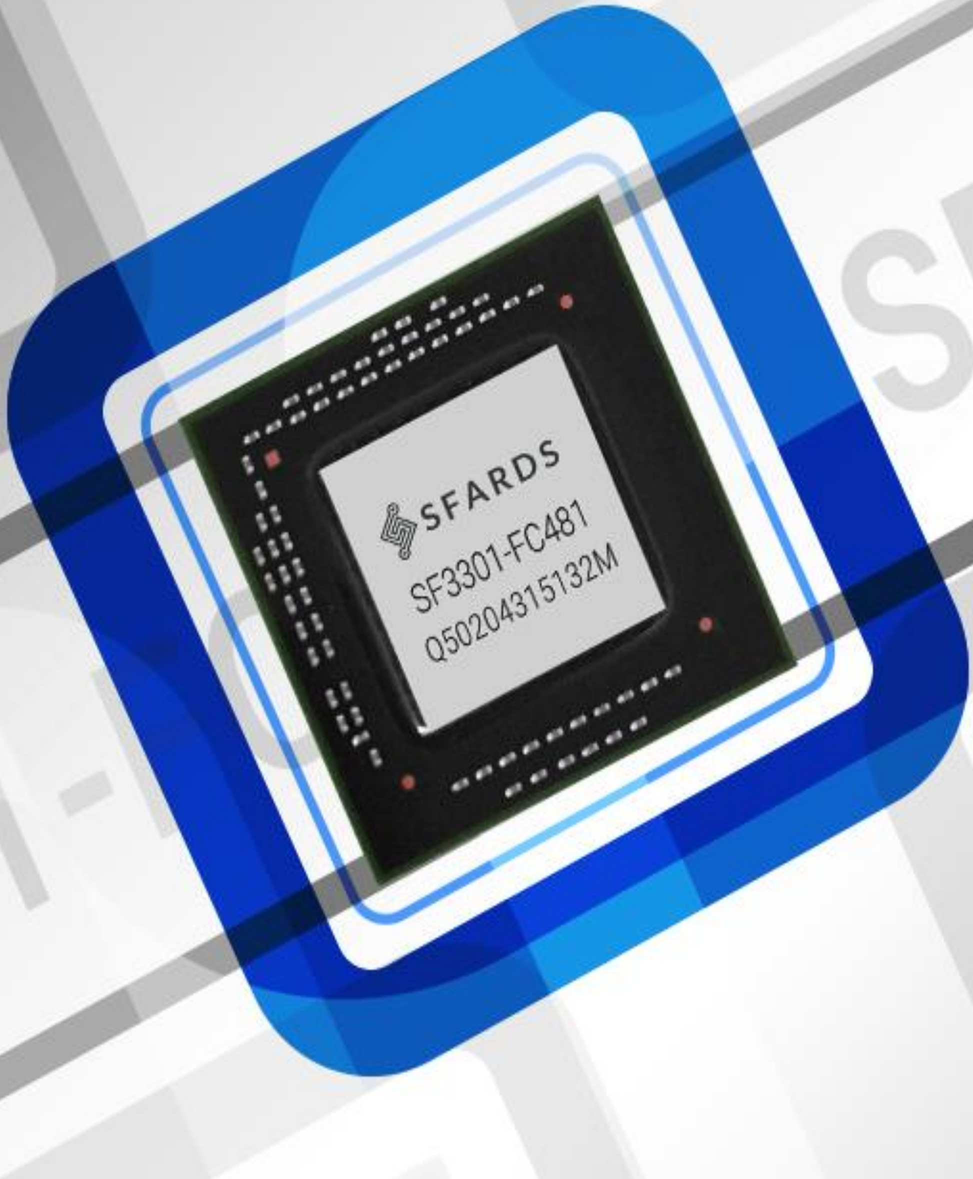
SFARDS

SF3301 DATASHEET



SFARDS
www.sfards.com

Disclaimer

The information contained in this datasheet is for general information purposes only. The information is provided by Sfards Technology and while we endeavour to keep the information up-to-date and correct, we make no representations or warranties of any kind, express or implied, about the completeness, accuracy, reliability, suitability or availability with respect to the datasheet. Any reliance you place on such information is therefore strictly at your own risk. All users should evaluate the data suitability for each intended application of that data under actual use conditions. In no event will we be liable for any loss or damage including without limitation, indirect or consequential loss or damage, or any loss or damage whatsoever arising from loss of profits arising out of or in connection with the use of this datasheet.

Revision History

Version	Date	Comments
0.5	2015.04.20	Initial release version.
0.51	2015.05.05	Fix pin’s name of pins section

