SHILOV'S SLab LABORATORY

LitterA

ЛитерА LP-LED24.1

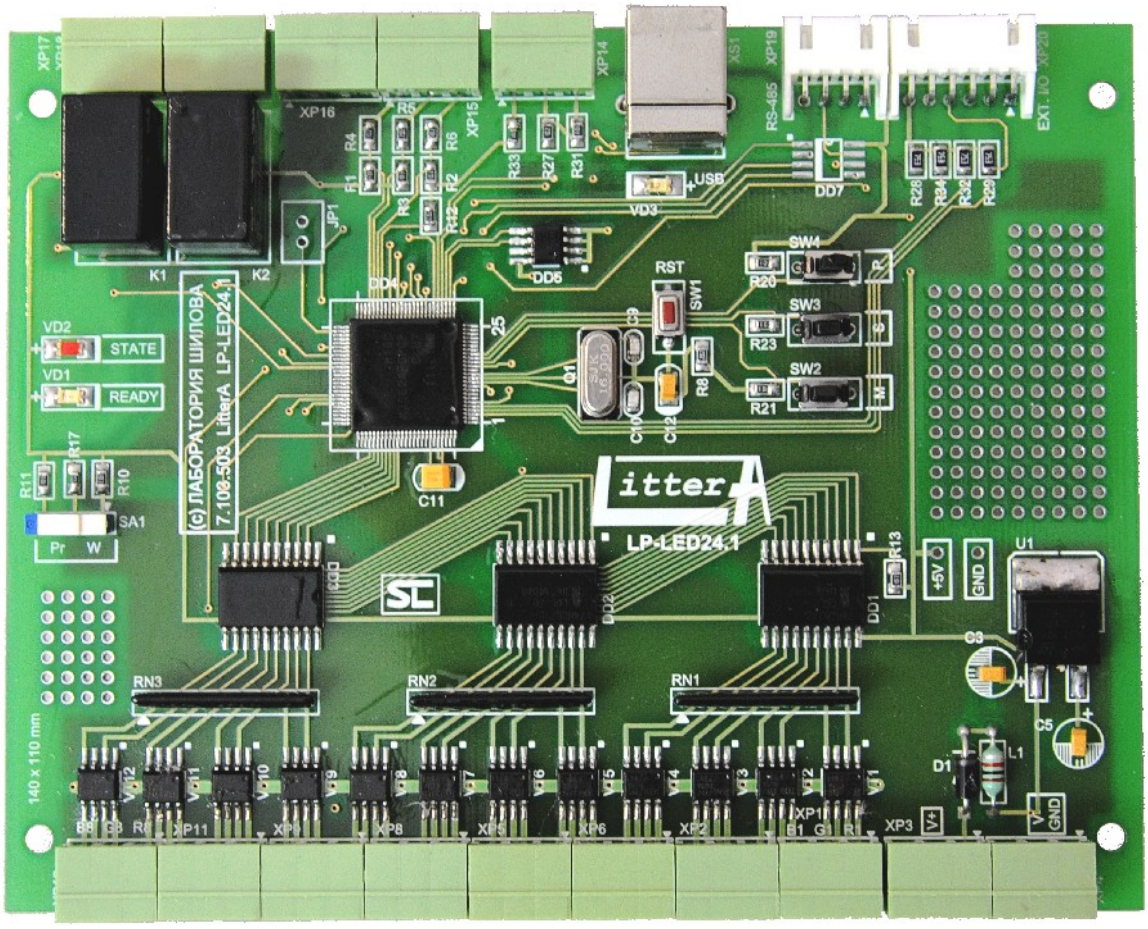
Контроллер управления светодиодами

Краткое руководство пользователя

1. Состав.

Плата состоит из следующих электронных компонентов:

- микропроцессор;
- 24 силовых ключа управления светодиодами (3 порта по 8 бит каждый);
- 2 реле (On-Off-On);
- интерфейс SPI ;
- модуль связи с ПЭВМ по шине USB;
- разъемные винтовые клеммники и разъемы для соединений с внешними устройствами.



вид платы контроллера

2. Основные технические характеристики.

