Данная библиотека предназначена для работы с Ethernet чипом W5500. Она позволяет быстро настроить чип под выполнение базовых функций принятия и отправки данных.

В неё входят все необходимые функции чтения/записи регистров и структуры. Все они имеют префикс W5500 в названии.

Принципиально чип W5500 делится на две основные области – это главный регистр и регистры сокетов.

Главный регистр один и хранит в себе основную информацию: IP, mac, маска подсети, адрес шлюза и т.д.

Сокетов на данной модели 8, каждый из них может работать параллельно с другими. У каждого сокета задаётся: объём памяти буфера принятия и отправки данных, порт, режим работы с пакетами UDP или TCP и т.д.

Далее будут описаны структуры, функции и алгоритмы использования без углубления в техническую составляющую в большинстве случаев. Более подробные характеристики по W5500 смотрите в документации производителя.

**Структуры:**

**W5500\_Socket – структура с основными регистрами сокетов, содержит в себе:**

Num – Номер сокет, отсчёт идёт от 0 до 7

Mode- Режим работы сокета, UDP или TCP

Status – Последний статус сокета, о нём далее будет сказано

Last interrupt – Последнее прерывание, подробнее в разделе прерываний

Port – Порт сокета, каждый сокет может иметь свой порт

RX buf size – Размер буфера приёма данных, размер может быть только: 0, 2, 4, 8, 16

TX buf size – Размер буфера отправки данных, размер может быть только: 0, 2, 4, 8, 16

TX rd p, TX wr p, RX rd p, RX wr p – указатели на точки начала чтения и записи. НЕ МЕНЯЙТЕ их без знания технической документации! За более подробной информацией обратитесь к даташиту.

Keep alive timer – Время между отправками служебного пакета, подтверждающего наличие TCP соединения клиента с сервером. Считается как 5s \* ваше число

Dipr – IP получателя, может быть изменён при отправке данных серверу

Dport – порт получателя, может быть изменён при отправке данных серверу

**W5500\_User\_Funcs – структура содержащая пользовательские функции И массив из 8 сокетов:**

SPI transmit func – Пользовательская функция отправки данных через SPI. Её сигнатура: (const uint8\_t\* data, uint16\_t len)

SPI receive func - Пользовательская функция принятия данных через SPI. Её сигнатура: (uint8\_t\* buffer, uint16\_t len)

CS func – Пользовательская функция Chip select, опускает и поднимает линию CS. Её сигнатура: (uint8\_t type\_operation): Если передаётся 1 , то линия опускается и чип “выбран”. Если передаётся 0, то линия поднимается и чип ”не выбран”. Другими словами 1 перед отправкой, 0 после отправки.

Delay func – функция задержки, должна принимать целое значение в миллисекундах. Её сигнатура: (uint32\_t delay), delay в миллисекундах

**W5500\_User\_Callbacks – структура содержащая пользовательские функции обработки 5 прерываний:**

Callback Send Ok – прерывание на успешную отправку данных от W5500 к получателю.

Callback Timeout – прерывание, происходящее при ARP timeout и TCP timeout , когда устройство не получает ответа от удаленного устройства в течение заданного времени. Задаётся через регистры RTR И RCR, по умолчанию не изменяются. Если вы хотите изменить время между повторными отправками и счётчиком повторных отправок, то используйте следующие функции: W5500\_Set\_RTR(uint16\_t retry\_time, W5500\_User\_Funcs\* UF) – устанавливает следующее время rtr = 100 **мкс** \* retry\_time. Пример: retry\_time = 4000, следовательно rtr = 100 мкс \* 4000 = 400 мс uint8\_t W5500\_Set\_RCR(uint8\_t retry\_count, W5500\_User\_Funcs\* UF) – устанавливает максимальное кол-во RTR, то есть если после retry\_count ретрансляций “не был получен ответ”, то инициируется прерывание Timeout.

