# A96L322
# User's Manual

## CMOS Single-chip 8-bit MCU with 10-bit ADC and Operational Amplifier

Version 1.00

## Introduction

This user manual contains complete information for application developers who use A96L322 for their specific needs.

A96L322 is an advanced 8-bit CMOS MCU with 4 Kbytes of Flash. Offering the users the convenience of Flash multi-programming features, this device can provide fire alarm and smoke detectors systems with a simple, robust, and cost effective solution.

As a complete set of semiconductor products to implement the smart alarm systems, AL1113 can be used for power supply peripheral.

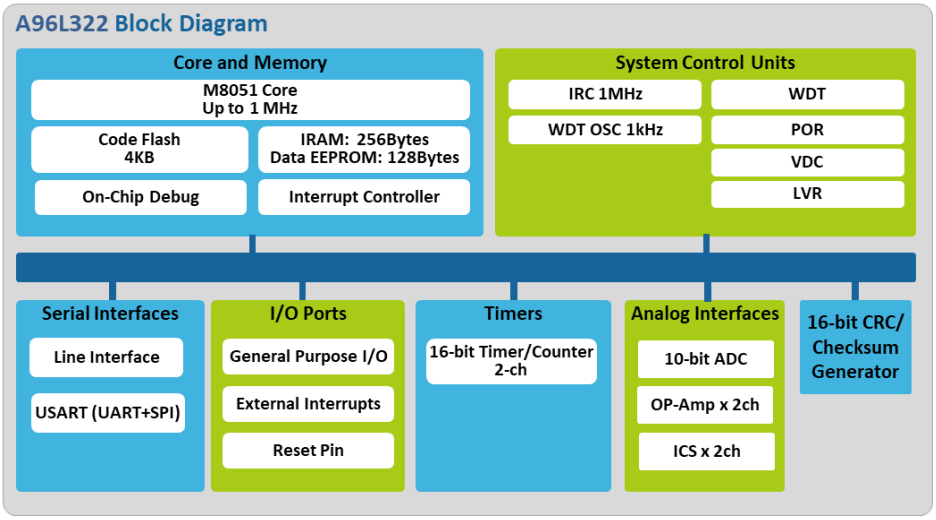



**Figure 1. A96L322 Block Diagram**

## Reference documents

- A96L322 Datasheet: information on mechanical characteristics, development methods, and ordering information.
- AL1113 Datasheet: information on mechanical characteristics, development methods, and ordering information.

# Contents

# List of figures

# List of tables

# 1 Description

A96L322 is an advanced CMOS 8-bit microcontroller with 4 Kbytes of FLASH. This is a powerful microcontroller which provides low power consumption and cost effective solution to smoke detector applications.

Table 1 introduces features of A96L322 and peripheral counts. In addition, A96L322 supports power down modes to reduce power consumption.

## 1.1 Device overview

**Table 1. A96L322 Device Features and Peripheral Counts**

| Peripheral / Device | | A96L322 |
|---|---|---|
| CPU | | 8-bit CISC core (M8051, 2 clocks per cycle) |
| Flash | | • 4 Kbytes with self r/w capability<br>• On chip debug and ISP<br>• Endurance: 10,000 cycles (sector 0 to 123)/ 100,000 cycles (sector 124 to 127) |
| iRAM | | 256 bytes |
| EEPROM | | • 128 bytes<br>• Endurance: 100,000 cycles |
| GPIO | | • Normal I/Os<br>• 14 ports: P0[7:0], P1[5:0] |
| Timer/ counter | | • BIT 8-bit x 1-ch<br>• WDT 8-bit x 1-ch: 1 KHz internal RC oscillator for WDT<br>• 16-bit x 2-ch (T0/T1)<br>• Siren (by T1) |
| Programmable pulse generation | | Pulse generation (by T0/T1) |
| Line interface | | Three Rx types and five Tx modes |
| ADC | | 10-bit ADC, 9 input channels |
| Operational amplifier | | • 2-ch<br>• Rail-to-rail output |
| CRC and checksum generator | | • 16-bit<br>• Auto and user CRC/ checksum mode |
| Reset | Power on reset | Reset release level (1.4V) |
| | Low voltage reset | 3 level detect (1.60V/ 2.20V/ 2.70V) |

| Peripheral / Device | A96L322 |
|---|---|
| Constant sink current generator | • 2-ch<br>• 16-steps selectable<br>• Max. 274mA sink current |
| USART | UART + SPI<br>• 8-bit UART x 1-ch<br>• 8-bit SPI x 1-ch |
| Interrupt sources | • External interrupts: EINT0/1/2/3/10/11, 6<br>• Timer0/1, 2<br>• WDT1<br>• BIT1<br>• Line interface Rx/ Tx (2)<br>• ADC 1<br>• Siren 1<br>• USART Rx/ Tx 2 |
| Internal RC oscillator | 1MHz ± 3.0% ($T_A$ = -40℃ to +85℃) |
| Power down mode | STOP, IDLE |
| Operating voltage and frequency | • 2.0V to 3.6V @ 0.125 to 1.0MHz with IRC<br>• Voltage dropout converter included for core |
| Minimum instruction execution time | 2us @ 1MHz IRC |
| Operating temperature | -40℃ to +85℃ |
| Package type | • 16 SOPN<br>• Pb-free package |

## 1.2 Block diagram

Figure 2 describes A96L322 in a block diagram.

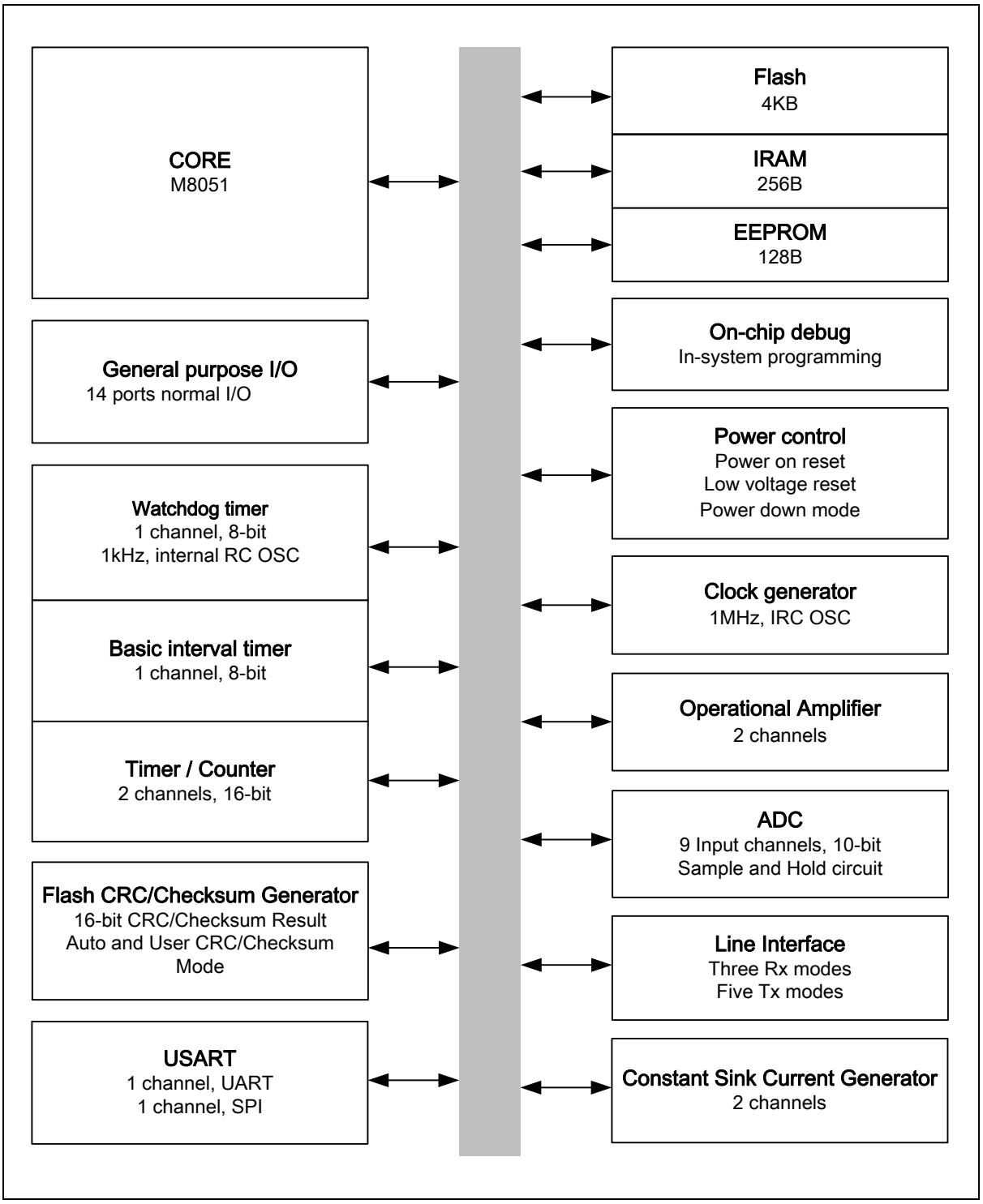
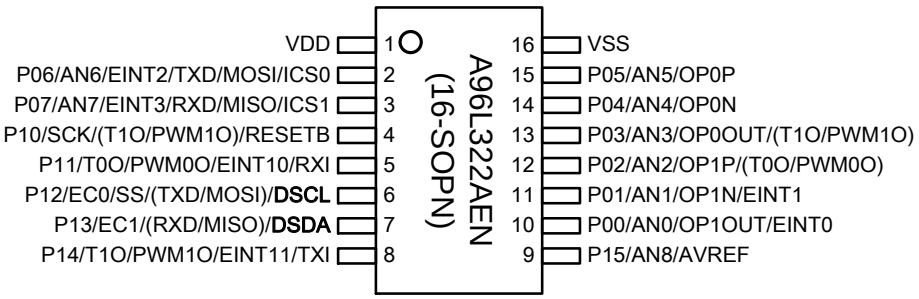



Figure 2. A96L322 Block Diagram

# 2 Pinouts and pin descriptions

In this chapter, A96L322 pinouts and pin descriptions are introduced.

