Digital Addressable Lighting Interface (DALI) control gear

XMC™ microcontrollers

July 2016

Agenda

1. Key features
2. Specification
3. System block diagram
4. Hardware overview
5. Software overview
6. Highlight MCU features
7. Hands-on training

DALI control gear
Key features

Target Application

- DALI control gear (DALI102)

Key Features

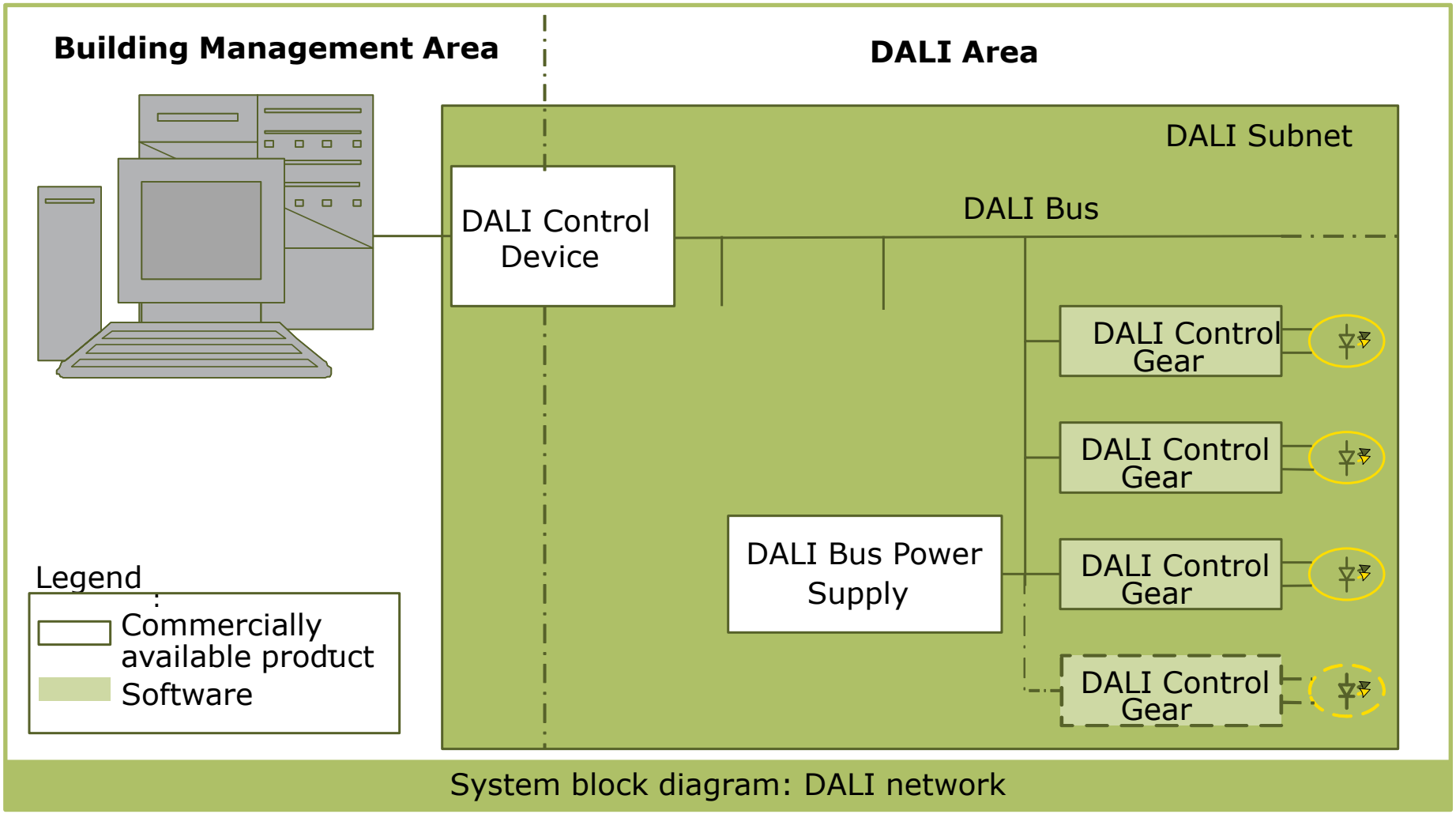
- Up to 64 addressable lighting devices in a subnet
- Standard 2-core cable (1.5 mm^2)
- Free polarity wiring
- Free wiring topology
- Single bus for power and data
- Up to 16 groups in a subnet
- Up to 16 scenes per lighting device
- 349 commands
- 2-way communication
- Changes/configurations via software

DALI control gear
Key features

Specifications

- Typical DALI bus voltage: 16 V (max: 22.5 V)
- Max current supplied to DALI bus: 250 mA
- Max drop between any 2 devices on DALI bus: 2 V
- Max cable length (at 1.5 mm^2): 300 m

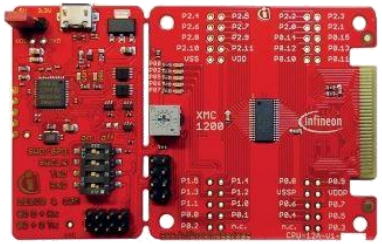
DALI control gear
System block diagram



System block diagram: DALI network

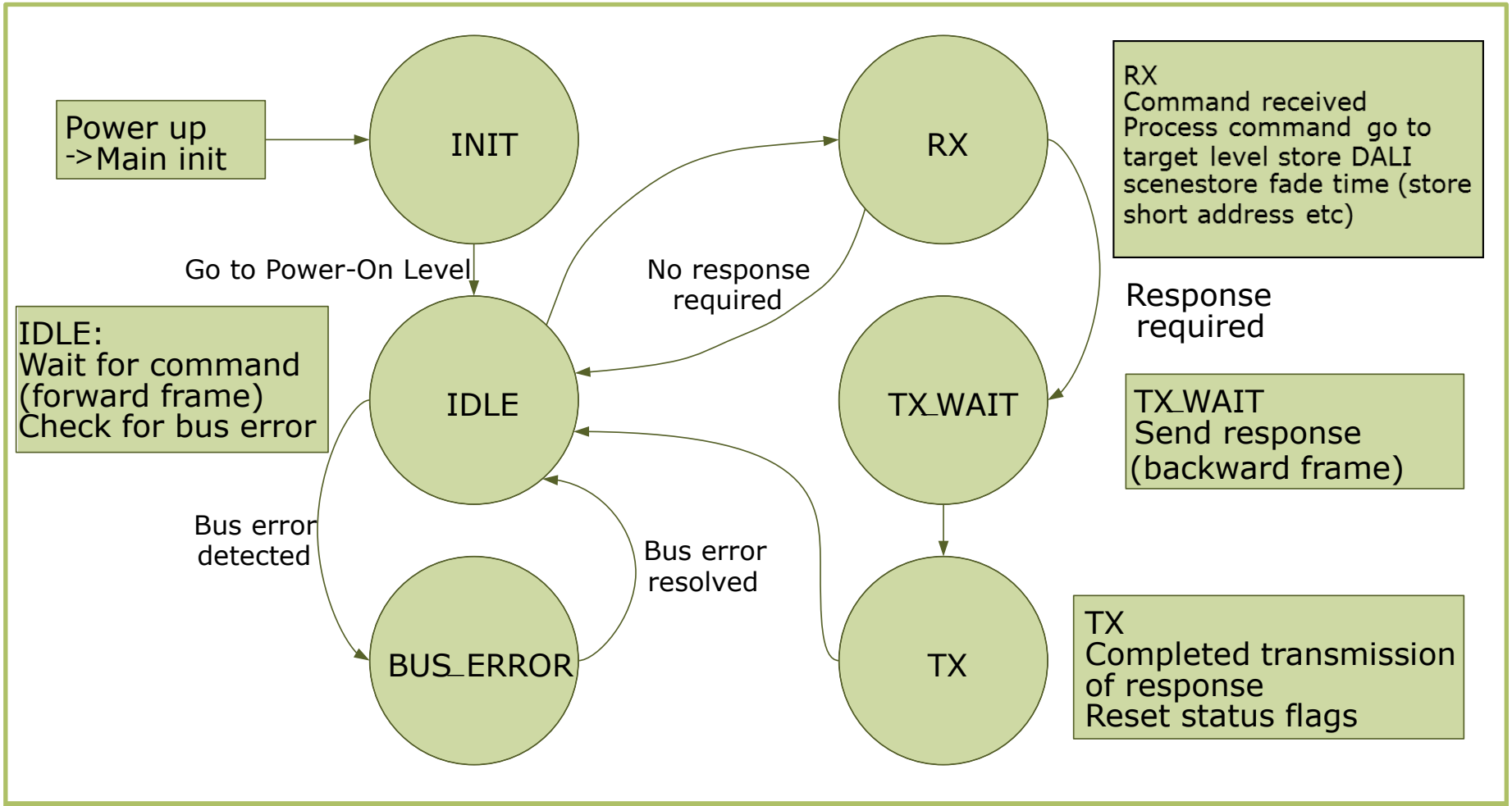
DALI control gear
Hardware overview

- XMC1000 LED Lighting Application Kit comprising of
  - XMC1200 Boot Kit
  - White LED or Color LED card



- Kit schematics, documentation
  - http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=db3a30443ba77cfd013baec9c7880ca9

DALI control gear
Software overview



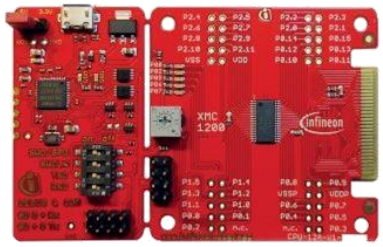
Flow chart: DALI control gear – software overview

DALI control gear
Highlight MCU features

- BCCU
  - Dimming along an exponential curve with adjustable dimming time
  - 12-bit dimming level for smooth and natural dimming
  - Dimming engine performs dimming automatically without CPU load
  - Up to 9 channels: convenient for driving multi-channel lamps
  - Separate dimming and color control: dimming level can be adjusted while preserving color output naturally, vice versa
- CCU4
  - Capture mode readily detects rising and falling edges in forward frame
  - Capture timer provides convenient way of measuring time lapse between edges
- PRNG
  - Generates high quality random data quickly for DALI random addresses

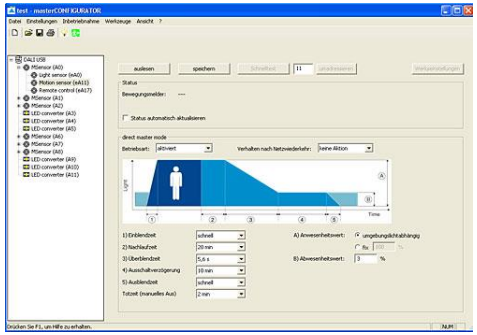
DALI control gear
Hands-on training

- Control Gear
  - XMC1200 Bootkit + White LED Card



- Control Device
  - Tridonic DALI USB + DALI PS1 + masterCONFIGURATOR + DALImonitor

