32-Bit MCU
HR8P506

Datasheet

☐**Brief**

☑**Datasheet**

☐**Specification**

SHANGHAI EASTSOFT MICROELECTRONICS CO., LTD.

2017. 12.08

Application Notes

Power On/Off Sequence

Eastsoft MCUs are designed with separate power pins. If a MCU is applied in a system which has multiple power supplies, the MCU should be powered on first or at the same time together with other devices involved in the system. Conversely, the MCU should be powered off after all other devices are powered off. Reversed steps may cause excessive voltage or current on the MCU internal components, which is very likely to cause malfunction and weaken components performances. For details, please refer to the datasheet.

Reset

Eastsoft MCUs provide an internal power-on reset circuit, which could possibly be invalid in different fast/slow power on/off systems. To ensure a proper reset function, the following resets are recommended: external reset, brown-out reset, watchdog reset, etc. The triode reset or the RC reset is recommended when the external reset circuit is used; otherwise, it is suggested to connect the reset pin to power supply with a resistor or to use other protection circuits if necessary. For details, please refer to the datasheet.

Clock

Eastsoft MCUs offer internal and external clock sources. The internal clock frequency may experience a variation due to unstable temperature or voltage, which may affect the accuracy of the clock source. When a ceramic or crystal oscillator circuit is used as an external clock source, it is suggested to enable the oscillator start-up timer. When a RC oscillator circuit is used, it is suggested to take account of the capacitor matching and resistor matching. When an external active oscillator or external clock input is used, consider to input a high-voltage or low-voltage. For details, please refer to the datasheet

Initialization

For different application systems, it is necessary to initialize registers, memories and function modules, especially the multiplexed I/O pins, in order to avoid the unknown status of I/O pins when MCU is powered on.

Pins

Eastsoft MCUs are designed with wide-range input voltage. It is suggested that the input high-level voltage should be higher than V_{IHMIN}, the input low-level voltage should be lower than V_{ILMAX}. To avoid the noise entering into MCU, the input voltage should not be set between V_{IHMIN} and V_{ILMAX}. The unused I/O pins are suggested to be set to input mode, and should be connected to VDD via pull-up resistors or to GND via pull-down resistors. Alternatively, set unused pins to output mode with fixed voltage and leave them floating. How to handle unused pins varies from application to application and it is important to follow the specific application specification and instructions.

ESD Protection

Eastsoft MCUs have industrial ESD standard protection circuit. It is suggested to take proper protection measures depending on application/storage environment to prevent MCUs from static electricity. Special attention should be paid to the humidity of application environment. Do not use the insulators which could cause static electricity. Use anti-static-electricity container/shields or conductive materials to store and transport the MCUs. Ground all the testing tools, measuring tools, including the workbench. Use anti-static-electricity belts or gloves, and do not touch the MCU with fingers directly.

EFT Protection

Eastsoft MCUs have industrial EFT standard protection circuit.. When MCUs are used in PCB systems, the related design requests should be satisfied, including wiring of VDD or GND (i.e. separation of digital/analog power supply,

single-point/multi-point grounding and so on), protection circuits for reset pins, decoupling capacitors between VDD and GND, separate processing of high/low frequency circuits, selection of single-layer/multi-layer board and so on.

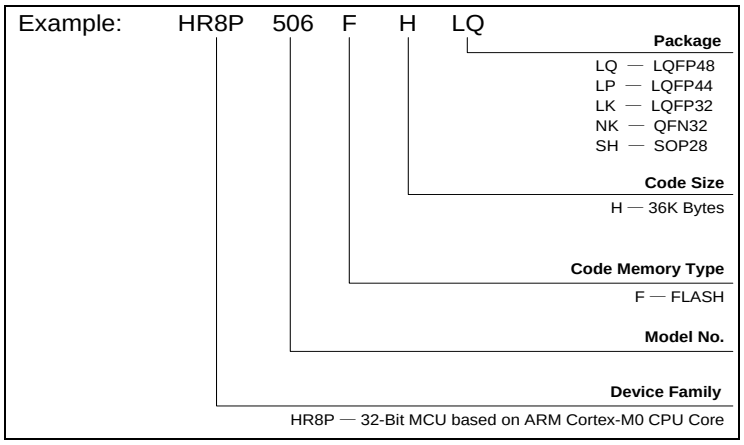
Development Environment

Eastsoft MCUs have a complete software/hardware development environment with protected intellectual property.. When using the development tools, such as assembler, compiler, burner and firmware emulator of Shanghai Eastsoft IC Co, Ltd. or by the appointed third party, please follow the related specifications and instructions.

Note: For any questions arising from product development, please contact us through the sales department or other ways.

Ordering Information

Part Number	FLASH	RAM	I/O	Timer	RTC	UART/ EUART	SPI	I2C	ADC	LCDC/LEDC	LVD	Package
HR8P506FHLQ	36KB	8KB	46	16-bit X 4, 32-bit X 1	1	3	2	1	12bit×16	8COM X 28SEG	√	LQFP48
HR8P506FHLP			42						12bit×12	8COM X 24SEG		LQFP44
HR8P506FHLK			30						12bit×10	8COM X 13SEG		LQFP32
HR8P506FHNK			30						12bit×10	8COM X 13SEG		QFN32
HR8P506FHSH			26						12bit×11	8COM X 10SEG		SOP28



Address: 5th Floor, Building2A, Tianhua Information Science and Technology Park,
No.299 Longcao Road, Shanghai, P.R.CHINA
Zip Code : 200235
E-mail :support@essemi.com
Tel : +86-21-60910333
Fax : +86-21-60914991
Website : http://www.essemi.com/

Revision History

Date	Change
2017-12-8	Initial version, based on HR8P506_Datasheet_C_v1.5

Note: Should you have any questions, please refer to the corresponding version of the HR8P506_Datasheet_C.

Contents

List of Figures

List of Tables

Chapter1 Introduction

1. 1 Overview

The HR8P506 series is a highly integrated MCU product with 32-bit ARM Contex-M0 CPU core. It is designed with rich peripherals, including 16/32-bit Timers/Counters, UART modules with infrared modulation function, 7816 protocol compatible communication interfaces, SPI and I2C modules, RTC module, LCD driver with display-off and twinkling capability, 12-bit ADC and LVD module etc.

- Operating Conditions
  - Voltage range: 2.2V ~ 5.5V
  - Temperature range: -40 ~ 85℃ (industrial)
  - Frequency range: 32KHz~48MHz
  - Operating current: Ivdd = 3.5mA (@internal HRC 16MHz, typical)
  - Standby current: Ivdd = 5uA (@room temperature, typical)
- Packages
  - LQFP48 (46x I/Os)
  - LQFP44 (42x I/Os)
  - LQFP32/QFN32 (30x I/Os)
  - SOP28 (26x I/Os)
- Power Supply
  - System power input VDD, applicable to application systems with operating voltage 5V or 3.3V
  - Low power LVD available to monitor VDD during brown-out and power-on. Capable to generate brown-out interrupt or power-on interrupt.
- Resets
  - Power-on Reset POR
  - Brown-out Reset BOR
  - External resets
- Clocks
  - External crystal oscillator with low-speed 32KHz and high-speed 1~20MHz options, can be configured as system clock source.
  - Internal 16MHz RC oscillator (HRC) with ±2% factory calibration accuracy over full temperature and voltage range, can be configured as system clock source.
  - Internal 32KHz RC oscillator (LRC) clocking WDT, can be configured as system clock source.
  - PLL frequency multiplier with selectable clock sources, up to 48MHz, can be configured as system clock source.

- Core
  - Embedded ARM Cortex-M0 32-bit processor core
  - SWD serial debug interface with 2 watchpoints and 4 breakpoints
  - Two pairs of SWD debug interfaces selected with the DEBUG_S bit of the configuration word
  - Built-in vectored interrupt controller NVIC
  - Wake-up interrupt controller WIC
  - NVIC contains one non-maskable interrupt
  - Built-in system timer SysTick
- Hardware Watchdog
  - Selectable clock sources
  - Capable to wake-up CPU from low power mode
  - Interrupt or reset on overflow
- Memories
  - 36K Bytes Flash Memory
    - In-system programming ISP
    - Two pairs of ISP interfaces selectable, automatically recognized by hardware
    - Supports IAP programming, and part of Flash can be used for data storage
    - Flash programming code encryption
  - 8K Bytes SRAM
    - Bit-banding available in SRAM space
- I/O Ports
  - Up to 46 bidirectional I/O ports
    - Port A(PA0~PA31)
    - Port B(PB0~PB13)
  - 8 external interrupt inputs with configurable triggering. Each I/O port can be used as an external interrupt input source.
  - 1 key interrupt input with configurable triggering. Each I/O port can be used as a key interrupt input source.
- Timers/Counters
  - T16N0: 16-bit timer/counter with a prescaler, supports capture input and PWM output
  - T16N1:16-bit timer/counter with a prescaler , supports capture input and PWM output
  - T16N2: 16-bit timer/counter with a prescaler, supports capture input and PWM output y
  - T16N3: 16-bit timer/counter with a prescaler, supports capture input and PWM output

- ◇ T32N0: 32-bit timer/counter with a prescaler, supports capture input and PWM output
- ◇ RTC: one real time clock

◆ UART

- ◇ Two UART communication interfaces available, UART0 and UART1
- ◇ Full and half duplex
- ◇ Configurable baud rate
- ◇ 8-level transmit and receive buffers
- ◇ 7/8/9-bit data length
- ◇ Odd/even parity check with automatic hardware determined parity bit
- ◇ Idle frame detection
- ◇ Receive framing error, overrun, odd/even parity error flags
- ◇ Transmit and receive interrupts
- ◇ PWM output with programmable duty cycle
- ◇ Receive port with infrared wake-up capability
- ◇ UART ports with configurable input/output mode

◆ EUART

- ◇ Single EUART communication interface available, EUART0
- ◇ Compatible with UART, can be configured as UART
- ◇ Asynchronous half duplex receive/transmit (in 7816 mode)
- ◇ 8-bit data and 1-bit parity (in 7816 mode)
- ◇ Automatic re-transmit and re-receive mode (in 7816 mode)
- ◇ Configurable internal clock output (in 7816 mode)
- ◇ Dual-channel communication available (in 7816 mode)

◆ I2C

- ◇ Single I2C communication interface available, I2C0
- ◇ Master and slave modes
- ◇ Standard I2C bus protocol, up to 400K bits/s
- ◇ 7-bit addressing
- ◇ Data order with MSB first
- ◇ Receive and transmit interrupts
- ◇ SCL/SDA port with push-pull/open drain. Internal weak pull-up or external pull-up resistors must be enabled in open drain mode.
- ◇ SCL with clock stretching

◆ SPI

- ◇ Two SPI communication interfaces available, SPI0 and SPI1

- ◇ Master and slave mode
- ◇ 4 data formats available
- ◇ 4-level receive and transmit buffers
- ◇ Receive and transmit interrupts

◆ ADC

- ◇ 12-bit conversion result and 11-bit effective resolution.
- ◇ 16 analog input channels
- ◇ Multiple reference voltage sources
- ◇ Supports interrupt
- ◇ Auto conversion and compare
- ◇ Timed ADC conversion trigger

◆ LCDC

- ◇ Up to 8 COMs x 28SEGs
- ◇ Clock source options: LRC divided by 4, LOSC divided by 4 and PCLK divided by 4096
- ◇ Grayscale adjustment
- ◇ Twinkling with configurable frequency
- ◇ Two types of LCD driver waveforms
- ◇ Internal bias voltage adjustable

◆ LEDC

- ◇ 1~ 8 pieces of 8-segment common cathode LED
- ◇ Clock source options: LRC divided by 4, LOSC divided by 4 and PCLK divided by 4096

◆ RTC

- ◇ Only reset by POR. Supports program write protection. Immune to system interference.
- ◇ Clock source: external 32.768KHz crystal oscillator for high precision requirement
- ◇ High accuracy digital calibration
- ◇ Two calibration accuracies: ±1.5ppm (or ±0.5ppm) in calibration range ±384ppm (or ±128ppm)
- ◇ Time (seconds, minutes, hours) and calendar (days, weeks, months and years), BCD format.
- ◇ 5 programmable timer interrupts
- ◇ 2 programmable calendar alarms
- ◇ Single configurable clock output
- ◇ Auto leap year recognition till year 2099

◇ 12-hour or 24 –hour format

◇ Low consumption: operating current =0.5μA (typ) @ VDD=5.0V

1. 2 Applications

The MCU is ideal for home appliances, small home appliances, industrial control instruments and other applications.

1. 3 Block Diagram

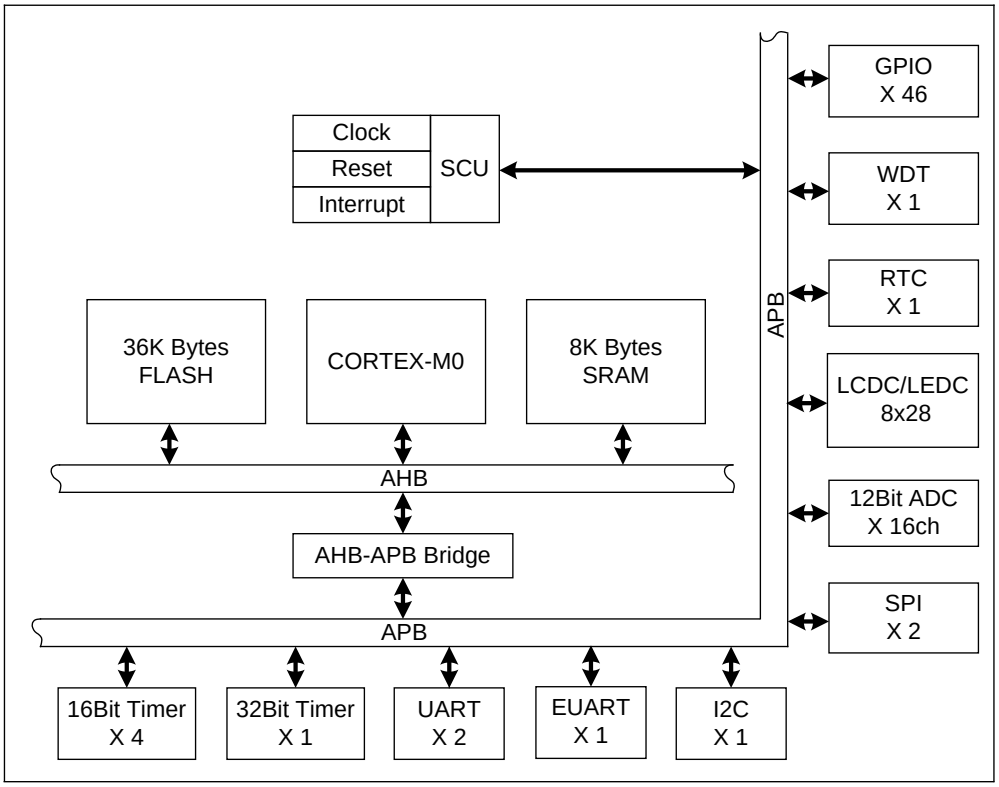


Figure 1-1 HR8P506 Block Diagram

1. 4 Pin Diagrams

1. 4. 1 LQFP48

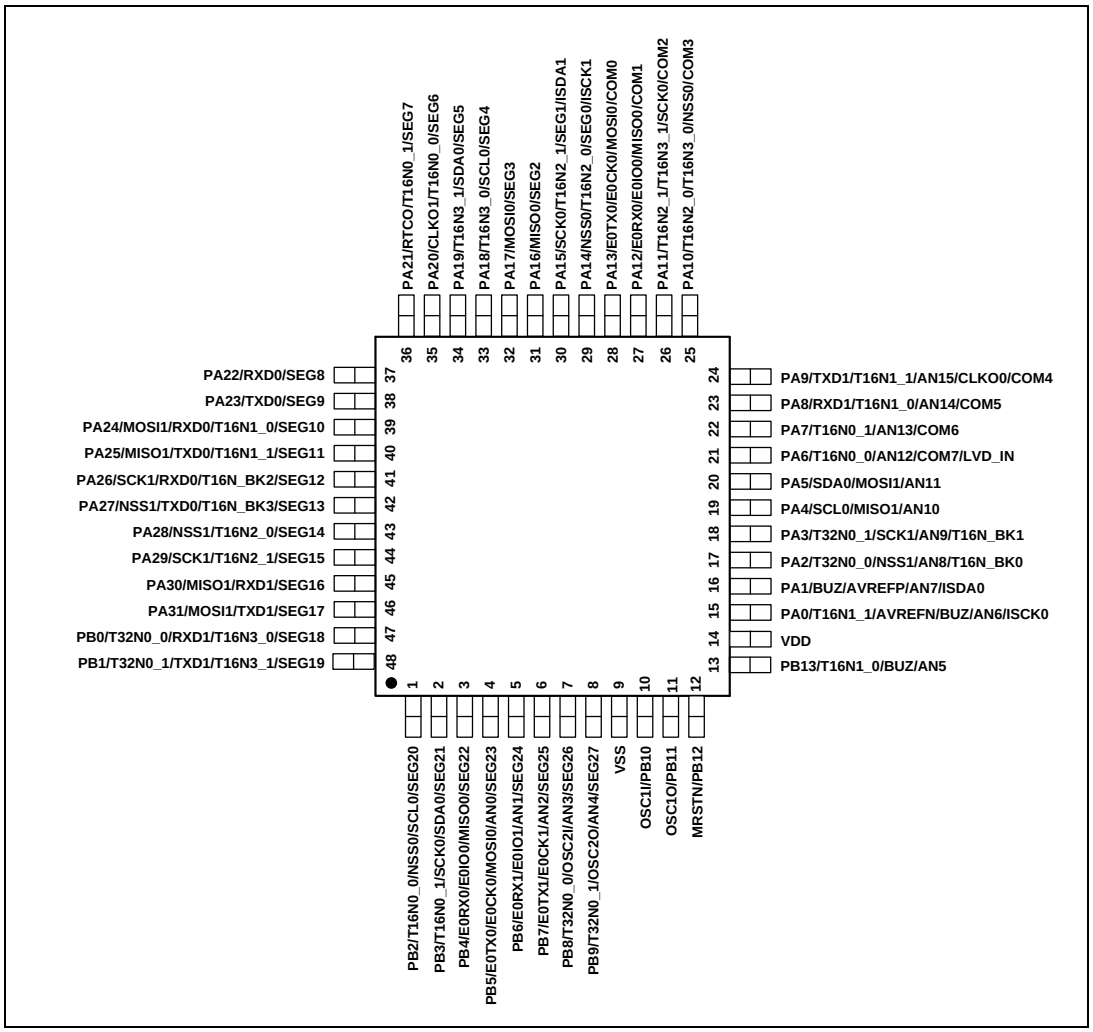


Figure 1-2 LQFP48 Top View

1. 4. 2 LQFP44

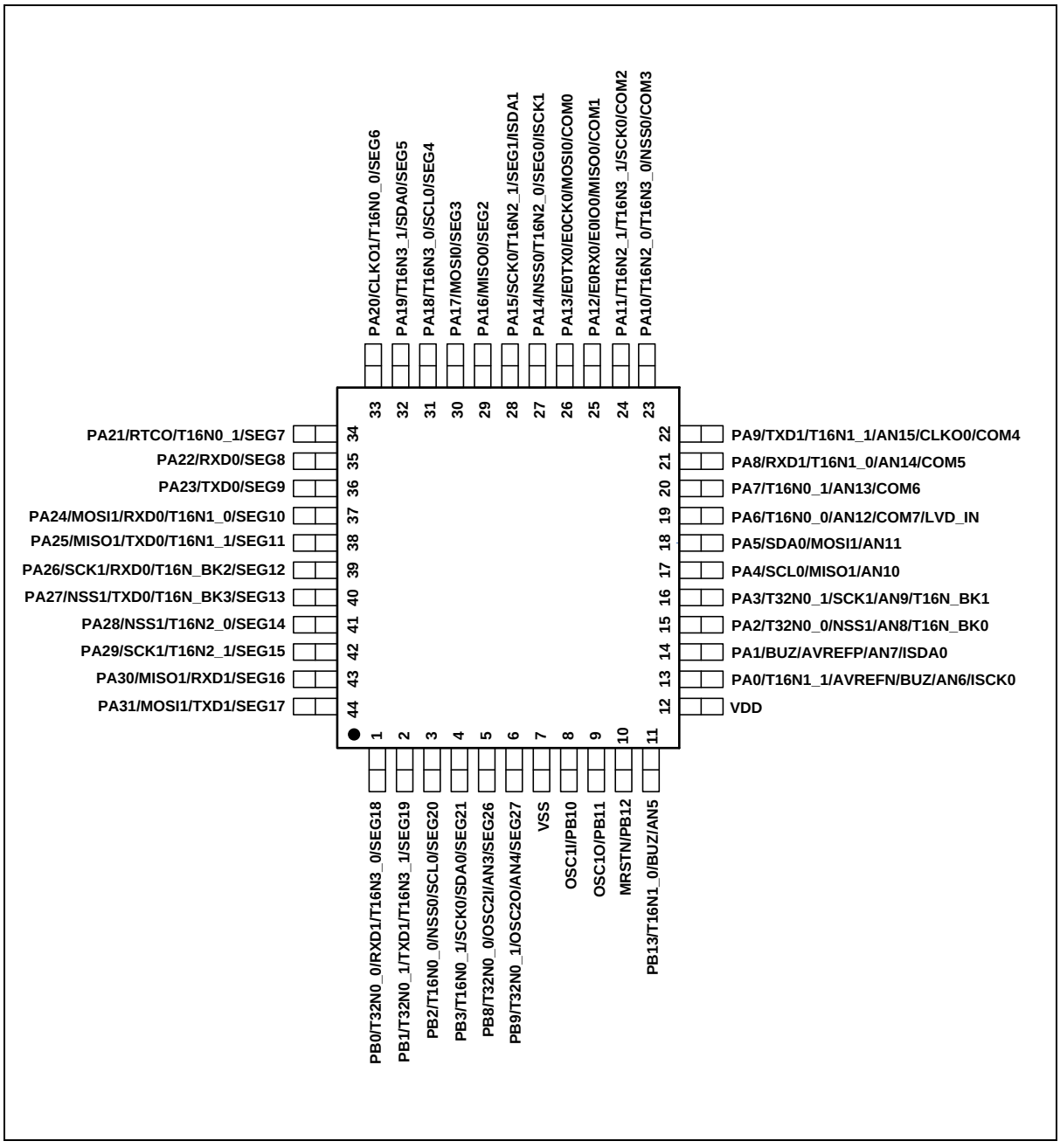


Figure 1-3 LQFP44 Top View

1. 4. 3 LQFP32

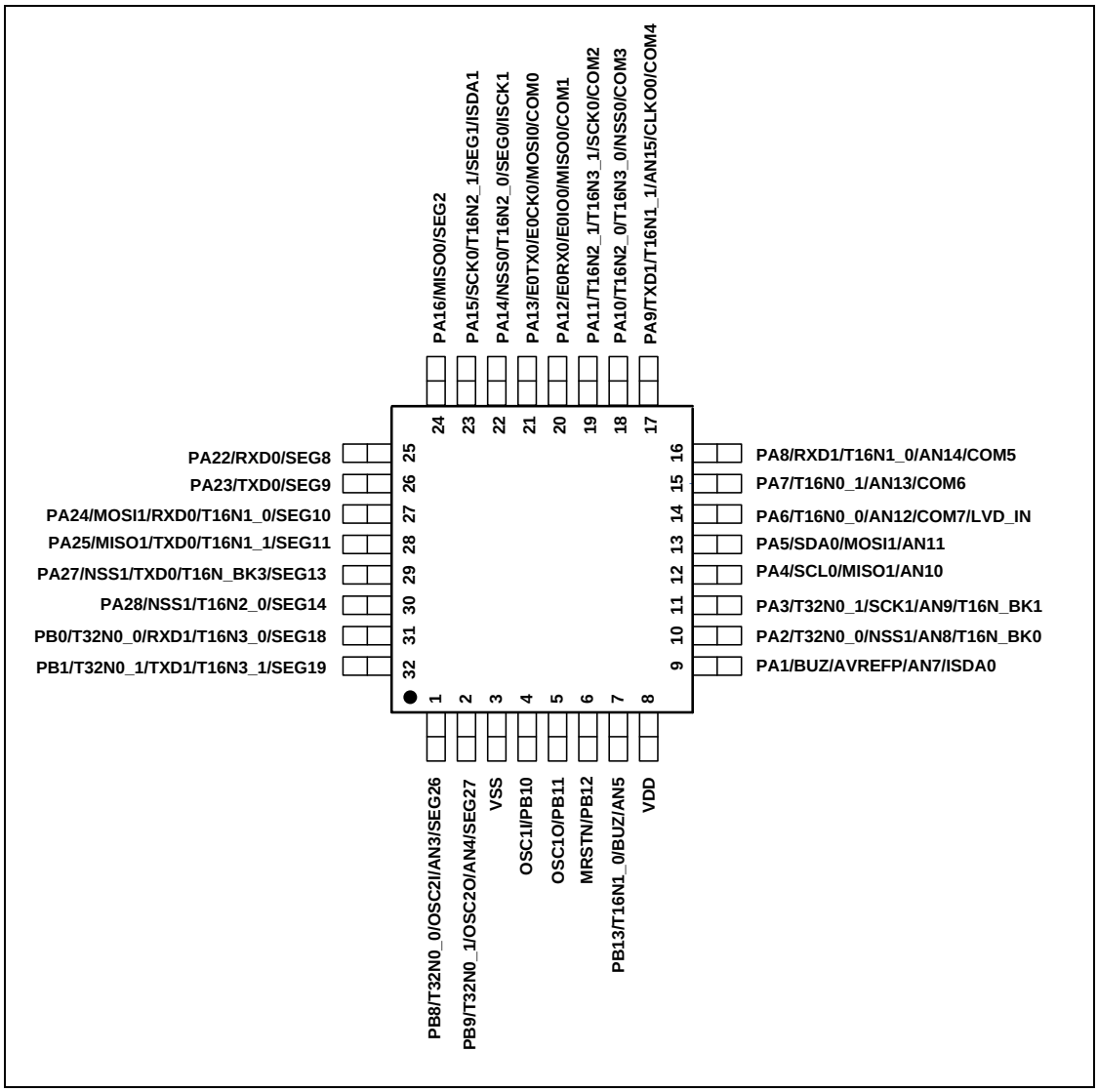


Figure 1-4 LQFP32 Top View

1. 4. 4 QFN32

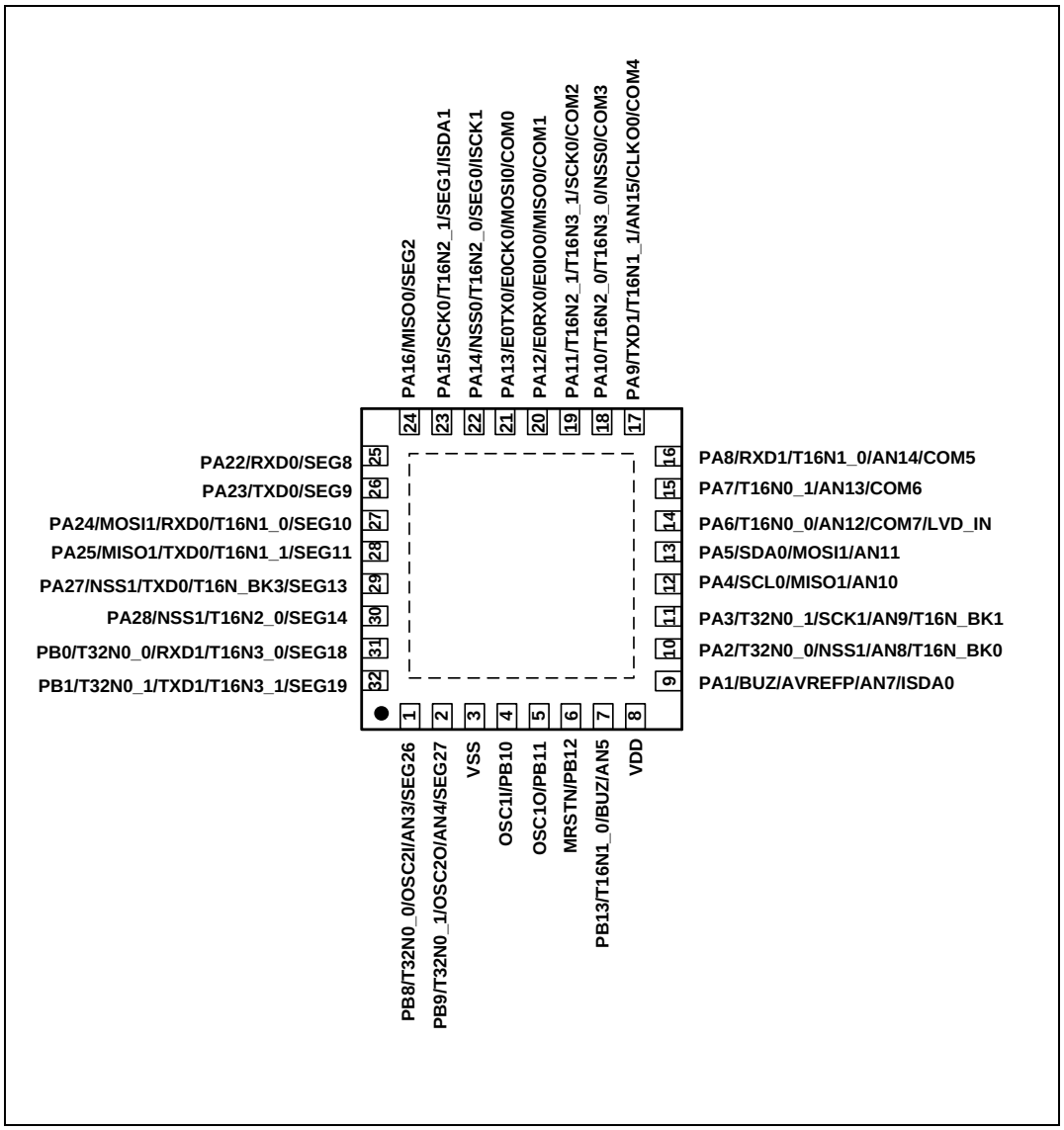


Figure 1-5 QFN32 Top View

Note: For QFN32 packaging, the bottom pad should be soldered to ground. As shown in the figure above, the dotted area should be connected to VSS.

1. 4. 5 SOP28

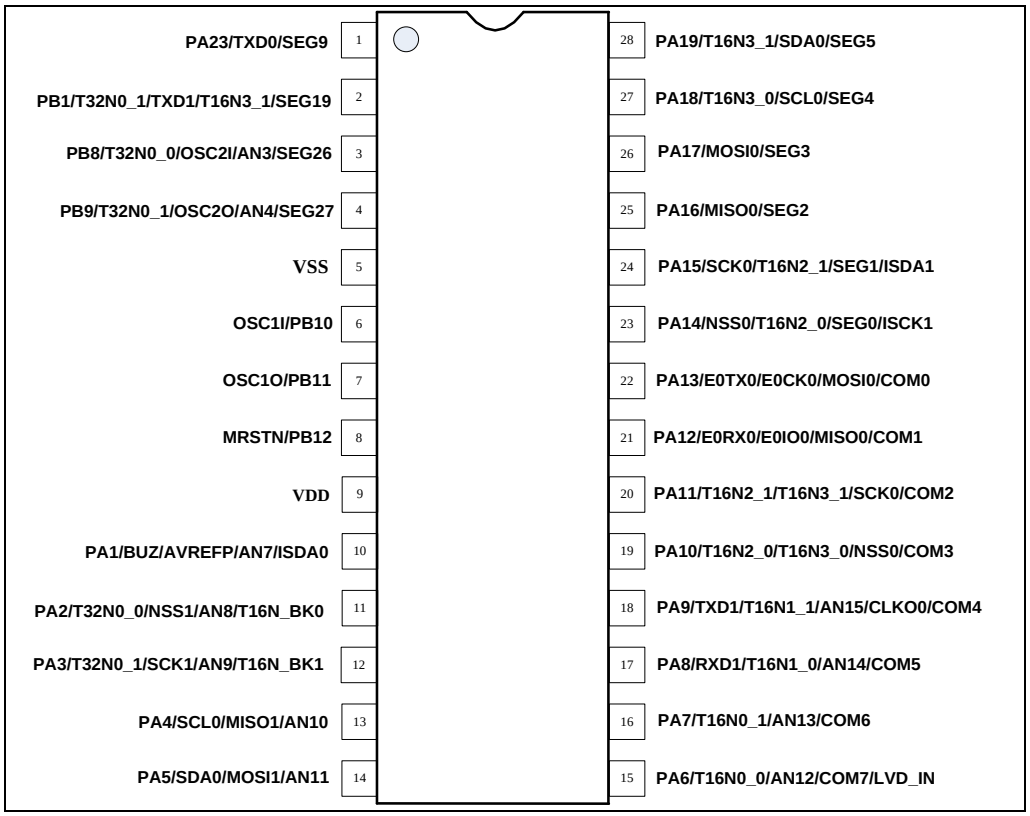


Figure 1-6 SOP28 Top View

Note1: The program/debug interface has a combination of 5 wires: VDD wire, GND wire, MRSTN wire, ISCK wire and ISDA wire. The default function of MRSTN is reset which needs to be programmed/ debugged through the 5-wire interface. When the MRSTN pin is configured as a GPIO, the MRSTN wire must be disconnected during debug.

Note2: It is recommended that the PA6 pin set to output mode and its voltage should not be higher than VDD or lower than VSS, which might cause irregular behavior of the device.

1. 5 Pin Descriptions

1. 5. 1 Pin Descriptions

Pin Name	Input Type	Output Type	A/D	Description
PA0~PA31	CMOS	CMOS	D	General-purpose I/O
PB0~PB13	CMOS	CMOS	D	General-purpose I/O
ISCK0/1	CMOS	—	D	Programming/debug serial clock port
ISDA0/1	CMOS	CMOS	D	Programming/debug serial data port
AIN0~AIN15	—	—	A	ADC analog channels 0~15
TX0~TX1	—	CMOS	D	UART0~UART1 transmit output ports
RX0~RX1	CMOS	—	D	UART0~UART1 receive input ports
E0TX0/1	—	CMOS	D	EUART0 transmit output port

Pin Name	Input Type	Output Type	A/D	Description
E0RX0/1	CMOS	—	D	EUART0 receive input port
E0CK0/1	—	CMOS	D	EUART0 internal clock output port
E0IO0/1	CMOS	CMOS	D	EUART0 data input/output port
SCK0~SCK1	CMOS	CMOS	D	SPI0~SPI1 clock input/output ports
NSS0~NSS1	CMOS	—	D	SPI0~SPI1 chip-select port
MISO0~MISO1	CMOS	CMOS	D	SPI0~SPI1 master input/slave output ports
MOSI0~MOSI1	CMOS	CMOS	D	SPI0~SPI1 master output/slave input ports
SCL0	CMOS	CMOS	D	I2C clock input/output port
SDA0	CMOS	CMOS	D	I2C data input/output port
SEG0~SEG27	—	—	A	LCD Segment ports
COM0~COM7	—	—	A	LCD Common ports
LCD_V1~LCDV4	—	—	A	LCD external bias voltage input port
T16N0_0, T16N0_1 T16N1_0, T16N1_1 T16N2_0, T16N2_1 T16N3_0, T16N3_1	CMOS	CMOS	D	T16N0/T16N1/T16N2/T16N3 external clock input/ capture input/ PWM output port
T32N0_0, T32N0_1	CMOS	CMOS	D	T32N0 external clock input/ capture input/ PWM output port
T16N_BK0 T16N_BK1 T16N_BK2 T16N_BK3	CMOS	—	D	T16N0/T16N1/T16N2/T16N3 break input port (only one active at a time)
LVD_IN	—	—	A	LVD analog input channel
AVREFP	—	—	A	ADC external positive reference voltage
AVREFN	—	—	A	ADC external negative reference voltage
MRSTN	CMOS	—	D	Main reset, active low
OSC1I	—	—	A	External crystal oscillator port 1
OSC1O	—	—	A	
OSC2I	—	—	A	External crystal oscillator port 2
OSC2O	—	—	A	
VDD	—	—	P	Power supply
VSS	—	—	P	Ground

Table 1-1 Pin Descriptions

Notes

1. A = Analog port, D = Digital port and P = Power supply/Ground
2. The T16N0_0 represents three different functions of T16N0, which are T16N0CK0/ T16N0IN0/ T16N0OUT0. Similarly, unless otherwise stated, T16N0_1/ T16N1_0/ T16N1_1/ T16N2_0/ T16N2_1/ T16N3_0/ T16N3_1 and T32N0_0/ T32N0_1 represent their respective functions.

1. 5. 2 Pin Multiplexing

Pin Name (FUN0(D))	FUN1(D)	FUN2(D)	FUN3(D)	FUN4(A)	FUN5(A)
PB0	T32N0_0	RXD1	T16N3_0	—	SEG18
PB1	T32N0_1	TXD1	T16N3_1	—	SEG19
PB2	T16N0_0	NSS0	SCL0	—	SEG20
PB3	T16N0_1	SCK0	SDA0	—	SEG21
PB4	E0RX0/E0IO0	MISO0	—	—	SEG22
PB5	E0TX0/E0CK0	MOSI0	—	AIN0	SEG23
PB6	E0RX1/E0IO1	—	—	AIN1	SEG24
PB7	E0TX1/E0CK1	—	—	AIN2	SEG25
PB8/OSC2I	—	—	T32N0_0	AIN3	SEG26
PB9/OSC2O	—	—	T32N0_1	AIN4	SEG27
PB10/OSC1I	—	—	—	—	—
PB11/ OSC1O	—	—	—	—	—
PB12/MRSTN	—	—	—	—	—
PB13	T16N1_0	BUZ	—	AIN5	—
PA0	T16N1_1	—	BUZ	AVREFN/	—
PA1	BUZ	—	—	AVREFP/	—
PA2	T16N_BK0	T32N0_0	NSS1	AIN8	LCD_V1
PA3	T16N_BK1	T32N0_1	SCK1	AIN9	LCD_V2
PA4	SCL0	—	MISO1	AIN10	LCD_V3
PA5	SDA0	—	MOSI1	AIN11	LCD_V4
PA6	—	T16N0_0	—	AIN12/ LVD_IN	COM7
PA7	—	T16N0_1	—	AIN13	COM6
PA8	RXD1	T16N1_0	—	AIN14	COM5
PA9	TXD1	T16N1_1	CLKO0	AIN15	COM4
PA10	T16N2_0	T16N3_0	NSS0	—	COM3
PA11	T16N2_1	T16N3_1	SCK0	—	COM2
PA12	—	E0RX0/E0IO0	MISO0	—	COM1
PA13	—	E0TX0/E0CK0	MOSI0	—	COM0
PA14	NSS0	T16N2_0	—	—	SEG0
PA15	SCK0	T16N2_1	—	—	SEG1
PA16	MISO0	—	—	—	SEG2
PA17	MOSI0	—	—	—	SEG3
PA18	T16N3_0	SCL0	—	—	SEG4
PA19	T16N3_1	SDA0	—	—	SEG5

Pin Name (FUN0(D))	FUN1(D)	FUN2(D)	FUN3(D)	FUN4(A)	FUN5(A)
PA20	—	CLKO1	T16N0_0	—	SEG6
PA21	—	RTCO	T16N0_1	—	SEG7
PA22	RXD0	—	—	—	SEG8
PA23	TXD0	—	—	—	SEG9
PA24	MOSI1	RXD0	T16N1_0	—	SEG10
PA25	MISO1	TXD0	T16N1_1	—	SEG11
PA26	SCK1	T16N_BK2_B	RXD0	—	SEG12
PA27	NSS1	T16N_BK3_B	TXD0	—	SEG13
PA28	—	NSS1	T16N2_0	—	SEG14
PA29	—	SCK1	T16N2_1	—	SEG15
PA30	—	MISO1	RXD1	—	SEG16
PA31	—	MOSI1	TXD1	—	SEG17

Table 1-2 Pin Multiplexing

Notes

1. FUN0(D)/FUN1(D)/ FUN2(D)/FUN3(D) denotes a digital port and FUN4(A)/ FUN5(A) denotes an analog port.
2. FUN4(A)/ FUN5(A) is an analog port. No need to be configured through the control register.
3. GPIO_PAFUNC/GPIO_PBFUNC.
4. Two pairs of programming/debug interfaces are available. One pair is ISCK0 (PA0) and ISDA0 (PA1), and the other is ISCK1 (PA14) and ISDA1 (PA15).

Chapter2 System Control and Operations

2. 1 System Control and Protection

2. 1. 1 Overview

Accessing the system control register can affect the operating status of the device. In order to prevent the irregular behavior of the device, the system configuration protection register is provided. Before modifying the system control unit, disable the write protection first and enable it again following the completion of modification.

See section 3.4.3 System Control Unit (SCU) for the register list and base address.

2. 1. 2 Special Function Register

System Configuration Protection Register (SCU_PROT)

Offset address: 00_H

Reset values: $00000000\_00000000\_00000000\_00000001_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

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

—	bit31-1	W	When writing 0x55AA6996 to SCU_PROT<31:0>, the PROT bit is 0; When writing other values, the PROT is 1.
PROT	bit0	R/W	**SCU Write Protection bit** 0: Write protection disabled 1: Write protection enabled

Notes

1. Only writing 0x55AA6996 (in word) to the register SCU_PROT can disable write protection. Other than that will enable write protection.
2. The following registers are under protection of the register SCU_PROT: SCU_NMICON, SCU_PWRC, SCU_FAULTFLAG, SCU_FLASHWAIT, SCU_SOFTCFG, SCU_LVDCON, SCU_CCM, SCU_PLLLKCON, SCU_TIMEREN, SCU_TIMERDIS, SCU_SCLKEN0, SCU_SCLKEN1, SCU_PCLKEN, SCU_WAKEUPTIME and SCU_TBLREMAPEN.

2. 2 System Power

2. 2. 1 Block Diagram

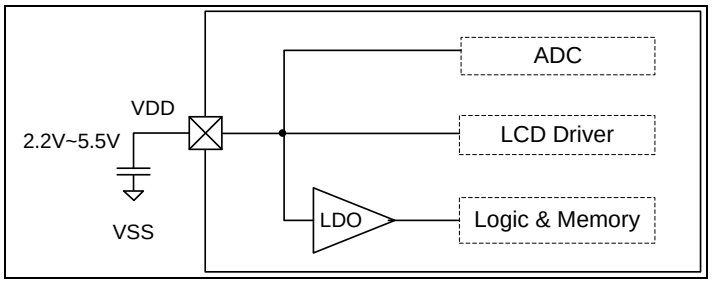


Figure 2-1 System Power Block Diagram

2. 2. 2 Power Supply

The power supply VDD supplies to GPIO ports, ADC module and LCDC driver. The output of the internal LDO supplies to the digital logic, Flash memory and SRAM. VSS is the reference ground.

2. 3 System Resets

2. 3. 1 Overview

- ◇ Power-on Reset POR
- ◇ Brown-out Reset BOR
- ◇ External Reset MRSTN
- ◇ Watchdog Overflow Reset
- ◇ Cortex-M0 Debug Interface Software Reset

2. 3. 2 Block Diagram

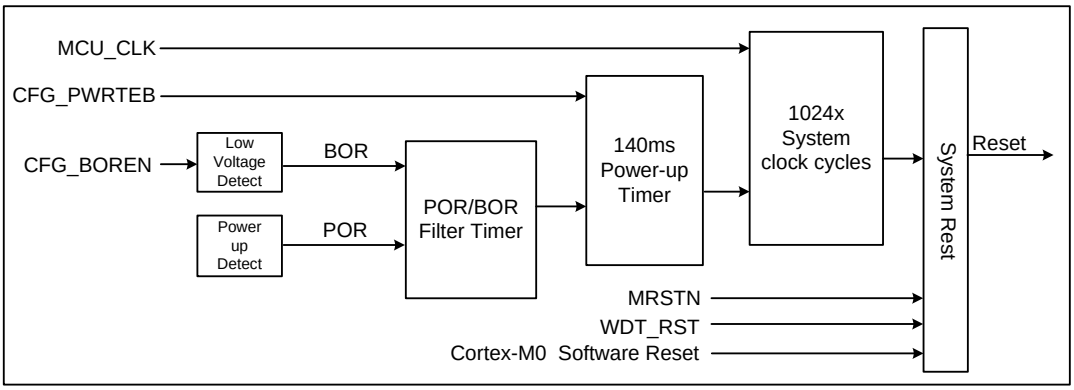


Figure 2-2 System Reset Block Diagram

Notes

1. For 140ms power-on timer, when the MRSTN pin is multiplexed with GPIO function or after the BOR has occurred, the power-up timer remain enabled regardless of the CFG_PWRTEB bit.

2. When the voltage has been stabilized after power-on, if an external reset, WDT overflow reset or software reset occurred, the device would immediately exit the reset state once the reset condition expires, and then it would resume normal operation, regardless of all timers in Figure 2-2.

2. 3. 3 Reset Timing Diagram

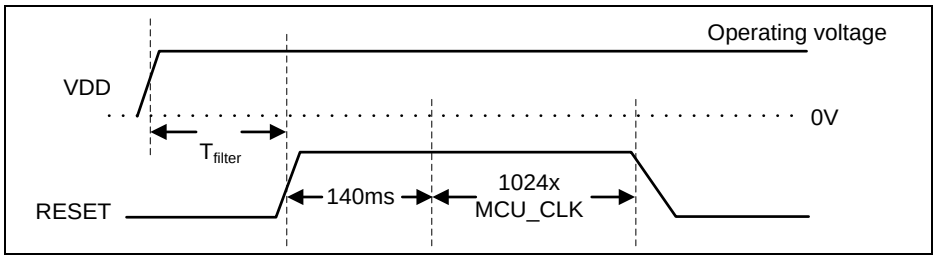


Figure 2-3 Power-on Reset Timing Diagram

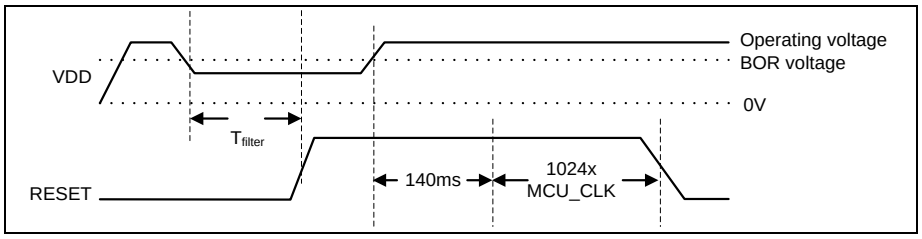


Figure 2-4 BOR Reset Timing Diagram

2. 3. 4 External Reset MRSTN Reference

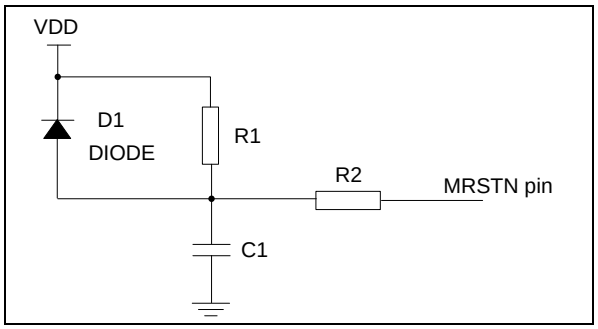


Figure 2-5 MRSTN Reset Reference Circuit 1

Notes

1. Figure above shows the RC reset circuit, where 47KΩ≤R1≤100KΩ, C1= (0.1μF), R2 is a current limiting resistor and 0.1KΩ≤R2≤1KΩ.
2. When the MRSTN pin is used for external reset, there is an internal integrated weak pull-up resistor of 45KΩ, and R1 can be omitted.

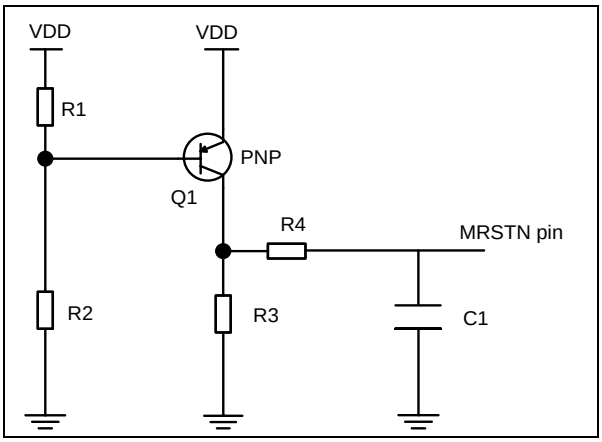


Figure 2-6 MRSTN Reset Reference Circuit 2

Notes

1. Figure above shows the PNP reset circuit, the voltage divided by R1 (2KΩ) and R2 (10KΩ) is used as a base input; the emitter connects to VDD. One end of the collector connects to ground through R3 (20KΩ), the other connects to ground through R4 (1KΩ) and C1 (0.1μF). The ungrounded end of C1 is used as an input to MRSTN pin
2. When the MRSTN pin is used for external reset, there is an internal integrated weak pull-up resistor of 45KΩ, and the R1 can be omitted.

2. 3. 5 Special Function Register

Reset Register (SCU_PWRC)

Offset Address: 08_H

Reset Value: 00000000_00000000_00000000_xxxxxxxx $_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							CFG_RST	POR_LOST	SOFT_RSTF	MRSTF	WDTRSTF	BORF	PORRSTF	PORRCF	PORF

—	bit31-9	—	—
CFG_RST	bit8	R/W	**Configuration word read flag bit (don't care, for internal test only)** 0:Configuration word not read 1:Configuration word has been read
POR_LOST	bit7	R/W	**POR lost flag bit (for internal test only, see note 2)** 0:No POR lost 1:POR lost
SOFT_RSTF	bit6	R/W	**Software reset flag bit** 0: No software reset

			1: Software reset occurred
MRSTF	bit5	R/W	**MRSTN reset flag bit** 0: No MRSTN reset 1: MRSTN reset occurred
WDTRSTF	bit4	R/W	**WDT reset flag bit** 0: No WDT reset 1: WDT reset occurred
BORF	bit3	R/W	**BOR flag bit** 0: No BOR 1: BOR occurred
PORRSTF	bit2	R/W	**PORRST flag bit (for internal test only, see note 2)** 0: No PORRST 1: PORRST occurred
PORRCF	bit1	R/W	**PORRC reset flag bit** 0: No PORRC reset 1: PORRC reset occurred
PORF	bit0	R/W	**POR flag bit (don't care, for internal test only)** 0: No POR 1: POR occurred

Notes

1. The PORRCF may become invalid due to the abnormal power supply.
2. The user must first clear the PORRSTF bit after power-on, or else the bit 3, 4, 5 and 6 would not be set even if the corresponding reset event has occurred.
3. Prior to writing to the SCU_PWRC register, configure the SCU_PROT register to disable the write protection.

2. 4 Low Voltage Detection LVD

2. 4. 1 Overview

The embedded low voltage detection LVD module monitors the VDD and the LVD_IN analog channel voltage. When the LVD_IN analog channel is monitored, the voltage threshold is 1.2V. The trigger can take place during brown-out and power-on. The LVD interrupt flag is set after triggering. An LVD interrupt request is generated when the LVD interrupt is enabled. During normal and deep sleep, an LVD interrupt can wake up the device.

2. 4. 2 Special Function Register

Low Voltage Detection Control Register (SCU_LVDCON)

Offset address:28_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LVDO	Reserved		IFS<2:0>			IE	IF	VS<3:0>				Reserved		FLTEN	EN

—	bit31-16	—	—
LVDO	bit15	R	**LVD output status bit** 0: The detected voltage > voltage threshold 1: The detected voltage < voltage threshold
—	bit14-13	—	—
IFS<2:0>	bit12-10	R/W	**These bits select the way in which the LVD interrupt flag is set** 000:LVDO rising edge 001:LVDO falling edge 010:LVDO high level 011:LVDO low level 1xx:LVDO rising edge and falling edge
IE	bit9	R/W	**LVD interrupt enable bit** 0: Disable 1: Enable
IF	bit8	R/W	**LVD interrupt flag bit** 0: No LVD trigger event 1:LVD trigger event occurred In edge-triggered mode, clear the flag by writing 1. In level-triggered mode, the flag is read-only and it is automatically cleared when the triggering level disappears.

VS<3:0>	bit7-4	R/W	**LVD voltage select bits** 0000:2.0V 0001:2.1V 0010:2.2V 0011:2.4V 0100:2.6V 0101:2.8V 0110:3.0V 0111:3.6V 1000:4.0V 1001:4.6V 1010:2.3V 1011, 1100, 1101, 1110: Reserved 1111: Detects LVD_IN, voltage threshold is 1.2V
—	bit3-2	—	**—**
FLTEN	bit1	R/W	**LVD filter enable bit** 0: Disable 1: Enable
EN	bit0	R/W	**LVD enable bit** 0: Disable 1: Enable

Notes

1. Prior to writing to the SCU_LVDCON register, configure the SCU_PROT register to disable write protection.
2. Based on the actual power supply and environment, as well as the application system requirement, determine whether to enable the filter FLTEN. When the FLTEN is enabled, it will filter out the power supply jitter but it also reduces the sensitivity of the LVD circuit to the power supply fluctuation.

2. 5 Low Power Mode

2. 5. 1 Overview

When configuring the peripheral clock control register SCU_PCLKEN, individually disable the clock of each peripheral module to minimize the power consumption

The device can enter sleep mode through WFI instruction. The SLEEPDEEP bit of the SCB_SCR register selects the normal sleep mode or deep sleep mode.

After entering sleep mode, all I/Os maintain the status they had before entering sleep. To reduce the power consumption, all I/Os should stay high or low. Meanwhile, keep the input floating pins high/low through pull-up/ pull-down to avoid leakage current caused by the floating pins.

After entering sleep mode, the operating status of clocks is tabulated as below.

Clock source	Normal Sleep	Deep Sleep
XTAL	Enabled (if XTAL_EN =1)	Enabled (if XTAL_EN=1 and MOSC_EN =1)
HRC	Enabled (if HRC_EN =1)	Enabled (if HRC_EN =1 and MOSC_EN =1)
LRC	Enabled	Enabled

Table 2-1 Operating Status of Clocks in Low Power Mode

2. 5. 2 Normal Sleep Mode

In normal sleep mode, the core clock stops and the instruction execution is aborted. Resets or interrupts can wake up the device from normal sleep mode.

Sleep mode is entered by the following steps:

1) Set the SLEEPDEEP bit =0

2) Execute the wait-for-interrupt WFI instruction to enter normal sleep mode.

In normal sleep mode, the peripherals continue running and can produce interrupts for the core processor to resume operation. No accessing to the memory, related controllers and internal bus in normal sleep mode.

In normal sleep mode, the status of the core processor and the content of the registers, peripheral registers and internal SRAM are maintained, as well as the logic level on the ports.

2. 5. 3 Deep Sleep Mode

In Deep Sleep mode, the core clock stops and the instruction execution is aborted. Resets and interrupts can wake up the device from Deep Sleep mode.

Deep sleep mode is entered by the following steps:

1) Set the SLEEPDEEP bit =1

2) Execute the wait-for-interrupt WFI instruction to enter deep sleep mode.

In deep sleep mode, the peripheral clock PCLK stops, consequently the peripheral modules clocked from the PCLK will stop running. Other peripheral module clocked from the internal low-speed clock LRC or external XTAL will continue operating. No accessing to the memory, related controllers and internal bus in deep sleep mode.

In deep sleep mode, the status of the core processor and the content of the registers, peripheral registers and internal SRAM are maintained, as well as the logic level on the ports.

Prior to entering deep sleep mode, determine whether to disable the XTAL, PLL, HRC and clock filter CLKFTL via the clock control bit MOSC_EN of the system wake-up time control register SCU_WAKEUPTIME. If disable the clock mode (MOSC_EN=0), it can reduce the consumption in deep sleep mode, however, it also increases the time spent on wake-up.

2. 5. 4 Wake-up

The following events can wake up the device from the sleep mode, and then execute the next instruction or the interrupt service routine (ISR). If the device wakes up by an interrupt and the interrupt is enabled, then it will execute the ISR once waken.

◇ Wake-up from normal sleep

- All interrupts
- Device resets

◇ Wake-up from deep sleep

- External interrupt port PINT
- External interrupt key KINT
- RTC interrupt
- LVD interrupt
- WDT interrupt (clocked from LRC)
- ADC interrupt (clocked from LRC)
- Device resets

2. 5. 5 Wake-up Time

Wake-up time from deep sleep mode consists of the system clock start-up time and the internal LDO voltage settling time. In deep sleep mode, the specific wake-up time depends on the system clock source and if the system clock is enabled or not.

The start-up time of the internal HRC clock is around 80us and the external clock XTAL 16MHz is about 5ms. For the XTAL 32KHz clock, the start-up time is around 1.2 seconds.

The internal LDO voltage settling time is configured by software as Tpclk*WAKEUPTIME (where Tpclk is the system clock period, WAKEUPTIME is the wake up time control bits WAKEUPTIME<11:0>). It is recommended that the LDO settling time > 40us, otherwise the device might exhibit irregular behavior after wake-up.

Example: if the internal HRC is used as the system clock
When MOSC_EN=0, the minimum wake-up time from deep sleep is 80us+40us=120us.

When MOSC_EN=0, the minimum wake-up time from deep sleep is 40us.

For normal sleep mode, the wake-up time is independent from MOSC_EN and WAKEUPTIME. As long as a wake-up event occurs, the device will immediately wake up from normal sleep and start running programs.

2. 5. 6 Flash Memory Wait Function

Frequent access to the Flash memory will increase the power consumption. Reducing the system clock frequency can make the access to the Flash memory less frequent, thereby lowering down the power consumption. However, it will also reduce the operating speed of the peripherals.

The Flash memory is designed with additional wait function, lowering the frequency to fetch the instruction or data without reducing the system clock frequency. Flash memory supports the accessing frequency up to 24MHz. If the system clock frequency is beyond 24MHz, it also requires the Flash memory to set the wait time. Otherwise, it will result in Flash access error.

Configure the ACCT<3:0> bits of the SCU_FLASHWAIT register to set the desired wait-time on accessing Flash.

If ACCT<3:0>=0, the system clock frequency can be up to 24MHz.

If ACCT<3:0>=1, the system clock frequency can be up to 40MHz.

If ACCT<3:0>=2~F, the system clock frequency can be up to 48MHz.

According to the above, if the 32MHz derived from PLL frequency multiplier is selected as system clock, the ACCT<3:0> cannot be 0, and it requires a minimum of 2 clock cycles to successfully access the Flash memory. If the 48MHz derived from PLL frequency multiplier is selected as system clock, the ACCT<3:0> cannot be 0 or 1, and it requires a minimum of 3 clock cycles to successfully access the Flash memory. Therefore, prior to switching the system clock to 32MHz or 48MHz, the ACCT<3:0> needs to be configured first to select a proper access time. Otherwise, it may result in error in executing the instruction.

2. 5. 7 Special Function Register

Flash Access Wait Register (SCU_FLASHWAIT)

Offset address:20_H

Reset value:00000000_00000000_00000000_00000010$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												ACCT<3:0>			

—	bit31-4	—	—
ACCT<3:0>	bit3-0	R/W	**Flash access wait-time select bits** 0:1xTCLK to complete Flash read 1:2xTCLK 2:3xTCLK ... F:16xTCLK

Notes

1. Configure the SCU_PROT register to disable write protection prior to writing to the SCU_FLASHWAIT register.
2. TCLK = System clock cycle

2. 6 System Clock

2. 6. 1 Overview

Four clock source options are available.

- ◇ External clock source with two mode options. High-speed mode HS/XT (HOSC, 1~20MHz) and low-speed mode LP (LOSC, 32KHz). 2 sets of pins available to connect external crystal oscillator
- ◇ Internal high frequency RC clock source HRC: 16MHz
- ◇ Internal low frequency RC clock source LRC: about 32KHz
- ◇ Internal integrated phase locked loop PLL: input frequency 32KHz or 4MHz (4MHz is the automatically obtained by HRC dividing by 4); output frequency 32MHz or 48MHz
- ◇ System clock supports frequency division 1~128.
- ◇ 2 sets of I/O ports available to output system clock frequency
- ◇ External clock source off detection. Automatically switch to LRC clock and generate an interrupt if off is detected.
- ◇ PLL lost lock detection. Automatically switch to the clock source before PLL was used and generate an interrupt.
- ◇ Supports system clock filtering to improve the stability of system operation.

The following content describes how to select each clock source as system clock.

1) Selecting the external XTAL as system clock: program the configuration word to select the pins to connect the crystal and select the HS/XT mode or LP mode. Set CLK_SEL=10 of the SCU_SCLKEN0 register, set the XTAL_EN =1 of the SCU_SCLKEN register, and clear the PLL_EN bit of the SCU_SCLKEN1 register.

2) Selecting the internal 16MHz HRC as system clock: program the configuration word to select the internal 16MHz clock. Set CLK_SEL=00 of the SCU_SCLKEN0 register, and clear the PLL_EN bit of the SCU_SCLKEN1.

3) Selecting the internal 32KHz LRC as system clock: set CLK_SEL=01 of the SCU_SCLKEN0 register, and clear the PLL_EN bit of the SCU_SCLKEN1.

4) Selecting the internal PLL frequency multiplier as system clock: set CLK_SEL=00 of the SCU_SCLKEN register, configure the PLL_REF_SEL bits to select the PLL input clock source, configure the PLL_48M_SEL bits to select the PLL output clock frequency, and set PLL_EN=1.

2. 6. 2 Block Diagram

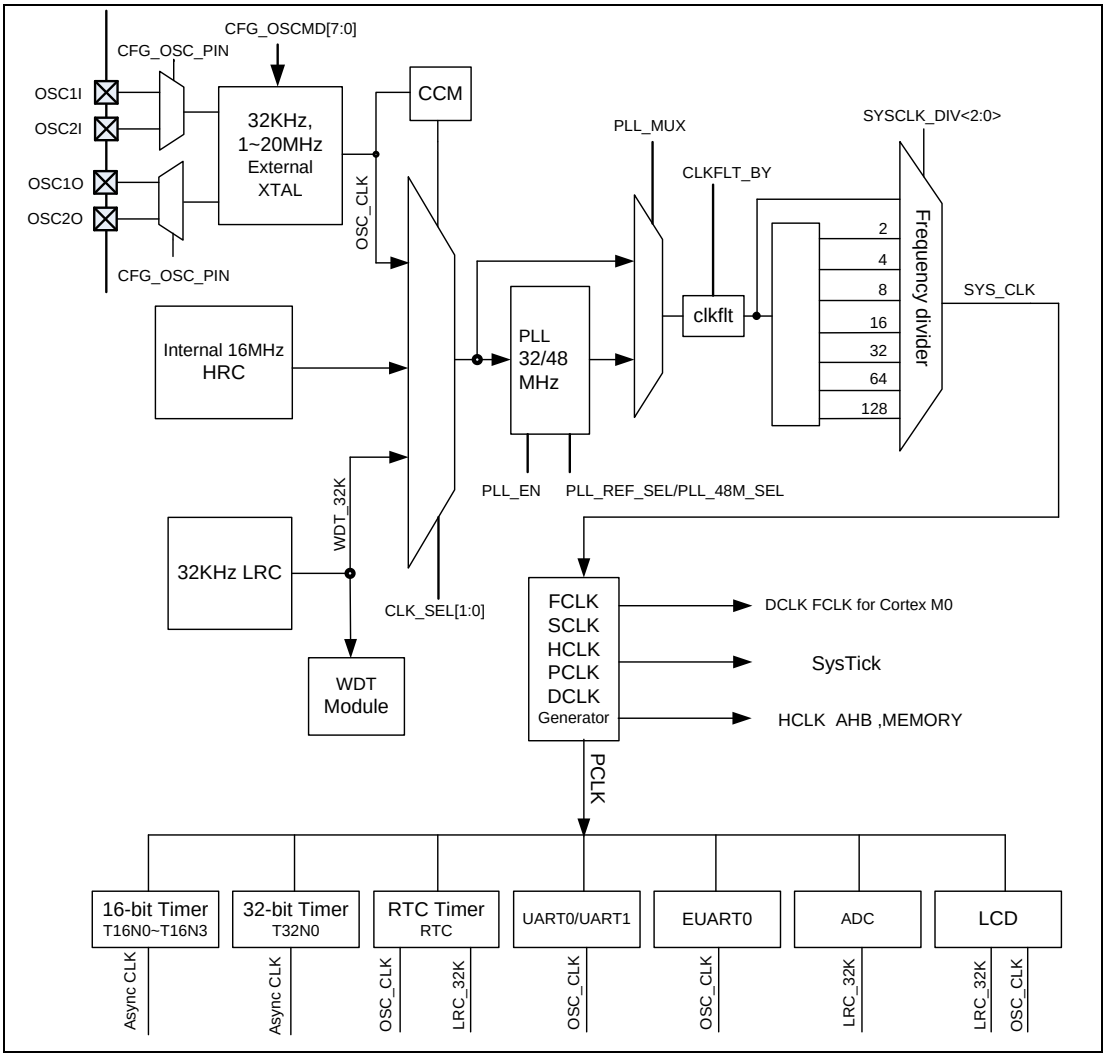


Figure 2-7 System Clock Block Diagram

2. 6. 3 Clock Source

2. 6. 3. 1 External Clock XTAL

PB8 and PB9 or PB10 and PB11 can be selected by the configuration word, to connect the external oscillator. Once the external oscillator module is enabled (XTAL_EN =1), the corresponding I/O ports are used as analog ports, and the digital input/output function is disabled.

Two modes are available for the external clock source, the high-speed mode HS/XT (or called HOSC, 1~20MHz) and low-speed mode LP (or called LOSC, 32KHz), programmed by the configuration word. A 32.768KHz crystal oscillator is recommended for the low-speed LP mode, a 5~20MHz crystal for the HS mode, and the 1~4MHz crystal for the XT mode.

When using the external oscillator, matching capacitors are required to connect externally as below.

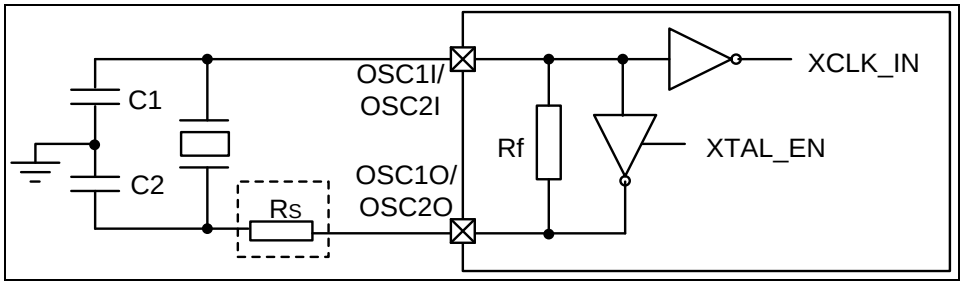


Figure 2-8 XTAL Circuit

Note

1. Rs is optional.
2. C1 and C2 are matching capacitors and their capacitances are determined depending on the crystal used. For reference, the capacitance range is 10~20pF. 15pf is recommended if 1~20MHz crystal is used and 12pf is recommended if the 32.768KHz is used.

The internal high-speed clock HRC is the default system clock after power-on, and the external clock XTAL can be switched to by software. See the related example for details. If MOSC_EN=0 and the device enters deep sleep, the XTAL will be automatically off and it will be automatically on after wake-up. If MOSC_EN=1 and the device enters deep sleep, the XTAL will remain active.

If the XTAL is used as system clock, during normal operation, disabling the XTAL (XTAL_EN=0) is not recommended, because the system clock will automatically be switched to the internal LRC.

The XTAL_LP bit can be configured to select the power consumption mode; however, this is only valid when the XTAL is in low speed LP mode. After the external oscillator has been stabilized (XTAL_RDY=1), the user can configure the XTAL for low power mode to lower down the consumption. When the external oscillator is in HS mode, it will stay in the high power consumption mode and cannot be configured to low power mode.

2. 6. 3. 2 Internal High Speed Clock HRC

The internal high-speed clock 16MHZ HRC has the frequency accuracy ±2% over full temperature range. The HRC is the system clock by default after power-on and it can be disabled by the HRC_EN bit.

When the HRC is operating as system clock, it is not recommended to disable it (HRC_EN=0) or else the system clock will automatically be switched to internal LRC.

When HRC_EN=1, if MOSC_EN=0 and the device enters deep sleep, the HRC clock will automatically be disabled and will automatically be enabled after wake-up. If MOSC_EN=1 and the device enters deep sleep, the HRC will not be disabled.

2. 6. 3. 3 Internal Low Speed Clock LRC

The device has an internal low speed 32KHz clock LRC, which keeps operating all the

time and cannot be disabled. The frequency accuracy of the LRC clock is ±40% over full temperature. The main system, WDT, LCDC, LEDC and RTC module can be clocked by the internal LRC. For those module requiring high frequency accuracy, the LRC is not recommended.

2. 6. 3. 4 Phase Locked Loop PLL

The input clock of PLL can be the XTAL (32.768KHz, 4MHz, 8MHz and16MHz), HRC (16MHz), and LRC (32KHz).

During PLL operation, the PLL_REF_SEL<2:0> selects the input clock source; meanwhile, the selected clock source (HRC, LRC or XTAL) must be configured properly. The PLL_48M_SEL bit selects the frequency multiplication factor to output the desired frequency. See the detailed description below:

When PLL_REF_SEL<2:0>=000/001, it is disabled.

When PLL_REF_SEL<2:0> =010, the HRC must be configured to 16MHz via the configuration word, and the input clock source of the PLL is the 16MHz HRC divided by 4. When PLL_48M_SEL=0, the frequency multiplication factor is 8, and the output frequency of PLL is 32MHz. When PLL_48M_SEL=1, the frequency multiplication factor is 12, and the output frequency of PLL is 48MHz.

When PLL_REF_SEL<2:0> =011, the input clock source of the PLL is the 32KHz LRC. When PLL_48M_SEL=0, the frequency multiplication factor is 1024, and the output frequency of PLL is 32.768MHz. When PLL_48M_SEL=1, the frequency multiplication factor is 1536, and the output frequency of PLL is 49.152MHz.

When PLL_REF_SEL<2:0> =100, the XTAL must be configured as XT mode and externally connect to a 4MHz oscillator, and the input clock source of the PLL is the 4MHz XTAL. When PLL_48M_SEL=0, the frequency multiplication factor is 8, and the output frequency of PLL is 32MHz. When PLL_48M_SEL=1, the frequency multiplication factor is 12, and the output frequency of PLL is 48MHz.

When PLL_REF_SEL<2:0> =101, the XTAL must be configured to HS mode and externally connect to an 8MHz oscillator, and the input clock source of the PLL is the 8MHz XTAL divided by 2. When PLL_48M_SEL=0, the frequency multiplication factor is 8, and the output frequency of PLL is 32MHz. When PLL_48M_SEL=1, the frequency multiplication factor is 12, and the output frequency of PLL is 48MHz.

When PLL_REF_SEL<2:0> =110, the XTAL must be configured to HS mode and externally connect to a 16MHz oscillator, and the input clock source of the PLL is the 16MHz XTAL divided by 4. When PLL_48M_SEL=0, the frequency multiplication factor is 8, and the output frequency of PLL is 32MHz. When PLL_48M_SEL=1, the frequency multiplication factor is 12, and the output frequency of PLL is 48MHz.

When PLL_REF_SEL<2:0> =111, the XTAL must be configured to LP mode and externally connect to a 32.768KHz oscillator, and the input clock source of the PLL is the 32KHz XTAL. When PLL_48M_SEL=0, the frequency multiplication factor is 1024,

and the output frequency of PLL is 32.768MHz. When PLL_48M_SEL=1, the frequency multiplication factor is 1536, and the output frequency of PLL is 49.152MHz.

When operating with the PLL module, the PLL module must be enabled after the input clock source has been stabilized. See PLL example for details.

If PLL_EN =1, when MOSC_EN=0 and the device enters deep sleep, the PLL will automatically be disabled and will be enabled again after wake-up. When MOSC_EN=1 and the device enters deep sleep, the PLL will not be off.

The PLL module supports lock interrupt or lost lock interrupt, selected by the LK_IFS<2:0>. When LK_IFS<2:0> =000 or 010, an interrupt will be generated on a successful PLL lock. When LK_IFS<2:0> =001 or 011, an interrupt will be generated on a failed PLL lock. When LK_IFS<2:0> =100, 101, 110 or 111, an interrupt can be generated no matter the PLL succeeded in locking or failed to lock.

During PLL operation, it is not recommended to bypass the PLL lock signal. If PLL_BYLOCK=0, when the PLL clock lost lock, the system will automatically switch to the original clock source used before PLL (determined by CLK_SEL). If PLL_BYLOCK =1, when the PLL clock lost lock, the system will remain clocked by the PLL, which might cause irregular behavior of the device.

Two I/O ports are available to output the clock, where the CLKO0 port is dedicated to directly outputting the high frequency while the CLKO1 port is dedicated to outputting the high frequency divided by 512. The corresponding GPIO_PAFUNC/ GPIO_PBFUNC register must be configured to enable the output clock function on the pin. When the high frequency clock is being output directly, enable the high current drive on the associated pin to avoid waveform distortion.

2. 6. 3. 5 External Clock Check Management CCM

The following conditions must exist to enable the external clock check management CCM which is able to detect the clock-off state.

1) The CCM enable bit EN of the SCU_CCM register must be set, and it defaults to be enabled.

2) The external clock must be selected as system clock by writing CLK_SEL<1:0>=10 of the SCU_SCLKEN0 register. Note that, in the case that the external clock is selected as the input clock source for the PLL, if either condition of the above is not true, the external CCM will not work.

During the CCM operation, if the external clock is found off, the system clock will automatically be switched to LRC clock and the corresponding interrupt flag will be set at the same time. It is worth mentioning that when MOSC_EN=0, the device will enter deep sleep mode, and the external clock source will be disabled, which causes the external clock to stop oscillating. In this case, the external oscillator off FLAG will not be set. Following the interrupt, switch the system clock to either LRC or HRC by writing the CLK_SEL<1:0> bits. If the PLL is used, the PLL module needs to be disabled too

(PLL_MUX=0, PLL_EN=0). After the clock-off issue is resolved, the external clock must be enabled again by software. See the CCM interrupt service routine for details.

External clock check management interrupt service routine

```
INT_CCM PROC
PUSH {LR}
LDR     R0, =SCU_SCLKEN0
LDR     R1, [R0]
LDR     R2, = 0XFFFFFFFD    ; Switch the system clock to LRC
ADDS    R1, R1, R2

LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, = 0XFFFFFFFE    ; Disable external clock oscillator
ADDS    R1, R1, R2
…
```

2. 6. 3. 6 Clock Filter CLKFLT

The HR8P506 supports system clock filtering before frequency division.

Upon enabling the system clock filtering, the CLKFLT_EN bit of the SCU_WAKEUPTIME register must be set and select the filtered system clock by writing CLKFLT_BY≠0x55 of the SCU_SCLKEN0.

Upon disabling the clock filter, bypass the CLKFKT first by writing CLKFLT_BY=0x55, and then write the CLKFLT_EN=0. See the clock filter example for details.

In the case of CLKFLT_EN=1, when MOSC_EN=0 and deep sleep mode is entered, the CLKFLT will automatically be disabled, and again automatically enabled after wake up. When MOSC_EN=1 and deep sleep mode is entered, the CLKFLT will not be disabled.

When the system clock is the PLL output 48MHz, you need to set CLKFLT_BY<7:0>=0x55 to bypass the clock filter. When the system clock is one of other clock sources, you need to set CLKFLT_BY<7:0>=0x00 (or other values rather than 0x55).

To ensure the reliability of the system operation, disabling the CLKFLT is not recommended except the system clock is 48MHz output by PLL.

2. 6. 3. 7 Device Status in Sleep Mode

In sleep mode, the status of the core processor, the content of the registers, peripheral registers and SRAM, and the port levels will remain the same with what they were before entering sleep mode.

2. 6. 3. 8 Normal Sleep Mode

In normal sleep mode, the core clock stops running, and the peripheral clock PCLK

and the device clock source continue to run.

2. 6. 3. 9 Deep Sleep Mode

In deep sleep mode, both the core clock and peripheral clock PCLK stop running.

1) When MOSC_EN=0, the XTAL, HRC, PLL and CLKFLT are all disabled except for LRC. The peripherals clocked from the LRC will continue operating. However, if the peripherals are clocked from other clock source, they will stop running (the asynchronous wake-up function is still active). Following wake-up, the XTAL, HRC, PLL and CLKFLT will resume operating.

2) When MOSC_EN=1, the clock source, PLL and CLKFLT continue operating. The peripherals clocked from LRC and XTAL will continue running. However, if the peripherals are clocked from PLCK or other divided clocks, they will stop running (the asynchronous wake-up function is still active).

2. 6. 4 Special Function Registers

System Clock Enable Register 0(SCU_SCLKEN0)

Offset address:40_H

Reset value:$00000000\_00000000\_00000000\_00000100_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				CLKOUT1_SEL<1:0>		CLKOUT0_SEL<1:0>		CLKFLT_BY<7:0>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	SYSCLK_DIV			Reserved			PLL_MUX	Reserved					XTAL_LP	CLK_SEL<1:0>	

—	bit 31-28	—	—
CLKOUT1_SEL<1:0>	bit27-26	R/W	**CLKO1 pin output select bits** 00: Clock output disabled 01: System clock output (divided by 512) 10: LRC clock output 11: HRC clock output (divided by 512)
CLKOUT0_SEL<1:0>	bit 25-24	R/W	**CLKO0 pin output select bits** 00: Clock output disabled 01: System clock output 10: LRC clock output 11: HRC output
CLKFLT_BY<7:0>	bit 23-16	R/W	**CLKFLT bypass enable bits** 8'h55: Bypassing CLKFLT enabled Other: Bypassing CLKFLT disabled CLKFLT is the system clock filter. To ensure the stability of the system, bypassing the CLKFLT is not recommended.

—	bit15	—	—
SYSCLK_DIV<2:0>	bit14-12	R/W	**System clock postscaler ratio select bits** 000:1:1 001:1:2 010:1:4 011:1:8 100:1:16 101:1:32 110:1:64 111:1:128
—	bit11-9	—	—
PLL_MUX	bit8	R/W	**System clock select bit** 0: Original clock(selected by CLK_SEL**)** 1: PLL clock
—	bit7-3	—	—
XTAL_LP	bit2	—	**Power mode select bit in external low-speed mode** 0: Low power (fixed to 0 via software) 1: High power (for test only)
CLK_SEL<1:0>	bit1-0	R/W	**Original clock source select bits** 00: 16MHz HRC 01: 32KHz LRC 10: XTAL clock (HS, XT or LP selected by CFG_OSCMD) 11: 16MHz HRC

Notes

1. Prior to writing to the SCU_SCLKEN0 register, configure the SCU_PROT register to disable the write protection.
2. Upon selecting 32MHz or 48MHz PLL clock as the system clock, configure the ACCT<3:0> of the SCU_FLASHWAIT register to select the proper Flash read time, then switch to 32MHz or 48MHz or else it may result in error in instruction execution. See Flash Memory Wait Function for more details.
3. The XTAL_LP bit can be written only after the XTAL_EN bit of the SCU_SCLKEN1 register is enabled.
4. When the configuration word sets the external oscillator to HS and XT mode, the XTAL_LP software configuration is invalid and the external oscillator is always in high power mode. When the external oscillator is set to LP mode, XTAL_LP must be set to 0 to enable the external clock to operate in low power mode.

System Clock Enable Register1(SCU_SCLKEN1)

Offset address:44_H

Reset value:$00000000\_00000010\_00000000\_00000010_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													PLL_RDY	HRC_RDY	XTAL_RDY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PLL_BYLOCK	PLL_EN	PLL_48M_SEL	PLL_REF_SEL<2:0>			Reserved						HRC_EN	XTAL_EN

—	bit 31-19	—	—
PLL_RDY	bit18	R	**PLL clock ready flag bit** 0: Not stabilized 1: Stabilized
HRC_RDY	bit17	R	**Internal high speed HRC ready flag bit** 0: Not stabilized 1: Stabilized
XTAL_RDY	bit16	R	**External oscillator ready flag bit** 0: Not stabilized 1: Stabilized This flag is only active when XTAL_EN=1
—	bit15-14	—	—
PLL_BYLOCK	bit13	R/W	**PLL lock bypass enable bit** 0: Disabled 1: Enabled
PLL_EN	bit12	R/W	**PLL enable bit** 0: Disabled 1: Enabled (ensure the stability of the clock source selected by PLL_REF_SEL clock source before enabling the PLL)
PLL_48M_SEL	bit11	R/W	**PLL output clock select bit** 0: Output 32MHz clock 1: Output 48MHz clock
PLL_REF_SEL<2:0>	bit10-8	R/W	**PLL input clock source select bit** 000: Disabled 001: Disabled 010: Internal HRC clock (about 16MHz) 011: Internal LRC clock (about 32KHz) 100: External 4MHz clock 101: External 8MHz clock 110: External 16MHz clock 111: External 32KHz clock
—	bit7-2	—	—
HRC_EN	bit1	R/W	**Internal high speed HRC enable bit**

			0: Disabled 1: Enabled
XTAL_EN	bit0	R/W	**External oscillator enable bit** 0: Disabled 1: Enabled

Notes

1. Prior to writing to the SCU_SCLKEN1 register, configure the SCU_PROT register to disable the write protection.
2. If the low speed LP mode is selected for the XTAL clock with the configuration word CFG_OSCMD, prior to setting the XTAL_EN to enable the external oscillator, set XTAL_LP=0 by software to put the LP oscillator in low power mode.
3. If the LP mode needs to be selected for the external XTAL clock by the configuration word CFG_OSCMD, first set the XTAL_EN=1, then set XTAL_LP=0 of the SCU_SCLKEN0 to put the LP oscillator in low power mode.

