Лабораторная работа № 8

Изучение интерфейса RS485.

Протокол Modbus

Цель лабораторной работы: познакомиться с принципами организации систем с применением RS485, а также научиться использовать протокол Modbus для обмена информацией посредством интерфейса RS485.

Теоретическая часть

Интерфейс RS485

Интерфейс RS-485 (EIA/TIA-485) - один из наиболее распространенных стандартов физического уровня связи. Приемопередатчик RS-485 является интерфейсом для реализации сетей с последовательной передачей данных, предназначенных для жестких условий эксплуатации в промышленных применениях и в системах автоматизированного управления зданиями.

Данный стандарт последовательного интерфейса обеспечивает обмен данными с высокой скоростью на сравнительно большое расстояние по одной дифференциальной линии.

К числу интерфейсов последовательной передачи данных относятся CAN, RS-232, RS-485, I^2C, SPI, USB, однако RS-485 по-прежнему остаётся наиболее надежным, особенно в жестких условиях эксплуатации.

Сеть, построенная на интерфейсе RS-485, представляет собой приемопередатчики, соединенные при помощи витой пары - двух скрученных проводов. В основе интерфейса RS-485 лежит принцип дифференциальной (балансной) передачи данных. Суть его заключается в передаче одного сигнала по двум проводам. Причем по одному проводу идет оригинальный сигнал, а по другому - его инверсная копия. Таким образом, между двумя проводами витой пары всегда есть разность потенциалов: при "1" она положительна, при "0" - отрицательна.

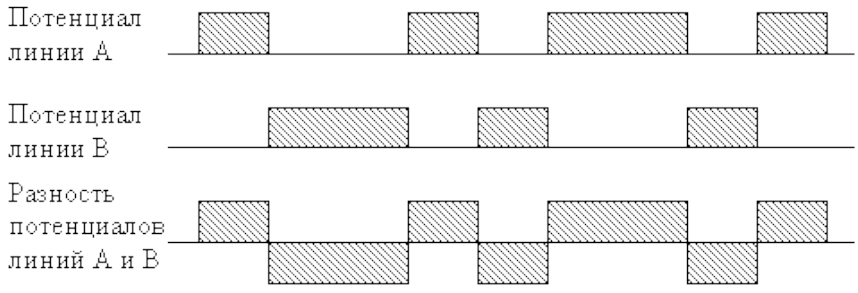

Рисунок 1 – Распределение потенциалов на линиях RS485

Такой способ передачи обеспечивает высокую устойчивость к синфазной помехе. Синфазной называют помеху, действующую на оба провода линии одинаково. К примеру, электромагнитная волна, проходя через участок линии связи, наводит в обоих проводах потенциал. Если сигнал передается потенциалом в одном проводе относительно общего, как в RS-232, то наводка на этот провод может исказить сигнал относительно хорошо поглощающего наводки общего (GND). Кроме того, на сопротивлении длинного общего провода будет падать разность потенциалов земель - дополнительный источник искажений. При **дифференциальной передаче** искажения не происходит. Если два провода пролегают близко друг к другу, да еще перевиты, то наводка на оба провода одинакова. Потенциал в обоих одинаково нагруженных проводах изменяется одинаково, при этом информативная разность потенциалов остается без изменений.

Аппаратная реализация интерфейса - микросхемы приемопередатчиков с дифференциальными входами/выходами (к линии) и цифровыми портами (к портам UART контроллера). Существуют два варианта такого интерфейса: RS-422 и RS-485.

RS-422 - **полнодуплексный интерфейс**. Прием и передача идут по двум отдельным парам проводов. На каждой паре проводов может быть только по одному передатчику.

RS-485 - полудуплексный интерфейс. Прием и передача идут по одной паре проводов с разделением по времени. В сети может быть много передатчиков, так как они могут отключаются в режиме приема.