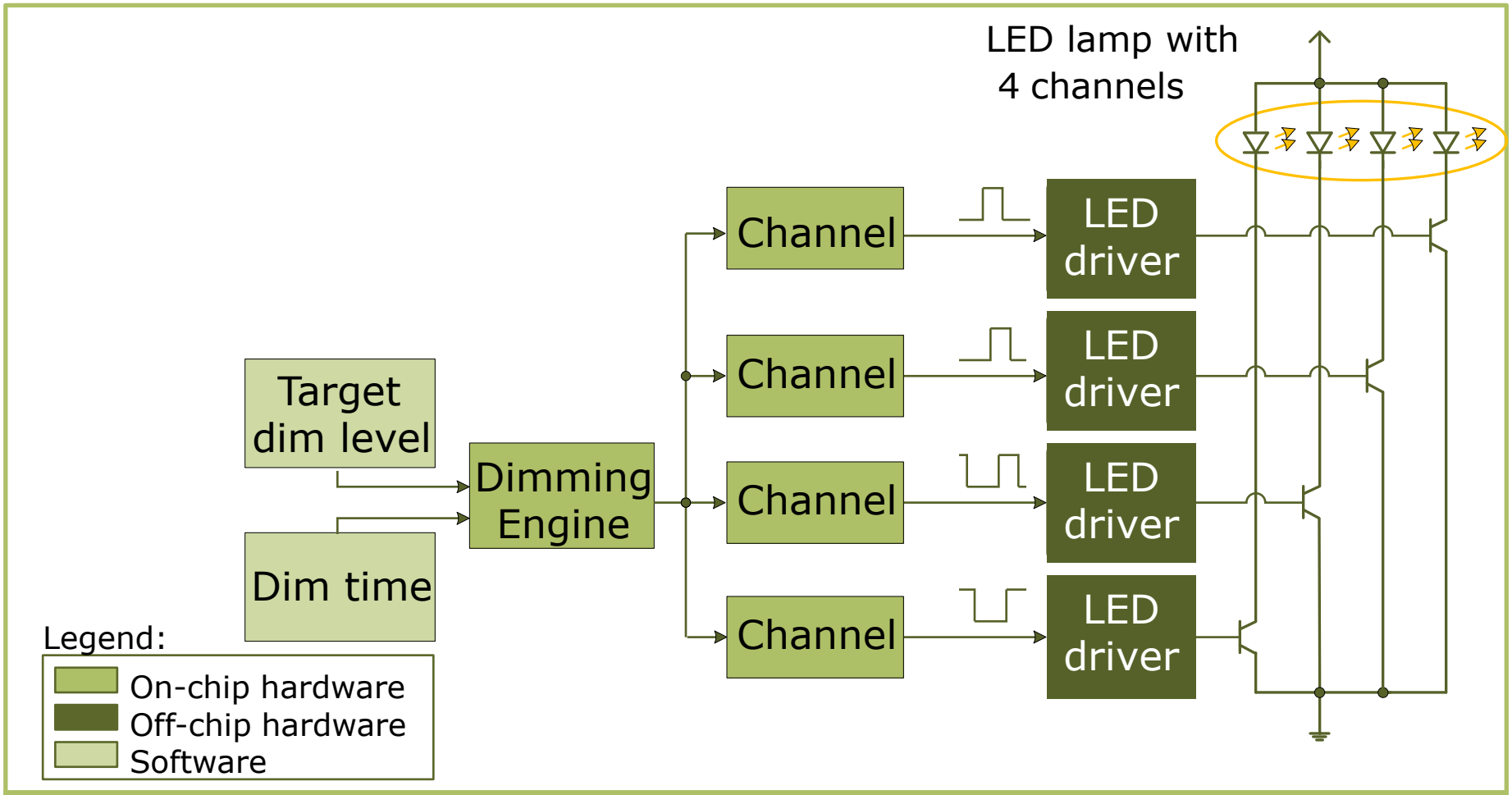
Source: http://www.tridonic.com



- HOTs in coming slides shall provide a step-by-step guide to set up demo

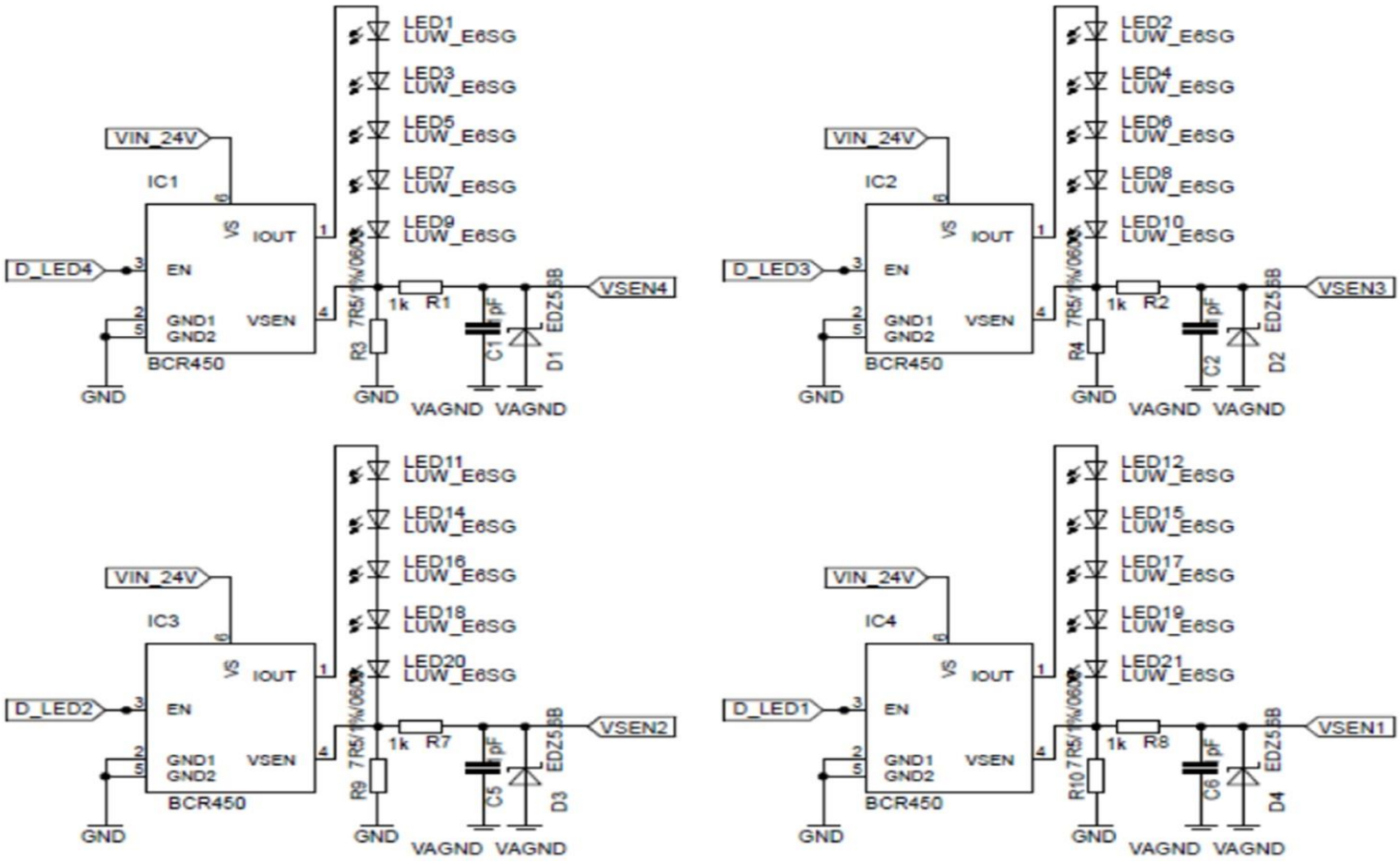
HOT1: 4-Channel Lamp

DALI control gear
4-Channel lamp - block diagram



Block diagram: DALI control gear – 4-Channel lamp

DALI control gear
4-Channel lamp – board schematics



Schematic: DALI control gear – 4-Channel lamp

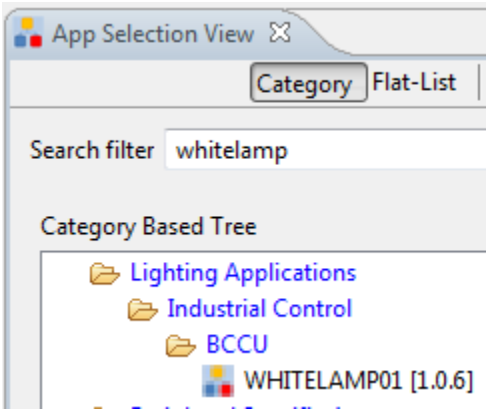
DALI control gear
4-Channel lamp: Software, list of DAVE™ APP

- WHITELAMP01

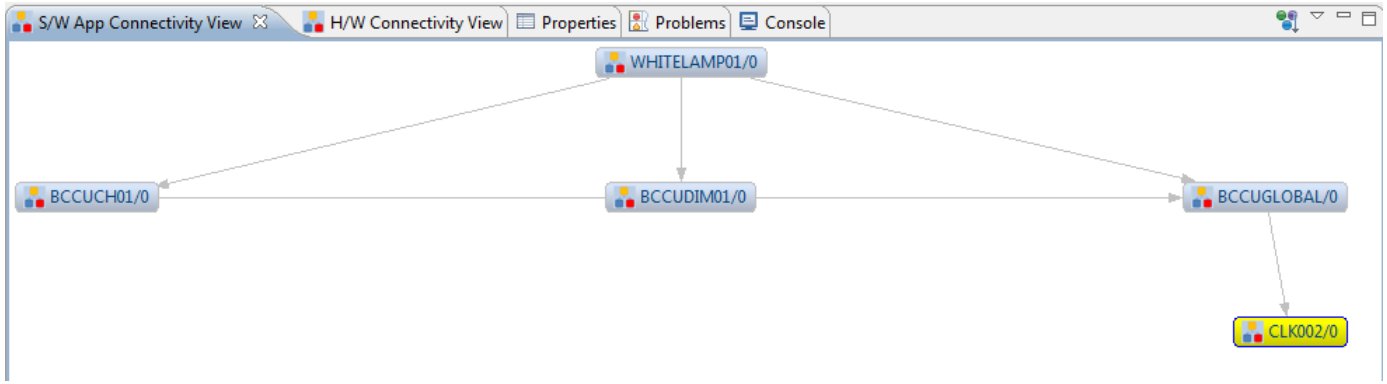
  Aggregates BCCUGLOBAL, BCCUDIM01 and BCCUCH01 APPs. Provides configurations and dimming control for multi-channel white LED lamp.

DALI control gear
4-Channel lamp – HOT (1/8)

- Select WHITELAMP01 from the App Selection View Window and add to project

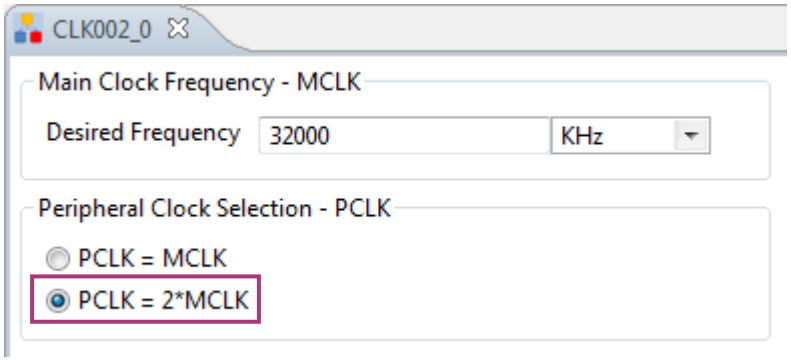


- Double-click on CLK002 APP in S/W App Connectivity View to open the UI Editor

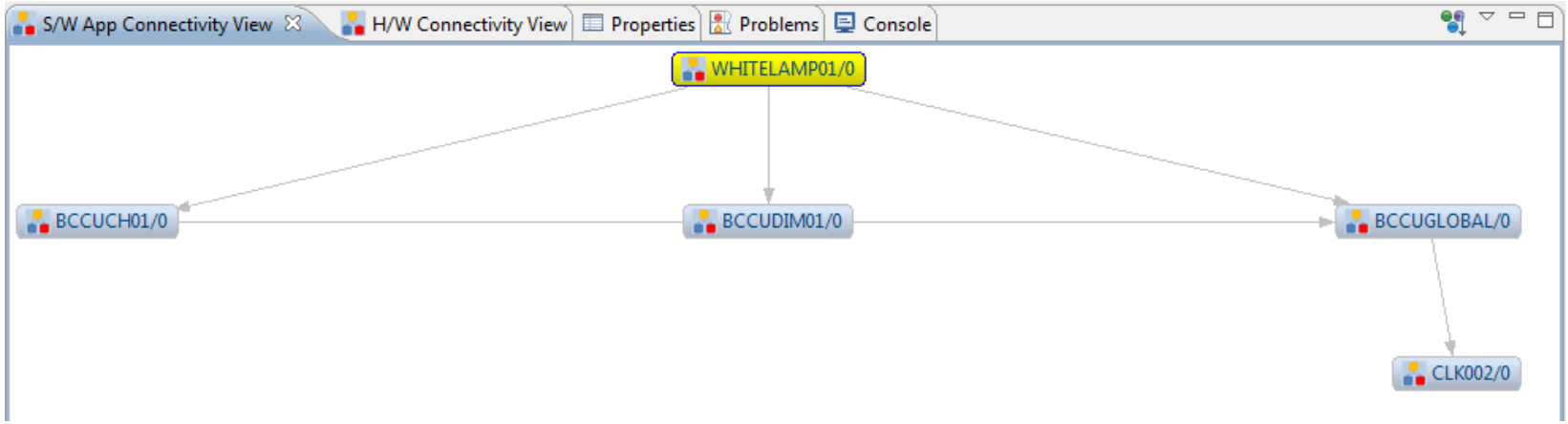


DALI control gear
4-Channel lamp – HOT (2/8)

› Select *PCLK = 2*MCLK*

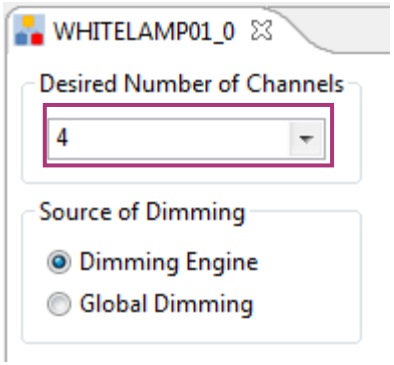


› Double-click on WHITELAMP01 APP in S/W App Connectivity View to open the UI Editor

