

# Rock2

## DATA SHEET

VERSION 1.1

2006-03-07

**Rockchip Electronics**

## 1 Features

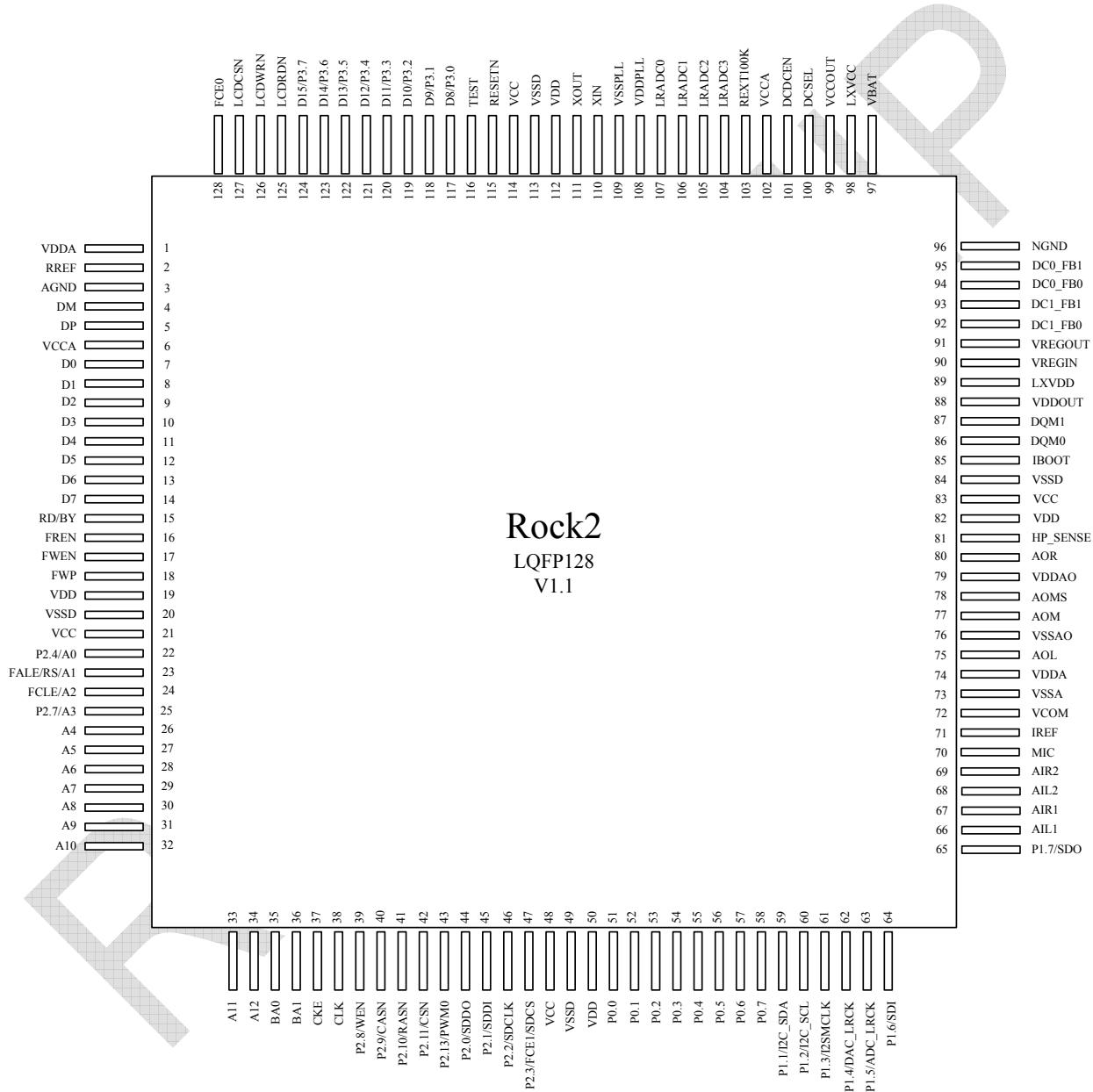
- ◊ 128/100 pins LQFP package
- ◊ Typical power voltage 3.3V(IO), 1.8V(Core)
- ◊ Use one 24MHz crystal oscillator
- ◊ 32 GPIO (8bits P0,P1,P3,16bits P2)
- ◊ 10-bit low resolution ADC with 4-channel Analog Input
- ◊ Build in Stereo 24-bit Delta-Sigma DAC with on-chip headphone amplifier
- ◊ Build in Stereo 16-bit Sigma-Delta ADC (Line-in /FM Input/ Microphone with analog mixer)
- ◊ 40 levels digital volume control
- ◊ Support external CODEC through I2DSP interface
- ◊ Support I2C interface
- ◊ Support USB 2.0 high speed and full speed
- ◊ Integrated 3 Channel DMA
- ◊ Embedded DSP Core:
  - ◆ 4K words Boot Sync ROM
  - ◆ 128K words Sync SRAM
  - ◆ 2K words Register Space for Peripherals
  - ◆ Upgradable firmware through USB/Flash interface
- ◊ Memory interface:
  - ◆ External up to 2(cs) x 64M-2G bytes Nand type Flash accessed by DMA
  - ◆ Support both 8-bit (X8 device) and 16-bit (X16 device) IO bus
  - ◆ Support SDRAM
  - ◆ Support SD/MMC
  - ◆ Support External NOR-Flash boot up
- ◊ Video Driver: Support TFT LCD/ OLED Interface
- ◊ Pulse Width Modulators for EL backlights
- ◊ Support watchdog timer
- ◊ DSP-based Software:
  - ◆ MPEG1/2/2.5 Audio Layer 1, 2, 3 decoding, Layer3 encoding
  - ◆ WMA 9 decoding
  - ◆ G.729 based voice recording and playback
  - ◆ equalizer
  - ◆ MPEG-4 decoding
- ◊ Headphone driver output 2x9mW @32 Ohm(TYP) ; SNR: 90dB (DAC TYP)  
Standby Leakage Current: 25uA  
Low Power Consumption, <70mW at typical MP3 decoder solution (@12MHz)

# Rock2 DATA SHEET V1.1

---

## 2 Pin Description

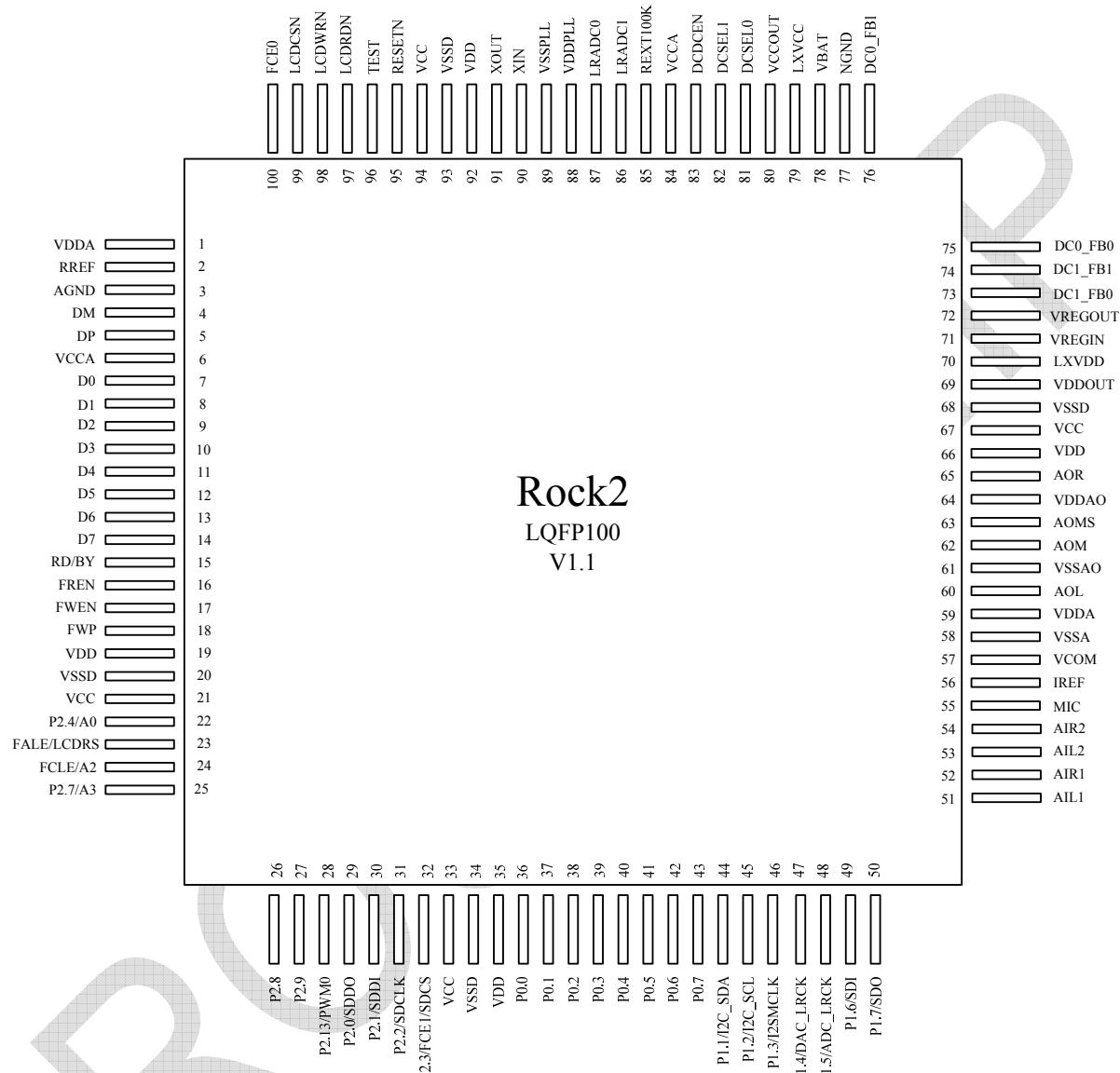
### LQFP128(14x14) PIN



Rock2 128 pins LQFP package

# Rock2 DATA SHEET V1.1

## LQFP100(14x14) PIN



**Rock2**  
**LQFP100**  
**V1.1**

Rock2 100 pins LQFP package

## Rock2 DATA SHEET V1.1

---

### Pin & Pad Descriptions:

<b>128 PIN</b>	<b>100 PIN</b>	<b>Pad Names</b>	<b>PAD Direction</b>	<b>Pin Description</b>
1	1	VDDA	P	Analog 1.8V power output, Connect one external 10uF CAP
2	2	RREF	I	Reference Resistor input
3	3	AGND	P	Analog GND
4	4	DM	A I/O	USB data minus
5	5	DP	A I/O	USB data plus
6	6	VCCA	P	Analog 3.3V power input
7	7	D0	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 0
8	8	D1	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 1
9	9	D2	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 2
10	10	D3	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 3
11	11	D4	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 4
12	12	D5	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 5
13	13	D6	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 6
14	14	D7	I/O Pull up	Flash/ LCD/ SDRAM data bus bit 7
15	15	RD/BY	I, Pull up	Flash ready/busy signal
16	16	FREN	O	Flash read enable
17	17	FWEN	O	Flash write enable
18	18	FWP	O	Flash write protect
19	19	VDD	P	Digital Core power(1.8V)
20	20	VSSD	P	Digital Core Ground
		VSS	P	Digital Ground
21	21	VCC	P	I/O POWER(3.3V)
22	22	P2.4/A0	I/O	SDRAM/ SRAM Address Bit 0    GPIO
23	23	P2.5/A1	I/O	SDRAM/ SRAM Address Bit 1    GPIO;    FALE/LCDRS
24	24	P2.6/A2	I/O	SDRAM/ SRAM Address Bit 2    GPIO;    FCLE
25	25	P2.7/A3	I/O	SDRAM/ SRAM Address Bit 3    GPIO
26	X	A4	O	SDRAM/ SRAM Address Bit 4
27	X	A5	O	SDRAM/ SRAM Address Bit 5
28	X	A6	O	SDRAM/ SRAM Address Bit 6
29	X	A7	O	SDRAM/ SRAM Address Bit 7
30	X	A8	O	SDRAM/ SRAM Address Bit 8

## Rock2 DATA SHEET V1.1

31	X	A9	O	SDRAM/ SRAM Address Bit 9
32	X	A10	O	SDRAM Address Bit 10
33	X	A11	O	SDRAM Address Bit 11 SRAM Address Bit 10
34	X	A12	O	SDRAM Address Bit 12 SRAM Address Bit 11
35	X	BA0	O	SDRAM Bank Address 0 SRAM Address Bit 12
36	X	BA1	O	SDRAM Bank Address 1 SRAM Address Bit 13
37	X	CKE	O	SDRAM clock enable to SDRAM
38	X	CLK	O	system clock to SDRAM
39	26	P2.8/WEN	I/O	SDRAM write enable GPIO
40	27	P2.9/CASN	I/O	SDRAM column address strobe GPIO
41	X	P2.10/RASN	I/O	SDRAM row address strobe GPIO
42	X	P2.11/CSN	I/O	SDRAM chip strobe GPIO
43	28	P2.13/PWM0	I/O	GPIO/ PWM output0
44	29	P2.0/SDDO	I/O	SD/MMC Data output, Rock2 as input Connect to SD/MMC SDDO
45	30	P2.1/SDDI	I/O	SD/MMC Data input, Rock2 as output Connect to SD/MMC SDDI
46	31	P2.2/SDCLK	I/O	SD/MMC Clock output
47	32	P2.3/SDCS	I/O	SD/MMC chip select output
48	33	VCC	P	I/O POWER(3.3V)
49	34	VSSD	P	Digital Ground
50	35	VDD	P	Digital Core power(1.8V)
51	36	P0.0	I/O Pull up	GPIO, External int0
52	37	P0.1	I/O Pull up	GPIO, External int1
53	38	P0.2	I/O Pull up	GPIO, External int2
54	39	P0.3	I/O Pull up	GPIO, External int3
55	40	P0.4	I/O Pull up & EN	GPIO
56	41	P0.5	I/O Pull up & EN	GPIO
57	42	P0.6	I/O Pull up & EN	GPIO
58	43	P0.7	I/O Pull up & EN	GPIO
59	44	P1.1/ I2C_SDA	I/O Pull up & EN	GPIO, External SDA of I2C
60	45	P1.2/ I2C_SCL	I/O Pull up & EN	GPIO, External SCL of I2C
61	46	P1.3/ I2SMCLK	I/O Pull up & EN	GPIO, I2SMCLK of External CODEC
62	47	P1.4 DAC_LRCK	I/O Pull up & EN	GPIO, DAC LRCK of External CODEC
63	48	P1.5/	I/O	GPIO, ADC LRCK of External CODEC

## Rock2 DATA SHEET V1.1

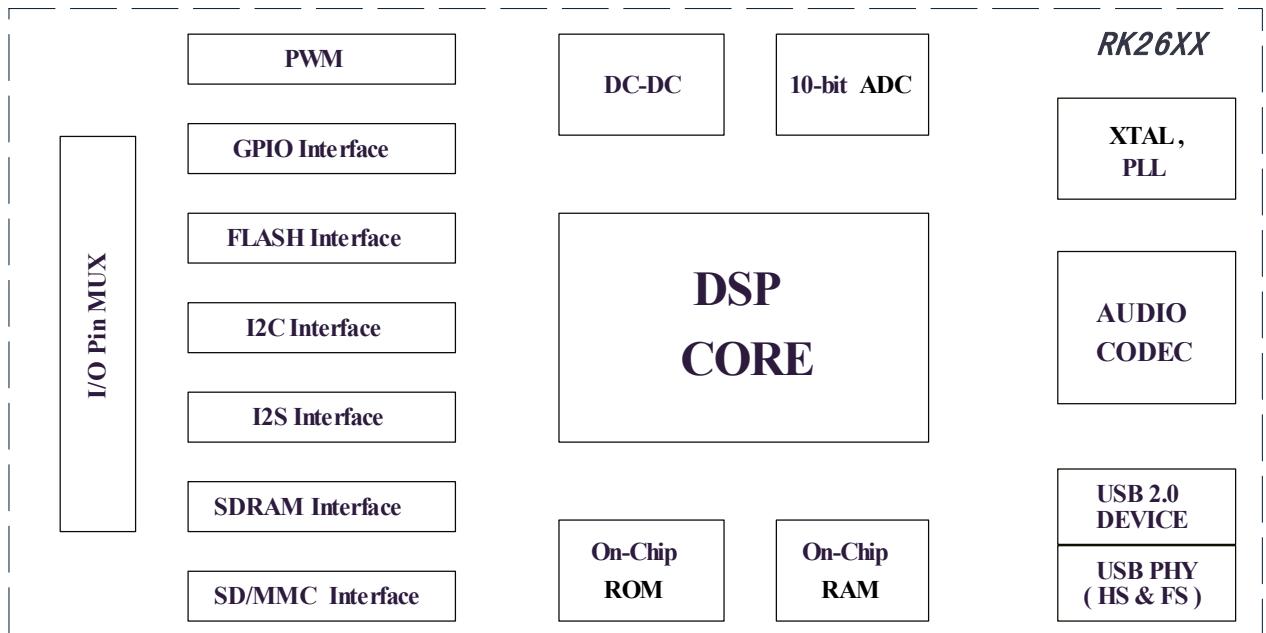
		ADC_LRCK	Pull up & EN	
64	49	P1.6/ SDI	I/O Pull up & EN	GPIO, SDI Data input Connect to SDO of External CODEC
65	50	P1.7/ SDO	I/O Pull up & EN	GPIO, SDO Data output Connect to SDI of External CODEC
66	51	AIL1	I	L-channel single-end input 1
67	52	AIR1	I	R-channel single-end input 1
68	53	AIL2	I	L-channel single-end input 2
69	54	AIR2	I	R-channel single-end input 2
70	55	MIC	I	Mic single-end analog input
71	56	IREF	I	Bias current reference of CODEC
72	57	VCOM	I	Internal biasing voltage for CODEC
73	58	VSSA	P	Negative power supply for CODEC
74	59	VDDA	P	Positive power supply for CODEC
75	60	AOL	O	L-channel single ended analog output
76	61	VSSAO	P	Negative power supply to output amplifiers
77	62	AOM	O	Common mode analog output
78	63	AOMS	I	Common mode sense input
79	64	VDDAO	P	Positive power supply to output amplifiers
80	65	AOR	O	R-channel single ended analog output
81	X	HP_SENSE	I	Sense of jack insertion
82	66	VDD	P	Digital Core power(1.8V)
83	67	VCC	P	I/O POWER(3.3V)
84	68	VSSD	P	Digital Ground
85	X	IBOOT	I, Pull up	Boot select, (Internal 100K pull up)
86	X	DQM0	O	SDRAM DQM0
87	X	DQM1	O	SDRAM DQM1
88	69	VDDOUT	P	DCDC 1.8V output
89	70	LXVDD	P	DCDC 1.8V switching Power input
90	71	VREGIN	P	Regulator 3.3V voltage input
91	72	VREGOUT	P	Regulator 1.8V voltage output
92	73	DC1_FB0	I	NC
93	74	DC1_FB1	I	NC
94	75	DC0_FB0	I	NC
95	76	DC0_FB1	I	NC
96	77	NGND	G	Power GND
97	78	VBAT	P	NC
98	79	LXVCC	P	NC
99	80	VCCOUT	P	NC

## Rock2 DATA SHEET V1.1

100	81	DCSEL0	I	NC	
	82	DCSEL1	I	NC	
101	83	DCDCEN	I	NC	
102	84	VCCA	P	ADC 3.3V power input	
		VREF	I	ADC reference voltage input	
103	85	REXT100K	I	ADC reference Resistor input	
104	X	LRADC3	I	Low resolution ADC input3	
105	X	LRADC2	I	Low resolution ADC input2	
106	86	LRADC1	I	Low resolution ADC input1	
107	87	LRADC0	I	Low resolution ADC input0	
108	88	VDDPLL	P	Analog power of PLL	
109	89	VSSPLL	P	Analog GND of PLL	
110	90	XIN	I, OSC	Crystal 24MHz OSC input PAD	
111	91	XOUT	O, OSC	Crystal 24MHz QSC output PAD	
112	92	VDD	P	Digital Core power(1.8V)	
113	93	VSSD	G	Digital Ground	
114	94	VCC	P	I/O POWER(3.3V)	
115	95	RESET	I, Pull up	System reset pin, low enable	
116	96	TEST	I, Pull down	Test mode, ( Internal 100K pull down)	
117	X	D8/P3.0	I/O	Flash/ LCD/ SDRAM data bus bit 8 GPIO	
118	X	D9/P3.1	I/O	Flash/ LCD/ SDRAM data bus bit 9 GPIO	
119	X	D10/P3.2	I/O	Flash/ LCD/ SDRAM data bus bit 10 GPIO	
120	X	D11/P3.3	I/O	Flash/ LCD/ SDRAM data bus bit 11 GPIO	
121	X	D12/P3.4	I/O	Flash/ LCD/ SDRAM data bus bit 12 GPIO	
122	X	D13/P3.5	I/O	Flash/ LCD/ SDRAM data bus bit 13 GPIO	
123	X	D14/P3.6	I/O	Flash/ LCD/ SDRAM data bus bit 14 GPIO	
124	X	D15/P3.7	I/O	Flash/ LCD/ SDRAM data bus bit 15 GPIO	
125	97	LCDRDN	O	LCD Read execution pin	
126	98	LCDWRN	O	LCD Write execution pin	
127	99	LCDCSN	O	LCD driver chip select	
128	100	FCE0	O	Flash chip select 0	

## 3 Function Description

### 3.1 Block Diagram



ROCK!

## 3.2 DMA

Rock2 include 3 DMA channels.

DMA0 has a cache of 32bytes.

DMA1 has a cache of 16bytes.

DMA2 has a cache of 8bytes, which is adapted to transmitting data between RAM and CODEC.

Support for memory-to-memory, memory-to-peripheral, peripheral-to-memory, and peripheral-to-peripheral DMA transfers

Support of interrupt enabling and masking

### 3.2.1 Register

#### **SAR x:** Source Address Register for Channel x (x=0,1,2)

Bits	Name	Direction	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31:0	SAR	R/W	0x0	<b>Current Source Address of DMA transfer.</b> Updated after each source AMBA transfer. The SINC field in the <b>CTLx</b> register determines whether the address increments, decrements, or is left unchanged on every source AMBA transfer throughout the block transfer.

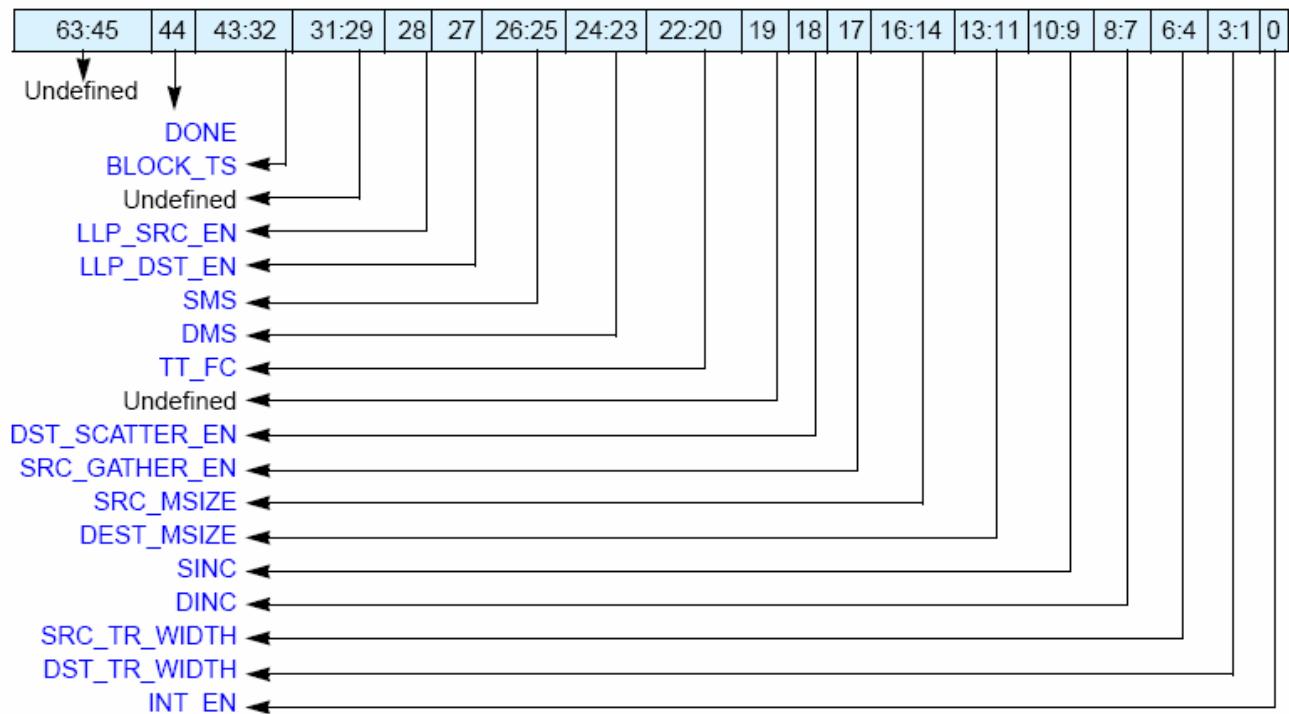
#### **DAR x:** Destination Address Register for Channel x (x=0,1,2)

Bits	Name	Direction	Reset	Description
63:32	Undefined	N/A	0x0	Reserved
31:0	DAR	R/W	0x0	<b>Current Destination address of DMA transfer.</b> Updated after each destination AMBA transfer. The DINC field in the <b>CTLx</b> register determines whether the address increments, decrements or is left unchanged on every destination AMBA transfer throughout the block transfer.

#### **CTL x:** Control Register for Channel x (x=0,1,2)

## Rock2 DATA SHEET V1.1

---



Bits	Name	R/W	Description
63:45	Undefined	N/A	Reserved
44	DONE	R/W	<p><b>Done bit</b>            If status write-back is enabled, the upper word of the control register, CTLx[63:32], is written to the control register location of the Linked List Item (LLI) in system memory at the end of the block transfer with the done bit set.</p> <p>Software can poll the LLI CTLx.DONE bit to see when a block transfer is complete. The LLI CTLx.DONE bit should be cleared when the linked lists are set up in memory prior to enabling the channel.</p> <p><b>Reset Value:</b>0x0</p>

## Rock2 DATA SHEET V1.1

b:32 (See description)	BLOCK_TS	R/W	<b>Block Transfer Size.</b> Writes this field before the channel is enabled in order to indicate the block size. The number programmed into BLOCK_TS indicates the total number of single transactions to perform for every block transfer; a single transaction is mapped to a single AMBA beat. <b>Width:</b> The width of the single transaction is determined by TLx.SRC_TR_WIDTH. <b>Reset Value:</b> 0x2
31:29	Undefined	N/A	Reserved
28	LLP_SRC_EN	R/W	Block chaining is enabled on the source side only if the LLP_SRC_EN field is high and LLPx.LOC is non-zero; <b>Reset Value:</b> 0x0
27	LLP_DST_EN	R/W	Block chaining is enabled on the destination side only if the LLP_DST_EN field is high and LLPx.LOC is non-zero. <b>Reset Value:</b> 0x0.
26:25	SMS	R/W	<b>Source Master Select.</b> Identifies the Master Interface layer from which the source device (peripheral or memory) is accessed. 00 = AHB master 1 01 = AHB master 2
24:23	DMS	R/W	<b>Destination Master Select.</b> Identifies the the Master Interface layer where the destination device (peripheral or memory) resides. 00 = AHB master 1 01 = AHB master 2
22:20	TT_FC	R/W	<b>Transfer Type and Flow Control.</b> The following transfer types are supported. • Memory to Memory • Memory to Peripheral • Peripheral to Memory • Peripheral to Peripheral
19	Undefined	N/A	Reserved
18	DST_SCATTER_EN	R/W	Destination scatter enable bit: 0 = Scatter disabled 1 = Scatter enabled Scatter on the destination side is applicable only when the CTLx.DINC bit indicates an incrementing or decrementing address control. <b>Reset Value:</b> 0x0
17	SRC_GATHER_EN	R/W	Source gather enable bit: 0 = Gather disabled 1 = Gather enabled Gather on the source side is applicable only when the CTLx.SINC bit indicates an incrementing or decrementing address control. <b>Reset Value:</b> 0x0
16:14	SRC_MSIZEx	R/W	<b>Source Burst Transaction Length.</b> Number of data items, each of width

## Rock2 DATA SHEET V1.1

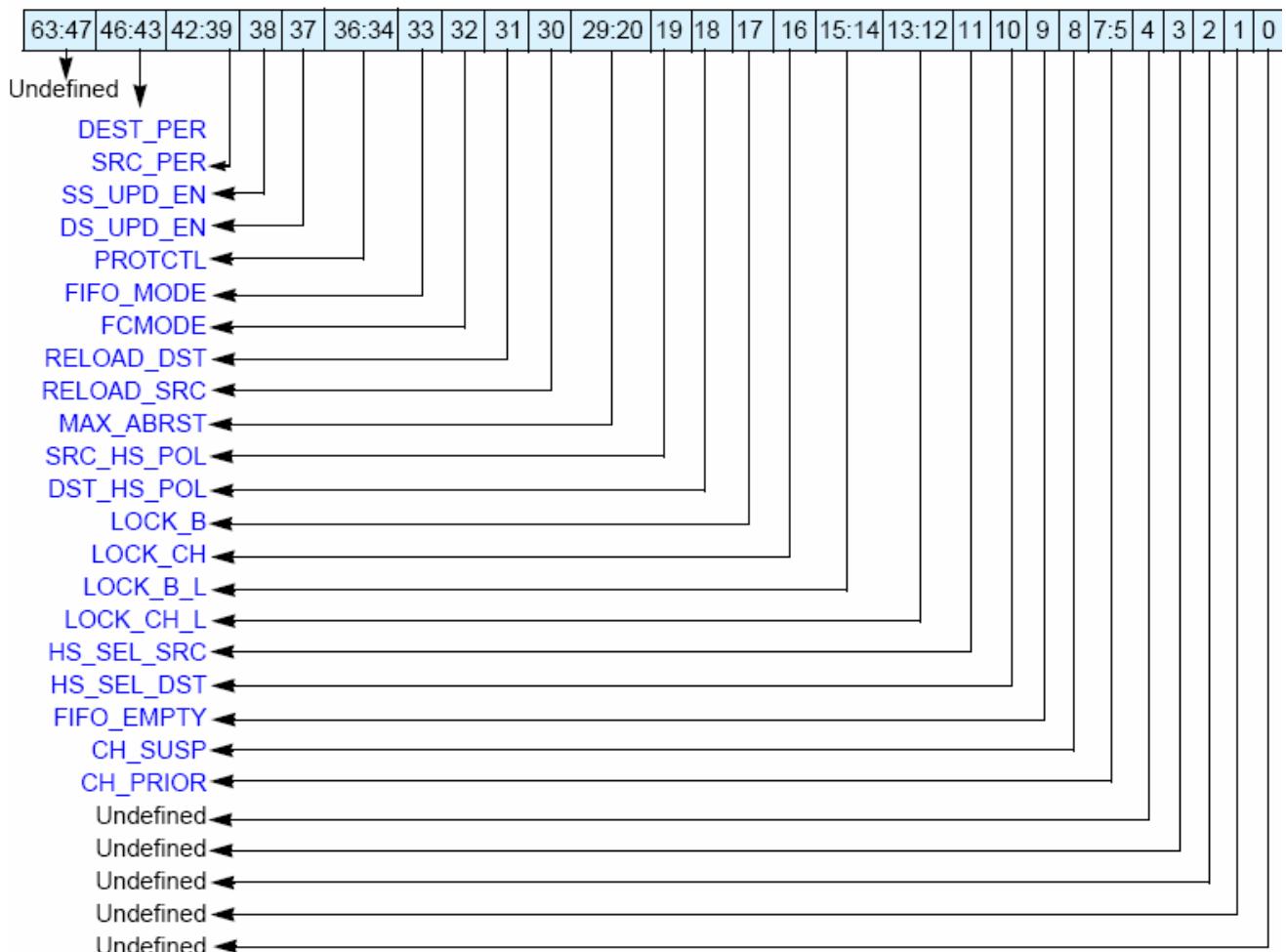
---

			CTLx.SRC_TR_WIDTH, to be read from the source every time a source burst transaction request is made from either the corresponding hardware or software handshaking interface. <b>Reset Value:</b> 0x1
13:11	DEST_MSIZEx	R/W	<b>Destination Burst Transaction Length.</b> Number of data items, each of width CTLx.DST_TR_WIDTH, to be written to the destination every time a destination burst transaction request is made from either the corresponding hardware or software handshaking interface. <b>Reset Value:</b> 0x1
10:9	SINC	R/W	<b>Source Address Increment.</b> Indicates whether to increment or decrement the source address on every source AMBA transfer. If the device is fetching data from a source peripheral FIFO with a fixed address, then set this field to “No change.” 00 = Increment 01 = Decrement 1x = No change <b>Reset Value:</b> 0x0
8:7	DINC	R/W	<b>Destination Address Increment.</b> Indicates whether to increment or decrement the destination address on every destination AMBA transfer. If your device is writing data to a destination peripheral FIFO with a fixed address, then set this field to “No change.” 00 = Increment 01 = Decrement 1x = No change <b>Reset Value:</b> 0x0
6:4	SRC_TR_WIDTH	R/W	<b>Source Transfer Width.</b> Mapped to AHB bus “hsize.” For a non-memory peripheral, typically the peripheral (source) FIFO width. This value must be less than or equal to DMAH_M <sub>x</sub> _HDATA_WIDTH, where <sub>x</sub> is the AMBA layer 1 to 4 where the source resides.
3:1	DST_TR_WIDTH	R/W	<b>Destination Transfer Width.</b> Mapped to AHB bus “hsize.” For a non-memory peripheral, typically rgw peripheral (destination) FIFO width. This value must be less than or equal to DMAH_M <sub>k</sub> _HDATA_WIDTH, where <sub>k</sub> is the AMBA layer 1 to 4 where the destination resides.
0	INT_EN	R/W	<b>Interrupt Enable Bit.</b> If set, then all interrupt-generating sources are enabled. <b>Reset Value:</b> 0x1

---

## Rock2 DATA SHEET V1.1

### CFG x: Configuration Register for Channel x (x=0,1,2)



Bits	Name	Direction	Reset	Description
63:47	Undefined	N/A	0x0	Reserved
46:43	DEST_PER	R/W	0x0	Assigns a hardware handshaking interface (0 -DMAH_NUM_HS_INT-1) to the destination of channel xif the CFGx.HS_SEL_DST field is 0. Otherwise, this field is ignored. The channel can then communicate with the destination peripheral connected to that interface via the assigned hardware handshaking interface.
42:39	SRC_PER	R/W	0x0	Assigns a hardware handshaking interface (0 -DMAH_NUM_HS_INT-1) to the source of channel xif the CFGx.HS_SEL_SRC field is

## Rock2 DATA SHEET V1.1

---

				0. Otherwise, this field is ignored. The channel can then communicate with the source peripheral connected to that interface via the assigned hardware handshaking interface.
38	SS_UPD_EN	R/W	0x0	<b>Source Status Update Enable.</b> Source status information is only fetched from the location pointed to by the <b>SSTATArx</b> register, stored in the <b>SSTATx</b> register and written out to the SSTATxlocation of the LLI if <i>SS_UPD_EN</i> is high.
37	DS_UPD_EN	R/W	0x0	<b>Destination Status Update Enable.</b>
36:34	PROTCTL	R/W	0x1	<b>Protection Control</b> bits used to drive the AMBA HPROT[3:1] bus.
33	FIFO_MODE	R/W	0x0	<b>FIFO Mode Select.</b> Determines how much space or data needs to be available in the FIFO before a burst transaction request is serviced. 0 = Space/data available for single AMBA transfer of the specified transfer width. 1 = Space/data available is greater than or equal to half the FIFO depth for destination transfers and less than half the FIFO depth for source transfers. The exceptions are at the end of a burst transaction request or at the end of a block transfer.
32	FCMODE	R/W	0x0	<b>Flow Control Mode.</b> Determines when source transaction requests are serviced when the Destination Peripheral is the flow controller. 0 = Source transaction requests are serviced when they occur. Data pre-fetching is enabled. 1 = Source transaction requests are not serviced until a destination transaction request occurs. In this mode the amount of data transferred from the source is limited such that it is guaranteed to be transferred to the destination prior to block termination by the destination. Data pre-fetching is disabled.
31	RELOAD_DST	R/W	0x0	<b>Automatic Destination Reload.</b> The <b>DARx</b> register can be automatically reloaded from its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated.
30	RELOAD_SRC	R/W	0x0	<b>Automatic Source Reload.</b> The <b>SARx</b> register can be automatically reloaded from

---

## Rock2 DATA SHEET V1.1

				its initial value at the end of every block for multi-block transfers. A new block transfer is then initiated.
29:20	MAX_ABRST	R/W	0x0	<b>Maximum AMBA Burst Length.</b>
19	SRC_HS_POL	R/W	0x0	<b>Source Handshaking Interface Polarity.</b> 0 = Active high 1 = Active low
18	DST_HS_POL	R/W	0x0	<b>Destination Handshaking Interface Polarity.</b> 0 = Active high 1 = Active low
17	LOCK_B	R/W	0x0	<b>Bus Lock Bit.</b>
16	LOCK_CH	R/W	0x0	<b>Channel Lock Bit.</b>
15:14	LOCK_B_L	R/W	0x0	<b>Bus Lock Level.</b>
13:12	LOCK_CH_L	R/W	0x0	<b>Channel Lock Level.</b>
11	HS_SEL_SRC	R/W	0x1	<b>Source Software or Hardware Handshaking Select.</b> This register selects which of the handshaking interfaces, hardware or software, is active for source requests on this channel. 0 = Hardware handshaking interface. Software-initiated transaction requests are ignored. 1 = Software handshaking interface. Hardware-initiated transaction requests are ignored. If the source peripheral is memory, then this bit is ignored.
10	HS_SEL_DST	R/W	0x1	<b>Destination Software or Hardware Handshaking Select.</b> This register selects which of the handshaking interfaces, hardware or software, is active for destination requests on this channel. 0 = Hardware handshaking interface. Software-initiated transaction requests are ignored. 1 = Software handshaking interface. Hardware Initiated transaction requests are ignored. If the destination peripheral is memory, then this bit is ignored.
9	FIFO_EMPTY	R	0x0	Indicates if there is data left in the channel's FIFO. Can be used in conjunction with CFGx.CH_SUSP to cleanly disable a channel. 1 = Channel's FIFO empty 0 = Channel's FIFO not empty
8	CH_SUSP	R/W	0x0	<b>Channel Suspend.</b> Suspends all DMA data transfers from the source until this bit is cleared. There is no guarantee that the current transaction will complete. Can also

## Rock2 DATA SHEET V1.1

---

				be used in conjunction with CFGx.FIFO_EMPTY to cleanly disable a channel without losing any data. 0 = Not Suspended. 1 = Suspend. Suspend DMA transfer from the source.
7:5	CH_PRIOR	R/W	Channel Number For example: Chan0=0 Chan1=1	<b>Channel priority.</b> A priority of 7 is the highest priority, and 0 is the lowest. This field must be programmed within the following range: 0: (DMAH_NUM_CHANNELS – 1) A programmed value outside this range will cause erroneous behavior.
4:0	Undefined	N/A	0x0	Reserved

### **SGR x:** Source Gather Register for Channel x

Bits	Name	Direction	Reset	Description
63:32	Undefined	N/A	0x0	Reserved.
b:20 (Refer to description)	SGC	R/W	0x0	Source gather count. Source contiguous transfer count between successive gather boundaries. $b = \log_2 (\text{DMAH\_CH}_x\text{\_MAX\_BLK\_SIZE} + 1) + 19$ Bits 31:b+1 do not exist and read back as 0.
19:0	SGI	R/W	0x0	Source gather interval.

