NX5850

User's Manual

Version 0.5 Sept. 23, 2005

Preliminary

NEXIADEVICE

1. Table of Contents

2. Product Overview

2. Product Overview

NX5850 audio application SoC provides a high-performance audio processing for MP3 and WMA applications such as the digital audio players, PDAs, voice recorders, MP3 recorders, and cell phones.

Based on the Hardwired Logic Design, NX5850 offers the superior power efficiency coupled with high performance. The SoC integrates the DMA used for the audio data processing, the USB 2.0 device and USB1.1 host used for downloading and uploading audio data from/to a PC and a comprehensive set of peripherals to support some medias such as MMCs, SDs, Flash memories and etc.

2.1. Features

General Features

- Operating Voltage : 3.3V (I/O) / 1.8V (Core)
- Package : 128LQFP (14mm x 14mm)
- Low Frequency and Ultra Low Power Operating
- Crystal Oscillator : 12MHz
- Program ROM : External NOR/NAND Flash
- Update Firmware through USB Ports
- LCD Interface : Serial/Parallel/DMA
- Audio Effect Control
  srs wow 3D effects
  10 band equalizer
  Bypass / Volume / Bass / Treble / Normal / Mute
- Includes RTC(Real Time Clock)
- Includes MP3 Decoder, MP3 Encoder, WMA Decoder, DMA, USB, GPIOs
- Media Interfaces and 8bit RISC MCU of 8051 Compatible

Audio / Voice CODEC

MP3 Full Function for Encoding and Decoding

- Includes MP3 decoder and MP3 encoder
- MP3 Encoder for MPEG1/2 Audio Layer 3 and MP3 Decoder for MPEG1/2/2.5 Audio Layer 3
- Includes CODEC for Voice (8 ~ 160Kbps)
- Real Time Processing of Encoding/Decoding in MP3 Format
- Based on the fully Hardwired Logic Design for Power Efficiency

WMA Decoder Full function

- Based on the fully Hardwired Logic Design for Power Efficiency
- Includes WMA decoder
- Real time processing of decoding in WMA format
- Based on the fully hard-wired logic design for power efficiency
- WMA DRM version 9 (PD-DRM) is supported

DMA Control

- Structure to handle memories for DMA with high performance

USB 2.0 Full Speed

- The embedded USB2.0 full speed controller for device function

USB 1.1 Host Controller

- The embedded USB 1.1 controller for host function
- NX5850 supports one port of USB host interface that has the following features such as OHCI

Media Interfaces

- SMC and NAND Flash (with ECC)
- MMC, SD, CF, NAND, SRAM

Analog Part

- Includes 4 channel ADC for Voice Recording, Battery Detection and Key Function
- Includes PLL

WMA DRM

- WMA DRM Version 9 (PD-DRM) is supported

Equalizer 10 Bands
SRS WOW 3D Sounds
Real Time Clock

2.2. Target Application

MP3 / WMA Player and Voice Recorder

- Audio Applications to handle Audio Function for the Portable, Home and Car categories
- Voice Applications to handle Voice Recording Functions
- Others such as Education System, Mobile phone, PDA and etc.

2.3. Block Diagram of NX5850

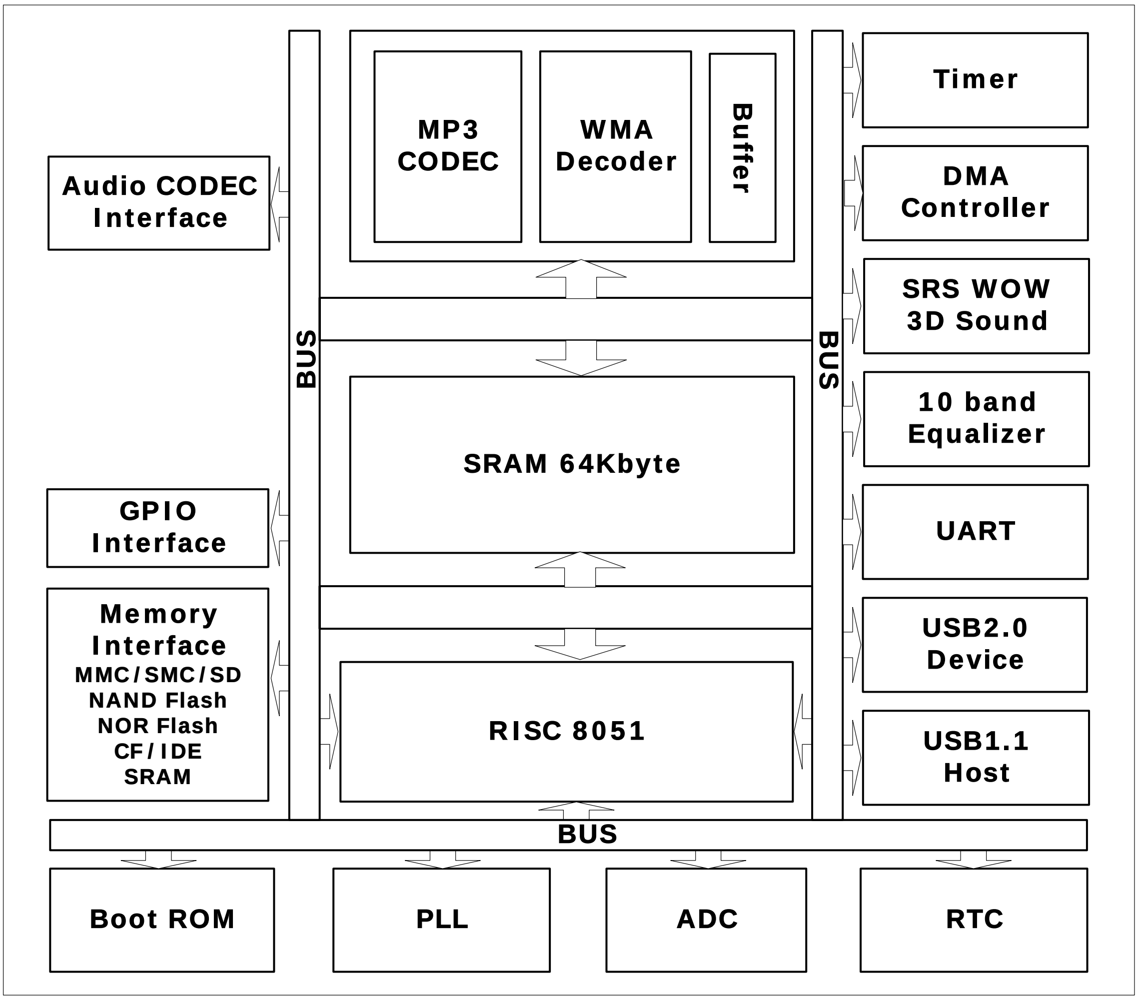

Figure 1. The Block Diagram of NX5850

3. Signal Descriptions

3. Signal Descriptions

Table 1. 8051 Ports Signal Description

Signal Name	Type	Description	Pin Number
P1 [7:0]	I/O	**8051 Port 1** P1 is an 8-bit bidirectional I/O port.	35,34,33,32,31,30,29,28
P2 [7:0]	I/O	**8051 Port 2** P2 is an 8-bit bidirectional I/O port.	82,73,72,83,69,24,70,71
P3 [7:0]	I/O	**8051 Port 3** P3 is an 8-bit bidirectional I/O port.	45,44,43,42,41,40,39,38
MADDR [7:0]	I/O	**8051 Lower Address [7:0]** These bits are latched output of 8051 Port 0 in NX5850 .	84,85,86,87,10,11,12,13
UADDR [5:0]	I/O	**Nor flash/RAM Upper Address [21:16]** These bits can be used when external NOR flash/RAM program code data exceeds 64Kbyte.	79,78,77,76,75,74

Table 2. System Signal Description

Signal Name	Type	Description	Pin Number
RESETn	I	**Chip Reset**	126
TM	I	**Chip Test Mode Selection pin**	4

Table 3. General Purpose IO Description

Signal Name	Type	Description	Pin Number
GPIO0 [7:0]	I/O	**GPIO0 [7:0] (PSEN,ALE,GPIO5~GPIO0)** General purpose input/output group 0	25,46,79,78,77,76,75,74
GPIO1 [7:0]	I/O	**GPIO1 [4:0] (DM,DP,F_RNB,LCD,XRM)** General purpose input/output group 1	9,8,88,128,127
GPIO2 [7:0]	I/O	**GPIO2 [7:0] (ADDR[7:0])** General purpose input/output group 2	84,85,86,87,10,11,12,13
GPIO3 [7:0]	I/O	**GPIO3 [7:0] (LCDDATA[7:0]=P0[7:0])** General purpose input/output group 3	23,22,21,20,19,16,15,14
GPIO4 [5:0]	I/O	**GPIO4 [5:0] (P1[7:0])** General purpose input/output group 4	35,34,33,32,31,30,29,28
GPIO5 [6:0]	I/O	**GPIO5 [6:0] (P2[7:0])** General purpose input/output group 5	82,73,72,83,69,24,70,71
GPIO6 [7:0]	I/O	**GPIO6 [7:0] (P3[7:0])** General purpose input/output group 6	45,44,43,42,41,40,39,38
GPIO7 [7:0]	I/O	**GPIO7 [7:0] (FD [7:0]/NRA[15:8])** General purpose input/output group 7	89,90,91,92,93,94,95,96
GPIO8 [7:0]	I/O	**GPIO8 [7:0] (NorLCD_D[15:8])** General purpose input/output group 8	56,55,54,53,52,51,50,49
GPIO9 [7:0]	I/O	**GPIO9 [7:0] (MCMD,MDAT,MCLK,MCK,SCK,CCK,SDI,SDO)** General purpose input/output group 9	66,65,64,63,62,61,60,59
GPIO10 [7:0]	I/O	**GPIO10 [7:0] (CE3,CE2,CE1,CE0,FCLE,FALE,FWEN,FREN)** General purpose input/output group 10	97,98,99,100,103,104,105,106

Table 4. Analog Signal Description

Signal Name	Type	Description	Pin Number
AIN [3:0]	AI	**Analog Input** These pins are analog input for internal ADC, which converts the analog input signal into 8-bit binary digital codes at a maximum conversion rate of 1MSPS with 10Mhz clock.	118,119,120,121
VBOT	AI	**Reference Bottom** Bottom level of the ADC. The pin is connected to analog ground.	122
VTOP	AI	**Reference Top** The analog input is single-ended type and the range is from VTOP to VBOT. The analog input voltage follows reference voltage range fundamentally. So, if you want to alter into the another input range, you should change the voltage value of VTOP.	123

Table 5. External Memory Interface (MMC) Signal Description

Signal Name	Type	Description	Pin Number
MMC CLK	I/O	**MMC Clock**	64
MMC DATA	I/O	**MMC Data 0**	65
MMC CMD	I/O	**MMC Command**	66
MMC DATA1	I/O	**GPIO0** **MMC DATA 1**	74
MMC DATA2	I/O	**GPIO1** **MMC DATA 2**	75
MMC DATA3	I/O	**GPIO2** **MMC DATA 3**	76

Table 6. External Nand Flash Memory Interface Signal Description

Signal Name	Type	Description	Pin Number
FCEN3	I/O	**NAND Chip Enable 3**	97
FCEN2	I/O	**NAND Chip Enable 2**	98
FCEN1	I/O	**NAND Chip Enable 1**	99
FCEN0	I/O	**NAND Chip Enable 0**	100
FCLE	I/O	**NAND Command Latch Enable**	103
FALE	I/O	**NAND Address Latch Enable**	104
FWEN	I/O	**NAND Write Enable Strobe**	105
FREN	I/O	**NAND Read Enable Strobe**	106
FRNB	I/O	**NAND Ready and Busy**	88
FIO [7:0]	I/O	**NAND I/O [7:0]**	89,90,91,92,93,94,95,96

Table 7. External Nor Flash Memory Interface Signal Description

Signal Name	Type	Description	Pin Number
NRCS1	I/O	**NOR Chip Enable 1 (F_CE1)**	99
NRCS0	I/O	**NOR Chip Enable 0 (F_CE0)**	100
NWR	I/O	**NOR Write Data Strobe (P3.6)**	44
NRD	I/O	**NOR Read Data Strobe (P3.7)**	45
ADDR [7:0]	O	**NOR Address [7:0]**	84,85,86,87,10,11,12,13
ADDR [15:8]	O	**NOR Address [15:8]**	89,90,91,92,93,94,95,96

ADDR [21:16]	O	**NOR Address[21:16]**	79,78,77,76,75,74
DATA [7:0]	I/O	**NOR Data[7:0] (P0[7:0])**	23,22,21,20,19,16,15,14
DATA [15:8]	I/O	**NOR Data[15:8] (NorLCD_D[15:8])**	56,55,54,53,52,51,50,49

Table 8. RTC Interface (Real Time Clock) Signal Description

Signal Name	Type	Description	Pin Number
RTCIN	I	**Real Time Crystal Oscillator Input** To use the Real Time Clock counter, a 32.768KHz crystal oscillator is connected this pin. If an external oscillator is used, its output is connected to this pin. RTCIN is the clock source for internal RTC counter only.	1
RTCOUT	O	**Real Time Crystal Oscillator Output** To use the Real Time Clock counter, a 32.768KHz crystal oscillator is connected to this pin. If an external oscillator is used, leave RTCOUT unconnected.	2
1.8V RTC VDD	P	**Independent Power for RTC Timer only**	115
3.3V RTC VDD	P	**Independent Power for RTC Oscillator**	125
RTC GND	P	**Power Ground**	114,124

Table 9. Boot Selection and Firmware Update Signal Description

Signal Name	Type	Description	Pin Number
Firmware Update	I	**Firmware Update**	3
Boot Selection	I	**Nor/Nand Flash Memory Boot**	5

Table 10. Audio Codec Interface Signal Description

Signal Name	Type	Description	Pin Number
ACMCK	I/O	**Audio Codec Master Clock**	63
ACSCK	I/O	**Audio Codec Sample Clock**	62
ACCCK	I/O	**Audio Codec Channel Clock**	61
ACSDI	I/O	**Audio Codec Data Input**	60
ACSDO	I/O	**Audio Codec Data Output**	59

Table 11. Clock Signal Description

Signal Name	Type	Description	Pin Number
XIN	I	**System Clock Crystal Oscillator Input** To use the internal oscillator, a crystal/resonator circuit is connected this pin. If an external oscillator is used, its output is connected to this pin. XIN is the clock source for internal timing.	109
XOUT	O	**System Clock Crystal Oscillator Output** To use the internal oscillator, a crystal/resonator circuit is connected to this pin. If an external oscillator is used, leave X2 unconnected.	110

Table 12. USB Interface Signal Description

Signal Name	Type	Description	Pin Number
USBDP	I/O	**USB Positive Data Line**	8
USBDM	I/O	**USB Negative Data Line**	9

Table 13. LCD Interface Signal Description

Signal Name		Type	Description	Pin Number
Parallel Mode	LCDNCE	I/O	**LCD Module Chip Enable**	128
	LCDRS	I/O	**LCD Module Mode Select**	13
	LCDNWR	I/O	**LCD Module Write Enable**	44
	LCDIO [15:0]	I/O	**LCD Module Data [7:0]** **LCD Module Data [15:8]**	23,22,21,20,19,16,15,14 56,55,54,53,52,51,50,49
	LCDNRD	I/O	**LCD Module Read Enable**	45

Table 14. Power Signal Description

Signal Name	Type	Description	Pin Number
VDDe6	P	**Analog I/O Power for Digital Part**	6
GNDe7	G	**Analog I/O Ground for Digital Part**	7
VDDe17	P	**Digital I/O Power for Digital Part**	17
GNDe18	G	**Digital I/O Ground for Digital Part**	18
VDDi26	P	**Digital Core Power for Digital Part 1.8V**	26
GNDi27	G	**Digital Core Ground for Digital Part**	27
GNDe36	G	**Digital I/O Ground for Digital Part**	36
VDDe37	P	**Digital I/O Power for Digital Part**	37
GNDi47	G	**Digital Core Ground for Digital Part**	47
VDDi48	P	**Digital Core Power for Digital Part 1.8V**	48
GNDe57	G	**Digital I/O Ground for Digital Part**	57
VDDe58	P	**Digital I/O Power for Digital Part**	58
GNDi67	G	**Digital Core Ground for Digital Part**	67
VDDi68	P	**Digital Core Power for Digital Part 1.8V**	68
GNDe80	G	**Digital I/O Ground for Digital Part**	80
VDDe81	P	**Digital I/O Power for Digital Part**	81
VDDi101	P	**Digital Core Power for Digital Part 1.8V**	101
GNDi102	G	**Digital Core Ground for Digital Part**	102
VDDe107	P	**Digital I/O Power for Digital Part**	107
GNDe108	G	**Digital I/O Ground for Digital Part**	108
AVDD111	P	**Analog Power for Analog Part 1.8V**	111
AGND112	G	**Analog Ground for Analog Part**	112
AGND114	G	**Digital Ground for Real Time Clock Timer Part**	114
AVDD115	P	**Digital Power for Real Time Clock Timer Part 1.8V**	115
AGND116	G	**Digital Ground**	116
NC	NC	**NC**	117
GND124	G	**Digital Ground for Real Time Clock Part**	124
VDD125	P	**Digital Power for Real time Clock Part**	125

4. System Boot

4. System Boot

4.1. Boot Mode

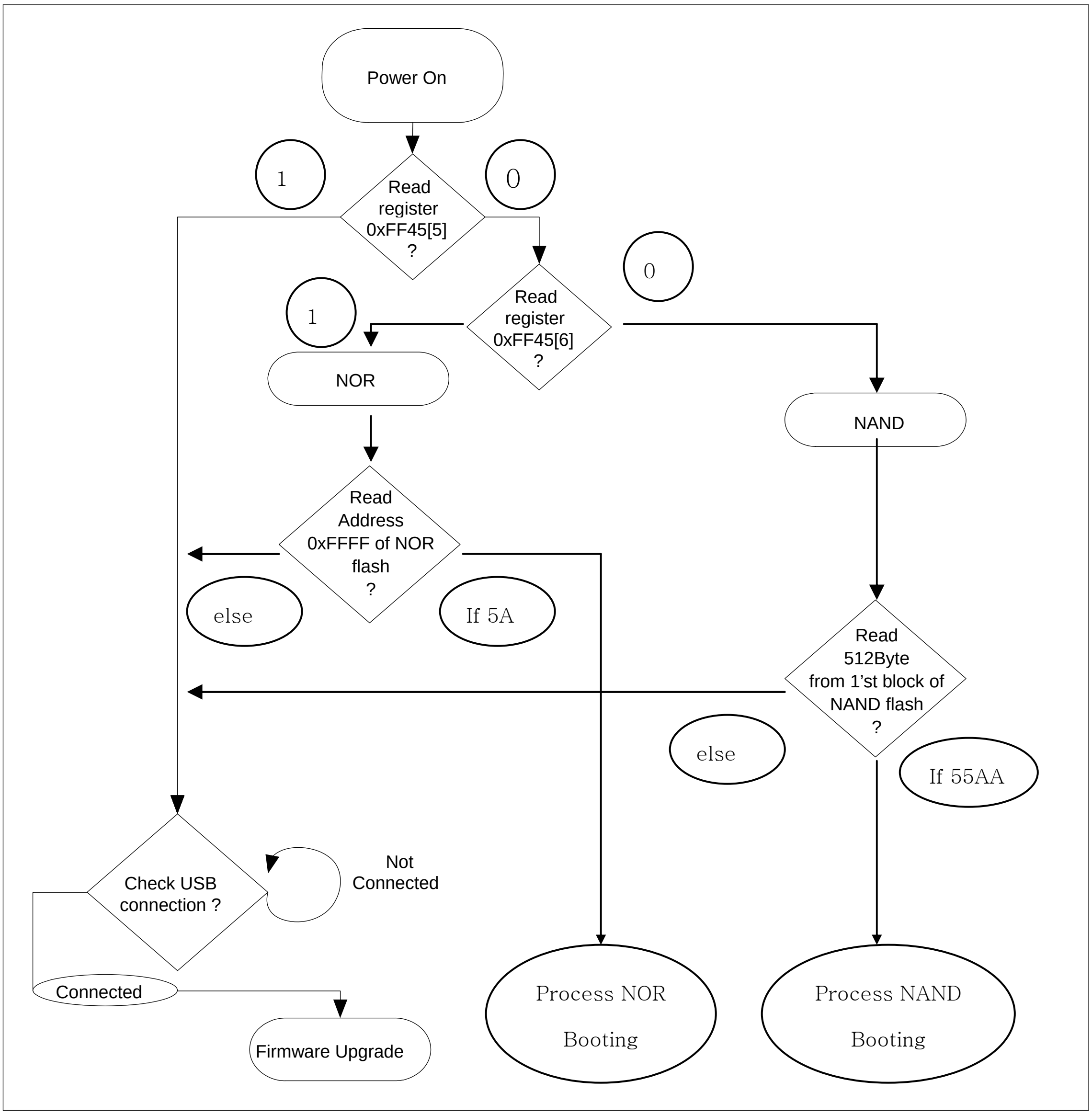

Figure 2. Boot Mode Selection flow chart

Figure 2. shows the selection procedure of the Boot Mode. There are three boot modes such as NOR Flash Memory, Nand Flash Memory and firmware update booting, and the modes are selected by the pin number 3 and 5 on NX5850. User can determine a boot mode of three with any state combination of pin number 3 and 5. But This data sheet recommends pin number 3 as firmware update mode pin and pin number 5 as Nor Flash Memory boot mode pin. If pin 3 is high it's firmware update mode, if pin3 is low normal booting mode. If pin5 is high it's Nor Flash memory booting mode and if pin5 is low it's Nand Flash memory boot mode. Figure 2 is the boot flow chart in this recommend case.

- **NOR Flash Boot Mode**

The program is executed with the program code on the NOR Flash. The NOR Flash(EEPROM) can be upgraded.

- **NAND Flash Boot Mode**

A program is executed with the program code on the NAND flash. The NAND flash can be upgraded.

4.2. The Booting Procedure of NOR Flash Memory

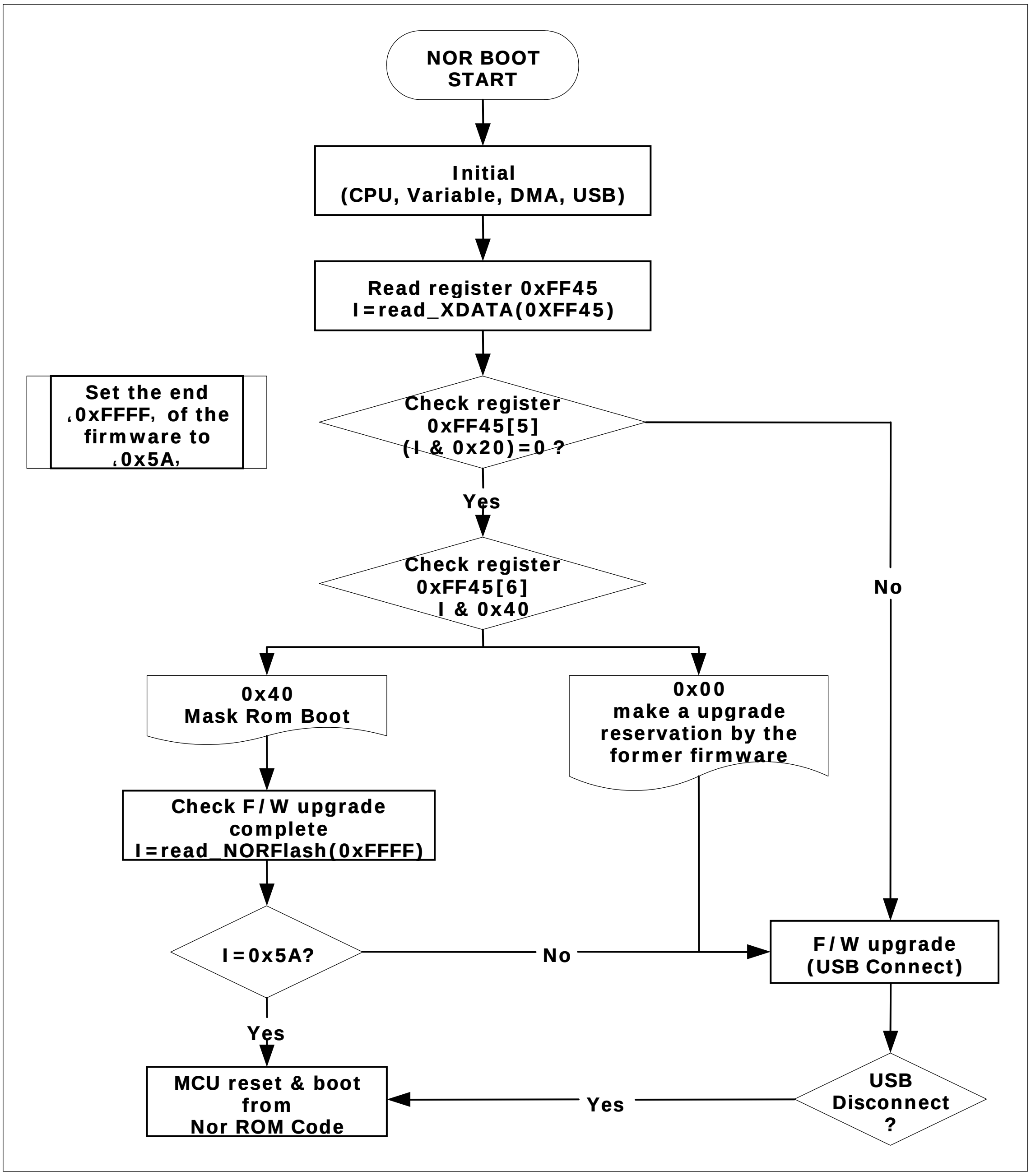

Figure 3. The Flow Chart for NOR Flash Boot Mode

Figure 3. shows the boot mode flow of the NOR flash. The status of the NorFlash/T3(External pin) can be read at the register ‘0xFF45[6]’.

When executing the internal ROM boot by resetting, the NorFlash/T3 has low normally, and when upgrading a firmware by a hardware switch, executing the mask ROM boot, NorFlash/T3 has high. When the NorFlash/T3 is low, it need to check if the firmware in NOR flash is available or not, and then if the firmware is available, get the NOR flash booting to be perform without the firmware upgrade . The way that checks whether the firmware in the NOR flash is available or not is that checks whether the last address ‘0xFFFF’ of the NOR flash is ‘0x5A’ or not. According to this, the firmware to write into the NOR flash have to be 64Kbyte, which is the NOR flash capacity with putting ‘0x5A’ at the last address ‘0xFFFF’ when converting Hex to Binary. The maximum size of the firmware is the size that is lack of 1 byte. The other warning is the firmware upgrade that uses the mask ROM boot code. It is that only operates with the firmware upgrade PC application.

4.3. The Booting Procedure of NAND flash

The followings are a procedure to find a basic code after reading the block 0 of the NAND flash.

Table 15.The contents of Block 0 of the NAND Flash

	0x0	0x1	0x2	0x3	0x4	0x5	0x6	0x7	0x8	0x9	0xA	0xB	0xC	0xD	0xE	0xF
0x0000	0x55	0xAA	0x08	0x00	0x04	0x00	0x00	0x00	0x01	0x00	Rev	Rev	0x30	0x01	0x02	
0x0010																
0x0020																
0x0030																
•																
•																
0x01F0																

1) 0x00 ~ 0x01 : Signature = ‘0x55AA’.
2) 0x02 ~ 0x03 : Sector Size = ‘0x0800’ (2048 byte).
3) 0x04 : Address Number = ‘0x04’ (4 Addresses).
4) 0x05 : Flash Memory Command = ‘0x00’ (Read Command).
5) 0x06 ~ 0x0B : Address Area = ‘0x0A ~ 0x0B’ is reserved because Address is four as above.
6) 0x0C : This is used when the second cycle command is needed. The command ‘0x30’ means the read command on the second cycle. If you need not the second cycle, ‘0xFF’ should be written.
7) 0x0D : Shows the number of sectors of basic code to read.
8) 0x0E : Row Address Number.
9) 0x0F ~ 0x01FF : Reserved Area

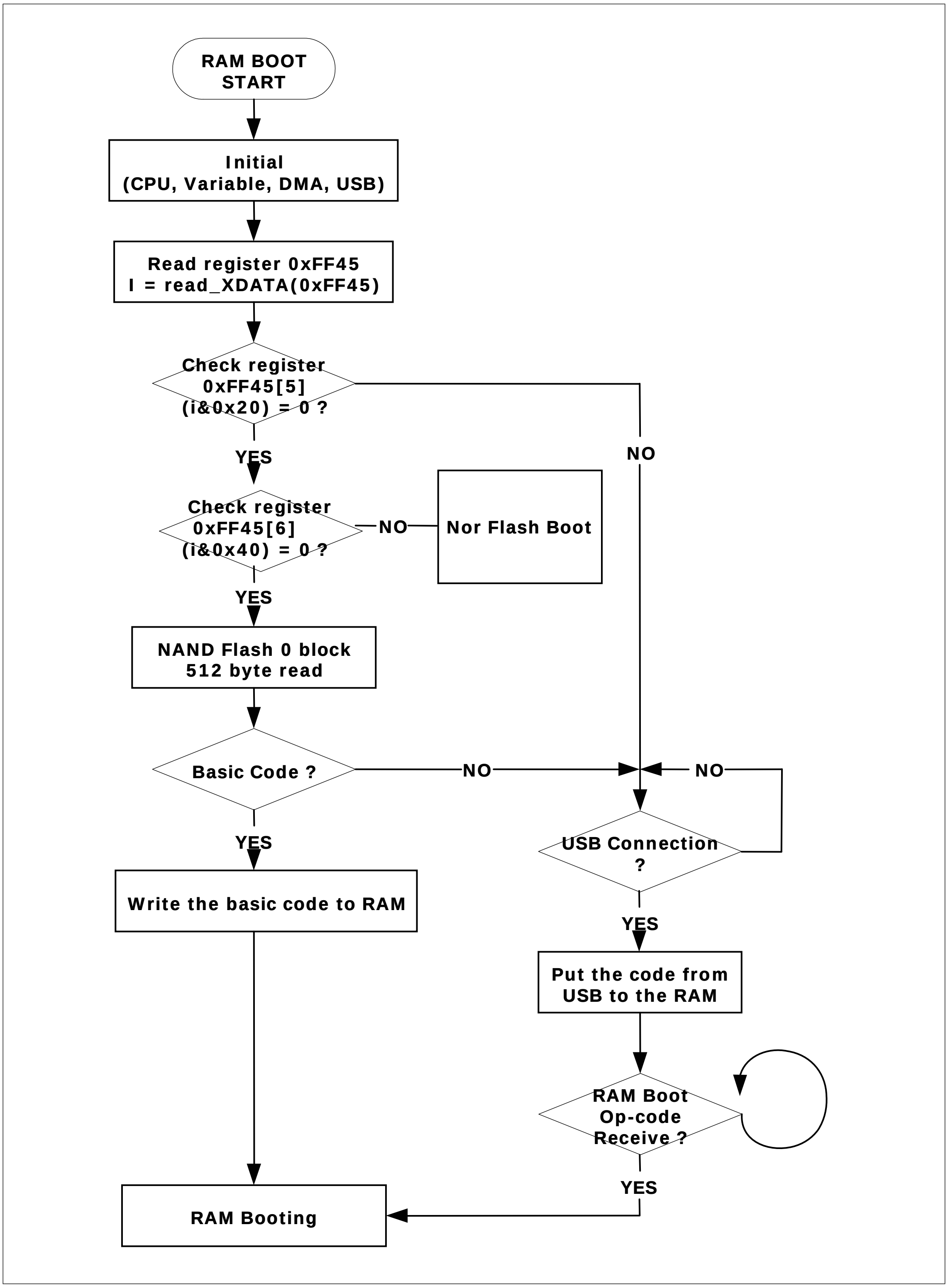

Figure 4. Flow Chart for RAM Boot Mode

A Explanation for Advance Flash Memory of 128Mbyte (Table 15)

If the signature of Table 15. don't has a value of 0x55AA, the booting mode goes to the firmware upgrade mode because it judges that there is no basic code. This means that being in a waiting mode for getting data from the USB like as looking at the flow chart of Figure 4. The code for the upgrading the firmware is put on the RAM, and the code for the firmware upgrade is executed as performing a RAM booting when a OP-code is transferred to notice that the code download is done. If there is the signature, judges that a basic code is exist and executes like follows:

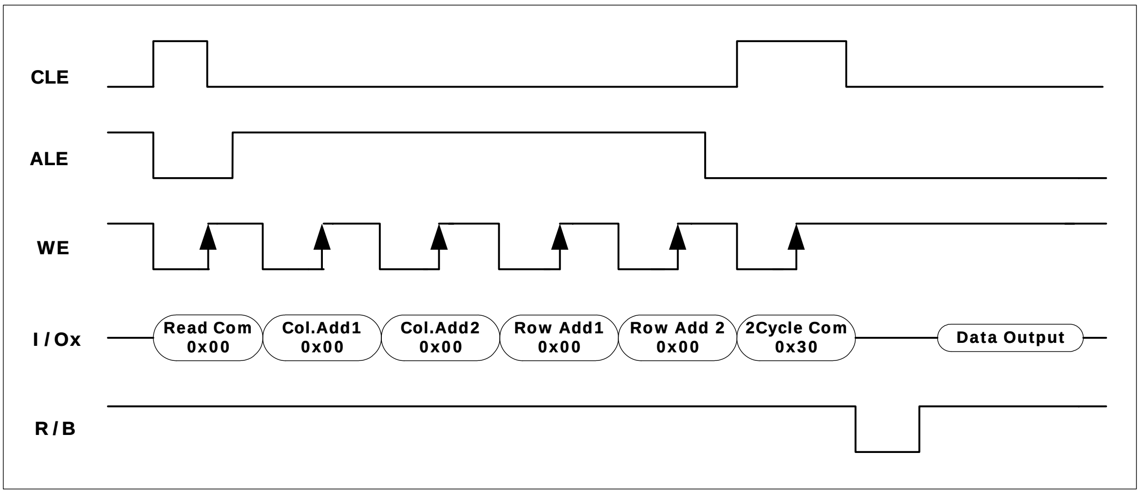

Figure 5. The procedure of Basic code[1]

The Table 15. shows the configuration of the block 0, which has a location for the basic code on the NAND flash.
As seeing the Table 15, the address '0x05' has 0x00(read command), and the '0x04' has 0x04(the number of valid address). The four valid addresses are '00, 00, 01, 00'(0x06 ~ 0x09)and they are written into the NAND flash data IO line. The 0x0C is the second cycle command, and the value is 0x30(read command). To doing the RAM booting, read the 2048 byte(1 sector) from the NAND flash to NX5850 by the DMA transfer. And then the system is reset and the booting code executes on the RAM. If the 0x0D is '2', the booting code executes after reading the second sector. We have to know the start position of the row address in the address '0x06 ~ 0x0B' because the some kinds of NAND flashes have a different size of the column address. So, the 0x0E has 0x02, which is noticed the start of the row address.

4.4. Boot Mode Control Register

Table 16. Boot Mode Register Map

Function	Address (Hex)	Type	Reset	Description
BOOT_MODE_SELECT FIRMWARE UPDATE	0xff45	R[6:0] W[4:0]	8'bXXX00000	BOOT_MODE_SELECT FIRMWARE UPDATE

Boot Mode Select (BOOT_MODE_SELECT, 0xFF45) : Read[6:0] / Write[4:0]

7	6	5	4	3	2	1	0
	Nor Flash Boot	Firmware Update	USB DM	USB DP	F_R/B	LCD	XRM

This register is used to set the boot mode.

Nor Flash Boot : This bit is used to check the Nor Flash Memory booting mode. Read only GPI.

0 : Perform the Nand Flash Memory Boot.
1 : Perform the Nor Flash Memory Boot.

Firmware Update : This bit is used to check the Firmware Update mode. Read only GPI.

0 : Nand or Nor Flash Memory Boot.
1 : Perform the firmware upgrade from USB.

USB DM : This bit is used as GPIO Read/Write at the time of GPIO mode.

USB DP : This bit is used as GPIO Read/Write at the time of GPIO mode.

F_R/B : This bit is used as GPIO Read/Write at the time of GPIO mode.

LCD : This bit is used as GPIO Read/Write at the time of GPIO mode.

XRM : This bit is used as GPIO Read/Write at the time of GPIO mode.