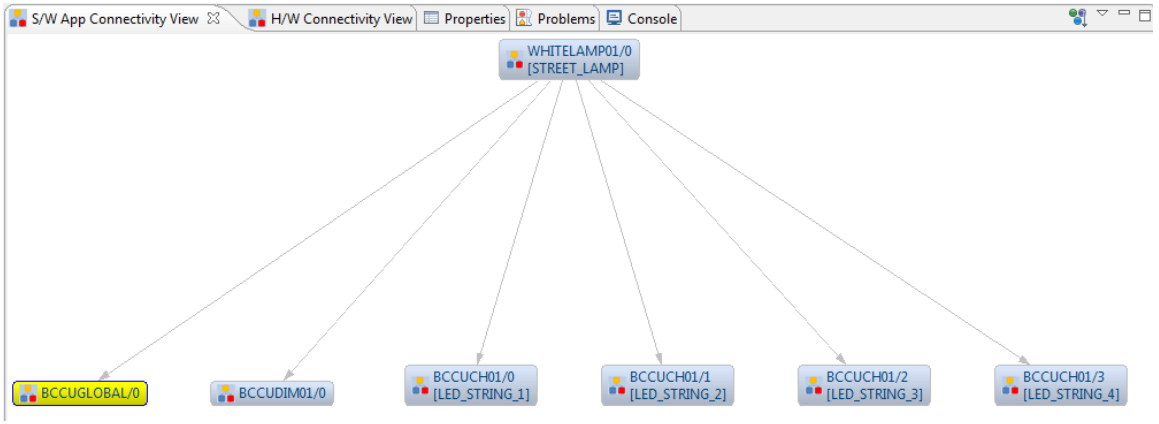


DALI control gear
4-Channel lamp – HOT (3/8)

› Configure *Desired Number of Channels* to 4

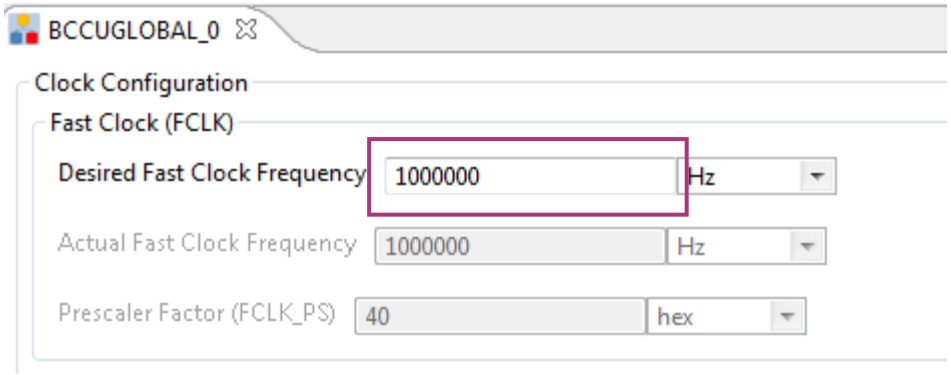


› Double-click on BCCUGLOBAL APP in S/W App Connectivity View to open the UI Editor

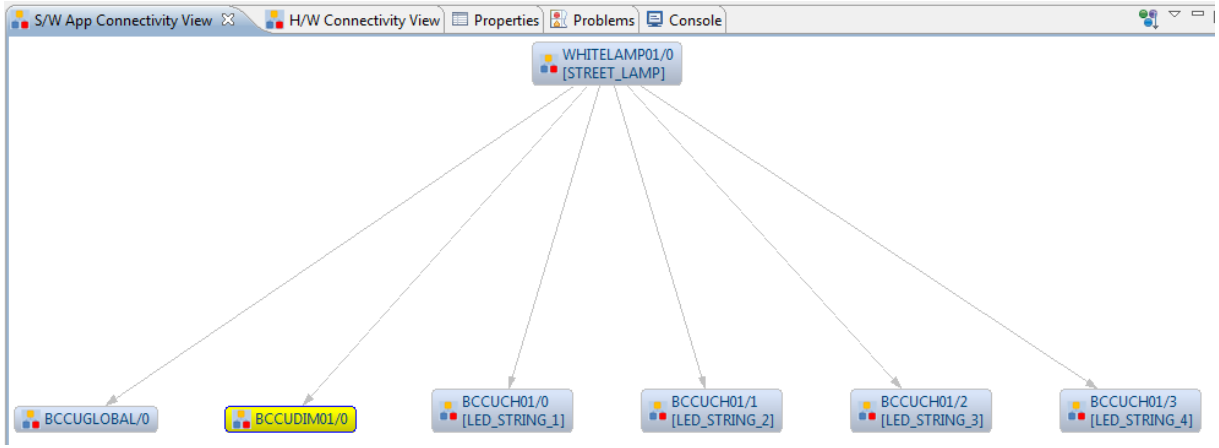


DALI control gear
4-Channel lamp – HOT (4/8)

› Configure *Desired Fast Clock Frequency* to 1 MHz

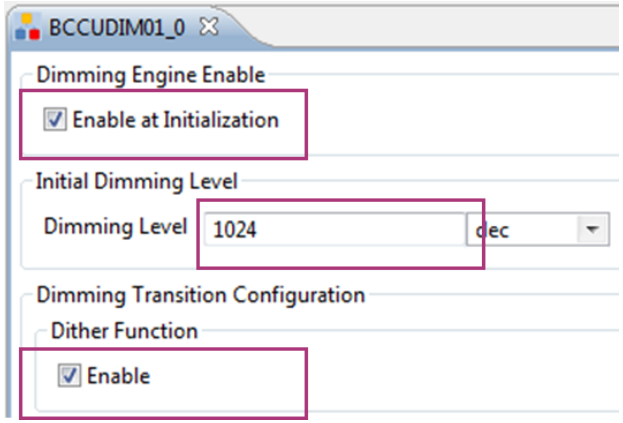


› Double-click on BCCUDIM01 APP in S/W App Connectivity View to open the UI Editor

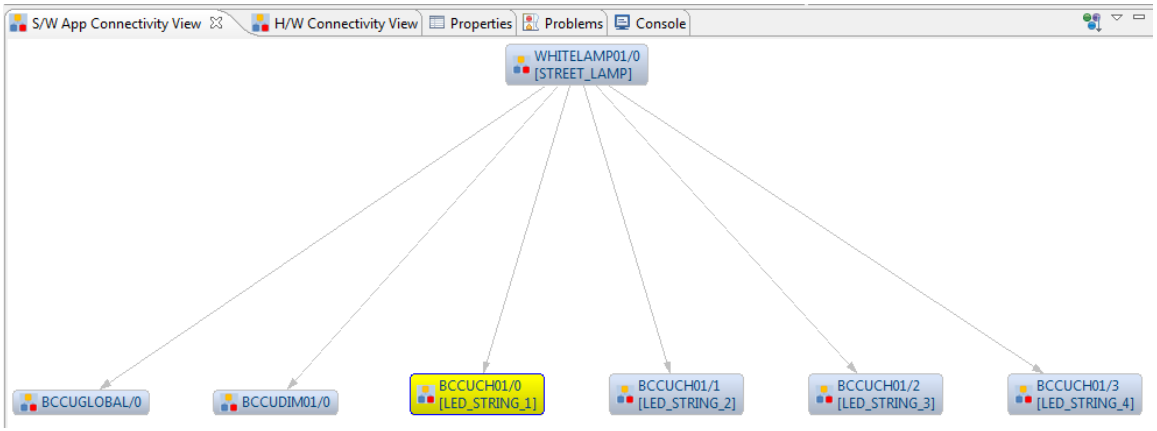


DALI control gear
4-Channel lamp – HOT (5/8)

- Enable dimming engine at initialization
- Configure *Dimming Level* to 1024
- Enable dither function

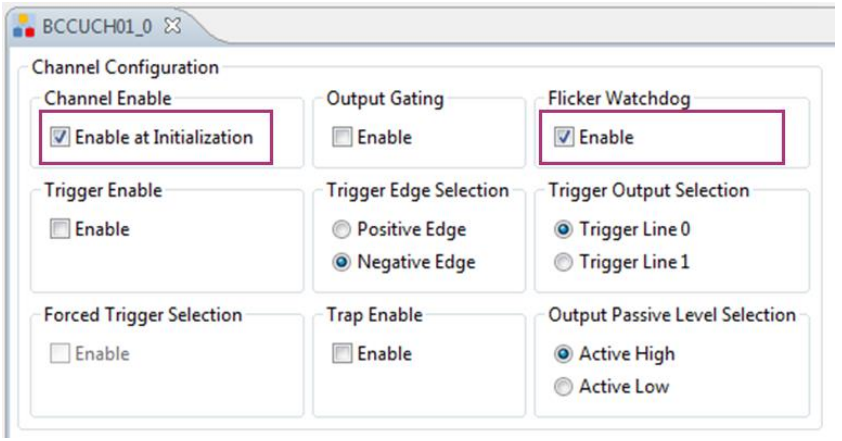


- Double-click on BCCUCH01 APP in S/W App Connectivity View to open the UI Editor

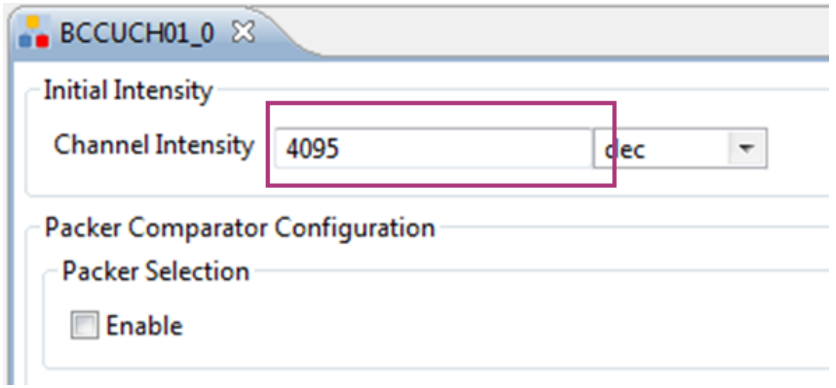


DALI control gear
4-Channel Lamp – HOT (6/8)

- Under *Channel Configuration tab,*
  - Enable channel at initialization
  - Enable flicker watchdog

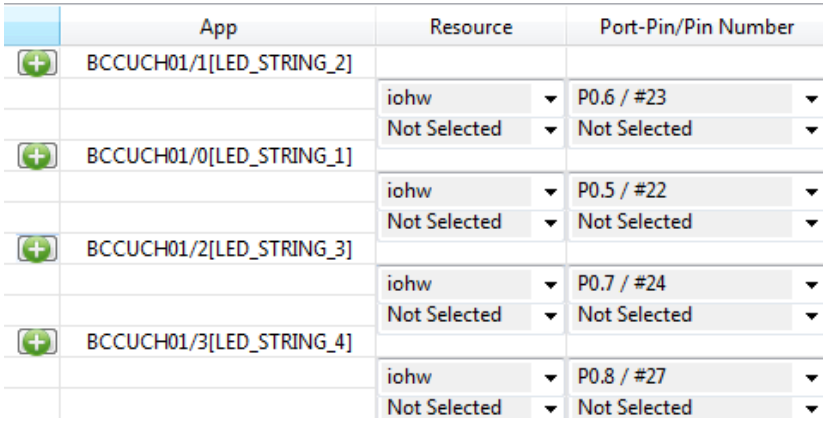


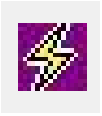
- Under *Intensity and Packer Configuration* tab,
  - Configure *Channel Intensity* to 4095
- Repeat above steps for the other 3 BCCUCH01 APPs



DALI control gear
4-Channel lamp – HOT (7/8)

- Open “Manual Pin Assignment” window by clicking on the shortcut button 



- Assign the pins accordingly:
  - BCCUCH01/0/1/2/3: P0.5/ P0.6/P0.7/P0.8
- Click “Solve and Save”
- Click “Close” 
- Generate code

DALI control gear
4-Channel lamp – HOT (8/8)

- In Main.c, add code to dim to 100 % brightness in 7 s
  - 100 % brightness (Dimlevel = 4095)
  - 7 s dim time (DimDiv = 100)
  - WHITELAMP01_SetDimLevelExponential (&WHITELAMP01_Cfghandle0,4095U,100U,0xDB);

