APPLICATION NOTE

Atmel AT03922: DALI Slave with XMEGA E – Software User's Guide

Atmel AVR XMEGA E

Features

- DALI stack compliant with IEC62386 LED modules standard
- DALI data encoding and decoding with USART and XCL modules on Atmel® AVR® XMEGA® E

Introduction

DALI stands for Digital Address Lighting Interface which is lighting control protocol defined in the technical standard IEC62386. The demo software is created to demonstrate DALI slave solution of LED modules based on Atmel AVR XMEGA E device.

This document describes the basic architecture and the application programming interfaces (API) of the demo software. The Atmel AT04022: DALI Slave with XMEGA E – Hardware User Guide application note describes the reference design hardware in detail.

Figure 1. DALI Slave Demo of LED Modules.

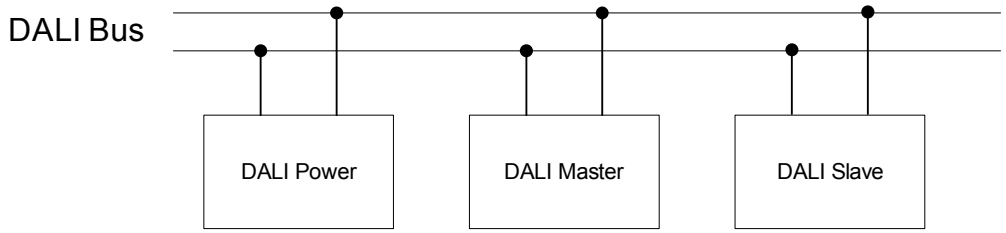
For this reference design, the hardware design files (schematic, BoM and PCB gerber) and software source code can be downloaded from Atmel website. The provided hardware documentation can be used with no limitations to manufacture the reference hardware solution for the design.

Table of Contents

1. Overview

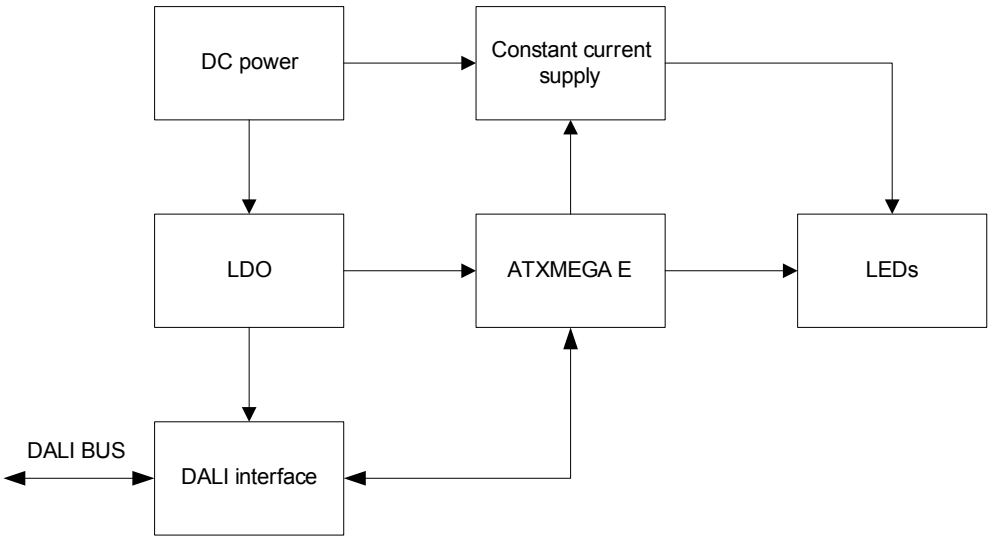
The application demo acts as slave connected to DALI bus. DALI master sends commands and DALI slave responds the commands, especially the lighting control commands. In some cases, the slave sends back information that master inquires. DALI power supplies the DALI bus power. It can be a stand-alone power supply unit or it can be integrated into any master or slave connected to the DALI bus. Figure 1-1 shows the typical DALI master-slave structure connected to the bus.

