LED Control

- Signalling Led
- LCD Backlight
- KEYPAD Backlight

Contact information

Scognamiglio Gaetano

E-mail: gaetano.scognamiglio@neonseven.com

NEONSEVEN

Services offered by the LED driver

- Signalling led:
  - Application:
    - Charger events: connected, disconnected, battery low, charging completed
    - MS status: In service, out of service, Limited Service or in Call
- LCD backlight:
  - Action: ON, OFF, and FADING
- KEYPAD backlight:
  - Configuration of up to 3 color LEDs with RGB coding (8 bit for each color), including color-combinations

Signalling LED

Main characteristics:

- Use EPOWERlite led driver
- Are switchable low side constant current sources
- Can be controlled by the bits SLED1ON and SLED2On in register LEDCTRL2.

NEONSEVEN

LCD Backlight LED

Main characteristics:

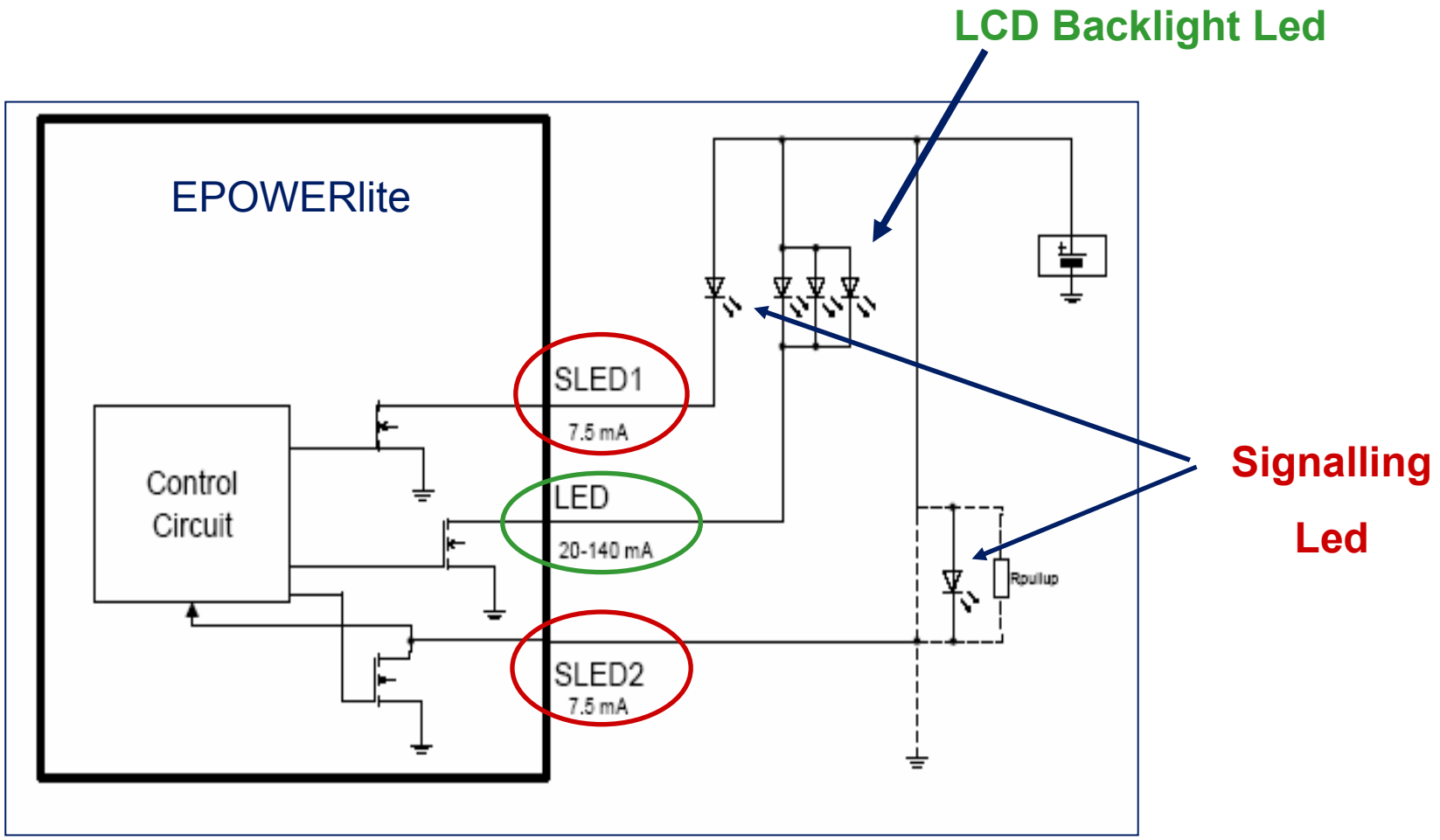
- Use EPOWERlite led driver
- Is suitable for driving most commercially available LEDs in the red, orange, yellow and green range.
- Is a switchable low side constant current source.
- Is programmable in 7 steps in the range from 20 mA to 140 mA with a step width of 20 mA

