九齊科技股份有限公司
Nyquest Technology Co., Ltd.

USER MANUAL

NX1 Series

Single-Chip 32-bit MCU with CELP, SBC, ADPCM Coding & 16-ch MIDI

Version 2.2

Jun. 2, 2020

Revision History

Version	Date	Description	Modified Page
1.0	2016/11/30	Formal release.	-
1.1	2017/02/24	1. Update NX1 line-up. 2. Modify block diagram. 3. Modify pin names and description. 4. Modify development environment. 5. Update NX1 hardware pictures.	9 12 13 77 81, 82
1.2	2017/05/31	1. Add LVR values per I_HRC conditions. 2. Rename VDD_SPI0 to SPI0_VDD 3. DC Characteristics. 4. Modify NX1 Memory Map. 5. 5. Correction at Halt mode.	10 13 15, 16 19 22
1.3	2017/08/30	1. Modify PWM-IO from 4 pins to 8 pins for NX12P54A / NX12P64A. 2. Modify LVR values. 3. Change NX12P44AU24 SSOP24 to NX12P44AQ32 LQFP-32. 4. Rename pin names from ADC# to AIN#. 5. Change Core LDO voltage.	9 10 12 13 77
1.4	2017/11/30	1. Update Push-Pull output power from 1.3-watt to 1.5-watt. 2. Add SOP-16 & SOP-8 MCP packages. 3. Update product line-up for NX11P21A. 4. Update SBC bandwidth from 12KHz to 16KHz. 5. Add pad description for PA2/AIN2/RSTb/Vpp. 6. Update DC/AC Characteristics. 7. Add DAC Characteristics. 8. Update interrupt vector table. 9. Update initial values for various registers. 10. Add the description for GPIO wake-up & external INT. 11. MIX_EN control bit changed to DAC1 Control Register. 12. Modify Configuration Options.	8, 10 8, 12 9 12 13 15, 16 17 22 23 ~ 76 27 75 77

Version	Date	Description	Modified Page
1.5	2018/02/24	1. Modify Trim accuracy for LO_CLK. 2. Update Product Line-Up of NX1 Series. 3. Update Configuration Options. 4. Add user-configurable LVR options. 5. Correct the typo for DAC output. 6. Modify RTC interrupt options. 7. Add MCP feature. 8. Modify Block Diagram. 9. Add PA2 / RSTb / Vpp for NX11M2xA. 10. Modify Figure 18 Development Environment of NX1 Series.	10 10 11, 79 11 12 13, 68, 69 13 14 15 80
1.6	2018/05/31	1. Improve Trim accuracy for LO_CLK. 2. 1.8V Option of SPI0_VDD not supported for NX11M. 3. PWMA / CSC not supported for NX11P21A / NX11M. 4. S/W-based algorithm appended with VR, Voice Changer, Ultrasound communication, etc. 5. OTP / SPI Flash programming interface pins MUX with PA2 for NX11P21A / NX11M. 6. Push-Pull PA output power added for NX11P21A / NX11M.	10 10 12, 15, 16 13 15, 16 19
1.7	2018/08/31	1. Remove NX12P24A body. 2. Adjust LVR settings. 3. Modify Pad Description for NX12P44A / NX12P64A. 4. Add external interrupt 1 for NX12P64A.	10 11, 80 15, 16 34, 35
1.8	2018/11/28	Update LVR Voltage option for HI_CLK @ 32MHz.	11, 81
1.9	2019/01/17	1. Add NX13P44A and NX13P64A. 2. Swap the contents of ALT1 for ALT2 (PA4 - PA7). 3. Modify the description of PA2/PA14/PA15Modify GPIO's description. 4. Add PA14_MFP1 for NX12P64A and later products.	10 15 15,16 32
2.0	2019/11/22	1. Line-up updated. 2. Add NX11S series. 3. Add 4.5Kbps for SBC.	11 12, 13, 15 14

Version	Date	Description	Modified Page
2.1	2020/03/20	1. Add description for Test pad. 2. ALT1 and ALT2 swapped for PA14 pin of NX12P64A / NX13P64A / NX12M / NX13M.	18 17
2.2	2020/06/02	1. Rename programming pin names to avoid confusion. 2. Update Product Line-Up. 3. P_Pull_High_CFG typo correction.	16 11 31, 37

Table of Contents

List of Figures

List of Tables

Chapter 1. Introduction

1.1 General Description

The NX1 series is a 32-bit MCU based high-quality speech/MIDI processor, which is specially designed for customers to innovate with advanced DSP power. It is embedded with OTP (One Time PROM) for mass production, such that no mask is required while MOQ / Lead Time are kept minimized.

With Instruction / Data Local Memory bus (ILM/DLM) built in CPU, the NX1 can run 1.57 DMIPS per MHz and up to 50+ DMIPS @ 32MHz. In addition, the dual clock design let customers switch between fast / slow clocks for achieving the best power consumption and performance ratio, or may employ 32.768KHz X'tal oscillator for time-keeping applications.

The NX1 series consists of several derivatives with respect to ROM (OTP), RAM, I/O and functions. With memory-mapped architecture, the NX1 can address up to 16MB space that includes memory, register files, peripheral and SPI flash storage (including instruction / data modes). SBC (Sub-Band Coding) is achieved with greatly enhanced quality & much less memory size compared against traditional ADPCM coding due to the incorporation of efficient DSP algorithms as well as the upgrade of H/W spec. Via the high performance of 32-bit MCU, the S/W-based MIDI synthesizer can reach more than 16-ch polyphonic channels. All data including SBC / MIDI files, wavetable timbres, XIP codes and general user data, can be accessed from the external SPI flash.

There are various useful features inside the NX1 series: Three sets of 16-bit Timers; 8-ch H/W PWM-IO pins, which provide complimentary outputs (with dead zone generator) and capture feature; 8-channel, 12-bit SAR ADC with MIC pre-amplifier & AGC / PGA that supports differential MIC input; 14-bit DAC + 1.5-watt Push-Pull power amplifier to drive speaker directly; independently configurable GPIO per pin with alternate functions; IR TX that supports 38KHz / 57KHz / 125KHz / 500KHz carrier for Infrared or QFID applications; SPI0 for SPI flash control, powered by embedded optional 3.3V / 1.8V LDO, which supports single/dual/quad I/O mode with XIP (e**X**ecute **I**n **P**lace) capability; SPI1 for 2.4GHz RF module or other SPI devices; SDHC 2.0-compliant memory card; I^2C H/W interface and UART TX / RX for serial communication.

Except developing by C language at *NYIDE* environment, which provides customers with more controllability over complicated projects. Moreover, The NX1 series brings *Q-Code* (High-level programming) to 32-bit MCU, which provides customers with an easy-to-use, highly productive development environment to cope with the importance of in-time product concept realization. The multi-purpose NX_Programmer (USB-2-Serial adaptor) H/W provides customers with various functionalities: program code debugging @ ICE, prototype demo @ FDB (Flash Demo Board) and mass production @ OTP. Besides, NX_Programmer can also provide ICP (In-Circuit Programming) function to program SPI flash for customers to fabricate PCBA in advance.

Various package forms are available for the NX1 series: Dice, SOP-16, SSOP-24, LQFP-32, LQFP-48, LQFP-64 and the innovative SOP-8 / SOP-16 MCP (NX11M2xA Multi-Chip Package), where SPI flash is stacked inside for applications that require small footprints.

1.2 Features

- Wide Operating Voltage: 2.4V ~ 5.5V
  - SPI flash @ SPI0 powered by embedded LDO, 3.3V (default) / 1.8V (not for NX11P21A, NX11M) optional.
  - Min. operating voltage is 3.0V @ 32MHz maximum CPU clock, and 2.2V @ 12MHz minimum CPU clock.
- 32-bit CPU core Andes N705-S, like ARM Cortex-M0+.
  - Max. CPU clock: 32MHz @ 3.3V, 50+ DMIPS
  - ILM / DLM (Instruction / Data Local Memory)
  - 16MB addressing space
  - 24 H/W interrupts with 4 programmable priority levels
  - 1-cycle fast multiplier
  - Benchmarking: 1.57 DMIPS per MHz, 2.57 CoreMark per MHz
- Product line-up

P/N	RAM	OTP	Flash	I/O	SPI 0/1	16-bit Timer	PWM-IO	12-bit ADC	MIC	I^2C	UART	X'tal	Push-Pull PA	VR	Package
NX11P21A	4KB	32KB	-	18	v / -	2	-	-	-	-	-	-	v	-	Dice, SOP-16, SSOP-24
NX11P22A	4KB	32KB	-	24	v / -	2	4	-	-	-	-	v	v	-	Dice, SSOP-24, LQFP-48
NX12P44A	8KB	64KB	-	32	v / v	3	4	8-ch	v	v	v	v	v	v	Dice, LQFP-32, LQFP-64
NX12P64A	12KB	64KB	-	32	v / v	3	8	8-ch	v	v	v	v	v	v	Dice, SSOP-24, LQFP-48
NX13P44A	8KB	64KB	-	32	v / v	3	4	8-ch	v	v	v	v	v	-	Dice, LQFP-32, LQFP-64
NX13P64A	12KB	64KB	-	32	v / v	3	8	8-ch	v	v	v	v	v	-	Dice, SSOP-24, LQFP-48
NX11S21A	4KB	32KB	2Mb	12	v / -	2	-	-	-	-	-	-	v	-	SOP-8, SOP-16
NX11S22A	4KB	32KB	4Mb	12	v / -	2	-	-	-	-	-	-	v	-	SOP-8, SOP-16
NX11M22A	4KB	32KB	4Mb	12	v / -	2	-	-	-	-	-	-	v	-	SOP-8, SOP-16
NX11M23A	4KB	32KB	8Mb	12	v / -	2	-	-	-	-	-	-	v	-	SOP-8, SOP-16
NX11M24A	4KB	32KB	16Mb	12	v / -	2	-	-	-	-	-	-	v	-	SOP-8, SOP-16
NX11M25A	4KB	32KB	32Mb	12	v / -	2	-	-	-	-	-	-	v	-	SOP-8, SOP-16
NX12M52A	10KB	96KB	4Mb	16	v / -	3	4	4-ch	v	v	v	v	v	v	SOP-16, SSOP-24
NX12M53A	10KB	96KB	8Mb	16	v / -	3	4	4-ch	v	v	v	v	v	v	SOP-16, SSOP-24
NX12M54A	10KB	96KB	16Mb	16	v / -	3	4	4-ch	v	v	v	v	v	v	SOP-16, SSOP-24
NX12M55A	10KB	96KB	32Mb	16	v / -	3	4	4-ch	v	v	v	v	v	v	SOP-16, SSOP-24
NX13M52A	10KB	96KB	4Mb	16	v / -	3	4	4-ch	v	v	v	v	v	-	SOP-16, SSOP-24
NX13M53A	10KB	96KB	8Mb	16	v / -	3	4	4-ch	v	v	v	v	v	-	SOP-16, SSOP-24
NX13M54A	10KB	96KB	16Mb	16	v / -	3	4	4-ch	v	v	v	v	v	-	SOP-16, SSOP-24
NX13M55A	10KB	96KB	32Mb	16	v / -	3	4	4-ch	v	v	v	v	v	-	SOP-16, SSOP-24

Table 1 Product Line-Up of NX1 Series

- Dual Clock Operation, either internal oscillators or external X'tal.
  - System clock @ HI_CLK (32MHz) and LO_CLK (32,768Hz)
  - Built-in oscillators for I_HRC ((Internal High RC Oscillator, max. 32MHz) and I_LRC (Internal Low RC Oscillator, 32,768Hz), accuracy trimmed to +/-0.5% for HI_CLK and +/-1.5% for LO_CLK.
  - External X'tal @ Xin / Xout pins available for either E_HXT (External High Crystal Oscillator) or E_LXT (External Low Crystal Oscillator)
  - CPU runs @ system clock while RTC connected to 32.768KHz X'tal for precise time-keeping purpose.

- Power management to support 4 operating modes for high performance and power saving.
  - Normal mode: CPU running @ HI_CLK (12MHz / 16MHz / 24MHz / 32MHz)
  - Slow mode: CPU running @ LO_CLK (32.768KHz)
  - Standby mode: CPU stopped, only peripherals with LO_CLK as source (like RTC and Timer) are active.
  - Halt mode: everything stops, I_{HALT} < 1uA
- Code Options for customers to configure the system.
  - HI_CLK: 32MHz (default) / 24MHz / 16MHz / 12MHz
  - HI_CLK source: I_HRC (Internal High RC) / E_HXT (External High X'tal)
  - LO_CLK (32.768KHz) source: I_LRC (Internal Low RC) / E_LXT (External Low X'tal)
  - SPI0_VDD voltage for SPI flash: 3.3V (default) / 1.8V (not for NX11P21A, NX11S2xA, NX11M2xA)
  - VDD voltage: 3.0V / 4.5V (default)
  - WDT: Enable (default) / Disable
  - External Reset: Enable (default)
  - LVR Voltage
- LVD (Low Voltage Detection)
  - Total 6-level options: 3.6V, 3.4V, 3.2V, 2.6V, 2.4V, 2.2V (+/-0.1V accuracy)
- LVR (Low Voltage Reset)
  - User-configurable, default values are 2.8V @ 32MHz for NX12, 2.7V @ 32MHz for NX11, 2.4V @ 24MHz, 2.0V @ 16MHz, 1.8V @ 12MHz.
- Timers (Timer0 / Timer1 / Timer2)
  - Each Timer consists of divider and 16-bit down-counter with various clock sources.
  - 16-bit counter is readable on the fly and register-controllable for automatic re-load upon underflow.
- Two PWM Generators (PWMA / PWMB)
  - Each generator consists of a clock divider, a 16-bit timer (that can be used as a general timer) and four 16-bit duty cycle registers.
  - Up to 4 independent PWM channels per generator (PWMA0 ~ PWMA3 / PWMB0 ~ PWMB3)
  - Provide complementary outputs @ PWMA0 / PWMA1 and PWMB0 / PWMB1, with dead-zone generator
  - Capture supported @ PWMA
    - 16-bit counter shared with PWMA
    - Support rising/falling-edge capture interrupt of capture inputs (PA0 ~ PA15)
    - Re-load counter automatically upon interrupt
- ADC (Analog to Digital Conversion)
  - 8-ch, 12-bit resolution SAR (Successive Approximate Register) ADC
  - Support pre-charge to ½ * VDD_ADC by register control
  - Analog input voltage range: 0 ~ VDD_ADC
  - Separate VREF_ADC pin supported
  - Trigger by underflow of Timer 0 / 1 / 2, or software manually

- Built-in MIC bias, 2-stage of pre-amplifiers and AGC/PGA for gain control
- Built-in DAC + Push-Pull power amplifier
  - DAC: voltage-type, true 14-bit resolution, stereo DAC (only for NX12P46 / NX12P88)
  - Power amplifier: 1.3-Watt / 1.5-Watt (for NX11P22A / NX12P44A only) @ 4-ohm speaker, BTL, VDD=5V
    - Voltage-type DAC output is available @ PP1 pin to use external amplifiers
    - Stereo DAC output is available @ PP1 / PP2 pins to connect to stereo PA externally
  - Line-In mixed with speech/MIDI output @ power amplifier to drive speaker
- GPIO
  - Bit configurability for every I/O pin by register control, except pull-high value by byte
    - Direction, data, key change wake-up, pull-high, pull-high value, sink type
  - Multi-Function via register control
  - Up to 40-pin GPIO
  - 8 pins (PA8 ~ PA15) supported with CSC (Constant Sink Current) capability (not for NX11P21A / NX11S / NX11M)
- IR TX, programmable with following features
  - Carrier frequency (38K / 57KHz / 125KHz / 500KHz)
  - Stop state @ high / low (1 bit)
  - Mask Enable / Disable (1 bit)
- SPI0 (mainly for SPI flash) / SPI1 (for communication)
  - Master mode
  - Up to 32MHz clock speed
  - 32-bit mode supported to address more than 128Mb SPI flash
  - Support Data mode and XIP mode (e**X**ecute **I**n **P**lace, for SPI0 only)
  - Support Single / Dual / Quad I/O mode of SPI flash (SPI1 for single I/O only)
  - Polarity and Phase functions supported (SPI flash Mode 0 and Mode 3)
- UART
  - Full-duplex TX/RX with 4-level FIFO
  - Baud rate @ 4,800 / 9,600 / 38,400 / 115,200 bps
- I^2C
  - Master / Slave mode
  - 100KHz / 400KHz / 1MHz Clock
  - 4-byte FIFO
- SD Card Controller
  - 8-word TX / RX FIFO
  - Protocols supported
    - SD memory card protocol ver. 2.0

- SDIO bus protocol ver. 1.1
- MMC bus protocol ver. 4.1

  - 1-bit / 4-bit data modes
  - Sector addressing @ SDHC to reach 32GB of memory cards

- RTC
  - Support 16KHz (or 4KHz) / 1KHz / 64Hz / 2Hz interrupts
- WDT (Watch-Dog Timer)
  - Support 188ms / 750ms resets
- MCP (Multi-Chip Package) for NX11M2xA series
  - SPI Flash stacked inside low-cost SOP-8 / SOP-16 packages
  - Support 4Mb / 8Mb / 16Mb / 32Mb SPI Flash density
  - SPI0 interface bonded inside the package, leaving more GPIO pins available
  - Built-in Push-Pull PA to drive speaker directly
  - Max. GPIO pins: 4 @ SOP-8, 12 @ SOP-16
  - Master or Slave operation
- OTP Protection Mechanism
  - OTP Security Lock: Yes (default) / No
- Easy-to-use Development Environment
  - High-level *Q-Code* programming
  - *NYIDE* for advanced programming with C language
  - Multi-purpose NX_Programmer with FDB, OTP and ICE.
  - Q-FDB_Writer and OTP_Writer also support NX1 series
- S/W-based Speech/MIDI Codec
  - ADPCM Codec (Adaptive Differential PCM): 4 bits / 5 bits per sample
  - SBC Codec (Sub-Band Coding): 4.5K ~ 32Kbps with maximum 16KHz bandwidth
  - CELP Decoder (Code-Excitation Linear Prediction): 4.8Kbps @ 8KHz SR for human voice only
  - MIDI: Up to 16-channel MIDI @ 32KHz Output Sample Rate
  - Voice Recognition, Voice Changer, Ultrasound communication, etc.
- Noise filter @ 4x Up-Sampling
- Shipping Form
  - Dice
  - Package: SOP-16 / SSOP-24 / LQFP-32 / LQFP-48 / LQFP-64, SOP-8 / SOP-16 MCP (only for NX11S / NX11M / NX12M / NX13M series).

1.3 Block Diagram

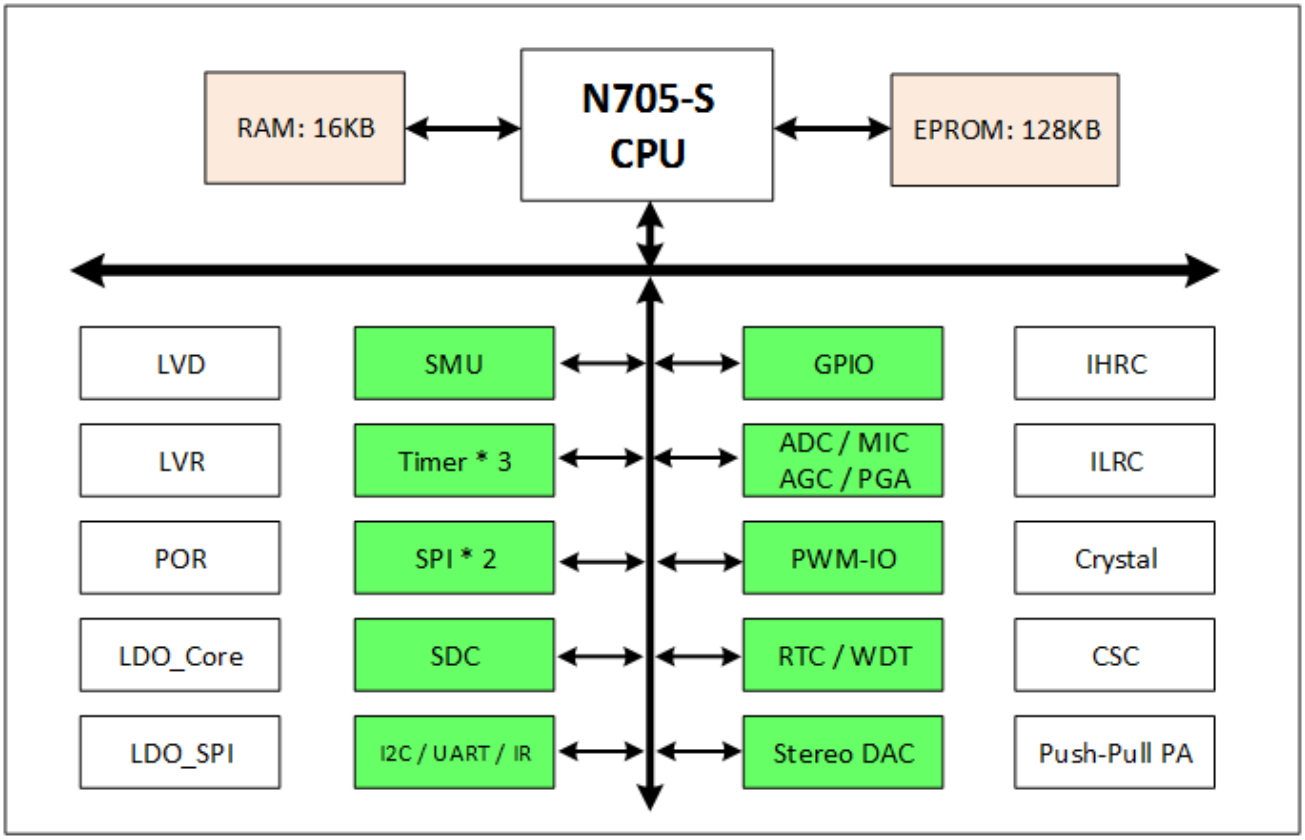


Figure 1 Block Diagram of NX1

1.4 Pad Description

Name	ALT 1	ALT 2	Type	Description
Power & Ground				
VDD#	-	-	P, I	Power input
VDD_ADC	-	-	AP, I	Power input for MIC / ADC / AGC
VREF_ADC	-	-	AP, I	ADC's Reference Power input
ADC_VDD	-	-	P, O	Power output for analog circuitry at NX12M/NX13M only. This pin must be connected with 0.1uF cap to ground and is bonded to VDD_ADC & VREF_ADC internally.
SPI0_VDD	-	-	P, O	Power output for SPI0 flash via internal LDO
VDD_SPI1	-	-	P, I	Power input for IO pins PB8 ~ PB11
VDD_SDC	-	-	P, I	Power input for IO port C, SDC interface
VSS#	-	-	P, I	Ground
AGC				
VMIC	-	-	AO	MIC bias
MICP	-	-	AI	MIC+
MICN	-	-	AI	MIC-
MICOUT	-	-	AO	Output of MIC pre-amp (for NX12P44A only)
OPI	-	-	AI	Input of OP Amp (for NX12P44A only)
AGC	-	-	AO	Auto Gain Control (for NX12P44A only)
PP / DAC				
ACIN	-	-	AI	Line in
PP1	DAC0	-	AO	Push-Pull PA output 1 or DAC0
PP2	DAC1	-	AO	Push-Pull PA output 2 or DAC1
Port A				
PA0	AIN0	-	I/O, AI	PA0 or Analog input 0
PA1	AIN1	-	I/O, AI	PA1 or Analog input 1
PA2/Vpp	AIN2	RSTb	I/O, AI I/P	PA2/Vpp (only for OTP / SPI Flash programming @ NX11P21A / NX11M2xA), Analog input 2. PA2 as RSTb pin if IC without ADC function, otherwise PA15 will be the RSTb pin. (Except NX12P44A / NX11P22A)
PA3/SCK	AIN3	-	I/O, AI	PA3/Programming Clock, or Analog input 3
PA4/MOSI	TM0	AIN4	I/O, AI, I	PA4/Programming Input, Analog input 4 or Timer input 0
PA5/MISO	IR	AIN5	I/O, AI, O	PA5/Programming Output, Analog input 5 or IR output
PA6	TXD	AIN6	I/O, AI, O	PA6, Analog input 6 or TX output of UART
PA7	RXD	AIN7	I/O, AI, I	PA7, Analog input 7 or RX input of UART
PA8	PWMA0	-	I/O, O	PA8 or PWMA0 output. Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
PA9	PWMA1	-	I/O, O	PA9 or PWMA1 output. Constant Sink Optional (not for NX11P21A / NX11S /

Name	ALT 1	ALT 2	Type	Description
				NX11M / NX12M / NX13M)
PA10	PWMA2	-	I/O, O	PA10 or PWMA2 output. Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
PA11	PWMA3	-	I/O, O	PA11 or PWMA3 output. Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
PA12	SDA	-	I/O, I/O	PA12 or I²C's SDA. Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
PA13	SCL	-	I/O, O	PA13 or I²C's SCL. Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
PA14	TM1	INT1	I/O, I, I	PA14, Timer input 1, or ext. Interrupt 1 (For NX12P64A / NX12M / NX13M only). Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
PA15/Vpp	INT0	RSTb	I/O, I, I	PA15/Vpp (only for OTP / SPI Flash programming @ NX12P64A), ext. Interrupt 0. PA15 as RSTb pin if IC with ADC function, otherwise PA2 will be the RSTb pin. (Except NX12P44A / NX11P22A) Constant Sink Optional (not for NX11P21A / NX11S / NX11M / NX12M / NX13M)
Port B				
PB0	SPI0_CSb	-	I/O, O	PB0 or SPI0's CSb
PB1	SPI0_SCK	-	I/O, O	PB1 or SPI0's SCK
PB2	SPI0_IO0	-	I/O, I/O	PB2 or SPI0's IO0 (MOSI)
PB3	SPI0_IO1	-	I/O, I/O	PB3 or SPI0's IO1 (MISO)
PB4	SPI0_IO2	-	I/O, I/O	PB4 or SPI0's IO2
PB5	SPI0_IO3	-	I/O, I/O	PB5 or SPI0's IO3
PB6	Xin	-	I/O, I	PB6 or Xin of X'tal
PB7	Xout	-	I/O, O	PB7 or Xout of X'tal
PB8	SPI1_CSb	-	I/O, O	PB8 or SPI1's CSb
PB9	SPI1_SCK	-	I/O, O	PB9 or SPI1's SCK
PB10	SPI1_SDO	-	I/O, O	PB10 or SPI1's SDO (MOSI)
PB11	SPI1_SDI	-	I/O, I	PB11 or SPI1's SDI (MISO)
PB12	PWMB0	-	I/O, O	PB12 or PWMB0 output
PB13	PWMB1		I/O, O	PB13 or PWMB1 output
PB14	PWMB2	-	I/O, O	PB14 or PWMB2 output
PB15	PWMB3	-	I/O, O	PB15 or PWMB3 output
Port C				
PC0	SDC_CLK	-	I/O, O	PC0 or CLK for SDC
PC1	SDC_CMD	-	I/O, I/O	PC1 or CMD for SDC
PC2	SDC_IO0	-	I/O, I/O	PC2 or IO0 for SDC
PC3	SDC_IO1	-	I/O, I/O	PC3 or IO1 for SDC

Name	ALT 1	ALT 2	Type	Description
PC4	SDC_IO2	-	I/O, I/O	PC4 or IO2 for SDC
PC5	SDC_IO3	-	I/O, I/O	PC5 or IO3 for SDC
PC6	SDC_CD	-	I/O, I	PC6 or CD for SDC
PC7	SDC_WP	-	I/O, I	PC7 or WP for SDC
Other				
Test	-	-	I	Test pin, which should be left open for normal operation.
RSTb/Vpp	-	-	I/P	Reset pin / Vpp (for OTP / SPI Flash programming @ NX11P22A / NX12P44A only)

Pad Type: P = Digital Power, I = Digital Input, O = Digital Output, AI = Analog Input, AO = Analog output, AP = Analog Power.

