 і рис. 2.9.3. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

// Ініціалізація даних і змінних

```
#define Led 5 // define LED Interface
#define SENSOR A5 // define the Hall magnetic sensor interface;
int analog_val ; // define numeric variables;
int digital_val ; // define numeric variables;
int readSensor(int valPin); // user function read;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

// Налаштування мікроконтролера

```
void setup () {
  Serial.begin(9600);
  pinMode (Led, INPUT) ; // define LED as output interface;
  pinMode (SENSOR, INPUT) ; // define the Hall magnetic sensor line as input;
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

// Реалізація управління МПС

```
void loop() {
  analog_val = readSensor(SENSOR); // user function
  digital_val = digitalRead (Led); // read sensor line
  if (digital_val == HIGH) // when the Hall sensor detects a magnetic field
  {
    Serial.println("Hall sensor detects:");
    Serial.println(analog_val);
    digitalWrite (Led, HIGH);
  }
  if (analog_val < 526) // level of detection a magnetic field
  {
    Serial.println("Not sensor detects");
    digitalWrite (Led, LOW);
  }
  delay(1000);
}
```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```
int readSensor(int valPin) {
    return analogRead(valPin); // read sensor line
}
```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика Хола KY-024.

2.10. Датчик нахилу на LM393

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком нахилу на мікросхемі LM393 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Цифровий датчик нахилу з компаратором LM393 призначений для використання в автомобільних системах сигналізації та і системах розумний будинок (рис. 2.10.1). Поріг спрацьовування сенсору регулюється резистором налаштування на платі.

Характеристики датчик нахилу:

- компаратор: LM393;
- напруга живлення: 3.3 - 5 В;
- індикатор виходу: світлодіодний;
- вихідний сигнал: цифровий з порогом чутливості;
- чутливість: регульована (налаштовується резистором);
- розміри плати: 30 x 15 мм.

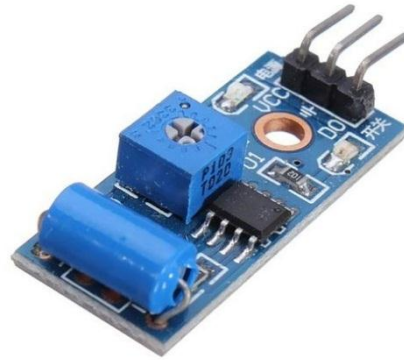


Рис. 2.10.1. Датчик нахилу на мікросхемі LM393

Підключення цифрового датчика нахилу на мікросхемі LM393 до МПС на основі контролера Arduino UNO представлено на рис. 2.10.2.

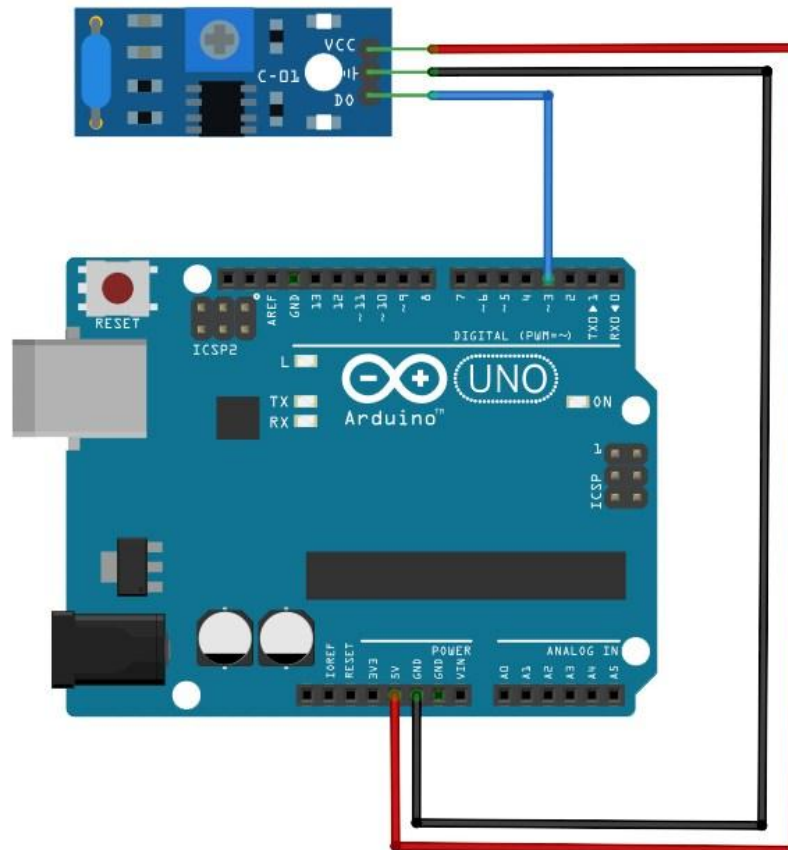


Рис. 2.10.2. Підключення цифрового датчика нахилу на мікросхемі LM393 до МПС

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.9.2, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.9.3, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.9.4.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.9.1 і рис. 2.9.2. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define Led 5           // define LED Interface
#define SENSOR 13      // define the Slope sensor interface;
int sensor_val;       // define numeric variables;
int signal_val;       // define numeric variables;
int readSensor(int valPin); // user function read;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

// Налаштування мікроконтролера

```
void setup () {
  Serial.begin(9600);
  pinMode (Led, INPUT);      // define LED as output interface;
  pinMode (SENSOR, INPUT);  // define the Slope sensor line as input;
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

// Реалізація управління МПС

```
void loop() {
  sensor_val = readSensor(SENSOR); // user function
  signal_val = digitalRead (Led);   // read sensor line
  if (signal_val == HIGH)           // when the Slope sensor detects an angle
  {
    Serial.println("Slope sensor detects:");
    Serial.println(sensor_val);
    digitalWrite (Led, HIGH);
  }
  if (signal_val < 0) // level of detection an angle
  {
    Serial.println("Not sensor detects");
    digitalWrite (Led, LOW);
  }
  delay(1000);
}
```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```
int readSensor(int valPin) {
  int data;
  data = digitalRead(valPin); // read sensor line
  if (data > 0) // level of detection a slope
  {
```

```

        digitalWrite (Led, HIGH);
    } else {
        digitalWrite (Led, LOW);
    }
    return data;
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика нахилу з компаратором LM393.

2.11. Датчик алкоголю MQ-3

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком алкоголю MQ-3 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Датчик алкоголю MQ-3 призначений для реєстрації та вимірювання концентрації парів алкоголю (2.11.1). Модуль побудований на базі газоаналізатора MQ-3, який дозволяє виявляти наявність парів спирту: від парфумерії або спиртних напоїв, в повітрі або диханні.

Характеристики алкоголю MQ-3:

- напруга живлення: 5 В;
- струм: 150 мА;
- діапазон вимірювання: 0,05 мг/л - 10 мг/л;
- чутливість: налаштовується резистором (тримером).



Рис. 2.11.1. Цифровий датчик алкоголю MQ-3

В газоаналізатор вбудований нагрівальний елемент, який необхідний для хімічної реакції. Тому під час роботи сенсор буде гарячим, це нормально. Для отримання стабільних показань новий сенсор необхідно один раз прогріти (залишити включеним) протягом 24 годин. Після цього стабілізація характеристик після включення газоаналізатора MQ-3 буде займати близько хвилини.

Підключення цифрового датчика алкоголю MQ-3 до МПС на основі Arduino UNO представлено на рис. 2.11.2

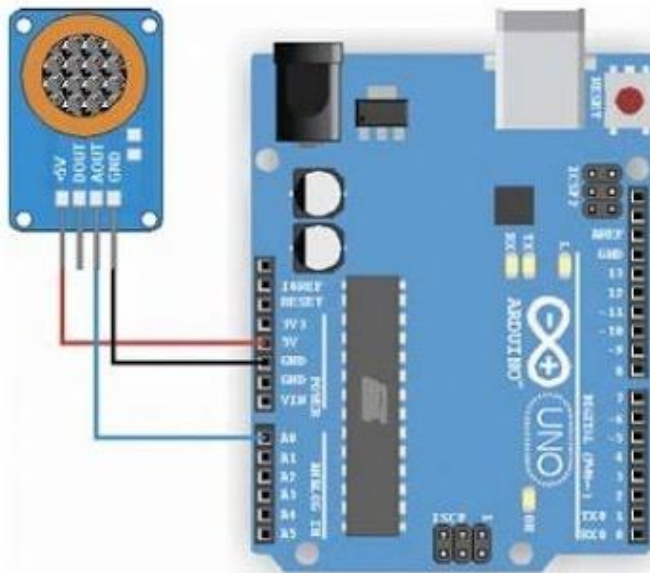


Рис. 2.11.2. Підключення цифрового датчика алкоголю MQ-3 до МПС

Вихідним результатом є аналоговий сигнал, пропорційний концентрації парів спирту навколо газоаналізатора. Чутливість датчика алкоголю MQ-3 може бути налаштована за допомогою тримеру на платі датчика.

Показання сенсора залежать від впливу температури і вологості навколишнього повітря. Тому в разі використання датчика газу в мінливому середовищі, для отримання точних показань, необхідно реалізувати компенсацію цих параметрів.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку Numeric = 1 у структурі «Case Structure» представлена на рис 2.11.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку Numeric = 0 у структурі «Case Structure» представлена на рис 2.11.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