NEONSEVEN

LCD Backlight LED

- The anode of the LEDs can be attached directly to the battery.
- Consists of a PWM control with a fundamental frequency of about 60 Hz generated from the internal clock frequency of 500 kHz by division by 8192.
- Using register LEDCTRL1 the duty cycle of the LED current can be set in 63 steps in the range from 0.5% to 100%.
- By sending a sequence of appropriate commands to EPOWERlite a soft dim function can easily be implemented.

LCD Backlight and Signalling control



NEONSEVEN

Keypad Backlight LED

Main characteristics:

- Configuration of up to 3 color LEDs with RGB coding (8 bit for each color), including color-combinations
- Action: ON, OFF, FLASH, COLORSTREAM, FADING
- Timing parameters can be changed run time
- Often used parameters can be stored in the EEPROM
- Low power consumption due to synchronization with the GSM/GPRS timing

NEONSEVEN

Keypad Backlight LED

Functional description

- LED driver uses three different capcoms in compare mode 3 to change the intensity of each color of a multicolor led.

- LED intensity is linearly proportional to time intervals during which the led is on. The intensity can be increased or decreased by adjusting the duty cycle D of the voltage supplied, defined as

$$D = \frac{t_{ON}}{t_{ON} + t_{OFF}} = \frac{t_{ON}}{Ts} = fs\ t_{ON}$$

where t_{ON} and t_{OFF} are the time interval during which the LED is in and off; $Ts = t_{ON} + t_{OFF}$ is the duration of a cycle.

