MAX1781 Advanced Smart Battery-Pack Controller

General Description

The MAX1781 smart battery-pack controller integrates a user-programmable microcontroller core, a Coulomb-counting fuel gauge, a multi-channel data-acquisition unit, and an SMBus v1.1 compliant master/slave SMBus interface. The 8-bit, RISC microcontroller core has an integrated 12k bytes of user programmable EEPROM, which provides battery-pack designers with complete flexibility in developing fuel gauging and control algorithms. The MAX1781 is equipped with full ICE (in-circuit emulator) capability for efficient firmware development and debugging.

The 16-bit data-acquisition unit measures individual cell voltages, cell stack voltage, chip internal/external temperature, and two general purpose analog inputs. Individual cell voltage measurements with 0.5% accuracy and over-current protection allow the MAX1781 to eliminate a separate 1st level protection IC. Adjustable over-current thresholds and delay timers provide a flexible solution.

The integrating fuel gauge module provides a typical input offset of less than 1uV, and gain accuracy of better than 1% with no trimming required during pack manufacture. The MAX1781 has a wide 4V to 28V operating voltage range. The IC is available in a 7mm x 7mm 48-TQFN package with a maximum thickness of 0.8mm.

Applications

Notebook Battery Packs
Proprietary Battery Packs
Data-Acquisition Systems
Remote Data Logging

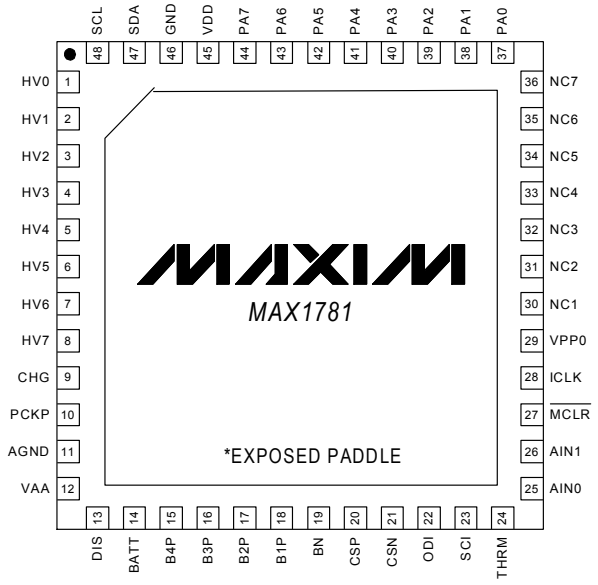
Ordering Information

PART	TEMP RANGE	PIN-PACKAGE
MAX1781ETM	-40°C to +85°C	48 TQFN

Features

- **Accurate Fuel Gauge Uses V to F Method**
  - **<1µV Input Offset Voltage**
  - **<1% Gain Accuracy**
  - **No Calibration Required**
- **8-bit RISC Microcontroller Core**
- **12k Bytes of Program/Data EEPROM**
- **64 Bytes of Byte Writeable EEPROM**
- **264 Bytes of RAM**
- **Watchdog Timer**
- **ICE for Firmware Debugging**
- **Integrated Development Environment (IDE)**
  - **C Compiler**
  - **Inline Assembler**
- **Eliminates Separate Primary-Protection IC**
  - **Individual Cell Voltage Measurements with 0.5% accuracy**
  - **Built-In Protection MOSFET Gate Drivers**
  - **8% Accurate Overcurrent Protection**
  - **Programmable Over-Current Delay Timers**
- **SMBus v1.1-Compliant, with Master Capability**
- **Typical Operating Current <120µA**
- **Typical Shutdown Current <10nA**
- **Integrated 32kHz Oscillator: No Crystal**
- **Fully Integrated LDO (V_{IN} = 4V to 28V)**

Pin Configuration



Typical Operating Circuit

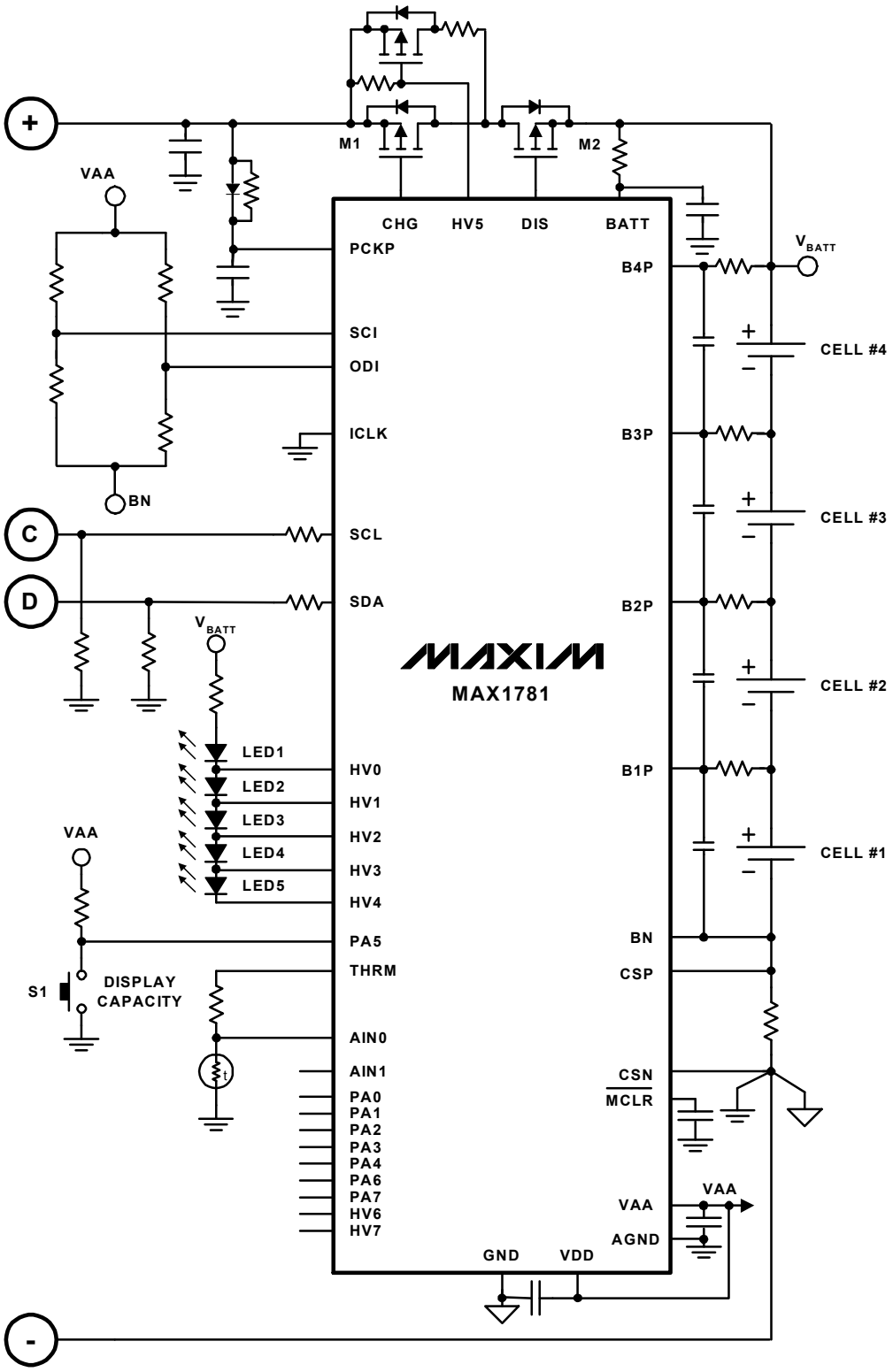


TABLE OF CONTENTS

Table Of Figures

Table Of Registers

Absolute Maximum Ratings

(V_{DD} = V_{AA}, AGND = GND, unless otherwise noted.)

BATT, PCKP, SCL, SDA, HV[7:0] to GND ..... -0.3V to +30V
B3P, B2P, B1P, DIS to GND..... -0.3V to V_{BATT}+0.3V
B4P to BATT ..... -0.3V to +0.3V
CHG to GND ..... -0.3V to V_{PCKP}+0.3V
PA[7:0] to GND ..... -0.3V to V_{DD}+0.3V
AIN0, AIN1, ODI, SCI (Note 1), THRM, ICLK to GND..... -0.3V to V_{AA}+0.3V
V_{AA}, V_{DD}, $\overline{MCLR}$ to GND..... -0.3V to 6V
BN, CSP to GND..... -2V to 6V
CSN to GND..... -1V to 1V
AGND to GND..... -0.3V to 0.3V
SCI source current ..... -10mA to +10mA
Continuous Power Dissipation (T_A=+70°C)
TQFN (derating 26.3mW/ºC).....2105mW
Storage Temperature ..... -60°C to +150°C
Lead Temperature (soldering, 10s)..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications in not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS

(V_{BATT} = V_{PCKP} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = 0°C to +85°C**, unless otherwise noted. Typical values are at T_A = +25ºC.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
BATT LINEAR REGULATOR (V_{AA})						
Input Voltage Range	Includes dropout operation		3.5		28	V
Output Voltage	4V < V_{BATT} < 28V, 0 < I_{LOAD} < 10mA		3.2	3.4	3.6	V
V_{AA} to BATT Leakage Current	V_{BATT} = 0V, V_{AA} = 3.4V, BATTON = 1			3	10	µA
	V_{BATT} = 0V, V_{AA} = 3.4V, BATTON = 0			23	100	µA
Short-Circuit Current	V_{DD} = 2.8V, 4V < V_{BATT} < 28V		20	40	80	mA
PCKP LINEAR REGULATOR (V_{AA})						
Input Voltage Range	Includes dropout operation		3.5		28	V
Output Voltage	4V < V_{PCKP} < 28V, 0 < I_{LOAD} < 10mA		3.2	3.4	3.6	V
V_{AA} to PCKP Leakage Current	V_{PCKP} = 0V, V_{AA} = 3.4V, BATTON = 0			23	100	µA
	V_{PCKP} = 0V, V_{AA} = 3.4V, BATTON = 1			23	100	µA
PCKP Supply Current	BATTON = 1	V_{PCKP} = 20V		5	15	µA
		V_{PCKP} = 28V		5	50	µA
Short-Circuit Current	V_{AA} = 0V, 4V< V_{PCKP} < 28V		20	40	90	mA
Power-Fail Warning Trip Level (PFW)	Falling V_{AA}		2.91	3.0	3.09	V
	Rising V_{AA}				3.2	
	Hysteresis on rising V_{AA}			24		mV
Power-On Reset Trip Level ($\overline{MCLR}$)	Falling V_{AA}		2.62	2.7	2.78	V
	Rising V_{AA}				2.9	
	Hysteresis on rising V_{AA}			24		MV
SUPPLY CURRENT (BATT) BATTON = 1						
SHUTDOWN (Everything Off)	BATTON = 0			0.001	1	µA
SLEEP:	Linear Regulator, 32kHz Clock			56	100	µA

(V_{BATT} = V_{PCKP} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = 0°C to +85°C**, unless otherwise noted. Typical values are at T_A = +25ºC.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
SLEEP + FG + ODSC:	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC State		110	200	µA
CPU RUN + FG + ODSC:	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC state, Instructions Executing		1800	4000	µA
SLEEP + AFE/ADC + FG + ODSC:	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC state, ADC Running		810	1400	µA
PME2	Programming Program Memory EEPROM		1000	2500	µA
CPURUN + FG + ODSC + DME2	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC state, Instructions Executing, Programming Data Memory EEPROM		1900	4000	µA
Cancellation Circuit Supply Current from BATT (Only present during ADC Conversions with Cancellation Enabled, Not Included in Currents Above).	One cancellation circuit on. 15V applied to cancellation pin;		30	60	µA
	Two cancellation circuits on. 15V applied to cancellation pins		60	120	
FUEL GAUGE					
Input Voltage Offset			1		µV
Input Resistance	Between CSP and CSN	75	150	300	kΩ
Charge Coulomb-Counter Accumulation Rate	V_{CSP} - V_{CSN} = 125mV	49,250	50,000	50,750	Counts/s
Discharge Coulomb-Counter Accumulation Rate	V_{CSP} - V_{CSN} = -125mV	49,250	50,000	50,750	Counts/s
Gain Variation	-125mV < V_{CSP} - V_{CSN} < 125mV, T_A = +25°C		0.3		%
	-125mV < V_{CSP} - V_{CSN} < 125mV		0.5		
Current Measurement Accuracy (When Using Internal 32kHz Oscillator as a Time Base)	V_{CSP} - V_{CSN} = 125mV, Excluding Offset Error	-1		+1	%
ADC					
Conversion Time	9-Bit		0.885		ms
	10-Bit		1.62		
	11-Bit		3.08		
	12-Bit		6.01		
	13-Bit		11.9		
	14-Bit		23.6		

(V_{BATT} = V_{PCKP} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = 0°C to +85°C**, unless otherwise noted. Typical values are at T_A = +25°C.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
	15-Bit		47.0		
	16-Bit		93.9		
LSB Bit Weight	High voltage, full scale = 20.48V		500		µV
	Low voltage, full scale = 5.12V		125		
	BN to CSN, full scale = 0.4096V		10		
Quantization Error		-1		+1	LSB
Gain Accuracy at Full Scale	V_{AA}, THRM, AIN0, AIN1, V_{DD} = V_{AA} = V_{BATT} = 3.4V	-1		+1	%
	V_{BN}-V_{CSN} = 0.4096V	-1.5		+1.5	
	B1P, B2P, B3P, B4P referenced to BN	-1		+1	
BN-CSN Gain Matching between Positive and Negative Inputs	BN-CSN, V_{BN}-V_{CSN}=+/-0.4V	-1		+1	%
Integral Nonlinearity (INL)			0.01		%
Input Offset Voltage	V_{BN}-V_{CSN} (12-bit resolution)		2	200	µV
Input Voltage Range	B1P, B2P, B3P, B4P referenced to BN	-0.2		+20.48	V
	V_{BN} for measurements of B1P, B2P, B3P, B4P	-2		+5.12	
	THRM, AIN0, AIN1	-0.2		V_{AA} + 0.2	
	V_{BN} – V_{CSN}	-0.4092		+0.4092	
Input Resistance to AGND with ADC (ON)	High-voltage range: BN, B1P, B2P, B3P, B4P; No input bias current cancellation		4		MΩ
	Low Voltage range: AIN0, AIN1, THRM		1		
	BN, CSN		80		kΩ
Input Bias Current with ADC and Input Bias Current Cancellation On (B3P, B2P, B1P)	V_{B4P} = 20V, V_{B3P} = V_{B2P} = V_{B1P} = 15V		80		nA
Input Bias Current with ADC Off	V_{B4P} = V_{B3P} = V_{B2P} = V_{B1P} = 20.48V		0	1	µA
Input Bias Current with ADC Off	V_{BN} = V_{AIN0} = V_{AIN1} = 3.4V		0	1	µA
ADC CELL DIFFERENTIAL VOLTAGE MEASUREMENT ACCURACY (14-BIT RESOLUTION):					
B4P to B3P	V_{B3P} = 12.75V, V_{B4P} = 17V	-33	0	+33	mV
B3P to B2P	V_{B2P} = 8.5V, V_{B3P} = 12.75V	-33	0	+33	mV
B2P to B1P	V_{B1P} = 4.25V, V_{B2P} = 8.5V	-33	0	+33	mV
B1P to BN	V_{BN} = 0V, V_{B1P} = 4.25V	-33	0	+33	mV
TEMPERATURE SENSOR					
THRM Pull High Impedance	V_{AA}= V_{DD} = V_{PCKP} = V_{BATT} = 3.4V, 2mA source current		30	75	Ω
THRM High-Impedance Leakage				1	µA
SHORT-CIRCUIT AND OVER-DISCHARGE COMPARATORS					
ODI, SCI Input Offset Voltage		-2.5		+2.5	mV

(V_{BATT} = V_{PCKP} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = 0°C to +85°C**, unless otherwise noted. Typical values are at T_A = +25°C.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
ODI, SCI Input Bias Current			2		nA
ODI, SCI Comparator Propagation Delay in Fault-Trip Direction	20mV input overdrive		2		µs
ODI Delay	ODSCDLY=0x83	2.869	2.982	3.102	ms
SCI Delay	ODSCDLY=0x83	418	488	560	µs
SCI Service-Mode Entry Current		-3		-2	mA
CHARGE AND DISCHARGE FET DRIVERS					
DIS Source Current	V_{BATT} = 20V, V_{BATT} - V_{DIS} = 3V	1.6	5		mA
CHG Source Current	V_{PCKP} = 20V, V_{PCKP} - V_{CHG} = 3V	1.6	5		mA
CHG, DIS Output Sink Current	V_{PCKP} = V_{BATT} = 20V, V_{DIS} = V_{BATT} - V_{CLAMP} + 2V, V_{CHG} = V_{PCKP} - V_{CLAMP} + 2V,	0.3	1.2		mA
CHG, DIS Pull-Up Resistance	V_{BATT} = 20V, V_{BATT} - V_{DIS} = 0.5V, V_{PCKP} = 20V, V_{PCKP} - V_{CHG} = 0.5V		4		MΩ
DIS Output Voltage High V_{BATT} - V_{DIS}			0	0.2	V
CHG Output Voltage High V_{PCKP} - V_{CHG}			0	0.2	V
DIS Clamp Voltage (V_{CLAMP}) V_{BATT} - V_{DIS}	V_{BATT} = 20V, no load	5	8.2	11	V
CHG Clamp Voltage (V_{CLAMP}) V_{PCKP} - V_{CHG}	V_{PCKP} = 20V, no load	5	8.2	11	V
DIS, CHG Voltage Output Low	No load, V_{BATT} = V_{PCKP} = 4V	1.2			V
INSTRUCTION OSCILLATOR					
Frequency	2.7V < V_{DD} < 3.6V	5.25	6	6.75	MHz
INTERNAL TIMER OSCILLATOR					
32.768kHz Oscillator Accuracy	12-cycle average	-1.5		1.5	%
PORT A LOGIC INPUTS/OUTPUTS—PA[7:0]					
Input Voltage Low	2.7V < V_{DD} < 3.6V			0.8	V
Input Voltage High	2.7V < V_{DD} < 3.6V	2.4			V
High-Impedance Leakage Current	I/O pins programmed to Hi-Z		0.01	1	µA
Output Voltage Low	I_{SINK} = 2mA			0.25	V
Output Voltage High	I_{SOURCE} = 2mA	V_{DD} - 0.2			V
$\overline{MCLR}$ RESET PIN					
$\overline{MCLR}$ Output Voltage Low	I_{SINK} = 1mA, and V_{BATT} = V_{DD} = 2.5V			0.2	V
$\overline{MCLR}$ Output Voltage High	I_{SOURCE} = 0, V_{BATT} = V_{DD} = 3.4V	V_{DD} - 0.2			V
$\overline{MCLR}$ Pullup Resistance	Internally connected from $\overline{MCLR}$ to V_{AA}	10	20	40	kΩ
$\overline{MCLR}$ Input High Voltage		2.4			V
$\overline{MCLR}$ Input Low Voltage				1	V
$\overline{MCLR}$ Input Hysteresis			300		mV
HIGH-VOLTAGE OPEN-DRAIN OUTPUTS (HV0–HV7)					

(V_{BATT} = V_{PCKP} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = 0°C to +85°C**, unless otherwise noted. Typical values are at T_A = +25ºC.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Output Voltage Low	I_{SOURCE} = 10mA		0.7	1.4	V
Leakage Current	$V_{HV[7:0]}$ = 28V			1	µA
HIGH-VOLTAGE PINS IN PARALLEL ACCESS MODE (HV0–HV3)					
Input High Voltage		V_{DD} - 0.6			V
Input Low Voltage				0.6	V
Input Leakage Current				1	µA
SMBus PINS, SCL and SDA					
Input Low Voltage	Clock/Data input low voltage			0.8	V
Input High Voltage	Clock/Data input high voltage	2.1			V
Output Low Voltage	I_{SINK} = 2mA			0.4	V
Leakage Current	V_{SCL} = V_{SDA} = 28V			1	µA
Short-Circuit Current	V_{SCL}= V_{SDA} = 3.4V	6		50	mA

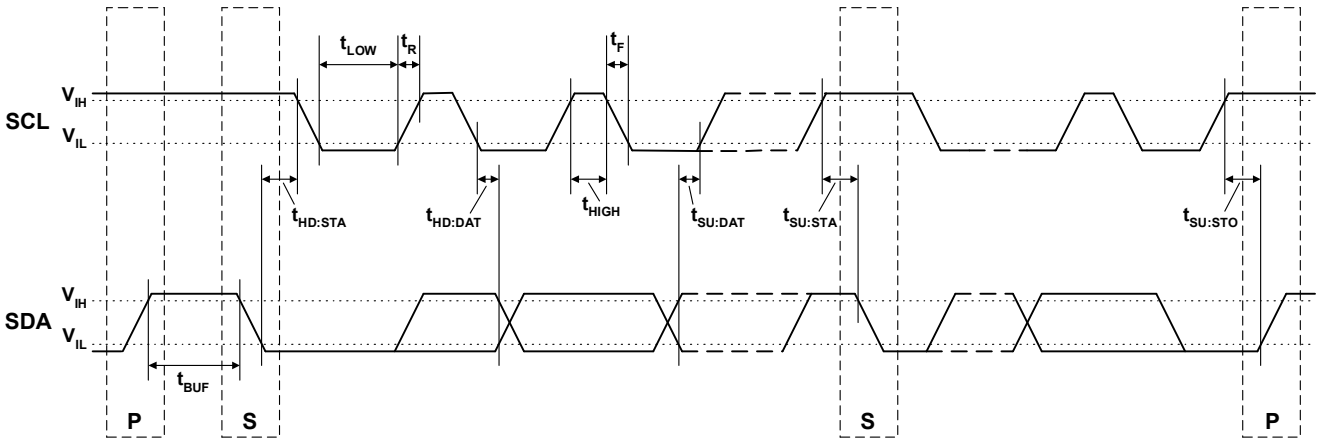


Figure 1. SMBus Timing Diagram

SMBus SLAVE MODE AC CHARACTERISTICS

(V_{BATT} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, Glitch Filter Enabled, **T_A = 0°C to +85°C**. Typical values are at +25ºC.)

SYMBOL	PARAMETER/CONDITIONS		MIN	TYP	MAX	UNITS
$t_{TIMEOUT}$	Timeout for maximum clock low time		26.8	27.4	28.0	ms
$t_{LOW:SEXT}$	Timeout for cumulative slave clock stretch time		23.0	23.4	24.0	ms
t_{BUF}	Bus free time between a stop and start condition		4.7			µs
$t_{HD:STA}$	Hold time after a start or repeated start condition		4.0			µs
$t_{SU:STA}$	Repeated start setup time		600			ns
$t_{SU:STO}$	Stop condition setup time		600			ns
$t_{HD:DAT}$	Data Hold time.	Transmit	780		2900	ns
		Receive	-30			ns

(V_{BATT} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, Glitch Filter Enabled, **T_A = 0°C to +85°C**. Typical values are at +25ºC.)

SYMBOL	PARAMETER/CONDITIONS		MIN	TYP	MAX	UNITS
$t_{SU:DAT}$	Data Setup Time	Transmit	250			ns
		Receive	200			
t_{LOW}	SCL low period		4.7			µs
t_{HIGH}	SCL high period		4.0			µs
t_{GREJ}	Glitch pulse width rejected,			1000		ns

SMBus MASTER MODE AC CHARACTERISTICS

(V_{BATT} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, Instruction Oscillator is On, Glitch Filter Enabled, **T_A = 0°C to +85°C**. Typical values are at +25ºC.)

SYMBOL	PARAMETER/CONDITIONS		MIN	TYP	MAX	UNITS
t_{BUF}	Bus free time between a stop and start condition		8.1			µs
$t_{LOW:MEXT}$	Timeout for cumulative master clock stretch time		9.6	9.8	10.0	ms
$t_{HD:STA}$	Hold time after a (repeated) start condition		7.3			µs
$t_{SU:STA}$	Repeated start setup time		5.7			µs
$t_{SU:STO}$	Stop condition setup time		5.7			µs
$t_{HD:DAT}$	Data Hold Time	Transmit	300			ns
		Receive	-30			ns
		Acknowledge	300			ns
$t_{SU:DAT}$	Data Setup Time	Transmit	250			ns
		Receive	250			ns
		Acknowledge	250			ns
f_{SCL}	SCL clock frequency (50% typical duty cycle)	MBAUD=010		85.7		kHz
		MBAUD=011 (default)		50.8		kHz
		MBAUD=100		28.0		kHz
		MBAUD=101		14.8		kHz
		MBAUD=110		7.6		kHz
		MBAUD=111		3.9		kHz

ELECTRICAL CHARACTERISTICS

(V_{BATT} = V_{PCKP} = 12V, GND=AGND=CSN=0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = -40°C to +85°C**, unless otherwise noted. Note 2.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
BATT LINEAR REGULATOR (V_{AA})					
Input Voltage Range	Includes dropout operation	3.5		28	V
Output Voltage	4V < V_{BATT} < 28V, 0 < I_{LOAD} < 10mA	3.2		3.6	V
Short-Circuit Current	V_{DD} = 2.8V, 4V < V_{BATT} < 28V	20		80	mA
PCKP LINEAR REGULATOR (V_{AA})					
Input Voltage Range	Includes dropout operation	3.5		28	V
Output Voltage	4V < V_{PCKP} < 28V, 0 < I_{LOAD} < 10mA	3.2		3.6	V
Short-Circuit Current	V_{AA} = 0V, 4V< V_{PCKP} < 28V	20		100	mA
Power-Fail Warning Trip Level (PFW)	Falling V_{AA}	2.91		3.09	V
	Rising V_{AA}			3.2	
Power-On Reset Trip Level ($\overline{MCLR}$)	Falling V_{AA}	2.62		2.78	V
	Rising V_{AA}			2.9	
SUPPLY CURRENT (BATT) BATTON = 1					
SLEEP + FG + ODSC:	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC State			200	µA
CPU RUN + FG + ODSC:	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC state, Instructions Executing			4000	µA
SLEEP + AFE/ADC + FG + ODSC:	Linear Regulator, 32kHz Clock, Fuel Gauge, Precision Reference, ODSC Comparators, MOSFET Drivers in DC state, ADC Running			1400	µA
FUEL GAUGE					
Charge Coulomb-Counter Accumulation Rate	V_{CSP} - V_{CSN} = 125mV	49,000		51,000	Counts/s
Discharge Coulomb-Counter Accumulation Rate	V_{CSP} - V_{CSN} = -125mV	49,000		51,000	Counts/s
Current Measurement Accuracy (When Using Internal 32kHz Oscillator as a Time Base)	V_{CSP} - V_{CSN} = 125mV, Excluding Offset Error	-1.5		+1.5	%
ADC					
Gain Accuracy at Full Scale	V_{AA}, THRM, AIN0, AIN1, V_{DD} = V_{AA} = V_{BATT} = 3.4V	-1.5		+1.5	%
	V_{BN} – V_{CSN} = 0.4096V	-1.5		+1.5	
	B_P referenced to BN	-1.5		+1.5	
Input Voltage Range	B1P, B2P, B3P, B4P referenced to BN	-0.2		+20.48	V
	V_{BN} for measurements of B1P, B2P, B3P, B4P	-2		+5.12	

(V_{BATT} = V_{PCKP} = 12V, GND=AGND=CSN=0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = -40°C to +85°C**, unless otherwise noted. Note 2.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
	THRM, AIN0, AIN1	-0.2		V_{AA} + 0.2	
	$V_{BN} - V_{CSN}$	-0.4092		+0.4092	
ADC CELL DIFFERENTIAL VOLTAGE MEASUREMENT ACCURACY (14-BIT RESOLUTION):					
B4P to B3P	V_{B3P} = 12.75V, V_{B4P} = 17V	-44		+44	mV
B3P to B2P	V_{B2P} = 8.5V, V_{B3P} = 12.75V	-44		+44	mV
B2P to B1P	V_{B1P} = 4.25V, V_{B2P} = 8.5V	-44		+44	mV
B1P to BN	V_{BN} = 0V, V_{B1P} = 4.25V	-44		+44	mV
TEMPERATURE SENSOR					
THRM Pull High Impedance	V_{AA}= V_{DD} = V_{PCKP} = V_{BATT} = 3.4V, 2mA source current			75	Ω
SHORT-CIRCUIT AND OVER-DISCHARGE COMPARATORS					
ODI, SCI Input Offset Voltage		-2.5		+2.5	mV
ODI Delay	ODSCDLY=0x83	2.869		3.102	ms
SCI Delay	ODSCDLY=0x83	418		560	µs
CHARGE AND DISCHARGE FET DRIVERS					
DIS Source Current	V_{BATT} = 20V, V_{BATT} - V_{DIS} = 3V	1.6			mA
CHG Source Current	V_{PCKP} = 20V, V_{PCKP} - V_{CHG} = 3V	1.6			mA
CHG, DIS Output Sink Current	V_{PCKP} = V_{BATT} = 20V, V_{DIS} = V_{BATT} - V_{CLAMP} + 2V, V_{CHG} = V_{PCKP} - V_{CLAMP} + 2V,	0.3			mA
DIS Output Voltage High V_{BATT} - V_{DIS}				0.2	V
CHG Output Voltage High V_{PCKP} - V_{CHG}				0.2	V
DIS Clamp Voltage (V_{CLAMP}) V_{BATT} - V_{DIS}	V_{BATT} = 20V, no load	5		11	V
CHG Clamp Voltage (V_{CLAMP}) V_{PCKP} - V_{CHG}	V_{PCKP} = 20V, no load	5		11	V
DIS, CHG Voltage Output Low	No load, V_{BATT} = V_{PCKP} = 4V	1.2			V
INSTRUCTION OSCILLATOR					
Frequency	2.7V < V_{DD} < 3.6V	5.25		6.75	MHz
INTERNAL TIMER OSCILLATOR					
32.768kHz Oscillator Accuracy	12-cycle average	-2		2	%
PORT A LOGIC INPUTS/OUTPUTS—PA[7:0]					
Input Voltage Low	2.7V < V_{DD} < 3.6V			0.8	V
Input Voltage High	2.7V < V_{DD} < 3.6V	2.4			V
Output Voltage Low	I_{SINK} = 2mA			0.25	V
Output Voltage High	I_{SOURCE} = 2mA	V_{DD} - 0.2			V
HIGH-VOLTAGE OPEN-DRAIN OUTPUTS (HV0–HV7)					
Output Voltage Low	I_{SOURCE} = 10mA			1.4	V
SMBus PINS, SCL and SDA					
Input Low Voltage	Clock/Data input low voltage			0.8	V
Input High Voltage	Clock/Data input high voltage	2.1			V

(V_{BATT} = V_{PCKP} = 12V, GND=AGND=CSN=0, V_{DD} = V_{AA}, C_{VAA} = 1µF, instruction oscillator is off, **T_A = -40°C to +85°C**, unless otherwise noted. Note 2.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Output Low Voltage	I_{SINK} = 2mA			0.4	V
Short-Circuit Current	V_{SCL}= V_{SDA} = 3.4V	6		50	mA

SMBus SLAVE MODE AC CHARACTERISTICS

(V_{BATT} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, Glitch Filter Enabled, **T_A = -40°C to +85°C**. Note 2.)