Contents

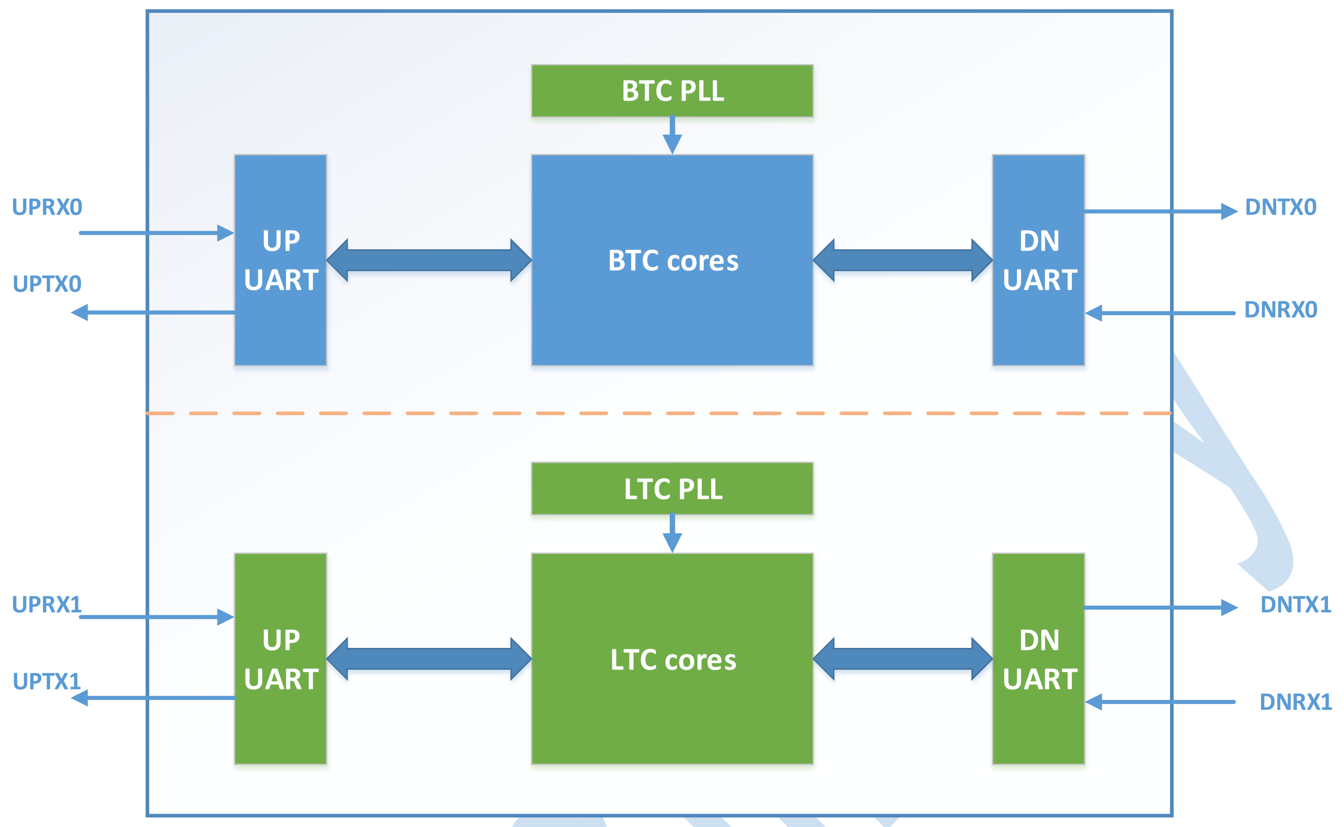
Product Outline

SF3301 is high performance and low power SHA256/SCRYPT dual processor designed by SFARDS. With advanced technology and highly integrated design, the SF3301 is targeted to provide multiple functions and a low cost solution in SHA256/SCRYPT application fields.

Specifications:

- 160 BTC Units
- 31 LTC Units
- BTC mode up to 80GH/s with 0.31W/GH
- LTC mode up to 1.89MH/s with 2.0W/MH
- Dual-Mining mode: 100GH/s BTC & 1.75MH/s LTC
- Highly integrated with PLL and Pre-Calculation Engine of BTC
- 2-wires UART interface
- Support Crystal and Oscillator
- Fully adjustable clock frequency
- Support body-bias adjust
- On-chip thermal sensor

1. Architecture



2.Protocol Specification

The SF3301 uses two separate groups of uart ports for BTC and LTC. Both BTC and LTC each have two uart ports, one for input from the software or the previous SF3301 in the chain (pins UPRX0 UPRX1) and feedback to software or the previous SF3301 (pins UPTX0 UPTX1), the other for downstream to next SF3301 (pins DNTX0 DNTX1) and feedback from next SF3301 (pins DNRX0 DNRX1).

SF3301 register configuration format includes command header and data.

An 8 bit UART bus is used all the time. Every frame is made of 32 bit command header followed by 32 bit data or multiple 32 bit data if any. All the 32 bit are comprised of 8 bit uart.

Command header is marked by 55, which should be sent firstly, then host send CHIP_ADDR (8 bits), UNIT_ADDR (8 bits), REG_ADDR (8 bits) sequentially.

After command header, host is able to write N*32 bits data consecutively（N≧1）to REG_ADDR, REG_ADDR+1, REG_ADDR+2, … REG_ADDR+N-1.

Whatever command header and data, the order of 32 bits is:

Byte0/Bit [7:0], Byte1/Bit [15:8], Byte2/Bit [23:16], Byte3/Bit [31:24].

Because UART will transmit each byte from bit [0] to bit [7], 32bits data is transmitted from bit [0] to bit [31] on UART.

The command header and data format are reviewed in following table：

	Byte3	Byte2	Byte1	Byte0	Transmit Order	Data@Register
Command Header	REG_ADDR	UNIT_ADDR	CHIP_ADDR	55	55 CHIP_ADDR UNIT_ADDR REG_ADDR	
DATA0	B3	B2	B1	B0	B0 B1 B2 B3	0xB3B2B1B0 @ REG_ADDR
DATA1	B7	B6	B5	B4	B4 B5 B6 B7	0xB7B6B5B4 @ REG_ADDR+1
DATA2	BB	BA	B9	B8	B8 B9 BA BB	0xBBBAB9B8 @ REG_ADDR+1

3.Command Reference

There are 8bits CHIP_ID register inside each SF3301 chip. SF3301 only accept register access when CHIP_ADDR of command header match CHIP_ID or CHIP_ADDR is 0xff (It means this is a broadcasting command)

SF3301 supports burst mode for register configuration, so the below 2 sequences have the same function. SF3301 request time interval between 2 commands must longer than UART timeout value.

Sequence 1: Command Header (REG_ADDR), DATA0 (32bits), DATA1 (32bits), DATA2 (32bits), DATA3 (32bits)

Sequence2: Command Header (REG_ADDR), DATA0 (32bits) -> Command Header (REG_ADDR), DATA1 (32bits), DATA2 (32bits), DATA3 (32bits)

SF3301 report format:

1. Store Mode = 1

Report is 32 bits, only have payload for register read data, BTC/LTC result. There is no header on UART upstream.

