**1. Блок захвата данных с весов**

- Захватить и сохранить в памяти пакет подаваемый на дисплей

***LCD\_Data\_Capture(LCD\_bufer)***

LCD\_bufer – адрес ячейки начиная с которой располагается буфер для чтения данных – функция размещается в блоке обработки внешнего прерывания

**1. Блок распознавания данных с весов**

- Распознать и преобразовать в ASCII код считанные данные

***LCD\_Data\_Recognize(LCD\_bufer, ASCII\_data)***

LCD\_bufer - адрес ячейки начиная с которой располагается буфер считанных с весов данных

ASCII\_data – распознанные данные в виде ASCII кода (поочередно – сначала 1я строка 5 символов, потом 2я строка 5 символов, потом 3я строка 6 символов, 2 байта с позициями запятых)

**2. Блок программы работы с дисплеем (вывод на дисплей):**

- Продублировать полученные с весов данные (вывести данные буфера на дисплей)

***LCD\_Data\_Duplicate(lcd\_data\_addr)***

lcd\_data\_addr - адрес ячейки начиная с которой располагается буфер считанных с весов данных

- Преобразовать ASCII буфер (16 символов+2 байта с позициями запятых) в код для вывод на дисплей

***Convert\_ASCII\_to\_LCD(lcd\_data\_addr, ASCII\_data)***

lcd\_data\_addr - адрес ячейки начиная с которой располагается буфер для вывода на дисплей

ASCII\_data – данные в виде ASCII кода (поочередно – сначала 1я строка 5 символов, потом 2я строка 5 символов, потом 3я строка 6 символов)

- Преобразовать символ в коде ASCII в код дисплея

***ASCII\_to\_LCDcode(ASCIIcode):LCDcode***

ASCIIcode - символ в коде ASCII

LCDcode - символ в коде дисплея

Отправка данных буфера в дисплей

***Data\_to\_LCD(lcd\_data\_addr)***

lcd\_data\_addr - адрес ячейки начиная с которой располагается буфер считанных с весов данных

- Вывести символ в указанной позиции

***PrintSimb(SimbCode, Simb\_Pos)***

SimbCode - ASCII код выводимого символа

Simb\_Pos – позиция в которой выводить символ (от 0 до 15: 0-4 первая строка, 5-9 вторая строка, 10-15 третья строка)

- Вывести сообщение в первой строке

***PrintFirstStr(MsgAddr)***

MsgAddr – адрес начала области памяти где хранятся 5 символов в ASCII коде для вывода в первую строку

- Вывести сообщение во второй строке

***PrintSecondStr(MsgAddr)***

MsgAddr – адрес начала области памяти где хранятся 5 символов в ASCII коде для вывода во вторую строку

- Вывести сообщение в третьей строке

***PrintThirdStr(MsgAddr)***

MsgAddr – адрес начала области памяти где хранятся 6 символов в ASCII коде для вывода в третью строку

- Вывести сообщение во все строки (16 символов)

***PrintMsg(MsgAddr)***

MsgAddr – адрес начала области памяти где хранятся 16 символов в ASCII коде для вывода на дисплей

- Установить запятую

***SetKoma(komapos)***

Komapos – разряд перед которым надо поставить запятую

- Убрать запятую

***ClrKoma(komapos)***

Komapos – разряд перед которым надо убрать запятую

**3. Блок программы работы с модулем часов реального времени – часы и микросхема памяти на модуле**

- Считать текущее состояние времени/даты/для недели и т.д.

***ReadRTCData(HH,MM,SS,WD,DD,MO,YY)***

- Считать отдельно значение времени

***ReadTime(HH,MM,SS)***

- Считать отдельно дату

***ReadDate(DD,MO,YY)***

- Считать отдельно день недели

***ReadDayofWeek(DW)***

- Установить все временные параметры

***WriteRTC(HH,MM,SS,DW,DD,MO,YY)***

- Установить время (не меняя при этом все остальное)

***WriteTime(HH,MM,SS)***

- Установить дату (не меняя при этом все остальное)

***WriteDate(DD,MO,YY)***

- Установить день недели (не меняя при этом все остальное)

***WriteDayofWeek(DW)***

- Записать байт по указанному адресу

***WriteADDR(ADDR)***

- Записать последовательность байт (до 64) начиная с адреса

***WriteBlock(ADDR,LEN)***

- Считать байт по указанному адресу

***ReadADDR(ADDR)***

- Считать последовательность байт (до 64) начиная с адреса

***ReadBlock(ADDR,LEN)***

**4. Блок распознавания (анализа) команд (данных во второй строке) вводимых пользователем**

**5. Блок работы с UART**

- Включение UART

***UART\_ON***

- Выключение UART

***UART\_OFF***

- Установка скорости UART

***SetUARTSpeed(UARTSpeed)***

- Отправить байт по UART

***SendBytetoUART(UARTbyte)***

- Отправить пакет (до 64) байт (сохраненных в памяти начиная с указанного адреса) по UART

***SendPackettoUART(ADDR,LEN)***

- Поместить считанный из UART байт в регистр

***GetUARTByte(ADDR)***

**6 .Блок работы с SIM800 – основные функции**

- запрос баланса (получаем не значение баланса а ответ оператора модулю)

***GetBalanceStr(BufAddr,Length)***

BufAddr – адрес буфера где будет храниться ответ оператора

Length – длина считанной строки

- определить баланс (находит в строке значение баланса)

***GetBalance(BufAddr, Balance)***

- запрос данных об операторе (получаем не значение баланса а ответ оператора модулю)

***GetOperatorStr(BufAddr,Length)***

- запрос уровня покрытия сети (получаем не значение баланса а ответ оператора модулю)

***GetNetworkStr(BufAddr,Length)***

- запрос координат (получаем не значение баланса а ответ оператора модулю)

***GetCoordStr(BufAddr,Length)***

- определение широты из строки ответа оператора модулю

***GetLatitude(BufAddr,Latitude)***

- определение долготы из строки ответа оператора модулю

***GetLongitude(BufAddr,Longitude)***

**7. Блок работы с SIM800 – отправка и прием SMS**

**8. Блок работы с SIM800 – передача и прием данных по GPRS**

- установить адрес сервера для связи

***SetServerAddr(ServerAddr)***

- отправить GET запрос по HTTP (результат успешно или нет, код ответа и количество байт ответа)

***HTTPGET(Query,AnswCode,AnswCnt)***

- отправить GET запрос по HTTPS (-.-)

***HTTPSGET(Query,AnswCode,AnswCnt)***

- отправить POST запрос по HTTP (-.-)

***HTTPPOST(Query,AnswCode,AnswCnt)***

- отправить POST запрос по HTTPS (-.-)

***HTTPSPOST(Query,AnswCode,AnswCnt)***

- прочитать ответ сервера на запрос (!ответ обычно храниться в модуле – можно считать не сразу)

***ReadServerAnswer(Buf,AnswCnt)***

1. ПРОВЕРИТЬ ВОЗМОЖНОСТЬ РАБОТЫ С ВНЕШНИМИ ФУНКЦИЯМИ