### **DSR x:** Destination Scatter Register for Channel x

Bits	Name	Direction	Reset	Description
63:32	Undefined	N/A	0x0	Reserved.
b:20 (Refer to description)	DSC	R/W	0x0	Destination scatter count. Destination contiguous transfer count between successive scatter boundaries. $b = \log_2 (\text{DMAH\_CH}_x\text{\_MAX\_BLK\_SIZE} + 1) + 19$ Bits 31:b+1 do not exist and read 0.
19:0	DSI	R/W	0x0	Destination scatter interval.

## DmaCfgReg: DW\_ahb\_dmac Configuration Register

Bits	Name	Direction	Reset	Description
63:1	Undefined	N/A	0x0	Reserved
0	DMA_EN	R/W	0x0	<b>DW_ahb_dmac Enable bit.</b> 0 = DW_ahb_dmac Disabled 1 = DW_ahb_dmac Enabled.

## ChEnReg: DW\_ahb\_dmac Channel Enable Register

Bits	Name	Direction	Reset	Description
63:8+dnca	Undefined	N/A	0x0	Reserved
7+dnca:8	CH_EN_WE	W	0x0	Channel enable write enable.
7:dnca, b	Undefined	N/A	0x0	Reserved
dnca-1:0	CH_EN	R/W	0x0	Enables/Disables the channel. Setting this bit enables a channel, clearing this bit disables the channel. 0 = Disable the Channel 1 = Enable the Channel The ChEnReg.CH_EN bit is automatically cleared by hardware to disable the channel after the last AMBA transfer of the DMA transfer to the destination has completed. Software can therefore poll this bit to determine when this channel is free for a new DMA transfer.

### 3.2.2 Example Software

```

_init_dmac2_codec_dac: //channel 2
    mov r6, DMAR_SAR2
    mov r5, 0x00
    lda r4, data_table      !_PCM_Buffer0
    shll.e r4, 1
    add r5, 0x02            ! 04 as DRAM, 02,03 as IRAM
    stdu r4, r6, 2

    mov r6, DMAR_DAR2
    mov r5, 0x0
    mov r4, I2DSP_TXDB ! Transmit to CODEC
    shll.e r4, 1
    stdu r4, r6, 2

    mov r6, DMAR_CTL2
    mov r5, 0x10          ! Memory to Peripheral
    mov r4, 0x125         ! int enable
    stdu r4, r6, 2

    mov r5, 0x0
    mov r4, 2050          ! data length
    stdu r4, r6, 2

    mov r6, DMAR_MaskBlock
    mov r4, 0x404
    st r4, r6

    mov r6, DMAR_CFG2      ! CFG is 64bits width
    mov r4, 0x40
    mov r5, 0x0
    stdu r4, r6, 2
    mov r4, 0x1004        ! dist req 2: txreq, [46:43]
    stdu r4, r6, 2

    mov r6, DMAR_DmaCfgReg
    mov r4, 0x1            ! enable DMA
    st r4, r6

    mov r6, DMAR_ChEnReg
    mov r4, 0x0404         ! enable channel 2
    st r4, r6
    ret

_init_dmac1_codec_adc: //channel 1
    mov r6, DMAR_SAR1
    mov r5, 0x0
    mov r4, I2DSP_RXDB ! Read data from CODEC
    shll.e r4, 1
    stdu r4, r6, 2

    mov r6, DMAR_DAR1

```

## Rock2 DATA SHEET V1.1

```
mov r5, 0x00
lda r4, data_table1 ! data_table
shll.e r4, 1
add r5, 0x04           !04 as DRAM, 02,03 as IRAM
stdu r4, r6, 2

mov r6, DMAR_CTL1
mov r5, 0x20           ! Peripheral to Memory
mov r4, 0x425          ! int enable
stdu r4, r6, 2

mov r5, 0x0
mov r4, 2050           ! length
stdu r4, r6, 2

mov r6, DMAR_MaskBlock
mov r4, 0x202
st r4, r6

mov r6, DMAR_CFG1
mov r4, 0x40
mov r5, 0x0
stdu r4, r6, 2
mov r4, 0x0184         ! source Req 3: rxreq, [42:39]
stdu r4, r6, 2

mov r6, DMAR_DmaCfgReg
mov r4, 0x1             ! enable DMA
st r4, r6

mov r6, DMAR_ChEnReg
mov r4, 0x202           ! enable channel 1
st r4, r6
ret

init_dmac: //chanel 0
    mov r11, DMAR_SAR0
    mov r1, 0x04
    mov r0, 0x6000
    stdu r0, r11, 2

    mov r11, DMAR_DAR0
    mov r1, 0x10
    mov r0, 0x0000
    stdu r0, r11, 2

    mov r11, DMAR_CTL0
    mov r1, 0x00
    mov r0, 0x24
    stdu r0, r11, 2
    mov r1, 0x0
    mov r0, length
    stdu r0, r11, 2
```

## Rock2 DATA SHEET V1.1

---

```
mov r11, DMAR_CFG0  
mov r0, 0x0  
mov r1, 0x0  
std u r0, r11, 2  
mov r0, 0x04  
std u r0, r11, 2  
  
mov r11, DMAR_DmaCfgReg  
mov r0, 0x1  
st r0, r11  
  
mov r11, DMAR_ChEnReg  
mov r0, 0x101  
st r0, r11
```

ROCKCHIP

### **3.3 USB PHY and Controller**

Compliant with USB2.0 and USB1.1 specification  
Supports HS (480Mbps) / FS (12Mbps) modes  
Supports USB suspend state.



## 3.4 SDRAM Controller

Supports up to 13 SDRAM address bits

SDRAM memory data width is 16

Supports 2K to 4K rows, 256 to 1K columns, and 2 to 4 banks

Supports auto refresh with programmable refresh intervals

Supports self-refresh

Supports SDRAM power-down mode

### 3.4.1 Registers

#### SDRAM Config Register (SCONR)

Bits	Name	Default	Description
31:21	Unused		
20	s_sda_oe_n	1	rev
19	s_sd	0	rev
18	s_scl	1	rev
17:15	s_sa	0	rev
14:13	s_data_width	0	rev
12:9	s_col_addr_width	1000	Number of address bits for column address: 15 – reserved 7-14 – correspond to 8-15 bits 0-6 – reserved
8:5	s_row_addr_width	1011	Number of address bits for row address: 10-15 – correspond to 11-16 bits 0-10 – reserved
4:3	s_bank_addr_width	1	Number of bank address bits; values of 0-3 correspond to 1-4 bits, and therefore select 2-16 banks
2:0	Unused		

#### SDRAM Timing Registers

The STMG0R and STMG1R registers hold the SDRAM timing parameters;

The SDRAM controller uses the CAS latency value during the initialization sequence in order to program the mode register of the SDRAM. The user can also specifically force SDRAM controller to do a mode register update by programming the set\_mode\_reg bit (bit 9 of SCTRLR). If you want to change the value of CAS latency during normal operation, you should first program the STMG0R timing register, and then program bit 9 of SCTRLR to 1. SDRAM controller will reset this bit once it has updated the mode register.

#### SDRAM Timing Register0 (STMG0R)

Bits	Name	Default	Description
25:22	t_rc	T_RC - 1 0110b	Active-to-active command period; values of 0-15 correspond to t_rc of 1-16 clocks.

## Rock2 DATA SHEET V1.1

31:27	extended_t_xsr	T_XSR – 1 00000b	Exit self-refresh to active or auto-refresh command time; minimum time controller should wait after taking SDRAM out of self-refresh mode before issuing any active or auto-refresh commands; values 0-511 correspond to t_xsr of 1-512 clocks
21:18	t_xsr	0111b	
17:14	t_rcar	T_RCAR–1 0110b	Auto-refresh period; minimum time between two auto-refresh commands; values 0-15 correspond to t_rcar of 1-16 clocks.
13:12	t_wr	T_WR – 1 01b	For writes, delay from last data in to next precharge command; values 0-3 correspond to t_wr of 1-4 clocks
11:9	t_rp	T_RP – 1 010b	Precharge period; values of 0-7 correspond to t_rp of 1-8 clocks
8:6	t_rcd	T_RCD – 1 001b	Minimum delay between active and read/write commands; values 0-7 correspond to t_rcd values of 1-8 clocks
5:2	t_ras_min	T_RAS_MIN – 1 0100b	Minimum delay between active and precharge commands; values of 0-15 correspond to T_RAS_MIN of 1-16 clocks
26	extended_cas_latency	CAS_LATENCY-1 001b	Delay in clock cycles between read command and availability of first data 0 – 1 clock    1 – 2 clocks 2 – 3 clocks    3 – 4 clocks 4,5,6, 7 – reserved
1:0	cas_latency		

### SDRAM Timing Register1 (STMG1R)

Bits	Name	Default	Description
31:22	Reserved		
21:20	t_wtr	T_WTR-1 00b	Rev for DDR
19:16	num_init_ref	NUM_INIT_REF - 1 0111b	Number of auto-refreshes during initialization; values 0-15 correspond to 1-16 auto-refreshes
15:0	t_init	T_INIT 0x0008	Number of clock cycles to hold SDRAM inputs stable after power up, before issuing any commands.

### SDRAM Control Registers

You can program SDRAM control registers at any time after power-up. However, the SDRAM controller does not poll the registers until the SDRAM controller finishes current and pending SDRAM accesses in the write FIFO.

### SDRAM Control Register (SCTRLR)

Bits	Name	Default	Description
31:18	Reserved		
17	s_rd_ready_mode	0	SDRAM read-data-ready mode; set to 1, indicates SDRAM read data is sampled after s_rd_ready goes active.
16:12	num_open_banks	2	Number of SDRAM internal banks to be open at any time; values of 0-15 correspond to 0-15 banks open

## Rock2 DATA SHEET V1.1

---

11	self_refresh_status	0	Read only. When “1,” indicates SDRAM is in self-refresh mode. When “self_refresh/deep_power_mode” bit (bit 1 of SCTRLR) is set, it may take some time before SDRAM is put into self-refresh mode, depending on whether all rows or one row are refreshed before entering self-refresh mode defined by full_refresh_before_sr bit Before gating clock in self-refresh mode, ensure this bit is set
10	sync_flash_soft_seq	0	rev
9	set_mode_reg	0	Set to 1, forces controller to do update of SDRAM mode register; bit is cleared by controller once it has finished mode register update
8:6	read_pipe	2	Indicates number of registers inserted in read data path for SDRAM in order to correctly latch data; values 0-7 indicate 0-7 registers
5	full_refresh_after_sr	0	Controls number of refreshes done by SDRAM controller after it is taken out of self-refresh mode: 1 – Refresh all rows before entering self-refresh mode 0 – Refresh just 1 row before entering self-refresh mode
4	full_refresh_before_sr	0	Controls number of refreshes done before putting SDRAM into self-refresh mode: 1 – Refresh all rows before entering self-refresh mode 0 – Refresh just one row before entering self-refresh mode
3	precharge_algorithm	1	Determines when row is precharged: 0 – Immediate precharge; row precharged at end of read/write operation 1 – Delayed precharge; row kept open after read/write operations
2	power_down_mode	0	Forces DW_memctl to put SDRAM in power-down mode
1	self_refresh/deep_power_mode	0	Forces DW_memctl to put SDRAM in self-refresh mode. Bit can be cleared by writing to this bit or by clear_sf_dp pin, generated by external power management unit
0	initialize	0	Forces DW_memctl to initialize SDRAM; bit reset to 0 by SDRAM controller once initialization sequence is complete

### SDRAM Refresh Interval Register (SREFR)

Bits	Name	Default	Description
31:24	gpi	-	Rev
23:16	gpo	0	Rev
15:0	t_ref	100 clocks, assuming a refresh period of 7.8us, at system frequency of 12Mhz	Number of clock cycles between consecutive refresh cycles;

### Address Mask Registers (SMSKR0)

Bits	Name	Default	Description
31:11	Reserved		

## Rock2 DATA SHEET V1.1

10:8	reg_select	REG_SELECT <sub>n</sub> 000b	Register determines which timing parameters of memory connect to associated chip select; primarily used for specifying static memories 0 – register set 0 1 – register set 1 2 – register set 2
7:5	mem_type	CHIP_SELECT <sub>n</sub> _MEM 000b	Type of memory connected to corresponding chip select: 0 – SDRAM, Others – Reserved
4:0	mem_size	BLOCK_SIZE <sub>n</sub> 0x0B	Size of memory connected to corresponding chip select; 0 – No memory is connected to the chip select 1 – 64KB 2 – 128KB 3 – 256KB 4 – 512KB 5 – 1MB 6 – 2MB 7 – 4MB 8 – 8MB 9 – 16MB 10 – 32MB 11 – 64MB 12 – 128MB 13 – 256MB 14 – 512MB 15 – 1GB 16 – 2GB 17 – 4GB

### 3.4.2 Programming Sequence

Program the SDRAM controller registers using the following sequence:

- SDRAM Mask Registers (SMSKR)
- Program Mask Register
- SDRAM bank address, row/column address and data width (SCONR)
- Program SDRAM Configuration Register
- SDRAM timing parameters (STMG0R)
- Program SDRAM Timing Register0
- The remaining SDRAM timing parameters (STMG1R)
- Program SDRAM Timing Register1
- Refresh period (SREFR)
- Program SDRAM Refresh Interval register
- SDRAM control Register (SCTRLR). Set the "initialize" bit for the controller
- to do auto SDRAM initialization sequence. Since setting the "initialize" bit starts the SDRAM initialization sequence, this register should be programmed last
- Program SDRAM Control Register

### 3.4.3 SDRAM Controller Functional

#### Power-On Initialization

The SDRAM controller follows the JEDEC-recommended SDR-SDRAM power-on initialization sequence as follows:

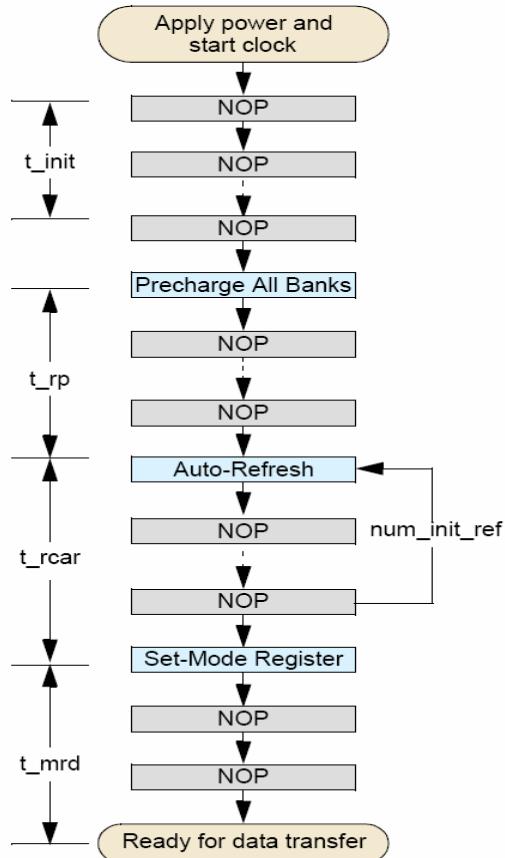
1. Apply power and start clock; maintain a NOP condition at the inputs
2. Maintain stable power, stable clock, and NOP input conditions for a minimum of  $t_{init}$  clock cycles
3. Issue precharge commands for all banks of the device
4. Issue auto-refresh commands, depending on the value num\_init\_ref in the programmable register
5. Issue a set-mode register command to initialize the mode register

The SDRAM controller performs a power-on sequence of the SDRAM under these circumstances:

- Immediately after reset
- When the programmable initialize bit (bit 0 of SCTRLR) is set, the SDRAM controller resets the bit when it comes out of initialization.

All SDRAM read/write requests that occur during initialization are queued in the memory controller.

Figure 16 illustrates the commands issued to the SDRAM by the controller during the power-on initialization.



**Figure 16: SDR-SDRAM Power-On Command Sequence**

The T\_INIT, T\_RP, and T\_RCAR compile-time parameters are the default values for t\_init, t\_rp and t\_rcar, respectively, which you can use for initialization. If you feel that the reset time of the system is long enough to take care of the t\_init time, then you can assign a value of zero to the T\_INIT parameter.

The SDRAM controller initializes the SDRAM after reset using the specified default compile-time timing parameters. After reset, if you feel that these timing parameters are not adequate, then you can program the timing parameters accordingly and then program the initialize bit – that is, bit 0 of SCTRLR – to 1. This forces the SDRAM controller to initialize the SDRAM.

The t\_mrd is fixed at a value of 3 clock cycles, according to the JEDEC standard.

## Set-Mode Register

The DW\_memctl automatically sets the SDR-SDRAM mode register and extended-mode register (extended-mode register is only for Mobile-SDRAM) during the power-up initialization. During normal operation, if you want to set the mode register or extended-mode register, you need to set set\_mode\_reg (bit 9 of SCTRLR) in the control register (SCTRLR) to 1. After the memory controller finishes the mode register setting, it clears the set\_mode\_reg bit to 0.

The “burst length” field and the “burst type” field of the SDR-SDRAM-mode register are fixed by the DW\_memctl to “010” (burst length 4) and “0” (sequential burst), respectively. The DW\_memctl programs the “CAS latency” field and the “operating mode” field of the mode register according to the values provided by the user in the control and timing registers.

## Read/Write Operations

The DW\_memctl converts all AHB bursts to 4-word bursts on the SDRAM side. The memory bursts are concatenated to achieve continuous data flow for long AHB bursts. You can terminate the memory read/write

## Rock2 DATA SHEET V1.1

burst with either a precharge command or terminate command, depending on which precharge mode – immediate precharge or delayed precharge – that you program. You can also terminate the write burst with a Sub sequent write burst.

The DW\_memctl does not use auto-precharge mode.

The DW\_memctl supports two precharge modes – immediate precharge and delayed precharge. If you program for an immediate precharge mode, then the DW\_memctl closes the open row after a read or write access. If you program for a delayed precharge mode, then the DW\_memctl keeps the row open after an access. The DW\_memctl can keep multiple banks open at the same time, depending on the value of num\_open\_bank in the programmable register. When the number of open banks reaches the num\_open\_bank and an access to a new bank comes, the DW\_memctl will close the oldest bank (the bank opened first) before opening the new bank.

Example of Register setting:

SDRAM Register Setup (k4s640832d)  
32`h0001bc54, SMSKR0 = 32`h00b  
32`h0001bc00, SCONR = 32`h1c1168  
32`h0001bc04, STMG0R = 32`h19d9451 (or 32`h1ddd696?)  
32`h0001bc08, STMG1R = 32`h70008  
32`h0001bc10, SREFR = 32`h3e8 (for 100MHz clock)  
32`h0001bc0c, SCTRLR = 32`h9 (or 20009)

### 3.4.4 Example Software

init\_sdram:

```
    mov r0, mempcr
    mov r1, 0x03
    st r1, r0

    mov r0, MEMCTL_SCONR
    mov r2, 0x1168
    mov r3, 0x1c
    std u r2, r0, 2

    mov r0, MEMCTL_STMG0R
    mov r2, 0x9451
    mov r3, 0x19d
    std u r2, r0, 2

    mov r0, MEMCTL_STMG1R
    mov r2, 0x08
    mov r3, 0x7
    std u r2, r0, 2

    mov r0, MEMCTL_SREFR
    mov r2, 0x3e8
    mov r3, 0x0
    std u r2, r0, 2

    mov r0, MEMCTL_SCTRLR
    mov r2, 0x2009
    mov r3, 0x0
    std u r2, r0, 2
```

## 3.5 Flash / Video Controller

The EX\_MEMCTL is an AHB slave in ROCK2 and provides all the functionality for read/write transactions from multiple AHB masters to the off-chip memory device. Asynchronous interface is supported.

The EX\_MEMCTL include a register file, an ECC calculator and external interface. The external interface accessible space is logically segmented into LCD interface region, flash/IDE interface region, CF card interface region.

LCD interface have 256 Word space corresponds a chip select named LCDCS.

Flash and IDE interface have 256Word space which split into two chip selects named FMCS0, FMCS1.

CF card interface have 2KWord space corresponds a chip select named CFCS.

The EX\_MEMCTL also include a store buffer to help prevent pipeline stalls during stores to external memory.

The EX\_MEMCTL interfaces share address, data and write/read control pins, except Flash/IDE interface have a set of indicate control pins which include ALE, CLE, WP, FWR, FRD, RDY.

- Asynchronous interface is supported.
- ECC calculator supports 8bit and 16bit external device
- Supports four chip select line: FMCS0, FMCS1, CFCS, LCDCS
- Supports up to 11 address line
- The wait cycles inserted can be programmed or corresponded to the hardware input control signal-RDY

The external interface signals are:

**madder[13:0] External Memory Address Bus                      Output**

This bus contains the address for external memory accesses. During internal memory accesses, the madder[13:0] pins retain their previous state.

The actual address signal output is the alternative of madder[13:0] and sram s\_addr[ba1,ba0,a11:0].

**rdata[15:0] External Memory interface input Data Bus                      Input**

This bus is the data bus for external memory and external memory-mapped peripheral read.

**wdata[15:0] External Memory interface output Data Bus                      Output**

This bus is the data bus for external memory and external memory-mapped peripherals write. The actual data signal output is the alternative of wdata[15:0] and sram s\_wdata[15:0].

**epcsn Data Memory Chip Select                      Output**

The chip selects enable off-chip expansion for memory space corresponding to EPROM region. Rock2 asserts the chip select for the duration of the external access based upon the address space being accessed. The values set in the uwait registers determine the timing of these signals. CFCSn timing is relative to the RDN and WRN strobes.

**fmcs[1:0]n Memory-Mapped Peripheral Chip Selects                      Output**

These chip selects provide decoded selects for memory-mapped peripherals residing in external memory space. Rock2 asserts the appropriate chip select for the duration of the external access based upon the address space being accessed. The values set in the fmwait registers determine the timing of these signals. FMCS[1:0]N timing is relative to the FMRDN and FMWRN strobes.

**lcdcsn Memory-Mapped Peripheral Chip Select                      Output**

## Rock2 DATA SHEET V1.1

---

These chip selects provide decoded selects for memory-mapped peripherals residing in external memory space. Rock2 asserts the appropriate chip select for the duration of the external access based upon the address space being accessed. The values set in the lcdwait registers determine the timing of these signals. LCDCSn timing is relative to the RDN and WRN strobes.

**wrn Write Strobe**

**Output**

Rock2 asserts this active-LOW write strobe for external memory writes normally. The WRN signal is an invert version when both CFCSn asserts and the bit wrp in SYSCTL register is set to "1".

**rdn Write Strobe**

**Output**

Rock2 asserts this read strobe during external memory reads.

**fmwrrn Write Strobe**

**Output**

Rock2 asserts this active-LOW write strobe for external peripherals flash segment writes.

**fmrdrn Write Strobe**

**Output**

Rock2 asserts this read strobe during external peripherals flash segment reads.

**fmcle Flash Command Latch Enable**

**Output**

The FMCLE output signal is connected to NAND flash memory device CLE input. This signal is assert active-HIGH during an operation write to external peripherals flash segment base address +02.

**fmale Flash Address Latch Enable**

**Output**

The FMALE output signal is connected to NAND flash memory device ALE input. This signal is assert active-HIGH during an operation write to external peripherals flash segment base address +04.

**fmwp Flash Write Protect**

**Output**

The FMWP output signal is connected to NAND flash memory device WPN input. This signal active-LOW provides inadvertent write/erase protect during power transitions.

**rdy Hardwire Wait State**

**Pull-up Input**

Rock2 samples this signal and then appropriately set the bit rdy "1" or "0" in the FMST register.

**lcdrs LCD Register Select**

**Output**

The LCDRS output signal indicate a operation to the lcd register. This signal is assert active-HIGH during accesses to external peripherals lcd segment base address +02. The signal also assert even EX\_CTL accept a external memory operation at address over than 0x02\_8000.

**m\_dout\_valid Data Write Out Valid**

**Output**

Ex\_memctl valid signal for write data to external memory; decides direction of data flow All bits are identical; one bit per byte of data provided to improve drive strength, routing, and timing closure

**muxsel MUX Select**

**Output**

Address signal MUX and write data signal MUX select input.

**Registers:**

Register	Description	Address	R/W	Default
ECC0	ECC RESULT0	+00	R	_
ECC1	ECC_RESULT1	+04	R	_
ECC2	ECC_RESULT2	+08	R	_
ECC3	ECC_RESULT3	+0C	R	_
ECCCTL	ECC CONTROL REGISTER	+10	W	
ECCST	ECC STATUS REGISTER	+14	R	
FMCTL	FLASH CHIP SELECT CONTROL REGISTER	+18	W	
FMST	FLASH MEMORY STATUS READY INDICATE	+1C	R	
CFWAIT	CF CARD MEMORY REGION WAIT STATE	+20	R/W	
FMWAIT	FLASH MEMORY WAIT STATE	+24	R/W	
LCDWAIT	LCD INTERFACE WAIT STATE	+28	R/W	
SYSCTL	SYSTEM CONTROL REGISTER	+2C	W	

# Rock2 DATA SHEET V1.1

---

## ECC result register

32	24	16	8	0
Res		ECC0		

Res Reserved

This bits are reserved

ECC0 ECC result 0

The current ECC result

32	24	16	8	0
Res		ECC1		

Res Reserved

This bits are reserved

ECC1 ECC result 1

The 1st backuped ECC result

32	24	16	8	0
Res		ECC2		

Res Reserved

This bits are reserved

ECC2 ECC result 2

The 2nd backuped ECC result

32	24	16	8	0
Res		ECC3		

Res Reserved

This bits are reserved

ECC3 ECC result 3

The 1th backuped ECC result

After 512byte data had complete, the content of ECC2 is copied to ECC3, ECC1 is copied to ECC2 and ECC0 is copied to ECC1, and then the result of ECC will save into the register ECC0.

## ECC control register

32	13	12	11	8	7	4	3	2	1	0
Res	region	addrh		addrl		epd	RDn	x16	erst	

Res Reserved

This bits are reserved

Region Select the ECC active region

00: CF CARD region

01: Flash memory region

10: LCD device region

11: Register file region

Addrh(l) Select the ECC active range

The ECC should be activing when access in range addrl to addrh.

Epd ECC power down

The ECC will power down when this bit is set

RDn indicate data flow direction

0:data direction as output

1: data direction as input

x16 Data bus width

0:data width is 16

1:data width is 8

# Rock2 DATA SHEET V1.1

---

**erst** ECC reset

This bit resets the ECC. When set, the ECC clear all the result and then operate restart and this bit will auto cleared.

## ECC status register


**Res** Reserved

This bits are reserved

**erdy** Flash ready indicate

0:ECC is busy

1:ECC is ready

## Flash memory control register


**Res** Reserved

This bits are reserved

**Fcs0** Flash memory chip select control

1:hold flash memory chip select activity

0:flash memory chip select activity free

**fsel** Flash memory chip select PIN selection

1:select chip1 (PIN FMCS1N)

0:select chip0 (PIN FMCS0N)

## Flash memory status register


**Res** Reserved

This bits are reserved

**Rdy** Flash ready indicate

0:flash is busy

1:flash is ready

This bit is the sample of PIN FMRDY.

**Wp** Flash write protect

0:flash program/erase disable

1:flash program/erase enable

This bit is out put to the PIN FMWP

## EPWait state register


**rdy** Hardware handshaking controller bit

when this bit is set to "1", an external device asserts signal "RDY" to extend a wait-state access and the rest bits in this register will be ignored.

**csrw** Chip Select to R/W Strobe Leading Edge

This field specifies the number of processor clock cycles from the falling edge of CFCSn to the falling edge of RDN or WRn. If this bit is set to 0x0, the processor uses a csrw value of 0x1.

**rwpw** R/W Pulse Width

This field controls the width of RDN or WRn in processor clock cycles. If this bit is set to 0x0, the processor uses an rwpw value of 0x1. rwcs R/W Strobe to Enable Deassertion [4:0] This field specifies the number of processor clock cycles from the rising edge of RDN or WRn to the rising edge of CFCSn. If this bit is set to 0x0, the processor uses an rwcs value of 0x1.

**rwcs** R/W Strobe to Enable Deassertion

## Rock2 DATA SHEET V1.1

---

This field specifies the number of processor clock cycles from the rising edge of RDN or WRN to the rising edge of CFCSn. If this bit is set to 0x0, the processor uses an rwcs value of 0x1.

### **FMWait state register**

32	15	12	11	10	5	4	0
Res		csrw	rdy	rwpw		rwcs	

- rdy Hardware handshaking controller bit  
when this bit is set to “1”, an external device asserts signal “RDY” to extend a wait-state access and the rest bits in this register will be ignored.
- csrw Chip Select to R/W Strobe Leading Edge  
This field specifies the number of processor clock cycles from the falling edge of FMCS[1:0]N to the falling edge of FMRDN or FMWRN. If this bit is set to 0x0, the processor uses a csrw value of 0x1.
- rwpw R/W Pulse Width  
This field controls the width of FMRDN or FMWRN in processor clock cycles. If this bit is set to 0x0, the processor uses an rwpw value of 0x1. rwcs R/W Strobe to Enable Deassertion [4:0] This field specifies the number of processor clock cycles from the rising edge of FMRDN or FMWRN to the rising edge of FMCS[1:0]N. If this bit is set to 0x0, the processor uses an rwcs value of 0x1.
- rwcs R/W Strobe to Enable Deassertion  
This field specifies the number of processor clock cycles from the rising edge of FMRDN or FMWRN to the rising edge of FMCS[1:0]N. If this bit is set to 0x0, the processor uses an rwcs value of 0x1.

### **LCDWait state register**

32	15	12	11	10	5	4	0
Res		csrw	rdy	rwpw		rwcs	

- rdy Hardware handshaking controller bit  
when this bit is set to “1”, an external device asserts signal “RDY” to extend a wait-state access and the rest bits in this register will be ignored.
- csrw Chip Select to R/W Strobe Leading Edge  
This field specifies the number of processor clock cycles from the falling edge of LCDCSn to the falling edge of RDN or WRN. If this bit is set to 0x0, the processor uses a csrw value of 0x1.
- rwpw R/W Pulse Width  
This field controls the width of RDN or WRN in processor clock cycles. If this bit is set to 0x0, the processor uses an rwpw value of 0x1. rwcs R/W Strobe to Enable Deassertion [4:0] This field specifies the number of processor clock cycles from the rising edge of RDN or WRN to the rising edge of LCDCSn. If this bit is set to 0x0, the processor uses an rwcs value of 0x1.
- rwcs R/W Strobe to Enable Deassertion  
This field specifies the number of processor clock cycles from the rising edge of RDN or WRN to the rising edge of LCDCSn. If this bit is set to 0x0, the processor uses an rwcs value of 0x1.

### **System control register**

32	3	2	1	0	
res		fmwRP	wRP	sisel	

- Res Reserved  
This bits are reserved
- Sisel Sdram interface select  
0:EX\_MEMCTL external interface signal assert  
1:DW\_MEMCTL sdram interface signal assert  
This bit control the output level of selsdram. Signal selsdram control the output address bus MUX and data bus MUX.
- Wrp WRN and RDN signal polarity  
0:WRN and RDN signal normal output  
1:invert WRN and RDN signal  
This bit specifies the polarity for the WRN and RDN signal.
- fmwRP FMWRN and FMRDN signal polarity  
0:WRN and FMRDN signal normal output

## Rock2 DATA SHEET V1.1

---

1:invert FMWRN and FMRDN signal

This bit specifies the polarity for the WRN and RDN signal.