## 2.1 Pinouts




**NOTE**: For the On-Chip Debugging, ISP assigns P1[2:3] pins for DSCL and DSDA respectively.

**Figure 3. A96L322AEN 16 SOPN Pinouts**

## 2.2 Pin description

**Table 2. 16 SOPN Pin Description**

| Pin name | I/O | Function | @reset | Shared with |
|---|---|---|---|---|
| P00 | I/O | The port 0 is a bit-programmable I/O port which can be configured as an input (P00/P01/P06/P07: Schmitt trigger input), a push-pull output, or an open-drain output. A pull-up resistor can be specified in 1-bit unit. | Input | AN0/OP1OUT/EINT0 |
| P01 | | | | AN1/OP1N/EINT1 |
| P02 | | | | AN2/OP1P/(T0O/PWM0O) |
| P03 | | | | AN3/OP0OUT/(T1O/PWM1O) |
| P04 | | | | AN4/OP0N |
| P05 | | | | AN5/OP0P |
| P06 | | | | AN6/EINT2/TXD/MOSI/ICS0 |
| P07 | | | | AN7/EINT3/RXD/MISO/ICS1 |
| P10 | I/O | The port 1 is a bit-programmable I/O port which can be configured as a Schmitt-trigger input, a push-pull output, or an open-drain output. A pull-up resistor can be specified in 1-bit unit. | Input | SCK/(T1O/PWM1O)/RESETB |
| P11 | | | | T0O/PWM0O/EINT10/RXI |
| P12 | | | | EC0/SS/(TXD/MOSI)/DSCL |
| P13 | | | | EC1/(RXD/MISO)/DSDA |
| P14 | | | | T1O/PWM1O/EINT11/TXI |
| P15 | | | | AN8/AVREF |
| EINT0 | I/O | External interrupt inputs | Input | P00/AN0/OP1OUT |
| EINT1 | | | | P01/AN1/OP1N |

| Pin name | I/O | Function | @reset | Shared with |
|---|---|---|---|---|
| EINT2 | | | | P06/AN6/TXD/MOSI/ICS0 |
| EINT3 | | | | P07/AN7/RXD/MISO/ICS1 |
| EINT10 | I/O | External interrupt input and Timer 0 capture input | Input | P11/T0O/PWM0O/RXI |
| EINT11 | I/O | External interrupt input and Timer 1 capture input | | P14/T1O/PWM1O/TXI |
| T0O | I/O | Timer 0 interval output | | P11/PWM0O/EINT10/RXI (P02/AN2/OP1P/PWM0O) |
| T1O | I/O | Timer 1 interval output | | P14/PWM1O/EINT11/TXI (P03/AN3/OP0OUT/PWM1O) (P10/SCK/PWM1O/RESETB) |
| PWM0O | I/O | Timer 0 pulse output | | P11/T0O/EINT10/RXI(P02/AN2/OP1P/T0O) |
| PWM1O | I/O | Timer 1 pulse output | | P14/T1O/EINT11/TXI (P03/AN3/OP0OUT/T1O) (P10/SCK/T1O/RESETB) |
| EC0 | I/O | Timer 0 event count input | | P12/SS/DSCL |
| EC1 | I/O | Timer 1 event count input | | P13/DSDA |
| RXI | I/O | Line interface receive input | | P11/T0O/PWM0O/EINT10 |
| TXI | I/O | Line interface transmit output | | P14/T1O/PWM1O/EINT11 |
| AN0 | I/O | A/D converter analog input channels | Input | P00/OP1OUT/EINT0 |
| AN1 | | | | P01/OP1N/EINT1 |
| AN2 | | | | P02/OP1P/(T0O/PWM0O) |
| AN3 | | | | P03/OP0OUT/(T1O/PWM1O) |
| AN4 | | | | P04/OP0N |
| AN5 | | | | P05/OP0P |
| AN6 | | | | P06/EINT2/TXD/MOSI/ICS0 |
| AN7 | | | | P07/EINT3/RXD/MISO/ICS1 |
| AN8 | | | | P15/AVREF |
| AVREF | I/O | A/D converter reference voltage | Input | AN8/P15 |
| OP0P | I/O | OP-AMP 0 positive input | Input | P05/AN5 |
| OP0N | I/O | OP-AMP 0 negative input | Input | P04/AN4 |
| OP0OUT | I/O | OP-AMP 0 output | Input | P03/AN3/(T1O/PWM1O) |
| OP1P | I/O | OP-AMP 1 positive input | Input | P02/AN2/(T0O/PWM0O) |
| OP1N | I/O | OP-AMP 1 negative input | Input | P01/AN1/EINT1 |

| Pin name | I/O | Function | @reset | Shared with |
|---|---|---|---|---|
| OP1OUT | I/O | OP-AMP 1 output | Input | P00/AN0/EINT0 |
| TXD | I/O | UART data output | Input | P06/EINT2/AN6/MOSI/ICS0 (P12/EC0/SS/MOSI/DSCL) |
| RXD | I/O | UART data input | Input | P07/EINT3/AN7/MISO/ICS1 (P13/EC1/MISO/DSDA) |
| MOSI | I/O | SPI master output, slave input | Input | P06/EINT2/AN6/TXD/ICS0 (P12/EC0/SS/TXD/DSCL) |
| MISO | I/O | SPI master input, slave output | Input | P07/EINT3/AN7/RXD/ICS1 (P13/EC1/RXD/DSDA) |
| SCK | I/O | SPI clock input/output | Input | P10/(T1O/PWM1O)/RESETB |
| SS | I/O | SPI slave select input | Input | P12/EC0/DSCL |
| ICS0 | I/O | Constant sink current pins | Input | P06/AN6/EINT2/TXD/MOSI |
| ICS1 | | | | P07/AN7/EINT3/RXD/MISO |
| RESETB | I/O | System reset pin with a pull-up resistor when it is selected as the RESETB by CONFIGURE OPTION. | Input | P10/SCK/(T1O/PWM1O) |
| DSCL | I/O | On chip debugger clock input | Input | EC0/P12/SS |
| DSDA | I/O | On chip debugger data input/output | Input | EC1/P13 |
| VDD, VSS | – | Power input pins | – | – |

**NOTES**:

1. The P10/RESETB pin is configured as one of the P10/SCK and the RESETB pin by the "CONFIGURE OPTION".
2. If the P13/DSDA and P12/DSCL pins are connected to an emulator during reset or power-on reset, the pins are automatically configured as the debugger pins.
3. The P13/DSDA and P12/DSCL pins are configured as inputs with an internal pull-up resistor only during the reset or power-on reset.

