Функции

1. Включить в проект Файлы:

FindX.h FindX.c

stat.h stat.c

StatPM.h StatPM.c

1. Основной файл взаимодействия с основным ПО

StatPM.h

#define St\_Alarm 0x01 // 1 - Превышение порога поискового алгоритма

// 0 - Норма

#define St\_Call 0x02 // 1 - Режим Калибровка

// 0 - Режим Поиск

// Status\_Stat

typedef struct {

uint8\_t Status; // Flag Status

uint8\_t nSigm; // Количество превышение над порогом ср.-их отклонений для управления сигнализацией

float CPS; // Текущая скорость счета

uint8\_t Delta; // Статистическая погрешность [2..99] %

} TStatus\_Stat;

// Function

void Stat\_Init(float nSigm, uint16\_t tCall, float tau); // Инициализация

void Stat\_GetStatus(TStatus\_Stat \*Status); // Опрос статуса

void Stat\_AddData250ms(int32\_t Ni); // Заполнение

void Stat\_Reset(void); // Cброс статистики

void Stat\_Start\_Call(void); // Запуск на калибровку

1. Пример:

В основном проекте:

// Status

#include "StatPM.h"

TStatus\_Stat Status; // Выходные данные

**1. Инициализируем алгоритм**

// sigm=3.4 время калибровки 60 сек; мёртвое время 1мкс (может быть 0);

Stat\_Init(3.4,60,1E-6);

**2. Передаём за каждые 250 ms Ni- количество импульсов со счётчика в алгоритм**

// uint32\_t Ni – количество импульсов

Stat\_AddData250ms(Ni);

1. **Опрашиваем статус**

Stat\_GetStatus(&Status);

1. **Выводим данные**

if ((Status.Status&St\_Alarm)==St\_Alarm) {

// Включаем сигнализацию

} else {

// Выключаем сигнализацию

}

//

Status.CPS // Текущая скорость счета имп/cек

Status. Delta // Погрешность в %