Характеристика	*Значение*
Источник напряжения питания	внешний
Напряжение внешнего источника питания,В	+9...+30В
Защита от переполюсовки подключения внешнего ИП	есть
Всего каналов управления светодиодами из них полноцветных (RGB)	24 8
Напряжение питания каждого канала управления, В	+12...+30
Максимальный ток каждого канала управления, А	3
Количество реле управления	2
Режим коммутации каждого реле	частота < 2 Гц напряжение < 100В ток < 5А
Количество подключаемых внешних кнопок контроля (концевых выключателей)	3
Режим коммутации каждой кнопки	замыкание (U=5В I>1мА)
Связь с ПЭВМ	USB

2. Перечень используемых объектов.

Для использования доступны следующие объекты:

- **порт**, состоящий из 3-х объектов (порт_1 … порт_3), каждый из которых состоит из 8-ми объектов (8 бит), состояние которых (0/1) транслируется в силовые ключи;
  у каждого порта имеется свое свойство — яркость. (ярк_1...ярк_3).
- **свд**, состоящий из восьми объектов (свд_1...свд_8), каждый из которых представляет собой физический объект — RGB светодиод. Каждый объект имеет свойства: цвет и яркость.
- **реле** (реле_1, реле_2), состояние каждого включено / выключено;
- **кнопка** (кн_1...кн_3), состояние каждой нажата (включена) / отжата (выключена);
- переменная А, возможные значения = 0...255;
- переменная В, возможные значения = 0...255;
- **сохр_1 и сохр_2** — хранилища для временного сохранения свойств/состояния объектов.

3. Программирование контроллера.

3.1 Установка программного обеспечения LitterA LED24.

ОС Windows XP / 7 / 8 ®

1. Программное обеспечение не нуждается в установке (portable) и может располагаться в любом месте. Для начала работы — просто стартуйте файл LitterA_xxx.exe из рабочего каталога.
2. Для взаимодействия с платой необходимо наличие виртуального COM-порта, который устанавливается в операционную систему в виде драйвера.
   (см. установочный файл CP210x_VCP_Win_.xx.. .exe, где xx — версия ОС).
   После установки драйвера необходимо задать номер порта для чего:
   - подключить кабель USB к ПЭВМ и плате «LitterA»;
   - в свойствах «Мой компьютер» выбрать «Диспетчер оборудования» и далее в перечне устройств в разделе «Порты COM/ LPT» найти виртуальный COM-порт «Silicon Labs CP210x UART USB Bridge» и открыть его «Свойства». Далее перейти на вкладку «Параметры порта» - «Дополнительно» и установить имя порта «COM1» или «COM2» или «COM3». Подтвердить сделанные изменения.
3. После старта программы «LitterA_xxx» убедиться, что связь с ПЭВМ установлена. Если этого не произошло, следует нажать кн. «Настройки» и выбрать то же имя COM-порта, которое было установлено в настройках виртуального COM-порта в диспетчере устройств.

