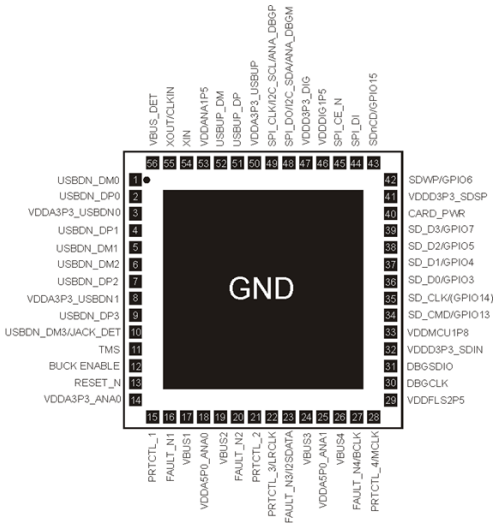
Boston USB Hub with SD Card Port ASIC

Product Specification

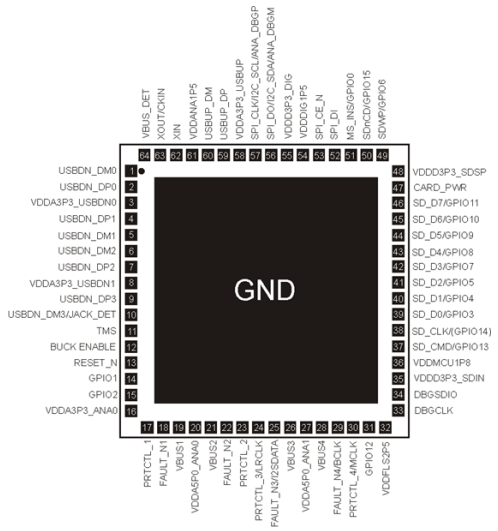
Main Features:

- USB 2.0 – Compliant[1] HUB operation.
- Up to 4 exposed downstream ports available.
- AEC-Q100 Qualified
- Integrated USB 2.0 Device with three interfaces: I2S interface, Mass Storage Class SD Card Read/Write Operation and HID Class microcontroller interface.
- In-package Cortex M0 microcontroller with 8KB RAM and 160KB FLASH memory.
- Integrated USB frequency synthesis (external crystal required).
- Host-to-Host Bridge Controller
- Supports indefinite shorts of USB DP/DM pins to ground and VBUS.
- Integrated power management supports core digital and low-voltage SD Card interface operation.
- Low-power VBUS current-draw detection for wake-up from sleep mode.
- Flexible USB Charging Emulator Termination supports SDP/CDP when USB hub is active and multiple proprietary DCP charging terminations otherwise.
- Current measurement circuitry that reports programmable overcurrent conditions on a hub and port-by-port basis.
- Accumulated charge measurement circuitry that reports total charge drawn from the car battery over a programmable interval.
- Robust reset circuitry intended for automotive environment.
- I2C/SPI Master interface.

1. With the exception of suspend current.



Boston QFN 8mmX8mm 56-Pin Pinout



Boston QFN 9mmX9mm 64-Pin Alternate Pinout

1.0 Table of Contents

BOSTON USB HUB WITH SD CARD PORT ASIC 1

PRODUCT SPECIFICATION 1

1.0 TABLE OF CONTENTS 2

3.0 INTRODUCTION AND GENERAL DESCRIPTION 11

4.0 PIN DESCRIPTION 12

4.1 Pin Types 12

4.2 Pin List 12

4.3 Pin State Upon Power-On Reset 15

5.0 MAXIMUM RATINGS 16

6.0 TYPICAL OPERATING CONDITIONS AND CURRENT DRAW 17

7.0 DETAILED DESCRIPTION 18

7.1 Top Level System Description 18
- 7.1.1 Clocks 22
- 7.1.2 Reset Control 24
  - 7.1.2.1 Simple Specification 24
  - 7.1.2.2 Reset Circuitry 26
  - 7.1.2.3 PMU Sequencer Logic 31
  - 7.1.2.4 PMU Enable (PMU_EN) Holding Circuitry 35
- 7.1.3 Modes of Operation 37
  - 7.1.3.1 USB Active Hub Mode (Mode 1) 37
  - 7.1.3.2 Hub Suspend Mode (Mode 2) 37
  - 7.1.3.3 Charger Only Mode (Mode 3) 37
  - 7.1.3.4 Low Power Current Detect Wake Up (LPCDWU) Sleep Mode (Mode 4) 37
  - 7.1.3.5 Deep Sleep Mode (Mode 5) 38

7.2 Analog Functions 41
- 7.2.1 Clock Sources and Generation 41
  - 7.2.1.1 Pierce Crystal (XTAL) Oscillator 41

7.2.1.1.1 Electrical Characteristics 41
7.2.1.2 Frequency Multiplying PLL and Frequency Dividers....42
7.2.1.2.1 Electrical Characteristics 42
7.2.1.3 IC Oscillator (ICOSC) (4MHz)....43
7.2.1.3.1 Electrical Characteristics 43
7.2.1.4 Ultra-Low-Power Oscillator (LPOSC) (10kHz)....44
7.2.1.4.1 Electrical Characteristics 44
7.2.2 USB Hub Analog PHY 45
7.2.2.1 Top Level 45
7.2.2.2 Terminations Control....46
7.2.2.3 FS/LS Driver 46
7.2.2.4 FS/LS Differential Receiver....46
7.2.2.5 Single-Ended Receivers 46
7.2.2.6 High-Speed Current Driver 46
7.2.2.7 Disconnect Envelope Detector....46
7.2.2.8 Transmit Envelope Detector 47
7.2.2.9 High Speed Differential Receiver....47
7.2.2.10 CDR DLL....47
7.2.2.11 VBUS Comparator 47
7.2.2.12 Electrical Characteristics....47
7.2.3 Charge Measurement 48
7.2.3.1 Description and Interface 48
7.2.3.2 Coarse Requirements / Features....49
7.2.3.3 Architecture....50
7.2.3.3.1 Input Chopping Switches 50
7.2.3.3.2 Resistive Division-By-2/3 50
7.2.3.3.3 Gm-C Integration Cell 50
7.2.3.3.4 Gm-C Calibration 51
7.2.3.3.5 ADC Buffer....51
7.2.3.3.6 Clock Generation 51
7.2.3.3.7 Digital Back End 52
7.2.3.4 Electrical Characteristics....53
7.2.4 Flexible Charging Emulator Termination (FCET)....54
7.2.4.1 Description / Interface 54
7.2.4.2 Programmable Resistive Divider and Pull-Down Resistor 55
7.2.4.2.1 Description / Interface 55
7.2.4.2.2 Coarse Requirements / Features....55
7.2.4.2.3 Architecture....55
7.2.4.2.4 Electrical Characteristics 60
7.2.4.3 Terminal Short 60
7.2.4.3.1 Description / Interface 61
7.2.4.3.2 Coarse Requirements / Features....61
7.2.4.3.3 Architecture....61
7.2.4.3.4 Electrical Characteristics 61
7.2.4.4 Programmable Voltage Source (DAC)....62
7.2.4.4.1 Description / Interface 62
7.2.4.4.2 Coarse Requirements....62
7.2.4.4.3 Architecture....63

7.2.4.4.4 Electrical Characteristics 63
7.2.4.5 Programmable Current Sink 64
7.2.4.5.1 Description/Interface 64
7.2.4.5.2 Coarse Requirements 64
7.2.4.5.3 Architecture 64
7.2.4.5.4 Electrical Characteristics 64
7.2.4.6 ADC Access 65
7.2.4.6.1 Description and Interface 65
7.2.4.6.2 Coarse Requirements 65
7.2.4.6.3 Architecture 65
7.2.4.6.4 Electrical Characteristics 66
7.2.5 Low Power Current Detect Wake-Up (LPCDWU) 67
7.2.5.1 Description and Interface 67
7.2.5.2 Coarse Requirements / Features 68
7.2.5.3 Architecture 68
7.2.5.3.1 Bandgap-less Architecture 68
7.2.5.3.2 5V-Tolerant Comparator 69
7.2.5.3.3 PMOS Resistor Array 69
7.2.5.3.4 Turning Elements of PMOS Resistor Array On and Off 70
7.2.5.3.5 Current-Limiting Feature 70
7.2.5.3.6 Calibration 70
7.2.5.4 Electrical Characteristics 70
7.2.6 ADC (Used for Housekeeping ADC and Charge Measurement ADC) 71
7.2.6.1 Description/Interface 71
7.2.6.2 Coarse Requirements 72
7.2.6.3 Architecture 72
7.2.6.3.1 Sampling Time 72
7.2.6.4 Electrical Characteristics 72

7.3 Digital Functions 73
7.3.1 USB Hub Digital IP Core 73
7.3.2 USB Device Digital IP Core 74
7.3.3 SD Card Host Digital IP Core 75
7.3.4 USB Audio Input (I2S) 75
7.3.4.1 Architecture 76
7.3.4.2 Frequency Generation 77
7.3.4.3 I2S Interface 77
7.3.4.4 USB Packetizing 78
7.3.4.4.1 Payload Data and Data Rate Synchronization 78
7.3.4.4.2 Packet Format 80
7.3.4.4.3 Frame Tracking / Synchronization / USB Protocol Error Handling 81
7.3.5 Host to Host Bridge 82
7.3.5.1 Port Switching 83
7.3.5.2 Bridged Endpoints 87
7.3.5.2.1 Bulk Transfer 87
7.3.5.2.2 Interrupt 89
7.3.5.2.3 Isochronous PCM Audio Stream 89
7.3.5.3 Firmware Interrupts 91

7.3.6 Charge Measurement Digital Back End (DBE) 91
7.3.6.1 Description/Interface 91
7.3.6.1.1 Background 91
7.3.6.1.2 System Partitioning 92
7.3.6.1.3 Interface 93
7.3.6.2 Coarse Requirements / Features 95
7.3.6.3 Architecture 96
7.3.6.3.1 Analog Circuit and ADC Management 96
7.3.6.3.2 ADC Result to Current Estimate 97
7.3.6.3.3 Current Estimate to IRQ (IRQ 10) 102
7.3.7 Housekeeping ADC Digital Back End (DBE) 103
7.3.7.1 Simple Specification 103
7.3.7.2 Merging Customer Requirements with USB BCS Requirements 103
7.3.7.3 HKADC (Housekeeping ADC) DBE Organization 105
7.3.7.3.1 Organization 105
7.3.7.3.2 Operation 106
7.3.7.3.3 One Shots 107
7.3.7.4 System Implications 108
7.3.7.4.1 Power Dissipation 108
7.3.7.4.2 Enable/Disable 108
7.3.7.4.3 Sharing of ADC 108
7.3.7.4.4 Control over ADC 108
7.3.7.4.5 Test 108
7.3.8 Miscellaneous Digital Features 108
7.3.8.1 LPOSC (10kHz) Calibration Counter 108
7.3.8.2 Suspend Clock Handover and Suspend IRQ (IRQ14) 109
7.3.8.3 Port Power Control (PPC) Digital Interface 110
7.3.8.3.1 Interface 111
7.3.8.3.2 Functionality 111
7.3.8.3.3 Overcurrent IRQ (IRQ 11) 112
7.3.8.4 Port State Change IRQ / Port State Registers (IRQ 7) 113
7.3.8.5 LPCDWU IRQ (IRQ 9) 114
7.3.8.6 Upstream VBUS Detect & Sleep Timer IRQ (IRQ 12) 114
7.3.8.7 500Ω Pull Down Control 115
7.3.8.8 Downstream PHY disable function 116

7.4 Power Management (PMU) 116
7.4.1 Bandgap Voltage Reference 116
7.4.1.1 Description/Interface 116
7.4.1.2 Electrical Characteristics 117
7.4.2 Linear Voltage Regulators (LDOs) 118
7.4.2.1 Description/Interface 118
7.4.2.2 3.3V to 1.5V Analog LDO Regulator (Low Current – 15mA) 119
7.4.2.3 3.3V to 1.5V Digital LDO Regulator (High Current – 200mA) 119
7.4.2.4 3.3V to 1.8V MCU LDO Regulator (High Current – 120mA) 119
7.4.2.5 3.3V to 2.5V FLASH LDO Regulator (Low Current – 10mA) 120
7.4.2.6 Electrical Characteristics 120

7.5 Peripheral Interfaces 121
7.5.1 SD Card Interface 121
7.5.1.1 Power and I/O Supply Switching Scheme 122
7.5.1.1.1 Differences between SD Card Specification Example and Boston....122
7.5.1.1.2 Control Features 122
7.5.1.2 Power Switch 123
7.5.1.2.1 Description/Interface 123
7.5.1.2.2 Coarse Requirements 124
7.5.1.2.3 Electrical Characteristics 124
7.5.1.3 I/O Supply Switch (Regulator Selector) 125
7.5.1.4 SD Card I/Os (Signaling Interface) 125
7.5.2 SPI/I2C Interface....126
7.5.3 Test Interfaces 126
7.5.3.1 Digital Test Port 126
7.5.3.2 Analog Test Port 126

7.6 Clough Cortex M0 ARM Microcontroller....127
7.6.1.1 Memory Description and Map 127
7.6.1.2 RAM 129
7.6.1.3 User Programmable Flash Memory 129
7.6.1.4 Flash Memory Registers 130
7.6.1.5 Flash Memory Erase, Write and Protect 134
7.6.1.5.1 Block Erase....134
7.6.1.5.2 Erase Sector 135
7.6.1.5.3 Write Byte 135
7.6.1.5.4 Code Protect....136
7.6.1.6 Interrupt Enabling/Disabling Process....137
7.6.1.7 Peripherals....139
7.6.1.8 Systick Timer 139
7.6.1.9 Timern (0, 1 and 2) 140
7.6.1.9.1 Timern Registers 140
7.6.1.9.2 Timern Operation....141
7.6.1.10 Watchdog....142
7.6.1.10.1 WDT Registers 142

7.7 Recommended Firmware Operation 143

8.0 PACKAGING INFORMATION 146

9.0 APPENDIX 1 – REGISTER DEFINITIONS 147

10.0 APPENDIX 2 – INTERRUPT DEFINITIONS....147

11.0 REFERENCES....149

12.0 REVISION HISTORY 149

2.0 Table of Figures

Figure 7-1 Boston Simplified Top Level Diagram....18
Figure 7-2 Boston Clock Domains....22
Figure 7-3 Relationship of Reset (PMU Sequencer) Circuitry to Rest of Boston....24
Figure 7-4 Boston Analog Brown Out Reset (BOR) Circuitry Simplified Schematic....26
Figure 7-5 Boston dV/dt-Based POR Simplified Schematic....28
Figure 7-6 Boston Level-Based POR Simplified Schematic....29
Figure 7-7 Boston PMU Sequencer Logic State Transition Diagram....31
Figure 7-8 Boston XTAL Oscillator Circuit Simplified Schematic....41
Figure 7-9 Boston Frequency Multiplying PLL and Frequency Dividers Circuit Simplified Schematic...42
Figure 7-10 Boston IC Oscillator Circuit (ICOSC) Simplified Schematic....43
Figure 7-11 Boston Ultra-Low-Power Oscillator (LPOSC) (10kHz) Circuit Simplified Schematic....44
Figure 7-12 Boston Analog PHY Block Diagram....45
Figure 7-13 Boston Charge Measurement Analog Circuit Simplified Schematic....48
Figure 7-14 Boston Flexible Charging Emulator Termination Top Level Block Diagram....54
Figure 7-15 Boston Programmable Resistive Divider Circuit (Pull-Down Portion Shown Only) Simplified Schematic....56
Figure 7-16 Boston Programmable Pull-Down Resistor Value Coverage (Nominal)....57
Figure 7-17 Boston Programmable Resistive Divider Total Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.3,2.0,2.7,3.5V)....58
Figure 7-18 Boston Programmable Resistive Divider Total Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.65V,2.35V,3.1V)....58
Figure 7-19 Boston Programmable Resistive Divider Thevenin Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.3,2.0,2.7,3.5V)....59
Figure 7-20 Boston Programmable Resistive Divider Thevenin Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.65V,2.35V,3.1V)....59
Figure 7-21 Boston FCET Terminal Short Simplified Schematic....60
Figure 7-22 Boston FCET Voltage Source (DAC) Simplified Schematic....62

Figure 7-23 Boston FCET Programmable Current Sink Simplified Schematic 64
Figure 7-24 Boston Low Power Current Detect Wake-Up Circuit Simplified Schematic 67
Figure 7-25 USB Audio Input System Architecture 76
Figure 7-26 I2S Protocol 78
Figure 7-27 Example asynchronous mode packet sequence, slow host 79
Figure 7-28 Example asynchronous mode packet sequence, fast host 79
Figure 7-29 Example asynchronous mode packet sequence, nearly matched host 80
Figure 7-30 Example synchronous mode packet sequence, slow host 80
Figure 7-31 Example synchronous mode packet sequence, fast host 80
Figure 7-32 Example synchronous mode packet sequence, nearly matched host 80
Figure 7-33 Audio frame bit ordering (ignoring NRZI coding and bit stuffing) 81
Figure 7-34 Data packet composition and ordering 81
Figure 7-35(a) Port Switch Sequence 84
Figure 7-36(b) Port Switch Sequence 85
Figure 7-37(c) Port Switch Sequence 86
Figure 7-38 H2H Bridge Dataflow 87
Figure 7-39 Boston Charge Measurement System End-to-End Diagram 96
Figure 7-40 Boston Charge Measurement Digital Back End ALU Control State Diagrams 97
Figure 7-41 Boston Charge Measurement Example Program DSP Flowchart 99
Figure 7-42 Boston Charge Measurement DBE ADC to Current ALU Subsystem 100
Figure 7-43 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 0 101
Figure 7-44 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 1 101
Figure 7-45 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 2 102
Figure 7-46 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 3 102
Figure 7-47 Boston Charge Measurement DBE Current to Interrupt Subsystem 103
Figure 7-48 Boston SD Card Interface Power and I/O Supply Switching 123

Figure 7-49 Proposed Firmware State Diagram (Per Port) for CDP Handshaking During USB Active Mode 144

Figure 7-50 Proposed Firmware State Diagram (Per Port) for Charging Mode 145

3.0 Introduction and General Description

The Boston ASIC is a combination USB Hub with an integrated USB Device supporting three interfaces: I2S audio interface, a Mass Storage Class SD Card Host and a HID Class microcontroller access side-channel. The Boston ASIC can support up to 4 external downstream ports, each of which can withstand an indefinite short of the DP/DM pins to 5V.

The Boston ASIC implements USB Host to Host Bridge component which enable dual role functionality at a downstream USB hub port.

The Boston ASIC contains flexible charging emulator terminations which allow each port to independently masquerade as either a USB Battery Charging Specification – specified termination or one of an infinite number of proprietary dedicated charging port terminations. The Boston ASIC has also been designed to support charging from an automotive vehicle battery during a predefined charging period after deactivation of the vehicle. In order to conserve current, it contains circuitry to wake itself up from sleep mode when current is drawn from a port's VBUS power rail.

The Boston ASIC is assembled with an in-package Cortex M0 microcontroller which, with the appropriate firmware loaded, mediates data transfers to and from the USB Device and the SD Card Host. The Cortex M0 microcontroller, along with the appropriate firmware, can also mediate the handshaking required for successful flexible charging emulator termination implementation.

The Boston ASIC contains charge measurement circuitry which measures the voltage across port power control switches with on-resistances varying from 30mΩ to 70mΩ and obtains a resultant output current measurement. Using a DSP back end, Boston can compensate for temperature-induced resistance changes when programmed with a correct thermal model of the Hub module. The digital back end of this charge measurement circuitry issues a microcontroller interrupt when any one of a number of programmable overcurrent, over-charge, or undercurrent events occur.

The Boston ASIC also integrates linear voltage regulators to support 1.5V core digital device operation and 1.8V high speed SD Card interfacing. Boston also contains an I2C-or-SPI master interface which can be used to communicate with other integrated circuits on a product board.

4.0 Pin Description

4.1 Pin Types

There are several types of pins. The primary packaging variant of Boston will be the 56-pin package.

- **ANAINP=Analog Input**
- **ANAOUT=Analog Output**
- **ANAIO=Analog Input/Output.**
- **VREG= Regulated Output.**
- **DIGGPIO=Digital GPIO Pin with 12mA drive capability.**
- **DIGGPIOPUD=Digital GPIO Pin with 12mA drive capability with programmable 50kΩ nominal pull-up and pull-down resistors.**
- **DIGIN=Digital Input-Only Pin, No pull-up.**
- **DIGINPD1=Digital Input-Only Pin, 20 kΩ pull down always on.**
- **DIGINPD2=Digital Input-Only Pin, 50 kΩ pull down always on.**
- **DIGOUT= Digital Output Pin with 12mA drive capability.**
- **MCUIO=MCU Digital Pin.**
- **SDGPIO=Digital GPIO Pin with Switched Supply (3.3V or 1.8V) Compliant with [4].**
- **SUPPLY=Supply Pin.**

4.2 Pin List

Table 4-1 Boston 56-Pin Main Variant Pin List

#	Name	Type	Direction	Description
1	USBDN_DM0	ANAIO	I/O	USB Downstream Port 0 Minus Terminal
2	USBDN_DP0	ANAIO	I/O	USB Downstream Port 0 Plus Terminal
3	VDDA3P3_USBDN0	SUPPLY	I/O	3.3V Supply for USBDN_0/1
4	USBDN_DP1	ANAIO	I/O	USB Downstream Port 1 Plus Terminal
5	USBDN_DM1	ANAIO	I/O	USB Downstream Port 1 Minus Terminal
6	USBDN_DM2	ANAIO	I/O	USB Downstream Port 2 Minus Terminal

Table 4-1 Boston 56-Pin Main Variant Pin List

#	Name	Type	Direction	Description
7	USBDN_DP2	ANAIO	I/O	USB Downstream Port 2 Plus Terminal
8	VDDA3P3_USBDN1	SUPPLY	I/O	3.3V Supply for USBDN_2/3
9	USBDN_DP3	ANAIO	I/O	USB Downstream Port 3 Plus Terminal
10	USBDN_DM3/JACK_DET	ANAIO	I/O	USB Downstream Port 3 Minus Terminal/Jack Detect
11	TMS	DIGINPD1	IN	Test Mode Select for Scan Chain Digital ATE Test
12	BUCK ENABLE	DIGOUT	OUT	Primary Function: Hold Enable Pin of Buck Converter
13	RESET_N	DIGIN	IN	External Reset Pin
14	VDDA3P3_ANA0	SUPPLY	I/O	3.3V Supply For Charge Measurement Features
15	PRTCTL_1	DIGOUT	OUT	Port Power Switch Enable
16	FAULT_N1	DIGGPIOPUD	I/O	Port Power Switch Fault Detect (Active Low)
17	VBUS1	ANAIO	I/O	Sense Terminal for PPC Switch Drain Voltage
18	VDDA5P0_ANA0	SUPPLY	I/O	5V Supply For 5V Protection / Charge Meas. Features
19	VBUS2	ANAIO	I/O	Sense Terminal for PPC Switch Drain Voltage
20	FAULT_N2	DIGGPIOPUD	I/O	Port Power Switch Fault Detect (Active Low)
21	PRTCTL_2	DIGOUT	OUT	Port Power Switch Enable
22	PRTCTL_3/LRCLK	DIGOUT	OUT	Port Power Switch Enable/I2S LRCLK
23	FAULT_N3/I2SDATA	DIGGPIOPUD	I/O	Port Power Switch Fault Detect (Active Low)/I2S Data
24	VBUS3	ANAIO	I/O	Sense Terminal for PPC Switch Drain Voltage

Table 4-1 Boston 56-Pin Main Variant Pin List

#	Name	Type	Direction	Description
25	VDDA5P0_ANA1	SUPPLY	I/O	5V Supply For 5V Protection / Charge Meas. Features
26	VBUS4	ANAIO	I/O	Sense Terminal for PPC Switch Drain Voltage
27	FAULT_N4/BCLK	DIGGPIOPUD	I/O	Port Power Switch Fault Detect (Active Low)/I2S BCLK
28	PRTCTL_4/MCLK	DIGOUT	OUT	Port Power Switch Enable/I2S MCLK
29	VDDFLS2P5	VREG	I/O	MCU FLASH Memory and I/O Regulator Bypass
30	DBGCLK	MCUIO	IN	MCU Serial Interface Clock
31	DBGSDIO	MCUIO	I/O	MCU Serial Interface Data I/O
32	VDDD3P3_SDIN	SUPPLY	I/O	3.3V Supply for SD Card Interface
33	VDDMCU1P8	VREG	I/O	MCU and SD Card VDDIO Regulator Bypass
34	SD_CMD/GPIO13	SDGPIO	I/O	SD Card Command
35	SD_CLK/(GPIO14)	SDGPIO	OUT	SD Card Clock. OE/PU/PD addressed as GPIO14 but pin is not used as a GPIO.
36	SD_D0/GPIO3	SDGPIO	I/O	SD Card 4/8-Bit Bus Data Bit 0
37	SD_D1/GPIO4	SDGPIO	I/O	SD Card 4/8-Bit Bus Data Bit 1
38	SD_D2/GPIO5	SDGPIO	I/O	SD Card 4/8-Bit Bus Data Bit 2
39	SD_D3/GPIO7	SDGPIO	I/O	SD Card 4/8-Bit Bus Data Bit 3
40	CARD_PWR	ANAIO	I/O	SD Card Power Switch Output
41	VDDD3P3_SDSP	SUPPLY	I/O	3.3V Supply for SD Card Power Supply Switch
42	SD_WP/GPIO6	DIGGPIOPUD	I/O	SD Card Write Protect
43	SD_nCD/GPIO15	DIGGPIOPUD	I/O	SD Card Detect
44	SPI_DI	DIGGPIO	IN	SPI Data In

Table 4-1 Boston 56-Pin Main Variant Pin List

#	Name	Type	Direction	Description
45	SPI_CE_N	DIGGPIO	OUT	SPI Chip Select / Slave Transmit Enable
46	VDDDIG1P5	VREG	I/O	Core Digital Regulator Bypass Capacitor Connection
47	VDDD3P3_DIG	SUPPLY	I/O	3.3V Supply for Core Digital Regulator
48	SPI_DO/I2C_SDA/ANA_DBGM	DIGGPIO	I/O	SPI Data Out / I2C Data / Analog Debug
49	SPI_CLK/I2C_SCL/ANA_DBGP	DIGGPIO	I/O	SPI Clock / I2C Clock / Analog Debug
50	VDDA3P3_USBUP	SUPPLY	I/O	3.3V Supply for USBUS / 1.5V Analog Regulator
51	USBUP_DP	ANAIO	I/O	USB Upstream Port Plus Terminal
52	USBUP_DM	ANAIO	I/O	USB Upstream Port Minus Terminal
53	VDDANA1P5	VREG	I/O	USB XTAL/PLL Regulator Bypass
54	XIN	ANAINP	IN	USB XTAL Crystal Pin Input
55	XOUT/CKIN	ANAOUT	OUT	USB XTAL Crystal Pin Output or External Clock Input
56	VBUS_DET	DIGIN	IN	Detects the level of VBUS_US
DB	GND	SUPPLY	I/O	Ground down bond.

4.3 Pin State Upon Power-On Reset

- Unless otherwise noted, all pins default to tri-state (high impedance) upon power-on reset.
- Pins otherwise noted include:
  - SUPPLY pins.
  - VREG pins: output a regulator voltage upon power-on reset.
  - DIGPDIN1/DIGPDIN2 pins: are pulled to ground with weak pull-down resistors.
  - XOUT/CKIN pin: is pulled down to ground.
  - SPI_CLK and SPI_DO pins: are pulled down to ground.

        - This pull-down is released via analog trim register setting.
    - PRTCTL_1/2/3/4 pins: are pulled down to ground.
        - PRTCTL_1/2/3/4 assert value during digital core off state is analog trim-register programmable but defaults to logic low.
- Several pins do not have current-gating READ ENABLE control due to their use as scan test vector or reset inputs. In order to prevent excess leakage through the input buffer, these pins must be tied to a well-defined logic level either externally or through enabling the pin output driver. Pins included in this category include:
    - SPI_DI
    - SPI_CE_N
    - BUCK_ENABLE
    - VBUS_DET
    - RESET_N

5.0 Maximum Ratings

Table 5-1 Absolute Maximum Ratings, Voltages Referenced to Ground				
Name	**Conditions**	**Min**	**Max**	**Unit**
VDDA5P0 pins	no damage	-0.3	5.5	V
VDDA3P3, VDDD3P3 pins	no damage	-0.3	3.6	V
Analog Input and Output: CARD_POWER	no damage	-0.3	3.6	V
Analog Input and Output: XIN, XOUT/CKIN	no damage	-0.1	1.65	V
Analog Input and Output (not all supplies present): USBDN_DP0, USBDN_DM0, USBDN_DP1, USBDN_DM1, USBDN_DP2, USBDN_DM2, USBDN_DP3, USBDN_DM3, USBUP_DP, USBUP_DM, VBUS1, VBUS2, VBUS3, VBUS4	no damage	-0.3	3.6	V
Analog Input and Output (all supplies present): USBDN_DP0, USBDN_DM0, USBDN_DP1, USBDN_DM1, USBDN_DP2, USBDN_DM2, USBDN_DP3, USBDN_DM3, USBUP_DP, USBUP_DM, VBUS1, VBUS2, VBUS3, VBUS4	no damage	-0.3	5.5	V
DIGGPIO, DIGGPIOPUD, SDGPIO, DIGIN, DIGINPD1, DIGINPD2, DIGOUT, VREG pins	configured as input, no damage	-0.3	3.6	V
Storage Temperature	de-rated performance , full functionality	-55	+150	°C
HBM (all pins)	any pin to any other pin or ground, loose part	-2	2	kV

Table 5-1 Absolute Maximum Ratings, Voltages Referenced to Ground				
Name	**Conditions**	**Min**	**Max**	**Unit**
CDM (corner pins – AEC-Q100 Requirement)	no damage	-750	750	V
CDM (non-corner pins – AEC-Q100 Requirement)	no damage	-500	500	V
MM (all pins – AEC-Q100 Requirement)	no damage	-200	200	V

6.0 Typical Operating Conditions and Current Draw

Table 6-1 - Recommended Operating Conditions, Voltage Referenced to Ground					
Name	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Temperature	Operating Junction Temperature. De-rated performance, full functionality.	-40	25	125	°C
VDD5P0 voltage		4.5	5.0	5.5	V
VDD3P3 voltage	Do not raise high before VDD5P0 voltage is high.	3.0	3.3	3.6	V
Voltage range: all pins except XIN and XOUT/CKIN		0		VDD3P3	V
Voltage range: XIN and XOUT/CKIN		0		VDDANA1P5	V

Table 6-2 – Specified Current Draw Over Recommended Operating Conditions					
Name	**Conditions**	**Min**	**Typ**	**Max**	**Unit**

Table 6-2 – Specified Current Draw Over Recommended Operating Conditions					
Name	**Conditions**	**Min**	**Typ**	**Max**	**Unit**
Maximum Current Draw, Full Functionality	Full USB 2.0 Operation with SD Card Access. 400mA SD Card Power Not Included. All Conditions.			280	mA
Sleep Current	4 LPCDWU Enabled, All Conditions.			50	µA
Sleep Current	4 LPCDWU Enabled with Half Current, All Conditions.			35	µA
Sleep Current	0 LPCDWU Enabled, All Conditions.			20	µA

7.0 Detailed Description

7.1 Top Level System Description

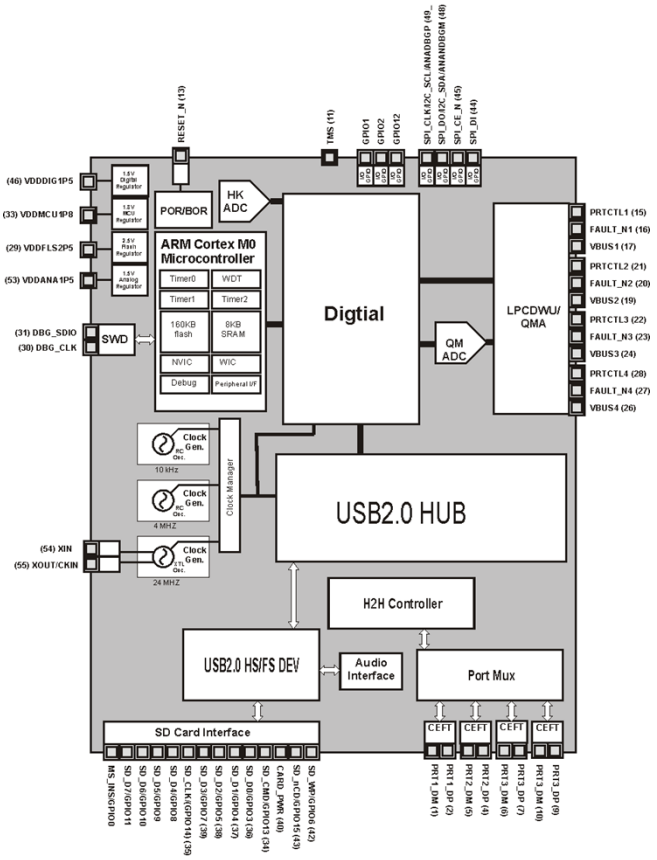


Figure 7-1 Boston Simplified Top Level Diagram

The simplified top-level depiction of the Boston ASIC situated is shown in Figure 7-1. The Boston ASIC can roughly be divided into four sections, consisting of the following blocks:

Analog:

- USB Analog PHY (Upstream (US) and Downstream (DS))
  - Drivers and Receivers for USB line communications.
- XTAL Oscillator, Frequency Multiplication (PLL), Dividers
  - Perform accurate frequency generation for USB operations.

- 4MHz Oscillator (ICOSC)
  - A low power oscillator intended for use when the USB frequency generation is not active.
- 10kHz Oscillator (LPOSC)
  - An ultra-low power oscillator intended for use during sleep and wakeup. This circuit times the power-up sequence and can be used to operate the on-board sleep timer.
- Flexible Charging Emulator Termination (FCET)
  - A collection of voltage sources, current sinks, resistors, and a shorting switch connected to the USB DP and DM lines that can mimic the linear component network of a wide variety of charging ports utilizing USB connectors.
- Low Power Current Detect Wake Up (LPCDWU)
  - A circuit that implements a connection between the 5V supply and VBUS during sleep and charging modes to detect the connection of a device to be charged by its current draw.
- Charge Measurement Analog
  - A circuit which amplifies and low-pass filters the voltage across the port power control switch and presents the buffered result to the Charge Measurement ADC.
- Charge Measurement ADC
  - A 12-bit resolution, 10-bit accurate ADC which rotates among the four Charge Measurement Analog circuits and presents results to the Charge Measurement Digital Back End (DBE)
- House Keeping ADC
  - A 12-bit resolution, 10-bit accurate ADC which rotates among up to 32 internal connections on a programmable basis, including the ADC input connections in the FCET.

Digital:

- USB HUB and Digital PHY IP Core
  - Routes data from the USB upstream port to downstream port, configures downstream port.
- USB Device IP Core
  - Implements a USB Device with two interfaces: Mass Storage Class for SD Card Host and HID Class for a microcontroller bridge. Acts as a USB protocol decoder and data buffer.
- Host to Host Bridge Core

  - Provide two additional USB device controllers and data bridging functions to allow an external USB device attached to one of the “downstream” ports to transfer data to/from the upstream USB host while the external device is itself acting as a USB host
- USB Audio Interface
  - Support for USB audio-in streaming using an external I2S-compliant ADC input for the audio source
- SD Card Host IP Core
  - Implements a bridge between the USB device and the SD Card.
- Clough MCU Interface
  - Interfaces the core digital circuitry with the microcontroller.
- Dig PHY (HS,FS,LS)
  - Implements a bridge between the USB Hub IP and the USB Device IP
- Charge Measurement Digital Back End (DBE)
  - Implements a programmable ALU to perform DSP operations on the Charge Measurement ADC outputs which correspond to port power control (PPC) switch voltage. These DSP operations account for changes in the resistance of the PPC switch and provide a current output. This current output is then used to generate a microcontroller interrupt under a number of interesting conditions.
- House Keeping ADC Digital Back End (DBE)
  - Implements a housekeeping schedule in order to allow the ADC to make periodic voltage measurements at predefined locations around the ASIC without requiring the microcontroller to continually poll the ADC.
- Port Power Control Interface
  - Provides an interface between the firmware and digital to the external port power control switch.

Microcontroller (Clough ASIC):

- A Cortex M0 microcontroller with 160KB FLASH memory and 8KB SRAM.

PMU:

- 3.3V→1.5V 200mA Regulator
  - Supplies the digital core circuitry.
- 3.3V→1.8V 120mA Regulator
  - Supplies the microcontroller core digital circuitry and the SD Card interface.

- 3.3V→1.8V 10mA Regulator
  - Supplies the microcontroller FLASH and I/O circuitry.
- 3.3V→1.5V 15mA Regulator
  - Supplies the XTAL Oscillator and Frequency Multiplying PLL.
- Bandgap (and Buffer)
  - Provides a precision voltage reference at 1.213V for the ASIC
- V2I
  - Provides a bandgap-voltage derived current reference for the ASIC
- Trim V2I
  - Provides a calibrated bandgap-voltage derived current reference for the ASIC that is required for USB High-Speed operation.
- Resets (External, Power On, Brownout)
  - These blocks work together to ensure that the ASIC is reset whenever the state of its digital registers may have become corrupted.

7.1.1 Clocks

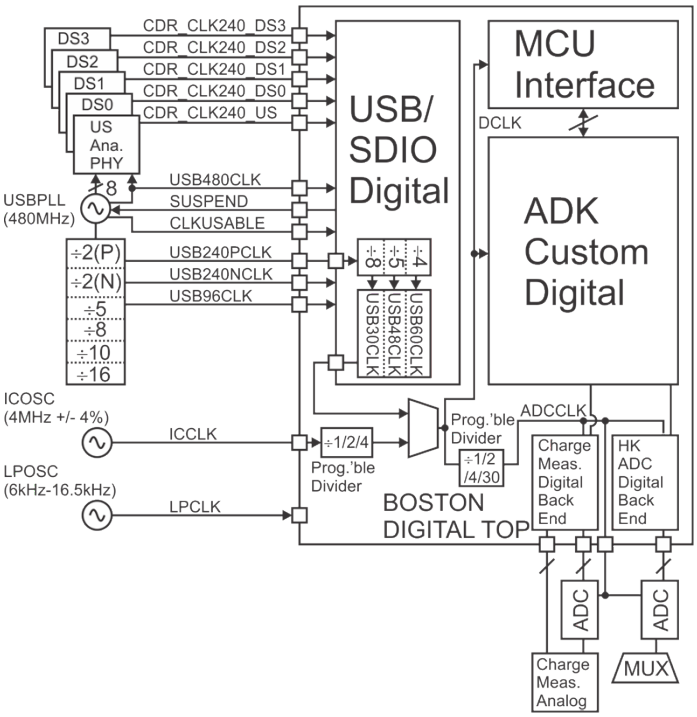


Figure 7-2 Boston Clock Domains

Boston has three ultimate clock sources:

1. The USBPLL (the same as the XTAL Oscillator and the Frequency Multiplier).
2. The ICOSC, an accurate, low power current-capacitor (hence IC) oscillator.
3. The LPOSC, an ultra low power oscillator intended for use in sleep modes to drive the Reset Circuitry/PMU Sequencer.