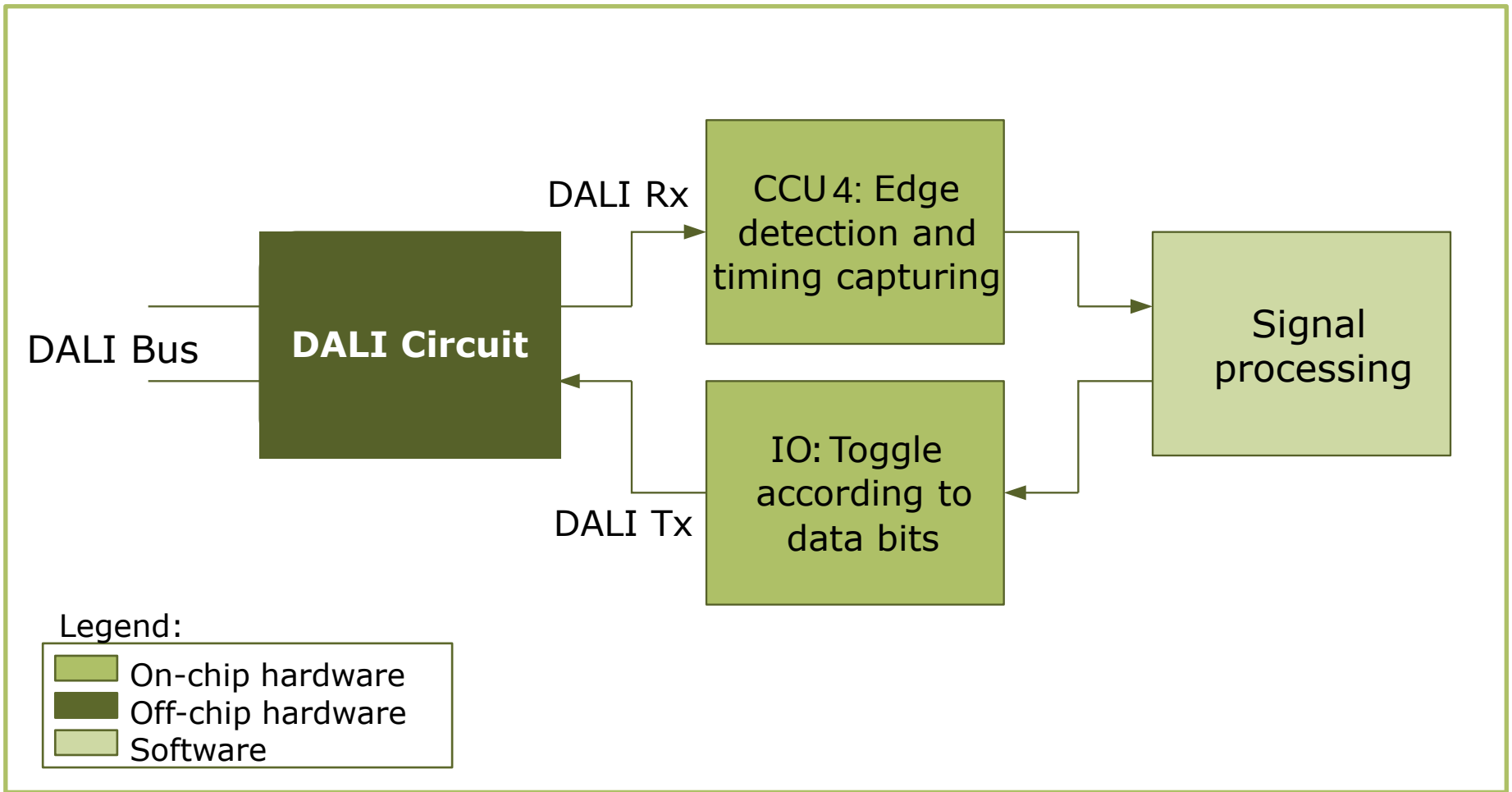
Main.c

```
int main(void)
{
//  status_t status;        // Declaration of return variable for DAVE3 APIs (toggle

    DAVE_Init();            // Initialization of DAVE Apps

    WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0, 4095U, 100U, 0xDB);
```

- Build project 
- Connect XMC1200 Boot kit to PC
- Download code 
- Start code 
- Observe LEDs on White LED card fade up

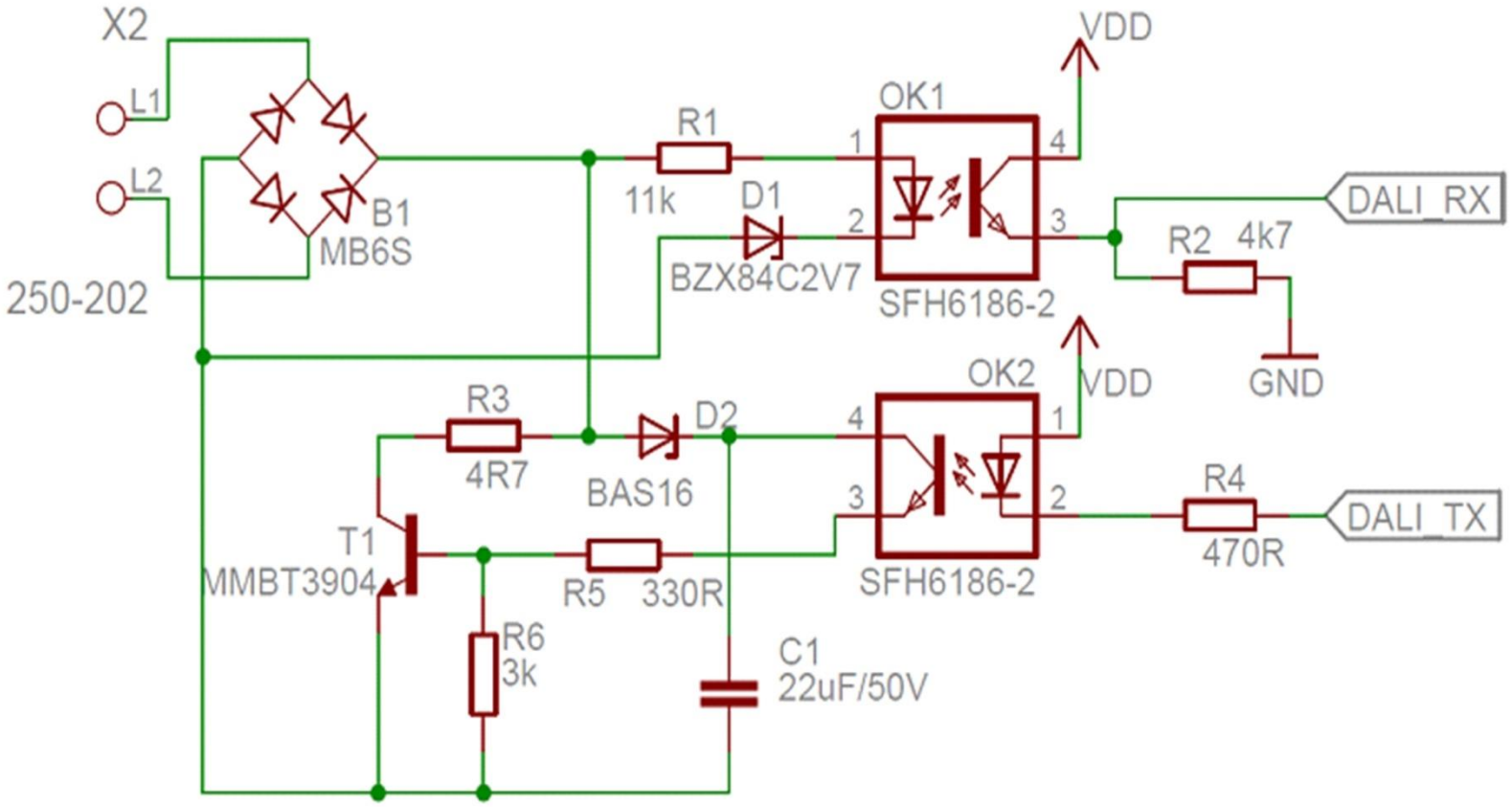
HOT2– DALI communication

DALI control gear
DALI communication – block diagram



Block diagram: DALI control gear – DALI communication

DALI control gear
DALI communication – board schematics



Schematic: DALI control gear – DALI communication

DALI control gear - DALI communication: Software, list of DAVE™ APP

- DALICG02

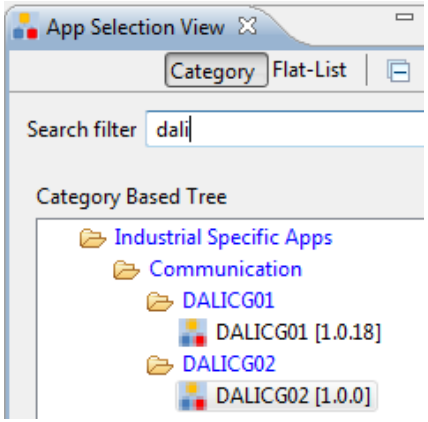
  Software stack for DALI standard -102 (Control gear)

- MANC01

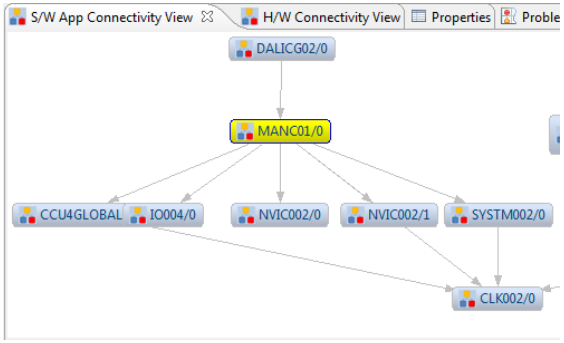
  Detects incoming forward frame, processes received data and transmits forward frame when requested

DALI control gear
DALI communication – HOT (1/6)

- Select DALICG02 from the App Selection View Window and add to project

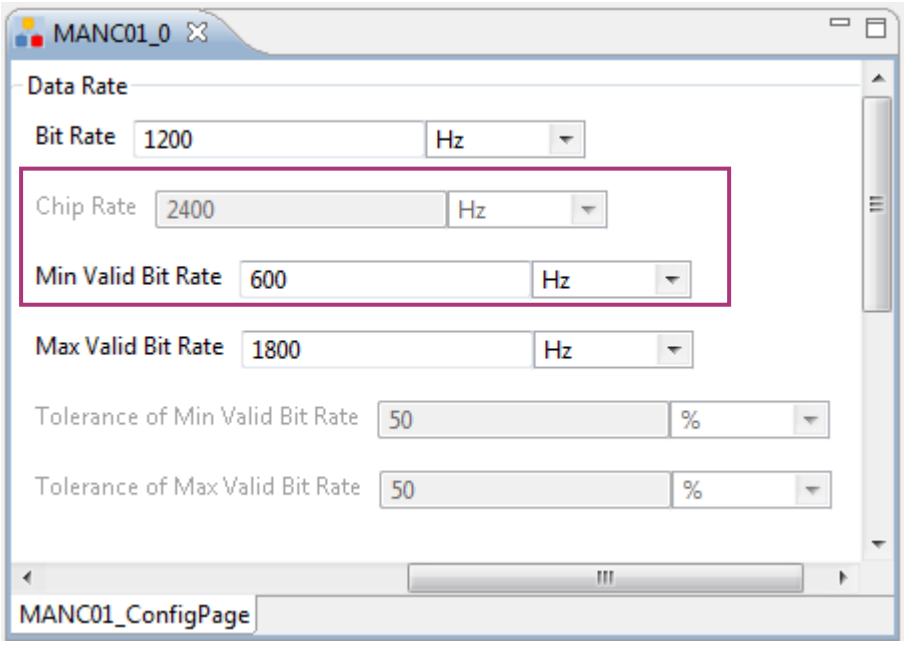


- DALICG02 aggregates MANC01 APP. Double-click on MANC01 APP in S/W App Connectivity View to open the UI Editor



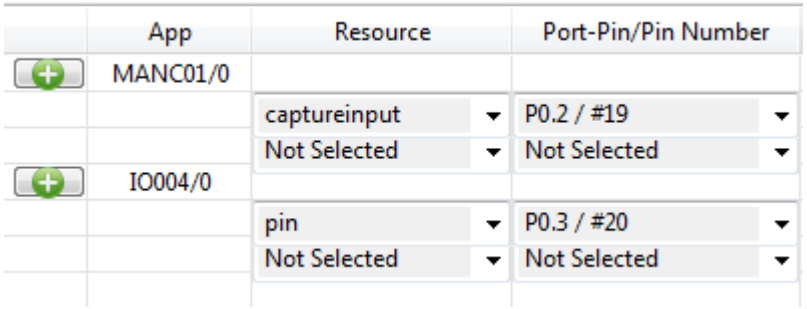
DALI control gear
DALI communication – HOT (2/6)

- Configure *Min Valid Bit Rate* and *Max Valid Bit Rate* to 600 Hz and 1800 Hz respectively



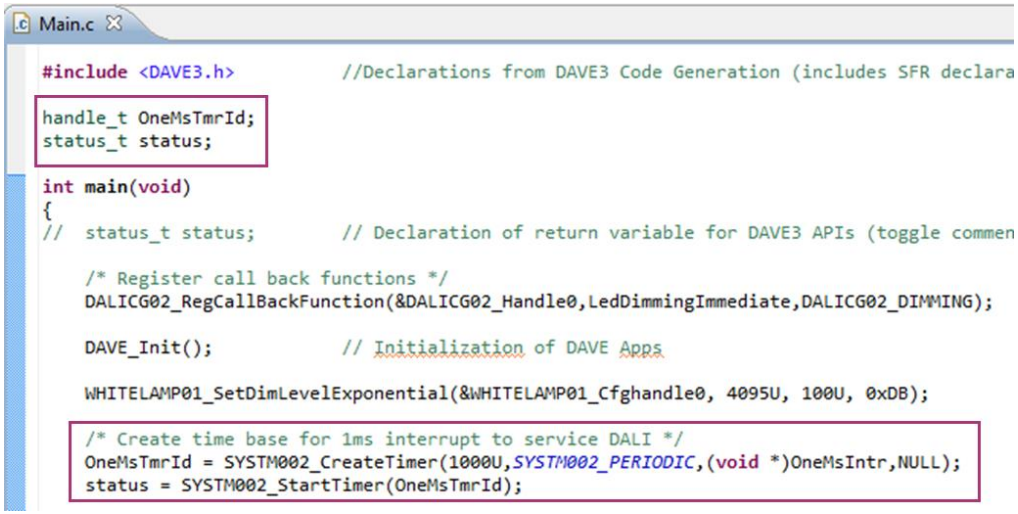
DALI control gear
DALI communication – HOT (3/6)

