Getting started with the STSW-ILL066V2 firmware for LED lighting applications

Introduction

The STSW-ILL066V2 firmware runs on the STLUX385A digital controller to enable and manage the features and functionality of the STEVAL-ILL066V2 LED evaluation board. The firmware implements state machine, event driven (SMED) architecture to guarantee real time MOSFET protection.

Together, the firmware and hardware form a complete and configurable solution to manage a single high brightness LED string using the STLUX385A(For more information about STLUX385A and the complete STLUX product family, please refer to the STLUX Family Datasheet and STLUX Reference Manual.) digital controller to drive two power conversion stages as well as handling the DALI protocol and the 0-10V bus. The STEVAL-ILL066V2 topology implemented is a PFC + LC with primary side regulation, which doesn't require secondary side feedback and hence doesn't use any opto-couplers, increasing the LED driver lifetime.

The firmware uses the following libraries to manage power conversion:

- The first library manages the PFC hardware, regulates the PFC output voltage changing the PFC ON or OFF time, provides protection and monitors PFC activity.
- The second library manages the half bridge hardware, regulates the output current using the ZVS principle and monitors the output load.

1 STSW-ILL066V2 source code overview

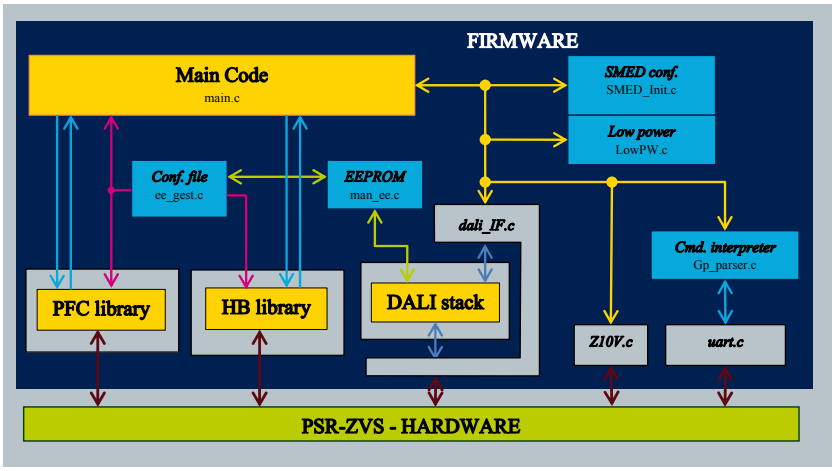
The STSW-ILL066V2 firmware code manages the two-stage digital converter and certain interfaces. The two stages drive the PFC MOSFET and the half bridge (HB) MOSFETs using the SMED and two dedicated libraries.

The main code manages the firmware start-up phases, calls other enabled modules like DALI or 0-10V interface and monitors the board. The source files (modules) are listed below:

- **main.c**: the main source code
- **ee_gest.c**: the application EEPROM area handler (the DALI stack manages its own EEPROM area directly)
- **man_ee.c**: the EEPROM driver for writing to and reading from the EEPROM area through polling
- **LowPW.c**: the low power routines when the DALI stack switches off the output current
- **GPGui\gp_parser.c**: the serial command interpreter to manage the GPGUI commands
- **SMED_Init.c**: the SMED configurator output file used to initialize the SMED behaviour
- **stlux_itc.c**: subroutines to handle interruptions (not the interrupt handlers)
- **stlux_optonbyte.c**: restores the default STM8 option byte (used only for Raisonance compiler)
- **dali_IF.c**: the DALI driver and the interfaces from DALI stack and main code
- **Dali_Stack**: the DALI stack directory
- **uart.c**: the serial line driver
- **Z10V.c**: the 0-10V interface driver
- **temp_ntc.c**: manages board temperature
- **HB_lib**: the half bridge library directory for the ZVS stage
- **PFC_lib**: the PFC library directory for the PFC stage

The interaction between the modules is illustrated in Figure 1. Source code interaction - top view

Figure 1. Source code interaction - top view



1.1 main.c

This file contains the entry for the “c” program after “c” initialization. This code executes the entire main task and calls all the libraries and subroutines. This file initializes the input and output signals, calls the hardware initialization, the `ee_gest.c` module and the necessary routines to initialize all the firmware modules.

During the initialization phases, the ADC is configured to continuously convert the entire eight-input channel. To obtain the value of one ADC channel during normal execution, read the last ADC conversion from the ADC output register. Using the 6 MHz ADC clock, one channel is converted every 3 µs and a new conversion on the same channel is available every 24 µs.

`main.c` contains the reference for the STMR interrupt routine, which is the base timer to monitor and control all the hardware. The STMR interrupt routine also calls the PFC and HB algorithms and sets flags to inform other routines when 100 µs or 10 ms has elapsed. The routine updates a 32-bit internal counter every second.

`main.c` calls the `ee_gest.c` module to populate the RAM area with values stored in the EEPROM data area. Some variables also define which interfaces are active and whether the PFC and HB libraries are enabled.

`main.c` also checks whether the STLUX type is valid (STLUX385A); if this check fails, execution is disabled and the device enters debug mode with only the serial line remaining active and all the other board functions disabled. When new code is loaded in the STLUX, debug mode is automatically enabled. You can exit debug mode through the GPGUI program.

`main.c` also checks the internal STEVAL-ILL066V2 board power. If the internal power voltage is not between VD_14V (12V) and VCC (3.3V), a warning message is generated on the serial line, but execution continues. During start-up, the IWDG peripheral is enabled to reset the STLUX if an infinite loop occurs. The predefined IWDG reset time is 10 ms.

When all the initial startup routines have been executed, the main module enters the main loop and the firmware checks and calls certain routines in the PFC and HB libraries. If new data is available from external interfaces (UART, DALI, 0-10V), it calls the appropriate subroutines to perform necessary actions.

Whenever the main loop concludes without any pending actions, the STM8 core enters a wait for interrupt (wfi) state to reduce power.

To check the STLUX core load, simply check the ON time of the green LED with an oscilloscope. When the green LED is on, the STLUX core is active and is executing code; when the green LED is off, the STLUX CPU executes the wfi command.

1.2 ee_gest.c

This file contains the routines to manage the EEPROM file system used to modify STEVAL-ILL066V2 evaluation board behavior; it contains a data structure to identify all the modifiable parameters.

The main calls the `ck_ee()` routine to check if the EEPROM data area is initialized; if not, the default value is loaded before continuing. Following this initial check, the `ck_ee()` routine initializes all the RAM variables using the values stored in the EEPORM data area.

The routine then checks the parameter #10 (ROP) value to initialize the ROP register if necessary. When the ROP register is set to TRUE, the parameter file is read only and no modification is possible.

This file system is read or modified through the GPUGI program. When you modify a parameter value, the `ee_set()` routine is called. New data in this area is only applied during startup, not on the fly.

This module uses the *man_ee.c* module to physically write to and read from the EEPROM.

1.3 man_ee.c

This file provides the low level write and the read routines used by the DALI stack and the *ee_gest.c* routines to manage the EEPROM contents; it is possible to write and read a byte (u8) and a word (u16).

An EEPROM read or write operation waits for the previous EEPROM write to end before actually reading or writing any data. This allows the interrupt routine to start. If a new write operation starts before the previous operation has ended, the CORE waits until the end of the write operation (stops the CORE clock).

1.4 LowPW.c

This file provides the low power module used to enter or exit low power mode. It is called by the *main.c* file when the DALI stack sets the LED to off.

Low power mode involves switching off PFC and HB functionality, as well as all the unnecessary peripherals. When the off procedure ends, the module calls the wfi instruction and modifies the internal watchdog time to 500 ms.

When the DALI RX pin receives a high-to-low transition, the CPU is activated at the maximum clock frequency (16 MHz) and checks the relative DALI RX frame. If the output power in the DALI RX frame is active, the module exits low power mode and restores full STLUX functionality; otherwise, it returns to low power mode after two seconds. This ensures very low power consumption when the output current is switched off.

When the CORE enters low power mode, all the others communication channels are disabled.

1.5 gp_parser.c

This file provides the command line interpreter for UART commands. It analyzes and validates all the characters received by the serial line and then executes any corresponding commands.

This module works in conjunction with the *uart.c* file which provides the UART driver.

1.6 SMED_Init.c

This file is the optimization of the output of the SMED configurator program. It is called by *main.c* to initialize the SMED registers during the start-up phases.

This file does not activate the SMED function; it only loads a predefined configuration value on the SMED register.

1.7 stlux_itc.c

This file provides the `ITC_SetSoftwarePriority()` function to change the default interrupt priority level; it also provides the TRAP and NMI interrupt entry routines.

The TRAP and the NMI interrupts on the STEVAL-ILL066V2 evaluation board are not used.

1.8 stlux_optonbyte.c

This file provides the function to restore the default option byte value when the code is downloaded through the SWIM interface. This file also provides the entry to enable or disable the boot serial loader by changing the option byte value.

To manage the boot serial loader procedure, use the GPGUI program.

When the serial load is enabled, the application program waits one second for the boot serial loader startup procedure before executing.

This one-second wait is automatically removed when a new program is loaded.

1.9 dali_IF.c

This module is the interface between the main and the DALI stack. It manages the DALI peripheral interrupts and informs the DALI stack when a good frame is received. It also manages the DALI byte transmission using another interrupt vector.

The `Dali_Init()` function manages the startup of the DALI interfaces, sets the baud-rate, sets the 500 ms timer to detect the disconnection of the DALI bus, and initialize certain DALI stack structures and variables. It also starts the AUXILIARY timer to manage the 1 ms timeout.

The `DALI_F_Disable()` function disables DALI communication and the DALI interrupt. This function is called by the main when the main power is removed.

The `DALI_F_Enable()` function enables DALI communication and the DALI interrupt.

The `DALI_Enable()` function initializes and enables the DALI communication and the DALI interrupt.

The `DALI_CheckAndExecuteReceivedCommand()` function is called by the main when a new frame is received by the DALI bus and calls the DALI stack to take necessary action.

The `Get_DALI_Random()` function is called by the DALI stack to get a u8 random value.

The `Send_DALI_Frame()` function is called by the DALI stack to schedule the transmission of the u8 byte. The physical transmission starts after 3 ms.

The `DALI_CheckAndExecuteTimer()` function is called by the main when the 1 ms timer elapses. It calls the function associated with the DALI FADE function and other timer-based functions.

The `set_dimm()` function manages the output current request from the DALI stack. It activates and deactivates the low power module when necessary.

This module contains the DALI interrupt routine that is called when a new DALI frame is received or a byte is transmitted on the DALI bus.

1.10 Dali_Stack

The entire DALI stack is located in a separate directory. It only requires the CCO interrupt function for the internal timing (1 ms) and the DALI tx-rx interrupt to receive and transmit the DALI frame.

In this application, the DALI interface is mutually exclusive with the 0-10V interface.

The DALI stack provides a DATA structure and functionality compliant with the IEC 62386-102 ed.2.0 and the IEC 62386-207 rev1.0 requisites.

The DALI stack consists of:

- *dali_cmd.c* to implement the DALI command interpreter and execution, as well as provide the memory location for the two memory banks required by the IEC standard.
- *dali_config.c* module to store the data structure use by the DALI stack. It stores:
  - the default value for DALI variable
  - the internal value table (logarithmic or linear mode) used by the DALI stack to set the output current level
  - the FADE, EXTENDED FADE, FAST FADE and FADE rate time table used by the DALI stack to calculate the corresponding time or step rate.
- *dali_pub.c* module to implement all the internal DALI stack subroutines used by the STACK to manage the DALI protocol.
- d*ali_regs.c* module to implement and manage the internal DALI register.
- *eeprom.c* module to implement and manage access to the EEPROM area register used by the DALI stack.
- *external_EEPROM.c* module not used on this DALI stack.
- *iec62386_201to210.c* module to implement and manage the DALI 207 extended command.
- *lite_timer.c* module to implement the timer routine used by the DALI stack.

Note: Any change, including a simple code recompile, requires a new run of the DALI test suite to obtain the DALI certification.

1.11 uart.c

This module manages the serial line data transfer.

The default baud rate is 115200 bps but you can modify the source code to use a different baud rate.

This module implements an input/output drive using the UART interrupt to read and write each byte on the serial line. The buffer length is modified by changing the TXSIZE (default is 160 bytes) and RXSIZE (default is 16 bytes) constant used to send or receive the byte.

This module also contains the interfaces to the standard `printf` subroutine.

Note: Never insert a `printf` function in the interrupt routine code as this will generate an error.

1.12 Z10V.c

This module manages the 0-10V interfaces using the voltage on the ADCIN[3] input pin and CCO output pin; the CCO output pin creates the frequency used for this interface.

The 0-10V interface is mutually exclusive with DALI interfaces; only one interface can operate at one time.