From these clock sources, a number of other clocks are derived:

Table 7-1 List of Boston Clocks

Clock Name	Source	Serves	Notes
LPCLK	LPOSC (10kHz)	Reset Circuitry	For operation of 3.3V PMU sequencer. It is always running
ICCLK	ICOSC (4MHz)	DCLK Mux	For low-power (USB

Table 7-1 List of Boston Clocks			
Clock Name	**Source**	**Serves**	**Notes**
			SUSPEND Mode) operation. It is powered on upon Boston reset. It is powered off in sleep modes (4 & 5).
DCLK	DCLK MUX	ADK Main Custom Digital	The main digital and microcontroller clock.
USB30CLK	Internal Dividers	DCLK Mux / USB Digital	Is turned off during USB SUSPEND mode or when USB operation is disabled. CLKUSABLE indicates if this clock is active or not.
USB48CLK	Internal Dividers	USB Digital	
USB60CLK	Internal Dividers	USB Digital	
USB96CLK	USBPLL Dividers	SD Card Digital	
USB240PCLK	USBPLL Dividers	USB Digital	
USB240NCLK	USBPLL Dividers	USB Digital	
CDR_CLK240_US	USB US CDR	USB Digital	Is synchronized with USB HS RX data. Each CDR output is associated with a CDR VALID signal which indicates the validity of the CDR clock.
CDR_CLK240_DS0	USB DS0 CDR	USB Digital	
CDR_CLK240_DS1	USB DS1 CDR	USB Digital	
CDR_CLK240_DS2	USB DS2 CDR	USB Digital	
CDR_CLK240_DS3	USB DS3 CDR	USB Digital	
USB480CLK	USB PLL	USB Hub-Device Interface	Is turned off during USB SUSPEND mode or when USB operation is disabled.
ADCCLK	Programmable Divider	ADCs and ADC Digital Back Ends	ADCCLK is 50% duty cycle.

NOTE: Due to digital designer mistake, the ÷1/2/4 programmable divider was transposed with the USB30CLK/ICCLK multiplexer in Figure 7-2 (i.e. the actual implementation is different from this figure). For all practical purposes, this mistake precludes setting the ÷1/2/4 programmable divider to any setting besides ÷1. Since the reduced power consumption during suspend mode which would be achieved with

the ÷4 setting was not seen as important during customer product development, this architectural result was retained in order to minimize risk while bringing the chip into production.

7.1.2 Reset Control

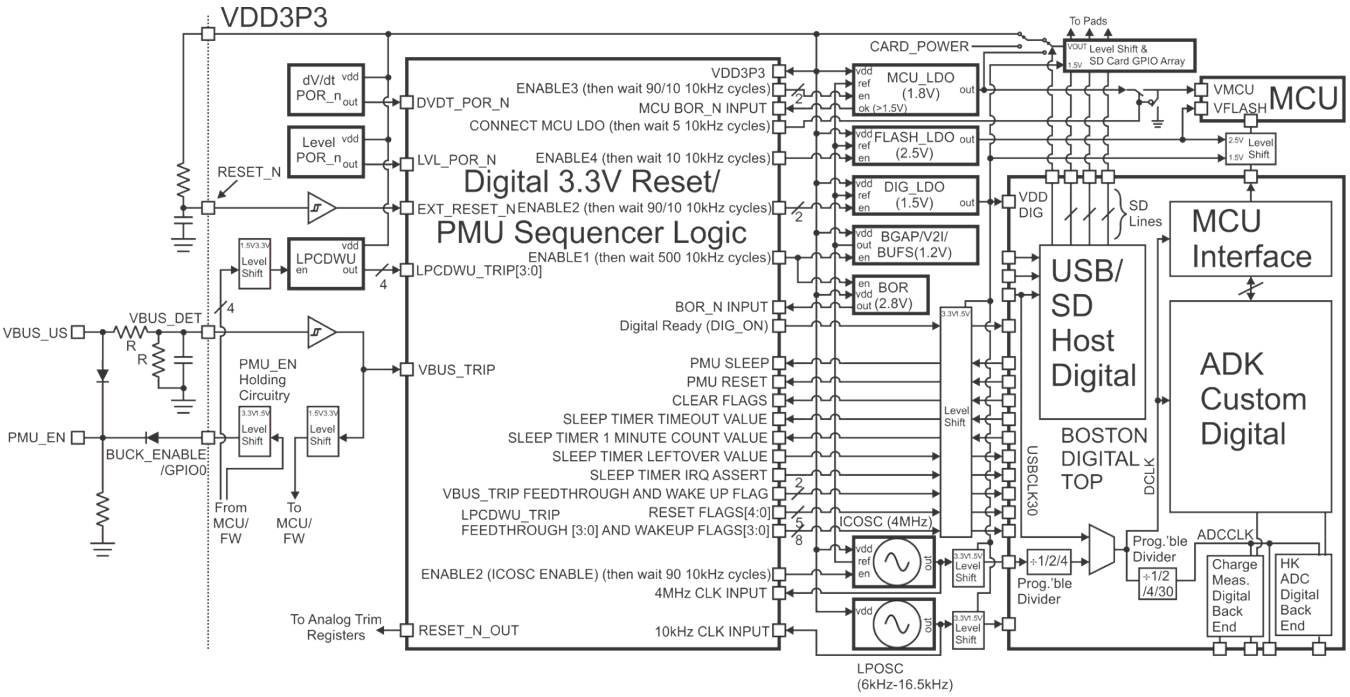


Figure 7-3 Relationship of Reset (PMU Sequencer) Circuitry to Rest of Boston

7.1.2.1 Simple Specification

The Reset Control circuitry comprises the:

- Analog reset circuitry (POR and BOR-labeled blocks) and digital reset circuitry (to be contained in the 3.3V logic)
- PMU Sequencer 3.3V logic
- PMU EN Holding circuitry

The Reset Control circuitry must reset Boston into a default state:

i. During a ramp-up of the VDD3P3 supply voltage.

ii. When any of the power supply waveforms of the car battery described in the documents below are experienced:

(if the resultant Boston supply voltages are such that Boston is outside of the recommended operating supply range or such that Boston can enter into a lockup or other unintended state from which it cannot recover without said reset circuitry.)

- GMW3097 Jul 2006.pdf
- GMW3172_AUG2008.pdf
- ISO 16750-2_2012

Note that to ascertain exactly what the Boston supply voltages are doing, knowledge of the external PMU circuitry is required. The external PMU circuitry consists of a 12V->5V buck converter and a 5V->3.3V LDO regulator. The configuration of these PMU blocks in the existing product, along with their associated external components, can be found in the following documents provided by the customer:

- 002766_AUXPODVAR#3_PSUP_REV_B_SCHEM.pdf
- 002767_AUXPODVAR#3_HUB_REV_A_SCHEM.pdf

The existing PMU component data sheets include:

- TPS5431 - Buck Controller.pdf
- NCP1117-D - 3.3V LDO.pdf

The proposed PMU component data sheets can be found at:

- MAX16955 - Buck Converter With Pulse Skipping.pdf
- TLV1117LV33 - Low Power 3.3V LDO.pdf

The Boston ASIC will support at least both sets of components.

The allowable time interval between VDD rising and the firmware gaining control of Boston can be inferred from the following document obtainable from the USB-IF:

- USB 2.0 Connect Timing ECN.pdf

In this document, on page 4, it states (by reference to the main USB specification document) that the HUB must signal device attach 1 second after it has received VBUS_US > 4.0V. Given

that the external PMU circuitry takes about 10ms to enable after VBUS_US > 4.0V, it is acceptable to specify a total Boston power-up time of no greater than **400ms** from the time that the operational range of the 3.3V supply is entered or from the time that VBUS_US > 4.0V if the 3.3V supply has already been established.

iii. When the external RESET_N pin drops to logic low.

iv. During a brownout (VDD3P3 < 2.8(+/-0.1)V) event when the ASIC is active.

v. During a register brownout (VDD3P3 < ~1.6V) even when the ASIC is in a sleep state.

7.1.2.2 Reset Circuitry

The strategy used in Boston will be to design reset circuitry that guarantees a chip reset immediately prior to and potentially thereafter:

(Non-sleep Condition): The supply voltage rises from below the brownout voltage of 2.8V to above the brownout voltage of 2.8V.

(Any Condition): The supply voltage rises from below the level-based reset voltage of > 1.5V to above the level-based reset voltage of > 1.5V.

Note that the minimum level-based reset voltage over corner should be above the holding voltage of a 3.3V-device-based register (1 Vth or ~0.8V, worst-case). The reason is that the Boston ASIC should be reset during the sleep condition if there exists a possibility that any register state has been changed by VDD3P3 dropping to a very low voltage.

A. Analog Brown Out Reset (BOR) Circuitry

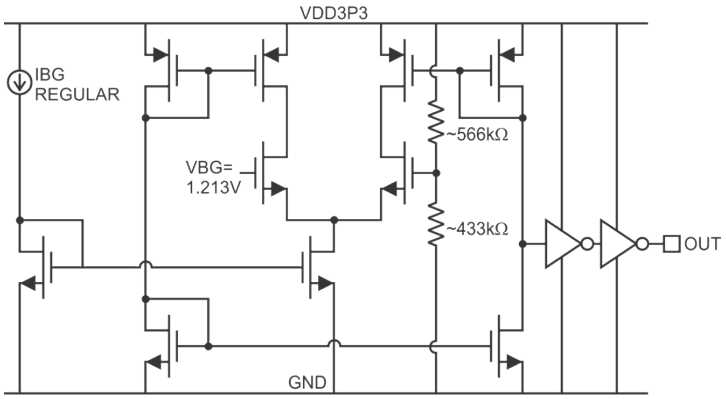


Figure 7-4 Boston Analog Brown Out Reset (BOR) Circuitry Simplified Schematic

- The primary protective reset mechanism of Boston will be the BOR. The BOR will trigger at a fixed nominal (but metal-mask programmable to values of 2.6V and 2.7V) brown-out voltage of 2.8V.
- The BOR will be enabled as part of the PMU power-up sequence and be armed (i.e. the reset will become active) at well-defined states of the PMU sequencer.
- The BOR will be disabled when Boston is put into a sleep mode and the BOR reset will be disarmed before the BOR comparator itself is disabled.
- The BOR threshold voltage of 2.8V will be achieved with an error of no greater than +/- 0.1V (3.57%).
- This error tolerance takes into account any upstream inaccuracies (e.g. the bandgap) that are incurred prior to any circuit calibration trim settings are applied.
- The BOR comparator will consume a current of less than 10uA when active.
- The BOR comparator will trigger in less than 1us.
- The BOR reset output (BOR_N) will be active low.
- The BOR will trigger a complete PMU power-cycling and reset of Boston when triggered.
- The BOR, along with the Bandgap, will provide a correct logic output well-past (~200mV) the supply voltage at which the level-based POR trips for a given corner.
- The BOR shall fulfill the above requirement for ramping of the power supply at any rate.
- The BOR shall have a minimum output pulse width of 10ns to guarantee a global register reset.
- A BOR event shall not be detected as a result of power supply disturbances up to 100mVpp, principally from 100kHz to 1MHz.
- Accuracy: assert BOR_N when VDD3P3 < 2.8V +/- 100mV including pre-trim bandgap and bandgap buffer inaccuracy. Please beat this specification with margin if possible.
- Valid logical output is generated for VDD3P3 < Level-Based POR trip voltage - 200mV.

B. Analog Power-On Reset (POR) Circuitry

- The analog POR circuitry will consist of both a dV/dt-based POR and a level-based POR.
- The reason for including a level-based POR is to reset the part for VDD3P3 excursions outside of the valid operating range that happen with time constants on the order of minutes.
- The BOR circuit performs an accurate level-based POR but is not available when the bandgap and V2I are disabled (e.g. during a sleep mode).

- The BOR circuit also becomes inaccurate (over corner) below some VDD3P3 voltage, above which the level-based POR must be guaranteed to trip over PVT variation.
- Therefore, a separate level-based POR circuit is required.
- The reason for continuing to include a dV/dt-based POR is to ensure that the part will reset for VDD3P3 ramps that are too fast for the level-based reset to accurately respond.

i. dV/dt-Based POR

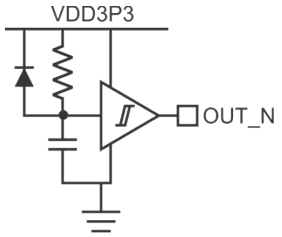


Figure 7-5 Boston dV/dt-Based POR Simplified Schematic

- The dV/dt-based POR must induce a reset for supply voltage ramps (0V to a valid operating voltage - 3.0V to 3.6V) ranging from 0s to 20ms.
- The dV/dt-based POR must not falsely trigger for VDD3P3 jumps within the range of 2.5V to 3.6V.
- Note that said voltage jumps can in principle occur over a very short time interval (e.g. 1ns).
- The dV/dt-based POR shall have a minimum output pulse width of 10ns to guarantee a global register reset.

ii. Level-based POR

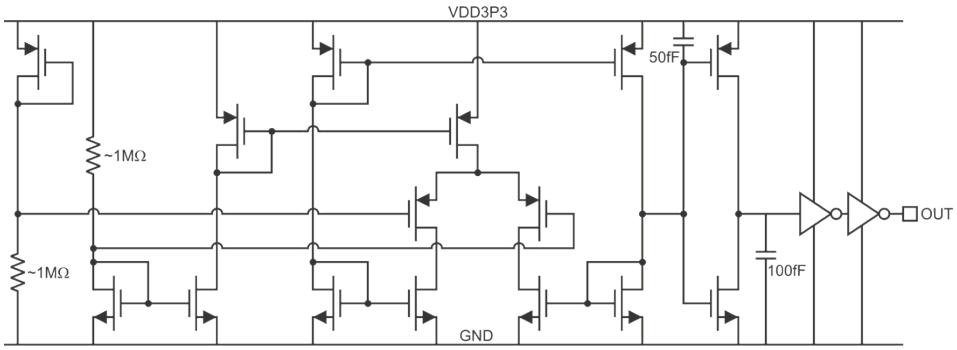


Figure 7-6 Boston Level-Based POR Simplified Schematic

- The level-based POR must consume less than 2uA of standby current.
- The level-based POR must behave within specified limits (those listed below) when the supply voltage ramp time is greater than 100us (i.e. it takes longer than 100us for the supply voltage to reach its final value).
- The level-based POR must trigger at a voltage no less than 1.6V and no greater than 2.6V.
- The level-based POR must not exhibit any runt pulses or high intervals prior to the intended trip point of the circuit.
- The level-based POR shall have a minimum output pulse width of 10ns to guarantee a global register reset.

C. External Reset

- There will exist an external reset pin that is connected to the Reset Control Circuitry via a Schmitt buffer.
- The Schmitt buffer shall have a minimum output pulse width of 10ns to guarantee a global register reset.
- This will be an active low reset pin that under typical usage is tied to VDD3P3 via an external pull-up resistor and to ground via an external capacitor.
- One purpose of this pin is to provide an external pin from which the chip can be awoken during one of its sleep modes.
- Another purpose of this pin is to provide a back-up power-on reset terminal.
- The assertion of the output of this circuit constitutes a reset event that always puts the PMU Sequencer back to STATE0: RESET.

D. Digital Reset Circuitry

- The digital reset circuitry ties together the outputs of the aforementioned resets and also implements a reset timeout using the LPOSC.
- The outputs of the dV/dt POR_N and level-based POR_N and EXT_RESET_N should be combined via an AND operation to create another active low reset signal.
- This ANDing must be structurally coded and each of the reset inputs must be pulse-extended.
- This combined signal will be termed Merged POR_N for the purposes of this document.

- When Merged POR_N goes low, the PMU sequencer should:
    a. Power down immediately whatever PMU sequencer - controlled blocks are currently active and re-initiate the PMU start-up sequence.
    b. In other words, the reset state (and there should be only one reset state) should be entered in this case.
    c. Assert and hold an active-low reset signal on the RESET_N_OUT pin, which resets all 3.3V trim registers.
        i. Also during a reset event, ensure that core digital logic registers are reset to a known state.
        ii. Also during a reset event, ensure that core digital logic registers existing as part of the USB Analog PHY IP are reset.
    d. Wait for the Merged POR_N signal to go high.

- After the Merged POR_N signal goes high, the digital reset circuitry will enter its reset timeout phase.
- In this phase of operation, the PMU sequencer will wait for 300 LPCLK clock cycles before proceeding to the next PMU sequencer state.
- A reset synchronizer must be used.
- All state machines, trim settings, and resets provided to external blocks must be reset by a signal with controlled timing.
- The only circuitry which is allowed to receive signals with timing from the reset is the reset synchronizer itself.
- The reset synchronizer must be structurally coded.

7.1.2.3 PMU Sequencer Logic

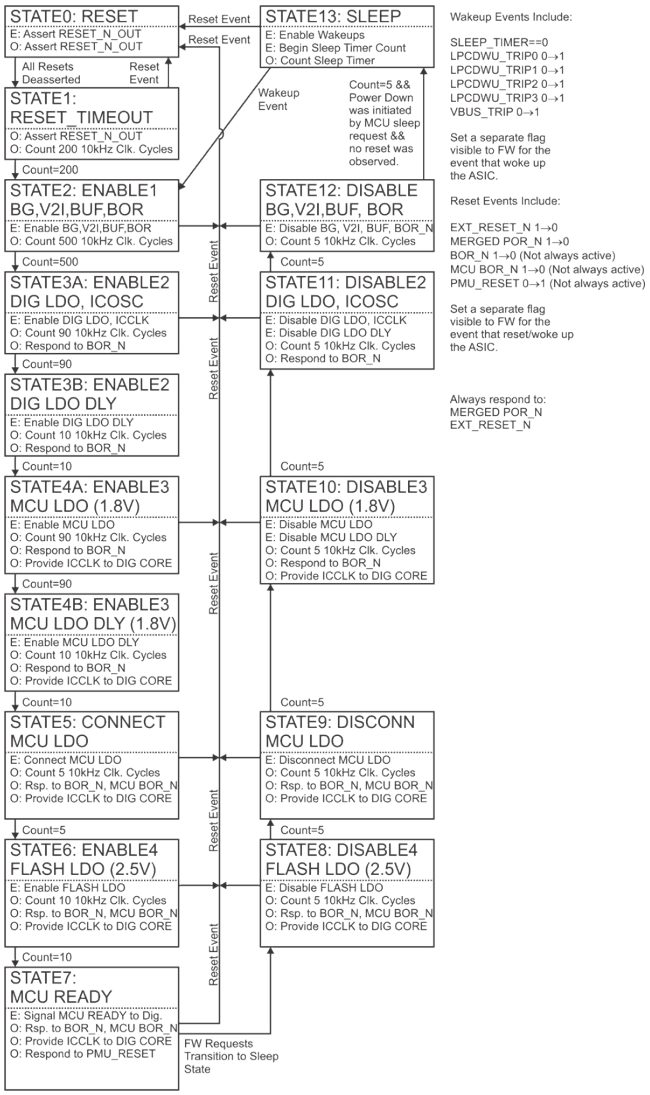


Figure 7-7 Boston PMU Sequencer Logic State Transition Diagram

A. Reset Phase

The reset phase of the PMU sequencer consists of the progression of states:

STATE0: RESET

STATE1: RESET TIMEOUT

In all of these states, assert RESET_N_OUT.

B. BOR Enable Phase

This second phase of the PMU sequencer progresses the chip circuitry to the point where the BOR circuit is enabled. After STATE2, the PMU sequencer / reset circuitry should respond to BOR_N.

STATE2: ENABLE1 - Enable Bandgap / Bandgap Buffers / V2I / BOR

Wait 500 LPCLK clock cycles

C. PMU Enable Phase

This third phase of the PMU sequencer progresses the chip circuitry to the point where firmware control of the system is achieved.

STATE3: ENABLE2 - Enable DIG_LDO (1.5V High Current) and ICOSC (4MHz oscillator)

Wait 90 LPCLK clock cycles for part A and 10 LPCLK clock cycles for part B.

Gate ICCLK signal from the core digital circuitry until subsequent states.

STATE4: ENABLE3 - Enable MCU_LDO (1.8V High Current)

Wait 90 LPCLK clock cycles for part A and 10 LPCLK clock cycles for part B.

Since the MCU_LDO supplies GPIO which drive pads exposed to the Hub module exterior, the MCU must have a BOR circuit associated with it.

The system must respond to this MCU BOR after this state.

STATE5: Connect MCU_LDO (Force MCU POR)

Wait 5 LPCLK clock cycles

STATE6: ENABLE4 - Enable FLASH_LDO (2.5V Low Current)

Wait 10 LPCLK clock cycles

STATE7: MCU READY

At this time, the digital designer may, at his discretion, signal to the 1.5V digital core circuitry that it is OK to begin operation because all supplies are present.

D. Post-power up features

After firmware gains control over the system, the following features must be available in Boston:

i. Sleep (and Wakeup)

- The PMU sequencer should support a sleep input from firmware/digital that will cause the PMU state progression to follow that shown in the Reset State Machine Diagram
- After the bandgap is disabled, the PMU sequencer will be in STATE13: SLEEP.
- In this state,
  - RESET_N_OUT will not be asserted.
  - The analog (trim) register contents will not be modified from their last setting by firmware.
- In this state, the ASIC will be awoken if:
  - A high value is observed on one of the four LPCDWU_TRIP signals.
    - Asynchronous detection of the LPCDWU_TRIP signals is required in order to detect very short current pulses.
  - The SLEEP_TIMER is armed and hits its limiting count value of zero.
  - A reset event (Merged POR_N or External RESET_N) occurs.
  - A logic high value is observed on the VBUS_DET signal.
    - Synchronous detection of the VBUS_DET signal is employed since the VBUS_DET signal is well-filtered at the product level external to the ASIC.
- The PMU sequencer should report via a flag which event woke up the ASIC.
  - For the LPCDWU, there are 4 separate events that should be reported separately, one for each instantiation of the LPCDWU circuit.
  - All such flags will exist in one clear-upon-read register called “PMU_WAKEUP_FLAG”.
    - If any bit in this register equals logic high when the sleep state is entered, the ASIC will wake up immediately.
    - Therefore, this register must be read before entering the sleep state.
- Entering sleep mode can be averted by holding the TMS pin at logic high. This is useful in avoiding a chip-bricking scenario if a firmware engineer writes code which always forces the ASIC to sleep soon after the firmware main() routine is entered (and hence entering sleep cannot in practice be averted via use of a debugger tool).

ii. Sleep Timer

- The sleep timer will be a count-down counter that wakes up the chip via a normal PMU sequence progression from the SLEEP state once it counts down to zero.
- Also when the timer reaches zero:
  - The 3.3V logic should assert an IRQ (only once) to the MCU so that the firmware knows that it is time to enter the "knockout" mode (to be defined below).
    - Note that this IRQ is only received by the MCU if the ASIC has first been awoken by some other event prior to the sleep timer counter reaching zero.
  - The sleep timer stops counting down and stays at a count value of zero.
- The sleep timer count limit (number of minutes before waking up) will be settable by the firmware prior to entering the SLEEP state in the SLP_TIMER_TIMEOUT register.
  - In order for the sleep timer to work properly, the minute prescalar (number of LPCLK cycles in 1 minute) must first be computed and register AUX_CLK_1MIN_CNT must be set by the firmware.
  - The typical value to which the AUX_CLK_1MIN_CNT register should be set is 0x000927C0.
  - In order for the firmware to determine the number of LPCLK cycles in 1 minute, it must set the AUX_CLK_CALIB_CTL register and read the AUX_CLK_CALIB_CNT register. Assuming that the ICCLK is exactly 4MHz, the AUX_CLK_CALIB_CNT register will read the number of 250ns periods in 10 LPCLK periods.
- The sleep timer will be armed and set running as soon as one of the following events occurs:
  - The sleep timer count limit register is set.
  - The minute prescalar value register is set.
- The sleep timer will be able to count up to 3 hours and have a resolution in minute-long intervals and will be able to be programmed in said minute-long intervals.
- If Boston is awoken by the LPCDWU_TRIP signal, the residual count left on the sleep timer should be reported to the firmware.
  - The reason for this is that there is only a limited time interval after the car engine is turned off during which charging is allowed.
  - Some of this time interval will be spent in the SLEEP state, which must be recorded without the assistance of the MCU. Once awoken, the MCU must know how much of the allotted charging interval has elapsed in order to take any required actions as a result of a particular amount of time elapsing.

- However, the sleep timer in the 3.3V logic should continue counting after waking up to avoid requiring the use of additional counters in the MCU to determine when the prescribed charging interval has elapsed.

- The default behavior of the sleep timer is to always count down. If the sleep timer count-down-to-zero event is to be delayed, the firmware must write a new sleep count value to the 3.3V logic.
- By default, the sleep timer counter is set to zero. The implication of this is that by default, the chip will wake up immediately after being put to sleep. A sleep count value must be programmed into the sleep counter in order for the ASIC to stay asleep for any meaningful amount of time. Another implication of this feature of the design is that the ASIC cannot be put to sleep indefinitely.

iii. FW (Firmware)-Defined Reset

- The PMU sequencer will also accept an input from 1.5V core digital circuitry and firmware called PMU RESET.
- The assertion of this signal constitutes a reset event as shown in Figure 7-7.

7.1.2.4 PMU Enable (PMU_EN) Holding Circuitry

In the current customer USB HUB design, the VBUS_DET signal is derived from the upstream USB VBUS signal (VBUS_US) via a resistive voltage divider. The enable signal PMU_EN for the buck controller is also currently derived from the upstream USB VBUS signal via a resistive voltage divider. The customer is asking Boston to implement a feature to accomplish the following:

- When VBUS_US falls, Boston must detect this event but still hold the PMU_EN signal above the buck converter logic high voltage level.
- The reason for holding the PMU_EN signal high is because if Boston does not do this, then its power supplies will collapse.
- When VBUS_US falls, Boston must enter the LPCDWU sleep mode.
- When either the programmable charging time interval or programmable charge limit is set, Boston firmware will assert BUCK_ENABLE low, letting the PMU_EN voltage fall to ground.
- At this point, the ASIC is in what is termed the "Knockout" state and the buck converter will turn off and both the 5V and 3.3V supplies will collapse.
- If at any point during sleep the VBUS_US goes high again, this event must wake up Boston and it must resume USB operation.

- The way that PMU_EN will be held high is with two diodes, one from VBUS_US and one from BUCK_ENABLE.
- The way that PMU_EN will go low is when neither VBUS_US nor BUCK_ENABLE are high, a pull-down resistor will pull PMU_EN low.
- The state of the BUCK_ENABLE output driver circuit should be tri-state upon power up.
- The intended cycle of usage of this circuit will be as follows:
  - i. Car engine is turned on. USB Host becomes active and raises VBUS_US.
  - ii. VBUS_US rising causes PMU_EN to rise to ~4V. This enables the buck converter.
  - iii. The external 5V and 3.3V supplies rise and Boston becomes active after all reset signals are de-asserted and all PMU blocks are enabled.
  - iv. BUCK_ENABLE is disabled and its output tri-stated upon its trim registers receiving RESET_N_OUT.
  - v. VBUS_DET digital input pad circuit (acting as a comparator) checks VBUS_DET and finds that it is at a voltage corresponding to a high logic level.
  - vi. The output of the VBUS_DET input pad is made available to firmware as bit 4 of the LPCDWU_STATUS register.
  - vii. Boston firmware reads this bit and sets the output of BUCK_ENABLE high to guarantee that the PMU stays active even if VBUS_DET drops to a logic low voltage.
  - viii. Boston then commences normal USB operation.
  - ix. At some point, the car will be turned off and VBUS_US will drop low.
  - x. This event must be registered by Boston digital circuitry, which in turn issues a VBUS_DET interrupt (IRQ12) to the MCU.
  - xi. At this point, the firmware disables USB operation and enters LPCDWU sleep mode in which all blocks except the LPCDWU circuits are disabled and Boston is put to sleep. Also in this case, the firmware guarantees that BUCK_ENABLE continues to be asserted logic high.
  - xii. Before the firmware puts the ASIC sleep, the firmware must set the SLEEP_TIMER.
  - xiii. At some point, there will either be a wake-up event generated by the 3.3V PMU sequencer logic due to the SLEEP_TIMER or a LPCDWU_TRIP level transition.
  - xiv. At some point after, either the charging timeout limit or the charging charge limit will have been reached.
  - xv. At this point, the firmware will assert BUCK_ENABLE to logic low and the system enters the "Knockout" state.
  - xvi. At this point, the supplies collapse and eventually one or more resets trigger.

7.1.3 Modes of Operation

Breaking up the intended operation of Boston into a set of operational modes is a handy method to conceptualize which blocks and clocks should be active at any given time. A table showing a list of the blocks, their function, intended mode of operation, and current is shown below in Table 7-2.

7.1.3.1 USB Active Hub Mode (Mode 1)

USB Hub Active mode refers to operating Boston as a full USB hub not in suspend mode. In this case, the microcontroller and AyDeeKay custom digital hardware runs off of the 30MHz clock, which is required to obtain enough clock cycles to permit high-throughput USB Device-to-SD Card operations. Here, the FCET is allowed to perform CDP and SDP handshaking, but to assume no proprietary or USB DCP “wall-wart” configurations. Charge Measurement and Housekeeping ADC functionality is ongoing in this mode.

7.1.3.2 Hub Suspend Mode (Mode 2)

Hub suspend mode occurs when the USB circuitry is enabled but when the host has issued a command to suspend the hub. In this condition, the USB-compliant frequency synthesis is disabled by the USB Hub digital IP to save power for suspend mode (where USB specification [1] requires a time-averaged current draw of less than 2.5mA). When this event occurs, the Boston digital hardware implements a clock handover to the ICCLK in order to conserve power. In this condition, Charge Measurement and Housekeeping ADC functionality is still present, but now the microcontroller runs off of a divided-down version of ICCLK clock. CDP and SDP handshaking is still possible in this mode. When the upstream host signals to the hub to exit this mode, Mode 1 will be re-entered.

7.1.3.3 Charger Only Mode (Mode 3)

Charger Only mode is intended for use for a time interval occurring immediately after the vehicle has been disabled. In this case, all USB operation has ceased but the ASIC has awoken due to an LPCDWU wake-up event in order to allow charging from the car battery to one or more of the downstream ports. Again, Charge Measurement and Housekeeping ADC functionality is present and the microcontroller again runs off of a divided-down version of ICCLK.

7.1.3.4 Low Power Current Detect Wake Up (LPCDWU) Sleep Mode (Mode 4)

LPCDWU mode is the same as deep sleep mode but with the difference that about 20uA more quiescent current is drawn from the power supplies due to enabling of the LPCDWU circuits. When enough current is drawn from a downstream port on which the LPCDWU is enabled, the LPCDWU comparator will trip and issue a command to wake up the ASIC. Also during LPCDWU mode, a sleep timer is programmed with the allowable after-vehicle-off charging interval such that if an LPCDWU event never occurs, the ASIC is awoken regardless so that the firmware can release the external PMU enable signal to completely power off the Hub module.

7.1.3.5 Deep Sleep Mode (Mode 5)

Deep sleep mode is entered via a firmware command to a specific register within AyDeeKay custom digital hardware. In this case, the PMU sequencer progressively disables the PMU blocks and enters the sleep state shown in Figure 7-7. In order to achieve minimal power consumption and predictable operation, it is recommended that all analog and digital blocks be disabled by firmware where possible prior to entering this state. For all future discussions regarding the Boston ASIC, it is assumed that this has been done unless explicitly mentioned otherwise. In this case, LPCLK remains active in order to run the sleep timer and to time the PMU wake-up sequence. Exiting sleep mode is done when any wake-up event occurs or a POR or External Reset occurs.

Table 7-2 List of Functional Blocks and Modes Used

Sub-category	Block	Unit Pwr. (mA)	# Inst.	Tot. Pwr. (mA)	Function	Modes Used
Digital	USB HS HUB + DIG PHY	N/A	1	N/A	USB Signal Routing	1
	USB HS DEV	N/A	1	N/A	USB Data Sink/Source	1
	SD Card HOST	N/A	1	N/A	SD Card Controller	1
	DEV-SD Card-MCU Interface	N/A	1	N/A	Inter-block control and communication	1
	Charge Measurement DBE	N/A	1	N/A	Perform integration of ADC results.	1,2,3
	HKADC DBE	N/A	1	N/A	Manage ADC sampling without firmware intervention.	1,2,3
	Digital PMU Sequencer	N/A	1	N/A	Power up/power down PMU blocks in a timed order.	4,5
	Misc.	N/A	1	N/A	Misc.	1-3
	Total Digital	90	1	90	Digital	1
		TBD	1	TBD	Digital	2
		TBD	1	TBD	Digital	3
		0.001	1	0.001	Digital	4-5

Table 7-2 List of Functional Blocks and Modes Used

Sub-category	Block	Unit Pwr. (mA)	# Inst.	Tot. Pwr. (mA)	Function	Modes Used
Clock Sources	XTAL OSC, Freq. Mult. PLL, Dividers	10	1	10	Provide clocks required by USB PHY and SD HOST IP	1
	ICOSC	0.03	1	0.03	Low-power clock to non-USB blocks during suspend mode	1-3
	LPOSC(10 kHz Clock)	0.002	1	0.002	Ultra-low power clock during deep sleep mode.	1-5
Analog PHY	Analog PHY	27	5	115 (4TX, 1RX)	Analog USB Line RX/TX	1
	Analog PHY	N/A	5	0.1	Analog USB Line RX/TX	2
Charging Emulator	Flexible Charging Emulator Termination	Varies	4	Varies	Programmable termination V,I,R required to mimic many different chargers.	1-3
	Current Measurement Analog	0.055	4	0.22	Measure voltage across port power control switch.	1-3
	Charge Measurement ADC	0.001 (Time Avg.)	1	0.001 (Time Avg.)	Converts output of C.M. Analog	1-3
	Low Power Current Detect Wake Up	0.005	4	0.02	Wakes chip from sleep if a device is inserted and draws power.	4
HKADC	HKADC	0.001 (Time Avg.)	1	0.001 (Time Avg.)	Measures Flexible Emulator ADC outputs, PTAT voltage, etc.	1-3

Table 7-2 List of Functional Blocks and Modes Used

Sub-category	Block	Unit Pwr. (mA)	# Inst.	Tot. Pwr. (mA)	Function	Modes Used
PMU (Analog)	Bandgap Core	0.040	1	0.040	1% Reference Voltage	1-3
	Bandgap Buffers	0.009	2	0.018	Low-Out-Z Buffers	1-3
	V2I Regular	0.023	1	0.023	Reference current that tracks RPPOLY.	1-3
	V2I Trim	0.023	1	0.023	4.5% Accuracy Reference Current (after calibration)	1-3
	1.5V 15mA LDO	0.038	1	0.038	Quiet Supply for XTAL, Frequency Multiplying PLL	1-3
	2.5V 10mA LDO	0.050	1	0.050	Supply for FLASH	1-3
	1.8V 120mA LDO	0.050	1	0.050	Supply for MCU and SD Card HF Signaling	1-3
	1.5V 200mA LDO	0.050	1	0.050	Supply for Digital Core	1-3
	POR	0	1	0	Reset chip upon supply ramp	1-5
	BOR	0.003	1	0.003	Reset chip upon supply dip	1-5
MCU	MCU (30MHz)	3	1	3	Firmware Control – USB Active	1
	MCU (1 MHz)	0.1	1	0.1	Firmware Control – Low Power	2-3
Test	Test Mux Buffer	0	1	0	Buffer test signals	N/A
SD Card Interface	Power Switch	0.05	1	0.05	Provides power to SD Card	1
	Signaling Interface	35	1	35	Provides signal to SD Card	1
Pads	Pads	0	N/A	0	Area=4*sqrt(core area)*(110um+70um)	N/A

Table 7-2 List of Functional Blocks and Modes Used						
Sub-category	**Block**	**Unit Pwr. (mA)**	**# Inst.**	**Tot. Pwr. (mA)**	**Function**	**Modes Used**
Total	Total	N/A	1	255	ASIC Operation, Not Including SD Card Load Current	1

7.2 Analog Functions

7.2.1 Clock Sources and Generation

7.2.1.1 Pierce Crystal (XTAL) Oscillator

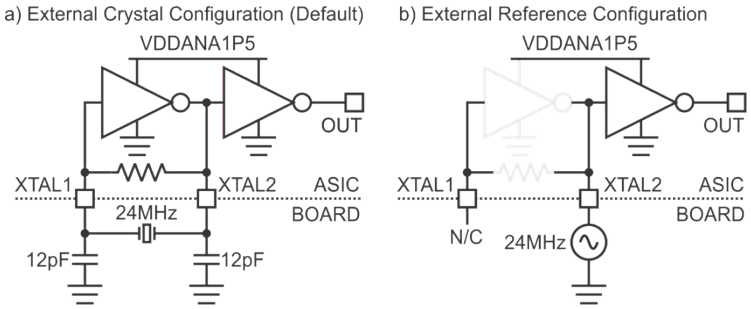


Figure 7-8 Boston XTAL Oscillator Circuit Simplified Schematic

The Pierce Crystal (XTAL) oscillator is an inverter-based circuit which uses an external crystal reference in order to generate a high-quality frequency reference at 24MHz for use in USB timing. The oscillator requires an output network composed of a 24MHz crystal (part number: Murata XRCGB24M000F3M00R0) and two capacitances each of value 12pF. Note that this 12pF should consist of the board trace routing capacitance and an explicit SMT capacitor in parallel. The XTAL oscillator can also accept an external 24MHz clock for the purposes of debugging or synchronizing with another USB module.

7.2.1.1.1 Electrical Characteristics

Table 7-3 Target Performance Pierce XTAL Oscillator

Typ: Simulated under process corner: TM, $V_{1.5V}$=1.5V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{1.5V}$=1.35V-1.65V, Temp=-40C to 125C

Name	Conditions/notes	Min	Typ	Max	Unit
Supply Current(average)		0.9	1.18	1.5	mA
Supply Current (peak-peak)		3.4	4.4	5.4	mA

Table 7-3 Target Performance Pierce XTAL Oscillator

Typ: Simulated under process corner: TM, $V_{1.5V}$=1.5V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{1.5V}$=1.35V-1.65V, Temp=-40C to 125C

Name	Conditions/notes	Min	Typ	Max	Unit
Duty Cycle		58	59	62.5	%
Standby Current		0.5	37	5350	nA

7.2.1.2 Frequency Multiplying PLL and Frequency Dividers

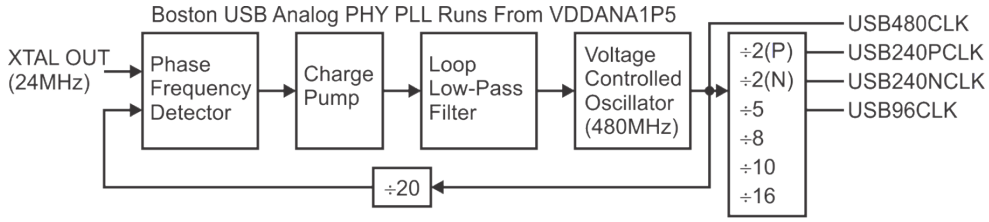


Figure 7-9 Boston Frequency Multiplying PLL and Frequency Dividers Circuit Simplified Schematic

The frequency multiplying PLL and dividers generate the high-frequency references required for full speed and high-speed USB operation. The PLL also generates an 8-phase 480MHz clock bus to operate the clock-and data recovery circuits of the USB RX PHYs. Along with the XTAL oscillator, this circuit is designed so that the frequency accuracy of the Hub complies with the USB specification [1]. The PLL dividers also generate a 96MHz clock not explicitly used by the USB Hub IP but rather by the high-speed SD Card interface. The divide-by-10 output is not a 50% duty-cycle clock but is rather a 40%/60% duty cycle clock.

This portion of the circuit is only enabled when USB is active and not in suspend mode.

In order to improve timing margin in the digital domain, the USB60CLK, USB48CLK, and USB30CLK are not sent to the digital core. Rather, they are divided down internally from the USB240PCLK so that the digital physical implementation tools can better manage timing between all of the different clock domains of the digital core.

