INTELLIGENT ARLIGHT

ОПИСАНИЕ СКРИПТОВОГО ЯЗЫКА DALI LOGIC USB

Содержание

О ДОКУМЕНТЕ

Это руководство по Smart Script для DALI LOGIC USB шлюза Arlight. Документ разделен на две части: (1) ядро интеллектуального движка USB-шлюза и (2) формат файла скрипта и описание языка.

ЯДРО SMART ENGINE USB DALI GATEWAY

СОСТАВ И НАЗНАЧЕНИЕ КОМПОНЕНТОВ

Script Runtime Engine: Ядро скриптового движка. Загружает и выполняет бинарные коды из Flash. Бинарный код инструктирует движок загружать/вычислять/сохранять данные в RAM или NVM, а также отдавать команды движку DALI-канала.
DALI Channel: Обрабатывает весь обмен между физическим DALI-каналом и скриптовым движком.
Thread Stack: Движок имеет 4 стека потоков, поэтому может выполнять максимум 4 потока одновременно. Все четыре потока разделяют время выполнения ядра и работают параллельно (конкурентно).
Local Variables: В каждом стеке потока доступно по 32 локальные переменные. Каждая переменная имеет размер 32 бита.
Ext Variables: Каждый поток может определять расширенные локальные переменные. Эти переменные отображаются в общую RAM (shared RAM).
Flash: Постоянное хранилище кода и данных. Содержимое может обновляться только через USB-интерфейс.
RAM: Оперативная (непостоянная) память. Общая для всех потоков. Каждая RAM-переменная — 32-битное значение.
NVM: Энергонезависимая память (постоянное хранилище). Может обновляться выполняемым кодом. Имеет ограниченное число циклов стирания/записи; не обновляйте содержимое с очень высокой частотой. Каждая NVM-переменная — 32-битное значение.

ФОРМАТ ФАЙЛА SMART SCRIPT

Файл состоит из двух секций: первая — объявления переменных, вторая — пользовательский код.

Пример структуры файла:

```
[Раздел 1]
#@DECLARATION
  Int #VarName1 = 0 @RAM[32]
#@END

[Раздел 2]

Terminate()
```

РАЗДЕЛ 1 — ОБЪЯВЛЕНИЯ ПЕРЕМЕННЫХ

Раздел 1 начинается директивой #@DECLARATION и завершается директивой #@END. Все, что находится внутри — объявления переменных.
Формат объявления:

[Type] [#NAME] = [Default] @[LOCATION]

Поля объявления:

Type — тип данных. Допустимые типы: BIT, BYTE, SHORT, INT.
Размеры: BIT — 1 бит; BYTE — 8 бит; SHORT — 2 байта; INT — 4 байта.
#NAME — имя переменной.
Default — значение по умолчанию. Если оно задано, компилятор добавит инструкции установки значения по умолчанию в начале кода.
LOCATION — где размещается переменная: RAM, NVM или FLASH.
Можно указать конкретный адрес, например @RAM[32].

Примеры объявлений:

INT #PEREMENNAYA_0 @RAM[0] — объявление переменной в оперативной памяти.

INT # PEREMENNAYA_0 = 100@NVM[0] — объявление переменной в энергонезависимой памяти. Тут значение 100 будет установлено по умолчанию.

Для доступа к локальным переменным потока используйте ключевое слово LV#nn, где nn — адрес локальной переменной.

СКРИПТОВЫЙ ЯЗЫК

Язык предоставляет арифметические операции, проверку условий и DALI-функции. Ниже приведено описание.

