Лабораторна робота №6

Моделювання модуля універсального асинхронного приймача-передавача сім'ї AVR

Тема: Моделювання модуля універсального асинхронного приймача - передавача сім'ї AVR

Мета: Користуючись пакетом PROTEUS дослідити моделювання модуля універсального асинхронного приймача-передавача сім'ї AVR

1. Порядок виконання роботи

- створити модель пристрою в пакеті Proteus 8.6
- розробити схему алгоритму роботи моделі та робочу програму
- створити hex-файл та підключити його до мікроконтролера
- запустити модель та виконати її дослідження згідно методичних вказівок
- зробити відповідні висновки.

2. Стислі теоретичні відомості

2.1. Опис моделі

Робочу модель дослідження універсального асинхронного приймача - передавача (УАПП) наведено на рисунку 1.

Розберемо цю модель докладніше. Зліва ми бачимо вісім кнопок, за допомогою яких можна задавати байт (8 біт) даних, який ми будемо передавати через модуль УАПП.

Кнопки підключено до восьми ліній порту PA мікроконтролера: PA.0, PA.1, ... , PA.7. Якщо якась із кнопок відпущена, то через резистори R11...R18 на відповідну лінію порта PA подається логічна 1. Якщо кнопка нажата, то вводиться логічний нуль. Праворуч зображено вісім світлодіодів, які підключено до ліній порту PC: PC.0, PC.1,..., PC.7. Катоди світлодіодів підключено до спільного проводу (землі), а на аноди через резистори К2 ... Я9, які обмежують струм, із виходів порту PC подаються логічні одиниці, коли треба засвітити відповідний світлодіод.

До ліній TxD-вихід передавача УАПП та RxD-вхід приймача підключено

осцилограф, та Virtual Terminal, який вибрано з віртуальних інструментів програми Proteus. На них ми будемо відображати повідомлення, які вводиться та виводяться через модуль УАПП в/з мікроконтролера.

До виводів XTAL1 та XTAL2 підключають кварцовий резонатор частотою, яка визначає тактову частоту генератора мікроконтролера. В нашому прикладі вона дорівнює f_{BQ}=8МГц. Також підключають конденсатори С1 та С2, ємністю 30пФ, які призначені для підвищення стабільності роботи системного генератора. Резонатор та конденсатори потрібні в практичній схемі. В модель Proteus їх можна не вводити. Для завдання частоти треба двічі лівою кнопкою миші клацнути на мікроконтролері та в опції Clock Frequency вказати значення частоти.