# 3 Port structures

## 3.1 GPIO port structure

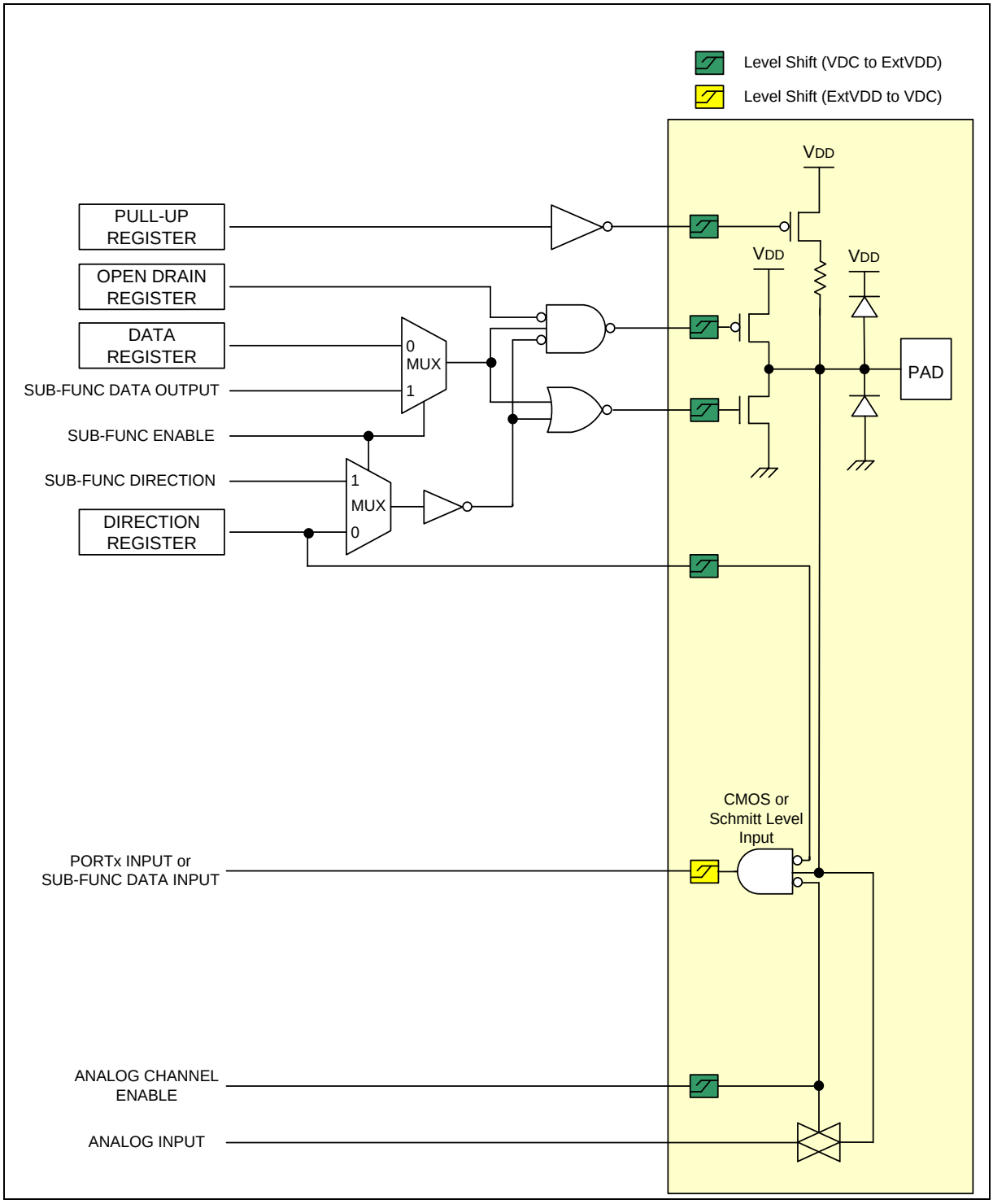



Figure 4. General Purpose I/O Port Structure

## 3.2 External interrupt I/O port structure

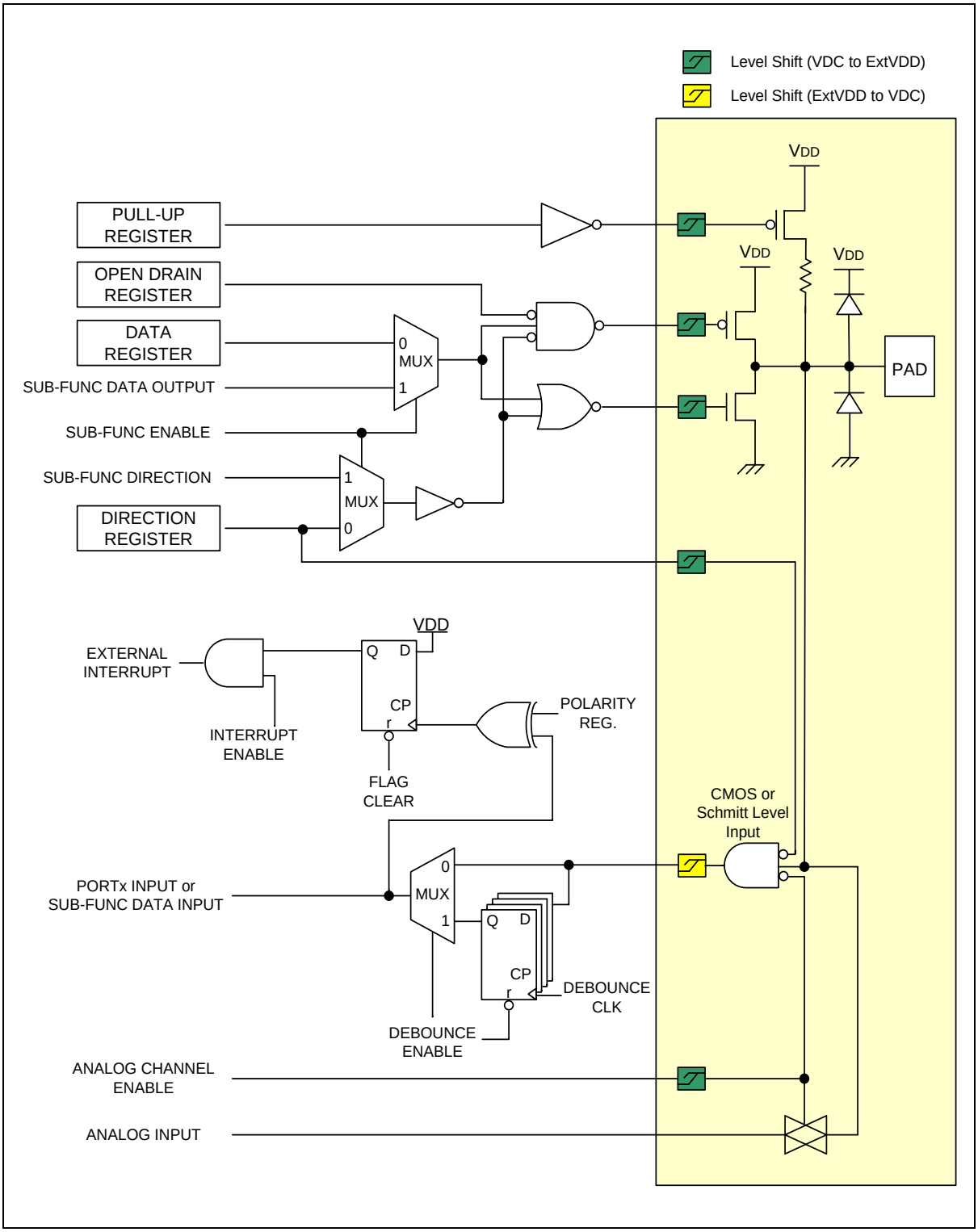



Figure 5. External Interrupt I/O Port Structure

# 4 Memory organization

A96L322 addresses two separate memory spaces:

- Program memory
- Data memory

By means of this logical separation of the memory, 8-bit CPU address can access the data memory more rapidly. 16-bit data memory address is generated through the DPTR register.

A96L322 provides on-chip 4 Kbytes of the ISP type flash program memory, which is readable and writable. Internal data memory (iRAM) is 256 bytes and it includes the stack area.

## 4.1 Program memory

A 16-bit program counter is capable of addressing up to 64 Kbytes, but A96L322 has only 4 Kbytes program memory space. After reset, CPU begins execution from location 0000H. Each interrupt is assigned to a fixed location of the program memory. The interrupt causes the CPU to jump to that location, where it commences an execution of a service routine.

For example, an external interrupt 1 is assigned to location 000BH. If the external interrupt 1 is going to be used, its service routine must begin at location 000BH. If the interrupt is not going to be used, its service location is available as general purpose program memory. If an interrupt service routine is short enough (frequent cases with a control application), the service routine can reside entirely within an 8 byte interval.

A longer service routine can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use. Figure 6 shows a map of the lower part of the program memory.

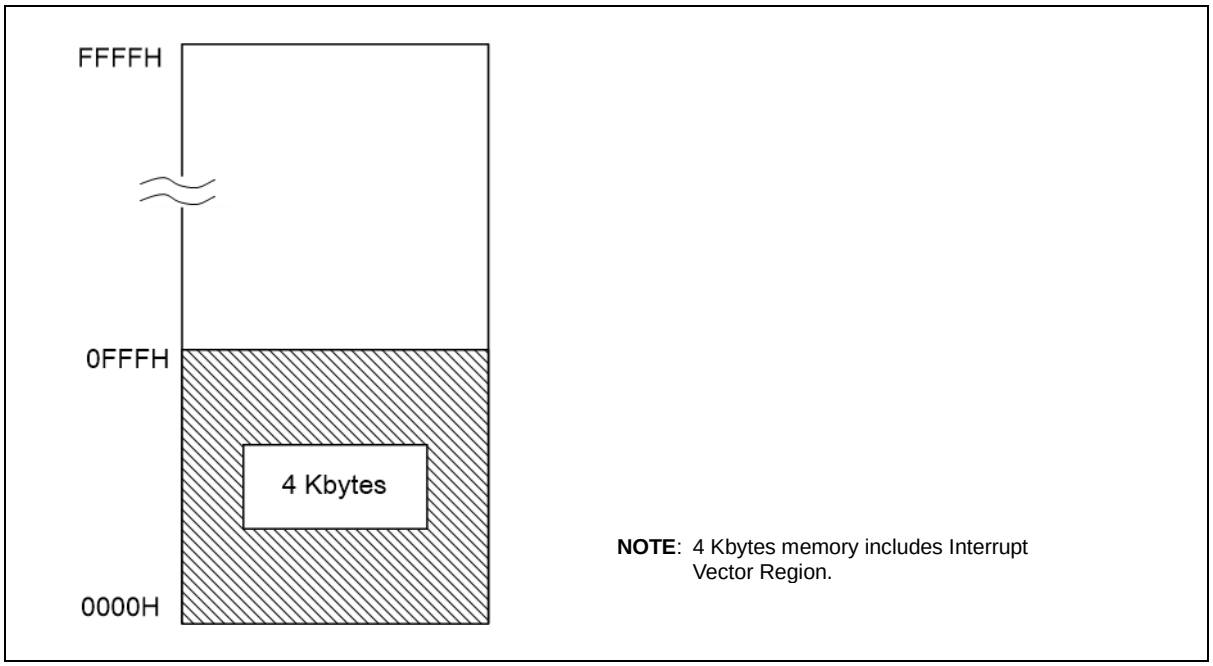



Figure 6. Program Memory

More detailed description of program memory is introduced in chapter 20. Flash memory later part in this document.

