ClockMatrix

EEPROM Programming Instructions

Introduction

The 8A3xxxx Family of products from IDT are designed to power-up and operate using internal resources. However there may be cases where it is necessary or convenient to make use of an external I2C serial EEPROM for the device to obtain alternate information from at reset.

This document provides information about the EEPROM hardware connections, how to program the EEPROM using the Clock Matrix chip, and other miscellaneous information.

Contents

1. EEPROM Overview

The ClockMatrix 8A3xxxx product family from Renesas is designed to power-up and operate using internal resources. However, there may be cases where it is necessary or convenient to use an external I2C serial EEPROM for the device in order to obtain alternate information at reset.

The external EEPROM can be used to contain one or more of the following:

- Device configuration information block (Config)
- Micro-controller Firmware Full Application Load (FW Application)
- Micro-controller Firmware Patch (FW Hotfix)

These blocks may appear in any order in the EEPROM, but only one FW Application or one FW Hotfix will be applied. A FW Hotfix may not be applied to a FW Application block loaded from EEPROM; only to the internal (embedded) firmware. Any number of Config blocks are allowed.

Renesas will provide the EEPROM image in the form of a .hex file that is formatted according to the Intel Hex format. Details of the structure of the EEPROM is beyond the scope of this document.

The following instructions describe the hardware connections for the EEPROM, how to program the EEPROM using the Clock Matrix chip, and other miscellaneous information.

2. Hardware Setup

The following is an example setup schematic.

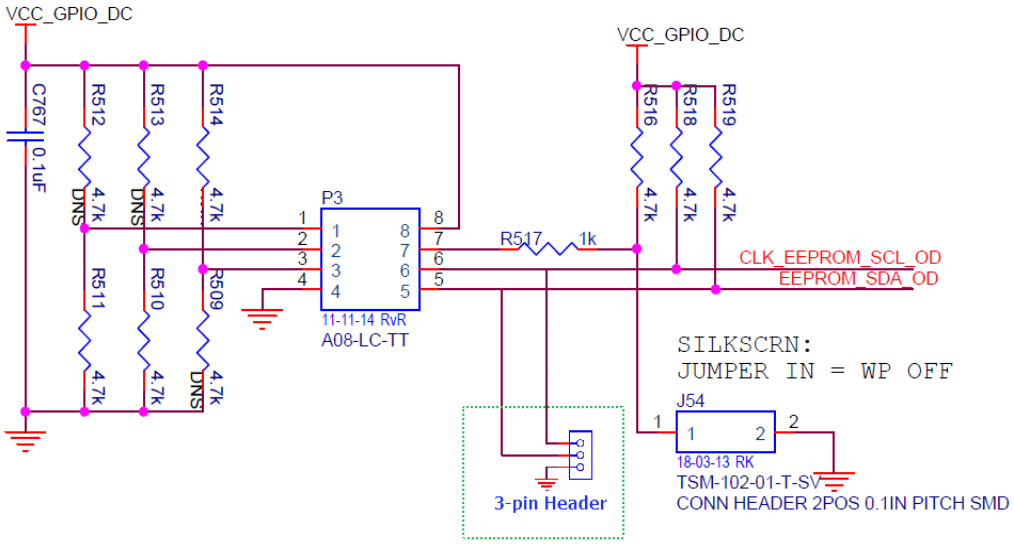

Figure 1. EEPROM Circuit

Connect the SCLK and SDA traces to the ClockMatrix device master I2C port (SCLK and SDA). Another recommendation is to also place a 3-pin header that connects to the EEPROM SCLK, SDA, and GND pin (see green outline in Figure 1). Using this method, an external programmer can be connected to the system for programming. Be aware that external programmers may only support 3.3V levels, so if the VCC_GPIO on the chip is 1.8V or 2.5V a level translator is required.

In the case of prototype boards, a PDIP socket, such as the DILV8P-223TLF can be used (see Figure 2). This allows for easy removal of the EEPROM so that it can be programmed using an external board, such as the Total Phase EEPROM Socket Board.