Peripheral Clock Enable Register (SCU_PCLKEN)

Offset address:48_H

Reset value:$00010011\_00010011\_00011111\_11110111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			I2C0_EN	Reserved		SPI1_EN	SPI0_EN	Reserved			EUART0_EN	Reserved		UART1_EN	UART0_EN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			T32N0_EN	T16N3_EN	T16N2_EN	T16N1_EN	T16N0_EN	WDT_EN	LCD_EN	RTC_EN	ADC_EN	Reserved	IAP_EN	GPIO_EN	SCU_EN

—	bit 31-29	—	—
I2C0_EN	bit28	R/W	**I2C0 clock enable bit** 0: Disabled 1: Enabled
—	bit 27-26	—	—
SPI1_EN	bit 25	R/W	**SPI1 clock enable bit** 0: Disabled 1: Enabled
SPI0_EN	bit 24	R/W	**SPI0 clock enable bit** 0: Disabled 1: Enabled
—	bit 23-21	—	—
EUART0_EN	bit 20	R/W	**EUART0 clock enable bit** 0: Disabled 1: Enabled

—	bit 19-18	—	—
UART1_EN	bit 17	R/W	**UART1 clock enable bit** 0: Disabled 1: Enabled
UART0_EN	bit 16	R/W	**UART0 clock enable bit** 0: Disabled 1: Enabled
—	bit 15-13	—	—
T32N0_EN	bit 12	R/W	**T32N0 clock enable bit** 0: Disabled 1: Enabled
T16N3_EN	bit 11	R/W	**T16N3 clock enable bit** 0: Disabled 1: Enabled
T16N2_EN	bit 10	R/W	**T16N2 clock enable bit** 0: Disabled 1: Enabled
T16N1_EN	bit 9	R/W	**T16N1 clock enable bit** 0: Disabled 1: Enabled
T16N0_EN	bit 8	R/W	**T16N0 clock enable bit** 0: Disabled 1: Enabled
WDT_EN	bit 7	R/W	**WDT clock enable bit** 0: Disabled 1: Enabled
LCD_EN	bit 6	R/W	**LCDC clock enable bit** 0: Disabled 1: Enabled
RTC_EN	bit 5	R/W	**RTC clock enable bit** 0: Disabled 1: Enabled
ADC_EN	bit 4	R/W	**ADC clock enable bit** 0: Disabled 1: Enabled
—	bit 3	—	—
IAP_EN	bit 2	R/W	**FLASH_IAP clock enable bit** 0: Disabled 1: Enabled
GPIO_EN	bit 1	R/W	**GPIO clock enable bit** 0: Disabled 1: Enabled
SCU_EN	bit 0	R/W	**SCU clock enable bit**

			0: Disabled 1: Enabled

Notes

1. Prior to writing to the SCU_PCLKEN register, configure the SCU_PROT register to disable the write protection.
2. Prior to enabling the clock for any peripheral, first enable the SCU clock by setting the SCU_EN in the SCU_PCLKEN register.

   For example: configuring WDT clock

```
LDR   R0, =SCU_PCLKEN
LDR   R1, =0X00000001
STR   R1, [R0]              ; enable SCU clock
LDR   R1, =0X00000081
STR   R1, [R0]              ; enable SCU and WDT clock
```

3. When the WDT clock enable bit is 0 (WDT_EN=0), any read or write to the WDT module is prohibited. However, the WDT counter keeps running and the watchdog timer is still active.

System Wake-up Time Control Register (SCU_WAKEUPTIME)

Offset address:$4C_H$

Reset value:$00000000\_00000000\_00110011\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved													LDOLP_VOSEL<2:0>		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
-	FLASHPW_PD	CLKFLT_EN	MOSC_EN	WAKEUPTIME<11:0>											

—	bit31-19	—	—
LDOLP_VOSEL<2:0>	bit18-16	W/R	**LDO output voltage select bits in deep sleep** 000: 1.5V 100: 1.4V (recommended) Others: For test only, do not set.
—	bit15	—	—
FLASHPW_PD	bit14	W/R	**Flash power control in sleep mode** 0: On 1: Off (for test only. It is prohibited to off the Flash power in practical use)
CLKFLT_EN	bit13	R/W	**System clock filter enable bit** 0: Disabled 1: Enabled To ensure the stability of the system, during normal operation, the CLKFLT must stay enabled. During DEEP SLEEP, the CLKFLT can be disabled to reduce the power consumption.

MOSC_EN	bit12	R/W	**Clock control during deep sleep** 0: HRC, PLL, XTAL and CLKFLT auto disabled during deep sleep 1: HRC, PLL, XTAL and CLKFLT auto enabled during deep sleep
WAKEUPTIME<11:0>	bit11-0	R/W	**Wake-up time control** T_{PCLK}* WAKEUPTIME

Notes

1. Prior to writing to the SCU_WAKEUP register, configure the SCU_PROT register to disable the write protection.
2. The LDOLP_VOSEL<2:0> bits need to be fixed to 100 by software during chip initialization, to reduce the power consumption in deep sleep mode.
3. The FLASHPW_PD bit needs to be fixed to 0 by software. It is prohibited to write the bit to1 or else it may cause irregular behavior of the device.
4. During deep sleep mode, when MOSC_EN =1, the HRC, PLL, XTAL and CLKFLT can be enabled with their individual enable bit HRC_EN, PLL_EN, XTAL_EN and CLKFLT_EN set.
5. The WAKEUPTIME<11:0> bits are used to set the wait time for the disabled HRC, PLL and XTAL to resume operating after waking up from deep sleep mode. Usually, the wait time is maintained by default, and it can be adjusted based on the actual operating status of the application system. During deep sleep, if the HRC, PLL and XTAL remain enabled, then the wait time can be set to 0.

External Clock Check Management Register (SCU_CCM)

Offset address:54_H

Reset value:$00000000\_00000000\_00000000\_00000001_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															FLAG

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							IF	IFS<2:0>			IE	Reserved			EN

—	bit31-17	—	—
FLAG	bit16	R	**External oscillator off flag bit** 0: external oscillator keeps running 1: external oscillator stopped
—	bit15-9	—	—
IF	bit8	R/W	**CCM interrupt flag bit** 0: CCM trigger event not occurred 1: CCM trigger event occurred In edge-triggered mode, clear the flag by writing 1. In level-triggered mode, the flag is read-only, and it will be automatically cleared when the triggering level

			disappears.
IFS<2:0>	bit7-5	R/W	**CCM interrupt trigger mode select bits** 000: CCM FLAG rising edge triggers the interrupt and the oscillator stops. 001: CCM FLAG falling edge triggers the interrupt and the oscillator resumes. 010: CCM FLAG high level triggers the interrupt and the oscillator stops. 011:CCM FLAG low level triggers the interrupt and the oscillator resumes 1xx:CCM FLAG change (rising or falling edge) triggers the interrupt
IE	bit 4	R/W	**External clock interrupt enable bit** 0: Disabled 1: Enabled
—	bit3-1	—	—
EN	bit0	R/W	**External clock check enable bit** 0: Disabled 1: Enabled

Note: Prior to writing to the SCU_CCM register, configure the SCU_PROT register to disable write protection.

PLL Lock Interrupt Control Register (SCU_PLLLKCON)

Offset address:30_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															LK_FLAG

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							IF	Reserved	LK_IFS<2:0>			Reserved			IE

—	bit31-17	—	—
LK_FLAG	bit16	R	**PLL lock flag bit** 0:PLL not locked 1:PLL locked
—	bit15-9	—	—
IF	bit8	R/W	**PLL interrupt flag bit** 0: PLL lock flag trigger event not occurred 1: PLL lock flag trigger event occurred In edge-triggered mode, clear the flag by writing 1. In level-triggered mode, the flag is read-only, and it will be automatically cleared when the triggering level disappears.

—	bit7	—	—
LK_IFS<2:0>	bit6-4	R/W	**PLL lock interrupt trigger mode select bits** 000: PLL lock flag rising edge triggers the interrupt and the locking is successful. 001: PLL lock flag falling edge triggers the interrupt and the lock fails. 010: PLL lock flag high level triggers the interrupt and the locking is successful. 011: PLL lock flag low level triggers the interrupt and the locking fails. 1xx: PLL lock flag change (rising or falling edge) triggers the interrupt.
—	bit3-1	—	—
IE	bit0	R/W	**PLL lock interrupt enable bit** 0: Disabled 1: Enabled

Note: Configure the SCU_PROT register to disable write protection prior to writing to the SCU_PLLLKCON register.

2. 6. 5 System Clock Application Notes

For the operations below, the system protection register SCU_PROT has been disabled and the SCU clock CLKEN_SCU bit has been enabled.

2. 6. 5. 1 External Clock XTAL

Using external clock XTAL:

```
SWITCH_XTAL PROC
PUSH    {LR}
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X01
ORRS    R1, R1, R2
STR     R1, [R0]                    ; Enable XTAL_EN
WAIT_XTAL_FLAG
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X010000
TST     R1, R2
BEQ     WAIT_XTAL_FLAG              ; Wait for XTAL_RDY

; If the external low speed LP clock is used as system clock and works in the low
  power mode, the setting is as follows.
LDR     R0, =SCU_SCLKEN0
```

```
LDR     R1, =0X02
STRB    R1, [R0]                ;Configure the low speed LP clock in
                                ;low power mode
POP     {PC}
ALIGN
LTORG
ENDP
```

2. 6. 5. 2 Internal High Speed Clock HRC

Using the internal high speed clock HRC:

```
SWITCH_HRC PROC
PUSH    {LR}
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X02
ORRS    R1, R1, R2
STR     R1, [R0]                ; Enable HRC_EN
WAIT_HRC_FLAG
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X020000
TST     R1, R2
BEQ     WAIT_HRC_FLAG           ; Wait for HRC_RDY

LDR     R0,=SCU_SCLKEN0
LDRB    R1,[R0]
LDR     R2, =0XFC
ANDS    R1, R1, R2
STRB    R1, [R0]                ; Select HRC as system clock
POP     {PC}
ALIGN
LTORG
ENDP
```

2. 6. 5. 3 Internal Low Speed Clock LRC

Using the internal low speed clock LRC:

```
SWITCH_LRC PROC
PUSH    {LR}
LDR     R0,=SCU_SCLKEN0
LDRB    R1,[R0]
LDR     R2, =0XFC
ANDS    R1, R1, R2
LDR     R2,=0X01
ORRS    R1,R1,R2
```

```
STRB    R1, [R0]                ; Select LRC as system clock
POP     {PC}
ALIGN
LTORG
ENDP
```

2. 6. 5. 4 Phase Locked Loop PLL

Prior to using PLL, first enable the PLL input clock source and wait for the clock source to be stabilized.

Example: input external 16MHz clock to the PLL and output 48MHz

```
SWITCH_XTAL16M_PLL48M PROC
PUSH    {LR}
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X01
ORRS    R1, R1, R2
STR     R1, [R0]                ; Enable XTAL_EN. Note that the
                                ; external oscillator must be of 16MHz
WAIT_XTAL_FLAG
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X010000
TST     R1, R2
BEQ     WAIT_XTAL_FLAG          ; Wait for XTAL_RDY
LDR     R0,=SCU_SCLKEN1
LDR     R1,=0X1E                ;Select the input clock source to the PLL
STRB    R1,[R0,#0X01]           ; Not bypass the PLL lock signal and
                                ; enable PLL module
WAIT_PLL_FLAG
LDR     R0, =SCU_SCLKEN1
LDR     R1, [R0]
LDR     R2, =0X040000
TST     R1, R2
BEQ     WAIT_PLL_FLAG           ; Wait for PLL_RDY
LDR     R0, =SCU_SCLKEN0
LDR     R1, [R0]
LDR     R2, =0X0100
ORRS    R1, R1, R2
STR     R1, [R0]                ; Select PLL multiplier as system clock
POP     {PC}
ALIGN
LTORG
ENDP
```

To disable PLL:

```
POWER_OFF_PLL PROC
PUSH    {LR}
LDR     R0, =SCU_SCLKEN0
LDRB    R1, [R0, #0X01]
LDR     R2, =0XFE
ANDS    R1, R1, R2
STRB    R1, [R0, #0X01]          ;  Select the original clock as system clock
LDR     R0, =SCU_SCLKEN1
LDRB    R1, [R0, #0X01]
LDR     R2, =0X0F
ANDS    R1, R1, R2
STRB    R1, [R0, #0X01]          ; Disable the PLL module
POP     {PC}
ALIGN
LTORG
ENDP
```

2. 6. 5. 5 Clock Filter CLKFLT

Using CLKFLT:

```
POWER_ON_CFT PROC
PUSH    {LR}
LDR     R0, =SCU_WAKEUPTIME
LDRB    R1, [R0, #0X01]
LDR     R2, =0X02
ORRS    R1, R1, R2
STRB    R1, [R0, #0X01]          ; Enable CLKFLT_EN
LDR     R0, =SCU_SCLKEN0
LDRB    R1, =0X55
STRB    R1, [R0, #0X02]          ;  Select the filtered clock as system clock
POP     {PC}
ALIGN
LTORG
ENDP
```

To disable CLKFLT:

```
POWER_OFF_CFT PROC
PUSH    {LR}
LDR     R0, =SCU_SCLKEN0
LDRB    R1, =0X00
STRB    R1, [R0, #0X02]          ;  Select the un-filtered clock as system clock
LDR     R0, =SCU_WAKEUPTIME
LDRB    R1, [R0, #0X01]
```

```
LDR     R2, =0XFD
ANDS    R1, R1, R2
STRB    R1, [R0, #0X01]                 ; Disable CLKFLT_EN
POP     {PC}
ALIGN
LTORG
ENDP
```

2. 7 Interrupts and Exceptions Handler

2. 7. 1 Interrupts and Exceptions

In the Cortex-M0 processor, the built-in NVIC (Nested Vectored Interrupt Controller) supports the following:

- ◇ Interrupt nesting
- ◇ Interrupt vectors
- ◇ Dynamic adjustment of interrupt priority
- ◇ Maskable interrupts.

For Cortex-M0 core, exceptions are events that cause changes in program flow control outside a normal code sequence, and interrupts is one kind of exceptions. To make it easier to understand, this document defines the core interrupts as exceptions and peripheral interrupts as interrupts.

The table below describes the operations of exceptions/interrupt priority.

Operation	Description
Preempt	Condition: a higher priority exception/interrupt occurs during ISR execution or thread mode Result : if currently in thread mode, a pending interrupt will be generated. If during ISR execution, the interrupt nesting will be generated, and the processor will automatically save the operating status and perform pushing.
Tail Chaining	Condition: Following the completion of the current ISR execution, a higher priority exception/interrupt occurs during returning. Result : skip the popping process to handle the new exception/interrupt
Return	Condition: Following the completion of the current ISR execution, no higher priority exception/interrupt occurs during returning. Result : perform the popping process and restore the status of the processor it had before entering ISR
Late Arrival	Condition: Following the beginning of the current ISR execution, a higher priority exception/interrupt occurs during storing. Result : the processor switches to handle the higher priority exception/interrupt first

Table 2-2 Operation Description of Exception/Interrupt Priority

Note: ISR – Interrupt Service Routine

Exception Number	Type	Priority	Descriptions
0	N/A	N/A	No exception
1	Reset	-3 (Highest)	Reset
2	NMI	-2	Nonmaskable interrupt (from external interrupt input)
3	Hard Fault	-1	All disabled Faults will be upgraded to Hard Fault
4~10	Reserved	NA	—
11	SVC	Programmable	Supervisor call via SVC instruction
12~13	Reserved	NA	—
14	PendSV	Programmable	Pending request for system service
15	SysTick	Programmable	System tick timer
16	IRQ0	Programmable	Peripheral interrupt 0
17	IRQ1	Programmable	Peripheral interrupt 1
…	…	…	…
47	IRQ31	Programmable	Peripheral interrupt 31

Table 2-3 List of Exceptions/Interrupts

The Cortex-M0 supports the following exceptions/interrupts:

NMI interrupts, Hard Fault, SVC exception, PendSV exception, SysTick exception and 32 interrupt requests IRQ0~IRQ31.

Hard Fault, SVC exception, PendSV exception and SysTick exception are core exceptions which are controlled by Cortex-M0 while NMI interrupts and 32 IRQs are controlled by the configuration word.

Although Cortex-M0 does not support the NMI enable bit, the chip provides the enable bit NMIEN to avoid generating NMI interrupt source before completion of power-on initialization. The enable bit can be set (NMIEN=1) after the NMI interrupt source configuration is completed.

The Cortex-M0 offers 32 individual IRQ enable bits. The IRQs can be enabled and disabled independently by configuring the interrupt control registers NVIC_ISER and NVIC_ICER.

Configuring the priority control registers NVIC_PR0~NVIC_PR7 can set the priority level of IRQ0~IRQ31. If several IRQs are generated concurrently, the highest priority IRQ will be first responded. If the IRQs have the same priority level, the IRQ having the lowest number, according to the interrupt vector table, will be responded first. For example, if IRQ0 and IRQ1 have the same priority level and are asserted at the same time, IRQ0 will be first responded.

2. 7. 2 Interrupt and Exception Vector Allocation

Vector Number	Type	Function	Descriptions
0~15	Exception	—	Cortex-M0 internal exceptions, including NMI
16	IRQ0	PINT0 interrupt	External port interrupt 0
17	IRQ1	PINT1 interrupt	External port interrupt 1
18	IRQ2	PINT2 interrupt	External port interrupt 2
19	IRQ3	PINT3 interrupt	External port interrupt 3
20	IRQ4	PINT4 interrupt	External port interrupt 4
21	IRQ5	PINT5 interrupt	External port interrupt 5
22	IRQ6	PINT6 interrupt	External port interrupt 6
23	IRQ7	PINT7 interrupt	External port interrupt 7
24	IRQ8	T16N0 interrupt	16-bit timer/counter 0 interrupt
25	IRQ9	T16N1 interrupt	16-bit timer/counter 1 interrupt
26	IRQ10	T16N2 interrupt	16-bit timer/counter 2 interrupt
27	IRQ11	T16N3 interrupt	16-bit timer/counter 3 interrupt
28	IRQ12	T32N0 interrupt	32-bit timer/counter 0 interrupt
29	IRQ13	Reserved	Reserved
30	IRQ14	Reserved	Reserved
31	IRQ15	Reserved	Reserved
32	IRQ16	WDT interrupt	Watchdog interrupt
33	IRQ17	RTC interrupt	Real time clock interrupt
34	IRQ18	KINT interrupt	External key input interrupt
35	IRQ19	ADC interrupt	Analog to digital conversion interrupt
36	IRQ20	Reserved	Reserved
37	IRQ21	LVD interrupt	Low voltage detection interrupt
38	IRQ22	PLLLK interrupt	PLL lost lock interrupt
39	IRQ23	UART0 interrupt	UART0 interrupt
40	IRQ24	UART1 interrupt	UART1 interrupt
41	IRQ25	EUART0 interrupt	EUART0 interrupt
42	IRQ26	Reserved	Reserved
43	IRQ27	SPI0 interrupt	SPI0 interrupt
44	IRQ28	SPI1 interrupt	SPI1 interrupt
45	IRQ29	I2C0 interrupt	I2C0 interrupt
46	IRQ30	Reserved	Reserved
47	IRQ31	CCM interrupt	External oscillator stop interrupt

Table 2-4 List of IRQ Allocation

2. 7. 3 Interrupt Vector Table Re-map

The Cortex-M0 itself does not support the interrupt vector table re-map. However, in HR8P506, there are two special registers, which are interrupt vector table re-map enable register and interrupt vector offset register, supporting the interrupt vector table re-map.

See Flash In-Application Programming (IAP) for more details.

2. 7. 4 Special Function Registers

Nonmaskable Interrupt Control Register(SCU_NMICON)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										NMICS<4:0>					NMIEN

—	bit31-6	—	—
NMICS<4:0>	bit5-1	R/W	**NMI select bits** 00000:IRQ0 00001:IRQ1 … 11111:IRQ31
NMIEN	bit0	R/W	**NMI enable bit** 0: Disabled 1: Enabled

Note: Prior to writing to the SCU_NMICON register, configure the SCU_PROT register to disable write protection.

Interrupt Vector Table Re-map Enable Register (SCU_TBLREMAPEN)

Offset address:60_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

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

—	bit31-1	—	—
EN	bit0	R/W	**Interrupt vector table re-map enable bit** 0: The interrupt table is located in Flash memory and occupies a space of size of 192 bytes from the address “0”. It currently supports 48 interrupt vectors. 1: The interrupt table has a size of 192 bytes with the starting address specified by the interrupt vector table offset register

Note: Prior to writing to the SCU_TBLREMAPEN register, configure the SCU_PROT register to disable write protection.

Interrupt Vector Table Offset Register (SCU_TBLOFF)

Offset address:64_H

Reset value:$00100000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBLOFF<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBLOFF<15:8>								TBLOFF<7:0>							

TBLOFF<31:0>	bit31-0	R/W	**Interrupt vector table offset address** This register stores the starting address of the re-mapped interrupt vector table and is valid only with the interrupt vector table re-map enable register being set. The upper 24 bits TBLOFF<31:8> are both readable and writable. However, the lower 8 bits TBLOFF<7:0> is read-only, read as zero.

Note: To select the starting address, first acquire the total number of vectors, and then increase the number to make it to the power of 2. Say, given 32 interrupt and 16 exceptions, the total number of vectors 32+16 =48, which needs to be increased to 2^4 = 64. In addition, the address must be divisible by 64x4= 256; therefore, the legal starting address can be 0x000, 0x100 and 0x200 etc.

Hard Fault Flag Register (SCU_FAULTFLAG)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													FLAG2	FLAG1	FLAG0

—	bit31-3	—	—
FLAG2	bit2	R/W	**Hard fault 2 flag bit** 0: No write operation occurred in exception area 1: Write operation occurred in exception area (automatically set to 1 by hardware, cleared by software writing 1)

FLAG1	bit1	R/W	**Hard fault 1 flag bit** 0: No instruction fetch occurred in exception area 1: Instruction fetch occurred in exception area (automatically set to 1 by hardware, cleared by software writing 1)
FLAG0	bit0	R/W	**Hard fault 0 flag bit** 0: No empty instruction code was read 1: Empty instruction code was read (automatically set to 1 by hardware, cleared by software writing 1)

Notes
1. Reading an empty instruction code means that the value read is $FFFFFFFF_H$ by the Cortex-M0 reading the instruction in Flash program memory.
2. Upon clearing the hard fault flags, configure the SCU_PROT register to disable write protection.

For the NVIC register list and base address, see section 3.5.2

IRQ0~31 Interrupt Set Enable Register (NVIC_ISER)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETENA<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETENA<15:0>															

SETENA<31:0>	bit31-0	R/W	**IRQ set enable bit** 0: Interrupt set disabled 1: Interrupt set enabled Enable the interrupt request by software writing1. Writing 0 to the bit has no effect.

Note: For each IRQ enable bit of the NVIC_ISER register, write 1 to enable the interrupt request while writing 0 has no effect. When reading the IRQ set enable bit, 1 indicates the interrupt set is enabled while 0 indicates the interrupt set is disabled.

IRQ0~31 Interrupt Clear Enable Register (NVIC_ICER)

Offset address:80_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRENA <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRENA <15:0>															

CLRENA<31:0>	bit31-0	R/W	**IRQ clear enable bit** 0: Interrupt clear disabled 1: Interrupt clear enabled Disable the interrupt request by software writing1. Writing 0 has no effect.

Note: For each IRQ disable bit of the NVIC_ICER register, write 1 to disable the interrupt request while writing 0 has no effect. When reading the IRQ clear enable bit, 1 indicates the interrupt clear is enabled while 0 indicates the interrupt clear is disabled.

IRQ0~31 Interrupt Pending Set Register(NVIC_ISPR)

Offset address:100_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
SETPEND <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SETPEND <15:0>															

SETPEND<31:0>	bit31-0	R/W	**IRQ pending set bit** 0: No interrupt is pended 1: Interrupt is pended Pend the interrupt by software writing1. Writing 0 has no effect

Note: For each IRQ pending bit of the NVIC_ISPR register, write 1 to pend the interrupt while writing 0 has no effect. When reading the IRQ pending bit, 1 indicates that the interrupt is in pending status while 0 indicates the interrupt is not pended.

IRQ0~31 Interrupt Pending Clear Register (NVIC_ICPR)

Offset address:180_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CLRPEND <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CLRPEND <15:0>															

CLRPEND<31:0>	bit31-0	R/W	**IRQ pending clear bit** 0: No interrupt is pended 1: Interrupt is pended

			Clear pending by software wring 1. Writing 0 has no effect.

Note: For each IRQ pending clear bit of the NVIC_ICPR register, write 1 to clear interrupt pending while writing 0 has no effect. When reading the IRQ pending clear bit, 1 indicates that the interrupt is pended while 0 indicates the interrupt is not pended.

IRQ0~3 Priority Control Register (NVIC_PR0)

Offset address:300_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_3<1:0>		Reserved						PRI_2<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_1<1:0>		Reserved						PRI_0<1:0>		Reserved					

PRI_3<1:0>	bit31-30	R/W	**IRQ3 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_2<1:0>	bit23-22	R/W	**IRQ2 priority bit** 00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_1<1:0>	bit15-14	R/W	**IRQ1 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_0<1:0>	bit7-6	R/W	**IRQ0 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ4~7 Priority Control Register (NVIC_PR1)

Offset address:304_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_7<1:0>		Reserved						PRI_6<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_5<1:0>		Reserved						PRI_4<1:0>		Reserved					

PRI_7<1:0>	bit31-30	R/W	**IRQ7 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_6<1:0>	bit23-22	R/W	**IRQ6 priority bit** 00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_5<1:0>	bit15-14	R/W	**IRQ5 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_4<1:0>	bit7-6	R/W	**IRQ4 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ8~11 Priority Control Register (NVIC_PR2)

Offset address:308_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_11<1:0>		Reserved						PRI_10<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_9<1:0>		Reserved						PRI_8<1:0>		Reserved					

PRI_11<1:0>	bit31-30	R/W	**IRQ11 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_10<1:0>	bit23-22	R/W	**IRQ10 priority bit** 00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_9<1:0>	bit15-14	R/W	**IRQ9 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_8<1:0>	bit7-6	R/W	**IRQ8 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ12~15 Priority Control Register (NVIC_PR3)

Offset address:30C $_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_15<1:0>		Reserved						PRI_14<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_13<1:0>		Reserved						PRI_12<1:0>		Reserved					

PRI_15<1:0>	bit31-30	R/W	**IRQ15 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_14<1:0>	bit23-22	R/W	**IRQ14 priority bit** 00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_13<1:0>	bit15-14	R/W	**IRQ13 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_12<1:0>	bit7-6	R/W	**IRQ12 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ16~19 Priority Control Register (NVIC_PR4)

Offset address:310$_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_19<1:0>		Reserved						PRI_18<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_17<1:0>		Reserved						PRI_16<1:0>		Reserved					

PRI_19<1:0>	bit31-30	R/W	**IRQ19 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_18<1:0>	bit23-22	R/W	**IRQ18 priority bit**

			00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_17<1:0>	bit15-14	R/W	**IRQ17 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_16<1:0>	bit7-6	R/W	**IRQ16 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ20~23 Priority Control Register (NVIC_PR5)

Offset address:314_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_23<1:0>		Reserved						PRI_22<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_21<1:0>		Reserved						PRI_20<1:0>		Reserved					

PRI_23<1:0>	bit31-30	R/W	**IRQ23 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_22<1:0>	bit23-22	R/W	**IRQ22 priority bit** 00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_21<1:0>	bit15-14	R/W	**IRQ21 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_20<1:0>	bit7-6	R/W	**IRQ20 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ24~27 Priority Control Register (NVIC_PR6)

Offset address:318_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_27<1:0>		Reserved						PRI_26<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_25<1:0>		Reserved						PRI_24<1:0>		Reserved					

PRI_27<1:0>	bit31-30	R/W	**IRQ27 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_26<1:0>	bit23-22	R/W	**IRQ26 priority bit** 00: Highest priority 11: Lowest priority
—	bit21-16	—	—
PRI_25<1:0>	bit15-14	R/W	**IRQ25 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_24<1:0>	bit7-6	R/W	**IRQ24 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

IRQ28~31 Priority Control Register (NVIC_PR7)

Offset address:$31C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_31<1:0>		Reserved						PRI_30<1:0>		Reserved					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PRI_29<1:0>		Reserved						PRI_28<1:0>		Reserved					

PRI_31<1:0>	bit31-30	R/W	**IRQ31 priority bit** 00: Highest priority 11: Lowest priority
—	bit29-24	—	—
PRI_30<1:0>	bit23-22	R/W	**IRQ30 priority bit** 00: Highest priority

			11: Lowest priority
—	bit21-16	—	—
PRI_29<1:0>	bit15-14	R/W	**IRQ29 priority bit** 00: Highest priority 11: Lowest priority
—	bit13-8	—	—
PRI_28<1:0>	bit7-6	R/W	**IRQ28 priority bit** 00: Highest priority 11: Lowest priority
—	bit5-0	—	—

2. 8 System Control Block (SCB)

2. 8. 1 Overview

The system control block SCB provides the status information of the core system implementation and controls operation of the core system.

See section 3.5.3 List of System Control Block (SCB) for details.

2. 8. 2 Special Function Registers

SCB_CPUID Register (SCB_CPUID)

Offset address:00_H

Reset value:$01000001\_00001100\_11000010\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
IMPLEMENTER<7:0>								VARIANT<3:0>				CONSTANT<3:0>			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PARTNO<11:0>												REVISION<3:0>			

IMPLEMENTER<7:0>	bit31-24	R	**Implementer code** 0x41, ARM
VARIANT<3:0>	bit23-20	R	**Variant** R=0x0, as the primary number in the numbering format for the rnpn version
CONSTANT<3:0>	bit19-16	R	**Constant** 0xC, ARMv6-M
PARTNO<11:0>	bit15-4	R	**Part number** 0xC20, Cortex-M0
REVISION<3:0>	bit3-0	R	**Revision** P=0x0, as the secondary number in the numbering format for the rnpn version

Interrupt Control and Status Register (SCB_ICSR)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NMIPENDSET	Reserved			PENDSTSET	PENDSTCLR			ISRPENDDING	Reserved				VECTPENDING<5:4>		

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VECTPENDING<3:0>				Reserved						VECTACTIVE<5:0>					

NMIPENDSET	bit31	R/W	**Pend NMI interrupt set bit** 0: No effect

			1: Write 1 to pend NMI
—	bit30-27	—	—
PENDSTSET	bit26	R/W	**Pend SysTick exception set bit** 0:No effect 1: Write 1 to pend SysTick exception
PENDSTCLR	bit25	W	**Pend SysTick exception clear bit** 0: No effect 1: Write 1 to clear SysTick exception
—	bit24-23	—	—
ISRPENDDING	bit22	R	**Interrupt pending flag bit** 0: No pended interrupt 1: Interrupt pended
—	bit21-18	—	—
VECTPENDING	bit17-12	R	**These bits indicate the vector number of the highest priority pending exception/ interrupt** 0x0: No exception/ interrupt is currently pended. Others: The exception number of the highest priority pending exception/ interrupt
—	bit11-6	—	—
VECTACTIVE	bit5-0	R	**These bits indicate the vector number of the currently served exception/interrupt** 0x0: Thread mode Others: The vector number of the currently served exception/interrupt

Application Interrupt and Reset Control Register (SCB_AIRCR)

Offset address:$0C_H$

Reset value:$11111010\_00000101\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VECTKEY<15:0>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ENDIANNESS	Reserved											SYSRESET REQ		VECTCLR ACTIVE	Reserved

VECTKEY<15:0>	bit31-16	W	**Register access key.** Write 0x05FA to these bits. Other values would be ignored.
ENDIANNESS	bit15	R	**System endianness** 0: Little endian 1: Big endian
—	bit14-3	—	—

SYSRESETREQ	bit2	W	**System reset request** 0: No effect 1: Write 1 to request a system reset, and the bit is automatically cleared following the reset.
VECTCLRACTIVE	bit1	W	**Exception/Interrupt clear bit** This bit can only be written as 0. Writing 1 would cause Hard Fault.
—	bit0	—	**—**

Note: The SCB_AIRCR register can only be accessed in word, and the upper half-word can only be written as 0x05FA. Otherwise, the write operation would be ignored.

System Control Register (SCB_SCR)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											SEVONPEND	Reserved	SLEEPDEEP	SLEEPONEXIT	Reserved

—	bit31-5	—	**—**
SEVONPEND	bit4	R/W	**Send event on pend bit** 0: When set to 0, this feature is disabled. 1: When set to 1, an event is generated for each new pending of an interrupt. This can be used to wake up the processor.
—	bit3	—	**—**
SLEEPDEEP	bit2	R/W	**Processor deep sleep and sleep mode select bit** 0: Normal sleep mode 1: Deep sleep mode
SLEEPONEXIT	bit1	R/W	**Sleep-on-exit enable bit** 0: When set to 0, this feature is disabled. 1: When set to 1, enter sleep mode automatically on exiting an ISR and returning to thread mode.
—	bit0	—	**—**

Configuration and Control Register (SCB_CCR)

Offset address:14_H

Reset value:00000000_00000000_00000010_00001000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

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

—	bit31-10	—	—
STKALIGN	bit9	R	Double word exception stacking alignment behavior is always used. Always read as 1.
—	bit8-4	—	—
UNALIGN_TRP	bit3	R	Instruction trying to carry out an unaligned access always causes a fault exception. Always read as 1.
—	bit2-0	—	—

System Handler Priority Register 2(SCB_SHPR2)

Offset address:$1C_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_11<1:0>		Reserved													

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

PRI_11<1:0>	bit31-30	R/W	**SVCall (exception number 11) priority setup bits**
—	bit29-0	—	—

System Handler Priority Register 3 (SCB_SHPR3)

Offset address:20_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PRI_15<1:0>		Reserved						PRI_14<1:0>		Reserved					

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

PRI_15<1:0>	bit31-30	R/W	**SysTick (exception number15) priority bits**
—	bit29-24	—	—
PRI_14<1:0>	bit23-22	R/W	**PendSV (exception number 14) priority bits**
—	bit21-0	—	—

2. 9 SysTick Timer

2. 9. 1 Overview

- ◇ 24-bit down counter for operating system, automatically reload value when reaching 0
- ◇ Able to generate SysTick exception at regular interval to allow multiple tasks to run at different time slots in embedded operating system; or allow a task to be executed at regular interval in non-embedded operating system
- ◇ Also can be used as a regular timer
- ◇ The priority of the SysTick exception is set by the PRI_15<1:0> of the SHPR3 register.
- ◇ Pend the SysTick exception by setting the PENDSTSET bit of the SCB_ICSR register
- ◇ Operating clock options: HCLK or HCLK divided by 3

The SysTick timer is a down counter, and its reload value can be set by the SYST_RVR register. After reaching 0, the COUNTFLAG will be set, and the SysTick will reload the value in SYST_RVR. The SysTick will stop counting if debugging stops. During counting, if the SYST_RVR register is set to 0, the SysTick will stop after reaching 0.

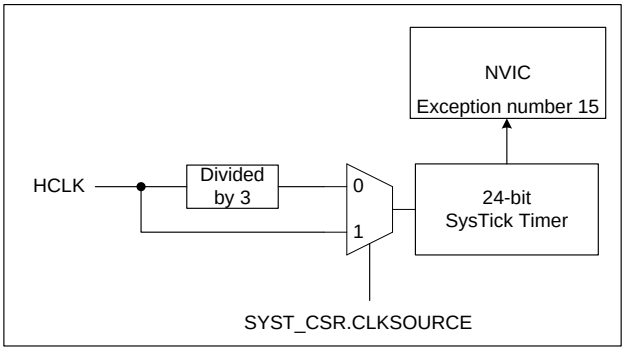


Figure 2-9 SysTick Timer Block Diagram

The current counter value of the SysTick can be obtained by reading the SYST_CVR register. Any write operation to the SYST_CVR register will cause the register itself and the COUNTFLAG to be cleared, but it will not trigger any SysTick exception.

The word-oriented operation is required when accessing the SysTick register. The following describes how to configure the SysTick counter.

1) Configure the reload value register SYST_RVR
2) Clear the current value register SYST_CVR
3) Configure the control and status register SYST_CSR

See section 3.5.1 List of SysTick Register for more details.

2. 9. 2 Special Function Registers

SysTick Control and Status Register (SYST_CSR)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														COUNTFLAG	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													CLKSOURCE	TICKINT	ENABLE

—	bit31-17	—	—
COUNTFLAG	bit16	R	**Counter flag** 0: 0 not reached 1: Set to 1 when the SysTick reaches 0 Clear the flag by either reading this register or writing the SYST_CVR register
—	bit15-3	—	—
CLKSOURCE	bit2	R/W	**SysTick clock source select bit** 0: reference clock 1: core clock
TICKINT	bit1	R/W	**SysTick interrupt enable bit** 0: No SysTick exception is generated when the SysTick counts down to 0. 1: When this bit is set, the SysTick exception is generated when the SysTick counts down to 0.
ENABLE	bit0	R/W	**SysTick counter enable bit** 0: Disabled 1: Enabled

Notes

1. The core clock is the HCLK of which the clock frequency is the same as the system clock frequency.
2. The reference clock is the core clock divided by 3, $F_{HCLK}/3$.

SysTick Reload Value Register (SYST_RVR)

Offset address:14_H

Reset value:$00000000\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								RELOAD<23:16>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RELOAD<15:0>															

—	bit31-24	—	—
RELOAD<23:0>	bit23-0	R/W	**SysTick counter reload value** The range is 0x00_0001~0xFF_FFFF. If it is 0, the SysTick does not count.

SysTick Current Value Register (SYST_CVR)

Offset address:18_H

Reset value:$00000000\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								CURRENT <23:16>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CURRENT<15:0>															

—	bit31-24	—	—
CURRENT<23:0>	bit23-0	R/W	**SysTick counter current value** On read returns the current value of the SysTick counter. Write to the register with any value to clear the register and the COUNTFLAG.

SysTick Calibration Register (SYST_CALIB)

Offset address:$1C_H$

Reset value:$01000000\_00000010\_10001011\_00001010_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
NOREF	SKEW	Reserved						TENMS<23:16>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TENMS<15:0>															

NOREF	bit31	R	**Reference clock flag bit** 0: External reference clock is not available but internal reference clock $F_{HCLK}/3$ is available. 1: External reference clock is available.
SKEW	bit30	R	**TENMS calibration value check bit** 0: The TENMS calibration value is accurate. 1: The TENMS calibration value is not accurate.
—	bit29-24	—	—
TENMS<23:0>	bit23-0	R/W	**SysTick Calibration value** If the bits read as 0, it indicates that the calibration value is unknown.

Note: This product only provides the internal reference clock, $F_{HCLK}/3$.

2. 10 Software Configuration Word

When the CFG_BORV<1:0> of the configuration word is 11, the BOR threshold voltage can be selected via the SCU_SOFTCFG register. When the power supply goes lower than the threshold voltage specified by the BORV<3:0> bits, a BOR reset will be generated.

System Software Configuration Register (SCU_SOFTCFG)

Offset address:24_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												BORV<3:0>			

—	bit31-4	—	—
BORV<3:0>	bit3-0	R/W	**BOR threshold voltage select bits (valid only when CFG_BORV=11)** 0000:1.7V 0001:2.0V 0010:2.1V 0011:2.2V 0100:2.3V 0101:2.4V 0110:2.5V 0111:2.6V 1000:2.8V 1001:3.0V 1010:3.1V 1011:3.3V 1100:3.6V 1101:3.7V 1110:4.0V 1111:4.3V

Note: Prior to writing the SCU_SOFTCFG register, configure the SCU_PROT register to disable write protection.

2. 11 Synchronization Control for T16N/T32N

2. 11. 1 Overview

The SCU_TIMEREN and SCU_TIMERDI register allow several T16N/T32N timers to be enabled or disabled at the same time. For other applications, the EN bit of T16N_CON0 or T32N_CON0 can be individually configured to enable or disable the timer. Note that, the priority level of the SCU_TIMEREN and SCU_TIMERDIS is higher than that of the EN bit of the T16N_CON0 and T32N_CON0. Moreover, the SCU_TIMEREN register has the higher priority than that of the SCU_TIMERDIS register.

2. 11. 2 Special Function Registers

Timer Enable Register (SCU_TIMEREN)

Offset address:34_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							T32N0EN	Reserved				T16N3EN	T16N2EN	T16N1EN	T16N0EN

—	Bit31-9	—	—
T32N0EN	bit8	R/W	**T32N0 enable bit** 0:— 1: Enabled
—	bit7-4	—	—
T16N3EN	bit3	R/W	**T16N3 enable bit** 0:— 1: Enabled
T16N2EN	bit2	R/W	**T16N2 enable bit** 0:— 1: Enabled
T16N1EN	bit1	R/W	**T16N1 enable bit** 0:— 1: Enabled
T16N0EN	bit0	R/W	**T16N0 enable bit** 0:— 1: Enabled

Notes

1. For each bit of the SCU_TIMEREN register, write 1 to enable the timer and the bit will be automatically cleared by hardware. Writing 0 to those bits has no effect.
2. Prior to writing to the SCU_TIMEREN register, configure the SCU_PROT register to disable write protection.

Timer Disable Register (SCU_TIMERDIS)

Offset address:38_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							T32N0DIS	Reserved				T16N3DIS	T16N2DIS	T16N1DIS	T16N0DIS

—	bit31-9	—	—
T32N0DIS	bit8	R/W	**T32N0 disable bit** 0:— 1: Disabled
—	bit7-4	—	—
T16N3DIS	bit3	R/W	**T16N3 disable bit** 0:— 1: Disabled
T16N2DIS	bit2	R/W	**T16N2 disable bit** 0:— 1: Disabled
T16N1DIS	bit1	R/W	**T16N1 disable bit** 0:— 1: Disabled
T16N0DIS	bit0	R/W	**T16N0 disable bit** 0:— 1: Disabled

Notes

1. For each bit of the SCU_TIMERDIS register, write 1 to disable the timer and the bit will be automatically cleared by hardware. Writing 0 to those bits has no effect.
2. Prior to writing to the SCU_TIMERDIS register, configure the SCU_PROT register to disable write protection

Chapter3 Memory

3. 1 Internal Memory Map

The internal memory resource contains the program memory, data memory, peripheral registers and core registers. Figure 3-1 shows the mapping of each region and reflects the details of the region of core registers.

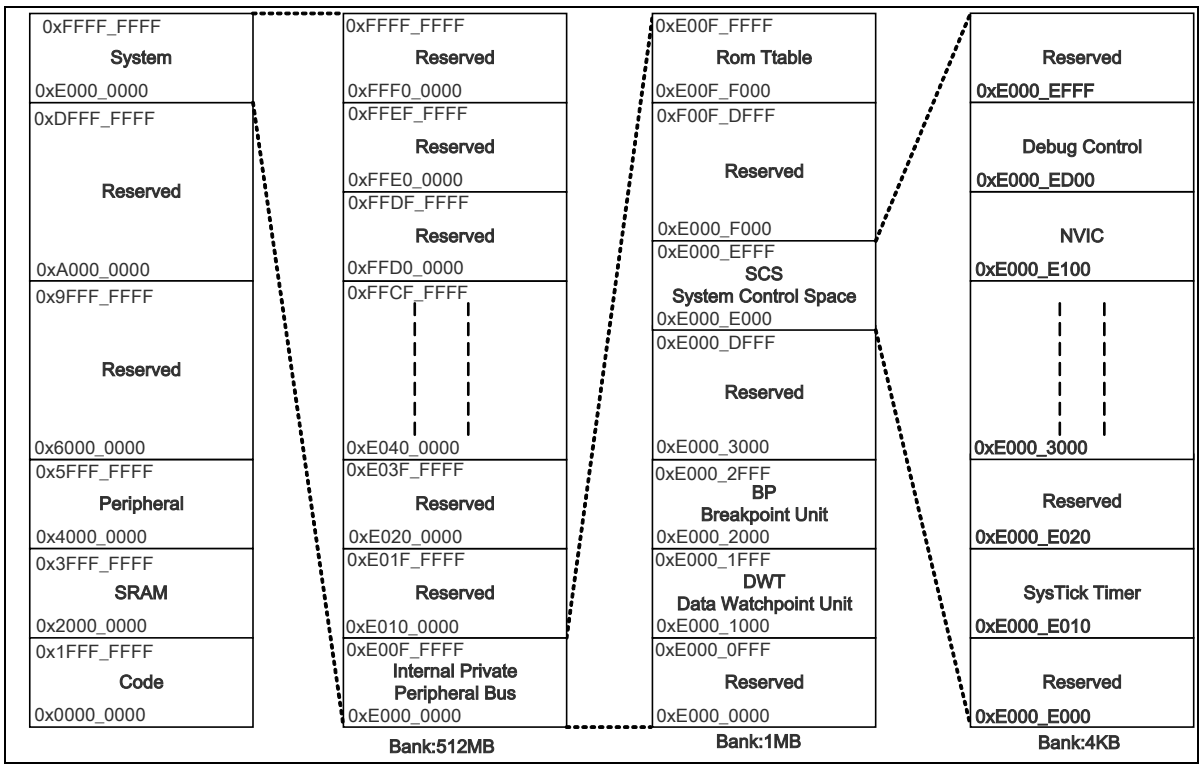


Figure 3-1 Internal Memory Map

3. 2 Flash Memory

3. 2. 1 Information Region FLASH

The information region FLASH in the chip is divided into two partitions: INFO0 and INFO1.

◇ The INFO0 information region is used to store the chip configuration words CFG_WORD0, CFG_WORD1, CFG_WORD2

- Programmer readable/writable

◇ The INFO1 information region contains 96 bits chip unique identifier UID

- The chip unique identification code UID has been fixed at factory and cannot be changed. The program is read-only

3. 2. 1. 1 Configuration Word

The configuration word is located at the information region of the Flash memory, and the user can configure it through ISP programming. All the function configurations are

accomplished together by the configuration word and function related registers. The configuration word is used for the external XTAL operating mode selection, WDT enabling and BOR threshold voltage selection etc.

Name	CFG_WORD0	
Address	00100004_H	
-	bit4-0	-
CFG_OSC_PIN	bit5	XTAL oscillator pin select bit 0: PB8 and PB9 1: PB10 and PB11(default)
INTOSC_SEL	bit6	**Internal HRC clock frequency select bit** 0:16MHz (default) 1: Others (for internal test only, prohibited to set to 1)
CFG_PWRTEB	bit7	140ms power-on time enable bit 0:Enabled (default) 1:Disabled This bit is valid only when the MRSTN pin is used for external reset (CFG_MRSTN≠00). It is recommended to enable the bit.
CFG_BORV	bit 9-8	BOR threshold voltage select bits 00:3.7V 01:2.5V 10:2.1V 11:1.7V (default, the BOR threshold voltage can be selected via the SCU_SOFTCFG register)
-	bit10	-
CFG_MRSTN	bit 12-11	MRSTN pin multiplexing configuration 00:GPIO Other: MRSTN function (default)
DEBUG_S	bit13	**SWD debug pin select bit** 0:PA14 and PA15 are selected as debug pins. 1:PA0 and PA1 are selected as debug pins (default)
CFG_DEBUG	bit 15-14	SWD debug mode enable bit 0x: Disabled 10: Enabled (default). The pins selected by the DEBUG_S bit are forced to be the SWD ports, and the user has no control for the selected pins. 11: Disabled

Notes

1. The CFG_PWRTEB is valid only when the MRSTN pin is used for external reset (CFG_MRSTN≠00). It is recommended to enable the bit (CFG_PWRTEB=0). Only when the device is required to quickly enter into operating mode after power on, and the power supply is stable and reliable, can the CFG_PWRTEB bit be disabled.

2. The 45KΩ weak pull-up resistor is built-in for the MRSTN pin being used for external reset.
3. When the MRSTN is used as a GPIO, or after a BOR has occurred, the power-on time 140ms is fixed to be enabled, and is independent from the CFG_PWRTEB.
4. When performing encryption programming on Flash, the CFG_DEBUG must be disabled. Otherwise, the encryption has no effect, and the programming tool would prompt an error message.

Name	CFG_WORD1	
Address	0010000C$_H$	
-	bit 0	-
CFG_WDTEN	bit1	Watchdog enable bit 0: Disabled (Default) 1: Enabled (The WDT clock source is the fixed internal LRC clock)
WDTINTEN	bit 2	Watchdog interrupt enable bit (valid only when CFG_WDTEN=1) 0: Disabled (default) 1: Enabled
-	bit 4-3	-
WDTRL	bit7-5	**WDT reload value select bit after power on (valid only when CFG_WDTEN=1)** 000:0x0000_0200 (WDT overflow time around 16ms) 001:0x0000_0400 (WDT overflow time around 32ms) 010:0x0000_1000 (WDT overflow time around 128ms) 011:0x0000_4000 (WDT overflow time around 512ms) 100:0x0000_8000 (WDT overflow time around 1s) 101:0x0001_0000 (WDT overflow time around 2s) 110:0x0002_0000 (WDT overflow time around 4s) 111:0x0004_0000 (WDT overflow time around 8s) (default)
FWPS	bit12-8	**Flash memory location write protection control bit** 00000: memory locations (0000_0000$_H$~0000_07FF$_H$) are write-protected. 00001: memory locations (0000_0000$_H$~0000_0FFF$_H$) are write-protected. 00010: memory locations (0000_0000$_H$~0000_17FF$_H$) are write-protected. 00011: memory locations (0000_0000$_H$~0000_1FFF$_H$) are write-protected. …… 10000: memory locations (0000_0000$_H$~0000_87FF$_H$) are write-protected. 1xxx1: memory locations (0000_0000$_H$~0000_8FFF$_H$) are write-protected (default).
FWPEB	bit13	**Flash IAP write-protected area enable bit**

		0: Enabled 1: Disabled (default)
-	bit15-14	-

Notes

1. After enabling the write-protected area, the memory locations in this area do not support IAP erase and programming.
2. In SWD mode, the WDT needs to be disabled. Otherwise, the WDT would keep counting and cause an overflow reset, which might lead to an irregular debugging behavior.

Name	CFG_WORD2	
Address	00100010_H	
-	bit7-0	-
CFG_OSCMD	bit15-8	XTAL oscillator operating mode select bits 0x2C: high-speed HS mode (5~20MHz) (default) 0x48: high-speed XT mode (1~4MHz) 0xF0: Low-speed LP mode (32KHz)

3. 2. 1. 2 Unique Identification Code UID

96 bits UID is located at the information region INFO1 of the Flash memory in byte, which include 12 bytes and is readable by software. UID11~UID0 bytes description is as below.

Unique Identification Code（UID11~UID0）		
Address	$001001D7_H$ (UID11) ~ $001001D0_H$ (UID4), $001001C6_H$ (UID3), $001001C4_H$ (UID2) ~ $001001C2_H$ (UID0)	
UID11	Bits 95~88	UID11, byte address is $001001D7_H$
UID10	Bits 87~80	UID10, byte address is $001001D6_H$
UID9	Bits 79~72	UID9, byte address is $001001D5_H$
UID8	Bits 71~64	UID8, byte address is $001001D4_H$
UID7	Bits 63~56	UID7, byte address is $001001D3_H$
UID6	Bits 55~48	UID6, byte address is $001001D2_H$
UID5	Bits 47~40	UID5, byte address is $001001D1_H$
UID4	Bits 39~32	UID4, byte address is $001001D0_H$
UID3	Bits 31~24	UID3, byte address is $001001C6_H$
UID2	Bits 23~16	UID2, byte address is $001001C4_H$
UID1	Bits 15~8	UID1, byte address is $001001C3_H$
UID0	Bits 7~0	UID0, byte address is $001001C2_H$

3. 2. 2 Flash Program Memory

The 36KB Flash memory contains 36 pages and each page has 1K bytes, and the range is 0000_0000_H~0000_8FFF_H. The Flash supports at least 100K erase/write cycles and more than 10 years data retention.

Programming and page erase can be performed to the Flash memory through IAP. Programming one word takes about 20us and one page erasing takes about 2ms.

Programming, erase and read operation to the Flash are available in SWD mode. The SWD mode can be configured by the CFG_DEBUG and DEBUG_S of the configuration word CFG_WORD0.

3. 2. 3 In-Application Programming IAP

IAP (In-application Programming) capability is supported in Flash memory. With the write-protected area enabled, the memory location in the write-protected area does not support IAP erase or programming. The FWPEB bit of the CFG_WORD1 enables the write-protected area in Flash and the FWPS bits select the desired range of memory locations which require write protection.

3. 2. 3. 1 Overview

- ◇ Supports Flash data protection. Prior to IAP, disable the write-protection for the associated registers.
- ◇ Supports Flash erase all (only in SWD mode) and page erase.
- ◇ Word programming, each word containing 4 bytes
- ◇ During IAP operation, the global interrupt can be disabled. And it can also be enabled by copying the interrupt vector table and ISR to SRAM, and configuring the interrupt vector table re-mapping enable register SCU_TBLREMAPEN and interrupt vector table offset register SCU_TBLOFF to call the ISR in SRAM
- ◇ Upon IAP operation, IAP will automatically be locked up and enter the Flash protection state. And it needs to be unlocked when performing the next IAP operation.
- ◇ IAP code needs to be copied to and executed in SRAM. The result checking of the erase and programming to Flash is also done in the IAP code.
- ◇ The built-in ROM is available and can be called in IAP code, to reduce the IAP code size in SRAM.

3. 2. 3. 2 IAP Operations

- ◇ The IAP operation flow is as follows:
  - Set the FLASH_REQ =1 via the IAP control register and poll the FLASH_ACK until it is 1.
  - Proceed with the corresponding IAP operation, including Flash erase all, page erase and programming.
  - On the completion of the IAP operation, clear the FLASH_REQ, and poll the FLASH_ACK until it is 0.

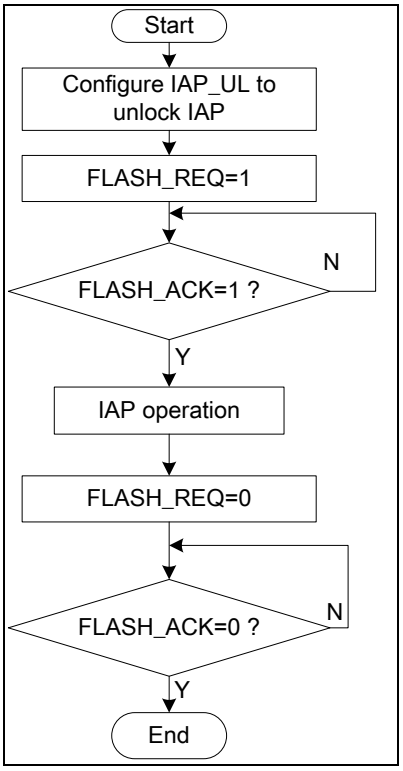


Figure 3-2 IAP Operation Request Flowchart

◇ IAP Erase All

The IAP erase all operation is only active in debug mode (the bit15 of CFG_WORD0 (0010_0004H) is 1).

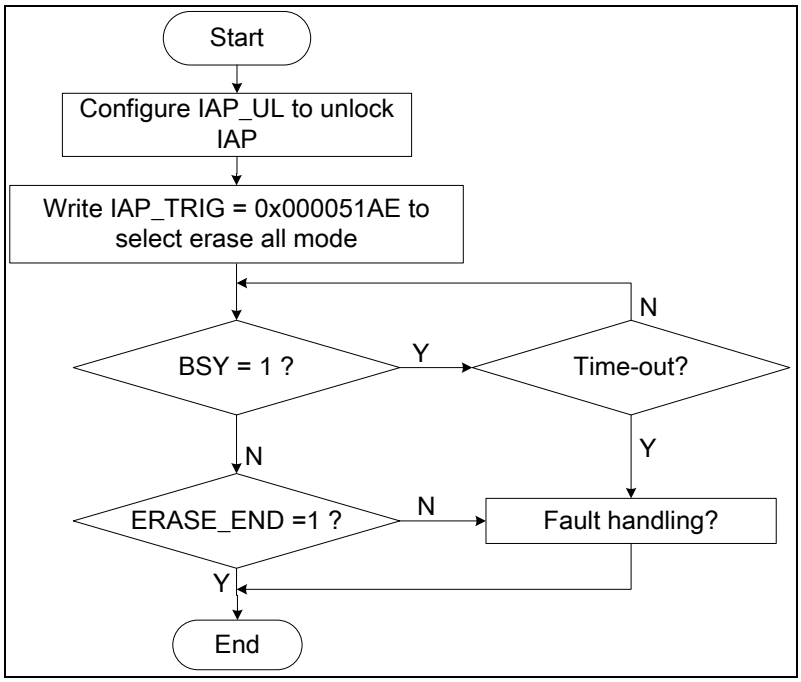


Figure 3-3 IAP Erase All Flowchart

◇ IAP Page Erase

The address of the target page can be set by the IAPPA register. When the erasing is done, the IAPPA is increased by 1.

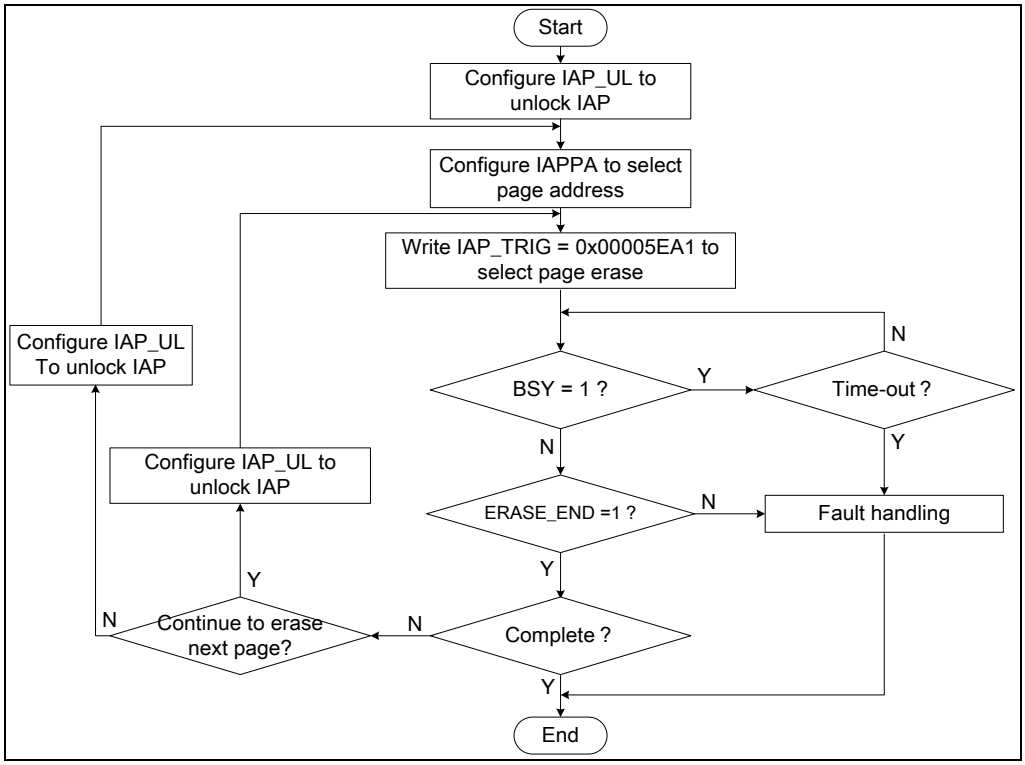


Figure 3-4 IAP Page Erase Flowchart

◇ IAP Programming

The address of the target memory location can be set by the IAPCA. The IAPCA will be automatically added by 1 after the target memory location is programmed. If words programming on consecutive memory addresses is used, there is no need to modify the IAPCA register. Because the IAPCA is only capable of addressing the memory location on the current page, the IAPCA needs to be re-written to specify the address when programming crossing pages.

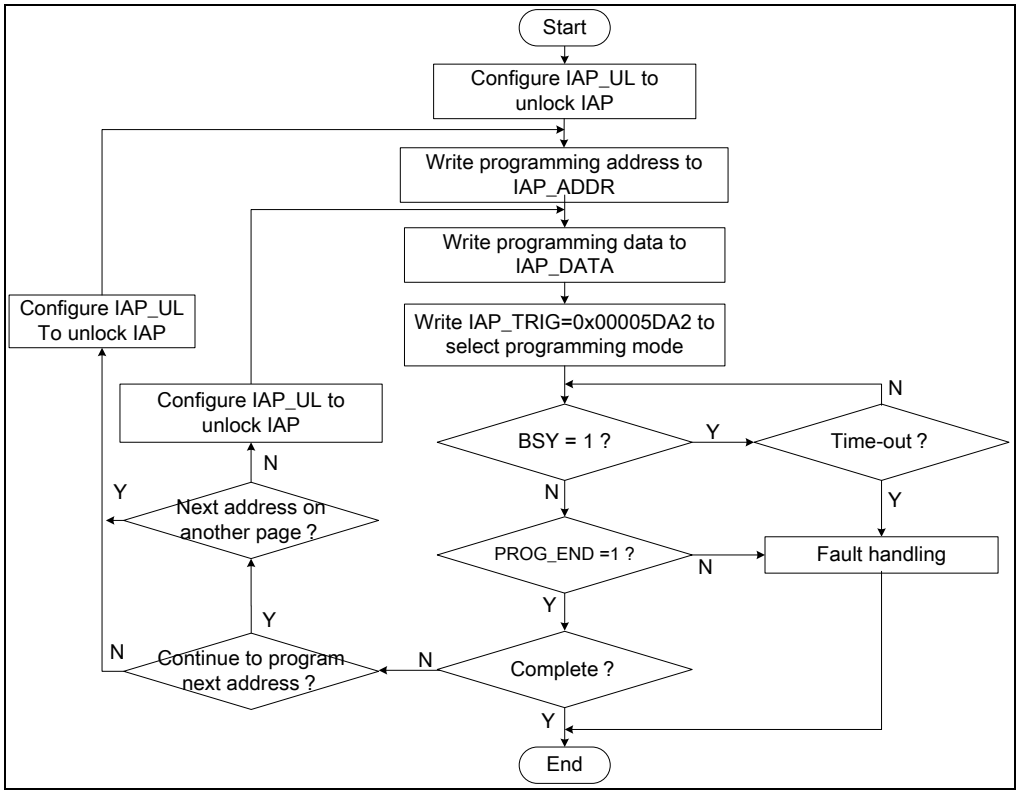


Figure 3-5 IAP Programming Flowchart

3. 2. 3. 3 IAP ROM

The built-in ROM is available and can be called in IAP code, to reduce the IAP code size in SRAM.

Page erase, word program and words program are supported in ROM, and each operation is accomplished by the following function.

◇ Page Erase Function (IAP_PageErase)

- Entry address: stored at 0x10000004
- Parameter input:R0- the first address of the page to be erased
- Parameter output:R0- function execution status (R0=1 succeeded, R0=0 failed)

◇ Word Program Function (IAP_WordProgram)

- Entry address: stored at 0x10000008
- Parameter input: R0- Flash address to be programmed, R1- data to be programmed
- Parameter output: R0- function execution status (R0=1 succeeded, R0=0 failed)

◇ Words Program Function (IAP_WordsProgram)

- Entry address: stored at 0x10000000
- Parameter input: R0- the first Flash address to be programmed, R1- the first address of data to be programmed in SRAM, R2- the length of data to be programmed, R3-whether or not to perform page erase when programming the first address of a page (if R3= non-zero, perform an erase operation; if R3=0, do not erase)

- Parameter output: R0- function execution status (R0=1 succeeded, R0=0 failed)

The word program function and words program function flowchart can be referred to Figure 3-5. For word program function, 'continue to program next address' can be ignored.

3. 2. 4 Special Function Registers

IAP Unlock Register (IAP_UL)

Offset address:10_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
UL<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
UL<15:0>															

UL<31:0>	bit31-0	R/W	IAP unlock: write 0x0000_00A5 to unlock IAP IAP lock: any of the following operations can lock IAP - Write other values to lock IAP - Write to IAP trigger register IAP_TRIG to cause IAP automatically to be locked - Write a reserved address to lock IAP - IAP will be locked up following a software reset

Notes

1. After IAP has been locked up, the IAP_CON, IAP_ADDR, IAP_DATA and IAP_TRIG registers are write-protected.
2. Reserved addresses are those undefined addresses in 40000800_H~$40000BFF_H$.

IAP Control Register (IAP_CON)

Offset address:00_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FLASH_FAIL		FLASH_ACK	FLASH_REQ	Reserved		RST	EN

—	bit31-8	—	—
FLASH_FAIL	bit7	R	**IAP accessing Flash fail flag bit** 0: protected region not accessed by IAP

			1: IAP has accessed the protected region and failed.
—	bit6	R/W	—
FLASH_ACK	bit5	R	**Flash ACK** 0: IAP accessing not allowed 1: IAP accessing Flash allowed
FLASH_REQ	bit4	R/W	**IAP accessing Flash request** 0: No request 1:IAP requests to access Flash
—	bit3-2	—	—
RST	bit1	W	**IAP software reset** 0: Always read as 0 1: Reset
EN	bit0	R/W	**IAP enable bit** 0: Disabled 1: Enabled

Note: Prior to writing to the IAP_CON register, set the IAP_UL register to unlock the IAP and remove the write-protection.

IAP Address Register (IAP_ADDR)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											IFREN	Reserved			

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
IAPPA<5:0>						IAPCA<7:0>								Reserved	

—	bit31-21	—	—
IFREN	bit20	R/W	**IAP information region enable bit** 0: IAP operation on Flash information region disabled 1: IAP operation on Flash information region enabled
—	bit19-16	—	—
IAPPA<5:0>	bit15-10	R/W	**IAP page address(erase mode)** 1) 0x00~0x23: total of 36 pages with corresponding address range 0x0000_0000~0x0000_8FFF 2) IAPPA is invalid on an operation on Flash information region
IAPCA<7:0>	bit9-2	R/W	**IAP address** 1) In erase mode, an address is not valid. 2) In programming mode - During an operation on non Flash information region, the IAPCA<7:0> is the corresponding address being

			programmed of the current page. Each page contains 256 addresses and each address has 4 bytes. The address must be erased before programming to it. - During an operation on Flash information region, only IAPCA [5:0] is valid. The information region contains 64 addresses and each address has 4 bytes.
—	bit1-0	—	—

Notes

1. Prior to writing to the IAP_ADDR register, set the IAP_UL register to unlock the IAP and remove the write-protection.
2. On completion of page erase, the IAPPA will automatically be added by 1.
3. On completion of address programming, the IAPCA will automatically be added by 1. Since the IAPCA only performs the addressing on the current page, the IAPCA needs to be rewritten to specify the next page address when programming crossing the pages.

IAP Data Register (IAP_DATA)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA<15:0>															

DATA<31:0>	bit31-0	R/W	IAP data

Note: Prior to writing to the IAP_DATA register, set the IAP_UL register to unlock the IAP and remove the write-protection.

Trigger Register (IAP_TRIG)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TRIG<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TRIG<15:0>															

TRIG<31:0>	bit31-0	R/W	**IAP operation instruction (After the register is written, the IAP will be relocked up)** 0x0000_51AE: Erase all (only for SWD)

			0x0000_5EA1: Page erase (An erase all operation must be performed before erasing the information region. Otherwise, it is invalid) 0x0000_5DA2: Programming Others: No operation (On completion of IAP operation, hardware will alter the register to no operation.)

Note: Prior to writing to the IAP_TRIG register, set the IAP_UL register to unlock the IAP and remove the write-protection.

IAP Status Register (IAP_STA)

Offset address:14_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												TIMEOUT_ERR	PROG_END	ERASE_END	BSY

—	bit31-4	—	—
TIMEOUT_ERR	bit3	R/W	**IAP time-out error flag bit** 0: This flag can be cleared by writing 0, triggering IAP_TRIG, or IAP software reset. 1: Automatically set by hardware when an IAP operation has timed out
PROG_END	bit2	R/W	**IAP programming end flag bit** 0: This flag can be cleared by writing 0, triggering IAP_TRIG, or IAP software reset. 1: Automatically set by hardware when the current address programming has completed.
ERASE_END	bit1	R/W	**IAP page erase end flag bit** 0: This flag can be cleared by writing 0, triggering IAP_TRIG, or IAP software reset. 1: Automatically set by hardware when the current page has been erased
BSY	bit0	R	**IAP busy bit** 0: Idle. The IAP software reset can clear the bit. 1: An IAP operation is in progress.

Note: When the TIMEOUT_ERR bit of the IAP_STA register is 1, hardware will automatically clear the EN bit of the IAP_CON register.

3. 3 Data Memory (SRAM)

The integrated 8KB data memory SRAM has the address range 2000_0000_H~2000_1FFF_H.

3. 3. 1 SRAM Map

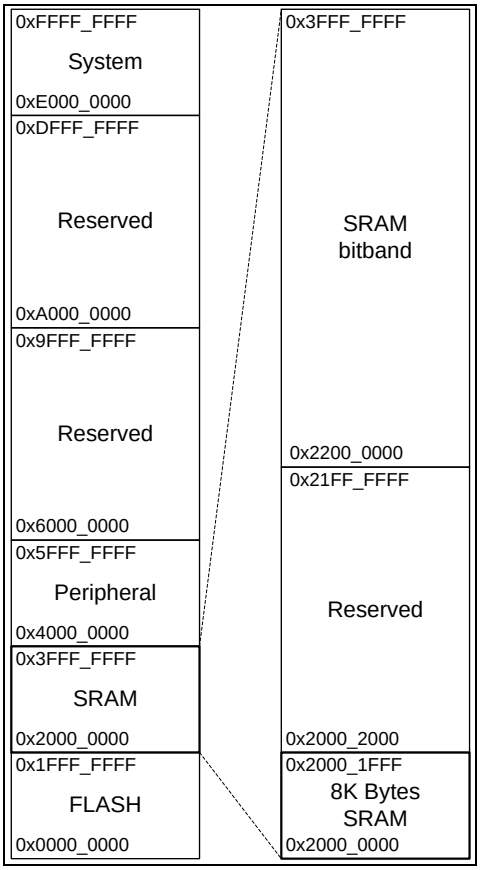


Figure 3-6 SRAM Map

3. 3. 2 SRAM Bit Banding

Bit banding is available in SRAM, and a bit can be written and read using load and store instructions. With bit banding, besides from the space with the starting address 0x2000_0000, the SRAM can also be accessed by a bit from the bit-band region with the starting address 0x2200_0000.

The bit-band region maps each word in an alias region to a bit in the bit-band region. Accessing a word in the alias region has the same effect as accessing the targeted bit in the bit-band region.

The following example shows how to map bit N (0≤N≤7) of the byte located at SRAM address A to the alias region,

AliasAddress_A_N = 0x2200_0000 + (A – 0x2000_0000) x 32 + N x 4

3. 4 Peripheral Registers

3. 4. 1 Peripheral Registers Map

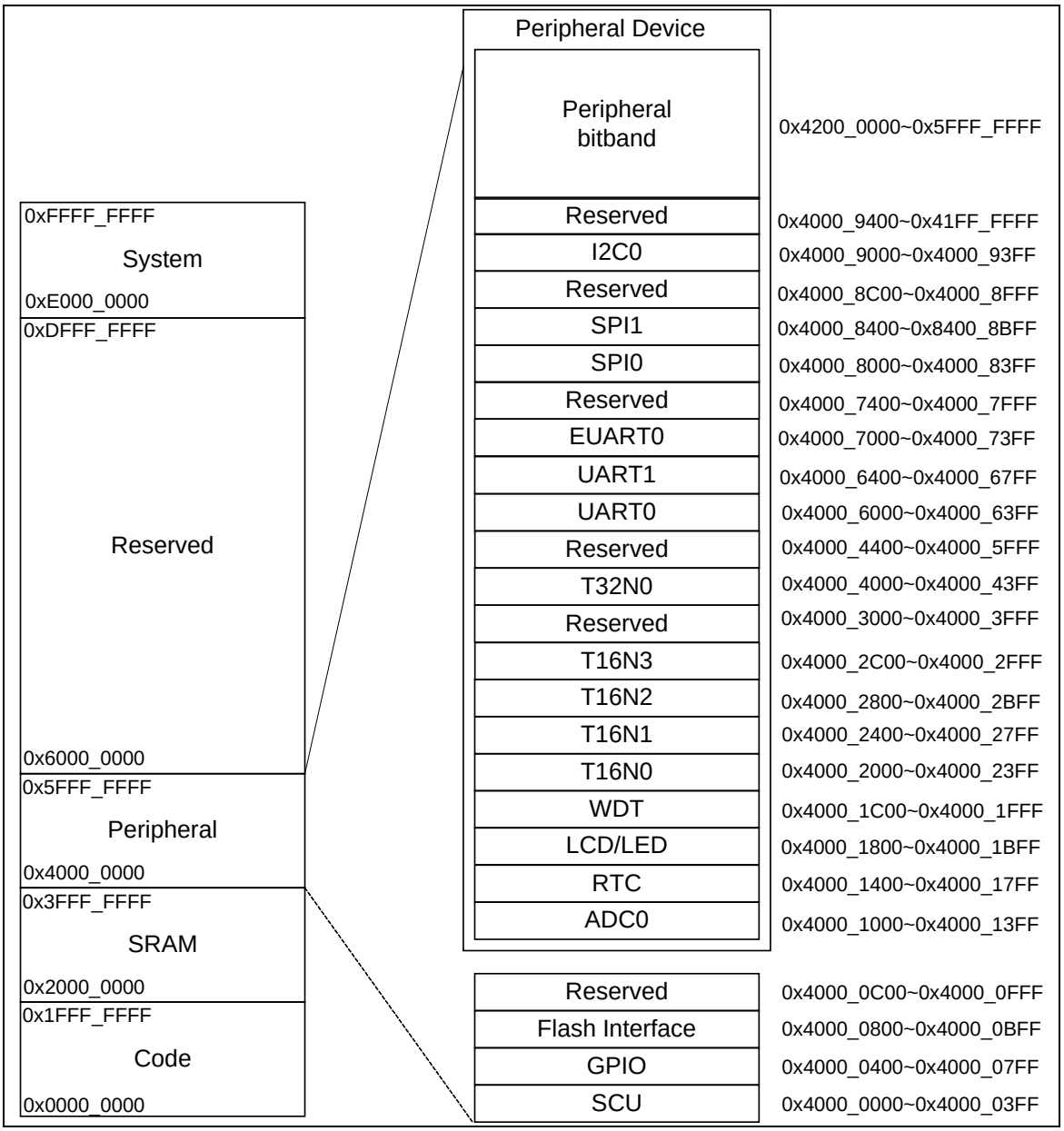


Figure 3-7 Peripheral Registers Map

Note: Reserved regions are read-only, and read as 00000000_H.

3. 4. 2 Peripheral Registers Bit Banding

Bit banding is available in peripheral registers, and a bit can be written and read using load and store instructions. With bit banding, besides from the space with the starting address 0x4000_0000, the peripheral registers can also be accessed by a bit from the bit-band region with the starting address 0x4200_0000.

The bit-band region maps each word in an alias region to a bit in the bit-band region. Accessing a word in the alias region has the same effect as accessing the targeted bit in

the bit-band region.

The following example shows how to map bit N (0≤N≤7) of a peripheral register located at address A to the alias region,

AliasAddress_A_N = 0x4200_0000 + (A – 0x4000_0000) x 32 + N x 4

The GPIO port registers, GPIO_PADATABSR, GPIO_PADATABCR, GPIO_PADATABR, GPIO_PADIRBSR, GPIO_PADIRBCR, GPIO_PADIRBRR, GPIO_PBDATABSR, GPIO_PBDATABCR, GPIO_PBDATABRRl, GPIO_PBDIRBSR, GPIO_PBDIRBCR and GPIO_PBDIRBRR, are used to implement the bit operation of GPIO port data registers and direction control registers, and bit banding is not supported by these registers. Apart from these registers, other peripheral registers are all with bit banding feature.

3. 4. 3 List of System Control Unit (SCU) Registers

System Control Unit (SCU)		
Register Name	**Offset Address**	**Descriptions**
SCU base address:4000_0000$_H$		
SCU_PROT	0000$_H$	System configuration protection register
SCU_NMICON	0004$_H$	Non-maskable interrupt control register
SCU_PWRC	0008$_H$	Reset register
SCU_FAULTFLAG	000C$_H$	Hard fault flag register
SCU_FLASHWAIT	0020$_H$	Flash access wait register
SCU_SOFTCFG	0024$_H$	System software configuration register
SCU_LVDCON	0028$_H$	LVD control register
SCU_CCM	002C$_H$	External clock check management register
SCU_PLLLKCON	0030$_H$	PLL lock interrupt control register
SCU_TIMEREN	0034$_H$	Timer enable register
SCU_TIMERDIS	0038$_H$	Timer disable register
SCU_SCLKEN0	0040$_H$	System clock enable register0
SCU_SCLKEN1	0044$_H$	System clock enable register1
SCU_PCLKEN	0048$_H$	Peripheral clock enable register
SCU_WAKEUPTIME	004C$_H$	System wake-up time control register
SCU_TBLREMAPEN	0060$_H$	Interrupt vector table re-map enable register
SCU_TBLOFF	0064$_H$	Interrupt vector table offset register

3. 4. 4 List of GPIO Registers

GPIO Registers		
Register Name	**Offset Address**	**Description**
GPIO base address:4000_0400$_H$		
GPIO_PAPORT	0000$_H$	PA port status register
GPIO_PADATA	0004$_H$	PA output data register
GPIO_PADATABSR	0008$_H$	PA output data bit set register

GPIO Registers		
Register Name	**Offset Address**	**Description**
GPIO_PADATABCR	$000C_H$	PA output data bit clear register
GPIO_PADATABRR	0010_H	PA output data bit invert register
GPIO_PADIR	0014_H	PA direction register
GPIO_PADIRBSR	0018_H	PA direction bit set register
GPIO_PADIRBCR	$001C_H$	PA direction bit clear register
GPIO_PADIRBRR	0020_H	PA direction bit invert register
GPIO_PAFUNC0	0024_H	PA[7:0] function select register
GPIO_PAFUNC1	0028_H	PA[15:8] function select register
GPIO_PAFUNC2	$002C_H$	PA[23:16] function select register
GPIO_PAFUNC3	0030_H	PA[31:24] function select register
GPIO_PAINEB	0034_H	PA input enable register
GPIO_PAODE	0038_H	PA open-drain enable register
GPIO_PAPUE	$003C_H$	PA weak pull-up enable register
GPIO_PAPDE	0040_H	PA weak pull-down enable register
GPIO_PADS	0044_H	PA drive strength control register
GPIO_PBPORT	0080_H	PB port status register
GPIO_PBDATA	0084_H	PB output data register
GPIO_PBDATABSR	0088_H	PB output data bit set register
GPIO_PBDATABCR	$008C_H$	PB output data bit clear register
GPIO_PBDATABRR	0090_H	PB output data bit invert register
GPIO_PBDIR	0094_H	PB direction register
GPIO_PBDIRBSR	0098_H	PB direction bit set register
GPIO_PBDIRBCR	$009C_H$	PB direction bit clear register
GPIO_PBDIRBRR	$00A0_H$	PB direction bit invert register
GPIO_PBFUNC0	$00A4_H$	PB[7:0] function select register
GPIO_PBFUNC1	$00A8_H$	PB[13:8] function select register
GPIO_PBINEB	$00B4_H$	PB input enable register
GPIO_PBODE	$00B8_H$	PB open-drain enable register
GPIO_PBPUE	$00BC_H$	PB weak pull-up enable register
GPIO_PBPDE	$00C0_H$	PB weak pull-down enable register
GPIO_PBDS	$00C4_H$	PB drive strength control register
GPIO_PINTIE	0300_H	PINT interrupt enable register
GPIO_PINTIF	0304_H	PINT interrupt flag register
GPIO_PINTSEL	0308_H	PINT interrupt source select register
GPIO_PINTCFG	$030C_H$	PINT interrupt configuration register
GPIO_KINTIE	0310_H	KINT interrupt enable register
GPIO_KINTIF	0314_H	KINT interrupt flag register
GPIO_KINTSEL	0318_H	KINT interrupt source select register
GPIO_KINTCFG	$031C_H$	KINT interrupt configuration register
GPIO_IOINTFLTS	0330_H	Port interrupt 20ns filter select register
GPIO_TMRFLTSEL	0340_H	TMR input port 20ns filter select register

GPIO Registers		
Register Name	**Offset Address**	**Description**
GPIO_SPIFLTSEL	0344_H	SPI input port 20ns filter select register
GPIO_TXPWM	0380_H	PWM register
GPIO_BUZC	0384_H	Buzz counter register

3. 4. 5 List of IAP Registers

IAP Registers		
Register Name	**Offset Address**	**Descriptions**
IAP base address:4000_0800_H		
IAP_CON	0000_H	IAP control register
IAP_ADDR	0004_H	IAP address register
IAP_DATA	0008_H	IAP data register
IAP_TRIG	$000C_H$	IAP trigger register
IAP_UL	0010_H	IAP unlock register
IAP_STA	0014_H	IAP status register

3. 4. 6 List of ADC Registers

ADC Registers		
Register Name	**Offset Address**	**Description**
ADC base address:4000_1000_H		
ADC_DR	0000_H	ADC result register
ADC_CON0	0004_H	ADC control register0
ADC_CON1	0008_H	ADC control register1
ADC_CHS	$000C_H$	ADC channel select register
ADC_IE	0010_H	ADC interrupt enable register
ADC_IF	0014_H	ADC interrupt flag register
ADC_ACPC	0028_H	ADC auto conversion Compare control register
ADC_ACPCMP	0030_H	ADC auto conversion compare threshold register
ADC_ACPMEAN	0034_H	ADC mean register
ADC_VREFCON	0040_H	ADC voltage reference control register

3. 4. 7 List of RTC Registers

RTC Registers		
Register Name	**Offset Address**	**Description**
RTC base address:4000_1400_H		
RTC_CON	0000_H	RTC control register
RTC_CAL	0004_H	RTC calibration register
RTC_WA	0008_H	RTC week alarm register

RTC Registers		
Register Name	**Offset Address**	**Description**
RTC_DA	$000C_H$	RTC day alarm register
RTC_HMS	0010_H	RTC hour minute second register
RTC_YMDW	0014_H	RTC year month day week register
RTC_IE	0018_H	RTC interrupt enable register
RTC_IF	$001C_H$	RTC interrupt flag register
RTC_WP	0020_H	RTC write-protection register

3. 4. 8 List of LCDC Registers

LCD Registers		
Register Name	**Offset Address**	**Description**
LCD:4000_1800_H		
LCD_CON0	0000_H	LCD control register0
LCD_TWI	0004_H	LCD twinkle register
LCD_SEL	0008_H	LCD segment enable register <27:0>
LCD_CON1	0010_H	LCD control register1
LCD_D0	0020_H	LCD data register0
LCD_D1	0024_H	LCD data register1
LCD_D2	0028_H	LCD data register2
LCD_D3	$002C_H$	LCD data register3
LCD_D4	0030_H	LCD data register4
LCD_D5	0034_H	LCD data register5
LCD_D6	0038_H	LCD data register 6

Note: The registers LCD_CON0 and LED_CON0, LCD_SEL and LED_SEL, LCD_CON1 and LED_CON1, LCD_D0 and LED_D0, and LCD_D1 and LED_D1 of the LCD and LED module share the same addresses.

3. 4. 9 List of LEDC Registers

LED Registers		
Register Name	**Offset Address**	**Description**
LED base address:4000_1800_H		
LED_CON0	0000_H	LED control register0
LED_SEL	0008_H	LED segment enable register<27:0>
LED_CON1	0010_H	LED control register1
LED_D0	0020_H	LED data register0
LED_D1	0024_H	LED data register1

Note: The registers LCD_CON0 and LED_CON0, LCD_SEL and LED_SEL, LCD_CON1 and LED_CON1, LCD_D0 and LED_D0, and LCD_D1 and LED_D1 of the LCD and LED module share the same addresses.

