AN11174

DALI slave using the LPC111x

Rev. 1 — 1 March 2012 **Application note**

Document information

Info	Content
Keywords	Lighting networks, DALI, CORTEX M0, ARM, LPC111x, Microcontroller
Abstract	This application note describes how to create a DALI slave device with the OM13026 LPC111x DALI slave board.

Revision history

Rev	Date	Description
1	20120301	Initial version.

Contact information

For more information, please visit: http://www.nxp.com

For sales office addresses, please send an email to: salesaddresses@nxp.com

1. Introduction

The aim of this document is to describe how to create a *DALI slave device* with the OM13026 DALI slave board.

It covers the following aspects:

- Short introduction into DALI lighting networks
- OM13026 LPC111x DALI slave board design description
- Global software overview
- Supported DALI command set
- Software configuration items

This application note does *not* include items covering the light generation aspect, especially:

- Description of lighting drivers (CFL, SSL)
- Lamp feedback (color, temperature)

1.1 Overview

This application note is intended for usage with the DALI slave board OM13026 as depicted in Fig 1.

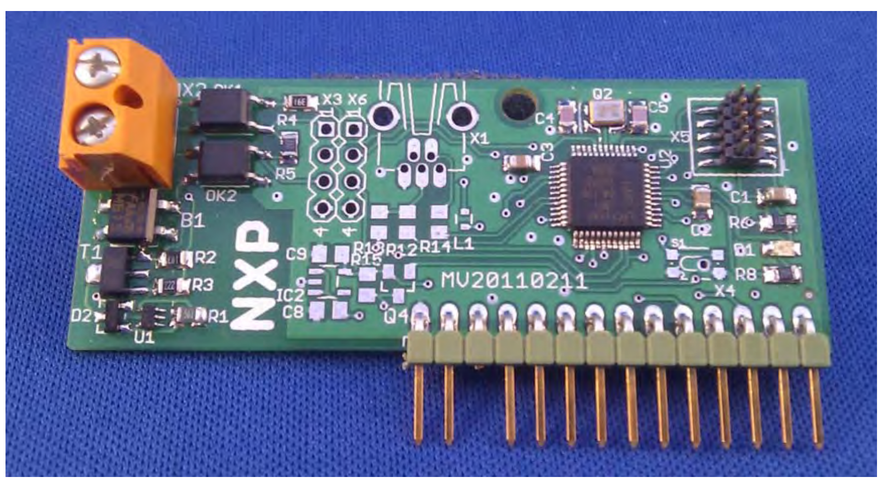


Fig 1. OM13026 LPC111x DALI Slave board

The OM13026 board contains an example implementation of an isolated physical layer for the DALI bus with a Cortex M0 LPC111x microcontroller for the DALI protocol handling, and many I/O functions to steer external lighting drivers for solid state or compact fluorescent lighting applications.

1.2 DALI

This section briefly describes the Digital Addressable Lighting Interface (DALI) to understand the terms and concepts used in the other chapters of this document. For more information on the DALI standard, see References [1], [2], and [3].

1.2.1 Bus structure

DALI uses a wired bus structure to create a communication path between control devices (switches or central gateways) and control gear (lighting units). The topology is not specified; it can be a star, bus or point to point. A ring structure is not allowed.

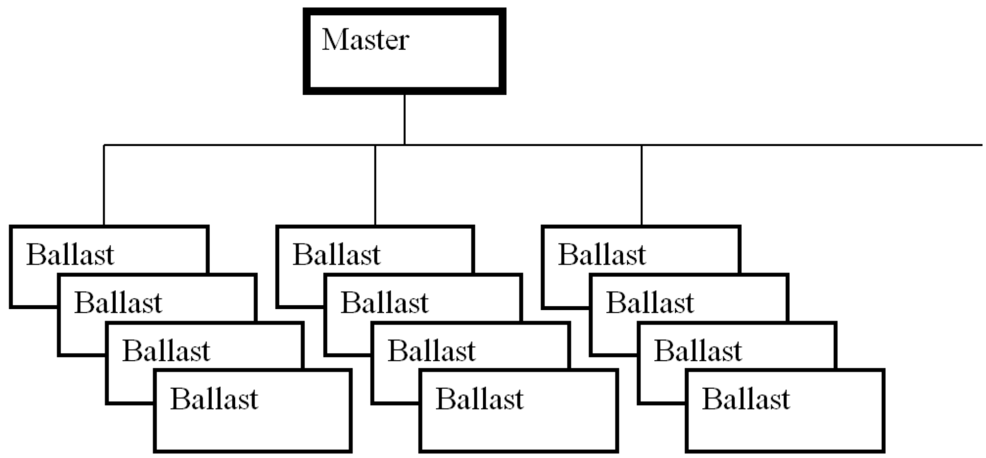


Fig 2. Example bus of control device and multiple control gear

For installation purposes, the DALI signal wiring can be combined with mains connections in one cable as shown in Fig 3.

Fig 3. DALI wires combined with mains power wires

1.2.2 Physical layer

The DALI bus consists of two wires on which data is transmitted in frames. There are two different frame types: a “forward” frame (2 bytes sent by the master to the slave) and a “backward” frame (1 byte sent by the slave to the master on request of the master).

DALI uses a bi-phase (also called Manchester) encoding, which means that the data is transmitted using the edges of the signal. A rising edge indicates a ‘1’, and a falling edge indicates a ‘0’ (see Fig 4).

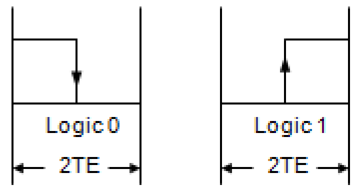


Fig 4. Bi-phase encoding of data

The encoded bits are actually represented by high and low voltage levels on the bus. Typically, the low voltage is 0 V and the high voltage is 16 V. The maximum and minimum bus voltage at both the transmitting unit and the receiving unit are given in Fig 5.