7.2.1.2.1 Electrical Characteristics

Table 7-4 Target Performance PLL and Frequency Dividers

Typ: Simulated under process corner: TM, $V_{1.5V}$=1.5V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{1.5V}$=1.35V-1.65V, Temp=-40C to 125C

Name	Conditions/notes	Min	Typ	Max	Unit
Supply Current(average)		7.5	8.2	9.0	mA

7.2.1.3 IC Oscillator (ICOSC) (4MHz)

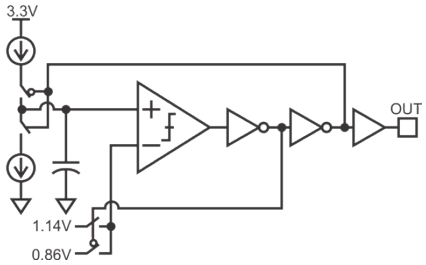


Figure 7-10 Boston IC Oscillator Circuit (ICOSC) Simplified Schematic

The IC Oscillator (ICOSC) is a low-power circuit which generates a semi-accurate on-chip clock (ICCLK) for use when the USB XTAL Oscillator and Frequency-Multiplying PLL are not active. The IC Oscillator requires calibration during production testing. (During initial development this can be done by hand but it is likely that the default code will provide acceptable accuracy for a USB-only solution). Once calibrated, it achieves an accuracy of +/- 4% from an absolute frequency of 4MHz over temperature and supply voltage changes.

The IC oscillator clock (ICCLK) can also be used to obtain a better estimate of the LP Oscillator clock (LPCLK) frequency so that that latter can be used as a more-accurate sleep timer reference while in LPCDWU sleep mode. Also, this sleep timer is continued to be used after the ASIC has woken up and entered the Charging Mode. Once the sleep timer has expired, the after-vehicle-off charging interval is expired and the ASIC firmware powers off the entire Hub module. Hence, the accuracy of the IC oscillator affects the timing accuracy of this interval.

The accuracy of the IC oscillator also affects the accuracy of the Charge Measurement subsystem when the ASIC is not in the USB Active Mode. This is because a windowed integrator circuit (whose gain is proportional to the period of the ADCCLK) is utilized for the gain/filtering functionality of this subsystem.

7.2.1.3.1 Electrical Characteristics

Table 7-5 Target Performance IC Oscillator (ICOSC)

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current		23.8	28.4	35	µA
Sleep Current		0.8	0.9	14.5	nA
Startup Time				100	µs
Disable Time			2.2	3.0	ns
Absolute Accuracy				+/- 4	%
Duty Cycle		50	51	52	%

7.2.1.4 Ultra-Low-Power Oscillator (LPOSC) (10kHz)

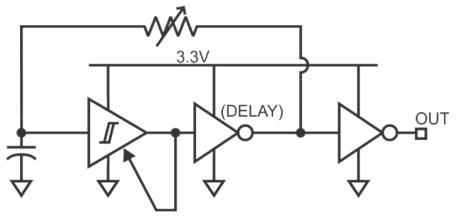


Figure 7-11 Boston Ultra-Low-Power Oscillator (LPOSC) (10kHz) Circuit Simplified Schematic

The Ultra-Low-Power Oscillator (LPOSC) is an uncalibrated RC oscillator that draws near-uA current but whose output clock (LPCLK) frequency varies substantially over process, voltage, and temperature. The LPCLK is used primarily to operate the 3.3V PMU Sequencer and Reset Circuitry and to provide a backup clock for the core digital circuitry. Hence, the total startup time of the ASIC will vary along with the actual frequency of this circuit. Furthermore, this circuit is used to run the sleep timer, although the effective accuracy of this sleep timer can be improved by taking advantage of a feature in the digital circuitry in which the number of DCLK samples in a given LPCLK sample are counted and then used to modify the value of the sleep counter accordingly.

7.2.1.4.1 Electrical Characteristics

Table 7-6 Target Performance Ultra-Low-Power Oscillator (LPOSC) (10kHz)

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit

Table 7-6 Target Performance Ultra-Low-Power Oscillator (LPOSC) (10kHz) **Typ:** Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C **Min/Max:** Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, Temp=-40C, +125C					
Name	**Conditions/notes**	**Min**	**Typ**	**Max**	**Unit**
Quiescent Current		0.56	1.1	2.27	µA
Output Frequency		6.2	10.0	16.3	kHz
Startup Time		145		350	µs

7.2.2 USB Hub Analog PHY

7.2.2.1 Top Level

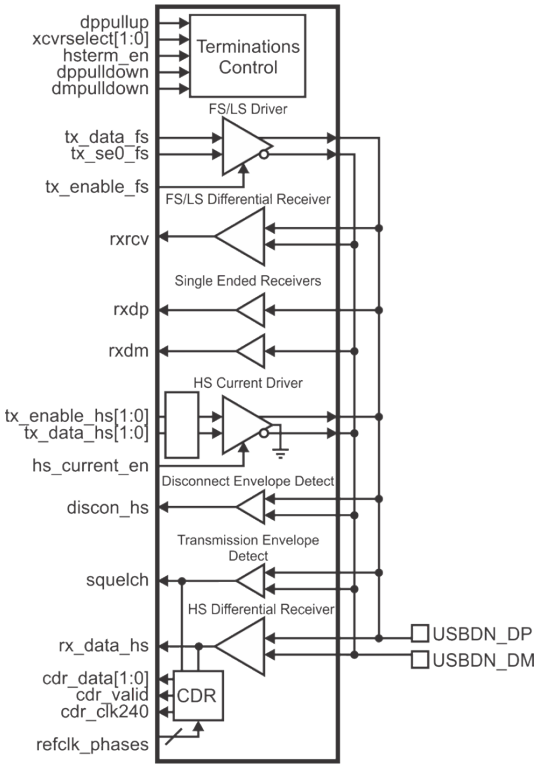


Figure 7-12 Boston Analog PHY Block Diagram

The USB Hub Analog PHY provides the interface between the complex digital USB functionality of Boston and the external line load. The Analog PHY is designed in conjunction with the digital portion of the USB Hub to be fully compliant (with the exception of suspend current) with the USB 2.0 Specification. Furthermore, the Analog PHY is capable of withstanding indefinite shorts of the USB DP and DM signaling lines to either VBUS or ground after the 3.3V supply of the ASIC is enabled.

Due to the 10% accuracy required on the USB DP/DM 45Ω signaling termination resistors, calibration is performed on their values during production testing.

The current output of the HS PHY can be increased by up to 10% of the nominal current value via trim memory setting, but doing so may result in USB transmission eye diagram specifications not being met.

7.2.2.2 Terminations Control

The Terminations Control block controls the placement of the following resistors on the USB DP and DM lines:

- 1.5kΩ signaling resistor on the USB upstream port DP line to indicate to the host that this hub is capable of Full-Speed/High-Speed Operation.
- 15kΩ pull down resistors on both USB DP and DM lines that provide a leakage path for residual charge on the signaling lines.
- Applying the calibrated 45Ω resistors as a grounded termination for High-Speed signaling.

7.2.2.3 FS/LS Driver

The FS/LS driver is a slew-rate controlled logic-level (i.e. 0V to VDD3P3) buffer utilized for LS and FS signaling up to 12MHz.

7.2.2.4 FS/LS Differential Receiver

The FS/LS receiver is a differential comparator with a wide input common mode range.

7.2.2.5 Single-Ended Receivers

The single-ended receivers are utilized to detect the SE0 signaling condition – i.e. when both USB DP and DM signal lines are low. This block contains circuitry to ignore portions of the signaling where both USB DP and DM nets are close to the signaling midpoint during chip transitions.

7.2.2.6 High-Speed Current Driver

The High-Speed current driver toggles a 17.78mA current source at 480MHz between the USB DP and DM pins while the 45Ω calibrated termination resistors are connected between the USB DP/DM pins and ground.

7.2.2.7 Disconnect Envelope Detector

The disconnect envelope detector is used in HS mode to determine when a cable has been disconnected. Since each side of the cable is terminated in 45Ω, removing one side causes an approximate doubling of signal amplitude at the transmitting end, which can be detected with this circuit.

7.2.2.8 Transmit Envelope Detector

The transmit envelope detector is used in HS mode to determine when the device, hub, or host on the other end of the cable has ceased communicating. This is done by determining that the envelope has dropped below a threshold of about 150mV.

7.2.2.9 High Speed Differential Receiver

The high speed differential receiver is a high-speed comparator designed to accept the +/-400mV signaling level of USB HS communications (along with cable attenuation) with significant changes in signaling common mode level.

7.2.2.10 CDR DLL

Because of differences in the crystal frequencies between two ends of a USB communications link, the clock of the other end of the communication link must be recovered. In order to do this extremely rapidly, an 8-phase DLL that uses the 8 clock phases generated by the Frequency Multiplying PLL is utilized in this design.

7.2.2.11 VBUS Comparator

The VBUS comparator (not shown in Figure 7-12) is utilized on the upstream port to determine if the upstream VBUS is present. This circuit assumes that the 4.3V-5.25V VBUS signal has been divided down by 2 in voltage prior to connection to the Boston ASIC. This circuit is implemented by a non-Schmitt logic buffer input pad structure with logic level inputs VOL=0.8V and VOH=2.0V. Simulation shows that the threshold of the input pad logic buffer is between 1.05V and 1.6V over process, temperature, and corner variation.

7.2.2.12 Electrical Characteristics

Table 7-7 Target Performance USB Analog PHY (numbers below are for 1 PHY)

Typ: Simulated under process corner: TM, $V_{1.5V}$=1.5V , $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Sim'd under corners: TM, WS, WP, WO, WZ, $V_{1.5V}$=1.35V-1.65V ,$V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V,Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	TX in FS Operational Mode (CL=150pF)			22.36	mA
Quiescent Current	TX in LS Operational Mode (CL=600pF)			5	mA

Table 7-7 Target Performance USB Analog PHY (numbers below are for 1 PHY) **Typ:** Simulated under process corner: TM, $V_{1.5V}$=1.5V , $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C **Min/Max:** Sim'd under corners: TM, WS, WP, WO, WZ, $V_{1.5V}$=1.35V-1.65V ,$V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V,Temp=-40C, +125C					
Name	**Conditions/notes**	**Min**	**Typ**	**Max**	**Unit**
Quiescent Current	TX in HS Operational Mode			22	mA
Quiescent Current	RX in HS Operational Mode			2.4	mA
Quiescent Current	RX in FS/LS Operational Mode			0.1	mA
Quiescent Current	USB Analog PHY Local Digital (i.e. CDR, decoding) during USB Operation			5.2	mA

7.2.3 Charge Measurement

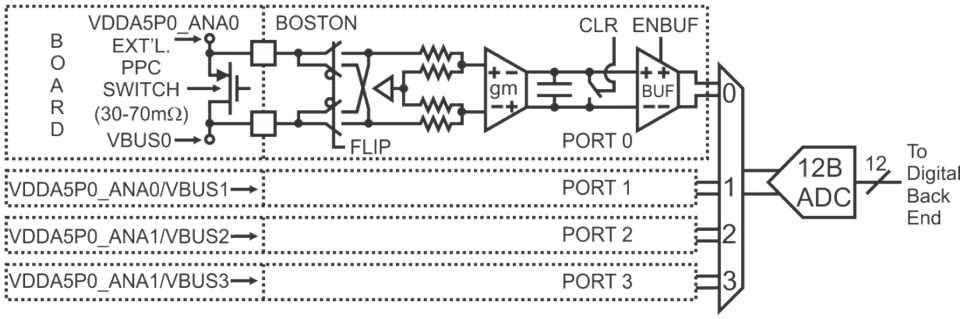


Figure 7-13 Boston Charge Measurement Analog Circuit Simplified Schematic

7.2.3.1 Description and Interface

The USB protocol is no longer used merely for transmission of information - it is now the primary interface by which the wired charging of mobile devices takes place. Charging takes place from the 5V "VBUS" line in the 4-wire USB interface. Boston implements per-port power control on each of 4 downstream USB ports.

There will exist a Charge Measurement feature on Boston by which the total charge drawn per port will be measured over a programmably-defined time interval not to exceed 3 hours. The purpose of this feature is to limit the amount of charge that can be drawn off of the car battery after the car alternator has been turned off. Boston is required to disable all USB downstream port charging after the first of: elapsing of the aforementioned time interval OR reaching a programmable charge limit of 1-20 Amp-

hours.

Per-port power control is implemented with a power MOSFET between a 5V master supply and the downstream "VBUS" line. This power control is implemented by an independent circuit residing external to Boston. The target device to be used for this purpose is the Texas Instruments TPS2560.

Boston will use the voltage measured across this power switch in order to infer the current flowing through it. Because the on-resistance of this switch varies from part-to-part and as a function of temperature, Boston will have a Digital Back End for this circuit containing a flexible digital signal processing (DSP) unit in order to utilize a thermal model of the Hub module to recompute the current from the ambient temperature and estimated power dissipation within the power switch. Furthermore, the specified charge measurement accuracies listed below assume that the resistance of the switch as a function of temperature is known perfectly and that any deviation from this assumption impacts the accuracy accordingly.

The use of the port power control switch as a current-measuring device presents an additional complication in that the current-limiting mechanism of the port power control permits current draw for an arbitrary voltage range on the downstream VBUS. In this case, VBUS may range between ~5V and 0V while the current remains the same. This is handled in the Charge Measurement DBE by setting a ceiling on the estimated current equal to the nominal value of the port power control current limit.

The input interface to each of the 4 Charge Measurement circuits will be the 5V supply line and the corresponding VBUS line. It is recommended that for the most accurate results, star-routed sense lines from the port power control switch to the 5V supply pins/VBUS pins on Boston be made.

The output interface of each of the 4 Charge Measurement Analog circuits is an algorithmic ADC with 12 bits of resolution and 10-bit linearity. The ADC will take samples (proportional to input current) of each of the 4 Charge Measurement Analog circuit outputs at a period of about 1024µs. Each sample will be initiated by a pulse generated by the Charge Measurement Digital Back End circuitry that is synchronized with the 1MHz ADCCLK and is 1 cycle wide. The results of this sampling will be processed and then integrated in digital circuitry in order to compute Amp-hours to be compared to a limit. This ADC will be shared between all 4 Charge Measurement Analog circuits but not shared with the remainder of Boston. This ADC will operate from the ADCCLK.

7.2.3.2 Coarse Requirements / Features

- The complete solution is required to measure charge integrated over a time interval of up to 3 hours and compare this total charge to a limit of 1-20 Amp hours.
- This comparison should be done with an error of no more than 25%.
- The minimum charging current limit allowed in USB is 100mA. Integrating this over 3 hours on all 4 ports yields a total charge of 1.2 Amp-hours.

- Given the above, it is desired to measure 100mA accurately on a per-port basis. In the case where the switch resistance is as low as 30mΩ, the input voltage to the Charge Measurement circuit is 3mV.
- The maximum charging current conceivably allowed per port in this system is 3A. In the case where the switch resistance is as high as 70mΩ, the input voltage to the Charge Measurement circuit is 210mV.
- Put the Gm-C differential output on the analog test bus.
- Put the buffer differential output on the analog test bus.
- Implement capability to short inputs of Gm-C cell and buffers via register settings.
- Implement capability to open all chopping switches and short input of resistor divider via register settings.
- Tolerate a VBUS range of 5.5V to 0V.

7.2.3.3 Architecture

7.2.3.3.1 Input Chopping Switches

In order to remove the offset of the analog circuitry, chopping will be performed on the entire analog signal path, namely, at the analog input and digital output (after ADC conversion). Chopping will be performed on a sample-by-sample basis. That is, an ADC sample will be taken with the input chopping switches in one configuration and a subsequent sample will be taken in the other configuration. The second sample will be subtracted by 2^{12} in digital in order to "chop" it and the result will be added to the charge integration.

The Input Chopping Switches will therefore need to switch quickly after each ADC sampling interval. This will be done during the Gm-C capacitor clearing (explained below), which should take no more than 1us.

The input chopping switches will need to operate at the VBUS voltage (from 0V to 5.5V) without breaking down the constituent 3.3V devices.

7.2.3.3.2 Resistive Division-By-2/3

In order to support a constant-Gm circuit (explained below) over an input common mode range of 0V to 5.5V, a resistive division-by-2 is utilized. Although this increases the total input-referred offset, the total input-referred offset is still considerably less than 50mV. This divide-by-2 circuit will be composed of resistors about 1MΩ each for a total current draw of about 4*2*5V/(2e6)=20uA for all 4 ports.

For larger input voltages, a programmable division-by-3 mode is also available. The additional division is accomplished by placing a resistance in between the two divided-down voltage nodes of the divide-by-2 circuit, thus accomplishing an additional differential voltage division but not an additional common-mode voltage division. This division-by-3 mode will be the trim register default setting.

7.2.3.3.3 Gm-C Integration Cell

A Gm-C Integration Cell will perform short-term integration of the voltage measured across a 30mΩ-70mΩ measurement resistor. The advantage of this cell is that (in the absence of blind spots) in the frequency domain, there exist nulls at multiples of the sampling rate, preventing aliasing of high-frequency signals directly to DC where our signal of interest lies. This allows the overall architecture to dramatically reduce the sampling burden on the ADC, which by design is limited to only taking samples once every 25us and must be multiplexed among the 4 Charge Measurement Analog circuits.

The integration capacitor should be tied in a fixed floating fashion to the differential lines leading from the Gm-cell outputs to the buffer inputs. A clearing switch will also be placed across this capacitor to clear the capacitor at the end of each integration interval.

7.2.3.3.4 Gm-C Calibration

The voltage gain of the Gm-C cell, which is equal to gm/(C*Fs) (where Fs≈1MHz), needs to remain roughly constant. The reason is that gm and C can each vary up to +/- 15% and together this variation can eat into the output voltage swing by about 50%.

Biasing for the Gm-cell is derived from a constant-Gm cell with the constant-Gm proportional to some R. The R will be trimmed based on a mirrored copy of the Gm-cell bias current in order to avoid wildly varying bias current (the bias current varies as the square of R).

Fs will be calibrated as part of the ICOSC calibration procedure for the times in which the USB30CLK is unavailable.

After these calibrations, C must be calibrated. This will be done in production by applying multiple known voltages to the input of the Charge Measurement Circuit and by determining the output of the ADC. This will provide gain and offset errors of the system. The gain error will be compensated for by trimming the value of C. The offset will be removed by chopping instead of calibration since it has multiple contributors.

7.2.3.3.5 ADC Buffer

The ADC Buffers are required to drive the input capacitors of the ADC. This ADC buffer need not be active all of the time, only near the time at which the ADC must sample its output. However, in order to simplify the design, the ADC buffer is made active all of the time that the Charge Measurement circuit is active.

7.2.3.3.6 Clock Generation

The Clock Generation will receive a 1MHz ADCCLK clock signal that should be buffered and passed through to the ADC.

The Clock Generation should control the following:

- Turn the ADC buffer on and off at the appropriate times.
- Send a strobe-in (STRBI) signal to the ADC.
- Keep the buffer output valid for the 1MHz clock cycle upon which STRBI is asserted and for 2 additional 1MHz clock cycles thereafter.
- Clear the integration capacitor for a particular measurement circuit after sampling of that measurement circuit by the ADC.
- Rotate the connections from each of the 4 buffers to the ADC.
- Provide at least 2 bits out indicating the active connection of the ADC. Keep this signal valid until after ADC STRBO is de-asserted.
- Provide at least 1 bit out indicating the chopping state. Keep this signal valid until after ADC STRBO is de-asserted.

7.2.3.3.7 Digital Back End

The digital circuitry is responsible for the following:

- Maintaining 4 >35-bit registers, one for each measurement circuit, to hold the accumulated charge values. Why 35 bits? Because about 2048*(1000)*60*60*3 < 2^35 measurements are made over 4 hours.
- Adding all 4 registers up and comparing the result to a threshold read into a local register read in from firmware.

- Reading the state of the clock generation circuit and flipping the digitized result if called for and integrating the appropriate register with the possibly flipped result.
- Throwing an interrupt if the total charge measurement exceeds the programmed threshold.
- Making all 4 accumulation registers visible to firmware.
- Making the instantaneous current measurement (the last 4 ADC outputs) visible to firmware. Each of these registers should map to one charge measurement circuit.
- Compensate for the temperature variation of the port power control switch resistor.

More information regarding the digital back end of this subsystem is detailed in the digital portion of this specification.

7.2.3.4 Electrical Characteristics

Table 7-8 Target Performance Charge Measurement

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C

Assuming use of the TPS2560 and that switch resistance stays bounded within 30mΩ-70mΩ from part to part and temperature.

Assuming overcurrent conditions are not allowed to persist for cumulative time intervals of >1s. When an overcurrent condition exists on one port of a two-port PPC switch, the other port may see up to a 20%-low current estimate during that time.

Assuming perfect Hub module / PPC ASIC / Boston ASIC thermal circuit model in Charge Measurement DBE DSP.

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	Per 4 Ports Instantiation, Time-Averaged		165	215	µA
Sleep Current	Per 4 Ports Instantiation			60	nA
Input Resistance	Per port, at VBUS, divide by 3 mode	1.1	1.4	1.7	Mega-Ω
CM Rejection	Maintain accuracy with 100mVpp ripple from 100kHz-1MHz.				
PS Rejection	Maintain accuracy with 100mVpp ripple from 100kHz-1MHz.				

Table 7-8 Target Performance Charge Measurement

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C

Assuming use of the TPS2560 and that switch resistance stays bounded within 30mΩ-70mΩ from part to part and temperature.

Assuming overcurrent conditions are not allowed to persist for cumulative time intervals of >1s. When an overcurrent condition exists on one port of a two-port PPC switch, the other port may see up to a 20%-low current estimate during that time.

Assuming perfect Hub module / PPC ASIC / Boston ASIC thermal circuit model in Charge Measurement DBE DSP.

Name	Conditions/notes	Min	Typ	Max	Unit
Charging Current Measurement Range		100		3000	mA
Charging Charge Measurement Range		1		20	Amp-hours
Trip Charge Accuracy	For any number of activated ports.	0.8* I_{limit}	I_{limit}	1.25 * I_{limit}	Amp-hours

7.2.4 Flexible Charging Emulator Termination (FCET)

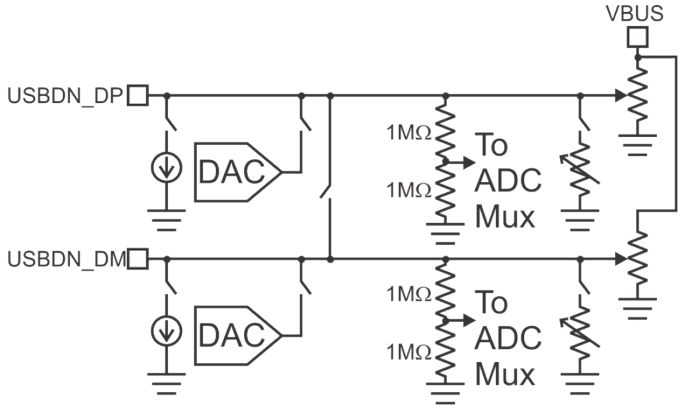


Figure 7-14 Boston Flexible Charging Emulator Termination Top Level Block Diagram

7.2.4.1 Description / Interface

Ever since the advent of USB, many companies have found the USB connector to be a convenient charging interface for their device, whether the device was fully USB-compliant or not. Companies also realized that wall-socket power adaptors were capable of providing far more current than the 500mA allotted by the USB specification [1]. For that reason, they developed power adaptors that provided far higher charging current but all contained different proprietary terminations to evidence compatibility with a certain class of device.

To put an end to the resulting mish-mash of USB charging adaptor terminations seen on the market, the USB-IF introduced the Battery Charging Specification (BCS) [2], which increased the maximum charging current for different classes of devices to 1.5A. However, because even this level of current is insufficient for some OEM demands and for purposes of product differentiation, many OEMs continue to use and introduce new proprietary terminations.

In order to future-proof Boston for new proprietary terminations, a Flexible Charging Emulator Termination (FCET) is specified. The FCET contains five different subsystems:

- Programmable Resistive Divider and Pull-Down Resistor (for SDP and Vendor-Specific Dedicated Charging Port (VS-CDP) Configurations)
- Terminal Short (for USB Dedicated Charging Port (USB-DCP) Configuration)
- Programmable Voltage Source (for Charging Downstream Port (CDP) Configuration)
- Programmable Current Sink
- ADC Access (for Charging Downstream Port (CDP) Configuration)

The FCET interfaces with the USBDN_DP and USBDN_DM pins and the VBUS line for a given downstream port. The FCET also interfaces with the 5V supply rail in order to provide operation up to 5V and protection against shorts of the USBDN_DP and USBDN_DM pins to VBUS. Any combination of FCET subcircuits may be enabled at a given time, although many combinations are nonsensical. When a given FCET subcircuit is disabled, it is placed into a high-impedance state, but may draw a small (~100nA) amount of leakage current if the voltage at USBDN_DP or USBDN_DM were to rise above the VDD3P3 supply voltage.

7.2.4.2 Programmable Resistive Divider and Pull-Down Resistor

7.2.4.2.1 Description / Interface

The Programmable Resistive Divider and Pull-Down Resistor are implemented as the same circuit in Boston. The reason is that a pull-down resistor array that has been designed in a particular fashion can be mirrored as a pull-up array to VBUS and the combination of the two can be shown to cover all of the Resistive Divider requirements.

7.2.4.2.2 Coarse Requirements / Features

Resistive Pull Down:

- Programmable Range: 1.5kΩ-200kΩ.
- Range Step Coverage: Finer than 5-bit logarithmic.
- Absolute Resistor Value Accuracy (Deviation from Nominal): Less than 25%.
- Protect against USBDN_DP/USBDN_DM short to VBUS or ground.

Resistive Divider:

- Programmable impedance between VBUS and ground: 15kΩ-200kΩ.
- Absolute Resistor Value Accuracy (Deviation from Nominal): Less than 25%.
- Voltage Division Accuracy (Deviation from Nominal): Less than 5%.
- Programmable Divider Ratio: Finer than 4 bit, 0% to 80% of full scale.
- Prevent large leakage from USBDN_DP/USBDN_DM lines if VBUS→0V.
- Protect against USBDN_DP/USBDN_DM short to VBUS or ground.

7.2.4.2.3 Architecture

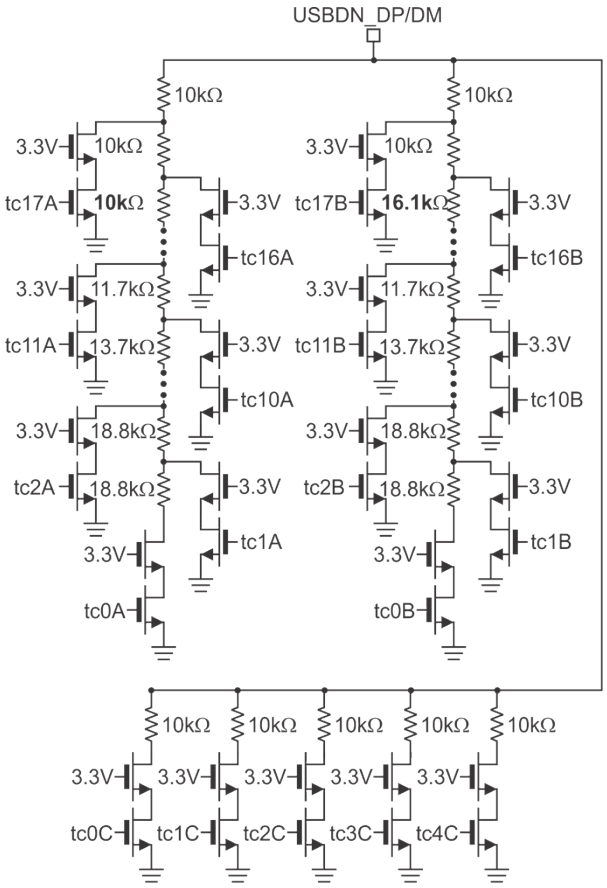


Figure 7-15 Boston Programmable Resistive Divider Circuit (Pull-Down Portion Shown Only) Simplified Schematic

The architecture of the programmable pull-down resistor is shown in Figure 7-15. Signals tcXX (where XX represents a variable string) represent the digital logic outputs of a binary-to-thermometer decoder. One can see that all switches are protected against a short of USBDN_DP or USBDN_DM to VBUS while enabled or disabled and that the value of the resistance does not significantly change as USBDN_DP or USBDN_DM approaches 5V provided that the switch device W/L ratio is made large enough. Note that the pull-up branch does not have the section of 10kΩ resistors switched by signals tcXC but does have the sections switched by tcXA and tcXB.

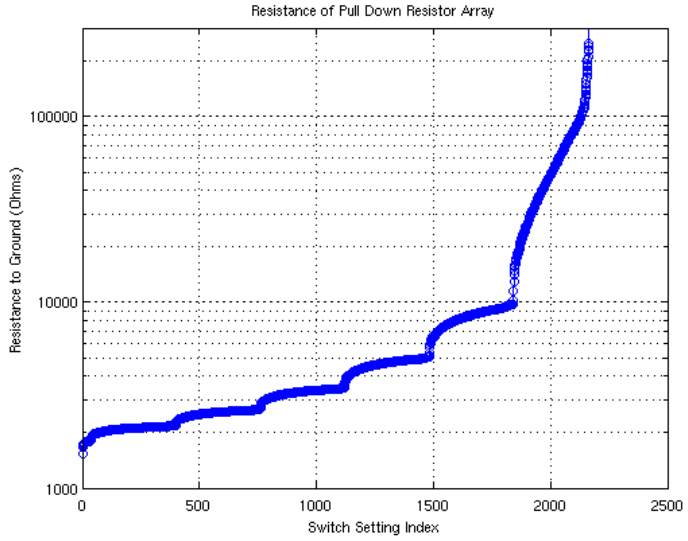


Figure 7-16 Boston Programmable Pull-Down Resistor Value Coverage (Nominal).

The resistor value coverage of the proposed programmable pull-down resistor is shown in Figure 7-16. It is verified that the maximum resistor step size at any point is less than 13.6% of the previous resistor value. This represents an improvement over a 5-bit logarithmically-spaced array in which the maximum resistor step size at any point is less than 17.1% of the previous resistor value.

Choosing the correct resistor array settings to achieve a particular pull-down resistor or a particular resistive divider configuration can be done with the assistance of AyDeeKay or an AyDeeKay-provided computer script.

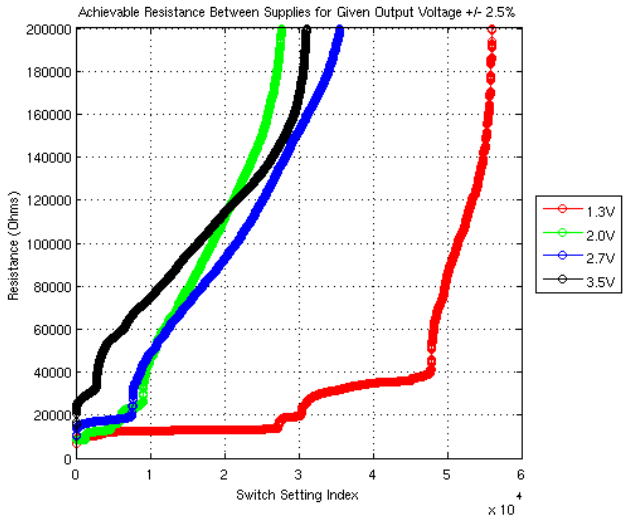


Figure 7-17 Boston Programmable Resistive Divider Total Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.3,2.0,2.7,3.5V)

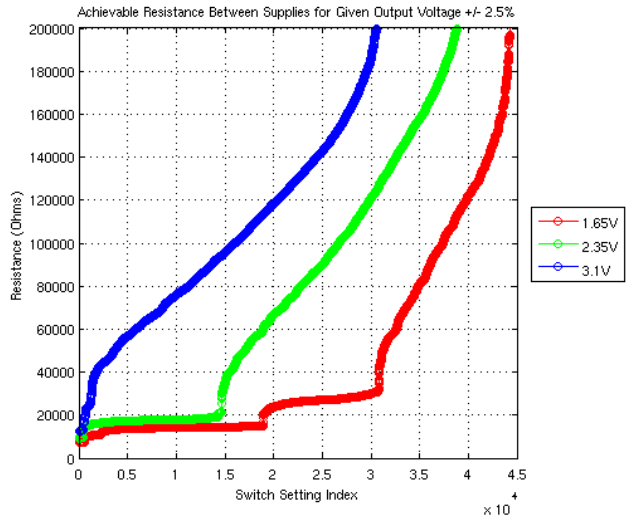


Figure 7-18 Boston Programmable Resistive Divider Total Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.65V,2.35V,3.1V)

The coverage of the resistance between the VBUS supply and ground for the proposed programmable resistive divider is shown in Figure 7-17 and Figure 7-18. The plots were obtained by finding every resistor combination that produced the desired output voltage +/- 2.5%. The maximum impedance step from 20 kΩ up to 200kΩ from the previous array for the curves shown in these plots is as follows:

- 1.3V (Minimum customer original specification of 25% full scale VBUS voltage): 9.4%
- 1.65V: 8.8%
- 2.0V (Apple Charger Resistive Divider Value Type 1): 8.4%
- 2.35V: 20.0%
- 2.7V (Apple Charger Resistive Divider Value Type 2): 12.8%
- 3.1V: 6.3%
- 3.5V: (Near Sony Resistive Divider Value): 3.9%

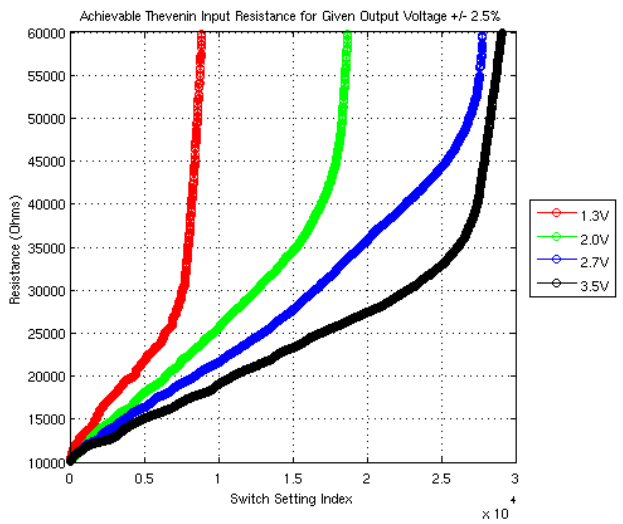


Figure 7-19 Boston Programmable Resistive Divider Thevenin Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.3,2.0,2.7,3.5V)

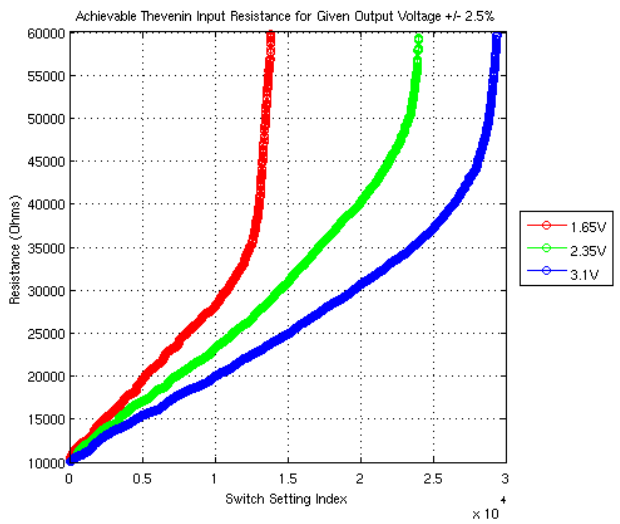


Figure 7-20 Boston Programmable Resistive Divider Thevenin Resistance Coverage for Given Output Voltage (VBUS=5.15V, VOUT=1.65V,2.35V,3.1V)

Perhaps more important than the resistance between VBUS and ground is the Thevenin impedance between the USBDN_DP/DM terminals and AC ground. The reason is that a device sensing what it is plugged into can only sense the Thevenin output voltage and impedance of the network it is connected to. The coverage of the Thevenin resistance between the USBDN_DP/DM pin and AC ground for the proposed programmable resistive divider is shown in Figure 7-19 and Figure 7-20. The maximum Thevenin impedance step up to 60kΩ from the previous array for the curves shown in these plots is as follows:

- 1.3V (Minimum customer original specification of 25% full scale VBUS voltage): 4.1%
- 1.65V: 2.0%
- 2.0V (Apple Charger Resistive Divider Value Type 1): 1.1%
- 2.35V: 1.7%
- 2.7V (Apple Charger Resistive Divider Value Type 2): 1.5%
- 3.1V: 0.7%
- 3.5V: (Near Sony Resistive Divider Value): 0.8%

7.2.4.2.4 Electrical Characteristics

Table 7-9 Target Performance FCET Resistive Divider and Resistive Pull-Down

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent/Sleep Current	Per 1 Ports Instantiation	13		56	nA
Leakage to USBDN_DP/DM	Output=GND, all switches off.			15	nA
Leakage from USBDN_DP/DM	Output=VDDA5P0, all switches off.			6	nA
Voltage Division Accuracy (Deviation From Nominal Value)	Worst-case, verified in MATLAB for Vout,nominal=0.25*VBUS to 0.6*VBUS			5	%
Resistor Value Absolute Accuracy	R_{nom}=Nominal Resistor Value for a Given Setting	0.8* R_{nom}	R_{nom}	1.25 * R_{nom}	Ω

7.2.4.3 Terminal Short

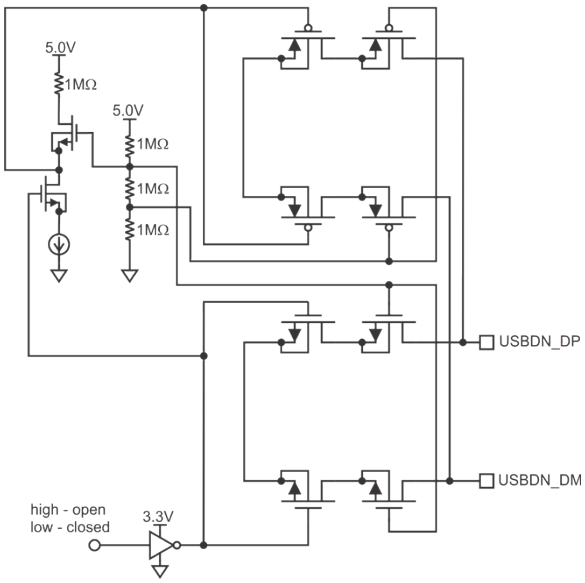


Figure 7-21 Boston FCET Terminal Short Simplified Schematic

7.2.4.3.1 Description / Interface

The terminal short between USBDN_DP and USBDN_DM pins satisfies the requirements of the USB BCS 1.2 requirement to provide a termination for a DCP. The terminal short also satisfies the requirements of the Chinese Telecommunication Industry Standard YD/T 1591-2009.

7.2.4.3.2 Coarse Requirements / Features

The terminal short must provide less than a 200Ω impedance across all corners, temperatures, and supply voltage across the USBDN_DP and USBDN_DM terminals from an arbitrary voltage ranging from 0.4V to (VDD3P3) on USBDN_DP and 0.4V to (VDD3P3) on USBDN_DM without inducing device oxide voltage reliability violations.

7.2.4.3.3 Architecture

To the external world, this feature will merely look like a switch across the USBDN_DP and USBDN_DM terminals.

7.2.4.3.4 Electrical Characteristics

Table 7-10 Target Performance FCET – Terminal Short

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	Per 1 Port Instantiation			1	µA
Sleep Current	Per 1 Port Instantiation	10		200	nA
Net current leakage to/from terminals (DC)	Per 1 Port Instantiation, Switch On			1	µA
Current leakage between terminals (DC)	Switch Off			200	nA
Net current leakage to/from terminals (DC)	Switch Off			200	nA
USBDN_DP/DM shorting resistance				200	Ω
Operating Range	USBDN_DP and USBDN_DM voltage over which the aforementioned metrics apply.	0.4		V3.3	V
Off-state cross channel isolation between USBDN_DP and USBDN_DM	Assuming that both USBDN_DP and USBDN_DM are terminated by 45Ω // 5pF to ground. Simulated at frequencies less than 250MHz.		40		dB

7.2.4.4 Programmable Voltage Source (DAC)

7.2.4.4.1 Description / Interface

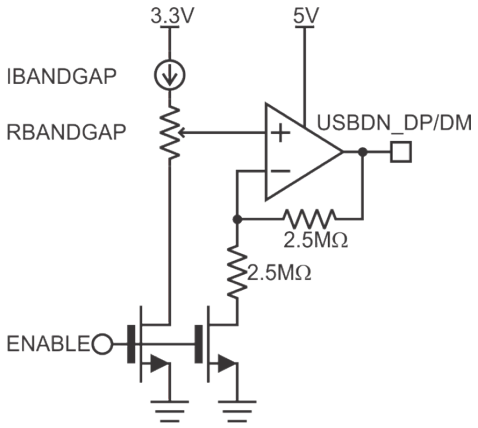


Figure 7-22 Boston FCET Voltage Source (DAC) Simplified Schematic

A programmable voltage source is included on the USBDN_DP and USBDN_DM lines of each downstream port in order to satisfy the requirements of USB Battery Charging Specification (BCS) [2] CDP Mode handshaking. The voltage source is made programmable in order to accommodate future revisions of the BCS and also to potentially support proprietary charging protocols that require voltage sources.