5. System Clock Control

5. System Clock Control

NX5850 has one PLL with external Crystal Oscillator input and system internal clock source is supplied by PLL output through several divider to each function block. Each function block clock is divider output frequency which is PLL output frequency divided by divider value. User sets PLL coefficient register value for PLL out put frequency control and sets divider register value to determine divider output frequency.

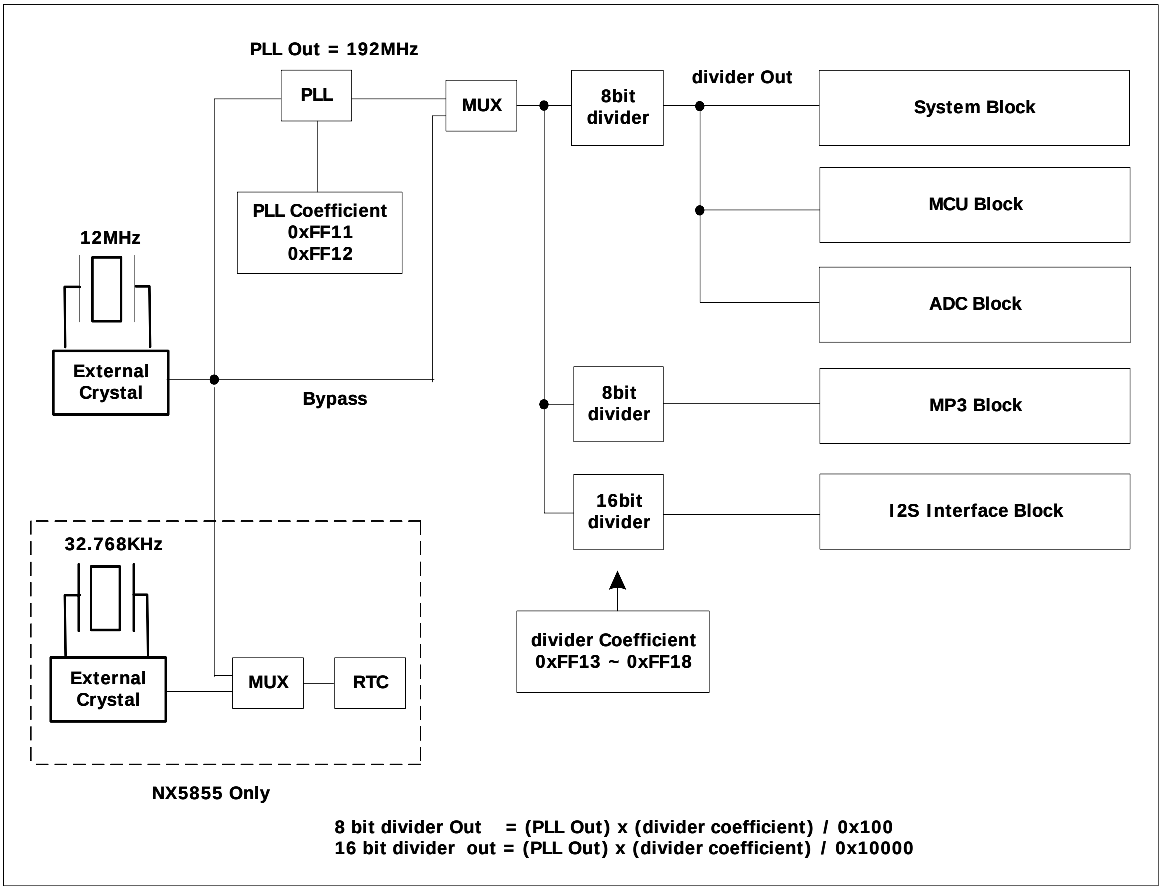

Figure 6. The Diagram of the Clock Distribution

5.1. Clock Source Control Register

Table 17. Clock source control Register Map (P2 = 0xFF)

Function	Address (Hex)	Type	Reset	Description
CLOCK_SOURCE_CONTROL	0x04	R/W	0x07	Clock Source Control
DEFAULT_COEFFICIENT_ENABLE	0x10	R/W	0x1f	Use Default PLL and divider coefficient
PLL_COEFFICIENT_LO	0x11	R/W	0x00	Controls PLL Output Frequency
PLL_COEFFICIENT_HI	0x12	R/W	0x00	
DIVIDER_COEFFICIENT_VALUE	0x13 ~0x18	R/W	0x00	Controls divider Output Frequency

AUDIO_CLOCK_STATUS	0x19	RO	0x0F	Audio Clock Status Register

Clock Source Control (CLOCK_SOURCE_CONTROL, 0xFF04) : Read/Write, 0x07

7	6	5	4	3	2	1	0
Reserved					XTAL_EN	XTAL_ SE	PLL_PWDN

XTAL_EN :
- **0** : stops crystal oscillation.
- **1** : starts crystal oscillation.

XTAL_SE :
- **0** : use pll output clock as internal.
- **1** : use crystal clock as internal clock.

PLL_PWDN :
- **0** : starts pll operation.
- **1** : stops pll operation.

Default Coefficient Enable (DEFAULT_COEFFICIENT_ENABLE, 0xFF10) : Read/Write, 0x1f

7	6	5	4	3	2	1	0
Reserved			PLL_COEF	SYS_DIV	MP3_DIV	reserved	AUD_DIV

PLL_COEF :
- **0** : use register(0xFF11-0xFF12) value as PLL coefficient
- **1** : use default value as PLL coefficient(0x0ABE).

SYS_DIV :
- **0** : use register(0xFF13) value as 8bit divider coefficient
- **1** : use default value as 8bit divider coefficient(0x40).

MP3_DIV :
- **0** : use register(0xFF14) value as 8bit divider coefficient
- **1** : use default value as 8bit divider coefficient(0x10).

reserved :

AUD_DIV :
- **0** : use register(0xFF17-0xFF18) value as 16bit divider coefficient
- **1** : use default value as 16bit divider coefficient(varies depending on sampling frequency).

PLL Coefficient Low (PLL_COEFFICIENT_LOW, 0xFF11) : Read/Write, 0x00

7	6	5	4	3	2	1	0
PLL_COEF_LOW							

This register is used to read/write the lower 8-bit of the 16-bit PLL Coefficient.

PLL Coefficient High (PLL_COEFFICIENT_HIGH, 0xFF12) : Read/Write, 0x00

7	6	5	4	3	2	1	0
PLL_COEF_HIGH							

This register is used to read/write the upper 8-bit of the 16-bit PLL Coefficient.
Pll out = pll in * (coefficient[7:0]+2) / ((coefficient[13:8]+2)*2^coefficient[15:14])

System Divider Value (SYS_DIVDER_VALUE, 0xFF13) : Read/Write, 0x00

7	6	5	4	3	2	1	0
SYS_DIV_VAL							

MP3 Divider Value (MP3_DIVIDER_VALUE, 0xFF14) : Read/Write, 0x00

7	6	5	4	3	2	1	0
MP3_DIV_VAL							

Audio Divider Value Low (AUDIO_DIVIDER_VALUE_LOW, 0xFF17) : Read/Write, 0x00

7	6	5	4	3	2	1	0
AUD_DIV_VAL_LOW							

This register is used to read/write the lower 16-bit of the Audio Divider Value.

Audio Divider Value High (AUDIO_DIVIDER_VALUE_HIGH, 0xFF18) : Read/Write, 0x00

7	6	5	4	3	2	1	0
AUD_DIV_VAL_HIGH							

This register is used to read/write the upper 16-bit of the Audio Divider Value.

Audio Clock Status (AUDIO_CLOCK_STATUS, 0xFF19) : Read Only

7	6	5	4	3	2	1	0
Reserved				AUD_STA			

This register indicates the audio sampling frequency

0000 : 11.025Khz
0001 : 11.025Khz
0010 : 22.05Khz
0011 : 44.1Khz

0100 : 12Khz
0101 : 12Khz
0110 : 24Khz
0111 : 48Khz

1000 : 8Khz
1001 : 8Khz
1010 : 16Khz
1011 : 32Khz

1100 : 8Khz
1101 : 8Khz
1110 : 16Khz
1111 : 32Khz

6. MCU

6. MCU

CPU operating on NX5850 is similar to the general 8051 H/W structure and its operating method. Therefore, the programming method is the same to the general 8051 CPU, only to warn is to set up the controllers by the given specification to interface with respective device controller on NX5850.

6.1. Features

- 8051 8-bit CPU
- General IO ports (Port 1, 2, 3)
- 64Kbyte external RAM on the chip (Compatible to data, code use)
- 256Byte internal RAM on the chip
- Two 16-bit timer/counters
- Full-duplex serial port (UART)
- Power-saving modes support
- DMA interface (Efficiency H/W interface composition)
- Event interrupt 0 (Composition to increase DMA efficiency)

6.2. Address Map

The following Table 18. is the address configuration for the register map by controlling the MCU.

Table 18. Address Allocation Map

Address Range	description	Remarks
0x0000 – 0xF7FF	Internal SRAM area	SRAM for program and data
0xF800 – 0xF8FF	NOR_CTRL area	Control block for LCD/NOR Flash memory
0xF900 – 0xF9FF	DRM_CTRL area	Control block for DRM
0xFA00 – 0xFAFF	MP3_CTRL area	Control block for MP3 decoder/encoder
0xFB00 – 0xFBFF	MMC_CTRL area	Control block for Multi Media Card
0xFC00 – 0xFCFF	NAND_CTRL area	Control block for NAND Flash memory
0xFD00 – 0xFDFF	USB_CTRL area	Control block for USB
0xFE00 – 0xFEFF	DMA_CTRL area	Control block for DMA
0xFF00 – 0xFFFF	SYS_CTRL area	Control block for internal system

7. Nor Flash /LCD Interface

7. Nor Flash/LCD Interface

Nor Flash/LCD Interface is used for Nor Flash memory boot and parallel LCD interface control

7.1. Nor Flash/LCD Interface Register

The description of control registers for Nor Flash memory interface and LCD interface.

Table 19. Nor flash / LCD interface Control Register Map (P2 = 0xF8)

Function	Address (Hex)	Type	Reset	Description
NOR_DMA_INTERRUPT_ENABLE	0x00	R/W	0x00	DMA end interrupt enable control
NOR_DMA_INTERRUPT_STATUS	0x01	R/W	0x00	DMA end interrupt status control
NOR_STROBE_TIMING	0x02	R/W	0x00	NOR DMA strobe timing control
NOR_CONTROL	0x03	R/W	0x00	NOR access type control
NOR_ADDR_HI	0x04	R/W	0x00	NOR address high
NOR_ADDR_LO	0x05	R/W	0x00	NOR address low
NOR_DATA_HI	0x06	R/W	0x00	NOR data high
NOR_DATA_LO	0x07	R/W	0x00	NOR data low

The method to set 20bit address for random access : use offset address 0x04, 0x05, GPIO0~GPIO5.

16bit mode read : Read data from offset address 0x07 first and then offset address 0x06.

16bit mode write : Write data to offset address 0x06 and then offset address 0x07.

8bit mode read/write : use data from/to offset address 0x07 only.

DMA End Interrupt Enable Control (NOR_DMA_INTERRUPT_ENABLE, 0xF800) : Read/Write

7	6	5	4	3	2	1	0
Reserved							INT_EN

NOR_DMA_INTERRUPT_ENABLE[7:1] : Reserved.

INT_EN : Writing each bit with 1 enables interrupt generation when each interrupt condition occurs.

DMA End Interrupt Status Control (NOR_DMA_INTERRUPT_STATUS, 0xF801) : Read/Write

7	6	5	4	3	2	1	0
Reserved							INT_STA

NOR_DMA_INTERRUPT_STATUS[7:1] : Reserved.

INT_STA : Writing 1 clears interrupt status.

NOR Strobe Timing Control(NOR_STROBE_TIMING, 0xF802) : Read / Write

7	6	5	4	3	2	1	0
HI_LEN				LO_LEN			

This defines /RD pin and /WR pin output pulse width(defines read / write speed).

HI_LEN[7:4] : Show the nor strobe timing high period length.

LO_LEN[3:0] : Show the nor strobe timing Low period length.

NOR Flash Access Type Control(NOR_CONTROL, 0xF803) : Read / Write

7	6	5	4	3	2	1	0
Reserved				16_MOD	ADD_INC	DMA_RD	DMA_WR

This defines nor flash memory access type.

NOR_CONTROL[7:4] : Reserved.

16_MOD : This bit used to nor flash memory data access mode.

0 : 16bit data access mode select.
1 : 8bit data access mode select.

ADD_INC : This bit is used to continuously incremental addressing to nor flash memory.

0 : disables address increment.
1 : enables address increment.

DMA_RD : This bit is used to external DMA read enable.

0 : Disabled.
1 : Enabled.

DMA_WR : This bit is used to external DMA write enable.

0 : Disabled.
1 : Enabled.

NOR Flash Address High(NOR_ADDR_HI, 0xF804) : Read / Write

7	6	5	4	3	2	1	0
ADDR_HI							

This register is used to read / write the upper 8-bit of 16-bit of the NOR flash address.

NOR Flash Address Low(NOR_ADDR_LO, 0xF805) : Read / Write

7	6	5	4	3	2	1	0
ADDR_LO							

This register is used to read / write the lower 8-bit of 16-bit of the NOR flash address.

NOR Flash Data High(NOR_DATA_HI, 0xF806) : Read / Write

7	6	5	4	3	2	1	0
DATA_HI							

This register is used to read / write the upper 8-bit of 16-bit of the NOR flash data.

NOR Flash Data Low(NOR_DATA_LOW, 0xF807) : Read / Write

7	6	5	4	3	2	1	0
DATA_LO							

This register is used to read / write the lower 8-bit of 8-bit/16-bit of the NOR flash data.

8. System Control Block

8. System Control Block

System Control Block is for controlling interrupt to CPU, controlling reset and clock source to each function block, controlling RTC, and controlling GPIO.

8.1. System Control Block Register

The description of control registers for System Control Block.

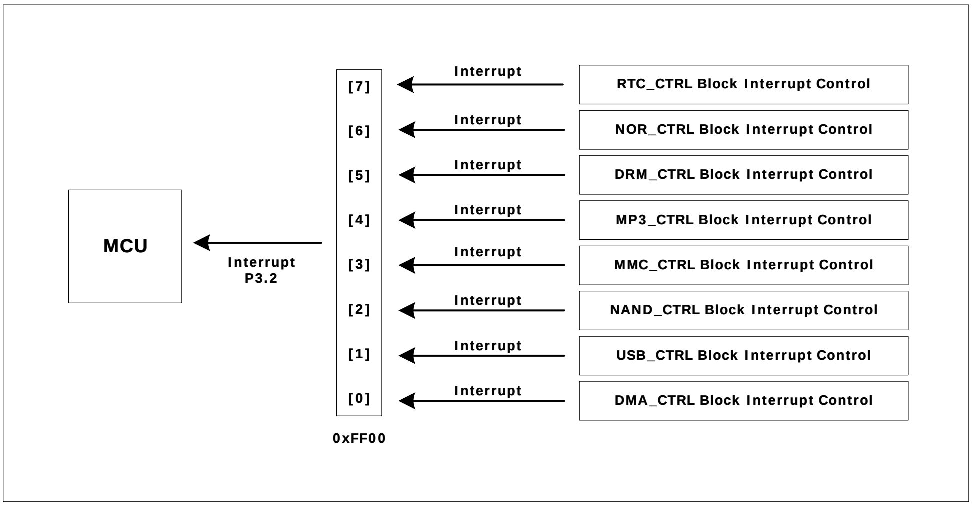

Figure 7. External Interrupt Mask to MCU

Table 20. System Control Register Map (P2 = 0xFF)

Function	Address (Hex)	Type	Reset	Description
BLOCK_INTERRUPT_ENABLE	0x00	R/W	0x00	Each system block enable control
BLOCK_INTERRUPT_STATUS	0x01	RO	-	Each system block status read
ROM_BOOT_MODE	0x02	R/W	0x01	ROM boot mode select
FUNCTION_BLOCK_RESET	0x03	R/W	0x7F	Each function block reset control
NOR_HI_ADDR_ENABLE	0x05	R/W	0x00	Nor flash high address enable control
SYS_BLOCK_POWER_CONTROL	0x06	R/W	0x00	Each system block power control
MP3_BLOCK_POWER_CONTROL	0x07	R/W	0x00	MP3 block power control
DYNAMIC_POWER_CONTROL	0x08	R/W	0x00	Auto power-down control

System Block Interrupt Enable(BLOCK_INTERRUPT_ENABLE, 0xFF00) : Read/Write

7	6	5	4	3	2	1	0
RTC_CTRL	NOR_CTRL	DRM_CTRL	MP3_CTRL	MMC_CTRL	NAND_CTRL	USB_CTRL	DMA_CTRL

Writing each bit with 1 enables interrupt generation if each interrupt condition occurs.

RTC_CTRL : This bit is used to RTC block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

NOR_CTRL : This bit is used to NOR flash or LCD interface block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

DRM_CTRL : This bit is used to DRM block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

MP3_CTRL : This bit is used to MP3 block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

MMC_CTRL : This bit is used to MMC block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

NAND_CTRL : This bit is used to NAND flash interface block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

USB_CTRL : This bit is used to USB block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

DMA_CTRL : This bit is used to DMA block interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

System Block Interrupt Status(BLOCK_INTERRUPT_STATUS, 0xFF01) : Read only

7	6	5	4	3	2	1	0
RTC_STA	NOR_STA	DRM_STA	MP3_STA	MMC_STA	NAND_STA	USB_STA	DMA_STA

Each bit indicates interrupt status of each block. And access to each block which the interrupt generated clears the corresponding bit.

RTC_STA : This bit is used to indicates interrupt generated at the RTC block.

0 : No action.
1 : indicates interrupt generated.

NOR_STA : This bit is used to indicates interrupt generated at NOR flash or LCD interface block.

0 : No action.
1 : indicates interrupt generated.

DRM_STA : This bit is used to indicates interrupt generated at DRM block.

0 : No action.
1 : indicates interrupt generated.

MP3_STA : This bit is used to indicates interrupt generated at MP3 block.

0 : No action.
1 : indicates interrupt generated.

MMC_STA : This bit with indicates interrupt generated at MMC block.

0 : No action.
1 : indicates interrupt generated.

NAND_STA : This bit is used to indicates interrupt generated at NAND flash block.

0 : No action.
1 : indicates interrupt generated.

USB_STA : This bit is used to indicates interrupt generated at USB block.

0 : No action.
1 : indicates interrupt generated.

DMA_STA : This bit is used to indicates interrupt generated at DMA block.

0 : No action.
1 : indicates interrupt generated.

Boot by ROM Mode Control(ROM_BOOT_MODE, 0xFF02) : Read/Write

7	6	5	4	3	2	1	0
Reserved							ROM_BOOT

If user want to reboot from RAM, user write program into RAM from boot address and the write this bit with 0. This resets CPU only and boots from RAM.

ROM_BOOT : This bit is used to boot mode select.

0 : RAM boot mode.
1 : ROM boot mode select. Power reset makes ROM boot mode.

Function Block Reset Control(FUNCTION_BLOCK_RESET, 0xFF03) : Read/Write

7	6	5	4	3	2	1	0
Reserved		ADC_RST	USB_RST	NOR_RST	DRM_RST	MMC_RST	NAND_RST

Writing 0 to each bit makes corresponding function block reset.

ADC_RST : This bit is used to ADC block reset.

0 : ADC Block reset.
1 : No action.

USB_RST : This bit is used to USB block reset.

0 : USB Block reset.
1 : No action.

NOR_RST : This bit is used to NOR block reset.

0 : NOR Block reset.
1 : No action.

DRM_RST : This bit is used to DRM block reset.

0 : DRM Block reset.
1 : No action.

MMC_RST : This bit is used to MMC block reset.

0 : MMC Block reset.
1 : No action.

NAND_RST : This bit is used to NAND flash interface block reset.

0 : NAND flash interface Block reset.
1 : No action.

NOR flash high address enable control (NOR_HI_ADDR_ENABLE, 0xFF05) : Read / Write

7	6	5	4	3	2	1	0
Reserved							NOR_EN

NOR_EN : Writing 0x01 makes 0xF804(NOR flash address high register) usable and then writing NOR flash memory address[15:8] value into register 0xF804 makes NAND flash memory data pin(Fdata[7:0]) output NOR flash memory address[15:8] value.

0 : No action.
1 : NOR flash high address Enable.

System Block Power Control (SYSTEM_BLOCK_POWER_CONTROL, 0xFF06) : Read / Write

7	6	5	4	3	2	1	0
RTC_PD	ADC_PD	USB0_PD	USB1_PD	NOR_PD	DRM_PD	MMC_PD	NAND_PD

Writing 1 to each bit disables clock supply into corresponding block and results in power down of corresponding block.

RTC_PD : This bit is used to RTC block power down.

0 : No action.
1 : RTC block power down.

ADC_PD : This bit is used to ADC block power down.

0 : No action.
1 : ADC block power down.

USB0_PD : This bit is used to USB0 block power down.

0 : No action.
1 : USB0 block power down.

USB1_PD : This bit is used to USB1 block power down.

0 : No action.
1 : USB1 block power down.

NOR_PD : This bit is used to NOR block power down.

0 : No action.
1 : NOR block power down.

DRM_PD : This bit is used to DRM block power down.

0 : No action.
1 : DRM block power down.

MMC_PD : This bit is used to MMC block power down.

0 : No action.
1 : MMC block power down.

NAND_PD : This bit is used to NAND flash interface block power down.

0 : No action.

1 : NAND flash interface block power down.

MP3 Block Power Control (MP3_BLOCK_POWER_CONTROL, 0xFF07) : Read / Write

7	6	5	4	3	2	1	0
Reserved			AUD_PD	COM_PD	ENC_PD	DEC_PD	WMA_PD

Writing 1 to each bit disables clock supply into corresponding block and results in power down of corresponding block.

AUD_PD : This bit is used to audio block power down.

0 : No action.
1 : Audio block power down.

COM_PD : This bit is used to common block power down.

0 : No action.
1 : Common block power down.

ENC_PD : This bit is used to MP3 encoder block power down.

0 : No action.
1 : MP3 encoder block power down.

DEC_PD : This bit is used to MP3 decoder block power down.

0 : No action.
1 : MP3 decoder block power down.

WMA_PD : This bit is used to WMA decoder block power down.

0 : No action.
1 : WMA decoder block power down.

Auto Power Down Control (DYNAMIC_POWER_CONTROL, 0xFF08) : Read / Write

7	6	5	4	3	2	1	0
Reserved	MP3_CON	SYS_CON	DMA_CON	NOR_CON	DRM_CON	MMC_CON	NAND_CON

Writing 1 to each bit disables clock supply into corresponding block and results in power down of corresponding block.

MP3_CON : This bit is used to MP3 block power control.

0 : No action.
1 : MP3 block power down.

SYS_CON : This bit is used to system block power control.

0 : No action.
1 : System block power down.

DMA_CON : This bit is used to DMA block power control.

0 : No action.
1 : DMA block power down.

NOR_CON : This bit is used to NOR flash block power control.

0 : No action.
1 : NOR flash block power down.

DRM_CON : This bit is used to DRM block power control.

0 : No action.
1 : DRM block power down.

MMC_CON : This bit is used to MMC block power control.

0 : No action.
1 : MMC block power down.

NAND_CON : This bit is used to NAND block power control.

0 : No action.
1 : NAND block power down.

9. USB Controller

9. USB Controller

NX5850 has the USB2.0 Full Speed function controller and the host controller which is compatible with OHCI Rev 1.0 specification. These two controllers can't be used at the same time and if one is in working, the other gets stopping.

9.1. USB2.0 Full Speed Function Controller

The USB Full-Speed Function Controller consists of three end-points and each of endpoints is capable of handling the bulk, the interrupt and the isochronous data transfers at the baud rate of 12 Mbps. The endpoints can be handled by the 8051 on NX5850. The automatic data retry, the data toggle and the power management functions such as suspend and resume are all supported.

The serial information is transmitted by the USB interface containing layers of communication protocols, the most basic of which are fields. The core fields include: sync, packet identifier, address, endpoint, frame number, data, and CRC fields. Fields are used to produce packets. Depending on the function of the packet, the different combination and a number of fields are used. Packet types include: token, start of frame, data, and handshake packets. Packets are then assembled into groups to produce frames. These frames or transactions fall into four groups: bulk, control, interrupt, and isochronous. Endpoint 0, by default, is used only to communicate control transactions to configure the USB controller after it is reset or hooked up (physically connected to an active USB host or hub). Endpoint 0's responsibilities include: connection, address assignment, endpoint configuration, bus enumeration, and disconnect. Endpoint 1 is used to perform bulk OUT data transactions and receiving data from the USB host; endpoint 2 is used to perform bulk IN data transactions and transmitting data to the USB host and vice versa.

The UDC(USB Device Controller) uses two separate FIFOs to buffer incoming and outgoing data to or from the host(128-entry x 8-bit for transmitting and another 128-entry x 8-bit for receiving). The FIFOs can be filled or emptied either by the DMA or the CPU, with service requests being signaled when either FIFO is Half-Full or Empty. Interrupts are signaled when the receive FIFO experiences an overrun and the transmit FIFO experiences an under-run. The control endpoint 0 has an additional 64-entry x 8-bit FIFO that can only be read or written by processor reads and writes.

The external pins dedicated to this interface are UDC+(USB DP, Pin 8) and UDC-(USB DM, Pin 9). The USB protocol uses differential signaling between the two pins for half-duplex data transmission. A 1.5-Kohm pull-up resistor is required to be connected to the USB cable D+ signal to pull the UDC+ pin high when not driven. This signifies the UDC is a high-speed, 12-Mbps device and provides the correct polarity for data transmission. Using differential signaling allows multiple states to be transmitted on the serial bus. These states are combined to transmit data as well as various bus conditions, including: idle, resume, start of packet, end of packet, disconnect, connect, and reset.

9.1.1. Operation

Following a reset of NX5850 or whenever the USB controller is attached to a USB bus, all endpoints are automatically configured by the core and the core is forced to use the USB default address of zero. The host then assigns NX5850 a unique address. At this point, the USB controller is under the host's control and responds to its commands that are transmitted to endpoint 0 using control transactions. Endpoint 1 is used to perform bulk OUT data transactions, receiving data from the USB host, and endpoint 2 bulk IN data transactions, transmitting data to the USB host.

9.1.2. Signaling Levels

USB uses differential signaling to encode data and to communicate various bus conditions. The USB specification refers to the J and K data states to differentiate between high- and low-speed transmissions. Because the UDC supports only 12-Mbps transmission, references are made only to actual data state 0 and actual data state 1.

Four distinct states are represented using differential data by decoding the polarity of the UDC+ and UDC- pins. Two of the four states are used to represent data. A one is represented when UDC+ is high and UDC- is low; a zero is represented when UDC+ is low and UDC- is high. The remaining two states and pairings of the four encoding are further decoded to represent the current state of the USB bus. Table 32. shows how seven different bus states are represented using differential signaling.

Hosts and hubs have pull-down resistors on both the D+ and D- lines. When a device is not attached to the cable, the pull-down resistors cause D+ and D- to be pulled down below the single-ended low threshold of the host or hub. This creates a state called single-ended zero (SE0).

Table 21. USB bus signal level description in according to current status

Bus State	UDC+ / UDC- Signal Levels
Idle	UDC+ high, UDC- low (same as 1)
Resume	UDC+ low, UDC- high (same as 0)
Start of Packet	Transition from Idle to Resume
End of Packet	UDC+ and UDC- low for 2-bit times followed by an idle for 2-bit time.
Disconnect	UDC+ and UDC- below single-ended low threshold for more than 2.5 usec. (Disconnect is a static bus condition that result no devices is plugged-into a hub port)
Connect	UDC+ OR UDC- high for more than 2.5 usec.
Reset	UDC+ AND UDC- low for more than 2.5 usec. (Reset is driven by the host controller and sensed by a device controller.)

A disconnect is detected by the host when an SE0 persists for more than 2.5 usec (30-bit times). When the UDC is connected to the USB cable, the pull-up resistor on the UDC+ pin causes D+ to be pulled above the single-ended high threshold level. After 2.5 usec elapse, the host detects a connect.

After this point, the bus is in the idle state because UDC+ is high and UDC- is low. A start of packet is signaled by transitioning the bus from the idle to the resume state (a 1 to 0 transition). The beginning of each USB packet begins with a sync field, which starts with the 1-to-0 transition.

After the packet data has been transferred, an end of packet is signaled by pulling both UDC+ and UDC- low for 2-bit times, followed by an idle for 1-bit time. If the idle persists for more than 3 msec, the UDC enters suspend mode and it is placed in low-power mode. The UDC can be awakened from the suspend state by the host by switching the bus to the resume state via normal bus activity, or by signaling a reset. Under normal operating conditions, the host ensures that devices do not enter the suspend state by periodically signaling an end of packet (EOP).

9.1.3. Bit Encoding

The USB uses non-return-to-zero inverted (NRZI) to encode individual bits. Both the clock and the data are encoded and transmitted within the same signal. Instead of representing data by controlling the state of the signal, transitions are used. A zero is represented by a transition, and a one is represented by no transition (this produces the data).
Each time a zero occurs, the receiver logic synchronized the baud clock to the incoming data (this produces the clock). To ensure the receiver is periodically synchronized, any time six consecutive ones are detected in the serial bit stream, a zero is automatically inserted by the transmitter. This procedure is known as "Bit stuffing". The receiver logic, in turn, automatically detects stuffed bits and removes them from the incoming data. Bit stuffing causes a transition on the incoming signal at least once every seven bit-times to guarantee baud clock lock. Bit stuffing is enabled for an entire packet beginning when the start of packet is detected until the end of packet is detected (enabled during the sync field all the way through the CRC field).
Figure 11. shows the NRZI encoding of the data byte 0b1101 0010.

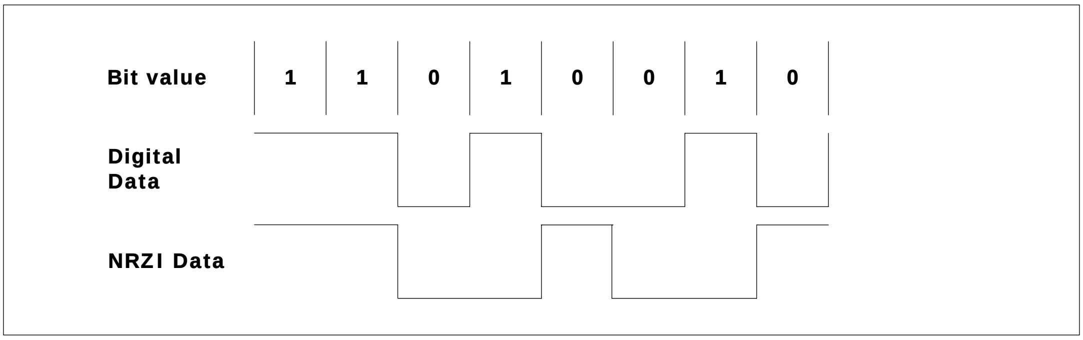

Figure 8. NRZI bit encoding example

9.1.4. Field Formats

Individual bits are assembled into groups called fields. Fields are used to construct packets and packets are used to construct frames or transactions. The seven USB field types include: sync, packet identifier, address, endpoint, frame number, data, and CRC fields.

Sync is preceded by the idle state on the USB bus and is always the first field of every packet. The first bit of a sync field signals the start of packet (SOP) to the UDC or host. Sync is 8 bits wide and consists of seven zeros followed by a one (0x80).

The packet identifier (PID) is 1 byte wide and always follows the sync field. The first 4 bits contain an encoded value that represents packet type (token, data, handshake and special), packet format, and type of error detection. The last four bits contain a check field that ensures the PID is transmitted without errors. The check field is generated by performing a ones complement of the PID. The UDC automatically XORs the PID and check field and takes the appropriate action (as prescribed by the USB standard) if the result does not contain all ones, indicating an error has occurred in transmission.

The UDC's three endpoints are accessed using the address and endpoint fields. The address field contains 7 bits and permits 128 unique devices to be placed on the USB. After NX5850 is reset, or a reset is signaled via the USB bus, the UDC (and all other 127 possible devices) is assigned the default address of zero. The host is then responsible for assigning unique addresses for each device on the bus. This is performed in the enumeration process one device at a time. Once the host assigns the UDC an address, it responds only to transactions addressed to it. The address field is transmitted in every packet and follows the PID field.

When the UDC detects that a packet is addressed to it, the endpoint field is used to determine which of the UDC's three endpoints are being addressed. The endpoint field is 4 bits. However, only the encoding for endpoints 0 through 2 is allowed. The endpoint field follows the address field. Table 20. shows the valid values for the endpoint field. The frame number is an 11-bit field that is incremented by the host each time a frame is transmitted. When it reaches its maximum value of 2047 (0x7FF), it rolls over. It is transmitted in the start of frame (SOF) packet, which is output by the host in 1 msec intervals. The frame number field is used only by device controllers to control isochronous transfers, and therefore, does not affect the UDC. Data fields are used to transmit the bulk data between the host and the UDC. A data field is made up of 0 to 1023 bytes. Each byte is transmitted LSB first.

Table 22. Endpoint field addressing

Endpoint Field Value	UDC Endpoint Selected
0000	Endpoint 0
0001	Endpoint 1
0010	Endpoint 2
0011	Invalid
01xx	Invalid
10xx	Invalid
11xx	Invalid

Cyclic redundancy check fields are used to detect errors introduced during transmission of token and data packets, and is applied to all the fields in the packet except the PID field (recall the PID contains its own 4-bit ones complement check field for error detection). Token packets use a 5-bit CRC (x 5 + x 2 + 1) and data packets use a 16-bit CRC (x 16 + x 15 + x 2 + 1). For both CRCs, the checker is reset to all ones at the start of each packet.

9.1.5. Packet Formats

USB supports four packet types: token, data, handshake, and special. A token packet is placed at the beginning of a frame and is used to identify OUT, IN, SOF, and SETUP transactions. OUT and IN frames are used to transfer data, SOF packets are used to time isochronous transactions, and SETUP packets are used for control transfers to configure endpoints. A token packet consists of a sync, a PID, an address, an endpoint, and a CRC5 field. For OUT and SETUP transactions, the address and endpoint fields are used to select which UDC endpoint is to receive the data, and for an IN transaction, which endpoint must transmit data.

Table 23. IN, OUT, and SETUP token packet format

8 bits	8 bits	7 bits	4 bits	5 bits
Sync	PID	Address	Endpoint	CRC5

A start of frame (SOF) is a special type of token packet that is issued by the host once every 1 msec. SOF packets consist of a sync, a PID, a frame number (which is incremented after each frame is transmitted), and a CRC5 field, as shown in Table 22. Even though the UDC on the Scorpio does not make use of the frame number field, the presence of SOF packets every 1ms will prevent the UDC from going into suspend mode.

Table 24. SOF token packet format

8 bits	8 bits	11 bits	5 bits
Sync	PID	Frame Number	CRC5

Data packets follow token packets, and are used to transmit data between the host and UDC. There are two types of data packets as specified by the PID: DATA0 and DATA1. These two types are used to provide a mechanism to guarantee data sequence synchronization between the transmitter and receiver across multiple transactions. During the handshake phase, both communicate and agree which data token type to transmit first. For each subsequent packet transmitted, the data packet type is toggled (DATA0, DATA1, DATA0, and so on). A data packet consists of a sync, a PID, from 0 - 256 bytes of data, and a CRC16 field, as shown in Table 23.

Table 25. Data packet format

8 bits	8 bits	0 – 256 bytes	16 bits
Sync	PID	Data	CRC16

Handshake packets consist of only a sync and a PID. Handshake packets do not contain a CRC because the PID contains its own check field. They are used to report data transaction status, including whether data was successfully received, flow control, and stall conditions. Only transactions that support flow control can return handshakes. The three types of handshake packets are: ACK, NAK, and STALL. ACK indicates that a data packet was received without bit stuffing, CRC, or PID check errors. NAK indicates that the UDC was unable to accept data from the host or it has no data to transmit. NAK is also used by endpoint 1 to indicate no interrupts are pending. STALL indicates that the UDC is unable to transmit or receive data, and requires host intervention to clear the stall condition. Bit stuffing, CRC, and PID errors are signaled by the receiving unit by omitting a handshake packet. Table 24. shows the format of a handshake packet.