- Open “Manual Pin Assignment” window by clicking on the shortcut button 
- Assign the pins accordingly:
  - DALI Rx: P0.2
  - DALI Tx: P0.3
- Click “Solve and Save”
- Click “Close”
- Generate code 



DALI control gear
DALI communication – HOT (4/6)

- Next, create a software timer for a 1 ms periodic interrupt



- This interrupt is required for the following task:
  - Maintaining DALI communication by periodically calling DALICG02 API *DALICG02_MainThread()*

DALI control gear
DALI communication – HOT (5/6)

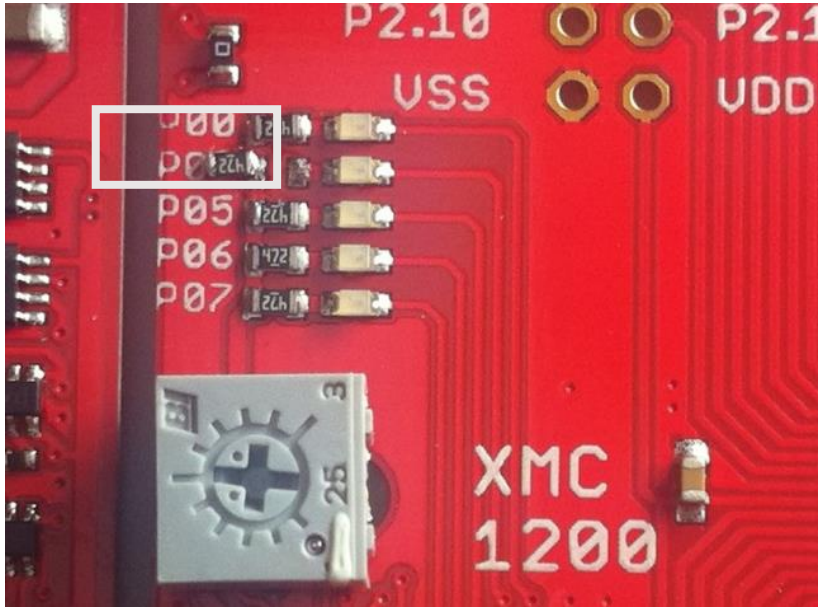
› Define the interrupt service routine for the periodic interrupt

Main.c

```
handle_t OneMsTmrId;
status_t status;

/* Service Routine for 1ms Periodic SW Timer Interrupt */
void OneMsIntr(void)
{
    uint32_t bccu_fade_status;

    /* Stop the timer */
    status = SYSTM002_StopTimer(OneMsTmrId);

    /* Call DALICG02_MainThread to maintain DALI communication*/
    DALICG02_MainThread(&DALICG02_Handle0);

    /* Restart timer */
    status = SYSTM002_StartTimer(OneMsTmrId);
}

int main(void)
{
```

DALI control gear
DALI communication – HOT (6/6)

- Disconnect LED from P0.2 by removing resistor

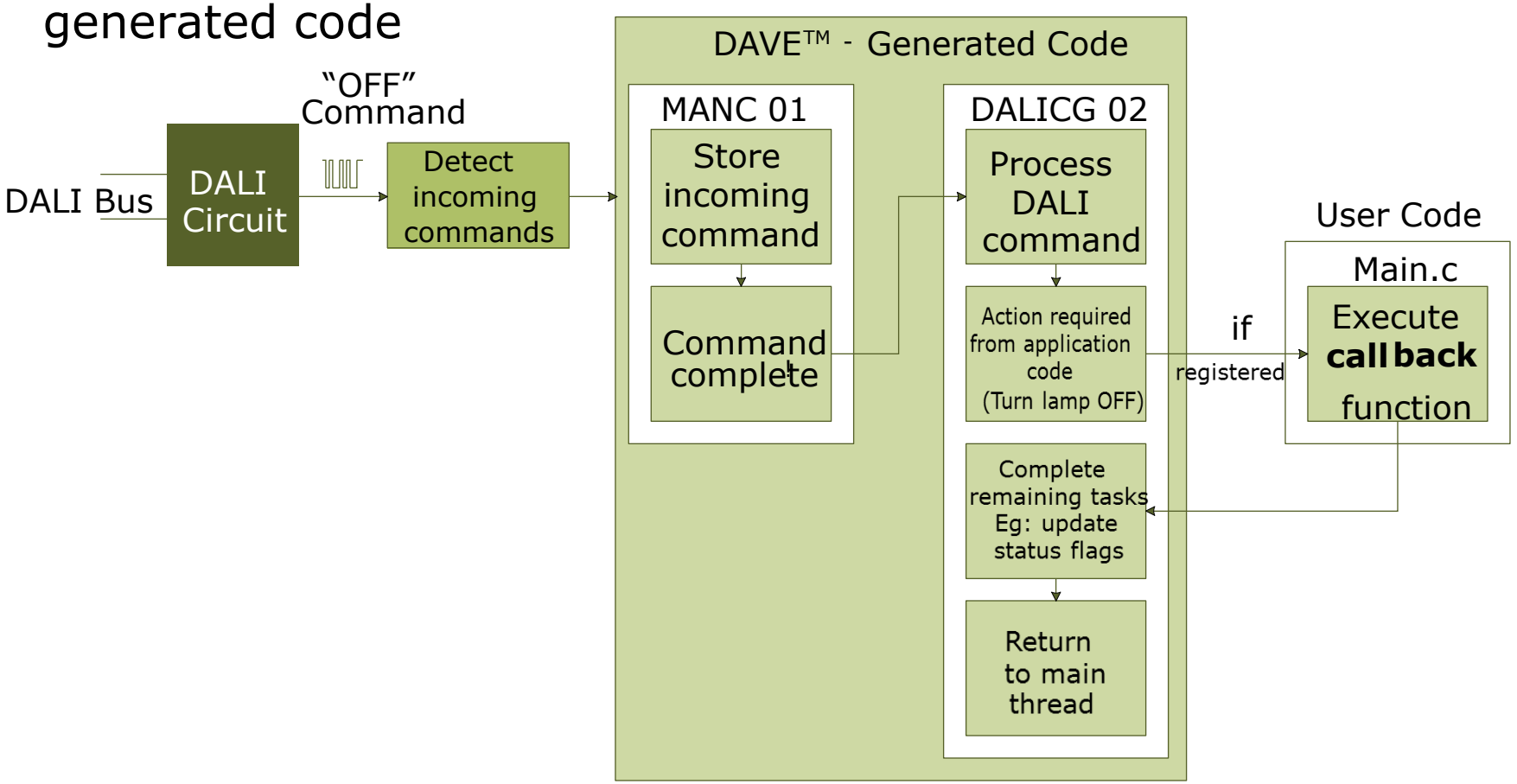


- Now, you're ready for DALI communication. Proceed to next HOT!

HOT3– immediate dimming without fading

DALI control gear – immediate dimming: Introduction to call backs

- Call backs allow user to incorporate customized code into DAVE™ generated code without having to understand the generated code

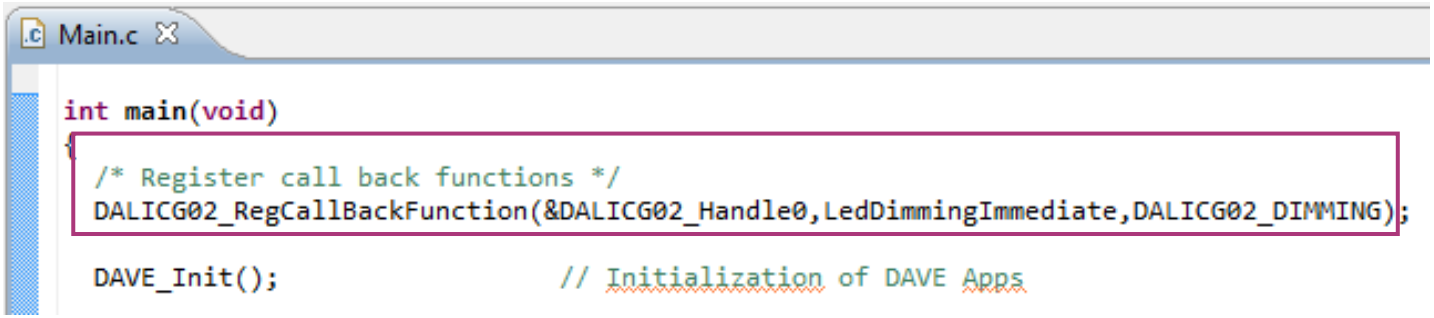


DALI control gear – immediate dimming: Registering call backs

- Condition: Call back functions have to be registered!
- Various call backs for different tasks
  - E.g. call back for immediate dimming without fading; call back for dimming with fade time; call back for dimming with fade rate for 200 ms etc.
- Identification via call back name/IDs
  - More information can be found in DALICG02 APP Help Documentation (under App Configuration Documentation section)
- API for registering call back:
  - DALICG02_RegCallBackFunction (DALICG02_HandlePointer, CallBack_Function_Name, CallBack_Name);

DALI control gear – immediate dimming: HOT(1/7)

- Call back name for immediate dimming without fading is DALICG02_DIMMING
- In Main.c, register call back
  - Format: DALICG02_RegCallBackFunction (DALICG02_HandlePointer, CallBack_Function_Name, CallBack_Name);



DALI control gear – immediate dimming: HOT(2/7)

- In Main.c, define call back function

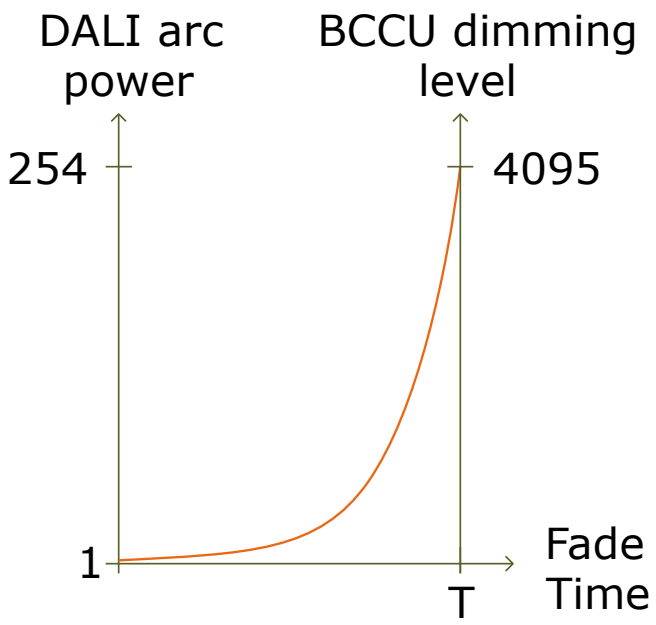
Main.c

```
/* Call Back Function
 * LED Dimming Immediate - goes to target level immediately without fading
 * Commands that use this function are OFF, STEP UP, STEP DOWN, RECALL MAX LEVEL, RESET etc.
 */
void LedDimmingImmediate(void)
{
    if(DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr != DALICG02_MASK)
    {
        /* Go to requested level immediately - DIMDIV=0, DCLK_PS=don't care */
        WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
                Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
                0U,1U);

        /* Update DALI actual level with target level */
        DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl =
                DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr;
    }
    /* Reset DALI fade status flag, since no fading took place */
    DALICG02_Handle0.DALI102_Handle->stStatus_info.bFade_running = DALICG02_BIT_ZERO;
    if(DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl == 0U)
    {
        /* Reset DALI light ON status flag if DALI actual level is 0 (OFF) */
        DALICG02_Handle0.DALI102_Handle->stStatus_info.bLight_on = DALICG02_BIT_ZERO;
    }
}
```

DALI control gear – immediate dimming: HOT(3/7)

```
/* Go to requested level immediately - DIMDIV=0, DCLK_PS=don't care */
WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
  Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
        0U,1U);
```

- DALI arc power levels range from 1-254
- BCCU dim levels range from 1-4095
- Dimlevel[] is a look-up table (LUT) which converts DALI arc power level to its equivalent BCCU dim level
  - LUT is automatically generated by DALICG02 APP when BCCU apps exist in project



DALI control gear – immediate dimming: HOT(4/7)

```
/* Go to requested level immediately - DIMDIV=0, DCLK_PS=don't care */
WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
  Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
        0U,1U);
```

- uiReq_arc_pwr: the requested arc power level by the DALI control device
- Immediate dimming without fading (dim time = 0 s)
  - ∴DimDiv = 0 (Dimming engine is bypassed)
  - DimClkPS = doesn't matter

DALI control gear – immediate dimming: HOT(5/7)

```
/* Go to requested level immediately - DIMDIV=0, DCLK_PS=don't care */
WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
  Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
        0U,1U);
```

- uiReq_arc_pwr: the requested arc power level by the DALI control device

```
/* Go to requested level immediately - DIMDIV=0, DCLK_PS=don't care */
WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
  Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
        0U,1U);
```