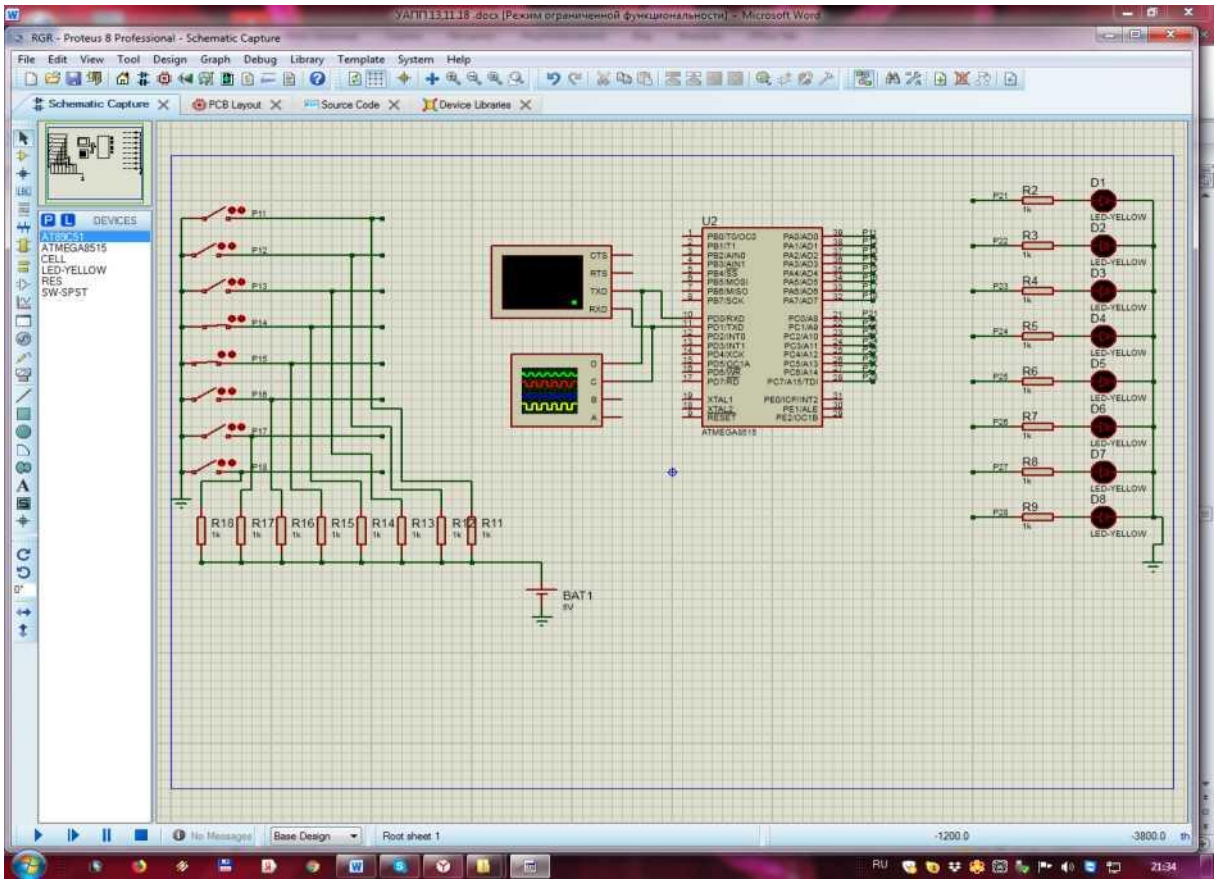

Рисунок 1 - Схема робочої моделі

В якості мікроконтролера використано мікросхему ATmega8515. Мікроконтролер побудований за процесорною архітектурою AVR, тобто він вміє

виконувати асемблерні команди, які описано цим стандартом.

2.2. Порядок моделювання

1. Перш за все, потрібно пересвідчитись, що Terminal правильно налаштовано. Для цього треба двічі натиснути на нього лівою кнопкою миші, та виставити відповідні налаштування. Нижче наведено приклад для випадку, коли потрібно передавати зі швидкістю 9600 біт/с: 8 біт даних та один стоп-біт з перевірки на не парність (рисунок 2).

Якщо треба використовувати перевірку на парність або непарність, то це треба вказати в опції Parity (рисунок 2).

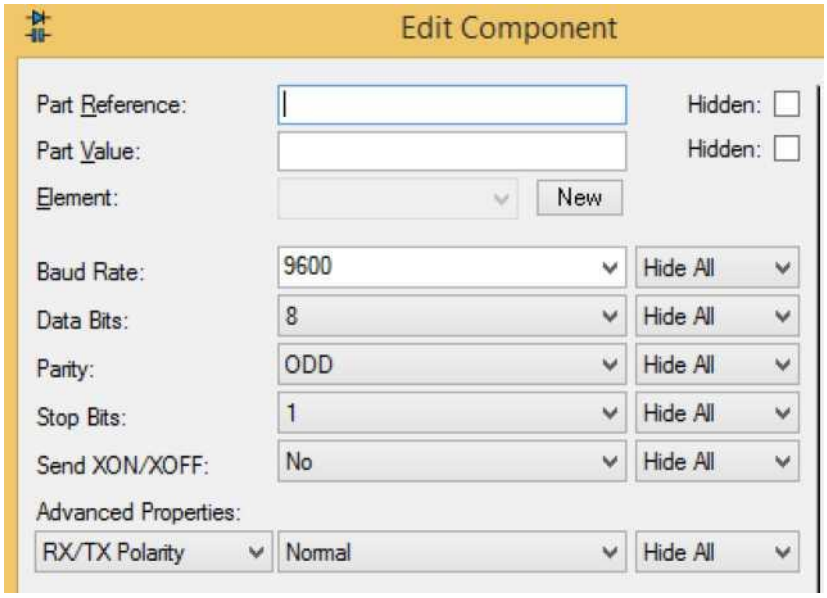

Рисунок 2

2. Далі натискаємо Start VSM Debugging (рисунок 3).