## 4.2 Internal data memory

Internal data memory is divided into three spaces as shown in figure 7. Those three spaces are generally called as,

- Lower 128 bytes
- Upper 128 bytes
- Special Function Registers (SFR space)

Internal data memory addresses are always one byte wide, which implies an address space of 256 bytes.

In fact, the addressing modes of the internal data memory can accommodate up to 384 bytes by using a simple trick. Direct addresses higher than 7FH access one memory space and indirect addresses higher than 7FH access a different memory space. By means of this method, the upper 128 bytes and SFR space can occupy the same block of addresses, 80H through FFH, although they are physically separate entities as shown in figure 3.

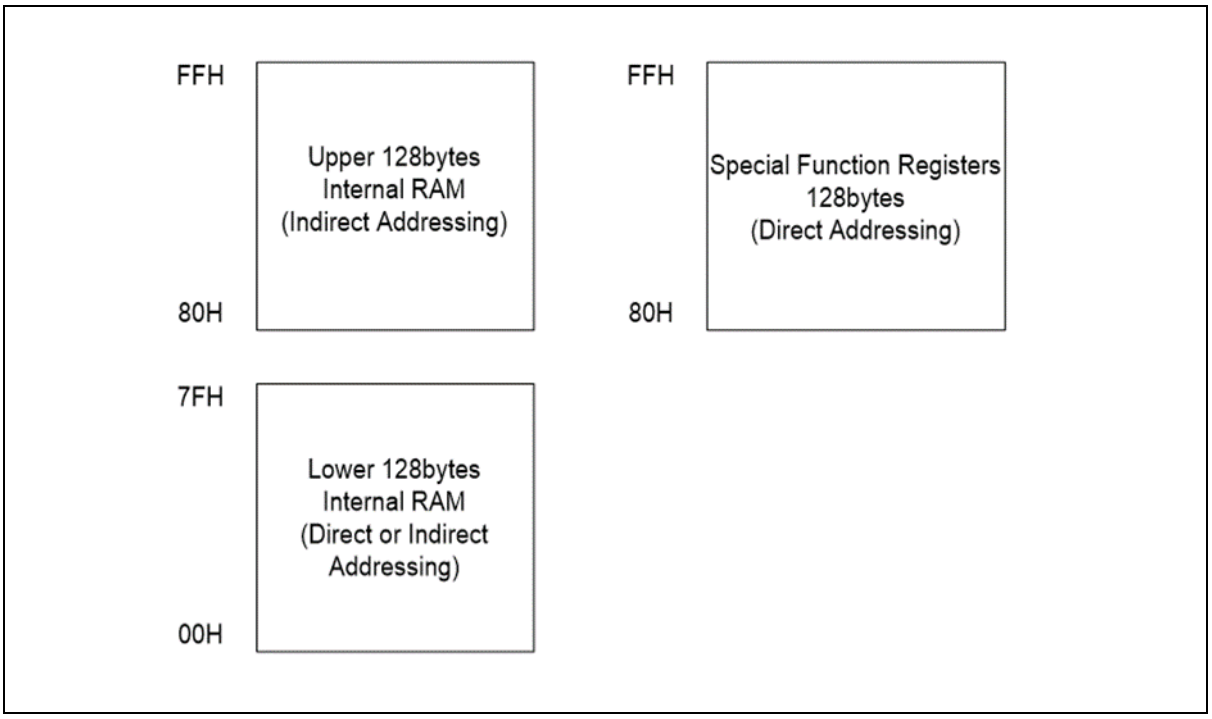



**Figure 7. Internal Data Memory Map**

The lower 128 bytes of RAM are present in all 8051 devices as mapped in figure 8. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing.

The next 16 bytes above the register banks form a block of bit-addressable memory space. The 8051 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

Entire bytes in the lower 128 bytes can be accessed by either direct or indirect addressing, while the upper 128 bytes RAM can only be accessed by indirect addressing. These spaces are used for data RAM and stack.

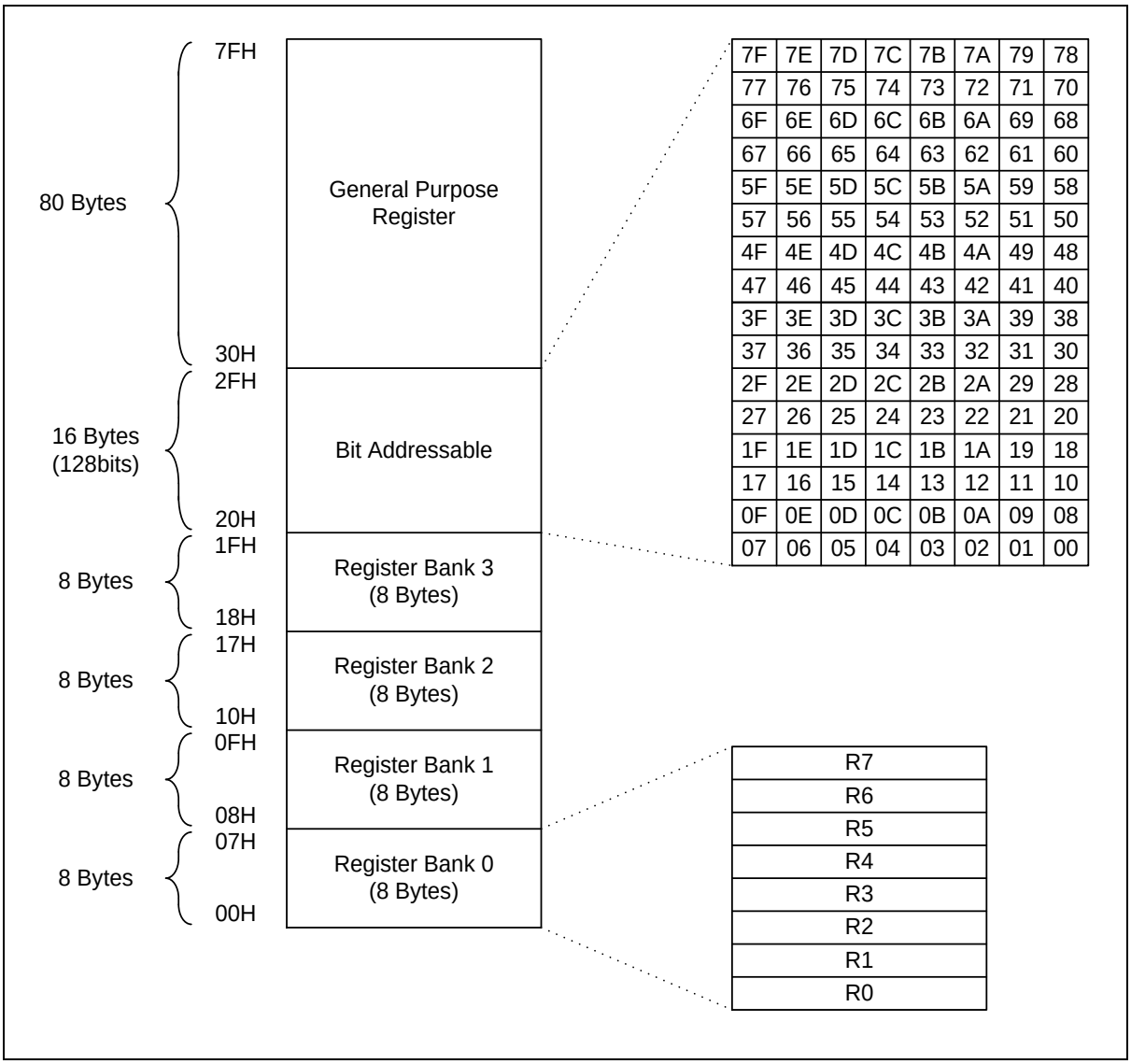



**Figure 8. Lower 128 bytes Internal RAM**

## 4.3 Extended SFR area

Extended SFR area has no relation with RAM nor FLASH. This area can be read or written to by using SFR in 8-bit unit.