7.2.4.4.2 Coarse Requirements

- Output range 0.4V to 3.5V
- Withstand an indefinite short to VBUS or ground, including when disabled.
- Output voltage step: 100mV.
- Output accuracy (deviation from nominal value): 10% of the output voltage.
- Supports load capacitances of 5pF to 1nF.
- <3ms settling time after being enabled, <2ms disabling time. This is sufficient to meet the 10ms voltage source power up time requirement in the USB Battery Charging Specification [2] CDP handshaking.
- Current sink/source ability > 250µA to satisfy requirement of USB Battery Charging Specification.
- Current sink/source limiting to +/-400µA to satisfy requirement of USB Battery Charging Specification.

7.2.4.4.3 Architecture

As depicted in Figure 7-22, the Boston FCET DAC is a bandgap-referenced resistor-string DAC buffered by a gain-of-2 opamp. The opamp is a class-AB opamp, meaning that its quiescent current is

much less than its output current sink and source capability. The opamp also has internal current limiting, preventing it from pulling the USBDN_DP or USBDN_DM line low if a legacy USB device were to be connected to the downstream port with a fixed pull-up resistor.

7.2.4.4.4 Electrical Characteristics

Table 7-11 Target Performance FCET – Voltage Source (DAC)

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	Per 1 Port Instantiation (2 DACs)			12.5	µA
Sleep Current	Per 1 Port Instantiation (2 DACs)			23.5	nA
Output Voltage Range	Steps of 100mV	0.4		3.5	V
Output Voltage Accuracy (Deviation from Nominal Value)	Relative to the output voltage.			10	%
Compatible load capacitances		5		1000	pF
Power Supply Rejection	For triangular voltage ripple waveforms on VDDA5P0V	16			dB
RMS Output Noise				15	mV

7.2.4.5 Programmable Current Sink

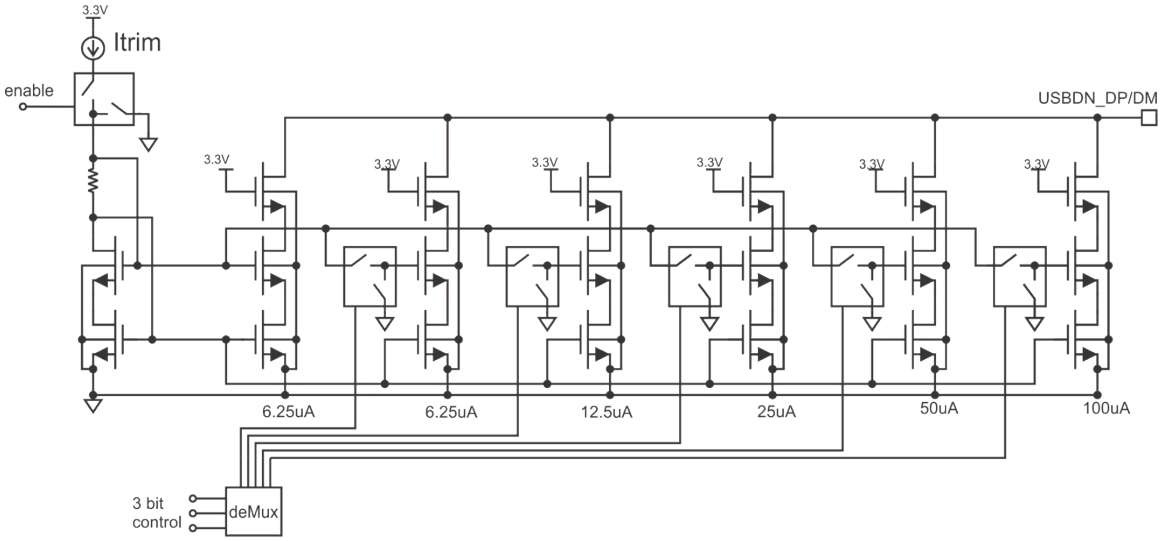


Figure 7-23 Boston FCET Programmable Current Sink Simplified Schematic

7.2.4.5.1 Description/Interface

The FCET programmable current sink is a current sink ranging in value from 6.25µA to 200µA from each of the downstream port USBDN_DP/DM pins to ground.

7.2.4.5.2 Coarse Requirements

- Supports a maximum voltage of 5.5V and a minimum voltage of 0.4V over which current accuracy is maintained.
- Supports output current values of 6.25µA, 12µA, 25µA, 50µA, 100µA, 200µA.
- Achieves an accuracy of better than +/-10% from the desired output current.
- Achieves a small-signal output impedance greater than 0.5MΩ for output voltages greater than 0.6V at 200µA sinking current setting.

7.2.4.5.3 Architecture

The architecture of the programmable current source is shown in Figure 7-23.

7.2.4.5.4 Electrical Characteristics

Table 7-12 Target Performance FCET – Current Sink **Typ:** Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C **Min/Max:** Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C					
Name	**Conditions/notes**	**Min**	**Typ**	**Max**	**Unit**
Quiescent Current	Per Port.			3	µA
Leakage Current	Per Port.			2.5	nA
Output Voltage Range	Range over which specified current accuracy is achieved.	0.4		5.5	V
Output Current Accuracy (deviation from nominal value)	Relative to the desired current.			10	%
Settling Time				0.2	ms
Output Impedance	For output voltages greater than 0.6V. For a DC output current of 200µA	0.5			MΩ

7.2.4.6 ADC Access

7.2.4.6.1 Description and Interface

The ADC access portion of the FCET allows the Housekeeping ADC to access the USBDN_DP and USBDN_DM pins of all of the downstream ports without significantly impacting the existing loading on the USBDN_DP and USBDN_DM lines.

7.2.4.6.2 Coarse Requirements

- Permit 10-bit linearity sampling by the ADC of the USBDN_DP and USBDN_DM lines on downstream ports.
- Provide USB-specification compliant isolation of the ADC from the USBDN_DP and USBDN_DM lines.
- Provide protection to the ADC against 5V shorts.
- Rail-to-rail operation relative to the USBDN_DP/DM lines.

7.2.4.6.3 Architecture

The ADC Access is a simple resistive divide-by-2 network from the USBDN_DP and USBDN_DM lines to ground. The resistors in this network are 1 MΩ apiece for a total leakage current of

V(USBDN_DP/DM)/2 MΩ. Since USB assumes a leakage resistance of greater than 300kΩ, this is well within the specification limits. The ADC accesses the resistive network at the divide-by-two point. Hence, there is a momentary doubling of leakage current on the USBDN_DP/DM lines when the 3.3pF (assume 5pF) sampling capacitor of the ADC touches the sampling point, but the USB leakage resistance specification is still met.

In order to perform this sampling with acceptable (>10bit) accuracy, the ADC will need to be operated with an extended sampling time option greater than 17.5µs via the HKADC table (either 32µs, 64µs, or possibly 16µs will be sufficient).

7.2.4.6.4 Electrical Characteristics

Table 7-13 Target Performance FCET – ADC Access

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	Per Port.			0	µA
Leakage Resistance	Per Port.		2		MΩ
Resistive Divider Offset Accuracy (Deviation from Nominal Value)	With 1.2V input on USBDN_DP/DM pin.			+/-1	mV

7.2.5 Low Power Current Detect Wake-Up (LPCDWU)

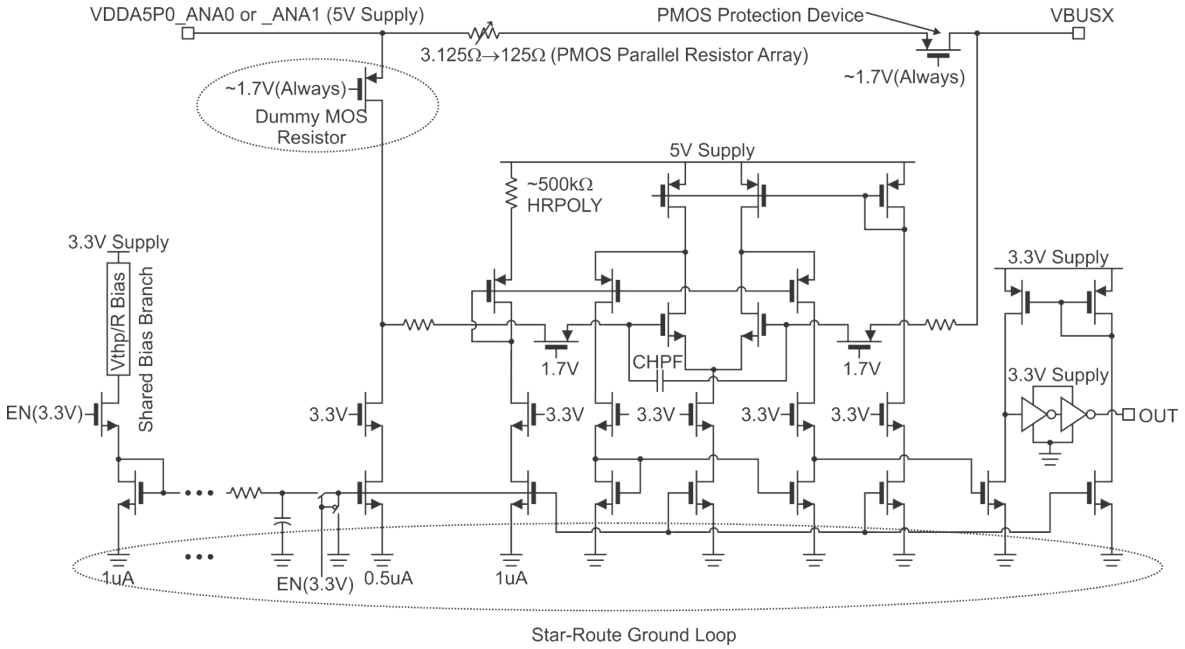


Figure 7-24 Boston Low Power Current Detect Wake-Up Circuit Simplified Schematic

7.2.5.1 Description and Interface

In an automotive application, the USB/Charging module must be in a low-current state < 100uA at the product level when the car is off, USB communications are disabled, and no external devices are drawing charging current from the USB ports.

However, when a device (or devices) is (are) plugged in, then anywhere from 1-20 Amp-hours of charge may be drawn from the USB/Charging module. This set of requirements dictates a need for a low-power (< 25uA spread across 4 ports) wake-up circuit that determines when a device is plugged into the USB port by sensing the small current draw taken to power up the device power protocol handshaking circuitry. That is, when a device is plugged into a USB port, it will first draw a small amount of current (<100mA under the USB BCS dead battery provision and the Connect Timing ECN) to commence negotiation operations with the USB port.

The goal of the Low Power Current Detect Wake-Up (LPCDWU) feature is to detect this current and to wake up Boston when said current is detected.

Note that this feature cannot count on detecting the pull-up resistor of the upstream port of the device.

The reason is that one cannot assume that what is trying to charge from the USB port is a USB-compliant device but may rather be some device that expects to find a wall-wart like termination. (such devices typically present networks of voltage sources and current sources and/or sinks to the host downstream port instead of a pull-up resistor).

This circuit interacts with the charging system after the main power port control switch has been opened by interposing an on-chip resistive element between the VDDA5P0 pin and one of the downstream VBUS pins. This is done by enabling the LPCDWU feature immediately before placing Boston into deep sleep mode. As mentioned earlier in this specification document, this modified sleep mode is termed LPCDWU Sleep Mode. The voltage developed across this resistive element is measured and compared to a voltage threshold. When this voltage threshold is exceeded, the comparator trips and toggles the state of its output line from low to high, activating the PMU sequencer and waking up Boston.

7.2.5.2 Coarse Requirements / Features

- Have a programmable current wake-up threshold of 2, 5, 10, 20, 40, 80mA ± 25%.
    - Thresholds of 0.5mA and 1mA are also included but not considered part of specification.
- When the current wake-up threshold is reached, have an internal voltage drop from the 5V supply to VBUS of no more than 0.3V.
- This implies the use of resistors of maximum values of 3.125, 6.25, 12.5, 25, 62.5, and 125Ω and allowing for up to 1Ω of internal metal routing resistance.
    - Resistors of maximum values 250Ω and 500Ω are also included but not considered part of specification.
- Keep the entire current draw of Boston (all 4 downstream ports) to under 50uA while in the LPCDWU mode, with a target of 25uA.
- In this LPCDWU mode, the only things active in Boston are the 4 LPCDWU circuits and the LPOSC.
- Withstand a short of VBUS to ground indefinitely while active and while disabled.
- Withstand conditions indefinitely whereby current past the trip limit is pulled through the MOS resistor such that VBUS is at 4V, 3V, 2V, 1V, 0V, etc.
- Withstand the condition indefinitely whereby the 5V supply is on but the 3.3V supply is not.

7.2.5.3 Architecture

7.2.5.3.1 Bandgap-less Architecture

Since the bandgap reference and associated V2I converters are disabled in Mode 4, the LPCDWU circuit must be designed such that its detection threshold is not substantially variant with supply voltage and process parameters. Another key objective is to minimize the size of the sense resistor array.

If POLY resistors are used for the sense resistor array, the 3.125Ω resistor (using RPPOLY to minimize the temperature coefficient) would be massive in size in order to avoid end resistance effects. Furthermore, the MOS switches required to bypass the other resistors would be similarly large, since their resistance value should be smaller than the resistors that they intend to bypass.

Therefore, it makes sense to use MOS resistors for the resistor array. However, MOS resistor values vary wildly with supply voltage and MOS oxide thickness, with the big problem being the oxide thickness (+/- 30%). (R is proportional to 1/(upCoxVdd)) (Here, Vdd is the 5V supply). Therefore, the design needs a comparator whose threshold tracks gate oxide (and perhaps mobility) in the same fashion.

An easy way to do this is to develop a reference threshold voltage composed of a current flowing through a MOS resistor with its drain terminal tied to the 5V supply and the gate terminal tied to the 1.7V PMOS protection voltage.

In the absence of the bandgap reference, bias current will be generated by a Vthp/R reference circuit which has good immunity from supply voltage variation and ripple.

7.2.5.3.2 5V-Tolerant Comparator

A 5V tolerant comparator shown in Figure 7-24 is proposed. A divide-by-two on the input signals is not proposed here for reasons of input-referred offset.

Some notes with regards to the comparator:

- The PMOS protection resistors on the NMOS input devices. These devices should prevent the gates of the NMOS devices from dropping below 1.7V even if VBUS is at 0V.
- The HPF capacitor “CHPF” improves the voltage ripple immunity of the comparator by isolating the terminals of the comparator from the 10uF VBUS decoupling capacitor. Without this capacitor and associated series resistor network, the low-pass filter created by the current detection resistor and the 10uF VBUS decoupling capacitor will create a mismatch in supply voltage ripple seen by the two comparator input terminals, potentially causing a spurious trip of the comparator.
- The comparator must be enabled for some time (~1ms) before its output becomes valid. Therefore, it must be enabled about 1ms before placing the ASIC into LPCDWU sleep mode in order to avoid the ASIC waking back up immediately.

7.2.5.3.3 PMOS Resistor Array

This design requires at least two groups of MOS devices in parallel to protect the gate oxide when the 5V supply is active and the VBUS is pulled to ground.

The first PMOS device "group" (shown as a PMOS device at the upper right of Figure 7-24) will connect VBUS to an internal net and will have a fixed gate voltage of 1.7V (derived from the 5V supply MOS diode divider). The maximum resistance of this device over corner should be 1.5625Ω. This device acts as a protection device, preventing 5V from appearing across the second device group if VBUS is shorted to 0V.

The second PMOS device group (shown as a variable resistor at the upper left of Figure 7-24) will have a programmable number of devices in parallel (thus actuating the series resistance) and will connect the 5V supply to the aforementioned internal net. The maximum resistance of this device group over corner should be 1.5625Ω at its minimum setting.

7.2.5.3.4 Turning Elements of PMOS Resistor Array On and Off

An important requirement for Boston while it is powering up is that it keep a very-high-impedance connection between the 5V supply and VBUS. Therefore, the LPCDWU PMOS resistor device must remain in a high-impedance state until the firmware has gained control of the system. Another important requirement of driving the elements of the PMOS resistor array is to level-shift 3.3V logic levels to 1.7V - 5V logic levels.

7.2.5.3.5 Current-Limiting Feature

Depending on the setting of the LPCDWU, it may pass over 1A of current per instantiation if it is enabled and VBUS is shorted to ground. This will cause an unacceptable local junction temperature rise and unacceptable electromigration stress on the local metallization.

In order to solve this problem, digital logic is added to the design to set the LPCDWU programmable resistance to the larger of the register-programmed resistance and the resistance which is at most 250Ω once the LPCDWU comparator has tripped. This limits the current sunk from the 5V supply to the grounded VBUS terminal to between 2.5mA and 5.5mA over corner and temperature.

The implication of this fact is that if VBUS has been discharged and it is desired to charge VBUS to the voltage level of the 5V supply before proceeding through a firmware state machine, then the LPCDWU must be enabled for a significant period of time before VBUS will be completely charged. Simulation shows that the worst-case time required to charge VBUS from 0V to 5V with 10uF of supply decoupling capacitance from VBUS to ground is about 20ms.

7.2.5.3.6 Calibration

In order to guarantee the specified accuracy, a 1-point calibration is required to be done on the part during production testing.

7.2.5.4 Electrical Characteristics

Table 7-14 Target Performance Low Power Current Detect Wake Up Circuits **Typ:** Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{5.0V}$=5.0V, Temp=27C **Min/Max:** Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{5.0V}$=4.5V-5.5V, Temp=-40C, +125C					
Name	**Conditions/notes**	**Min**	**Typ**	**Max**	**Unit**
Quiescent Current	Per 4 Ports Instantiation		20		µA
Leakage Current	Per 4 Ports Instantiation			1.3	µA
Switching Time		0.2		10	ms
(Vdd5V-VBUS) Voltage Drop at Trip Point			150	300	mV
Trip Current	All Conditions and Device Mismatch (Monte Carlo).	0.8* Inom	Inom	1.25 * Inom	V
PSRR	Up to 1MHz	20			dB

7.2.6 ADC (Used for Housekeeping ADC and Charge Measurement ADC)

7.2.6.1 Description/Interface

The ADC used in Boston is a 12-bit resolution (i.e. 4096 steps), 10-bit linearity converter running on the ADCCLK domain at a clock rate of 1MHz. Although the clock rate of the ADC is 1MHz, its maximum achievable sampling rate is 50kHz due to the successive approximation architecture utilized. However, the ADC does have the advantage that it can be used to sample voltages around the ASIC on a time scale faster than those interesting to the USB Battery Charging Specification [2] (which are at minimum 10ms).

Neither ADC in Boston is accessible directly from the microcontroller. This is due to the fact that both ADCs must be run continually while the microcontroller may be occupied with overseeing high-bandwidth USB-to-SD Card transfers. The Charge Measurement ADC is managed by the Charge Measurement Digital Back End, which provides its own interface to the firmware, while the Housekeeping ADC also provides separate interface to the firmware. Each of these interfaces is described in the Digital portion of this specification document.

The ADC operates in one of two modes:

- Differential, 2*VBandgap(2.426V)-referenced (Used for Charge Measurement ADC)
    - Differential voltage range is from -1.213V<Vinp-Vinn<1.213V.

  - Output is in signed 2's complement format.
  - Is always enabled so long as Charge Measurement system is enabled.
- Single-Ended, 3*VBandgap(3.639V)-referenced (Used for Housekeeping ADC)
  - Single-ended voltage range is from 0V<Vinp<3.639V.
  - Output is in unsigned integer format.
  - ADC is only enabled when a conversion is being made (AUTOEN mode).
    - Typical/Maximum charge of 1.04nC/1.39nC is consumed per conversion.
    - For clarity – AUTOEN mode is a metal-mask programmable feature of the ADC under which the ADC only consumes quiescent current when a conversion strobe is asserted to the ADC by another digital subsystem.

7.2.6.2 Coarse Requirements

- 12 Bit Resolution
- 10 Bit Linearity (10 Bit Accuracy after offset and gain error are measured and compensated for).
- Sub-100uA Quiescent Current
- >40dB PSR in 100kHz-1MHz Range

7.2.6.3 Architecture

The Boston ADC uses a successive-approximation method charge redistribution architecture while running from the 1MHz ADCCLK domain. The input sampling capacitor of the ADC is less than 5pF and therefore may form a noticeable time constant while sampling high-impedance nodes such as the FCET ADC Access point. In cases such as this, extended sampling times are available for choosing via the Housekeeping ADC interface.

7.2.6.3.1 Sampling Time

The number of ADCCLK (1MHz) clock periods required to process one ADC sample is as follows:

- Input Sampling Period: 1-64 (Programmable via register setting)
- Bias Settling (in AUTOEN mode): 8 (Only for HKADC)
- Data Conversion: 13
- Total: 14-85

7.2.6.4 Electrical Characteristics

Table 7-15 Target Performance ADC

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, $V_{1.5V}$=1.5V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V-3.6V, $V_{1.5V}$=1.35V-1.65V, Temp=-40C, +125C

Name	Conditions/notes	min	typ	max	unit
Quiescent Current	During Continuous Operation, CMADC		50	100	µA
Charge Per Conversion	Each Conversion, HKADC		1.04	1.39	nC
Sleep Current	When Powered Down	0.4	0.4	2.7	nA
Power Supply Rejection	Up to 1MHz	40			dB
Power Supply Rejection	Up to 100MHz	25			dB
Input Capacitance				5	pF
Required Clock		0.8	1	1.2	MHz
Resolution			12		Bits
Linearity		10			Bits
Gain Error	Uncalibrated,3σ			0.9	%
Offset Error, Differential Mode	Uncalibrated,3σ			±5.5	mV
Offset Error, Single-Ended Mode	Uncalibrated,3σ			±16.5	mV

7.3 Digital Functions

The Boston digital circuitry will be implemented in 0.13um 1.5V core devices and draws supply power from the 3.3V→1.5V regulator. Although the 1.5V regulator voltage accuracy is specified at 3.3%, digital synthesis and timing analysis should assume a deviation of +/- 10%.

7.3.1 USB Hub Digital IP Core

Boston incorporates a USB 2.0-compliant (with the exception of suspend current) Hub supporting the following features:

- Supports up to 4 exposed downstream ports and 1 internal downstream port.
- The internal downstream port internally attaches to a SD Card reader device as a USB Compound device.
- Can operate in hub modules with 0-4 exposed downstream ports and 0-1 SD Card readers.
- Supports High-Speed, Full-Speed, and Low-Speed modes of operation.
- Supports an individual transaction translator (TT) on each downstream port to maximize data throughput when multiple Full-Speed devices are placed on downstream ports.
- Supports USB port power control in an individual or ganged configuration, including enabling of external port power control switches and the receipt of overcurrent/overtemperature fault event signaling.
    - Note: Ganged USB PPC not available as a native feature in Boston A1 USB Hub Digital IP, but may be implemented with firmware.
- Supports configuring the delay time for filtering the over-current sense inputs.
- Supports configuring the delay time for turning on downstream port power control switches.
- Supports enabling an arbitrary number of downstream ports.
    - Note: This feature is not available in Boston A1, may be implemented in future revisions.
- By default, the SD Card Device is hard-wired to Hub Port 1 and the four downstream exposed ports 0-3 are hard-wired to Hub Ports 2-5, respectively.
- Features a programmable configuration ROM to customize the device descriptors sent back to the USB Host during enumeration.

7.3.2 USB Device Digital IP Core

Boston incorporates a Compound USB Device supporting the following features:

- Endpoint 0
    - Standard Control Endpoint.
- Endpoint 1 IN/OUT each with 2KB total buffering RAM.
    - Supports a Mass Storage Class Bulk Only Transfer interface.
    - This interface is intended for use with an SD Card Host.
- Endpoint 2 IN/OUT each with 1KB total buffering RAM.

- Supports an HID Class interface.
- This interface is intended for use with a side channel between the host and the integrated microcontroller.

7.3.3 SD Card Host Digital IP Core

Boston incorporates an SD Card Host which supports the following features:

- Read/write access to SD Memory Cards compatible with SD Host Controller Specification 3.00.
- All SD Memory Capacity standards including SD, SDHC, SDXC.
- The following interface modes: Default Speed, High Speed, UHS-I speeds (SDR12, SDR25, SDR50, DDR50).
- 35MB/s data throughput for card latency < 100µs.

7.3.4 USB Audio Input (I2S)

This function provides support for USB audio-in streaming using an external I2S-compliant ADC input for the audio source. This function requires minimal microcontroller intervention and, except for exceptional events, operates in a set and forget fashion. The burden of packetizing the audio data and managing the isochronous stream is implemented in hardware.

Note that in the following description, the term ***frame*** is used in two senses: the ***USB frame*** (as defined in the USB Spec) and the ***audio frame*** (as defined in the USB Audio Formats Spec). The term ***USB frame*** refers to approximately 1ms time discretization of the USB protocol delimited by start-of-frame (SOF) tokens. The term audio frame refers a single time-sample of a PCM audio stream, which for stereo contains one left-channel and one right-channel sample. Similarly, the term ***audio subframe*** refers to a single left or right sample.

It is the firmware's responsibility to provide configuration information to the hardware consistent with the device descriptors it provides to the USB host in response to default pipe traffic on the control endpoint. The audio packetizer provides support for the following USB audio configuration(s):

- PCM sample format: wFormatTag = 0x0001
- Two audio channels: bNrChannels = 2
- 48kS/s ±1% audio rate: bSamFreqType = 0
  bLowerSamFreq = 47524
  bUpperSamFreq = 48480
- Support ±1% frame rate
- 16 or 24 bit sample framing: bSubframeSize = {2,3}
- 16 or 24 bit reported resolution: bBitResolution = {16,24})
- 1ms packet rate: bInterval = 4 (HS mode)
  bInterval = 1 (FS mode)

- Sychronization modes:
  - Synchronous: bmAttributes = 0x0D
  - Asynchronous: bmAttributes = 0x05

The audio subframe size and reported bit resolution must be the same. There is no requirement that the ADC provide all of the significant bits for the USB-reported resolution. The MCU can control how many of the ADC bits are significant and the famer will insert the required number of 0s in the lsb positions to make up the desired USB subframe size.

Synchronous mode transfers act in a pseudo-synchronous mode where the difference between input and output sample rate is rectified by insertion or deletion of single audio frames. The audio packetizer tracks the USB frame rate and compares it to the expected one (1ms). If the USB frame is faster than expected, then single audio frames will be inserted as required, where the inserted audio frames are simple copies of the previous audio frame's samples.

Asynchronous mode transfers generate packets with the actual number of audio frames which had occurred between the detected SOF tokens bounding the USB frame wherein those samples were aqcuired.

7.3.4.1 Architecture

This feature is implemented using:

- a programmable ADC clock source derived from the 480MHz USB PLL
- an I2S receiver synchronized to the ADC clock rate
- a USB audio framer synchronized to the MCU bus and USB device controller, providing frame data to the ISO in endpoint and exchanging configuration and exception data with the MCU.

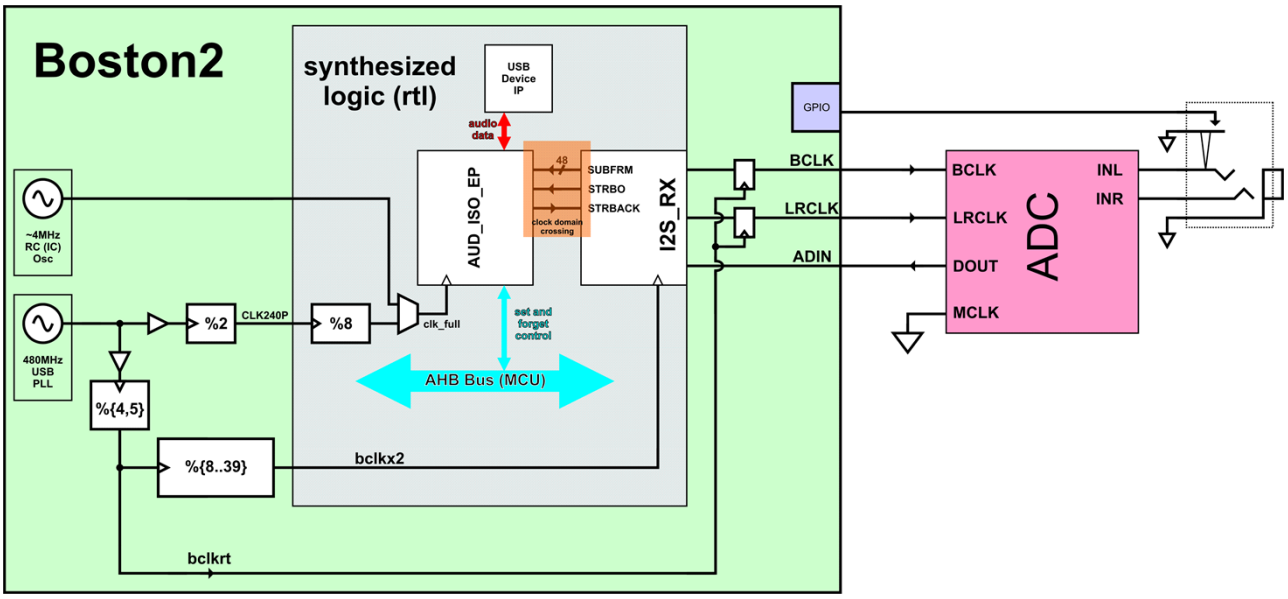


Figure 7-25 USB Audio Input System Architecture

7.3.4.2 Frequency Generation

The primary ADC frequency source is the 480MHz PLL. The 48kHz sampling rate implies a LRCLK rate of:

$$fLRCLK = fclk480 / 10000 = fclk480 / (5^4 * 2^4)$$

Since there is only four factors of two in the division from clk480 to the sampling rate, it is not possible to create the typical ADC oversampled clock rates (e.g. 256X LRCLK). Therefore the external ADC must be selected with the ability to generate its own desired oversampling rate clock using the supplied BCLK or LRCLK as a reference clock.

The frequency generation circuitry is a cascade of programmable integer-N dividers. A programmable {4,5} prescalar generates bclkrt which is then further divided by a programmable divider to produce bclkx2. This clock is provided to the I2S receiver which divides it by a fixed factor of 2 to generate BCLK. The I2S receiver will then further divide BCLK to produce LRCLK. Since either the BCLK or the LRCLK could be used as a source clock for the ADC's timebase, the BCLK and LRCLK signals are retimed by the bclkrt clock to remove data-dependent jitter before being provided to the external ADC.

The programming of BCLK frequency is accomplished by BCKMD[3:0] and BCKPD registers. The BCLK frequency can be calculated as:

fBCLK = 480MHz / (2*(4+BCKPD)*(8+BCKMD))

Keep in mind that care must be taken to ensure that BCLK is an even multiple of LRCLK in order for the data connection to the I2S interface to be periodic at the audio sampling rate. Appropriate frequency plans will be presented in the next secion.

7.3.4.3 I2S Interface

The data connection to the external ADC is implemented using the I2S protocol with the external ADC operated in I2S slave mode. Signals BCLK (data bit clock) and LRCLK (left/right sample clock) are provided to the ADC and the ADC is expected to return audio data in compliance with the I2S specification. No MCLK (master clock) is supplied to the ADC as it is expected that the ADC will be programmed to generate its own master clock using either BCLK or LRCLK as a reference clock.

The ADC is expected to respond appropriately to additional BCLKs beyond what is strictly necessary to transmit the audio data words. The I2S receiver should be programmed with the number of significant bits expected from the ADC using the NBADC[4:0] register. All serial bits provided by the ADC beyond the programmed number will be treated as zero by the receiver. For correct USB data framing, the programmed frequency plan should ensure that at least 24 BCLKs are provided for each ½ period of LRCLK even if the number of significant data bits provided by the ADC is less than this.

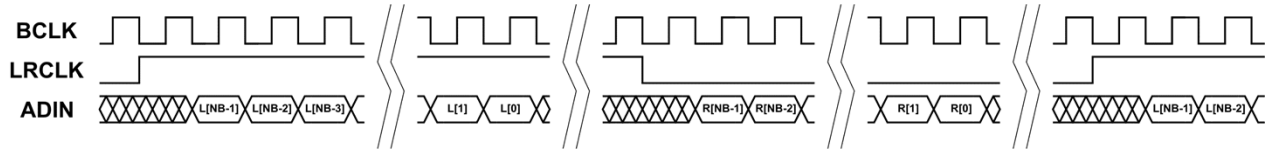


Figure 7-26 I2S Protocol

LRCLK is generated by programmable division of BCLK, with the division rate programmable via the NBSAMP[5:0] register. The LRCLK rate can be calculated as:

$$fLRCLK = fBCLK / (2*NBSAMP)$$

Example frequency plans can be found in the following table:

Table 7-16 Example I2S Frequency Plans

BCKPD	BCKMD	NBSAMP	fBCLK	fLRCLK	Comment
1	12	25	2.4MHz	48kHz	
1	2	50	4.8MHz	48kHz	recommended mode

The I2S receiver will provide 24 bit audio subframes (stereo samples) independently of NBADC or NBSAMP settings. The MSB of each sample is the ADC sample's MSB and the (24-NBADC) LSBs will be zeros.

The firmware must enable the I2S interface for it to operate by writing a 1 to the I2SENA register. When not enabled, the BCLK and LRCLK will be held at 0 and no data will be provided to the USB packetizer.

The firmware may set the I2SMUTE register bit to ignore ADC data and deliver zero-value samples to the USB framer. Changes in this bit only take effect on audio frame boundaries.

7.3.4.4 USB Packetizing

The USP packetizer receives the continuous audio samples from the I2S receiver and assembles them into USB Isochronous payload packets for transmission to the host.

7.3.4.4.1 Payload Data and Data Rate Synchronization

The USB packetizer only supports polling rate of 1ms. For the host to poll data at this interval, the value of bInterval in the endpoint descriptor must be adjusted according to whether the device is connected in FS or HS mode. For use in FS mode, bInterval must be set to 1 to request polling each 1ms frame. For HS mode, bInterval must be set to 4 to request polling every 8th 125us microframe. The firmware must inform the packetizer whether it is to operate in HS or FS mode by appropriately setting the ISOHS register bit.

The packetizer supports two USB Isochronous synchronization methods, which can be selected by the firmware using the ISOSYNC register bit. This bit should be provided by the firmware with value consistent with the bmAttributes reported in the endpoint descriptor.

When ISOSYNC is set to 0 (corresponding to bmAttributes = 0x05) then the packetizer makes no attempt to match the incoming data rate to the outgoing packet rate. The packetizer will divide the audio data into time-slices synced to the received USB start-of-frame (SOF) tokens. In the case of FS mode, each SOF will denote the boundary of a time slice. In the case of HS mode, every SOF corresponding to the first micro-frame in a USB frame will denote the boundary of a time slice. As explained in the later error-handling section, a single missing SOF token will be interpolated if it is expected based on frame timing but is not received. This interpolated SOF will also be included in the calculation of this time-slicing. The audio packets will then enter a queue for transmission in response to ISO IN data requests from the host. The packet FIFO depth will <either 2 or 3 TBD depending on implementation details> so that the packet being transmitted during the Nth time slice had been sampled during <the $(N-1)^{th}$ or $(N-2)^{th}$ time slice respectively>.

In FS mode, one queued data packet will be armed for transmission each frame. In HS mode one queued packet will be armed for transmission on the beginning of the first micro-frame of each USB frame. After each ISO IN transaction, the endpoint will disarm and respond with zero-length packets to any subsequent ISO IN transactions until armed again.

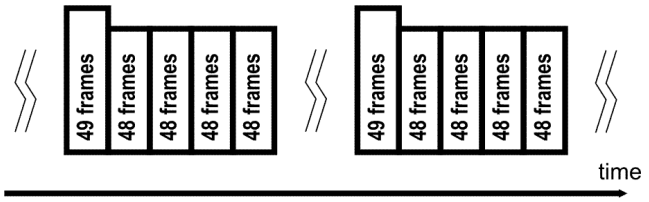


Figure 7-27 Example asynchronous mode packet sequence, slow host

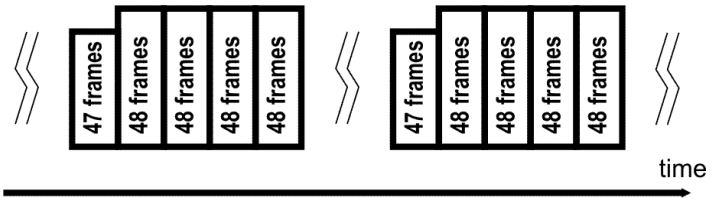


Figure 7-28 Example asynchronous mode packet sequence, fast host

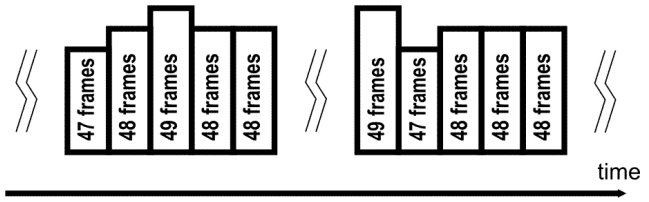


Figure 7-29 Example asynchronous mode packet sequence, nearly matched host

When ISOSYNC is set to 1 (corresponding to bmAttributes = 0x0D) then the packetizer will provide exactly 48 audio frames in each ISO packet. Since the incoming data rate does not exactly match this outgoing data rate, samples will be inserted and/or deleted in order to match the rate. The audio stream is divided into time-slices exactly as in the asynchronous mode and the sample insertion/deletion is performed on a per-packet basis. This results in a balanced average synchronization, so the inserted / deleted audio frame packets will be infrequent and non-consecutive from other inserted / deleted packets respectively. In the case that the incoming and outgoing audio rates nearly match, it is possible that there will be both inserted and deleted frame packets in approximately equal number as the audio sampling clock beats with the nominally 1ms frame rate.

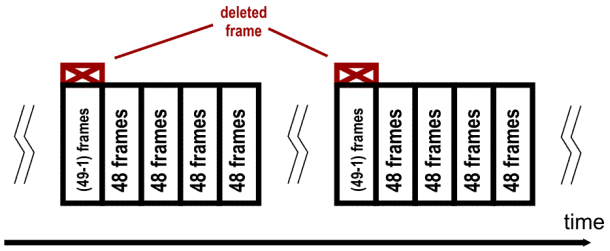


Figure 7-30 Example synchronous mode packet sequence, slow host

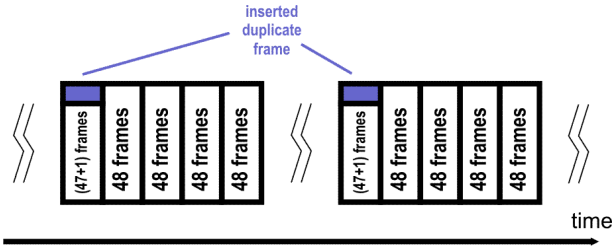


Figure 7-31 Example synchronous mode packet sequence, fast host

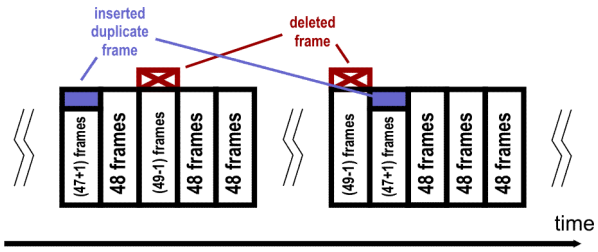


Figure 7-32 Example synchronous mode packet sequence, nearly matched host

7.3.4.4.2 Packet Format

The USB packet format is compliant with the PCM format (wFormatTag = 0x0001) standard as described in the *Universal Serial Bus Device Class Definition for Audio Data Formats Release 1.0* and *Universal Serial Bus Device Class Definition for Audio Devices Release 1.0* standards. Audio data is transferred so that each packet contains an integer number of audio frames transmitted in time-increasing order. Each audio frame is comprised of two byte-aligned audio subframes consisting of left and right PCM samples, with left sample transmitted first. Each subframe is transmitted LSB first, and the final transmitted MSB is the ½ full scale weighted bit (the 2^{s} complement sign bit).