SYMBOL	PARAMETER/CONDITIONS		MIN	TYP	MAX	UNITS
$t_{TIMEOUT}$	Timeout for maximum clock low time		26.8		28	ms
$t_{LOW:SEXT}$	Timeout for cumulative slave clock stretch time		23.0		24	ms
t_{BUF}	Bus free time between a stop and start condition		4.7			µs
$t_{HD:STA}$	Hold time after a start or repeated start condition		4.0			µs
$t_{SU:STA}$	Repeated start setup time		600			ns
$t_{SU:STO}$	Stop condition setup time		600			ns
$t_{HD:DAT}$	Data Hold time.	Transmit	780		2900	ns
		Receive	-30			ns
$t_{SU:DAT}$	Data Setup Time	Transmit	250			ns
		Receive	200			
t_{LOW}	SCL low period		4.7			µs
t_{HIGH}	SCL high period		4.0			µs

SMBus MASTER MODE AC CHARACTERISTICS

(V_{BATT} = 12V, GND = AGND = CSN = 0, V_{DD} = V_{AA}, C_{VAA} = 1µF, Instruction Oscillator is On, Glitch Filter Enabled, **T_A = -40°C to +85°C**. Note 2)

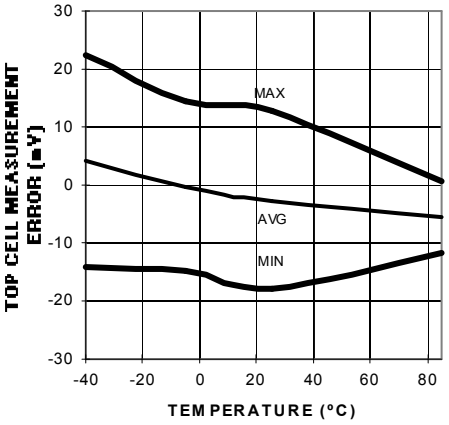
SYMBOL	PARAMETER/CONDITIONS		MIN	TYP	MAX	UNITS
t_{BUF}	Bus free time between a stop and start condition		8.1			µs
$t_{LOW:MEXT}$	Timeout for cumulative master clock stretch time		9.6		10.0	ms
$t_{HD:STA}$	Hold time after a (repeated) start condition		7.3			µs
$t_{SU:STA}$	Repeated start setup time		5.7			µs
$t_{SU:STO}$	Stop condition setup time		5.7			µs
$t_{HD:DAT}$	Data Hold Time	Transmit	300			ns
		Receive	-30			ns
		Acknowledge	300			ns
$t_{SU:DAT}$	Data Setup Time	Transmit	250			ns
		Receive	250			ns
		Acknowledge	250			ns

Note 1: The SCI pin is allowed to go to more negative voltages if the current is limited as shown in the *Absolute Maximum Ratings*.

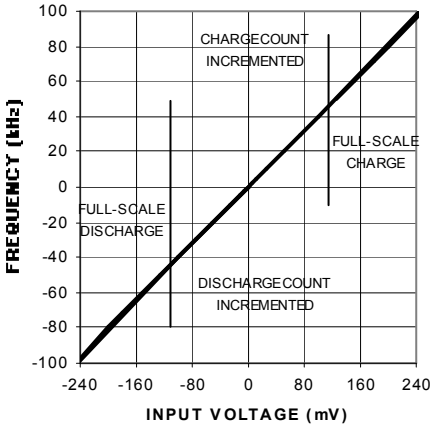
Note 2: Operation to -40ºC is guaranteed by design, not production tested.

Typical Operating Characteristics

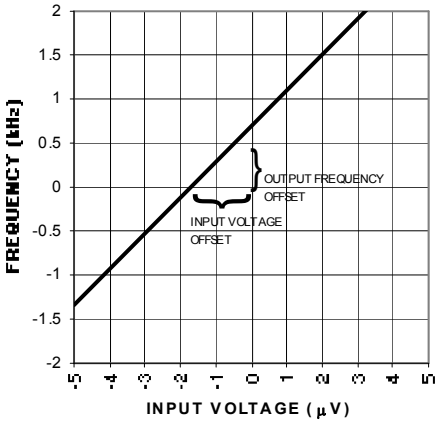
TYPICAL ADC VOLTAGE MEASUREMENT ERROR



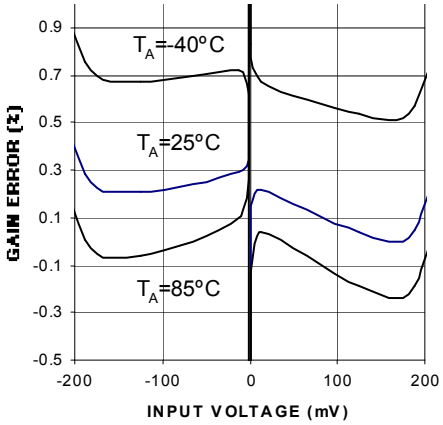
FUEL GAUGE FREQUENCY vs. INPUT VOLTAGE



FUEL GAUGE FREQUENCY vs. INPUT VOLTAGE



GAIN ERROR vs. FUEL GAUGE INPUT VOLTAGE



Pin Description

PIN	NAME	FUNCTION
1	HV0	High-Voltage Outputs. Each open-drain output can be pulled up as high as 28V using a pullup resistor and can be used for driving LEDs. See the *High-Voltage Output Port* section for more information.
2	HV1	
3	HV2	
4	HV3	
5	HV4	
6	HV5	
7	HV6	
8	HV7	
9	CHG	Charge-Protection MOSFET Driver. CHG drives an external P-channel MOSFET that allows/prevents the battery from receiving current. CHG is driven to PCKP when off and typically provides 8V gate drive when on. See the *CHG and DIS Output Driver Description* section for more information.
10	PCKP	Pack+ Connection. The system load and battery charger connect to the battery pack at PCKP. PCKP provides power to the IC and V_{AA} linear regulator when BATTON = 0 (see the Linear Regulators for System Power and Startup section).
11	AGND	Analog Ground Connection. Connect AGND to GND at a single point to form a star ground.
12	V_{AA}	3.4V Linear Regulator Output. Bypass V_{AA} to AGND with a 0.47µF capacitor close to the IC. Connect V_{AA} to V_{DD}.
13	DIS	Discharge Protection MOSFET Driver. DIS drives an external P-channel MOSFET that allows/prevents the battery from delivering current. DIS is driven to BATT when off and typically provides 8V gate drive when on. See the *CHG and DIS Output Driver Description* section for more information.
14	BATT	Battery Connection. Connect BATT to the battery. BATT should not be exposed to the outside of the battery pack. BATT provides power to the IC and V_{AA} linear regulator during normal operation when BATTON = 1 (see the Linear Regulators for System Power and Startup section).
15	B4P	Positive Terminal of Top Cell. B4P senses the voltage at the top of the battery stack.
16	B3P	Positive Terminal of Third Cell. B3P senses the voltage at the positive terminal of the third cell. Connect B1P, B2P, and B3P for 2 cell packs. Connect B2P and B3P for 3 cell packs.
17	B2P	Positive Terminal of Second Cell. B2P senses the voltage at the positive terminal of the second cell.
18	B1P	Positive Terminal of First Cell. B1P senses the voltage at the positive terminal of the first cell.
19	BN	Negative Terminal of Battery. BN senses the voltage at the negative terminal of the battery.
20	CSP	Battery Current-Sense Positive Input. Battery current for fuel gauging is sensed from CSP to CSN.
21	CSN	Battery Current-Sense Negative Input. Battery current for fuel gauging is sensed from CSP to CSN. Connect CSN to AGND.
22	ODI	Over-Discharge Current Sense. Connect a resistive voltage divider from a reference to the Current Sense Resistor, with ODI as the center-tap. Over-discharge is detected when V_{ODI} is below CSN for a set time. See the *Overcurrent Protection and Protection MOSFET Drivers* section for more information.
23	SCI	Short-Circuit Current Sense. Connect a resistor-divider from a reference to the Current Sense Resistor, with SCI as the center-tap. A short circuit is detected when V_{SCI} is below CSN for a set time. See the *Overcurrent Protection and Protection MOSFET Drivers* section for more information. Drive this pin 3mA below ground during reset to place the part into Service Mode.
24	THRM	Thermistor Connection and General-Purpose ADC Channel. THRM can be configured to connect to V_{AA} using an internal 100Ω switch or left floating for general-purpose ADC measurements. See the *AIN and THRM Pins* section for more information.
25	AIN0	General-Purpose ADC Channels. See the *AIN and THRM Pins* section for more information.
26	AIN1	
27	$\overline{MCLR}$	Manual Clear/Reset Input. Pull $\overline{MCLR}$ low to reset the MAX1781. $\overline{MCLR}$ is pulled low internally during power failure or watchdog timeout. See the Linear Regulators for System Power

		and Startup section for more information.
28	ICLK	Instruction Oscillator Input. Connect ICLK to GND. Not used in normal operation.
29	V_{PP0}	Charge-Pump Output. Do not connect this pin.
30–36	NC1–NC7	No Connection. Not internally connected.
37	PA0	Port A Input/Output. Each pin PA0–PA7 can be independently configured as either a logic-level input or output. PA3 and PA4 can also be configured as edge-triggered interrupts. PA6 and PA7 can be configured to output the status of the CHG and DIS MOSFETs. See the Port A section for more information.
38	PA1	
39	PA2	
40	PA3	
41	PA4	
42	PA5	
43	PA6	
44	PA7	
45	V_{DD}	Digital Power Supply. Connect V_{DD} to V_{AA}. Bypass V_{DD} with a 0.47µF capacitor to GND.
46	GND	Digital Ground. Connect GND to AGND.
47	SDA	SMBus Open-Drain Data Input/Output. See the *SMBus Interface* section for more information.
48	SCL	SMBus Open-Drain Clock Input/Output. In master mode SCL is an output. In slave mode it is an input. See the *SMBus Interface* section for more information.
Paddle	AGND	Connect the exposed paddle to AGND.

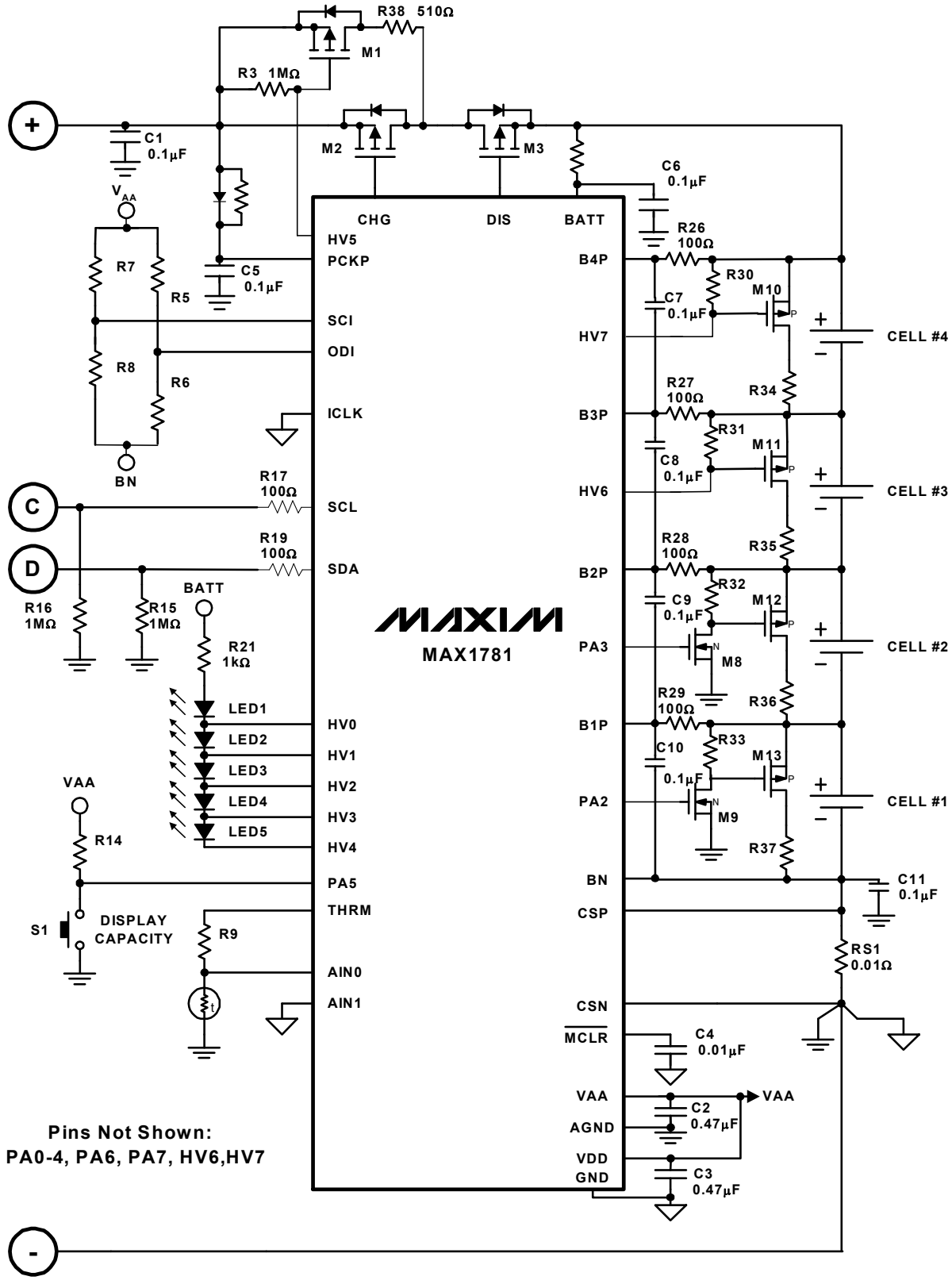


Figure 2. Max1781 Typical Application Circuit

Architectural Overview

The MAX1781 advanced smart battery-pack controller provides smart battery-pack designers with a flexible solution that accurately measures individual cell characteristics. It combines an 8-bit RISC microprocessor core, 6k x 16-bit EEPROM program memory, 264 bytes of data memory, and 64 bytes of data EEPROM, with an arsenal of precision analog peripherals.

Harvard Architecture

The MAX1781 uses a Harvard architecture in which program and data are accessed on separate buses. This improves bandwidth over traditional von Neumann architectures where program and data are fetched on the same bus. Separating program and data memory further allows instructions to be sized differently than the 8-bit wide data word. A 13-bit wide program memory bus fetches a 16-bit instruction in a single cycle. The 16-bit wide instruction op codes make it possible to have all single word instructions.

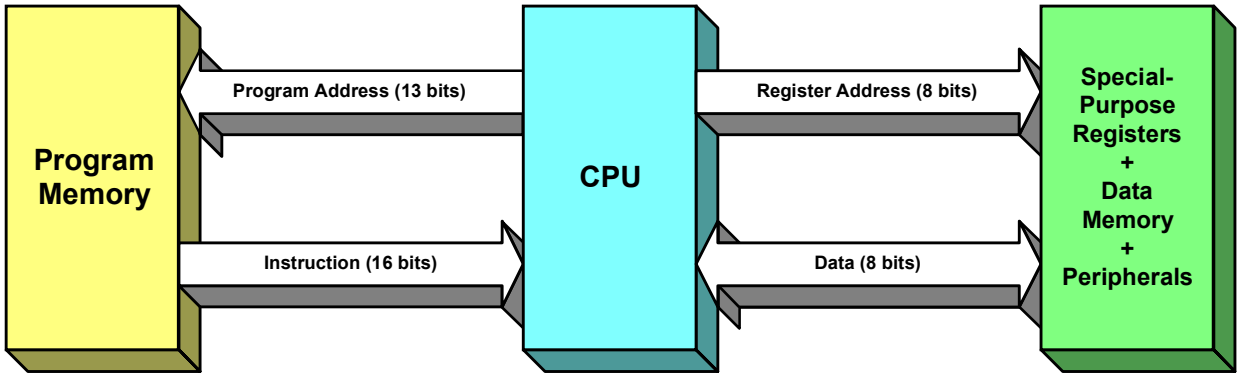


Figure 3. MAX1781 Harvard Architecture

The MAX1781 addresses 6k x 16-bit of internal program EE memory, organized in a single 6k x 16-bit program memory block. The program memory can be programmed under CPU control, SMBus control, or by an 8-bit parallel interface on the PA[7:0] pins (see the *Parallel Programming Mode* section).

Figure 3 shows a block diagram of the MAX1781 microcontroller. The MAX1781 memory is organized into program memory and data memory. Paging of program memory is not required in the MAX1781. Data memory is composed of RAM and registers, organized as sixteen 256 byte groups or pages. The data memory is divided into page-independent registers and page-dependent registers. Data memory pages are selected using the file selection register (FSR).

On-board peripherals are memory mapped onto page 0. Most of the on-board peripherals function autonomously, generating interrupts to the processor core when service is required.

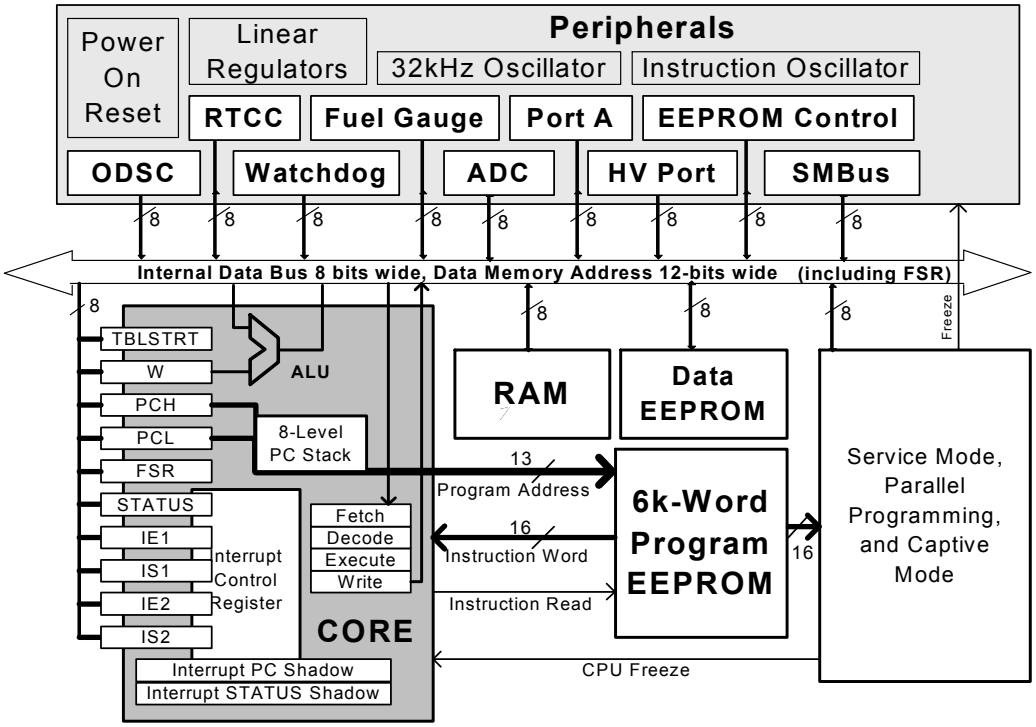


Figure 4. MAX1781 System Block Diagram

MAX1781 Digital Core

Instruction Execution

The MAX1781 processor core incorporates a two-stage pipeline. An instruction cycle takes 4 cycles of the instruction oscillator (6MHz typ). The instruction fetch and execution are pipelined such that an instruction fetch takes one instruction cycle, while the instruction decode and execution takes from 1 to 3 instruction cycles. Due to pipelining, each fetch effectively takes no time. If an instruction causes the program counter to change, for example a GOTO instruction, then a penalty of one cycle is incurred to fetch the new instruction and fill the pipeline. Figure 5 shows how instruction pipelining increases the processor throughput.

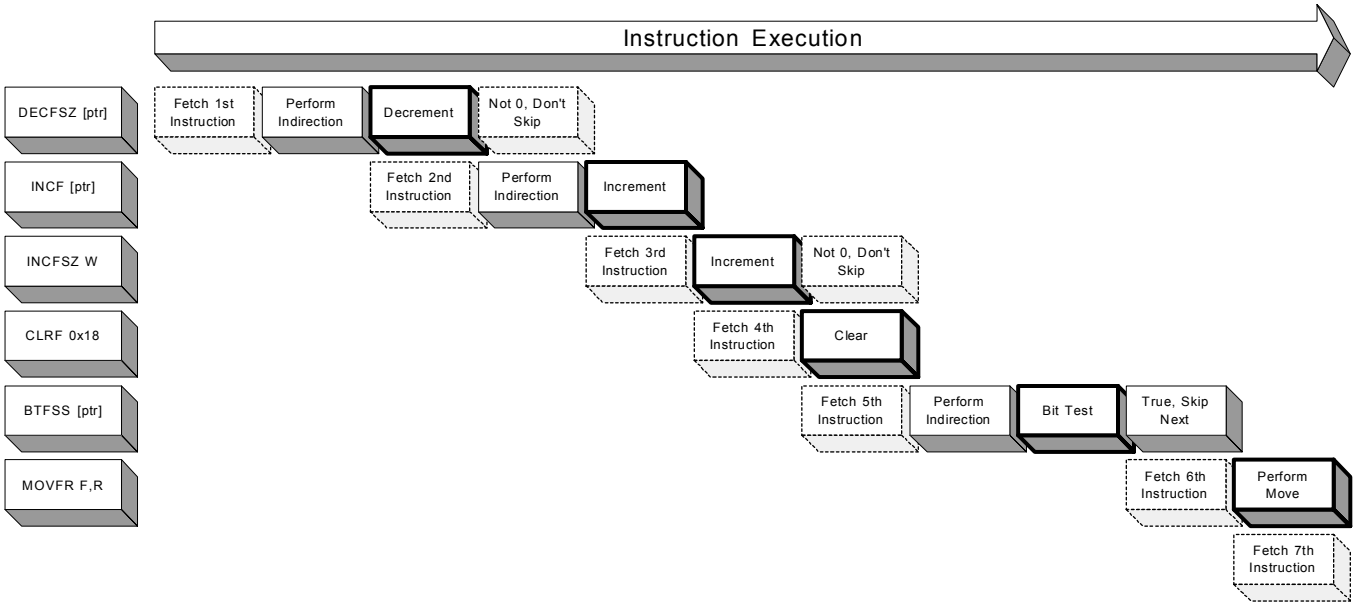


Figure 5. MAX1781 Instruction Pipeline

Instruction Pipeline

Instruction execution is completed within 1 instruction cycle with the following exceptions:

1) If indirect addressing is used (see Table 5. Instruction Table), an additional instruction cycle is required to fetch the pointer's address as well as the data pointed to by the pointer.
2) Instructions that redirect program flow require an extra instruction cycle if program flow is redirected because the MAX1781 has to modify the program counter.
3) TBLRDW requires an additional instruction cycle in order to access the working (W) register, TBLSTRT, and the program memory in question.
4) An additional instruction cycle is required when data EEPROM is accessed.

Core Register Descriptions

Program Counter (PCH and PCL)

As a program instruction is executed, the program counter (PC) contains the address of the next program instruction to be executed. The PC value is increased by 1 every instruction cycle, unless it is changed by an instruction such as GOTO, CALL, Interrupt, RETURN, RETW, or RETFIE.

Program Stack and Interrupt PC Shadow

The MAX1781 has an eight-level hardware stack used for saving and restoring the program counter during subroutine calls. The stack allows program developers to create code with nested subroutine calls. A CALL instruction pushes the current value onto the stack to save the processor context. Since the MAX1781 program stack has eight levels, subroutine calls must be used carefully. Code should not have unlimited nesting of subroutine calls, or a stack overflow is possible. The 9th subroutine call generates a stack overflow. If interrupts are enabled, the ISR (Interrupt Service Routine) is executed and the MAX1781 resets after the corresponding

status bit is cleared. Program execution cannot be completed properly after this point and the stack overflow handler should perform a final program cleanup.

The MAX1781 also includes a PC shadow register and STATUS shadow register used for processing interrupts. Just before executing the ISR the corresponding registers are saved to these shadow registers. Both the Program Counter and STATUS register are restored upon the execution of a RETFIE instruction.

A subroutine call is completed with a RETURN or RETLW instruction, both of which pop the contents of the stack into the program counter. If the 8-level stack is already empty an underflow interrupt is generated, and a reset will follow after PCOVFI is cleared. The RETLW instruction also loads the W register with the literal value specified in the instruction.

Working Register (W)

The W register serves as an accumulator or temporary storage register for many instructions. The W register is also used in every arithmetic operation.

File Selection Register (FSR)

The page-dependent memory locations (with lower 8-bit address higher than 0x80) are selected using the File Selection Register (FSR). Because the most commonly used pages neighbor page 0, program memory is conserved by modifying the FSR with INCF and DECF instead of MOVLW+MOVWF, thereby saving one instruction during page switching. See the *Data Memory Map* section for more details about paged and unpaged memory.

Arithmetic Logic Unit (ALU)

The MAX1781 CPU contains an 8-bit ALU and working register. The ALU is a general-purpose arithmetic unit. It performs arithmetic and Boolean functions between data in the working register and any data memory location.

The ALU is 8-bits wide and capable of addition, subtraction, shift, and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions one operand is W register. The other operand is either a register or an immediate constant. In single operand instructions, the ALU operates on any register.

Depending on the instruction executed, the ALU can affect the values of the carry (C), digit carry (D), and zero (Z) bits in the STATUS register.

Status Register (STATUS)

The STATUS register contains status bits Z, D, and C, which are modified according to the results of ALU instructions. The zero flag, Z, indicates that the result of the previous operation was zero. The carry flag, C, indicates that there was an overflow/underflow during the previous add, subtract, or rotate operation. The half carry flag, D, indicates that there was an overflow in the lower nibble during the previous add or subtract. See Table 5. Instruction Table for details regarding the STATUS bits effected by various instructions.

The STATUS register also contains the global interrupt enable bit (GIE) and the INTPROC bit, which indicates an interrupt is in progress. Unlike other interrupt status bits, the INTPROC bit is cleared automatically upon returning from interrupt (RETFIE) and does not need to be cleared manually.

TBLSTRT and Accessing Data Tables in Program Memory

The MAX1781 has provisions for reading program memory EEPROM through firmware. This is often used to store information about battery efficiency/capacity vs. temperature, discharge current, charge current, cell voltage,

and cycle life. This data is accessed using the TBLRDW (see the *Instruction Set* section) instruction and the TBLSTRT register. The TBLRDW instruction is a two operand instruction which combines the TBLSTRT register and register **f** to address program memory as shown in Table 1. The LSB indicates whether the least or most significant byte is read, since program memory is word addressed. Table 1 shows the mapping of TBLSTRT and register **f** to Program Memory. If bit 0 of register **f** is 0 the LSByte at the program memory address is stored in W. If D0 of f is 1 the MSByte at the program memory address is stored in W.

Table 1. Program Memory Addressing using TBLRDW

TBLSTRT								Data Pointed to By Operand f from TBLRDW Instruction							
D7	D6	D5	D4	D3	D2	D1	D0	D7	D6	D5	D4	D3	D2	D1	D0
Not Used		Program Memory Address													0 -> LSByte
		PM12	PM11	PM10	PM9	PM8	PM7	PM6	PM5	PM4	PM3	PM2	PM1	PM0	1 -> MSByte

Core Registers

W—Working Register

(Read/Write)
Address 0x00

BIT	POR	NAME	DESCRIPTION	TYPE
D7-D0	0x00	W[7:0]	Working register.	R/W

STATUS—Status Register

(Read/Write)
Address 0x01

BIT	POR	NAME	DESCRIPTION	TYPE
D7	0	INTPROC	Processor is busy processing interrupt.	R
D6	0	GIE	Global Interrupt Enable.	R/W
D5	0	—	Reads as 0.	—
D4	0	—	Reads as 0.	—
D3	0	—	Reads as 0.	—
D2	0	C	Full carry/borrow flag.	R/W
D1	0	D	Half carry/borrow flag.	R/W
D0	0	Z	Zero flag.	R/W

PCL—Program Counter Low Byte

(Read/Write)
Address 0x02

BIT	POR	NAME	DESCRIPTION	TYPE
D7-D0	0x00	PC[7:0]	Program Counter Low Byte.	R/W

PCH—Program Counter High Byte

(Read/Write)
Address 0x03

BIT	POR	NAME	DESCRIPTION	TYPE
D7-D0	0x00	PC[15:8]	Program Counter High Byte (also PC[15:8]). PC[15:13] read as 0.	R/W

FSR—File Select Register

(Read/Write)
Address 0x04

BIT	POR	NAME	DESCRIPTION	TYPE
D7-D4	0000	FSR[7:4]	Reads as 0.	R
D3-D0	0000	FSR[3:0]	Page select for the upper half of the data memory locations 0x80 to 0xFF.	R/W

TBLSTRT—Table Starting Page Register for TBLRDW Instruction

(Read/Write)
Address 0x05

BIT	POR	NAME	DESCRIPTION	TYPE
D7-D0	0x00	TBLSTRT[7:0]	Table start boundary. Bit 7 and 6 are not used or available.	R/W

Program Memory Map

The program counter in the MAX1781 is 13 bits wide, which supports an 8k-word address space. The instruction encoding allows for 13-bit program memory page size (can access up to 8192 locations). Program memory is linear and paging is not necessary. The MAX1781 has 6k words of program EEPROM as shown in *Figure 6*. See the *EEPROM Memory* section for details regarding programming the program memory.

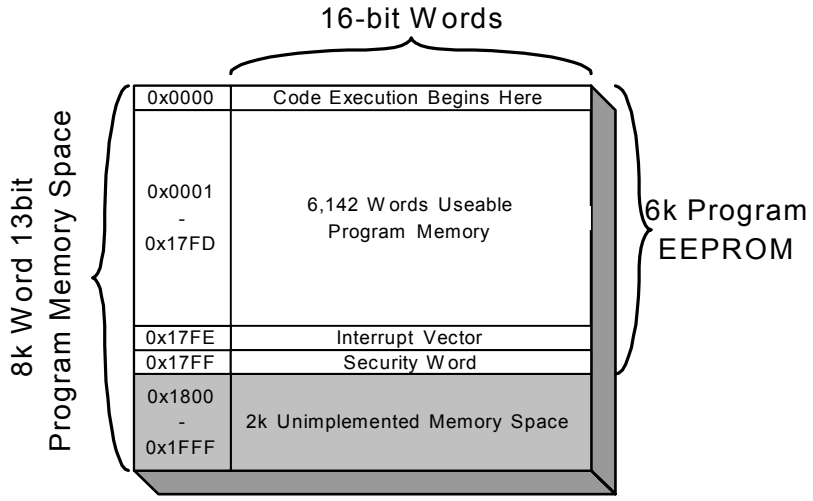


Figure 6. Program Memory Map

Data Memory Map

The MAX1781 utilizes two types of data memory: page independent and page dependent. Page-independent memory is the same across all values of the FSR. Addresses 0x00 to 0x7F are page independent. Locations 0x00–0x40 are page-independent and directly addressable. Locations 0x40–0x7F are unpaged but are only accessible with MOVFR/MOVRF instructions and indirect addressing. All the register addresses following these registers are paged (i.e., 0x80 to 0xFF). Paged registers are also only addressable with the MOVFR/MOVRF instructions or with indirect addressing. Core registers, general-purpose registers, and frequently accessed peripheral locations are located in addresses from 0x00–0x3F. Infrequently accessed peripheral registers, such as port configuration, are located in paged peripheral addresses from 0x0D0–0x0FF. Paged data memory is located in address range between 0x80–0xFF. Paged addresses utilize the FSR (see the *File Selection Register* (FSR) section) to select their banks. See

Table 2. MAX1781 Data Memory Map.