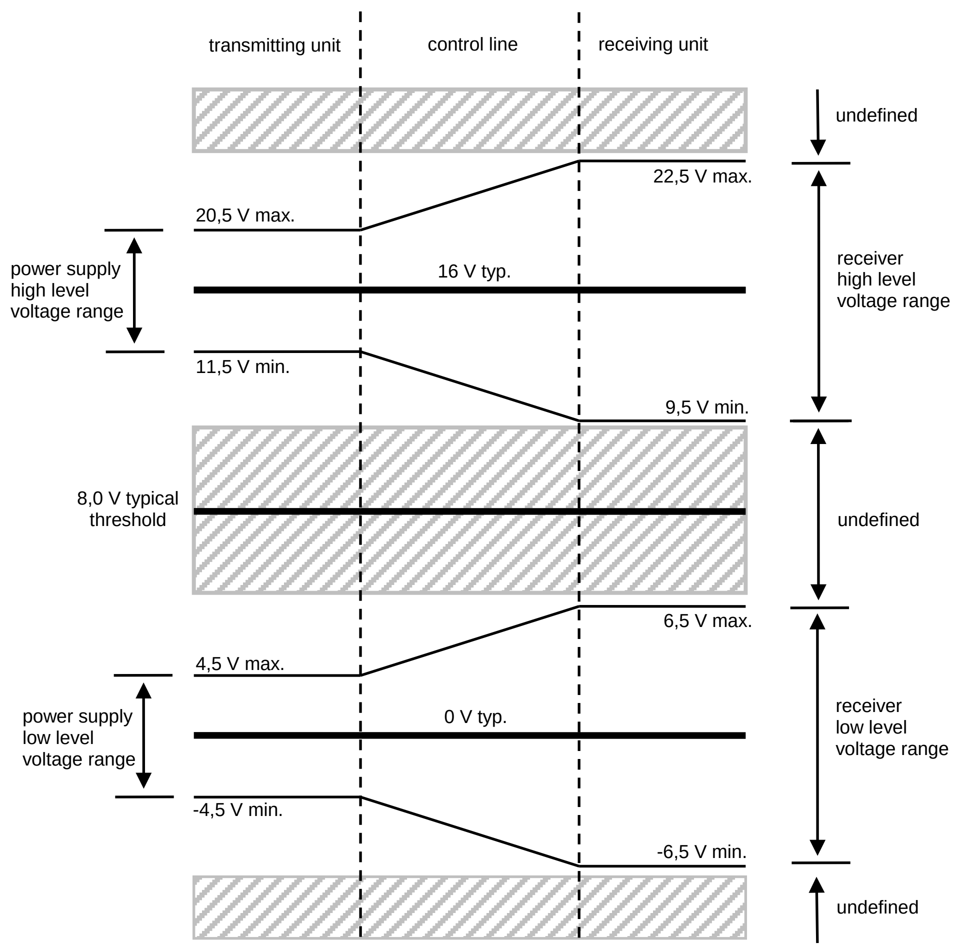


Fig 5. Bus voltage levels

The bus is powered via a bus power supply, which can be part of the master on the bus and has a current limit of 250 mA. Whenever a control gear on the bus wants to modulate the bus voltage, it will short circuit the bus to create a low voltage level. If a high level is desired, it will put its transmission stage in a high impedance state.

1.2.3 Logical layer

In a typical application, a DALI-bus consists of one controller (master), and multiple slaves (normally TL-ballasts). It can control up to 64 different slaves (ballasts) within the same control system. It’s possible to transmit commands to single ballasts or to a group of ballasts.

Every bit takes two periods TE. The defined bit rate of DALI is 1200 bps. So a 1 bit period (2TE) is ~834 µsec. A frame is started by a start bit, and ends with two high-level stop bits (no change of phase). Data is transmitted with the MSB first. Between frames, the bus is in idle (high) state (see Fig 6).

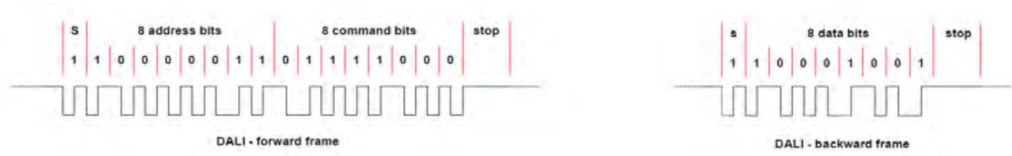


Fig 6. Physical encoding

Additional protocol timing requirements for transmission are:

- The settling time between two subsequent forward frames shall be at least 9.17 ms. This means that 4 forward frames with accompanying periods of 9.17 ms shall fit exactly in 100 ms.
- The settling time between forward and backward frames (transition from forward to backward) shall be between 2.92 ms and 9.17 ms. After sending the forward frame, the master unit will wait for 9.17 ms. If no backward frame has been started after 9.17 ms this is interpreted as “no answer” from slave.

The settling time between backward and forward frames (transition from backward to forward) shall be at least 9.17 ms (see Fig 7).

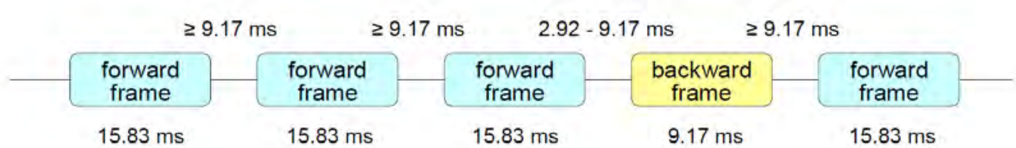


Fig 7. Timing of forward and backward frames

Every DALI slave is able to react to a short address, 16 group addresses and broadcast. For addressing the following scheme is shown in Fig 8:

type of address	address byte
Short or group address	YAAAAAAS
Short addresses (0 - 63)	0AAAAAAS
Group addresses (0- 15)	100AAAAS
Broadcast	1111111S
Special command	101CCCC1
Special command	110CCCC1

S: selector bit — S = '0' direct arc power level following; S = '1' command following

Y: short- or group address — Y = '0' short address; Y = '1' group address or broadcast

A: significant address bit

C: significant command bit

Fig 8. DALI addressing types

After the address byte, a second byte follows the forward frame. This second byte contains the direct arc power level for the ballast or a command byte depending on the selector bit. There are four types of commands:

1. Direct/Indirect arc power control commands – used to set ballast power level
2. Configuration commands – configures the ballast (for example: add to a group or store level). Command must be repeated within 100 ms, otherwise it's ignored.
3. Query commands – ask slave (ballast) for status information (for example: power level or version number). The slave can send a backward frame.
4. Special commands – used to initialize and setup the ballast, some must be repeated within 100 ms, and some require an answer from the slave. Most commands are only processed within 15 minutes after an "INITIALIZE" command is received.

2. Hardware description

This section describes the hardware of the DALI slave board. First the physical layer of the board will be discussed followed the complete board description.

2.1 Physical layer

The schematic of the DALI physical layer from board OM13026 is depicted in Fig 9. The DALI bus connects to the connections D. The microcontroller LPC111x is connected to the signals DALI1_TX and DALI1_RX.

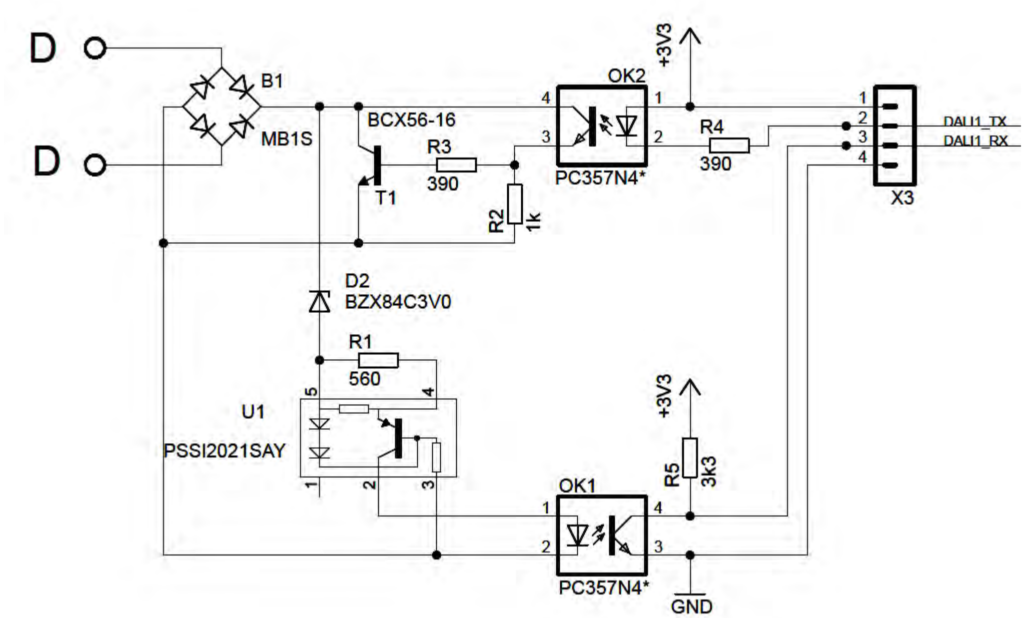


Fig 9. Physical layer schematic

By usage of bridge rectifier B1, the design is made polarity independent. Both terminals D are interchangeable.