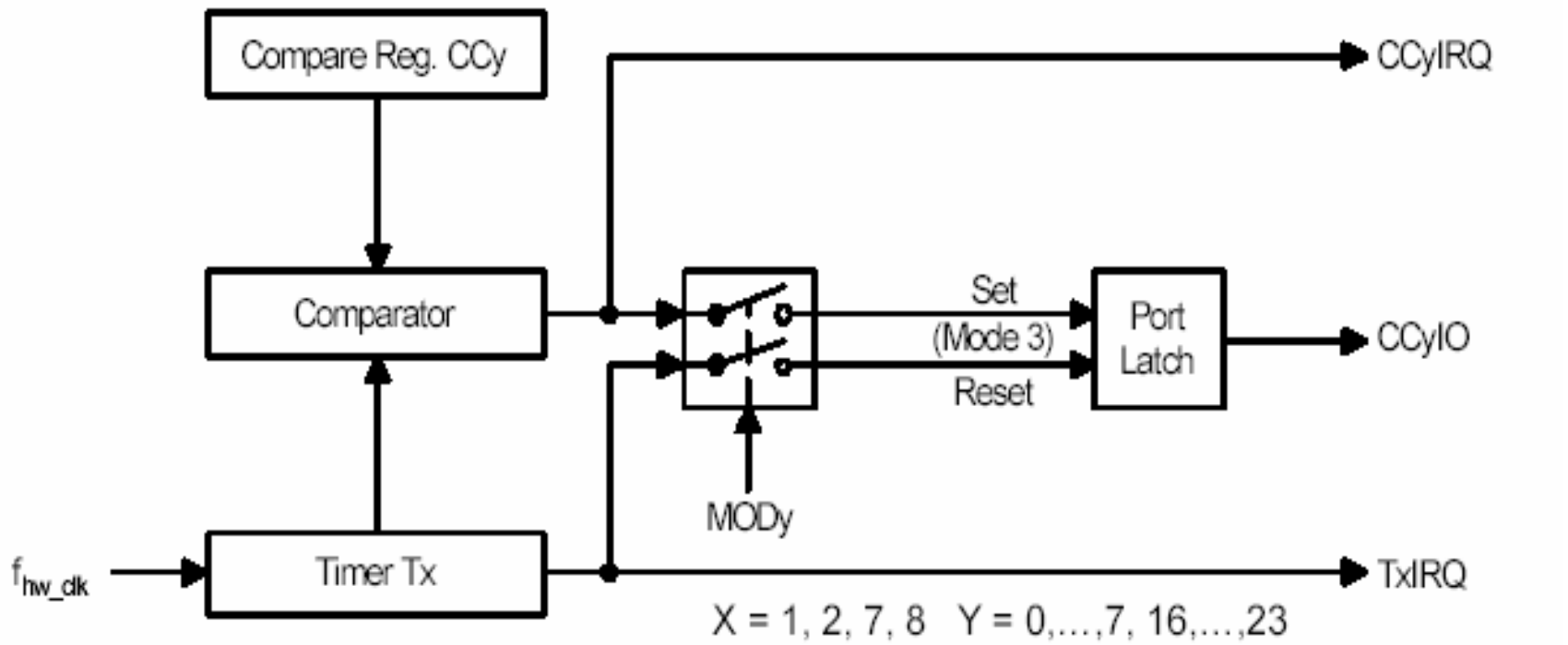
This pulse width modulation is generated using output CCyIO of capcom in compare mode 3