Figure 1-1. DALI System Structure.



DALI slave receives and transmits the DALI data through the bus. If lighting control commands are received, DALI slave shall adjust the LED lighting through the PWM output. It also contains constant current supply part to support the LEDs.

Figure 1-2. DALI Slave with XMEGA E Block Diagram.

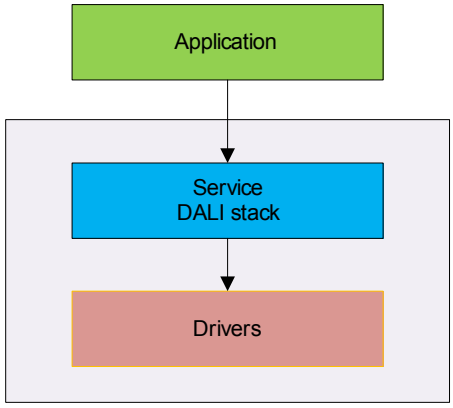


2. Software Architecture

The software is developed based on layer structure as Atmel ASF that makes it more convenient to port between different Atmel device platforms. The software is composed of three layer parts as below:

- Application
- Service
- Drivers

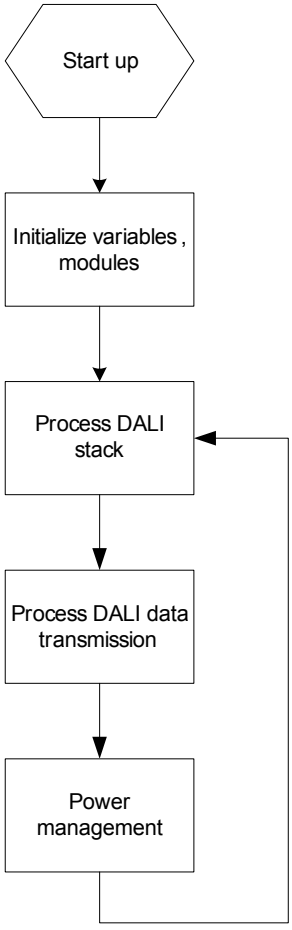
Figure 2-1. DALI Slave Software Architecture.



2.1 Application

The whole system runs in an infinite loop. After start up, it initializes the parameters and configures microcontroller modules. In main loop the system processes DALI stack, encoding and decoding DALI data sequentially. Then it goes into sleep mode. After woken up by interrupt, it will run again.

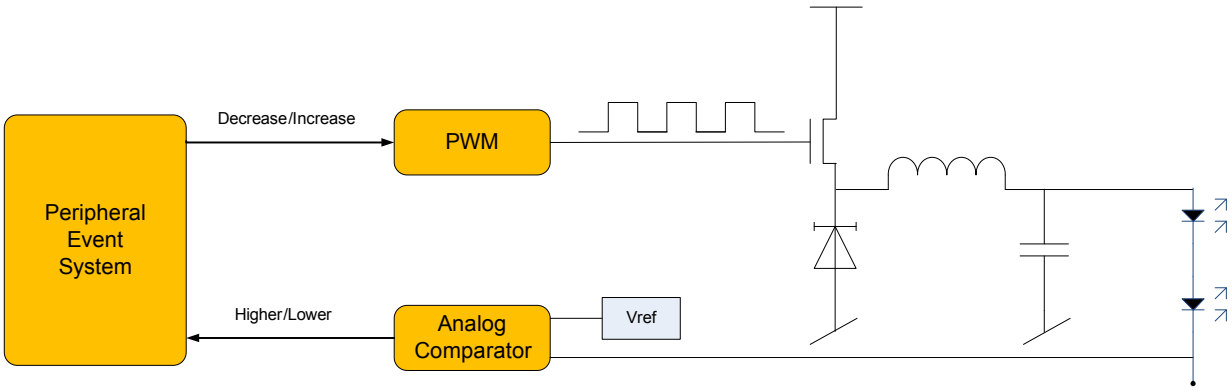
Figure 2-2. Application Flowchart.