Callback Discon – прерывание при отключении, справедливо только для TCP режима сокета

Callback Con – прерывание при подключении к серверу, справедливо только для TCP режима сокета

Все вышеперечисленные функции обработки прерывания имеют следующую сигнатуру: (W5500\_Socket\* socket, W5500\_User\_Funcs\* UF)

Callback Recv – прерывание при получении данных. Если вы выставили сокет в TCP сервер или выставили в UDP режим, и вам пришли данные, то вызывается прерывание Recv. Его сигнатура: (uint8\_t\* buf, uint8\_t size, W5500\_Socket\* socket, W5500\_User\_Funcs\* UF) В теле обработчика используйте функцию: uint16\_t W5500\_ReceiveData(uint8\_t socket\_num, uint8\_t\* buffer, uint16\_t max\_len, W5500\_User\_Funcs\* UF) Она возвращает размер данных в байтах. Пример 1: вы передали буфер длинной в 20 байт, в результате функция вернула вам значение 15, следовательно в вашем буфере сейчас 15 байт информации и 5 незначащих. Пример 2: вы передали буфер длинной в 20 байт, в результате функция вернула вам значение 47, следовательно в вашем буфере 20 байт из W5500 и ещё 47-20= 27 байт не прочитаны, вы можете повторно вызвать эту функцию два раза и прочитать оставшиеся данные. При новом вызове вы продолжите читать с того места, на котором остановились в прошлый раз.

**Маска прерываний.** По умолчанию при инициализации сокетов все прерывания становятся активными. Это не отразится негативно на работе, если ваш сокет в режиме TCP Сервера, то прерывание Send Ok не возникнет, также и наоборот с TCP клиентом и прерыванием Recv.

При UDP режиме у вас могут возникнуть прерывания Send Ok и Recv, но без Con и Discon. Другими словами, хоть все прерывания и активны, это не значит, что при UDP режиме у вас возникнет Con прерывание. Однако если вы желаете строго задать маску, то вам следует:

Подать в функцию битовую последовательность, где 1 – вкл, 2 выкл

uint8\_t W5500\_Set\_Sn\_IMR(uint8\_t socket\_num, uint8\_t interrupt\_mask, W5500\_User\_Funcs\* UF)

Порядок битов и прерываний:

Con = 0b00001 0x1

Discon = 0b00010 0x2

Recv = 0b00100 0x4

Timeout = 0b01000 0x8

Send Ok = 0b10000 0x10

Пример включения всех прерываний: interrupt\_mask = 0b11111 = 0x1F

Пример включения только Send Ok и Discon: interrupt\_mask = 0b10010 = 0x12

**W5500\_Main\_Struct – главная структура, включающая указатели на W5500\_User\_Funcs и W5500\_User\_Callbacks**

W5500\_User\_Funcs\* UF – указатель на объект структуры W5500\_User\_Funcs

W5500\_User\_Callbacks\* UCb – указатель на объект структуры W5500\_User\_Callbacks

**Основные функции:**

**Функция быстрой инициализации главного регистра**

uint8\_t W5500\_QuickInit\_Common(

const uint8\_t\* mac\_address, - мак адрес чипа

const uint8\_t\* ip\_address, - IP адрес чипа

const uint8\_t\* subnet\_mask, - маска подсети

const uint8\_t\* gateway\_address, - адрес шлюза

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

Возвращает 0, если успешно. В противном случаи возвращает число, указывающее на позицию в ф теле функции где произошла ошибка. Для больших подробностей смотрите тело функции.

**Функция быстрой инициализации сокета в UDP режиме**

uint8\_t W5500\_QuickInit\_UDP(

uint8\_t socket\_num, - номер сокета, от 0 до 7

uint16\_t src\_port, - порт сокета

uint16\_t tx\_b\_s, - размер буфера отправки, принимает 0, 2, 4, 8, 16

uint16\_t rx\_b\_s, - размер буфера принятия, принимает 0, 2, 4, 8, 16

uint8\_t keep\_alive\_timer, - не используется

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

В данном режиме сокет всегда принимает входные данные

Возвращает 0, если успешно. В противном случаи возвращает число, указывающее на позицию в ф теле функции где произошла ошибка. Для больших подробностей смотрите тело функции.

