МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ ІМ. В. ДАЛЯ
ФАКУЛЬТЕТ ІНФОРМАЦІЙНИХ ТЕХНОЛОГІЙ ТА ЕЛЕКТРОНІКИ
КАФЕДРА КОМП'ЮТЕРНИХ НАУК ТА ІНЖЕНЕРІЇ

До захисту допускається
Завідувач кафедри
_____ І. С.Скарга-Бандурова
«____»_____ 2020 р.

ДИПЛОМНИЙ ПРОЕКТ (РОБОТА) БАКАЛАВРА
ПОЯСНЮВАЛЬНА ЗАПИСКА

НА ТЕМУ:

«Апаратно-програмні засоби підключення мікроконтролерів AVR до USB»

Освітній ступінь «бакалавр»
Спеціальність 123 « Комп'ютерна інженерія»

Керівник проекту: _____ (підпис) Кардашук В. С. (ініціали, прізвище)

Консультант з охорони праці: _____ (підпис) Критська Я. О. (ініціали, прізвище)

Здобувач вищої освіти: _____ (підпис) Прядченко В. С. (ініціали, прізвище)

Група: КІ-16 д

Сєвєродонецьк 2020

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СХІДНОУКРАЇНСЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ВОЛОДИМИРА ДАЛЯ

Факультет Інформаційних технологій та електроніки
Кафедра Комп'ютерних наук та інженерії
Освітній ступінь бакалавр
Спеціальність 123 «Комп'ютерна інженерія»

ЗАТВЕРДЖУЮ:

Т.в.о. завідувача кафедри КНІ
_____ С.О. Сафонова
«_____» _____ 2020 р.

ЗАВДАННЯ
НА ДИПЛОМНИЙ ПРОЕКТ (РОБОТУ) СТУДЕНТУ

Прядченку Вадиму Сергійовичу
(прізвище, ім'я, по-батькові)

1. Тема проекту (роботи): «Апаратно-програмні засоби підключення мікроконтролерів AVR до USB» затверджена наказом по університету № 73/15.15 від «30» квітня 2020 р.
2. Строк здачі студентом закінченого проекту (роботи): 10.06.2020 р.
3. Вихідні дані проекту (роботи): матеріали переддипломної практики
4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити):
   1. Аналіз функцій шини USB.
   2. Обґрунтування та вибір елементної бази апаратної реалізації шини USB.
   3. Розроблення програмної реалізації з'єднання мікроконтролера та комп'ютера.
   4. Охорона праці.
5. **Перелік графічного матеріалу (з точною назвою обов'язкових креслень):**

Електронні плакати

6. Консультанти роботи, з вказівкою розділів, що до них відносяться

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Основна частина	Кардашук В. С. к.т.н., доц.		
Охорона праці	Критська Я. О. ст. викл.		

7. Дата видачі завдання _____

Керівник _____ Кардашук В. С.
(підпис)

Завдання до виконання прийняв _____ Прядченко В. С.
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Найменування етапів дипломного проекту (роботи)	Строк виконання етапів проекту (роботи)	Примітки
1.	Отримання завдання, збір матеріалів	18.05.20- 24.05.20	
2.	Огляд літератури й постановка задачі на розроблення	25.05.20–28.05.20	
3.	Вибір варіантів підключення МК	29.05.20 – 30.05.20	
4.	Вибір елементної бази	31.05.20 – 01.06.20	
5.	Реалізація апаратно-програмного забезпечення	02.06.20 – 08. 06.20	
6.	Оформлення пояснювальної записки	08.06.20 – 09.06.20	
7.	Підготовка та подання роботи до захисту	09.06.20 – 10.06.20	

Здобувач вищої освіти _____ Прядченко В. С.
(підпис) (ініціали, прізвище)

Керівник _____ Кардашук В. С.
(підпис) (ініціали, прізвище)

РЕФЕРАТ

Пояснювальна записка до дипломного проекту бакалавра: 84 сторінки, 19 рисунків, 4 таблиці, 25 джерел.

У дипломному проекті розглянуті та запропоновані варіанти підключення мікроконтролерів AVR до USB порту комп'ютера.

Універсальна послідовна шина (USB) стала надзвичайно популярною за рахунок надання ряду зручностей кінцевим користувачам.

У дипломну проекті проведено огляд та аналіз апаратних та програмних засобів реалізації послідовної шини USB з використанням мікроконтролерів AVR. За результатами дослідження сформульовані мета та завдання випускної роботи. Здійснена постановка задачі на розробку апаратної та програмної частини реалізації послідовної шини USB. Визначені шляхи реалізації поставленого завдання та елементна база з використанням мікроконтролерів AVR фірми Atmel.

У якості реалізації апаратної частини обрано мікроконтролера ATTiny2313 фірми Atmel для застосування в системах передачі інформації. Програмування мікроконтролера проведено в інструментальному середовищі налаштування AVR Studio 4.

ПОСЛІДОВНА ШИНА, АЛГОРИТМ, МІКРОКОНТРОЛЕР, ПЕРЕДАЧА ДАНИХ, НАЛАШТУВАННЯ.

Умови отримання дипломного проекту:

СНУ ім. Володимира Даля, пр. Центральний 59а, м. Сєвєродонецьк, 93406**.**

ЗМІСТ

ВСТУП

USB (Universal Serial Bus – універсальна послідовна шина) є промисловим стандартом розширення архітектури ПК, орієнтованим на інтеграцію з телефонією і пристроями побутової електроніки. Версія 1.0 була опублікована на початку 1996 року, більшість пристроїв підтримує версію 1.1, яка вийшла восени 1998 року, – в ній були усунуті знайдені проблеми першої редакції. Весною 2000 року опублікована специфікація USB 2.0, в якій передбачено 40-кратне підвищення пропускної здатності шини. Спочатку (у версіях 1.0 і 1.1) шина забезпечувала дві швидкості передавання інформації: повна швидкість FS (full speed) – 12 Мбіт/с і низька швидкість LS (Low Speed) – 1,5 Мбіт/с. У версії 2.0 визначена ще й висока швидкість HS (High Speed) – 480 Мбіт/с, яка дозволяє суттєво розширити обсяг пристроїв, які підключаються до шини. На сучасному рівні активно використовується версія USB 3.0.

На відміну від громіздких дорогих шлейфів паралельних шин ATA і особливо шини SCSI з її різноманітністю роз'ємів і складністю правил підключення, кабельний набір USB простий. Кабель USB містить одну екрановану виту пару з імпедансом 90 Ом для сигнальних кіл і одну неекрановану для подання живлення (+5 В), допустима довжина сегмента – до 5 м. Для низької швидкості може використовуватися не витий неекранований кабель довжиною до 3 м.

USB забезпечує обмін даними між хост-комп'ютером і множиною периферійних пристроїв (ПП). Згідно з специфікацією USB, пристрої (devices) можуть бути хабами, функціями або їх комбінацією. Пристрій-хаб (hub) лише забезпечує додаткові точки підключення пристроїв до шини. Пристрій-функція (function) USB надає системі додаткові функціональні можливості, наприклад підключення до ISDN, цифровий джойстик, акустичні колонки з цифровим інтерфейсом і т. п. Комбінований пристрій (compound

device), який містить декілька функцій, подається як хаб з підключеними до нього декількома пристроями.

Одне з питань, що викликають підвищений інтерес у початківців (та й у досвідчених) конструкторів мікропроцесорних систем є питання взаємодії мікроконтролера з комп'ютером. І це не дивно. Іноді дуже потрібно створити якийсь пристрій, яке може працювати не тільки автономно, але і під управлінням комп'ютера. Іноді бажано оперативно зчитувати інформацію з пристрою на мікроконтролері або завдання полягає в тому, що б розробити мікроконтролерних приставку для сполучення комп'ютера і який не будь зовнішньої керованої ним системою. Природно, що два пристрої, що працюють на основі мікропроцесорних технологій (комп'ютер і мікроконтролер) завжди можна змусити обмінюватися між собою інформацією.

Метою даної дипломної роботи є реалізація варіантів підключення мікроконтролерів AVR фірми ATMEL через USB порт комп’ютера.

1 АПАРАТНО-ПРОГРАМНА РЕАЛІЗАЦІЯ ШИНИ USB

1.1 Загальні положення

Пристрій USB повинно мати інтерфейс USB, який забезпечує повну підтримку протоколу USB, виконання стандартних операцій (конфігурація і скидання) і подання інформації, яка описує пристрій. Роботою всієї системи USB керує хост-контролер (host controller), який є програмно-апаратною підсистемою хост-комп’ютера. Шина дозволяє підключати, конфігурувати, використовувати і відмикати пристрої під час роботи хосту і самих пристроїв [1].

Шина USB є хост-центровою: єдиним ведучим пристроєм, який керує обміном є хост-комп’ютер, а всі приєднані до неї периферійні пристрої – виключно ведені. Фізична топологія шини USB – багатоярусна зірка. Її вершиною є хост-контролер, об’єднаний з кореневим хабом (root hub), як правило двопортовим. Хаб є пристроєм-розгалужувачем, він може бути і джерелом живлення для підключених до нього пристроїв.

До кожного порту хаба може безпосередньо підключатись периферійний пристрій або проміжний хаб; шина допускає до п’яти рівнів каскадів хабів (не враховуючи кореневого). Оскільки комбіновані пристрої всередині себе місять хаб, їх підключення до хабу шостого ярусу вже недопустиме. Кожний проміжний хаб має декілька низхідних (downstream) портів для підключення периферійних пристроїв і один висхідний (upsteam) порт для підключення до кореневого хаба або низхідного порту вищого за ієрархією хаба.

Логічна топологія USB – просто зірка: для хост-контролера хаби створюють ілюзію безпосереднього підключення кожного пристрою. На відміну від шин розширення (ISA, PCI, PC Card), де програма взаємодіє з пристроями за допомогою звернень за фізичними адресами комірок пам’яті, портів введення-виведення, переривань і каналів DMA, взаємодія додатків з

пристроями USB виконується лише через програмний інтерфейс. Цей інтерфейс, який забезпечує незалежність звернень до пристроїв, пропонується системним ПЗ контролера USB.

На відміну від громіздких дорогих шлейфів паралельних шин ATA і особливо шини SCSI з її різноманітністю роз'ємів і складністю правил підключення, кабельний набір USB простий. Кабель USB містить одну екрановану виту пару з імпедансом 90 Ом для сигнальних кіл і одну неекрановану для подання живлення (+5 В), допустима довжина сегмента – до 5 м. Для низької швидкості може використовуватися не витий неекранований кабель довжиною до 3 м (він дешевший).

Система кабелів і концентраторів USB не дає можливості помилитись при підключенні пристроїв. Для розпізнання роз'єму USB на корпусі пристрою ставиться стандартне символічне позначення. Гнізда типу «А» встановлюються лише на низхідних портах хабів, вилки типу «А» - на шнурах периферійних пристроїв або вищих портів хабів. Гнізда і вилки типу «В» використовуються лише для шнурів, які від'єднуються від периферійних пристроїв висхідних портів хабів. Окрім стандартних роз'ємів, показаних на рисунку 1.1 (а, б), використовуються і мініатюрні варіанти (рис. 1.1 в,г).

Хаби і пристрої забезпечують можливість «гарячого» підключення і відключення. Для цього роз'єми забезпечують більш раннє з'єднання і пізнє від'єднання кіл живлення відносно сигнальних, крім того, передбачений протокол сигналізації підключення і відключення пристроїв. Всі кабелі USB «прямі» – в них з'єднуються однойменні кола роз'ємів.

Призначення виводів роз'ємів USB наведені в табл. 1.1, нумерація контактів показана на рис. 1.1 [2].

Таблиця 1.1 – Призначення виводів роз’єму USB

Контакт	Сигнал
1	VBus (+5В)
2	D–
3	D+
4	GND

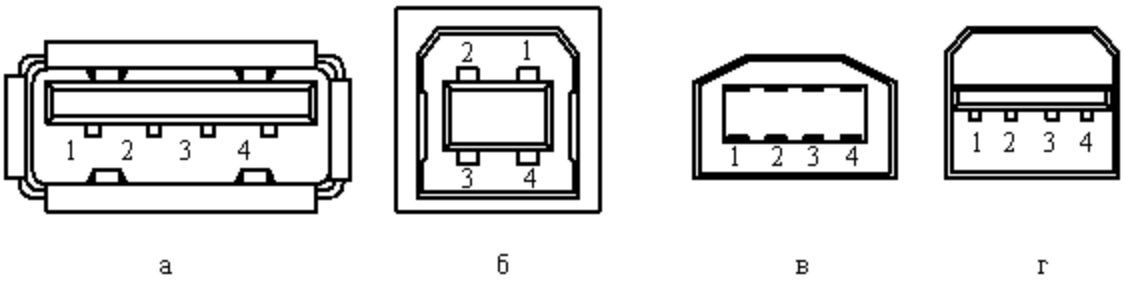


Рисунок 1.1 – Гнізда USB: а – тип «А», б – тип «В» стандартні, в, г – мініатюрні типу «В»

У шині використовується диференціальний спосіб передачі сигналів D+ і D– по двох проводах (рис. 1.2) [2].

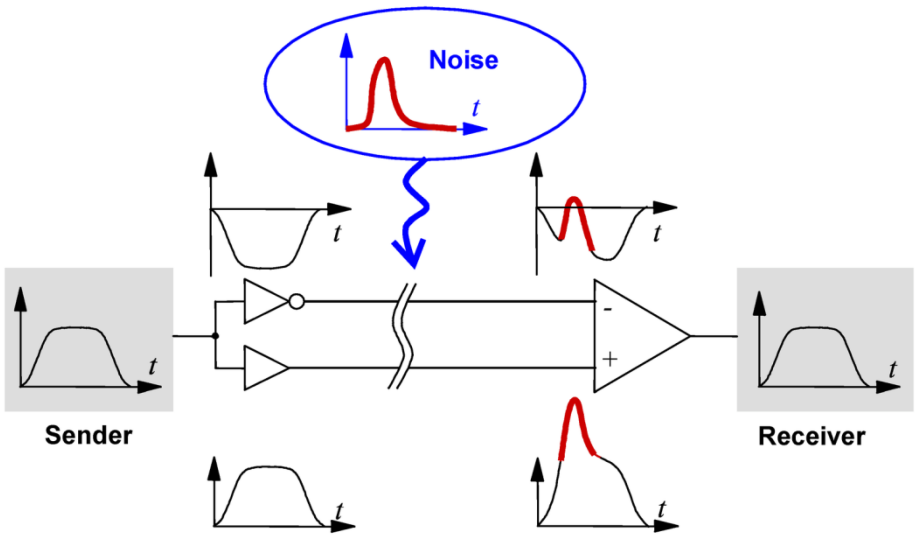


Рисунок 1.2 – Ілюстрація диференціального способу передачі сигналів

Швидкість пристрою, підключеного до конкретного порту, визначається хабом за рівнями сигналів на лініях D+ і D–, зміщених навантажувальними резисторами прийомопередавачів: пристрої із низькою швидкістю підвищують до високого рівня лінію D–, з повною – D+. Підключення пристрою HS визначається на етапі обміну конфігураційною інформацією – фізично на перший час пристрій HS повинен підключатись як FS. Передавання по двох провідниках в USB не обмежується диференціальними сигналами.

Крім диференціального приймача, кожний пристрій має лінійні приймачі сигналів D+ і D–, а передавачі цих ліній керуються індивідуально. Це дозволяє розрізняти більше двох станів лінії, які використовуються для організації апаратного інтерфейсу.

Введення високої швидкості (480 Мбіт/с – всього в два рази менше ніж Gigabit Ethernet) потребує ретельного узгодження прийомопередавачів і лінії зв’язку. На цій швидкості може працювати лише кабель з екранованою витою парою для сигнальних станцій. Для високої швидкості апаратура USB повинна мати додаткові спеціальні прийомопередавачі. На відміну від формувачів потенціалу для режимів FS і LS, передавачі HS є джерелами струму, орієнтованими на присутність резисторів-термінаторів на обох сигнальних лініях.

Швидкість передавання даних (LS, FS або HS) вибирається розробником периферійного пристрою відповідно до потреб цього пристрою. Реалізація низьких швидкостей для пристрою коштує дещо дешевше (прийомопередавачі простіші, а кабель для LS може бути і неекранованою витою парою). Якщо в попередній версії USB пристрої можна було, не задумуючись, підключати будь-який вільний порт будь-якого хаба, то в USB 2.0 при наявності пристроїв і хабів різних версій з’явились можливості вибору між оптимальними, неоптимальними і нероботопридатними конфігураціями.

Хаби USB 1.1 зобов’язані підтримувати швидкості FS і LS, швидкість підключеного до хабу пристрою визначається автоматично за різницею потенціалів сигнальних ліній. Хаби USB1.1 при передаванні пакетів є просто повторювачами, які забезпечують прозорий зв’язок периферійного пристрою з контролером.

Передавання на низькій швидкості досить неефективно використовують потенційну пропускну здатність шини: за той час, на який вони займають шину, високошвидкісний пристрій може передати даних у 8 раз більше. Але заради спрощення і здешевлення всієї системи на ці втрати пішли, а за розподіленням смуги між різними пристроями слідкує планувальник транзакцій хост-контролера.

В специфікації 2.0 швидкість 480 Мбіт/с повинна затримуватись з попередніми, але при такому співвідношенні швидкостей зміни на FS і LS займуть можливу смугу пропускання шини. Щоб цього не сталося, хаби USB 2.0 набувають властивості комутаторів пакетів. Якщо до порту такого хаба підключений високошвидкісний пристрій (або аналогічний хаб), то хаб працює в режимі повторювача, і транзакція з пристроєм на HS займає весь канал до хост-контролера на весь час свого виконання.

Якщо ж до порту такого хабу USB 2.0 підключається пристрій або хаб 1.1, то по частині каналу до контролера пакет проходить на швидкості HS, запам’ятовується в буфері хаба, а до старого пристрою або хабу йде вже на його швидкості FS або LS. При цьому функції контролера хаба 2.0 (включно з кореневим) ускладнюються.

Кожен пристрій на шині USB (їх може бути до 127) при підключенні автоматично отримує свою унікальну адресу. Логічно пристрій являє собою набір незалежних кінцевих точок (endpoint, EP), з якими хост-контролер (і клієнтське ПЗ) обмінюються інформацією. Кожна кінцева точка має свій номер і описується такими параметрами:

- необхідна частота доступу до шини і допустимі затримки обслуговування;

- необхідна смуга пропускання каналу;
- вимоги до обробки посилок;
- максимальні розміри переданих і прийнятих пакетів;
- тип передавання;
- напрям передавання (для передавання масивів і ізохронного обміну).

Кожний пристрій обов'язково містить кінцеву точку з номером 0, яка використовується для ініціалізації, загального управління і опитування стану пристрою. Ця точка завжди встановлена при включенні живлення і підключенні пристрою до шини. Вона підтримує передавання типу «керування».

1.2 Принцип дії послідовної шини USB

На рис. 1.3 наведені осцилограми сигналів низькошвидкісного драйвера USB [2].

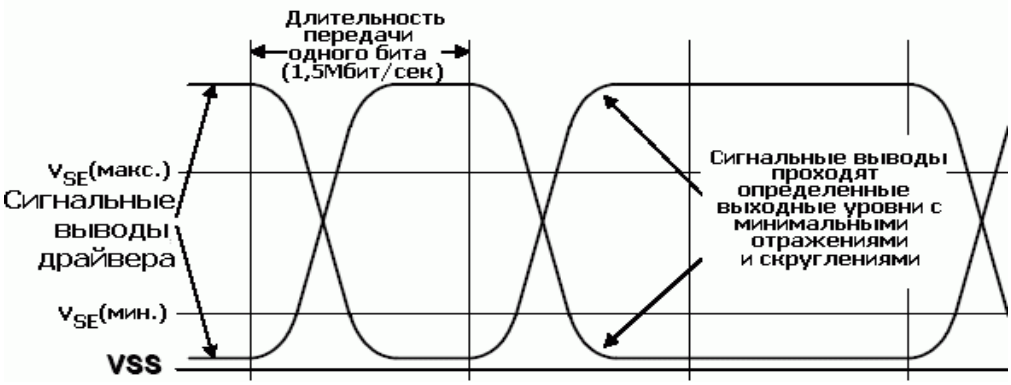


Рисунок 1.3– Осцилограми сигналів низькошвидкісного драйвера USB

На рис. 1.4 наведені рівні напруг при передачі пакетів по шині USB [2].

