Freescale Semiconductor
Application Note

AN4536
Rev. 3.0, 1/2014

MMPF0100 OTP Programming Instructions

1 Introduction

This document provides a detailed description of the One-Time Programmable (OTP) function of the MMPF0100. It provides the system requirements and instructions to program the fuses for a selected power-up configuration. All examples assume the customer is using silicon revision P1.1 or higher.

Freescale analog ICs are manufactured using the SMARTMOS process, a combinational BiCMOS manufacturing flow that integrates precision analog, power functions and dense CMOS logic together on a single cost-effective die.

Contents

2 OTP Overview

The regulators in the MMPF0100 have been designed with flexibility and configurability to allow them to be adapted for a wide variety of applications. One-Time-Programmable (OTP) fuses in the MMPF0100 are used to exercise this flexibility. Key startup parameters and regulator configuration information can be programmed into the MMPF0100 to enable it to power the system. These parameters are:

- **General**: I^{2}C slave address, PWRON pin configuration, regulator start-up sequence and timing, RESETBMCU configuration
- **Buck regulators**: Output voltage, dual/single phase or independent mode configuration, switching frequency, and soft start ramp rate
- **Boost regulator and LDOs**: Output voltage

MMPF0100 starts up based on the contents of the TBBOTP registers. The TBBOTP registers can be loaded from different sources as shown in Table 1. The default setting is hard-coded in the MMPF0100 and is available in all non-programmed and programmed MMPF0100 devices. Once OTP programming is complete, TBBOTP can be either loaded from the default values or from the OTP fuses.

The OTP block in the MMPF0100 also features a ‘Try-Before-Buy’ (TBB) mode which allows experimentation with different voltages and sequences of the regulators. In the TBB mode, TBBOTP registers are directly written to and used for startup of the MMPF0100. Contents of the TBBOTP registers can be maintained in the absence of the main input supply, VIN, by using a coin cell at the LICELL pin.

2.1 Power-up Configuration

The PF0100 powers up based on the contents of the TBBOTP registers. Depending on certain pin and bit settings, the TBBOTP registers are loaded from different sources as shown in Table 1.

Table 1. Power-up Configuration Source and Conditions

Source	Condition	Power-Up Configuration
ROM	VDDOTP = VCOREDIG(1)	MMPF0100 starts up using the factory default settings
TBBOTP Registers	VDDOTP = 0 V and TBB_POR = 1	MMPF0100 starts up from current values of TBBOTP registers. This is referred to as the 'Try-Before-Buy' mode
OTP Fuses	VDDOTP = 0 V and TBB_POR = 0 and FUSE_POR_XOR = 1(2)	The MMPF0100 starts up from the OTP fuse values

Notes:

1. Pull-up VDDOTP to VCOREDIG with a 100 k resistor.
2. In MMPF0100, FUSE_POR1, FUSE_POR2 and FUSE_POR3 are XOR’ed into the FUSE_POR_XOR bit. The FUSE_POR_XOR has to be 1 for fuses to be loaded. This can be achieved by setting any one or all of the FUSE_PORx bits. In MMPF0100A, the XOR function is removed. It is required to set all of the FUSE_PORx bits to be able to load the fuses.

The TBBOTP registers serve as temporary storage for any of the following:

1. The values to be written to the fuses, or
2. The values read from the fuses, or
3. The values to start from during TBB development, or
4. The values read from the default configuration.

The TBBOTP registers are located within the Extended Page 1 of the MMPF0100 register map.

During a power-up, the TBBOTP registers behave as follows:

- The contents of the TBBOTP registers are initialized to zero when a valid VIN is first applied.
- The values that are then loaded into the TBBOTP registers depend on the setting of the VDDOTP pin, and on the value of the TBB_POR, and the FUSE_POR_XOR bits. Refer to Table 1.
- If VDDOTP = VCOREDIG (1.5 V), the TBBOTP values are loaded from ROM.
- If VDDOTP = 0 V, TBB_POR = 0 and FUSE_POR_XOR = 1; the TBBOTP values are loaded from the fuses.
- If VDDOTP = 0, TBB_POR = 0 and FUSE_POR_XOR=0; the TBBOTP registers remain initialized at zero.
- The initial value of TBB_POR is always “0”, only when VDDOTP = 0 V and TBB_POR is set to “1”, are the values from the TBBOTP registers maintained and not loaded from a different source.

The contents of the TBBOTP registers may be modified by I^{2}C. To communicate with I^{2}C, VIN must be valid and VDDIO, to which SDA and SCL are pulled up, must be powered by a 1.7 V to 3.6 V supply. VIN or the coin cell voltage must be valid to maintain the contents of the TBBOTP registers. To power on with the contents of the TBBOTP registers, a valid turn-on event must occur with the following conditions: a valid VIN, optional LICELL, VDDOTP = 0 V, TBB_POR = 1.

2.2 OTP Programming Example

The One-Time-Programmable memory is realized using fuses. The startup configuration can be programmed into the MMPF0100 by changing the state of these fuses as required during the OTP programming process.

There are 10 banks of fuses with each bank consisting of 26 fuses. Of the 26 fuses in a bank, 20 are programmable by the user. The remaining 6 are redundant fuses that allow implementation of Error Correction. An Error Correction Code within the MMPF0100 corrects single bit errors if they occur in the bank.

The following is an example of programming the MMPF0100 and MMPF0100A. MMPF0100A refers to the newer silicon version of MMPF0100. Refer to the product Data Sheet for complete details.

Note that the programming voltage and time-delay during fuse programming is different between the two silicon revisions. The programming voltage should have a tolerance of +/-3% and OTP programming should be done at room temperature. For reliability reasons, do not OTP program a given part more than once.