When the 0-10V interface is used, the serial command is limited regarding the output current: the UART interface is not able to change the output current using the dedicated command because the 0-10V interface takes precedence and overwrites the GPGUI command.

1.13 temp_ntc.c

This module manages board temperature using the voltage on the ADCIN[7] input pin.

The external component used is the STLM20W87F ultra-low current 2.4 V precision analog temperature sensor. The relationships between the ADC value and temperature are recorded in the adc_temp[] table. If you change the external component, you may need to update this table.

The main calls the `read_NTC` function to acquire the external temperature. When a new temperature reading is available (800µs x 64 loops = 51 ms), a check is performed to determine whether the temperature is inside the defined range. If the temperature is too high, the `therm_overload` routine is called to return the required amount of output current reduction.

2 PFC and HB libraries

There are two libraries in the STEVAL-ILL066V2 evaluation board firmware: one to handle the PFC HW circuit and one to handle the half bridge HW circuit.

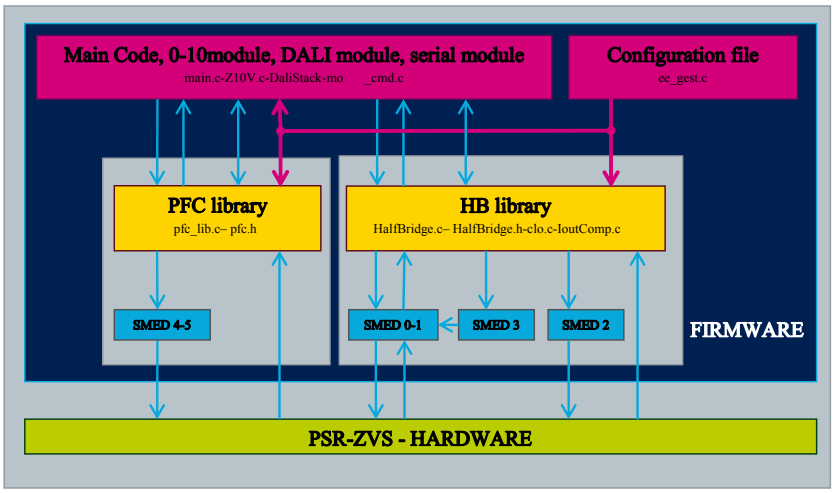
The libraries are referenced with an include file that include all the variables and subroutines called from other source codes.

These libraries also contain some interrupt vectors.

All the modifiable parameters used into these two libraries are loaded during startup, before the PFC and HB functions are enabled. To modify the startup parameters, use the GPGUI program and serial line.

The following diagram shows the interaction between each library and other source code components (main, command interfaces and configuration files).

Figure 2. Library interaction - top view



The libraries receive their parameters from a configuration file(see Variable parameter description) during power-on (after reset). The file is stored into EEPROM area with parameters relating to SMED activity, external hardware and library functionality. It also controls board functionality.

The configuration file is modifiable using the GPGUI over a serial line, but new data is only applied after a power-on reset.

Certain parameters are modifiable on the fly (power-on reset not required) through dedicated GPGUI commands. These direct commands do not change the startup parameters.

Note: Parameter files that may be modified through the GPGUI are identified herein with the corresponding firmware index preceded by the "#" (hash) character.

2.1 PFC algorithm

It is important to understand the control loop of a PFC compensation network to achieve control objectives. In Figure 3. Control loop of PFC stage: block diagram, $G3_{(S)}$, $G4_{(S)}$ and H are implemented in the hardware, while Vref and the PI algorithm are implemented in the STLUX firmware.

The STLUX creates the necessary ON-OFF time and the SMED event takes the ZCD event and protection (not figured) into account.

Figure 3. Control loop of PFC stage: block diagram

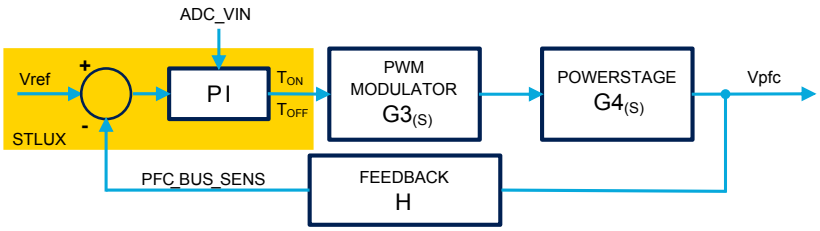
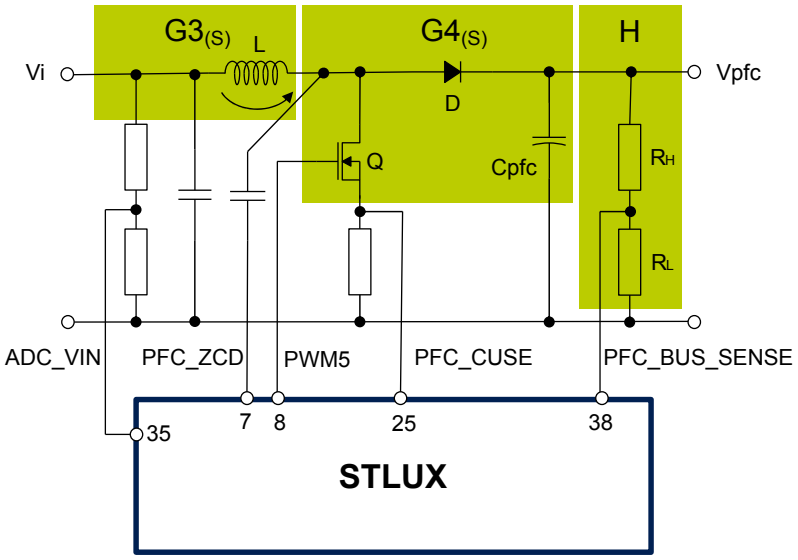


Figure 4. Control loop of PFC stage: electrical circuit and main quantities illustrates how the various blocks in Figure 3. Control loop of PFC stage: block diagram relate to the internal and external circuitry of the STLUX. The PFC loop gain must have a very low crossover frequency (fc) to keep PWM5 reasonably constant over a given half line cycle and ensure a high PF.

Inside the SMED, the PFC current peak is only detected to protect the PFC stage. The SMED also directly manages the ZCD event to start the ON time when inductor valley currents occur.

Figure 4. Control loop of PFC stage: electrical circuit and main quantities



Users can select appropriate hardware component based on personal experience regarding analog chips. The values of P and of I can be modified through the four parameters defined in the parameters file.

The STLUX implements a standard Proportional-Integral (PI) algorithm, but also has a gain scaler to respond quickly during PFC load changes. The parameters #46 and #47 define the PFC limit when fast response is active; #48 defines the correction slope when the PFC level is below #47 and fast response is active.

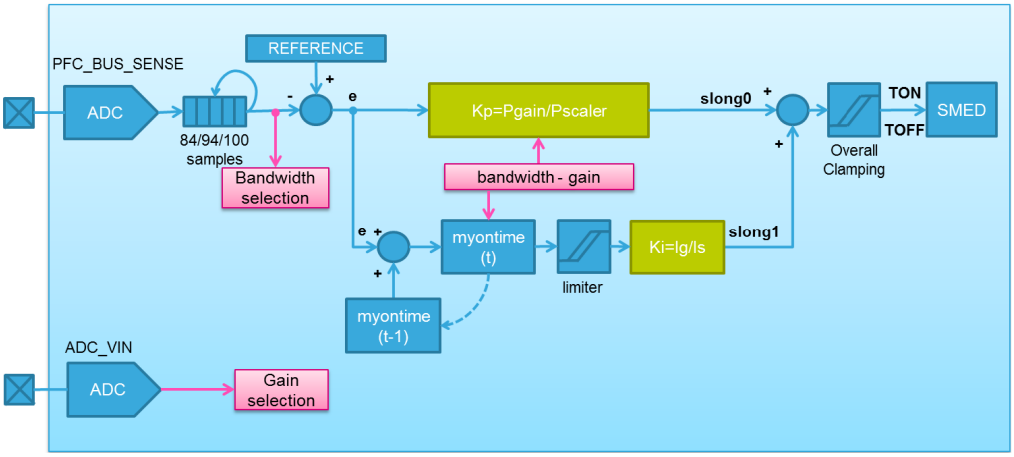
Figure 5. STLUX PFC algorithm shows the PFC algorithm implemented on the STEVAL-ILL066V2 evaluation board. You can adapt the four PI parameters to your own hardware (see Section 4 Variable parameter descriptions).

The PI loop senses the PFC output voltage using the PFC_BUS_SENSING ADC input voltage. The ADC samplings are accumulated in an internal 32-bit variable and are applied every half-AC cycle using the Kp and Ki parameters. The result of this PI algorithm defines the PFC TON-TOFF time used during the following half-AC cycles. The accumulation phases are performed using a sampling rate of 100 µs. With this timing rate, the samples are 84 at 60 Hz and 100 at 50 Hz.

As the input frequency is unknown when the board starts, a fixed 55 Hz (93 samples) setting is used. The PFC output voltage target point used in this PI algorithm is fixed at 1.1143 V at the ADC input pin (PFC_BUS_SENSE - STLUX pin 38).

The outputs of the PI algorithm are clamped to a maximum PFC_Ton (1800 on SMED5.S3 register) and to a maximum PFC_Toff (65520 on SMED4.S2 register).

Figure 5. STLUX PFC algorithm



The bandwidth selection function accelerates the PFC reaction during transaction (load change increases or decreases). When the value of the accumulation is outside a predefined range (approx. +3.35% or -6% with respect to the target PFC - see parameters #46 and #47), the value of Kp gain is increased or decreased by 25%.

A Gain selector in the PI algorithm is applied to the Kp gain. This value depends on the input AC value acquired on the ADC_VIN (STLUX pin 35).

The gains applied to Kp are shown in the following table.

Table 1. Kp Gain function of Vac input voltage

Input voltage (STLUX pin 35) (ADC_VIN)	Approximate Vac input on STEVAL-ILL066V2 board	Gain applied on Kp
More than 0.899 V	More than 233 Vac	1/4
0.899 V to 0.727 V	233 Vac to 189 Vac	2/4
0.727 V to 0.555 V	189 Vac to 144 Vac	3/4
Below 0.555 V	Below 144 Vac	4/4

The standard PI equation in literature is:

$$u(t) = Kp\,e(t) + Ki\int_{t_0}^{t} e(\tau)d\tau \quad (1)$$

A simplified version is:

$$R_{PI}(s) = K_P\frac{1+T_IS}{T_IS},\ where\ T_I = \frac{K_P}{K_I} \quad (2)$$

By default, the STEVAL-ILL066V2 evaluation board with the V3R36 FW revision implements the following parameters when the PFC output voltage is within the #46 and #47 range.

$$Kp = \frac{\#\,44}{2^{\#45}} * Kp\_g = \frac{130}{2^{13}} * Kp\_g = 0.058691 * Kp\_g \quad (3)$$

$$Ki = \frac{\#\,42}{2^{\#43}} = \frac{140}{2^{15}} = 0.004272 \quad (4)$$

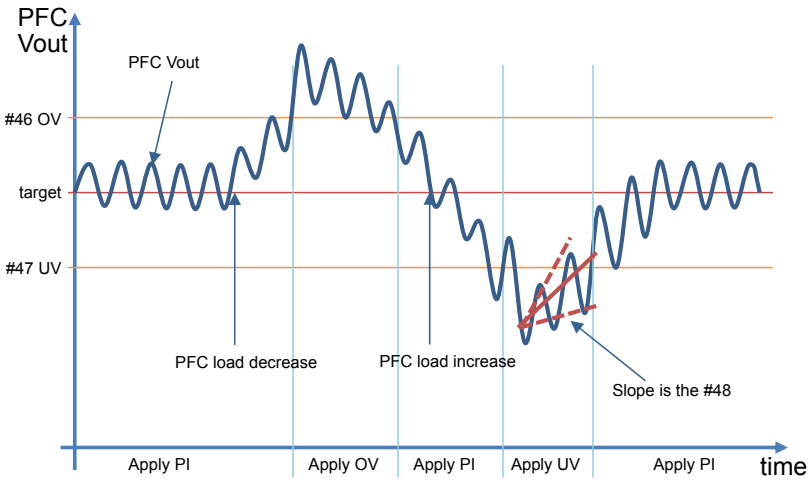
Where:

#42, #43, #44 and #45 are the values in the parameters file (see Section 4 Variable parameter descriptions)

Kp_g is a function of input Vac (see Table 1. Kp Gain function of Vac input voltage)

The figure below shows the relation for the PFC PI implementation.

