Controller Area Network (CAN) example project

3.30

Features

- CAN Configuration example
- Transmitting and receiving Full messages example
- Handling Tx and Rx error example

General Description

This example project demonstrates the CAN Component operation with the configured Full mailboxes.
This is only one part of example project, for completing the entire example you should also find the CANExampleControl project.

Development kit configuration

1. This project is written for a 2X16 LCD display as the one available in the Cypress kit CY8CKIT-001.
2. The two PSoC3/5 DVK CY8CKIT-001 boards besides default configuration should have LCD power jumper (J12).
3. The two CY8CKIT-017 CAN/LIN Expansion Boards, connected to DVK1 PORT A 2x20 connector with installed jumpers: JP2 (CAN Termination Resistor), CANEXTPWR and JP6 set to Vdd-V5_0 (default setting for 5V operation). Any jumpers on the board not mentioned above should have no jumper installed.
4. CAN – USB analyzer for analyzing data traffic can be used.
5. Build the project and program the hex file on to the target device.
6. Power cycle the device and observe the results on the LCD.
7. Pin 0[1] connected to LED1, Pin 0[5] connected to LED4.

Project configuration

The example project consists of the CAN, Character LCD and RTC components. The top design schematic is as shown in Figure 1. The Character LCD component is used for receiving data and displaying time. The RTC component is used for system time information. To ensure proper functioning of the example projects, the customer should create a mini network from at least two

CAN nodes as show in Figure 2. The network shown in Figure 2 includes the CAN USB analyzer for data traffic analysis.

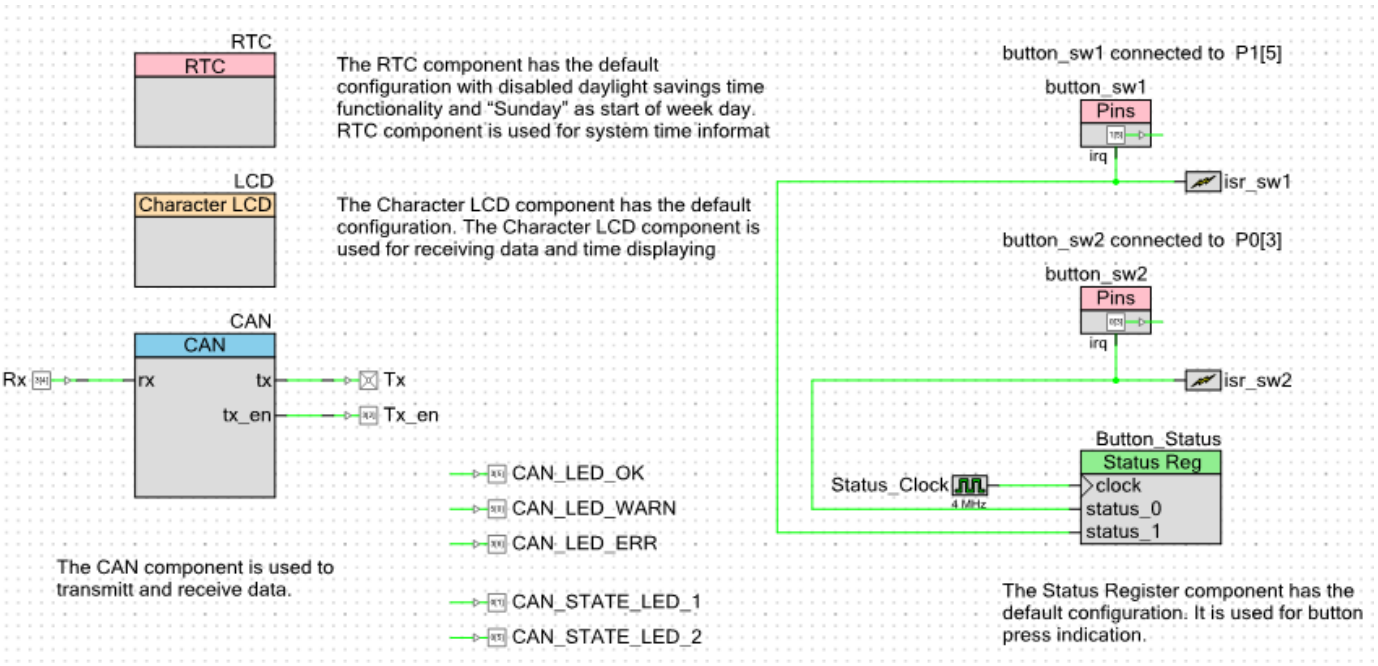

Figure 1. Top design schematic.