Note: All code examples in this document represent a script using the KITPFPGMEVME and the associated GUI. Command syntax may vary if the user utilizes a different tool for communication.

```
//----------------------------------------------------------------------------
// F0 - Sample Configuration
// Set VDDOTP = 0 V, PWRON = HIGH, LICELL = 3.0 V (Optional), VIN = VDDIO = 3.3 V
//----------------------------------------------------------------------------
WRITE_I2C:7F:01 // Access PF0100 EXT Page1
//[Extended Page 1 Registers: 0xA0 - 0xAF] ----------------------------------
WRITE_I2C:A0:2B // Sw1AB Voltage = 1.375 V
WRITE_I2C:A1:01 // Sw1AB Sequence = 1
WRITE_I2C:A2:05 // Sw1AB Freq = 2MHZ, Mode = Single phase
WRITE_I2C:A8:2B // Sw1c Voltage = 1.375 V
WRITE_I2C:A9:02 // Sw1c Sequence = 2
WRITE_I2C:AA:01 // Sw1c Freq = 2 MHZ
WRITE_I2C:AC:72 // Sw2 Voltage = 3.30 V
WRITE_I2C:AD:05 // Sw2 Sequence = 5
WRITE_I2C:AE:01 // Sw2 Freq = 2 MHZ
//[Extended Page 1 Registers: 0xB0 - 0xBF] ----------------------------------
WRITE_I2C:B0:2C // Sw3A Voltage = 1.500 V
```

```
WRITE_I2C:B1:03 // Sw3A Sequence = 3
WRITE_I2C:B2:05 // Sw3A Freq = 2 MHZ, Mode = Single phase
WRITE_I2C:B4:2C // Sw3B Voltage = 1.500 V
WRITE_I2C:B5:03 // Sw3B Sequence = 3
WRITE_I2C:B6:01 // Sw3B Freq = 2 MHZ
WRITE_I2C:B8:6F // Sw4 Voltage = 3.150 V
WRITE_I2C:B9:06 // Sw4 Sequence = 6
WRITE_I2C:BA:01 // Sw4 Freq = 2 MHZ
WRITE_I2C:BC:00 // Swbst Voltage = 5 V
WRITE_I2C:BD:0D // Swbst Sequence = 13
//[Extended Page 1 Registers: 0xC0 - 0xCF] ---------------------------------
WRITE_I2C:C0:06 // Vsnvs Voltage = 3 V
WRITE_I2C:C4:03 // Vsnvs Sequence = 3
WRITE_I2C:C8:0E // Vgen1 Voltage = 1.50 V
WRITE_I2C:C9:09 // Vgen1 Sequence = 9
WRITE_I2C:CC:0E // Vgen2 Voltage = 1.5 V
WRITE_I2C:CD:0A // Vgen2 Sequence = 10
//[Extended Page 1 Registers: 0xD0 - 0xDF] ---------------------------------
WRITE_I2C:D0:07 // Vgen3 Voltage = 2.5 V
WRITE_I2C:D1:0B // Vgen3 Sequence = 11
WRITE_I2C:D4:00 // Vgen4 Voltage = 1.8 V
WRITE_I2C:D5:07 // Vgen4 Sequence = 7
WRITE_I2C:D8:0A // Vgen5 Voltage = 2.8 V
WRITE_I2C:D9:0C // Vgen5 Sequence = 12
WRITE_I2C:DC:0F // Vgen6 Voltage = 3.3 V
WRITE_I2C:DD:08 // Vgen6 Sequence = 8
//[Extended Page 1 Registers: 0xE0 - 0xEF] ---------------------------------
WRITE_I2C:E0:0E // Power-up DVS = 1.5625 mV/us, SeqCLK = 2 ms, PWRON config = 0
WRITE_I2C:E1:0E // Power-up DVS = 1.5625 mV/us, SeqCLK = 2 ms, PWRON config = 0
WRITE_I2C:E2:0E // Power-up DVS = 1.5625 mV/us, SeqCLK = 2 ms, PWRON config = 0
WRITE_I2C:E8:00 // Power Good = Disabled
//[Extended Page 1 Registers: 0xF0 - 0xFF] ---------------------------------
WRITE_I2C:FF:08 // I2C Device Address = 0x08
//============================================================================
// PROGRAMMING COMMANDS FOLLOW
//============================================================================
WRITE_I2C:E4:02 // FUSE POR=1 (This Enables OTP Programming)
WRITE_I2C:E5:02 // FUSE POR=1 (This Enables OTP Programming)
WRITE_I2C:E6:02 // FUSE POR=1 (This Enables OTP Programming)
//----------------------------------------------------------------------------
WRITE_I2C:F0:1F // Enable ECC for fuse banks 1 to 5
WRITE_I2C:F1:1F // Enable ECC for fuse banks 6 to 10
WRITE_I2C:7F:02 // Access PF0100 EXT Page2
```

```
WRITE_I2C:D0:1F // Set Auto ECC for fuse banks 1 to 5
WRITE_I2C:D1:1F // Set Auto ECC for fuse banks 6 to 10
//-----------------------------------------------------------------------------
WRITE_I2C:F1:00 // Reset Bank 1 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F2:00 // Reset Bank 2 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F3:00 // Reset Bank 3 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F4:00 // Reset Bank 4 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F5:00 // Reset Bank 5 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F6:00 // Reset Bank 6 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F7:00 // Reset Bank 7 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F8:00 // Reset Bank 8 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F9:00 // Reset Bank 9 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:FA:00 // Reset Bank 10 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-----------------------------------------------------------------------------
VPGM:ON // Turn ON 9.5 V supply for PF0100A. Turn ON 9.0 V supply for PF0100.
// VPGM:ON turns on supply to the VDDOTP pin
DELAY:500 // Adds 500 msec delay to allow VPGM time to ramp up
//-----------------------------------------------------------------------------
// PF0100 OTP MANUAL-PROGRAMMING (BANK 1 thru 10)
//-----------------------------------------------------------------------------
// BANK 1
//-----------------------------------------------------------------------------
WRITE_I2C:F1:03 // Set Bank 1 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F1:0B // Set Bank 1 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F1:03 // Reset Bank 1 ANTIFUSE_EN
WRITE_I2C:F1:00 // Reset Bank 1 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-----------------------------------------------------------------------------
// BANK 2
//-----------------------------------------------------------------------------
WRITE_I2C:F2:03 // Set Bank 2 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F2:0B // Set Bank 2 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F2:03 // Reset Bank 2 ANTIFUSE_EN
WRITE_I2C:F2:00 // Reset Bank 2 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-----------------------------------------------------------------------------
// BANK 3
//-----------------------------------------------------------------------------
WRITE_I2C:F3:03 // Set Bank 3 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F3:0B // Set Bank 3 ANTIFUSE_EN
DELAY:100 // Allow 100ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F3:03 // Reset Bank 3 ANTIFUSE_EN
WRITE_I2C:F3:00 // Reset Bank 3 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
```

```
//-------------------------------------------------------------------------
// BANK 4
//-------------------------------------------------------------------------
WRITE_I2C:F4:03 // Set Bank 4 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F4:0B // Set Bank 4 ANTIFUSE_EN
DELAY:100 // Allow 100ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F4:03 // Reset Bank 4 ANTIFUSE_EN
WRITE_I2C:F4:00 // Reset Bank 4 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-------------------------------------------------------------------------
// BANK 5
//-------------------------------------------------------------------------
WRITE_I2C:F5:03 // Set Bank 5 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F5:0B // Set Bank 5 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F5:03 // Reset Bank 5 ANTIFUSE_EN
WRITE_I2C:F5:00 // Reset Bank 5 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-------------------------------------------------------------------------
// BANK 6
//-------------------------------------------------------------------------
WRITE_I2C:F6:03 // Set Bank 6 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F6:0B // Set Bank 6 ANTIFUSE_EN
DELAY:100 // Allow 100ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F6:03 // Reset Bank 6 ANTIFUSE_EN
WRITE_I2C:F6:00 // Reset Bank 6 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-------------------------------------------------------------------------
// BANK 7
//-------------------------------------------------------------------------
WRITE_I2C:F7:03 // Set Bank 7 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F7:0B // Set Bank 7 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F7:03 // Reset Bank 7 ANTIFUSE_EN
WRITE_I2C:F7:00 // Reset Bank 7 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-------------------------------------------------------------------------
// BANK 8
//-------------------------------------------------------------------------
WRITE_I2C:F8:03 // Set Bank 8 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F8:0B // Set Bank 8 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F8:03 // Reset Bank 8 ANTIFUSE_EN
WRITE_I2C:F8:00 // Reset Bank 8 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//-------------------------------------------------------------------------
// BANK 9
//-------------------------------------------------------------------------
```

```
WRITE_I2C:F9:03 // Set Bank 9 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:F9:0B // Set Bank 9 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:F9:03 // Reset Bank 9 ANTIFUSE_EN
WRITE_I2C:F9:00 // Reset Bank 9 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//------------------------------------------------------------------------------
// BANK 10
//------------------------------------------------------------------------------
WRITE_I2C:FA:03 // Set Bank 10 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
WRITE_I2C:FA:0B // Set Bank 10 ANTIFUSE_EN
DELAY:100 // Allow 100 ms for PF0100A. Use 50 ms for PF0100.
WRITE_I2C:FA:03 // Reset Bank 10 ANTIFUSE_EN
WRITE_I2C:FA:00 // Reset Bank 10 ANTIFUSE_RW and ANTIFUSE_BYPASS bits
//------------------------------------------------------------------------------
WRITE_I2C:D0:00 // Clear
WRITE_I2C:D1:00 // Clear
//------------------------------------------------------------------------------
VPGM:OFF // Turn off 8.5 V Boost Supply
DELAY:500 // Adds delay to allow VPGM to bleed off
PWRON:LOW // PWRON LOW to reload new OTP data
DELAY:500
PWRON:HIGH
```

After OTP programming is complete by following the above steps, read the registers 0xA0 to 0xE8 in Extended Page 1 and compare to the required register values as in the script. Additionally, read the ECC Interrupt bit, OTP_ECCI in register 0x0E. If there is an error in the programmed values or if the OTP_ECCI bit is set to 1, reject the part as the programming process resulted in errors.

2.3 Try-Before-Buy Mode Example

As shown in Table 1, it is possible to start the MMPF0100 directly from the TBBOTP registers without actually programming the part. Shown below is an example of the Try-Before-Buy mode.

Note: All code examples in this document represent a script using the KITPFPGMEVME and the associated GUI. Command syntax may vary if the user utilizes a different tool for communication.

```
///-----------------------------------------------------------------------------
// F0 – Sample Try-Before-Buy Configuration
// Set VDDOTP = 0 V, PWRON = HIGH, LICELL = 3.0 V (Optional), VIN = VDDIO = 3.3 V
//-----------------------------------------------------------------------------
WRITE_I2C:7F:01 // Access PF0100 EXT Page1
//[Extended Page 1 Registers: 0xA0 - 0xAF] ----------------------------------
WRITE_I2C:A0:2B // Sw1AB Voltage = 1.375 V
WRITE_I2C:A1:01 // Sw1AB Sequence = 1
WRITE_I2C:A2:05 // Sw1AB Freq = 2 MHZ, Mode = Single phase
WRITE_I2C:A8:2B // Sw1c Voltage = 1.375 V
WRITE_I2C:A9:02 // Sw1c Sequence = 2
WRITE_I2C:AA:01 // Sw1c Freq = 2.0 MHZ
WRITE_I2C:AC:72 // Sw2 Voltage = 3.30 V
WRITE_I2C:AD:05 // Sw2 Sequence = 5
WRITE_I2C:AE:01 // Sw2 Freq = 2 MHZ
//[Extended Page 1 Registers: 0xB0 - 0xBF] ----------------------------------
WRITE_I2C:B0:2C // Sw3A Voltage = 1.500 V
WRITE_I2C:B1:03 // Sw3A Sequence = 3
WRITE_I2C:B2:05 // Sw3A Freq = 2 MHZ, Mode = Single phase
WRITE_I2C:B4:2C // Sw3B Voltage = 1.500 V
WRITE_I2C:B5:03 // Sw3B Sequence = 3
WRITE_I2C:B6:01 // Sw3B Freq = 2 MHZ
WRITE_I2C:B8:6F // Sw4 Voltage = 3.150 V
WRITE_I2C:B9:06 // Sw4 Sequence = 6
WRITE_I2C:BA:01 // Sw4 Freq = 2 MHZ
WRITE_I2C:BC:00 // Swbst Voltage = 5.0 V
WRITE_I2C:BD:0D // Swbst Sequence = 13
//[Extended Page 1 Registers: 0xC0 - 0xCF] ----------------------------------
WRITE_I2C:C0:06 // Vsnvs Voltage = 3.0 V
WRITE_I2C:C4:03 // Vsnvs Sequence = 3
WRITE_I2C:C8:0E // Vgen1 Voltage = 1.50 V
WRITE_I2C:C9:09 // Vgen1 Sequence = 9
WRITE_I2C:CC:0E // Vgen2 Voltage = 1.5 V
WRITE_I2C:CD:0A // Vgen2 Sequence = 10
//[Extended Page 1 Registers: 0xD0 - 0xDF] ----------------------------------
WRITE_I2C:D0:07 // Vgen3 Voltage = 2.5 V
WRITE_I2C:D1:0B // Vgen3 Sequence = 11
```

```
WRITE_I2C:D4:00 // Vgen4 Voltage = 1.8 V
WRITE_I2C:D5:07 // Vgen4 Sequence = 7
WRITE_I2C:D8:0A // Vgen5 Voltage = 2.8 V
WRITE_I2C:D9:0C // Vgen5 Sequence = 12
WRITE_I2C:DC:0F // Vgen6 Voltage = 3.3 V
WRITE_I2C:DD:08 // Vgen6 Sequence = 8
//[Extended Page 1 Registers: 0xE0 - 0xEF] ----------------------------------
WRITE_I2C:E0:0E // Power-up DVS = 1.5625 mV/us, SeqCLK = 2 ms, PWRON config = 0
WRITE_I2C:E1:0E // Power-up DVS = 1.5625 mV/us, SeqCLK = 2 ms, PWRON config = 0
WRITE_I2C:E2:0E // Power-up DVS = 1.5625 mV/us, SeqCLK = 2 ms, PWRON config = 0
WRITE_I2C:E8:00 // Power Good = Disabled
//[Extended Page 1 Registers: 0xF0 - 0xFF] ----------------------------------
WRITE_I2C:FF:08 // I2C Device Address = 0x08
//==========================================================================
// TRY-BEFORE-BUY COMMANDS FOLLOW
//==========================================================================
WRITE_I2C:E4:80 // TBB POR=1 (This Enables TBB Mode)
//--------------------------------------------------------------------------
// BANK 9
//--------------------------------------------------------------------------
PWRON:LOW // PWRON LOW
DELAY:500
PWRON:HIGH // PWRON HIGH to Start PMIC from desired TBB Configuration
```

Optionally, PWRON on can be kept HIGH and VIN can be toggled when a valid LICELL is present.