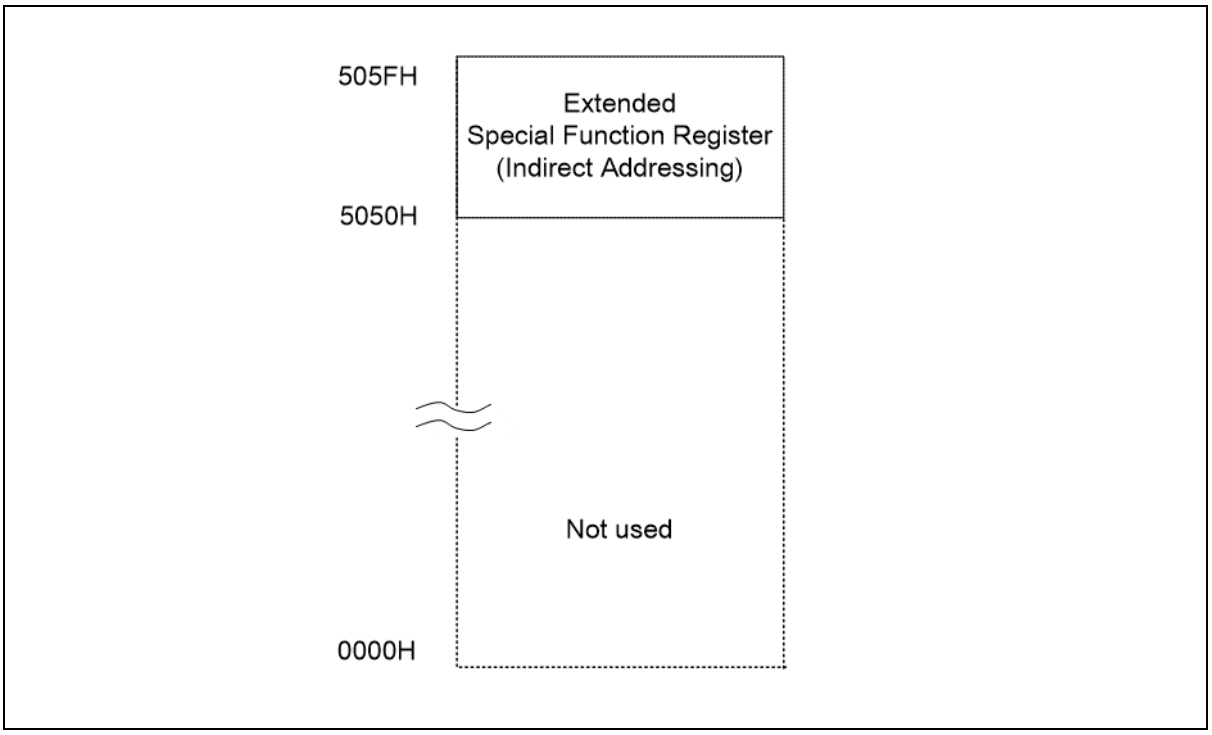



Figure 9. Extended SFR (XSFR) Area

## 4.4 EEPROM area

EEPROM area has no relation with RAM nor FLASH. This area can be read by using DPTR. EEPROM area can be erased or written to by using a buffer.

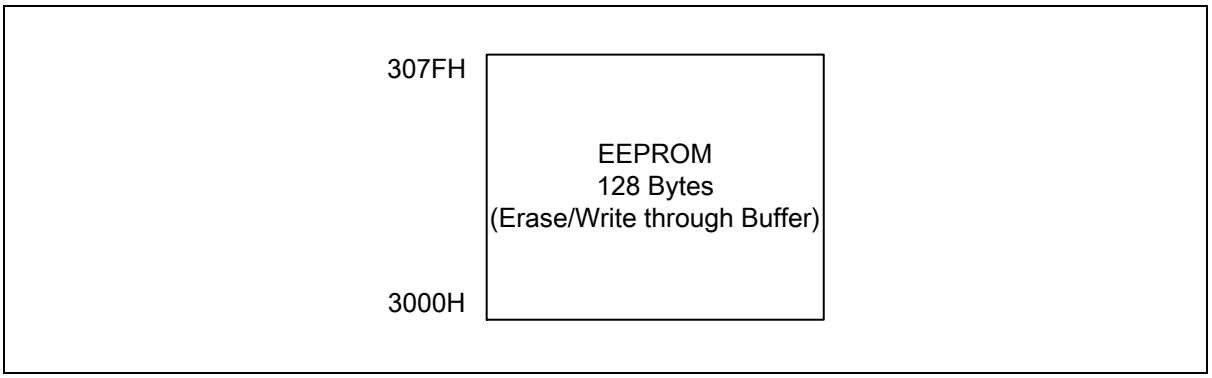



Figure 10. EEPROM Area

Detailed information about EEPROM, please refer to Chapter 21 EEPROM memory.

## 4.5 SFR map

In this section, information of SFR map and map summaries are introduced through tables 3 to 6.

### 4.5.1 SFR map summary

**Table 3. SFR Map Summary**

– Reserved  M8051 compatible

| | 00H/8H[NOTE] | 01H/9H | 02H/0AH | 03H/0BH | 04H/0CH | 05H/0DH | 06H/0EH | 07H/0FH |
|---|---|---|---|---|---|---|---|---|
| **0F8H** | IP1 | – | FSADRH | FSADRM | FSADRL | FIDR | FMCR | – |
| **0F0H** | B | – | EESADRL | EESADRH | EEIDR | EEMCR | – | – |
| **0E8H** | RSTFR | RXBLEN | TMINRL | TMINRH | TMAXRL | TMAXRH | TENDRL | TENDRH |
| **0E0H** | ACC | LIRXDR | TRXARL | TRXARH | – | ICSCR | ICSDR0 | ICSDR1 |
| **0D8H** | LVRCR | TXBLEN | TTXCRL | TTXCRH | TTXDRL | TTXDRH | TTXRRL | TTXRRH |
| **0D0H** | PSW | LITXDR | TTXARL | TTXARH | TTXBRL | TTXBRH | – | FCDIN |
| **0C8H** | OSCCR | LITXTINF | ADCCRL | ADCCRH | ADCDRL | *ADCDRH* | – | – |
| **0C0H** | LISTATR | LICR0 | LICR1 | LICR2 | LICAPL | LICAPH | TDLYRL | TDLYRH |
| **0B8H** | IP | – | T1CRL | T1CRH | T1ADRL | T1ADRH | T1BDRL | T1BDRH |
| **0B0H** | – | – | T0CRL | T0CRH | T0ADRL | T0ADRH | T0BDRL | T0BDRH |
| **0A8H** | IE | IE1 | IE2 | IE3 | – | CHPCR | AMP0CR | AMP1CR |
| **0A0H** | EIFLAG | – | EO | – | EIPOL0 | EIPOL1 | – | – |
| **98H** | – | P1IO | P1OD | P1PU | P1FSRL | P1FSRH | – | IRCIDR |
| **90H** | – | P0IO | P0OD | P0PU | P0FSRL | P0FSRH | P01DB | IRCTRM |
| **88H** | P1 | – | SCCR | BITCR | BITCNT | WDTCR | WDTDR/ WDTCNT | IRCTCR |
| **80H** | P0 | SP | DPL | DPH | DPL1 | DPH1 | – | PCON |

**NOTE**: Registers 00H/8H are bit-addressable.

### 4.5.2 Extended SFR map summary

**Table 4. XSFR Map Summary**

– Reserved

| | 00H/8H | 01H/9H | 02H/0AH | 03H/0BH | 04H/0CH | 05H/0DH | 06H/0EH | 07H/0FH |
|---|---|---|---|---|---|---|---|---|
| **5058H** | FCDRL | – | – | – | – | – | – | LVRIDR |
| **5050H** | FCSARH | FCEARH | FCSARM | FCEARM | FCSARL | FCEARL | FCCR | FCDRH |
| **…** | – | – | – | – | – | – | – | – |
| **1010H** | DWMAT | DWBNDL | DWDECD | DWINCM | UPMAT | UPBNDL | UPINCD | UPDECM |
| **1008H** | SIRENCR | – | – | – | MAXDRL | MAXDRH | MINDRL | MINDRH |
| **1000H** | USTCR1 | USTCR2 | USTCR3 | USTST | USTBD | USTDR | – | – |