**Функция быстрой инициализации сокета в TCP режиме**

uint8\_t W5500\_QuickInit\_TCP(

uint8\_t socket\_num, - номер сокета, от 0 до 7

uint16\_t src\_port, - порт сокета

uint16\_t tx\_b\_s, - размер буфера отправки, принимает 0, 2, 4, 8, 16

uint16\_t rx\_b\_s, - размер буфера принятия, принимает 0, 2, 4, 8, 16

uint8\_t keep\_alive\_timer, - не используется

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

В данном режиме сокет по умолчанию выступает клиентом и не принимает входные данные, для перехода в режим сервера смотрите W5500\_TCP\_Listen

Возвращает 0, если успешно. В противном случаи возвращает число, указывающее на позицию в ф теле функции где произошла ошибка. Для больших подробностей смотрите тело функции.

**Функция TCP подключения (ВКЛЮЧЕНО В ФУНКЦИИ ОТПРАВКИ И НЕ ТЕРБУЕТСЯ ПЕРЕД КАЖДОЙ ОТПРАВКОЙ)**

uint8\_t W5500\_TCP\_Connect(

uint8\_t socket\_num, - номер сокета

const uint8\_t\* dest\_ip, - IP адрес сервера

uint16\_t dest\_port, - порт сервера

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

Возвращает 0, если успешно. В противном случаи возвращает число, указывающее на позицию в теле функции где произошла ошибка. Для больших подробностей смотрите тело функции.

**Функция отправки данных без прерывания**

uint8\_t W5500\_SendData(

uint8\_t socket\_num, - номер сокета

uint8\_t func\_mode, - режим записи IP и порта

const uint8\_t\* dest\_ip, - IP адрес сервера

uint16\_t dest\_port, - порт сервера

const uint8\_t\* data, - указатель на данные

uint16\_t len, - длинна данных

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

func\_mode - режим выполнения записи IP, порта и закрытие сокета

Если выставить 1 на место:

0-й бита - не перезаписывать IP и порт (меньше нагрузка на SPI)

1-й бита - перезаписать только IP

2-й бита - перезаписать только порт

1-й и 2-й бита - перезаписать IP и порт

4-й бит - закрыть сокет до и после отправки, РЕКОМЕНДУЕТСЯ для TCP, иначе выполните это сами

Данная функция отправляет данные без пользовательской обработки прерывания Send Ok, она проверяет наличие этого прерывания самостоятельно каждые 200 мс 10 раз.

Возвращает 0 при успешной отправке, 1 при неудачном подключении (если сокет TCP) и 2 при неудачной отправке

**Функция отправки данных без прерывания**

uint8\_t W5500\_SendData\_IR(

uint8\_t socket\_num, - номер сокета

uint8\_t func\_mode, - режим записи IP и порта

const uint8\_t\* dest\_ip, - IP адрес сервера

uint16\_t dest\_port, - порт сервера

const uint8\_t\* data, - указатель на данные

uint16\_t len, - длинна данных

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

func\_mode – аналогичен

Данная функция отправляет данные с учётом пользовательской обработки прерывания Send Ok и не проверяет состояние прерывания.

Возвращает 0 при конце отправке (результат отправки проверяется в прерывании), 1 при неудачном подключении (если сокет TCP)

ВАЖНО: Если сокет в режиме TCP, то перед каждой отправкой сокет открывается и закрывается, соединение с сервером так же разрывается и заново подключается.