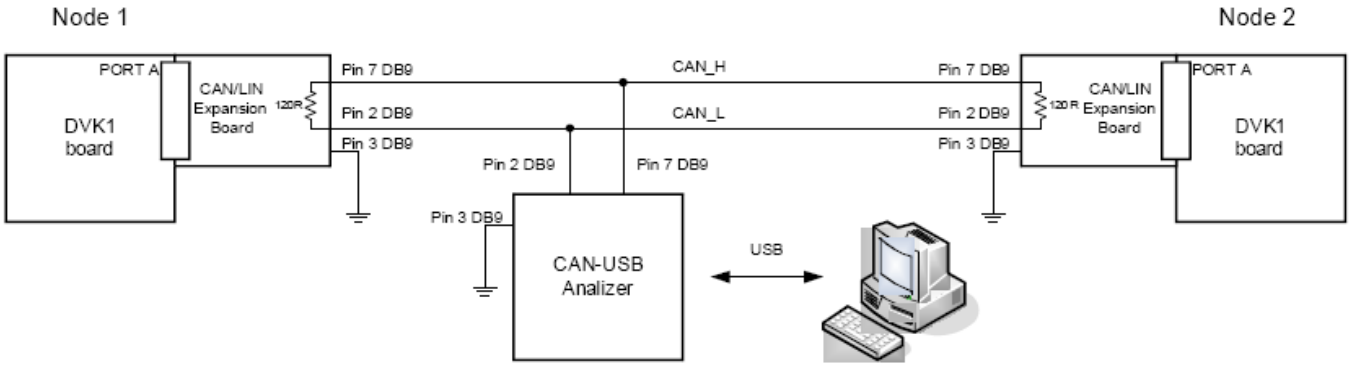

Figure 2. Test CAN network topology.

The Character LCD Component has the default configuration. The RTC Component is configured with daylight savings time functionality disabled and "Sunday" as start of week day.

The Status Register component has the default configuration. It is used for button press indication. The CAN Component configuration is described in Figures 3 - 6 show below:

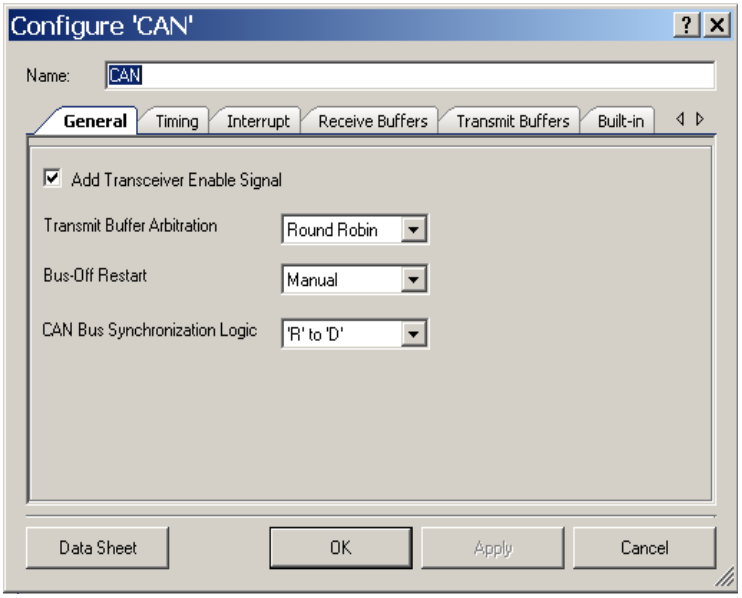

Figure 3. CAN Component general configuration.