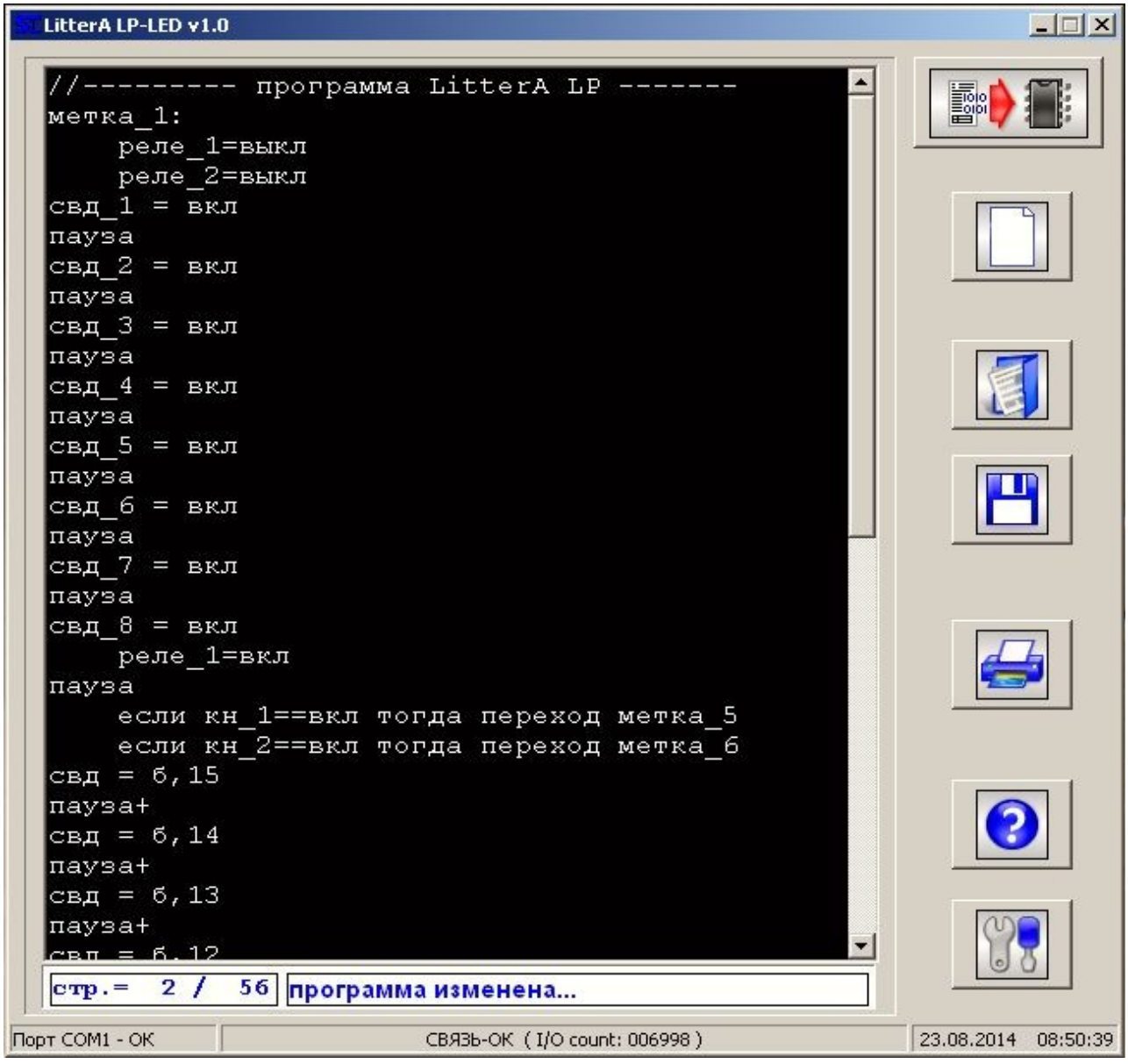
ОС Linux Ubuntu / Mint

1. Подключить кабель USB к ПЭВМ и плате «LitterA».
2. В консоле дать команду **`dmesg | tail -1`** и посмотреть к какому номеру ttyUSB произошло подключение устройства Silicon Labs CP210x.
3. В консоле дать команду **`ln -s /dev/ttyUSBX ~/.wine/dosdevices/com1`**
   где **X** — ранее установленный номер.
4. В консоле дать команду **`sudo chmod 777 -v /dev/ttyUSBX`**
   где **X** — тот же установленный номер.
5. Стартовать программу LitterA_xxx.exe в окружении Wine.
6. Убедиться, что связь с ПЭВМ установлена. Если этого не произошло,
   следует нажать кн. «Настройки» и выбрать имя COM-порта = COM1, в предположении, что физически COM-порт (RS-232) - отсутствует. Иначе, имя COM1 везде следует заменить на COM2. Если в ПЭВМ представлены оба COM-порта, следует указать имя COM3.

3.2 Режимы работы ПО LitterA.

Программа имеет два режима работы: режим редактора и режим консоли управления.

Редактор — предназначен для ввода текста программы, ее проверки и загрузки на исполнение в плату контроллера. Имеет стандартные функции: открыть / сохранить файл, распечатать файл. Формат файлов программ — текстовой (.txt). Таким образом, программы могут быть созданы и отредактированы любым текстовым редактором.
В правой части окна редактора имеется панель отладки, куда выводится информация в реальном времени о состоянии всех объектов контроля и управления. Панель доступна только при наличии связи с платой контроллера.
Редактор имеет настойки: цвета фона, цвета текста, номера порта связи.