Figure 6. STLUX PFC PI parameter relationships



2.2 Half bridge algorithm

The half bridge topology is a voltage fed LC converter. A frequency defines the current on the LC tank and, as the used frequency is five times higher than the resonant frequency, the tank current is approximated to a triangular shape. The primary pick current is nearly half of the output current if the output transformer ratio is 1:1.

The real output transformer ratio used on STEVAL-ILL066V2 evaluation board is 1:1.85.

On the STEVAL-ILL066V2 evaluation board, the Iout_setup parameter in GPGUI modifies the output current. This value defines certain internal behavior(The relationships between the raw Iout_setup values and the output current are defined in the STEVAL-ILL066V2 user manual.)

Figure 7. Half bridge signal relations shows the relationship between the output current, HB frequency and CN_CNT voltage. The three functional zones are described below.

HBZ_SKIPK

This zone defines the lower output current zone. The algorithm implemented here uses the SMED3 to force the half bridge to start and stop appropriate activity.

The SMED3 on/off time is a function of the output current requested by the user (from DALI, 0-10V or GPGUI command). To set the lower output current (Iout_setup=539), the SMED3 activates the half bridge frequency for one cycle only, and sets it off for 125 cycles.

Similarly, this zone sets the maximum output current when all the 126 cycles set the HB switching active. The half bridge frequency in this zone is fixed to a maximum frequency defined by the #31 parameter file.

This zone is selected when the “Iout_setup” raw value is between 539 and 1100.

HBZ_FIXFR

This zone is a small and unstable area which connects the other two zones.

This zone is selected when the Iout_setup value is between 1100 and 1150.

HBZ_CLOSE

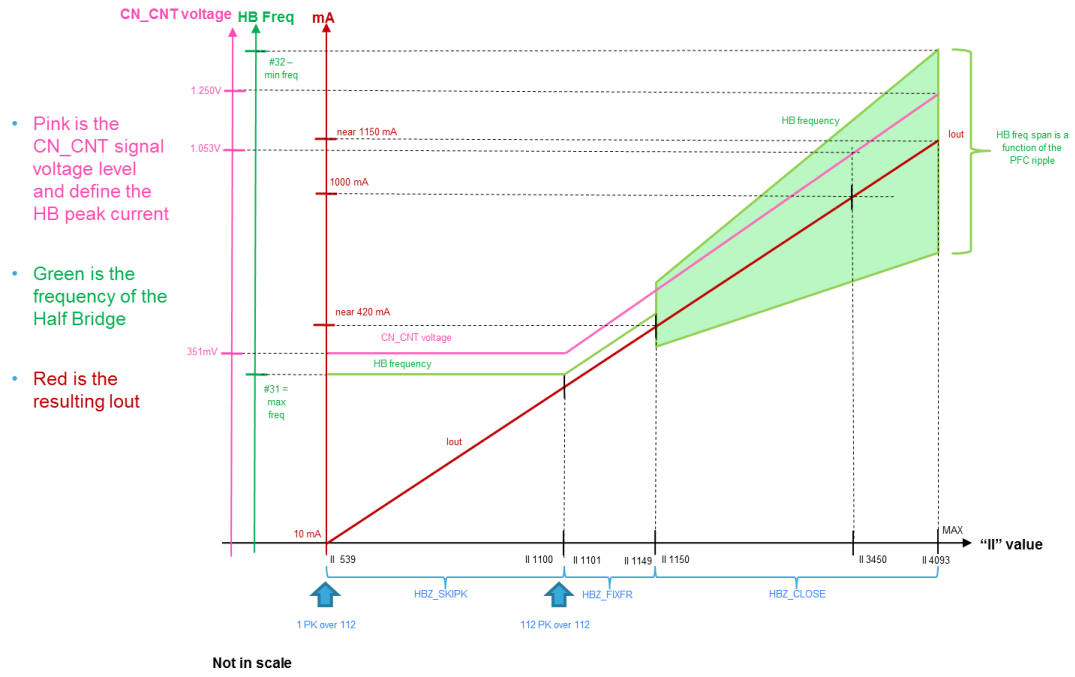
This zone controls the output current using the voltage on the CN_CNT signal, which is a function of the output current defined by the user.

The half bridge frequency is moved to a set value during the output current transaction. After the output current transaction, the half bridge frequency is regulated by the CN_CNT signal, whose voltage is a function of the required output current.

The relationship between the Iout_setup raw value and output voltage is (Iout_setup/4)*(ADC_top/ADC_level)

This zone is selected when the Iout_setup raw value is between 1150 and 4093.

Figure 7. Half bridge signal relations



Not in scale

3 Library interfaces

This section describes the functions called by the main (or other source module) that control the PFC and the half bridge hardware.(To identify the signal names and components described here, refer to the STEVAL-ILL066V2 evaluation board schematics available on www.st.com.)

Each library module has routines to initialize your variables, to start or stop the output PWM PIN, some other routines are called during the STMR timer interrupt.

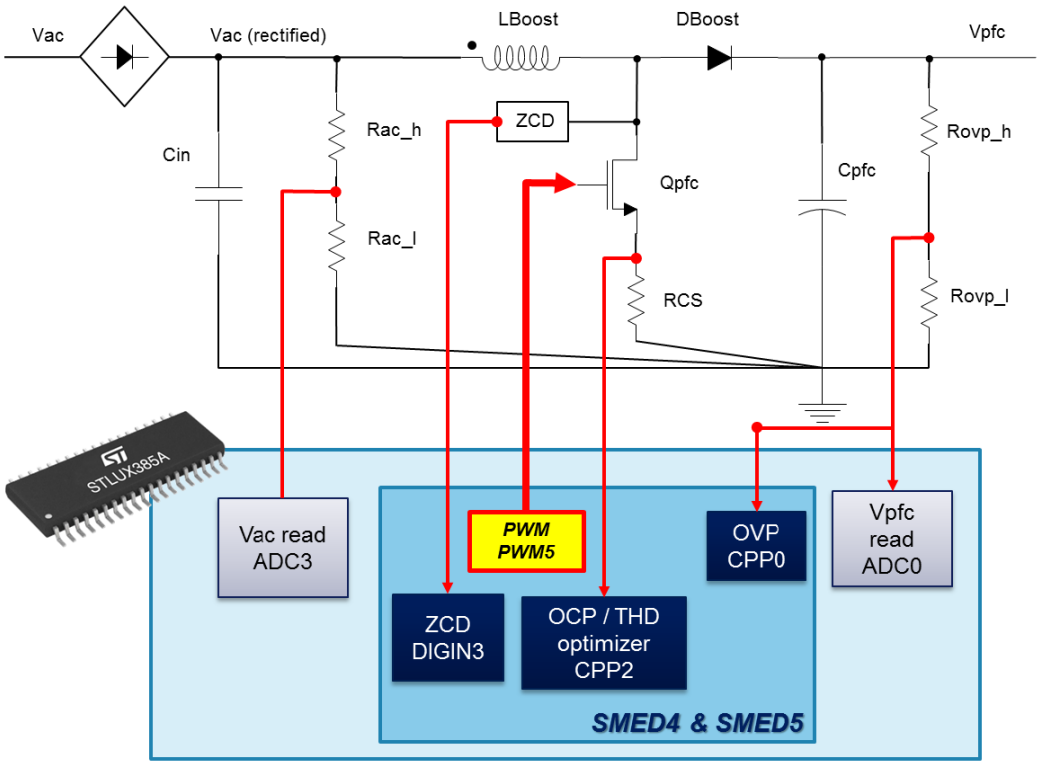
There is a global variable to control the module behavior.

Inside every library module there is a routine called during the associated SMED interrupt.

3.1 PFC library Interface

Certain functions in the PFC library manage the PFC PWM signal. The PFC module is implemented using the SMED4 and the SMED5 resources, and controls the SMED5 PWM output signal connected to the PFC gate driver.

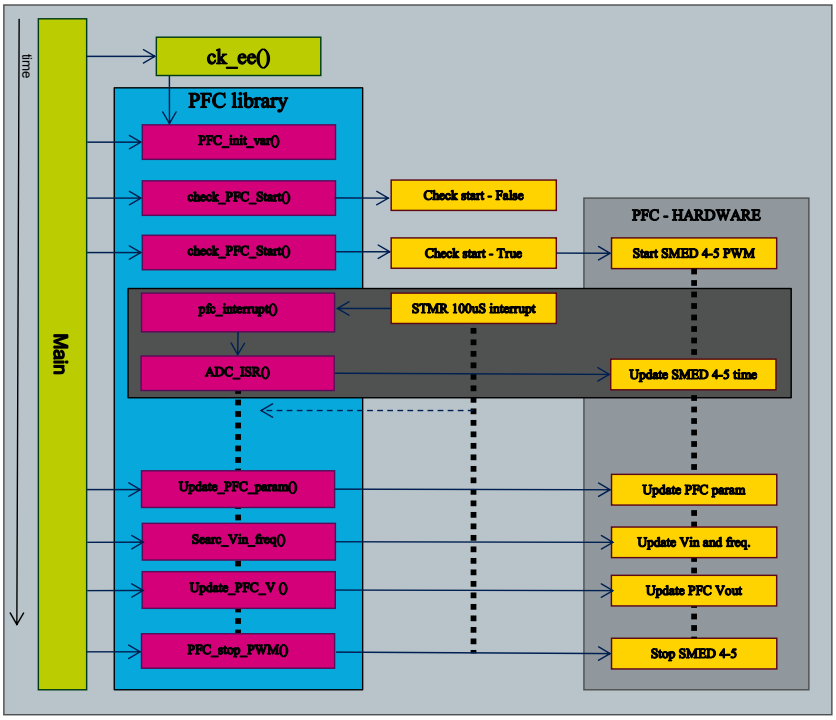
Figure 8. PFC concept



The PFC library module references the SMED4 and SMED5 interrupts.

Figure 9. PFC library concept offers a system overview of the STLUX385A PFC library implementation. It shows the call sequence of the most important PFC library routines.

Figure 9. **PFC library concept**



3.1.1 u8 PFC_init_var(void)

The main calls this routine to initialize the PFC internal variable. Certain variables are stored in the EEPROM data area and the main loop calls the `ck_ee` subroutine before calling this routine.

You should only call this function once during start-up.

The number returned by the subroutine identifies the PFC library revision.

3.1.2 void check_PFC_Start(u8)

The main calls this function to start the PFC when it is not running, which first checks certain external signals and variables to verify that the PFC can be activated.

For example, the PFC can only be started when the input rectified Vac voltage is greater than a predefined level (`PFC_in_min`). When the parameters passed to this subroutine equal 0, the routine detects the start-up time overlap, sets an error (`ERR_PFC_OT`) and activates the restart time procedure (using `SERV_START` service).

This routine also calls other internal functions to read the rectified Vac voltage and to determine the input frequency.

3.1.3 void pfc_interrupt(void)

This routine interrupts the PFC interrupt and starts the PFC control loop procedure.

The main.c file calls this function during the STMR timer interrupts. This routine is synchronized by the STMR interrupt routine enable the ADC interrupt routine to start the PFC control loop procedure.

This routine also calls other internal functions to read the rectified Vac voltage and to determine the input frequency.

This routine also manages the PFC start-up procedure during the `check_PFC_Start()` routine.

3.1.4 INTERRUPT_HANDLER (ADC_ISR, 22)

This routine is called by the ADC ISR interrupt when a new ADC conversion is available and only when the STMR interrupt requests it. It accumulates the PFC output voltage in one AC cycle. When an AC cycle ends, the routine calculates the new ON/OFF time and updates the SMED register for the next half-AC cycle.

This routine uses a Proportional Integration algorithm. The Proportional and Integration values are defined in the parameters file, which you can modify to suit your hardware.

This interrupt routine implements the entire PFC main algorithm. To regulate the PFC output voltage, the PFC on time or the PFC off time are modified.

The function is not called when the PFC is stopped.

3.1.5 u8 Update_PFC_param(void)

This function updates the PFC parameters according to the input AC voltage. It checks the input voltage through parameters #22, #23 and #24 in the parameter file.

There are three ranges:

- low range (typically a USA range)
- medium range (between low and high)
- high range (typically the European range).

The altered PFC parameters define the PFC behavior according to the input voltage. For example, define the OCP protection levels according the input voltage.

This subroutine is called every 800µs.

The return parameters identify when a change is performed during the call:

- 0 = no change performed during the call
- 1 = high level European range (default)
- 2 = middle range between EU and USA ranges
- 3 = lower level USA range.

3.1.6 void PFC_set_vout_story(void)

This subroutine is called every 100 µs to update the PFC Vout story array and update some global variables used to determine the input frequency.

3.1.7 u8 update_PFC_V(void)

This routine read the PFC Vout voltage and calculate the medium value every AC cycle. The calculated medium output voltage is used to identify the PFC output UVLO condition. The main make the appropriate action in case of output UVLO protection.

This routine is called every 100 µs.