1.5 Electrical Characteristics

1.5.1 Absolute Maximum Rating

Symbol	Parameter	Rated Value	Unit
V_{DD} - V_{SS}	Supply voltage	-0.5 ~ +7.5	V
V_{IN}	Input voltage	V_{SS} - 0.3 ~ V_{DD} + 0.3	V
T_{OP}	Operating Temperature	0 ~ +70	°C
T_{ST}	Storage Temperature	-25 ~ +85	°C

1.5.2 DC Characteristics *(T_A=25ºC, unless otherwise specified)*

Symbol	Parameter		V_{DD}	Min.	Typ.	Max.	Unit	Condition
V_{DD}	Operating voltage		-	3.0	4.5	5.5	V	CPU_CLK=32MHz
				2.7	4.5	5.5		CPU_CLK=24MHz
				2.2	3.0	5.5		CPU_CLK=16MHz
				2.2	3.0	5.5		CPU_CLK=12MHz
				2.2	3.0	5.5		CPU_CLK=32.768KHz
I_{HALT}	Halt Current		3		0.1		uA	CPU stop, all functions off, Disable SPI0_VDD*1
			4.5		0.1			
			3		1.1		uA	CPU stop, all functions off, Enable SPI0_VDD*1
			4.5		1.3			
I_{SB}	Standby Current		3		3.5		uA	CPU stop, all functions off, RTC on, Enable SPI0_VDD*1
			4.5		4.5			
I_{OP}	Operating Current	Slow Mode	3		58.3		uA	CPU_CLK=32.768KHz, Enable SPI0_VDD*1
			4.5		75			
		Normal Mode	3		6.1		mA	CPU_CLK = 12MHz, Core_LDO = 2.3V, Enable SPI0_VDD*2
			3		7.5		mA	CPU_CLK = 16MHz, Core_LDO = 2.3V, Enable SPI0_VDD*2
			4.5		12.0		mA	CPU_CLK = 24MHz, Core_LDO = 2.7V, Enable SPI0_VDD*2
			4.5		19.1		mA	CPU_CLK = 32MHz, Core_LDO = 3.3V, Enable SPI0_VDD*2

Symbol	Parameter		V_{DD}	Min.	Typ.	Max.	Unit	Condition
I_{IL}	Input current (Internal pull-high)	Weak (1MΩ)	3		-2.7		uA	V_{IL} = 0V
			4.5		-7.2			
		Strong (100KΩ)	3		-30			
			4.5		-78			
		SDC pad (20KΩ)	3		-144			
			4.5		-215			
I_{OH}	Normal drive current (PA, PB[7:6], PB[15:12])		3		-8.7			V_{OH} = 2.0V
			4.5		-13.7			V_{OH} = 3.5V
	Normal drive current (SPI0, SPI1, SDC)		3		-12.9		mA	V_{OH} = 2.0V
			4.5		-20.2			V_{OH} = 3.5V
	Large drive current (SPI0, SPI1, SDC)		3		-24.8			V_{OH} = 2.0V
			4.5		-38.0			V_{OH} = 3.5V
I_{OL}	Normal sink current		3		12.4		mA	V_{OL} = 1.0V (CSC, constant sink current, not for NX11P21A and NX11M)
			4.5		19.2			
	Large sink current		3		24.3			
			4.5		37.1			
	Normal constant sink current (PA[15:8])		3		13			
			4.5		13			
	Large constant sink current (PA[15:8])		3		20			
			4.5		20			
ΔF/F	Frequency deviation by voltage drop (I_HRC=32MHz/24MHz)		4.5		-0.5		%	$\frac{Fosc(4.5v) - Fosc(3.3v)}{Fosc(4.5v)}$
	Frequency deviation by voltage drop (I_HRC=16MHz/12MHz)		3		-0.5			$\frac{Fosc(3.0v) - Fosc(2.4v)}{Fosc(3.0v)}$
			4.5		-0.5			$\frac{Fosc(4.5v) - Fosc(3.0v)}{Fosc(4.5v)}$
ΔF/F	Frequency deviation by lot		3	-0.5		0.5	%	$\frac{Fosc(3.0v) - Ftyp(3.0v)}{Ftyp(3.0v)}$

*[*1] Light-Load LDO @ SPI0_VDD is rated at 5mA for lowering power consumption @ Halt, Standby, and Slow modes.*

*[*2] Heavy-Load LDO @ SPI0_VDD is rated at 40mA for read / erase / write operation to the SPI Flash @ Normal mode.*

1.5.3 ADC Characteristics *(V_{DD}=3.3V, T_A=25ºC, unless otherwise specified)*

Symbol	Characteristics	Min.	Typ.	Max.	Unit
V_{INL}	ADC LINE_IN Input Voltage Range from PA[7:0]	V_{SS} - 0.3		V_{DD} + 0.3	V
V_{INM}	ADC Microphone Input Voltage Range	V_{SS} - 0.3		V_{DD} + 0.3	V
B_{RES}	Resolution of ADC			12	Bit
ENOB	Effective Number of Bits		10		Bit
INL	Integral Non-Linearity of ADC		+/-4		LSB
DNL	Differential Non-Linearity of ADC		+1.5 / -0.9		LSB
F_{CONV}	AD Conversion Rate			SYS_CLK/184	Hz

1.5.4 DAC Characteristics *(V_{DD}=5V, T_A=25ºC, unless otherwise specified)*

Symbol	Characteristics	Min.	Typ.	Max.	Unit	Condition
B_{RES}	Resolution of DAC			14	Bit	-
DR	Dynamic Range (Vin = -60 dBFS)		-73		dBr A	No Load
SNR	Noise at No Signal (Vin = -90 dBFS)		-97		dBr A	
P_O	THD+N 1%		0.7		W	4Ω Load
	THD+N 10%		1.3[*1] 1.5[*2]		W	

[*1] *1.3-Watt for NX11P21A / NX11M / NX12P64A.*

[*2] *1.5-Watt for NX11P22A / NX12P44A.*

Chapter 2. Hardware Architecture

2.1 N705-S Core

The N705-S processor is a 2-stage, 32-bit RISC processor. It has an AMBA AHB-Lite interface and includes an Internal Vectored Interrupt Control (IVIC) component. It is built with a fast multiplier and 4-level programmable-priority interrupt control.

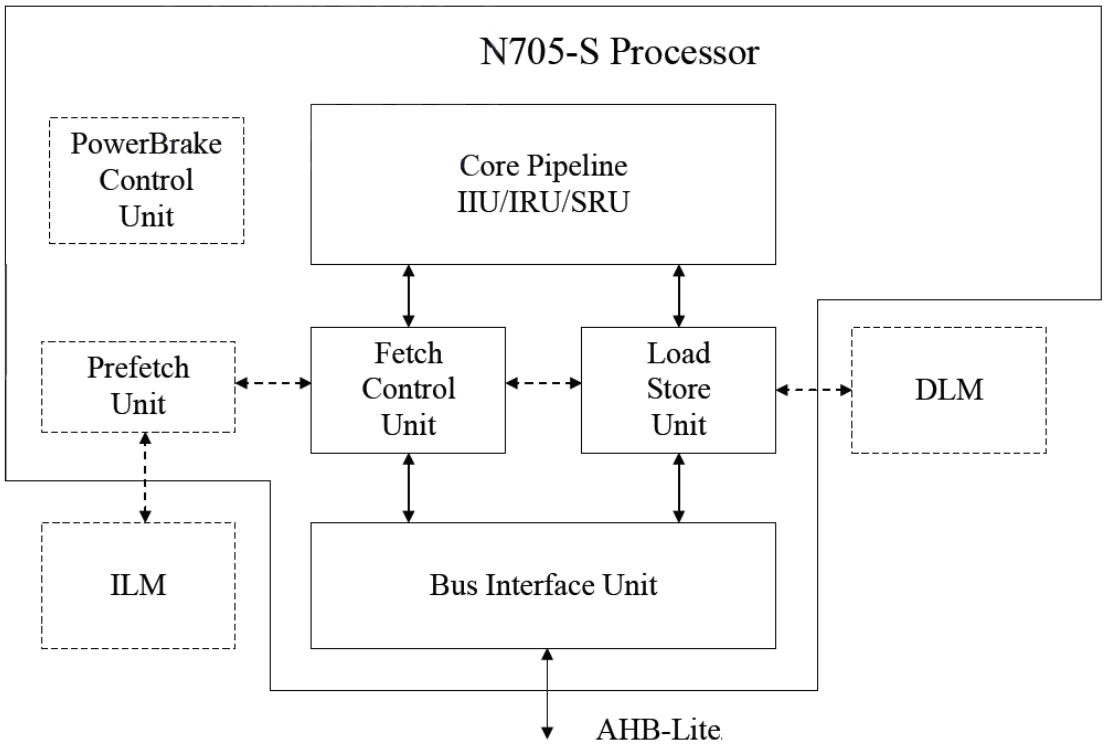


Figure 2 Block diagram of N705-S Core

The implemented device provides:

- 2-stage pipeline design
- 2R1W register file
- 16/32-bit mixed instructions with 16 GPRs (General-Purpose Registers)
- Hardware 1-cycle multiplier (32x32=32)
- Addressable space: 16MB (24-bit address)
- Interrupt vectored interrupt control: 4-level programmable-priority

2.2 Memory Map

The NX1 series provides a 16M-byte address space for programmers. The memory allocation of all peripheral blocks is shown in Table 2. The detailed register, memory addressing and programming will be

described in the following sections for individual on-chip modules. The NX1 series only supports little-endian data format.

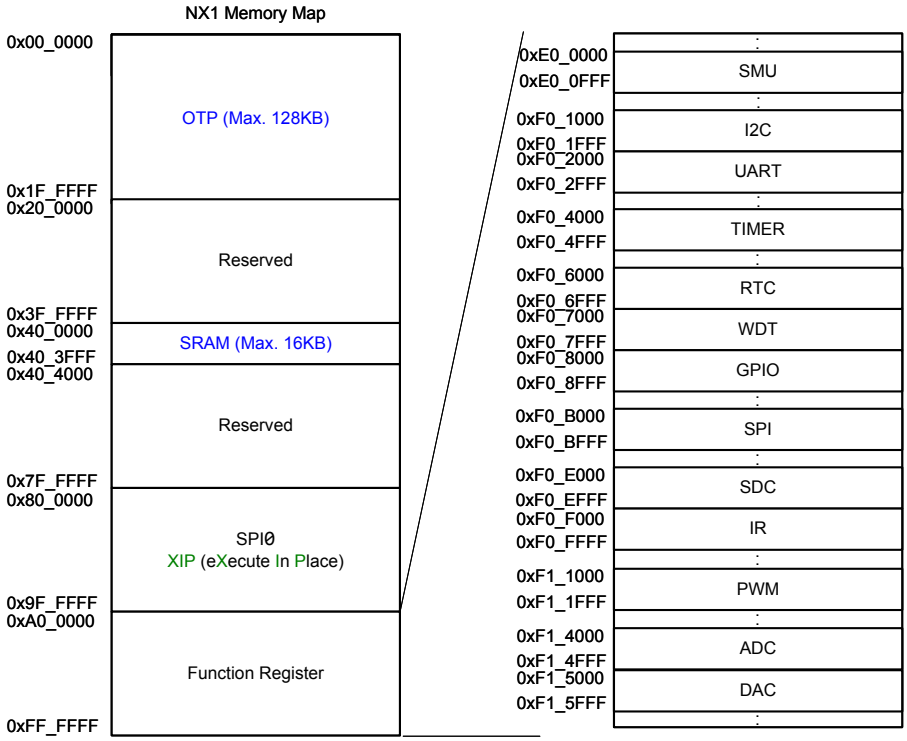


Table 2 Memory Map

2.3 Clock Generator

The clock generator consists of 3 clock sources:

- Built-in high clock (I_HRC): Output frequency can be 32MHz, 24MHz, 16MHz, or 12MHz by option.
- Built-in low clock (I_LRC): Output frequency is 32,768Hz.
- External crystal with built-in QPLL: Output frequency of QPLL is the double of crystal oscillator if enabled. When crystal is 32,768Hz, the system directly connects it to MUX without enabling QPLL.

The sources for HI_CLK and LO_CLK are decided by options. When users select Crystal as the clock source, the hardware will use built-in clock (either I_HRC or I_LRC) as clock source before crystal frequency turns stable. When the crystal output stabilized, the hardware will automatically switch to crystal frequency as the clock source and the flag, P_SMU_CLK_Ctrl[1] (Crystal_CLK_OK), will be set to 1 accordingly.

The SYS_CLK's source is decided by P_SMU_CLK_Ctrl[4] (SLOW). When SLOW is set to 1, the SYS_CLK's source is from LO_CLK, otherwise from HI_CLK. The CPUCLK's source is SYS_CLK divided by 2^n (n = 0 ~ 7), except that the system sets it to SYS_CLK in SLOW mode.

About the clock source for peripherals, please refer to the individual sections for further information.

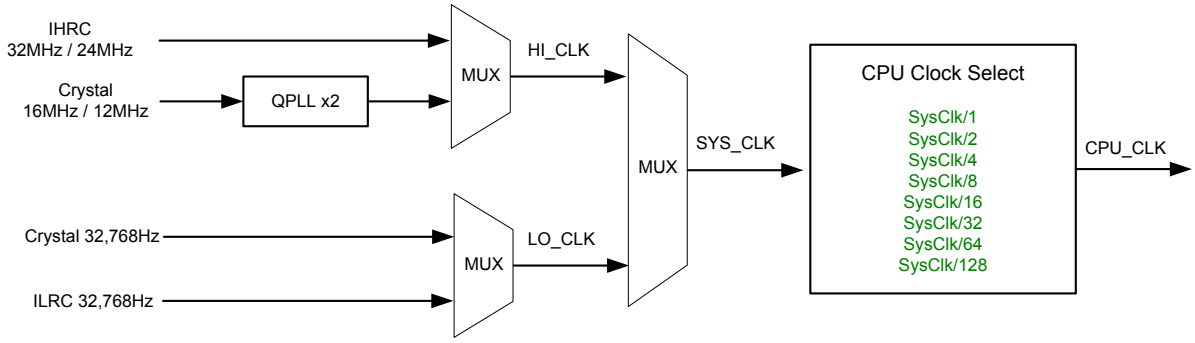


Figure 3 Clock Generator Block Diagram

2.4 System Reset

The system reset consists of 3 sources: Power-On Reset and Low Voltage Reset, External Reset pin (RSTb) and Watch-Dog Timer Reset.

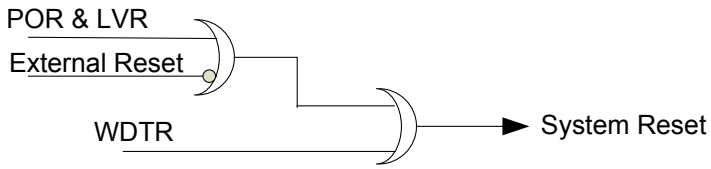


Figure 4 System Reset Block Diagram

2.4.1 POR & LVR (Power-On Reset & Low Voltage Reset)

Power on Reset (POR): When the NX1 is first powered up, the POR signal will reset the chip for a period then release it.

Low Voltage Reset (LVR): When the NX1 is running and supply power is lower than a specified voltage, the LVR signal is generated to reset the whole chip.

2.4.2 External Reset

The RSTb is an external reset pin. When it is in low state, the whole chip enters reset state. When the RSTb changes from low to high, the NX1 will release the reset state after two LO_CLK period.

2.4.3 WDTR (Watch-Dog Timer Reset)

There is an on-chip free-running oscillator and counter in the NX1 series, which is used by WDT. The NX1 provides two kinds of time settings for the WDT: 188ms or 750ms. When CPU executes STANDBY instruction, it will be cleared to zero until CPU released from standby state. Detailed information please refer to 2.17Watchdog Timer (WDT).

2.5 Operating Mode

The NX1 series provides four kinds of operating modes to tailor for various kinds of applications while saving power consumption. These operating modes are normal mode, slow mode, standby mode and halt mode.

Normal mode is designated for high-speed, high-performance operation, while slow mode is designated for low-speed to save power consumption. At standby mode, the NX1 series will stop almost all operations, except peripheral blocks with clock source from LO_CLK, to wake-up periodically. At halt mode, the NX1 series will stop all operations, waiting for external events to wake it up.

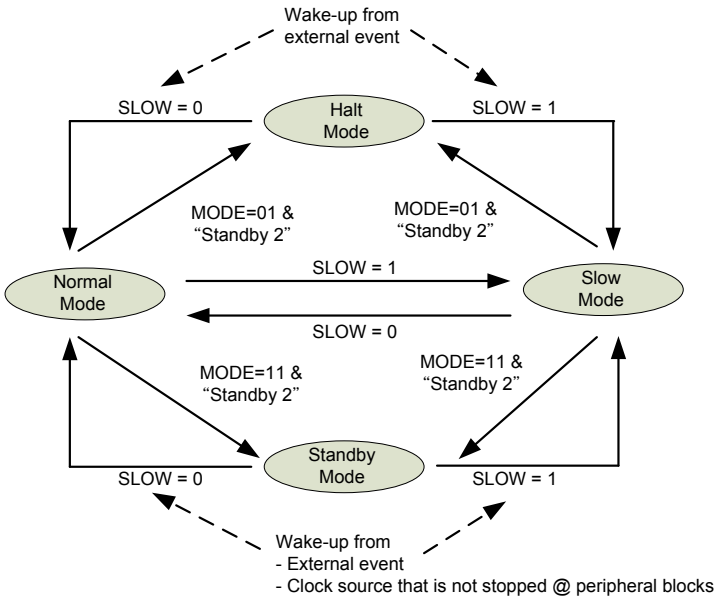


Figure 5 Four Operating Modes

2.5.1 Normal Mode

After any Reset Event occurred and chip released from reset state, the NX1 will start executing program under normal mode. At normal mode, SYS_CLK's clock source comes from HI_CLK. It provides the highest performance, which in turn leads to the largest power consumption among four operating modes.

2.5.2 Slow Mode

The NX1 will enter slow mode by clearing P_SMU_CLK_Ctrl[4] (SLOW) to 0. At Slow mode, the SYS_CLK's clock source selects from LO_CLK. It saves power consumption while keeping CPU running. However, HI_CLK is not disabled automatically under slow mode. Most of the NX1 peripheral blocks are still running by clock source from HI_CLK. Therefore, users can clear P_SMU_CLK_Ctrl[0] (HICLK_EN) to 0 in slow mode to further reduce power consumption. Beware that it is forbidden to enter slow mode and stop HI_CLK at the same time. It is required to enter slow mode firstly, then disable HI_CLK. Otherwise, the program may hang up.

2.5.3 Standby Mode

The NX1 will enter standby mode by setting P_SMU_PWR_Ctrl[1:0] (MODE) to 3 and executing "STANDBY 2" instruction. Some peripheral blocks are still active with clock source from LO_CLK. It can wake-up the NX1 when certain peripheral blocks generate interrupt events.

2.5.4 Halt Mode

The NX1 will enter halt mode by setting P_SMU_PWR_Ctrl[1:0] (MODE) to 1 and executing "STANDBY 2" instruction. At halt mode, HI_CLK and LO_CLK are automatically stopped. All peripheral blocks are disabled; instruction execution is stopped and only GPIO can wake-up the NX1. Therefore, halt mode is the most power-saving mode.

2.6 Interrupt

Interrupt signals are directly connected to the N705-S processor. The assignment of interrupt numbers is shown in Table 3. The interrupt priority is controlled by the processor. Each interrupt is assigned with 2 bits to represent 4 possible priority levels ranging from 0 (highest) to 3 (lowest). The hardware compares the priority level first: the smaller the priority level, the higher the priority. With the same priority, the lower the interrupt number, the higher the priority.