3. 4. 10 List of WDT Registers

WDT Registers		
Register Name	**Offset Address**	**Description**
WDT base address:4000_1C00$_H$		
WDT_LOAD	0000$_H$	WDT reload value register
WDT_VALUE	0004$_H$	WDT current value register
WDT_CON	0008$_H$	WDT control register
WDT_INTCLR	000C$_H$	WDT interrupt flag clear register
WDT_RIS	0010$_H$	WDT interrupt flag register
WDT_LOCK	0100$_H$	WDT lock register
WDT_ITCR	0300$_H$	WDT test register, for test only
WDT_ITOP	0304$_H$	WDT test register, for test only

Note: The WDT_ITCR and WDT_ITOP registers are for test only, and users are prohibited to write to them. Otherwise, it would lead to irregular behavior of the device.

3. 4. 11 List of T16N0/T16N1/T16N2/T16N3 Registers

T16N Registers		
Register Name	**Offset Address**	**Description**
T16N0 base address:4000_2000$_H$ T16N1 base address:4000_2400$_H$ T16N2 base address:4000_2800$_H$ T16N3 base address:4000_2C00$_H$		
T16N_CNT0	0000$_H$	T16N counter register0
T16N_CNT1	0004$_H$	T16N counter register1
T16N_PRECNT	0008$_H$	T16N prescaler counter register
T16N_PREMAT	000C$_H$	T16N prescaler counter match register
T16N_CON0	0010$_H$	T16N control register0
T16N_CON1	0014$_H$	T16N control register1
T16N_CON2	0018$_H$	T16N control register2
T16N_IE	0020$_H$	T16N interrupt enable register
T16N_IF	0024$_H$	T16N interrupt flag register
T16N_PDZ	0028$_H$	T16N PWM dead zone register
T16N_PTR	002C$_H$	T16N PWM trigger ADC register
T16N_MAT0	0030$_H$	T16N counter match register0
T16N_MAT1	0034$_H$	T16N counter match register1
T16N_MAT2	0038$_H$	T16N counter match register2
T16N_MAT3	003C$_H$	T16N counter match register3
T16N_TOP0	0040$_H$	T16NCNT0 top value register0
T16N_TOP1	0044$_H$	T16NCNT1 top value register1

3. 4. 12 List of T32N0 Registers

T32N Registers		
Register Name	**Offset Address**	**Description**
T32N0 base address:4000_4000$_H$		
T32N_CNT	0000$_H$	T32N counter register
T32N_CON0	0004$_H$	T32N control register0
T32N_CON1	0008$_H$	T32N control register1
T32N_PRECNT	0010$_H$	T32N prescaler counter register
T32N_PREMAT	0014$_H$	T32N prescaler counter match register
T32N_IE	0018$_H$	T32N interrupt enable register
T32N_IF	001C$_H$	T32N interrupt flag register
T32N_MAT0	0020$_H$	T32N counter match register0
T32N_MAT1	0024$_H$	T32N counter match register1
T32N_MAT2	0028$_H$	T32N counter match register2
T32N_MAT3	002C$_H$	T32N counter match register3

3. 4. 13 List of UART0/UART1 Registers

UART Registers		
Register Name	**Offset Address**	**Description**
UART0 base address:4000_6000$_H$ UART1 base address:4000_6400$_H$		
UART_CON0	0000$_H$	UART control register0
UART_CON1	0004$_H$	UART control register1
UART_BRR	0010$_H$	UART baud rate register
UART_STA	0014$_H$	UART status register
UART_IE	0018$_H$	UART interrupt enable register
UART_IF	001C$_H$	UART interrupt flag register
UART_TBW	0020$_H$	UART transmit data write register
UART_RBR	0024$_H$	UART receive data read register
UART_TB0	0040$_H$	UART transmit buffer register0
UART_TB1	0044$_H$	UART transmit buffer register1
UART_TB2	0048$_H$	UART transmit buffer register2
UART_TB3	004C$_H$	UART transmit buffer register3
UART_TB4	0050$_H$	UART transmit buffer register4
UART_TB5	0054$_H$	UART transmit buffer register5
UART_TB6	0058$_H$	UART transmit buffer register6
UART_TB7	005C$_H$	UART transmit buffer register7
UART_RB0	0060$_H$	UART receive buffer register0
UART_RB1	0064$_H$	UART receive buffer register1
UART_RB2	0068$_H$	UART receive buffer register2
UART_RB3	006C$_H$	UART receive buffer register3
UART_RB4	0070$_H$	UART receive buffer register4

UART Registers		
Register Name	**Offset Address**	**Description**
UART_RB5	0074_H	UART receive buffer register5
UART_RB6	0078_H	UART receive buffer register6
UART_RB7	$007C_H$	UART receive buffer register7

3. 4. 14 List of EUART0 Registers

EUART Registers		
Register Name	**Offset Address**	**Description**
EUART0 base address:4000_7000_H		
EUART_CON0	0000_H	EUART control register0
EUART_CON1	0004_H	EUART control register1
EUART_CON2	0008_H	EUART control register2
EUART_BRR	0010_H	EUART baud rate register
EUART_IE	0018_H	EUART interrupt enable register
EUART_IF	$001C_H$	EUART interrupt flag register
EUART_TBW	0020_H	EUART transmit data write register
EUART_RBR	0024_H	EUART receive data read register
EUART_TB01	0040_H	EUART transmit buffer register0/1
EUART_TB23	0044_H	EUART transmit buffer register2/3
EUART_RB01	0048_H	EUART receive buffer register0/1
EUART_RB23	$004C_H$	EUART receive buffer register2/3

3. 4. 15 List of SPI0/ SPI1 Registers

SPI Registers		
Register Name	**Offset Address**	**Description**
SPI0 base address:4000_8000_H SPI1 base address:4000_8400_H		
SPI_CON	0000_H	SPI control register
SPI_TBW	0008_H	SPI transmit data write register
SPI_RBR	$000C_H$	SPI receive data read register
SPI_IE	0010_H	SPI interrupt enable register
SPI_IF	0014_H	SPI interrupt flag register
SPI_TB	0018_H	SPI transmit buffer register
SPI_RB	$001C_H$	SPI receive buffer register
SPI_STA	0020_H	SPI status register
SPI_CKS	0024_H	SPI baud rate register

3. 4. 16 List of I2C0 Registers

I2C Registers		
Register Name	**Offset Address**	**Description**
I2C0 base address:4000_9000$_H$		
I2C_CON	0000$_H$	I2C control register
I2C_MOD	0004$_H$	I2C operating mode register
I2C_IE	0008$_H$	I2C interrupt enable register
I2C_IF	000C$_H$	I2C interrupt flag register
I2C_TBW	0010$_H$	I2C transmit data write register
I2C_RBR	0014$_H$	I2C receive data read register
I2C_TB	0018$_H$	I2C transmit buffer register
I2C_RB	001C$_H$	I2C receive buffer register
I2C_STA	0020$_H$	I2C status register

3. 5 Core Registers

3. 5. 1 List of SysTick Timer Registers

SysTick Registers		
Register Name	**Offset Address**	**Description**
SYSTICK base address:E000_E000$_H$		
SYST_CSR	0010$_H$	SysTick control/status register
SYST_RVR	0014$_H$	SysTick reload value register
SYST_CVR	0018$_H$	SysTick current value register
SYST_CALIB	001C$_H$	SysTick calibration value register

3. 5. 2 List of Nested Vectored Interrupt Controller Registers (NVIC)

Nested Vectored Controller Register (NVIC)		
Register Name	**Offset Address**	**Description**
NVIC base address:E000_E100$_H$		
NVIC_ISER	0000$_H$	IRQ0~31 interrupt set enable register
NVIC_ICER	0080$_H$	IRQ0~31 interrupt clear enable register
NVIC_ISPR	0100$_H$	IRQ0~31 interrupt set pending register
NVIC_ICPR	0180$_H$	IRQ0~31 interrupt clear pending register
NVIC_PR0	0300$_H$	IRQ0~3 priority control register
NVIC_PR1	0304$_H$	IRQ4~7 priority control register
NVIC_PR2	0308$_H$	IRQ8~11 priority control register
NVIC_PR3	030C$_H$	IRQ12~15 priority control register
NVIC_PR4	0310$_H$	IRQ16~19 priority control register
NVIC_PR5	0314$_H$	IRQ20~23 priority control register
NVIC_PR6	0318$_H$	IRQ24~27 priority control register
NVIC_PR7	031C$_H$	IRQ28~31 priority control register

3. 5. 3 List of System Control Block (SCB) Registers

System Control Block (SCB)		
Register Name	**Offset Address**	**Description**
SCB base address:E000_ED00$_H$		
SCB_CPUID	0000$_H$	SCB_CPUID register
SCB_ICSR	0004$_H$	Interrupt control and status register
SCB_AIRCR	000C$_H$	Application interrupt and control register
SCB_SCR	0010$_H$	System control register
SCB_CCR	0014$_H$	Configuration and control register
SCB_SHPR2	001C$_H$	System handler priority register2
SCB_SHPR3	0020$_H$	System handler priority register3

Chapter4 General Purpose IOs (GPIO)

4. 1 Overview

There are two groups of GPIOs available, total up to 46 I/O pins.

All I/Os are CMOS Schmitt inputs and CMOS drive outputs with configurable open-drain feature, and the multiplexing and operating modes of each I/O are configured by port function register GPIO_PAFUNC/GPIO_PBFUNC.

When an I/O port is configured for digital mode, its output mode is controlled by the port direction control register GPIO_PADIR/ GPIO_PBDIR and the input mode is controlled by the corresponding input control register GPIO_PAINEB/ GPIO_PBINEB. When an I/O port is in output mode, its port level is determined by the port data register GPIO_PADATA/ GPIO_PBDATA, where 1 indicates high level and 0 indicates low level. When an I/O port is in input mode, its port level can be acquired by reading the corresponding port status register GPIO_PAPORT/ GPIO_PBPORT.

Bitwise operation is supported for port output levels. Setting a bit of the GPIO output set register GPIO_PADATABSR/ GPIO_PBDATABSR to 1 allows the corresponding GPIO to output high level. Setting a bit of the GPIO output clear register GPIO_PADATABCR / GPIO_PBDATABCR to 1 allows the corresponding GPIO to output low level. Setting a bit of the GPIO output invert register GPIO_PADATABRR/ GPIO_PBDATABRR to 1 allows the corresponding GPIO to invert the output.

Bitwise operation is also supported for port direction control. When a bit of the GPIO direction set register GPIO_PADIRBSR/ GPIO_PBDIRBSR is 1, the corresponding GPIO works in input mode. When a bit of the GPIO direction clear register GPIO_PADIRBCR/ GPIO_PBDIRBCR is 1, the corresponding GPIO works in output mode. When a bit of the GPIO direction bit invert register GPIO_PADIRBRR/ GPIO_PBDIRBRR is 1, the direction of the corresponding GPIO is reversed.

When an I/O is configured for the peripheral function, its output mode is still configured by the port direction control register GPIO_PADIR/ GPIO_PBDIR and input mode by the corresponding GPIO_PAINEB/ GPIO_PBINEB register.

Each I/O port is designed with open-drain feature, which is enabled by the corresponding port open-drain enable register GPIO_PAODE/ GPIO_PBODE.

Each I/O port can be individually configured for weak pull-up or weak pull-down, respectively enabled by the corresponding GPIO_PAPUE/ GPIO_PBPUE and GPIO_PAPDE/ GPIO_PBPDE registers. Note that the weak pull-up and pull-down function cannot be enabled at the same time.

The port drive strength control register GPIO_PADS/ GPIO_PBDS selects the drive capability, high drive and normal drive, for each I/O port. The PA6~PA13 ports can be multiplexed with the COM7~COM0 of LCDC/LEDC modules, and they have much stronger drive capability when they are configured as high drive I/Os. See Appendix2 Electrical

Characteristics for details.

4. 2 Block Diagram

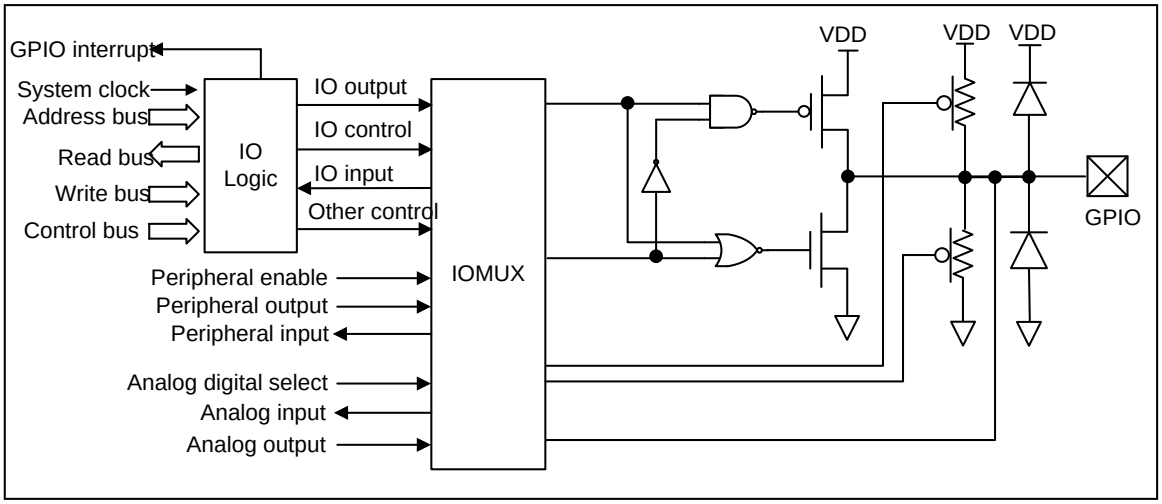


Figure 4-1 IO Port Block Diagram

4. 3 External Port Interrupt

All ports have external interrupt capability. All the interrupts are divided into 8 groups and each group corresponds to one IRQ. The PINT7CFG~PINT0CFG bit of the GPIO_PINTCFG register configures the trigger event for each external interrupt. If the high level or low level is configured as a trigger event, the interrupt flag can only be cleared by writing 1 to the corresponding bit of the GPIO_PINTIF register after the triggering level has been toggled.

The input mask enable bit PMASK of the GPIO_PINTIE register can mask the external interrupt input. If the interrupt input source is masked, the corresponding interrupt will not occur and the flag will not be set.

The external interrupt enable bit PINTIE of the GPIO_PINTIE register enables an interrupt request for each external interrupt flag PINTIF.

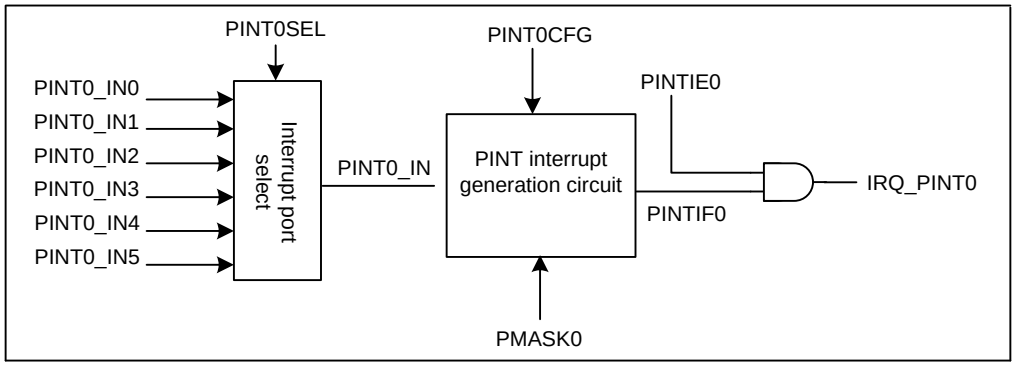


Figure 4-2 External Port PINT0 Interrupt Block Diagram

The figure above shows an example of the PINT0 interrupt circuit, with 6 external interrupt sources respectively from PA0, PA8, PA16, PA24, PB0 and PB8. The table below lists the external interrupt sources for PINT0~PINT7.

PINT	SEL0	SEL1	SEL2	SEL3	SEL4	SEL5
PINT0	PA0	PA8	PA16	PA24	PB0	PB8
PINT1	PA1	PA9	PA17	PA25	PB1	PB9
PINT2	PA2	PA10	PA18	PA26	PB2	PB10
PINT3	PA3	PA11	PA19	PA27	PB3	PB11
PINT4	PA4	PA12	PA20	PA28	PB4	PB12
PINT5	PA5	PA13	PA21	PA29	PB5	PB13
PINT6	PA6	PA14	PA22	PA30	PB6	-
PINT7	PA7	PA15	PA23	PA31	PB7	-

Table 4-1 PINT Selection Table

4. 4 External Key Interrupt

The device offers one external key interrupt KINT with eight key inputs KINT0~KINT7. Any key input can trigger the key interrupt. A key input source can be selected from one of the 6 I/O ports.

The input mask enable bit KMASK can mask the input source KINT. When the KMASK bit is enabled, the corresponding flag KINTIF will remain unchanged regardless of the status of the KINT_IN. The interrupt configuration register GPIO_KINTCFG selects the valid triggering edge or level for each KINT.

The key interrupt enable bit KINTIE enables an interrupt request for each key interrupt flag KINTIF.

The key interrupt configuration bit KINT7CFG~KINT0CFG of the GPIO_KINTCFG register configures the trigger event for each input source. When switching the trigger event, first mask the input source to avoid unnecessary interrupts. Alternatively, disable the key interrupt and enable it again after the switching has been done and the interrupt flag has been cleared.

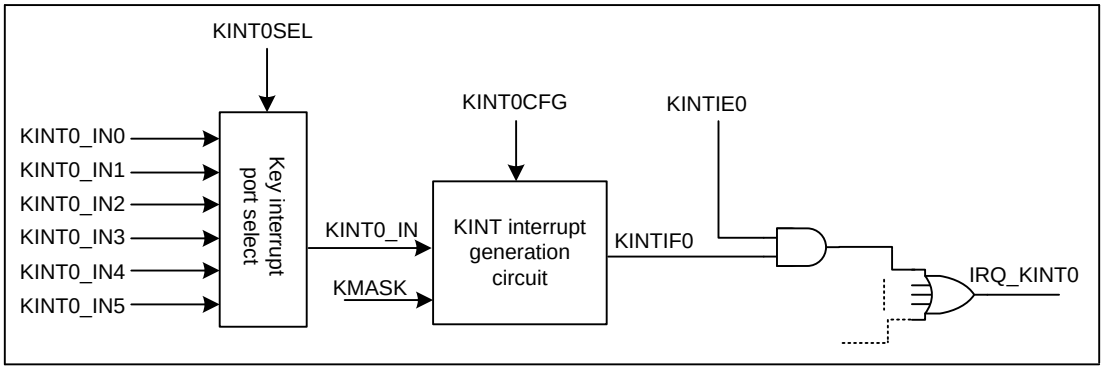


Figure 4-3 External Key KINT0 Interrupt Block Diagram

The figure above shows an example of the KINT0 interrupt circuit, with 6 external interrupt sources respectively from PA0, PA8, PA16, PA24, PB0 and PB8. The table below lists the external interrupt sources for KINT0~KINT7.

KINT	SEL0	SEL1	SEL2	SEL3	SEL4	SEL5
KINT0	PA0	PA8	PA16	PA24	PB0	PB8
KINT1	PA1	PA9	PA17	PA25	PB1	PB9
KINT2	PA2	PA10	PA18	PA26	PB2	PB10
KINT3	PA3	PA11	PA19	PA27	PB3	PB11
KINT4	PA4	PA12	PA20	PA28	PB4	PB12
KINT5	PA5	PA13	PA21	PA29	PB5	PB13
KINT6	PA6	PA14	PA22	PA30	PB6	-
KINT7	PA7	PA15	PA23	PA31	PB7	-

Table 4-2 KINT Selection Table

4. 5 Buzz Output

Buzz output is used for audio devices such as buzzer.

The control register GPIO_BUZC is used to enable the buzz and set the desired frequency of the output signal. The frequency of a buzz signal is expressed as

$$F_{BUZ} = \frac{Fpclk}{2 \times (BUZ\_LOAD + 1)}$$

The GPIO_PAFUNC0 and GPIO_PBFUNC1 registers select a port, PA0, PA1 or PB13, for the buzz signal to output to.

A buzz output of fixed frequency can also be modulated to a TXPWM0 signal by TXD0 signal output from the UART0, or modulated to a TXPWM1 signal by E0TX0 signal output from the EUART0, and sent to the corresponding output port. The control bit TX0PLV and TX1PLV can select a high level modulation or low level modulation.

When the TX0PS control bits = 11, the TXPWM0 signal can be output to TXD0, T16N0OUT0, T16N0OUT1 or BUZ pin, selected by the TX0_S3~TX0_S0 of the GPIO_TXPWM register.

When the TX1PS control bit = 11, the TXPWM1 signal can be output to E0TX0, T16N1OUT0, T16N1OUT1 or BUZ pin, selected by the TX1_S3~ TX1_S1 of the GPIO_TXPWM register.

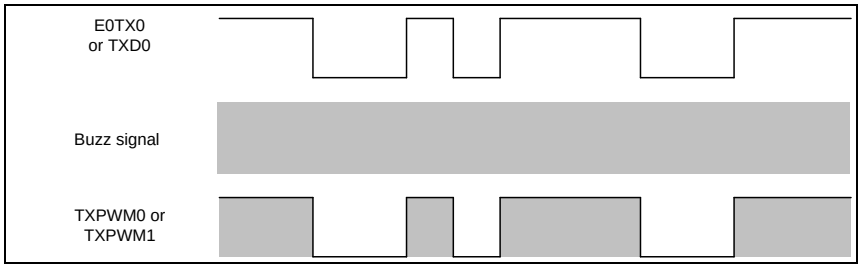


Figure 4-4 High Level Modulated Buzz Output Waveform

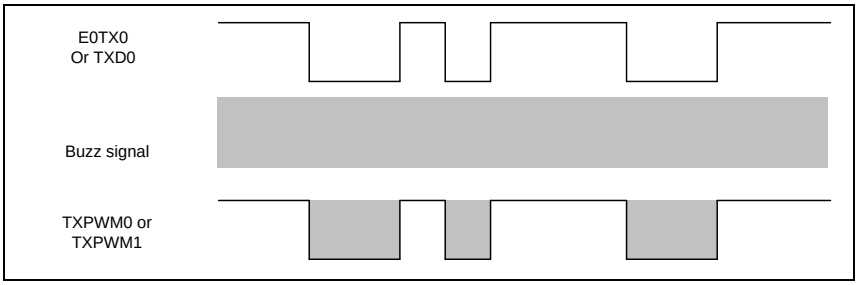


Figure 4-5 Low Level Modulated Buzz Output Waveform

4. 6 Special Function Registers

PA Port Status Register (GPIO_PAPORT)

Offset address:00_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PORT<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PORT<15:0>															

PORT<31:0>	bit 31-0	R	**PA port level** 0: Low 1: High

PA Output Data Register (GPIO_PADATA)

Offset address:04_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATA<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATA<15:0>															

DATA<31:0>	bit 31-0	R/W	**PA output data** 0: Output low 1: Output high

PA Output Data Bit Set Register (GPIO_PADATABSR)

Offset address:08_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATABSR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATABSR<15:0>															

DATABSR<31:0>	bit 31-0	W	**PA output data bit set** 0: Stay unchanged 1: Output high level on the corresponding pin

PA Output Data Bit Clear Register (GPIO_PADATABCR)

Offset address: $0C_H$

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATABCR <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATABCR <15:0>															

DATABCR<31:0>	bit 31-0	W	**PA output data bit clear** 0: Stay unchanged 1: Output low level on the corresponding pin

PA Output Data Bit Invert Register (GPIO_PADATABRR)

Offset address: 10_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DATABRR <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DATABRR <15:0>															

DATABRR<31:0>	bit 31-0	W	**PA output data bit invert** 0: Stay unchanged 1: Invert the output on the corresponding pin

PA Direction Register (GPIO_PADIR)

Offset address: 14_H

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIR <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIR <15:0>															

DIR<31:0>	bit0 31-0	R/W	**PA direction control bit** 0: Output 1: Non-output (If the corresponding bit of GPIO_PAINEB is 0, it is used as digital input. If the analog channel needs to be enabled, the corresponding bit of GPIO_PAINEB and GPIO_PADIR must be 1 to disable the digital input/output function.)

PA Direction Bit Set Register (GPIO_PADIRBSR)

Offset address: 18_H

Reset value:$xxxxxxxx\_xxxxxxxx\_xxxxxxxx\_xxxxxxxx_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIRBSR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIRBSR<15:0>															

DIRBSR<31:0>	bit 31-0	W	**PA direction bit set** 0: The GPIO_PADIR value stays unchanged 1: Set the corresponding bit of GPIO_PADIR to 1

PA Direction Bit Clear Register(GPIO_PADIRBCR)

Offset address: $1C_H$

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIRBCR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIRBCR<15:0>															

DIRBCR<31:0>	bit 31-0	W	**PA direction bit clear** 0: The GPIO_PADIR value stays unchanged 1: Set the corresponding bit of GPIO_PADIR to 0

PA Direction Bit Invert Register (GPIO_PADIRBRR)

Offset address: 20_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DIRBRR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DIRBRR<15:0>															

DIRBRR<31:0>	bit 31-0	W	**PA direction bit invert** 0: The GPIO_PADIR value stays unchanged 1: Set 1 to invert the corresponding bit of GPIO_PADIR

PA[7:0] Function Select Register (GPIO_PAFUNC0)

Offset address:24_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PA7<1:0>		Reserved		PA6<1:0>		Reserved		PA5<1:0>		Reserved		PA4<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PA3<1:0>		Reserved		PA2<1:0>		Reserved		PA1<1:0>		Reserved		PA0<1:0>	

—	bit31-30	—	—
PA7<1:0>	bit29-28	R/W	**PA7 function select bits** 00:FUN0 01:FUN1 10:FUN2 11:FUN3

—	bit27-26	—	—
PA6<1:0>	bit25-24	R/W	**PA6 function select bits** 00~11:FUN0~FUN3
—	bit23-22	—	—
PA5<1:0>	bit21-20	R/W	**PA5 function select bits** 00~11:FUN0~FUN3
—	bit19-18	—	—
PA4<1:0>	bit17-16	R/W	**PA4 function select bits** 00~11:FUN0~FUN3
—	bit15-14	—	—
PA3<1:0>	bit13-12	R/W	**PA3 function select bits** 00~11:FUN0~FUN3
—	bit11-10	—	—
PA2<1:0>	bit9-8	R/W	**PA2 function select bits** 00~11:FUN0~FUN3
—	bit7-6	—	—
PA1<1:0>	bit5-4	R/W	**PA1 function select bits** 00~11:FUN0~FUN3
—	bit3-2	—	—
PA0<1:0>	bit1-0	R/W	**PA0 function select bits** 00~11:FUN0~FUN3

PA[15:8] Function Select Register (GPIO_PAFUNC1)

Offset address:28_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PA15<1:0>		Reserved		PA14<1:0>		Reserved		PA13<1:0>		Reserved		PA12<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PA11<1:0>		Reserved		PA10<1:0>		Reserved		PA9<1:0>		Reserved		PA8<1:0>	

—	bit31-30	—	—
PA15<1:0>	bit29-28	R/W	**PA15 function select bits** 00:FUN0 01:FUN1 10:FUN2 11:FUN3
—	bit27-26	—	—
PA14<1:0>	bit25-24	R/W	**PA14 function select bits** 00~11:FUN0~FUN3
—	bit23-22	—	—

PA13<1:0>	bit21-20	R/W	**PA13 function select bits** 00~11:FUN0~FUN3
—	bit19-18	—	**—**
PA12<1:0>	bit17-16	R/W	**PA12 function select bits** 00~11:FUN0~FUN3
—	bit15-14	—	**—**
PA11<1:0>	bit13-12	R/W	**PA11 function select bits** 00~11:FUN0~FUN3
—	bit11-10	—	**—**
PA10<1:0>	bit9-8	R/W	**PA10 function select bits** 00~11:FUN0~FUN3
—	bit7-6	—	**—**
PA9<1:0>	bit5-4	R/W	**PA9 function select bits** 00~11:FUN0~FUN3
—	bit3-2	—	**—**
PA8<1:0>	bit1-0	R/W	**PA8 function select bits** 00~11:FUN0~FUN3

PA[23:16] Function Select Register (GPIO_PAFUNC2)

Offset address:$2C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PA23<1:0>		Reserved		PA22<1:0>		Reserved		PA21<1:0>		Reserved		PA20<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PA19<1:0>		Reserved		PA18<1:0>		Reserved		PA17<1:0>		Reserved		PA16<1:0>	

—	bit31-30	—	**—**
PA23<1:0>	bit29-28	R/W	**PA23 function select bits** 00:FUN0 01:FUN1 10:FUN2 11:FUN3
—	bit27-26	—	**—**
PA22<1:0>	bit25-24	R/W	**PA22 function select bits** 00~11:FUN0~FUN3
—	bit23-22	—	**—**
PA21<1:0>	bit21-20	R/W	**PA21 function select bits** 00~11:FUN0~FUN3
—	bit19-18	—	**—**
PA20<1:0>	bit17-16	R/W	**PA20 function select bits**

			00~11:FUN0~FUN3
—	bit15-14	—	—
PA19<1:0>	bit13-12	R/W	**PA19 function select bits** 00~11:FUN0~FUN3
—	bit11-10	—	—
PA18<1:0>	bit9-8	R/W	**PA18 function select bits** 00~11:FUN0~FUN3
—	bit7-6	—	—
PA17<1:0>	bit5-4	R/W	**PA17 function select bits** 00~11:FUN0~FUN3
—	bit3-2	—	—
PA16<1:0>	bit1-0	R/W	**PA16 function select bits** 00~11:FUN0~FUN3

PA[31:24] Function Select Register (GPIO_PAFUNC3)

Offset address:30_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PA31<1:0>		Reserved		PA30<1:0>		Reserved		PA29<1:0>		Reserved		PA28<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PA27<1:0>		Reserved		PA26<1:0>		Reserved		PA25<1:0>		Reserved		PA24<1:0>	

—	bit31-30	—	—
PA31<1:0>	bit29-28	R/W	**PA31 function select bits** 00:FUN0 01:FUN1 10:FUN2 11:FUN3
—	bit27-26	—	—
PA30<1:0>	bit25-24	R/W	**PA30 function select bits** 00~11:FUN0~FUN3
—	bit23-22	—	—
PA29<1:0>	bit21-20	R/W	**PA29 function select bits** 00~11:FUN0~FUN3
—	bit19-18	—	—
PA28<1:0>	bit17-16	R/W	**PA28 function select bits** 00~11:FUN0~FUN3
—	bit15-14	—	—
PA27<1:0>	bit13-12	R/W	**PA27 function select bits** 00~11:FUN0~FUN3

—	bit11-10	—	—
PA26<1:0>	bit9-8	R/W	**PA26 function select bits** 00~11:FUN0~FUN3
—	bit7-6	—	—
PA25<1:0>	bit5-4	R/W	**PA25 function select bits** 00~11:FUN0~FUN3
—	bit3-2	—	—
PA24<1:0>	bit1-0	R/W	**PA24 function select bits** 00~11:FUN0~FUN3

Notes

1. Only one from PA2/PA3/PA27/PA26 can be configured as a break signal for PWM output, and their priority level is PA2>PA3>PA27>PA26. For example, if PA3 is already a break signal, PA27 and PA26 cannot be configured as break signals.
2. The GPIO_PAFUNC register selects the input or output for digital function only. To enable the analog function, the corresponding bit of GPIO_PADIR and GPIO_PAINEB must be set to 1 to disable the digital input/output.

PA Input Enable Register GPIO_PAINEB)

Offset address:34_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INEB<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INEB<15:0>															

INEB<31:0>	bit31-0	R/W	**Digital input enable bit** 0: Enabled 1: Disabled

PA Open-drain Enable Register (GPIO_PAODE)

Offset address:38_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
ODE<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ODE<15:0>															

ODE<31:0>	bit31-0	R/W	**Port output open-drain enable bit** 0: Open-drain is disabled, and the port is in push-pull mode. 1: Open-drain is enabled

PA Weak Pull-up Enable Register (GPIO_PAPUE)

Offset address:$3C_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PUE <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PUE <15:0>															

PUE<31:0>	bit31-0	R/W	**Weak pull-up enable bit** 0: Disabled 1: Enabled

PA Weak Pull-down Enable Register (GPIO_PAPDE)

Offset address:40_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PDE <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PDE <15:0>															

PDE<31:0>	bit31-0	R/W	**Weak pull-down enable bit** 0: Disabled 1: Enabled

PA Drive Strength Control Register (GPIO_PADS)

Offset address:44_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
DS <31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
DS <15:0>															

DS<31:0>	bit31-0	R/W	**Output driven strength select bit** 0: Normal current driven 1: High current driven

PB Port Status Register (GPIO_PBPORT)

Offset address:80_H

Reset value:00000000_00000000_00xxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PORT<13:0>													

—	bit31-14	—	—
PORT<13:0>	bit13-0	R	**PB port level** 0: Low 1: High

PB Output Data Register (GPIO_PBDATA)

Offset address:84_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DATA<13:0>													

—	bit31-14	—	—
DATA<13:0>	bit13-0	R/W	**PB output level** 0: Output low level 1: Output high level

PB Output Data Bit Set Register (GPIO_PBDATABSR)

Offset address:88_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DATABSR<13:0>													

—	bit31-14	—	—
DATABSR<13:0>	bit13-0	W	**PB output data bit set** 0: Stay unchanged 1: Output high level the on the corresponding pin

PB Output Data Bit Clear Register (GPIO_PBDATABCR)

Offset address:$8C_H$

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DATABCR<13:0>													

—	bit31-14	—	—
DATABCR<13:0>	bit13-0	W	**PB output data bit clear** 0: Stay unchanged 1: Output low level on the corresponding pin

PB Output Data Bit Invert Register (GPIO_PBDATABRR)

Offset address:90_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DATABRR<13:0>													

—	bit31-14	—	—
DATABRR<13:0>	bit13-0	W	**PB output data bit invert** 0: Stay unchanged 1: Invert the output on the corresponding pin

PB Direction Register (GPIO_PBDIR)

Offset addrss:94_H

Reset value:11111111_11111111_11111111_11111111$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DIR<13:0>													

—	bit31-14	—	—
DIR<13:0>	bit13-0	R/W	**PB direction control bit** 0: Output 1: Non-output (If the corresponding bit of GPIO_PBINEB is 0, it is used as digital input. If the analog channel needs to be enabled, the corresponding bit of GPIO_PBINEB and GPIO_PBDIR must be 1 to disable the digital input/output function.)

PB Direction Bit Set Register (GPIO_PBDIRBSR)

Offset address:98_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DIRBSR<13:0>													

—	bit31-14	—	—
DIRBSR<13:0>	bit13-0	W	**PB direction bit set** 0: The GPIO_PBDIR value stays unchanged 1: Set the corresponding bit of GPIO_PBDIR to 1

PB Direction Bit Clear Register (GPIO_PBDIRBCR)

Offset address:$9C_H$

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DIRBCR<13:0>													

—	bit31-14	—	—
DIRBCR<13:0>	bit13-0	W	**PB direction bit clear** 0: The GPIO_PBDIR value stays unchanged 1: Set the corresponding bit of GPIO_PBDIR to 0

PB Direction Bit Invert Register (GPIO_PBDIRBRR)

Offset address:$A0_H$

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DIRBRR<13:0>													

—	bit31-14	—	—
DIRBRR<13:0>	bit13-0	W	**PB direction bit invert** 0: The GPIO_PBDIR value stays unchanged 1: Invert the corresponding bit of GPIO_PBDIR

PB[7:0] Function Select Register(GPIO_PBFUNC0)

Offset address:$A4_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		PB7<1:0>		Reserved		PB6<1:0>		Reserved		PB5<1:0>		Reserved		PB4<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PB3<1:0>		Reserved		PB2<1:0>		Reserved		PB <1:0>		Reserved		PB0<1:0>	

—	bit31-30	—	—
PB7<1:0>	bit29-28	R/W	**PB7 function select bits** 00:FUN0 01:FUN1 10:FUN2 11:FUN3
—	bit27-26	—	—
PB6<1:0>	bit25-24	R/W	**PB6 function select bits** 00~11:FUN0~FUN3
—	bit23-22	—	—
PB5<1:0>	bit21-20	R/W	**PB5 function select bits** 00~11:FUN0~FUN3
—	bit19-18	—	—
PB4<1:0>	bit17-16	R/W	**PB4 function select bits** 00~11:FUN0~FUN3
—	bit15-14	—	—
PB3<1:0>	bit13-12	R/W	**PB3 function select bits** 00~11:FUN0~FUN3

—	bit11-10	—	—
PB2<1:0>	bit9-8	R/W	**PB2 function select bits** 00~11:FUN0~FUN3
—	bit7-6	—	—
PB1<1:0>	bit5-4	R/W	**PB1 function select bits** 00~11:FUN0~FUN3
—	bit3-2	—	—
PB0<1:0>	bit1-0	R/W	**PB0 function select bits** 00~11:FUN0~FUN3

PB[15:8] Function Select Register (GPIO_PBFUNC1)

Offset address:$A8_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										PB13<1:0>		Reserved		PB12<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PB11<1:0>		Reserved		PB10<1:0>		Reserved		PB9<1:0>		Reserved		PB8<1:0>	

—	bit31-22	—	—
PB13<1:0>	bit21-20	R/W	**PB13 function select bits** 00:FUN0 01:FUN1 10:FUN2 11:FUN3
—	bit19-18	—	—
PB12<1:0>	bit17-16	R/W	**PB12 function select bits** 00~11:FUN0~FUN3
—	bit15-14	—	—
PB11<1:0>	bit13-12	R/W	**PB11 function select bits** 00~11:FUN0~FUN3
—	bit11-10	—	—
PB10<1:0>	bit9-8	R/W	**PB10 function select bits** 00~11:FUN0~FUN3
—	bit7-6	—	—
PB9<1:0>	bit5-4	R/W	**PB9 function select bits** 00~11:FUN0~FUN3
—	bit3-2	—	—
PB8<1:0>	bit1-0	R/W	**PB8 function select bits** 00~11:FUN0~FUN3

Note: The GPIO_PBFUNC register selects the input or output for digital function only. To enable the analog function, the corresponding bit of GPIO_PBDIR and GPIO_PBINEB must be set to 1 to disable the digital input/output.

PB Input Enable Register (GPIO_PBINEB)

Offset address:$B4_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		INEB<13:0>													

—	bit31-14	—	—
INEB<13:0>	bit13-0	R/W	**Digital input enable bit** 0: Enabled 1: Disabled

PB Open-drain Enable Register (GPIO_PBODE)

Offset address:$B8_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		ODE<13:0>													

—	bit31-14	—	—
ODE<13:0>	bit13-0	R/W	**Output open-drain enable bit** 0: Open-drain is disabled, and the port is in push-pull mode. 1: Open-drain is enabled

PB Weak Pull-up Enable Register (GPIO_PBPUE)

Offset address:BC_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PUEN<13:0>													

—	bit31-14	—	—
PUEN<13:0>	bit13-0	R/W	**Weak pull-up enable bit** 0: Disabled 1: Enabled

Note: If the MRSTN pin is used for external reset function, there is an internal integrated weak pull-up resister of 45KΩ, and it is not controlled by the weak pull-up enable register.

PB Weak Pull-down Enable Register (GPIO_PBPDE)

Offset address:$C0_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		PDEN<13:0>													

—	bit31-14	—	—
PDEN<13:0>	bit13-0	R/W	**Weak pull-down enable bit** 0: Disabled 1: Enabled

PB Drive Strength Control Register (GPIO_PBDS)

Offset address:$C4_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DS<13:0>													

—	bit31-14	—	—
DS<13:0>	bit13-0	R/W	**Output driven strength select bit** 0:Normal current driven 1:High current driven

PINT Interrupt Enable Register (GPIO_PINTIE)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PMASK<7:0>								PINTIE<7:0>							

—	bit31-16	—	—
PMASK<7:0>	bit15-8	R/W	**PINT interrupt source mask bit** 0:Not masked 1:Masked
PINTIE<7:0>	bit7-0	R/W	**PINT enable bit** 0: Disabled 1: Enabled

PINT Interrupt Flag Register (GPIO_PINTIF)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PINTIF<7:0>							

—	bit 31-8	—	—
PINTIF<7:0>	bit7-0	R/W	**GPIO external interrupt flag bit** 0: No interrupt occurred 1: Interrupt occurred Clear the flag by software writing 1, and writing 0 has no effect.

Note: For each flag bit of the GPIO_PINTIF register, they have to be cleared by software writing1, and writing 0 has no effect. When reading, 1 indicates that an interrupt has occurred.

PINT0~7 Interrupt Source Select Register (GPIO_PINTSEL)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16
Reserved	PINT7SEL<2:0>	Reserved	PINT6SEL<2:0>	Reserved	PINT5SEL<2:0>	Reserved	PINT4SEL<2:0>

15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
Reserved	PINT3SEL<2:0>	Reserved	PINT2SEL<2:0>	Reserved	PINT1SEL<2:0>	Reserved	PINT0SEL<2:0>

—	bit31	—	—
PINT7SEL<2:0>	bit30-28	R/W	**PINT7 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit27	—	—
PINT6SEL<2:0>	bit26-24	R/W	**PINT6 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit23	—	—
PINT5SEL<2:0>	bit22-20	R/W	**PINT5 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit19	—	—
PINT4SEL<2:0>	bit18-16	R/W	**PINT4 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit15	—	—
PINT3SEL<2:0>	bit14-12	R/W	**PINT3 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit11	—	—
PINT2SEL<2:0>	bit10-8	R/W	**PINT2 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit7	—	—
PINT1SEL<2:0>	bit6-4	R/W	**PINT1 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit3	—	—
PINT0SEL<2:0>	bit2-0	R/W	**PINT0 input select bits** 000~101: SEL0~SEL5 Others: SEL0

PINT Interrupt Configuration Register (GPIO_PINTCFG)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16
Reserved	PINT7CFG<2:0>	Reserved	PINT6CFG<2:0>	Reserved	PINT5CFG<2:0>	Reserved	PINT4CFG<2:0>

15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
Reserved	PINT3CFG<2:0>	Reserved	PINT2CFG<2:0>	Reserved	PINT1CFG<2:0>	Reserved	PINT0CFG<2:0>

—	bit31	—	—
PINT7CFG<2:0>	bit30-28	R/W	**PINT7 configuration bits** See GPIO_PINTCFG for details
—	bit27	—	—
PINT6CFG<2:0>	bit26-24	R/W	**PINT6 configuration bits** See GPIO_PINTCFG for details
—	bit23	—	—
PINT5CFG<2:0>	bit22-20	R/W	**PINT5 configuration bits** See GPIO_PINTCFG for details
—	bit19	—	—
PINT4CFG<2:0>	bit18-16	R/W	**PINT4 configuration bits** See GPIO_PINTCFG for details
—	bit15	—	—
PINT3CFG<2:0>	bit14-12	R/W	**PINT3 configuration bits** See GPIO_PINTCFG for details
—	bit11	—	—
PINT2CFG<2:0>	bit10-8	R/W	**PINT2 configuration bits** See GPIO_PINTCFG for details
—	bit7	—	—
PINT1CFG<2:0>	bit6-4	R/W	**PINT1 configuration bits** See GPIO_PINTCFG for details
—	bit3	—	—
PINT0CFG<2:0>	bit2-0	R/W	**PINT0 configuration bits** See GPIO_PINTCFG for details

Register Name	GPIO_PINTCFG		
PINTCFG<2:0>	bit 2-0	R/W	**Port interrupt trigger event select bits** 000: Rising edg 001: Falling edge 010: High level 011: Low level 1xx: Both rising and falling edge

KINT Interrupt Enable Register (GPIO_KINTIE)

Offset address:10_H

Reset value:00000000_00000000_11111111_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
KMASK<7:0>								KINTIE<7:0>							

—	bit31-16	—	—
KMASK<7:0>	bit15-8	R/W	**KIN key input mask bit** 0: Not masked 1: Masked
KINTIE<7:0>	bit7-0	R/W	**KINT interrupt enable bit** 0: Disabled 1: Enabled

KINT Interrupt Flag Register (GPIO_KINTIF)

Offset address:14_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								KINTIF<7:0>							

—	bit 31-8	—	—
KINTIF<7:0>	bit7-0	R/W	**GPIO key interrupt flag bit** 0: No interrupt occurred 1: Interrupt occurred Clear the flag bit by software writing 1, and writing 0has no effect.

Note: For each flag bit of the GPIO_KINTIF register, they have to be cleared by software writing1, and writing 0 has no effect. When reading, 1 indicates that an interrupt has occurred.

KINT0~7 Interrupt Source Select Register (GPIO_KINTSEL)

Offset address:18_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16
Reserved	KINT7SEL<2:0>	Reserved	KINT6SEL<2:0>	Reserved	KINT5SEL<2:0>	Reserved	KINT4SEL<2:0>

15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
Reserved	KINT3SEL<2:0>	Reserved	KINT2SEL<2:0>	Reserved	KINT1SEL<2:0>	Reserved	KINT0SEL<2:0>

—	bit31	—	**—**
KINT7SEL<2:0>	bit30-28	R/W	**KINT7 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit27	—	**—**
KINT6SEL<2:0>	bit26-24	R/W	**KINT6 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit23	—	**—**
KINT5SEL<2:0>	bit22-20	R/W	**KINT5 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit19	—	**—**
KINT4SEL<2:0>	bit18-16	R/W	**KINT4 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit15	—	**—**
KINT3SEL<2:0>	bit14-12	R/W	**KINT3 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit11	—	**—**
KINT2SEL<2:0>	bit10-8	R/W	**KINT2 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit7	—	**—**
KINT1SEL<2:0>	bit6-4	R/W	**KINT1 input select bits** 000~101: SEL0~SEL5 Others: SEL0
—	bit3	—	**—**
KINT0SEL<2:0>	bit2-0	R/W	**KINT0 input select bits** 000~101: SEL0~SEL5 Others: SEL0

KINT Interrupt Configuration Register (GPIO_KINTCFG)

Offset address:$1C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30 29 28	27	26 25 24	23	22 21 20	19	18 17 16
Reserved	KINT7CFG<2:0>	Reserved	KINT6CFG<2:0>	Reserved	KINT5CFG<2:0>	Reserved	KINT4CFG<2:0>

15	14 13 12	11	10 9 8	7	6 5 4	3	2 1 0
Reserved	KINT3CFG<2:0>	Reserved	KINT2CFG<2:0>	Reserved	KINT1CFG<2:0>	Reserved	KINT0CFG<2:0>

—	bit31	—	—
KINT7CFG<2:0>	bit30-28	R/W	**KINT7 configuration bits** See GPIO_KINTCFG for details
—	bit27	—	—
KINT6CFG<2:0>	bit26-24	R/W	**KINT6 configuration bits** See GPIO_KINTCFG for details
—	bit23	—	—
KINT5CFG<2:0>	bit22-20	R/W	**KINT5 configuration bits** See GPIO_KINTCFG for details
—	bit19	—	—
KINT4CFG<2:0>	bit18-16	R/W	**KINT4 configuration bits** See GPIO_KINTCFG for details
—	bit15	—	—
KINT3CFG<2:0>	bit14-12	R/W	**KINT3 configuration bits** See GPIO_KINTCFG for details
—	bit11	—	—
KINT2CFG<2:0>	bit10-8	R/W	**KINT2 configuration bits** See GPIO_KINTCFG for details
—	bit7	—	—
KINT1CFG<2:0>	bit6-4	R/W	**KINT1 configuration bits** See GPIO_KINTCFG for details
—	bit3	—	—
KINT0CFG<2:0>	bit2-0	R/W	**KINT0 configuration bits** See GPIO_KINTCFG for details

Register Name	GPIO_KINTCFG		
GPIO_KINTCFG	bit 2-0	R/W	**Key interrupt trigger event select bits** 000: Rising edge 001: Falling edge 010: High level 011: Low level 1xx: Both rising and falling edge

Port Interrupt 20ns Filter Select Register (GPIO_IOINTFLTS)

Offset address:30_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								FLT_S<7:0>							

—	bit31-8	—	—
FLT_S<7:0>	bit7-0	R/W	**20ns filter select bits** 0: Used to filter the PINT interrupt source 1: Used to filter the KINT interrupt source

TMR 20ns Filter Select Register (GPIO_TMRFLTSEL)

Offset address:40_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
FLT3_SEL<3:0>				FLT2_SEL<3:0>				FLT1_SEL<3:0>				FLT0_SEL<3:0>			

—	bit31-16	—	—
FLT3_SEL<3:0>	bit15-12	R/W	**TMR FLT3 select bits** 0000: TMR_FLT3 used by T16N0_0 0001: TMR_FLT3 used by T16N0_1 0010: TMR_FLT3 used by T16N1_0 0011: TMR_FLT3 used by T16N1_1 0100: TMR_FLT3 used by T16N2_0 0101: TMR_FLT3 used by T16N2_1 0110: TMR_FLT3 used by T16N3_0 0111: TMR_FLT3 used by T16N3_1 1000: TMR_FLT3 used by T32N0_0 1001: TMR_FLT3 used by T32N0_1 Others: Reserved
FLT2_SEL<3:0>	bit11-8	R/W	**TMR FLT2 select bits** 0000: TMR_FLT2 used by T16N0_0 0001: TMR_FLT2 used by T16N0_1 0010: TMR_FLT2 used by T16N1_0 0011: TMR_FLT2 used by T16N1_1 0100: TMR_FLT2 used by T16N2_0

			0101: TMR_FLT2 used by T16N2_1 0110: TMR_FLT2 used by T16N3_0 0111: TMR_FLT2 used by T16N3_1 1000: TMR_FLT2 used by T32N0_0 1001: TMR_FLT2 used by T32N0_1 Others: Reserved
FLT1_SEL<3:0>	bit7-4	R/W	**TMR FLT1 select bits** 0000: TMR_FLT1 used by T16N0_0 0001: TMR_FLT1 used by T16N0_1 0010: TMR_FLT1 used by T16N1_0 0011: TMR_FLT1 used by T16N1_1 0100: TMR_FLT1 used by T16N2_0 0101: TMR_FLT1 used by T16N2_1 0110: TMR_FLT1 used by T16N3_0 0111: TMR_FLT1 used by T16N3_1 1000: TMR_FLT1 used by T32N0_0 1001: TMR_FLT1 used by T32N0_1 Others: Reserved
FLT0_SEL<3:0>	bit3-0	R/W	**TMR FLT0 select bits** 0000: TMR_FLT0 used by T16N0_0 0001: TMR_FLT0 used by T16N0_1 0010: TMR_FLT0 used by T16N1_0 0011: TMR_FLT0 used by T16N1_1 0100: TMR_FLT0 used by T16N2_0 0101: TMR_FLT0 used by T16N2_1 0110: TMR_FLT0 used by T16N3_0 0111: TMR_FLT0 used by T16N3_1 1000: TMR_FLT0 used by T32N0_0 1001: TMR_FLT0 used by T32N0_1 Others: Reserved

SPI 20ns Filter Select Register (GPIO_SPIFLTSEL)

Offset address:44_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												FLT_SEL<3:0>			

—	bit31-4	—	—
FLT_SEL<3:0>	bit3-0	R/W	**SPI Filter select bits**

			1: The filter is assigned to SPI1 0: The filter is assigned to SPI0

PWM Register (GPIO_TXPWM)

Offset address:80_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TX1_S3	TX1_S2	TX1_S1	TX1_S0	Reserved	TX1PLV	TX1PS<1:0>		TX0_S3	TX0_S2	TX0_S1	TX0_S0	Reserved	TX0PLV	TX0PS<1:0>	

—	bit31-16	—	—
TX1_S3	bit15	R/W	**TXPWM1 output enable (valid only when TX0_S3 = 0, where the TXPWM0 is not enabled and output to BUZ pin)** 0: Normal BUZ pin 1: TXPWM1 output is enabled and output to the BUZ pin.
TX1_S2	bit14	R/W	**TXPWM1 output pin enable** 0: Normal T16N1OUT1 pin 1: TXPWM1 output is enabled and output to the T16N1OUT1 pin
TX1_S1	bit13	R/W	**TXPWM1 output pin enable** 0: Normal T16N1OUT0 pin 1: TXPWM1 output is enabled and output to the T16N1OUT0 pin
TX1_S0	bit12	R/W	**TXPWM1 output pin enable** 0: Normal E0TX0 pin 1: TXPWM1output is enabled and output to the E0TX0 pin
—	bit11	—	—
TX1PLV	bit10	R/W	**TXPWM1 modulation level select bit** 0: Low level (perform hardware OR operation on E0TX0 and the signal selected by TX1PS) 1: High level (perform hardware AND operation on E0TX0 and the signal selected by TX1PS)
TX1PS<1:0>	bit9-8	R/W	**TXPWM1 modulation signal select bit** 00: Modulation disabled

			01: T16N1OUT0 signal 10: T16N1OUT1 signal 11: BUZ output signal
TX0_S3	bit7	R/W	**TXPWM0 output pin enable** 0: Normal BUZ pin 1: TXPWM0 output is enabled and output to the BUZ pin
TX0_S2	bit6	R/W	**TXPWM0 output pin enable** 0: Normal T16N0OUT1 pin 1: TXPWM0 output is enabled and output to the T16N0OUT1 pin
TX0_S1	bit5	R/W	**TXPWM0 output pin enable** 0: Normal T16N0OUT0 pin 1: TXPWM0 output is enabled and output to the T16N0OUT0 pin
TX0_S0	bit4	R/W	**TXPWM0 output pin enable** 0: Normal TX0 pin 1: TXPWM0 output is enabled and output to the TXD0 pin
—	bit3	—	—
TX0PLV	bit2	R/W	**TXPWM0 modulation level select bit** 0: Low level (perform hardware OR operation on TX0 and the signal selected by TX0PS) 1: High level (perform hardware AND operation on TX0 and the signal selected by TX0PS)
TX0PS<1:0>	bit1-0	R/W	**TXPWM0 modulation signal select bit** 00: Modulation disabled 01: T16N0OUT0 signal 10: T16N0OUT1 signal 11: BUZ output signal

BUZ Control Register GPIO_BUZC)

Offset address:84_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				BUZ_LOAD<19:8>											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BUZ_LOAD<7:0>								Reserved							BUZEN

—	bit31-28	—	—
BUZ_LOAD<19:0>	bit27-8	R/W	**BUZ load value**

			BUZ signal frequency calculation: $F_{BUZ} = \frac{Fpclk}{2 \times (BUZ\_LOAD + 1)}$
—	bit7-1	—	—
BUZEN	bit0	R/W	**BUZ enable bit** 0: Disabled 1: Enabled

Chapter5 Peripherals

5. 1 Timer/Counter

5. 1. 1 16-bit Timer/Counter T16N

In this section, T16N0 is introduced and described in detail. Other timers/counters T16N1/ T16N2/ T16N3 are identical with T16N0.

5. 1. 1. 1 Overview

- 1x 8-bit configurable prescaler, a prescaled clock is used as a counter clock for T16N_CNT0/1 timer/counter
  - ◇ Prescaler clock source options: PCLK or T16N0CK0/T16N0CK1
  - ◇ Initial value of prescaler counter set by the T16N_PRECNT register
  - ◇ Prescaler ratio set by the T16N_PREMAT register
- 2x16-bit configurable timer/counter registers T16N_CNT0/T16N_CNT1
  - ◇ T16N_CNT1 only available in independent PWM mode, or used to trigger ADC conversion
- 2x16-bit top value registers T16N_TOP0/T16N_TOP1
  - ◇ T16N_TOP0/T16N_TOP1 is cleared when the counter reaches their individual top value.
  - ◇ In PWM mode, if the ADC trigger function is enabled, a trigger signal is generated to activate the AD conversion when T16N_CNT0/ T16N_CNT1 reach their top value.
- Timer/Counter mode
  - ◇ 4x16-bit match registers T16N_MAT0/T16N_MAT1/T16N_MAT2/ T16N_MAT3 are available. Following operations are supported after a match.
    - Interrupt
    - T16N_CNT0 operating modes: hold, clear and continue to count
    - Operations on T16N0OUT0/T16N0OUT1 port: hold, clear, set and toggle
- Input capture mode
  - ◇ Configurable capture edge
  - ◇ Selectable capture times
- Output PWM mode
  - ◇ Configure match registers and output polarity of the ports to obtain PWM output
  - ◇ ADC conversion triggered by the following
    - A match between MAT0, MAT1, MAT2, MAT3, TOP0 and T16_CNT0
    - A match between TOP1 and T16N_CNT1
  - ◇ Three configurable modes for 2 channels of PWM
    - Independent mode: T16N0OUT0 and T16N0OUT1 output distinct PWM

waveforms

- Synchronous mode: T16N0OUT0 and T16N0OUT1 output identical PWM waveforms
- Complementary mode: T16N0OUT0 and T16N0OUT1 output complementary PWM waveforms with programmable dead-zone time.

◇ Break control

- Any signal on PA2 (T16N_BK0), PA3 (T16N_BK1), PA26 (T16N_BK2) or PA27 (T16N_BK3) can be used as an external break signal, selected by GPIO_PAFUNC register, and its active level is configurable.
- Configurable break output level

5. 1. 1. 2 Block Diagram

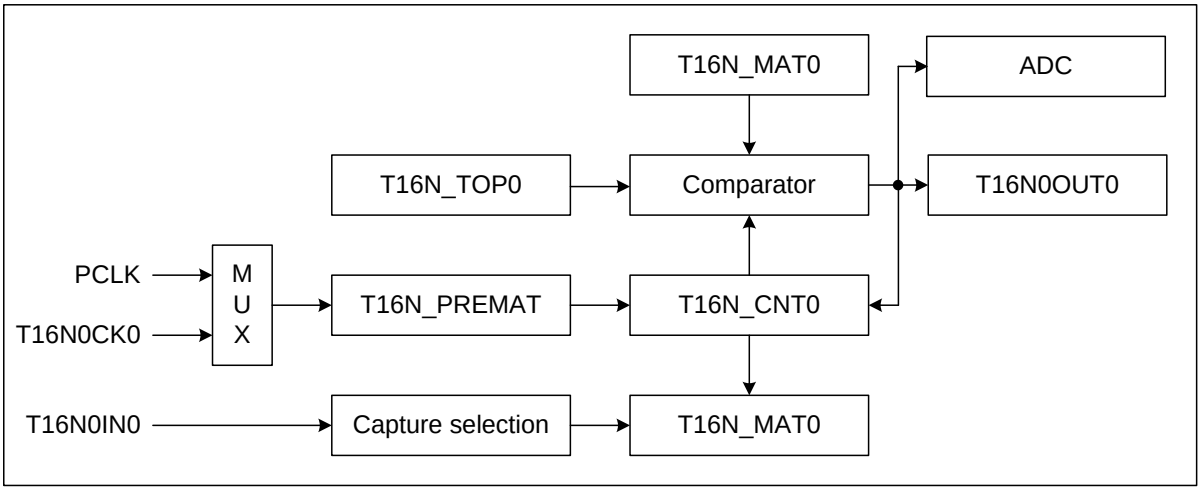


Figure 5-1 T16N0 Block Diagram

5. 1. 1. 3 T16N Timer/Counter

When the MOD<1:0> bits of the T16N_CON0 register is 00 or 01, the T16N works in timer/counter mode.

When the EN bit of the T16N_CON0 register is set, T16N is enabled and the counter register T16N_CNT0/T16N_CNT1 starts to increment from the initial value.

The CS<1:0> bits of the T16N_CON0 register select the counter clock source. When the internal clock PCLK is selected, T16N works in timer mode. When the external clock input T16N0CK0/T16N0CK1 is selected, T16N works in counter mode.

The SYNC bit of the T16N_CON0 register can be configured to synchronize the external clock T16N0CK0/T16N0CK1 with the internal clock PCLK. When synchronized, T16N works in synchronous counter mode. Otherwise, it works in asynchronous counter mode. In synchronous counter mode, the pulse width of the high/low level of the T16N0CK0/T16N0CK1 input must be greater than 2 PCLK clock cycles.

The counter can be configured to count at rising edge, falling edge or both rising and falling edges by the EDGE<1:0> bits of the T16N_CON0 register. Counting at both

rising and falling edge is only active in synchronous counter mode.

The MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0> bits of the T16N_CON0 register select the operation of the T16N_CNT0/ T16N_CNT1 register after a match has occurred.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=00: when the counter value of the T16N_CNT0/ T16N_CNT1 matches the T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, it will continue to count and no interrupt will be generated. When reaching 0xFFFF, an overflow will occur and an interrupt will be generated, and it will restart from 0x0000.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=01: when the counter value of the T16N_CNT0/ T16N_CNT1 matches the T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, the counter value will be held until the next prescaled counter clock arrives. Meanwhile, the counter will stop counting and an interrupt will be generated.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=10: when the counter value of the T16N_CNT0/ T16N_CNT1 matches the T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, the counter value will be cleared on the arrival of the next prescaled counter clock. Meanwhile, an interrupt will be generated and the counter will restart counting.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=11: when the counter value of the T16N_CNT0/ T16N_CNT1 matches the T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, it will continue to count up and an interrupt will be generated on the arrival of the next prescaled counter clock. When reaching 0xFFFF, an overflow will occur and an interrupt will be generated, and it will restart from 0x0000.

For different match values T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, when the counter value matches one of them, the corresponding interrupt will be generated. Following an interrupt, the T16N continues to count up, if the match interrupt flag has not been read in time, it is possible to read several valid match interrupt flags at the same time.

If the counter value T16N_CNT0/ 16N_CNT1 matches T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, when MOE0=1 of the T16N_CON2 register, the T16N0OUT0 will be toggled; when MOE1=1, the T16N0OUT1 will be toggled. The T16N0OUT0 and T16N0OUT1 can be enabled at the same time.

Example: When the counter value of the T16N_CNT0 register matches the T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, any of the following operations can be carried out.

If T16N_MAT0<15:0>=0x0002 and MAT0S<1:0>=00 of the T16N_CON0, it will continue to count up and no interrupt will be generated.

If T16N_MAT1<15:0>=0x0004 and MAT1S<1:0>=11 of theT16N_CON0, it will continue to count and generate an interrupt.

If T16N_MAT2<15:0>=0x0006 and MAT2S<1:0>=10 of the T16N_CON0, the counter

value will be cleared, an interrupt will be generated and the counter will restart counting.

When the prescaler ratio is set to 1:1, the internal PCLK is used. Figure below shows the counter match diagram.

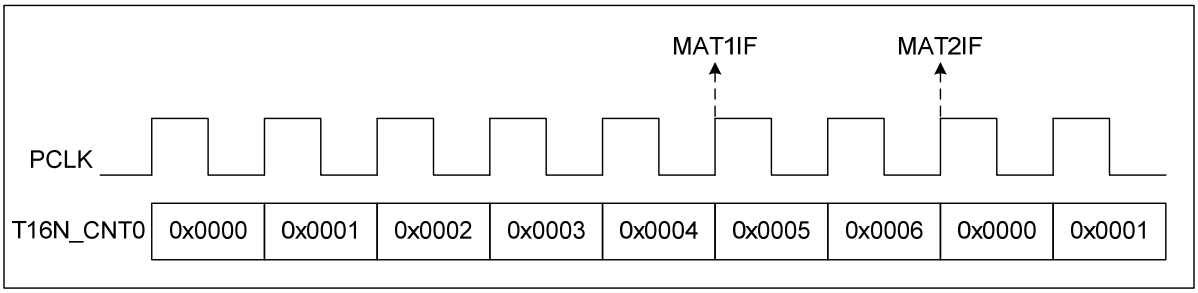


Figure 5-2 T16N0 Counter Match Diagram

Note: All match registers T16N_MAT are available to T16N_CNT0. However, when the counter is configured for capture mode (MOD=0x2), only T16N_MAT0 and T16N_MAT1 are available to T16N_CNT0. For T16N_CNT1, only T16N_MAT2 and T16N_MAT3 can be used.

5. 1. 1. 4 T16N Input Capture Mode

T16N works in capture mode by writing the MOD<1:0>=10 of the T16N_CON0 register. The T16N_CNT1 register is not available in capture mode.

In capture mode, writing CS<1:0>=00 in the T16N_CON0 register allows the T16N_CNT0 to use the PCLK as a counter clock. And when MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=00, it does not affect the operation of T16N_CNT0.

The following content describes the status detection of the T16N0IN0 and T16N0IN1 port.

When a capture event occurs on the T16N0IN0 port, load the current content of T16N_CNT0 and T16N_PRECNT to T16N_MAT0 and T16N_MAT2 respectively, and an interrupt CAP0IF occurs. The T16N_CNT0 and T16N_PRECNT can be cleared on a CAP0IF interrupt by enabling the CAPL0 of the T16N_CON1 register. If the CAPL0 is disabled, the counter continues to increment.

When a capture event occurs on the T16N0IN1 port, load the current content of T16N_CNT0 and T16N_PRECNT to T16N_MAT1 and T16N_MAT3 respectively, and an interrupt CAP1IF occurs. The T16N_CNT0 and T16N_PRECNT can be cleared on a CAP1IF interrupt by enabling the CAPL1 of the T16N_CON1 register. If the CAPL1 is disabled, the counter continues to increment.

If there has not been any capture event detected by the time when T16N_CNT0 overflows, the T16N_CNT0 will be cleared and restart counting.

The CAPPE and CAPNE bits of the T16N_CON1 register are used to select capture events on the T16N0IN0 and T16N0IN1 ports.

Write the CAPPE bit to 1 to select rising edge.

Write the CAPNE bit to 1 to select falling edge.

Write the CAPPE and CAPNE bit to 1 to select both rising and falling edges.

The T16N0IN0 port can be enabled as a capture input by setting the CAPIS0 bit of the T16N_CON1 register; the T16N0IN1 port can also be enabled as a capture input by setting the CAPIS1 bit. Both ports can be enabled at the same time.

The number of times of a capture can be configured by the CAPT<3:0> bits of the T16N_CON1 register.

The configuration below shows how to capture both rising and falling edges 8 times on the T16N0IN0 with prescaler ratio 1:1.

In T16N_CON0 register, write MOD<1:0>=10, CS<1:0>=00 and MAT0S<1:0>=00.

In T16N_CON1 register, write CAPPE=1, CAPNE=1, CAPIS=1 and CAPT<3:0>=0111.

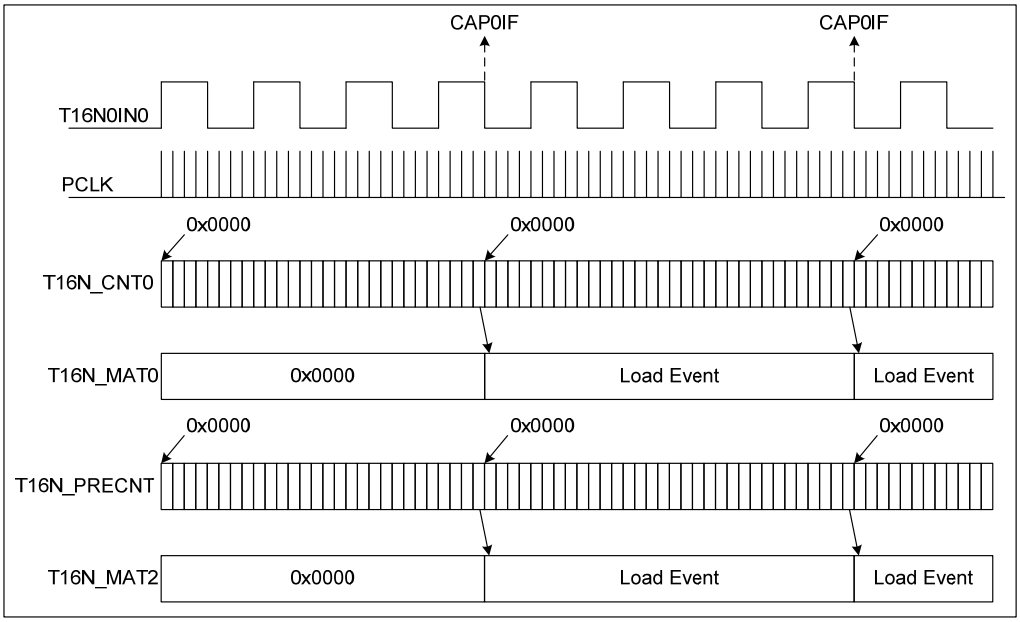


Figure 5-3 T16N0 Capture Function Diagram

In capture mode, the prescaler counter will not be cleared when altering the T16N_PREMAT register. Therefore, the first capture can be started from a non-zero prescaler counter value. When a capture event occurs, the interrupt flag must be cleared by software, and the captured value in T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3 register must be read in time. Otherwise, the old captured value will be overwritten by the new captured value when the next capture occurs.

5. 1. 1. 5 T16N Output PWM Mode

The T16N works in PWM mode by writing MOD<1:0>=11 in the T16N_CON0 register.

In PWM mode, writing CS<1:0>=00 of the T16N_CON0 register allows the T16N_CNT0/ T16N_CNT1 to use the internal PCLK as a counter clock.

The T16N0OUT0 port can be enabled by setting the MOE0 bit of the T16N_CON2 register. After enabling, a compare can be performed between T16N_MAT0/ T16N_MAT1 and T16N_CNT0. The T16N0OUT1 port can be enabled by setting the MOE1 bit of the T16N_CON2 register. After enabling, a compare can be performed between T16N_MAT2/ T16N_MAT3 and T16N_CNT0 or T16N_CNT1 (only in independent PWM mode).

Configure the MOM0/ MOM1/ MOM2/ MOM3<1:0> bits to select one of the operations, hold, clear, set and toggle, on the T16N0OUT0/ T16N0OUT1 port when a match occurs.

2 channels of PWM can be configured for independent mode, synchronous mode and complementary mode by setting the PWMMOD<1:0> bits of the T16N_CON2 register.

In independent mode, the T16N0OUT0 and T16N0OUT1 port output distinct PWM waveforms, where the T16N0OUT0 is dependent on the match between T16N_CNT0 and T16N_MAT0/ T16N_MAT1 while the T16N0OUT1 is dependent on the match between T16N_CNT1 and T16N_MAT2/ T16N_MAT3.

In synchronous mode, the T16N0OUT0 and T16N0OUT1 output identical PWM waveforms. Both T16N0OUT0 and T16N0OUT1 are dependent on the match between T16N_CNT0 and T16N_MAT0/ T16N_MAT1. In this case, no need to configure the T16N_CNT1.

In complementary mode, the T16N0OUT0 and T16N0OUT1 output the PWM waveforms that are complementary to each other with a programmable dead-zone time. Both T16N0OUT0 and T16N0OUT1 are dependent on the match between T16N_CNT0 and T16N_MAT0/ T16N_MAT1. The PWM period is determined by the T16N_TOP0, which is T16N_TOP0 + 1. In this case, no need to configure the T16N_CNT1.

ADC conversion can be activated in PWM mode. When a match between T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3, T16N_TOP0 and T16N_CNT0, or a match between T16N_TOP1 and the T16N_CNT1 occurs, a trigger signal can be generated to trigger the ADC.

The following shows two examples for independent mode and complementary mode.

1) In independent mode, 2 channels of distinct double-edge PWM waveforms are generated on T16N0OUT0 and T16N0OUT1.

In the T16N_CON2 register,

Write MOE0=1 and MOE1=1 to enable both T16N0OUT0 and T16N0OUT1 ports

Write MOM0<1:0>=10 (matched with T16N_MAT0), the T16N0OUT0 outputs high level.

Write MOM1<1:0>=01 (matched with T16N_MAT1), the T16N0OUT0 outputs low level.

Write MOM2<1:0>=10 (matched with T16N_MAT2), the T16N0OUT1 outputs high

level.

Write MOM3<1:0>=01 (matched with T16N_MAT3), the T16N0OUT1 outputs low level.

Write PWMMOD<1:0>=00, the T16N is in independent mode.

In the T16N_CON0 register,

Write MOD<1:0>=11 to select the PWM output mode

Write MAT0S<1:0>=11, the T16N_CNT0 keeps counting and generates an interrupt.

Write MAT1S<1:0>=11, the T16N_CNT0 keeps counting and generates an interrupt.

Write MAT2S<1:0>=11, the T16N_CNT1 keeps counting and generates an interrupt.

Write MAT3S<1:0>=10, the T16N_CNT1 is cleared and generates an interrupt.

In other registers,

Write T16N_MAT0 = 0x0002, T16N_MAT1 = 0x0004, T16N_MAT2 = 0x0006 and T16N_MAT3 = 0x0008.

Set a proper value to T16N_TOP0/ T16N_TOP1.

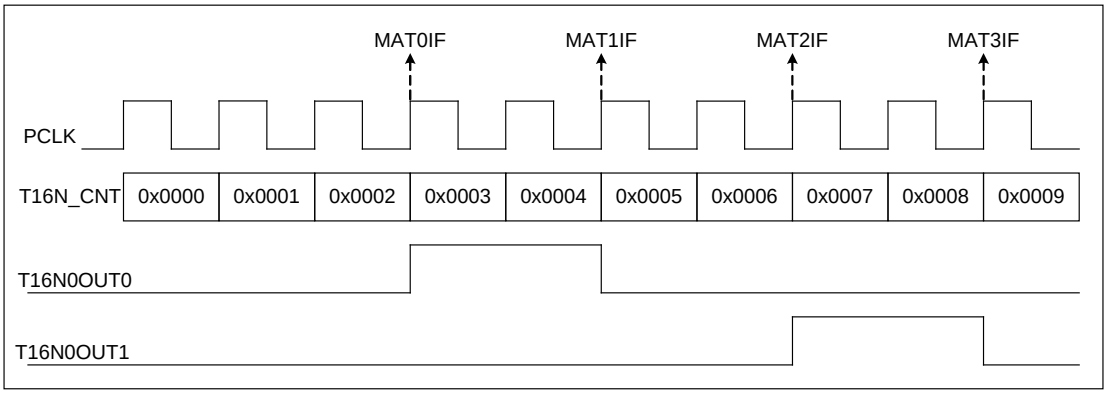


Figure 5-4 T16N0 PWM Modulation in Independent Mode

Note: The match event of T16N_MAT0 has higher priority than the one of T16N_MAT1. If a same value is set to T16N_MAT0 and T16N_MAT1, the output level of T16N0OUT0 is depends on MOM0 only.
The match event of T16N_MAT2 has higher priority than the one of T16N_MAT3. If a same value is set to T16N_MAT2 and T16N_MAT3, the output level of T16N0OUT1 is depends on MOM2 only.

2) In complementary mode, with PCLK=48MHz, two 24MHz PWM waveforms generated on the T16N0OUT0 and T16N0OUT1 ports are complementary to each other.

In the T16N_CON2 register,

Write MOE0=1 and MOE1=1 to enable both T16N0OUT0 and T16N0OUT1 ports.

Write MOM0<1:0>=10 (matched with T16N_MAT0), the T16N0OUT0 outputs high while the T16N0OUT1 outputs low.

Write MOM1<1:0>=01 (matched with T16N_MAT1), the T16N0OUT0 outputs low while the T16N0OUT1 outputs high.

Write PWMMOD<1:0>=11 to select the complementary mode.

Write PWMDZE=0 to disable the dead-zone time.

In the T16N_CON0 register,

Write MOD<1:0>=11 and MAT0S<1:0>=11, the T16N_CNT0 keeps counting and generates an interrupt.

Write MAT1S<1:0>=11, the T16N_CNT0 keeps counting and generates an interrupt.

In other registers,

Write T16N_MAT0 = 0x0000, T16N_MAT1 = 0x0001 and T16N_TOP0=0x0001.

Write T16N_PREMAT=0, no prescaling is to be carried out.

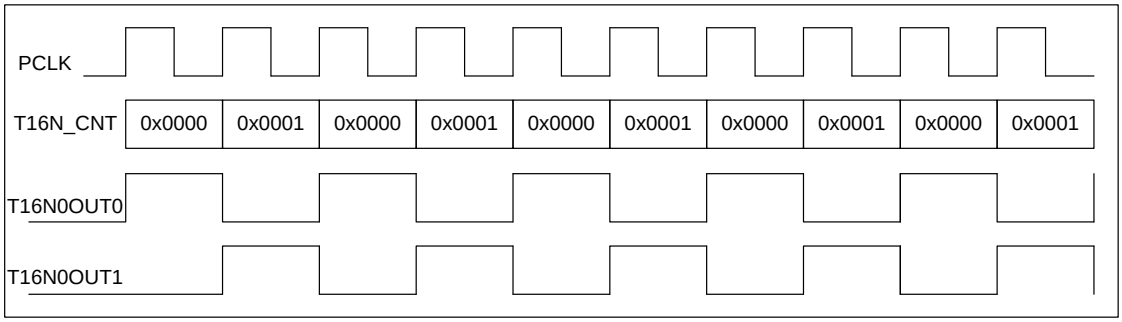


Figure 5-5 T16N0 PWM Modulation in Complementary Mode

3) In complementary mode with dead-zone, with PCLK=48MHz, PWM waveforms generated on the T16N0OUT0 and T16N0OUT1 ports are complementary with dead-zone to each other.

In the T16N_CON2 register,

Write MOE0=1 and MOE1=1 to enable both T16N0OUT0 and T16N0OUT1 ports.

Write MOM0<1:0>=10 (matched with T16N_MAT0), the T16N0OUT0 outputs high while the T16N0OUT1 outputs low.

Write MOM1<1:0>=01 (matched with T16N_MAT1), the T16N0OUT0 outputs low while the T16N0OUT1 outputs high.

Write PWMMOD<1:0>=11 to select the complementary mode.

Write PWMDZE=1 to enable the dead-zone time.

In the T16N_CON0 register,

Write MOD<1:0>=11 and MAT0S<1:0>=11, the T16N_CNT0 keeps counting and generates an interrupt.

Write MAT1S<1:0>=11, the T16N_CNT0 keeps counting and generates an interrupt.

In other registers,

T16N_MAT0, T16N_MAT1 and T16N_TOP0 register set the appropriate value to confige the PWM period, duty, etc.; T16N_PREMAT can be set to a small value, such as 0, which the prescaler ratio is 0, dead-zone time and PWM duty ratio have higher accuracy.

As a comparison, the PWM complementary output waveforms without dead-zone time and with dead-zone time are given respectively. It should be noted that the dead-zone time will compress the high level widths of T16N0OUT0 and T16N0OUT1.

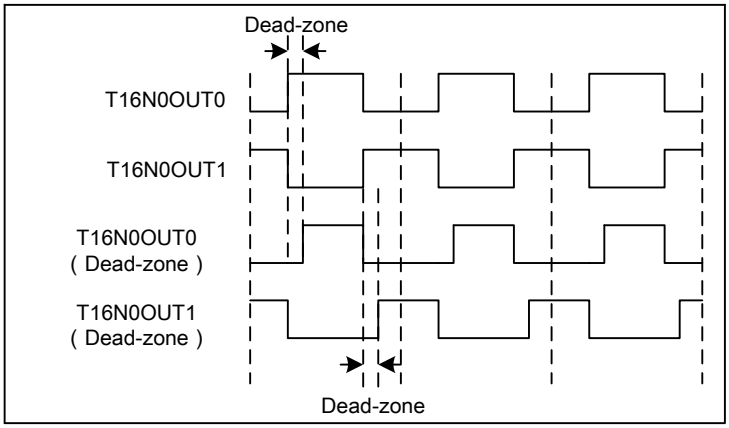


Figure 5-6 T16N0 PWM Modulation in Complementary Mode with dead-zone

5. 1. 1. 6 Special Function Registers

T16N Counter Register0 (T16N_CNT0)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT0<15:0>															

—	bit31-16	—	—
CNT0<15:0>	bit 15-0	R/W	T16N_CNT0 counter value

Note: All match registers T16N_MAT are available to T16N_CNT0. However, when the counter is configured for capture mode (MOD=0x2), only T16N_MAT0 and T16N_MAT1 are available to T16N_CNT0. For T16N_CNT1, only T16N_MAT2 and T16N_MAT3 can be used.

T16N Counter Register1 (T16N_CNT1)

Offset address:04_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT1<15:0>															

—	bit31-16	—	—
CNT1<15:0>	bit 15-0	R/W	T16N_CNT1 counter value

T16N Prescaler Counter Register (T16N_PRECNT)

Offset address:08_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PRECNT<7:0>							

—	bit31-8	—	—
PRECNT<7:0>	bit7-0	R/W	T16N prescaler counter value

T16N Prescaler Counter Match Register (T16N_PREMAT)

Offset address:$0C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PREMAT<7:0>							

—	bit31-8	—	—
PREMAT<7:0>	bit7-0	R/W	**Prescaler ratio select bits** 00:1:1 01:1:2 02:1:3 …… FE:1:255 FF:1:256

T16N Control Register0(T16N_CON0)

Offset address:10_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ASYWEN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT3S<1:0>		MAT2S<1:0>		MAT1S<1:0>		MAT0S<1:0>		MOD<1:0>		EDGE<1:0>		SYNC	CS<1:0>		EN

—	bit 31~17	—	—
ASYWEN	bit 16	R/W	**In synchronous counter mode, counter write enable bit** 0: Writing T16N_CNT1 is enabled. In synchronous counter mode, timer mode and PWM mode, writing T16N_CNT0 and T16N_PRECNT is enabled (in asynchronous mode, do not write T16N_CNT0 or T16N_PRECNT when ASYWEN=0) 1: Writing T16N_CNT1 is disabled. In synchronous counter mode, timer mode and PWM mode, writing T16N_CNT0 and T16N_PRECNT is disabled. In asynchronous mode, writing T16N_CNT0 and T16N_PRECNT is enabled.
MAT3S<1:0>	bit 15~14	R/W	**Operation select bits following a match between T16N_CNT0/ T16N_CNT1 and T16N_MAT3** 00: T16N_CNT0/ T16N_CNT1 keeps counting and no interrupt is generated. 01: T16N_CNT0/ T16N_CNT1 holds and generates an interrupt. 10: T16N_CNT0/ T16N_CNT1 is cleared and restarts counting, and generates an interrupt. 11: T16N_CNT0/ T16N_CNT1 keeps counting and generates an interrupt.
MAT2S<1:0>	bit 13~12	R/W	**Operation select bits following a match between T16N_CNT0/ T16N_CNT1 and T16N_MAT2** 00: T16N_CNT0/ T16N_CNT1 keeps counting and no interrupt is generated. 01: T16N_CNT0/ T16N_CNT1 holds and generates an interrupt. 10: T16N_CNT0/ T16N_CNT1 is cleared and restarts counting, and generates an interrupt. 11: T16N_CNT0/ T16N_CNT1 keeps counting and generates an interrupt.