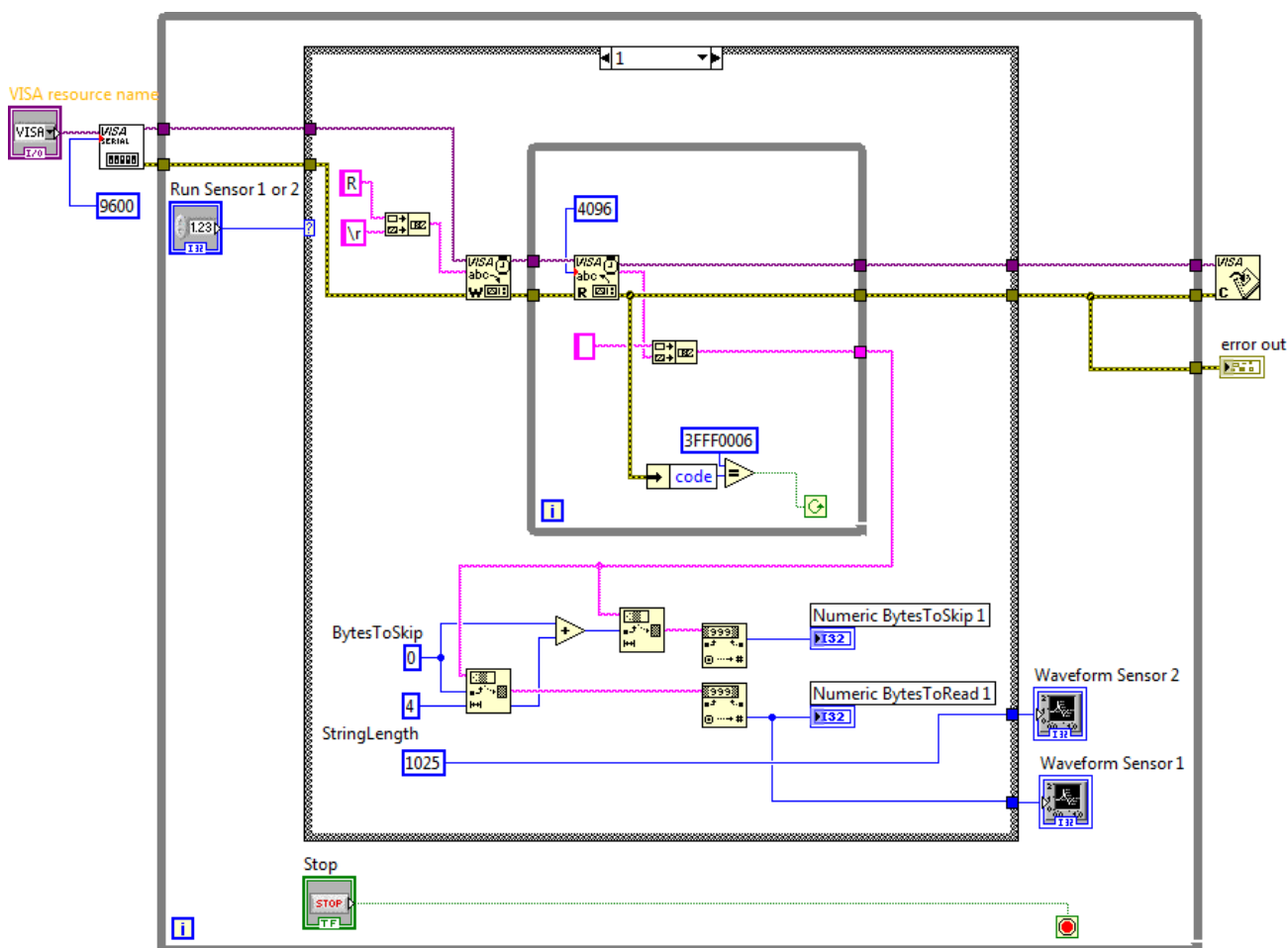


Рис. 2.11.3. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 1

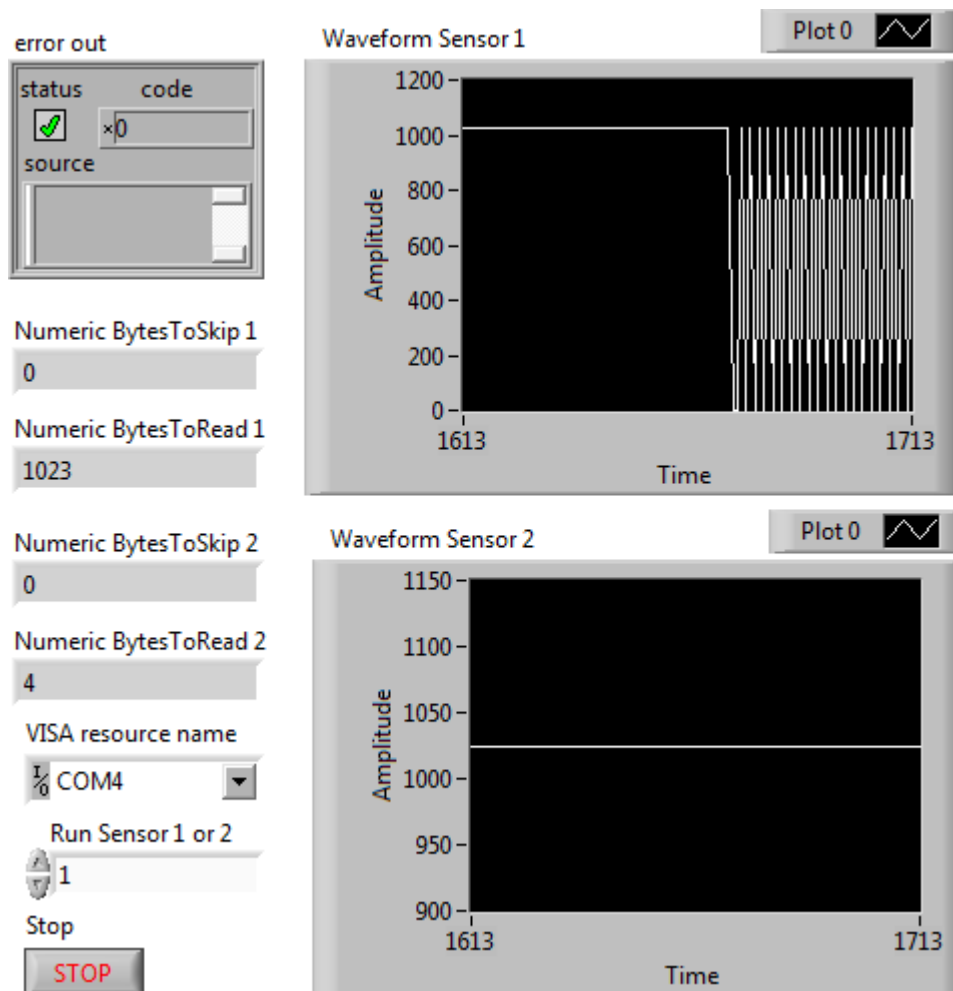


Рис. 2.11.5 Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.11.1 і рис. 2.11.2. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define ctsPin A0 // Підключення датчика алкоголю
#define levPin 13 // Підключення LED
```

```
#define Nulled 0 // Default Value
int count = 0;
int oldStat = 0;
int keyValue;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
void setup() {
  Serial.begin(9600);
  pinMode(levPin, OUTPUT);
  pinMode(ctsPin, INPUT);
  digitalWrite(levPin, HIGH);
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
void loop() {
  count=Serial.read();
  if (count<0) {
    delay(100);
    count=oldStat;
  }
  if(count=='R') {
    keyValue = SetDetectionA0();
    if (keyValue != 0){
      Serial.println(keyValue);
      digitalWrite(levPin, HIGH);
    }
    delay(100);
    count = 'R';
  }
}
```



```

    if (count=='S') {
        digitalWrite(levPin, LOW);
        delay (100);
        count='S';
    }
    delay(500);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

int SetDetectionA0()
{
    int data;
    data = analogRead(ctsPin); // Читання датчика алкоголю
    return data;
}

```

Завантажити програмне забезпечення для мікроконтролеру Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролеру Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика парів спирту MQ-3.

2.12. Цифровий датчик рівня рідини

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління цифровим датчиком рівня рідини у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Датчик рівня рідини (Water Sensor) призначений для вимірювання рівня поверхнево неактивних речовин (рис. 2.12.1). Це простий у використанні і недорогий датчик рівня рідини широко застосовується в системах автоматизації і в проектах розумного будинку.



Рис. 2.12.1. Цифровий датчик рівня рідини

Технічні характеристики рівня рідини:

- робоча напруга: DC3-5V;
- робочий струм: менше 20 мА;
- тип датчика: аналоговий;
- площа виявлення: 40 x 16 мм;
- виробничий процес: FR4 двостороння HASL;
- робоча температура: 10 - 30°C;
- вологість: 10% - 90% без конденсації;
- вага датчику: 3,5 г;
- розмір: 62 x 20 x 8 мм.

Підключення цифровий датчик рівня рідини до МПС на основі контролера Arduino UNO представлено на рис. 2.12.2

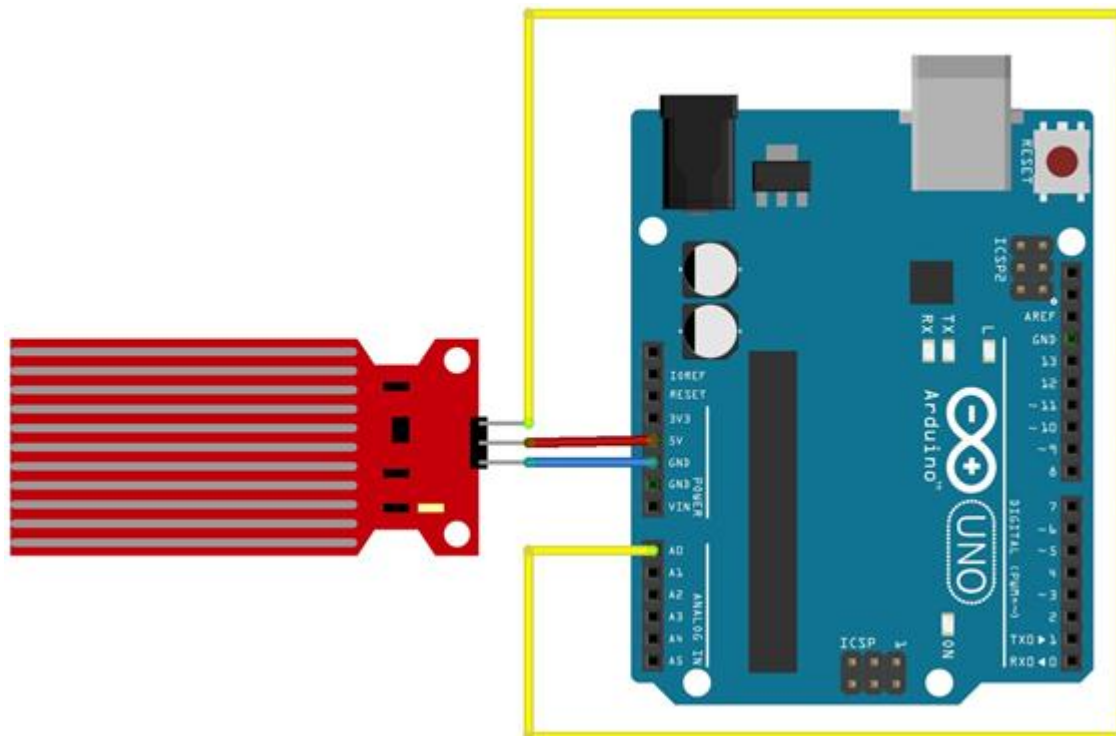


Рис. 2.12.2 Підключення цифрового датчика рівня рідини до МПС

Для вимірювання рівня рідини використовується струмопровідний метод. Датчик в зборі складається з алюмінієвих провідників, що нанесені на текстоліт. Вихідний сигнал відповідає рівням квантування, що калібрують відповідно до рівнів заповнення резервуару.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку Numeric = 2 у структурі «Case Structure» представлена на рис 2.12.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку Numeric = 0 у структурі «Case Structure» представлена на рис 2.12.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.12.5.

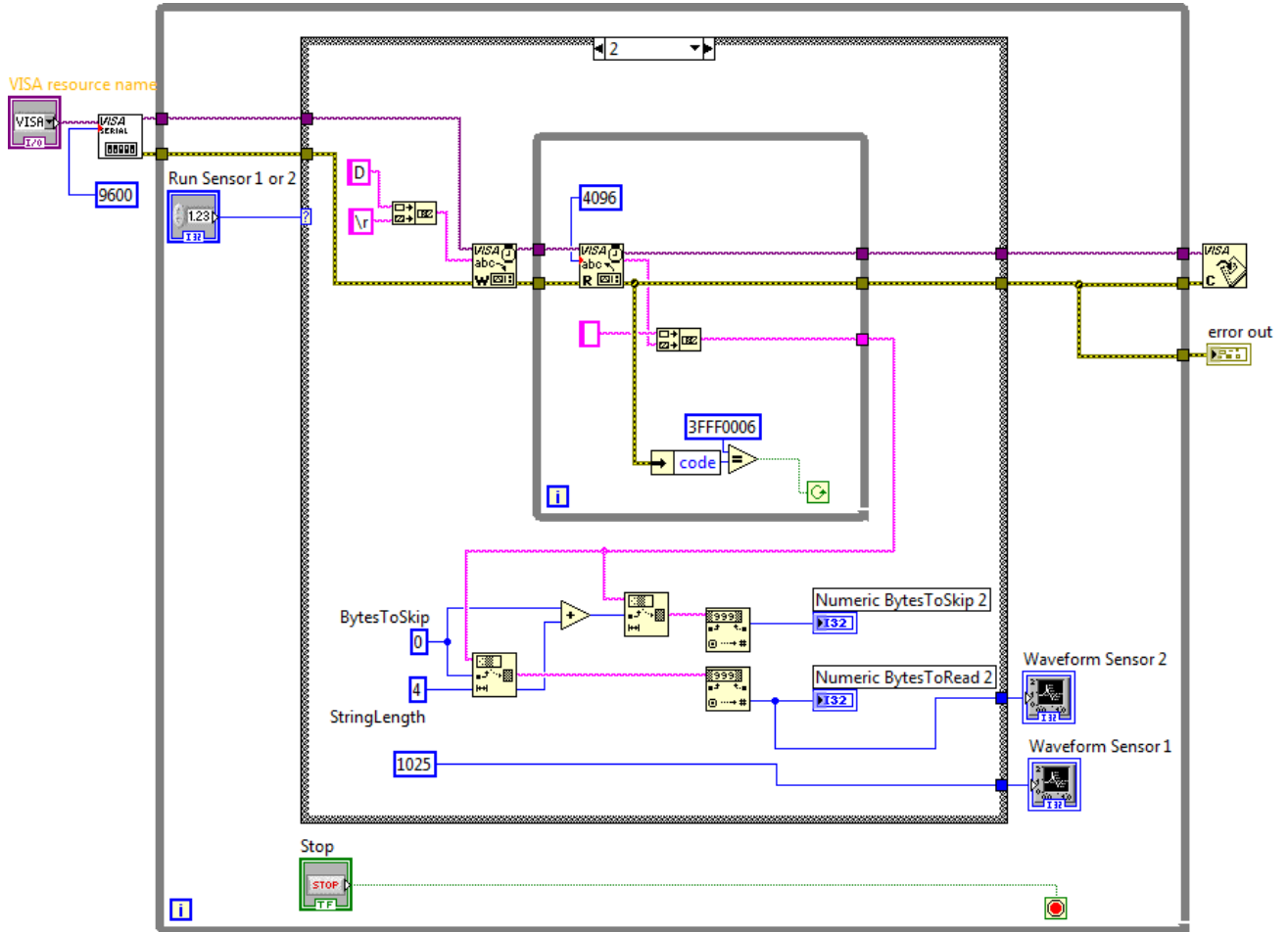


Рис. 2.12.3. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 2

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

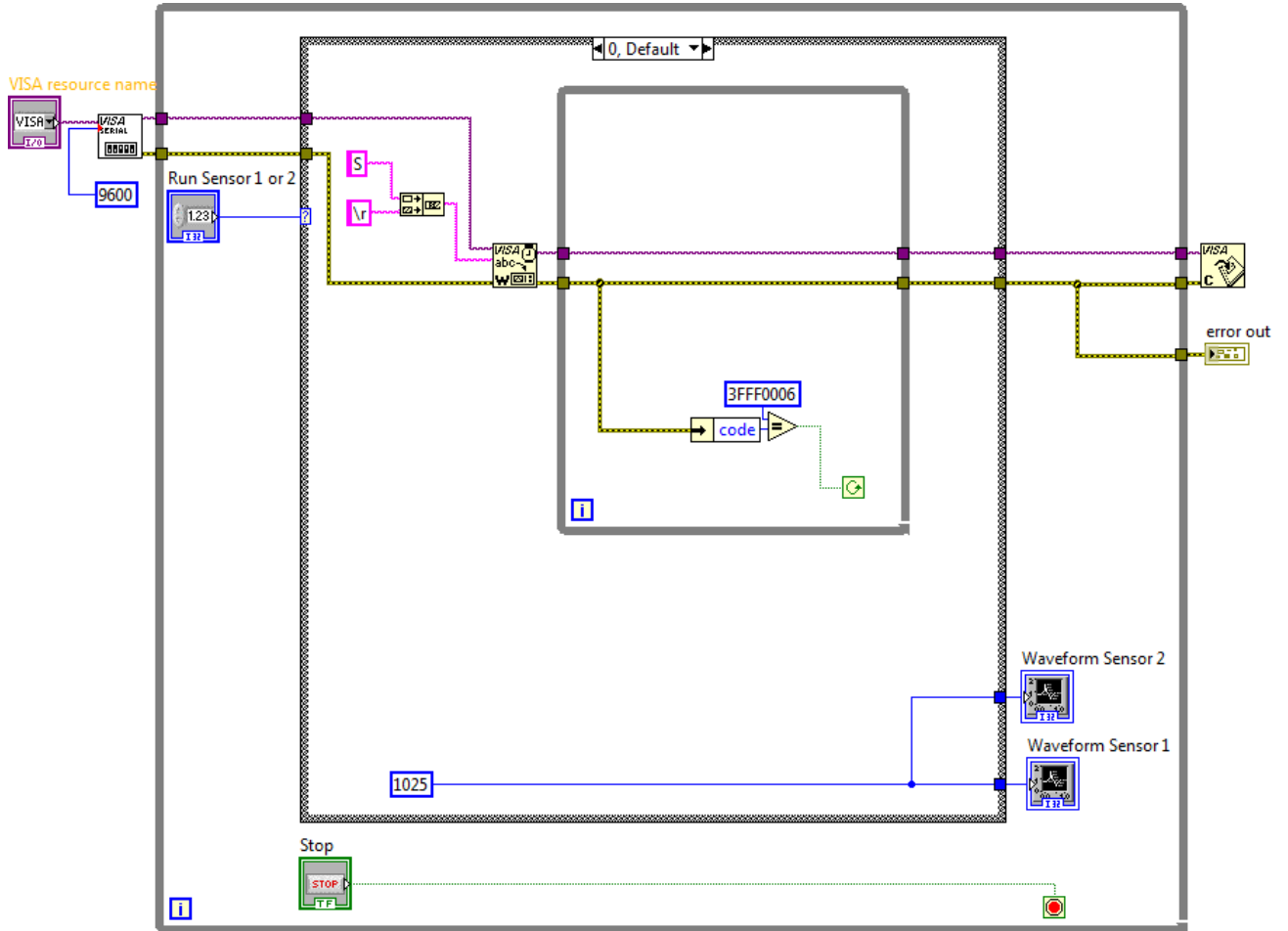


Рис. 2.12.4. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 0

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

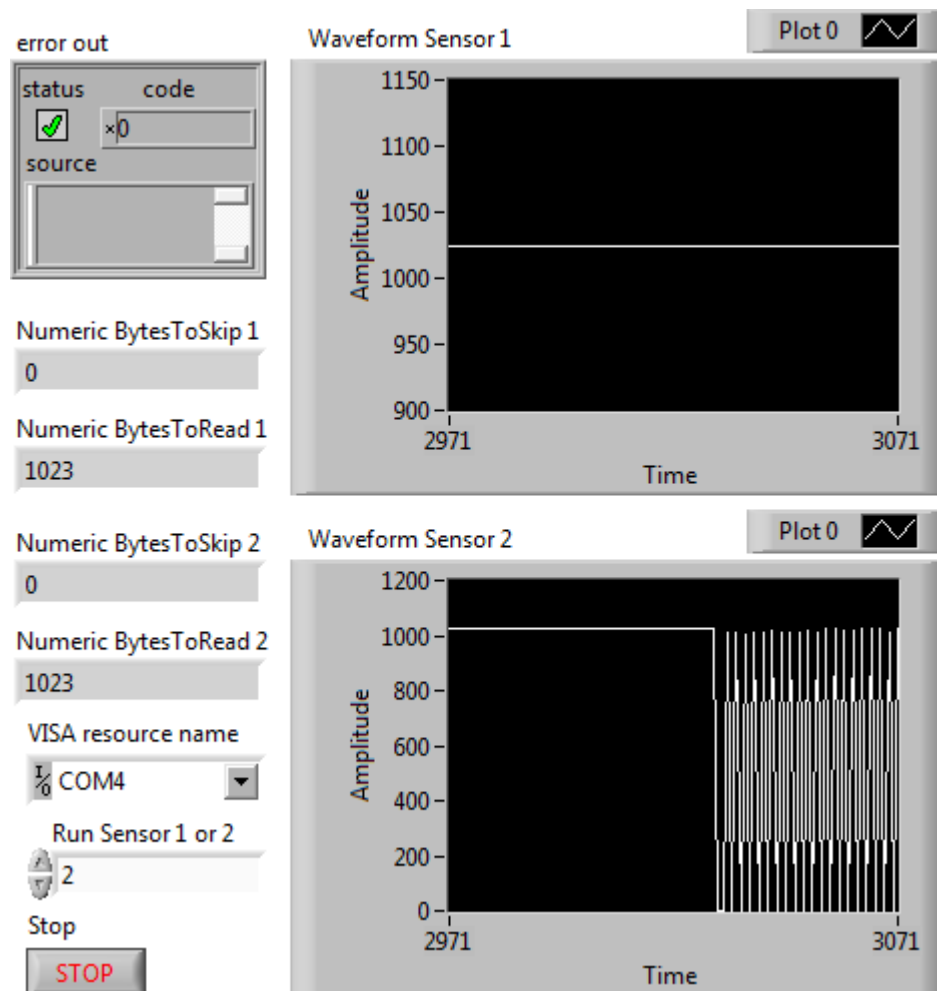


Рис. 2.12.5. Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.12.3 і рис. 2.12.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
```

```
#define ctsPin A0 // Підключення датчика алкоголю
```

```
#define keyPin A5 // Підключення датчика рівня рідини
```

```

#define levPin 13 // Підключення LED
#define Nulled 0 // Default Value

int count = 0;
int oldStat = 0;
int keyValue;
int ctsValue;