The firmware can select 16 bit or 24 bit audio subframe size by setting the SFRMSZ register bit to 0 or 1 respectively. When using 16 bit subframes, the most significant 16 bits of each subframe from the I2S module are transmitted.

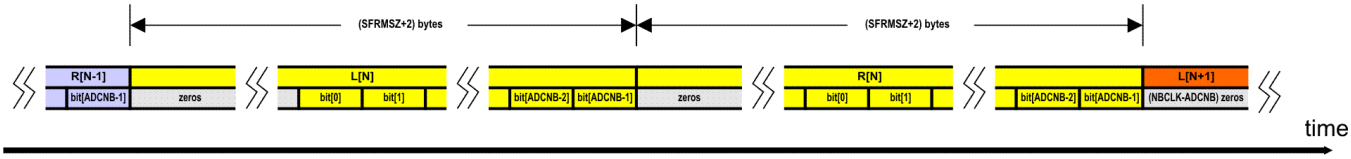


Figure 7-33 Audio frame bit ordering (ignoring NRZI coding and bit stuffing)

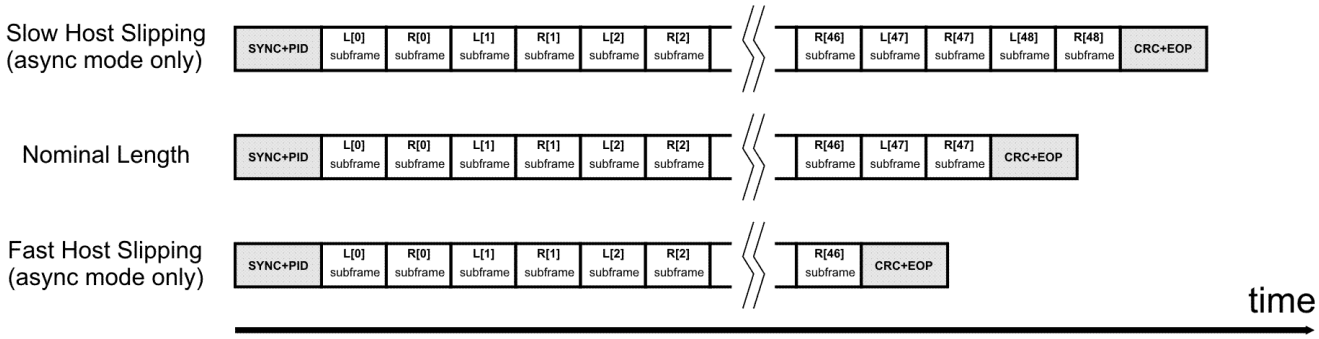


Figure 7-34 Data packet composition and ordering

7.3.4.4.3 Frame Tracking / Synchronization / USB Protocol Error Handling

The USB audio packetizer supports automatic handling of isolated missing/mangled SOF tokens. If a SOF token is expected and is not received, the packetizer will continue its packet schedule as if the SOF had been received at the expected time. If two consecutive frames of missed SOF occur, or a single detected instance of a badly timed SOF token is detected then an interrupt will be generated to alert the firmware.

The packetizer supports automatic data clearing in the case of missing ISO IN tokens. This could occur because the host does not request the data, or because of a bit error in the reception of an ISO IN token. At the end of any USB frame (the end of the 8th micro-frame of each frame in HS mode), if the host has not requested data during that USB frame then the packet at the head of the queue is flushed to reposition the queue to the same state as if the packet had been transmitted.

Initial queue synchronization and recovery from any subsequent loss of synchronization is accomplished by not arming the endpoint for transmission if the number of packets in the queue is less than expected. In this event an interrupt will be generated to alert the firmware.

The firmware can assert the I2SZLP register bit to clear out the packet FIFO and respond to any ISO IN transactions with zero-length packets. Upon de-assert, ISO IN transactions will continue to return zero-length packets until the packet queue reaches its natural depth whereupon normal operation commences again. Note that I2SZLP will be automatically set whenever I2SENA is not set.

7.3.5 Host to Host Bridge

The host-to-host (H2H) bridge function provides two additional USB device controllers and data bridging functions to allow an external USB device attached to one of the “downstream” ports to transfer data to/from the upstream USB host while the external device is itself acting as a USB host. To implement this function the firmware should set the hub’s DS4 data path to be logically connected to the first USB device controller (UDC_DS) instead of the off-chip USB connector. It is expected that this data-path change will occur during startup and will not be dynamically adjusted during operation. The second USB device controller (UDC_US) can be logically connected to any of the three remaining

"downstream" ports so that any device which connects to this port can act as a USB host for that device controller.

The UDC_US device control endpoints, IN0 and OUT0, will be managed by the firmware in the usual fashion. It is the expectation that the system and firmware will cause this device to connect to the upstream host and use these control endpoints to exchange information with the upstream host to coordinate H2H datapath switching operations for the user's external USB device.

The UDC_DS device control endpoints, IN0 and OUT0, will be managed by the firmware in the usual fashion. The device is **cannot** be connected to the hub, but at firmware discretion can be attached to any one of three external USB ports PRT1, PRT2 or PRT3 after detaching that port from the hub. Whenever the UDC_DS device is logically connected to one of the external USB ports, the firmware can interact with the device controller and control endpoints, IN0 and OUT0, in the usual manner. The expectation is that the firmware will make the UDC_DS device controller attempt to connect to the USB network (as a USB device) under the expectation that the end-user's mobile device is acting as a host on that port.

The remaining endpoints on UDC_DS and UDC_US consist of matched pairs, wherein each endpoint in UDC_DS is paired with an equivalent opposite direction endpoint on UDC_US. Each of these endpoint pairs are connected through a packet transfer bridge which can be configured to automatically transfer data between the endpoints in a manner appropriate for that particular endpoint / data content type. The firmware is expected to manage the initial setup of these bridges and intervene in exceptional cases, but otherwise is not expected to assist in data transfers.

For test purpose, the firmware can directly interact with the data endpoints' packet data buffers in the same manner as the application USB device controller (the full device controller function is available to the firmware). Such direct interaction with an endpoint should not be done while that endpoint is being automated by a data pipe or else data corruption may result.

7.3.5.1 Port Switching

Each of the external USB ports has logical datapath muxing to allow the firmware to adjust the port and associated hub downstream port datapaths. The DS PRT4 could be switched to UDC_US while one of DS PRT1/2/3 is switched to UDC_DS. All of them are accomplished at an internal digital interface inside the IC and does not touch the DP/DM/VBUS lines or change the functionality of the FCET/ CMADC subsystems. The firmware has ability to switch between a default datapath H2H related secondary datapath. When one of DS PRT1/2/3 is switched to UDC_DS, the hub behaves as if there is a physical port with no device attached.

0x50013022 : H2H_PORTCTRL			
Bit index	Name	Access mode	Description
7	reserved		

6	h2h_fuse	R/W -1'b0	MCU use this bit to select the DWN_PORT[4] as an external port or an internal H2H bridge hub-facing port 0 -> external port; 1 -> internal port
5	h2h_d_utm ivbusvalid	R/W -1'b0	The vbus status bit used by the H2HD PHY, set it to 1 when H2H is enabled
4:0	h2h_port_ switch	R/W - 5'b00000	Set one bit to select the corresponding port as a H2HD port 01000->Port3; 00100->Port2; 00010->Port1; The rest value of the register will be treated invalid and won't trigger the port switch

External USB port PRT4 has secondary H2H function of connecting the UDC_US to the DS4 hub downstream port. The associated default datapath which must be disconnected for this H2H function is the USB port PRT4 to hub downstream port DS4 datapath. The firmware controls this connectivity using the H2H_FUSE register setting. It is recommended that this setting be made during initial firmware boot up and not subsequently changed, since only one of the H2H bridge or external DS PRT4 will be implemented.

External USB ports PRT1, PRT2 and PRT3 have secondary function of connecting to the UDS_DS device controller. The associated default datapath which must be disconnected for this H2H function is that port's external USB port to hub downstream port datapath. Only one external USB port can be so connected at any given time. It is recommended that any external port should change this connectivity only when there is no device or host connected (in the USB sense) to it. Note that this does not require the device's cable to be removed, only that there is no USB network interacting with that port at the time of the switching. The firmware controls this datapath with the H2H_PORT_SWITCH register settings. Setting more than one of these bits to 1 is not supported. The H2H_D_UTMIVBUSVALID is set to allow the UDC_DS device to connect to the potential host which is attached to the selected port.

Although the port switching is handled by firmware, the anticipated port switching sequence is depicted in the following figure.

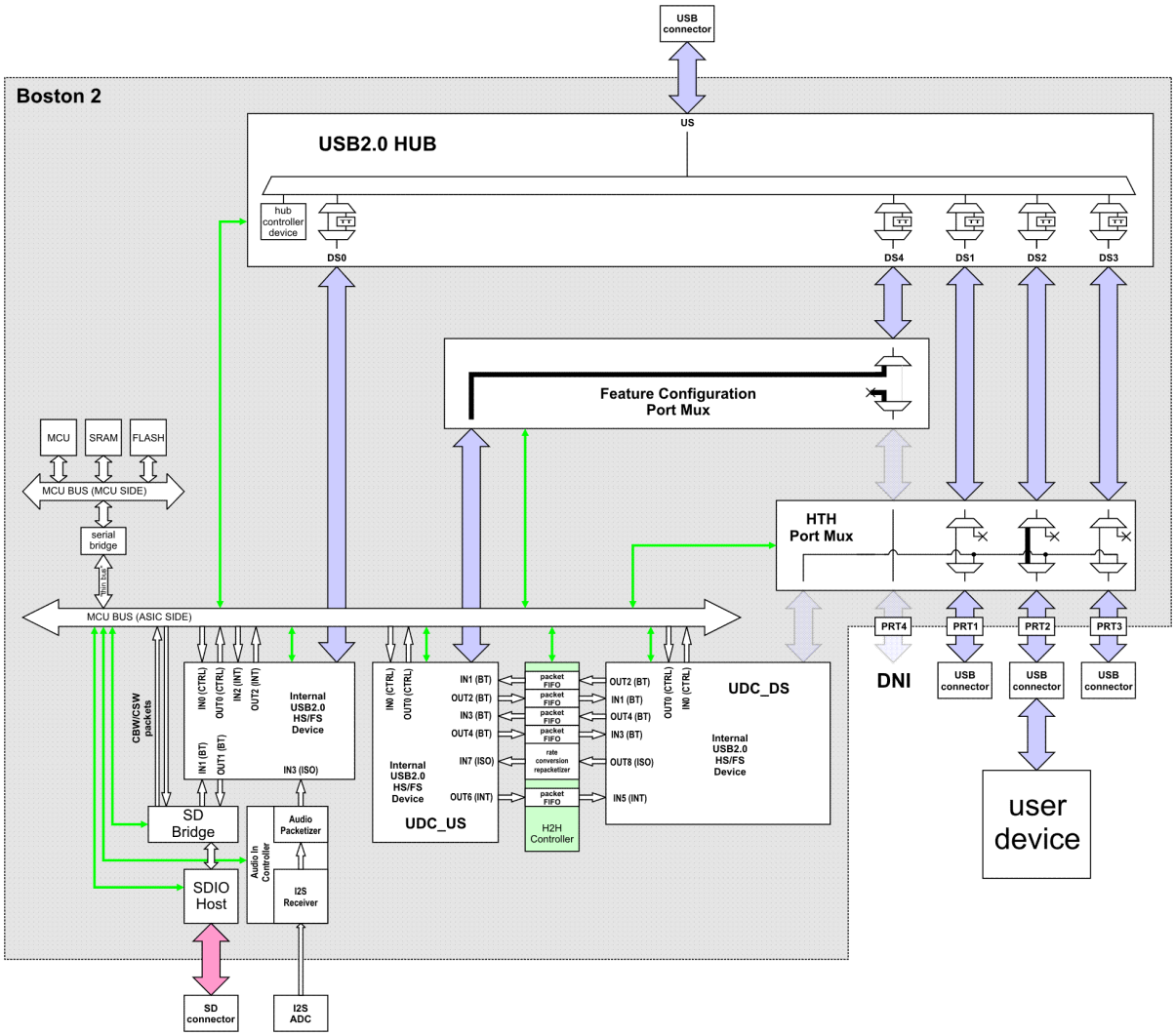


Figure 7-35(a) Port Switch Sequence

At boot up the firmware sets H2H_FUSE to 1'b1. It is expected that there will be no USB connector soldered to the PCB for PRT4 in products implementing this feature, so trivially there will be no USB device connected to the port at this time. The firmware will instruct UDC_US device to connect to the upstream host and assist in enumeration. The upstream host will examine the descriptors and load appropriate drivers for subsequent H2H operation. The end user will later plug in a device to one of the three remaining external ports (PRT2 in this example), which will connect to the upstream host through the hub in the usual manner, placing the system in the (a) configuration. The upstream host will examine the user device's descriptors and determine that this device is compatible with the H2H

function.

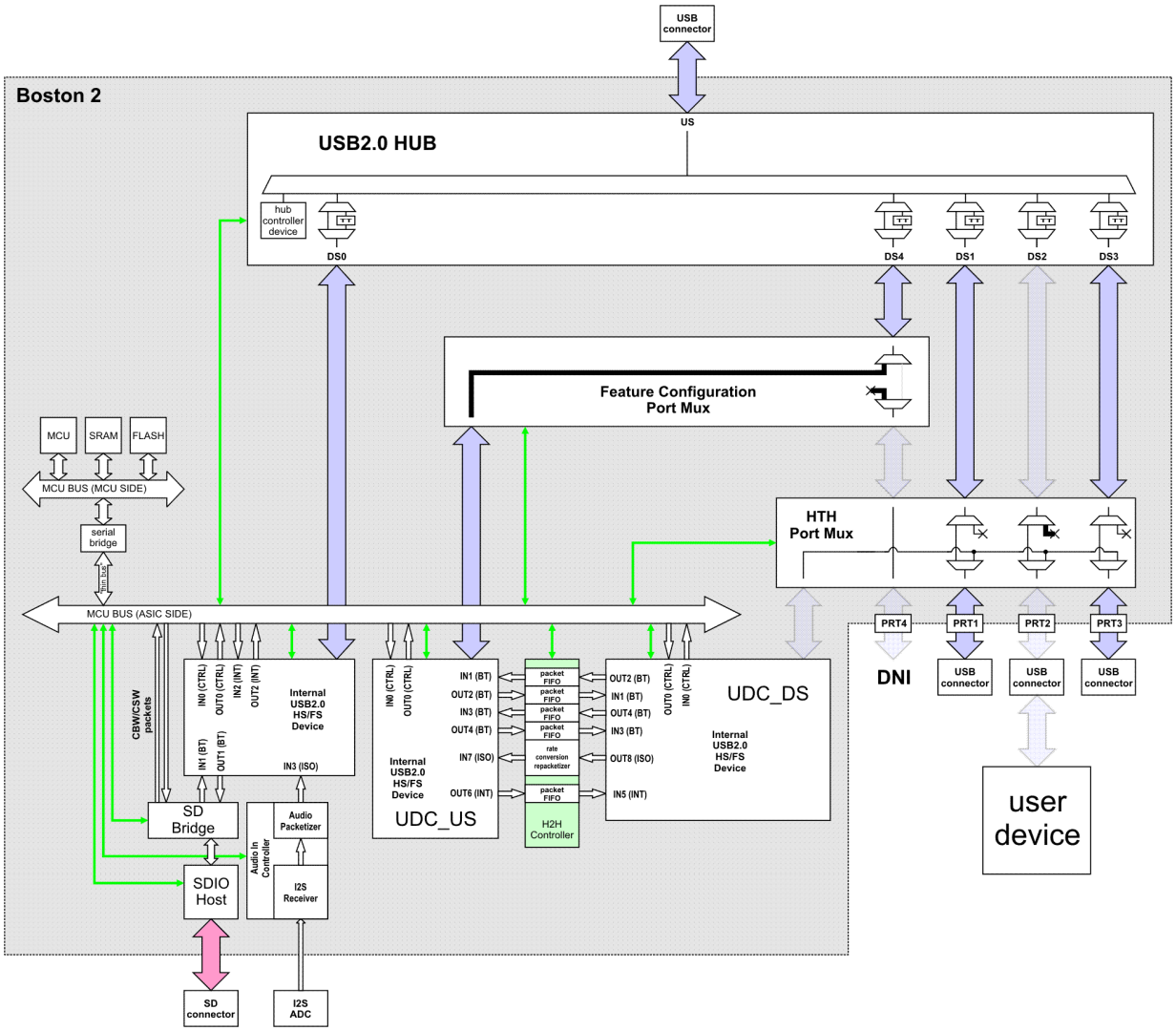


Figure 7-36(b) Port Switch Sequence

The upstream host and user's device may negotiate between themselves over the USB network to arrange for the H2H function to activate. The H2H device drivers running on the upstream host will then communicate with the Boston firmware using UDC_US control endpoints OUT0 and IN0 to arrange for the port change and inform the Boston firmware which hub port number is involved. Having previously negotiated the H2H function with the upstream host, the user's device will perform a USB disconnect. This places the system in the (b) configuration.

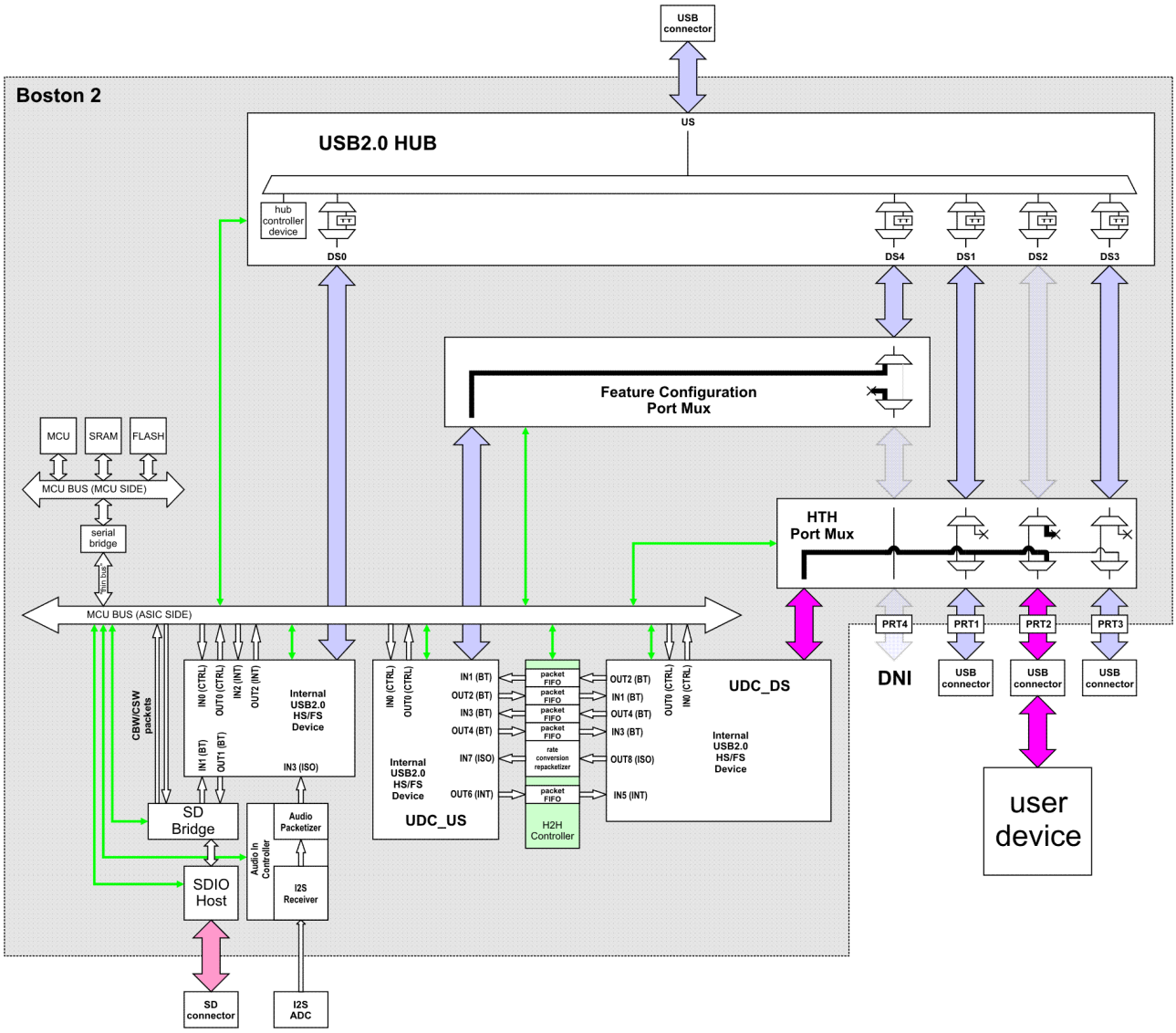


Figure 7-37(c) Port Switch Sequence

The Boston firmware will then set the appropriate value to H2H_PORT_SWITCH to route the user device logical connectivity to UDC_DS. Having waited an appropriate time, the user's device will reconnect itself to the USB signals as a host of a 2nd USB network. The Boston firmware will then set 1 to H2HD_UTMIVBUSVALID, enabling UDC_DS to connect to this new USB network and placing the system in the (c) configuration.

The reverse connection process can be performed in similar manners. If the upstream host wishes to deactivate the H2H bridge then it can negotiate with the Boston firmware to do so. If the user's device is physically disconnected, the Boston firmware should shut down the UDC_DS controller and either reconnect the external port to the hub or await further instruction from the upstream host depending on design requirements.

Changes to the datapath connectivity have no effect on any FCET terminations or Charge Measurement settings. Any firmware settings for these subsystems continue to apply to the external USB port independently of any logical connectivity change on the datapath associated with that port.

When an external port is switched from the hub to the UDC_DS device, the hub loses visibility of any over-current event on the VBUS line for that port and it becomes solely the firmware's responsibility to manage corrective action in the vent of over-current on that port.

7.3.5.2 Bridged Endpoints

Six pairs of bridged endpoints are implemented. Four bulk transfer pairs implement two sets of bidirectional bulk transfer packet data pipes between the hosts. One interrupt pair implements an interrupt packet pipe from upstream host to the user device. One isochronous pipe implements a PCM audio stream pipe.

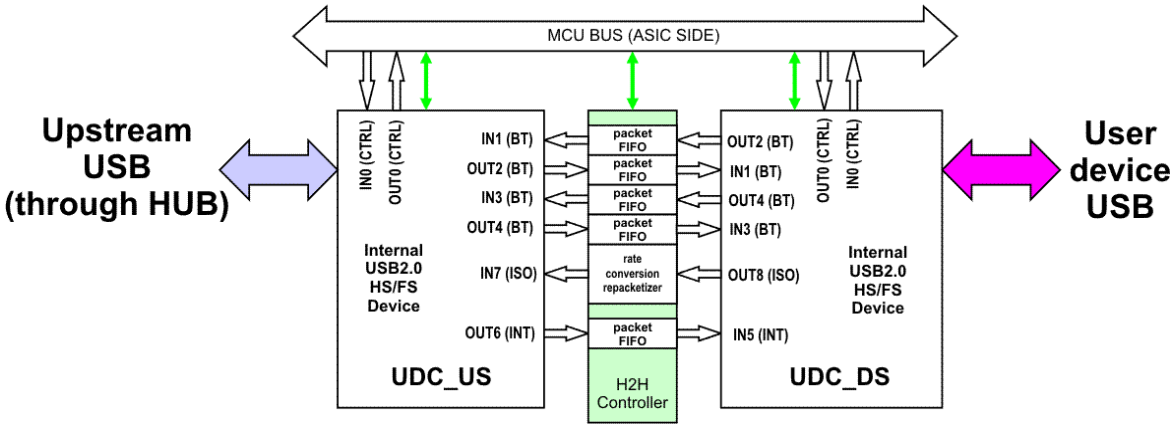


Figure 7-38 H2H Bridge Dataflow

7.3.5.2.1 Bulk Transfer

There are four bulk transfer endpoint pair pipes:

1. D2H1 pair: UDC_DS:OUT2 → UDC_US:IN1
2. H2D1 pair: UDC_US:OUT2 → UDC_DS:IN1
3. D4H3 pair: UDC_DS:OUT4 → UDC_US:IN3
4. H4D3 pair: UDC_US:OUT4 → UDC_DS:IN3

Each endpoint pair implements packet transfers between the independent packet FIFOs on each bridged endpoint. Whenever a pipe is enabled, that pipe transfers packets from the associated OUT endpoint to the associated IN endpoint whenever a packet exists in the OUT endpoint FIFO and space exists in the IN endpoint FIFO. Packets are transferred in a manner to preserve packet size and packet boundaries.

OUT endpoints implement normal USB2.0 ping protocol: The OUT endpoint NAKs any transferred packet if its FIFO is full. If the FIFO will become full then the endpoint acknowledges with NYET. PING is responded to appropriately according to whether the packet FIFO can accept a packet. The maximum FIFO depth available for each OUT endpoint FIFO is 4. The FIFO depth is configured in the same manner as the application function USB device controller.

IN endpoints implement normal USB2.0 protocol: IN tokens are responded to with data if available, otherwise the response is NAK. The maximum FIFO depth available for each IN endpoint FIFO is 4. The FIFO device controller FIFO depth should configured to be 4. The maximum packet queuing depth can be controlled by the BT_PKTD SFR setting. All bulk-transfer IN endpoints use the same BT_PKTD queuing depth.

Table 7-17 BT_PKTD Coding

BT_PKTD	IN FIFO Depth Used By Pipe
00	1
01	2
10	3
11	4

The firmware can enable/disable the receiver of the various pipes using the D2H1_ENA, D4H3_ENA, H2D1_ENA and H4D3_ENA special function registers bits. When disabled the endpoints can be accessed by the firmware in the usual manner. A pipe may be enabled any time the device controllers are both properly configured for endpoint traffic. A pipe may be disabled at any time, but be aware that the endpoint buffers should not be accessed for a short time thereafter in case a packet transfer was already in progress when the pipe was disabled (the packet will compete its transfer).

The maximum packet size for the bulk transfer endpoints is 512. The firmware is responsible for configuring the device endpoints and buffer memory map through the device controller.

For each pipe, the firmware can be alerted to congestion in the OUT FIFO by receiving an interrupt whenever an OUT packet is NAKd. The pipe which is full is indicated by a corresponding IRQ flag bit, D2H1_IRQF, H2D1_IRQF, D4H3_IRQF or H4D3_IRQF. This interrupt can be enabled/disabled for each pipe by setting the corresponding interrupt enable bit: D2H1_IRQENA, H2D1_IRQENA, D4H3_IRQENA or H4D3_IRQENA.

7.3.5.2.2 Interrupt

There is one interrupt transfer endpoint pair pipe:

H6D5 pair: UDC_US:OUT6 → UDC_DS:IN5

The interrupt pipe behaves in a similar manner as the bulk-transfer pipes, except that the maximum packet size is 64 bytes. The pipe enable signal is H6D5_ENA and the IN FIFO queuing depth is set using the INT_PKTD special function register. The interrupt for the OUT FIFO full is H6D5_IRQF, and the interrupt enable is H6D5_IRQENA.

7.3.5.2.3 Isochronous PCM Audio Stream

There is one isochronous PCM audio endpoint pair pipe:

D8H7 pair: UDC_DS:OUT8 → UDC_US:IN7

Note that in the following description, the term ***frame*** is used in two senses: the ***USB frame*** (as defined in the USB Spec) and the ***audio frame*** (as defined in the USB Audio Formats Spec). The term ***USB frame*** refers to approximately 1ms time discretization of the USB protocol delimited by start-of-frame (SOF) tokens. The term audio frame refers a single time-sample of a PCM audio stream, which for stereo contains one left-channel and one right-channel sample. Similarly, the term ***audio subframe*** refers to a single left or right sample.

This endpoint pair is bridged with its own audio frame (stereo audio sample) FIFO. The FIFO does not preserve packet size, but instead adjusts the outgoing packet sizes to perform balanced average rate synchronization. This rate synchronizer expects incoming packets with a nominal polling period of 1ms and exactly 48 audio frames per packet. It calculates output packet sizes for a nominal polling period of 1ms to match the average data rate and retain a constant (averaged over the traffic bursts) mean FIFO level. As such, the UDC_US:IN7 endpoint should be declared in the descriptors as a synchronous ISO OUT endpoint bInterval consistent with a 1ms service period. The UDC_DS:OUT8 should be declared in the descriptors as an asynchronous ISO IN endpoint with implicit feedback and bInterval consistent with a 1ms service period.

The data format, packet sequencing, frame tracking and error handling considerations for the transmit interface (UDC_US:IN7) are the same as for the I2S audio endpoint and the relevant descriptions should be considered incorporated here by reference.

The receive side of the bridge supports automatic handling of isolated missing/mangled SOF tokens. If a SOF token is expected and is not received, the receiver will continue its packet schedule tracking as if the SOF had been received at the expected time. If a packet is expected and not received an interrupt will be generated to alert the firmware and the AUDD_IRQF will be set.

The receiver supports automatic data insertion in the case of missing ISO IN tokens. This could occur because the host does not provide the data, or because of a bit error in the reception of an ISO OUT token. At the end of any USB frame (the end of the 8th micro-frame of each frame in HS mode), if the host has not provided data during that USB frame then the receiver will insert 48 zero value samples into the FIFO if the AUD_TXF bit is set. This is optional and not recommended. When the AUD_TXF bit is not set, the sample FIFO receives no additional samples and the rate tracking system will restore the buffering depth by dynamically adjusting audio frames per packet delivered to the IN endpoint. It is likely that this will cause less noticeable audio artifacts than inserting zero samples when packet error rates are negligible.

At the end of each polling interval, the pipe will automatically retire the packet at the head of the transmit queue as per the USB ISO specification.

If the data rate difference between the OUT and IN endpoints is sufficiently different that the sample buffer overflows or underflows, an interrupt will be generated and the AUDB_IRQF flag will be set. The current number of samples in the buffer can be read by the firmware from the AUDBUFC register.

The firmware can assert the AUD_CLR register hold a preload of zero-value samples into the FIFO. The firmware has independent control over the enables for the various components of the audio pipe in order to properly sequence the pipe. The AUD_RADENA bit enables the rate adapter tracking and should be the first to be enabled. It is recommended that the rate adapter be enabled whenever both USB device controllers are connected (even if the audio pipe is disabled) to allow continuous acquisition of the rate differential. The OUT side of the pipe can be enabled using the AUD_RXENA register bit. When the AUD_RXENA bit is set then packets are automatically transferred from the OUT endpoint FIFO into the sample buffer. The IN side of the audio pipe can be enabled by using the AUD_TXENA bit. When AUD_TXENA is set then the IN FIFO will be loaded with a new audio packet each 1ms (as measured by the SOF rate on the UDC_US network) period. If AUD_TXENA is not set but AUD_RXENA is set, then the audio data will simply be removed from the head of the sample buffer.

The recommended startup sequence is:

1 – assert AUD_CLR and wait for completion

2 – ensure / wait until both device controllers are connected.

2 – set AUD_RADENA

3 – set AUD_RXENA

4 – set AUD_TXENA

The audio pipe must be made aware of the intended audio bits per sample by using the AUD_SAMPSZ register bit.

The rate tracker can be configured using the RADBW, RADHGTHR and RADDEP registers. The rate tracker implements a digital PLL which attempts to match changes in the frequency difference over short time periods while managing the buffering depth over longer time periods. The RAD_ACW register field controls the gain of the frequency adaptive portion, with smaller numbers producing faster response. The RAD_INTG register field controls the gain of the buffering-depth-adaptive portion, with larger numbers providing faster response. The RADDEP register programs the desired buffering depth in audio frames.

To accommodate exceptional conditions, the rate adapter enters slewing mode if the difference between the current buffering depth and the desired buffering depth exceeds a limit. When the limit is exceeded then the rate adapter takes special measures to restore the buffering depth to within this limit. The limit is set by the RADHGTHR register. The exceptional action to take is set by the RAD_SLWTYP register bit. If RAD_SLWTYP bit is set then the rate adapter sets the number of samples per packet to be always +1 or -1 from the nominal number of 48 while the condition persists. If

the RAD_SLWTYP bit is not set then the rate tracker's bandwidth is increased while the condition persists.

7.3.5.3 Firmware Interrupts

The host-to-host controller operates with a single interrupt with multiple event sources. To resolve which of the several possible events had caused the interrupt, the firmware should interrogate the H2HIRQF register. The H2HIRQF register is clear-on-read. Event sources can be enabled/disabled using the H2HIRQENA register.

7.3.6 Charge Measurement Digital Back End (DBE)

7.3.6.1 Description/Interface

7.3.6.1.1 Background

The USB protocol is no longer used merely for transmission of information - it is now the primary interface by which the wired charging of mobile devices takes place. Charging takes place from the 5V "VBUS" line in the 4-wire USB interface.

The customer has requested a Charge Measurement feature on Boston by which the ASIC will measure the total charge drawn per port over a programmably-defined time interval not to exceed 3 hours. The purpose of this feature is to limit the amount of charge that can be drawn off of the car battery after the car alternator has been turned off. Boston is required to disable all USB downstream port charging after the first of: elapsing of the aforementioned time interval OR reaching a programmable charge limit of 1-20 Amp-hours.

Per-port power control switching is implemented with an external power MOSFET interposed between a 5V master supply and the downstream "VBUS" line. Although this port power control switch is implemented by an independent circuit residing external to Boston, it is controlled by Boston as depicted in Figure 7-1.

Due to customer cost considerations, Boston must make current measurements not by measuring the voltage across an accurate current-sense resistor but rather by measuring the voltage across the power MOSFET switch itself. The difficulty here is that the resistance of the MOSFET switch varies substantially as a function of the switch junction temperature, a quantity to which Boston does not have direct observability. Rather, Boston must infer the value of this temperature using a modified version of the equation shown on page 15 of the TPS2560 data sheet and then compute the value of the resistance based on Figure 14 of this data sheet. Obtaining an accurate estimate of the true current requires excellent knowledge of this model for a given part, including part-to-part variation. In addition, the resistance of this MOSFET switch varies dramatically if it is opened by Boston deasserting its

PRTCTL signal or if the switch enters its thermal shutdown state.

7.3.6.1.2 System Partitioning

The Charge Measurement System is partitioned into analog and digital portions. The master clock for the entire system is the ADCCLK (1MHz).

The analog portion (described earlier in Section 7.2.3) consists of 4 port instantiations, with the input of each connected across one of the 4 downstream port power control (PPC) switches. Note that not all implementations of Boston will interface with 4 downstream PPC switches, 1, 2, and 3 PPC switches are also possible. The analog portion performs voltage amplification and anti-alias filtering. Offset is cancelled to first order in the analog portion by flipping the analog circuit and taking consecutive measurements destined to be subtracted from each other in the digital back end. The voltage at the outputs of the 4 port instantiations is sampled by a single ADC, which is multiplexed between the 4 instantiations.
The outputs of the ADC are in the following order (in time) and are delivered to the digital back end in streaming fashion:

Port 0 (CVOUT0), Port 1 (CVOUT1), Port 2 (CVOUT2), Port 3 (CVOUT3), Port 0 (CVOUT0FLIP), Port 1 (CVOUT1FLIP), Port 2(CVOUT2FLIP), Port 3(CVOUT3FLIP), Port 0 (CVOUT0), ...

The spacing between the ADC STRBOs corresponding to each of these outputs is 256 ADCCLK clock cycles.

The digital portion is partitioned into 3 subsystems:
1. Analog Circuit and ADC Management.
2. ADC to Current Conversion Subsystem.
3. Current to IRQ Subsystem.

The Analog Circuit and ADC Management Subsystem:

- Sends a series of clock and strobe-in signals to the analog circuitry to clear state variables and force ADC sampling.
- Sends a series of ADC samples, strobe-out, and channel & flip indicator signals to the “ADC to Current Conversion Subsystem”.

The ADC to Current Conversion Subsystem:

- Runs a simplified ALU intended to model the external port power control switch resistance as a function of Boston junction temperature (assumed to be the same as ambient temperature within the Hub module) and the voltages across the external port power control switches.

- Supports 32 bit state variables to model very large thermal time constants (on the order of 10 seconds).
- Performs blanking of ADC output data based on the PRTCTL signal for a given port. That is, if PRTCTL1 is low, then ADC data from Channel 1 is set to zero prior to passing the data downstream because under this condition it is known that the current is actually zero.
- Sends four current values (in units of mA) (IOUT0, IOUT1, IOUT2, IOUT3) to the Current to IRQ Subsystem.

The Current to IRQ Subsystem:

- Performs a number of accumulation, filtering, and comparison operations, which when triggered, result in the Charge Measurement IRQ being thrown.
- Accumulates all of the IOUT measurements over time and compares the summation of those computations to an absolute charge (Amp-hour) limit.
- Low-pass filters the IOUT measurement and throws an IRQ when the low-pass filtered current on any of the ports drops below a programmable threshold.
- Throws an IRQ if the current in any of the ports or the total current exceeds a dynamically programmable threshold.

7.3.6.1.3 Interface

The interface to the Charge Measurement System is made nearly entirely through the digital portion. There exists an overall enable signal from the firmware which turns on the entire system.

Analog Circuit and ADC Management:

Testbench access. These inputs should be tied to constant logic levels in the digital top level

```
input     fast_bias,   // set to 1 in testbench to skip past initial bias state after only one cycle (should be tied to 1'b0 in the top level hookup).
input     gate_adcclk, // set to 1 in testbench to not have clock edges when the adc is not biased (may want to move into a register for test access).
```

Configuration interface

```
input  [3:0] chan_exists,  // set to one by firmware (register exists in some other block) to indicate whether to not bias each channel
```

Baud rate interface

```
input     re_strb,   // rise edge strobe - one indicates that next clk fall edge will correspond to an aclk rise edge (ignored if full_spd == 1)
input     fe_strb,   // fall edge strobe = one indicates that next clk fall edge will correspond to an aclk fall edge (ignored if full_spd == 1)
```

ADC interface

```
output    adc_clk,   // clock to adc
output    adc_strbi, // control strobe to adc to get it to take a new sample
```

output adc_ena, // bias enable for the ADC
output [1:0] adc_sampcyc, // number of clocks/samples control to adc
// 00 => 1 cycle
// 01 => 2 cycles
// 10 => 4 cycles
// 11 => 8 cycles
input adc_strbo, // output strobe from the adc to indicate a new sample is available
input [11:0] adc_dval, // output data from adc, also make this available to firmware.

Analog interface

output [3:0] ana_ena, // bias enable for analog blocks
output [3:0] adc_chan, // control for analog mux. each of the four channels, is enabled by making its output high level
output [3:0] ana_clr, // indication to clear the integrator, analog should retime this signal
output [3:0] ana_frz, // indication to freeze the integrator value, analog should retime this signal
output [3:0] ana_chp // polarity indication for chopping, analog should retime this signal.