Table 26. Handshake packet format

8 bits	8 bits
Sync	PID

9.1.6. Transaction Formats

Packets are assembled into groups to form transactions. Four different transaction formats are used in the USB protocol. Each is specific to a particular endpoint type: bulk, control, interrupt, and isochronous. Note that isochronous and interrupt transactions are not supported by the UDC and are not described in this section. Endpoint 0, by default, is a control endpoint and receives only control transactions; both endpoints 1 and 2 use bulk transactions. Note that all USB bus transactions are initiated by the host controller and that transmission takes place between the host and UDC one direction at a time (half-duplex).

Bulk transactions guarantee error-free transmission of data between the host and UDC by using packet error detection and retry. The three packet types used to construct bulk transactions are: token, data, and handshake. The eight possible types of bulk transactions based on data direction, error, and stall conditions are shown in Table 25. Note that packets sent by the UDC to the host are highlighted in boldface type, and packets sent by the host to the UDC are not.

Table 27. Bulk transaction formats (Packets from UDC to host are boldface)

Action	Token Packet	Data Packet	Handshake Packet
Host successfully received data from UDC	PID	Data0/Data1	Ack
UDC temporarily unable to transmit data	IN	None	NAck
UDC endpoint needs host intervention	IN	None	Stall
Host detected PID, CRC, or bit stuff error	IN	Data0/Data1	None
UDC successfully received data from host	Out	Data0/Data1	**Ack**
UDC temporarily unable to receive data	Out	Data0/Data1	**NAck**
UDC endpoint needs host intervention	Out	Data0/Data1	**Stall**
UDC detected PID, CRC, or bit stuff error	Out	Data0/Data1	None

Control transactions are used by the host to configure endpoints and query their status. Like bulk transactions, control transactions begin with a setup packet, followed by an optional data packet, then a handshake packet. Note that control transactions, by default, use DATA0 type transfers.

Table 39. shows the four possible types of control transactions. Note that packets sent by the UDC to the host are highlighted in boldface type, and packets sent by the host to the UDC are not.

Table 28. Control transaction formats

Action	Token Packet	Data Packet	Handshake Packet
UDC successfully received control from host	SETUP	Data0	**Ack**
UDC temporarily unable to receive data	SETUP	Data0	**NAck**
UDC endpoint needs host intervention	SETUP	Data0	**Stall**
UDC detected PID, CRC, or bit stuff error	SETUP	Data0	**None**

Control transfers are assembled by the host by first sending a control transaction to tell the UDC what type of control transfer is taking place (control read or control write), followed by two or more bulk data transactions. The control transaction, by default, uses a DATA0 transfer, and each subsequent bulk data transaction toggles between DATA1 and DATA0 transfers. For a control write to an endpoint, OUT transactions are used. For control reads, IN transactions are used. The transfer direction of the last bulk data transaction is reversed. It is used to report status and functions as a handshake. The last bulk data transaction always uses a DATA1 transfer by default (even if the previous bulk transaction used DATA1). For a control write, the last transaction is an IN from the UDC to the host, and for a control read, the last transaction is an OUT from the host to the UDC.

9.1.7. UDC Device Requests

The UDC's control, status, and data registers are used only to control and monitor the transmit and receive FIFOs for endpoints 1 and 2. All other UDC configuration and status reporting is controlled by the host via the USB bus using device requests that are sent as control transactions to endpoint 0. Each setup packet to endpoint 0 is 8 bytes long and specifies:

- Data transfer direction: host to device, device to host
- Data transfer type: standard, class, and vendor
- Data recipient: device, interface, endpoint, other
- Number of bytes to transfer
- Index or offset
- Value: used to pass a variable-sized data parameter
- Device request

Table 27. shows a summary of all device requests. Users should refer to the Universal Serial Bus Specification Revision 1.0 for a full description of host device requests.

Table 29. Host device request summary

Request	Name
SET_FEATURE	Used to enable a specific feature such as device remote wake-up and endpoint stalls.
CLEAR_FEATURE	Used to clear or disable a specific feature.
SET_CONFIGURATION	Configures the UDC for operation. Used following a reset of the Scorpio or after a reset has been signaled via the USB bus.
GET_CONFIGURATION	Returns the current UDC configuration to the host.
SET_DESCRIPTOR	Used to set existing descriptors or add new descriptors. Existing descriptors include: device, configuration, string, interface, and endpoint.
GET_DESCRIPTOR	Returns the specified descriptor if it exists.
SET_INTERFACE	Used to select an alternate setting for the UDC's interface.
GET_INTERFACE	Returns the selected alternate setting for the specified interface.
GET_STATUS	Returns the UDC's status including: remote wake-up, self-powered, data direction, endpoint number, and stall status.
SET_ADDRESS	Sets the UDC's 7-bit address value for all future device accesses.
SYNCH_FRAME	Used to set and then report an endpoint's synchronization frame.

9.1.8. USB Device Application

The USB Device specification will be added later.

9.2. USB1.1 Host Controller

The USB host controller specification will be added later.

9.3. USB Control Register

Table 30. USB Control Register Map (P2 = 0xFD)

Function	Address (Hex)	Type	Reset	Description
USB_DMA_RESET	0x40	R/W	-	USB DMA reset control
USB_PACKET_LOOP_CNT	0x41	R/W	0x00	USB packet loop count control
USB_MAX_PACKE_LEN	0x42 ~0x43	R/W	0x00	USB Max packet length
USB_DMA_CONTROL	0x44	R/W	0x00	USB / DMA control
USB_PACKET_DATA_CNT	0x46 ~0x47	RO	-	USB packet data count transferred
USB_LAST_PACKET_LEN	0x48 ~0x49	R/W	0x00	USB last packet length
USB_PIN_INTERRUPT_ENABLE	0x4C	R/W	0x00	DP, DM pin interrupt enable
USB_PIN_INTERRUPT_STATUS	0x4D	RO	-	DP, DM pin interrupt status

USB_DMA Reset Control (USB_DMA_RESET, 0xFD40) : Read/Write

7	6	5	4	3	2	1	0
Reserved							USB_RST

Writing 1 makes USB_DMA reset.

USB_RST : This bit used to USB_DMA reset.

0 : No action.
1 : USB_DMA reset.

USB Packet Loop Count (USB_PACKET_LOOP_CNT, 0xFD41) : Read/Write

7	6	5	4	3	2	1	0
LP_CNT							

Including last packet that indicated by register (0xFD48-0xFD49) as 1 packet count, this register determines the count of packet to be transferred, packet length is indicated by register(0xFD42-0xFD43).

For example,

0xFD41

(Data_length_to_be_transferred – 1) / packet_length + 1

0xFD42-0xFD43

packet_length – 1

0xFD48-0xFD49

(Data_length_to_be_transferred – 1) % packet_length

USB Max Packet Length (USB_MAX_PACKET_LEN, 0xFD42) : Read / Write

7	6	5	4	3	2	1	0
MAX_PAK							

This register is used to read / write the lower 8-bit of 10-bit of the USB_MAX_PACKET_LEN data.

one less than packet length to be transferred.

USB Max Packet Count (USB_MAX_PACKET_CNT, 0xFD43) : Read / Write

7	6	5	4	3	2	1	0
Reserved						MAX_PAK	

This register is used to read / write the upper 2-bit of 10-bit of the USB_MAX_PACKET_LEN data.

USB And DMA Control (USB_DMA_CONTROL, 0xFD44) : Read / Write

7	6	5	4	3	2	1	0
Reserved			MCU_DMA	USB_DMA	END_POT		

MCU_DMA : This bit is used to DMA or MCU mode selection.

0 : DMA operation starts on USB dma request
1 : DMA operation starts without USB dma request

USB_DMA : This bit is used to from USB to DMA mode selection.

0 : From DMA buffer to USB transfer.
1 : From USB to DMA buffer transfer.

END_POT :

Determines end point used by DMA operation

USB Packet data count (USB_PACKET_DATA_CNT, 0xFD46) : Read Only

7	6	5	4	3	2	1	0
DAT_CNT							

This register shows USB packet data count transferring currently.

USB Packet data count (USB_PACKET_DATA_CNT, 0xFD47) : Read Only

7	6	5	4	3	2	1	0
Reserved						DAT_CNT	

USB last packet length (USB_LAST_PACKET_LEN, 0xFD48) : Read / Write

7	6	5	4	3	2	1	0
LST_CNT							

Write last packet length in byte unit to this register. Indicates one less than length.

USB last packet count (USB_LAST_PACKET_CNT, 0xFD49) : Read / Write

7	6	5	4	3	2	1	0
Reserved						LST_CNT	

USB DP/DM Pin Enable (USB_PIN_INTERRUPT_ENABLE, 0xFD4C) : Read / Write

7	6	5	4	3	2	1	0
Reserved				DP_RI	DP_FA	DM_RI	DM_FA

DP_RI : This bit is used to rise interrupt enable of the USB DP pin.

0 : DP rise interrupt disable.
1 : DP rise interrupt enable.

DP_FA : This bit is used to fall interrupt enable of the USB DP pin.

0 : DP fall interrupt disable.
1 : DP fall interrupt enable.

DM_RI : This bit is used to rise interrupt enable of the USB DM pin.

0 : DM rise interrupt disable.
1 : DM rise interrupt enable.

DM_FA : This bit is used to fall interrupt enable of the USB DM pin.

0 : DM fall interrupt disable.
1 : DM fall interrupt enable.

USB DP/DM Pin Status (USB_PIN_INTERRUPT_STATUS, 0xFD4D) : Read Only

7	6	5	4	3	2	1	0
Reserved				DP_RI_ST	DP_FA_ST	DM_RI_ST	DM_FA_ST

1 means corresponding interrupt is generated.

DP_RI_ST : This bit indicates rise interrupt status of the USB DP pin.

0 : No action.
1 : DP rise interrupt generate.

DP_FA_ST : This bit indicates fall interrupt status of the USB DP pin.

0 : No action.
1 : DP fall interrupt generate.

DM_RI_ST : This bit indicates rise interrupt status of the USB DM pin.

0 : No action.
1 : DM rise interrupt generate.

DM_FA_ST : This bit indicates fall interrupt status of the USB DM pin.

0 : No action.
1 : DM fall interrupt generate.

10. DMA Controller

10. DMA Controller

The DMA controller (DMAC) is a core function block to transfer data between the internal system memory (64KB SRAM) and external storage devices (MMC / SD / Nor FLASH / Nand Flash / USB) in high speed. Though 8051 has a lower power consumption, small area of the hardware and the general purpose, it has the low data band width. To compensate the weak points of the 8051, NX5850 has the DMA structure to perform the best functions with the 8051.

10.1. DMA Operation

The DMAC needs to have some basic controls. First, set registers to define DMA buffer (a part of Internal 64Kbyte SRAM) for the MCU operation. For this, define the base address of the buffer (DMA_BUFn_BASE_ADDR_HI/LO), "read/write start address" and "read/write end address" of the buffer. "read/write start/end address" are offset address from base address and the buffer size is from "read/write start address" to "read/write end address". The DMAC generates an interrupt of the Full/Empty status for MCU to get to know (Figure 15), or set the register to read or write data in the polling method. For example, to transfer the block data (512Byte), the source and destination of the DMA buffer address need to have the initial values. The register address '0xfe00~0xfe3f are used for DMA.

DMA has four buffers (Buffer0, Buffer1, Buffer2, and Buffer3), and the address and the size of some area on the system memory are selected by a system configuration. To communicate to MCU, two methods are supported. They are the Interrupt and Polling method.

The interrupt method has the Full-Interrupt with using MCU and Half-Interrupt without using MCU. The DMA buffer size is decided by setting the "BUFn base address + BUFn write/read address" as the DMA Buffer Start Address and the "BUFn base address + BUFn write/read end address" as the Buffer End Address of DMA register block.

There are two ways to see if an interrupt occurs. One is that making a environment for occurring an interrupt as giving the data transfer size for the Full and the Empty condition. The other is that keeping reading the status of "buffer Status" register in the polling method.

When the two different devices communicate with same DMA buffer, one is in reading and the other is in writing, the status keeps in a cycle like as 'Empty->Full->Empty'.

Followings give an explanation for the DMAC control based on above.

Program 1. DMA Operation.

```
B0_SetAddr(0xA000);         // set DMA Buffer0 Address to 0xA000
B0_Reset();                 // initialize DMA Buffer0. ReadPtr->0, WritePtr->0,
                            // Buffer0_Status->Empty
B0_WriteEnd(511);           // set Write End Pointer to 511. Data are 512Byte
write_XDATA(DMA_FLASH_MODE, 0x04);  //0xFE03, give access between flash controller and DMAC

// perform a write command between flash controller and DMA buffer0
// WritePtr has the same value to ReadPtr zero, but perform write operation because the register
// 'B0_BSTATUS (Buffer0 Status)' is Empty.
// WritePtr increases, return into zero if WritePtr has '511' of Write End Pointer.
// B0_BSTATUS goes to Full when WritePtr has the same value to ReadPtr zero and gives the Full
// status to the flash controller.

B0_ReadEnd(511); // set Read End Pointer to '511'. Data is 512Byte
write_XDATA(DMA_MP3_MODE, 0x04); //0x7E05, gives access between MP3 Codec and DMAC

// perform a read command between MP3 CODEC and DMA Buffer. WritePtr has the same value to
// ReadPtr zero, but performs read operation because B0_BSTATUS is Full. ReadPtr increases, return
// into zero if ReadPtr has '511' of Read End Pointer. B0_BSTATUS goes to Empty when ReadPtr has
// the same value to WritePtr zero, and gives Empty status to the MP3 CODEC.
B0_SetAddr(0xA000);         //set DMA Buffer0 Address to 0xA000
B0_Reset();                 // initialize DMA Buffer0, ReadPtr->0 and WritePtr->0
                            // B0_BSTATUS->Empty
B0_WriteEnd(511);           //set Write End Pointer to '511' (Data size : 512Byte)
Write_XDATA(DMA_FLASH_MODE,0x04);
//0x7E03 give access to both DMA and NAND flash Controller

// The write order is performed between the flash controller and the DMA buffer0. When
B0_BSTATUS
// is Empty and that WritePtr and ReadPtr have zero, the write operation is available. Increasing
// WritePtr, when Write End Pointer is '511', return to zero. When WritePtr and ReadPtr are zero,
// B0_BSTATUS goes to Full and notice the status Full to the flash controller.
B0_ReadEnd(511);                     // set Read End Point to '511' (data size : 512Byte)
Write_XDATA(DMA_MP3_MODE, 0x04);     // 0xFE05, give access to both MP3 CODEC and
DMAC

// Read order is performed between MP3 CODEC and the DMA buffer0
// When B0_BSTATUS is Full and that WritePtr and ReadPtr have zero, the write operation is
// available.
// Increasing ReadPtr, when Read End Pointer is '511', return to zero
// When ReadPtr and WritePtr are zero, B0_BSTATUS goes to Empty and notice the Empty status to
// MP3 CODEC.
```

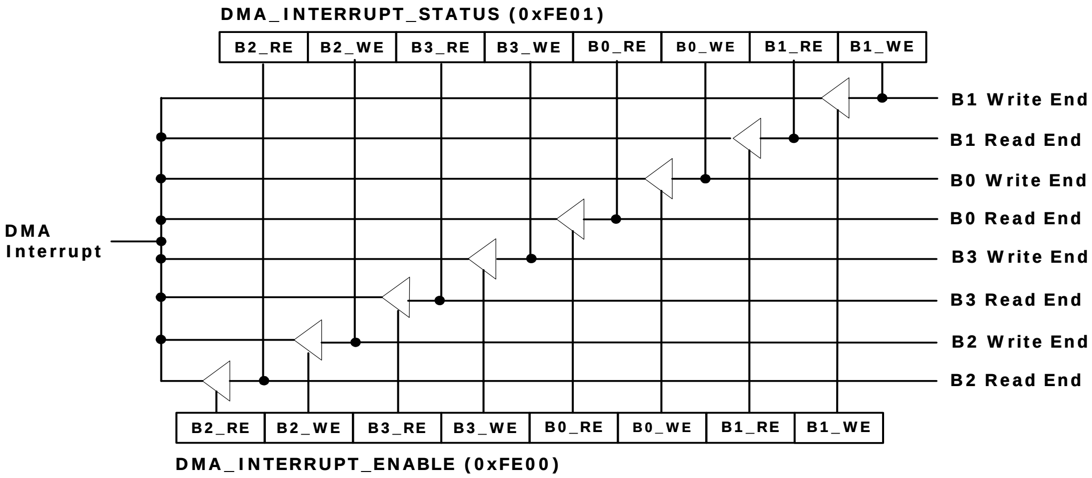

Figure 9. The Block Diagram for generating the DMA interrupts

10.1.1. Full Interrupt Communication of DMA Controller

For the Full-Interrupt communication of the DMA, use the DMA Interrupt and the MCU Interrupt. To use the DMA Interrupt, set MCU Master interrupt enable, MCU port3.2 interrupt enable, the register 'BLOCK_INTERRUPT_ENABLE(0xff00[0])' and 'DMA_INTERRUPT_ENABLE(0xfe00)'. To set MCU Master interrupt enable, MCU port3.2 interrupt enable see 8051 MCU data sheet. Refer to "Figure 7. External Interrupt Mask to MCU.

The following is a DMA test program. The function 'DMA_test()' shows an operation for the interrupt generation. After setting the dma interrupt enable(0xfe00), the 8051 Interrupt and the DMAC communication environment, the flash controller orders to write the data to the DMA buffer0 and then the data 512Byte are written into the DMA buffer0. When the 'Buffer Status' goes to Full, the DMA interrupt is occurred by setting the dma interrupt enable(0xfe00), and the interrupt signal out of the 8051 interrupt pin is occurred by setting the 'interrupt enable(0xff00[0]).

The function 'EX0_int()' is performed after the 8051 takes the interrupt under setting the interrupt environment of the 8051. Reading the 'MCU interrupt status(0xff01[0]) in the EX0_int() get to know occurring a interrupt, and if the interrupt is occurred, call the function 'DMA_isr()'. The DMA buffer0 puts Full out when the data in the NAND flash are transferred up to the DMA buffer0. And reading the register 'dma interrupt status(0xFE01)' in the DMA_isr(), get to know occurring the interrupt.
The interrupt signal stops when read the MCU interrupt status(0xff01[0])and the 'dma interrupt status(0xFE01)'. It results in blocking access the flash control to the DMAC. When giving access the USB to the DMAC, the B0_BSTATUS is Full and the USB takes the data by reading. When the B0_BSTATUS goes to Empty, the interrupt process is performed one more again.

At this time, the interrupt is cleared after reading the register 'MCU interrupt status(0xff01[0])' and 'dma interrupt status(0xFE01)', and then, make the DMA_FLASH_MODE[2] disabled and the DMA_USB_MODE[2] enabled. The DMA buffer0 has the Full status as the result of the transfer for the flash controller communication, and then the DMA buffer0 has Empty as performing that the USB controller reads the data from the DMA buffer0.

10.1.2. A Example for the Full-Interrupt Routine of the DMA Controller

Program 2. The full-interrupt routine of the DMA controller.

```
void DMA_isr()
{
    data uchar channel;
    channel = read_XDATA(DMA_INT_STATUS);

    switch(channel & 0x70)
    {
        case 0x10:                                    //USB
            write_XDATA(DMA_USB_MODE, 0x00);          //0xFE02
        break;
        case 0x20:                                    //Flash Controller
            write_XDATA(DMA_FLASH_MODE, 0x00);        //0xFE03
            write_XDATA(DMA_USB_MODE, 0x04);          //USB to DMAC channel open
        break;
    }
}


void EX0_int(void) interrupt 0
{
    data uchar i;
    i = read_XDATA(PERI_INT_ST);
    i &= Vperi_int_en;
    if(i & IR_DMA)
        DMA_isr();
}


//------------------------------------------------------------------------------------------------------
// DMA_test : This is a function that 512byte data from Flash device read into DMA buffer0,
// and transfer the data to USB block.
//------------------------------------------------------------------------------------------------------

void DMA_test()
{
    EX0 = 1;       // set to use the Interrupt generated at P3.2 Interrupt pin of 8051.
    Vperi_int_en = 0x40;  // only when DMA Interrupt is generated, the interrupt put on the
                          // pin of the 8051.

    write_XDATA(PERI_INT_EN, Vperi_int_en);
    write_XDATA(DMA_INT_EN, 0x0f);            // set Interrupt Enable in DMA block

    B0_SetAddr(0xA000);   // set DMA Buffer0 Address to 0xA000.
    B0_Reset();           // initialize DMA Buffer0.
                          // ReadPtr->0, WritePtr->0, Buffer0_Status->Empty.
    B0_ReadEnd(511);      // set Read End Pointer to 511.
                          // USB Block Data is 512Byte.
    B0_WriteEnd(511);     // set Write End Pointer to 511.
                          // Flash Block Data is 512Byte
    write_XDATA(DMA_FLASH_MODE, 0x04);   //0xFE03, flash controller to DMAC channel open

// The flash controller orders to write data into DMA buffer. Refer to the access to flash controller.
// It has a assume setting to read data of DMA Buffer through the USB
}
```

10.1.3. Half-Interrupt communication of DMA Controller

The DMAC Half-Interrupt method uses the DMA interrupt but not uses the MCU interrupt. The different point to the DMAC Full-Interrupt is that the EX0 is low.
The following program shows how interrupts are occurred. After the DMA_INTERRUPT_ENABLE and the BLOCK_INTERRUPT_ENABLE are set and the 8051 interrupt is disabled, when the flash controller orders to write into the DMA buffer0 by setting the DMA transfer, 512 Byte are written into the DMA buffer0. At this time, the DMA_BUF0_STATUS is Full and the DMA Interrupt is occurred by setting the DMA_INT_EN, and the interrupt puts at 8051 Interrupt pin by setting the BLOCK_INTERRUPT_ENABLE. The function 'EX0_int()' is not performed because the 8051 doesn't get any interrupt under the 8051 interrupt is disabled.
The interrupt is cleared (return to high) by reading the register 'BLOCK_INTERRUPT_STATUS' and 'DMA_INTERRUPT_STATUS (0xFE01)' after the P3.2 interrupt pin turns into low for executing the sentence 'while(INTOPORT)', what the P3.2 interrupt pin turns into low means that the interrupt is occurred by the DMA communication.

The DMA_FLASH_MODE[2] is disabled when finishing to transfer the data of the flash device to the DMA buffer. The 8051 calls the function 'EX0_int()' to handle the interrupt when EX0 is high under the register 'BLOCK_INTERRUPT_STATUS' and 'DMA_INTERRUPT_STATUS(0xFE01)' is not read.

10.1.4. A Example for the Half-Interrupt Routine of the DMA Controller

Program 3. The half-interrupt routine of the DMA controller.

```
sbit INT0PO = P3^2;
EX0 = 0;                    //don't look at the interrupt pin,P3.2, of 8051.
Vperi_int_en = 0x40;        //give the interrupt pin of 8051 when the DMA Interrupt is only
                            //occurred.
write_XDATA(PERI_INT_EN, Vperi_int_en);
write_XDATA(DMA_INT_EN, 0x0f);   //set the interrupt enable in the DMA Block.
B0_SetAddr(0xA000);              //set the DMA Buffer0 Address to 0xA000.
B0_Reset();                      //initialize DMA Buffer0.
                                //ReadPtr->0, WritePtr->0, B0S->Empty
B0_WriteEnd(511);               //set the Write End Pointer to 511.
                                          // Data size that Flash Block wants is 512Byte
write_XDATA(DMA_FLASH_MODE, 0x04);    //0xFE03, give access to Flash DMA.
                   :
/* Flash controller give the command to write to DMA buffer. Refer to the giving access to Flash
Controller */

while(INT0PORT);                          // detect the Interrupt Signal
buf = read_XDATA(DMA_INT_STATUS);         // clear DMA interrupt
buf = read_XDATA(PERI_INT_ST);            // clear 8051 interrupt pin
write_XDATA(DMA_FLASH_MODE, 0x00);        // 0xFE03, flash controller to DMAC channel close
EX0 = 1;
```

10.1.5. The Polling Communication of the DMA Controller

The Polling DMA doesn't use the DMA interrupt. It has no effect with setting the BLOCK_INTERRUPT_ENABLE and the MCU interrupt. The different point of above two ways is that the register 'DMA_INTERRUPT_ENABLE(0xFE00)' has 0x00. The following example program shows that keeps reading the status of the DMA_BUF_STATUS(0xFE06) when the DMA_BUF0_STATUS is Full. If the DMA_BUF0_STATUS is Full, the process for transferring data of 512 Byte from the flash device is done as closing the channel between the flash controller and the DMA, and the interrupt is not occurred because the register 'DMA_INTERRUPT_ENABLE(0xFE00)' is 0x00.

10.1.6. A Example for the Polling Routine of the DMA Controller

Program 4. The polling routine of the DMA controller.

```
uchar buf;

write_XDATA(DMA_INT_EN, 0x00);   // set the interrupt enable in DMA Block
B0_SetAddr(0xA000);              // set DMA Buffer0 Address to 0xA000
B0_Reset();                      // initialize DMA Buffer0
                                 // ReadPtr->0, WritePtr->0, Buffer0_Status->Empty
B0_WriteEnd(511);                // set Write End Pointer to511
                                 // Flash Block Data is 512Byte
write_XDATA(DMA_FLASH_MODE, 0x04); //0xFE03, flash controller to DMAC channel open

// The flash controller gives an order writing into the DMA buffer. Refer to 'give access to the flash
// controller'.

do{
    buf = read_XDATA(BSTATUS);          //0xFE06
} while((buf&0x08) == 0);               // check whether the DMA Buffer0 is Full.
write_XDATA(DMA_FLASH_MODE, 0x00);      //0xFE03, flash controller to DMAC channel close
```

10.2. DMA Control Register

Table 31. DMA Control Register Map (P2 = 0xFE)

Function	Address (Hex)	Type	Reset	Description
DMA_INTERRUPT_ENABLE	0x00	R/W	0x00	DMA buffer interrupt enable
DMA_INTERRUPT_STATUS	0x01	R/W	0x00	DMA buffer interrupt status
WR_BUF0_FROM_SOURCE_SEL	0x02	R/W	0x00	Write buffer0 from source selected
RD_BUF0_TO_DEST_SEL	0x03	R/W	0x00	Read buffer0 to destination selected
WR_BUF1_FROM_SOURCE_SEL	0x04	R/W	0x00	Write buffer1 from source selected
RD_BUF1_TO_DEST_SEL	0x05	R/W	0x00	Read buffer1 to destination selected
DMA_BUF_STATUS	0x06	RO	0x55	Indicates buffer full or empty
DMA_BUF_RESET	0x07	WO	-	DMA buffer reset
DMA_BUF0_STATUS	0x08	R/W	0x01	Indicates buffer0 full or empty
BUF0_BASE_ADDR_LO	0x0A	R/W	0x00	Buffer0 base address low
BUF0_BASE ADDR_HI	0x0B	R/W	0x00	Buffer0 base address high
BUF0_WR_END_ADDR_LO	0x0C	R/W	0x00	Buffer0 write end address low
BUF0_WR_END_ADDR_HI	0x0D	R/W	0x00	Buffer0 write end address high
BUF0_RD_END_ADDR_LO	0x0E	R/W	0x00	Buffer0 read end address low
BUF0_RD_END_ADDR_HI	0x0F	R/W	0x00	Buffer0 read end address high
BUF0_WR_ADDR_LO	0x10	R/W	0x00	Buffer0 write offset address low
BUF0_WR_ADDR_HI	0x11	R/W	0x00	Buffer0 write offset address high

BUF0_RD_ADDR_LO	0x12	R/W	0x00	Buffer0 read offset address low
BUF0_RD_ADDR_HI	0x13	R/W	0x00	Buffer0 read offset address high
DMA_BUF1_STATUS	0x14	R/W	0x01	Indicates buffer1 full or empty
BUF1_BASE_ADDR_LO	0x16	R/W	0x00	Buffer1 base address low
BUF1_BASE ADDR_HI	0x17	R/W	0x00	Buffer1 base address high
BUF1_WR_END_ADDR_LO	0x18	R/W	0x00	Buffer1 write end address low
BUF1_WR_END_ADDR_HI	0x19	R/W	0x00	Buffer1 write end address high
BUF1_RD_END_ADDR_LO	0x1A	R/W	0x00	Buffer1 read end address low
BUF1_RD_END_ADDR_HI	0x1B	R/W	0x00	Buffer1 read end address high
BUF1_WR_ADDR_LO	0x1C	R/W	0x00	Buffer1 write offset address low
BUF1_WR_ADDR_HI	0x1D	R/W	0x00	Buffer1 write offset address high
BUF1_RD_ADDR_LO	0x1E	R/W	0x00	Buffer1 read offset address low
BUF1_RD_ADDR_HI	0x1F	R/W	0x00	Buffer1 read offset address high
WR_BUF2_FROM_SOURCE_SEL	0x22	R/W	0x00	Write buffer2 from source selected
RD_BUF2_TO_DEST_SEL	0x23	R/W	0x00	Read buffer2 to destination selected
WR_BUF3_FROM_SOURCE_SEL	0x24	R/W	0x00	Write buffer3 to destination selected
RD_BUF3_TO_DEST_SEL	0x25	R/W	0x00	Read buffer3 to destination selected
DMA_BUF2_STATUS	0x28	R/W	0x01	Indicates buffer2 full or empty
BUF2_BASE_ADDR_LO	0x2A	R/W	0x00	Buffer2 base address low
BUF2_BASE ADDR_HI	0x2B	R/W	0x00	Buffer2 base address high
BUF2_WR_END_ADDR_LO	0x2C	R/W	0x00	Buffer2 write end address low
BUF2_WR_END_ADDR_HI	0x2D	R/W	0x00	Buffer2 write end address high
BUF2_RD_END_ADDR_LO	0x2E	R/W	0x00	Buffer2 read end address low
BUF2_RD_END_ADDR_HI	0x2F	R/W	0x00	Buffer2 read end address high
BUF2_WR_ADDR_LO	0x30	R/W	0x00	Buffer2 write offset address low
BUF2_WR_ADDR_HI	0x31	R/W	0x00	Buffer2 write offset address high
BUF2_RD_ADDR_LO	0x32	R/W	0x00	Buffer2 read offset address low
BUF2_RD_ADDR_HI	0x33	R/W	0x00	Buffer2 read offset address high
DMA_BUF3_STATUS	0x34	R/W	0x01	Indicates buffer3 full or empty
BUF3_BASE_ADDR_LO	0x36	R/W	0x00	Buffer3 base address low
BUF3_BASE ADDR_HI	0x37	R/W	0x00	Buffer3 base address high
BUF3_WR_END_ADDR_LO	0x38	R/W	0x00	Buffer3 write end address low
BUF3_WR_END_ADDR_HI	0x39	R/W	0x00	Buffer3 write end address high

BUF3_RD_END_ADDR_LO	0x3A	R/W	0x00	Buffer3 read end address low

BUF3_RD_END_ADDR_HI	0x3B	R/W	0x00	Buffer3 read end address high
BUF3_WR_ADDR_LO	0x3C	R/W	0x00	Buffer3 write offset address low
BUF3_WR_ADDR_HI	0x3D	R/W	0x00	Buffer3 write offset address high
BUF3_RD_ADDR_LO	0x3E	R/W	0x00	Buffer3 read offset address low
BUF3_RD_ADDR_HI	0x3F	R/W	0x00	Buffer3 read offset address high

DMA Buffer Interrupt Enable (DMA_INTERRUPT_ENABLE, 0xFE00) : Read / Write

7	6	5	4	3	2	1	0
B2_RD	B2_WR	B3_RD	B3_WR	B0_RD	B0_WR	B1_RD	B1_WR

each bit with 1 enables interrupt generation when each interrupt condition occurs.

B2_RD : This bit is used to buffer2 read end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B2_WR : This bit is used to buffer2 write end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B3_RD : This bit is used to buffer3 read end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B3_WR : This bit is used to buffer3 write end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B0_RD : This bit is used to buffer0 read end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B0_WR : This bit is used to buffer0 write end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B1_RD : This bit is used to buffer1 read end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

B1_WR : This bit is used to buffer1 write end interrupt enable.

0 : interrupt disable.
1 : interrupt enable.

DMA Buffer Interrupt Status (DMA_INTERRUPT_STATUS, 0xFE01) : Read / Write

7	6	5	4	3	2	1	0
B2_RD	B2_WR	B3_RD	B3_WR	B0_RD	B0_WR	B1_RD	B1_WR

1 is read at a bit means corresponding interrupt is generated. Writing 1 to a bit clears corresponding interrupt status.

B2_RD : This bit indicates buffer2 read end interrupt status.

0 : buffer2 read don't stop.
1 : buffer2 read stopped.

B2_WR : This bit indicates buffer2 read end interrupt status.

0 : buffer2 write don't stop.
1 : buffer2 write stopped.

B3_RD : This bit indicates buffer3 read end interrupt status.

0 : buffer3 read don't stop.
1 : buffer3 read stopped.

B3_WR : This bit indicates buffer3 write end interrupt status.

0 : buffer3 write don't stop.
1 : buffer3 write stopped.

B0_RD : This bit indicates buffer0 read end interrupt status.

0 : buffer0 read don't stop.
1 : buffer0 read stopped.

B0_WR : This bit indicates buffer0 write end interrupt status.

0 : buffer0 write don't stop.
1 : buffer0 write stopped.

B1_RD : This bit indicates buffer1 read end interrupt status.

0 : buffer1 read don't stop.
1 : buffer1 read stopped.

B1_WR : This bit indicates buffer1 write end interrupt status.

0 : buffer1 write don't stop.
1 : buffer1 write stopped.

Buffer0 write source select (WR_BUF0_FROM_SOURCE_SEL, 0xFE02) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B0_WR_SEL		

"Write buffer0" means copying data from source to RAM(buffer0).

B0_WR_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer0 for USB.
010 : buffer0 for MP3.
011 : buffer0 for MMC.

100 : buffer0 for NAND.
101 : buffer0 for DRM.
110 : buffer0 for NOR.
111 : no source selection.

Read Buffer0 To Destination selected (RD_BUF0_TO_DEST_SEL, 0xFE03) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B0_RD_SEL		

B0_RD_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer0 for USB.
010 : buffer0 for MP3.
011 : buffer0 for MMC.
100 : buffer0 for NAND.
101 : buffer0 for DRM.
110 : buffer0 for NOR.
111 : no source selection.

Buffer1 write source select (WR_BUF1_FROM_SOURCE_SEL, 0xFE04) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B1_WR_SEL		

"Write buffer1" means copying data from source to RAM(buffer1).

B1_WR_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer1 for USB.
010 : buffer1 for MP3.
011 : buffer1 for MMC.
100 : buffer1 for NAND.
101 : buffer1 for DRM.
110 : buffer1 for NOR.
111 : no source selection.

Read Buffer1 To Destination selected (RD_BUF1_TO_DEST_SEL, 0xFE05) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B1_RD_SEL		

B1_RD_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer1 for USB.
010 : buffer1 for MP3.
011 : buffer1 for MMC.
100 : buffer1 for NAND.
101 : buffer1 for DRM.
110 : buffer1 for NOR.
111 : no source selection.

Buffer Status (DMA_BUF_STATUS, 0xFE06) : Read Only

7	6	5	4	3	2	1	0
B2_FULL	B2_EMP	B3_FULL	B3_EMP	B0_FULL	B0_EMP	B1_FULL	B1_EMP

Each bit indicates full or empty status of each buffer.

B2_FULL : This bit indicates of buffer2 full status.

0 : buffer2 is not full.
1 : Buffer2 is full.

B2_EMP : This bit indicates of buffer2 empty status.

0 : buffer2 is not empty.
1 : Buffer2 is empty.

B3_FULL : This bit indicates of buffer2 full status.

0 : buffer3 is not full.
1 : Buffer3 is full.

B3_EMP : This bit indicates of buffer2 empty status.

0 : buffer3 is not empty.
1 : Buffer3 is empty.

B0_FULL : This bit indicates of buffer0 status.

0 : buffer0 is not full.
1 : Buffer0 is full.

B0_EMP : This bit indicates of buffer0 status.

0 : buffer0 is not empty.
1 : Buffer0 is empty.

B1_FULL : This bit indicates of buffer1 status.

0 : buffer1 is not full.
1 : Buffer1 is full.

B1_EMP : This bit indicates of buffer1 status.

0 : buffer1 is not empty.
1 : Buffer1 is empty.

Buffer Reset (DMA_BUF_RESET, 0xFE07) : Write Only

7	6	5	4	3	2	1	0
Reserved				B2_RST	B3_RST	B0_RST	B1_RST

Writing 1 to each bit resets corresponding buffer's full, empty, wptr, rptr registers, and default value is set.

B2_RST : This bit is used to buffer 2 reset.