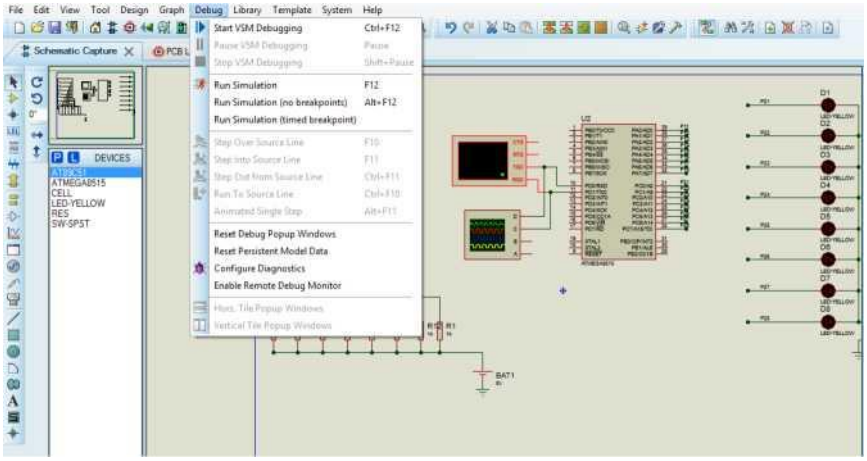

Рисунок 3

3. Потім натискаємо на кнопку Pause VSM Debugging

4. Переходимо на вкладку Source Code та ставимо дві точки зупинки. Для цього потрібно лівою кнопкою миші двічі клацнути з лівої сторони двох команд, як показано на рисунку 4.

5. Натиснемо Run Simulation (F12).

6. Перейдемо до вікна Virtual Terminal, натиснемо правою кнопкою миші на вікні, та виберемо Hex Display Mode та Echo Typed Characters (рисунок 5).