3.1.8 void PFC_stop_PWM(void)

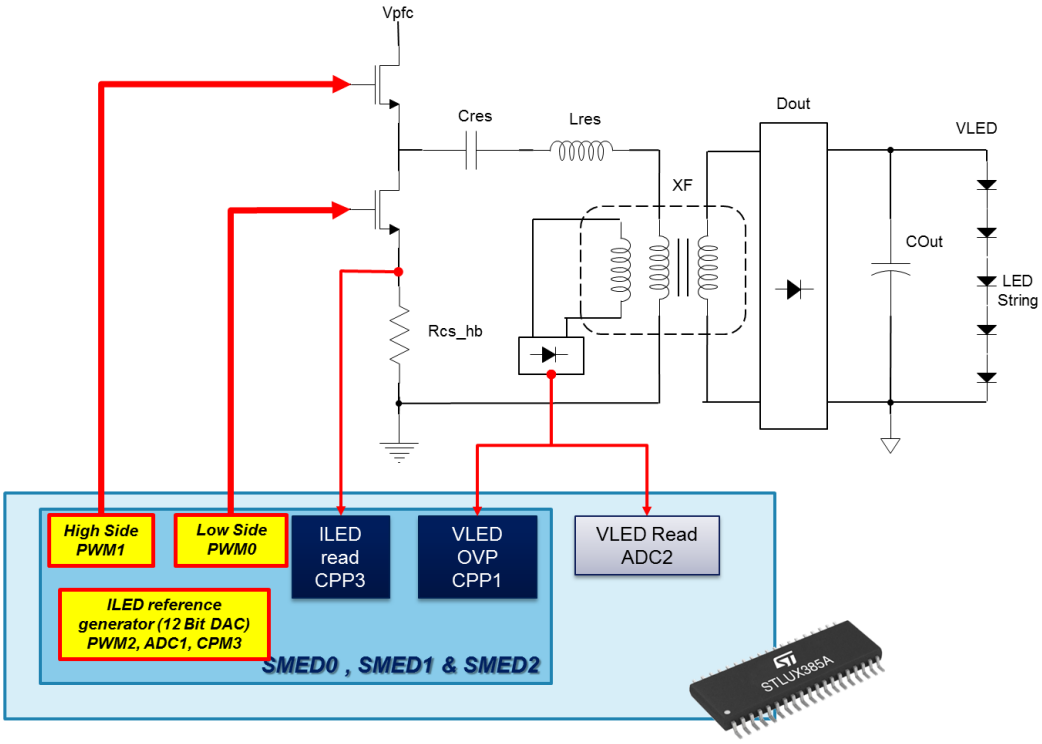
This subroutine stops the PFC module and the PWM output signal. This subroutine clear internal variable to stop the PFC activity. Immediately after this call the main call also the HB switch off routine to stop the HB activity otherwise the output current is out of control.

3.2 Half bridge library Interface

This section describes the functions implemented in the half bridge library to manage the two half bridge MOSFET gates.

The half bridge module is implemented using SMED0 for the low side and SMED1 for the high side. SMED2 is used to generate the analog reference voltage and SMED3 is used to manage the low current on the HB module.

Figure 10. ZVS concept

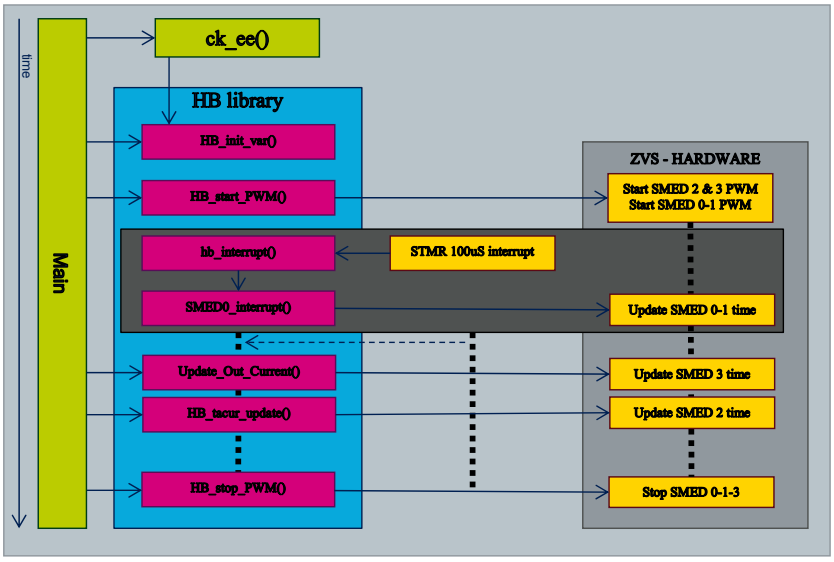


This module references the SMED0, the SMED1, the SMED2 and the SMED3 interrupt routines. One routine is called during the STMR timer interrupt to activate the HB current regulation function every 100 µs.

Figure 11. HB library concept offers a system overview on STLUX385A half bridge library implementation. It shows the most important call sequences of the HB library routine.

Two other modules are added in the HB library directory: one for CLO functionality and the other for Iout compensation reading the Vout. The CLO is enabled by default; the Iout compensation is disabled and needs to be changed in the GPGUI parameter file.

Figure 11. **HB library concept**



3.2.1 u8 HB_init_var(void)

The main calls this routine to initialize the half bridge internal variable. Some RAM variables are initialized through the `ck_ee` subroutine because they are stored in the EEPROM data area.

You should only call this function once during start-up.

The number returned by the subroutine identifies the HB library revision.

3.2.2 void hb_interrupt(void)

The STMR timer interrupt calls this routine to check and adjust of the output current.

Call this routine every 100 µs synchronous from the STMR timer. The execution time depends on the HB frequency.

When called, this routine enables the SMED0 interrupt time, if necessary.

3.2.3 INTERRUPT_HANDLER(SMED0_ISR, 6)

The SMED0 interrupt updates the SMED0 and SMED1 ON time, which changes the output current on the LED string. The correct time is acquired using the SMED “dump” facility.

3.2.4 void HB_start_PWM(void)

This function activates the half bridge PWM signal.

The start-up frequency of the half bridge module is defined as the maximum frequency, defined by parameter #31. This parameter represents the SMED cycles (expressed using a 96MHz SMED tick) during state 2 on SMED 0 and SMED1.

The start-up output current is defined by the parameter #4 and is applied after the HB start-up phases if DALI and 0-10 interfaces are disabled.

3.2.5 u8 check_S_CH0(void)

This subroutine acquires the input voltage on the ADC CH2 (S_CH0 signal) and checks when this voltage is lower or higher than parameter #33 in the configuration file. If the compare is higher, the subroutine returns a TRUE value, which indicates that the board output voltage (J4) is higher than the Open circuit detection point. In this case, the main code applies error #7 (ERR_OVP_CH0) and stops PFC and HB activity.

The subroutine is called every 100 µs, but requires a maximum of 400 µs to determine a new value of the S_CH0 signal.

3.2.6 void update_CPM_val(void)

This function acquires the value of ADC CH1 (CN_CNT) and stores the value in an accumulation variable used during CN_CNT voltage loop regulation.

This function is called every 100 µs and uses a few microseconds of processor time.

3.2.7 void HB_tacur_update(void)

This function regulates the CN_CNT voltage and consequently regulates the output current.

To regulate the `CN_CNT` voltage, the routine changes the SMED2 state 0 time through the `New_PWM2` subroutine. The function compensates the external tolerance (VCC, resistor and capacitor variation) using the voltage acquired on ADC CH1 using the `update_CPM_val` function.

The function is called every 400 µs.

The minimum output voltage is defined by the `HB_OL_LIM` limit (default 1150), so the minimum output voltage at the ADC-CH1 pin (`CN_CNT`) is 0.3512 V (typ.). The maximum theoretical output voltage at the ADC CH1 pin is 1.25 V, but also depends on the R3, R4, C8 and VCC values.

This function also sets up the ON time of the HB stage to support the required Iout value (specified in the `PWM2_on_time` variable) during the output current transition. The `hb_ont[]` table defines every intermediate point and the maximum Iout slope.

3.2.8 void HB_burst_mode(u16)

This function is used to define the SMED3 ON/OFF time.

The SMED3 defines the packet skipping mode on the HB PWM line. The parameters passed to the subroutines represent the level of the output current expressed in the equivalent `Iout_setup` value. The correct parameter range is from 1 to `HB_SM3_SM01` (default 1100). If the parameter passed to the subroutine is lower than `HB_MIN_PWM2` (default 539), the routine applies the lowest value. If it is higher than `HB_MIN_PWM2`, the routine does nothing.

3.2.9 void New_PWM2(B2W)

This function is used to change the value of PWM2 state 0 time.

This state timer defines the output off time of PWM2 and consequently defines the voltage on the CNT_CN signal. The minimum allowed value is 2, which represents the maximum output voltage on the CN_CNT signal. The maximum allowed value is 4094, which represents the minimum voltage on the CN_CNT signal.

3.2.10 void Update_Out_Current(void)

This routine regulates the output current when an open loop is forced. An open loop is defined as when the `Iout_setup` (output current) value is lower than the `HB_SM3_SM01` value. It is called every 800 µs and defines the rise and fall rate of output current during the open loop phases.

This routine controls SMED3 activity and regulates the HB frequency when the `HBZ_FIXFR` mode is active and other internal half bridge activities. To determine the appropriate action, it looks up certain internal HB variables.

This routine also updates `HB_zone` to identify which zone the HB is currently in. The user (using GPGUI, DALI or 0-10V) changes the `PWM2_on_time` variable to define the target current.

The routine enters the HBZ_CLOSE zone when the output current rises above the HB_OL_LIM value (default is 1150). When the output current is regulated using the HBZ_CLOSE zone, this routines does nothing because the output current is regulated by modifying the CN_CNT voltage through the HB_tacur_update routine.

The routine enters the HBZ_SKIPK zone when the output current is lower than HB_SM3_SM01 and higher than HB_SM3_LIM.

The middle values use the HBZ_FIXFR zone, which is not stable and is only used during the transition from HBZ_SKIPK to HB_OL_LIM and vice versa.

3.2.11 void HB_stop_PWM(void)

This routine takes the necessary action to STOP the half bridge PWM frequency and set up some internal variables for the following half bridge start.

3.2.12 void search_mm_HB_on(void)

This routine is used for debug purposes and stores the minimum and the maximum SMED0-S2 timer value.

This routine is called every 800 µs only.

4 Variable parameter descriptions

The following variable parameters are present in the STEVAL-ILL066V2 evaluation board library:

1. Board configuration parameters: disable or enable certain interfaces and board functions. For example, you can enable the DALI interface or enable the half bridge current loop control. These application level parameters can only be changed with the GPGUI program and are applied after a power-on reset.
2. PFC parameters: the PFC level parameters control some of the PFC activity, but not all, as some PFC parameters also depend on hardware values. For example, the PFC output voltage is defined by an external resistor value because the value of the PFC target voltage is fixed in the STLUX firmware.
3. Half bridge parameters: also in this case, some HB level parameters are defined by the external hardware.

The GPGUI program can be used to modify certain parameters.

Some PFC parameters are split into high, middle and low ranges that refer to the input Vac level. The value of these three PFC areas is specified through parameters #22, #23 and #24.