MAT1S<1:0>	bit 11~10	R/W	**Operation select bits following a match between T16N_CNT0 and T16N_MAT1** 00: T16N_CNT0 keeps counting and no interrupt is generated. 01: T16N_CNT0 holds and generates an interrupt. 10: T16N_CNT0 is cleared and restarts counting, and generates an interrupt. 11: T16N_CNT0 keeps counting and generates an interrupt.
MAT0S<1:0>	bit 9~8	R/W	**Operation select bits following a match between T16N_CNT0 and T16N_MAT0** 00:T16N_CNT0 keeps counting and no interrupt is generated. 01:T16N_CNT0 holds and generates an interrupt. 10:T16N_CNT0 is cleared and restarts counting, and generates an interrupt. 11:T16N_CNT0 keeps counting and generates an interrupt.
MOD<1:0>	bit 7~6	R/W	**Operating mode select bits** 00: Timer/counter mode 01: Timer/counter mode 10: Capture mode 11: PWM mode
EDGE<1:0>	bit 5~4	R/W	**External clock counter edge select bits** 00: Rising edge 01: Falling edge 10: Both rising and falling edges (for synchronous counter mode only) 11: Both rising and falling edges (for synchronous counter mode only)
SYNC	bit 3	R/W	**External clock synchronize bit** 0: The external clock T16N0CK0/T16N0CK1 is not synchronized, and the counter is in asynchronous mode. 1: The external clock T16N0CK0/T16N0CK1 is synchronized with the PCLK, and the counter is in synchronous mode. The high and low level of the external clock must stay a minimum of 2 PCLK clock cycles.
CS<1:0>	bit 2~1	R/W	**T16N counter clock source select bits** 00: Internal clock PCLK 01: External clock T16N0CK0 10: External clock T16N0CK1 11: Internal clock PCLK

EN	bit 0	R/W	**T16N enable bit** 0: Disabled 1: Enabled

Note: The PCLK is the internal clock for peripheral modules, and its frequency is the same with the system clock frequency.

T16N Control Register1(T16N_CON1)

Offset address:14_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CAPL1	CAPL0	CAPT<3:0>				CAPIS1	CAPIS0	CAPNE	CAPPE

—	bit31-10	—	—
CAPL1	bit9	R/W	**Reload counter enable bit on capture input 1** 0:Disabled 1:Enabled
CAPL0	bit8	R/W	**Reload counter enable bit on capture input 0** 0: Disabled 1: Enabled
CAPT<3:0>	bit7-4	R/W	**Capture times select bits** 0: Capture once and reload 1: Capture twice and reload 2: Capture thrice and reload …… F: Capture 16 times and reload
CAPIS1	bit3	R/W	**Capture input source T16N0IN1 enable bit** 0: Disabled 1: Enabled
CAPIS0	bit2	R/W	**Capture input source T16N0IN0 enable bit** 0: Disabled 1: Enabled
CAPNE	bit1	R/W	**Capture falling edge enable bit** 0: Disabled 1: Enabled
CAPPE	bit0	R/W	**Capture rising edge enable bit** 0: Disabled 1: Enabled

T16N Control Register 2 (T16N_CON2)

Offset address:18_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	24	23	22	21	20	19	18	17	16
Reserved						PWMBKF	PWMBKP1	PWMBKP0	Reserved		PWMBKL1	PWMBKL0	PWMBKE1	PWMBKE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MOM3<1:0>		MOM2<1:0>		MOM1<1:0>		MOM0<1:0>		Reserved	PWMDZE	PWMMOD<1:0>		POL1	POL0	MOE1	MOE0

—	31-25	—	—
PWMBKF	bit24	R/W	**PWM break event flag bit** 0: No break event occurred 1: A break event occurred and the PWM ports outputs the break level. This flag bit is cleared by software writing 1. The PWM port resumes normal output after clearing.
PWMBKP1	bit23	R/W	**PWM channel 1 break polarity select bit** 0: Active high 1: Active low
PWMBKP0	bit22	R/W	**PWM channel 0 break polarity select bit** 0: Active high 1: Active low
—	bit21-20	—	—
PWMBKL1	bit19	R/W	**PWM channel 1 break output select bit** When POL1=0 0: Output low 1: Output high When POL1=1 0: Output high 1: Output low
PWMBKL0	bit18	R/W	**PWM channel 0 break output select bit** When POL0=0 0: Output low 1: Output high When POL0=1 0: Output high 1: Output low
PWMBKE1	bit17	R/W	**PWM channel 1 break enable bit** 0: Disabled 1: Enabled

PWMBKE0	bit16	R/W	**PWM channel 0 break enable bit** 0: Disabled 1: Enabled
MOM3<1:0>	bit15-14	R/W	**Operation select bits following a match with T16N_MAT3 on T16N0OUT1** 00: Hold 01: Clear 10: Set 11: Toggle
MOM2<1:0>	bit13-12	R/W	**Operation select bits following a match with T16N_MAT2 on T16N0OUT1** 00: Hold 01: Clear 10: Set 11: Toggle
MOM1<1:0>	bit11-10	R/W	**Operation select bits following a match with T16N_MAT1 on T16N0OUT0** 00: Hold 01: Clear 10: Set 11: Toggle
MOM0<1:0>	bit9-8	R/W	**Operation select bits following a match with T16N_MAT0 on T16N0OUT0** 00: Hold 01: Clear 10: Set 11: Toggle
—	bit7	—	—
PWMDZE	bit6	R/W	**Dead zone enable bit in PWM complementary mode** 0: Disabled 1: Enabled
PWMMOD<1:0>	bit5-4	R/W	**PWM mode select bits** 0x: Independent mode 10: Synchronous mode 11: Complementary mode
POL1	bit3	R/W	**T16N0OUT1 output polarity select bit** 0: Positive 1: Negative
POL0	bit2	R/W	**T16N0OUT0 output polarity select bit** 0: Positive 1: Negative
MOE1	bit1	R/W	**T16N0OUT1 enable bit** 0: Disabled

			1: Enabled
MOE0	bit0	R/W	**T16N0OUT0 enable bit** 0: Disabled 1: Enabled

Note1: A break signal is selected from PA2/PA3/PA27/PA26 via GPIO_PAFUNC register, and only one can be selected at a time.

Note2: When break is enabled, if PWMBKF flag is 1, the MOE bit of the corresponding output is automatically cleared by hardware. After the flag bit is cleared, the MOE bit needs to be set to1 again by software for PWM ports to resume outputting.

T16N Interrupt Enable Register (T16N_IE)

Offset address:20_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						PBK1IE	PBK0IE	CAP1IE	CAP0IE	TOP1IE	TOP0IE	MAT3IE	MAT2IE	MAT1IE	MAT0IE

—	bit31-10	—	—
PBK1IE	bit9	R/W	**Break interrupt enable bit on PWM channel 1** 0: Disabled 1: Enabled
PBK0IE	bit8	R/W	**Break interrupt enable bit on PWM channel 0** 0: Disabled 1: Enabled
CAP1IE	bit7	R/W	**Capture interrupt enable bit on input port T16N0IN1** 0: Disabled 1: Enabled
CAP0IE	bit6	R/W	**Capture interrupt enable bit on input port T16N0IN0** 0: Disabled 1: Enabled
TOP1IE	bit5	R/W	**Interrupt enable bit on T16N_CNT1 matching TOP1** 0:Disabled 1:Enabled
TOP0IE	bit4	R/W	**Interrupt enable bit on T16N_CNT0 matching TOP0** 0:Disabled

			1:Enabled
MAT3IE	bit3	R/W	**MAT3 interrupt enable bit** 0:Disabled 1:Enabled
MAT2IE	bit2	R/W	**MAT2 interrupt enable bit** 0:Disabled 1:Enabled
MAT1IE	bit1	R/W	**MAT1 interrupt enable bit** 0:Disabled 1:Enabled
MAT0IE	bit0	R/W	**MAT0 interrupt enable bit** 0:Disabled 1:Enabled

T16N Interrupt Flag Register (T16N_IF)

Offset address:24_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						PBK1IF	PBK0IF	CAP1IF	CAP0IF	TOP1IF	TOP0IF	MAT3IF	MAT2IF	MAT1IF	MAT0IF

—	bit31-10	—	—
PBK1IF	bit9	R/W	**Break interrupt flag bit on PWM channel 1** 0: No break interrupt occurred 1: An interrupt occurred on channel 1 This flag bit is cleared by software writing 1, writing 0 has no effect.
PBK0IF	bit8	R/W	**Break interrupt flag bit on PWM channel 0** 0: No break interrupt occurred 1: An interrupt occurred on channel 0 This flag bit is cleared by software writing 1, writing 0 has no effect.
CAP1IF	bit7	R/W	**Capture interrupt flag bit on capture input port 1** 0: Capture failed on input port 1 1: Capture succeeded on input port 1 This flag bit is cleared by software writing 1, writing 0 has no effect.
CAP0IF	bit6	R/W	**Capture interrupt flag bit on capture input port 0**

			0: Capture failed on input port 0 1: Capture succeeded on input port 0 This flag bit is cleared by software writing 1, writing 0 has no effect.
TOP1IF	bit5	R/W	**Interrupt flag bit on T16N_CNT1 matching TOP1** 0: Not matched 1: Matched This flag bit is cleared by software writing 1, writing 0 has no effect.
TOP0IF	bit4	R/W	**Interrupt flag bit on T16N_CNT0 matching TOP0** 0: Not matched 1: Matched This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT3IF	bit3	R/W	**MAT3 interrupt flag bit** 0: The counter value did not match MAT3 1: The counter value matched MAT3 This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT2IF	bit2	R/W	**MAT2 interrupt flag bit** 0: The counter value did not match MAT2 1: The counter value matched MAT2 This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT1IF	bit1	R/W	**MAT1 interrupt flag bit** 0: The counter value did not match MAT1 1: The counter value matched MAT1 This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT0IF	bit0	R/W	**MAT0 interrupt flag bit** 0: The counter value did not match MAT0 1: The counter value matched MAT0 This flag bit is cleared by software writing 1, writing 0 has no effect.

Notes

1. In timer, counter, capture and PWM mode, the T16N_CNT0 can always be compared to T16N_MAT0/ T16N_MAT1/ T16N_MAT2/ T16N_MAT3. However, in PWM mode, if independent mode is configured, the T16N_CNT0 can only be compared with T16N_MAT0 and T16N_MAT1 while the T16N_CNT1 is compared with T16N_MAT2 andT16N_MAT3.
2. When T16N interrupt is disabled, the corresponding interrupt flag will be set if an interrupt is triggered, however, no interrupt request will be generated.

3. For all interrupt flag bits of T16N_IF register, they can be cleared by software writing 1 and writing 0 has no effect. When they are being read, 1 indicates that an interrupt has occurred.
4. When break in enabled (PWMBKE0 or PWMBKE1=1), the interrupt flag bit PBK0IF or PBK1IF of T16N_IF register will only be set when a break event has occurred.

PWM Dead Zone Register (T16N_PDZ)

Offset address:28_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PDZ<7:0>							

—	bit31-8	—	—
PDZ<7:0>	bit7-0	R/W	**PWM dead zone time select bits** 0x00: 1x counter clock cycle 0x01: 2x counter clock cycles …… 0xFF: 256x counter clock cycles

PWM Trigger Register (T16N_PTR)

Offset address:$2C_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								P1TOP1TRE	P1MAT3TRE	P1MAT2TRE	Reserved	P0TOP0TRE	P0MAT1TRE	P0MAT0TRE	Reserved

—	bit31-8	—	—
P1TOP1TRE	bit7	R/W	**TOP1 trigger enable bit on PWM channel 1** 0: Disabled 1: Enabled to trigger ADC converter
P1MAT3TRE	bit6	R/W	**MAT3 trigger enable bit on PWM channel 1** 0: Disabled 1: Enabled to trigger ADC converter
P1MAT2TRE	bit5	R/W	**MAT2 trigger enable bit on PWM channel 1** 0: Disabled

			1: Enabled to trigger ADC converter
—	bit4	—	—
P0TOP0TRE	bit3	R/W	**TOP0 trigger enable bit on PWM channel 0** 0: Disabled 1: Enabled to trigger ADC converter
P0MAT1TRE	bit2	R/W	**MAT1 trigger enable bit on PWM channel 0** 0: Disabled 1: Enabled to trigger ADC converter
P0MAT0TRE	bit1	R/W	**MAT0 trigger enable bit on PWM channel 0** 0: Disabled 1: Enabled to trigger ADC converter
—	bit0	—	—

T16N Counter Match Register0 (T16N_MAT0)

Offset address:30_H

Reset value:$00000000\_00000000\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT0<15:0>															

—	bit31-16	—	—
MAT0<15:0>	bit15-0	R/W	T16N match value 0

T16N Counter Match Register1 (T16N_MAT1)

Offset address:34_H

Reset value:$00000000\_00000000\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT1<15:0>															

—	bit31-16	—	—
MAT1<15:0>	bit15-0	R/W	T16N match value 1

T16N Counter Match Register2 (T16N_MAT2)

Offset address:38_H

Reset value:00000000_00000000_11111111_11111111$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT2<15:0>															

—	bit31-16	—	—
MAT2<15:0>	bit15-0	R/W	T16N match value2

T16N Counter Match Register3 (T16N_MAT3)

Offset address:$3C_H$

Reset value:00000000_00000000_11111111_11111111$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT3<15:0>															

—	bit31-16	—	—
MAT3<15:0>	bit15-0	R/W	T16N match value3

T16N_CNT0 Top Value Register0 (T16N_TOP0)

Offset address:40_H

Reset value:00000000_00000000_11111111_11111111$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOP0<15:0>															

—	bit31-16	—	—
TOP0<15:0>	bit15-0	R/W	T16N_CNT0 top value 0

T16N_CNT1 Top Value Register1 (T16N_TOP1)

Offset address:44_H

Reset value:$00000000\_00000000\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOP1<15:0>															

—	bit31-16	—	—
TOP1<15:0>	bit15-0	R/W	T16N_CNT1 top value 1

5. 1. 1. 7 T16N Application Notes

There are four 16-bit timers/counters are available, which are T16N0, T16N1, T16N2 and T16N3.

5. 1. 2 32-bit Timer/Counter T32N (T32N0)

5. 1. 2. 1 Overview

- 1x 8-bit configurable prescaler, the prescaled clock is used for timer/counter T32N_CNT
  - Prescaler clock source options: PCLK or T32N0CK0/T32N0CK1
  - Initial value of prescaler counter set by the T32N_PRECNT register
  - Prescaler ratio set by the T32N_PREMAT register
- 1x 32-bit configurable timer/counter register T32N_CNT
- Timer/Counter mode
  - 4x32-bit match registers T32N_MAT0/ T32N_MAT1/ T32N_MAT2/ T32N_MAT3 are available. Following operations are supported when a match occurs
    - Interrupt
    - T32N_CNT operating modes: hold, clear and continue to count
    - Operations on T32N0OUT0/T32N0OUT1 port: hold, clear, set and toggle
  - Input capture mode
    - Configurable capture edge
    - Selectable capture times
  - Output PWM mode

5. 1. 2. 2 Block Diagram

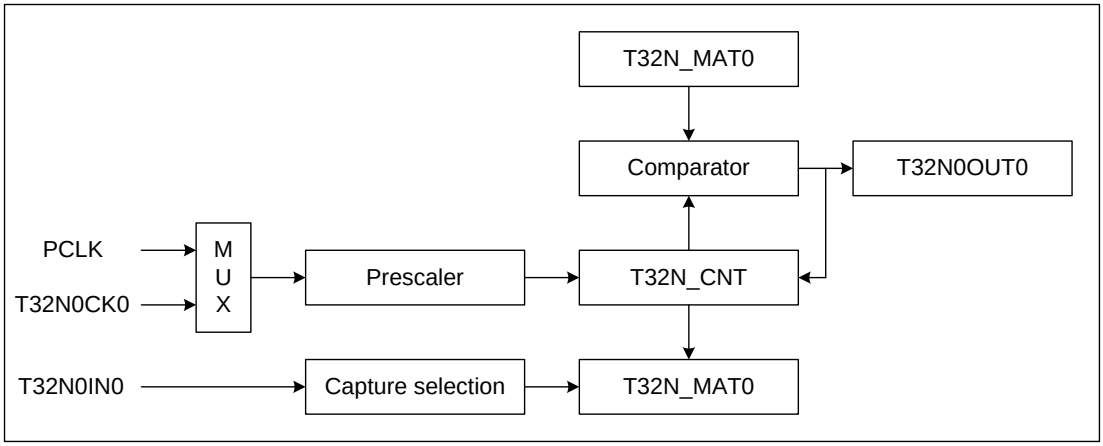


Figure 5-7 T32N0 Block Diagram

5. 1. 2. 3 T32N Timer/Counter

When the MOD<1:0> bits of the T32N_CON0 register is 00 or 01, the T32N works in timer/counter mode.

When the EN bit of the T32N_CON0 register is set, T32N is enabled and the counter register T32N_CNT starts to increment from the initial value.

The CS<1:0> bits of the T32N_CON0 register select the counter clock source. When the internal clock PCLK is selected, T32N works in timer mode. When the external clock input T32N0CK0/T32N0CK1 is selected, T16N works in counter mode.

The SYNC bit of the T32N_CON0 register can be configured to synchronize the external clock T32N0CK0/T32N0CK1 with the internal clock PCLK. When synchronized, T32N works in synchronous counter mode. Otherwise, it works in asynchronous counter mode. In synchronous counter mode, the pulse width of the high/low level of the T32N0CK0/T32N0CK1 input must be greater than 2 PCLK clock cycles.

The counter can be configured to count at rising edge, falling edge or both rising and falling edges by the EDGE<1:0> bits of the T32N_CON0 register. Counting at both rising and falling edge is only active in synchronous counter mode.

The MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0> bits of the T32N_CON0 register select an operation of the T32N_CNT register following a match.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=00: when the counter value of the T32N_CNT matches the T32N_MAT0/ T32N_MAT1/T32N_MAT2/T32N_MAT3, it will continue to count and no interrupt will be generated. When reaching 0xFFFFFFFF, an overflow will occur and an interrupt will be generated, and it will restart from 0x00000000.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=01: when the counter value of the T32N_CNT matches the T32N_MAT0/ T32N_MAT1/T32N_MAT2/T32N_MAT3, the counter value will be held until the next prescaled counter clock arrives. Meanwhile, the counter will stop counting and an interrupt will be generated.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=10: when the counter value of the T32N_CNT matches the T32N_MAT0/ T32N_MAT1/T32N_MAT2/T32N_MAT3, the counter value will be cleared on the arrival of the next prescaled counter clock. Meanwhile, an interrupt will be generated and the counter will restart counting.

MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=11: when the counter value of the T32N_CNT matches the T32N_MAT0/ T32N_MAT1/T32N_MAT2/T32N_MAT3, it will continue to count up and an interrupt will be generated on the arrival of the next prescaled counter clock. When reaching 0xFFFFFFFF, an overflow will occur and an interrupt will be generated, and it will restart from 0x00000000.

For different match values T32N_MAT0/ T32N_MAT1/T32N_MAT2/T32N_MAT3, when the counter value matches one of them, the corresponding interrupt will be generated. Following an interrupt, the T32N continues to count up, if the match interrupt flag has not been read in time, it is possible to read several valid match interrupt flags at the same time.

When MOE=1, If the counter value of the T32N_CNT register matches T32N_MAT0/ T32N_MAT1/ T32N_MAT2/ T32N_MAT3, the T32N0OUT0 will be toggled.

Example: When the counter value of the T32N_CNT register matches the T32N_MAT0/ T32N_MAT1/ T32N_MAT2/ T32N_MAT3, any of the following operations can be carried out.

In T32N_CON0 register,

If MAT0<31:0>=0x00000002 and MAT0S<1:0>=00, it will continue to count up and no interrupt will be generated.

If MAT1<31:0>=0x00000004 and MAT1S<1:0>=11, it will continue to count and generate an interrupt.

If MAT2<31:0>=0x00000006and MAT2S<1:0>=10, the counter value will be cleared, an interrupt will be generated and the counter will restart counting.

When the prescaler ratio is set to 1:1, the internal PCLK is used. Figure below shows the counter match diagram.

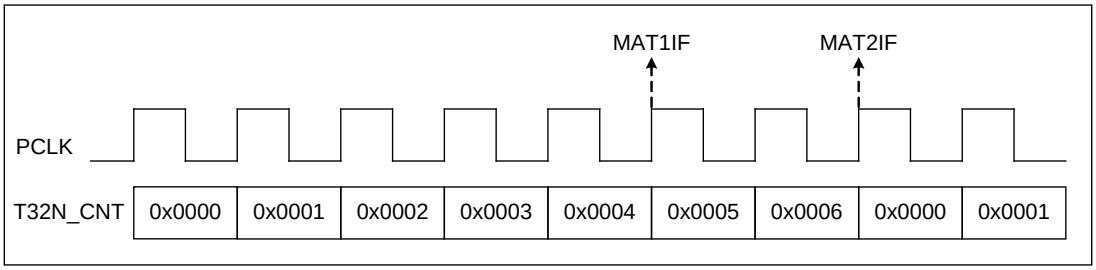


Figure 5-8 T32N0 Counter Match Diagram

5. 1. 2. 4 T32N Input Capture Mode

T32N works in capture mode by writing the MOD<1:0>=10 of the T32N_CON0 register.

In capture mode, writing CS<1:0>=00 in the T32N_CON0 register allows the T32N_CNT to use the PCLK as a counter clock. And when MAT0S/ MAT1S/ MAT2S/ MAT3S<1:0>=00, it does not affect the operation of T32N_CNT.

The following content describes the status detection of the T32N0IN0 and T32N0IN1 port.

When a capture event occurs on the T32N0IN0 port, load the current content of T32N_CNT0 and T32N_PRECNT to T32N_MAT0 and T32N_MAT2 respectively, and an interrupt CAP0IF occurs. The T32N_CNT and T32N_PRECNT can be cleared on a CAP0IF interrupt by enabling the CAPL0 of the T32N_CON1 register. If the CAPL0 is disabled, the counter continues to count up.

When a capture event occurs on the T32N0IN1 port, load the current content of T32N_CNT and T32N_PRECNT to T32N_MAT1 and T32N_MAT3 respectively, and an interrupt CAP1IF occurs. The T32N_CNT and T32N_PRECNT can be cleared on a CAP1IF interrupt by enabling the CAPL1 of the T32N_CON1 register. If the CAPL1 is disabled, the counter continues to count up.

If there has not been any capture event detected by the time when T32N_CNT overflows, the T32N_CNT will be cleared and restart counting.

The CAPPE and CAPNE bits of the T32N_CON1 register are used to select capture events on the T32N0IN0 and T32N0IN1 ports: rising edge, falling edge or both rising and falling edge.

The T32N0IN0 port can be enabled as a capture input by setting the CAPIS0 bit of the T32N_CON1 register; the T32N0IN1 port can also be enabled as a capture input by setting the CAPIS1 bit. Both ports can be enabled at the same time.

The number of times of a capture can be configured by the CAPT<3:0> bits of the T32N_CON1 register.

When MOE0=1, if a capture event occurs on T32N0IN0 port, the output T32N0OUT0 will be toggled.

When MOE1=1, if a capture event occurs on T32N0IN1 port, the output T32N0OUT1 will be toggled.

The configuration below shows how to capture both rising and falling edges 8 times on the T32N0IN0 with prescaler ratio 1:1

In T32N_CON0 register,

Write MOD<1:0>=10, CS<1:0>=00 and MAT0S<1:0>=00.

In T32N_CON1 register,

Write CAPPE=1, CAPNE=1, CAPIS=1 and CAPT<3:0>=0111.

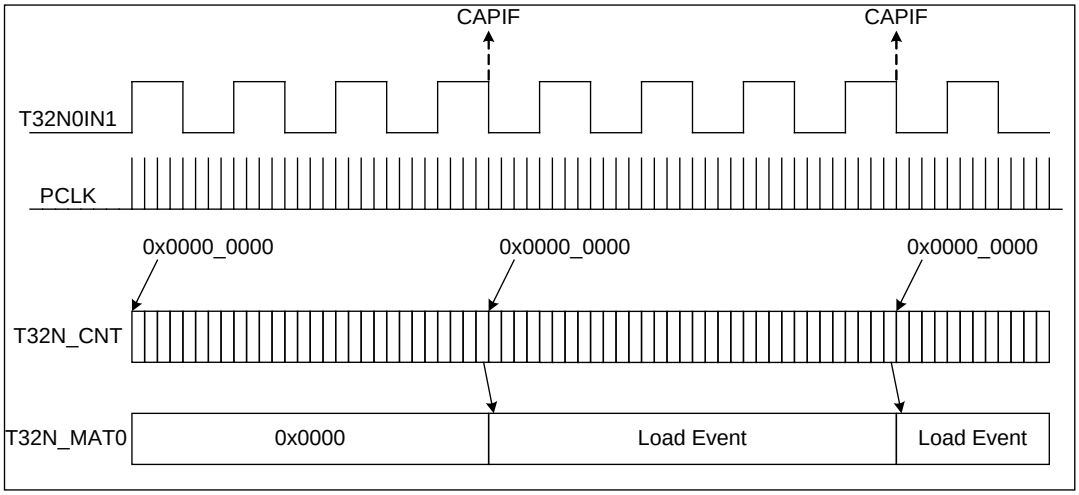


Figure 5-9 T32N0 Capture Diagram

In capture mode, the prescaler counter will not be cleared when altering the T32N_PREMAT register. Therefore, the first capture can be started from a non-zero prescaler counter value. When a capture event occurs, the interrupt flag must be cleared by software, and the captured value in T32N_MAT0/ T32N_MAT1/ T32N_MAT2/ T32N_MAT3 register must be read in time. Otherwise, the old captured value will be overwritten by the new captured value when the next capture occurs.

5. 1. 2. 5 T32N Output PWM Mode

The T32N works in PWM mode by writing MOD<1:0>=11 in the T32N_CON0 register.

In PWM mode, writing CS<1:0>=00 of the T32N_CON0 register allows the T32N_CNT

to use the internal PCLK as a counter clock.

The T32N0OUT0 port can be enabled by setting the MOE0 bit of the T32N_CON1 register. After enabling, a compare can be performed between T32N_MAT0/ T32N_MAT1 and T32N_CNT. The T32N0OUT1 port can be enabled by setting the MOE1 bit of the T32N_CON1 register. After enabling, a compare can be performed between T32N_MAT2/ T32N_MAT3 and T32N_CNT.

Configure the MOM0/ MOM1/ MOM2/ MOM3<1:0> bits to select one of the operations, hold, clear, set and toggle, on the T32N0OUT0/ T32N0OUT1 port when a match occurs.

The following configuration shows double-edge PWM waveforms generated on T32N0OUT0 and T32N0OUT1 ports.

In the T32N_CON1 register,

Write MOE0=1 and MOE1=1 to enable both T32N0OUT0 and T32N0OUT1 ports

Write MOM0<1:0>=10, the T32N0OUT0 outputs high.

Write MOM1<1:0>=01, the T32N0OUT0 outputs low.

Write MOM2<1:0>=10, the T32N0OUT1 outputs high.

Write MOM3<1:0>=01, the T32N0OUT1 outputs low.

Write MAT0 = 0x00000002 and MAT1 = 0x00000004.

Write MAT2 = 0x00000006 and MAT3 = 0x00000008.

In the T32N_CON0 register,

Write MOD<1:0>=11 to select the PWM mode

Write MAT0S<1:0>=11, the T32N_CNT keeps counting and generates an interrupt.

Write MAT1S<1:0>=11, the T32N_CNT keeps counting and generates an interrupt.

Write MAT2S<1:0>=11, the T32N_CNT keeps counting and generates an interrupt.

Write MAT3S<1:0>=10, the T32N_CNT is cleared and generates an interrupt.

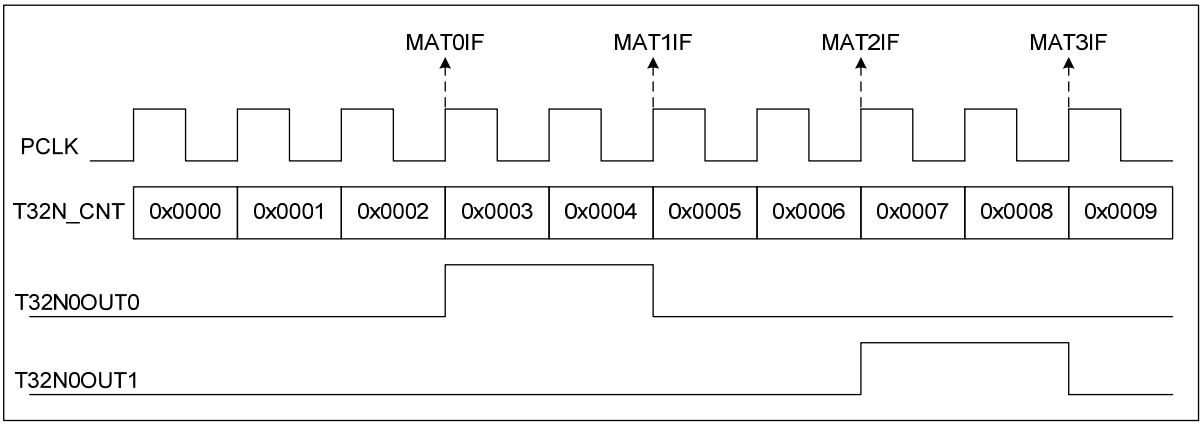


Figure 5-10 T32N0 Output PWM Diagram

Note: The match event of T32N_MAT0 has higher priority than the one of T32N_MAT1. If a same value is set to T32N_MAT0 and T32N_MAT1, the output level of T32N0OUT0 is depends on MOM0 only.
The match event of T32N_MAT2 has higher priority than the one of T32N_MAT3. If a same value is set to T32N_MAT2 and T32N_MAT3, the output level of T32N0OUT1 is depends on MOM2 only.

5. 1. 2. 6 Special Function Registers

T32N Counter Register (T32N_CNT)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
CNT<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CNT<15:0>															

CNT<31:0>	bit 31-0	R/W	T32N counter value

T32N Control Register0 (T32N_CON0)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															ASYNCWREN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT3S<1:0>		MAT2S<1:0>		MAT1S<1:0>		MAT0S<1:0>		MOD<1:0>		EDGE<1:0>		SYNC	CS<1:0>		EN

—	bit31-17	—	—
ASYNC_WREN	bit16	R/W	**In synchronous counter mode, counter write enable bit**

			0: Writing to T32N_CNT and T32N_PRECNT is disabled. (To avoid a write error to the counter, users are not suggested to write 0 to this bit) 1: Writing to T32N_CNT and T32N_PRECNT is enabled
MAT3S<1:0>	bit15-14	R/W	**Operation select bits following a match between T32N_CNT and T32N_MAT3** 00:T32N_CNT keeps counting and no interrupt is generated. 01:T32N_CNT holds and generates an interrupt. 10:T32N_CNT is cleared and restarts counting, and generates an interrupt. 11:T32N_CNT keeps counting and generates an interrupt.
MAT2S<1:0>	bit13-12	R/W	**Operation select bits following a match between T32N_CNT and T32N_MAT2** 00:T32N_CNT keeps counting and no interrupt is generated. 01:T32N_CNT holds and generates an interrupt. 10:T32N_CNT is cleared and restarts counting, and generates an interrupt. 11:T32N_CNT keeps counting and generates an interrupt.
MAT1S<1:0>	bit11-10	R/W	**Operation select bits following a match between T32N_CNT and T32N_MAT1** 00:T32N_CNT keeps counting and no interrupt is generated. 01:T32N_CNT holds and generates an interrupt. 10:T32N_CNT is cleared and restarts counting, and generates an interrupt. 11:T32N_CNT keeps counting and generates an interrupt.
MAT0S<1:0>	bit9-8	R/W	**Operation select bits following a match between T32N_CNT and T32N_MAT0** 00:T32N_CNT keeps counting and no interrupt is generated. 01:T32N_CNT holds and generates an interrupt. 10:T32N_CNT is cleared and restarts counting, and generates an interrupt. 11:T32N_CNT keeps counting and generates an interrupt.
MOD<1:0>	bit7-6	R/W	**Operating mode select bits** 00: Timer/counter mode 01: Timer/counter mode

			10: Capture mode 11: PWM mode
EDGE<1:0>	bit5-4	R/W	**External clock counter edge select bits** 00:Rising edge 01:Falling edge 10:Both rising and falling edges (for synchronous counter mode only) 11:Both rising and falling edges (for synchronous counter mode only)
SYNC	bit3	R/W	**External clock synchronize bit** 0: The external clock T32N0CK0/ T32N0CK1 is not synchronized, and the counter is in asynchronous mode. 1: The external clock T32N0CK0/ T32N0CK1 is synchronized with the PCLK, and the counter is in synchronous mode. The high and low level of the external clock must stay a minimum of 2 PCLK clock cycles.
CS<1:0>	bit2-1	R/W	**T32N counter clock source select bits** 00: Internal clock PCLK 01: External clock T32N0CK0 input 10: External clock T32N0CK1 input 11: Internal clock PCLK
EN	bit0	R/W	**T32N enable bit** 0: Disabled 1: Enabled

T32N Control Register1(T32N_CON1)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MOM3<1:0>		MOM2<1:0>		MOM1<1:0>		MOM0<1:0>		Reserved						MOE1	MOE0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						CAPL1	CAPL0	CAPT<3:0>				CAPIS1	CAPIS0	CAPNE	CAPPE

MOM3<1:0>	bit31-30	R/W	**Operation select bits following a match with T32N_MAT3 on T32N0OUT1** 00: Hold 01: Clear 10: Set 11: Toggle

MOM2<1:0>	bit29-28	R/W	**Operation select bits following a match with T32N_MAT2 on T32N0OUT1** 00: Hold 01: Clear 10: Set 11: Toggle
MOM1<1:0>	bit27-26	R/W	**Operation select bits following a match with T32N_MAT1 on T32N0OUT0** 00: Hold 01: Clear 10: Set 11: Toggle
MOM0<1:0>	bit25-24	R/W	**Operation select bits following a match with T32N_MAT0 on T32N0OUT0** 00: Hold 01: Clear 10: Set 11: Toggle
—	bit23-18	—	—
MOE1	bit17	R/W	**Output port T32N0OUT1 enable bit** 0: Disabled 1: Enabled
MOE0	bit16	R/W	**Output port T32N0OUT0 enable bit** 0: Disabled 1: Enabled
—	bit15-10	—	—
CAPL1	bit9	R/W	**Reload counter enable bit on capture input 1** 0:Disabled 1:Enabled
CAPL0	bit8	R/W	**Reload counter enable bit on capture input 0** 0:Disabled 1:Enabled
CAPT<3:0>	bit7-4	R/W	**Capture times select bits** 0: Capture once and reload 1: Capture twice and reload 2: Capture thrice and reload …… F: Capture 16 times and reload
CAPIS1	bit3	R/W	**Capture input port T32N0IN1 enable bit** 0: Disabled 1: Enabled
CAPIS0	bit2	R/W	**Capture input port T32N0IN0 enable bit** 0: Disabled 1: Enabled

CAPNE	bit1	R/W	**Capture falling edge enable bit** 0: Disabled 1: Enabled
CAPPE	bit0	R/W	**Capture rising edge enable bit** 0: Disabled 1: Enabled

T32N Prescaler Counter Register (T32N_PRECNT)

Offset address:10_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PRECNT<7:0>							

—	bit31-8	—	—
PRECNT<7:0>	bit7-0	R/W	T32N prescaler counter value

T32N Prescaler Counter Match Register (T32N_PREMAT)

Offset address:14_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PREMAT<7:0>							

—	bit31-8	—	—
PREMAT<7:0>	bit7-0	R/W	**Prescaler ratio select bits** 00:1:1 01:1:2 02:1:3 …… FE:1:255 FF:1:256

T32N Interrupt Enable Register (T32N_IE)

Offset address:18_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CAP1IE	CAP0IE	IE	MAT3IE	MAT2IE	MAT1IE	MAT0IE

—	bit31-7	—	—
CAP1IE	bit6	R/W	**Capture interrupt enable bit on input port T32N0IN1** 0:Disabled 1:Enabled
CAP0IE	bit5	R/W	**Capture interrupt enable bit on input port T32N0IN0** 0:Disabled 1:Enabled
IE	bit4	R/W	**Match 0xFFFFFFFF interrupt enable bit** 0:Disabled 1:Enabled
MAT3IE	bit3	R/W	**MAT3 interrupt enable bit** 0:Disabled 1:Enabled
MAT2IE	bit2	R/W	**MAT2 interrupt enable bit** 0:Disabled 1:Enabled
MAT1IE	bit1	R/W	**MAT1 interrupt enable bit** 0:Disabled 1:Enabled
MAT0IE	bit0	R/W	**MAT0 interrupt enable bit** 0:Disabled 1:Enabled

T32N Interrupt Flag Register (T32N_IF)

Offset address:$1C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									CAP1IF	CAP0IF	IF	MAT3IF	MAT2IF	MAT1IF	MAT0IF

—	bit31-7	—	—
CAP1IF	bit6	R/W	**Capture interrupt flag bit on capture input port 1** 0: Capture failed on input port 1 1: Capture succeeded on input port 1 This flag bit is cleared by software writing 1, writing 0 has no effect.
CAP0IF	bit5	R/W	**Capture interrupt flag bit on capture input port 0** 0: Capture failed on input port 1 1: Capture succeeded on input port 1 This flag bit is cleared by software writing 1, writing 0 has no effect.
IF	bit4	R/W	**Match 0xFFFFFFFF interrupt flag bit** 0: The counter value did not match 0xFFFFFFFF 1: The counter value matched 0xFFFFFFFF This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT3IF	bit3	R/W	**MAT3 interrupt flag bit** 0: The counter value did not match MAT3 1:The counter value matched MAT3 This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT2IF	bit2	R/W	**MAT2 interrupt flag bit** 0: The counter value did not match MAT2 1: The counter value matched MAT2 This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT1IF	bit1	R/W	**MAT1 interrupt flag bit** 0: The counter value did not match MAT1 1: The counter value matched MAT1 This flag bit is cleared by software writing 1, writing 0 has no effect.
MAT0IF	bit0	R/W	**MAT0 interrupt flag bit**

			0: The counter value did not match MAT0 1: The counter value matched MAT0 This flag bit is cleared by software writing 1, writing 0 has no effect.

Notes

1. In timer, counter, capture and PWM mode, the T32N_CNT can always be compared to the T32N_MAT0/ T32N_MAT1/ T32N_MAT2/ T32N_MAT3.
2. When T32N interrupts are disabled, the corresponding interrupt flag will be set if an interrupt is triggered, however, no interrupt request will be generated.
3. For all interrupt flag bits of T32N_IF register, they can be cleared by software writing 1 and writing 0 has no effect. When they are being read, 1 indicates that an interrupt has occurred.

T32N Counter Match Register 0 (T32N_MAT0)

Offset address:20_H

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAT0<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT0<15:0>															

MAT0<31:0>	bit31-0	R/W	T32N match value 0

T32N Counter Match Register 1 (T32N_MAT1)

Offset address:24_H

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAT1<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT1<15:0>															

MAT1<31:0>	bit31-0	R/W	T32N match value 1

T32N Counter Match Register 2(T32N_MAT2)

Offset address:28_H

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAT2<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT2<15:0>															

MAT2<31:0>	bit31-0	R/W	T32N match value 2

T32N Counter Match Register 3(T32N_MAT3)

Offset address:$2C_H$

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
MAT3<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
MAT3<15:0>															

MAT3<31:0>	bit31-0	R/W	T32N match value 3

5. 1. 2. 7 T32N Application Notes

The device offers one 32-bit timer/counter T32N0.

5. 2 Universal Asynchronous Receiver Transmitter

The device provides two identical UARTs, and this section uses UART0 as an example.

5. 2. 1 Overview

◇ Asynchronous communication

◇ Built-in baud rate generator: 12-bit integer and 4-bit fraction

◇ Compatible with RS-232/RS-442/RS-485 interfaces

◇ Full and half duplex communications

◇ Asynchronous receiver

- Independent receive shift register
- Automatic baud rate detection by hardware
- 8-level receive buffer
- 7-bit, 8-bit and 9-bit data length options with configurable odd/even parity check
- Automatic parity bit detection by hardware
- Idle frame detection
- FIFO interrupts: byte full interrupt, half-word full interrupt, word full interrupt and full interrupt
- Receive error interrupts: receive FIFO overrun error, parity error and framing error

◇ Asynchronous transmitter

- Independent transmit shift register
- 8-level transmit FIFO
- 7-bit, 8-bit and 9-bit data length options with configurable odd/even parity check
- 1 or 2 stop bits
- Automatic generation of parity bit (odd or even) by hardware
- FIFO interrupts: byte empty interrupt, half-word empty interrupt, word empty interrupt and empty interrupt
- Transmit FIFO write error interrupt

◇ PWM output with configurable duty cycle

◇ Configurable polarity for UART inputs and outputs

◇ Infrared wake-up capability on UART receive ports

5. 2. 2 Block Diagram

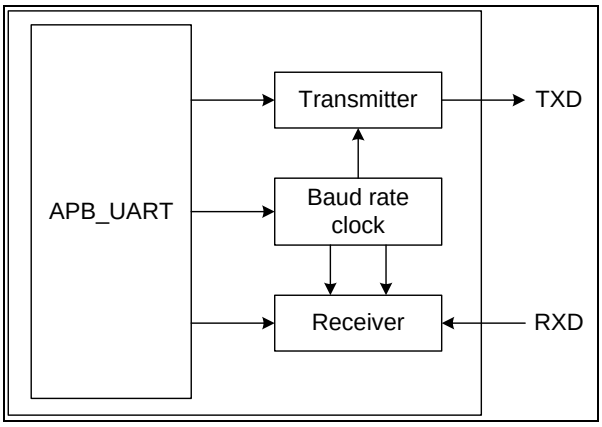


Figure 5-11 UART Block Diagram

5. 2. 3 Data Format

In UART communication, each frame consists of 1 start bit, 7 (8 or 9) data bits, parity bit and stop bit(s). TXMOD<3:0> bits and RXMOD<3:0> bits of UART_CON0 register select the transmit data format and receive data format, respectively. TXFS bit selects 1 stop bit or 2 stop bits. When receiving, only the first stop bit is checked. A framing error interrupt flag is set if the stop bit is not high. The communication pins are in high state when there is no data transfer.

Figures below show formats of 7-bit, 8-bit and 9-bit data frames.

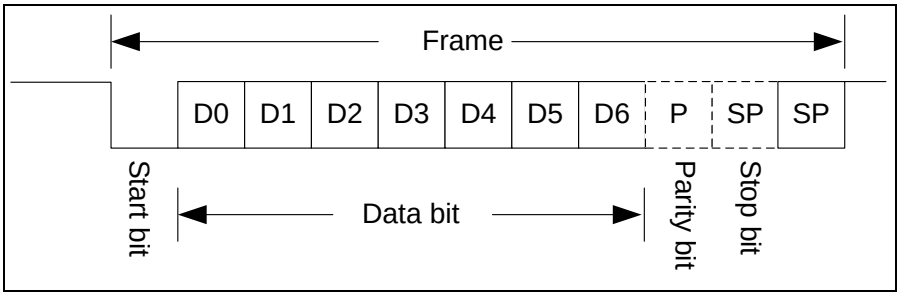


Figure 5-12 UART 7-bit Data Format

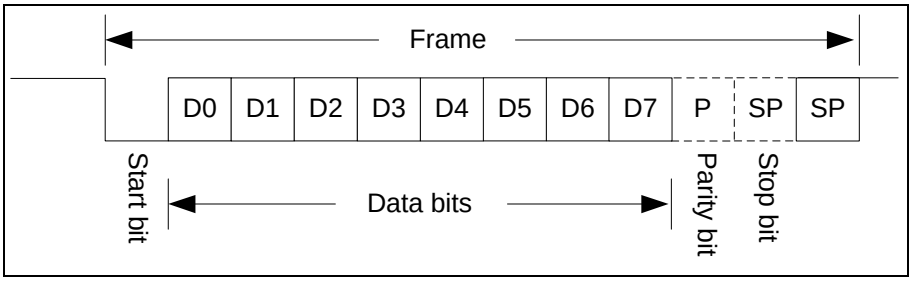


Figure 5-13 UART 8-bit Data Format

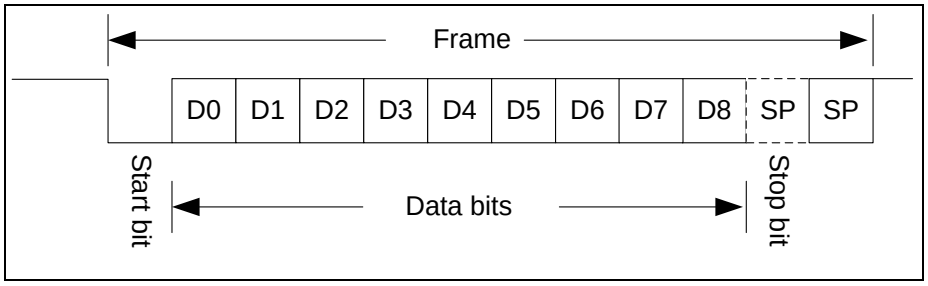


Figure 5-14 UART 9-bit Data Format

During data transmission and reception, the least significant bit LSB bit is always sent and received first. Write a data to be transmitted to UART_TBW register and read a received data from UART_RBR register.

5. 2. 4 Asynchronous Transmitter

When transmitting data, the start bit and stop bit are automatically generated by the hardware circuit. All that the user needs to do is to configure the corresponding functions on the associated I/O pins: set the baud rate by configuring the BCS<2:0> bits of UART_CON0 register and UART_BRR register; select the transmit data format by configuring the TXMOD<3:0> bits of UART_CON0 register; select the number of stop bit by configuring the TXFS bit; set the TXEN to enable the transmitter and write the data to be transmitted into the UART_TBW register to start the data transmission in asynchronous mode. If the data format involves odd/even parity bit, the hardware circuit will generate a parity bit depending on the data bits and will automatically sent it after data.

The TXP bit of UART_CON0 register selects the polarity of the transmit port. When the positive polarity is selected, the data output from the transmit port is non-inverted; when the negative polarity is selected, the transmit data will be inverted. For example, when the transmit data is 1, it will be inverted to 0 on the transmit port.

The 8-level transmit buffers TB0~TB7 and one transmit shift register are available to accomplish the continuous data transmission and up to 9 frames of data can be written and sent. The TXFS bit of the UART_CON0 register can select the transmission interval between two frames. The transmit buffer register TB0~TB7 are read-only and can only be written via the UART_TBW register.

The transmit buffer write register UART_TBW is a virtual register and has no physical address. When writing to the address of this register, it actually writes the data into a transmit buffer TB0~TB7, and then the data is shifted to the transmit shift register and finally sent via the transmit port TX0.

The transmit buffer write register UART_TBW can be written in byte, half-word and word.

For 7-bit and 8-bit data formats, when the UART_TBW register is written in byte, the transmit data will be written into TB7; when the UART_TBW register is written in half-word, the transmit data will be written into TB7 and TB6 with the low byte stored in TB6; when the UART_TBW register is written in word, the transmit data will be written

into TB7, TB6, TB5 an TB4 with the lowest byte stored in TB4.

For 9-bit data format, the UART_TBW register can only be written either in half-word or word and the transmit data is all written into TB7.

Figure below shows the flow of data transmission.

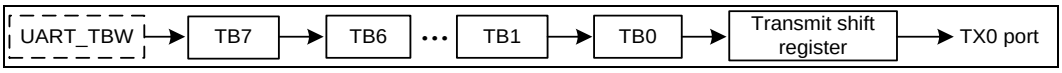


Figure 5-15 UART0 Transmit Data Flow

When transmit buffers TB0~TB7 have been written, the corresponding full flag bits TBFF0~TBFF7 of the UART_TB0~UART_TB7 register will be set automatically by hardware. After the data has been shifted to the next buffer or the transmit shift register, the flag bits will be automatically cleared by hardware. If flag bit TBFF7 of the TB7 is set (full), it indicates that all the 8-level transmit buffers and the transmit shift register are full. At this point, if the user continues to write a new data to the UART_TBW, the write overrun interrupt flag bit TBWOIF will be set, the new data written is invalid and the data in the buffers are maintained. When accessed in byte but the byte written is not the low byte, or when accessed in half word but the half word written is the low half word, the transmit buffer write error interrupt flag will be set, the new data written is invalid and the data in the buffers are maintained.

During transmission, the transmit status bit TXBUSY will be set to 1. When all the 8-level buffer registers and the transmit shift register are empty, the transmission complete interrupt flag bit TCIF of the UART_IF register will be set, and the TXBUSY bit will be cleared, indicating a completion of the current data transmission.

The TBIM<1:0> bits select the interrupt mode for the transmit buffer empty interrupt.

Write TBIM<1:0> = 00 to select the byte empty interrupt. When the TB7 is empty, the transmit buffer empty interrupt flag bit TBIF of the UART_IF register will be set.

Write TBIM<1:0> = 01 to select the half-word empty interrupt. When both TB7 and TB6 are empty, the transmit buffer empty interrupt flag bit TBIF of the UART_IF register will be set.

Write TBIM<1:0> = 10 to select the word empty interrupt. When TB7, TB6, TB5 and TB4 are empty, the transmit buffer empty interrupt flag bit TBIF of the UART_IF register will be set.

Write TBIM<1:0> = 11 to select the empty interrupt. When all the TB7 ~ TB0 are empty, the transmit buffer empty interrupt flag bit TBIF of the UART_IF register will be set.

The TBCLR bit of the UART_CON0 register can clear all the data of the transmit buffers, meanwhile the flags TBFF0~TBFF7 of the UART_TB0~UART_TB7 register can also be cleared. The cleared data of the transmit buffers will not be sent, however, the data of the shift register will still be sent.

The TRST bit of the UART_CON0 register can reset the transmitter by software.

Following a reset, the data transmission will be disabled (TXEN=0); the related interrupt enable bits of the UART_IE will also be disabled (TBIE=0 and TBWEIE=0); the related interrupt flag bits will be reset to their default values (TBIF=1 and TBWEIF=0); the transmit busy bit TXBUSY will be cleared and each transmit buffer flag bits TBFF0~TBFF7 will also be cleared.

The flowchart of the data transmission is shown as follows.

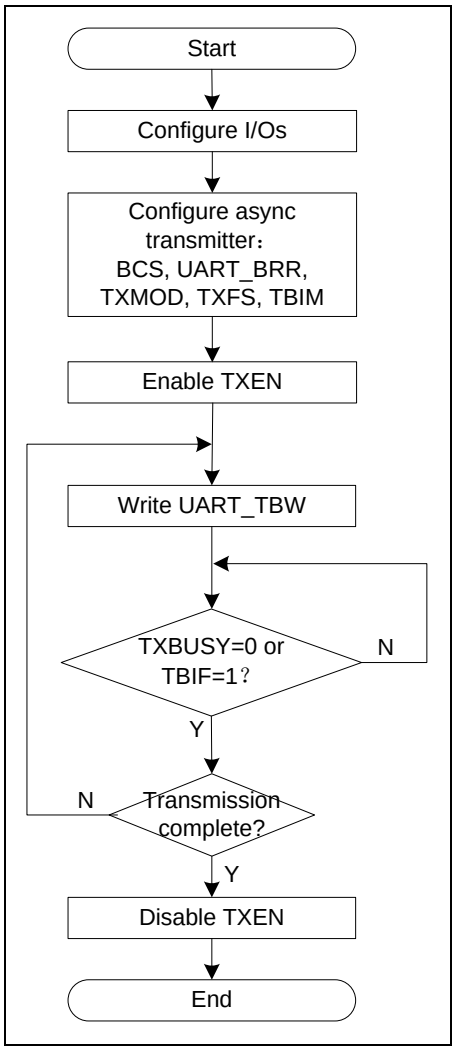


Figure 5-16 Flowchart of UART0 Data Transmission

5. 2. 5 Asynchronous Receiver

When receiving data, configure the corresponding functions on the associated I/O pins: set the baud rate by configuring the BCS<2:0> bits of UART_CON1 register and UART_BRR register; select the receive data format by configuring the RXMOD<3:0> bits and set the RXEN to enable the receiver to start data reception in asynchronous mode. If the data format involves odd/even parity bit, the hardware circuit will automatically check the parity. The parity error flag bit PEIF will be set if the detected parity bit is incorrect. If the first stop bit received is not at high level, the framing error flag bit FEIF of the UART_IF register will be set.

The RXP bit of UART_CON0 register selects the polarity of the receive port. When the positive polarity is selected, the received data on the receive port is non-inverted; when the negative polarity is selected, the received data will be inverted. For example, when the received data is 1, it will be inverted to 0 on the receive port.

The 8-level receive buffers RB0~RB7 and one receive shift register are available to accomplish continuous data reception. A read operation can proceed after up to 9 frames of data are received. Reading the receive buffer read register UART_RBR can obtain the received data and can also clear the corresponding full flag bit RBFF0~RBFF7 of the UART_RB0~ UART_RB7 register. Alternatively, reading the receive buffer register RB0~RB7 can obtain the received data too but it will not clear the corresponding flag bit RBFF0~RBFF7.

The receive buffer read register UART_RBR is a virtual register and has no physical address. When reading this register, it actually reads the data of a receive buffer RB0~RB7.

The receive buffer read register UART_RBR can be read in byte, half-word and word.

For 7-bit and 8-bit data formats, when the UART_RBR register is read in byte, it actually reads the data of RB0; when the UART_RBR register is read in half-word, it actually reads the data of RB0 and RB1 with the low byte stored in RB0; when the UART_RBR register is read in word, it actually reads the data of RB0, RB1, RB2 an RB3 concurrently with the lowest byte stored in RB0.

For 9-bit data format, the UART_RBR register can only be read either in half-word or word and the received data is all read from RB0.

Figure below shows the flow of data reception

Figure 5-17 UART0 Receive Data Flow

When the data of a receive buffer has been shifted to the next buffer, the corresponding receive full flag bit will be cleared.

When all the 8-level receive buffers and one receive shift register are full, if another start bit is received, the receive overrun interrupt flag bit ROIF will be set, meanwhile, no more new data will be accepted and the current data of the buffers are maintained.

When all the 8-level receive buffers and one receive shift register are empty, the receive busy flag bit RXBUSY will be cleared, indicating there is no data reception.

The RBIM<1:0> bits select the interrupt mode for the receive buffer full interrupt.

Write RBIM<1:0> = 00 to select the byte full interrupt. When the RB0 is full, the receive buffer full interrupt flag bit RBIF of the UART_IF register will be set.

Write RBIM<1:0> = 01 to select the half-word full interrupt. When both RB0 and RB1 are

full, the receive buffer full interrupt flag bit RBIF of the UART_IF register will be set.

Write RBIM<1:0> = 10 to select the word full interrupt. When RB0, RB1, RB2 and RB3 are full, the receive buffer full interrupt flag bit RBIF of the UART_IF register will be set.

Write RBIM<1:0> = 11 to select the full interrupt. When all the RB0 ~ RB7 are full, the receive buffer full interrupt flag bit RBIF of the UART_IF register will be set.

The RBCLR bit of the UART_CON0 register can clear all the receive buffers, meanwhile the full flag bits RBFF0~RBFF7 can also be cleared. Those data that is already in receiving process will not be affected.

The RRST bit of the UART_CON0 register can reset the receiver by software. Following a reset, the data reception will be disabled (RXEN=0); the related interrupt enable bits of the UART_IE will also be disabled (RBIE=0, ROIE=0, FEIE=0 and PEIE=0); the related interrupt flag bits will be reset to their default values (RBIF=0, ROIF=0, FEIF=0 and PEIF=0); the receive busy bit RXBUSY will be cleared; each receive buffer full flag bits RBFF0~RBFF7 will also be cleared, as well as the receive buffer error flag bits: framing error and parity error (FE0~FE7=0, PE0~ PE0=0)

The flowchart of the data reception is shown as follows.

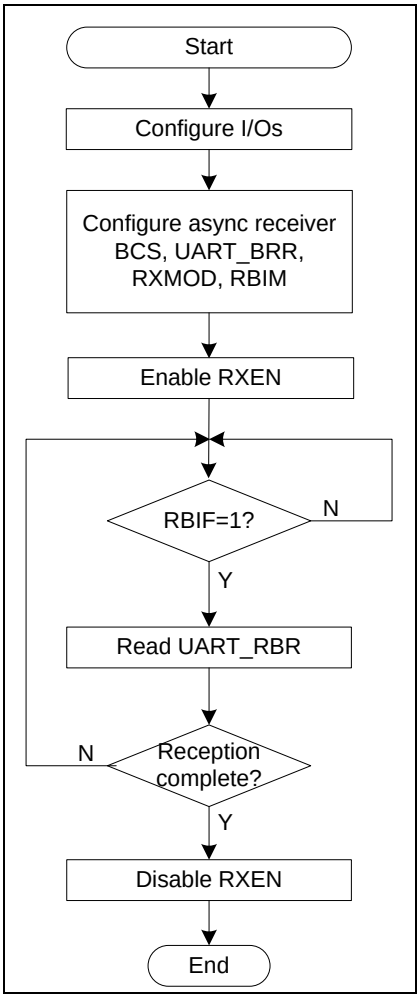


Figure 5-18 Flowchart of UART0 Data Reception

5. 2. 6 UART Transmit Modulation

In modulation mode, a PWM signal source generated by T16N or a BUZ signal modulates the signal transmitted by UART, and the modulated signal is then output from the transmit port TX0. Configure the TX0PS bits to select the PWM signal. The TX0PLV bit of the GPIO_TXPWM register selects the level to be modulated on the transmit port TX0. Configure the TX0_S0, TX0_S1, TX0_S2 and TX0_S3 to enable the modulation output port.

A PWM signal source used for UART modulation is provided by T16N or BUZ. Configure the TX0PS bits to select the PWM source to modulate the TX0 output.

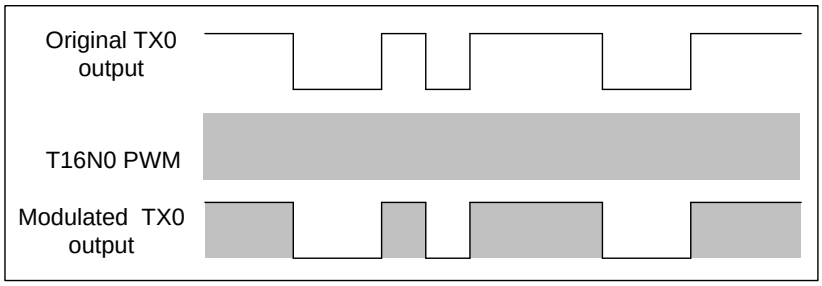


Figure 5-19 High Level Modulated TX0 Output

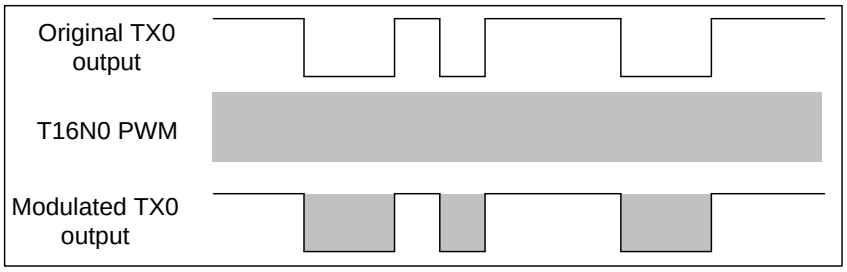


Figure 5-20 Low Level Modulated TX0 Output

5. 2. 7 UART Infrared Wake-up

The UART receive ports are designed with infrared wake-up function, which is accomplished by software by enabling the external port interrupt PINT on the UART receive ports. For the detailed operations for interrupts and wake-up, see the related sections on external port interrupt, normal sleep mode and wake-up mode.

5. 2. 8 UART Port Polarity

The TXP and RXP bits of the UART_CON0 register controls the polarity of the transmit port TX0 and receive port RX0, respectively. When the positive is selected, the transferred data will not be inverted on UART ports; when the negative is selected, the transferred data will be inverted on UART ports.

5. 2. 9 UART Auto Baud Rate Detection

On receiving data, set the BDEN bit of the UART_CON0 register to enable the automatic baud rate detection, and configure the BDM<1:0> bits to select the desired detection mode.

In the UART_CON1 register,

Write BDM<1:0>=00 to select Mode 1. In Mode 1, the received data flow needs to begin with 1 in binary (that is to say the lowest bit of the received data is 1B), and UART detects the baud rate of the start bit.

Write BDM<1:0>=01 to select Mode 2. In Mode 2, the received data flow needs to begin with 10 in binary (that is to say the lowest 2 bits of the received data are 01B), and UART detects the baud rate of the start bit and the first data bit.

Write BDM<1:0>=10 to select Mode 3. In Mode 3, the received data flow needs to begin with 1111_1110 in binary (that is to say the first frame of the received data is $7F_H$), and UART detects the baud rate of the start bit and the first 7 data bits.

Write BDM<1:0>=11 to select Mode 4. In Mode 4, the received data flow needs to begin with 1010_1010 in binary (that is to say the first frame of the received data is 55H), and UART detects the baud rate of the start bit and the first 7 data bits.

Following completion of the baud rate detection, the BDEN bit will be automatically cleared by hardware and the auto baud rate detection function will be disabled. If the detection is successful, the baud rate register UART_BRR will be updated, and the frame will be written into the receive buffer for users to read. If the detection times out, the baud detection error interrupt flag BDEIF in the UART_IF register will be set, consequently, the baud rate register will not be updated and the new data will not be written into the receive buffer.

In the case that the 7-bit data format is used or the parity bit after 7 data bits is not a fixed 0, Mode 3 or Mode 4 will not be applicable; instead, Mode 1 or Mode 2 must be selected. For other data formats, as long as it meets the requirement for any of the modes, the suitable mode can be configured.

The baud rate detection is not affected by the initial baud rate setting. However, a proper prescaled baud clock must be selected by the BCS<2:0> bits based on the specific application.

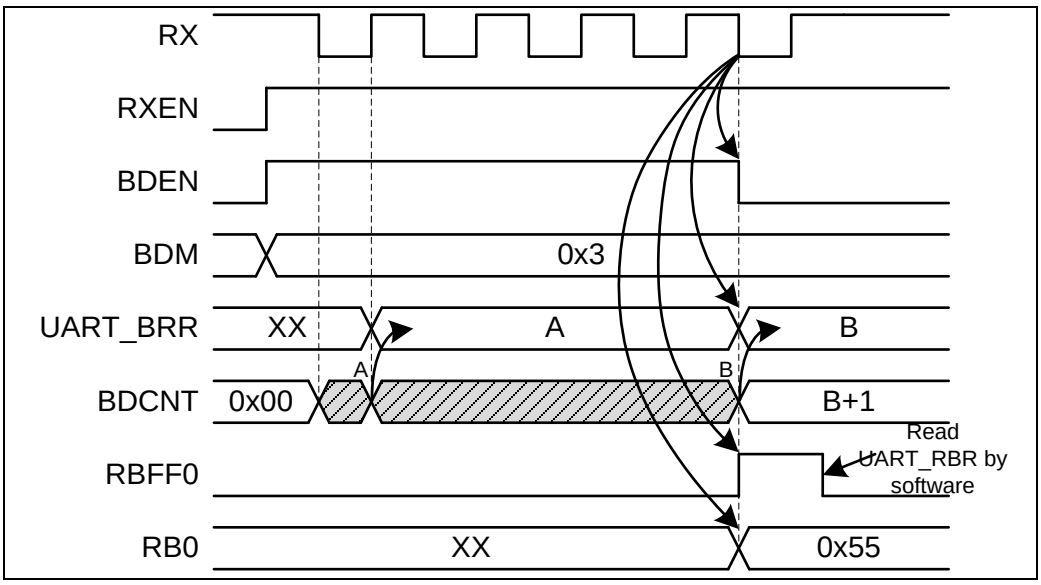


Figure 5-21 Timing Diagram of Auto Baud Rate Detection

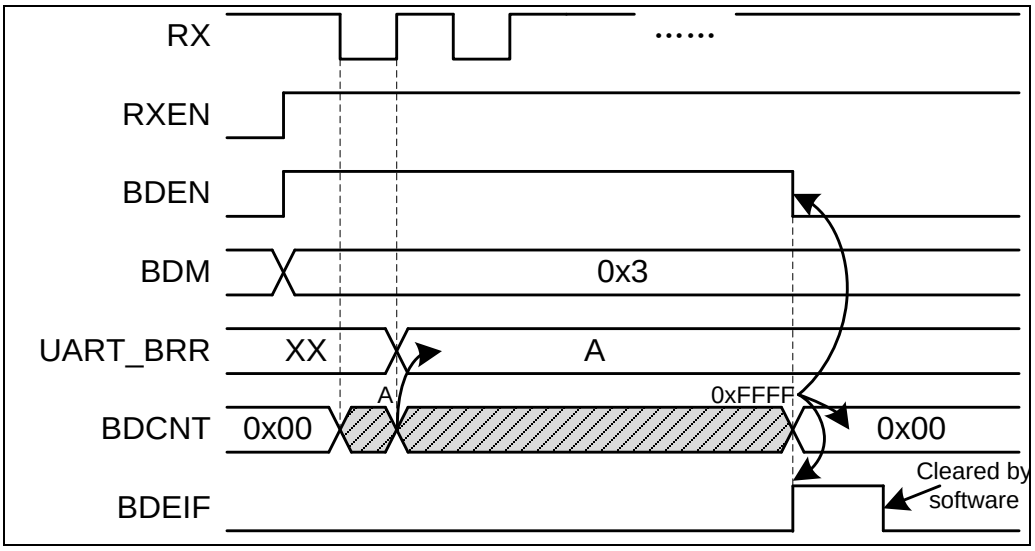


Figure 5-22 Timing Diagram of Auto Baud Rate Detection Error

5. 2. 10 UART Idle Frame Detection

On receiving data, set the IDEN bit of the UART_CON0 register to enable the idle frame detection function, and configure the IDM<1:0> bits of the UART_CON1 to select the detection mode.

Write IDM<1:0> = 00 to detect 10 continuous idle bits of the received frames.

Write IDM<1:0> = 01 to detect 11 continuous idle bits of the received frames.

Write IDM<1:0> = 10 to detect 12 continuous idle bits of the received frames.

Write IDM<1:0> = 11 to detect 13 continuous idle bits of the received frames.

When an idle frame has been detected, the receive idle frame interrupt flag bit IDIF bit will be set. The idle frame detection will only be triggered after hardware receiving the data. When the idle frame detection is enabled, if the receive line stays idle, it will not

affect the idle frame interrupt flag bit.

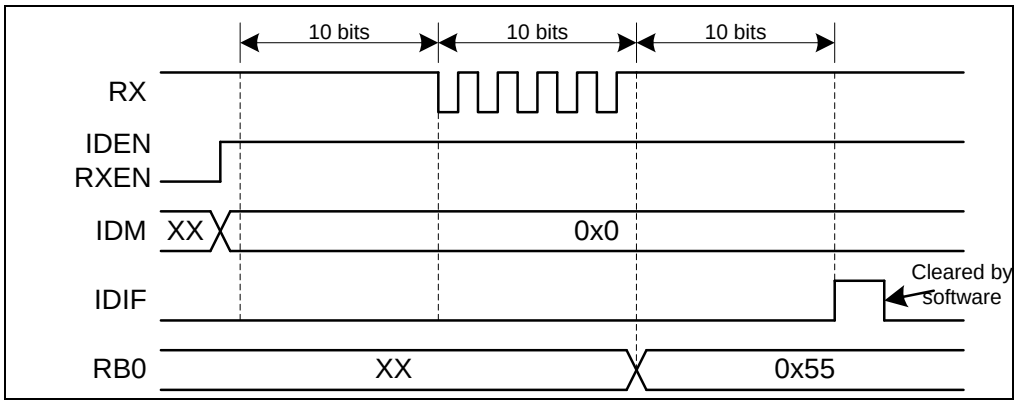


Figure 5-23 Timing Diagram of Idle Frame Detection

5. 2. 11 UART Transmission and Reception Halt

Enabling the TXI bit can halt the data transmission. The TX0 port will be at idle level and the data of the transmit buffers will remain unchanged. If the transmission is currently on-going, the transmission will be halted after the current transmission is done.

Enabling the RXI bit can halt the data reception, and the data of the receive buffers will remain unchanged. If the reception is currently in process, then the current frame will be discarded. The parity check and the idle frame detection will not be affected after the data reception is halted.

5. 2. 12 Special Function Registers

UART Control Register 0(UART_CON0)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			RXP	RXMOD<3:0>				Reserved		IDEN	BDEN	RXI	RBCLR	RRST	RXEN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TXFS	TXP	TXMOD<3:0>				Reserved				TXI	TBCLR	TRST	TXEN

—	bit31-29	—	—
RXP	bit28	R/W	**Receive port polarity select bit** 0: Positive (non-inverted) 1: Negative (inverted)
RXMOD<3:0>	bit27-24	R/W	**Receive mode select bits** 0000: 8-bit data 0010: 9-bit data 0100: 7-bit data 1000: 8-bit data + even parity bit 1001: 8-bit data + odd parity bit

			1010: 8-bit data + 0 (fixed) 1011: 8-bit data +1 (fixed) 1100: 7-bit data + even parity bit 1101: 7-bit data + odd parity bit 1110: 7-bit data + 0(fixed) 1111: 7-bit data + 1(fixed) Others: Not used
—	bit23-22	—	—
IDEN	bit21	R/W	**Idle frame detection enable bit** 0: Disabled 1: Enabled
BDEN	bit20	R/W	**Auto baud rate detection enable bit** 0: Disabled 1: Enabled
RXI	bit19	R/W	**Reception halt enable bit** 0: Disabled 1: Enabled
RBCLR	bit18	W	**Receive buffer clear bit** 0: Always read as 0 1: Clear receive buffers
RRST	bit17	W	**Receiver software reset bit** 0: Always read as 0 1: Software reset
RXEN	bit16	R/W	**Receiver enable bit** 0: Disabled 1: Enabled
—	bit15-14	—	—
TXFS	bit13	R/W	**Transmit frame stop bit select bit** 0:1 stop bit 1:2 stop bits
TXP	bit12	R/W	**Transmit port polarity select bit** 0: Positive (non-inverted) 1: Negative (inverted)
TXMOD<3:0>	bit11-8	R/W	**Transmit mode select bits** 0000: 8-bit data 0010: 9-bit data 0100: 7-bit data 1000: 8-bit data + even parity bit 1001: 8-bit data + odd parity bit 1010: 8-bit data + 0 (fixed) 1011: 8-bit data + 1 (fixed) 1100: 7-bit data + even parity bit 1101: 7-bit data + odd parity bit 1110: 7-bit data + 0 (fixed)

			1111:7 -bit data + 1 (fixed) Others: Not used
—	bit7-4	—	—
TXI	bit3	R/W	**Transmission halt enable bit** 0: Disabled 1: Enabled
TBCLR	bit2	W	**Transmit buffer clear bit** 0: Always read as 0 1: Clear transmit buffers
TRST	bit1	W	**Transmitter software reset bit** 0: Always read as 0 1:Software reset
TXEN	bit0	R/W	**Transmitter enable bit** 0: Disabled 1: Enabled

UART Control Register 1(UART_CON1)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														IDM<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		BDM<1:0>		Reserved	BCS<2:0>			Reserved		RBIM<1:0>		Reserved		TBIM<1:0>	

—	bit31-18	—	—
IDM<1:0>	bit17-16	R/W	**Idle frame detection mode select bits** 00: 10 consecutive idle bits 01: 11 consecutive idle bits 10: 12 consecutive idle bits 11: 13 consecutive idle bits
—	bit15-14	—	—
BDM<1:0>	bit13-12	R/W	**Auto baud rate detection mode select bits** 00: Mode 1 01: Mode 2 10: Mode 3 11: Mode 4
—	bit11	—	—
BCS<2:0>	bit10-8	R/W	**Baud rate clock select bits** 000: Baud rate clock disabled 001: PCLK 010: PCLK/2

			011: PCLK/4 1xx: PCLK/8
—	bit7-6	—	—
RBIM<1:0>	bit5-4	R/W	**Receive buffer interrupt mode select bits** 00: Byte full interrupt (more than 1 byte in the receive buffers) 01: Half-word full interrupt (more than 2 bytes in the receive buffers) 10: Word full interrupt (more than 4 bytes in the receive buffers) 11: Full interrupt (all buffers full)
—	bit3-2	—	—
TBIM<1:0>	bit1-0	R/W	**Transmit buffer interrupt mode select bits** 00: Byte empty interrupt (more than 1 byte is empty in the transmit buffers) 01: Half-word empty interrupt (more than 2 bytes are empty in the transmit buffers) 10: Word empty interrupt (more than 4 bytes are empty in the transmit buffers) 11: Empty interrupt (all buffers empty)

Notes

1. Following is the mode description of the BDM bits of the UART_CON1 register.

   Mode 1: if the data stream starts with 1B (the lowest bit of the received data is 1B), check the baud rate of the start bit.

   Mode 2: if the data stream starts with 10B (the lowest 2 bits of the received data is 10B), check the baud rate of the start bit and the first data bit.

   Mode 3: if the data stream starts with 1111_1110B (the received data is 7FH), check the baud rate of the start bit and the first 7 data bits.

   Mode 4: if the data stream starts with 1010_1010B (the received data is 55H), check the baud rate of the start bit and the first 7 data bits.

2. If the received data is in 7-bit data format or the parity bit is not fixed to 0, mode 3 and mode 4 will not be applicable, and mode 1 and mode 2 must be selected. For other data formats, all modes are applicable.

UART Baud Rate Register (UART_BRR)

Offset address:10_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
BRINT<11:0>												BRFRA<3:0>			

—	bit31-16	—	—
BRINT<11:0>	bit15-4	R/W	**Baud rate integer bits**
BRFRA<3:0>	bit3-0	R/W	**Baud rate fraction bits**

Notes

1. The UART_BRR register contains the frequency division rate of the UART baud rate (BRRDIV), which is a 16-bit unsigned value, 12-bits integer and 4-bit fraction. Examples: given 0x0104 (260 in decimal), then BRRDIV=260/16=16.25; given 0x0156 (342 in decimal), then BRRDIV=342/16=21.375.
2. If the value in the UART_BRR register is less than 1.0, then BRRDIV =1.0.
3. The baud rate of UART is expressed as

$$BAUD = \frac{Fpclk}{16 \times n \times BRRDIV}$$

Where Fpclk is the system clock frequency, n is the baud prescaler ratio determined by BCS<2:0> of the UART_CON1 register

BCS<2:0>=001: n = 1

BCS<2:0>=010: n = 2

BCS<2:0>=011: n = 4

BCS<2:0>=1xx: n = 8

UART Status Register (UART_STA)

Offset address:14_H

Reset value:00000000_00000000_00000001_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								PER3	FER3	PER2	FER2	PER1	FER1	PER0	FER0

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		RXBUSY	RBOV	RBPTR<3:0>				Reserved		TXBUSY	TBOV	TBPTR<3:0>			

—	bit31-24	—	—
PER3	bit23	R	**Parity error of current BYTE3** 0: No error 1: An error occurred
FER3	bit22	R	**Framing error of current BYTE3**

			0: No error 1: An error occurred
PER2	bit21	R	**Parity error of current BYTE2** 0: No error 1: An error occurred
FER2	bit20	R	**Framing error of current BYTE2** 0: No error 1: An error occurred
PER1	bit19	R	**Parity error of current BYTE1** 0: No error 1: An error occurred
FER1	bit18	R	**Framing error of current BYTE1** 0: No error 1: An error occurred
PER0	bit17	R	**Parity error of current BYTE0** 0: No error 1: An error occurred
FER0	bit16	R	**Framing error of current BYTE0** 0: No error 1: An error occurred
—	bit15-14	—	**—**
RXBUSY	bit13	R	**Receiver busy bit** 0: Idle 1: Busy
RBOV	bit12	R	**Receive buffer overrun bit** 0: No overrun 1: An overrun occurred
RBPTR<3:0>	bit11-8	R	**Receive buffer pointer which points to the remaining bytes of received data** 0000:0 byte …… 1000:8 bytes
—	bit7-6	—	**—**
TXBUSY	bit5	R	**Transmitter busy bit** 0: Idle 1: Busy
TBOV	bit4	R	**Transmit buffer overrun bit** 0: No overrun 1: An overrun occurred
TBPTR<3:0>	bit3-0	R	**Transmit buffer pointer which points to the current byte of transmit data** 0000:0 byte ……

			1000:8 bytes

UART Interrupt Enable Register (UART_IE)

Offset address:18_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		RBROIE	RBREIE	BDEIE	PEIE	FEIE	ROIE	Reserved						IDIE	RBIE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TBWOIE	TBWEIE	Reserved						TCIE	TBIE

—	bit31-30	—	**—**
RBROIE	bit29	R/W	**Receive buffer read overrun interrupt enable bit** 0: Disabled 1: Enabled
RBREIE	bit28	R/W	**Receive buffer read error interrupt enable bit** 0: Disabled 1: Enabled
BDEIE	bit27	R/W	**Baud rate detection error interrupt enable bit** 0: Disabled 1: Enabled
PEIE	bit26	R/W	**Receive parity error interrupt enable bit** 0: Disabled 1: Enabled
FEIE	bit25	R/W	**Receive framing error interrupt enable bit** 0: Disabled 1: Enabled
ROIE	bit24	R/W	**Receive overrun interrupt enable bit** 0: Disabled 1: Enabled
—	bit23-18	—	**—**
IDIE	bit17	R/W	**Receive idle frame interrupt enable bit** 0: Disabled 1: Enabled
RBIE	bit16	R/W	**Receive buffer full interrupt enable bit** 0: Disabled 1: Enabled
—	bit15-10	—	**—**
TBWOIE	bit9	R/W	**Transmit buffer write overrun interrupt enable bit** 0: Disabled 1: Enabled
TBWEIE	bit8	R/W	**Transmit buffer write error interrupt enable bit** 0: Disabled

			1: Enabled
—	bit7-2	—	—
TCIE	bit1	R/W	**Transmission complete interrupt enable bit** 0: Disabled 1: Enabled
TBIE	bit0	R/W	**Transmit buffer empty interrupt enable bit** 0: Disabled 1: Enabled

UART Interrupt Flag Register (UART_IF)

Offset address:$1C_H$

Reset value:$00000000\_00000000\_00000000\_00000001_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		RBROIF	RBREIF	BDEIF	PEIF	FEIF	ROIF	Reserved						IDIF	RBIF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						TBWOIF	TBWEIF	Reserved						TCIF	TBIF

—	bit31-30	—	—
RBROIF	bit29	R/W	**Receive buffer read overrun interrupt flag bit** 0: No overrun 1: An overrun occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
RBREIF	bit28	R/W	**Receive buffer read error interrupt flag bit** 0: No read error 1: A read error occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
BDEIF	bit27	R/W	**Baud rate detection error interrupt flag bit** 0: No error 1: An error detected This flag bit is cleared by software writing 1 and writing 0 has no effect.
PEIF	bit26	R/W	**Receive parity error interrupt flag bit** 0: No parity error 1: A parity error occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
FEIF	bit25	R/W	**Receive framing error interrupt flag bit** 0: No framing error

			1: A framing error occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
ROIF	bit24	R/W	**Receive overrun interrupt flag bit** 0: No overrun 1: An overrun occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
—	bit23-18	—	**—**
IDIF	bit17	R/W	**Receive idle frame interrupt flag bit** 0: No idle frame received 1: An idle frame received This flag bit is cleared by software writing 1 and writing 0 has no effect.
RBIF	bit16	R	**Receive buffer full interrupt flag bit** 0: Not full 1: Full (any of the four conditions, selected by RBIM, is true) Read the UART_RBR register to clear this flag bit
—	bit15-10	—	**—**
TBWOIF	bit9	R/W	**Transmit buffer write overrun flag bit** 0: No overrun 1: An overrun occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
TBWEIF	bit8	R/W	**Transmit buffer write error interrupt flag bit** 0: No write error 1: A write error occurred This flag bit is cleared by software writing 1 and writing 0 has no effect.
—	bit7-2	—	**—**
TCIF	bit1	R/W	**Transmission complete interrupt flag bit** 0: No completed 1: Completed This flag bit is cleared by software writing 1 and writing 0 has no effect.
TBIF	bit0	R	**Transmit buffer empty interrupt flag bit** 0: Not empty 1: Empty (any of the four conditions, selected by TBIM, is true) Read the UART_TBW register to clear this flag bit

Notes

1. When the UART interrupts are disabled, the corresponding interrupt flag will be set if an interrupt is triggered, however, no interrupt request will be generated.
2. For each interrupt flag in the UART_IF register, they can be cleared by software writing 1 and writing 0 has no effect. When reading, 1 indicates that the corresponding interrupt has occurred.

