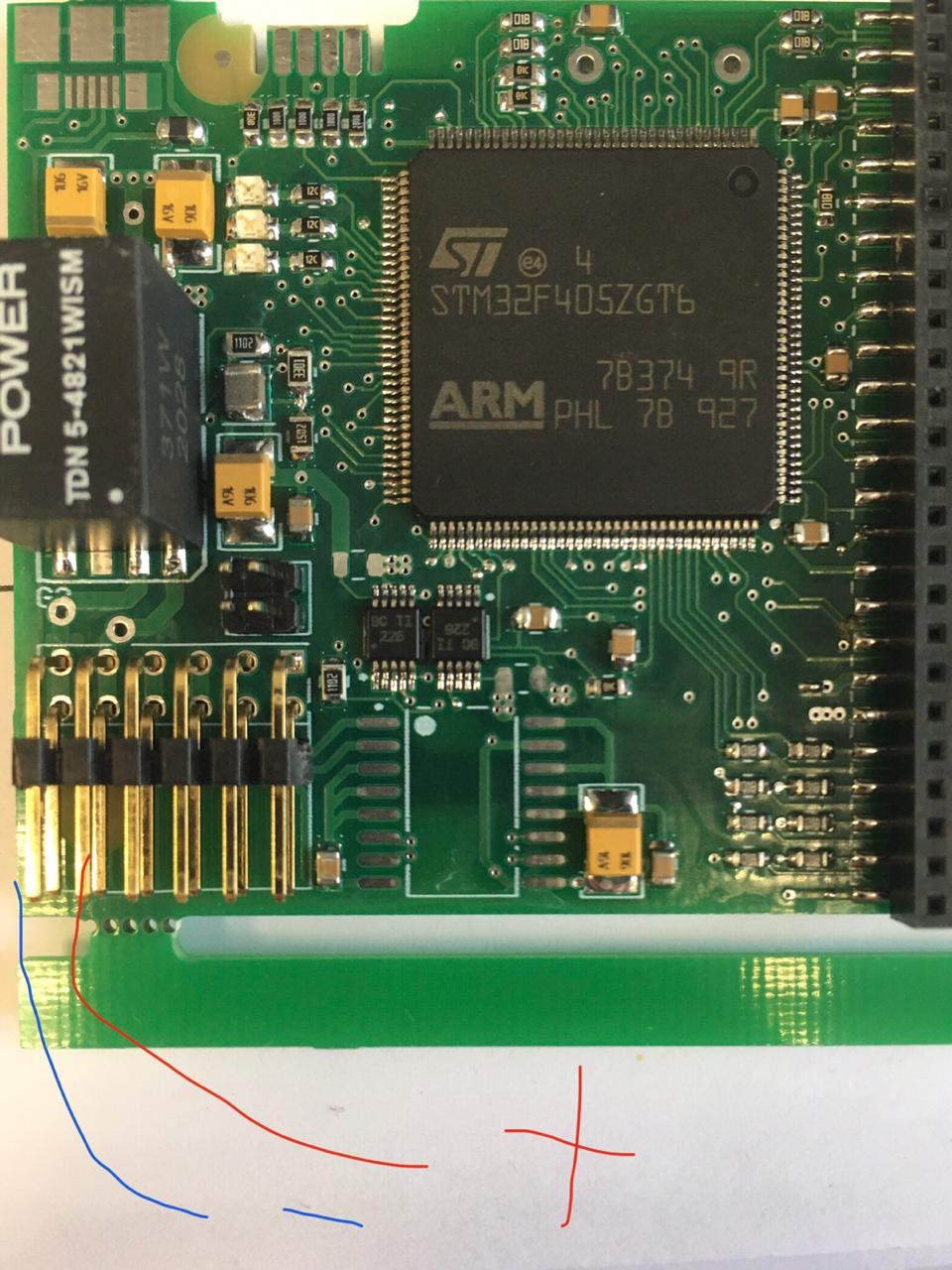
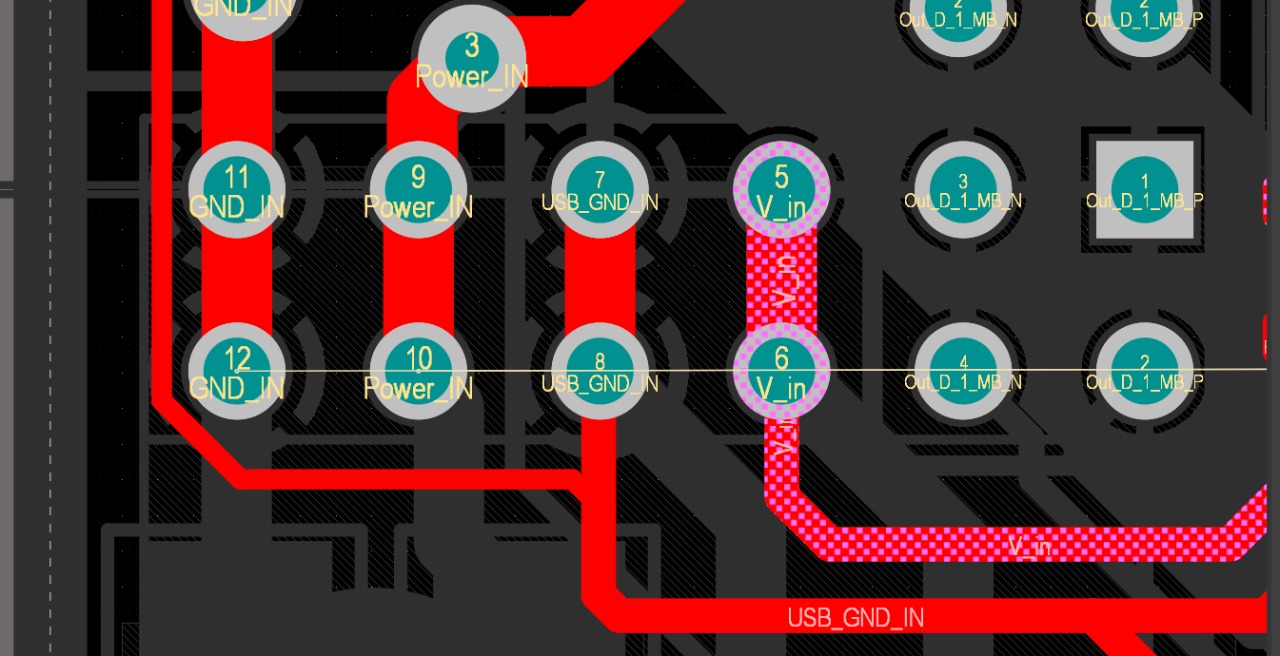
Цифровой модуль запитывается от внешнего источника питания, напряжением от 20 до 48 Вольт