DALI data encoding and decoding are implemented by XMEGA E USART and XCL modules. For details of this process refer to the application note Atmel AT03335: Manchester Transceiver Using the USART and XCL Modules on XMEGA E.

Microcontroller hardware sources are configured to implement constant current source for LED lighting. Analog comparator compares the input voltage with internal constant voltage and is configured as event source. Event resulted from analog comparator acts on timer PWM output via fault extension. PWM output switches the hardware buck transformer. In this way constant voltage can be ensured at analog comparator input point and then constant current source is generated. For hardware design details, refer to the application schematic.

Figure 2-3. Buck LED Driver.



2.2 Service

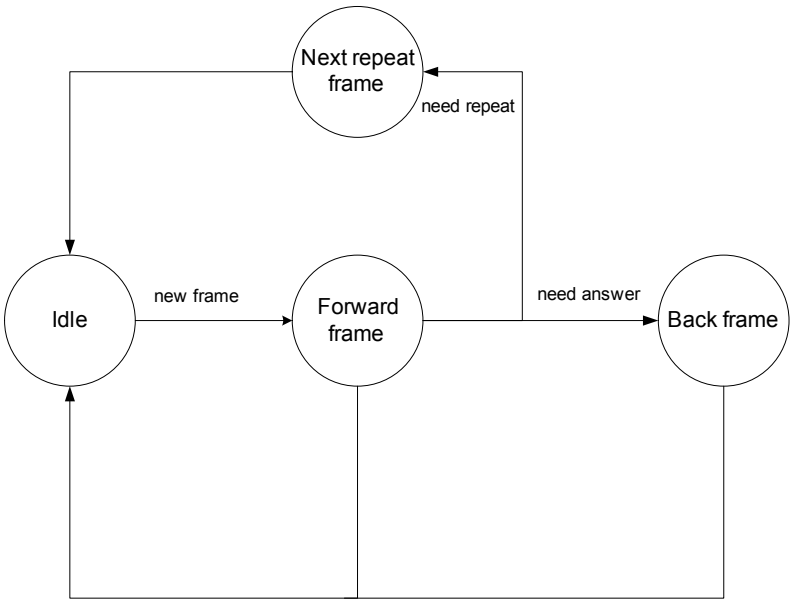
Service realizes the DALI slave stack. It processes frame sequence timing and implements DALI commands.

State machine is used to process DALI frame. Firstly DALI slave goes into idle state and implements state initialization. After this, the slave shall go to forward frame state immediately and wait for DALI frame arrival. If DALI frame is received, it should be processed. Then according to frame information, the state machine goes to next repeat state, backward state or idle state.

In DALI standard definition, a configuration command shall be received twice within 100ms to reduce the probability of incorrect reception. If received correctly twice, the configuration command should be executed and then DALI slave goes to idle state. Otherwise the DALI slave goes into idle state.