### 4.5.3 SFR map

Table 5. SFR Map

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 80H | P0 Data Register | P0 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 81H | Stack Pointer | SP | R/W | 0 | 0 | 0 | 0 | 0 | 1 | 1 | 1 |
| 82H | Data Pointer Register Low | DPL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 83H | Data Pointer Register High | DPH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 84H | Data Pointer Register Low 1 | DPL1 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 85H | Data Pointer Register High 1 | DPH1 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 86H | Reserved | – | – | – | | | | | | | |
| 87H | Power Control Register | PCON | R/W | – | – | – | – | – | – | 0 | 0 |
| 88H | P1 Data Register | P1 | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |
| 89H | Reserved | – | – | – | | | | | | | |
| 8AH | System and Clock Control Register | SCCR | R | – | – | – | – | – | – | 0 | 0 |
| 8BH | Basic Interval Timer Control Register | BITCR | R/W | 0 | 0 | 0 | – | 0 | 0 | 0 | 1 |
| 8CH | Basic Interval Timer Counter Register | BITCNT | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8DH | Watch Dog Timer Control Register | WDTCR | R/W | 0 | 0 | 0 | – | – | – | 0 | 0 |
| 8EH | Watch Dog Timer Data Register | WDTDR | W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 8EH | Watch Dog Timer Counter Register | WDTCNT | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8FH | Internal RC Trim Control Register | IRCTCR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 90H | Reserved | – | – | – | | | | | | | |
| 91H | P0 Direction Register | P0IO | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 92H | P0 Open-drain Selection Register | P0OD | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 93H | P0 Pull-up Resistor Selection Register | P0PU | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 94H | Port 0 Function Selection Low Register | P0FSRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 95H | Port 0 Function Selection High Register | P0FSRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 96H | P0/P1 Debounce Enable Register | P01DB | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 97H | Internal RC Trim Register | IRCTRM | R/W | x | x | x | x | x | x | x | x |
| 98H | Reserved | – | – | – | | | | | | | |
| 99H | P1 Direction Register | P1IO | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |
| 9AH | P1 Open-drain Selection Register | P1OD | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |
| 9BH | P1 Pull-up Resistor Selection Register | P1PU | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |
| 9CH | Port 1 Function Selection Low Register | P1FSRL | R/W | – | 0 | – | 0 | – | 0 | 0 | 0 |
| 9DH | Port 1 Function Selection High Register | P1FSRH | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| 9EH | Reserved | – | – | – | | | | | | | |
| 9FH | Internal RC Trim Identification Register | IRCIDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A0H | External Interrupt Flag Register | EIFLAG | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |
| A1H | Reserved | – | – | – | | | | | | | |
| A2H | Extended Operation Register | EO | R/W | – | – | – | 0 | – | 0 | 0 | 0 |
| A3H | Reserved | – | – | – | | | | | | | |
| A4H | External Interrupt Polarity 0 Register | EIPOL0 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| A5H | External Interrupt Polarity 1 Register | EIPOL1 | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| A6H | Reserved | – | – | – | | | | | | | |
| A7H | Reserved | – | – | – | | | | | | | |
| A8H | Interrupt Enable Register | IE | R/W | 0 | – | 0 | 0 | 0 | 0 | 0 | 0 |
| A9H | Interrupt Enable Register 1 | IE1 | R/W | – | – | – | – | 0 | 0 | – | 0 |
| AAH | Interrupt Enable Register 2 | IE2 | R/W | – | – | 0 | 0 | – | 0 | 0 | 0 |
| ABH | Interrupt Enable Register 3 | IE3 | R/W | – | – | – | 0 | 0 | – | – | – |
| ACH | Reserved | – | – | – | | | | | | | |
| ADH | Chopper Control Register | CHPCR | R/W | – | – | – | – | – | – | 0 | 0 |
| AEH | OP-AMP Control Register 0 | AMPCR0 | R/W | – | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| AFH | OP-AMP Control Register 1 | AMPCR1 | R/W | 0 | 0 | 0 | 0 | 0 | – | 0 | 0 |
| B0H | Reserved | – | – | – | | | | | | | |
| B1H | Reserved | – | – | – | | | | | | | |
| B2H | Timer 0 Control Low Register | T0CRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| B3H | Timer 0 Control High Register | T0CRH | R/W | 0 | – | 0 | 0 | – | 0 | – | 0 |
| B4H | Timer 0 A Data Low Register | T0ADRL | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B5H | Timer 0 A Data High Register | T0ADRH | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B6H | Timer 0 B Data Low Register | T0BDRL | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B7H | Timer 0 B Data High Register | T0BDRH | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| B8H | Interrupt Priority Register | IP | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |
| B9H | Reserved | – | – | – | | | | | | | |
| BAH | Timer 1 Control Low Register | T1CRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| BBH | Timer 1 Control High Register | T1CRH | R/W | 0 | – | 0 | 0 | – | – | – | 0 |
| BCH | Timer 1 A Data Low Register | T1ADRL | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| BDH | Timer 1 A Data High Register | T1ADRH | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| BEH | Timer 1 B Data Low Register | T1BDRL | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| BFH | Timer 1 B Data High Register | T1BDRH | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| C0H | Line Interface Status Register | LISTATR | R/W | – | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C1H | Line Interface Control Register 0 | LICR0 | R/W | 0 | 0 | 0 | – | 0 | 0 | 0 | 0 |
| C2H | Line Interface Control Register 1 | LICR1 | R/W | – | – | – | – | – | 0 | 0 | 0 |
| C3H | Line Interface Control Register 2 | LICR2 | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| C4H | Line Interface Capture Data Low Register | LICAPL | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C5H | Line Interface Capture Data High Register | LICAPH | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C6H | Delay Time Data Low Register | TDLYRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C7H | Delay Time Data High Register | TDLYRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| C8H | Oscillator Control Register | OSCCR | R/W | – | – | – | 1 | 1 | – | – | – |
| C9H | Line Interface Transmit Toggle Information Register | LITXTINF | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| CAH | A/D Converter Control Low Register | ADCCRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| CBH | A/D Converter Control High Register | ADCCRH | R/W | 0 | – | – | – | 0 | 0 | 0 | 0 |
| CCH | A/D Converter Data Low Register | ADCDRL | R | x | x | x | x | x | x | x | x |
| CDH | A/D Converter Data High Register | ADCDRH | R | x | x | x | x | x | x | x | x |

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| CEH | Reserved | – | – | – | | | | | | | |
| CFH | Reserved | – | – | – | | | | | | | |
| D0H | Program Status Word Register | PSW | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D1H | Line Interface Transmit Data Register | LITXDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D2H | Transmit Time A Data Low Register | TTXARL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D3H | Transmit Time A Data High Register | TTXARH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D4H | Transmit Time B Data Low Register | TTXBRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D5H | Transmit Time B Data High Register | TTXBRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D6H | Reserved | – | – | – | | | | | | | |
| D7H | Flash CRC Data In Register | FCDIN | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| D8H | Low Voltage Reset Control Register | LVRCR | R/W | 0 | – | – | – | – | 0 | 0 | 0 |
| D9H | Transmit bits Length Counter | TXBLEN | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DAH | Transmit Time C Data Low Register | TTXCRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DBH | Transmit Time C Data High Register | TTXCRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DCH | Transmit Time D Data Low Register | TTXDRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DDH | Transmit Time D Data High Register | TTXDRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DEH | Transmit Time Rx Data Low Register | TTXRRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| DFH | Transmit Time Rx Data High Register | TTXRRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E0H | Accumulator Register | ACC | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E1H | Line Interface Receive Data Register | LIRXDR | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| E2H | Receive Time A Data Low Register | TRXARL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E3H | Receive Time A Data High Register | TRXARH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| E4H | Reserved | – | – | – | | | | | | | |
| E5H | Constant Sink Current Control Register | ICSCR | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| E6H | Constant Sink Current Data Register 0 | ICSDR0 | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| E7H | Constant Sink Current Data Register 1 | ICSDR1 | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| E8H | Reset Flag Register | RSTFR | R/W | 1 | x | 0 | 0 | x | – | – | – |
| E9H | Receive bits Length Counter | RXBLEN | R | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| EAH | Minimum Time Data Low Register | TMINRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| EBH | Minimum Time Data High Register | TMINRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| ECH | Maximum Time Data Low Register | TMAXRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| EDH | Maximum Time Data High Register | TMAXRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| EEH | End Time Data Low Register | TENDRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| EFH | End Time Data High Register | TENDRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F0H | B Register | B | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F1H | Reserved | – | – | – | | | | | | | |
| F2H | EEPROM Sector Address Low Register | EESADRL | R/W | 0 | 0 | 0 | 0 | – | – | – | – |
| F3H | EEPROM Sector Address High Register | EESADRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F4H | EEPROM Identification Register | EEIDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| F5H | EEPROM Mode Control Register | EEMCR | R/W | 0 | – | – | – | – | 0 | 0 | 0 |
| F6H | Reserved | – | – | – | | | | | | | |
| F7H | Reserved | – | – | – | | | | | | | |
| F8H | Interrupt Priority Register 1 | IP1 | R/W | – | – | 0 | 0 | 0 | 0 | 0 | 0 |

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| F9H | Reserved | - | – | – | | | | | | | |
| FAH | Flash Sector Address High Register | FSADRH | R/W | – | – | – | – | 0 | 0 | 0 | 0 |
| FBH | Flash Sector Address Middle Register | FSADRM | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| FCH | Flash Sector Address Low Register | FSADRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| FDH | Flash Identification Register | FIDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| FEH | Flash Mode Control Register | FMCR | R/W | 0 | – | – | – | – | 0 | 0 | 0 |
| FFH | Reserved | - | – | – | | | | | | | |

### 4.5.4 Extended SFR map

Table 6. XSFR Map

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 1000H | USART Control Register 1 | USTCR1 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1001H | USART Control Register 2 | USTCR2 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1002H | USART Control Register 3 | USTCR3 | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1003H | USART Status Register | USTST | R/W | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1004H | USART Baud Rate Generation Register | USTBD | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 1005H | USART Data Register | USTDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1006H | Reserved | – | – | – | | | | | | | |
| 1007H | Reserved | – | – | – | | | | | | | |
| 1008H | Siren Control Register | SIRENCR | R/W | – | – | 0 | 0 | – | 0 | – | 0 |
| - - - - - - - - - - - - - - - - - - - - | | | | | | | | | | | |
| 100CH | Siren Max Data Low Register | MAXDRL | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 100DH | Siren Max Data High Register | MAXDRH | R/W | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 100EH | Siren Min Data Low Register | MINDRL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 100FH | Siren Min Data High Register | MINDRH | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1010H | Siren down match times register | DWMAT | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1011H | Siren down bundle times register | DWBNDL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1012H | Siren down decrement data register | DWDECD | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1013H | Siren down increment match times register | DWINCM | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1014H | Siren up match times register | UPMAT | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1015H | Siren up bundle times register | UPBNDL | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1016H | Siren up increment data register | UPINCD | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1017H | Siren up decrement match times register | UPDECM | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| - - - - - - - - - - - - - - - - - - - - | | | | | | | | | | | |
| 5050H | Flash CRC Start Address High Register | FCSARH | R/W | – | – | – | – | – | – | – | 0 |
| 5051H | Flash CRC End Address High Register | FCEARH | R/W | – | – | – | – | – | – | – | 0 |
| 5052H | Flash CRC Start Address Middle Register | FCSARM | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

| Address | Function | Symbol | R/W | @ Reset | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 5053H | Flash CRC End Address Middle Register | FCEARM | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5054H | Flash CRC Start Address Low Register | FCSARL | R/W | 0 | 0 | 0 | 0 | – | – | – | – |
| 5055H | Flash CRC End Address Low Register | FCEARL | R/W | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| 5056H | Flash CRC Control Register | FCCR | R/W | 0 | 0 | 0 | – | 0 | 0 | 0 | 0 |
| 5057H | Flash CRC Data High Register | FCDRH | R | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 5058H | Flash CRC Data Low Register | FCDRL | R | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| - - - - - - - - - - - - - - - - - - - - - - | | | | | | | | | | | |
| 505FH | LVR Write Identification Register | LVRIDR | R/W | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