The upper part of Fig 9 contains the transmission part of the DALI slave. It is created around T1, R2, R3, OK2 and R4. The signal DALI1_TX is driven by the microcontroller at 0 V or 3.3 V. For high signals of DALI1_TX, Optocoupler OK2 will connect the junction of R2 and R3 to the DALI bus. This will create a drive current for the base of T1 that will start to conduct and short circuit the DALI bus via bridge B1. When the signal DALI1_TX is low, transistor T1 will not conduct and the bus will be in the ‘high’ state. The resistor R3 of 390 Ohm is chosen such that T1 will sufficiently go into saturation and thus create a low voltage level on the bus while having a low voltage drop on T1. This keeps the power dissipation of T1 limited as it should be able to sink the maximum DALI bus current of 250 mA. Also, for this reason, T1 should have a high h_{FE} of minimum 100. In this way no additional cooling area is needed.

The reception path is shown in the lower part of Fig 9. It is created around optocoupler OK1, U1, R1, R5 and zener diode D2. When the DALI bus is idle (high) a constant current source of about 1 mA is created using U1 and R1. This current is used to drive optocoupler OK1 that signals the level of the DALI bus to the microcontroller via DALI1_RX. The current source limits the maximal current load the circuit creates when not in transmission mode. Zener D2 and bridge rectifier B1 drop the received bus voltage to a level to guarantee that a low level voltage of 6.5 V does not drive the optocoupler.

The circuit around OK1 and R3 creates an inverted signal to the microcontroller: A high DALI bus level will connect DALI1_RX to low. A low DALI bus level will create a high signal on DALI1_RX.

All components are chosen to withstand several factors (70 V to 80 V) of the highest allowed DALI bus voltage level of 22.5 V. The physical layer does not contain overvoltage protection or suppression components. This is left up to the reader.

The optocouplers create isolation between the microcontroller side and the DALI bus. The isolation is sufficient for evaluation of the DALI software stack when the microcontroller is connected in a non isolated way to the mains supply; any re-use of this design should be made compliant to the isolation requirements as specified in section 5.4 of Reference [1].

2.2 Microcontroller

The section of the design which handles the incoming and outgoing DALI messages and controls the lighting is shown in Fig 10. The DALI1_RX signal from the DALI physical interface is connected to a 32-bit timer/capture unit CT32B0 of U2. The DALI1_TX signal is connected to a general purpose IO pin that is software controlled to generate the DALI signal timing. Connector X6 makes is possible to connect a second, different, physical interface to the timer/capture unit CT32B1 of U2.

Debugging and flashing connection is provided by means of header X5, which complies with the 10-pin SWD standard as supported by many flash and software tools.

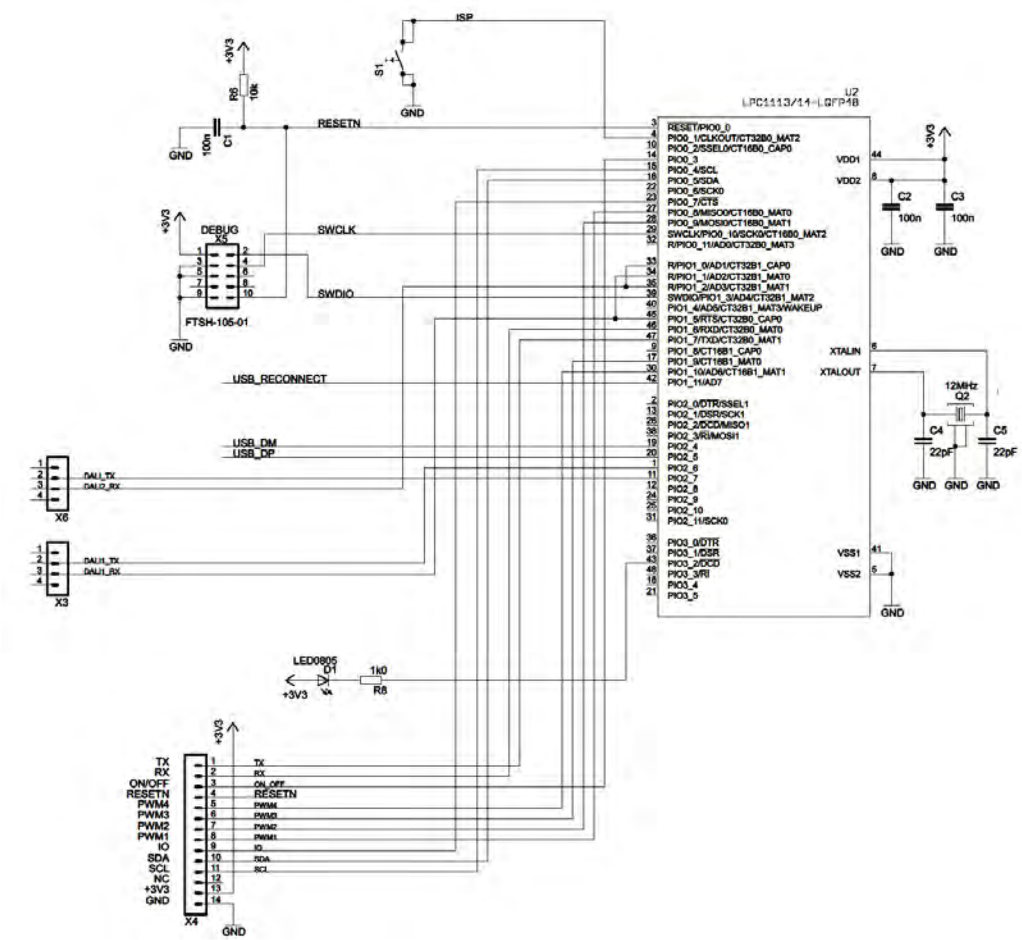


Fig 10. Microcontroller connections

The OM13026 board can derive its microcontroller clock from a) an external connected crystal (circuit Q2, C4, C5) or b) an internal RC oscillator. This is configurable in the application software.

The function of header X4 is twofold: it contains the 3.3 V supply on pins 13 and 14 and contains many input/output controls on pins 1 to 11. The board should be powered by an external 3.3 V supply. The current consumed is in the range between 2 mA and 15 mA depending on the clock configuration of the microcontroller (internally clocked vs external crystal vs clock speed running up to 48 MHz).

The input/output controls are listed in Table 1. It shows the default function of the pins on header X4.

