

## Общее описание файла конфигурации

Файл конфигурации представляет из себя документ в формате xml содержащий данные конфигурации, описанные в этом документе. Как и любой другой xml-файл, он начинается со строки

```
<?xml version="1.0" encoding="utf-8"?>
```

Далее данные конфигурации заключены в тэги `<Config>``</Config>`. Данные вне этих тэгов игнорируются.

Названия тэгов и параметров, а также их значений, чувствительны к регистру. Все цифровые значения являются десятичными целыми, если не указано иное.

## Порядок загрузки файла конфигурации

Загрузка конфигурации происходит в 2 этапа.

1. При запуске программы происходит загрузка общей конфигурации программы
2. При подключении к последовательному порту и выбору устройства происходит загрузка конфигурации заданного устройства

### Общая конфигурация программы

Общая конфигурация содержит данные о возможных скоростях обмена данных (бод-рейтов) последовательного порта и список устройств с общим идентификатором.

Список доступных для выбора пользователем бод-рейтов записывается набором тэгов `<BaudRate value="9600"/>`. Эти данные будут доступны для выбора пользователю программы в меню *Port -> Select baudrate*. Каждый элемент меню добавляется новым тэгом *BaudRate*.

Устройства с общим идентификатором – это набор устройств, которые по запросу команды 0x0702 возвращают код 0x0115. При попытке подключения к такому устройству пользователю будет предложено выбрать вручную тип устройства из списка предложенных. Список устройств формируется динамически из данных, представленных внутри тэга `<CommonIDDevices>`. Каждый элемент списка описывается тэгом `<CIDD id="14">SF6015 v2</CIDD>`, где *id* – уникальный айди устройства (используется для связи с конфигурацией устройства и должен быть УНИКАЛЬНЫМ), а “SF6015 v2” – выводимое название устройства.

### Пример общей конфигурации

```
<BaudRate value="9600"/>
<BaudRate value="57600"/>
<BaudRate value="115200"/>
<CommonIDDevices>
  <CIDD id="14">SF6015 v2</CIDD>
  <CIDD id="15">SF6030 v2</CIDD>
  <CIDD id="16">SF6040</CIDD>
  <CIDD id="17">SF6090</CIDD>
  <CIDD id="13">SF6100 v2</CIDD>
  <CIDD id="18">SF6250</CIDD>
  <CIDD id="11">SF6015 v1</CIDD>
  <CIDD id="12">SF6030 v1</CIDD>
  <CIDD id="10">SF6100 v1</CIDD>
</CommonIDDevices>
```

## Конфигурация устройства

Конфигурация устройства описывается внутри тэга

```
<Device id="10" name="SF6100 v1" stopCommandDelayMs="300" minStopCommandDelayMs="0" maxStopCommandDelayMs="1000"> ... </Device>
```

Где

- *id* – уникальный идентификатор устройства (**шестнадцатеричное число**);
- *name* – название устройства;
- *stopCommandDelayMs* (не обязательный параметр) – таймаут запроса следующей команды после команды остановки генерации. Значение по умолчанию (используется если параметр не указан) = 150 мс.
- *minCommandDelayMs* (не обязательный параметр) – минимальное значение величины таймаута между командами. Значение по умолчанию 50 мс.
- *maxCommandDelayMs* (не обязательный параметр) - минимальное значение величины таймаута между командами. Значение по умолчанию 1000 мс.

Параметры *minCommandDelayMs* и *maxCommandDelayMs* задают пределы пользовательской регулировки величины таймаута между командами.

### Информация об устройстве

Информация об устройстве выводится в окне, всплывающем при нажатии кнопки *Device options*, описывается внутри тэга `<Content> ... </Content>` следующими тэгами: Image, Description, Link.

`<Image>SF6100-1.jpg</Image>` - название файла картинки устройства. Загружается и показывается пользователю из папки *devices* в корневой папке программы. Может быть чувствительно к регистру файла в некоторых ОС.