### 4.5.5 SFR map

**ACC (Accumulator Register): E0H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ACC | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| ACC | Accumulator |

**B (B Register): F0H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| B | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| B | B Register |

**SP (Stack Pointer): 81H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| SP | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 07H

| | |
|---|---|
| SP | Stack Pointer |

### DPL (Data Pointer Register Low): 82H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DPL | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| DPL | Data Pointer Low |

### DPH (Data Pointer Register High): 83H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DPH | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| DPH | Data Pointer High |

### DPL1 (Data Pointer Register Low 1): 84H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DPL1 | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| DPL1 | Data Pointer Low 1 |

### DPH1 (Data Pointer Register High 1): 85H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DPH1 | | | | | | | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| DPH1 | Data Pointer High 1 |

### PSW (Program Status Word Register): D0H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CY | AC | F0 | RS1 | RS0 | OV | F1 | P |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| CY | Carry Flag |
| AC | Auxiliary Carry Flag |
| F0 | General Purpose User-Definable Flag |
| RS1 | Register Bank Select bit 1 |
| RS0 | Register Bank Select bit 0 |
| OV | Overflow Flag |
| F1 | User-Definable Flag |
| P | Parity Flag. Set/Cleared by hardware each instruction cycle to indicate an odd/even number of '1' bits in the accumulator |

### EO (Extended Operation Register): A2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | TRAP_EN | – | DPSEL2 | DPSEL1 | DPSEL0 |
| – | – | – | R/W | – | R/W | R/W | R/W |

Initial value: 00H

TRAP_EN Select the Instruction **(Keep always '0')**.

0 Select MOVC @(DPTR++), A

1 Select Software TRAP Instruction

DPSEL[2:0] Select Banked Data Pointer Register

| DPSEL2 | DPSEL1 | SPSEL0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | DPTR0 |
| 0 | 0 | 1 | DPTR1 |
| Reserved | | | |

# 5 Ports

## 5.1 I/O ports

A96L322 has two groups of I/O ports, P0 and P1. Each port can be easily configured as an input pin, an output, or an internal pull up and open-drain pin by software. The port configuration pursues to meet various system configurations and design requirements. P0 and P1 have a function generating interrupts in accordance with a change of state of the pin.

## 5.2 Port registers

### 5.2.1 Data register (Px)

Data register (Px) is related to a bidirectional I/O port. If a port is configured as an output port, data can be written to the corresponding bit of the Px. If a port is configured as an input, data can be read from the corresponding bit of the Px.

### 5.2.2 Direction register (PxIO)

Each I/O pin can be used as an input or an output independently by setting a PxIO register. If a bit is cleared in this read/write register, the corresponding pin of Px will be an input. While setting bits in this register will configure the corresponding pins to output.

Most bits are cleared by a system reset, but some bits are set by the system reset.

### 5.2.3 Pull-up register selection register (PxPU)

On-chip pull-up resistors can be connected to I/O ports individually by configuring a pull-up resistor selection register (PxPU). Setting a PxPU register can enable or disable a pull-up resister of each port. If a certain bit in PxPU register is 1, a pull-up resister of the corresponding pin is enabled. While the bit is 0, the pull-up resister is disabled. All bits are cleared by a system reset.

### 5.2.4 Open-drain selection register (PxOD)

There are internal open-drain selection registers (PxOD) for P0. Setting a PxOD register can enable or disable an open-drain of each port.

Most ports become push-pull by a system reset, but some ports become open-drain by the system reset.

### 5.2.5 Debounce enable register (P0DB)

P00, P01, P06, P07, P11, and P14 support a debounce function. Debounce clocks of the ports are fx/1, fx/4, fx/16, and fx/64 respectively.

### 5.2.6 Port function selection register (P0FSRH, P0FSRL, P1FSRH, P1FSRL)

Port function selection registers define alternative functions of ports. Please remember that these registers must be set properly for alternative port functions. A reset clears the P0FSRH, P0FSRL, P1FSRH and P1FSRL register to '00H', which makes all pins to normal I/O ports.

### 5.2.7 Register map

Table 7. Port Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| P0 | 80H | R/W | 00H | P0 Data Register |
| P0IO | 91H | R/W | 00H | P0 Direction Register |
| P0OD | 92H | R/W | 00H | P0 Open-drain Selection Register |
| P0PU | 93H | R/W | 00H | P0 Pull-up Resistor Selection Register |
| P01DB | 96H | R/W | 00H | P0/P1 Debounce Enable Register |
| P0FSRH | 95H | R/W | 00H | P0 Function Selection High Register |
| P0FSRL | 94H | R/W | 00H | P0 Function Selection Low Register |
| P1 | 88H | R/W | 00H | P1 Data Register |
| P1IO | 99H | R/W | 00H | P1 Direction Register |
| P1OD | 9AH | R/W | 00H | P1 Open-drain Selection Register |
| P1PU | 9BH | R/W | 00H | P1 Pull-up Resistor Selection Register |
| P1FSRH | 9DH | R/W | 00H | Port 1 Function Selection High Register |
| P1FSRL | 9CH | R/W | 00H | Port 1 Function Selection Low Register |

## 5.3 Port P0

### 5.3.1 Port description of P0

As an 8-bit I/O port, P0 controls the following registers:

- P0 data register (P0)
- P0 direction register (P0IO)
- debounce enable register (P01DB)
- P0 pull-up resistor selection register (P0PU)
- P0 open-drain selection register (P0OD)

For detailed information of P0 function selection, please refer to Port function selection registers.

### 5.3.2 Register description of P0

**P0 (P0 Data Register): 80H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P07 | P06 | P05 | P04 | P03 | P02 | P01 | P00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P0[7:0] I/O Data

**P0IO (P0 Direction Register): 91H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P07IO | P06IO | P05IO | P04IO | P03IO | P02IO | P01IO | P00IO |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P0IO[7:0] P0 Data I/O Direction.
0 Input
1 Output
**NOTE:** EINT0/EINT1/EINT2/EINT3 function possible when input

**P0PU (P0 Pull-up Resistor Selection Register): 93H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P07PU | P06PU | P05PU | P04PU | P03PU | P02PU | P01PU | P00PU |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P0PU[7:0] Configure Pull-up Resistor of P0 Port
0 Disable
1 Enable

**P0OD (P0 Open-drain Selection Register): 92H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| P07OD | P06OD | P05OD | P04OD | P03OD | P02OD | P01OD | P00OD |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P0OD[7:0] Configure Open-drain of P0 Port
0 Push-pull output
1 Open-drain output

### P01DB (P0/P1 Debounce Enable Register): 96H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DBCLK1 | DBCLK0 | P14DB | P11DB | P07DB | P06DB | P01DB | P00DB |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | | | |
|---|---|---|---|
| DBCLK[1:0] | Configure Debounce Clock of Port | | |
| | DBCLK1 | DBCLK0 | Description |
| | 0 | 0 | fx/1 |
| | 0 | 1 | fx/4 |
| | 1 | 0 | fx/16 |
| | 1 | 1 | fx/64 |
| P14DB | Configure Debounce of P14 Port | | |
| | 0 | Disable | |
| | 1 | Enable | |
| P11DB | Configure Debounce of P11 Port | | |
| | 0 | Disable | |
| | 1 | Enable | |
| P07DB | Configure Debounce of P07 Port | | |
| | 0 | Disable | |
| | 1 | Enable | |
| P06DB | Configure Debounce of P06 Port | | |
| | 0 | Disable | |
| | 1 | Enable | |
| P01DB | Configure Debounce of P01 Port | | |
| | 0 | Disable | |
| | 1 | Enable | |
| P00DB | Configure Debounce of P00 Port | | |
| | 0 | Disable | |
| | 1 | Enable | |

**NOTES**:

1. If the same level is not detected on enabled pin three or four times in a row at the sampling clock, the signal is eliminated as noise.
2. A pulse level should be input for the duration of 3 clock or more to be actually detected as a valid edge.
3. The port debounce is automatically disabled at stop mode and recovered after stop mode release.

**P0FSRH (Port 0 Function Selection High Register): 95H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PFSRH07 | PFSRH06 | PFSRH05 | PFSRH04 | PFSRH03 | PFSRH02 | PFSRH01 | PFSRH00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

PFSRH0[7:6] P07 Function select

| PFSRH07 | PFSRH06 | Description |
|---|---|---|
| 0 | 0 | I/O Port (EINT3 function possible when input) |
| 0 | 1 | RXD/MISO Function |
| 1 | 0 | AN7 Function |
| 1 | 1 | ICS1 Function |

PFSRH0[5:4] P06 Function select

| PFSRH05 | PFSRH04 | Description |
|---|---|---|
| 0 | 0 | I/O Port (EINT2 function possible when input) |
| 0 | 1 | TXD/MOSI Function |
| 1 | 0 | AN6 Function |
| 1 | 1 | ICS0 Function |

PFSRH0[3:2] P05 Function Select

| PFSRH03 | PFSRH02 | Description |
|---|---|---|
| 0 | 0 | I/O Port |
| 0 | 1 | OP0P Function |
| 1 | 0 | AN5 Function |
| 1 | 1 | Not used |

PFSRH0[1:0] P04 Function Select

| PFSRH01 | PFSRH00 | Description |
|---|---|---|
| 0 | 0 | I/O Port |
| 0 | 1 | OP0N Function |
| 1 | 0 | AN4 Function |
| 1 | 1 | Not used |

**NOTE**: If OP-AMP0 is used, the P04 and P05 pins must be set to OP0N and OP0P functions regardless of using internal or external gain resistors.

