Системные функции. Вводная часть.

Ниже приведено описание всех системных функций. С точки зрения Lua, это внешние глобальные функции. Поэтому синтаксические ошибки в именах функций не проверяются. В случае если имени функции написано неверно, при выполнении LUA приложения возникнет ошибка исполнения. Ошибка и имя функции, на которую не была найдена ссылка, отобразиться в телеметрических данных системы.

При использовании системных функций, стоит следить за типом данных передаваемых как параметры. Система контролирует только выход параметров за допустимые диапазоны в рамках типа данных. Однако для повышения производительности системы, проверка типа данных не производится. Поэтому, в случае если в качестве параметра необходимо передать целое беззнаковое, а передастся число с плавающей точкой или отрицательное число, произойдёт автоматическое преобразование типа данных по правилам ядра LUA. Результат этого преобразования может быть не предсказуем. **Речь идет прежде всего о параметрах системных функций, когда данные передаются за пределы Lua программы. Внутри самой Lua программы преобразования работают корректно.**

Все системные функции, если это не указано отдельно, являются функциями немедленного действия. Они приостанавливают выполнения прикладной программы на время их выполнения ядром системы.

Функции общего назначения.

**SYSTEM\_RESTASRT()**

Функция перезапуска системы. Перезапуск будет выполнен после завершения текущего цикла прикладной программы. Т.е. сначала будет выполнен весь код от SYSTEM\_RESTASRT() до Yield. При этом буду применены новые значения выходных сигналов и только после этого произойдет перезапуск. При перезапуске все выходные каналы будут выключены, после этого произойдет полная реинициализация системы, включая выгрузку прикладной программы из памяти.

**SetEEPROM( пра1, пар2)**

Функция для записи данных в EEPROM. Система поддерживает адресацию до 409 4-х байтных объектов. Сохранять можно следующие Lua типы данных: float, integer, boolean. При записи переменной в память сохраняется ее тип. При чтении из памяти, возвращается переменная того же типа, который был при записи.

Скорость записи данных значительно ниже общего быстродействие системы, поэтому для того, что не снижать быстродействие алгоритма, не рекомендуется производить запись без необходимости. EEPROM в первую очередь предназначен для хранения уставок и/или записи ошибок, возникающие в процессе эксплуатации.

### ***пар1. -обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : адрес в памяти

### *Допустимые значения:* от 0 до 408.

### *Тип данных*: беззнаковое целое

### ***пар2 – обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : данные для записи

### *Допустимые значения:* любая переменная или константа типа float, integer, boolean.

### *Тип данных*: float, integer, boolean

Возвр1 GetEEPROM( пар1)

Функция чтения данных из EEPROM с кэшированием. При вызове функция проверяет системный кэш, и если в нем данные отличны от нуля, то возвращает их обратно, без непосредственного чтения EEPROM. После старта системы, весь кэш нулевой. При любой записи и чтении данные (если это не 0), дублируются по соответствующему адресу в кэш.

### ***пар1. -обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : адрес в памяти

### *Допустимые значения:* от 0 до 408.

### *Тип данных*: беззнаковое целое

### ***Возвр1– возвращаемое значение.***

### *Значение параметра* : данные по соответствующему адресу.

### *Тип данных*: float, integer, Boolean, в зависимости от типа данных, записанных в по адресу.

**Функции работы с выходными каналами.**

### **setOutConfig( пар1, пар2, <пар3>,<пар4>,<пар5>)**

### Функция конфигурация канала. Если не вызвать setOutConfig, то канал будет выключен на уровне ядра и не будет реагировать ни на какие команды

### ***пар1. -обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : номер канала

### *Допустимые значения:* от 1 до 20.

### *Тип данных*: беззнаковое целое

### ***пар2 – обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : номинальный ток.

### *Допустимые значения:* Для каналов 1-8 до 30.0А, для каналов 9-20 до 15.0А.

### *Тип данных*: положительное c плавающей точкой. Кол-во знаков после запятой ограничено представление вещественных числе в LUA. Точность срабатывания уставки зависит от стабильности питания и потребителя. В общем случае рекомендуется исходить из возможных колебаний тока потребления нагрузки в пределах +-0,5А.

### ***пар3. – необязательный, значение по умолчанию - 1***

### *Значение параметра* : сброс перегрузки выключение канала. При 0, ошибка на канале сбрасывается только рестартом системе. При 1, выключение канала из прикладной программы.