`<Description>...</Description>` - Описание устройства - текст.

`<Link>https://www.maimanelectronics.com/archive</Link>` - Ссылка на страницу устройства на сайте.

### Список запрашиваемых значений

Далее следует указывать список запрашиваемых значений. Он должен идти сразу за описанием устройства или перед ним, т.к. другие данные (описываемые в последующих разделах этой инструкции) использую ссылки на этот список. Если список не будет загружен к моменту загрузки остальных данных конфигурации, могут возникнуть ошибки конфигурации. Список запрашиваемых значений (команд) описывается внутри тэгов `<Commands></Commands>`. Каждое значение задается тэгом

```
<Command code="0101" divider="10" interval="100" isSigned="" />
```

Где

- *code* – **шестнадцатеричное** значение параметра, запрашиваемого у устройства;
- *divider* – (не обязательный параметр, значение с плавающей точкой), величина делителя значения параметра. По умолчанию равен 1.
- *interval* - (не обязательный параметр) – период запроса команды, может принимать значения от 1 до 100 (включительно). По умолчанию равен 1. Если параметр равен 1, то команда будет запрашиваться каждый цикл опроса, если равен 10, то каждый десятый цикл опроса.
- *isSigned* – (не обязательный параметр) – признак знака, значение параметра не важно, но наличие кавычек и символа равно обязательно! При наличии данного атрибута, значение

параметра будет иметь знаковый тип. По умолчанию все параметры являются беззнаковыми.

### Установка пределов

Пределы устанавливаются внутри тэгов `<Limits> ... </Limits>`, каждый предел описывается тэгом `<Limit...>` и выводится в меню *limits* главного окна программы. Текст внутри тэга используется как название устанавливаемой величины.

```
<Limit upperCode="200" maxCode="ffff" minCode="15" bottomCode="10" unit="V"
show="both">Voltage</Limit>
```

- *minCode* - (шестнадцатеричное), код параметра для установки нижнего предела
- *maxCode* - (шестнадцатеричное), код параметра для установки верхнего предела
- *bottomCode* - (шестнадцатеричное), код параметра установки абсолютной нижней границы параметра
- *upperCode* - (шестнадцатеричное), код параметра установки абсолютной верхней границы параметра
- *unit* - (строка). Обозначение единицы измерения величины
- *show* - тип регулируемых пределов:
  - *min* - Показывает и позволяет редактировать только нижний предел - обязательные поля в этом случае *bottomCode*, *minCode*, *maxCode*.
  - *max* - Показывает и позволяет редактировать только верхний предел - обязательные поля в этом случае *minCode*, *maxCode*, *upperCode*.
  - *both* - Показывает и позволяет редактировать нижний и верхний пределы - все поля типа *\*Code* являются обязательными

Все атрибуты тэга *Limit*, кроме *show*, имеют по умолчанию - пустое значение.

### Калибровка параметров

Некоторые параметры могут быть откалиброваны. Список доступных параметров находится внутри тэгов `<CalibrationKoeFs> ... </CalibrationKoeFs>` и описываются тэгом *Calibrate* с текстом - названием калибровки, которая будет показана пользователю в меню Calibrate

```
<Calibrate code="030E" min="9500" max="10500" >Current calibration</Calibrate>
```

- *code* - (шестнадцатеричное) код параметра для считывания\установки нового значения калибровки
- *min* - минимально возможное значение калибровки. По умолчанию 95.00%
- *max* - максимально возможное значение калибровки. По умолчанию 105.00%

### Отображение регулируемых параметров

Регулируемые параметры описаны внутри конструкции `<ParamControls> ... </ParamControls>` с помощью следующего тэга

```
<Param unit="A" min="0301" max="0302" value="0300" real="0307" >Current</Param>
```