```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

// Налаштування мікроконтролера

```

void setup() {
    Serial.begin(9600);
    pinMode(levPin, OUTPUT);
    pinMode(ctsPin, INPUT);
    pinMode(keyPin, INPUT);
    digitalWrite(levPin, HIGH);
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

// Реалізація управління МПС

```

void loop() {
    count=Serial.read();
    if (count<0) {
        delay(100);
        count=oldStat;
    }
    if(count=='R') {
        keyValue = SetDetectionA0();
        if (keyValue != 0) {
            Serial.println(keyValue);
            digitalWrite(levPin, HIGH);
        }
    }
}

```

```

        delay(100);
        count = 'R';
    }
    if(count=='D') {
        ctsValue = SetDetectionA5();
        if (ctsValue != 0) {
            Serial.println(ctsValue);
            digitalWrite(levPin, HIGH);
        }
        delay(100);
        count = 'D';
    }
    if (count=='S') {
        digitalWrite(levPin, LOW);
        delay (100);
        count='S';
    }
    delay(500);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

int SetDetectionA0()
{
    int data;
    data = analogRead(ctsPin); // Читання датчика алкоголю

    return data;
}
int SetDetectionA5()
{
    int data;
    data = analogRead(keyPin); // Читання датчика рівня рідини
    return data;
}

```


Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика рівня рідини (Water Sensor).

2.13. Цифровий датчик вогню

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління цифровим датчиком вогню у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Цифровий датчик вогню дозволяє виявляти полум'я або джерело світла з довжиною хвилі 760 - 1100 нм. Датчик впевнено виявляє полум'я на відстані до 80 см (рис. 2.13.1).

Кут спрацьовування сенсору до 60 градусів. Вихід у датчика цифровий, 0 або 1. Модуль побудовано на компараторі LM393. Для надійного кріплення модуля на платі передбачено кріпильний отвір.



Рис. 2.13.1. Цифровий датчик вогню

Характеристики датчика вогню:

- робоча напруга 3.3 - 5 В;
- довжина хвилі: 760 - 1100 нм;
- відстані до 80 см;
- кут спрацьовування: до 60 градусів;
- чутливість датчика регулюється налаштуванням потенціометра;
- розміри: 47.2 x 14 x 7.3 мм;
- вага: 2.1 гр.

Підключення цифрового датчика вогню до МПС на основі контролера Arduino UNO здійснюється аналогічно підключенню цифрового датчика перешкод до МПС (рис. 2.8.2).

Використаний датчик полум'я, має інвертований вихід, а значить, він буде повертати значення False, якщо в межах його видимості є полум'я, і істину (True) – у разі відсутності відкритого полум'я.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.13.2, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.13.3, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролера Arduino UNO представлений на рис. 2.13.4.

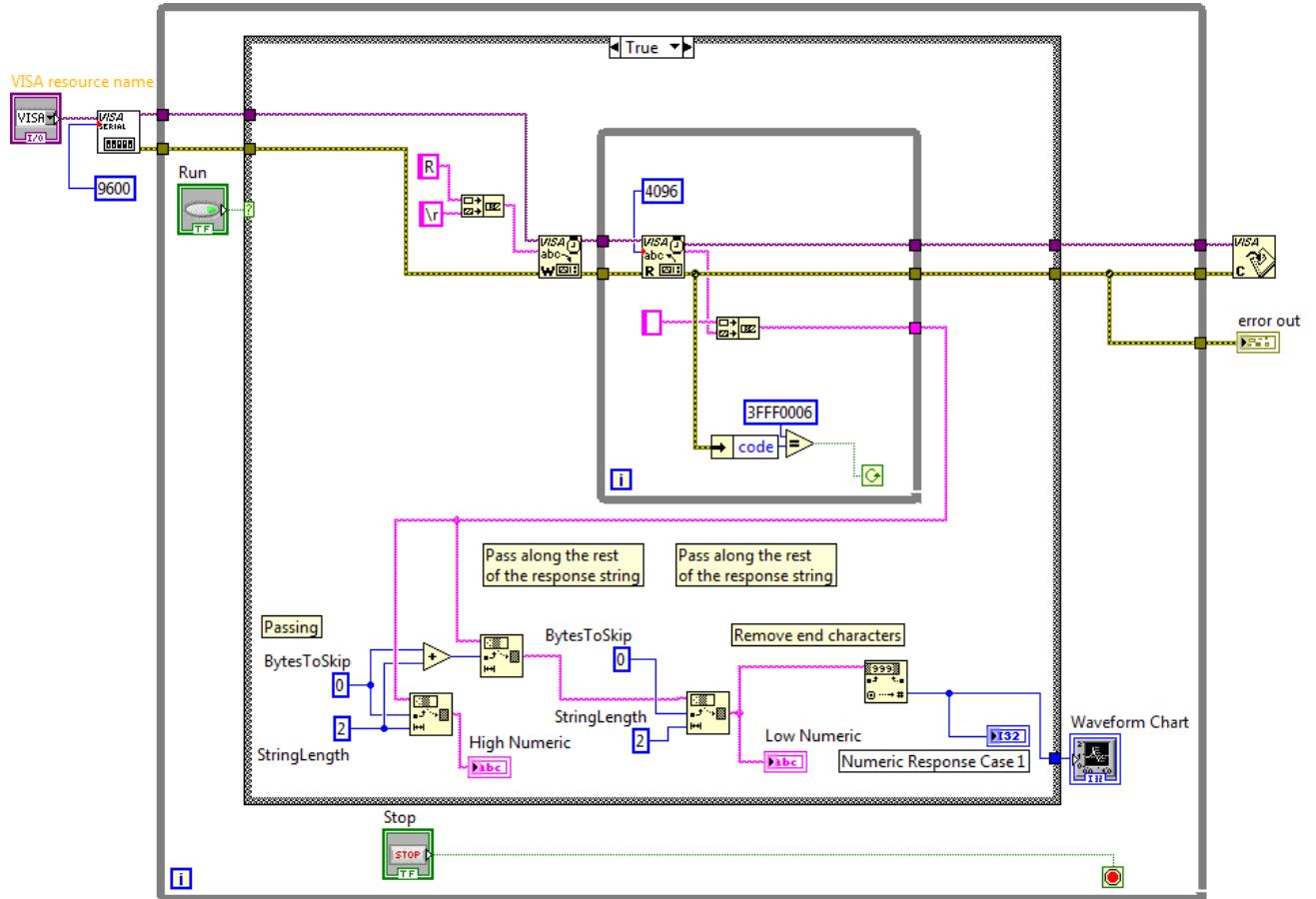


Рис. 2.13.2. Схема приладу в NI LabVIEW 2010 для випадку True

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

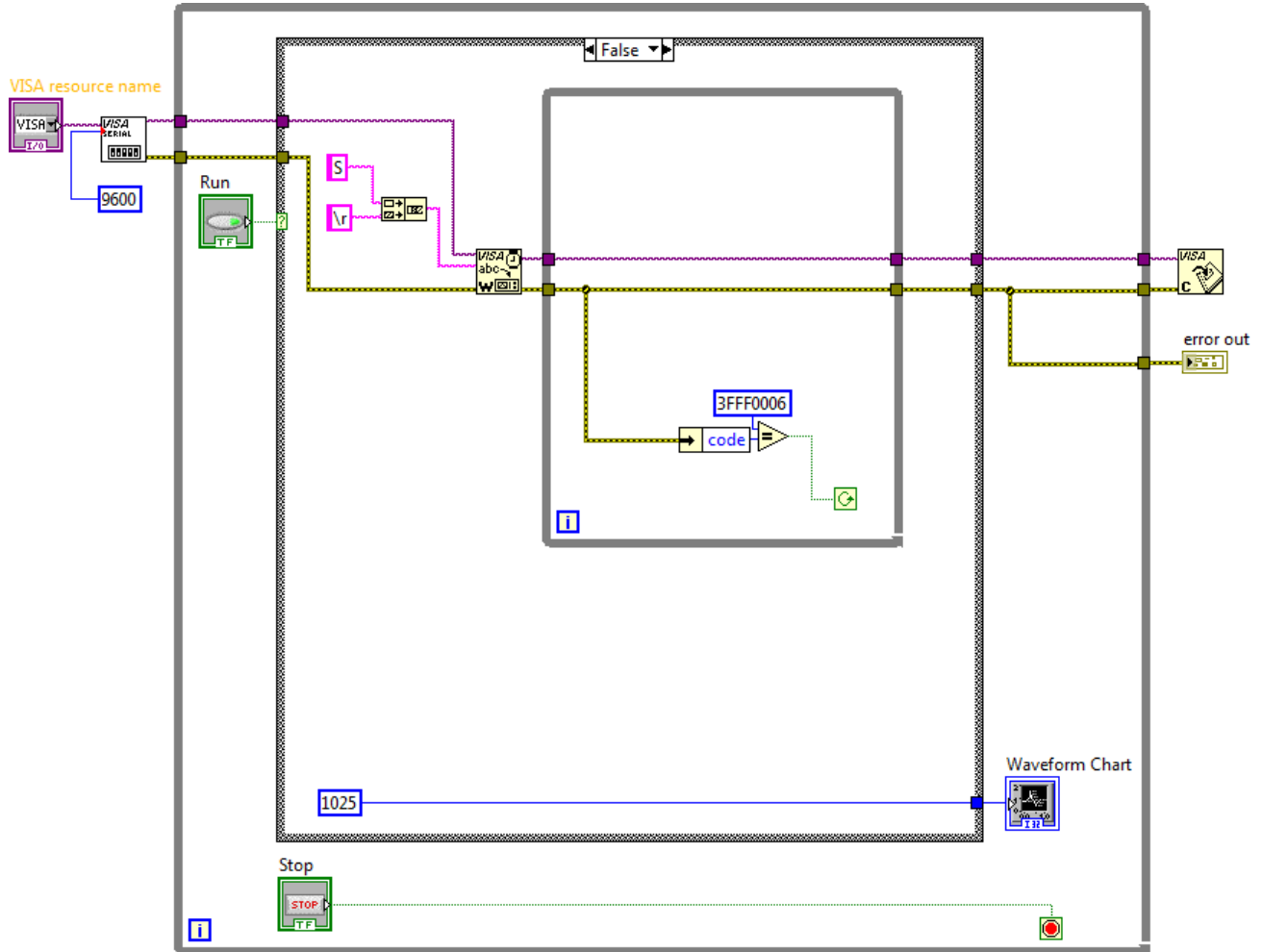


Рис. 2.13.3. Схема приладу в NI LabVIEW 2010 для випадку False

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

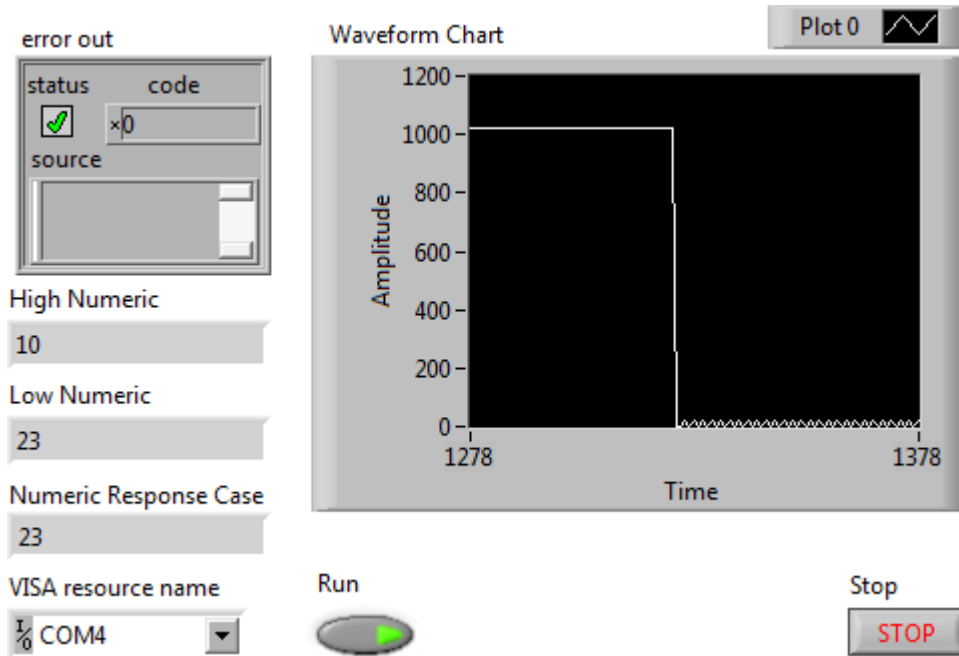


Рис. 2.13.4. Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.13.2 і рис. 2.13.3. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define Led 5           // define LED Interface
#define SENSOR 2       // define the Fire sensor interface;
int sensor_val;        // define numeric variables;
int signal_val;        // define numeric variables;
int readSensor(int valPin); // user function read;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
```

```

void setup () {
  Serial.begin(9600);
  pinMode (Led, INPUT);      // define LED as output interface;
  pinMode (SENSOR, INPUT);  // define the Fire sensor line as input;
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

// Реалізація управління МПЦ