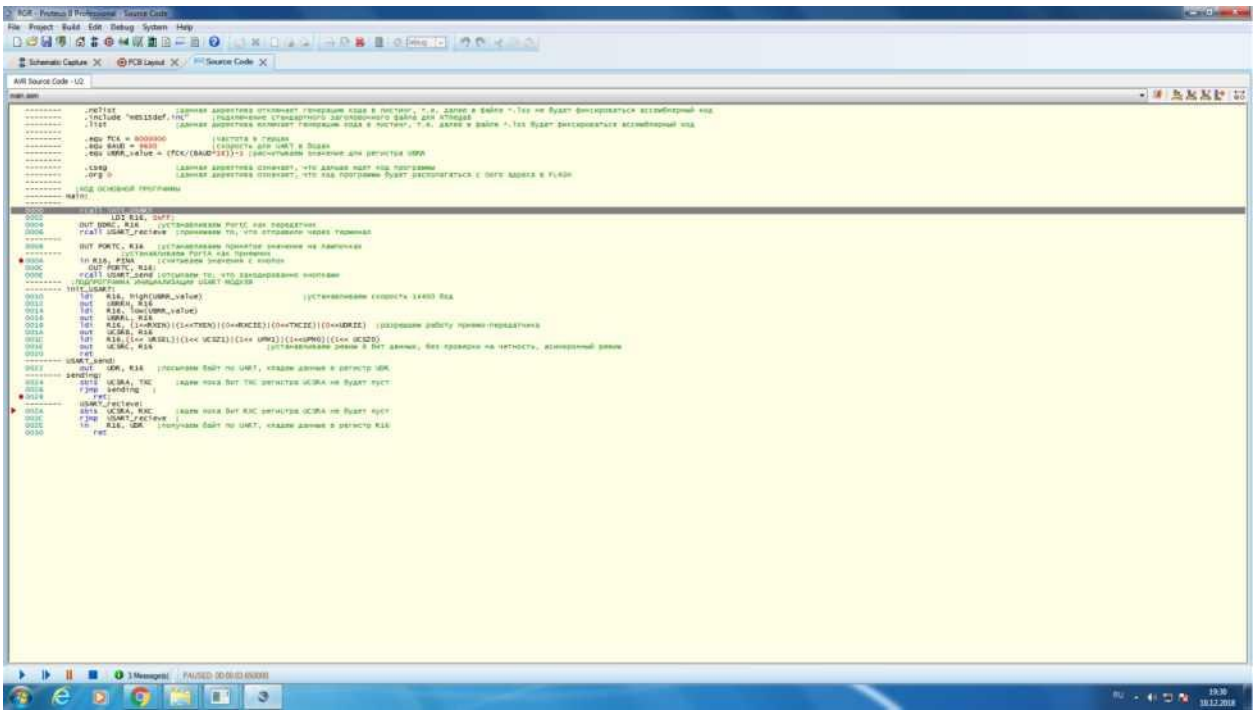

Рисунок 4

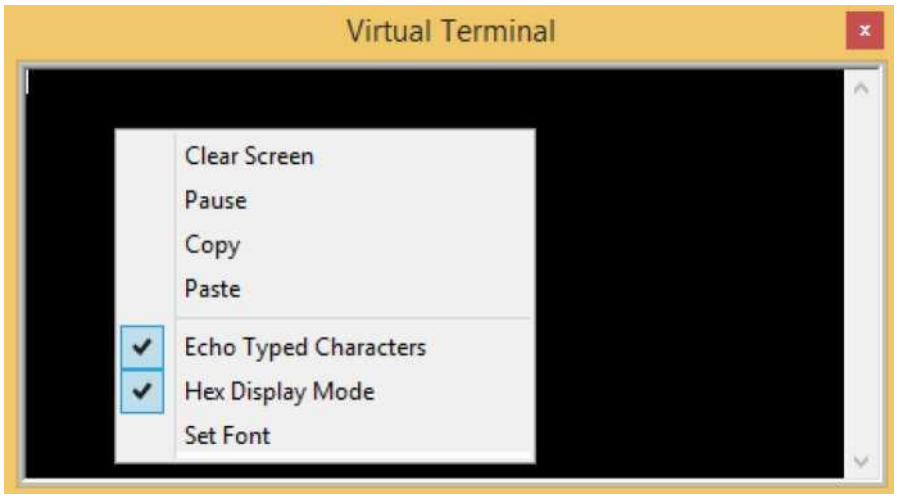

Рисунок 5

Якщо вікно Virtual Terminal закрито, то його можна відкрити із вкладки Debug.

7. Далі введемо якийсь символ, який хочемо надіслати від віртуального терміналу до мікроконтролера через УАПП. При введенні символу у вікні терміналу повинен стояти курсор. Символ, який ми вводимо з клавіатури, віртуальний термінал перетворює в ASCII-код згідно таблиці, яку наведено у [4].

Введемо, наприклад, цифру 7, ASCII код якої 37 (hex) = 0011 0111 (bin). Програма перейде на рядок, де ми поставили першу точку зупину. Натиснемо F10, зробимо один наступний крок програми, та перейдемо на вкладку зі схемою Schematic Capture. Звернемо увагу на світлодіоди та на індикатори, які на виході порту C (рисунок 6).

Світлодіоди та індикатори точно відображають бінарний код символу, який ми відправили (якщо дивитися знизу уверх, а бінарне число читати зліва на право), нуль - лампочка вимкнена, один - лампочка увімкнена (прийнято символ 00110111в = 37h).

8. Далі перевіримо як мікроконтролер через УАПП відправить символ, який ми закодували кнопками: замкнута кнопка - нуль, а розімкнута - одиниця (рисунок 7).

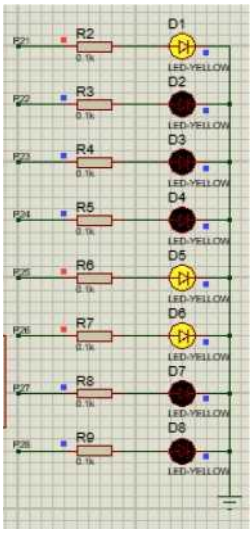

Рисунок 6

Якщо дивитися на кнопки знизу уверх, то на рисунку 7 набрано число 1110 0111 (bin) = E7 (hex).

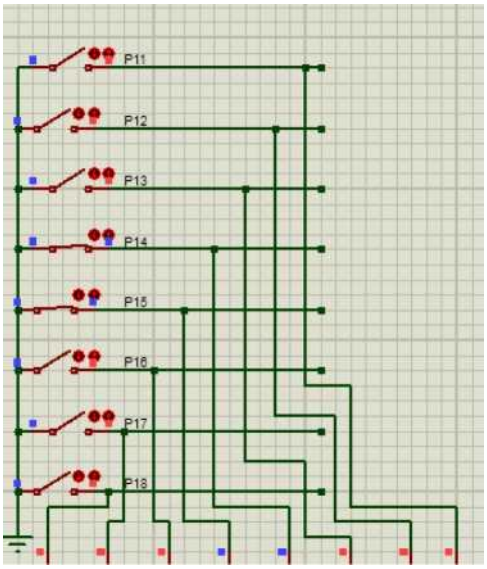

Рисунок 7

9 Тепер подивимося на вікно осцилографа. Для того, щоб побачити переданий сигнал на осцилографі, потрібно натиснути кнопку на опції One -Shot (один кадр), як показано на рисунку 8.