Rockchip

## 3.6 WDT

### 3.6.1 Registers

**WDT\_CR: Control Register**

Bits	Name	Direction	Description
31:5	N/A	N/A	Reserved and read as zero (0).
4:2	RPL	R/W	Reset pulse length. Writes have no effect as WDT_HC_RPL is 1, these bits are read-only, which is equals to 011. The Reset pulse length is 16 pclk cycles.
1	RMOD	R/W	Response mode. Selects the output response generated to a timeout. 0 = Generate a system reset. 1 = First generate an interrupt and if it is not cleared by the time a second timeout occurs then generate a system reset.
0	WDT_EN	R/W	WDT enable. This bit is used to enable and disable the wdt. When disabled, the counter does not decrement. Thus, no interrupts or system resets are generated. Once this bit has been enabled, it can be cleared only by a system reset. 0 = WDT disabled. 1 = WDT enabled. <b>Reset Value:</b> 0

**WDT\_TORR: Timeout Range Register**

Bits	Name	Dir.	Description
31:8	N/A	N/A	Reserved and read as zero (0).
7:4	TOP_INIT	R/W	Timeout period for initialization. <b>Reset Value:</b> fixed at zero.
3:0	TOP	R/W	Timeout period. (Set to 0x07 for normal use, verify for 0x0) The range of values is limited by 7. The range of values available for a 32-bit watchdog counter are: Where $i = \text{TOP}$ and $t = \text{timeout period}$ For $i = 0$ to 7 $t = 2(16 + i)$ <b>Reset Value:</b> 7

## Rock2 DATA SHEET V1.1

---

### **WDT\_CCVR: Current Counter Value Register**

Bits	Dir.	Description
22:0	R	This register, when read, is the current value of the internal counter. <b>Reset Value:</b> 7f ffff

### **WDT\_CRR: Counter Restart Register**

Bits	Direction	Description
31:8	N/A	Reserved and read as zero.
7:0	W	This register is used to restart the WDT counter. As a safety feature to prevent accidental restarts, the value <b>0x76</b> must be written. A restart also clears the WDT interrupt. Reading this register returns zero. <b>Reset Value:</b> 0

### **WDT\_STAT: Interrupt Status Register**

Bits	Direction	Description
31:1	N/A	Reserved and read as zero.
0	R	This register shows the interrupt status of the WDT. 1 = Interrupt is active regardless of polarity. 0 = Interrupt is inactive. <b>Reset Value:</b> 0

### **WDT\_EOI: Interrupt Clear Register**

Bits	Direction	Description
31:1	N/A	Reserved and read as zero.
0	R	Clears the watchdog interrupt. This can be used to clear the interrupt without restarting the watchdog counter. <b>Reset Value:</b> 0

#### **3.6.2 Operation Flow**

- 1 Select required timeout period at WDT\_TORR.
- 2 Set reset pulse length, response mode, and enable WDT at WDT\_CR.
- 3 Write 0x76 to WDT\_CRR(Counter Restart Register).
- 4 Starts back to selected timeout period.
- 5 Can clear by reading WDT\_EOI (Interrupt Clear Register) or restarting (kicking) the counter by writing 0x76 to WDT\_CRR (Counter Restart Register).

### 3.6.3 Example Software

```
mov r0, WDT_TORR      ! Normal use: 0x7, test it: 0x0  
mov r2, 0x7             !  $2^{16+7} = 8,388,608$  cycles generate wdt interrupt,  
st  r2, r0              !  $2^{16+8} = 16,777,216$  cycles generate wdt reset
```

```
mov r0, WDT_CR         ! [1]: first generate an wdt interrupt; [0]:enable  
mov r2, 0x0f  
st  r2, r0
```

```
mov r0, WDT_CRR        ! 0x76, clear the counter  
mov r2, 0x76  
st  r2, r0
```

..... (a period of program)

```
mov r0, WDT_CRR        ! 0x76, clear the counter  
mov r2, 0x76  
st  r2, r0
```

## 3.7 CLOCK GENERATOR

### 3.7.1 Overview

- Input frequency : 24 MHz.
- Programmable frequency divider
- Generate the DSP frequency MCLK, frequency range : 12MHz – 80MHz.
- Generate the CODEC frequency I2SMCLK, 12MHz
- Generate the USB 12MHz Clock, AHB clock, APB clock, SD/MMC card clock ,ADC clock, PWM clock

*MCLK, HCLK, PCLK要时序同步，其它CLOCK没有要求。*

最高工作频率：

PLLVCO (CLK\_OUT): 360MHz  
MCLK: 80MHz ;      HCLK: 80MHz  
PCLK: 50MHz ;      USB12M: 12MHz ;  
I2SMCLK: 12MHz ;    ADCCLK: 12MHz ;  
PWM COUNT\_CLK: 1MHz ;  
SDCICLK: 50MHz;      GPIO: 50MHz;  
CLK\_30\_60: 30MHz

## 3.7.2 Block diagram

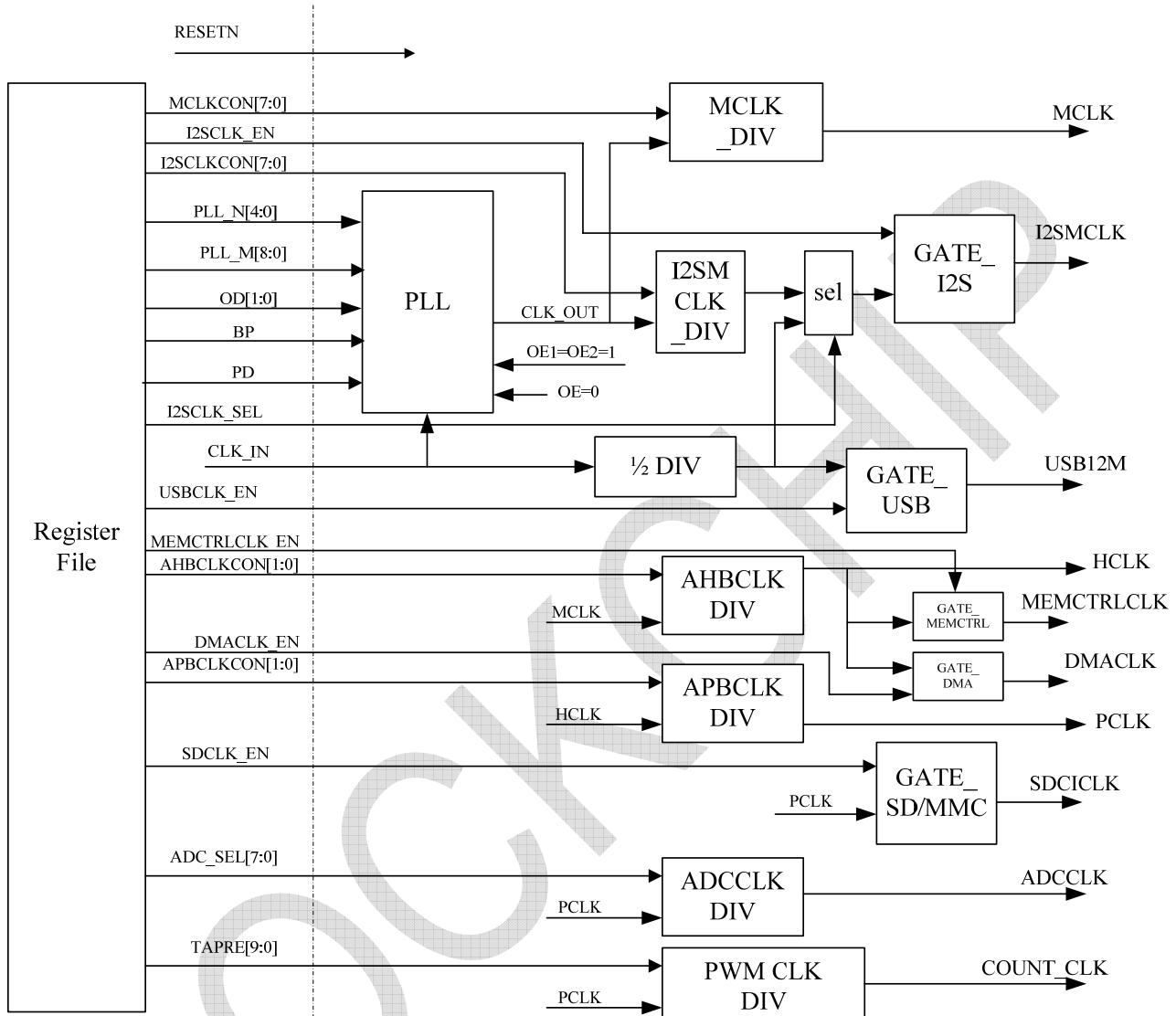


Figure 1. Block Diagram

Figure 1 shows a block diagram of the clock generator. An external crystal clock is connected to the oscillation amplifier, and the PLL (Phase-Locked-Loop) converts the low input frequency into a high-frequency clock CLK\_OUT required by Rock2. And then divide frequency CLK\_OUT to the needed clocks.

# Rock2 DATA SHEET V1.1

---

## 3.7.3 Block I/Os

### ADC CONTROLLER I/O DESCRIPTION

Register File interface			
N[5:0]	I	5-bit input divider control to PLL	
M[8:0]	I	9-bit feedback divider control to PLL	
OD[1:0]	I	Output divider control signal to PLL	
BP	I	Bypass PLL signal to PLL	
CLK_OUT	O	PLL adjustable Clock output	
PD	I	Disable power down mode	
MCLKCON[7:0]	I	8-bit divider control to MCLK	
I2SMCLKCON[7:0]	I	8-bit divider control to I2SMCLK	
I2SMCLK_EN	I	I2SMCLK Output Enable	
I2SMCLK_SEL	I	I2SMCLK output select	
AHCLKCON[1:0]	I	2-bit divider control to AHB	
APBCLKCON[1:0]	I	2-bit divider control to APB	
MEMCTRLCLK_EN	I	DMA clock Enable	
DMACLK_EN	I	DMA clock Enable	
SDCLK_EN	I	SDCICLK Output Enable	
USBCLK_EN	I	USB12M clock output enable	
ADC_SEL[2:0]	I	3-bit divider control to ADC clock	Reference to ADC Module
TAPRE[9:0]	I	10-bit divider control to PWM clock	Reference to PWM module
Register File interface I/Os total:			
Application interface			
MCLK	O	Main clock for DSP	
I2SMCLK	O	I2S Main clock	
USB12M	O	12MHz clock for USB	
HCLK	O	AHB clock	
PCLK	O	APB clock	
ADCCLK	O	ADC clock	
COUNT_CLK	O	PWM clock	
Application I/Os total: #			
Control Interface			
CLK_IN	I	External oscillator clock input	

---

## PLL I/O DESCRIPTION

PLL I/O interface			
OE1=1	I	Disable CLK1	
OE2=1	I	Disable CLK2	
OE=0	I	Enable CLK_OUT	
RESET	I	In normal mode, RESET=0, When in TEST mode 0, Set to External ~reset mode	
PLL_TST[1:0]	I	Test signal to PLL	

### 3.7.4 Divider Timing Characteristics

Figure 2. Divider Timing Characteristics

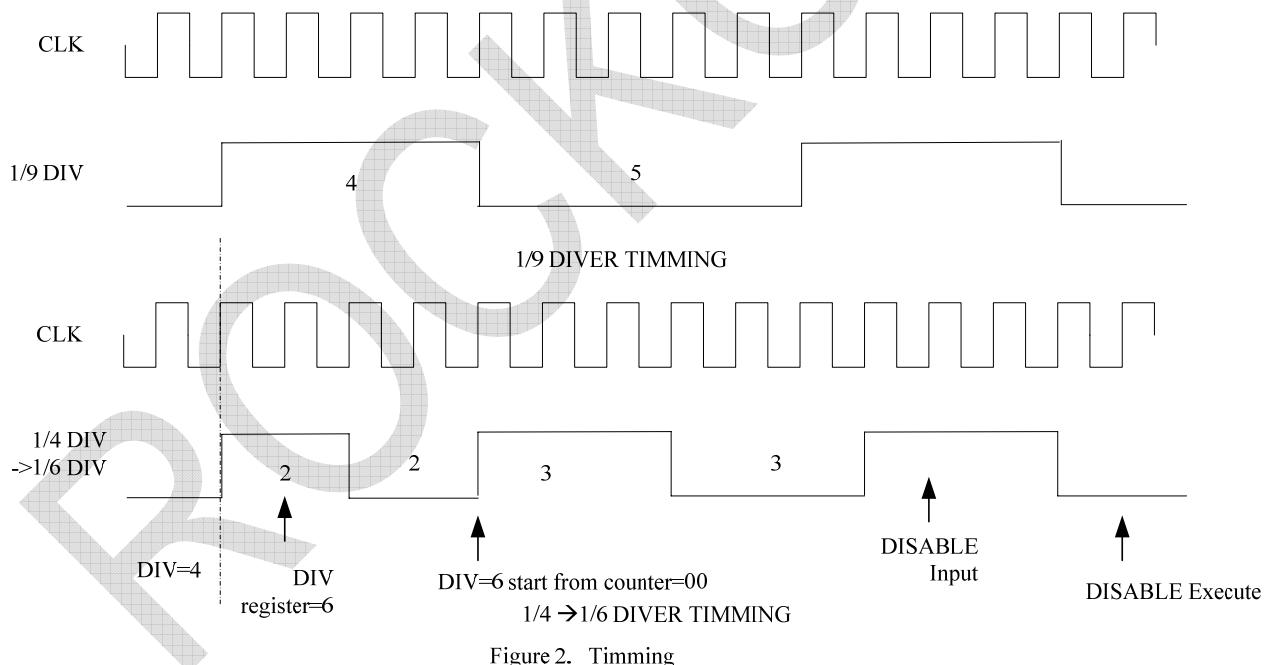


Figure 2. Timming

在改变分频器的分频系数后（即修改了寄存器的值），只有在上一次的波形完整后（即记数值为 0），才会用新的分频系数分频。

在停止分频器的分频输出控制后（即修改 ENABLE 的值为 0），只有在上一次的波形输出值为 0 后，才会停止分频输出。

## Rock2 DATA SHEET V1.1

---

### 3.7.5 Main Clock Generate :

CLK Generate: OSC (24MHz)

N=17, M=128: 24MHz/17x128=180.706MHz

N=25, M=128: 24MHz/25x128=122.88MHz

$F_{vco}/(P \cdot Q) = I2SMclk$ ,  $T=MCLK\_DIV+1$ ,  $P \cdot Q = I2SMCLK\_DIV+1$

Note: when use 256x32k,  $P \cdot Q = 15$

(此表中, N, M, P\*Q 为计算后的数值, 实际设置的数值为: N-2, M-2, P\*Q-1)

F <sub>vco</sub> [MHz]	T	MCLK(DSP) [MHz]	P	Q	I2SMclk [KHz]
N=17 M=256 361.412M	6	60.235	P=16 22.588M	2	256 x 44.118k
				4	256 x 22.059k
				8	256 x 11.028k
N=17 M=128 180.706M	4	45.176	P=8 22.588M	2	256 x 44.118k
		30.118		4	256 x 22.059k
		22.588		8	256 x 11.028k
		18.071			
		15.059			
		12.908			
N=25 M=256 245.76M	6	40.96	P=10 24.576M	1	256 x 96k
				2	256 x 48k, 384 x 32k
				3	256 x 32k
				4	256 x 24k, 384 x 16k
				6	256 x 16k
				12	256 x 8k
N=25 M=128 122.88M	2	61.44	P=5 24.576M	1	256 x 96k
		30.72		2	256 x 48k, 384 x 32k
		20.48		3	256 x 32k
		15.36		4	256 x 24k, 384 x 16k
		12.288		6	256 x 16k
				12	256 x 8k

CLK Generate: OSC (24MHz)

## Rock2 DATA SHEET V1.1

N=8, M=64: 24MHz/8x64=192MHz

$F_{VCO}/(P*Q) = I2SMclk$ ,  $T=MCLK\_DIV+1$ ,  $P*Q=I2SMCLK\_DIV+1$

(PLL 要求: 1MHz <= OSC/N <= 15MHz; 100MHz <= Fvco <= 500MHz)

此表中, N, M, T, P\*Q 为计算后的数值, 实际设置给寄存器的数值为:  $N\_DIV = N - 2$

$M\_DIV = M - 2$

$MCLK\_DIV = T - 1$

$I2SMCLK\_DIV = P*Q - 1$

Fvco	T	MCLK(DSP) [MHz]	P*Q	I2SMclk (Use PLL)	I2SMclk (Use OSC/2)
N=8 M=128 384MHz	4	96	32	12MHz	12MHz
	6	64			
	8	48			
	12	32			
	16	24			
	32	12			
N=8 M=64 192MHz	2	96	16	12MHz	12MHz
	4	48			
	6	32			
	8	24			
	12	16			
	16	12			
N=8 M=32 96MHz (Initial)	1	96	8	12MHz	12MHz
	2	48			
	4	24			
	6	16			
	8	12			
N=12 M=50 100MHz	1	100	8	-	12MHz (OSC)
	2	50			
	4	25			
	6	16.66			
	8	12.5			
N=12 M=100 200MHz	2	100	8	-	12MHz (OSC)
	4	50			
	6	33.33			
	8	25			
	10	20			

# Rock2 DATA SHEET V1.1

## 3.7.6 Register Descriptions

Register	Address	R/W	Description	Reset Value
MCLKCON	0x0001_EE00	R/W	Main Clock control Register	0x 0001
I2SMCLKCON	0x0001_EE04	R/W	I2S Clock control Register	0x 0107
AHBCLKCON	0x0001_EE08	R/W	AHB Clock control Register	0x 0000
APBCLKCON	0x0001_EE0C	R/W	APB Clock control Register	0x 0000
PLL_NDIV	0x0001_EE10	R/W	PLL 5-bit divider count register	0x 0006
PLL_MDIV	0x0001_EE14	R/W	PLL 9-bit Feedback divider count register	0x 001E
PLL_ODDIV	0x0001_EE18	R/W	PLL output divider count register	0x 0000
PLL_PDBP	0x0001_EE1C	R/W	PLL output control register	0x 0001
PWRCON	0x0001_EE38	R/W	Clock power control register	0x 001F

### MAIN CLOCK CONTROL REGISTER (MCLKCON)

MCLKCON	Bit	Description	Initial State
Reserved	[15:8]	Reserved	-
MCLK_DIV	[7:0]	8-bit prescaler value (The input clock is PLL CLK_OUT) MCLK = input clock / (MCLK_DIV + 1)	0000,0001b

### I2S MAIN CLOCK CONTROL REGISTER (I2SMCLKCON)

I2SMCLKCON	Bit	Description	Initial State
Reserved	[15:8]	Reserved	-
I2SMCLK_SEL	[8]	0: Select I2SMCLK_DIV clock 1: Select 12MHz	1
I2SMCLK_DIV	[7:0]	8-bit prescaler value (The input clock is PLL CLK_OUT) I2SMCLK = CLK_OUT / (I2SMCLK_DIV + 1)	0000,0111b

# Rock2 DATA SHEET V1.1

## AHB CLOCK CONTROL REGISTER (AHBCLKCON)

AHBCLKCON	Bit	Description	Initial State
Reserved	[15:2]	Reserved	-
AHBCLK_DIV	[1:0]	AHB clock output divider count register (The input clock is MCLK) 00: MCLK 01: MCLK/2 10: MCLK/4 11: MCLK/8	00b

## APB CLOCK CONTROL REGISTER (APBCLKCON)

APBCLKCON	Bit	Description	Initial State
Reserved	[15:2]	Reserved	-
APBCLK_DIV	[1:0]	APB clock output divider count register (The input clock is HCLK) 00: HCLK 01: HCLK /2 10: HCLK /4 11: HCLK /8	00b

## PLL INPUT DIVIDER REGISTER (PLL\_NDIV)

PLL_NDIV	Bit	Description	Initial state
Reserved	[15:5]	Reserved	-
PLL_NDIV	[4:0]	PLL 5-bit input divider count register, This signal must connect to PLL $N = \text{PLL\_NDIV} + 2$	0,0110b

## PLL INPUT DIVIDER REGISTER (PLL\_MDIV)

PLL_MDIV	Bit	Description	Initial state
Reserved	[15:9]	Reserved	-
PLL_MDIV	[8:0]	PLL 9-bit Feedback divider count register, This signal must connect to PLL $M = \text{PLL\_MDIV} + 2$	0,0001,1110b

## Rock2 DATA SHEET V1.1

---

### PLL FEEDBACK DIVIDER REGISTER (PLL\_ODDIV)

<b>PLL_ODDIV</b>	<b>Bit</b>	<b>Description</b>	<b>Initial state</b>
Reserved	[15:2]	Reserved	-
PLL_ODDIV	[1:0]	PLL output divider count register, This signal must connect to PLL 00: 1/1 01: 1/2 10: 1/2 11: 1/4	00b

### PLL OUTPUT ENABLE REGISTER (PLL\_PDBP)

<b>PLL_PDBP</b>	<b>Bit</b>	<b>Description</b>	<b>Initial State</b>
-	[15:2]	Reserved	-
PLL_PD	[1]	1= PLL Power down; 0= PLL is power on This signal must connect to PLL	0
PLL_BP	[0]	1= PLL Bypass; 0= Use PLL output This signal must connect to PLL	1

### CLOCK POWER CONTROL REGISTER (PWRCON)

<b>PWRCON</b>	<b>Bit</b>	<b>Description</b>		<b>Initial State</b>
-	[15:5]	Reserved		-
DMACLK_EN	[4]	0 = Disable	1 = Enable	1
MEMCTRLCLK_EN	[3]	0 = Disable	1 = Enable	1
SDCLK_EN	[2]	0 = Disable	1 = Enable	1
I2SCLK_EN	[1]	0 = Disable	1 = Enable	1
USBCLK_EN	[0]	0 = Disable	1 = Enable	1

### 3.7.7 Example Software

```

!-----PLL: 140MHZ---70MHz-----
mov r0, CLOCK_MCLKCON ! MCLK= CLKOUT/4
mov r2, 0x03
st r2, r0
call delay_20nop

mov r0, CLOCK_PLL_PDBP ! BP=1; CLKOUT=24MHz
mov r2, 0x01           ! MCLK=HCLK=PCLK=24/4=6MHz
st r2, r0
call delay_100nop

mov r0, CLOCK_AHCLKCON ! HCLK=MCLK
mov r2, 0x00
st r2, r0
call delay_20nop

mov r0, CLOCK_APBCLKCON ! PCLK=HCLK
mov r2, 0x00
st r2, r0
call delay_20nop

mov r0, CLOCK_PLL_NDIV ! PLL_NDIV=10+2=12
mov r2, 10
st r2, r0
call delay_20nop

mov r0, CLOCK_PLL_MDIV ! PLL_MDIV=68+2=70
mov r2, 68             ! PLLCLK=24MHz =70/12*24=140MHz
st r2, r0
call delay_500us

mov r0, CLOCK_PLL_PDBP ! BP=0, CLKOUT= PLLCLK=140MHz
mov r2, 0x00           ! MCLK=HCLK=PCLK=140/4=35MHz
st r2, r0
call delay_20nop

mov r0, CLOCK_APBCLKCON ! PCLK=HCLK/4, From 0→4→2
mov r2, 0x02
st r2, r0
call delay_20nop

mov r0, CLOCK_APBCLKCON ! PCLK=HCLK/2=17.5MHz
mov r2, 0x01
st r2, r0
call delay_20nop

mov r0, CLOCK_MCLKCON ! MCLK=140/2=70MHz, HCLK=70MHz, PCLK=HCLK/2=35MHz
mov r2, 0x01
st r2, r0
call delay_20nop

mov r0, CLOCK_AHCLKCON ! HCLK=MCLK/4
mov r2, 0x02

```

## Rock2 DATA SHEET V1.1

```
st    r2, r0

mov  r0, CLOCK_AHBCLKCON ! MCLK=70MHz, HCLK=MCLK/2=35MHz, PCLK=HCLK/2=17.5MHz
mov  r2, 0x01
st   r2, r0
call delay_20nop

mov  r0, CLOCK_APBCLKCON ! MCLK=70MHz, PCLK=HCLK=MCLK/2=35MHz
mov  r2, 0x00
st   r2, r0
call delay_20nop

.....
!-----PLL: 100MHZ---50MHZ-----
mov  r0, CLOCK_MCLKCON ! MCLK= CLKOUT/4
mov  r2, 0x03
st   r2, r0
call delay_20nop

mov  r0, CLOCK_PLL_PDBP ! BP=1; CLKOUT=24MHz
mov  r2, 0x01           ! MCLK=24/4=6MHz, PCLK=HCLK=MCLK/2=3MHz
st   r2, r0
call delay_20nop

mov  r0, CLOCK_PLL_MDIV ! PLL_MDIV=48+2=50
mov  r2, 48              ! PLLCLK =50/12*24=100MHz
st   r2, r0
call delay_500us

mov  r0, CLOCK_PLL_PDBP ! BP=0, CLKOUT= PLLCLK=100MHz
mov  r2, 0x00           ! MCLK= 100/4=25MHz, HCLK=PCLK= MCLK/2=12.5MHz
st   r2, r0
call delay_20nop

mov  r0, CLOCK_MCLKCON ! MCLK=100/2=50MHz, HCLK=PCLK= MCLK/2=25MHz
mov  r2, 0x01
st   r2, r0
call delay_20nop
```

注：1、在切换 MCLK, HCLK, PCLK 从无分频到有分频时，分频先设为 4，然后再切换为其它值。  
2、在改变 PLL 值时，要先设置到 BYPASS 态（BP=1），然后更改 PLL 值，延时 0.5 毫秒后，再切换到 PLL 态（BP=0）。

## 3.8 PWM

### 3.8.1 Overview

Rock2 has an internal timer A. It supported mode of PWM (Pulse Width Modulation) .

The timer has internally one pre-scale register (TA\_PRE), one counter register (TA\_CNT), and two data registers (TA\_DATA0, TA\_DATA1). The TA\_CNT is incremented by Count Clock that is pre-scaled by TA\_PRE value from Pclk. The role of TA\_DATA0 and TA\_DATA1 is different.

#### Function list:

- 16-bits timer
- support PWM mode
- 0 ~ 100 % duty ratio PWM signal generation

#### Clock Requirement:

Re-scaling the counting clock ( PCLK ) with the 10-bits pre-scale register (TA\_PRE)

$$\text{COUNT\_CLK} = \text{PCLK} / (\text{TA\_PRE} + 1)$$

### 3.8.2 Block diagram

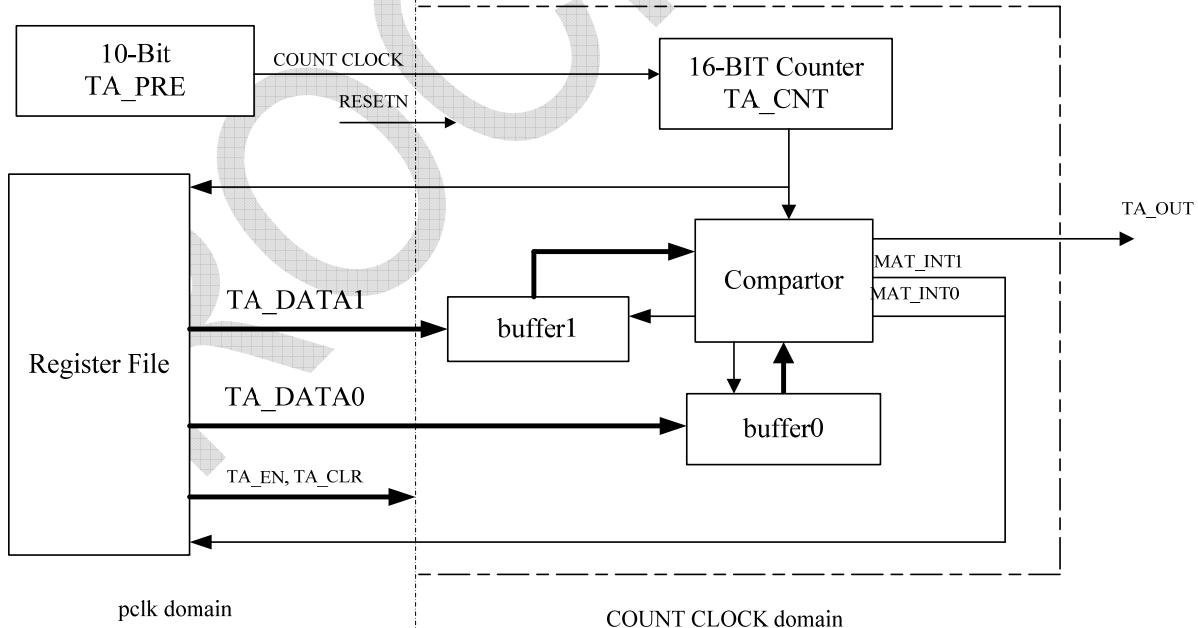


Figure 2. Block Diagram

## Rock2 DATA SHEET V1.1

---

### 3.8.3 Block I/Os

#### PWM I/O DESCRIPTION

Register Files interface		
TA_CLR	I	Clear operation
TA_EN	I	Timer enable
TA_DATA0(15:0)	I	Timer data0
TA_DATA1(15:0)	I	Timer data1
TACNT(15:0)	O	Timer counter
<b>APB Slave interface I/Os total:</b>		
Application interface		
COUNT_CLK	I	Counter clock
Resetn	I	An active-low, asynchronous reset.
MAT_INT0	O	Timer match DATA0 interrupt
MAT_INT1	O	Timer match DATA1 interrupt
TA_OUT	O	PWM output

## PWM MODE

The TA\_CNT value is compared to the two buffers that are updated with the values of TA\_DATA0 and TA\_DATA1 register. When (TA\_CNT) is equal to the buffer of TA\_DATA0, TA\_MAT\_INT0 interrupt occurs and the timer continues the counting operation without clearing the counter. When (TA\_CNT) is equal to the buffer of TA\_DATA1, the timer generates TA\_MAT\_INT1 interrupt, clears the TA\_CNT register, and updates the internal buffers with the values of TA\_DATA0 register and TA\_DATA1 register. For each interrupt, the TA\_OUT is toggled. As the values of TA\_DATA0 and TA\_DATA1 register are updated after the TA\_MAT\_INT1 interrupt, the new values in TA\_DATA0 and TA\_DATA1 registers have an effect after TA\_MAT\_INT1 occurs.

This mode is used to generate a configurable PWM signal. The clock period of PWM signal can be set in TA\_DATA1 register and the duty ratio can be set in TA\_DATA0 register. The operation of this mode is described in the Figure 5.

The pulse is low/(low+high)==(TA\_DATA0+1)/(TA\_DATA1+1)

Ex: TA\_DATA0:TA\_DATA1=5:5=1 → when 5 ADC\_CLK, TA\_OUT will always high;  
 TA\_DATA0:TA\_DATA1=5:3>1 → TA\_OUT will always low;  
 TA\_DATA0:TA\_DATA1=5:7=1 → TA\_OUT 6 clocks low, 2 clocks high;

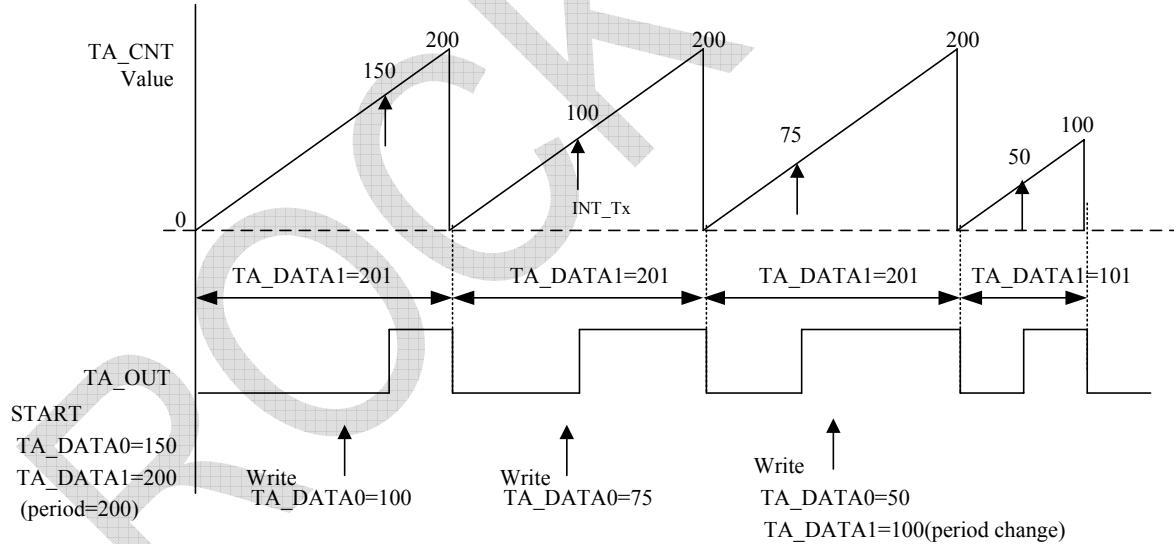
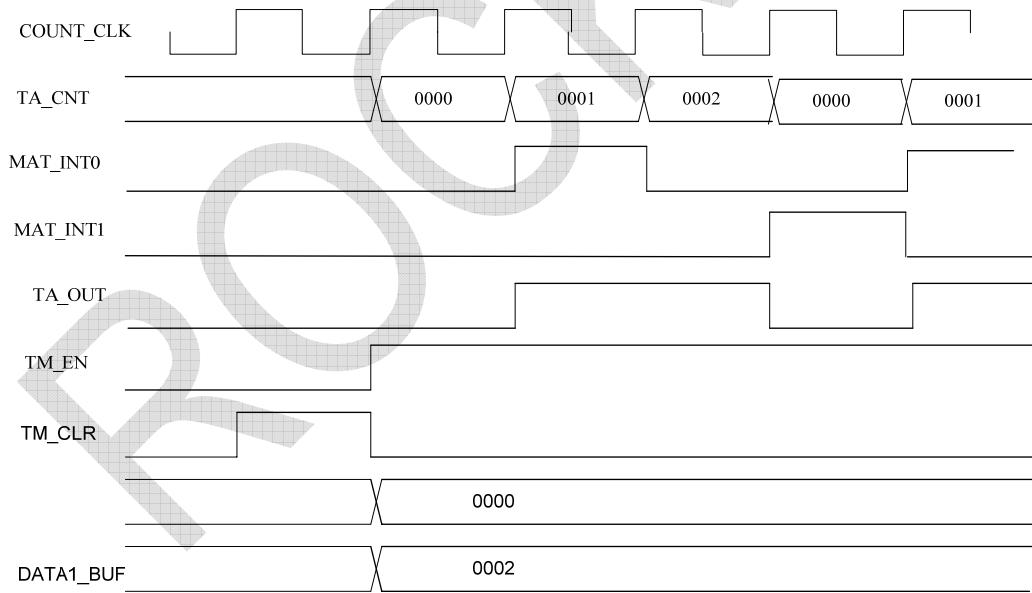
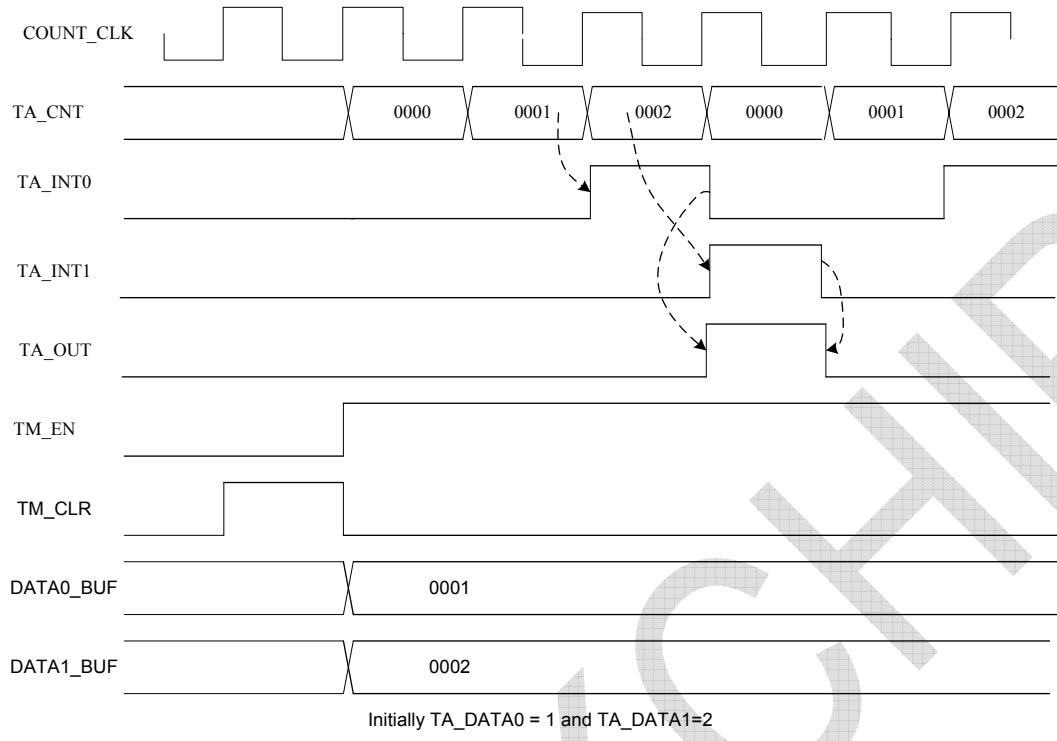


Figure 5. PWM MODE OPERATION

Depending on the values of the TA\_DATA0 and TA\_DATA1, the shapes of the PWM signals are different. The detailed waveforms in PWM mode are shown in Figure 6, Figure 7, Figure 8, and Figure 9.