UART Transmit Data Write Register (UART_TBW)

Offset address:20_H

Reset value: $XXXXXXXX\_XXXXXXXX\_XXXXXXXX\_XXXXXXXX_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBW<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW<15:0>															

TBW<31:0>	bit31-0	W	**Write transmit data** Write in byte: write to UART_TBW<7:0> Write in half-word: write to UART_TBW<15:0> Write in word: write to UART_TBW<31:0>

UART Receive Data Read Register(UART_RBR)

Offset address:24_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RBR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBR<15:0>															

RBR<31:0>	bit31-0	R	**Read receive data** Read in byte: read from UART_RBR<7:0> Read in half-word: read from UART_RBR<15:0> Read in word: read from UART_RBR<31:0>

UART Transmit Buffer Register 0 (UART_TB0)

Offset address:40_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF0	TP0	Reserved			TB0<8:0>								

—	bit31-14	—	**—**
TBFF0	bit13	R	**Transmit buffer 0 full flag bit** 0: Empty 1: Full
TP0	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 0.
—	bit11-9	—	**—**
TB0<8:0>	bit8-0	R	**Transmit buffer 0 data**

UART Transmit Buffer Register 1 (UART_TB1)

Offset address:44_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF1	TP1	Reserved			TB1<8:0>								

—	bit31-14	—	**—**
TBFF1	bit13	R	**Transmit buffer 1 full flag bit** 0: Empty 1: Full
TP1	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 1.
—	bit11-9	—	**—**
TB1<8:0>	bit8-0	R	**Transmit buffer 1 data**

UART Transmit Buffer Register 2 (UART_TB2)

Offset address:48_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF2	TP2	Reserved			TB2<8:0>								

—	bit31-14	—	**—**
TBFF2	bit13	R	**Transmit buffer 2 full flag bit** 0: Empty 1: Full
TP2	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 2.
—	bit11-9	—	**—**
TB2<8:0>	bit8-0	R	**Transmit buffer 2 data**

UART Transmit Buffer Register 3 (UART_TB3)

Offset address:$4C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF3	TP3	Reserved			TB3<8:0>								

—	bit31-14	—	**—**
TBFF3	bit13	R	**Transmit buffer 3 full flag bit** 0: Empty 1: Full
TP3	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 3
—	bit11-9	—	**—**
TB3<8:0>	bit8-0	R	**Transmit buffer 3 data**

UART Transmit Buffer Register 4 (UART_TB4)

Offset address:50_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF4	TP4	Reserved			TB4<8:0>								

—	bit31-14	—	**—**
TBFF4	bit13	R	**Transmit buffer 4 full flag bit** 0: Empty 1: Full
TP4	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 4
—	bit11-9	—	**—**
TB4<8:0>	bit8-0	R	**Transmit buffer 4 data**

UART Transmit Buffer Register 5 (UART_TB5)

Offset address:54_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF5	TP5	Reserved			TB5<8:0>								

—	bit31-14	—	**—**
TBFF5	bit13	R	**Transmit buffer 5 full flag bit** 0: Empty 1: Full
TP5	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 5
—	bit11-9	—	**—**
TB5<8:0>	bit8-0	R	**Transmit buffer 5 data**

UART Transmit Buffer Register 6 (UART_TB6)

Offset address:58_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF6	TP6	Reserved			TB6<8:0>								

—	bit31-14	—	**—**
TBFF6	bit13	R	**Transmit buffer 6 full flag bit** 0: Empty 1: Full
TP6	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 6
—	bit11-9	—	**—**
TB6<8:0>	bit8-0	R	**Transmit buffer 6 data**

UART Transmit Buffer Register 7 (UART_TB7)

Offset address:$5C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBFF7	TP7	Reserved			TB7<8:0>								

—	bit31-14	—	**—**
TBFF7	bit13	R	**Transmit buffer 7 full flag bit** 0: Empty 1: Full
TP7	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 7
—	bit11-9	—	**—**
TB7<8:0>	bit8-0	R	**Transmit buffer 7 data**

UART Receive Buffer Register 0 (UART_RB0)

Offset address:60_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE0	FE0	RBFF0	RP0	Reserved			RB0<8:0>								

—	bit31-16	—	**—**
PE0	bit15	R	**Receive buffer 0 parity error flag bit** 0: No error 1: Error occurred
FE0	bit14	R	**Receive buffer 0 framing error flag bit** 0: No error 1: Error occurred
RBFF0	bit13	R	**Receive buffer 0 full flag** 0: Empty 1: Full
RP0	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 0
—	bit11-9	—	**—**
RB0<8:0>	bit8-0	R	**Receive buffer 0 data**

UART Receive Buffer Register 1 (UART_RB1)

Offset address:64_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE1	FE1	RBFF1	RP1	Reserved			RB1<8:0>								

—	bit31-16	—	**—**
PE1	bit15	R	**Receive buffer 1 parity error flag bit** 0: No error 1: Error occurred
FE1	bit14	R	**Receive buffer 1 framing error flag bit** 0: No error 1: Error occurred

RBFF1	bit13	R	**Receive buffer 1 full flag** 0: Empty 1: Full
RP1	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 1
—	bit11-9	—	**—**
RB1<8:0>	bit8-0	R	**Receive buffer 1 data**

UART Receive Buffer Register 2 (UART_RB2)

Offset address:68_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE2	FE2	RBFF2	RP2	Reserved			RB2<8:0>								

—	bit31-16	—	**—**
PE2	bit15	R	**Receive buffer 2 parity error flag bit** 0: No error 1: Error occurred
FE2	bit14	R	**Receive buffer 2 framing error flag bit** 0: No error 1: Error occurred
RBFF2	bit13	R	**Receive buffer 2 full flag** 0: Empty 1: Full
RP2	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 2
—	bit11-9	—	**—**
RB2<8:0>	bit8-0	R	**Receive buffer 2 data**

UART Receive Buffer Register 3 (UART_RB3)

Offset address:$6C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE3	FE3	RBFF3	RP3	Reserved			RB3<8:0>								

—	bit31-16	—	**—**
PE3	bit15	R	**Receive buffer 3 parity error flag bit** 0: No error 1: Error occurred
FE3	bit14	R	**Receive buffer 3 framing error flag bit** 0: No error 1: Error occurred
RBFF3	bit13	R	**Receive buffer 3 full flag** 0: Empty 1: Full
RP3	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 3
—	bit11-9	—	**—**
RB3<8:0>	bit8-0	R	**Receive buffer 3 data**

UART Receive Buffer Register 4 (UART_RB4)

Offset address:70_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE4	FE4	RBFF4	RP4	Reserved			RB4<8:0>								

—	bit31-16	—	**—**
PE4	bit15	R	**Receive buffer 4 parity error flag bit** 0: No error 1: Error occurred
FE4	bit14	R	**Receive buffer 4 framing error flag bit** 0: No error 1: Error occurred

RBFF4	bit13	R	**Receive buffer 4 full flag** 0: Empty 1: Full
RP4	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 4
—	bit11-9	—	**—**
RB4<8:0>	bit8-0	R	**Receive buffer 4 data**

UART Receive Buffer Register 5 (UART_RB5)

Offset address:74_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE5	FE5	RBFF5	RP5	Reserved			RB5<8:0>								

—	bit31-16	—	**—**
PE5	bit15	R	**Receive buffer 5 parity error flag bit** 0: No error 1: Error occurred
FE5	bit14	R	**Receive buffer 5 framing error flag bit** 0: No error 1: Error occurred
RBFF5	bit13	R	**Receive buffer 5 full flag** 0: Empty 1: Full
RP5	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 5
—	bit11-9	—	**—**
RB5<8:0>	bit8-0	R	**Receive buffer 5 data**

UART Receive Buffer Register 6 (UART_RB6)

Offset address:78_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE6	FE6	RBFF6	RP6	Reserved			RB6<8:0>								

—	bit31-16	—	**—**
PE6	bit15	R	**Receive buffer 6 parity error flag bit** 0: No error 1: Error occurred
FE6	bit14	R	**Receive buffer 6 framing error flag bit** 0: No error 1: Error occurred
RBFF6	bit13	R	**Receive buffer 6 full flag bit** 0: Empty 1: Full
RP6	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 6
—	bit11-9	—	**—**
RB6<8:0>	bit8-0	R	**Receive buffer 6 data**

UART Receive Buffer Register 7 (UART_RB7)

Offset address:$7C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE7	FE7	RBFF7	RP7	Reserved			RB7<8:0>								

—	bit31-16	—	**—**
PE7	bit15	R	**Receive buffer 7 parity error flag bit** 0: No error 1: Error occurred
FE7	bit14	R	**Receive buffer 7 framing error flag bit** 0: No error 1: Error occurred
RBFF7	bit13	R	**Receive buffer 7 full flag bit**

			0: Empty 1: Full
RP7	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 7
—	bit11-9	—	**—**
RB7<8:0>	bit8-0	R	**Receive buffer 7 data**

5. 2. 13 UART Application Notes

The device offers two universal asynchronous receivers transmitters, UART0 and UART1.

The modulating signal from UART transmitter is modulated by a PWM signal generated by T16N or BUZ, and the modulated signal is output via TX0/TX1 port. For the PWM signal itself, whether it is output from T16N0OUT0, T16N0OUT1 or BUZ port has nothing to do with the modulation. That is to say, even if those ports are not multiplexed, the PWM signal can still modulate the output signal on TX0 and TX1 after proper configurations.

5. 3 Enhanced Universal Asynchronous Receiver Transmitter

5. 3. 1 Overview

- ◇ Can be configured as normal UART and compatible with all UART functions
- ◇ Supports 7816 mode and 7816 communication protocol
- ◇ Asynchronous receiver and transmitter
- ◇ Half duplex mode
- ◇ 8-bit data, 1 parity bit
- ◇ Auto re-transmission and re-reception
- ◇ Configurable internal clock output
- ◇ Configurable dual-channel communication

5. 3. 2 Block Diagram

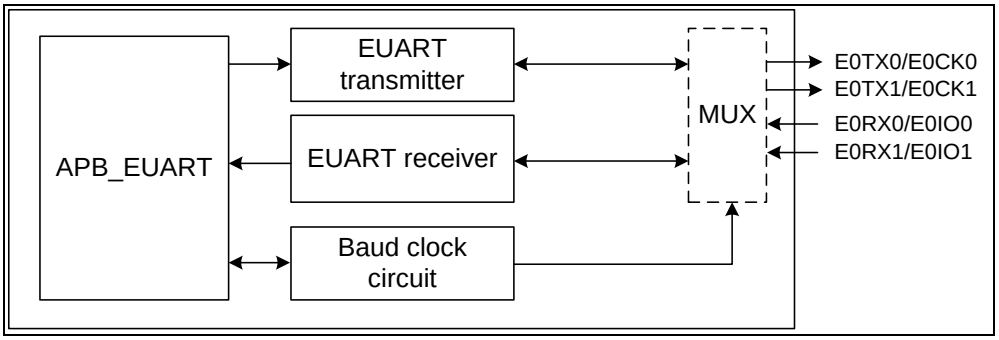


Figure 5-24 EUART0 Block Diagram

5. 3. 3 EUART Port Multiplexing

EUART Multiplexed Port	Normal UART Mode	7816 Mode
E0TX0/E0CK0	E0TX0	E0CK0
E0TX1/E0CK1	E0TX1	E0CK1
E0RX0/E0IO0	E0RX0	E0IO0
E0RX1/E0IO1	E0RX1	E0IO1

Configure the corresponding I/O ports for the EUART function.

In the EUART_CON2 register,

Write MOD =0 to select the normal UART mode, and the E0TX0/ E0TX1 and E0RX0/ E0RX1 function are selected for the ports.

Write MOD =1 to select the 7816 mode, and the E0CK0/ E0CK1 and E0IO0/ E0IO1 function are selected for the ports.

The EUART module has two independent internal clock output ports: E0CK0 and E0CK1. Setting the CK0E/ CK1E bit can enable the clock port E0CK0/ E0CK1 to output the internal clock. The CKS<1:0> bits select the desired clock source.

The EUART module has two data pins (E0IO0 and E0IO1), and they can realize dual-channel communication by using the time division multiplexing technique. Select the desired the E0IO0/ E0IO1pin by the CHS.

5. 3. 4 Normal UART Mode

When the EUART module is configured as a normal UART, it supports all UART functions except for PWM modulation. Users can refer to the related sections for details on normal UART mode.

5. 3. 5 Asynchronous Receiver and Transmitter in 7816 Mode

The 7816 mode works in half-duplex mode. In this mode, the RXEN and TXEN bits of the EUART_CON0 register are invalid. Instead, the receiver and transmitter are enabled by the I/O direction control bit IOC of the EUART_CON2 register. The IOC bit also transmits or receives the ACK signal and controls the input and output state of the E0IO0/ E0IO1 port.

When IOC = 1, the EUART transmits data and receives ACK signals. The data transmission format begins with one start bit, followed by 8 data bits (LSB first) and ends with one parity bit. It also receives ACK signals. When two continuous data frames are being sent back to back, the time interval between the two frames must be larger than certain time, which is called Guard Time. The ETUS<7:0> bits of the EUART_CON2 register select the desired guard time. An ACK signal will be received after each frame is sent. Reading the RNACK bit of the EUART_CON2 register can determine whether an ACK signal is received or not. An ACK signal is a high level signal while a NACK signal is a low level signal. If the ACK signal is not received within the guard time, the NACK interrupt flag bit RNAIF will be set.

When IOC = 0, the EUART receives data and transmits ACK signals. The reception starts with the start bit, data bits (8-bit, LSB first) followed by the parity bit. If the received parity bit is correct, an ACK signal will then be sent. If not, an NACK signal will be sent. An ACK signal is a high level signal while a NACK signal is a low level signal. The TNAS<1:0> bits select the pulse width of NACK signals.

Configure the PS bit to select the odd or even parity check. On transmission, hardware will automatically generate and send an odd or even parity bit based on the data to be transmitted. On reception, hardware will perform the parity check based on the received data. If the parity check fails, the parity error interrupt flag bit PEIF of the EUART_IF register will be set.

4 transmit buffers: TB0, TB1, TB2 and TB3, and one transmit shift register are available. All buffer registers are read-only, and they can only be written via the EUART_TBW register.

Figure below shows the flow of data transmission.

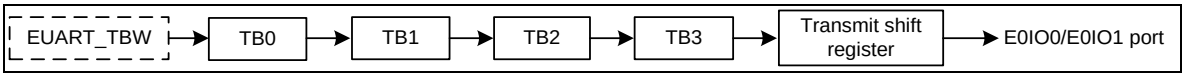


Figure 5-25 EUART Transmit Data Flow in 7816

4 receive buffers: RB0, RB1, RB2 and RB3, and one receive shift register are available. The received data can be obtained by reading the EUART_RBR register.

Figure below shows the flow of data reception.

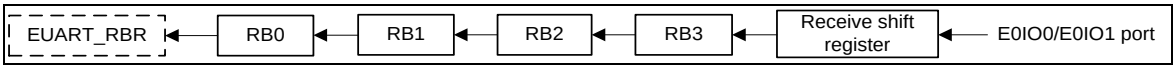


Figure 5-26 EUART Receive Data Flow in 7816

The ERST bit of the EUART_CON2 register is used to reset the 7816 module by software. Following an software reset, the data transfer will be disabled (TXEN=0 and RXEN=0); all associated interrupts will be disabled (TBIE=0, TBWEIE=0, RBIE=0, ROIE=0, FEIE=0, PEIE=0 and RNAIE=0); the related interrupt flag bits will be set to their default values (TBIF=1, BWEIF=0, RBIF=0, ROIF=0, FEIF=0, PEIF=0 and RNAIF=0); the busy flags will be cleared (TXBUSY=0 and RXBUSY=0); all transmit buffer empty flags will be set (TBEF0~TBEF3=1); all receive buffer full flags will be cleared (RBFF0~RBFF3=0); and error flags of all receive buffers will be cleared (FE0~FE3=0 and PE0~PE3=0).

5. 3. 6 Data Format in 7816 Mode

In 7816 mode, a data frame consists of an 8-bit data and 1 odd or even parity bit. Two data conventions are available, direct convention and inverse convention.

When DAS = 0 of the EUART_CON2 register, the direct convention is selected. The data written into the EUART_TBW register is identical with the actual data to be transmitted.

For example, write 0x50 into the EUART_TBW register in byte, clock it to the transmit shift register as 0x50 and generate a corresponding parity bit. The data sequence transmitted from the E0IO0/ E0IO port will be 0 (start bit) + 00001010 + parity bit. Similarly, if the data received from the E0IO0/E0IO1 port is 0x50, the actual data obtained by reading the EUART_RBR register is also 0x50.

When DAS = 1 of the EUART_CON2 register, the inverse convention is selected, where the data order is reversed and the data bit is in inverted polarity.

For example, write 0x50 (01010000 in binary) into the EUART_TBW register in byte. Reverse the data order to obtain 00001010, and invert the data bits to obtain 11110101 (0xF5), which will then be clocked to the transmit shift register. The corresponding parity bit will be generated based on 0xF5. The data sequence transmitted from the E0IO0/ E0IO1 port will be 0 (start bit) + 10101111 + parity bit. Similarly, if the data received at the E0IO0/ E0IO1 port is 0xF5, the actual data obtained by reading the EUART_RBR register is 0x50.

5. 3. 7 Auto Re-transmission in 7816 Mode

Auto re-transmission is supported in 7816 mode.

Write ARTE = 0 of the EUART_CON2 register to disable the auto-retransmission. When an NACK (ACK =0) is received, the NACK interrupt flag RNAIF will be set, as well as the RNACK (RNAIF =1, RNACK =1). After that, the next data frame will continue to be sent, and the RNACK will get updated accordingly, but the RNAIF will stay at 1 until the EUART is reset by software or the flag bit is cleared by software.

Here shows an example if the auto re-transmission is disabled during data transfer. Given that the transmit shift register holds the data 0x55 and the transmit buffer TB3 holds the data 0xAA, if the data 0x55 is successfully transmitted and an ACK is received, then the NACK interrupt flag RNAIF = 0 and RNACK = 0. In this case, the transmitter will go on to transmit 0xAA, and RNAIF and RNACK will get updated accordingly. If the transmitter fails to send 0x55 and an ACK is not received, then RNAIF=1, RNACK=1, and 0xAA will continue to be sent anyway. In this case, the RNACK will get updated accordingly but the RNAIF will stay at 1 no matter the 0xAA transmission succeeds or fails.

Write ARTE =1 of the EUART_CON2 register to enable the auto re-transmission. When an NACK is received, the previous data frame will be automatically re-transmitted, which will affect the RNACK but not the interrupt flag RNAIF. Configure ARTS<1:0> bits can select the number of times of re-transmission. During data re-transmission, when an ACK is received, the data re-transmission will be terminated. If the data transmission still fails by the time when it has reached the upper limit of re-transmission times, the current data and the following data transmission will be terminated, the auto re-transmission error interrupt flag ARTEIF will be set and the transmit busy flag TXBUSY will be cleared.

Here shows an example if the auto re-transmission is enabled during data transfer. Given that the transmit shift register holds the data 0x55, the transmit buffer TB3 holds the data 0xAA, and the number of times of re-transmission is 2, if the transmitter fails to send 0x55, then the transmit shift register still holds the data 0x55, the transmit buffer TB3 holds the data 0xAA, RNAIF=1, RNACK=1, ARTEIF=0 and TXBUSY=1. Thus, the data 0x55 will be re-transmitted for the first time. If an ACK is received, then RNACK =0, RNAIF =1, ARTEIF=0, TXBUSY=1, and the transmitter will proceed with the data 0xAA. If an NACK is received instead, then RNACK =1, RNAIF =1, ARTEIF=0, TXBUSY=1, and 0x55 will be re-transmitted for the second time. If the second attempt still fails, the data transmission will be terminated, RNACK =1, RNAIF =1, ARTEIF =1, TXBUSY =0, and the transmitter will not proceed with 0xAA.

5. 3. 8 Auto Re-reception in 7816 Mode

The auto re-reception is supported in 7816 mode.

In the EUART_CON2 register, write ARRE =0 to disable the auto re-reception function. An ACK will be sent within the guard time no matter the received parity bit is correct or not.

Write ARRE = 1 to enable the auto re-reception function. A NACK will be sent within the guard time if the received parity bit is incorrect. If the transmitter is able to re-transmit, the previous data frame will be sent after receiving the NACK.

5. 3. 9 Special Function Registers

EUART Control Register 0(EUART_CON0)

Offset address :00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			RXP	RXMOD<3:0>				Reserved					RBCLR	RRST	RXEN

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TXFS	TXP	TXMOD<3:0>				Reserved					TBCLR	TRST	TXEN

—	bit31-29	—	—
RXP	bit28	R/W	**Receive port polarity select bit** 0: Positive (non-inverted) 1: Negative (inverted)
RXMOD<3:0>	bit27-24	R/W	**Receive mode select bits** 0000: 8-bit data 0010: 9-bit data 0100: 7-bit data 1000: 8-bit + even parity bit 1001: 8-bit + odd parity bit 1010: 8-bit + 0 (fixed) 1011: 8-bit + 1 (fixed) 1100: 7-bit + even parity bit 1101: 7-bit + odd parity bit 1110: 7-bit + 0 (fixed) 1111: 7-bit + 1 (fixed) Others: Not used
—	bit23-19	—	—
RBCLR	bit18	W	**Receive buffer clear bit** 0: Always read as 0 1: Clear receive buffer
RRST	bit17	W	**Receiver software reset bit** 0: Always read as 0 1: Software reset
RXEN	bit16	R/W	**Receive enable bit** 0: Disabled 1: Enabled
—	bit15-14	—	—
TXFS	bit13	R/W	**Transmit frame stop bit select**

			0: 1 stop bit 1: 2 stop bits
TXP	bit12	R/W	**Transmit port polarity select bit** 0: Positive (non-inverted) 1: Negative (inverted)
TXMOD<3:0>	bit11-8	R/W	**Transmit mode select bits** 0000: 8-bit data 0010: 9-bit data 0100: 7-bit data 1000: 8-bit + even parity bit 1001: 8-bit + odd parity bit 1010: 8-bit + 0 (fixed) 1011: 8-bit + 1 (fixed) 1100: 7-bit + even parity bit 1101: 7-bit + odd parity bit 1110: 7-bit + 0 (fixed) 1111: 7-bit + 1 (fixed) Others: Not used
—	bit7-3	—	—
TBCLR	bit2	W	**Transmit buffer clear bit** 0: Always read as 0 1: Clear transmit buffer
TRST	bit1	W	**Transmitter software reset bit** 0: Always read as 0 1: Software reset
TXEN	bit0	R/W	**Transmit enable bit** 0: Disabled 1: Enabled

EUART Control Register 1 (EUART_CON1)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														RXBUSY	TXBUSY

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BCS<2:0>			Reserved		RBIM <1:0>		Reserved		TBIM<1:0>	

—	bit31-18	—	—
RXBUSY	bit17	R	**Receiver busy bit** 0: Idle 1: Busy

TXBUSY	bit16	R	**Transmitter busy bit** 0: Idle 1: Busy
—	bit15-11	—	**—**
BCS<2:0>	bit10-8	R/W	**Baud rate clock select bits** 000: Baud rate clock disabled 001: PCLK 010: PCLK/2 011: PCLK/4 1xx: PCLK/8
—	bit7-6	—	**—**
RBIM <1:0>	bit5-4	R/W	**Receive buffer full interrupt mode select bits** 00: Byte full interrupt (buffers contain more than one byte) 01: Half-word full interrupt (buffers contain more than two bytes) 1x: Full interrupt (all buffers are full)
—	bit3-2	—	**—**
TBIM<1:0>	bit1-0	R/W	**Transmit buffer empty interrupt mode select bits** 00: Byte empty interrupt (more than one byte is empty in buffers) 01: Half-word empty interrupt (more than two bytes are empty in buffers) 1x: Empty interrupt (all buffers are empty)

EUART Control Register 2 (EUART_CON2)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		TXFEND	RNACK	Reserved			BGTE	ETUS<7:0>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
CKS<1:0>		ARTS<1:0>		TNAS<1:0>		ARRE	ARTE	PS	DAS	IOC	CHS	CK1E	CK0E	ERST	MOD

—	bit31-30	—	**—**
TXFEND	bit29	R/W	**Transmit end flag bit** 0: Not completed 1: Completed
RNACK	bit28	R/W	**Receive NACK bit** 0: Not received 1: Received
—	Bit27-25	—	**—**

BGTE	bit24	R/W	**Data block guard time enable bit (22ETU)** 0: Disabled 1: Enabled
ETUS<7:0>	bit23-16	R/W	**ETU guard time select bit** 0: 2x ETU 1: 3x ETU … 255: 257x ETU
CKS<1:0>	bit15-14	R/W	**ECK output clock source select bit** 00: UBC 01: UBC/2 10: UBC/4 11: UBC/8
ARTS<1:0>	bit13-12	R/W	**Auto retransmission times select bits** 00: Once 01: Twice 10: Thrice 11: Keep re-transmitting until the data transmission is successful
TNAS<1:0>	bit11-10	R/W	**Transmit NACK pulse width select bits** 00:1x ETU 01:1.5x ETU 1x: 2x ETU
ARRE	bit9	R/W	**Auto re-reception enable bit** 0: Disabled 1: Enabled
ARTE	bit8	R/W	**Auto re-transmission enable bit** 0: Disabled 1: Enabled
PS	bit7	R/W	**Parity check select bit** 0: Odd parity check 1: Even parity check
DAS	bit6	R/W	**Data format select bit** 0: Direct convention 1: Inverse convention
IOC	bit5	R/W	**EIO port direction control bit** 0: receive data, transmit ACK 1: transmit data, receive ACK
CHS	bit4	R/W	**EIO communication channel select bit** 0: E0IO0 port 1: E0IO1 port
CK1E	bit3	R/W	**ECK1 port enable bit** 0: Disabled 1: Enabled

CK0E	bit2	R/W	**ECK0 port enable bit** 0: Disabled 1: Enabled
ERST	bit1	W	**EUART module software reset bit** 0: Always read as 0 1: Software reset
MOD	bit0	R/W	**EUART mode select bit** 0: Normal UART mode 1: 7816 mode

Note: UBC is a prescaled baud rate clock, selected by BCS<2:0>.

EUART Baud Rate Register (EUART_BRR)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved					BRR<10:0>										

—	bit31-11	—	—
BRR<10:0>	bit10-0	R/W	**Baud rate configuration bits**

Note: The baud rate calculations for EUART are as follows:

When BCS<2:0>=001, Fpclk/((BRR+1)*16).

When BCS<2:0>=010, Fpclk/((BRR+1)*32).

When BCS<2:0>=011, Fpclk/((BRR+1)*64).

When BCS<2:0>=1xx, Fpclk/((BRR+1)*128).

EUART Interrupt Enable Register (EUART_IE)

Offset address:18_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			RBREIE	Reserved	PEIE	FEIE	ROIE	Reserved							RBIE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		RNAIE	ARTEIE	Reserved			TBWEIE	Reserved						TCIE	TBIE

—	bit31-29	—	—
RBREIE	bit28	R/W	**Receive buffer read error interrupt enable bit**

			0: Disabled 1: Enabled
—	bit27	—	**—**
PEIE	bit26	R/W	**Receive parity error interrupt enable bit** 0: Disabled 1: Enabled
FEIE	bit25	R/W	**Receive framing error interrupt enable bit** 0: Disabled 1: Enabled
ROIE	bit24	R/W	**Receive overrun interrupt enable bit** 0: Disabled 1: Enabled
—	bit23-17	—	**—**
RBIE	bit16	R/W	**Receive buffer full interrupt enable bit** 0: Disabled 1: Enabled
—	bit15-14	—	**—**
RNAIE	bit13	R/W	**Receive NACK interrupt enable bit** 0: Disabled 1: Enabled
ARTEIE	bit12	R/W	**Auto re-transmission error interrupt enable bit** 0: Disabled 1: Enabled
—	bit11-9	—	**—**
TBWEIE	bit8	R/W	**Transmit buffer write error interrupt enable bit** 0: Disabled 1: Enabled
—	bit7-2	—	**—**
TCIE	bit1	R/W	**Transmit complete interrupt enable bit** 0: Disabled 1: Enabled
TBIE	bit0	R/W	**Transmit buffer empty interrupt enable bit** 0: Disabled 1: Enabled

EUART Interrupt Flag Register (EUART_IF)

Offset address:$1C_H$

Reset value:$00000000\_00000000\_00000000\_00000001_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved			RBREIF	Reserved	PEIF	FEIF	ROIF	Reserved							RBIF

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		RNAIF	ARTEIF	Reserved			TBWEIF	Reserved						TCIF	TBIF

—	bit31-29	—	**—**
RBREIF	bit28	R/W	**Receive buffer read error interrupt flag bit** 0: No read error occurred 1: Read error occurred Clear this bit by software writing 1and writing 0 has no effect.
—	bit27	—	**—**
PEIF	bit26	R/W	**Receive parity error interrupt flag bit** 0: No parity error occurred 1: Parity error occurred Clear this bit by software writing 1and writing 0 has no effect.
FEIF	bit25	R/W	**Receive framing error interrupt flag bit** 0: No framing error occurred 1: Framing error occurred Clear this bit by software writing 1and writing 0 has no effect.
ROIF	bit24	R/W	**Receive overrun interrupt flag bit** 0: No overrun occurred 1: Overrun occurred Clear this bit by software writing 1and writing 0 has no effect.
—	bit23-17	—	**—**
RBIF	bit16	R	**Receive buffer full interrupt flag bit** 0: Not full 1: Full (depends on the interrupt mode selected by RBIM) Clear this bit by reading the EUART_RBR register
—	bit15-14	—	**—**
RNAIF	bit13	R/W	**Receive NACK interrupt flag bit** 0: NACK not received 1: NACK received Clear this bit by software writing 1and writing 0

			has no effect.
ARTEIF	bit12	R/W	**Auto re-transmission error interrupt flag** 0: No re-transmission error occurred 1: Re-transmission error occurred Clear this bit by software writing 1and writing 0 has no effect.
—	bit11-9	—	—
TBWEIF	bit8	R/W	**Transmit buffer write error interrupt flag** 0: No write error occurred 1: Write error occurred Clear this bit by software writing 1and writing 0 has no effect.
—	bit7-2	—	—
TCIF	bit1	R/W	**Transmit complete interrupt flag bit** 0: Not completed 1: Completed Clear this bit by software writing 1and writing 0 has no effect.
TBIF	bit0	R	**Transmit buffer empty interrupt flag bit** 0: Not empty 1: Empty (depends on the interrupt mode selected by TBIM) Clear this bit by writing the EUART_TBW register

Notes

1. When EUART interrupts are disabled, the corresponding interrupt flag will be set if an interrupt is triggered, however, no interrupt request will be generated.
2. For all interrupt flag bits in the EUART_IF register, writing 1 will clear the corresponding flag, and writing 0 has no effect. When reading, 1 indicates that the corresponding interrupt has occurred.

EUART Transmit Data Write Register (EUART_TBW)

Offset address:20_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBW<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW<15:0>															

TBW<31:0>	bit31-0	W	**Write transmit buffer** Write in byte: write to TBW<7:0>

			Write in half-word: write to TBW<15:0> Write in word: write to TBW<31:0>

EUART Receive Data Read Register (EUART_RBR)

Offset address:24_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RBR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBR<15:0>															

RBR<31:0>	bit31-0	R	**Read receive buffer** Read in byte: read from RBR<7:0> Read in half-word: read from RBR<15:0> Read in word: read from RBR<31:0>

EUAR Transmit Buffer Register 0/1 (EUART_TB01)

Offset address:40_H

Reset value:00100000_00000000_00100000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		TBEF1	TP1	Reserved			TB1<8:0>								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBEF0	TP0	Reserved			TB0<8:0>								

—	bit31-30	—	—
TBEF1	bit29	R	**Transmit buffer 1 empty flag bit** 0: Full 1: Empty
TP1	bit28	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 1.
—	bit27-25	—	—
TB1<8:0>	bit24-16	R	**Transmit buffer 1 data**
—	bit15-14	—	—
TBEF0	bit13	R	**Transmit buffer 0 empty flag bit** 0: Full 1: Empty
TP0	bit12	R	**Transmit parity bit**

			The parity bit corresponds to the data in transmit buffer 0.
—	bit11-9	—	**—**
TB0<8:0>	bit8-0	R	**Transmit buffer 0 data**

EUART Transmit Buffer Register 2/3 (EUART_TB23)

Offset address:44_H

Reset value:$00100000\_00000000\_00100000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved		TBEF3	TP3	Reserved			TB3<8:0>								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		TBEF2	TP2	Reserved			TB2<8:0>								

—	bit31-30	—	**—**
TBEF3	bit29	R	**Transmit buffer 3 empty flag bit** 0: Full 1: Empty
TP3	bit28	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 3.
—	bit27-25	—	**—**
TB3<8:0>	bit24-16	R	**Transmit buffer 3 data**
—	bit15-14	—	**—**
TBEF2	bit13	R	**Transmit buffer 2 empty flag bit** 0: Full 1: Empty
TP2	bit12	R	**Transmit parity bit** The parity bit corresponds to the data in transmit buffer 2.
—	bit11-9	—	**—**
TB2<8:0>	bit8-0	R	**Transmit buffer 2 data**

EUART Receive Buffer Register 0/1 (EUART_RB01)

Offset address:48_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PE1	FE1	RBFF1	RP1	Reserved			RB1<8:0>								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE0	FE0	RBFF0	RP0	Reserved			RB0<8:0>								

PE1	bit31	R	**Receive buffer 1 parity error flag bit** 0: No error occurred 1: Error occurred
FE1	bit30	R	**Receive buffer 1 framing error flag bit** 0: No error occurred 1: Error occurred
RBFF1	bit29	R	**Receive buffer 1 full flag bit** 0: Empty 1: Full
RP1	bit28	R	**Received parity bit** The corresponding parity bit received in receive buffer 1
—	bit27-25	—	**—**
RB1<8:0>	bit24-16	R	**Receive buffer 1 data**
PE0	bit15	R	**Receive buffer 0 parity error flag bit** 0: No error occurred 1: Error occurred
FE0	bit14	R	**Receive buffer 0 framing error flag bit** 0: No error occurred 1: Error occurred
RBFF0	bit13	R	**Receive buffer 0 full flag bit** 0: Empty 1: Full
RP0	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 0
—	bit11-9	—	**—**
RB0<8:0>	bit8-0	R	**Receive buffer 0 data**

EUART Receive Buffer Register 2/3 (EUART_RB23)

Offset address:4C $_H$

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
PE3	FE3	RBFF3	RP3	Reserved			RB3<8:0>								

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PE2	FE2	RBFF2	RP2	Reserved			RB2<8:0>								

PE3	bit31	R	**Receive buffer 3 parity error flag bit** 0: No error occurred 1: Error occurred
FE3	bit30	R	**Receive buffer 3 framing error flag bit** 0: No error occurred 1: Error occurred
RBFF3	bit29	R	**Receive buffer 3 full flag bit** 0: Empty 1: Full
RP3	bit28	R	**Received parity bit** The corresponding parity bit received in receive buffer 3
—	bit27-25	—	**—**
RB3<8:0>	bit24-16	R	**Receive buffer 3 data**
PE2	bit15	R	**Receive buffer 2 parity error flag bit** 0: No error occurred 1: Error occurred
FE2	bit14	R	**Receive buffer 2 framing error flag bit** 0: No error occurred 1: Error occurred
RBFF2	bit13	R	**Receive buffer 2 full flag bit** 0: Empty 1: Full
RP2	bit12	R	**Received parity bit** The corresponding parity bit received in receive buffer 2
—	bit11-9	—	**—**
RB2<8:0>	bit8-0	R	**Receive buffer 2 data**

5. 3. 10 EUART Application Notes

The device offers one enhanced universal asynchronous receiver transmitter EUART0.

5. 4 Serial Peripheral Interface SPI (SPI0 /SPI1)

The device provides two identical SPIs, and this section uses SPI0 as an example.

5. 4. 1 Overview

- ◇ Master and slave mode
- ◇ 4 data format options
- ◇ Configurable clock rate in master mode
- ◇ Multiple frame width options: 1bit through 8 bits
- ◇ 4-level transmit buffers and 4-level receive buffers
- ◇ Transmit buffer empty interrupt flag and receive buffer full interrupt
- ◇ Receive overrun interrupt, transmit data write error interrupt, and transmit data error interrupt (slave mode only)
- ◇ Chip-select interrupt in slave mode, idle interrupt in master mode
- ◇ Receive delay in master mode
- ◇ Transmit interval in master mode

5. 4. 2 Block Diagram

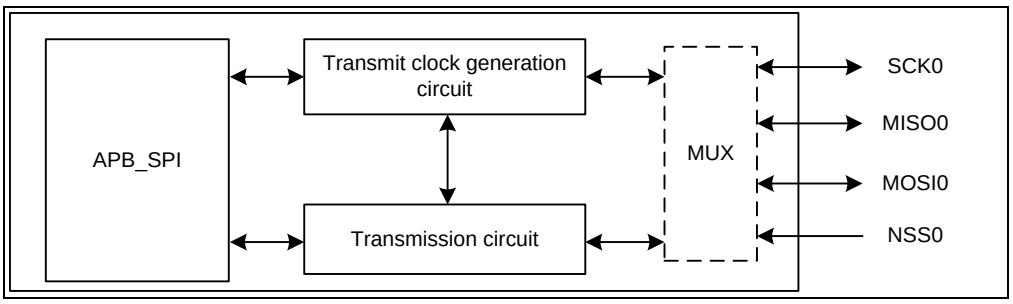


Figure 5-27 SPI0 Block Diagram

5. 4. 3 SPI Communication Mode

The SPI module supports both master mode and slave mode, selected by the MS bit of the SPI_CON register.

SCK0: the serial clock pin

NSS0: the chip select pin to select a slave device (in slave mode)

MOSI0: Master output/ slave input pin

MISO0: Master input/ slave output pin

In master mode, any I/O pin can be used as a chip-select pin.

SPI Port	Master Mode	Slave Mode
SCK0	Support	Support
MOSI0	Support	Support
MISO0	Support	Support
NSS0	—	Support

5. 4. 4 SPI Data Format

The DFS bits of the SPI_CON register selects the data format. The data is transmitted and received with the MSB first. If the SPI transmits the data first and receives data later, the output pin MOSI0 (or MISO0) will send the MSB of the data on the first edge of SCK0. On the contrary, the output pin MOSI0 (or MISO0) will send the MSB before the first edge of SCK0.

Figures below show the data clock timing diagrams in slave mode

DFS<1:0> = 00: transmit on a rising edge (first), receive on a falling edge (later)

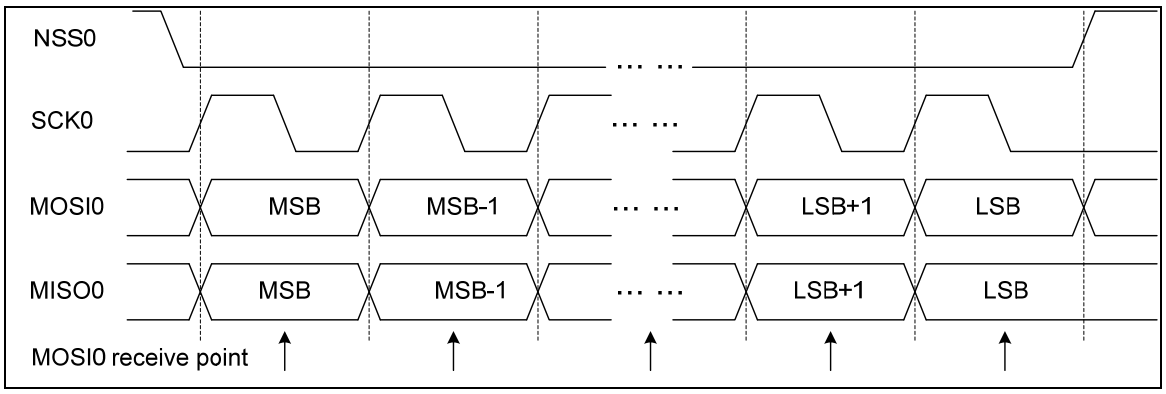


Figure 5-28 Transmit on a rising edge and receive on a falling edge

DFS<1:0> = 01, transmit on a falling edge (first), receive on a rising edge (later)

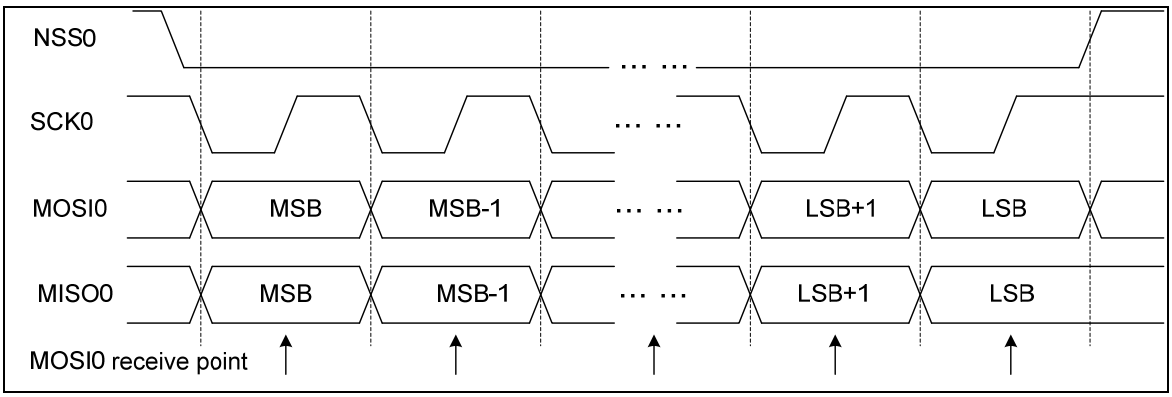


Figure 5-29 Transmit on a falling edge and receive on a rising edge

DFS<1:0> = 10, receive on a rising edge (first), transmit on a falling edge (later)

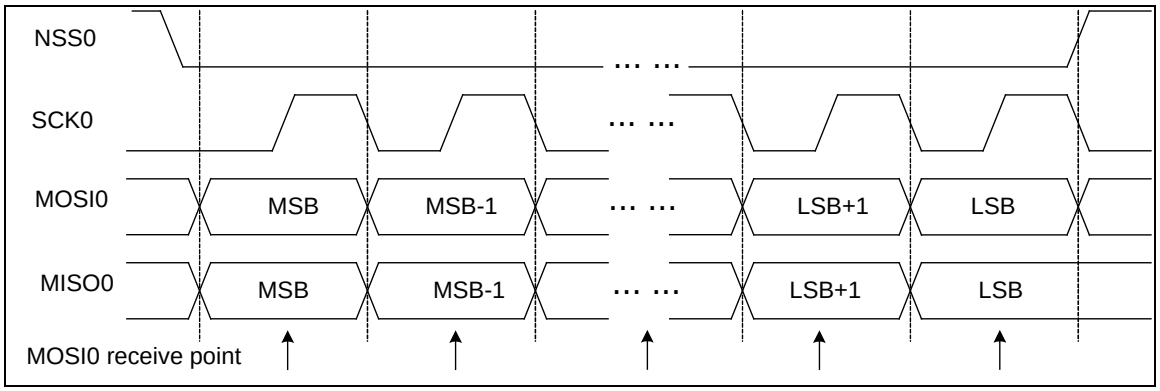


Figure 5-30 Receive on a rising edge and transmit on a falling edge

DFS<1:0> = 11, receive on a falling edge (first), transmit on a rising edge (later)

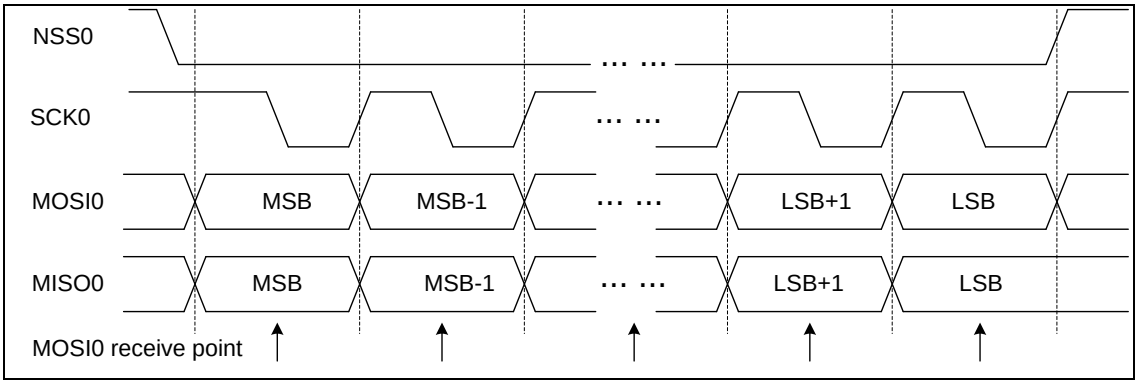


Figure 5-31 Receive on a falling edge and transmit on a rising edge

5. 4. 5 SPI Data Width

The DW<2:0> bits of the SPI_CON register configure the data width with options from 1 bit through 8 bits.

The SPI module supports synchronous transmitter/ receiver with capacity of 4 bytes. The transmit/receive buffer employs byte-alignment, and each buffer stores one frame of data. The transmitter or receiver can store 4+1 frames of data. The data in the buffer is right-aligned. The +1 frame is stored in shift register.

5. 4. 6 SPI Synchronous Transmitter

The synchronous transmitter offers 4 transmit buffers (TB0, TB1, TB2 and TB3) and 1 transmit shift register, which is able to accomplish the continuous data transmission, and up to 5 frames can be written and transmitted. The transmit buffers TB0~TB3 are read-only, and can be written via the transmit buffer write register SPI_TBW.

The SPI_TBW register is a virtual address location and has no physical address. When written, the data is actually written into the transmit buffer. After that, the data is shifted to the transmit shift register and then transmitted via the MOSI0 pin (or MISO0).

The data can be written in byte, half-word and word.

When the SPI_TBW is written in byte, the data is stored in TB0. When written in half-word, the data is stored in TB0 and TB1 with low byte in TB1. When written in word, the data is stored in TB0, TB1, TB2 and TB3 with the lowest byte in TB3.

Data transmission in master mode is shown below.

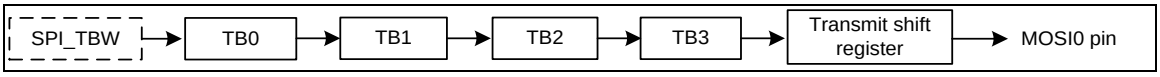


Figure 5-32 SPI Data Transmission Diagram

The transmit buffer interrupt can be configured for different modes by the TBIM bits of the SPI_IE register.

Write TBIM<1:0> =00 to select the byte empty interrupt. The interrupt flag TBIF of the SPI_IF register will be set if the TB0 is empty.

Write TBIM<1:0> =01 to select the half-word empty interrupt. The interrupt flag TBIF of the SPI_IF register will be set if TB0 and TB1 are both empty.

Write TBIM<1:0> =10 to select the word empty interrupt. The interrupt flag TBIF of the SPI_IF register will be set if TB0, TB1, TB2 and TB3 are all empty.

The transmit buffer write error interrupt is supported. The write error interrupt flag TBWEIF will be set if a write access error occurs.

5. 4. 7 SPI Synchronous Receiver

The synchronous receiver offers 4 receive buffers (RB0, RB1, RB2 and RB3) and 1 receive shift register, which is able to realize the continuous data reception, and up to 5 frames can be received in serial.

Reading the receive buffer register SPI_RBR can obtain the received data, and can also clear the corresponding receive buffer full flag RBFF0~RBFF3. Alternatively, reading the receive buffer RB0~RB3 can also obtain the received data, but will not clear the corresponding flag RBFF0~RBFF3.

The SPI_RBR register is a virtual address location and has no physical address. When read, it is actually reading the receive buffer RB0~RB3.

The data can be read in byte, half-word and word.

When the SPI_RBR is read in byte, it reads the data received in RB0. When read in half-word, it reads the data received in RB0 and RB1 with low byte in RB0. When read in word, it reads the data received in RB0, RB1, RB2 and RB3 with the lowest byte in RB0.

Data reception in master mode is shown below.

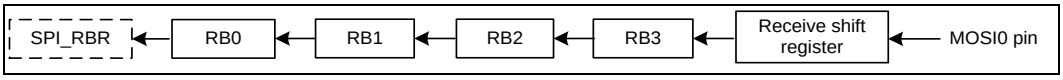


Figure 5-33 SPI Data Reception Diagram

The synchronous receiver operates in the following sequence

When receive buffers are all empty, the receive shift register shifts the data to RB0 one by one.

When RB1, RB2 and RB3 are empty, the receive shift register shifts the data to RB1.

When RB2 and RB3 are empty, the receive shift register shifts the data to RB2.

When RB3 is empty, the receive shift register shifts the data to RB3.

When the data of a receive buffer has been shifted to the next buffer, the corresponding flag RBFF0~RBFF3 will be cleared.

When the four receive buffers and 1 receive shift register are full, if more data is arriving, the receive overrun interrupt flag will be set, no new data will be accepted, and the data in the buffers will be maintained.

The receive buffer interrupt can be configured for different modes by the RBIM bits of the SPI_IE register.

Write RBIM<1:0> =00 to select the byte full interrupt. The interrupt flag RBIF of the SPI_IF register will be set if the RB0 is full.

Write RBIM<1:0> =01 to select the half-word full interrupt. The interrupt flag RBIF of the SPI_IF register will be set if RB0 and RB1 are both full.

Write RBIM<1:0> =10 to select the word full interrupt. The interrupt flag RBIF of the SPI_IF register will be set if RB0, RB1, RB2 and RB3 are full.

5. 4. 8 SPI Communication Control

The SPI module can be configured in master mode and slave mode. When configured in master mode, configure the SPI_CKS to set the clock rate, and determine whether to enable the receive delay and transmit interval. When configured in slave mode, the clock rate is issued by the master. Configure the EN and REN of the SPI_CON register to enable the data transmission and reception. Write the data to be transmitted to the SPI_TBW to start transmission, and read the SPI_RBR to obtain the received data.

In master mode, the master device will enter idle state after all the data of the transmit buffers and transmit shift register have been sent out. The IDLE flag of the SPI_STA register will be set, as well as the idle interrupt flag IDIF in the SPI_IF register.

In slave mode, if the slave device receives a serial clock from a master, and the transmit buffers and transmit shift register are all empty, the transmit error interrupt flag TEIF will be set.

In slave mode, the chip-select interrupt is enabled by the NSSIE bit of the SPI_IE register.

The SPI module can be reset by software by configuring the RST bit of the SPI_CON register. Following a software reset, the data transfer is disabled EN =0; the associated interrupts are disabled TBIE=0, TBWEIE=0, RBIE=0, TEIE=0, ROIE=0, IDIE=0 and NSSIE=0; the corresponding interrupt flags are reset to their default values TBIF=1,

TBWEIF=0, RBIF=0, TEIF=0, ROIF=0, IDIF=0 and NSSIF=0; the idle flag is set IDLE=1; all transmit buffer empty flags are set TBEF0~TBEF3 =1; and all receive buffer full flags are cleared RBFF0~RBFF3=0.

5. 4. 9 SPI Receive Delay

During data transfer, the data transmission and reception are synchronized by using the rising/falling edge. In normal communication, the data transmitted by a slave is supposed to arrive at the master receive port within half a clock period, otherwise, it would cause data lost.

In master mode, it is design with the receive delay capability, which can be enabled by the DRE bit of the SPI_CON register. Once enabled, a master can delay another half a clock cycle to receive the data on the next clock edge, which results to a maximum delay close to 1 clock cycle from a slave transmit port to a master receive port

Example: Write DFS<1:0> =00 to transmit on a rising edge (first) and receive on a falling edge (later).

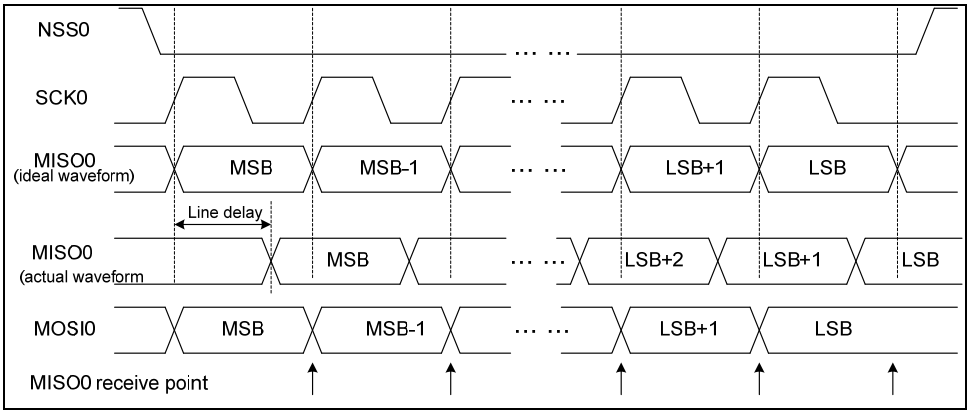


Figure 5-34 SPI Receive Delay Waveforms

5. 4. 10 SPI Transmit Interval between Data Frames

Configure the TME bit of the SPI_CON register to enable the transmit interval. The number of interval cycles is selected by the TMP bits. When the transmit interval is enabled, the transmitter will wait for a preset interval after the completion of each data transmission then proceed with the next data frame.

5. 4. 11 Special Function Registers

SPI Control Register (SPI_CON)

Offset address:00_H

Reset value:$00000111\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RXCLR	TXCLR	Reserved			DW<2:0>			TMP<5:0>						TMS	TME

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								DFS<1:0>		DRE	Reserved	REN	MS	RST	EN

RXCLR	bit31	W	**SPI receive buffer clear bit** 0: No effect 1: Clear receive buffer
TXCLR	bit30	W	**SPI transmit buffer clear bit** 0: No effect 1: Clear transmit buffer
—	bit29-27	—	**—**
DW<2:0>	bit26-24	R/W	**SPI data width (1~8 bits)** One frame consists DW<2:0> +1 bit
TMP<5:0>	bit23-18	R/W	**SPI transmit interval between data frames (master mode only)** See details in Note 1
TMS	bit17	R	**SPI transmit interval status bit (master mode only)** 0: Non transmit interval state 1: During transmit interval state
TME	bit16	R/W	**SPI transmit interval enable bit (master mode only)** 0: Disabled 1: Enabled
—	bit15-8	—	**—**
DFS<1:0>	bit7-6	R/W	**SPI data format select bits** 00: Transmit on rising edge (first), receive on falling edge (later) 01: Transmit on falling edge (first), receive on rising edge (later) 10: Receive on rising edge (first), transmit on falling edge (later) 11: Receive on falling edge (first), transmit on rising edge (later)
DRE	bit5	R/W	**SPI receive delay enable bit (master mode only)** 0: Disabled

			1: Enabled
—	bit4	—	—
REN	bit3	R/W	**SPI receive enable bit** 0: Disabled 1: Enabled (EN needs to be enabled at the same time)
MS	bit2	R/W	**SPI mode select bit** 0: Master mode 1: Slave mode
RST	bit1	W	**SPI software reset bit** 0: Always read as 0 1: Software reset. Automatically cleared by software.
EN	bit0	R/W	**SPI transmit enable bit** 0: Disabled 1: Enabled, data transmit function is enabled only.

Notes

1. The transmit interval calculation is expressed as follows.
   TSCK * (1 + TMP), which suggests the interval can be from 1~ 64x TSCK
2. Due to the fact that different data formats require different initial voltage levels (see the waveforms in the corresponding section), if the initial values of the SPI ports are undetermined, select the data format first and the port initial values will be automatically configured. After that, enable the transmitter and receiver by setting EN and REN, which in this case requires two separate write operations to the SPI_CON register; otherwise it would cause a communication error.

SPI Transmit Data Write Register (SPI_TBW)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBW<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW<15:0>															

TBW<31:0>	bit31-0	W	**Write transmit buffer** Write in byte: write to TBW<7:0> Write in half-word: write to TBW<15:0> Write in word: write to TBW<31:0>

SPI Receive Data Read Register (SPI_RBR)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RBR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBR<15:0>															

RBR<31:0>	bit31-0	R	**Read receive data** Read in byte: read from RBR<7:0> Read in half-word: read from RBR<15:0> Read in word: read from RBR<31:0>

SPI Interrupt Enable Register (SPI_IE)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				RBIM<1:0>		TBIM<1:0>		Reserved	TBWEIE	NSSIE	IDIE	ROIE	TEIE	RBIE	TBIE

—	bit31-12	—	**—**
RBIM<1:0>	bit11-10	R/W	**SPI receive buffer full interrupt mode select bits** 00: An interrupt occurs if RB0 is full 01: An interrupt occurs if RB0 and RB1 are both full 10: An interrupt occurs if RB0, RB1, RB2 and RB3 are full 11: Reserved
TBIM<1:0>	bit9-8	R/W	**SPI transmit buffer empty interrupt mode select bits** 00: An interrupt occurs if TB0 is empty 01: An interrupt occurs if TB0 and TB1 are both empty 10: An interrupt occurs if TB0, TB1, TB2 and TB3 are empty 11: Reserved
—	bit7	—	**—**
TBWEIE	bit6	R/W	**SPI transmit data write error interrupt enable**

			bit 0: Disabled 1: Enabled
NSSIE	bit5	R/W	**SPI chip-select change interrupt enable bit (slave mode only)** 0: Disabled 1: Enabled
IDIE	bit4	R/W	**SPI idle interrupt enable bit (master mode only)** 0: Disabled 1: Enabled
ROIE	bit3	R/W	**SPI receive overrun interrupt enable bit** 0: Disabled 1: Enabled
TEIE	bit2	R/W	**SPI transmit error interrupt enable bit (slave mode only)** 0: Disabled 1: Enabled
RBIE	bit1	R/W	**SPI receive buffer full interrupt enable bit** 0: Disabled 1: Enabled
TBIE	bit0	R/W	**SPI transmit buffer empty interrupt enable bit** 0: Disabled 1: Enabled

SPI Interrupt Flag Register (SPI_IF)

Offset address:14_H

Reset value:$00000000\_00000000\_00000000\_00000001_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved									TBWEIF	NSSIF	IDIF	ROIF	TEIF	RBIF	TBIF

—	bit31-7	—	**—**
TBWEIF	bit6	R/W	**SPI transmit buffer write error interrupt flag bit** 0: No write error. 1: Write error occurred. Following access errors may happen. - When written in word, not all buffers are empty.

			- When written in half-word, more than 2 buffers are not empty. - When written in byte, all buffers are full. - Write SPI_TBW<31:16> with a half-word - Write SPI_TBW<31:8> with a word Clear this bit by software writing 1 and writing 0 has no effect.
NSSIF	bit5	R/W	**Chip-select change interrupt flag bit (slave mode only)** 0: Chip-select signal not changed 1: Chip-select signal changed Clear this bit by software writing 1 and writing 0 has no effect.
IDIF	bit4	R/W	**Idle interrupt flag bit (master mode only)** 0: Not entered idle state 1: Entered idle state Clear this bit by software writing 1 and writing 0 has no effect. Alternatively, write the SPI_TBW register to clear the flag.
ROIF	bit3	R/W	**SPI receive overrun interrupt flag bit** 0: No overrun 1:Overrun occurred Clear this bit by software writing 1 and writing 0 has no effect.
TEIF	bit2	R/W	**SPI transmit error interrupt flag bit (slave mode only)** 0: No error 1: Error occurred: a slave receives a clock from a master while the transmit buffers and shift register are all empty. Clear this bit by software writing 1 and writing 0 has no effect.
RBIF	bit1	R	**SPI receive buffer full interrupt flag bit** Read the SPI_RBR register to clear the flag.
TBIF	bit0	R	**SPI transmit buffer empty interrupt flag bit** Write the SPI_TBW register to clear the flag.

Notes

1. When SPI interrupts are disabled, the corresponding interrupt flag will be set if an interrupt is triggered, however, no interrupt request will be generated..
2. For each interrupt flag in the SPI_IF register, they can be cleared by software writing 1 and writing 0 has no effect. When reading, 1 indicates that the corresponding interrupt has occurred.

SPI Transmit Buffer Register (SPI_TB)

Offset address:18_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TB3<7:0>								TB2<7:0>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TB1<7:0>								TB0<7:0>							

TB3<7:0>	bit31-24	R	Transmit buffer 3 data
TB2<7:0>	bit23-16	R	Transmit buffer 2 data
TB1<7:0>	bit15-8	R	Transmit buffer 1 data
TB0<7:0>	bit7-0	R	Transmit buffer 0 data

SPI Receive Buffer Register (SPI_RB)

Offset address:$1C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RB3<7:0>								RB2<7:0>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RB1<7:0>								RB0<7:0>							

RB3<7:0>	bit31-24	R	Receive buffer 3 data
RB2<7:0>	bit23-16	R	Receive buffer 2 data
RB1<7:0>	bit15-8	R	Receive buffer 1 data
RB0<7:0>	bit7-0	R	Receive buffer 0 data

SPI Status Register (SPI_STA)

Offset address:20_H

Reset value:00000000_00000001_00001111_10000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															IDLE

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBFF3	RBFF2	RBFF1	RBFF0	TBEF3	TBEF2	TBEF1	TBEF0	NSS	Reserved						

—	bit31-17	—	—
IDLE	bit16	R	**SPI idle flag bit (master mode only)** 0: Not idle

			1: Idle
RBFF3	bit15	R	**RB3 full flag bit** 0: Empty 1: Full
RBFF2	bit14	R	**RB2 full flag bit** 0: Empty 1: Full
RBFF1	bit13	R	**RB1 full flag bit** 0: Empty 1: Full
RBFF0	bit12	R	**RB0 full flag bit** 0: Empty 1: Full
TBEF3	bit11	R	**TB3 empty flag bit** 0: Full 1: Empty
TBEF2	bit10	R	**TB2 empty flag bit** 0: Full 1: Empty
TBEF1	bit9	R	**TB1 empty flag bit** 0: Full 1: Empty
TBEF0	bit8	R	**TB0 empty flag bit** 0: Full 1: Empty
NSS	bit7	R	**SPI chip-select flag bit (slave mode only)** 0: Selected 1: Not selected
—	bit6-0	—	**—**

SPI Baud Rate Register (SPI_CKS)

Offset address:24_H

Reset value:00000000_00000000_00000000_00001000B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CKS<7:0>							

—	bit31-8	—	—
CKS<7:0>	bit7-0	R/W	**SPI baud rate select bits (master mode only)** SPI baud rate calculations When CKS<7:0> = 0x00: F_{PCLK}

			When CKS<7:0> = 0x01~0xFF: F_{PCLK}/(CKS * 2)

5. 4. 12 SPI Application Notes

Two synchronous serial interfaces are available, SPI0 and SPI1.

To ensure proper communications, the following configurations must be observed.

1) When a filter is used on the SPI ports, the SPI clock frequency must be less than 20Mz. To ensure of the stability of the communication, enabling the port filter is recommended.

2) The following relations exist between SPI data width and the CKS configuration.

   When the frame width is 5~8 bits, CKS >= 0;

   When the frame width is 2~4 bits, CKS >= 1;

   When the frame width is 1 bit, CKS > 2.

3) Due to that different data formats require different initial voltage levels (see the waveforms in the corresponding section), if the initial values of the SPI ports are undetermined, select the data format first and the port initial values will be automatically configured. After that, enable the transmitter and receiver by setting EN and REN, which in this case requires two separate write operations to the SPI_CON register.

5. 5 Inter-integrated Circuit Interface (I2C0)

5. 5. 1 Overview

◇ Single master mode

- Auto repeated call function
- Auto Stop bit transmission
- ACK delay
- Transmit interval between data frames
- Software triggered Start bit
- Software triggered Stop bit
- Software triggered data reception with configurable receive mode

◇ Slave mode

- 7-bit slave address
- Slave address match interrupt flag
- Receive stop bit interrupt flag
- Auto clock stretching
- Auto NACK transmission

◇ 4-level transmit buffers and 4-level receive buffers

◇ Configurable push-pull output or open drain output on both SCL0 and SDA0 pins

◇ Configurable samplers (speed:16X) available on both SCL0 and SDA0 pins

◇ Transmit buffer empty interrupt and receive buffer full interrupt

◇ Start bit interrupt and stop bit interrupt

◇ Receive overrun interrupt and transmit data write error interrupt

5. 5. 2 Block Diagram

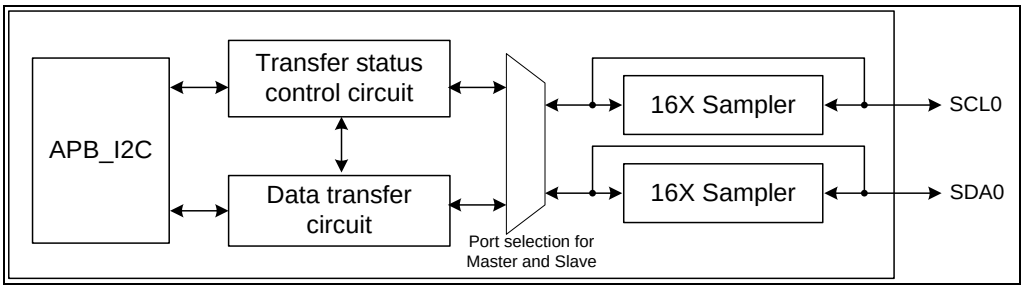


Figure 5-35 I2C Block Diagram

5. 5. 3 I2C Bus Basic Principle

5. 5. 3. 1 I2C Communication Protocol

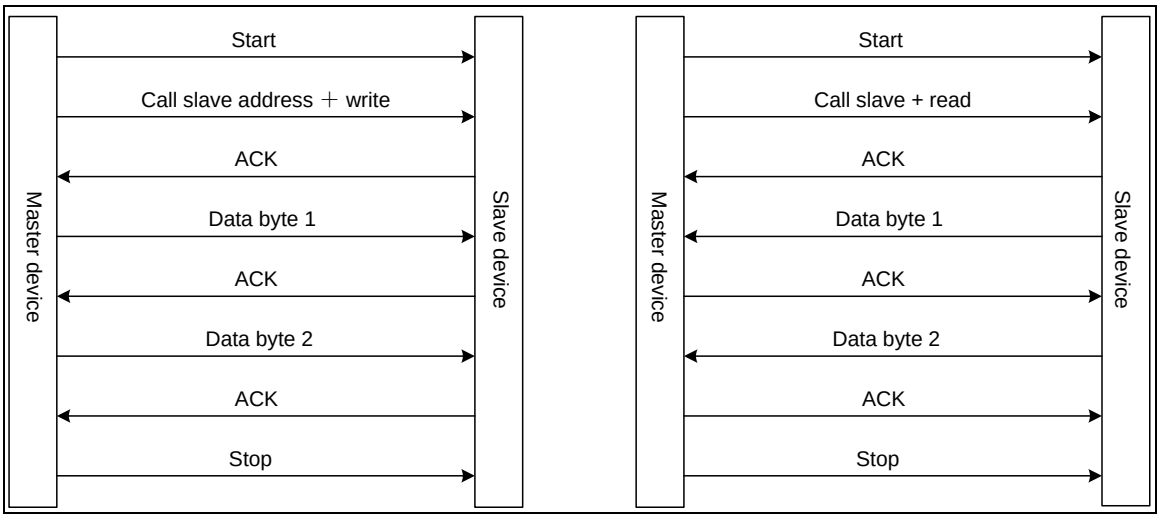


Figure 5-36 I2C Bus Communication Protocol Diagram

The following protocol must be strictly observed in I2C communication.

A master device initiates a communication by sending a start bit to control the bus and a stop bit to release the bus.

The bus has multi-master capability (prior to that, each master must support multi-master arbitration mechanism) and at least 1 slave. Each slave must have an independent and unique address.

The master sends a start bit, followed by a slave address and a read/write bit.

The read/write bit R/$\overline{W}$ (also called direction bit) is to notify the slave the direction of the data transfer. 0 indicates the master is writing data to the slave while 1 indicates the master is reading data from the slave.

I2C communication protocol supports the acknowledge mechanism. A receiver must respond with an ACK signal or NACK signal after a data byte (including the slave address) is transmitted by a transmitter, thereby, the transmitter will act accordingly.

If the SCL of both master and slave device are open drain and the master supports SCL wait request, the slave can stretch the SCL when it is at low level, allowing the master to wait for the slave until the slave releases the SCL.

The MSB is always sent first.

During I2C communication, the level of the SDA changes only when the clock line SCL is held low, and remains unchanged when the SCL is held high. If the SDA changes while the SCL is high, a start bit or a stop bit will be triggered, where a high to low transition triggers a start bit and a low to high transition triggers a stop bit.

5. 5. 3. 2 I2C Data Transfer

The data transfer format is determined by the specific design specification of a slave device. The following format introduced is the most commonly used.

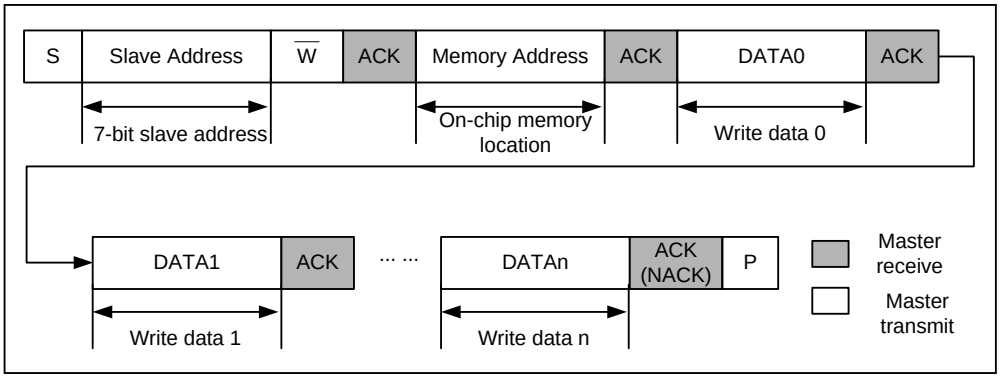


Figure 5-37 Master Writing to Slave

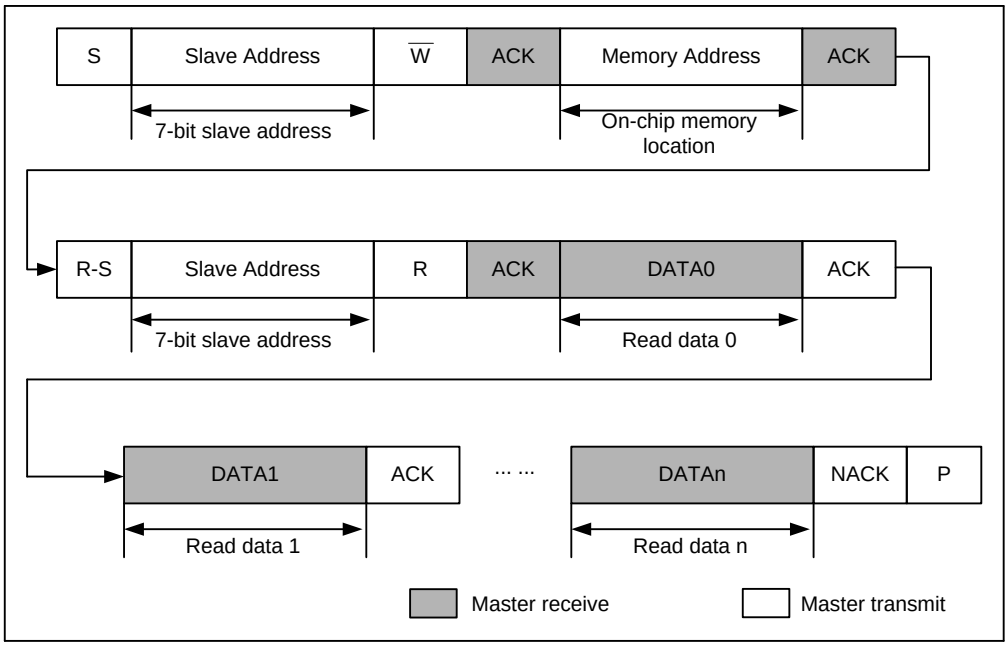


Figure 5-38 Master Reading from Slave

5. 5. 4 I2C Port Configuration

The SCL0 and SDA0 pins are both designed with push-pull output and open-drain output, controlled by SCLOD and SDAOD of the I2C_CON register.

The push-pull output is a standard output of I/O ports. When the output data is 0 and 1, the level on the port is also 0 and 1.

For push-pull output mode, a port level collision might exist. For example, master device outputs 0 and slave device outputs 1; this is where a collision takes place which makes the port status undetermined.

Open-drain output is the standard mode of I2C bus protocol, which can avoid the

collision issue. The figure below shows the open-drain output design.

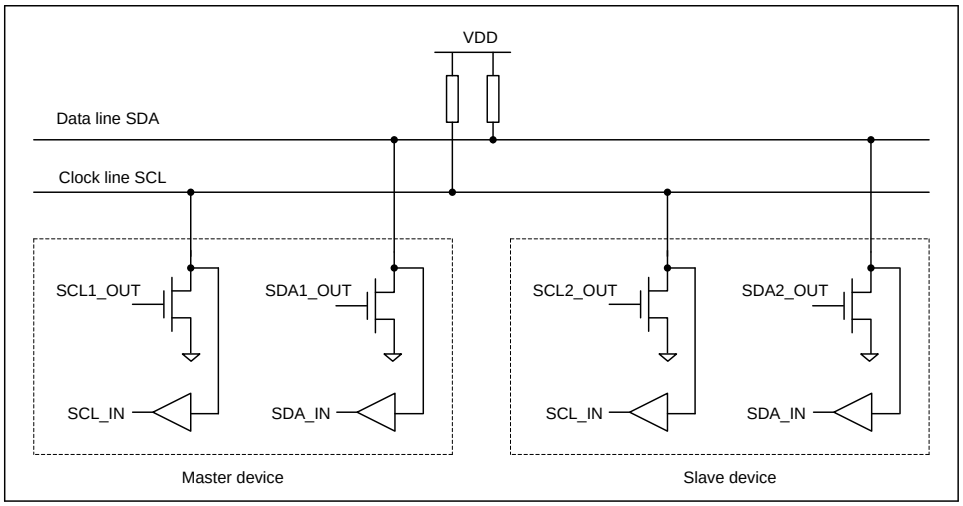


Figure 5-39 Open-drain Output Diagram

The high level of open-drain output is sourced from the pull-up resistors on the I2C bus while the low level is determined by a master together with a slave. Either a master or slave can pull down the bus level to 0. However, only when both the master and slave release the bus can the bus level be pulled up to 1.

5. 5. 5 I2C Time-based Timer and 16x Sampler

Both SCL0 and SDA0 pins support a sampler (speed: 16X). Configure the SCLSE bit of the I2C_CON register can enable the samplers on both pins. The counter clock period of the I2C time-based timer is the sampling period of the 16X sampler.

In master mode, the I2C time-based timer also generates the baud rate for data transfer.

If a 16X sampler is used or a master mode is configured, the time-based timer must be enabled by the TJE bit of the I2C_CON register. The TJP bits select a desired period of the time-based timer.

In master mode, the timing of different parameters is tabulated as below.

Parameter	Symbol	With 16X sampler enabled	With 16X sampler disabled
Start/restart bit set up time	TSU:S	> Tosc x (TJP+1) x 12	Tosc x (TJP+1) x 8
Start/restart bit hold time	THD:S	> Tosc x (TJP+1) x 12	Tosc x (TJP+1) x 8
Stop bit set up time	TSU:P	> Tosc x (TJP+1) x 12	Tosc x (TJP+1) x 8
Stop bit hold time	THD:P	> Tosc x (TJP+1) x 12	Tosc x (TJP+1) x 8
Data/ACK bit set up time	TSU:DA	> Tosc x (TJP+1) x 4	Tosc x (TJP+1) x 4
Data/ACK bit hold time	THD:DA	> Tosc x (TJP+1) x 8	Tosc x (TJP+1) x 4
Clock high time	THIGH	Tosc x (TJP+1) x 12	Tosc x (TJP+1) x 8
Clock low time	TLOW	Tosc x (TJP+1) x 12	Tosc x (TJP+1) x 8

Table 5-1 Timing of Different Parameters

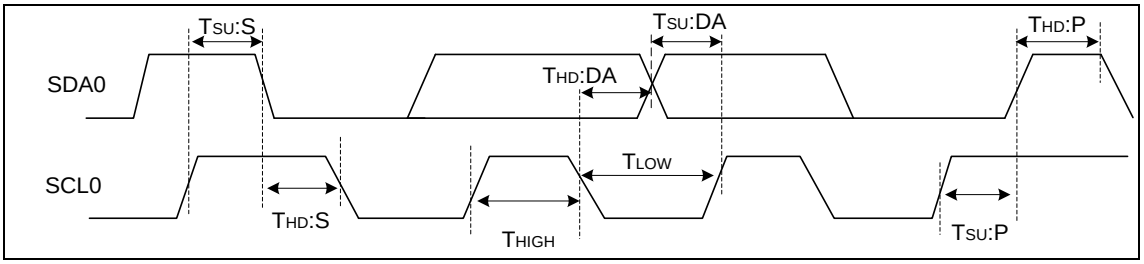


Figure 5-40 Waveforms of SDA0 and SCL0 Signal

After 16X samplers are enabled, they start to sample the signal on the I2C bus. The signal is unstable at first because of the pull-up resistor. Only when the signal becomes stable will the output of the sampler be stable. Consequently, the baud rate could decrease within that duration and the decrement depends on the transition time from low to high.

In master mode, the baud rate calculation is as follows (Fosc is the system clock)

With 16X samplers enabled: F_{SCL}=Fosc / ((TJP+1) x 24)

With 16X samplers disabled: F_{SCL}=Fosc / ((TJP+1) x 16)

5. 5. 6 I2C Transmitter

The I2C transmitter offers 4 transmit buffers (TB0, TB1, TB2 and TB3) and 1 transmit shift register, which is able to realize the continuous data transmission, and up to 5 frames can be written and transmitted. The transmit buffers TB0~TB3 are read-only, and can be written via the transmit buffer write register I2C_TBW.

The I2C_TBW register is a virtual address location and has no physical address. When written, the data is actually written into a transmit buffer. After that, the data is shifted to the transmit shift register and then transmitted via the SDA0 pin.

The data can be written in byte, half-word and word.

When the I2C_TBW is written in byte, the data is stored in TB0. When written in half-word, the data is stored in TB0 and TB1 with low byte in TB1. When written in word, the data is stored in TB0, TB1, TB2 and TB3 with the lowest byte in TB3.

Data transmission is shown below.

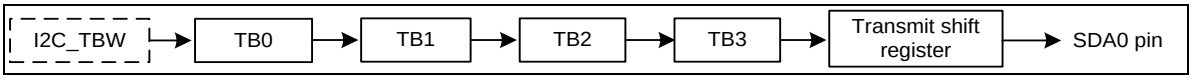


Figure 5-41 I2C Data Transmission Diagram

The transmit buffer interrupt can be configured for different modes by the TBIM bits of the I2C_IE register.

Write TBIM<1:0> = 00 to select the byte empty interrupt. The interrupt flag TBIF of the I2C_IF register will be set if the TB0 is empty.

Write TBIM<1:0> = 01 to select the half-word empty interrupt. The interrupt flag TBIF of the I2C_IF register will be set if TB0 and TB1 are both empty.

Write TBIM<1:0> =10 to select the word empty interrupt. The interrupt flag TBIF of the I2C_IF register will be set if TB0, TB1, TB2 and TB3 are empty.

The transmit buffer write error interrupt is supported. The write error interrupt flag TBWEIF will be set if a write access error occurs.

5. 5. 7 I2C Receiver

The I2C receiver offers 4 receive buffers (RB0, RB1, RB2 and RB3) and 1 receive shift register, which is able to realize the continuous data reception, and up to 5 frames can be received in serial. Reading the receive buffer read register I2C_RBR can obtain the received data, and can also clear the corresponding receive buffer full flag of the I2C_STA register. Alternatively, reading the receive buffer RB0~RB3 can also obtain the received data, but will not clear the corresponding flag RBFF0~RBFF3.

The I2C_RBR register is a virtual address location and has no physical address. When read, it is actually reading the receive buffer RB0~RB3.

The data can be read in byte, half-word and word.

When the I2C_RBR is read in byte, it reads the data received in RB0. When read in half-word, it reads the data received in RB0 and RB1 with low byte in RB0. When read in word, it reads the data received in RB0, RB1, RB2 and RB3 with the lowest byte in RB0.

Data reception is shown below

Figure 5-42 I2C Data Reception Diagram

When the data of a receive buffer has been shifted to the next buffer, the corresponding full flag will be cleared.

When the four receive buffers and 1 receive shift register are full, if more data is arriving, the receive overrun interrupt flag ROIF will be set, no new data will be accepted.

The receive buffer interrupt can be configured for different modes by the RBIM bits of the I2C_IE register.

Write RBIM<1:0> = 00 to select the byte full interrupt. The interrupt flag RBIF of the I2C_IF register will be set if the RB0 is full.

Write RBIM<1:0> = 01 to select the half-word full interrupt. The interrupt flag RBIF of the I2C_IF register will be set if RB0 and RB1 are both full.

Write RBIM<1:0> = 10 to select the word full interrupt. The interrupt flag RBIF of the I2C_IF register will be set if RB0, RB1, RB2 and RB3 are full.

5. 5. 8 I2C Communication Control

The I2C module can be reset by software by configuring the RST bit of the I2C_CON register. Following a software reset, the data transfer is disabled EN =0; the associated interrupts are disabled SRIE=0, SPIE=0, TBIE=0, TBWEIE=0, RBIE=0, TEIE=0, ROIE=0, and NAIE=0; the corresponding interrupt flags are reset to their default values SRIF=0, SPIF=0, TBIF=1, TBWEIF=0, RBIF=0, TEIF=0, ROIF=0 and NAIF=0; the idle flag is set IDLE=1; all transmit buffer empty flags are set TBEF0~TBEF3 =1; and all receive buffer full flags are cleared RBFF0~RBFF3= 0.

5. 5. 8. 1 I2C Start Bit

The SRT bit can trigger a start bit to start or restart a data transfer.

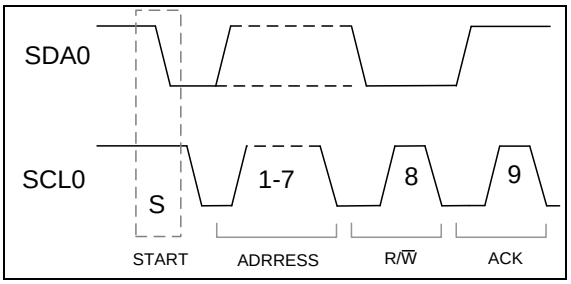


Figure 5-43 I2C Start Bit Waveform

In master mode, an auto-call function is available. Set the SRAE bit of the I2C_MOD register can enable the auto-call function. A master is able to automatically determine an “address ACK” signal. If a NACK is received, a start bit is transmitted automatically to restart the call operation and keep calling until an ACK is received. Please make sure that the slave address do exist, otherwise the master will restart and keep calling.

Example: A write wait time exists when the EEPROM is being written via I2C. If the device is called by a master while the EEPROM is performing programming, a NACK will be received. Two approaches can address this issue. One is to set a call interval by master, calling the device after its EEPROM completes writing. The other is to start the auto-call function, keep calling the device until an ACK is received.

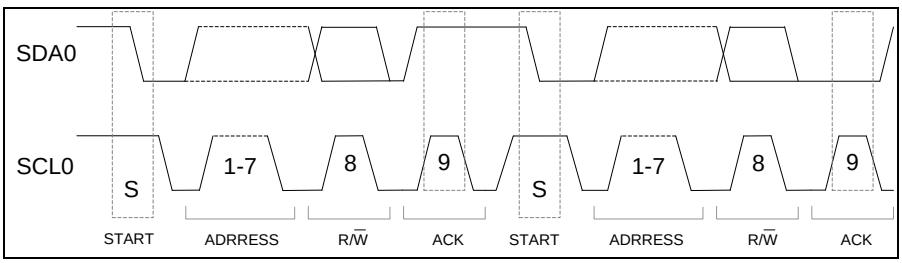


Figure 5-44 I2C Auto-call Waveform

5. 5. 8. 2 I2C Stop Bit

The SPT bit of the I2C_MOD register can trigger a stop bit to terminate the current data transfer.

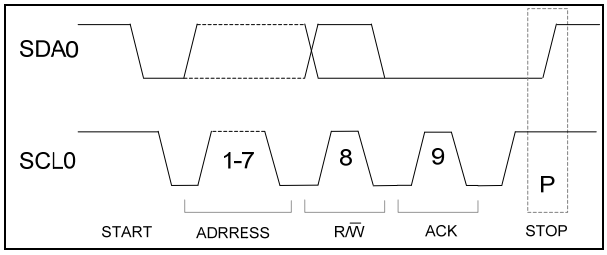


Figure 5-45 I2C Stop Bit Waveform

In master mode, a stop bit can be automatically transmitted. Set the SPAE bit of the I2C_MOD register to enable the stop bit auto-transmission. After transmitting a NACK or receiving a NACK, a stop bit is automatically transmitted to terminate the unsuccessful data transfer. The stop bit auto-transmission has a lower priority than the auto-call function.