- Immediate dimming without fading (dim time = 0 s)
  - ∴DimDiv = 0 (Dimming engine is bypassed)
  - DimClkPS = doesn’t matter

DALI control gear – immediate dimming: HOT(6/7)

```
/* Update DALI actual level with target level */
DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl =
                DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr;
```

- uiActual_arc_lvl: the lamp's current arc power level

```
/* Reset DALI fade status flag, since no fading took place */
DALICG02_Handle0.DALI102_Handle->stStatus_info.bFade_running = DALICG02_BIT_ZERO;
```

- stStatus_info: DALI status information byte made up of the following bits
  - bControl_gear
  - bLamp_failure
  - bLight_on
  - bLimit_err
  - bFade_running
  - bReset_state
  - bMissing_short_addr
  - bPwr_failure

DALI control gear – immediate dimming: HOT(7/7)

- Build project 
- Download code 
- Start code 
- With the Control Device, send commands:
  - RECALL MIN (Observation: LEDs go to min level immediately)
  - QUERY ACTUAL LEVEL (Ans: 1)
  - RECALL MAX (Observation: LEDs go to max level immediately)
  - QUERY ACTUAL LEVEL (Ans: 254)
  - OFF (Observation: LEDs turn off immediately)
  - QUERY STATUS (Ans: XXXXX0XXb)

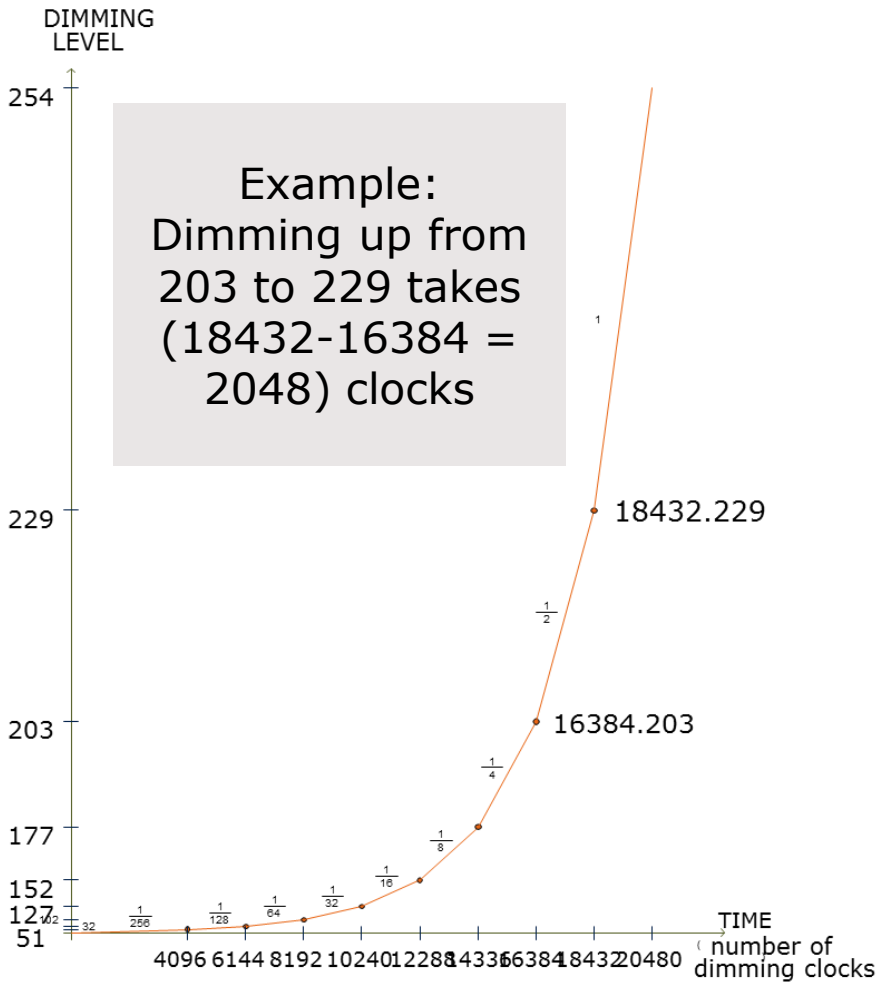
HOT4 – Dimming with fade time

DALI control gear – dimming with fade time
Translating DALI fade time

- To achieve the desired DALI fade time, the following BCCU parameters need to be determined:
  - No. of dimming clocks
    - Easily determined from LUT
  - DIMDIV
    - Easily determined from LUT
  - DCLK_PS
    - Calculation required

DALI control gear – dimming with fade time
Translating DALI fade time

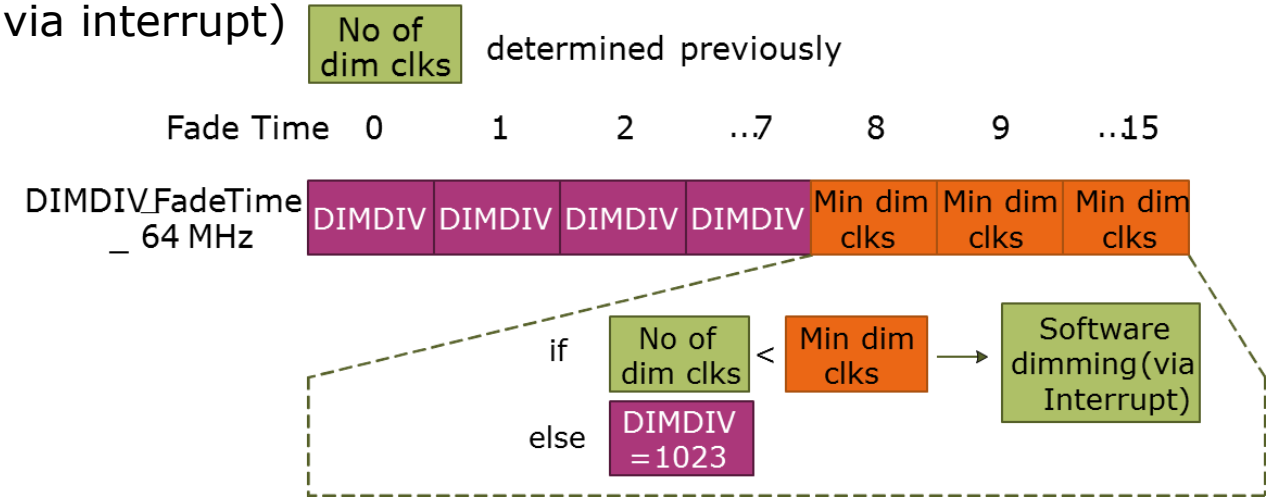
- No. of dimming clocks
  - Required no. of dimming clocks (along BCCU coarse exponential curve) from actual arc level to requested arc level
  - LUTs auto-generated in DALICG02
    - clocks_from_zero_coarse[] (when dimming up)
    - clocks_from_4095_coarse[] (when dimming down)



Example:
Dimming up from 203 to 229 takes (18432-16384 = 2048) clocks

DALI control gear – dimming with fade time
Translating DALI fade time

- DIMDIV
  - LUT auto-generated in DALICG02
    - DIMDIV_FadeTime_64MHZ[]
- Due to DIMDIV resolution, not all fade times can be achieved when the no. of dimming clocks is too low
  - Such cases occur when DALI Fade Time > 7
  - Min dimming clock is stored in DIMDIV_FadeTime_64MHZ[] for Fade Time >7
  - If no. of dimming clocks < min dimming clock, software dimming should be carried out (via interrupt)



DALI control gear – dimming with fade time
Translating DALI fade time

- DCLK_PS
  - To be calculated only when carrying out hardware dimming

$$DCLKPS = trunc\left(\frac{f_{BCCU\_clk} \times FadeTime_{from\_DALI}}{nr\_of\_clocks \times DIMDIV} \right)$$

DALI control gear – dimming with fade time HOT(1/6)

- Call back name for dimming with fade time is DALICG02_DIMMING_DARC
- In Main.c, register call back
  - Format: DALICG02_RegCallBackFunction (DALICG02_HandlePointer, CallBack_Function_Name, CallBack_Name);

Main.c

```
int main(void)
{
  /* Register call back functions */
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingImmediate,DALICG02_DIMMING);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingFadeTime,DALICG02_DIMMING_DARC);

  DAVE_Init();                  // Initialization of DAVE Apps
```

DALI control gear – dimming with fade time HOT(2/6)

- In Main.c, define call back function
  - For this demo, we will only use Fade Time = 0 (0.7 s)

Main.c

```
/* Call Back Function
 * LED Dimming Exponential with Fade Time - goes to target level with fading
 * Commands that use this function are Direct Arc Power Commands
*/
void LedDimmingFadeTime(void)
{
  /* variable to hold no. of dimming clocks required (for HW-based dimming)
   */
  uint32_t locNumDimClks;
  /* variable to hold dimmer clock prescaler (for HW-based dimming)
   */
  uint32_t locDclkps;
  /* variable to hold BCCU DIMDIV value (for HW-based dimming)
   */
  uint32_t locDimDiv;

  /* Calculate the no. of dimming clocks required from the LUTs */
  if(DALICG02_Handle0.DALI102_Handle->stDALICG02_flags.bDim_dir) /* Dimming Up  */
  {
    locNumDimClks = (clocks_from_zero_coarse[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr]
      - clocks_from_zero_coarse[DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl]);
  }
  else /* Dimming Down  */
  {
    locNumDimClks = (clocks_from_4095_coarse[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr]
      - clocks_from_4095_coarse[DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl]);
  }

  /* Get the BCCU DIMDIV value to use from the LUT */
  locDimDiv = DIMDIV_FadeTime_64MHZ[DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_FADE_TIME]];

  /* Calculate the Dimmer Clock Prescaler (DCLK_PS) */
  locDclkps = ((DALICG02_Handle0.DALI102_Handle->ausFade_time_tbl[DALICG02_Handle0.DALI102_Handle->
    aucDALICG02_var[DALICG02_FADE_TIME]] * 64000U) / (locDimDiv * locNumDimClks));
  if(locDclkps == 0U)
  {
    locDclkps = 1U;
  }

  /* Start BCCU dimming */
  WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
    Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],locDimDiv,locDclkps);
  /* Set DALI fade_running status flag */
  DALICG02_Handle0.DALI102_Handle->stStatus_info.bFade_running = DALICG02_BIT_ONE;
}
```

DALI control gear – dimming with fade time HOT(3/6)

```
/* Get the BCCU DIMDIV value to use from the LUT */
locDimDiv = DIMDIV_FadeTime_64MHZ[DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_FADE_TIME]];
```

- aucDALICG02_var[]: DALI variable array in RAM
  - aucDALICG02_var[0]: Short address
  - aucDALICG02_var[1]: Power-on Level
  - aucDALICG02_var[2]: System failure level
  - …
- To know more, refer to DALICG02.h

DALICG02.h

```
/* DALI Variables  */
#define DALICG02_SHORT_ADDR             0U
#define DALICG02_PWRON_ARC_LVL          1U
#define DALICG02_SYSTEM_FAIL_LVL        2U
#define DALICG02_MIN_LVL                3U
#define DALICG02_MAX_LVL                4U
#define DALICG02_FADE_RATE              5U
```

DALI control gear – dimming with fade time HOT(4/6)

- In the previous call back function, we have started the dimming of the lamp
- We need to monitor the status of the dimming in order to update DALI status flags
- This will be done in the 1ms periodic software timer interrupt routine, which we have defined earlier (see HOT2)

DALI control gear – dimming with fade time HOT(5/6)

› Add the following code to the interrupt service routine

Main.c

```
/* Service Routine for 1ms Periodic SW Timer Interrupt */
void OneMsIntr(void)
{
    uint32_t bccu_fade_status;

    /* Stop the timer */
    status = SYSTM002_StopTimer(OneMsTmrId);

    /* Call DALICG02_MainThread to maintain DALI communication*/
    DALICG02_MainThread(&DALICG02_Handle0);

    /* Restart timer */
    status = SYSTM002_StartTimer(OneMsTmrId);

    /* Check if BCCU has completed dimming - only for HW-based dimming */
    bccu_fade_status = BCCUDIM01_FadeCompletionStatus(&BCCUDIM01_Handle0);
    if(!bccu_fade_status)
    {
        if(DALICG02_Handle0.DALI102_Handle->stStatus_info.bFade_running)
        {
            /* Reset DALI fade status flag */
            DALICG02_Handle0.DALI102_Handle->stStatus_info.bFade_running = DALICG02_BIT_ZERO;
            /* Update DALI actual level */
            DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl = DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr;
            if(DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl == 0U)
            {
                /* Reset DALI light ON status flag if DALI actual level is 0 (OFF) */
                DALICG02_Handle0.DALI102_Handle->stStatus_info.bLight_on = DALICG02_BIT_ZERO;
            }
        }
    }
}
```

DALI control gear – dimming with fade time HOT(6/6)