2. Store Mode = 0

Report is 64 bits, format is

- For Register:

If read the CPM register:

Header = 0x55, Chip_id[7:0], {4'h0, 2'b00, 2'b0}, reg_addr;

Else

Header = 0x55, Chip_id[7:0], {4'h0, 2'b01, 2'b0}, reg_addr;

Data = reg_data[31:0]

- For BTC:

Rpt_length= 1:

Header = 0x55(Marker), CHIP_ID(8 bits), {RPT_TYP(1: BTC; 0: Reg), mem_remain[6:0]}, {6'h0, taskID[1:0]};

Rpt_length= 0:

Header = 0x55(Marker), CHIP_ID(8 bits), {RPT_TYP(1: BTC; 0: Reg), mem_remain[6:0]}, 8'h0;

Data = Nonce(32bits)

- For LTC:

Header = 0x55(Marker), CHIP_ID(8 bits), {RPT_TYP(1: BTC; 0: Reg), mem_remain[6:0]}, 8'h0;

Data = Nonce(32bits)

4. Registers Reference

The 32 bit registers inside each chip are described as below.

4.1. CPM Registers

UNIT_ADDR=0xF0 (BTC CPM/LTC CPM are physically independent, logically they use 0xF0 address)

REG ADDR(8 bits)	NAME	Default Value	Description
0x0	[31]pll_strb	0x0	Asynchronous strobe input to the output divider. When high, signals to the output divider that new values of pll_OD are to be loaded. Cleared by logic on the next cycle. This signal is not controlled by cfg_cpm. It only functions when pll is under normal mode. Fix width 480ns=12x40ns.
	[30]pll_BP	0x0	PLL Bypass mode select, Fout = Fin
	[29:24]pll_OD	0x0(ltc)/0x2(btc)	PLL Output divider, Fout = Fvco/pll_OD (pll_BP = 0) 000000: Fout = Fvco 000001: Fout = Fvco 000010: Fout = Fvco/2 000011: Fout = Fvco/3 000100: Fout = Fvco/4 ..... 111110: Fout = Fvco/62 111111: Fout = Fvco/63

	[23:17]pll_F	0x1a(for ltc) /0x20(for btc)	PLL Feedback divider Fvco = Fref*pll_F 000_0XXX: NA 000_1000: Fvco = Fref*8 000_1001: Fvco = Fref*9 ….. 111_1110: Fvco = Fref*126 111_1111: Fvco = Fref*127
	[16:14]pll_R	0x0	PLL Input divider Fref = Fin/pll_R 000: Fref = Fin 001: Fref = Fin 010: Fref = Fin/2 011: Fref = Fin/3 100: Fref = Fin/4 101: Fref = Fin/5 110: Fref = Fin/6 111: Fref = Fin/7
	[13]count_on	0x1	When low, PHI is gated to zero 1'b1: Pll output clock enable 1'b0: Pll output clock disable
	[12]Reserved	-	x
	[11]LOCKP10	RO	PLL lock state. This signal when high, indicates that the loop is in the state of COARSE LOCK. This signal indicates that the output frequency is within +/-10% (approximately) of the desired frequency.
	[10]LOCKP3	RO	PLL lock state. This signal when high, indicates that the loop is in the state of COARSE LOCK. This signal indicates that the output frequency is within +/-3.5% (approximately) of the desired frequency.
	[9:6]Reserved	-	x

	[5]core clock output divider disable	0x1	For debug pll clk output. 1’b1: disable core output clock divider 1’b0: enable core output clock divider
	[4]core_clk_sel	0x1	Outside clock as Default select. 1’b1: core clock = external clock 1’b0: core clock = PLL output clock
	[3]pll_enable	0x1	When this bit is at 0, PLL will enter power down mode. Write 1 to enter normal mode. If use pll_recfg to reconfig the PLL, this bit should be set 1.
	[2]pll_recfg	0x0	Re-config PLL
	[1]Reserved	-	x
	[0]cfg_cpm	0x0	Config CPM, Write 1,Clear by HW All pll parameters except pll_strb should be updated until SW sets this bit.
0x1	[31:17]Reserved	-	x
	[16]dbg_sel(only in ltc cpm register)	0x0	DBG pin used for debug. Select between two source: 1’b1: output core_clock div. 1’b0: output OSC_OK.
	[15]Reserved	-	-
	[14]xclkout_oen(only in ltc cpm register)	0x0	Xclkout output enable. Low active. 1’b1: PAD is input mode, need configure the Pull up and pull down. 1’b0: PAD output Xclkout.
	[13]xclkout_hyst(only in ltc cpm register)	0x0	Xclkout HYST enable. High active.
	[12]xclkout_lowemi(only in ltc cpm register)	0x0	Xclkout LOWEMI enable. High active.
	[11]xclkout_pdn(only in ltc cpm register)	0x1	Xclkout internal pull down disable 1’b1: Disable pull down 1’b0: Enable pull down

	[10]xclkout_pun(only in ltc cpm register)	0x1	Xclkout internal pull up disable 1’b1: Disable pull up 1’b0: Enable pull up
	[9]dntx_pdn	0x1	Downtream Uart Tx internal pull down disable 1’b1: Disable pull down 1’b0: Enable pull down
	[8]dntx_pun	0x1	Downtream Uart Tx internal pull up disable 1’b1: Disable pull up 1’b0: Enable pull up
	[7]dnrx_pdn	0x1	Downtream Uart Rx internal pull down disable 1’b1: Disable pull down 1’b0: Enable pull down
	[6]dnrx_pun	0x1	Downtream Uart Rx internal pull up disable 1’b1: Disable pull up 1’b0: Enable pull up
	[5]uptx_pdn	0x1	Uptream Uart Tx internal pull down disable 1’b1: Disable pull down 1’b0: Enable pull down
	[4]uptx_pun	0x1	Uptream Uart Tx internal pull up disable 1’b1: Disable pull up 1’b0: Enable pull up
	[3]uprx_pdn	0x1	Uptream Uart Rx internal pull down disable 1’b1: Disable pull down 1’b0: Enable pull down
	[2]uprx_pun	0x1	Uptream Uart Rx internal pull up disable 1’b1: Disable pull up 1’b0: Enable pull up