вид окна Редактор

Описание кнопок управления (сверху вниз):

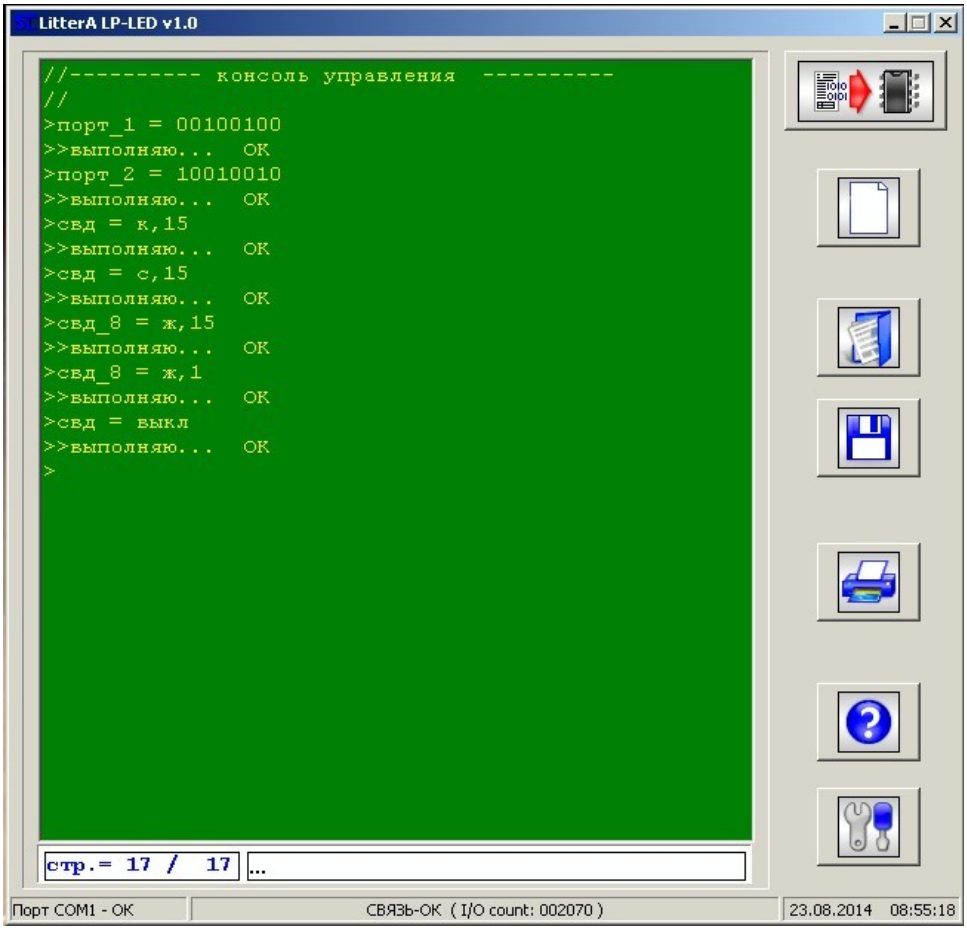
- проверить и загрузить (F8) — команда для загрузки программы в плату контроллера на исполнение, первым этапом выполнения команды является проверка программы. Результат проверки и загрузки выводится в строку сообщений.
- создать новый — очистка окна для ввода новой программы;
- открыть файл
- сохранить файл
- распечатать файл
- помощь (F1) — открытие окна с кратким справочником команд;
- конфигурация — открытие окна для задания конфигурации редактора.

Основные требования к тексту программ:

- длинна строки, включая пробелы и табуляцию — не более 128 символов;
- признак комментария — два символа // ;
- все команды вводятся маленькими буквами;
- все параметры команд допустимо вводить маленькими или заглавными буквами;
- всего строк программы — не более 1000.

Консоль управления — режим предназначен для проведения отладочных работ.

Переключение между редактором и консолью — кн. F10.
Команды управления объектами вводятся обычным образом и после нажатия Enter немедленно исполняются.



вид окна Консоль управления

3.3 Перечень используемых команд.

3.3.1 Команды управления объектами.

Объект порт

порт = вкл / порт = выкл	— вкл / выкл сразу все светодиоды
порт_x = вкл / порт_x = выкл	— вкл/выкл все 8 светодиодов (8 бит) порта x=1...3
порт_x.y = 1 / порт_x.y = вкл	— включить один светодиод порта (бит=1) y=1...8
порт_x.y = 0 порт_x.y = выкл	— выключить один светодиод порта (бит=0)
порт_x = 00010001	— установить соответствующие биты порта x в 0 или в 1 (соответственно, включить или выключить светодиоды). Нумерация бит слева направо.
ярк_1 = 4	— установить яркость порта 1 равную 4 (1...15)
ярк_2 = 7	— установить яркость порта 2 равную 7 (1...15)
ярк_3 = 5	— установить яркость порта 3 равную 5 (1...15)

Например

```
порт_1 = 00001111
ярк_1 = 3
```

будут выключены светодиоды 1-й/2/3/4-й и включены 5-й/6/7/8-й, при этом включенные светодиоды будут иметь яркость равную 3.