Table 2. STEVAL-ILL066V2 evaluation board library parameters

FW IDX	Level	Description	Default value	Unit	Range		Cross ref.
					Min	Max	
1	APP	PFC activation enable when 1	1	Bit (1)	0	1	Section 4.1 PFC activation enable
2	APP	HB activation enable when 1	1	Bit	0	1	Section 4.2 HB activation enable
3	APP	Dali interface enable when 1	1	Bit	0	1	Section 4.3 Dali interface enable
4	HB	Start-up output current level	539	Pure number	539	4093	Section 4.4 Startup output current level
5	n.u.	Reserved (2)					
6	APP	Half bridge current loop compensation enable when 1	1	Bit	0	1	Section 4.5 Half bridge current loop compensation
7	APP	Debug mode enable when 1	1	Bit	0	1	Section 4.6 Debug mode enable
8	APP	0-10 V interface enable when 1	0	Bit	0	1	Section 4.7 0-10 V interface enable
9	PFC	PFC THD maximum time	2000	1/96 MHz (3)	1	9600	Section 4.8 PFC THD maximum time
10	APP	Enable ROP code protection when 1	0	Bit	0	1	Section 4.9 Enable the ROP code protection
11	n.u.	Reserved					
12	PFC	DAC level (OCP) during emergency mode	7	DAC-bit (4)	4	12	Section 4.10 PFC DAC level (OCP)
13	PFC	DAC level (OCP) during high level mode	8	DAC-bit	5	13	
14	PFC	DAC level (OCP) during mid level mode	11	DAC-bit	6	14	
15	PFC	DAC level (OCP) during low level mode	13	DAC-bit	7	15	

FW IDX	Level	Description	Default value	Unit	Range		Cross ref.
					Min	Max	
16	PFC	Define the PFC activation level - high	503	ADC-bit (5)	181	1023	Section 4.11 PFC activation level
17	PFC	Define the PFC activation level - mid	407	ADC-bit	181	1023	
18	PFC	Define the PFC activation level - low	296	ADC-bit	181	1023	
19	PFC	Define the PFC input voltage stop level - high	445	ADC-bit	180	1023	Section 4.12 PFC input voltage stop level
20	PFC	Define the PFC input voltage stop level - mid	376	ADC-bit	180	1023	
21	PFC	Define the PFC input voltage stop level - low	267	ADC-bit	180	1023	
22	PFC	Define the PFC level entering into high zone	518	ADC-bit	180	1023	Section 4.13 PFC level to enter zones
23	PFC	Define the PFC level entering into mid zone	422	ADC-bit	180	1023	
24	PFC	Define the PFC level entering into low zone	251	ADC-bit	180	1023	
25	PFC	PFC UVLO on output level - high	329	ADC-bit	180	1023	Section 4.14 PFC UVLO on output level
26	PFC	PFC UVLO on output level - mid	289	ADC-bit	180	1023	
27	PFC	PFC UVLO on output level - low	189	ADC-bit	180	1023	
28	PFC	PFC Start On time	70	1/96MHz	10	2047	Section 4.15 PFC On time during start-up
29	PFC	PFC Start Off time	1	1/96MHz	1	336	Section 4.16 Off time during PFC start
30	PFC	Delay to detect input Vac missing	100	ms (6)	2	255	5.17
31	HB	Maximum available HB frequency - SMED0&1-S2	61	1/96MHz	61	260	Section 4.18 Maximum available HB frequency
32	HB	Minimum available HB frequency - SMED0&1-S2	950	1/96MHz	260	1120	Section 4.19 Minimum available HB frequency
33	HB	No load level protection - ADC CH2	818	ADC-Bit	736	1023	Section 4.20 HB No load level protection
34	HB	Propagation delay mismatch compensation	30	1/96MHz	0	100	Section 4.21 Propagation delay mismatch compensation
35	HB	Trimming HIGH side delay	4	1/96MHz	0	20	Section 4.22 Trimming high side delay
36	HB	HB dead time	33	1/96MHz	24	192	Section 4.23 Dead time delay
37	n.u.	Reserved					
38	PFC	Δ error to entering into LOOP OK - high zone	7	Pure number	0	100	Section 4.24 PFC error to LOOP OK
39	PFC	Δ error to entering into LOOP OK - mid zone	10		0	100	
40	PFC	Δ error to entering into LOOP OK - low zone	16		0	100	
41	HB	When true activate the fast output current change	1	Bit	0	1	Section 4.25 Fast HB current regulation
42	PFC	PFC Integral Gain	140	Number	1	254	Section 4.26 PFC Integral Gain

FW IDX	Level	Description	Default value	Unit	Range		Cross ref.
					Min	Max	
43	PFC	PFC Integral Gain Scaler	15	Number	1	24	Section 4.27 PFC Integral Gain Scaler
44	PFC	PFC Proportional Gain	130	Number	1	254	Section 4.28 PFC Proportional Gain
45	PFC	PFC Proportional Gain Scaler	13	Number	1	24	Section 4.29 PFC Proportional Gain Scaler
46	PFC	PFC Over Voltage high speed level enable	1023	ADC-Bit	920	1023	Section 4.30 PFC Over Voltage high speed level enable
47	PFC	PFC Under Voltage high speed level enable	850	ADC-Bit	600	900	Section 4.31 PFC Under Voltage high speed level enable
48	PFC	PFC under voltage high speed gain	500	Number	1	10000	Section 4.32 PFC under voltage high speed gain
49	HB	Level to detect the short on the output	100	ADC-bit	1	1023	Section 4.33 Level to detect the short on the output
50	TEMP	Temperature to shut down the application	900	THER	400	1200	Section 4.34 Temperature to shut down the application
51	TEMP	Temperature to start the power reduction	750	THER	300	1200	Section 4.35 Temperature to start power reduction
52	TEMP	`Iout_setup` slope reduction	100	THER	1	1000	Section 4.36 Iout_setup slope reduction

1. *A bit is a two-state value: 1 activates the function, 0 disables it*
2. *Reserved parameters are used only for backwards compatibility with older source code and are not modifiable by the user.*
3. *This unit represents the clock cycle of the SMED. For SMED0, SMED1 SMED4 and SMED5, this time is 10.416666 nS (or 1/96 MHz).*
4. *This unit is used with the STLUX DAC peripherals. The value defined with this unit is copied into the relative DAC register when necessary.*
5. *This unit is the value read from the ADC register. To collect the voltage at the STLUX pin, multiply the value by 1.221896 mV (1.25 V/1023).*
6. *This 1-ms time unit time specifies the delay before an action commences.*

4.1 PFC activation enable

This parameter signals whether the PFC firmware module is enabled (1, default) or disabled (0). It is used during the hardware debug phases to verify the external circuit and the external resistor partition.

This bit is the GPGUI `2 - PFC enable` bit in the STEVAL-ILL066V2 board control field parameter.

4.2 HB activation enable

This parameter signals whether the HB firmware module is enabled (1, default) or disabled (0). It is used during the hardware debug phases to verify the external circuit and the external resistor partition.

This bit is the GPGUI `4 - Half bridge enable` bit in the STEVAL-ILL066V2 board control field parameter.

4.3 Dali interface enable

This parameter identifies whether the DALI interface module is enabled (1, default) or is disabled (when 0). It is used during the hardware debug phases to verify the external circuit and the external resistor partition.

Note: To enable the 0-10 interfaces (#8), it is necessary to disable the DALI interface because these two interfaces share some STLUX internal resources.

This bit is the GPGUI `8 - DALI interface enable` bit in the STEVAL-ILL066V2 board control field parameter.

4.4 Startup output current level

This parameter identifies the output current when the evaluation board is powered. This value is only used when the DALI and 0-10V interfaces are disabled. It is the same as the value used during the `Iout_setup` command.

The minimum value is 539 (minimum output current) and the maximum is 4093 (maximum output current).

This parameter is also the GPGUI `Iout start when no I/F` field.

4.5 Half bridge current loop compensation

This parameter signals whether the automatic current control loop is active on the HB module. If the current loop compensation is disabled, the HB frequency is fixed.

To change the output current (by changing the HB frequency) when the current loop compensation is disabled, use the `hb` serial command.

This bit is the GPGUI `20 - HB closed loop control` bit in the STEVAL-ILL066V2 board control field parameter.

4.6 Debug mode enable

This parameter signals whether the board is active or totally disabled. When this parameter is 1, all board activity is frozen.

This bit is the GPGUI `1 - Debug mode active` bit in the STEVAL-ILL066V2 board control field parameter.

4.7 0-10 V interface enable

This parameter signals whether the 0-10V interfaces are active and control the output current. This interface is automatically disabled when the DALI interface is enabled.

To enable the 0-10 V interface, disable the DALI interface. The default value of this parameter is 0 (disabled).

This bit is the GPGUI `10 - 0-10 interface enable` bit in the STEVAL-ILL066V2 board control field parameter.

4.8 PFC THD maximum time

This parameter identifies the maximum numbers of clock cycles (ticks) to wait for the PFC peak current to reach a predefined level.

This number is expressed in clock cycles (1/96 MHz) and is applied to SMED5 state 2. This wait time is applied outside the PFC start-up period and outside the PFC light output power.

The available range for this parameter ranges from 1 (no THD optimization) to 9600 ticks (wait for a maximum of 100 µs for the minimum current). During this time, the PFC gate driver is enabled and the MOSFET is closed. (Refer to the STLUX Reference Manual and the SMED configuration program for more information on SMED activity).

The PFC minimum peak current on the STEVAL-ILL066V2 evaluation board is 206 mA (typ.) and is defined by the U18 comparator and the R100, R101 partition.

This parameter is the GPGUI `PFC THD optimization time` register.

4.9 Enable the ROP code protection

This parameter signals whether the program code is protected from being downloaded or any modification of configurable parameters. When this parameters is 1, all the commands used to modify the parameters and code download via SWIM are disabled.

Note: The only way to remove this setting is to wipe the entire chip, including the configuration file.

This bit is the GPGUI `4 - Enable ROP register` bit in the STEVAL-ILL066V2 board setup field parameter.

4.10 PFC DAC level (OCP)

This parameter identifies the Over Current Protection level. This parameter is used to setup the STLUX DAC level used for the PFC stage. There is the possibility to make four level setting. The first level is defined using an internal behavior define “emergency” and is triggered when the PFC module detect an immediate PFC stage protection event. The other three levels define three different current protection level when the input AC voltages is into a predefined level. All the values of these parameters are referred to the internal DAC value. The protection voltages step is 0.082V and the PFC current protection depends from the PFC shunt resistor value. The SMED5 triggered an interrupt event when the OCP event arrives.

This parameter are the GPGUI “DAC level during emergency”, “DAC level during high Vac”, “DAC level during middle Vac” and “DAC level during low Vac” registers.

4.11 PFC activation level

This parameter is the PFC input voltage at the STLUX pin for PFC module activation. When the PFC output voltage is below this parameter value, the PFC is not started (but will remain active if it is already running). These levels depend on the input Vac zone defined in parameters #25, #26 and #27. The only significant parameter is #18 - the real minimum start-up level. The value defined here is the ADC conversion value.

To obtain the equivalent input voltage at the STLUX pin, multiply the parameter value by 1.221896 mV. To obtain the equivalent input voltage at the V_IN signal, see the external resistor partition.

To obtain the correct V_IN voltage on the STEVAL-ILL066V2 evaluation board, divide the parameters in this field by 2.2259.

This parameter is the GPGUI `Vac start-up level` register.

4.12 PFC input voltage stop level

These parameters identify the PFC input level to stop the PFC modules. The PFC module controls the V_IN signal and, when it remains below these parameters for a minimum of 80 ms, the PFC module stops PFC activity and enters in emergency mode.

These levels also depend on the input Vac level defined in parameters #25, #26 and #27.

To obtain the equivalent input voltage at the STLUX pin, multiply the parameter value by 1.221896 mV. To obtain the equivalent input voltage at the V_IN signal, see the external resistor partition.

To obtain the correct V_IN voltage on the STEVAL-ILL066V2 evaluation board, divide the parameters in this field by 2.2259.

This parameter is the GPGUI `Vac UVLO voltage` register.

4.13 PFC level to enter zones

This parameter identifies the three zones of the total input voltage (V_IN signal).

If the AC cycle input voltage is higher than the high zone, the PFC module enters the high zone. If the input voltage is lower than high zone, it check whether it is higher than the middle zone to enter the middle zone. If the input voltage is lower than middle zone, it checks whether it is higher than the low zone to enter the low zone.

To obtain the equivalent input voltage at the STLUX pin, multiply the parameter value by 1.221896 mV. To obtain the equivalent input voltage at the V_IN signal, see the external resistor partition.

To obtain the correct V_IN voltage on the STEVAL-ILL066V2 evaluation board, divide the parameters in this field by 2.2259.

These parameters are the GPGUI `PFC start high range zone`, `PFC start middle range zone` and `PFC start low range zone` registers.

4.14 PFC UVLO on output level

This parameter identifies the level of UVLO protection on the PFC output voltage.

The PFC output voltage is continuously acquired and compared with the value defined in these parameters. When the PFC output voltage is less than the value defined in this parameter, the PFC module stops activity.

To obtain the equivalent input voltage at the STLUX pin, multiply the parameter value by 1.221896 mV. To obtain the equivalent input voltage at the PFC_OUT signal, see the external resistor partition.

To obtain the correct PFC_OUT voltage on the STEVAL-ILL066V2 evaluation board, divide the parameters in this field by 2.2259.

These parameters are the GPGUI `PFC output UVLO high range zone`, `PFC output UVLO middle range zone` and `PFC output UVLO low range zone` registers.

4.15 PFC On time during start-up

This parameter identifies the SMED5-S3 counter value used during the start-up of the PFC module.

This parameter depends primarily on the PFC inductor value, from the Vac input voltage and the load when the PFC starts. These parameters are expressed in the SMED clock cycle (1/96 MHz) and define the value of the SMED5-S3 (state 3) time during start-up of the PFC. Basically, these parameters define the PFC output voltage ramp-up time. The SMED5-S3 values are automatically changed when the PFC output voltage reaches near the target value.

To calculate the real PFC on time during the PFC ramp-up, add the SMED5-S1 (fixed to 40), the SMED5-S2 (during ramp-up fixed to 1) and the value of this parameter. For example, the default value of this parameter on the STEVAL-ILL066V2 evaluation board is: 40+1+70 SMED clock cycles, or 1.15 µs.

