Application - digital multiplex (DMX512) receiving device

XMC™ microcontrollers
September 2016
## Agenda

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Key features</td>
</tr>
<tr>
<td>2</td>
<td>Specification</td>
</tr>
<tr>
<td>3</td>
<td>System block diagram</td>
</tr>
<tr>
<td>4</td>
<td>Hardware overview</td>
</tr>
<tr>
<td>5</td>
<td>Software overview</td>
</tr>
<tr>
<td>6</td>
<td>Highlight MCU features</td>
</tr>
<tr>
<td>7</td>
<td>Hands-on training</td>
</tr>
<tr>
<td>8</td>
<td>Extras</td>
</tr>
</tbody>
</table>
DMX512 receiving device – Key features

Target application
› DMX512 receiving device

Key features
› RS-485
› Daisy chain
  – Single master (transmitting device)
  – Up to 32 slaves per branch (receiving device)
<table>
<thead>
<tr>
<th></th>
<th>Agenda</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Key features</td>
</tr>
<tr>
<td>2</td>
<td>Specification</td>
</tr>
<tr>
<td>3</td>
<td>System block diagram</td>
</tr>
<tr>
<td>4</td>
<td>Hardware overview</td>
</tr>
<tr>
<td>5</td>
<td>Software overview</td>
</tr>
<tr>
<td>6</td>
<td>Highlight MCU features</td>
</tr>
<tr>
<td>7</td>
<td>Hands-on training</td>
</tr>
<tr>
<td>8</td>
<td>Extras</td>
</tr>
</tbody>
</table>
Specifications

› Connectors: 5-pin XLR
  – Often 3-pin XLR is used
› Cable: twisted-pair, shielded, low-capacitance data cable
› Data: 250 kHz transmission
Agenda

1. Key features
2. Specification
3. System block diagram
4. Hardware overview
5. Software overview
6. Highlight MCU features
7. Hands-on training
8. Extras
DMX512 receiving device – System block diagram

System block diagram: DMX system
Agenda

1. Key features
2. Specification
3. System block diagram
4. Hardware overview
5. Software overview
6. Highlight MCU features
7. Hands-on training
8. Extras
DMX512 receiving device – Hardware overview

› XMC1000 LED Lighting Application Kit comprising of
  – XMC1200 Boot Kit
  – Colour LED card

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

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

Copyright © Infineon Technologies AG 2016. All rights reserved.
Flow chart: DMX512 receiving device – Software overview

- **Power up -> Main init**
- **IDLE: Wait for slots.**
- **RX:** Slots received. Go to target levels.
Agenda

1. Key features
2. Specification
3. System block diagram
4. Hardware overview
5. Software overview
6. Highlight MCU features
7. Hands-on training
8. Extras
DMX512 receiving device – Highlight MCU features

› BCCU
  - 12-bit intensities
  - 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
  - 12-bit dimming level

› USIC
  - USIC channel detects DMX512 packet and break automatically

› [Optional] CCU4
  - Capture mode and timer automatically and accurately detects break
<table>
<thead>
<tr>
<th></th>
<th>Agenda</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Key features</td>
</tr>
<tr>
<td>2</td>
<td>Specification</td>
</tr>
<tr>
<td>3</td>
<td>System block diagram</td>
</tr>
<tr>
<td>4</td>
<td>Hardware overview</td>
</tr>
<tr>
<td>5</td>
<td>Software overview</td>
</tr>
<tr>
<td>6</td>
<td>Highlight MCU features</td>
</tr>
<tr>
<td>7</td>
<td>Hands-on training</td>
</tr>
<tr>
<td>8</td>
<td>Extras</td>
</tr>
</tbody>
</table>
DMX512 receiving device
Hands-on training

› Receiving device
  - XMC1200 Boot Kit + Colour LED card

› Transmitting device
  - eldoLED DimWheel Colour EU DMX Controller
DMX512 receiving device
Hands-on: hardware setup

eldoLED
DimWheel Color

3 DMX slots:
(1) Red, (2) Green, (3) Blue

LED Lighting Application Kit
XMC1200 CPU Card
Colour LED Card

USB
XMC1200
DALI
DMX
RF
DMX512 receiving device
Hands-on: block diagram

Block diagram: DMX512 receiving device demo

Legend:
- Green: On-chip hardware
- Blue: Off-chip hardware
- Orange: Software

Copyright © Infineon Technologies AG 2016. All rights reserved.
DMX512 receiving device
Hands-on: board schematic