CAPCOM compare mode 3 overview

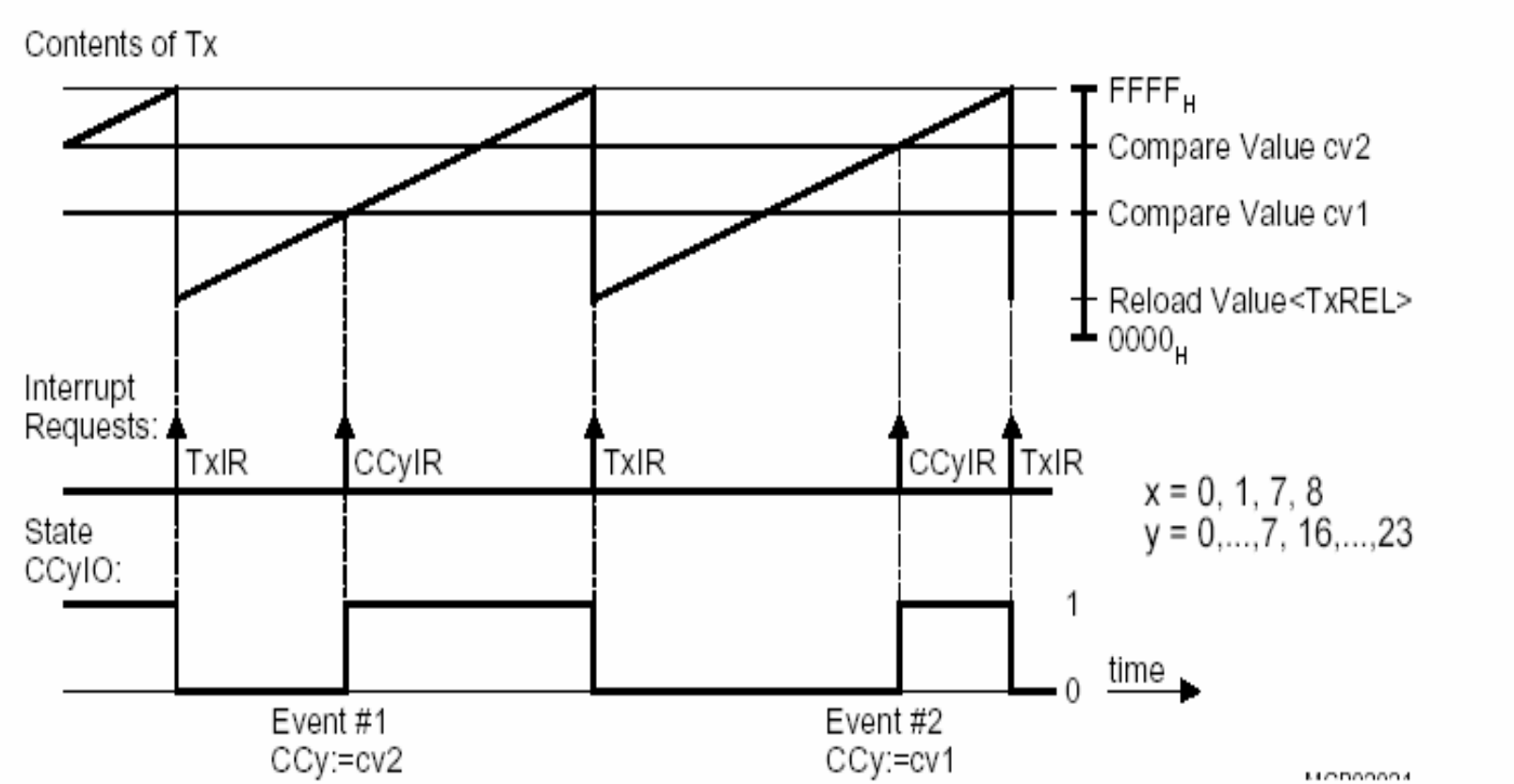
- Compare Mode 3 is selected for register **CCy** by setting bit field **CCMz.MODy** of the corresponding mode control register to 111B.
- Only one compare event is generated per timer period.
- When the first match within the timer period is detected the interrupt request flag CCyIR is set and the output pin CCyIO is set.
- The signal is cleared when the allocated timer overflows.