	[1:0]pad_output_mode	0x3	Pad_output_mode: 00: open-drain mode 01: output low 10: open-drain mode, same to 00 11: normal mode
0x2	core_clk_en[31:0]	0xffffffff	BTC core clock N enable； N:0-159 LTC core clock N enable: N: 0-30
0x3	core_clk_en[63:32]	0xffffffff	
0x4	core_clk_en[95:64]	0xffffffff	
0x5	core_clk_en[127:96]	0xffffffff	
0x6	core_clk_en[159:128]	0xffffffff	
0x1e	[1] uart_rpt_rstn	0x1	reset uart rpt, low active
	[0] core_rstn	0x1	Reset core, low active
0x20	[31]uart_update_en	0x0	Uart parameter update enable. Write this bit will generate a pulse to clear uart internal counter.
	[30]uart_tsm_slow	0x0	Uart transmitter slow mode
	[29]timeout_en	0x1	Uart timeout enable Enable UART to clear internal state when timeout happens.
	[28]uart_rev_fast	0x1	Uart receiver fast mode 0: Uart receiver needs to wait 1 bit time on UART after stop bit 1: Uart receiver does not wait
	[27:26]uart_mode	0x1	Baud rate divider mode 10: 4 times oversample 01: 8 times oversample 00: 16 times oversample

	[25:16]uart_divider0	10‘d130	Baud rate divider fraction
			
			X.Y = External Clock Frequency/(Target Baudrate*uart_mode)
			
			uart_divider0 = INT(0.Y*1024)

	[15:8]uart_divider1	‘d27	Baud rate divider integer
			X.Y = External Clock Frequency/(Target Baudrate*uart_brdiv_mode)
			uart_divider1 = X

	[7:0]timeout_value	‘d31	Uart timeout threshold Time interval between 2 commands. The unit is 1 UART bit time.
0x21	[31:24] downstream mode timer	0x22	When uart mode changes or uart bps changes, it will effect after this delay, the delay value is downstream mode timer << 8
	[23:16] upstream mode timer	0x11	When uart sends a report to host, it will insert delay between 2 transactions, the delay value is upstream mode timer << 8
	[5] store mode	0	0: add report header;1: no report header
	[4] upstream mode	0	1: Insert delay between reports; 0: Don't insert delay between reports
	[3:2]	reserved	
	[1] downstream uart0 disable	0	1: downstream uart0 disable
	[0] downstream mode	0	1: bypass mode; 0: forward mode
0x30	Reserved	-	x
0x50(Only in LTC)	[31:30]Reserved	-	-
	[29:25]ths_clk_div	5’d25	Thermal sensor clock divider. It needs a clock frequency in 400kHz—800kHz. Fclk_ths = Fxclk/(ths_clk_div*2)
	[24]ths_ovfl	RO	High indicates overflow of the digital adder.
	[23:16]ths_first_data	RO	The first output data. Used for calibration.
	[15:8]ths_data	RO	8 bit digital word that indicates junction temperature
	[7:6]Reserved	-	-
	[5:1]ths_correct	5’d16	Set the offset correction value.

	[0]ths_pdn	1’b0	Thermal sensor power down. 1’b1: Normal work 1’b0: Power down mode
0x60	spare	32’b0	Reserved register. Can be read and written. No function. The low 8bits is cleaned by HW.
0x75	DEVICE_ID	0x4743328B(BTC) 0x47433281 (LTC)	Read-only, DEVICE_ID
0x7f (configure this register before access other registers, and which can be configured only once)	[31]unconfig		IsAutoCfgCmd Write: 1 means this is AutoCfgCmd Read: 1: chip is unconfig; 0: chip is config
	[30] EnUartPipeAfterAutoCfg	0	1: HW switch to bypass mode after AutoCfgCmd 0: HW do not switch to bypass mode after AutoCfgCmd
	[29:12] Reserved		
	[11] AutoCfgNonce	WO	1: HW AutoCfgNonce 0: HW does not AutoCfgNonce
	[10:8] AutoCfgNonce Mode	WO	Nonce_Init=chip_addr<<(24+ AutoCfgNonce Mode), valid when AutoCfgNonce is 1
	[7:0]chip_id	0	CHIP ID

4.2. LTC Registers

UNIT_ADDR=0x80-0x9E, 0xBF means broadcast to all LTC Units. Only 0x3f is readable

REG ADDR (8 bits)	NAME	DEFAUT VALUE	Description

0x0 (reset controlled by hw_rst_n)	INIT_NONCE	0	Initial nonce of LTC units
0x1-0x8	D1-D8	0	LTC Target
0x9-0x10	D9-D16	0	LTC MIDSTATE
0x11-0x23	D17-D35	0	LTC DATA_IN
0x2c	[31:16]LTC DLY0	16'd2048	core delay in unit
	[15:0] LTC DLY0	23*1024/31	unit delay
0x2d	LTC DLY1	1024*23+10	round delay of unit
0x2f	[31:0]Reserved	-	
	[31] cfg_done_rstn_enable	1	1: when starting a new task, the core will be reset automatically. 0: when starting a new_ task, the core will not be reset automatically. So before a new task, please use "stop_cur_task" or "sw_rst_n" to stop the current task
	[30]nonce_mask	1'b0	1'b1: LTC core nonce will not add step. For bist mode
	[29:21] reserved		
	[20] clk_enable	0	1'b1: clock enable 1'b0: clock gating
	[19:16]gate_scrypt_core	0x0	gate scrypt_core[3|2|1|0] of ltc unit 1'b1: clock disable 1'b0: clock enable
	reserved		
	[3]stop_cur_task	0	Write 1 to Stop the current LTC task, HW will clear this bit automatically after write 1
	[1] auto_cmp_mode	1	If auto_cmp_mode==1 & LTC_Target0 != 0, LTC will only match unless scrypt result == Target
0x30	[0] ltc_cmp_mode	0	LTC compare mode