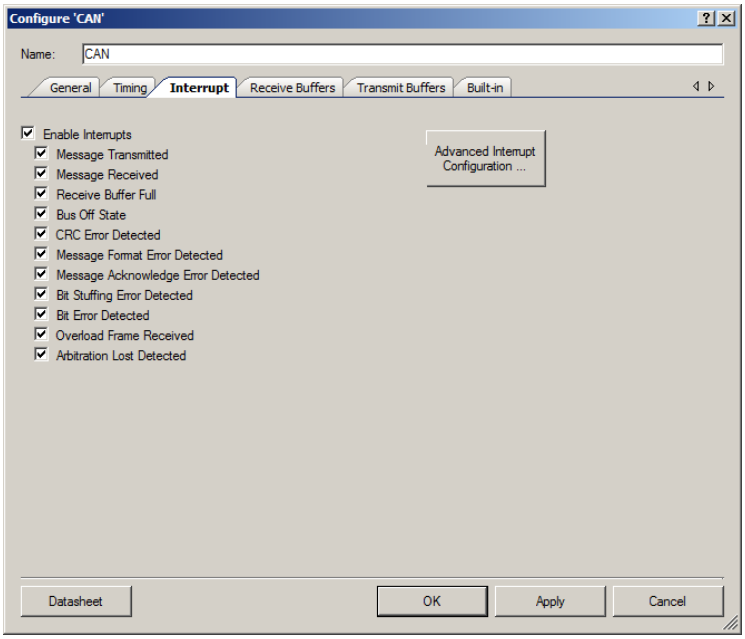

Figure 4. CAN Component interrupt configuration.

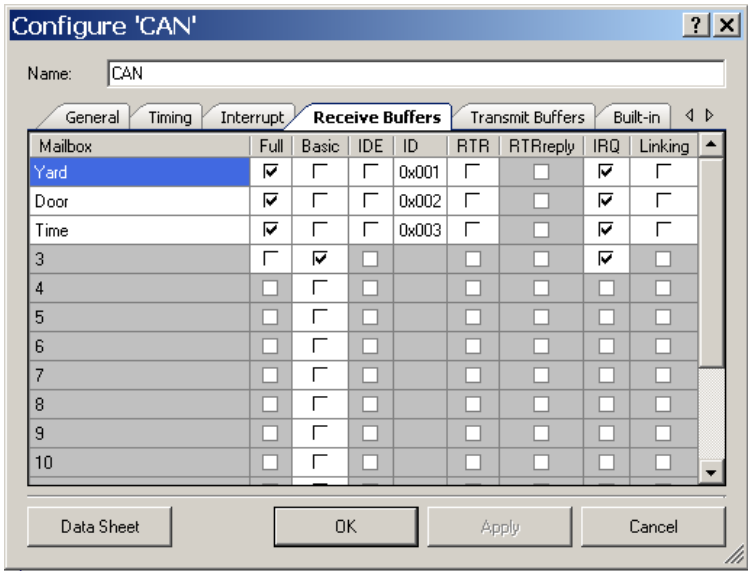

Figure 5. CAN Component receive buffers configuration.

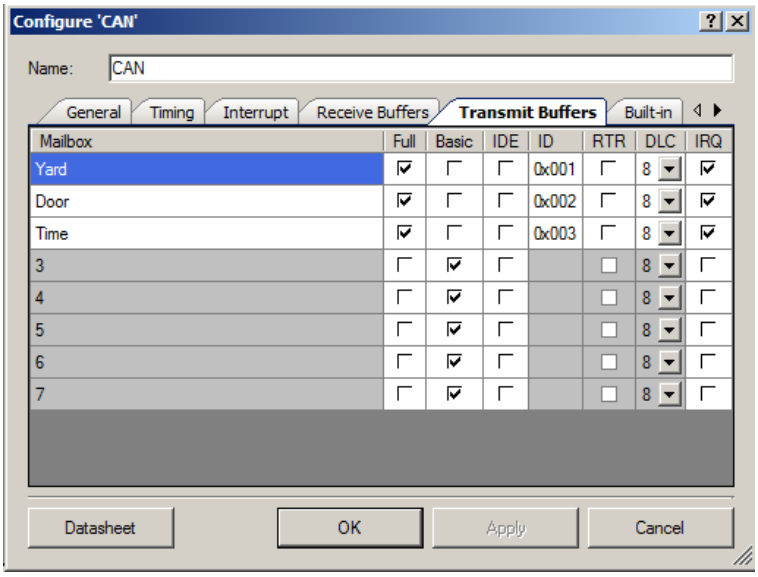

Figure 6. Component transmit buffers configuration.