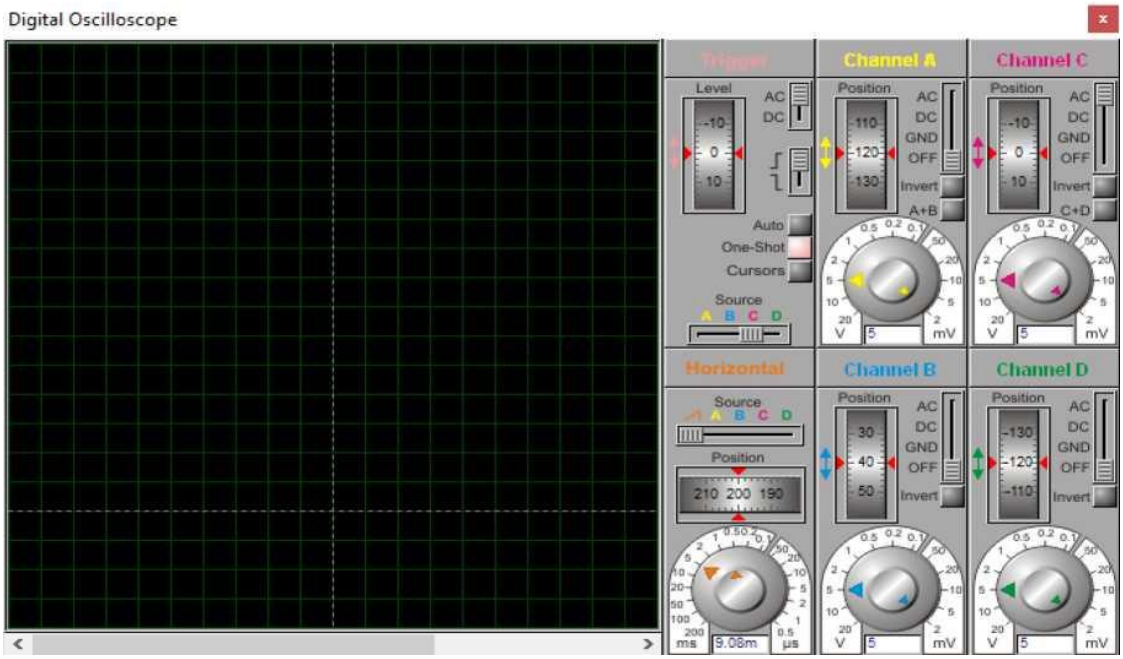

Рисунок 8

10 Натиснемо Run Simulation для продовження виконання програми (рисунок 4.49). Програма дійде до другої точки зупину.

11 Розберемо більш детально сигнали, що відображаються на осцилографі (рисунок 9).

Осцилограф відображає сигнали за двома променями: верхній (червоний) - сигнал, який передається від кнопок, нижній (зелений) - сигнал, який приймається від терміналу: 37(hex).

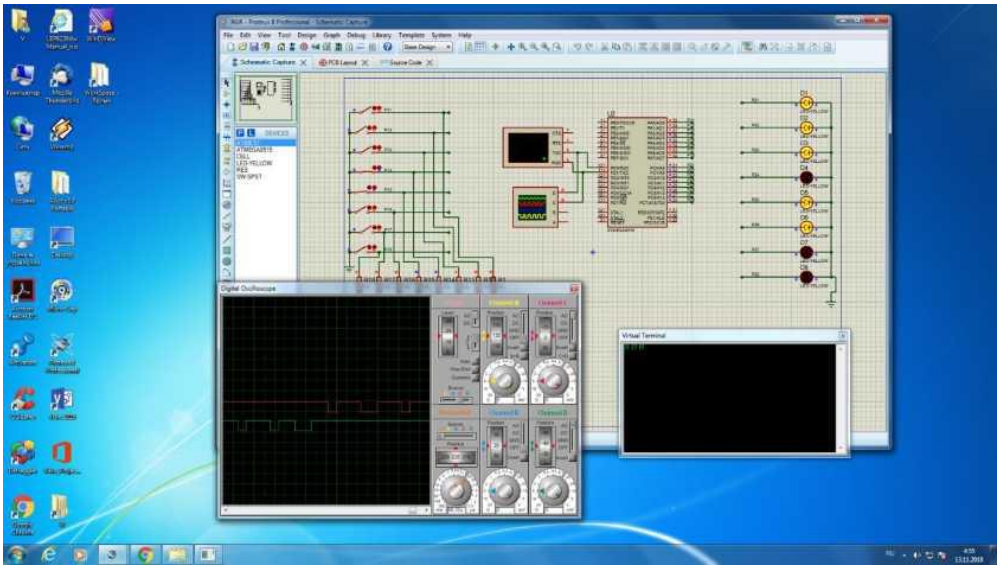

Рисунок 9

Розглянемо сигнал, який передається від кнопок.