- *unit* - (строка) единица измерения величины. Строка "(deg)" будет автоматически заменена на символ градуса "°". Если параметр есть температура, то дальнейшее указание единиц не требуется.
- *min* - (шестнадцатеричное) код параметра минимального значения величины
- *max* - (шестнадцатеричное) код параметра максимального значения величины
- *value* - (шестнадцатеричное) код параметра устанавливаемого значения величины
- *real* - (шестнадцатеричное) код параметра измеренного значения величины

- `isTemperature` - (не обязательный флаг). Принимает значения 0 или 1. 0 - равносильно отсутствию флага. Флаг используется для конвертации значений между системами Цельсия (C) и Фаренгейта (F). Соответствующее обозначения автоматически добавляется к единицам измерения. См. следующий пример

`<Param unit="(deg)" isTemperature="1" real="0AF4" divider="10">Driver Temperature</Param>`

Текст внутри тэга используется как название параметра. Если параметр не имеет атрибута `value` - он будет отображен в окошке справа (окно измеряемых параметров). Если параметр имеет атрибут `value`, не зависимо от наличия атрибута `real`, он будет отображен в качестве регулируемого параметра.

### Чек-боксы

Команды с возможностью бинарного выбора описаны внутри тэга `<BinaryOptions> ...`

`</BinaryOptions>`. Каждая команда будет отображена в виде чек-бокса в окне программы.

Описание производится следующим тэгом

`<CheckBox code="0700" onCommand="0020" offCommand="0040" mask="0004">Internal Current Set</CheckBox>`

- `code` - (шестнадцатеричное) - код параметра
- `onCommand` - (шестнадцатеричное) - значение параметра, отправляемое при выставлении чек-бокса
- `offCommand` - (шестнадцатеричное) - значение параметра, отправляемое при стирании чек-бокса
- `mask` - (шестнадцатеричное) - маска состояния. Маска накладывается (побитовое И) на считанное значение и если результат нулевой, то чек-бoks очищается, если отличен от нуля чек-бoks устанавливается.

Текст внутри тэга является подписью к чек-боксу.

### Кнопки

Кнопки описаны внутри конструкции `<Button> ... </Buttons>`. Каждая кнопка описывается следующим тэгом

`<Button name="laser" mask="02" code="0700" onCommand="0008" offCommand="0010"/>`

- `name` - имя-идентификатор кнопки. На данный момент программа умеет обрабатывать лишь 2 кнопки, а остальные игнорирует. Поддерживаются кнопки: `laser`, `tes`.
- `mask` - (шестнадцатеричное) - маска состояния. Действие аналогично маски чек-боксов
- `code` - (шестнадцатеричное) - код параметра
- `onCommand` - (шестнадцатеричное) - значение параметра, отправляемое при переводе состояния кнопки в активное состояние
- `offCommand` - (шестнадцатеричное) - значение параметра, отправляемое при переводе состояния кнопки в неактивное состояние

### “Светодиоды”

Светодиоды - бинарный элемент, имеющий два состояния, отображающие состояние маски(ок).

Расположены внутри тэга `<Leds> ... </Leds>`. Светодиоды описываются тэгом `<Led ...>` с

вложенным(и) тэгом(ами) `LedMask` с описанием масок.

`<Led label="IntLock">`

`<LedMask code="0800" mask="0002" maskColor="#ffff00">Interlock</LedMask>`

`<LedMask code="0800" mask="0020" maskColor="#ff0000">LD Overheat</LedMask>`

</Led>

- label – (строка) название светодиода
- code – (шестнадцатеричное) – код параметра для считывания
- mask – (шестнадцатеричное) – маска состояния. Действие аналогично маски чек-боксов
- maskColor – (#RRGGBB) – цвет активной маски. По умолчанию (#00ff00 - зелёный)

Светодиод может содержать как одну, так и несколько масок. В случае совпадения нескольких масок, будет выведено сообщение, состоящее из перечисления строк всех масок.