В файле DOC/Лог 21\_03\_18.txt информация по импульсам. Эти данные пока можно использовать как образец сигналов.

!!!! ТЗ на Март 2018г.

0) ~~Переделываем стартовое меню "НАСТРОЙКИ" для увеличения количества пунктов и подпунктов.~~

1) Формируем переменные для хранения количества импульсов прерывания. Переменная должна храниться в энергонезависимой памяти. Память какая? Уже стоит на плате с контроллером?

По каждому каналу своя переменная. Предусмотреть ввод данных как с меню, так и дистанционно по RS485.

Переменные нужны для контроля количества импульсов. При меньшем или большем количестве поступивших прерываний от заданного – ошибка.

2) Формируем переменные для хранения дельт количества импульсов. К примеру +\- 2 импульса. Ведь не всегда планка может точно встать в исходное или еще чего хз.

По каждому каналу своя дельта. Предусмотреть ввод данных дельт как с меню, так и дистанционно по RS485.

3) Создаем подпрограмму формирования образцовых графиков по каждому каналу. Храним в файле.

а) Формирование графиков по однократному срабатыванию системы.

б) Формирование возможно по фактическому принудительному срабатыванию системы.

в) В случае если систему нельзя трогать, формируем по первому срабатыванию системы.

Для этого выставляем флаг "сформировать образец по первому срабатыванию". После фиксации срабатывания флаг снимается.

Возможно нужен флаг "эталон сформирован" иначе сложно сравнивать не имея эталона.

г) Создание эталона вручную, создание текстового файла с последующей загрузкой на SD.

д) Формируем дельты отклонений по каждому каналу. Точное совпадение сигналов мало вероятно.

4) Регистрируем дату и время срабатывания системы. Пишем в файл. Формат хранения, папка???

5) Регистрируем количество срабатывания защиты. Пишем в файл. Формат хранения, папка???

6) Формируем переменные для хранения количества срабатываний моторесурса. Предусмотреть ввод данных количества срабатываний моторесурса как с меню, так и дистанционно по RS485.

7) Подсчитываем моторесурс системы. Принцип формирования моторесурса. К примеру, закладываем 1000 срабатываний. Регистрируем количество срабатываний и высчитываем процент

использованного ресурса. Возможно необходимо отображать остаток ресурса.

8) Определяем, что было отключение или подключение штанги. Пишем в файл. Формат хранения, папка???

9) Записываем данные на SD в удобоваримом виде. Уточнить формат данных. Формат хранения, папка???

Возможно в нескольких разных файлах типа данные для построения графика, таблицу количества и даты срабатывания системы.

10) Сравниваем полученный график с образцовым (формируем отдельно с возможностью редактирования, задания параметров).

11) При срабатывании системы зажигаем желтый светодиод. Параллельно зажигаем зеленый светодиод если параметры в норме или красный если параметры вышли за пределы дельт отклонений.

12) Контролируем вход релейной защиты (PWM11, PWM12), Пока фиксируем срабатывание и выставляем сигнал на контакт №4 (описание контактов ниже). Можно применить прерывания (если получиться обработать).

12) ~~Сброс сигнала срабатывания системы желтой кнопкой~~ или дистанционно по RS485.

13) Формируем сигнал срабатывания системы на выводах АСУ ТП

Выходные сигналы АСУ ТП (PIN42, PIN43, PIN44, PIN50)

№1 - НО контакта: в схему УРОВ (срабатывание УРОВ) (параллельно желтому светодиоду)

№2 - 1 НЗ контакт: в схему ЦС (Live-контакт) хз что это. Просто зарезервируем.

№3 - НО контакт: «неисправность выключателя» (параллельно красному светодиоду. При выходе параметров кривой движения за допустимые границы)

№4 - НО контакт: фиксируем срабатывание по входу релейной защиты.

Возможно создаем механизм правил, а там посмотрим, что на что будет реагировать. Точной информации от заказчика нет.