Table 1. Header X4 description

Pin	Description	Direction	LPC1114 pin	Remark
1	TX	O	PIO1_7/TXD/CT32B0_MAT1	UART
2	RX	I	PIO1_6/RXD/CT32B0_MAT0	UART
3	ON_OFF	O	PIO0_3	
4	RESETN	I/O	nReset/PIO0_0	
5	PWM1	O	PIO0_8/MISO0/CT16B0_MAT0	
6	PWM2	O	PIO0_9/MOSI0/CT16B0_MAT1	
7	PWM3	O	PIO1_9/CT16B1_MAT0	
8	PWM4	O	PIO1_10/AD6/CT16B1_MAT1	
9	IO	I/O	PIO0_7/nCTS	
10	SDA	I/O	PIO0_5/SDA	I2C
11	SCL	O	PIO0_4/SCL	I2C
12	nc			
13	+3V3	I	VDD1,2	
14	GND	I	VSS1,2	

Up to four PWM signals are available to independently drive different lighting units. The frequency and resolution of the signals is software programmable. The ON_OFF signal can be used independently from the PWM signals to switch a lighting driver into an OFF or ON state. The I^{2}C-bus pins can be used to externally connect other devices like EEPROM or a temperature sensor. An analog A/D input is available via pin 8. The IO signal is left open for the end user.

UART, I^{2}C and A/D converter functionality is not included in the software release.

2.3 Board layout

The layout of the board is given in Fig 11. On header X4, pin 12 is removed to function as key to circumvent misplacement when attaching it to a lighting driver.

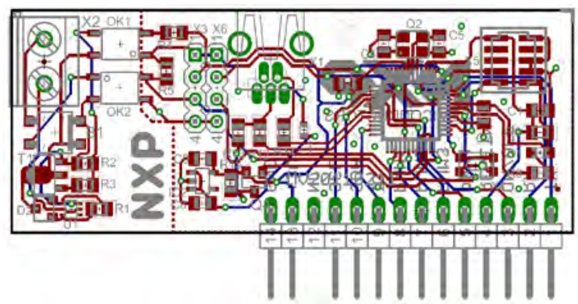


Fig 11. PCB layout

In Fig 11, the optical isolated DALI physical interface is placed on the left side of the dotted line. On the right side of the board the LPC1114 microcontroller with the 10-pin SWD debug/programming header is placed. The middle part of the printed circuit board is not populated and not necessary for use as DALI slave unit. Header X6 is meant to connect a second, different DALI physical layer or can be used for an additional GPIO or 32-bit timer output. The other components are used when U2 is mounted with the pin compatible LPC1343.

The PCB does not have milling at the isolation border between the microcontroller side and the DALI bus interface. It is advised to recheck the layout design to comply with the isolation requirements as specified in section 5.4 of Reference [1].

2.4 Component list

The list of components is given in Table 2. The oscillator circuit around C4, C5 and Q2 is listed as optional; the microcontroller can also use its internal reference clock to generate its internal clock. This is configurable by software.

The USB circuitry around components IC2, L1, X1, Q4, R15, R12, R14, R13, C8 and C9 is not mounted: this part of the circuit only applies when the footprint of U2 is mounted with the pin compatible LPC1343, which is not the case for the OM13026 board.

Table 2. List of components

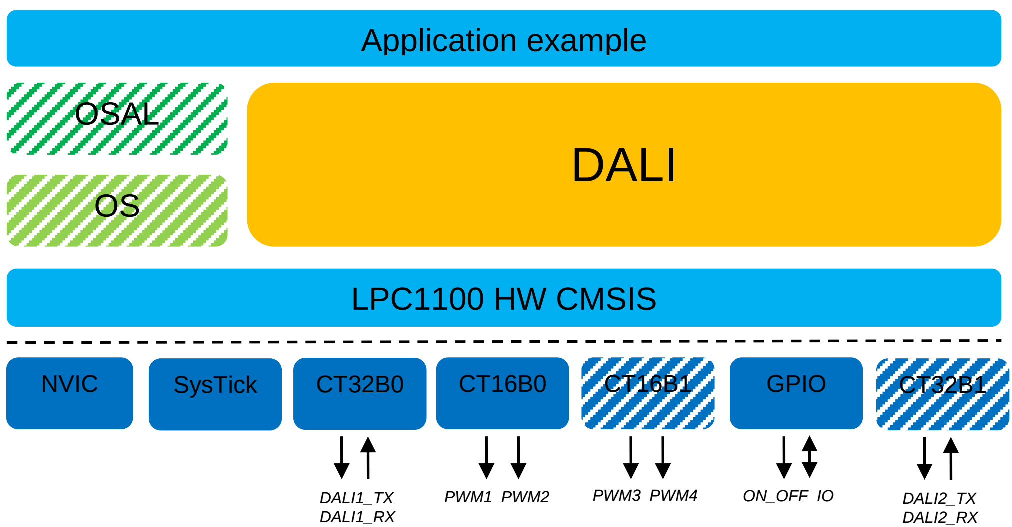
Part ref.	Description	Manufacturer	Package	Remarks
B1	MB1S		SOIC-4	
C1	100nF		0805	
C2	100nF		0805	
C3	100nF		0805	
C4	22pF		0805	optional
C5	22pF		0805	optional
C8	4u7/10V/X7R		0805	not mounted
C9	2u2/6V3/X7R		0805	not mounted
D1	LED0805		CHIPLED_0805	
D2	BZX84C3V0	NXP Semiconductors	TO236	
IC2	SA57000-33D	NXP Semiconductors		not mounted
L1	BLM18AG601S		0603	not mounted
OK1	PC357N4	Sharp		
OK2	PC357N4	Sharp		
Q2	12MHz			optional
Q4	BSH205	NXP Semiconductors	SOT23	
R1	560R		0805	
R2	1k		0805	
R3	390R		0805	
R4	390R		0805	
R5	3k3		0805	
R6	10k		0805	
R8	1k0		0805	
R12	33R		0805	not mounted
R13	1k5		0805	not mounted
R14	33R		0805	not mounted
R15	nc		0805	
S1	KMR211G	C&K Components		not mounted
T1	BCX56-16	NXP Semiconductors	SOT89-BCE	
U1	PSSI2021SAY	NXP Semiconductors		SOT353
U2	LPC111x	NXP Semiconductors	LQFP48	
X1	565790519	MOLEX	MINI-AB USB	not mounted
X2	MKDSN1,5/2-5			
X3	MA04-1R			not mounted
X4	MA14-1W			pin 12 removed
X5	FTSH-105-01	SAMTEC		
X6	MA04-1R			not mounted

3. Software description

This chapter describes the structure and components of the software project for usage with hardware board OM13026 to create a DALI control gear. After a small software overview the DALI stack will be explained in detail. The other components of the example software project are not discussed in depth.

3.1 Decomposition

The decomposition of the software is given in Fig 12. Inclusion of shaded parts into the DALI control gear can be selected at compile time. For the software interface to the hardware peripherals of the LCP111x the Cortex Microcontroller Software Interface Standard (CMSIS) is used.



(1) shaded/striped items are optional

Fig 12. System decomposition

The DALI component handles the reception and transmission of DALI frames using the CT32B0 timer/capture unit. For ballast control the 16 bit timer units CT16B0 and CT16B1 are used to generate pulse width modulated (PWM) signals. General purpose IO pins can be used to signal a lighting driver to switch on or off or to read lamp failure information into the DALI component.

The DALI stack and the example application use a small operating system abstraction layer (OSAL). This OSAL provides service like message transfers via queues and thread creation. The complete set of functions of the OSAL layer is not topic of this application note and is not discussed.