Schematic: Simple non-isolated DMX512 interface
DMX512 receiving device
Hands-on: board schematic

Schematic: Isolated DMX512 interface
DMX512 receiving device
Hands-on: software, list of DAVE™ APPs

› PDM_DIMMED_LED_LAMP
  - Aggregates GLOBAL_BCCU and PDM_BCCU APPs
  - Provides configurations, color and dimming control for RGB LED lamp

› DMX512_RD
  - DMX512 application stack

› DIGITAL_IO
  - Initializes multiple IO pins that are connected to the other 2 unused RGB LEDs on board
  - Also initializes DMX input pin

› [Optional] EVENT_DETECTOR, EVENT_GENERATOR
  - For accurate break detection
DMX512 receiving device
Hands-on: HOT (1/14)

› Add one instance of PDM_DIMMED_LED_LAMP APP to the project

› PDM_DIMMED_LED_LAMP aggregates DIM_BCCU, PDM_BCCU, GLOBAL_BCCU and CLOCK_XMC1. Double-click PDM_DIMMED_LED_LAMP in APP Dependency View to open the UI Editor
DMX512 receiving device
Hands-on: HOT (2/14)

› Under *General Settings* tab,
  - Select 3 LED channels
  - Select *Global Dimming* as dimming source

› Under *Dimming and Intensities Settings* tab,
  - Set intensities to 0
Assign PDM_BCCU APPs to the right channels

- Hover mouse cursor over the connecting arrow to a PDM_BCCU APP
- A label will appear momentarily e.g. LED0/LED1/LED2
The labels correspond to the LED channels in the UI

- Right-click PDM_BCCU APP
- Select “Rename Instance Label”

<table>
<thead>
<tr>
<th>Label</th>
<th>New Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED0</td>
<td>RED</td>
</tr>
<tr>
<td>LED1</td>
<td>GREEN</td>
</tr>
<tr>
<td>LED2</td>
<td>BLUE</td>
</tr>
</tbody>
</table>

Repeat the above steps with the other 2 PDM_BCCU APP instances
DMX512 receiving device
Hands-on: HOT (5/14)

› Open UI of a PDM_BCCU APP
› Enable *Flicker Watchdog*

› Repeat for other 2 PDM_BCCU instances
DMX512 receiving device
Hands-on: HOT (6/14)

› Open UI of GLOBAL_BCCU APP
› Under Function Settings tab,
  – Set initial global dimming level to 4095
  – Set Flicker Watchdog threshold to 800
DMX512 receiving device
Hands-on: HOT (7/14)

- Add 6 instances of DIGITAL_IO APPs to the project for the pins to the unused LEDs
- Open the UI of a DIGITAL_IO APP
- Configure the pin direction as “Input/Output”

![Digital IO Interface]

- Repeat for other 5 DIGITAL_IO instances
- Rename instance label for all 6 DIGITAL_IO instances as “UNUSED_LEDx” where x is 1 to 6
DMX512 receiving device
Hands-on: HOT (8/14)

› Add an instance of DMX512_RD to the project
› Open UI of DMX512_RD APP
› Under General Settings tab,
  › Configure First relevant slot to 1
  › Configure Number of relevant slots to 3
DMX512 receiving device
Hands-on: HOT (9/14)

› Add one instance of DIGITAL_IO APP for configuring DMX input pin
› Open UI of DIGITAL_IO APP
› Configure pin direction as “Input”

› Rename instance label as “DMX_INPUT”
Connect DMX_INPUT to DMX512_RD APP

- Right-click DMX_INPUT
  - Select “HW Signal Connections”
  - Configure as follows:

<table>
<thead>
<tr>
<th>Source APP Instance Name</th>
<th>Source Signal</th>
<th>Connect To</th>
<th>Target APP Instance Name</th>
<th>Target Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>DMX_INPUT</td>
<td>pin</td>
<td>----&gt;</td>
<td>DMX512_RD_0</td>
<td>dmx512_input</td>
</tr>
<tr>
<td>Not Selected</td>
<td>Not Selected</td>
<td>----&gt;</td>
<td>Not Selected</td>
<td>Not Selected</td>
</tr>
</tbody>
</table>

- Click “Solve and Save”
- Click “Close”
DMX512 receiving device
Hands-on: HOT (11/14)

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

› Assign the pins as follows:

<table>
<thead>
<tr>
<th>APP Instance Name</th>
<th>APP Pin Name</th>
<th>Pin Number (Port)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLUE</td>
<td>PDM Output pin</td>
<td>#18 ( P0.1 )</td>
</tr>
<tr>
<td>DMX_INPUT</td>
<td>pin</td>
<td>#36 ( P2.1 )</td>
</tr>
<tr>
<td>GREEN</td>
<td>PDM Output pin</td>
<td>#30 ( P0.11 )</td>
</tr>
<tr>
<td>RED</td>
<td>PDM Output pin</td>
<td>#21 ( P0.4 )</td>
</tr>
<tr>
<td>UNUSED_LED1</td>
<td>pin</td>
<td>#22 ( P0.5 )</td>
</tr>
<tr>
<td>UNUSED_LED2</td>
<td>pin</td>
<td>#23 ( P0.6 )</td>
</tr>
<tr>
<td>UNUSED_LED3</td>
<td>pin</td>
<td>#24 ( P0.7 )</td>
</tr>
<tr>
<td>UNUSED_LED4</td>
<td>pin</td>
<td>#27 ( P0.8 )</td>
</tr>
<tr>
<td>UNUSED_LED5</td>
<td>pin</td>
<td>#28 ( P0.9 )</td>
</tr>
<tr>
<td>UNUSED_LED6</td>
<td>pin</td>
<td>#29 ( P0.10 )</td>
</tr>
</tbody>
</table>

› Click “Solve and Save”
› Click “Close”
Generate code

In Main.c, define the DMX512_RD callback function:

```c
void DMX512_RD_UserCallBack(void)
{
    /* Extract 8-bit information for Red color */
    PDM_DIMMED_LED_LAMP.config->led_intensity[0] = DMX512_RD_0_rx_array[0] << 4U;
    /* Extract 8-bit information for Green color */
    /* Extract 8-bit information for Blue color */
    /* Change lamp color */
    PDM_DIMMED_LED_LAMP_SetColor(&PDM_DIMMED_LED_LAMP_0);
}
```
DMX512 receiving device
Hands-on: HOT (13/14)

- DMX512_RD callback function
  - It is called after slot data detection
  - Typically, the LED channel intensities are updated here to achieve what the DMX512 master is requesting
  - 8-bit intensity information in this example
DMX512 receiving device
Hands-on: HOT (14/14)

› Build project
› Connect XMC1200 Boot Kit to PC
› Download code
› Start code
› Dial knob on eldoLED DimWheel Colour
› Observe LEDs on Colour LED card
Agenda

1. Key features
2. Specification
3. System block diagram
4. Hardware overview
5. Software overview
6. Highlight MCU features
7. Hands-on training
8. Extras
DMX512 receiving device
Extras: 16-bit slots

- If DMX512 transmitting device transmits slots of 16 bits,
  - In DMX512_RD UI Editor:
    - Set no. of relevant slots to 6

- Re-generate code
- Adjust code as follows:

```c
void DMX512_RD_UserCallBack(void)
{
    RGB_LAMP.config->led_intensity[0] = (uint16_t)((DMX512_RD_0_rx_array[0] << 4U) + (DMX512_RD_0_rx_array[1] >> 4U)); /* 16-bit information for Red color */
    PDM_DIMMED_LED_LAMPSetColor(&RGB_LAMP);
}
```

- Rebuild project
- Download code
DMX512 receiving device
Extras: accurate break detection (1/6)