2.4 OTP Registers Description

There are ten banks with a total of 260 fuses, where each bank contains 26 fuses. Each fuse represents one bit of the TBBOTP register map. Table 2 to Table 11 show the banks, their fuses and the corresponding bits in the register map.

Table 2. Bank 1

Fuses	Register Name	Register bits	Description
5:0	OTP SW1AB VOLT	SW1AB_VOLT[5:0]	SW1AB power-up voltage
6	–	–	RSVD
11:7	OTP SW1AB SEQ	SW1AB_SEQ[4:0]	SW1AB power-up sequence
13:12	OTP SW1AB CONFIG	SW1AB_FREQ[1:0]	SW1AB power-up frequency
15:14	OTP SW1AB CONFIG	SW1AB_CONFIG[1:0]	SW1A/B/C power-up configuration
18:16	OTP I2C ADDR	I2C_SLV_ADDR[3:0]	3 LSBs of the slave address
19	OTP EN ECC0	EN_ECC_BANK1	Enable ECC for OTP fuse bank 1
25:20	–	–	ECC check bits for fuse bank 1

Table 3. Bank 2

Fuses	Register Name	Register bits	Description
5:0	OTP SW1C VOLT	SW1C_VOLT[5:0]	SW1C power-up voltage
6	–	–	RSVD
11:7	OTP SW1C SEQ	SW1C_SEQ[4:0]	SW1C power-up sequence
13:12	OTP SW1C CONFIG	SW1C_FREQ[1:0]	SW1C power-up frequency
15:14	OTP SWBST VOLT	SWBST[1:0]	SWBST power-up voltage
18:16	–	–	RSVD
19	OTP EN ECC0	EN_ECC_BANK2	Enable ECC for OTP fuse bank 2
25:20	–	–	ECC check bits for fuse bank 2

Table 4. Bank 3

Fuses	Register Name	Register bits	Description
6:0	OTP SW2 VOLT	SW2_VOLT6:0]	SW2 power-up voltage
11:7	OTP SW2 SEQ	SW2_SEQ[4:0]	SW2 power-up sequence
13:12	OTP SW2 CONFIG	SW2_FREQ[1:0]	SW2 power-up frequency
18:14	OTP SWBST SEQ	SWBST_SEQ[4:0]	SWBST power-up sequence
19	OTP EN ECC0	EN_ECC_BANK3	Enable ECC for OTP fuse bank 3
25:20	–	–	ECC check bits for fuse bank 3

Table 5. Bank 4

Fuses	Register Name	Register bits	Description
6:0	OTP SW3A VOLT	SW3A_VOLT[6:0]	SW3A power-up voltage
11:7	OTP SW3A SEQ	SW3A_SEQ[4:0]	SW3A power-up sequence
13:12	OTP SW3A CONFIG	SW3A_FREQ[1:0]	SW3A power-up frequency
15:14	OTP SW3A CONFIG	SW3_CONFIG	SW3A/B power-up configuration
18:16	–	–	RSVD
19	OTP EN ECC0	EN_ECC_BANK4	Enable ECC for OTP fuse bank 4
25:20	–	–	ECC check bits for fuse bank 4

Table 6. Bank 5

Fuses	Register Name	Register bits	Description
6:0	OTP SW3B VOLT	SW3B_VOLT[6:0]	SW3B power-up voltage
11:7	OTP SW3B SEQ	SW3B_SEQ[4:0]	SW3B power-up sequence
13:12	OTP SW3B CONFIG	SW3B_FREQ[1:0]	SW3B power-up frequency
18:14	OTP VREFDDR SEQ	VREFDDR_SEQ[4:0]	VREFDDR power-up sequence
19	OTP EN ECC0	EN_ECC_BANK5	Enable ECC for OTP fuse bank 5
25:20	–	–	ECC check bits for fuse bank 5

Table 7. Bank 6

Fuses	Register Name	Register bits	Description
6:0	OTP SW 4 VOLT	SW4_VOLT[6:0]	SW4 power-up voltage
11:7	OTP SW 4 SEQ	SW4_SEQ[4:0]	SW4 power-up sequence
13:12	OTP SW 4 CONFIG	SW4_FREQ[1:0]	SW4 power-up frequency
14	OTP SW 4 CONFIG	VTT	SW4 tracking mode select
17:15	OTP VSNVS VOLT	VSNVS_VOLT[2:0]	VSNVS power-up voltage
18	–	–	RSVD
19	OTP EN ECC1	EN_ECC_BANK6	Enable ECC for OTP fuse bank 6
25:20	–	–	ECC check bits for fuse bank 6

Table 8. Bank 7

Fuses	Register Name	Register bits	Description
3:0	OTP VGEN1 VOLT	VGEN1_VOLT[3:0]	VGEN1 power-up voltage
8:4	OTP VGEN1 SEQ	VGEN1_SEQ[4:0]	VGEN1 power-up sequence
12:9	OTP VGEN2 VOLT	VGEN2_VOLT[3:0]	VGEN2 power-up voltage
17:13	OTP VGEN2 SEQ	VGEN2_SEQ[4:0]	VGEN2 power-up sequence
18	–	–	RSVD
19	OTP EN ECC1	EN_ECC_BANK 7	Enable ECC for OTP fuse bank 7
25:20	–	–	ECC check bits for fuse bank 7

Table 9. Bank 8

Fuses	Register Name	Register bits	Description
3:0	OTP VGEN3 VOLT	VGEN3_VOLT[3:0]	VGEN3 power-up voltage
8:4	OTP VGEN3 SEQ	VGEN3_SEQ[4:0]	VGEN3 power-up sequence
12:9	OTP VGEN4 VOLT	VGEN4_VOLT[3:0]	VGEN4 power-up voltage
17:13	OTP VGEN4 SEQ	VGEN4_SEQ[4:0]	VGEN4 power-up sequence
18	–	–	RSVD
19	OTP EN ECC1	EN_ECC_BANK 8	Enable ECC for OTP fuse bank 8
25:20	–	–	ECC check bits for fuse bank 8

Table 10. Bank 9

Fuses	Register Name	Register bits	Description
3:0	OTP VGEN5 VOLT	VGEN5_VOLT[3:0]	VGEN5 power-up voltage
8:4	OTP VGEN5 SEQ	VGEN5_SEQ[4:0]	VGEN5 power-up sequence
12:9	OTP VGEN6 VOLT	VGEN6_VOLT[3:0]	VGEN6 power-up voltage
17:13	OTP VGEN6 SEQ	VGEN6_SEQ[4:0]	VGEN6 power-up sequence
18	OTP PWRGD_EN	PWRGD_EN	Enable different functionality for RESETBMCU
19	OTP EN ECC1	EN_ECC_BANK9	Enable ECC for OTP fuse bank 9
25:20	–	–	ECC check bits for fuse bank 9

Table 11. Bank 10

Fuses	Register Name	Register bits	Description
1:0	OTP PU CONFIG1	SEQ_CLK_SPEED1[1:0]	Power-up sequence delay, bits are XORed
3:2	OTP PU CONFIG1	SWDVS_CLK1[1:0]	Power-up slew rate for all switching regulators, bits are XORed
4	OTP PU CONFIG1	PWRON_CFG1	Power button configuration, bits is XORed
5	OTP FUSE POR1	FUSE_POR1	Loads fuse values to TBBOTP registers, bit is XORed
7:6	OTP PU CONFIG2	SEQ_CLK_SPEED2[1:0]	Power-up sequence delay, bits are XORed
9:8	OTP PU CONFIG2	SWDVS_CLK2[1:0]	Power-up slew rate for all switching regulators, bits are XORed
10	OTP PU CONFIG2	PWRON_CFG2	Power button configuration, bits is XORed
11	OTP FUSE POR2	FUSE_POR2	Loads fuse values to TBBOTP registers, bit is XORed
13:12	OTP PU CONFIG3	SEQ_CLK_SPEED3[1:0]	Power-up sequence delay, bits are XORed
15:14	OTP PU CONFIG3	SWDVS_CLK3[1:0]	Power-up slew rate for all switching regulators, bits are XORed
16	OTP PU CONFIG3	PWRON_CFG3	Power button configuration, bits is XORed
17	OTP FUSE POR3	FUSE_POR3	Loads fuse values to TBBOTP registers, bit is XORed
18	OTP DONE	OTP_DONE	Prevents any further programming to fuses and further writes to TBBOTP registers
19	OTP EN ECC1	EN ECC BANK10	Enable ECC for OTP fuse bank 10
25:20	–	–	ECC check bits for fuse bank 10

The TBBOTP registers store data for programming the fuses. These registers are written to and read from using the I^{2}C interface.

Once the TBBOTP registers are loaded with the correct values, the fuses can then be programmed. Before discussing the programming process, some salient features of the OTP function are described.

2.4.1 TBBOTP Registers Description

The TBBOTP registers for configuring the switching regulators are listed in Table 12 and Table 13 to Table 18 provide a general description of the TBBOTP registers for all the switching regulators.