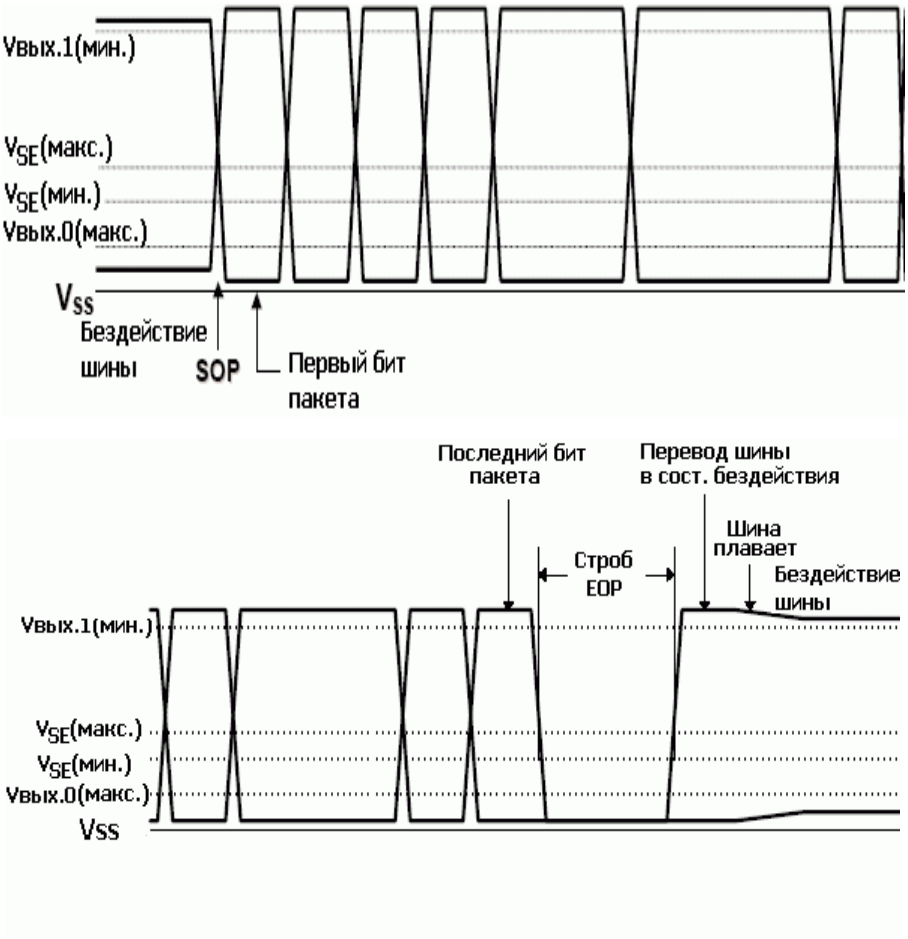


Рисунок 1.4 – Рівні напруг при передачі пакетів по шині USB

Всі приклади прив’язані до лінійки МК AT90S* фірми ATMEL. Така лінійка МК дуже популярна серед розробників.

1.3 Варіанти з`єднання мікроконтролера з комп’ютером

Проблема з'єднання мікроконтролера і комп'ютера в різні часи вирішувалася по-різному. У часи, коли царювали комп'ютери під назвою PC XT, одним з простих і ефективних способів підключити свій власний пристрій до такого комп'ютера було самостійне виготовлення нестандартного внутрішнього модуля, який вставлявся в шину ISA всередині корпусу комп'ютера на материнську плату. Шина ISA в ті часи використовувалася для підключення всіх пристроїв розширення, таких як відеокарти, модулі додаткової пам'яті, контролери жорстких дисків тощо.

Роз'єм для підключення до такої шині можна було при певній вправності, виготовити самостійно. На зміну шині ISA давно прийшла шина, під назвою PCI, а потім PCI-Express. У ній використовуються набагато більш дрібних контактних площадок і логіка її роботи теж набагато складніше. Тепер для підключення зовнішніх пристроїв нам доводиться використовувати один із стандартних інтерфейсів введення виведення. Якщо відкинути такі складні в реалізації варіанти, як підключення по ІК порту або по Bluetooth (блютуз), то залишається тільки три варіанти: підключення по LPT, підключення за допомогою COM-порту і нарешті, підключення по USB.

1.3.1 Підключення за допомогою LPT порту

Це найстаріший і найпростіший з точки зору програмної і апаратної реалізації спосіб. Справа в тому, що LPT порт комп'ютера влаштований таким чином, що дозволяє легко безпосередньо програмним шляхом керувати всіма його виходами і читати інформацію з усіх його входів. Стандартний LPT порт має 8 ліній шини даних, які можуть працювати як входи і як виходи, чотири виходи управління, і п'ять входів для службових сигналів. Якщо ви підключаєте до порту своє власне мікропроцесорний пристрій, то використовувати лінії порту за призначенням зовсім не обов'язково. Всі його входи і всі виходи, як основні, так і службові, ви можете використовувати на свій розсуд. В результаті ви отримуєте інтерфейс, протокол роботи якого який ви можете розробляти на свій розсуд. Ви можете так само програмним шляхом реалізувати деякі стандартні протоколи. Наприклад, програматор PonyProg, описаний в [1] при роботі з LPT портом програмно реалізує послідовний канал зв'язку SPI. Це один з протоколів, за допомогою якого можна «прошити» програму в мікроконтролер AVR.

Використання LPT для зв'язку з комп'ютером мало і свої недолік.

По-перше, операційні системи, зокрема, останні версії Windows XP і тим більше Windows 7 захищають LPT порт від прямого доступу з боку призначених для користувача програм. Тому написання програм, що працюють з LPT вкрай ускладнюється.

По-друге, LPT порти просто виходять з ужитку. Всі сучасні принтери вже зараз підключаються тільки по USB. У ноутбуках вже стало стандартом відсутність як LPT, так і COM портів.

1.3.2 Підключення за допомогою COM-порту

Так само, як LPT послідовні COM-порти теж вже вийшли з ужитку. Працювати з COM портом значно складніше, ніж з LPT. Стандартний COM порт має набагато менше виводів. І пряме управління можливо лише деякими з них. Використовувати їх не має сенсу. Передача інформації в COM-порт відбувається всього по двох лініях. Це лінія RxD (прийом даних), і лінія TxD (передача даних). Для передачі даних використовується спеціальний протокол, який називається RS232. Більшість сучасних мікроконтролерів мають вбудований інтерфейс, сумісний з RS232. Це полегшує підключення. Однак є одна трудність. Повноцінний протокол RS232 передбачає свій власний стандарт рівнів вихідних і вхідних сигналів.

Сигнал на виході TxD COM-порту приймає два значення: логічна одиниця - плюс 12В, логічний нуль - мінус 12В. Такий розмах прийнятий для зменшення впливу перешкод.

Такий же сигнал потрібно подавати і на вхід RxD. Послідовний канал мікроконтролера підтримує інші значення рівнів сигналу. Там сигнал використовуються стандартні логічні рівні. Логічний нуль - 0В. Логічна одиниця - приблизно +5. Тому для зв'язку COM порту і мікроконтролера необхідно узгодити рівні сигналів. Для цього зазвичай застосовується спеціальна мікросхема фірми MAXIM. Це мікросхема називається MAX232A. Це дуже зручна у використанні мікросхема. Вона вимагає лише

одного напруги живлення: +5 В. Усередині мікросхема містить два перетворювача, які використовуються для отримання необхідних для роботи напруг + 12В і 12В.

1.3.3 Підключення за допомогою USB порту

Безсумнівно, це сучасний та найперспективніший спосіб підключення. У той же час і найскладніший з точки зору програмної реалізації. USB порт був спеціально розроблений, як універсальний послідовний порт для підключення всіх видів зовнішніх периферійних пристроїв. За допомогою цього порту до комп'ютера можуть підключатися зовнішні накопичувачі на жорстких дисках і на Флеш-пам'яті, джойстики, мишки, звукові системи, WEB-камери, MP3 плеєри і навіть зовнішні TV-тюнери. Чинний нині стандарт USB версії 2.0 підтримує передачу даних зі швидкістю від 1.5 Мбіт/сек до 480 Мбіт/сек.

Складність реалізації цього способу полягає в тому, що протокол USB - це серйозний багаторівневий протокол передачі інформації. Найнижчий рівень цього протоколу визначає правила стосуються формування імпульсів: тривалість, розмах, спосіб кодування даних, методи синхронізації, методи перевірки помилок. На низькому рівні так само визначається, як біти складаються в байти. Наступний, більш високий рівень стосується передачі самих даних. Адже передаються не просто байти. З цих байтів складаються команди протоколу USB.

Протокол USB високого рівня - це набір команд, що дозволяють опитувати підключені до порту зовнішні пристрої, запитувати у цих пристроїв інформацію про їхній тип, назву, виробника, підтримуваних режимах роботи. Якщо ви працювали з USB, то напевно помітили, що при підключенні до USB накопичувача на жорсткому диску комп'ютера миттєво виявить його і в списку дисків з'явиться ще один диск. При відключенні накопичувача від порту він тут же зникне зі списку. При підключенні

звукового пристрою, в списку звукових пристроїв з'явиться новий пристрій. Так само відбувається і при підключенні камери, сканера, джойстика і т.п. Всі ці типи пристроїв порт USB розпізнає автоматично тому, що в списку стандартних команд USB протоколу є команди, що дозволяють все це ідентифікувати. Тому, пристрій, що підключається до комп'ютера через USB порту повинно вміти підтримувати весь цей протокол. Воно повинно правильно відповідати на всі запити комп'ютера.

Але це не єдина проблема. Більш серйозна проблема - занадто висока швидкість роботи USB інтерфейсу. Це так само накладає певні обмеження і додаткові вимоги. Для підключення мікроконтролера до комп'ютера по каналу USB існує три способи.

1.4 Способи підключення мікроконтролера до комп'ютера

1.4.1 Використання мікроконтролера з вбудованим апаратним модулем USB

Фірма Atmel виробляє кілька видів подібних мікроконтролерів. Наприклад, AT90USB1287 або AT90USB647 [3]. Такий мікроконтролер містить вбудований апаратний USB інтерфейс. Цей інтерфейс бере на себе всю обробку USB протоколу, тому основне ядро мікроконтролера не завантажується цим завданням і може бути зайнято виконанням своєї головної програми. Використання мікросхеми з вбудованим USB каналом - це самий грамотний підхід з точки зору якості і надійності роботи всієї системи. Недолік - велика вартість таких мікроконтролерів (приблизно 25 ... 30 у.о.). Не дивлячись на наявність апаратного USB інтерфейсу, програма, зашита в мікроконтролер, повинна мати відповідну процедуру, для управління і настройки цього інтерфейсу. Створення такого інтерфейсу вимагає досить серйозних знань протоколу USB.

1.4.2 Використання додаткової мікросхеми - перетворювача USB - RS232

Така мікросхема є, наприклад, у продукції фірми Future Technology Devices International Limited (FTDI) і називається FT232RL (рис. 1.5) [4].

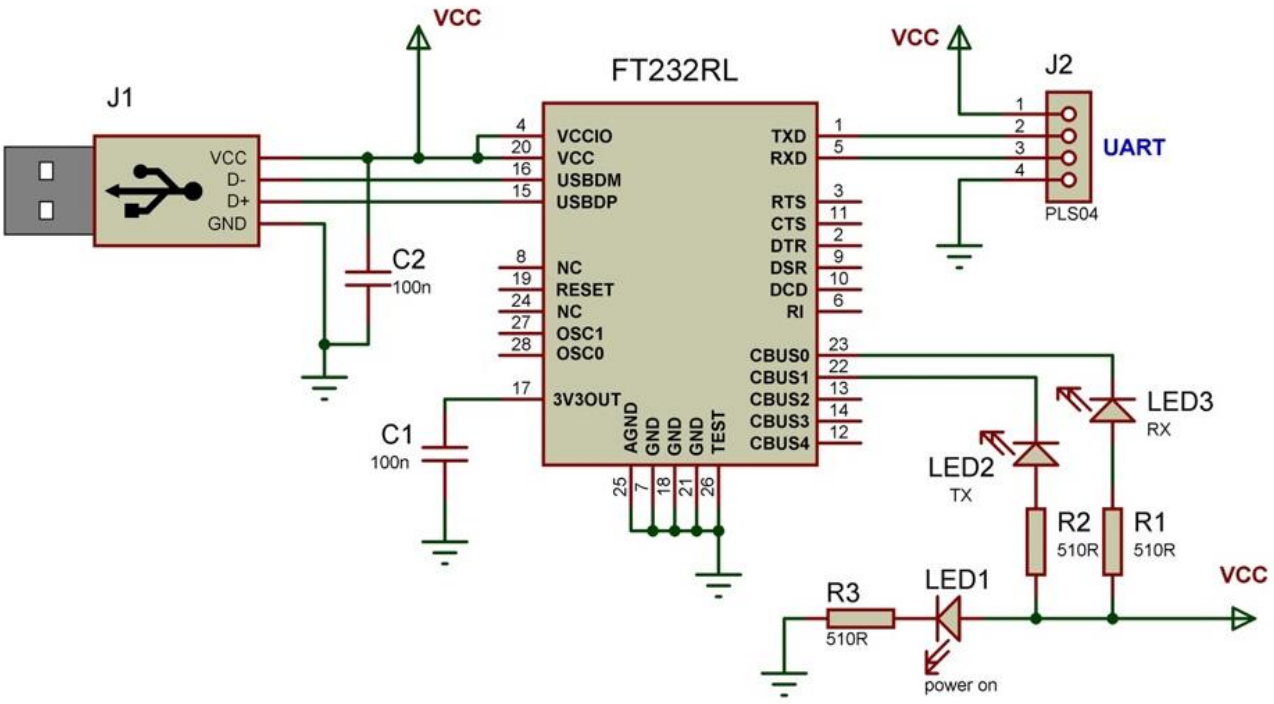


Рисунок 1.5 – Перетворювач USB- RS232

Мікросхема містить апаратний USB інтерфейс і інтерфейс RS232. Основне призначення мікросхеми - перетворення даних, що надходять по USB інтерфейсу в формат RS232 і навпаки. Як вже говорилося вище, більшість сучасних мікроконтролерів мають вбудований послідовний інтерфейс, сумісний з RS232. Тому проблема приєднання вирішується дуже просто. Причому в даному випадку навіть не потрібно піклуватися про сумісність за рівнем сигналів. У мікросхемі FT232RL, так само, як і в мікроконтролері RS232 інтерфейс працює з рівнями сигналів 0В ... +5.

Використання окремої мікросхеми, перетворювача USB-RS232 в даний час є найпоширенішим рішенням розглянутої нами проблеми. По-перше, це спрощує розробку програмного забезпечення, так як робота з RS232

незрівнянно простіше, ніж робота з USB. Крім того, таке рішення оптимально за ціною. Мікросхема FT232RL коштує приблизно 8 у.о.

1.4.3 Використання додаткової мікросхеми - перетворювача USB - FIFO

Ця мікросхема є подальшим вдосконаленням попереднього способу. Прикладом може служити мікросхема FT245R фірми Future Technology Devices International Ltd (рис. 1.6) [5].

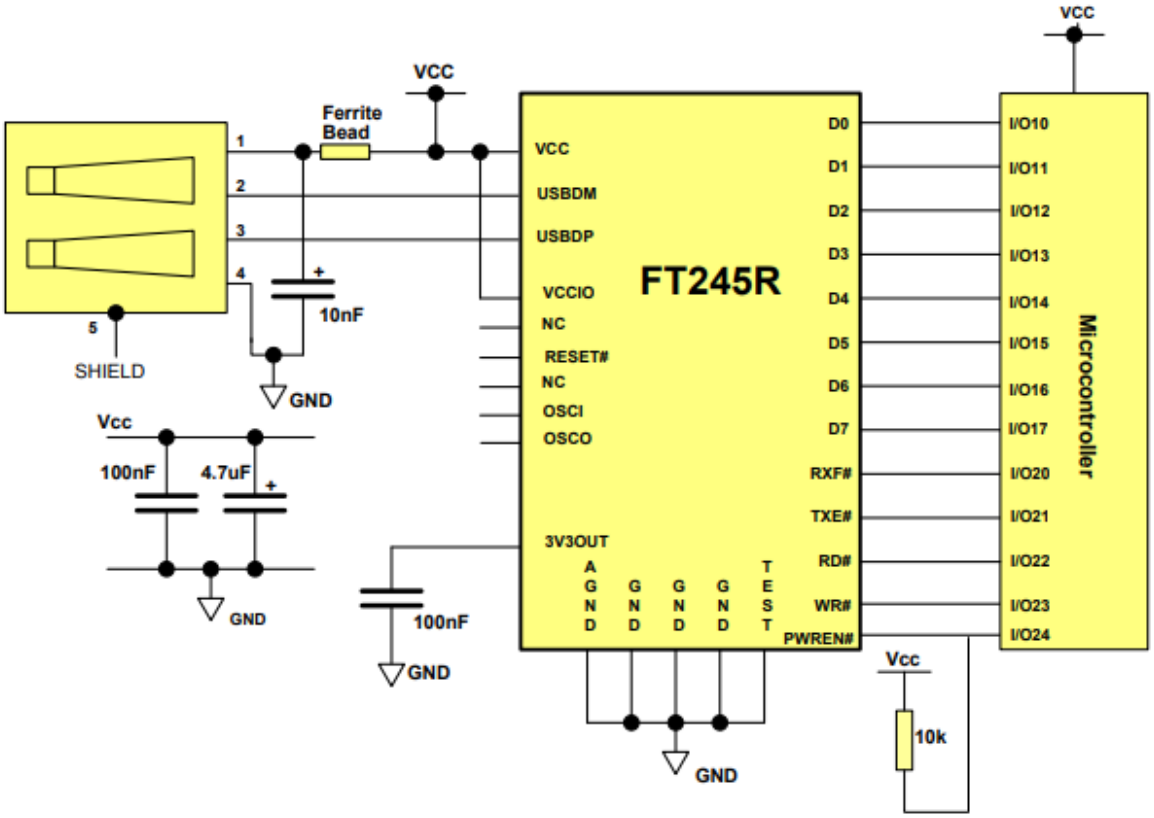


Рисунок 1.6 – Підключення мікросхема FT245R мікроконтролера

Мікросхему зручно застосовувати в тому випадку, коли мікроконтролер не має вбудованого послідовного інтерфейсу, сумісного з RS232. Мікросхема FT245R так само, як і в попередньому випадку, емулює віртуальний COM порт в комп'ютері. Однак дані, надіслані комп'ютером в цей порт записуються всередині мікросхеми в буфер, який працює за

принципом черги. Такий буфер в обчислювальній техніці називається буфер FIFO (скорочення від «First In, First Out»: перший увійшов, перший вийшов). Інформація зберігається в буфері доти, поки не буде зчитана звідти мікроконтролером. Для передачі інформації від мікроконтролера в комп'ютер мікросхема FT232RL має другий FIFO буфер. Комп'ютер зчитує з цього буфера через той же самий віртуальний COM порт. Мікроконтролер обмінюється інформацією з обома FIFO буферами мікросхеми FT232RL за допомогою простої восьмирозрядної шини. сайті. Читати. Детальніше про цей спосіб підключення читайте в спеціальній статті на нашому

1.5 Вимоги щодо програмної та апаратної реалізації послідовної шини USB

Згідно вимог USB2.0 апаратура, що працює з цим інтерфейсом, має підтримувати низькошвидкісний режим режиму USB-зв'язку на швидкості 1.5Мбіт/сек.

Програмна реалізація повинна працювати в складі AVR-мікроконтролерів з об'ємом пам'яті від 2 Кбайт і вище та включати в себе процедури передачі та прийому інформації, процедуру переривання.

Для реалізації апаратної підтримки з використанням МК необхідно тільки кілька зовнішніх компонентів:

– один резистор для детекції низької швидкості USB;

– дільник/стабілізатор напруги з фільтрацією.

1.6 Постановка задачі на розроблення

Необхідно реалізувати наступні функції:

– безпосереднє керування лінією вводу-виводу;

– перетворення USB–RS232;

Для реалізації таких функцій необхідно використати допоміжний регістр EEPROM та застосувати DLL-бібліотеку функцій з використанням мови програмування C++.

1.7 Висновки до розділу 1

У першому розділу дипломного проекту здійснено огляд апаратної реалізації послідовної шини USB з використанням мікроконтролерів.

Розглянуті питання принципу дії послідовної шини USB, варіанти підключення мікроконтролера до персонального комп'ютера, використання мікроконтролера з вбудованими апаратними USB модулями.

Визначені вимоги щодо програмної та апаратної реалізації послідовної шини USB з використанням мікроконтролерів.

1.8 Перелік джерел посилань до розділу 1

1. Интерфейс USB. [Електронний ресурс] – Режим доступу: http://www.gaw.ru/html.cgi/txt/interface/usb/index.htm (дата звернення 18.05.2020).

2. Структура інтерфейсу USB та його топологія. Режим доступу: http://pps.vtc.vn.ua/-8-usb (дата звернення 19.05.2020).

3. Архітектура новітніх мікроконтролерів. Режим доступу: https://acts.kpi.ua/app/uploads/2017/05/ПОСІБНИК-ARM-15-03-периф.pdf (дата звернення 20.05.2020).

4. FT232R USB UART IC Datasheet. Режим доступу: https://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT232R.pdf (дата звернення 22.05.2020).

5. FT245R USB FIFO IC Datasheet USB FIFO IC Datasheet. Режим доступу: https://www.ftdichip.com/Support/Documents/DataSheets/ICs/DS_FT245R.pdf (дата звернення 23.05.2020).

2 АПАРАТНО-ПРОГРАМНА ПІДТРИМКА ШИНИ USB З ВИКОРИСТАННЯМ МІКРОКОНТРОЛЕРІВ AVR

2.1 Елементна база мікроконтролерів Atmel

У 2002 році фірма Atmel почала випуск нових серій 8-розрядних МК на базі ядра AVR. МК AVR на сьогоднішній час діляться на 3 групи: Classic, Mega і Tiny.