Table 2. MAX1781 Data Memory Map

	FSR															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x00 … 0x17	**24 Core and Peripheral Registers.** All Instructions can access this memory space using direct addressing Memory is unpaged (independent of FSR)															
0x18 … 0x3F	**40 RAM locations** All instructions can access this memory space using direct addressing Memory is unpaged (independent of FSR)															
0x40 … 0x7F	**64 Unpaged RAM locations** Only MOVFR, MOVRF, and indirect addressing using pointers Memory is unpaged (independent of FSR)															
0x80 … 0xBF	**80 Banked RAM** Only MOVFR/MOVRF, and indirect addressing using pointers	**80 Banked RAM** Only MOVFR/MOVRF, and indirect addressing using pointers	Unused/Invalid Memory												64 Bytes EEPROM Memory Latches	64 Bytes Data EEPROM
0xC0 … 0xCF																
0xD0 … 0xFF	**48 Banked Peripheral Registers** MOVFR/MOVRF and indirect addressing	Unused/Invalid Memory													Unused/Invalid Memory	Unused/ Invalid Memory

Table 3. Page Independent Data Memory Map

Add.	Name		D7	D6	D5	D4	D3	D2	D1	D0
0x00	W	Working Register								
0x01	STATUS	Core	INTPROC	GIE	000			C	D	Z
0x02	PCL	Program Counter Low Byte								
0x03	PCH	Program Counter High Byte								
0x04	FSR	File Select Register								
0x05	TBLSTRT	Table Start Boundary								
0x06	IS1	Interrupt Status 1	000			PE2I	DE2I	PAI	TRAPI	PCOVFI
0x07	IE1	Interrupt Enable 1	000			PE2EI	DE2EI	PAEI	TRAPEI	PCOVFEI
0x08	IS2	Interrupt Status 2	0	WDG	FGI	PFW	ADCI	ODSCI	SMBI	TI
0x09	IE2	Interrupt Enable 2	0	WDEI	FGEI	PFWEI	ADCEI	ODSCEI	SMBEI	TEI
0x0A	SMBRDR	SMBus Received Data and Manufacturer Access Address								
0x0B	SMBXDR	SMBus Transmit Data Register and Manufacturer Access Data								
0x0C	SMBCFG1	SMBus Config 1	MSMODE		BUSBUSY	ACKEN	XSTOP	XSTART	0	GF
0x0D	SMBCFG2	SMBus Config 2	MBAUD			SCLSTRETCH	CPEC	RPEC	XPEC	--
0x0E	SMBST1	SMBus Status 1	NACK	ACK	SRWB	STOP	RSTRT	START	SADRMATCH	
0x0F	SMBST2	SMBus Status 2	RXDONE	TXDONE	PERR	ANERR	MFGACCMD	TIMEOUT		MARBLOSS
0x10	AFECFG	AFE Config	THRM	BIASCANEN	BATTON	Not Used	AFESELECT			
0x11	ADCCFG	ADC Config	OVERFLOW	SIGN	ADCLIMIT	INTMODE	ADCRES			ADCSTCONV
0x12	ADCL	ADC Low Byte	ADC_D7	ADC_D6	ADC_D5	ADC_D4	ADC_D3	ADC_D2	ADC_D1	ADC_D0
0x13	ADCH	ADC High Byte	ADC_D15	ADC_D14	ADC_D13	ADC_D12	ADC_D11	ADC_D10	ADC_D9	ADC_D8
0x14	WDT	Watchdog/Timer	CLRRTCC	--	RTCCS1	RTCCS0	CLRWDT	WDTEN	WDTS1	WDTS0
0x15	ODSC	Over-discharge/Short Circuit	--			OCLO	ODINTEN	ODINT	ODHI	ODLO
0x16	FGINT	Fuel Gauge Interrupt	OVFRES		ENDIR CHANGE	ENDISOV	ENCHGOV	DIRCHANGE	DISOV	CHGOV
0x17	PAINTST	Port A Interrupt Status	--			PA4INT	PA3INT	--		
0x18 -0x3F	**40 Direct Addressable Page-Independent RAM Locations**									
0x40 -0x7F	**64 Indirect Addressable Page-Independent RAM Locations**									

Page-dependent memory is unique for each value of the FSR. The page-dependent memory is defined in the table below. The most significant 4 bits of the 12-bit address are defined by the FSR.

Table 4. Page Dependent Data Memory Map:

Add.	Name		D7	D6	D5	D4	D3	D2	D1	D0
0x080–0x0CF	80 Page-Dependent RAM Locations									
0x0D0	SMBEI	SMB Enable Interrupts	EIMFGACC	EIRXDONE	EITXDONE	EISP	EIADRMTCH	EITOUT	EIARL	EIPEC
0x0D1	SMBPSADR		Primary Slave Address							0
0x0D2	SMBSSADR		Secondary Slave SMBus Address							0
0x0D3	SMBPDR	PEC Data	Computed PEC until this point in time							
0x0D4	SMBMACR	MACC Command	Manufacturer Access Command Received Over the SMBus							
0x0D5	PAEN	Port A Enable	PAEN7	PAEN6	PAEN5	PAEN4	PAEN3	PAEN2	PAEN1	PAEN0
0x0D6	PA	Port A	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
0x0D7	PAIN	Port A Input	PAIN7	PAIN6	PAIN5	PAIN4	PAIN3	PAIN2	PAIN1	PAIN0
0x0D8	PACFG	Port A Config.	DISEN	CHGEN	0	PA4INTEN	PA3INTEN	0	ODEN	0
0x0D9	PHV	HV Port	HV7	HV6	HV5	HV4	HV3	HV2	HV1	HV0
0x0DA	FGCFG	FG Config.	FGON	FGCALON	FGMUX		FGCHGSTAT	FGLATCHNOW	FGCLRDIS	FGCLRCHG
0x0DB	FGDISL		Fuel Gauge Discharge Counter Low Byte							
0x0DC	FGDISH		Fuel Gauge Discharge Counter High Byte							
0x0DD	FGCHGL		Fuel Gauge Charge Counter Low Byte							
0x0DE	FGCHGH		Fuel Gauge Charge Counter High Byte							
0x0DF	ODSCDLY	ODSC Delay	Short Circuit Current Delay				Over Discharge Current Delay			
0x0E0	MCFG1	EEPROM Config. 1	BADDR	BCLEAR	MCLEAR	MOD1	MOD0	MSEL	OE\	CE\
0x0E1	Reserved	Factory Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
0x0E2	MADRL		Program/Data EEPROM Address Low Byte							
0x0E3	MADRH		Program/Data EEPROM Address High Byte							
0x0E4	MDATL		Program/Data EEPROM Data Low Byte							
0x0E5	MDATH		Program/Data EEPROM Data High Byte							
0x0E6 -0x0E7		Invalid Memory								
0x0E8 -0x0EB		Factory Reserved Use								
0x0EC -0x0EF		Invalid Memory								
0x0F0 -0x0F7		Trim Data	Read register and write the result back into these registers to initialize trim data during system startup							
0x0F8 -0x0FB		Trim Data	Factory Reserved							
0x0FC	ISPDBGAC	ISP and DBG Access	Reserved	Locations Read as 0			Reserved	LOCK	SDBGEN	Reserved
0x0FD	ISPDBGCFG	Writeable from SMBus MACC command only	Reserved	FREEZEALL	FREEZE	DBGEN	Reserved	ISPCPUFREEZE	PROGEXITMODE	PROGMODE
0x0FE	TMCfgSta	Writeable from SMBus MACC command only	Write 0 to these locations						PARPROG	Write 0 to this location
0x0FF	Invalid Memory									
0x180 -0x1CF	80 Page-Dependent RAM Locations									
0x1CF -0xE7F	Invalid Memory									
0xE80 -0xEBF	64 EEPROM Write Only Memory Latches. Used for programming 32 words of 16-bit program memory.									
0xEC0 -0xF7F	Invalid Memory									
0xF80 -0xFBF	64 Data EEPROM locations									
0xFC0 -0xFFF	Invalid Memory									

Memory labeled invalid may read as any value.

Interrupts

There are two interrupt enable registers, IE1 and IE2. Set the appropriate bit (bit = 1) to enable an interrupt from the associated interrupt source. Clear the bit (bit = 0) to disable interrupts from that source. Tables below detail the interrupt enable registers.

Any of the enabled interrupt events wake up the processor. Any interrupt enable bit that is cleared does not allow that event to awaken the processor.

There are two interrupt status registers, IS1 and IS2. The corresponding status bit is set when an interrupt is generated by the corresponding peripheral. Because the status bits are set regardless of the interrupt enable bits, it is good practice to clear the status bit as well as all corresponding peripheral interrupt status bits before enabling an interrupt.

If an interrupt status bit is set, the interrupt is enabled, and interrupts are enabled (GIE = 1), then the processor performs an interrupt call to 0x17FE. The current program counter and C, D, and Z of the status register are pushed into a special (and non-addressable) shadow register reserved for interrupt processing. W is not pushed/saved and must be stored and recovered manually by software. Finish interrupt processing by executing the RETFIE instruction, which restores the Program Counter and STATUS register.

IE1—Interrupt Enable 1 Register

(Read/Write)
Address 0x07

BIT	NAME	POR	DESCRIPTION
D7–D5	Unused	0	Not used and not available.
D4	PE2EI	0	Program EEPROM interrupt enable. Enables PE2I interrupt to be generated when a program EEPROM write/clear command is completed ($\overline{CE} = \overline{OE} = 1$ in MCFG1 register).
D3	DE2EI	0	Data EEPROM Interrupt Enable. Enables DE2I interrupt to be generated when a data EEPROM write/clear command is completed ($\overline{CE} = \overline{OE} = 1$ in MCFG1 register).
D2	PAEI	0	PortA Interrupt Enable. PortA interrupt (PAI) depends on the status of bits PA4INT/PA3INTEN in PACFG register and bits PA4EN/PA3EN in PAEN register (see the *Port A* section).
D1	TRAPEI	0	Software Trap Interrupt Enable. The TRAP interrupt (TRAPI) is initiated by TRAP instruction.
D0	PCOVFEI	0	PC Stack Overflow/Underflow Interrupt Enable. PC Stack overflow (PCOVFI) occurs when the 8-level stack is full and the CALL instruction is executed. Stack underflow occurs when the 8-level stack is empty and a RETLW or RETURN is executed.

IE2—Interrupt Enable 2 Register

(Read/Write)
Address 0x09

BIT	NAME	POR	DESCRIPTION
D7	Unused	0	Not used and not available.
D6	WDEI	0	Watchdog interrupt. For debugging only. Watchdog does not reset part when DBGEN bit in ISPDBGCFG register is 1.
D5	FGEI	0	Fuel-gauge interrupt.
D4	PFWEI	0	Power-fail enable interrupt. Interrupt persists as long as V_{AA} < 3.0V
D3	ADCEI	0	ADC conversion interrupt.
D2	ODSCEI	0	Over-discharge/Short-Circuit Current Interrupt Enable.
D1	SMBEI	0	SMBus Interrupt Enable. See the *SMBus* section and SMBEI register for further information.
D0	TEI	0	Timer Interrupt Enable. Also known as the RTCC interrupt. Enables TI interrupt to be generated by the internal 32kHz oscillator, which is the time base for the ADC and may be used as the time base for latching fuel-gauge data. Configure the RTCC timer interrupt in the watchdog and timer register (WDT, 0x14)

IS1—Interrupt Status 1 Register

(Read/Write)
Address 0x06

BIT	NAME	POR	DESCRIPTION
D7–D5	Unused	—	This bit is not used and not available.
D4	PE2I	0	Program EEPROM interrupt
D3	DE2I	0	Data EEPROM Interrupt.
D2	PAI	0	PortA interrupt.
D1	TRAPI	0	Software Trap Interrupt.
D0	PCOVFI	0	PC Stack Overflow/Underflow Interrupt.

IS2—Interrupt Status 2 Register

(Read/Write)
Address 0x08

BIT	NAME	POR	DESCRIPTION
D7	Unused	0	This bit is not used and not available.
D6	WDG	0	Watchdog interrupt. Valid only in debug mode.
D5	FG	0	Fuel-Gauge Interrupt. (2nd-level enable/status in FGINT register.)
D4	PFW	0	Power-Fail Enable Interrupt.
D3	ADCI	0	ADC conversion done interrupt.
D2	ODSCI	0	Over-discharge/Short-Circuit Current Interrupt. (2nd-level enable/status in ODSC register.)
D1	SMBI	0	SMBus interrupt.
D0	TI	0	Timer interrupt.

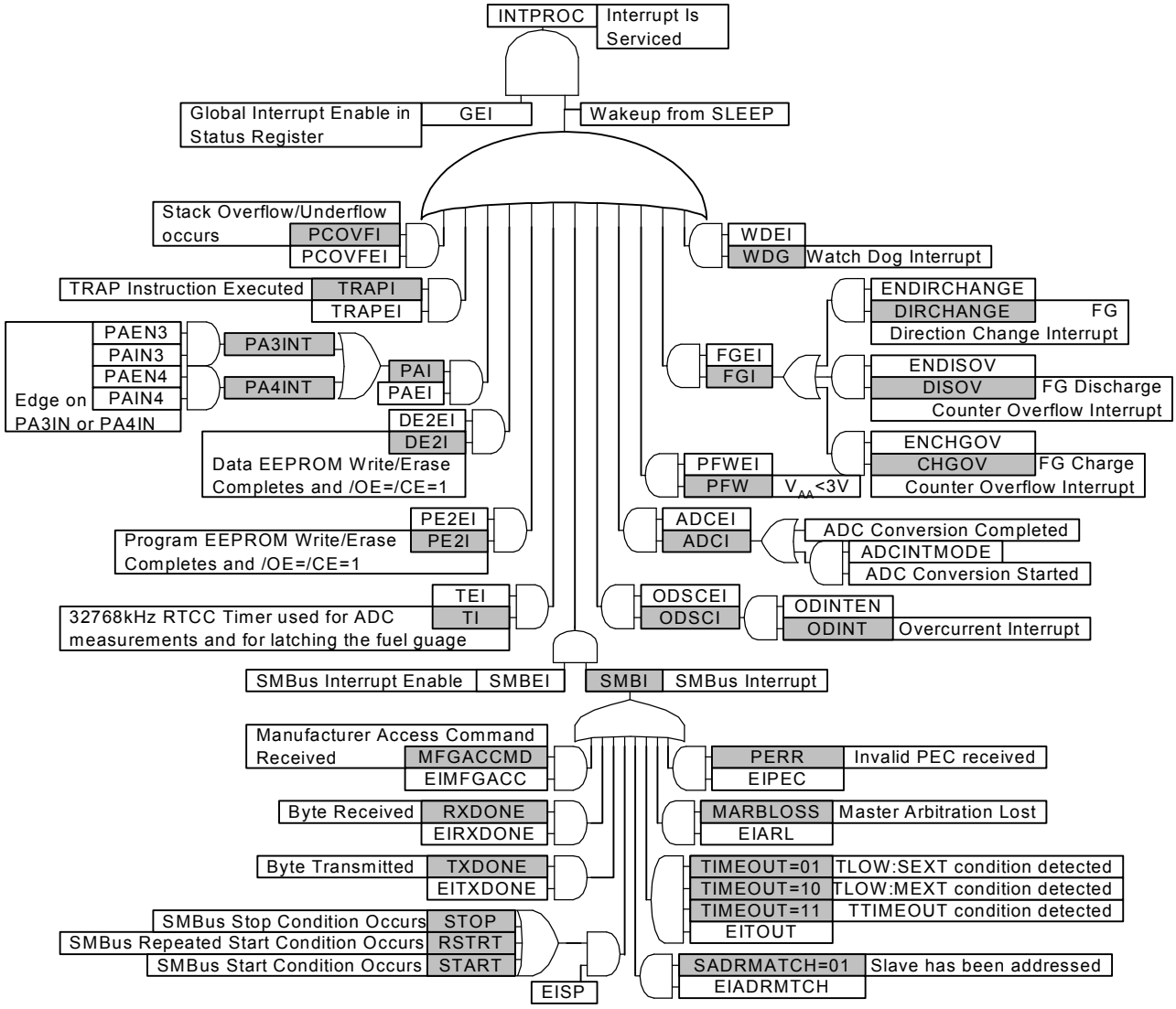


Figure 7. MAX1781 Interrupt Tree

The interrupt tree shown in *Figure 7*. relates IS1, IS2, IE1, and IE2 to the corresponding interrupt status/enable bits in the various peripherals. Before returning from the interrupt service routine, the program needs to clear all the peripheral interrupt status bits corresponding to the interrupt event serviced, in addition to the interrupt status bit in IS1 or IS2. For example, to clear the overcurrent interrupt, clear ODINT in the ODSC register (address 0x15), and then clear ODSCI in IS2. Status bits are latched on the instruction clock cycle following the interrupt condition. Writing a 1 to any status bit in IS1 or IS2 has no effect on the bit.

PFW is the only level-triggered interrupt. The interrupt is maintained as long as V_{AA} < 3.0V and the interrupt is enabled (see the *Power-Supply Sequencing* section for further details). All other interrupts are edge-triggered at the CPU and the interrupt passes after the appropriate status bits have been cleared.

Instruction Set

The MAX1781 is a RISC microcontroller that supports 16-bit instructions. Many instructions utilize both direct and indirect addressing. For direct addressing, the source or destination is at the 6-bit address location provided (operand f). For indirect addressing, the source or destination is pointed to by the data at 6-bit address location f.

The notation for indirect addressing is [f]. For example, "MOVWF [0x18]" moves W to the address pointed to by 0x18 and address 0x18 remains unchanged. Most direct addressed instructions complete execution within 1 clock cycle. Indirect addressing requires an additional instruction cycle. Because direct-addressable memory is limited to 64 bytes (0x00 to 0x3F), and 24 of these bytes (0x00 to 0x17) are already reserved for registers, the remaining 40 bytes should be used carefully for optimum program size/execution. Wise use of MOVFR and MOVRF upon entering/exiting subroutines to save/restore direct-addressable memory can result in faster execution time without consuming the direct-addressable memory.

Table 5 lists the instructions available on the MAX1781. **Table 6** lists the instructions grouped by function. See *Appendix A: Instruction Set Manual* for extended function description and example code.

Table 5. Instruction Table

MNEMONIC, OPERAND	DESCRIPTION	CYCLES	ENCODING	STATUS AFFECTED
ADDWF f, d	Add contents of W and f	1	0001,0000,0dff,ffff	C, DC, Z
ADDWF [f], d	Add contents of W and [f]	2	0001,0000,1dff,ffff	C, DC, Z
ADDLW k	Add literal with W	1	0000,1100,kkkk,kkkk	C, DC, Z
ANDLW k	AND literal with W	1	0000,1000,kkkk,kkkk	Z
ANDWF f, d	AND W with f	1	0001,0001,idff,ffff	Z
ANDWF [f], d	AND W with [f]	2	0001,0001,idff,ffff	Z
BCF f, b	Bit clear f	1	0011,00ib,bbff,ffff	—
BCF [f], b	Bit clear [f]	2	0011,00ib,bbff,ffff	—
BSF f, b	Bit set f	1	0011,01ib,bbff,ffff	—
BSF [f], b	Bit set [f]	2	0011,01ib,bbff,ffff	—
BTFSC f, b	Bit test f, skip if clear	1 (2)	0011,10ib,bbff,ffff	—
BTFSC [f], b	Bit test [f], skip if clear	2 (3)	0011,10ib,bbff,ffff	—
BTFSS f, b	Bit test f, skip if set	1 (2)	0011,11ib,bbff,ffff	—
BTFSS [f], b	Bit test [f], skip if set	2 (3)	0011,11ib,bbff,ffff	—
CALL m	Call subroutine	2	011m,mmmm,mmmm,mmmm	—
CLRF f	Clear f	1	0001,0010,i1ff,ffff	Z
CLRF [f]	Clear [f]	2	0001,0010,i1ff,ffff	Z
COMF f, d	One's complement f	1	0001,0011,idff,ffff	Z
COMF [f], d	One's complement [f]	2	0001,0011,idff,ffff	Z
DECF f, d	Decrement f	1	0001,0100,idff,ffff	Z
DECF [f], d	Decrement [f]	2	0001,0100,idff,ffff	Z
DECFSZ f, d	Decrement f, skip if zero	1 (2)	0001,0101,idff,ffff	—
DECFSZ [f], d	Decrement [f], skip if zero	2 (3)	0001,0101,idff,ffff	—
GOTO m	Unconditional branch	1	010m,mmmm,mmmm,mmmm	—
INCF f, d	Increment f	1	0001,0110,idff,ffff	Z
INCF [f], d	Increment [f]	2	0001,0110,idff,ffff	Z
INCFSZ f, d	Increment f, skip if zero	1 (2)	0001,0111,idff,ffff	—
INCFSZ [f], d	Increment [f], skip if zero	2 (3)	0001,0111,idff,ffff	—
IORLW k	Inclusive OR W with k	1	0000,1001,kkkk,kkkk	Z
IORWF f, d	Inclusive OR W with f	1	0001,1001,idff,ffff	Z
IORWF [f], d	Inclusive OR W with [f]	2	0001,1001,idff,ffff	Z
MOVFW f	Move f into W	1	0001,1000,i1ff,ffff	Z
MOVFW [f]	Move [f] into W	2	0001,1000,i1ff,ffff	Z
MOVWF f	Move W to f	1	0001,1011,i0ff,ffff	—
MOVWF [f]	Move W to [f]	2	0001,1011,i0ff,ffff	—
MOVLW k	Move literal to W	1	0000,1010,kkkk,kkkk	—
MOVFR f, r	Move f to r	2	11rr,rrrr,rrff,ffff	—
MOVRF r, f	Move r to f	2	10rr,rrrr,rrff,ffff	—
NOP	No operation	1	0000,0001,0000,0001	—
RETFIE	Return from interrupt	1	0000,0100,0000,0000	INTPROC
RETLW k	Return, place literal in W	1	0000,0101,kkkk,kkkk	—
RETURN	Return from subroutine	1	0000,0111,0000,0000	—
RLF f, d	Rotate f left through carry	1	0001,1100,idff,ffff	C
RLF [f], d	Rotate [f] left through carry	2	0001,1100,idff,ffff	C

MNEMONIC, OPERAND	DESCRIPTION	CYCLES	ENCODING	STATUS AFFECTED
RRF f, d	Rotate f right through carry	1	0001,1101,idff,ffff	C
RRF [f], d	Rotate [f] right through carry	2	0001,1101,idff,ffff	C
SKIPZ f	Skip next instruction if f is zero	1 (2)	0010,0000,i0ff,ffff	—
SKIPZ [f]	Skip next instruction if [f] is zero	2 (3)	0010,0000,i0ff,ffff	—
SLEEP	Enter sleep mode	1	0000,0010,0000,0010	---
SUBWF f, d	Subtract W from f	1	0001,1111,idff,ffff	C, DC, Z
SUBWF [f], d	Subtract W from [f]	2	0001,1111,idff,ffff	C, DC, Z
SUBLW k	Subtract literal from W	1	0000,1101,kkkk,kkkk	C, DC, Z
SWAPF f, d	Swap nibbles f	1	0001,1010,idff,ffff	—
SWAPF [f], d	Swap nibbles [f]	2	0001,1010,idff,ffff	—
TRAP	Software Trap (Interrupt)	1	0000,0000,0000,0000	---
XORLW k	Exclusive or Literal with W	1	0000,1011,kkkk,kkkk	Z
XORWF f, d	Exclusive OR W with f	1	0001,1110,idff,ffff	Z
XORWF [f], d	Exclusive OR W with [f]	2	0001,1110,idff,ffff	Z
TBLRDW f	Table lookup instruction Read into "W" the contents of table location in program memory at the address generated by concatenation of the contents of {(TBLSTRT), (f)}	2	0010,1111,01ff,ffff	—

Note:

( ) In the "CYCLES" column this indicates the number of instruction cycles when a branching instruction changes the Program Counter.

f This represents any 6-bit addressable memory location.

[**f**] Used for indirect memory access. This represents any 6-bit directly addressable memory location that points to an 8-bit addressable memory location.

r This represents any 8-bit addressable memory location.

d This indicates whether the result is stored in register **f** (if d is 1) or **W** (if d is 0)

k This is any literal 8-bit number

b This represents a number from 0 to 7 which indicates which bit is acted on, with 7 being the MSB and 0 being the LSB.

m This is a 13-bit program memory address.

Note: Instructions reading data EEPROM will require one additional instruction cycle.

Table 6. Instructions Grouped By Function

MATH OPERATIONS	
ADDWF [f], d	Add contents of W and [f]
ADDLW k	Add literal with W
SUBWF [f], d	Subtract W from [f]
SUBLW k	Subtract literal from W
DECF [f], d	Decrement [f]
DECFSZ [f], d	Decrement [f], skip if zero
INCF [f], d	Increment
INCFSZ [f], d	Increment [f], skip if zero
CLRF [f]	Clear [f]
Bit Operations	
BCF [f], b	Bit Clear [f]
BSF [f], b	Bit Set [f]
Special Purpose Instructions	
NOP	No Operation
SLEEP	Enter Sleep Mode
SWAPF [f], d	Swap nibbles [f]
TBLRDW f	Table lookup instruction.

LOGIC OPERATIONS	
ANDLW k	AND literal with W
ANDWF [f], d	AND W with [f]
IORLW k	Inclusive OR W with k
IORWF [f], d	Inclusive OR W with [f]
XORLW k	Exclusive or Literal with W
XORWF [f], d	Exclusive OR W with [f]
COMF [f], d	Complement [f]
RLF [f], d	Rotate left [f] through carry
RRF [f], d	Rotate right [f] through carry
MOVE Instructions	
MOVFW [f]	Move [f] into W
MOVWF [f]	Move W to [f]
MOVLW k	Move literal to W
MOVFR f, r	Move f to r
MOVRF r, f	Move r to f

BRANCHING	
Skip Instructions	
BTFSC [f], b	Bit test [f,] skip if clear
BTFSS [f], b	Bit test [f], skip if set
DECFSZ [f], d	Decrement [f], skip if zero
SKIPZ f	Skip next instruction if Register is zero
Branch/Return Instructions	
GOTO m	Unconditional Branch
CALL m	Call subroutine
RETLW k	Return, place Literal in W
RETURN	Return from subroutine
TRAP	Software Trap (Interrupt)
RETFIE	Return from interrupt

Normal Modes of Operation

The MAX1781 has four normal modes of operation:

1) Shutdown (PCKP = floating, and BATTON = 0, $V_{AA} = V_{DD} < 2.6$)
2) Sleep (instruction oscillator off, CPU waiting for interrupt)
3) Master Clear ($\overline{MCLR}$ = low)
4) Program Execution

Shutdown

In shutdown mode, the MAX1781 consumes practically no current (<1nA). This mode is useful for reducing the self-discharge of battery packs in transit to customers or being stored for long periods.

Entering Shutdown

To enter shutdown mode, the PCKP pin should be unpowered and the BATTON (bit 6 of the AFECFG register) bit cleared. The discharge FET (controlled by DIS) must be turned off using the ODSC register causing PCKP to provide power to the MAX1781. The PCKP node discharges and powers the MAX1781 until its voltage falls below 4V, after which the system resets (when $V_{AA} = V_{DD}$ falls below 2.7V). In this state, the current from the BATT, HV[7:0], DIS, and CHG pins is typically below 1nA.

Recovering From Shutdown

To recover from shutdown mode, connect PCKP to a charger with at least 4V of source voltage. This voltage on PCKP starts up the PCKP linear regulator and brings the MAX1781 out of reset. If the pack should operate after this initial startup, then the BATTON bit can be set to 1 by software. See the Linear Regulators for System Power and Startup section for more information.

Sleep

In sleep mode, instruction execution is suspended. The internal instruction oscillator can be on or off in sleep mode depending on whether or not the part is in an EE write cycle, or acting as an SMBus master. The internal instruction oscillator turns on whenever sleep mode is exited.

Entering Sleep Mode

Executing a SLEEP instruction enters sleep mode.

Wake-Up from Sleep Mode

The MAX1781 can wake up from SLEEP through one of the following events:

1) An external reset input on the $\overline{MCLR}$ pin.
2) A watchdog timeout.
3) An interrupt from any enabled source wakes up the MAX1781, regardless of the state of the GIE (global interrupt enable) bit in the STATUS Register.

Master Clear

Placing a logic-level LOW on the $\overline{MCLR}$ pin causes the MAX1781 to reset all its internal logic circuitry. While $\overline{MCLR}$ is low, all program execution is halted. Additionally, PA[7:0] and HV[7:0] pins are high impedance, and the CHG and DIS pin drivers rise to PCKP and BATT, respectively, opening both pack-protection MOSFETs.

Releasing the logic LOW on the $\overline{MCLR}$ pin, allowing it to rise to V_{DD}, allows the processor core to begin program execution.

Program Execution

Whenever instructions are executing, the MAX1781 is in Program Execution mode.

Factory Calibration Initialization

The MAX1781 is factory calibrated to produce an accurate ADC, fuel gauge, instruction oscillator, and 32kHz oscillator. Memory registers 0xF0-0xFB contain this calibration data. Registers 0xF0-0xF7 must be initialized at startup to provide accurate analog peripherals. Read address 0xF0-0xF7 yields the nonvolatile calibration data, and write the data back to 0xF0-0xF7 to initialize the volatile calibration latches. This memory may only be written to once after reset.

EEPROM Memory

General Description

EEPROM memory in the MAX1781 consists of program and data memory. The embedded program memory is organized as 6k x 16-bit words and the embedded EEPROM data memory is configured as 64 bytes. Writing and clearing are done 1 byte at a time for data EEPROM. A block clear is available to clear the entire program memory or data memory. Both blocks provide a minimum data retention of 10 years and minimum CLEAR-WRITE endurance of 100,000 cycles. A regulated on-chip charge pump is included to make writing/clearing the EEPROM possible without the need for an external 12V power supply.

The term “page” is 2k x 16 bits of program memory. The term “sector” is 32 x 16 bits of program memory block.

EEPROM Memory Commands

Clearing EEPROM resets bits to 0. EEPROM must first be cleared before writing. The MAX1781 EEPROM supports the following functions: clear all program EEPROM, clear all data EEPROM, clear program EEPROM page (2k-word), clear/write program EEPROM sector (32 words), clear/write data EEPROM byte, and read program EEPROM word. When executing a program memory EEPROM command program execution is halted. When executing a data EEPROM command program execution continues. All EEPROM write/clear commands take integer multiples of the clear/write time (t_{CW} from the EC table).

Clear All Program EEPROM and Unlock —MCFG1 = 0x4E, or MACC Command ISPERPM

This command can only be used in service mode. This command may be executed over the SMBus by initiating an ISPERPM MACC command (see the *Access and Security* section). In parallel programming mode this command may also be initiated by writing 0x4E to MCFG1. This command may not be executed by firmware. This also sets LOCK = 0 and SDBGEN = 1 in the ISPDBGAC register and unlocks the microcontroller’s EEPROM, allowing program and data EEPROM and RAM to be read and written from SMBus or in parallel

programming mode. Approximately 4 t_{CW} periods after issuing ISPERPM, $\overline{CE}$ (in the MCFG1 register) is set to 1, indicating that program EEPROM has been completely cleared. After the clear cycle completes program execution will continue at the instruction following the MCFG1 write unless PROGEXITMODE = 1 (see ISPDBGCFG register.)

To insure that memory is reliably cleared, first clear the 64 EEPROM latches at data memory addresses 0xE80 to 0xEBF. If the MAX1781 is locked it may be necessary to follow the following sequence:

1) Execute the ISPERPM MACC command.
2) Clear the 64 EERPOM latches (0xE80 to 0xEBF). This may be achieved through an SMBus block memory write when the part is in service mode.
3) Execute the ISPERPM MACC command again.

Clear All Data EEPROM
—MCFG1 = 0x2A

To clear the data EEPROM write 0x2A to MCFG1 (address 0xE0). One t_{CW} period later $\overline{CE}$ (in the MCFG1 register) is set to 1, indicating that data EEPROM has been completely cleared. When $\overline{CE}$ becomes 1, DE2 in the interrupt status byte (IS1) is set, and an interrupt is generated if enabled (DE2EI = GIE = 1).

Clear Program EEPROM Page
—MCFG1 = 0x2E

Program memory is organized as three 2k-word pages. To clear a page, write the page address to MADRH (0xE3) and MADRL (0xE2). *To insure that memory is reliably cleared, first clear the 64 EEPROM latches at data memory addresses 0xE80 to 0xEBF*. Bits A10 through A0 of the address are ignored. Initiate the clear cycle by writing 0x2E to MCFG1 (0xE0). Wait t_{CW} for the EEPROM page to be cleared. When memory is completely cleared $\overline{CE}$ (in the MCFG1 register) becomes 1, PE2 in the interrupt status byte (IS1) is set, and an interrupt is generated if the interrupt is enabled (PE2EI = GIE = 1). After the clear cycle completes program execution will continue at the instruction following the MCFG1 write unless PROGEXITMODE = 1 (see ISPDBGCFG register).

