DC11548
Driver 1.00
May 2025
F. Devescovi

Interfacing FlashRunner with ST Microelectronics LDLL16EN led driver

1. Introduction

ST offers a large portfolio of constant-current LED row drivers and LED array drivers to meet stringent automotive requirements.
The LDLL16EN is a monolithic 16-channel LED driver designed for automotive exterior and interior LED lighting applications. It guarantees up to 20 V output driving capability, allowing users to connect several LEDs in series, and features 16 regulated current sinkers able to provide from 6.3 mA to 100 mA programmable current to drive LEDs. The LDLL16EN integrates a robust purely automotive CAN FD light compatible serial communication interface which allows a high rate data transmission (up to 1 Mbit/s) and uses CAN FD structure for long frames.
This differential interface offers a detailed diagnostic of the device itself, as well as of the controlled LED strings, and makes the device suitable for high data rate transmission.
The device can be controlled by microcontroller via CAN FD light compatible interface. Stand-alone operation (no microcontroller used) is also possible thanks to direct drive functionality and full configurability by FTP (Few Times Programmable) registers. They can be programmed max. 1000 times and no counter is stored both in the LDLL16EN and FlashRunner.

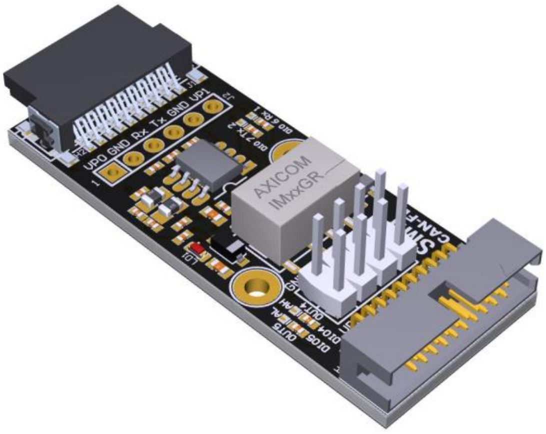
You can download the latest version of this document from this static link:
Interfacing FlashRunner with ST Microelectronics LDLL16EN led driver

2. Contents

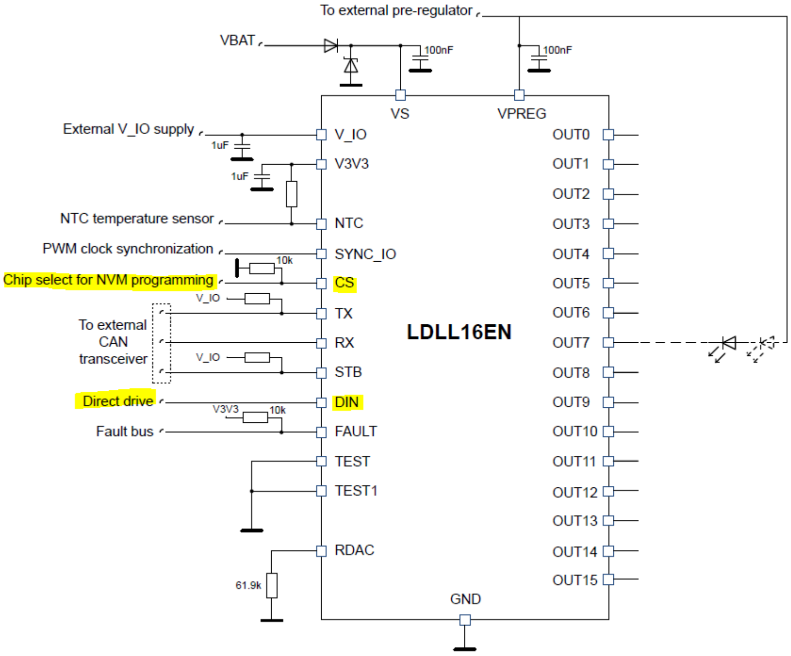
3. Tips and tricks to write LDLL16EN FTPs

In this section, we want to explain what operations are performed in the target devices and how these operations work. This knowledge may help you to correctly setup your project so to be suitable to your application.

The FlashRunner uses CAN FD protocol to access the FTP (Few-Times Programmable) non-volatile memory of this device.
To accomplish that, a CAN-FD hardware adapter is mandatory.



Communication frequency is fixed and equal to **1 Mbit/s**.
For technical reasons that we don't mention here the **DIN** pin of all the devices sharing the same CAN FD Light network **must be pulled-up to 5V** (on the board itself or externally).