Базовий комплект системи команд сімейства Classic, Mega та Tiny МК налічує до 120 команд різноманітного призначення.

Популяризації використання AVR-мікроконтролерів сприяло використання RISC-архітектури, яка характеризуються потужним набором інструкцій, більшість яких виконуються за один машинний цикл. Це означає, що при рівній частоті тактового генератора вони забезпечують продуктивність в 12 (6) раз більше продуктивності попередніх мікроконтролерів на основі CISC-архітектури (наприклад, MCS51).

З іншого боку, в рамках однієї програми із заданим швидкодією, AVR-мікроконтролер може тактуватись в 12 (6) раз меншою тактовою частотою, забезпечуючи таку ж саму швидкодію, але при цьому споживаючи набагато меншу потужність.

Таким чином, AVR-мікроконтролери представляють більш широкі можливості по оптимізації продуктивності/енергоспоживання, що особливо важливо при розробці додатків з батарейним харчуванням.

Мікроконтролери забезпечує продуктивність до 16 млн. оп. в секунду і підтримують флеш-пам'ять програм різної ємності: 1...256 кбайт. AVR-архітектура оптимізована під мову програмування високого рівня С, а більшість представників сімейства megaAVR містять 8-канальний 10-розрядний АЦП, а також сумісний з IEEE 1149.1 інтерфейс JTAG або debugWIRE для вбудованого налагодження.

Для прикладу на рисунку 2.1 наведені технічні характеристики деяких мікроконтролерів ATtiny фірми Atmel [1].

ATtiny2313	1.8-5.5	20	15	2K	128	128	SPI UART	-	1x8bit 1x16bit
ATtiny24	1,8...5,5	20	12	2K	128	128	USI 4xPWM RTC	8x10bit	1x8bit 1x16bit
ATtiny25	2,7...5,5	20	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed
ATtiny25 Automotive	2,7...5,5	16	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed
ATtiny25V	1.8 - 5.5	10	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed
ATtiny26	2.7-5.5	16	16	1K	128	128	SPI UART	11x10bit	2x8bit
ATtiny261	1.8-5.5	20	16	2K	128	128	PWM USI	11x10bit	1x8bit 1x16bit
ATtiny461	1.8-5.5	20	16	4K	256	256	PWM USI	11x10bit	1x8bit 1x16bit
ATtiny28L	1.8-5.5	4	20	2K	-	-	-	-	1x8bit
ATtiny44	1,8...5,5	20	12	4K	256	256	USI 4xPWM RTC	8x10bit	1x8bit 1x16bit

Рисунок 2.1 – Характеристики мікроконтролерів ATtiny фірми Atmel

Крім того, всі мікроконтролери megaAVR з флеш-пам'яттю ємністю 16 кбайт і більше можуть програмуватися через інтерфейс JTAG.

На рисунках 2.2 і 2.3 показана схема підключення мікроконтролера до шини USB. Дані схеми мають специфічне призначення: перетворювач USB–RS232. На них також реалізовані специфічні функції, такі як безпосереднє керування виведенням і читання/запис EEPROM.

На рисунку 2.2 наведена загальна структурна схема МК AVR [1].

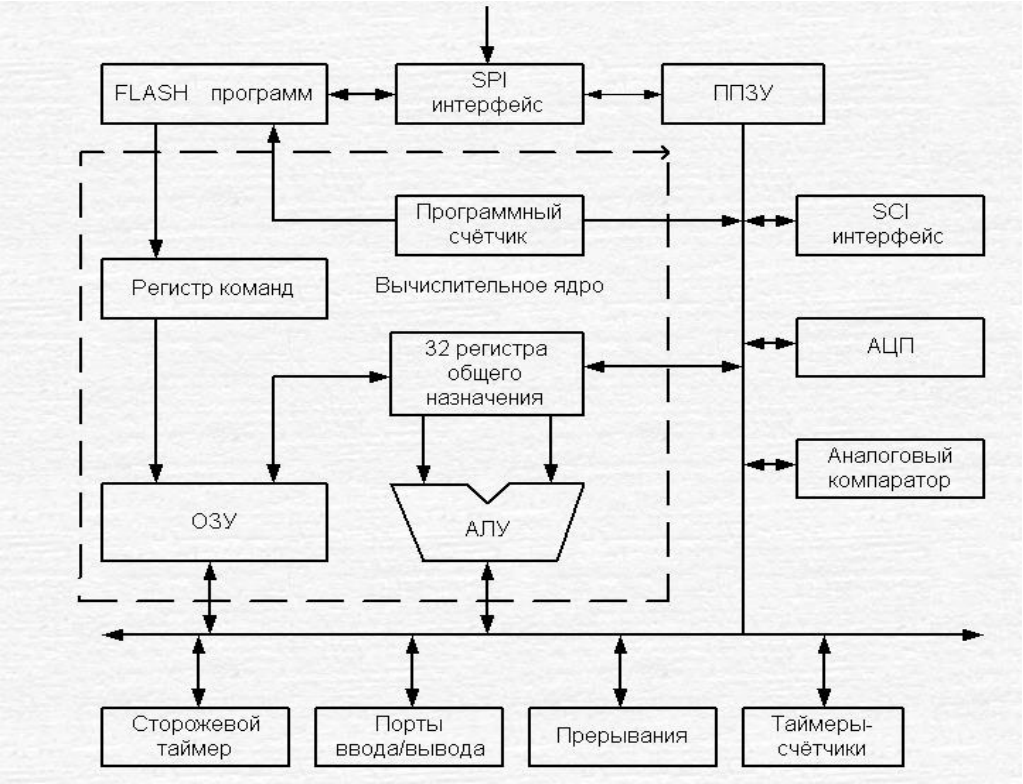


Рисунок 2.2– Загальна структурна схема МК AVR

2.1.1 Апаратне підключення мікроконтролерів до USB порту

Найдешевший спосіб підключення мікроконтролера до комп'ютера – за допомогою USB порту. Таке з`єднання передбачає наявність лише самого контролера. Як лінії USB інтерфейсу використовуються дві будь-які лінії одного з портів введення-виведення контролера. А весь USB протокол реалізується програмним шляхом. При певних хитрощах і за умови використання самого повільного режиму передачі інформації це виявилося цілком можливим.

Однак такий спосіб підключення має масу обмежень.

По-перше, через те, що мікроконтролер працює на межі своєї обчислювальної потужності, доводиться розділяти його роботу на цикл обміну інформацією по USB і цикл виконання основного завдання. Одночасно виконувати те й інше не вийде.

По-друге, нестиковка за рівнями сигналів. Рівень сигналу логічної одиниці порту USB повинен бути в межах 3 ... 3,6. А рівень сигналу порту вводу-виводу мікроконтролера приблизно дорівнює його напрузі живлення, тобто +5. Тому доводиться впроваджувати додаткові технічні заходи. Наприклад, зменшувати напругу живлення мікроконтролера до 3,5 вольт. А це тягне за собою інші незручності, що виникають при підключенні до мікроконтролеру інших зовнішніх елементів.

2.1.2 Підключення мікроконтролерів MAX3420E до USB порту

Мікросхема MAX3420E фірми Maxim (рис. 2.3) містить цифрову логічну та аналогову схему, необхідну для реалізації повношвидкісної USB-периферії, сумісної зі специфікацією USB rev 2.0 [7].

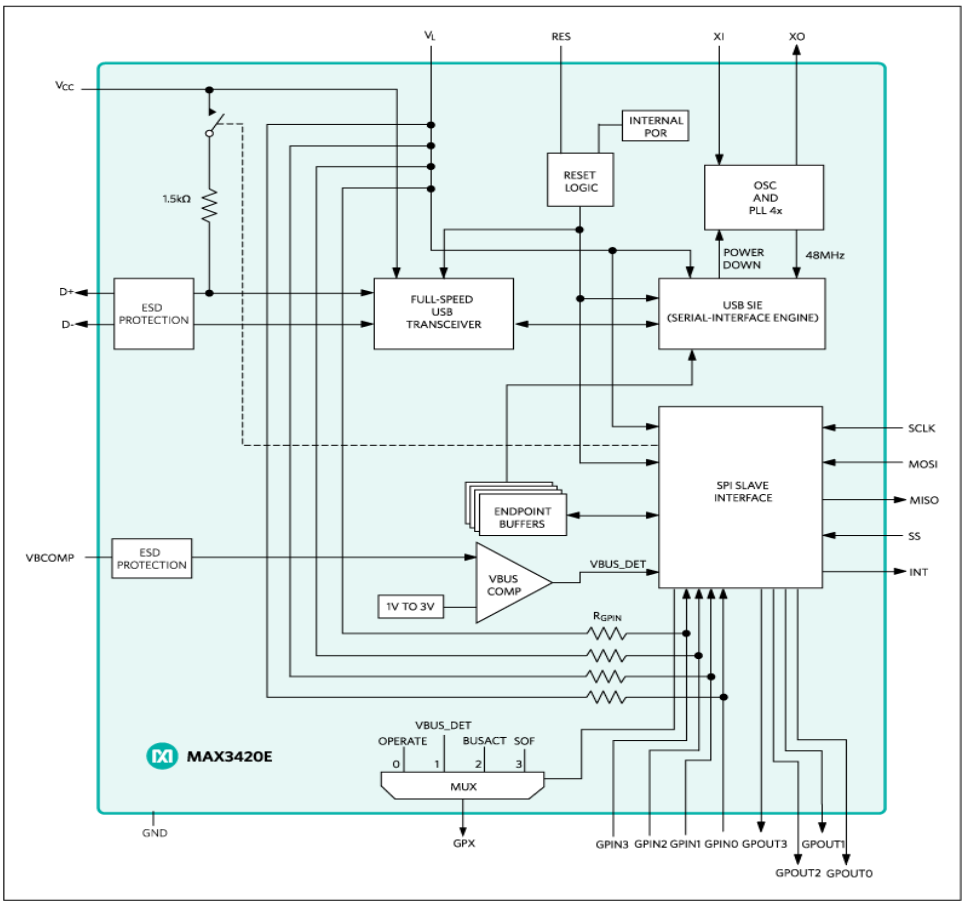


Рисунок 2.3 – Мікросхема MAX3420E фірми Maxim

Вбудований повношвидкісний приймач оснащений захистом ± 15 кВ від статичної напруги та програмованим підключенням та відключенням USB. Внутрішній механізм послідовного інтерфейсу (SIE) обробляє деталі протоколу USB низького рівня, такі як перевірка помилок та спроби шини. MAX3420E працює за допомогою набору регістрів, доступ до якого здійснюється через інтерфейс SPI™, який працює до 26 МГц. Будь-майстер SPI (мікропроцесор, ASIC, DSP тощо) може доповнювати функціональність USB, використовуючи простий 3- або 4-провідний інтерфейс SPI.

Внутрішні перетворювачі дозволяють інтерфейсу SPI працювати при напрузі в системі від 1,71 до 3,6 В. Операції з присвоєнням USB проводяться всередині MAX3420E, наданими після завершення операції ініціалізації, тому SPI-майстру не потрібні таймери, щоб відповідати вимогам синхронізації USB.

MAX3420E включає чотири входи та виходи загального призначення, тому будь-який мікропроцесор, який використовує функції вводу/виводу для реалізації інтерфейсу SPI, може змінити напрямок вводу / виводу та отримати додаткові функції.

MAX3420E працює в розширеному діапазоні температур від -40°C до + 85°C і доступний в 32-контактному пакеті LQFP (7 мм x 7 мм) і 24-контактному 24-контактному пакеті TQFN (4 мм x 4 мм).

2.1.3 Підключення мікроконтролерів ATtiny2313 до USB порту

На рис. 2.4 наведена схема підключення шини USB до мікроконтролера ATtiny2313 в якості перетворювача USB–RS232 з 32-байтним буфером FIFO, 8-розрядним керуванням вводом/виводом та 128-байтним EEPROM [3].

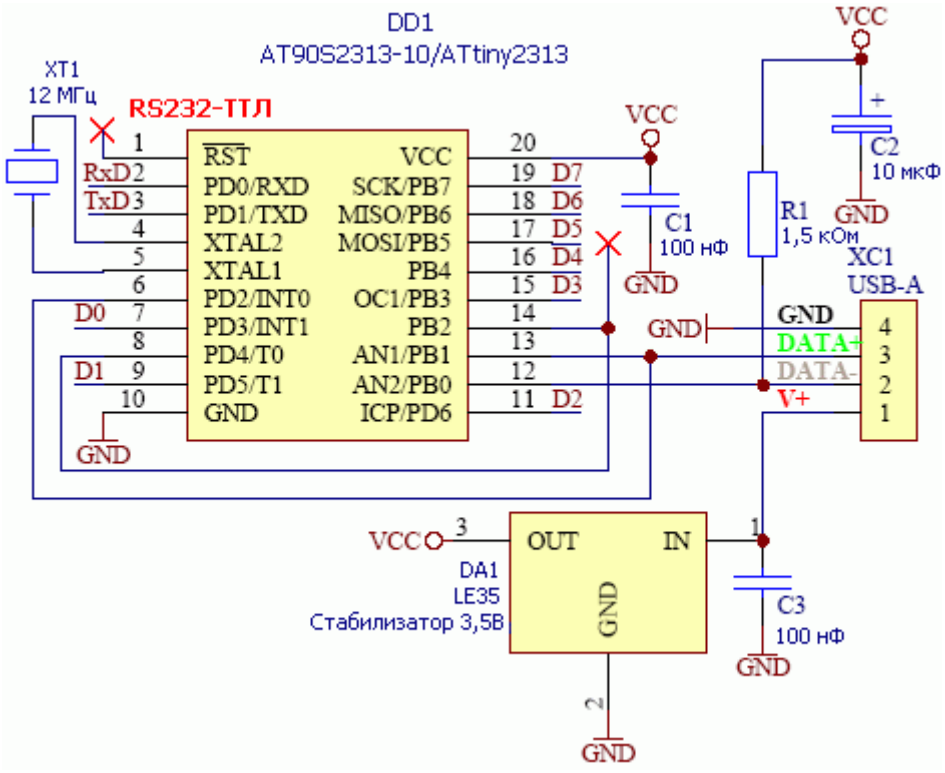


Рисунок 2.4 – Підключення мікроконтролера ATtiny2313 до шини USB

2.1.4 Підключення мікроконтролера ATmega8/48/88/168 до USB порту

На рис. 2.5 наведена схема підключення шини USB с мікроконтролером ATmega8/48/88/168 в якості перетворювача USB–RS232 з 800-байтним буфером FIFO, керуванням вводом-виводом та 512-байтним EEPROM.

Лінії даних USB, DATA– і DATA+, підключені до виводів PB0 і PB1 мікроконтролера AVR. Це з'єднання не можна змінити, тому що в програмі використовуються унікальності ядра AVR для швидкого прийому сигналу: бітовий сигнал захоплюється з ліній даних і молодший розряд (PB0) зсувається вправо в перенос, а потім в приймальний регістр, який накопичує біти, прийнятих з ліній даних. PB1 використовується в якості вхідного сигналу, оскільки у 8-вивідного AT90S2323 даний контакт може використовуватися в якості входу зовнішнього переривання без будь-яких

зовнішніх підключень до INT0. В інших мікроконтролерах AVR необхідно зовнішнє підключення DATA+ виводу INT0, що гарантує незмінність програми при переході між мікроконтролерами [3].

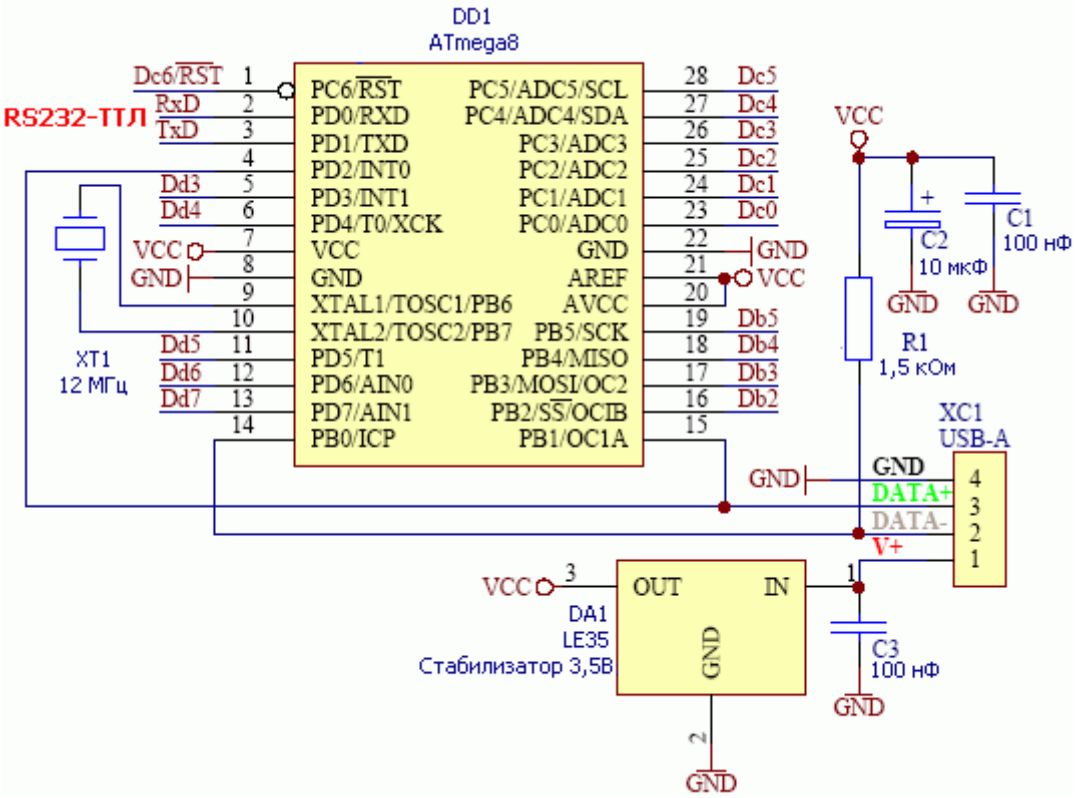


Рисунок 2.5 – Підключення мікроконтролера ATmega8/48/88/168 до USB порту

Для відповідного підключення USB-пристрою і сигналізації мікроконтролер AVR, що працює як низькошвидкісний USB-пристрій, повинен мати підтягуючий резистор 1,5 кОм на лінії DATA–.

Напруга Vcc, що надходить від USB-порту комп'ютера, може змінюватись в діапазоні 4.4...5.25В. Перед його подачею до мікроконтролера і підтягуючого опору 1,5 кОм дана напруга стабілізується на рівні 3.0…3.6В. Тип стабілізатора напруги залежить від рівня навантаження цільової системи. Стабілізатор напруги повинен відноситься до типу стабілізаторів з малим мінімальним перепадом напруги.

У схемах на рисунках 2.2 і 2.3 використовується стабілізатор LE35 з номінальною вихідною напругою 3,5 В. Однак можна використовувати будь-яке інше рішення по стабілізації напруги, якщо воно відповідає вимогам проектованої системи. В деяких випадках може бути цілком прийнятним параметричний стабілізатор на основі стабілітрона.

Інші компоненти необхідні для відповідної роботи мікроконтролера: кварцовий резонатор для синхронізації мікроконтролера і конденсатори для фільтрації живлення.

Таким чином, для отримання завершеного пристрою, який забезпечує зв'язок з комп'ютером через інтерфейс USB, необхідно невелику кількість елементів. Для розширення функціональних можливостей можуть бути додані додаткові компоненти.

Інфрачервоний датчик TSOP1738 може використовуватися для прийому інфрачервоного сигналу. Перетворювач рівня ТТЛ–RS232 MAX232 необхідно додати при необхідності розробки перетворювача інтерфейсів USB-RS232. Для керування світлодіодами або дисплеєм їх необхідно підключити до ліній вводу-виводу безпосередньо або через резистори.

2.2 Синхронізація роботи мікроконтролера з USB портом

На рис. 2.6 наведена синхронізація шини USB [2].

Рисунок 2.6 – Синхронізація шини USB

На рис. 2.7 наведено кодування даних у коді NRZI.

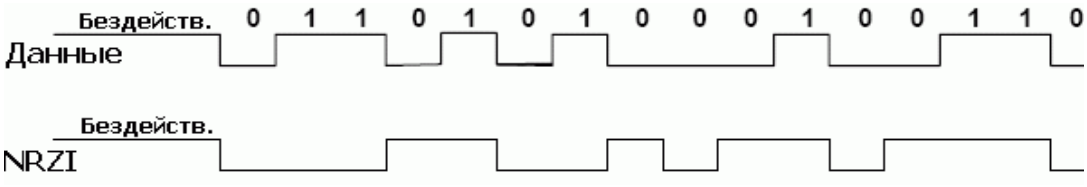


Рисунок 2.7 – Кодування даних у коді NRZI

На рис. 2.8 наведена послідовність кодування даних [2].