Table 12. OTP Switching Regulators Register Summary

Register	Address	Output
OTP SW1AB VOLT	0xA0	SW1AB OTP Output voltage set point
OTP SW1AB SEQ	0xA1	SW1AB OTP power-up sequence selection
OTP SW1AB CONFIG	0xA2	SW1AB OTP operation mode and frequency selection
OTP SW1C VOLT	0xA8	SW1C OTP Output voltage set point
OTP SW1C SEQ	0xA9	SW1C OTP power-up sequence selection
OTP SW1C CONFIG	0xAA	SW1C OTP frequency selection
OTP SW2 VOLT	0xAC	SW2 OTP Output voltage set point
OTP SW2 SEQ	0xAD	SW2 OTP power-up sequence selection
OTP SW2 CONFIG	0xAE	SW2 OTP frequency selection
OTP SW3AVOLT	0xB0	SW3A OTP Output voltage set point
OTP SW3A SEQ	0xB1	SW3A OTP power-up sequence selection
OTP SW3A CONFIG	0xB2	SW3A OTP operation mode and frequency selection
OTP SW3B VOLT	0xB4	SW3B OTP Output voltage set point
OTP SW3B SEQ	0xB5	SW3B OTP power-up sequence selection
OTP SW3B CONFIG	0xB6	SW3B OTP frequency selection
OTP SW4 VOLT	0xB8	SW4 OTP output voltage set point
OTP SW4 SEQ	0xB9	SW4 OTP power-up sequence selection
OTP SW4 CONFIG	0xBA	SW4 OTP frequency and VTT mode selection
OTP SWBST VOLT	0xBC	SWBST OTP output voltage set point
OTP SWBST SEQ	0xBD	SWBST OTP power-up sequence selection

Table 13. OTP SW1x VOLT Register Description (SW1A/B and SW1C)

Name	Bit #	Description
SW1x_VOLT	5:0	Sets the SW1x output voltage to be programmed on the OTP fuses and loaded during power-up. Refer to SW1A/B/C output voltage configuration table on Data Sheet for all possible configurations.
UNUSED	7:6	UNUSED

Table 14. OTP SWx VOLT Register Description (SW2 - SW4)

Name	Bit #	Description
SWx_VOLT	6:0	Sets the SWx output voltage to be programmed on the OTP fuses and loaded during power-up. Refer to the respective SWx output voltage configuration table on datasheet for all possible configurations.
UNUSED	7	UNUSED

Table 15. OTP SWx SEQ Register Description

Name	Bit #	Description
SWx_SEQ	4:0	Assign the power-up sequence slot 0-31 for SWx
UNUSED	7:5	UNUSED

Table 16. OTP SWx CONFIG Register Description

Name	Bit #	Description	
SWx_FREQ	1:0	SWx OTP Frequency configuration 00 = 1.0 MHz 01 = 2.0 MHz 10 = 4.0 MHz 11 = Reserved	
SWx_CONFIG	3:2	SWx configuration[3]	
		SW1A/B 00 = A/B/C Single Phase 01 = A/B Single Phase - C Independent 10 = A/B Dual Phase - C Independent 11 = Reserved	SW3A 00 = A/B Single Phase 01 = A/B Single Phase 10 = A/B Dual Phase 11 = A/B Independent mode
VTT	4	Enable SW4 in VTT mode [4]	
UNUSED	7:5	UNUSED	

Notes
3. Only on OTP SW1AB CONFIG and OTP SW3A CONFIG registers. UNUSED on all other OTP SWx CONFIG registers.
4. Only on OTP SW4 CONFIG register. UNUSED on all other OTP SWx CONFIG registers.

Table 17. OTP SWBST VOLT Register Description

Name	Bit #	Description
SWBST_VOLT	1:0	SWBST OTP output voltage setpoint 00 = 5.00 V 01 = 5.05 V 10 = 5.10 V 11 = 5.15 V
UNUSED	7:2	UNUSED

Table 18. OTP SWBST SEQ Register Description

Name	Bit #	Description
SWBST_SEQ	4:0	Assign the power-up sequence slot 0-31 for SWBST
UNUSED	7:5	UNUSED

Table 19 shows a summary of all the registers related to the linear regulators, and Table 20 to Table 22 provide a general bit description of the linear regulator OTP registers.

Table 19. OTP Linear Regulators Register Summary

Register	Address	Output
OTP VSNVS VOLT	0xC0	VSNVS OTP Output voltage set point
OTP VREFDDR SEQ	0xC4	VREFDDR OTP power-up sequence selection
OTP VGEN1 VOLT	0xC8	VGEN1 OTP Output voltage set point
OTP VGEN1 SEQ	0xC9	VGEN1 OTP power-up sequence selection
OTP VGEN2 VOLT	0xCC	VGEN2 OTP Output voltage set point
OTP VGEN2 SEQ	0xCD	VGEN2 OTP power-up sequence selection
OTP VGEN3 VOLT	0xD0	VGEN3 OTP Output voltage set point
OTP VGEN3 SEQ	0xD1	VGEN3 OTP power-up sequence selection
OTP VGEN4 VOLT	0xD4	VGEN4 OTP Output voltage set point
OTP VGEN4 SEQ	0xD5	VGEN4 OTP power-up sequence selection
OTP VGEN5 VOLT	0xD8	VGEN5 OTP output voltage set point
OTP VGEN5 SEQ	0xD9	VGEN5 OTP power-up sequence selection
OTP VGEN6 VOLT	0xDC	VGEN6 OTP output voltage set point
OTP VGEN6 SEQ	0xDD	VGEN6 OTP power-up sequence selection

Table 20. OTP VSNVS VOLT Register Description

Name	Bit #	Description
VSNVS_VOLT	2:0	Sets the VSNVS output voltage to be programmed on the OTP fuses and loaded during power-up 000 = 1.0 V 001 = 1.1 V 010 = 1.2 V 011 = 1.3 V 100 = 1.5 V 101 = 1.8 V 110 = 3.0 V 111 = RSVD
UNUSED	7:3	UNUSED

Table 21. OTP VGENx VOLT Register Description

Name	Bit #	Description
VGENx_VOLT	3:0	Sets the VGENx output voltage to be programmed on the OTP fuses and loaded during power-up. Refer to the VGENx output voltage configuration table on Data Sheet for all possible configurations.
UNUSED	7:4	UNUSED

Table 22. OTP xxxx SEQ Register Description

Name	Bit #	Description
xxxx_SEQ	4:0	Assign the power-up sequence slot 0-31 for the specific linear regulator or VREFDDR voltage
UNUSED	7:5	UNUSED

2.4.2 OTP Redundant Bits

Some functions are assigned redundant bits, which are XORed, to allow that function to be changed multiple times.

These are the functions that have redundant OTP fuse bits:

- Sequence Clock Frequency (Sequence delay)
- DVS Clock Frequency (Initial slew rate)
- Power button function
- Fuse POR

Table 23 shows the I^{2}C registers in the Extended Page 1 dedicated to redundant bits for the four functions mentioned. The XORed bits are read-only.

Table 23. OTP Redundant Bits Registers

Extended Pg 1		I^{2}C Data Bits							
Addr	**Reg Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
E0	OTP PU CONFIG1	–	–	–	PWRON CFG1	SWDVS_CLK1[1:0]		SEQ_CLK SPEED1[1:0]	
		0	0	0	x	x	x	x	x
E1	OTP PU CONFIG2	–	–	–	PWRON CFg2	SWDVS_CLK2[1:0]		SEQ_CLK SPEED2[1:0]	
		0	0	0	x	x	x	x	x
E2	OTP PU CONFIG3	–	–	–	PWRON CFG3	SWDVS_CLK3[1:0]		SEQ_CLK SPEED3[1:0]	
		0	0	0	x	x	x	x	x
E3	OTP PU CONFIG XOR	–	–	–	PWRON CFG_XOR	SWDVS_CLK3_XOR		SEQ_CLK_SPEED XOR	
		0	0	0	x	x	x	x	x
E4	OTP FUSE POR1	TBB_POR	SOFT_FUSE POR	–	–	–	–	FUSE POR1	–
		0	0	0	0	0	0	x	0
E5	OTP FUSE POR1	RSVD	RSVD	–	–	–	–	FUSE POR2	–
		0	0	0	0	0	0	x	0
E6	OTP FUSE POR1	RSVD	RSVD	–	–	–	–	FUSE POR3	–
		0	0	0	0	0	0	x	0
E7	OTP FUSE POR XOR	RSVD	RSVD	–	–	–	–	FUSE POR_XOR	–
		0	0	0	0	0	0	x	0

Table 24. OTP PU CONFIGx Bits Definition

Bit	Name	Description
1:0	SEQ_CLK_SPEEDx[1:0]	Sequence delay between steps, bits are XORed 00 = 500 µs 01 = 1000 µs 10 = 2000 µs 11 = 4000 µs
3: 2	SWDVS_CLKx[1:0]	Start-up slew rate, bits are XOR'd 00 = 25 mV/2 µs 01 = 25 mV/4 µs 10 = 25 mV/8 µs 11 = 25 mV/16 µs
4	PWRON_CFGx	Set the power on button initial configuration 0 = Power button is level sensitive 1 = Power button is edge sensitive and turn-off is based on time held low
7:5	RSVD	Reserved

Table 25. OTP_PU_CONFIG XOR Bits Definition

Bit	Name	Description
1:0	SEQ_CLK_SPEED_XOR	Final result of the XOR function of the SEQ_CLK_SPEEDx[1:0] bits
3: 2	SWDVS_CLK_XOR	Final result of the XOR function of the SWDVS_CLKx[1:0] bits
4	PWRON_CFG_XOR	Final result of the XOR function of the SEQ_PWRON_CFGx bits
7:5	RSVD	Reserved

Table 26. OTP_FUSE_PORx Bits Definition

Bit	Name	Description
0	RSVD	Reserved
1	FUSE_PORx[5]	Load fuse values to TBB_OTP registers 0 = No Fuse value loaded 1 = Programmed fuse values loaded to TBB_OTP registers
5:2	RSVD	Reserved
6	SOFT_FUSE_POR[6]	Software version of the FUSE_PORx bit
7:5	TBB_POR[6]	Prototyping enable bit 0 = Prototyping disabled 1 = Prototyping enabled

5. In MMPF0100 FUSE_POR1, FUSE_POR2 and FUSE_POR3 are XOR'ed into the FUSE_POR_XOR bit. The FUSE_POR_XOR has to be 1 for fuses to be loaded. This can be achieved by setting any one or all of the FUSE_PORx bits. In MMPF0100A, the XOR function is removed. It is required to set all of the FUSE_PORx bits to be able to load the fuses.