- Build project 
- Download code 
- Start code 
- With the Control Device, send commands:
  - Direct Arc Power Command, Data = 1 (Observation: LEDs dim down to min level)
  - QUERY ACTUAL LEVEL (Ans: 1)
  - Direct Arc Power Command, Data = 100 (Observation: LEDs dim up)
  - QUERY ACTUAL LEVEL (Ans: 100)
  - Direct Arc Power Command, Data = 254 (Observation: LEDs dim up to max level)
  - QUERY ACTUAL LEVEL (Ans: 254)

HOT5 – Dimming for 200 ms with fade rate

DALI control gear – dimming with fade rate
Translating DALI fade rate

- To achieve the desired DALI fade rate, the following BCCU parameters need to be determined:
  - DIMDIV
    - LUTs auto-generated in DALICG02
      - dali_faderate_dimdiv_up[]
      - dali_faderate_dimdiv_dwn[]
  - DCLK_PS
    - Pre-determined fix value of 276 (dim up) and 273 (dim down)

DALI control gear – dimming with fade rate HOT(1/3)

- Call back name for dimming for 200 ms with fade rate is DALICG02_DIMMING200MS
- In Main.c, register call back
  - Format: DALICG02_RegCallBackFunction (DALICG02_HandlePointer, CallBack_Function_Name, CallBack_Name);

Main.c

```
int main(void)
{
  /* Register call back functions */
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingImmediate,DALICG02_DIMMING);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingFadeTime,DALICG02_DIMMING_DARC);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimming200ms,DALICG02_DIMMING200MS);

  DAVE_Init();                  // Initialization of DAVE Apps
```

DALI control gear – dimming with fade rate HOT(2/3)

- In Main.c, define call back function

Main.c

```
/* Call Back Function
 * Exponential Dimming based on DALI setting
 * This function is used
 */
void LedDimming200ms(void)
{
    /* Check if fading is still on-going -> abort if yes */
    if(BCCUDIM01_FadeCompletionStatus(&BCCUDIM01_Handle0))
    {
        BCCUDIM01_AbortDimming(&BCCUDIM01_Handle0,
                GET_CHANNEL_DIM_MASK(BCCUDIM01_Handle0.DE_Num));
    }

    /* Check dim direction */
    if(DALICG02_Handle0.DALI102_Handle->stDALICG02_flags.bDim_dir == 1U)
    {
        /* Dim up */
        if((DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl)!=
                DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_MAX_LVL])
        {
            /* Start dim up */
            WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
                    Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
                    dali_faderate_dimdiv_up[DALICG02_Handle0.DALI102_Handle->
                                            aucDALICG02_var[DALICG02_FADE_RATE]],276U);
        }
    }
    else
    {
        /* Dim down */
        if((DALICG02_Handle0.DALI102_Handle->uiActual_arc_lvl)!=
                DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_MIN_LVL])
        {
            /* Start dim down */
            WHITELAMP01_SetDimLevelExponential(&WHITELAMP01_Cfghandle0,
                    Dimlevel[DALICG02_Handle0.DALI102_Handle->uiReq_arc_pwr],
                    dali_faderate_dimdiv_dwn[DALICG02_Handle0.DALI102_Handle->
                                             aucDALICG02_var[DALICG02_FADE_RATE]],273U);
        }
    }
}
```

DALI control gear – dimming with fade rate HOT(3/3)

- Build project 
- Download code 
- Start code 
- With the Control Device, send commands:
  - DOWN (10 times)
  - QUERY ACTUAL LEVEL (Ans: 165)
  - UP (10 times)
  - QUERY ACTUAL LEVEL (Ans: 254)

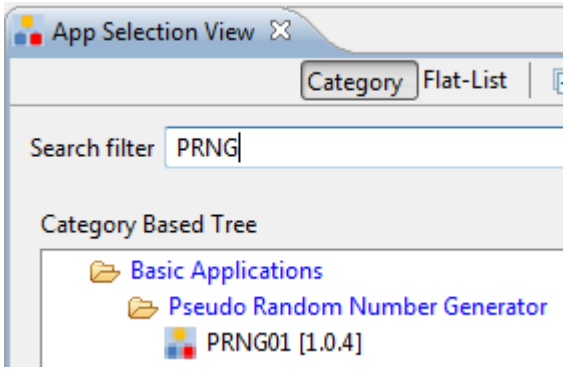
HOT6 – Random addressing

DALI control gear
Random addressing – PRNG

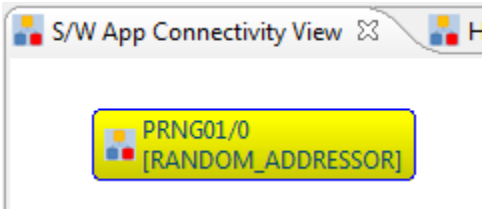
- "RANDOMISE" command
  - Control gear is expected to generate a new random address (24 bits)
- XMC1x00 has a Pseudo Random Number Generator (PRNG) which provides high quality random data with fast generation times
  - Eliminates the need for algorithm software

DALI control gear
Random addressing - HOT (1/5)

- Select PRNG01 from the App Selection View Window and add to project

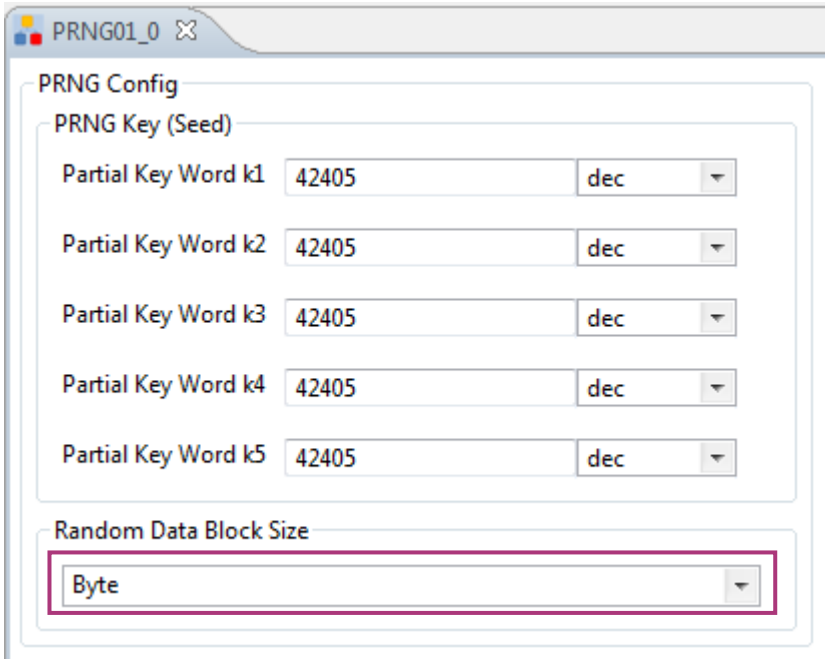


- Double-click on PRNG01 APP in S/W App Connectivity View to open the UI Editor



DALI control gear
Random addressing – HOT (2/5)

- Configure *Random Data Block Size* as Byte



- Generate Code 

DALI control gear
Random addressing – HOT (3/5)

- Call back name for generating random address is DALICG02_RANDOMISE_ADDR
- In Main.c, register call back
  - Format: DALICG02_RegCallBackFunction (DALICG02_HandlePointer, CallBack_Function_Name, CallBack_Name);

Main.c

```
int main(void)
{
  /* Register call back functions */
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingImmediate,DALICG02_DIMMING);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingFadeTime,DALICG02_DIMMING_DARC);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimming200ms,DALICG02_DIMMING200MS);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,GenerateRandomAddr,DALICG02_RANDOMISE_ADDR);

  DAVE_Init();                  // Initialization of DAVE Apps
```

DALI control gear
Random addressing – HOT (4/5)

- In Main.c, define call back function

Main.c

```
/* Call Back Function
 * Uses PRNG to generate random addresses
 * Command that uses this function is RANDOMISE
 */
void GenerateRandomAddr(void)
{
    DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_RANDOM_ADDR_H] = PRNG01_GetPseudoRandomNumber();
    DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_RANDOM_ADDR_M] = PRNG01_GetPseudoRandomNumber();
    DALICG02_Handle0.DALI102_Handle->aucDALICG02_var[DALICG02_RANDOM_ADDR_L] = PRNG01_GetPseudoRandomNumber();
}
```

- PRNG01_GetPseudoRandomNumber()
  - returns random 8-bit number

DALI control gear
Random addressing – HOT (5/5)

- Build project 
- Download code 
- Start code 
- With the Control Device, send commands:
  - INITIALISE(0)
  - RANDOMISE
  - QUERY RANDOM ADDRESS (H)
  - QUERY RANDOM ADDRESS (M)
  - QUERY RANDOM ADDRESS (L)

HOT7 – Persistent memory

DALI control gear
Persistent memory – flash EEPROM emulation

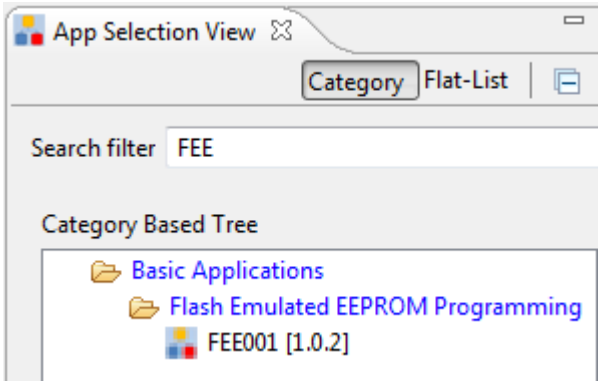
- Persistent memory:
  - Memory banks
  - Some DALI variables e.g. Short address, DTR etc.

- Persistent memory is realized with Flash EEPROM Emulation
  - DAVE™ APP: FEE001

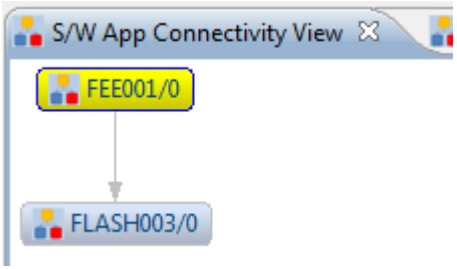
- DALICG02 also stores Memory banks 0 & 1 (up to 256 bytes) and DALI variables in RAM
  - Copies memory and variables from flash to RAM in DALI initialization
  - Copies memory and variables to flash whenever there are updates in RAM

DALI control gear
Persistent memory – HOT (1/14)

- Select FEE001 from the App Selection View Window and add to project

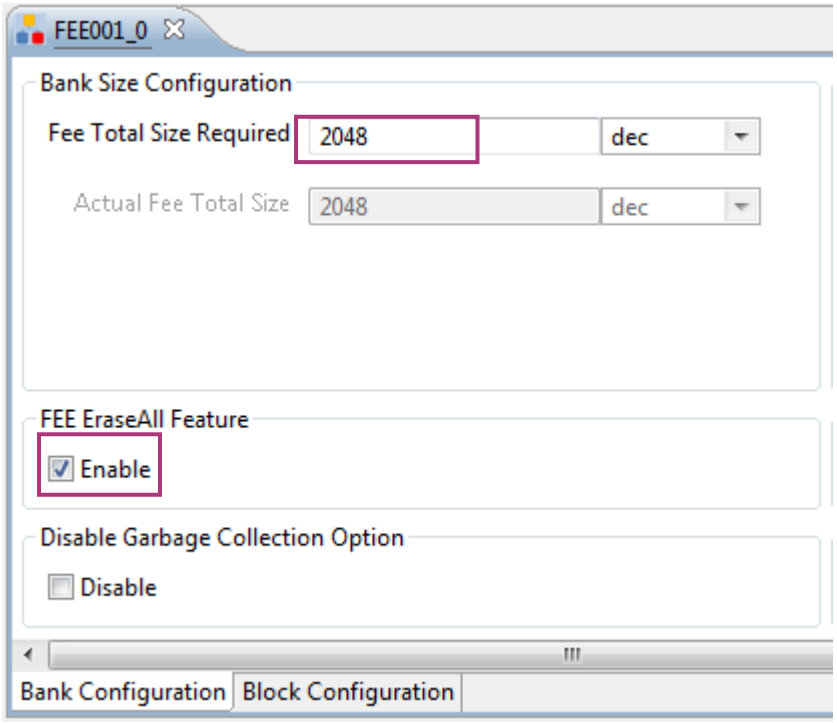


- Double-click on FEE001 APP in S/W App Connectivity View to open the UI Editor



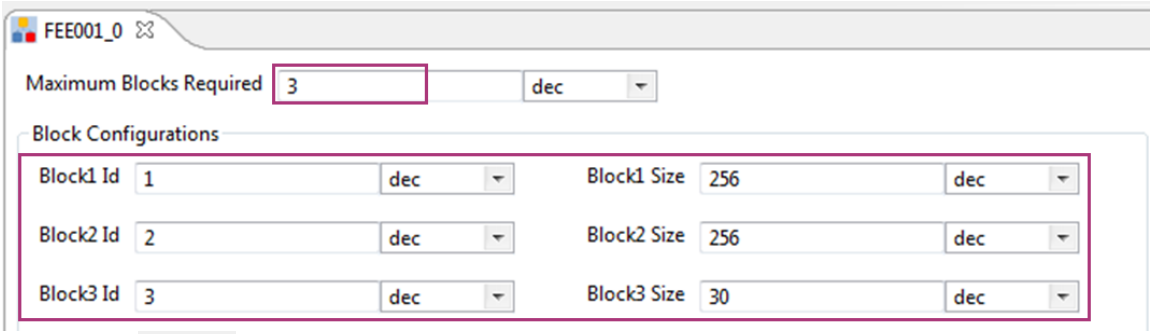
DALI control gear
Persistent memory – HOT (2/14)