The clock system configuration is shown in Figure 7. The XTAL 32 kHz should be enabled for proper RTC component operation.

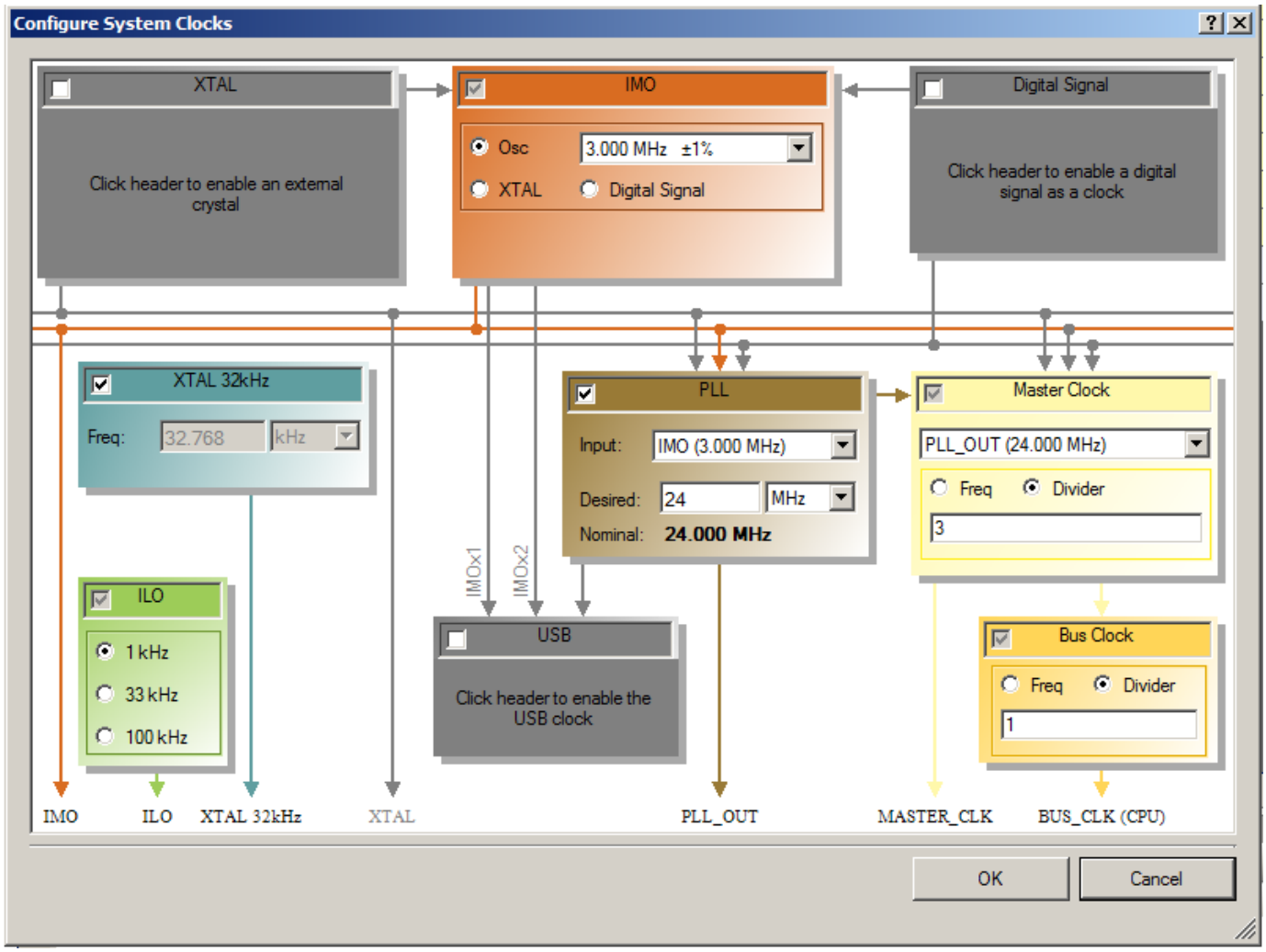

Figure 7. The “Configure System Clocks” window.

Project description

This example of the CAN Component usage suggests a very simple model of a Smart house system. The system consists of two nodes (PSoC chips). One of nodes is the control center, while another node acts as the remote device. The control center manages the system behavior. It monitors the system state and requests from the remote device. The control center also sends commands to the remote device to open or close doors. The remote device drive peripheral is also part of the system. It can send requests to the control center and process replies from it.

Node 1 - Control Center

The control center device stays in the IDLE state and displays the current time while no requests are made from the remote device. The IDLE state is indicated by blinking icons of closed doors. A call from the remote device is displayed on the LCD with a blinking icon near the door icon along with a note. The LCD displays the time of the last call. Pressing the button opens one of the doors (sw1 - yard door, sw2 - house door). After a button is pressed – a blinking icon appears along with a note indicative of an open door, while it sends a message to remote device about the opening of the door and the time of current command. After a certain time, the device turns to the IDLE state and the door is closed.