На GND\_IN подать GND, На POWER\_IN подать +20-48В.

После подачи питания, должен загореться нижний красный светодиод, сигнализирующий о том, что есть питание по 3.3 В.

Модуль подключается к ПК посредством USB разъема, находящегося в верхней части слева.

Модуль определяется как Virtual Com Port, у каждого модуля есть свой идентификатор который можно узнать при помощи базового ПО.

В модуле реализована система команд по протоколу ModbusRTU, подробно можно ознакомиться здесь <https://ipc2u.ru/articles/prostye-resheniya/modbus-rtu/>

В модуле реализованы команды для чтения и записи аналоговых входов и аналоговых выходов. Дискретные входы/выходы не используются.

Управление модулем и получение данных с модуля осуществляется посредством записи и чтения соответствующих регистров.

Регистр представляет собой 16 битное слово, хранящееся по некому адресу.

Регистры в памяти структурированы карта регистров представлена ниже.

Необходимо помнить что данные в модуле хранятся в формате младший байт впереди. это учтено в базовом ПО верхнего уровня. Необходимо выставить флаг byte order revers.

В модуле представлены следующие регистры:  
Регистры АЦП. Позволяют вычитывать данные АЦП, Запускать и останавливать АЦП.

Регистры ЦАП. Позволяют генерировать сигнал сложной формы, запускать и останавливать ЦАП.

Регистры GPIO. Позволяют конфигурировать GPIO как входы или выходы. Получить текущее состояние GPIO

Регистры UART1, UART2. Позволяют настроить UART1/2 независимо друг от друга (Baudrate, Бит четности и пр). Позволяют производить отправку пакетов длиной до 128 байт. Прием осуществляется в кольцевой буфер размером 2048 байт.

Основная структура хранения данных в модуле

Справа обозначено смещение данной структуры в байтах и регистрах.

typedef struct

{

// analog in аналоговые входы.

type\_adc\_data\_struct mb\_adc; /+0

type\_ina\_226\_data ina226\_3v3; //+16 bytes //+8 regs

type\_ina\_226\_data ina226\_5v; //+22 bytes //+11 regs

type\_uart\_recive\_struct mb\_uart1\_recive\_struct; //+28 bytes //+14 regs

type\_uart\_recive\_struct mb\_uart2\_recive\_struct; //+2084 bytes //+1042 regs

uint16\_t dummy1[MB\_DATA\_SIZE - (sizeof(type\_adc\_data\_struct)/2) -(sizeof(type\_uart\_recive\_struct)) - (sizeof(type\_ina\_226\_data))]; // массив для выравнивания

//analog\_out аналоговые выходы

type\_dac\_data\_struct mb\_dac1; //+0 структура управления ЦАП 1

type\_dac\_data\_struct mb\_dac2; // +1056 bytes // + 528 regs 0x210 структура управления ЦАП 2

type\_adc\_settings mb\_adc\_settings;// +2112 bytes // + 1056 regs 0x420 сруктура управления АЦП

type\_gpio\_config\_union mb\_gpio\_config\_union; //+2118bytes// + 1059 regs 0x423 структура управления GPIO

type\_gpio\_out\_union mb\_gpio\_out\_union; //+2128 bytes // + 1064 regs 0x428 структура выставления значений на GPIO

type\_gpio\_in\_union mb\_gpio\_in\_union; //+2136 bytes // + 1068 regs 0x42C структура считывания значений GPIO

type\_uart\_transmit\_struct mb\_uart1\_transmit\_struct; //+2144 bytes // +1072 regs 0x430 передача данных UART1

type\_uart\_setting\_union mb\_uart1\_setting\_struct; // +2280 bytes // + 1140 regs 0x474 настройки UART1

type\_uart\_transmit\_struct mb\_uart2\_transmit\_struct; // +2300 bytes // + 1150 regs 0x47E передача данных UART2

type\_uart\_setting\_union mb\_uart2\_setting\_struct; // +2436 bytes // + 1218 regs 0x4C2 настройка UART2

uint16\_t dummy2[MB\_DATA\_SIZE - (sizeof(type\_uart\_setting\_union)/2) - (sizeof(type\_dac\_data\_struct)/2)-(sizeof(type\_dac\_data\_struct)/2) - (sizeof(type\_adc\_settings)/2)-(sizeof(type\_gpio\_config\_union)/2) - (sizeof(type\_gpio\_out\_union)/2) - (sizeof(type\_gpio\_in\_union)/2)]; // sizeof in bytes, uint8, massive is uint16 // массив для выравнивания

uint16\_t dummy3[MB\_DATA\_SIZE];

uint16\_t dummy4[MB\_DATA\_SIZE];

}type\_modbus\_data\_named;

Структура данных АЦП

Регистр данных.

Хранит данные 8 каналов АЦП. Обновляется 1 раз в 500 мс.