Before the PFC starts, there is a 30 ms delay to acquire the Vac input voltage to determine the applied input AC level. To learn more about the SMED behavior in the PFC stage, monitor the SMED4 and SMED5 evolution using the SMED configurator.

This parameter is the GPGUI `PFC Controlled ON time used during start-up` register.

4.16 Off time during PFC start

This parameter identifies the minimum PFC OFF time before searching for the ZCD event. To learn more, see the PFC (SMED4 and SMED5) evolution using the SMED configurator.

These parameters are expressed in SMED clock cycles (1/96 MHz) and define the minimum value of the SMED4 State 0 time.

This parameter is the GPGUI `PFC Controlled OFF time used during start` register.

4.17 Delay to detect PFC input Vac missing

This parameter identifies the minimum time to detect the missing Vac input voltage.

The minimum time to detect this event is a one-Vac rectified cycle (10 ms for 50 Hz). This time cannot be lower because the firmware in the STLUX verifies the Vac peak to peak voltage once every Vac cycle.

The unit of this parameter is in milliseconds.

This parameter is the GPGUI `Time (ms) to detect AC missing` register.

4.18 Maximum available HB frequency

This parameter identifies the maximum available frequency used during HB current regulation. These numbers identify the maximum frequency of the half bridge stage.(Refer to the STEVAL-ILL066V2 User Manual to learn more about ZVS topology).

This parameter is expressed in SMED clock cycles (1/96 MHz).

This time represents the minimum number loaded in SMED0 and SMED1 state 2. The maximum time also defines the output current when the Iout_setup field is set to a raw value of 1100. To calculate the HB frequency, this parameter is added to the Dead Time plus the LEB time plus the HOLD transaction (2.5 + 2.5 cycles) multiplied by 2 SMED used for the HB stage.

On the STEVAL-ILL066V2 evaluation board, this time is:

(((33+24+61)*2)+5)*(1/96) = 241*10.41 ns, which equals approximately 398 KHz.

This parameter is the GPGUI `Minimum HB on time` register.

4.19 Minimum available HB frequency

This parameter identifies the minimum available frequency used during HB current regulation. These numbers identify the minimum frequency of the half bridge stage and consequently the maximum output current.

This parameter is expressed in SMED clock cycles (1/96 MHz).

This time defines the maximum number loaded in SMED0 and SMED1 state 2. On the STEVAL-ILL066V2 evaluation board, this time is:

(((33+24+950)*2)+5)*(1/96) = 1619*10.41 ns, which equals approximately 47 KHz.

This minimum frequency depends on the load, output transformer characteristics and HB supply voltage.

This parameter is the GPGUI `Maximum HB on time` register.

4.20 HB No load level protection

This parameter determines the ADC value which defines the no load detection point during half bridge operation. On the STEVAL-ILL066V2 evaluation board, this value is approximately 1 V.

To find the real voltage on the output connector, you must also determine the T2 transformer ratio and the R92-R28-R29 resistor ratio. On the STEVAL-ILL066V2 evaluation board, the T2 transformer ratio is 5.2 and the shunt resistor ratio is about 18.43.

The maximum default output voltage on the STEVAL-ILL066V2 evaluation board at the J4 connector pin is approximately 100.4 V.

This parameter is the GPGUI `Output voltage to detect no load` register.

4.21 Propagation delay mismatch compensation

This parameter identifies the propagation delay mismatch compensation due to the external circuit (HB driver) delay, the external RC filter and STLUX internal propagation delay. This parameter is expressed in SMED clock cycles (1/96 MHz) and the default value for the STEVAL-ILL066V2 evaluation board is 30 SMED cycles, or 312.5 ns.

This parameter is the GPGUI `HB propagation delay time` register.

4.22 Trimming high side delay

This parameter identifies the time difference from the low side to high side on the HB stage.

On the STEVAL-ILL066V2 evaluation board, the circuit delay is about 41 ns.

The difference is due to the external driver delay difference from the low side to the high side. This parameter is expressed in SMED clock cycles (1/96 MHz). This time is added to the HB high side signal to compensate the external difference time. The default value on STEVAL-ILL066V2 evaluation board is 4, or 42 ns.

This parameter is the GPGUI `HB high side symmetry time` register.

4.23 Dead time delay

This parameter defines the dead time on the half bridge stage. The time is applied on the low side and on the high side stage together.

On the STEVAL-ILL066V2 evaluation board, the default value is 343.75 ns (or 33 SMED cycles) and the valid range is from 250 ns to 2 µs (or 24 to 192 SMED cycles).

Note: The external drive (L6388E) forces the minimum time due to an internal minimum dead time (typ. 320 ns).

This parameter is the GPGUI `HB dead time` register.

4.24 PFC error to LOOP OK

These parameters are the numbers of errors to enable the PFC_B_LPOK flag. There are three zones defined by the Vac input value.

This parameter is not inserted in the GPGUI register.

4.25 Fast HB current regulation

When this flag is true (default), it enables the fast Iout regulation and the typical switch on time (10 mA to 1 A output) is 150 ms typical. If the DALI interface is used, the DALI stack regulates the fade time.

When the flag is false (equal to 0) the minimum fade time managed by the evaluation board is 333 mA/s.

The use of the bit equal to 0 is deprecated because some DALI requests are not satisfied.

This parameter is not inserted in the GPGUI register.

4.26 PFC Integral Gain

This parameter identifies the Integral Gain used in the PFC PI algorithm. The default value applied on the STEVAL-ILL066V2 evaluation board is 140 and is the multiplication factor applied to the error accumulated during the last AC cycle.

The sampling rate of the ADC is 100 µs.

Refer to Section 2.1 PFC algorithm for more information regarding the PFC PI algorithm.

This parameter is the GPGUI `PFC Integral Gain` register.

4.27 PFC Integral Gain Scaler

This parameter identifies the Integral Gain scaler used in the PFC PI algorithm. The default value applied on the STEVAL-ILL066V2 evaluation board is 15 and is the exponent applied to the error accumulated during the last AC cycle.

The sampling rate of the ADC is 100µs.

Refer to Section 2.1 PFC algorithm for more information regarding the PFC PI algorithm.

This parameter is the GPGUI `PFC Integral divisor` register.

4.28 PFC Proportional Gain

This parameter identifies the Proportional Gain used in the PFC PI algorithm. The default value applied on the STEVAL-ILL066V2 evaluation board is 130 and is the multiplication factory applied to the error accumulated during the last AC cycle.

The sampling rate of the ADC is 100 µs.

Refer to Section 2.1 PFC algorithm for more information regarding the PFC PI algorithm.

This parameter is the GPGUI `PFC Proportional divisor` register.

4.29 PFC Proportional Gain Scaler

This parameter identifies the Proportional Gain scaler used in the PFC PI algorithm. The default value applied on the STEVAL-ILL066V2 evaluation board is 13 and is the exponent applied to the error accumulated during the last AC cycle.

The sampling rate of the ADC is 100 µs.

Refer to Section 2.1 PFC algorithm for more information regarding the PFC PI algorithm.

4.30 PFC Over Voltage high speed level enable

This parameter defines the PFC high threshold where the high-speed gain is enabled. When the PFC output voltage is more than this value, the PI bandwidth is increased to accelerate the time to arrive to the target. When the PFC output voltage is less than this value, the PI bandwidth is the default value.

To obtain the equivalent input voltage at the STLUX pin, multiply the parameter value by 1.221 mV (1.25 V/1023).

This parameter is the GPGUI `High level to trig PFC fast reaction` register.

4.31 PFC Under Voltage high speed level enable

This parameter defines the PFC low level where the high-speed gain is enabled. When the PFC output voltage is less than this value, the PI bandwidth is increased to accelerate the time to arrive to the target. When the PFC output voltage is more than this value, the PI bandwidth is the default value.

To obtain the equivalent input voltage at the STLUX pin, multiply the parameter value by 1.221896 mV (1.25 V/1023).

This parameter is the GPGUI `Low level to trig PFC fast reaction` register.

4.32 PFC under voltage high speed gain

This parameter defines the slope of the convergence correction when the voltage is lower than the value specified in parameter #47. A larger number sets a lower time.

This parameter is the GPGUI `PFC fast reaction gain` register.

4.33 Level to detect the short on the output

This parameters defines the level of the output voltage (read on pin 36 [ADCIN2]) to identify a short circuit on the output (J4). When the output is below this level, the FW assumes a short circuit on the output. The number is the ADC value acquired on the input pin.

This parameter is the GPGUI `Vout to detect short circuit` register.

4.34 Temperature to shut down the application

This parameter sets the temperature trigger point to switch off the board. The value is expressed in degrees centigrade. The default value of 900 equates to 90.0 °C.

This parameter is the GPGUI `Temperature to shut down` register.

4.35 Temperature to start power reduction

This parameter sets the temperature trigger point to start power reduction for overtemperature. The value is expressed in degrees centigrade. The default value of 750 equates to 75.0 °C.

The power reduction slope is defined using the Iout_setup slope reduction parameter file.

This parameter is the GPGUI `Temperature to trigger power reduction` register.

4.36 Iout_setup slope reduction

This parameter sets the downward output current slope when the temperature trigger point is above Temperature to start power reduction parameter.

The Iout_setup slope determines the Iout_setup value reduction for every degree Celsius starting from the Temperature to start power reduction parameter level.

This parameter is the GPGUI `ll slope when power reduction` register.

5 Fixed parameters

The values of fixed parameters defined in the PFC or HB library are set in the STLUX when external circuits (typically a resistor partition) define the level of the controlled variable.

The following table shows the fixed values for the STEVAL-ILL066V2 evaluation board relating to the PFC and half bridge library.

Table 3. Fixed parameters table

HW IDX	Types	Description	Default value	Unit	Circuit	Cross ref.
1	PFC	Target PFC output voltage	1.1151	V	R46-R52-R55-R58	Section 5.1 Target PFC output voltage
2	PFC	PFC input voltage rectified @220Vac or 311Vdc	0.8462	V	Fixed on STLUX	Section 5.2 PFC input voltage rectified
3	PFC	PFC output - second OVP level	1.250	V	Fixed on STLUX	Section 5.3 PFC output - second OVP level
4	HB	HB time increment during time stamp	30	1/96MHz	Fixed on STLUX	Section 5.4 HB time increment during time stamp
5	HB	HB control voltage @ 1A	1.0539	V	R36-R37	Section 5.5 HB control voltage
5	APP	ADC input voltage on ADC[5] (pin 33) @ 14V	1.0370	V	R96-R97	Section 5.6 ADC input voltage on ADC[5]
6	APP	ADC input voltage on ADC[6] (pin 32) @ 3V3	1.1647	V	R98-R99	Section 5.7 ADC input voltage on ADC[6]

The default voltage value expressed in the above table relates to the STLUX input pin.

5.1 Target PFC output voltage

This fixed parameter identifies the target PFC output voltage applied to the STLUX pin. It is fixed on the STLUX pin because an external resistor defines the real PFC output voltage. On the STEVAL-ILL066V2 evaluation board, this level is fixed at 1.1151 V and is used as a reference for the PFC output voltage.

For the STEVAL-ILL066V2 evaluation board, the PFC output voltage is around 410 V.

The R46, R52, R55 total value equals 6M6Ω and the R58 is fixed to 18 KΩ. The internal ADC conversion value is 912, which corresponds to 1.1151 V on the STLUX pin. If you change the R46, R52, R55 resistance value, you must limit the total resistor partition to a maximum of 10 MΩ.

The total capacitance present on this pin also limits the bandwidths of the ADC acquisition. The actual internal ADC bandwidths is fixed at 10 KHz for the STEVAL-ILL066V2 evaluation board (one ADC acquisition every 100 µs).

5.2 PFC input voltage rectified

This fixed parameter identifies the PFC input voltage. The resistor partition on this STLUX pin is the same as the PFC output resistor partition.

The STEVAL-ILL066V2 evaluation board reads 0.8462 V (at STLUX pin) when 220 Vac peak is applied on the Vac input pin.

5.3 PFC output - second OVP level

This parameter identifies the voltage which defines the second PFC output voltage triggering level protection. The second PFC output voltage protection is implemented directly in the SMED function without any firmware activity.

This is the last resource to limit the PFC output voltage. For the STEVAL-ILL066V2 evaluation board, it is fixed at 452 V (1.23 V on the STLUX pin).

5.4 HB time increment during time stamp

This parameter identifies the SMED cycle increment value when a new time stamp is started. This time also defines the maximum Δi/Δt variation during the output current change.

5.5 HB control voltage

This parameter identifies the voltage present on the LLC_CN signal (and on the CN_CNT signal) when the output current is 1 A. This fixed voltage is the peak current level on the R36//R37 when the output current on LED string is 1 A.