АРИФМЕТИКА

#Variable = Constant — Присваивание константы переменной.
#Variable ++ — Увеличить значение переменной на 1.
#Variable -- — Уменьшить значение переменной на 1.
#Var1 += #Var2 — #Var1 = #Var1 + #Var2.
#Var1 -= #Var2 — #Var1 = #Var1 - #Var2.
#Var1 *= #Var2 — #Var1 = #Var1 * #Var2.
#Var1 /= #Var2 — #Var1 = #Var1 / #Var2.
#Var1 &= #Var2 — #Var1 = #Var1 (побитовое AND) #Var2.
#Var1 |= #Var2 — #Var1 = #Var1 (побитовое OR) #Var2.
#Var1 ^= #Var2 — #Var1 = #Var1 (побитовое XOR) #Var2.
#Var1 = #Var2 + #Var3 — Оператор '+' также может быть заменен на '-', '*', '/', '&', '|', '^', '!=' и '~='.

ПРОВЕРКА УСЛОВИЙ

Синтаксис:

```
IF (condition1)
  {
  }
ELSE
  {
  }
```

IF и ELSE используются для проверки условия и выполнения разных частей кода. Если condition1 истинно — выполняется код в первой паре фигурных скобок, иначе — во второй.

ПЕРЕХОДЫ И ВЕТВЛЕНИЯ

Поддерживается безусловный переход GOTO. Для этого требуется объявить метку перехода.

Пример:

#LabelName:

...коды...

GOTO#LabelName

Примечание: метка должна заканчиваться символом ':' (двоеточие).

ВЫЗОВ ФУНКЦИЙ И ВОЗВРАТ

Движок поддерживает вызовы функций с глубиной вложенности до 4 уровней (то есть максимум 4 уровня CALL-RETURN внутри CALL-RETURN)

Инструкции:

CALL @Label — вызвать функцию по адресу/метке @Label.
RETURN — вернуться по адресу предыдущего CALL.

ТАЙМЕРЫ

Wait (#ms) — приостановить поток на #ms миллисекунд.
Timer.Create(#No, #tim) — создать таймер с номером #No и установить время #tim миллисекунд.
Timer.Wait(#No) — ожидать, пока таймер #No не истечет (timeout).
Timer.Clear(#No) — очистить таймер #No.
Timer.GetCounter(#No) — получить счетчик таймера #No; результат сохраняется в LV#12.

КОМАНДЫ DALI TRACE

Trace.IsMore() — Проверить, есть ли еще DALI-сообщения. Результат записывается в локальную переменную LV#0.
Trace.ClearAll() — Очистить все текущие DALI-сообщения.
Trace.GetNext() — Получить следующее DALI-сообщение и поместить его в LV#14...LV#16.
Trace.GetCurrent() — Получить текущее (по индексу) DALI-сообщение и поместить его в LV#14...LV#16.
Trace.Is102DirectDim() — Проверить, является ли текущее сообщение командой IEC 62386-102 direct dimming.
Trace.Get102TargetAddress() — Получить целевой адрес (target address) для текущего сообщения 102.
Trace.Get102TargetGroup() — Получить целевую группу (target group) для текущего сообщения 102.
Trace.Get103TargetAddress() — Получить целевой адрес (target address) для текущего сообщения 103.
Trace.Get103TargetGroup() — Получить целевую группу (target group) для текущего сообщения 103.

DALI-КОМАНДЫ

DALI.Send(#Channel, #Size, #Data) и **DALI.SendTwice(#Channel, #Size, #Data)** — прямые инструкции отправки DALI-команд. Каждый вызов сохраняет результат в LV#12.

Формат результата (LV#12):

Нижние 8 бит — данные ответа (reply data).
Биты 16...23 — количество байтов (numbers of bytes).

Примеры результата:

0x000100FF — команда отправлена, получен ответ 1 байт со значением 0xFF.
0x00FF0000 — ответ поврежден (corrupted).
0x80000000 — ответа нет (no reply).

Сигнатуры:

DALI.Send(#Channel, #Size, #Data)
DALI.SendTwice(#Channel, #Size, #Data)

Параметры:

#Channel — номер канала. (В примерах — 0.)
#Size — количество бит для отправки: 16 или 24.
#Data — данные для отправки.