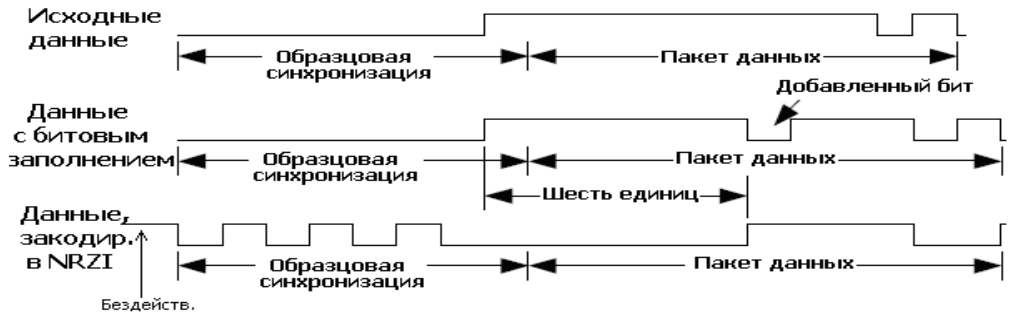


Рисунок 2.8 – Послідовність кодування даних

Закінчення передачі даних виконується за допомогою передачі сигналу "кінець пакету" (EOP). EOP передається шляхом установки низьких рівнів на обох лініях даних DATA+ і DATA–. EOP передається нетривалий час (мінімум два періоди швидкості даних). Після цього виконується наступна транзакція.

На рис. 2.9 наведена часова діаграма сигналу EOP [2]/

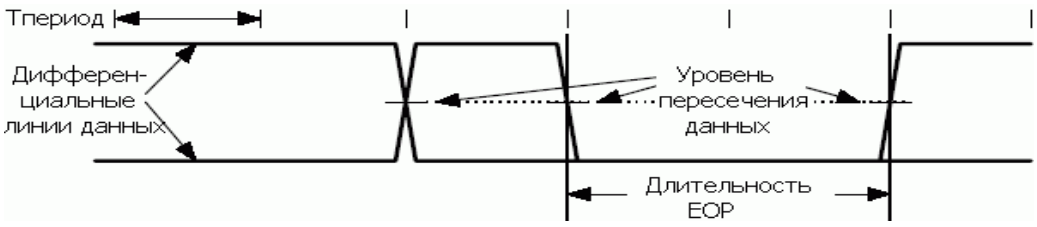


Рисунок 2.9 – Часова діаграма сигналу EOP

Дані, які передаються між зразковою синхронізацією і EOP, закодовані в коді NRZI. Потік даних складається з пакетів, пакет в свою чергу складається з декількох полів: поле синхронізації (зразкова синхронізація), ідентифікатор пакету (PacketID, PID), поле адреси (ADDR), поле кінцевої точки (ENDP), дані і поле циклічного контролю (CRC). USB реалізує чотири типи передачі: передача керування, передача переривання, ізохронна передача і передача потоку. Кожен з цих типів передач специфічний для різних вимог пристрою.

На апаратному рівні периферія підтримує до чотирьох «кінцевих точок». Кожна кінцева точка має свій апаратний регістр управління / стану і свої буфери для прийому / відправки даних. Кінцева точка з індексом 0 («нульова кінцева точка», та, яка потрібна для управління) апаратно «урізана» - розмір буферів прийому / відправки обмежений - 8 байт. Решта кінцеві точки універсальні (можуть бути будь-якого типу) і мають один буфер для відправки даних (розмір буфера - 64 байта) і 2 буфера (2 банки) для читання даних (по 64 байта кожен). Наявність двох банків для прийому даних дозволяє підвищити швидкість обміну: поки програма буде вичитувати дані з одного банку, периферія МК може приймати дані від хоста в інший. Перемикати банки потрібно вручну, через відповідний регістр.

За USB інтерфейсом МК закріплено апаратне переривання. Для того, щоб усередині переривання визначити тип події, яке викликало переривання (джерело), відповідний статусний регістр. Як варіант, можна працювати з USB «за опитуванням», без переривань.

Дані по шині USB передаються пакетами, по кілька байт у кожному. Розмір пакета визначається кожним пристроєм, але його граничний розмір обмежений. Для низькошвидкісних пристроїв максимальний розмір пакета дорівнює 8 байтам. Даний 8-байтний пакет разом з початковим і кінцевим полем повинні бути прийняті в буфер пристрою за одну USB-передачу. В апаратно-реалізованих USB-приймачах різні частини передачі автоматично

дешифруються і пристрій повідомляється, коли всі повідомлення призначено для окремого пристрою.

При програмній реалізації USB-повідомлення дешифрується програмно після прийому в буфер всього повідомлення. Внаслідок цього виникають вимоги і обмеження: пристрій повинен мати буфер для зберігання всього USB-повідомлення, мати інший буфер для USB-передачі (підготовка даних для передачі), а також виконувати адміністрування з дешифруванням і перевіркою повідомлень. Крім того, необхідна програма для виконання швидкого і точного синхронного прийому з фізичних ліній в буфер і передачі з буфера на лінії. Дані можливості обмежуються швидкодією мікроконтролера і розміром пам'яті програм/даних, тому що програма повинна бути ретельно оптимізована. В деяких випадках обчислювальні можливості мікроконтролера дуже близькі до мінімальних вимог, тому, вся програма, як правило, реалізується на Асемблері.

2.3 Реалізація програмного забезпечення

У якості практичного завдання необхідно створити CDC-ACM пристрій. На практиці, за скороченням CDC-ACM стоїть «звичайний» віртуальний COM-порт.

На рівні ОС пристрій буде автоматично розпізнано як послідовний інтерфейс COM-порт в ОС Windows.

USB - інтерфейс, що використовується для підключення периферійних пристроїв. Відповідно, фактично розуміють «головний пристрій» (хост, що керує обмеженим даними через інтерфейс, виконує функцію ініціатора обстеження) і «периферійний пристрій» (клієнт, в процесі обміну даними) «підчиняється» хосту).

На логічному рівні зв’язок із загальними даними відбувається через логічні (віртуальні) канали всередині одного фізичного інтерфейсу USB. Такі канали називають «Кінцеві точки» (EndPoints).

Кінцеві точки (канали) мають 4 види:

Control – даний тип каналу використовує хост для управління периферійним обладнанням. Даний тип каналу використовується для передачі даних.

Bulk – даний тип каналу використовується для обміну даними. Гарантування даних про цільові дані та гарантовану доставку даних для даного типу каналу реалізованої в мікроконтролері. Незважаючи на швидкість передачі даних по такому каналу, обмежений.

Ізохронний – даний тип каналу в основному використовується для спостереження потокових потоків. Цілість і доставка даних не контролюються, за тим, що швидкість значно вище, ніж для Bulk каналу.

Переривання - використовуються для реалізації логічних переривань. Такі «перевірки» використовуються логічно і не пов`язані з апаратними засобами МК або ОС.

Мінімальна реалізація USB-пристрою вимагає наявності всього одного каналу управління Control (так звана «нульова кінцева точка»). Останні типи каналів, як їх кількість визначає розробник пристрою, що знаходиться з функціональних пристроїв.

Однак, існують деякі стандартизовані класи USB пристроїв. Для кожного такого класу кількість каналів, їх типи і призначення встановлено стандартом для даного класу пристроїв.

В роботі створено пристрій класу CDC (communications device class). Використання стандарту, в даному випадку, позбавляє від необхідності писати драйвер для ОС. Як правило, драйвера для стандартних класів пристроїв вже «вшиті» в усі популярні ОС.

В розробленій реалізації створено 3 канали – Control канал для управління і два Bulk канали для передачі даних за напрямом «ПК-МК» і, відповідно, «МК-ПК».

2.4 Програмна реалізація високорівневого API «стека»

Почнемо з високорівневого API нашого «стека». Так як наш пристрій буде працювати як віртуальний COM-порт, API буде складатися з трьох функцій: ініціалізація стека, прийом даних і передача даних. Надалі, програма МК буде використовувати ці функції для обміну даними з ПК [4].

```
#define TRANSFER_STATUS_SUCCESS 0
#define TRANSFER_STATUS_ABORTED 1

typedef void (*TrasferCallback)(BYTE * buffer, WORD size,
void * param, BYTE status);
void USB_Init(void);
DWORD USB_Read(BYTE * data, DWORD length, TrasferCallback
callback, void * param);
DWORD USB_Write(const BYTE * data, DWORD length, TrasferCallback
callback, void * param);
```

Функція USB_Init () ініціалізує відповідну периферію МК, «запускає» стек і завершується після того, як хост закінчить конфігурацію пристрою (присвоїть пристрою адресу і т. Д.).

Функції читання/запису підтримують два режими роботи - синхронний і асинхронний.

У синхронному режимі в функцію передається покажчик на буфер обміну даними (data) і розмір буфера/розмір даних в буфері (length). Параметри, які залишилися (callback, param), передаються в NULL. У такому режимі функція завершиться після того, як вказану кількість байт буде отримано/передано.

Однак, поки функція буде очікувати завершення прийому/передачі даних, будуть «даремно» витрачатися дорогоцінні такти МК.

Більш оптимальним, в деяких випадках, буде робота зі стеком в асинхронному режимі. У цьому випадку, як параметр callback передається покажчик на функцію, яка буде викликана по завершенню прийому/передачі

даних. Сама функція USB_Read () / USB_Write () завершиться одразу після початку обміну даними, не чекаючи завершення обміну.

Визначимо декілька кінцевих точок.

CurrentConfiguration – це глобальна змінна, значення якої (відмінне від 0) призначимо з переривання, коли хост призначить нам «USB конфігурацію».

Як висновок, для того щоб передати дані через кінцеву точку достатньо записати ці дані в апаратний буфер кінцевої точки (але, потрібно пам'ятати про те, що розмір буфера обмежений) і виставити в регістрі управління прапор AT91C_UDP_TXPKTRDY. Після того, як дані будуть передані хосту, відбудеться апаратне переривання - можна передавати наступну порцію.

Читання даних відбувається тоді, коли дозволено переривання для відповідної кінцевої точки. Коли хост передасть дані, периферія МК запише їх у один з внутрішніх буферів (в один з банків) і ініціює апаратне переривання.

Вся логіка роботи USB реалізована в обробнику переривання.

Детально робота програми описана в додатках А,Б,В.

2.5 Використання програмної бібліотеки V-USB

V-USB – назва програмної бібліотеки, що дозволяє отримати підтримку протоколу USB на мікроконтролерах AVR (сімейств Classic, Tiny і Mega фірми Atmel), які не мають апаратної підтримки USB. Авторство бібліотеки належить компанії , яка поширює і просуває V-USB за ліцензією GNU GPL і комерційної ліцензії (вихідні коди бібліотеки вільно доступні) [4, 6].

Стара назва бібліотеки – AVR-USB, але після того як бібліотека стала популярною, назву довелося змінити, щоб не вступати в конфлікт з існуючими іменами Atmel.

Протокол USB реалізований програмно, і швидкодії ядра AVR вистачає тільки на реалізацію стандарту USB 1.1 на низькій швидкості (low-speed). З

цієї причини бібліотека V-USB добре підходить тільки для низькошвидкісних пристроїв введення-виведення (наприклад USB HID). Оскільки вимоги до швидкодії при обробці сигналів USB (D + і D-), дуже жорсткі, то низькорівневий код написаний на асемблері, і підтримується тільки певний ряд тактових частот ядра.

Спочатку тактова частота була тільки 12 МГц, але потім стало можливо використовувати кварцові резонатори на 12, 15, 16, 16.5 і 20 МГц. Мікроконтролери, які мають PLL (ФАПЧ, фазову автопідстроювання частоти) для генерації тактової частоти, можуть використовувати внутрішній RC-генератор (тобто працювати без кварцу), за умови калібрування частоти RC-генератора за сигналом SOF (Start Of Frame) протоколу USB. Високорівневі процедури і функції бібліотеки написані на мові C. Вимоги до мікроконтролеру AVR невисокі - необхідно як мінімум 2 Кбайти пам'яті програм (flash) і 128 байт ОЗУ (RAM). З апаратних ресурсів використовується тільки переривання по зміні сигналу на виводі (зазвичай INT0, приєднаний до сигналу D + шини USB). Таким системним вимогам задовольняють більшість мікроконтролерів сімейства AVR.

Завдяки тому, що разом з бібліотекою поставляються хороші приклади firmware для USB-пристроїв (призначений для користувача клас USB, клас USB HID, миша USB, управління портами мікроконтролера, читання і запис його EEPROM), з'явилося багато корисних розробок, що використовують бібліотеку V-USB - USB - програматори, пристрої введення і виведення, макетні плати, перетворювачі інтерфейсів (наприклад, USB-RS232) і багато іншого (див. посилання). Разом з бібліотекою поставляються також приклади програм для комп'ютера (ПО хосту), які працюють з пристроями на бібліотеці V-USB. Приклади ПО хосту використовують іншу вільну бібліотеку - libusb.

Таким чином, бібліотека V-USB дозволяє непрофесіоналам в програмуванні інтерфейсу USB швидко почати створювати USB-пристрої і писати для них комп'ютерні програми.

Крім того, відсутні фінансові витрати при розробці програмного забезпечення – оскільки на бібліотеку розповсюджується ліцензія GNU.

2.6 Апаратно-програмна реалізація проекту зі створення CDC-232

CDC-232 створює віртуальний COM-порт на PC, навіть якщо він не має реального порту RS-232C (рис. 2.10) [5].

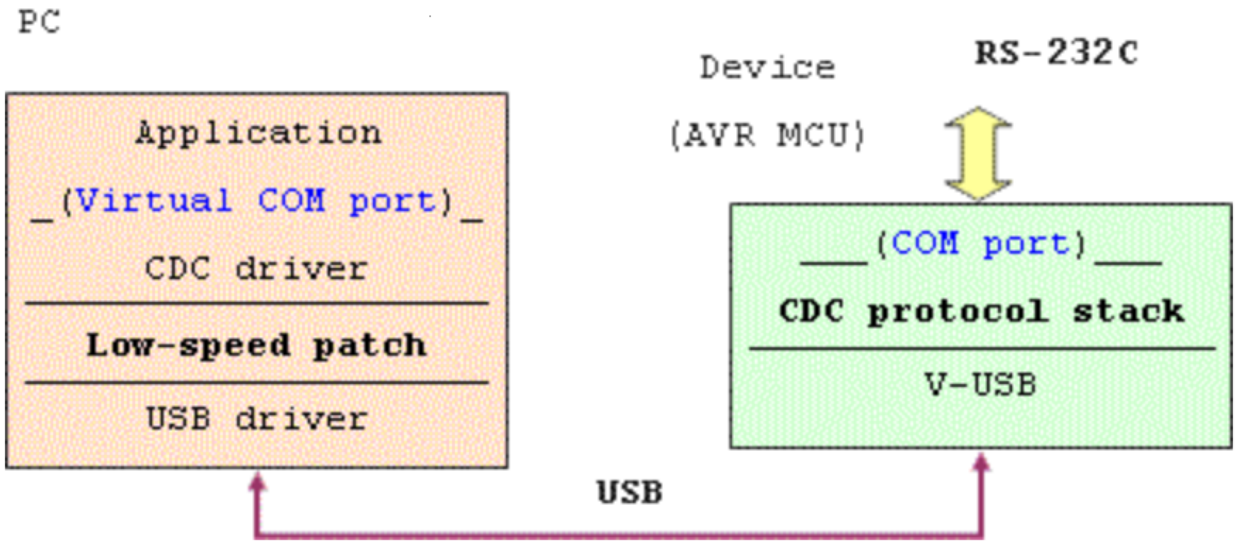


Рисунок 2.10 – Програмна реалізація проекту CDC-232

Це дозволяє проводити обмін даними RS-232C (без сигналів управління) після приєднання пристрою і установки драйвера.

Такі проекти являють собою просту схему на мікроконтролері AVR (ATtiny45/85, ATtiny461, ATtiny2313, ATmega8/48/88 та ряду інших), яка при підключенні до порту USB комп'ютера (далі просто PC) створює віртуальний COM-порт.

В обох проектах використовується безкоштовна бібліотека V-USB, яка дозволяє засобами firmware, прошитого в мікроконтролер AVR, підтримати роботу інтерфейсу USB.

Проект CDC-232 створює на комп'ютері віртуальний COM-порт, через який можна обмінюватися даними з яким-небудь іншим пристроєм, що має низьковольтний RS-232C (наприклад, з мікроконтролером AT89C51 Atmel).

Проект (CDC-IO) теж створює віртуальний COM-порт, але він призначений для управління ресурсами мікроконтролера (регістрами внутрішнього призначення). Тобто посилкою простих текстових команд в консольної програмі можна змінювати стан ніжок мікроконтролера AVR, читати їх стан, управляти таймерами - лічильниками, PWM, читати ADC тощо.

Апаратна реалізація проекту CDC-232 наведена на рис. 2.11 [5].

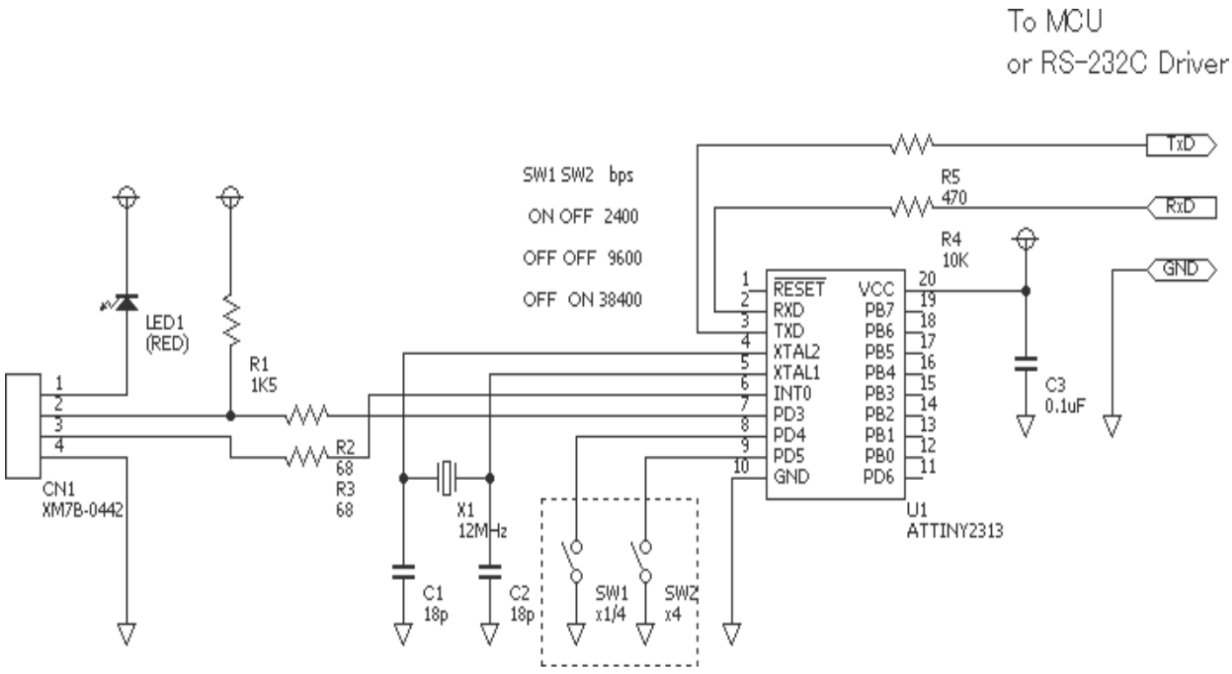


Рисунок 2.11 – Програмна реалізація проекту CDC-232

Наведена схема розрахована на широкий клас мікроконтролерів Atmel, зокрема, ATtiny45/85, ATtiny2313/AT90S2313, ATmega8/48/88/168 та ряду інших.

У всіх цих мікроконтролерах прошивка програмується через ISP. Світлодіод LED знижує напругу USB від 5V до 3.3V і надає її для AVR. Струм споживання близько 10mA, і його мало для живлення інших вузлів.

Коли підключаєтеся до іншого MCU (мікроконтролеру), необхідно з'єднати Gnd і з TxD і RxD хрест-навхрест.

Опір R4 обмежує зворотний струм, коли зовнішній MCU живиться від Vcc 5V (можна R4 не ставити, якщо напруга живлення зовнішнього MCU 3.3V). R5 захищає ніжку TxD, коли замкнута на Gnd.

При використанні MAX232 можна не встановлювати опори R4 і R5, якщо з'єднання відбувається через драйвер RS-232C.

При такій апаратній реалізації необхідно використовувати кварцовий резонатор. Незважаючи на те, що керамічний резонатор в більшості випадків працює добре, але для даного застосування він не підходить, оскільки діапазон відхилення частоти занадто великий, і тому пристрій може працювати нестабільно.

2.7 Висновки до розділу 2

У другому розділі дипломного проекту розглянута та проаналізована апаратно-програмна підтримка шини USB з використанням мікроконтролерів AVR, елементна база мікроконтролерів Atmel, апаратне підключення мікроконтролерів MAX3420E та ATmega8/48/88/168 до USB порту, синхронізація роботи мікроконтролера з USB портом.

Розглянуті програмна реалізація високорівневого API «стека», використання програмної бібліотеки V-USB та апаратно-програмна реалізація проекту зі створення CDC-232 з метою подальшої реалізації проекту

2.8 Перелік посилань до розділу 2

1. Микроконтроллеры AVR семейств Tiny и Mega фирмы Atmel. Режим доступу: http://mega-avr.com.ua/a-v-evstifeev-mikrokontrollery-avr-semejstv-tiny-i-mega-firmy-atmel/ (дата звернення 23.05.2020).