0 : No action.
1 : Buffer2 reset.

B3_RST : This bit is used to buffer 3 reset.

0 : No action.
1 : Buffer3 reset.

B0_RST : This bit is used to buffer 0 reset.

0 : No action.
1 : Buffer0 reset.

B1_RST : This bit is used to buffer 1 reset.

0 : No action.

1 : Buffer1 reset.

Buffer0 Status (DMA_BUF0_STATUS, 0xFE08) : Read / Write

7	6	5	4	3	2	1	0
Reserved						B0_FULL	B0_EMP

Each bit indicates full or empty status of buffer0. Simultaneous BUF0_FU = 1, BUF0_EMP = 1 is impossible.

B0_FULL : This bit indicates of buffer0 full.

0 : buffer0 is not full.
1 : Buffer0 is full.

B0_EMP : This bit indicates of buffer0 empty.

0 : buffer0 is not empty.
1 : Buffer0 is empty.

Buffer0 Base Address Low (BUF0_BASE_ADDR_LO, 0xFE0A) : Read / Write

7	6	5	4	3	2	1	0
B0_BADDR_LO							

This register is used to read / write the lower 8-bit of 16-bit of the buffer0 base address.

Buffer0 Base Address High (BUF0_BASE_ADDR_HI, 0xFE0B) : Read / Write

7	6	5	4	3	2	1	0
B0_BADDR_HI							

This register is used to read / write the upper 8-bit of 16-bit of the buffer0 base address.

Buffer0 Write End Address Low (BUF0_WR_END_ADDR_LO, 0xFE0C) : Read / Write

7	6	5	4	3	2	1	0
B0_WEADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer0 write end address.

Buffer0 Write End Address High (BUF0_WR_END_ADDR_HI, 0xFE0D) : Read / Write

7	6	5	4	3	2	1	0
B0_WEADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer0 write end address.

Buffer0 Read End Address Low (BUF0_RD_END_ADDR_LO, 0xFE0E) : Read / Write

7	6	5	4	3	2	1	0
B0_READDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer0 read end address.

Buffer0 Read End Address High (BUF0_RD_END_ADDR_HI, 0xFE0F) : Read / Write

7	6	5	4	3	2	1	0
B0_READDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer0 read end address.

Buffer0 Write Address Low (BUF0_WR_ADDR_LO, 0xFE10) : Read / Write

7	6	5	4	3	2	1	0
B0_WADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer0 write address.

Buffer0 Write Address High (BUF0_WR_ADDR_HI, 0xFE11) : Read / Write

7	6	5	4	3	2	1	0
B0_WADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer0 write address.

Buffer0 Read Address Low (BUF0_RD_ADDR_LO, 0xFE12) : Read / Write

7	6	5	4	3	2	1	0
B0_RADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer0 read address.

Buffer0 Read Address High (BUF0_RD_ADDR_HI, 0xFE13) : Read / Write

7	6	5	4	3	2	1	0
B0_RADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer0 read address.

Buffer1 Status (DMA_BUF1_STATUS, 0xFE14) : Read / Write

7	6	5	4	3	2	1	0
Reserved						B1_FULL	B1_EMP

Each bit indicates full or empty status of buffer1. Simultaneous BUF1_FU = 1, BUF1_EMP = 1 is impossible.

B1_FULL : This bit indicates of buffer1 full.

0 : buffer1 is not full.
1 : Buffer1 is full.

B1_EMP : This bit indicates of buffer1 empty.

0 : buffer1 is not empty.
1 : Buffer1 is empty.

Buffer1 Base Address Low (BUF1_BASE_ADDR_LO, 0xFE16) : Read / Write

7	6	5	4	3	2	1	0
B1_BADDR_LO							

This register is used to read / write the lower 8-bit of 16-bit of the buffer1 base address.

Buffer1 Base Address High (BUF1_BASE_ADDR_HI, 0xFE17) : Read / Write

7	6	5	4	3	2	1	0
B1_BADDR_HI							

This register is used to read / write the upper 8-bit of 16-bit of the buffer1 base address.

Buffer1 Write End Address Low (BUF1_WR_END_ADDR_LO, 0xFE18) : Read / Write

7	6	5	4	3	2	1	0
B1_WEADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer1 write end address.

Buffer1 Write End Address High (BUF1_WR_END_ADDR_HI, 0xFE19) : Read / Write

7	6	5	4	3	2	1	0
B1_WEADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer1 write end address.

Buffer1 Read End Address Low (BUF1_RD_END_ADDR_LO, 0xFE1A) : Read / Write

7	6	5	4	3	2	1	0
B1_READDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer1 read end address.

Buffer1 Read End Address High (BUF1_RD_END_ADDR_HI, 0xFE1B) : Read / Write

7	6	5	4	3	2	1	0
B1_READDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer1 read end address.

Buffer1 Write Address Low (BUF1_WR_ADDR_LO, 0xFE1C) : Read / Write

7	6	5	4	3	2	1	0
B1_WADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer1 write address.

Buffer1 Write Address High (BUF1_WR_ADDR_HI, 0xFE1D) : Read / Write

7	6	5	4	3	2	1	0
B1_WADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer1 write address.

Buffer1 Read Address Low (BUF1_RD_ADDR_LO, 0xFE1E) : Read / Write

7	6	5	4	3	2	1	0
B1_RADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer1 read address.

Buffer1 Read Address High (BUF1_RD_ADDR_HI, 0xFE1F) : Read / Write

7	6	5	4	3	2	1	0
B1_RADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer1 read address.

Buffer2 Write Source Select (WR_BUF2_FROM_SOURCE_SEL, 0xFE22) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B2_WR_SEL		

"Write buffer2" means copying data from source to RAM(buffer2).

B2_WR_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer2 for USB.
010 : buffer2 for MP3.
011 : buffer2 for MMC.
100 : buffer2 for NAND.
101 : buffer2 for DRM.
110 : buffer2 for NOR.
111 : no source selection.

Read Buffer2 To Destination selected (RD_BUF2_TO_DEST_SEL, 0xFE23) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B2_RD_SEL		

B2_RD_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer2 for USB.
010 : buffer2 for MP3.
011 : buffer2 for MMC.
100 : buffer2 for NAND.
101 : buffer2 for DRM.
110 : buffer2 for NOR.
111 : no source selection.

Buffer3 write source select (WR_BUF3_FROM_SOURCE_SEL, 0xFE24) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B3_WR_SEL		

"Write buffer3" means copying data from source to RAM(buffer3).

B3_WR_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer3 for USB.
010 : buffer3 for MP3.
011 : buffer3 for MMC.
100 : buffer3 for NAND.
101 : buffer3 for DRM.
110 : buffer3 for NOR.
111 : no source selection.

Read Buffer3 To Destination selected (RD_BUF3_TO_DEST_SEL, 0xFE25) : Read / Write

7	6	5	4	3	2	1	0
Reserved					B3_RD_SEL		

B3_RD_SEL : These bits are used to set the selection for six sources.

000 : no source selection.
001 : buffer3 for USB.
010 : buffer3 for MP3.
011 : buffer3 for MMC.

100 : buffer3 for NAND.
101 : buffer3 for DRM.
110 : buffer3 for NOR.
111 : no source selection.

Buffer2 Status (DMA_BUF2_STATUS, 0xFE28) : Read / Write

7	6	5	4	3	2	1	0
Reserved						B2_FULL	B2_EMP

Each bit indicates full or empty status of buffer2. Simultaneous BUF2_FU = 1, BUF2_EMP = 1 is impossible.

B2_FULL : This bit indicates of buffer2 full.

0 : buffer2 is not full.
1 : Buffer2 is full.

B2_EMP : This bit indicates of buffer2 empty.

0 : buffer2 is not empty.
1 : Buffer2 is empty.

Buffer2 Base Address Low (BUF2_BASE_ADDR_LO, 0xFE2A) : Read / Write

7	6	5	4	3	2	1	0
B2_BADDR_LO							

This register is used to read / write the lower 8-bit of 16-bit of the buffer2 base address.

Buffer2 Base Address High (BUF2_BASE_ADDR_HI, 0xFE2B) : Read / Write

7	6	5	4	3	2	1	0
B2_BADDR_HI							

This register is used to read / write the upper 8-bit of 16-bit of the buffer2 base address.

Buffer2 Write End Address Low (BUF2_WR_END_ADDR_LO, 0xFE2C) : Read / Write

7	6	5	4	3	2	1	0
B2_WEADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer2 write end address.

Buffer2 Write End Address High (BUF2_WR_END_ADDR_HI, 0xFE2D) : Read / Write

7	6	5	4	3	2	1	0
B2_WEADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer2 write end address.

Buffer2 Read End Address Low (BUF2_RD_END_ADDR_LO, 0xFE2E) : Read / Write

7	6	5	4	3	2	1	0
B2_READDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer2 read end address.

Buffer2 Read End Address High (BUF2_RD_END_ADDR_HI, 0xFE2F) : Read / Write

7	6	5	4	3	2	1	0
B2_READDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer2 read end address.

Buffer2 Write Address Low (BUF2_WR_ADDR_LO, 0xFE30) : Read / Write

7	6	5	4	3	2	1	0
B2_WADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer2 write address.

Buffer2 Write Address High (BUF2_WR_ADDR_HI, 0xFE31) : Read / Write

7	6	5	4	3	2	1	0
B2_WADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer2 write address.

Buffer2 Read Address Low (BUF2_RD_ADDR_LO, 0xFE32) : Read / Write

7	6	5	4	3	2	1	0
B2_RADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer2 read address.

Buffer2 Read Address High (BUF2_RD_ADDR_HI, 0xFE33) : Read / Write

7	6	5	4	3	2	1	0
B2_RADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer2 read address.

Buffer3 Status (DMA_BUF3_STATUS, 0xFE34) : Read / Write

7	6	5	4	3	2	1	0
Reserved						B3_FULL	B3_EMP

Each bit indicates full or empty status of buffer3. Simultaneous BUF3_FU = 1, BUF3_EMP = 1 is impossible.

B3_FULL : This bit indicates of buffer3 full.

0 : buffer3 is not full.
1 : Buffer3 is full.

B3_EMP : This bit indicates of buffer3 empty.

0 : buffer3 is not empty.
1 : Buffer3 is empty.

Buffer3 Base Address Low (BUF3_BASE_ADDR_LO, 0xFE36) : Read / Write

7	6	5	4	3	2	1	0
B3_BADDR_LO							

This register is used to read / write the lower 8-bit of 16-bit of the buffer3 base address.

Buffer3 Base Address High (BUF3_BASE_ADDR_HI, 0xFE37) : Read / Write

7	6	5	4	3	2	1	0
B3_BADDR_HI							

This register is used to read / write the upper 8-bit of 16-bit of the buffer3 base address.

Buffer3 Write End Address Low (BUF3_WR_END_ADDR_LO, 0xFE38) : Read / Write

7	6	5	4	3	2	1	0
B3_WEADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer3 write end address.

Buffer3 Write End Address High (BUF3_WR_END_ADDR_HI, 0xFE39) : Read / Write

7	6	5	4	3	2	1	0
B3_WEADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer3 write end address.

Buffer3 Read End Address Low (BUF3_RD_END_ADDR_LO, 0xFE3A) : Read / Write

7	6	5	4	3	2	1	0
B3_READDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer3 read end address.

Buffer3 Read End Address High (BUF3_RD_END_ADDR_HI, 0xFE3B) : Read / Write

7	6	5	4	3	2	1	0
B1_READDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer1 read end address.

Buffer3 Write Address Low (BUF3_WR_ADDR_LO, 0xFE3C) : Read / Write

7	6	5	4	3	2	1	0
B3_WADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer3 write address.

Buffer3 Write Address High (BUF3_WR_ADDR_HI, 0xFE3D) : Read / Write

7	6	5	4	3	2	1	0
B3_WADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer3 write address.

Buffer3 Read Address Low (BUF3_RD_ADDR_LO, 0xFE3E) : Read / Write

7	6	5	4	3	2	1	0
B3_RADDR_LO							

This register is used to read / write the lower 8-bit of 12-bit of the buffer3 read address.

Buffer3 Read Address High (BUF3_RD_ADDR_HI, 0xFE3F) : Read / Write

7	6	5	4	3	2	1	0
B3_RADDR_HI							

This register is used to read / write the upper 4-bit of 12-bit of the buffer3 read address.

11. MMC/SD Card Interface

11. MMC/SD Card Interface

NX5850 has several Media interface such as SD/MMC, LCD, Nor Flash and Nand Flash interface. The control and the data interface for each media interface has independent port and control registers except multiplexed with Nor Flash interface pin.

11.1. MMC/SD Interface

To use the MMC block, we have to clear the power-down as setting the SYS_BLOCK_POWER_CONTROL (0Xff06)[1] to 0 and be disabled the GPIO mode of the following pins which is used in the register 'GPIO_1_ENABLE(0xFF43)' for the MMC interface. 1 bit data mode use mmc_dat pin only as data pin of MMC interface. 4bit data mode uses mmc_dat(bit0), GPIO0 in(bit1), GPIO1(bit2), and GPIO2(bit3).

- mmc_clk
- mmc_cmd
- mmc_dat(one bit only or bit0), GPIO0(bit1), GPIO1(bit2), GPIO2(bit3)

11.2. Initialization of MMC and SD Card

There are two modes for the communication to the MMC card such as 'MMC mode' and 'SPI mode', and three modes for the communication to the SD card such as '1-bit SD Bus', '4-bit SD Bus' and 'SPI Bus Interface'. The MMC mode of the MMC card and the 1-bit SD bus method of the SD card is the almost same except some commands. NX5850 supports the MMC mode of the MMC card and 1-bit or 4bit SD bus method of the SD card.

The way to distinguish the SD card from the MMC card is as follows:

If there is the response to command of ACMD41, the SD card is selected, and If not, MMC card is selected. The following flow chart shows the initialization of the SD card.

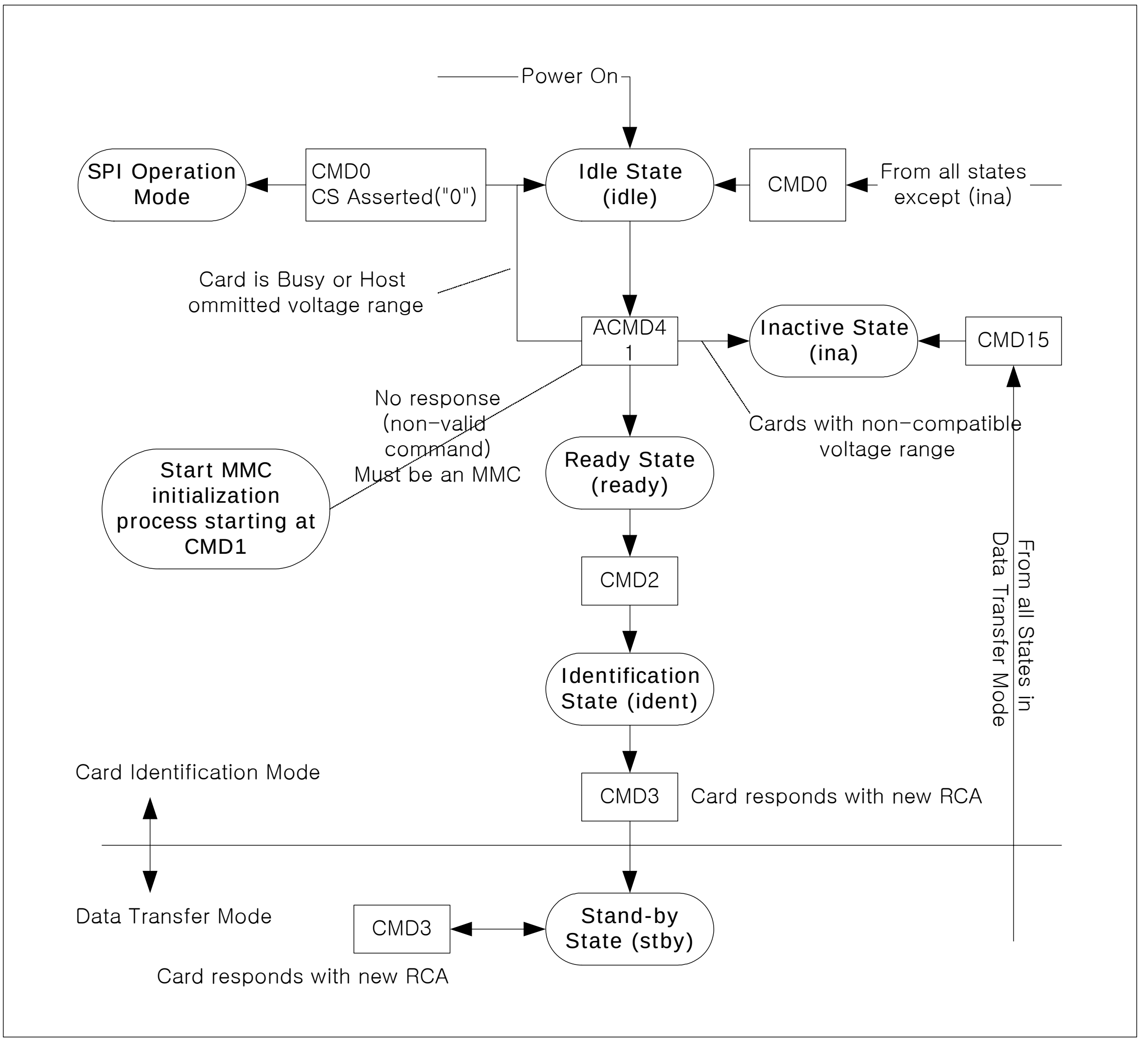

Figure 10. The initial procedure of the SD Card

The following flow chart shows the initialization procedure of the MMC card when the MMC card is selected due to no response to ACMD41.

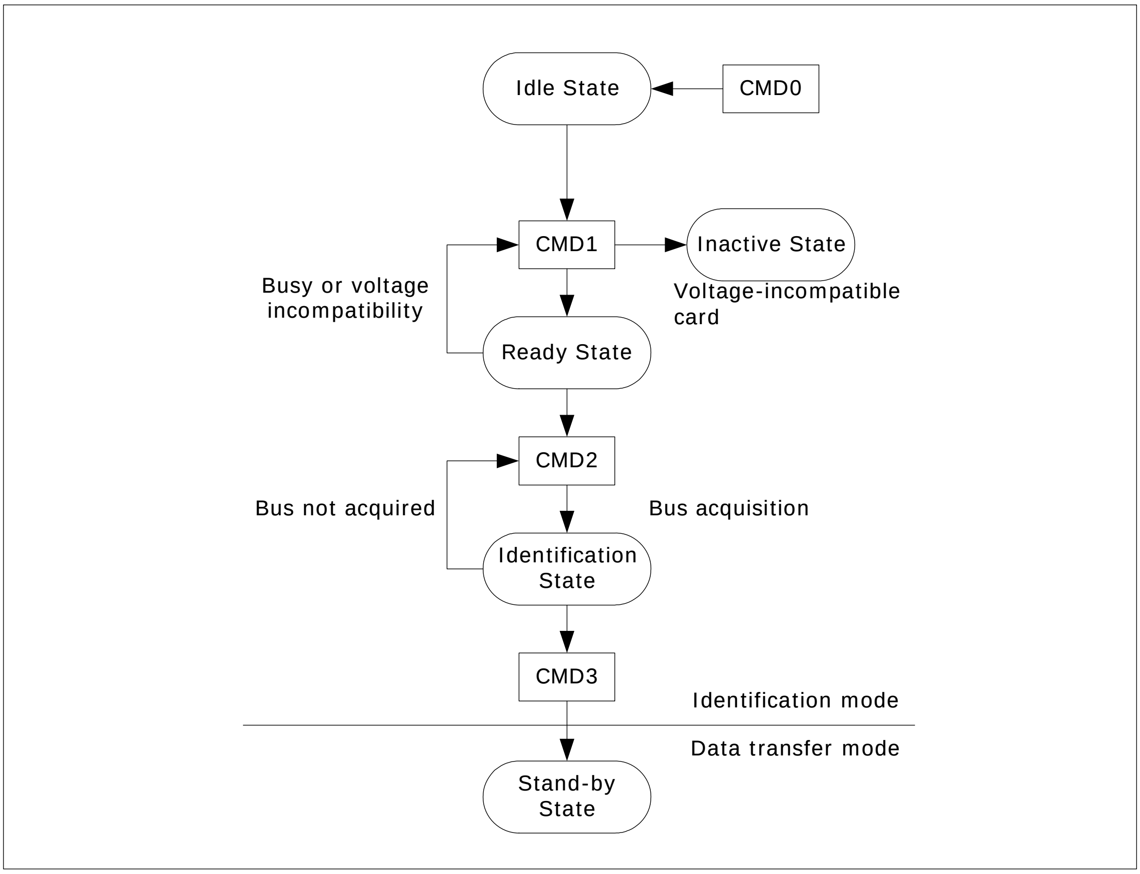

Figure 11. The initial procedure of the MMC Card

The data transfer mode gets ready to communicate the data when the initialization procedure like above is finished. The commands for the SD card and the MMC card are almost same. The following flow chart shows the operation of the data transfer mode.

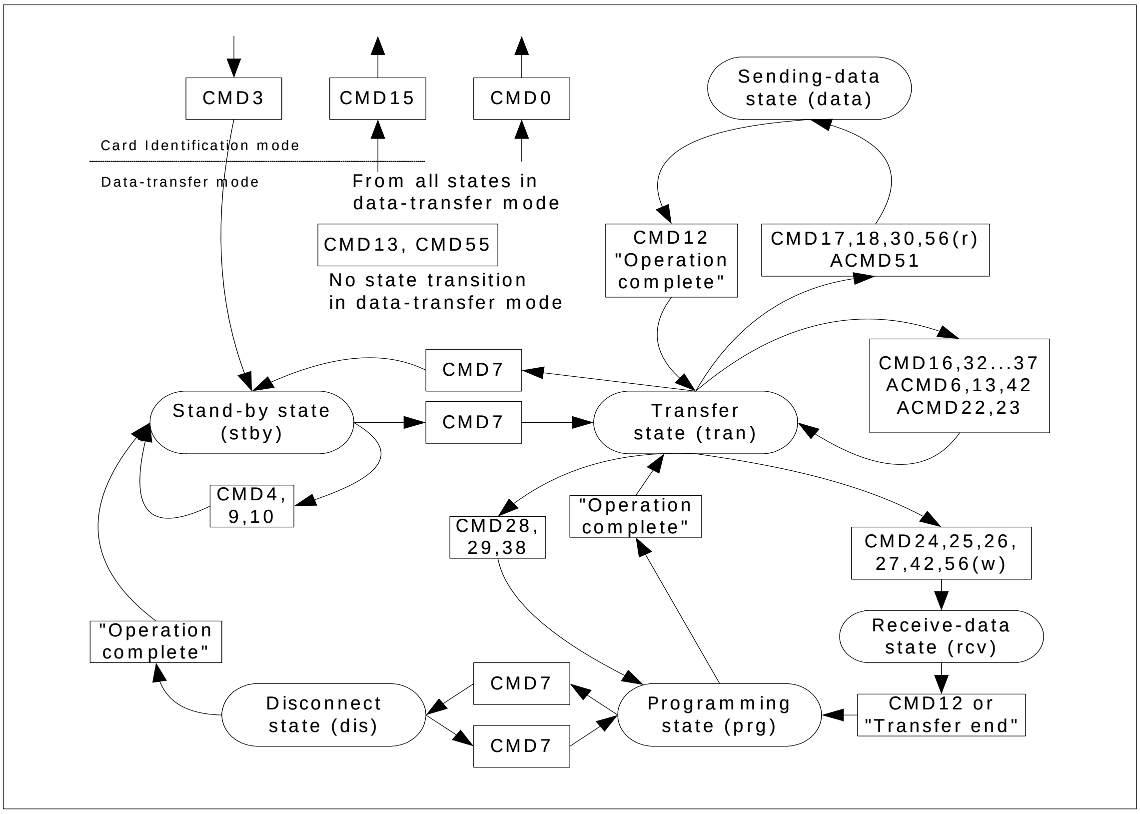

Figure 12. Data-transfer Mode Flowchart

The following Table 21 shows some commands used frequently. For more details, refer to the specification of the MMC and the SD.

Table 32. Command Description of MMC

CMD Index	Abbreviation	Function	MMC Mode		
			Type	Argument	Resp.
CMD0	GO_IDLE_STATE	MMC reset	bc	[31:0] Stuff bits	-
CMD1	SEND_OP_COND	MMC R/B polling Operation voltage setting (MMC mode only)	bcr	[31:0] OCR without busy	R2
CMD2	ALL_SEND_CID	CID transmission request	bcr	[31:0] Stuff bits	R3
CMD3	SET_RELATIVE_ADDR	RCA setting	ac	[31:16] RCA [15:0] Stuff bits	R1
CMD4	SET_DSR	DSR setting	bc	[31:16] DSR [15:0] Stuff bits	-
CMD7	SELECT/ DESELECT_CARD	Selection of MMC to be accessed	ac	[31:16] RCA [15:0] Stuff bits	R1b (only from the selected card)
CMD9	SEND_CSD	CSD transmission request	ac	[31:16] RCA [15:0] Stuff bits	R2
CMD17	READ_SINGLE_BLOCK	Single block read	adtc	[31:0] Data address	R1
CMD24	WRITE_BLOCK	Single block write	adtc	[31:0] Data address	R1

11.3. MMC/SD Card Control Register

Table 33. MMC/SD Card Control Register Map (P2 = 0xFB)

Function	Address (Hex)	Type	Reset	Description
MMC_CMD_RES_RW	0x00 ~0x10	R/W	-	RD or WR with Command or response.
MMC_CLK_CONTROL	0x11	R/W	0x00	Supply with MMC driving frequency.
MMC_CMD_CONTROL	0x12	R/W	0x00	Control the MMC command.
MMC_CMD_STATUS	0x13	RO	-	Indicates MMC command status.
MMC_DATA_CONTROL	0x14	R/W	0x00	RD or WR with MMC card.
MMC_DATA_STATUS	0x15	RO	-	Indicates MMC data status.
MMC_DATA_TIME_OUT_LO	0x16	R/W	0x00	MMC data time out low.
MMC_DATA_TIME_OUT_HI	0x17	R/W	0x00	MMC data time out high.
MMC_INT_ENABLE	0x18	R/W	0x00	MMC interrupt enable.
MMC_INT_STATUS	0x19	R/W	0x00	MMC interrupt status.

Command And Response Read Or Write (MMC_CMD_RES_RW, 0xFB00~0xFB10) : Read / Write

7	6	5	4	3	2	1	0
CMD_RES_RW							

Write command for MMC to this register with command LSB to address 0xFB00 LSB and command MSB to higher address. MSB of 48bit MMC command goes to address 0xFB05 MSB. The command written to these register is sent to MMC by writing address 0xFB12 command operation mode. After response data LSB to address 0xFB00 LSB and response data MSB to higher address.

0xFB00~0xFB05 is for command and response register. But 0xFB06~0xFB10 is for response only.

Offset	Command	Response	Response
0xFB00	command[7:0]	-	response[7:0]
0xFB01	command[15:8]	-	response[15:8]
0xFB02	command[23:16]	-	response[23:16]
0xFB03	command[31:24]	-	response[31:24]
0xFB04	command[39:32]	-	response[39:32]
0xFB05	command[47:40]	-	response[47:40]
0xFB06	-	-	response[55:48]
0xFB07	-	-	response[63:56]
0xFB08	-	-	response[71:64]
0xFB09	-	-	response[79:72]
0xFB0A	-	-	response[87:80]
0xFB0B	-	response[7:0]	response[95:88]
0xFB0C	-	response[15:8]	response[103:96]
0xFB0D	-	response[23:16]	response[111:104]
0xFB0E	-	response[31:24]	response[119:112]
0xFB0F	-	response[39:32]	response[127:120]
0xFB10	-	response[47:40]	response[135:128]

MMC Clock Control (MMC_CLK_CONTROL, 0xFB11) : Read / Write

7	6	5	4	3	2	1	0
CLK_SUP	CLK_DIV						

This register is used to MMC clock control.
MMC driving frequency = system clock / (1 + divider)

CLK_SUP : This bit is used to MMC clock supply.

0 : Makes MMC clock supplied when command or data is sent or received only, and no command or no data makes no clock, so less power consumption.
1 : Makes continuous MMC clock supplied to MMC.

CLK_DIV : MMC driving clock divider controls MMC clock speed and clock source is system clock.

MMC Command Control (MMC_CMD_CONTROL, 0xFB12) : Read / Write

7	6	5	4	3	2	1	0
Reserved				TOUT_CHK	CRC_GNR	CMD_OPR	

Period of response time out is 64 MMC clock after command is sent. It makes command in command register sent to MMC by writing [1:0] with 1, 2 or 3 according to command type that user writes to address 0xFB00~0xFB05.

TOUT_CHK : This bit is used to response time out check enable.

0 : Response time out check disable.
1 : Response time out check enable.

CRC_GNR : This bit is used to command CRC generation.

0 : MCU makes command CRC and send it.
1 : Command CRC auto generation by MMC block.

CMD_OPR : These bits are used to set the selection for command operation mode.

00 : Idle state.
01 : No response command.
10 : 48bit response command.
11 : 136bit response command.

MMC Command Control (MMC_CMD_STATUS, 0xFB13) : Read Only

7	6	5	4	3	2	1	0
Reserved						TOUT_ERR	CRC_ERR

Read this register for checking response error or not.

TOUT_ERR : This register is used to checking response time out error.

0 : Response time out no error.
1 : Response time out error.

CRC_ERR : This register is used to checking response CRC error.

0 : Response CRC no error.
1 ; Response CRC error.

MMC Data Control Register (MMC_DATA_CONTROL, 0xFB14) : Read / Write

7	6	5	4	3	2	1	0
Reserved			4B_MODE	STR_PND	TOUT_CHK	WR_STR	RD_STR

This register is used to read or write for MMC card.

4B_MODE : This bit is used to enables 4bit data line or 1bit data line.

0 : 1bit mode.
1 : 4bit mode.

4bit mode data pin is bit0 = M_DATA (pin 65), bit1 = PORT0 (pin 74), bit2 = PORT1 (pin 75), bit3 = PORT2 (pin 76). And 1bit mode data pin is M_DATA(pin 65).

STR_PND : Much data that is bigger than maximum DMA transfer size needs next DMA setting time to transfer remaining data by DMA, SRT_PND needs to be enabled at this time.

0 : No action.
1 : DMA start pending enable.

TOUT_CHK : This bit is used to checks time out after read command.

0 : read data time out check disable.
1 : read data time out check enable.

WR_STR : This bit is used to makes start writing data to MMC by DMA.

0 : No action.
1 : Write start command.

RD_STR : This bit is used to makes start reading data from MMC by DMA.

0 : No action.
1 : Read start command.

MMC Data Status (MMC_DATA_STATUS, 0xFB15) : Read Only

7	6	5	4	3	2	1	0
Reserved						RD_TOUT	RW_CRC

This register is used to checking the MMC data status.

RD_TOUT : If data does not come in during the time set at address 0xFB16 and 0xFB17 register after command of data read, this bit become 1.

0 : No action.
1 : Occurred read data time out.

RW_CRC : If data CRC error is occurred during reading or writing data, this bit becomes 1.

0 : No action.
1 : Occurred read or write data CRC error.

MMC Data Time Out Low (MMC_DATA_TIME_OUT_LO, 0xFB16) : Read / Write

7	6	5	4	3	2	1	0
TOUT_LO							

This register is used to read or write the lower 8-bit of the 16-bit MMC data time out.

Maximum waiting time the data come in after command of data read. If data does not come in until the time set after command of data read, address 0xFB15[1] is set.

Timer counter speed is clock speed generated by "MMC driving clock divider" of this block offset 0xFB11[6:0].

MMC Data Time Out High (MMC_DATA_TIME_OUT_HI, 0xFB17) : Read / Write

7	6	5	4	3	2	1	0
TOUT_HI							

This register is used to read or write the upper 8-bit of the 16-bit MMC data time out.

MMC Interrupt enable (MMC_INT_ENABLE, 0xFB18) : Read / Write

7	6	5	4	3	2	1	0
Reserved		RES_TOUT	RD_TOUT	RES_CRC	RW_CRC	CMD_END	DATA_END

If corresponding interrupts generated of this register bits occurs after enabling, corresponding bits of address 0xFB19 is set and interrupt signal goes to system block "MMC control block interrupt".

RES_TOUT : This bit is used to response time out interrupt enable.

0 : Interrupt disable.
1 : Interrupt enable.

RD_TOUT : This bit is used to read data time out interrupt enable.

0 : Interrupt disable.
1 : Interrupt enable.

RES_CRC : This bit is used to response CRC error interrupt enable.

0 : Interrupt disable.
1 : Interrupt enable.

RW_CRC : This bit is used to data CRC error interrupt enable.

0 : Interrupt disable.
1 : Interrupt enable.

CMD_END : This bit is used to command end interrupt enable.

0 : Interrupt disable.
1 : Interrupt enable.

DATA_END : This bit is used to data end interrupt enable.

0 : Interrupt disable.
1 : Interrupt enable.

MMC Interrupt status (MMC_INT_STATUS, 0xFB19) : Read / Write

7	6	5	4	3	2	1	0
Reserved		RES_TOUT	RD_TOUT	RES_CRC	RW_CRC	CMD_END	DATA_END

This register is used to checking the MMC interrupt status. If 1 is read at each bit, corresponding bit interrupt is generated. Writing 1 to the bit clears the bit.

RES_TOUT : This bit is used to checking the response time out interrupt status.

0 : No action.
1 : Generated response time out interrupt.

RD_TOUT : This bit is used to checking the read data time out interrupt status.

0 : No action.
1 : Generated read data time out interrupt.

RES_CRC : This bit is used to checking the response CRC error interrupt status.

0 : No action.
1 : Generated response CRC error interrupt.

RW_CRC : This bit is used to checking the R/W data CRC error interrupt status.

0 : No action.
1 : Generated R/W data CRC error interrupt.

CMD_END : This bit is used to checking the command end interrupt status.

0 : No action.
1 : Generated command end interrupt.

DATA_END : This bit is used to checking the data end interrupt status.

0 : No action.
1 : Generated data end interrupt.

12. NAND Controller

12. NAND Controller

Nand Flash Memory Controller supports continuous reading 2Kbytes data with ECC and DMA supports.

12.1. NAND Flash Interface

The flash memory can be used up to four. NX5850 gets the MCU to configure the pins for the NAND flash interface. MCU can control signals such as ALE and CLE, directly. For example, to give a command to the flash memory, the bit[3:1] of the register ‘NAND_PIN_CONTROL(0xfc02)’ has to be set to 0x2, it results in CLE=High and ALE=Low.
There are some ways to set commands for the flash memory, but here gives four commands that are used frequently (It assumes that the No. 0 cell NAND flash memory is selected). To use the flash block, the power-down mode has to be disabled as the SYSTEM_BLOCK_POWER_CONTROL(0xFF06[0]) is set to low, and the GPIO mode has to be disabled. (Refer to below register maps for more detailed)

12.1.1. Reading data from NAND Flash Memory

Figure 16. is a timing diagram to show reading the data from the NAND flash memory. For this operation, after setting CLE to high and ALE to low, write the READ command (0x00) into the I/O line(address 0xfc03).

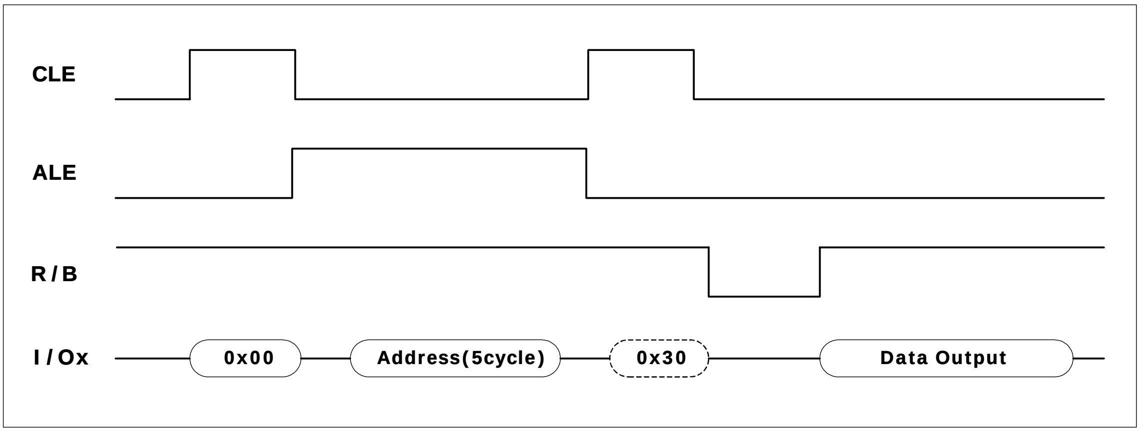

Figure 13. The Read Timing of NAND Flash Memory

And then, after setting CLE to low and ALE to high, write the address of the data to the I/O line. The data can be read from the I/O line. The timing diagram of Figure 16. shows 0x30 is used in case of the advanced flash only.
To read the data from the NAND flash, user can use DMA to transfer data from Nand Flash to the DMA buffer by setting the register ‘NAND_DMA_MODE_CONTROL(0xfc09), NAND_DMA_START_CONTROL[5] (0xfc0a)’, defining DMA buffer with register(0xfe02[4] if BUF0 use, 0xfe0a~0xfe0d, 0xfe10, 0xfe11) of DMA_ctrl block to enable and start the DMA transfer.