```

void loop() {
  sensor_val = readSensor(SENSOR); // user function
  signal_val = digitalRead (Led);   // read sensor line
  if (signal_val == HIGH)           // when the Fire sensor detects light
  {
    Serial.println("Fire sensor detects:");
    Serial.println(sensor_val);
    digitalWrite (Led, HIGH);
  }
  if (signal_val < 0) // level of detection a light
  {
    Serial.println("Not sensor detects");
    digitalWrite (Led, LOW);
  }
  delay(1000);
};

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

int readSensor(int valPin) {
  int data;
  data = digitalRead(valPin); // read sensor line
  if (data > 0) // level of detection a light
  {
    digitalWrite (Led, HIGH);
  } else {

```

```

        digitalWrite (Led, LOW);
    }
    return data;
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика вогню (Fire sensor).

2.14. Аналогово-цифровий датчик освітленості

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління аналогово-цифровим датчиком освітленості у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Аналогово-цифровий датчик освітленості представляє компактний модуль до складу якого входять: датчика світла (фоторезистор), пороговий компаратор LM393, аналогово-цифровий перетворювач, тример регулювання порога спрацьовування (рис. 2.14.1).

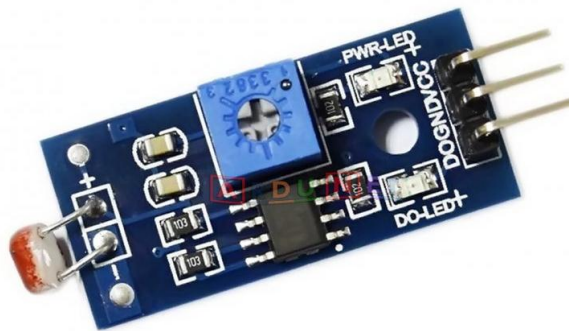


Рис. 2.14.1. Аналогово-цифровий датчик освітленості

Характеристики датчика освітленості:

- чутливий елемент: фоторезистор;
- вихід компаратора: більш 15 мА;
- робоча напруга: від 3.3 В до 5В;
- компаратор LM393;
- регулювання порога спрацьовування: змінний резистор (тример);
- цифровий вихід компаратора (0 і 1);
- аналоговий вихід датчика освітленості;
- кріпильний отвір;
- розміри: 3.2 см x 1.4 см.

Призначення виводів датчика:

- VCC: вхід напруги живлення 3.3-5 В
- GND: загальна земля
- DO: цифровий вихід компаратора
- AO: аналоговий вихід світлочутливого датчика

Чутливий елемент – фоторезистор змінює свій опір залежно від інтенсивності падаючого на нього світла. Наприклад, у темряві опір фоторезистора VT83N1 дорівнює 12кОм, а при певній тестовому освітлені – 100кОм. Крім фоторезистора, в датчиках освітленості використовують фотодіод і фототранзистор.

Підключення аналого-цифрового датчика освітленості до МПС на основі контролеру Arduino UNO здійснюється аналогічно підключенню датчика перешкод до МПС (рис. 2.8.2).

Використовується датчик освітленості в системах автоматичного освітлення, детектування перешкоди в робототехнічних системах.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 2.13.2, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 2.13.3, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.13.4.

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.13.2 і рис. 2.13.3. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define Led 5 // define LED Interface
#define SENSOR A2 // define the Light sensor interface;
int analog_val ; // define numeric variables;
int digital_val ; // define numeric variables;
int readSensor(int valPin); // user function read;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролеру
void setup () {
  Serial.begin(9600);
  pinMode (Led, INPUT) ; // define LED as output interface;
  pinMode (SENSOR, INPUT) ; // define the Light sensor line as input;
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
void loop() {
    analog_val = readSensor(SENSOR); // user function
    digital_val = digitalRead (Led);    // read sensor line
    if (digital_val == HIGH)           // when the Light sensor detects a color
    {
        Serial.println("Light sensor detects:");
        Serial.println(analog_val);
        digitalWrite (Led, HIGH);
    }
    if (analog_val < 128) // level of detection a light
    {
        Serial.println("Not sensor detects");
        digitalWrite (Led, LOW);
    }
    delay(1000);
}
```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```
int readSensor(int valPin) {
    return analogRead(valPin); // read sensor line
}
```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?

3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи аналогово-цифрового датчика освітленості.

2.15. Датчик вібрації SW-420

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління датчиком вібрації SW-420 у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Датчик вібрації SW-420 реагує на вібрацію. Модуль використовується для детектування вібрацій в протиугінних системах, охоронних сигналізаціях і навіть для детектування землетрусів (рис. 2.15.1).

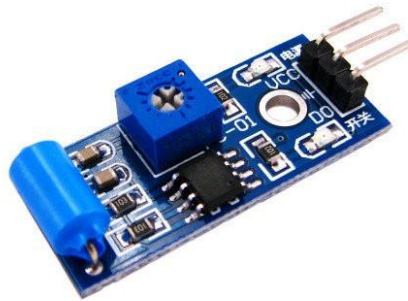


Рис. 2.15.1. Датчик вібрації SW-420

У корпусі модуля розташовані два світлодіоди. Червоний світлодіод підключений до виводу живлення і випромінює, коли на модуль подається напруга живлення. Зелений підключений до цифрового виходу і загорається, коли датчик спрацьовує. Чутливість датчика регулюється налаштуванням резистору, який розташованим на платі.

Характеристики датчика вібрації SW-420:

- напруга живлення: від 3.3 до 5В;
- вихід: цифровий;
- компаратор: LM393;

- кріплення: монтажний отвір;
- розміри плати: 3.2x1.4 см.

Підключення датчику вібрації SW-420 до МПС на основі Arduino UNO здійснюється аналогічно підключенню датчика нахилу на мікросхемі LM393 до МПС (рис. 2.10.2).

На цифровому виході D0 присутня логічна 1, якщо немає спрацювання і логічний 0, якщо датчик спрацював від вібрації.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку Numeric = 1 у структурі «Case Structure» представлена на рис 2.15.2, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку Numeric = 0 у структурі «Case Structure» представлена на рис 2.15.3, що відповідає зупинці МПС при натисненні кнопки «Stop».