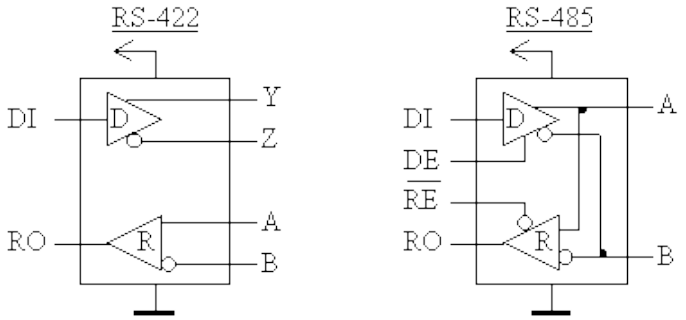

Рисунок 2 – Аппаратная реализация RS422 и RS485, где

D (driver) - передатчик;
R (receiver) - приемник;
DI (driver input) - цифровой вход передатчика;
RO (receiver output) - цифровой выход приемника;
DE (driver enable) - разрешение работы передатчика;
RE (receiver enable) - разрешение работы приемника;
A - прямой дифференциальный вход/выход;
B - инверсный дифференциальный вход/выход;
Y - прямой дифференциальный выход (RS-422);
Z - инверсный дифференциальный выход (RS-422).

Стандарт TIA/EIA-485 допускает использование RS-485 на расстоянии до 1200 м. На более коротких дистанциях скорости передачи данных – более 40 Мбит/с. Использование дифференциального сигнала обеспечивает интерфейсу RS-485 более высокую дальность, однако скорость передачи данных уменьшается по мере увеличения длины линии.

Интерфейс RS-485 может использоваться в полудуплексном режиме с применением одной витой пары проводов или в дуплексном режиме с одновременными передачей и приемом данных, что обеспечивается использованием двух витых пар (четыре провода).

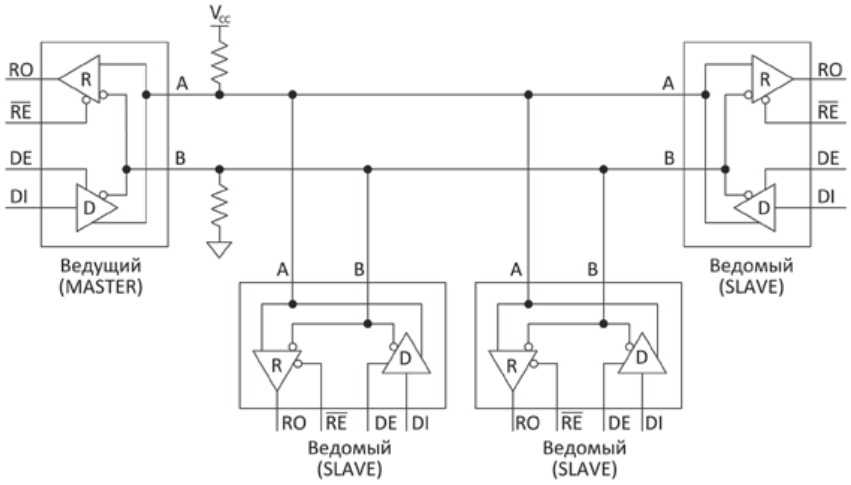

Рисунок 3 – Схема организации сети на основе интерфейса RS485

Чувствительность приемника составляет ±200 мВ. Поэтому, для распознавания одного бита данных уровни сигнала в точке подключения приемника должны быть больше +200 мВ для нуля и меньше -200 мВ для единицы. При этом приемник будет подавлять помехи, уровень которых находится в диапазоне ±200 мВ.

Цифровой выход приемника (RO) подключается к порту приемника UART (RX). Цифровой вход передатчика (DI) к порту передатчика UART (TX). Поскольку на дифференциальной стороне приемник и передатчик соединены, то во время приема нужно отключать передатчик, а во время передачи - приемник. Для этого служат управляющие входы - разрешение приемника (RE) и разрешения передатчика (DE). Так как вход RE инверсный, то его можно соединить с DE и переключать приемник и передатчик одним сигналом с любого порта контроллера. При уровне "0" - работа на прием, при "1" - на передачу.

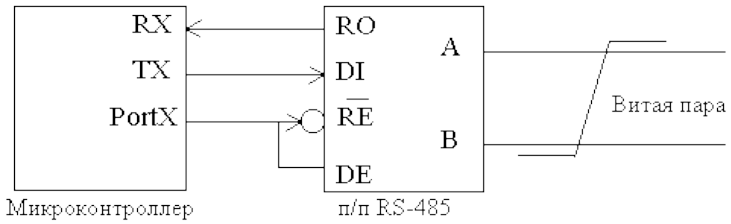

Рисунок 4 – Подключение приёмопередатчика RS485

Приемник, получая на дифференциальных входах (AB) разность потенциалов переводит их в цифровой сигнал на выходе RO.