The function ‘DownFromFlash()’ shows the DMA setting and the process of giving the read the command to the NAND flash.

12.1.2. Example for NAND Flash Read

Program 5. The function for reading the NAND flash.

```
//---------------------------------------------------------------------------------------------------
// DownFromFlash()
// Read data as much as length from the iAddr Sector of the NAND flash memory to DMA buffer.
// (NAND flash -> Internal RAM)
// uchar *Buff_Addr : DMA Buffer Address to store data read
// LongChar iAddr          : Address of the Flash memory to read based on sector.
// uint length : Data size to read
//---------------------------------------------------------------------------------------------------

#define         FLASH_FCONTROL              0xFC03
#define         FLASH_FCOMMAND              0xFC05
#define         FLASH_F01                   0xFC05
#define         FLASH_F02                   0xFC05
#define         FLASH_F03                   0xFC05
#define         FLASH_F04                   0xFC05
#define         FLASH_F05                   0xFC05
#define         FLASH_FSTATUS               0xFC05
#define         FLASH_FDMA_READ             0xFC07
#define         FLASH_FDMA_WRITE            0xFC08

void DownFromFlash(uchar *Buff_Addr, LongChar iAddr, uint length)
{
    uint baddr = Buff_Addr;
    if(Flash_Bank_Num == 0) {
        // buffer 0 address
        B0_SetAddr(baddr);
        B0_Reset();
        B0_ReadEnd(length-1);
        B0_WriteEnd(length-1);
        write_XDATA(DMA_FLASH_MODE, 0x04);
    }
    else {
        // buffer 1 adress
        B1_SetAddr(baddr);
        B1_Reset();
        B1_ReadEnd(length-1);
        B1_WriteEnd(length-1);
        write_XDATA(DMA_FLASH_MODE, 0x05);
    }
    write_XDATA(FLASH_FCONTROL, 0xe2);
    write_XDATA(FLASH_FCOMMAND, 0x00);
    write_XDATA(FLASH_FCONTROL, 0xe4);

    if(Is_Advanced)
        write_XDATA(FLASH_F05, 0);
    write_XDATA(FLASH_F04, 0);
    write_XDATA(FLASH_F03, iAddr.c[3]);
    write_XDATA(FLASH_F02, iAddr.c[2]);
    write_XDATA(FLASH_F01, iAddr.c[1]);

    if(Is_Advanced) {
        write_XDATA(FLASH_FCONTROL, 0xe2);
        write_XDATA(FLASH_FCOMMAND, 0x30);
    }
    write_XDATA(FLASH_FCONTROL, 0xe0);
    write_XDATA(FLASH_FDMA_READ, 0x2f);
}
```

12.1.3. Writing data to NAND Flash memory

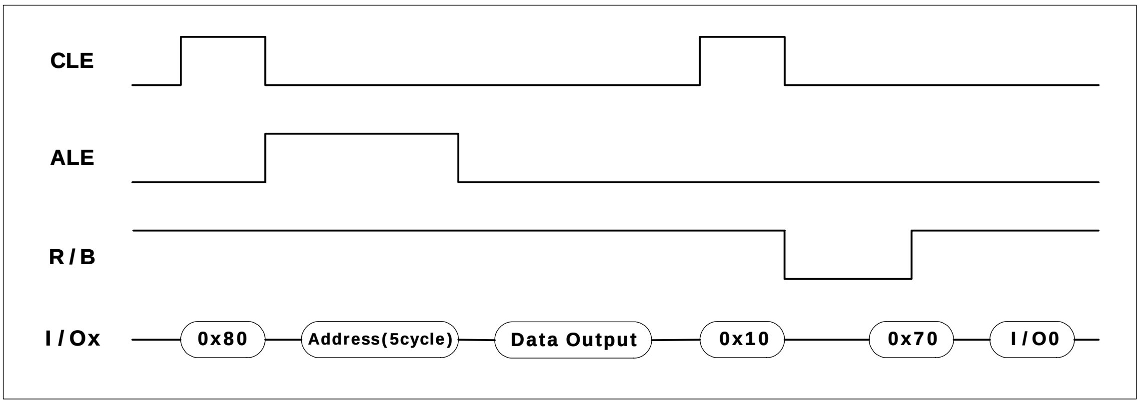

Figure 14. The NAND Flash Write Timing

Figure 17. is the timing diagram to show that writing the data to the NAND flash memory. For this operation, after setting CLE to high and ALE to low, write the sequential data input command(80H) to the I/O line. After setting CLE to low and ALE to high, write the address into the I/O line and that the data on the I/O line are written into the NAND flash. When the data are written through by the DMA transfer, the sequential data input process is done. And then, write the program command(10H). If you want to check whether the process is done or not, you should check the BUSY status at the R/B pin with performing the READ Status command(70H). To write the data to the NAND flash, user can use DMA to transfer data to Nand Flash from the DMA buffer by setting the register 'NAND_DMA_MODE_CONTROL(0xfc09), NAND_DMA_START_CONTROL[4] (0xfc0a)', defining DMA buffer with register(0xfe02[4] if BUF0 use, 0xfe0a, 0xfe0b, 0xfe0e, 0xfe0f, 0xfe12, 0xfe13) of DMA_ctrl block to enable and start the DMA transfer.

The function 'UpToFlash()' shows the DMA setting and the process of giving the write command to the NAND flash.

12.1.4. Example for NAND Flash Write

Program 6. The function for writing the NAND flash.

```
//-----------------------------------------------------------------------------------------------------
// UpToFlash ()
// Write data by length from DMA Buffer to jAddr Sector of Flash Memory.
// (Internal RAM -> Nand Flash)
// uchar *Buff_Addr : DMA Buffer Address having data to be written.
// LongChar iAddr    : The base of the NAND flash memory is 'Sector'.
// uint length        : data length to be written.
//----------------------------------------------------------------------------------------------------

void UpToFlash(uchar *Buff_Addr, LongChar iAddr, uint length)
{
    uint baddr = Buff_Addr;

    if(Flash_Bank_Num == 0) {
        // buffer 0 address
        B0_SetAddr(baddr);
        B0_Reset();
        B0_ReadEnd(length-1);
        B0_WriteEnd(length-1);
        write_XDATA(B0_BSTATUS,0x02);
        write_XDATA(DMA_FLASH_MODE, 0x04);
    }
    else {
        // buffer 1 adress
        B1_SetAddr(baddr);
        B1_Reset();
        B1_ReadEnd(length-1);
        B1_WriteEnd(length-1);
        write_XDATA(B1_BSTATUS,0x02);//check
        write_XDATA(DMA_FLASH_MODE, 0x05);
    }

    write_XDATA(FLASH_FCONTROL, 0xe2);
    write_XDATA(FLASH_FCOMMAND, 0x80);
    write_XDATA(FLASH_FCONTROL, 0xe4);

    if(Is_Advanced)
        write_XDATA(FLASH_F05, 0);

    write_XDATA(FLASH_F04, 0);
    write_XDATA(FLASH_F03, iAddr.c[3]);

    write_XDATA(FLASH_F02, iAddr.c[2]);
    write_XDATA(FLASH_F01, iAddr.c[1]);
    write_XDATA(FLASH_FCONTROL, 0xe0);
    write_XDATA(FLASH_FDMA_WRITE, 0x0f);
}
```

12.1.5. Read ID Operation

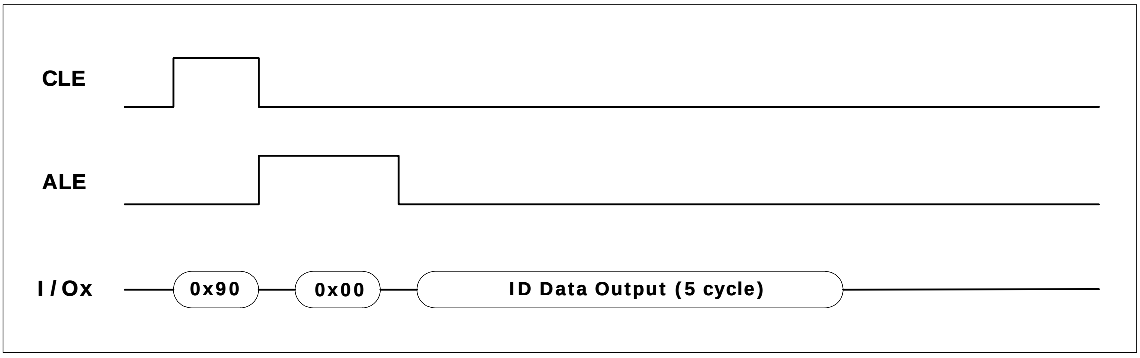

Figure 15. The Read ID Operation Timing

Figure 18. is the timing diagram to read the ID of the NAND flash. For this operation, after setting CLE to high and ALE to low, write the read the ID command(0x90) into the I/O line. After setting CLE to low and ALE to high, write Address '0x00' into the I/O line. And then, read the ID data from the I/O line.

The function 'Flash_ReadIO()' is a process to read the ID of the flash memory.

12.1.6. Example for NAND Flash ID Read

Program 7. The function for reading the NAND flash ID.

```
//-----------------------------------------------------------------------------------------------------
// Flash_ReadID ()
// Get the data from NAND Flash Memory.
// uchar *DeviceID: Device IO of NAND Flash Memory.
// uchar Maker: Maker of NAND Flash Memory.
//-----------------------------------------------------------------------------------------------------


uchar Flash_ReadID(uchar *DeviceID)
{
    uchar Maker;

    write_XDATA(FLASH_FCONTROL, 0xe2);
    write_XDATA(FLASH_FCOMMAND, 0x90);

    write_XDATA(FLASH_FCONTROL, 0xe4);
    write_XDATA(FLASH_F01, 0x00);
    write_XDATA(FLASH_FCONTROL, 0xe0);

    Maker = read_XDATA(FLASH_FSTATUS);    // Maker
    *DeviceID = read_XDATA(FLASH_FSTATUS);         // Device

    return Maker;
}
```

12.1.7. Block Erase Operation of NAND Flash

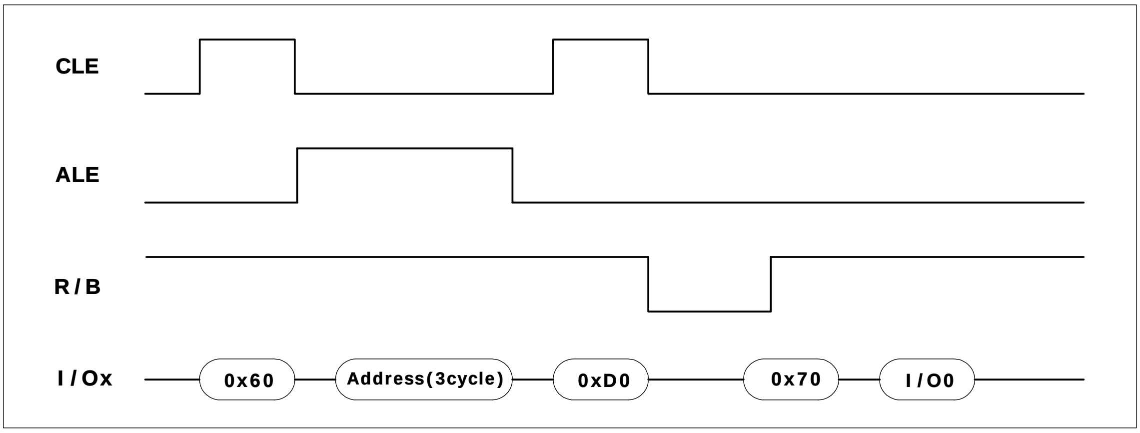

Figure 16. The Block Erase Operation Timing of the NAND Flash

Figure 19. is the timing diagram to erase a block of the NAND flash. For this operation, after setting CLE to high and ALE to low, write the block erase command(0x60) into the I/O line. After CLE to low and ALE to high, write the block address into the I/O line. After setting CLE to high and ALE to low, write the command(0xD0) into the I/O line to perform the block erase. If you want to check whether the process is done or not, you should check the status of the R/B pin if it has the busy status, performing the command (0x70) for the read status.

The function 'Memory_Erase()' is a process to erase the block of the NAND flash.

12.1.8. Example for Block Erase of NAND Flash

Program 8. The function for the block erase of the NAND flash.

```
void Flash_Ready()
{
    uchar buf;

    do{
        write_XDATA(FLASH_FCONTROL, 0xe2);//cle
        write_XDATA(FLASH_FCOMMAND, 0x70);
        write_XDATA(FLASH_FCONTROL, 0xe0);
        buf = read_XDATA(FLASH_FSTATUS);
    } while((buf&0x40)==0);
}


void Memory_Erase(LongChar iAddr)
{
    write_XDATA(FLASH_FCONTROL, 0xe2);//cle
    write_XDATA(FLASH_FCOMMAND, 0x60);
    write_XDATA(FLASH_FCONTROL, 0xe4);//ale
    write_XDATA(FLASH_F03, iAddr.c[3]);
    write_XDATA(FLASH_F02, iAddr.c[2]);
    write_XDATA(FLASH_F01, iAddr.c[1]);
    write_XDATA(FLASH_FCONTROL, 0xe2);//cle
    write_XDATA(FLASH_FCOMMAND, 0xD0);

    Flash_Ready();
}
```

< ECC Overview >

The error checking and correction applied at NX5850 are designed in the Haming algorithm. The ECC code of 3 byte is made with the data of 512 byte unit. Therefore, the ECC code for the data 4096 bit is 24 bit. The lower 6 bits of the ECC code of 24 bits are the column parity and upper 18 bits are the line parity.
The ECC process is performed in the hardware to minimize the MCU operation. The ECC procedure is like as Figure 20.
The data with the ECC code are stored into the NAND flash. At this time, the ECC code is stored into the special spare area on the NAND Flash as the Figure 22. The block data(512byte) from NAND flash are read with the ECC code stored, and the new ECC code is created with the data automatically. And the result that doing the exclusive OR with the two ECC codes is stored in. The MCU performs the some processes by the result.

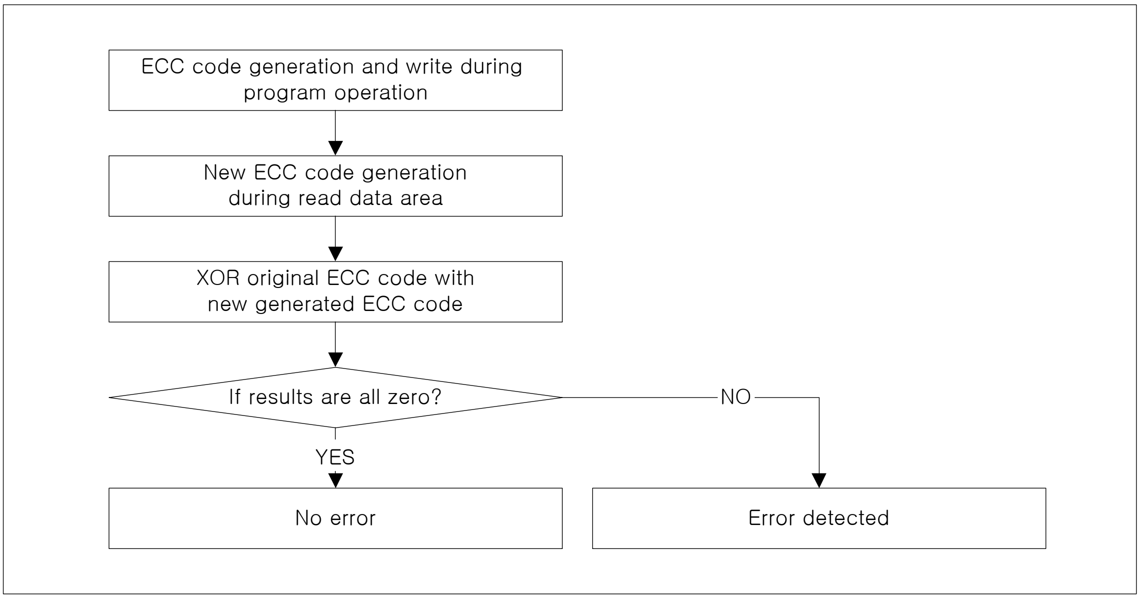

Figure 17. ECC Processing Flow

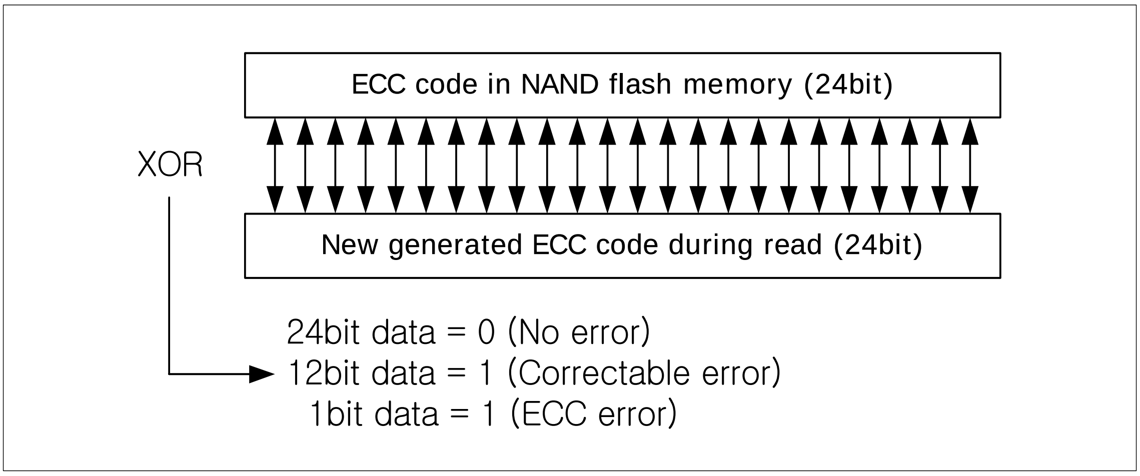

Figure 18. ECC code compare

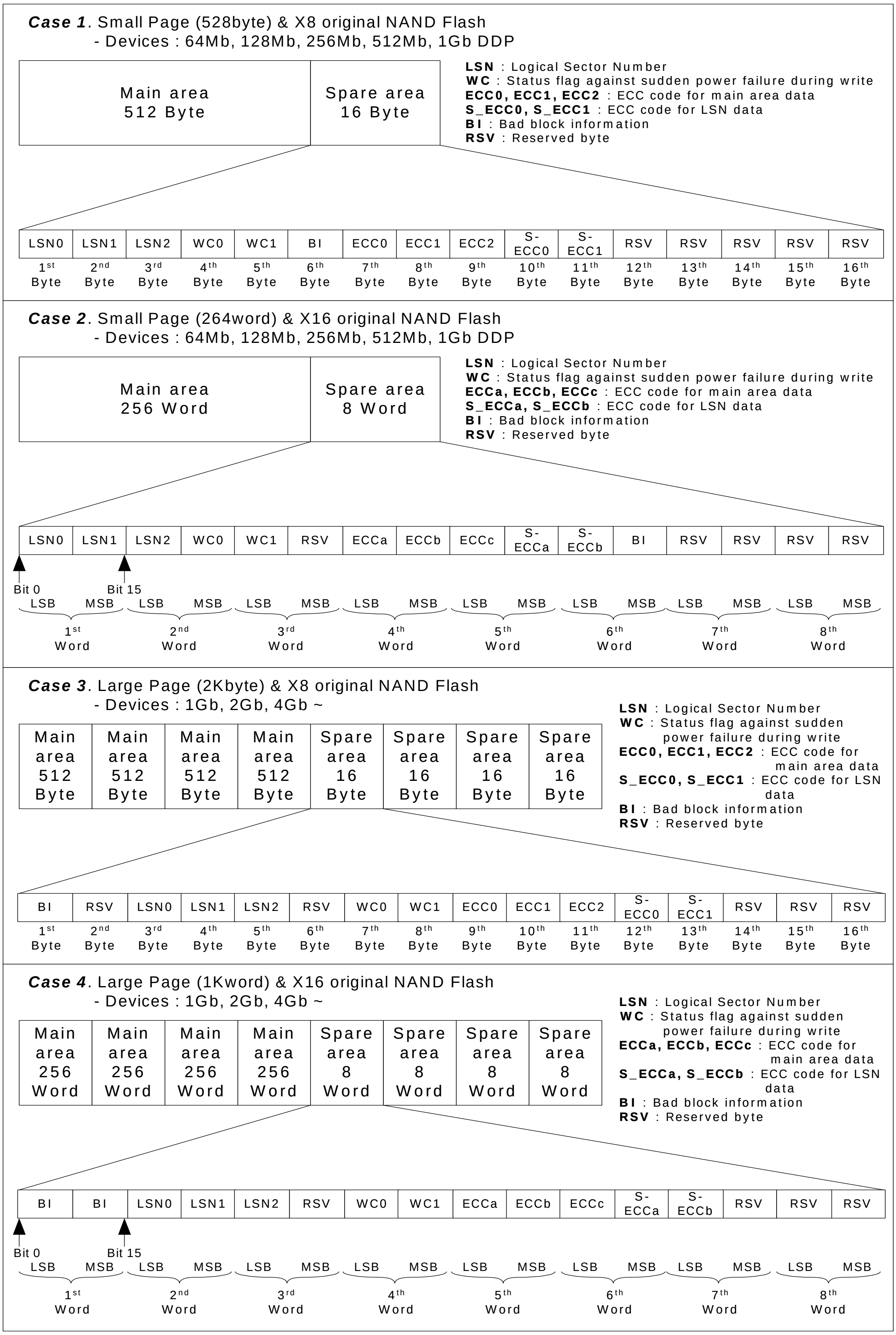

Figure 19. Spare Area Assignment Standard

	1st Byte	2nd Byte	3rd Byte	4th Byte	5th Byte	6th Byte	7th Byte	8th Byte	9th Byte	Nth Byte
Original Data (When writing)	1	2	3	4	5	***6***	7	8	9	?
Changed Data (After reading)	1	2	3	4	5	***7***	7	8	9	?

	Even data of the ECC code	
	Row Address	Column Address
Original ECC code (When writing)	001000011	001
Changed ECC code (After reading)	001000110	001
XOR result (After reading)	000000101 => means 6th byte is fail	000 => means 1st bit is fail

	P2048	P2048'	P1024	P1024'	P512	P512'	P256	P256'	P128	P128'	P64	P64'	P32	P32'	P16	P16'	P8	P8'	P4	P4'	P2	P2'	P1	P1'
Origin al ECC	0	0	0	0	1	1	0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	1	1
New ECC	0	1	0	1	1	0	0	1	0	1	0	1	1	0	1	0	0	1	0	1	0	1	1	0
Result of XOR	0	1	0	1	0	1	0	1	0	1	0	1	1	0	0	1	1	0	0	1	0	1	0	1

The result data of XOR, each pair have alternative data that means correctable error. Because we know fail bit address, can correct fail bit to original data.

Figure 20. The Example of the ECC code compare

NAND_ECC_RESULT_STATUS (0xFC10) : Read Only

7	6	5	4	3	2	1	0
eccA result status		eccB result status		eccC result status		eccD result status	

The ECC process for the NAND flash performs with 512 byte unit. The 512 byte is '1 sector'. The result of the '1 sector' is stored in "eccA result status", "eccB result status", "eccC result status", and "eccD result status". The reason that the register '**NAND_ECC_RESULT_STATUS**' is divided into four parts is to support the advanced NAND flash(AN flash) and is because the AN flash handles the data by maximum 2048 byte at one time.

<In case that writing the data into the NAND flash>

To use ECC module, set write mode to Nand Flash Memory first, and NAND_DMA_MODE_CONTROL[0] has to be set before data transfer to Nand Flash Memory and then The result for ECC of the first '1 sector' data transfer is stored in "NAND_ECCx_RESULT_CODE_HI/MID/LO (ECCx is ECCD,C,B,A)" registers, write this NAND_ECCx_RESULT_CODE_HI/MID/LO value to spare areas of the NAND flash.
The results for ECC are stored into the NAND_ECCA_RESULT_CODE_HI/MID/LO (the first '1 sector'), the NAND_ECCB_RESULT_CODE_HI/MID/LO (the second '1 sector'), NAND_ECCC_RESULT_CODE_HI/MID/LO (the third '1 sector'), and the NAND_ECCD_RESULT_CODE_HI/MID/LO (the fourth '1 sector').

The following result for ECC of the fifth '1 sector' is stored into the register 'ECCD_COD' again, and the previous result is shifted and the first result is erased.

<In case that reading the data in the NAND flash>

Read the first '1 sector' with the ECC code stored when writing, the ECC code is stored into the internal register 'NAND_CURRENT_ECC_CODE_HI/MID/LO'. The new ECC code, which is made at reading the data, is stored into the register 'NAND_ECCx_RESULT_CODE_HI/MID/LO, and then the result of the exclusive OR operation of the two registers is stored into the register 'NAND_RESULT_STATUS[x]'. The case of the second '1 sector' is the same to the first, too, and the ECC code of the NAND_ECCD_RESULT_CODE_HI/MID/LO, which was used to handle the first data is shifted into the NAND_ECCC_RESULT_CODE_HI/MID/LO after the XOR operation and the NAND_ECCD_RESULT_CODE_HI/MID/LO has the second ECC code which read from the NAND flash.

The content of the NAND_RESULT_STATUS[D] is shifted into the NAND_RESULT_STATUS[C] and the result of the second XOR operation is stored into the NAND_RESULT_STATUS[D].

The third and fourth ECC code is performed as above, too, the result is stored into NAND_RESULT_STATUS[A] (the first '1 sector'), NAND_RESULT_STATUS[B] (the second '1 sector'), NAND_RESULT_STATUS[C] (the third '1 sector'), and NAND_RESULT_STATUS[D] (the fourth '1 sector') finally.

After handling the all of four '1 sector', the MCU checks the register 'NAND_RESULT_STATUS' and then see if there is an error. There are four styles of errors.

In case of the original NAND flash, MCU can handle the four sectors or one sector at one time. When MCU handles the one sector at one time, it brings the performance out.

12.2. NAND Flash Control Register

Table 34. NAND Flash Control Register Map (P2 = 0xFC)

Function	Address (Hex)	Type	Reset	Description
NAND_INT_ENABLE	0x00	R/W	0x00	
NAND_INT_STATUS	0x01	R/W	0x00	
NAND_PIN_CONTROL	0x02	R/W	0xFF	
NAND_DATA_PATH	0x03	R/W	-	
NAND_STROBE_SIGNAL_DIV	0x08	R/W	0x00	
NAND_DMA_MODE_CONTROL	0x09	R/W	0x00	
NAND_DMA_START_CONTROL	0x0A	R/W	0x00	
NAND_ECC_RES_STATUS	0x10	RO	-	
NAND_ECCA_RES_CODE_HI	0x14	R/W	-	

NAND_ECCA_RES_CODE_MID	0x15	R/W	-	
NAND_ECCA_RES_CODE_LO	0x16	R/W	-	
NAND_ECCB_RES_CODE_HI	0x17	R/W	-	
NAND_ECCB_RES_CODE_MID	0x18	R/W	-	
NAND_ECCB_RES_CODE_LO	0x19	R/W	-	
NAND_ECCC_RES_CODE_HI	0x1A	R/W	-	
NAND_ECCC_RES_CODE_MID	0x1B	R/W	-	
NAND_ECCC_RES_CODE_LO	0x1C	R/W	-	
NAND_ECCD_RES_CODE_HI	0x1D	R/W	-	
NAND_ECCD_RES_CODE_MID	0x1E	R/W	-	
NAND_ECCD_RES_CODE_LO	0x1F	R/W	-	

Nand Flash Interrupt Enable (NAND_INT_ENABLE, 0xFC00) : Read / Write

7	6	5	4	3	2	1	0
Reserved							ST_EN

If enabled, interrupt is generated when DMA stops after transfer completion.

ST_EN :

0 : interrupt generation disable.
1 : interrupt generation enable.

Nand Flash Interrupt Status (NAND_INT_STATUS, 0xFC01) : Read / Write

7	6	5	4	3	2	1	0
Reserved							ST_STA

If 1 is read, it means interrupt is generated by DMA stop after transfer completion. Writing 1 clears status of interrupt.

ST_STA :

0 : interrupt condition dont occur.
1 : interrupt condition occurred.

Nand Flash Pin Control (NAND_PIN_CONTROL, 0xFC02) : Read / Write

7	6	5	4	3	2	1	0
FCEN3	FCEN2	FCEN1	FCEN0	FRNB	FALE	FCLE	Reserved

Writing each bit with 0 or 1 makes the data outputted to the corresponding external pin. MCU writes command or address of Nand flash memory through this register and address 0xFC03 register. And then, use DMA to read or write continuous much data from / to nand flash memory.

FCEN3 : external FCEN3 pin value

0 : drives external FCEN3 pin as 0.
1 : drives external FCEN3 pin as 1.

FCEN2 : external FCEN2 pin value

0 : drives external FCEN2 pin as 0.
1 : drives external FCEN2 pin as 1.

FCEN1 : external FCEN1 pin value

0 : drives external FCEN1 pin as 0.
1 : drives external FCEN1 pin as 1.

FCEN0 : external FCEN0 pin value

0 : drives external FCEN0 pin as 0.
1 : drives external FCEN0 pin as 1.

FRNB : external FRNB pin value

0 : drives external FRNB pin as 0.
1 : drives external FRNB pin as 1.

FALE : external FALE pin value

0 : drives external FALE pin as 0.
1 : drives external FALE pin as 1.

FCLE : external FCLE pin value

0 : drives external FCLE pin as 0.
1 : drives external FCLE pin as 1.

Nand Flash data Control (NAND_ DATA_PATH, 0xFC03) : Read / Write

7	6	5	4	3	2	1	0
WR_PATH							

8bit nand data(FD7-FD0) can be written/read to/from this register.

Nand Flash Strobe Signal Divider (NAND_STROBE_SIGNAL_DIV, 0xFC08) : Read / Write

7	6	5	4	3	2	1	0
LO_STB_DIV				HI_STB_DIV			

This register controls clock speed of F_REN and F_WEN pins (read and write enable pin of nand flash memory.)

LO_STB_DIV : Value written is one less than clock number to keep low state(period of low state).
HI_STB_DIV : Value written is one less than clock number to keep high state(period of high state).

For example, write 3 to [7:4] to keep low state during 4 IDE clock and write 2 to [3:0] to keep high state during 3 IDE clock. These have effect on F_REN pin and F_WEN pin.

Nand Flash DMA Mode Control (NAND_DMA_MODE_CONTROL, 0xFC09) : Read / Write

7	6	5	4	3	2	1	0
Reserved		FRNB0_ST	FRNB1_ST	FRNB0_EN	FRNB1_EN	ERR_ADDR	ECC_CAL

This register is used for DMA operation. ERR_ADDR bit ha no effect on ECC operation and only shows address of error bit found or CRC remainder according to ECC operation result in next address 0xFC14 ~ 0xFC1F register. If remainder mode is used, MCU should calculates position of error bit.

FRNB0_ST : If FRNB pin is 0, start low strobe.

FRNB1_ST : If FRNB pin is 1, start low strobe.

FRNB0_EN : If FRNB pin is 0, end low strobe.

FRNB1_EN : If FRNB pin is 1, end low strobe.

ERR_ADDR :

0 : Shows CRC remainder of ECC operation result.
1 : Shows error address after ECC operation result.

ECC_CAL :

0 : Disable.
1 : Makes ECC calculation be done.

Nand Flash DMA Start Control (NAND_DMA_START_CONTROL, 0xFC0A) : Read / Write

7	6	5	4	3	2	1	0
Reserved		RD_ST	WR_ST	FCEN3_VL	FCEN2_VL	FCEN1_VL	FCEN0_VL

RD_ST : start Nand flash read and write DMA buffer.

WR_ST : start read DMA buffer and write NAND flash.

FCEN3_VL : indicate value that will be assigned to FCEN3 pin after DMA operation stops.

FCEN2_VL : indicate value that will be assigned to FCEN2 pin after DMA operation stops.

FCEN1_VL : indicate value that will be assigned to FCEN1 pin after DMA operation stops.

FCEN0_VL : indicate value that will be assigned to FCEN0 pin after DMA operation stops.

Nand Flash ECC result status (NAND_ECC_RES_STATUS, 0xFC10) : Read Only

7	6	5	4	3	2	1	0
ECCA_STA		ECCB_STA		ECCC_STA		ECCD_STA	

It is possible to do continuous 4 times ECC calculation and save the results to 4 different places(ECCA, ECCB, ECCC, ECCD). Each status code means :

00 : ECC success.
01 : Correctable error.
10 : ECC error.
11 : Uncorrectable error.

Nand Flash ECCA Result Code High (NAND_ECCA_RES_CODE_HI, 0xFC14) : Read / Write

7	6	5	4	3	2	1	0
ECCA_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCA Result Code Middle (NAND_ECCA_RES_CODE_MID, 0xFC15) : Read / Write

7	6	5	4	3	2	1	0
ECCA_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCA Result Code Low (NAND_ECCA_RES_CODE_LO, 0xFC16) : Read / Write

7	6	5	4	3	2	1	0
ECCA_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCB Result Code High (NAND_ECCB_RES_CODE_HI, 0xFC117) : Read / Write

7	6	5	4	3	2	1	0

ECCB_RES_COD

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCB Result Code Middle (NAND_ECCB_RES_CODE_MID, 0xFC18) : Read / Write

7	6	5	4	3	2	1	0
ECCB_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCB Result Code Low (NAND_ECCB_RES_CODE_LO, 0xFC19) : Read / Write

7	6	5	4	3	2	1	0
ECCB_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCC Result Code High (NAND_ECCC_RES_CODE_HI, 0xFC1A) : Read / Write

7	6	5	4	3	2	1	0
ECCC_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCC Result Code Middle (NAND_ECCC_RES_CODE_MID, 0xFC1B) : Read / Write

7	6	5	4	3	2	1	0
ECCC_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCC Result Code Low (NAND_ECCC_RES_CODE_LO, 0xFC1C) : Read / Write

7	6	5	4	3	2	1	0
ECCC_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCD Result Code High (NAND_ECCD_RES_CODE_HI, 0xFC1D) : Read / Write

7	6	5	4	3	2	1	0
ECCD_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCD Result Code Middle (NAND_ECCD_RES_CODE_MID, 0xFC1E) : Read / Write

7	6	5	4	3	2	1	0
ECCD_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

Nand Flash ECCD Result Code Low (NAND_ECCD_RES_CODE_LO, 0xFC1F) : Read / Write

7	6	5	4	3	2	1	0
ECCD_RES_COD							

24 bit ECC result for 512 bytes data block is saved by ECC module.

13. DRM Controller

13. DRM Controller

The DRM Controller is a 20 bytes length calculation processor. This is used for DRM cryptographic decryption calculation but can be used for general calculation which is difficult to calculate with 8051 MCU.

13.1. DRM Operation

The possible calculation is addition, subtraction and multiplication. Addition and subtraction requires 12 clocks and multiplication requires 340 clocks.

13.2. DRM Control Register

Table 35. DRM Control Register Map (P2 = 0xF9)

Function	Address (Hex)	Type	Reset	Description
DRM_DATA_REG	0x00 ~0x13	R/W	-	20Byte DRM data register.
DRM_DATA_REG_BANK_SEL	0x18	R/W	-	DRM data register bank select.
DRM_INT_EN	0x19	R/W	-	DRM interrupt enable.
DRM_INT_STA	0x1a	R/W	-	DRM interrupt status.
DRM_CMD_REG	0x1b	R/W	0x00	DRM command control register.