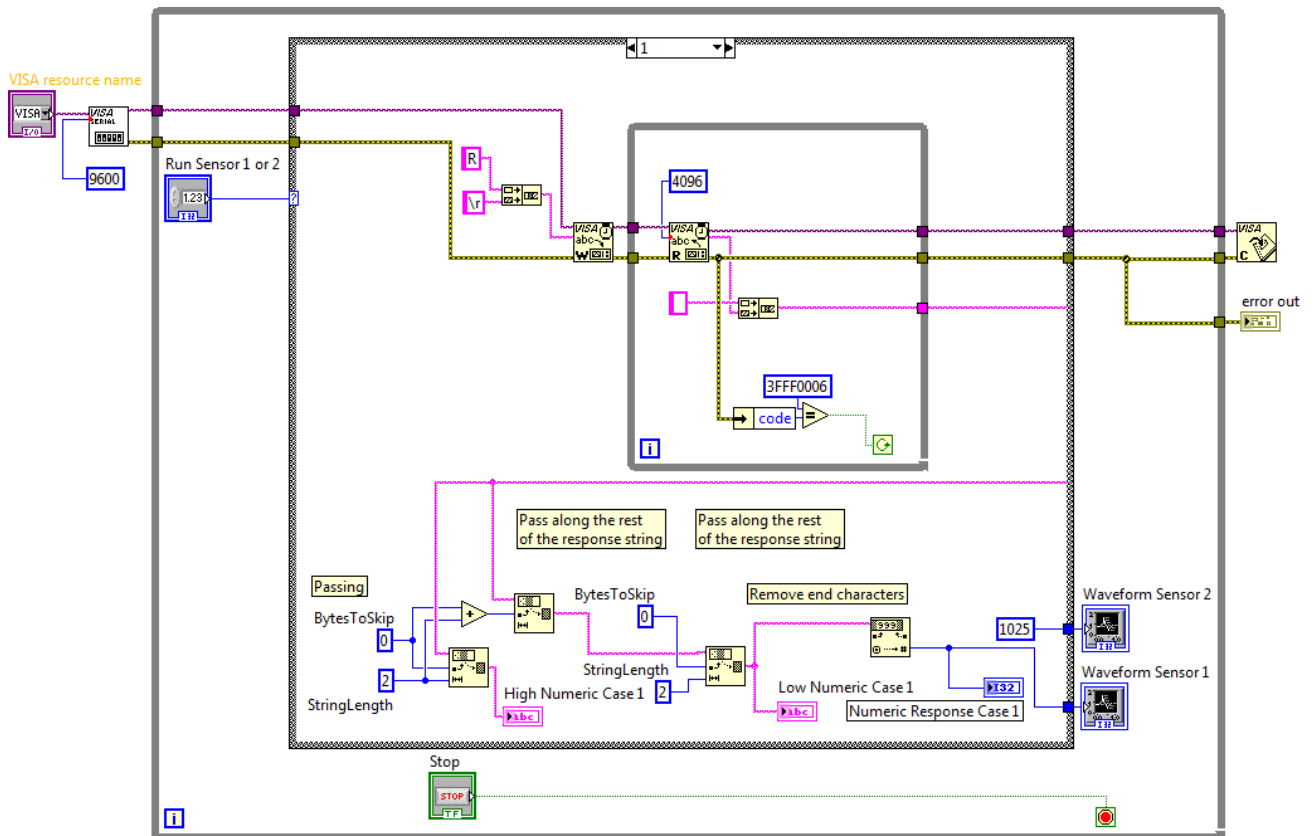


Рис. 2.15.2. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 1

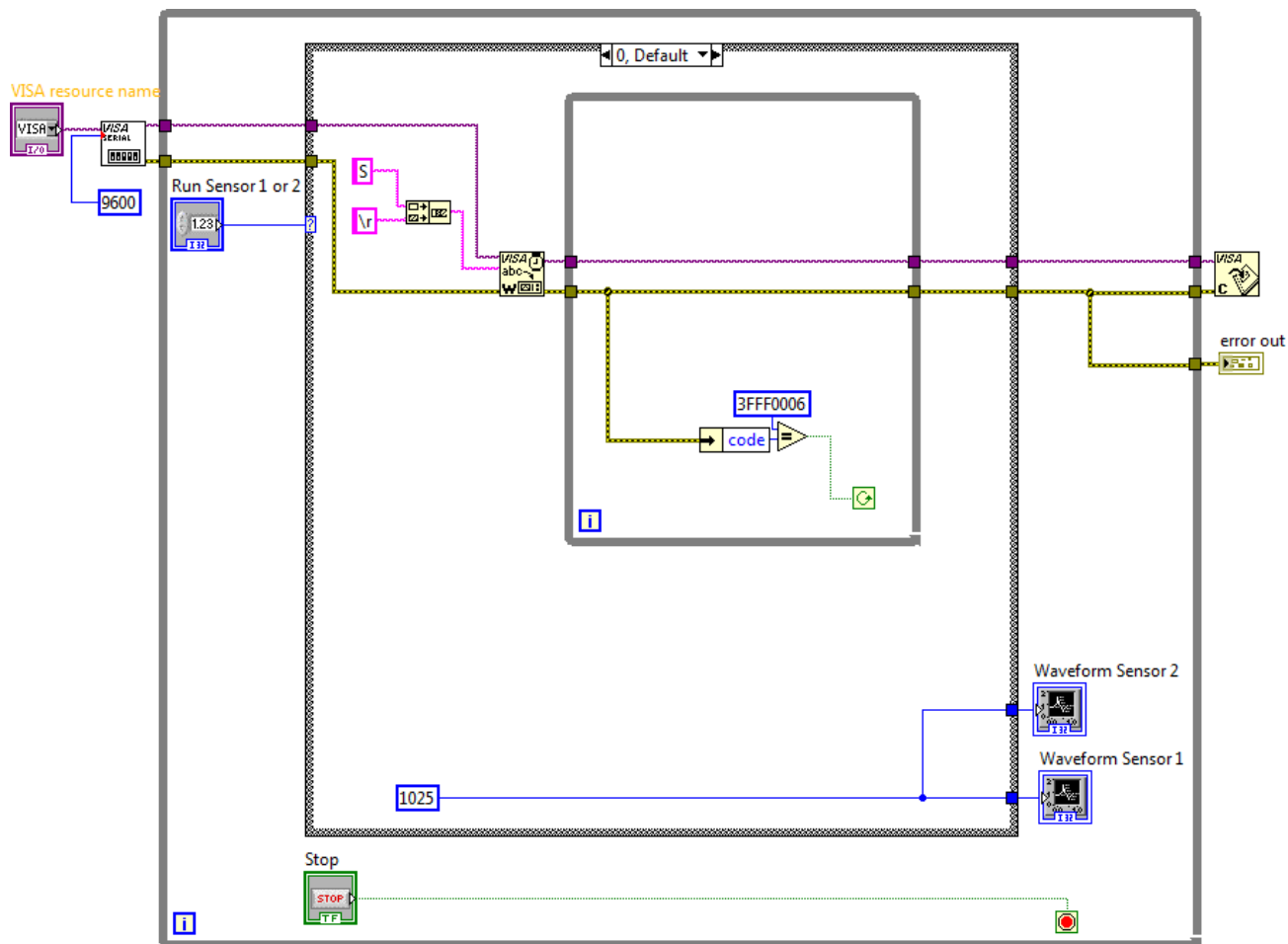


Рис. 2.15.3. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 0

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролера Arduino UNO представлений на рис. 2.15.4.

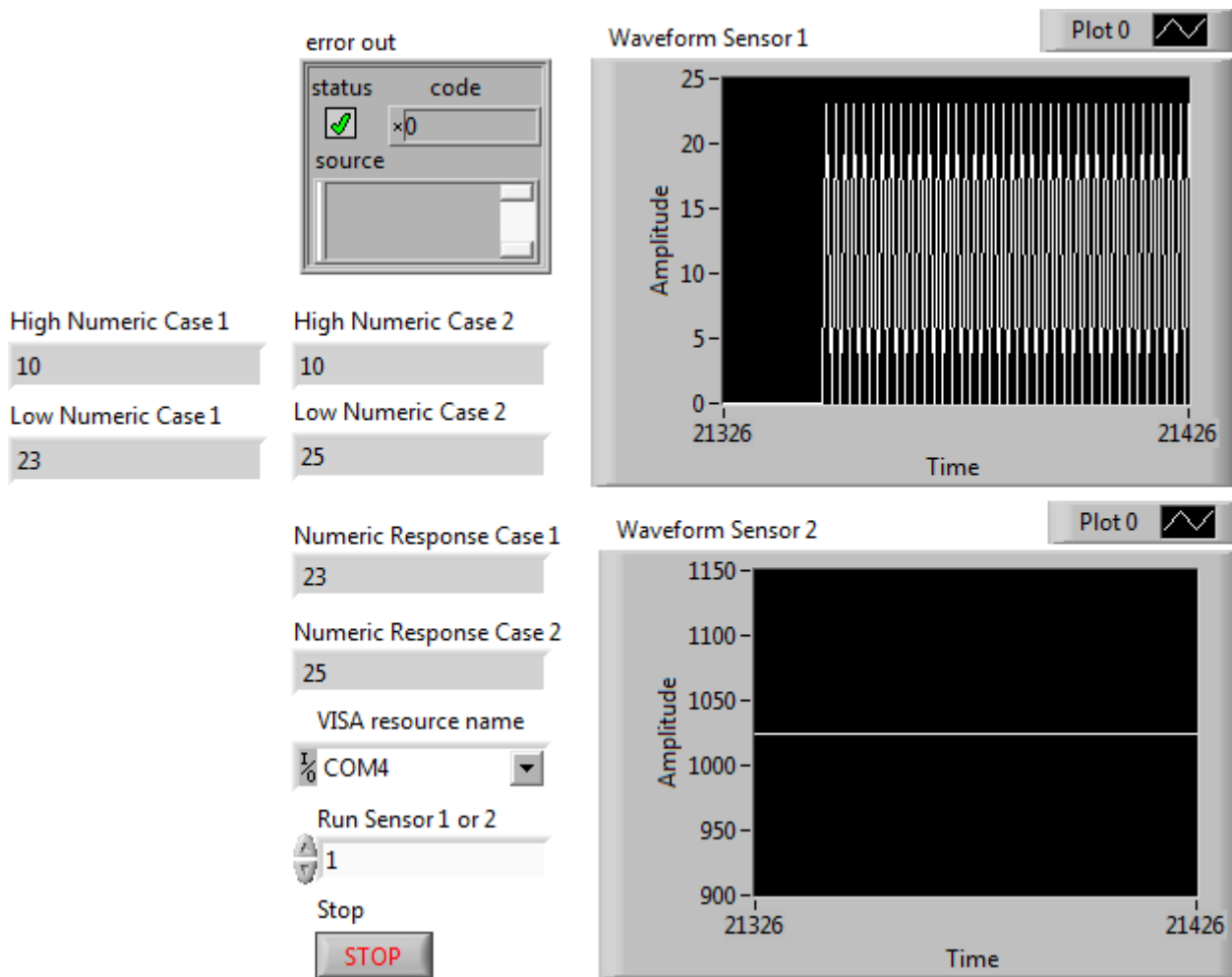


Рис. 2.15.4. Зовнішній вигляд системи керування МПС

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.15.2 і рис. 2.15.3. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define ctsPin 13 // Підключення вібраційного датчика
#define keyPin A0
#define ledPin 5 // Підключення світлодіоду
void vibcount(int count, int maxcount);
const int maxcount = 1024;
int keyState = 0;
int count = 0;
int oldStat = 0;
int ctsValue;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
void setup() {
  Serial.begin(9600);
  pinMode(ledPin, OUTPUT);
  pinMode(ctsPin, INPUT);
  pinMode(keyPin, INPUT);
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
void loop() {
  count=Serial.read();
  if (count<0) {
    delay(100);
```

```

        count=oldStat;
    }
    if(count=='R') {
        ctsValue = digitalRead(ctsPin);
        keyState = analogRead(keyPin);
        if (ctsValue == HIGH) {
            // Sig Output в high, сенсор увімкнувся
            digitalWrite(ledPin, HIGH);
            Serial.println("TOUCHED");
        } else{
            digitalWrite(ledPin,LOW);
            Serial.println("not touched");
        }
        if (keyState != oldStat) {
            count=0;
            oldStat = keyState;
        }
        discount(count, maxcount);
        delay(500);
        count = 'R';
    }
    if (count=='S') {
        delay (100);
        count='S';
    }
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

void vibcount(int count, int maxcount) {
    if (count < maxcount){
        digitalWrite(ledPin, HIGH);
    }
}

```



```

        Serial.println("level Count: ");
        Serial.println(keyState); // display
    }
    else {
        Serial.println("max Count: ");
        Serial.println(keyState); // display
        digitalWrite(ledPin, LOW);
    }
    if (count<=maxcount+1) {
        ++count;
    }
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика вібрації SW-420.

2.16. Цифровий датчик удару

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління цифровим датчиком удару у середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Цифровий датчик удару KY-031 реєструє удари по корпусу об'єкта в якому він змонтований (рис. 2.16.1). Встановлюється в охоронні системи автомобілів, мотоциклів, велосипедів. Модуль датчика удару застосовується в системах охоронної сигналізації розумного будинку.

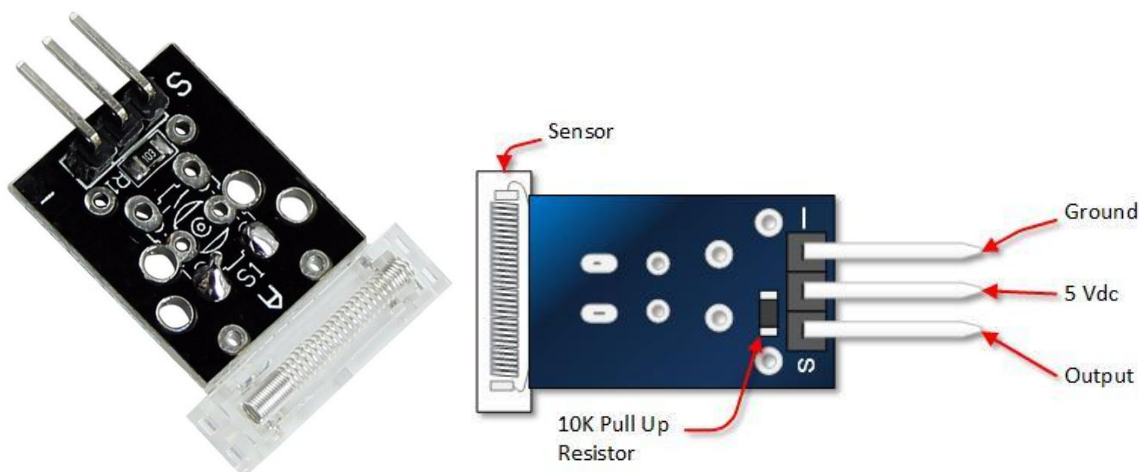


Рис. 2.16.1. Цифровий датчик удару

Модуль датчика найбільш чутливий до ударів, що спрямовані поперек площині плати. Вплив сприймає чутливий елемент, що представляє собою пружину, кінець якої оточений контактами. Між входом живлення і виходом датчика удару знаходиться резистор 10 кОм. При ударі пружина згинається, кінець пружини торкається контактів і ланцюг датчика KY-031 замикається. При спрацюванні датчика замикається контакт, який може бути з'єднаний з входом самих різних приладів.