2. USB Peripheral Controller with SPI Interface. Режим доступу: https://www.maximintegrated.com/en/products/interface/controllers-expanders/MAX3420E.html (дата звернення 24.05.2020).

3. ATtiny 2313– 8 битный AVR микроконтроллер с 2 КБ программируемой в системе Flash памяти. http://www.gaw.ru/pdf/Atmel/AVR/attiny2313.pdf (дата звернення 25.05.2020).

4. V-USB. Режим доступу: https://ru.wikipedia.org/wiki/V-USB (дата звернення 25.05.2020).

5. Виртуальный COM-порт CDC-232 на AVR-микроконтроллере. Режим доступу: https://cxem.net/comp/comp140.php (дата звернення 26.05.2020).

6. Бібліотека V-USB. [Електронний ресурс] – Режим доступу: https://www.obdev.at/products/vusb/index.html (дата звернення 24.05.2020).

7. Maxim Integrated. [Електронний ресурс] – Режим доступу: https://www.maximintegrated.com/en/products/interface/controllers-expanders/MAX3420E.html (дата звернення 21.05.2020).

3 ПРОГРАМНЕ СЕРЕДОВИЩА НАЛАШТУВАННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ

3.1 Програмне середовище налаштування

Для програмування МК сімейства AVR існує багато засобів розробки, проте, найбільш популярним, поза сумнівом, є програмний пакет AVR Studio 4. Є ряд причин такої популярності – це безкоштовний програмний пакет, розроблений фірмою ATMEL, який об'єднує в собі текстовий редактор та програмний емулятор МК різних типів. Пакет AVR Studio 4 використовується також спільно з апаратними засобами налаштування фірми ATMEL [1, 2].

Вибір мови програмування залежить від вимог, що пред'являються до самої програми. Якщо потрібна максимальна швидкодія, компактність коду і його надійність, та "ручне" гнучке керування різними елементами МК, то це, звичайно, Асемблер. Якщо ж потрібна простота та комфортність в написанні, а також наочність програмного коду, то це C++, проте в цьому випадку буде деякий програш в продуктивності. Синтаксис Асемблера дуже простий, всі його команди базуються тільки на архітектурі самого МК, тобто, при програмуванні все зводиться до системи команд.

Протокол USB-прийому та дешифрування повністю реалізований програмно. Програма спочатку приймає бітовий потік USB в один USB-пакет у внутрішньому буфері. Початок прийому ініціюється за зовнішнім переривання INT0, яке відповідає за зразкову синхронізацію. У процесі прийому перевіряється тільки сигнал кінця пакета (визначається тільки EOP). Це необхідно з огляду при дуже високій швидкості передачі даних по шині USB. Після успішного прийому програма дешифрує пакети даних та аналізує їх.

На рисунку 3.1 наведено робочі вікна програми AVR STUDIO 4.

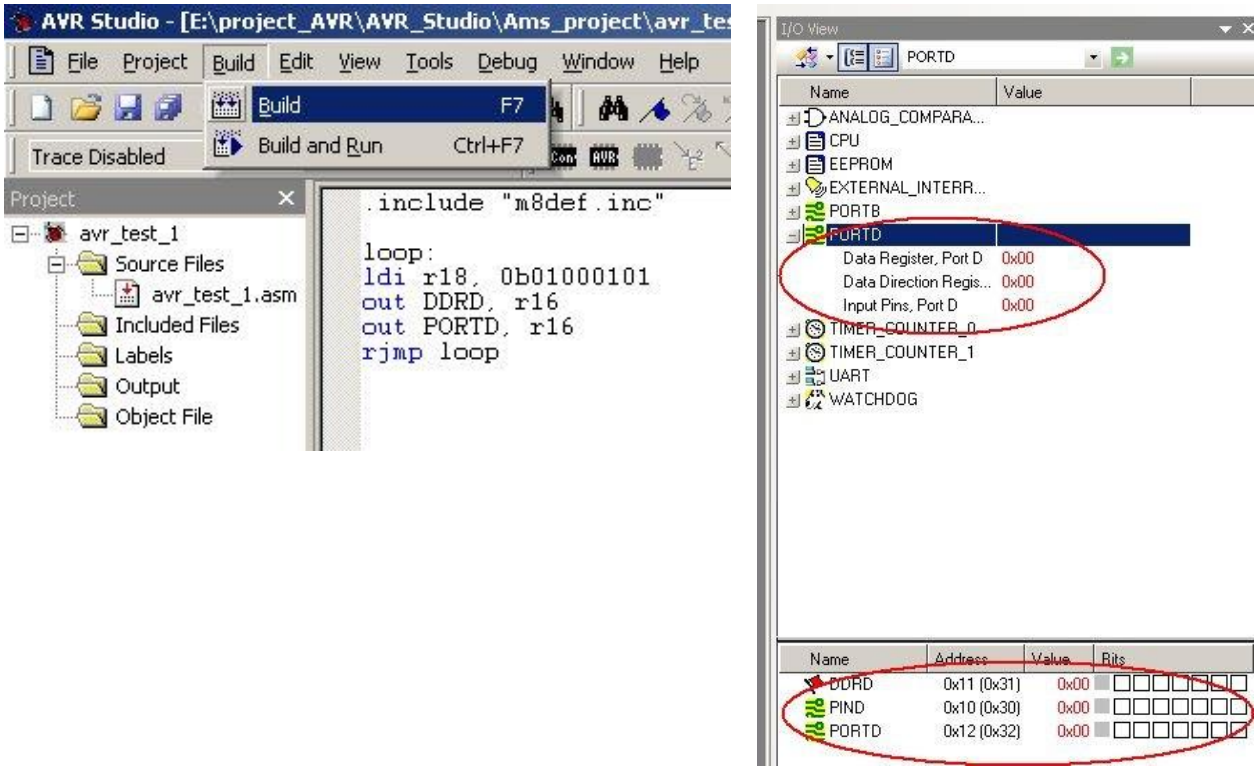


Рисунок 3.1 – Робочі вікна програми AVR STUDIO 4

По-перше, за допомогою адреси аналізується, що прийнятий пакет належить до даного пристрою. Адреса передається при кожній USB-транзакції і, отже, пристрій знає, що передані дані відносяться саме до нього. Дешифрування USB-адреси повинна виконуватися дуже швидко, оскільки, у разі визначення дійсного пакета із заданим адресою, пристрій повинен відповісти комп'ютера підтверджує пакетом ACK. Таким чином, дешифрування адреси є критичною частиною USB-відповіді.

Після прийому бітового потоку отримуємо послідовність, закодовану в коді NRZI, шляхом порозрядного заповнення вхідного буфера. У процесі дешифрування ми спочатку видаляємо порозрядне заповнення, а потім NRZI-кодування. Всі ці зміни виконуються у другому буфері (копія приймального буфера). Після дешифрування поточного пакета може прийматися новий пакет. Для даної точки, швидкість дешифрування не так важлива, тому що

пристрій може затримати свою відповідь. Якщо комп'ютер запитує відповідь в процесі дешифрування, то пристрій повинен відповісти негайно NAK (немає підтвердження), виходячи з чого комп'ютер зрозуміє про неготовність пристрою.

Таким чином, мікроконтролер повинен бути здатний приймати пакети від комп'ютера в процесі дешифрування, визначити чи відноситься транзакція до пристрою, а потім відправити пакет NAK, якщо дешифрування ще знаходиться в процесі виконання. У цьому випадку комп'ютер відправить запит знову.

Мікроконтролер також дешифрує основну USB-транзакцію і виконує запитувану дію; наприклад, відправлення символу по лінії RS232 і очікування завершення транзакції, а також підготовка відповідної відповіді. При виконанні даного процесу пристрій буде перериватися деякими пакетами, що надходять від комп'ютера, зазвичай пакти IN для отримання відповіді від пристрою.

На дані пакети IN пристрій повинен відповісти пакетами NAK. Якщо відповідь готова і пристрій виконав необхідну дію, відповідь має спочатку пройти через функцію обчислення і приєднання CRC, потім виконується NRZI-кодування, а потім порозрядне заповнення. Тепер, коли комп'ютер запитує відповідь, даний бітовий потік передається по лініях даних відповідно до вимог стандарту USB.

Алгоритм прийому інформації USB портом наведена в додатку Е.

Програма розділена на блоки: процедури переривання, процедури дешифрування, USB-прийом, USB-передача, дешифрування запитуваної дії та виконання необхідних дій.

Користувач, при необхідності, може додати власні функції. У коді програми можна знайти приклади, як зробити специфічні для користувача функції. Таким чином, користувач на основі існуючих вбудованих функцій може написати власні розширення. Наприклад, на основі вбудованої функції безпосереднього керування виводу можна додати підтримку TWI.

3.2 Процедура оброблення переривання

Зовнішнє переривання 0 перебуває в активному стані протягом усього часу виконання програми. Дана процедура ініціює прийом даних, послідовно передаються по шині USB. Зовнішнє переривання виникає при появі наростаючого фронту на контакті INT0. Наростаючий фронт вказує на початок зразкової синхронізації (див. рис. 1.4) і ініціює виконання процедури USB-прийому.

Спочатку процес переведення в цифровий код даних повинен бути синхронізовано до середини біта. Це виконується відповідно до зразкової синхронізації. Оскільки тривалість передачі одного біта дорівнює 8 періодам синхронізації XTAL і виникнення переривання може бути затримано (+/– 4 періоди), то синхронізацію до фронту зразкової синхронізації необхідно виконати ретельно. Закінчення передачі зразковою синхронізацією і початок передачі біт даних визначається наступним за зразковою синхронізацією двома бітами з низьким рівнем.

Після цього починається фактична переведення в цифровий код даних. Переведення в цифровий код даних виконується по середині бита. При швидкості передачі даних 1,5 Мбіт/с (1.5МГц) і частоті синхронізації мікроконтролера 12 МГц в розпорядженні є тільки 8 тактів для переведення в цифровий код даних, запису результату в однобайтний буфер, зсуву даних в однобайтному буфері, перевірки на прийом всього байта, запису байта в статичну ОП та визначення закінчення пакета (EOP). Це найбільш критична частина програми; все повинно бути зроблено синхронно і за встановлені часові рамки. Після прийому всього USB-пакета необхідно виконати дешифрування пакета. Спочатку необхідно швидко визначити тип пакету (SETUP, IN, OUT, DATA) і прийняти USB-адресу. Швидке дешифрування повинно виконуватися всередині процедури обробки переривання, тому що необхідно дуже швидко відповісти після прийому USB-пакета (пристрій

повинен відповісти пакетом ACK, якщо прийнятий пакет з адресою пристрою і NAK, якщо прийнятий пакет, адресований пристрою, але відповідь ще не готовий).

В кінці процедури прийому (після відправки пакета підтвердження ACK/NAK) записані в буфер дані повинні бути скопійовані в інший буфер, в якому виконується дешифрування даних. Це необхідно для звільнення приймального буфера для прийому нового пакета.

У процесі прийому дешифрується тип пакета і встановлюється відповідне значення прапора. Даний прапор перевіряється в основному циклі програми і залежно від його значення виконується відповідна дія і готується відповідний відповідь без пред'явлення будь-яких вимог по швидкодії мікроконтролера [3].

Для збереження високої швидкості обробки зовнішнього переривання INT0 воно повинно залишатися активним постійно, навіть при обробці інших переривань (наприклад, переривання по послідовному прийому). Швидкість прийому в процедурі обробки переривання INT0 дуже важлива, тому програму необхідно оптимізувати за швидкодією і часом виконання. Важливою проблемою є оптимізація резервування регістра в процедурах обробки переривань.

3.3 Програма для ПК

Для забезпечення зв'язку з налаштуванням необхідна програмна підтримка на стороні ПК. Програма розділена на три рівні [4]:

1. Драйвер пристрою: використовується для зв'язку на низькому рівні з пристроєм і для інсталяції в операційну систему Windows XP.

2. DLL-бібліотека використовується для інкапсуляції функцій пристрою і зв'язку з драйвером пристрою. DLL спрощує доступ до функцій пристрою з програми користувача. До складу бібліотеки входять деякі функції пристрою і операційної системи (завдання, буфери та ін.)

3. Додаток користувача створює користувальницький інтерфейс для зручного зв'язку між користувачем і пристроєм. Викликаються функції тільки з DLL-бібліотеки.

Спочатку, USB-пристрій підключається до комп'ютерного USB-порту, потім операційна система визначить пристрій і запросить файли драйвера. Даний процес називається інсталяцією пристрою. Для виконання інсталяції необхідно не тільки створити драйвер пристрою, але також інсталяційний скрипт, в якому описується послідовність інсталяції.

Драйвер працює під всіма версіями 32-розрядних операційних систем Windows. Інсталяційний скрипт, записаний в INF-файл, використовується в процесі інсталяції пристрою. В даному INF-файлі описані різні інсталяційні кроки. Файл "AVR309.inf" створений з допомогою текстового редактора. Він запитується операційною системою в процесі інсталяції.

Після ініціації інсталяції файл драйвера копіюється в операційну систему, а потім виконуються необхідні системні зміни. INF-файл гарантує інсталяцію DLL-бібліотеки в системну папку, що гарантує простоту її виклику з різних додатків. Для інсталяції пристрою необхідно три файли: INF-файл "AVR309.inf", драйвер "AVR309.sys" і DLL-бібліотека "AVR309.dll" [5, 6].

3.4 Використання бібліотеки AVR309.dll фірми Atmel

DLL-бібліотека пов'язується з драйвером пристрою і всіма функціями пристрою, реалізованих в цій бібліотеці. Спосіб написання програми кінцевого користувача гранично спрощений. DLL-бібліотека гарантує привілейований доступ до пристрою (впорядковує доступ до пристрою), містить системний буфер для прийому даних через RS232 і створює одну системну задачу для читання буфера даних RS232.

Упорядкування в DLL гарантує, що тільки один додаток/завдання буде пов'язано з пристроєм в даний час. Це необхідно через можливість накладення запитів/відповідей від різних додатків в одне і теж час.

Системний буфер для прийому даних через RS232 гарантує, що прийняті дані будуть розміщені в одному буфері, який є загальним для всіх програм. В такому разі прийняті дані пристроєм будуть відправлятися всіх програм. Не варто турбуватися про те, що додаток прийме не повністю дані через те, що інший додаток до цього прочитував буфер даних.

Для всіх програм існує тільки одна системна задача, і дані будуть періодично запитуватися у пристрою через RS232. В цьому випадку завдання буде зберігати прийняті дані в системний буфер. Наявність тільки одного системного буфера гарантує невелику завантаження ЦПУ (в порівнянні, коли кожен додаток має свою власну задачу) і спрощує збереження даних в системний буфер. Всі функції пристрою визначено в бібліотеці DLL, а їх експорт виконується в зручній формі: не у вигляді номера функції і параметрів, а у вигляді зручного імені функції з параметрами. Деякі функції більш складні внутрішньо, наприклад, функція читання буфера даних RS232. Таким чином, розробники можуть швидко розробляти свої програми, використовуючи тільки інтерфейс DLL. Немає необхідності вивчати функції низького рівня пристрої, тому що DLL-бібліотека розділяє рівень прикладного програмування від апаратного рівня.

3.5 Похибка генерації швидкості універсального асинхронного передавача/приймача

Мікроконтролер використовує тактову частоту 12МГц для здійснення перетворення USB. Недоліком використання такої частоти синхронізації є наявність невеликої похибки при генерації стандартних швидкостей зв'язку. Проте за рахунок високого значення частоти синхронізації похибка мінімальна. Максимально допустима похибка генерації швидкості зв'язку

повинна бути 4%, тому що максимальна похибка дорівнює відношенню тривалості половини біта (0.5) до максимальної тривалості пакета 12 біт = 1 старт-біт + 8 біт даних + 1 біт паритету + 2 стоп-бита.

Таким чином, похибка дорівнює 0.5/12 * 100% = 4.1%.

Функції в DLL автоматично обчислюють похибку і встановлюють таку швидкість зв'язку, відхилення від стандартної якої не перевищує 4%. У разі визначення швидкості, яка не підтримується, виводиться повідомлення про помилку. Однак не рекомендується використовувати швидкості з похибкою більше 2%.

У таблиці 3.1 підсумовуються похибки в генерації стандартних швидкостей при роботі мікроконтролера на частоті 12 МГц.

Таблиця 3.1 – Похибки генерації швидкостей USB і мікроконтролера AVR

Стандартна швидкість USB	Швидкість в AVR	Похибка, %
600	602	0,33
1200	1204	0,33
2400	2408	0,33
4800	4808	0,17
9600	9616	0,17
19200	19230	0,16
28800	28846	0,16
38400	38462	0,16
57600	57692	0,16
115200	115384	0,16

3.6 Вихідний код програми для мікроконтролера AVR

Вихідний код програми USBtoRS232.asm для мікроконтролера ATTiny2313 реалізовано в AVR Studio 4. Програма займає в пам'яті МК AVR 68 кБ (додаткок А).

```
     Фрагмент ініціалізації МК.
     ;* Init program
     reset:    ;initialization of processor and variables to
right values
          ldi  temp0,StackBegin    ;initialization of stack
          out  SPL,temp0
          clr  XH   ;RS232 pointer
          clr  YH   ;USB pointer
          clr  ZH   ;ROM pointer
          sts  RS232BufferBegin+0,YH    ;clear lengths of RS232
code in buffer
          ldi  temp0,RS232BufferBegin+4
          sts  RS232BufferBegin+1,temp0;znuluj ukazovatel citania
          sts  RS232BufferBegin+2,temp0;znuluj ukazovatel zapisu
          clr  RS232BufferFull
     rcall     InitACKBufffer ;initialization of ACK buffer
     call InitNAKBufffer ;initialization of NAK buffer

     rcall     USBReset       ;initialization of USB addresses
          sbi  TSOPpullupPort,TSOPpin   ;set pull-up on TSOP
input
          ldi  temp0,(1<<LEDlsb0)+(1<<LEDlsb1)+(1<<LEDlsb2)
          out  LEDPortLSB,temp0    ;set pull-up on all LED LSB
          out  LEDPortMSB,temp0    ;set pull-up on all LED MSB
          sbi  PORTD,0   ;set pull-up on RxD input
          ldi  temp0,InitBaudRate  ;set UART speed
          out  UBRRL,temp0
          sbi  UCSRB,TXEN     ;enable transmiting of UART
          sbi  UCSRB,RXEN     ;enable receiving of UART
          sbi  UCSRB,RXCIE    ;enable interrupt from receiving of
UART
          ldi  temp0,0x0F          ;INT0 - respond to leading
edge
          out  MCUCR,temp0    ;
          ldi  temp0,1<<INT0  ;enable external interrupt INT0
          out  GIMSK,temp0
```

Бібліотека AVR309.dll [4] використана з сайту фірми Atmel і реалізована на мові C++.

Приклад використання бібліотеки DLL у вигляді додатку користувача представлена у вигляді файлу AVR309USBdemo.exe, що виконується (рис. 3.2) [6].

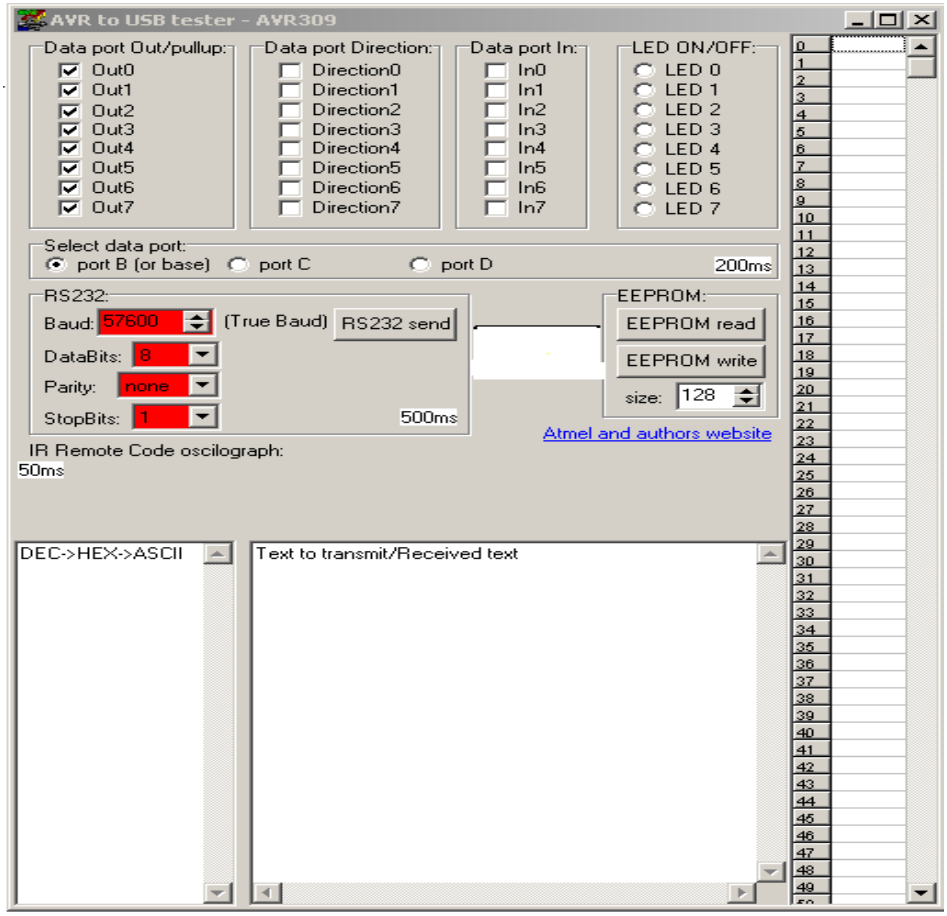


Рисунок 3.2 – Додаток користувача з використанням бібліотеки AVR309.dll

3.7 Висновки до розділу 3

У третьому розділі дипломного проекту проведено розроблення апаратно-програмної підтримки шини USB з використанням мікроконтролерів AVR.