**Функция принятия данных**

uint16\_t W5500\_ReceiveData(

uint8\_t socket\_num, - номер сокета

uint8\_t\* buffer, - указатель на начало буфера

uint16\_t max\_len, - длина буфера

W5500\_User\_Funcs\* UF) - указатель на объект структуры W5500\_Main\_Struct

Данная функция принимает данные из сокета

Возвращает размер данных в байтах.

Пример 1: вы передали буфер длинной в 20 байт, в результате функция вернула вам значение 15, следовательно в вашем буфере сейчас 15 байт информации и 5 незначащих.

Пример 2: вы передали буфер длинной в 20 байт, в результате функция вернула вам значение 47, следовательно в вашем буфере 20 байт из W5500 и ещё 47-20= 27 байт не прочитаны, вы можете повторно вызвать эту функцию два раза и прочитать оставшиеся данные.

При новом вызове вы продолжите читать с того места, на котором остановились в прошлый раз.

Если вы не используете прерывания, то должны самостоятельно читать состояние регистра прерываний или периодично вызывать данную функцию, пока она не будет возвращать значение 0.

**Функция TCP сервера**

void W5500\_TCP\_Listen(

uint8\_t socket\_num, - номер сокета

W5500\_Main\_Struct\* MS); - указатель на объект структуры W5500\_Main\_Struct

Данная функция переводит сокет c режимом TCP в TCP cервер, предварительно закрывая и открывая сокет в режиме прослушивания канала.

Если вы хотите выйти из режима TCP сервера, закройте сокет.

Ничего не возвращает.

**Функция открытия сокета**

uint8\_t W5500\_OpenSocket(

uint8\_t socket\_num, - номер сокета

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

Открывает сокет, по умолчанию после инициализации UDP или TCP сокет открыт

Возвращает 0 при удачной отправке команды через SPI, 1 при неудачной

**Функция закрытия сокета**

uint8\_t W5500\_CloseSocket(

uint8\_t socket\_num, - номер сокета

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

Закрывает сокет и отчищает регистр прерываний сокета

Возвращает 0 при удачном закрытии или статус сокета при неудачном. Подробности статусов смотрите в даташите Sn\_Sr

**Функция отключения от сервера**

uint8\_t W5500\_TCP\_Disconnect(

uint8\_t socket\_num, - номер сокета

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

Разрывает соединение с сервером и закрывает сокет

Возвращает 0 при удачном отключении и 1 при неудачном

**Функция обработки прерываний**

void W5500\_IR\_processing(

uint8\_t\* buf, - указатель на буфер приём

uint8\_t size, - размер буфера приёма

W5500\_Main\_Struct\* MS) - указатель на объект структуры W5500\_Main\_Struct

Данная функция считывает значение регистра прерываний сокетов, после проходит по всем сокетам и обрабатывает их прерывания через пользовательские функции обработки.

Пример: вы проинициализировали сокет 3 и 4. После вы фиксируете прерывание по каналу INT чипа и вызываете функцию W5500\_IR\_processing. Она получает от W5500 номера сокетов, на которых произошло прерывание. После у каждого из этих сокетов (в данном случаи у 3-го или 4-го, или у обоих) берётся тип прерывания, а после вызывается пользовательский обработчик, например обработчик Cb\_RECV Callback\_Recv для 3-го сокета и Cb\_SEND\_OK Callback\_Send\_OK для 4-го.

**Функция чтения регистра прерываний сокета**

uint8\_t W5500\_Get\_Sn\_IR(

uint8\_t socket\_num, - номер сокета

W5500\_User\_Funcs\* UF) - - указатель на объект структуры W5500\_User\_Func

Данная функция позволяет вручную проверить состояние регистра прерываний конкретного сокета.

Возвращает битовую последовательность прерывания или 0xFF в случаи ошибки

Порядок битов и прерываний:

Con = 0b00001 0x1

Discon = 0b00010 0x2

Recv = 0b00100 0x4

Timeout = 0b01000 0x8

Send Ok = 0b10000 0x10