Підключення цифрового датчика удару до МПС на основі Arduino UNO представлено на рис. 2.16.2.

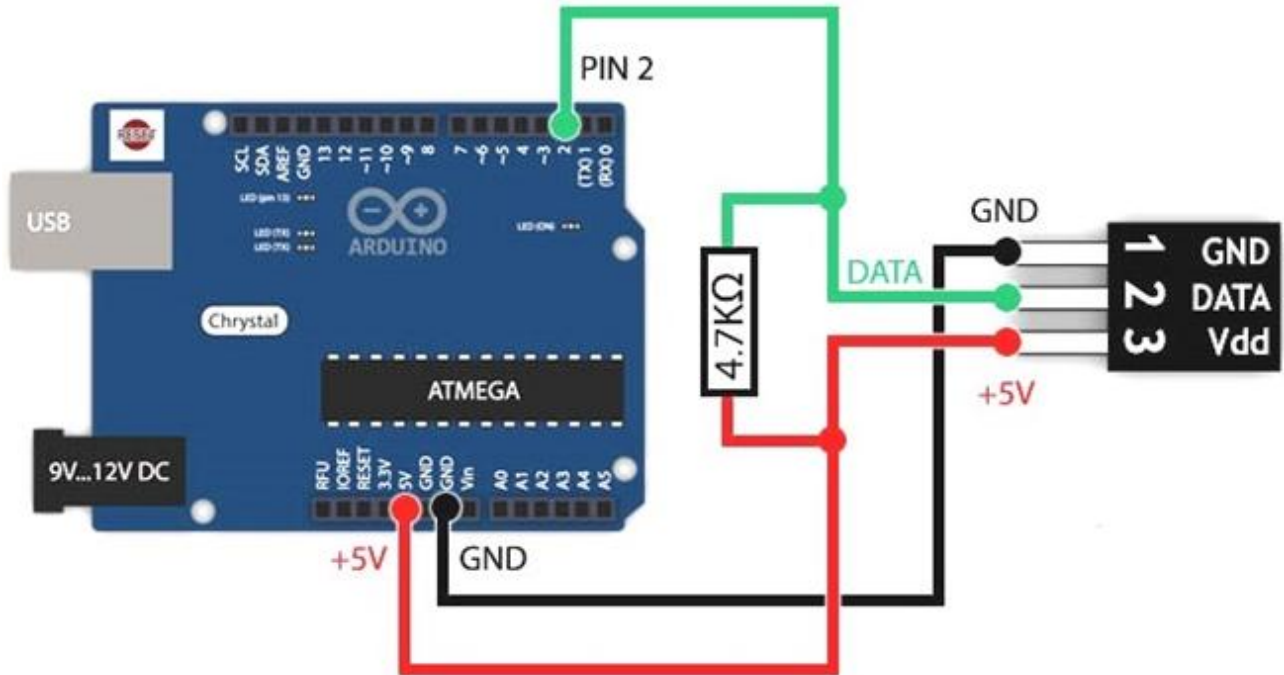


Рис. 2.16.2. Підключення цифрового датчика удару до МПС

При використанні контакту живлення (VDC) на виході датчика (Output) при відсутності спрацювання присутня напруга живлення, а при спрацьовуванні реєструються короткі імпульси в діапазоні від 5 В до 0 В.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку Numeric = 2 у структурі «Case Structure» представлена на рис 2.16.3, що відповідає запуску МПС. Блок-схема віртуального приладу для випадку Numeric = 0 у структурі «Case Structure» представлена на рис 2.16.4, що відповідає зупинці МПС при натисненні кнопки «Stop».

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 2.16.5.