20Byte DRM Data Register (DRM_DATA_REG, 0xF900~0xF913) : Read / Write

7	6	5	4	3	2	1	0
DRM_DAT							

P, Y, M, X can be read or written through this register according to address 0xF914 register. Use little endian format.

DRM Data Register Bank Select (DRM_DATA_REG_BANK_SEL, 0xF918) : Read / Write

7	6	5	4	3	2	1	0
Reserved					BIG_END	BANK_SEL	

Selects one of P, Y, M or X register with this register and then read or write the 20-bit register selected through "DRM_DATA_REG(0xF900~0xF913)".

BIG_END : Select endian format with which register data will be read or written.

0 : little endian format.
1 : big endian format.

BANK_SEL

00 : Selects 20Bytes P register
01 : Selects 20Bytes Y register
10 : Selects 20Bytes M register
11 : Selects 20Bytes X register

DRM Interrupt Enable (DRM_INT_EN, 0xF919) : Read / Write

7	6	5	4	3	2	1	0
Reserved							DRM_INT

If calculation is completed, interrupt is generated and this bit becomes 1 to indicate interrupt generation. Writing 1 clears this bit.

DRM_INT : This bit is used to DRM interrupt enable.

0 : Disable.
1 : Enable.

DRM Interrupt Status (DRM_INT_STA, 0xF91a) : Read / Write

7	6	5	4	3	2	1	0
Reserved							INT_STA

If calculation is completed, interrupt is generated and this bit becomes 1 to indicate interrupt generation. Writing 1 clears this bit.

INT_STA : This bit is used to check the DRM interrupt.

0 : No interrupt is generated.
1 : Interrupt is generated.

DRM Command Register (DRM_CMD_REG, 0xF91b) : Read / Write

7	6	5	4	3	2	1	0
WR_EN	RD_EN	ATM_COD		DAT_TAG_SEL		DAT_SRC_SEL	

This register is cleared at the end of calculation. DMA data transfer is done by predefined format.

WR_EN : This bit is used to DMA data register write enable.

0 : no operation.
1 : before calculation, register data are written by DMA operation.

RD_EN : This bit is used to DMA data register read enable.

0 : no operation.
1 : after calculation, register data are read by DMA operation.

ATM_COD : These bits are used to arithmetic code mode select.

00 : No operation.
01 : Modular addition.
10 : Modular subtract.
11 : modular multiplication.

No operation : No Change.

Modular addition : P and Y is added and then save result to P and X register. No change in Y and M register.

Modular subtract : P minus Y is done and then save result to P and X register. No change in Y and M register.

Modular multiplication : multiply X and Y and then save result to P and X register.

DAT_TAG_SEL : These bits are used to select DMA target register.

00 : P
01 : Y
10 : M
11 : X

DAT_SRC_SEL : These bits are used to select DMA source register.

00 : P
01 : Y
10 : M
11 : X

* if register bit [7:4] is zero, contents selected by bit [3:2] are copied to register selected by bit [1:0].

14. Hardwired Audio / Voice Engine

14. Hardwired Audio / Voice Engine

14.1. Overview

MP3 CODEC and WMA decoder are based on a hardwired structure, which offers ultra-low-power consumption for portable products. The core operates at 1.8V and has not only MP3 CODEC and WMA decoder function but also a lot of other functions such as digital volume control, bass/treble control, mute, bit rate control for encoding, mono/stereo control, 10 band equalizer, SRS WOW 3D Sound effect and etc. It offers a power-down mode and a test mode controlled by MCU.

The encoder accepts data with I^2S format from an external audio CODEC for encoding in MP3 format and the decoder gives an external audio CODEC the data with I^2S format from itself as 16 bit PCM data.

Its core offers the voice recording/playback functions that have a new algorithm to record voice in high compression and quality (8, 16, 24, 32, ~ 320Kbps).

MP3 Encoder

The encoder operates in real-time with the ultra-low-power and the high quality, and can compress data in various bit rates. It gives two kinds of encoding methods such as audio and voice.

Audio (Music)

- Acceptable input data :
  - 32, 44.1 and 48 KHz (16-bit stereo data from ADC)
  - Data from wave file

- Data compression rate (MPEG1 Layer 3) :
  - 32, 40, 48, 56, 64, 80, 96, 112, 128, 160, 192, 224, 256 and 320 Kbps
  - stereo/joint-stereo/mono/dual encoded data output

Voice

- Acceptable input data : 16 KHz 16-bit mono data from ADC
  - Data from wave file

- Data compression rate (MPEG2 Layer 3)
  - 8, 16, 24, 32, 40, 48, 56, 64, 80, 96, 112, 128, 144 and 160 Kbps
  - mono/stereo encoded data output

MP3 / WMA (Windows Media Audio) Decoder

The MP3 decoder operates in MPEG1/2/2.5 layer3 bit stream with the ultra-low-power and offers a lot of useful functions for audio applications such as digital volume/bass/treble control and serial DAC interface. Also, this SoC implements WMA decoding as a hardwired logic for ultra-low-power operation.

MP3

- **Audio (Music)**
  MPEG1 layer3, MPEG2 layer3 bit stream input
- **Voice**
  MPEG1 layer3, MPEG2 layer3 bit stream input

WMA

- WMA decoding
- WMA header information extraction

Audio Effects

- **Digital volume / bass / treble control**

- **10 band graphic equalizer**

Equalizer has 10 band volume control from +16dB to -15dB range for each band and each band audio power can be read. Frequency bandwidth of each band is one tenth of half sampling rate frequency.
The digital volume has 4 adjustable values that can be changed by 1 db unit in –111 to +16db range on the left and right channel.
The bass and the treble have one value that can be changed by 1 db unit in –15 to +15db range.
All adjusted data go to the DAC input after changing their values.

14.2. MP3 Encoding Control Register

There is a CODEC interface block between the hardwired audio engine and the external audio CODEC. The CODEC interface block has some functions such as the I2S mode setting, the gain control, the bypass, the volume control and the audio CODEC data buffer.

Table 36. MP3 Encoding Control Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
MP3_INT_ENABLE	0x00	R/W	0x00	
MP3_INT_STATUS	0x01	R/W	-	
MP3_SOFT_RESET	0x02	WO	0xFF	
MP3_PCM_FREQ_CONTROL	0x05	RO[7:4] RW[3:0]	0xCC	
MP3_OPERATION_CONTROL	0x08	R/W	0x00	
MP3_CODEC_DAT_REQ_STATUS	0x0A	RO	0x00	
MP3_CODEC_COMP_DATA	0x0B	R/W	0x00	
MP3_DEC_BUF_STATUS	0x0C	RO	0x00	
MP3_FRAME_LENGTH_MODE	0x0D	R/W	0x00	

MP3 Interrupt Enable (MP3_INT_ENABLE, 0xFA00) : Read / Write

7	6	5	4	3	2	1	0
APU	ACCC	ACDC	ACWU	EMFS	EMFE	EDBE	EDFT

Writing 0 disables interrupt of the bit. Writing 1 to each bit makes corresponding interrupt enable. If interrupt enable bit is enabled and the condition of the bit enabled is occurred, corresponding “interrupt status(0xFA01) bit is set and interrupt signal goes to MCU via interrupt control of system control block.

APU : enables interrupt that indicates update of graphical equalizer power information
ACCC : enables interrupt that indicates the change of audio codec channel
ACDC : enables interrupt that indicates update of audio codec data
ACWU : enables interrupt that indicates the use of written audio codec data

EMFS : enables interrupt that indicate the start of mute frame by mp3 encoder
EMFE : enables interrupt that indicate the end of mute frame by mp3 encoder

EDBE : enables interrupt that indicates the error status of buffer control
EDFT : enables interrupt that indicates the change of frame by mp3 encoder/decoder

MP3 Interrupt Status (MP3_INT_STATUS, 0xFA01) : Read / Write

7	6	5	4	3	2	1	0
APU	ACCC	ACDC	ACWU	EMFS	EMFE	EDBE	EDFT

If interrupt is generated, corresponding bit is set to 1 and writing 1 to the bit clears 1 to 0.

APU : indicates update of graphical equalizer power information
ACCC : indicates the change of audio codec channel
ACDC : indicates update of audio codec data
ACWU : indicates the use of written audio codec data
EMFS : indicate the start of mute frame by mp3 encoder
EMFE : indicate the end of mute frame by mp3 encoder
EDBE : indicates the error status of buffer control
EDFT : indicates the change of frame by mp3 encoder/decoder

MP3 Software Reset (MP3_SOFT_RESET, 0xFA02) : Write Only

7	6	5	4	3	2	1	0
Reserved			REG_RST	COM_RST	ENC_RST	DEC_RST	WMA_RST

Writing 0 to each bit makes the corresponding part reset and initialized by default.

REG_RST : all register block
COM_RST : common functional block
ENC_RST : mp3 encoder functional block
DEC_RST : mp3 decoder functional block
WMA_RST : wma decoder functional block

MP3 PCM frequency control (MP3_PCM_FREQ_CONTROL, 0xFA05) : RO[7:4] / RW[3:0]

7	6	5	4	3	2	1	0
OPR_SAM_FRQ				PCM_SAM_FRQ			

OPR_SAM_FRQ : By reading this register, user can know current audio operating frequency and sampling frequency.

PCM_SAM_FRQ : Shows sampling frequency at PCM mode.

0 : 11.025kHz

1 : 12.000kHz

2 : 8.000kHz

3 : 8.000kHz

4 : 11.025kHz

5 : 12.000kHz

6 : 8.0000kHz

7 : 8.000kHz

8 : 22.050kHz

9 : 24.000kHz

10 : 16.000kHz

11 : 16.000kHz

12 : 44.100kHz

13 : 48.000kHz

14 : 32.000kHz

15 : 32.000kHz

MP3 Operation control (MP3_OPERATION_CONTROL, 0xFA08) : Read / Write

7	6	5	4	3	2	1	0
MONO	BYTE	DMA_WE	DMA_RE	ADC_WE	ADC_RE	COD_MD	

MONO : indicates mono at PCM mode
BYTE : indicates 8bit data at PCM mode, otherwise 16bit mode
DMA_WE : indicates the writing of pcm data by dma block to pcm buffer
DMA_RE : indicates the reading of pcm data by dma block from pcm buffer
ADC_WE : indicates the writing of pcm data by audio codec to pcm buffer
ADC_RE : indicates the reading of pcm data by audio codec from pcm_buffer
COD_MD : indicates the codec mode
0 : pcm mode
1 : mp3 encoder mode
2 : mp3 decoder mode
3 : wma decoder mode

MP3 CODEC Data Request Status (MP3_CODEC_DAT_REQ_STATUS, 0xFA0A) : Read Only

7	6	5	4	3	2	1	0
Reserved	WHD	PDDR	PEDR	reserved	MEDR	MDDR	WDDR

WHD : wma header detected
PDDR : pcm buffer write data request
PEDR : pcm buffer read data request
reserved : 0
MEDR : mp3 encoder bitstream read request
MDDR : mp3 decoder bitstream write request
WDDR : wma decoder bitstream write request

MP3 CODEC Compressed Data (MP3_CODEC_COMP_DATA, 0xFA0B) : Read / Write

7	6	5	4	3	2	1	0
MCCD							

Bitstream data can be transferred by writing or reading this register.

MP3 Decoder Buffer Status (MP3_DEC_BUF_STATUS, 0xFA0C) : Read Only

7	6	5	4	3	2	1	0
Reserved				WDBF	WDBE	MDBF	MDBE

WDBF : wma decoder bit stream buffer full
WDBE : wma decoder bit stream buffer empty
MDBF : mp3 decoder bit stream buffer full
MDBE : mp3 decoder bit stream buffer empty

MP3 Decoder Sync Mode (MP3_FRAME_LENGTH_MODE, 0xFA0D) : Read / Write

7	6	5	4	3	2	1	0
Reserved						MFLM	

MFLM : Indicates the method of initializing the table used in decoding mp3 files.
0 : no action
1 : free format table initialization

2 : all format table initialization
3 : all format table calcaulation

14.3. Audio CODEC Interface

The audio CODEC interface is controlled by the I2S communication and it has the interface as shown in Figure 21. and it shows the interface of 256Fs, 16bit data, left low and MSB first.

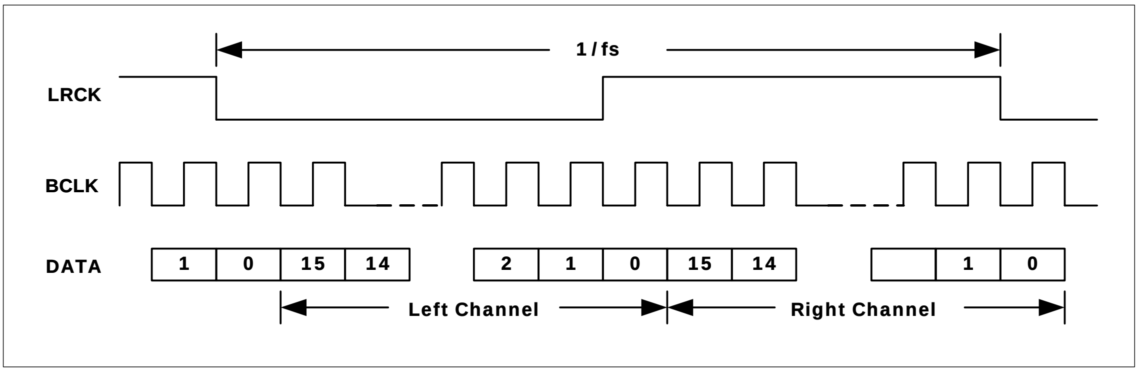

Figure 21. 256Fs I2S timing

Table 37. Audio CODEC Control Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
AUD_CODEC_MODE	0x10	R/W	0x00	
AUD_CODEC_CONTROL	0x11	R/W	0x00	
AUD_CODEC_STATUS	0x12	RO	-	
AUD_CODEC_DATA_LO	0x13	R/W	0x00	
AUD_CODEC_DATA_HI	0x14	R/W	0x00	

Audio CODEC Mode (AUD_CODEC_MODE, 0xFA10) : Read / Write

7	6	5	4	3	2	1	0
Reserved						CD_JUST	CHAN_INV

Default(0x00) is I2S format. Use one of 0x00, 0x01 or 0x02 values.

CD_JUST :
Indicates that channel clock is justified by codec data

CHAN_INV :
Indicates that channel clock is inverted.

Audio CODEC Control (AUD_CODEC_CONTROL, 0xFA11) : Read / Write

7	6	5	4	3	2	1	0
Reserved		CPEN	MAEN	ACEN	ACBP	ACMT	ACPA

CPEN : enables the function of preventing clipping
MAEN : enables the use of 8bit ADC data as audio
ACEN : enables the audio clock generation
ACBP : enables the function of bypassing audio data

ACMT : enables the function of muting audio
ACPA : enables the function of pausing audio

Audio CODEC Status (AUD_CODEC_STATUS, 0xFA12) : Read Only

7	6	5	4	3	2	1	0
Reserved			PBE	PBNF	PBF	HDDWE	ACCS

PBE : pcm buffer empty
PBNF : pcm buffer near full
PBF : pcm buffer full
HDDWE : indicates the timing of writing audio DAC data
ACCS : indicates the channel state of audio codec

Audio CODEC Data Low (AUD_CODEC_DATA_LO, 0xFA13) : Read / Write

7	6	5	4	3	2	1	0
ACDL							

These register is used by MCU to write or read data directly to I2S when address 0xFA12 register bit[1] is 1.

Audio CODEC Data High (AUD_CODEC_DATA_HI, 0xFA14) : Read / Write

7	6	5	4	3	2	1	0
ACDH							

14.5. MPEG1/2 Encoder Control

Table 39. SRS / WOW Control Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
ENC_SELECT	0x20	R/W	0x03	
ENC_MPEG1_CONTROL	0x21	R/W	0x19	
ENC_MPEG2_CONTROL	0x22	R/W	0xB2	

Encoder1/2 Select (ENC_SELECT, 0xFA20) : Read / Write

7	6	5	4	3	2	1	0
Reserved						Reserved1	MD_SEL

Reserved 1 : Must be 1
MD_SEL : indicates the mode of mp3 encoding
1 : mpeg1
0 : mpeg2

Encoder MPEG1 Control (ENC_MPEG1_CONTROL, 0xFA21) : Read / Write

7	6	5	4	3	2	1	0
SPL_FREQ		MD_SEL		BITR			

SPL_FREQ :

00 : 44.1KHz
01 : 48KHz
10 : 32KHz
11 : Reserved

MD_SEL :
00 : Stereo
01 : Joint stereo
10 : Dual channel
11 : Single channel

BITR :
0000 : Reserved
0001 : 32 Kbps
0010 : 40 Kbps
0011 : 48 Kbps
0100 : 56 Kbps
0101 : 64 Kbps
0110 : 80 Kbps
0111 : 96 Kbps
1000 : 112 Kbps
1001 : 128 Kbps
1010 : 160 Kbps
1011 : 192 Kbps
1100 : 224 Kbps
1101 : 256 Kbps
1110 : 320 Kbps
1111 : Reserved

Encoder MPEG2 Control (ENC_MPEG2_CONTROL, 0xFA22) : Read / Write

7	6	5	4	3	2	1	0
SPL_FREQ		MD_SEL		BITR			

SPL_FREQ :

00 : 22.05KHz
01 : 24KHz
10 : 16KHz
11 : Reserved

MD_SEL :
00 : Stereo
01 : Joint stereo
10 : Dual channel
11 : Single channel

BITR :
0000 : Reserved
0001 : 8 Kbps
0010 : 16 Kbps
0011 : 24 Kbps
0100 : 32 Kbps
0101 : 40 Kbps
0110 : 48 Kbps
0111 : 56 Kbps
1000 : 64 Kbps
1001 : 80 Kbps
1010 : 96 Kbps
1011 : 112 Kbps
1100 : 128 Kbps
1101 : 144 Kbps
1110 : 160 Kbps
1111 : Reserved

14.6. Filter

NX5850 has the low pass filter and high pass filter to improve the sound when the MP3 encoding. The low pass filter passes the lower frequency than the threshold defined, and the high pass filter passes the higher frequency than the threshold defined.

The way for taking the threshold is as following. If the sampling frequency for the encoding is 2f and the frequency of the low pass filter is F_{LP}, 'f' is 0x240 of the maximum value of the threshold.

Threshold value = 0x240 * F_{LP} / f

For example, under the condition that encoding with 16 KHz sampling frequency if you'd like to remove the sound of over 4.5 KHz,

MP_ELPF = 570 * 4500 / 8000 = 0x140.

The threshold of the high pass filter can take in the same way, too. If you want not to use the filter, the threshold value defined in the chip can be used.

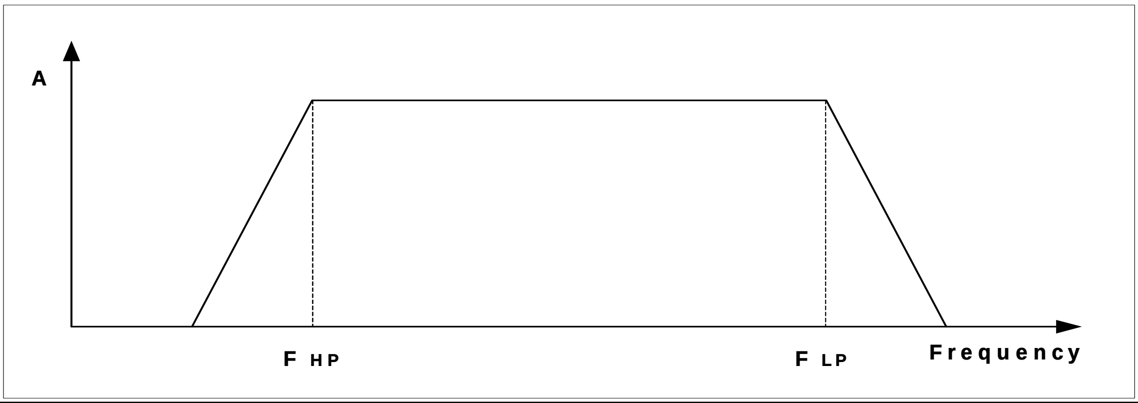

Figure 22. The pass band of Filter

The following code shows the operation that encoding with 16KHz sampling frequency at the low pass filter of 4.5KHz and the high pass filter of 200Hz.

Program 9. The encoding with low pass filter and high pass filter.

```
if( FILTER_USE ) {
    // Low Pass Filter Threshold value   about 4.5Khz
    write_XDATA(MP3+LOWFIL_LO_REG,0x40);
    write_XDATA(MP3+LOWFIL_HI_REG,0x01);

    // High Pass Filter Threshold value   about 200hz
    write_XDATA(MP3+HIGHFIL_LO_REG,0x0d);
    write_XDATA(MP3+HIGHFIL_HI_REG,0x00);
}
else {
    write_XDATA(MP3+LOWFIL_LO_REG,0x00);
    write_XDATA(MP3+LOWFIL_HI_REG,0x00);
    write_XDATA(MP3+HIGHFIL_LO_REG,0x00);
    write_XDATA(MP3+HIGHFIL_HI_REG,0x00);
}
```

Table 40. Filter Control Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
ENC_LP_FILTER_THRE_LO	0x23	R/W	0x00	
ENC_LP_FILTER_THRE_HI	0x24	R/W	0x00	
ENC_HP_FILTER_THER_LO	0x25	R/W	0x00	
ENC_HP_FILTER_THER_HI	0x26	R/W	0x00	

Encoder Low Pass Filter Threshold Low (ENC_LP_FILTER_THRE_LO, 0xFA23) : Read / Write

7	6	5	4	3	2	1	0
ELFTL							

ELFTL :

Encoder Low Pass Filter Threshold High (ENC_LP_FILTER_THRE_HI, 0xFA24) : Read / Write

7	6	5	4	3	2	1	0
ELFTH							

ELFTH :

Encoder High Pass Filter Threshold Low (ENC_HP_FILTER_THRE_LO, 0xFA25) : Read / Write

7	6	5	4	3	2	1	0
EHFTL							

EHFTL :

Encoder High Pass Filter Threshold High (ENC_HP_FILTER_THRE_HI, 0xFA26) : Read / Write

7	6	5	4	3	2	1	0

EHFTH

EHFTH :

14.7. Mute Detect Function

NX5850 has a mute detect function. It is used to detect the no sound during the encoding. The encoder is in working although there is the mute detection. The data record under no sound is decided by the firmware. The mute detection is performed with the mute power threshold register 'ENC_MUTE_POW_THRE_LO/MID/HI (0xfa28, 0xfa29, 0xfa2a)' for recognizing the mute level, and the encoder mute controls register 'ENC_MUTE_CONTROL (0xfa27)' after giving the minimum time (unit: frame) for recognizing the mute to the encoder mute detection frame register 'ENC_MUTE_FRAME_NUM_LO /HI(0xfa2f,0xfa30)'. If the input signal is smaller than the signal level of the MP3_ENC_MUTE_POWER_ THRESHOLD_LO/MID/HI, the encoder mute frame number.

register ENC_MUTE_FRAME_NUM_LO/HI' is increased. If the value of the ENC_MUTE_FRAME_NUM_LO/HI is the same the mute power threshold of the ENC_MUTE_POW_THRE_LO/MID/HI, the mute is began and a interrupt is occurred to notice this. If the input signal is not higher than the signal level of the MP3_ENC_MUTE_POW_THRE_LO/MID/HI, the MP3_ENC_MUTE_FRAME_NUM_LO/HI keeps the same value. If the input signal is higher than the mute power threshold of the ENC_MUTE_POW_THRE_LO/MID/HI, the MP3_ENC_MUTE_FRAME_NUMBER_LO/HI goes to low.

The following code shows the mute detection of the Polling method, not interrupt method.

Program 10. The Mute Detection

```
//MUTE DETECT sec define
#define VOR_1SEC          25
#define VOR_2SEC          50
#define VOR_3SEC          75
#define VOR_4SEC          100

void MuteDetect(UINT8 on_flag, UINT8 level, UINT8 sec)
{
    if(on_flag) { //mute detect enable
        write_XDATA(POW_THRSHL, level & 0xff);        //level midle value
        write_XDATA(POW_THRSHM, (level>>8) & 0xff);
        write_XDATA(POW_THRSHH, (level>>16) & 0x07);
        write_XDATA(MUTE_INT_FL, sec);     //define a number of mute frame의 (75 //frames 3sec)
        write_XDATA(MUTE_INT_FH, 0x00);
        write_XDATA(MUTE_DET_EN, 0x01);    //mute detect enable
    } else { //mute detect disable
        write_XDATA(POW_THRSHL, 0x00);
        write_XDATA(POW_THRSHM, 0x00);
        write_XDATA(POW_THRSHH, 0x00);
        write_XDATA(MUTE_INT_FL, 0x00);
        write_XDATA(MUTE_INT_FH, 0x00);
        write_XDATA(MUTE_DET_EN, 0x00);    //mute detect disable
    }
}

// Enable the MUTE_DETECT function

MuteDetect ( 1, 0x21ff, VOR_3SEC);

while(1){
    vor_status = read_XDATA(MUTE_FRAME_NUMBERL);//read mute frame count

    if(vor_status == VOR_3SEC)             //null frame count
        Lcd_Text(Text_NO_SIGNAL);
    else
        break;
}
```

Table 41. Mute Detection Control Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
ENC_MUTE_CONTROL	0x27	R/W	0x00	
ENC_MUTE_POW_THRE_LO	0x28	R/W	0x00	
ENC_MUTE_POW_THRE_MID	0x29	R/W	0x00	
ENC_MUTE_POW_THRE_HI	0x2A	R/W	0x00	
ENC_MUTE_DETECT_FRM_LO	0x2B	R/W	0x60	
ENC_MUTE_DETECT_FRM_HI	0x2C	R/W	0x00	
ENC_FRAME_NUMBER_LO	0x2D	RO	0x00	
ENC_FRAME_NUMBER_HI	0x2E	RO	0x00	
ENC_MUTE_FRAME_NUM_LO	0x2F	RO	0x00	
ENC_MUTE_FRAME_NUM_HI	0x30	RO	0x00	

MP3 Encoder Mute Control (ENC_MUTE_CONTROL, 0xFA27) : Read / Write

7	6	5	4	3	2	1	0
Reserved							MFDE

This register is used to set the MP3 encoder mute.

EMFDE[7:1] : Reserved.

MFDE : This bit is used to set the mute frame detection enable.

0 : Disable.
1 : Enable

MP3 Encoder Mute Power Threshold Lower 8-bit (ENC_MUTE_POW_THRE_LO, 0xFA28) : Read / Write

7	6	5	4	3	2	1	0
EMPTL							

This register is used to set the lower 8-bit of the MP3 encoder mute power threshold (MP_EMPT[18:0]).

EMPTL[7:0] : MP_EMPT[7:0].

MP3 Encoder Mute Power Threshold Middle 8-bit (ENC_MUTE_POW_THRE_MID, 0xFA29) : Read / Write

7	6	5	4	3	2	1	0
EMPTM							

This register is used to set the middle 8-bit of the MP3 encoder mute power threshold (MP_EMPT[18:0]).

EMPTM[7:0] : MP_EMPT[15:8]

MP3 Encoder Mute Power Threshold Upper 3-bit (ENC_MUTE_POW_THRE_HI, 0xFA2A) : Read / Write

7	6	5	4	3	2	1	0

Reserved	EMPTH

This register is used to set the upper 3-bit of the MP3 encoder mute power threshold (MP_EMPT[18:0]).

EMPTH[2:0] : MP_EMPT[18:16]

MP3 Encoder Mute Detection Frame Lower 8-bit (ENC_MUTE_DETECT_FRM_LO, 0xFA2B) : Read/Write

7	6	5	4	3	2	1	0
EMDFL							

This register is used to set the lower 8-bit of the MP3 encoder mute detect frame (MP_EMDF[15:0]).

EMDFL[7:0] : MP_EMDF[7:0].

MP3 Encoder Mute Detection Frame Upper 8-bit (ENC_MUTE_DETECT_FRM_HI, 0xFA2C) : Read/Write

7	6	5	4	3	2	1	0
EMDFH							

This register is used to set the upper 8-bit of the MP3 encoder mute detect frame (MP_EMDF[15:0]).

EMDFH[7:0] : MP_EMDF[15:8].

MP3 Encoder Mute Frame Lower 8-bit (ENC_FRAME_NUMBER_LO, 0xFA2D) : Read Only

7	6	5	4	3	2	1	0
EFNL							

EFNL :

MP3 Encoder Mute Frame Upper 8-bit (ENC_FRAME_NUMBER_HI, 0xFA2E) : Read Only

7	6	5	4	3	2	1	0
EFNH							

EFNH :

MP3 Encoder Mute Frame Lower 8-bit (ENC_MUTE_FRAME_NUM_LO, 0xFA2F) : Read Only

7	6	5	4	3	2	1	0
EMFNL							

This register is used to set the lower 8-bit of the MP3 encoder mute frame number (MP_EMFN[15:0]).

EMFNL[7:0] : MP_EMFN[7:0].

MP3 Encoder Mute Frame Upper 8-bit (ENC_MUTE_FRAME_NUM_HI, 0xFA30) : Read Only

7	6	5	4	3	2	1	0
EMFNH							

This register is used to set the upper 8-bit of the MP3 encoder mute frame number (MP_EMFN[15:0]).

EMFNH[7:0] : MP_EMFN[15:8]

14.8. Sound Effect Control

Table 42. Sound Effect Control Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
SOUND_EFFECT_CONTROL	0x31	R/W	0x00	
PRESCALE_GAIN_INDEX	0x32	R/W	0x00	
EQ_BAND0_GAIN_INDEX	0x33	R/W	0x00	
EQ_BAND1_GAIN_INDEX	0x34	R/W	0x00	
EQ_BAND2_GAIN_INDEX	0x35	R/W	0x00	
EQ_BAND3_GAIN_INDEX	0x36	R/W	0x00	
EQ_BAND4_GAIN_INDEX	0x37	R/W	0x00	
EQ_BAND5_GAIN_INDEX	0x38	R/W	0x00	
EQ_BAND6_GAIN_INDEX	0x39	R/W	0x00	
EQ_BAND7_GAIN_INDEX	0x3A	R/W	0x00	
EQ_BAND8_GAIN_INDEX	0x3B	R/W	0x00	
EQ_BAND9_GAIN_INDEX	0x3C	R/W	0x00	
TONE_BASS_GAIN_INDEX	0x3D	R/W	0x00	
TONE_TREBLE_GAIN_INDEX	0x3E	R/W	0x00	
DEC_L2L_VOL	0x42	R/W	0x00	
DEC_L2R_VOL	0x43	R/W	0x00	
DEC_R2L_VOL	0x44	R/W	0x00	
DEC_R2R_VOL	0x45	R/W	0x00	
BAND0_AVG_LEFT_POW	0x46	RO	-	
BAND1_AVG_LEFT_POW	0x47	RO	-	
BAND2_AVG_LEFT_POW	0x48	RO	-	
BAND3_AVG_LEFT_POW	0x49	RO	-	
BAND4_AVG_LEFT_POW	0x4A	RO	-	
BAND5_AVG_LEFT_POW	0x4B	RO	-	
BAND6_AVG_LEFT_POW	0x4C	RO	-	
BAND7_AVG_LEFT_POW	0x4D	RO	-	
BAND8_AVG_LEFT_POW	0x4E	RO	-	
BAND9_AVG_LEFT_POW	0x4F	RO	-	

MP3 decoder Sound Effect Control (SOUND_EFFECT_CONTROL, 0xFA31) : Read / Write

7	6	5	4	3	2	1	0

SEBM	SWE	SWHE	APUE	EE	TE	BE	BME

SEBM : enable decoder buffer mode
SWE : enables srswow1 function
SWHE : enables srswow2 function
APUE : enables update of average power value
EE : enables 10band equalizer
TE : enables tone functions
BE : enables bbe function
BME : enables bbemp function

***if [7] bit changes, you must reset common function register(0xfa02[3]).**

MP3 decoder Prescale Gain Index (PRESCALE_GAIN_INDEX, 0xFA32) : Read / Write

7	6	5	4	3	2	1	0
Reserved	PRE_GAIN						

This is for controlling input level to protect level clipping before data goes to internal sound effect module.

PRE_GAIN :

0000000 : +16 dB
0010000 : 0 dB
1111111 : -111 dB

MP3 decoder Equalizer Band0 Gain Index (EQ_BAND0_GAIN_INDEX, 0xFA33) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ0_GAIN				

EQ is equalizer block that has 10 bands. Each one band has one tenth(1/10) bandwidth of half sampling rate frequency. Writing 0 to the register makes +16dB level amplification of band0 frequencies of input audio data to EQ. And writing 31 makes -15dB gain of band0 frequencies.

EQ0_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band1 Gain Index (EQ_BAND1_GAIN_INDEX, 0xFA34) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ1_GAIN				

Gain control of band1 frequencies of input audio data to EQ.

EQ1_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band2 Gain Index (EQ_BAND2_GAIN_INDEX, 0xFA35) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ2_GAIN				

Gain control of band2 frequencies of input audio data to EQ.

EQ2_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band3 Gain Index (EQ_BAND3_GAIN_INDEX, 0xFA36) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ3_GAIN				

Gain control of band3 frequencies of input audio data to EQ.

EQ3_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band4 Gain Index (EQ_BAND4_GAIN_INDEX, 0xFA37) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ4_GAIN				

Gain control of band4 frequencies of input audio data to EQ.

EQ4_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band5 Gain Index (EQ_BAND5_GAIN_INDEX, 0xFA38) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ5_GAIN				

Gain control of band5 frequencies of input audio data to EQ.

EQ5_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band6 Gain Index (EQ_BAND6_GAIN_INDEX, 0xFA39) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ6_GAIN				

Gain control of band6 frequencies of input audio data to EQ.

EQ6_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band7 Gain Index (EQ_BAND7_GAIN_INDEX, 0xFA3A) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ7_GAIN				

Gain control of band7 frequencies of input audio data to EQ.

EQ7_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band8 Gain Index (EQ_BAND8_GAIN_INDEX, 0xFA3B) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ8_GAIN				

Gain control of band8 frequencies of input audio data to EQ.

EQ6_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Equalizer Band9 Gain Index (EQ_BAND9_GAIN_INDEX, 0xFA3C) : Read / Write

7	6	5	4	3	2	1	0
Reserved			EQ9_GAIN				

Gain control of band9 frequencies of input audio data to EQ.

EQ6_GAIN :

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Tone Bass Gain Index (TONE_BASS_GAIN_INDEX, 0xFA3D) : Read / Write

7	6	5	4	3	2	1	0
Reserved			TB_GAIN				

TB_GAIN : indicates the gain level of bass tone

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Tone Treble Gain Index (TONE_TREBLE_GAIN_INDEX, 0xFA3E) : Read / Write

7	6	5	4	3	2	1	0
Reserved			TT_GAIN				

TT_GAIN : indicates the gain level of treble tone

00000 : +16 dB
10000 : 0 dB
11111 : -15 dB

MP3 decoder Left to Left Volume (DEC_L2L_VOL, 0xFA42) : Read / Write

7	6	5	4	3	2	1	0
SI	VOL						

This register is used to set the volume control with the data from the left output of the MP3/WMA decoder to the left channel of the DAC on the audio CODEC.

SI : This bit is used to set the sign invert.

VOL : These bits are used to set the volume.

0000000 : +16 dB
0010000 : 0 dB
1111111 : -111 dB

MP3 decoder Left to Right Volume (DEC_L2R_VOL, 0xFA43) : Read / Write

7	6	5	4	3	2	1	0
SI	VOL						

This register is used to set the volume control with the data from the left output of the MP3/WMA decoder to the right channel of the DAC on the audio CODEC.

SI : This bit is used to set the sign invert.

VOL : These bits are used to set the volume.

0000000 : +16 dB
0010000 : 0 dB
1111111 : -111 dB

MP3 decoder Right to Left Volume (DEC_R2L_VOL, 0xFA44) : Read / Write

7	6	5	4	3	2	1	0
SI	VOL						

This register is used to set the volume control with the data from the right output of the MP3/WMA decoder to the left channel of the DAC on the audio CODEC.

SI : This bit is used to set the sign invert.

VOL : These bits are used to set the volume.

0000000 : +16 dB
0010000 : 0 dB
1111111 : -111 dB

MP3 decoder Right to Right Volume (DEC_R2R_VOL, 0xFA45) : Read / Write

7	6	5	4	3	2	1	0
SI	VOL						

This register is used to set the volume control with the data from the right output of the MP3/WMA decoder to the right channel of the DAC on the audio CODEC.