Входное сопротивление приемника со стороны линии обычно составляет 12 кОм. Так как мощность передатчика не беспредельна, это создает ограничение на количество приемников, подключенных к линии. Согласно спецификации RS-485 с учетом согласующих резисторов передатчик может вести до 32 приемников. Однако есть ряд микросхем с повышенным входным сопротивлением, что позволяет подключить к линии значительно больше 32 устройств.

При больших расстояниях между устройствами, связанными по витой паре и высоких скоростях передачи начинают проявляться так называемые эффекты длинных линий. Скорость распространения электромагнитных волн в проводниках существенно меньше скорости света в вакууме и составляет немногим больше 200 мм/нс. Электрический сигнал имеет также свойство отражаться от открытых концов линии передачи и ее ответвлений. Грубая аналогия - желоб, наполненный водой. Волна, созданная в одном конце, идет по желобу и, отразившись от стенки в конце, идет обратно, отражается опять и так далее, пока не затухнет. Для коротких линий и малых скоростей передачи этот процесс происходит быстро и остается незамеченным. Однако, время реакции приемников - десятки/сотни наносекунд. В таком масштабе времени несколько десятков метров электрический сигнал проходит отнюдь не мгновенно. И если расстояние достаточно большое, фронт сигнала, отразившийся в конце линии и вернувшийся обратно, может исказить текущий или следующий сигнал. В таких случаях нужно каким-то образом подавлять эффект отражения.

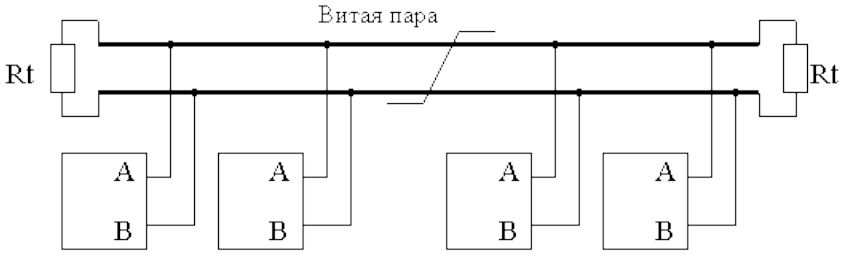

Рисунок 5 – Применение резисторов для согласования линии

У любой линии связи есть такой параметр, как волновое сопротивление **Zв**. Оно зависит от характеристик используемого кабеля, но не от длины. Для обычно применяемых в линиях связи витых пар **Zв=120 Ом**. Оказывается, что если на удаленном конце линии, между проводниками витой пары включить резистор с номиналом равным волновому сопротивлению линии, то электромагнитная волна дошедшая до "тупика" поглощается на таком резисторе. Отсюда его названия - согласующий резистор или "терминатор".

Минус согласования на резисторах - повышенное потребление тока от передатчика, ведь в линию включается низкоомная нагрузка.

Для коротких линий (несколько десятков метров) и низких скоростей (меньше 38400 бод) согласование можно вообще не делать

Протокол Modbus

Modbus — открытый коммуникационный протокол, основанный на архитектуре ведущий — ведомый (master-slave). Широко применяется в промышленности для организации связи между электронными устройствами.

Основные достоинства стандарта — открытость и массовость. Промышленностью сейчас выпускается очень много моделей датчиков, исполнительных устройств, модулей обработки и нормализации сигналов и др. Практически все промышленные системы контроля и управления имеют программные драйверы для работы с MODBUS-сетями.

Логический уровень

Modbus ASCII

Данные кодируются символами из таблицы ASCII и передаются в шестнадцатеричном формате. Начало каждого пакета обозначается символом двоеточия, а конец — символами возврата каретки и переноса строки. Это позволяет использовать протокол на линиях с большими задержками и оборудовании с менее точными таймерами.

Modbus RTU

В протоколе Modbus RTU данные кодируются в двоичный формат, и разделителем пакетов служит временной интервал. Этот протокол критичен к задержкам и не может работать, например, на модемных линиях. При этом, накладные расходы на передачу данных меньше, чем в Modbus ASCII, так как длина сообщений меньше.

Modbus TCP

Структура пакетов схожа с Modbus RTU, данные также кодируются в двоичный формат, и упаковываются в обычный TCP-пакет, для передачи по IP-сетям. Проверка целостности, используемая в Modbus RTU, не применяется, так как TCP уже имеет собственный механизм контроля целостности.