Объект свд

свд_x = к, 4	— светодиод x цвет Красный, яркость 4 (x=1...8)
свд_x = к	— светодиод x цвет Красный, яркость 15
свд_x = вкл	— светодиод x цвет Белый, яркость 15
свд_x = выкл	— светодиод 1 цвет черный (выключить)
свд = к, 4	— светодиоды 1-8 цвет Красный, яркость 4
свд = к	— светодиоды 1-8 цвет Красный, яркость 15
свд = вкл	— светодиоды 1-8 цвет Белый, яркость 15
свд = выкл	— светодиоды 1-8 цвет черный (выключить)

Параметр цвет: с — синий, з — зеленый, к — красный, ж — желтый, б — белый,
г — голубой, п — пурпурный, ф — фиолетовый.
Параметр яркость: 1...15.

ВАЖНО: параметр яркость у объектов порт и свд — не совпадает. При выполнении первой команды управления портами — активными становятся значения яркости портов (ярк_1...ярк_8), при выполнении первой команды управления объектами свд — активными становятся яркости объектов свд.

Объект реле

реле_х = вкл / реле_х = выкл — включить / выключить реле х=1...2

На плате контроллера используются механические реле с перекидным контактом, которым для замыкания / размыкания необходимо время несоизмеримо большее, чем время выполнения одной команды при исполнении программы. При написании программы, поэтому, необходимо контролировать, чтобы между командой реле_х = вкл и командой реле_х = выкл был промежуток времени не менее 0,5 сек. (см. команда Пауза).

Объект хранилище

В системе существует два объекта-хранилища (сохр_1 и сохр_2), предназначенных для временного сохранения статуса и состояния объектов управления.
Состояние (статус) объекта, помещенного в хранилище может быть изменено только новой командой помещения (сохранения).

Команда сохранения: **сохр_1 = объект (сохр_2 = объект)**

Команда восстановления: **объект = сохр_1 (объект = сохр_2)**

В качестве объекта могут быть: порт, порт_х, свд, свд_х, реле, А, В.

Например

```
        свд_1 = к, 15
        свд_2 = с, 15
        свд_3 = з, 15
        пауза++
        сохр_1 = свд
        свд = выкл
        пауза++
        свд_1 = ж, 15
        свд_2 = ж, 15
        свд_3 = ж, 15
        пауза++
        сохр_2 = свд
        А=1
    метка_1:
        А=А+1
        свд = сохр_1
        пауза++
        свд = сохр_2
        если А<8 тогда переход метка_1
        . . .
```

здесь: создание и использование двух шаблонов объекта СВД.

3.3.2 Команды управления скоростью выполнения команд.

Контроллер при выполнении загруженной программы выполняет команды одну за другой последовательно. Время выполнения любой команды составляет очень малую величину (менее 1/1000 секунды). Поэтому, часто, для наблюдения результатов необходимо иметь возможность выполнения команд через заданный промежуток времени.

Например

```
свд_1 = вкл
свд_1 = выкл
```

в данном примере команда на выключение выполнится сразу же за командой на включение,и, как следствие — будет казаться, что RGB-светодиод не включался вовсе.

пауза	сделать паузу до выполнения следующей команды в 1 секунду
пауза+	сделать паузу до выполнения следующей команды в 0,1с (100 мс)
пауза++	сделать паузу до выполнения следующей команды в 0,2с (200 мс)
…	
пауза++++	сделать паузу до выполнения следующей команды в 0,5с (500 мс)

Например

```
свд_1 = вкл
пауза
свд_1 = выкл
```

включить RGB-светодиод и через 1 секунду выключить.

3.3.3 Команды выполнения арифметических операций.

Для простых арифметических операций существуют две переменные: A и B.
ВАЖНО: допустимые значения переменных = 0...255.