This parameter depends on the T2 inductor specification, T1 inductance value, output current, output voltage, output rectified diode and frequency of the HB stage.

The LLC_CN signal is the low side current acquired from the HB sensing resistor and is compared with the target value define by the CN_CNT level. The default value for 1 A Iout is 1.0539 V. The minimum value (due to noise rejection) is 0.351 V (Iout about 400 mA). Below this point, the output current is regulated using HBZ_SKIPK mode.

You can change the R36//R37 shunt resistor when the output current is different to the STEVAL-ILL066V2 evaluation board. The value for the LLC_CN pin is 1.0539 V when the output current is at the target level.

5.6 ADC input voltage on ADC[5]

This parameter identifies the level of the ADC[5] (pin 33 on STLUX385A) when the VD_14 signal is 14 V. This value is acquired to monitor the VD_14 signal.

The limit on this pin is 11 V minimum or 15 V maximum.

5.7 ADC input voltage on ADC[6]

This parameter identifies the level of the ADC[6] (pin 32 on STLUX385A) when the VD_3V3 signal is 3.3 V. This value is acquired to monitor the VD_3V3 signal.

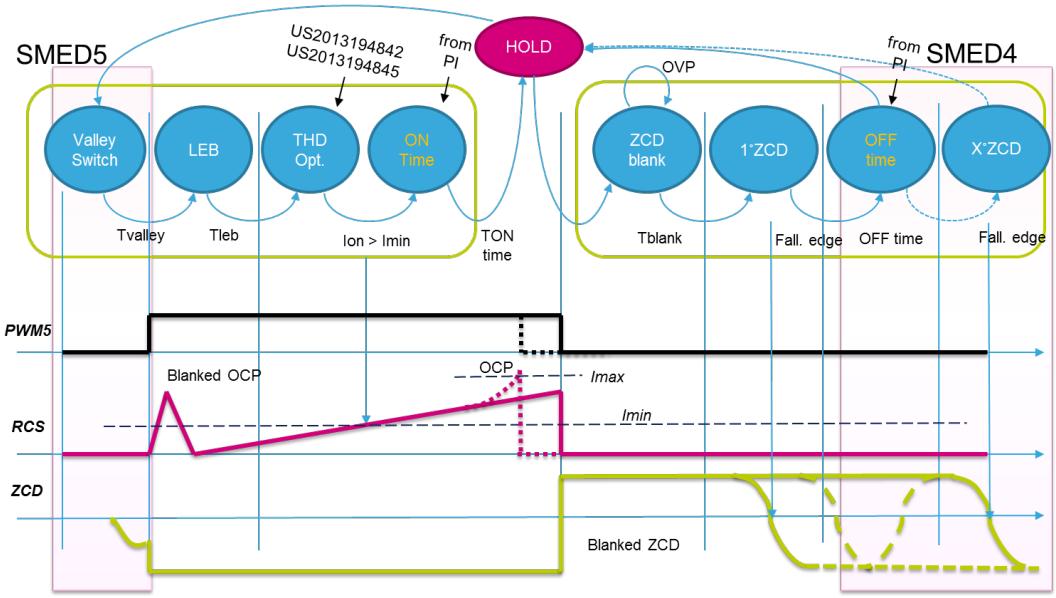
6 PFC check procedure

Note: This procedure applies a dangerously high voltage on the DUT and the DUT itself creates high internal voltages (in some circumstance). Please use caution and do not touch the board (even when no power is applied). Only qualified persons should perform the test procedure.

6.1 Review of SMED states

To be able to test PFC functionality, you need an understanding of the SMED peripherals.

Figure 12. STLUX PFC - SMED state machine



In the previous figure, the PFC PI regulation is only based on ON time and OFF time.

- The PFC maximum OFF time allowed in the PI regulation is 65520 SMED ticks at 96 MHz
- The PFC maximum ON time allowed in the PI regulation is 1800 SMED ticks at 96 MHz

6.2 Conditions prior to testing

Ensure the following conditions are met before you start:

1. Ensure the board does not have any hardware issues on the PFC stage (PFC_ZCD and PFC_MIN signals working correctly, the PFC stage circuit is correct)
2. Compile and load a new FW package with serial command enables (USE_SERIAL_DEB 1 and USE_DEBUG_BOARD 1 in the led.h file). This disables the GPGUI protocol and restores the command line environment. Please see the STEVAL-ILL066V1 User manual for more information regarding the messages described below
3. The serial line is available and working
4. The PFC inductor characteristic is known and matches the design requirements
5. These serial commands are sent to the DUT:
   a. in 1 0 → to disable the PFC stage
   b. in 2 0 → to disable the HB stage
   c. in 3 0 → to disable the DALI interfaces
   d. in 6 0 → to disable the HB HBZ_CLOSE mode

These serial line commands ensure the STLUX does nothing. PFC and HB functionality remain in the STOP state and there is no activity on the respective output PWM pins. The input signal is also monitored and can be displayed via a serial command.

6.3 Recommended test procedure

This first phase involves a simple check to determine whether all the necessary input analog pins are connected and working correctly.

Follow the procedure below to test PFC functionality (after having performed the necessary safety checks for working with high voltages that are not described here).

Procedure

Step 1. Connect an electronic load to the PFC output.

Use an electronic load in CCH mode and setup the load to 10 mA – 500 Vdc.

Step 2. Connect a high voltage probe to the PFC output voltage.

Step 3. Set the power generator to 220 V - 50 Hz and connect to the AC input of the DUT.

Step 4. Turn the power on.

Step 5. View the DUT output strings through Hyper Terminal.

Check that the PFC output voltage is in line with the value defined by design. Check the output against the strings highlighted in bold text below.

- `Option` returns the active option, but as some `in` commands were used during setup, no functions are active.
- `frq` line returns the number of 100 µs time units in an input AC half-cycle.

```
STEVAL385LEDPSR V3R35 03:02
Build Apr  1 2015 08:07:16 core is 385A ok
Option R enabled
Ready
frq=99
```

Step 6. Enter the command `st` to view the STEVAL-ILL066V2 board status and check the output against the strings highlighted in bold text below.

The values in bold are subject to a tolerance range of about 5 to 10 units, depending on the AC output voltage and the components mounted on the DUT.

- `Vp` is the Vac top voltage acquired on the STLUX pin
- `st` is the PFC status: 0 → PFC not active.

```
stPSR-ZVS status is 0:0
Vin Vi=689 Vp=693 Vmin=689 Vmax=692 freq=100
PFC st=0x0 Vo=689 S0=1 S3=70 I=8
HB st=0x0 Z=2 S0=33 S3=61 dt=30 O0 L=9 H=9 CH0=0
PW2 ta=0:0 re=11 now=539:16507 O1:0 dump=0 ndump=0
```

Step 7. Enter the command `gp` to return the PFC status and check the output against the strings highlighted in bold text below.

- `Vo` is the PFC output voltage acquired on STLUX pin; this value should be same as the previous `Vp` (minus the diode Vf)

```
gp
PFC
f=0x0 S0=1 S3=70 e=132 Vo=689 ta=912 pw=0 Vol=691 is=1
on=70:70 off=1:1 s_t=3 ZCD=104 L=5988 fl=17
L0=:224:224:224:223:224:224
```

Step 8. Enter the command `in 1 1` to enable the PFC.

Step 9. Power off the DUT and wait for the PFC output capacitor to discharge completely.

Step 10. Power on the DUT and monitor the PFC output voltage as it ramps up using a high voltage probe connected to the PFC output voltage.

Proceed with caution as this is the first step that involves STLUX PFC functionality.

Step 11. Check the output against the strings highlighted in bold text below.

– `frq=99` represents the number of 100 µs time units in an input AC half-cycle.

```
STEVAL385LEDPSR V3R35b9 03:02
Build Apr  1 2015 08:07:16 core is 385A ok
Option PFC R enabled
Readyfrq=99
```

Step 12. Enter the command `st` and check the output against the strings highlighted in bold text below.

The values in bold are subject to a tolerance range of about 5 units, depending on the AC output voltage and the components mounted on the DUT.

– `S0` indicates the OFF time used in the last AC cycles and is a function of PFC input voltage, PFC inductor and PFC load.

```
st
PSR-ZVS status is 0:0
Vin Vi=150 Vp=693 Vmin=127 Vmax=690 freq=100
PFC st=0x41 Vo=909 S0=1688 S3=1 I=8
HB st=0x0 Z=2 S0=33 S3=61 dt=30 O0 L=9 H=9 CH0=0
PW2 ta=0:0 re=11 now=539:16507 O1:0 dump=0 ndump=0
```

Step 13. Enter the command `gp` command and check the output against the strings highlighted in bold text below.

– `Vo` defines the PFC output voltage during the last AC cycles. The value returned is subject to a tolerance of about ±5 units.

– `on` represents the PFC ON time used during the PFC regulation algorithm. When the load is the minimum (10mA on electronics load), this value must be 1 (the minimum controlled on time).

– `off` represents the PFC OFF time used to regulate the PFC output voltage. The value depends on the AC input voltage, the PFC inductor value used on the DUT and the load on the PFC output. Using this value, it is also possible to verify the inductor tolerance during board production if the other parameters remain constant. The two numbers define the minimum-maximum OFF time during the last half AC cycle.

– `L0` represents the PFC error during the last 153.6 ms. If the six numbers are the same, the PFC output voltage is stable. A tolerance of one or two digits is possible.

```
gp
PFC
f=0x0 S0=2280 S3=1 e=164 Vo=909 ta=912 pw=0 Vol=911 is=1
on=1:1 off=2280:2280 s_t=0 ZCD=104 L=5988 fl=0
L0=:4:4:4:4:4:4
```

Step 14. Increase or decrease the electronic load to the target load defined by design.

Step 15. Monitor the PFC output voltage to verify whether the PFC output voltage remains at the target voltage.

Step 16. Check the signals ZCD, MIN_CUR, MOSFET ON TIME stability, MOSFET source current and PFC output voltage during load variation.

Step 17. Power off the DUT when you have finished testing.

7 Half bridge fine tuning

We suggest that you follow a specific HB trimming sequence when you debug your own hardware board.

The procedure involves changing the HB R_sensing resistor or the parameter file.

You should only perform this test when you are certain there are no hardware issues and the PFC is functioning properly.

You can use Figure 7. Half bridge signal relations for reference.

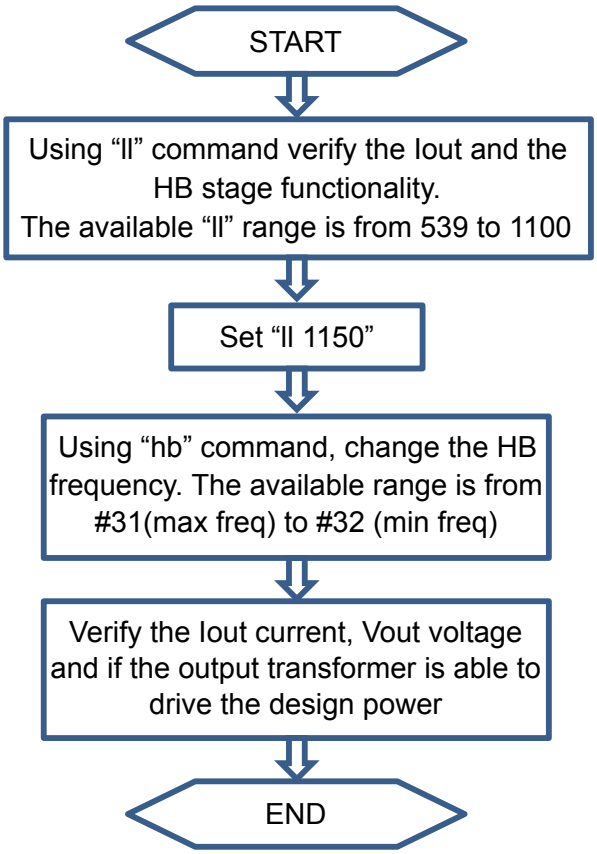
7.1 Check the half bridge output power

The first thing you need to do is test half bridge output performance by modifying the HB stage behavior to work in open loop mode, set through parameter file #6. When parameter file #6 is set to 0, the output current is not defined by the value `ll`, but from the HB frequency (only when `ll` is greater than 1150).