ADC to Current Conversion Subsystem:

input [11:0] adc_dval, // output data from adc.
input [3:0] portctl, // Port Control signal which is outgoing from the Boston pad to the external PPC switch.
input [6:0] prog_ctr_limit, // Specifies the end of the program.
input/output [26:0] prog_reg_data, // Data to be loaded into / read from the Program Data Register Array.
input [6:0] prog_reg_addr, // Address of data to be loaded into / read from the Program Data Register Array.
input prog_reg_fw_ctrl, //Specifies whether FW should be in control of program data register for reading back a program.
input [15:0] varb_reg_data_in, //Data to be loaded into the Variable Data Register Array.
output [15:0] varb_reg_data_out, //Data to be read back from the Variable Data Register Data Array.
input varb_reg_fw_ctrl, //Provide FW control of Variable Data Register Array .
input [5:0] varb_reg_addr_in, //Address of data to be loaded into the Variable Data Register Array.
input [5:0] varb_reg_addr_out, //Address of data to be read from the Variable Data Register Array.
input [5:0] iout0_prog_result_reg_addr, //Address of data from Variable Data Register Array to be read into the Current to IRQ subsystem for Port 0
input [5:0] iout1_prog_result_reg_addr, //Address of data from Variable Data Register Array to be read into the Current to IRQ subsystem for Port 1
input [5:0] iout2_prog_result_reg_addr, //Address of data from Variable Data Register Array to be read into the Current to IRQ subsystem for Port 2
input [5:0] iout3_prog_result_reg_addr, //Address of data from Variable Data Register Array to be read into the Current to IRQ subsystem for Port 3

Current to IRQ Subsystem:

input [3:0] lpf_tc, // Sets time constant of low pass filter (also called "X" in diagram).
input [15:0] iout0minthresh, // Sets min current trip threshold for port 0.
input [15:0] iout1minthresh, // Sets min current trip threshold for port 1.
input [15:0] iout2minthresh, // Sets min current trip threshold for port 2.
input [15:0] iout3minthresh, // Sets min current trip threshold for port 3.
input [15:0] iout0maxthresh, // Sets max current trip threshold for port 0.
input [15:0] iout1maxthresh, // Sets max current trip threshold for port 1.
input [15:0] iout2maxthresh, // Sets max current trip threshold for port 2.

```
input [15:0] iout3maxthresh, // Sets max current trip threshold for port 3.
input [15:0] iouttotmaxthresh, // Sets max current trip threshold for all ports in total.
input [15:0] ioutacmmaxthresh, // Sets max current trip threshold for all ports in total after accumulation.
input [9:0] chgmeasirqenable, //Enable each of the 10 IRQ events of this subsystem.

input [7:0] clear_state_var, //Clear each of the 8 state variables in this subsystem. Clear register to 0 after reset is complete.

output        irq_trip, // Assert when an IRQ has been thrown
output [9:0] irq_result, // Show which of 10 IRQ events tripped the Charge Measurement IRQ.
output [15:0] iout0, // Output current of port 0.
output [15:0] iout1, // Output current of port 1.
output [15:0] iout2, // Output current of port 2.
output [15:0] iout3, // Output current of port 3.
output [15:0] iout0lpf, // Low-pass filtered output current of port 0.
output [15:0] iout1lpf, // Low-pass filtered output current of port 1.
output [15:0] iout2lpf, // Low-pass filtered output current of port 2.
output [15:0] iout3lpf, // Low-pass filtered output current of port 3.
output [35:0] iout0accum, // Output charge through port 0 since last state clear.
output [35:0] iout1accum, // Output charge through port 1 since last state clear.
output [35:0] iout2accum, // Output charge through port 2 since last state clear.
output [35:0] iout3accum, // Output charge through port 3 since last state clear.
output [37:0] iouttotaccum, // Output charge through all ports.
output [17:0] iouttot, // Output current through all ports.
```

7.3.6.2 Coarse Requirements / Features

- Current measurements must be inferred by measuring the voltage across the TPS2560 port power control (PPC) switch and by Boston junction temperature.
- The resistance range of this PPC switch is 30mΩ to 70mΩ.
- The complete system is required to measure charge integrated over a time interval of up to 3 hours and compare this total charge to a limit of 1-20 Amp hours.
- This comparison should be done with an error of no more than 25%.
- The minimum charging current limit allowed in USB is 100mA. Integrating this over 3 hours on all 4 ports yields a total charge of 1.2 Amp-hours.
- The maximum current to be allowed per port by the customer is 3A. In this case, the input voltage to the Charge Measurement circuit is 210mV.
- The complete system is also required to report overcurrent events for each port independently and for all active ports together.
- These overcurrent events should trip with an error (deviation from nominal) of less than 15% for load currents above 500mA.

- The thresholds for these overcurrent events must be dynamically programmable to accommodate different current allocations to each port.
- The complete system is required to report undercurrent events for each port independently. This is because the lack of current is the only way for the system to know that a device may be disconnected in Boston Charging Mode.
- The thresholds for these undercurrent events must be programmable and must extend well below 50mA.
- The current consumed by the digital back end of this feature should be less than 100uA, time averaged over 10ms.
- Performance accuracy is guaranteed only with a perfect Hub module / PPC ASIC / Boston ASIC thermal circuit model implemented in the Charge Measurement DBE DSP section.

7.3.6.3 Architecture

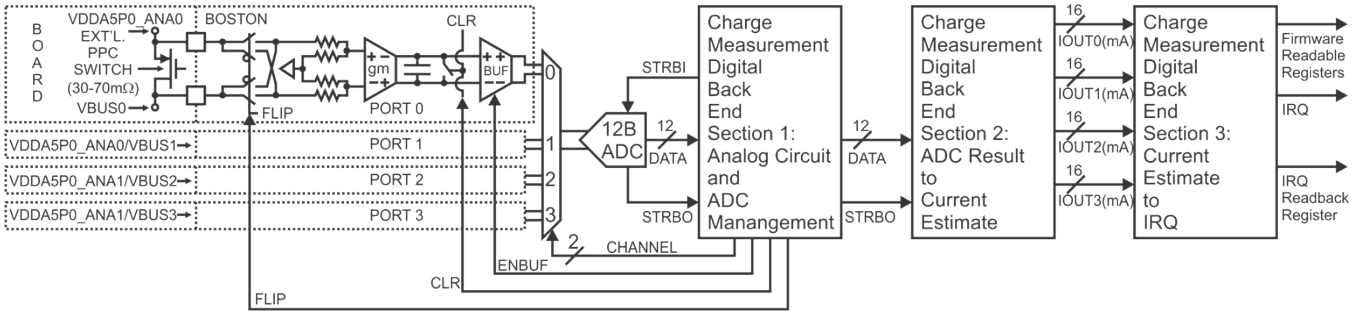


Figure 7-39 Boston Charge Measurement System End-to-End Diagram

The overall architecture of the Charge Measurement DBE is depicted in Figure 7-39, showing the relationship between the three constituent subsystems and the data inputs/outputs at each interface.

7.3.6.3.1 Analog Circuit and ADC Management

- The Analog Circuit and ADC management portion of the DBE performs the following tasks:
  - Sends a strobe-in (STRBI) signal to the ADC.
  - Keeps the ADC buffer output valid for the 1MHz clock cycle upon which STRBI is asserted and for 2 additional 1MHz clock cycles thereafter.
  - Clears the integration capacitor for a particular measurement circuit after sampling of that measurement circuit by the ADC.
  - Rotates the connections for each of the 4 buffers to the ADC.

- Provides 2 bits out indicating the active connection of the ADC. Keep these signals valid until after ADC STRBO is de-asserted.
- Provides 1 bit out indicating the chopping state. This signal is not kept valid until after ADC STRBO is de-asserted but at the time ADC_STRBO is de-asserted it is actually the negation of the state under which the sample was obtained.

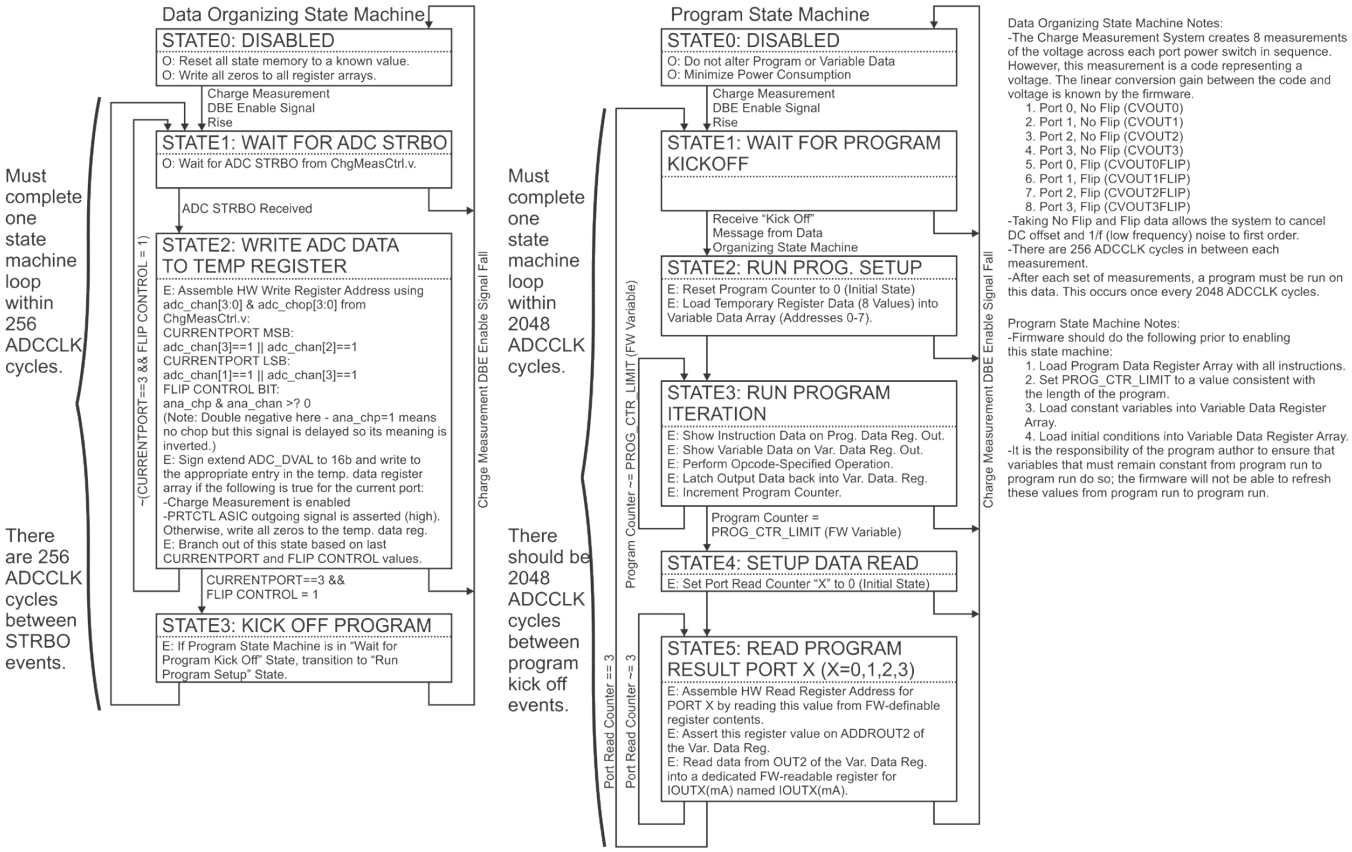
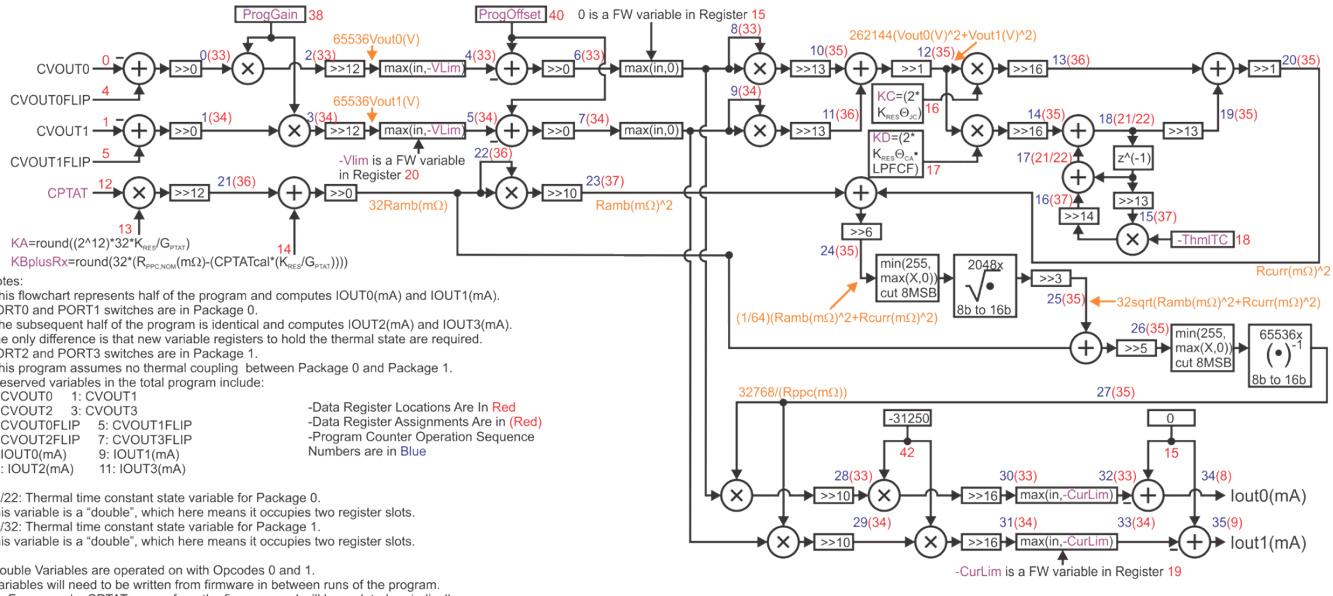


Figure 7-40 Boston Charge Measurement Digital Back End ALU Control State Diagrams

7.3.6.3.2 ADC Result to Current Estimate

The ADC output-to-current ALU subsystem is controlled by the state diagrams shown in Figure 7-40. Shown in this diagram are two state machines:

- Data Organizing State Machine:
  - Waits for ADC Output Strobe (STRBO).
  - Write ADC Data to a temporary register.
  - Kicks off a run of the program for a completed set of data from the 4 channels.
- Program State Machine:
  - Waits for the program to be kicked off.
  - Sets up the program.
  - Steps through the program memory until the program is completed.
  - Reads the program result into the IOUT result registers.



Notes:
-This flowchart represents half of the program and computes IOUT0(mA) and IOUT1(mA).
PORT0 and PORT1 switches are in Package 0.
-The subsequent half of the program is identical and computes IOUT2(mA) and IOUT3(mA).
The only difference is that new variable registers to hold the thermal state are required.
PORT2 and PORT3 switches are in Package 1.
-This program assumes no thermal coupling between Package 0 and Package 1.
-Reserved variables in the total program include:
0: CVOUT0 1: CVOUT1
2: CVOUT2 3: CVOUT3
4: CVOUT0FLIP 5: CVOUT1FLIP
6: CVOUT2FLIP 7: CVOUT3FLIP
8: IOUT0(mA) 9: IOUT1(mA)
10: IOUT2(mA) 11: IOUT3(mA)

-Data Register Locations Are In Red
-Data Register Assignments Are in (Red)
-Program Counter Operation Sequence Numbers are in Blue

21/22: Thermal time constant state variable for Package 0.
This variable is a "double", which here means it occupies two register slots.
31/32: Thermal time constant state variable for Package 1.
This variable is a "double", which here means it occupies two register slots.

-Double Variables are operated on with Opcodes 0 and 1.
-Variables will need to be written from firmware in between runs of the program.
-For example, CPTAT comes from the firmware and will be updated periodically.
-When CPTAT is changed, the firmware needs to update the value in the Variable Data Register Array for the next program run.

-Operations 0-1 Perform the cancellation inherent in the chopping operation.
-Operations 2-5 Scale input signal and place a ceiling on the input signal to minimize the effect of an overcurrent situation on the state memory.
-Operations 6-18 Compute the effect of power dissipation and junction temperature rise on the PORT switch resistance, including the thermal time constant.
-Operations 19-26 Compute the switch resistance of PORT0 and PORT1.
-Operations 27-29 Divide switch voltage by resistance to obtain current.
-Operations 30-31 Convert units: V/mΩ to mA.
-Operations 32-35 Model the current limiting behavior of the PORT switches.

Variables:
CPTAT: The value of the PTAT measurement generated by the HKADC.
CPTATcal: The value of the PTAT measurement at 27°C stored in the calibration FLASH memory. Typical value is 1751.
K_{RES}: The temperature coefficient of the port power control resistance in mΩ/°C. Typical value is 0.18mΩ/°C from TPS2560 Data Sheet.
$R_{PPC,NOM}$: The resistance of the port power control switch in mΩ. Typical value is 47mΩ from TPS2560 Data Sheet.
G_{PTAT}: ΔCPTAT/ΔTEMP. Simulated/measured to be 5.8. Does not change significantly over corner/mismatch.
ProgGain: Converts the ADC code value into a power-of-2 multiple of Vout. Typical value is 18401. This value is computed from a calibrated value as shown below.
ProgGain is equal to round(65536*((2^12)/(2^14))*(50e-3Ω/1000)*(Charge Measurement Gain Error Correction Factor)).
Vlim: 65536*Vmax, where Vmax is the maximum voltage drop across the switch to be processed by the system.
The recommended value for Vmax=Imax*Rmax=(3A)*(70e-3Ω). Then, Vlim=13762.
ProgOffset: round((ProgGain/(2^12))*((2^14)/Charge Meas. Gain Err. Corrt. Fact.)*(Charge Meas. Offset Err. Corrt. Fact.)), where the latter two values are calibrated values.
Θ_{JC}: The thermal resistance of the port power control switch from junction to contact, in °C/kW. Typical value is 10000°C/kW from TPS2560 Data Sheet.
Θ_{CA}: The thermal resistance of the port power control switch from junction to contact, in °C/kW. Typical value is 30000°C/kW from TPS2560 Data Sheet.
ThmITC: A constant inversely proportional to the thermal time constant of the PPC switch in the hub module. Typical value is 22000.
This constant will change from hub module to hub module and depends on module size, air flow, etc.
Is equal to round((2^14)*(2^13)*(1/(Fs*Thermal Time Constant)). Nominal value of Fs is 488.125Hz.
Thermal time constant is guessed to be about 12.5 seconds from review of various application notes.
LPFCF: A normalization constant to ensure that the low pass filter modeling the thermal impedance network has the proper gain.
Its value is equal to ThmITC/(2^14). For the typical variable values above, its value is 1.34.
CurLim: The current limit of the port power control switch, in mA. Typical value is 2800.

Figure 7-41 Boston Charge Measurement Example Program DSP Flowchart

An example half-DSP program which computes the output current from the ADC voltage codes is shown in Figure 7-41. In addition to standard arithmetic operations, the program requires the use of look-up tables for square root and inversion operations. The need for the square root operation fundamentally derives from the linear relationship between switch resistance and junction temperature and the square-law relation between junction temperature and voltage across the switch.

The program structure is developed by utilizing the information in the TPS2560 data sheet (namely the equations on Pages 2 and 15, and Figure 14 on Page 8) to solve for the port power control switch junction temperature, then for the on-resistance of the port power control switch. Finally, Ohm's law is used to solve for the current flowing through the switch. The program structure in Figure 7-41 also

makes use of various scalings in order to maximize the signal-to-quantization noise ratio without resulting in numerical overflow. In Figure 7-41 above, Ramb represents the component of the switch resistance present in the absence of self-heating, while Rcurr represents the component of the switch resistance added by self-heating effects. Rppc is the total on-resistance of the switch.

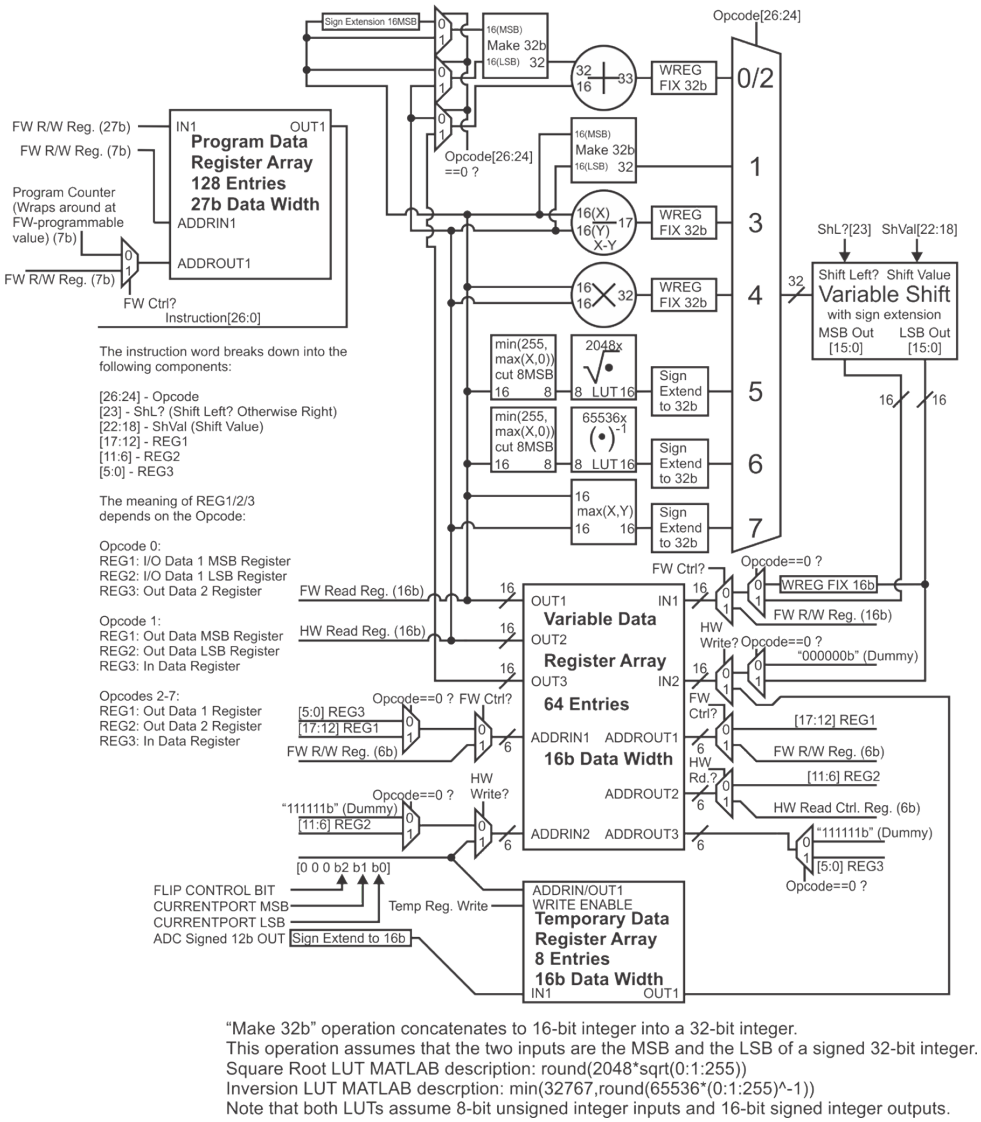


Figure 7-42 Boston Charge Measurement DBE ADC to Current ALU Subsystem

The DSP thermal modeling program is executed by an ALU, shown in Figure 7-42, whose program can be read in from the firmware into the program data register array. This allows customization of a given thermal modeling program to a particular USB Hub module. The firmware can also populate the variable data register array to set constants and initial conditions for the program.

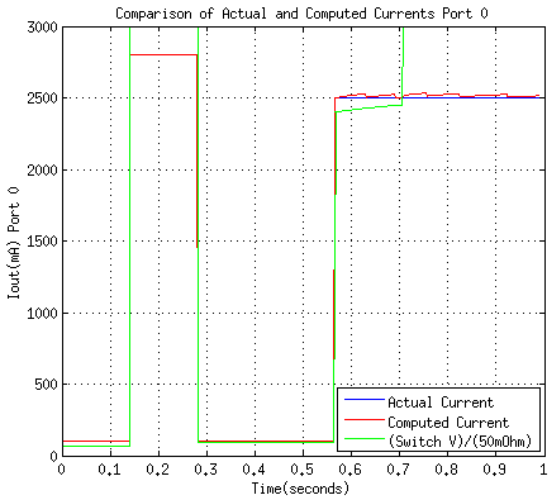


Figure 7-43 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 0

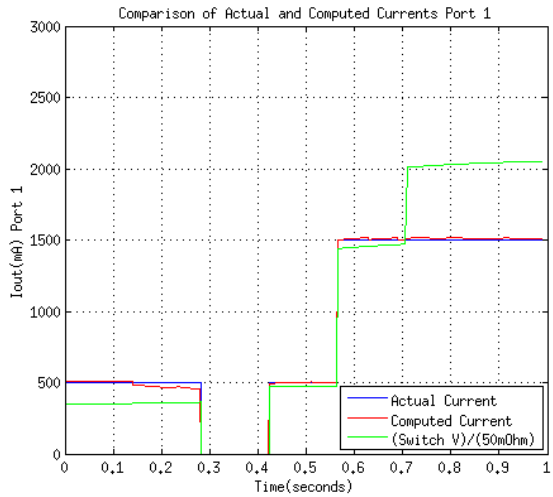


Figure 7-44 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 1

The results of a sample program run over varying current load and temperature conditions are shown in Figure 7-43, Figure 7-44, Figure 7-45, and Figure 7-46. In this case, the thermal model of the Hub module and the junction temperature-on resistance relation of the VBUS switch are assumed to be known perfectly. Note that close agreement is obtained between the actual and computed current even when the voltage across the switch is changing due to changes in ambient temperature (abrupt jumps in the green curve (voltage) for a given current) and changes in junction temperature due to device self-heating (smooth changes in the voltage for a given current). The one exception to this close agreement occurs between 3 and 6 seconds in Figure 7-44. This discrepancy occurs when one of the VBUS switches in a two-port package experiences a current limiting event at cold temperature. In this case, the voltage across Port 0 rails out the internal analog signal chain. The DBE program clips the computed current properly to 2800mA but because this clipping is done after the thermal computation, the portion of the program responsible for thermal computation still interprets the amount of current flowing through Port 0 as significantly higher than it really is. Because the Port 0 current is overestimated early on in the program, the amount of self-heating within the two-port package is overestimated and hence the resistance of Port 1 is overestimated. Hence, the computed current on

Port 1 is computed to be about 6% less than the actual current for the duration of this overcurrent event. Future versions of the firmware may be improved to prevent this scenario.

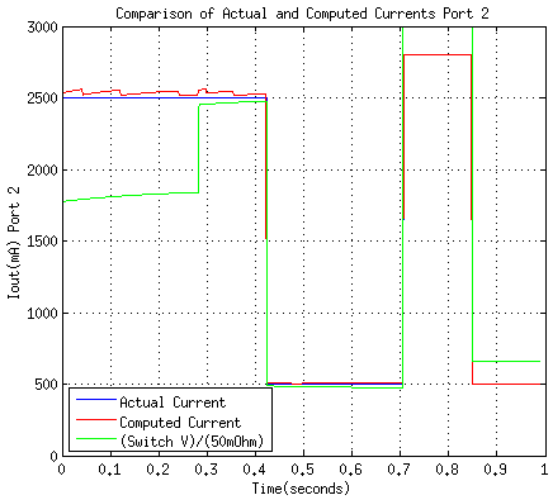


Figure 7-45 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 2

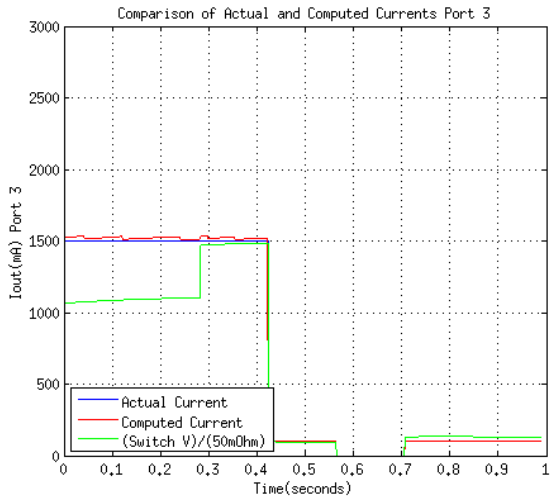


Figure 7-46 Boston Charge Measurement Example Program DSP Result Over Varying Temperature and Current Load Conditions – Port 3

7.3.6.3.3 Current Estimate to IRQ (IRQ 10)

The current estimate to IRQ portion of the charge measurement digital back end is shown in Figure 7-47. The specification for this subsystem is provided in the notes of Figure 7-47. The Charge Measurement IRQ is assigned to IRQ 10.

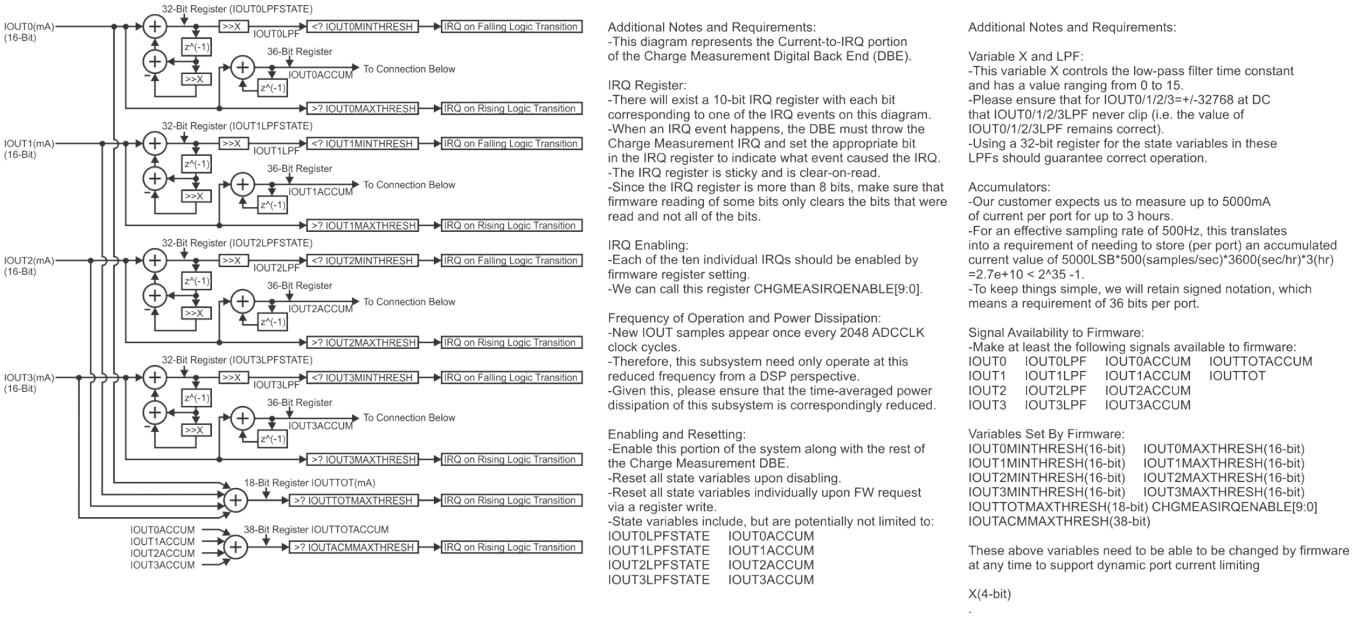


Figure 7-47 Boston Charge Measurement DBE Current to Interrupt Subsystem

7.3.7 Housekeeping ADC Digital Back End (DBE)

7.3.7.1 Simple Specification

The Housekeeping ADC interface must satisfy the requirements of the following documents in conjunction with minimal firmware oversight. In other words, the specifications of the documents below must be satisfied using interrupts to the microcontroller rather than ADC polling due to the requirement of reserving the bulk of the microcontroller clock cycles for managing high-throughput SD Card operation. The specifications of the following documents must also be satisfied while the USB Device / Mass Storage Class Device / SD Host Driver firmware is in operation:

- USB Battery Charging Specification Revision 1.2 (with errata) October 12, 2011 [2]
- USB Charger Emulation Flow.pdf (Specification from the customer)
- IMAG0137-RobFlexibleChargingEmulation.jpg (Specification from the customer)

7.3.7.2 Merging Customer Requirements with USB BCS Requirements

- USB BCS 1.2 specifies a number of voltage comparators to be used to detect whether the USBDN_DP and USBDN_DM pins are within specific voltage ranges.

- For example, Figure 3-7 in the USB BCS documentation shows a signal PRTBL_DET that is high when DP is between VDAT_REF (max 0.4V) and VLGC (min 0.8V).

- However, the customer has specified the use of ADCs rather than comparators on the USBDN_DP and USBDN_DM lines to ascertain voltages on these lines.
  - The interface to the ADCs is specified as part of the Flexible Charging Emulator Termination.
  - Using a shared ADC on all of these lines is also a good idea because the time constants described in the BCS are at minimum 10ms, but the ADC can actually sample 32 nodes in < 1ms.
  - Using a shared ADC on all of these lines is superior to using a programmable threshold comparator for each line because area is saved, power is saved, and analog design time is saved.
  - Using a shared ADC on all of these lines also gives greater allowable resolution in setting thresholds.
- The problem with using an ADC that is operated from firmware is that one can only use this ADC when an event is detected but here it is necessary to use an ADC to detect an event.
  - One solution to this problem is continual polling of ADC input nodes of interest with the MCU/firmware.
  - However, the Boston MCU will be potentially occupied fully with the USB Device / Mass Storage Class Device / SD Host Driver firmware.
  - Boston will have as many as 22 nodes (provisioning for sampling 32 nodes shall be made) to be polled, each taking about 25us to be polled.
  - This represents a total housekeeping sweep time of 0.8ms if no ADC reconfiguration is performed.
  - A housekeeping sweep is defined as one pass of the all of the housekeeping ADC polling operations.
  - In order for this task to not be a major burden on the MCU/firmware, each housekeeping sweep would need to take place once every 100ms or longer.
  - The problem is that the requirements of the USB BCS demand a response to a voltage change event within 10ms-20ms of the event happening (see Table 5-5 of the USB BCS).
  - Therefore, one cannot satisfy the twin requirements of high SD Card throughput and firmware-driven housekeeping ADC operations.
  - The typical use case will be one housekeeping sweep every 1ms.

- The proposed solution to the problem is to have a digital back end (DBE) to drive the housekeeping ADC operation.
  - The DBE would, among other things listed below, manage a housekeeping schedule that specifies when ADC polling operations are to take place and with what ADC settings.
  - The housekeeping schedule is defined by firmware, but run in digital hardware on the ADCCLK domain.
  - The DBE would generate interrupts on a single IRQ line (IRQ 8) when a given ADC voltage moves from outside a programmably-specified range to inside the programmably-specified range when programmed to do so.
  - The DBE would generate interrupts on a single IRQ line (IRQ 8) when a given ADC voltage moves from inside a programmably-specified range to outside the programmably-specified range when programmed to do so.
  - Each IRQ event needs to have a programmable debounce interval associated with it.
  - The firmware would then need to read a single clear-on-read register to see which of the 32 possible housekeeping entries had experienced a threshold trip.

7.3.7.3 HKADC (Housekeeping ADC) DBE Organization

7.3.7.3.1 Organization

Conceptually, the HKADC DBE should be organized into a table format with ADC MUX register settings defining the rows (of which there should be 32) and should be organized into a table format with the following columns:

a. ADC MUX register setting (6 bits) (default: 000000)

b. ADC Sample Wait Cycle (3 bits) (default: 000) (Use higher value for high impedance nodes)

c. Threshold Low Code (12 bits) (default: 000000000000)

d. Threshold High Code (12 bits) (default: 000000000000)

e. Low threshold code exceeded (read only bit)?

f. High threshold code exceeded (read only bit)?

g. Outside -> inside threshold IRQ armed (one bit)? (default:0)

h. Inside -> outside threshold IRQ armed (one bit)? (default:0)

i. Delay (in 1MHz clock cycles) to next schedule entry ADC measurement (12 bits) (default:0).

j. Delay (in 1MHz clock cycles) between ADC MUX register setting and actual ADC sampling (4 bits) (default:0).

k. Entry Active (one bit) (When bit is set high, this HKADC schedule table entry is serviced. When this bit is set low, this HKADC schedule table entry is not serviced) (default:0).

l. One Shot Set (one bit) (To be explained below). (default:0)

m. Last ADC output result (read-only register - 12 bits).

n. IRQ debounce interval - number of table entry passes (4 bits). (default:0)

- These columns should be organized in such a way to minimize the number of reads and writes to set and obtain relevant information.
  - Items to be read frequently include items e,f,m (14 bits).
  - Items to be written frequently include items g,h,k,l (4 bits).
- There should also exist a 32-bit register which provides an indication as to which entries in the HKADC table have caused an IRQ event.
  - This register should be clear-on-read.
  - This register should be sticky (i.e. the register is not cleared until the firmware reads it).
- This table, as described, represents 2944 register bits.
  - Please consider the use of SRAM or Register file to store these bits during operation.
  - In this case, the firmware would be responsible for populating the memory prior to use of the HKADC DBE.
  - However, upon digital reset, the digital circuitry should populate the table memory with the default values.

7.3.7.3.2 Operation

The HKADC will cycle through all of the items of the housekeeping schedule one by one, wrapping around to the beginning of the list when all entries have been serviced. At each item, the following must be performed when the Entry Active bit is high for that entry. At each item, nothing should be done while the Entry Active bit is low for that entry.

a. ADC MUX register setting is loaded into digital ADC MUX register.

b. ADC configuration register settings are set.

c. The specified number of delay (1MHz) cycles are taken prior to sampling by the ADC.

d. STRBI is asserted to the ADC.

e. The specified number of sampling (1MHz) cycles are taken by the ADC to allow the sampled voltage to settle on the sampling capacitor.

f. Since AUTOEN=1 for the HKADC, 8 1MHz delay cycles are required for the ADC to bias itself.
g. After 13 1MHz delay cycles, STRBO is received from the ADC.
h. After receiving STRBO, load the ADC digital output result into the Last ADC Result register.
i. Check to see if the new ADC output now falls into the threshold limits.
j. Consider the condition in which the ADC output falls into the limits as a signal A at logic '1'.
k. Consider the condition in which the ADC output falls outside the limits as a signal A at logic '0'.
l. Place the logic level of signal A into a readable register bit.
m. If the threshold IRQ is armed for this table entry, throw an IRQ (a shared IRQ for the whole table) when the logic level of signal "A" changes states from 0 to 1 and stays at 1 for the duration of the IRQ debounce interval.
n. Check to see if the new ADC output now falls outside the threshold limits.
o. Consider the condition in which the ADC output falls into the limits as a signal A at logic '1'.
p. Consider the condition in which the ADC output falls outside the limits as a signal A at logic '0'.
q. Place the logic level of signal A into a readable register bit.
r. If the threshold IRQ is armed for this table entry, throw an IRQ (a shared IRQ for the whole table) when the logic level of signal "A" changes states from 1 to 0 and stays at 0 for the duration of the IRQ debounce interval.
s. The specified number of delay cycles are taken before moving to the next housekeeping schedule item.
t. One-Shots will be explained below.

The following bits must be settable during HKADC DBE operation:

a. Outside -> inside threshold IRQ armed?
b. Inside -> outside threshold IRQ armed?
c. Entry Active?
d. One Shot Set?

In addition, when a table entry is not active, all non-read-only registers of that table entry must be writable.

7.3.7.3.3 One Shots

One limitation of the HKADC DBE scheduler is that it precludes the use of arbitrary ADC scheduling as has been implemented in other AyDeeKay ASICs. To incorporate the flexibility of this feature into the HKADC DBE scheduler, the concept of a One Shot is introduced. The One Shot Set bit works similar to the ADC_START bit in RedHead in that it forces an ADC operation on a given net as soon as possible. In this case, the One Shot feature will work as follows:

a. If the One Shot bit in a table entry is set, the HKADC DBE attends to that entry immediately after its current ADC table entry measurement and comparison operation has been completed.
b. If multiple One Shot registers are set, then the HKADC DBE attends to them in the order in which they appear in the HKADC DBE table.
c. Once all active One Shot table entries are attended to, the HKADC DBE resumes its housekeeping schedule at the same point at which it left off.
d. The One Shot register should automatically clear when a conversion is done.