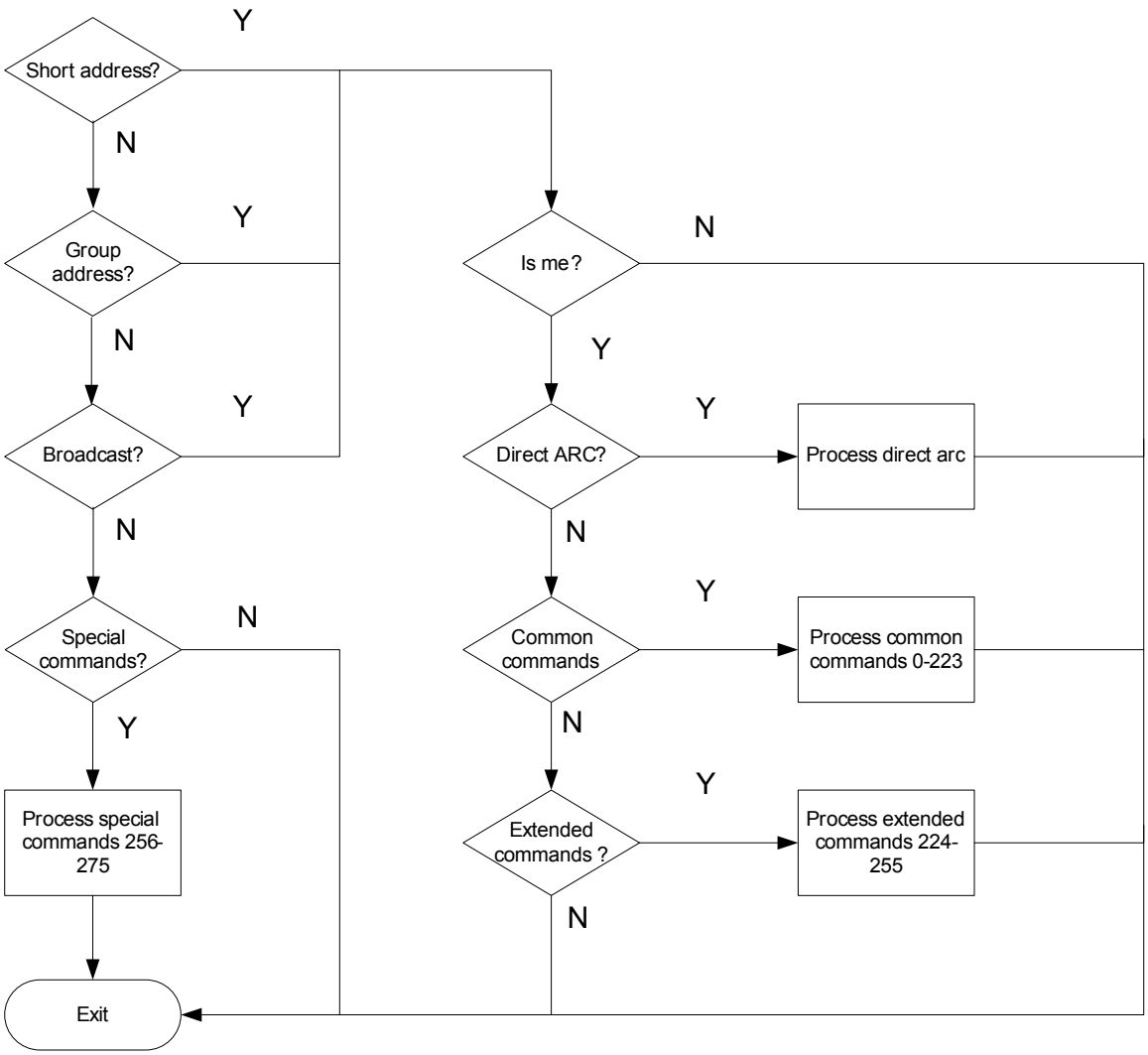
A backward frame shall only be sent after the reception of a query command or a write memory command. Depending on the command the backward frame shall be either a ‘Yes’ / ‘No’ or 8-bit information.If backward frame is ‘No’, the DALI slave should not react (idle line). Otherwise DALI slave should send the answer to master and after this it shall go to idle mode.

Figure 2-4. Frame State Transition.



After DALI slave receives a forward frame, it should decode the two bytes of address and data. The address type includes short address, group address, broadcast or special commands. Address and data compose the command code. For every command code, the corresponding DALI function should be executed.

Figure 2-5. Commands Process Flowchart.



2.3 Drivers

The software utilizes the drivers in the ASF (Atmel Software Framework), which is integrated in Atmel Studio. The drivers include TC, USART, XCL, EDMA, etc.

3. Main API Introduction

3.1 Application Layer API Introduction

Table 3-1. Application Layer File Names.

Source file	Header file
main.c	dali_top.h
dali_ac.c	dali_ac.h
dali_bit.c	dali_bit.h
dali_tc.c	dali_tc.h

3.1.1 Main.c file

- main() function

  The entry point of the application is the main( ) function. The main( ) function must contain two mandatory parts. One initializes the system variables and configures microcontroller modules. The other implements DALI slave function in the infinite loop.