{

uint16\_t data[8]; // +0

}

type\_adc\_data\_struct;

Хранятся данные АЦП. Data[0] ADC in 1 Data[1] ADC in 2 и тд.

Разрядность АЦП 12 бит. 0xFF соответствует 2.048 Вольт.

Регистр управления АЦП

{

uint16\_t settings\_scaler; //+0

uint16\_t start; //+2 байта // +1 регистр

uint16\_t settings; //+4 байта // +2 регистра

}type\_adc\_settings;

settings\_scaler – регистр который служит счетчиком. В случае обновления данных в регистре управления, счетчик необходимо инкрементировать на 1.

Start – если Start == 0x01, АЦП запущен, 0x00 остановлен. По умолчанию АЦП запущен.

Settings – резерв

Данные сдатчиков INA226 (собственное питание по 3.3 В и 5 В)

{

int16\_t voltage; //+0

int16\_t current; //+2 байта // +1 регистр

int16\_t power; //+4 байта // +2 регистра

}

type\_ina\_226\_data;

Voltage напряжение в единицах LSB

Current ток в единицах LSB

Power – резерв

UART

{

uint16\_t scaler; // +0

uint16\_t start; //+2 bytes // +1 regs

uint16\_t transmit\_flag; // +4 bytes //+2 regs

uint16\_t len; // +6 bytes // +3 regs

uint8\_t data[UART\_TRANSMIT\_DATA\_BUFF]; //+ 8 bytes // +4 regs

}

type\_uart\_transmit\_struct;

scaler – счетчик обновления структуры, необходим для того чтобы МК увидел обновление этой структуры. инкрементируется пользователем

start – если равен 0x01 будет отправлен пакет длиной len

transmit\_flag – по завершению отправки будет выставлен в 0x01

len – длина пакета на отправку в байтах. Если превысить 128. Будет отправлен мусор. Пофиксю в будущем.

Data[] данные на отправку. Отправляются с нулевого по len.

typedef struct

{

uint16\_t scaler;

uint16\_t start;

uint16\_t write\_ptr;

uint16\_t len;

uint8\_t data[UART\_RECIVE\_DATA\_BUFF];

}

type\_uart\_recive\_struct;

scaler – резерв

start – резерв

write\_ptr – указывает до какого байта в данных произведена запись. Следующий пришедший байт будет записан в этот элемент массива.

Len – резерв

Data – собственно данные. Длина 2048.

typedef struct

{

uint16\_t scaler; //+0

uint16\_t LOW\_BAUD; //+2 bytes //+1regs

uint16\_t HIGH\_BAUD; // +4bytes // +2regs

uint16\_t LOW\_WORDlenght; //+6 bytes // +3 regs

uint16\_t HIGH\_WORDlenght; //+8 bytes // +4 regs

uint16\_t LOW\_STOPBITS; //+10 bytes // +5 regs

uint16\_t HIGH\_STOPBITS; //+12 bytes //+6 regs

uint16\_t LOW\_PARITY; //+14 bytes //+7 regs

uint16\_t HIGH\_PARITY; //+16 bytes //+8 regs

uint16\_t flag; //+18 bytes // 10regs

}

type\_uart\_setting\_struct;

LOW- младший регистр HIGH старший регистр

Регистры управления UART – 32 битные, соответственно чтобы получить 32 битный регистр, нужно записать два регистра Modbus

Регистры управления UART прописываются значениями по умолчанию. 115200.

Scaler - счетчик обновления структуры необходимо инкрементировать для обновления данных в памяти мк.

BAUD – баудрейт 28800/57600/115200 и тд. Можно выставить любой кроме 0. При записи 0,будут установлены значения по умолчанию.

WORDlenght –

STOPBITS –

PARITY –

Flag – выставляется в 0x01 когда настройки прописаны и UART запущен с новыми настройками. При изменении настроек, данные приемника перетираются нулями, указатель записи выставляется на 0.

