MAX77655
Programmer's Guide

Abstract

The MAX77655 is a compact, four-channel power supply solution using a continuous conduction mode (CCM), single-inductor-multiple-output (SIMO) architecture. The MAX77655 data sheet lists the electrical characteristics and full register map. This guide gives general advice to control the MAX77655 through digital communication.

Table of Contents

OTP Options

Certain registers have different reset values for different part numbers. For example, one One-Time Programmable (OTP) option sets SBB2 to 0.7V by default. Refer to the **Part Number Decode** section of the MAX77655 data sheet for the full table of different OTP options and their reset values.

Register Reset Conditions

All registers except the ERCFLAG register reset if one of the following events occur:

- UVLO/OVLO/OTLO
- Manual Reset
- Software Off Command
- Software Cold Reset Command

The ERCFLAG register resets only upon a power-on reset ($V_{IN} < V_{POR}$).

Interrupts and Status Registers

The interrupt register (INT_GLBL) has flags to indicate if a fault or event happened in the past. The status register (STAT_GLBL) provides information on the device state. Refer to the **Register Map** section in the MAX77655 data sheet for information on each interrupt and status bit.

Interrupts

The interrupt flags in an interrupt register are cleared when it is read.

Interrupts can be configured to pull nIRQ LOW when they are set or unmasked. This is typically used as an external interrupt for a host controller. The host controller reads the INT_GLBL to determine which event occurred after detecting the interrupt. Set the appropriate bits in the INTM_GLBL register to unmask the interrupts. The interrupt flags in the register continue updating even if they are masked.

ERCFLAG

This register contains special interrupt flags that retain their values even after a reset. These flags, such as UVLO and SFT_OFF_F, indicate which event caused a reset. Read from the ERCFLAG register to identify the cause if the device does not power up, unexpectedly resets, or unexpectedly powers down.

Unlike other interrupt flags, the interrupt flags in the ERCFLAG register cannot be configured to pull nIRQ LOW when set.

Example Pseudocode

```
// On power-up, check if any faults occurred that caused a reset.
int slave = 0x44;
ercflag_type erc = i2c.read(slave, 0x05);
if (erc.tovld || erc.ovlo || erc.uvlo){
    // …
}

//…

// Set up channel faults to assert the nIRQ pin.
i2c.write(slave, 0x03, 0xF0); // Unmask the SBBx_F flags.
//…
void nirq_handler(){
    // Read the INT_GLBL register to see which interrupt occurred.
    int_type interrupts = i2c.read(slave, 0x02);
    if (interrupts.sbb0_f){ // SBB0 had a fault.
        //…
    }
    //…
}
```

Global Management

Configuring nEN

Choose an OTP with the most appropriate nEN settings since nEN triggers the power-up sequence and enables I^{2}C communication. Do the following if custom settings are required:

- Write to CNFG_GLBL_A.nEN_MODE to choose a mode: push-button (0), slide-switch (1), or logic (2).
    - Use logic mode if nEN is connected to a digital pin, for example a GPIO.
- Write to CNFG_GLBL_A.PU_DIS to enable (0) or disable (1) the internal nEN pullup resistor.
- Write to CNFG_GLBL_A.DBEN_nEN to configure the debounce time.
- Write to CNFG_GLBL_A.MRST to configure the manual reset time.

Configuring the Bias

The device can be configured to draw lower quiescent current by changing the bias power mode. However, the tradeoff for lower quiescent current is slower response time to load changes and higher ripple. Write to the CNFG_GLBL_A.BIAS_LPM bit to set the bias in low (1) or normal (0) power mode. BIAS_LPM = 1 is a request for the device to be in the low-power mode. The device automatically transitions to the normal-power mode as necessary.

Software Control

nEN is the hardware method used to enable or disable the device. The host controller can command the device to power off or reset with the CNFG_GLBL_B.SFT_CTRL[2:0] bit field.

Standby State

Write CNFG_GLBL_B.SFT_CTRL[2:0] = 0x3 to transition to the standby state to only turn off channels in the flexible power sequence while keeping the device enabled. The host controller can reconfigure channels and reorganize the power sequence while in the standby state.

Write CNFG_GLBL_B.SFT_CTRL[2:0] = 0x4 to exit the standby state.

Example Pseudocode

```
// Configure nEN to be in logic mode and disable the internal pullup.
int slave = 0x44;
cnfg_glbl = i2c.read(slave, 0x0);
cnfg_glbl &= 0b1111 1001
cngf_glbl |= 0b0010 0100; // Set nEN_MODE = 0b10 and PU_DIS = 1.
i2c.write(slave, 0x0, cnfg_glbl);

//…

// Enter standby state to place SBB3 in slot 1 and set its voltage for the next power-up
i2c.write(slave, 0x01, 0x03); // Enter standby state
i2c.write(slave, 0x0E, 0x7C); // Set SBB3 = 3.6V
i2c.write(slave, 0x0F, 0b1001); // Place SBB3 in FPS slot 1
i2c.write(slave, 0x01, 0x04); // Exit standby state

//…
```

Single-Inductor-Multiple-Output (SIMO) Regulator

Setting the Output Voltage

Set the output voltage of SBBx with CNFG_SBBx_A.TV_SBBx[7:0]. Calculate the value with:

$$TV\_SBBx[7:0] = \frac{V_{SBBx} - 0.5V}{0.025V}$$

The values of 0x8D and higher are reserved.

Checking for Overload

Check the STAT_GLBL.SBBx_S status to check if the channel SBBx is overloaded and most likely out of regulation. Check the INT_GLBL.SBBx_F interrupts to check if the channel SBBx was overloaded in the past.

Example Pseudocode

```
// Set SBB1/2/3 to 2.5V, 0.6V, and 1.0V, respectively.
int slave = 0x44;
i2c.write(slave, 0x0A, 0x50); // Set SBB1 = 2.5V.
i2c.write(slave, 0x0C, 0x04); // Set SBB2 = 0.6V.
i2c.write(slave, 0x0E, 0x14); // Set SBB3 = 1.0V.

//…

// Unmask the SBB2_F interrupt. If the interrupt was set, check if it is still overloaded.
i2c.write(slave, 0x03, 0b0100 0000);
//…
void nirq_handler(){
    // Read the INT_GLBL register to see which interrupt occurred.
    int_type interrupts = i2c.read(slave, 0x02);
    if (interrupts.sbb2_f){ // SBB2 had a fault.
        // Check if the channel is still overloaded
        stat_type statuses = i2c.read(slave, 0x04);
        if(!statuses.sbb2_s){
            //…
        }
        //…
    }
    //…
}

//…
```

Power Sequencing and Enabling/Disabling the Regulators

Each SIMO channel can be enabled, disabled, or placed in a power sequence slot. Set CNFG_SBBx_B.EN_SBBx[2:0] = 0x6 (enable) or 0x4 (disable) to enable or disable the channels. Write the slot number to EN_SBBx[2:0] to set a regulator in a power sequence slot. The slot numbers range from 0 to 3.

Active Discharge

Each channel has an active discharge resistor to quickly drop the voltage to 0V when the regulator is disabled. Set CNFG_SBBx_B.ADE_SBBx = 1 to enable a regulator's active discharge resistor.

Revision History

REVISION NUMBER	REVISION DATE	DESCRIPTION	PAGES CHANGED
0	04/20	Initial release	—