Interrupt Number	Interrupt Vector	System and Peripheral Interrupt	Interrupt Number	Interrupt Vector	System and Peripheral Interrupt
0	0	Reset/NMI	17	68	SPI0_INT
1	4	TLB fill	18	72	GPIOB_INT
2	8	PTE not present	19	76	RTC_INT
3	12	TLB misc.	20	80	PWMTMR0_INT
4	16	TLB VLPT miss	21	84	PWMTMR1_INT
5	20	Machine error	22	88	CAP_INT
6	24	Debug related	23	92	Reserved
7	28	General exception	24	96	Reserved
8	32	Syscall	25	100	GPIOC_INT
9	36	DAC0_INT	26	104	I2C_INT
10	40	DAC1_INT	27	108	UART_INT
11	44	ADC_INT	28	112	SPI1_INT
12	48	Reserved	29	116	SDC_INT
13	52	TMR0_INT	30	120	SW_INT
14	56	TMR1_INT	31	124	SWA_INT
15	60	TMR2_INT	32	128	Reserved
16	64	GPIOA_INT			

Table 3 Interrupt Vector

2.7 System Management Register

2.7.1 Overview and Features

The following functions are included:

- Clock frequency and power consumption control
- Low voltage detection setting
- Operating mode setting
- Software interrupt
- Software reset control and reset flag register
- Peripheral block function enable control and function reset control

2.7.2 Register Map

Register	Offset	R/W	Description	Initial Value
SMU Base Address: SMU_BA = 0xE0_0000				
P_SMU_CLK_Ctrl	SMU_BA+0x00	R/W	Clock Control Register	0x0000_00E3
P_SMU_PWR_Ctrl	SMU_BA+0x04	R/W	Power Control Register	0x0000_0001
P_SMU_SW_INT	SMU_BA+0x10	R/W	Software Interrupt Control Register	0x0000_0000
P_SMU_RST_Flag	SMU_BA+0x20	R/W1C	Reset Flag Register	0x0000_0008
P_SMU_RST_SW_Ctrl	SMU_BA+0x24	W	Reset Control Register	0x0000_0000
P_SMU_FUNC_Ctrl	SMU_BA+0x30	R/W	Function Control Register	0x0000_0000
P_SMU_FUNC_RST	SMU_BA+0x34	W	Function Reset Register	0x0000_0000

2.7.3 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to Clear, **R/W**: Read and Write

- **P_SMU_CLK_Ctrl (Clock Control Register)**

Register	Offset		Description	Initial Value
P_SMU_CLK_Ctrl	SMU_BA+0x00		Clock Control Register	0x0000_00E3
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0
Bit[7:5]	CPUCLKDIV	R/W	111: SYS_CLK/1 110: SYS_CLK/2 101: SYS_CLK/4 100: SYS_CLK/8 011: SYS_CLK/16 010: SYS_CLK/32 001: SYS_CLK/64 000: SYS_CLK/128	0x7
Bit[4]	SLOW	R/W	1: SYS_CLK from LOCLK 0: SYS_CLK from HICLK	0x0
Bit[3:2]	Reserved			
Bit[1]	Crystal_CLK_OK	R	1: Crystal CLK OK 0: Crystal CLK unstable	0x1
Bit[0]	HICLK_EN	R/W	1: HICLK enabled 0: HICLK disable	0x1

- **P_SMU_PWR_Ctrl (Power Control Register)**

Register	Offset		Description	Initial Value
P_SMU_PWR_Ctrl	SMU_BA+0x04		Power Control Register	0x0000_0001
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[14:12]	LVD_SEL	R/W	111: (reserved) 110: (reserved) 101: 3.6V 100: 3.4V 011: 3.2V 010: 2.6V 001: 2.4V 000: 2.2V	0x0
Bit[11:10]	Reserved			0x0
Bit[9]	LVD_FLAG	R	1: detect low voltage 0: no detect low voltage	0x0
Bit[8]	LVD_EN	R/W	1: LVD enable 0: LVD disable	0x0
Bit[7]	LDOSPI0_MODE	R/W	1 : Light load 0 : Heavy load	0x0
Bit[6]	SDC_PWDN	R/W	1: SDC's input pin mask to 1 0: SDC's input pin unmask	0x0
Bit[5]	SPI1_PWDN	R/W	1: SPI1's input pin mask to 1 0: SPI1's input pin unmask	0x0
Bit[4]	LDOSPI0_EN	R/W	1: LDOSPI0 IP enabled 0: LDOSPI0 IP disabled	0x0
Bit[3]	CSC_BIAS_EN	R/W	1: CSC bias enabled 0: CSC bias disabled	0x0
Bit[2]			Reserved	0x0
Bit[1:0]	MODE	R/W	11: Standby mode 01: Halt mode x0: Only CPU stops	0x1

- **P_SMU_SW_INT (Software Interrupt Control Register)**

Register	Offset		Description	Initial Value
P_SMU_SW_INT	SMU_BA+0x10		Software Interrupt Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Protection Key	W		0x0
Bit[15:1]	Reserved			0x0
Bit[0]	SWA_INT	W/R	1: Software interrupt request, when Protection Key = 5AC3 0: Do nothing	0x0

- **P_SMU_RST_Flag (Reset Flag Register)**

Register	Offset		Description	Initial Value
P_SMU_RST_Flag	SMU_BA+0x20		Reset Flag Register	0x0000_0008
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:6]	Reserved			0x0
Bit[5]	RSTF_ROMFAIL	R/W1C	1: Flag. reset by dl_opn rom fail 0: No reset	0x0
Bit[4]	RSTF_WDT	R/W1C	1: Flag. reset CHIP by WDT 0: No reset	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[3]	RSTF_POR	R/W1C	1: Flag. reset CHIP by POR / LVR 0: No reset	0x1
Bit[2]	RSTF_ERST	R/W1C	1: Flag to indicate CHIP reset by hardware. (External Reset) 0: No reset	0x0
Bit[1]	RSTF_CHIP	R/W1C	1: Flag to indicate CHIP reset by software. (write register CHIP_RST) 0: No reset	0x0
Bit[0]	RSTF_CPU	R/W1C	1: Flag to indicate CPU reset by software. (write register CPU_RST) 0: No reset	0x0

- **P_SMU_RST_SW_Ctrl (Reset Control Register)**

Register	Offset		Description	Initial Value
P_SMU_RST_SW_Ctrl	SMU_BA+0x24		Reset Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Protection Key	W		0x0
Bit[15:2]	Reserved			0x0
Bit[1]	CHIP_RST	W	When Protection Key=5AC3 & CHIP_RST=1, Whole CHIP reset	0x0
Bit[0]	CPU_RST	W	When Protection Key=5AC3 & CPU_RST=1, CPU reset (AMBA reset)	0x0

- **P_SMU_FUNC_Ctrl (Function Control Register)**

Register	Offset		Description	Initial Value
P_SMU_FUNC_Ctrl	SMU_BA+0x30		Function Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:11]	Reserved			0x0
Bit[10]	IRCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[9]	URCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[8]	SDCCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[7]	DACCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[6]	ADCCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[5]	IICCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[4]	RTCCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[3]	SPICLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[2]	PWMCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[1]	TMRCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0
Bit[0]	GPIOCLK_EN	R/W	1: IP block's clock enable 0: IP block's clock disabled (default)	0x0

- **P_SMU_FUNC_RST (Function Reset Register)**

Register	Offset		Description	Initial Value
P_SMU_FUNC_RST	SMU_BA+0x34		Function Reset Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:11]	Reserved			0x0
Bit[10]	IRBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[9]	URBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[8]	SDCBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[7]	DACBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[6]	ADCBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[5]	IICBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[4]	Reserved			0x0
Bit[3]	SPIBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[2]	PWMBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[1]	TMRBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0
Bit[0]	GPIOBK_RST	W	1: Gen Reset pulse 0: No Reset	0x0

2.8 IO Ports

2.8.1 Overview and Features

- Max. 40 Individually programmable input/output pins, default as input at reset
- Support multiple functions

2.8.2 Block Diagram

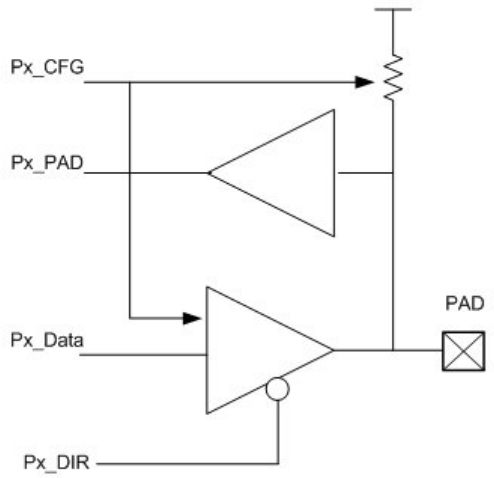


Figure 6 I/O Pad Structure

2.8.3 Function Description

Up to 40 pins are available. These are shared multiple function pins under control of the alternate multiple function registers. A total of 40 pins are arranged in 3 ports named with Port A (PA), Port B (PB) and Port C (PC). The PA port has 16 pins, the PB port has 16 pins and the PC port has 8 pins.

Each pin can be configured as input or output, weak / strong pull-high resistor and can generate interrupt signal to CPU upon wakeup. PA15 also has external interrupt function. External interrupt and Port A wakeup interrupt share the same service routine.

2.8.4 Register Map

Register	Offset	R/W	Description	Initial Value
GPIO Base Address: GPIO_BA = 0xF0_8000				
P_PA_DIR	GPIO_BA+0x00	R/W	Port A input output control	0x0000_FFFF
P_PA_Data	GPIO_BA+0x04	R/W	Port A data output register	0x0000_FFFF
P_PA_PAD	GPIO_BA+0x08	R/W	Port A pad input register	0x0000_uuuu
P_PA_Wakeup_Mask	GPIO_BA+0x0C	R/W	Port A Config register0	0xFFFF_0000
P_PA_CFG	GPIO_BA+0x10	R/W	Port A Config register1	0xFFFF_FFFF
P_PA_MFP	GPIO_BA+0x14	R/W	Port A multi-function register	0x0000_0000
P_PB_DIR	GPIO_BA+0x20	R/W	Port B input output control	0x0000_FFFF
P_PB_Data	GPIO_BA+0x24	R/W	Port B data output register	0x0000_FFFF
P_PB_PAD	GPIO_BA+0x28	R/W	Port B pad input register	0x0000_uuuu
P_PB_Wakeup_Mask	GPIO_BA+0x2C	R/W	Port B Config register0	0xFFFF_0000
P_PB_CFG	GPIO_BA+0x30	R/W	Port B Config register1	0xFFFF_FFFF
P_PB_MFP	GPIO_BA+0x34	R/W	Port B multi-function register	0x0000_0000
P_PC_DIR	GPIO_BA+0x40	R/W	Port C input output control	0x0000_00FF
P_PC_Data	GPIO_BA+0x44	R/W	Port C data output register	0x0000_00FF
P_PC_PAD	GPIO_BA+0x48	R/W	Port C pad input register	0x0000_00uu
P_PC_Wakeup_Mask	GPIO_BA+0x4C	R/W	Port C Config register0	0x00FF_0000
P_PC_CFG	GPIO_BA+0x50	R/W	Port C Config register1	0x00FF_00FF
P_PC_MFP	GPIO_BA+0x54	R/W	Port C multi-function register	0x0000_0000
P_EXT_INT_Trig	GPIO_BA+0x60	R/W	EXT INT TRIG Select	0x0000_0000
P_EXT_INT_Ctrl	GPIO_BA+0x64	R/W	EXT INT Interrupt Enable	0x0000_0000
P_EXT_INT_Flag	GPIO_BA+0x68	R/W1C	EXT INT Interrupt Flag	0x0000_0000
P_PULL_HIGH_CFG	GPIO_BA+0x70	R/W	Port pull-high Config register	0x0000_0000

2.8.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_PA_DIR (Port A Input Output Control)**

Register	Offset		Description	Initial Value
P_PA_DIR	GPIO_BA+0x00		Port A input output control	0x0000_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[15:0]	PA_DIR	R/W	I/O Direction 1: input 0: output	0xFFFF

- **P_PA_Data (Port A Data Output Register)**

Register	Offset		Description	Initial Value
P_PA_Data	GPIO_BA+0x04		Port A data output register	0x0000_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0
Bit[15:0]	PA_DATA	R/W	Data output 1: data output 1 0: data output 0	0xFFFF

- **P_PA_PAD (Port A Pad Input Register)**

Register	Offset		Description	Initial Value
P_PA_PAD	GPIO_BA+0x08		Port A pad input register	0x0000_uuuu
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0
Bit[15:0]	PA_PAD	R/W	Read pad data	0xuuuu

- **P_PA_Wakeup_Mask (Port A Config Register0)**

Register	Offset		Description	Initial Value
P_PA_Wakeup_Mask	GPIO_BA+0x0C		Port A Config register0	0xFFFF_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	PA_CCS1	R/W	PA15 ~ PA8 Sink / CSC current select 1: Normal Sink / CSC 66% 0: Large Sink / CSC 100%	0xFF
Bit[23:16]	PA_CCS0	R	PA7 ~ PA0 Sink current select 1: Normal Sink 0: Large Sink	0xFF
Bit[15:0]	PA_MASK	R/W	PA15 ~ PA0 Wakeup Function Mask 1: Have wakeup function 0: No Wake function	0x0

- **P_PA_CFG (Port A Config Register1)**

Register	Offset		Description	Initial Value
P_PA_CFG	GPIO_BA+0x10		Port A Config register1	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	PA_CSCENB1	R/W	PA15~PA8 Sink or CSC mode select 1: Sink mode 0: CSC mode	0xFF
Bit[23:16]	PA_CSCENB0	R	PA7 ~ PA0 select sink mode	0xFF
Bit[15:0]	PA_PUENB	R/W	PA15 ~ PA0 Pull-up Resistor select 1: Pull-up Resistor disabled 0: Pull-up Resistor enabled	0xFFFF

- **P_PA_MFP (Port A Multi-Function Register)**

Register	Offset		Description	Initial Value
P_PA_MFP	GPIO_BA+0x14		Port A multi-function register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31]	Reserved			0x0
Bit[30]	PA14_MFP 1	R/W	1 : External INT source 1 0 : GPIO PA14	
Bit[29:24]	Reserved			
Bit[23]	PA7_MFP1	R/W	1: Channel 7 of ADC 0: GPIO PA7	0x0
Bit[22]	PA6_MFP1	R/W	1: Channel 6 of ADC 0: GPIO PA6	0x0
Bit[21]	PA5_MFP1	R/W	1: Channel 5 of ADC 0: GPIO PA5	0x0
Bit[20]	PA4_MFP1	R/W	1: Channel 4 of ADC 0: GPIO PA4	0x0
Bit[19:16]	Reserved			0x0
Bit[15]	PA15_MFP	R/W	1: External INT source 0 0: GPIO PA15	0x0
Bit[14]	PA14_MFP	R/W	1: Timer1 & timer2 external input 0: GPIO PA14	0x0
Bit[13]	Reserved			0x0
Bit[12]	PA12_MFP	R/W	1: SDA & SCL for I^2C 0: GPIO PA12 ~ PA13	0x0
Bit[11]	PA11_MFP	R/W	1: PWM3 output 0: GPIO PA11	0x0
Bit[10]	PA10_MFP	R/W	1: PWM2 output 0: GPIO PA10	0x0
Bit[9]	PA9_MFP	R/W	1: PWM1 output 0: GPIO PA9	0x0
Bit[8]	PA8_MFP	R/W	1: PWM0 output 0: GPIO PA8	0x0
Bit[7]	PA7_MFP	R/W	1: RXD input 0: GPIO PA7	0x0
Bit[6]	PA6_MFP	R/W	1: TXD output 0: GPIO PA6	0x0
Bit[5]	PA5_MFP	R/W	1: IR0 output pin 0: GPIO PA5	0x0
Bit[4]	PA4_MFP	R/W	1: Timer0 external input 0: GPIO PA4	0x0
Bit[3]	PA3_MFP	R/W	1: Channel 3 of ADC 0: GPIO PA3	0x0
Bit[2]	PA2_MFP	R/W	1: Channel 2 of ADC 0: GPIO PA2	0x0
Bit[1]	PA1_MFP	R/W	1: Channel 1 of ADC 0: GPIO PA1	0x0
Bit[0]	PA0_MFP	R/W	1: Channel 0 of ADC 0: GPIO PA0	0x0

- **P_PB_DIR (Port B Input Output Control)**

Register	Offset		Description	Initial Value
P_PB_DIR	GPIO_BA+0x20		Port B input output control	0x0000_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0

Bit[15:0]	PB_DIR	R/W	I/O Direction 1: Input 0: Output	0xFFFF

- **P_PB_Data (Port B Data Output Register)**

Register	Offset		Description	Initial Value
P_PB_Data	GPIO_BA+0x24		Port B data output register	0x0000_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0
Bit[15:0]	PB_DATA	R/W	Data output 1: Data output 1 0: Data output 0	0xFFFF

- **P_PB_PAD (Port B Pad Input Register)**

Register	Offset		Description	Initial Value
P_PB_PAD	GPIO_BA+0x28		Port B pad input register	0x0000_uuuu
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0
Bit[15:0]	PB_PAD	R/W	Read pad data	0xuuuu

- **P_PB_Wakeup_Mask (Port B Config Register0)**

Register	Offset		Description	Initial Value
P_PB_Wakeup_Mask	GPIO_BA+0x2C		Port B Config register0	0xFFFF_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	PB_CCS1	R/W	PB15 ~ PB8 Sink current select 1: Normal Sink 0: Large Sink	0xFF
Bit[23:16]	PB_CCS0	R	PB7 ~ PB0 Sink current select 1: Normal Sink 0: Large Sink	0xFF
Bit[15:0]	PB_MASK	R/W	PB15 ~ PB0 Wakeup Function Mask 1: Have wakeup function 0: No Wake function	0x0

- **P_PB_CFG (Port B Config Register1)**

Register	Offset		Description	Initial Value
P_PB_CFG	GPIO_BA+0x30		Port B Config register1	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	PB_CSCENB1	R	PB15 ~ PB8 select sink mode	0xFF
Bit[23:16]	PB_CSCENB0	R	PB7 ~ PB0 select sink mode	0xFF
Bit[15:0]	PB_PUENB	R/W	PB15 ~ PB0 Pull-up Resistor select 1: Pull-up Resistor disabled 0: Pull-up Resistor enabled	0xFFFF

- **P_PB_MFP (Port B Multi-Function Register)**

Register	Offset		Description	Initial Value
P_PB_MFP	GPIO_BA+0x34		Port B multi-function register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0

Bit[15]	PB15_MFP	R/W	1: PWM7 output 0: GPIO PB15	0x0
Bit[14]	PB14_MFP	R/W	1: PWM6 output 0: GPIO PB14	0x0
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[13]	PB13_MFP	R/W	1: PWM5 output 0: GPIO PB13	0x0
Bit[12]	PB12_MFP	R/W	1: PWM4 output 0: GPIO PB12	0x0
Bit[11]	Reserved			0x0
Bit[10]	Reserved			0x0
Bit[9]	PB9_11_MFP	R/W	1: SPI1_SCLK, SPI1_MISO, SPI1_MOSI 0: GPIO PB9 ~ PB11	0x0
Bit[8]	PB8_MFP	R	Reserved	0x0
Bit[7]	PB7_MFP	R	1: OSC's Xout pin 0: GPIO PB7	0x0
Bit[6]	PB6_7_MFP	R	1: OSC's Xin pin 0: GPIO PB6	0x0
Bit[5]	PB5_MFP	R	Reserved	0x0
Bit[4]	PB4_5_MFP	R/W	1: SPI0_IO2, SPI0_IO3 0: GPIO PB4, PB5	0x0
Bit[3]	Reserved			0x0
Bit[2]	Reserved			0x0
Bit[1]	PB1_3_MFP	R/W	1: SPI0_SCLK / IO0 / IO1 0: GPIO PB1 ~ PB3	0x0
Bit[0]	PB0_MFP	R/W	1: # SPI0_CS 0: GPIO PB0	0x0

- **P_PC_DIR (Port C Input Output Control)**

Register	**Offset**		**Description**	**Initial Value**
P_PC_DIR	GPIO_BA+0x40		Port C input output control	0x0000_00FF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0
Bit[7:0]	PC_DIR	R/W	I/O Direction 1: Input 0: Output	0xFF

- **P_PC_Data (Port C Data Output Register)**

Register	**Offset**		**Description**	**Initial Value**
P_PC_Data	GPIO_BA+0x44		Port C data output register	0x0000_00FF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0
Bit[7:0]	PC_DATA	R/W	Data output 1: Data output 1 0: Data output 0	0xFF

- **P_PC_PAD (Port C Pad Input Register)**

Register	**Offset**		**Description**	**Initial Value**
P_PC_PAD	GPIO_BA+0x48		Port C pad input register	0x0000_00uu
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0
Bit[7:0]	PC_PAD	R/W	Read pad data	0xuu

- **P_PC_Wakeup_Mask (Port C Config Register0)**

Register	Offset		Description	Initial Value
P_PC_Wakeup_Mask	GPIO_BA+0x4C		Port C Config register0	0x00FF_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	Reserved			0x0
Bit[23:16]	PC_CCS0	R	PC7 ~ PC0 Sink current select 1: Normal Sink 0: Large Sink	0xFF
Bit[15:8]	Reserved			0x0
Bit[7:0]	PC_MASK	R/W	PC7 ~ PA0 Wakeup Function Mask 1: Have wakeup function 0: No Wake function	0x0

- **P_PC_CFG (Port C Config Register1)**

Register	Offset		Description	Initial Value
P_PC_CFG	GPIO_BA+0x50		Port C Config register1	0x00FF_00FF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	Reserved			0x0
Bit[23:16]	PC_CSCENB0	R	PC7 ~ PC0 select sink mode	0xFF
Bit[15:8]	Reserved			0x0
Bit[7:0]	PC_PUENB	R/W	PC7 ~ PC0 Pull-up Resistor select 1: Pull-up Resistor disabled 0: Pull-up Resistor enabled	0xFF

- **P_PC_MFP (Port C Multi-Function Register)**

Register	Offset		Description	Initial Value
P_PC_MFP	GPIO_BA+0x54		Port C multi-function register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0
Bit[7]	PC7_MFP	R/W	1: SDC_WP 0: GPIO PC7	0x0
Bit[6]	PC6_MFP	R/W	1: SDC_CD 0: GPIO PC6	0x0
Bit[5:1]	Reserved			0x0
Bit[0]	PC0_5_MFP	R/W	1: SDC_CLK / CMD / IO0 ~ IO3 0: GPIO PC0 ~ PC5	0x0

- **P_EXT_INT_Trig (EXT INT TRIG Select)**

Register	Offset		Description	Initial Value
P_EXT_INT_Trig	GPIO_BA+0x60		EXT INT TRIG Select	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:4]	Reserved			0x0
Bit[3:2]	TRIG_EXT1	R/W	1x: rising edge & falling edge 01: Falling edge 00: Rising edge	0x0
Bit[1:0]	TRIG_EXT0	R/W	1x: rising edge & falling edge 01: Falling edge 00: Rising edge	0x0

- **P_EXT_INT_Ctrl (EXT INT Interrupt Enable)**

Register	Offset		Description	Initial Value
P_EXT_INT_Ctrl	GPIO_BA+0x64		EXT INT Interrupt Enable	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:2]	Reserved			0x0
Bit[1]	EXT1_IEN	R/W	1: EXT1 interrupt enabled 0: EXT1 interrupt disabled (default)	0x0
Bit[1:0]	EXT0_IEN	R/W	1: EXT0 interrupt enabled 0: EXT0 interrupt disabled (default)	0x0

- **P_EXT_INT_Flag (EXT INT Interrupt Flag)**

Register	Offset		Description	Initial Value
P_EXT_INT_Flag	GPIO_BA+0x68		EXT INT Interrupt Flag	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:2]	Reserved			0x0
Bit[1]	EXT1_INTF	R/W1C	1: Have EXT1 interrupt flag 0: No EXT1 interrupt flag	0x0
Bit[1:0]	EXT0_INTF	R/W1C	1: Have EXT0 interrupt flag 0: No EXT0 interrupt flag	0x0

- **P_PULL_HIGH_CFG (Port pull-high Config register)**

Register	Offset		Description	Initial Value
P_PULL_HIGH_CFG	GPIO_BA+0x70		Port pull-high Config register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:5]	Reserved			0x0
Bit[4]	PC_L_WSB	R/W	1: PC0 ~ PC7 use Strong pull-up (100K) 0: PC0 ~ PC7 use Weak pull-up (1M)	0x0
Bit[3]	PB_H_WSB	R/W	1: PB8 ~ PB15 use Strong pull-up (100K) 0: PB8 ~ PB15 use Weak pull-up (1M)	0x0
Bit[2]	PB_L_WSB	R/W	1: PB0 ~ PB7 use Strong pull-up (100K) 0: PB0 ~ PB7 use Weak pull-up (1M)	0x0
Bit[1]	PA_H_WSB	R/W	1: PA8 ~ PA15 use Strong pull-up (100K) 0: PA8 ~ PA15 use Weak pull-up (1M)	0x0
Bit[0]	PA_L_WSB	R/W	1: PA0 ~ PA7 use Strong pull-up (100K) 0: PA0 ~ PA7 use Weak pull-up (1M)	0x0

2.9 Timer Controller

2.9.1 Overview and Features

The NX1 has three 16-bit timers: TIMER0 / TIMER1 / TIMER2, which can be used as a trigger source for DAC / PWM or as a function of time delay, clock generation, etc.

- Programmable source of timer clock
- 16-bit counter for each timer

2.9.2 Block Diagram

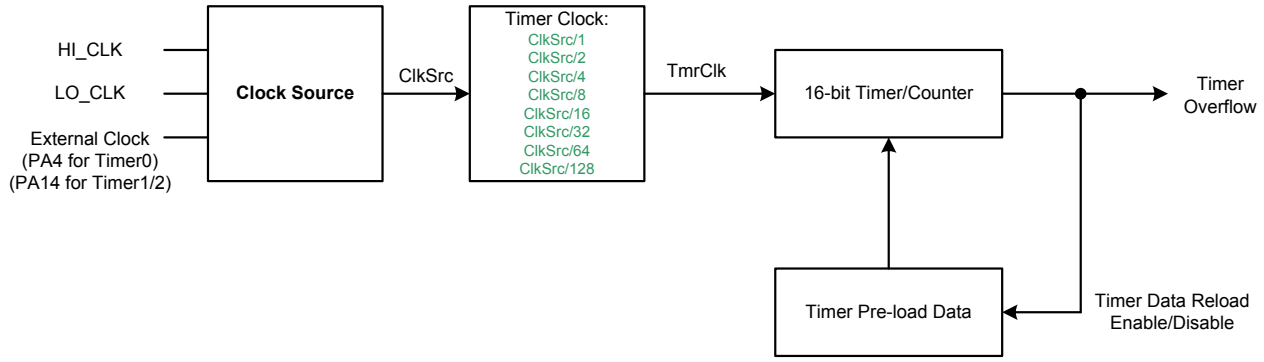


Figure 7 Timer Block Diagram

2.9.3 Function Description

The NX1 provides up to three multi-function timers for users to implement a variety of applications. Beside clock sources selected from High and Low clocks, these timers also accept external clock sources from PA port as well. When external clocks are used as the source for the timer, users can further decide if to synchronize with SYS_CLK.