A=значение	**B=значение**	— присвоить значение переменной
A=A+значение	**B=B+значение**	— сложить переменную со значением
A=A-значение	**B=B-значение**	— вычесть значение из переменной
A=A+B / **B=B+A**		— сложить переменные
A=A-B / **B=B-A**		— вычесть переменные
A=B / **B=A**		— приравнять переменные

ВАЖНО: при наборе команд арифметики — пробелы не допускаются!

3.3.4 Команды управления ветвлением программы.

метка_х: - — установить в программе точку для перехода x=1...15
переход метка_х — перейти на метку (в точку перехода)
возврат — перейти на команду, следующую за последней выполненной командой «переход метка_х»

Например

```
    свд = выкл
    переход метка_2
метка_1:
    свд_7 = с
    пауза++
    свд_7 = з
    пауза++
    свд_7 = к
возврат
метка_2:
    свд_1 = вкл
    переход метка_1
    свд_2 = вкл
    переход метка_1
    порт_3 = вкл
    переход метка_1
        ...
```

здесь:

*после старта программа выключит RGB-светодиоды и перейдет в точку метка_2: включит RGB_1 белым и перейдет в точку метка_1: включит RGB_7 синим, через 0,2 сек зеленым, через 0,2с - красным, и выполнит команду **возврат**, т. е. перейдет в точку «свд_2 = вкл». Т.о. можно многократно использовать одну последовательность команд.*

3.3.5 Команды проверки условия.

Команды проверки условия имеют одинаковый синтаксис:

если условие **тогда** команда

Условием может быть :

кн_х==вкл	**кн_х==выкл**	— кнопка включена / выключена x=1...4
А==значение	**В==значение**	— переменная равна значению
А>значение	**В>значение**	— переменная больше значения
А<значение	**В<значение**	— переменная меньше значения

ВАЖНО: при проверке условий равенства вводится двойной знак равно!

Командой может быть любая команда, рассмотренная выше.

Например

```
если В==0 тогда А=А+1
если кн_1==вкл тогда реле_1 = вкл
если В>100 тогда В = 0
если кн_3==выкл тогда свд_1 = к
```

3.3.6 Команды консоли управления.

В качестве команд могут быть команды управления портами, ключами, реле и звуком.

Дополнительные команды

ППЗУ = записать записать в микросхему ППЗУ программу, которая сейчас загружена для исполнения в плату контроллера.

После включения питания платы или аппаратного сброса, программа будет считана из ППЗУ и может быть сразу же исполнена (ком. Старт). Это позволяет быть плате автономной и работать без участия ПЭВМ.
(ППЗУ — перезаписываемое постоянное запоминающее устройство).

ППЗУ = стереть очистить микросхему ППЗУ

сброс выполнить полный перезапуск платы
очистить очистить окно консоли управления

3.4 Порядок программирования контроллера.

1. Подключить плату контроллера к ПЭВМ посредством USB.
2. Если используется внешнее питание — включить питание.
3. Стартовать программу Littera LED.
4. Открыть файл с ранее сохраненной программой или составить новую программу.
5. Загрузить программу для исполнения в контроллер (F8).
   Программа начнет исполняться.

   ВАЖНО: в отличие от многих существующих систем, предназначенных для обучения программированию, в контроллерах LitterA программы исполняются в оперативной памяти (ОЗУ) контроллера. При этом, программа, хранящаяся в ППЗУ контроллера никак повреждена или изменена быть не может, пока не будет перезаписана посредством команды консоли управления. Таким образом, в контроллере одновременно существуют две программы: одна - в ППЗУ (автоматически считывается в ОЗУ после включения питания / аппаратного сброса) и вторая — в ОЗУ, исполняемая в текущий момент времени.
   Это позволяет проводить отладку программы и загружать ее на исполнение столько раз, сколько требуется.

   ВАЖНО: так так программа всегда исполняются в ОЗУ контроллера — сам микрокод прошивки контроллера никогда изменен или поврежден быть не может. Какие бы ошибки не были допущены в алгоритме программы пользователя — контроллер всегда будет работать, с ним всегда можно установить связь.

   ВАЖНО: программа, загруженная на исполнение в контроллер и не записанная в ППЗУ при выключении питании — будет утеряна.

6. В случае принятия решения на запись программы в ППЗУ — необходимо воспользоваться консолью управления. Максимальное количество циклов записи в ППЗУ контроллера составляет 10 000 раз.