Примеры:

DALI.SEND (0, 16, 0x80FE)
DALI.SEND (0, 24, 0x232011)

ДРУГИЕ DALI-КОМАНДЫ

DALI.Dim (#Channel, #Gear, #Lv) — Установить уровень яркости (диммирование) устройства.
DALI.DimGroup (#Channel, #Gear, #Lv) — Установить уровень яркости группы.
DALI.RecallScene (#Channel, #Gear, #Scene) — Вызвать уровень сцены устройства.
DALI.RecallGroupScene (#Channel, #Gear, #Scene) — Вызвать уровень сцены группы.
DALI.SetScene (#Channel, #Gear, #Scene, #Lv) — Установить уровень сцены устройства.
DALI.SetGroupScene (#Channel, #Gear, #Scene, #Lv) — Установить уровень сцены группы.
DALI.102JoinGroup (#Channel, #Gear, #Group) — Добавить устройство в группу #Group.
DALI.102LeaveGroup (#Channel, #Gear, #Group) — Удалить устройство из группы #Group.
DALI.SetFadeTime (#Channel, #Gear, #FT) — Изменить время плавного перехода (fade time) устройства.
DALI.SetGroupFadeTime (#Channel, #Gear, #FT) — Изменить время плавного перехода (fade time) группы.
DALI.Write102Bank (#Channel, #Gear, #Bank, #Location, #Data) — Записать данные в банк памяти устройства.
DALI.Write102BankNext (#Channel, #Data) — Записать следующий байт в ранее выбранное устройство/банк/позицию.
DALI.Write103Bank (#Channel, #Control, #Bank, #Location, #Data) — Записать данные в банк памяти устройства (DALI-103).
DALI.Write103BankNext (#Channel, #Data) — Записать следующий байт (DALI-103) для последнего записываемого устройства.
DALI.Enable103App (#Channel, #Control) — Включить контроллер приложений (application controller) устройства (103).
DALI.Disable103App (#Channel, #Control) — Выключить контроллер приложений (application controller) устройства (103).
DALI.Enable103Inst (#Channel, #Control, #Instance) — Включить канал устройства (103).
DALI.Disable103Inst (#Channel, #Control, #Instance) — Выключить канал устройства (103).
DALI.ActiveFeedback (#Channel, #Control, #Instance) — Активировать статус канала (feedback instance).
DALI.DeactiveFeedback (#Channel, #Control, #Instance) — Деактивировать статус канала (feedback instance).
DALI.ActFeedbackRGB (#Channel, #Control, #Instance, #Data/#R.G.B) — Установить цвет активированного статуса (feedback).
DALI.DeactFeedbackRGB (#Channel, #Control, #Instance, #Data/#R.G.B) — Установить цвет деактивированного статуса (feedback).
DALI.ActFeedbackBright (#Channel, #Control, #Instance, #Data) — Установить яркость активированного статуса (feedback).
DALI.DeactFeedbackBright (#Channel, #Control, #Instance, #Data) — Установить яркость деактивированного статуса (feedback).
DALI.FeedbackTiming (#Channel, #Control, #Instance, #Data/#P.D.C) — Настроить временные параметры статуса (feedback).
DALI.MuteAll (#Channel, #Control) — Отключить/заглушить все устройства (103 Start Quiescent).
DALI.UnmuteAll (#Channel, #Control) — Снова включить все устройства (103 Stop Quiescent).

КОМАНДЫ УПРАВЛЕНИЯ (CONTROL INSTRUCTIONS)

Self.BusDown (#Channel) — Отключить питание шины DALI (bus power off).
Self.BusUp (#Channel) — Включить питание шины DALI (bus power on).
Self.Event (#Para, #Data) — Отправить системные сообщения контроллерам системы.

Параметр #Para = Size (размер), с побитовым OR:
0x8000 = collision (коллизия)
0x0080 = Send (отправка)

ВАЖНО!!! ПРИМЕРЫ СКРИПТОВ И ИХ ЗАГРУЗКИ В КОНТРОЛЛЕР ВЫ МОЖЕТЕ НАЙТИ НА САЙТЕ:
https://smart-wiki.arlight.ru/dali/DALI-USB-Exmaples