  The main( ) function implementation shows as below:

```
int main(void)
{
  Initialization code;
  while (1) {
     DALI slave implementation code;
  }
}
```

3.1.2 Dali_ac.c file

- dali_ac_init( ) function
  Initialize analog comparator module as part of buck transformer.

3.1.3 Dali_bit.c file

- dali_bit_init( ) function
  Initialze the USART and XCL modules for encoding and decoding DALI data.
- dali_bits_tx( ) function
  This function transmits DALI data depending request from service.
- dali_bits_rx( ) function
  This function receives DALI data from EDMA channel linked with USART module. After received, the data should be transferred to service.

3.1.4 Dali_tc.c file

- dali_tc_init() function
  This function initializes TCC4, TCC5 and TCD5 modules. TCC4 is used for PWM output and timer of LED dimming. TCC5 serves as switch of buck transformer with PWM output. TCD5 is used for DALI frame sequence timing.

3.2 Service Layer API Introduction

Table 3-2. Service Layer File Names.

Source file	Header file
dali_frame.c	dali_frame.h
dali_cmd.c	dali_cmd.h

3.2.1 Dali_frame.c file

- dali_frame_init( ) function
  This function initializes the variables used in DALI stack.
- dali_frame_process_state( ) function
  DALI frame timing is implemented through state machine.

3.2.2 Dali_cmd.c file

- dali_cmd_direct_arc_power_control( ) function
  This function implements DALI direct arc power control command.
- dali_cmd_process_common( ) function
  This function processes the DALI common commands from 0 to 223.
- dali_cmd_process_extended_application( ) function
  This function processes the DALI application extended commands from 224 to 255. Commands 224 to 254 are defined in part 207 of IEC62386 standard. Optional features in command 240 are not supported here.
- dali_cmd_process_special( ) function
  This function processes the DALI special commands from 256 to 275.

4. Software Package Content

The directory structure of the software package is shown as follows:

```
├─ASF
│ ├─common
│ │ ├─boards
│ │ │ └─user_board
│ │ ├─services
│ │ │ ├─clock
│ │ │ │ └─xmega
│ │ │ ├─gpio
│ │ │ │ └─xmega_gpio
│ │ │ ├─ioport
│ │ │ │ └─xmega
│ │ │ └─sleepmgr
│ │ │   └─xmega
│ │ └─utils
│ │   ├─interrupt
│ │   └─make
│ └─xmega
│   ├─drivers
│   │ ├─ac
│   │ ├─cpu
│   │ ├─edma
│   │ ├─nvm
```

```
│   │   ├─pmic
│   │   ├─sleep
│   │   ├─tc45
│   │   ├─usart
│   │   └─xcl
│   │       └─docsrc
│   └─utils
│       ├─assembler
│       ├─bit_handling
│       └─preprocessor
├─config
└─dali
    ├─dali_ac
    ├─dali_bit
    ├─dali_lib
    │   ├─dali_command
    │   ├─dali_frame
    │   └─dali_hal
    └─dali_timer
```

- ASF:
  In this folder Atmel software framework provides the module drivers of microcontroller.
- config:
  Provide board, clock, EDMA and USART configuration files.
- dali:
  Provide the DALI application and service files.
  - dali_ac, dali_bit, dali_timer
    In these folders, provide corresponding application files.
  - dali_lib
    DALI stack is available in this folder.
    - dali_frame
      Provide the DALI frame process files.
    - dali_cmd
      Provide DALI command implementation files.
    - dali_hal
      Hardware abstraction layer, including a complete set of APIs for using hardware resources by DALI stack that is convenient for rapid design-in and smooth integration with varied peripherals.

5. Memory Usage

Building with Atmel Studio 6.1 and compiler optimization level O1, the XMEGA E5 memory usage is:

13846 bytes of CODE memory;

144 bytes of DATA memory;

453 bytes of EEPROM memory.