6. Reserved on Addresses E5 and E6

Table 27. OTP FUSE POR XOR Bits Definition

Bit	Name	Description
0	RSVD	Reserved
1	FUSE_POR_XOR	Final result of the XOR function of the FUSE_PORx bits
7:2	RSVD	Reserved

For example, if FUSE_POR1 is programmed to “1”, then fuse values are loaded as FUSE_POR_XOR is “1”. If FUSE_POR2 is then programmed, FUSE_POR_XOR becomes “0” and fuses values cannot be loaded. SOFT_FUSE_POR is a software version of the FUSE_PORx bits, i.e. it is XORed with the FUSE_PORx bits to determine whether fuses can be loaded.

See Table 28 for the XOR truth table for the previous functions listed.

Table 28. Redundant Bit XOR Function Truth Table

Bit1	Bit2	Bit3	XOR bit
0	0	0	0
1	0	0	1
1	1	0	0
1	1	1	1

Note: The desired function of the redundant bits must be determined when ECC is configured and its bits programmed, or the ECC logic attempts to correct the newly programmed redundant bits. ECC is discussed in Error Correction Code (ECC).

2.4.3 OTP Register Reloading without Turn-on Event

After the fuses are programmed, their values may be loaded into the digital control logic without toggling VIN or PWRON. To update the TBBOTP registers by reloading the fuse values automatically, set bits in the OTP LOAD MASK register depending on the functionality required. Refer to Table 30 for a description of the OTP LOAD MASK register.

Table 29. OTP Load Mask Register

Extended Page 1		I^2C Data Bits							
Addr	**Reg Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
84	OTP LOAD MASK	START	RL PWRTN	FORCE PWRCTL	RL PWRCTL	RL OTP	RL OTP ECC	RL OTP FUSE	RSVD
		0	0	0	0	0	0	0	0

Table 30. OTP Reload Mask Register Bit Description

Bit	Name	Description
0	RSVD	Reserved
1	RL_OTP_FUSE	Reload the OTP fuse latch from the analog fuse bit 0 = Disable loading 1 = Enable loading
2	RL_OTP_ECC	Reload the OTP ECC registers. Set this bit irrespective of whether ECC is enabled or disabled. 0 = Disable loading 1 = Enable loading
3	RL_OTP	Reload the TBBOTP registers from the fuses 0 = Disable loading 1 = Enable loading of fuses if ECC is disabled. Enable loading of ECC corrected fuses if ECC is enabled.
4	RL_PWRCTL	Reload the power control registers from the TBBOTP registers 0 = Disable loading 1 = Enable loading
5	FORCE_PWRCTL	Forces the power control registers to be reloaded if they are being used to control the regulators 0 = No reload forced 1 = Power control register value affects regulators when the reload sequence is enabled and RL PWRCTL bit is enabled.This is needed when changing output voltage of switching regulators from low-voltage range to high-voltage range
6	RL_PWRTN	Reloads the register that controls how the PWRON button works 0 = PWRON configuration setting does not change until a shutdown and restart event 1 = PWRON behavior switch to new OTP PWRON button configuration when START bit is enabled
7	START	Reload sequence start bit 0 = Reload sequence disabled 1 = Starts the reload sequence, when the sequence is done all of the OTP_LOAD_MASK bits are reset

Often only bits 1, 2, and 3 need to be set, as well as the START bit, to reload the TBBOTP registers after the fuses are programmed. The TBBOTP register values could then be checked to make sure the correct values have been loaded from the fuses. Setting bits 4 and 5, updates the regulator parameters immediately. This should be done with caution if PWRON is already asserted. A PWRON event triggers a complete reload using the same logic. When a '1' is written to Bit 7 of the OTP_LOAD_MASK registers, the MMPF0100 is turned off momentarily and then turned back on to reload the fuses. If it is desired to reload the fuses without first turning off the MMPF0100, clear Bit 0 of the PWRCTRL_OTP_CTRL register prior to writing to the OTP_LOAD_MASK register. Note that the OTP_LOAD_MASK is register 0x84 in Extended Page 1 whereas the PWRCTRL_OTP_CTRL is register 0x88 in Extended Page 2.

2.4.4 Direct OTP Fuse Read

The OTP_FUSE_READ_EN bit allows the reading of the uncorrected fuse values when it is set HIGH. If ECC is not enabled, or there is no programming error, the values loaded into the TBBOTP registers are identical to the fuse values. If ECC is enabled and a single-bit error occurs during programming, the fuse values may be different from the values loaded into the TBBOTP registers. The values loaded into the TBBOTP registers are the error-corrected values. Table 31 shows the OTP FUSE READ EN register.

Table 31. OTP Fuse Read Enable Register

FSL Extended Page 1		I^2C Data Bits							
Addr	**Register Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
80	OTP FUSE READ EN	–	–	–	–	–	–	–	OTP_FUSE_READ_EN
		0	0	0	0	0	0	0	0

2.5 Fuse Programming and Error Correction Code (ECC)

2.5.1 OTP Fuse Control Register

An example script for OTP programing is shown in section OTP Programming Example. The OTP_FUSE_CTLx registers, located in the Extended Page 2, must be written to in order to program fuses. There are ten such registers, one for each bank, Refer to Table 32 and Table 33 for a description of the registers.

Table 32. General OTP Fuse Control Register Bits

FSL Extended Page 2	I^2C Data Bits							
Reg Name	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
OTP_FUSE_CTLx	–	–	–	–	ANTIFUSEx_EN	ANTIFUSEx_ LOAD	ANTIFUSEx_ RW	BYPASSx

Table 33. OTP Fuse Control Bits Description

Bit	Name	Description
0	BYPASSx	Multiplexor that selects between the value stored in the digital fuse latch and the value on the TBBOTP register 0 = Select from digital latch 1 = Select from TBBOTP register
1	ANTIFUSEx_RW	Allows programming the fuse bank when VDDOTP is 8.25 V 0 = Disable program fuse 1 = Enable program fuse
2	ANTIFUSEx_LOAD	Clock input to the digital latch that stores the state of the analog fuse cell, it is active high and is pulsed while the ANTIFUSE_EN bit is high to load the value of the analog fuse state into the digital latch.
3	ANTIFUSEx_EN	Turns on the bias to the analog fuse cell so that it can be written to or read from 0 = Analog bias disabled 1 = Analog bias enabled
4-7	Not used	Not used

2.5.2 Error Correction Code (ECC)

Error correction is off by default, but it is recommended for all OTP programming operations. When enabled, it reports and corrects a single bit error per fuse bank, but only reports a double bit error per fuse bank. Fuses may be programmed without using ECC. However, after verifying that the part is configured properly, ECC may enabled, and the error check bits can be programmed.

It should be noted that double bit errors can prevent regulators from powering up, or can result in a configuration that does not match the external components. Although such occurrence is rare, it is still a good practice to employ ECC to at least alert the user of such an occurrence.

Note: The desired function of the redundant bits must be determined when ECC is configured and its bits programmed, or the ECC logic attempts to correct the newly programmed redundant bits.

Sections **2.5.2.1** through **2.5.2.3** are for advanced users. For a simple script that enables ECC, proceed to section OTP Programming Example.

2.5.2.1 ECC Interrupt

With ECC enabled, if a single fuse in a bank has the wrong value, the ECC logic corrects that bit and the corrected value is loaded into the TBBOTP register for that bank. The single error bit for that bank is set and also the main interrupt ECC bit is set. If two or more bits are in error, in a bank, the ECC is not able to correct them. The double error bit error for that bank is set and the ECC interrupt bit is set. The single error and double error bits may be read from registers 0x8A to 0x8D in the Extended Page1 of the register map. The ECC interrupt bit may be read from register, 0xE, on the functional page of the register map.

Table 34. ECC Error Detection Registers

Extended Page 1		I^{2}C Data Bits							
Addr	**Reg Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
8A	OTP ECC SE1	–	–	–	ECC5_SE	ECC4_SE	ECC3_SE	ECC2_SE	ECC1_SE
		x	x	x	0	0	0	0	0
8B	OTP ECC SE2	–	–	–	ECC10_SE	ECC9_SE	ECC8_SE	ECC7_SE	ECC6_SE
		x	x	x	0	0	0	0	0
8C	OTP ECC DE1	–	–	–	ECC5_DE	ECC4_DE	ECC3_DE	ECC2_DE	ECC1_DE
		x	x	x	0	0	0	0	0
8D	OTP ECC DE2	–	–	–	ECC10_DE	ECC9_DE	ECC8_DE	ECC7_DE	ECC6_DE
		x	x	x	0	0	0	0	0

Table 35. OTP ECC SE1 and 2 Register Description

Bit	Name	Default	Description
OTP ECC SE1			
0	ECC1_SE	0	Single error detection in fuse bank 1 0 = No single error detected 1 = Single error detected
1	ECC2_SE	0	Single error detection in fuse bank 2 0 = No single error detected 1 = Single error detected
2	ECC3_SE	0	Single error detection in fuse bank 3 0 = No single error detected 1 = Single error detected
3	ECC4_SE	0	Single error detection in fuse bank 4 0 = No single error detected 1 = Single error detected
4	ECC5_SE	0	Single error detection in fuse bank 5 0 = No single error detected 1 = Single error detected
7:5	RSVD	0	Reserved
OTP ECC SE2			
0	ECC6_SE	0	Single error detection in fuse bank 6 0 = No single error detected 1 = Single error detected
1	ECC7_SE	0	Single error detection in fuse bank 7 0 = No single error detected 1 = Single error detected
2	ECC8_SE	0	Single error detection in fuse bank 8 0 = No single error detected 1 = Single error detected
3	ECC9_SE	0	Single error detection in fuse bank 9 0 = No single error detected 1 = Single error detected
4	ECC10_SE	0	Single error detection in fuse bank 10 0 = No single error detected 1 = Single error detected
7:5	RSVD	0	Reserved

Table 36. OTP ECC DE1 and 2 Register Description

Bit	Name	Default	Description
OTP ECC DE1			
0	ECC1_DE	0	Dual error detection in fuse bank 1 0 = No single error detected 1 = Single error detected
1	ECC2_DE	0	Dual error detection in fuse bank 2 0 = No single error detected 1 = Single error detected
2	ECC3_DE	0	Dual error detection in fuse bank 3 0 = No single error detected 1 = Single error detected
3	ECC4_DE	0	Dual error detection in fuse bank 4 0 = No single error detected 1 = Single error detected
4	ECC5_DE	0	Dual error detection in fuse bank 5 0 = No single error detected 1 = Single error detected
7:5	RSVD	0	Reserved
OTP ECC DE2			
0	ECC6_DE	0	Dual error detection in fuse bank 6 0 = No single error detected 1 = Single error detected
1	ECC7_DE	0	Dual error detection in fuse bank 7 0 = No single error detected 1 = Single error detected
2	ECC8_DE	0	Dual error detection in fuse bank 8 0 = No single error detected 1 = Single error detected
3	ECC9_DE	0	Dual error detection in fuse bank 9 0 = No single error detected 1 = Single error detected
4	ECC10_DE	0	Dual error detection in fuse bank 10 0 = No single error detected 1 = Single error detected
7:5	RSVD	0	Reserved

All interrupts are masked by default, therefore the ECC interrupt should be unmasked after fuses are programmed, with ECC enabled, to determine if single or double bit errors exist in any of the banks. The location of the error bits may be read from the registers described in Table 34.