			1’b1: LTC will only match unless scrypt result == Target 1’b0: LTC will only match unless scrypt result <= Target
0x3e	[31:0]bist result	0(RO)	Under bist mode, it will indicate the bist result. If the bit is 1, the core is good, otherwise is fault.
0x3f*(This register is shared by all LTC Units)	[10:7] bist_result_level		[10] all fault [9] 50%<=GoodCores<80% [8] 80%<=GoodCores<100% [7] GoodCores == 100%
	[6] bist_mode	0	Ltc bist mode
	[5] single_shift	0	Register value shifts to compute unit in every single Register access.
	[3:0]pulse width	8	LTC TOP CFG Pulse Width Cycle Number Should be set to >= 3

4.3. BTC Registers

UNIT_ADDR=0x00-0x9F, 0xEF means broadcasting to all BTC units. Only 0x1f is readable.

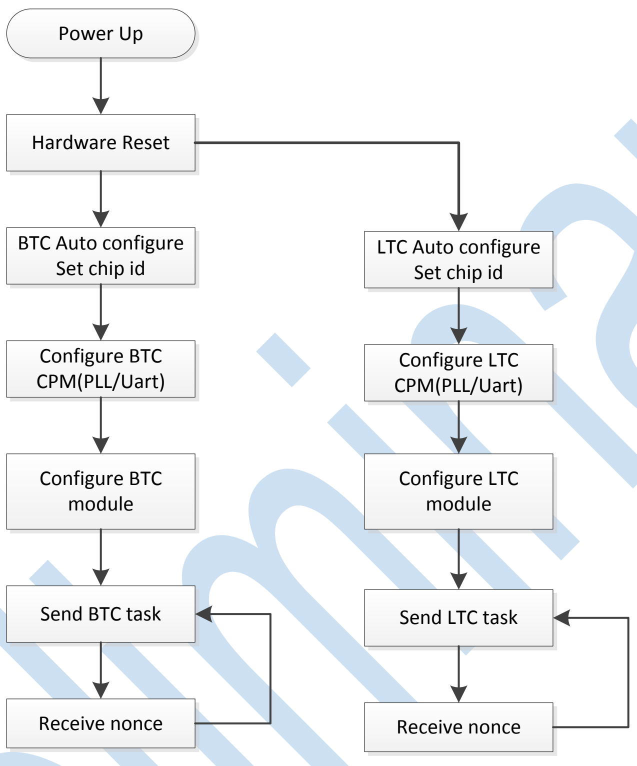
REG ADDR (8 bits)	NAME	DEFAUT VALUE	Description
0x0	INIT_NONCE	0	Initial nonce of BTC units
0x1	Tgt_diff	0	The tgt[6]
0x2-0x9	Midstate 0-7	0	Midstate
0xa-0xc	Data2 0-2	0	Data2
0x1d	[31:0]bist result	0(RO)	Under bist mode, it will indicate the bist result. If the bit is 1, the core is good, otherwise is fault.
0x1e	[31:3] reserved	-	
	[2] btc_task_en	0	Btc task enable: Default, the core will start when configuration done. If SW need to stop the btc task, it needs to

			be set to 0
	[1] cmp_tgt_en	1	Compare the hash with tgt[6]
	[0] rpt_length	1	1: The reported nonce includes taskID 0: The reported nonce excludes taskID
	[31:12]nonce_comp	((32*19*pulse_width*2*(160+1) + 16*pulse_width*2))/16	If cal_nonce add this value overflows, the BTC will start to load next job.
	[11]rpt_length_top	1	This bit must be the same value as 0x1e bit[0] rpt_length. 1'b1: The reported nonce includes taskID, which for btc_top level 1'b0: The reported nonce excludes taskID, which for btc_top level
0x1f*(This register is shared by all BTC_Units)	[10:7] bist_level	0	[10] all fault [9] 50%<=GoodCores<80% [8] 80%<=GoodCores<100% [7] GoodCores == 100%
	[6]bist_mode	1	For bist test
	[5]single_shift	0	1: the job data will shift to core when top receive every data(must include data3) 0: only when the job data is received, then the top can shift the data to core
	[4]force_start	0	1: when the job has been configured, the core will start. 0: only when the current job overflow, the core will start next job.
	[3:0]pulse width	8	The pulse width should set to >= 3

5.Firmware Implementation Reference Guideline

5.1. Workflow

The workflow of firmware is shown in Fig.1. It is noted that the BTC and LTC should configure independently after a reset because BTC /LTC module are physically independent.



Power Up
Hardware Reset
BTC Auto configure Set chip id
LTC Auto configure Set chip id
Configure BTC CPM(PLL/Uart)
Configure LTC CPM(PLL/Uart)
Configure BTC module
Configure LTC module
Send BTC task
Send LTC task
Receive nonce
Receive nonce

Fig 1. Firmware workflow

5.2. Auto Configure

After reset, SF3301 become un-configured mode and CHIP ID is 0, and miner controller send auto configure command, all the chip will be set to configure mode in the chain, and the chip id will auto increase 1 at the next level chip. As the Fig.2 show, when setting auto configure the first SF3301 chip id to n(0 <= n < 0xff), the following chips id in the chain will set to n+1, n+2, ..., n+m(n+m < 0xff). After configured, host can access all the register of each chip in the chain by chip id. Auto configuration only does once time after hardware reset.