7.3.7.4 System Implications

7.3.7.4.1 Power Dissipation

The maximum power dissipation of this feature must fit into the 400uA total suspend current budget for digital circuitry.

7.3.7.4.2 Enable/Disable

It should be possible to disable this feature if for no other reason than to determine its power consumption during debugging.

7.3.7.4.3 Sharing of ADC

The HKADC DBE will not be sharing its associated ADC with any other digital interface on the ASIC.

7.3.7.4.4 Control over ADC

This HKADC DBE should completely operate the ADC and ADC MUX.

7.3.7.4.5 Test

It should be possible to test the functionality of the ADC and ADC MUX via serial-wire writes through the HKADC DBE interface.

7.3.8 Miscellaneous Digital Features

7.3.8.1 LPOSC (10kHz) Calibration Counter

- The LPCLK is used to clock the SLEEP TIMER.

- The SLEEP TIMER needs to be accurate within ± 8%. (1 hour +/- 5 minutes).
- However, the LPCLK frequency varies about ± 50% over PVT corner.
- In order to fix this disparity, functionality is included within the Boston core digital circuitry to 'calibrate' the sleep timer by counting how many ICCLK (divided down) periods occur within 10 LPCLK periods.
- According to the customer, it can be assumed that the result of this calibration will remain valid over the Charging interval determined by the SLEEP TIMER initial value.
- Using this information, firmware can then load a value representing the number of LPCLK clock cycles in one minute into the 3.3V PMU sequencer to be used during Low Power Current Detect Wake-Up sleep.

The interface to this subsystem should be as follows:

- There should exist a register bit that firmware will set to run the counter, essentially strobing the frequency difference between the ICCLK (divided down) and the LPCLK.
- This register bit should auto-clear when the count is complete.
- The counter should be a 16-bit counter. The reason is that there may be up to 10*(4200000/6000)=7000 cycles counted, so at least 13 bits are required. The extra 3 bits are for margin.
- In the other extreme case, the count is 10*(975609/16000)=610, which still gives good resolution on the LPCLK frequency relative to the ICCLK (divided down) frequency.
- The counter should be cleared at the beginning of each new strobe to this subsystem.
- The counter value at the end of every count should be stored on a set of output registers that is visible to the firmware.

7.3.8.2 Suspend Clock Handover and Suspend IRQ (IRQ14)

The DCLK should be assigned one of two signals: 1) ICCLK, or 2) USBCLK30 (backup: ICCLK) based on firmware settings.

In the case of 2):

- USBCLK30 should be provided to DCLK when both the "suspend" and "clkusable" signals described in the USB HUB digital IP documentation are logic high. For the purposes of this document, one can call this state USBCLK30_OK.
- When one or both of these signals transitions to logic low, the clock must switch over to ICOSC. For the purposes of this document, one can call this state USBCLK30_NOK.

- When both "suspend" and "clkusable" signals are again logic high, the clock must switch back to USBCLK30.
- Additional control of the PLL are provided. FW_SUSPEND(Bit 5 of 0x50013017, **BYTE WRITE ONLY**) is used to force turning off PLL. PLL_FORCE_ON(Bit 1 of 0x50000000) is used to force PLL staying on. If both of them are set, PLL will stay on.

The following requirements must also be met for this feature:

- Clock handovers must take place such that short pulses are not generated.
  - Clock handover logic should be designed to ensure an orderly transition between clock sources.
  - For example, Charge Measurement and HKADC-based processes that run from ADCCLK should not be disrupted during a clock source handover.
- When in suspend mode, DCLK should be gated from all USB digital logic.
  - The reason is because the USB digital logic is not designed to operate from a clock while the HUB is in suspend.
  - Another reason is that a goal of the design is to minimize current while in suspend.
- An IRQ (IRQ 14 - Suspend IRQ) should be generated whenever there is a transition from USBCLK30_OK to USBCLK30_NOK and vice versa.
  - The firmware should be able to read "suspend" and "clkusable" outputs from the USB PLL to determine which state had been transitioned into.
  - This is important because if the firmware engineers are using internal MCU timers for anything, the meaning of those timers will change when the source of DCLK changes.

7.3.8.3 Port Power Control (PPC) Digital Interface

The purpose of the PPC digital interface is to provide the external interface between the USB HUB IP and/or Firmware to access the external PPC module.

The external PPC module used in the customer's legacy solution is described in the following data sheet:
TPS2066C Port Power Control Switch.pdf

The external PPC module used in the customer's future solution is described in the following data sheet:
TPS2560 Adjustable Current Limit Port Power Control.pdf

7.3.8.3.1 Interface

Each of the 4 exposed downstream ports must accommodate the following signals:

External Pads:
Each Port will have 2 external pads (where X stands for the port number ranging from 0-3).

- PRTCTL_X - Connects to the external module enable signal
- FAULT_NX - Connects to the external module fault_bar signal

Internal Control Lines:

- DWN_DRVVBUS_USB (Incoming from USB Digital IP, but must be visible to FW)
- DWN_DRVVBUS_FW (Incoming from FW/Digital) (Default: 0)
- DWN_DRVVBUS_MODE[2:0] (Incoming from FW/Digital) (Default:000)
- DWN_DRVVBUS (Outgoing to Pin)
- DWN_OVER_CURRENT_USB_N (Outgoing to FW)
- DWN_OVER_CURRENT_FW_N (Incoming from FW/Digital) (Default: 0)
- DWN_OVER_CURRENT_MODE[1:0] (Incoming from FW/Digital) (Default:00)
- DWN_OVER_CURRENT_N (Outgoing to USB Digital IP)

7.3.8.3.2 Functionality

DWN_DRVVBUS_MODE[2:0] dictates how DWN_DRVVBUS is controlled. This is for the most part a debug register. The intentional mode of operation is to set this register to 0b011. Note that ANDing is used because for the port to be active, all three of the firmware, USB HUB, and PPC switch itself must be OK with enabling this switch.

Table 7-18 DWN_DRVVBUS_MODE[2:0] Functionality Table

DWN_DRVVBUS_MODE[2:0]	**DWN_DRVVBUS==**
111	DWN_DRVVBUS_FW && DWN_DRVVBUS_USB && DWN_OVER_CURRENT_N
110	DWN_DRVVBUS_FW && DWN_OVER_CURRENT_N

Table 7-18 DWN_DRVVBUS_MODE[2:0] Functionality Table	
DWN_DRVVBUS_MODE[2:0]	**DWN_DRVVBUS==**
101	DWN_DRVVBUS_USB && DWN_OVER_CURRENT_N
100	DWN_OVER_CURRENT_N
011	DWN_DRVVBUS_FW && DWN_DRVVBUS_USB
010	DWN_DRVVBUS_FW
001	DWN_DRVVBUS_USB
000	logic 0

DWN_DRVVBUS is passed directly to the GPIO pin interfacing with PRTCTL_X.
FAULT_BAR of the TPS2066C is connected to FAULT_NX.

FAULT_NX must have a pull-up resistor of 10kΩ or 100kΩ that can be enabled on this line.
The resultant logic-level signal from FAULT_NX is then passed to DWN_OVER_CURRENT_USB_N(an active low signal).

DWN_OVER_CURRENT_MODE[1:0] dictates how DWN_OVER_CURRENT_N is controlled.
This is for the most part a debug register, the intentional mode of operation is to set this register to 0b01. Note that an ANDing operation is used because the signals of interest are active low and the USB Hub IP should respond to either the firmware **or** external PPC overcurrent signal.

Table 7-19 DWN_OVER_CURRENT_MODE[1:0] Functionality Table	
DWN_OVER_CURRENT_MODE[1:0]	**DWN_OVER_CURRENT_N==**
11	DWN_OVER_CURRENT_FW_N && DWN_OVER_CURRENT_USB_N
10	DWN_OVER_CURRENT_FW_N
01	DWN_OVER_CURRENT_USB_N
00	logic 1

7.3.8.3.3 Overcurrent IRQ (IRQ 11)

- Throw the Over Current IRQ (IRQ 11) to the MCU when the DWN_OVER_CURRENT_USB_N signal is asserted (it is active low).
- Make the DWN_OVER_CURRENT_USB_N signal for each port available as a readable register to the firmware.
- Also report in a separate 4-bit register which port(s) tripped the Over Current IRQ.
- This register should be sticky - if multiple ports trip the IRQ prior to the FW being able to read this register, then all of the bits corresponding to ports which have experienced over current events should be high.
- This register should clear upon reading by the FW.
- The reason for this separate set of registers is to capture the condition where an IRQ is tripped but the condition that tripped the IRQ has gone away before the firmware can service the IRQ.

- Use only one interrupt line for all 4 port power controls (IRQ 11). Let FW read which port experienced an overcurrent event.
- One reason for notifying the MCU of an overcurrent event is that in Boston Charger Mode the optimal charging strategy may be to record this overcurrent event and to perhaps choose a different charging profile as a result.

7.3.8.4 Port State Change IRQ / Port State Registers (IRQ 7)

In order to support USB Active Mode Charge Emulation, the firmware must be notified of changes in the port state. The reason is that the timings of USB BCS handshaking protocols are defined in relation to port connect and disconnect events. The easiest way to notify firmware of port connect and disconnect events is to issue an IRQ when port state changes take place.

- Transitions to the following downstream port states for any downstream port must trigger the Port State Change Interrupt (IRQ 7):
    - Not Configured
    - Powered-off
    - Disconnected
    - Any Other Port State
        - Note that "Any Other Port State" includes the states "Disabled", "Resetting", "Enabled", "Transmit", "TransmitR", "Suspended", "Resuming", "SendEOR", "Restart_S", "Restart_E", and "Testing"

- Switching from one of the "Any Other Port States" to another one of the "Any Other Port States" should not trigger an IRQ.
- Switching from "Not Configured", "Powered-off", and "Disconnected" to "Any Other Port State" should trigger an IRQ.

- The digital HW should contain a 4-bit register showing which port(s) triggered the IRQ.
    - This register should be sticky - if multiple ports trip the IRQ prior to the FW being able to read this register, then all of the bits corresponding to ports which have experienced port change events should read logic high.
    - This register should clear upon reading by the FW.
    - The port state of each port (including the specific states "Not Configured", "Powered-off", "Disconnected", "Disabled", "Resetting", "Enabled", "Transmit", "TransmitR", "Suspended", "Resuming", "SendEOR", "Restart_S", "Restart_E", and "Testing") should be readable from a separate register.
- An example usage of this IRQ is shown in Figure 7-49.

7.3.8.5 LPCDWU IRQ (IRQ 9)

- There must be an IRQ associated with the Low-Power Current Detect Wake Up (LPCDWU) Circuits (IRQ 9).
- Note that there is an LPCDWU circuit on each of the 4 downstream ports.
- This IRQ should trip when the ASIC is not in a sleep state and the value of the LPCDWU output for any port transitions from low to high.
    - The reason for this is because if Boston has been woken up from LPCDWU sleep and is currently charging on one port, the FW must be notified if a device is plugged into another port.
    - In the absence of USB operation (which is the case in Charging Mode), the LPCDWU circuit is the only way for Boston to tell if a device has been plugged into one of the upstream ports.
- The digital HW should contain a 4-bit register showing which port(s) triggered the IRQ.
    - This register should be sticky - if multiple ports trip the IRQ prior to the FW being able to read this register, then all of the bits corresponding to ports which have experienced LPCDWU low-to-high transition events should read logic high.
    - This register should clear upon reading by the FW.
- The digital HW should also contain a 4-bit register showing the logical levels of the 4 LPCDWU outputs that is firmware-accessible.

- An example usage of this IRQ is shown in Figure 7-50.

7.3.8.6 Upstream VBUS Detect & Sleep Timer IRQ (IRQ 12)

- There must be an IRQ (IRQ 12) associated with
    1. the VBUS digital input pad circuit (acting as a comparator), which detects the logical polarity of the upstream VBUS signal.
    2. the SLEEP TIMER.
- Note that there is only one VBUS digital input pad circuit.
- This IRQ should trip
    1. when the value of the VBUS digital input pad circuit output changes logical value, either from low-to-high or high-to-low.
    2. Boston core digital is active and the SLEEP TIMER rolls down to 0.
- A certain register INT_ID(0x50013023, **Byte Read ONLY**) can be read to distinguish the VBUS and Sleep Timer. The INT_ID will be cleared after read.
- The reason for requiring an IRQ on VBUS is that its rising transition is a signal to the firmware to exit charging mode and to enter USB mode. Also, the falling transition of this signal is a signal to the firmware to enter the LPCDWU Sleep mode.
- The reason for requiring an IRQ for sleep timer is to relay to the firmware that the end of the Charging Mode interval has occurred and that Boston must enter the "Knockout" state.
- The digital HW should contain a 1-bit register showing the logical level of VBUS digital input pad circuit that is firmware-accessible.
- An example usage of this IRQ is shown in Figure 7-50.

7.3.8.7 500Ω Pull Down Control

- There will exist four outgoing bits for the 500Ω pull-down that is required to be implemented on each of the four VBUSX pins.
- This outgoing signal will be called EN_500OHM_PULL_DOWN[3:0] (active high).
- There will also exist a signal (global) 500OHM_PULL_DOWN_MODE[1:0].
- There will also exist a signal 500OHM_PULL_DOWN_MODE_FW[3:0] (one bit for each port).

The functionality of these signals (per port) is as follows:

Table 7-20 500OHM_PULL_DOWN_MODE[1:0] Functionality Table

500OHM_PULL_DOWN_MODE [1:0]	EN_500OHM_PULL_DOWN==
11	~DWN_DRVVBUS || EN_500OHM_PULL_DOWN_FW
10	EN_500OHM_PULL_DOWN_FW
01	~DWN_DRVVBUS
00	logic 0

7.3.8.8 Downstream PHY disable function

- A certain register (0x50013017, **BYTE WRITE ONLY**) can be used to disable the corresponding downstream PHY.
- This function will mask out all the information from the disabled port to our hub, which would be helpful when we have different implementations.

7.4 Power Management (PMU)

7.4.1 Bandgap Voltage Reference

7.4.1.1 Description/Interface

In order to provide a stable on-chip voltage reference of about 1.2V, a bandgap circuit is incorporated into the power management of the Boston die. Associated with the bandgap will be three arrays of reference current sources: one which supplies PTAT currents, one V2I which supplies currents proportional to the bandgap voltage V_{BG} (alternatively denoted $V_{BANDGAP}$) divided by a high-accuracy resistor, and one V2I which supplies currents proportional to the bandgap voltage V_{BG} divided by a very high-accuracy production-test-trimmed resistor.

The bandgap will have some associated trim memory in the microcontroller die in order to fine-tune its accuracy during production testing. The trimmed V2I will also have associated trim memory.

In order to examine its output voltage during production test, the internal bandgap core voltage must be buffered by a circuit capable of driving both the analog test port bus and the production tester analog measurement load (~200pF). Thus, this buffer circuit must be stable from a 0pF to 200pF load. For this test to be valid, the bandgap voltage will be calibrated at the buffer output and this buffer output used to generate all V_{BG} – derived currents and voltages around the chip. This buffer will not have an external pin with which it may access a large filtering capacitor.

In order to examine the trimmed V2I current during production test, all elements of said V2I current source array (with the exception of the one supplying bias current to the ICOSC) shall be placed in parallel and be connected to the analog test port for calibration measurement. Both the V2I and test

port shall be designed so that the presence of the test port does not interfere with the accuracy of the measurement.

7.4.1.2 Electrical Characteristics

Table 7-21 Target Performance 1.2V Bandgap Reference and Associated V2Is

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V,3.6V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	Bandgap Core.			40	µA
Quiescent Current	Regular V2I Circuit.	11.7	14.8	15.3	µA
Quiescent Current	Trim V2I Circuit.	15.7	19.7	24.5	µA
Quiescent Current	Each Bandgap Buffer (there are 2 on chip)			4.8	µA
Sleep Current	Bandgap Core.			45	nA
Sleep Current	Regular V2I Circuit.		0.6	150	nA
Sleep Current	Trim V2I Circuit.		1.3	274	nA
Sleep Current	Each Bandgap Buffer (there are 2 on chip)			1	nA
Voltage Accuracy	All Conditions and Device Mismatch (Monte Carlo). After Calibration.	1.201	1.213	1.225	V
Regular V2I Absolute Current Accuracy Deviation from Nominal	All Conditions and Device Mismatch (Monte Carlo). After Bandgap Calibration.	-17		+25	%
Regular V2I Current Matching	All Conditions and Device Mismatch (Monte Carlo).			+/- 1.5	%
Regular V2I 2.5uA Current Source Units	Includes dummies.	88			

Table 7-21 Target Performance 1.2V Bandgap Reference and Associated V2Is **Typ:** Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C **Min/Max:** Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V,3.6V, Temp=-40C, +125C					
Name	**Conditions/notes**	**Min**	**Typ**	**Max**	**Unit**
Trim V2I Absolute Current Source Array Accuracy Deviation from Nominal (Calibrated Current Source)	All Conditions and Device Mismatch (Monte Carlo). Not including bandgap voltage variation. This quantity represents the current mismatch error of all devices in the array in parallel (i.e. the effective current source which is subject to calibration) against an absolute reference.			+/- 1.51	%
Trim V2I Current Matching	All Conditions and Device Mismatch (Monte Carlo). This quantity represents the current mismatch of a given device in the current source array to the average current in the current source array.			+/- 1.69	%
Trim V2I 2.5uA Current Source Units	Includes Dummies.	36			
Startup Time	All blocks in series.			0.5	ms
Power Supply Rejection	Up to 600kHz	40			dB
Instantaneous Bandgap Noise Voltage	Can be derived from RMS-integrated noise from 100Hz.			0.2	mV
Buffer External Load Capacitor		0		200	pF

7.4.2 Linear Voltage Regulators (LDOs)

7.4.2.1 Description/Interface

The Boston die must incorporate a number of voltage regulators to supply both core (1.5V) devices and the specific voltages required by the MCU and FLASH.

Each low current regulator (defined as those capable of sourcing 10mA and 15mA) will have an external pin and will be loaded with a 1uF capacitor in parallel with a 100nF capacitor. (Both capacitors +/-20% tolerance).

Each high current regulator (defined as those capable of sourcing 120mA and 200mA) is required to have inrush current limiting to avoid overloading the external 3.3V LDO.

Each high current regulator will have an external pin and will be loaded with a 10uF capacitor in parallel with a 100nF capacitor. (Both capacitors +/-20% tolerance).

Throughout this document, these regulators are also denoted as LDOs (low dropout regulators) because they utilize an LDO architecture, although the dropout voltage realized may not strictly be "low".

7.4.2.2 3.3V to 1.5V Analog LDO Regulator (Low Current – 15mA)

This regulator will supply those sections of the analog circuitry running from the core voltage, namely the PLL and internal USB Analog PHY circuits.

7.4.2.3 3.3V to 1.5V Digital LDO Regulator (High Current – 200mA)

This regulator will supply those sections of the digital circuitry running from the core voltage, namely all IP cores, digital circuitry associated with analog and mixed signal blocks such as the USB CDR and PLL, and microcontroller interface.

7.4.2.4 3.3V to 1.8V MCU LDO Regulator (High Current – 120mA)

This regulator supplies the core of the MCU and the SD Card signaling interface in low-voltage mode. The specification for this block is derived with the following worst-case assumptions:

- The total capacitance seen by each Boston driver in the SD Card 1.8V signaling interface is 30pF as described in Section 6.7.2 of [4].
- All 8 data lines and the command line are toggling at a peak rate of 50MHz in an 8-wire interface. In a 4-wire interface, the 4 data lines and the command line toggle at a peak rate of 100MHz. The latter case is the worst-case condition.
- The clock line is toggling at a rate of 100MHz.
- There exists a pull-up resistance as low as 10kΩ inside the SD Card as described in Section 6.6.5 of [4].
- There exists an additional pull-up resistance as low as 10kΩ inside the Boston SD Card I/O drivers as described in Section 6.6.5 of [4].
- Therefore the maximum average signaling interface current can be calculated as:
    - $6 \cdot ((30pF) \cdot (1.9V) \cdot (100MHz) + 0.5 * (1.9V)/(5kOhms)) = 35mA$

- In order to provide some margin, it is therefore specified that 110mA of current-sourcing capability be allocated for the purpose of supplying the SD Card interface.
- However, 10mA of current-sourcing capability must also be allocated for the microcontroller core, for a total specification current of 120mA.

The current sourced by this block is limited by the SD Card I/O drivers, which are sufficient to protect the IC from physical damage caused by a short circuit of the SD Card clock, command, and data pins to ground. If the current is high enough to cause the voltage to drop low enough so that the MCU does not meet timing requirements at an operational frequency of 30MHz (1.62V), an ASIC reset will be issued.

7.4.2.5 3.3V to 2.5V FLASH LDO Regulator (Low Current – 10mA)

This regulator supplies the MCU FLASH and I/O devices.

7.4.2.6 Electrical Characteristics

Table 7-22 Target Performance Linear Regulators (LDOs)

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V,3.6V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current	3.3V→1.5V Analog LDO.	32	44	59	µA
Quiescent Current	3.3V→1.5V Digital LDO.			50	µA
Quiescent Current	3.3V→1.8V MCU/SD Card Interface LDO.			50	µA
Quiescent Current	3.3V→2.5V FLASH LDO.	29	48.4	53.8	µA
Sleep Current	3.3V→1.5V Analog LDO.			50	nA
Sleep Current	3.3V→1.5V Digital LDO.			63	nA
Sleep Current	3.3V→1.8V MCU/SD Card Interface LDO.			63	nA
Sleep Current	3.3V→2.5V FLASH LDO.			50	nA
Rated Load Current	3.3V→1.5V Analog LDO.			15	mA
Rated Load Current	3.3V→1.5V Digital LDO.			200	mA
Rated Load Current	3.3V→1.8V MCU/SD Card Interface LDO.			120	mA
Rated Load Current	3.3V→2.5V FLASH LDO.			10	mA

Table 7-22 Target Performance Linear Regulators (LDOs)

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V,3.6V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Voltage Accuracy (Deviation from Nominal)	Including deviation of bandgap reference. All Conditions and Device Mismatch (Monte Carlo). All regulators.			+/- 3.3	%
Power Supply Rejection	3.3V→1.5V Analog LDO, All Frequencies.	46	47		dB
Power Supply Rejection	3.3V→1.5V Digital LDO. From 100kHz to 1MHz.	30			dB
Power Supply Rejection	3.3V→1.8V MCU/SD Card Interface LDO. From 100kHz to 1MHz.	35			dB
Power Supply Rejection	3.3V→2.5V FLASH LDO, All Frequencies	38	43		dB
Droop during maximum current step	When current is ramped from 0 to $I_{LOAD,MAX}$ at some ramp rate much faster than the open loop bandwidth. All LDO regulators except 3.3V→1.5V Digital LDO. Not counting HF ringing due to bondwire inductance.			100	mV
Droop during maximum current step	When current is ramped from 0 to $I_{LOAD,MAX}$ at some ramp rate much faster than the open loop bandwidth. 3.3V→1.5V Digital LDO regulator. Not counting HF ringing due to bondwire inductance.			150	mV
Inrush Current	3.3V→1.5V Analog LDO.	35	61	100	mA
Inrush Current	3.3V→1.5V Digital LDO.			40	mA
Inrush Current	3.3V→1.8V MCU/SD Card Interface LDO.			40	mA
Inrush Current	3.3V→2.5V FLASH LDO.	70	121	200	mA
Startup Time	Analog, FLASH LDO Regulators			0.5	ms
Startup Time	Digital, MCU/SD Card LDO Regulators			4	ms

Table 7-22 Target Performance Linear Regulators (LDOs) **Typ:** Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C **Min/Max:** Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.0V,3.6V, Temp=-40C, +125C					
Name	**Conditions/notes**	**Min**	**Typ**	**Max**	**Unit**
MCU LDO Regulator Brown-Out Trip Voltage		1.62		1.68	V

7.5 Peripheral Interfaces

7.5.1 SD Card Interface

Boston will implement a signaling interface compliant with the SD Card Physical Layer Specification Version 4.10 [4] for card versions up to UHS-I. The Boston signaling interface is capable of signaling at both VDDD3P3_SDIN for low-speed communications and at VDDMCU1P8 for high speed communications up to 100MHz (SDR50 and DDR50).

7.5.1.1 Power and I/O Supply Switching Scheme

A depiction of the power and I/O switching scheme for the SD Card interface is shown in Figure 7-48. This figure is analogous to the example power and I/O switching scheme shown in Figure 4-12 in the SD Card Physical Layer Specification and depicts the potential connections and control features of the:

- SD Card Power Switch
- SD Card I/O Supply Switch
- SD Card I/O Bank

7.5.1.1.1 Differences between SD Card Specification Example and Boston

- The 1.8V regulator is not powered by CARD_POWER.
    - This is because this regulator also supplies the MCU digital core circuitry in Boston and should be supplied from a more stable voltage (i.e. one that does not drop to 0V).
- Furthermore, the 3.3V supply to the I/O interface can be supplied by either CARD_POWER or VDDD3P3_SDIN due to concerns over local supply voltage glitching tripping the overcurrent protect mechanism on the CARD_POWER switch.
    - In this case where VDDD3P3_SDIN is used as the supply to the SD Card I/O bank, the SD Card Physical Layer Specification requirement that the I/O peak output voltage is less than CARD_POWER+0.3V is met by the fact that that maximum current draw by the card in 3.3V signaling mode is 200mA and the maximum board resistance (due to ASIC

and ferrite) over corner will be 1.15Ω for a peak voltage difference of 0.23V between the two signaling levels.

- For this specification to be met, additional board routing resistance must therefore be minimized.
- External 33Ω resistors on the SD Card signaling lines will prevent any additional overvoltage peaks.

7.5.1.1.2 Control Features

- FW/HW Control
  - Each switch in the SD Card Interface can be placed under either firmware (via analog trim memory) or hardware (via the SD Card host) control.
  - This allows the Power and I/O Supply switches to maintain a well-defined state during reset, power-up, and sleep.
  - This also facilitates test and automated debugging.
  - By default:
    - The switches are under firmware control.
    - The power switch is opened.
    - The 1.8V/3.3V I/O supply switch has selected 3.3V.
    - The VDDD3P3_SDIN/CARD_POWER supply switch has selected VDDD3P3_SDIN.
- Automatic grounding of I/O Supply when Power Switch is opened.
  - This guarantees that I/O output voltages go to 0V when Power Switch is opened as required by the SD Card Physical Layer Specification.
  - This feature is not enabled by default and should not be needed since review of the SD Card Host Digital IP shows that the I/O outputs are forced low when the Power Switch is opened.

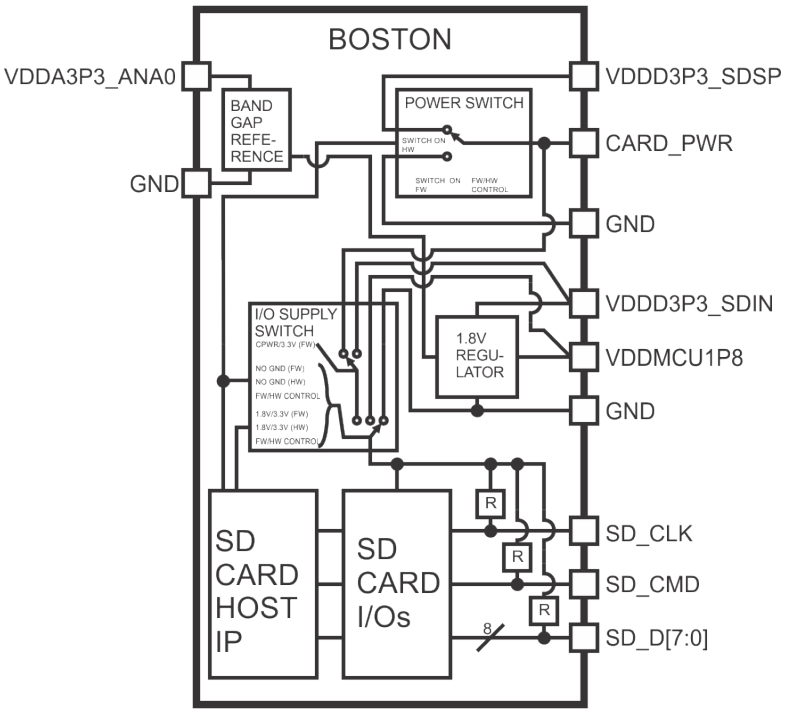


Figure 7-48 Boston SD Card Interface Power and I/O Supply Switching

7.5.1.2 Power Switch

7.5.1.2.1 Description/Interface

The SD Card power switch is a very-low impedance single-pole double throw implementation between VDDD3P3_SDSP and CARD_POWER with current limiting. The power switch is designed and laid out to support the 400mA current draw required by UHS-I-compatible SD Cards.

The current limiting circuitry is internally bootstrapped for rapid (~100ns) response during hot card insertion events. Although the power switch is low impedance (with an estimated maximum 1.3Ω total series routing resistance on the ASIC, ferrite, and board), the SD Card minimum CARD_POWER voltage requirement of 2.7V (see Table 6-2 of [4]) cannot be met for 400mA load current unless a 3.3V regulator with an accuracy of less than 2% deviation from nominal voltage is utilized.

The SD Card power switch current limiting is implemented by switching the pass device gate terminal connection between ground (for the lowest possible impedance) and a current mirror (for constant current output). The trip point between the two modes corresponds to a voltage across the pass device that is equal to the voltage across the pass device with a grounded gate with a reference current flowing through it. One way to think about this is that if a voltage source were to be placed on CARD_POWER and its voltage value lowered starting from the voltage on VDDD3P3_SDSP, the

current through the switch would increase until some trip point is reached, after which point the output current will be some (potentially) much lower value that is relatively invariant with any further changes in output voltage. The value of this output current when CARD_POWER is at 0V is termed the short-circuit current. The value of this current when the voltage at CARD_POWER is equal to the trip point voltage is roughly half of the short-circuit current. The trip point voltage is programmable such that the maximum current through the switch is slightly larger than 100mA, 200mA, 400mA, which represent the maximum current specifications of SD Card types UHS-I and earlier (see Section 6.3.3 of [4]).

The CARD_POWER connection to ground is current-limited by the resistance of the switch, which is approximately 50Ω. This results in a discharge of a 10µF capacitor (5µF of board and 5µF of SD Card capacitance) to below 0.5V in 1ms.

7.5.1.2.2 Coarse Requirements

- The power switch will implement the following features:
- Programmable current trip point @ 151mA, 255mA, 470mA nominal current.
- Programmable short circuit current @ 50mA, 100mA, 200mA, 400mA nominal current.
- Total VDDD3P3_SDSP-CARD_PWR voltage drop of 0.42V at 400mA load current, corresponding to a resistance of 1.05Ω.
- Firmware override for test and debug purposes.
- Rapid-response current limiting to protect the ASIC in the event of a SD Card hot insertion.

7.5.1.2.3 Electrical Characteristics

Table 7-23 Target Performance SD Card Power Switch

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.234V-3.366V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Quiescent Current			50	60	µA
Leakage Current			10	400	nA
Current Spike Peak	Upon hot insertion of SD Card with 10µF supply capacitor, for 470mA trip point setting.			700	mA
Current Spike Duration	Time during which spike is above trip point, for 470mA trip point setting.			150	ns

Table 7-23 Target Performance SD Card Power Switch

Typ: Simulated under process corner: TM, $V_{3.3V}$=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, $V_{3.3V}$=3.234V-3.366V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
Trip Point Accuracy (Deviation from Nominal)	All trip point settings.	-50		+50	mA
Switch On Resistance	Between VDDD3P3_SDSP and CARD_PWR pins, at 400mA load current.			1.05	Ω
SD Card Charging Time	50mA short circuit current limit setting, 10 µF SD Card capacitor.		0.8	1	ms
Short-Circuit Current Accuracy (Deviation from Nominal)	This accuracy only valid when V(CARD_POWER)=0V. Current-limited current value decreases as V(CARD_POWER) increases.	-20		20	%

7.5.1.3 I/O Supply Switch (Regulator Selector)

The I/O supply switch must pass more than 400mA of instantaneous current during simultaneous switching events. In the case of 1.8V signaling, the 1.8V regulator will be set at a nominal voltage of 1.85V with an accuracy of less than +/-3% deviation from nominal voltage, with a minimum of 1.7945V. The SD Card specification states that the high logic level of the pad shall be above 1.4V (see Table 6-13 of [4]). The Boston design will set a target of 1.6V. In this case, the total routing resistance including the regulator selector switch must be less than 0.486Ω.

7.5.1.4 SD Card I/Os (Signaling Interface)

The SD Card signaling interface will be a set of GPIOs whose supplies are tied to the regulator selector. The GPIOs will be consistent with SD Card 1.8V Signaling Driver Type B performance (Section 6.7.1.2 of [4]), capable of driving 30pF for UHS50 card with a rise/fall time greater than 0.7ns and less than 2.75ns as per Table 6-8 of [4]. GPIO input leakage current with pull-ups disconnected will have a magnitude less than 2µA as per Table 6-14 of [4].

Table 7-24 Target Performance SD Card I/Os (Signaling Interface)

Typ: Simulated under process corner: TM, V_{CARD_POWER}=3.3V, Temp=27C

Min/Max: Simulated under process corners: TM, WS, WP, WO, WZ, V_{CARD_POWER}=2.7V-3.6V, Temp=-40C, +125C

Name	Conditions/notes	Min	Typ	Max	Unit
GPIO Leakage Current	3.3V Signaling Mode	-10		10	µA
Clock Rise/Fall Time	3.3V Signaling Mode, 40pF Load Capacitance. Between 0.25*V_{CARD_POWER} and 0.625*V_{CARD_POWER}.			3	ns
GPIO Leakage Current	1.8V Signaling Mode, Per Pin	-2		2	µA
Clock Rise/Fall Time	1.8V Signaling Mode, 30pF Load Capacitance, between 0.58V and 1.27V.			2	ns
CMD, Data Rise/Fall Time	1.8V Signaling Mode, 30pF Load Capacitance, between 0.45V and 1.4V.	0.7	1.83	2.75	ns

7.5.2 SPI/I2C Interface

There will exist an interface between the microcontroller and the pinout to permit bidirectional SPI and I2C communications. The digital hardware inside Boston permits Boston to behave as either an SPI Master or an I2C Master, but not both concurrently. The I2C interface will not include on-board pull-up resistors; these resistors must be placed externally.

7.5.3 Test Interfaces

7.5.3.1 Digital Test Port

There will exist a digital test port that will stream out the serial conversion-by-conversion results of the two on-board ADCs. The interface mapping is defined as follows:

ADCCLK → SD_CMD pin

HKADC Serial Data → SD_D0 pin

HKADC STRBO → SD_D1 pin

Charge Measurement ADC Serial Data → SD_D2 pin

Charge Measurement ADC STRBO → SD_D3 pin

Enabling the digital test port will override SD Card operation.

7.5.3.2 Analog Test Port

There will exist an analog test port that will allow the probing of internal voltage and current nodes within the ASIC for debugging and calibration purposes. This port will multiplex with the two SPI pins

that are not shared with the I2C interface. Operating this analog test port concurrently with the SPI interface will produce unintended results for both SPI and analog test port operation.

7.6 Clough Cortex M0 ARM Microcontroller

The Cortex M0 ARM microcontroller provides a flexible and programmable option for managing data flow from the Boston USB Device Endpoint Buffers to the SDIO-HOST. The programmable FLASH memory may also be used to configure the Boston ASIC.

AyDeeKay's Cortex M0 core (CM0160K8) implements an ARM Cortex-M0 low cost 32-bit microcontroller containing:

ARM Architecture:

- ARM Cortex-M0 processor
- System Tick Timer (SysTick – 24 bits, interruptible)
- Serial Wire Debugger (2 pins)
- Built-in Nested Vectored Interrupt Controller (NVIC)
- Four Breakpoints and two Watchpoints

AyDeeKay Architecture:

- Programmable Watch-Dog Timer (Trigger time: 2^{13}, 2^{19} , 2^{22} , and 2^{32} clock cycles)
- Three General Purpose 32-bit Timers (Timer0, Timer1 and Timer2)

Memory:

- 160KB of Flash Program Memory
- 8KB of SRAM
- Self Programming

In this document, the characteristics that are specific to AyDeeKay LLC Cortex M0 implementation will be discussed. For information regarding the Cortex M0 core please refer to:

http://www.arm.com/products/processors/cortex-m/cortex-m0.php

7.6.1.1 Memory Description and Map

ARM Cortex M0 uses a unified memory model with a linear address space (Von Neumann architecture) including Flash and RAM memories as well as registers address space.

AyDeeKay LLC implementation of the Cortex M0 core (CM0160K8) uses a high density 160KB Flash cell along with 8KB of SRAM.

The following table defines the several regions of the address space:

Table 7-25 Cortex M0 Core Memory Map

Address	Memory	Description	References
0x00000000 - 0x00027FFF	Flash	160Kbytes of Flash Memory, user programmable.	
0x00028000 - 0x0003FFFF	N/A	Reserved	
0x00040000 - 0x000400FF	N/A	Reserved	
0x00040100 - 0x000401FF	N/A	Reserved	
0x00040200 - 0x1FFFFFFF	N/A	Reserved	
0x20000000 - 0x20001FFF	SRAM	8Kbytes of SRAM	
0x20002000 - 0x4FFFFFFF	N/A	Reserved	
0x50000000 - 0x5000007F	Peripherals	(Implementation Dependent)	
0x50000080 - 0x50000085	N/A	Reserved	
0x50000086 - 0x5000FFFF	N/A	Reserved	
0x50010000 - 0x5001FFFF	Peripherals		
0x50020000 - 0x5002001F	Peripherals	General Purpose Timers Controls	
0x50020020 - 0x50020047	Flash	Flash Programming/Erase Control	
0x50020048 - 0xDFFFFFFF	N/A	Reserved	
0xE0000000 - 0xE00FFFFF	Private peripheral bus	ARM peripherals	
0xE0100000 - 0xEFFFFFFF	N/A	Reserved	
0xF0000000 - 0xF0001FFF	System ROM tables	ARM core IDs	

0xF0002000 - 0xFFFFFFFF	N/A	Reserved	

7.6.1.2 RAM

CM0160K8 implements 8Kbytes of SRAM from addresses 0x20000000 - 0x20001FFF. One useful feature of the Cortex architecture is that it can execute code from the RAM memory.

7.6.1.3 User Programmable Flash Memory

CM0160K8 implements a Programmable Flash Memory from address 0x00000 to 0x27FFF (0 to 163839 decimal). It is organized in bytes, sectors (256 bytes) and blocks (2 Kbytes).

The first 148 bytes of Flash Memory are organized following the standard created by ARM. In this standard the Address 0x00000 contains the top-of-stack address (four bytes). The following addresses contain interrupt vectors used by the microcontroller:

Table 7-26 Interrupt Vector

Vector Name	Address	Comments
STACK_VALUE	0x00000	Typically set to 0x20001FFFF (Top of SRAM)
Reset_Handler	0x00004	
Reserved	0x00008	
HardFault Handler	0x0000C	
Reserved	0x00010 to 0x00028	
SVC_Handler	0x0002C	
Reserved	0x00030-0x00034	
PendSV_Handler	0x00038	
SysTick_Handler	0x0003C	
Dependent on product implementation	0x00040 to 0x0007C	
Timer0_Handler	0x00080	
Timer1_Handler	0x00084	
Timer2_Handler	0x00088	
Watchdog_Handler	0x0008C	
BTE_Handler	0x00090	
Reserved	0x00094	

All other addresses in the flash memory can be used for the user's program.

The meanings of the standard interrupt vectors (Provided with the ARM Cortex M0 core) are defined in ARM's documentation. One of the sources of information is:

http://infocenter.arm.com/help/topic/com.arm.doc.dui0497a/DUI0497A_cortex_m0_r0p0_generic_ug.pdf

7.6.1.4 Flash Memory Registers

The following registers are used to control the access to the Flash memory:

Note: These registers must be accessed in 32-bit transfers.