2.5.2.2 Analyzing a Single Bit ECC Error

Although not necessary, when a single bit error occurs, the ECC check bits may be read to find out what fuse in a given bank is in error. The check bits for each bank may be read from bits[5:0], in registers 0xE1 to 0xEA, in the Extended Page 2. See Table 37. For example, if there is an error in bit[5] of fuse bank 3, reading bits[5:0] of register 0xE3 yields a hexadecimal code of 0x15. Refer to Table 40 and Table 41 which describe the error control registers.

Table 37. ECC Error Location Coding

Bit in Error	ECC check bit code
0	07
1	0B
2	0D
3	0E
4	13
5	15
6	16
7	19
8	1A
9	1C
10	23
11	25
12	26
13	29
14	2A
15	2C
16	31
17	32
18	34
19	38
20	01
21	02
22	04
23	08
24	10
25	20

2.5.2.3 Fuse Programming with ECC

To program fuses with ECC, bits in the following registers must be enabled:

- OTP EN ECC0 and OTP EN ECC1 in the Extended Page 1
- OTP AUTO ECC0 and OTP AUTO ECC1 in the Extended Page 2.

The ECC enable registers are shown in Table 38. Enable error correction for any bank by setting the appropriate bit. Bits in the OTP EN ECCx registers are programmed and not just set in software.

The OTP AUTO ECC registers are shown in Table 39. After the fuses are programmed, their values may be loaded to the TBBOTP registers. The values loaded are the error-corrected values if there was a single bit error in any bank. To view the uncorrected or raw fuse values see the Direct OTP Fuse Read section. To determine if there was an error when programming fuses, the following options are available:

- Checking the fuse values against what was written.
- Monitor the INTB signal, but first the ECC interrupt must be unmasked.
- Read bits[5:0] from the OTP ECC CTRLx registers in the Extended Page 2. See Table 37 to decipher single bit error codes and Table 41 for a description of the ECC registers.

Table 38. ECC Enable Registers

Extended Pg 1		**I²C Data Bits**							
Addr	**Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
F0	OTP EN ECC0	–	–	–	EN_ECC _BANK5	EN_ECC _BANK4	EN_ECC _BANK3	EN_ECC _BANK2	EN_ECC _BANK1
		–	–	–	0	0	0	0	0
F1	OTP EN ECC1	–	–	–	EN_ECC _BANK10	EN_ECC _BANK9	EN_ECC _BANK8	EN_ECC _BANK7	EN_ECC _BANK6
		–	–	–	0	0	0	0	0

Table 39. Automatic ECC Mode Enable Registers

Extended Pg 2		**I²C Data Bits**							
Addr	**Name**	**7**	**6**	**5**	**4**	**3**	**2**	**1**	**0**
D0	OTP AUTO ECC0	–	–	–	AUTO_ECC _BANK5	AUTO_ECC _BANK4	AUTO_ECC _BANK3	AUTO_ECC _BANK2	AUTO_ECC _BANK1
		–	–	–	0	0	0	0	0
D1	OTP AUTO ECC1	–	–	–	AUTO_ECC _BANK10	AUTO_ECC _BANK9	AUTO_ECC _BANK8	AUTO_ECC _BANK7	AUTO_ECC _BANK6
		–	–	–	0	0	0	0	0

Table 40. ECC Control Registers in the Extended Page 2

Extended Page 2		I²C Data bits							
Addr	Name	D7	D6	D5	D4	D3	D2	D1	D0
E1	ECC_CTRL1	ECC1_EN_TBB	ECC1_CALC_CIN	ECC1_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E2	ECC_CTRL2	ECC2_EN_TBB	ECC2_CALC_CIN	ECC2_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E3	ECC_CTRL3	ECC3_EN_TBB	ECC3_CALC_CIN	ECC3_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E4	ECC_CTRL4	ECC4_EN_TBB	ECC4_CALC_CIN	ECC4_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E5	ECC_CTRL5	ECC5_EN_TBB	ECC5_CALC_CIN	ECC5_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E6	ECC_CTRL6	ECC6_EN_TBB	ECC6_CALC_CIN	ECC6_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E7	ECC_CTRL7	ECC7_EN_TBB	ECC7_CALC_CIN	ECC7_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E8	ECC_CTRL8	ECC8_EN_TBB	ECC8_CALC_CIN	ECC8_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
E9	ECC_CTRL9	ECC9_EN_TBB	ECC9_CALC_CIN	ECC9_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0
EA	ECC_CTRL10	ECC10_EN_TBB	ECC10_CALC_CIN	ECC10_CIN_TBB[5:0]					
		0	0	0	0	0	0	0	0

Table 41. ECC_CTRLx Registers Description

Bit	Name	Default	Description
5:0	ECCx_CIN_TBB	0	ECC error location code See codes on Table 37
6	ECCx_CALC_CIN	0	Calculate the ECC check bit values 0 = Calculation disabled 1 = Calculation enabled
7	ECCx_EN_TBB	0	This bit is XORed with the EN_ECC_BANKx bit • If EN_ECC_BANKx bit is programmed, a 1 disables the ECC function for the respective bank • If EN_ECC_BANKx bit is not programmed, the ECCx_EN_TBB acts as a soft EN_ECCx bit

3 Hardware Considerations

The minimum system requirements to allow programming of the OTP fuses are:

1. An I^{2}C communication bridge to communicate with the PF0100
2. A 1.7 V to 3.6 V power supply at VDDIO (power to I^{2}C block and pull-up resistors for the SCL and SDA lines). Using the KITPFPGMEVME requires a minimum of 3.0 V at VDDIO
3. An 9.5 V/9.25 V, 100 mA power supply at VDDOTP bypassed by 2 x 10 μF capacitors. The voltage depends on the silicon revision used. See section OTP Programming Example for details.
4. An input voltage of 3.3 V at the VIN pin

Figure 1 shows the minimum requirements for programming the MMPF0100. For programming the MMPF0100 on an application board some hardware considerations have to be made.

.

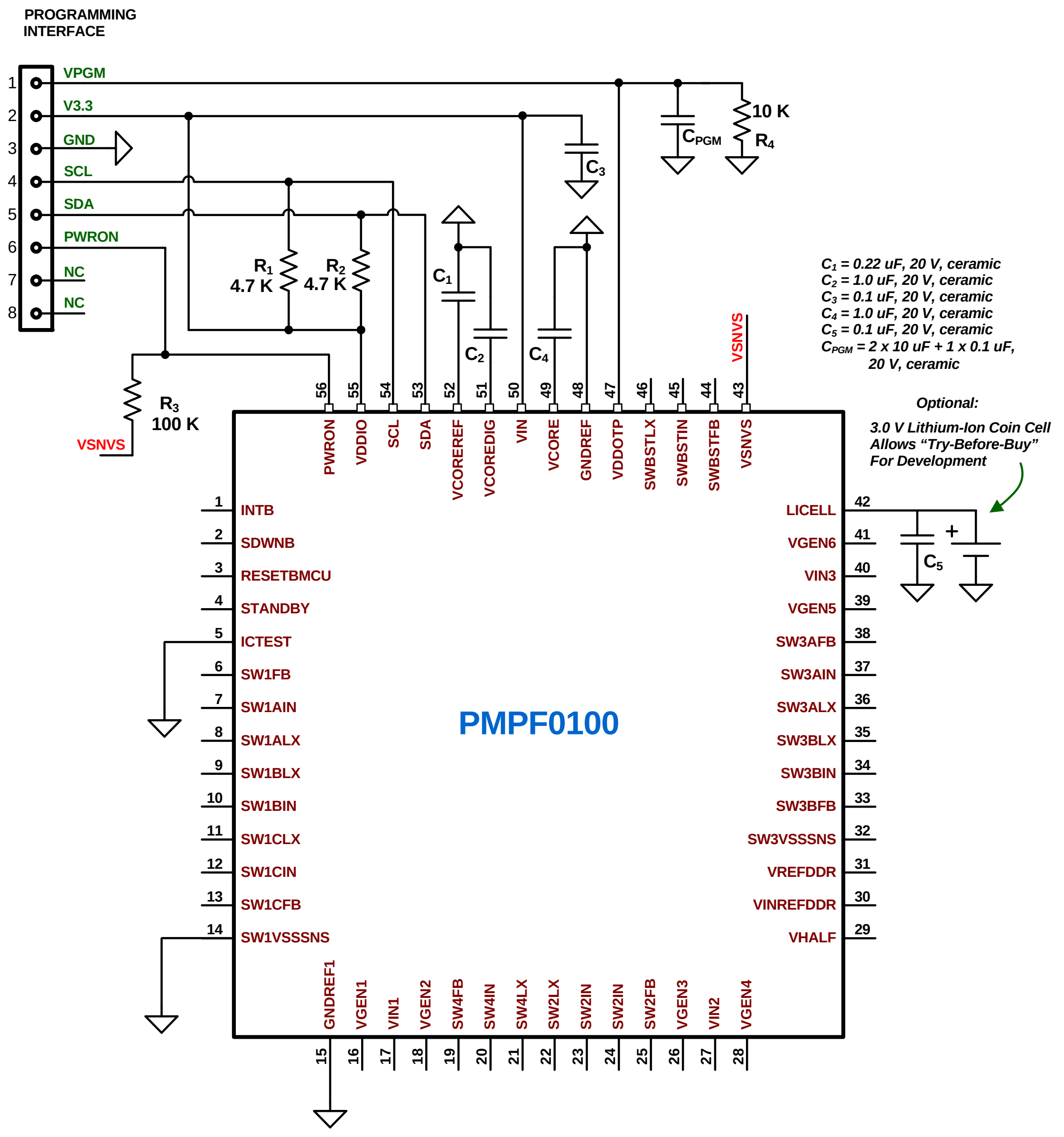

Figure 1. Minimum OTP Programming Requirements Diagram

3.1 Programming the MMPF0100 on an Application Board

When programming the MMPF0100 in an application board, voltages must be applied at the VIN, VDDIO and VDDOTP pins. Considerations must be made to allow voltages to be applied on these rails in a fully populated system board.