14) Продумать регистрацию возврата системы в исходное состояние. Создание флага "состояние исходное".

15) Подсчет общего времени движения планки, поканально

16) Индикация версии ПО на дисплее, а также передача информации о версии по каналу RS485.

17) Создание протокола обмена информацией по каналу RS485. В последующих версиях будет подключение и ESP WiFi.

19) Контролируем источники питания системы. Показатели на первом экране. При норме цифры зеленого цвета. Вне диапазона - красного.

а)«200вольт» - AD9,

б)«3V3»- AD10,

в)«5V0»-AD11

Запись в лог диагностики.

20) Контроль температуры в корпусе. Запись в лог диагностики. Формат хранения, папка???

21) Контролируем исправность подключения индуктивных датчиков. Входа AD6, AD7, AD8. При исправных присутствует некий потенциал (уточним). Обрыв - потенциал ноль. Сравниваем с некой величиной по каждому каналу.

Запись в лог диагностики. Формат хранения, папка???

22) ~~Создаем (или уже создан) механизм контроля нажатия кнопок. Как дальше применять уточним.~~

23) Запускаем механизм Watch-Dog микроконтроллера. Необходимо уточнить подпрограмму и настройки. Применять на завершающем этапе.

24) Создание меню просмотра файлов эталонов, логов, событий.

25) Ручная и дистанционная очистка архива (кривые перемещения, долговременная диагностика – после замены или кап ремонта выключателя).

!!!!!!!!!!!!!!!!!!!!!! Дополнения, уточнения:

На экране вывода графика прерывания необходимо добавить:

~~Вверху дату и время, когда произошло событие.~~

Внизу количество импульсов по каналам. Если количество в пределах нормы - зеленый цвет. Вне нормы – красный.

В конце строки – процент наработки.

Справа добавить кнопки "Фиксировать изображение", "Продолжить (или Завершить)".

~~При просмотре более 5 минут (величина настраиваемая) автоматически выходить из режима просмотра.~~

Добавить три цветные кнопки "Сохранить как эталон" (справа от графика). Названия кнопок могут быть короче или в виде номеров

Подписаны общим текстом. Текст может быть расположен и внизу графика.

На первом экране вверху справа колонкой сверху вниз вывести показания источников питания 3.3в, 5.0в. 200в. analogRead не работает, наверное, виной всему измерение ADC, которое реализовано.

~~В пределах нормы - зеленым цветом, вне нормы красным. Я думаю 10 процентов отклонения достаточно.~~

На первом экране справа от графика (почти вплотную) сверху вниз показания исправности индуктивных датчиков.

Индикация в виде квадратов или прямоугольников с цифрой (номером) датчика. Зеленый цвет – норма, красный - вне нормы.

Пороговые значения записать в переменные, которые можно изменять из меню или по RS485.

Необходимо создать цифровую клавиатуру для ввода параметров.

Клавиши 0-9, "Очистить", "Ввод", "Выход".

Необходимо создать шестнадцатеричную клавиатуру для ввода параметров в шестнадцатеричном виде.

Клавиши 0-9, А,В,С,В,Е,F, "Очистить", "Ввод", "Выход".

Расположение (координаты) клавиш клавиатуры могут быть такими как в коде. Код в файле "Пример клавиатуры.txt"

Клавиши лучше организовать твоими средствами. В файле только пример расположения на экране.

Клавиатура применяется в пунктах меню "Импульсы", "Дельта", "Наработка", "Порог мах.", "Порог мин.","RS485" и.т.д

При вводе обязательно показывать текущее значение. Варианты команд "Ввод", "Отмена", "Выход".