Node 2 - Remote Device

The remote device stays in the IDLE state and displays the current time while no commands are received from the control center. There are two buttons (bells). Pressing the button causes the bell to ring on one of the doors (sw1 - yard door, sw2 - house door). After receiving a command from the control center, the LCD displays the time when command was received along with an invitation to enter. After a certain time, the device turns to the IDLE state.

Expected results

Display legend is described below:

First string: Node state
Second string: “Curr Time”, “Call Time” and current time from RTC and time received from another node

There are several state of the Nodes as described below:

Node 1 in IDLE state - shown in Figure 8
Node 2 in IDLE state - shown in Figure 9
Call from remote device to control center - shown in Figure 10
Response from control center to remote device - shown in Figure 11

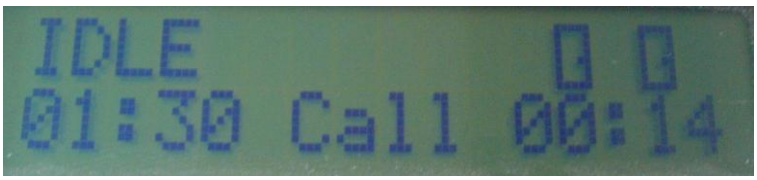

Figure 8. Node 1 (Control Centre) in IDLE state.

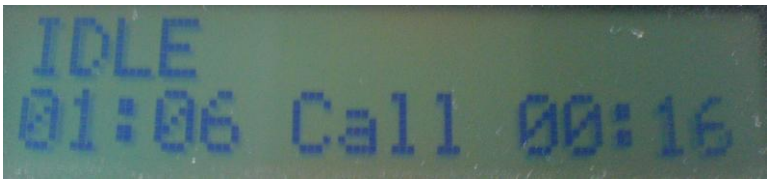

Figure 9. Node 2 (Remote Device) in IDLE state.

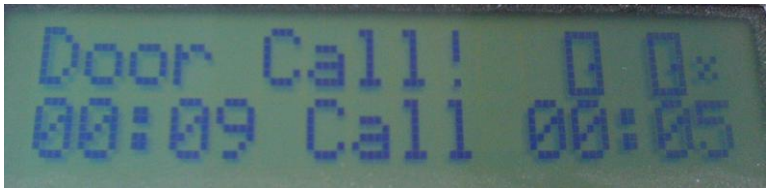

Figure 10. Call from remote device to control center.



Figure 11. Response from control center to remote device.

© Cypress Semiconductor Corporation, 2009-2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.