Ensure the following conditions are met before you start:

1. The board used does not have any open hardware issues (especially on PFC and HB stages)
2. Compile and load a new FW revision with serial command enable (USE_SERIAL_DEB 1 and USE_DEBUG_BOARD). This disables the GPGUI protocol and restores the command line environment.
3. The serial line is available and working
4. The PFC is working and stable in all output power ranges
5. The PFC Vout matches the PFC Vout design target
6. Board Vin is set to the fixed voltage (e.g., 220 Vac)
7. Vout is at nominal LED string voltage (maximum allowed)
8. The load is a real LED (don’t use an electronic load to trim the HB stage)
9. The HB parameters #34, #35 and #36 are controlled and adapted to the HW
10. The HB control loop is disabled (#6 = 0) but functions correctly and Iout is stable (i.e., no hardware issues)

Figure 13. **HB output power check procedure**



- `ll 539` sets the HB stage to force one HB switching cycle every 112 HB switching cycles; this drives the minimum Iout current
- `ll 1100` sets the HB stage to force all the 112 HB switching cycles, this forces maximum current for HBZ_SKIPK mode
- Parameter #31 defines the output current when `ll` equals 1100
- The real minimum HB frequency requires a margin (about 20%) to compensate for an eventual PFC voltage drop
- The real minimum current is defined during the fine-tuning procedure.

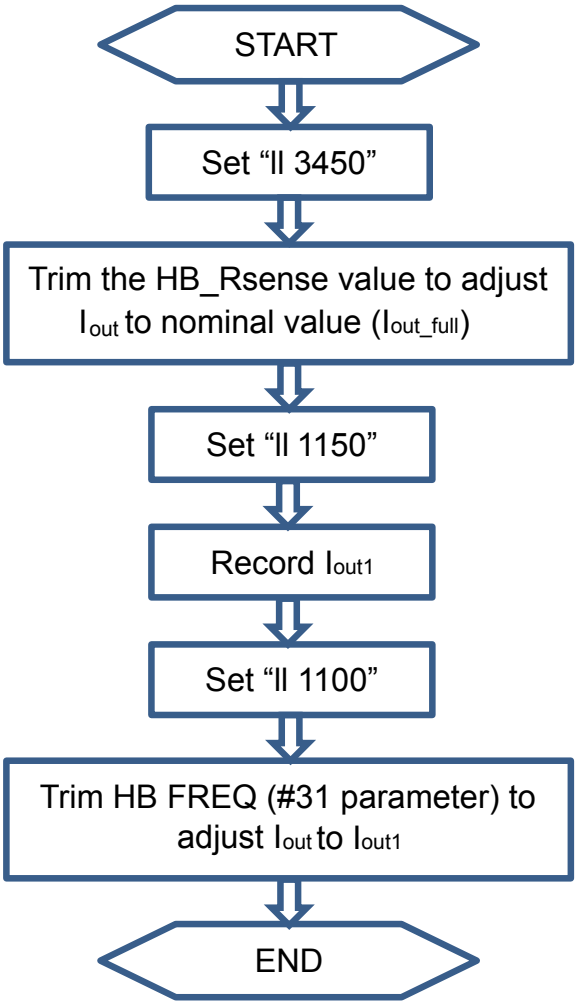
7.2 Fine-tune the half bridge stage

Ensure the following conditions are met before you start:

1. The board used does not have any open hardware issues (especially on PFC and HB stages)
2. Compile and load a new FW revision with serial command enable (USE_SERIAL_DEB 1 and USE_DEBUG_BOARD). This disables the GPGUI protocol and restores the command line environment.
3. The serial line is available and working
4. The PFC is working and stable in all output power ranges
5. The PFC Vout matches the PFC Vout design target
6. Board Vin is set to the fixed voltage (e.g., 220 Vac)
7. Vout is at nominal LED string voltage (maximum allowed)
8. The load is a real LED (don’t use an electronic load to trim the HB stage)

9. The HB parameters #34, #35 and #36 are controlled and adapted to the HW
10. The HB control loop is enabled (#6 = 1) and functions correctly and Iout is stable (i.e., no hardware issues)

Figure 14. HB fine tuning procedure



- `ll 3450` sets up the CN_CNT control voltage to 1.054 V, Iout (nominal) is fixed to this voltage level. If you change this point, the corresponding DALI table and 0-10V table need to be updated.
- `ll 1150` sets up the minimum output control using the CN_CNT voltage in HBZ_CLOSE mode
- `ll 1100` sets up the maximum Iout using HBZ_SKIPK mode

Refer to Section 2.2 Half bridge algorithm for more details regarding the half bridge.

A Hints and tips

A.1 Double the secondary output voltage

You can double the output voltage with this simple modification: remove the SMD resistor R102 and place D42 and D43 (STTH3R06S ultrafast rectifier - SMC case) diodes on the PCB.

When this modification, the maximum output voltage is around 200 V, but the maximum output current is 500 mA.

A.2 Input voltage range and PFC Vout

The STEVAL-ILL066V2 evaluation board voltage range is 90 Vac-265 Vac. The PFC output voltage is 410 Vdc. For a 277 Vac input, the maximum input voltage is 305 Vac, so it is necessary to modify the PFC output voltage to 460 Vdc.

From the schematic, the resistor network R46, R52, R55, and R58 provide voltage feedback to STLUX385A. The internal reference voltage Vref is 1.1151 V.

Since:

$$Vref = Vout^{*}\frac{R58}{(R58 + R55 + R52 + R46)} \quad (1)$$

…we can change R58 from 18K to 16K to get 460 V.

The OVP level will be changed as well. The internal OVP trigger point is 1.23 V. Therefore, by changing R58, the OVP level is moved to 507 V.

It is also necessary to change the MOSFET to 600 V rating and increase bulk capacitor voltage up to 510 V accordingly.

A.3 Output LED current

The output LED current can be calculated by the equation

$$ILED = Ipk^{*}\frac{n}{2} \quad (1)$$

…where Ipk is the primary side peak current and n is the turn ratio of the transformer.

The STEVAL-ILL066V2 evaluation board max output current 1 A (`Iout_setup` command value is 3450). The reference voltage Vref for the max current is 1.053 V and the turn ratio of the transformer is 1.85. Therefore, max LED current is:

$$ILED = \left(\frac{Vref}{Rcs}\right)^{*}\frac{1.85}{2} \quad (2)$$

The current sensing resistor Rcs is R36 and R37 in parallel. Because of propagation delay, the calculated value is lower than the actual value, so the resistor value in the circuit needs to be fine-tuned to obtain a precise LED current.

The maximum output current on the LEDs can be changed by modifying R36//R37. For example, to change the output current to 1.28 A at the same output voltage, R36//R37 can be changed to 0.90 Ω value (or use two 1.8 Ω resistors). Similarly, if the max output LED current needs to be reduced, the value of R36//R37 can be increased (see Section 5.5 HB control voltage).

A.4 Change Vout on LED string

To change the maximum output voltage from 100 V, try selecting transformer turn ratios that reflect the evaluation board.

For example, if the maximum output voltage needs to be 42 v, the new turn ratio would be:

$$N = n^{*}\frac{(100 + Vd)}{(42 + Vd)} \quad (1)$$

Where N is the new turn ratio, n is the old turn ratio 1:85; Vd is output diode voltage drop, 1 V.

Hence

$$N = 1.85 * \frac{101}{43} = 4.3 \quad (2)$$

A.5 Design of the circuit

If you are designing a circuit according to your own requirements, we have a design spreadsheet is available on request. It is a very helpful resource for designing the PFC choke, resonant inductor and output transformer.

Revision history

Table 4. **Document revision history**

Date	Revision	Change
13-Sep-2017	1	Initial release.
29-Jan-2018	2	Minor text edits

Contents

1 **STSW-ILL066V2 source code overview 2**
1.1 main.c 2
1.2 ee_gest.c 3
1.3 man_ee.c 3
1.4 LowPW.c 3
1.5 gp_parser.c 4
1.6 SMED_Init.c 4
1.7 stlux_itc.c 4
1.8 stlux_optonbyte.c 4
1.9 dali_IF.c 4
1.10 Dali_Stack 5
1.11 uart.c 5
1.12 Z10V.c 6
1.13 temp_ntc.c 6
2 **PFC and HB libraries 7**
2.1 PFC algorithm 7
2.2 Half bridge algorithm 10
3 **Library interfaces 12**
3.1 PFC library Interface 12
3.1.1 u8 PFC_init_var(void) 13
3.1.2 void check_PFC_Start(u8) 13
3.1.3 void pfc_interrupt(void) 13
3.1.4 INTERRUPT_HANDLER (ADC_ISR, 22) 14
3.1.5 u8 Update_PFC_param(void) 14
3.1.6 void PFC_set_vout_story(void) 14
3.1.7 u8 update_PFC_V(void) 14
3.1.8 void PFC_stop_PWM(void) 14
3.2 Half bridge library Interface 14
3.2.1 u8 HB_init_var(void) 16
3.2.2 void hb_interrupt(void) 16

3.2.3 INTERRUPT_HANDLER(SMED0_ISR, 6) . . . 16
3.2.4 void HB_start_PWM(void) . . . 16
3.2.5 u8 check_S_CH0(void) . . . 16
3.2.6 void update_CPM_val(void) . . . 17
3.2.7 void HB_tacur_update(void) . . . 17
3.2.8 void HB_burst_mode(u16) . . . 17
3.2.9 void New_PWM2(B2W) . . . 17
3.2.10 void Update_Out_Current(void) . . . 17
3.2.11 void HB_stop_PWM(void) . . . 18
3.2.12 void search_mm_HB_on(void) . . . 18
4 Variable parameter descriptions . . . 19
4.1 PFC activation enable . . . 21
4.2 HB activation enable . . . 21
4.3 Dali interface enable . . . 21
4.4 Startup output current level . . . 22
4.5 Half bridge current loop compensation . . . 22
4.6 Debug mode enable . . . 22
4.7 0-10 V interface enable . . . 22
4.8 PFC THD maximum time . . . 22
4.9 Enable the ROP code protection . . . 22
4.10 PFC DAC level (OCP) . . . 23
4.11 PFC activation level . . . 23
4.12 PFC input voltage stop level . . . 23
4.13 PFC level to enter zones . . . 23
4.14 PFC UVLO on output level . . . 24
4.15 PFC On time during start-up . . . 24
4.16 Off time during PFC start . . . 24
4.17 Delay to detect PFC input Vac missing . . . 24
4.18 Maximum available HB frequency . . . 25
4.19 Minimum available HB frequency . . . 25
4.20 HB No load level protection . . . 25

4.21 Propagation delay mismatch compensation . . . 25
4.22 Trimming high side delay . . . 25
4.23 Dead time delay . . . 26
4.24 PFC error to LOOP OK . . . 26
4.25 Fast HB current regulation . . . 26
4.26 PFC Integral Gain . . . 26
4.27 PFC Integral Gain Scaler . . . 26
4.28 PFC Proportional Gain . . . 27
4.29 PFC Proportional Gain Scaler . . . 27
4.30 PFC Over Voltage high speed level enable . . . 27
4.31 PFC Under Voltage high speed level enable . . . 27
4.32 PFC under voltage high speed gain . . . 27
4.33 Level to detect the short on the output . . . 27
4.34 Temperature to shut down the application . . . 28
4.35 Temperature to start power reduction . . . 28
4.36 Iout_setup slope reduction . . . 28

5 Fixed parameters . . . 29

5.1 Target PFC output voltage . . . 29
5.2 PFC input voltage rectified . . . 29
5.3 PFC output - second OVP level . . . 29
5.4 HB time increment during time stamp . . . 30
5.5 HB control voltage . . . 30
5.6 ADC input voltage on ADC[5] . . . 30
5.7 ADC input voltage on ADC[6] . . . 30

6 PFC check procedure . . . 31

6.1 Review of SMED states . . . 31
6.2 Conditions prior to testing . . . 31
6.3 Recommended test procedure . . . 32

7 Half bridge fine tuning . . . 34

7.1 Check the half bridge output power . . . 34

7.2 Fine-tune the half bridge stage 35
A Hints and tips 37
A.1 Double the secondary output voltage 37
A.2 Input voltage range and PFC Vout 37
A.3 Output LED current 37
A.4 Change Vout on LED string 37
A.5 Design of the circuit 38
Revision history 39

List of figures

Figure 1. Source code interaction - top view . . . 2
Figure 2. Library interaction - top view . . . 7
Figure 3. Control loop of PFC stage: block diagram . . . 8
Figure 4. Control loop of PFC stage: electrical circuit and main quantities . . . 8
Figure 5. STLUX PFC algorithm . . . 9
Figure 6. STLUX PFC PI parameter relationships . . . 10
Figure 7. Half bridge signal relations . . . 11
Figure 8. PFC concept . . . 12
Figure 9. PFC library concept . . . 13
Figure 10. ZVS concept . . . 15
Figure 11. HB library concept . . . 16
Figure 12. STLUX PFC - SMED state machine . . . 31
Figure 13. HB output power check procedure . . . 35
Figure 14. HB fine tuning procedure . . . 36

List of tables

Table 1. Kp Gain function of Vac input voltage . . . 9
Table 2. STEVAL-ILL066V2 evaluation board library parameters . . . 19
Table 3. Fixed parameters table . . . 29
Table 4. Document revision history . . . 39