2.9.4 Register Map

Register	Offset	R/W	Description	Initial Value
Timer Base Address: TMR_BA = 0xF0_4000				
P_TMR0_Ctrl	TMR_BA+0x00	R/W	Timer 0 Control Register	0x0000_0302
P_TMR0_Data	TMR_BA+0x04	R/W	Timer 0 Data Register	0xFFFF_FFFF
P_TMR0_INT_Ctrl	TMR_BA+0x08	R/W	Timer 0 Interrupt Control Register	0x0000_0000
P_TMR0_Flag	TMR_BA+0x0C	R/W1C	Timer 0 Interrupt FLAG	0x0000_0000
P_TMR1_Ctrl	TMR_BA+0x40	R/W	Timer 1 Control Register	0x0000_0302
P_TMR1_Data	TMR_BA+0x44	R/W	Timer 1 Data Register	0xFFFF_FFFF
P_TMR1_INT_Ctrl	TMR_BA+0x48	R/W	Timer 1 Interrupt Control Register	0x0000_0000
P_TMR1_Flag	TMR_BA+0x4C	R/W1C	Timer 1 Interrupt FLAG	0x0000_0000
P_TMR2_Ctrl	TMR_BA+0x80	R/W	Timer 2 Control Register	0x0000_0302
P_TMR2_Data	TMR_BA+0x84	R/W	Timer 2 Data Register	0xFFFF_FFFF
P_TMR2_INT_Ctrl	TMR_BA+0x88	R/W	Timer 2 Interrupt Control Register	0x0000_0000
P_TMR2_Flag	TMR_BA+0x8C	R/W1C	Timer 2 Interrupt FLAG	0x0000_0000

2.9.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_TMR0_Ctrl (Timer 0 Control Register)**

Register	Offset	Description	Initial Value
P_TMR0_Ctrl	TMR_BA+0x00	Timer 0 Control Register	0x0000_0302

Bit	Name	R/W	Descriptions	Initial Value
Bit[31:10]	Reserved			0x0000_0000
Bit[9:8]	CKS	R/W	11: TMCK's source from LOCLK (default) 10: TMCK's source from HICLK 01: TMCK's source from EXCLK (no sync w/ SYS_CLK) 00: TMCK's source from EXCLK (sync w/ SYS_CLK)	0x3
Bit[7]	Reserved			0x0
Bit[6:4]	TMRSEL	R/W	111: TMRCK=TMCK0/1 110: TMRCK=TMCK0/2 101: TMRCK=TMCK0/4 100: TMRCK=TMCK0/8 011: TMRCK=TMCK0/16 010: TMRCK=TMCK0/32 001: TMRCK=TMCK0/64 000: TMRCK=TMCK0/128	0x0
Bit[3:2]	Reserved			
Bit[1]	TMR_LD	R/W	1: TMR reload in overflow (TMR) 0: TMR no reload in overflow (CNT)	0x1
Bit[0]	TMR_EN	R/W	1: TMR enable 0: TMR disabled (default)	0x0

- **P_TMR0_Data (Timer 0 Data Register)**

Register	Offset		Description	Initial Value
P_TMR0_Data	TMR_BA+0x04		Timer 0 Data Register	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	TMR_DAT	R	TMR's counter data	0xFFFF
Bit[15:0]	TMR_LDAT	R/W	TMR's preload Data	0xFFFF

- **P_TMR0_INT_Ctrl (Timer 0 Interrupt Control Register)**

Register	Offset		Description	Initial Value
P_TMR0_INT_Ctrl	TMR_BA+0x08		Timer 0 Interrupt Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_IEN	R/W	1: TMR's interrupt enabled 0: TMR's interrupt disabled (default)	0x0

- **P_TMR0_Flag (Timer 0 Interrupt FLAG)**

Register	Offset		Description	Initial Value
P_TMR0_Flag	TMR_BA+0x0C		Timer 0 Interrupt FLAG	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_INTF	R/W1C	1: TMR's interrupt flag 0: TMR's interrupt no flag (default)	0x0

- **P_TMR1_Ctrl (Timer 1 Control Register)**

Register	Offset		Description	Initial Value
P_TMR1_Ctrl	TMR_BA+0x40		Timer 1 Control Register	0x0000_0302
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:10]	Reserved			0x0000_0302

Bit	Name	R/W	Descriptions	Initial Value
Bit[9:8]	CKS	R/W	11: TMCK's source from LOCLK (default) 10: TMCK's source from HICLK 01: TMCK's source from EXCLK (no sync w/ SYS_CLK) 00: TMCK's source from EXCLK (sync w/ SYS_CLK)	0x3
Bit[7]	Reserved			0x0
Bit[6:4]	TMRSEL	R/W	111: TMRCK=TMCK0/1 110: TMRCK=TMCK0/2 101: TMRCK=TMCK0/4 100: TMRCK=TMCK0/8 011: TMRCK=TMCK0/16 010: TMRCK=TMCK0/32 001: TMRCK=TMCK0/64 000: TMRCK=TMCK0/128	0x0
Bit[3:2]	Reserved			
Bit[1]	TMR_LD	R/W	1: TMR reload in overflow (TMR) 0: TMR no reload in overflow (CNT)	0x1
Bit[0]	TMR_EN	R/W	1: TMR enable 0: TMR disabled (default)	0x0

- **P_TMR1_Data (Timer 1 Data Register)**

Register	Offset		Description	Initial Value
P_TMR1_Data	TMR_BA+0x44		Timer 1 Data Register	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	TMR_DAT	R	TMR's counter data	0xFFFF
Bit[15:0]	TMR_LDAT	R/W	TMR's preload Data	0xFFFF

- **P_TMR1_INT_Ctrl (Timer 1 Interrupt Control Register)**

Register	Offset		Description	Initial Value
P_TMR1_INT_Ctrl	TMR_BA+0x48		Timer 1 Interrupt Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_IEN	R/W	1: TMR's interrupt enabled 0: TMR's interrupt disabled (default)	0x0

- **P_TMR1_Flag (Timer 1 Interrupt FLAG)**

Register	Offset		Description	Initial Value
P_TMR1_Flag	TMR_BA+0x4C		Timer 1 Interrupt FLAG	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_INTF	R/W1C	1: TMR's interrupt flag 0: TMR's interrupt no flag (default)	0x0

- **P_TMR2_Ctrl (Timer 2 Control Register)**

Register	Offset		Description	Initial Value
P_TMR2_Ctrl	TMR_BA+0x80		Timer 2 Control Register	0x0000_0302
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:10]	Reserved			0x0000_0302

Bit	Name	R/W	Descriptions	Initial Value
Bit[9:8]	CKS	R/W	11: TMCK's source from LOCLK (default) 10: TMCK's source from HICLK 01: TMCK's source from EXCLK (no sync w/ SYS_CLK) 00: TMCK's source from EXCLK (sync w/ SYS_CLK)	0x3
Bit[7]	Reserved			0x0
Bit[6:4]	TMRSEL	R/W	111: TMRCK=TMCK0/1 110: TMRCK=TMCK0/2 101: TMRCK=TMCK0/4 100: TMRCK=TMCK0/8 011: TMRCK=TMCK0/16 010: TMRCK=TMCK0/32 001: TMRCK=TMCK0/64 000: TMRCK=TMCK0/128	0x0
Bit[3:2]	Reserved			
Bit[1]	TMR_LD	R/W	1: TMR reload in overflow (TMR) 0: TMR no reload in overflow (CNT)	0x1
Bit[0]	TMR_EN	R/W	1: TMR enable 0: TMR disabled (default)	0x0

- **P_TMR2_Data (Timer 2 Data Register)**

Register	Offset		Description	Initial Value
P_TMR2_Data	TMR_BA+0x84		Timer 2 Data Register	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	TMR_DAT	R	TMR's counter data	0xFFFF
Bit[15:0]	TMR_LDAT	R/W	TMR's preload Data	0xFFFF

- **P_TMR2_INT_Ctrl (Timer 2 Interrupt Control Register)**

Register	Offset		Description	Initial Value
P_TMR2_INT_Ctrl	TMR_BA+0x88		Timer 2 Interrupt Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_IEN	R/W	1: TMR's interrupt enabled 0: TMR's interrupt disabled (default)	0x0

- **P_TMR2_Flag (Timer 2 Interrupt FLAG)**

Register	Offset		Description	Initial Value
P_TMR2_Flag	TMR_BA+0x8C		Timer 2 Interrupt FLAG	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_INTF	R/W1C	1: TMR's interrupt flag 0: TMR's interrupt no flag (default)	0x0

2.10 I²C Serial Interface Controller (Master/Slave)

2.10.1 Overview and Features

The controller is an I²C (Inter-Integrated Circuit) master/slave controller.

- Support Standard-mode (100 Kb/s), Fast-mode (400 Kb/s) and Fast-mode Plus (1 Mb/s) protocols
- Programmable Master/Slave mode
- Support 7-bit and 10-bit addressing mode
- Support general call address
- Auto clock stretching

2.10.2 Block Diagram

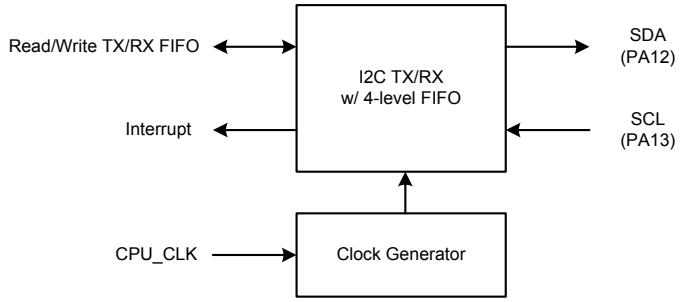


Figure 8 I²C Block Diagram

2.10.3 Function Description

The controller can act as either an I²C master device or an I²C slave device, depending on the control register settings.

I²C Master: As an I²C master, every transaction can be delineated by four phases: Start, Address, Data and Stop.

I²C Slave: As an I²C slave, the controller is addressed when the address byte of an I²C transaction matches the Address Register. An Address Hit interrupt can be generated for the software to prepare for the subsequent operations.

General Call Address: The controller at the slave mode will respond with an ACK to the general call address and set the GenCall field of the Status Register.

Auto Clock Stretch: When the software is not ready for the next byte of data or when the FIFO is full, it can automatically pause bus transactions by stretching clocks on the I²C-bus

Auto-ACK: With Auto-ACK, it automatically generates proper acknowledgements for each byte received except for the last byte, which should be responded with a NACK according to the I²C-bus protocol. By enabling the Byte Receive Interrupt, user can turned off Auto-ACK to determine each byte's acknowledgement status.

2.10.4 Register Map

Register	Offset	R/W	Description	Initial Value
I²C Base Address: IIC_BA = 0xF0_1000				
P_I2C_IDREV	I2C_BA+0x00	R	I²C ID and Revision Register	0x2021_0002
P_I2C_FIFO_Ctrl	I2C_BA+0x10	R	I²C FIFO Configuration Register	0x0000_0001
P_I2C_INT_Ctrl	I2C_BA+0x14	R/W	I²C Interrupt Enable Register	0x0000_0000
P_I2C_Status	I2C_BA+0x18	R/W1C	I²C Status Register	0xu000_0001
P_I2C_Addr	I2C_BA+0x1C	R/W	I²C Address Register	0xu000_0000
P_I2C_Data	I2C_BA+0x20	R/W	I²C Data Register	0xu000_0000
P_I2C_Mode	I2C_BA+0x24	R/W	I²C Control Register	0x0000_1E00
P_I2C_Cmd	I2C_BA+0x28	R/W	I²C Command Register	0x0000_0000
P_I2C_Ctrl	I2C_BA+0x2C	R/W	I²C Controller Setting Register	0x0525_2100

2.10.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_I2C_IDREV (I²C ID and Revision Register)**

Register	Offset		Description	Initial Value
P_I2C_IDREV	I2C_BA+0x00		I²C ID and Revision Register	0x2021_0002
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	ID	R	ID number for ATCIIC100	0x2_0210
Bit[11:4]	RevMajor	R	Major revision number	0x0
Bit[3:0]	RevMinor	R	Minor revision number	0x2

- **P_I2C_FIFO_Ctrl (I²C FIFO Configuration Register)**

Register	Offset		Description	Initial Value
P_I2C_FIFO_Ctrl	I2C_BA+0x10		I²C FIFO Configuration Register	0x0000_0001
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:2]	Reserved			0x0
Bit[1:0]	FIFOSize	R	FIFO SIZE 0: 2 bytes 1: 4 bytes 2: 8 bytes 3: 16 bytes	0x1

- **P_I2C_INT_Ctrl (I²C Interrupt Enable Register)**

Register	Offset		Description	Initial Value
P_I2C_INT_Ctrl	I2C_BA+0x14		I²C Interrupt Enable Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:10]	Reserved			0x0
Bit[9]	Cmpl	R/W	Set to enable the Completion Interrupt Master: interrupts when a transaction is issued without losing the bus arbitration Slave: interrupts when a transaction addressing the controller completes	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit{8]	ByteRecv	R/W	Set to enable the Byte Receive Interrupt Interrupts when a byte of data is received Auto-ACK will be disabled if this interrupt is enabled, that is, the software needs to ACK/NACK the received byte manually.	0x0
Bit[7]	ByteTrans	R/W	Set to enable the Byte Transmit Interrupt Interrupts when a byte of data is transmitted	0x0
Bit[6]	Start	R/W	Set to enable the START Condition Interrupt Interrupts when a START condition/repeated START condition is detected	0x0
Bit[5]	Stop	R/W	Set to enable the STOP Condition Interrupt Interrupts when a STOP condition is detected	0x0
Bit[4]	ArbLose	R/W	Set to enable the Arbitration Lose Interrupt Master: interrupts when the controller loses the arbitration Slave: arbitration lose interrupt is not available at the slave mode	0x0
Bit[3]	AddrHit	R/W	Set to enable the Address Hit Interrupt Master: interrupts when the addressed slave returned an ACK Slave: interrupts when the controller is addressed	0x0
Bit[2]	FIFOHalf	R/W	Set to enable the FIFO Half Interrupt Receiver: Interrupts when the FIFO is half-full Transmitter: Interrupts when the FIFO is half-empty This interrupt depends on the transaction direction; do not enable this interrupt unless the transfer direction is determined, otherwise unintended interrupts may be triggered.	0x0
Bit1]	FIFOFull	R/W	Set to enable the FIFO Full Interrupt Interrupts when the FIFO is full	0x0
Bit[0]	FIFOEmpty	R/W	Set to enable the FIFO Empty Interrupt Interrupts when the FIFO is empty	0x0

- **P_I2C_Status (I²C Status Register)**

Register	Offset		Description	Initial Value
P_I2C_Status	I2C_BA+0x18		I²C Status Register	0xu000_0001
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:15]	Reserved			0x0
Bit[14]	LineSDA	R	Indicates the status of the SDA line on the bus 1: High 0: Low	SDA line status
Bit[13]	LineSCL	R	Indicates the status of the SCL line on the bus 1: High 0: Low	SCL line status
Bit[12]	GenCall	R	Indicates that the address of the current transaction is a general call address 1: General call 0: Not general call	0x0
Bit[11]	BusBusy	R	Indicates that the bus is busy The bus is busy when a START condition is on bus and it ends when a STOP condition is seen on bus. 1: Busy 0: Not busy	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[10]	ACK	R	Indicates the type of the last received/transmitted acknowledgement bit 1: ACK 0: NACK	0x0
Bit[9]	Cmpl	R/W1C	Transaction Completion Master: Indicate a transaction is issued without losing arbitration. Slave: Indicate a transaction addressing the controller completes. This status bit must be cleared to receive the next transaction; otherwise, the next incoming transaction will be blocked.	0x0
Bit{8]	ByteRecv	R/W1C	Indicates that a byte is received	0x0
Bit[7]	ByteTrans	R/W1C	Indicates that a byte is transmitted	0x0
Bit[6]	Start	R/W1C	Indicates that the START Condition is transmitted/received	0x0
Bit[5]	Stop	R/W1C	Indicates that the STOP Condition is transmitted/received	0x0
Bit[4]	ArbLose	R/W1C	Indicates that the controller loses bus arbitration (master mode only)	0x0
Bit[3]	AddrHit	R/W1C	Master: indicates a slave will respond to the transaction. Slave: indicates that a transaction is targeting the controller.	0x0
Bit[2]	FIFOHalf	R/W	Transmitter: Indicates that the FIFO is half-full. Receiver: Indicates that the FIFO is half-empty	0x0
Bit1]	FIFOFull	R/W	Indicates that the FIFO is full	0x0
Bit[0]	FIFOEmpty	R/W	Indicates that the FIFO is empty	0x01

- **P_I2C_Addr (I²C Address Register)**

Register	Offset		Description	Initial Value
P_I2C_Addr	I2C_BA+0x1C		I²C Address Register	0xu000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:10]	Reserved			
Bit[9:0]	Addr	R/W	The slave address. For 7-bit addressing mode, the most significant 3 bits are ignored and only the least-significant 7 bits of Addr are valid	0x0

- **P_I2C_Data (I²C Data Register)**

Register	Offset		Description	Initial Value
P_I2C_Data	I2C_BA+0x20		I²C Data Register	0xu000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0000_0000
Bit[7:0]	Data	R/W	Write this register to put one byte of data to the FIFO Read this register to get one byte of data from the FIFO	0x0

- **P_I2C_Mode (I²C Control Register)**

Register	Offset		Description	Initial Value
P_I2C_Mode	I2C_BA+0x24		I²C Control Register	0x0000_1E00
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:13]	Reserved			0x0
Bit[12]	Phase_start	R/W	Enable this bit to send a START condition at the beginning of transaction Master mode only	0x1

Bit	Name	R/W	Descriptions	Initial Value
Bit[11]	Phase_addr	R/W	Enable this bit to send the address after START condition Master mode only	0x1
Bit[10]	Phase_dat	R/W	Enable this bit to send the data after Address phase Master mode only	0x1
Bit[9]	Phase_stop	R/W	Enable this bit to send a STOP condition at the end of a transaction Master mode only	0x1
Bit[8]	Dir	R/W	Transaction direction Master: Set this bit to determine the direction for the next transaction. 0: Transmitter 1: Receiver Slave: The direction of the last received transaction. 0: Receiver 1: Transmitter	0x0
Bit[7:0]	DataCnt	R/W	Data counts in bytes. Master: The number of bytes to transmit/receive. 0 means 256 bytes. DataCnt will be decreased by one for each byte transmitted/received. Slave: the meaning of DataCnt depends on the DMA mode: If DMA is not enabled, DataCnt is the number of bytes transmitted/received from the bus master. It is reset to 0 when the controller is addressed and then increased by one for each byte of data transmitted/received. If DMA is enabled, DataCnt is the number of bytes to transmit/receive. It will not be reset to 0 when the slave is addressed and it will be decreased by one for each byte of data transmitted/received.	0x0

- **P_I2C_Cmd (I²C Command Register)**

Register	Offset		Description	Initial Value
P_I2C_Cmd	I2C_BA+0x28		I²C Command Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:3]	Reserved			0x0
Bit[2:0]	CMD	R/W	Write this register with the following values to perform the corresponding actions: 0x0: no action 0x1: issue a data transaction (Master only) 0x2: respond with an ACK to the received byte 0x3: respond with a NACK to the received byte 0x4: clear the FIFO 0x5: reset the I²C controller (abort current transaction, clear the Status Register and clear the FIFO) When issuing a data transaction by writing 0x1 to this register, the CMD field stays at 0x1 for the duration of the entire transaction, and it is only cleared to 0x0 after when the transaction has completed or when the controller loses the arbitration. ***Note: No transaction will be issued by the controller when all phases (Start, Address, Data and Stop) are disabled.***	0x0

- **P_I2C_Ctrl (I²C Controller Setting Register)**

Register	Offset		Description	Initial Value
P_I2C_Ctrl	I2C_BA+0x2C		I²C Controller Setting Register	0x0525_2100
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:29]	Reserved			0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[28:24]	T_SUDAT	R/W	T_SUDAT defines the data setup time before releasing the SCL Setup time = (4 + T_SP + T_SUDAT) * tpclk tpclk = PCLK period	0x5
Bit[23:21]	T_SP	R/W	T_SP defines the pulse width of spikes that must be suppressed by the input filter Pulse width = T_SP * tpclk	0x1
Bit[20:16]	T_HDDAT	R/W	T_SUDAT defines the data hold time after SCL goes LOW Hold time = (4 + T_SP + T_HDDAT) * tpclk	0x5
Bit[15:14]	Reserved			0x0
Bit[13]	T_SCLRatio	R/W	The LOW period of the generated SCL clock is defined by the combination of T_SCLRatio and T_SCLHi values. When T_SCLRatio = 0, the LOW period is equal to HIGH period. When T_SCLRatio = 1, the LOW period is roughly two times of HIGH period. SCL LOW period = (4 + T_SP + T_SCLHi * ratio) * tpclk 1: ratio = 2 0: ratio = 1 This field is only valid when the controller is in the master mode.	0x1
Bit[12:4]	T_SCLHi	R/W	The HIGH period of generated SCL clock is defined by T_SCLHi. SCL HIGH period = (4 + T_SP + T_SCLHi) * tpclk The T_SCLHi value must be greater than T_SP and T_HDDAT values. This field is only valid when the controller is in the master mode.	0x10
Bit[3]	DMAEn	R/W	Enable the direct memory access mode data transfer 1: Enabled 0: Disabled	0x0
Bit[2]	Master	R/W	Configure this device as a master or a slave 1: Master mode 0: Slave mode	0x0
Bit[1]	Addressing	R/W	I^2C addressing mode: 1: 10-bit addressing mode 0: 7-bit addressing mode	0x0
Bit[0]	IICEN	R/W	Enable the ATCIIC100 I^2C controller 1: Enabled 0: Disabled	0x0

2.11 Serial Peripheral Interface (SPI) Controller

2.11.1 Overview and Features

- There are 2 SPI masters (SPI0 and SPI1). Each master can connect 1 slave device only.
- The SPI0 and SPI1 support Transmit and Receive mode and only the SPI0 supports XIP modes.
- There are 16 bytes and 8 bytes of receive, and transmit FIFO in SPI0 and SPI1, respectively.
- The SPI0 supports single/dual/quad I/O mode, and SPI1 only supports single I/O mode.
- The maximum frequency of bit rate clock is SYS_CLK.

2.11.2 Block Diagram

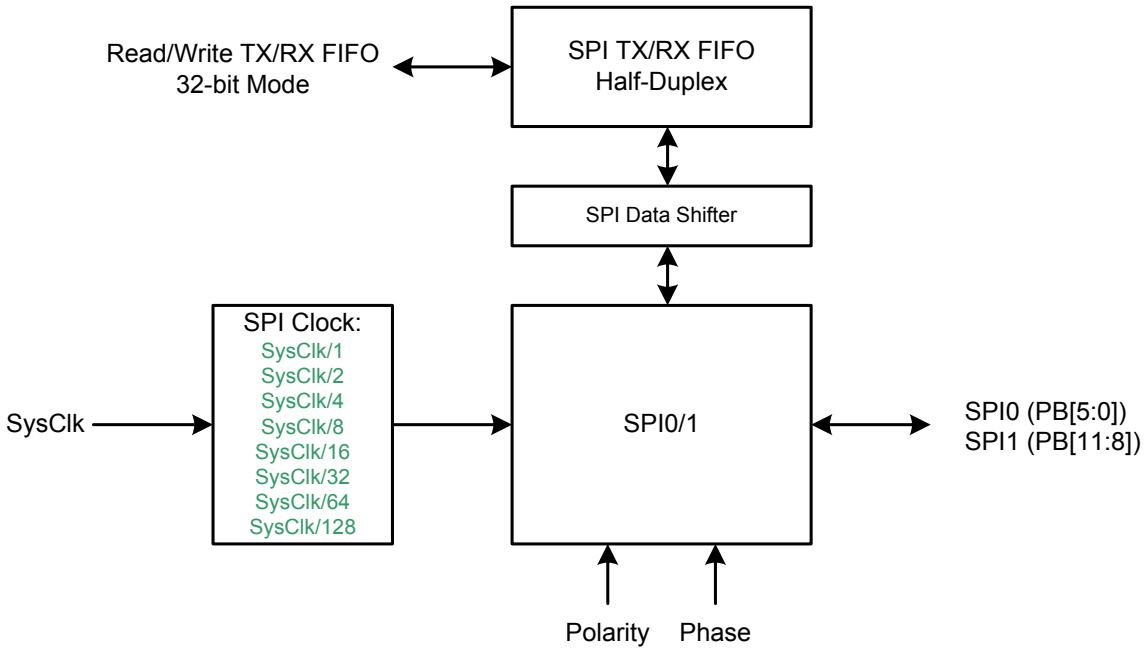


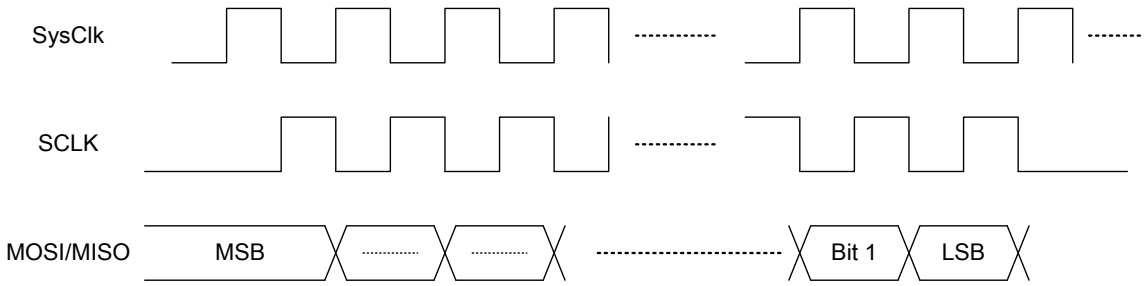
Figure 9 SPI Block Diagram

2.11.3 Function Description

There are 2 components in the SPI controller. One is a SPI0 single/dual/quad I/O master series interface protocol, the other is SPI1 single I/O master series interface protocol. They can be selected for connecting to a series-slave peripheral device, respectively.

2.11.3.1 Clock Ratios

The SPI controller is selected, the maximum frequency of the bit-rate clock (SCLK) is the same the frequency of SYC_CLK. The minimum frequency of the bit-rate clock (SCLK) is SYS_CLK / 128. The P_SPIx_Ctrl[10:8] (x=0, or 1) (SPISEL) bits are selected to generate the bit-rate clock (SCLK).



2.11.3.2 Interrupt

The SPI controller supports individual interrupt request. The SPI interrupts are described below.

- While the transmission byte number is equivalent to send byte number.
- While the transmission byte number is equivalent to receive byte number.