Рисунок 6 - Логический уровень Modbus

Формат пакета Modbus

Все устройства Modbus взаимодействуют, следуя модели master-slave. Запросы может инициировать только master-устройство, slave-устройства могут только отвечать на запросы, и не могут самостоятельно начинать передачу данных. В зависимости от реализации протокола, заголовки пакета различаются. Вот основные составляющие пакета, которые важно знать:

ADU (Application Data Unit) — пакет Modbus целиком, со всеми заголовками, PDU, контрольной суммой, адресом и маркерами. Отличается, в зависимости от реализации протокола.

PDU (protocol data unit) — основная часть пакета, одинаковая для всех реализаций протокола. Содержит сам payload.

Адрес устройства — адрес получателя, то есть slave-устройства. В одном сегменте Modbus-сети могут находится до 247 устройств. Только slave-устройства имеют различающиеся адреса, master-устройство не имеет адреса. Адрес «0» используется для **широковещательных запросов** от master, при этом, slave-устройства не могут отвечать на эти широковещательные пакеты.

Контрольная сумма — алгоритмы проверки целостности пакетов. В Modbus RTU и ASCII используется 2 байта контрольной суммы. В Modbus RTU применяется алгоритм CRC16, в Modbus ASCII

— более простой и менее надежный LRC8. В Modbus TCP контрольная сумма не добавляется в ADU, так как целостность проверяется на уровне TCP.

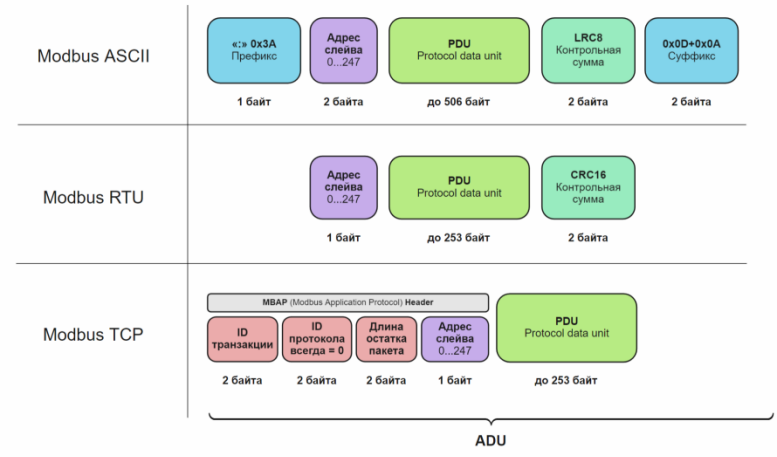

Рисунок 7 – Структура пакета Modbus

Регистры и функции

Структура запросов Modbus состоит из кода функции (чтение/запись), и данных, которые нужно считать или записать. При этом, коды функции различаются для разных типов данных:

- **Discrete Inputs** — дискретные входы устройства, доступны только для чтения. Диапазон адресов регистров: с 10001 по 19999. Имеют функцию «02» — чтение группы регистров
- **Coils** — дискретные выходы устройства, или внутренние значения. Доступны для чтения и записи. Диапазон адресов регистров: с 20001 по 29999. Имеет функции: «01» — чтения группы регистров, «05» — запись одного регистра, «15» — запись группы регистров
- **Input Registers** — 16-битные входы устройства. Доступны только для чтения. Диапазон адресов регистров: с 30001 по 39999. Имеют функцию: «04» — чтение группы регистров
- **Holding Registers** — 16-битные выходы устройства, либо внутренние значения. Доступны для чтения и записи. Диапазон адресов регистров: с 40001 по 49999. Имеют

Несмотря на названия, входы и выходы могут на самом деле являться внутренними переменными, хранить счетчики, флаги, или быть управляющими триггерами. В разных устройствах могут быть задействованы разные диапазоны регистров, или же все сразу.

Modbus на Arduino

Для использования интерфейса RS-485 в плате Arduino можно использовать модуль MAX485, в основе которого лежит микросхема преобразования UART в RS485. Модуль является двунаправленным и обеспечивает последовательную связь на расстоянии до 1200 метров. В полудуплексном режиме он обеспечивает скорость передачи данных 2,5 Мбит/с.

Модуль использует питающее напряжение 5V и логический уровень напряжения также 5V, что позволяет без проблем подключать его к платам Arduino.

Данный модуль имеет следующие особенности:

- работает с напряжениями 5V;
- имеет в своем составе чип MAX485;
- отличается низким энергопотреблением;
- всеми его контактами можно управлять с помощью микроконтроллера;
- размеры платы модуля: 44 x 14mm.

Внешний вид модуля RS-485 показан на следующем рисунке.