Clear Program EEPROM Sector
—MCFG1 = 0x0E

To clear a 32-word program EEPROM sector write the sector address to MADRH (0xE3) and MADRL (0xE2). Bits A4 through A0 of the address are ignored. *To insure that memory is reliably cleared, first clear the 64 EEPROM latches at data memory addresses 0xE80 to 0xEBF*. Initiate the clear cycle by writing 0x0E to MCFG1 (0xE0). Wait t_{CW} for the EEPROM sector to be cleared. When memory is completely cleared $\overline{CE}$ (in the MCFG1 register) becomes 1, PE2 in the interrupt status byte is set, and an interrupt is generated if enabled (PE2EI = GIE = 1). After the clear cycle completes program execution will continue at the instruction following the MCFG1 write unless PROGEXITMODE = 1 (see ISPDBGCFG register).

Clear Data EEPROM Byte
—MCFG1 = 0x0A

To clear a data EEPROM byte, write the address to MADRH (0xE3) and MADRL (0xE2). Initiate the clear cycle by writing 0x0A to MCFG1 (0xE0). Wait t_{CW} for the EEPROM byte to be cleared. When memory is completely

cleared $\overline{CE}$ (in the MCFG1 register) becomes 1, DE2 in the interrupt status byte is set, and an interrupt is generated if enabled (DE2EI = GIE = 1).

Write Program EEPROM Sector
—MCFG1 = 0x16

Before writing an EEPROM sector ensure that the sector is first cleared (see *Clear Program EEPROM Sector —MCFG1* = **0x0E**.) To write a 32-word program EEPROM sector, write the sector address to MADRH (0xE3) and MADRL (0xE2). Bits A4 through A0 of the address are ignored. Write the corresponding data to the 64 EEPROM latches at data memory addresses 0xE80 to 0xEBF. 0xE80 and 0xE81 form the lowest address in the sector where 0xE80 is the LSByte and 0xE81 is the MSByte. Initiate the write cycle by writing 0x16 to MCFG1 (0xE0). Wait t_{CW} for the EEPROM sector to be written. When memory is completely written $\overline{CE}$ (in the MCFG1 register) becomes 1, PE2 in the interrupt status byte is set, and an interrupt is generated if enabled (PE2EI = GIE = 1). After the write cycle completes program execution will continue at the instruction following the MCFG1 write unless PROGEXITMODE = 1 (see ISPDBGCFG register).

Write Data EEPROM Byte
—MCFG1 = 0x12

Before writing a data EEPROM byte ensure that the byte is first cleared (see *Clear Data EEPROM Byte —MCFG1* = **0x0A**.) To write a single EEPROM byte, write the data memory address to MADRH (0xE3) and MADRL (0xE2). Write the FSR (0x0F for all data EEPROM) to the lower nibble of MADRH. Write the lower 8 bits of data memory address to MADRL. Write the corresponding byte to MDATL (0xE4). Initiate the write cycle by writing 0x12 to MCFG1 (0xE0). Wait t_{CW} for the EEPROM byte to be written. When the byte is completely written $\overline{CE}$ (in the MCFG1 register) becomes 1, and DE2 in the interrupt status byte is set, and an interrupt is generated if enabled (DE2EI = GIE = 1).

Read Program EEPROM Word
—MCFG1 = 0x04 or TBLRDW Instruction

There are two ways to read a program memory word, using either the TBLRDW instruction (see *TBLSTRT and Accessing Data Tables in Program Memory*) or by accessing MCFG1. The TBLRDW instruction can be used for reading the program memory using firmware. This is useful when tables of data need to be accessed. To read a single EEPROM word from the SMBus using MACC commands, or in parallel programming mode, write the program memory address to MADRH (0xE3) and MADRL (0xE2). Initiate the read cycle by writing 0x04 to MCFG1 (0xE0). This command can only be initiated with an SMBus MACC command or through parallel programming mode. The word is immediately available at MDATL (0x0E4) and MDATH (0x0E5). PE2 in IS1 is set after this operation.

Read Data EEPROM Byte
To read a data memory EEPROM byte, simply read the address corresponding to the byte. Data memory cannot be read through MCFG1, MDATH, and MDATL.

Write ISPDBGAC EEPROM Byte
—MCFG1 = 0x12

The MAX1781 may be secured by using the ISPDBGAC EEPROM (see *Access and Security.*) To write ISPDBGAC, write the data memory address, 0x00FC, to MADRH (0xE3) and MADRL (0xE2). Write the

corresponding byte to MDATL (0xE4). Initiate the write cycle by writing 0x12 to MCFG1 (0xE0). Wait t_{CW} for the EEPROM byte to be written. When the byte is completely written $\overline{CE}$ (in the MCFG1 register) becomes 1, and DE2 in the interrupt status byte is set, and an interrupt is generated if enabled (DE2EI = GIE = 1).

Program Memory Protection

The MAX1781 has provisions for protecting the program memory against accidental clears and writes. The last location in program memory (0x17FF) is reserved as the security word, which sets the number of program memory sectors that are protected. A value of 0x0000 corresponds to no program memory sectors protected. Values above 0x00FF are interpreted as 0x00FF. All program memory between 0x0000 and [contents of 0x17FF]*32 are protected. For example, to protect program memory between 0x0000 and 0x1100 write 0x88 to program memory location 0x17FF. Sector 191, the last sector, cannot be protected. If a write is attempted on a protected address BADDR (in MCFG1) becomes 1 six clock cycles after /CE\ goes low.

If the LOCK bit is set, Program/Data EEPROM cannot be read or written from the SMBus interface, or through parallel programming. Program and Data memory can still be written by firmware using MCFG1, and Program memory is readable by the CPU using the TBLRDW instruction (but not through the use of MCFG1). After initiating a Program EEPROM sector write, the CPU is frozen until the write is complete. The instruction following the write to MCFG1 will be executed after the sector write is complete unless PROGEXITMODE = 1 (see ISPDBGCFG register).

EEPROM Memory Registers

MADRL—EEPROM Address Pointer Register

(Read/Write)
Addresses 0xE2

NAME	TYPE	DESCRIPTION
MADR[7:0]	R/W	MADRL is the lower 8 bits of the address of program or data memory used for a write, clear, or read command.

MADRH—EEPROM Address Pointer Register

(Read/Write)
Addresses 0xE3

NAME	TYPE	DESCRIPTION
MADR[15:8]	R/W	MADRH is the upper 8 bits of the address of program or data memory used for a write, clear, or read command.

MDATL—EEPROM Read/Write Data Register

(Read/Write)
Addresses 0xE4

NAME	TYPE	DESCRIPTION
MDAT[7:0]	R/W	To write data memory, set MDATH to the data intended to be written. When reading program memory MDATL is the lower byte of the data read.

MDATH—EEPROM Read/Write Data Register

(Read/Write)
Addresses 0xE5

NAME	TYPE	DESCRIPTION
MDAT[15:8]	R/W	When reading program memory MDATH is the upper byte of the data read.

MCFG1—EEPROM Configuration Register

(Read/Write)
Address 0xE0

BIT	NAME	POR	DESCRIPTION
D7	BADDR	0	Bad Address. This bit is set to indicate an attempt to write to an invalid or protected data or program memory location. This bit is read only.
D6	BCLEAR	0	Block clear mode enable. Set to 1 to clear all program EEPROM. If clearing all program EEPROM, the LOCK bit in ISPDBGAC is cleared, SDBGEN in ISPDBGAC is set, and the IC becomes unlocked. Block clear can only be set in service mode. Its function is the same as the ISPERPM MACC command.
D5	MCLEAR	0	Mass clear mode enable. Set to 1 for 2k program EEPROM page clear. Set to 0 for all other operations.
D4	MOD1	0	Mode selection bit 1. Set to 0 for an EEPROM read or clear cycle. Set to 1 for an EEPROM write cycle.
D3	MOD0	0	Mode selection bit 0. Set to 0 for an EEPROM read or write cycle. Set to 1 for an EEPROM clear cycle.
D2	MSEL	0	Memory block selection. Set MSEL to 1 to access program memory. Set MSEL to 0 to access data memory.
D1	$\overline{OE}$	1	Set to 0 to initiate an EEPROM read cycle. When action is complete this bit is set to 1.
D0	$\overline{CE}$	1	Set to 0 to initiate an EEPROM read, clear, or write cycle. When action is complete this bit is set to 1 and an interrupt is asserted if enabled.

Analog Peripherals

6MHz Instruction Oscillator

The MAX1781 internal instruction oscillator does not require an external crystal for operation. It is factory trimmed to 6MHz. In sleep mode, the internal instruction oscillator shuts off to save power, unless it is required to maintain SMBus master communication or to execute an EEPROM write/clear. The internal oscillator is guaranteed to start up within 1µs, minimizing interrupt latency.

32kHz Oscillator

The MAX1781 includes a trimmed 32,768Hz oscillator. The 32kHz time base is used by the following peripherals: RTCC, watchdog, SMBus timeouts, the data-acquisition unit, and the fuel gauge. The internal 32kHz time base has a typical accuracy better than +/-0.3%, and tracks internal temperature coefficients shared with the ADC and FG peripherals to minimize measurement error. Note that a precision offset-cancellation scheme is used in the oscillator that guarantees accurate timing every 12th period. Successive periods have substantial jitter, but every 12th edge of this clock is very stable over temperature and supply.

Linear Regulators for System Power and Startup

The MAX1781 contains two linear regulators: one connected to BATT and one connected to PCKP. The BATTON bit in the AFECFG (0x10) register selects which linear regulator is turned on.

BATTON's power-on reset state is 0, which enables the PCKP regulator at startup. Since the protection MOSFETs (CHG and DIS) are off and the batt linear regulator, after reset, there is no path for current to flow from the battery. In this state the part enters a 0µA shutdown state until PCKP is biased by connecting an external battery charger. This prevents the MAX1781 from consuming current from the battery during shipping. See *Figure 8. Dual Linear Regulators*

To get the part out of shutdown, the battery pack must be connected to an external power source. Once connected to an external power source, PCKP starts up the part through the PCKP linear regulator. Once the part is started, CPU program control can set BATTON = 1, and the part continues to operate even after the external supply is removed.

To enter shutdown, the CPU must set BATTON = 0 and open the protection MOSFETs. When this is done, only PCKP can supply the chip's 3.4V supply. If the pack is disconnected from an external supply, PCKP is unable to power the chip's 3.4V supply, and this supply slowly collapses. When the V_{AA} supply drops under the POR threshold (2.7V), the chip's outputs are guaranteed to be in a high-impedance state to avoid other possible leakage paths. See Table 7 for the BATTON truth table.

The low-dropout linear regulators regulate a 4V to 28V DC input voltage at BATT or PCKP, down to 3.4V at V_{AA}. V_{AA} supplies the MAX1781 internal circuitry, and can deliver up to 10mA total load. For proper operation connect V_{AA} and V_{DD} together. Bypass V_{AA} to AGND with a 0.47µF capacitor close to the IC. Bypass V_{DD} to GND with another 0.47µF capacitor close to the IC.

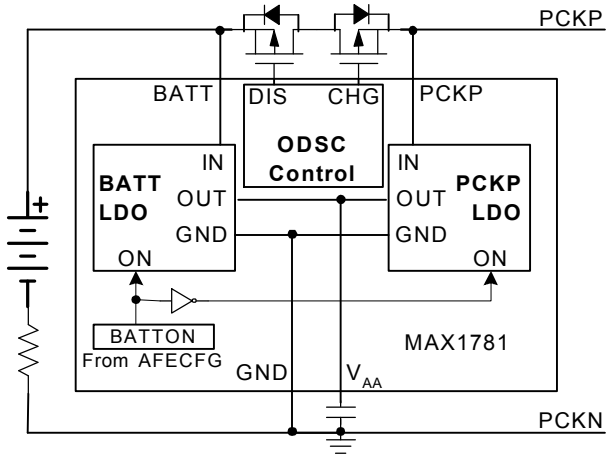


Figure 8. Dual Linear Regulators

Note: In order to guarantee proper startup operation, PCKP must have sufficient dV/dt during initial charger insertion. Often a filter is placed in series with PCKP, which can cause a race condition between the V_{AA} output and the MCLR output, interrupting the normal POR function of the MAX1781. Maxim recommends the following component values to prevent this race condition: C_{MCLR}=0.01, C_{AA}=2.2uF, R_{PCKP}<200Ohms.

The following table summarizes the operation of BATTON:

Table 7. BATTON functionality

V_{AA}>2.7V	$\overline{MCLR}$	BATTON	V_{AA} POWER SOURCE
0	X	X	PCKP Linear Reg
1	X	0	PCKP Linear Reg
1	1	1	BATT Linear Reg

Precision Bandgap Reference

The precision bandgap reference provides 1.232V to the analog peripherals. It is used internally and not brought out to a pin. The MAX1781 powers up with this reference off. When the fuel gauge, ADC, or EEPROM write cycle is started, this reference is automatically turned on. The precision reference draws 12µA of supply current when enabled. Wait at least 250µs to allow for the reference to stabilize before relying on the accuracy of an ADC conversion. This is necessary because the reference takes longer than the ADC to start up. To ensure that a fully

accurate ADC measurement is made if the reference is not already forced on by the FG or EE write cycle, the conversion needs to be restarted, or the FG needs to be enabled 250µs before the measurement is started. In typical applications the fuel gauge is continuously on.

Power-Supply Sequencing

The MAX1781 has separate analog and digital supplies. The digital circuitry begins operating with V_{DD} voltages as low as 2.7V, whereas the analog circuitry becomes available at 3.0V. The MAX1781 CPU can begin executing program code before its analog peripherals have powered up and reach full accuracy.

The MAX1781 uses two internal control signals, POR for the digital circuitry, and PFW (in IS1) for the analog circuitry, to determine if the V_{AA} supply has reached a voltage level high enough for both the digital and analog circuitry to operate. Figure 9. *Supply Voltage Sequencing*

illustrates the MAX1781 power-supply sequencing. Upon power-up firmware should test PFW. Until PFW is 0 the analog peripherals should not be used.

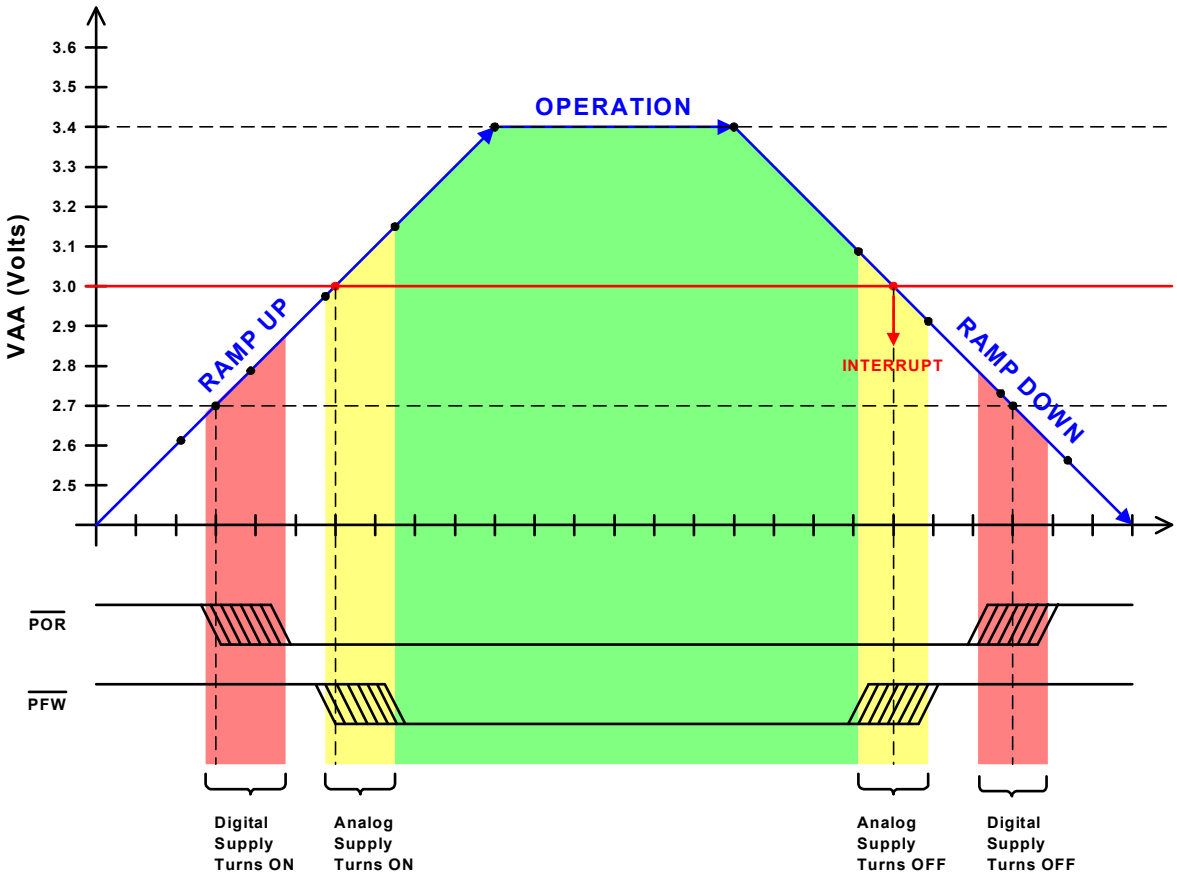


Figure 9. Supply Voltage Sequencing

An interrupt is generated when V_{AA} falls below 3.0V. Please see *Interrupts* section for a detailed description of this interrupt. Immediately after power-up, the interrupt can be used to indicate that both the digital and analog supplies have reached a reliable voltage level for operation. PFW interrupts that occur after the initial power-up sequence indicate that V_{AA} has either momentarily fallen below 3.0V, or is powering off. Table 8 summarizes the minimum supply requirements of the peripherals in the MAX1781.

Table 8. Supply Voltage Requirements for MAX1781 Peripherals

BLOCK	MIN SUPPLY	DESCRIPTION
EE Program/Clear	3V	EE programming or clear cycles. If V_{AA} below the 3V PFW threshold in a programming cycle any data programmed is unreliable.
ADC/AFE	3V	All ADC measurements; block powers down when V_{AA} < 3V. If V_{AA} falls below the 3V PFW threshold during an ADC conversion the conversion is halted and an ADC interrupt will not occur. Any data in ADCL and ADCH is inaccurate.
FG	3V	Fuel gauge; block powers down when V_{AA} < 3V. When V_{AA} falls below the 3V PFW threshold the fuel gauge stops and fuel gauge counter values are preserved. When V_{AA} returns above 3V there is typically a 16ms delay before the fuel gauge begins again.
REF	3V	Precision reference.
ODSC	2.7	Overcurrent detection and protection MOSFET drivers
LIN	2.7	Linear regulator, supply supervisor
IOSC	2.7	Instruction oscillator
32kHz	2.7	32kHz clock
CPU/Logic/EE read/SMBus	2.7	Logic and EEPROM reads

Mixed Signal Peripherals

Fuel-Gauge Unit

General Description

The fuel gauge measures the cumulative charge into (charging) and out of (discharging) the system battery pack and stores a count in one of two internal, independent charge and discharge counters. The unit also informs the host of changes in the direction of current flow. Communication with the fuel gauge occurs through memory-mapped registers that allow access to charge/discharge counters and internal registers.

Features

- True Coulomb Counting, Integrating Fuel Gauge
- Separate 16-bit Charge and Discharge Counters
- Counter Overflow and Current Direction Change Interrupts
- Three Counter Latching Sources
- Automatic Cancellation of Input Offset Voltage Without Calibration

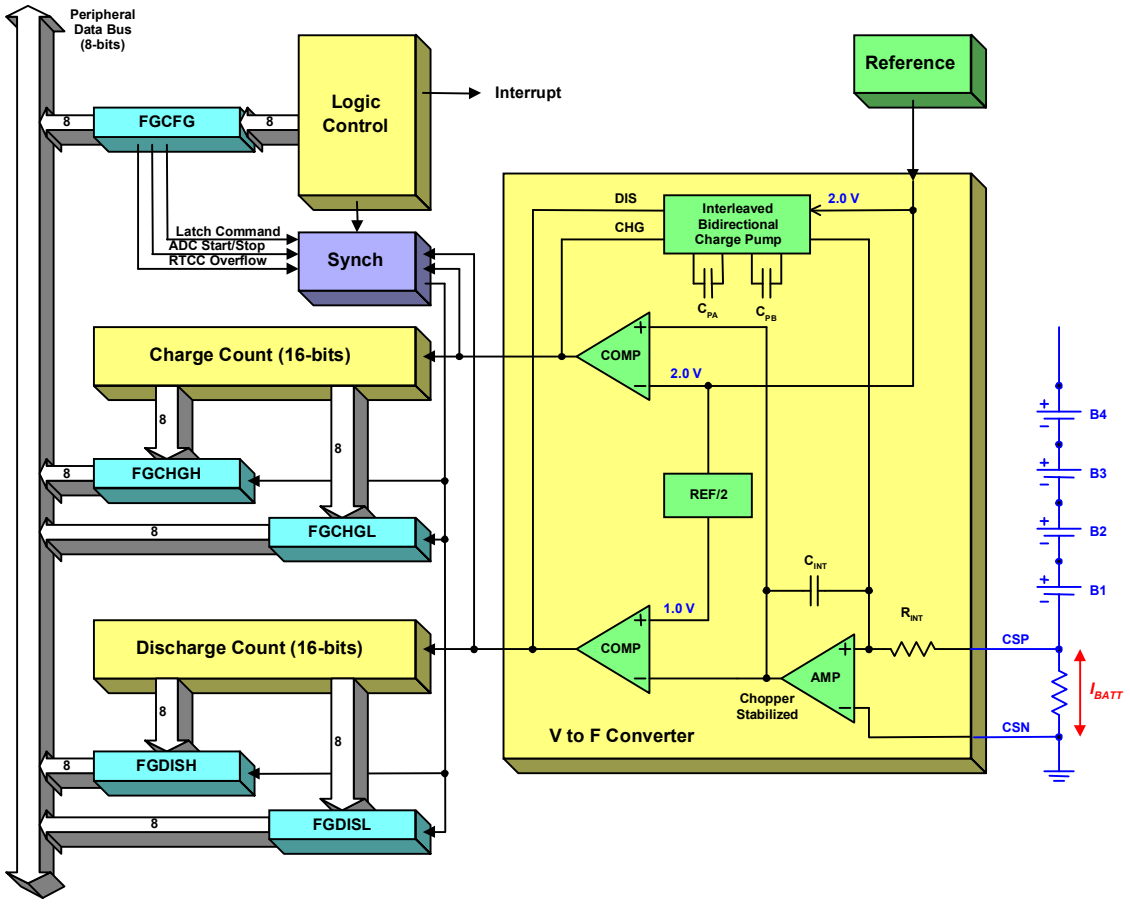


Figure 10. Fuel-Gauge Block Diagram

Automatic Cancellation of Input Offset Voltage

The MAX1781 fuel gauge uses a chopper-stabilized amplifier to couple the voltage across the current-sense resistor to the voltage-to-frequency converter. This voltage can be as high as ±125mV at heavy load currents or only a few microvolts at light loads. Therefore, the amount of input offset voltage determines the minimum current sensitivity of the fuel gauge. The chopper-stabilized amplifier continually samples and corrects for its offset. This cancellation occurs in conjunction with the fuel gauge's Coulomb counting circuitry. While the circuit is performing fuel-gauge measurements, it is continually canceling the internal input offset voltage. With careful attention to PC board layout, input offsets of less than 1µV can be expected without software calibration.

Coulomb Counting

The fuel gauge's Coulomb-counting circuit monitors the differential voltage present at the CSP and CSN pins. In a typical application, connect CSP and CSN across a sense resistor in series with the battery pack cells. The fuel gauge converts this voltage to a frequency proportional to the battery current, and increments either the charge counter or the discharge counter.

Charge and Discharge Counters

Figure 10 shows the functional diagram of the fuel gauge. Each voltage to frequency output increments (but never decrements) one of two independent 16-bit counters: charge count for charging currents, and discharge count for discharging currents. By independently counting the charge and discharge currents, the MAX1781 accommodates algorithms to account for battery-pack energy-conversion efficiency. The 16-bit charge and discharge count latch registers are each divided into 2 bytes: FGCHGL, FGCHGH, FGDISL, and FGDISH. See the *Fuel-Gauge Register* section for details of the registers.

Charge and discharge counters reset to zero whenever a reset occurs, when FGCLRCHG or FGCLRDIS (in FGCFG) is set, or when an overflow occurs. Do not use FGCLRCHG and FGCLRDIS bits (in FGCFG) to clear the fuel gauge counters in the fuel-gauging algorithm, since counts can be lost during the clear operation. A charge/discharge interrupt may be generated depending on the status of ENDISOV and ENCHGOV bits in the FGINT register and FGEI in the interrupt enable 2 (IE2, 0x09) register.

An analog power failure (PFW interrupt) powers down the fuel gauge, but does not reset fuel-gauge logic. The fuel gauge stops counting and continues when power returns. There is no need to restart the fuel gauge.

Writing a one to the FGLATCHNOW bit in the fuel gauge's FGCFG register latches the instantaneous counts for both the charge and discharge counters without clearing the counters. Read FGCHGL and FGCHGH to obtain the charge count and FGDISL and FGDISH to obtain the discharge count.

Fuel-Gauge Gain

The gain factor is the ratio of the fuel-gauge count frequency to the current-sense voltage. The ELECTRICAL CHARACTERISTICS table specifies the maximum v-to-f converter frequency and the full-scale sense-resistor voltage. With these two values the gain factor ($GAIN_{FG}$) can be calculated as follows:

$$GAIN_{FG} = \frac{50KHz}{125mV} = 400kHz/V$$

Fuel-Gauge Current

To measure average battery current using the fuel gauge, calculate the differences in counter values at a regular interval using the RTCC (TEI) interrupt. The average current is the product of the number of counts and the resolution in A/count. Calculate the resolution using the following formula (calculated for 10mΩ and 1s integration time):

$$I_{LSB} = \frac{Amperes}{Count} = \frac{V_{FS}}{F_{FS} \cdot R_{SENSE} \cdot T_{INT}} = \frac{0.125V}{50000 \cdot 0.01 \cdot 1\text{sec}} = 250uA/Count \text{ , for } T_{INT}$$

where F_{FS} is the full-scale charge frequency (see *ELECTRICAL CHARACTERISTICS*), V_{FS} is the fuel gauge's full-scale voltage (see *ELECTRICAL CHARACTERISTICS*), R_{SENSE} is the current-sense resistance, and T_{INT} is the integration period, typically based on the RTCC (TEI) interrupt.

$$I_{BATT} = I_{LSB} \cdot \Delta Count$$

A very high discharge current or long integration time can cause the $\Delta Count$ to exceed 65,536 counts. Under these circumstances, the average current could give faulty readings unless fuel-gauge overflow interrupt information is considered. For the above values this occurs at 16.384A (65536 counts x 250µA/count).

Fuel-Gauge Resolution

The amount of energy necessary to cause an overflow interrupt on either the charge or discharge counters is given by the following equation.

$$Q_{LSB} = \frac{mA \cdot hr}{OverflowInterrupt} = \frac{V_{FS}}{F_{FS} \cdot R_{SENSE}} \cdot \frac{1hr}{3600s} \cdot \frac{65536}{(OVFRES + 1)} = 4.5511mA \cdot hr / Overflow$$

If R_{Sense} is equal to 0.010Ω, and OVFRES = 0 the fuel gauge charge or discharge counters trigger an interrupt whenever 4.5511mAh of energy have flowed into or out of the battery. Set OVFRES[1:0] (in FGINT) to adjust the frequency of the interrupts.

Current Direction-Change Detection Function

The fuel gauge's direction-change detection function informs the host whenever the current flow changes direction. The FGCFG register's FGCHGSTAT bit is set to 1 when the voltage across CSP and CSN is positive. When the voltage from CSP to CSN is negative (discharge) this status bit is set to 0. The FGCHGSTAT bit is read only. The FGINT register has a dual edge-triggered input for the FGCHGSTAT bit called DIRCHANGE. DIRCHANGE sets anytime there is a change in current-flow direction on the sense resistor. This is useful for interrupting the CPU for routines in which the host must be informed immediately of a change in current-flow direction. To enable this interrupt set the ENDIRCHANGE bit in the FGINT register to a 1. DIRCHANGE has about 6 LSB of debouncing intrinsic to the fuel gauge's design.

Counter Latching Source and Arbiter

The FGMUX bits in the FGCFG register select one of three sources for latching the fuel gauge's charge and discharge counter values.

Direct CPU Control

To latch the fuel-gauge counters by direct CPU control, set the FGMUX bits D4= D5=0 in the FGCFG register, and then write a 1 to the FGLATCHNOW bit D2 in the FGCFG register. This action takes a snapshot of the instantaneous counter values and stores them in the FGDISH, FGDISL, FGCHGH, and FGCHGL registers. The charge and discharge counters are not affected by the latching operation.

ADC Conversion Start and Stop

Set the FGMUX bits D4=1 and D5=0 to latch the fuel-gauge charge and discharge counter values at the beginning and end of each ADC conversion. This method of latching the fuel-gauge counters allows the simultaneous measurement of cell voltage and current, and can be used to determine cell impedance. See *Analog-To-Digital Converter (ADC)*.

RTCC (TEI) Interrupt

To latch the fuel-gauge charge and discharge counter values at each RTCC interrupt set the FGMUX bits D4=0 and D5=1. RTCC is a programmable counter that is clocked by the 32,768Hz time base. This method of latching the fuel-gauge counters allows for measuring average current with great accuracy.

Arbitration Logic

The charge and discharge counters increment asynchronously with respect to the MAX1781 CPU instruction execution. Also, each of the three counter-latching sources can occur asynchronously with respect to charge and discharge counter updates. To ensure that erroneous counter values are not latched, the fuel gauge employs special arbitration logic. Arbitration does not occur while switching modes; therefore, it is prudent to only trust data latched by the most recently selected mode. Data read from the latches over the peripheral address bus when FGMUX[1:0] = {01, 10}, may not be reliable if the latch is strobed at the same time that the CPU is reading latch data. Use one of the following methods to ensure that the fuel gauge is latched correctly:

- Read and save/process fuel-gauge data only during the TEI and ADCI interrupts. Any references to the fuel gauge outside of the interrupt service routine should use a saved copy of the new as well as the old fuel-gauge measurements. FGMUX can be changed either inside or outside the interrupt service routine.

- Read and save/process fuel-gauge data at any time, either inside of the interrupt service routine or in the main program loop. However, FGMUX must only be altered while inside of the TEI/ADCI interrupt service routine, after latching fuel-gauge data. Any need within the main loop to change FGMUX must do so by handing a flag to the service routine and waiting for a TEI or ADCI interrupt to occur and process the flag.

- Read and save/process fuel-gauge data at any time, either inside of the interrupt service routine or in the main program loop. FGMUX can be altered outside of the interrupt service routine if it can be guaranteed that it is not modified 100ns before a TEI or ADCI interrupt is to occur.

FG Interrupts

The fuel gauge generates three different interrupts: direction change (DIRCHANGE), discharge counter overflow (DISOV), and charge counter overflow (CHGOV). The status and enable bits are contained in FGINT (0x16). The values in these registers all combine to flag the fuel gauge peripheral's interrupt (FGI) bit in interrupt status register 2 (IS2).

Service the FG interrupts as follows:

- The interrupt service routine detects that the FG interrupt flag is set in the peripheral interrupt register, and the interrupt service routine jumps to the FG interrupt handler.
- The FG interrupt handler reads the content in the FGINT status bits to determine which of the three types of interrupts occurred (DIRCHANGE, DISOV, or CHGOV).
- When the FG interrupt handler has serviced one of the pending FG interrupts, it clears the corresponding bit in the FGINT register by writing a 0 to it before leaving the interrupt handler. When all pending FG interrupts have been serviced, software clears the status bit in the peripheral interrupt register.

Note: When an interrupt is disabled, the status bits in the FGINT register are still active and report correct status. Enabling an interrupt in the FGINT register whose corresponding status bit is high immediately generates an interrupt.