5. 5. 8. 3 I2C ACK Delay

In master mode, the ACK delay function is available. Set the ADE bit of the I2C_MOD register to enable the ACK delay function, and the ADLY bits select the delay time. When the function is enabled, a master is able to delay an ACK transmission.

When a slave is not able to receive or transmit an ACK at a normal baud rate, the master can enable the ACK delay function, and select the delay time depending on the specific design specification of the slave.

Example: when ADLY<2:0> = 001 of the I2C_MOD register, the delay time is 1x T_{SCL0}. See the corresponding waveform below.

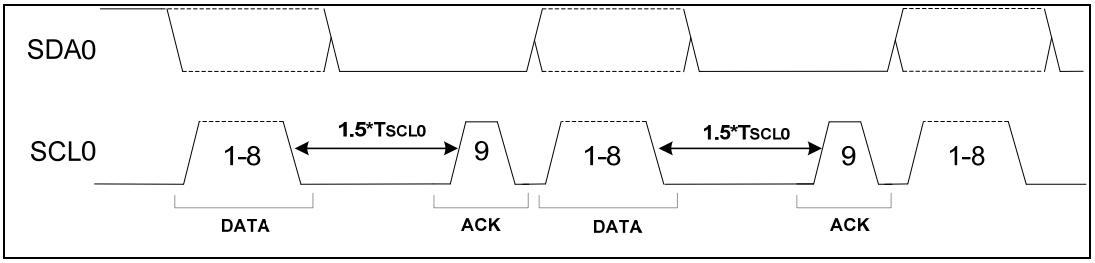


Figure 5-46 I2C ACK Delay Waveform

5. 5. 8. 4 I2C Transmit Interval between Data Frames

In master mode, configure the TIS bits of the I2C_MOD register to enable and select the transmit intervals. Once enabled, the next data frame will be delayed for the set interval after receiving an ACK of the current data frame.

When a slave is not able to read the received data in time, or is not yet ready to transmit the data, a transmit interval can be set depending on the specific design specification.

Example: write TIS<3:0> = 0001 to select a transmit interval of 1x T_{SCL0}.

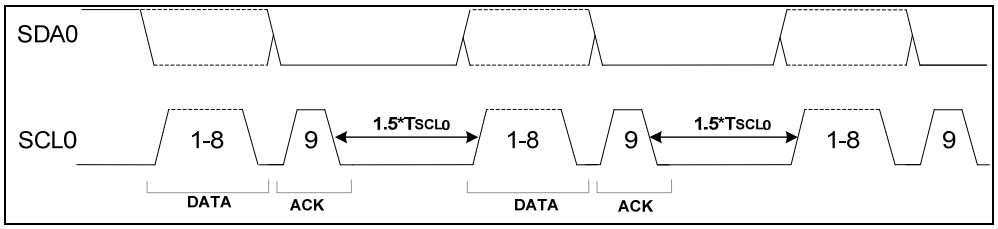


Figure 5-47 I2C Transmit Interval between Data Frames

5. 5. 8. 5 I2C Clock Stretching

In slave mode, the clock stretching capability is available, and it can be enabled by setting the CSE bit of the I2C_MOD register.

In addition, the SCL0 pin must be configured for open-drain output by the SCLOD bit of the I2C_CON register, which allows the slave device to hold the clock line until it is ready to continue communicating with the master device.

Usually, the clock line SCL0 is completely controlled by a master device. However, when a slave device is not ready to continue communicating, it can delay the transfer through clock stretching, by forcing the clock line low when it is at low level (cannot not force it low when it is at high level) and releasing the clock line when the slave device is ready.

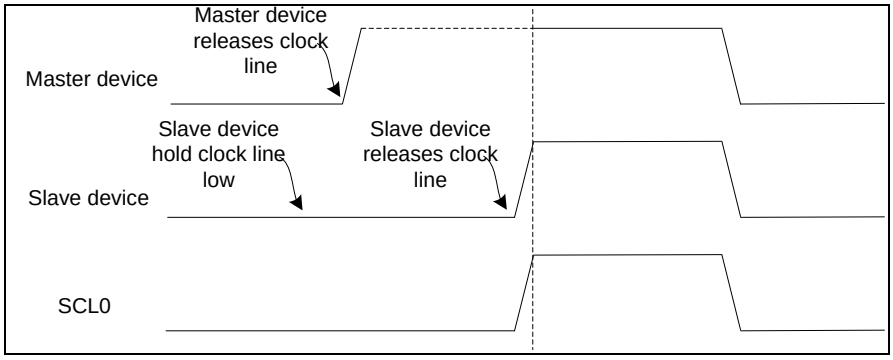


Figure 5-48 I2C Clock Stretching

With the clock stretching enabled, when the slave device receives its call address and a read bit, if transmit buffers and the transmit shift register are empty, the clock line will automatically be stretched low. When the slave device receives its call address and a write bit, if receive buffers and the receive shift register are full, the clock line will automatically be stretched low.

5. 5. 8. 6 I2C Auto NACK Transmission

In slave mode, it is featured with the auto NACK transmission, enabled by the ANAE bit of the I2C_MOD register. Enabling the auto NACK transmission will not force the slave device to control the clock line. Besides, the auto NACK transmission can be used in both push-pull output and open-drain output mode for SCL0 pin.

With the auto NACK transmission enabled, when the slave device receives its call

address and a read bit, if transmit buffers and the transmit shift register are empty, a NACK will automatically be transmitted. When the slave device receives its call address and a write bit, if receive buffers and the receive shift register are full, a NACK will automatically be transmitted, notifying the master device to restart the communication.

5. 5. 9 Special Function Registers

I2C Control Register (I2C_CON)

Offset address:00_H

Reset value:$00000000\_00000000\_11111111\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved								SA<6:0>							RW

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TJP<7:0>								TJE	Reserved	SDASE	SCLSE	SDAOD	SCLOD	RST	EN

—	bit31-24	—	**—**
SA<6:0>	bit23-17	R/W	**Slave address** Master mode: automatically send the slave address after start/restart bit is triggered Slave mode: used to compare and match after receiving start/restart bit
RW	bit16	R/W	**I2C read write bit** 0: Write 1: Read In master mode: this bit is readable and writable. It will be automatically sent after the start/restart bit is triggered. In slave mode, this bit is read-only. After the matched slave address, this bit will be updated automatically by hardware according to the received RW bit.
TJP<7:0>	bit15-8	R/W	**I2C time-based timer period select bit** 00~FF: 1~256 T_{PCLK}
TJE	bit7	R/W	**I2C time-based timer enable bit** 0: Disabled 1: Enabled
—	bit6	—	**—**
SDASE	bit5	R/W	**SDA 16X sampler enable bit** 0: Disabled 1: Enabled
SCLSE	bit4	R/W	**SCL 16X sampler enable bit** 0: Disabled

			1: Enabled
SDAOD	bit3	R/W	**SDA output mode select bit** 0: Push-pull output 1: Open-drain output
SCLOD	bit2	R/W	**SCL output mode select bit** 0: Push-pull output 1: Open-drain output
RST	bit1	W	**I2C software reset bit** 0: Always read as 0 1: Software reset, cleared by hardware automatically
EN	bit0	R/W	**I2C enable bit** 0: Disabled 1: Enabled

I2C Operating Mode Register (I2C_MOD)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved							TAS	Reserved				BLD	RDT	SPT	SRT

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TIS<3:0>				ADE	ADLY<2:0>			SPAE	SRAE	ANAE	CSE	RDM<2:0>			MS

—	bit31-25	—	**—**
TAS	bit24	R/W	**I2C transmit ACK bit (slave mode only)** 0: Transmit ACK 1: Transmit NACK
—	bit23-20	—	**—**
BLD	bit19	R/W	**I2C bus release control bit (for master only)** 0: No effect 1: When SDA port level is high, the SCL transmits 8 clock cycles. The start and stop bit needs to be configured together to release the clock and data line.
RDT	bit18	R/W	**I2C receive data trigger bit (for master only)** 0: No effect 1: Issue the clock and start data reception. Data format is configured by RDM<2:0>
SPT	bit17	R/W	**I2C stop bit trigger bit (for master mode**

			only) 0: No effect 1: Trigger a stop bit
SRT	bit16	R/W	**I2C start bit trigger bit (for master mode only)** 0: No effect 1:Trigger a start bit and set the start bit transmission complete interrupt flag
TIS<3:0>	bit15-12	R/W	**I2C transmit interval select bits (for master mode only)** 0000: Disabled 0001~1111: 1~15x I2C clock cycles
ADE	bit11	R/W	**I2C ACK delay enable bit (for master mode only)** 0: Disabled 1: Enabled
ADLY<2:0>	bit10-8	R/W	**I2C ACK delay time select bit (for master mode only)** 000: 0.5x I2C clock cycle 001: 1x I2C clock cycles 010: 1.5x I2C clock cycles 011: 2 x I2C clock cycles 100: 2.5x I2C clock cycles 101: 3x I2C clock cycles 110: 3.5x I2C clock cycles 111: 4 x I2C clock cycles
SPAE	bit7	R/W	**I2C stop bit auto-transmission enable bit (for master mode only)** 0: Disabled 1: Enabled (automatically send the stop bit after transmitting or receiving a NACK. It has a lower priority than SRAE)
SRAE	bit6	R/W	**I2C auto-call enable bit (for master mode only)** 0: Disabled 1: Enabled (if a NACK is received after sending the call address, the master device will restart calling.)
ANAE	bit5	R/W	**I2C auto NACK transmission enable bit (for slave mode only)** 0: Disabled 1: Enabled
CSE	bit4	R/W	**I2C clock stretching enable bit (for slave mode only)**

			0: Disabled 1: Enabled
RDM<2:0>	bit3-1	R/W	**I2C receive mode select bit (for master only)** 000: Receive 1 byte then send ACK 001: Receive 1 byte then send NACK 010: Continuously receive 2 bytes and send ACK for each byte 011: Continuously receive 2 byte, send ACK for byte 1 and NACK for byte 2 100: Continuously receive 4 bytes, and send ACK for each byte 101: Continuously receive 4 bytes, and send ACK for first 3 bytes, NACK for the byte 4. 110: Continuously receive data bytes, and send ACK for each byte 111: Send NACK after completing the current data byte reception
MS	bit0	R/W	**I2C mode select bit** 0: Master mode 1: Slave mode

I2C Interrupt Enable Register (I2C_IE)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			TIDLEIE	RBIM<1:0>		TBIM<1:0>		TBWEIE	NAIE	ROIE	TEIE	RBIE	TBIE	SPIE	SRIE

—	bit31-13	—	**—**
TIDLEIE	bit12	R/W	**I2C transmit idle interrupt enable bit** 0: Disabled 1: Enabled
RBIM<1:0>	bit11-10	R/W	**I2C receive buffer full interrupt mode select bits** 00: Byte full interrupt 01: Half-word full interrupt 10: Word full interrupt 11: Reserved
TBIM<1:0>	bit9-8	R/W	**I2C transmit buffer empty interrupt mode**

			select bits 00: Byte empty interrupt 01: Half-word empty interrupt 10: Word empty interrupt 11: Reserved
TBWEIE	bit7	R/W	**I2C transmit data write error interrupt enable bit** 0: Disabled 1: Enabled
NAIE	bit6	R/W	**I2C NACK interrupt enable bit** 0: Disabled 1: Enabled
ROIE	bit5	R/W	**I2C receive overrun interrupt enable bit** 0: Disabled 1: Enabled
TEIE	bit4	R/W	**I2C transmit error interrupt enable bit** 0: Disabled 1: Enabled
RBIE	bit3	R/W	**I2C receive buffer full interrupt enable bit** 0: Disabled 1: Enabled
TBIE	bit2	R/W	**I2C transmit buffer empty interrupt enable bit** 0: Disabled 1: Enabled
SPIE	bit1	R/W	**I2C stop bit interrupt enable bit** 0: Disabled 1: Enabled
SRIE	bit0	R/W	**I2C start bit interrupt enable bit** 0: Disabled 1: Enabled

I2C Interrupt Flag Register (I2C_IF)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000000\_00000100_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			TIDLEIF	Reserved				TBWEIF	NAIF	ROIF	TEIF	RBIF	TBIF	SPIF	SRIF

—	bit31-13	—	—
TIDLEIF	bit12	R/W	**I2C transmit idle interrupt flag bit**

			0: No idle interrupt 1:Idle interrupt occurred Clear this bit by software writing 1, and writing 0 has no effect.
—	bit11-8	—	—
TBWEIF	bit7	R/W	**I2C transmit buffer write error interrupt flag bit** 0: No write error 1: Write error occurred. The following errors may happen. - When written in word, not all buffers are empty. - When written in half-word, more than 2 buffers are not empty. - When written in byte, all buffers are full. - Write I2C_TBW<31:16> with a half-word - Write I2C_TBW<31:8> with a word Clear this bit by software writing 1 and writing 0 has no effect.
NAIF	bit6	R/W	**I2C NACK interrupt flag bit** 0: No NACK interrupt 1: NACK interrupt occurred When a NACK is received or transmitted, the NAIF will be set. Clear this bit by software writing 1 and writing 0 has no effect.
ROIF	bit5	R/W	**I2C receive overrun interrupt flag bit** 0: No overrun 1: Overrun occurred Clear this bit by software writing 1 and writing 0 has no effect.
TEIF	bit4	R/W	**I2C transmit error interrupt flag bit** 0: No transmit error. 1: Transmit error occurred. A clock is received from a master while the transmit buffers and transmit shift register are all empty. Clear this bit by software writing 1 and writing 0 has no effect.
RBIF	bit3	R	**I2C receive buffer full interrupt flag bit** 0: Not full 1: Full Clear this bit by reading the I2C_RBR register.
TBIF	bit2	R	**I2C transmit buffer empty interrupt flag bit** 0: Not empty 1:Empty Clear this bit by writing the I2C_TBW register.
SPIF	bit1	R/W	**I2C stop bit interrupt flag bit** 0: No stop bit received/transmitted 1: Stop bit received/transmitted In master mode: the flag bit will be set after transmitting a stop bit. In slave mode: the flag bit will be set after receiving a stop bit

			Clear this bit by software writing 1 and writing 0 has no effect.
SRIF	bit0	R/W	**I2C start bit interrupt flag bit** 0: No start bit interrupt 1: Start bit interrupt occurred In master mode: if the auto-call is disabled, the flag bit will be set after transmitting start bit and address bits and receiving ACK/NACK. If the auto-call is enabled, the flag bit will be set after transmitting start bit and address bits and receiving ACK. In slave mode: the flag bit will be set after receiving start bit and matched address bit, and sending ACK. Clear this bit by software writing 1 and writing 0 has no effect.

Note: For each interrupt flag in the I2C_IF register, they can be cleared by software writing 1 and writing 0 has no effect. When reading, 1 indicates that the corresponding interrupt has occurred.

I2C Transmit Data Write Register (I2C_TBW)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TBW<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TBW<15:0>															

TBW<31:0>	bit31-0	W	**Write transmit data** Write in byte: write data to TBW<7:0> Write in half-word: write data o TBW<15:0> Write in word: write data to TBW<31:0>

I2C Receive Data Read Register (I2C_RBR)

Offset address:14_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RBR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBR<15:0>															

RBR<31:0>	bit31-0	R	**Read receive data** Read in byte: read from RBR<7:0> Read in half-word: read from RBR<15:0> Read in word: read from RBR<31:0>

I2C Transmit Buffer Register (I2C_TB)

Offset address:18_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
TB3<7:0>								TB2<7:0>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TB1<7:0>								TB0<7:0>							

TB3<7:0>	bit31-24	R	Transmit buffer 3 data
TB2<7:0>	bit23-16	R	Transmit buffer 2 data
TB1<7:0>	bit15-8	R	Transmit buffer 1 data
TB0<7:0>	bit7-0	R	Transmit buffer 0 data

I2C Receive Buffer Register (I2C_RB)

Offset address:$1C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
RB3<7:0>								RB2<7:0>							

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RB1<7:0>								RB0<7:0>							

RB3<7:0>	bit31-24	R	Receive buffer 3 data
RB2<7:0>	bit23-16	R	Receive buffer 2 data
RB1<7:0>	bit15-8	R	Receive buffer 1 data
RB0<7:0>	bit7-0	R	Receive buffer 0 data

I2C Status Register (I2C_STA)

Offset address:20_H

Reset value:$00000000\_00000010\_00001111\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved														IDLE	ACK

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RBFF3	RBFF2	RBFF1	RBFF0	TBEF3	TBEF2	TBEF1	TBEF0	Reserved							

—	bit31-18	—	**—**
IDLE	bit17	R	**I2C idle flag bit** 0: Not idle 1: Idle
ACK	bit16	R	**I2C ACK bit** 0: ACK 1:NACK
RBFF3	bit15	R	**RB3 full flag bit** 0: Empty 1: Full
RBFF2	bit14	R	**RB2 full flag bit** 0: Empty 1: Full
RBFF1	bit13	R	**RB1 full flag bit** 0: Empty 1: Full
RBFF0	bit12	R	**RB0 full flag bit** 0: Empty 1: Full
TBEF3	bit11	R	**TB3 empty flag bit** 0: Full 1: Empty
TBEF2	bit10	R	**TB2 empty flag bit** 0: Full 1: Empty
TBEF1	bit9	R	**TB1 empty flag bit** 0: Full 1: Empty
TBEF0	bit8	R	**TB0 empty flag bit** 0: Full 1: Empty
—	bit7-0	—	**—**

5. 5. 10 I2C Application Notes

The chip provides one serial communication controller I2C0.

For I2C continuous data transmission, if the transmit idle interrupt flag TIDLEIF is used to trigger an interrupt and start the data transmission, take note of the following.

1) In master mode, after setting the memory address in the main program, the transmit idle interrupt flag TIDLEIF will be set to 1 and will then trigger an interrupt. During the interrupt service routine, write the data into I2C_TBW and start data transmission.
2) In slave mode, write the first data of the desired memory address to the I2C_TBW. After the master sending a read bit, the TIDLEIF bit will be set and will then trigger the interrupt. The received data will be read during the interrupt service routine. The master must send a stop bit after reading data. Otherwise, it might cause an transmit error if another read operation needs to be started right away.

The TBIM bit of the I2C_IE register has influence on the TIDLEIF. When transmitting in byte or half-word, it is better to write TBIM =10, or else it might cause multiple byte empty interrupts. If the TBIM = 00 or TBIM = 01, all the 4 transmit buffers must be written with data, or else it might also cause multiple byte empty interrupts or half-word empty interrupts.

The advantage of using the TIDLEIF is that the continuous data transmission can be accomplished only by clearing the TIDLEIF bit when the TIDLEIE bit is enabled and valid.

5. 6 Analog-to-Digital Converter (ADC)

5. 6. 1 Overview

- ◇ 12-bit conversion result with 11-bit effective resolution
- ◇ Sampling rate up to 125ksps (kilo-samples per second)
- ◇ 16 analog input channels
- ◇ ADC interrupt, able to wake up from sleep mode (only when clocked by the LRC clock)
- ◇ Configurable positive and negative reference voltage
- ◇ Configurable conversion clock
- ◇ Auto conversion and compare function

5. 6. 2 Block Diagram

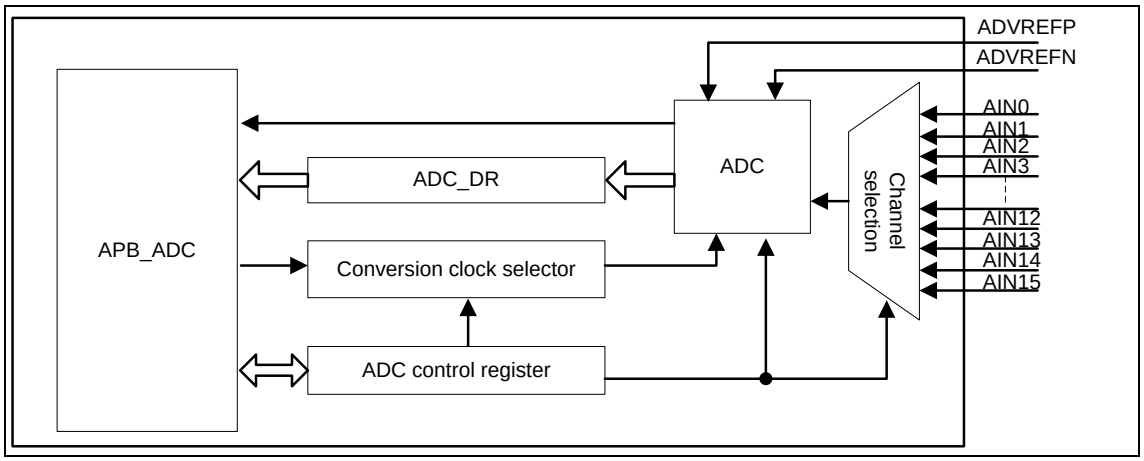


Figure 5-49 ADC Block Diagram

5. 6. 3 ADC Basic Configuration

Configure a port to an ADC analog input channel in the following way:

Configure the GPIO_PAINEB/GPIO_PBINEB register to disable digital input mode of the port. Configure the GPIO_PADIR/ GPIO_PBDIR register to disable digital output mode of the port.

Configure the CHS<4:0> bits of the ADC_CHS register to select the desired ADC channel.

To ensure of the proper operation of the ADC, the VREF_EN and IREF_EN bits of the ADC_VREFCON register, the EN of the ADC_CON0 register and the VCMBUF_EN bit of the ADC_CON1 register must be enabled.

5. 6. 4 ADC High Precision Reference Voltage

The ADC module provides one internal 1.8V or 2.6V high precision reference source. The VREF_EN and IREF_EN bits must be enabled for ADC operation.

5. 6. 5 ADC Conversion Description

Configure the CHS<4:0> bits of the ADC_CHS register to select an analog channel; configure the CLKS bits of the ADC_CON1 register to select an operating clock source; configure the CLKDIV<2:0> bits to select a clock source prescaler ratio and configure the VREFP<1:0> to select a positive reference voltage and the VREFN to select a negative reference voltage. When the VREFP<1:0> = 01, the VRBUF_EN bit must be enabled. Set the EN bit of the ADC_CON0 register to enable the ADC. Lastly, configure the TRIG bit to start a conversion. Once the conversion is complete, the TRIG bit will automatically be cleared by hardware.

Each time when a conversion is complete, the interrupt flag IF of the ADC_IF register will be set, which needs to be cleared by software. To start the next conversion, the TRIG bit needs to be re-configured.

The sampling time can be controlled by hardware (by default) or software. The fastest sampling time is one ADC clock cycle (depending on the actual application condition and the ADC clock source). The conversion time takes 14 ADC clock cycles if the conversion resolution is 12 bits. When VDD is used as a reference voltage, the fastest sampling rate is 125Ksps, which outputs 125K high precision conversion results per second. A proper ADC clock can be produced by configuring the CLKS and CLKDIV bits of the ADC_CON1 register.

The recommended setting is tabulated as below when the system clock is used as the ADC clock source and internal VREFP is used as the reference voltage.

System clock	ADC clock frequency division	ADC resolution	Conversion rate
48MHz	32	10.5 bits	93.75Ksps
32MHz	8	9 bits	250Ksps
32MHz	32	12 bits	62.5Ksps

Table 5-2 Recommended Setting

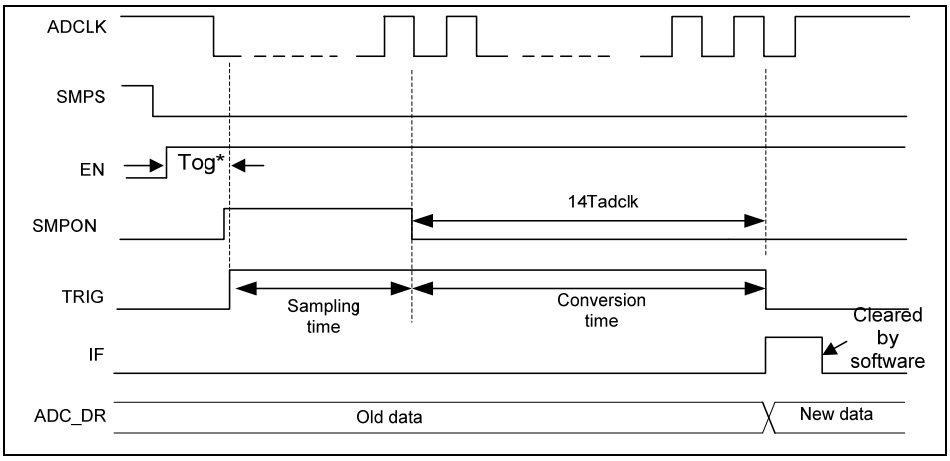


Figure 5-50 ADC Conversion Timing Diagram (SMPS = 0, sampling controlled by software)

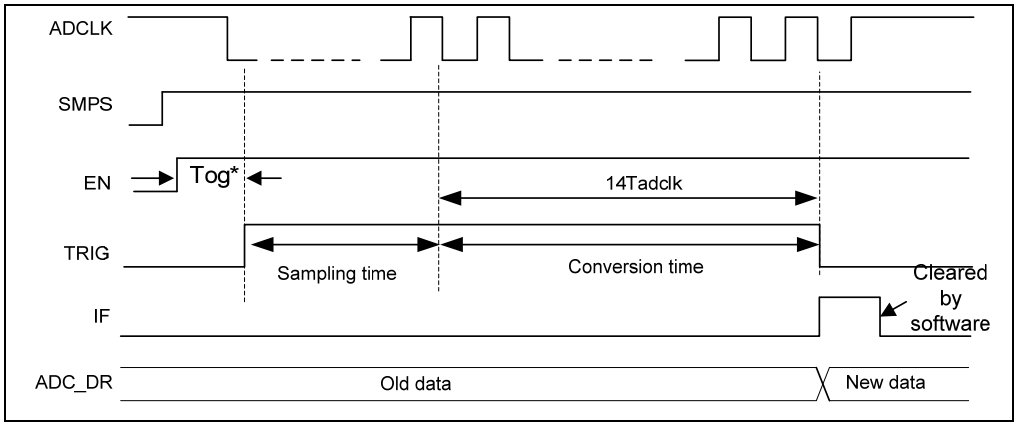


Figure 5-51 ADC Conversion Timing Diagram (SMPS = 1, sampling controlled by hardware)

Notes

1. Tog>0
2. The conversion clock period Tadclk can be configured for different frequencies by the CLKS and CLKDIV<2:0> of the ADC_CON1 register.

Start a single AD conversion

```
LDR     R0, = ADC_VREFCON       ; Enable VREF_EN and IREF_EN and
                                ; select 2.6V
LDR     R1, =0X07
STR     R1, [R0]
LDR     R0, =ADC_CON1           ; Select PCLK/32 as the ADC clock source
                                ; Select VREF 2.6V as positive reference
                                ; Enable VREF BUF and select hardware
                                ; controlled sampling time
                                ; Enable VCM BUF in high speed mode
                                ; Enable ADC high speed mode
LDR     R1, =0X03005905
STR     R1, [R0]
LDR     R0, =ADC_CHS            ; Select AIN3
LDR     R1, =0X03
STR     R1, [R0]
LDR     R0, =ADC_CON0           ; Enable ADC
LDR     R1, =0X01
STR     R1, [R0]
LDR     R0, =ADC_CON0           ; Start A/D conversion
LDR     R1, =0X03
STR     R1, [R0]
WAIT4IF
LDR     R0, =ADC_IF             ; Wait for ADC interrupt
LDR     R1, =0X01
```

```
TST     R0, R1
BEQ     WAIT4IF
STR     R1, [R0]            ; Clear ADC interrupt
LDR     R0, =ADC_CON0       ; Disable ADC
LDR     R1, =0X00
STR     R1, [R0]
```

5. 6. 6 Auto Conversion and Compare Function

The ADC module is featured with an auto conversion and compare function, allowing the ADC to automatically complete multiple conversions and calculate the mean value. The mean value is then compared with a threshold value to generate a corresponding interrupt. The mean value and the conversion result are both readable.

When the ACP_EN bit of the ADC_CON0 register is 1, write the TRIG bit to 1 to start conversion and compare in continuous mode; use the hardware, which is fixed, to control the sampling time and writing the SMPS to 0 has no effect. Prior to that, complete the following configurations first:

Set the ST bits of the ADC_CON1 register to configure the sampling time. It is recommended to have the sampling time > 1us.

Configure the TIMES of the ADC_ACPC register to select the number of conversion times within each overflow time.

Configure the OVFL_TIMES to set the overflow time for auto-conversion and compare. After an overflow occurs, the mean value of the conversion results will automatically be calculated and saved in the ADC_ACPMEAN register. If the EN bit of the ADC_CON0 register is disabled, the hardware will automatically clear the overflow counter and ADC_ACPMEAN register. If the overflow time has expired while the maximum number of conversion times has not reached (selected by TIMES<1:0>), the next cycle of overflow time will not be counted until all the required conversions are completed.

Configure the CLKS of the ADC_ACPC register to select the clock source (PCLK or LRC (32KHz) divided by 256) for the overflow counter. If the ADC is required to convert and compare during normal sleep or deep sleep, before entering sleep, the LRC divided by 256 needs to be selected as the counter clock source, and the CLKS needs to select the LRC as the clock source.

Set the CMP_MIN bit of the ADC_ACPCMP register to a low threshold value. If the MEAN_DATA of the ADC_ACPMEAN register is less than or equal to this low threshold value, the interrupt flag ACPMINIF will be set to 1.

Set the CMP_MAX bit of the ADC_ACPCMP register to a high threshold value. If the MEAN_DATA of the ADC_ACPMEAN register is greater than or equal to this high threshold value, the interrupt flag ACPMAXIF will be set to 1.

Start a single auto AD conversion

```
LDR     R0, = ADC_VREFCON       ; Enable VREF_EN and IREF_EN and
```

```
                                        ; select 2.6V
    LDR     R1, =0X07
    STR     R1, [R0]
    LDR     R0, =ADC_CON1               ; Select PCLK/32 as the ADC clock source
                                        ; Select VREF 2.6V as positive reference
                                        ; Enable VREF BUF and select hardware
                                        ; controlled sampling time
                                        ; Enable VCM BUF in high speed mode
                                        ; Enable ADC high speed mode
    LDR     R1, =0X03005905
    STR     R1, [R0]
    LDR     R0, =ADC_CHS                ; Select AIN3
    LDR     R1, =0X03
    STR     R1, [R0]
    LDR     ADC_IE, =0X07
    LDR     ADC_ACPCMP, =0X00010001     ;Set automatic high/low threshod values
    LDR     ADC_ACPC, =0x0013001F       ; ACP operating clock LRC
                                        ; Automatically convert 8 times within
                                        ;overflow duration which is 32x Tacp
    LDR     R0, =ADC_CON0               ; Enable ADC and enable auto conversion
                                        ; and compare function
    LDR     R1, =0X05
    STR     R1, [R0]
    LDR     R0, =ADC_CON0               ; Start AD conversion
    LDR     R1, =0X07
    STR     R1, [R0]
    ……
```

5. 6. 7 Special Function Registers

ADC Voltage Reference Control Register (ADC_VREFCON)

Offset address:40_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													IREF_EN	VREF_SEL	VREF_EN

—	bit31-3	—	—
IREF_EN	bit2	R/W	**IREF enable bit** 0: Disabled 1: Enabled
VREF_SEL	bit1	R/W	**Internal VREFP voltage select**

			bit 0: 1.8V 1: 2.6V
VREF_EN	bit0	R/W	**Internal VREFP enable bit** 0: Disabled 1: Enabled

Notes

1. For proper AD conversion, the VREF_EN and IREF_EN bits are enabled.
2. The bits 31-3 of the ADC_VREFCON register are reserved for test. The user is required to write 0, otherwise, it might lead to irregular behavior of ADC.

ADC Result Register (ADC_DR)

Offset address:00_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				DR<11:0>											

—	bit31-12	—	—
DR<11:0>	bit11-0	R	AD conversion result

ADC Control Register 0 (ADC_CON0)

Offset address:04_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved													ACP_EN	TRIG	EN

—	bit31-3	—	—
ACP_EN	bit2	R/W	**Auto-conversion compare function enable bit** 0: Disabled 1: Enabled
TRIG	bit1	R/W	**AD conversion trigger bit** 0: AD conversion not triggered, or AD conversion completed (cleared by hardware with high priority)

			1: Set this bit to 1 to start conversion. If SMPS is 0, writing TRIG has no effect (controlled by hardware based on SMPON software sampling and ADC conversion progress), and it cannot be read as the conversion complete flag.
EN	bit0	R/W	**AD conversion enable bit (when ACP_EN is 1, this bit is invalid)** 0: Disabled 1: Enabled

Notes

1. The TRIG bit can only be written to 1 by software and will be cleared automatically by hardware.
2. When the SMPON is off, the TRIG bit and the IF bit of the ADC_IF register both can be considered as conversion complete flags. When the SMPON is on, only the IF bit can be considered as the conversion complete flag. It is suggested to determine the completion of conversion by checking the IF bit no matter the SMPON is on or off.

ADC Control Register 1(ADC_CON1)

Offset address:08_H

Reset value:$00000000\_00000100\_00010000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved						VCMBUF_HS	VCMBUF_EN	Reserved			ST<4:0>				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved	HSEN	SMPON	SMPS	VRBUF_EN	VREFN	VREFP<1:0>		Reserved				CLKS	CLKDIV<2:0>		

—	bit31-26	—	—
VCMBUF_HS	bit25	R/W	**ADC common mode voltage VCM BUF high speed enable bit** 0: Reserved for test only 1: Enabled
VCMBUF_EN	bit24	R/W	**ADC common mode voltage VCM BUF enable bit** 0: Disabled 1: Enabled
—	bit23-21	—	**—**
ST<4:0>	bit20-16	R/W	**AD sampling time select bits (valid when controlled by hardware)** Sampling time :ST*2 + 1Tadclk
—	bit15	—	**—**

HSEN	bit14	R/W	**AD conversion speed control bit** 0: Reserved for test only 1: High speed
SMPON	bit13	R/W	**AD sampling software control bit (invalid when ACP_EN = 1)** 0: AD sampling off 1: AD sampling on
SMPS	bit12	R/W	**AD sampling mode select bit (fixed to 1 when ACP_EN =1)** 0: Software control 1: Hardware control
VRBUF_EN	bit11	R/W	**VREF BUF enable bit** 0: Disabled 1: Enabled
VREFN	bit10	R/W	**Negative reference voltage select bit** 0: Internal GND VSS 1: External reference voltage AVREFN
VREFP<1:0>	bit9-8	R/W	**Positive reference voltage select bit** 00: VDD 01: Internal reference voltage VREFP (2.6V or 1.8V) with the AVREFP pin used as a normal I/O. 10: Internal reference voltage VREFP (2.6V or 1.8V) with the AVREFP pin outputting internal reference voltage VREF 11: External reference voltage AVREFP, which should be no more than VDD and no less than 1.3V. In SWD mode, AVREP cannot be used.
—	bit7-4	—	**—**
CLKS	bit3	R/W	**Clock source select bit** 0: PCLK 1: LRC(32KHz)
CLKDIV<2:0>	bit2-0	R/W	**Clock source division rate select bits** 000 = 1:1 001 = 1:2 010 = 1:4 011 = 1:8 100 = 1:16 101 = 1:32 110 = 1:64 111 = 1:256

Notes

1. The VRBUF_EN bit must be set to 1 when the internal reference voltage VREF (2.6V or 1.8V) selected as the ADC positive reference voltage.
2. During ADC operation, the VCMBUF_HS, VCMBUF_EN and HSEN must set to 1.
3. When VDD (VREFP = 00) or the external reference voltage (VREFP = 11) are used as the reference voltage, the maximum conversion clock frequency is 2MHz. When the internal reference voltage is used ((VREFP = 01 or 10), the maximum conversion clock frequency is 1MHz.
4. When the external reference is used, the voltage cannot be less than 1.3V, otherwise, it might lead to irregular behavior of ADC.

ADC Channel Select Register (ADC_CHS)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000001\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved							VDD5_FLAG_EN	Reserved			CHS<4:0>				

—	bit31-9	—	—
VDD5_FLAG_EN	bit8	R/W	**VDD detection enable bit** 1: Enable VDD detection (for internal test only and must not be set to 1 during application) 0: Disable VDD detection (should be set to 1 during application)
—	bit7-5	—	—
CHS<4:0>	bit4-0	R/W	**Analog channel select bits** 00000: Channel 0 (AIN0) 00001: Channel 1 (AIN1) 00010: Channel 2 (AIN2) 00011: Channel 3 (AIN3) 00100: Channel 4 (AIN4) 00101: Channel 5 (AIN5) 00110: Channel 6 (AIN6) 00111: Channel 7 (AIN7) 01000: Channel 8 (AIN8) 01001: Channel 9 (AIN9) 01010: Channel 10 (AIN10) 01011: Channel 11 (AIN11) 01100: Channel 12 (AIN12)

			01101: Channel 13 (AIN13) 01110: Channel 14 (AIN14) 01111: Channel 15 (AIN15) Others: No channel connected

ADC Interrupt Enable Register (ADC_IE)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												ACPOVIE	ACPMAXIE	ACPMINIE	IE

—	bit31-4	—	—
ACPOVIE	bit3		**ADC overflow interrupt enable bit** 0: Disabled 1: Enabled
ACPMAXIE	bit2	R/W	**Interrupt enable bit (mean value higher than the high threshold)** 0: Disabled 1: Enabled
ACPMINIE	bit1	R/W	**Interrupt enable bit (mean value lower than the low threshold)** 0: Disabled 1: Enabled
IE	bit0	R/W	**ADC interrupt enable bit** 0: Disabled 1: Enabled

ADC Interrupt Flag Register (ADC_IF)

Offset address:14_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												ACPOVIF	ACPMAXIF	ACPMINIF	IF

—	bit31-4	—	—
ACPOVIF	bit3	R/W	**Overflow interrupt flag bit** 0: The overflow time has not reached 1: The overflow time has reached (set to 1 by hardware and cleared by software) This bit is cleared by software writing 1 and writing 0 has no effect.
ACPMAXIF	bit2	R/W	**Interrupt flag bit (mean value higher than the high threshold)** 0: Mean value is less than the high threshold value 1: Mean value is greater than or equal to the high threshold value. (set to 1 by hardware and cleared by software) This bit is cleared by software writing 1 and writing 0 has no effect.
ACPMINIF	bit1	R/W	**Interrupt flag bit (mean value lower than the low threshold)** 0: Mean value is higher than the low threshold value 1: Mean value is less than or equal to the low threshold value (set to 1 by hardware and cleared by software) This bit is cleared by software writing 1 and writing 0 has no effect.
IF	bit0	R/W	**ADC interrupt flag bit** 0: Conversion in progress 1:Conversion completed (set to 1 by hardware and cleared by software) This bit is cleared by software writing 1 and writing 0 has no effect.

Notes

1. When ADC interrupts are disabled, the corresponding flag will be set if an interrupt is triggered, but no interrupt request will be generated.
2. For all interrupt flag bits of ADC_IF register, they can be cleared by software writing 1 and writing 0 has no effect. When they are being read, 1 indicates that an interrupt has occurred.

ADC Auto Conversion Compare Control Register (ADC_ACPC)

Offset address:28_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved											CLKS			TIMES<1:0>	

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				OVFL_TIME<11:0>											

—	bit31-21	—	**—**
CLKS	bit20	R/W	**Overflow counter clock source select bit** 0: F_{PCLK}/256 (PCLK divided by 256) 1: F_{LRC}/256 (LRC divided by 256)
—	bit19-18	—	**—**
TIMES<1:0>	bit17-16	R/W	**Conversion times select bits (within the duration select by OVFL_TIME)** 00: Once 01: Twice 10: Four times 11: Eight times
—	bit15-12	—	**—**
OVFL_TIME<11:0>	bit11-0	R/W	**Overflow time select bits, configurable range is 0~$9C3_H$ and counter clock period is Tacp.** 0: 1 xTacp 1: 2 x Tacp 2: 3 x Tacp … $9C3_H$: 2500 x Tacp Tacp is the clock source period of ACP overflow counter select by the CLKS.

Note: The overflow time selected by the OVFL_TIME bits must be greater than one cycle of sampling time and conversion time.

ADC Auto Conversion Compare Threshold Register (ADC_ACPCMP)

Offset address:30_H

Reset value:00001111_11111111_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				CMP_MAX<11:0>											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				CMP_MIN<11:0>											

—	bit31-28	—	—
CMP_MAX<11:0>	bit 27-16	R/W	**High threshold value**
—	bit15-12	—	—
CMP_MIN<11:0>	bit 11-0	R/W	**Low threshold value**

ADC Mean Register (ADC_ACPMEAN)

Offset address:34_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved				MEAN_DATA <11:0>											

—	bit31-12	—	—
MEAN_DATA<11:0>	bit 11-0	R	**Mean value of the conversion results**

5. 7 Real Time Clock (RTC)

5. 7. 1 Overview

- ◇ Only reset by POR, supports program write protection and immune to system interference.
- ◇ Clock sources: use the external 32.768KHz crystal oscillator for high-precision requirement; use the internal LRC clock for low-precision requirement; use PCLK or PCLK divided for 256 for the RTC being used as a normal counter
- ◇ High-precision digital calibration
- ◇ Two calibration accuracies: ±1.5 ppm (or ±0.5ppm) in calibration range ±384ppm (or ±128ppm)
- ◇ Time with seconds, minutes, hours; calendar with days, weeks, months and years, in BCD format.
- ◇ 5 programmable timer interrupts
- ◇ 2 programmable calendar alarms
- ◇ 1 configurable clock output
- ◇ Auto leap year recognition till year 2099
- ◇ 12-hour or 24 –hour format
- ◇ Low consumption: operating current =0.5μA (typ.) @ VDD=5.0V

5. 7. 2 RTC Write Protection

The RTC write protection register RTC_WP is used to protect other RTC associated registers from writing by mistakes (the RTC_WP register itself excluded).

The RTC_WP register is a virtual register. Prior to writing to those RTC associated registers, the write protection needs to be removed by writing 0x55AAAA55 to the RTC_WP register. Writing other values to the RTC_WP register will enable write protection. Any write operation will be ignored when those registers are write protected.

Read the RTC_WP register to determine whether the RTC module is under write protection. When read as 0x55AAAA55, the RTC is not write protected and write operations can be performed onto the registers. When read as 0x00000000, the RTC module is currently under write protection. 0x55AAAA55 and 0x00000000 are the only two possible readouts.

5. 7. 3 Time and Date Setting

A RTC time counter is accessed through buffers and cannot be directly read or written because the APB clock and the RTC time counter are not synchronous. The RTC_HMS and RTC_YMDW registers are buffers for reading and writing a RTC time counter. How to read a time counter is described in following procedures.

1) Write the write/read bit TMWR = 0 to select the read operation
2) Write the write/read trigger bit TMUP = 1 to trigger a read operation

3) After reading, the value of the time counter will be read to the RTC_HMS and RTC_YMDW registers, and the TMUP bit will be automatically cleared.

How to write a time counter is described in following procedures (after a reset or the last read/write operation is completed)

1) Configure the HSWI bit to select 12hr or 24hr format
2) Write a value (s) to the RTC_HMS and/or RTC_YMDW register
3) Write TMWR = 1 to select the write operation
4) Write TMUP = 1 to trigger a write operation
5) After writing, the updated values in the RTC_HMS and RTC_YMDW will be written into the time counter, and the TMUP bit will be automatically cleared.

Notes

1. Writing a time counter is to write the updated values of the RTC_HMS and/or RTC_YMDW register to the corresponding time counter after a reset or a read operation. Other time counters those are not updated are not affected by a write operation.
2. When the TMUP bit is 1, the current write operation can be aborted by clearing the TMWR bit. The result of the aborted write operation is undetermined.

The time and date registers employ the BCD coding. The second counter counts from 00 to 59, and when it completes one cycle, it transitions from 59 to 00. The minute counter also works the same way as the second counter. The hour counter counts depending on the format selected. When the 12hr format is selected, after one cycle, it transitions from PM11 to AM12 or AM11 to PM 12. When the 24hr format is selected, after one cycle, it transitions from 23 to 00.

The week day counter is a cyclic shift register. During setup, the corresponding week day is set to 1, and others are set to 0.

The day counter transitions to the next month by adding one to the last day of the current month.

January, March, May, July, August, October and December: 1 to 31

April, June, September and November: 1 to 30

February: 1 to 28 (normal year), 1 to 29 (leap year)

The month counter counts from 1 to 12 and transitions to 1 after one cycle.

The year counter counts from 00 to 99 (00, 04, 08, …, 92 and 96 are leap years). After reaching 99, it will no longer transition to 00.

The 12/24hr mapping table is as follows:

24Hr Format	12Hr Format	24Hr Format	12Hr Format
00	12 (AM12)	12	32 (PM12)
01	01 (AM1)	13	21 (PM1)
02	02 (AM2)	14	22 (PM2)
03	03 (AM3)	15	23 (PM3)
04	04 (AM4)	16	24 (PM4)
05	05 (AM5)	17	25 (PM5)
06	06 (AM6)	18	26 (PM6)
07	07 (AM7)	19	27 (PM7)
08	08 (AM8)	20	28 (PM8)
09	09 (AM9)	21	29 (PM9)
10	10 (AM10)	22	30 (PM10)
11	11 (AM11)	23	31 (PM11)

Table 5-3 12/24 Hour Format Mapping

5. 7. 4 RTC Interrupt Sources

There are 7 interrupt sources available for the RTC module.

- ◇ Week alarm interrupt WAFG
- ◇ Day alarm interrupt DAFG
- ◇ 5 periodic interrupts: month interrupt, day interrupt, hour interrupt, minute interrupt and second interrupt

Each interrupt source has an independent enable bit, which affects the generation of the interrupt request (IRQ) but do not affect the interrupt flag. Even the corresponding interrupt is disabled; the flag bit can still be polled. In the case that multiple interrupts are enabled, each interrupt can generate the IRQ by an OR operation. The IRQ will only be removed after all interrupt flags are cleared.

5. 7. 5 RTC Timer

If an application system requires high precision in RTC timer, the external 32.768KHz crystal oscillator is used as the RTC clock source. In this case, the external clock pins OSCxI and OSCxO are connected with the 32.768KHz crystal oscillator. In addition, the corresponding configuration word and control registers are configured as follows:

1) Configure the XTAL as a low speed oscillator through the CFG_OSCMD
2) Select the system clock source by the CLK_SEL<1:0> bits of the SCU_SCLKEN0 register
3) Write XTAL_EN = 1 of the SCU_SCLKEN1 register to enable the external clock oscillator
4) Write CLKS<1:0> = 00 of the RTC_CON register to select the external 32.768KHz oscillator as the system clock source.

In deep sleep mode, if the external 32.768KHz oscillator is required to clock the RTC, then the MOSC_EN of the SCU_WAKEUPTIME register must be written to 1 to allow the

XTAL clock to operate during deep sleep.

If an application system does not require high precision RTC, the internal LRC clock (about 32KHz) can be used as the clock source.

5. 7. 6 Special Function Registers

RTC Write Protection Register (RTC_WP)

Offset address:20_H

Reset value:00000000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
WP<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WP<15:0>															

WP<31:0>	bit31-0	R/W	0x0000_0000: RTC module is write protected. 0x55AA_AA55: Write operations can be performed on RTX module Writing other values can also enable write protection.

Note: The RTC_CON, RTC_CAL, RTC_WA, RTC_DA, RTC_HMS, RTC_YMDW, RTC_IE and RTC_IF registers are under protection of the RTC_WP register.

RTC Control Register (RTC_CON)

Offset address:00_H

Reset value:00000000_00000000_00001000_10000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								PON	XST	CLKS<1:0>		Reserved	HSWI	TMWR	TMUP

—	bit31-8	—	—
PON	bit7	R/W	**RTC power on reset flag bit** (cleared by software) 0: Normal RTC operation 1: RTC power on reset detected (must be cleared by software, only then the RTC starts working)
XST	bit6	R	**Oscillator stop flag bit** 0:Oscillator continues to work 1:Oscillator stopped

CLKS<1:0>	bit5-4	R/W	**RTC clock source select bit (must be configured before writing time data)** 00: 32.768 KHz oscillator clock source (high RTC precision) 01: LRC clock (low RTC precision) 10: PCLK/256 (RTC used as a normal counter) 11: PCLK (RTC used as a normal counter)
—	bit3	—	—
HSWI	bit2	R/W	**12/24 hour format select bit (must be configured before writing time data)** 0: 12 hour format 1: 24 hour format
TMWR	bit1	R/W	**Timer counter write read select bit** 0: Read operation 1: Write operation
TMUP	bit0	R/W	**Time counter write read trigger bit** (only can be written to 1 and automatically cleared on the completion of a read or write operation) 0: Write or read operation has been completed. 1: Write or read operation is in progress

Notes

1. The RTC module will stay in reset state after a power on reset. Only when the PON bit is cleared will the RTC enter operating mode.
2. To ensure of high precision, writing the CLKS bits to 00 is recommended, to select the external 32.768KHz crystal oscillator. For an application requiring low precision, the LRC clock source can be applied. Only when the RTC is used as a normal timer will the CLKS bits set to 10 or 11.

RTC Calibration Register (RTC_CAL)

Offset address:04_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			CLKC	COCR<2:0>			DEV	CALF<7:0>							

—	bit31-13	—	—
CLKC	bit12	R/W	**RTC output port enable bit** 0: Disabled 1: Enabled (The port needs to be multiplexed with RTCO for RTC output)

COCR<2:0>	bit11-9	R/W	**RTC output frequency select bits** 000: 32KHz 001: 1024Hz 010: 32Hz 011: 1Hz 100: Output 1Hz clock after calibration 111~101: Reserved
DEV	bit8	R/W	**Calibration select bit** 0: Calibrate every 20 seconds (calibration takes place on 00, 20 and 40 seconds) 1: Calibrate every 60 seconds (calibration takes place on 00 second)
CALF<7:0>	bit7-0	R/W	**RTC calibration value**

Notes

1. If CALF<6:1> = 000000, then the calibration value increases or decreases to 0.
   If CALF<7> = 0, then the calibration value increases by ((CALF<6:0>) - 1) x 2.
   If CALF<7> =1, then the calibration value decreases by ((~CALF<6:0>) + 1) x 2.
2. If DEV = 0, the calibration time step is 3.051ppm in the range -384ppm~384ppm, and the calibration accuracy is up to ±1.5ppm.
   If DEV = 1, the calibration time step is 1.017ppm in the range -128ppm~128ppm, and the calibration accuracy is up to ±0.5ppm.

RTC Week Alarm Register (RTC_WA)

Offset address:08_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved									WW<6:0>						

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		WH<5:0>						Reserved	WM<6:0>						

—	bit31-23	—	—
WW<6:0>	bit22~16	R/W	WW<6>: Saturday alarm bit WW<5>: Friday alarm bit WW<4>: Thursday alarm bit WW<3>: Wednesday alarm bit WW<2>: Tuesday alarm bit WW<1>: Monday alarm bit WW<0>: Sunday alarm bit
—	bit15-14	—	—

WH<5:0>	bit13~8	R/W	WH<5>: 24 hr format:20 hours bit 12hr format:1 presents PM, 0 presents AM WH<4>: 10 hours bit WH<3>: 8 hours bit WH<2>: 4 hours bit WH<1>: 2 hours bit WH<0>: 1 hour bit
—	bit7	—	—
WM<6:0>	bit6~0	R/W	WM<6>: 40 minutes bit WM<5>: 20 minutes bit WM<4>: 10 minutes bit WM<3>: 8 minutes bit WM<2>: 4 minutes bit WM<1>: 2 minutes bit WM<0>: 1 minute bit

RTC Day Alarm Register (RTC_DA)

Offset address:$0C_H$

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DH<5:0>						Reserved		DM<6:0>					

—	bit31-14	—	—
DH<5:0>	bit13~8	R/W	DH<5>: 24 hr format:20 hours bit 12hr format:1 presents PM, 0 presents AM DH<4>: 10 hours bit DH<3>: 8 hours bit DH<2>: 4 hours bit DH<1>: 2 hours bit DH<0>: 1 hour bit
—	bit7	—	—
DM<6:0>	bit6~0	R/W	DM<6>: 40 minutes bit DM<5>: 20 minutes bit DM<4>: 10 minutes bit DM<3>: 8 minutes bit DM<2>: 4 minutes bit DM<1>: 2 minutes bit DM<0>: 1 minute bit

RTC Hour Minute Second Register (RTC_HMS)

Offset address:10_H

Reset value:00000000_xxxxxxxx_xxxxxxxx _xxxxxxxx $_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										HOUR<5:0>					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		MIN<6:0>						Reserved		SEC<6:0>					

—	bit31-22	—	—
HOUR<5:0>	bit21~16	R/W	HOUR<5>: 24 hr format:20 hours bit 12hr format:1 presents PM, 0 presents AM HOUR<4>: 10 hours bit HOUR<3>: 8 hours bit HOUR<2>: 4 hours bit HOUR<1>: 2 hours bit HOUR<0>: 1 hour bit
—	bit15	—	—
MIN<6:0>	bit14~8	R/W	MIN<6>: 40 minutes bit MIN<5>: 20 minutes bit MIN<4>: 10 minutes bit MIN<3>: 8 minutes bit MIN<2>: 4 minutes bit MIN<1>: 2 minutes bit MIN<0>: 1 minute bit
—	bit7	—	—
SEC<6:0>	bit6~0	R/W	SEC<6>: 40 seconds bit SEC<5>: 20 seconds bit SEC<4>: 10 seconds bit SEC<3>: 8 seconds bit SEC<2>: 4 seconds bit SEC<1>: 2 seconds bit SEC<0>: 1 second bit

RTC Year Month Day Week Register (RTC_YMDW)

Offset address:14_H

Reset value:xxxxxxxx_xxxxxxxx_xxxxxxxx _xxxxxxxx$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
YEAR<7:0>								Reserved			MON<4:0>				

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved		DAY<5:0>						Reserved	WEEK<6:0>						

YEAR<7:0>	bit31~24	R/W	YEAR<7>: 80 years bit YEAR<6>: 40 years bit YEAR<5>: 20 years bit YEAR<4>: 10 years bit YEAR<3>: 8 years bit YEAR<2>: 4 years bit YEAR<1>: 2 years bit YEAR<0>: 1 year bit
—	bit23-21	—	—
MON<4:0>	bit20~16	R/W	MON<4>: 10 months bit MON<3>: 8 months bit MON<2>: 4 months bit MON<1>: 2 months bit MON<0>: 1 month bit
—	bit15-14	—	—
DAY<5:0>	bit13~8	R/W	DAY<5>: 20 days bit DAY<4>: 10 days bit DAY<3>: 8 days bit DAY<2>: 4 days bit DAY<1>: 2 days bit DAY<0>: 1 day bit
—	bit7	—	—
WEEK<6:0>	bit6~0	R/W	WEEK<6>: Saturday bit WEEK<5>: Friday bit WEEK<4>: Thursday bit WEEK<3>: Wednesday bit WEEK<2>: Tuesday bit WEEK<1>: Monday bit WEEK<0>: Sunday bit

RTC Interrupt Enable Register (RTC_IE)

Offset address:18_H

Reset value:00000000_00000000_00000000_10000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						WALE	DALE	Reserved			MONIE	DAYIE	HORIE	MINIE	SCDIE

—	bit31-10	—	—
WALE	bit9	R/W	**Week alarm enable bit** 0: Disabled 1: Enabled
DALE	bit8	R/W	**Day alarm enable bit** 0: Disabled 1: Enabled
—	bit7-5	—	—
MONIE	bit4	R/W	**Month interrupt enable bit** 0: Disabled 1: Enabled
DAYIE	bit3	R/W	**Day interrupt enable bit** 0: Disabled 1: Enabled
HORIE	bit2	R/W	**Hour interrupt enable bit** 0: Disabled 1: Enabled
MINIE	bit1	R/W	**Minute interrupt enable bit** 0: Disabled 1: Enabled
SCDIE	bit0	R/W	**Second interrupt enable bit** 0: Disabled 1: Enabled

RTC Interrupt Flag Register (RTC_IF)

Offset address:$1C_H$

Reset value:00000000_00000000_00000000_10000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved						WAFG	DAFG	Reserved			MONIF	DAYIF	HORIF	MINIF	SCDIF

—	bit31-10	—	—

WAFG	bit9	R/W	**Week alarm flag bit** 0: Week alarm event did not occur 1: Week alarm event occurred Clear the flag bit by software writing 1, and writing 0 has no effect.
DAFG	bit8	R/W	**Day alarm flag bit** 0: Day alarm event did not occur 1: Day alarm event occurred Clear the flag bit by software writing 1, and writing 0 has no effect.
—	bit7-5	—	—
MONIF	bit4	R/W	**Month interrupt flag bit** (interrupt occurs on 00hr 00min 00sec on the first day in a month) 0: No month interrupt 1: Month interrupt occurred Clear the flag bit by software writing 1, and writing 0 has no effect.
DAYIF	bit3	R/W	**Day interrupt flag bit** (interrupt period is every day) 0: No day interrupt 1: Day interrupt occurred Clear the flag bit by software writing 1, and writing 0 has no effect.
HORIF	bit2	R/W	**Hour interrupt flag bit** (interrupt period is every hour) 0: No hour interrupt 1: Hour interrupt occurred Clear the flag bit by software writing 1, and writing 0 has no effect.
MINIF	bit1	R/W	**Minute interrupt flag bit** (interrupt period is every minute) 0: No minute interrupt 1: Minute interrupt occurred Clear the flag bit by software writing 1, and writing 0 has no effect.
SCDIF	bit0	R/W	**Second interrupt flag bit** (interrupt period is every second) 0: No second interrupt 1: Second interrupt occurred Clear the flag bit by software writing 1, and writing 0 has no effect.

Note: For all interrupt flags of the RTC_IF register, only writing 1 to them can clear the flags and writing 0 has no effect. When reading, 1 indicates that the corresponding interrupt has occurred.

5. 8 Liquid Crystal Display Controller (LCDC)

5. 8. 1 Overview

- ◇ 8COM x 28SEG
- ◇ Grayscale adjustment
- ◇ Configurable twinkling frequency
- ◇ Adjustable internal bias resistor
- ◇ 3 clock source options

5. 8. 2 Block Diagram

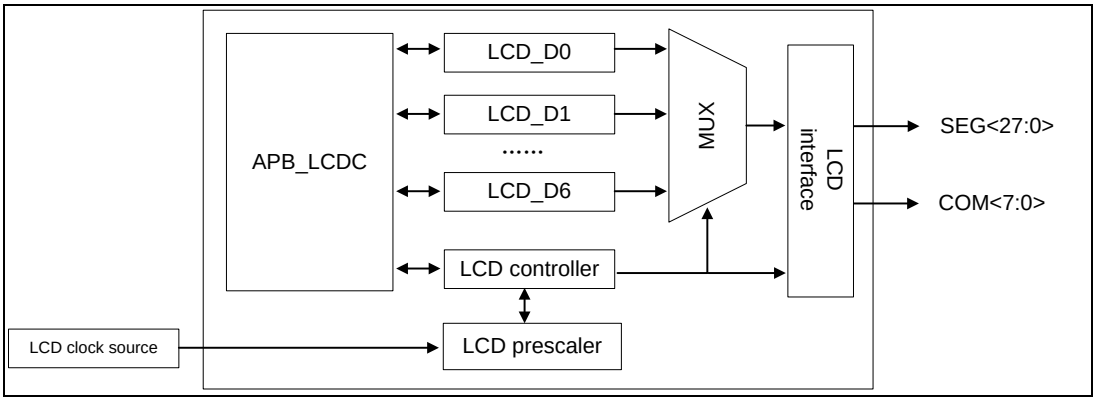


Figure 5-52 LCD Driver Block Diagram

5. 8. 3 LCD Basic Settings

Configure the LCDC drive module in the following procedure:

1) Configure the SEL bit of the LCD_CON1 register to select the LCD driver
2) Configure the LCDC operating clock
3) Configure the LCDC port: configure the COMS<2:0> bits of the LCD_CON0 register to select the common; configure the SEG<27:0> bits of the LCD_SEL register to enable the segment and select bias mode; configure the GPIO_PAINEB/ GPIO_PBINEB register to disable the corresponding digital inputs and configure the GPIO_PADIR/GPIO_PBDIR register to disable the corresponding digital outputs.
4) Initialize the pixel data register LCD_D0~LCD_D6.
5) Enable the LCDC drive module

5. 8. 4 LCD Bias Selection

There are internal bias voltage and external bias voltage for selection.

When VLCDEN = 0 of the LCD_CON0 register, the internal bias voltage is selected, and the internal reference source is the VDD. The following bias types are available.

- ◇ 1/2 Bias (3 voltage levels :VSS,1/2 VBIAS and VBIAS)

◇ 1/3 Bias (4 voltage levels:VSS,1/3 VBIAS, 2/3 VBIAS and VBIAS)

◇ 1/4 Bias (5 voltage levels:VSS,1/4 VBIAS, 2/4 VBIAS, 3/4 VBIAS and VBIAS)

The user only needs to select the desired bias type, and the bias voltage will automatically be generated by the internal circuit.

When VLCDEN = 1 of the LCD_CON0 register, the external bias voltage is selected. The external bias voltage is input from PA2~PA5 which are required to be in analog mode. The resistor network of the external bias voltage has to match the bias type selected by the BIAS<1:0>. Table 5-4 shows the input configuration of the external bias voltage on PA2~PA5.

	1/2 Bias	1/3 Bias	1/4 Bias
VLCD1(PA2)	1/2 VDD	1/3 VDD	1/4 VDD
VLCD2(PA3)	1/2 VDD	2/3 VDD	2/4 VDD
VLCD3(PA4)	VDD	VDD	3/4 VDD
VLCD4(PA5)	—	—	VDD

Table 5-4 LCDC External Bias Voltage Input Configuration

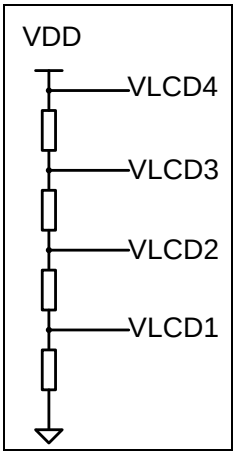


Figure 5-53 1/4 VDD External Bias Voltage Reference Circuit

5. 8. 5 LCD Pixel Mapping Table

COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
D6[31]	D6[30]	D6[29]	D6[28]	D6[27]	D6[26]	D6[25]	D6[24]	SEG27
D6[23]	D6[22]	D6[21]	D6[20]	D6[19]	D6[18]	D6[17]	D6[16]	SEG26
D6[15]	D6[14]	D6[13]	D6[12]	D6[11]	D6[10]	D6[9]	D6[8]	SEG25
D6[7]	D6[6]	D6[5]	D6[4]	D6[3]	D6[2]	D6[1]	D6[0]	SEG24
D5[31]	D5[30]	D5[29]	D5[28]	D5[27]	D5[26]	D5[25]	D5[24]	SEG23
D5[23]	D5[22]	D5[21]	D5[20]	D5[19]	D5[18]	D5[17]	D5[16]	SEG22
D5[15]	D5[14]	D5[13]	D5[12]	D5[11]	D5[10]	D5[9]	D5[8]	SEG21
D5[7]	D5[6]	D5[5]	D5[4]	D5[3]	D5[2]	D5[1]	D5[0]	SEG20
D4[31]	D4[30]	D4[29]	D4[28]	D4[27]	D4[26]	D4[25]	D4[24]	SEG19
D4[23]	D4[22]	D4[21]	D4[20]	D4[19]	D4[18]	D4[17]	D4[16]	SEG18

COM7	COM6	COM5	COM4	COM3	COM2	COM1	COM0	
D4[15]	D4[14]	D4[13]	D4[12]	D4[11]	D4[10]	D4[9]	D4[8]	SEG17
D4[7]	D4[6]	D4[5]	D4[4]	D4[3]	D4[2]	D4[1]	D4[0]	SEG16
D3[31]	D3[30]	D3[29]	D3[28]	D3[27]	D3[26]	D3[25]	D3[24]	SEG15
D3[23]	D3[22]	D3[21]	D3[20]	D3[19]	D3[18]	D3[17]	D3[16]	SEG14
D3[15]	D3[14]	D3[13]	D3[12]	D3[11]	D3[10]	D3[9]	D3[8]	SEG13
D3[7]	D3[6]	D3[5]	D3[4]	D3[3]	D3[2]	D3[1]	D3[0]	SEG12
D2[31]	D2[30]	D2[29]	D2[28]	D2[27]	D2[26]	D2[25]	D2[24]	SEG11
D2[23]	D2[22]	D2[21]	D2[20]	D2[19]	D2[18]	D2[17]	D2[16]	SEG10
D2[15]	D2[14]	D2[13]	D2[12]	D2[11]	D2[10]	D2[9]	D2[8]	SEG9
D2[7]	D2[6]	D2[5]	D2[4]	D2[3]	D2[2]	D2[1]	D2[0]	SEG8
D1[31]	D1[30]	D1[29]	D1[28]	D1[27]	D1[26]	D1[25]	D1[24]	SEG7
D1[23]	D1[22]	D1[21]	D1[20]	D1[19]	D1[18]	D1[17]	D1[16]	SEG6
D1[15]	D1[14]	D1[13]	D1[12]	D1[11]	D1[10]	D1[9]	D1[8]	SEG5
D1[7]	D1[6]	D1[5]	D1[4]	D1[3]	D1[2]	D1[1]	D1[0]	SEG4
D0[31]	D0[30]	D0[29]	D0[28]	D0[27]	D0[26]	D0[25]	D0[24]	SEG3
D0[23]	D0[22]	D0[21]	D0[20]	D0[19]	D0[18]	D0[17]	D0[16]	SEG2
D0[15]	D0[14]	D0[13]	D0[12]	D0[11]	D0[10]	D0[9]	D0[8]	SEG1
D0[7]	D0[6]	D0[5]	D0[4]	D0[3]	D0[2]	D0[1]	D0[0]	SEG0

Table 5-5 LCDC Pixel Mapping Table

5. 8. 6 LCDC Clock Sources

The LCDC module has three clock source options.

- ◇ LRC clock divided by 4: active during normal sleep and deep sleep
- ◇ LOSC clock divided by 4: active during normal sleep and deep sleep
- ◇ PCLK clock divided by 4096: inactive during normal sleep and deep sleep

The PRS<5:0> bits of the LCD_CON0 register select the prescaler ratio of the LCD operating clock.

5. 8. 7 LCD Frame Frequency

Below shows the frame frequency calculations.

COMS<2:0> = 001: frame frequency = clock source /(1 × 4 ×(PRS<5:0>+1))

COMS<2:0> = 010: frame frequency = clock source /(1 × 6 ×(PRS<5:0>+1))

COMS<2:0> = 011: frame frequency = clock source /(1 × 8 ×(PRS<5:0>+1))

COMS<2:0> = 10x: frame frequency = clock source /(2 × 2 ×(PRS<5:0>+1))

COMS<2:0> = 11x: frame frequency = clock source /(1 × 3 ×(PRS<5:0>+1))

5. 8. 8 LCD Twinkling

Set the FLIK bit of the LCD_CON0 register to enable the LCD twinkling function. The

twinkling on and off time can be configured through the LCD_TWI register. Prior to enabling the twinkling function, configure the LCD_TWI register first.

5. 8. 9 LCD Low Power Mode

The RS<2:0> bits of the LCD_CON0 register control the value of the internal bias resistor. Reducing the resistance of the bias resistor gives better display effect but leads to more power consumption. In order to reduce the power consumption while maintaining the display effect, the LCD module offers the automatic internal bias resistor switching, which allows the user to select the proper bias resistor based on the actual display effect. The RT<2:0> bits of the LCD_CON0 register control the switching time. Only when the LCD is configured in switching mode, the RT<2:0> bits are valid.

The BVS<3:0> bits of the LCD_CON0 register control the grayscale of the LCD. The higher the grayscale voltage, the better the display effect but more the power consumption. The user can set the desired the grayscale based on the actual display effect and power requirement.

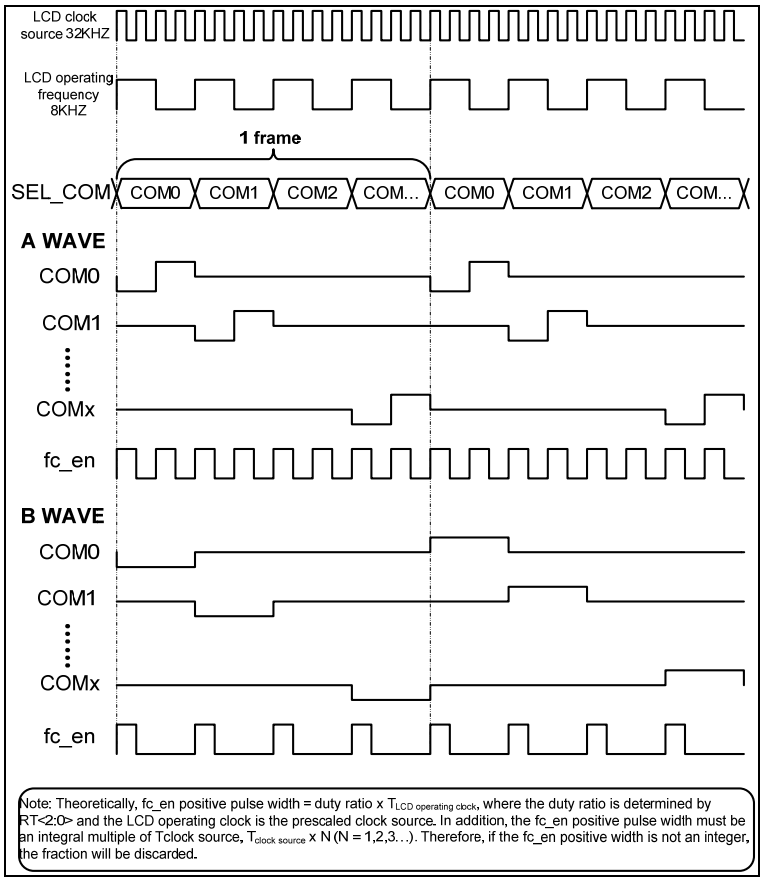


Figure 5-54 Fast Charging Discharging Diagram

5. 8. 10 Special Function Registers

LCD Control Register 0(LCD_CON0)

Offset address:00_H

Reset value:11110000_00000000_00000000_00000000$_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
BVS<3:0>				Reserved						PRS<5:0>					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
RT<2:0>			RS<2:0>			BIAS<1:0>		CLK_SEL<1:0>		WFS	FLIK	VLCDEN	COMS<2:0>		

BVS<3:0>	bit31-28	R/W	**LCD bias voltage bits (reference source VDD)** 0000: V_{BIAS} = VDD/2 0001: V_{BIAS} = VDD× (16/30) 0010: V_{BIAS} = VDD× (17/30) 0011: V_{BIAS} = VDD× (18/30) 0100: V_{BIAS} = VDD× (19/30) 0101: V_{BIAS} = VDD× (20/30) 0110: V_{BIAS} = VDD× (21/30) 0111: V_{BIAS} = VDD× (22/30) 1000: V_{BIAS} = VDD× (23/30) 1001: V_{BIAS} = VDD× (24/30) 1010: V_{BIAS} = VDD× (25/30) 1011: V_{BIAS} = VDD× (26/30) 1100: V_{BIAS} = VDD× (27/30) 1101: V_{BIAS} = VDD× (28/30) 1110: V_{BIAS} = VDD× (29/30) 1111: V_{BIAS} = VDD
—	bit27-22	—	**—**
PRS<5:0>	bit21-16	R/W	**LCDC clock source prescaler ratio select bits** 000000 = 1:1 000001 = 1:2 000010 = 1:3 … … 111111 = 1:64
RT<2:0>	bit15-13	R/W	**LCDC 60kΩ resistor hold time select bits** 000: 1/4 COM cycle 001: 1/8 COM cycle 010: 1/16 COM cycle 011: 1/32 COM cycle 100: 1/64 COM cycle Others: Reserved
RS<2:0>	bit12-10	R/W	**LCDC internal bias resistor select bits** 000: 225KΩ (in 1/3 and 1/4 BIAS),150KΩ (in 1/2 BIAS)

			001: 900KΩ (in 1/3 and 1/4 BIAS), 600KΩ (in 1/2 BIAS) 01x: 60KΩ (in 1/3 and 1/4 BIAS), 40KΩ (1/2 BIAS) 100: Auto switching between 60KΩ and 225KΩ (in 1/3 and 1/4 BIAS); auto switching between 40KΩ and 150KΩ (in 1/2 BIAS) 101: Auto switching between 60KΩ and 900KΩ (in 1/3 and 1/4 BIAS); auto switching between 40KΩ and 600KΩ(in 1/2 BIAS) Others: Reserved
BIAS<1:0>	bit9-8	R/W	**LCDC bias type select bits** 00: 1/2 Bias 01: 1/3 Bias 10: Reserved 11: 1/4 Bias
CLK_SEL<1:0>	bit7-6	R/W	**LCDC clock source select bits** 00: LRC clock divided by 4 (internal 32KHz clock) 01: LOSC clock divided by 4 (external 32KHz clock) 10: PCLK clock divided by 4096 11: Reserved
WFS	bit5	R/W	**LCD drive wave type select bit** 0: Type A waveform (the phase changes within each common type) 1: Type B waveform (the phase changes on each frame boundary)
FLIK	bit4	R/W	**LCD twinkling enable bit** 0: Disabled 1: Enabled
VLCDEN	bit3	R/W	**External bias voltage enable bit** 0: Disabled 1: Enabled
COMS<2:0>	bit2-0	R/W	**Common select bits (see table below)**

COMS<2:0>	COM Multiplex	Max. Pixel
11X	1/3 (COM2~COM0)	84
10X	1/2 (COM1~COM0)	56
011	1/8 (COM7~COM0)	224
010	1/6 (COM5~COM0)	168
001	1/4 (COM3~COM0)	112
000	—	—

Table 5-6 Common Multiplex Configuration

LCD Twinkle Register (LCD_TWI)

Offset address:04_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
TOFF<7:0>								TON<7:0>							

—	bit31-16	—	—
TOFF<7:0>	bit15-8	R/W	**LCD off duration** (TOFF+1) x 0.25 seconds
TON<7:0>	bit7-0	R/W	**LCD on duration** (TON+1) x 0.25 seconds

LCD Segment Enable Register (LCD_SEL)

Offset address:08_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved				SEG<27:16>											

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEG<15:0>															

—	bit31-28	—	—
SEG<27:0>	bit27-0	R/W	**LCD segment enable bit** 0: Disabled 1: Enabled

LCD Control Register 1 (LCD_CON1)

Offset address:10_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											RST	Reserved		SEL	EN

—	bit31-5	—	—
RST	bit4	W	**LCDC module software reset bit**

			0: Always read as 0 1: Software reset
—	bit3-2	—	—
SEL	bit1	R/W	**LCDC/LEDC driver select bit** 0: Select LCD driver (Disable LED driver) 1: Disable LCD driver (Select LED driver)
EN	bit0	R/W	**LCDC module enable bit** 0: Disabled 1: Enabled

Note: Following a LCDC software reset, the LCDC module will be disabled (EN=0).

LCD Data Register 0(LCD_D0)

Offset address:20_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D0<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D0<15:0>															

D0<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LCD Data Register 1 (LCD_D1)

Offset address:24_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D1<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D1<15:0>															

D1<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LCD Data Register 2 (LCD_D2)

Offset address:28_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D2<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D2<15:0>															

D2<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LCD Data Register 3 (LCD_D3)

Offset address:$2C_H$

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D3<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D3<15:0>															

D3<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LCD Data Register 4 (LCD_D4)

Offset address:30_H

Reset value:00000000_00000000_00000000_00000000_B

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D4<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D4<15:0>															

D4<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LCD Data Register 5 (LCD_D5)

Offset address:34_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D5<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D5<15:0>															

D5<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LCD Data Register 6 (LCD_D6)

Offset address:38_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D6<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D6<15:0>															

D6<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

5. 9 LED Display Controller (LEDC)

5. 9. 1 Overview

- ◇ 1~ 8 pieces of 8-segment common cathode LED
- ◇ 3 clock source options

5. 9. 2 Block Diagram

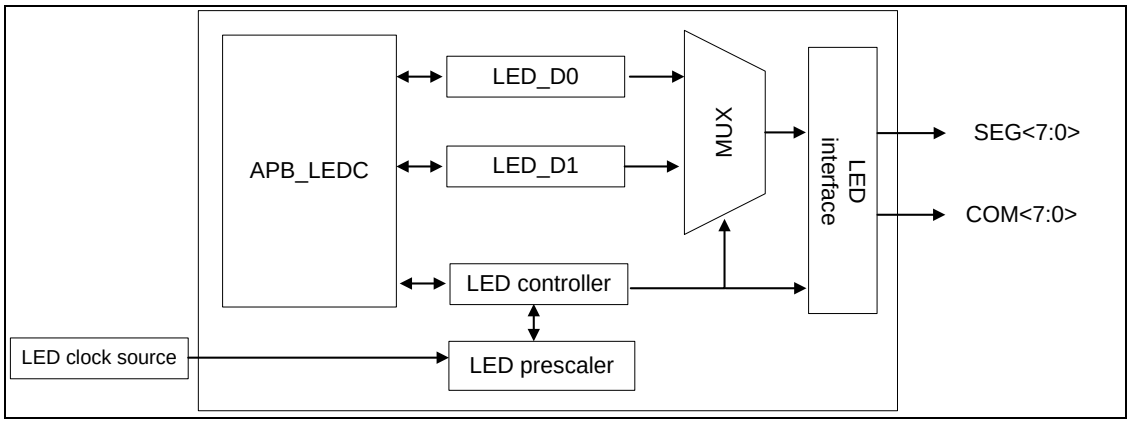


Figure 5-55 LEDC Driver Module Block Diagram

5. 9. 3 LEDC Basic Settings

Configure the LED driver module in the following procedures.

1) Configure the SEL of the LED_CON1 register to select the LED driver.
2) Configure the LED operating clock.
3) Configure the MUX bits to select the number of commons of the LED
4) Configure the GPIO_PAFUNC/ GPIO_PBFUNC as normal I/Os, and LED ports as outputs; set the COM to have high current drive; enable the common and segment function of the ports with COM<7:0> and SEG<7:0> of the LED_SEL register.
5) Initialize the LEDC data register LED_D0~LED_D1.
6) Enable the LEDC driver module.

5. 9. 4 LEDC Pixel Mapping Table

SEG7	SEG6	SEG5	SEG4	SEG3	SEG2	SEG1	SEG0	
D1 [31]	D1 [30]	D1 [29]	D1 [28]	D1 [27]	D1 [26]	D1 [25]	D1 [24]	COM7
D1 [23]	D1 [22]	D1 [21]	D1 [20]	D1 [19]	D1 [18]	D1 [17]	D1 [16]	COM6
D1 [15]	D1 [14]	D1 [13]	D1 [12]	D1 [11]	D1 [10]	D1 [9]	D1 [8]	COM5
D1 [7]	D1 [6]	D1 [5]	D1 [4]	D1 [3]	D1 [2]	D1 [1]	D1 [0]	COM4
D0 [31]	D0 [30]	D0 [29]	D0 [28]	D0 [27]	D0 [26]	D0 [25]	D0 [24]	COM3
D0 [23]	D0 [22]	D0 [21]	D0 [20]	D0 [19]	D0 [18]	D0 [17]	D0 [16]	COM2
D0 [15]	D0 [14]	D0 [13]	D0 [12]	D0 [11]	D0 [10]	D0 [9]	D0 [8]	COM1
D0 [7]	D0 [6]	D0 [5]	D0 [4]	D0 [3]	D0 [2]	D0[1]	D0[0]	COM0

Table 5-7 LEDC Pixel Mapping Table

5. 9. 5 LEDC Clock Sources

The LED module has 3 clock source options.

- ◇ LRC clock divided by 4: active during sleep
- ◇ LOSC clock divided by 4: active during sleep
- ◇ PCLK clock divided by 4096: inactive during sleep

The PRS<5:0> bits of the LED_CON0 register select the prescaler ratio of the LED operating clock.

5. 9. 6 LEDC Operating Diagram

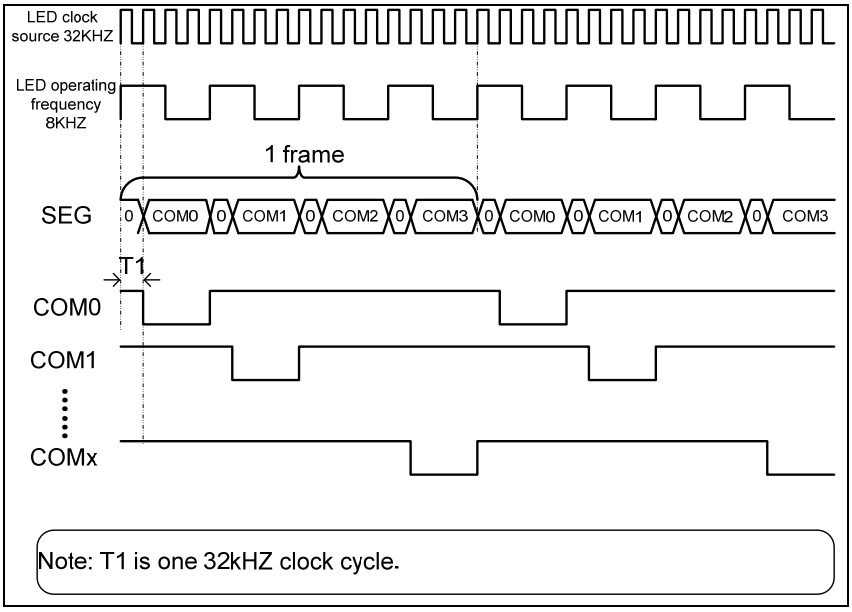


Figure 5-56 LEDC Operating Diagram

5. 9. 7 Special Function Registers

LED Control Register 0 (LED_CON0)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved										PRS<5:0>					

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved								CLK_SEL<1:0>		Reserved			MUX<2:0>		

—	bit31-22	—	—
PRS<5:0>	bit21-16	R/W	**LEDC clock source prescaler ratio select bits** 000000 = 1:1 000001 = 1:2 000010 = 1:3 … …

			111111 = 1:64
—	bit15-8	—	—
CLK_SEL<1:0>	bit7-6	R/W	**LEDC clock source select bits** 00: LRC clock divided by 4 (internal 32KHz clock) 01: LOSC clock divided by 4 (external 32KHz clock) 10 :PCLK clock divided by 4096 11: Reserved
—	bit5-3	—	—
MUX<2:0>	bit2-0	R/W	**COM multiplex select bits** 000: COM0 multiplex 001: COM0~COM1 multiplex 010: COM0~COM2 multiplex 011: COM0~COM3 multiplex 100: COM0~COM4 multiplex 101: COM0~COM5 multiplex 110: COM0~COM6 multiplex 111: COM0~COM7 multiplex

LED Segment Enable Register (LED_SEL)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SEG<15:8>								COM<7:0>							

—	bit31-16	—	—
SEG<7:0>	bit15-8	R/W	**LED segment enable bits** 0: Disabled 1: Enabled
COM<7:0>	bit7-0	R/W	**LED common enable bits** 0: Disabled 1: Enabled

LED Control Register 1 (LED_CON1)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved											RST	Reserved		SEL	EN

—	bit31-5	—	—
RST	bit4	W	**LEDC driver module software reset bit** 0: Always read as 0 1: Software reset
—	bit3-2	—	—
SEL	bit1	R/W	**LEDC/LCDC driver select bit** 0: Disable LEDC driver (select LCDC driver) 1: Select LEDC driver (disable LCDC driver)
EN	bit0	R/W	**LEDC drive module enable bit** 0: Disabled 1: Enabled

LED Data Register 0(LED_D0)

Offset address:20_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D0<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D0<15:0>															

D0<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

LED Data Register 1 (LED_D1)

Offset address:24_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
D1<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
D1<15:0>															

D1<31:0>	bit31-0	R/W	**Pixel on bit** 0: Pixel off (clear) 1: Pixel on (dark)

5. 10 Watchdog Timer (WDT)

5. 10. 1 Overview

When CFG_WDTEN is 1 of the CFG_WORD1, the hardware watchdog is enabled, and the WDT_LOAD register cannot be configured by software. The WDT starts counting right after power on reset using internal 32KHz LRC clock as the counter clock. The WDT loads the value which is 1/4 of the corresponding value selected by the WDTRL of the CFG_WORD1 register, and counts down. When the counter counts down to 0, the window counter will be increased by 1, and the counter will reload the value before the next arriving clock and continue to count down. When the window counter reaches 2 (the accumulated WDT counter value is half of the WDTRL value), a WDT interrupt will occur. Before the window counter reaches 4 (the accumulated WDT counter value is equal to the WDTRL value), if there has been no feeding during the corresponding feed window, a reset signal will be generated. The WDT_LOCK register is readable and writable, the WDT_INTCLR is write-only, and other WDT associated registers are read-only. The value obtained by reading the WDT_LOAD register is the reload value selected by WDTRL. Reading the WDT_CON register will return 0x0000_000F.