The functions of the OSAL layer map to multiple operating systems (OS). The target OS can be selected at compile time. The software described supports two mappings: a version without an OS version and a version that maps onto the FreeRTOS OS. The version without OS support implements a minimal set of functions to enable the DALI stack to run. It does not support multiple threads or other advanced functions of the OSAL. The FreeRTOS version supports all the standard functions like queues, semaphores, threads and their scheduling. The default configuration of the software is to not use an OS.

The applications example, the ‘main’ of the program, is started on top of the OSAL and directly calls the DALI stack. It does not provide any other services in the case the OSAL uses the OS-less version. In the case the OSAL maps to real OS, the application layer can implement multiple applications using different threads.

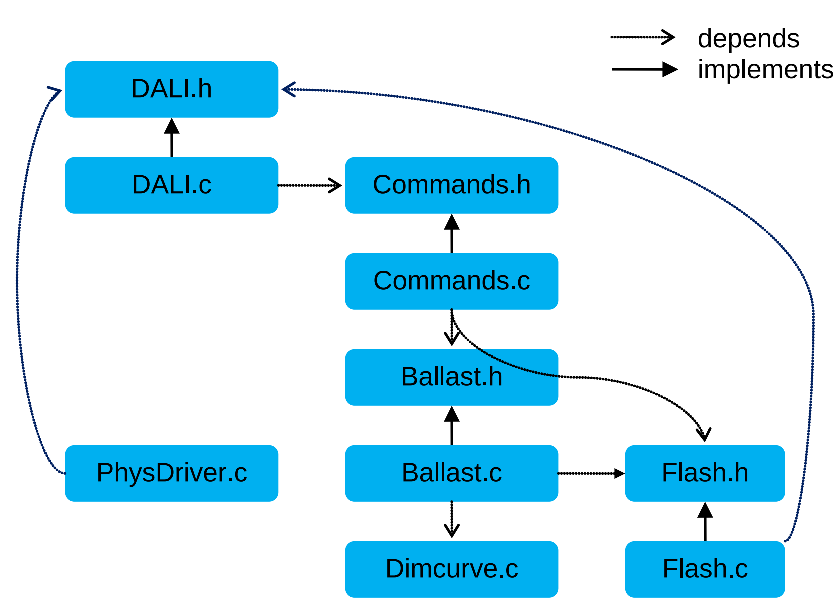
3.2 Component structure

The DALI component is responsible for:

- receiving and transmitting DALI frames from the physical interface
- acting upon received commands
- driving the connected lighting unit
- persistent storage of parameters

The functions for the physical DALI bus interface for forward frame and backward frame transmission are bundled in the file *DALI_PhysDriver.c*. The physical drives passes the commands on to *DALI.c*, that executes a command handling loop. The command parsing and backward frame information assembly is done in *DALI_Commands.c*. The actual control of the ballast is done in *DALI_Ballast.c*. Storage of nonvolatile parameters is done using *Flash.c*.

All the files of the DALI component are located in the directory *DALI*. The subdirectories *inc* and *src* contain the respective header files and source files. The relationship between the files is given in Fig 13.



(1) Filenames are prefixed with DALI_ , except for DALI.c and DALI.h.

Fig 13. DALI component file structure

The *DALI.h* contains the global definitions used in the component. It defines the external interfaces of the component and all type definition for the internal administration of the component. All DALI component internal administration is held in a *workspace* that is used in all files.

For each physical interface called *channel* the administration holds the *status* of that channel and a compound of {*timer*, *channel_config*, *driver* and *device class*} in which *timer* is a reference to the timer/capture unit in use for this interface, *channel_config* holds the PIO port and pins of the interface, *driver* is the internal interrupt driver workspace and *device class* holds all DALI device configuration information. This setup makes it possible to create multiple logical DALI devices using the same physical bus interface.

3.3 Run time flow

The DALI component has one main control flow, shown in the left side of Fig 14. After initialization the DALI component waits for a forward frame from a control device. This forward frame is received by the Interrupt Service Routine (ISR) shown on the right side of Fig 14. When the ISR has a complete forward frame it puts the message in a queue for the waiting main loop. The main loop unblocks and executes the desired actions depending on the contents of the forward frame. If the receiving command in the forward frame requires a response, the ISR is instructed to generate a backward frame.

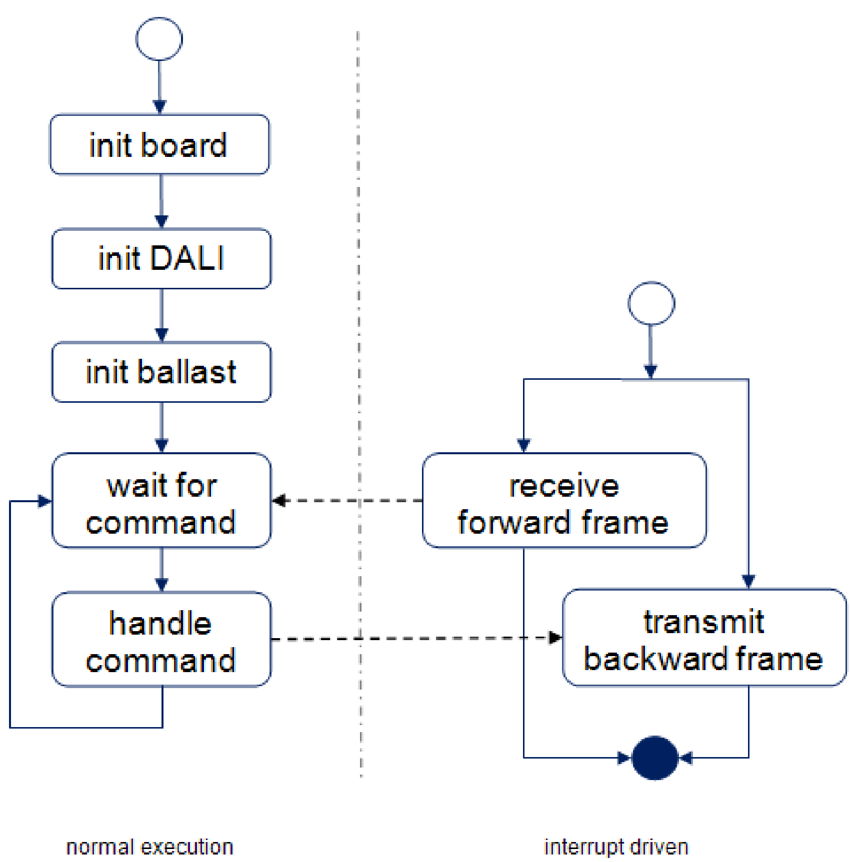


Fig 14. Simplified execution flow

By default the ISR routine is in receive mode and will switch to transmit mode when instructed by the main loop. When the transmission of the backward frame is ready, the ISR routine automatically switches back to receive mode.

The interrupt driven reception of DALI frames is done bit by bit; multiple calls of the ISR routine concatenate the received bits until a full frame is received. The same holds for the transmission part of the ISR: it generates a frame on the bus on the basis of subsequent invocations of the ISR, and at each invocation the bus is driven to create bi-phase coded bits on the bus.

Additionally, the system contains a system tick ISR routine for creating a system time, lighting fading effects and an ISR to enable interpolation steps between DALI arc power levels.