Fuel-Gauge Registers

FGCFG—Fuel-Gauge Configuration Register

(Read/Write)
Address 0xDA

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	FGON	0	Writing a 1 turns ON the fuel-gauge block
D6	FGCALON	0	Set to 0 for normal operation. Used for factory test. 0 = Track the charge flow between CSP and CSN 1 = Internally connects CSP to CSN. Note the CSP pin is high impedance.

FGCFG—Fuel-Gauge Configuration Register

(Read/Write)
Address 0xDA

D5-D4	FGMUX[1:0]	00	FGMUX[1:0] chooses the source signal for latching counter data. 00 = Enables FGLATCHNOW 01 = Charge and discharge counts are latched at the start and completion of an ADC conversion. 10 = RTCC (TEI) interrupt latches data. 11 = Not used.
D3	FGCHGSTAT	—	0 = Discharge current direction. 1 = Charge current direction. (This bit is READ ONLY.)
D2	FGLATCHNOW	0	Writing a 1 into this bit latches the contents of both fuel-gauge counters into FGDISH, FGDISL, FGCHGH, and FGCHGL if FGMUX=00.
D1	FGCLRDIS	0	Writing a 1 into this bit clears the discharge counter
D0	FGCLRCHG	0	Writing a 1 into this bit clears the charge counter

FGINT—Fuel-Gauge Interrupt Configuration and Status

(Read/Write)
Address 0x16

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7-D6	OVFRES[1:0]	00	Sets overflow interrupt frequency. 00 = 65,536 counts (2.27555mAhr/interrupt at 10mΩ) 01 = 32,768 counts (1.13777mAhr/interupt at 10mΩ) 10 = 16,384 counts (0.56888mAhr/interrupt at 10mΩ) 11 = 8192 counts (0.284444mAhr/interrupt at 10mΩ)
D5	ENDIRCHANGE	0	Enable/mask direction change interrupt (0 = mask interrupt; 1 = enable interrupt)
D4	ENDISOV	0	Enable/mask discharge overflow interrupt (0 = mask interrupt; 1 = enable interrupt)
D3	ENCHGOV	0	Enable/mask charge overflow interrupt (0 = mask interrupt; 1 = enable interrupt)
D2	DIRCHANGE	0	Set high (until it is cleared) whenever there is a direction change of the current flowing through the sense resistor. (write a 0 to clear; write 1 does nothing)
D1	DISOV	0	Discharge-counter overflow interrupt status (high when pending) (write a 0 to clear; writing 1 does nothing)
D0	CHGOV	0	Charge-counter overflow interrupt status (high when pending) (write a 0 to clear; writing 1 does nothing)

FGDISL—Fuel-Gauge Discharge Count Low Byte

(Read Only)
Address 0xDB

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	FGDDIS[7:0]	0	Fuel-gauge discharge count latch low byte.

FGDISH—Fuel-Gauge Discharge Count High Byte

(Read Only)
Address 0xDC

FGDISH—Fuel-Gauge Discharge Count High Byte

(Read Only)
Address 0xDC

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	FGDISH[15:8]	0	Fuel-gauge discharge count latch high byte.

FGCHGL—Fuel-Gauge Charge Count Low Byte

(Read Only)
Address 0xDD

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	FGCHG[7:0]	0	Fuel-gauge charge count latch low byte.

FGCHGH—Fuel-Gauge Charge Count High Byte

(Read Only)
Address 0xDE

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	FGCHG[15:8]	0	Fuel-gauge charge count latch high byte.

Data-Acquisition Unit

General Description

The MAX1781 data-acquisition unit combines a high-voltage analog front-end/multiplexer (AFE), and precision-integrating ADC. The AFE can be directly connected to two-, three-, or four-series lithium-ion cells, and configured to provide sixteen different voltage measurements. The ADC is configurable for eight resolutions: 9, 10, 11, 12, 13, 14, 15, and 16 bits, and automatically shuts down after conversions to conserve power. Communication to and from the data-acquisition unit is through data memory mapped registers. *Figure 11* below shows the architecture of the unit.

Features

- Differential Measurement of Individual Cell Voltages
- Automatic Cancellation of AFE Input Bias Currents from Battery Cells
- User-Selectable ADC Resolution (9 to 16 bits)
- Automatic Shutdown upon Conversion Completion
- Conversions Independent of CPU Operation
- User-Maskable Interrupt upon End of Conversion
- On-Chip Temperature Sensor
- Maximum Differential Cell-Voltage Measurement Error ±22mV

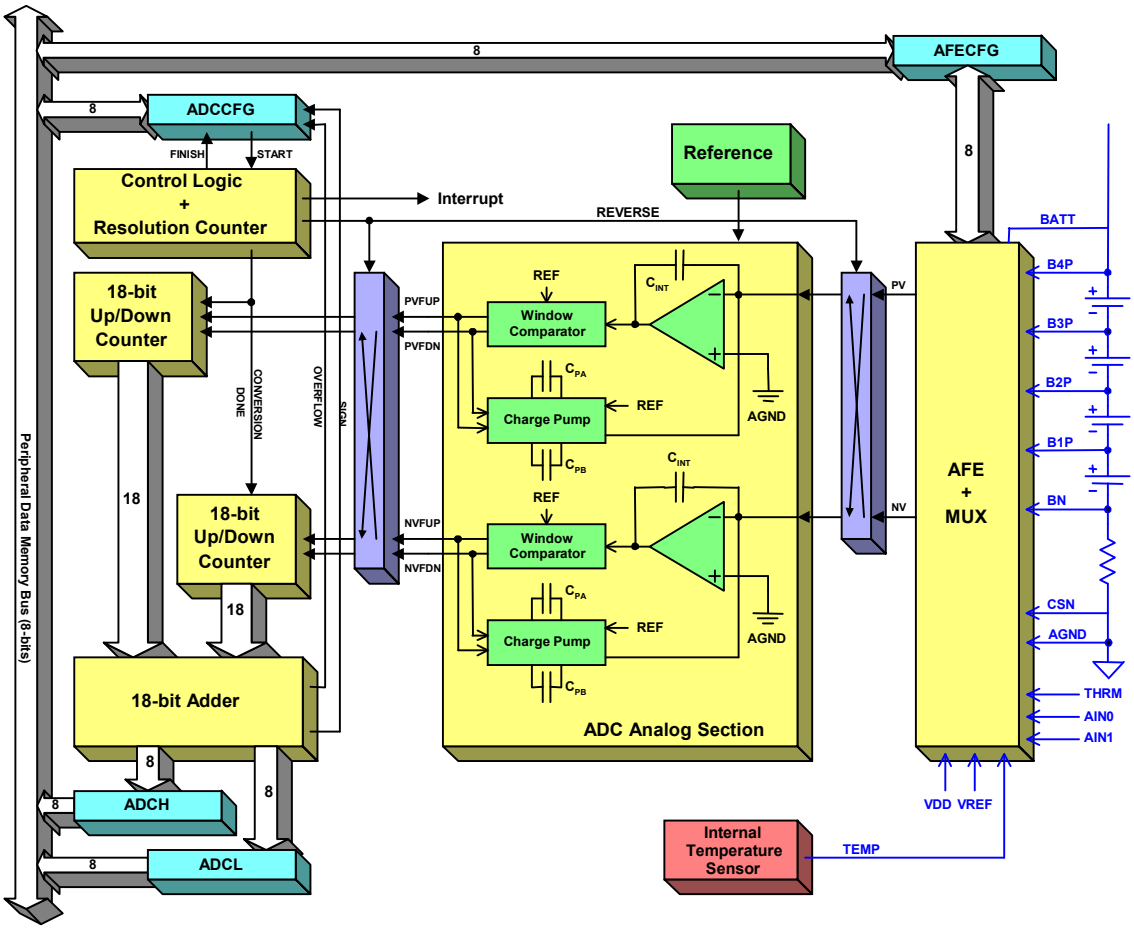


Figure 11. Data-Acquisition Unit Block Diagram

Analog Front-End/Multiplexer (AFE)

The analog front-end selects and scales the desired input signal for the ADC. The AFE has four measurement ranges:

1) High-voltage for cell stack voltages with 20.48V full scale.

2) Low-voltage for reference, V_{AA}, offset, thermistor, and general-purpose measurements with 5.12V full scale.

3) Micro-Voltage for quick cell-current measurements between BN and CSN with 0.4092V full scale.

4) Temperature for measurement of die temperature with 640º Kelvin full scale.

Input Bias Cancellation

While the MAX1781's ADC measures battery-pack cell voltages in the high-voltage range, each connection to the ADC connects a 4MΩ resistor from the measured pin to ground during conversion. Although the resulting input bias current is small, even small currents from intermediate cell-stack voltages can imbalance a cell stack over time. To protect against this, the MAX1781 incorporates a special input-bias-current cancellation circuit (ICAN). Programming the BIASCANEN bit in the AFECFG register automatically enables the ICAN circuit for intermediate cell voltage measurements during a conversion. Note that while the ICAN circuit is enabled, about 65µA of additional supply current is drawn from the BATT pin for each pin whose input bias current is being canceled. When enabled, the ICAN circuit typically reduces the uncancelled input bias current by a factor of 100. Connect unused high-voltage inputs as follows:

Cell connections for 4-cell: B4P, B3P, B2P, B1P, BN

Cell connections for 3-cell: B4P, B3P, B2P = B1P, BN

Cell connections for 2-cell: B4P, B3P = B2P = B1P, BN

AFE Register

AFECFG—AFE Configuration Register

(Read/Write)
Address 0x10

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	THRM	0	When this bit is 1, THRM is internally connected to V_{AA} with a 50Ω PMOS switch. When THRM is 0, the THRM pin is high impedance.
D6	BIASCANEN	0	1 = Input Bias Cancellation Enabled
D5	BATTON	0	Selects which linear regulator to use for part. Set BATTON = 0 to power V_{AA} from PCKP. Set BATTON = 1 to power V_{AA} from BATT.
D4	—	0	Reserved for Future Use. Location is read/writeable. Function may change in future parts.
D3-D0	AFESELECT[3:0]	0000	See table below

Analog Full-Scale	16-bit LSB	AFESELECT		Channel
5.12V	0.125mV	0	**0000**	AGND
		2	**0010**	V_{DD}
		3	**0011**	V_{REF}
		5	**0101**	AIN0
		6	**0110**	AIN1
		7	**0111**	THRM
0.4096V	10µV	4	**0100**	BN to CSN
358ºK (85ºC)	0.015625ºK	1	**0001**	Die Temperature
20.48V	0.5mV	8	**1000**	B1P to BN
		9	**1001**	B2P to BN
		A	**1010**	B3P to BN
		B	**1011**	B4P to BN
		C	**1100**	B1P to BN
		D	**1101**	B2P to B1P
		E	**1110**	B3P to B2P
		F	**1111**	B4P to B3P

Analog-To-Digital Converter (ADC)

Operation

To perform an ADC conversion, first program the AFECFG register with the desired measurement mode. ADCSTCONV initiates the conversion as shown in *Figure 12*. Start the conversion by setting the ADCSTCONV bit to a 1 in the ADCCFG register. The ADC powers up and begins a conversion. After 6 clock cycles of the 32,768Hz oscillator an interrupt is generated if the ADC is configured to interrupt upon conversion start (ADCINTMODE = 1). After the interrupt is serviced ADCI (see IS2) is cleared. Upon completion of the conversion, the control logic clears the ADCSTCONV bit in the ADCCFG register and the conversion results are stored in ADCH and ADCL. The status of ADCSTCONV indicates if the interrupt is caused by conversion start or completion. At the end of each conversion cycle, the ADC automatically powers itself down to conserve energy.

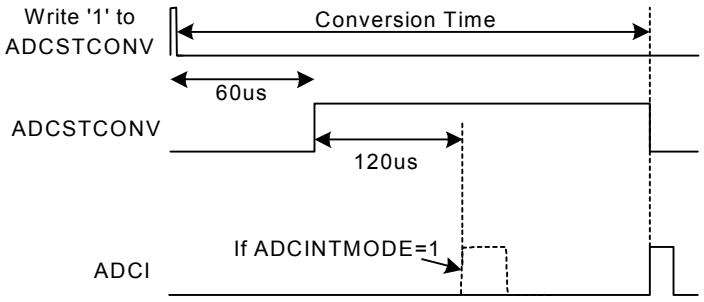


Figure 12. ADCSTCONV and ADCI Timing

When performing a 16-bit conversion, the ADC has an analog full-scale of 40,960 counts, and a digital full-scale of 65,536 counts. Values measured above the analog full-scale of 40,960 counts have reduced accuracy.

Digital Counter/Adder

The ADC is composed of two voltage-to-frequency converters that produce a count proportional to the input voltage. There are two 18-bit up/down counters whose counts are one's-complement-added to obtain a 16-bit binary conversion value. Although the highest resolution is 16 bits, the counters and adder are 18 bits wide to implement sign and overflow. An 18-bit adder sums the two counter outputs and latches the final results in ADCH and ADCL.

ADCCFG Register Description

The ADCCFG register provides SIGN, OVERFLOW, and ADCLIMIT bits that are used to interpret the ADC conversion results. The SIGN bit in ADCCFG indicates the polarity of the measurement. SIGN, when combined with ADCH and ADCL, produces the signed 17-bit measurement. When OVERFLOW = 1, the ADC measurement is above digital full-scale. If limiting is enabled (ADCLIMIT = 1) the ADC result equals 0xFFFF for measurements above digital full-scale and 0x0000 for negative voltages. For BN to CSN measurements (AFESELECT = 0100) the OVERFLOW bit and limiting will be invalid for results above twice the digital full-scale (1.31072V).

Bits D3, D2, D1 of ADCCFG form ADCRES, which sets the sampling time and therefore the resolution of conversion. Adjust the resolution from 9 to 16-bit by setting ADCRES as follows:

Resolution = 9 + ADCRES

All measurements are left-shifted in ADCH and ADCL so the bit weights are the same for each resolution setting. The LSBs contain zeros for the lower resolution settings.

Table 9. ADC Range and Resolution

FULL SCALE	20.48V	5.12V	0.4096V BN-CSN	358ºK
RESOLUTION (bits)	LSB SIZE (mV)	LSB SIZE (mV)	LSB SIZE (mV)	LSB SIZE (°K)
9	±64	±16	±1.28	±2.0
10	±32	±8.0	±0.64	±1.0
11	±16	±4.0	±0.32	±0.5
12	±8	±2.0	±0.16	±0.25
13	±4	±1.0	±0.08	±0.125
14	±2	±0.5	±0.04	±0.0625
15	±1	±0.25	±0.02	±0.03125
16	±0.5	±0.125	±0.01	±0.015625

ADC/AFE Interrupts

The ADC strobes its interrupt signal high at the beginning and end of a conversion, as shown in *Figure 12*. The fuel gauge can be configured to latch its data according to these interrupts, which allows simultaneous battery current and voltage measurements. The ADCCFG register can disable this feature and only generate an interrupt at the end of conversion by setting INTMODE = 0. Interrupt status and enable bits are in the IS2 and IE2 registers.

FREEZE

When the analog peripherals are frozen using the SMBus, write a 1 to FREEZEALL or FREEZE bits in ISPDBGCFG to suspend conversion. The analog circuitry remains active and the part continues to draw full current.

When FREEZE goes high, the 32,768Hz clock to the ADC state machine is gated off after a synchronization delay. This delay along with analog start/stop details creates a maximum of -16 count to +3 count conversion error per FREEZE transition.

Because there is no arbitration between the FREEZE and end of conversion, there exists a 1:500,000 synchronization failure possibility that could load unpredictable results into ADCH and ADCL at 9-bit resolution. This possibility only occurs if FREEZE is asserted during conversion. This possibility decreases at increased resolution. i.e., 9-bits: 1:500000, 16-bits: 1:128,000,000. There is no concern of synchronization failure during normal operation.

ADC Operation with PFW

Power-fail warning (PFW) stops ADC conversion but does not reset the ADC's counter results. If a PFW interrupt occurs during an end-of-conversion interrupt routine, it is still safe to read the ADC results in this interrupt routine. If PFW is pending before the ADC interrupt the ADC results cannot be trusted. Starting a conversion with PFW active clears the ADCH and ADCL to 0x0000.

Understanding ADC Error Sources

The MAX1781's ADC has four sources of error:

Gain Error: ADC gain error is the ratio of the output code for full-scale input relative to the ideal output code at full scale. Differential cell measurements have a gain error of +/-0.5%. All other channels have a +/-1% typical gain error.

Integral Nonlinearity Error (INL): The ADC integral nonlinearity error is less than 2mV in high-voltage mode, and 0.5mV in low-voltage mode. The ADC is guaranteed monotonic.

Offset Error: The input offset of the two Voltage-to-frequency converters introduces an offset error into the result. The offset error is typically only a concern for AIN0, AIN1, and THRM conversions and can be ignored for other channels. For AIN0, AIN1, and THRM the typical input offset is 300µV. For other channels the error is below 1 16-bit LSB.

Quantization Error: Each V-to-f converter has ±0.5 LSB quantization error. Because two V-to-f converters are digitally subtracted, the MAX1781 ADC has ±1 LSB quantization error. Although most conversions exhibit ±0.5 LSB quantization error, the worst-case quantization error is ±1 LSB.

To calculate the worst-case ADC error in Volts use the following equation:

$$V_{ERR} = (V_{IN} \cdot GainError) + INL \cdot V_{FullScale} + V_{Offset} + (V_{Quantization} \cdot (17 - Resolution))$$

where,

Resolution is the conversion resolution in bits (9 bits to 16 bits)

V_{IN} is the measured voltage

GainError is the full-scale gain error

INL is the integral nonlinearity in percent (see ELECTRICAL CHARACTERISTICS)

$V_{FullScale}$ is the analog full-scale

V_{Offset} is the ADC's offset

$V_{Quantization}$ is the quantization error

ADC Registers

ADCCFG—ADC Configuration Register

(Read/Write)
Address 0x11

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	OVERFLOW	0	0 = In Range (read only) 1 = Digital Overflow
D6	SIGN	0	0 = Positive (read only) 1 = Negative
D5	ADCLIMIT	1	1 = Limiting On 0 = Limiting Off Limiting affects ADCL/ADCH (0x012 and 0x013). Conversion does not have to be repeated to toggle between limited and nonlimited results. Negative result returns 0x0000, positive overflow returns 0xFFFF.
D4	INTMODE	0	0 = Interrupt at end of conversion only 1 = Interrupt at start and end of conversion Facilitates simultaneous current and voltage measurements when used with the fuel gauge.
D3 D2 D1	ADCRES[3:1]	011	000: Resolution 9 bits: Conversion Time 0.885ms. 001: Resolution 10 bits: Conversion Time 1.62ms. 010: Resolution 11 bits: Conversion Time 3.08ms. 011: Resolution 12 bits: Conversion Time 6.01ms. 100: Resolution 13 bits: Conversion Time 11.9ms. 101: Resolution 14 bits: Conversion Time 23.6ms. 110: Resolution 15 bits: Conversion Time 47.0ms. 111: Resolution 16 bits: Conversion Time 93.9ms.
D0	ADCSTCONV	0	Writing a 1 to this bit starts a conversion cycle. Upon completion of a conversion cycle, the ADC logic resets this bit to 0 indicating that sampled data is ready to be read.

ADCL—ADC Result Low Byte

(Read Only)
Address 0x12

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	ADC_D7	0	ADC Result Bit 7. (9-bit LSB)
D6	ADC_D6	0	ADC Result Bit 6. (10-bit LSB)
D5	ADC_D5	0	ADC Result Bit 5. (11-bit LSB)
D4	ADC_D4	0	ADC Result Bit 4. (12-bit LSB)
D3	ADC_D3	0	ADC Result Bit 3. (13-bit LSB)
D2	ADC_D2	0	ADC Result Bit 2. (14-bit LSB)

ADCL—ADC Result Low Byte

(Read Only)
Address 0x12

D1	ADC_D1	0	ADC Result Bit 1. (15-bit LSB)
D0	ADC_D0	0	ADC Result Bit 0. (16-bit LSB)

ADCH—ADC Result High Byte

(Read Only)
Address 0x13

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	ADC_D15	0	ADC Result Bit 15
D6	ADC_D14	0	ADC Result Bit 14
D5	ADC_D13	0	ADC Result Bit 13
D4	ADC_D12	0	ADC Result Bit 12
D3	ADC_D11	0	ADC Result Bit 11
D2	ADC_D10	0	ADC Result Bit 10
D1	ADC_D9	0	ADC Result Bit 9
D0	ADC_D8	0	ADC Result Bit 8

AIN and THRM Pins

Many battery packs use a thermistor to monitor cell temperature during operation. For Li+ packs, pack temperature is used to inhibit charging at extremely high and low ambient temperatures, while in NiMH packs the rate-of-change of cell temperature is used to terminate charge cycles. The MAX1781 provides three general-purpose analog channels, AIN0, AIN1, and THRM. Although THRM can be used as another 5.12V analog channel just as AIN0 and AIN1, it has a switch to V_{AA} (see THRM bit in AFECFG), making it well-suited for measuring temperature through the use of an external thermistor yet conserving power when temperature measurements are not necessary.

To measure an external thermistor set the THRM bit in the AFECFG register to power up the ratiometric thermistor circuit. Connect the thermistor as shown in Figure 13. Measure THRM and AIN0 using the ADC. By measuring both THRM and AIN0, voltage drop errors in the P-channel MOSFET and V_{AA} supply variation are calibrated out of the measurement. Two thermistors can be measured by connecting a second thermistor-divider to THRM and the AIN1 pin.

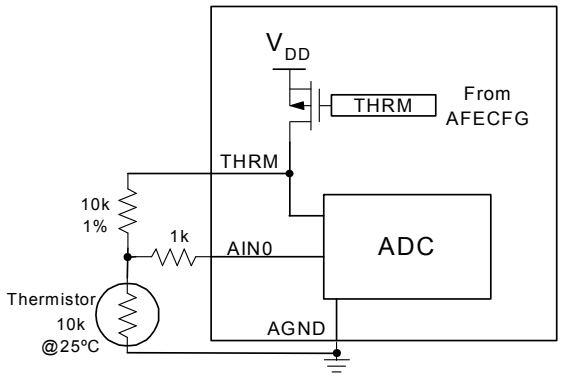


Figure 13. MAX1781 Thermistor Circuit

BN to CSN Current-Sense Resistor Measurement

The MAX1781 includes a 0.4096V ADC channel for accurately measuring the battery current. Although battery current can be measured using the fuel gauge, it can be measured more quickly using the ADC. This input has the following features:

- 0.4096V Analog Full Scale (40.96A using a 10mΩ R_{SENSE} resistor)
- 10µV LSB 16-bit Resolution (1mA using a 10mΩ R_{SENSE} resistor)
- <10µV Offset typical (1mA using a 10mΩ R_{SENSE} resistor)
- 80kΩ of Input Resistance
- ±1% Absolute Accuracy Limits

Temperature Sensor

The MAX1781 has an on-chip temperature sensor, a bipolar proportional-to-absolute-temperature (PTAT) circuit, which closely tracks the MAX1781 die temperature. This parameter is un-trimmed in production.

Overcurrent Protection and Protection MOSFET Drivers

Description

The MAX1781 continuously monitors the discharge current to detect an over-discharge current or short-circuit current condition. Over-discharge and short-circuit are detected when the voltage at ODI or SCI is below the voltage at CSN. Connect the SCI and ODI inputs to voltage-dividers from the V_{AA} supply as shown in *Figure 14*.

The MAX1781 detects two discharge current thresholds: over-discharge, set at the ODI pin, and short-circuit, set at the SCI pin. The short-circuit/over-discharge thresholds are 12% accurate (see

Setting Over-Discharge Current and Short-Circuit Current Thresholds.) ODSCDLY sets a blanking timer for each threshold. Use OD[3:0] in ODSCDLY to adjust the over-discharge current delay from 0 to 14.5ms in 0.96ms increments. Use SC[3:0] to adjust the short-circuit current delay from 0 to 1ms in 66µs increments.

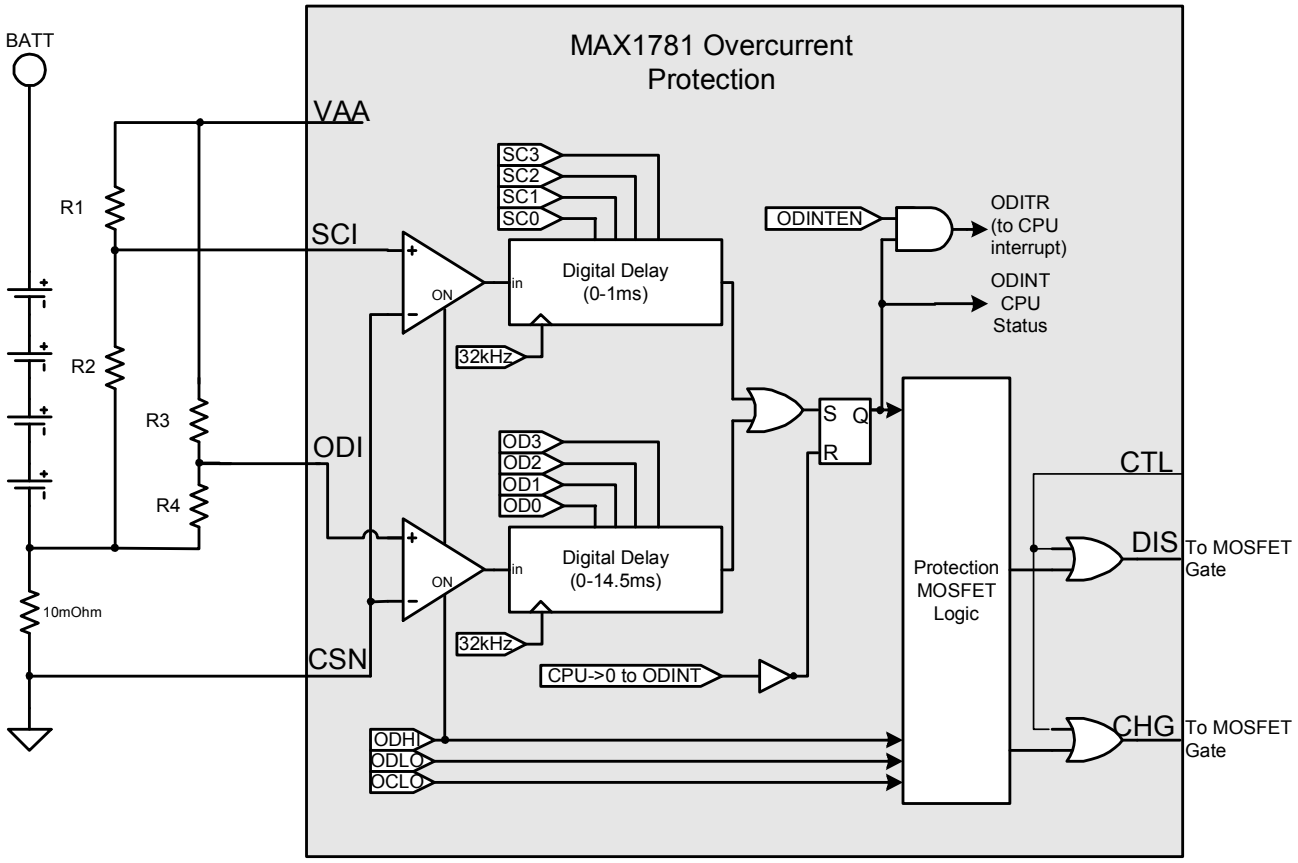


Figure 14. Overcurrent Comparator Functional Diagram

The interrupt occurs only when the over-charge or short-circuit persists for longer than the time set by ODSCDLY (see ODSCDLY.)

Upon detection of over-charge or short-circuit an interrupt is generated and ODINT (in ODSC) and ODI (in IS2) are set if the interrupt is enabled (ODINTEN=1 in ODSC and ODSCEI=1 in IE2).

When the ODHI bit is 1, and the ODINT interrupt is pending, the protection MOSFET driver automatically opens the discharge MOSFET. The discharge MOSFET remains off until 0 is written to the ODINT bit of the ODSC register. The overcurrent protection block continues to provide protection even when the processor is in sleep mode.

ODHI enables/disables the over-discharge comparators.

Table 10. Over-Discharge Current Logic

MODE	ODHI	ODLO	DIS	EXTERNAL DISCHARGE MOSFET STATE	OVERCURRENT COMPARATORS

Direct Control	Force DIS High	0	0	V_{BATT}	Off	Off
	Force DIS Low	0	1	V_{BATT} - 8V	On	Off
Auto Control	Force DIS High	1	0	V_{BATT}	Off	On
	Auto Low	1	1	V_{BATT} or V_{BATT} - 8V	On (Off during overcurrent)	On

Table 11. Charge Current Logic

MODE	OCLO	CHG	EXTERNAL CHARGE MOSFET STATE
Force CHG low	1	V_{PCKP} - 8V	On
Force CHG high	0	V_{PCKP}	Off

Using Software to Control the Protection MOSFETs

Control of the discharge overcurrent MOSFET is either automatically controlled by the overcurrent protection block, or directly controlled. The ODHI and ODLO bits of the ODSC register determine the behavior of the discharge MOSFET.

Table 10 shows the discharge MOSFET as a function of ODHI and ODLO. When ODHI = ODLO = 1 this MOSFET is automatically controlled. When in auto control, avoid shutting down the comparators after an overcurrent event has been detected by setting ODHI=1 and ODLO=0 to force the discharge MOSFET off. When ODHI = 0 the state of the discharge MOSFET is determined by ODLO.

The charge MOSFET is controlled only through firmware and is not effected by the over-discharge current or short-circuit current comparators. OCLO controls the state of the charge MOSFET as shown in table 11.

Clearing Overcurrent Interrupts

When an overcurrent condition is detected on ODI or SCI, an interrupt is generated and ODINT is set. ODINT is level sensitive, so if an overcurrent condition persists, so does the interrupt.

Interrupt enable bits, ODSCEI in IE2 and ODINTEN in ODSC, must both be 1 to enable the interrupt. The status bits, ODSCI and ODINT, are both set to 1 when the interrupt condition occurs. ODINT is level sensitive, and ODSCI (in IS2) is edge sensitive. Although ODINT persists during an overcurrent event it must be manually cleared after the overcurrent event is removed. The net effect is an edge-sensitive interrupt if ODINT is cleared before ODSCI, and a level sensitive interrupt if ODSCI is cleared before ODINT. Writing a 1 to either ODSCI or ODINT does nothing.

It may take a few hundred microseconds for the ODINT signal to be clearable after the protection MOSFETs are opened. For this reason it is important to track ODINT and periodically clear and poll the status bit to determine if the overcurrent event is complete.

ODSC FREEZE Operation

The FREEZE bit (in ISPDBGCFG) disables the overcurrent protection block. CHG and DIS remain in their previous state.

CHG and DIS Output Driver Description

The MAX1781 directly drives two high-side drain-connected external PMOS protection FETs. The driver contains an 8V gate-drive clamp, allowing the use of 12V V_{GS}-rated MOSFETs. The driver's class B output saves quiescent current and provides a low-impedance drive during shutdown. When the V_{DD} supply is low the

MAX1781 MOSFET drivers pull the gate within 1V of the source. A 4MΩ internal resistance pulls the gate all the way to the source and reduces MOSFET leakage during shutdown. The MAX1781 incorporates fast P-channel MOSFET drivers, as demonstrated in the *Typical Operating Characteristics*.