SI : This bit is used to set the sign invert.

VOL : These bits are used to set the volume.

0000000 : +16 dB
0010000 : 0 dB
1111111 : -111 dB

MP3 decoder Band0 Average Left Power (BAND0_AVG_LEFT_POW, 0xFA46) : Read Only

7	6	5	4	3	2	1	0
B0_ALP							

This register is used to Equalizer display Control.

B0_ALP : shows left channel band0 power.

MP3 decoder Band1 Average Left Power (BAND1_AVG_LEFT_POW, 0xFA47) : Read Only

7	6	5	4	3	2	1	0
B1_ALP							

This register is used to Equalizer display Control.

B1_ALP : shows left channel band1 power.

MP3 decoder Band2 Average Left Power (BAND2_AVG_LEFT_POW, 0xFA48) : Read Only

7	6	5	4	3	2	1	0
B2_ALP							

This register is used to Equalizer display Control.

B2_ALP : shows left channel band2 power.

MP3 decoder Band3 Average Left Power (BAND3_AVG_LEFT_POW, 0xFA49) : Read Only

7	6	5	4	3	2	1	0
B3_ALP							

This register is used to Equalizer display Control.

B3_ALP : shows left channel band3 power.

MP3 decoder Band4 Average Left Power (BAND4_AVG_LEFT_POW, 0xFA4A) : Read Only

7	6	5	4	3	2	1	0
B4_ALP							

This register is used to Equalizer display Control.

B4_ALP : shows left channel band4 power.

MP3 decoder Band5 Average Left Power (BAND5_AVG_LEFT_POW, 0xFA4B) : Read Only

7	6	5	4	3	2	1	0
B5_ALP							

This register is used to Equalizer display Control.

B5_ALP : shows left channel band5 power.

MP3 decoder Band6 Average Left Power (BAND6_AVG_LEFT_POW, 0xFA4C) : Read Only

7	6	5	4	3	2	1	0
B6_ALP							

This register is used to Equalizer display Control.

B6_ALP : shows left channel band6 power.

MP3 decoder Band7 Average Left Power (BAND7_AVG_LEFT_POW, 0xFA4D) : Read Only

7	6	5	4	3	2	1	0
B7_ALP							

This register is used to Equalizer display Control.

B7_ALP : shows left channel band7 power.

MP3 decoder Band8 Average Left Power (BAND8_AVG_LEFT_POW, 0xFA4E) : Read Only

7	6	5	4	3	2	1	0
B8_ALP							

This register is used to Equalizer display Control.

B8_ALP : shows left channel band8 power.

MP3 decoder Band9 Average Left Power (BAND9_AVG_LEFT_POW, 0xFA4F) : Read Only

7	6	5	4	3	2	1	0
B9_ALP							

This register is used to Equalizer display Control.

B9_ALP : shows left channel band9 power.

14.9. MP3 / WMA decoder registers

Table 43. MP3 / WMA decoder Register Map (P2 = 0xFA)

Function	Address (Hex)	Type	Reset	Description
DEC_HEADER_STATUS0	0x50	RO	-	
DEC_HEADER_STATUS1	0x51	RO	-	
DEC_HEADER_STATUS2	0x52	RO	-	
DEC_ANC_BIT_NUM_LO	0x53	RO	-	
DEC_ANC_BIT_NUM_HI	0x54	RO	-	
DECODER_ANC_DATA	0x55	RO	-	
BAND0_AVG_RIGHT_POW	0x56	RO	-	
BAND1_AVG_RIGHT_POW	0x57	RO	-	
BAND2_AVG_RIGHT_POW	0x58	RO	-	
BAND3_AVG_RIGHT_POW	0x59	RO	-	
BAND4_AVG_RIGHT_POW	0x5A	RO	-	
BAND5_AVG_RIGHT_POW	0x5B	RO	-	
BAND6_AVG_RIGHT_POW	0x5C	RO	-	
BAND7_AVG_RIGHT_POW	0x5D	RO	-	
BAND8_AVG_RIGHT_POW	0x5E	RO	-	
BAND9_AVG_RIGHT_POW	0x5F	R/W	-	
BAND_POW_UPDATE_INTERVAL	0x60	R/W	0xFF	
BAND_POW_SHIFT_VAL	0x61	RO	0x07	
AUD_DAC_BUF_DATA_LO	0x62	RO	-	
AUD_DAC_BUF_DATA_HI	0x63	RO	-	
AUD_DAC_LEFT_BUF_DATA_LO	0x64	RO	-	
AUD_DAC_LEFT_BUF_DATA_HI	0x65	RO	-	

AUD_DAC_RIGHT_BUF_DATA_LO	0x66	RO	-	
AUD_DAC_RIGHT_BUF_DATA_HI	0x67	RO	-	
WMA_DEC_STATUS	0x68	RO	-	
PACKET_CNT_LO	0x69	RO	-	
PACKET_CNT_HI	0x6A	RO	-	
DATA_PACKET_POS_LO	0x6B	RO	-	
DATA_PACKET_POS_HI	0x6C	RO	-	
DRM_POS_LO	0x6D	RO	-	
DRM_POS_HI	0x6E	RO	-	
SAMP_FREQ_LO	0x70	RO	-	
SAMP_FREQ_HI	0x71	RO	-	
BPS_LO	0x72	RO	-	
BPS_HI	0x73	RO	-	
PACKET_SIZE_LO	0x74	RO	-	
PACKET_SIZE_HI	0x75	RO	-	
ENC_OPTION	0x76	RO	-	
STREAM_ID	0x77	RO	-	
CODEC_TYPE_LO	0x78	RO	-	
CODEC_TYPE_HI	0x79	RO	-	
CONTENT_POS_LO	0x7A	RO	-	
CONTENT_POS_HI	0x7B	RO	-	
EXT_CONTENT_POS_LO	0x7C	RO	-	
EXT_CONTENT_POS_HI	0x7D	RO	-	
META_OBJ_POS_LO	0x7E	RO	-	
META_OBJ_POS_HI	0x7F			

MP3 Header Status0 (DEC_HEADER_STATUS0, 0xFA50) : Read Only

7	6	5	4	3	2	1	0
reserved			MFS		MLS		PROT

This register is used to check the status of the MP3 decoder header.

MFS : These bits are used to check the MPEG format.

00 : MPEG2.5.
01 : Reserved.
10 : MPEG2.
11 : MPEG1.

MLS[1:0] : This bit is used to check the MPEG layer.

00 : Reserved.
01 : Layer 3.
10 : Layer 2.

11 : Layer 1.

PROT : This bit is used to check the CRC error.

0 : Protected by CRC.
1 : Not protected by CRC

MP3 Header Status1 (DEC_HEADER_STATUS1, 0xFA51) : Read Only

7	6	5	4	3	2	1	0
BRI				SF		PADD	PRIV

This register is used to check the status of the MP3 decoder header.

BRI: These bits are used to check the bit rate index.

	MPEG1	MPEG2	MPEG2.5
0000	Free	Free	Free
0001	32	8	8
0010	40	16	16
0011	48	24	24
0100	56	32	32
0101	64	40	40
0110	80	48	48
0111	96	56	56
1000	112	64	64
1001	128	80	80
1010	160	96	96
1011	192	112	112
1100	224	128	128
1101	256	144	144
1110	320	160	160
1111	Invalid	Invalid	Invalid

SF[1:0] : These bits are used to check the bit rate index.

	MPEG1	MPEG2	MPEG2.5
00	44100	22050	11025
01	48000	24000	12000
10	32000	16000	8000
11	Invalid	Invalid	Invalid

PADD : padding bit

PRIV : private bit

MP3 Header Status2 (DEC_HEADER_STATUS3, 0xFA52) : Read Only

7	6	5	4	3	2	1	0
AM		SME		CP	ORG	EMP	

This register is used to check the status of the MP3 decoder header.

AM[1:0] : These bits are used to check the audio mode

00 : Stereo
01 : Joint stereo
10 : Dual channel
11 : Single channel

SME[1:0] : These bits are used to check the audio mode extension.

	Intersity_stereo	Ms_stereo
00 :	Off	Off
01 :	On	Off
10 :	Off	On
11 :	On	On

CP : This bit is used to check the copy.

ORG : This bit is used to check the original.

EMP[1:0] : These bits are used to check the emphasis.

00 : None
01 : 50/15us
10 : Reserved
11 : CCITT J.17

MP3 Ancillary Bit Number Lower 8-bit (DEC_ANC_BIT_NUM_LO, 0xFA53) : Read Only

7	6	5	4	3	2	1	0
DABNL							

This register is used to check the lower 8-bit of the MP3 decoder ancillary bit number.

DABNL : MW_DABN[7:0].

MP3 Ancillary Bit Number Upper 8-bit (DEC_ANC_BIT_NUM_HI, 0xFA54) : Read Only

7	6	5	4	3	2	1	0
DABNH							

This register is used to check the upper 8-bit of the MP3 decoder ancillary bit number.

DABNH : MW_DABN[15:8].

MP3 Ancillary Data (DECODER_ANC_DATA, 0xFA55) : Read Only

7	6	5	4	3	2	1	0
DAD							

This register is used to check the MP3 decoder ancillary data.

DAD : These bits are used to check the MP3 decoder ancillary data.

MP3 Band0 Average Right Power (BAND0_AVG_RIGHT_POW, 0xFA56) : Read Only

7	6	5	4	3	2	1	0
B0_ARP							

B0_ARP : shows band0 average power of right channel

MP3 Band1 Average Right Power (BAND1_AVG_RIGHT_POW, 0xFA57) : Read Only

7	6	5	4	3	2	1	0
B1_ARP							

B1_ARP : shows band1 average power of right channel

MP3 Band2 Average Right Power (BAND2_AVG_RIGHT_POW, 0xFA58) : Read Only

7	6	5	4	3	2	1	0

B2_ARP

B2_ARP : shows band2 average power of right channel

MP3 Band3 Average Right Power (BAND3_AVG_RIGHT_POW, 0xFA59) : Read Only

7	6	5	4	3	2	1	0
B3_ARP							

B3_ARP : shows band3 average power of right channel

MP3 Band4 Average Right Power (BAND4_AVG_RIGHT_POW, 0xFA5A) : Read Only

7	6	5	4	3	2	1	0
B4_ARP							

B4_ARP : shows band4 average power of right channel

MP3 Band5 Average Right Power (BAND5_AVG_RIGHT_POW, 0xFA5B) : Read Only

7	6	5	4	3	2	1	0
B5_ARP							

B5_ARP : shows band5 average power of right channel

MP3 Band6 Average Right Power (BAND6_AVG_RIGHT_POW, 0xFA5C) : Read Only

7	6	5	4	3	2	1	0
B6_ARP							

B6_ARP : shows band6 average power of right channel

MP3 Band7 Average Right Power (BAND7_AVG_RIGHT_POW, 0xFA5D) : Read Only

7	6	5	4	3	2	1	0
B7_ARP							

B7_ARP : shows band7 average power of right channel

MP3 Band8 Average Right Power (BAND8_AVG_RIGHT_POW, 0xFA5E) : Read Only

7	6	5	4	3	2	1	0
B8_ARP							

B8_ARP : shows band8 average power of right channel

MP3 Band9 Average Right Power (BAND9_AVG_RIGHT_POW, 0xFA5F) : Read Only

7	6	5	4	3	2	1	0
B9_ARP							

B9_ARP : shows band9 average power of right channel

MP3 Band Power Update Interval (BAND_POW_UPDATE_INTERVAL, 0xFA60) : Read Only

7	6	5	4	3	2	1	0
BPUI							

BPUI : indicates the update interval of average power calculation

MP3 Band Power Shift Value (BAND_POW_SHIFT_VAL, 0xFA61) : Read Only

7	6	5	4	3	2	1	0

BPSV

BPSV : indicates the gain level of average power calculation

MP3 Audio DAC Buffer Date Lower 8-bit (AUD_DAC_BUF_DATA_LO, 0xFA62) : Read Only

7	6	5	4	3	2	1	0
ADBDL							

ADBDL : shows the current audio dac data

MP3 Audio DAC Buffer Date Upper 8-bit (AUD_DAC_BUF_DATA_HI, 0xFA63) : Read Only

7	6	5	4	3	2	1	0
ADBDH							

ADBDH : shows the current audio dac data

MP3 Audio DAC Left Buffer Date Lower 8-bit (AUD_DAC_LEFT_BUF_DATA_LO, 0xFA64) : Read Only

7	6	5	4	3	2	1	0
ADLBDL							

ADLBDL : shows the current audio dac left data

MP3 Audio DAC Left Buffer Date Upper 8-bit (AUD_DAC_LEFT_BUF_DATA_HI, 0xFA65) : Read Only

7	6	5	4	3	2	1	0
ADLBDH							

ADLBDH : shows the current audio dac left data

MP3 Audio DAC Right Buffer Date Lower 8-bit (AUD_DAC_RIGHT_BUF_DATA_LO, 0xFA66) : Read Only

7	6	5	4	3	2	1	0
ADRBDL							

ADRBDL : shows the current audio dac right data

MP3 Audio DAC Right Buffer Date Upper 8-bit (AUD_DAC_RIGHT_BUF_DATA_HI, 0xFA67) : Read Only

7	6	5	4	3	2	1	0
ADRBDH							

ADRBDH : shows the current audio dac right data

WMA Decoder Status (WMA_DEC_STATUS, 0xFA68) : Read Only

7	6	5	4	3	2	1	0
Reserved					BF	DC	HD

This register is used to check the WMA decoder status.

BF : broadcast flag
DC : stereo mode

HD : This bit is used to check whether or not to detect a header.

0 : Head undetected.
1 : Head detected.

WMA Packet Count Low 8-bit (PACKET_CNT_LO, 0xFA69) : Read Only

7	6	5	4	3	2	1	0

PACKET_CNT _LO

This register is used to check the lower packet count.
It indicates the data packet number existing in Header and indicates only 2 byte out of 8 byte in total. But, in case of more than 2 byte, it will be indicated into 0xFFFF. It comes into effect only when broadcast flag in 0x61 is 0.

PACKET_CNT_LO [7:0] : PACKET_CNT[7:0].

WMA Packet Count High 8-bit (PACKET_CNT_HI, 0xFA6A) : Read Only

7	6	5	4	3	2	1	0
PACKET_CNT_HI							

This register is used to check the upper packet count.

PACKET_CNT_HI [7:0] : PACKET_CNT[15:8].

WMA DATA section Position Low (DATA_PACKET_POS_LO, 0xFA6B) : Read Only

7	6	5	4	3	2	1	0
PACKET_POS_LO							

PACKET_POS_LO : indicates the byte offset of data section in file

WMA DATA section Position High (DATA_PACKET_POS_HI, 0xFA6C) : Read Only

7	6	5	4	3	2	1	0
PACKET_POS_HI							

PACKET_POS_HI :

WMA DRM Position Low (DRM_POS_LO, 0xFA6D) : Read Only

7	6	5	4	3	2	1	0
DRM_POS_LO							

DRM_POS_LO : indicates the byte offset of drm header in file

WMA DRM Position High (DRM_POS_HI, 0xFA6E) : Read Only

7	6	5	4	3	2	1	0
DRM_POS_HI							

DRM_POS_HI :

WMA Decoder Sampling Frequency Lower 8-bit (SAMP_FREQ_LO, 0xFA70) : Read Only

7	6	5	4	3	2	1	0
SAMP_FREQ_LO							

This register is used to check the lower 8-bit of WMA decoder sampling frequency.

SAMP_FREQ_LO: SAMP_FREQ [7:0].

WMA Decoder Sampling Frequency Upper 8-bit (SAMP_FREQ_HI, 0xFA71) : Read Only

7	6	5	4	3	2	1	0
SAMP_FREQ_HI							

`

This register is used to check the upper 8-bit of WMA decoder sampling frequency.

SAMP_FREQ_HI: SAMP_FREQ [15:8].

WMA Decoder Byte per Second Lower 8-bit (BPS_LO, 0xFA72) : Read Only

7	6	5	4	3	2	1	0
BPS_LO							

This register is used to check the lower 8-bit of WMA decoder byte per second.

BPS_LO : BPS[7:0].

WMA Decoder Byte per Second Upper 8-bit (BPS_HI, 0xFA73) : Read Only

7	6	5	4	3	2	1	0
BPS_HI							

This register is used to check the upper 8-bit of WMA decoder byte per second.

BPS_HI : BPS[15:8].

WMA Decoder Packet Size Lower 8-bit (PACKET_SIZE_LO, 0xFA74) : Read Only

7	6	5	4	3	2	1	0
PACKET_SIZE_LO							

This register is used to check the lower 8-bit of WMA decoder packet size.

PACKET_SIZE_LO : PACKET_SIZE[7:0].

WMA Decoder Packet Size Upper 8-bit (PACKET_SIZE_HI, 0xFA75) : Read Only

7	6	5	4	3	2	1	0
PACKET_SIZE_HI							

This register is used to check the upper 8-bit of WMA decoder packet size.

PACKET_SIZE_HI : PACKET_SIZE[15:8].

Encoder Options (ENC_OPTION, 0xFA76) : Read Only

7	6	5	4	3	2	1	0
ENC_OPTION							

This register is used to check encoder option.

ENC_OPTION :

Stream ID(STREAM_ID, 0xFA77) : Read Only

7	6	5	4	3	2	1	0
STREAM_ID							

This register is used to check stream ID.
If it is 0, it means that there isn't audio packet.

STREAM_ID :

Codec Type Lower 8-bit (CODEC_TYPE_LO, 0xFA78) : Read Only

7	6	5	4	3	2	1	0
CODEC_TYPE_LO							

This register is used to check the lower 8-bit of codec type
Only 0x161 can do decoding.

CODEC_TYPE_LO : CODEC_TYPE[7:0].

Codec Type Upper 8-bit (CODEC_TYPE_HI, 0xFA79) : Read Only

7	6	5	4	3	2	1	0
CODEC_TYPE_HI							

This register is used to check the upper 8-bit of codec type

CODEC_TYPE_HI : CODEC_TYPE[15:8].

Content Position Lower 8-bit (CONTENT_POS_LO, 0xFA7A) : Read Only

7	6	5	4	3	2	1	0
CONTENT_POS_LO							

CONTENT_POS_LO : CONTENT_POS[7:0].

Indicates the byte offset of contents object in file

Content Position Upper 8-bit (CONTENT_POS_HI, 0xFA7B) : Read Only

7	6	5	4	3	2	1	0
CONTENT_POS_HIGH							

CONTENT_POS_HI : CONTENT_POS[15:8].

Extended Content Position Lower 8-bit (EXT_CONTENT_POS_LO, 0xFA7C) : Read Only

7	6	5	4	3	2	1	0
EXT_CONTENT_POS_LO							

EXT_CONTENT_POS_LO : EXT_CONTENT_POS[7:0].

Extended Content Position Upper 8-bit (EXT_CONTENT_POS_HI, 0xFA7D) : Read Only

7	6	5	4	3	2	1	0
EXT_CONTENT_POS_HI							

EXT_CONTENT_POS_HI : EXT_CONTENT_POS[15:8].

Indicates the byte offset of extended contents object in file

Meta Object Position Lower 8-bit (META_OBJ_POS_LO, 0xFA7E) : Read Only

7	6	5	4	3	2	1	0
META_OBJ_POS_LO							

META_OBJ_POS_LO : META_OBJ_POS[7:0].

Indicates the byte offset of meta object in file

Meta Object Position Upper 8-bit (META_OBJ_POS_HI, 0xFA7F) : Read Only

7	6	5	4	3	2	1	0
META_OBJ_POS_HI							

META_OBJ_POS_HI : META_OBJ_POS[15:8].

15. RTC (Real Time Clock)

15. RTC (Real Time Clock)

NX5850 has a timer named RTC that it works with independent backup battery and independent clock (32.768KHz) or system clock or 1Hz external clock. The RTC has simple counter it starts count after reset. User can use only variance value of counter from the reference time that user remember in a program. The counter value can not be changed.

15.1. RTC Power Supply & Clock

RTC power NX5850 is independent from system power. Backup Battery power must be connected to pin 114, 115, 124 and 125 for normal operation of RTC timer at main battery off and independent 32.768KHz crystal oscillator must be connected to pin 1 and 2.

15.2. RTC Control Register

This register sets the RTC options. RTC register is in SYS_CTRL block, address 0xff30~0xff3C, but is an independent block with independent power and clock. RTC counter is RTC_COUNTER_VALUE address 0xff37~0xff3a (25 bit), and increased by 1 at every 1 second interval.
RTC_ALARM_VALUE(0xff33~0xff35 (25bit)) is used for alarm interrupt. If user writes a time value to RTC_ALARM_VALUE, the value is compared with RTC_COUNTER_VALUE and then alarm interrupt is generated when two counter value is same.
RTC_TIMER_VALUE(0xff3b,0xff3c) can be used for regular interval timer interrupt. If user writes a time value to RTC_TIMER_VALUE, the timer interrupts occurs at every time of the time value interval written.
RTC_BLOCK_CONTROL_0(0xff30) selects RTC block clock source and makes RTC block register active or not.
RTC_BLOCK_CONTROL_1(0xff31) makes RTC block register writable or not.

Table 44. RTC Control Register Map (P2 = 0xFF)

Function	Address (Hex)	Type	Reset	Description
RTC_BLOCK_CONTROL0	0x30	R/W	0x00	RTC clock mode select.
RTC_BLOCK_CONTROL1	0x31	R/W	0x00	RTC register write enable.
RTC_BLOCK_CONTROL2	0x32	R/W	0x00	Timer and alarm interrupt control.
RTC_ALARM_VALUE	0x33 ~0x36	R/W	0x00	RTC alarm value.
RTC_COUNTER_VALUE	0x37 ~0x3A	RO	0x00	RTC counter value.
RTC_TIMER_VALUE	0x3B ~0x3C	R/W	0x00	RTC timer value.

RTC Clock Mode Select (RTC_BLOCK_CONTROL0, 0xFF30) : Read / Write

7	6	5	4	3	2	1	0
Reserved				CLK_MOD		RST_CNT	Reserved

Writing RST_CNT with 0 makes RTC counter value all 0.

CLK_MOD :

00 : 32768Hz clock.
01 : 12MHz clock.
10 : 1Hz clock.
11 : Reserved

RST_CNT : This bit is used to reset RTC time counter.

0 : RTC time counter reset.
1 : No action.

RTC Register Write Enable (RTC_BLOCK_CONTROL1, 0xFF31) : Read / Write

7	6	5	4	3	2	1	0
Reserved							WR_EN

This bit must be high to write 0xFF30 ~ 0xFF3C block.

RTC Register Write Enable (RTC_BLOCK_CONTROL2, 0xFF32) : Read / Write

7	6	5	4	3	2	1	0
Reserved		TMR_INT	ALM_INT	Reserved		TMR_EN	ALM_EN

TMR_INT : This bit is used to timer interrupt.

0 : No action.
1 : Timer interrupt occurs.

ALM_INT : This bit is used to alarm interrupt.

0 : No action.
1 ; Alarm interrupt occurs.

TMR_EN : This bit is used to timer interrupt enable.

0 : No action.
1 : Timer interrupt enable.

ALM_EN : This bit is used to alarm interrupt enable.

0 : No action.
1 : Alarm interrupt enable.

RTC Alarm Value (RTC_ALARM_VALUE, 0xFF33) : Read / Write

7	6	5	4	3	2	1	0
ALM_VAL							

ALM_VAL[7:0]

RTC Alarm Value (RTC_ALARM_VALUE, 0xFF34) : Read / Write

7	6	5	4	3	2	1	0
ALM_VAL							

ALM_VAL[15:8]

RTC Alarm Value (RTC_ALARM_VALUE, 0xFF35) : Read / Write

7	6	5	4	3	2	1	0
ALM_VAL							

ALM_VAL[23:16]

RTC Alarm Value (RTC_ALARM_VALUE, 0xFF36) : Read / Write

7	6	5	4	3	2	1	0
ALM_VAL							

ALM_VAL[31:24]

RTC Counter Value (RTC_COUNTER_VALUE, 0xFF37) : Read Only

7	6	5	4	3	2	1	0
CNT_VAL							

CNT_VAL[7:0]

RTC Counter Value (RTC_COUNTER_VALUE, 0xFF38) : Read Only

7	6	5	4	3	2	1	0
CNT_VAL							

CNT_VAL[15:8]

RTC Counter Value (RTC_COUNTER_VALUE, 0xFF39) : Read Only

7	6	5	4	3	2	1	0
CNT_VAL							

CNT_VAL[23:16]

RTC Counter Value (RTC_COUNTER_VALUE, 0xFF3A) : Read Only

7	6	5	4	3	2	1	0
CNT_VAL							

CNT_VAL[31:24]

RTC Timer Value (RTC_TIMER_VALUE, 0xFF3B) : Read / Write

7	6	5	4	3	2	1	0
TMR_VAL							

TMR_VAL[7:0]

RTC Timer Value (RTC_TIMER_VALUE, 0xFF3C) : Read / Write

7	6	5	4	3	2	1	0
Reserved		TMR_VAL					

TMR_VAL[13:8]

16. GPIO

16. GPIO (General Purpose Input Output)

The register related with GPIO has a group of three registers, and it is set as the GPIO/Function pins. The following diagram shows I/O status set by the three register.

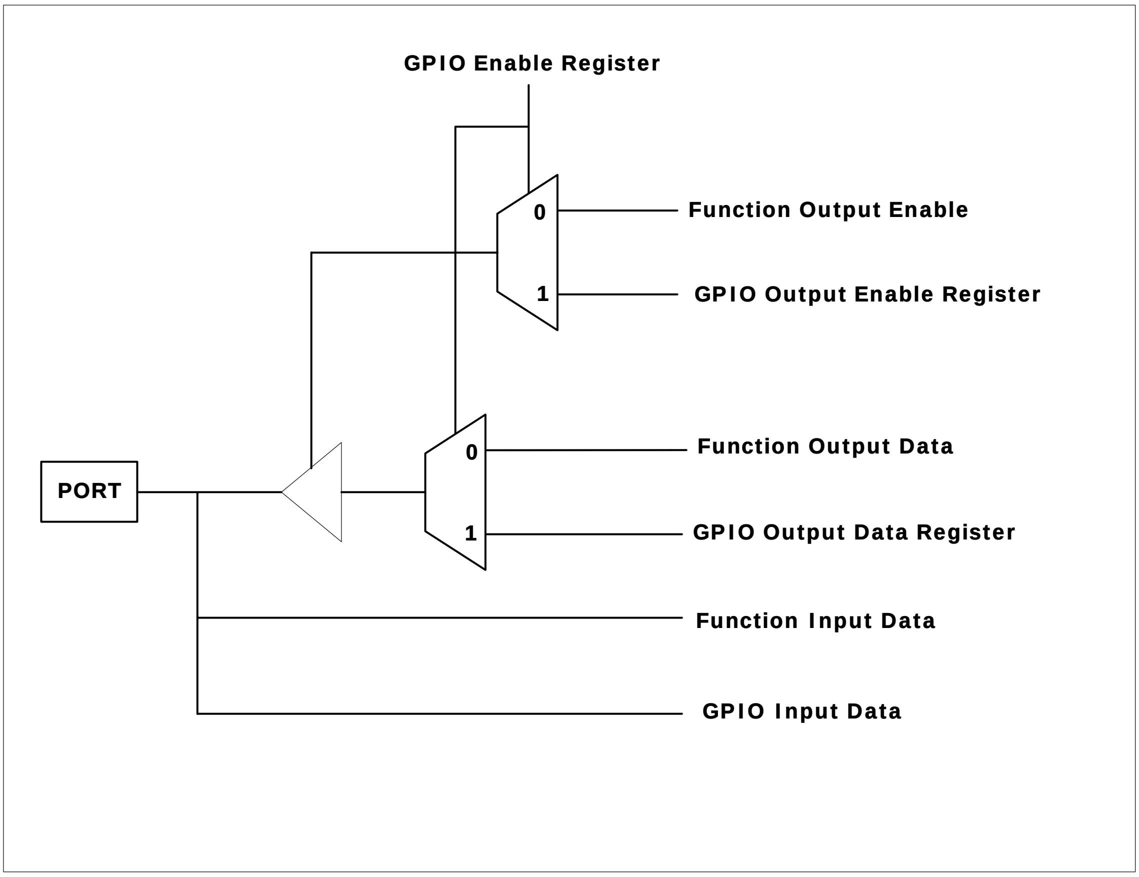

Figure 23. Block Diagram of GPIO

When the GPIO enable is high and the GPIO output control is low, the data from MCU are written into GPIO data. The data are stored into Flip-Flops and the output of Flip-Flops get into the GPIO output data. These data go to the related ports.
When the GPIO enable is high and the GPIO output control is high, the GPIO input data coming into ports are stored into Flip-Flops and the MCU can read the GPIO data from the Flip-Flops. When the GPIO enable is low, they work as Input/Output ports of the function defined by each function control.

16.1. GPIO Control Register

The GPIO Control Registers are in SYS_CTRL block address 0xff40~0xff60.

Table 45. GPIO Control Register Map (P2 = 0xFF)

Function	Address (Hex)	Type	Reset	Description
GPIO0_ENABLE	0x40	R/W	0xFF	Setting GPIO0 enable
GPIO0_INPUT_ENABLE	0x41	R/W	0xFF	Setting the direction of GPIO0
GPIO0_DATA	0x42	R/W	0xFF	Setting GPIO0 data port
GPIO1_ENABLE	0x43	R/W	0x07	Setting GPIO1 enable
GPIO1_INPUT_ENABLE	0x44	R/W	-	Setting the direction of GPIO1
GPIO1_DATA	0x45	RO/WO	-	Setting GPIO1 data port
GPIO2_ENABLE	0x46	R/W	0xFF	Setting GPIO2 enable
GPIO2_INPUT_ENABLE	0x47	R/W	0xFF	Setting the direction of GPIO2
GPIO2_DATA	0x48	R/W	0xFF	Setting GPIO2 data port
GPIO3_ENABLE	0x49	R/W	0xFF	Setting GPIO3 enable
GPIO3_INPUT_ENABLE	0x4A	R/W	0xFF	Setting the direction of GPIO3
GPIO3_DATA	0x4B	R/W	0xFF	Setting GPIO3 data port
GPIO4_ENABLE	0x4C	R/W	0xFF	Setting GPIO4 enable
GPIO4_INPUT_ENABLE	0x4D	R/W	0xFF	Setting the direction of GPIO4
GPIO4_DATA	0x4E	R/W	0xFF	Setting GPIO4 data port
GPIO5_ENABLE	0x4F	R/W	0xFF	Setting GPIO5 enable
GPIO5_INPUT_ENABLE	0x50	R/W	0xFF	Setting the direction of GPIO5
GPIO5_DATA	0x51	R/W	0xFF	Setting GPIO5 data port
GPIO6_ENABLE	0x52	R/W	0xFF	Setting GPIO6 enable
GPIO6_INPUT_ENABLE	0x53	R/W	0xFF	Setting the direction of GPIO6
GPIO6_DATA	0x54	R/W	0xFF	Setting GPIO6 data port
GPIO7_ENABLE	0x55	R/W	0xFF	Setting GPIO7 enable
GPIO7_INPUT_ENABLE	0x56	R/W	0xFF	Setting the direction of GPIO7
GPIO7_DATA	0x57	R/W	0xFF	Setting GPIO7 data port
GPIO8_ENABLE	0x58	R/W	0xFF	Setting GPIO8 enable
GPIO8_INPUT_ENABLE	0x59	R/W	0xFF	Setting the direction of GPIO8
GPIO8_DATA	0x5A	R/W	0xFF	Setting GPIO8 data port
GPIO9_ENABLE	0x5B	R/W	0xFF	Setting GPIO9 enable
GPIO9_INPUT_ENABLE	0x5C	R/W	0xFF	Setting the direction of GPIO9
GPIO9_DATA	0x5D	R/W	0xFF	Setting GPIO9 data port
GPIOA_ENABLE	0x5E	R/W	0xFF	Setting GPIOA enable
GPIOA_INPUT_ENABLE	0x5F	R/W	0xFF	Setting the direction of GPIOA
GPIOA_DATA	0x60	R/W	0xFF	Setting GPIOA data port

GPIO0 Enable (GPIO0_ENABLE, 0xFF40) : Read / Write

7	6	5	4	3	2	1	0
PSEN	ALE	PORT5	PORT4	PORT3	PORT2	PORT1	PORT0

This register is used to enable to use for GPIO. For example, writing ALE with 1 enables ALE pin as GPIO pin function.

PSEN :

0 : use **PSEN** as internal purpose
1 : use **PSEN** as GPIO

ALE :

0 : use **ALE** as internal purpose
1 : use **ALE** as GPIO

PORT5 :

0 : use **PORT5** as internal purpose
1 : use **PORT5** as GPIO

PORT4:

0 : use **PORT4** as internal purpose
1 : use **PORT4** as GPIO

PORT3 :

0 : use **PORT3** as internal purpose
1 : use **PORT3** as GPIO

PORT2 :

0 : use **PORT2** as internal purpose
1 : use **PORT2** as GPIO

PORT1 :

0 : use **PORT1** as internal purpose
1 : use **PORT1** as GPIO

PORT0 :

0 : use **PORT0** as internal purpose
1 : use **PORT0** as GPIO

GPIO0 Input Enable (GPIO0_INPUT_ENABLE, 0xFF41) : Read / Write

7	6	5	4	3	2	1	0
INP_EN							

Writing 0 makes GPIO0 output mode, Writing 1 makes GPIO0 input mode.

GPIO0 Data (GPIO0_DATA, 0xFF42) : Read / Write

7	6	5	4	3	2	1	0
GPIO0_DATA							

GPIO input or output data register of GPIO0 pin.

GPIO1 Enable (GPIO1_ENABLE, 0xFF43) : Read / Write

7	6	5	4	3	2	1	0
Reserved				DP_DM	FRNB	LCD	XRM

DP_DM :

0 : Disable.
1 : Enable to use for GPIO.

FRNB :

0 : Disable.
1 : Enable to use for GPIO.

LCD :

0 : Disable.
1 : Enable to use for GPIO.

XRM :

0 : Disable.
1 : Enable to use for GPIO.

GPIO1 Input Enable (GPIO1_INPUT_ENABLE, 0xFF44) : Read / Write

7	6	5	4	3	2	1	0
Reserved			USB_SUS	USB_INP	FRNB_INP	LCD_INP	XRM_INP

USB_SUS :

0 : enable **USB** transceiver
1 : Suspends **USB** transceiver.

USB_INP :

0 : use **USB** pins as output
1 : use **USB** pins as input

FRNB_INP :

0 : use **FRNB** pin as input
1 : use **FRNB** pin as input

LCD_INP :

0 : use **LCD** pin as input
1 : use **LCD** pin as input

XRM_INP :

0 : use **XRM** pin as input
1 : use **XRM** pin as input

GPIO1 Data (GPIO1_DATA, 0xFF45) : Read Only / Write Only

7	6	5	4	3	2	1	0
Reserved	NOR_BOOT	FIRM_UP	USB_DM	USB_DP	FRNB_DATA	LCD_DATA	XRM_DATA

NOR_BOOT and FIRM_UP bit is use for boot mode selection. GPIO1_DATA[4:0] has means as GPIO when address 0xFF43 bits are written with 1. GPIO input or output data register of GPIO1 pin. Reading is input data and writing is output data.

NOR_BOOT : Read Only
0 : NAND Boot Mode
1 : NOR Boot Mode

FIRM_UP : Read Only
0 : Normal Operation Mode
1 : Firmware Upgrade Mode

USB_DM : USB transceiver DM pin data

USB_DP : USB transceiver DP pin data

FRNB_DATA : FRNB pin data

LCD_DATA : LCD pin data

XRM_DATA : XRM pin data

GPIO2 Enable (GPIO2_ENABLE, 0xFF46) : Read / Write

7	6	5	4	3	2	1	0
GPIO2_EN							

Writing ‘1’ in this register enables the use of ADDR[7:0] pin as gpio.

GPIO2 Input Enable (GPIO2_INPUT_ENABLE, 0xFF47) : Read / Write

7	6	5	4	3	2	1	0
GPIO2_INP_EN							

Writing ‘1’ in this register enables the use of ADDR[7:0] pin as gpio input, otherwise gpio output.

GPIO2 Data (GPIO2_DATA, 0xFF48) : Read / Write

7	6	5	4	3	2	1	0
GPIO2_DATA							

Reading or Writing of this register access ADDR[7:0] data as gpio.

GPIO3 Enable (GPIO3_ENABLE, 0xFF49) : Read / Write

7	6	5	4	3	2	1	0
GPIO3_EN							

Writing ‘1’ in this register enables the use of LDATA[7:0] pin as gpio.