З метою розроблення визначена елементна база мікроконтролерів Atmel, підключення мікроконтролерів MAX3420E та ATmega8/48/88/168 до USB порту.

Розглянуті питання синхронізації роботи мікроконтролера з USB портом, реалізація програмного забезпечення, програмної реалізації

високорівневого API «стека», використання програмної бібліотеки V-USB апаратно-програмної реалізація проекту зі створення CDC-232.

Проведено вибір програмного середовища налаштування та програмна на базі інструментального середовища налаштування AVR Studio для мікроконтролерів Atmel.

Розроблено вихідний код програми для мікроконтролера Atmel AT91C.

3.8 Перелік джерел посилань до розділу 3

1. Трамперт В. AVR–RISC микроконтроллеры.: Пер. с нем. – К.: «МК–Пресс», 2006. – 464 с. , ил.

2. Продукція фірми Atmel. [Електронний ресурс] – Режим доступу: www.atmel.com (дата звернення 24.04.2020).

3. Белов А.В. Самоучитель разработчика устройств на микроконтроллерах AVR / Белов А.В. – СПб.: Наука и техника, 2008. – 544 с.: ил.

4. Програмне забезпечення AVR Studio. [Електронний ресурс] – Режим доступу: https://atmel-studio.software.informer.com/6.0/ (дата звернення 24.04.2020).

5. Objective Development [Електронний ресурс] – Режим доступу: https://www.obdev.at/index.html (дата звернення 28.04.2020).

6. Бібліотека AVR309 [Електронний ресурс] – Режим доступу: http://www.gaw.ru/pdf/Atmel/app/avr/AVR309.pdf (дата звернення 20.05.2020).

4 ОХОРОНА ПРАЦІ

4.1 Загальні питання з охорони праці

Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. В законі України «Про охорону праці» [1] визначається, що охорона праці - це система правових, соціально-економічних, організаційно-технічних, санітарно-гігієнічних і лікувально-профілактичних заходів та засобів, спрямованих на збереження життя, здоров'я і працездатності людини у процесі трудової діяльності.

При роботі з обчислювальною технікою змінюються фізичні і хімічні фактори навколишнього середовища: виникає статична електрика, електромагнітне випромінювання, змінюється температура і вологість, рівень вміст кисню і озону в повітрі. Неправильна організація робочого місця сприяє загальному і локальній напрузі м'язів шиї, тулуба, верхніх кінцівок, викривлення хребта і розвитку остеохондрозу. На всіх підприємствах, в установах, організаціях повинні створюватися безпечні і нешкідливі умови праці. Забезпечення цих умов покладається на власника або уповноважений ним. Умови праці на робочому місці, безпека технологічних процесів, машин, механізмів, устаткування та інших засобів виробництва, стан засобів колективного та індивідуального захисту, що використовуються працівником, а також санітарно-побутові умови повинні відповідати вимогам нормативних актів про охорону праці. Роботодавець повинен впроваджувати сучасні засоби техніки безпеки, які запобігають виробничому травматизмові, і забезпечувати санітарно-гігієнічні умови, що запобігають виникненню професійних захворювань працівників. Він не має права вимагати від працівника виконання роботи, поєднаної з явною небезпекою для життя, а

також в умовах, що не відповідають законодавству про охорону праці. Працівник має право відмовитися від дорученої роботи, якщо створилася виробнича ситуація, небезпечна для його життя чи здоров'я або людей, які його оточують, і навколишнього середовища.

Основним організаційним напрямом у здійсненні управління в сфері охорони праці є усвідомлення пріоритету безпеки праці і підвищення соціальної відповідальності держави і особистої відповідальності працівників.

Державна політика в галузі охорони праці визначається відповідно до Конституції України Верховною Радою України і спрямована на створення належних, безпечних і здорових умов праці, запобігання нещасним випадкам та професійним захворюванням. Відповідно до статті 3 Закону України «Про охорону праці» законодавство про охорону праці складається з Закону, Кодексу законів про працю України, Закону України "Про загальнообов'язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захворювання, які спричинили втрату працездатності" [2] та прийнятих відповідно до них нормативно-правових актів, норм міжнародного договору.

На законодавчому рівні визначено такі пріоритетні напрямки з безпеки праці:

- кожен працівник несе безпосередню відповідальність за порушення зазначених Законом, нормами і правилами вимог;
- напрямки реалізації конституційного права громадян на їх життя і здоров’я в процесі трудової діяльності:
- пріоритет життя і здоров’я працівників по відношенню до результатів виробничої діяльності підприємства;
- повна відповідальність роботодавця за створення належних – безпечних і здорових умов праці;

- соціальний захист працівників, повне відшкодування збитків особам, які потерпіли від нещасних випадків на виробництві та професійних захворювань;
- соціальний захист працівників, повне відшкодування збитків особам, які потерпіли від нещасних випадків на виробництві та професійних захворювань;
- використання світового досвіду організації роботи щодо поліпшення умов і підвищення безпеки праці на основі міжнародної співпраці.

Наявні трудові відносини між працівниками і роботодавцями в Україні за темою дипломного проекту регулюються Кодексом законів про працю (КЗпП) [3] України, відповідно до якого права працюючої людини на охорону праці охороняються всебічно та норми охорони праці неухильно інтегровані до правил внутрішнього розпорядку організації/підприємства.

4.2 Аналіз стану умов праці

Робота над апаратної реалізації шини USB на базі мікроконтролерів Atmel, буде проходити в приміщенні жилого будинку, де проживає студент. Для даної роботи достатньо однієї людини, для якої надано робоче місце зі стаціонарним комп’ютером.

4.2.1 Вимоги до приміщень

Геометричні розміри приміщення зазначені в табл. 4.1.

Таблиця 4.1 – Розміри приміщення

Найменування	**Значення**
Довжина, м	5
Ширина, м	3
Висота, м	2,5
Площа, $м^2$	16
Об’єм, $м^3$	38

Згідно з ДСН 3.3.6.042-99 [4] розмір площі для одного робочого місця оператора персонального комп’ютера має бути не менше 6 кв. м, а об’єм — не менше 20 куб. м. Отже, дане приміщення цілком відповідає зазначеним нормам.

Для забезпечення потрібного рівного освітленості кімната має вікно та систему загального рівномірного освітлення, що встановлена на стелі. Для дотримання вимог пожежної безпеки допущена можливість швидкого виходу з приміщення через вікно з можливістю вентилювання. Також у приміщенні завжди наявна ковдра, яку можна використати при пожежі.

4.2.2 Вимоги до організації місця праці

При порівнянні відповідності характеристик робочого місця з нормативними основними вимогами до організації робочого місця за ДСанПіН 3.3.2.007-98 [5] (табл. 4.2) і відповідними фактичними значеннями для робочого місця, констатуємо повну відповідність.

Таблиця 4.2 – Характеристика робочого місця

Найменування параметра	Фактичне значення	Нормативне значення
Висота робочої поверхні, мм	700	680 ÷ 800
Висота простору для ніг, мм	650	не менше 600
Ширина простору для ніг, мм	630	не менше 500
Глибина простору для ніг, мм	670	не менше 650
Висота поверхні сидіння, мм	420	400 ÷ 500
Ширина сидіння, мм	400	не менше 400
Глибина сидіння, мм	400	не менше 400
Висота поверхні спинки, мм	500	не менше 300
Ширина опорної поверхні спинки, мм	470	не менше 380
Радіус кривини спинки в горизонтальній площині , мм	400	400
Відстань від очей до екрану дисплея, мм	720	700 ÷ 800

4.3 Виробнича санітарія

На підставі аналізу небезпечних та шкідливих факторів при виробництві (експлуатації), пожежної безпеки можуть бути надалі вирішені питання необхідності забезпечення працюючих достатньою кількістю освітлення, вентиляції повітря, організації заземлення, тощо.

4.3.1 Аналіз небезпечних та шкідливих факторів при виробництві (експлуатації) виробу

Аналіз небезпечних та шкідливих виробничих факторів виконується у табличній формі (табл. 4.3). Роботу, пов'язану з ЕОП з ВДТ, у тому числі на тих, які мають робочі місця, обладнані ЕОМ з ВДТ і ПП, виконують із забезпеченням виконання НПАОП 0.00-7.15-18 [6], яке встановлюють вимоги безпеки до обладнання робочих місць, до роботи із застосуванням ЕОМ з

ВДТ і ПП. Переважно роботи за проектами виконують у кабінетах чи інших приміщеннях, де використовують різноманітне електрообладнання, зокрема персональні комп’ютери (ПК) та периферійні пристрої. Основними робочими характеристиками персонального комп'ютера є:

- робоча напруга U=+220В +-5%;
- робочий струм I=2А;
- споживана потужність P=350 Вт.

Таблиця 4.3 – Аналіз небезпечних і шкідливих виробничих факторів

Небезпечні шкідливі виробничі фактори	Джерела факторів	Кількісна оцінка	Нормативні документи
1	2	3	4
Фізичні			
Підвищена або знижена вологість повітря	Експлуатація ЕОМ, принтерів, сканерів або серверного обладнання для роботи	3	[1]
Підвищений рівень напруги електричної мережі, замикання якої може відбуватися через тіло людини	Експлуатація ЕОМ, принтерів, сканерів або серверного обладнання для роботи	2	[2]
Недостатнє освітлення робочої зони	Порушення гігієнічних параметрів виробничого середовища	1	[4]
Психофізіологічні:			
Нервово-психічне перевантаження	- формулювання теми; - пошук інформація про предметну область; - виконання роботи; - оформлення записки.	4	[3]
Фізичні(статичне-сидіння)	Порушення умов праці організації робочого місця(безперервна робота)	2	[2] [3]

4.3.2 Пожежна безпека

Пожежна безпека при застосуванні ЕОМ забезпечується:

1) системою запобігання пожежі;
2) системою протипожежного захисту;
3) організаційно-технічними заходами.

Потенційними джерелами запалювання можуть бути:

1) іскри і дуги короткого замикання;
2) електрична іскра при замиканні і розмиканні ланцюгів;
3) перегріви від тривалого перевантаження;
4) відкритий вогонь і продукти горіння;
5) наявність речовин, нагрітих вище за температуру самозаймання;
6) розрядна статична електрика.

Причинами можливого загоряння і пожежі можуть бути:

1) несправність електроустановки;
2) конструктивні недоліки устаткування;
3) коротке замикання в електричних мережах;
4) запалювання горючих матеріалів, що знаходяться в безпосередній близькості від електроустановки.

4.3.3 Електробезпека

На робочому місці виконуються наступні вимоги електробезпеки: ПК, периферійні пристрої та устаткування для обслуговування, електропроводи і кабелі за виконанням та ступенем захисту відповідають класу зони за ПУЕ (правила улаштування електроустановок), мають апаратуру захисту від струму короткого замикання та інших аварійних режимів. Лінія електромережі для живлення ПК, периферійних пристроїв і устаткування для обслуговування, виконана як окрема групова три-провідна мережа, шляхом прокладання фазового, нульового робочого та нульового захисного провідників. Нульовий захисний провідник використовується для заземлення (занулення) електроприймачів. Штепсельні з'єднання та електророзетки крім контактів фазового та нульового робочого провідників мають спеціальні контакти для підключення нульового захисного провідника. Електромережа штепсельних розеток для живлення персональних ПК, укладено по підлозі поруч зі стінами відповідно до затвердженого плану розміщення обладнання

та технічних характеристик обладнання. Металеві труби та гнучкі металеві рукави заземлені. Захисне заземлення включає в себе заземлюючих пристроїв і провідник, який з'єднує заземлюючий пристрій з обладнанням, яке заземлюється - заземлюючий провідник.

4.4 Гігієнічні вимоги до параметрів виробничого середовища

4.4.1 Освітлення

Розрахунок освітлення.

Для виробничих та адміністративних приміщень світловий коефіцієнт приймається не менше -1/8, в побутових – 1/10:

$$\sqrt{a^2 + b^2} \cdot \text{Sb} = \left(\frac{1}{8} \div \frac{1}{10}\right) \times \text{Sn} \qquad (4.1)$$

де S_b – площа віконних прорізів, м^2;

S_n – площа підлоги, м^2.

$S_n = a \cdot b = 5 \cdot 3 = 15$ м^2

$S_{вік} = 1/8 \cdot 15 = 1{,}875$ м^2

Приймаємо вікно площею S = 1,875 м^2.

Розрахунок штучного освітлення виробляється по коефіцієнтах використання світлового потоку, яким визначається потік, необхідний для створення заданої освітленості при загальному рівномірному освітленні. Розрахунок кількості світильників n виробляється за формулою (4.2):

$$n = \frac{E \cdot S \cdot Z \cdot K}{F \cdot U \cdot M} \qquad (4.2)$$

де n – кількість світильників;

E – нормована освітленість робочої поверхні, визначається нормами – 300 лк;

S – освітлювана площа, м^2 (S = 15 м^2);

Z – поправочний коефіцієнт світильника (1.15 для ламп розжарювання та ДРЛ; 1.1 для люмінесцентних ламп), приймаємо рівним 1.1;

K – коефіцієнт запасу, що враховує зниження освітленості в процесі експлуатації – 1.5;

U – коефіцієнт використання, залежний від типу світильника, показника індексу приміщення і т.п., 0.575M ;

M – число люмінесцентних ламп в світильнику, 1 одиниця;

F – світловий потік - 5400лм (для ЛБ-80).

Підставивши числові значення у формулу (4.2), отримуємо:

$$n = \frac{300 \cdot 15 \cdot 1.15 \cdot 1.5}{5400 \cdot 0.575 \cdot 1} = 2.5 \approx 3$$

Приймаємо освітлювальну установку, яка складається з 3-х світильників, оснащених лампами типу ЛБ (одна – 80 Вт) зі світловим потоком 5400 лм.

4.5 Вентилювання

Здійснюється провітрювання приміщення, в залежності від погодних умов, тривалість повинна бути не менше 10 хв. Найкращий обмін повітря здійснюється при наскрізному провітрюванні.

4.6 Розрахунок захисного заземлення (забезпечення електробезпеки будівлі)

Згідно з класифікацією приміщень за ступенем небезпеки ураження електричним струмом НПАОП 40.1-1.01-97 [7], приміщення в якому проводяться всі роботи відноситься до першого класу (без підвищеної небезпеки). Під час роботи використовуються електроустановки з напругою живлення 36 В, 220 В, та 360 В. Опір контуру заземлення повинен мати не більше 4 Ом.

Послідовність розрахунку.

1) Визначається необхідний опір штучних заземлювачів $R_{шт.з.}$:

$$R_{шт.з.} = \frac{R_{\delta.} \cdot R_{пр.з.}}{R_{пр.з.} - R_{\delta.}} \qquad (4.3)$$

де $R_{пр.з.}$ – опір природних заземлювачів;

Rδ – допустимий опір заземлення.

Якщо природні заземлювачі відсутні, то $R_{шт.з.} = R_{д.}$

Підставивши числові значення у формулу (4.3), отримуємо:

$$R_{шт.з.} = \frac{4 \cdot 40}{40 - 4} \approx 4 \text{ Ом}$$

2) Опір заземлення в значній мірі залежить від питомого опору ґрунту ρ, Ом·м. Приблизне значення питомого опору глини приймаємо ρ = 40 Ом•м (табличне значення).

3) Розрахунковий питомий опір ґрунту, $P_{розр}$, Ом·м, визначається відповідно для вертикальних заземлювачів $P_{розр.в}$, і горизонтальних $P_{розр.г.}$, Ом·м за формулою:

$$P_{розр.} = \psi \cdot \rho \qquad (4.4)$$

де ψ – коефіцієнт сезонності для вертикальних заземлювачів і кліматичної зони з нормальною вологістю землі, приймається для вертикальних заземлювачів $\rho_{розр.в} = 1.7$ і горизонтальних $\rho_{розр.г} = 5.5$ Ом·м.

$P_{розр.в.} = 1.7 \cdot 40 = 68$ Ом/м

$P_{розр.г.} = 5.5 \cdot 40 = 220$ Ом/м

4) Розраховується опір розтікання струму вертикального заземлювача $R_в$, Ом, за (4.5).

$$R_в = \left(\ln\frac{2 \cdot 1в}{dcT} + \frac{1}{2} \cdot \ln\frac{4 \cdot t + 1в}{4 \cdot t - 1в}\right) \quad (4.5)$$

де $l_в$ – довжина вертикального заземлювача (для труб - 2−3 м; $l_в$=3 м);

dст – діаметр стержня (для труб - 0,03−0,05 м; dст=0,05 м);

t – відстань від поверхні землі до середини заземлювача, яка визначається за ф. (4.6);

$$t = h_в + \frac{1_в}{2} \quad (4.6)$$

де $h_в$ – глибина закладання вертикальних заземлювачів (0,8 м); тоді

$t = 0.8 + \frac{3}{2} = 2{,}3$ м

$$R_в = \frac{68}{2 \cdot \pi \cdot 3} \cdot \left(\ln\frac{2 \cdot 3}{0.05} + \frac{1}{2} \cdot \ln\frac{4 \cdot 2.3 + 3}{4 \cdot 2.3 - 3}\right) = 18.5 \text{ Ом}$$

5) Визначається теоретична кількість вертикальних заземлювачів n штук, без урахування коефіцієнта використання ηв:

$$n = \frac{2 \cdot R_в}{R_д} = \frac{2 \cdot 18.5}{4} = 9.25 \quad (4.7)$$

6)Визначається необхідна кількість вертикальних заземлювачів з урахуванням коефіцієнта використання $n_в$:

$$n_в = \frac{2 \cdot R}{R_д \cdot n_в} = \frac{2 \cdot 18.5}{4 \cdot 0.57} = 16.2 \approx 16 \quad (4.8)$$

7)Визначається довжина з'єднувальної стрічки горизонтального заземлювача lc:

$$lc = 1.05 \cdot L_в \cdot (n_в - 1) \quad (4.9)$$

де Lв – відстань між вертикальними заземлювачами, (прийняти за Lв = 3м);

nв – необхідна кількість вертикальних заземлювачів.

$$lc = 1.05 \cdot 3 \cdot (16 - 1) \approx 48м$$

8)Визначається опір розтіканню струму горизонтального заземлювача $R_г$, Ом:

$$R_г = \frac{P_{розр.г}}{2 \cdot \pi \cdot 1c} \cdot \ln\frac{2 \cdot 1^2}{d_{см} \cdot h} \quad (4.10)$$

де $d_{см}$ – еквівалентний діаметр смуги шириною b, $d_{см}$ =0,95b, b = 0,15 м;

h_r – глибина закладання горизонтальних заземлювачів (0.5 м);

1_c – довжина з'єднувальної стрічки горизонтального заземлювача 1_c , м

$$R_г = \frac{220}{2 \cdot \pi \cdot 48} \cdot \ln \frac{2 \cdot 48^2}{0.95 \cdot 0.15 \cdot 0.5} = 8.1 \text{ Ом}$$

9) Визначається коефіцієнт використання горизонтального заземлювача ηс. відповідно до необхідної кількості вертикальних заземлювачів nв.

Коефіцієнт використання з'єднувальної смуги η_c=0,3 (табличне значення).

10) Розраховується результуючий опір заземлювального електроду з урахуванням з'єднувальної смуги:

$$R_{заг} = \frac{R_в . R_г}{R_в \cdot n_c \cdot R_г \cdot n_в \cdot n_в} \leq R_д \qquad (4.11)$$

Висновок: дане захисне заземлення буде забезпечувати електробезпеку будівлі, так як виконується умова: Rзаг <4 Ом, а саме:

$$R_{заг} = \frac{18.5 \cdot 8.1}{18.5 \cdot 0.3 \cdot +8.1 \cdot 16 \cdot 0.57} = 1.9 \leq R_д$$

4.7 Висновки до розділу 4

В результаті проведеної роботи було зроблено аналіз умов праці, шкідливих та небезпечних чинників, з якими стикається робітник. Було визначено параметри і певні характеристики приміщення для роботи над запропонованим проектом написаному в кваліфікаційній роботі, описано, які заходи потрібно зробити для того, щоб дане приміщення відповідало необхідним нормам і було комфортним і безпечним для робітника.

Приведені рекомендації щодо організації робочого місця, а також важливу інформацію щодо пожежної та електробезпеки. Були наведені розміри приміщення та значення температури, вологості й рухливості повітря, необхідна кількість ламп та інші параметри, значення яких впливає на умови праці робітника, а також – наведені інструкції з охорони праці, техніки безпеки при роботі на комп’ютері.