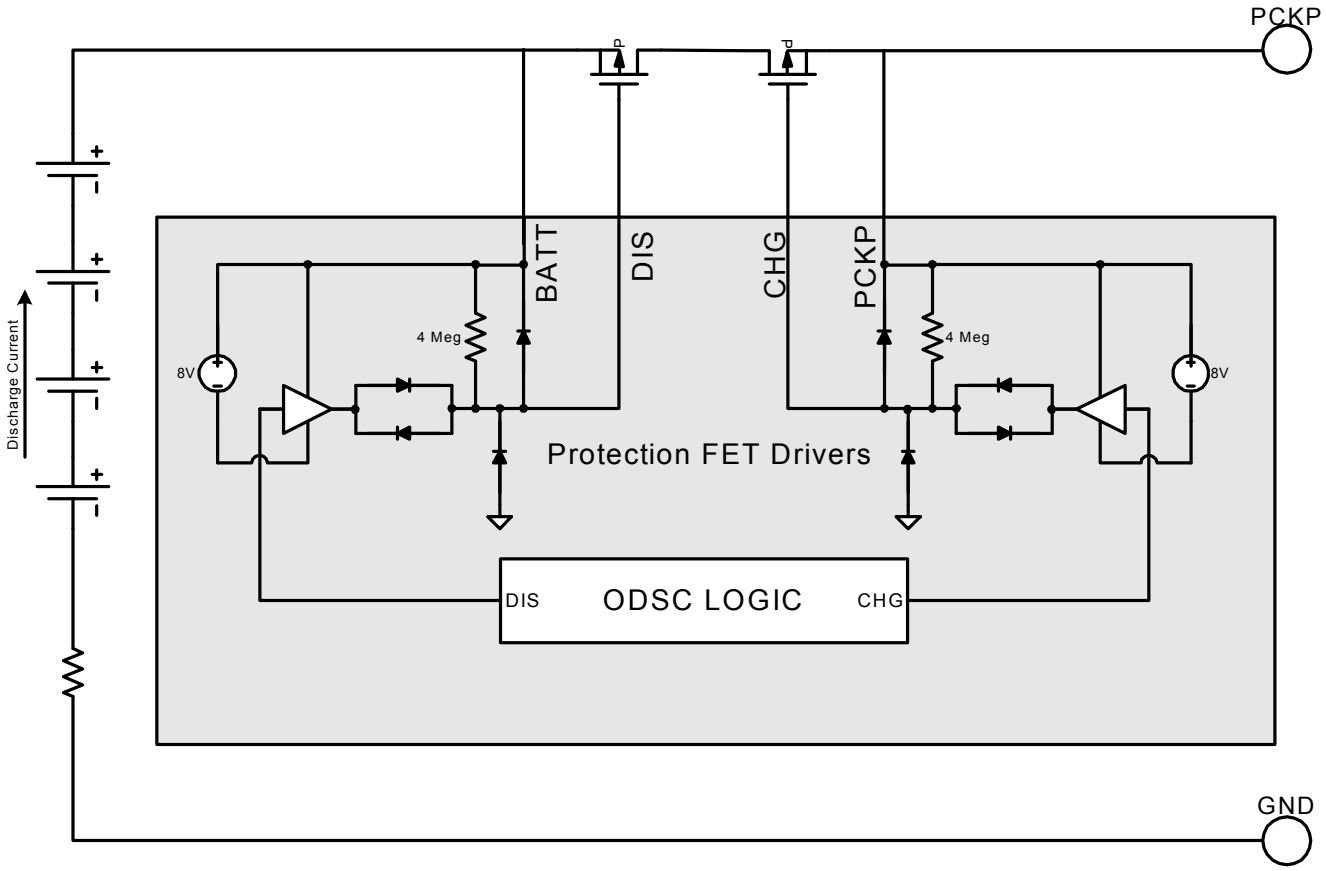


Figure 15. CHG and DIS pin Driver Simplified Schematic

ODSC Registers

ODSC—ODSC Configuration Register

(Read/Write)
Address 0x15

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7-D5	Unused	0	Reads 0.
D4	OCLO	0	Controls the CHG output state. 0 = CHG FET OFF 1 = CHG FET ON
D3	ODINTEN	0	1 = Enable ODINT 0 = Mask ODINT
D2	ODINT	0	Over-discharge Interrupt Status - Level sensitive. This bit is set when there is too much discharge current (i.e., short circuit of battery pack). ODINT stays high until the overcurrent condition stops and ODINT is cleared by writing 0. Writing a 1 does nothing.
D1	ODHI	0	First of the two bits controlling the DIS output state. 0 = Direct Control, Over-discharge Comparator Off 1 = Auto Control, Over-discharge Comparator On

ODSC—ODSC Configuration Register

(Read/Write)
Address 0x15

D0	ODLO	0	Second of the two bits controlling the DIS output state. 0 = DIS FET OFF 1 = DIS FET ON in direct control or if ODINT = 0

ODSCDLY—Over-discharge and Short-Circuit Delay

(Read/Write)
Address 0x0DF

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7-D4	SC[3:0]	0000	Short-Circuit Current Delay Setting (61µs per count)
D3-D0	OD[3:0]	0000	Over-discharge Current Delay Setting (976µs per count)

High-Voltage Output Port

Description

The MAX1781 provides eight high-voltage outputs, HV7–HV0. These open-drain outputs can be pulled up to 28V. Register PHV controls the state of the HV[7:0] outputs.

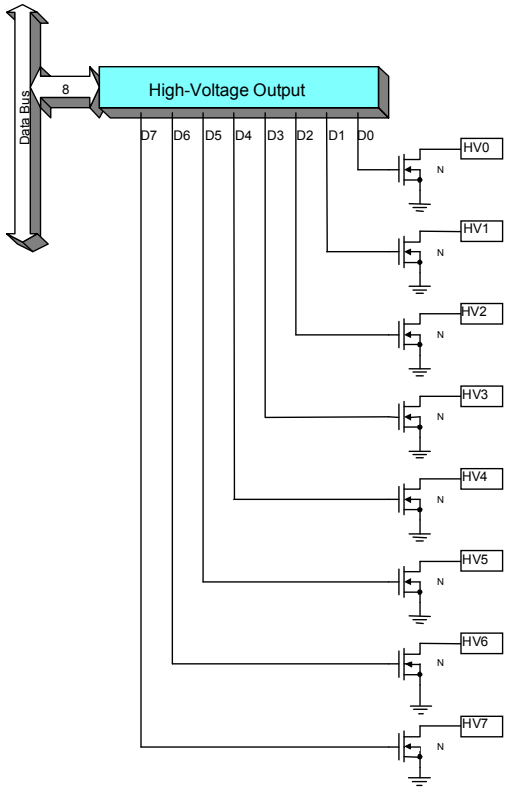


Figure 16. High-Voltage Output Port

HV Register

PHV—High-Voltage Output Port Register

(Read/Write)
Address 0x0D9

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	HV7	0	0 = HV7 is high impedance. 1 = HV7 is pulled to GND.
D6	HV6	0	0 = HV6 is high impedance. 1 = HV6 is pulled to GND.
D5	HV5	0	0 = HV5 is high impedance. 1 = HV5 is pulled to GND.
D4	HV4	0	0 = HV4 is high impedance. 1 = HV4 is pulled to GND.
D3	HV3	0	0 = HV3 is high impedance. 1 = HV3 is pulled to GND.
D2	HV2	0	0 = HV2 is high impedance. 1 = HV2 is pulled to GND.
D1	HV1	0	0 = HV1 is high impedance. 1 = HV1 is pulled to GND.
D0	HV0	0	0 = HV0 is high impedance. 1 = HV0 is pulled to GND.

Digital Peripherals

Port A

PA7–PA0 can be individually programmed to be either: CMOS inputs, CMOS outputs, or special functions, as determined by the PACFG register. PA7–PA0 can also be used as open-drain outputs by setting PA = 0 and driving PAEN with inverted data. When port A IO are used as inputs do not leave them unconnected. Program unused Port A IO as outputs.

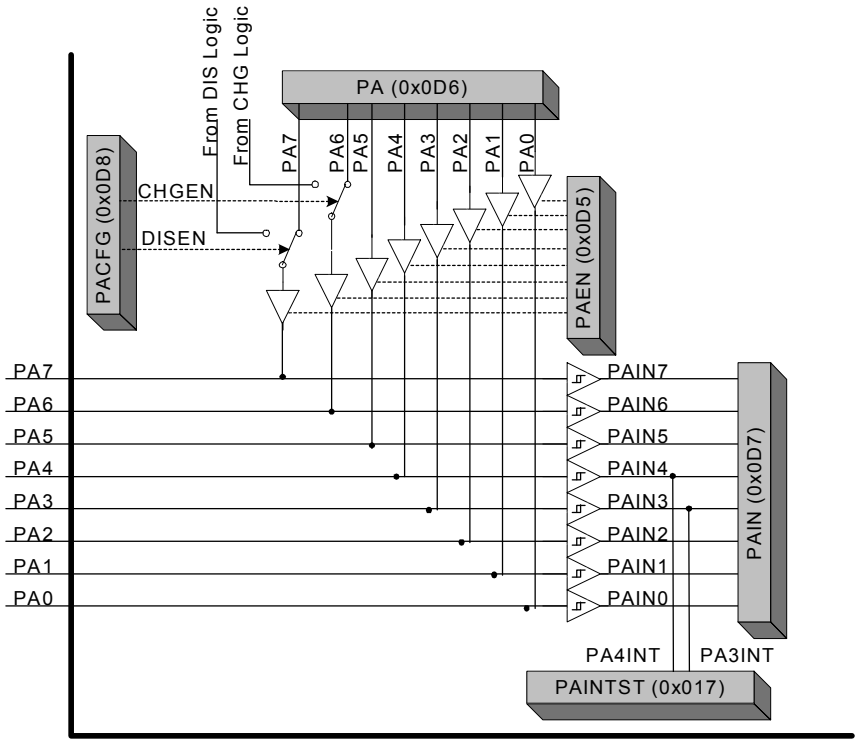


Figure 17. Port A Block Diagram

Port A consists of a latch (PA), an output driver, and an input buffer (PAIN). The port latch is memory mapped so that the CPU can read from or write to the latch directly. Firmware can read or write the latch (PA) or read the input buffers (PAIN). PAIN may not always immediately equal PA due to capacitance at the output.

Port A interrupts

PA3 and PA4 can be configured as interrupt inputs on the MAX1781. An interrupt occurs when a positive or negative edge is detected on PA4 or PA3. Enable the interrupt by setting PA4INTEN or PA3INTEN to 1 and PAEI=1 (in IE1). PA4INT or PA3INT remain high until cleared and are active even when the corresponding interrupts are disabled. All interrupt sources are asynchronous to the CPU clock and edge sensitive.

Port A Registers

PAEN—Port A Output Enable Register

(Read/Write)
Address 0x0D5

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	PAEN7	0	PA7 port output enable. 1 = Output, 0 = High Impedance.
D6	PAEN6	0	PA6 port output enable. 1 = Output, 0 = High Impedance.
D5	PAEN5	0	PA5 port output enable. 1 = Output, 0 = High Impedance.
D4	PAEN4	0	PA4 port output enable. 1 = Output, 0 = High Impedance.
D3	PAEN3	0	PA3 port output enable. 1 = Output, 0 = High Impedance.
D2	PAEN2	0	PA2 port output enable. 1 = Output, 0 = High Impedance.
D1	PAEN1	0	PA1 port output enable. 1 = Output, 0 = High Impedance.
D0	PAEN0	0	PA0 port output enable. 1 = Output, 0 = High Impedance.

PA—Port A Output Register

(Read/Write)
Address 0x0D6

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	PA7	0	PA7 output latch. 1 = Output High, 0 = Output Low, if PAEN7 = 1.
D6	PA6	0	PA6 output latch. 1 = Output High, 0 = Output Low, if PAEN6 = 1.
D5	PA5	0	PA5 output latch. 1 = Output High, 0 = Output Low, if PAEN5 = 1.
D4	PA4	0	PA4 output latch. 1 = Output High, 0 = Output Low, if PAEN4 = 1.
D3	PA3	0	PA3 output latch. 1 = Output High, 0 = Output Low, if PAEN3 = 1.
D2	PA2	0	PA2 output latch. 1 = Output High, 0 = Output Low, if PAEN2 = 1.
D1	PA1	0	PA1 output latch. 1 = Output High, 0 = Output Low, if PAEN1 = 1.
D0	PA0	0	PA0 output latch. 1 = Output High, 0 = Output Low, if PAEN0 = 1.

PAIN—Port A Input Register

(Read Only)
Address 0x0D7

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	PAIN7	0	PA7 pin input state.
D6	PAIN6	0	PA6 pin input state.
D5	PAIN5	0	PA5 pin input state.
D4	PAIN4	0	PA4 pin input state.
D3	PAIN3	0	PA3 pin input state.
D2	PAIN2	0	PA2 pin input state.
D1	PAIN1	0	PA1 pin input state.

PAIN—Port A Input Register

(Read Only)
Address 0x0D7

D0	PAIN0	0	PA0 pin input state.

PACFG—Port A Configuration Register

(Read/Write)
Address 0x0D8

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	DISEN	0	Used for connection to 2nd level protector or external MOSFET driver. 1 = Output DIS MOSFET state enable. 0 = Disable
D6	CHGEN	0	Used for connection to 2nd level protector or external MOSFET driver. 1 = Output CHG MOSFET state enable. 0 = Disable
D5	Unused	0	Reads 0.
D4	PA4INTEN	0	1 = PA4 interrupt enable. 0 = Disable
D3	PA3INTEN	0	1 = PA3 interrupt enable. 0 = Disable
D2	Unused	0	Reads 0.
D1	Reserved	0	Factory Reserved. Write 0 to this bit.
D0	Unused	0	Reads 0.

PAINTST—Port A Interrupt Status Register

(Read/Write)
Address 0x017

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	Unused	0	Reads as 0.
D6	Unused	0	Reads as 0.
D5	Unused	0	Reads as 0.
D4	PA4INT	0	PA4INT goes high on a rising or falling edge of PA4 regardless of status of PA4EN. 1 = PA4 rising or falling edge detected. 0 = No edge detected on PA4 since last clear.
D3	PA3INT	0	PA3INT goes high on a rising or falling edge of PA3 regardless of status of PA4EN. 1 = PA3 rising or falling edge detected. 0 = No edge detected on PA3 since last clear.
D2	Unused	0	Reads as 0.
D1	Unused	0	Reads as 0.
D0	Unused	0	Reads as 0.

RTCC Timer and Watchdog

The timer block consists of a watchdog timer (WDT), a real-time clock counter (RTCC), and a prescalar. The timers are up-counting timers and the POR states are all 0.

The prescalar provides the clock to both WDT and RTCC and its frequency is fixed. The prescalar cannot be cleared by software and only a global reset signal can reset this counter. Input clock frequency for the prescalar is nominally 32,768Hz and the output clock period is 1.953ms.

T_{pre} = (1 / 32768Hz) x 2 ^ 6 = 1.953ms

If the watchdog timer reset is enabled (WDTEN = 1) the watchdog timer generates a system reset when the selected tap changes state from 1 to 0 because of a failure of software to clear the watchdog timer. The enable bit and WDTS[1:0] frequency-select bits can only be written once after system reset. These bits must be written

within 250ms of startup if the watchdog timer is ever to be used. Hardware ignores the rest of the write attempts for these three bits after this register has been written once. Set the CLRWDT bit in the WDT register to reset the watchdog timer.

RTCC provides an accurate timebase to the software service routine and it has four selectable frequencies. The RTCC generates an interrupt when the selected tap changes from 1 to 0 if the timer interrupt enable bit (TEI in IE2 register) is set. Set the CLRRTCC bit in the WDT register to reset the RTCC. Clear the RTCC less than 62.5ms before changing the RTCCS[1:0] frequency selection bits in order to avoid unintended interrupts.

The counters are not readable. Whenever FREEZE is high the 32,768Hz clock to the prescalar is gated off so the counters do not advance.

Watchdog Timer Register

WDT—Timer Register

(Read/Write)
Address 0x014

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	CLRRTCC	0	Write only. Set CLRRTCC to 1 to clear RTCC
D6	Unused	0	Read as 0.
D5 D4	RTCCS1 RTCCS0	0 0	RTCC frequency select bits 00 : RTCC time = (1 / 32kHz) x 4,096 = 0.125s 01 : RTCC time = (1 / 32kHz) x 8,192 = 0.250s 10 : RTCC time = (1 / 32kHz) x 16,384 = 0.500s 11 : RTCC time = (1 / 32kHz) x 32,768 = 1.000s
D3	CLRWDT	0	Write only. Set CLRWDT to 1 to clear watchdog timer
D2	WDTEN*	0	Watchdog Timer Reset Enable. 0 = disable. 1 = enable
D1 D0	WDTS1* WDTS0*	0 0	Watchdog Timer frequency select bits 00 : WDT time = (1 / 32kHz) x 16,384 = 0.5s 01 : WDT time = (1 / 32kHz) x 32,768 = 1s 10 : WDT time = (1 / 32kHz) x 65,536 = 2s 11 : WDT time = (1 / 32kHz) x 131,072 = 4s

**Bit [2:0] are write-once-read-many bits.*

SMBus Interface

Introduction

The system management bus (SMBus), is a two-wire bi-directional serial bus, that provides a simple and efficient interface for data exchange between devices. The SMBus uses a serial data line (SDA) and a serial clock line (SCL) for data transfer. SMBus data is transmitted in 8-bit packets. Data on the SMBus can be changed only when SCL is low and must be held stable when SCL is high. The most significant bit is transmitted first, and each byte is followed by an acknowledge bit (ACK). An ACK is generated by the device receiving data by pulling the SDA line low on the 9th SCL clock cycle. Therefore one complete data byte transfer needs 9 SCL clocks. If the receiver does not acknowledge (NACK) the transmitter after a byte has been transmitted, this NACK signals an end of data to the transmitter. The master then generates a STOP to signal the end of the transmission.

A complete specification for the SMBus can be found in the **System Management Bus Specification**, Revision 1.1, December 11, 1998.

Features

- Completely Interrupt Driven Operation
- Programmable Primary and Secondary SMBus Addresses, Broadcast Address Support
- Generates a Wake-Up Interrupt on Address Match
- Automatic bit-by-bit Arbitration of SMBus Communications
- Timers Automatically Recognize $T_{TIMEOUT}$, $T_{LOW:SEXT}$ and $T_{LOW:MEXT}$ Conditions
- Eight Software-Selectable Master SCL Clock Frequencies
- Programmable Glitch Filter with Two Filter Settings
- Predefined Manufacturer Access Commands for Test, Debug, and Programming Access

Description

The MAX1781 includes an interrupt driven SMBus interface. This allows the MAX1781 processor core to either sleep or process other tasks while the relatively slow SMBus transactions are handled.

The SMBus interface transmits and receives as either a master or slave device. In slave mode, data is received or transmitted while another device acts as the SMBus master. An SMBus master provides the clock or SCL signal for the transaction. In master mode the MAX1781 provides the SCL, as well as START and STOP. The master can transmit at eight user-selectable SCL clocks rates. The default master SCL clock rate is typically 50.8kHz. The slave receiver can respond to two user-programmable addresses at SCL clock speeds up to 500kHz (typical), well above the 100kHz specification of the SMBus. Once initialized, the slave receiver responds to SMBus communications at its primary and secondary address, even if the MAX1781 processor core is in SLEEP mode. master mode operations keep the instruction oscillator powered to initiate START generation, byte operations, and STOP generation. Enabling and disabling of the instruction oscillator is handled fully by the SMBus interface to minimize power consumption.

The SMBus interface compares each byte transmitted with the data on the bus in order to immediately detect a bus arbitration loss. When bus arbitration is lost, the SMBus interface switches to slave mode and continues to operate without loss of data.

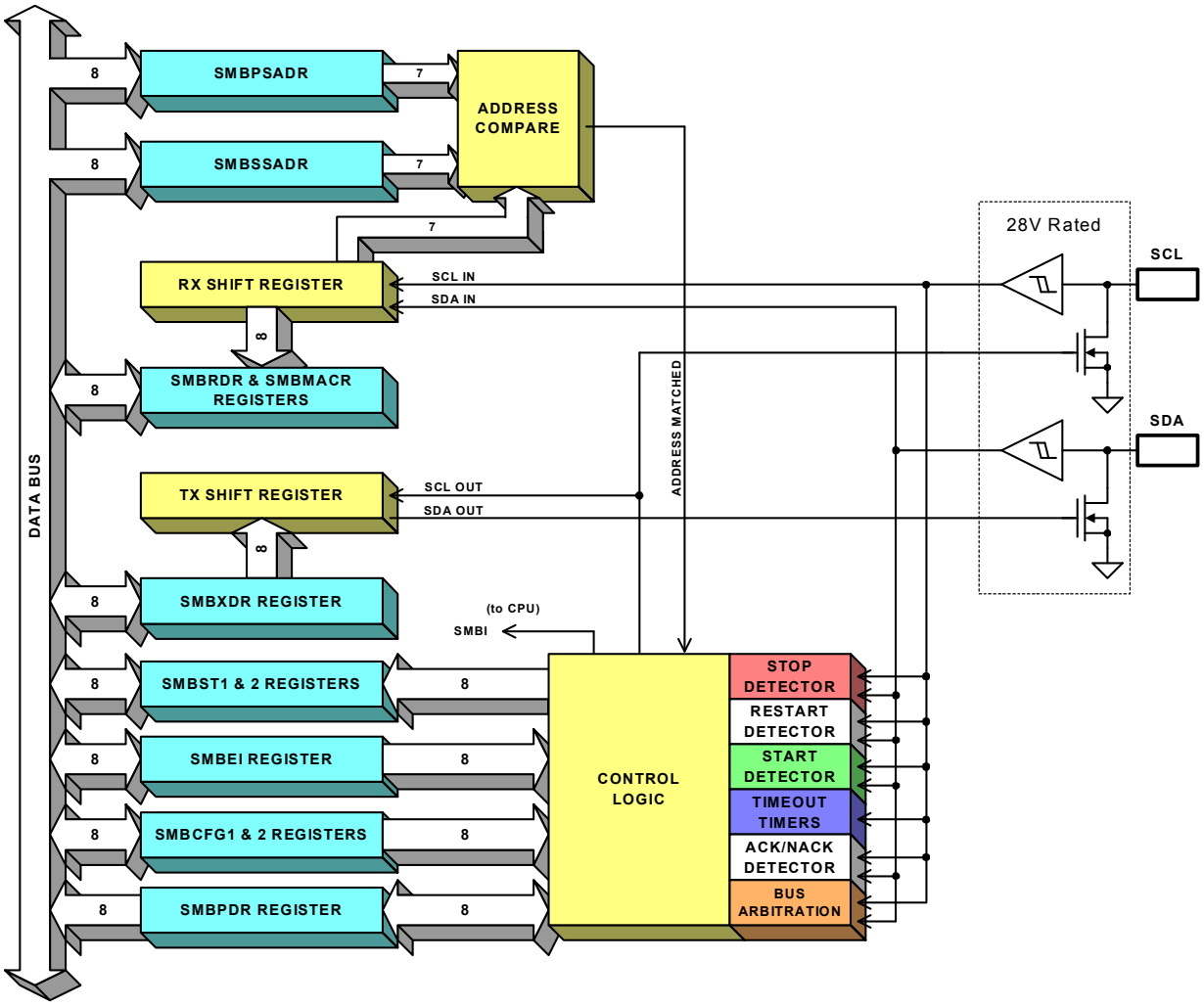


Figure 18. SMBus Peripheral Block Diagram

Transmit and Receive Buffers

Byte transmission and reception occur through the transmit and receive buffers SMBXDR and SMBRDR. Received bytes are placed in the SMBus received data register, SMBRDR. Place transmit bytes in the SMBus transmit data register, SMBXDR. A byte written to SMBXDR will begin transmission when the SMBus becomes free. Do not write another byte to SMBXDR until a TXDONE (transmit done) interrupt is received (see SMBST2). When any byte is transmitted TXDONE is set to 1. If loss of arbitration is detected an interrupt is triggered (MARBLOSS=1) and the MAX1781 stops transmitting and becomes slave.

Special SMBus hardware is activated when the first byte after the address is 0x00. In this case the command is treated as a smart battery data ManufacturerAccess() command (see *Manufacturer Access Commands*). For ManufacturerAccess() commands the op-code, which follows the 0x00 entry byte, is stored in the SMBus manufacturer access command receive register, SMBMACR. As the remaining bytes are received they are placed into the SMBXDR. SMBXDR effectively becomes a receive data register during ManufacturerAccess() commands.

Slave Sequence Control

The MAX1781 SMBus interface is set to slave mode by default. Write 00 to MSMODE[1:0] to configure the SMBus for slave mode. In slave mode the firmware is responsible for writing SMBXDR or reading SMBRDR, depending on the read/write bit of the address byte (R/W). The SMBus slave does not need to initiate a START or STOP, but is responsible for ACK/NACK control when receiving data. Normally set ACKEN to 1 unless firmware wishes to terminate the transaction.

After byte transmission (TXDONE interrupt) the clock is stretched (SCL held low) and the next transmit byte should be loaded into SMBXDR. Because clock stretching occurs after any byte is transmitted or received the SCLSTRETCH bit (in SMBCFG2) must be cleared to allow the next byte to be clocked out.

After byte reception (RXDONE interrupt) the clock is stretched (SCL held low) and the byte is available at SMBRDR. The SCLSTRETCH bit must be cleared to allow the next byte to be clocked in.

The MAX1781 can be configured to respond to a primary and secondary slave address. When in slave mode the first byte following START in any transaction is compared against either the general-call address (0x00), or the primary and secondary slave address. If any addresses match, an interrupt occurs (SADRMATCH[1:0] in SMBST1), if the interrupt is enabled (EIADRMTCH in SMBEI). After the address byte is transmitted, even when the address byte does not match the primary, secondary, or general-call address, an RXDONE interrupt is generated. For this reason RXDONE interrupts should typically be disabled in slave mode until an SADRMATCH interrupt occurs. Unlike normal byte reception the clock is not stretched after an address byte is received in slave mode.

START and STOP bits cannot be generated by the MAX1781 when in slave mode. Do not write to XSTART or XSTOP when in slave MODE.

Interrupts and status bits are available to indicate slave address match (EIADRMTCH in SMBEI), START, Repeated START (RSTRT), STOP, ACK, and NACK. A status bit, SRWB, indicates whether the MAX1781 is in slave transmit or slave receive mode, depending on R/W bit of the address byte.

Master Sequence Control

In master mode the firmware is responsible for initiating START and STOP as well as writing SMBXDR and reading SMBRDR, depending on the read/write bit of the address byte (R/W). Normally set ACKEN to 1 unless firmware wishes to terminate the transaction.

START, Repeated START, STOP, ACK, NACK, and master transmit/receive are controlled by the SMBus configuration register SMBCFG1, status register SMBST1, and interrupt enable register SMBEI. A START or STOP is initiated by writing 1 to XSTART or XSTOP in SMBCFG1. START will occur as soon as the bus is idle (BUSBUSY = 0). To generate a Repeated START, set XSTART to 1 during a transmission. The Repeated START immediately follows the byte currently in transmission. If XSTART is set to 1 when another master is communicating a START bit is generated after the bus becomes free. In this way START may be queued when another master is active. A STOP condition may be generated following the last byte's RXDONE or TXDONE interrupt, by setting XSTOP to 1. A STOP condition is generated automatically following any NACK. Following the STOP condition the MAX1781 is automatically reset to SMBus slave mode and MSMODE[1:0] is set to 00.

START, Repeated START, and STOP conditions are detected by the START, RSTRT, and STOP bits in the SMBus status register 1, SMBST1. An interrupt may be generated for any of these by enabling EISP in the SMBus Interrupt Enable Register (SMBEI).

Set MSMODE[1:0] to 10 to configure the MAX1781 for master transmit mode. After byte transmission (TXDONE interrupt) the clock is stretched (SCL held low) and the next transmit byte should be loaded into SMBXDR. Because clock stretching occurs after any byte is transmitted or received the SCLSTRETCH bit (in SMBCFG2) must be cleared to allow the next byte to be clocked out. If the bit transmitted does not equal the bit received, bus arbitration is lost (MARBLOSS=1 in SMBST2) and the MAX1781 returns to SMBus slave mode to

insure that SMBus communications are not lost. After arbitration loss the MAX1781 slave responds as normal. For master transmit mode without arbitration set MSMODE[1:0] to 11. If both XSTART and XSTOP bits are set in master transmit mode, a start, followed by the contents of the SMBXDR, followed by a stop, are sequentially transmitted on the bus.

The address byte is the first byte transmitted and is processed differently. RXDONE and TXDONE interrupts are generated simultaneously after transmission of the address byte. Unlike in slave mode, in master mode SCL is stretched at the end of the address byte.

Set MSMODE[1:0] to 01 to configure the MAX1781 for master receive mode. After a byte is received (RXDONE interrupt) the clock is stretched (SCL held low) and the byte is available in SMBRDR. The SCLSTRETCH bit in SMBCFG2 must be cleared to allow the next byte to be clocked in.

Interrupts and status bits are available to indicate master arbitration loss (MARBLOSS), START, Repeated START (RSTRT), STOP, ACK, and NACK.

Packet Error Code (PEC)

The MAX1781 includes Packet Error Code hardware which provides a fault tolerant method for transmitting or receiving data. A byte is calculated based on the SMBus sequence and transmitted at the end of that sequence. The receiving side compares it's calculation against the byte received to determine if there was an error in the transmission. This byte is called the PEC. The PEC is computed using CRC-8 as detailed in the SMBus 1.1 specification. The SMBus PEC data register (SMBPDR) contains the PEC as computed since the register was last cleared, which occurs automatically upon the transmission/reception of a START condition. The PEC is calculated both during transmit or receive mode, and whether the MAX1781 is configured as a Master or a Slave.

Set CPEC=1 (see SMBCFG2) to clear the computed PEC value. If implementing PEC, the PEC should be cleared at the end of any transmit or receive event. In this way SMBPDR always contains the correct PEC value.

Set RPEC=1 (see SMBCFG2) to compare the next byte received to the PEC held in SMBPDR. An ACK condition is automatically sent if the received PEC is correct. If the PEC is incorrect, a NACK is generated. After receipt of the PEC (as indicated by a RXDONE interrupt) clear RPEC by firmware.

Set XPEC=1 (see SMBCFG2) to transmit the PEC. Unlike other transmit operations the SMBXDR register should not be written. After the PEC is transmitted (as indicated by a TXDONE interrupt) clear XPEC by firmware.

Timeouts and Clock Stretching

At the end of each transmitted or received byte the MAX1781 stretches the clock to give the CPU time to process the interrupts and decide what to transmit or receive next. After the TXDONE or RXDONE interrupt is processed the firmware must clear the SCLSTRETCH bit to release the SCL line. If this is done too late a timeout interrupt is generated.

A timeout interrupt is generated and the SCL/SDA lines are released if one of the timeout conditions is detected. A $T_{LOW:MEXT}$ timeout occur when the MAX1781 is configured as a master and a transmitted or received byte takes longer than 10ms. A $T_{LOW:SEXT}$ timeout occurs when the MAX1781 is configured as a slave and its cumulative clock stretching (including all clock stretching from START to STOP) is longer than 23.9ms. A $T_{TIMEOUT}$ timeout occurs when either side continuously holds the SCL line low for longer than 27.9ms.

If any timeout condition is detected in slave mode the SDA line is released before the SCL line is released. If a timeout condition is detected in master mode the MAX1781 transmits a NACK followed by a STOP.

Glitch Filter

The SMBus block includes a glitch filter to improve data transmission in a noisy environment. SCL and SDA pulses briefer than 1us are ignored when the glitch filter is enabled. Set GF (see SMBCFG1) to 1 to enable the glitch filter.

BAUD Rate Setting

The SMBus BAUD rate during master mode is generated using the internal instruction oscillator and a clock divider configured by MBAUD[2:0] in SMBCFG2. The glitch filter effects the resulting baud rate. See Table 12 for the effective BAUD rate for specific settings of MBAUD and GF. Note that the minimum and maximum values in the table account for the oscillator tolerance over the 0 to 85°C temperature range.

Table 12. Master Transmit BAUD Rate

MBAUD[2:0]	MASTER TRANSMIT FREQUENCY (kHz)					
	GF = 1			GF = 0		
BITS 7 TO 5	MIN	TYP	MAX	MIN	TYP	MAX
000	116.5	176.5	284.1	206.0	272.7	378.8
001	92.4	130.4	189.4	141.0	176.5	227.3
010	65.3	85.7	113.6	86.4	103.4	126.3
011 (Default)	41.2	50.8	63.1	48.7	56.6	66.9
100	23.7	28.0	33.4	26.0	29.7	34.4
101	12.8	14.8	17.2	13.5	15.2	17.5
110	6.7	7.6	8.7	6.9	7.7	8.8
111	3.4	3.9	4.4	3.4	3.9	4.4

SMBus Registers

SMBRDR—SMBus Received Data Register

(Read Only)
Address 0x00A

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	RDAT	0x00	Received Data Register. After each byte is received it is stored into SMBRDR and RXDONE is set. SMBRDR Holds manufacturer access op-code in the manufacturer access mode.