The general format of the description of the registers in this section is as follows:

Name of the Register		Address (Hex)			Reset Value (Hex)		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Bit_Name	Bit_Name	Bit_Name	Bit_Name	Bit_Name	Bit_Name	Bit_Name	Bit_Name

Msb Lsb

FLADDR: Flash byte, sector and block address used to write and erase.

FLADDR		0x50020020			0x00000000		
W	W	W	W	W	W	W	W
FLADD7	FLADD6	FLADD5	FLADD4	FLADD3	FLADD2	FLADD1	FLADD0
FLADD15	FLADD14	FLADD13	FLADD12	FLADD11	FLADD10	FLADD9	FLADD8
FLADD23	FLADD22	FLADD21	FLADD20	FLADD19	FLADD18	FLADD17	FLADD16
FLADD31	FLADD30	FLADD29	FLADD28	FLADD27	FLADD26	FLADD25	FLADD24

Bit31-0 **FLADD31: FLADD0:** Address in Flash of a byte, sector or block

FLWRDT: Flash data byte.

FLWRDT		0x50020024			0x00000000		
W	W	W	W	W	W	W	W
FLDT7	FLDT6	FLDT5	FLDT4	FLDT3	FLDT2	FLDT1	FLDT0
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Bit7-0 **FLDT7: FLDT0:** Data byte to be written into Flash Memory.

Bit31-8 Ignored.

UNLBWR: Flash data byte-unlock register.

UNLBWR		0x50020028			0x00000000		
W	W	W	W	W	W	W	W
UBW7	UBW6	UBW5	UBW4	UBW3	UBW2	UBW1	UBW0
UBW15	UBW14	UBW13	UBW12	UBW11	UBW10	UBW9	UBW8
UBW23	UBW22	UBW21	UBW20	UBW19	UBW18	UBW17	UBW16
UBW31	UBW30	UBW29	UBW28	UBW27	UBW26	UBW25	UBW24

Bit31-0 **UBW31: UBW8:** Pattern used to unlock Flash Memory for byte write. The pattern used must be:

0x55555555

BWRSTRT: Flash byte-write start register.

BWRSTRT		0x5002002C			0x00000000		
W	W	W	W	W	W	W	W
BRST7	BRST6	BRST5	BRST4	BRST3	BRST2	BRST1	BRST0
BRST15	BRST14	BRST13	BRST12	BRST11	BRST10	BRST9	BRST8
BRST23	BRST22	BRST21	BRST20	BRST19	BRST18	BRST17	BRST16
BRST31	BRST30	BRST29	BRST28	BRST27	BRST26	BRST25	BRST24

Bit31-0 **BRST31: BRST0:** Pattern used to start Flash Memory byte write. The pattern used must be:

0xAAAAAAAA

UNLSER: Flash sector erase unlock register.

UNLSER		0x50020030			0x00000000		
W	W	W	W	W	W	W	W
USE7	USE6	USE5	USE4	USE3	USE2	USE1	USE0
USE15	USE14	USE13	USE12	USE11	USE10	USE9	USE8
USE23	USE22	USE21	USE20	USE19	USE18	USE17	USE16
USE31	USE30	USE29	USE28	USE27	USE26	USE25	USE24

Bit31-0 **USE31: USE0:** Pattern used to unlock Flash Memory for sector erase. The pattern used must be:

0x66666666

SERSTRT: Flash sector erase start register.

SERSTRT		0x50020034			0x00000000		
W	W	W	W	W	W	W	W
SES7	SES6	SES5	SES4	SES3	SES2	SES1	SES0
SES15	SES14	SES13	SES12	SES11	SES10	SES9	SES8
SES23	SES22	SES21	SES20	SES19	SES18	SES17	SES16
SES31	SES30	SES29	SES28	SES27	SES26	SES25	SES24

Bit31-0 **SES31: SES0:** Pattern used to start Flash Memory sector erase. The pattern must be:

0x99999999

UNLBLKER: Flash block erase unlock register.

UNLBLKER		0x50020038			0x00000000		
W	W	W	W	W	W	W	W
UBKE7	UBKE6	UBKE5	UBKE4	UBKE3	UBKE2	UBKE1	UBKE0
UBKE15	UBKE14	UBKE13	UBKE12	UBKE11	UBKE10	UBKE9	UBKE8
UBKE23	UBKE22	UBKE21	UBKE20	UBKE19	UBKE18	UBKE17	UBKE16
UBKE31	UBKE30	UBKE29	UBKE28	UBKE27	UBKE26	UBKE25	UBKE24

Bit31-0 **UBKE31: UBKE0:** Pattern used to unlock Flash Memory for block erase. The pattern used must be:

0x77777777

BKERSTRT: Flash block erase start register.

BKERSTRT		0x5002003C			0x00000000		
W	W	W	W	W	W	W	W
BES7	BES6	BES5	BES4	BES3	BES2	BES1	BES0
BES15	BES14	BES13	BES12	BES11	BES10	BES9	BES8
BES23	BES22	BES21	BES20	BES19	BES18	BES17	BES16
BES31	BES30	BES29	BES28	BES27	BES26	BES25	BES24

Bit31-0 **BES31: BES0:** Pattern used to start Flash Memory block erase. The pattern must be:

0x88888888

FLSCTRL: Flash control register.

FLSCTRL		0x50020040			0x00000000		
R	R	R/W	R/W	R/W	R/W	R/W	R/W
SECER	PRGFLG	WECD2	WECD1	WECD0	RWC2	RWC1	RWC0
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	R
-	-	-	-	-	-	-	BLKER
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
-	-	-	-	-	-	-	-
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved
-	-	-	-	-	-	-	-

Bit2-0 **RWC2: RWC0:** Read wait cycles. (Number of wait states used in the reading process)

Bit5-3 **WECD2: RWC0:** Write/erase clock frequency divider setting.

000 = Clock/1

001 = Clock/2

000 = Clock/3

001 = Clock/4

000 = Clock/5

001 = Clock/6

000 = Clock/7

001 = Clock/8

Bit6 **PRGFLG:** Program flag.

0 = Flash available

1 = Flash being programmed

Bit7 **SECER:** Sector being erased flag.

0 = Flash available

1 = Flash sector being erased

Bit8 **BLKER:** Block being erased flag.

0 = Flash available

1 = Flash block being erased

FLSCP: Flash code protection register.

FLSCP		0x50020044			0x00000000		
W	W	W	W	W	W	W	W
FCP7	FCP6	FCP5	FCP4	FCP3	FCP2	FCP1	FCP0
FCP15	FCP14	FCP13	FCP12	FCP11	FCP10	FCP9	FCP8
FCP23	FCP22	FCP21	FCP20	FCP19	FCP18	FCP17	FCP16
FCP31	FCP30	FCP29	FCP28	FCP27	FCP26	FCP25	FCP24

Bit31-0 **FCP31: FCP0:** Flash Code Protection pattern:

0xF2E11047

NOTE1: This register does not lock the Flash Memory against read/write/erase by the applications program. Instead what it does is to disable all communications with the debug interface, therefore preventing any external attack. The application code is still able to modify the Flash content.

NOTE2: Upon Power-On Reset or Normal Reset CM0160K8 disables the communication for a small time interval (8192 clock cycles). If the application needs to be protected it is mandatory to set this register with the appropriate code in the beginning of the initialization process and before the internal hardware enable the debug communication. To allow for debug communication the application has only to write a different value in the register.

7.6.1.5 Flash Memory Erase, Write and Protect

In normal operation the ARM core fetches instructions (or data permanently stored) from the Flash memory but it is also possible for a program to alter the content of the flash memory. The following operations can be performed in the Flash Memory:

- Byte Write
- Sector Erase
- Block Erase
- Code Protect

These options will be detailed in the next subsections.

7.6.1.5.1 Block Erase

CM0160K8 memory is divided into 2KB blocks. All except the last one can be erased on a per block basis. The last block has 1792 bytes (0x27800 to 0x27EFF) available as the last sector is reserved. To erase a block the following sequence must be followed:

1. Point to an address inside the block to be erased by loading the **FLADDR** register.
2. Unlock the block for erasure by loading the 0x77777777 pattern into the **UNLBLKER** register.
3. Start the block erase process by loading the 0x88888888 pattern into the **BKERSTRT** register.

Code Example:

```
*FLADDR   = 0x000050A0;       //Point to the block starting at 0x00005000
*UNLBLKER = 0x77777777;       //Unlock block erase
*BKERSTRT = 0x88888888;       //Start erase process
while ( *FLSCTRL & 0x100 );   //Check the Block Erase Flag, wait until done
```

NOTE: The erase process of a block or sector can take up to 10msec and it stalls the execution of code from Flash Memory. If necessary while erasing the microcontroller can execute code from RAM.

7.6.1.5.2 Erase Sector

Flash memory is divided into sectors of 256 bytes. All except the last one can be erased using the following standard procedure.

To erase a sector the following sequence must be followed:

1. Point to an address inside the sector to be erased by loading the **FLADDR** register.
2. Unlock the sector for erasure by loading the 0x66666666 pattern into the **UNLSER** register.
3. Start the sector erase process by loading the 0x99999999 pattern into the **SERSTRT** register.

Code Example:

```
*FLADDR = 0x000050A0;         //Point to the block starting at 0x00005000
*UNLSER = 0x66666666;         //Unlock sector erase
*SERSTRT = 0x99999999;        //Start erase process
while ( *FLSCTRL & 0x80 );    //Check the Sector Erase Flag, wait until done
```

NOTE: The erase process of a block or sector can take up to 10msec.

7.6.1.5.3 Write Byte

To write a byte the following sequence must be followed:

4. Point to an address of the byte to be written by loading the **FLADDR** register.
5. Unlock the byte write by loading the 0x55555555 pattern into the **UNLBWR** register.
6. Start the writing process by loading the 0xAAAAAAAA pattern into the **BWRSTRT** register.
7. Load the value to be written into the **FLWRDT** register. (Bits [31…8] are ignored)

Code Example:

```
*FLADDR  = 0x000050BB;        //Point to the byte address
*UNLBWR  = 0x55555555;        //Unlock byte write
*BWRSTRT = 0xAAAAAAAA;        //Start write process
*FLWRDT  = 0x000000AB;        //Load 0xAB to be written
while ( *FLSCTRL & 0x40 );    //Check the Byte Write Flag, wait until done
```

NOTE: The byte write process can take up to 20µsec. During the writing process the microcontroller is stalled.

7.6.1.5.4 Code Protect

CM0160K8 implements a protection scheme to avoid eavesdropping and tampering. It works as follows:

1. After any reset (Power-On or otherwise) the internal HW prevents any communication between the CPU and the debug interface during the first 8192 clock cycles.
2. If during this interval the FLSCP register is loaded with a specific pattern (0xF2E11047) then the communication remains disabled after the end of this interval and stays disabled until this register is loaded with a different pattern.
3. After the first 8192 clock cycles it is still possible to disable the communications between the CPU and the debug interface at any moment using the same procedure previously defined but the CPU will be susceptible to an attack between the end of the 8192 clock cycles and the loading of the FLSCP register.

Note: If a part is protected the emulator can still erase and program the part, but first it will be required to erase the Flash content, therefore protecting it.

Code Example:

```
*FLSCP = 0xF2E11047;          //Code protection activated
```

NOTE: If code protection is necessary it is important to consider the time it takes from the moment the part is reset until the execution reaches the main function. In some exceptional cases where the initialization of the system (stack, heap, global variables) take more than 8K clock cycles it may be necessary to add code directly in the initialization routine.

7.6.1.6 Interrupt Enabling/Disabling Process

Cortex-M0 implements a NVIC (**N**ested **V**ector **I**nterrupt **C**ontroller) peripheral capable of handling up to 16 AyDeeKay LLC peripheral's interrupts. Upon reset the microcontroller can answer only to Reset, NMI (Non-Maskable Interrupt) and Hard Fault interrupts/exceptions. All other interrupts must be enabled.

To enable and disable the interrupts the user must use access the ISER (Interrupt Set Enable Register) and ICER (Interrupt Clear Enable Register) registers associated with the desired interrupt. In the CM0160K8 it can be done by invoking two functions:

```
static __INLINE void NVIC_EnableIRQ( IRQn_Type IRQn )
```

```
static __INLINE void NVIC_DisableIRQ(IRQn_Type IRQn)
```

The IRQn parameter defines the interrupt to be enabled or disabled.

NOTE: both inline functions and all parameters are defined in the product_file.h file, which must be included in the source files. Besides that the product_file.h file contains a list of available interrupts. The format of this list is as follows:

```
typedef enum IRQn
{
  //******  Cortex-M0 Processor Exceptions Numbers ****************************
  NonMaskableInt_IRQn         = -14,    // Non Maskable Interrupt
  HardFault_IRQn              = -13,    // Hard Fault Interrupt
  SVCall_IRQn                 = -5,     // SV Call Interrupt
  PendSV_IRQn                 = -2,     // Pend SV Interrupt
  SysTick_IRQn                = -1,     // System Tick Interrupt

  //*****  CM0IKMCU Cortex-M0 specific Interrupt Numbers **********************
  IRQ00_IRQn                  = 0,      // Product specific
  IRQ01_IRQn                  = 1,      // Product specific
  IRQ02_IRQn                  = 2,      // Product specific
  IRQ03_IRQn                  = 3,      // Product specific
  IRQ04_IRQn                  = 4,      // Product specific
  IRQ05_IRQn                  = 5,      // Product Specific
  IRQ06_IRQn                  = 6,      // Product Specific
  IRQ07_IRQn                  = 7,      // Product Specific
  IRQ08_IRQn                  = 8,      // Product Specific
  IRQ09_IRQn                  = 9,      // Product Specific
  IRQ10_IRQn                  = 10,     // Product Specific
  IRQ11_IRQn                  = 11,     // Product Specific
  IRQ12_IRQn                  = 12,     // Product Specific
  IRQ13_IRQn                  = 13,     // Product Specific
  IRQ14_IRQn                  = 14,     // Product Specific
  IRQ15_IRQn                  = 15,     // Product Specific
  TIMER0_IRQn                 = 16,     // Timer 0
  TIMER1_IRQn                 = 17,     // Timer 1
  TIMER2_IRQn                 = 18,     // Timer 2
  WATCHDOG_IRQn               = 19      // Watchdog timer
} IRQn_Type;
```

Code Example: The following code snippet enables the Timer2 interrupt:

```
#include "boston.h"

void main(void)
{
  //Code that execute other initializations…

  TMR_Config( 2, TMERON, 0xFFC00000);     //Initialize Timer2
  NVIC_EnableIRQ(TIMER2_IRQn );


  //And more code…

}
```

Code Example for the Interrupt Handler: The following code snippet processes the Timer2 interrupt.

Note: This function is originally defined as a “weak” function to be overridden by the application code function. For more on this topic please refer to: http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0205g/CIACBCDA.html

```
void Timer2_Handler( void )
{
  *TMR2REG = 0xFFC00000;      //Reload timer2 register
  //Process application code from this point
}
```

7.6.1.7 Peripherals

The CM0160K8 implements a group of peripherals:

- SysTick Clock
- Timer0
- Timer1
- Timer2
- Watch Dog Timer

The following chapters will describe the peripherals and related interrupt handlers.

7.6.1.8 Systick Timer

This timer is an optional peripheral created by ARM and implemented in the Cortex M0 160/8. It is fully described in the Cortex-M0 Devices Generic User Guide (Chapter 4.4 Optional System Timer, Systick) found at:

http://infocenter.arm.com/help/index.jsp?topic=/com.arm.doc.dui0497a/Babieigh.html

Its interrupt can be enabled as follows:

```
NVIC_EnableIRQ(SysTick_IRQn);
```

7.6.1.9 Timern (0, 1 and 2)

The CM0160K8 implements three identical timers: Timer0, Timer1 and Timer2. All three timers operate using the system clock as clock source. They increment at the system clock rate until they roll over from 0xFFFFFFFF to 0x00000000. At this point an interrupt is generated if enabled.

7.6.1.9.1 Timern Registers

The following registers are associated with Timer0, Timer1 and Timer2:

TMR*n*REG: Timer n counter register. (32-bit, n=0,1,2)

TMRnREG		0x50020000/8/10			0x00000000		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
T0REG7	T0REG6	T0REG5	T0REG4	T0REG3	T0REG2	T0REG1	T0REG0
T0REG15	T0REG14	T0REG13	T0REG12	T0REG11	T0REG10	T0REG9	T0REG8
T0REG23	T0REG22	T0REG21	T0REG20	T0REG19	T0REG18	T0REG17	T0REG16
T0REG31	T0REG30	T0REG29	T0REG28	T0REG27	T0REG26	T0REG25	T0REG24

Bit31-0 **T0REG31: T0REG0**: Initial counter value. The timer will count from this value to 0xFFFFFFFF and roll over to 0x00000000. At this point it will generate an interrupt if enabled. The interrupt routine is responsible for reloading the value if needed as this timer does not auto-reload the original content.

Example Code: Loading the value of Timer1:

```
*TMR1REG = 0xABCDEF00;
```

TMR*n*CTRL: Timer n control register. (32-bit, n=0,1,2)

TMRnCTRL		0x50020004/C/14			0x00000000		
Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	Reserved	R/W
-	-	-	-	-	-	-	TnSTART
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Bit0 **T*n*START**: This bit starts/stops the timer.

1 = Timer Running

0 = Timer Inactive

7.6.1.9.2 Timern Operation

The operation of the Timers is quite straightforward. Load the TMRnREG with the required initial value, enable the timer and if required also enable the related interrupt. If the interrupt is enabled an additional code to handle the interrupt must be added.

Code Example1: If we are running from a 15MHz System Clock and we want to use timer1 to generate a time delay of 2msec without interrupt we can calculate the necessary timer counter:

$$T = 2m\sec \times 15MHz = 30000$$

Given that the timer counts to roll from 0xFFFFFFFF to 0x00000000 we have to take the negative value. Code fragment:

```
  *TMR1REG  = -30000;         //Load counting value
  *TMR1CTRL = 1;              //Enable timer
  while( TMR1REG < 0 );       //Wait to reach '0'
  *TMR1CTRL = 0;              //Disable timer
```

Using the interrupt would require to have it enabled and the interrupt handler defined. The following code generates an interrupt every 2msec under the same conditions:

```
  *TMR1REG  = -30000;         //Load counting value
  *TMR1CTRL = 1;              //Enable timer
  NVIC_EnableIRQ(TIMER1_IRQn);//Enable interrupt

void Timer2_Handler (void)    //Interrupt Handler
{
    TMR1REG  = -30000;        //Load counting value (-30000 = 0xFFFF8AD0)


      //From this point the function goes to process other tasks //required for
      //this timer.
}
```

7.6.1.10 Watchdog

CM0160K8 implements a WDT (**W**atch **D**og **T**imer) that can operate in one of two basic ways:

- Interrupt Mode: In the event of a WDT rollover an interrupt will be executed.
- Reset Mode: In the event of a WDT rollover the microcontroller will reset.

7.6.1.10.1 WDT Registers

The Watch Dog Timer implements two 32-bit registers:

WDTCTRL: WDT (**W**atch **D**og **T**imer) Control Register. (32-bit)

WDTCTRL		0x50020018			0x0000000x		
Reserved	Reserved	Reserved	R/W	R/W	R/W	R/W	R/W
-	-	-	WDTPRES1	WDTPRES0	RSTFLAG	RESETEN	WDTEN
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-
-	-	-	-	-	-	-	-

Bit0 **WDTEN**: WDT enable. This bit can be asserted but it cannot be de-asserted. It means that once the WDT is enabled it cannot be turned off until a Reset or Power-On Reset occurs.

Bit1 **RESETEN**: Reset enable. If enabled a WDT time-out will force the microcontroller to reset. This bit can be asserted but it cannot be de-asserted.

Bit2 **RSTFLAG**: Reset Flag. This flag is set by the system at the initialization if the initialization was caused by a reset triggered by the WDT. The bit can be cleared by the application.

Bit4-3 **WDTPRES1: WDTPRES0**: WDT Prescaler:

00 = 2^{13}/SystemClock

01 = 2^{19}/SystemClock

10 = 2^{22}/SystemClock

11 = 2^{32}/SystemClock

It means that the WDT internal counter will count from 0 to the prescaler value at the system clock speed and trigger if not cleared. For instance, a system running from a 30MHz Crystal with

WDTPRES[1…0] = 10 will trigger the WDT after approximately 0.14 seconds if not cleared properly and in time by the application.

WDTCLR: WDT Clear Register. (32-bit)

WDTCLR		0x5002001C			0x0000000x		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
WCLR7	WCLR6	WCLR5	WCLR4	WCLR3	WCLR2	WCLR1	WCLR0
WCLR15	WCLR14	WCLR13	WCLR12	WCLR11	WCLR10	WCLR9	WCLR8
WCLR23	WCLR22	WCLR21	WCLR20	WCLR19	WCLR18	WCLR17	WCLR16
WCLR31	WCLR30	WCLR29	WCLR28	WCLR27	WCLR26	WCLR25	WCLR24

Bit31-0**WCLRn**: Clear Register. To clear the WDT counting the following words must be written in this order and without any other instruction between then:

0x3C570001

0x007F4AD6

Warning: Programming WDTCLR with other values or in the wrong order will cause the watchdog to throw an interrupt or reset the system if enabled. The following code fragments show how to enable the WDT, how to enable it's interrupt handler and also shows the sequence to reset it's counter:

```
      *WDTCTRL = 0x11;                    //WDT enabled and Twdt = 2^22/SystemClock
      NVIC_EnableIRQ(WATCHDOG_IRQn);      //Enable interrupt

      *WDTCLR = 0x3C570001                //Sequence that resets the WDT
      *WDTCLR = 0x007F4AD6

void Watchdog_Handler( void )
{
      //Here the application executes the code that handles the WDT overflow
}
```

7.7 Recommended Firmware Operation

Shown below in Figure 7-49 and Figure 7-50 are recommended strategies for utilizing the various interrupt sources on the ASIC along with their readback registers in order to execute desired tasks without continuous microcontroller intervention. For example, Figure 7-49 shows a state transition diagram that permits the ASIC to wait indefinitely for a CDP-compliant device to attach to the USB port during USB operation without requiring the microcontroller to continuously poll this port. This allows

high-speed SD Card traffic to take place on other ports while the system is waiting for a CDP-compliant device to connect.

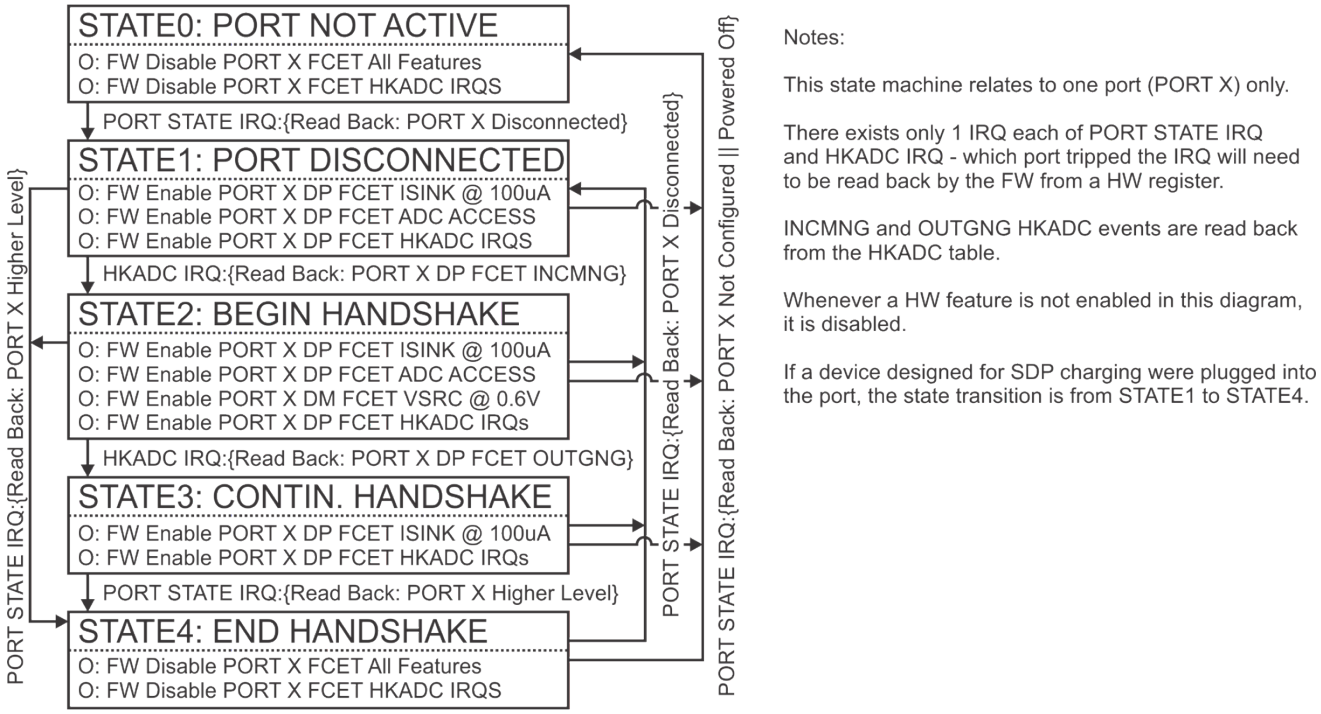


Figure 7-49 Proposed Firmware State Diagram (Per Port) for CDP Handshaking During USB Active Mode

Shown in Figure 7-50 is a recommended procedure for operating the flexible charging emulator termination during Charging Mode. In this case, the use of interrupts and TIMER0 is important to avoid the use of FW-defined counters while waiting for long-term events such as port power control (PPC) port clearing (which can take place on the order of 10ms-1000ms).

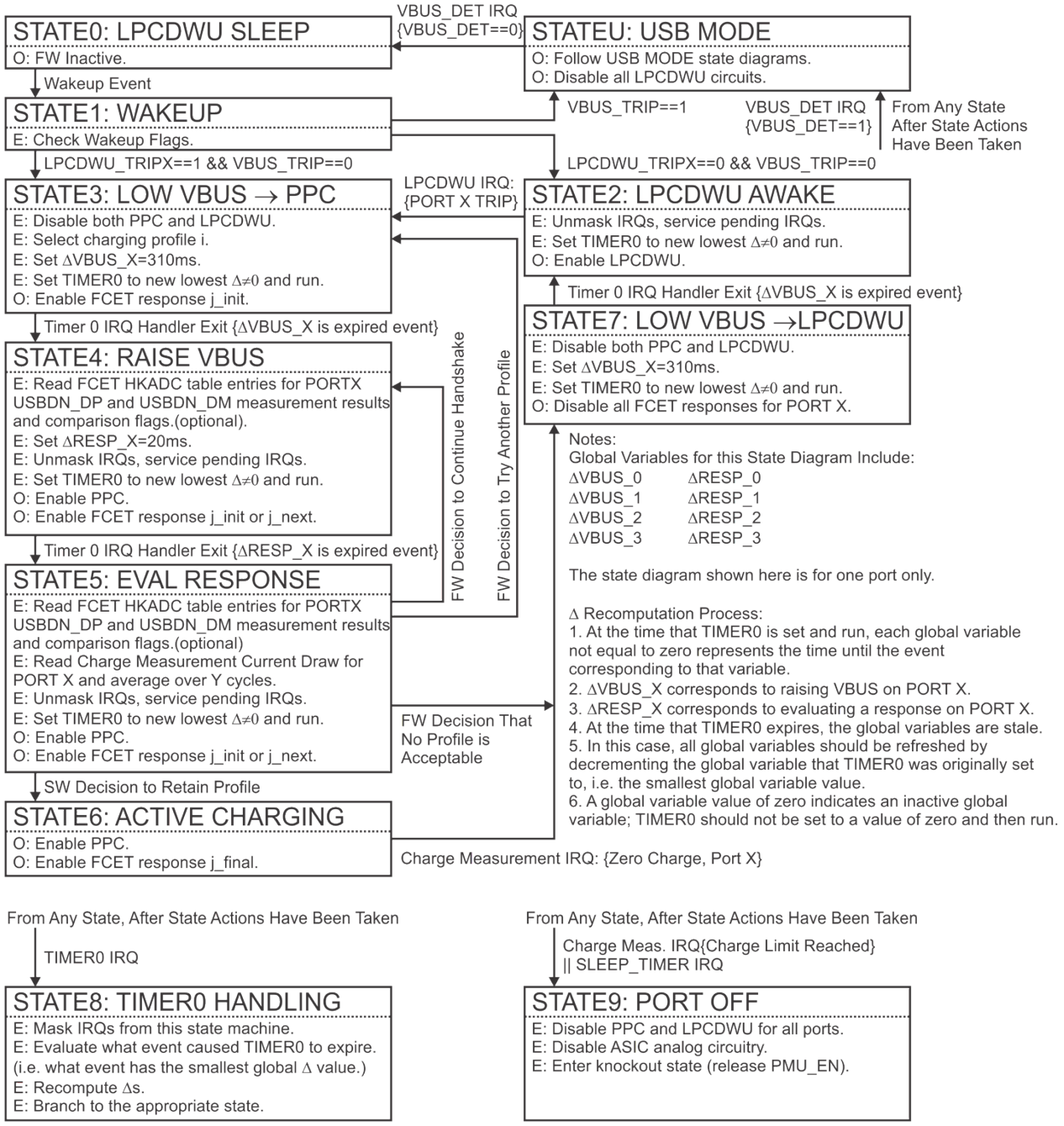
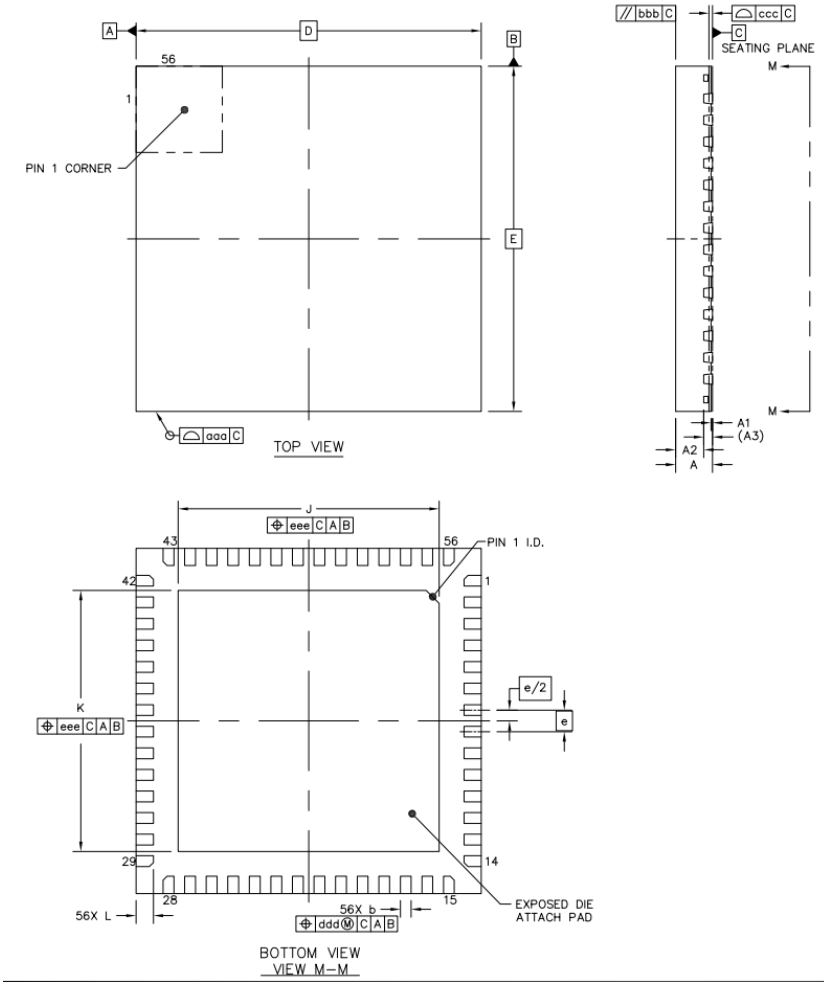


Figure 7-50 Proposed Firmware State Diagram (Per Port) for Charging Mode

8.0 Packaging Information

The following information is mechanical information which is to be transmitted to the customer separately for readability, however it is being presented here for completeness.



		SYMBOL	MIN	NOM	MAX
TOTAL THICKNESS		A	0.8	0.85	0.9
STAND OFF		A1	0	0.035	0.05
MOLD THICKNESS		A2	---	0.65	0.67
L/F THICKNESS		A3	0.203 REF		
LEAD WIDTH		b	0.2	0.25	0.3
BODY SIZE	X	D	8 BSC		
	Y	E	8 BSC		
LEAD PITCH		e	0.5 BSC		
EP SIZE	X	J	5.95	6.05	6.15
	Y	K	5.95	6.05	6.15
LEAD LENGTH		L	0.35	0.4	0.45
PACKAGE EDGE TOLERANCE		aaa	0.1		
MOLD FLATNESS		bbb	0.1		
COPLANARITY		ccc	0.08		
LEAD OFFSET		ddd	0.1		
EXPOSED PAD OFFSET		eee	0.1		

NOTES

1.0 COPLANARITY APPLIES TO LEADS, CORNER LEADS AND DIE ATTACH PAD.

ASE Group		PROJ.	
TITLE: PACKAGE OUTLINE 56 LD QFN EXPOSED PAD 8 X 8 X 0.9 PKG. 0.5 PITCH		DWG. NO.	REV.
		98A0056QN041	O
		SHEET	SIZE
		1 OF 2	A4
UNIT	DIMENSION AND TOLERANCES	REFERENCE DOCUMENT	
MM	ASME Y14.5M	---	

9.0 Appendix 1 – Register Definitions

The complete register map for Boston is entitled: Boston Register Map.doc and an up-to-date version of this document is retained in the AyDeeKay project repository under the directory: /Boston/Specification.

10.0 Appendix 2 – Interrupt Definitions

Table 10-1 Cortex M0 (Boston) NVIC Boot/IRQ Table

Number	Name	Description
0x00	SRAM_TOP	Top of stack
0x01	Reset_Handler	Reset Handler.
0x02	NMI_Handler	NMI Handler
0x03	HardFault_Handler	Hard Fault Handler
0x04	0	<RESERVED>
0x05	0	<RESERVED>
0x06	0	<RESERVED>
0x07	0	<RESERVED>
0x08	0	<RESERVED>
0x09	0	<RESERVED>
0x0A	0	<RESERVED>
0x0B	SVC_Handler	SVCall Handler
0x0C	0	<RESERVED>
0x0D	0	<RESERVED>
0x0E	PendSV_Handler	PendSV Handler
0x0F	SysTick_Handler	SysTick Handler
0x10	SD_INT (IRQ 0)	SD Card Host Interrupt Request
0x11	FIFOINTREQ (IRQ 1)	USB Device Endpoint FIFO Interrupt Request
0x12	USBINTREQ (IRQ 2)	USB Interrupt for internal device at DS0
0x13	WUINTREQ (IRQ 3)	USB Wakeup Interrupt for internal device at DS0
0x14	H2HD_USBINTREQ (IRQ 4)	H2H device-side DEV USB interrupt
0x15	SPI_I2C_INT (IRQ 5)	Combined SPI/I2C Interrupt. Exclusive to each other since SPI/I2C share the interface, selected by PIN_TEST_MODE bit at table 52 of Boston register map.
0x16	HSF_INT (IRQ 6)	Unused.
0x17	PORT_STATE_CHANGE (IRQ 7)	Reports to MCU when an interesting downstream port state change event has occurred. See Section 7.3.8.4 for more details.

Table 10-1 Cortex M0 (Boston) NVIC Boot/IRQ Table		
Number	**Name**	**Description**
0x18	HKADC_INT (IRQ 8)	Reports to MCU when one of the preprogrammed HKADC comparison thresholds has been exceeded. See Section 7.3.7 for more details.
0x19	LPCDWU_INT (IRQ 9)	Reports to MCU when one of the LPCDWU circuits has been triggered while in Charging Mode. See Section 7.3.8.5 for more details.
0x1A	CHARGE_MEAS_INT (IRQ 10)	Reports to MCU when one of the Charge Measurement DBE thresholds has been triggered. See Section 7.3.6.3.3 for more details.
0x1B	OVER_CURR_INT (IRQ 11)	Reports to MCU when one of the external port power control switches reports an over current event. See Section 7.3.8.3.3 for more details.
0x1C	VBUS_TIMER_INT (IRQ 12)	Combined VBUS_INT/SLEEP_TIMER_INT, read the flag INT_ID to determine the interrupt source. See table 166b at Boston register map. VBUS_INT reports to MCU that a change in state of the VBUS_DET pin has been detected, either signaling to go to sleep or to exit Charging Mode and enter USB Active Mode. See Section 7.3.8.6 for more details. SLEEP_TIMER_INT reports to MCU when the after-vehicle-turnoff charging period has expired. See Section 7.3.8.6 and 7.1.2 for more details.
0x1D	H2HD_USBINTREQ (IRQ 13)	H2H host-side DEV USB interrupt
0x1E	SUSPEND_INT (IRQ 14)	Reports to MCU when the USB Hub has been placed into suspend mode. In this case, the meaning of any timer has changed because the MCU clock has also changed. See Section 7.3.8.2 for more details.
0x1F	H2H_IRQ (IRQ 15)	host-to-host supervisor interrupt See Section 7.3.5.3 for more details.
0x20	IRQ_Timer0_Handler	Timer 0 overflow
0x21	IRQ_Timer1_Handler	Timer 1 overflow
0x22	IRQ_Timer2_Handler	Timer 2 overflow
0x23	IRQ_Watchdog_Handler	Watchdog timer overflow
0x24	IRQ_BTE_Handler	Block transfer complete
0x25	IRQ_SDIO_HANDLER	Serial link error

11.0 References

[1] Universal Serial Bus Specification Revision 2.0, April 27, 2000

[2] USB Battery Charging Specification Revision 1.2, October 12, 2011

[3] Anderson, D., “Universal Serial Bus System Architecture,” Second Edition, Addison-Wesley 2001

[4] SD Specifications Part 1 Physical Layer Specification Version 4.10 October 22, 2012

12.0 Revision History

Rev #	Date	Action	By
0.1	3 Apr 2013	Initial Draft for Discussion with customer team.	EAK
0.2	5 Apr 2013	Added Host Spoofing, SD Card Interface, Total Current, Misc.	EAK
0.3	6 Apr 2013	Updated Reset / PMU Sequencer TBD numbers. Removed ODGPIO from FAULT_N pins.	EAK
0.4	10 Apr 2013	After customer first review, update results of SD Card Verification, USB Hub Port Reordering, add AEC-Q100 requirement and AEC-Q100 ESD requirements, add additional behavior of SLEEP TIMER.	EAK
0.5	27 Aug 2013	Updated Pin List with Tapeout Ordering	EAK
0.6	07 Sep 2013	Updated Pin List with Final Packaging Ordering	EAK
0.9	13 Oct 2013	First pass Post-Tapeout Revision	EAK
1.0	14 Oct 2013	Final pass Post-Tapeout Revision	EAK
2.0	18 Oct 2013	First pass with added features (Audio and H2H)	DDK
2.01	15 Feb 2014	Update Charge Measurement DBE program to relate program variables to physical parameters.	EAK
2.02	10 Mar 2014	Update host-to-host section to correspond to taped out RTL (note, port switching must still be updated by Hao)	SDK

Rev #	Date	Action	By
2.03	11 Mar 2014	Update 7.6.3 H2H port switching section	HAO
2.04	12 Mar 2014	Update Appendix 2 interrupt definitions	HAO
2.05	16 Sep 2014	Update voltage range of operation for FCET Shorting Switch and change 3.63V device voltage drop requirement to a more general statement regarding reliability.	EAK
2.06	10 Feb 2015	Update with C1 qualitative changes. Deleted host spoofing material. Noted new pin types for SPI_DI and SPI_CE_N. Removed references to 4MHz clock being dividable down to 1MHz. Removed claims of meeting USB suspend current requirement. Updated FCET shorting switch and resistive divider sections. Removed references to specification noncompliance in revision A1.	EAK
2.07	3 Mar 2015	Added packaging information	JI

Aptiv Exhibit 2023 Page 158
Microchip v. Aptiv IPR2024-00230