- DMX512 defines break as low signal for minimum duration of 92μs
- By default, DMX512_RD uses USIC Sync break for break detection
  - Functional but does not confirm the minimum duration
- Accurate detection can be achieved by using a CCU4 slice
  - In DMX512_RD UI Editor:
    - Enable accurate break detection
DMX512 receiving device
Extras: accurate break detection (2/6)

› Add EVENT_DETECTOR and EVENT_GENERATOR APPs (one instance each) to project
  - This is necessary because the input pin (P2.1) is not connected to any CCU4 slice so the input signal has to be rerouted via ERU

› Open the UI of EVENT_DETECTOR
DMX512 receiving device
Extras: accurate break detection (3/6)

› Select “B” as request source as P2.1 is connected to input source B of the ERU slice

<table>
<thead>
<tr>
<th>Global Inputs/Outputs</th>
<th>Connected To</th>
<th>I/O</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ERU0.0B2</td>
<td>ORC0.OUT</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>ERU0.0B3</td>
<td>VADC0.G1BFLOUT0</td>
<td>I</td>
<td>from ADC boundary flag</td>
</tr>
<tr>
<td>ERU0.1A0</td>
<td>ACMP1.OUT</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>ERU0.1A1</td>
<td>P2.5</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>ERU0.1A2</td>
<td>ORC3.OUT</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>ERU0.1A3</td>
<td>VADC0.G0BFLOUT1</td>
<td>I</td>
<td>from ADC boundary flag</td>
</tr>
<tr>
<td>ERU0.1B0</td>
<td>P2.1</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>ERU0.1B1</td>
<td>P2.3</td>
<td>I</td>
<td></td>
</tr>
</tbody>
</table>

› Select “Rising edge” detection
› Enable status flag autoclear
DMX512 receiving device
Extras: accurate break detection (4/6)

› Open the UI of EVENT_GENERATOR
› Enable pattern detection

› Connect DMX_INPUT pin to EVENT_DETECTOR
  – Right-click DMX_INPUT
  – Select “HW Signal Connections”
  – Configure as follows:

<table>
<thead>
<tr>
<th>DMX_INPUT</th>
<th>pin</th>
<th>-----</th>
<th>DMX512_RD_0</th>
<th>dmx512_input</th>
</tr>
</thead>
<tbody>
<tr>
<td>EVENT_DETECTOR_0</td>
<td>pin</td>
<td>-----</td>
<td>signal_b</td>
<td></td>
</tr>
</tbody>
</table>

  – Click “Save”
  – Click “Close”
Connect EVENT_DETECTOR status signal to EVENT_GENERATOR

- Right-click EVENT_DETECTOR
- Select “HW Signal Connections”
- Configure as follows:

```
<table>
<thead>
<tr>
<th>EVENT_DETECTOR_0</th>
<th>status</th>
<th>-----&gt;</th>
<th>EVENT_GENERATOR_0</th>
<th>pattern</th>
</tr>
</thead>
</table>
```

- Click “Save”
- Click “Close”
DMX512 receiving device
Extras: accurate break detection (6/6)

› Connect EVENT_GENERATOR pattern detect signal to DMX512_RD for break detection
  – Right-click EVENT_GENERATOR
  – Select “HW Signal Connections”
  – Configure as follows:

![EVENT_GENERATOR_0](image)

  - Click “Save”
  - Click “Close”

› Re-generate code
› Rebuild project
› Download code
General information

› Where to buy kit?
  - [www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=db3a30443ba77cfd013baec9c7880ca9](www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=db3a30443ba77cfd013baec9c7880ca9)

› For latest updates, please refer to:
  - [www.infineon.com/xmc1000](www.infineon.com/xmc1000)

› For support:
  - [www.infineonforums.com](www.infineonforums.com)
Resource listing

› DMX512 receiving device DAVE™ project
http://www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=eb3a30443ba77cfd013baec9c7880ca9
(look under Documents tab)

› LED Lighting Application Kit documentation
www.infineon.com/cms/en/product/evaluation-boards/KIT_XMC1X_AK_LED_001/productType.html?productType=db3a30443ba77cfd013baec9c7880ca9

› eldoLED DimWheel Colour EU
Part of your life. Part of tomorrow.