SMBXDR—SMBus Transmit Data Register

(Read/Write)
Address 0x00B

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	XDAT	0x00	Transmit Data Register. Write a byte to SMBXDR to transmit a byte in either master or slave mode. During manufacturer access this register holds the data received and is used to transmit data.

SMBCFG1—SMBus Configuration Register 1

(Read/Write)
Address 0x00C

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7- D6	MSMODE[1:0]	00	Master/Slave Mode Configuration. This bit configures the MAX1781 for master or slave mode. 00: Slave Mode. This is the default reset value. 01: Master Receive Mode. Stays in this mode after the byte has been received. 10: Master Transmit Mode with Arbitration.

SMBCFG1—SMBus Configuration Register 1

(Read/Write)
Address 0x00C

			11: Master Transmit Mode without Arbitration. Upon completion of a transaction (STOP condition or timeout occurs) or loss of arbitration, the SMBus is reset to slave mode (MSMODE[1:0] = 00)
D5	BUSBUSY	0	Indicate the SMBus status (read only). This bit is set by a START condition and cleared by a STOP condition. This bit is useful for determining if a master transaction can be started. If this bit is set the bus is busy and the transaction will be queued. 0: The SMBus is idle. 1: The SMBus is busy.
D4	ACKEN	1	0: ACK is not generated after receiving a byte. 1: Automatically generate ACK after receiving a byte. This bit has no effect when the MAX1781 is in transmit mode. Set this bit to 1 to allow the MAX1781 to respond to its address in slave mode.
D3	XSTOP	0	Set to 1 to queue a STOP condition at the end of the current transaction. This bit is reset to 0 when the STOP is generated. Do not write 1 to this bit when the MAX1781 is in slave mode.
D2	XSTART	0	Set to 1 to transmit a START condition when the MAX1781 is the active bus master. If another master owns the bus the START occurs when the bus becomes free. If the MAX1781 is currently the bus master a repeated START is generated. This bit is reset to 0 when the START is generated. Do not write 1 to this bit when the MAX1781 is in slave mode.
D1	Reserved	0	Reserved for Future Use. This bit is not used and not available.
D0	GF	1	Glitch Filter Configuration 0: Glitch Rejection disabled. 1: 1µs glitch rejection for both SDA and SCL pins.

SMBCFG2—SMBus Configuration Register 2

(Read/Write)
Address 0x00D

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7-D5	MBAUD[2:0]	011	Master Baud Rate Setting See Table 12, which accounts for glitch filter setting and instruction oscillator tolerance.
D4	SCLSTRETCH	0	SMBus Transceiver Clock Stretching: 0: Discontinue clock stretching 1: Clock is being stretched SCLSTRETCH is set to 1 by hardware after each byte transmitted or received, excluding PEC transmission. SCLSTRETCH must be cleared by firmware after each TXDONE and RXDONE interrupt.
D3	CPEC	0	Set to 1 to Clear SMBPDR. This bit always reads as 0. 0: No effect on PEC 1: Clears computed PEC to 0x00
D2	RPEC	0	Receive PEC byte and compare to computed PEC (SMBPDR). After PEC byte is received, RPEC must be reset to 0 by firmware. 0: No effect 1: Next byte received is compared to PEC.
D1	XPEC	0	Transmit a PEC byte (SMBPDR) through the shift register (after the current byte has been transmitted or received if a transaction is in progress). After the PEC byte is transmitted, XPEC must be reset to 0 by firmware. 0: PEC byte is not transmitted. 1: Queue current PEC for transmission.
D0	—	1	Uncommitted bit, useable as storage.

SMBST1—SMBus Status Register 1

(Read/Write)
Address 0x00E

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	NACK	0	NACK is set to 1 after a NACK is received. NACK is updated during the ACK/NACK cycle of the byte transmitted, and is valid until the next ACK/NACK cycle. NACK occurs after the RXDONE interrupt. Firmware can only reset the bit to zero.
D6	ACK	0	ACK is set to 1 after an ACK is received. ACK is updated during the ACK/NACK cycle of the byte transmitted, and is valid until the next ACK/NACK cycle. ACK occurs after the RXDONE interrupt. Firmware can only reset the bit to zero.
D5	SRWB	0	Indicate the slave operation mode (read only). 0: in slave receive mode (write command) 1: in slave transmit mode (read command) SRWB indicates if the bit in the address designates the command as a read or write operation. This is a read only bit. The bit is cleared when a stop condition is detected. A zero denotes the part is acting as a slave receiver, while one means the part is acting as a slave transmitter. This bit has no purpose in master mode.
D4	STOP	0	STOP indicates that a STOP condition has been detected on the SMBus. Firmware can only reset the bit to zero.
D3	RSTRT	0	RSTRT indicates that a Repeated START condition has been detected on the SMBus. Firmware can only reset the bit to zero.
D2	START	0	START indicates that a START condition has been detected on the SMBus. Firmware can only reset the bit to zero.
D1-D0	SADRMATCH[1:0]	00	Slave Address Match: 00: No address match 01: Primary Slave address match (SMBBPSADR) 10: Secondary Slave address match (SMBSSADR) 11: General-call (broadcast address 0x00) match SADRMATCH indicates that one of the slave addresses has been detected. Firmware can only reset the bits to zero.

SMBST2—SMBus Status Register 2

(Read/Write)
Address 0x00F

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	RXDONE	0	Receive of a data byte is complete. RXDONE is set after the LSB of a byte has been received. Firmware can only reset the bit to zero.
D6	TXDONE	0	Transmit of a data byte is complete. TXDONE is set after the LSB of a byte has been transmitted. Firmware can only reset the bit to zero.
D5	PERR	0	PEC error received. PERR is set when the PEC received does not match the PEC computed by the MAX1781. Firmware can only reset the bit to zero.
D4	ANERR	0	NACK is detected during the ACK/NACK cycle following a byte transmission. After the NACK is detected ANERR is set to 1. Firmware can only reset the bit to zero.
D3	MFGACCMD	0	Manufacturer access entry (0x00) received. MFGACCMD is set during receive when the byte immediately following the address is 0x00. Firmware can only reset the bit to zero.
D2-D1	TIMEOUT[1:0]	00	TIMEOUT condition detected. 00: No timeout has occurred. 01: $T_{LOW:SEXT}$ condition detected in the slave mode. 10: $T_{LOW:MEXT}$ condition detected in the master mode. 11: $T_{TIMEOUT}$ condition detected in either master or slave mode.
D0	MARBLOSS	0	Master Arbitration Loss. MARBLOSS is set when arbitration (byte received is not equal to byte transmitted) is lost while transmitting as a bus master. Firmware can only reset the bit to zero.

SMBEI—SMBus Interrupt Enable Register

(Read/Write)
Address 0x0D0

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7	EIMFGACC	0	MACC Command Interrupt Enable. Set to 1 to enable the MFGACCMD interrupt on receiving a manufacturer access entry byte (0x00). Set to 0 to disable the MFGACCMD interrupt; the MFGACCMD status bit will continue to operate.
D6	EIRXDONE	0	Receive Interrupt Enable. Set to 1 to enable the RXDONE interrupt when byte reception is complete. Set to 0 to disable this interrupt; the RXDONE status bit will continue to function.
D5	EITXDONE	0	Transmit Interrupt Enable. Set to 1 to enable the TXDONE interrupt when byte transmission is complete. Set to 0 to disable this interrupt; the TXDONE status bit will continue to function.
D4	EISP	0	START and STOP Interrupt Enable. Set to 1 to enable the START, Repeated START, or STOP interrupts. Set to 0 to disable these interrupts; the status bits will continue to function.
D3	EIADRMTCH	0	Slave Address Interrupt Enable. Set to 1 to enable the SADRMATCH interrupt on primary or secondary slave ID or general-call (0x00) match; the SADRMATCH status bits will continue to function.
D2	EITOUT	0	Timeout Interrupt Enable. Set to 1 to enable interrupts for $T_{TIMEOUT}$, $T_{LOW:SEXT}$, and $T_{LOW:MEXT}$ timeout conditions. Set to 0 to disable these interrupts; the TIMEOUT status bits will continue to function.
D1	EIARL	0	Master Arbitration Loss Interrupt Enable. Set to 1 to enable the MARBLOSS interrupt. Set to 0 to disable this interrupt; the MARBLOSS status bit continues to function.
D0	EIPEC	0	PEC Error (CRC-8) Interrupt Enable. Set to 1 to enable the PERR interrupt. Set to 0 to disable this interrupt; the PERR status bit continues to function.

SMBPSADR—SMBus Primary Slave Address

(Read/Write)
Address 0x0D1

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D1	SMBPADR	0001011	Primary Slave SMBus ID (address). The MAX1781 will respond to SMBus requests at this address unless ACKEN=0.
D0	Reserved	0	Reserved for Future Use. This bit is not used and not available.

SMBSSADR—SMBus Secondary Slave Address

(Read/Write)
Address 0x0D2

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7-D0	SMBSADR	0001011	Secondary Slave SMBus ID (address). The MAX1781 will respond to SMBus requests at this address unless ACKEN=0.
D0	Reserved	0	Reserved for Future Use. This bit is not used and not available.

SMBPDR—SMBus PEC

(Read Only)
Address 0x0D3

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	PEC	0x00	Computed PEC (CRC-8). Stores the current computation for the PEC.

SMBMACR—SMBus Manufacturer Access Command

(Read Only)
Address 0x0D4

BIT	NAME	POR	FUNCTION/DESCRIPTION
D7–D0	MACC	0x00	Manufacturer Access Command op-code received over the SMBus. Bit 7 indicates a block mode transfer. See *Manufacturer Access Commands*.

Access and Security

The MAX1781 provides a service mode to allow programming and aid product development. Service mode allows read/write access of data memory, data and program EEPROM, and is useful for troubleshooting. This access is achieved via Manufacturer Access Commands (MACC) over the SMBus or through a parallel programming interface using PORTA[7:0] and HV[3:0]. The MAX1781 uses service mode and lock protection to provide the following system security:

- Read/write/clear protection from malicious software that may have access to the SMBus.
- Read/write code protection to prevent reverse engineering of algorithms.
- Block-Clear to prevent accidental lockout during development.

Access is summarized in Table 13 and is dependent on Service/Normal Mode, and the LOCK and SDBGEN bits in ISPDBGAC. Service Mode, LOCK, and SDBGEN are described in *Locking/Unlocking the MAX1781* and *Service/Normal Mode.*

Table 13. Security Based on Operating Modes

MODE	ISPDBGAC		MANUFACTURER ACCESS COMMANDS OVER SMBUS				PARALLEL PROGRAMMING
	LOCK	SDBGEN	EEPROM BLOCK CLEAR (ISPERPM)	RAM/EEPROM R/W	ISPDBGCFG	TMCFGSTA	
Service	1	X	Yes	No	Yes	No	
	0	X		Yes		Yes	
Normal	1	X	No[1,2]	No[1,2]	No[1]	No	
	X	0					
	0	1		Yes	Yes		

[1] *Pass SMBus command to firmware if LOCK = 1. If LOCK = 0 SMBus MACC commands are not forwarded to firmware.*

[2] *Ignore (NACK) if LOCK = 0.*

If CPU is frozen SMBus commands are not passed to Firmware.

Locking/Unlocking the MAX1781

Special SMBus commands called manufacturer access commands (MACC, see the *Manufacturer Access Commands* section) provide access to data memory, program and data EEPROM, and special security registers ISPDBGAC, ISPDBGCFG, and TMCFGSTA.

Lock the MAX1781 by setting LOCK = 1 in the ISPDBGAC register. Program ISPDBGAC as any other data EEPROM (see the *EEPROM Memory* section). When the MAX1781 is locked data and program memory are inaccessible externally over either SMBus or the parallel programming interface. Once the part is locked it cannot be unlocked except through firmware (by modifying ISPDBGAC) or the execution of the block-clear ISPERPM MACC command. In addition to unlocking the part, ISPERPM sets ISPDBGAC to its default state and clears all program EEPROM, keeping the code secure.

All RAM/EEPROM access requires that the instruction clock is active (the part is not in sleep mode) and the CPU is frozen (set ISPCPUFREEZE = 1 with MACC command ISPDBGCFG, op-code 0x40).

Service/Normal Mode

On an unlocked part, program/data memory read/write/clear access is provided through MACC commands in either normal or service mode. To put the MAX1781 in service mode, pull SCI below GND with 2mA to 3mA and reset the part as shown in *Figure 19*. Block-clear (ISPERPM) is available only in service mode so malicious

software cannot clear the part without also having access to the SCI pin, which is not available to the host in a normal smart battery system. The MAX1781 is initially frozen (ISPCPUFREEZE = 1) immediately after being reset into service mode. This is required to enable memory access in service mode. The CPU may be unfrozen by setting ISPCPUFREEZE=0 in ISPDBGCFG using MACC command 0x40.

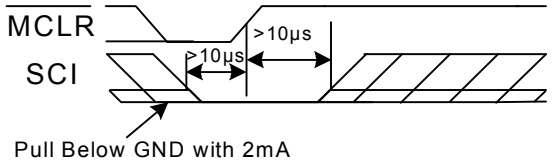


Figure 19. Putting the MAX1781 in Service Mode

Manufacturer Access Commands

The MAX1781 supports Manufacturers Access Commands (MACC) to assist firmware development by providing memory and peripheral access via the SMBus interface. This MACC hardware enables the developer to: program the MAX1781 program and data EEPROM, access the analog peripherals, and inspect the results of firmware by reading data memory. The MAX1781 also includes security to lock out this access before the product is shipped. See *Access and Security* for details on security.

Table 14 summarizes the manufacturer access commands supported by the MAX1781 hardware. Unlisted MACC commands are passed to firmware, allowing expansion of the MACC command set.

Table 14. MAX1781 MACC Command Support

MACC op-code	MACC Command
0x20	TMCFGSTA (to enable parallel programming)
0x40	ISPDBGCFG (to freeze for memory read/write or for parallel programming mode)
0x70	ISPERPM Block Clear
0x00	Data Memory Read/Write Byte
0x80-0xFF	Block Data Memory Read/Write

- **ISPERPM Block Clear Command (0x70)**

Block Clear. This command clears the entire program EEPROM and resets LOCK=0 and SDBGEN=1. This command is write only.

MACC ISPERPM Block Clear

S	Device Address + W	A	MACC entry 0x00	A	MACC op-code 0x70	A	PEC	A	P

- **MACC Write/Read Byte Command for TMCFGSTA (0x20) or ISPDBGCFG (0x40)**

This command is used for reading/writing TMCFGSTA (for setting parallel programming mode) or ISPDBGCFG. The op-code is 0x20 for TMCFGSTA or 0x40 for ISPDBGCFG.

MACC write to TMCFGSTA or ISPDBGCFG.

S	Device Address + W	A	MACC entry 0x00	A	MACC op-code	A	Data	A	PEC	A	P

MACC read from TMCFGSTA or ISPDBGCFG

S	Device Address + W	A	MACC entry 0x00	A	MACC op-code	A	Sr
	Device Address + R	A	Data	A	PEC	Ã	P

- **MACC Write/Read Byte Command for DM (Data Memory) Register MACC Command (0x00)**

This command allows read/write access to data memory or peripherals. This command will only function when the part is not in sleep mode and the CPU is frozen (set ISPCPUFREEZE = 1 with MACC command ISPDBGCFG, op-code 0x40). MACC memory access also requires that the MAX1781 is not in parallel programming mode (PARPROG=0 in TMCFGSTA, see *Parallel Programming Mode.*)

Write Memory Byte Format:

S	Device Address + W	A	MACC entry 0x00	A	MACC op-code 0x00	A	FSR	A	DM address	A
	Data	A	PEC	A	P					

Read Memory Byte Format:

S	Device Address + W	A	MACC entry 0x00	A	MACC op-code 0x00	A	FSR	A	DM address	A	Sr
	Device Address + R	A	Data	A	PEC	Ã	P				

- **MACC Write/Read Block Command for DM (Data Memory) Register (0x80 – 0xFF)**

This command allows block read/write access to data memory or peripherals. The lower 7 bits of the op-code (N) determine the length of the block transmission. The MSB is always 1. Use 0x80 to read/write 128 bytes. The operand specifies the start of the data memory address followed by the actual received/transmitted data. This command will only function when the instruction clock is active (the part is not in sleep mode) and the CPU is frozen (set ISPCPUFREEZE = 1 with MACC command ISPDBGCFG, op-code 0x40). MACC block memory access also requires that the MAX1781 is not in parallel programming mode (PARPROG=0 in TMCFGSTA, see *Parallel Programming Mode.)*

Write Memory Block Format:

S	Device Address + W	A	MACC entry 0x00	A	Command N (# of Data)	A	FSR	A	DM address	A
	Data 1	A	Data 2	A	…	A	Data N	A	PEC	A	P

Read Memory Block Format:

S	Device Address + W	A	MACC entry 0x00	A	Command N (# of Data)	A	FSR	A	DM address	A	Sr
	Device Address + R	A	Data 1	A	Data 2	A	…	A	Data N	A	PEC	Ã	P

Legend:

- S: Start Condition
- Sr: Repeated Start Condition
- A: Acknowledge Condition
- Ã: Non-Acknowledge Condition
- P: Stop Condition
- W: Write Bit
- R: Read Bit
- Packet Error Code (PEC) calculated by the hardware
- Master Transmitting
- Master Receiving

Parallel Programming Mode

The MAX1781 provides a parallel programming mode to dramatically reduce programming time. Parallel programming mode is only available on an unlocked part in service mode when PARPROG = CPUFREEZE = 1 (in TMCFGSTA), and PROGMODE = 1 (in ISPDBGCFG). TMCFGSTA is write-able only through SMBus MACC command 0x20 and only when in service mode. TMCFGSTA is read only by firmware.

Any data memory address can be accessed from within parallel programming mode. Parallel programming mode uses PORTA for high-address-byte, low-address-byte, and data, and HV0–HV3 for strobing the address and data in and out of the part. The MAX1781 latches in the high byte of the address when HV2 (HILOB) is high during a falling edge of HV3 (ALE), and latches the low address byte when HV2 is low during the falling edge of HV3. Data is latched in on a falling edge of HV0 (WR) and latched out on a falling edge of HV1 (RD).

To read and write program memory in parallel programming mode simply access MCFG1, MADRL, MADRH, MDATL, and MDATH as explained in the *EEPROM Memory* section of this datasheet.

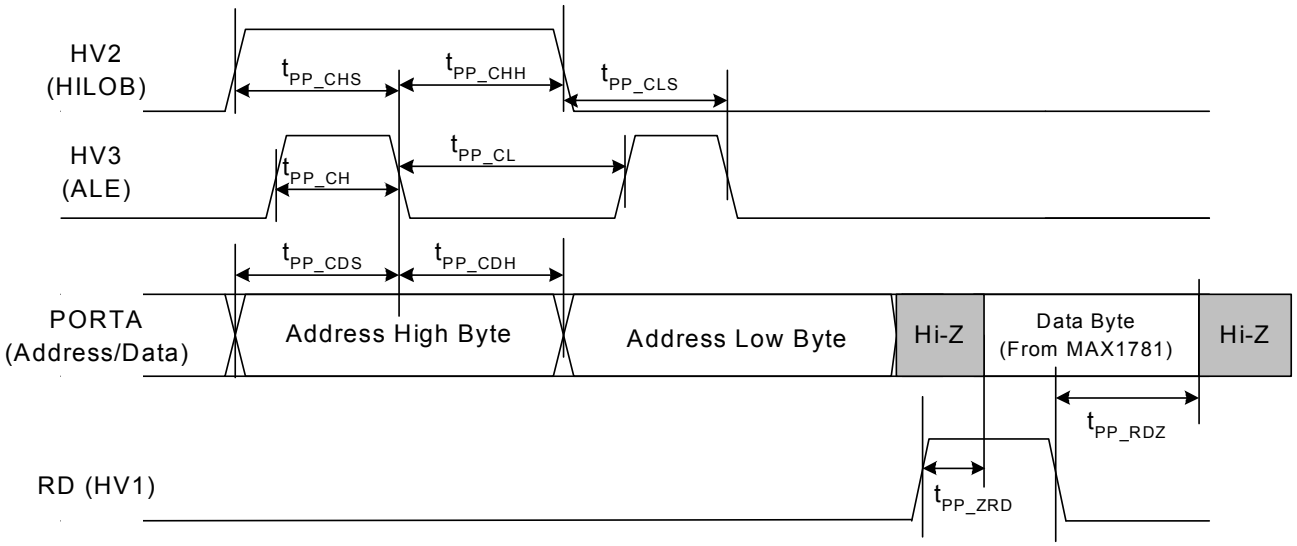


Figure 20. Parallel Programming Read Data Memory Byte Timing Diagram

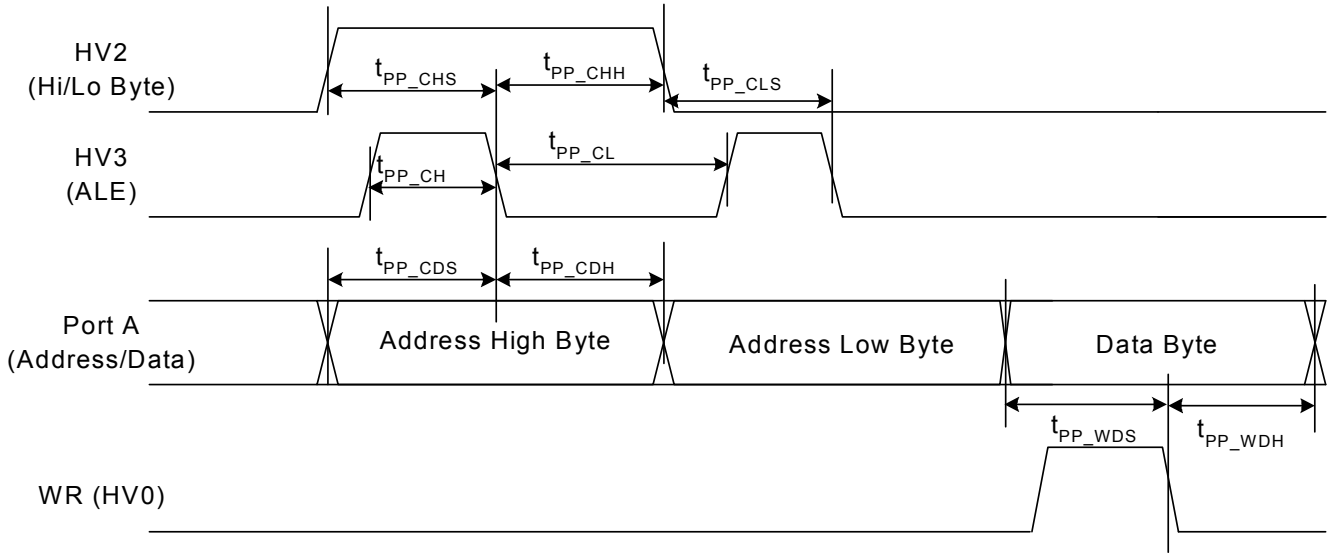


Figure 21. Parallel Programming Write Data Memory Byte Timing Diagram

Access and Security Registers

TMCFGSTA—Parallel Programming Register

(Writeable from SMBus MACC 0x20 only)
(Read only by firmware) Address 0xFE

BIT	NAME	POR	DESCRIPTION
D7–D4	Factory Reserved	0	Write 0 to these locations.
D3	CPUFREEZE	0	Set this bit to 1 to freeze the CPU in parallel programming mode. Set to 0 for all other modes.
D2	Factory Reserved	0	Write 0 to this location.
D1	PARPROG	0	Set to 1 to program the program EEPROM using parallel programming mode. This uses PORTA and HV pins.
D0	Factory Reserved	0	Write 0 to this location.

ISPDBGAC—In System Programming Debug Access Register

(**Data EEPROM**, see *EEPROM Memory*)
Address 0xFC

BIT	NAME	DEFAULT	DESCRIPTION
D7	Reserved	0	Write 0. Do not write a 1 to this bit.
D6	Unused	0	Unused. Read as 0.
D5	Unused	0	Unused. Read as 0.
D4	Unused	0	Unused. Read as 0.
D3	Reserved	0	Write 0. Do not write a 1 to this bit.
D2	LOCK	0	Set to 1 to disable RAM, program EEPROM, and data EEPROM access through SMBus MACC commands.
D1	SDBGEN	1	Set to 1 to allow RAM, program EEPROM, and data EEPROM access through SMBus MACC commands while in normal mode.
D0	Reserved	1	Write 1 to this location.

ISPDBGCFG—In System Programming Debug Configuration Register

SMBus MACC 0x40
Address 0xFD

BIT	NAME	POR	DESCRIPTION
D7	Reserved	0	Reserved
D6-D4	Reserved for ICE operation	000	Reserved for ICE operation. Write 0 to these bits.
D3	Reserved	0	Reserved — SMBus R/W
D2	ISPCPUFREEZE	0	0 = Run — SMBus R/W 1 = Freeze — Firmware Readable Write 1 to request freeze the CPU. If the CPU is frozen ISPCPUFREEZE=1 when read. This bit cannot be written to when the CPU is in sleep mode.
D1	PROGEXITMODE	0	0 = No Reset after program memory EEPROM write/clear is complete. — SMBus R/W 1 = Reset part after program memory EEPROM write/clear is complete — Firmware Readable
D0	PROGMODE	0	Set ProgMode to 1 to enable parallel programming mode. — SMBus R/W, Firmware Readable

Design and Applications Information

ESD and Short Protection on the Battery Pins

The Circuit of *Figure 22* includes optional resistors (R26-R29) and capacitors (C7-C10) which provide dual functionality. Since B1P-B4P and BN are often exposed during manufacturing these extra components can be used to provide extra ESD protection. For ESD purposes 0.1µF are typically adequate. Additionally the resistors protect against a single point failure in the MAX1781 producing a short on a cell.

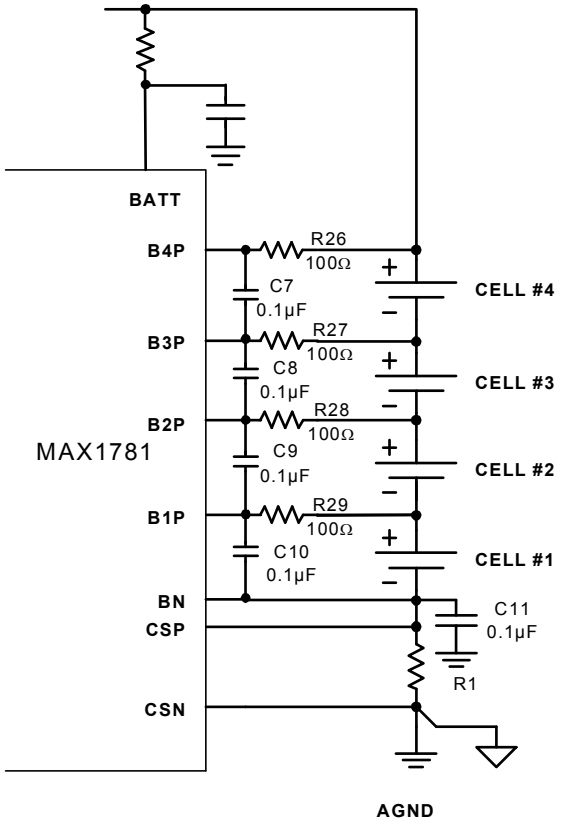


Figure 22. ESD and Battery Pin Short Protection

Setting Over-Discharge Current and Short-Circuit Current Thresholds

Over-discharge and short-circuit protection are configured using resistors R1 through R4 as shown in Figure 23. An over-current event occurs when ODI or SCI goes below ground for longer than the ODI/SCI blanking time.

The over-discharge current threshold is:

$$I_{OD} = \frac{V_{AA}}{R3} \cdot \frac{R4}{R_{SENSE}}$$

An over-discharge event only occurs when I_{OD} is persistently exceeded for over the over-discharge delay (adjustable from 0 to 14.5ms in ODSCDLY.)

The short-circuit current threshold is:

$$I_{SC} > \frac{V_{AA}}{R1} \cdot \frac{R2}{R_{SENSE}}$$

A short-circuit event occurs when the discharge current continuously exceeds I_{SC} for over the short-circuit delay (adjustable from 0 to 1ms in ODSCDLY.)

The accuracy of these thresholds is a function of V_{AA}, R_{SENSE}, R1, R2, R3, and R4, with the majority of the error typically due to V_{AA} variation (+/-6.25%). Choose R1, R2, R3, and R4 to be smaller than 600kΩ to ensure that ODI and SCI bias current does not contribute significant error.

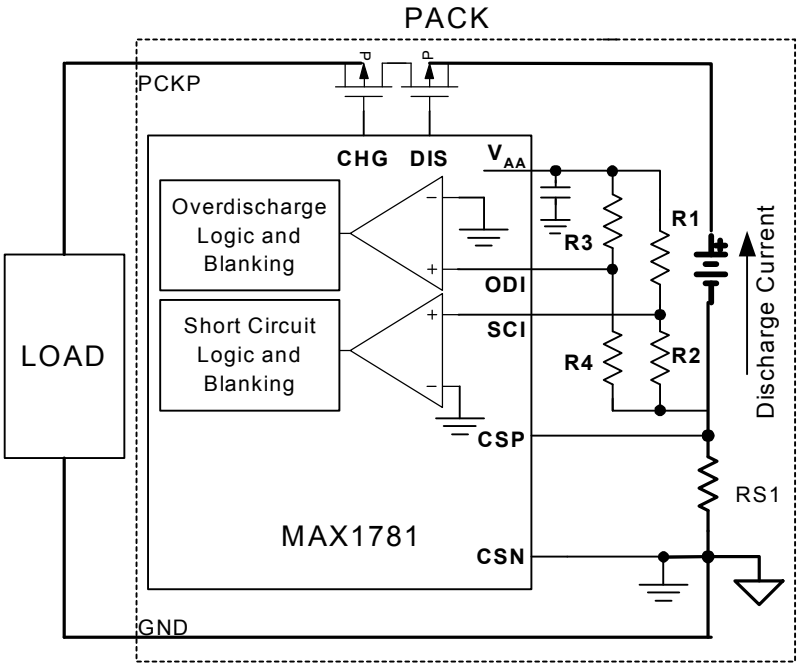


Figure 23. Overcharge And Short Circuit Configuration

The short-circuit threshold is typically set higher than the over-discharge threshold, with a longer timer configuration for the over-discharge threshold. This allows the MAX1781 to allow moderately heavy pulsed loads to persist while preventing continuous heavy loads and quickly reacting to extreme loading, such as a short-circuit. Set the over-discharge threshold according to the maximum continuous discharge current allowed by the system. Set the over-discharge delay to be long enough that an over-discharge event is not accidentally triggered by system inrush current or pulsed loading by the system. Set the short-circuit threshold to prevent collapse of the V_{AA} regulator and to prevent destruction of the DIS and CHG MOSFETs. Set the short-circuit delay to be long

enough to prevent accidental accidental short-circuit events during the inrush current upon normal battery connection. Set the short-circuit delay to be shorter than the time that the pack can safely sustain the short-circuit current.

Improving Fuel Gauge Measurement Accuracy

Layout is important to maintain the MAX1781's high fuel gauge accuracy. Figure 24 demonstrates the proper layout for the current sense traces and filter. Keep current sense traces close to each other and do not share the current sense branches with other circuitry or pins. Do not use the current sense traces for the star ground connection. Give special consideration to the CSN trace. Board layout software will typically connect any via's in the trace to the ground plane. Insure that via's on the trace do not connect to the ground plane.