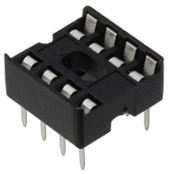

Figure 2. PDIP Socket

3. Programming the EEPROM

There are three options for programming the EEPROM:

- External EEPROM flash card
- Header
- Clock Matrix chip

3.1 External Flash Card Option

For this option, use the PDIP socket. Use the Total Phase EEPROM socket board to program the EEPROM and then place the EEPROM into the socket. For more information, see *Programming the 8A3x000 EEPROM with Total Phase*. Alternatively, another vendor's EEPROM flash card can be used.

3.2 Header Option

For this option, the EEPROM is installed in the system board and an accompanying 3-pin header is used as described in *Programming the 8A3x000 EEPROM with Total Phase*. However, connect the Total Phase Aardvark card directly to the 3-pin header and use the Total Phase software to program the EEPROM.

3.3 Clock Matrix Option

For this option, the ClockMatrix chip serves as the intermediary, transferring the data from the System Device to the EEPROM, as shown in Figure 3. Either a CPU, an FPGA, or a different System device will “talk” to the ClockMatrix device through the Main or Auxiliary Serial port and send it the firmware data. In turn, the ClockMatrix device will send the data to the EEPROM chip. Note that it is the customer's responsibility to parse the .hex file into a format that is usable by their system.

Figure 3. Programming the EEPROM through Clock Matrix

The ClockMatrix device contains a register buffer that functions as an EEPROM read/write buffer. Access to the buffer is obtained using the following registers. For register offset values, see the *8A3xxxx Family Programming Guide* that corresponds to the firmware present in the ROM.

Module: EEPROM

Access EEPROM.

Table 369: EEPROM Register Index

Offset (Hex)	Register Module Base Address: CF68h	
	Individual Register Name	Register Description
000h	EEPROM.EEPROM_I2C_ADDR	EEPROM I2C address.
001h	EEPROM.EEPROM_SIZE	EEPROM data transfer size.
002h	EEPROM.EEPROM_OFFSET	EEPROM offset.
004h	EEPROM.EEPROM_CMD	EEPROM command.

EEPROM.EEPROM_SIZE

Configure the number of bytes to read or write.

Table 371: EEPROM.EEPROM_SIZE Bit Field Locations and Descriptions

Offset Address (Hex)	EEPROM.EEPROM_SIZE Bit Field Locations							
	D7	D6	D5	D4	D3	D2	D1	D0
001h	BYTES[7:0]							

EEPROM.EEPROM_SIZE Bit Field Descriptions			
Bit Field Name	Field Type	Default Value	Description
BYTES[7:0]	R/W	0	Number of bytes to read or write. These bytes are transferred through OTP_EEPROM_BUFF. Valid range is 0 to 128.

EEPROM.EEPROM_I2C_ADDR

EEPROM I2C address.

Table 370: EEPROM.EEPROM_I2C_ADDR Bit Field Locations and Descriptions

Offset Address (Hex)	EEPROM.EEPROM_I2C_ADDR Bit Field Locations							
	D7	D6	D5	D4	D3	D2	D1	D0
000h	RESERVED	I2C_ADDR[6:0]						

EEPROM.EEPROM_I2C_ADDR Bit Field Descriptions			
Bit Field Name	Field Type	Default Value	Description
RESERVED	N/A	-	This field must not be modified from the read value
I2C_ADDR[6:0]	R/W	0	I2C address of the EEPROM.

EEPROM.EEPROM_OFFSET

Table 365: EEPROM.EEPROM_OFFSET Bit Field Locations and Descriptions

Offset Address (Hex)	EEPROM.EEPROM_OFFSET Bit Field Locations							
	D7	D6	D5	D4	D3	D2	D1	D0
002h	EEPROM_OFFSET[7:0]							
003h	EEPROM_OFFSET[15:8]							

EEPROM.EEPROM_OFFSET Bit Field Descriptions			
Bit Field Name	Field Type	Default Value	Description
EEPROM_OFFSET[15:0]	R/W	0	-

EEPROM.EEPROM_CMD

Table 366: EEPROM.EEPROM_CMD Bit Field Locations and Descriptions

Offset Address (Hex)	EEPROM.EEPROM_CMD Bit Field Locations							
	D7	D6	D5	D4	D3	D2	D1	D0
004h	EEPROM_CMD[7:0]							
005h	EEPROM_CMD[15:8]							