4.8 Перелік джерел посилань до розділу 4

1. Закон України "Про охорону праці". Вводиться в дію Постановою ВР № 2695-XII від 14.10.92, ВВР, 1992, № 49, ст.669. – Режим доступу: https://zakon.rada.gov.ua/laws/show/2694-12

2. Закон України "Про загальнообов'язкове державне соціальне страхування від нещасного випадку на виробництві та професійного захворювання, які спричинили втрату працездатності". Наказ від 21 грудня 2000 року N 2180-III. Режим доступу: https://zakon.rada.gov.ua/laws/show/1105-14

3. Кодекс законів про працю України. Затверджується Законом № 322-VIII від 10.12.71 ВВР, 1971. Режим доступу: https://zakon.rada.gov.ua/laws/show/322-08

4. Санітарні норми мікроклімату виробничих приміщень ДСН 3.3.6.042-99. Постанова N 42 від 01.12.99. Режим доступу: https://zakon.rada.gov.ua/rada/show/va042282-99

5. Державні санітарні правила і норми роботи з візуальними дисплейними терміналами електронно-обчислювальних машин ДСанПІН 3.3.2.007-98. Затверджено Постановою Головного державного санітарного лікаря України 10 грудня 1998 р. N 7. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98

6. НПАОП 0.00-7.15-18 «Вимоги щодо безпеки та захисту здоров’я працівників під час роботи з екранними пристроями». Зареєстровано в

Міністерстві юстиції України 25 квітня 2018 р. за № 508/31960. Режим доступу: www. URL: https://zakon.rada.gov.ua/rada/show/v0007282-98

7. НПАОП 40.1-1.01-97 «Про затвердження Правил безпечної експлуатації електроустановок». Зареєстровано в Міністерстві юстиції України 13 січня 1998 р. за № 11/2451 Режим доступу: https://zakon.rada.gov.ua/laws/show/z0011-98

ВИСНОВКИ

В дипломній роботі виконано розроблення апаратно-програмної підтримки підключення мікроконтролерів фірми Atmel до послідовної шини USB.

Проведено огляд та аналіз апаратних та програмних засобів реалізації послідовної шини USB з використанням мікроконтролерів AVR. За результатами дослідження сформульовані мета та завдання випускної роботи.

Здійснена постановка задачі на розробку апаратної та програмної частини реалізації послідовної шини USB. Визначені шляхи реалізації поставленого завдання та елементна база з використанням мікроконтролерів AVR фірми Atmel.

У якості реалізації апаратної частини обрано мікроконтролера AT91C фірми Atmel для застосування в системах передачі інформації. Програмування пам’яті мікроконтролера проведено в інструментальному середовищі налаштування AVR Studio 4.

В розробленому пристрої реалізовані наступні функції:

– безпосереднє керування лінією вводу-виводу;

– перетворення USB – RS232.

Для реалізації таких функцій використана DLL-бібліотека AVR309.dll фірми Atmel.

Розроблені заходи щодо охорони праці в умовах виробництва.

Результати роботи та запропоновані рішення можуть бути використані у навчальному процесі кафедри комп’ютерних наук та інженерії при вивченні дисципліни «Цифрова схемотехніка».

ДОДАТОК А

Функція ініціалізації

```
     void USB_Init(void)
     {
        // Устанавливаем делитель и включаем тактирование USB
периферии
        AT91C_BASE_CKGR->CKGR_PLLR |= AT91C_CKGR_USBDIV_1;
        AT91F_PMC_CfgSysClkEnableReg(AT91C_BASE_PMC,
AT91C_PMC_UDP);
        AT91F_PMC_EnablePeriphClock(AT91C_BASE_PMC, 1 <<
AT91C_ID_UDP);
        //Сбрасываем номер текущей конфигурации
        CurrentConfiguration = 0;
        //Сбрасываем все КТ
        memset(Endpoint, 0x00, sizeof(Endpoint));
        EndpointReset(AT91C_EP_CFG, EP_TYPE_CFG,
EP_CFG_BUFFER_SIZE);
        EndpointReset(AT91C_EP_OUT, EP_TYPE_BULK,
EP_BUFFER_SIZE);
        EndpointReset(AT91C_EP_IN, EP_TYPE_BULK,
EP_BUFFER_SIZE);
        //Разрешаем прерывания для "нулевой конечной точки"
        EndpointEnableInterrupt(EndpointCfg);
//Устанавливаем обработчик и разрешаем прерывания USB
контроллера
        AT91F_AIC_ConfigureItH(AT91C_BASE_AIC, AT91C_ID_UDP,
AT91C_AIC_PRIOR_LOWEST, AT91C_AIC_SRCTYPE_INT_HIGH_LEVEL,
&USB_IrqHandler);
        AT91F_AIC_EnableIt(AT91C_BASE_AIC, AT91C_ID_UDP);
//Ждем, пока хост не завершит конфигурацию
        while(!CurrentConfiguration);
}
```

ДОДАТОК Б

Функція читання/запису

```
     DWORD _USB_Write(volatile UsbEndpoint * endpoint, const BYTE
* pData, DWORD length, TrasferCallback callback, void * param) {
        //Заполняем структуру КТ
        endpoint->Buffer = (BYTE *) pData;
        endpoint->Size = length;
        endpoint->BytesReady = 0;
        endpoint->Callback = callback;
        endpoint->Param = param;
        endpoint->Status = EP_STATUS_WTITE;
        //Ждем, пока КТ будет готова к передаче
        while(*endpoint->CSR & AT91C_UDP_TXPKTRDY);
        //Записываем блок данных в аппаратный буфер КТ
        DWORD cpt = MIN(endpoint->Size - endpoint->BytesReady,
endpoint->MaxSize);
        while (cpt--) *endpoint->FDR = endpoint-
>Buffer[endpoint->BytesReady++];
        //Устанавливаем флаг "данные готовы к отправке"
        EndpointSetFlag((UsbEndpoint *) endpoint,
AT91C_UDP_TXPKTRDY);
        if(endpoint->Type != EP_TYPE_CFG) {
                //Разрешаем прерывания для данной КТ
                EndpointEnableInterrupt((UsbEndpoint *)
endpoint);
                //Для синхронного режима - ожидаем завершения
отправки
                if(!endpoint->Callback) while(endpoint->Status
== EP_STATUS_WTITE);
        }
        return endpoint->BytesReady;
}
DWORD _USB_Read(volatile UsbEndpoint * endpoint, BYTE * pData,
DWORD length, TrasferCallback callback, void * param) {
        //Заполняем структуру КТ
        endpoint->Buffer = pData;
        endpoint->Size = length;
        endpoint->BytesReady = 0;
        endpoint->Callback = callback;
        endpoint->Param = param;
        endpoint->Status = EP_STATUS_READ;
        if(endpoint->Type != EP_TYPE_CFG) {
//Разрешаем прерывания для данной КТ
       EndpointEnableInterrupt((UsbEndpoint *) endpoint);
//Для синхронного режима - ожидаем завершения
        if(!endpoint->Callback) while(endpoint->Status ==
EP_STATUS_READ);
        }
     return endpoint->BytesReady;
```

ДОДАТОК В

Робота зі стеком

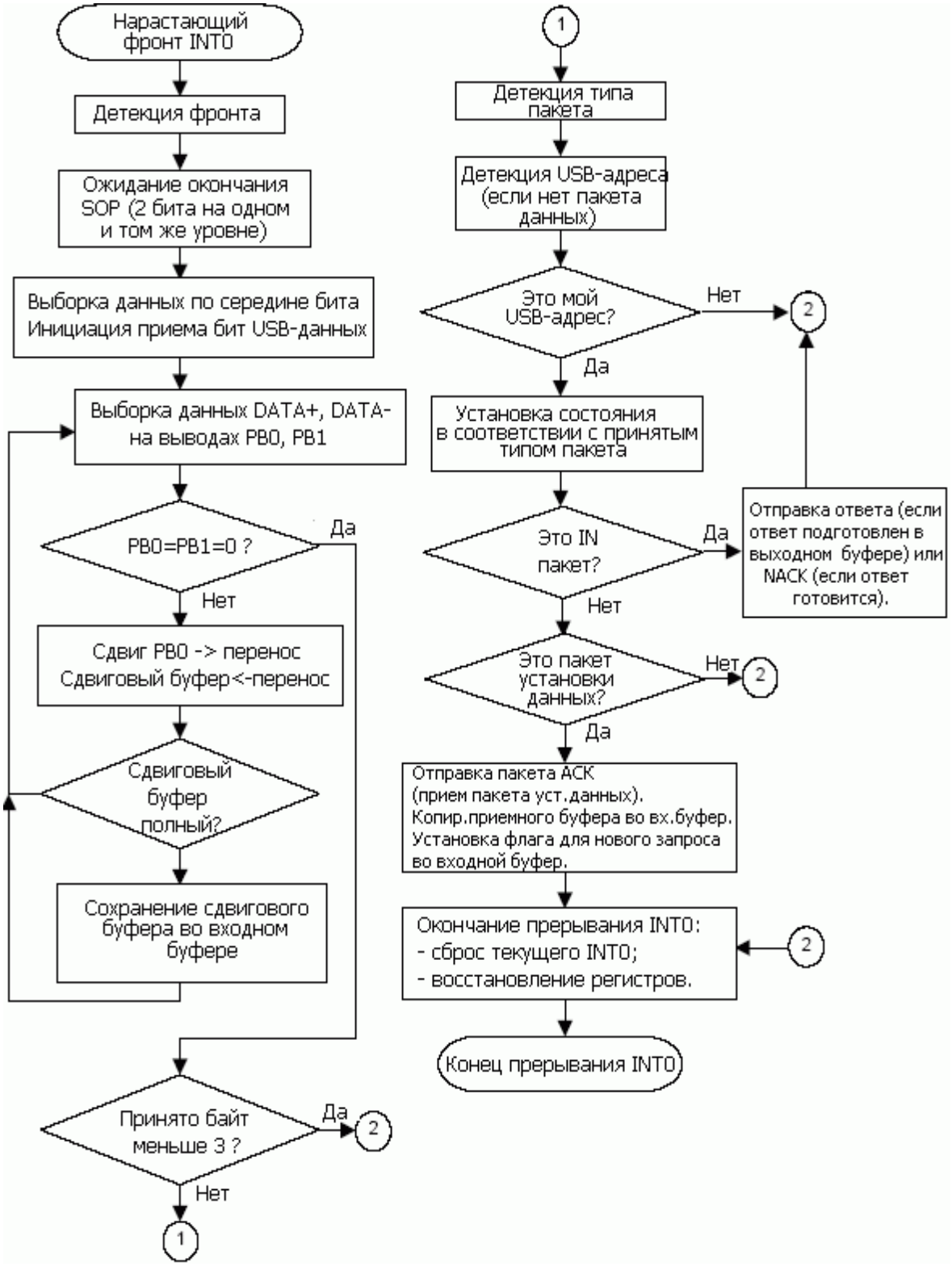
```
     //адреса конечных точек
#define AT91C_EP_CFG 0
#define AT91C_EP_OUT 1
#define AT91C_EP_IN  2
//типы конечных точек
#define EP_TYPE_CFG     0
#define EP_TYPE_BULK    1
#define EP_TYPE_INT     2
     //размеры аппаратного буфера для разных конечных точек
#define EP_BUFFER_SIZE           64
#define EP_CFG_BUFFER_SIZE  8
//логические состояния конечных точек
#define EP_STATUS_IDLE  0
#define EP_STATUS_READ  1
#define EP_STATUS_WTITE 2
     //Так як вся реалізація буде здійснюватись навколо кінцевих
точок - //об'єднаємо всі змінні логічно пов'язані з кінцевою
точкою в структуру і //напишемо кілька простих inline функцій
для базових операцій з кінцевою //точкою.
     typedef struct {
        BYTE Type;              //тип конечной точки
        BYTE MaxSize;           //размер буфера конечной точки
        BYTE * Buffer;          //указатель на буфер для
приема/отправки данных
        WORD Size;              //размер буфера/длина данных в
буфере
        WORD BytesReady;        //количество обработанных
//(полученных/отправленных) байт
        WORD Bank;              //номер текущего банка для
приема данных
        TrasferCallback Callback;       //указатель на Callback-
функцию
     void * Param;           //дополнительные (пользовательские)
параметры //Callback-функции
        volatile BYTE Status;   //текущее логическое состояние
конечной точки
        BYTE InterruptMask;     //маска для
разрешения/запрещения прерываний конечной точки
        AT91_REG * CSR;  //указатель на аппаратный регистр
управления конечной точки
        AT91_REG * FDR;         //указатель на аппаратный
регистр данных конечной точки
} UsbEndpoint;
//Конечные точки
static UsbEndpoint Endpoint[4];
static UsbEndpoint * EndpointCfg        = &Endpoint[0];
static UsbEndpoint * EndpointOut        = &Endpoint[1];
static UsbEndpoint * EndpointIn         = &Endpoint[2];
```

```
//--- Сброс конечной точки (КТ)
static void EndpointReset(BYTE ep, BYTE Type, BYTE MaxSize) {
        //Если КТ находится в процессе асинхронного
чтения/записи нужно сообщить приложению, что передача данных
прервана
        if(Endpoint[ep].Status !=  EP_STATUS_IDLE &&
Endpoint[ep].Callback) Endpoint[ep].Callback(NULL, 0, NULL,
TRANSFER_STATUS_ABORTED);
        Endpoint[ep].Type = Type;
        Endpoint[ep].MaxSize = MaxSize;
        //Сбрасываем все переменные в начальное значение
        Endpoint[ep].Buffer = NULL;
        Endpoint[ep].Size = 0;
        Endpoint[ep].ByteReady = 0;
        Endpoint[ep].Bank = AT91C_UDP_RX_DATA_BK0;
        Endpoint[ep].Callback = NULL;
        Endpoint[ep].Param = NULL;
        Endpoint[ep].Status = EP_STATUS_IDLE;
        //Сохраняем в структуре указатели на аппаратные регистры
для данной КТ
        Endpoint[ep].InterruptMask = 1 << ep;
        Endpoint[ep].CSR = &AT91C_BASE_UDP->UDP_CSR[ep];
        Endpoint[ep].FDR = &AT91C_BASE_UDP-
>UDP_FDR[ep];        //Аппаратно "передергиваем" КТ
        AT91C_BASE_UDP->UDP_RSTEP |= (1 << ep);
        AT91C_BASE_UDP->UDP_RSTEP &= ~(1 << ep);
}
//--- Установить биты по маске в аппаратном регистре управления
КТ
__inline void EndpointSetFlag(UsbEndpoint * endpoint, DWORD
flag) {
        *endpoint->CSR = *endpoint->CSR | flag;
        while((*endpoint->CSR & flag) == 0);
}
//--- Сбросить  биты по маске в аппартном регистре управления КТ
__inline void EndpointClearFlag(UsbEndpoint * endpoint, DWORD
flag) {
        *endpoint->CSR = *endpoint->CSR & ~(flag);
        while((*endpoint->CSR & flag) != 0);
}
//--- Запретить прерывания для данной КТ
__inline void EndpointDisableInterrupt(UsbEndpoint * endpoint) {
        AT91C_BASE_UDP->UDP_IDR = endpoint->InterruptMask;
}
//--- Разрешить прерывания для данной КТ
__inline void EndpointEnableInterrupt(UsbEndpoint * endpoint) {
        AT91C_BASE_UDP->UDP_IER = endpoint->InterruptMask;
}
//--- Прочитать один байт данных из внутреннего буфера КТ
__inline BYTE EndpointGetData(UsbEndpoint * endpoint) {
        return *endpoint->FDR;
}
//--- Записать один байт данных во внутренний буфер КТ
```

```
__inline void EndpointPutData(UsbEndpoint * endpoint, BYTE data)
{
        *endpoint->FDR = data;
      }
//--- Прочитать аппаратный регистр статуса КТ
__inline DWORD EndpointGetCSR(UsbEndpoint * endpoint) {
        return *endpoint->CSR;
}
//--- Сбросить флаг "данные приняты" и переключить банк
приемника
__inline void EndpointClearRxFlag(UsbEndpoint * endpoint) {
        if(endpoint->Type == EP_TYPE_CFG) {
                EndpointClearFlag(endpoint,
AT91C_UDP_RX_DATA_BK0);
        } else {
                EndpointClearFlag(endpoint, endpoint->Bank);
                endpoint->Bank = (endpoint->Bank ==
AT91C_UDP_RX_DATA_BK0) ? AT91C_UDP_RX_DATA_BK1 :
AT91C_UDP_RX_DATA_BK0;
        }
}
//--- Прием/передача завершены: вызвать Callback в асинхронном
режиме, сменить логическое состояние КТ на EP_STATUS_IDLE
__inline void EndpointEndOfTransfer(UsbEndpoint * endpoint, BYTE
status) {
        if(endpoint->Callback) endpoint->Callback(endpoint-
>Buffer, endpoint->BytesReady, endpoint->Param, status);
        endpoint->Status = EP_STATUS_IDLE;
```

ДОДАТОК Д

Алгоритм процедури прийому



ДОДАТОК Е

Презентація

Міністерство освіти і науки України
Східноукраїнський національний університет ім. Володимира Даля

«Апаратно-програмні засоби підключення мікроконтролерів AVR до USB»

Студент гр. КІ – 16 д Прядченко В. С.
Керівник Кардашук В. С.

1

Актуальність теми

- Універсальна послідовна шина (USB) стала надзвичайно популярною за рахунок надання ряду зручностей кінцевим користувачам, наприклад, функція "Plug and Play", яка дозволяє ідентифікувати підключений пристрій без необхідності рестарту комп'ютера.
- USB (Universal Serial Bus – універсальна послідовна шина) є промисловим стандартом розширення архітектури ПК, орієнтованим на інтеграцію з телефонією і пристроями побутової електроніки.
- Для розробників інтегрувати USB-інтерфейс у свої проекти виявилося більш складним у порівнянні, наприклад, з інтерфейсом RS232. Крім цього, на стороні ПК також повинен бути передбачений спеціальний драйвер пристрою.
- Одне з питань, що викликають підвищений інтерес у початківців (та й у досвідчених) конструкторів мікропроцесорних систем є питання взаємодії мікроконтролера з комп'ютером.
-
- Мета дипломного проекту – підключення мікроконтролерів AVR фірми ATMEL до USB порт комп'ютера.

2

Вимоги щодо програмної та апаратної реалізації послідовної шини USB

- Згідно вимог USB2.0 апаратура, що працює з інтерфейсом USB, має підтримувати низькошвидкісний режим режиму зв'язку на швидкості 1.5Мбіт/сек.
- Програмна реалізація повинна працювати в складі AVR-мікроконтролерів з об'ємом пам'яті від 2 кбайт і вище та включати в себе процедури передачі та прийому інформації, процедуру переривання.

- Для реалізації апаратної підтримки з використанням МК необхідно тільки кілька зовнішніх компонентів:
- – один резистор для детекції низької швидкості USB;
- – дільник/стабілізатор напруги з фільтрацією.

3

Для виконання поставленої задачі в дипломній роботі необхідно:

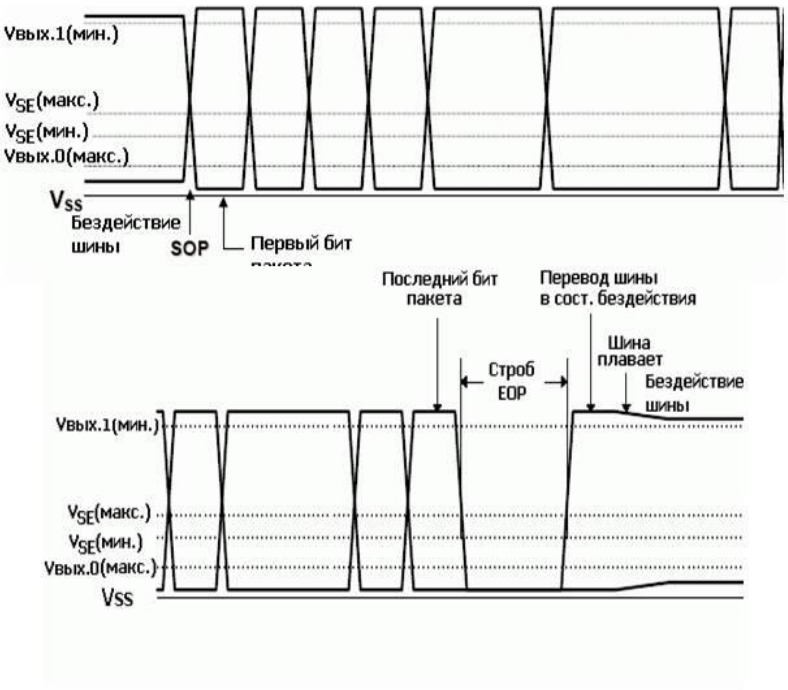
- провести аналіз та розробити апаратні та програмні засоби реалізації підключення мікроконтролерів Atmel до USB порту.

Реалізувати наступні функції:

- – безпосереднє керування лінією вводу-виводу USB;
- – перетворення USB–RS232;
- Для реалізації таких функцій необхідно використати допоміжний регістр EEPROM та застосувати DLL-бібліотеку функцій з використанням мови програмування C++.