With compiler optimization level Os, the CODE memory reduces to 11882 bytes.

6. Conclusion

This document shows the implementation of DALI slave taking advantage of XMEGA E resources. It describes the basic software architecture, flowcharts and the application programming interfaces (API), gives a description of the software package contents. Memory usage illustrates how much RAM, EEPROM and Flash this demo application consumes.

Appendix A. DALI Command Set

Command number	Command name
–	DIRECT ARC POWER CONTROL
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
10 – 15	Reserved
16 – 31	GO TO SCENE
32	RESET
33	STORE ACTUAL LEVEL IN THE DTR
34 – 41	Reserved
42	STORE THE DTR AS MAX LEVEL
43	STORE THE DTR AS MIN LEVEL
44	STORE THE DTR AS SYSTEM FAILURE LEVEL
45	STORE THE DTR AS POWER ON LEVEL
46	STORE THE DTR AS FADE TIME
47	STORE THE DTR AS FADE RATE
48 – 63	Reserved
64 – 79	STORE THE DTR AS SCENE
80 – 95	REMOVE FROM SCENE
96 – 111	ADD TO GROUP
112 – 127	REMOVE FROM GROUP
128	STORE DTR AS SHORT ADDRESS
129	ENABLE WRITE MEMORY
130 – 143	Reserved
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 ADDRESS
151	QUERY VERSION NUMBER
152	QUERY CONTENT DTR

153	QUERY DEVICE TYPE
154	QUERY PHYSICAL MINIMUM LEVEL
155	QUERY POWER FAILURE
156	QUERY CONTENT DTR1
157	QUERY CONTENT DTR2
158 – 159	Reserved
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/FADE RATE
166 – 175	Reserved
176 – 191	QUERY SCENE LEVEL (SCENES 0-15)
192	QUERY GROUPS 0-7
193	QUERY GROUPS 8-15
194	QUERY RANDOM ADDRESS (H)
195	QUERY RANDOM ADDRESS (M)
196	QUERY RANDOM ADDRESS (L)
197	READ MEMORY LOCATION
198 – 223	Reserved
224 – 254	See parts 207 of DALI Standard
255	QUERY EXTENDED VERSION NUMBER
256	TERMINATE
257	DATA TRANSFER REGISTER (DTR)
258	INITIALISE
259	RANDOMISE
260	COMPARE
261	WITHDRAW
262 – 263	Reserved
264	SEARCHADDRH
265	SEARCHADDRM
266	SEARCHADDRL
267	PROGRAM SHORT ADDRESS
268	VERIFY SHORT ADDRESS
269	QUERY SHORT ADDRESS
270	PHYSICAL SELECTION
271	Reserved
272	ENABLE DEVICE TYPE X
273	DATA TRANSFER REGISTER 1 (DTR1)
274	DATA TRANSFER REGISTER 2 (DTR2)

275	WRITE MEMORY LOCATION
276 – 349	Reserved

Appendix B. Revision History

Doc. Rev.	Date	Comments
42177B	01/2014	Updated memory usage as source code is modified
42177A	08/2013	Initial document release

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation
1600 Technology Drive
San Jose, CA 95110
USA
Tel: (+1)(408) 441-0311
Fax: (+1)(408) 487-2600
www.atmel.com

Atmel Asia Limited
Unit 01-5 & 16, 19F
BEA Tower, Millennium City 5
418 Kwun Tong Road
Kwun Tong, Kowloon
HONG KONG
Tel: (+852) 2245-6100
Fax: (+852) 2722-1369

Atmel Munich GmbH
Business Campus
Parkring 4
D-85748 Garching b. Munich
GERMANY
Tel: (+49) 89-31970-0
Fax: (+49) 89-3194621

Atmel Japan G.K.
16F Shin-Osaki Kangyo Building
1-6-4 Osaki, Shinagawa-ku
Tokyo 141-0032
JAPAN
Tel: (+81)(3) 6417-0300
Fax: (+81)(3) 6417-0370