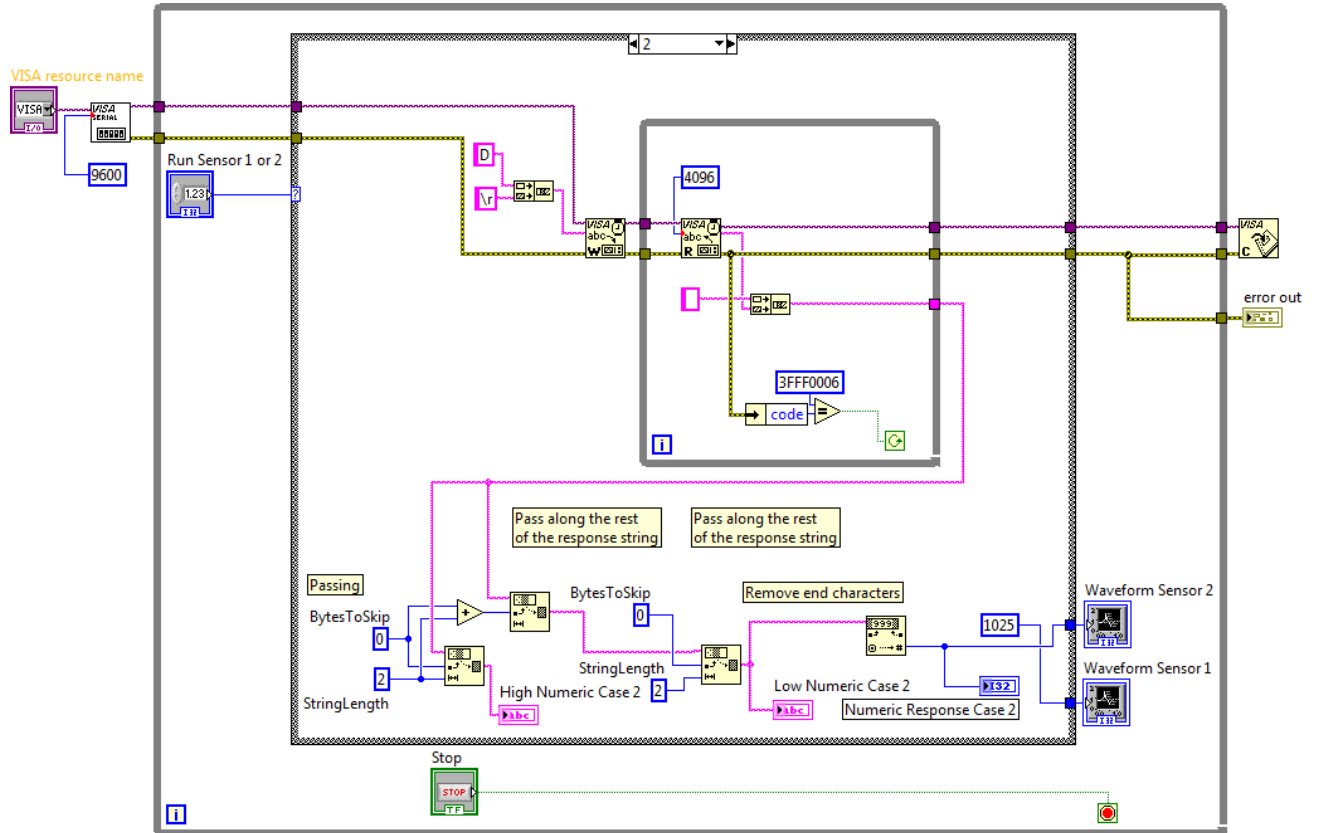


Рис. 2.16.3. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 2

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

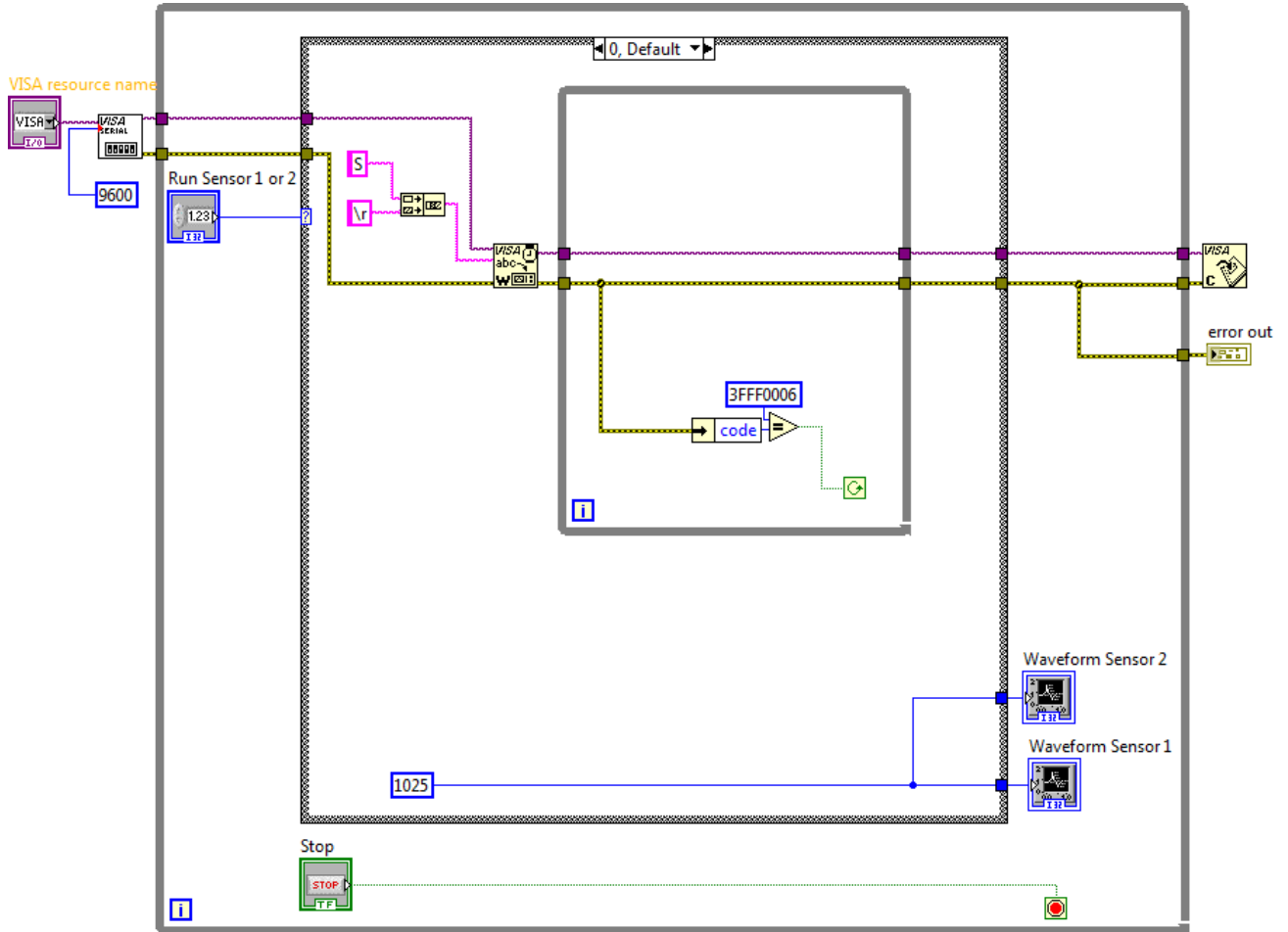


Рис. 2.16.4. Схема приладу в NI LabVIEW 2010 для випадку Numeric = 0

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

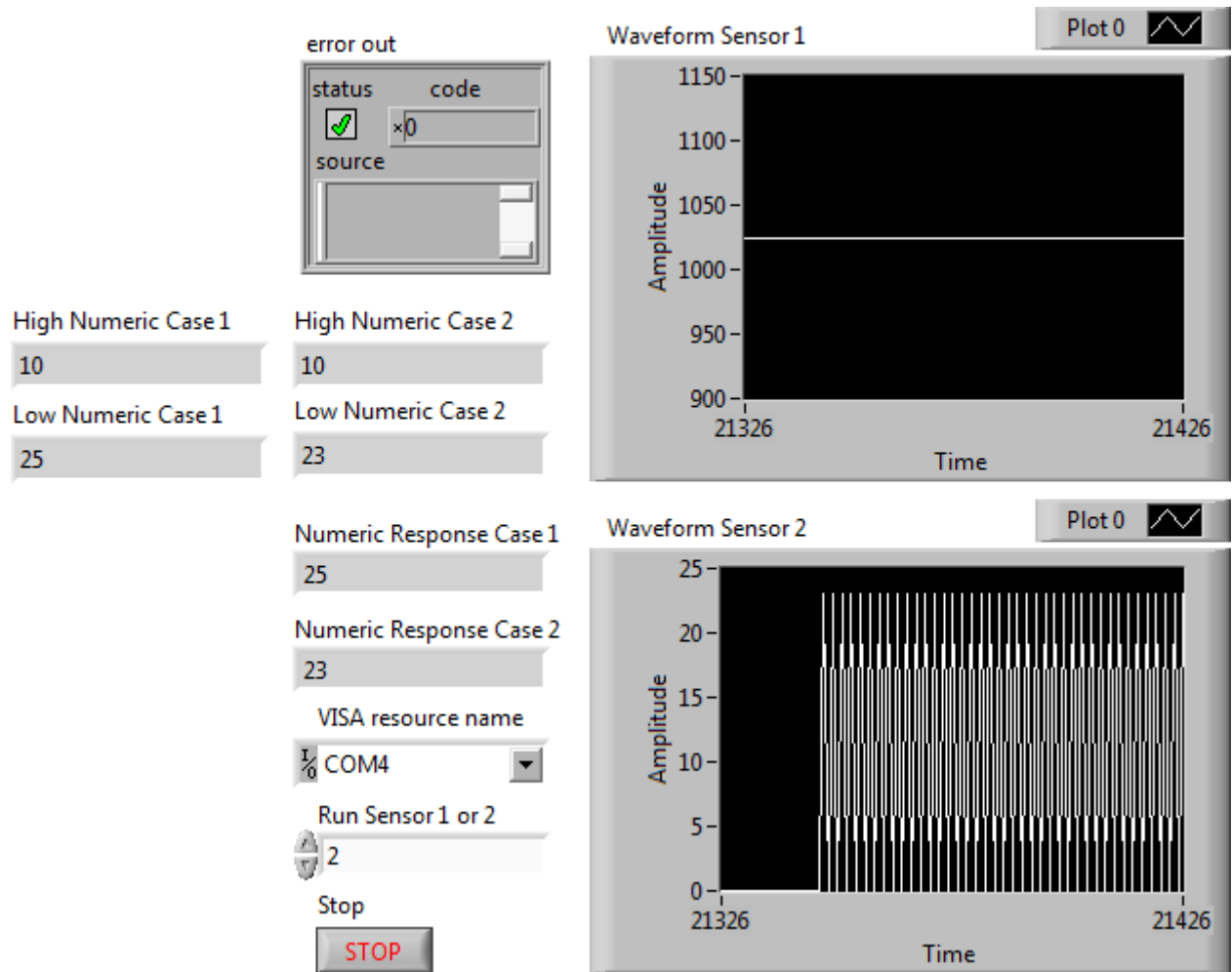


Рис. 2.16.5. Зовнішній вигляд системи керування МПС

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 2.16.3 і рис. 2.16.4. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
```

```
#define ctsPin 13 // Підключення вібраційного датчика
```

```
#define keyPin A2 // Підключення датчика удару
int count = 0;
int oldStat = 0;
int keyState;
int ctsValue;
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролера
void setup() {
    Serial.begin (9600);
    pinMode(ctsPin, INPUT); // Налаштування вібраційного датчика
    pinMode(keyPin, INPUT); // Налаштування датчика удару
    Serial.println("Start");
}
```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```
// Реалізація управління МПС
void loop() {
    count=Serial.read();
    if (count<0) {
        delay(100);
        count=oldStat;
    }
    if(count=='R') {
        keyState = SetDetectionD13();
        Serial.println(keyState);
        delay(100);
        count = 'R';
    }
    if(count=='D') {
        ctsValue = SetDetectionA5();
```

```

        Serial.println(ctsValue);
        delay(100);
        count = 'D';
    }
    if (count=='S') {
        delay (100);
        count='S';
    }
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

// Читання вібраційного датчика
int SetDetectionD13() {
    int data;
    data = digitalRead(ctsPin);
    return data;
}
// Читання датчика удару
int SetDetectionA5() {
    int data;
    data = analogRead(keyPin);
    return data;
}

```

Завантажити програмне забезпечення для мікроконтролеру Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролеру Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?

3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип роботи датчика удару KY-031.

3. Плата для підключення датчиків

Плати для мікроконтролерів Arduino дозволяють підключити велику кількість різноманітних модулів таких як сенсори, серво-приводи, реле, кнопки, потенціометри та ін. Вся периферія для плат Arduino, що пропонується виробниками обладнання, передбачає роз'єми в різноманітних формфакторах для подальшого з'єднання з кінцевими модулями. Плата або шилд одягається безпосередньо на контролер Arduino і не потребує додаткових комутацій і пайки – що є дуже зручним в проектуванні та експлуатації.

3.1. Багатофункціональний навчальний шилд

Ціль роботи – проектування цифрового приладу в LabVIEW з мікропроцесорним управлінням для управління декількома цифровими датчиками з використанням багатофункціонального шилду в середовищі програмування мікропроцесорів Arduino.

Теоретичні відомості

Багатофункціональний навчальний шилд призначений для ознайомлення з платформою Arduino, управління периферією і роботою з датчиками. Плата дозволяє отримати практичний досвід по роботі з декількома цифровими датчиками, зсувними регістрами, управління семисегментними індикаторами, аналоговими і цифровими датчиками температури, введенням з кнопок, модулями Bluetooth, і т.д. Шилд можна використовувати як для навчання так і в промислових конструкціях. Представлена реалізація МПС із багатофункціональним шилдом зводить до мінімуму ризик виходу з ладу елементів контролеру Arduino UNO при їх неправильному підключенні.

Робоче завдання

1. Зібрати досліджувану схему віртуального приладу для управління платою Arduino UNO в National Instruments LabVIEW 2010. Блок-схема віртуального приладу для випадку True у структурі «Case Structure» представлена на рис 3.1.2 і рис. 3.1.3, що відповідає запуску МПС для вимірювання з двома сенсорами. Блок-схема віртуального приладу для випадку False у структурі «Case Structure» представлена на рис 3.1.4 і рис. 3.1.5, що відповідає очікуванню та зупинці МПС при вимкненні тумблерів або натисненні кнопки «Stop».