4

Рівні напруг при передачі пакетів по шині USB



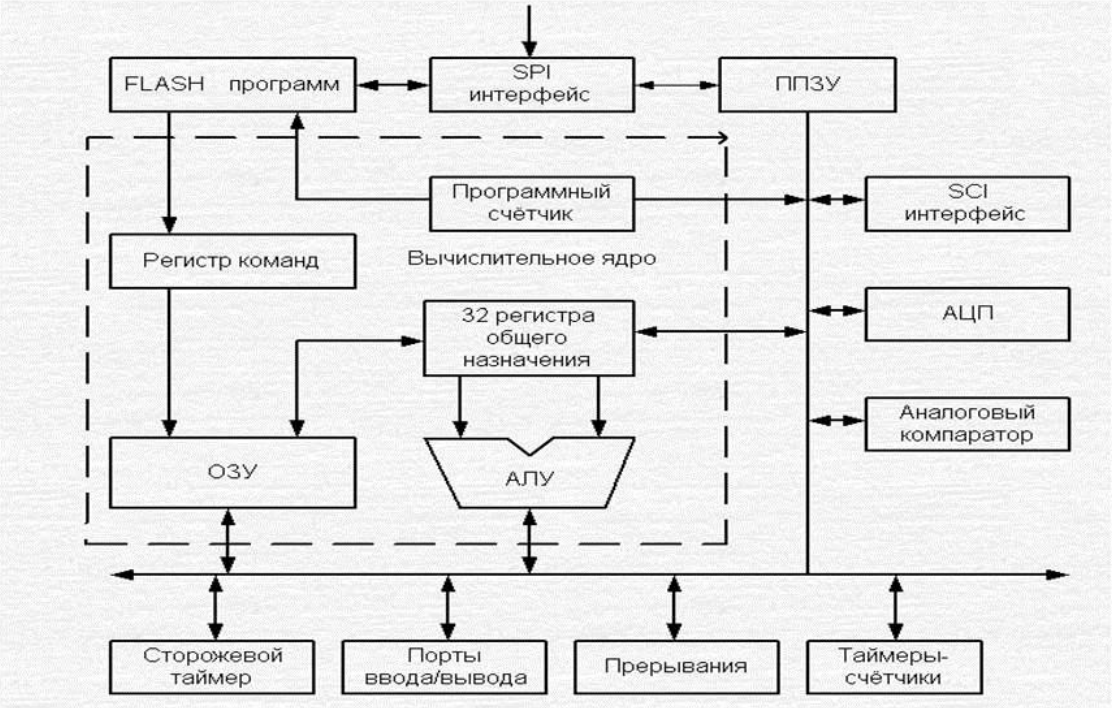
5

Характеристики мікроконтролерів ATtiny фірми Atmel

Тип	Напр. питания В	Такт. Частота МГц	I/O	Flash	EEPROM	SRAM	Интер-фейсы	АЦП	Таймеры	ISP	Корпус
ATtiny11	2.7-5.5	6	6	1K	-	-	-	-	1x8bit	-	PDIP8 SOIC8
ATtiny12	1.8-5.5	6	6	1K	64	-	-	-	1x8bit	I	PDIP8 SOIC8
ATtiny13	1.8-5.5	20	6	1K	64	64	-	4x10bit	1x8bit 2xPWM	I	PDIP8 SOIC8
ATtiny15	2.7-5.5	6	6	1K	64	-	-	4x10bit	2x8bit	I	PDIP8 SOIC8
ATtiny2313	1.8-5.5	20	15	2K	128	128	SPI UART	-	1x8bit 1x16bit	I	PDIP20 SOIC20 MLF32
ATtiny24	1,8...5,5	20	12	2K	128	128	USI 4xPWM RTC	8x10bit	1x8bit 1x16bit	S	PDIP14 MLF20 SOIC14
ATtiny25	2,7...5,5	20	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed	I	PDIP8 SOIC8
ATtiny25V	1.8 - 5.5	10	32	2K	128	128	SPI UART	4x10bit	1x8bit 1x8bit high speed	I	PDIP8 SOIC8
ATtiny26	2.7-5.5	16	16	1K	128	128	SPI UART	11x10bit	2x8bit	I	PDIP20 SOIC20 MLF32
ATtiny28	1.8-5.5	4	20	2K	-	-	-	-	1x8bit	-	PDIP20 SOIC20 MLF32

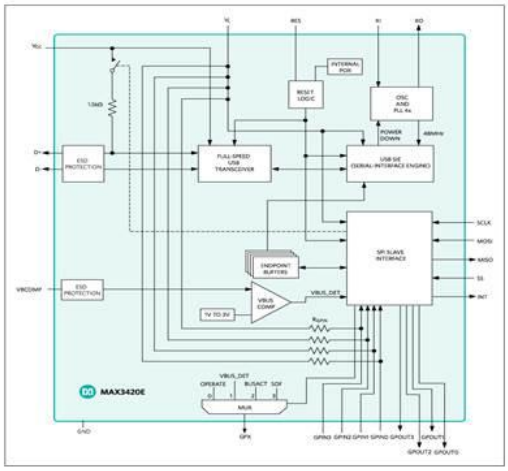
6

Загальна структурна схема МК AVR

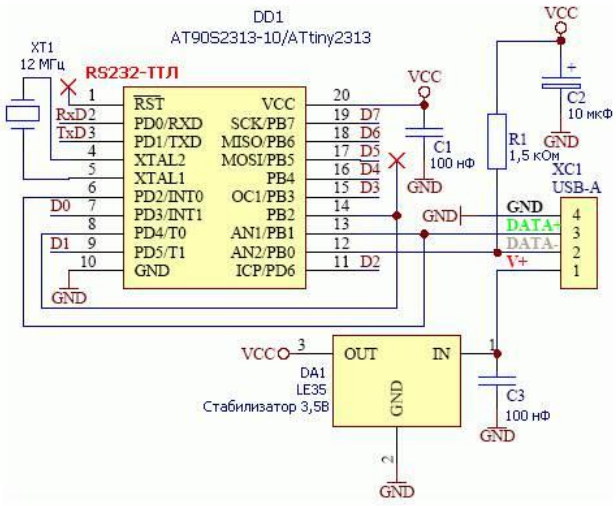


7

Апаратне підключення мікроконтролерів до USB порту



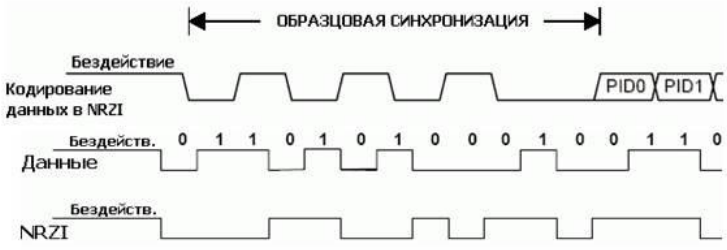
Підключення мікроконтролерів MAX3420E



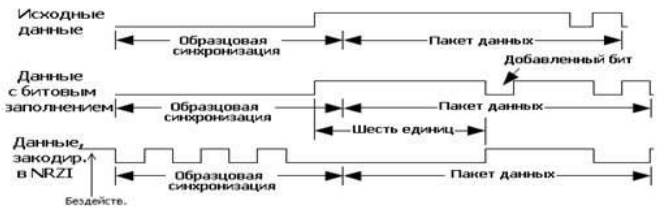
Підключення мікроконтролера ATtiny2313

8

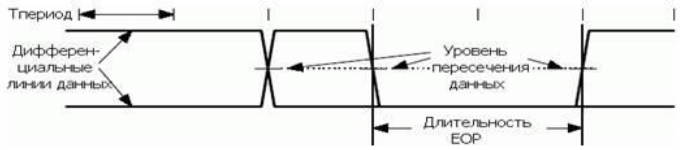
Синхронізація роботи мікроконтролера з USB портом



Кодування даних у коді NRZI



Послідовність кодування даних



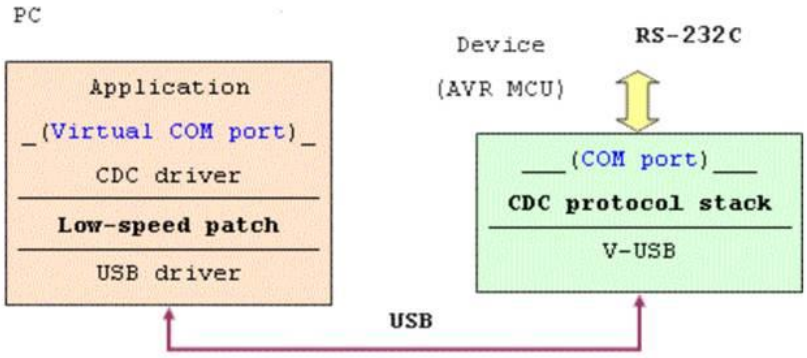
Часова діаграма сигналу EOP (кінець пакету) 9

Реалізація програмного забезпечення

- У якості практичної реалізації створюється CDC-ACM пристрій - віртуальний COM-порт.
- На рівні ОС пристрій буде автоматично розпізнано як послідовний інтерфейс COM-порт в ОС Winows.
- USB - інтерфейс, що використовується для підключення периферійних пристроїв. Відповідно, фактично розуміють «головний пристрій» (хост, що керує обмеженим даними через інтерфейс, виконує функцію ініціатора обстеження) і «периферійний пристрій» (кліент, в процесі обміну даними) «підчиняється» хосту).
- На логічному рівні зв'язок із загальними даними відбувається через логічні (віртуальні) канали всередині одного фізичного інтерфейсу USB. Такі канали називають «Кінцеві точки» (EndPoints).

10

Апаратно-програмна реалізація проекту зі створення CDC-232



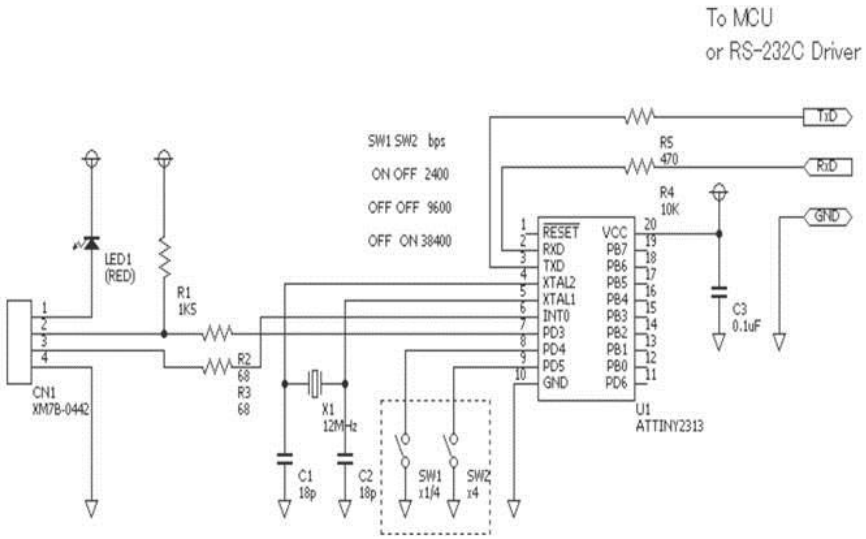
CDC-232 створює віртуальний COM-порт на PC, навіть якщо він не має реального порту RS-232C

В обох проектах використовується безкоштовна бібліотека V-USB, яка дозволяє засобами firmware, прошитого в мікроконтролер AVR, підтримати роботу інтерфейсу USB.

Це дозволяє проводити обмін даними RS-232C (без сигналів управління) після приєднання пристрою і установки драйвера.

11

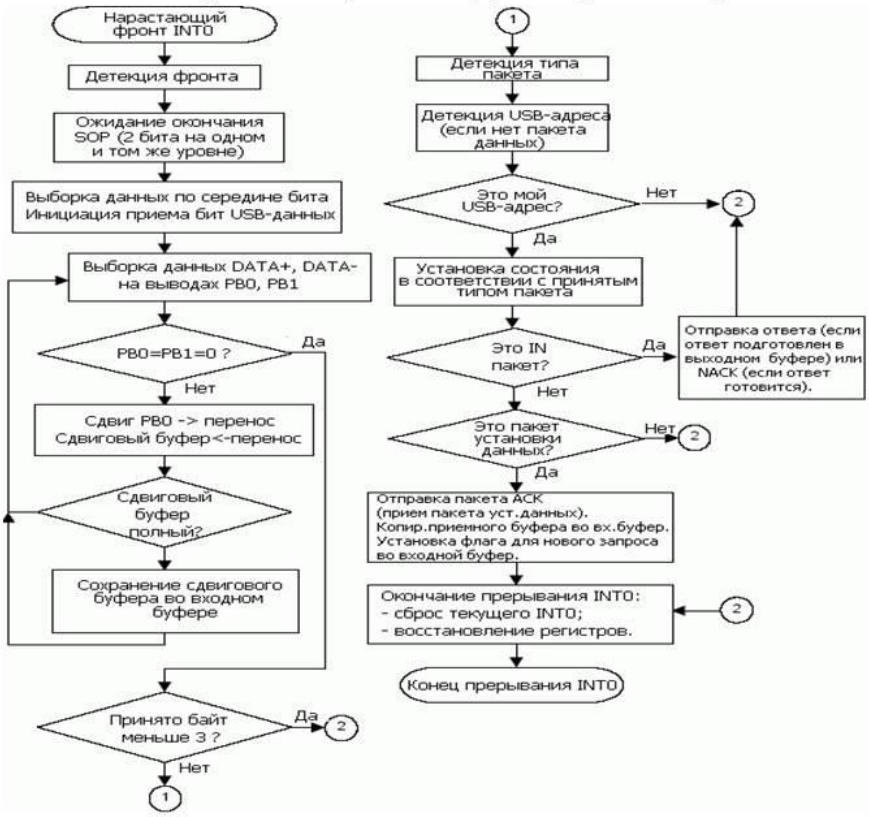
Апаратна реалізація проекту CDC-232



Наведена схема розрахована на широкий клас мікроконтролерів Atmel, зокрема, ATtiny45/85, ATtiny2313/AT90S2313, ATmega8/48/88/168 та ряду інших.

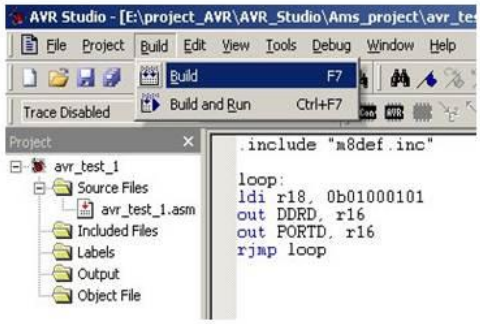
12

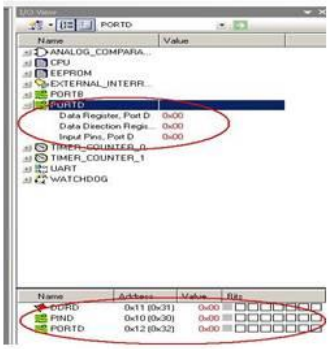
Алгоритм процедури прийому

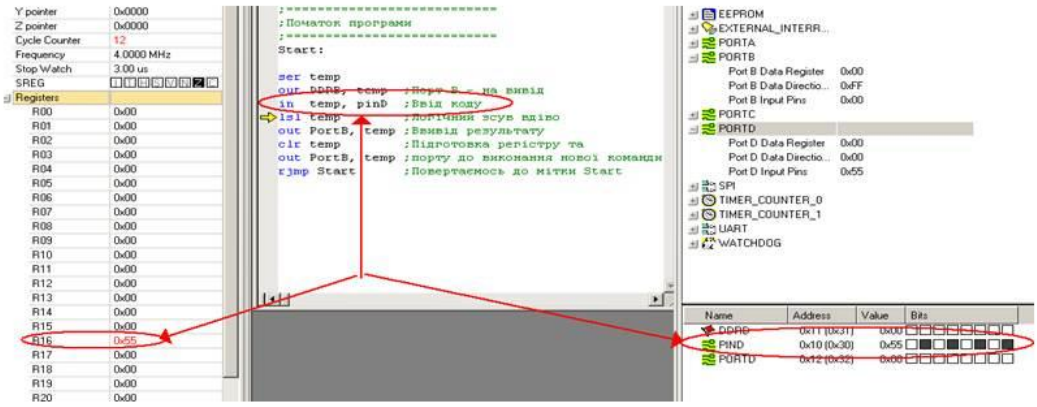


13

ПРОГРАМНЕ СЕРЕДОВИЩА НАЛАШТУВАННЯ AVR Studio

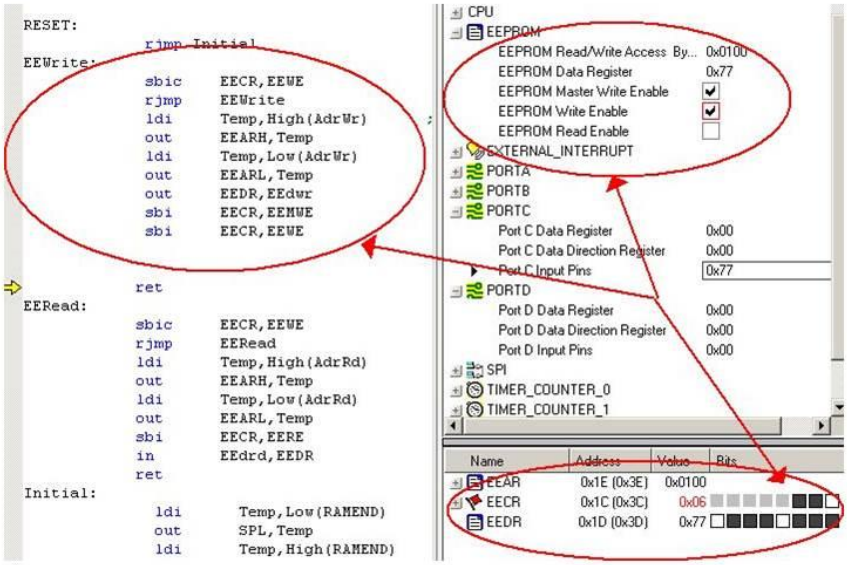






14

Програмна реалізація в AVR Studio



15

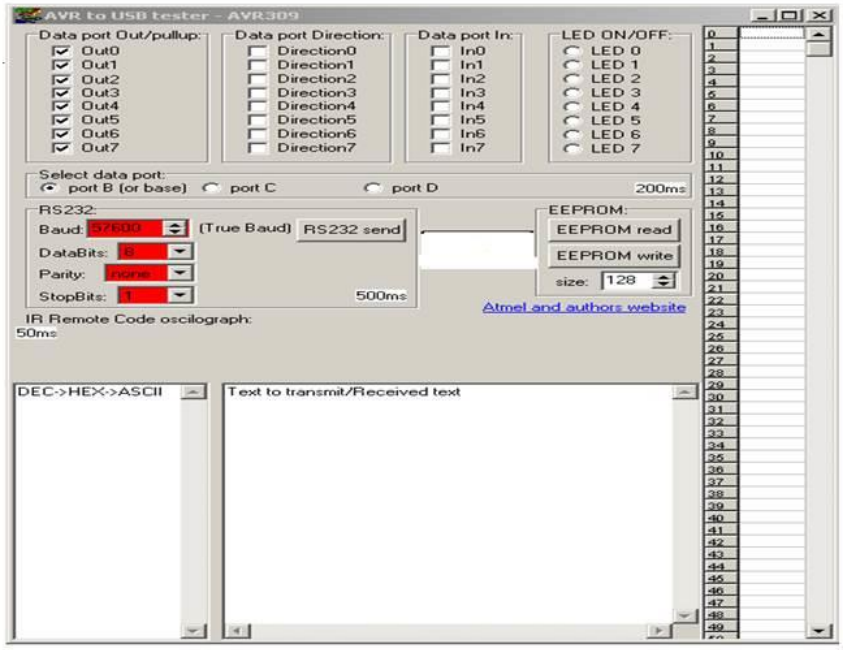
Похибка генерації швидкості універсального асинхронного передавача/приймача

Стандартна швидкість USB	Швидкість в AVR	Похибка, %
600	602	0,33
1200	1204	0,33
2400	2408	0,33
4800	4808	0,17
9600	9616	0,17
19200	19230	0,16
28800	28846	0,16
38400	38462	0,16
57600	57692	0,16
115200	115384	0,16

За рахунок високого значення частоти синхронізації похибка мінімальна.

16

Додаток користувача з використанням бібліотеки AVR309.dll



Приклад використання бібліотеки DLL у вигляді додатку користувача представлена у вигляді файлу AVR309USBdemo.exe, що виконується

17

Висновки

- Проведено розроблення апаратно-програмного підключення мікроконтролерів AVR до шини USB.
- З метою розроблення визначена елементна база мікроконтролерів Atmel, підключення мікроконтролерів MAX3420E та ATmega8/48/88/168 до USB порту.
- Розглянуті питання синхронізації роботи мікроконтролера з USB портом, реалізація програмного забезпечення та реалізації високорівневого API «стека», використання програмної бібліотеки V-USB апаратно-програмної реалізація проекту зі створення CDC-232.
- Проведено вибір програмного середовища налаштування та програмна на базі інструментального середовища налаштування AVR Studio для мікроконтролерів Atmel.
- Розроблено вихідний код програми для мікроконтролера Atmel AT91C.
- Розроблені заходи щодо охорони праці в умовах виробництва.
- Результати роботи та запропоновані рішення можуть бути використані у навчальному процесі кафедри комп'ютерних наук та інженерії при вивченні дисципліни «Цифрова схемотехніка».

18