# Rock2 DATA SHEET V1.1

---



## Rock2 DATA SHEET V1.1

---

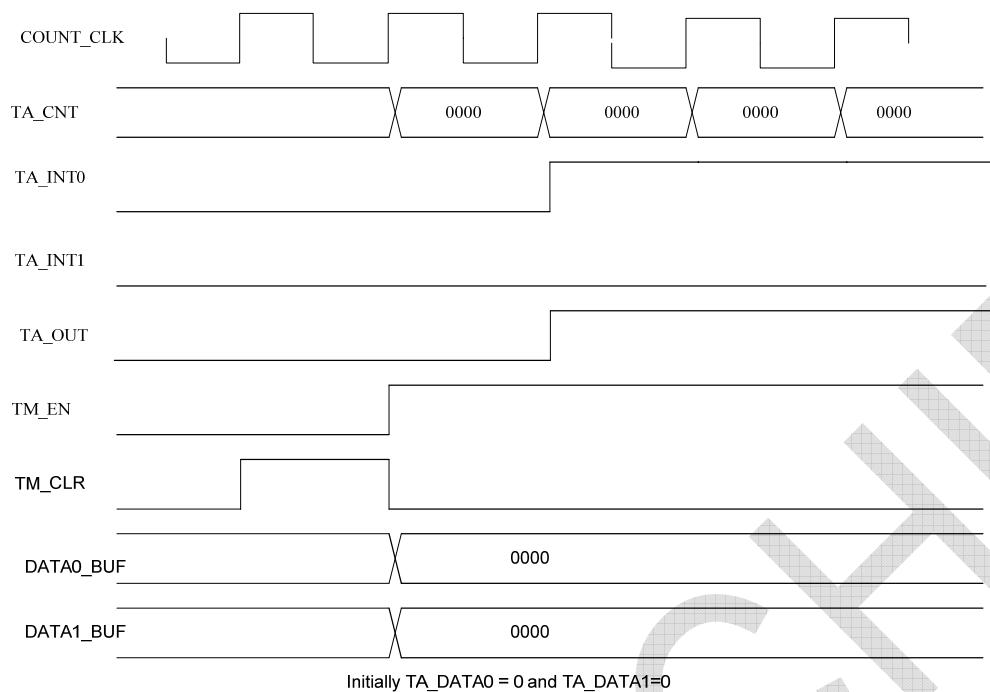


Figure 8. PWM MODE When TA\_DATA0 = 0 and TA\_DATA1=0

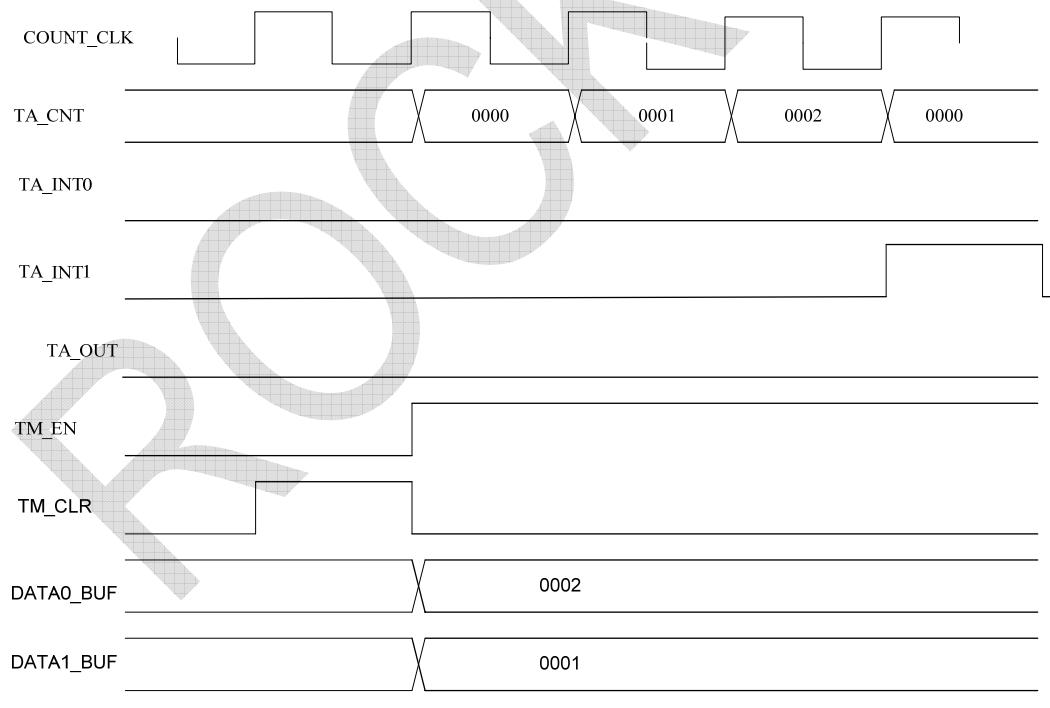


Figure 9. PWM MODE When TA\_DATA0 = 2 and TA\_DATA1=1

# Rock2 DATA SHEET V1.1

---

## 3.8.4 Register Descriptions

Name	Width	Address(Virtual)	R/W	Description	Reset
<b>Timer Registers</b>					
TACON	16	0x0001_EC80	R/W	Control Register	0x0000
TACMD	16	0x0001_EC84	R/W	Command Register	0x0000
TADATA0	16	0x0001_EC88	R/W	Data0 Register	0x0000
TADATA1	16	0x0001_EC8C	R/W	Data1 Register	0x0000
TAPRE	16	0x0001_EC90	R/W	Prescale Register	0x0000
TACNT	16	0x0001_EC94	R	Counter Register	0x0000

### TACON

Bits	Name	Type	Description
31-3			Reserved
2	TA_OUT	R	The PWM output
1	TA_INT1	R	Match interrupt 1 status. This field is updated when a match interrupt 1 occurs.
0	TA_INT0	R	Match interrupt 0 status. This field is updated when a match interrupt 0 occurs.

### TACMD

Bits	Name	Type	Description
31-2			Reserved
1	TA_CLR	W	Clear operation. This field is always zero when read. 0 = Nothing occurs 1 = Initialize the timer. - Clear the counter register. - TA_DATA0 and TA_DATA1 are updated to the internal buffers
0	TA_EN	R/W	Timer enable command 0 = Disable the timer 1 = Enable the timer

## Rock2 DATA SHEET V1.1

---

### TADATA0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA_DATA0															

Bits	Name	Type	Description
15:0	TA_DATA0	R/W	The target counting value is stored in this field. When the counter value equals to this register, a MAT_INT0 interrupt is generated. This field is updated to the internal data buffer 0 when MAT_INT0 occurs or when clear operation is executed.

### TADATA1

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA_DATA1															

Bits	Name	Type	Description
15:0	TA_DATA1	R/W	The target counting value is stored in this field. When the counter value equals to this register, a MAT_INT1 interrupt is generated. This field is updated to the internal data buffer 1 when MAT_INT1 occurs or when clear operation is executed.

### TAPRE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA_PRE															

Bits	Name	Type	Description
9:0	TA_PRE	R/W	Pre-scale value <i>(It is include in the CLOCK module)</i> COUNT_CLK = PCLK / (TA_PRE+1)

### TACNT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TA_CNT															

Bits	Name	Type	Description
15:0	TA_CNT	R	Counter register

## 3.8.5 Example Software

```
mov r0, PWM_TACMD
mov r2, 0x00          !disable
st r2, r0

mov r0, PWM_TAPRE
mov r2, 32
st r2, r0

mov r0, PWM_TADATA0  !2/3 int: 33*300=9900clks
mov r2, 199
st r2, r0
mov r0, PWM_TADATA1
mov r2, 299
st r2, r0
call delay_100nop

mov r0, PWM_TACMD
mov r2, 0x03          !enable and clr
st r2, r0

bits %imask, 4        !enable pwm interrupt

mov r0, GPIO_PCON2B   !PWM out to p2.13,14,15
mov r2, 0xfd55
mov r3, 0x0
stdu r2, r0, 2

.....
mov r0, GPIO_PCON2B   !PWM out to p2.14,15
mov r2, 0xf555
mov r3, 0x0
stdu r2, r0, 2

.....
mov r0, GPIO_PCON2B   !PWM out to p2.15
mov r2, 0xd555
mov r3, 0x0
stdu r2, r0, 2
```

## 3.9 10bit ADC

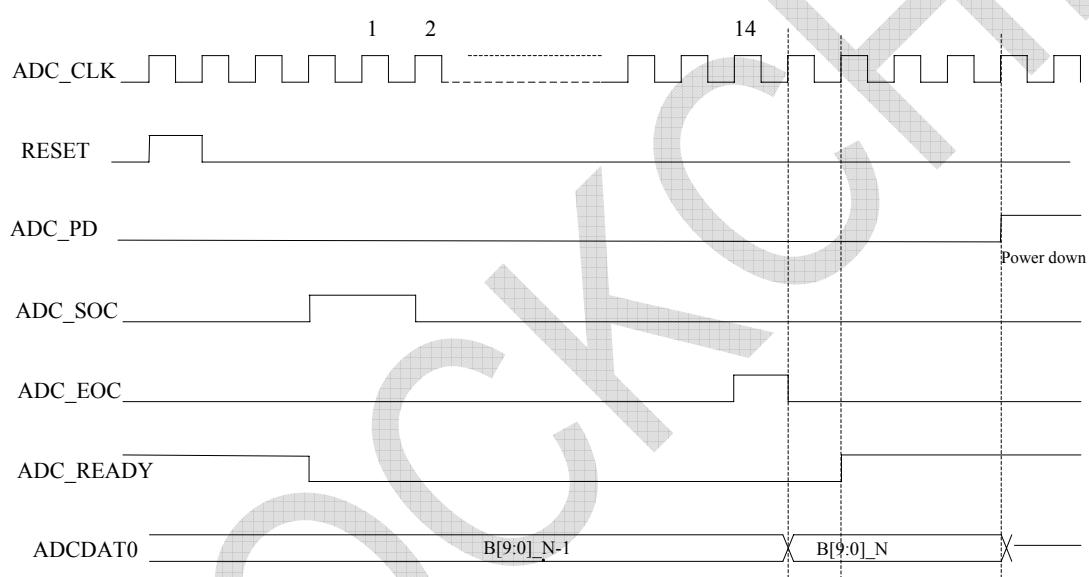
### 3.9.1 Overview

The 10-bit A/D converter (ADC) module is used to convert the analog signal into the digital signal.

ADC is generate by PCLK by setting the **ADCFRE** register. The adc\_clk can be prescale to be pclk/1, pclk/2, pclk/3....pclk/128.

### 3.9.2 Interface

Figure 3. ADC Controller Timing Characteristics



### 3.9.3 Block I/Os

#### ADC CONTROLLER I/O DESCRIPTION

Register file interface		
PD	I	A/D converter power down mode
SOC	I	A/D converter start
SEL(2:0)	I	A/D converter input select
EOC	I	End of conversion
B(9:0)	I	A/D converted data
ADC_READY	O	ADC Ready (high), when low, ADC is busy
ADC_RSTN	I	Reset signal, Low active

# Rock2 DATA SHEET V1.1

Register file interface I/Os total:		
Application interface		
AIN[3:0]	I	4 channel analog input
REXT100K	I	External 100K reference resistor
VREF	I	External reference voltage
ADC_CLK	I	A/D converter clock
ADC_RST	O	Reset signal, RESET=1, the chip is reset
PCLK	I	APB clock

## 3.9.4 Register Descriptions

### ADC CONTROLLER INTERNAL REGISTERS

Register	Address	R/W	Description	Reset Value
ADCCON	0x0001_ED00	R/W	ADC control register	0x0004
ADCDAT0	0x0001_ED04	R	ADC conversion data	-
ADCFRE	0x0001_ED08	R/W	ADC Frequency control register	0x0000
ADCRDY	0x0001_ED0C	R	ADC Ready signal register	0x0000

**ADC Control (ADCCON) Register**

ADCCON	Bit	R/W	Description	Initial State
	[15:5]		NC	
SEL_MUX	[4:2]	R/W	Analog input channel select. 000 = AIN0 001 = AIN1 010 = AIN2 011 = AIN3 100 = AIN4 [VBG]	000b
ADC_PD	[1]	R/W	Power down mode select. 0 = Normal operation mode 1 = Power down mode	0
ADC_SOC	[0]	R/W	A/D conversion starts by setting this bit. 0 = No operation 1 = A/D conversion starts and this bit is cleared after 2 ADC_CLK after the start-up.	0

## Rock2 DATA SHEET V1.1

---

### ADC Conversion Data (ADCDAT0) Register

ADCDAT0	Bit	R/W	Description	Initial State
B[9:0]	[9:0]	R	Normal ADC Conversion data value. (0 ~ 3FF)	-

### ADC Frequency Pre-scale Control (ADCFRE) Register

ADCFRE	Bit	R/W	Description	Initial State
	[15:8]	-	NC	
ADC_SEL	[7:0]	R/W	ADC Frequency scale as: 000 = pclk/1 ADC_CLK==pclk/(ADC_SEL+1)	00

### ADC Ready Control (ADCRDY) Register

ADCRDY	Bit	R/W	Description	Initial State
	[15:1]	-	NC	
ADC_READY	[0]	R/W	ADC is in Ready(read only) 0 = not ready 1 = ready, the data output is in ADCCDAT0	1

#### 3.9.5 Example Software

```

mov r0, ADC_ADCFRE ! set ADC_CLK to ~1MHz, PCLK/10
mov r2, 0x9
st r2, r0
mov r0, ADC_ADCCON ! [4:2]: sel ch, [1]: pd, [0]: soc
mov r2,0b00001 ! sel ADC0
st r2, r0
.....delay some clks....
mov r0, ADC_ADCRDY
ld r2, r0 ! ==1?
! if ==1, then
mov r0, ADC_ADCCDAT0 ! read ADC0 data to r2
ld r2, r0

```

## 3.10 I2DSP Master/Slave

### 3.10.1 Overview

The Audio Interface adopt I2DSP mode to transmits PCM audio data to external DAC and receives PCM audio data from external ADC. To minimize the number of pins required and to keep wiring simple, a 3-line serial bus which consists of a data line for time-multiplexed two-channel data(left/right), a word select line and a clock line is used. In Rock2, The I2DSP mode Audio Interface bus has 2 data lines(one for reception and 1 for transmission), 2 word select lines (LRCK) , and 1 Mclk line (I2DSP\_MCLK).

The I2DSP mode Audio Interface bus has two modes for data transfer. In transmission mode, I2DSP module makes a request for PCM audio data to DMA module and DMA module brings audio data from SRAM. In reception mode, I2DSP module gets audio data from external source and stores them into SRAM by requesting to DMA. 1 data lines are synchronized with 1 word select line and 1 clock line for transmission mode and 1 data line is synchronized with 1 word select line and 1 clock line for reception mode.

In I2DSP mode, the chip which generates the bit clock is called master. Either transmitter or receiver of audio data has to generate the bit clock and word select clock as a master since they use the same clock signal for data transfer. When Rock2 acts as a slave in I2DSP mode, it receives two clock signals (bit clock and word select clock) from a master even when it transfers audio data.

When Rock2 acts as a master in I2DSP mode, it generate two clock signals (bit clock and word select clock) to the codec.

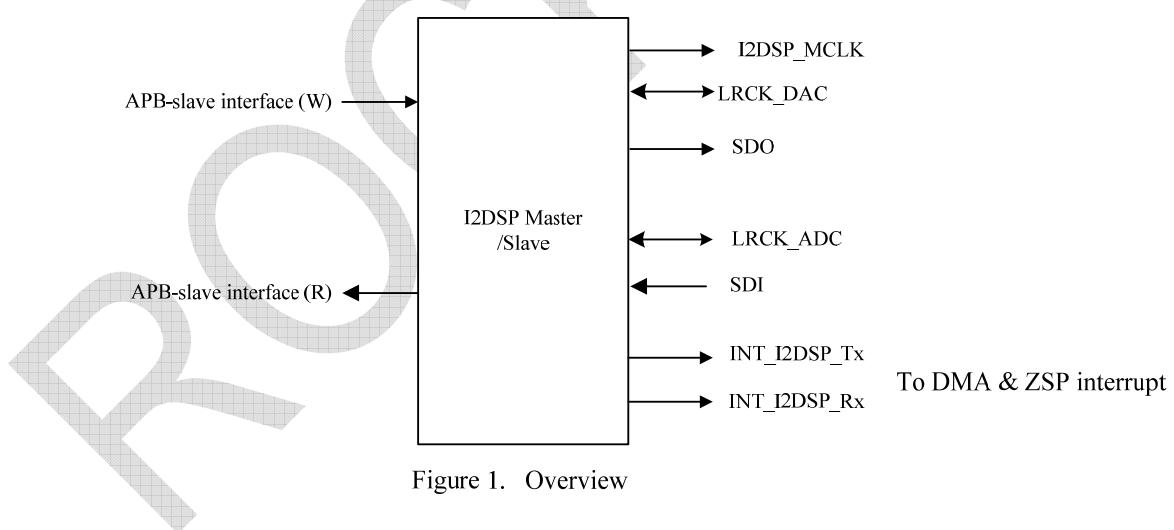


Figure 1. Overview

### Function list:

- 2 data transfer modes – transmission, reception
- DMA mode transfer, DSP interrupt/inquiry mode.
- 8 x 24-bit buffer for Transmission and 8 x 24-bit buffer for reception

## Rock2 DATA SHEET V1.1

- 16/18/20/24 bit data per channel
- 1 channel for transmission and 1 channel for reception
- MSB-first transfer mode only
- I2DSP Master and Slave mode
- 12M master clock output only
- 250,272/273fs(sampling frequency) serial bit clock per frame ( left channel + right channel )
- 250,272/273fs master clock(DAC clock)

### Clock Requirement:

Audio main clock (I2DSPMCLK) is 12M only. The sampling frequency of audio data come from external Codec. The relationship between sampling frequency (fs) and audio main clock is shown in Table 1. Serial bit clock (SCK) is equal to MCLK, data bit per channel(Table 2) is set by the value of configuration register(I2DSP\_TXCONF, I2DSP\_RXCONF). Word select signal (LRCK) has the same frequency as sampling frequency(fs).

Table 1. Sample rate look-up table for Master mode

**Table 1. The Frequency of Sample rate of master mode**

Mclk is 12M only

LRCK (fs)	8 KHz	11.025 KHz	16 KHz	22.05 KHz	32 KHz	44.10 KHz	48 KHz
fs	250*6	272*4	250*3	272*2	250*3/2	272	250

Table 2. Sample rate look-up table for slave mode

**Table 2. The Frequency of Sample rate of slave mode**

Mclk is 12M only, and LRCK is generate from CODEC

LRCK (fs)	8 KHz	11.025 KHz	16 KHz	22.05 KHz	32 KHz	44.10 KHz	48KHz
fs	250*6	272/273*4	250*3	272/273*2	250*3/2	272/273	250

## 3.10.2 Block diagram

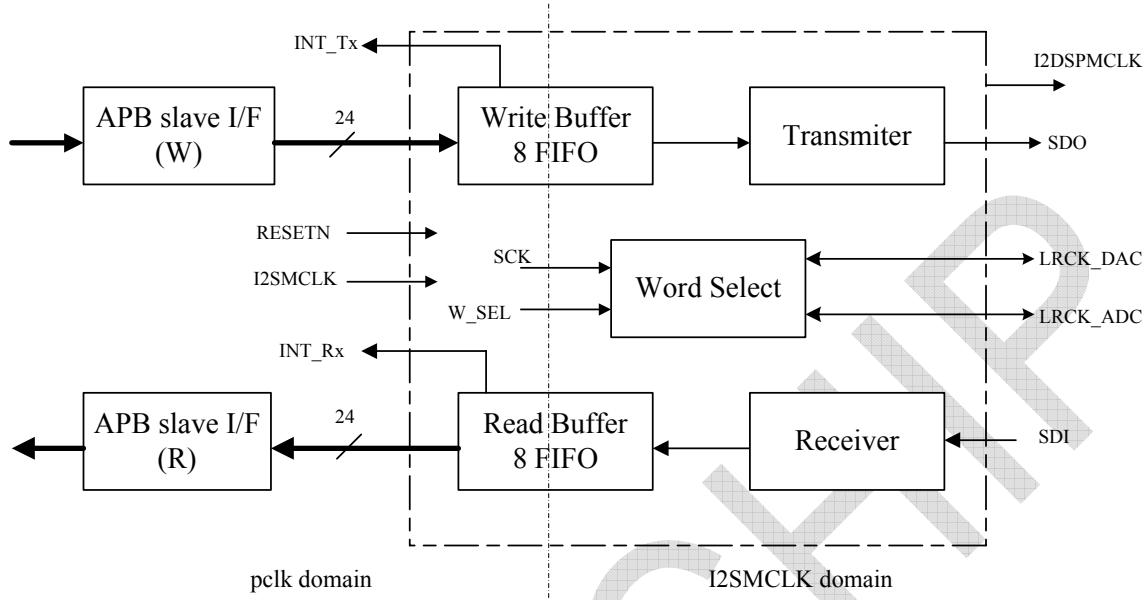


Figure 2. Block Diagram

## 3.10.3 Block I/Os

APB slave interface		
pclk	I	APB clock.
presetn	I	An active-low, asynchronous APB interface domain reset.
psel	I	APB peripheral select.
Paddr(9:0)	I	APB address bus.
Pwdata(31:0)	I	APB write data bus.
pwrite	I	APB write control.
penable	I	APB enable control that indicates the second cycle of the APB frame.
Prdata(31:0)	O	APB readback data.
APB Slave interface I/Os total:		
Application interface		
SCK	I	I2DSP mode serial data clock
LRCK	O	I2DSP mode word select.
SDO	O	I2DSP mode serial data output
I2SMCLK	I	I2DSP module main clock input
SDI	I	I2DSP mode serial data input
I2DSPMCLK	O	I2DSP mode main clock output to external codec
I2C_EXT	O	Select External I2DSP_EXT,I2C_EXT interface
RESETN	I	I2DSP Slave asynchronous reset. Active low
SW_CODEC_RSTN	O	CODEC asynchronous reset. Active low

### 3.10.4 Interface

Figure 3. I2DSP mode Transmit Interface

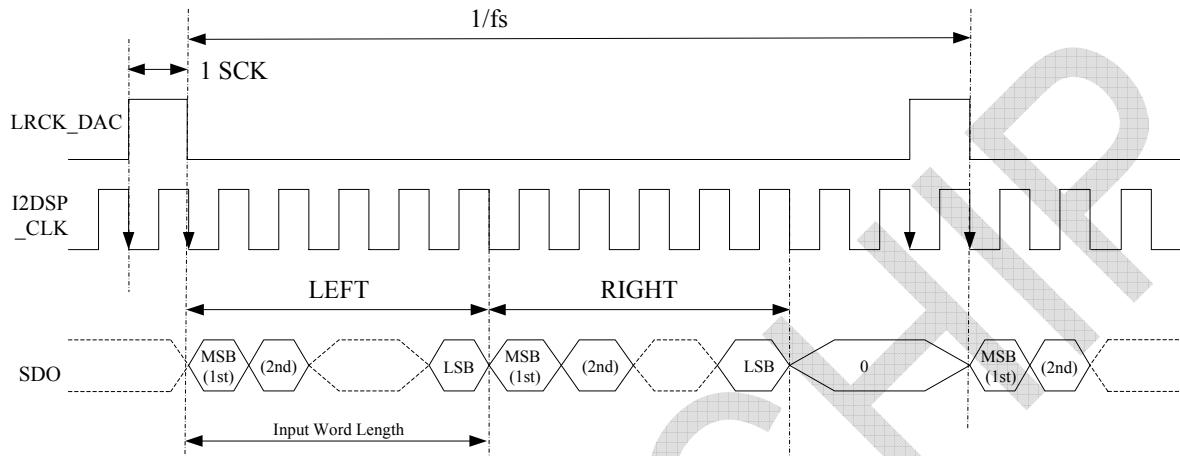


Figure 3 . DSP mode Write Timing

Figure 4. I2DSP mode Receive Interface

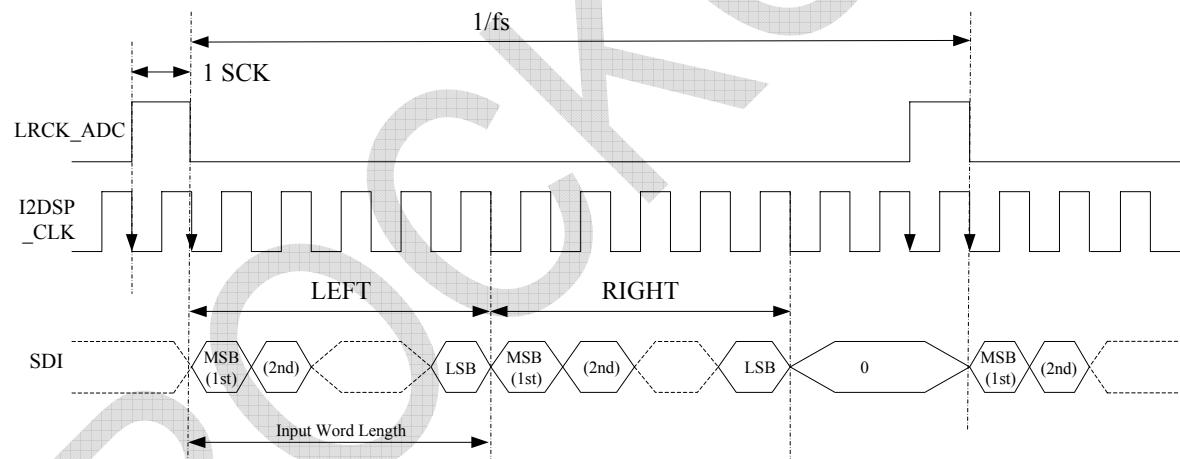


Figure 4 . DSP mode Read Timing

In Rock2, The I2DSP mode Audio Interface module is applicable to DSP interface format as shown above. I2DSP-Bus format starts data transfer at the next SCK clock after word select signal (LRCK) rising edge. I2DSP-Bus format transfer the Left channel and Right channel data continuously, and MSB of audio data are transferred first. If there exists more serial clock bits in one frame than serial data bits, the remaining clock bits are stuffed with zero's in each case.

## Start and Stop Condition

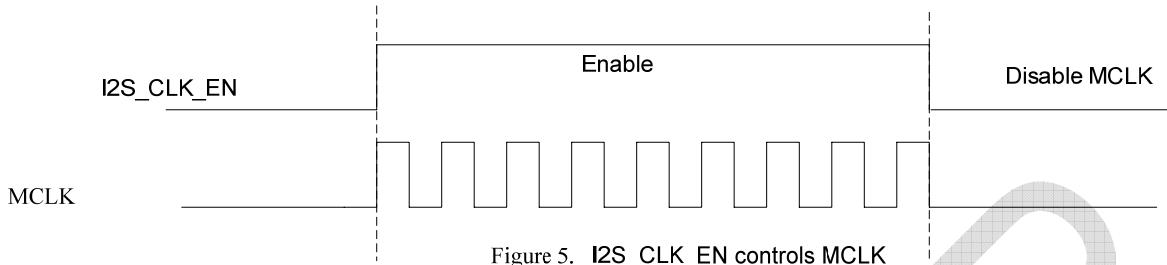


Figure 5. I2S\_CLK\_EN controls MCLK

To make I2DSP module active, I2S\_CLK\_EN bit in I2SCLKCON must be set to '1'. After I2DSP module becomes active, Setting command register (I2DSPTXCOM or I2DSPRXCOM) to '0x0000 000E' drives I2DSP to its function mode. I2S\_CLK\_EN bit in command register decides the generation of main clock(MCLK) and makes I2DSP bus stop immediately after it is set to '0'.

To make I2DSP inactive, the procedure is as follows. Set I2S\_CLK\_EN to '0'.

## FIFO model: (Using DMA)

The transmit and receive block all have 8Words (24bytes) FIFO.

When transmit, DMA first check signal I2DSP\_TX\_VILID, which is a request signal, if it is high, then transmit data to TxFIFO0-7. If the TxFIFOs are not full, I2DSP\_TX\_NFULL is high, it transmit I2DSP\_TX\_VILID to DMA. If the TxFIFOs are full, I2DSP\_TX\_NFULL is low, it resets I2DSP\_TX\_VILID to low, DMA is waiting a high of I2DSP\_TX\_VILID for next transmission. As the shift register reads a data from TxFIFOs, I2DSP\_TX\_NFULL comes high. And then the I2DSP\_TX\_VILID comes high as a request to DMA.

When receive, the Shift Register puts data to the RxFIFOs, I2DSP\_RX\_NEMPTY will be set to high, then transmit a I2DSP\_RX\_VILID to DMA, If DMA reads all the data, I2DSP will reset I2DSP\_RX\_NEMPTY to low, and I2DSP\_RX\_VILID will be low. When the RxFIFOs are full, the I2DSP\_RX\_FULL will be high, waiting for the DMA to read the data.

Note: I2DSP\_TX\_VILID is same as I2DSP\_TX\_NFULL, but when DMA\_ACK is from low to high, I2DSP\_TX\_VILID will be low first, if I2DSP\_TX\_NFULL is still high level, I2DSP\_TX\_VILID will be high at next PCLK.

I2DSP\_RX\_VILID is same as I2DSP\_RX\_NEMPTY, but when DMA\_ACK is from low to high, I2DSP\_RX\_VILID will be low first, if I2DSP\_RX\_NEMPTY is still high level, I2DSP\_RX\_VILID will be high at next PCLK.

## FIFO model: (Using DSP)

When transmit, DSP first check signal I2DSP\_TX\_NFULL, if it is High, then transmit data to TxFIFO0-7. If the TxFIFOs are full, it pulls I2DSP\_TX\_NFULL to Low, DSP waiting a High of I2DSP\_TX\_NFULL for next transmission. As the shift register reads a data from TxFIFOs, I2DSP\_TX\_NFULL comes High.

When receive, the Shift Register puts data to the RxFIFOs, DSP first check signal I2DSP\_RX\_NEMPTY, if it is High, then reads data from RxFIFOs. If it reads all the data, I2DSP will reset I2DSP\_RX\_NEMPTY to Low. When the RxFIFOs are full, the I2DSP\_RX\_FULL will be high

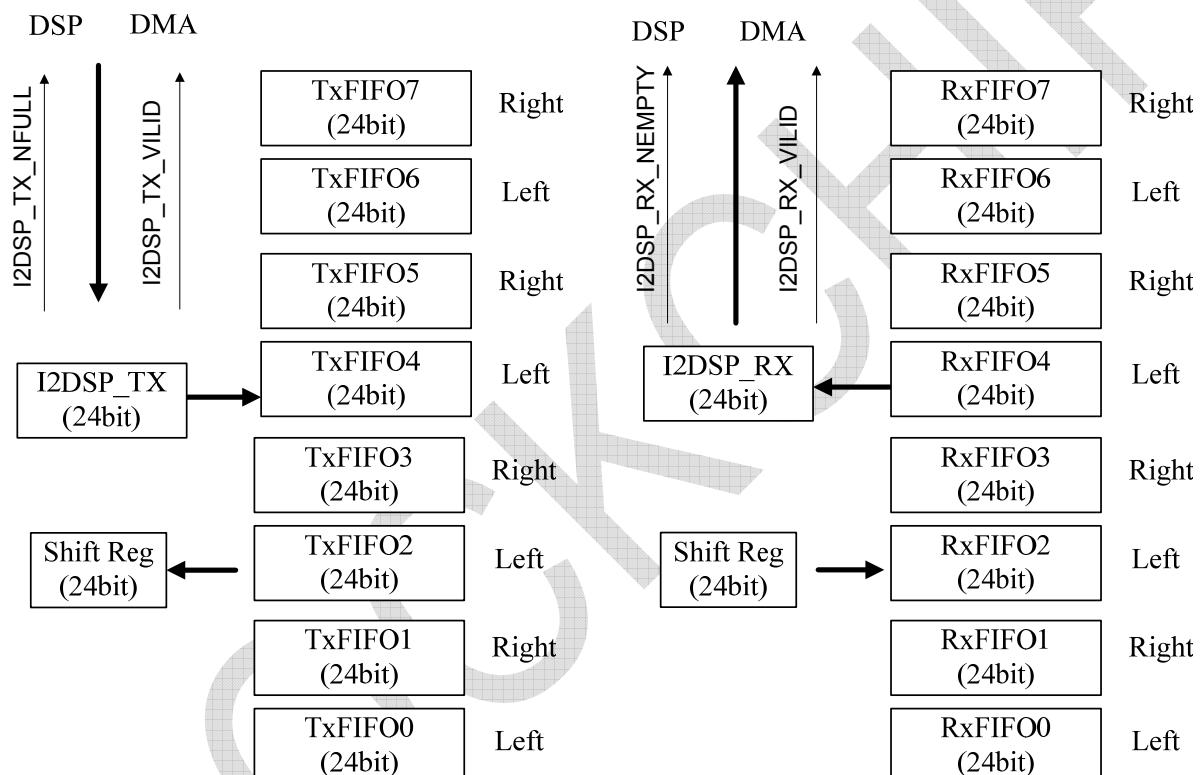


Figure 10. I2DSP FIFO Block

# Rock2 DATA SHEET V1.1

## 3.10.5 Register Descriptions

Name	Width	Address	R/W	Description	Reset value
I2DSP_TXCONF	16	0x0001_EC00	R/W	Tx configuration Register	0x0000
I2DSP_TXCOM	16	0x0001_EC08	R/W	Tx command Register	0x0000
I2DSP_RXDB	32	0x0001_EC10	W	Tx data buffer	0x0000 0000
I2DSP_DPCTRL	16	0x0001_EC20	R/W	Data pointer control	0x0000
I2DSP_RXCONF	16	0x0001_EC30	R/W	Rx configuration Register	0x0000
I2DSP_RXCOM	16	0x0001_EC34	R/W	Rx command Register	0x0000
I2DSP_STATUS	16	0x0001_EC38	R	status register	0x0000
I2DSP_RXDB	32	0x0001_EC40	R	Rx data buffer	0x0000 0000
I2C_EXT	16	0x0001_EC60	R/W	I2C External	0x0000
SW_CODEC_RSTN	16	0x0001_EC64	R/W	CODEC RSTN	0x0000

### I2DSP Tx Configuration Register (I2DSP\_TXCONF)

I2DSP_TXCONF	Bit	Description		Initial State
	[15:6]	Reserved		0
Transmit DAC Fs control	[5:3]	0 : 48K 1 : 44.1K 2 : 32K 3 : 22.05K 4 : 16K 5 : 11.025K 6 : 8K 7 : 8K		000b
Master/Slave mode select	[2]	0: Slave mode 1: Master mode		0
Serial Data Bit per Channel	[1:0]	00 = 16 bit 10 = 20 bit	01 = 18 bit 11 = 24 bit	00b

## Rock2 DATA SHEET V1.1

### I2DSP TX Command Register (I2DSP\_TXCOM)

I2DSP_TXCOM	Bit	Description	Initial State
	[15:3]	Reserved	0
I2DSP interface enable	[2]	0 = I2DSP interface disable (stop) 1 = I2DSP interface enable (start)	0
Tx enable select	[1]	0 = No transfer 1 = Transmit Mode On	0
DMA service request enable	[0]	0 = DMA request disable 1 = DMA request enable	0

### I2DSP Data Buffer Register (I2DSP\_TXDB)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSB(16/20/18/24)		DATA													
LSB18		X													
LSB18		LSB20		LSB24		X									

I2DSP_TXDB	Bit	Description	Initial State
I2DSP Transmit Data	[31:8]	24 bits Transmit Data to DAC	0
	[7:0]	Reserved	0

### I2DSP RX Configuration Register (I2DSP\_RXCONF)

I2DSP_RXCONF	Bit	Description	Initial State
	[15:6]	Reserved	0
Receive ADC Fs control	[5:3]	0 : 48K 1 : 44.1K 2 : 32K 3 : 22.05K 4 : 16K 5 : 11.025K 6 : 8K 7 : 8K	000b
Master/Slave mode select	[2]	0: Slave mode 1: Master mode	0
Serial Data Bit per Channel	[1:0]	00 = 16 bit 10 = 20 bit	00b
		01 = 18 bit 11 = 24 bit	

## Rock2 DATA SHEET V1.1

---

### I2DSP Rx Command Register (I2DSP\_RXCOM)

I2DSP_RXCOM	Bit	Description	Initial State
	[15:3]	Reserved	0
Rx enable select	[1]	0 = No Receive 1 = Receive Mode On	0
DMA service request enable	[0]	0 = DMA Request disable 1 = DMA Request enable	0

### I2DSP Rx Data Buffer Register (I2DSP\_RXDB)

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MSB(16/20/18/24)	DATA														LSB16
15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LSB18	LSB20	LSB24 X													

I2DSP_RXDB	Bit	Description	Initial State
I2DSP Receive Data	[31:8]	24 bits Receive Data from ADC	0
	[7:0]	Reserved	0

### I2DSP Status Register (I2DSP\_STATUS)

I2DSP_STATUS	Bit	Description	Initial State
	[15:2]	Reserved	0
RX_FIFO_NEMP	[1]	Rx data buffer state flag 1 = not empty 0 = empty	0
TX_FIFO_NFUL	[0]	Tx data buffer state flag 1 = not full 0 = full	1

## Rock2 DATA SHEET V1.1

---

### I2DSP Data Pointer Control (I2DSP\_DPCTRL) Register

Name	Bit	Description	Initial State
Reserved	[15:2]		0
RX_FIFORST	[1]	Reset the receive FIFO. 0 = No effect 1 = Reset the Receive FIFO	0
TX_FIFORST	[0]	Reset the transmit FIFO. 0 = No effect 1 = Reset the Transmit FIFO	0

### I2C External Control (I2C\_EXT) Register

Name	Bit	Description	Initial State
Reserved	[15:2]		0
I2DSP_EXT	[1]	Set I2DSP to external . 0 = To Internal Codec 1 = To external I2DSP pad.	0
I2C_EXT	[0]	Set I2C to external . 0 = To Internal Codec 1 = To external I2C pad.	0

### CODEC Reset (SW\_CODEC\_RSTN) Register

Name	Bit	Description	Initial State
Reserved	[15:1]		0
SW_CODEC_RSTN	[0]	Set Codec to reset, low active. 0 = To Internal Codec Reset 1 = Codec reset deactivate	0

Note: When the external resetn is low, the **SW\_CODEC\_RSTN** will be reset to 0, until the external resetn goes high, DSP will run the software and then set **SW\_CODEC\_RSTN** to high to enable CODEC.