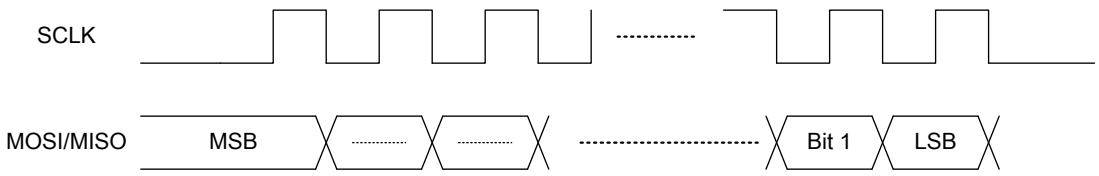
2.11.3.3 Transfer Mode

The SPI controller operates in the following two modes when transferring data on the serial bus:

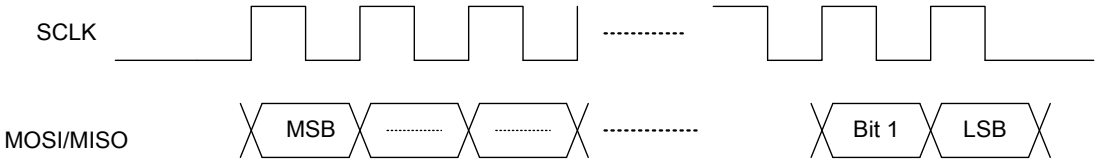
- Data mode

The P_SPIx_Ctrl[1] (TXD), or P_SPIx_Ctrl[0] (RXD) bit is set to start transmit data, or receive data. Transmit data are popped from FIFO and are sent through the MOSI line to the target device. The receive data from the target device through the MISO line is moved into FIFO. The P_SPIx_Ctrl[20-x:16] (x=0, or 1) (SPI_NUM) bits are expected transmit data bytes are equivalent to the transmit data bytes and the TXD bit is auto-clear, or expected receive data bytes are equivalent to the receive data bytes and the RXD bit is auto-clear. The P_SPI0_Ctrl[5:4] (SPI_DATS) bits are selected single, dual, or quad I/O mode to transmit/receive data.

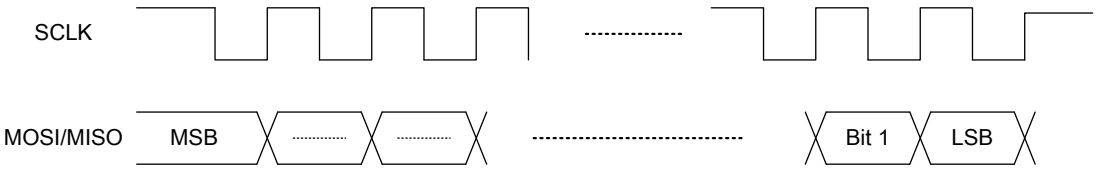
The clock polarity P_SPIx_Ctrl[7] (SPI_POL) configuration parameter determines whether the inactive state of the serial clock is high or low. To transmit data, both SPI peripherals must have identical serial clock phase P_SPIx_Ctrl[6] (SPI_PHA) and clock polarity P_SPIx_Ctrl[7] (SPI_POL) values. When the configuration parameter SPI_POL = 0, for SPI_PHA = 0, data are captured on the clock's rising edge and data is output on a falling edge, for SPI_PHA = 1, data are captured on the clock's falling and data is output on a rising edge. When the configuration parameter SPI_POL = 1, for SPI_PHA = 0, data are captured on the clock's falling edge and data is output on a rising edge, for SPI_PHA = 1, data are captured on the clock's rising and data is output on a falling edge.



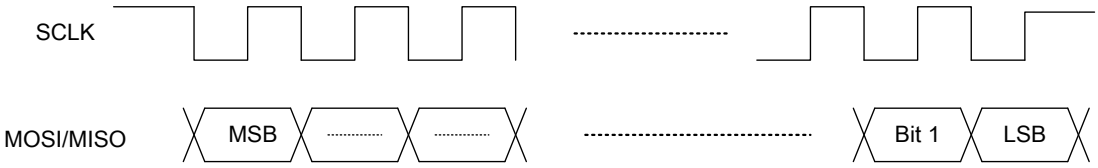
SPI_POL = 0, and SPI_PHA=0



SPI_POL = 0, and SPI_PHA=1



SPI_POL = 1, and SPI_PHA=0



SPI_POL = 1, and SPI_PHA=1

- XIP mode

The SPI0 supports XIP mode by using P_SPI0_Addr_Mode to control transfer I/O mode. The SPI0 in XIP mode is used to transmit a fixed command, an address and dummy cycles to the SPI flash device. Typically transmit data takes three data frames (8-bit command followed by 24/32-bit address and 0/8-bit dummy cycles). A transmission of the command and address is captured by a SPI flash device and the 32-bit data is shifted out by the SPI flash device. The P_SPI0_Addr_Mode[1:0] (SPI_FMT) bits are selected which command is sent to SPI flash device. The commands take Read Data(0x03), Fast Read Dual I/O(0xBB) and Fast Read Quad I/O(0xEB). The P_SPI0_Addr_Mode[2] (ADR_NUM) bit is selected 3/4-byte address mode. In XIP mode, the 2MB address range in SPI flash device is divided 2 parts, one is XIP_ROM1(1MB) and the other is XIP_ROM2(1MB). The address range of the XIP_ROM1 in the NX1 system is fixed at 0x800000 ~ 0x8FFFFF is mapped to 0x000000 ~ 0x0FFFFF in SPI flash device. The SPI0_INS[17:8] (SPI_BANK) bits are selected the address range of the XIP_ROM2 in the NX1 system is fixed at 0x900000 ~ 0x9FFFFF is mapped to 0xN00000 ~ 0xNFFFFF (N=SPI_BANK=0,1,2,...) in SPI flash device. For example: SPI_BANK = 2, the address range of the NX1 system is 0x900000 ~ 0x9FFFFF is mapped to 0x200000 ~ 0x2FFFFF in SPI flash device.

2.11.4 Register Map

Register	Offset	R/W	Description	Initial Value
SPI Base Address: SPI_BA = 0xF0_B000				
P_SPI0_Ctrl	SPI_BA+0x00	R/W	SPI0 Control Register	0x0000_0700
P_SPI0_Addr_Mode	SPI_BA+0x04	R/W	SPI0 Address Mode Setting Register	0x0000_0000
P_SPI0_FIFO_RST	SPI_BA+0x08	R/W	SPI0 FIFO Reset Register	0x0000_0004
P_SPI0_Data	SPI_BA+0x10	R/W	SPI0 Data Register	0x0000_0000
P_SPI1_Ctrl	SPI_BA+0x40	R/W	SPI1 Control Register	0x0000_0700
P_SPI1_FIFO_RST	SPI_BA+0x48	R/W	SPI1 FIFO Reset Register	0x0000_0004
P_SPI1_Data	SPI_BA+0x50	R/W	SPI1 Data Register	0x0000_0000

2.11.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_SPI0_Ctrl (SPI0 Control Register)**

Register	Offset		Description	Initial Value
P_SPI0_Ctrl	SPI_BA+0x00		SPI0 Control Register	0x0000_0700
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:20]	Reserved			0x0
Bit[19:16]	SPI_NUM	R/W	SPI Txd/Rcv 's Num byte	0x0
Bit[15:11]	Reserved			0x0
Bit[10:8]	SPISEL	R/W	111: SPICK0=SYSCLK/1 110: SPICK0=SYSCLK/2 101: SPICK0=SYSCLK/4 100: SPICK0=SYSCLK/8 011: SPICK0=SYSCLK/16 010: SPICK0=SYSCLK/32 001: SPICK0=SYSCLK/64 000: SPICK0=SYSCLK/128	0x7
Bit[7]	SPI_POL	R/W	1: SPICK stop at 1, when #CS = 1 0: SPICK stop at 0, When #CS = 1	0x0
Bit[6]	SPI_PHA	R/W	1: sampling on second clock edge 0: sampling on first clock edge	0x0
Bit[5:4]	SPI_DATS	R/W	11: SPI use 4 IO 10: SPI use 2 IO 0x: SPI use 1 IO	0x0
Bit[3:2]	Reserved			0x0
Bit[1]	TXD	R/W	1: start SEND DATA 0: do nothing	0x0
Bit[0]	RXD	R/W	1: start Receive DATA 0: do nothing	0x0

- **P_SPI0_Addr_Mode (SPI0 Address Mode Setting Register)**

Register	Offset		Description	Initial Value
P_SPI0_Addr_Mode	SPI_BA+0x04		SPI0 Address Mode Setting Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:18]	Reserved			0x0
Bit[17:8]	SPI_BANK	R/W	XIP mode Bank select	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[7:3]	Reserved			0x0
Bit[2]	ADR_NUM	R/W	1: 4 ADR for Instruction mode 0: 3 ADR for Instruction mode	0x0
Bit[1:0]	SPI_FMT	R/W	11: command EBH, 5W, 1-4-4 10: command BBH, 5W, 1-2-2 0x: command 03H, 5W, 1-1-1	0x0

- **P_SPI0_FIFO_RST (SPI0 FIFO Reset Register)**

Register	Offset		Description	Initial Value
P_SPI0_FIFO_RST	SPI_BA+0x08		SPI0 FIFO Reset Register	0x0000_0004
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31]	Reserved			0x0
Bit[30]	FIFO_RESET	R/W	1: FIFO reset 0: FIFO no reset	0x0
Bit[29:0]	Reserved			0x4

- **P_SPI0_Data (SPI0 Data Register)**

Register	Offset		Description	Initial Value
P_SPI0_Data	SPI_BA+0x10		SPI0 Data Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	FIFO_D	R/W	Write Data into FIFO Read Data from FIFO	0x0

- **P_SPI1_Ctrl (SPI1 Control Register)**

Register	Offset		Description	Initial Value
P_SPI1_Ctrl	SPI_BA+0x40		SPI1 Control Register	0x0000_0700
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:19]	Reserved			
Bit[18:16]	SPI_NUM	R/W	SPI Txd/Rcv 's Num byte	0x0
Bit[15:11]	Reserved			0x0
Bit[10:8]	SPISEL	R/W	111: SPICK1=SYSCLK/1 110: SPICK1=SYSCLK/2 101: SPICK1=SYSCLK/4 100: SPICK1=SYSCLK/8 011: SPICK1=SYSCLK/16 010: SPICK1=SYSCLK/32 001: SPICK1=SYSCLK/64 000: SPICK1=SYSCLK/128	0x7
Bit[7]	SPI_POL	R/W	1: SPICK stop at 1, when #CS = 1 0: SPICK stop at 0, When #CS = 1	0x0
Bit[6]	SPI_PHA	R/W	1: sampling on second clock edge 0: sampling on first clock edge	0x0
Bit[5:2]	Reserved			0x0
Bit[1]	TXD	R/W	1: start SEND DATA 0: do nothing	0x0
Bit[0]	RXD	R/W	1: start Receive DATA 0: do nothing	0x0

- **P_SPI1_FIFO_RST (SPI1 FIFO Reset Register)**

Register	Offset		Description	Initial Value
P_SPI1_FIFO_RST	SPI_BA+0x48		SPI1 FIFO Reset Register	0x0000_0004
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31]	Reserved			0x0
Bit[30]	FIFO_RESET	R/W	1: FIFO reset 0: FIFO no reset	0x0
Bit[29:0]	Reserved			0x4

- **P_SPI1_Data (SPI1 Data Register)**

Register	Offset		Description	Initial Value
P_SPI1_Data	SPI_BA+0x50		SPI1 Data Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	FIFO_D	R/W	Write Data into FIFO Read Data from FIFO	0x0

2.12 UART Interface Controller

2.12.1 Overview and Features

The NX1 provides two UART (Universal Asynchronous Receiver Transmitter) modules, UART is a full-duplex, asynchronous communication channel that communicates with peripheral devices and personal computers through protocols such as RS-232.

- Supports 5 to 8 bits per character
- Supports 1, 1.5 and 2 STOP bits
- Supports even, odd and stick parity bits
- Supports programmable baud rate
- Supports parity errors, framing errors and data overrun detection

2.12.2 Block Diagram

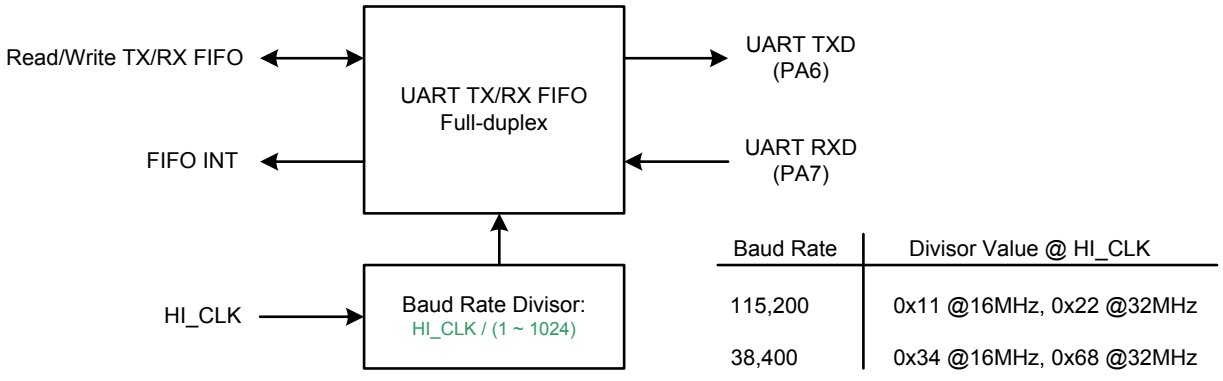


Figure 10 UART Block Diagram

2.12.3 Function Description

User need to enable UART's block by write 1 to P_SMU_FUNC_Ctrl [9] (URCLK_EN) before using UART function.

Baud Rate Generator: Program the Baud Rate registers (P_UART_BaudRate) to select the desired communication baud rate for the UART mode. The baud rate is given by:

$$\text{Baud rate} = \frac{\text{sys\_clk}}{(\text{P\_UART\_BaudRate}+1)*8}$$

Transmitter: Writes data to TX FIFO via data register (P_UART_Data), IP will start transmitter and stop until FIFO is empty.

Receiver: UART receive data and saves it to FIFO, user can read out the data via data register (P_UART_Data).

UART support six interrupt methods check P_UART0_Ctrl/ P_UART_Ctrl [5:0] for further info.

2.12.4 Register Map

Register	Offset	R/W	Description	Initial Value
UART Base Address: UART_BA = 0xF0_2000, offset = 0x40, n=set number (0,1)				
P_UART_Data	UART_BA+0x00	R/W	UART Data Register	0x0000_0000
P_UART_Ctrl	UART_BA+0x04	R/W	UART Control Register	0x000A_1300
P_UART_Flag	UART_BA+0x08	R/W1C	UART Status Register	0x0000_0022
P_UART_BaudRate	UART_BA+0x0C	R/W	UART Baud Rate Divider Register	0x0000_0022

2.12.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_UART_Data (UART Data Register)**

Register	Offset		Description	Initial Value
P_UART_Data	UART_BA+0x00		UART Data Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	Reserved			
Bit[7:0]	UDR	R/W	write to TXD FIFO in WR (THR) read from RXD FIFO in RD (RBR)	0x0

- **P_UART_Ctrl (UART Control Register)**

Register	Offset		Description	Initial Value
P_UART_Ctrl	UART_BA+0x04		UART Control Register	0x000a_1300
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:21]	Reserved			
Bit[19:18]	TX_FIFO_LEVEL	R/W	TX's FIFO level small than level have interrupt	0x2
Bit[17:16]	RX_FIFO_LEVEL	R/W	RX's FIFO level big than level have interrupt	0x2
Bit[15]	T_FIFO_R	W	1: Reset TXD's FIFO 0: no reset	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[14]	R_FIFO_R	W	1: Reset RXD's FIFO 0: no reset	0x0
Bit[13]	Reserved			0x0
Bit[12]	EP	R/W	1: even parity 0: odd parity	0x1
Bit[11]	PE	R/W	1: one parity bit 0: no parity bits	0x0
Bit[10]	STPS	R/W	1: if WLS=0, 1.5 stop, else 2 stop 0: 1 stop	0x0
Bit[9:8]	WLS	R/W	11: 8 bits 10: 7 bits 01: 6 bits 00: 5 bits	0x3
Bit[7]	Reserved			0x0
Bit[6]	LSR_IEN	R/W	1: LSR_INT enable 0: LSR_INT disable	0x0
Bit[5]	TX_FIFO_FLAG_IEN	R/W	1: TX_FIFO_FLAG_INT enable 0: TX_FIFO_FLAG_INT disable	0x0
Bit[4]	RX_FIFO_FLAG_IEN	R/W	1: RX_FIFO_FLAG_INT enable 0: RX_FIFO_FLAG_INT disable	0x0
Bit[3]	T_FIFO_FULL_IEN	R/W	1: T_FIFO_FULL_INT enable 0: T_FIFO_FULL_INT disable	0x0
Bit[2]	R_FIFO_FULL_IEN	R/W	1: R_FIFO_FULL_INT enable 0: R_FIFO_FULL_INT disable	0x0
Bit[1]	TX_DONE_IEN	R/W	1: T_DONE_INT enable 0: T_DONE_INT disable	0x0
Bit[0]	R_FIFO_NOEMPTY_IEN	R/W	1: R_FIFO_NOEMPTY_INT enable 0: R_FIFO_NOEMPTY_INT disable	0x0

- **P_UART_Flag (UART Status Register)**

Register	Offset		Description	Initial Value
P_UART_Flag	UART_BA+0x08		UART Status Register	0x0000_0022
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[11]		R/W1C	1: have framing error (stop not high) 0: no framing error	0x0
Bit[10]	PE	R/W1C	1: have parity error (parity not match) 0: no parity error	0x0
Bit[9]	OE	R/W1C	1: have overrun error (FIFO overrun) 0: no overrun error	0x0
Bit[8:6]	Reserved			0x0
Bit[5]	TX_FIFO_FLAG	R	1: TX's FIFO flag is set 0: TX's FIFO flag is not set	0x1
Bit[4]	RX_FIFO_FLAG	R	1: RX's FIFO flag is set 0: RX's FIFO flag is not set	0x0
Bit[3]	TX_FIFO_FULL	R	1: TX FIFO is FULL 0: TX FIFO not FULL (default)	0x0
Bit[2]	RX_FIFO_FULL	R	1: RX FIFO is FULL 0: RX FIFO not FULL (default)	0x0
Bit[1]	TX_FIFO_EMPTY	R/W1C	1: TX FIFO is EMPTY (default) 0: TX FIFO not EMPTY	0x1
Bit[0]	R_FIFO_NOEMPTY	R	1: RX FIFO not EMPTY 0: RX FIFO is EMPTY (default)	0x0

- **P_UART_BaudRate (UART BaudRate Divider Register)**

Register	Offset		Description	Initial Value
P_UART_BaudRate	UART_BA+0x0C		UART Baud Rate Divider Register	0x0000_0022
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:10]	Reserved			
Bit[9:0]	BDIV	R/W	Divisor's Data	0x22

2.13 SD Card Controller

2.13.1 Overview and Features

- Supports Secure Digital memory protocol commands
- Supports Secure Digital I/O protocol commands
- Supports the SD memory card protocol ver. 2.0
- No SPI mode included
- Variable clock rate: 16MHz of the SD card
- Hot insertion / removal
- Write-protect for the SD card

2.13.2 Block Diagram

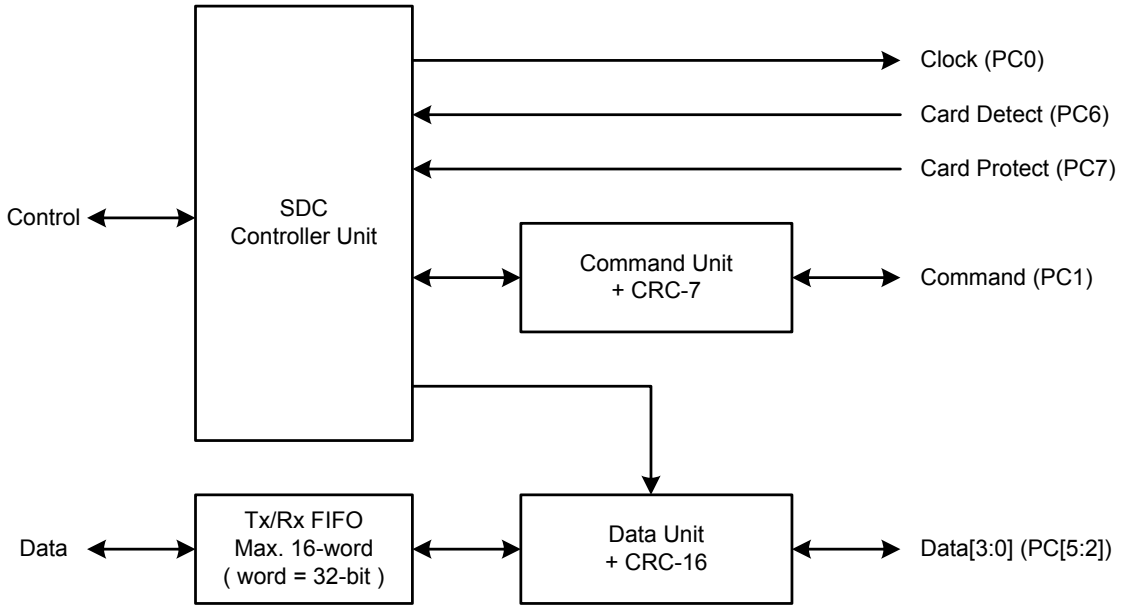


Figure 11 SDC Block Diagram

2.13.3 Function Description

The Secure Digital host controller (SDC) is to be as the master in a SD memory card interface. The core supports the SD bus protocol of the SD card and is compatible with SD memory card protocol version

2.0. There are 8 words depth of the TX FIFO and RX FIFO. In the card identification mode, the maximum clock frequency is 400KHz. During the data transfer mode, the maximum clock frequency is 16MHz for the SD memory card. A 7-bit CRC is appended to the commands and checked for the responses and a 16-bit CRC is appended and checked for every block of data transfer mode. The result of CRC check will be recorded in the status register. When once the data CRC error happens in the middle of a data transfer, the core will terminate the remained data transfer.

2.13.4 Register Map

Register	Offset	R/W	Description	Initial Value
SDC Base Address: SDC_BA = 0xF0_E000				
P_SDC_CMD	SDC_BA+0x00	R/W	SDC Command Register	0x0000_0000
P_SDC_ARG	SDC_BA+0x04	R/W	SDC Argument register	0x0000_0000
P_SDC_RSP_Status0	SDC_BA+0x08	R	Response register0	0x0000_0000
P_SDC_RSP_Status1	SDC_BA+0x0C	R	Response register1	0x0000_0000
P_SDC_RSP_Status2	SDC_BA+0x10	R	Response register2	0x0000_0000
P_SDC_RSP_Status3	SDC_BA+0x14	R	Response register3	0x0000_0000
P_SDC_RSP_CMD	SDC_BA+0x18	R	Responded command register	0x0000_0000
P_SDC_RW_Ctrl	SDC_BA+0x1C	R/W	Data control register	0x0000_0000
P_SDC_TimeOut	SDC_BA+0x20	R/W	Data timer register	0x0000_0000
P_SDC_LEN	SDC_BA+0x24	R/W	Data length register	0x0000_0000
P_SDC_Status	SDC_BA+0x28	R	Status register	0x0000_0000
P_SDC_CLR	SDC_BA+0x2C	W	Clear register	0x0000_0000
P_SDC_INT_Mask	SDC_BA+0x30	R/W	Interrupt mask register	0x0000_0000
P_SDC_PWR_Ctrl	SDC_BA+0x34	R/W	Power control register	0x0000_0010
P_SDC_CLK	SDC_BA+0x38	R/W	Clock control register	0x0000_01FF
P_SDC_BUS_Mode	SDC_BA+0x3C	R/W	Bus width register	0x0000_0009
P_SDC_Data	SDC_BA+0x40	R/W	Data window register	0xuuuu_uuuu
P_SDC_MMC_INT	SDC_BA+0x44	R/W	MMC interrupt response time register	0x0000_0042
P_SDC_GPIO_Out	SDC_BA+0x48	R/W	General purpose output	0x0000_0000
P_SDC_SDIO_Ctrl1	SDC_BA+0x6C	R/W	SDIO control register1	0x0000_0000
P_SDC_SDIO_Ctrl2	SDC_BA+0x70	R/W	SDIO control regiset2	0x0000_0000
P_SDC_SDIO_Status	SDC_BA+0x70	R	SDIO status register	0x0000_0000
P_SDC_FIFO_Ctrl	SDC_BA+0x9C	R	Feature register	0x0000_0008

2.13.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_SDC_CMD (SDC Command Register)**

Register	Offset		Description	Initial Value
P_SDC_CMD	SDC_BA+0x00		SDC Command Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[11]	MMC_INT_STOP	R/W	If set, the MMC host controller will send out R5 response to inform MMC card to exit Wait_IRQ state. Before setting this bit, user must confirm CMD_EN is 0.	0x0
Bit[10]	SDC_RST	R/W	1: the SD host controller reset 0: (default)	0x0
Bit[9]	CMD_EN	R/W	1: command enable 0: (default)	0x0
Bit[8]	APP_CMP	R/W	1: indicate the command is an application specific command 0: (default)	0x0
Bit[7]	LONG_RSP	R/W	1: this command waits for 136-bit long response 0: (default)	0x0
Bit[6]	NEED_RSP	R/W	1: this command waits for a response from card 0: (default)	0x0
Bit[5:0]	CMD_IDX	R/W	command index	0x0

- **P_SDC_ARG (SDC Argument register)**

Register	Offset		Description	Initial Value
P_SDC_ARG	SDC_BA+0x04		SDC Argument register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	CMD_ARG	R/W	Argument of command	0x0000_00000

- **P_SDC_RSP_Status0 (Response register0)**

Register	Offset		Description	Initial Value
P_SDC_RSP_Status0	SDC_BA+0x08		Response register0	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	CARD_STA	R	Card status send by card	0x0000_00000

- **P_SDC_RSP_Status1 (Response register1)**

Register	Offset		Description	Initial Value
P_SDC_RSP_Status1	SDC_BA+0x0C		Response register1	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	CARD_STA	R	Card status send by card	