Let us analyse the main algorithm commands:

- **BlankCheck**
  This command checks that all the bits of **FTP rows 0, 1, 2** and **12** are set to **0**.
  All the other FTP rows will be skipped since they are not writable.
  Optionally, it is also possible to send two additional parameters to this command: the address from where to start checking <start_address> and the number of bytes to check <size>. The alignment of both of them must be **FTP_ROW_SIZE = 15**.

  ```
  #TPCMD BLANKCHECK F
  ```

  Since the main purpose of the Blankcheck is to check whether the device is "Factory Shipped" **no DEVICE_ADDRESS has to be specified** and the **CS pin must be connected** to Flashrunner's **DIO5**.

- **Program**
  This command takes Customer's data from the FRB file and programs them into the FTP memory, also reading them back.
  Even if the whole FTP memory size image is provided only **FTP rows 0, 1, 2** and **12** will be programmed. All the other FTP rows will be skipped since they are not writable.
  Optionally, it is also possible to send two additional parameters to this command: the address from where to start programming <start_address> and the number of bytes to be programmed <size>. The alignment of both of them must be **FTP_ROW_SIZE = 15**.
  Furthermore the **CS pin** has to be connected to FlashRunner's **DIO5**.

  ```
  #TPCMD PROGRAM F 0x0 0x2D
  ```

  The above command will program FTP rows 0, 1, 2.

- **Verify Readout**
  This command checks that data contained in FTP memory of the device corresponds to FRB data.
  Even if the whole FTP memory size image is provided only **FTP rows 0, 1, 2** and **12** will be checked. All the other FTP rows will be skipped since they are not writable.
  Optionally, it is also possible to send two additional parameters to this command: the address from where to start checking <start_addess> and the number of bytes to check <size>.
  The alignment of both of them must be **FTP_ROW_SIZE = 15**.

  ```
  #TPCMD VERIFY F R 0x0 0xF
  ```

  The above command will verify FTP row 0 data.

This command works exactly like the program command with the only exception that the device reads (instead of writing) data from the memory and compares that with the data received from FlashRunner.
Since the mechanism is totally equal to the one used by the program command, if any error was introduced during the program command, it is possible that the same error could be introduced during the verify command and this could lead to a possible undetected error. For this reason, we suggest to also use the read/dump feauture.
The main purpose of verify is to check whether the device has been properly programmed.
While programming data FlashRunner also reads them back the VERIFY READOUT needs a **DEVICE_ADDRESS** to be specified before and can't be performed within the same power cycle of the PROGRAM because the programmed device address will become effective after a device reset. The same goes for READ and DUMP.

```
#TPCMD READ F 0xF 0xF
```

The above command will print to terminal FTP row 1 data.

4. Example 1 – Program U400 device

```
;DEVICE: LDLL16EN
;DRIVER: STLD 01.00
;PINMAP: DIO1=RX, DIO2=TX, DIO5=CS (Chip Select)

!ENGINEMASK 0x00000004
#LOADDRIVER libstld.so STMICROELECTRONICS LL16EN LDLL16EN
#TCSETDEV VDDMIN 5000
#TCSETDEV VDDMAX 5000
#TCSETDEV MEMMAP 0 F 0 0x00000000 0x000000EF 0x0 0xF 0 16 0x0 0x0 0x0 0x0 0
!CRC 0x142655BE
#TCSETPAR PROTCLK 1000000
#TCSETPAR PWDOWN 100
#TCSETPAR PWUP 100
#TCSETPAR VPROG0 5000
#TCSETPAR CMODE CAN
#TPSETSRC S3_LH_U400.frb
#TPSTART
#TPCMD CONNECT
#TPCMD PROGRAM F
#TPCMD DISCONNECT
#TPEND
```

5. Example 2 – Verify Readout U400 device

```
;DEVICE: LDLL16EN
;DRIVER: STLD 01.00
;PINMAP: DIO1=RX, DIO2=TX, DIO5=CS (Chip Select)

!ENGINEMASK 0x00000004
#LOADDRIVER libstld.so STMICROELECTRONICS LL16EN LDLL16EN
#TCSETDEV VDDMIN 5000
#TCSETDEV VDDMAX 5000
#TCSETDEV MEMMAP 0 F 0 0x00000000 0x000000EF 0x0 0xF 0 16 0x0 0x0 0x0 0x0 0
!CRC 0x142655BE
#TCSETPAR PROTCLK 1000000
#TCSETPAR DEVICE_ADDRESS 20
#TCSETPAR PWDOWN 100
#TCSETPAR PWUP 100
#TCSETPAR VPROG0 5000
#TCSETPAR CMODE CAN
#TPSETSRC S3_LH_U400.frb
#TPSTART
#TPCMD CONNECT
#TPCMD VERIFY F R
#TPCMD DISCONNECT
#TPEND
```