When using CODEC, you can set **SW\_CODEC\_RSTN** to low and then high to reset the CODEC.

## Rock2 DATA SHEET V1.1

### 3.10.6 Example Software

```
----- transmit dac & adc -----
! txconf = 32'b00_0011; // [5:3]: fs, [2]:master, [1:0]:24-16bits
! txcom = 32'b110;      // [2]: i2dsp_en, [1]: tx_en, [0]: nc
! dpctrl = 32'b11;       // [1]: rxfiforst, [0]: txfiforst
! rxconf = 32'b00_0011; // [5:3]: fs, [2]: master, [1:0]: 24-16bits //slave
! rxcom = 32'b10;        // [1]: rx_en, [0]: nc
!

mov r0, I2DSP_TXCONF    ! TX slave; [5:3]: fs=48KHz, [2]:slave mode, [1:0]: 20bits
mov r2, 0b000010
st r2,r0

mov r0,I2DSP_TXCOM     ! i2dsp_en, TX enable
mov r2,0b110
st r2,r0

mov r0, I2DSP_DPCTRL   ! rxfiforst, txfiforst
mov r2, 0b11
st r2, r0

mov r0, I2DSP_RXCONF   ! RX ; [5:3]: fs=48KHz, [2]: master,[1:0]: 24bits
mov r2, 0b000011
st r2, r0

mov r0, I2DSP_RXCOM    ! RX enable
mov r2, 0b10
st r2, r0

//---- read data from table, then write to I2DSP -----
lda r5, data_table      ! read from table
mov %loop0, 40           ! data length
write_i2dsp_loop:
    mov %loop1, 0x200

write_i2dsp_loop1:
    mov r0, I2DSP_STATUS ! read status of TX_FIFO_NFULL
    ld r6, r0
    mov r4, 0x1
    and r6, r4
    cmp r6, r4
    bz write_i2dsp_1      ! TX FIFO is not full, send next one
    agn1 write_i2dsp_loop1

write_i2dsp_1:
    bits %smode,5          !!!TABLE!!!
    lddu r2, r5, 2
    bitc %smode,5

    mov r0,I2DSP_TXDB
    stdu r2, r0, 2          ! r3[15:0], r2[15:8]: 24bits
    agn0 write_i2dsp_loop
```

## Rock2 DATA SHEET V1.1

---

```
//---read data from I2DSP ---  
bits    %smode,0  
mov    %loop0, 40          ! data length  
  
read_i2dsp_loop:  
    mov    r0, I2DSP_STATUS  
    ld     r6, r0  
    mov    r4, 0x2  
    and    r6, r4  
    cmp    r6, r4  
    bz    read_i2dsp_1        ! RX FIFO is not empty, read next one  
    agn1  read_i2dsp_loop1  
  
read_i2dsp_1:  
    mov    r0, I2DSP_RXDB+1    ! read only high 16bits  
    ld     r6, r0  
..... store r6  
    agn0  read_i2dsp_loop
```

ROCKCHIP

## 3.11 I2C

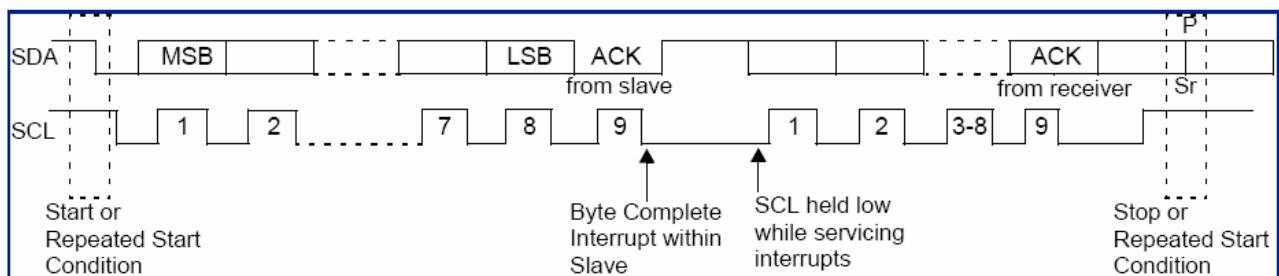
### 3.11.1 Overview

The I2C bus is a two-wire serial interface. The DW\_apb\_i2c module can operate in both standard mode (with data rates up to 100 Kb/s), fast mode (with data rates up to 400 Kb/s). The I2C serial clock determines the transfer rate.

The I2C interface protocol is setup with a master and slave. The master is responsible for generating the clock and controlling the transfer of data. The slave is responsible for either transmitting or receiving data to/from the master. The acknowledgement of data is sent by the device that is receiving data, which can be either the master or the slave. The protocol also allows multiple masters to reside on the I2C bus, which requires the masters to arbitrate for ownership.

The slaves each have a unique address that is determined by the system designer. When the master wants to communicate with a slave, the master transmits a start condition that is then followed by the slave's address and a control bit (R/W) to determine if the master wants to transmit data or receive data from the slave. The slave then sends an acknowledge (ACK) pulse after the address and R/W bit is received to notify the master that the slave has received the request.

If the master (master-transmitter) is writing to the slave (slave-receiver), the receiver receives a byte of data. This transaction continues until the master terminates the transmission with a stop condition. If the master is reading from a slave, the slave transmits a byte of data to the master, and the master then acknowledges the transaction with the ACK pulse. This transaction continues until the master terminates the transmission by not acknowledging the transaction after the last byte is received, and then the master issues a stop condition or addresses another slave after issuing a restart condition. This is illustrated in [Figure 3](#).

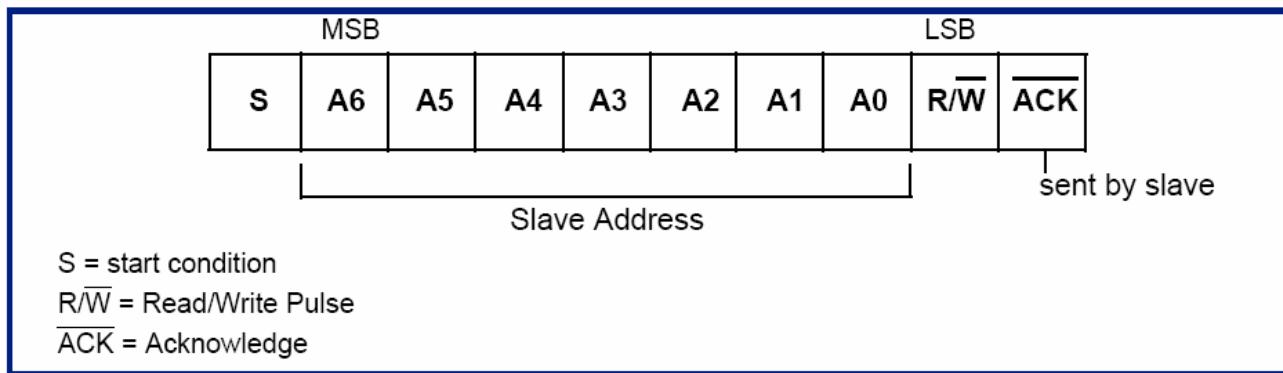


**Figure 3: DW\_apb\_i2c Start and Stop Condition**

The DW\_apb\_i2c is a synchronous serial interface. The data signal (SDA) is a bidirectional signal and changes only while the serial clock signal (SCL) is low. The output drivers are open-drain or open-collector to perform wire-AND functions on the bus. The maximum number of devices on the bus is limited by only the maximum capacitance specification of 400 pF. Data is transmitted in byte packages.

## Addressing Slave Protocol

There are two address formats: the 7-bit address format and the 10-bit address format. During the 7-bit address format, the first seven bits (bits 7:1) of the first byte set the slave address and the LSB bit (bit 0) is the R/W bit as shown in [Figure 5](#). When Bit 8 is set to 0, the master writes to the slave. When Bit 8 (R/W) is set to 1, the master reads from the slave. Data is transmitted most significant bit (MSB) first. During 10-bit addressing, two bytes are transferred to set the 10-bit address. The transfer of the first byte contains the following bit definition. The first five bits (bits 7:3) notify the slaves that this is a 10-bit transfer followed by the next two bits (bits 2:1), which set the slaves address bits 9:8, and the LSB bit (Bit 8) is the R/W bit. The second byte transferred sets bits 7:0 of the slave address. [Figure 6](#) shows the 10-bit address format, and [Table 3 on page 31](#) defines the special purpose and reserved first byte addresses.



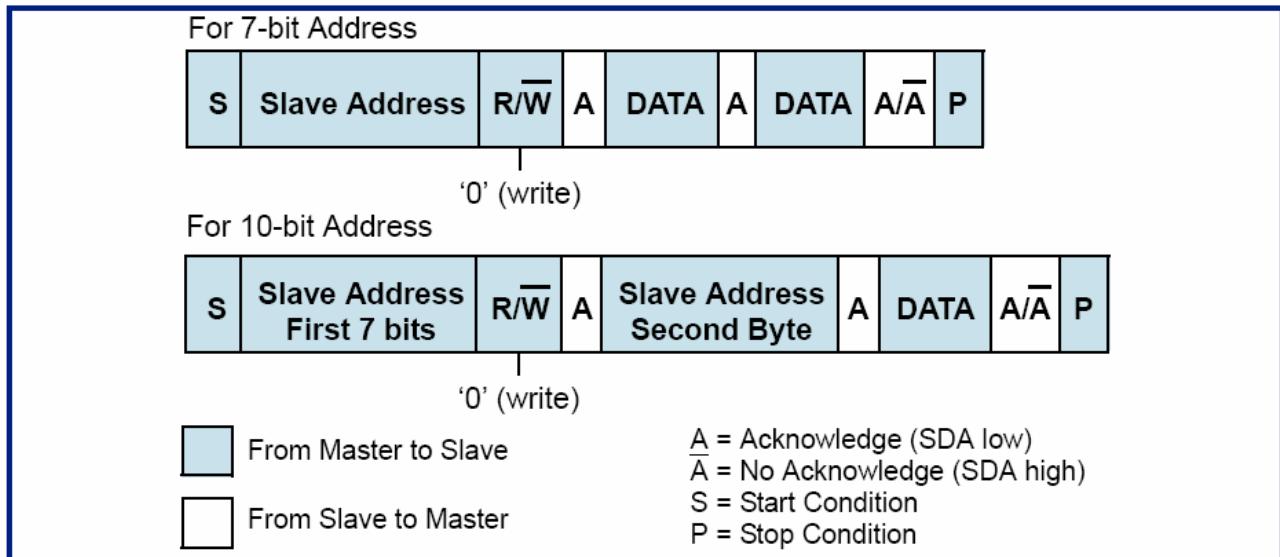
**Figure 5: 7-bit Address Format**

## Transmitting and Receiving Protocol

All data is transmitted in byte format, with no limit on the number of bytes transferred per data transfer. After the master sends the address and R/W bit or the master transmits a byte of data to the slave, the slave-receiver must respond with the acknowledge signal. When a slave-receiver does not respond with an acknowledge pulse, the master aborts the transfer by issuing a STOP condition. The slave shall leave the SDA line high so the master can abort the transfer.

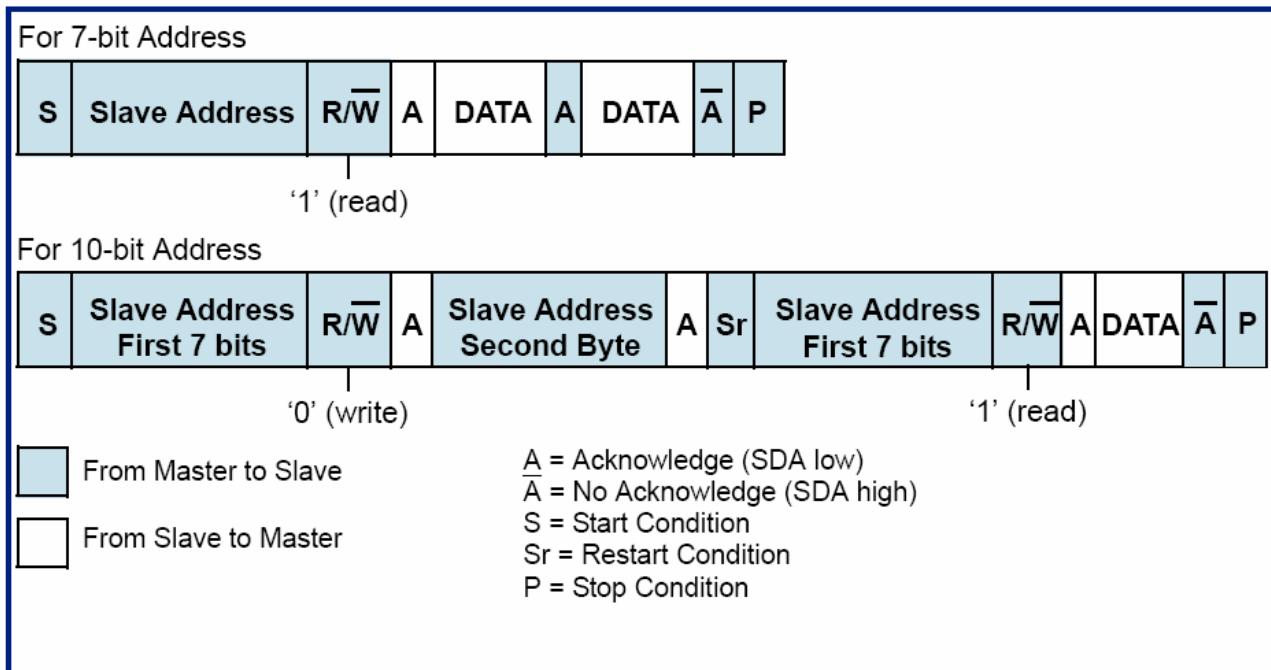
If the master-transmitter is transmitting data as shown in [Figure 7 on page 32](#), then the slave-receiver responds to the master-transmitter with an acknowledge pulse after every byte of data is received.

# Rock2 DATA SHEET V1.1



**Figure 7: Master-Transmitter Protocol**

If the master is receiving data as shown in Figure 8, then the master responds to the slave-transmitter with an acknowledge pulse after a byte of data has been received, except for the last byte. This is the way the master-receiver notifies the slave-transmitter that this is the last byte. The slave-transmitter relinquishes the SDA line after detecting the No Acknowledge so that the master can issue a STOP condition.



## Figure 8: Master-Receiver Protocol

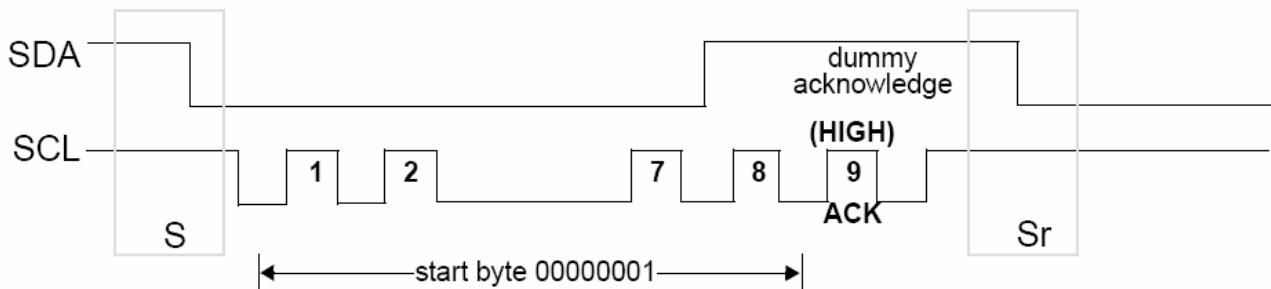
When a master does not want to relinquish the bus with a STOP condition, the master can issue a repeated start condition. This is identical to a START condition except it occurs after the ACK pulse. The master can then communicate with the same slave or a different slave.

### START BYTE Transfer Protocol

The START BYTE transfer protocol is set up for systems that do not have an on board dedicated I<sub>2</sub>C hardware module. When the DW\_apb\_i2c is addressed as a slave, it always samples the I<sub>2</sub>C bus at the highest speed supported so that it never requires a START BYTE transfer. However, when the DW\_apb\_i2c is a master, it supports the generation of START BYTE transfers at the beginning of every transfer in case a slave device requires it. The START BYTE protocol consists of seven zeros being transmitted followed by a 1, as illustrated in [Figure 9](#). This allows the processor that is polling the bus to under-sample the address phase until 0 is detected. Once the microcontroller detects a 0, it switches from the under sampling rate to the correct rate of the master.

The START BYTE procedure is as follows:

1. Master generates a START condition.
2. Master transmits the START byte (0000 0001).
3. Master transmits the ACK clock pulse.
4. No slave sets the ACK signal to 0.
5. Master generates a repeated START (Sr) condition



**Figure 9: START BYTE Transfer**

A hardware receiver does not respond to the START BYTE because it is a reserved address and resets after the Sr (restart condition) is generated.

## Master Mode Operation

### Initial Configuration

To use the DW\_apb\_i2c as a master, perform the following steps:

1. Disable the DW\_apb\_i2c by writing 0 to the [IC\\_ENABLE](#) register.
2. Write to the [IC\\_SAR](#) register to set the slave address, which is the address to which the DW\_apb\_i2c responds.
3. Write to the [IC\\_CON](#) register to set the maximum speed mode supported for slave operation and the desired speed of the DW\_apb\_i2c master-initiated transfers, either 7-bit or 10-bit addressing.
4. Write to the [IC\\_TAR](#) register to the address of the I<sub>2</sub>C device to be addressed. It also indicates whether adding a START BYTE or issuing a general call is going to occur.
5. *Only applicable for high-speed mode transfers.* Write to the [IC\\_HS\\_MADDR](#) register the desired

master code for the DW\_apb\_i2c.

6. Enable the DW\_apb\_i2c with the **IC\_ENABLE** register.

7. Commands and data to be sent may be written now to the **IC\_DATA\_CMD** register. If the **IC\_DATA\_CMD** register is written before the DW\_apb\_i2c is enabled, the data and commands are lost as the buffers are kept cleared when DW\_apb\_i2c is not enabled.

## Master Transmit and Master Receive

The DW\_apb\_i2c supports switching back and forth between reading and writing dynamically. To transmit data, write the data to be written to the lower byte of the IC\_DATA\_CMD register. The CMD bit, Bit 8, should be written to 0 for write operations. Subsequently, a read command may be issued by writing “don’t cares” to the lower byte of the IC\_DATA\_CMD register, and a 1 should be written to the CMD bit. As data is transmitted and received, the transmit and receive buffer status bits and interrupts change.

### 3.11.2 Register

**IC\_CON** I2C Control Register 0

Bits	Name	R/W	Description
15:7	Reserved	N/A	Reserved.
6	IC_SLAVE_DISABLE	R/W	This bit controls whether I2C has its slave disabled after reset. The slave can be disabled by programming a ‘1’ into IC_CON[6]. By default the slave is enabled. 0: slave is enabled 1: slave is disabled <b>Reset:</b> IC_SLAVE_DISABLE
5	IC_RESTART_EN	R/W	Determines whether restart conditions may be sent when acting as a master. Some older slaves do not support handling restart conditions. Restart conditions are used in several DW_apb_i2c operations. Disabling a restart does not allow the master to perform the following functions: • send multiple bytes per transfer (split) • change direction within a transfer (split) • send a start byte • perform any high-speed mode operation • perform combined format transfers in 7- or 10-bit addressing modes (split for 7 bit) • perform a read operation with a 10-bit address Split operations are broken down into multiple DW_apb_i2c transfers with a stop and start condition in between. The other operations are not performed at all and result in setting TX_ABRT. <b>Reset:</b> IC_RESTART_EN
4	IC_10BITADDR_MASTER	R/W	This bit controls whether the DW_apb_i2c starts its transfers in 10-bit addressing mode when acting as a master. 0: 7-bit addressing 1: 10-bit addressing <b>Reset:</b> 0

## Rock2 DATA SHEET V1.1

3	IC_10BITADDR_SLAVE	R/W	When acting as a slave, this bit controls whether the DW_apb_i2c responds to 7- or 10-bit addresses. 0: 7-bit addressing. 1: 10-bit addressing. <b>Reset:</b> 0
2:1	SPEED	R/W	Controls at which speed the DW_apb_i2c operates: 0: illegal; 1: standard mode (100 kbit/s) 2: fast mode (400 kbit/s) <b>Reset:</b> 2
0	MASTER_MODE	R/W	This bit controls whether the DW_apb_i2c master is enabled or not. The slave is always enabled. 0: master disabled 1: master enabled <b>Reset:</b> IC_MASTER_MODE

### IC\_TAR I2C Target Address Register

Bits	Name	R/W	Description
15:12	Reserved	N/A	Reserved.
11	SPECIAL	R/W	This bit indicates whether software would like to perform a general call or start byte I2C command. 0: ignore bit 10 GC_OR_START and use IC_TAR normally 1: perform special I2C command as specified in GC_OR_START bit <b>Reset:</b> 0x0
10	GC_OR_START	R/W	If bit 11 SPECIAL is set to 1, then this bit indicates whether a general call or start byte command is to be performed by the DW_apb_i2c. 0: General Call Address – after issuing a general call, only writes may be performed. Attempting to issue a read command results in setting TX_ABRT. The DW_apb_i2c remains in general call mode until the SPECIAL bit value is cleared. 1: Start Byte <b>Reset:</b> 0x0
9:0	IC_TAR	R/W	This is the target address for any master transactions. <b>Reset:</b> IC_DEFAULT_TAR_SLAVE_ADDR, which indicates loopback mode

### IC\_SAR: I2C Slave Address Register

Bits	Name	R/W	Description
15:10	Reserved	N/A	Reserved.

## Rock2 DATA SHEET V1.1

---

9:0	IC_SAR	R/W	The IC_SAR holds the slave address when the I2C is operating as a slave. IC_SAR holds the slave address to which the DW_apb_i2c responds. For 7-bit addressing, only IC_SAR[6:0] is used. This register can be written only when the I2C interface is disabled, which corresponds to the IC_ENABLE register being set to 0. Writes at other times have no effect. <b>Reset:</b> 0x55
-----	--------	-----	---

### IC\_DATA\_CMD I2C RX/TX Data Buffer and Command Register

Bits	Name	R/W	Description
15:9	Reserved	N/A	Reserved
8	CMD	R/W	This bit controls whether a read or a write is performed. 1 = Read. 0 = Write. For reads, the lower 8 (DAT) bits are ignored by the DW_apb_i2c. However, if the APB_DATA_WIDTH is 8, this “dummy” write is still required as there is coherency in this register. Reading this bit returns 0. Attempting to perform a read operation after a general call command has been sent results in TX_ABRT unless the SPECIALbit in the IC_TARregister has been cleared. If this bit is written to a ‘1’ after receiving RD_REQ, then a TX_ABRT occurs. <b>Reset:</b> 0x10
7:0	DAT	R/W	This register contains the data to be transmitted or received on the I2C bus. Read these bits to read out the data received on the I2C interface. Write these bits to send data out on the I2C interface. <b>Reset:</b> 0x0

### IC\_SS\_SCL\_HCNT: Standard Speed I2C Clock SCL High Count Register

Bits	Name	R/W	Description
15:0	IC_SS_SCL_HCNT	R/W1	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for standard speed. The minimum valid value is 6; hardware prevents values less than this being written, and if attempted results in 6 being set. <b>Reset:</b> 0x64

### IC\_SS\_SCL\_LCNT: Standard Speed I2C Clock SCL Low Count Register

## Rock2 DATA SHEET V1.1

Bits	Name	R/W	Description
15:0	<i>IC_SS_SCL_LCNT</i>	R/W1	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for standard speed. The minimum valid value is 8; hardware prevents values less than this being written, and if attempted results in 8 being set. <b>Reset:</b> 0x80

**IC\_FS\_SCL\_HCNT:** Fast Speed I2C Clock SCL High Count Register

Bits	Name	R/W	Description
15:0	<i>IC_FS_SCL_HCNT</i>	R/W1	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock high-period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. The minimum valid value is 6; <b>Reset:</b> 0x10

**IC\_FS\_SCL\_LCNT:** Fast Speed I2C Clock SCL Low Count Register

Bits	Name	R/W	Description
15:0	<i>IC_FS_SCL_LCNT</i>	R/W1	This register must be set before any I2C bus transaction can take place to ensure proper I/O timing. This register sets the SCL clock low period count for fast speed. It is used in high-speed mode to send the Master Code and START BYTE or General CALL. The minimum valid value is 8; <b>Reset:</b> 0x22

**IC\_RX\_TL:** I2C Receive FIFO Threshold Register

Bits	Name	R/W	Description
15:8	Reserved	N/A	Reserved.
7:0	<i>RX_TL</i>	R/W	Receive FIFO Threshold Level Controls the level of entries (or above) that triggers the RX_FULL interrupt. The valid range is 0-3, with the additional restriction that hardware does not allow this value to be set to a value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 1 entry, and a value of 3 sets the threshold for 4 entries. <b>Reset:</b> IC_RX_TL=2

## Rock2 DATA SHEET V1.1

---

### **IC\_TX\_TL:** I2C Transmit FIFO Threshold Register

Bits	Name	R/W	Description
15:8	Reserved	N/A	Reserved.
7:0	<i>TX_TL</i>	R/W	<p>Transmit FIFO Threshold Level Controls the level of entries (or below) that trigger the TX_EMPTY interrupt. The valid range is 0-3, with the additional restriction that it may not be set to value larger than the depth of the buffer. If an attempt is made to do that, the actual value set will be the maximum depth of the buffer. A value of 0 sets the threshold for 0 entries, and a value of 3 sets the threshold for 4 entries.</p> <p><b>Reset:</b> IC_TX_TL=1</p>

### **IC\_ENABLE:** I2C Enable Register

Bits	Name	R/W	Description
15:1	Reserved	N/A	Reserved.
0	<i>ENABLE</i>	R/W	<p>Controls whether the DW_apb_i2cis enabled. Writing a 1 enables the DW_apb_i2c, and writing a 0 disables it. Software should not disable the DW_apb_i2cwhile it is active. The <i>ACTIVITY</i> bit can be polled to determine if the DW_apb_i2cis active.</p> <p>If the module was transmitting, it stops as well as deletes the contents of the transmit buffer after the current transfer is complete. If the module was receiving, the DW_apb_i2cstops the current transfer at the end of the current byte and does not acknowledge the transfer.</p> <p>In systems with asynchronous pclk and ic_clk (IC_CLK_TYPE = 1), there is a two ic_clk delay when enabling or disabling the DW_apb_i2c.</p> <p><b>Reset:</b>0x0</p>

### **IC\_STATUS:** I2C Status Register

Bits	Name	R/W	Description
4	<i>RFF</i>	R	Receive FIFO Completely Full. When the receive FIFO is completely full, this bit is set. When the receive FIFO contains one or more empty location, this bit is cleared. 0 – Receive FIFO is not full 1 – Receive FIFO is full <b>Reset:</b> 0x0
3	<i>RFNE</i>	R	Receive FIFO Not Empty. Set when the receive FIFO contains one or more entries and is cleared when the receive FIFO is empty. This bit can be polled by software to completely empty the receive FIFO. 0 – Receive FIFO is empty 1 – Receive FIFO is not empty <b>Reset:</b> 0x0

## Rock2 DATA SHEET V1.1

---

2	<i>TFE</i>	R	Transmit FIFO Completely Empty. When the transmit FIFO is completely empty, this bit is set. When it contains one or more valid entries, this bit is cleared. This bit field does not request an interrupt. 0 – Transmit FIFO is not empty 1 – Transmit FIFO is empty <b>Reset:0x1</b>
1	<i>TFNF</i>	R	Transmit FIFO Not Full. Set when the transmit FIFO contains one or more empty locations, and is cleared when the FIFO is full. 0 – Transmit FIFO is full 1 – Transmit FIFO is not full <b>Reset:0x1</b>
0	<i>ACTIVITY</i>	R	I2C Activity Status. <b>Reset:0x0</b>

---

### 3.11.3 Example Software

```

bits      %smode,0
mov      r0, IC_ENABLE    ! Disable I2C
mov      r2, 0x0
st       r2, r0

mov      r0, IC_SAR        ! I2C slave addr: 10bits:[9:0], or 7bits:[6:0]
mov      r2, 0x55
st       r2, r0

mov      r0, IC_CON        ! [6]:slave_disable  [5]:restart_en  [4]:10bit_master
                  ! [3]:10bit_slave   [2:1]: speed   [0]:master
mov      r2, 0b1100101     ! dis, rest_en, 7bit, 7bit, fast, master
st       r2, r0

mov      r0,IC_TAR         ! [11]: special   [10]: GC_STAR    [9:0]: I2C_TAR
mov      r2, 0x55
st       r2, r0

mov      r0, IC_FS_HCNT   ! 0x10, (min:6)
mov      r2, 0x6
st       r2, r0

mov      r0, IC_FS_LCNT   ! 0x22, (min:8)
mov      r2, 0x8
st       r2, r0

mov      r0, IC_ENABLE     ! Enable I2C
mov      r2, 0x1
st       r2, r0

mov      r0, IC_DATA_CMD   ! write addr
mov      r2, 0x30
st       r2, r0

mov      r0, IC_DATA_CMD   !write data
mov      r2, 0x33
st       r2, r0
call    delay_300nop      ! wait for ready
.....
mov      r0, IC_DATA_CMD   ! write addr of CCR
mov      r2, P_ADD_CCR

```

## Rock2 DATA SHEET V1.1

```
st      r2, r0
mov    r0, IC_DATA_CMD
mov    r2, 0b00010010      ! write data to CCR: 44.1k dac and 32k adc
st      r2, r0
call   delay_300nop

mov    r0, IC_DATA_CMD
mov    r2, P_ADD_CR2      ! write addr of CR2
st      r2, r0
mov    r0, IC_DATA_CMD      ! write data to CR2
mov    r2, 0b01011000      ! [6:5]: dac 20bits; [4:3]: adc 24bits;
st      r2, r0      ! [2]: adc_hp; [1:0]: insel line1
call   delay_300nop

mov    r5, P_ADD_CR2
mov    r0, IC_DATA_CMD
st      r5, r0      !write CMD addr
mov    r7,0x100      !read CMD
st      r7,r0
call   delay_300nop      !wait for data ready

wait123:
mov    r0, IC_STATUS      ! [4]:RFF, [3]:RFNE, [2]:TFE, [1]TFNF, [0]:Activity
ld     r2, r0

mov    r3, 0b01000      ! test RFNE
and   r2, r3
cmp   r2, r3
bz    check_i2c_rxne_ok
br    wait123      ! ready

check_i2c_rxne_ok:
mov    r0, IC_DATA_CMD
ld     r7, r0      !read data to DSP
```

## 3.12 CODEC

### 3.12.1 Overview

- Single +1.8V (from 1.6V to 2.0V) power supply for the analog part
- Separate power-down modes for ADC and DAC
- 24-/20-/18-/16-bit programmable word length
- Serial audio interface I2DSP for digital audio data, I2C bridge for control data
- 2 stereo line inputs and one MIC input with 85dB A-Weighted SNR
- Input gain range from 0dB to 22.5dB with 1.5dB steps (4-bit programmable gain)
- Output load down to 16 Ohm with capacitor less connection and 90dB A-Weighted SNR
- Output DAC path gain range from +6dB to -32dB (5-bit programmable gain)
- Mixer function with programmable gains
- Programmable sampling frequency Fs between 8/11.025/16/22.05/32/44.1/48 kHz

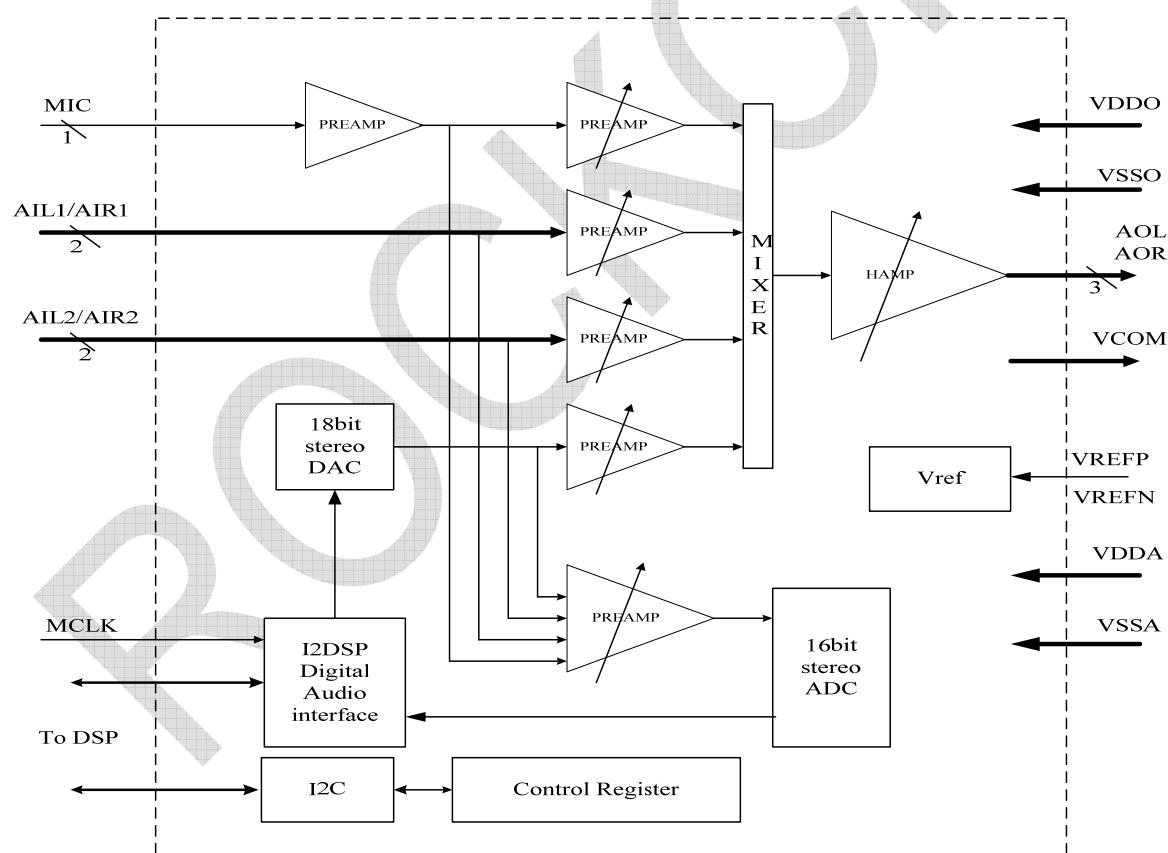


Fig1. CODEC block diagram

# Rock2 DATA SHEET V1.1

## 3.12.2 Register

### *AICR: Audio Interface Control Register*

R/W-0	R/W-1	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
-	-	-	-	-	-	<b>DAC_I2S</b>	<b>ADC_I2S</b>	

Bit 7

Bit 1: **DAC\_I2S**: Working mode of the DAC digital serial audio interface  
0= DSP mode, only DSP mode

Bit 0: **ADC\_I2S**: Working mode of the ADC digital serial audio interface  
0= DSP mode, only DSP mode

Bit 0

### *CR1: Control Register 1*

R-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-0
<b>JACK</b>	<b>MONO</b>	<b>DAC_MUTE</b>	<b>DAC_DEEMPH</b>	<b>DACSEL</b>	<b>BYPASS1</b>	<b>BYPASS2</b>	<b>SIDETONE</b>

Bit 7: **JACK**: Output Jack plug detection status  
0= no jack 1= output jack present

Bit 6: **MONO**: Stereo-to-mono conversion for DAC path  
0= stereo 1= mono

Bit 5: **DAC\_MUTE**: DAC soft mute mode  
0= inactive 1= puts the DAC in soft mute mode (internal input MUTE)

Bit 4: **DAC\_DEEMPH**: DAC De-emphasize filter enable  
0= inactive 1= drives the internal input DEEMPH

Bit 3: **DACSEL**: Mixer input selection  
0= DAC output ignored in input of the mixer  
1= DAC output selected as an input of the mixer (internal input DACSEL)

Bit 2: **BYPASS1**: Mixer input selection (line 1)  
0= Bypass path ignored in input of the mixer  
1= Bypass path selected as an input of the mixer (internal input DACSEL)

Bit 1: **BYPASS2**: Mixer input selection (line 2)  
0= Bypass path ignored in input of the mixer  
1= Bypass path selected as an input of the mixer (internal input DACSEL)

Bit 0: **SIDETONE**: Mixer input selection  
0= Sidetone path ignored in input of the mixer  
1= Sidetone path selected as an input of the mixer (internal input DACSEL)

## Rock2 DATA SHEET V1.1

---

### **CR2: Control Register 2**

R-0	R/W-1	R/W-1	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0
CCMC	DAC_AD WL1	DAC_AD WL0	ADC_AD WL1	ADC_AD WL0-	ADC_HPF	INSEL1	INSEL0

Bit 7: **CCMC**: Output short circuit detection status – Reserved for future use

0= inactive

1= indicates that a short circuit has been detected by the output stage. The conditions that reset the flag are given in the section 4.10.