When CFG_WDTEN is 0 of the CFG_WORD1, the hardware watchdog is disabled. However, the WDT still can be enabled by software. And the WDT_LOAD register is also software configurable. When the WDT is enabled by software (WDTEN=1 of the WDT_CON register), the WDT will load 1/4 of the value in the WDT_LOAD register, and starts to count down. When the counter value reaches 0, the window counter will be increased by 1, and will again reload the 1/4 value of the WDT_LOAD register before the next arriving clock and continue to count down. When the window counter reaches 2 (the accumulated WDT counter value is half of the WDT_LOAD value), an WDT interrupt will occur. Before the window counter reaches 4 (the accumulated WDT counter value is equal to the WDT_LOAD value), if there has been no feeding during the corresponding feed window, a reset signal will be generated. Select a counter clock source by configuring the CLKS and write the WDT_LOAD register with a reload value. Read the WDT_VALUE register to obtain the current counter value. When writing the WDT_LOAD register, the WDT_VALUE register will be cleared.

The WDT module supports write-protection. Write WDT_LOCK = 0x1ACCE551 to remove the protection in order to write those protected registers; otherwise those registers cannot be written.

Notes

1. In SWD mode, the WDT needs to be disabled. Otherwise, the WDT will keep counting during debug, resulting in a reset caused by an overflow, which thereby leads to an irregular debugging behavior
2. When deep sleep mode is entered and WDT is enabled, a WDT reset cannot wake up the system. Thus, the WDT interrupt must also be enabled to wake up the system.

5. 10. 2 Special Function Registers

WDT Reload Value Register (WDT_LOAD)

Offset address:00_H

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
LOAD<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
LOAD<15:0>															

LOAD<31:0>	bit31-0	W	**WDT reload value** 0x0000_0001~0xFFFF_FFFF. If the value is 0, the WDT does not count.

WDT Current Value Register (WDT_VALUE)

Offset address:04_H

Reset value:$11111111\_11111111\_11111111\_11111111_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
VALUE<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
VALUE<15:0>															

VALUE<31:0>	bit31-0	R	**WDT current counter value** When read, the WDT returns the current counter value.

WDT Control Register (WDT_CON)

Offset address:08_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved												CLKS	RSTEN	IE	EN

—	bit31-4	—	—
CLKS	bit3	R/W	**WDT counter clock select bit** 0: PCLK 1: LRC clock(about 32KHz)
RSTEN	bit2	R/W	**WDT reset enable bit**

			0: Disabled 1: Enabled. A reset will be generated when the counter value reaches 0.
IE	bit1	R/W	WDT interrupt enable bit 0: Disabled 1: Enabled. The interrupt flag will be set when the counter value reaches 0.
EN	bit0	R/W	WDT enable bit 0: Disabled 1: Enabled

Note: The bits of the WDT_CON are valid only when CFG_WDTEN = 0 of the CFG_WORD1.

WDT Interrupt Clear Register (WDT_INTCLR)

Offset address:$0C_H$

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
INTCLR<31:16>															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
INTCLR<15:0>															

INTCLR<31:0>	bit31-0	W	**WDT interrupt clear bit** Any write operations to the WDT_INTCLR register will clear all WDT interrupt flags, and the counter will reload the value of the WDT_LOAD register and continue to count down.

WDT Interrupt Flag Register (WDT_RIS)

Offset address:10_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

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

—	bit31-1	—	—
WDTIF	bit0	R	**WDT interrupt flag bit** 0: No interrupt occurred 1: An interrupt occurred when the WDT counter reaches 0. Writing the WDT_INTCLR register to clear the WDT interrupt flag.

WDT Lock Register (WDT_LOCK)

Offset address:00_H

Reset value:$00000000\_00000000\_00000000\_00000000_B$

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

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

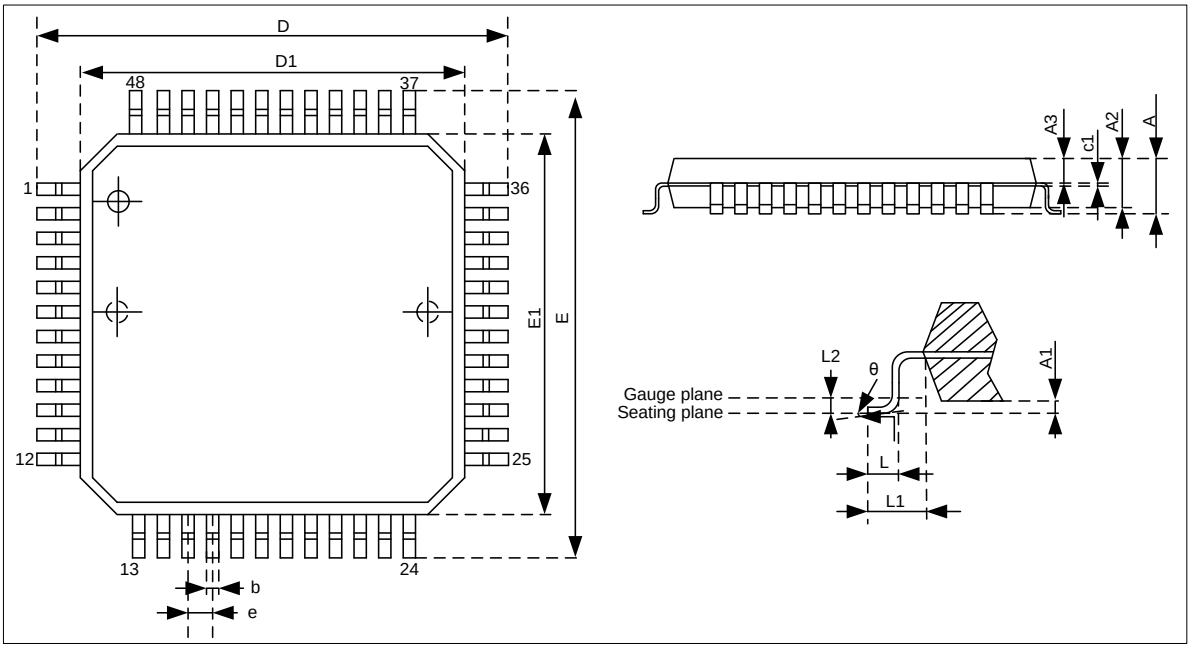
—	bit31-1	W	When writing 0x1ACCE551 to the WDT_LOCK<31:0>, the lock bit will be 0. Otherwise, the lock bit is 1.
LOCK	bit0	R/W	**WDT register lock bit** 0: WDT registers are not locked. 1: WDT registers are locked for protection. Write 0x1ACCE551 to the WDT_LOCK register to unlock the WDT registers. Write other values to the WDT_LOCK register to lock WDT registers for protection.

Notes

1. The WDT_LOCK register has 32 bits and is write-only, and only the LOCK bit is readable. The register must be accessed in word.
2. The WDT_LOCK register locks the WDT_LOAD, WDT_CON and WDT_INTCLR register for protection.

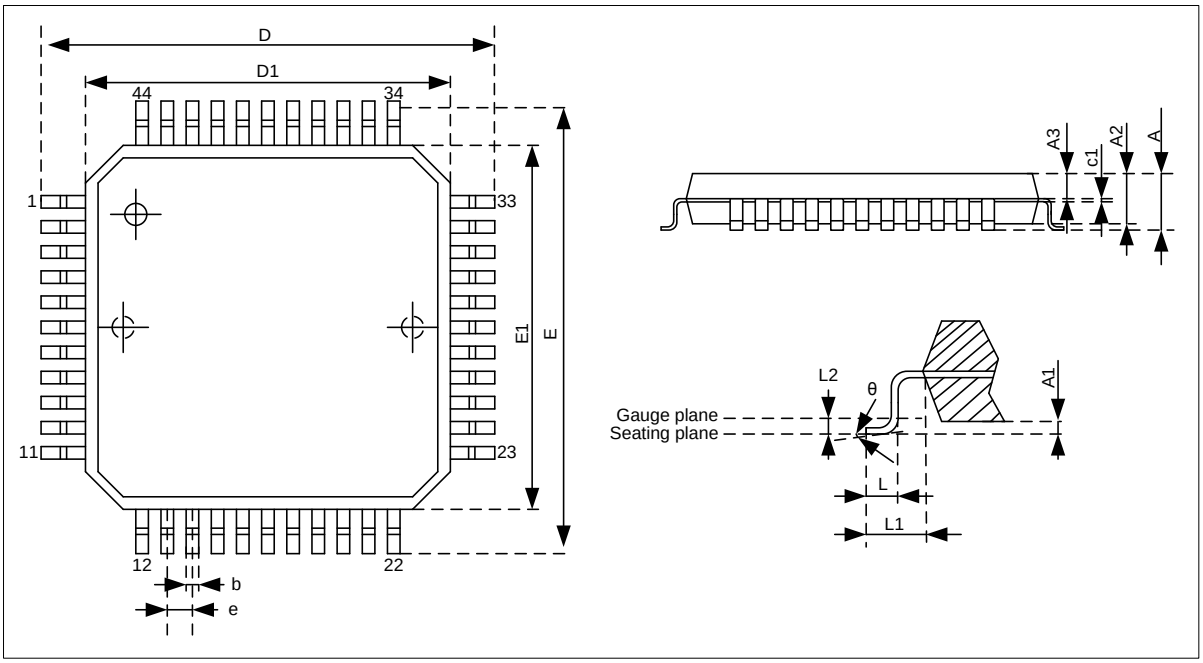
Chapter6 Packaging Information

6. 1 LQFP 48-pin Package Drawing



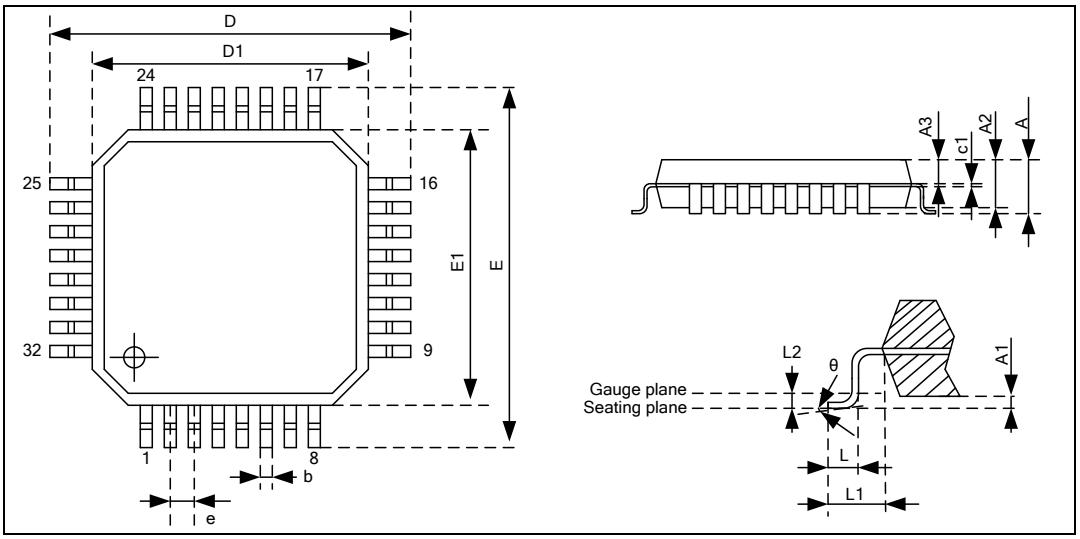
Symbol	（mm）			（inch）		
	MIN	NOM	MAX	MIN	NOM	MAX
A	—	—	1.60	—	—	0.063
A1	0.05	—	0.15	0.002	—	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
c1	0.09	—	0.20	0.003	—	0.008
D	8.80	9.00	9.20	0.346	0.354	0.362
D1	6.90	7.00	7.10	0.271	0.275	0.279
E	8.80	9.00	9.20	0.346	0.354	0.362
E1	6.90	7.00	7.10	0.271	0.275	0.279
b	0.17	0.22	0.27	0.006	0.008	0.011
e	0.40	0.50	0.60	0.016	0.02	0.024
L	0.45	0.60	0.75	0.018	0.024	0.029
L1	0.85	0.95	1.05	0.033	0.037	0.041
L2	—	0.25 BSC	—	—	0.010 BSC	—
θ	0°	3.5°	7°	0°	3.5°	7°

6. 2 LQFP 44-pin Package Drawing



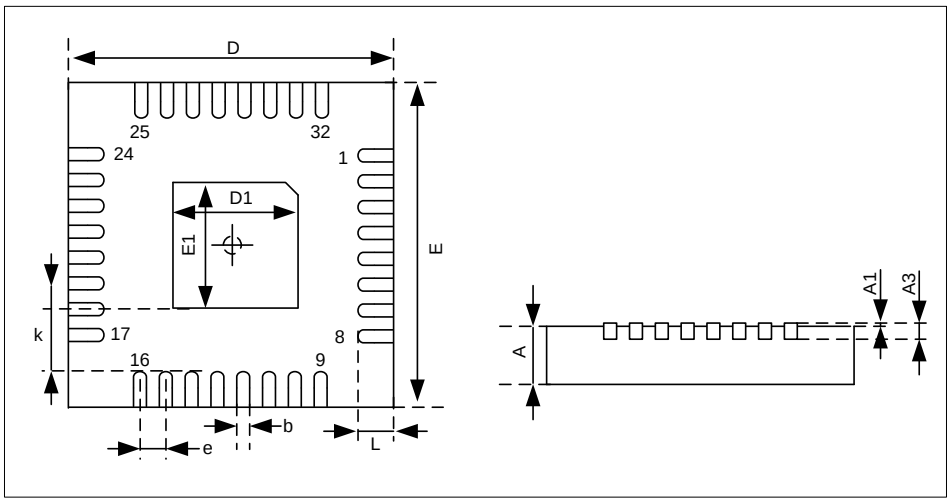
Symbol	（mm）			（inch）		
	MIN	NOM	MAX	MIN	NOM	MAX
A	1.45	1.55	1.65	0.057	0.061	0.065
A1	0.015	—	0.21	0.0005	—	0.0083
A2	1.30	1.40	1.50	0.050	0.055	0.060
c1	—	0.127	—	—	0.005	—
D	11.75	12.52	13.30	0.462	0.493	0.524
D1	9.85	9.95	10.05	0.388	0.392	0.396
E	11.75	12.52	13.30	0.462	0.493	0.524
E1	9.85	9.95	10.05	0.388	0.392	0.396
b	0.25	0.30	0.35	0.009	0.012	0.014
e	—	0.8	—	—	0.032	—
L	0.42	-	0.72	0.016	—	0.029
L1	0.95	1.32	1.70	0.037	0.052	0.067
θ	0	-	10°	0	—	10°

6. 3 LQFP 32-pin Package Drawing



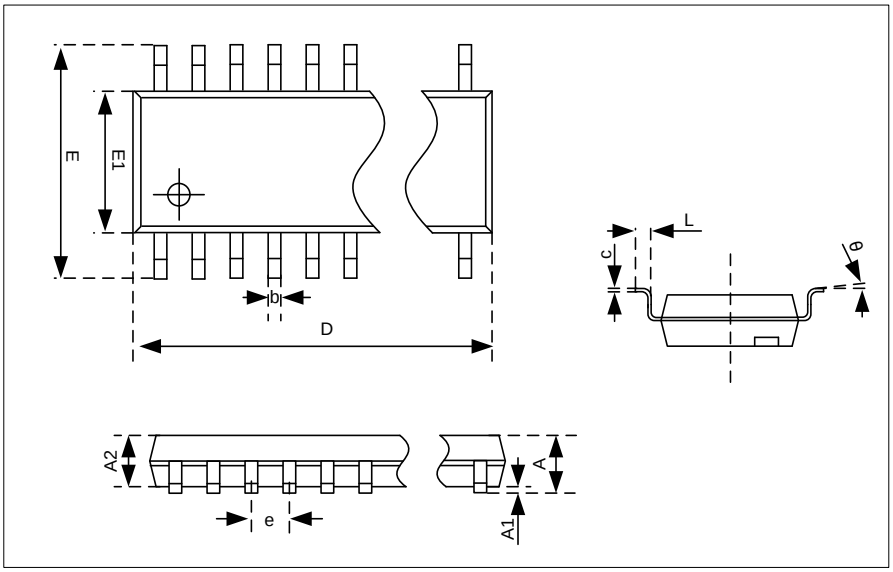
Symbol	（mm）			（inch）		
	MIN	NOM	MAX	MIN	NOM	MAX
A	—	—	1.60	—	—	0.063
A1	0.05	—	0.15	0.002	—	0.006
A2	1.35	1.40	1.45	0.053	0.055	0.057
A3	0.59	0.64	0.69	0.023	0.025	0.027
c1	—	0.127	—	—	0.005	—
D	8.80	9.00	9.20	0.346	0.354	0.362
D1	6.90	7.00	7.10	0.272	0.276	0.280
E	8.80	9.00	9.20	0.346	0.354	0.362
E1	6.90	7.00	7.10	0.272	0.276	0.280
b	0.32	—	0.43	0.013	—	0.017
e	—	0.80 BSC	—	—	0.031	—
L	0.45	0.60	0.75	0.018	0.024	0.030
L1	—	1.00 REF	—	—	0.039 REF	—
L2	—	0.25 BSC	—	—	0.010 BSC	—
θ	0°	3.5°	7°	0°	3.5°	7°

6. 4 QFN 32-pin Package Drawing



Symbol	（mm）			（inch）		
	MIN	NOM	MAX	MIN	NOM	MAX
A	0.700	0.800	0.900	0.028	0.031	0.035
A1	0.000	—	0.050	0.000	—	0.002
A3	0.203REF.			0.008REF.		
D	4.924	—	5.076	0.194	—	0.200
E	4.924	—	5.076	0.194	—	0.200
D1	3.300	—	3.500	0.130	—	0.138
E1	3.300	—	3.500	0.130	—	0.138
k	0.200MIN.			0.008MIN.		
b	0.180	—	0.300	0.007	—	0.012
e	0.500TYP.			0.020TYP.		
L	0.324	—	0.476	0.013	—	0.019

6. 5 SOP 28-pin Package Drawing



Symbol	（mm）			（inch）		
	MIN	NOM	MAX	MIN	NOM	MAX
A	2.30	2.50	2.70	0.090	0.098	0.107
A1	0.10	0.20	0.30	0.003	0.007	0.012
A2	2.10	2.30	2.50	0.082	0.090	0.099
D	17.89	18.09	18.29	0.704	0.712	0.721
E	10.10	10.30	10.50	0.397	0.405	0.414
E1	7.30	7.50	7.70	0.287	0.295	0.304
b	—	0.40	—	—	0.016	—
e	—	1.27	—	—	0.05	—
L	0.75	0.85	0.95	0.029	0.033	0.038
θ	0°	—	8°	0°	—	8°

Appendix1 Cortex-M0 Core

Appendix1. 1 Cortex-M0 Instruction Set

The Cortex-M0 instruction set has 56 basic instructions, 50 of them are 16-bit instructions and 6 of them are 32-bit instructions. Many of the instructions can extend their mnemonics to achieve different functions.

32-bit instructions are BL, DSB, DMB, ISB, MRS and MSR.

Syntax description:

1) < >: any operand inside the bracket can be the operand of the instruction

   Example <Rm｜#imm>: the operand can be register Rm, or immediate #imm

2) { }: any operand or symbol inside the bracket is optional

   Example MOV{S}: the mnemonic can be MOV or MOVS, the latter affects the condition flags

   Example {Rd,}: the destination Rd is optional depending on the instruction used.

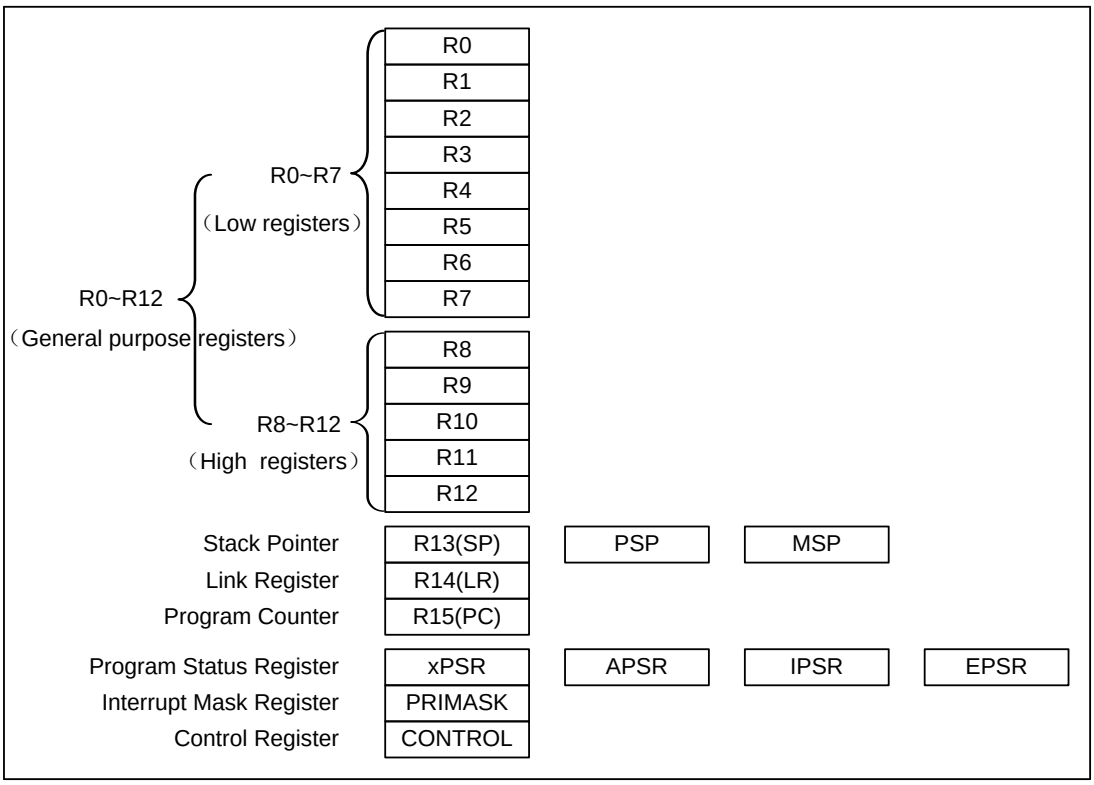
Mnemonic	Operand	Description	Affected Flag Bits
ADR	Rd,Label	Put the address of Label to a register	-
LDR	Rt,Label	Load Rt with word from memory	-
LDR	Rt,[Rn,<Rm｜#imm>]	Load Rt with word from memory	-
LDRB	Rt,[Rn,<Rm｜#imm>]	Load Rt with byte from memory	-
LDRH	Rt,[Rn,<Rm｜#imm>]	Load Rt with half word from memory	-
LDRSB	Rt,[Rn,Rm]	Load Rt with byte from memory (signed extend)	-
LDRSH	Rt,[Rn,Rm]	Load Rt with half word from memory (signed extend)	-
LDM	Rn{!},reglist	Load multiple registers from memory. Rn gets updated by address increment	-
STR	Rt,[Rn,<Rm｜#imm>]	Write word to memory	-
STRB	Rt,[Rn,<Rm｜#imm>]	Write byte to memory	-
STRH	Rt,[Rn,<Rm｜#imm>]	Write half word to memory	-
STM	Rn!,reglist	Store multiple registers to memory. Rn gets updated by address increment	-
PUSH	Reglist	Store register to stack	-
POP	Reglist	Read register to stack	-
MOV{S}	Rd, <Rm｜#imm>	Move register Rd= <Rm | #imm>	N,Z or -
MVNS	Rd,Rm	Logical bitwise NOT. Rd=NOT(Rm)	N,Z
MRS	Rd,spec_reg	Move special register into register, Rd=spec_reg	-

Mnemonic	Operand	Description	Affected Flag Bits
MSR	Spec_reg,Rm	Move register into special register, spec_reg=Rm	N,Z,C,V or -
ADCS	{Rd,}Rn,Rm	Add with carrier	N,Z,C,V
ADD{S}	{Rd,}Rn,<Rm｜#imm>	Add registers	N,Z,C,V or -
RSBS	{Rd,}Rn,#0	Reverse subtract (negative), Rd = 0–Rn	N,Z,C,V
SBCS	{Rd,}Rn,Rm	Subtract with borrow, Rd = Rn–Rm–C	N,Z,C,V
SUB{S}	{Rt,}Rn,<Rm｜#imm>	Subtract	N,Z,C,V or -
ANDS	{Rd,}Rn,Rm	Logic AND,Rd = Rn&Rm	N,Z
ORRS	{Rd,}Rn,Rm	Logical OR, Rd = Rn|Rm	N,Z
EORS	{Rd,}Rn,Rm	Logical Exclusive OR, Rd = Rn^Rm	N,Z
BICS	{Rd,}Rn,Rm	Logical bitwise clear	N,Z
ASRS	{Rd,}Rn,<Rm｜#imm>	Arithmetic shift right	N,Z,C
LSLS	{Rd,}Rn,<Rm｜#imm>	Logical shift left	N,Z,C
LSRS	{Rd,}Rn,<Rm｜#imm>	Logical shift right	N,Z,C
RORS	{Rd,}Rn,Rm	Rotate right	N,Z,C
CMP	{Rn,}<Rm｜#imm>	Compare	N,Z,C,V
CMN	Rn,Rm	Compare negative	N,Z,C,V
MULS	Rd,Rn,Rm	Multiply	N,Z
REV	Rd,Rm	Byte order reverse	-
REV16	Rd,Rm	Byte order reverse with half word	-
REVSH	Rd,Rm	Byte order reverse with lower half word, then signed extend result	-
SXTB	Rd,Rm	Signed extend lowest byte in a word data	-
SXTH	Rd,Rm	Signed extend lower half word in a word data	-
UXTB	Rd,Rm	Extend lowest byte in a word data	-
UXTH	Rd,Rm	Extend lower half word in a word data	-
TST	Rd,Rm	Test (bitwise AND)	N,Z
B{cond}	Label	(Conditional) branch to an address pointed by Label	-
BL	Label	Branch to an address pointed by Label and link	-
BX	Rm	Branch to address in register with exchange	-
BLX	Rm	Branch to address in register and link with exchange	-
CPSID	i	Set PRIMASK.PM=1, disable interrupt	-
CPSIE	i	Clear PRIMASK.PM=0, enable interrupt	-

Mnemonic	Operand	Description	Affected Flag Bits
SVC	#imm	Supervisor call	-
DMB	-	Data memory barrier	-
DSB	-	Data synchronization barrier	-
ISB	-	Instruction synchronization barrier	-
SEV	-	Send event	-
WFE	-	Wait for event	-
WFI	-	Wait for interrupt	-
BKPT	#imm	Software breakpoint	-
NOP	-	No operation	-

Appendix1. 2Cortex-M0 Core Register

Cortex-M0 Core Register



Appendix1. 2. 1 General Register R0~R12

Registers R0~R12 are 32-bit registers for general use.

Appendix1. 2. 2 Stack Pointer Register SP (R13)

MSP and PSP are two stack pointers in Cortex-M0 core, and cannot used at the same time. In thread mode, the stack pointer selection is determined by the SPSEL bit of the CONTROL register. The two stack pointers can be accessed using R13 or SP, as well as using the MRS/MSR instruction.

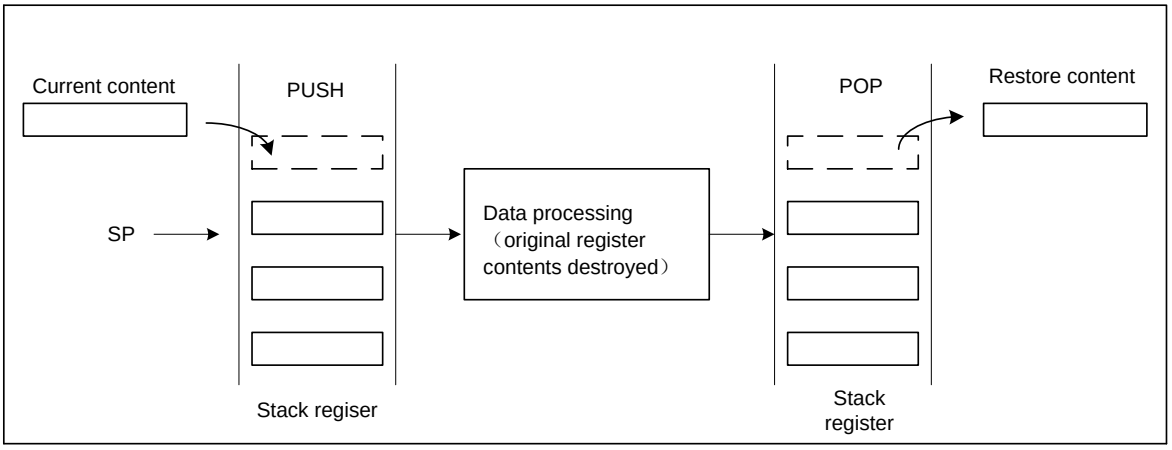
Main stack pointer (MSP): or SP_main, is the default stack pointer after reset, and it is used when running exception handlers/interrupt service routine.

Process stack pointer (PSP): or SP_process, can only be used in thread mode when not handling exceptions or interrupts.

The lowest two bits of the stack are always 0 because the stack is in word-alignment.

In many applications, the system can completely rely on the MSP and it is the default stack pointer for PUSH and POP.

Stack memory is a memory usage mechanism that allows the system memory to be used as temporary data storage that behaves as a first-in, last-out buffer. It is usually used before and after exception handler/ISR, to store and restore the key registers.



Appendix1. 2. 3 Link Register LR (R14)

LR or R14 is the link register, used for storing the return address of a subroutine or function call. For instance, when executing the BL instruction, the address of the next upcoming instruction will be saved to LR.

Appendix1. 2. 4 Program Counter PC (R15)

R15 is the program counter PC. When reading PC, the returned value is the current instruction address plus four; this is caused by the pipeline nature of the design.

Writing to R15 will cause a branch to take place (but unlike a function call, the LR does not get updated), and the new written data is the destination of the branch. Instruction addresses in the Cortex-M0 processor must be aligned to half-word address, which means the LSB of the PC is always 0. However, when attempting to write PC or carry out a branch, the LSB of the PC should be set to 1. This is to indicate that the branch target is a Thumb program region. Otherwise, it can imply trying to switch the processor to ARM state, which is not supported and will cause a fault exception.

Appendix1. 2. 5 Combined Program Status Register xPSR

The combined program status register xPSR provides information about program execution and it consists of three program status registers: application PSR (APSR), interrupt PSR (IPSR) and execution PSR (IPSR).

The three status registers can be accessed individually or at the same time via the MRS/MSR instruction. The register name IAPSR indicates an composition of the IPSR and APSR; the register EAPSR indicates an composition of the EPSR and APSR; the register IEPSR indicates an composition the IPSR and EPSR; and the register XPSR indicates an composition of the three status registers.

	31	30	29	28	27:25	24	23:6	5:0
xPSR	N	Z	C	V	Reserved	T	Reserved	Exception Number
APSR	N	Z	C	V	Reserved			
IPSR	Reserved							Exception Number
EPSR	Reserved					T	Reserved	

The following describes each status bit of the APSR

N: negative flag bit.

When the result is negative, then N=1. Otherwise, N=0

Z: zero flag.

When the result is 0, then Z=1. Otherwise, Z=0. If a compare instruction is executed, then Z =1 if the values are the same.

C: Carry or borrow flag bit

If an addition instruction is executed and an overflow occurred (result ≥232), then C=1. Otherwise, C=0.

If a subtract instruction is executed and no borrow (result≥0), then C =1. Otherwise, C=0.

If a rotate shift instruction is executed, the C flag bit is determined by the bit being shifted to it.

V: Overflow flag bit

For an addition between two negative values, an overflow occurred when the result is positive (bit<31>=0), then V=1. Otherwise, V=0.

For an addition between two positive values, an overflow occurred when the result is negative (bit<31>=1), then V=1. Otherwise, V=0.

For a negative value subtracting a positive value, an overflow occurred when the result is positive (bit<31>=0), then V=1. Otherwise, V=0.

For a positive value subtracting a negative value, an overflow occurred when the result is negative (bit<31>=1) then V=1. Otherwise, V=0.

The IPSR contains the current executing exception number/ISR number. If IPSR<5:0> = 0, it indicates it is now in thread mode, which means no exception/interrupt is being served.

The T-bit of the ESPR indicates that the processor is in the thumb state. On the

Cortex-M0 processor, this bit is normally set to 1 because the Cortex-M0 only supports the thumb state. If this bit is cleared, a hard fault exception will be generated. Reading the EPSR using MSR instruction returns a zero. A write to the EPSR using MSR instruction will be ignored.

Appendix1. 2. 6 Exception /Interrupt Mask Register PRIMASK

The PRIMASK register is a 1-bit-wide interrupt mask register, as shown below.

	31:1	0
PRIMASK	Reserved	PM

When PM=1, it blocks all interrupts apart from the non-maskable interrupt (NMI) and the hard fault exception. When PM = 0, it has no influence on the processor serving the exceptions/interrupts.

The PRIMASK register can be accessed using MSR and MRS instructions, as well as using the CPSID and CPSIE instructions.

Appendix1. 2. 7 CONTROL Register

The CONTROL register can be used to select a stack pointer in thread mode.

	31:2	1	0
CONTROL	Reserved	SPSEL	Reserved

When SPSEL = 0, the MSP (SP_main) is selected as the stack pointer. When SPSEL = 1, the PSP (SP_process) is used.

During running of an exception handler/ISR, only the MSP is used (SPSEL = 0) and the CONTROL register is read as 0. The CONTROL register can only be changed in thread mode or via the exception/ISR entrance and return mechanism. In thread mode, configure the SPSEL to select the stack pointer.

The two stack pointers can be accessed using the MRS/MSR instruction. After changing the SPSEL bit, an ISB instruction is required to execute, which makes sure that the next instructions will only be executed after the new stack point takes effect.

Appendix2 Electrical Characteristics

Appendix2. 1 Parameter Characteristics

Appendix2. 1. 1 Operating Conditions

◆ Absolute Maximum Ratings

Parameter	Symbol	Condition	Range	Unit
Power supply	VDD	VSS=0V	-0.3 ~ 7.5	V
Input voltage	V_{IN}	VSS=0V	-0.3 ~ VDD + 0.3	V
Output voltage	V_{OUT}	VSS=0V	-0.3 ~ VDD + 0.3	V
Max input current on VDD pin	I_{MAXVDD}	VDD=5.0V,25℃	100	mA
Max output current on VSS pin	I_{MAXVSS}	VDD=5.0V,25℃	120	mA
Storage temperature	T_{STG}	—	-55 ~ 125	℃

Notes

1. Stresses above listed under “Absolute Maximum Ratings” may cause permanent damages to the device.
2. The device is only guaranteed to operate normally within the specified operating conditions, as shown below.

◆ Operating Conditions

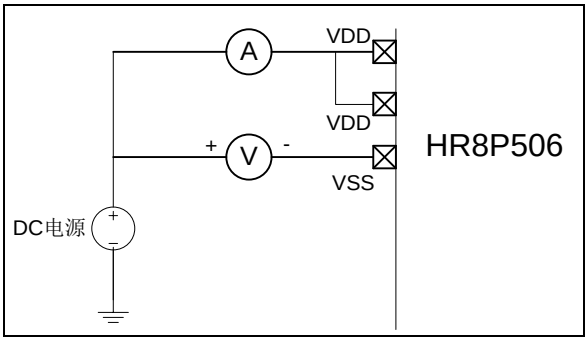
Parameter	Symbol	Condition	Min	Max	Unit
Operating temperature	T_{OPR}	—	-40	85	℃
Operating voltage	VDD	—	2.2	5.5	V
AHB frequency	F_{HCLK}	—	0	48	MHz
APB frequency	F_{PCLK}	—	0	48	MHz

◆ Operating Voltage for Functional Modules

Parameter	Symbol	Operating Temperature	VDD	Remark
ADC operating voltage	V_{ADC}	-40 ~ 85℃	2.2~5.5V	The positive reference voltage is VDD or internal VREFP 1.8V
			2.8~5.5V	The positive reference voltage is the internal VREFP 2.6V
LCD operating voltage	V_{LCD}	-40 ~ 85℃	2.2~5.5V	-

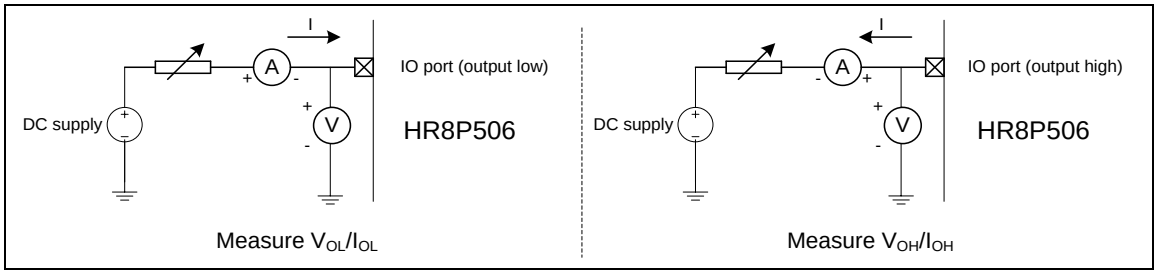
Appendix2. 1. 2 Parameter Measurement

◆ Power consumption measurement



Circuit Connection for Power Consumption Measurement

◆ Output Voltage Measurement on I/O pins



Circuit Connection for Output Voltage Measurement

Appendix2. 1. 3 Power Consumption Characteristics

◆ Power consumption parameters

Parameter	Symbol	Min.	Typ.	Max.	Unit	Operating Conditions
Power supply	VDD	2.2	—	5.5	V	-40℃ ~ 85℃
Static current	I_{DD}	—	300	—	μA	25℃, power on reset, VDD = 5V, all I/Os input low, MRSTN=0
Current in deep sleep mode	I_{PD1}	—	5	—	μA	25℃,VDD = 5V, BOR enabled ,WDT disabled, RTC disabled, all I/Os output fixed, no load
Current in normal sleep mode	I_{PD2}	—	1.2	—	mA	25℃,VDD = 5V, BOR enabled, WDT disabled, RTC disabled, all I/Os output fixed, no load, internal 16MHz RC as system clock
	I_{PD3}	—	1.4	—	mA	25℃,VDD = 5V, BOR enabled, WDT disabled, RTC disabled, all I/Os

Parameter	Symbol	Min.	Typ.	Max.	Unit	Operating Conditions
						output fixed, no load, external 16MHz HS as system clock
Current in normal operating mode	I_{OP1}	—	3.5	—	mA	25℃,VDD = 5V, BOR enabled ,WDT enabled, peripherals running, all I/Os output fixed, no load, clock pin used in 7816 mode as normal I/O and no clock output, ADC using internal VREFP as reference voltage, internal 16MHz RC as system clock
	I_{OP2}	—	4.3	—	mA	25℃,VDD = 5V,BOR enabled, WDT enabled, peripherals running, all I/Os output fixed, no load, clock pin used in 7816 mode as normal I/O and no clock output, ADC using internal VREFP as reference voltage, external 16MHz HS as system clock

◆ Power Consumption Parameters for Functional Modules

Parameter	Symbol	Min.	Typ.	Max.	Unit	Operating Condition
External XTAL 16MHz current	I_{XTAL}	—	1.0	—	mA	25℃,VDD = 5V
Internal high-speed HRC current	I_{HRC}	—	0.2	—	mA	25℃,VDD = 5V
BOR current	I_{BOR}	—	0.3	—	μA	25℃,VDD = 5V
LVD current	I_{LVD}	—	0.3	—	μA	25℃,VDD = 5V
WDT current	I_{WDT}	—	0.2	—	μA	25℃,VDD = 5V
ADC current	I_{ADC1}	—	850	—	μA	25 ℃ ,VDD = 5V,ADC conversion clock 500KHz, internal VREFP as

Parameter	Symbol	Min.	Typ.	Max.	Unit	Operating Condition
						positive reference voltage
	I_{ADC2}	—	350	—	μA	25 ℃ ,VDD = 5V,ADC conversion clock 500KHz, VDD as positive reference voltage
VREFP current	I_{VREFP}	—	500	—	μA	25℃,VDD = 5V
LCDC current	I_{LCDC}	—	25	—	μA	25℃,VDD = 5V,1/4 bias, frame frequency 64Hz, internal bias resistor 60K and 900K (auto-switched), hold time 1/4 COM period for 60K resister, no external LCD screen connected
RTC current	I_{RTC}	—	0.5	—	μA	25℃,VDD = 5V, external 32.768KHz oscillator as RTC clock source for high precision requirement
T16N current	I_{T16N}	—	80	—	μA	25 ℃ ,VDD = 5V, timer mode, counter clock 16MHz
T32N current	I_{T32N}	—	80	—	μA	25 ℃ ,VDD = 5V, timer mode, counter clock 16MHz
UART current	I_{UART}	—	100	—	μA	25℃,VDD = 5V, baud rate 9600bps
EUART current	I_{EUART}	—	100	—	μA	25℃,VDD = 5V, baud rate 9600bps
SPI current	I_{SPI}	—	50	—	μA	25℃,VDD = 5V, master mode, baud rate 1MHz, transmit interval 32 clock cycles
I2C current	I_{I2C}	—	50	—	μA	25℃,VDD = 5V, master mode, baud rate 400KHz, transmit interval 15 clock cycles
FLASH programming	I_{PROG}	—	—	4	mA	-40~85℃,VDD = 5V

Parameter	Symbol	Min.	Typ.	Max.	Unit	Operating Condition
current						
FLASH erase current	I_{ERAS}	—	—	4	mA	

Appendix2. 1. 4 I/O Ports Characteristics

◆ Input Port Characteristics

Operating temperature range:-40℃ ~ 85℃						
Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Input high voltage	V_{IH1}	0.8VDD	—	VDD	V	2.2V≤VDD≤5.5V
Input low voltage	V_{IL1}	VSS	—	0.2VDD	V	
Input high voltage on MRSTN pin	V_{IH}	0.8VDD	—	VDD	V	
Input low voltage on MRSTN pin	V_{IL}	VSS	—	0.2VDD	V	
Hysteresis voltage	V_{HYST}	—	0.8	—	V	VDD = 5.0V
Input leakage current	I_{IL}	—	—	1	μA	VDD = 5.0V (pin at high-impedance)
Weak pull-up resistor	R_{WPU}	35	45	60	kΩ	VDD = 5.0V Vpin = VSS
Weak pull-down resistor	R_{WPD}	25	35	50	kΩ	VDD = 5.0V Vpin = VDD

◆ Output Ports (PA0~PA5, PA14~PA31,PB0~PB13) Characteristics

Operating temperature range:-40℃ ~ 85℃						
Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Output high voltage (normal drive)	I_{OH1}	-5.0	—	—	mA	VDD = 5.0V V_{OH} = 4.6V
Output high voltage (high drive)	I_{OH2}	-8.0	—	—		
Output low voltage (normal drive)	I_{OL1}	6	—	—	mA	VDD = 5.0V V_{OL} = 0.4V
Output low voltage (high drive)	I_{OL2}	10	—	—		

◆ Output Ports (PA6~PA13) Characteristics

Operating temperature range::-40℃ ~ 85℃						
Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Output high voltage (normal drive)	I_{OH1}	-3.5	—	—	mA	VDD = 5.0V V_{OH} = 4.6V
Output high voltage (high drive)	I_{OH2}	-4.5	—	—		
Output low voltage (normal drive)	I_{OL1}	10	—	—	mA	VDD = 5.0V V_{OL} = 0.4V
Output low voltage (high drive)	I_{OL2}	40	—	—		

Appendix2. 1. 5 System Clock Characteristics

◆ System Clock Specification

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
System clock frequency	F_{OSC}	—	—	48M	Hz	2.2V≤VDD≤5.5V
System clock period	T_{OSC}	20.8	—	—	ns	
Machine cycle	T_{inst}	41.6	—	—	ns	
External clock high time and low time	T_{OSL},T_{OSH}	20	—	—	ns	
External clock rising time and falling time	T_{OSR},T_{OSF}	—	—	8	ns	

Appendix2. 1. 6 Functional Module Characteristics

◆ ADC

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Resolution	RES	—	—	12	bit	See notes
Reference voltage range	V_{ADVREF}	1.8	—	VDD	V	
Analog voltage input range	V_{IN}	VSS	—	V_{ADVREF}	V	
Input capacitance	C_{IN}	—	40	—	pF	
Analog channel input resistance (recommended)	R_{IN}	—	—	10k	Ω	

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
AD conversion clock period	T_{AD1}	1	—	—	μs	
	T_{AD2}	0.5	—	—	μs	
AD conversion time (sampling time excluded)	T_{CONV}	—	14	—	T_{AD}	
Differential Non linearity	DNL	—	±1	±2	LSB	-40℃~85℃
Offset	V_{OFFSET}	—	2	4	LSB	

Notes:

1. These parameters are design specification but not tested. The design condition is -40℃~85℃.
2. T_{AD1} is the conversion clock period when the internal VREFP is used as reference voltage. T_{AD2} is the conversion clock period when the VDD or external AVREFP is used as reference voltage.

◆ ADC Conversion Clock Source Selection

Clock Source	System clock operating frequency (Hz) (VREFP = 01or 10, internal VREFP used as positive reference voltage)			
	48M	32M	16M	4M
FPCLK	Not recommended	Not recommended	Not recommended	Not recommended
FPCLK /2	Not recommended	Not recommended	Not recommended	Not recommended
FPCLK /4	Not recommended	Not recommended	Not recommended	T_{ADCLK} = 1us
FPCLK /8	Not recommended	Not recommended	Not recommended	T_{ADCLK} = 2us
FPCLK /16	Not recommended	Not recommended	T_{ADCLK} = 1us	T_{ADCLK} = 4us
FPCLK /32	Not recommended	T_{ADCLK} = 1us	T_{ADCLK} = 2us	T_{ADCLK} = 8us
FPCLK /64	T_{ADCLK} = 1.3us	T_{ADCLK} = 2us	T_{ADCLK} = 4us	T_{ADCLK} = 16us
FPCLK /256	T_{ADCLK} = 5.3us	T_{ADCLK} = 8us	T_{ADCLK} = 16us	T_{ADCLK} = 64us
FLRC	T_{ADCLK} = 31us	T_{ADCLK} = 31us	T_{ADCLK} = 31us	T_{ADCLK} = 31us

Clock Source	System clock operating frequency (Hz) (VREFP=00 or 11, VDD or external AVREFP used as positive reference voltage)			
	48M	32M	16M	4M
FPCLK	Not recommended	Not recommended	Not recommended	Not recommended
FPCLK /2	Not recommended	Not recommended	Not recommended	T_{ADCLK} = 0.5us
FPCLK /4	Not recommended	Not recommended	Not recommended	T_{ADCLK} = 1us
FPCLK /8	Not recommended	Not recommended	T_{ADCLK} = 0.5us	T_{ADCLK} = 2us
FPCLK /16	Not recommended	T_{ADCLK} = 0.5us	T_{ADCLK} = 1us	T_{ADCLK} = 4us
FPCLK /32	T_{ADCLK} = 0.67us	T_{ADCLK} = 1us	T_{ADCLK} = 2us	T_{ADCLK} = 8us
FPCLK /64	T_{ADCLK} = 1.3us	T_{ADCLK} = 2us	T_{ADCLK} = 4us	T_{ADCLK} = 16us
FPCLK /256	T_{ADCLK} = 5.3us	T_{ADCLK} = 8us	T_{ADCLK} = 16us	T_{ADCLK} = 64us
FLRC	T_{ADCLK} = 31us	T_{ADCLK} = 31us	T_{ADCLK} = 31us	T_{ADCLK} = 31us

◆ ADC Internal Reference Voltage Characteristics

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Internal reference voltage VREF	V_{REF1}	1.782	1.8	1.818	V	25℃,VDD=5V
	V_{REF2}	2.574	2.6	2.626		

◆ Internal Clock Source Characteristics

Parameter	Symbol	Min.	Typ.	Max.	Unit	Test Conditions
HRC clock frequency	F_{HRC}	15.84	16	16.16	MHz	25℃, VDD=2.2V~5.5V
		15.68	16	16.32	MHz	-40℃~85℃, VDD=2.2V~5.5V
HRC start-up time	T_{HRC}	—	30	—	us	See note
LRC clock frequency	F_{LRC}	28.8	32	35.2	KHz	25℃, VDD=2.2V~5.5V
		12.8	32	51.2	KHz	-40℃~85℃, VDD=2.2V~5.5V
LRC start-up time	T_{LRC}	—	800	—	us	See note

Note: T_{HRC} and T_{LRC} are design specifications only but not tested. The design condition is -40℃~85℃.

◆ PLL Characteristics

Parameter		Symbol	Min.	Typ.	Max.	Unit	Test Conditions
Clock Source 32KHz	PLL input clock frequency	F_{PLLI1}	25	32	38	KHz	-40~85℃
	PLL multiplier output clock frequency	F_{PLLO1}	25.6	32.768	38.912	MHz	
		F_{PLLO2}	38.4	49.152	58.368		
	PLL lock time	T_{LOCK1}	—	300	500	us	See note
Clock Source 4MHz	PLL input clock frequency	F_{PLLI2}	3.2	4	4.8	MHz	-40~85℃
	PLL multiplier output clock frequency	F_{PLLO3}	25.6	32	38.4	MHz	
		F_{PLLO4}	38.4	48	57.6		
	PLL lock time	T_{LOCK2}	—	150	300	ms	

◆ BOR Characteristics 1 (BOR voltage is configured by CFG_BORV bits of CFG_WORD0 configuration word)

CFG_BORV<1:0>	Min.	Typ.	Max.	Unit	Test Conditions
00	3.6	3.7	3.8	V	-40~85℃
01	2.4	2.5	2.6	V	
10	2.0	2.1	2.2	V	
11	—	—	—	V	

Note: When CFG_BORV<1:0> = 11, the BOR voltage is configured by the BORV<3:0> of the SCU_SOFTCFG. See the particular register for details.

◆ BOR Characteristics 2 (BOR voltage is configured by BORV bits of SCU_SOFTCFG register)

BORV<3:0>	Min.	Typ.	Max.	Unit	Test Conditions
0000	1.65	1.7	2.15	V	-40~85℃
0001	1.9	2.0	2.15	V	
0010	2.0	2.1	2.2	V	
0011	2.1	2.2	2.3	V	
0100	2.2	2.3	2.4	V	
0101	2.3	2.4	2.5	V	
0110	2.4	2.5	2.6	V	
0111	2.5	2.6	2.7	V	
1000	2.7	2.8	2.9	V	
1001	2.9	3.0	3.1	V	
1010	3.0	3.1	3.2	V	
1011	3.2	3.3	3.4	V	
1100	3.5	3.6	3.7	V	
1101	3.6	3.7	3.8	V	
1110	3.9	4.0	4.1	V	
1111	4.2	4.3	4.4	V	

◆ LVD Characteristics

LVD_VS<3:0>		Min.	Typ.	Max.	Unit	Test Conditions
LVDO set to 1 when VDD Drops	0000	1.9	2.0	2.1	V	-40~85℃
	0001	2.0	2.1	2.2	V	
	0010	2.1	2.2	2.3	V	
	0011	2.3	2.4	2.5	V	
	0100	2.5	2.6	2.7	V	

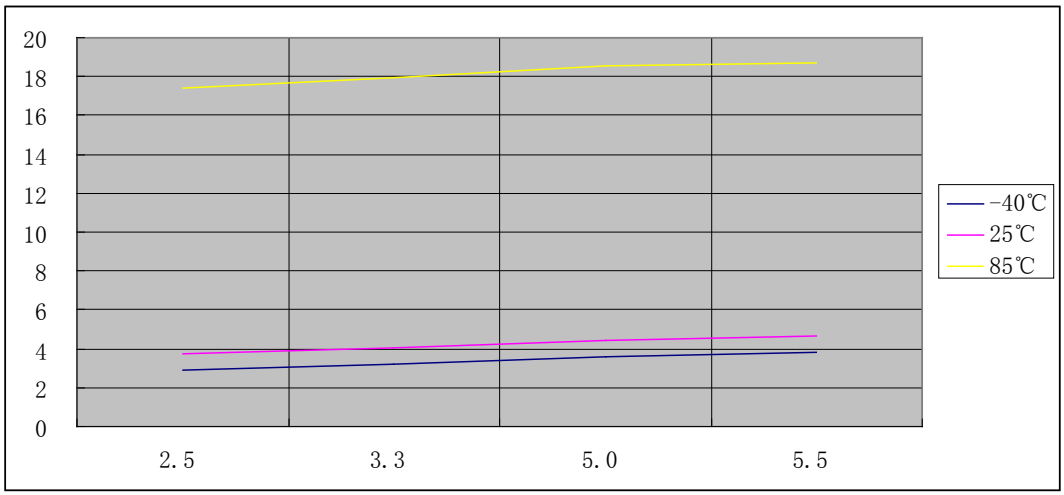
LVD_VS<3:0>		Min.	Typ.	Max.	Unit	Test Conditions
	0101	2.7	2.8	2.9	V	
	0110	2.9	3.0	3.1	V	
	0111	3.5	3.6	3.7	V	
	1000	3.9	4.0	4.1	V	
	1001	4.5	4.6	4.7	V	
	1010	2.2	2.3	2.4	V	
LVD hysteresis		—	—	40	80	-40~85℃

Appendix2. 2Characteristics Graphs

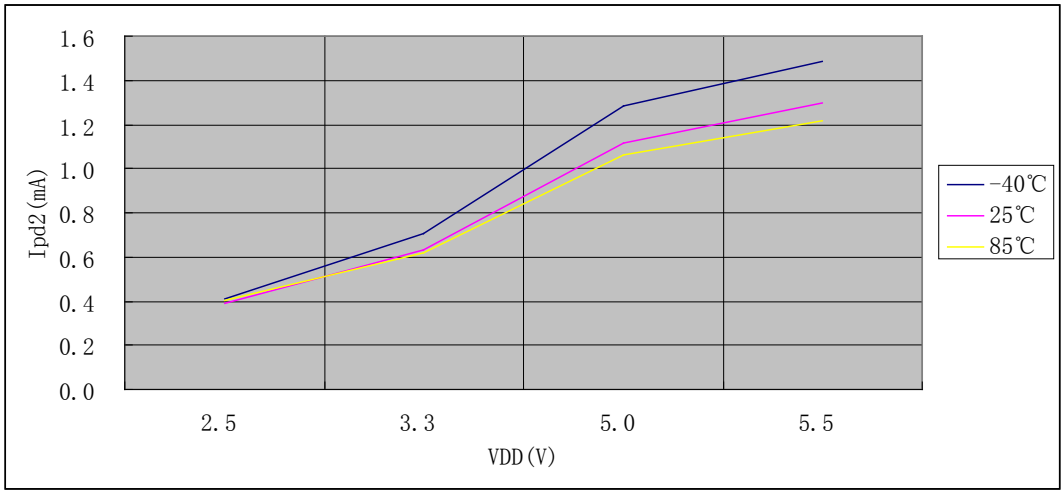
All the graphs below are provided for design reference only and have not been tested for mass production yet. Some data in part of the graphs are beyond specification and the MCU is guaranteed to operate normally only within the specified range.

Appendix2. 2. 1 Power Consumption Characteristics

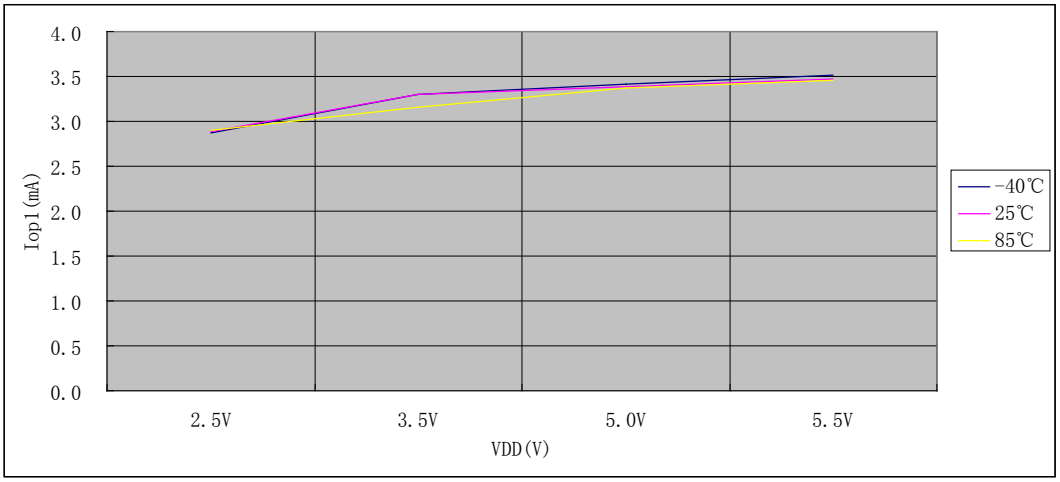
- I_{PD1} vs. VDD (WDT and BOR enabled, RTC disabled, fixed I/O output, no load)



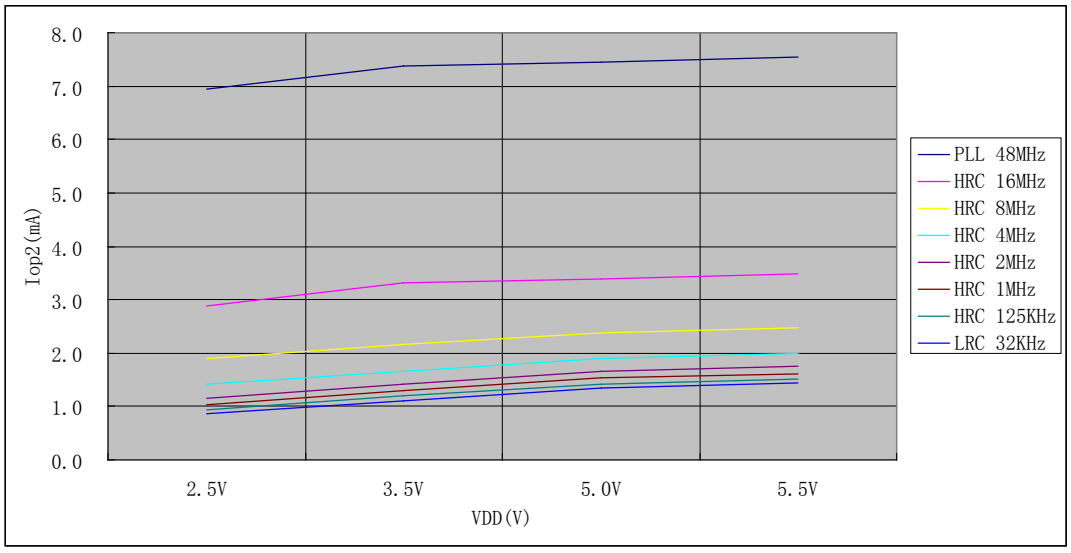
- I_{PD2} vs. VDD (WDT and BOR enabled, RTC disabled, fixed I/O output, no load, internal HRC 16MHz as system clock)



- I_{OP1} vs. VDD (WDT and BOR enabled, peripherals running, fixed I/O output, no load, 7816 clock pin used as normal I/O and no clock output, internal VREFP used as positive reference voltage, internal HRC 16MHz as system clock)



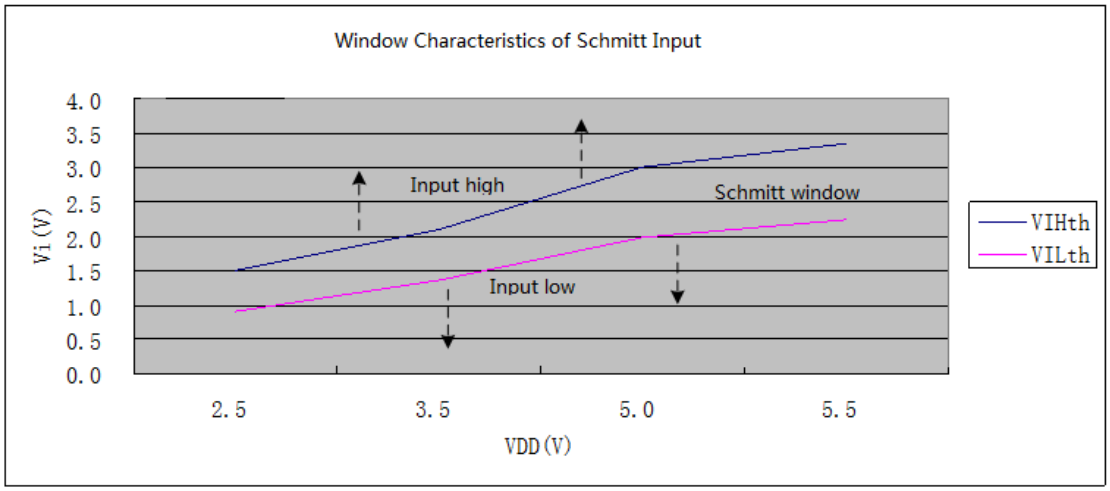
- I_{OP2} vs. VDD (WDT and BOR enabled, peripherals running, fixed I/O output, no load, 7816 clock pin used as normal I/O and no clock output, internal VREFP used as positive reference voltage, room temperature 25℃)



Note: When the PLL 48MHz is used as the system clock, the PLL input clock source is the internal HRC clock.

Appendix2. 2. 2 I/O Port Input Characteristics

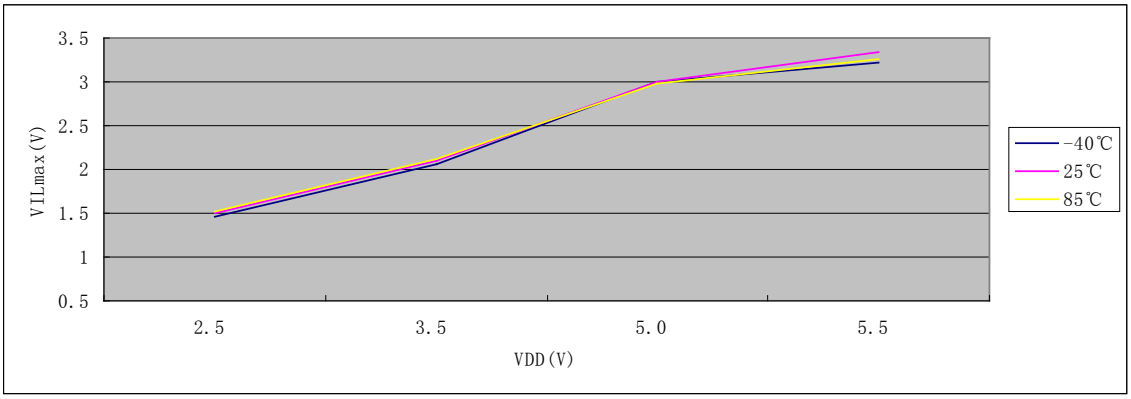
◆ I/O port input characteristic graph (25℃)



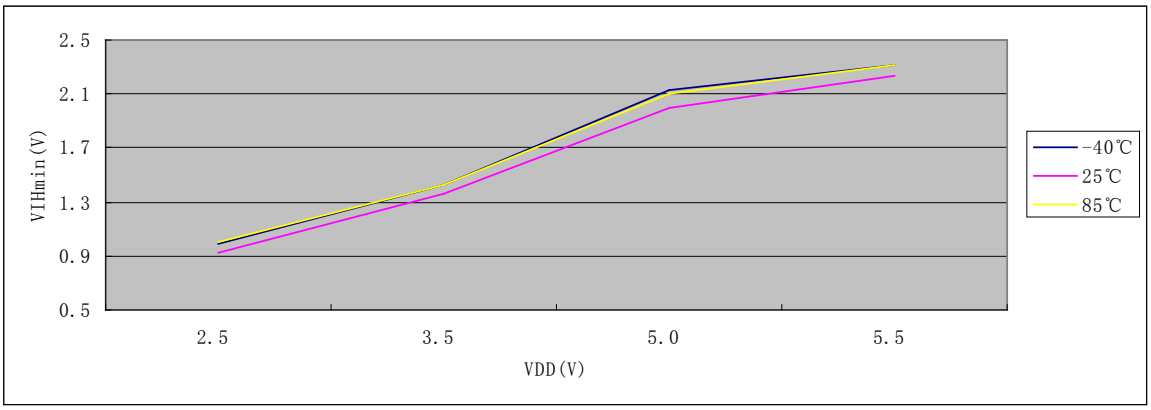
Notes

1. VIHth is the upper threshold voltage of the Schmitt window; an input greater than the upper threshold is considered high.
2. VILth is the lower threshold voltage of the Schmitt window; an input less than the lower threshold is considered low.
3. A Schmitt window has an upper boundary VIHth and lower boundary VILth. Any value inside the window is undermined; it can be high or low.

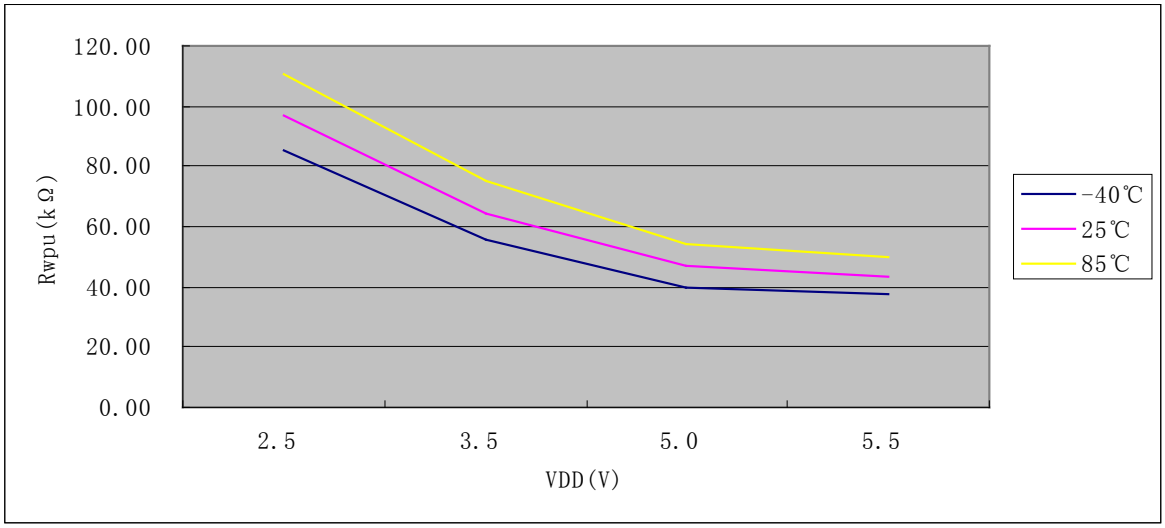
◆ V_{ILMAX} vs. VDD



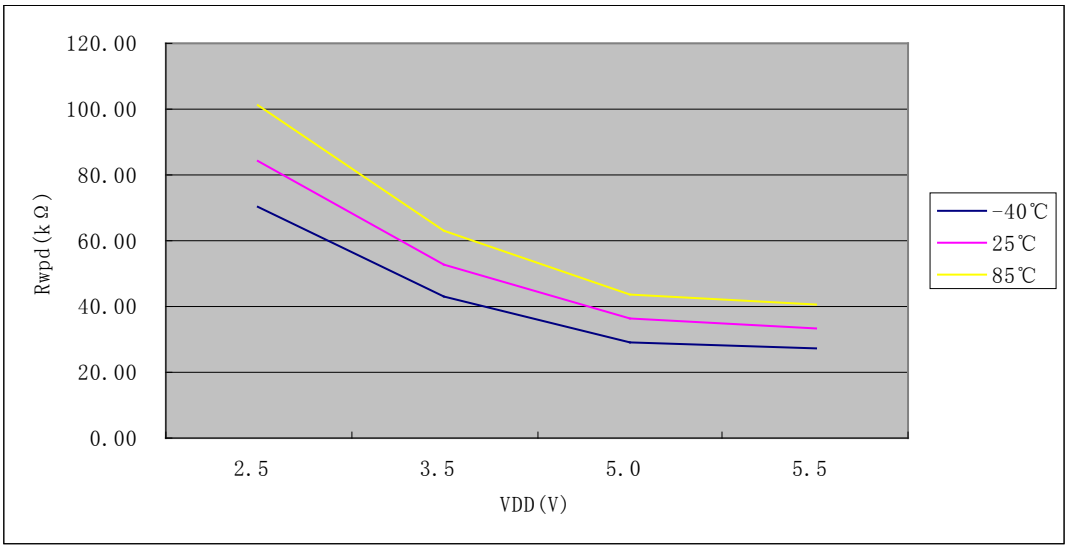
◆ V_{IHMIN} vs. VDD



◆ R_{wpu} vs. VDD

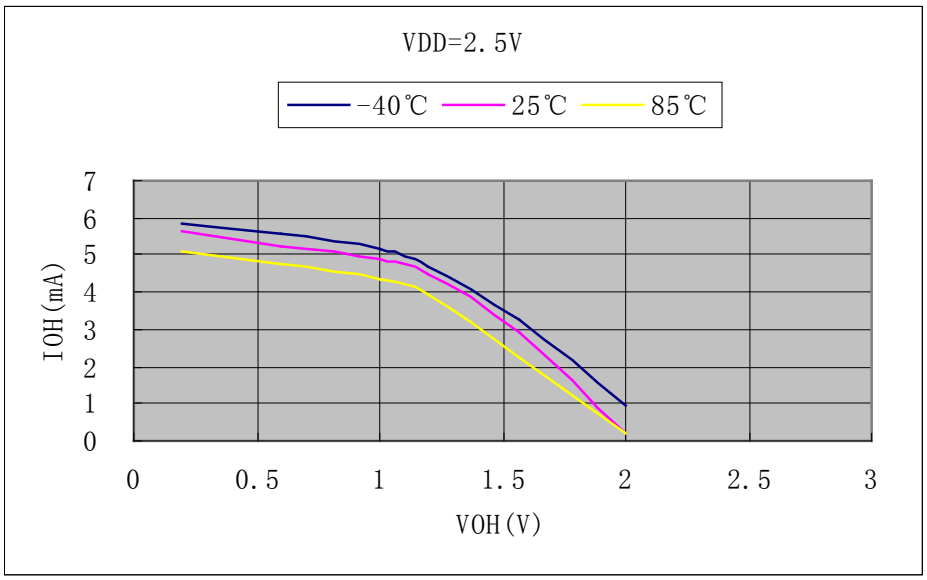


◆ R_{wpd} vs. VDD

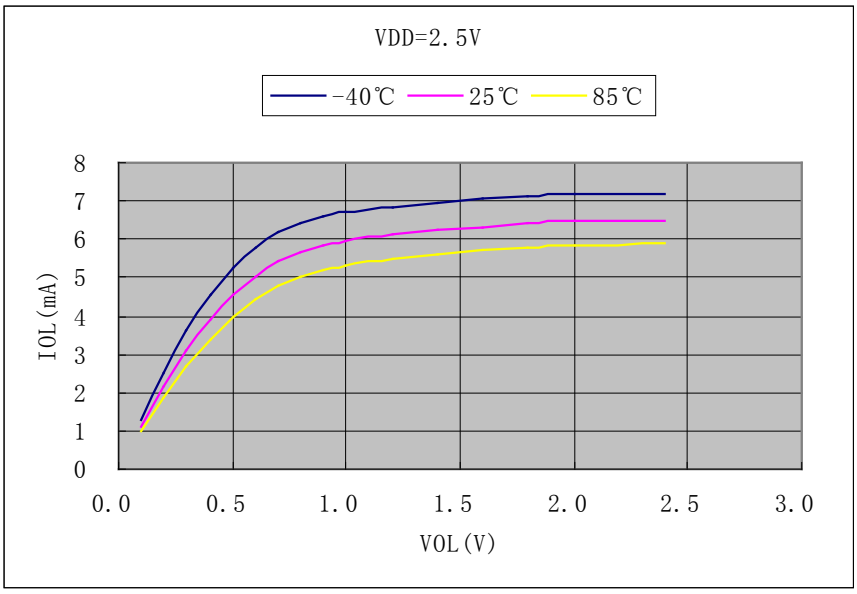


Appendix2. 2. 3 I/O Port Output Characteristics (Normal Drive, PA6~PA13 Excluded)

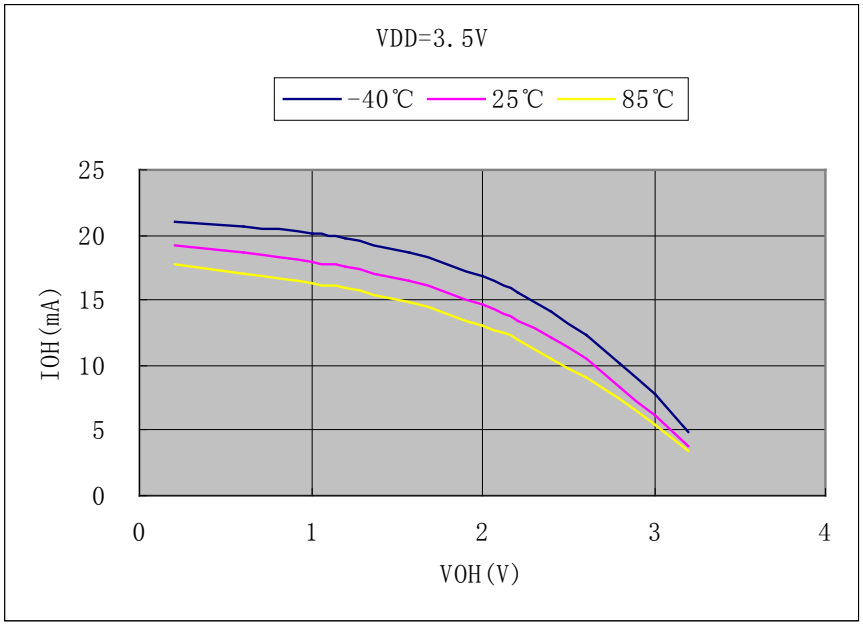
- I_{OH} vs. V_{OH} @ VDD = 2.5V (normal drive)



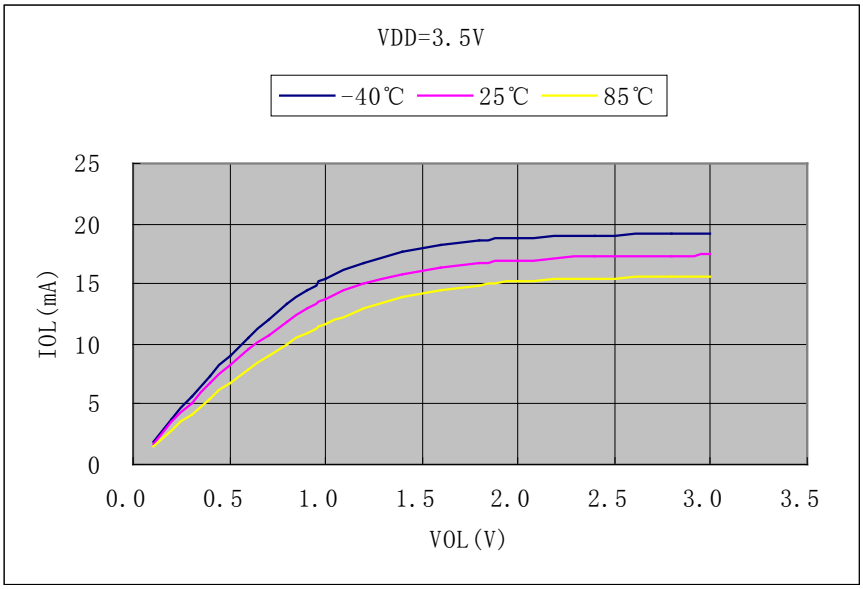
- I_{OL} vs. V_{OL} @ VDD = 2.5V (normal drive)



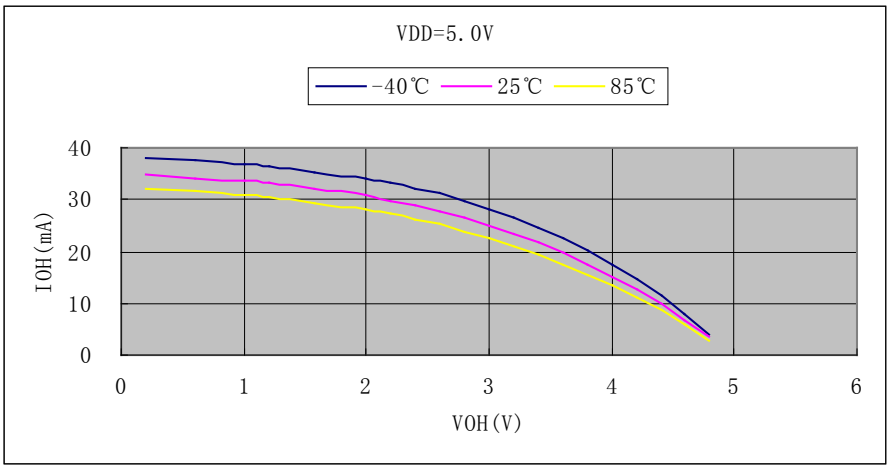
◆ I_{OH} vs. V_{OH} @ VDD = 3.5V (normal drive)



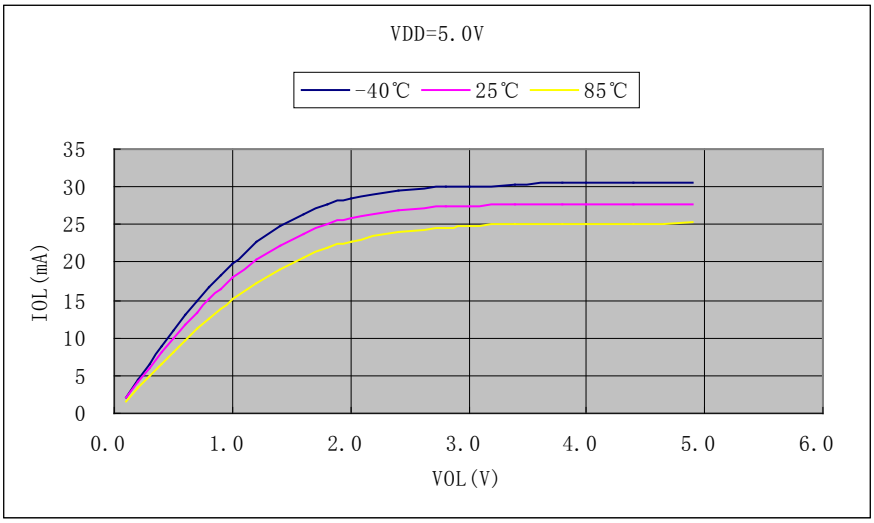
◆ I_{OL} vs. V_{OL} @ VDD = 3.5V (normal drive)



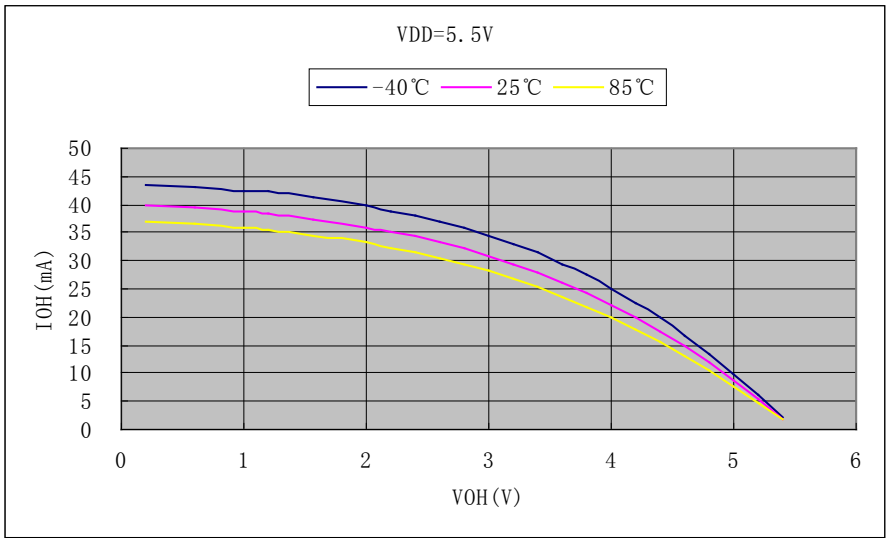
◆ I_{OH} vs. V_{OH} @ VDD = 5.0V (normal drive)