NEONSEVEN

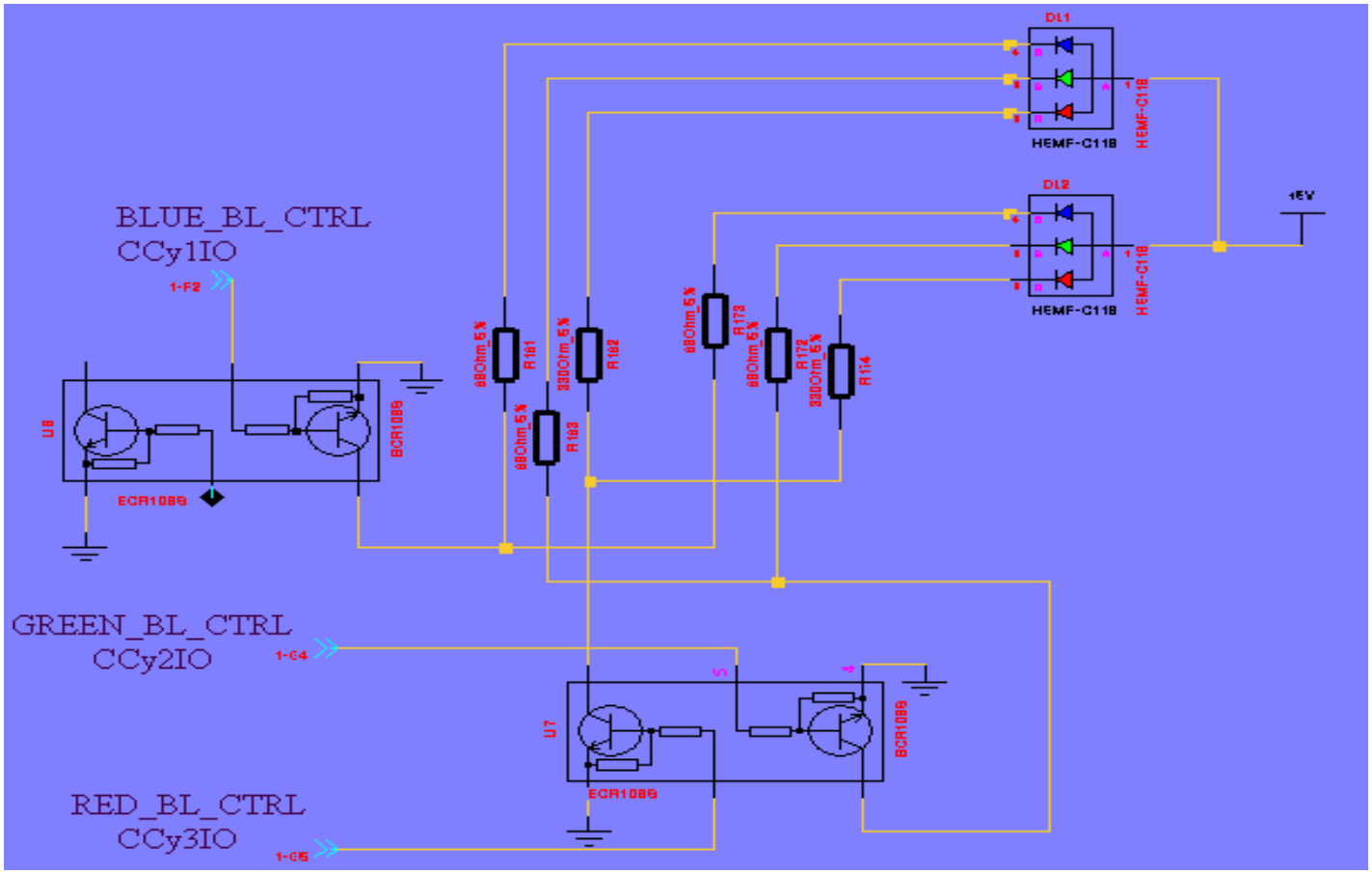
CAPCOM compare mode 3 block diagram



Timing example for compare mode 3



Multicolor led control



NEONSEVEN

Example of configuration

- Configure CCMz:
  - to allocate CAPCOM CCy to the respective timer Tx
  - to configure CCy in compare mode
- Set TxREL = FF00H to get 255 different intensity levels (Timer Tx overflows from FF00H to 0000H)
- Set the period P_{Tx} between two consecutive overflows of Tx greaten than 200Hz

$$P_{Tx} = \frac{(2^{16} - FF00H) \cdot 2^{(<TxL>+3)}}{fhw\_clk}$$

- Set CCyIO as output
- Change Led intensity by setting a value for CCy between FF00H and FFFFH

LED interface

- To change Led intensity

  LED_mmi_change_intensity(led_color_type color, ulong intensity)

  led_color_type: led_blue, led_red, led_green and combinations

  intensity: RGB led intensity;

  The first byte is empty and following are used respective for red, green and blue component.

NEONSEVEN

LED interface

- All programming of the service leds are done with

LED_main_control(color, action, total_time, on_time)

Color: led_blue, led_red, led_red_green, etc.