Bit 6-3: **DAC\_ADWL[1:0], ADC\_ADWL[1:0]**: Audio Data Word Length: (internal input ADWL) for respectively DAC and ADC paths

00 = 16-bit word length data      01 = 18-bit word length data

10 = 20-bit word length data      11 = 24-bit word length data

Bit 2: **ADC\_HPF**: ADC High Pass Filter enable

0= inactive

1= enables the ADC High Pass Filter (internal input HPF\_EN)

Bit 1-0: **INSEL[1:0]**: selection of the signal converted by the ADC

00 = Line 1 input      01 = Line 2 input

10 = Microphone input      11 = Mixer output

### **CCR: Control Clock Register**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
QUARTZ	DFREQ2	DFREQ1	DFREQ0	-	AFREQ2	AFREQ1	AFREQ0

Bit 7: **QUARTZ**: Selection of the MCLK frequency

0= 12 MHz, only 12MHz can be select

Bit 6-4: **DFREQ[2:0]**: Selection of the DAC sampling rate (Fs)

The sampling frequency value is given in the FREQ[2:0] table

Bit 2-0: **AFREQ[2:0]**: Selection of the ADC sampling rate (Fs)

The sampling frequency value is given in the FREQ[2:0] table

FREQ[2:0]	Sampling Rate (Fs)
000	48kHz
001	44.1kHz
010	32kHz
011	22.05kHz
100	16kHz
101	11.025kHz
110	8kHz
111	8kHz

# Rock2 DATA SHEET V1.1

---

## **PMR1: Power Mode Register 1**

R/W-0	R/W-0	R/W-1	R/W-0	R/W-0	R/W-1	R/W-1	R/W-1
<b>SB_DAC</b>	<b>SB_OUT</b>	<b>SB_MC</b>	<b>SB_ADC</b>	<b>SB_IN1</b>	<b>SB_IN2</b>	<b>SB_MIC</b>	<b>SB_IND</b>

Bit 7: **SB\_DAC**: DAC power-down mode

0= active      1= power-down

Bit 6: **SB\_OUT**: Output Stage and Mixer power-down mode

0= active      1= power-down

Bit 5: **SB\_MC**: Output Stage common mode buffer power-down

0= active (capacitor less headphone output configuration)

1= power-down (line output configuration)

Bit 4: **SB\_ADC**: ADC power-down mode

0= active      1= power-down

Bit 3: **SB\_IN1**: Line 1 Input conditioning circuitry power-down mode

0= active      1= power-down

Bit 2: **SB\_IN2**: Line 2 Input conditioning circuitry power-down mode

0= active      1= power-down

Bit 1: **SB\_MIC**: Microphone Input conditioning circuitry power-down mode

0= active      1= power-down

Bit 0: **SB\_IND**: Mixer to ADC circuitry power-down mode

0= active      1= power-down

## **PMR2: Power Mode Register 2**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
<b>LRGI</b>	<b>RLGI</b>	<b>LRGOD</b>	<b>RLGOD</b>	<b>GIM</b>	-	<b>SB</b>	<b>SB_SLEEP</b>

Bit 7-6: **LRGI, RLGI**: PGATM input gain coupling

00: Left and right channels gains are independent, respectively given by GIL and GIR

10: Left and right channels gain is given by GIL

01: Left and right channels gain is given by GIR

11: Left and right channels gain is given by GIL

Bit 5-4: **LRGOD, RLGOD**: DAC mixing gain coupling

00: Left and right channels gains are independent, respectively given by GODL and GODR

10: Left and right channels gain is given by GODL

01: Left and right channels gain is given by GODR

11: Left and right channels gain is given by GODL

Bit 3: **GIM**: Microphone amplifier gain control

0= 20 dB gain      1= 0 dB gain

# Rock2 DATA SHEET V1.1

---

Bit 1: **SB**: complete power-down mode  
0= normal mode (active)      1= complete power-down

Bit 0 **SB\_SLEEP**: sleep mode  
0= normal mode (active)      1= sleep mode

## *CGR1: Control Gain Register n°1*

| R/W-0        |
|--------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| <b>GODR3</b> | <b>GODR2</b> | <b>GODR1</b> | <b>GODR0</b> | <b>GODL3</b> | <b>GODL2</b> | <b>GODL1</b> | <b>GODL0</b> |

Bit 7-4: GODR[3:0]: DAC mixing right channel gain programming value

Bit 3-0: GODL[3:0]: DAC mixing left channel gain programming value

## *CGR2: Control Gain Register n°2*

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
<b>LRGOB1</b>	<b>RLGOB1</b>	-	<b>GOBL14</b>	<b>GOBL13</b>	<b>GOBL12</b>	<b>GOBL11</b>	<b>GOBL10</b>

Bit 7-6: **LRGOB1, RLGOB1**: Line 1 mixing gain coupling

00: Left and right channels gains are independent, respectively given by GOBL1 and GOBR1

10: Left and right channels gain is given by GOBL1

01: Left and right channels gain is given by GOBR1

11: Left and right channels gain is given by GOBL1

Bit 4-0: **GOBL1[4:0]**: Line 1 mixing left channel gain programming value

## *CGR3: Control Gain Register n°3*

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
			<b>GOBR14</b>	<b>GOBR13</b>	<b>GOBR12</b>	<b>GOBR11</b>	<b>GOBR10</b>

Bit 4-0: **GOBR1[4:0]**: Line 1 mixing right channel gain programming value

## *CGR4: Control Gain Register n°4*

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
<b>LRGOB2</b>	<b>RLGOB2</b>		<b>GOBL24</b>	<b>GOBL23</b>	<b>GOBL22</b>	<b>GOBL21</b>	<b>GOBL20</b>

Bit 7-6: **LRGOB2, RLGOB2**: Line 2 mixing gain coupling

00: Left and right channels gains are independent, respectively given by GOBL2 and GOBR2

10: Left and right channels gain is given by GOBL2

01: Left and right channels gain is given by GOBR2

11: Left and right channels gain is given by GOBL2

Bit 4-0: **GOBL2[4:0]**: Line 2 mixing left channel gain programming value

## Rock2 DATA SHEET V1.1

---

### **CGR5: Control Gain Register n°5**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
			<b>GOBR24</b>	<b>GOBR23</b>	<b>GOBR22</b>	<b>GOBR21</b>	<b>GOBR20</b>

Bit 4-0: **GOBR2[4:0]**: Line 2 mixing right channel gain programming value

### **CGR6: Control Gain Register n°6**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
<b>LRGOS</b>	<b>RLGOS</b>		<b>GOSL4</b>	<b>GOSL3</b>	<b>GOSL2</b>	<b>GOSL1</b>	<b>GOSL0</b>

Bit 7-6: **LRGOS, RLGOS**: Microphone mixing gain coupling

00: Left and right channels gains are independent, respectively given by GOSL and GOSR

10: Left and right channels gain is given by GOSL

01: Left and right channels gain is given by GOSR

11: Left and right channels gain is given by GOSL

Bit 4-0: **GOSL[4:0]**: Microphone mixing left channel gain programming value

### **CGR7: Control Gain Register n°7**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-0
			<b>GOSR4</b>	<b>GOSR3</b>	<b>GOSR2</b>	<b>GOSR1</b>	<b>GOSR0</b>

Bit 4-0: **GOSR[4:0]**: Microphone mixing right channel gain programming value

### **CGR8: Control Gain Register n°8**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1	R/W-0
<b>LRGO</b>	<b>RLGO</b>		<b>GOL4</b>	<b>GOL3</b>	<b>GOL2</b>	<b>GOL1</b>	<b>GOL0</b>

Bit 7-6: **LRGO, RLGO**: Output stages gain coupling

00: Left and right channels gains are independent, respectively given by GOL and GOR

10: Left and right channels gain is given by GOL

01: Left and right channels gain is given by GOR

11: Left and right channels gain is given by GOL

Bit 4-0: **GOL[4:0]**: Output stage left channel gain programming value

### **CGR9: Control Gain Register n°9**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-1	R/W-0	R/W-1	R/W-0
			<b>GOR4</b>	<b>GOR3</b>	<b>GOR2</b>	<b>GOR1</b>	<b>GOR0</b>

Bit 4-0: **GOR[4:0]**: Output stage right channel gain programming value

## Rock2 DATA SHEET V1.1

### **CGR10: Control Gain Register n°10**

| R/W-0 |
|-------|-------|-------|-------|-------|-------|-------|-------|
| GIL3  | GIL2  | GIL1  | GIL0  | GIR3  | GIR2  | GIR1  | GIR0  |

Bit 7-4: **GIL[3:0]**: ADC left channel PGATM input gain programming value

Drives the internal bus GIL[3:0]

Bit 3-0: **GIR[3:0]**: ADC right channel PGATM input gain programming value

Drives the internal bus GIR[3:0]

### **TR1: Test Register n°1**

R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0	R/W-0
STBYO	STBYI	TSTDAC	TSTADC	TEST	STOPULL	NOSC	CLKDIV4

Bit 7: **STBYO**: drives the internal input STBYO

0 in normal mode

Bit 6: **STBYI**: drives the internal input STBYI

0 in normal mode

Bit 5: **TSTDAC**: drives the internal input TSTDAC

0 in normal mode

Bit 4: **TSTADC**: drives the internal input TSTADC

0 in normal mode

Bit 3: **TEST**: Test mode

0 in normal mode

Bit 2: **STOPULL**: disables the input circuitry starting system

0 in normal mode

Bit 1: **NOSC**: disable the output short circuit protection

0 in normal mode

Bit 0: **CLKDIV4**: Clock test mode

0 in normal mode

### **TR2: Test Register n°2**

R/W-1	R/W-1	R/W-0	R/W-1	R/W-1	R/W-0	R/W-0	R/W-0
ECHOPD AC	FAENDAC	NENCOM P	FAENADC	ECHOPA DC	HIPAS	NO_RST	UNSTBL

Bit 7: **ECHOPDAC**: DAC chopper control

1 in normal mode

# Rock2 DATA SHEET V1.1

Bit 6: **FAENDAC**: Flow adapter command control bit (DAC path)

0: inactive

1: enables the flow adapter working mode

Bit 7: **NENCOMP**: Biasing bit control

0 in normal mode

Bit 4: **FAENADC**: Flow adapter command control bit (ADC path)

0: inactive

1: enables the flow adapter working mode

Bit 3: **ECHOPADC**: ADC chopper control

1 in normal mode

Bit 2: **HIPAS**: ADC NTF test control

0 in normal mode

Bit 1: **NO\_RST**: ADC auto reset control

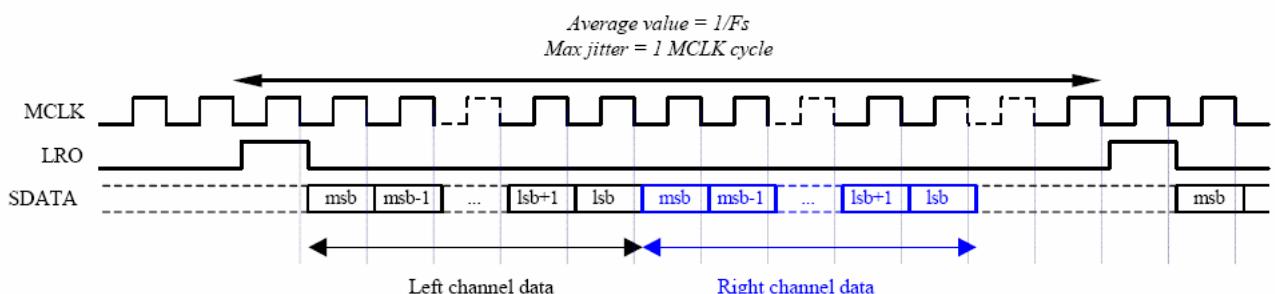
0 in normal mode

Bit 0: **UNSTBL**: ADC unstability status

0: reset of the status bit

### 3.12.3 Interface: I2DSP mode

DAC or ADC LRO is a one MCLK cycle high-level pulse. DAC or ADC SDATA transmission has to start one MCLK cycle after LRO rising edge, left channel first, from MSB to LSB, then right channel, from MSB to LSB. There is no delay between left and right channel transmission, so word length (16, 18, 20 or 24 bits data width) must be taken into account when emitting or receiving data.



### 3.12.4 Soft mute mode

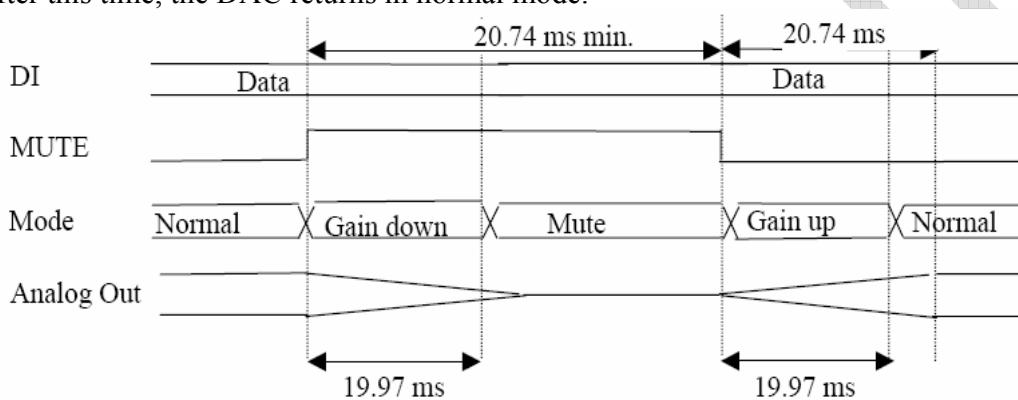
A soft mute is implemented in order to reduce audible parasites when the DAC enters or leaves the mute mode.

The MUTE signal is internally synchronized by a MCLK rising edge.

A high level puts the DAC in mute mode. This decreases progressively the gain in the digital part (DWL block) from 0dB to  $-\infty$  during 19.97 ms. After this time, the signal in output of the DAC is equal to 0 whatever the value of the digital input data.

During soft mute mode, the DAC is still converting but the output final voltages (AOUTL, AOUTR) are equal to VREF/2.

In the opposite, when the MUTE input is tied to a low level again, the DAC leaves the mute mode by increasing progressively the gain in the digital part from  $-\infty$  to 0dB during 19.97 ms. After this time, the DAC returns in normal mode.



Do not change the level of MUTE while the effect of the previous change is not reached, i.e. 20.48 ms (working not guaranteed).

When the DAC is in the gain up mode, do not enter in stand-by mode while the DAC has not completely settled to the normal mode, i.e. 20.74 ms (working not guaranteed).

### 3.12.5 Power-down and sleep mode

Ten stand-by inputs allow putting independently the different parts of hiCODlv-9001 in power-down mode. Two other stand-by inputs are available for test purpose: STBYI and STBYO.

Different working modes are sum-up in the following table (non exhaustive table):

## Rock2 DATA SHEET V1.1

SB	SB_SLEEP	SB_DAC	SB_OUT	SB_MC	SB_ADC	SB_MIC	SB_IN1	SB_IN2	SB_IND	Mode
1	X	X	X	X	X	X	X	X	X	Complete power-down mode
0	1	X	X	X	X	X	X	X	X	Sleep mode
0	0	X	X	X	0	X	0	X	X	Record mode, Line input 1
0	0	X	X	X	0	X	X	0	X	Record mode, Line input 2
0	0	X	X	X	0	0	X	X	X	Record mode, Microphone input
0	0	X	0	X	0	X	X	X	0	Record mode, Mixer output
0	0	0	0	0	X	X	X	X	X	Playback mode, DAC to 16 Ohm headphone out
0	0	0	0	1	X	X	X	X	X	Playback mode, DAC to 10 kOhm line out
0	0	X	0	0	X	X	0	X	X	Playback mode, Line1 to 16 Ohm headphone out
0	0	X	0	1	X	X	0	X	X	Playback mode, Line1 to 10 kOhm line out
0	0	X	0	0	X	X	X	0	X	Playback mode, Line2 to 16 Ohm headphone out
0	0	X	0	1	X	X	X	0	X	Playback mode, Line2 to 10 kOhm line out
0	0	X	0	0	X	0	X	X	X	Playback mode, MIC to 16 Ohm headphone out
0	0	X	0	1	X	0	X	X	X	Playback mode, MIC to 10 kOhm line out
0	0	0	0	0	X	0	0	0	X	Playback mode, all inputs mixed to 16 Ohm headphone out
0	0	0	0	1	X	0	0	0	X	Playback mode, all inputs mixed to 10 kOhm line out
0	0	0	0	0	0	0	0	0	X	Playback mode, all inputs mixed to 16 Ohm headphone out, record line or mic input
0	0	0	0	1	0	0	0	0	X	Playback mode, all inputs mixed to 10 kOhm line out, record line or mic input
0	0	0	0	0	0	0	0	0	0	Playback mode, all inputs mixed to 16 Ohm headphone out, record mixer output
0	0	0	0	1	0	0	0	0	0	Playback mode, all inputs mixed to 10 kOhm line out, record mixer output
0	0	0	0	0	0	1	0	1	1	Default mode

During the sleep mode and the complete power-down mode, the main clock MCLK may be stopped to reduce the power consumption to the leakage currents only. In other modes, the main clock MCLK must not be stopped.

## 3.13 GPIO PORTS

### 3.13.1 Overview

Rock2 has 40 multi-functional GPIO (general-purpose input/output) port pins organized into 4 port groups:

Each port can be easily configured by software to meet various system configuration and design requirements.

These multi-functional pins need to be properly configured before their use. If a multiplexed pin is not used as a dedicated functional pin, this pin can be configured as GPIO ports.

The initial pin states, before pin configurations, are configured elegantly to avoid some problems.

### 3.13.2 Block diagram

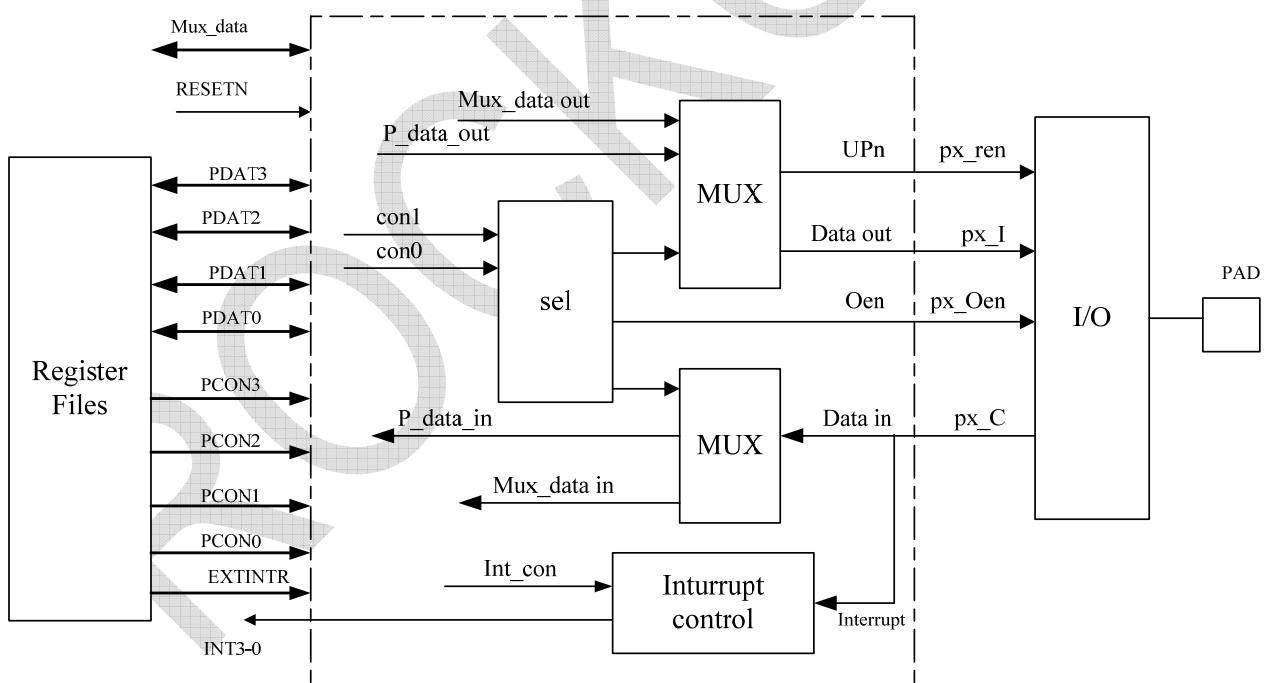


Figure 2. Block Diagram

## Rock2 DATA SHEET V1.1

---

### 3.13.3 Block I/Os

#### I/O DESCRIPTION

Register Files interface		
PCON0[16:0]	I	Port0 control
PDAT0[7:0]	I/O	Port0 Data
PCon1[16:0]	I	Port1 control
PDAT1[7:0]	I/O	Port1 Data
PCon2A[16:0]	I	Port2 control
PDAT2[15:0]	I/O	Port2 Data
PCon2B[16:0]	I	Port2 control
PCon3[16:0]	I	Port3 control
PDAT3[7:0]	I/O	Port3 Data
EXTINTR [7:0]	I	Interrupt control
Register File interface I/Os total:		
Application interface		
P0(8)	I/O	P0 Port
P1(8)	I/O	P1 Port
P2(16)	I/O	P2 Port
P3(8)	I/O	P3 Port

**Table 1. Port Configuration Overview**

Port 0		Selectable Pin Functions						
		Function 1		Function 2			Function 3	
Pin No		GPIO	Pin name	I/O	Module	Pin name	I/O	Module
P0.0	55	Input/output	INT0	I				
P0.1	56	Input/output	INT1	I				
P0.2	57	Input/output	INT2	I				
P0.3	58	Input/output	INT3	I				
P0.4	59	Input/output						
P0.5	60	Input/output						
P0.6	61	Input/output						
P0.7	62	Input/output						

## Rock2 DATA SHEET V1.1

---

Port 1		Selectable Pin Functions						
		Function 1	Function 2			Function 3		
Pin No.		GPIO	Pin name	I/O	Module	Pin name	I/O	Module
P1.0	65	Input/output	NORCS	O	NOR FLASH			
P1.1	66	Input/output	SDA	I/O	I2C			
P1.2	67	Input/output	SCL	I/O	I2C			
P1.3	68	Input/output	I2SMCLK	O	I2D			
P1.4	69	Input/output	DAC_LRCK	I/O	I2D			
P1.5	70	Input/output	ADC_LRCK	I/O	I2D			
P1.6	71	Input/output	SDI	I	I2D			
P1.7	72	Input/output	SDO	O	I2D			

Port 2		Selectable Pin Functions						
		Function 1	Function 2			Function 3		
Pin No.		GPIO	Pin name	I/O	Module	Pin name	I/O	Module
P2.0	22	Input/output	SDDO	I	SDMMC			
P2.1	23	Input/output	SDDI	O	SDMMC			
P2.2	24	Input/output	SDCLK	O	SDMMC			
P2.3	25	Input/output	SDCS	O	SDMMC	FCE1	O	FLASH
P2.4	26	Input/output	A0	O	SDRAM			
P2.5	27	Input/output	A1	O	SDRAM			
P2.6	28	Input/output	A2	O	SDRAM			
P2.7	29	Input/output	A3	O	SDRAM			
P2.8	43	Input/output	WEN	O	SDRAM			
P2.9	44	Input/output	CASN	O	SDRAM			
P2.10	45	Input/output	RASN	O	SDRAM			
P2.11	46	Input/output	CSN	O	SDRAM			
P2.12	47	Input/output						
P2.13	48	Input/output	PWM0	O	PWM			
P2.14	49	Input/output	PWM1	O	PWM			
P2.15	50	Input/output	PWM2	O	PWM			

Port 3		Selectable Pin Functions						
		Function 1		Function 2		Function 3		
Pin No.		GPIO	Pin name	I/O	Module	Pin name	I/O	Module
P3.0	120	Input/output	D8	I/O				
P3.1	121	Input/output	D9	I/O				
P3.2	122	Input/output	D10	I/O				
P3.3	123	Input/output	D11	I/O				
P3.4	124	Input/output	D12	I/O				
P3.5	125	Input/output	D13	I/O				
P3.6	126	Input/output	D14	I/O				
P3.7	127	Input/output	D15	I/O				

### 3.13.4 Register Descriptions

#### PORT CONTROL DESCRIPTIONS

##### Port Configuration Register (PCON0 – PCON3)

In Rock2, most pins are multiplexed, and the PCONn (port control register) determines which function is used for each pin.

##### Port Data Register (PDAT0 – PDAT3)

If Ports are configured as output ports, data can be written to the corresponding bit of PDATn.

If Ports are configured as input ports, the data can be read from the corresponding bit of PDATn.

##### External Interrupt Control Register

The Port0.0-3 external interrupts support various trigger mode: the trigger mode can be configured as falling-edge trigger and rising-edge trigger.

Because each external interrupt pin has an integrated digital noise filter, the interrupt controller can recognize the request signal that lasts longer than 2 clocks.

## Rock2 DATA SHEET V1.1

---

### GPIO PORTS SPECIAL FUNCTION REGISTERS

Register	Address	R/W	Description	Reset Value
PCON0	0x0001_EE80	R/W	Configures the pins of port 0	0x0000 0000
PDAT0	0x0001_EE88	R/W	The data register for port 0	Undefined
PCON1	0x0001_EE90	R/W	Configures the pins of port 1	0x0000 0000
PDAT1	0x0001_EE98	R/W	The data register for port 1	Undefined
PCON2A	0x0001_EEA0	R/W	Configures the pins of port 2[7:0]	0x0000 0000
PCON2B	0x0001_EEA4	R/W	Configures the pins of port 2[15:8]	0x0000 0000
PDAT2	0x0001_EEA8	R/W	The data register for port 2	Undefined
PCON3	0x0001_EEB0	R/W	Configures the pins of port 3	0x0000 0000
PDAT3	0x0001_EEB8	R/W	The data register for port 3	Undefined
EXTINTR	0x0001_EEC0	R/W	External Interrupt control register	0x0000 0000

### Port 0 Control Registers (PCON0, PDAT0)

PCON0	Bit	Description	
P0.0	[1:0]	00 = Input, pull up 10 = INT0, pull up	01 = Output, pull up 11 = Input, pull up
P0.1	[3:2]	00 = Input, pull up 10 = INT1, pull up	01 = Output, pull up 11 = Input, pull up
P0.2	[5:4]	00 = Input, pull up 10 = INT2, pull up	01 = Output, pull up 11 = Input, pull up
P0.3	[7:6]	00 = Input, pull up 10 = INT3, pull up	01 = Output, pull up 11 = Input, pull up
P0.4	[9:8]	00 = Input, pull up 10 = Input	01 = Output 11 = Input, pull up
P0.5	[11:10]	00 = Input, pull up 10 = Input	01 = Output 11 = Input, pull up
P0.6	[13:12]	00 = Input, pull up 10 = Input	01 = Output 11 = Input, pull up
P0.7	[15:14]	00 = Input, pull up 10 = Input	01 = Output 11 = Input, pull up

## Rock2 DATA SHEET V1.1

---

<b>PDAT0</b>	<b>Bit</b>	<b>Description</b>
P0[7:0]	[7:0]	When the port is configured as output port, the pin state is the same as the corresponding bit When the port is configured as functional pin, the undefined value will be read.

### Port 1 Control Registers (PCON1, PDAT1)

<b>PCON1</b>	<b>Bit</b>	<b>Description</b>	
P1.0	[1:0]	00 = output, NORCSN 10 = Input	01 = Output 11 = Input, pull up
P1.1	[3:2]	00 = Input, pull up 10 = Input	01 = Output 11 = I2C_SDA
P1.2	[5:4]	00 = Input, pull up 10 = Input	01 = Output 11 = I2C_SCL
P1.3	[7:6]	00 = Input, pull up 10 = Input	01 = Output 11 = I2SMCLK
P1.4	[9:8]	00 = Input, pull up 10 = I2D_DAC_LRCK output	01 = Output 11 = I2D_DAC_LRCK input
P1.5	[11:10]	00 = Input, pull up 10 = I2D_ADC_LRCK output	01 = Output 11 = I2D_ADC_LRCK input
P1.6	[13:12]	00 = Input, pull up 10 = Input	01 = Output 11 = I2D_SDI
P1.7	[15:14]	00 = Input, pull up 10 = Input	01 = Output 11 = I2D_SDO

<b>PDAT1</b>	<b>Bit</b>	<b>Description</b>
P1[7:0]	[7:0]	When the port is configured as output port, the pin state is the same as the corresponding bit. When the port is configured as functional pin, the undefined value will be read.

## Rock2 DATA SHEET V1.1

---

### Port 2 Control Registers (PCON2, PDAT2)

<b>PCON2A</b>	<b>Bit</b>	<b>Description</b>	
P2.0	[1:0]	00 = SDDO 10 = Input	01 = Output 11 = Input
P2.1	[3:2]	00 = SDDI 10 = Input	01 = Output 11 = Input
P2.2	[5:4]	00 = SDCLK 10 = Input	01 = Output 11 = Input
P2.3	[7:6]	00 = SDCS 10 = Input	01 = Output 11 = Flash FCE1
P2.4	[9:8]	00 = A0 10 = Input	01 = Output 11 = Input
P2.5	[11:10]	00 = A1 10 = Input	01 = Output 11 = Input
P2.6	[13:12]	00 = A2 10 = Input	01 = Output 11 = Input
P2.7	[15:14]	00 = A3 10 = Input	01 = Output 11 = Input

<b>PCON2B</b>	<b>Bit</b>	<b>Description</b>	
P2.8	[1:0]	00 = WEN 10 = Input	01 = Output 11 = Input
P2.9	[3:2]	00 = CASN 10 = Input	01 = Output 11 = Input
P2.10	[5:4]	00 = RASN 10 = Input	01 = Output 11 = Input
P2.11	[7:6]	00 = CSN 10 = Input	01 = Output 11 = Input
P2.12	[9:8]	00 = input 10 = Input	01 = Output 11 = Input
P2.13	[11:10]	00 = Input 10 = Input	01 = Output 11 = PWM
P2.14	[13:12]	00 = Input 10 = Input	01 = Output 11 = PWM
P2.15	[15:14]	00 = Input 10 = Input	01 = Output 11 = PWM

## Rock2 DATA SHEET V1.1

---

<b>PDAT2</b>	<b>Bit</b>	<b>Description</b>
P2[15:0]	[15:0]	When the port is configured as output port, the pin state is the same as the corresponding bit. When the port is configured as functional pin, the undefined value will be read.