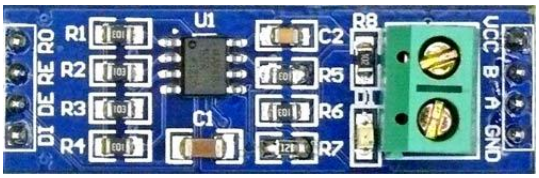

Рисунок 8 - Модуль RS-485

Назначение контактов модуля MAX485 приведена в следующей таблице.

Таблица 1 – Распиновка модуля MAX485

Название контакта	Назначение контакта
VCC	5V
A	вход/выход линии RS-485
B	вход/выход линии RS-485
GND	GND (0V)
R0	выход приемника (RX pin)
RE	разрешение работы приемника
DE	разрешение работы передатчика
DI	вход передатчика (TX pin)

Для передачи данных ведомой плате Arduino с помощью интерфейса RS-485 можно использовать программу Simply Modbus Master.

Перед началом работы необходимо ознакомиться со следующими терминами:

Slave ID (идентификатор ведомого)

Каждому ведомому устройству в сети назначается уникальный адрес в диапазоне от 1 до 127. Когда ведущее устройство запрашивает данные, то первый байт, который он передает, содержит адрес ведомого устройства. Благодаря этому каждое ведомое устройство знает стоит ли ему отвечать на этот запрос или нет.

Function code (функциональный код)

Второй байт, передаваемый ведущим, содержит функциональный код. Этот код определяет действие, которое необходимо выполнить (считать, записать и т.д.). Действия сгруппированы по таблицам. В протоколе Modbus существует четыре таблицы с данными:

Таблица 2 – Таблицы данных в Modbus

Таблица	Тип элемента	Тип доступа
Дискретные входы (Discrete Inputs)	один бит	только чтение
Регистры флагов (Coils)	один бит	чтение и запись
Регистры ввода (Input Registers)	16-битное слово	только чтение
Регистры хранения (Holding Registers)	16-битное слово	чтение и запись

На практике чаще всего встречаются устройства, в которых есть только таблица Holding Registers, иногда объединённая с таблицей Input Registers.

Для доступа к этим таблицам существует ряд стандартный функций ModBus:

Чтение:

1 (0x01) — чтение значений из нескольких регистров флагов (Read Coil Status).

2 (0x02) — чтение значений из нескольких дискретных входов (Read Discrete Inputs).

3 (0x03) — чтение значений из нескольких регистров хранения (Read Holding Registers).

4 (0x04) — чтение значений из нескольких регистров ввода (Read Input Registers).

Запись одного значения:

5 (0x05) — запись значения одного флага (Force Single Coil).

6 (0x06) — запись значения в один регистр хранения (Preset Single Register).

Запись нескольких значений:

15 (0x0F) — запись значений в несколько регистров флагов (Force Multiple Coils)

16 (0x10) — запись значений в несколько регистров хранения (Preset Multiple Registers)

Наиболее часто используемые на практике функции (функциональные коды) ModBus это 3, 6 и 16 («Read Holding Registers», «Preset Single Register» и «Preset Multiple Registers» — соответственно).

CRC

CRC расшифровывается как Cyclic Redundancy check и переводится как «циклический избыточный код». Это два байта, которые добавляются к каждому передаваемому сообщению протокола Modbus для обнаружения ошибок.

Программирование Arduino

Для использования протокола Modbus в плате Arduino Uno мы подключим в программе библиотеку <ModbusRtu.h>. Эта библиотека применяется для связи с ведущим или ведомым устройством Modbus RS-485 при помощи протокола RTU.