The init board, init DALI and init ballast are in the `main` function located in file *DALIDemo/main.c.* The endless loop that waits for forward frames and the handling forward frames is performed by the function `DALI_CommandHandler` in *DALI/src/DALI_Command.c*. The interrupt service `DALI_IRQHandler` for frame reception and transmission is located in /DALI/src/DALI_PhysDriver.c

3.4 DALI reception and transmission

The physical layer circuit of Fig 9 is connected to three pins of the microcontroller unit. The incoming DALI signal is connected to a capture input of the timer capture unit and to a GPIO input. The DALI TX signal is connected a GPIO output.

3.4.1 Inter frame timing

The match register 0 (MR0) of the timer capture unit is used to create the minimal frame delay between forward frames (see Fig 7). The MR0 is set to 22Te. When this match register creates an interrupt, the ISR reception state is set to receive mode and capture interrupts are enabled.

3.4.2 Forward frame reception

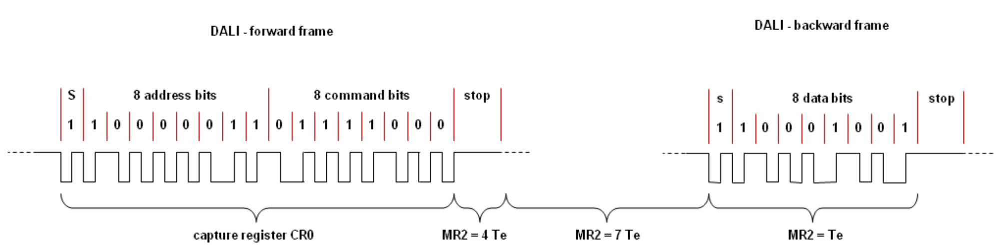
The capture function of the timer capture unit is used to measure the pulse width of the incoming DALI message; the timer capture unit is set to a clock of 1 MHz. Upon the first transition of the DALI1_RX line the timer counter is reset. During each following DALI1_RX transition, either to low or to high, the ISR is triggered and the width of the pulse is measured using capture register 0 (CR0). A bitstream is generated of which the contents depend on the width of the pulse (due to the Manchester encoding of the DALI bus signal). After receiving 17 manchester encoded bits the ISR detects the stopbits with a duration of 4*Te using match register 2 (MR2). When the ISR triggers on MR2 after 4*Te, and there has been no DALI line activity, the forward frame bi-phase bits are put in the reception queue by the ISR for processing by the main loop. Fig 15 illustrates the relation between the bus timing and the corresponding timer capture and match registers in use.

3.4.3 Command handling

The main loop blocks until a bi-phase encoded forward frame is put in the reception queue by the ISR. It copies the forward frame and decodes it to a command with data. This command is then parsed and the corresponding action is performed.

3.4.4 Backward frame transmission

When the main loop decides that a backward frame should be transmitted, it encodes the response into a Manchester encoded message and sets the ISR state to SEND mode while enabling the interrupt match register 2 (MR2) with an initial time of 7*Te. On the first time MR2 fires the ISR is triggered and the MR2 value is set to Te. On each subsequent interrupt by MR2 the DALI1_TX GPIO is set by the ISR to the corresponding high/low level of the Manchester encoded backward frame. When all bits of the backward frame are transmitted the ISR state is set to RECEIVE mode. This is illustrated in Fig 15.



Capture timer unit running on 1MHz

Fig 15. Physical encoding/decoding using capture timer unit

3.4.5 Interrupts

The following table shows the interrupts that are handled by the software.

Table 3. DALI slave interrupts in use

Interrupt	Description
`TIMER16_0_IRQ`	Generates signals PWM1 and PWM2 Used for driving lighting units
`TIMER16_1_IRQ`	Generates PWM2 and PWM3 signals Used for driving lighting units
`TIMER32_0_IRQ`	Handles DALI interface #1
`TIMER32_1_IRQ`	Handles DALI interface #2 (not used on demoboard)
`SysTick`	Generates system time

3.5 Lighting driver control

The software provides two ways to control lighting drivers: a high/low ON/OFF signal and up to four pulse width modulated (PWM) signals to steer the brightness level of a light device. Usage of the ON/OFF signal is optional. The PWM signals are generated using the 16 bit timer units. All the functionality of lighting driver control is contained in the file *DALI/src/DALI_ballast.c*.

Timer capture unit CT16B0 is used for PWM signals 1 and 2; timer capture unit CT16B1 is used for generating PWM signal 3 and 4. For each of these timer capture units the match register MR3 is used to set the PWM frequency. Match registers MR0 and MR1 are used in both timer units to set the duty cycle of the PWM signal. Once setup the resulting PWM signal is generated without software intervention. For more information on the mode of operation of the timer capture unit, refer to the timer chapter in the LPC111x user manual[4].

Each DALI arc power level is translated via a look up table to a duty cycle setting of the timer unit. For each output a separate lookup table can be used. The lookup tables are defined in *DALI_Dimcurve.c*. When using the DALI fade features, the software can calculate intermediate steps for creation of smooth fade effects. This is done in the function *FadeTickHandler*.

Coupling of lamp failure information is lighting driver dependent. Any lamp failure information can be coupled to the DALI stack in the function *LampFailure* in *DALI/src/DALI_ballast.c*.

3.6 Supported commands

The following tables give an overview of the supported commands of the accompanying software. If there is no remark the command is implemented.

Table 4. Indirect arc power control commands

Command nr	Description	Remarks
0	OFF	
1	UP	
2	DOWN	
3	STEP UP	
4	STEP DOWN	
5	RECALL MAX LEVEL	
6	RECALL MIN LEVEL	
7	STEP DOWN AND OFF	
8	ON AND STEP UP	
9	ENABLE DAPC SEQUENCE	unsupported
10-19	RESERVED COMMANDS	
16-31	GO TO SCENE	

Table 5. Configuration commands

Command nr	Description	Remarks
32	RESET	
33	STORE ACTUAL LEVEL IN DTR	
34-41	RESERVED COMMANDS	
42	STORE DTR AS MAX LEVEL	
43	STORE DTR AS MIN LEVEL	
44	STORE DTR AS SYSTEM FAILURE	
45	STORE DTR AS POWER ON LEVEL	
46	STORE DTR AS FADE TIME	
47	STORE DTR AS FADE RATE	
48-63	RESERVED COMMANDS	

Command nr	Description	Remarks
64-79	STORE DTR AS SCENE	
80-95	REMOVE FROM SCENE	
96-111	ADD TO GROUP	
112-127	REMOVE FROM GROUP	
128	STORE DTR AS SHORT ADR	
64-79	STORE DTR AS SCENE	
80-95	REMOVE FROM SCENE	
96-11	ADD TO GROUP	
112-127	REMOVE FROM GROUP	
128	STORE DTR AS SHORT ADR	
129	ENABLE WRITE MEMORY	unsupported
130-131	RESERVED COMMANDS	