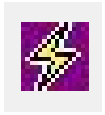
- Under Bank Configuration tab:
  - Configure *FEE Total Size Required* to 2048 bytes
  - Enable *EraseAll Feature*



DALI control gear
Persistent memory – HOT (3/14)

- Under Block Configuration tab:
  - Configure *Maximum Blocks Required* to 3 blocks
  - Set up the individual blocks as follows:
    - Block1 ID: 1, Block1 Size: 256 bytes (this is for Memory Bank 0)
    - Block2 ID: 2, Block2 Size: 256 bytes (this is for Memory Bank 1)
    - Block3 ID: 3, Block3 Size: 30 bytes (this is for DALI variables)



- Generate Code 

DALI control gear
Persistent memory – HOT (4/14)

- Reading of memory banks 0 and 1 are handled by DALICG02 APP
- Writing to memory banks 0 and 1 is via call back
- Call back name for writing to memory banks is DALICG02_MEMCALLBACK_ADDR
- In Main.c, register call back

Main.c

```
int main(void)
{
  /* Register call back functions */
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingImmediate,DALICG02_DIMMING);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingFadeTime,DALICG02_DIMMING_DARC);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimming200ms,DALICG02_DIMMING200MS);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,GenerateRandomAddr,DALICG02_RANDOMISE_ADDR);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,WriteToMemBank,DALICG02_MEMCALLBACK_ADDR);

  DAVE_Init();                 // Initialization of DAVE Apps
```

DALI control gear
Persistent memory – HOT (5/14)

- In Main.c, define call back function

Main.c

```
/* Call Back Function
 * Write to DALI Memory Location
 */
void WriteToMemBank(void)
{
    uint16_t count;

    Clear_ReadWriteBuffer();

    if(DALICG02_Handle0.DALI102_Handle->uiDTR1 == DALICG02_MEMORY_BANK0)
    {
        for(count=0;count<DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[DALICG02_LAST_MEM_ADDR];count++)
        {
            ReadWriteBuffer[count] = (uint8_t)DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[count];
        }
        FEE001_Write(1U, ReadWriteBuffer);
    }
    else if(DALICG02_Handle0.DALI102_Handle->uiDTR1 == DALICG02_MEMORY_BANK1)
    {
        for(count=0;count<DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank1[DALICG02_LAST_MEM_ADDR];count++)
        {
            ReadWriteBuffer[count] = (uint8_t)DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank1[count];
        }
        FEE001_Write(2U, ReadWriteBuffer);
    }
}
```

DALI control gear
Persistent memory – HOT (6/14)

```
ReadWriteBuffer[count] = (uint8_t)DALICG02_Handle0.I
```

- uint8_t ReadWriteBuffer[256U]: buffer for reading and writing data to and from Flash
  - Initialize in Main.c

```
Clear_ReadWriteBuffer();
```

- Function to clear the read write buffer
  - define function in Main.c

```
/* Function to clear and read write buffer */
void Clear_ReadWriteBuffer(void)
{
    uint16_t i;

    for(i=0;i<256;i++)
    {
        ReadWriteBuffer[i] = 0U;
    }
}
```

DALI control gear
Persistent memory – HOT (7/14)

```
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[count];
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank1[count];
```

- RAM copies of memory banks 0 & 1

```
FEE001_Write(1, ReadWriteBuffer);
```

- FEE001 API for writing Flash block with buffer data

DALI control gear
Persistent memory – HOT (8/14)

- Contents of memory banks 0 & 1 have to be initialized manually in Main.c
  - Example code:

Main.c

```
DAVE_Init();                    // Initialization of DAVE Apps

/* Here: Initialise/Write your Memory Bank 0 bytes */
DALICG02_Handle0.DALI102_Handle->uiDTR1 = DALICG02_MEMORY_BANK0;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x00U] = 0x0F;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x02U] = 0x01;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x03U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x04U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x05U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x06U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x07U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x08U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x09U] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x0AU] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x0BU] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x0CU] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x0DU] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x0EU] = 0x55;
DALICG02_Handle0.DALI102_Handle->aucDALICG02_memory_bank0[0x01U] = DALICG02_uiMem_bank_chksum(&DALICG02_Handle0);
WriteToMemBank();
```

- DALICG02_uiMem_bank_chksum()
  - Function that calculates the checksum of memory bank 0 or 1 (based on DTR1 value)

DALI control gear
Persistent memory – HOT (9/14)

- Reading of and writing to DALI variables flash block are via call backs
- Call back name for reading of DALI variables flash block is DALICG02_READ_VARIABLES_SECTOR
- In Main.c, register call back

Main.c

```
int main(void)
{
  /* Register call back functions */
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingImmediate,DALICG02_DIMMING);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingFadeTime,DALICG02_DIMMING_DARC);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimming200ms,DALICG02_DIMMING200MS);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,GenerateRandomAddr,DALICG02_RANDOMISE_ADDR);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,WriteToMemBank,DALICG02_MEMCALLBACK_ADDR);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,ReadVarSec,DALICG02_READ_VARIABLES_SECTOR);

  DAVE_Init();                  // Initialization of DAVE Apps
```

DALI control gear
Persistent memory – HOT (10/14)

- In Main.c, define call back function

Main.c

```
/* Call Back Function
 * Read Variables Flash Sector (Flash EEPROM Emulation)
 */
void ReadVarSec(void)
{
    uint8_t count;

    Clear_ReadWriteBuffer();
    status = FEE001_Read(3U,0U,ReadWriteBuffer,30U);
    for(count=0;count<30;count++)
    {
        DALICG02_Handle0.DALI102_Handle->aucDALICG02_FlashVariables_Sector_tbl[count] = (uint32_t)ReadWriteBuffer[count];
    }
}
```

- FEE001_Read()
  - FEE001 API for reading of flash block
- aucDALICG02_FlashVariables_Sector_tbl[]
  - Copy of DALI variables in RAM

DALI control gear
Persistent memory – HOT (11/14)

- Call back name for writing to DALI variables flash block is DALICG02_WRITE_VARIABLES_SECTOR
- In Main.c, register call back

Main.c

```
int main(void)
{
  /* Register call back functions */
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingImmediate,DALICG02_DIMMING);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimmingFadeTime,DALICG02_DIMMING_DARC);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,LedDimming200ms,DALICG02_DIMMING200MS);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,GenerateRandomAddr,DALICG02_RANDOMISE_ADDR);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,WriteToMemBank,DALICG02_MEMCALLBACK_ADDR);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,ReadVarSec,DALICG02_READ_VARIABLES_SECTOR);
  DALICG02_RegCallBackFunction(&DALICG02_Handle0,WriteBufferToVarSec,DALICG02_WRITE_VARIABLES_SECTOR);

  DAVE_Init();                 // Initialization of DAVE Apps
```

DALI control gear
Persistent memory – HOT (12/14)

- In Main.c, define call back function

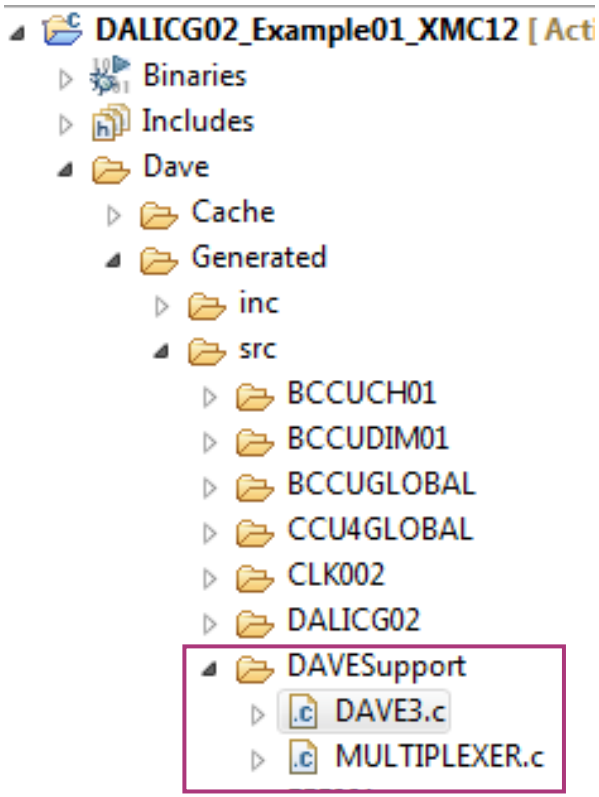
Main.c

```
/* Call Back Function
 * Write to Variables Flash Sector (Flash EEPROM Emulation)
 */
void WriteBufferToVarSec(void)
{
    uint8_t count;

    Clear_ReadWriteBuffer();
    ReadWriteBuffer[0U] = DALICG02_FLASH_SECT_NEW_DATA;
    for(count=1;count<28;count++)
    {
        ReadWriteBuffer[count] = (uint8_t)DALICG02_Handle0.DALI102_Handle->aucDALICG02_FlashVariables_Sector_tbl[count];
    }
    ReadWriteBuffer[29U] = DALICG02_FLASH_SECT_PROG;

    status = FEE001_Write(3U, ReadWriteBuffer);
}
```

- Macros defined for identifying new data programmed into flash block
  - DALICG02_FLASH_SECT_NEW_DATA (0x00) -> first byte of flash block
  - DALICG02_FLASH_SECT_PROG (0x81) -> last byte of flash block

DALI control gear
Persistent memory – HOT (13/14)

› Open DAVE3.c



› In DAVE_Init() function, check that FLASH003_Init() and FEE001_Init() are called before DALICG02_Init()

– If not, manually rearrange the order

```
        //  Initialization of app 'MANC01'
        MANC01_Init();

        //  Initialization of app 'FLASH003'
        FLASH003_Init();

        //  Initialization of app 'FEE001'
        FEE001_Init();

        //  Initialization of app 'DALICG02'
        DALICG02_Init();

        //  Initialization of app 'PRNG01'
        PRNG01_Init();

        //  MUX configurations
        DAVE_MUX_Init();
} //  End of function DAVE_Init
```

DALI control gear
Persistent memory – HOT (14/14)

- Build project 
- Download code 
- Start code 
- With the Control Device, send commands:
  - DTR = 2
  - STORE DTR AS SHORT ADDRESS
  - INITIALISE(0)
  - QUERY SHORT ADDRESS (Ans: 2)
  - Cycle power
  - INITIALISE(0)
  - QUERY SHORT ADDRESS (Ans: 2)

General information

- Where to buy kit?

  http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=db3a30443ba77cfd013baec9c7880ca9

- For latest updates, please refer to:

  http://www.infineon.com/xmc1000

- For support:

  http://www.infineonforums.com

Resource listing

- DALI Control Gear DAVE™ project

  http://www.infineon.com/cms/en/product/promopages/aim-mc/dave_downloads.html

- LED Lighting Application Kit documentation

  http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=db3a30443ba77cfd013baec9c7880ca9

- Tridonic DALI USB

  http://www.tridonic.com/com/en/products/2622.asp

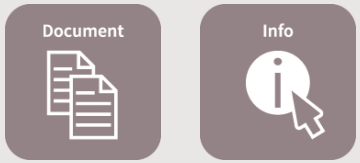
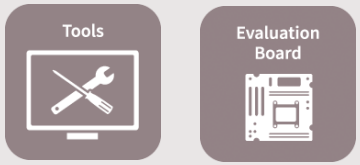
- Tridonic DALI PS1

  http://www.tridonic.com/com/en/products/2626.asp

- Master CONFIGURATOR

  http://www.tridonic.com/com/en/software-masterconfigurator.asp

Support material

Collaterals and Brochures 	› Product Briefs › Selection Guides › Application Brochures › Presentations › Press Releases, Ads	› www.infineon.com/XMC
Technical Material 	› Application Notes › Technical Articles › Simulation Models › Datasheets, MCDS Files › PCB Design Data	› www.infineon.com/XMC › Kits and Boards › DAVE™ › Software and Tool Ecosystem
Videos 	› Technical Videos › Product Information Videos	› Infineon Media Center › XMC Mediathek
Contact 	› Forums › Product Support	› Infineon Forums › Technical Assistance Center (TAC)

Disclaimer

The information given in this training materials is given as a hint for the implementation of the Infineon Technologies component only and shall not be regarded as any description or warranty of a certain functionality, condition or quality of the Infineon Technologies component.

Infineon Technologies hereby disclaims any and all warranties and liabilities of any kind (including without limitation warranties of non-infringement of intellectual property rights of any third party) with respect to any and all information given in this training material.

Part of your life. Part of tomorrow.

Infineon