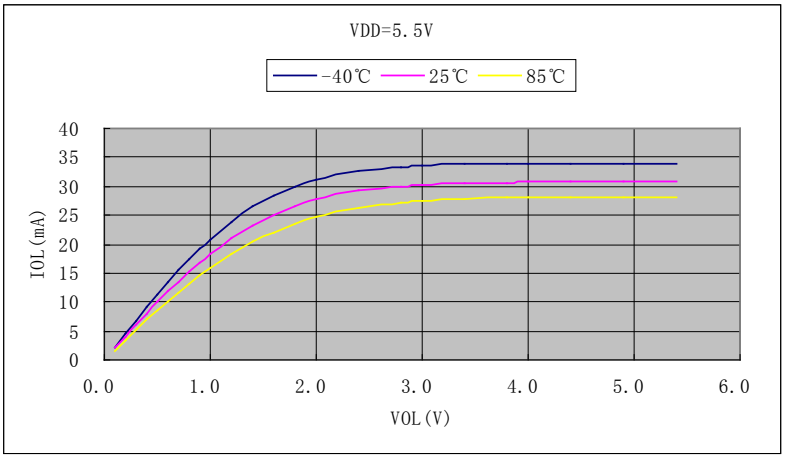
◆ I_{OL} vs. V_{OL} @ VDD = 5.0V (normal drive)



◆ I_{OH} vs. V_{OH} @ VDD = 5.5V (normal drive)

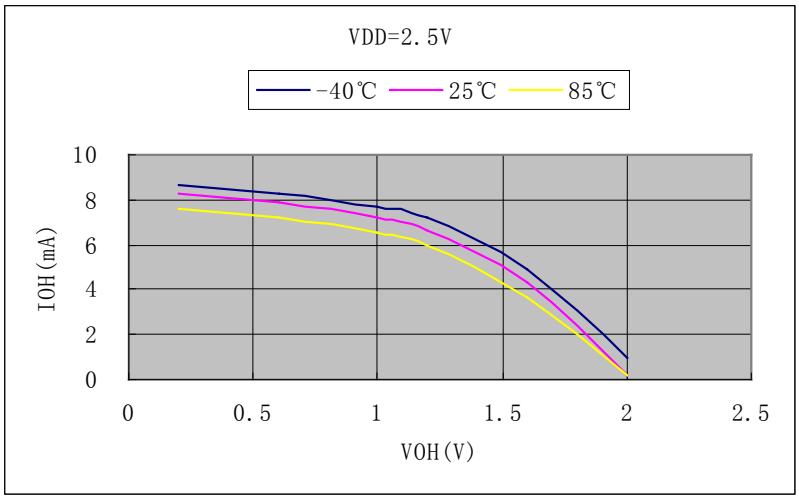


- I_{OL} vs. V_{OL} @ VDD = 5.5V (normal drive)

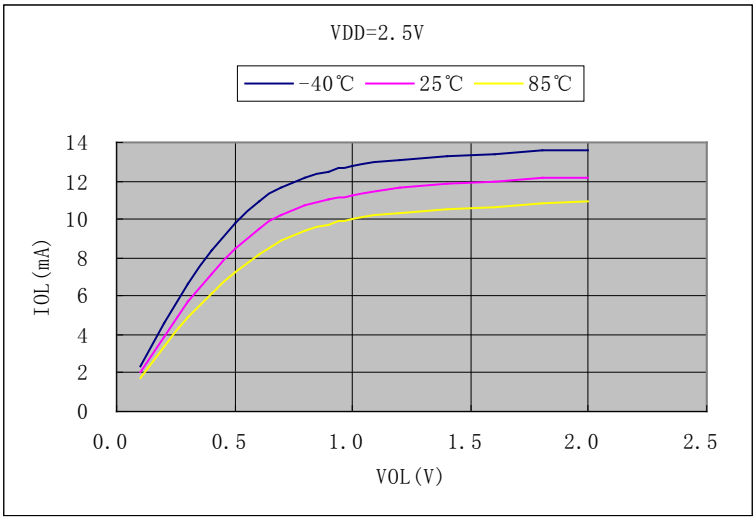


Appendix2. 2. 4 I/O Port Output Characteristics (High Drive, PA6~PA13 Excluded)

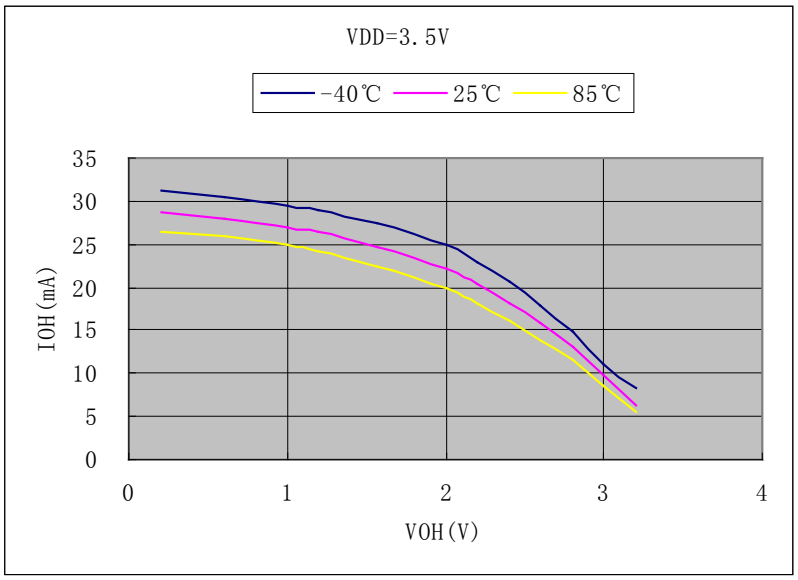
- I_{OH} vs. V_{OH} @ VDD = 2.5V (high drive)



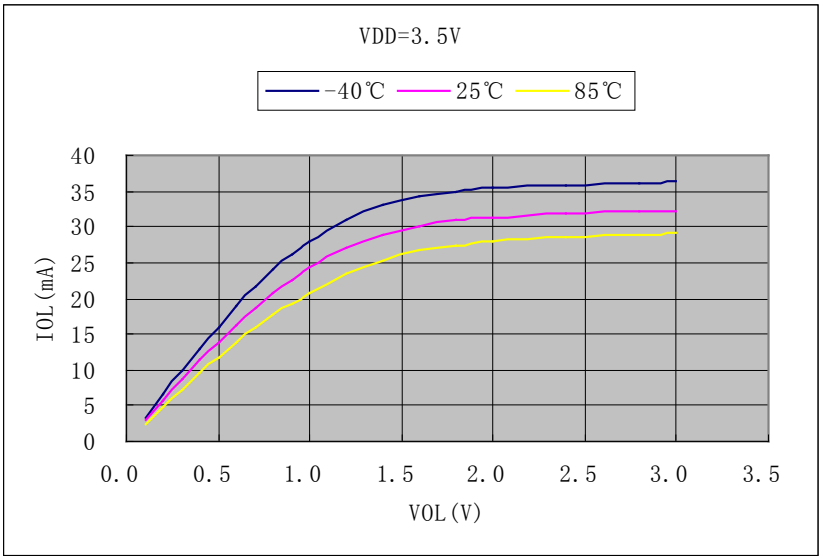
- I_{OL} vs. V_{OL} @ VDD = 2.5V(high drive)



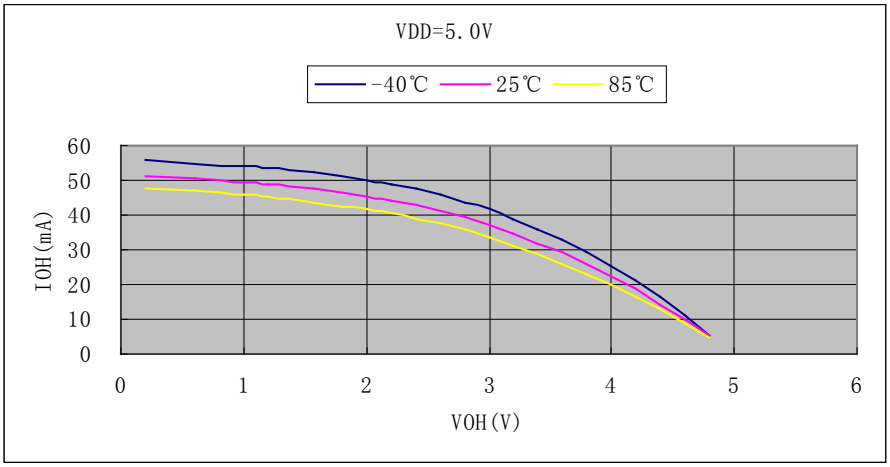
- I_{OH} vs. V_{OH} @ VDD = 3.5V (high drive)



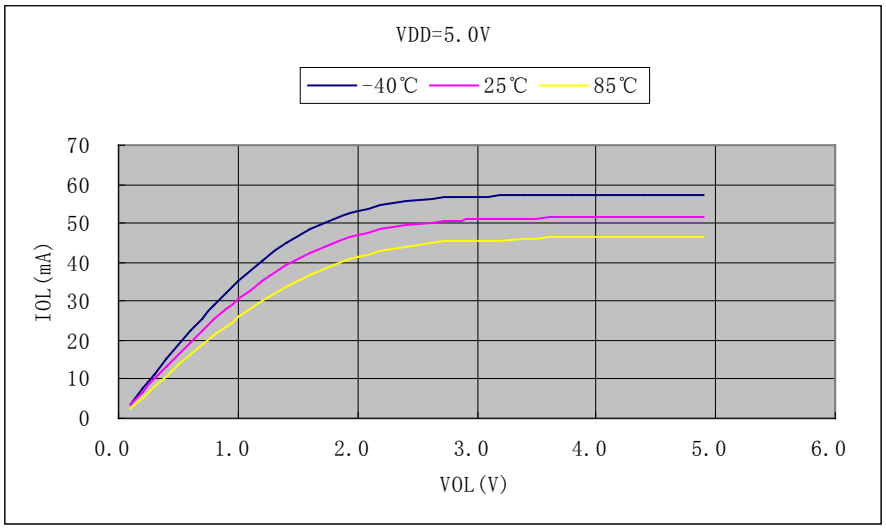
- I_{OL} vs. V_{OL} @ VDD = 3.5V (high drive)



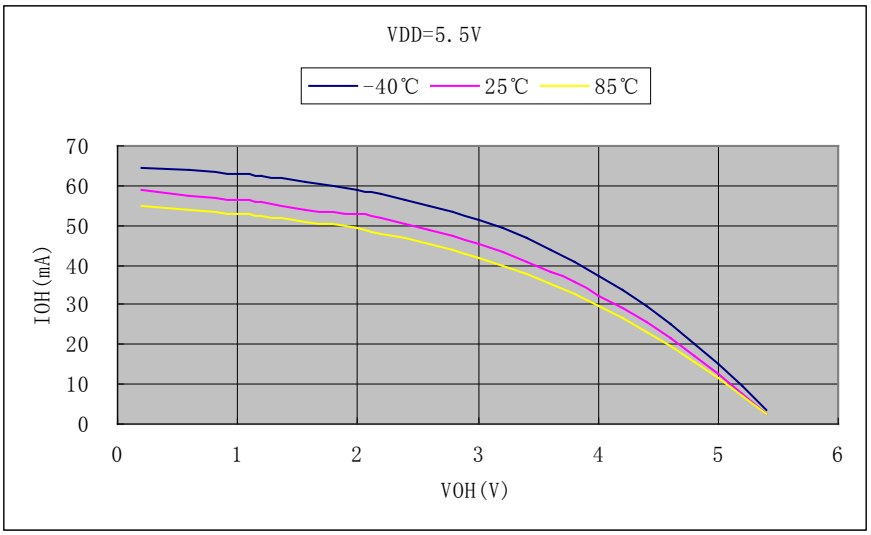
- I_{OH} vs. V_{OH} @ VDD = 5.0V (high drive)



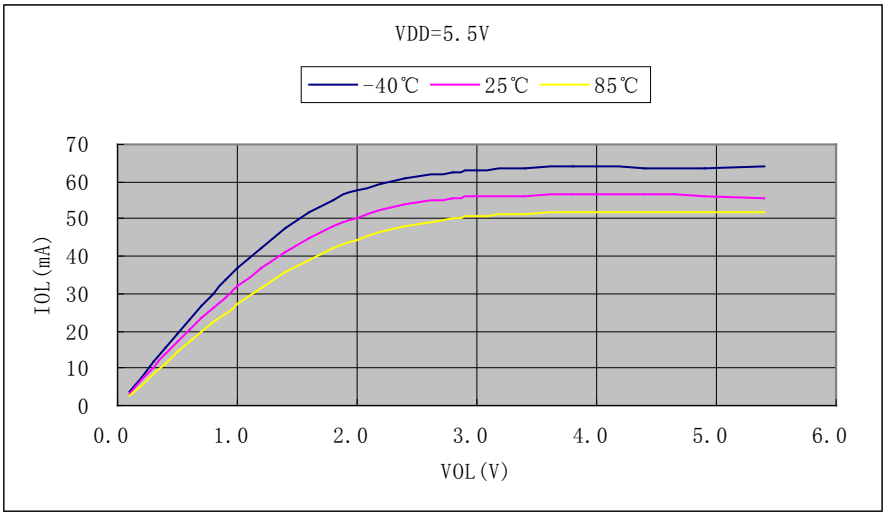
- I_{OL} vs. V_{OL} @ VDD = 5.0V (high drive)



- I_{OH} vs. V_{OH} @ VDD = 5.5V (high drive)

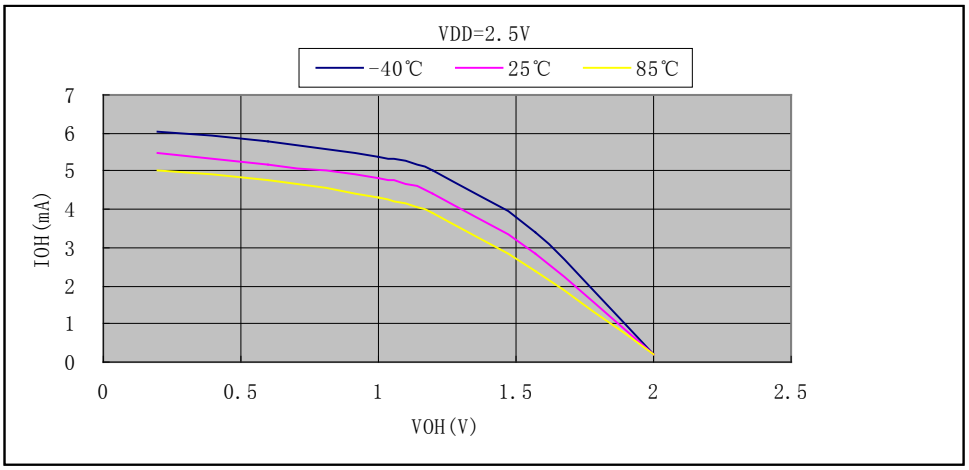


- I_{OL} vs. V_{OL} @ VDD = 5.5V (high drive)

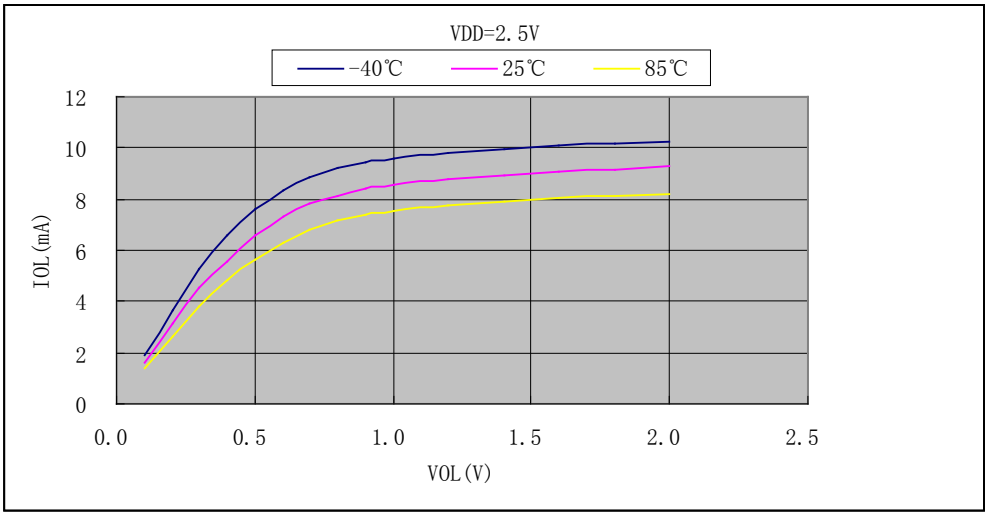


Appendix2. 2. 5 I/O Port Output Characteristics (Normal Drive, PA6~PA13)

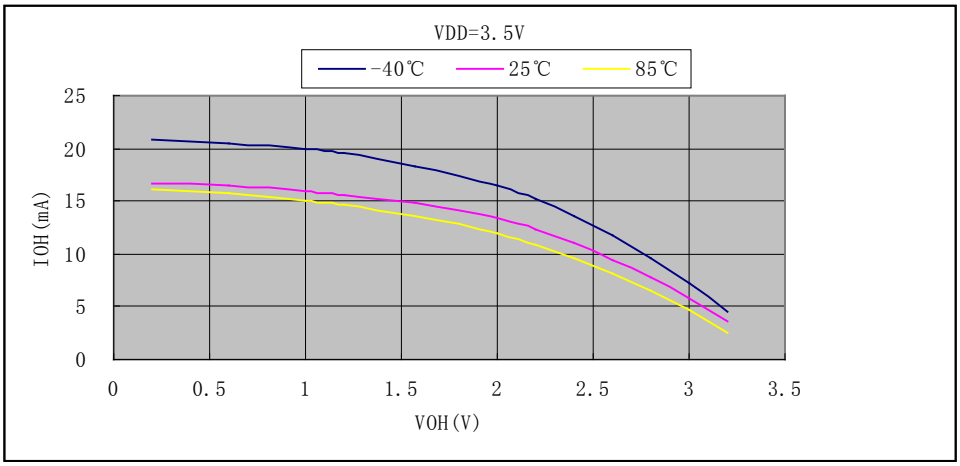
- I_{OH} vs. V_{OH} @ VDD = 2.5V (normal drive)



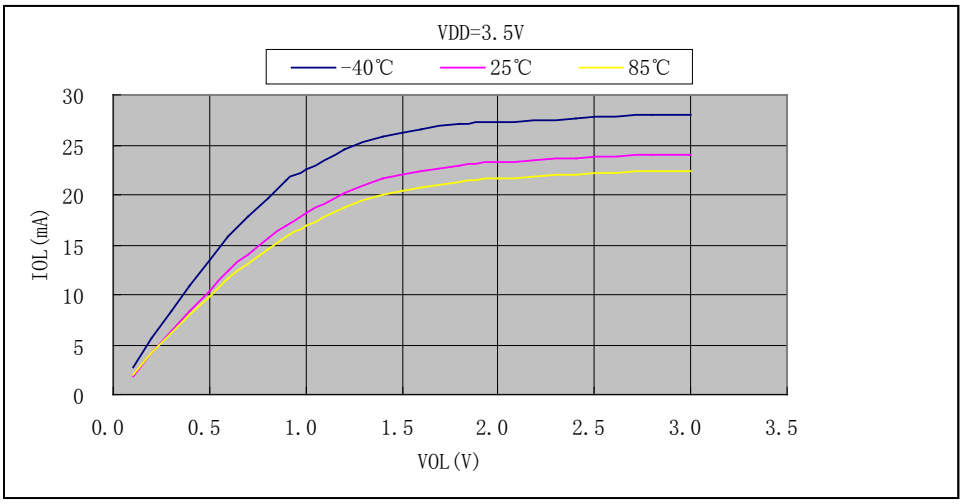
- I_{OL} vs. V_{OL} @ VDD = 2.5V (normal drive)



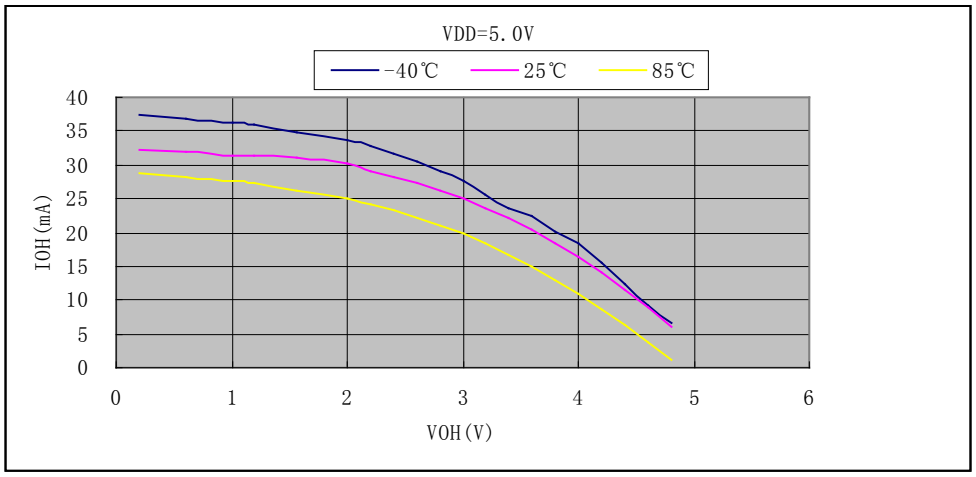
- I_{OH} vs. V_{OH} @ VDD = 3.5V (normal drive)



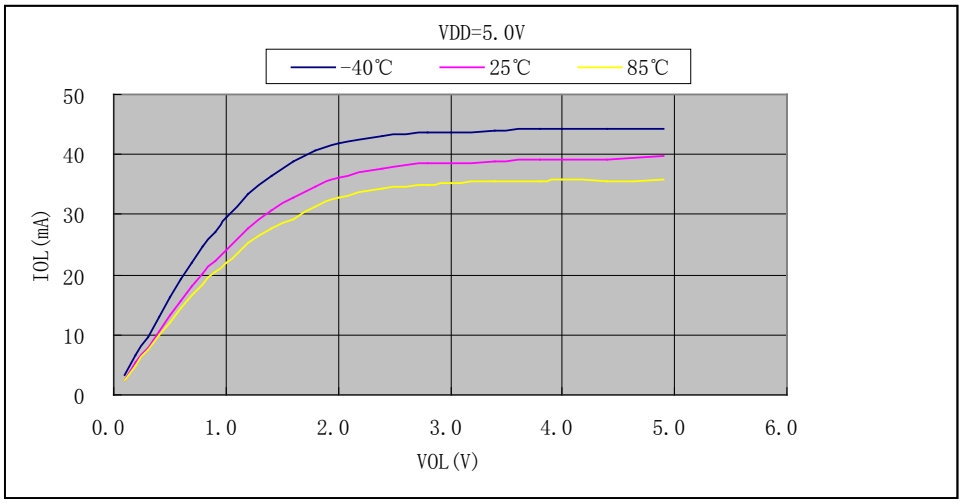
- I_{OL} vs. V_{OL} @ VDD = 3.5V (normal drive)



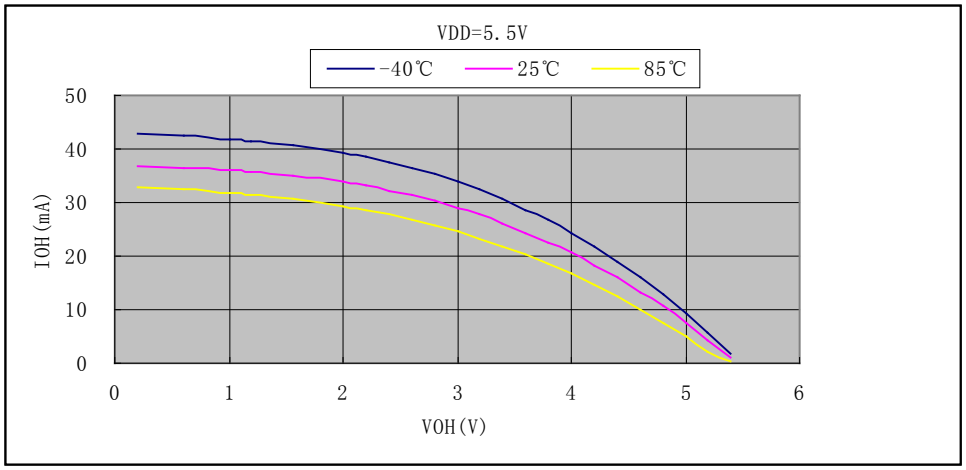
- I_{OH} vs. V_{OH} @ VDD = 5.0V (normal drive)



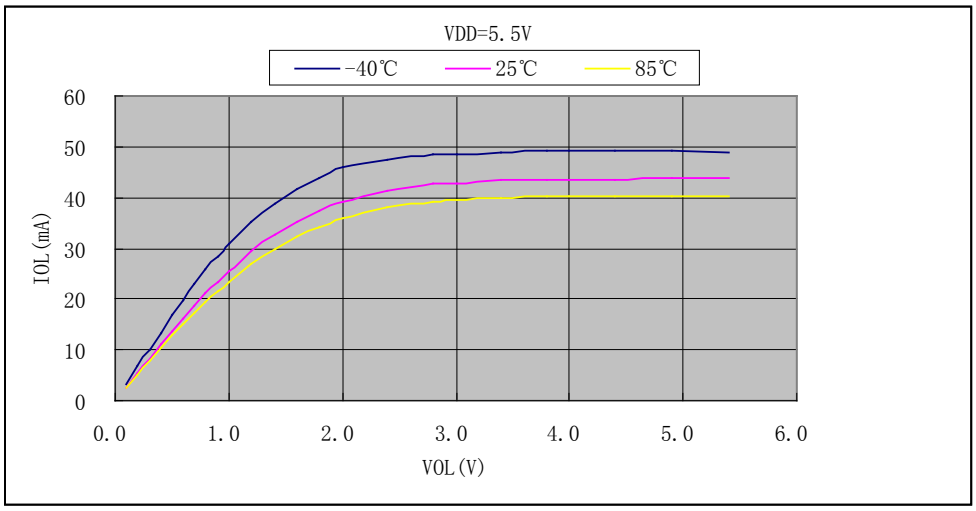
- I_{OL} vs. V_{OL} @ VDD = 5.0V (normal drive)



- I_{OH} vs. V_{OH} @ VDD = 5.5V (normal drive)

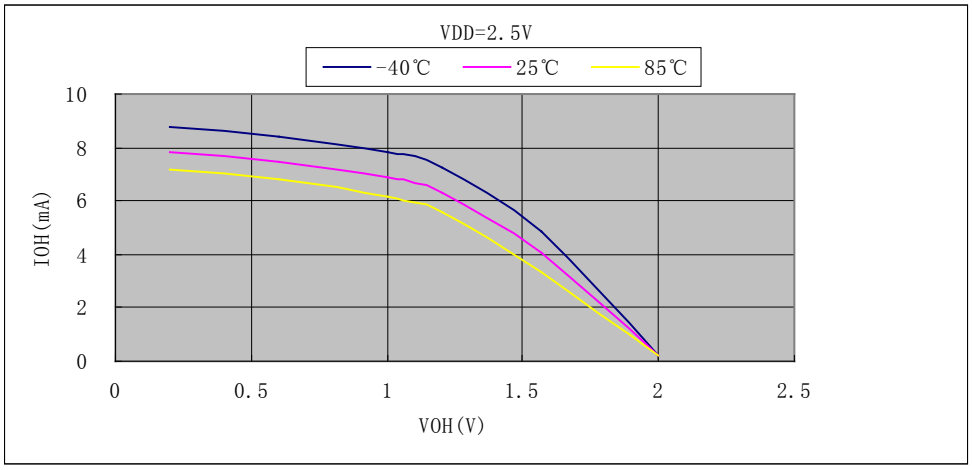


- I_{OL} vs. V_{OL} @ VDD = 5.5V (normal drive)

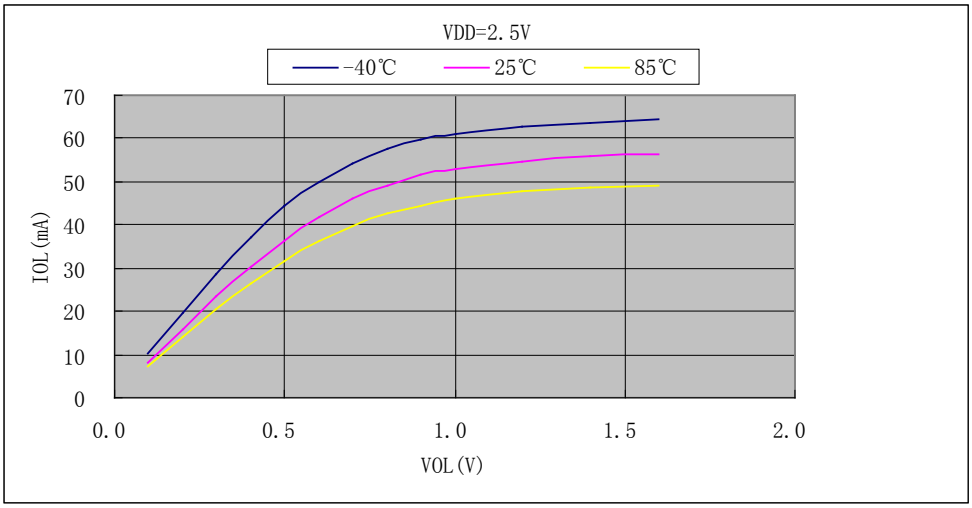


Appendix2. 2. 6 I/O Port Output Characteristics (High Drive, PA6~PA13)

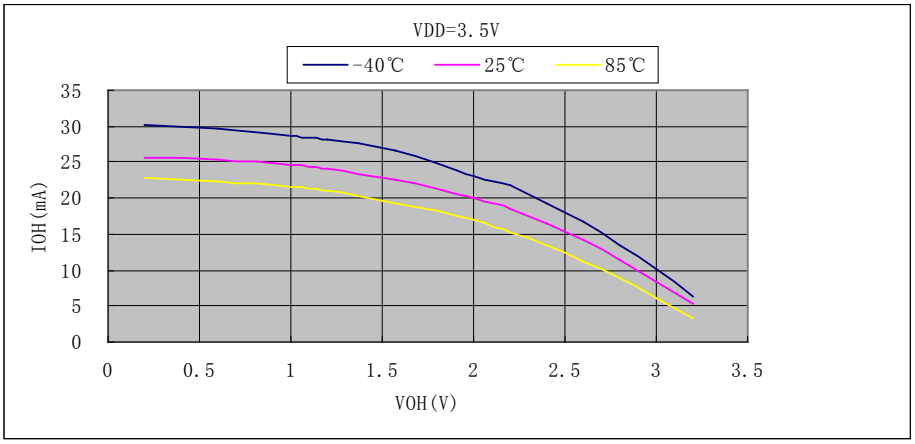
- I_{OH} vs. V_{OH} @ VDD = 2.5V (high drive)



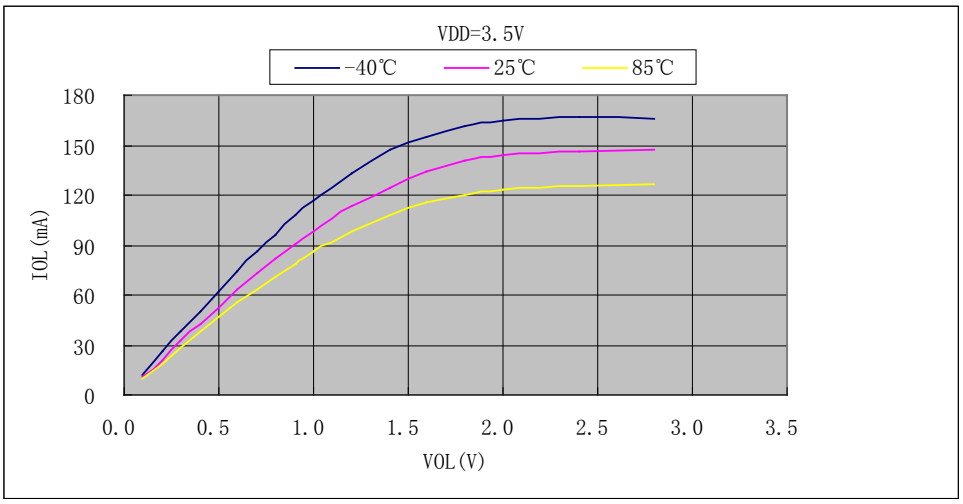
- I_{OL} vs. V_{OL} @ VDD = 2.5V (high drive)



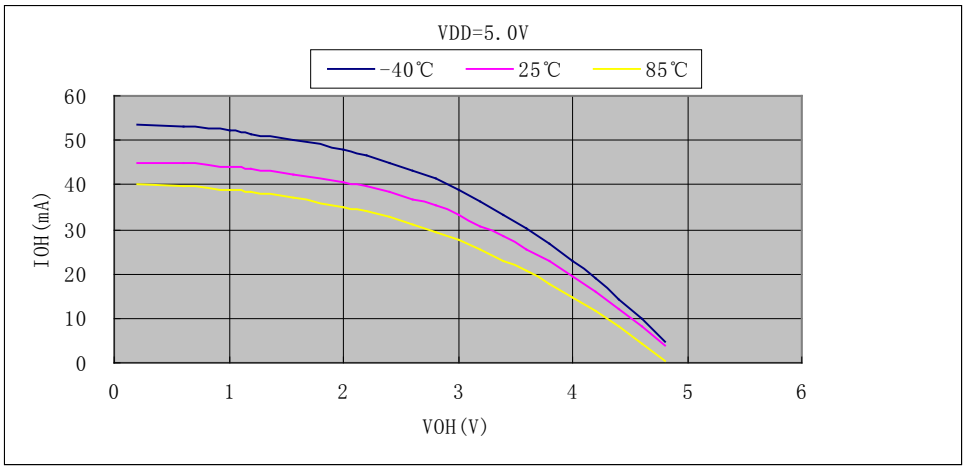
- I_{OH} vs. V_{OH} @ VDD = 3.5V (high drive)



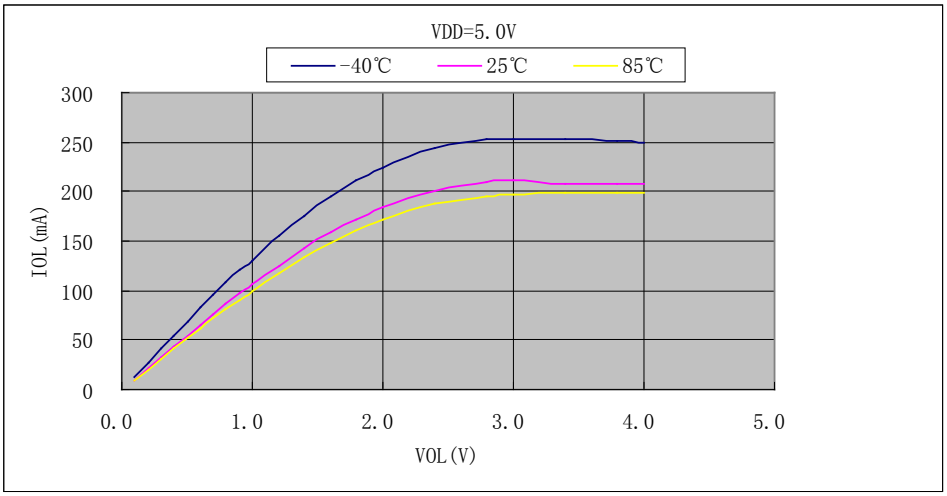
- I_{OL} vs. V_{OL} @ VDD = 3.5V (high drive)



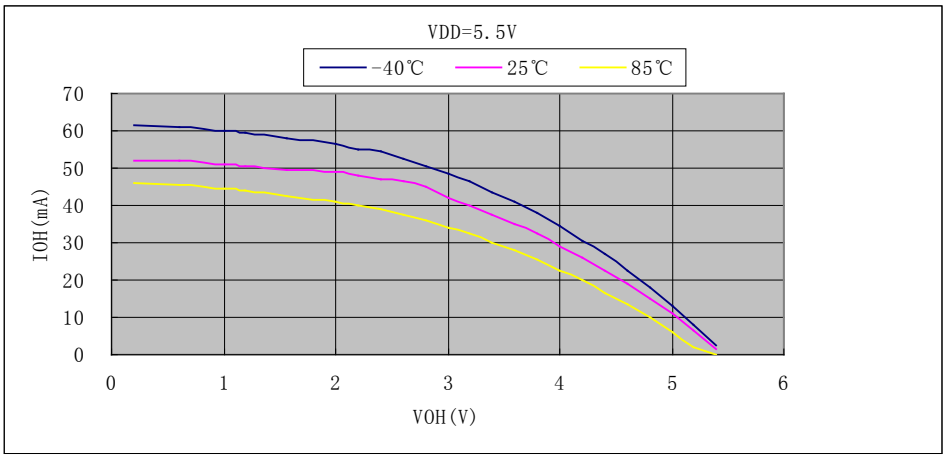
◆ I_{OH} vs. V_{OH} @ VDD = 5.0V (high drive)



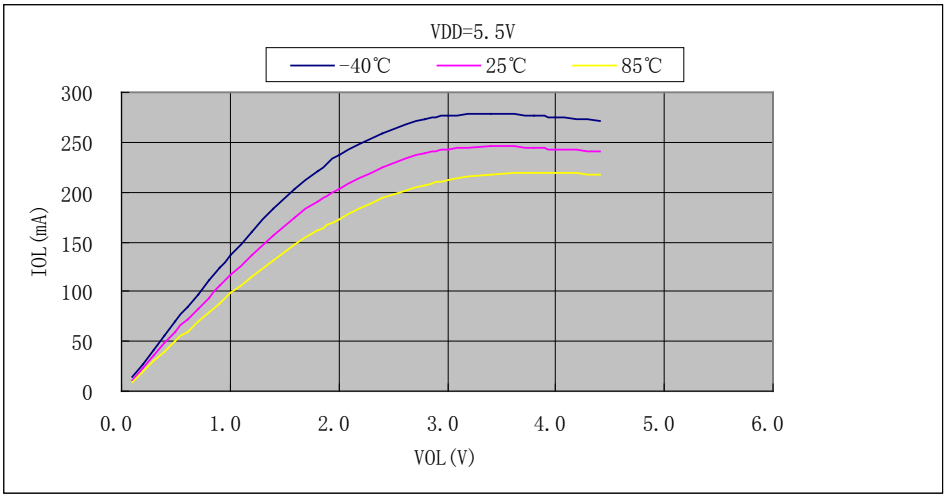
◆ I_{OL} vs. V_{OL} @ VDD = 5.0V (high drive)



◆ I_{OH} vs. V_{OH} @ VDD = 5.5V (high drive)



- I_{OL} vs. V_{OL} @ VDD = 5.5V (high drive)



Appendix3 Programming and Debug Interface

Appendix3. 1Overview

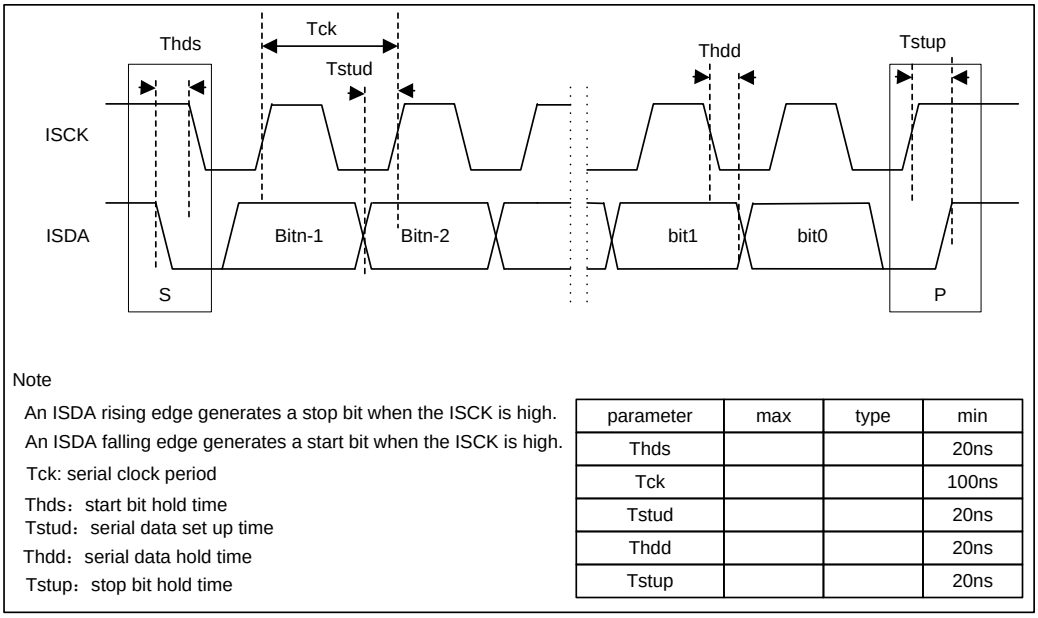
The integrated ISP (In-system programming) interface and SWD (Serial Wire Debug) interface are provided for easy debug of application programs and actual systems. With the company authorized ISP programmer, the SWD debugger is able to perform the in system programming, simulation and debugging.

The ISP and SWD share the 5-wire interface, including VDD, VSS, MRSTN, ISCK and ISDA.

Appendix3. 2ISP Interface

Appendix3. 2. 1 Communication Protocol

The ISP interface protocol adopts the 2-wire half-duplex technique. Each message packet contains N data bits, start bit and stop bit. The programmer is a master device and initiates the ISCK clock, and the chip is a slave device.



Note

An ISDA rising edge generates a stop bit when the ISCK is high.

An ISDA falling edge generates a start bit when the ISCK is high.

Tck: serial clock period

Thds：start bit hold time

Tstud：serial data set up time

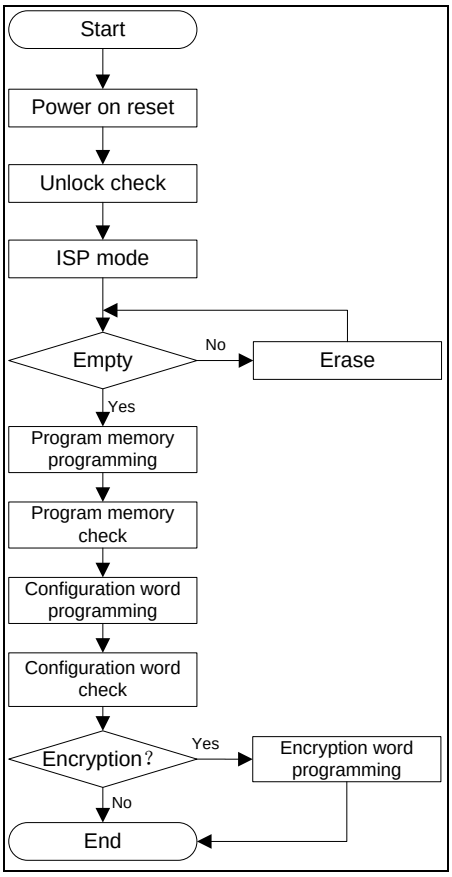
Thdd：serial data hold time

Tstup：stop bit hold time

parameter	max	type	min
Thds			20ns
Tck			100ns
Tstud			20ns
Thdd			20ns
Tstup			20ns

There are two sets of pins for ISP interface selection, PA0:PA1 and PA14:PA15. When one set it selected, the other will automatically be disabled.

Appendix3. 2. 2 Operation Flowchart



Appendix3. 3 SWD Interface

Appendix3. 3. 1 Overview

SWD is the built-in serial debug interface in Cortex-M0, and is compatible with the ARM CoreSight technique. The chip needs to power up again after downloading the debug program using the SWD debug (debug mode needs to be enabled by configuring the CFG_DEBUG). The SWDIO (multiplexed with ISDA) and SWCLK (multiplexed with ISCK) pins can be used.

SWCLK: Serial clock input pin, providing SWD clock

SWDIO: Serial data input/output pin

Two sets of SWD pins, PA0:PA1 and PA14:PA15, are controlled by the DEBUG_S of the CFG_WORD0 together with the CFG_DEBUG.

When the CFG_DEBUG =10, the debug mode is enabled, PA0:PA1 and PA14:PA15 are forced to be SWD pins, and are not available for other peripherals. If DEBUG_S = 0, PA14:PA15 are selected. If DEBUG_S = 1, PA0:PA1 are selected.

Appendix3. 3. 2 SWD Characteristics

Two debug approaches are available, intrusive debugging and non-intrusive debugging.

Intrusive debugging

- ◇ Halt
- ◇ Single stepping of program execution
- ◇ Hardware breakpoints (up to 4 hardware breakpoints)
- ◇ Software breakpoints (BKPT instruction)
- ◇ Alter PC value
- ◇ Data watchpoints DWT
- ◇ Data Watchpoint and Trace: only watchpoint is supported; Trace is not supported
- ◇ Internal registers and RAM read and write accessible
- ◇ Vector catch (Reset and Hard Fault exception included)

Non-intrusive debugging

- ◇ Program counter PC sampling

Appendix4 LCD Drive Waveforms

Appendix4. 1Overview

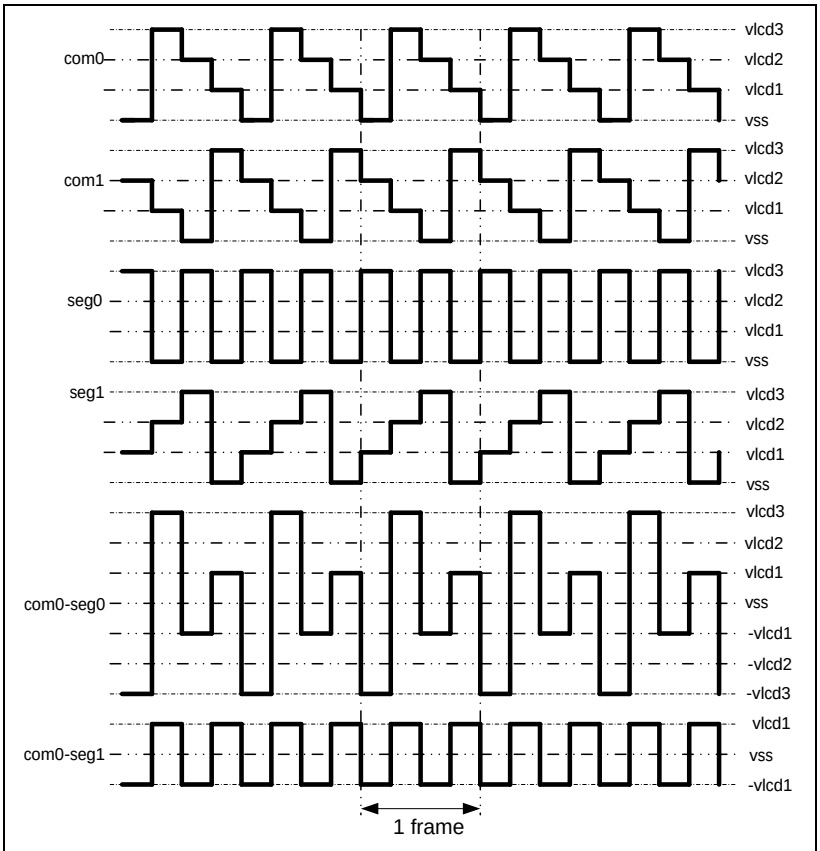
To generate LCD waveforms, the net AC voltage across the dark pixel should be maximized and the net AC voltage across the clear pixel should be minimized. And the net DC voltage across any pixel should be zero. The COM signal represents the time slice for each common while the SEG contains the pixel data.

The pixel signal (COM-SEG) will not have DC component and it can take only one of the two RMS values. The higher RMS value will generate a dark pixel while the lower RMS value will generate a clear pixel. With the growing number of commons, the difference between the two RMS values will decrease, and this difference represents the maximum contrast that the display can have.

There are two LCD waveforms available, Type A and Type B. In Type A waveform, the phase changes within each common type, whereas in Type B waveforms, the phase changes on each frame boundary. Therefore, Type A waveform maintains 0V DC over a single frame while Type B waveform requires two frames.

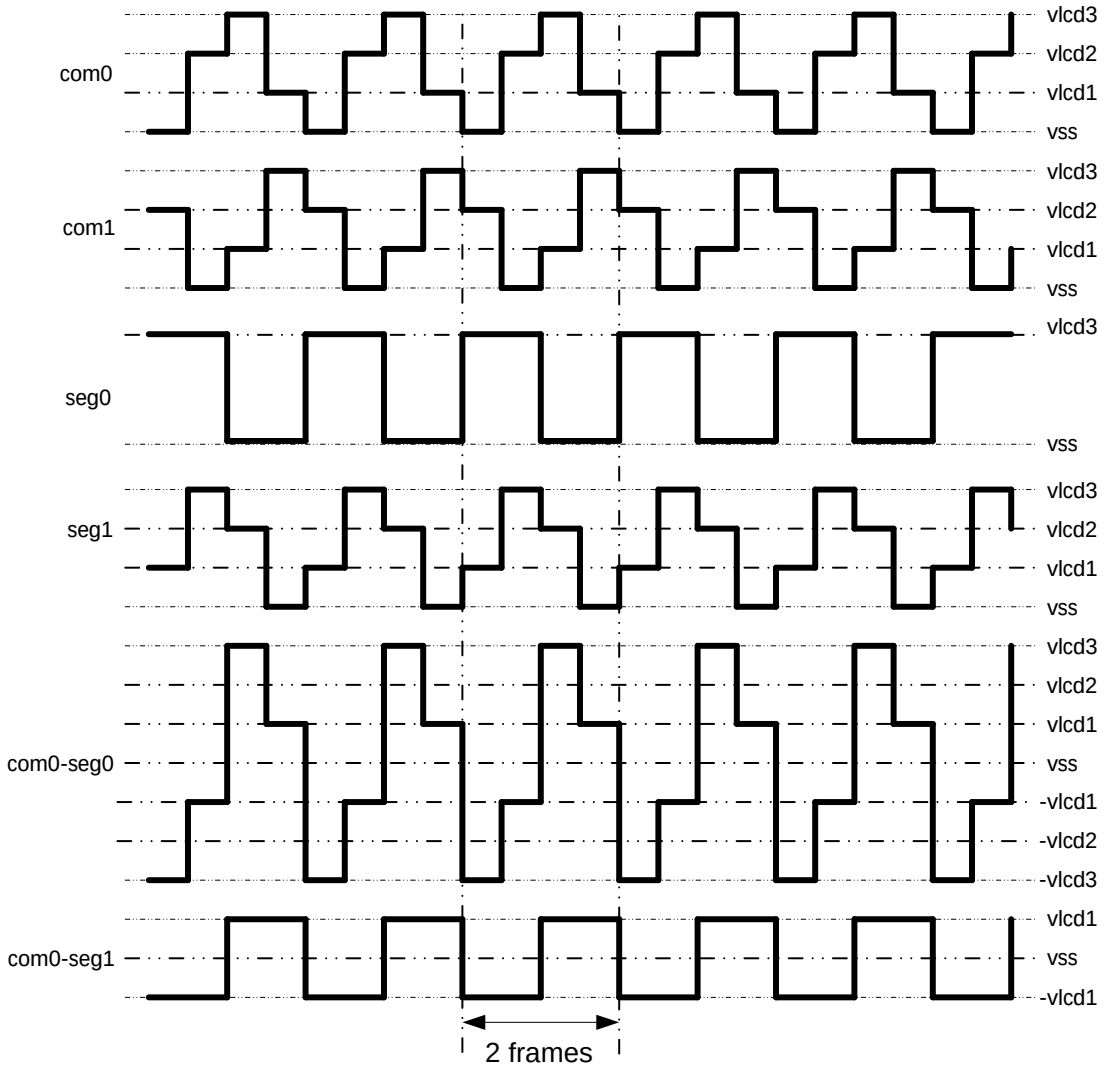
Appendix4. 2Drive Waveforms

◆ Type A Waveform in 1/2 Mux and 1/3 Bias Drive



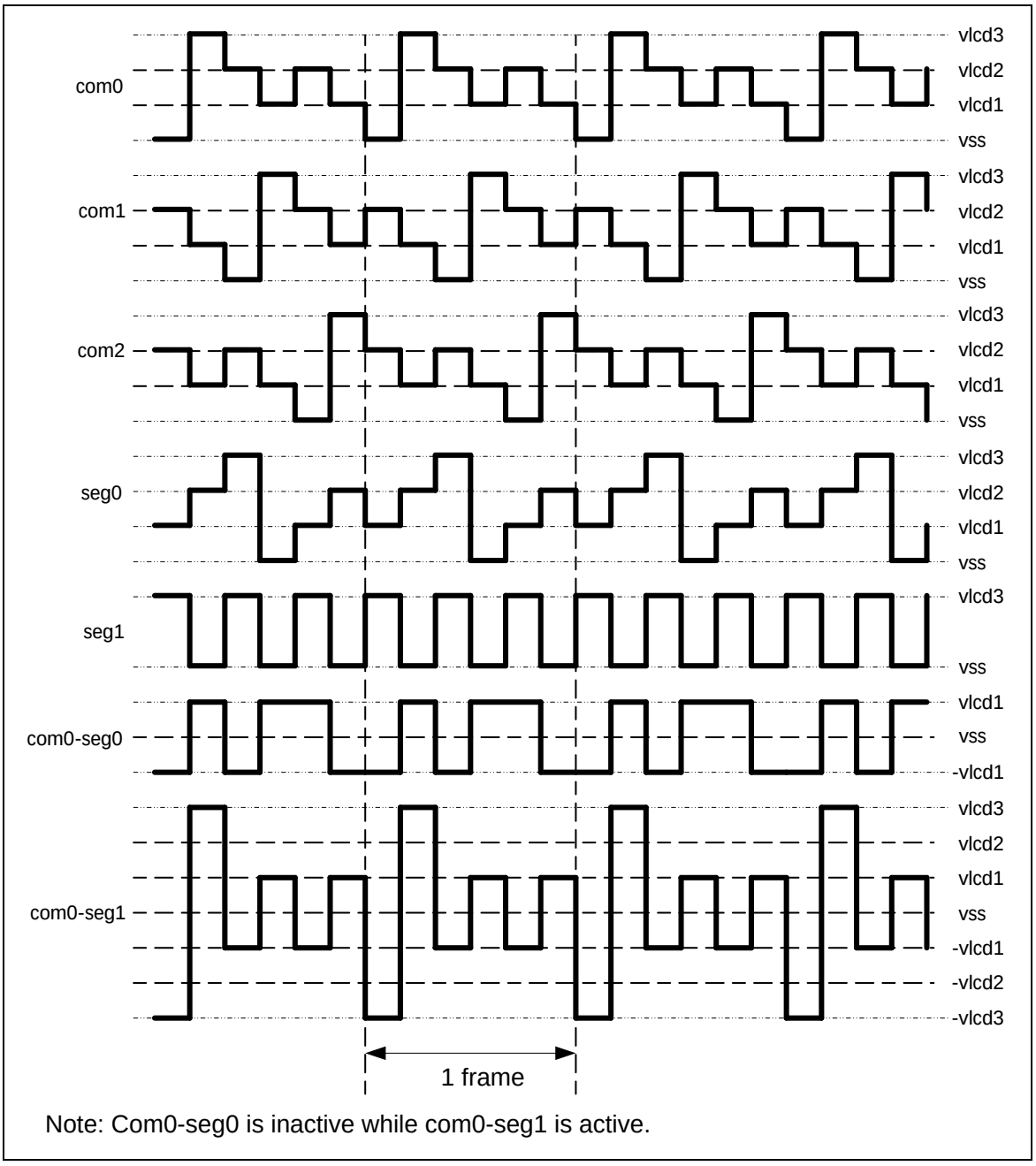
Note: Com0-seg0 is active while com0-seg1 is inactive.

◆ Type B Waveform in 1/2 Mux and 1/3 Bias Drive



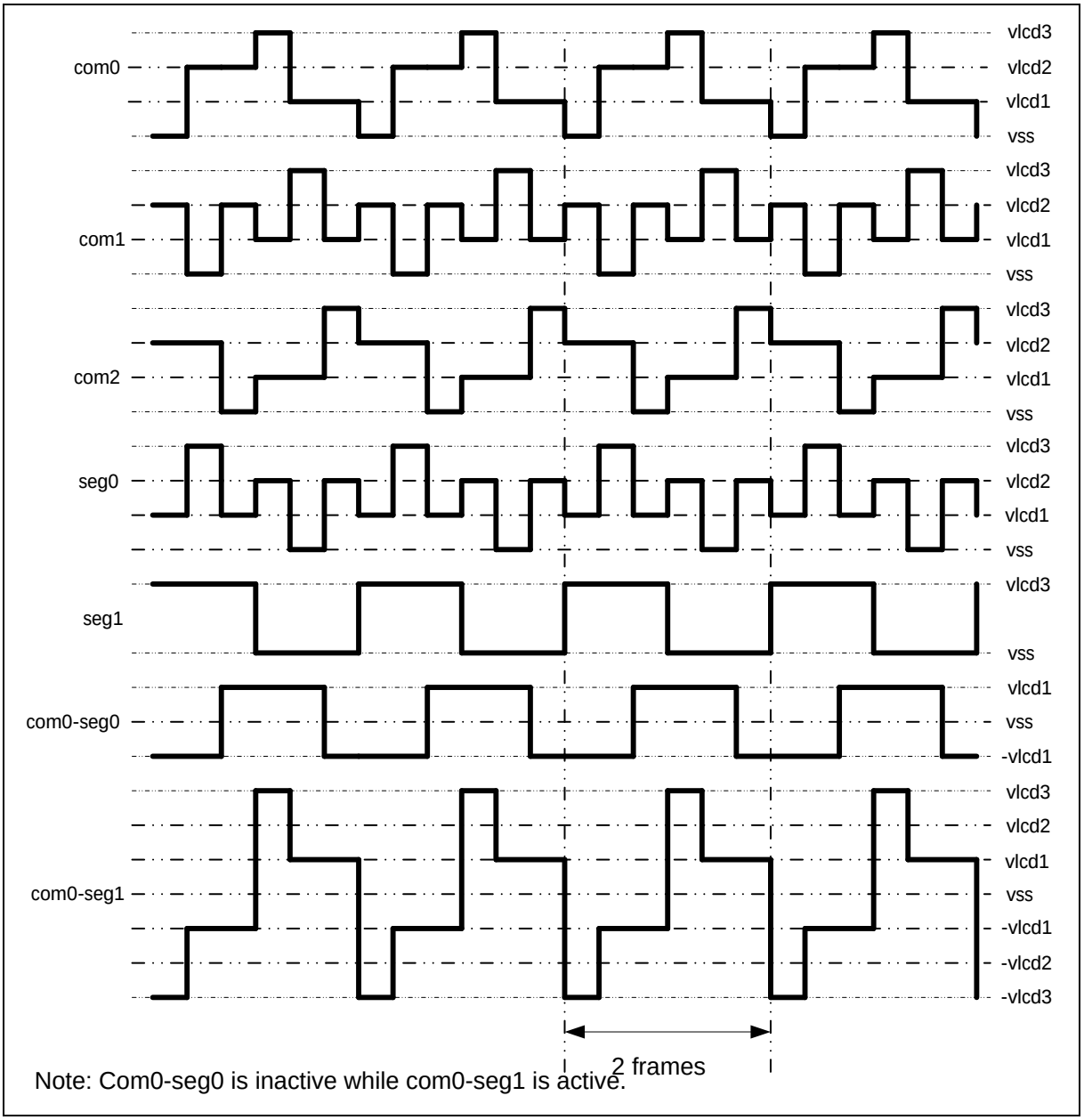
Note: Com0-seg0 is active while Com0-seg1 is inactive.

◆ Type A Waveform in 1/3 Mux and 1/3 Bias Drive



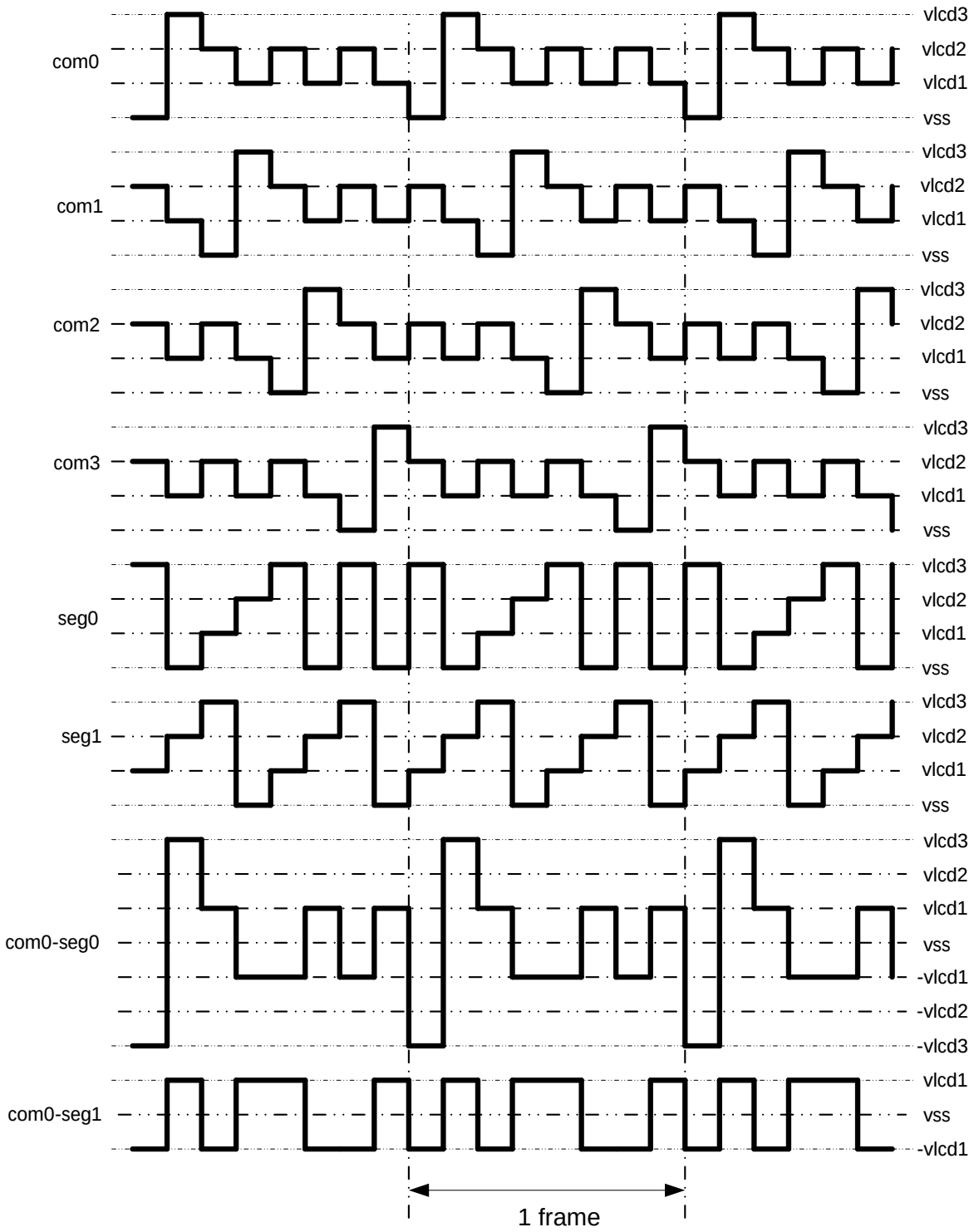
Note: Com0-seg0 is inactive while com0-seg1 is active.

◆ Type B Waveform in 1/3 Mux and 1/3 Bias Drive



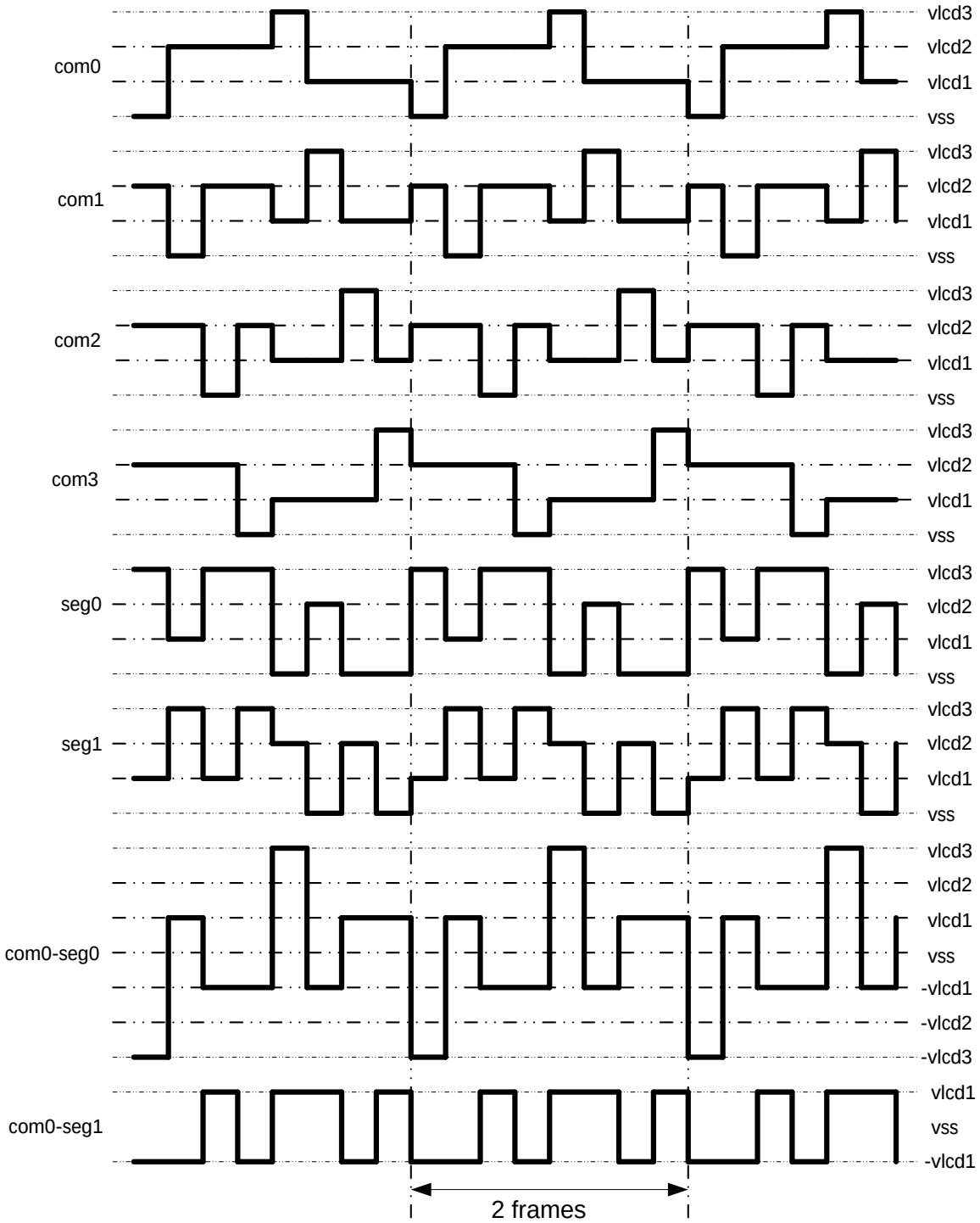
Note: Com0-seg0 is inactive while com0-seg1 is active.

◆ Type A Waveform in 1/4 Mux and 1/3 Bias Drive



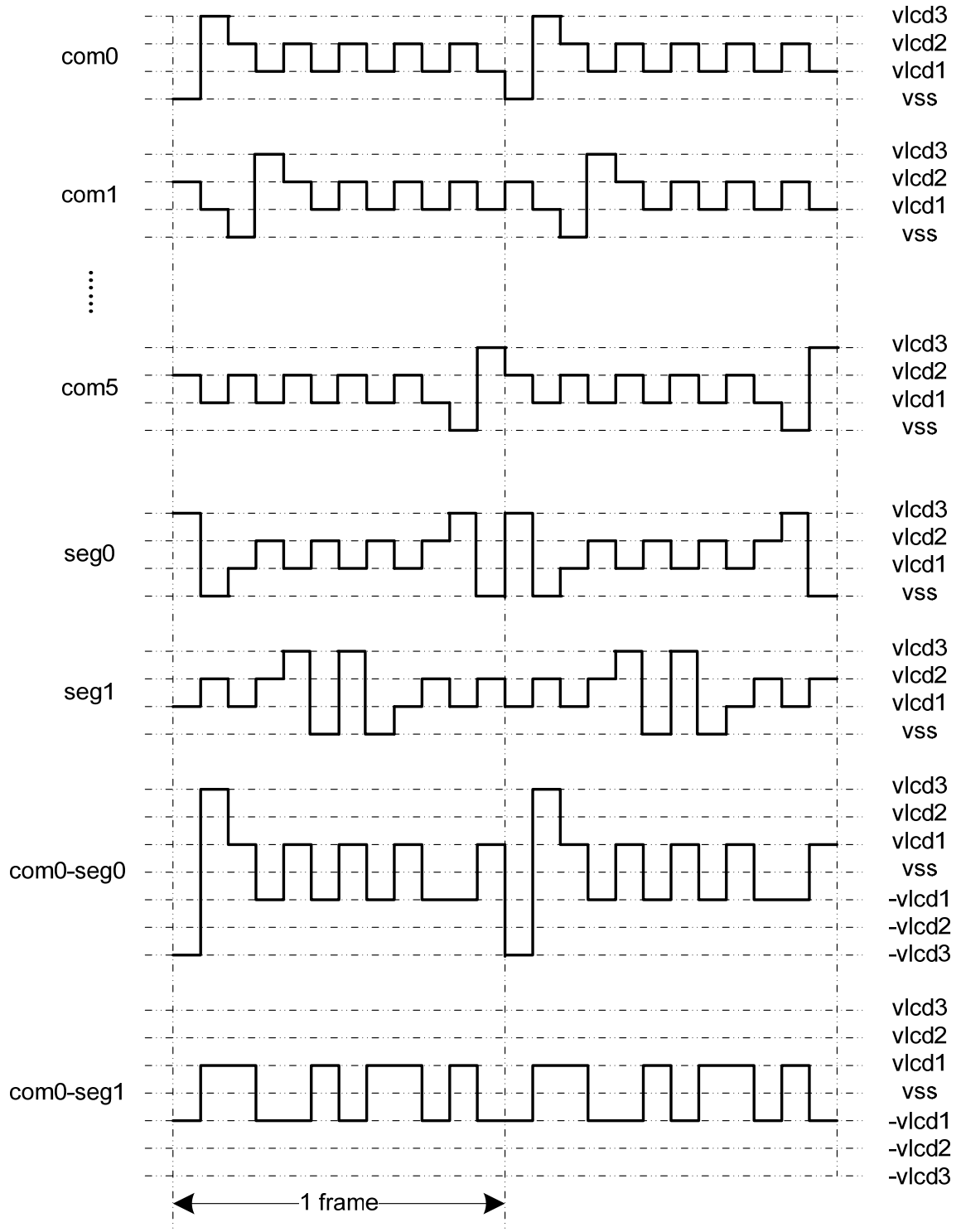
Note: Com0-seg0 is active while com0-seg1 is inactive.

◆ Type B Waveform in 1/4 Mux and 1/3 Bias Drive



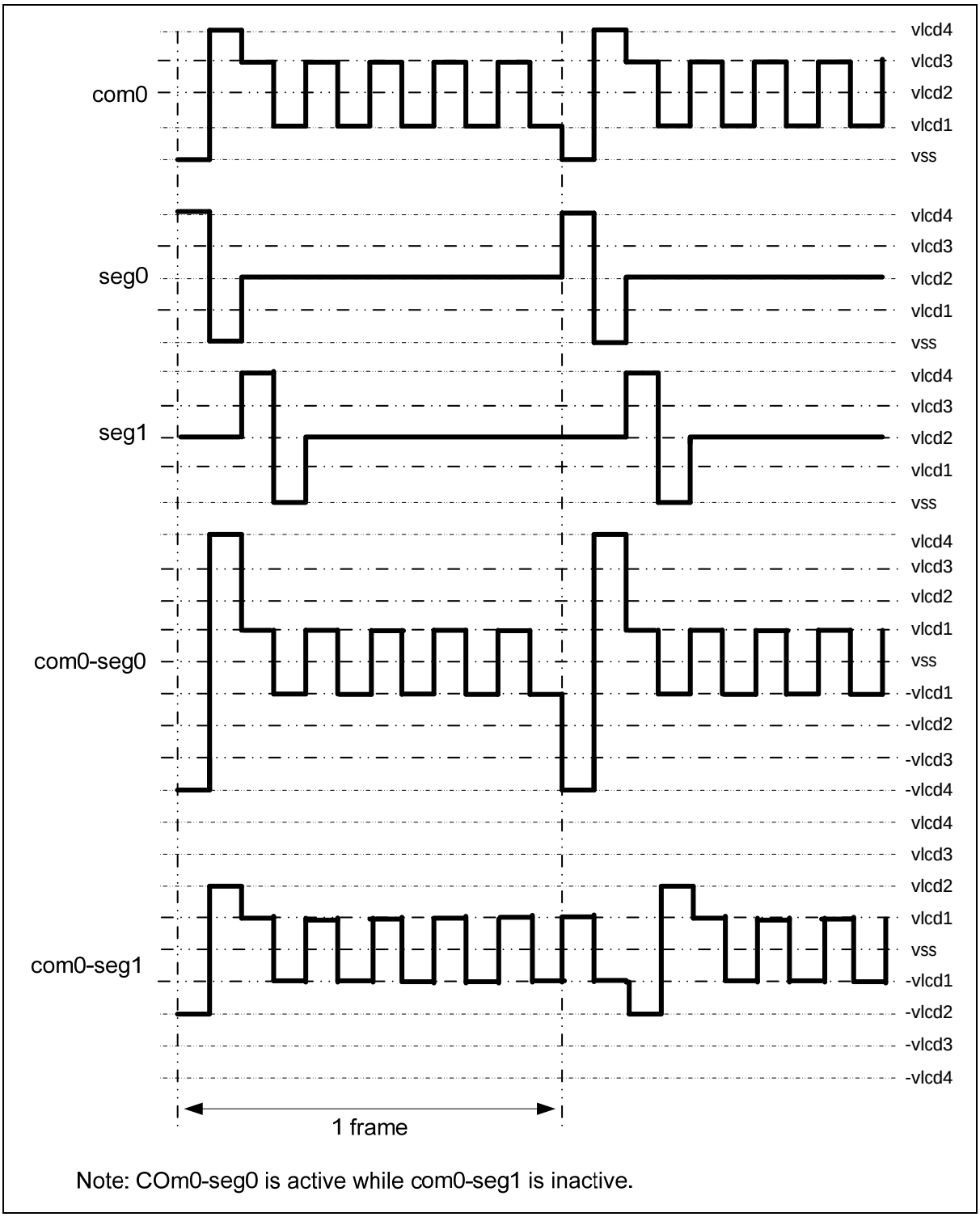
Note: Com0-seg0 is active while Com0-seg1 is inactive.

◆ Type A Waveform in 1/6 Mux and 1/3 Bias Drive



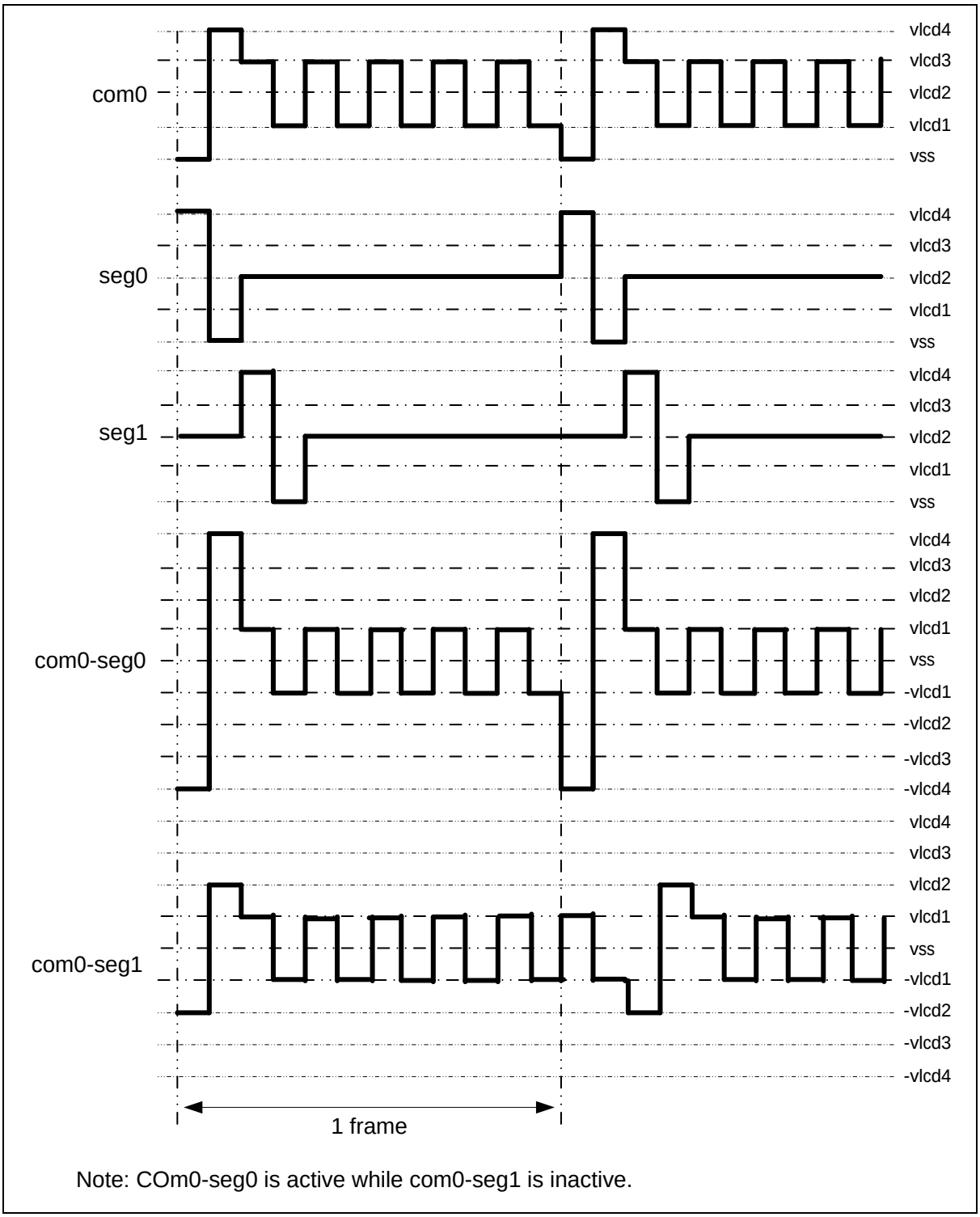
Note: Com0-seg0 is active while com0-seg1 is inactive.

◆ Type A Waveform in 1/6 Mux and 1/4 Bias Drive



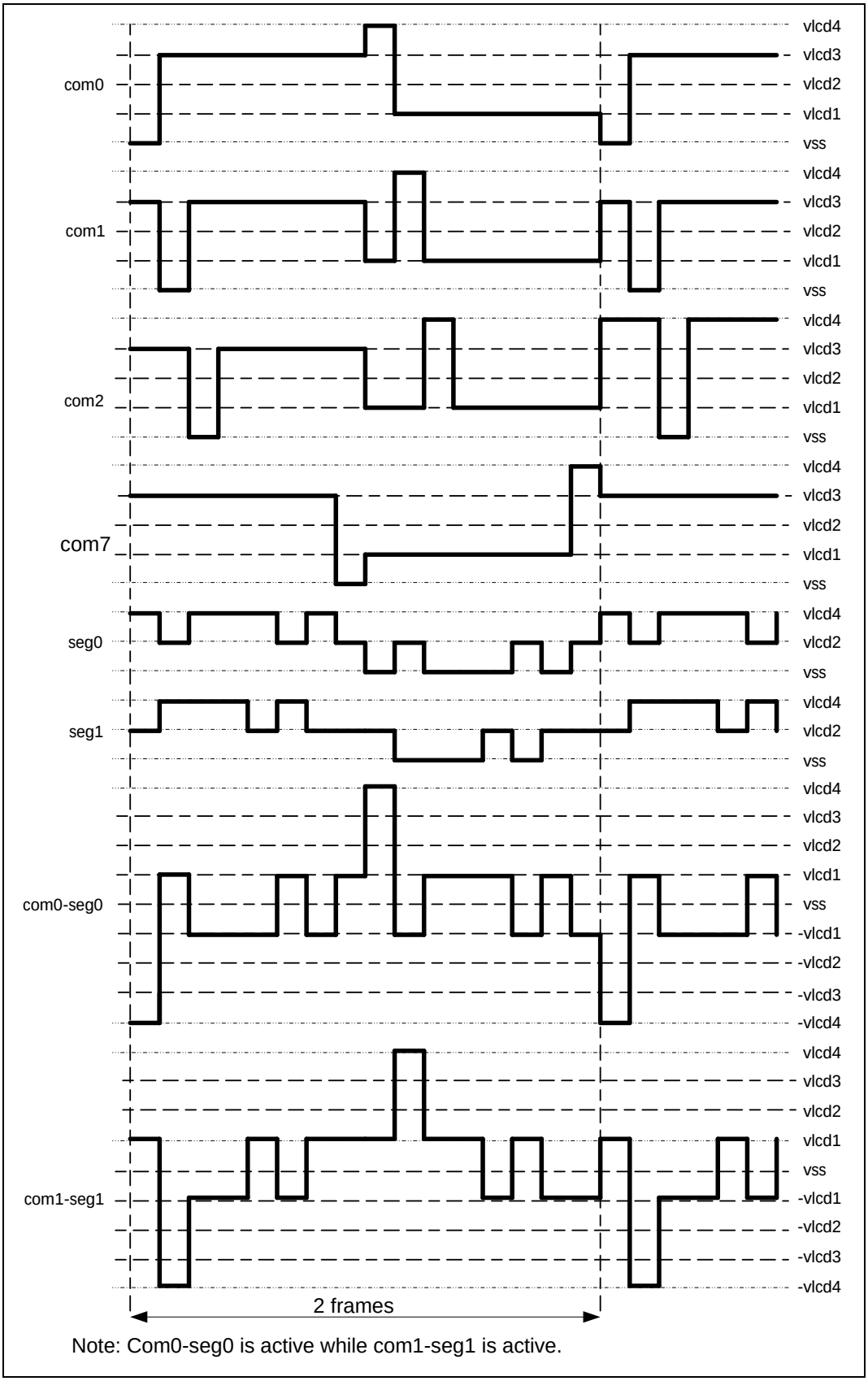
Note: COm0-seg0 is active while com0-seg1 is inactive.

◆ Type A Waveform in 1/6 Mux and 1/4 Bias Drive



Note: COm0-seg0 is active while com0-seg1 is inactive.

◆ Type B Waveform in 1/8 Mux and 1/4 Bias Drive



Note: Com0-seg0 is active while com1-seg1 is active.