Action: LED_ON, LED_OFF, LED_FLASH,
LED_COLORSTREAM, LED_FADING

Total_time (only when flashing):
The cycle time (led_on + led_off)

On_time (only when flashing):
The time the leds are on

NEONSEVEN

Example of Led actions

Red led on forever:

```
LED_main_control(led_red, LED_ON, 0,0);
```

Blue led on for 1000 ms

```
LED_main_control(led_blue, LED_ON, 1000, 0);
```

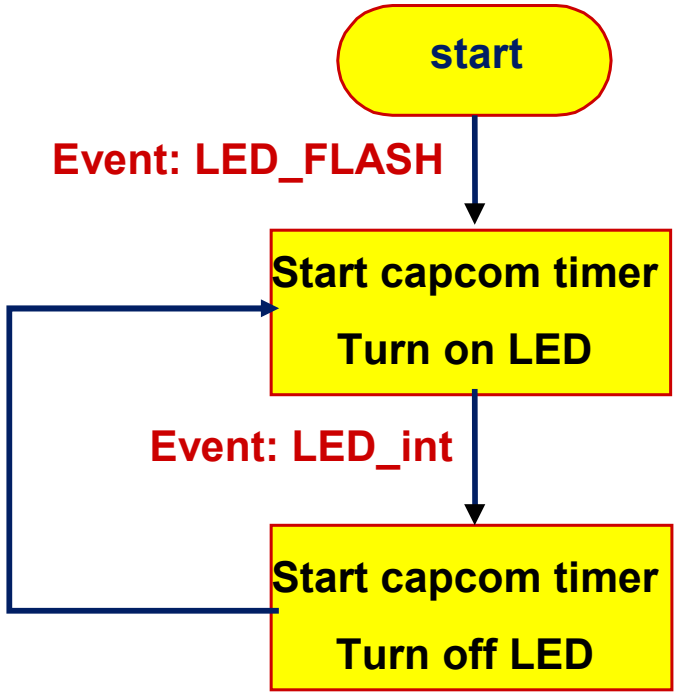
Green led off:

```
LED_main_control(led_green, LED_OFF, 0, 0);
```

NEONSEVEN

Led flashing

Handled using CAPCOM in compare mode 2 (is an interrupt-only mode; only one interrupt request per timer period is generated)



Led flashing

Example

Color: yellow

On_time: 40 ms

Total_time: 3 seconds

LED_main_control(led_red_green, LED_FLASH, 3000, 40)

NEONSEVEN

Led colorstream

Colorstreaming is a combination of colors, that changes rapidly.

- Support of up to 10 different combinations.
- The order of the colors are determined at run time.
- Each color can use individual timing
- Default colorstream timing is 250 ms ON/OFF
- Use CAPCOM in compare mode 2

Configure the colorstream:

```
LED_mmi_colorstream(ushort len, led_stream_data_type *led_stream_data);
```

Start colorstreaming:

```
LED_main_control(0, LED_COLORSTREAM, 0, 0);
```

NEONSEVEN

Led colorstream

- E.g. for streaming 3 colors, with different time periods, the led_stream_data structure is defined as:

```
led_stream_data_type led_stream_data[3];
led_stream_data[0].color = led_blue;
led_stream_data[0].total_time = 500;
led_stream_data[0].on_time = 200;
led_stream_data[1].color = led_green;
led_stream_data[1].total_time = 2000;
led_stream_data[1].on_time = 500;
led_stream_data[2].color = led_red;
led_stream_data[2].total_time = 0;
led_stream_data[2].on_time = 0;
LED_mmi_colorstream (3, led_stream_data);
LED_main_control(0, LED_COLORSTREAM, 0, 0);
```

Led parameters in EEPROM

Parameters for often used events (”led_service”, ”led_call” etc) can be stored in the eeprom.

The saved parameters are: Color, intensity, on_time and total_time.

Saving:

BOOL LED_mmi_set_eep_colors(led_event_type event, led_conf_type *config);

Getting:

led_conf_type *LED_mmi_get_eep_colors(led_event_type event);

Phone Tool – Led & Backlight