3.2 Isolating SCL/SDA

During OTP programming of the MMPF0100, commands are sent to the MMPF0100 via the SCL/SDA pins using an I^{2}C communications bridge, typically a programming dongle such as the KITPFPGMEVME. In a typical application, the SCL and SDA pins of the MMPF0100 are connected to communication ports of an I^{2}C master, typically the processor. Depending on how the ports in the processor are designed, it may or may not be valid to communicate with the MMPF0100 using an external dongle while the SCL/SDA pins are still connected to the processor especially when the processor is unpowered due to a yet-to-be-programmed MMPF0100.

It is recommended to isolate the SCL/SDA lines going to the processor while communicating with the MMPF0100 using an external dongle as shown in the example in Figure 2. In the normally closed position of the analog switch (NLAS3158 or similar), SCL and SDA of the MMPF0100 are connected to the processor. When the signal Programmer_Select_O/P is high, SCL and SDA of the MMPF0100 are connected to the external programming interface. The Programmer_Select_O/P signal can be generated by the programming interface as well.

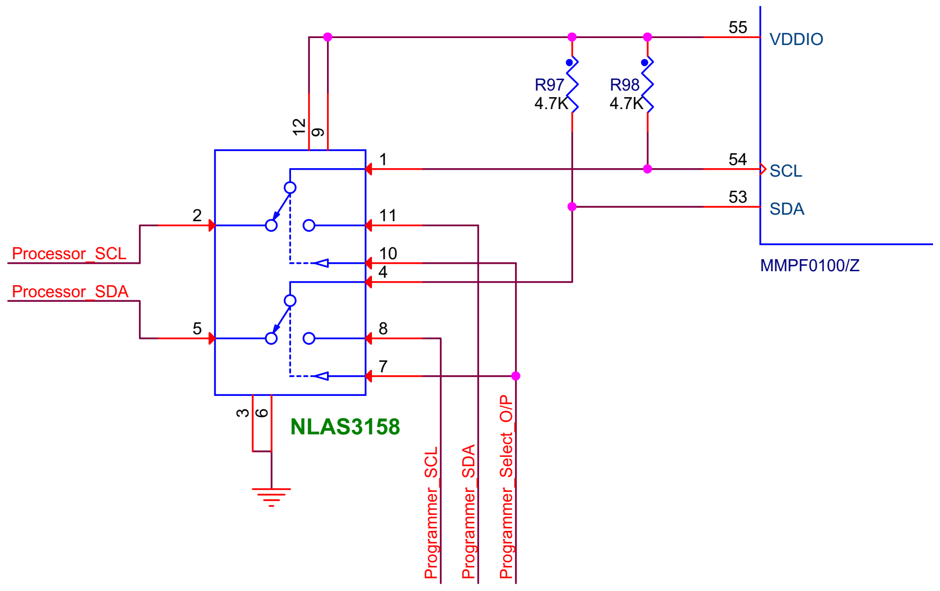

Figure 2. Isolating SCL and SDA Using an Analog Switch

Note: Using the analog switch may not be the most cost effective option to isolate the I^{2}C bus. Similar functionality can be achieved by using solder shorts or 0 Ohm resistors. However, minor rework of the board would be required once OTP programming is completed.

3.3 Programming using the PF-Programmer

The KITPFPGMEVME is Freescale's programming board that can be used to OTP program the MMPF0100. It integrates a 3.3 V LDO to power the MMPF0100 and a boost converter with an adjustable output voltage to generate the OTP programming voltage. An integrated USB-to-I^{2}C converter allows PC communication with the MMPF0100 using a Freescale supplied GUI. See Figure 3 for a block diagram of KITPFPGMEVME.

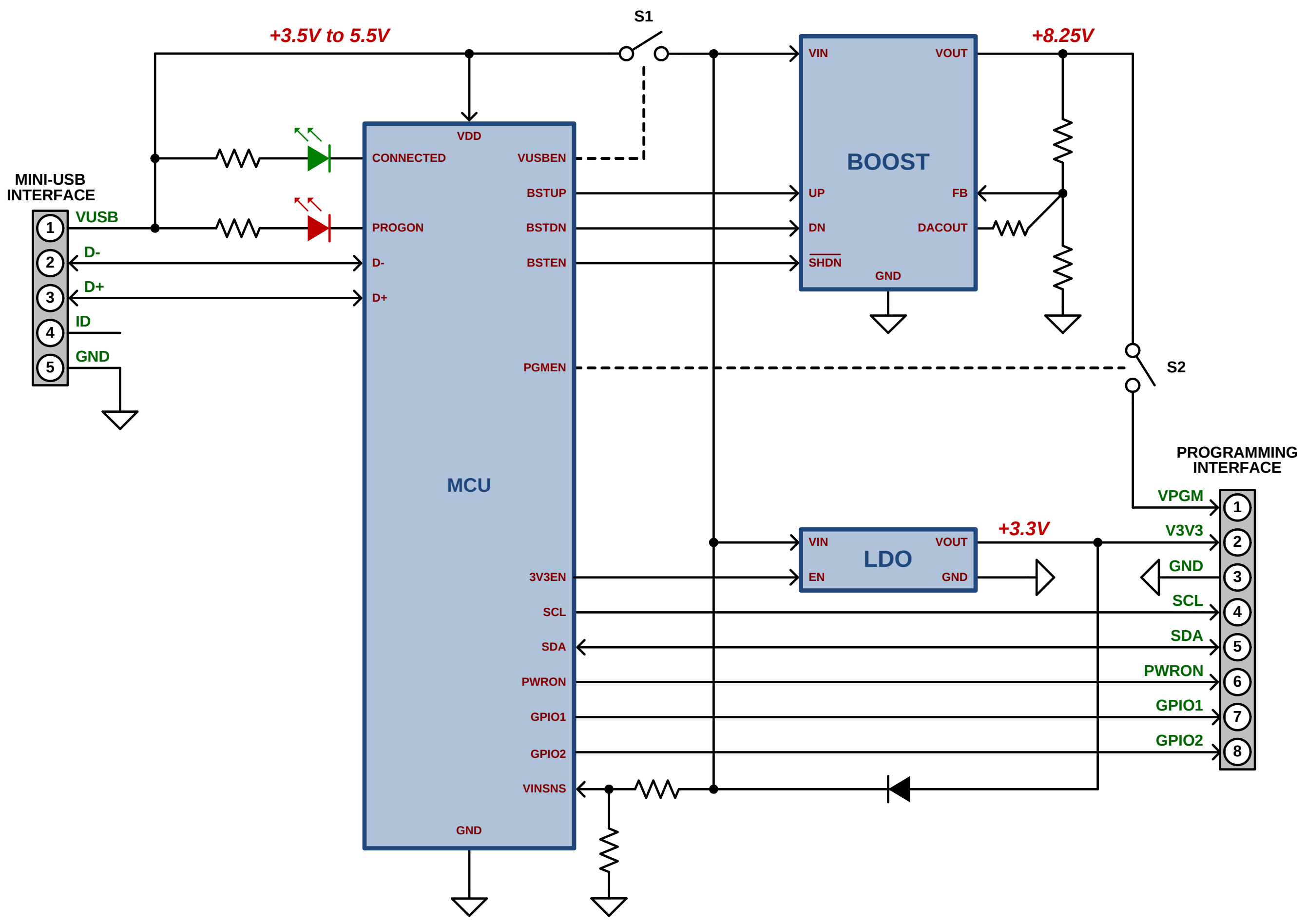

Figure 3. KITPFPGMEVME Block Diagram

The V3V3 rail generates a 3.3 V supply to power the VIN and VDDIO rails of the MMPF0100. Figure 4 shows how to interface the KITPFPGMEVME with the MMPF0100 in an application board. For applications which use a single rail for VIN and VDDIO, the connection is straightforward as shown in Figure 4. However, it must be ensured that other loads connected to the 3.3 V rail do not surpass the current rating of the LDO. If that is the case, isolation in the form of an analog switch, a solder short, or a 0 Ohm resistor is required.