GPIO

На плате цифрового модуля доступны 60 GPIO. Их можно конфигурировать по своему усмотрению как входы или как выходы. Выставлять значения 0/1 на выходах и считывать значения как со входов, так и текущее состояние выходов.

Регистры настройки GPIO.

Для того чтобы обновить конфигурацию GPIO необходимо записать 0x1 в Data\_updater

Init\_flag в регистре 1059 изменяет свое значение с 0 на 1 после завершения инициализации

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1059 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | | | | 6 | | 5 | 4 | 3 | | 2 | | 1 | | 0 | |
| reserved | | | | | | | | | | | | | | | | | | | | | | | | | | | | | Init\_flag | |
|  | | | | | | | | | | | | | | | | | | | | | | | | | | | | | 0/1 | |
| Reg num 1060 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | | | 6 | | | 5 | 4 | 3 | | 2 | | 1 | | 0 | |
| Data\_updater(scaler) | | | | | | | GPIO | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO1 | | 2 | | 3 | | 4 | | 5 | | | 6 | | | 7 | 8 | 9 | | 10 | | 11 | | GPIO12 | |
| 0/1 | | | | | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | | 0/1 | | | 0/1 | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | |
| Reg num 1061 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO13 | | 14 | | 15 | | 16 | | 17 | | 18 | | 19 | | 20 | | 21 | 22 | | | 23 | | 24 | | 25 | | 26 | | 27 | | GPIO28 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 1062 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO29 | | 30 | | 31 | | 32 | | 33 | | 34 | | 35 | | 36 | | 37 | 38 | | | 39 | | 40 | | 41 | | 42 | | 43 | | GPIO44 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 1063 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO45 | | 46 | | 47 | | 48 | | 49 | | 50 | | 51 | | 52 | | 53 | 54 | | | 55 | | 56 | | 57 | | 58 | | 59 | | GPIO60 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |

Регистры выставления GPIO

Для того чтобы обновить состояние GPIO необходимо записать 0x1 в Data\_updater  
После выставления значение будет изменено на 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1064 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | | | 6 | | 5 | 4 | 3 | | 2 | | 1 | | 0 | |
| Data\_updater(scaler) | | | | | | | GPIO | | | | | | | | | | | | | | | | | | | | | | |
| GPIO1 | | 2 | | 3 | | 4 | | 5 | | | 6 | | 7 | 8 | 9 | | 10 | | 11 | | GPIO12 | |
| 0-1 | | | | | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | | 0/1 | | 0/1 | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | |
| Reg num 1065 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO13 | | 14 | | 15 | | 16 | | 17 | | 18 | | 19 | | 20 | | 21 | 22 | | 23 | | 24 | | 25 | | 26 | | 27 | | GPIO28 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 1066 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO29 | | 30 | | 31 | | 32 | | 33 | | 34 | | 35 | | 36 | | 37 | 38 | | 39 | | 40 | | 41 | | 42 | | 43 | | GPIO44 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 1067 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO45 | | 46 | | 47 | | 48 | | 49 | | 50 | | 51 | | 52 | | 53 | 54 | | 55 | | 56 | | 57 | | 58 | | 59 | | GPIO60 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |

Регистры чтения GPIO находятся в области analog inputs

Регистры обновляются автоматически 1 раз в 500 мс. В регистрах хранится последнее считанное состояние.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 2070 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | | | 6 | | 5 | 4 | 3 | | 2 | | 1 | | 0 | |
| reserved | | | | | | | GPIO | | | | | | | | | | | | | | | | | | | | | | |
| GPIO1 | | 2 | | 3 | | 4 | | 5 | | | 6 | | 7 | 8 | 9 | | 10 | | 11 | | GPIO12 | |
| 0-16 | | | | | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | | 0/1 | | 0/1 | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | |
| Reg num 2071 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO13 | | 14 | | 15 | | 16 | | 17 | | 18 | | 19 | | 20 | | 21 | 22 | | 23 | | 24 | | 25 | | 26 | | 27 | | GPIO28 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 2072 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO29 | | 30 | | 31 | | 32 | | 33 | | 34 | | 35 | | 36 | | 37 | 38 | | 39 | | 40 | | 41 | | 42 | | 43 | | GPIO44 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 2073 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO45 | | 46 | | 47 | | 48 | | 49 | | 50 | | 51 | | 52 | | 53 | 54 | | 55 | | 56 | | 57 | | 58 | | 59 | | GPIO60 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |

Регистры альтернативного (временного) состояния GPIO

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1228 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Scaler | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Scaler – регистр запуска обновления команды.  
При записи 1 – обновляется и исполняется команда. После завершения команды регистр перезаписывается 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1229 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LOW\_Time | | | | | | | | | | | | | | | |
| Reg num 1230 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| HIGH\_Time | | | | | | | | | | | | | | | |

Регистры Low и High time вместе составляют 32 битную переменную определяющую время выставления альтернативного состояния. Время в микросекундах.

(uint32\_t) TIME = (HIGH\_time << 16 )| (LOW\_time)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1231 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Start | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |
| Reg num 1232 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| End\_flag | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |
| Reg num 1233 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Stop | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Регистры Start и Stop – обеспечивают запуск процесса выставления альтернативного состояния и принудительную остановку процесса соответственно. При записи 1 в Start запускается альтернативное состояние. При записи 1 в Stop процесс останавливается (даже если записана 1 в Start.

Регистр End\_Flag принимает значение 1 при штатном завершении процесса (прошло заданное время)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1234 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Process\_flag | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Регистр Proces\_flag = 1 на протяжении времени выставления альтернативного состояния GPIO.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1233 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| LOW\_Time\_left | | | | | | | | | | | | | | | |
| Reg num 1234 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| HIGH\_Time\_left | | | | | | | | | | | | | | | |

Регистры Low и High time left вместе составляют 32 битную переменную определяющую время которое осталось до конца выставления альтернативного состояния. Время в микросекундах. Обновляется 1 раз в 500 мс. Time = (HIGH\_Time\_left <<16)|(LOW\_time\_left)

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1242 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | | | 6 | | 5 | 4 | 3 | | 2 | | 1 | | 0 | |
| reserved | | | | | | | GPIO | | | | | | | | | | | | | | | | | | | | | | |
| GPIO1 | | 2 | | 3 | | 4 | | 5 | | | 6 | | 7 | 8 | 9 | | 10 | | 11 | | GPIO12 | |
| 0-16 | | | | | | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | | 0/1 | | 0/1 | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | |
| Reg num 1243 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO13 | | 14 | | 15 | | 16 | | 17 | | 18 | | 19 | | 20 | | 21 | 22 | | 23 | | 24 | | 25 | | 26 | | 27 | | GPIO28 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 1244 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO29 | | 30 | | 31 | | 32 | | 33 | | 34 | | 35 | | 36 | | 37 | 38 | | 39 | | 40 | | 41 | | 42 | | 43 | | GPIO44 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |
| Reg num 1245 | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| 15 | | 14 | | 13 | | 12 | | 11 | | 10 | | 9 | | 8 | | 7 | 6 | | 5 | | 4 | | 3 | | 2 | | 1 | | 0 |
| GPIO | | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| GPIO45 | | 46 | | 47 | | 48 | | 49 | | 50 | | 51 | | 52 | | 53 | 54 | | 55 | | 56 | | 57 | | 58 | | 59 | | GPIO60 |
| 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 | | 0/1 |

Регистры определяющие состояние GPIO на заданное время, аналогично предыдущим.

Регистры SPI

Регистры управления SPI. Находятся в области analog outputs.