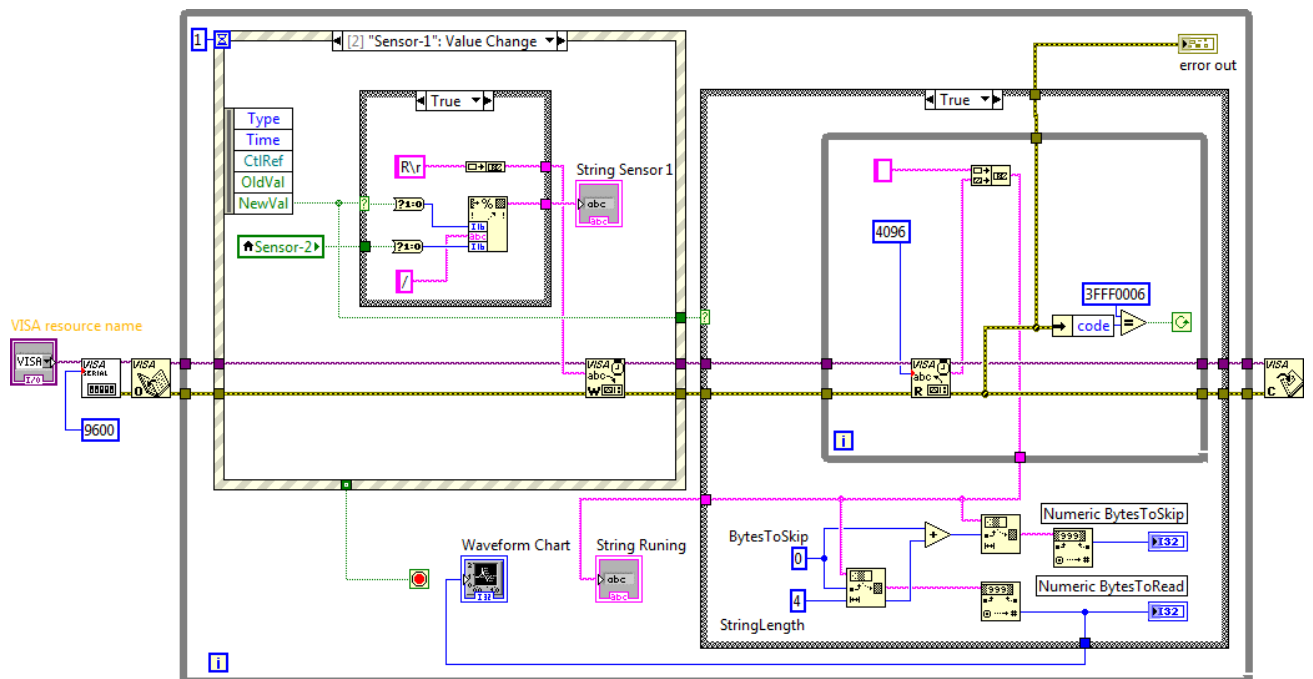


Рис. 3.1.2. Схема приладу в NI LabVIEW 2010 для True Sensor - 1

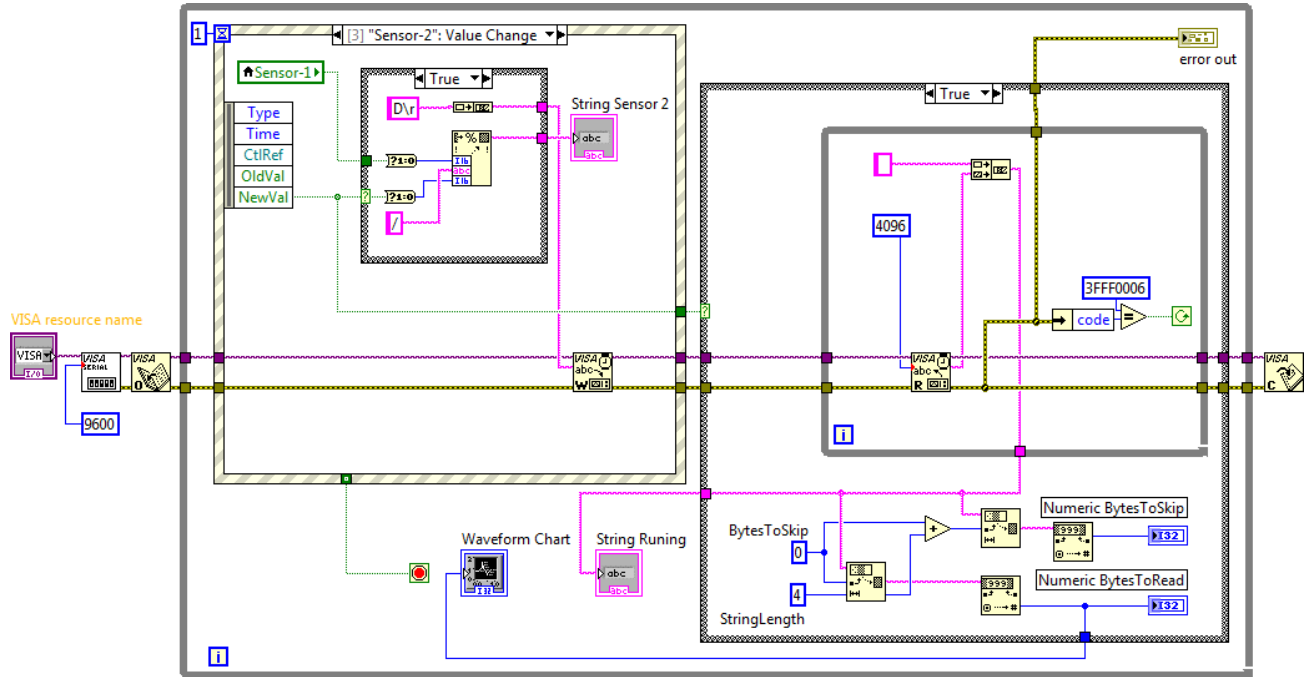


Рис. 3.1.3. Схема приладу в NI LabVIEW 2010 для True Sensor - 2

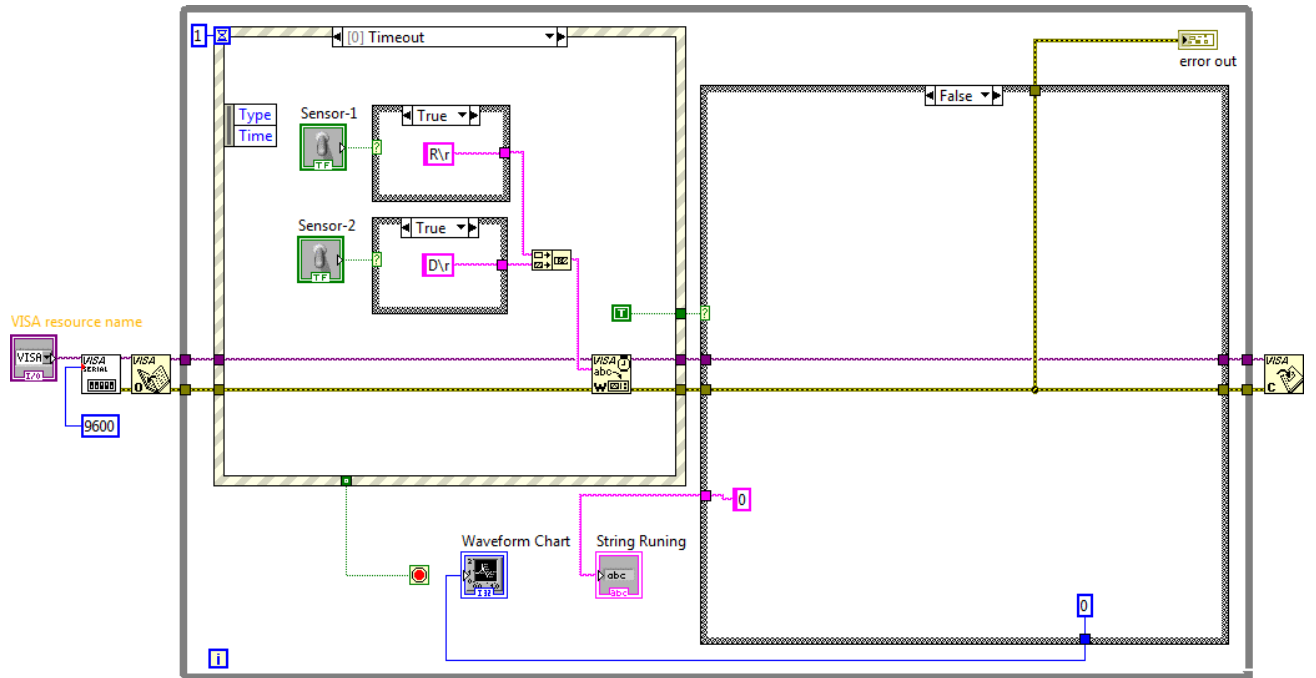


Рис. 3.1.4. Схема приладу в NI LabVIEW 2010 для False - Timeout

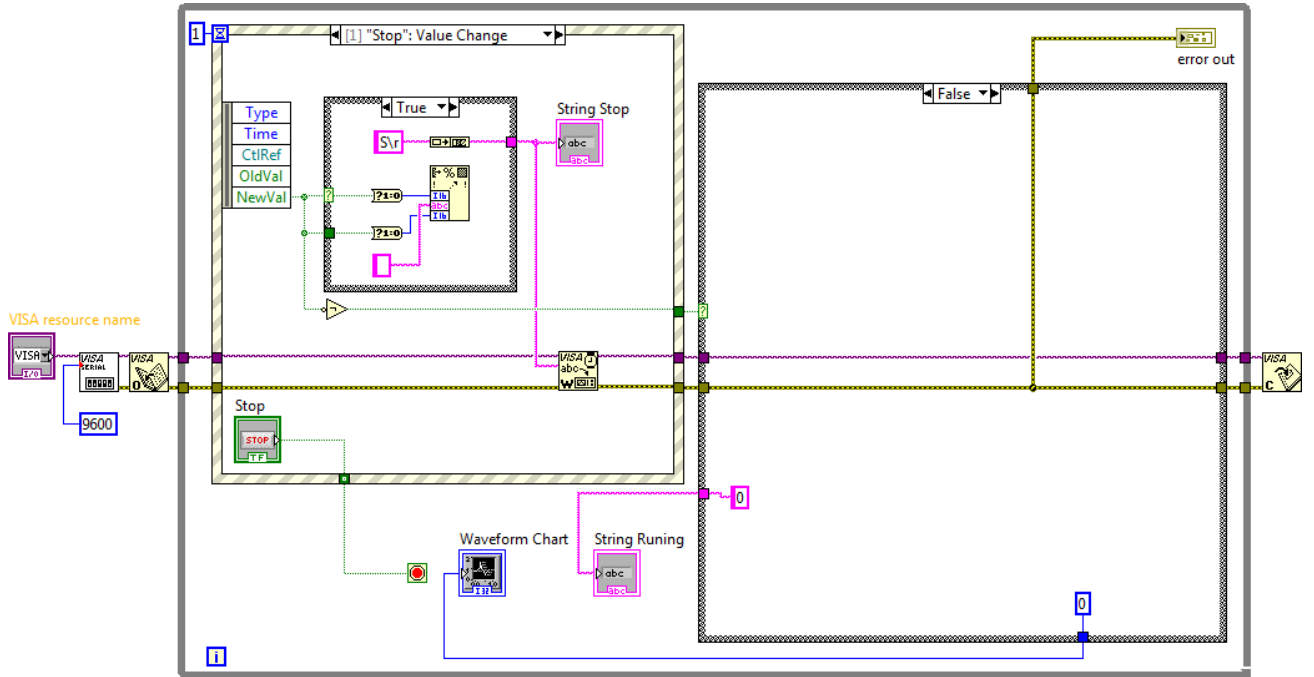


Рис. 3.1.5. Схема приладу в NI LabVIEW 2010 для False - Stop

Зовнішній вигляд системи в National Instruments LabVIEW 2010 для керування МПС на основі мікроконтролеру Arduino UNO представлений на рис. 3.1.6.

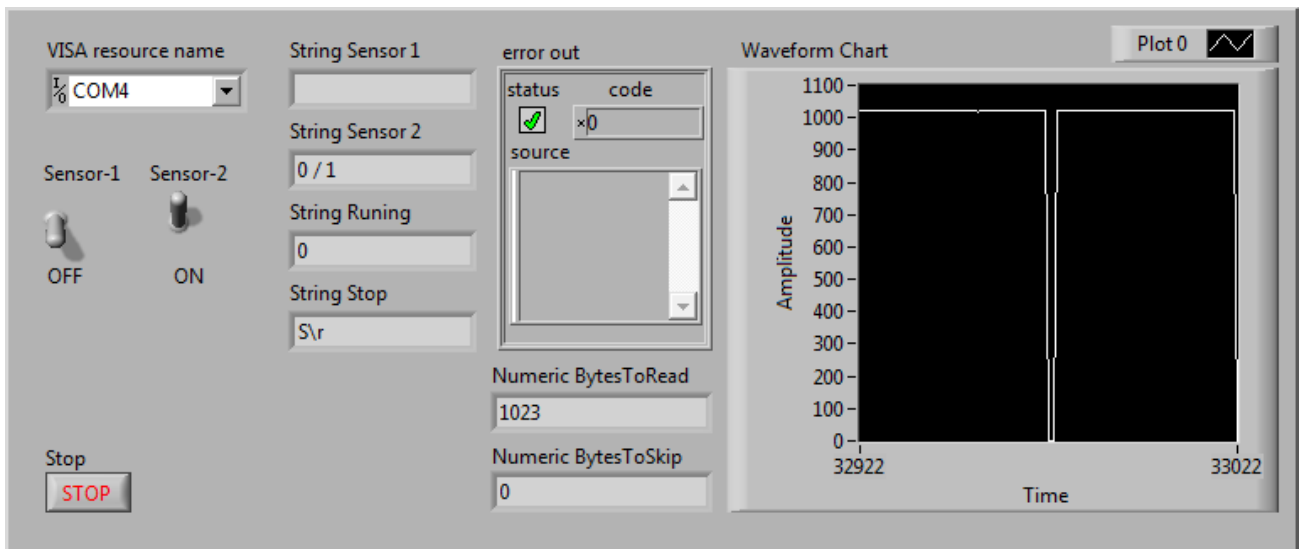


Рис. 3.1.6. Зовнішній вигляд системи керування МПС

2. Дослідити функціональні можливості компонентів LabVIEW 2010 для віртуального інтерфейсу NI Programming Function VI. Описати призначення і функцій роботи послідовним

портом вводу / виводу NI VISA в середовищі програмування мікропроцесорів Arduino.

3. Задokumentувати для звіту отримані в LabVIEW дані за допомогою графічних засобів Waveform Graph. Пояснити алгоритм виконання коду в середовищі програмування Arduino і функцій читання / запису даних в послідовний порт (Serial Port) для управління МПС на платі Arduino UNO.

Сформулювати у звіті по виконаній роботі висновки за результатами досліджень і підготувати відповіді на контрольні питання.

Методичні вказівки

1. Для виконання даної лабораторної роботи необхідно використати елементи керування National Instruments LabVIEW 2010, що представлені на рис. 3.1.2 - рис. 3.1.5. Виконати ініціалізацію даних і змінних у робочому проекті в середовищі програмування мікропроцесорів Arduino:

```
// Ініціалізація даних і змінних
#define levPin A5 // Вивід підключення датчика
#define trigPin 5 // Trigger Pin
/* Define shift register pins used for seven segment display */
#define LATCH_DIO 4
#define CLK_DIO 7
#define DATA_DIO 8
#define Pot1 0;
/* Segment byte maps for numbers 0 to 9 */
const byte SEGMENT_MAP[] = {0xC0,0xF9,0xA4,0xB0,0x99,0x92,0x82,0xF8,0X80,0X90};
/* Byte maps to select digit 1 to 4 */
const byte SEGMENT_SELECT[] = {0xF1,0xF2,0xF4,0xF8};
int ledPin = 9; // Вивід підключення світлодіоду
int levValue; // max Value = 780
```

2. Виконати попереднє налаштування МПС плати Arduino UNO в середовищі програмування мікропроцесорів Arduino:

```
// Налаштування мікроконтролеру
void setup() {
    Serial.begin(9600);
```

```

/* Set DIO pins to outputs */
pinMode(LATCH_DIO,OUTPUT);
pinMode(CLK_DIO,OUTPUT);
pinMode(DATA_DIO,OUTPUT);
pinMode(trigPin, OUTPUT);
pinMode(ledPin, OUTPUT);
pinMode(levPin, INPUT);
}

```

3. Використати методи роботи з послідовними портами вводу / виводу (Serial Port) в середовищі програмування мікропроцесорів Arduino:

```

// Реалізація управління МПЦ
void loop() {
    int PotValue;
    levValue = analogRead(levPin);
    PotValue = analogRead(Pot1);
    Serial.print("Sig Output Level: ");
    Serial.println(levValue);
    /* Update the display with the current counter value */
    WriteNumberToSegment(0 , levValue / 1000);
    WriteNumberToSegment(1 , (levValue / 100) % 10);
    WriteNumberToSegment(2 , (levValue / 10) % 10);
    WriteNumberToSegment(3 , levValue % 10);
}

```

4. Реалізувати процедури обробки функцій користувача в середовищі програмування мікропроцесорів Arduino:

```

/* Write a decimal number between 0 and 9 to one of the 4 digits of the display */
void WriteNumberToSegment(byte Segment, byte Value) {
    digitalWrite(LATCH_DIO,LOW);
    shiftOut(DATA_DIO, CLK_DIO, MSBFIRST, SEGMENT_MAP[Value]);
    shiftOut(DATA_DIO, CLK_DIO, MSBFIRST, SEGMENT_SELECT[Segment] );
    digitalWrite(LATCH_DIO,HIGH);
}

```

Завантажити програмне забезпечення для мікроконтролера Arduino UNO, що розроблено в середовищі програмування Arduino у лабораторний макет МПС. Запустити проект віртуального приладу в NI LabVIEW 2010. За допомогою графічних засобів Waveform Graph отримати характеристики віртуального приладу.

Контрольні питання

1. Які технічні характеристики МПС на основі мікроконтролера Arduino?
2. У чому полягає робота функцій з послідовним портом вводу / виводу NI VISA?
3. Поясніть методи роботи з послідовними портами вводу / виводу (Serial Port).
4. Поясніть принцип виведення даних на світлодіодну матрицю (SPI)

Література

1. Гусев, В.Г. Электроника и микропроцессорная техника: Учебник / В.Г. Гусев, Ю.М. Гусев. - М.: КноРус, 2013. - 800 с.
2. Брамм, П. Микропроцессор 80386 и его программирование / П. Брамм, Б. Брамм. - М.: Мир, 1990.
3. Гребенко, Ю.А. Микропроцессоры / Ю.А. Гребенко, В.К. Раков. - М.: Издательство МЭИ, 2000.
4. Гуревич, В.И. Микропроцессорные реле защиты. Устройство, проблемы, перспективы / В.И. Гуревич. - М.: Инфра-Инженерия, 2011. - 336 с.
5. Иванников, А.Д. Моделирование микропроцессорных систем / А.Д. Иванников. - М.: Энергоатомиздат, 1990.
6. Калабегов, Б.А. Цифровые устройства и микропроцессорные системы / Б.А. Калабегов. - М.: Горячая линия-Телеком, 2007.-336с.
7. Калабеков, Б.А. Цифровые устройства и микропроцессоры / Б.А. Калабеков. - М.: Радио и связь, 1997.
8. Калашников, В.И. Электроника и микропроцессорная техника: Учебник для студ. учреждений высш. проф. обр. / В.И. Калашников, С.В. Нефедов. - М.: ИЦ Академия, 2012. - 368 с.

9. Кузин, А.В. Микропроцессорная техника: Учебник для студ. сред. проф. образования / А.В. Кузин, М.А. Жаворонков. - М.: ИЦ Академия, 2013. - 304 с.
10. Новиков, Ю.В. Основы микропроцессорной техники: курс лекций / Ю.В. Новиков, П.К. Скоробогатов. - М.: ИНТУИТ.РУ, 2003. - 440 с.
11. Новиков, Ю.В. Основы микропроцессорной техники: Учебное пособие / Ю.В. Новиков, П.К. Скоробогатов. - М.: БИНОМ. ЛЗ, ИНТУИТ.РУ, 2012. - 357 с.
12. Новожилов, О.П. Основы микропроцессорной техники. В 2-х т. Т. 2. Основы микропроцессорной техники: Учебное пособие / О.П. Новожилов. - М.: ИП РадиоСофт, 2011. - 336 с.
13. Шевкопляс, Б.В. Микропроцессорные структуры. Инженерные решения: справочник / Б.В. Шевкопляс. - М.: Радио и связь, 1993.
14. Шонфелдер, Г. Измерительные устройства на базе микропроцессора ATmega: Пер. с англ. / Г. Шонфелдер, К. Шнайдер. - СПб.: БХВ-Петербург, 2012. - 288 с.