### Port 3 Control Registers (PCON3, PDAT3)

<b>PCON3</b>	<b>Bit</b>	<b>Description</b>	
P3.0	[1:0]	00 = D8, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.1	[3:2]	00 = D9, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.2	[5:4]	00 = D10, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.3	[7:6]	00 = D11, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.4	[9:8]	00 = D12, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.5	[11:10]	00 = D13, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.6	[13:12]	00 = D14, pull up 10 = Input	01 = Output 11 = Input, pull up
P3.7	[15:14]	00 = D15, pull up 10 = Input	01 = Output 11 = Input, pull up

<b>PDAT3</b>	<b>Bit</b>	<b>Description</b>
P3[7:0]	[7:0]	When the port is configured as output port, the pin state is the same as the corresponding bit. When the port is configured as functional pin, the undefined value will be read.

## Rock2 DATA SHEET V1.1

### EXTINTR (EXTERNAL INTERRUPT POLARITY CONTROL REGISTER)

The EXTINTR register selects the trigger types among various level or edge trigger mode of the external interrupt.

EXTINTR	Bit	Description
P0.0/INT0	[1:0]	Trigger mode of the EXTINT0. 00= Falling edge triggered 10= Falling or Rising edge 01 = Rising edge triggered 11= none
P0.1/INT1	[3:2]	Trigger mode of the EXTINT1. 00= Falling edge triggered 10= Falling or Rising edge 01 = Rising edge triggered 11= none
P0.2/INT2	[5:4]	Trigger mode of the EXTINT2. 00= Falling edge triggered 10= Falling or Rising edge 01 = Rising edge triggered 11= none
P0.3/INT3	[7:6]	Trigger mode of the EXTINT3. 00= Falling edge triggered 10= Falling or Rising edge 01 = Rising edge triggered 11= none

### NOTES:

If users want to change the trigger mode in the external interrupt mode, users are first required to switch the corresponding pin to input mode and then change the trigger mode.

## 3.14 SD/MMC Controller

The Secure Digital Card Interface (SDCI) can interface for SD memory card and Multi Media Card (MMC).

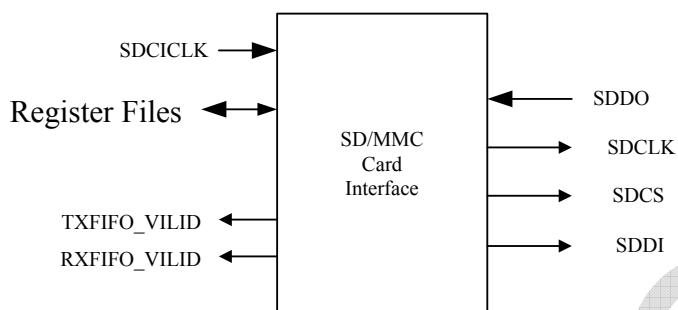


Figure 1. Overview

### FEATURES

- Supports Multi Media Card Specification Version 3.1
- Supports SD Memory Card Specification Version 1.0
- Cards Clock Rate up to PCLK
- 8 words (32 bytes) FIFO (depth 8) for data Transmit
- 8 words (32 bytes) FIFO (depth 8) for data Receive
- CRC7 & CRC16 Generator/Checker
- Normal Data Transfer Mode
- Support for Block and Multi-block Data Read and Write
- SPI Mode support

### Clock Requirement:

$$\text{SDCICLK} = \text{PCLK}$$

**!!!Note!!!: The DMA function has disable, only DSP read/write. Don't use DMA function.**

### 3.14.1 Block diagram

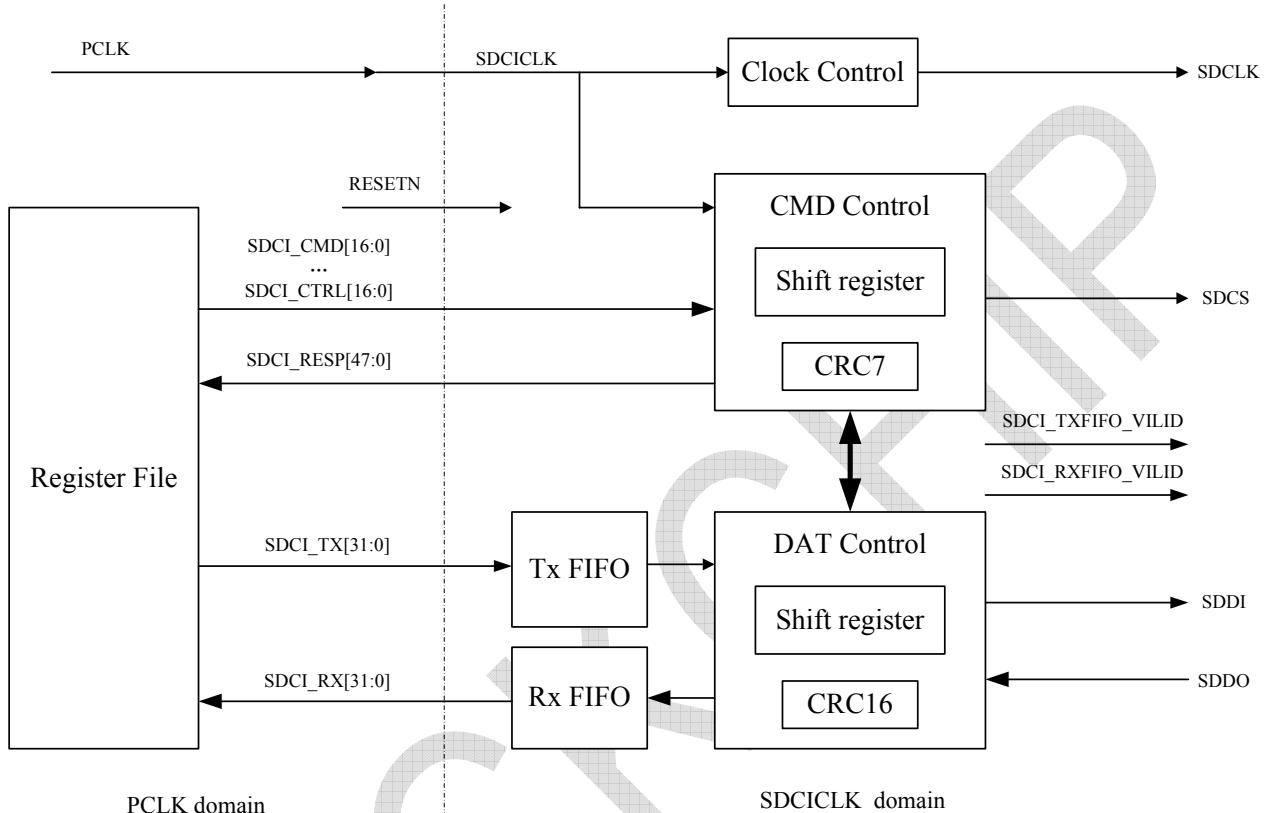


Figure 2. Block Diagram

### 3.14.2 Block I/Os

#### ADC CONTROLLER I/O DESCRIPTION

Register file interface		
SDCI_TX[31:0]	I	SD/MMC Interface data transmit to external SD/MMC card
SDCI_RX[31:0]	O	SDCI data receive from external SD/MMC card
SDCI_CTRL[15:0]	I	SDCI control signal from DSP
SDCI_DCTRL[15:0]	I	SDCI data control signal from DSP
SDCI_CMD[15:0]	I	SDCI command(CMD) signal from DSP
SDCI_ARG[31:0]	I	SDCI command(CMD) argument signal from DSP
SDCI_DLEN[16:0]	I	SDCI data block length signal from DSP
SDCI_TOKEN [15:0]	I	SDCI data block transmit TOKEN signal from DSP
SDCI_RESP0[31:0]	O	Response Data0 from external SD/MMC card
SDCI_RESP0[7:0]	O	Response Data1 from external SD/MMC card

## Rock2 DATA SHEET V1.1

SDCI_STA[15:0]	O	ADC1 transmit/receive status to DSP	
SDCI_STAC[15:0]	I	Clear transmit/receive status from DSP	
<b>Register file interface I/Os total:</b>			
Application interface			
SDCICLK	I	SD card Interface module clock input	
SDCS	O	External SD/MMC card select	
SDDI	O	Data output to External SD/MMC card	
SDCLK	O	Clock output to External SD/MMC card	
SDDO	I	Data input from External SD/MMC card	
SDDO_EN	O	Data output enable	
SD_TXFIFO_VILID	O	SD/MMC Card Interface Transmit FIFO not Full, to DMA	
SD_RXFIFO_VILID	O	SD/MMC Card Interface Receive FIFO not Empty, to DMA	

### SDCI OPERATION

In Rock2, SDCI select SPI as transmit and receive function. A serial clock line (SDCLK) is synchronized with the data output(SDDI) and the data input(SDDO).

*In which, SDDO/SDDI are directly connect to External SD card of SDDO/SDDI.*

### SDCI Configuration

After a hardware reset, the SDCI pins are selected as SPI mode.

### CMD Path Programming

1. (soft) Write the argument value to SDCI\_ARG register.
2. (soft) Write the command information to SDCI\_CMD register.  
– Confirm the ready of command transmission flag of SDCI\_STA[0].
3. (soft) Write the command start bit (CMDSTR) to SDCI\_CMD register.

(SDCI hard) When SDCI detected CMDSTR == 1, begin transmitting the CMD, and then receive the response data from external SD card. During this progress, the SDCI will response the progress to SDCI\_STA[4:1], which can be read by DSP and decide the next progress.

4. (soft) Check the status of SDCI command operation by DSP.  
– Command transmission is in progress, the SDCI\_STA[1] is set.  
– Command transmission is completed, the SDCI\_STA[2] is set.  
– Response reception is in progress, the SDCI\_STA[3] is set.  
– Response reception is completed, the SDCI\_STA[4] is set.
5. (soft) Check the Status of response if there is time-out error occur
6. (soft) Check the Card response, read from SDCI\_RESP1~0 register.
7. (soft) Clear the corresponding flag of the SDCI\_STA register by write the SDCI\_STAC register.

## Rock2 DATA SHEET V1.1

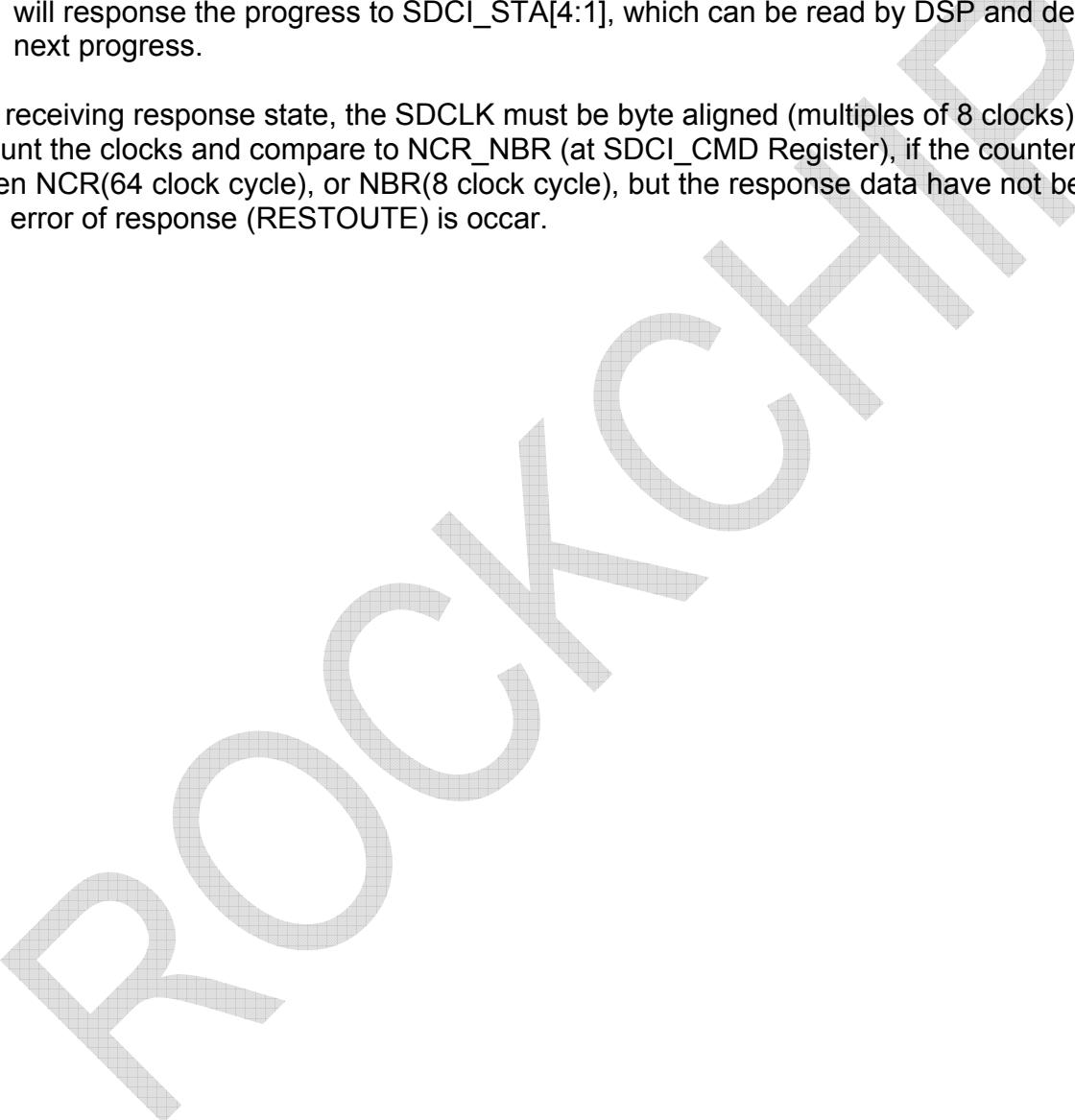
---

**Hard core progress:** In step 3.

When SDCI detected CMDSTR == 1

- 1) Reset CRC7 to 000,0000b
- 2) Transmit the 6bit CMD in SDCI\_CMD, 32bit command argument in SDCI\_ARG
- 3) Transmit the 7 bits of CRC7 and one clock of end(1).
- 4) Start receiving the response data from external SD card. During this progress, the SDCI will response the progress to SDCI\_STA[4:1], which can be read by DSP and decide the next progress.

At receiving response state, the SDCLK must be byte aligned (multiples of 8 clocks). It must count the clocks and compare to NCR\_NBR (at SDCI\_CMD Register), if the counter is bigger then NCR(64 clock cycle), or NBR(8 clock cycle), but the response data have not been receive, an error of response (RESTROUTE) is occur.



# Rock2 DATA SHEET V1.1

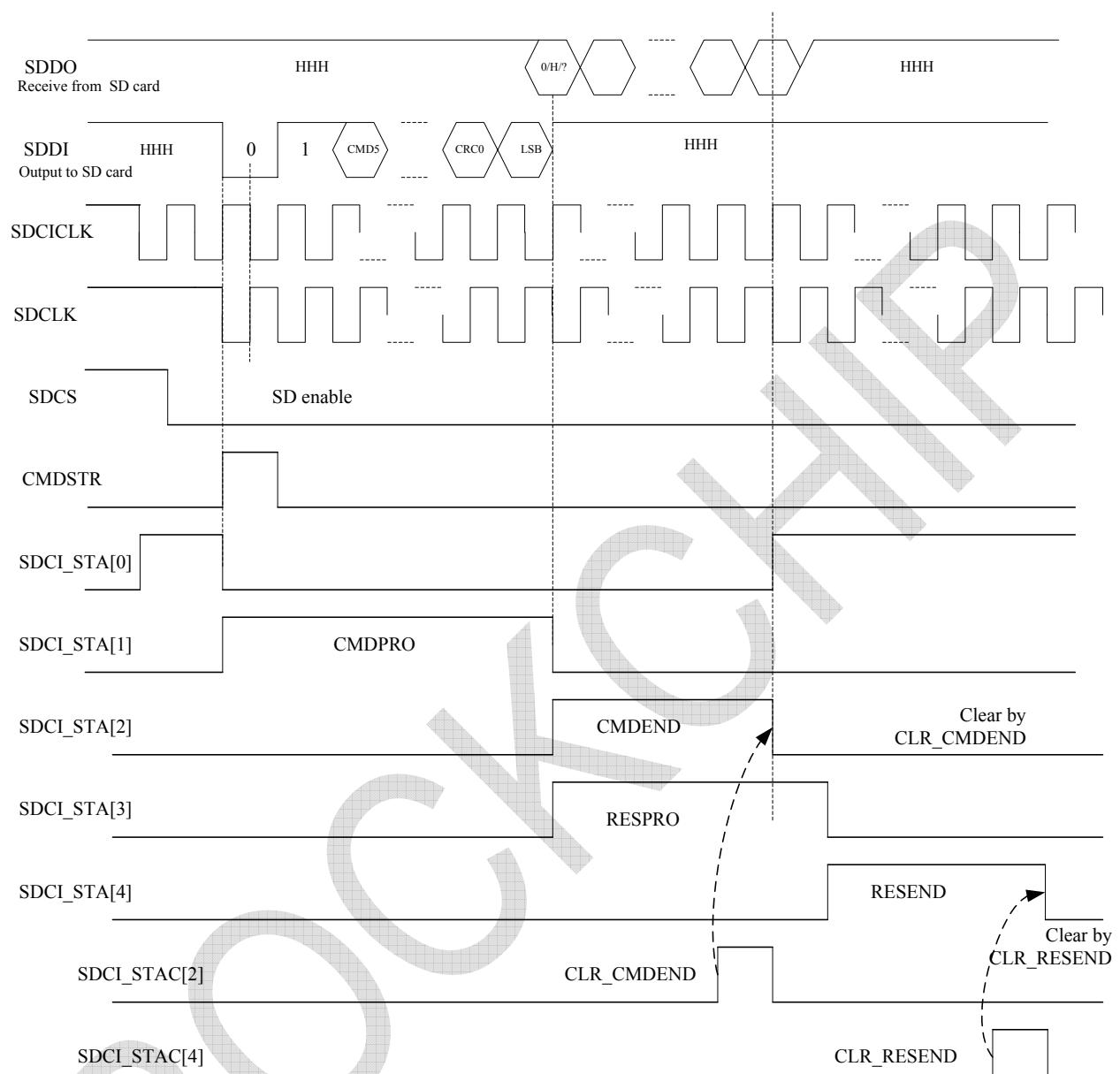


Fig3. Write CMD Timing

## DAT Path Programming (not use DMA)

-Operation of addressed data transfer commands (adtc)

1. (soft) Write the Data block length to the SDCI\_DCTRL register.
2. (soft) FIFO reset by writing the SDCI\_DCTRL register.
3. (soft and Hard core progress) Do **CMD path Programming**

## Rock2 DATA SHEET V1.1

4. (soft) Write the Data transmission start bit DATSTR to SDCI\_DCTRL[2] register(only data write operation).
5. (soft) When Write Operation, Write Tx data to SDCI\_TX register while TxFIFO is available by checking TXFIFO\_NFULL in SDCI\_STA[13:12].
6. (soft) When Read Operation, Read Rx data to SDCI\_RX register while RxFIFO is available by checking RXFIFO\_NEMPTY in SDCI\_STA[11:10].
7. (soft) Check the status of SDCI data operation.
  - Data transmission/reception is in progress, the SDCI\_STA[5] is set.
  - Data block transmission/reception is completed, the SDCI\_STA[6] is set.
  - Data CRC data transmission/reception is completed, the SDCI\_STA[7] is set.
  - CRC status token reception is completed, the SDCI\_STA[8] is set, only write command.
  - Card busy state, the SDCI\_STA[9] is set.
8. (soft) Check the Status of data transfer
9. (soft) Clear the corresponding flag of the SDCI\_STA register by write the SDCI\_STAC register.
10. If Multiple data block transfer, repeat 4~9.

**Hard core progress:** In step 4.

- In transmit progress, when SDCI detected DATSTR == 1
- 1) Begin transmitting the TOKEN, when the TOKEN have been transmit ok, SDCI resets the CRC16 to 0x0000.
  - 2) Start transmit the DATA BLOCK (Length of SDCI\_DLEN).
  - 3) Transmit the 2 bytes of CRC16.
  - 4) Start reading response data from external SD card. which can be read by DSP and decide the next progress.

During this progress, the SDCI will response the state to SDCI\_STA[8:5], which can be read by DSP and decide the next progress.

In receive progress, When SDCI detected DATSTR == 1

- 1) Begin reading the DATA BLOCK, when a block (Length of SDCI\_DLEN) has been receive ok, save the data of CRC16 to the SDCI\_CRC16 register.
- 2) Receive the 16bit CRC16 data.
- 3) Compare the 16bit CRC16 data to SDCI\_CRC16 register, if they equal, clear the CRC16E in SDCI\_STA register.

During this progress, the SDCI will response the state to SDCI\_STA[8:5], which can be read by DSP and decide the next progress.

# Rock2 DATA SHEET V1.1

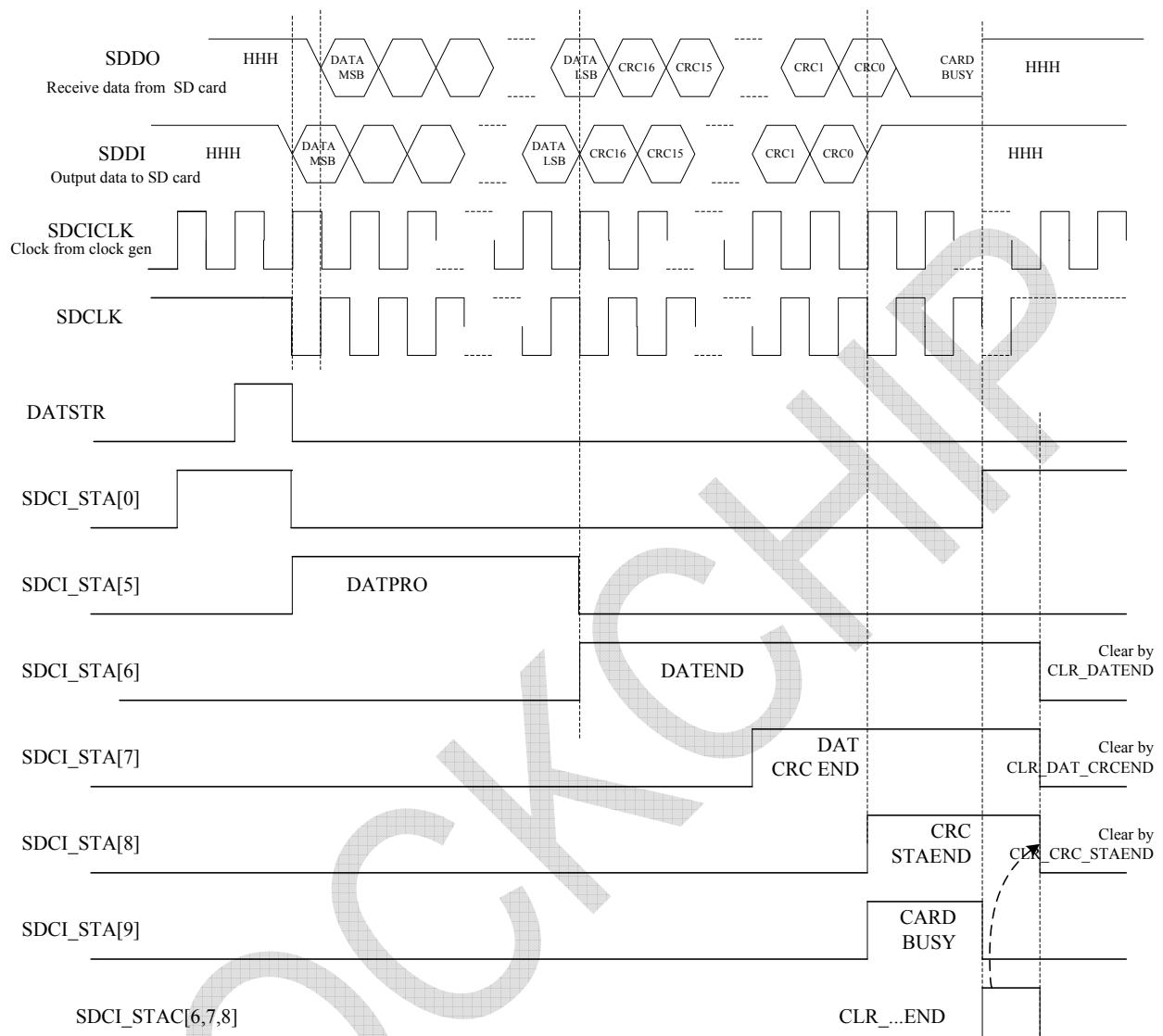


Fig4. Data transmit/receive Timing

## FIFO model: (Using DSP)

When transmit, DSP first check signal TX\_FIFO\_NFULL, if it is High, then transmit data to TxFIFO0-7. If the TxFIFOs are full, it pulls TX\_FIFO\_NFULL to Low, DSP waiting a High of TX\_FIFO\_NFULL for next transmission. As the shift register reads a data from TxFIFOs, TX\_FIFO\_NFULL comes High.

When receive, the Shift Register puts data to the RxFIFOs, DSP first check signal RX\_FIFO\_NEMPTY, if it is High, then reads data from RxFIFOs. If it reads all the data, SDCI will reset RX\_FIFO\_NEMPTY to Low. When the RxFIFOs are full, the RX\_FIFO\_FULL will be high, SDCI stops receiving data from SD card (Hold down the SDCLK).

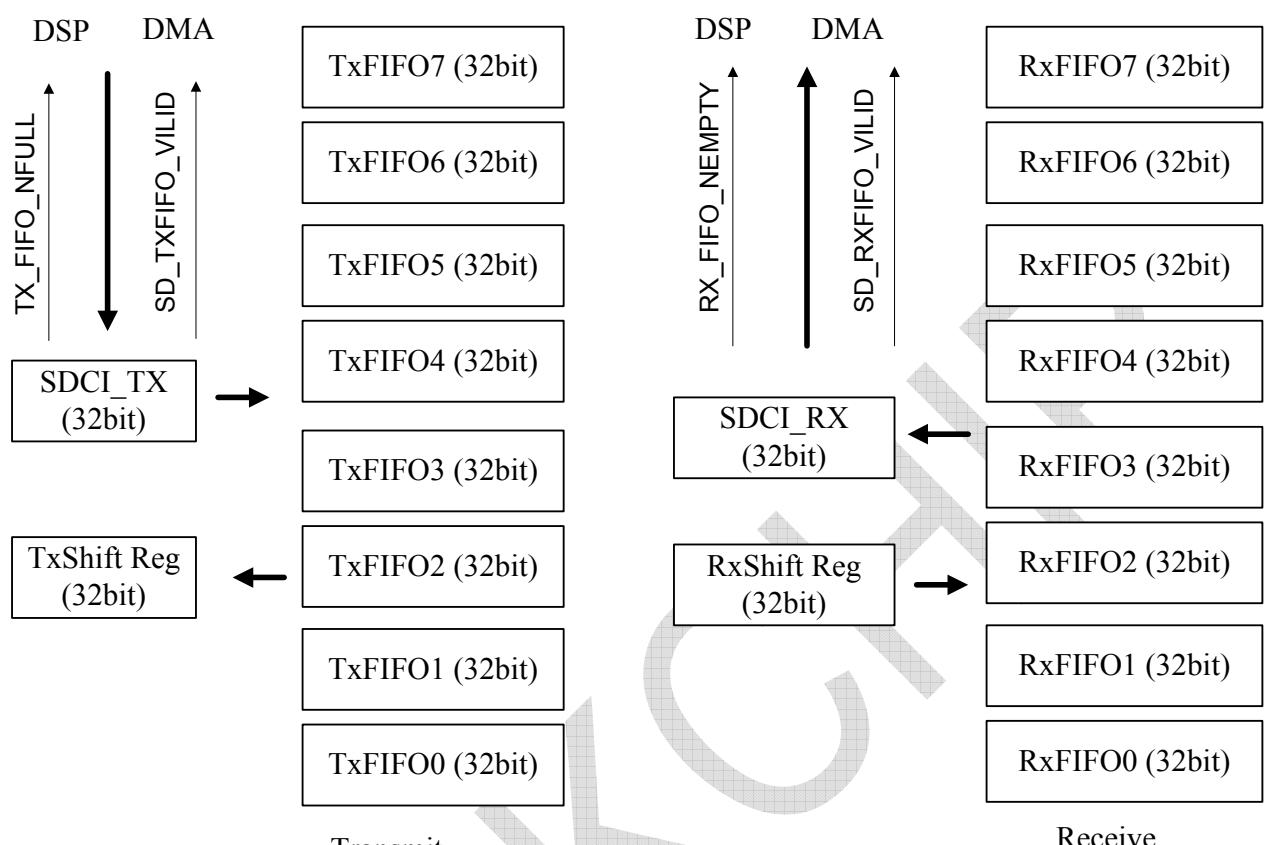


Fig 5. FIFO Block

The write data block length is only 512 bytes, which value is in the register **SDCI\_DLEN**.

### 3.14.3 Register Descriptions

#### SDCI REGISTER

Base Address = 0x0001\_EF00

Register Name	Width	Offset	Read/Write	Register Name	Initial Value
SDCI_CTRL	16bit	0x00	R/W	Control Register	0x0000
SDCI_DCTRL	16bit	0x04	R/W	Data Control Register	0x0000
SDCI_DLEN	16bit	0x08	R/W	Data Control Register	0x0000
SDCI_TOKEN	16bit	0x0C	R/W	Data Block Token	0x0000
SDCI_CMD	16bit	0x10	R/W	Command Register	0x0000
SDCI_ARG	32bit	0x14	R/W	Argument Register	0x0000_0000
SDCI_STAC	16bit	0x18	W	Status Clear Register	0x0000
SDCI_STA	16bit	0x1C	R	Status Register	0x3000

## Rock2 DATA SHEET V1.1

---

SDCI_RESP0	32bit	0x20	R	Response Register0	0x0000_0000
SDCI_RESP1	8bit	0x24	R	Response Register1	0x00
SDCI_TX	32bit	0x28	R/W	Data Transmit Register	0x0000_0000
SDCI_RX	32bit	0x2C	R/W	Data Receive Register	0x0000_0000
SDCI_CRC16	16bit	0x30	R/W	CRC16 buffer Register	0x0000_0000

**SD/MMC CLOCK CONTROL REGISTER (SDCLKCON):**  
**(This Register is in the CLOCK module)**

SDCLKCON	Bit	Description	Initial State
Reserved	[15:8]	Reserved	-
SDCICLK_DIV	[7:0]	8-bit prescaler value (The input clock is PCLK) $SDCICLK = PCLK / (SDCLK\_DIV + 1)$	0000,0000b

**SD/MMC Control Register (SDCI\_CTRL)**

Name	Bit	Description
Reserved	15:5	
SD_CSEN	4	SD card select 0 = Disable SDSCS, Pull the SDSCS pin High 1 = Enable SDSCS, Reset the SDSCS pin low
DMA_REQ_CON	3	Select DMA request condition, only Read Command 0 = DMA request when Receive FIFO is not empty. 1 = DMA request when Receive FIFO is full.
DMAEN	2	DMA enable 0 = DMA disable 1 = DMA enable
CARD_TYPE	1	Select the card type 0 = SD Card 1 = MM Card
SDCIEN	0	SDCI block enable 0 = Disable the SDCI 1 = Enable the SDCI

## Rock2 DATA SHEET V1.1

### SD/MMC Data Control (SDCI\_DCTRL) Register

Name	Bit	Description
Reserved	15:3	
DATSTR	2	Start the data transfer, automatically cleared. 0 = No effect 1 = Data Transmission Start
RXFIFORST	1	Reset the receive FIFO. 0 = No effect 1 = Reset the Receive FIFO
TXFIFORST	0	Reset the transmit FIFO. 0 = No effect 1 = Reset the Transmit FIFO

### SD/MMC Data Length (SDCI\_DLEN) Register

Name	Bit	Description
BLK_LEN	15:0	Determine the data block size (unit : byte)

### SD/MMC Data Token (SDCI\_TOKEN) Register

Name	Bit	Description
TOKEN	7:0	Data block Token

### SDCI Command (SDCI\_CMD) Register

Name	Bit	Description
TAAC	15:12	TAAC response clock cycle , $65536^*(TAAC+1)$
CMDSTR	11	Start the command transfer, automatically cleared. Only writable when SDCI_SR[0] = 1. 0 = No effect 1 = Command transmit start

## Rock2 DATA SHEET V1.1

---

NCR_NBR	10	Select the clock cycle command to response. 0 = NCR(<=64 clock cycle) 1 = NBR(<=8 clock cycle)
RES_FORMAT	9:8	Select response format 00 = No Response 01 = R1 (8bit), R1b(8bit+busy), Data Resp (8bit) 10 = R2 (16bit) 11 = R3 (40bit,R1+OCR)
Reserved	7	
CMD_READ	6	Select the command type 0 = General command 1 = Read data command
CMD_NUM	5:0	Set the command number

### SDCI Argument (SDCI\_ARG) Register

Name	Bit	Description
ARGUMENT	31:0	Set the argument value

### SDCI STATUS Clear (SDCI\_STAC) Register

Name	Bit	Description
CLR_DATCRC16E	15	Clear DATCRC16E at SDCI_STA register 0 = No effect 1 = Clear the RESENDE bit
CLR_RESTROUTE	14	Clear RESTROUTE at SDCR_STA register 0 = No effect 1 = Clear the RESTROUTE bit
Reserved	13:9	
CLR_CRC_STAEND	8	Clear CRC_STAEND at SDCI_STA register 0 = No effect 1 = Clear the CRC_STAEND bit
CLR_DAT_CRCEND	7	Clear DAT_CRCEND at SDCI_STA register 0 = No effect 1 = Clear the DAT_CRCEND bit

## Rock2 DATA SHEET V1.1

---

CLR_DATEND	6	Clear DAT_END at SDCL_STA register 0 = No effect 1 = Clear the DATEND bit
Reserved	5	
CLR_RESEND	4	Clear RESEND at SDCL_STA register 0 = No effect 1 = Clear the RESEND bit
Reserved	3	
CLR_CMDEND	2	Clear CMDEND at SDCL_STA register 0 = No effect 1 = Clear the CMDEND bit
Reserved	1:0	

### **SDCI STATUS (SDCI\_STA) Register**

Clear Condition:

A : According to the SDCI interface state.

C : Clear by write '1' to corresponding bit of SDCI\_STAC.

