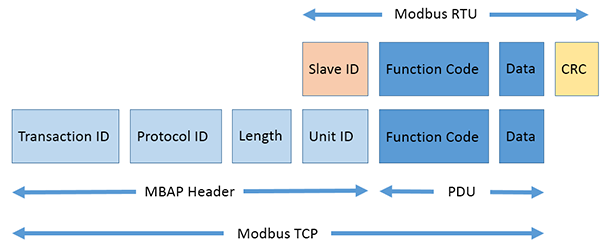
За основу можно взять протоколы Modbus RTU и Modbus TCP



Но сконфигурировав его из необходимых нам частей, а именно:

Length + Data[0] + Data[1…] + CRC[Length + Data]

1 байт + 1 байт + до 255 + 1 байт

На основе существующего варианта:

Входящие команды для микроконтроллера

Всего две команды + можно добавить третью:

1. 0 1 0 0 0 0 0 0 (0x80) - данные от энкодера
2. 0 0 x x x x x x - данные для управления двигателем

| | | | | |

| | | | | DIR

| | | | STEP

| | | MS3

| | MS2

| MS1

ENABLE

1. 0 1 x x x x x x - данные для управления двигателем + данные от энкодера после поворота? Есть некоторая сложность в отсутствии понимания, когда двигатель закончит поворот.
2. 1 0 0 0 0 x x x - управление светодиодами (1 - ON, 0 - OFF)

| | |

| | LED1

| LED2

LED3

Ответные команды:

1. От энкодера максимальное значение 1024 - 11 бит, то есть два байта по UART: [0000 0XXX] [XXXX XXXX]

Таким образом получим:

Для простых команд для микроконтроллера:

Length + Data[0] + Data[1] + CRC

0x02 + 0xXX + 0xXX + 0xXX

| |

| команда

номер сообщения

Ответ “наверх”:

Length + Data[0] + Data[1] + Data[2] + CRC

0x03 + 0xXX + 0x0X + 0xXX + 0xXX

| | |

| | младший байт от энкодера

| старший байт от энкодера

номер сообщения

/\* Функция расчета контрольной суммы на языке C.

Алгоритм: CRC-8 Полином: 0x31 x^8 + x^5 + x^4 + 1

Начальное значение: 0x5A

Проверочное значение: 0x94 («123456789») \*/

unsigned char Crc8(unsigned char \*pcBlock, unsigned int len)

{

unsigned char crc = 0x5A;

unsigned int i;

while (len--)

{

crc ^= \*pcBlock++;

for (i = 0; i < 8; i++)

crc = crc & 0x80 ? (crc << 1) ^ 0x31 : crc << 1;

}

return crc;

}

Bit-bang команды:

Добавить сложные команды:

1. Конфигурирование режима и направления вывода
2. Чтение состояния вывода
3. Установка состояния вывода
4. Инвертирование состояния вывода

[0] байт - команда

[1] байт - выбор порта (PA/PB/PC)

[2] байт - выбор пина (0...15)

[3…] байты - настройки конфигурирования