Измерение напряжения можно выполнить подпрограммой (с учетом коэффициентов делителей):

Как пример, можно и иначе при необходимости.

void measure\_power()

{ // Программа измерения напряжения питания с делителем 1/2

int m\_power = 0;

float ind\_power = 0;

m\_power = analogRead(power\_3V3); // Контроль источника питания +3.3в

power3V3 = m\_power \*(2.4 / 1024\*2); // Получить напряжение в вольтах

m\_power = analogRead(power\_5V0); // Контроль источника питания +5.0в

//Serial.println(m\_power);

power5V0 = m\_power \*(2.4 / 1024 \* 2); // Получить напряжение в вольтах

m\_power = analogRead(power\_200); // Контроль источника питания 200в

power200 = m\_power \*(2.4 / 1024 \* 100); // Получить напряжение в вольтах

}

На экранах просмотра логов, графиков, эталонов графиков справа от списков установить кнопки "Просмотр", "Отмена"

При нажатии на имя файла, выделить имя красной рамкой. При переборе файлов перемещать выделение.

Выделенный файл открывать кнопкой "Просмотр". Вызывается новый экран с отображением содержания файла (в сокращенном виде возможно) или график.

~~Даты создания файлов записываются в атрибуты файла (как новая функция работы с файлами в контроллере теплицы).~~

Информация по графикам выводится поканально - записанное значение и эталон для этого канала (для сравнения).

Предусмотреть вызов по номерам или другой механизм перебора графиков

Цветовая схема графиков немного изменяется. Красный цвет назначается эталону.

~~Красный цвет графика прерывания изменить на другой (возможно белый).~~

~~Время просмотра немного увеличиваем для комфортного принятия решения - "Фиксировать изображение" или нет.~~

Программа Watch-Dog микроконтроллера может быть организована:

Эта функция не срочная, оставить в последнюю очередь.

void watchdogSetup(void)

{

watchdogEnable(16000);

}

void watchdog\_Reset()

{

watchdogReset();

}

void setup()

{

watchdogEnable(100000); // Включить сторожевой таймер

}

Уточнить сброс сторожевого таймера. Возможно в меню предусмотреть отключение.

Пример: https://gist.github.com/facchinm/dff2c596f1fa23e1702f

http://forum.arduino.cc/index.php?topic=132986.0

!!!! Измерение источников питания и исправности индуктивных датчиков

Попробовать одновременное измерение 9 входов.

Если не получиться по быстродействию, выполнить второй вариант.

Периодичность 3-5 минут.

Входа:

ch1:A6 Вход индуктивного датчика №1 тест исправности датчика

ch0:A7 Вход индуктивного датчика №2 тест исправности датчика

ch10:A8 Вход индуктивного датчика №3 тест исправности датчика

ch11:A9 Измерение =200В

ch12:A10 Измерение 3V3

ch13:A11 Измерение +5V

1. Назначить новые данные в переменную для хранения входов АЦП типа:

#define ADC\_CHANNELS ADC\_CHER\_CH1 | ADC\_CHER\_CH0 | ADC\_CHER\_CH10 | ADC\_CHER\_CH11 | ADC\_CHER\_CH12 | ADC\_CHER\_CH13

1. Назначить количество каналов:

#define NUM\_CHANNELS 6 // Установить количество аналоговых входов

1. Определяем буфер данных:

#define BUFFER\_SIZE 50\*NUM\_CHANNELS // Определить размер буфера хранения измеряемого сигнала

#define NUMBER\_OF\_BUFFERS 6 // Установить количество буферов

Данные для установок АЦП подготовлены. Останавливаем прерывания АЦП.

NVIC\_DisableIRQ(ADC\_IRQn);

1. Повторяем все действия
2. void ADCSampler::begin(unsigned int samplingRate) // Заносим новые парвметры в АЦП
3. void ADCSampler::handleInterrupt() // Получаем 50 измерений по каждому каналу.
4. Расчет средней величины из 50 измерений (для усредненного результата).
5. Возвращаемся в измерение сигналов токовых трансформаторов(с установкой прежних параметров).