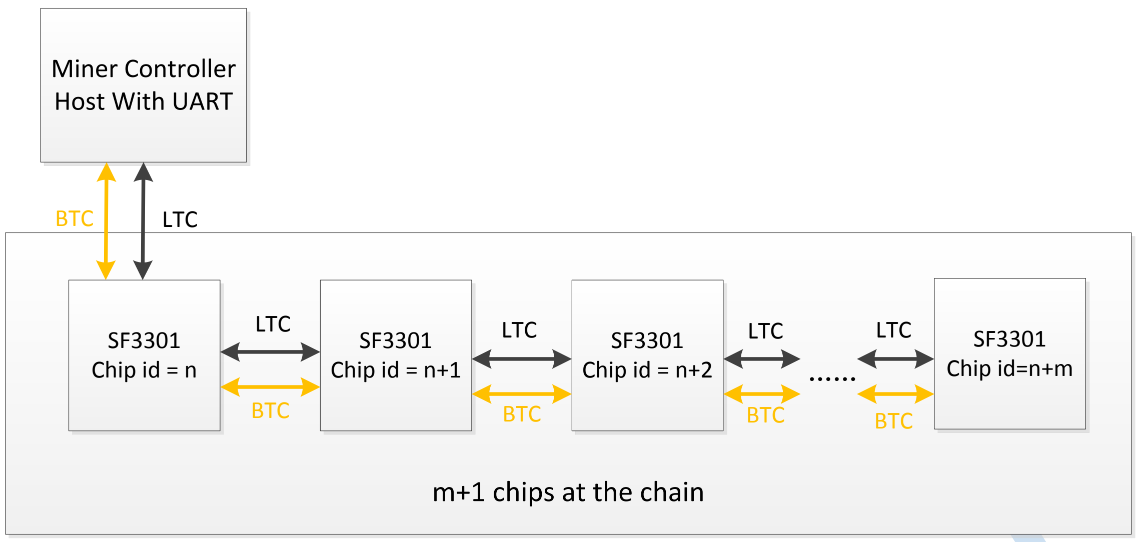


Fig 2. Auto configure chip id in chain-style net

5.3. PLL Setting

The PLL is bypass at default mode, so the clock of the core is from external crystal oscillator (Fin). When reconfigure the PLL, the calculation formula of PLL frequency is as follows:

*PLLout = Fin/pll_R*pll_F/pll_OD* (PLLout is up to 1200MHz)

1 <= pll_R <= 7, 8 <= pll_F <= 127, 1 <= pll_OD <=63

Table 1 list typical PLL setting example:

Table 1. PLL setting example

Fin(MHz)	pll_R	pll_F	pll_OD	PLLout(MHz)	BTC(Mhash/s)	LTC(Khash/s)
25	1	8	8	25	4000	134
25	1	32	8	100	16000	539
25	1	40	8	125	20000	673
25	1	8	1	200	32000	1078
25	1	9	1	225	36000	1213
25	1	12	1	300	48000	1617
25	1	14	1	350	56000	1886
25	1	16	1	400	64000	2156
25	1	18	1	450	72000	2426
25	1	20	1	500	80000	2695
25	1	24	1	600	96000	3234
25	1	28	1	700	112000	3773
25	1	32	1	800	128000	4313

When change the PLL setting, the first step should gate the PLL (PLL reg[13]=0), then configure the PLL and disable gating PLL (PLL reg[13]=1).

5.4. Uart Baud Rate Setting

When Fin=25MHz, the default uart baud rate is 115200bps, the recommend baud rate is 2Mbps to meet the high hash rate. The calculation formula of baud rate is as follows:

Baud rate = Fin/ Uart_mode/(Uart_Divider1 + Uart_Divider0/1024)

Table 2 list typical baud rate configuration:

Table 2. Uart baud rate typical setting

Fin(MHz)	Uart_mode	Uart_Divider1	Uart_Divider0	Baud rate(bps)
25	8	27	129	115200
25	8	13	576	230400
25	8	6	800	460800
25	8	3	400	961200
25	8	3	128	1000000
25	8	1	576	2000000

Note: It should be delayed at least 1ms to wait for the uart work normal after uart baud rate is changed.

5.5. BTC Setting

There are two modes to accept the BTC work task, force mode and FIFO mode. In the force mode, the BTC core will start the new work task immediately after all data of new task is received, while in the FIFO mode, the BTC core will start new work task, which is stored in the task FIFO, only the original work task is done (nonce overflow), and the task FIFO depth is 1, so the task feed time interval should be appropriate to avoid the FIFO is too much or too little, according to the BTC hash power.

BTC module provides 2bits task ID to double check which nonce belong to which task hit. If Rpt_length set to 1, the BTC task header REG_ADDR[6:5] is used as task ID, and the nonce header byte3[1:0] is used as returned task ID, while Rpt_length set to 0, the task ID function is disabled.

To filter more invalid nonce at the high network difficulty, the hash target[6] is applied to hash result comparison, and it will improve efficiency of host miner by reducing nonce check and even without nonce check at some network difficulty.

BTC configuration example:

//btc auto configure, start chip id = 1

{55 fe f0 7f 01 08 00 c0 }

//select pll, gating output

//set pll Fin=25M, pll_R[16:14] = 1, pll_F[23:17]= 32, pll_OD[29:24]=1, =>800M

{55 ff f0 00 09 00 40 01 }

// Pll output enable

{55 ff f0 00 09 20 40 01}

//config btc to force start mode

{55 ff ef 1f 18 08 e7 17}

//set btc initial nonce=0

{55 01 ef 00 00 00 00 00}

//send btc task and read nonce

{55 01 ef 01 target[6] midstate[0]- midstate[7] data[16]- data[18]}

5.6. LTC Setting

There are two comparison modes of scrypt hash result, the recommend configuration is setting comparison mode 0 (ltc_cmp_mode = 0) and disable auto comparison mode (auto_cmp_mode =0). Because of low speed of scrypt hash rate, the miner host could set initial LTC nonce of every SF3301 is divided equally 4G hash range in the chain. For example, there are 4 SF3301 chips in the chain, the four initial LTC nonces are 0x00000000, 0x40000000, 0x80000000, 0xc0000000.

Note: LTC module does not support task ID double check and work task FIFO.