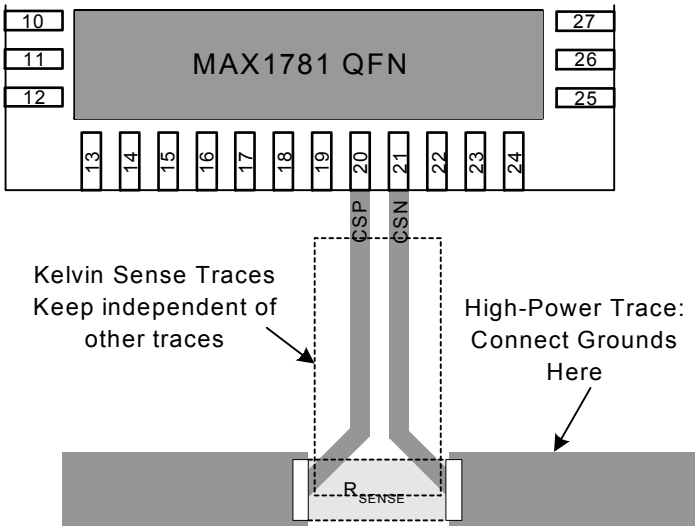


Figure 24. Layout Recommendation for Current Sense Inputs

R2 and C1 provide filtering for the fuel gauge and improve fuel gauge accuracy in a noisy environment. A resistance of 51Ω and a capacitance of 0.1µF significantly reduces the noise from CSP to CSN and is adequate in many applications.

Startup with Under-Voltage Cells

When the battery is over-discharged firmware will typically turn the discharge MOSFET off to prevent further power consumption. Firmware may continue to hibernate in sleep mode until a charger is applied. If a charger is not applied for a long enough time the firmware may also choose to return to its 0µA shutdown shipping state, by setting BATTON to 1. In this state the MAX1781 is completely shut down and V_{AA} is powered down. When a charger is applied to PCKP, V_{AA} begins regulating and the MAX1781 is reset, with the discharge MOSFET remaining off. Upon reset the firmware should check for battery undervoltage and decide whether to trickle charge the battery or allow full-charge. If the battery voltage is too low to supply the MAX1781 the circuit shown in Figure 25 may be used to provide the initial trickle charge for the battery. This circuit also provides protection from attempting to fast charge a damaged cell.

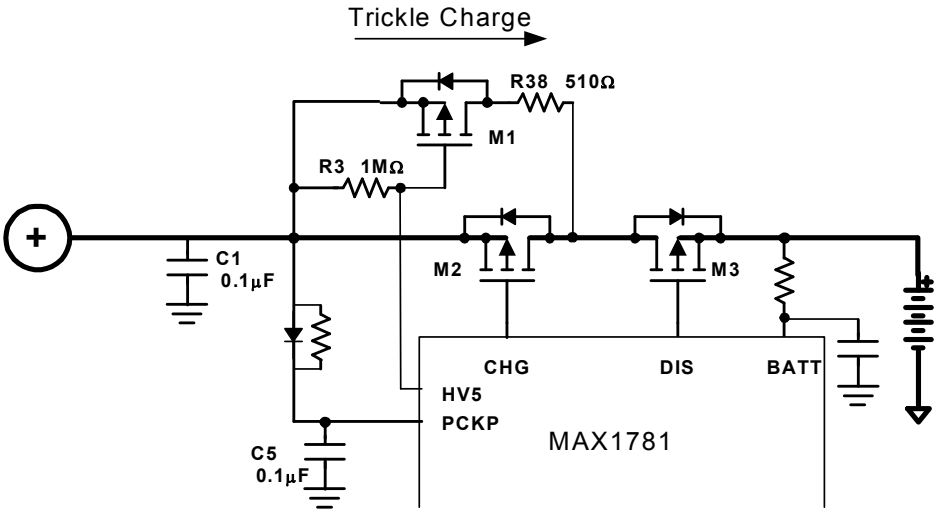


Figure 25. Trickle Charge Circuit

Chip Information

Transistor Count: 202,241
Substrate connected to AGND

Appendix A: Instruction Set Manual

The following instructions were written in a format compatible with the inline assembler included in the Maxim Integrated Development Environment.

The following notation is used in this instruction manual:

( ) In the "CYCLES" column this indicates the number of instruction cycles when a branching instruction changes the Program Counter.
f This represents any 6-bit addressable memory location (direct-addressed, see Data Memory Map.)
[**f**] Used for indirect memory access. This represents any 6-bit directly addressable memory location that points to an 8-bit addressable memory location.
r This represents any 8-bit addressable memory location.
d This indicates whether the result is stored in register **f** (if **d** is 1) or **W** (if **d** is 0)
k This is any literal 8-bit number
b This represents a number from 0 to 7 which indicates which bit is acted on, with 7 being the MSB and 0 being the LSB.
m This is a 13-bit program memory address.

ADDWF – Add W and F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
ADDWF f, d	Add contents of W and f	1	0001,0000,0dff,ffff	C, DC, Z
ADDWF [f], d	Add contents of W and [f]	2	0001,0000,1dff,ffff	C, DC, Z

The contents of **W** and direct or indirect variable **f** are added together and stored in either **W** or **f** (depending on **d**). The zero flag is set if the result is zero. The carry flag is set if the operation requires a borrow or carry. The half carry flag is set if a borrow or carry was applied to bit 4.

Example

```
j=j*5          ;often needed in Binary-Coded-Decimal conversions
      #define j        0x20

      MOVFW j
      ADDWF W,0   ;W=j*2
      ADDWF W,0   ;W=j*4
      ADDWF j,1   ;j=j*4+j
```

ADDLW – Add Literal to W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
ADDLW k	Add literal with W	1	0000,1100,kkkk,kkkk	C, DC, Z

W is added to literal k and the result is stored in **W**. The zero flag is set if the result is zero. The carry flag is set if the operation requires a borrow or carry. The half carry flag is set if a borrow or carry was applied to bit 5.

Example

```
W=W+0x5A
      ADDLW 0x5A ;
```

ANDLW – And Literal to W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
ANDLW k	AND literal with W	1	0000,1000,kkkk,kkkk	Z

W is anded to literal k and the result is stored in W. The zero flag is set if the result is zero.

Example

W=W and 0xA5

```
ANDLW 0xA5 ;                    0000,1000,1010,0101
```

ANDWF – AND W with F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
ANDWF f, d	AND W with f	1	0001,0001,0dff,ffff	Z
ANDWF [f], d	AND W with [f]	2	0001,0001,1dff,ffff	Z

The contents of **W** and direct or indirect variable **f** are anded together and stored in either **W** or **f** (depending on **d**). The zero flag is set if the result is zero.

Example

Call subroutine if bits 3, 5, or 7 on port A are high

```
#define W 0x00
#define PAIN 0xd7
#define Pt1 0x18
#define FSR 0x04

MOVLW PAIN ;
MOVWF Pt1  ; Pt1 now points to Port A
MOVLW 0xA4 ; for masking out bit 7,5, or 3
CLRF  FSR  ;
ANDWF [Pt1],0  ;
SKIPZ W    ;
CALL  SUB1 ;
; continue program here

SUB1:
; subroutine code here
RETURN     ;
```

BCF – Bit Clear

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
BCF f, b	Bit Clear f	1	0011,000b,bbff,ffff	None
BCF [f], b	Bit Clear [f]	2	0011,001b,bbff,ffff	None

Bit **b** of direct or indirect variable **f** is cleared. No status flags are modified.

Example

Clear ADC interrupt status

```
#define IS2 0x08
BCF  IS2.3 ;ADC status is bit 3
```

BSF – Bit Set

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
BSF f, b	Bit Set f	1	0011,010b,bbff,ffff	none
BSF [f], b	Bit Set [f]	2	0011,011b,bbff,ffff	none

Bit **b** of direct or indirect variable **f** is set. No status flags are modified.

Example

Initiate ADC conversion

```
#define ADCCFG 0x11
BSF  ADCCFG.0;ADC status is bit 0
```

BTFSC – Bit Test F, Skip If Clear

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
BTFSC f, b	Bit test f, skip if clear	1 (2)	0011,100b,bbff,ffff	none
BTFSC [f], b	Bit test [f], skip if clear	2 (3)	0011,101b,bbff,ffff	None

Tests bit **b** of direct or indirect variable **f**. The following instruction is skipped if the bit is clear. No status flags are modified.

Example

Switch-Case statement for interrupt handling routine

```
#define IS1 0x06

ISR:       ;interrupt service routine
BTFSC IS1.0;
GOTO  StackOverFlow
     ;it's important not to "CALL"
ISR_RET1:
BTFSC IS1.1;
GOTO  SoftwareTrap;
ISR_RET2:
BTFSC IS1.2;
GOTO  PortA_ISR;
;etc etc.
```

BTFSS – Bit Test F, Skip If Set

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
BTFSS f, b	Bit test f, skip if set	1 (2)	0011,110b,bbff,ffff	none
BTFSS [f], b	Bit test [f], skip if set	2 (3)	0011,111b,bbff,ffff	none

Tests bit **b** of direct or indirect variable **f**. The following instruction is skipped if the bit is set. No status flags are modified.

Example

Extending 8 bit addition to 16 bit addition. (A1,A0)=(A1,A0)+(B1,B0)

```
#define A0       0x80  ;A1 is 0x81
#define A1       0x81  ;A1 is 0x81
#define B0       0x82  ;B1 is 0x83
#define B1       0x83  ;B1 is 0x83
```

```
      #define Op1      0x18 ;operand 1
      #define Op2      0x19 ;operand 2
      #define W        0x00
      #define STATUS   0x01

      MOVLW A0   ;
      MOVWF Op1  ; Op1 now points to A
      MOVLW B0   ;
      MOVWF Op2  ; Op2 now points to B
      CALL  AddShort  ;
      ;continue program here

AddShort:
      MOVFW [Op2];
      ADDWF [Op1],1;A0=A0+B0
      BTFSS STATUS.2  ;check for carry
      GOTO  DoCarry   ;
      INCF  Op1,1     ;
RetFromCarry:
      INCF  Op2,1     ;
      MOVFW [Op2]     ;
      ADDWF [Op1],1   ;
      RETURN          ;
DoCarry:
      INCF  Op1,1     ;
      INCF  [Op1],1   ;
      GOTO  RetFromCarry;
```

CALL – Call Subroutine

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
CALL m	Call subroutine	2	011m,mmmm,mmmm,mmmm	none

Calls the subroutine located at memory location **m**. The current program counter is pushed onto the stack. A **CALL** should always be followed by a **RETURN**. Although the stack is 8 levels deep subroutines should not be used too carelessly. Use subroutines for functions which are general purpose. No Status flags are modified.

Example

Get the adjusted coulomb count

```
CALL GetCoulombs
```

CLRF – Clear F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
CLRF f	Clear f	1	0001,0010,01ff,ffff	Z
CLRF [f]	Clear f	2	0001,0010,11ff,ffff	Z

Clears direct or indirect variable **f**. This instruction sets the zero flag.

Example

Set the FSR to 0

```
#define FSR      0x04
CLRF FSR
```

COMF – 1's Complement F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
COMF f, d	1's Complement f	1	0001,0011,0dff,ffff	Z
COMF [f], d	1's Complement [f]	2	0001,0011,1dff,ffff	Z

Inverts direct or indirect variable **f** and stores the result in either **f** or **W**. This instruction sets the zero flag.

Example

2's complement by inverting and adding 1. X=-X

```
#define X  0x20
COMF X,1
INCF X,1
```

DECF – Decrement F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
DECF f, d	Decrement f	1	0001,0100,0dff,ffff	Z
DECF [f], d	Decrement [f]	2	0001,0100,1dff,ffff	Z

Decrements direct or indirect variable **f** and stores the result in either **f** or **W**. If the result is zero, the zero flag is set.

Example

Assembly translation of C code: for(x=32;x>=0;--x){;}

```
      #define X 0x20
      MOVLW      32
Loop:
      DECF       X,1
      ;do whatever
      SKIPZ      X
      GOTO       Loop
      ;done with loop
```

DECFSZ – Decrement F, Skip If Zero

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
DECFSZ f, d	Decrement f, skip if zero	1	0001,0101,0dff,ffff	Z
DECFSZ [f], d	Decrement f, skip if zero	2	0001,0101,1dff,ffff	Z

Decrements direct or indirect variable **f** and stores the result in either **f** or **W**. If the result is zero, the next instruction is skipped.

Example

Assembly translation of C code: for(x=32;x>0;x--){;}

```
      #define X  0x20
      MOVLW      32
Loop:
```

```
;do whatever
DECFSZ     X,1
GOTO       Loop
;done with loop
```

GOTO – Unconditional Branch

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
GOTO m	Unconditional Branch	1	010m,mmmm,mmmm,mmmm	-

Unconditional branch. Program execution continues at **m**. **GOTO** is often used following **SKIPZ**, **DECFSZ**, **INCFSZ**, **BTFSS**, and **BTFSC** to either jump to the conditional code or jump past the conditional code.

Example

If(x==0){;//conditional code}

```
#define X  0x20
SKIPZ X          ;
GOTO EndOfIf     ;
; do conditional code here
EndOfIf:
```

INCF – Increment F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
INCF f, d	Increment	1	0001,0110,0dff,ffff	Z
INCF [f], d	Increment	2	0001,0110,1dff,ffff	Z

Increments direct or indirect variable **f** and stores the result in either **f** or **W**. If the result is zero, the zero flag is set.

Example

A=A+B+1

```
#define A  0x20
#define B  0x21

INCF B,0   ;W=B+1
ADDWF A,1  ;A=A+W
```

INCFSZ – Increment F, Skip if Zero

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
INCFSZ f, d	Increment f, skip if zero	1 (2)	0001,0111,0dff,ffff	-
INCFSZ [f], d	Increment [f], skip if zero	2 (3)	0001,0111,1dff,ffff	-

Increments direct or indirect variable **f** and stores the result in either **f** or **W**. If the result is zero, the next instruction is skipped.

Example

Extended 16 bit increment: (A1,A0)=(A1,A0)+1

```
#define A0 0x20
```

```
#define A1 0x21

INCFSZ A0,1      ;A0++
GOTO   CarryDone;
INCF   A1,1      ;
CarryDone:
```

IORLW – Inclusive OR W With Literal

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
IORLW k	Inclusive OR W with k	1	0000,1001,kkkk,kkkk	Z

W and literal **k** are or'd together and the result is placed in **W**. If the result is zero, the zero flag is set.

Example

W=W or 7

```
IORLW 7     ;
```

IORWF – Inclusive OR W With F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
IORWF f, d	Inclusive OR W with f	1	0001,1001,0dff,ffff	Z
IORWF [f], d	Inclusive OR W with [f]	2	0001,1001,1dff,ffff	Z

W and direct or indirect variable **f** are or'd together and the result is placed in **W** or **f**. If the result is zero, the zero flag is set.

Example

X=X or 7

```
#define X   0x20
MOVLW 7    ;
IORWF X,1  ;
```

MOVFW – Move F into W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
MOVFW f	Move f into W .	1	0001,1000,01ff,ffff	Z
MOVFW [f]	Move [f] into W .	2	0001,1000,11ff,ffff	Z

The value of direct or indirect variable **f** is placed in **W**. If the result is zero, the zero flag is set.

Example

W=Pop(), using Software SP

```
#define SP  0x18;software Stack Pointer

Pop:
MOVFW  [SP];
INCF   SP,1;
RETURN     ;
```

MOVWF – Move W to F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
MOVWF f	Move W to f	1	0001,1011,00ff,ffff	none
MOVWF [f]	Move W to [f]	2	0001,1011,10ff,ffff	none

The value of **W** is placed in direct or indirect variable **f**. No flags are set by this operation.

Example

Push(W), using Software SP

```
     #define SP  0x18;software Stack Pointer

Push:
     DECF   SP,1;
     MOVWF  [SP];
     RETURN     ;
```

MOVLW – Move Literal to W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
MOVLW k	Move literal to W	1	0000,1010,kkkk,kkkk	none

Literal **k** is placed in **W**. No flags are set by this operation.

Example

X=0x17

```
     #define X   0x80
     #define W   0x00
     MOVLW  0x17;
     MOVFR  W,X ;
```

MOVFR – Move F to R

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
MOVFR f, r	Move f to r	2	11rr,rrrr,rrff,ffff	none

The value in the 6-bit direct addressed variable **f** is moved into the 8-bit addressed variable r. No flags are set by this operation.

Example

Saving static variable S in indirectly addressable space when the function call is over.

```
     #define S   0x80
     #define S_near 0x18
     #define Param1 0x19
     #define Param2 0x1a
     #define Param3 0x1b

BeginSub:
     CALL   POP   ;pop all parameters
     MOVWF  Param1 ;
     CALL   POP    ;
     MOVWF  Param2 ;
     CALL   POP    ;
```

```
MOVWF  Param3 ;
MOVRF  S,S_near;
;Subroutine body, which may modify S_near
MOVFR  S_near,S;
RETURN     ;
```

MOVRF – Move R to F

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
MOVRF r, f	Move r to f	2	10rr,rrrr,rrff,ffff	none

The value in the 8-bit addressed variable **r** is moved into the 6-bit direct addressed variable **f**. No flags are set by this operation.

Example

A=B-C

```
#define A   0x80
#define B   0x81
#define C   0x82
#define W   0x00
#define T1  0x18

MOVRF  C,0 ;
MOVRF  B,T1;
SUBWF  T1,0;
MOVFR  W,A ;
```

NOP – No Operation

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
NOP	No operation	1	0000,0001,0000,0001	None

No operation is performed.

RETFIE – Return From Interrupt

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
RETFIE	Return from interrupt	1	0000,0100,0000,0000	INTPROC

Interrupt execution is completed. The program counter is restored from the Program Counter Shadow Register. Generally, prior to **RETFIE** the program should restore **W** and **STATUS** to avoid effecting the operation of the main loop.

Example

```
;end of interrupt
#define Wbackup  0x80
#define Sbackup  0x81
#define FSRbackup     0x82
#define W        0x00
#define STATUS   0x01
```

```
MOVRF  Wbackup,0      ;
MOVRF  Sbackup,STATUS ;
MOVRF  FSRbackup,FSR  ;
RETFIE     ;
```

RETLW – Return, Place Literal in W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
RETLW k	Return, place Literal k in W	1	0000,0101,kkkk,kkkk	None

Return from subroutine. Literal **k** is placed The program counter is popped off of the 8-level stack. If the stack is empty a stack underflow is triggered. No status bits are effected.

Example

```
;return an error-code according to execution status
#define Errcode  0x30
BTFSC      errcode.0
RETLW      0;
BTFSC      errcode.1
RETLW      1;
BTFSC      errcode.2
RETLW      2;
BTFSC      errcode.3
RETLW      3;
```

RETURN – Return From Subroutine

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
RETURN	Return from subroutine	1	0000,0111,0000,0000	None

Return from subroutine. The program counter is popped off of the 8-level stack. If the stack is empty a stack underflow is triggered. No status bits are effected.

Example

```
#define RETVALUE 0x80
MOVRF RETVALUE,0
      ;Often return values are stored in W
RETURN     ;
```

RLF – Rotate Left Through Carry

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
RLF f, d	Rotate left f through carry	1	0001,1100,0dff,ffff	C
RLF [f], d	Rotate left [f] through carry	2	0001,1100,1dff,ffff	C

Rotate left through carry. Direct or indirect register **f** is rotated 1 bit to the right. The LSB is taken from the carry bit. The MSB rotates to the carry bit. No other status bits are effected.

Example

```
#define AH 0x30
#define AL 0x31
#define BH 0x32
#define BL 0x33
```

```
#define CH 0x34
#define CL 0x35
#define i  0x36

MULT_16B:  ; (AH,AL,BH,BL)=(AH,AL)*(CH,CL) 16-bit multiply
CLRF  BH
CLRF  BL
MOVLW 16   ;loop 16 times
MOVWF i

MULT1:
BCF   STATUS.STATUS_C
RLF   BL,1
RLF   BH,1
RLF   AL,1
RLF   AH,1
BTFSS STATUS.STATUS_C
GOTO  MULT2
MOVFW CL
ADDWF BL,1
MOVFW CH
BTFSC STATUS.STATUS_C
INCFSZ CH,0
ADDWF BH,1
BTFSC STATUS.STATUS_Z
MOVFW BL
BTFSS STATUS.STATUS_C
GOTO  MULT2
INCF  AL,1
BTFSC STATUS.STATUS_Z
INCF  AH,1

MULT2:
DECFSZ i,1
GOTO  MULT1
RETURN
```

RRF – Rotate Right Through Carry

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
RRF f, d	Rotate right f through carry	1	0001,1101,0dff,ffff	C
RRF [f], d	Rotate right [f] through carry	2	0001,1101,1dff,ffff	C

Rotate right through carry. Direct or indirect register **f** is rotated 1 bit to the right. The MSB is taken from the carry bit. The LSB rotates to the carry bit. No other status bits are effected.

Example

```
#define ACC_LO  0x30
#define ACC_HI  0x31
BinDivide: ; 16 bit divide of ACC_HI/ACC_LO by power of 2
BCF   STATUS,STATUS_C
RRF   ACC_LO,1
```

```
RRF   ACC_HI,1
DECFSZ W,0
GOTO  BinDivide
```

SKIPZ – Skip if Zero

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
SKIPZ f	Skip next instruction if f is zero	1 (2)	0010,0000,00ff,ffff	None
SKIPZ [f]	Skip next instruction if [f] is zero	2 (3)	0010,0000,10ff,ffff	None

Skip next instruction if direct/indirect register **f** is zero.

Example

```
;if(j==5)somefunc()

#define j  0x30

MOVLW 5
SUBWF j
SKIPZ W
CALL  somefunc
```

SLEEP – Enter Sleep Mode

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
SLEEP	Enter sleep mode	1	0000,0010,0000,0010	INTPROC

Program execution is halted until an enabled interrupt occurs (regardless of the state of GIE). Analog peripherals remain powered. The instruction oscillator is powered down.

Example

```
SLEEP
```

SUBWF – Subtract W from f

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
SUBWF f, d	Subtract W from f	1	0001,1111,0dff,ffff	C, DC, Z
SUBWF [f], d	Subtract W from [f]	2	0001,1111,1dff,ffff	C, DC, Z

W is subtracted from direct or indirect register **f** and the result is stored in either **W** or **f** (depending on **d**). If there is a carry the C bit is set, otherwise it is cleared. If there is a half carry the DC bit is set, otherwise it is cleared. If the result is zero, the zero flag is set, otherwise it is cleared.

Example

A=A-B+C

```
#define W   0x00
#define A   0x80
#define B   0x81
#define C   0x20
#define Pt1 0x18
#define Pt2 0x19
```

```
MOVLW A
MOVWF Pt1  ; Pt1 now points to A
MOVLW B
MOVWF Pt2  ; Pt2 now points to B
CLRF  W    ; W=0
ADDWF [Pt2],0; W=B
SUBWF C,0  ; W=C-B
ADDWF [Pt1],1; A=C-B+A
```

SUBLW – Subtract Literal from W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
SUBLW k	Subtract literal from W	1	0000,1101,kkkk,kkkk	C, DC, Z

Literal **k** is subtracted from **W** and the result is stored in **W**. If there is a carry the C bit is set, otherwise it is cleared. If there is a half carry the DC bit is set, otherwise it is cleared. If the result is zero, the zero flag is set, otherwise it is cleared.

Example

If(k<=23)somesub();

```
#define k  0x18

MOVFW k                                  0001,1000,0101,1000
SUBLW 24                                 0000,1101,0001,1000
BTFSS STATUS.2   ;skip if carry          0011,1100,1000,0001
CALL  somesub
```

SWAPF – Swap Nibbles

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
SWAPF f, d	Swap nibbles f	1	0001,1010,0dff,ffff	None
SWAPF [f], d	Swap nibbles [f]	2	0001,1010,1dff,ffff	None

Upper and lower nibbles of direct or indirect register **f** are swapped, and result is stored in either **f** or **W**. Status bits are unaffected.

Example

```
SWAPF W,0
```

TRAP – Software Interrupt

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
TRAP	Software Trap (Interrupt)	1	0000,0000,0000,0000	None

Interrupt is triggered.

Example

```
TRAP                                     0000,0000,0000,0000
```

XORLW – Exclusive Or Literal with W

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
XORLW k	Exclusive Or Literal with W	1	0000,1011,kkkk,kkkk	Z

Literal **k** is XOR'd with **W** and the result is stored in **W**. The zero flag is set if the result is zero, otherwise the zero flag is cleared.

Example

```
XORLW 0xAA
```

TBLRDW – Table Read Word

Mnemonic, Operand	Description	Cycles	Encoding	Status Affected
TBLRDW f	Table lookup instruction. Read into "W" the contents of table location in program memory at the address generated by concatenation of the contents of {(TBLSTRT), (f)}.	2	0010,1111,01ff,ffff	-

Program memory is read into **W**. The address accessed is determined by the concatenation of TBLSTRT and f as shown in the following table:

TBLSTRT								**Instruction parameter (f)**							
D7	D6	D5	D4	D3	D2	D1	D0	D7	D6	D5	D4	D3	D2	D1	D0
Not Used	**Program Memory Address**														
	PM12	PM11	PM10	PM9	PM8	PM7		PM6	PM5	PM4	PM3	PM2	PM1	PM0	Hi/Lo Byte

Although Program Memory is addressed in 16-bit words it can be accessed by the TBLRDW instruction only one byte at a time. The LSB of **f** determines whether MSByte or LSByte is read. If **f** is odd then the MSByte at the program memory address is read. If f is even then the LSByte at the program memory address is read. The STATUS register is not effected.

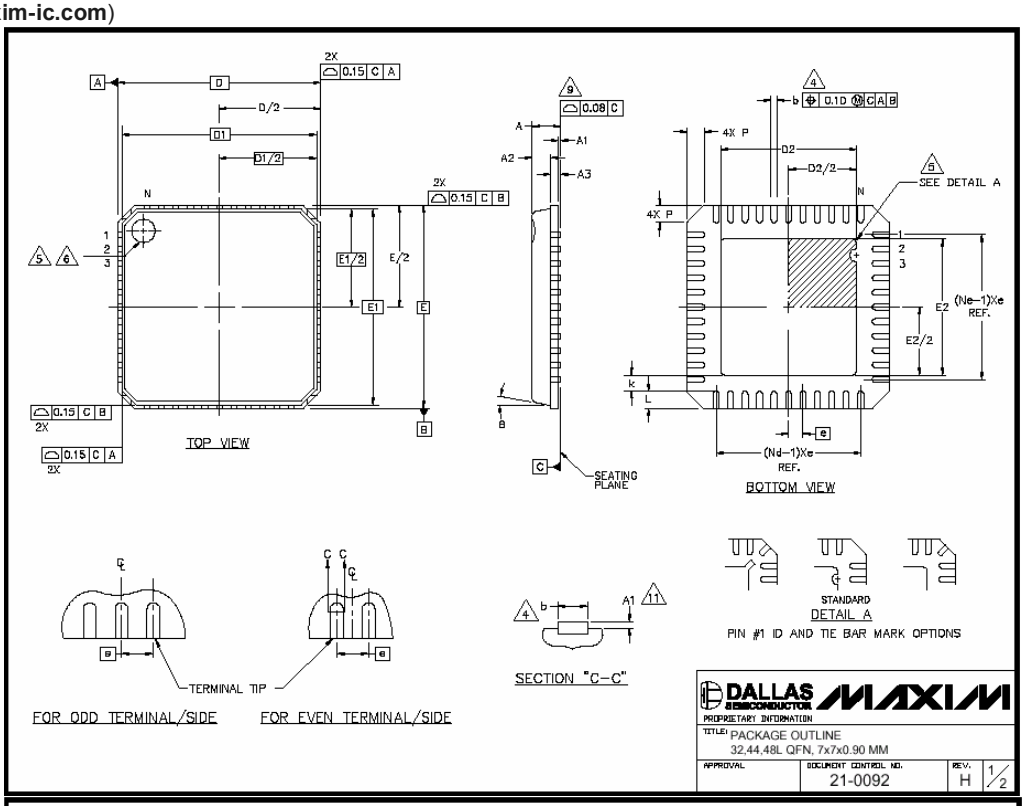
Example

Returns MSByte at 0x17AA

```
MOVLW 0x2E  ;
MOVWF TBLSTRT ;
MOVLW 0x55 ;0x2E55 reads 0x17AA
TBLRDW W
```

Package Information

(The package drawing(s) in this data sheet may not reflect the most current specifications. For the latest package outline information, go to **www.maxim-ic.com**)



COMMON DIMENSIONS									
PKG	32L 7x7			44L 7x7			48L 7x7		
SYMBOL	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
A	0.80	0.90	1.00	0.80	0.90	1.00	0.80	0.90	1.00
A1	0.00	0.01	0.05	0.00	0.01	0.05	0.00	0.01	0.05
A2	0.00	0.65	1.00	0.00	0.65	1.00	0.00	0.65	1.00
A3	0.20 REF			0.20 REF			0.20 REF		
b	0.23	0.28	0.35	0.18	0.23	0.30	0.18	0.23	0.30
D	6.90	7.00	7.10	6.90	7.00	7.10	6.90	7.00	7.10
D1	6.75 BSC			6.75 BSC			6.75 BSC		
E	6.90	7.00	7.10	6.90	7.00	7.10	6.90	7.00	7.10
E1	6.75 BSC			6.75 BSC			6.75 BSC		
e	0.65 BSC			0.50 BSC			0.50 BSC		
k	0.25	-	-	0.25	-	-	0.25	-	-
L	0.35	0.55	0.75	0.35	0.55	0.75	0.30	0.40	0.50
N	32			44			48		
Nd	8			11			12		
Ne	8			11			12		
P	0.00	0.42	0.60	0.00	0.42	0.60	0.00	0.42	0.60
θ	0°		12°	0°		12°	0°		12°

EXPOSED PAD VARIATIONS						
PKG. CODES	D2			E2		
	MIN.	NOM.	MAX.	MIN.	NOM.	MAX.
G3277-2	4.55	4.70	4.85	4.55	4.70	4.85
G4477-1	3.65	3.80	3.95	3.65	3.80	3.95
G4477-2	4.55	4.70	4.85	4.55	4.70	4.85
G4477-3	3.15	3.30	3.45	3.15	3.30	3.45
G4877-1	4.95	5.10	5.25	4.95	5.10	5.25
G4877-2	5.45	5.60	5.75	5.45	5.60	5.75

NOTES:

1. DIE THICKNESS ALLOWABLE IS 0.305mm MAXIMUM (.012 INCHES MAXIMUM).
2. DIMENSIONING & TOLERANCES CONFORM TO ASME Y14.5M. - 1994.
3. N IS THE NUMBER OF TERMINALS.
   Nd IS THE NUMBER OF TERMINALS IN X-DIRECTION & Ne IS THE NUMBER OF TERMINALS IN Y-DIRECTION.
4. DIMENSION b APPLIES TO PLATED TERMINAL AND IS MEASURED BETWEEN 0.20 AND 0.25mm FROM TERMINAL TIP.
5. THE PIN #1 IDENTIFIER MUST EXIST ON THE TOP SURFACE OF THE PACKAGE BY USING INDENTATION MARK OR INK/LASER MARKED. DETAILS OF PIN #1 IDENTIFIER IS OPTIONAL, BUT MUST BE LOCATED WITHIN ZONE INDICATED.
6. EXACT SHAPE AND SIZE OF THIS FEATURE IS OPTIONAL.
7. ALL DIMENSIONS ARE IN MILLIMETERS.
8. PACKAGE WARPAGE MAX 0.08mm.
9. APPLIED FOR EXPOSED PAD AND TERMINALS.
   EXCLUDE EMBEDDED PART OF EXPOSED PAD FROM MEASURING.
10. MEETS JEDEC MO220 EXCEPT DIMENSION "b" MINIMUM.
11. APPLY ONLY FOR TERMINAL.
12. THIS PACKAGE OUTLINE APPLIES TO ANVIL SINGULATION(STEPPED SIDES).

PROPRIETARY INFORMATION
TITLE: PACKAGE OUTLINE,
32,44,48L QFN, 7x7x0.90 MM
APPROVAL
DOCUMENT CONTROL NO.
21-0092
REV.
H
2/2