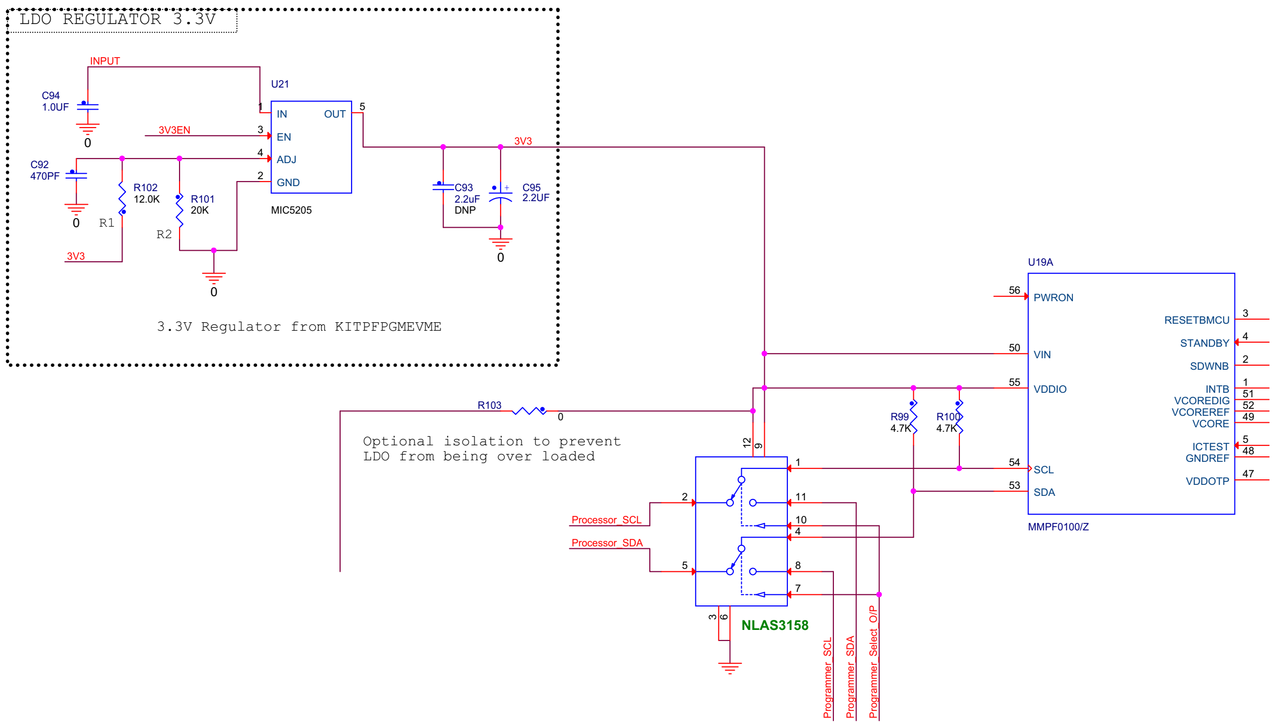

Figure 4. Interfacing KITPFPGMEVME for application board MMPF0100 programming (Systems with VIN = VDDIO)

In systems which use different rails for VIN and VDDIO, the requirements for interfacing the KITPFPGMEVME with the MMPF0100 in an application board are different. As the KITPGPGMEVME provides a single rail for VIN and VDDIO, it is necessary to short the two rails on the application board during programming. This requires that the other loads connected on the VDDIO rail in the system be isolated. These can be achieved using an analog switch as shown in Figure 5. When the signal Programmer_Select_O/P is low, the system VDDIO supply is connected to the VDDIO pin. When Programmer_Select_O/P is high, VIN and VDDIO are connected together allowing the KITPFPGMEVME to communicate with MMPF0100.

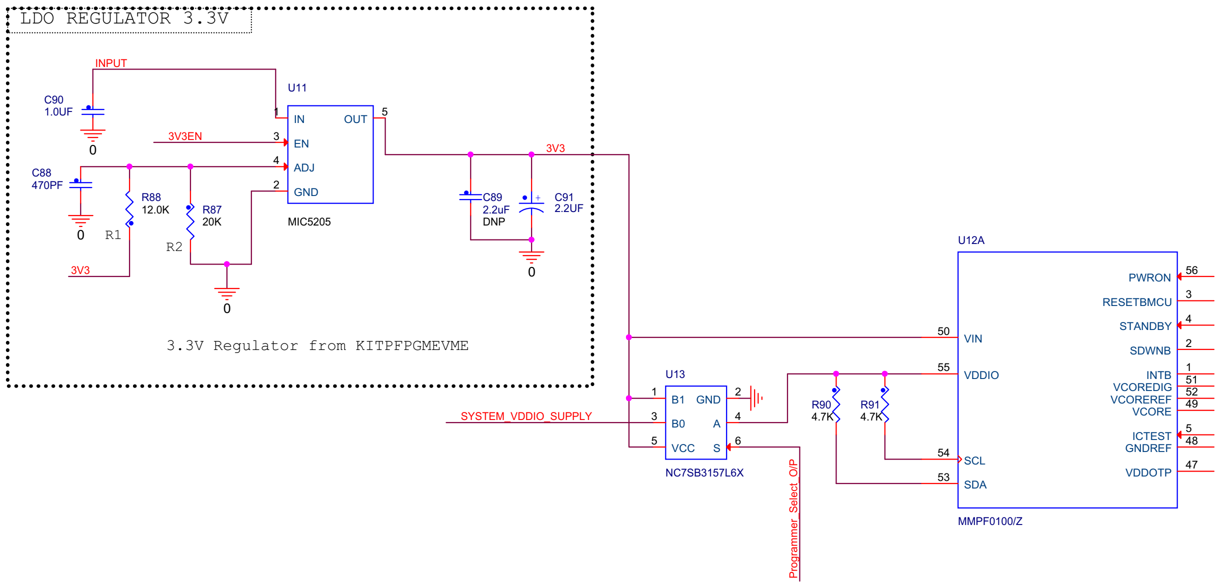

Figure 5. . Interfacing KITPFPGMEVME for application board MMPF0100 programming (Systems with different VIN and VDDIO)

Note: Using the analog switch may not be the most cost effective option to supply VIN and VDDIO. Similar functionality can be achieved by using solder shorts or 0 Ohm resistors. However, minor rework of the board would be required once OTP programming is completed.

The Programmer_Select_O/P signal can be generated using the GPIO2 pin on the KITPFPGMEVME. Controlling this signal can be part of the programming script.

3.4 Programming using a Generic Programmer

Following are the requirements if it is preferred to use a generic programmer board:

1. VIN power supply: 3.3 V, 100 mA
2. VDDIO power supply: 1.8 V to 3.3 V, 10 mA
3. I^{2}C Master
4. GPO signal to control MMPF0100's PWRON pin
5. GPO signal to control analog switch (Programmer_Select_O/P)
6. 9.5 V/9.25 V 100 mA power supply at VDDOTP bypassed by 2 x 10 μF capacitors. The voltage depends on the silicon revision used. See section OTP Programming Example for details.

An example is shown in Figure 6.

Note: Using the analog switch may not be the most cost effective option to isolate the I^{2}C bus. Similar functionality can be achieved by using solder shorts or 0 Ohm resistors. However, minor rework of the board would be required once OTP programming is completed.

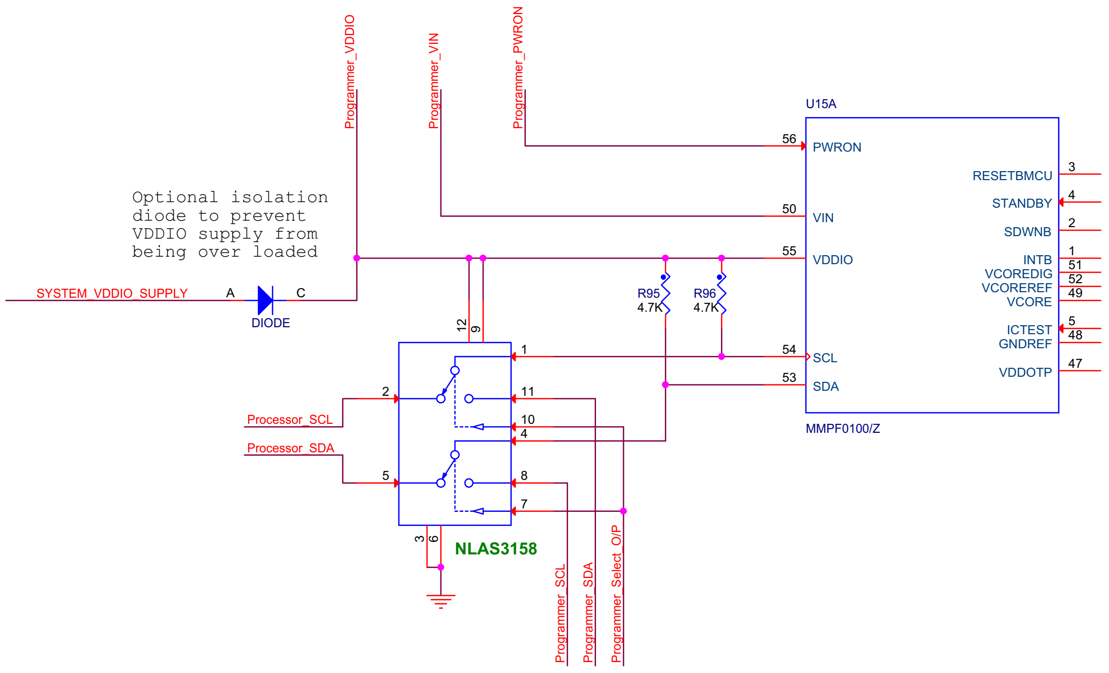

Figure 6. Interfacing a Generic Programmer to the MMPF0100 in an Application Board

4 References

Following are URLs where you can obtain information on Freescale products and application solutions:

Document Number and Description		URL
MMPF0100	Data Sheet	http://cache.freescale.com/files/analog/doc/data_sheet/MMPF0100.pdf
MMPF0100ER	Errata	http://cache.freescale.com/files/analog/doc/errata/MMPF0100ER.pdf
PFSERIESFS	Fact Sheet	http://cache.freescale.com/files/analog/doc/fact_sheet/PFSeriesFS.pdf
Freescale.com Support Pages		**URL**
Freescale.com		http://www.freescale.com
Product Summary Page		http://www.freescale.com/webapp/sps/site/prod_summary.jsp?code=MMPF0100
Analog Home Page		http://www.freescale.com/analog
Power Management Home Page		http://www.freescale.com/PMIC

5 Revision History

Revision	Date	Description
2.0	5/2013	• Initial release
3.0	1/2014	• Updated section 2.2 OTP Programming Example • Added **Section 2.3, Try-Before-Buy Mode Example**, page 8 • Deleted section 2.4.3 Example Prototyping with ECC • Added (2) and (5)

How to Reach Us:

Home Page:
freescale.com

Web Support:
freescale.com/support

Document Number: AN4536
Rev. 3.0
1/2014