LTC configuration example:

//ltc auto configure, start chip id = 1

{55 fe f0 7f 01 08 00 c0 }

//select pll to 650M, pll output gating

//set pll Fin=25M, pll_R[16:14] = 1, pll_F[23:17]= 16, pll_OD[29:24]=1, =>400M

{55 ff f0 00 09 00 20 01 }

// Pll output enable

{55 ff f0 00 09 20 20 01}

//set ltc cmp mode

{ 55 ff bf 30 08 00 00 80 }

//set ltc initial nonce=0

{55 01 bf 00 00 00 00 00}

//send ltc task and read nonce

{55 ff ef 01 target[0]-target[7] midstate[0]- midstate[7] data[0]- data[18]}

5.7. Thermal Sensor

The thermal sensor provides digital measurement of the junction temperature, and thermal sensor feature as following:

- Temperature measurement range: -40°C to 125°C,
- Resolution: 1°C, accuracy: +/- 6°C (after calibration),
- Input clock frequency: 400 kHz to 800 kHz.

Before using the thermal sensor, the thermal sensor should be calibrated correctly, and the thermal sensor calibration steps are explained in following steps:

1. Set the SF3301 in normal temperature environment, at a well-known junction temperature, Tj.
2. Set ths_correct [5:1] = 16.
3. Set ths_pdn = High. Read ths_first_data [23:16] code at the first DATAREADY pulse.
4. Compute temperature as Tm = ths_first_data [23:16] - 95 [95 is the scaling factor].
5. Compute the calibration error using the formula Te = Tm - Tj
6. Apply new ths_correct [5:1] = 16 - Te.
7. The corrected output ths_data[15:8] still follows the equation, Tj = ths_data[15:8] - 95.

Calibration example

The calibration starts with Tj = 40, ths_correct [5:1] = 16

1. ths_first_data [23:16] = 134 at the first DATAREADY pulse
2. Measured temperature, Tm (in C) = 134 - 95 = 39
3. Calculate Te = 39 - 40 = -1
4. New ths_correct [5:1] to be applied = 16 - (-1) = 17

Hence, the corrected ths_data[15:8] = 135

After calibration, the thermal register ths_data[15:8] - 95 is the actual temperature, and the miner host should regulate the hash rate(PLL) or dissipate heat to ensure the junction temperature not above 65°C.

Note: the thermal sensor only can be accessed by LTC uart port.

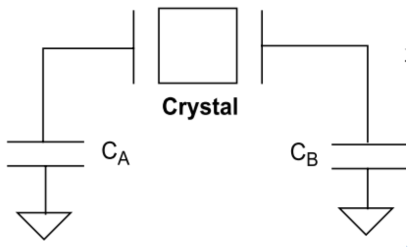
6. Electrical Specifications

6.1. Crystal And Oscillator

PARAMETER	MIN	TYP	MAX	UNIT
Clock Period		40		ns
Clock Frequency	24	25	30	MHz
Clock Duty Cycle	45	50	55	%
Clock Jitter			50	ps

6.2. PCB Board Design Recommendations

For crystal, the PCB board design require the following items:
Max Rm = 50 Ohm, Max Co (shunt capacitance) = 5pF
CA/CB= 20pF (+/- 20%)



6.3. Operation Condition

PARAMETER	MIN	TYP	MAX	UNIT	Max Current
BTC Core Supply Power	0.6	0.7	0.8	V	115A
BTC body-bias negative	0	-0.6	-1.1	V	20MA
BTC body-bias positive	0	0.6	1.1	V	20MA
LTC core supply Power	0.8	0.9	1.1	V	12A
PLL Supply Power	1.62	1.8	1.98	V	0.2A
IO Supply Power	1.62	1.8	1.98	V	0.2A
THS supply power	1.62	1.8	1.98	V	0.2A
OSC supply power	1.62	1.8	1.98	V	0.2A
CPS supply power	1.62	1.8	1.98	V	0.2A
Operating Temperature	0	25	125	℃	

6.4. Power Consumption

For the power consumption of the SF3301, please refer to “SF3301 power report.pdf”.