GPIO3 Input Enable (GPIO3_INPUT_ENABLE, 0xFF4A) : Read / Write

7	6	5	4	3	2	1	0
GPIO3_INP_EN							

Writing ‘1’ in this register enables the use of LDATA[7:0] pin as gpio input, otherwise gpio output.

GPIO3 Data (GPIO3_DATA, 0xFF4B) : Read / Write

7	6	5	4	3	2	1	0
GPIO3_DATA							

Reading or Writing of this register access LDATA[7:0] data as gpio.

GPIO4 Enable (GPIO4_ENABLE, 0xFF4C) : Read / Write

7	6	5	4	3	2	1	0
GPIO4_EN							

Writing '1' in this register enables the use of P1[7:0] pin as gpio.

GPIO4 Input Enable (GPIO4_INPUT_ENABLE, 0xFF4D) : Read / Write

7	6	5	4	3	2	1	0
GPIO4_INP_EN							

Writing '1' in this register enables the use of P1[7:0] pin as gpio input, otherwise gpio output.

GPIO4 Data (GPIO4_DATA, 0xFF4E) : Read / Write

7	6	5	4	3	2	1	0
GPIO4_DATA							

Reading or Writing of this register access P1[7:0] data as gpio.

GPIO5 Enable (GPIO5_ENABLE, 0xFF4F) : Read / Write

7	6	5	4	3	2	1	0
GPIO5_EN							

Writing '1' in this register enables the use of P2[7:0] pin as gpio.

GPIO5 Input Enable (GPIO5_INPUT_ENABLE, 0xFF50) : Read / Write

7	6	5	4	3	2	1	0
GPIO5_INP_EN							

Writing '1' in this register enables the use of P2[7:0] pin as gpio input, otherwise gpio output.

GPIO5 Data (GPIO5_DATA, 0xFF51) : Read / Write

7	6	5	4	3	2	1	0
GPIO5_DATA							

Reading or Writing of this register access P2[7:0] data as gpio.

GPIO6 Enable (GPIO6_ENABLE, 0xFF52) : Read / Write

7	6	5	4	3	2	1	0
GPIO6_EN							

Writing '1' in this register enables the use of P3[7:0] pin as gpio.

GPIO6 Input Enable (GPIO6_INPUT_ENABLE, 0xFF53) : Read / Write

7	6	5	4	3	2	1	0
GPIO6_INP_EN							

Writing '1' in this register enables the use of P3[7:0] pin as gpio input, otherwise gpio output.

GPIO6 Data (GPIO6_DATA, 0xFF54) : Read / Write

7	6	5	4	3	2	1	0
GPIO6_DATA							

Reading or Writing of this register access P3[7:0] data as gpio.

GPIO7 Enable (GPIO7_ENABLE, 0xFF55) : Read / Write

7	6	5	4	3	2	1	0
GPIO7_EN							

Writing ‘1’ in this register enables the use of FDATA[7:0] pin as gpio.

GPIO7 Input Enable (GPIO7_INPUT_ENABLE, 0xFF56) : Read / Write

7	6	5	4	3	2	1	0
GPIO7_INP_EN							

Writing ‘1’ in this register enables the use of FDATA[7:0] pin as gpio input, otherwise gpio output.

GPIO7 Data (GPIO7_DATA, 0xFF57) : Read / Write

7	6	5	4	3	2	1	0
GPIO7_DATA							

Reading or Writing of this register access FDATA[7:0] data as gpio.

GPIO8 Enable (GPIO8_ENABLE, 0xFF58) : Read / Write

7	6	5	4	3	2	1	0
GPIO8_EN							

Writing ‘1’ in this register enables the use of HDATA[7:0] pin as gpio.

GPIO8 Input Enable (GPIO8_INPUT_ENABLE, 0xFF59) : Read / Write

7	6	5	4	3	2	1	0
GPIO8_INP_EN							

Writing ‘1’ in this register enables the use of HDATA[7:0] pin as gpio input, otherwise gpio output.

GPIO8 Data (GPIO8_DATA, 0xFF5A) : Read / Write

7	6	5	4	3	2	1	0
GPIO8_DATA							

Reading or Writing of this register access HDATA[7:0] data as gpio.

GPIO9 Enable (GPIO9_ENABLE, 0xFF5B) : Read / Write

7	6	5	4	3	2	1	0
GPIO9_EN							

Writing ‘1’ in this register enables the use of each pin as gpio.

GPIO9_EN[7] : MCMD
GPIO9_EN[6] : MDAT
GPIO9_EN[5] : MCLK
GPIO9_EN[4] : MCK
GPIO9_EN[3] : SCK
GPIO9_EN[2] : CCK
GPIO9_EN[1] : SDI
GPIO9_EN[0] : SDO

GPIO9 Input Enable (GPIO9_INPUT_ENABLE, 0xFF5C) : Read / Write

7	6	5	4	3	2	1	0
GPIO9_INP_EN							

Writing ‘1’ in this register enables the use of each pin as gpio input, otherwise gpio output.

GPIO9 Data (GPIO9_DATA, 0xFF5D) : Read / Write

7	6	5	4	3	2	1	0
GPIO9_DATA							

Reading or Writing of this register access data as gpio.

GPIOA Enable (GPIOA_ENABLE, 0xFF5E) : Read / Write

7	6	5	4	3	2	1	0
GPIOA_EN							

Writing ‘1’ in this register enables the use of each pin as gpio.

GPIOA_EN[7] : FCEN3
GPIOA_EN[6] : FCEN2
GPIOA_EN[5] : FCEN1
GPIOA_EN[4] : FCEN0
GPIOA_EN[3] : FCLE
GPIOA_EN[2] : FALE
GPIOA_EN[1] : FWEN
GPIOA_EN[0] : FREN

GPIOA Input Enable (GPIOA_INPUT_ENABLE, 0xFF5F) : Read / Write

7	6	5	4	3	2	1	0
GPIOA_INP_EN							

Writing ‘1’ in this register enables the use of each pin as gpio input, otherwise gpio output.

GPIOA Data (GPIOA_DATA, 0xFF60) : Read / Write

7	6	5	4	3	2	1	0
GPIOA_DATA							

Reading or Writing of this register access data as gpio.

17. ADC

17. ADC (Analog-to-Digital Converter)

NX5850 has an ADC for check a battery and VOR(Voice Operation Recording) that can record only when the sound is aloud. The analog key function and the remote control key detection are used to reduce the external switches. NX5850 features a 8-bit successive approximation ADC. The ADC is connected to an 4-channel Analog Multiplexer which allows four single-ended voltage inputs. The single-ended voltage inputs refer to VBOT.

The ADC converts an analog input voltage to a 8-bit digital value through successive approximation. The minimum value represents VBOT and the maximum value represents the voltage on the VTOP pin minus 1 LSB.

The ADC is enabled by setting the ADC stand-by bit (SYS_BLOCK_POWER_CONTROL, 0xFF06[6]) and ADPD (ADC_CONTROL, 0xFF23[2]). Voltage reference and input channel selections will not go into effect until ADPD is set to stand-by. The ADC does not consume power when ADPD is low.

A single conversion is started by clearing the ADC Power Down Mode bit and the ADC generates a 8-bit result which is presented in the ADC data Registers, ADC_VALUE. After the conversion is complete (ADI, 'ADC_CONTROL, 0xFF23[3]', is high), the conversion result can be found in the ADC result register (ADC_VALUE).

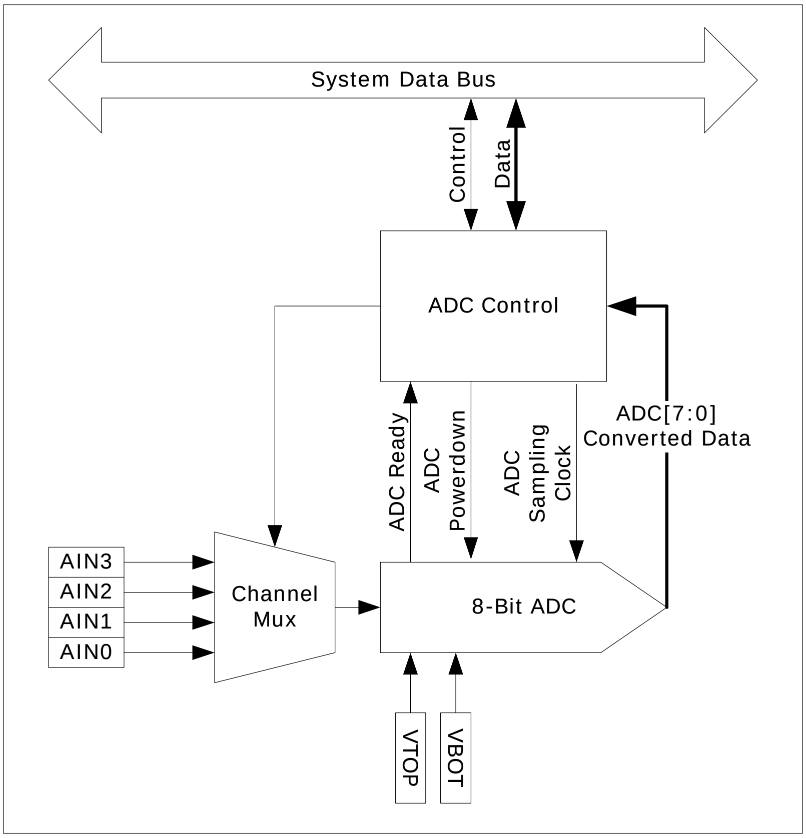

Figure 24. Analog-to-Digital Converter Block Diagram

18.1. ADC Control Register

ADC Control Registers are in SYS_CTRL block address 0xff20~0xff28.

Table 46. ADC Control Register Map (P2 = 0xFF)

Function	Address (Hex)	Type	Reset	Description
ADC_WAIT_TIME	0x20	R/W	0x00	
ADC_CLK_DIV	0x21	R/W	0x00	Controls ADC Output Frequency
AUD_CHANNEL_SELECT	0x22	R/W	0x00	
ADC_CONTROL	0x23	R/W	0x00	
ADC_CHANNEL0_DATA	0x24	RO	-	
ADC_CHANNEL1_DATA	0x25	RO	-	
ADC_CHANNEL2_DATA	0x26	RO	-	
ADC_CHANNEL3_DATA	0x27	RO	-	
ADC_CHANNEL4_DATA	0x28	RO	-	

ADC Wait Time (ADC_WAIT_TIME, 0xFF20) : Read / Write

7	6	5	4	3	2	1	0
WAIT_TIME							

ADC input is multiplexed input from 4 inputs. This register value is clock settlement time after selection of ADC multiplexer input source. ADC converting starts after this wait time.

ADC Clock Divider (ADC_CLK_DIV, 0xFF21) : Read / Write

7	6	5	4	3	2	1	0
CLK_DIV							

ADC clock is divided by this register as follows.

$$Fadc = Fsys / (2*(CLK\_DIV + 1))$$

Audio Channel Select (AUD_CHANNEL_SELECT, 0xFF22) : Read / Write

7	6	5	4	3	2	1	0
Reserved		LEFT_CHAN			RIGHT_CHAN		

If effective value is written to left or right channel register, corresponding ADC input is connected to the audio channel. Effective value means ADC in pin0, 1, 2, or 3. This is for ADC output channel selection of voice input or FM input to ADC.

LEFT_CHAN : left channel selection

Effective value is 0, 1, 2 or 3.

RIGHT_CHAN : right channel selection

Effective value is 0, 1, 2 or 3.

ADC Control (ADC_CONTROL, 0xFF23) : Read / Write

7	6	5	4	3	2	1	0
Reserved				ADC_RES	ADC_PWDN	ADC_CLK	ADC_COV

Enable ADC clock first and then enable ADC converting. Writing 0x0F(enable all) and then read data from address 0xFF24, 0xFF25, 0xFF26 and 0xFF27 registers which is the data converted from ADC. The data converted from ADC is the data continuously updated by clock.

ADC_RES : This bit is used to ADC block reset.

0 : ADC block reset.
1 : No action.

ADC_PWDN : This bit is used to ADC block power down.

0 : ADC block power down.
1 : No action.

ADC_CLK : This bit is used to ADC block clock enable.

0 : Disable.
1 : ADC block clock enable.

ADC_COV : This bit is used to ADC block convert enable.

0 : Disable.
1 : ADC block convert enable.

ADC Channel0 Data (ADC_CHANNEL0_DATA, 0xFF24) : Read Only

7	6	5	4	3	2	1	0
CHAN0_DAT							

ADC Channel1 Data (ADC_CHANNEL1_DATA, 0xFF25) : Read Only

7	6	5	4	3	2	1	0
CHAN1_DAT							

ADC Channel2 Data (ADC_CHANNEL2_DATA, 0xFF26) : Read Only

7	6	5	4	3	2	1	0
CHAN2_DAT							

ADC Channel3 Data (ADC_CHANNEL3_DATA, 0xFF27) : Read Only

7	6	5	4	3	2	1	0
CHAN3_DAT							

ADC Channel4 Data (ADC_CHANNEL4_DATA, 0xFF28) : Read Only

7	6	5	4	3	2	1	0
CHAN4_DAT							

18. LCD Control

18. LCD Control

NX5850 has three LCD control methods such as Parallel, Serial and DMA. The parallel method has the 8080/6800 mode, and the serial uses GPIOs. DMA can be used with parallel mode for efficient data transfer. Here shows the way LCD control, and the interface of the hardware(NX5850 demo board). Parallel mode can be 8bit or 16bit data bus interface and both bus mode can use DMA interface. Higher 8bits of 16 bits data bus use NorLcd_D8~NorLcd_D15 pins and other remaining pin connection is same as 8bit pin connection.

18.1. Parallel (8080 mode) Interface (8bit)

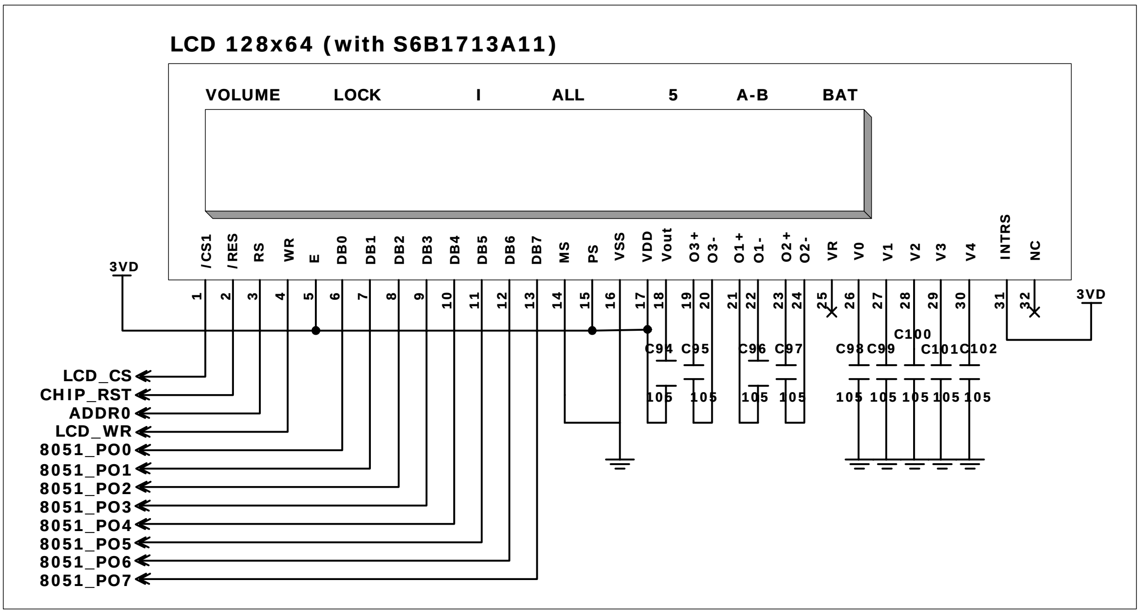

Figure 25. The Parallel interface to the 128x64 LCD module

In this mode, the LCD_CS signal is generated with the pin number 128 on NX5850 automatically as follows:

1. To use the LCD_CS (pin number 128), we have to define the bit 0 of the register 'GPIO_4_ENABLE (0xFF4C)' to the general port.

2. Define the address of the LCD area as follows and connect the address 0 to the LCD_RS.

```
#define  LCD_COMMAND     0xF100
#define  LCD_DATA        0xF101
#define write_XDATA(address,value)
(*((unsigned char volatile xdata*)address)=value)
```

3. Single command function.

```
void SingleCmd(UINT8 cmd) {
    write_XDATA(LCD_COMMAND, cmd);
}
void SingleDate(UINT8 data) {
    write_XDATA(LCD_DATA, data);
}
```

LCD can be controlled with the commands above and the data function. Following Figure 31. is the simple timing diagram to explain the function above.

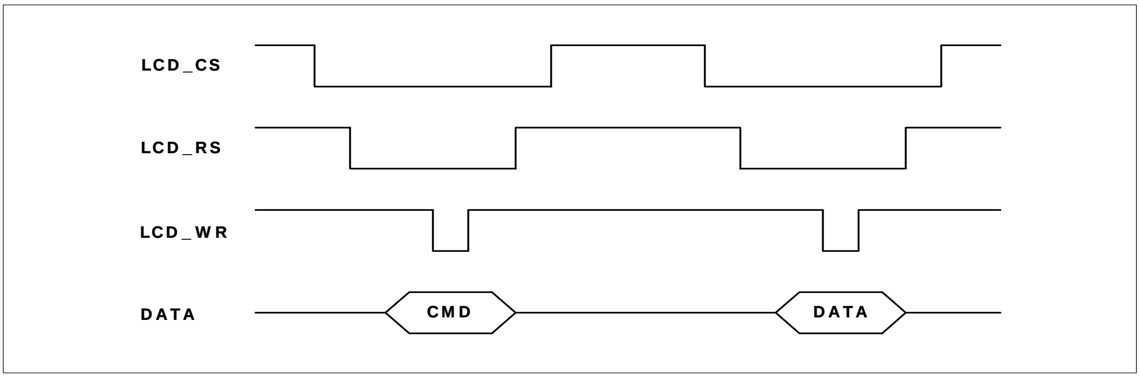

Figure 26. The LCD parallel interface mode timing

18.2 Serial Interface

18.2.1. Serial Interface Control Register

Use GPIOs for serial interface and emulate the GPIOs with software for serial communication with LCD.

19. Package

19. Package

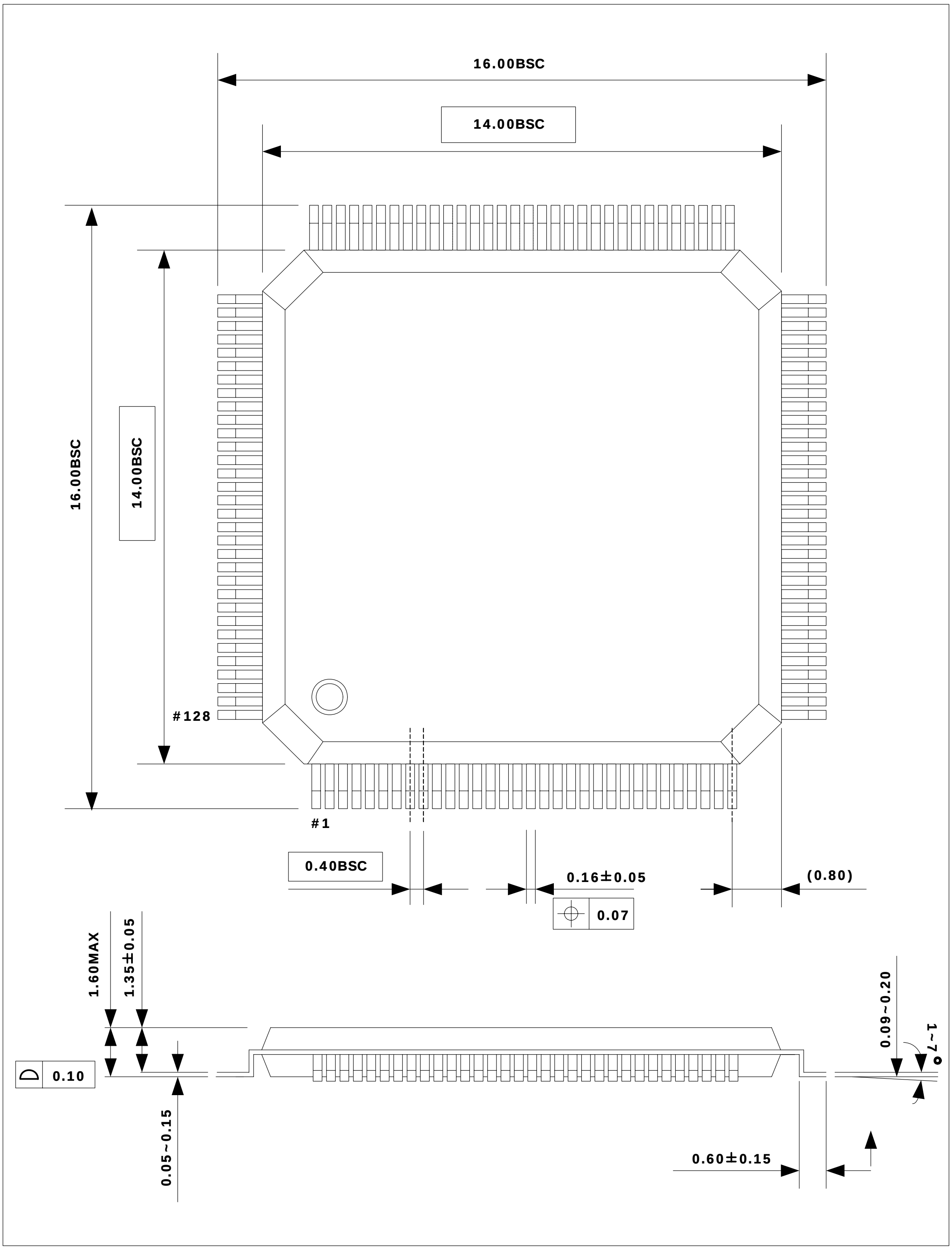

Figure 27. The Package Diagram

20. Electrical Characteristics

20. Electrical Characteristics

21.1 DC Specification, Note(1)

Parameter	Symbol	Min	Typ	Max	Unit
Input Low Voltage	VIL			0.8V	CMOS input
Input High Voltage	VIH	2.0V			
Input Low Voltage	VIL			0.8V	CMOS Schmitt input
Input High Voltage	VIH	2.0V			
Hysteresis		316mV	-	466mV	
Input High Current	IIH	-5uA	-	5uA	Vin = DVDD, Note (2)
Input with 52kΩ pull down			79.7uA		Vin = DVDD
Input Low Current	IIL	-5uA		5uA	Vin = Vss
Input with 63kΩ pull up			80uA		Vin = Vss
Output High Voltage	VOH	2.4V		3.6V	IOH = 2, 4, 8, 12, 16, 24 mA
Output Low Voltage	VOL	0.0V		0.4V	IOL = -2, -4, -8, -12, -16, -24 mA
Pull Up Resistor		47.3κΩ		99..9κΩ	
Pull Down Resistor		37.1κΩ		113.1κΩ	

Notes:

(1) When the ring voltage is 3.3V (typical), CMOS voltage levels and LVTTL voltage levels are the same. Therefore, any I/O cell with CMOS voltage level can be used for LVTTL voltage level.

For further information about LVTTL and CMOS output specifications refer to "Interface Standard for Nominal 3V/3.3V Supply Digital Integrated Circuit" (the latest JEDEC spec).

(2) DVDD is ring DC supply voltage as stated in the Operating Conditions table.

20.3 Electrical Characteristics for PLLs

20.3.1 Electrical Characteristics

Parameter	Symbol	Min	Typ	Max	Unit
Supply Voltage	VCC	1.6	1.8	2.0	V
Supply Current	Idd	-	4.0	5.0	MA
Power down current	Iddpdn		0.3		UA
Operating temperature	Ta	0	-	70	℃
Synthesize frequency	Fout	100	-	250	MHz
Duty cycle	Dcyc	40	-	60	%
Fin input duty cycle	-	40	-	60	%
Clock jitter(peak to peak)	-	-100	-	+100	PS
Frequency change to Fout stable time	-	-	10	20	US
Fout rise and fall time	Tr,Tf	-	-	0.8	NS
Input frequency	Ffin	4	-	30	MHz

20.4 Electrical Characteristics for ADC

20.4.1 Recommended Operating Conditions

Parameter	Symbol	Min	Typ	Max	Unit
Analog supply voltage	AVDD	1.62	1.8	1.98	V
Digital supply voltage	VDD	1.62	1.8	1.98	V
Reference top voltage range	Vreft	-	0.8AVDD	-	V
Reference bottom voltage range	Vrefb	-	0.2AVDD	-	V
Analog input differential range	Vain	-	Vreft-Vrefb	-	V
DC output voltage range	Vout	0		AVDD	V
Operating ambient temperature range	Top	0		70	℃
Input high threshold voltage	Vih	0.8VDD			V
Input low threshold voltage	Vil	-40		0.2VDD	V

20.4.2 DC Electrical Characteristics

(Typ:VDD=1.8V,Top=25℃)

Parameter	Symbol	Min	Typ	Max	Unit	Test Conditions
Differential Nonlinearity Error	DNL		±0.5		LSB	Vreft=0.8AVDD Vrefb=0.2AVDD
Integral Nonlinearity Error	INL		±1.0		LSB	Vreft=0.8AVDD Vrefb=0.2AVDD
Offset Voltage Error	OFE			1	%FSR	Vreft=0.8AVDD Vrefb=0.2AVDD
Gain Error	GAE			1	%FSR	Vreft=0.8AVDD Vrefb=0.2AVDD

20.4.3 AC Electrical Characteristics

(Typ:VDD=1.8V,Top=25℃)

Parameter	Symbol	Min	Typ	Max	Unit	Test Conditions
Conversion Rate	fC			1	MSPS	FSCLK=10MHz
Signal to Noise and Distortion	SNDR		33		dB	FSCLK=10MHz
Total Harmonic Distortion	THD		36		dB	FSCLK=10MHz
Operating Supply Current	IVdd		1.5		mA	Conversion Mode (FSCLK=10MHz)
Pwr. Down Mode Current	Ipd			4	uA	PDB=0,RSTB=0

21. Pin Description

21. Pin Description

No.	Pin Name	I/O	U/D	Io[mA]	Description
1	RTC_IN	I	-	-	Real Time Clock Crystal Oscillator input pin
2	RTC_OUT	O	-	-	Real Time Clock Crystal Oscillator output pin
3	FirmwareUp	I	-	-	Firmware Update mode selection
4	TMODE	I	-	-	Test mode
5	NorFlash	I	-	-	Nor Flash Memory boot mode selection
6	VDD3.3	DP	-	-	External Digital Power for Digital Part(3.3V)
7	GND3.3	DG	-	-	External Digital Ground for Digital Part
8	DP/GPIO1.3	B	-	-	USB D+ Port /GPIO 1 [3]
9	DM/GPIO1.4	B	-	-	USB D- Port/GPIO 1 [4]
10	ADDR3/GPIO2.3	B	U	4	Lower Address[3]
11	ADDR2/GPIO2.2	B	U	4	Lower Address[2]
12	ADDR1/GPIO2.1	B	U	4	Lower Address[1]
13	ADDR0/GPIO2.0	B	U	4	Lower Address[0]
14	LD0	B	U	4	80C51 P0[0] Port
15	LD1	B	U	4	80C51 P0[1] Port
16	LD2	B	U	4	80C51 P0[2] Port
17	VDD3.3	DP	-	-	External Digital Power for Digital Part(3.3V)
18	GND3.3	DG	-	-	External Digital Ground for Digital Part
19	LD3	B	U	4	80C51 P0[3] Port
20	LD4	B	U	4	80C51 P0[4] Port
21	LD5	B	U	4	80C51 P0[5] Port
22	LD6	B	U	4	80C51 P0[6] Port
23	LD7	B	U	4	80C51 P0[7] Port
24	P22/KEY2	B	U	4	80C51 P2[2] Port
25	MCU_PSENB	B	U	4	80C51 PSENB/GPIO 0 [7]
26	VDD1.8	DP	-	-	Internal Digital Power for Digital Part(1.8V)
27	GND1.8	DG	-	-	Internal Digital Ground for Digital Part
28	P10	B	U	4	80C51 P1[0] Port
29	P11	B	U	4	80C51 P1[1] Port
30	P12	B	U	4	80C51 P1[2] Port
31	P13	B	U	4	80C51 P1[3] Port
32	P14	B	U	4	80C51 P1[4] Port
33	P15	B	U	4	80C51 P1[5] Port
34	P16	B	U	4	80C51 P1[6] Port
35	P17	B	U	4	80C51 P1[7] Port
36	GND3.3	DG	-	-	External Digital Ground for Digital Part
37	VDD3.3	DP	-	-	External Digital Power for Digital Part(3.3V)
38	P30/ RXD	B	U	4	80C51 P3[0] Port/UART RXD
39	P31/ TXD	B	U	4	80C51 P3[1] Port/UART TXD
40	P32/INT0	B	U	4	80C51 P3[2] Port/INTERRUPT0
41	P33/INT1	B	U	4	80C51 P3[3] Port/INTERRUPT1
42	P34/ T0	B	U	4	80C51 P3[4] Port/TIMER0 CLOCK
43	P35/ T1	B	U	4	80C51 P3[5] Port/TIMER1 CLOCK
44	P36/Nr_WR	B	U	4	80C51 P3[6] Port/Nor Flash Memory Write Enable
45	P37/Nr_RD	B	U	4	80C51 P3[7] Port/ Nor Flash Memory Read Enable
46	MCU_ALE	B	U	4	80C51 ALE/GPIO 0 [6]

No.	Pin Name	I/O	U/D	Io[mA]	Description
47	GND1.8	DG	-	-	Internal Digital Ground for Digital Part
48	VDD1.8	DP	-	-	Internal Digital Power for Digital Part(1.8V)
49	HD8/GPIO8.0	B	U	4	Nor Flash Memory/LCD Data 8/GPIO8.0
50	HD9/GPIO8.1	B	U	4	Nor Flash Memory/LCD Data 9/GPIO8.1
51	HD10/GPIO8.2	B	U	4	Nor Flash Memory/LCD Data 10/GPIO8.2
52	HD11/GPIO8.3	B	U	4	Nor Flash Memory/LCD Data 11/GPIO8.3
53	HD12/GPIO8.4	B	U	4	Nor Flash Memory/LCD Data 12/GPIO8.4
54	HD13/GPIO8.5	B	U	4	Nor Flash Memory/LCD Data 13/GPIO8.5
55	HD14/GPIO8.6	B	U	4	Nor Flash Memory/LCD Data 14/GPIO8.6
56	HD15/GPIO8.7	B	U	4	Nor Flash Memory/LCD Data 15/GPIO8.7
57	GND3.3	DG	-	-	External Digital Ground for Digital Part
58	VDD3.3	DP	-	-	External Digital Power for Digital Part(3.3V)
59	SDO/GPIO9.0	B	U	4	Audio Codec Data Output
60	SDI/GPIO9.1	B	U	4	Audio Codec Data Input
61	CCK/GPIO9.2	B	U	4	Audio Codec Channel Clock
62	SCK/GPIO9.3	B	U	4	Audio Codec Sample Clock
63	MCK/GPIO9.4	B	U	4	Audio Codec Master Clock
64	MCLK/GPIO9.5	B	U	4	MMC Clock
65	MDAT/GPIO9.6	B	U	4	MMC Data
66	MCMD/GPIO9.7	B	U	4	MMC Command
67	GND1.8	DG	-	-	Internal Digital Ground for Digital Part
68	VDD1.8	DP	-	-	Internal Digital Power for Digital Part(1.8V)
69	P23/KEY3	B	U	4	80C51 P2[3] Port
70	P21/KEY1	B	U	4	80C51 P2[1] Port
71	P20/KEY0	B	U	4	80C51 P2[0] Port
72	P25/KEY5	B	U	4	80C51 P2[5] Port
73	P26/KEY6	B	U	4	80C51 P2[6] Port
74	PORT0/ M_DAT1	B	U	4	General Port 0 [0]/M_DAT1
75	PORT1/ M_DAT2	B	U	4	General Port 0 [1]/M_DAT2
76	PORT2/ M_DAT3	B	U	4	General Port 0 [2]/M_DAT3
77	PORT3	B	U	4	General Port 0 [3]
78	PORT4	B	U	4	General Port 0 [4]
79	PORT5	B	U	4	General Port 0 [5]
80	GND3.3	DG	-	-	External Digital Ground for Digital Part
81	VDD3.3	DP	-	-	External Digital Power for Digital Part(3.3V)
82	P27/KEY7	B	U	4	80C51 P2[7] Port
83	P24/KEY4	B	U	4	80C51 P2[4] Port
84	ADDR7/GPIO2.7	B	U	4	Lower address[7]
85	ADDR6/GPIO2.6	B	U	4	Lower address[6]
86	ADDR5/GPIO2.5	B	U	4	Lower address[5]
87	ADDR4/GPIO2.4	B	U	4	Lower address[4]
88	F_RNB/GPIO1.2	B	U	4	Flash Ready/Busy/GPIO 1 [2]
89	FD7/NA15/GPIO7.7	B	U	4	Flash Data7/Nor Flash Memory Addr15
90	FD6/NA14/GPIO7.6	B	U	4	Flash Data6/Nor Flash Memory Addr14
91	FD5/NA13/GPIO7.5	B	U	4	Flash Data5/Nor Flash Memory Addr13
92	FD4/NA12/GPIO7.4	B	U	4	Flash Data4/Nor Flash Memory Addr12
93	FD3/NA11/GPIO7.3	B	U	4	Flash Data3/Nor Flash Memory Addr11
94	FD2/NA10/GPIO7.2	B	U	4	Flash Data2/Nor Flash Memory Addr10
95	FD1/NA9/GPIO7.1	B	U	4	Flash Data1/Nor Flash Memory Addr9
96	FD0/NA8/GPIO7.0	B	U	4	Flash Data0/Nor Flash Memory Addr8

No.	Pin Name	I/O	U/D	Io[mA]	Description
97	FCEN3/GPIO10.7	B	U	4	Nand Flash Chip Select3
98	FCEN2/GPIO10.6	B	U	4	Nand Flash Chip Select2
99	FCEN1/GPIO10.5	B	U	4	Nand Flash Chip Select1
100	FCEN0/GPIO10.4	B	U	4	Nand Flash Chip Select0/Nor Flash Memory Chip Select
101	VDD1.8	DP	-	-	Internal Digital Power for Digital Part(1.8V)
102	GND1.8	DG	-	-	Internal Digital Ground for Digital Part
103	FCLE/GPIO10.3	B	U	4	Nand Flash Command Latch Enable
104	FALE/GPIO10.2	B	U	4	Nand Flash Address Latch Enable
105	FWEN/GPIO10.1	B	U	4	Nand Flash Write Enable
106	FREN/GPIO10.0	B	U	4	Nand Flash Read Enable
107	VDD3.3	DP	-	-	External Power for Digital Part(3.3V)
108	GND3.3	DG	-	-	External Ground for Digital Part
109	XIN	I	-	-	Crystal Input
110	XOUT	O	-	4	Crystal Output
111	AVDD1.8	AP	-	-	Analog Power for Analog Part(1.8V)
112	AGND1.8	AG	-	-	Analog Ground for Analog Part
113	N.C.	O	-	-	No Connection
114	RTC_GND,GND	DG	-	-	Digital Ground for RTC Part
115	RTC_VDD1.8,VDD1.8	DP	-	-	Digital Power for RTC Part(1.8V)
116	GND	AG	-	-	Ground
117	N.C	-	-	-	No Connection
118	AIN3	AI	-	-	ADC Analog Input3
119	AIN2	AI	-	-	ADC Analog Input2
120	AIN1	AI	-	-	ADC Analog input1
121	AIN0	AI	-	-	ADC Analog Input0
122	VBOT	AI	-	-	ADC Reference Bottom
123	VTOP	AI	-	-	ADC Reference Top
124	RTC_GND3.3	DG	-	-	Digital Ground for RTC Part
125	RTC_VDD3.3	DP	-	-	Digital Power for RTC Part(3.3V)
126	NRST	I	S	-	Chip reset
127	XRM/GPIO1.0	B	U	4	External Data RAM chip select/GPIO1[0]
128	LCD/GPIO1.1	B	U	4	External LCD chip select/GPIO 1 [1]