Пример программы для Arduino

```
//Circuit Digest
#include<ModbusRtu.h>       //Library for using Modbus in Arduino
#include<LiquidCrystal.h>   //Library for using 16x2 LCD display
#include <Servo.h>          //Library for using Servo Motor
#define led1 2              //Define as 2 led1
#define led2 5              //Define as 5 led2
LiquidCrystal lcd(8,9,10,11,12,13);   //initizlize lcd object with pins
(RS,E,D4,D5,D6,D7) for class liquid crystal
Servo servo;                          //Initilize servo object for class
Servo
Modbus bus;                          //Define Object bus for class
modbus
uint16_t modbus_array[] = {0,0,0};    //первоначально в массив
записываем нулевые значения

void setup()
{
  lcd.begin(16,2);                //режим 16x2 для ЖК дисплея
  lcd.print("RS-485 Modbus");     //приветственное сообщение
  lcd.setCursor(0,1);
  lcd.print("Arduino Slave");
  delay(5000);
  lcd.clear();

  pinMode(led1,OUTPUT);           //Led1 на вывод данных
  pinMode(led2,OUTPUT);           //Led2 на вывод данных
  servo.attach(6);                //Servo PWM pin 6 (контакт, к которому
подключен сервомотор)
  bus = Modbus(1,1,4);            //Modbus slave ID as 1 and 1 connected
via RS-485 and 4 connected to DE & RE pin of RS-485 Module
//адрес ведомого равен 1, используется связь через интерфейс RS-485
(вторая 1), 4 - номер контакта Arduino, к которому подключены контакты
DE & RE модуля RS-485
  bus.begin(9600);                //Modbus slave baudrate at 9600
(скорость 9600 бод)
}
void loop()
{
   bus.poll(modbus_array,sizeof(modbus_array)/sizeof(modbus_array[0]));
//Used to receive or write value from Master

  if (modbus_array[0] == 0)     //Depends upon value in modubus_array[0]
written by Master Modbus
  {
    digitalWrite(led1,LOW);     //выключаем первый светодиод
    lcd.setCursor(0,0);
    lcd.print("L1:OFF");
  }
  else
  {
     digitalWrite(led1,HIGH);  // включаем первый светодиод
     lcd.setCursor(0,0);
     lcd.print("L1:ON");
  }
 if (modbus_array[1] == 0)     //Depends upon value in modbus_array[1]
written by Master Modbus
  {
    digitalWrite(led2,LOW);    //LED OFF if 0
    lcd.setCursor(8,0);
    lcd.print("L2:OFF");
  }
```

```
  else
  {
     digitalWrite(led2,HIGH);  //LED ON if value other than 0
     lcd.setCursor(9,0);
     lcd.print("L2:ON");
  }

  int pwm = modbus_array[2]; //Depends upon value in modbus_array[1]
written by Master Modbus

  servo.write(pwm);           // поворачиваем сервомотор на угол pwm (от
0 до 180 градусов), принятый  от Modbus Master
  lcd.setCursor(0,1);
  lcd.print("Servo angle:");
  lcd.print(pwm);              //выводим значение угла на экран ЖК
дисплея 16x2
  delay(200);
  lcd.clear();
     }
```

Схема к программе

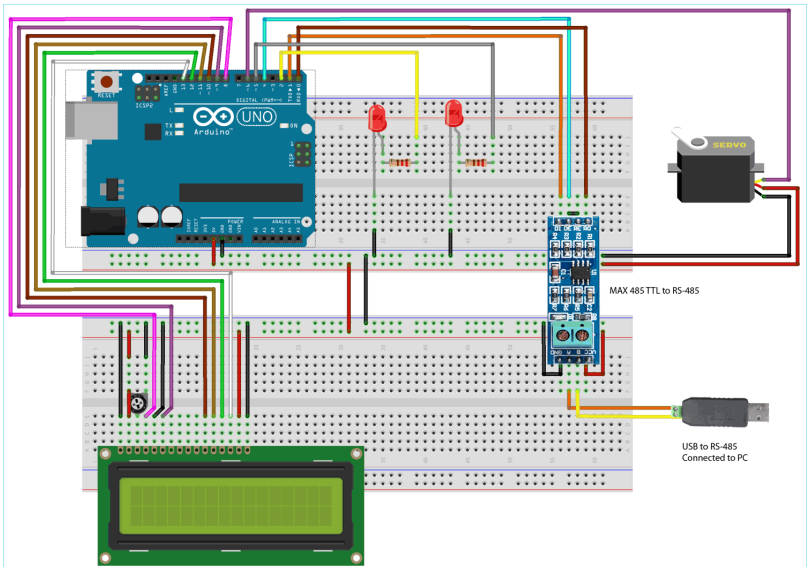

Рисунок 9 - Схема к примеру использования RS-485 и Modbus в Arduino

Практическая часть

1. С использованием библиотеки ModbusRtu.h и других необходимых библиотек запустите проект, описанный в теоретической части. Настройте программу Simply Modbus Master для работы с Arduino в режиме slave.

2. Развить задачу 1 на случай нескольких slave устройств. Все slave устройства должны быть оснащены дисплеями 16x2, сервоприводом и двумя светодиодами (см. Рисунок 9).

3. Реализовать режим master на Arduino и подключить к нему несколько slave устройств из задачи 2.