- **P_SDC_RSP_Status2 (Response register2)**

Register	Offset		Description	Initial Value
P_SDC_RSP_Status2	SDC_BA+0x10		Response register2	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	CARD_STA	R	Card status send by card	0x0000_00000

- **P_SDC_RSP_Status3 (Response register3)**

Register	Offset		Description	Initial Value
P_SDC_RSP_Status3	SDC_BA+0x14		Response register3	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	CARD_STA	R	Card status send by card	0x0000_00000

- **P_SDC_RSP_CMD (Responded command register)**

Register	Offset		Description	Initial Value
P_SDC_RSP_CMD	SDC_BA+0x18		Responded command register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:7]	Reserved			0x0
Bit[6]	RSP_CMD_APP	R	This bit indicates the command is an application specific command	0x0
Bit[5:0]	RSP_CMD_IDX	R	Response command index	0x0

- **P_SDC_RW_Ctrl Data control register)**

Register	Offset		Description	Initial Value
P_SDC_RW_Ctrl	SDC_BA+0x1C		Data control register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:11]	Reserved			0x0
Bit[10]	FIFO_RST	W	1: FIFO RESET	0x0
Bit[9:8]	DMA_TYPE	R/W	DMA type	0x0
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[7]	FIFOTH	R/W	FIFO threshold	0x0
Bit[6]	DATA_EN	R/W	1: enable data transfer cycle 0:	0x0
Bit[5]	DMA_EN	R/W	1: DMA transfer enable 0:	0x0
Bit[4]	DATA_WRITE	R/W	1: Write data to card 0: Read data from card	0x0
Bit[3:0]	BLK_SIZE	R/W	Data size per block (Bytes)	0x0

- **P_SDC_TimeOut (Data timer register)**

Register	Offset		Description	Initial Value
P_SDC_TimeOut	SDC_BA+0x20		Data timer register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	DATA_TIMER	R/W	Data timeout period (In card bus clock period)	0x0000_0000

- **P_SDC_LEN (Data length register)**

Register	Offset		Description	Initial Value
P_SDC_LEN	SDC_BA+0x24		Data length register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	DATA_LEN	R/W	Data bytes to be transferred The max volume cannot exceed 128MB in the SDIO and combo card application.	0x0000_0000

- **P_SDC_Status (Status register)**

Register	Offset		Description	Initial Value
P_SDC_Status	SDC_BA+0x28		Status register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:18]	Reserved			0x0
Bit[17]	DATAO_STATUS	R	Real status of the pin DATAO. The interrupt will be asserted when the pin "DATAO" is high	0x0
Bit[16]	SDIO_IRPT	R/W	SDIO card interrupt	0x0
Bit[15:13]	Reserved			0x0
Bit[12]	WRITE_PROT	R	1: Card is write-protected 0: Card is not write-protected	0x0
Bit[11]	CARD_DETECT	R	1: Card is removed 0: Card is inserted	0x0
Bit[10]	CARD_CHANGE	R	Card is inserted or removed	0x0
Bit[9]	FIFO_ORUN	R	Data FIFO is available to read	0x0
Bit[8]	FIFO_URUN	R	Data FIFO is available to write	0x0
Bit[7]	DATA_END	R	Data transfer finished	0x0
Bit[6]	CMD_SENT	R	Command sent	0x0
Bit[5]	DATA_CRC_OK	R	Data block send/received and CRC check passed	0x0
Bit[4]	RSP_CRC_OK	R	Command response received and CRC check passed	0x0
Bit[3]	DATA_TIMEOUT	R	Data read/programming timeout	0x0
Bit[2]	RSP_TIMEOUT	R	Command response timeout	0x0
Bit[1]	DATA_CRC_FAIL	R	Data block sent/received but CRC check failed	0x0
Bit[0]	RSP_CRC_FAIL	R	Command response received but CRC check failed	0x0

- **P_SDC_CLR (Clear register)**

Register	Offset		Description	Initial Value
P_SDC_CLR	SDC_BA+0x2C		Clear register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:17]	Reserved			0x0
Bit[16]	CLR_SDIO_IRPT	W	clear SDIO card interrupt flag	0x0
Bit[15:11]	Reserved			0x0
Bit[10]	CLR_CARD_CHANGE	W	Clear Card_CHANGE flag	0x0
Bit[9]	Reserved			0x0
Bit[8]	Reserved			0x0
Bit[7]	CLR_DATA_END	W	Clear Data_END flag	0x0
Bit[6]	CLR_CMD_SENT	W	Clear CMD_SENT flag	0x0
Bit[5]	CLR_DATA_OK	W	Clear DATA_CRC_OK flag	0x0
Bit[4]	CLR_RSP_OK	W	clear RSP_CRC_OK flag	0x0
Bit[3]	CLR_DATA_TIMEOUT	W	Clear DATA_TIMEOUT flag	0x0
Bit[2]	CLR_RSP_TIMEOUT	R	Clear RSP_TIMEOUT flag	0x0
Bit[1]	CLR_DATA_FAIL	W	Clear DATA_CRC_FAIL flag	0x0
Bit[0]	CLR_RSP_FAIL	W	Clear RSP_CRC_FAIL flag	0x0

- **P_SDC_INT_Mask (Interrupt mask register)**

Register	Offset		Description	Initial Value
P_SDC_INT_Mask	SDC_BA+0x30		Interrupt mask register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:18]	Reserved			0x0
Bit[17]	Mask17	R/W	Mask DATAO_STATUS flag	0x0
Bit[16]	Mask16	R/W	Mask SDIO card interrupt	0x0
Bit[15:13]	Reserved			0x0
Bit[12]	Reserved			0x0
Bit[11]	Reserved			0x0
Bit[10]	Mask10	R/W	Mask Card_CHANGE flag	0x0
Bit[9]	Mask9	R/W	Mask FIFO_ORUN flag	0x0
Bit[8]	Mask8	R/W	Mask FIFO_NRUN flag	0x0
Bit[7]	Mask7	R/W	Mask Data_END flag	0x0
Bit[6]	Mask6	R/W	Mask CMD_SENT flag	0x0
Bit[5]	Mask5	R/W	Mask DATA_CRC_OK flag	0x0
Bit[4]	Mask4	R/W	Mask RSP_CRC_OK flag	0x0
Bit[3]	Mask3	R/W	Mask DATA_TIMEOUT flag	0x0
Bit[2]	Mask2	R/W	Mask RSP_TIMEOUT flag	0x0
Bit[1]	Mask1	R/W	Mask DATA_CRC_FAIL flag	0x0
Bit[0]	Mask0	R/W	Mask RSP_CRC_FAIL flag	0x0

- **P_SDC_PWR_Ctrl (Power control register)**

Register	Offset		Description	Initial Value
P_SDC_PWR_Ctrl	SDC_BA+0x34		Power control register	0x0000_0010
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:5]	Reserved			0x0
Bit[4]	SD_POWER_ON	R/W	1: external power supply on 0: external power supply off	0x1
Bit[3:0]	SD_POWER	R/W	Control the external power supply output range	0x0

- **P_SDC_CLK (Clock control register)**

Register	Offset		Description	Initial Value
P_SDC_CLK	SDC_BA+0x38		Clock control register	0x0000_01FF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:9]	Reserved			0x0
Bit[8]	CLK_DIS	R/W	1: SD_CLK is disabled 0: SD_CLK is enable	0x1
Bit[7]	CLK_SD	R/W	1: SD memory card 0: MMC memory card	0x1
Bit[6:0]	CLK_DIV	R/W	Control the clock frequency of IO_SD_CLK IO_SD_CLK = SD_CLK/2x(CLK_DIV+1)	0x7

- **P_SDC_BUS_Mode (Bus width register)**

Register	Offset	Description	Initial Value
P_SDC_BUS_Mode	SDC_BA+0x3C	Bus width register	0x0000_0009

Bit	Name	R/W	Descriptions	Initial Value
Bit[31:6]	Reserved			0x0
Bit[5]	CARD_DETECT	R/W	1: Card detection is through the sense of DATA3 0: Card detection is through switch on the connector	0x0
Bit[4:3]	WIDE_BUS_SUPPORT	R/W	10: 8-bit bus width is implemented 01: 4-bit bus width is implemented 00: 1-bit bus width supported only	0x1
Bit[2]	WIDE_4_BUS	R/W	Bus width=4(DAT0~3) when WIDE_BUS_SUPPORT is enabled	0x0
Bit[1]	WIDE_8_BUS	R/W	Bus width=8(DAT0~7) when WIDE_BUS_SUPPORT is enabled	0x0
Bit[0]	SINGLE_BUS	R/W	Bus width=1(DAT0)	0x1

- **P_SDC_Data (Data window register)**

Register	Offset		Description	Initial Value
P_SDC_Data	SDC_BA+0x40		Data window register	0xuuuu_uuuu
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:0]	DATA_WIN	R/W	Data access port	0xuuuu_uuuu

- **P_SDC_MMC_INT (MMC interrupt response time register)**

Register	Offset		Description	Initial Value
P_SDC_MMC_INT	SDC_BA+0x44		MMC interrupt response time register	0x0000_0042
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			
Bit[7:0]	MMC_INT_TIME	R/W	MMC interrupt response time	0x42

- **P_SDC_GPIO_Out (General purpose output)**

Register	Offset		Description	Initial Value
P_SDC_GPIO_Out	SDC_BA+0x48		General purpose output	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:4]	Reserved			0x0
Bit[3:0]	SDC_GPO	R/W	Direct drive the level of the 4 general output ports	0x0

- **P_SDC_SDIO_Ctrl1 (SDIO control register1)**

Register	Offset		Description	Initial Value
P_SDC_SDIO_Ctrl1	SDC_BA+0x6C		SDIO control register1	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:15]	SDIO_BLK_NO	R/W	The total block number for a data transfer This field should be set before the data transfer. The maximum block number in a single data transfer is 131071, the minimum block number is 1.	0x0
Bit[14]	SDIO_EN	R/W	When a SDIO card inserts, set this bit to enable the SDIO related function.	0x0
Bit[13]	READ_WAIT_EN	R/W	If the SDIO card support the Read_Wait feature, set this bit.	0x0
Bit[12]	SDIO_BLK_MODE	R/W	If the host wants to initiate a multiple block transaction, set this bit.	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[11:0]	MMC_INT_TIME	R/W	Data size (Bytes) per block In a single block transaction, the maximum block size is 512. In a multiple block transaction, the maximum block size is 2048. Once the SDIO_EN is set, this field records the data block size no matter it is the memory or IO data transfer.	0x0

- **P_SDC_SDIO_Ctrl2 (SDIO control regiset2)**

Register	Offset		Description	Initial Value
P_SDC_SDIO_Ctrl2	SDC_BA+0x70		SDIO control regiset2	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:2]	Reserved			0x0
Bit[1]	SUSP_CMD_ABORT	R/W	When the users need to abort the suspend transaction, this bit must be set first. Then the abort command can be sent	0x0
Bit[0]	SUSP_READ_WAIT	R/W	This bit must be set before sending out the suspend command. The bus will keep the read-wait state until this bit is cleared	0x0

- **P_SDC_SDIO_Status (SDIO status register)**

Register	Offset		Description	Initial Value
P_SDC_SDIO_Status	SDC_BA+0x70		SDIO status register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:24]	Reserved			0x0
Bit[23:17]	FIFO_REMAIN_NO	R	This field records the number of the valid data (Words) remained in the FIFO	0x0
Bit[16:0]	SDIO_BLK_CNT	R	This field records the number of the block remained in a data transfer	0x0

- **P_SDC_FIFO_Ctrl (Feature register)**

Register	Offset		Description	Initial Value
P_SDC_FIFO_Ctrl	SDC_BA+0x9C		Feature register	0x0000_0008
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:9]	Reserved			0x0
Bit[8]	CPRM_FUNCTION	R	1: contain the CPRM function 0: don't contain the CPRM function	0x0
Bit[7:0]	FIFO_DEPTH	R	0x04: FIFO depth is 4 0x08: FIFO depth is 8 0x10: FIFO depth is16	0x8

2.14 PWM-IO Generator and Capture Timer

2.14.1 Overview and Features

The NX1 has 2 sets of PWM-IO groups - PWMA / PWMB, each with four PWM outputs (PWMA0 ~ PWMA3, PWMB0 ~ PWMB3). Each four PWM outputs share a PWM timer, every PWM output has its own duty and output port.

Complementary PWM pairs also provided for PWMA0 / PWMA1 and PWMB0 / PWMB1, where dead zone generator is supported as well. Capture function is supported at PWMA, where associated PWM outputs are disabled at the mean time.

- Four PWM outputs share a timer
- Programmable divider of timer clock
- 16-bit counter for each timer
- 8-bit length for dead zone time
- Capture source is one of PA0-PA15
- 16-bit PWM duty

2.14.2 Block Diagram

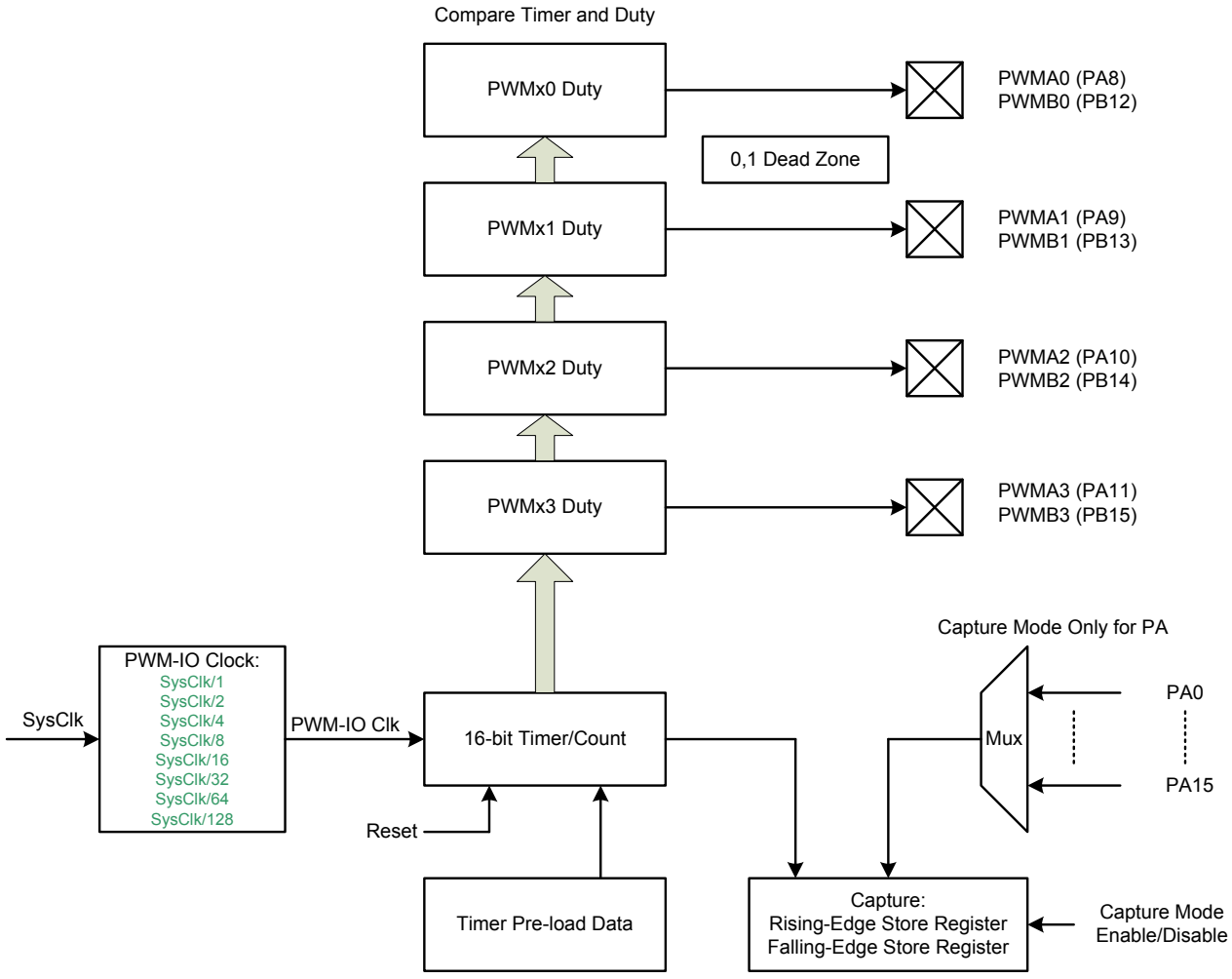


Figure 12 PWM-IO Block Diagram

2.14.3 Function Description

Each PWM-IO generator has its own duty register and four PWM output pins share one timer.

Capture mode: The NX1 monitors the rising/falling of the input signal (from PA0 ~ PA15). When a rising/falling-edge is detected, the NX1 will write the Timer/Counter values to the P_PWMA_CAP_Data register.

Dead zone: PWMA0/PWMA1 and PWMB0/PWMB1 are complementary PWM outputs, where the Dead Zone generator is used to insert a programmable time gap between rising PWM outputs to prevent from simultaneous active condition that may lead to very large current. Users can change the width by modifying P_PWMx_TMR_Ctrl [15:8] (DZ_IR).

2.14.4 Register Map

Register	Offset	R/W	Description	Initial Value
PWM Base Address: PWM_BA = 0xF1_1000				
P_PWMA_TMR_Ctrl	PWM_BA+0x00	R/W	PWMA Timer Control Register	0x0000_0000
P_PWMA_TMR_Data	PWM_BA+0x04	R/W	PWMA Timer Data Register	0xFFFF_FFFF
P_PWMA_TMR_INT_Ctrl	PWM_BA+0x08	R/W	PWMA Timer Interrupt Enable	0x0000_0000
P_PWMA_TMR_Flag	PWM_BA+0x0C	R/W1C	PWMA Timer Interrupt Flag	0x0000_0000
P_PWMA0_Duty	PWM_BA+0x10	R/W	PWMA0 duty Register	0x0000_0000
P_PWMA1_Duty	PWM_BA+0x14	R/W	PWMA1 duty Register	0x0000_0000
P_PWMA2_Duty	PWM_BA+0x18	R/W	PWMA2 duty Register	0x0000_0000
P_PWMA3_Duty	PWM_BA+0x1C	R/W	PWMA3 duty Register	0x0000_0000
P_PWMA_CAP_Data	PWM_BA+0x20	R	PWMA Capture Data Register	0x0000_0000
P_PWMA_CAP_INT_Ctrl	PWM_BA+0x24	R/W	PWMA Capture Interrupt Enable	0x0000_0000
P_PWMA_CAP_Flag	PWM_BA+0x28	R/W1C	PWMA Capture Interrupt Flag	0x0000_0000
P_PWMB_TMR_Ctrl	PWM_BA+0x40	R/W	PWMB Timer Control Register	0x0000_0000
P_PWMB_TMR_Data	PWM_BA+0x44	R/W	PWMB Timer Data Register	0xFFFF_FFFF
P_PWMB_TMR_INT_Ctrl	PWM_BA+0x48	R/W	PWMB Timer Interrupt Enable	0x0000_0000
P_PWMB_TMR_Flag	PWM_BA+0x4C	R/W1C	PWMB Timer Interrupt Flag	0x0000_0000
P_PWMB0_Duty	PWM_BA+0x50	R/W	PWMB0 duty Register	0x0000_0000
P_PWMB1_Duty	PWM_BA+0x54	R/W	PWMB1duty Register	0x0000_0000
P_PWMB2_Duty	PWM_BA+0x58	R/W	PWMB2 duty Register	0x0000_0000
P_PWMB3_Duty	PWM_BA+0x5C	R/W	PWMB3 duty Register	0x0000_0000

2.14.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_PWMA_TMR_Ctrl (PWMA Timer Control Register)**

Register	Offset		Description	Initial Value
P_PWMA_TMR_Ctrl	PWM_BA+0x00		PWMA Timer Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:20]	Reserved			
Bit[19:16]	CAP_SEL	R/W	1111: PA15 ~ 0000: PA0	0x0
Bit[15:8]	DZ_IR	R/W	Dead Zone Interval Register these 8 bits determine dead zone length	0x0
Bit[7]	Reserved			
Bit[6:4]	PWMTMRSEL	R/W	111: PWMTMRCK0= SYSCLK/1 110: PWMTMRCK0=SYSCLK/2 101: PWMTMRCK0=SYSCLK/4 100: PWMTMRCK0=SYSCLK/8 011: PWMTMRCK0=SYSCLK/16 010: PWMTMRCK0=SYSCLK/32 001: PWMTMRCK0=SYSCLK/64 000: PWMTMRCK0=SYSCLK/128	0x0
Bit[3]	CAP_EN	R/W	1: capture mode enabled 0: capture mode disabled	0x0
Bit[2]	DZ_EN	R/W	1: dead zone enabled 0: dead zone disabled	0x0
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[1]	PWMTMR_STR	W	1: reload data 0: do nothing	0x0
Bit[0]	PWMTMR_EN	R/W	1: PWMTMR enabled 0: PWMTMR disabled	0x0

- **P_PWMA_TMR_Data (PWMA Timer Data Register)**

Register	Offset		Description	Initial Value
P_PWMA_TMR_Data	PWM_BA+0x04		PWMA Timer Control Register	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMTMR_DAT	R	PWMTMR's counter data	0xFFFF
Bit[15:0]	PWMTMR_LDAT	R/W	PWMTMR's preload Data	0xFFFF

- **P_PWMA_TMR_INT_Ctrl (PWMA Timer Interrupt Enable)**

Register	Offset		Description	Initial Value
P_PWMA_TMR_INT_Ctrl	PWM_BA+0x08		PWMA Timer Interrupt Enable	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	PWMTMR_IEN	R/W	1: TMR's interrupt enabled 0: TMR's interrupt disabled	0x0

- **P_PWMA_TMR_Flag (PWMA Timer Interrupt Flag)**

Register	Offset	Description	Initial Value
P_PWMA_TMR_Flag	PWM_BA+0x0C	PWMA Timer Interrupt Flag	0x0000_0000

Bit	Name	R/W	Descriptions	Initial Value
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_INTF	R/W1C	1: TMR0's interrupt flag 0: TMR0's interrupt no flag	0x0

- **P_PWMA0_Duty (PWMA0 Duty Register)**

Register	Offset		Description	Initial Value
P_PWMA0_Duty	PWM_BA+0x10		PWMA IO0's duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMA0_DUTY	R/W	PWMA0 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMA0_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMA0_EN	R/W	1: PWMA0 enabled 0: PWMA0 disabled	0x0

- **P_PWMA1_Duty (PWMA IO1's duty Register)**

Register	Offset		Description	Initial Value
P_PWMA1_Duty	PWM_BA+0x14		PWMA1 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMA1_DUTY	R/W	PWMA1 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMA1_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMA1_EN	R/W	1: PWMA1 enabled 0: PWMA1 disabled	0x0

- **P_PWMA2_Duty (PWMA2 duty Register)**

Register	Offset		Description	Initial Value
P_PWMA2_Duty	PWM_BA+0x18		PWMA2 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMA2_DUTY	R/W	PWMA2 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMA2_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMA2_EN	R/W	1: PWMA2 enabled 0: PWMA2 disabled	0x0

- **P_PWMA3_Duty (PWMA3 duty Register)**

Register	Offset		Description	Initial Value
P_PWMA3_Duty	PWM_BA+0x1C		PWMA3 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMA3_DUTY	R/W	PWMA3 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMA3_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMA3_EN	R/W	1: PWMA3 enabled 0: PWMA3 disabled	0x0

- **P_PWMA_CAP_Data (PWMA Capture Data Register)**

Register	Offset		Description	Initial Value
P_PWMA_CAP_Data	PWM_BA+0x20		PWMA Capture Data Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	CAPR_DAT	R	capture rising edge latch data	0x0
Bit[15:0]	CAPF_DAT	R	capture falling edge latch data	0x0

- **P_PWMA_CAP_INT_Ctrl (PWMA Capture Interrupt Enable)**

Register	Offset		Description	Initial Value
P_PWMA_CAP_INT_Ctrl	PWM_BA+0x24		PWMA Capture Interrupt Enable	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:6]	Reserved			0x0
Bit[5]	CAPR_IEN	R/W	1: capture rising interrupt enabled 0: capture rising interrupt disabled	0x0
Bit[4]	CAPF_IEN	R/W	1: capture falling interrupt enabled 0: capture falling interrupt disabled	0x0
Bit[3:0]	Reserved			0x0

- **P_PWMA_CAP_Flag (PWMA Capture Interrupt Flag)**

Register	Offset		Description	Initial Value
P_PWMA_CAP_Flag	PWM_BA+0x28		PWMA Capture Interrupt Flag	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:6]	Reserved			0x0
Bit[5]	CAPR_INTF	R/W1C	1: have capture rising interrupt flag 0: no capture rising interrupt flag	0x0
Bit[4]	CAPF_INTF	R/W1C	1: have capture falling interrupt flag 0: no capture falling interrupt flag	0x0
Bit[3:0]	Reserved			0x0

- **P_PWMB_TMR_Ctrl (PWMB Timer Control Register)**

Register	Offset		Description	Initial Value
P_PWMB_TMR_Ctrl	PWM_BA+0x40		PWMB Timer Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:20]	Reserved			
Bit[19:16]	CAP_SEL	R/W	1111: PA15 ~ 0000: PA0	0x0
Bit[15:8]	DZ_IR	R/W	Dead Zone Interval Register these 8 bits determine dead zone length	0x0
Bit[7]	Reserved			
Bit[6:4]	PWMTMRSEL	R/W	111: PWMTMRCK1= SYSCLK/1 110: PWMTMRCK1=SYSCLK/2 101: PWMTMRCK1=SYSCLK/4 100: PWMTMRCK1=SYSCLK/8 011: PWMTMRCK1=SYSCLK/16 010: PWMTMRCK1=SYSCLK/32 001: PWMTMRCK1=SYSCLK/64 000: PWMTMRCK1=SYSCLK/128	0x0
Bit[3]	Reserved			0x0
Bit[2]	DZ_EN	R/W	1: dead zone enabled 0: dead zone disabled	0x0

Bit[1]	PWMTMR_STR	W	1: reload data 0: do nothing	0x0
Bit[0]	PWMTMR_EN	R/W	1: PWMTMR enabled 0: PWMTMR disabled	0x0

- **P_PWMB_TMR_Data (PWMB Timer Data Register)**

Register	Offset		Description	Initial Value
P_PWMB_TMR_Data	PWM_BA+0x44		PWMB Timer Control Register	0xFFFF_FFFF
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMTMR_DAT	R	PWMTMR's counter data	0xFFFF
Bit[15:0]	PWMTMR_LDAT	R/W	PWMTMR's preload Data	0xFFFF

- **P_PWMB_TMR_INT_Ctrl (PWMB Timer Interrupt Enable)**

Register	Offset		Description	Initial Value
P_PWM1_TMR_INT_Ctrl	PWM_BA+0x48		PWM1 Timer Interrupt Enable	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	PWMTMR_IEN	R/W	1: TMR's interrupt enable 0: TMR's interrupt disable	0x0

- **P_PWMB_TMR_Flag (PWMB Timer Interrupt Flag)**

Register	Offset		Description	Initial Value
P_PWMB_TMR_Flag	PWM_BA+0x4C		PWMB Timer Interrupt Flag	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			0x0
Bit[0]	TMR_INTF	R/W1C	1: TMR1's interrupt flag 0: TMR1's interrupt no flag	0x0

- **P_PWMB0_Duty (PWMB0 duty Register)**

Register	Offset		Description	Initial Value
P_PWMB0_Duty	PWM_BA+0x50		PWMB0 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMB0_DUTY	R/W	PWMB0 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMB0_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMB0_EN	R/W	1: PWMB0 enabled 0: PWMB0 disabled	0x0

- **P_PWMB1_Duty (PWMB1 duty Register)**

Register	Offset		Description	Initial Value
P_PWMB1_Duty	PWM_BA+0x54		PWMB1 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMB1_DUTY	R/W	PWMB1 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMB1_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0

Bit[0]	PWMB1_EN	R/W	1: PWMB1 enabled 0: PWMB1 disabled	0x0

- **P_PWMB2_Duty (PWMB2 duty Register)**

Register	Offset		Description	Initial Value
P_PWMB2_Duty	PWM_BA+0x58		PWMB2 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMB2_DUTY	R/W	PWMB2 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMB2_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMB2_EN	R/W	1: PWMB2 enabled 0: PWMB2 disabled	0x0

- **P_PWMB3_Duty (PWMB3 duty Register)**

Register	Offset		Description	Initial Value
P_PWMB3_Duty	PWM_BA+0x5C		PWMB3 duty Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	PWMB3_DUTY	R/W	PWMB3 duty data	0x0
Bit[15:2]	Reserved			0x0
Bit[1]	PWMB3_DSB	R/W	1: Sink (PWM start 1) 0: Drive (PWM start 0)	0x0
Bit[0]	PWMB3_EN	R/W	1: PWMB3 enabled 0: PWMB3 disabled	0x0

2.15 IR TX

2.15.1 Overview and Features

The NX1 provide a 5-bit IR carrier, which can generate different IR frequency by assigning different counter values.