Перший перепад із високого рівня у низький - старт-біт.

Потім ідуть вісім інформаційних бітів, починаючи з молодшого розряду, які будуть відображені на віртуальному терміналі та осцилографі, як 1110 0111 (bin) = E7 (hex). Далі іде 9-й біт , який дорівнює 0, тому що число одиниць в байті, який було передано - парне (розряд P в регістрі прапорців дорівнює 0). Останній 10-й біт, який дорівнює 1, це стоп-біт.

12 Тепер визначимо отриману швидкість передачі. Для знаходження швидкості передачі (в даному прикладі було обрано швидкість 9600 біт/сек = =9600 пос/сек = 9600 бод) підберемо таку розгортку сигналу на осцилографі, щоб сумістити бічні сторони імпульсу із вертикальними лініями сітки осцилографа.

Для швидкості передачі 9600 біт/сек отримаємо наступну картину (рисунок 10.

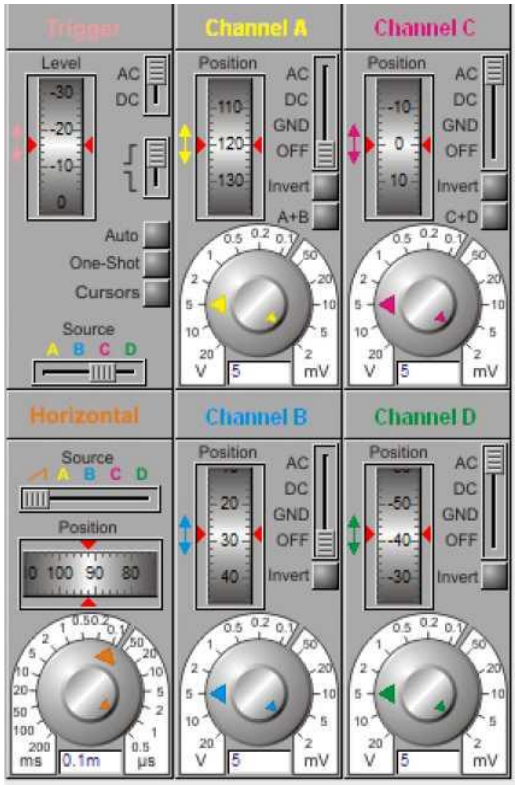

Рисунок 10

Як ми бачимо, тривалість одного імпульсу дорівнює 0,104 мс. Отже, якщо розділити один біт на цей час, та помножити на 1000, отримаємо задану швидкість:

$$V = \frac{1}{0{,}104} \cdot 1000 = 9600 \ \frac{біт}{с}$$

Нижче розглянуто ще один приклад (рисунки 11, 12), коли на кнопках було набрано число 67(hex)=01100111(bin). На верхньому промені осцилографа побачимо: перший нуль - старт- біт, потім 3 одиниці та один 0 - це число 7 на кнопках, далі йде нуль, дві одиниці та нуль - це число 6 на кнопках, а потім на осцилографі бачимо 1 - це біт доповнення до парності, та наступна одиниця - стоп - біт.