Name	Bit	Description	Clear Condition
DATCRC16E	15	Receive data CRC16 error 0 = No CRC16 Error occurs 1 = CRC16 Error occurs	C
RESTROUTE	14	Response timeout error (Ncr, Nbr) 0 = No Command to Response timeout error occurs 1 = Command to Response timeout error occurs	C
TX_FIFO_NFULL	13	Tx FIFO full 0 = Transmit FIFO is full 1 = Transmit FIFO is not full	A
TX_FIFO_EMPTY	12	Tx FIFO empty 0 = Transmit FIFO is not empty 1 = Transmit FIFO is empty	A
RX_FIFO_FULL	11	Rx FIFO full 0 = Receive FIFO is not full 1 = Receive FIFO is full	A
RX_FIFO_NEMPTY	10	Rx FIFO empty 0 = Transmit FIFO is empty 1 = Transmit FIFO is not empty	A

## Rock2 DATA SHEET V1.1

DAT_BUSY	9	Data line busy 0 = Card is not busy 1 = Card is busy This status is direct connected inverted SDDO of Card.	A
CRC_STAEND	8	Write data CRC status token receive end 0 = CRC status token reception is not ended 1 = CRC status token reception is ended	C
DAT_CRCEND	7	Data CRC transmit/receive end 0 = CRC transmission/reception is not ended 1 = CRC transmission/reception is ended	C
DATEND	6	Data transmit/receive end 0 = Data transmission/reception is not ended 1 = Data transmission/reception is ended	C
DATPRO	5	Data transfer in progress 0 = Data transmission/reception is not in progress 1 = Data transmission/reception is in progress	A
RESEND	4	Response receive end 0 = Response reception is not ended 1 = Response reception is ended	C
RESPRO	3	Response receive in progress 0 = Response reception is not in progress 1 = Response reception is in progress	A
CMDEND	2	Command transfer end 0 = Command transmission is not ended 1 = Command transmission is ended	C
CMDPRO	1	Command transfer in progress 0 = Command transmission is not in progress 1 = Command transmission is in progress	A
CMDRDY	0	Command ready 0 = Command transfer is not ready 1 = Command transfer is ready (SDCI_CMD, SDCI_ARG set complete)	A

### SDCI Response0 (SDCI\_RESP0) Register

Name	Bit	Description
------	-----	-------------

## Rock2 DATA SHEET V1.1

---

RESPONSE0	31:0	The RES_FORMAT[2:0] determines the value R1, R1b: response [7:0] , [31:8] == 0000,00 R2: response [15:00] , [31:16] == 0000 R3: response [31:00]
-----------	------	---

### SDCI Response1 (SDCI\_RESP1) Register

Name	Bit	Description
RESPONSE1	7:0	The RES_FORMAT[2:0] determines the value R1, R1b: 00 R2: 00 R3: response [7:0] == R1

### SDCI Data (SDCI\_TX) Register

Name	Bit	Description
SDCI_TX	31:0	Data buffer for transmit

**NOTE:** If data is not aligned to word (4byte), SDCI\_TX register value is following table(little\_endian).

Write Data	SDCI_TX[31:24]	SDCI_TX[13:16]	SDCI_TX[15:8]	SDCI_TX[7:0]
4n+1 byte	Write data[7:0]	stuff bits	stuff bits	stuff bits
4n+2 byte	Write data[15:8]	Write data[7:0]	stuff bits	stuff bits
4n+3 byte	Write data[23:16]	Write data[15:8]	Write data[7:0]	stuff bits

### SDCI Data (SDCI\_RX) Register

Name	Bit	Description
SDCI_RX	31:0	Data buffer for receive

## Rock2 DATA SHEET V1.1

**NOTE:** If data is not aligned to word(4byte), SDCI\_RX register value is following table  
(Little\_endian)

Read Data	SDCI_RX[31:24]	SDCI_RX[13:16]	SDCI_RX[15:8]	SDCI_RX[7:0]
4n+1 byte	0x00	0x00	0x00	Read data[7:0]
4n+2 byte	0x00	0x00	Read data[15:8]	Read data[7:0]
4n+3 byte	0x00	Read data[23:16]	Read data[15:8]	Read data[7:0]

### CRC16 Data (SDCI\_CRC16) Register

Name	Bit	Description
SDCI_CRC16	15:0	The value of CRC16 by the receive data

## 4 Programming Information

### 4.1 Address Maps (Memory spacing)

DSP internal memory space			
DMA_MASTER1			Low
DSP_MASTER2			High
DSP_IRAM	0x0000_0000--0x0000_EFFF	60KW	
DSP_BOOT	0x0000_F000--0x0000_FFFF	4KW	
DSP_DRAM	0x0000_0000--0x0000_EFFF	60KW	
System memory space			
Module	Address		
USB	0x0001_E000--0x0001_E3FF	1KB	4
APB	(0x0001_E400--0x0001_EFFF)	3KB	3
WDT	0x0001_E4000--0x0001_E7FF	1KB	
I2C	0x0001_E800--0x0001_EBFF	1KB	
Register file(APB)	(0x0001_EC00--0x0001_EFFF)	1KB	
I2S	0x0001_EC00--0x0001_EC7F	128B	
PWM	0x0001_EC80--0x0001_ECFF	128B	
4-1.ADC	0x0001_ED00--0x0001_ED7F	128B	
DC/DC	0x0001_ED80--0x0001_EDFF	128B	
CLOCK/PLL	0x0001_EE00--0x0001_EE7F	128B	
GPIO	0x0001_EE80--0x0001_EEFF	128B	
SD/MMC	0x0001_EF00--0x0001_EF7F	128B	
EXTERNAL	(0x0001_F000--0x0001_FBFF)	3KB	
	(0x0000_0000--0x0001_DFFF)	120KB	
	(0x0020_0000--0x003F_FFFF)	2MB	1
Register file(AHB)	0x0001_F800--0x0001_F8FF	256B	
VIDEO	0x0001_F900--0x0001_F9FF	256B	
	0x0001_0000--0x0001_DFFF	32KB	
Flash_CS0	0x0001_FA00--0x0001_FAFF	256B	
	0x0000_0000--0x0000_7FFF	32KB	
Flash_CS1	0x0001_FB00--0x0001_FBFF	256B	
	0x0000_8000--0x0000_FFFF	32KB	
EPROM	0x0001_F000--0x0001_F7FF	2KB/BOOTROM	
	0x0020_0000--0x003F_FFFF	2MB	
DMA	0x0001_FC00--0x0001_FFFF	1KB	2
Memory controller	0x0002_0000--0x0005_FFFF	256KB	5
IMEM	0x0002_0000--0x0003_FFFF	128KB/120KB	
DMEM	0x0004_0000--0x0005_FFFF	128KB/120KB	
SDRAM controller	0x0006_0000--0x0006_FFFF	64KB	6
SDRAM	0x0400_0000--0x07FF_FFFF	64M	7

## 4.2 Interrupt vector

interrupt	优先级	address	
P00_INT	0	16	
P01_INT	1	15	
P02_INT	2	14	
P03_INT	3	13	
PWM	4	12	
TIMER0	5	11	
TIMER1	6	10	
DMAC	7	9	
WDT	8	8	
-	9	7	
USB	10	6	
-	11	5	
-	12	4	
-	13	3	
-DEI	14	2	
NMI	15	1	
RESET	16	0	

## 4.3 Registers

```
#define mempcr      0xf807
#define gpoport     0xf806
#define DSP_STACK    0xbfff
#define LCD_DATA     (0x1f900 )/2
#define LCD_REG      (0x1f900+0x04)/2
#define FLASH0_DATA  (0x1fa00 )/2
#define FLASH0_ADDR  (0x1fa00+0x04)/2
#define FLASH0_CMD   (0x1fa00+0x08)/2
#define FLASH1_DATA  (0x1fb00 )/2
#define LCD_DATA1    0x10000
#define FLASH1_ADDR  (0x1fb00+0x04)/2
#define FLASH1_CMD   (0x1fb00+0x08)/2
#define EXT_REG_BASE 0x1f800
#define ECC0          (EXT_REG_BASE+0x00)/2
#define ECC1          (EXT_REG_BASE+0x04)/2
#define ECC2          (EXT_REG_BASE+0x08)/2
#define ECC3          (EXT_REG_BASE+0x0c)/2
#define ECCCTL        (EXT_REG_BASE+0x10)/2
#define FMCTL         (EXT_REG_BASE+0x14)/2
#define CFWAIT        (EXT_REG_BASE+0x18)/2
#define FMWAIT        (EXT_REG_BASE+0x1c)/2
```

## Rock2 DATA SHEET V1.1

---

```

#define LCDWAIT      (EXT_REG_BASE+0x20)/2
#define SYSCTL       (EXT_REG_BASE+0x24)/2
#define SDRAM_BASE   0x4000000/2
#define DSPMEM_BASE  0x20000/2
#define USB_BASE     0x1e000/2
#define EXTMEM_BASE  0x00000/2
#define EXTMEM_BASE2 0x80000/2
#define I2C_BASE     0x1e800/2
#define WDT_BASE     0x1e400/2

#define DW_APB_I2C_BASE 0x1e800
#define IC_CON        (DW_APB_I2C_BASE + 0x00)/2 // R/W [5:0] 0x04
#define IC_TAR        (DW_APB_I2C_BASE + 0x04)/2 // R/W [11:0] 0x55
#define IC_SAR        (DW_APB_I2C_BASE + 0x08)/2 // R/W [9:0] 0x55
#define IC_HS_MADDR   (DW_APB_I2C_BASE + 0x0c)/2
#define IC_DATA_CMD   (DW_APB_I2C_BASE + 0x10)/2 // R/W [8:0] 0x0
#define IC_SS_HCNT    (DW_APB_I2C_BASE + 0x14)/2 // R/W [15:0] 0x60
#define IC_SS_LCNT    (DW_APB_I2C_BASE + 0x18)/2 // R/W [15:0] 0x80
#define IC_FS_HCNT    (DW_APB_I2C_BASE + 0x1c)/2 // R/W [15:0] 0x10
#define IC_FS_LCNT    (DW_APB_I2C_BASE + 0x20)/2 // R/W [15:0] 0x22
#define IC_HS_HCNT    (DW_APB_I2C_BASE + 0x24)/2
#define IC_HS_LCNT    (DW_APB_I2C_BASE + 0x28)/2
#define IC_INTR_STAT  (DW_APB_I2C_BASE + 0x2c)/2
#define IC_INTR_MASK   (DW_APB_I2C_BASE + 0x30)/2
#define IC_RAW_INTR_STAT (DW_APB_I2C_BASE + 0x34)/2
#define IC_RX_TL       (DW_APB_I2C_BASE + 0x38)/2
#define IC_TX_TL       (DW_APB_I2C_BASE + 0x3c)/2
#define IC_CLR_INTR    (DW_APB_I2C_BASE + 0x40)/2
#define IC_CLR_RX_UNDER (DW_APB_I2C_BASE + 0x44)/2
#define IC_CLR_RX_OVER  (DW_APB_I2C_BASE + 0x48)/2
#define IC_CLR_TX_OVER  (DW_APB_I2C_BASE + 0x4c)/2
#define IC_CLR_RD_REQ   (DW_APB_I2C_BASE + 0x50)/2
#define IC_CLR_TX_ABRT  (DW_APB_I2C_BASE + 0x54)/2
#define IC_CLR_RX_DONE   (DW_APB_I2C_BASE + 0x58)/2
#define IC_CLR_ACTIVITY  (DW_APB_I2C_BASE + 0x5c)/2
#define IC_CLR_STOP_DET  (DW_APB_I2C_BASE + 0x60)/2
#define IC_CLR_START_DET (DW_APB_I2C_BASE + 0x64)/2
#define IC_CLR_GEN_CALL  (DW_APB_I2C_BASE + 0x68)/2
#define IC_ENABLE        (DW_APB_I2C_BASE + 0x6c)/2
#define IC_STATUS        (DW_APB_I2C_BASE + 0x70)/2 // R [4:0] 0x6
#define IC_TXFLR        (DW_APB_I2C_BASE + 0x74)/2
#define IC_RXFLR        (DW_APB_I2C_BASE + 0x78)/2
#define IC_SRESET        (DW_APB_I2C_BASE + 0x7c)/2
#define IC_TX_ABRT_SOURCE (DW_APB_I2C_BASE + 0x80)/2
#define IC_VERSION_ID    (DW_APB_I2C_BASE + 0xf8)/2
#define IC_DMA_CR        (DW_APB_I2C_BASE + 0x88)/2
#define IC_DMA_TDLR      (DW_APB_I2C_BASE + 0x8c)/2
#define IC_DMA_RDLR      (DW_APB_I2C_BASE + 0x90)/2
#define I2CPING_1BIT_WR  (IC_TAR)/2

#define DMACBASE        0x1fc00 // R/W Width Reset Value
#define DMAR_SAR0       (DMACBASE+0x000)/2 // R/W 32 0x0
#define DMAR_DAR0       (DMACBASE+0x008)/2 // R/W 32 0x0
#define DMAR_CTL0       (DMACBASE+0x018)/2 // R/W 64 0x00000002_00004825

```

---

## Rock2 DATA SHEET V1.1

---

```

#define DMAR_CFG0      (DMACBASE+0x040)/2 // R/W 64 0x00000004_00000c00
#define DMAR_SGR0      (DMACBASE+0x048)/2 // R/W 32 0x0
#define DMAR_DSR0      (DMACBASE+0x050)/2 // R/W 32 0x0
#define DMAR_SAR1      (DMACBASE+0x058)/2 // R/W 32 0x0
#define DMAR_DAR1      (DMACBASE+0x060)/2 // R/W 32 0x0
#define DMAR_CTL1      (DMACBASE+0x070)/2 // R/W 64 0x00000002_00004825
#define DMAR_CFG1      (DMACBASE+0x098)/2 // R/W 64 0x00000004_00000c20
#define DMAR_SGR1      (DMACBASE+0xa0)/2 // R/W 32 0x0
#define DMAR_DSR1      (DMACBASE+0xa8)/2 // R/W 32 0x0
#define DMAR_SAR2      (DMACBASE+0xb0)/2 // R/W 32 0x0
#define DMAR_DAR2      (DMACBASE+0xb8)/2 // R/W 32 0x0
#define DMAR_CTL2      (DMACBASE+0xc8)/2 // R/W 64 0x00000002_00004825
#define DMAR_CFG2      (DMACBASE+0xf0)/2 // R/W 64 0x00000004_00000c40
#define DMAR_SGR2      (DMACBASE+0xf8)/2 // R/W 32 0x0
#define DMAR_DSR2      (DMACBASE+0x100)/2 // R/W 32 0x0
#define DMAR_RawBlock   (DMACBASE+0x2c8)/2 // R 3 0x0
#define DMAR_RawSrcTran (DMACBASE+0x2d0)/2 // R 3 0x0
#define DMAR_RawDstTran (DMACBASE+0x2d8)/2 // R 3 0x0
#define DMAR_RawErr     (DMACBASE+0xe0)/2 // R 3 0x0
#define DMAR_RawTfr     (DMACBASE+0xc0)/2 // R 3 0x0
#define DMAR_StatusTfr  (DMACBASE+0xe8)/2 // R 3 0x0
#define DMAR_StatusBlock (DMACBASE+0x2f0)/2 // R 3 0x0
#define DMAR_StatusBlock (DMACBASE+0x2f0)/2 // R 3 0x0
#define DMAR_StatusSrcTran (DMACBASE+0x2f8)/2 // R 3 0x0
#define DMAR_StatusDstTran (DMACBASE+0x300)/2 // R 3 0x0
#define DMAR_StatusErr  (DMACBASE+0x308)/2 // R 3 0x0
#define DMAR_MaskTfr    (DMACBASE+0x310)/2 // R/W 3-3 0x0
#define DMAR_MaskBlock   (DMACBASE+0x318)/2 // R/W 3-3 0x0
#define DMAR_MaskSrcTran (DMACBASE+0x320)/2 // R/W 3-3 0x0
#define DMAR_MaskDstTran (DMACBASE+0x328)/2 // R/W 3-3 0x0
#define DMAR_MaskErr    (DMACBASE+0x330)/2 // R/W 3-3 0x0
#define DMAR_ClearTfr   (DMACBASE+0x338)/2 // W 3 0x0
#define DMAR_ClearBlock  (DMACBASE+0x340)/2 // W 3 0x0
#define DMAR_ClearSrcTran (DMACBASE+0x348)/2 // W 3 0x0
#define DMAR_ClearDstTran (DMACBASE+0x350)/2 // W 3 0x0
#define DMAR_ClearErr   (DMACBASE+0x358)/2 // W 3 0x0
#define DMAR_StatusInt  (DMACBASE+0x360)/2 // W 5 0x0
#define DMAR_ReqSrcReg  (DMACBASE+0x368)/2 // R/W 3-3 0x0
#define DMAR_ReqDstReg  (DMACBASE+0x370)/2 // R/W 3-3 0x0
#define DMAR_SglReqSrcReg (DMACBASE+0x378)/2 // R/W 3-3 0x0
#define DMAR_SglReqDstReg (DMACBASE+0x380)/2 // R/W 3-3 0x0
#define DMAR_LstSrcReg  (DMACBASE+0x388)/2 // R/W 3-3 0x0
#define DMAR_LstDstReg  (DMACBASE+0x390)/2 // R/W 3-3 0x0
#define DMAR_DmaCfgReg  (DMACBASE+0x398)/2 // R/W 1 0x0
#define DMAR_ChEnReg    (DMACBASE+0x3a0)/2 // R/W 3-3 0x0
#define DMAR_DmaldReg   (DMACBASE+0x3a8)/2 // R 32
#define DMAR_DmaTestReg (DMACBASE+0x3b0)/2 // R/W 1 0x0
#define DMAR_DMACID     (DMACBASE+0x3f8)/2 // R 64 0x044_57_11_10

#define DW_APB_WDT_BASE 0x1e400
#define WDT_CR        (DW_APB_WDT_BASE + 0x00)/2 // R/W [4:0] 0x0
#define WDT_TORR       (DW_APB_WDT_BASE + 0x04)/2 // R/W [3:0] 0x7
#define WDT_CCVR       (DW_APB_WDT_BASE + 0x08)/2 // R [31:0] 0x7fffff
#define WDT_CRR        (DW_APB_WDT_BASE + 0x0C)/2 // R/W [7:0] 0x0 //0x76
#define WDT_STAT       (DW_APB_WDT_BASE + 0x10)/2 // R [0] 0x0

```

---

## Rock2 DATA SHEET V1.1

---

```

#define WDT_EOI      (DW_APB_WDT_BASE + 0x14)/2 // R [0] 0x0
#define WDTPING_1BIT_WR (WDT_CR)/2

#define DW_MEMCTL_BASE      0x0000
#define MEMCTL_SCONR      (DW_MEMCTL_BASE + 0x00)/2 // R/W [31:0] 0x1c1168
#define MEMCTL_STMG0R      (DW_MEMCTL_BASE + 0x04)/2 // R/W [31:0] 0x19d9451
#define MEMCTL_STMG1R      (DW_MEMCTL_BASE + 0x08)/2 // R/W [31:0] 0x700008
#define MEMCTL_SCTLR      (DW_MEMCTL_BASE + 0x0c)/2 // R/W [31:0] 0x2009
#define MEMCTL_SREFR      (DW_MEMCTL_BASE + 0x10)/2 // R/W [31:0] 0x64
#define MEMCTL_SCSLR0_LOW  (DW_MEMCTL_BASE + 0x14)/2
#define MEMCTL_SCSLR0_HIGH (DW_MEMCTL_BASE + 0x34)/2
#define MEMCTL_SMSKR0      (DW_MEMCTL_BASE + 0x54)/2 // R/W [31:0] 0x80b
#define MEMCTL_SMTMGR_SET0 (DW_MEMCTL_BASE + 0x94)/2 // R/W [31:0] 0x10441
#define MEMCTL_SMTMGR_SET1 (DW_MEMCTL_BASE + 0x98)/2 // R/W [31:0] 0x7c4f5b
#define MEMCTL_SMTMGR_SET2 (DW_MEMCTL_BASE + 0x9c)/2 // R/W [31:0] 0x1c4f5b
#define MEMCTL_FLASH_TRPDR (DW_MEMCTL_BASE + 0xa0)/2 // R/W [31:0] 0xc8
#define MEMCTL_SMCTLR     (DW_MEMCTL_BASE + 0xa4)/2 // R/W [31:0] 0x2480
#define MEMCTLMEMCTL_PING_1BIT_WR (MEMCTL_SCONR)/2

```

////////// Registers of APB Register File //////////

```

#define I2DSP_TXCONF    0x1ec00/2 //w/r [5:0]; 6'h0; //i2dsp 0x1_ec00
#define I2DSP_TXCOM     0x1ec08/2 //w/r [2:0]; 3'h0; // tx_en,i1dsp_en,tx_dma_en
#define I2DSP_TXDB      0x1ec10/2 //w 16'h0;
#define I2DSP_DPCTRL    0x1ec20/2 //w/r [1:0]; 2'h0; // rxiforst,txiforst
#define I2DSP_RXCONF    0x1ec30/2 //w/r [5:0]; 6'h0; // [5:3]:fs,[2]:master,[1:0]:24-16bits
#define I2DSP_RXCOM     0x1ec34/2 //w/r [1:0]; 2'h0; // rx_en,i1dsp_en,rx_dma_en
#define I2DSP_STATUS    0x1ec38/2 //r [2:0]; 3'h0; // rxifonemp,txifonfull
#define I2DSP_RXDB      0x1ec40/2 //r [2:0]; 3'h0;
#define I2C_EXT         0x1ec60/2 //w/r [1:0]; 2'h0; //I2DSP_EXT,I2C_EXT
#define SW_CODEC_RSTN   0x1ec64/2 //w/r [0]; 1'h0; //SW_CODEC_RSTN

#define PWM_TACMD      0x1ec84/2 //w/r [1:0]; 2'h0; //pwm 0x1_ec80
#define PWM_TADATA0    0x1ec88/2 //w/r [15:0]; 16'h0;
#define PWM_TADATA1    0x1ec8c/2 //w/r [15:0]; 16'h0;
#define PWM_TAPRE       0x1ec90/2 //w/r [9:0]; 10'h0;

#define ADC_ADCCON     0x1ed00/2 //w/r [4:0]; 5'h0; //adc 0x1_ed00
#define ADC_ADCDAT0    0x1ed04/2 //r [9:0]; 10'h3ff
#define ADC_ADCFRE     0x1ed08/2 //w/r [7:0]; 8'h9;
#define ADC_ADCRDY     0x1ed0c/2 //r [0]; 1'h1

#define DCDC_CON0      0x1ed80/2 //w/r [2:0]; 3'h0; //dcdc 0x1_ed80
#define DCDC_CON1      0x1ed88/2 //w/r [2:0]; 3'h0;

#define CLOCK_MCLKCON  0x1ee00/2 //w/r [7:0]; 8'h1; //clock 0x1_ee00
#define CLOCK_I2SMCLKCON 0x1ee04/2 //w/r [8:0]; 9'h107
#define CLOCK_AHBCLKCON 0x1ee08/2 //w/r [1:0]; 2'h0;
#define CLOCK_APBCLKCON 0x1ee0c/2 //w/r [1:0]; 2'h0;
#define CLOCK_PLL_NDIV  0x1ee10/2 //w/r [4:0]; 5'h6;
#define CLOCK_PLL_MDIV  0x1ee14/2 //w/r [8:0]; 9'h1e;
#define CLOCK_PLL_ODDIV 0x1ee18/2 //w/r [1:0]; 2'h0;
#define CLOCK_PLL_PDBP  0x1ee1c/2 //w/r [1:0]; 2'h1;
#define CLOCK_PLL_BP    0x1ee1c/2 //w/r [1:0]; 2'h1;
#define CLOCK_PWRCON    0x1ee38/2 //w/r [2:0]; 3'h7;

```

## Rock2 DATA SHEET V1.1

```
#define GPIO_PCON0    0x1ee80/2 //w/r [15:0]; 16'b0; //gpio 0x1_ee80
#define GPIO_PDATA0   0x1ee88/2 //w/r [7:0]; 8'b0;
#define GPIO_PCON1    0x1ee90/2 //w/r [15:0]; 16'b0;
#define GPIO_PDATA1   0x1ee98/2 //w/r [7:0]; 8'b0;
#define GPIO_PCON2A   0x1eea0/2 //w/r [15:0]; 16'b0;
#define GPIO_PCON2B   0x1eea4/2 //w/r [15:0]; 16'b0;
#define GPIO_PDATA2   0x1eea8/2 //w/r [15:0]; 16'b0;
#define GPIO_PCON3    0x1eeb0/2 //w/r [15:0]; 16'b0;
#define GPIO_PDATA3   0x1eeb8/2 //w/r [7:0]; 8'b0;
#define GPIO_EXTINTR  0x1eec0/2 //w/r [7:0]; 8'b0;

#define SDCI_CTRL     0x1ef00/2 //w/r [4:0]; 5'h0; //sdmmc 0x1_ef00
#define SDCI_DCTRL    0x1ef04/2 //w/r [2:0]; 3'h0;
#define SDCI_DLEN     0x1ef08/2 //w/r [15:0]; 16'h0;
#define SDCI_TOKEN    0x1ef0c/2 //w/r [7:0]; 8'h0;
#define SDCI_CMD      0x1ef10/2 //w/r {[15:8],1'b0,[6:0]}; 16'h0;
#define SDCI_ARG      0x1ef14/2 //w/r [31:0]; 32'h0;
#define SDCI_STAC    0x1ef18/2 //w {[15:14],5'b0,[8:0]}; 16'h0;
#define SDCI_STA     0x1ef1c/2 //r [15:0]; 16'h3000;
#define SDCI_RESP0    0x1ef20/2 //r [31:0]; 32'h0;
#define SDCI_RESP1    0x1ef24/2 //r [7:0]; 8'h0;
#define SDCI_TX      0x1ef28/2 //w [31:0]; 32'h0;
#define SDCI_RX      0x1ef2c/2 //r [31:0]; 32'h0;
#define SDCI_CRC16   0x1ef30/2 //r [15:0]; 32'h0;

//codec i2c address
#define P_ADD_AICR   0b00000000 //0x00 //w/r [7:0]; 8'h40//00
#define P_ADD_CR1    0b00000010 //0x02 //w/r [7]r,[6:0]; 8'h08
#define P_ADD_CR2    0b000000100 //0x04 //w/r [7]r,[6:0]; 8'h78//88
#define P_ADD_CCR    0b000000110 //0x06 //w/r [7:4],[2:0]; 8'h00
#define P_ADD_PMR1   0b000001000 //0x08 //w/r [7:0]; 8'h27//26
#define P_ADD_PMR2   0b000001010 //0x0a //w/r [7:3],[1:0]; 8'h00
#define P_ADD_CGR1   0b000001100 //0x0c //w/r [7:0]; 8'h00
#define P_ADD_CGR2   0b000001110 //0x0e //w/r [7:6],[4:0]; 8'h04
#define P_ADD_CGR3   0b000010000 //0x10 //w/r [4:0]; 8'h04
#define P_ADD_CGR4   0b000010010 //0x12 //w/r [7:6],[4:0]; 8'h04
#define P_ADD_CGR5   0b000010100 //0x14 //w/r [4:0]; 8'h04
#define P_ADD_CGR6   0b000010110 //0x16 //w/r [7:6],[4:0]; 8'h04
#define P_ADD_CGR7   0b000011000 //0x18 //w/r [4:0]; 8'h04
#define P_ADD_CGR8   0b000011010 //0x1a //w/r [7:6],[4:0]; 8'h0a
#define P_ADD_CGR9   0b000011100 //0x1c //w/r [4:0]; 8'h0a
#define P_ADD_CGR10  0b000011110 //0x1e //w/r [7:0]; 8'h00
#define P_ADD_TR1    0b00100000 //0x20 //w/r [7:0]; 8'h00
#define P_ADD_TR2    0b00100010 //0x22 //w/r [7:0]; 8'hd8
```

## 5 Electrical Characteristics

### 5.1 Absolute maximum Ratings

#### Absolute Maximum Ratings

PARAMETER	MIN	MAX	UNITS
Ambient operating temperature	-10	80	° C
Storage temperature	-40	120	° C
VDDPLL	-0.3	2.0	V
VCC	-0.3	3.6	V
Input voltage on USB DM, DP pins (USBIO)	-0.3	3.6	V
Input voltage on any digital I/O pin relative to ground	-0.3	VCC+0.3	V
Input voltage on any analog pin relative to ground	-0.3	VDDA+0.3	V

### 5.2 Operating Conditions

**Table 11: Recommended operation conditions.**

PARAMETER	MIN	TYP	MAX	UNITS
Digital core supply voltage -- VDD	1.6	1.8	2.0	V
Digital I/O supply voltage – VCC	2.6	2.9	3.3	V
Analog supply voltage – VDDA	1.6	1.8	2.0	V
Analog HP-AMP supply voltage – VDDAO	1.6	1.8	2.0	V
10bit ADC supply voltage – VCCA	2.6	2.9	3.3	V
USB supply voltage – VCCA	2.6	2.9	3.3	V
USB supply voltage – VDDA (Generate internal)	1.6	1.8	2.0	V

## 5.3 DC Characteristics

**operating conditions:**  $V_{CC} = 3.0V \pm 0.3V$ ,  $V_{DD} = 1.8V \pm 0.2V$ ,  $T_j = -40$  to  $+120$  °C.

Parameter	Symbol	min	typ	max	unit
Quiescent Supply Current, $V_I = V_{DD}$ or $V_{SS}$	$I_{BDS}$				mA
High Level Output Voltage (2mA)	$V_{OH}$	0.8xVCC			V
Low Level Output Voltage (2mA)	$V_{OL}$			0.2xVCC	V
High Level Input Voltage 3V buffers	$V_{IH}$	0.7VCC		VCC+0.6	V
Low Level Input Voltage 3V buffers	$V_{IL}$	-0.3		0.4VCC	V
Input leakage current	$I_{LI}$			$\pm 10$	$\mu A$
Output leakage current	$I_{LO}$			$\pm 5$	$\mu A$
GPIO drive			8		mA
Operating supply current, $f_{clk}=12MHz$ , $T_a=25^{\circ}C$ , $V_{CC}=3.0V$ , $V_{DD}=1.8V$	$I_{VDD}$		35	50	mA

## 5.4 AC Characteristics

Line input to ADC

Measurement conditions: $T = 25^{\circ}C$ , $V_{DDA} = V_{DDAO} = V_{REFP} = 1.8V$ , input sine wave with a frequency of 1kHz, $F_{mclk} = 12MHz$ , $F_s = 48kHz$ , measurement bandwidth 20Hz – 20kHz, unless otherwise specified.					
Parameter	Test conditions	Min.	Typ	Max.	Unit
Input level	Full Scale, Gain GIL, GIR = 0dB (note 1)	1.34	1.53	1.72	Vpp
SNR	A-weighted, 1kHz sine wave @ Full Scale and gain GIL, GIR = 0dB	80	85		dB
THD	1kHz sine wave @ Full Scale -1dB and gain GIL, GIR = 0dB		-85	-80	dB
Dynamic range	A-weighted, 1kHz sine wave @ Full Scale -60dB and gain = 0dB (note 2)	80	85		dB
PSRR	100mVpp 1kHz sinewave is applied to VDDA, input data is 0 and gain GIL, GIR = 0dB		50		dB
Gain range	GIL, GIR 4-bit programmable range, @1kHz	0		22.5	dB
Gain step	GIL, GIR		1.5		dB
Gain accuracy	GIL, GIR @1kHz	-1		+1	dB

## Rock2 DATA SHEET V1.1

Input resistance		30	40	50	kOhm
Input capacitance	Includes 10pF for ESD, bonding and package pins capacitances			25	pF
Input bypass capacitor	Cbyline		1		uF
Offset (note 3)	Gain GIL, GIR = 0dB, without High Pass Filter		±200		LSB
	Gain GIL, GIR = 22.5dB, without HPF	-2000		+2000	
	Gain GIL, GIR = 0 or 22.5dB, with HPF	-4		+4	

Note 1: The Full Scale input voltage scales with VDDA, equals to  $0.85 \times VREF$  (typ)

Note 2: The specified value is extrapolated by adding 60dB to the measured SNR

Note 3: With 1 LSB =  $55\mu V$  for a 16-bit word and  $VREF=1.8V$

### Microphone input to ADC

Measurement conditions:						
$T = 25^\circ C$ , $VDDA = VDDAO = VREFP = 1.8V$ , input sine wave with a frequency of 1kHz, $Fmclk = 12MHz$ , $Fs = 48kHz$ , measurement bandwidth 20Hz – 20kHz, unless otherwise specified.						
Parameter	Test conditions	Min.	Typ	Max.	Unit	
Input level	Full Scale, Gain GIL, GIR = 0dB, boost gain GIM = 20dB (note 1)	0.134	0.153	0.172	Vpp	
	A-weighted, 1kHz sine wave @ Full Scale and gain GIL, GIR = 0dB, boost gain GIM = 0dB	80	85		dB	
SNR	A-weighted, 1kHz sine wave @ Full Scale and gain GIL, GIR = 0dB, boost gain GIM = 20dB	70	75		dB	
	1kHz sine wave @ Full Scale –1dB and		-65	-60		
THD	gain GIL, GIR = 0dB, boost gain GIM = 20dB				dB	
	A-weighted, 1kHz sine wave @ Full Scale –60dB and gain GIL, GIR = 0dB, boost gain GIM = 0dB (note 2)	80	85		dB	
	A-weighted, 1kHz sine wave @ Full Scale –60dB and gain GIL, GIR = 0dB, boost gain GIM = 20dB (note 2)	70	75		dB	
Dynamic range						

## Rock2 DATA SHEET V1.1

PSRR	100mVpp 1kHz sinewave is applied to VDDA, input data is 0 and gain GIL, GIR = 0dB, boost gain GIM = 20dB		50		dB
Gain boost	Boost gain GM when activated		20		dB
Input resistance	Gain GIL, GIR = 0 or 22.5dB, boost gain GIM = 20dB	14	20	26	kOhm
Input capacitance	Includes 10pF for ESD, bonding and package pins capacitances			25	pF
Input bypass capacitor	Cbymic		1		uF
Offset (note 3)	Gain GIL, GIR = 0dB, boost gain GIM = 20dB, without High Pass Filter		$\pm$ 1500		LSB
	Gain GIL, GIR = 22.5dB, boost gain GIM = 20dB, without HPF	-13e3		+13e3	
	Gain GIL, GIR = 0 or 22.5dB, boost gain GIM = 20dB, with HPF	-2		+2	

Note 1: The Full Scale input voltage scales with VDDA, equals to  $0.085 \times VREF$  (typ)

Note 2: The specified value is extrapolated by adding 60dB to the measured SNR

Note 3: With 1 LSB = 55μV for a 16-bit word and VREF=1.8V

DAC to headphone output

Measurement conditions:					
$T = 25^\circ C$ , VDDA = VDDAO = VREFP = 1.8V, input sine wave with a frequency of 1kHz, Fmclk = 12MHz, Fs = 48kHz, measurement bandwidth 20Hz – 20kHz, unless otherwise specified.					
Parameter	Test conditions	Min.	Typ	Max.	Unit
Output level	Full Scale, Gain GOL, GOR, GODL, GODR = 0dB (note 1)	1.34	1.53	1.72	Vpp
	RL = 16 Ohm		18		mW
Maximum power output	RL = 32 Ohm		9		
	A-weighted, 1kHz sine wave @ Full Scale and gain GOL, GOR, GODL, GODR = 0dB	85	90		dB
THD	1kHz sine wave @ Full Scale – 1dB and gain GOL, GOR, GODL, GODR = 0dB, 16 Ohm load		-65	-60	dB

## Rock2 DATA SHEET V1.1

	1kHz sine wave @ Full Scale – 1dB and gain GOL,GOR,GODL,GODR = 0dB, 10kOhm load		-85	-80	
Dynamic range	A-weighted, 1kHz sine wave @ Full Scale –60dB and gain GOL, GOR,GODL,GODR = 0dB (note 2)	85	90		dB
PSRR	100mVpp 1kHz sinewave is applied to VDDA, input data is 0 and gain GOL,GOR,GODL,GODR = 0dB		50		dB
Channel separation	16 Ohm loads @1kHz, AOM and AOMS have to be connected together as close as possible of the headphone connector		65		dB
Gain range	GOL, GOR 5-bit programmable range, mixed analog/digital control, @1kHz	-32		+6	dB
Gain step	GOL, GOR: +6 dB ~ +2 dB		0.5		dB
	GOL, GOR: +2 dB ~ -10 dB		1		
	GOL, GOR: -10 dB ~ -32 dB		2		
Gain accuracy	GOL, GOR @1kHz	-1		+1	dB
Gain range	GODL, GODR 4-bit programmable range, mixed analog/digital control, @1kHz	-22.5		+0	dB
Gain step	GODL, GODR		1.5		dB
Gain accuracy	GODL, GODR @1kHz	-1		+1	dB
Output resistance	Rl	16			Ohm
Output capacitance	Cp			100	pF
Output bypass capacitor	C1 (Rl = 10 kOhm)			1	uF
Offset error	Gain GOL,GOR,GODL,GODR = 0dB	-50		+50	mV

Note 1: The Full Scale output voltage scales with VDDA, equals to 0.85\*VREF (typ)

Note 2: The specified value is extrapolated by adding 60dB to the measured SNR

## **5.5 Package & Dimensions**

### **5.5.1 Packaging Type**

128/100 pins LQFP package

### **5.5.2 Demensions**

128/100: 14mm x 14mm

### **5.5.3 Marking**

Following markings are printed on the package of the ASIC

Manufacturer: Rockchip  
customer markings: Rock260X  
Manufacturer part number:  
Lot code:

ROCKCHIP