SPI Settings

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1246 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Scaler | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Scaler – регистр запуска обновления команды.  
При записи 1 – обновляется и исполняется команда. После завершения команды регистр перезаписывается 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1247 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Mode | | | | | | | | | | | | | | | |
| Reg num 1248 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Direction | | | | | | | | | | | | | | | |
| Reg num 1249 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Polaryty | | | | | | | | | | | | | | | |
| Reg num 1250 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Phase | | | | | | | | | | | | | | | |
| Reg num 1251 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Slave | | | | | | | | | | | | | | | |
| Reg num 1252 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Baud | | | | | | | | | | | | | | | |
| Reg num 1253 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| First\_bit | | | | | | | | | | | | | | | |
| Reg num 1247 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Ti\_mode | | | | | | | | | | | | | | | |

mode master - 0, slave -1

direction 2Line - 0, 2Line\_rx\_only - 1, 1Line - 2

data\_size 8bit - 0, 16bit -1

polarity low - 0, high - 1

phase one\_edge - 0, two\_edge - 1

slave not\_used\_et

baud 0 - 21MB/s, 1 - 10,5 MB/s, 2 - 5,25 MB/s, 3 - 2,625 MB/s, 4 - 1,3125 MB/s, 5 - 656,25 kB/s, 6 - 328,125 kB/s, 7 - 164,062 kB/s.

firs\_bit 0 - MSB\_first, 1 - LSB\_first

ti\_mode 0 - disable, 1 – enable

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1248 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Init\_flag | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Перед инициализацией сбрасывается принудительно в 0. После инициализации в случае успеха выставляется 1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1249 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Set\_default | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Если выставлен в 1, то не зависимо от того что записано в предыдущие настроечных регистрах, в инициализацию будут прописаны настройки по умолчанию.

Master, 2Lines, 8bit, Polarity\_low, Phase\_1Edge, Baud\_rate – 21MB/s. TiMode – disable.

Spi\_transmit registers.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1266 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Scaler | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Scaler – регистр запуска обновления команды.  
При записи 1 – обновляется и исполняется команда. После завершения команды регистр перезаписывается 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1267 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| N/U | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

N/U не используется

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1268 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Len | | | | | | | | | | | | | | | |

Len – длина данных на отправку. В 16 битном режиме максимальное значение 32. В 8 битном режиме максимальное значение 64.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1269 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Transaction\_end | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Transaction\_end – перед началом транзакции выставляется в 0. По завершению транзакции выставляется в 1.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1270 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Rx\_tx\_flag | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Rx\_tx\_flag – флаг работы в режиме приемопередатчика (при настройке в две линии)   
0 – режим работы только передатчик. 1 – режим работы передатчик/приемник. Данные приема складываются в регистры данных приемника.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1276… | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Data[0] | | | | | | | | Data[1] | | | | | | | |
| 0-255 | | | | | | | | 0-255 | | | | | | | |
| …Reg num 1307 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Data[62] | | | | | | | | Data[63] | | | | | | | |
| 0-255 | | | | | | | | 0-255 | | | | | | | |

Регистры данных на отправку.

Регистры приемника SPI

Spi\_receive registers.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1308 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Scaler | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Scaler – регистр запуска обновления команды.  
При записи 1 – обновляется и исполняется команда. После завершения команды регистр перезаписывается 0.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1309 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| N/U | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

N/U не используется

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1310 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Len | | | | | | | | | | | | | | | |

Len – длина данных на прием. В 16 битном режиме максимальное значение 64. В 8 битном режиме максимальное значение 128.

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 1311 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Transaction\_end | | | | | | | | | | | | | | | |
| 0/1 | | | | | | | | | | | | | | | |

Transaction\_end – перед началом транзакции выставляется в 0. По завершению транзакции выставляется в 1.

Регистры данных приемника SPI находятся в области analog\_inputs

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Reg num 2074… | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Data[0] | | | | | | | | Data[1] | | | | | | | |
| 0-255 | | | | | | | | 0-255 | | | | | | | |
| …Reg num 2137 | | | | | | | | | | | | | | | |
| 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| Data[126] | | | | | | | | Data[127] | | | | | | | |
| 0-255 | | | | | | | | 0-255 | | | | | | | |