EEPROM.EEPROM_CMD Bit Field Descriptions			
Bit Field Name	Field Type	Default Value	Description
EEPROM_CMD[15:0]	R/W	0	- EE01 = Read from EEPROM EE02 = Write to EEPROM EE03 = Write (no verify)

GENERAL_STATUS.EEPROM_STATUS

Table 11: GENERAL_STATUS.EEPROM_STATUS Bit Field Locations and Descriptions

Offset Address (Hex)	GENERAL_STATUS.EEPROM_STATUS Bit Field Locations							
	D7	D6	D5	D4	D3	D2	D1	D0
008h	EEPROM_STATUS[7:0]							
009h	EEPROM_STATUS[15:8]							

GENERAL_STATUS.EEPROM_STATUS Bit Field Descriptions			
Bit Field Name	Field Type	Default Value	Description
EEPROM_STATUS[15:0]	R/O	0	Current status of EEPROM. 0x0000 = no status 0x8000 = ok 0x8001 = unknown command 0x8002 = wrong size 0x8003 = out of range 0x8004 = read failed 0x8005 = write failed 0x8006 = verification failed

The EEPROM buffer register is 128 bytes long, and is the location where the data is either written to the EEPROM or read from the EEPROM. For PR4.0, the location is 0xCF80.

The following is an example of an SPI 2-byte addressing sequence that shows how to program the first 128 bytes of an EEPROM. This example is based on PR4.0 Firmware. For a different firmware, see the appropriate *8A3xxxx Family Programming Guide*.

3.4 Write EEPROM Example

```
7F FD 80 10 20  #Set the Clock Matrix page register (SPI 2-byte addressing)
4F 68 50        #Write to CF68, set EEPROM_I2C_ADDR=0x50
4F 69 80        #Write to CF69, set EEPROM_SIZE=128 bytes.
4F 6A 00 00     #Write to CF6A, set EEPROM_OFFSET= 0 (write starting at location 0).
4F 80 85 AA 85 AA 84 AA 84 AA 83 AA 83 AA 82 AA 82 AA 81 AA 81 AA 80 AA 80 AA 79 AA 79 AA
78 AA 78 AA 77 AA 77 AA 76 AA 76 AA 75 AA 75 AA 74 AA 74 AA 73 AA 73 AA 72 AA 72 AA 71 AA
71 AA 70 AA 70 AA 69 AA 69 AA 68 AA 68 AA 67 AA 67 AA 66 AA 66 AA 65 AA 65 AA 64 AA 64 AA
63 AA 63 AA 62 AA 62 AA 61 AA 61 AA 60 AA 60 AA 59 AA 59 AA 58 AA 58 AA 57 AA 57 AA 56 AA
56 AA 55 AA 55 AA 00 00 00 00
             #Write to CF80, these are the 128 bytes that will be written to the EEPROM.
4F 6C 02 EE  #Write to CF6C, EEPROM_CMD="0xEE02", trigger EEPROM write
```

#add a wait time to allow the chip to write to the EEPROM. For now, use 100mS. Alternatively, read back the EEPROM_STATUS and verify that it is 0x8000 and then proceed.

Repeat this for each 128 bytes of EEPROM data.

3.5 Read EEPROM Example

```
4F 6A 00 00     #Write to CF6A, set offset
4F 6C 01 EE     #Write to CF6C, trigger EEPROM read
#add a wait time to allow the chip to write to the EEPROM. For now, use 100mS.

7F FD 80 10 20  #Set page register
                #Read from CF80, read back buffer (note that MSB is set for read commands)
CF 80 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00
```

For instructions on how to update the firmware on an EEPROM on a ClockMatrix evaluation board, see the document *ClockMatrix Evaluation Board: Update Firmware on Device and EEPROM*.

4. Revision History

Rev.	Date	Description
1.00	Feb.10.2020	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers skilled in the art designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only for development of an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising out of your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use o any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.0 Mar 2020)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan

www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit:

www.renesas.com/contact/

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.