Table 6. Query commands

Command nr	Description	Remarks
144	QUERY STATUS	
145	QUERY CONTROL GEAR	
146	QUERY LAMP FAILURE	
147	QUERY LAMP POWER ON	
148	QUERY LIMIT ERROR	
149	QUERY RESET STATE	
150	QUERY MISSING SHORT ADR	
151	QUERY VERSION NUMBER	
152	QUERY CONTENT DTR	
153	QUERY DEVICE TYPE	
154	QUERY PHYS MIN LEVEL	
155	QUERY POWER FAILURE	
156	QUERY CONTENT DTR1	
157	QUERY CONTENT DTR2	
158-159	RESERVED COMMANDS	
160	QUERY ACTUAL LEVEL	
161	QUERY MAX LEVEL	
162	QUERY MIN LEVEL	
163	QUERY POWER ON LEVEL	
164	QUERY SYSTEM FAILURE LEVEL	
165	QUERY FADE TIME RATE	
166-175	RESERVED COMMANDS	
176-191	QUERY SCENE LEVEL	
192	QUERY GROUPS 0-7	
193	QUERY GROUPS 8-15	

Command nr	Description	Remarks
194	QUERY RANDOM ADDRESS (H)	
195	QUERY RANDOM ADDRESS (M)	
196	QUERY RANDOM ADDRESS (L)	
197	READ MEMORY LOCATION	unsupported
198-223	RESERVED COMMANDS	
224-254	APPLICATION EXTENDED COMMANDS	unsupported
255	QUERY EXTENDED VERSION NUMBER	unsupported

Table 7. Special commands

Command nr	Description	Remarks
256	TERMINATE	
257	DATA TRANSFER REGISTER	
258	INITIALIZE	
259	RANDOMIZE	
260	COMPARE	
261	WITHDRAW	
262	RESERVED1	
263	RESERVED2	
264	SEARCH ADDRESS H	
265	SEARCH ADDRESS M	
266	SEARCH ADDRESS L	
267	PROGRAM SHORT ADDRESS	
268	VERIFY SHORT ADDRESS	
269	QUERY SHORT ADDRESS	
270	PHYSICAL SELECTION	lighting driver coupling open
271	RESERVED	
271	RESERVED	
272	ENABLE DEVICE TYPE X	unsupported
273	DATA TRANSFER REGISTER 1	unsupported
274	DATA TRANSFER REGISTER 2	unsupported

3.7 Command extensions

In general, two command extension types are possible:

- unsupported commands
- multi byte commands

The first extension of unsupported commands complements the software of this application note with non supported commands that are part of the DALI specification. The second extension of multi byte commands is intended for three byte manufacturer specific commands.

To extend the software with unsupported commands, the functions `DALI_ProcessNormalCommand` and `DALI_ProcessSpecialCommand` in *DALI/src/DALI_Command.c* should be extended. The actual command definitions are in *DALI/inc/DALI.h.*

To extend the software with three byte commands the function `DALI_IRQHandler` should be extended to support three byte commands. First, the stop bit detection should be altered to discriminate between two and three byte commands. Second, the function `DALI_Decode` should be changed to decode three byte commands. Third, the `CommandHandler` in DALI/src/DALI_Command.c should be extended to handle the incoming three byte commands.

3.8 NVM storage

Each DALI ballast has a series of properties called 'variables' (see table 6 - IEC 62386) e.g., actual level, power level, min/max level, short address etc. These properties are persistently stored in the nonvolatile flash memory of the microcontroller. For this, sector 7 of the flash memory is used as storage.

All nonvolatile variables are stored in one record in the flash sector. When one of the variables is changed, the complete record is stored in the flash sector. As the record is relatively small in respect to the sector size, multiple records fit in one sector. The software routines in *DALI/src/DALI_Flash.c* ensure that only the last valid record is used. The record contains a version indicator. As the contents of the flash sector are 0xFF after an erase, it is guaranteed that the software always finds the first valid records when searching downwards from the highest address of the flash sector 7. Nonvolatile records are stored incrementally from the start of the sector.

When using multiple DALI addresses in one physical device all variables of each logical DALI address are stored separately. When one of the DALI variables is changed, writing into the flash memory is delayed until the device does not receive commands for `UPDATE_DEVICE_CONF_TIMEOUT` milliseconds. This prevents the DALI device from not responding to DALI frames; during flash erase/write no other tasks can be done on the processor core of the microcontroller.

3.9 Configuration options

To simplify the usage of the software, several pre-configured ballast configurations are included. The ballast configurations vary the numbers of DALI devices in the slave, the frequency of the PWM signals, and enable usage of the on/off signal. These configurations can be set in the file *DALI/inc/DALI.h*. Only one configuration can be active at any time.

Table 8. Pre-defined ballast configuration options

Configuration	# control gear	Ballast drive		
		PWM signals		On/off signal
		Frequency (Hz)	Inverted	
`CFL`	1	2200	No	Yes
`SSL1523`	1	300	No	No
`UBA3070_1CH`	1	732	Yes	No
`UBA3070_4CH`	4	732	Yes	No
`DRIVER_3CH`	3	732	No	No

The options of Table 8 set several ballast and communication configurations. A more elaborate overview of the configuration options is given in Table 9.

Table 9. Additional configuration options

Configuration	Description	File location
TE	Bit length in microseconds	DALI/inc/DALI.h
MIN_TE	Minimal Te length	DALI/inc/DALI.h
MAX_TE	Maximal Te length	DALI/inc/DALI.h
MIN_2TE	Minimal length for 1 bit	DALI/inc/DALI.h
MAX_2TE	Maximal length for 1 bit	DALI/inc/DALI.h
MAX_DALI_CHANNELS	Nr. of physical layer interfaces	DALI/inc/DALI.h
MAX_DALI_DEVICES_PER_CHANNEL	Nr. of control gear	DALI/inc/DALI.h
USE_FADE_MR3_INTERRUPT	Fade interpolation using PWM timer	DALI/inc/DALI_Ballast.h
USE_FADE_INTERRUPT	Fade interpolation using system timer	DALI/inc/DALI_Ballast.h
PWM_OUTPUT_INVERTED	inverts PWM outputs	DALI/inc/DALI_Ballast.h
DEFAULT_*	Various default control gear properties	DALI/inc/DALI_Ballast.h

3.10 Building the software

The software tree includes project files for LPCXpresso v4.0.6_151.

When using LPCXpresso for building the DALI slave, use the workspace location C:\nxp\lighting\DALI_SDK_v1.0 as shown in Fig 16.

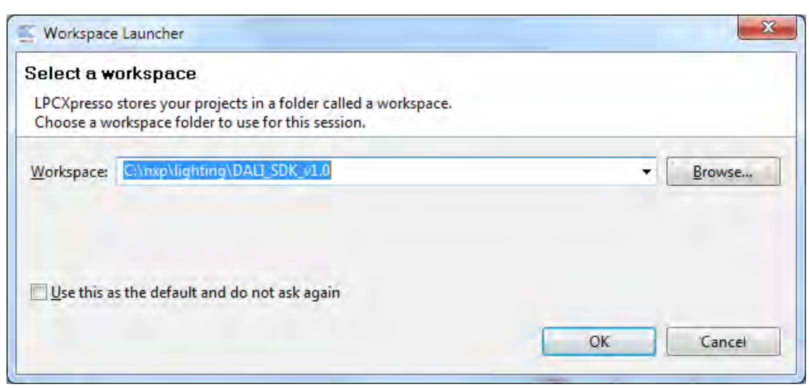


Fig 16. Start LPCXpresso and select the DALI SDK folder as workspace

Use the quick link "*import existing project*" and select as root folder the SDK installation path. Make sure to uncheck the tick mark "*copy projects into workspace*" and select the projects that are shown in Fig 17.