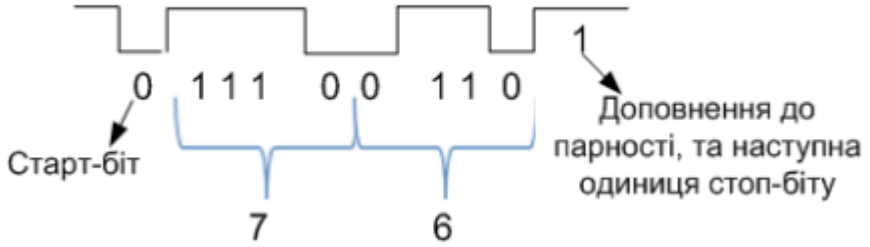

Рисунок 11

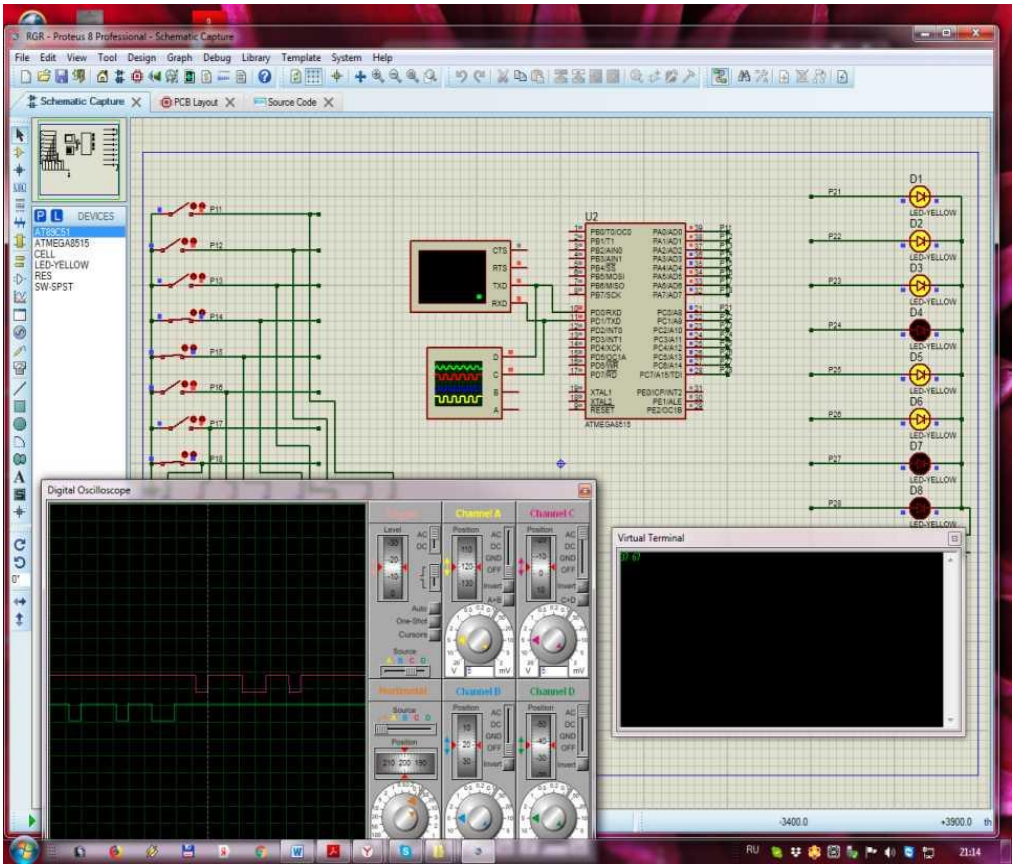

Рисунок 12

2.3. Схема алгоритму роботи

На рисунку 13 наведено схему алгоритму роботи.

Рисунок 13 - Схема алгоритму роботи

Лістинг робочої програми

Лістинг робочої програми, яку розроблено згідно з алгоритмом, який наведено на рисунку 13, представлено надалі. В дужках праворуч біля кожної команди цифрою відмічено номер блоку схеми алгоритму, який реалізує ця команда.