### *Допустимые значения:* 0 или 1.

### *Тип данных*: беззнаковое целое

### .

### ***Пар4. – необязательный, значение по умолчанию - 0***

### *Значение параметра* : время работы в перегрузке в мс

### *Допустимые значения:* от 0 до 32 000

### *Тип данных*: беззнаковое целое

### ***Пар5. – необязательный, значение по умолчанию – номинальный ток***

### *Значение параметра* : максимальный ток перегрузки

### *Допустимые значения:* Для каналов 1-8 до 120.0А, для каналов 9-20 до 40.0А.

### *Тип данных*: положительное c плавающей точкой. Кол-во знаков после запятой ограничено представлением чисел c плавающей точкой в LUA. Точность срабатывания уставки зависит от стабильности питания и потребителя. В общем случае рекомендуется исходить из возможных колебаний тока потребления нагрузки в пределах +-0,5А.

**OutResetConfig(пар1, пар2, пар3)**

Конфигурация поведения канала при перегрузке. Если функция не вызвана и канал сконфигурирован, то при перегрузке канал уходит в ошибку.

### ***пар1. -обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : номер канала

### *Допустимые значения:* от 1 до 20.

### *Тип данных*: беззнаковое целое

### ***пар2 – обязательный, при отсутствии функция игнорируется***

### *Значение параметра* : кол-во попыток запуска

### *Допустимые значения:* 0 – бесконечное кол-во пере запусков, 1, выключается при перегрузке, 2- две попытки включения, 3- три попытки включения. Максимальное кол-во 255

Имеется ввиду, что 2 попытки включения, это включение и один перезапуск в случае ошибки.

### *Тип данных*: беззнаковое целое

### ***пар3. – необязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : таймаут перед очередным перезапуском в мс.

### *Допустимые значения:* от 0 до 65 535

### *Тип данных*: беззнаковое целое

### .

**setPWMGroupeFreq(пар1, пар2)**

Установка частоты ШИП для группы каналов. Менять частоту возможно только для группы каналов, после вызова функции для всех каналов группы будет доступен ШИМ на указанной частоте.

### ***пар1. -обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : номер группы каналов

### *Допустимые значения:*

0– каналы 5,6,9,10

1 – каналы 11,12,16

2 – каналы 4,15

3 -каналы 1,2,8,20

4 – каналы 13,14,17,18

5 -каналы 7 и 19

### *Тип данных*: беззнаковое целое

### ***пар2 – обязательный, при отсутствии функция игнорируется***

### *Значение параметра* : частота ШИМ в ГЦ

### *Допустимые значения:* от 1 до 1999 Гц

### *Тип данных*: беззнаковое целое

**OutSetPWM( пар1, пар2)**

Установка скважность ШИМ сигнала для выбранного канала. Для того, что бы функция работала, необходимо что бы бал установлена частота ШИМ для соответствующей группы каналов и канал бы включен. Если канал выключен, то ШИМ на новой скважности запустился только после включения. При не установленной частое группы каналов, функция игнорируется.

### ***пар1. -обязательный, при отсутствии функция игнорируется.***

### *Значение параметра* : номер канала

### *Допустимые значения:* от 1 до 20.

### *Тип данных*: беззнаковое целое

### ***пар2 – обязательный, при отсутствии функция игнорируется***

### *Значение параметра* : скважность ШИМ сигнала в %

### *Допустимые значения:* 0 – 100. Во избежание ложных срабатывай защиты, рекомендуется менять параметр при включенном канале до 99%. Токовая защита при ШИМ сигнале работает по своему алгоритму. Переход на включенном канале с 50% на 100% фактический меняет алгоритм обработки на лету, что при определённых типах нагрузок может привести к ложному срабатыванию токовой защиты.

### *Тип данных*: беззнаковое целое

**Функции CAN**

Все функции CAN, как и соатльные все системные ыункции выполнюмят при вызвое. Однако они передают/получает данные не непрсордествно из CAN шин. Работаетм многоступенчатя систма, позводяющая максимально запрарлелить обмен по CAN шине и работу прикладного алгоритма.

При оптрвке данных из Lua, данные записываются в системную очередь. Дальше системный драйвер по мере физической отпрвки пакетов раскидывет очередь в аппратные FIFO буфера микропроцессора, который пытается выпихнуть их на шину. Такая многоступенчатя архитектура позволяте максиально снихить нагрузку на процессор и увелисть скорсть выполенения прикладной программы. Таким образом если в одном цикле програмы отпрвить пачку фреймов, физический последний из пактов могут уйти на лении уже на следующем цикле программы.

Прием пакетов организован по принципку почтовых ящиков. Почтовый ящик конфигурируется по ID. ID попадет в сисетму аппартной фильтрацииСисетма не принимает другие ID на уровне котроллера CAN в микропроцессоре. Если же ID проходит фильтр, то его данные записываются в соотвесующий почтовый ящик. На почтов ящике ставить метка о новых данных. Если данные не прочитать до прихода нового пактеа, старые данные будут утеряны.