

# EEPROM example project

## 1.20

## Features

- Erase and write row of data to on-chip EEPROM

## General Description

This example project demonstrates the usage of the EEPROM component. It includes blocking and non-blocking APIs for reading, writing, and erasing the EEPROM memory.

## Development kit configuration

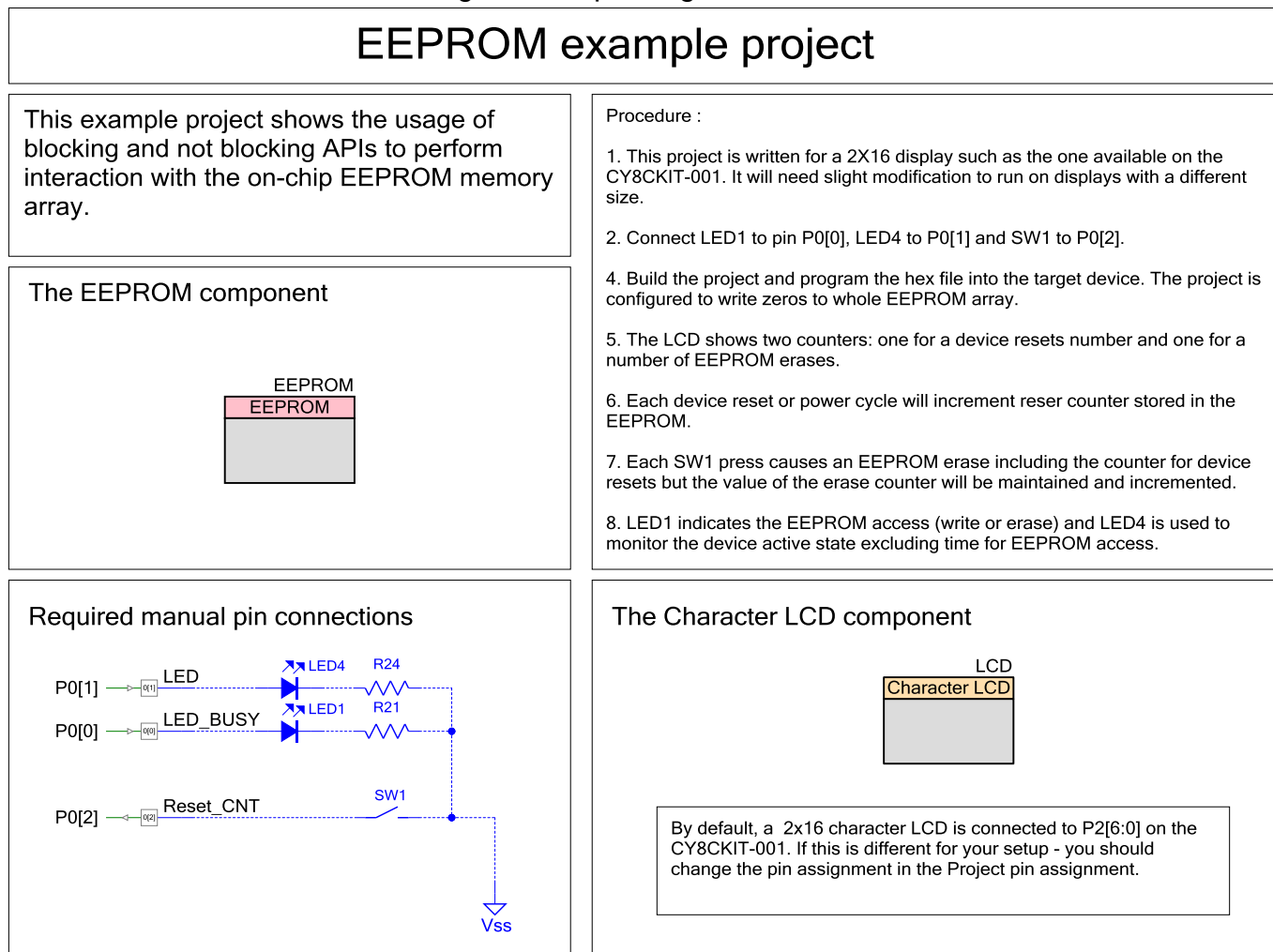
1. This project is written for a 2X16 display such as the one available on the CY8CKIT-001. It will need slight modification to run on displays with a different size.
2. Connect LED1 to pin P0[0], LED4 to P0[1], and SW1 to P0[2].
3. Build the project and program the hex file into the target device. The project is configured to write zeros to the whole EEPROM array.
4. The LCD shows two counters: one for a device resets number and one for a number of EEPROM erases.
5. Each device reset or power cycle increments the reset counter stored in the EEPROM.
6. Each SW1 press causes an EEPROM erase including the counter for device resets but the value of the erase counter will be maintained and incremented.
7. LED1 indicates the EEPROM access (write or erase) and LED4 is used to monitor the device active state excluding time for EEPROM access.

## Project configuration

The example project consists of the EEPROM, pins, and Character LCD components. The EEPROM has no configurable parameters. The EEPROM component can be configured only using API. The top design schematic is shown in Figure 1.

The Character LCD component has its default configuration. It is used to display the data that was written to the EEPROM memory.

Figure 1. Top Design Schematic



## Project description

This is an example of how to use the EEPROM component with designs in PSoC Creator. The EEPROM component provides an API to erase and write to the EEPROM memory. Erase a sector (64 rows) of the EEPROM memory using the `EEPROM_EraseSector()` function. Write a row (16 bytes) of data to the EEPROM using the `EEPROM_Write()` function. To read the EEPROM memory, you can either use an API or perform direct reads. For more information, refer to the component datasheet.

## Expected results

The first row of the character LCD should display the project name and the second will display two counters: one for the reset counter and one for the counter of erases of the EEPROM sector:

```
EEPROM example
```

```
RST=01 ERASE=00
```

The RST value is the reset counter and ERASE is the number of the sector erases performed. Both counters are stored in the EEPROM memory, but number of erases is saved to RAM and incremented on EEPROM erase, and after erasing is completed, it is written to the EEPROM. On each device startup, the value of the reset counter is read from the EEPROM, incremented, saved to the EEPROM and displayed on the LCD.

© Cypress Semiconductor Corporation, 2009-2015. 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.