7.Pins And Ballmap

7.1. Ballmap

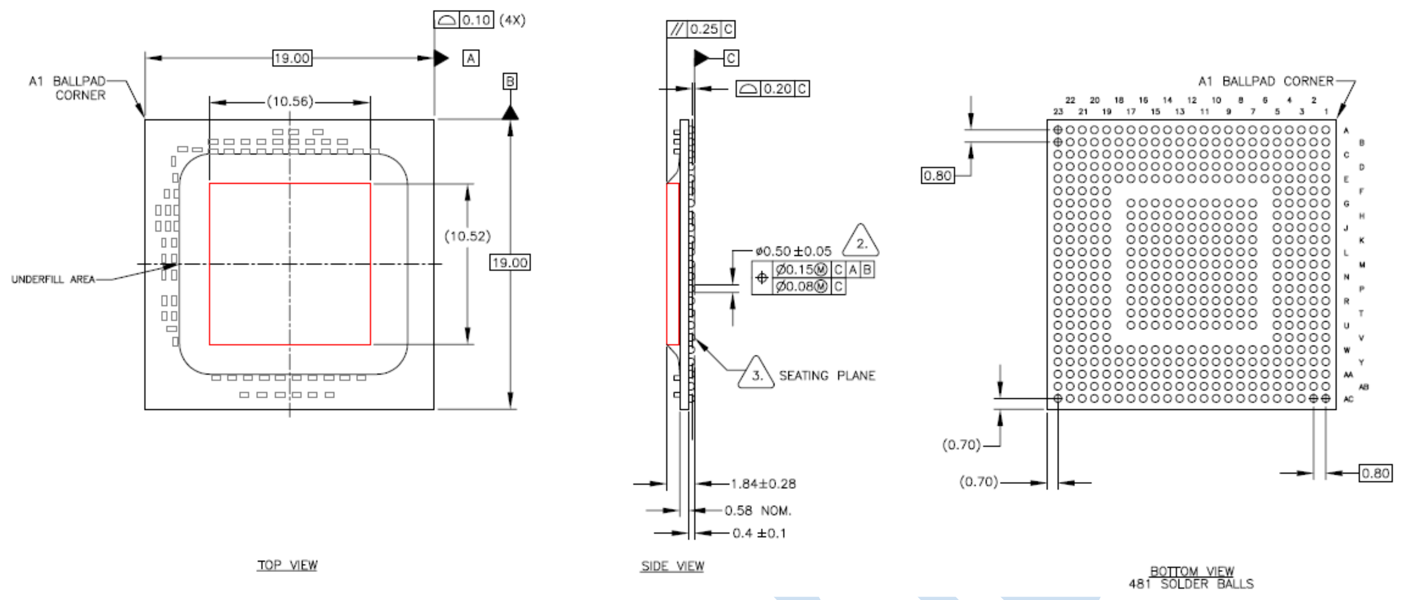
(Package Top View)																								
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	
A	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_GNDS_0	BTC_VDDS_0	AVDD_THS	BTC_GND	A
B	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	ESDSUB_ANA	ESDSUB_ANA	AGND_THS	B
C	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	ESDSUB_ANA	BTC_GND	C
D	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	ESDSUB_ANA	BTC_GND	D
E	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_GND	OSC_ZO	E
F	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD														BTC_VDD	BTC_GND	AGNDSUB_OSC	GNDA_OSC	XCLKIN	F
G	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	VDDA_OSC	VDDE	G
H	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	GNDE	H
J	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	UPRX0	UPTX0	J
K	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		BTC_VDD	BTC_GND	BTC_VDD	GNDE	XCLKOUT	K
L	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	UPTX1	UPRX1	L
M	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		BTC_GND	BTC_GND	BTC_VDD	BTC_GND	RSTN	M
N	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND		BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_VDD		VSSA_PLL0	VDDA_PLL0	BTC_GND	BTC_VDD	VDDE	N
P	BTC_GNDS_1	BTC_VDDS_1	BTC_VDD	BTC_GND	BTC_VDD		BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD		VSSA_PLL1	VDDA_PLL1	BTC_GND	GNDE	OSC_BP	P
R	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	LTC_GND		LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND		BTC_GND	BTC_GND	LTC_GND	DNRX0	DNTX0	R
T	BTC_VDD	BTC_GND	BTC_VDD	BTC_GND	LTC_VDD		LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD		LTC_VDD	LTC_GND	LTC_VDD	GNDE	VDDE	T
U	BTC_GND	BTC_VDD	BTC_GND	BTC_VDD	LTC_GND		LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND		LTC_GND	LTC_VDD	LTC_GND	DNRX1	DNTX1	U
V	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD														LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	DBG	V
W	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	VDDE	W
Y	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	GNDE	VDDE	Y
AA	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	GNDE	ANAREXTPAD	AA
AB	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	VDDE	GNDBGCOMP	AB
AC	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	LTC_VDD	LTC_GND	GNDE	VDDE	AC
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	

7.2. Pins

The pins can be described as followed：

Name	Description
VDDA_OSC	1.8V Supply for OSC
GNDA_OSC	GND for OSC
ESDSUB_ANA	GND for ESD
BTC_GNDS_0	Positive bias power voltage
BTC_VDDS_0	Negative bias power voltage
BTC_GNDS_1	Positive bias power voltage
BTC_VDDS_1	Negative bias power voltage
VDDA_PLL0	1.8V Supply for BTC's PLL
VSSA_PLL0	GND for BTC's PLL
VDDA_PLL1	1.8V Supply for LTC's PLL
VSSA_PLL1	GND for LTC's PLL
AGNDSUB_OSC	Ground for substrate.
AVDD_THS	1.8V Supply for Thermal Sensor
AGND_THS	GND for Thermal Sensor
VDDE	1.8V Supply for IO
GNDE	Ground for IO
BTC_VDD	Digital VDD 0.6V—0.8V
BTC_GND	Digital GND
LTC_VDD	Digital VDD 0.8V—1.10V
LTC_GND	Digital GND
ANAREXTPAD	It should be floating.
GNDBGCOMP	Dedicated ground for the reference cells.
XCLKIN	In Oscillation mode, crystal needs to be connected to this pin and ZO. In Bypass mode, single-end CMOS level clock needs to be applied to this pin.
OSC_ZO	In Oscillation mode, crystal needs to be connected to this pin and A.
RSTN	Reset input pin. Active Low.
UPRX0	Upstream UART0 receive pin
UPRX1	Upstream UART1 receive pin
UPTX0	Upstream UART0 transmit pin
UPTX1	Upstream UART1 transmit pin
DNRX0	Downstream UART0 receive pin
DNRX1	Downstream UART1 receive pin
DNTX0	Downstream UART0 transmit pin
DNTX1	Downstream UART1 transmit pin
DBG	For debug.
XCLKOUT	Output clock is the same as the input clock XCLK
OSC_BP	When high, internal oscillator is by passed.

8. Package



0.10 (4X)
19.00
A
B
A1 BALLPAD CORNER
(10.56)
(10.52)
19.00
UNDERFILL AREA
TOP VIEW
0.25 C
C
0.20 C
ø0.50 ±0.05
2.
Ø0.15 Ⓜ C A B
Ø0.08 Ⓜ C
3. SEATING PLANE
1.84±0.28
0.58 NOM.
0.4 ±0.1
SIDE VIEW
A1 BALLPAD CORNER
22 20 18 16 14 12 10 8 6 4 2
23 21 19 17 15 13 11 9 7 5 3 1
A B C D E F G H J K L M N P R T U V W Y AA AB AC
0.80
(0.70)
(0.70)
0.80
BOTTOM VIEW
481 SOLDER BALLS

9.Information

For more information, please visit SFARDS On Github:
 http://github.com/sfards

http://www.sfards.com