- Support output stop at 0 or 1.
- Support 5 bits reload data to adjust IR's frequency.

2.15.2 Block Diagram

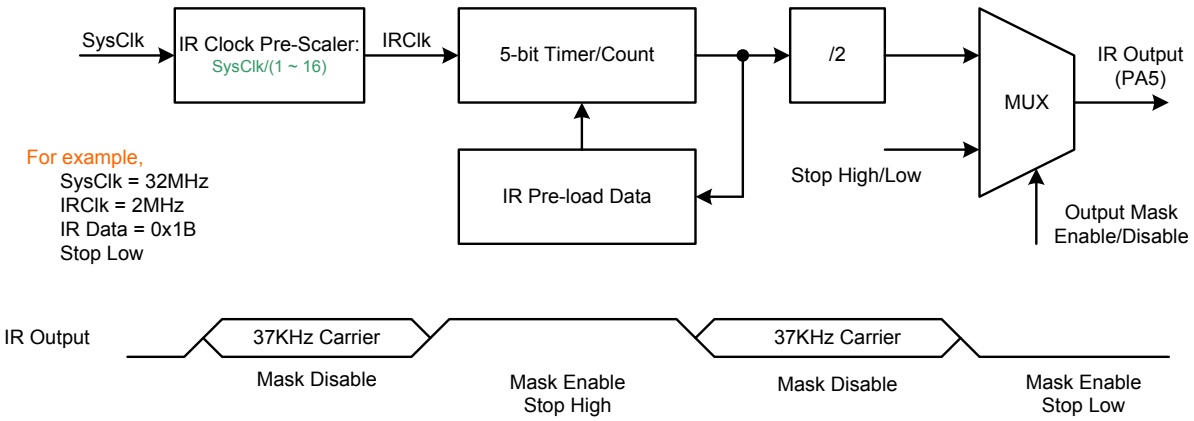


Figure 13 IR Block Diagram

2.15.3 Function Description

Set CLK_DIV to make IRClk be a 2MHz clock and then modify IR_LDAT to change the IR output frequency.

2.15.4 Register Map

Register	Offset	R/W	Description	Initial Value
IR Base Address: IR_BA = 0xF0_F000				
P_IR_Ctrl	IR_BA+0x00	R/W	IR Control Register	0x0000_1F00

2.15.5 Register Description

- **P_IR_Ctrl (IR Control Register)**

Register	Offset		Description	Initial Value
P_IR_Ctrl	IR_BA+0x00		IR Control Register	0x0000_1F00
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:13]	Reserved			0x0
Bit[12:8]	IR_LDAT	R/W	IR counter pre-load data (default 0x1F)	0x1F
Bit[7:4]	CLK_DIV	R/W	CLK divide to 2MHz	0x0
Bit[3:2]	Reserved			0x0
Bit[1]	IR_STP	R/W	1: IR disable stop to 1 0: IR disable stop to 0	0x0
Bit[0]	IR_EN	R/W	1: IR enabled 0: IR disabled	0x0

2.16 Real Time Clock (RTC)

2.16.1 Overview and Features

The RTC support periodic time tick interrupts with 4 period options: 62.5us (or 0.25ms), 1ms, 15.625ms and 0.5s.

2.16.2 Block Diagram

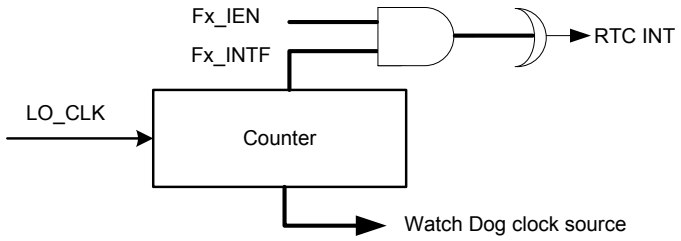


Figure 14 RTC Block Diagram

2.16.3 Function Description

Real Time Clock (RTC) provides fixed / regular interrupt signals for time-keeping applications. The clock source is either the internal built-in I_LRC, or from external crystal (32,768Hz) oscillator. The RTC supports time tick interrupts. There are 4 period options: 62.5us (or 0.25ms), 1ms, 15.625ms and 0.5s.

2.16.4 Register Map

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

Register	Offset	R/W	Description	Initial Value
RTC Base Address: RTC_BA = 0xF0_6000				
P_RTC_INT_Ctrl	RTC_BA+0x00	R/W	RTC Interrupt Enable Register	0x0000_0000
P_RTC_Flag	RTC_BA+0x04	R/W1C	RTC Interrupt Flag Register	0x0000_0000

2.16.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_RTC_INT_Ctrl (RTC Interrupt Enable Register)**

Register	Offset		Description	Initial Value
P_RTC_INT_Ctrl	RTC_BA+0x00		RTC Interrupt Enable Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:4]	Reserved			0x0
Bit[3]	F16KHZ_IEN (or F4KHZ_IEN)	R/W	1: interrupt enable 0: interrupt disable	0x0
Bit[2]	F1KHZ_IEN	R/W	1: interrupt enable 0: interrupt disable	0x0
Bit[1]	F64HZ_IEN	R/W	1: interrupt enable 0: interrupt disable	0x0
Bit[0]	F2HZ_IEN	R/W	1: interrupt enable 0: interrupt disable	0x0

- **P_RTC_Flag (RTC Interrupt Flag Register)**

Register	Offset		Description	Initial Value
P_RTC_Flag	RTC_BA+0x04		RTC Interrupt Flag Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:4]	Reserved			0x0
Bit[3]	F16KHZ_INTF (or F4KHZ_INTF)	R/W1C	1: have interrupt flag 0: no interrupt flag	0x0
Bit[2]	F1KHZ_INTF	R/W1C	1: have interrupt flag 0: no interrupt flag	0x0
Bit[1]	F64HZ_INTF	R/W1C	1: have interrupt flag 0: no interrupt flag	0x0
Bit[0]	F2HZ_INTF	R/W1C	1: have interrupt flag 0: no interrupt flag	0x0

2.17 Watchdog Timer (WDT)

2.17.1 Overview and Features

- Counter
- Programmable period register

2.17.2 Block Diagram

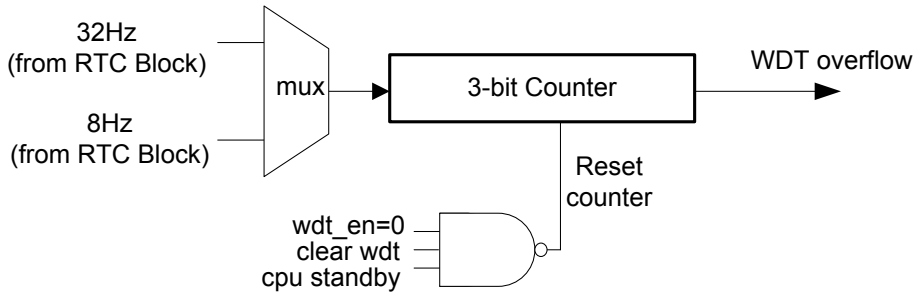


Figure 15 WDT Block Diagram

2.17.3 Function Description

The Watchdog Timer (WDT) is used to perform a system reset when the system is not responding as expected. It would prevent the system from hanging-up for an infinite period.

2.17.4 Register Map

R: Read only, **W**: Write only, **R/W**: Read and Write

Register	Offset	R/W	Description	Initial Value
WDT Base Address: WDT_BA = 0xF0_7000				
P_WDT_Ctrl	RTC_BA+0x00	R/W	WDT Control Register	0x0000_0000
P_WDT_CLR	RTC_BA+0x04	W	WDT Clear Register	0x0000_0000

2.17.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_WDT_Ctrl (WDT Control Register)**

Register	Offset		Description	Initial Value
P_WDT_Ctrl	RTC_BA+0x00		WDT Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:1]	Reserved			
Bit[0]	WDT_TIME	R/W	1: 188ms ~ 219ms (= 1/32Hz * 6 ~ 7) 0: 750ms ~ 875ms (= 1/8Hz * 6 ~ 7)	0x0

- **P_WDT_CLR (WDT Clear Register)**

Register	Offset		Description	Initial Value
P_WDT_CLR	RTC_BA+0x04		WDT Clear Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0
Bit[15:0]	CLR_WDT	W	When CLR_WDT = 0x3CA5, clear WDT counter	0x0

2.18 Analog-to-Digital Converter

2.18.1 Overview and Features

The NX1 provides one 12-bit Analog-to-Digital converter with eight input channels. The A/D converter supports single and continuous scan mode. It can be started by software or TIMER trigger.

- Provide 8 level FIFO or data registers for each channel.
- Auto scan mode.

2.18.2 Block Diagram

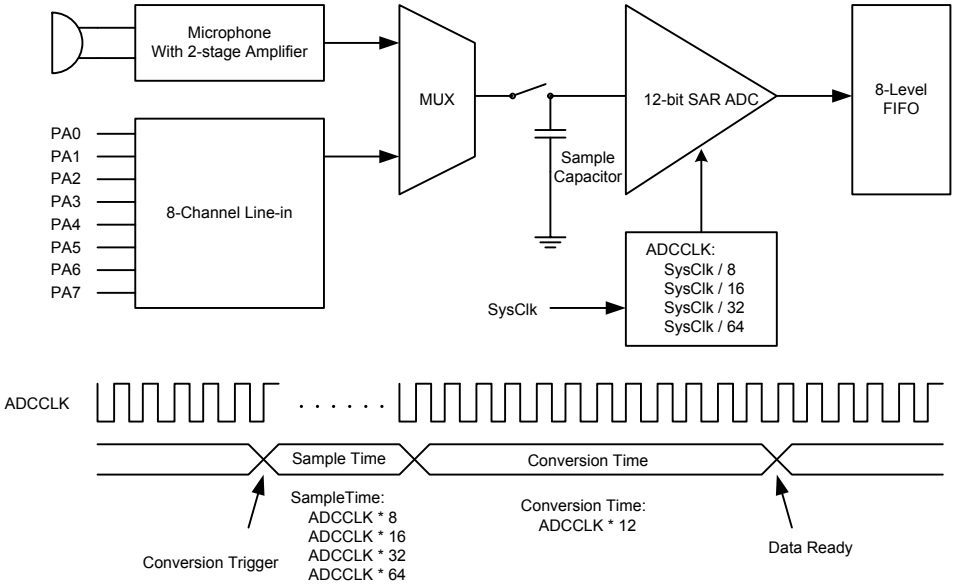


Figure 16 ADC Block Diagram

2.18.3 Function Description

The NX1 provides 4 trigger sources and 9 analog input signals as shown in Figure 16 ADC Block Diagram.

Data register: the NX1 provides 2 ways to store the data, one is to save all channels' data to FIFO; the other way is to save data to individual addresses.

Scan mode: the NX1 supports scan modes that can auto get 4/3/2 channel's data by just one trigger.

2.18.4 Register Map

Register	Offset	R/W	Description	Initial Value
ADC Base Address: ADC_BA = 0xF1_4000				
P_ADC_Ctrl	ADC_BA+0x00	R/W	ADC Control Register	0x0000_E000
P_ADC_AGC_Ctrl	ADC_BA+0x04	R/W	AGC Control Register	0x0000_00C0
P_ADC_FIFO_Ctrl	ADC_BA+0x08	R/W	ADC FIFO Control Register	0x0000_0040
P_ADC_Flag	ADC_BA+0x0C	R/W	ADC FIFO Status Register	0x0000_0000
P_ADC_Data_CH0	ADC_BA+0x10	R	ADC Channel 0 Register	0x0000_0000
P_ADC_Data_CH1	ADC_BA+0x14	R	ADC Channel 1 Register	0x0000_0000
P_ADC_Data_CH2	ADC_BA+0x18	R	ADC Channel 2 Register	0x0000_0000
P_ADC_Data_CH3	ADC_BA+0x1C	R	ADC Channel 3 Register	0x0000_0000
P_ADC_Data_CH4	ADC_BA+0x20	R	ADC Channel 4 Register	0x0000_0000
P_ADC_Data_CH5	ADC_BA+0x24	R	ADC Channel 5 Register	0x0000_0000
P_ADC_Data_CH6	ADC_BA+0x28	R	ADC Channel 6 Register	0x0000_0000
P_ADC_Data_CH7	ADC_BA+0x2C	R	ADC Channel 7 Register	0x0000_0000

2.18.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_ADC_Ctrl (ADC Control Register)**

Register	Offset		Description	Initial Value
P_ADC_Ctrl	ADC_BA+0x00		ADC Control Register	0x0000_E000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:15]	Reserved			0x0
Bit[15:14]	EN_4CLK	R/W	11: sample 4 clock 10: sample 8 clock 01: sample 16 clock 00: sample 32 clock	0x3
Bit[13]	CHG_EN	R/W	1: enable channel pre-charged to ½*VDD 0: disable channel pre-charged to ½*VDD	0x1
Bit[12]	ADC_EN	R/W	1: ADC enabled 0: ADC disabled	0x0
Bit[11]	Reserved			0x0
Bit[10]	SOFTWARE_TRIG	W		0x0
Bit[9:8]	ANSEL	R/W	11: ANCK=SYSCLK/(2*4) 10: ANCK=SYSCLK/(4*4) 01: ANCK=SYSCLK/(8*4) 00: ANCK=SYSCLK/(16*4)	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[7:6]	SCAN_MODE	R/W	11: CH3 -> CH2 -> CH1 -> CH0/MIC loop 10: CH2 -> CH1 -> CH0/MIC loop 01: CH1 -> CH0/MIC loop 00: SCAN MODE disabled	0x0
Bit[5:4]	ADC_TRIG_SEL	R/W	11: Timer2 Triggered 10: Timer1 Triggered 01: Timer0 Triggered 00: Software Triggered	0x0
Bit[3]	MIC_SEL	R/W	1: MIC channel selected 0: CH0 channel selected	0x0
Bit[2:0]	ADC_CHS	R/W	111: CH7 110: CH6 101: CH5 100: CH4 011: CH3 010: CH2 001: CH1 000: CH0 / MIC	0x0

- **P_ADC_AGC_Ctrl (AGC Control Register)**

Register	Offset		Description	Initial Value
P_ADC_AGC_Ctrl	ADC_BA+0x04		AGC Control Register	0x0000_00C0
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:15]	Reserved			0x0
Bit[14:12]	AGC_CKSEL	R/W	N = 0 ~ 6: AGCCLK = LOCLK/(2*(N+1)) N = 7: AGCCLK = LOCLK	0x0
Bit[11:9]	Reserved			0x0
Bit[8:4]	PGA_GAIN	R/W	PGA gain selection	0xC
Bit[3]	AGC_EN	R/W	1: AGC enabled 0: PGA enabled	0x0
Bit[2]	MAX_GAIN	R/W	1: mic gain 30 times (Only for AGC mode) 0: mic gain 15 times	0x0
Bit[1]	Reserved			0x0
Bit[0]	MIC_EN	R/W	1: V_MIC power supply enabled 0: V_MIC power supply disabled	0x0

- **P_ADC_FIFO_Ctrl (ADC FIFO Control Register)**

Register	Offset		Description	Initial Value
P_ADC_FIFO_Ctrl	ADC_BA+0x08		ADC FIFO Control Register	0x0000_0040
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:5]	Reserved			0x0
Bit[9]	ALLCH_USE_FIFO	R/W	1: all ADC channel's data store to FIFO 0: ADC channel's data store to its data address	0x0
Bit[8]	AFIFO_RESET	W	1: FIFO reset 0: FIFO no reset	0x0
Bit[7]	Reserved			0x0
Bit[6:4]	AFIFO_LEVEL	R/W	When FIFO number is big than level, set FIFO_FLAG	0x4
Bit[3]	ADIF_FLAG_IEN	R/W	1: ADC convert finish INT enabled 0: ADC convert finish INT disabled	0x0
Bit[2]	AFIFO_FLAG_IEN	R/W	1: AFIFO_FLAG_INT enabled 0: AFIFO_FLAG_INT disabled	0x0
Bit[1]	AFIFO_EMPTY_IEN	R/W	1: AFIFO_EMPTY_INT enabled 0: AFIFO_EMPTY_INT disabled	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[0]	AFIFO_FULL_IEN	R/W	1: AFIFO_FULL_INT enabled 0: AFIFO_FULL_INT disabled	0x0

- **P_ADC_Flag (ADC FIFO Status Register)**

Register	Offset		Description	Initial Value
P_ADC_Flag	ADC_BA+0x0C		ADC FIFO Status Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:4]	Reserved			0x0
Bit[3]	ADIF_FLAG	R/W	1: ADC convert finish flag 0: no ADC convert finish flag (default)	0x0
Bit[2]	AFIFO_FLAG	R/W1C	1: FIFO flag set 0: no FIFO flag	0x0
Bit[1]	AFIFO_EMPTY	R/W1C	1: FIFO empty flag set 0: no FIFO empty flag	0x0
Bit[0]	AFIFO_FULL	R/W1C	1: FIFO full flag set 0: no FIFO full flag	0x0

- **P_ADC_Data_CH0 (ADC Channel 0 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH0	ADC_BA+0x10		ADC Channel 0 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH0_DAT	R	ADC's Channel 0 Register	0x0
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH1 (ADC Channel 1 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH1	ADC_BA+0x14		ADC Channel 1 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH1_DAT	R	ADC's Channel 1 Register	0x0
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH2 (ADC Channel 2 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH2	ADC_BA+0x18		ADC Channel 2 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH2_DAT	R	ADC's Channel 2 Register	0x0
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH3 (ADC Channel 3 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH3	ADC_BA+0x1C		ADC Channel 3 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH3_DAT	R	ADC's Channel 3 Register	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH4 (ADC Channel 4 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH4	ADC_BA+0x20		ADC Channel 4 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH4_DAT	R	ADC's Channel 4 Register	0x0
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH5 (ADC Channel 5 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH5	ADC_BA+0x24		ADC Channel 5 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH5_DAT	R	ADC's Channel 5 Register	0x0
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH6 (ADC Channel 6 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH6	ADC_BA+0x28		ADC Channel 6 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH6_DAT	R	ADC's Channel 6 Register	0x0
Bit[3:0]	Reserved			0x0

- **P_ADC_Data_CH7 (ADC Channel 7 Register)**

Register	Offset		Description	Initial Value
P_ADC_Data_CH7	ADC_BA+0x2C		ADC Channel 7 Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[15:4]	CH7_DAT	R	ADC's Channel 7 Register	0x0
Bit[3:0]	Reserved			0x0

2.19 Digital-to-Analog Converter and Push-Pull Power Amplifier

2.19.1 Overview and Features

The NX1 provides two 14-bit Digital-to-Analog converters with interpolation function. It can be started by software or TIMER trigger.

- Provide 8-level FIFO
- Provide hardware interpolation function
- Support mixing mode for two DACs applications

2.19.2 Block Diagram

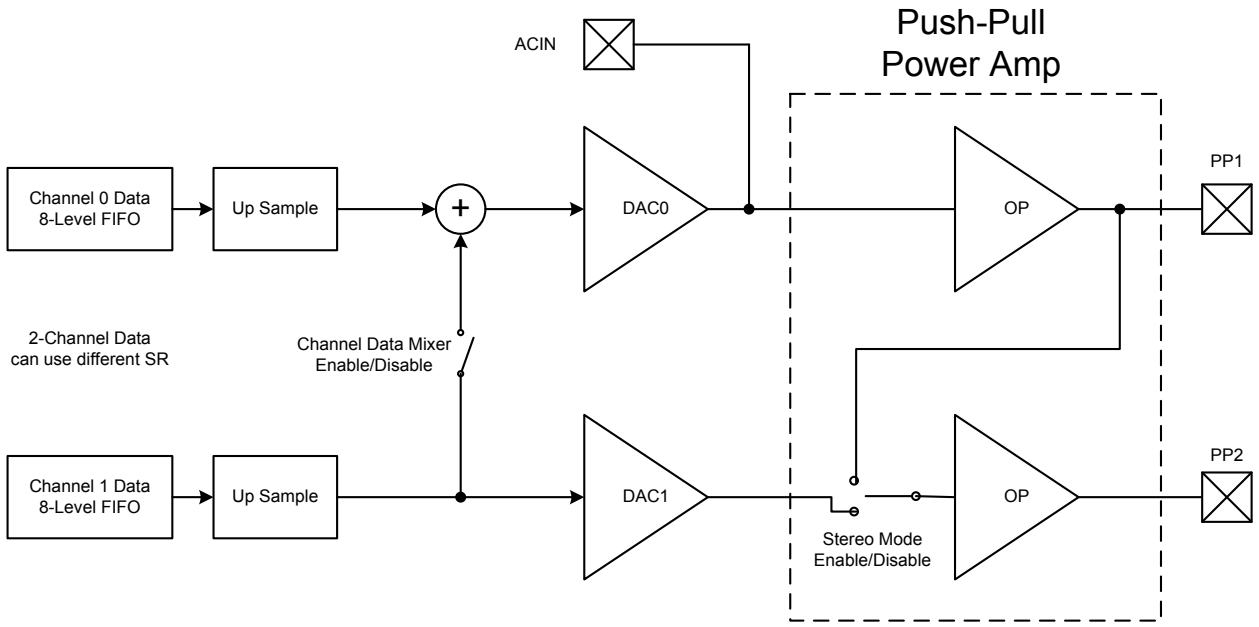


Figure 17 Block Diagram of DAC and PP PA

2.19.3 Function Description

The NX1 supports 4 trigger sources, DAC0 support DAC or push-pull mode, DAC1 only support DAC mode.

MIX: Mix mode supported only when two DACs are enabled.

ITP (**I**n**T**er**P**olation): Enable ITP also requires the speed-up of timer at the same time (by 4 times). The hardware ITP will generate another 3 data points between 2 consecutive FIFO data.