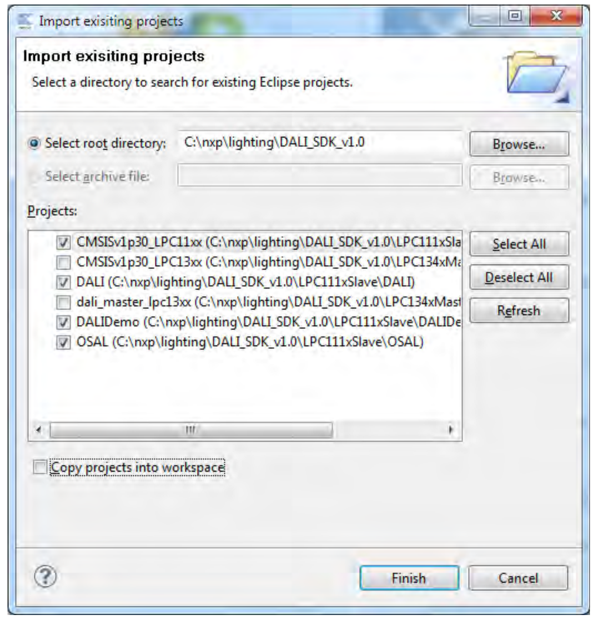


Fig 17. Importing projects for DALI slave

DALIDemo is the example application that uses the *DALI* library project. The *DALIDemo* project is configured to automatically build and include the *DALI*, *OSAL* and *CMSIS* library projects. In release mode the *DALIDemo* project uses the release builds of these projects; in debug mode the debug versions of the libraries are used.

Table 10 shows the size of the firmware in Release mode for different configurations of the software. Note that this does not contain the stack size and the size of the flash sector used for storage of NVM configuration which is 4 kB on the LPC111x device.

Table 10. DALI slave firmware size

Configuration	# control gear	Firmware size	
		Flash (bytes)	Ram (bytes)
CFL	1	9156	276
SSL1523	1	9060	276
UBA3070_1CH	1	9052	276
UBA3070_4CH	4	9860	732
DRIVER_3CH	3	9700	580

Using Table 10 it can be seen that the LPC1112 with 16 kB of flash is sufficient for creating a DALI slave device.

4. Document management

4.1 Abbreviations

Table 11. Abbreviations

Acronym	Description
CMSIS	Cortex Microcontroller Software Interface Standard
CPU	Central Processing Unit
CT	Counter Timer
DALI	Digital Addressable Lighting Interface
GPIO	General Purpose Input/Output
HW	Hardware
IDE	Integrated Development Environment
IRQ	Interrupt Request
LED	Light Emitting Diode
MCU	Micro Controller Unit
NVM	Non Volatile Memory
PC	Personal Computer
PCB	Printed Circuit Board
PIO	Input/Output Pin
PLL	Phase Locked Loop
SW	Software
USB	Universal Serial Bus

4.2 References

Table 12. References

Title		Version	Author	Issue Date
[1]	IEC62386-101: Digital addressable lighting interface, General requirements – system	Edition 1.0 2009-6	IEC	2009
[2]	IEC62386-102: Digital addressable lighting interface, General requirements – control gear	Edition 1.0 2009-6	IEC	2009
[3]	http://www.dali-ag.org			
[4]	UM10398: LPC111x/LPC11Cxx User manual	Rev. 7	NXP	2011

5. Legal information

5.1 Definitions

Draft — The document is a draft version only. The content is still under internal review and subject to formal approval, which may result in modifications or additions. NXP Semiconductors does not give any representations or warranties as to the accuracy or completeness of information included herein and shall have no liability for the consequences of use of such information.

5.2 Disclaimers

Limited warranty and liability — Information in this document is believed to be accurate and reliable. However, NXP Semiconductors does not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information and shall have no liability for the consequences of use of such information. NXP Semiconductors takes no responsibility for the content in this document if provided by an information source outside of NXP Semiconductors.

In no event shall NXP Semiconductors be liable for any indirect, incidental, punitive, special or consequential damages (including - without limitation - lost profits, lost savings, business interruption, costs related to the removal or replacement of any products or rework charges) whether or not such damages are based on tort (including negligence), warranty, breach of contract or any other legal theory.

Notwithstanding any damages that customer might incur for any reason whatsoever, NXP Semiconductors' aggregate and cumulative liability towards customer for the products described herein shall be limited in accordance with the Terms and conditions of commercial sale of NXP Semiconductors.

Right to make changes — NXP Semiconductors reserves the right to make changes to information published in this document, including without limitation specifications and product descriptions, at any time and without notice. This document supersedes and replaces all information supplied prior to the publication hereof.

Suitability for use — NXP Semiconductors products are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of an NXP Semiconductors product can reasonably be expected to result in personal injury, death or severe property or environmental damage. NXP Semiconductors and its suppliers accept no liability for inclusion and/or use of NXP Semiconductors products in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Applications — Applications that are described herein for any of these products are for illustrative purposes only. NXP Semiconductors makes no representation or warranty that such applications will be suitable for the specified use without further testing or modification.

Customers are responsible for the design and operation of their applications and products using NXP Semiconductors products, and NXP Semiconductors accepts no liability for any assistance with applications or customer product design. It is customer's sole responsibility to determine whether the NXP Semiconductors product is suitable and fit for the customer's applications and products planned, as well as for the planned application and use of customer's third party customer(s). Customers should provide appropriate design and operating safeguards to minimize the risks associated with their applications and products.

NXP Semiconductors does not accept any liability related to any default, damage, costs or problem which is based on any weakness or default in the customer's applications or products, or the application or use by customer's third party customer(s). Customer is responsible for doing all necessary testing for the customer's applications and products using NXP Semiconductors products in order to avoid a default of the applications and the products or of the application or use by customer's third party customer(s). NXP does not accept any liability in this respect.

Export control — This document as well as the item(s) described herein may be subject to export control regulations. Export might require a prior authorization from competent authorities.

Evaluation products — This product is provided on an "as is" and "with all faults" basis for evaluation purposes only. NXP Semiconductors, its affiliates and their suppliers expressly disclaim all warranties, whether express, implied or statutory, including but not limited to the implied warranties of non-infringement, merchantability and fitness for a particular purpose. The entire risk as to the quality, or arising out of the use or performance, of this product remains with customer.

In no event shall NXP Semiconductors, its affiliates or their suppliers be liable to customer for any special, indirect, consequential, punitive or incidental damages (including without limitation damages for loss of business, business interruption, loss of use, loss of data or information, and the like) arising out the use of or inability to use the product, whether or not based on tort (including negligence), strict liability, breach of contract, breach of warranty or any other theory, even if advised of the possibility of such damages.

Notwithstanding any damages that customer might incur for any reason whatsoever (including without limitation, all damages referenced above and all direct or general damages), the entire liability of NXP Semiconductors, its affiliates and their suppliers and customer's exclusive remedy for all of the foregoing shall be limited to actual damages incurred by customer based on reasonable reliance up to the greater of the amount actually paid by customer for the product or five dollars (US$5.00). The foregoing limitations, exclusions and disclaimers shall apply to the maximum extent permitted by applicable law, even if any remedy fails of its essential purpose.

5.3 Trademarks

Notice: All referenced brands, product names, service names and trademarks are property of their respective owners.

6. List of figures

7. List of tables

8. Contents

Date of release: 1 March 2012
Document identifier: AN11174