Робоча програма мовою Асемблер

```
.nolist                    ; директива відключає генерацію коду у лістинг,
        ;тобто далі у файлі *.lss не буде фіксуватися асемблерний код
.include "m8515def.inc"    ;підключення стандартного заголовочного
                           ; файлу для ATmega8
.list                      ;директива включає генерацію коду у лістинг,
        ;тобто далі у файлі *.lss буде фіксуватися асемблерний код
.equ fCK = 8000000         ;частота в герцах
.equ BAUD = 9600           ;швидкість для UART у бодах
. equ UBRR_value = (fCK/(BAUD* 16))-1   ;розраховуємо значення для
                                        регістра ;UBRR
.cseg                      ;директива означає, що далі іде код програми
.org 0                     ;директива означає, що код програми у FLASH буде
                           ;розміщено з нульової адреси
; КОД ОСНОВНОЇ ПРОГРАМИ
main:
ініціалізації з мітки init_USART
rcall init_USART           ; (блок 2) відносний виклик підпрограми
;налаштування порту С на передачу
LDI R16, 0xFF              ; (блок 2) R16 ← 0xFF
OUT DDRC, R16              ; (блок 2) DDRC ← R16
; (блок 3) отримуємо те, що відправив віртуальний термінал
rcall USART_recieve ; (блок 3) відносний виклик підпрограми з мітки
                           ; USART_recieve
; (блок 4) виводимо отримане від терміналу на світлодіоди
OUT PORTC, R16             ; (блок 4) PORTC←R16
; (блок 5) налаштовуємо порт А як передавач
LDI R16, 0x00              ; (блок 5) R16←0x00
```

```
OUT DDRA, R16          ; (блок 5) DDRA← R16

                       ; (блок 6) читаємо стан кнопок
   in R16, PINA        ; (блок 6) R16← PINA
   ; (блок 7) виводимо отримане від кнопок на світлодіоди
   OUT PORTC, R16      ; (блок 7) PORTC←R16
   ; (блок 8) підпрограма з міткою USART_send відправляє стан кнопок на
   осцилограф та термінал
   rcall USART_send    ; (блок 8) відносний виклик підпрограми з мітки
                       ; USART_ send
   ; (блок 1) ПІДПРОГРАМА ІНІЦІАЛІЗАЦІЇ USART - МОДУЛЯ init_USART:
   ; програмуємо швидкість обміну 9600бод
ldi R16, high(UBRR_value) ; (блок 1) R16←старший байт UBRR_value
out UBRRH, R16            ; (блок 1) UBRRH^R16
ldi R16, low(UBRR_value) ; (блок 1) R16←молодший байт UBRR_value
out UBRRL, R16            ; (блок 1) UBRRL ^R16
   ; дозволяємо роботу приймача-передавача модуля USART
ldi R16, (1<<RXEN)|(1<<TXEN)|(0<<RXCIE)|(0<<TXCIE)|(0<<UDRIE)
out UCSRB, R16                  ; (блок 1) UCSRB^R16
   ; програмуємо передачу 8 біт з перевіркою на парність
   ; у асинхроному режимі
ldi R16, (1<< URSEL)|(1<<UPM1)|(1<<UPM0)|(1<< UCSZ1)|(1<< UCSZ0)
   out UCSRC, R16       ; (блок 1) UCSRC← R16
   ret                  ; (блок 1) повернення з підпрограми ініціалізації
   ; передача через модуль USART
   USART_send:
   out UDR, R16                 ; (блок 8) регістр даних UDR← R16
   sending:
; чекаємо завершення передачі
```

```
sbis UCSRA, TXC ; (блок 8) якщо біт TXC =1, то наступна ;команда пропускається, інакше - наступна команда
rjmp sending        ; блок 8) безумовний перехід на мітку sending
ret                 ; (блок 8) повернення з підпрограми
USART_recieve:
sbis UCSRA, RXC     ; (блок 3), якщо біт RXC =1 (в буфері ПРМ є
; отриманий байт) , то пропустити наступну команду,
; інакше наступна команда
rjmp USART_recieve ; (блок 3) безумовний перехід на мітку
                    ;USART_recieve
in R16, UDR         ; (блок 3) R16←UDR (завантажуємо в
                    ; регістр R16 отриманий байт від термінала
ret                 ; (блок 3) повернення з підпрограми
```

3. Зміст звіту

Звіт по роботі повинен містити:

- схему моделі;
- схему алгоритму роботи моделі;
- робочу програму;
- формули за потребою.

Контрольні запитання

1. Що таке УСАПП (USART) та УАПП (UART)?
2. З яких трьох частин складається структура модуля УСАПП/УАПП? Які елементи містить кожна з них?
3. Які регістри використовуються для керування модулями УАПП та УСАПП?
4. Чим в асинхронному режимі задається швидкість прийому та передачі даних?

5. Поясніть структурну схему блока синхронізації модуля УСАПП для мікроконтролера Mega 128.
6. Які режими синхронізації підтримує УСАПП?
7. Як визначається швидкість обміну в модулі УСАПП?
8. Як визначається розмір слова даних у модулях УСАПП?
9. Як працює схема контролю парності?
10. Як проходить передача даних в модулях УСАПП?
11. Як проходить прийом даних в модулях УСАПП?
12. Що таке режим мультиконтролерного обміну?
13. Опишіть послідовність дій для здійснення обміну даними у мікроконтролерній мережі.
14. Як визначити швидкість передачі даних для асинхронного звичайного режиму?
15. Як визначити тривалість передачі кадру для асинхронного прискореного режиму?
16. Як в моделі підключено кнопки, які задають байт для передачі?
17. Як в моделі підключено світлодіоди, які відображають передану в моделі інформацію?
18. Як в моделі визначається тривалість переданого біта?
19. Опишіть формат та призначення розрядів регістра UCSRC.
20. Як в програмі відбувається налаштування порту А на передачу?
21. Як в програмі визначається час завершення передачі та прийому чергового байта?
22. Дайте характеристику ASCII-коду.
23. Поясніть як в моделі підключено світлодіоди та в якому випадку вони будуть світитися?
24. Як працюють схеми відновлення тактового сигналу і даних при асинхронному прийомі?