2.19.4 Register Map

Register	Offset	R/W	Description	Initial Value
DAC Base Address: DAC_BA = 0xF1_5000				
P_DAC_CH0_Ctrl	DAC_BA+0x00	R/W	DAC0 Control Register	0x0000_0800
P_DAC_CH0_Data	DAC_BA+0x04	W	DAC0 FIFO Data Register	0x0000_0000
P_DAC_CH0_FIFO	DAC_BA+0x08	R/W	DAC0 FIFO Control Register	0x0000_0040
P_DAC_CH0_Flag	DAC_BA+0x0C	R/W1C	DAC0 FIFO Status Register	0x0000_0006
P_DAC_CH1_Ctrl	DAC_BA+0x40	R/W	DAC1 Control Register	0x0000_0000
P_DAC_CH1_Data	DAC_BA+0x44	W	DAC1 FIFO Data Register	0x0000_0000
P_DAC_CH1_FIFO	DAC_BA+0x48	R/W	DAC1 FIFO Control Register	0x0000_0040
P_DAC_CH1_Flag	DAC_BA+0x4C	R/W1C	DAC1 FIFO Status Register	0x0000_0006

2.19.5 Register Description

R: Read only, **W**: Write only, **W1C**: Write 1 to clear, **R/W**: Read and Write

- **P_DAC_CH0_Ctrl (DAC0 Control Register)**

Register	Offset		Description	Initial Value
P_DAC_CH0_Ctrl	DAC_BA+0x00		DAC0 Control Register	0x0000_0800
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:12]	Reserved			0x0
Bit[11:8]	PP_GAIN	R/W	PP gain selection	0x8
Bit[7]	Reserved			0x0
Bit[6]	TRIG_ALL	R/W	1: DAC0's data trig by manually or DAC_TRIG_SEL 0: DAC0's data trig only by manually	0x0
Bit[5:4]	DAC_TRIG_SEL	R/W	11: Latch DATA to DAC by Timer2 10: Latch DATA to DAC by Timer1 01: Latch DATA to DAC by Timer0 00: Manual Latch	0x0
Bit[3]	ITP_EN	R/W	1: ITP enabled 0: ITP disabled	0x0
Bit[2]	ACIN_EN	R/W	1: ACIN enabled 0: ACIN disabled	0x0
Bit[1]	MODE	R/W	1: DAC mode 0: Push-Pull mode	0x0
Bit[0]	DAC_EN	R/W	1: Function enabled 0: Function disabled	0x0

- **P_DAC_CH0_Data (DAC0 FIFO Data Register)**

Register	Offset		Description	Initial Value
P_DAC_CH0_Data	DAC_BA+0x04		DAC0 FIFO Data Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:16]	Reserved			0x0
Bit[15:2]	DAC_DATA		DAC's DATA	0x0
Bit[1:0]	Reserved			0x0

- **P_DAC_CH0_FIFO (DAC0 FIFO Control Register)**

Register	Offset		Description	Initial Value
P_DAC_CH0_FIFO	DAC_BA+0x08		DAC0 FIFO Control Register	0x0000_0040
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:9]	Reserved			0x0
Bit[8]	DFIFO_RESET	W	1: FIFO reset 0: FIFO no reset (default)	0x0
Bit[7]	Reserved			0x0
Bit[6:4]	DFIFO_LEVEL	R/W	When FIFO number is smaller than level, set FIFO_FLAG (default=4)	0x4
Bit[3]	Reserved			0x0
Bit[2]	DFIFO_FLAG_IEN	R/W	1: DFIFO_FLAG_INT enabled 0: DFIFO_FLAG_INT disabled	0x0
Bit[1]	DFIFO_EMPTY_IEN	R/W	1: DFIFO_EMPTY_INT enabled 0: ADC_FIFO_EMPTY_INT disabled	0x0
Bit[0]	DFIFO_FULL_IEN	R/W	1: DFIFO_FULL_INT enabled 0: DFIFO_FULL_INT disabled	0x0

- **P_DAC_CH0_Flag (DAC0 FIFO Status Register)**

Register	Offset		Description	Initial Value
P_DAC_CH0_Flag	DAC_BA+0x0C		DAC0 FIFO Status Register	0x0000_0006
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:3]	Reserved			0x0
Bit[2]	DFIFO_FLAG	R/W1C	1: FIFO flag set 0: no FIFO flag	0x1
Bit[1]	DFIFO_EMPTY	R/W1C	1: FIFO empty flag set 0: no FIFO empty flag	0x1
Bit[0]	DFIFO_FULL	R/W1C	1: FIFO full flag set 0: no FIFO full flag	0x1

- **P_DAC_CH1_Ctrl (DAC1 Control Register)**

Register	Offset		Description	Initial Value
P_DAC_CH1_Ctrl	DAC_BA+0x40		DAC1 Control Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:8]	Reserved			0x0
Bit[7]	MIX_EN	R/W	1: DAC0 data and DAC1 data mix 0: DAC0 data and DAC1 data no mix	0x0
Bit[6]	TRIG_ALL	R/W	1: DAC0's data trig by manually or DAC_TRIG_SEL 0: DAC0's data trig only by manually	0x0
Bit[5:4]	DAC_TRIG_SEL	R/W	11: Latch DATA to DAC by Timer2 10: Latch DATA to DAC by Timer1 01: Latch DATA to DAC by Timer0 00: Manual Latch	0x0
Bit[3]	ITP_EN	R/W	1: ITP enabled 0: ITP disabled	0x0
Bit[2:1]	Reserved			0x0
Bit[0]	DAC_EN	R/W	1: Function enabled 0: Function disabled	0x0

- **P_DAC_CH1_Data (DAC1 FIFO Data Register)**

Register	Offset		Description	Initial Value
P_DAC_CH1_Data	DAC_BA+0x44		DAC1 FIFO Data Register	0x0000_0000
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:18]	Reserved			0x0
Bit[17:16]	Reserved			0x0
Bit[15:2]	DAC_DATA	W	DAC's DATA	0x0
Bit[1:0]	Reserved			0x0

- **P_DAC_CH1_FIFO (DAC1 FIFO Control Register)**

Register	Offset		Description	Initial Value
P_DAC_CH1_FIFO	DAC_BA+0x48		DAC1 FIFO Control Register	0x0000_0040
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:9]	Reserved			0x0
Bit[8]	DFIFO_RESET	W	1: FIFO reset 0: FIFO no reset (default)	0x0

Bit	Name	R/W	Descriptions	Initial Value
Bit[7]	Reserved			0x0
Bit[6:4]	DFIFO_LEVEL	R/W	When FIFO number is smaller than level, set FIFO_FLAG (default=4)	0x4
Bit[3]	Reserved			0x0
Bit[2]	DFIFO_FLAG_IEN	R/W	1: DFIFO_FLAG_INT enabled 0: DFIFO_FLAG_INT disabled	0x0
Bit[1]	DFIFO_EMPTY_IEN	R/W	1: DFIFO_EMPTY_INT enabled 0: ADC_FIFO_EMPTY_INT disabled	0x0
Bit[0]	DFIFO_FULL_IEN	R/W	1: DFIFO_FULL_INT enabled 0: DFIFO_FULL_INT disabled	0x0

- **P_DAC_CH1_Flag (DAC1 FIFO Status Register)**

Register	Offset		Description	Initial Value
P_DAC_CH1_Flag	DAC_BA+0x4C		DAC1 FIFO Status Register	0x0000_0006
Bit	**Name**	**R/W**	**Descriptions**	**Initial Value**
Bit[31:3]	Reserved			0x0
Bit[2]	DFIFO_FLAG	R/W1C	1: FIFO flag set 0: no FIFO flag	0x1
Bit[1]	DFIFO_EMPTY	R/W1C	1: FIFO empty flag set 0: no FIFO empty flag	0x1
Bit[0]	DFIFO_FULL	R/W1C	1: FIFO full flag set 0: no FIFO full flag	0x0

2.20 Low Voltage Detector

Low Voltage Detector (LVD) builds in precise band-gap reference circuit for accurately detecting VDD level. If P_SMU_PWR_Ctrl[8] (LVD_EN) = 1 and VDD voltage value falls below LVD voltage which is selected by P_SMU_PWR_Ctrl[14;12] (LVD_SEL), as shown in Table4 LVD voltage select, the LVD output P_SMU_PWR_Ctrl[9] (LVD_FLAG) will become high.

LVD_SEL[2:0]	Voltage
111	reserved
110	reserved
101	3.6V
100	3.4V
011	3.2V
010	2.6V
001	2.4V
000	2.2V

Table 4 LVD voltage select

2.21 Configuration Options

Item	Name	Options
1	High Oscillation Source	1. I_HRC 2. E_HXT
2	Low Oscillation Source	1. I_LRC 2. E_LXT
3	HI_CLK Frequency	1. 32MHz (Core LDO @ 3.3V) 2. 24MHz (Core LDO @ 2.7V / 2.8V) 3. 16MHz (Core LDO @ 2.3V / 2.5V) 4. 12MHz (Core LDO @ 2.3V / 2.5V)
4	VDD Voltage	1. 4.5V 2. 3.0V
5	SPI0_VDD Voltage	1. 3.3V 2. 1.8V
6	LVR Voltage	1. 3.0V / 2.9V / 2.8V / 2.7V / 2.6V (HI_CLK @ 32MHz, NX12) 2. 2.9V / 2.8V / 2.7V / 2.6V / 2.5V (HI_CLK @ 32MHz, NX11) 3. 2.6V / 2.5V / 2.4V / 2.3V / 2.2V (HI_CLK @ 24MHz) 4. 2.2V / 2.1V / 2.0V / 1.9V / 1.8V (HI_CLK @ 16MHz) 4. 2.0V / 1.9V / 1.8V / 1.7V / 1.6V (HI_CLK @ 12MHz)

Table 5 User Options

Chapter 3. Development System

The design methodology for the NX1 development system is in two folds: one is to provide the user-friendly environment based on *Q-Code* as used in other Nyquest product series; on the other hand, C-Code based approach is provided in the *NYIDE* environment to give control to developers for more complicated applications.

To facilitate users of Nyquest's existing product series, like NY3 / NY4 / NY5 / NY6 / NY7, the development system also supports Q-FDB_Writer and OTP_Writer.

3.1 Development Environment

The devolvement environment of the NX1 Series is depicted in Figure 18 as follows.

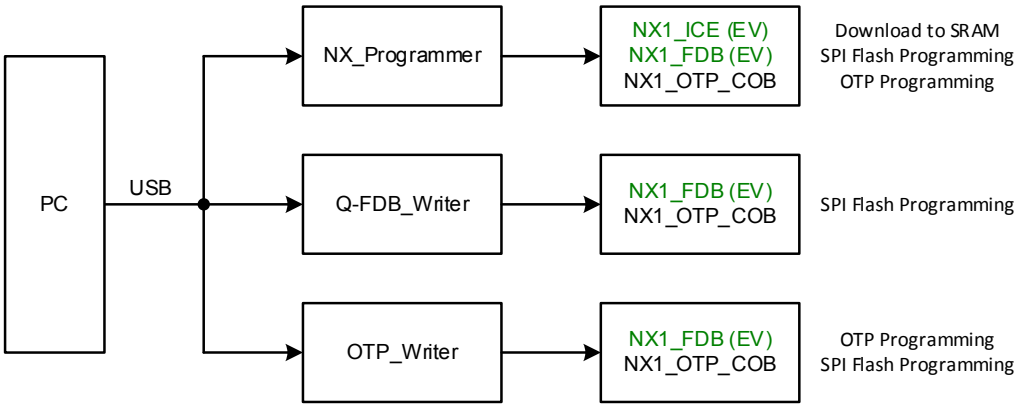


Figure 18 Development Environment of NX1 Series

- Hardware
  1. NX1_ICE: The EV chip (in LQFP-80 package) is on board with add-on peripherals like SD card, SPI flash, UART, while GPIO pins and tact switches are also provided for the convenience of project development. Users can use NX1_ICE to debug via downloading code to SRAM and to program the external SPI flash.
  2. NX1_FDB: The FDB (Flash Demo Board) is used mainly for prototyping and demo purposes. It can also be used like an ICE to help debug the project, since EV chip with Debug Module is inside the FDB.
  3. NX1_OTP_COB: Each OTP production chip has one such OTP_COB for users to verify the actual performance once project developed OK by NX1_FDB. The part numbers are NX11P22AB / NX12P24AB / NX12P34AB / NX12P46AB, etc.
  4. NX_Programmer: Multi-purpose programmer for ICE downloading, SPI flash programming and debugging for NX1_ICE / NX1_FDB, or as a writer for programming OTP at production chips.
  5. Q-FDB_Writer: Programming SPI flash at NX1_FDB / NX1_OTP_COB.
  6. OTP_Writer: Programming OTP for production chips, or SPI flash at NX1_FDB / NX1_OTP_COB.

Note: Both NX1_ICE and NX1_FDB contain the NX1K (EV chip with embedded debug module inside), so that they can execute code downloading, SPI flash programming and ICE debugging via NX_Programmer.

- PC software
  1. *Q-Code_NX1:* Above 1.00 or latest version
  2. *NYIDE:* Above 2.60 or latest version
  3. *Q-Writer:* Above 2.60 or latest version
  4. *Q-MIDI:* Above 2.40 or latest version
  5. *Q-Sound:* Above 1.81 or latest version
  6. *Q-Visio:* Above 1.50 or latest version
  7. *Voice_Encoder:* Above 3.50 or latest version
  8. *SPI_Encoder:* Above 1.00 or latest version

  Note: Please refer to the user manuals of above tools for further information.

- Connection

  Users can connect PC to NX1_ICE / NX1_FDB / NX1_OTP_COB via NX_Programmer, Q-FDB_Writer or OTP_Writer to execute download, SPI flash programming, debugging, or programming OTP. Please refer to Figure 18 Development Environment of NX1 Series.

- Debug flow with EV chip via NX_Programmer
  1. Use either *Q-Code* or *NYIDE*.
  2. Download code to internal SRAM with NX_Programmer connected, then execute program to check the function.
  3. Download code to external SPI flash and choose one of the following actions:
     - Detach NX1_ICE or NX1_FDB from NX_Programmer to operate in standalone mode. Connect with external power, then the EV chip will boot code from SPI flash to internal SRAM to execute the program.
     - Use XIP (Execute in Place) feature to execute program on SPI flash. Be noted that CPU's performance will be degraded due to serial accessing of XIP code @ SPI flash.

3.1.1 *Q-Code* and *NYIDE*

User can edit program or do emulation via *Q-Code_NX1* or *NYIDE* (C Module). Please refer to the user manuals of *Q-Code_NX1* and *NYIDE* for further information.

- UI of *Q-Code_NX1* is shown below

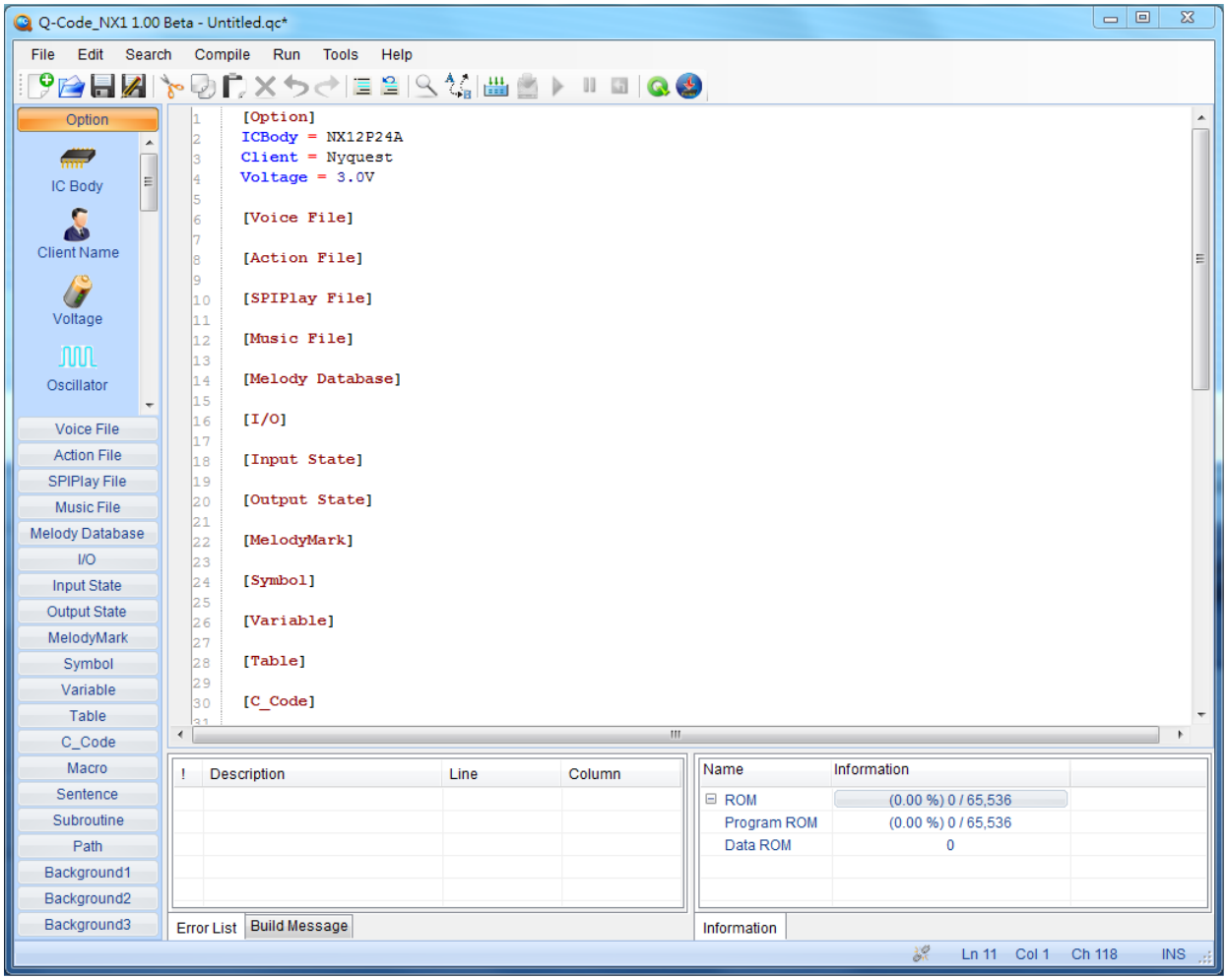


Figure 19 UI of *Q-Code_NX1*

- UI of *NYIDE* (for C-Module) is as follows

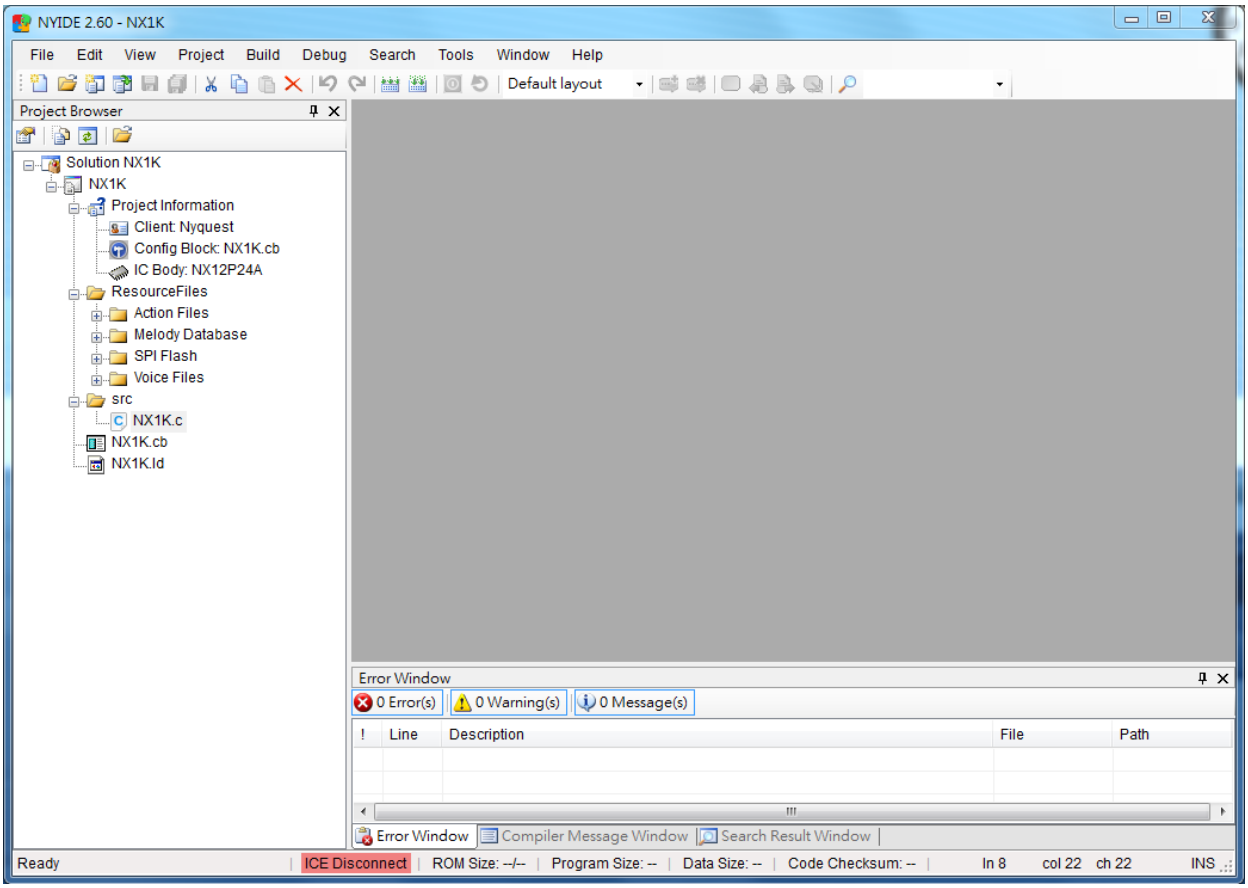
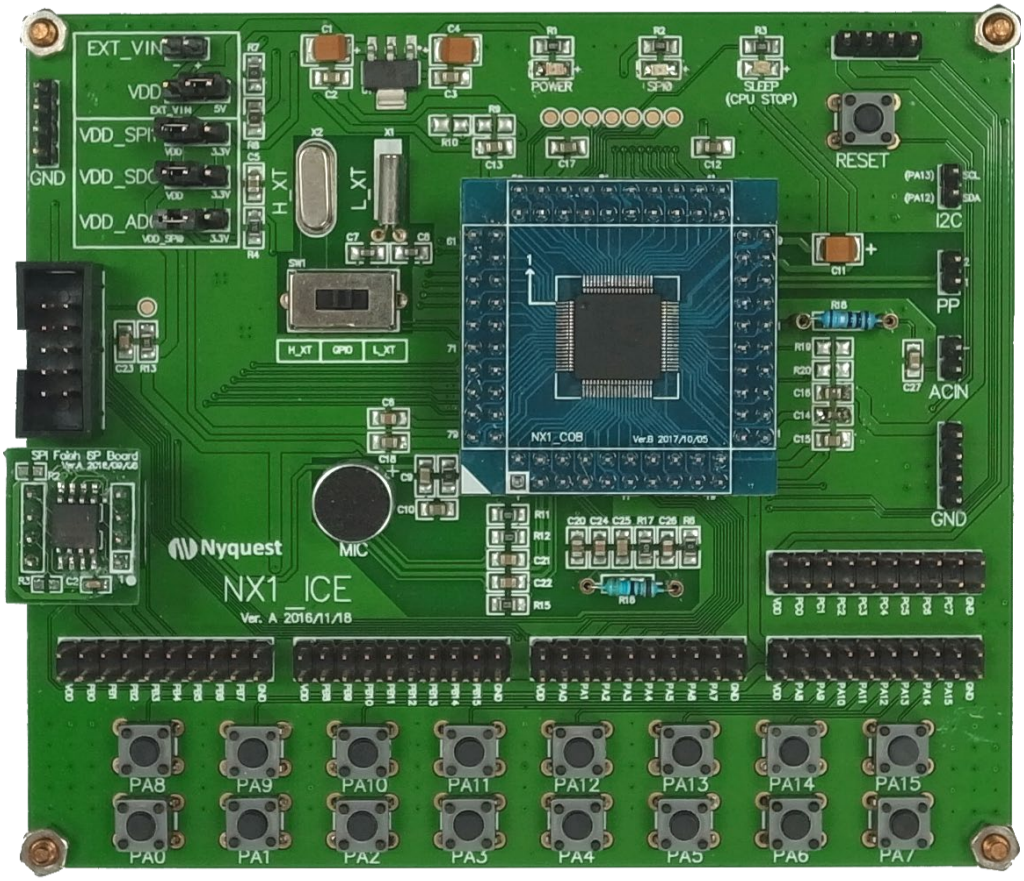


Figure 20 UI of *NYIDE* for NX1 series

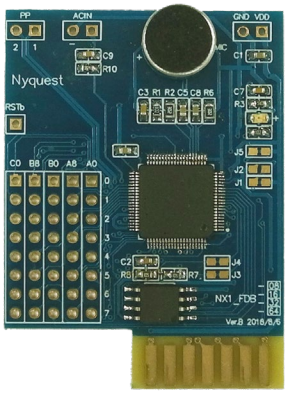
3.1.2 NX_Programmer with ICE / FDB

User can connect NX_Programmer to NX1_ICE via 5x2 pin connector (Female) or NX1_FDB via 7x2 golden finger connector (Female).

- NX1_ICE



- NX1_FDB (Ver.B)



There are 4 kinds of FDB in terms of memory size of the on-board SPI flash: 8Mb / 16Mb / 32Mb / 64Mb.

3.1.3 Q-FDB_Writer

In addition to NX_Programmer, users can also use *Q-Writer* to download data (binary file) to SPI flash on NX1_FDB and NX1_OTP_COB via Q-FDB_Writer.

3.1.4 OTP_Writer

Besides NX_Programmer, users can also program OTP chips of NX1 series via OTP_Writer.