**P0FSRL (Port 0 Function Selection Low Register): 94H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PFSRL07 | PFSRL06 | PFSRL05 | PFSRL04 | PFSRL03 | PFSRL02 | PFSRL01 | PFSRL00 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

PFSRL0[7:6] P03 Function select

| PFSRL07 | PFSRL06 | Description |
|---|---|---|
| 0 | 0 | I/O Port |
| 0 | 1 | OP0OUT Function |
| 1 | 0 | AN3 Function |
| 1 | 1 | T1O/PWM1O |

PFSRL0[5:4] P02 Function Select

| PFSRL05 | PFSRL04 | Description |
|---|---|---|
| 0 | 0 | I/O Port |
| 0 | 1 | OP1P Function |
| 1 | 0 | AN2 Function |
| 1 | 1 | T0O/PWM0O |

PFSRL0[3:2] P01 Function select

| PFSRL03 | PFSRL02 | Description |
|---|---|---|
| 0 | 0 | I/O Port (EINT1 function possible when input) |
| 0 | 1 | OP1N Function |
| 1 | 0 | AN1 Function |
| 1 | 1 | Not used |

PFSRL0[1:0] P00 Function select

| PFSRL01 | PFSRL00 | Description |
|---|---|---|
| 0 | 0 | I/O Port (EINT0 function possible when input) |
| 0 | 1 | OP1OUT Function |
| 1 | 0 | AN0 Function |
| 1 | 1 | Not used |

**NOTES**:

1. If OP-AMP0 is used, the P03 pin must be set to OP0OUT function regardless of using internal or external gain resistors.
2. If OP-AMP1 is used, the P00, P01, and P02 pins must be set to OP1OUT, OP1N, and OP1P functions regardless of using internal or external gain resistors.

## 5.4 Port P1

### 5.4.1 Port description of P1

As a 6-bit I/O port, P1 controls the following registers:

- P1 data register (P1)
- P1 direction register (P1IO)
- P1 pull-up resistor selection register (P1PU)
- P1 open-drain selection register (P1OD)

For detailed information of P1 function selection, please refer to Port function selection registers.

### 5.4.2 Register description of P1

**P1 (P1 Data Register): 88H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | P15 | P14 | P13 | P12 | P11 | P10 |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P1[5:0] I/O Data

**P1IO (P1 Direction Register): 99H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | P15IO | P14IO | P13IO | P12IO | P11IO | P10IO |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P1IO[5:0] P1 Data I/O Direction.
0 Input
1 Output
**NOTE:** EINT10/EINT11/EC0/EC1/RXI/SS function possible when input

**P1PU (P1 Pull-up Resistor Selection Register): 9BH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | P15PU | P14PU | P13PU | P12PU | P11PU | P10PU |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P1PU[5:0] Configure Pull-up Resistor of P1 Port
0 Disable
1 Enable

**P1OD (P1 Open-drain Selection Register): 9AH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | P15OD | P14OD | P13OD | P12OD | P11OD | P10OD |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

P1OD[5:0] Configure Open-drain of P1 Port
0 Push-pull output
1 Open-drain output

### P1FSRH (Port 1 Function Selection High Register): 9DH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | PFSRH13 | PFSRH12 | PFSRH11 | PFSRH10 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

PFSRH1[3:2] P15 Function select

| PFSR13 | PFSR12 | Description |
|---|---|---|
| 0 | 0 | I/O Port |
| 0 | 1 | AVREF Function |
| 1 | 0 | AN8 Function |
| 1 | 1 | Not used |

PFSRH1[1:0] P14 Function select

| PFSR11 | PFSR10 | Description |
|---|---|---|
| 0 | 0 | I/O Port (EINT11 function possible when input) |
| 0 | 1 | T1O/PWM1O Function |
| 1 | 0 | TXI Function |
| 1 | 1 | Not used |

### P1FSRL (Port 1 Function Selection Low Register): 9CH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | PFSRL16 | – | PFSRL14 | – | PFSRL12 | PFSRL11 | PFSRL10 |
| – | R/W | – | R/W | – | R/W | R/W | R/W |

Initial value: 00H

PFSRL16 P13 Function select

| | |
|---|---|
| 0 | I/O Port (EC1 function possible when input) |
| 1 | RXD/MISO Function |

PFSRL14 P12 Function select

| | |
|---|---|
| 0 | I/O Port (EC0/SS function possible when input) |
| 1 | TXD/MOSI Function |

PFSRL12 P11 Function select

| | |
|---|---|
| 0 | I/O Port (EINT10/RXI function possible when input) |
| 1 | T0O/PWM0O Function |

PFSRL1[1:0] P10 Function select

| PFSRL11 | PFSRL10 | Description |
|---|---|---|
| 0 | 0 | I/O Port |
| 0 | 1 | SCK Function |
| 1 | 0 | T1O/PWM1O Function |
| 1 | 1 | Not used |

**NOTE:** For more information settings of P10/RESETB, please refer to **Appendix: Configure option**.

# 6 Interrupt controller

Up to 16 interrupt sources are available in the A96L322. Allowing software control, each interrupt source can be enabled by defining separate enable register bit associated with it. It can also have four levels of priority assigned. The non-maskable interrupt source is always enabled with a higher priority than any other interrupt sources, and is not controllable by software.

The interrupt controller features the followings:

- Receives requests from 16 interrupt sources
- 6 group priority
- 4 priority levels
- Multi interrupt possibility
- If requests of different priority levels are received simultaneously, a request with higher priority level is served first.
- Each interrupt source can be controlled by an EA bit and an IEx bit
- Interrupt latency varies ranging from 3 to 9 machine cycles in a single interrupt system.

Non-maskable interrupt is always enabled, while maskable interrupts can be enabled through four pairs of interrupt enable registers (IE, IE1, IE2, IE3). Each bit of the four registers can individually enable or disable a particular interrupt source. Especially bit 7 (EA) in the register IE provides overall control. It must be set to '1' to enable interrupts as introduced in the followings:

- When EA is set to '0' ➔ all interrupts are disabled.
- When EA is set to '1' ➔ a particular interrupt can be individually enabled or disabled by the associate bit of the interrupt enable registers.

EA is always cleared to '0' jumping to an interrupt service vector and set to '1' executing the [RETI] instruction. A96L322 supports a four-level priority scheme. Each maskable interrupt is individually assigned to one of the four levels according to IP and IP1.

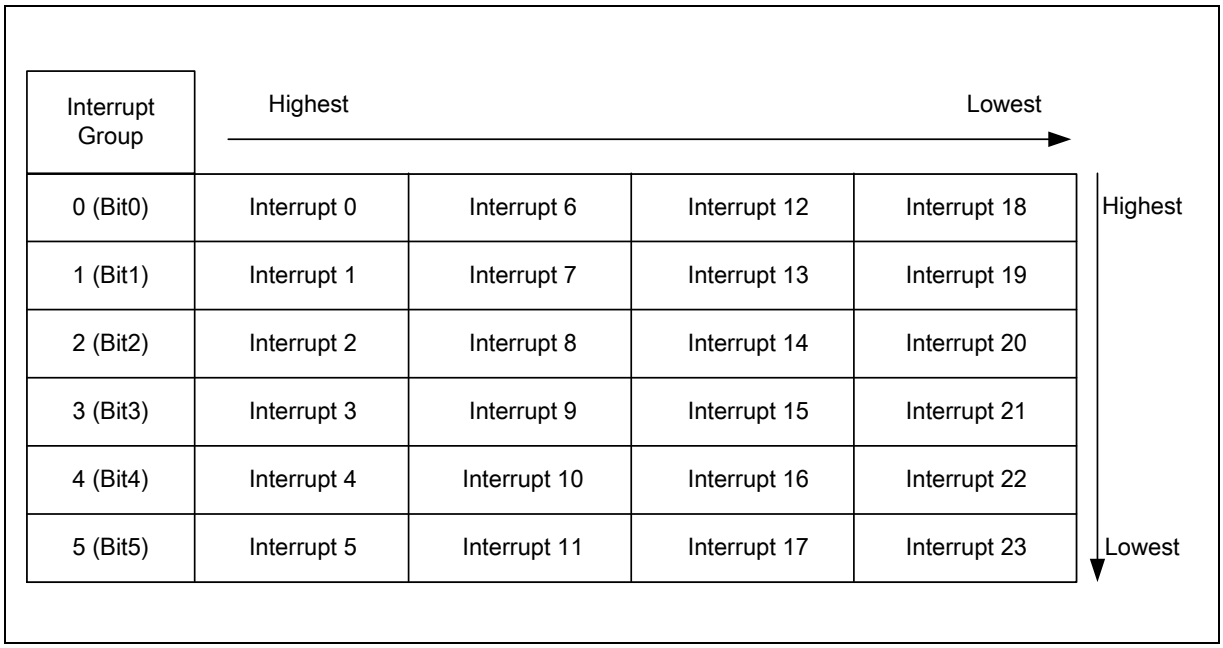



**Figure 11. Interrupt Group Priority Level**

Figure 11 introduces interrupt groups and their priority levels that is available for sharing interrupt priority. Priority of a group is set by 2 bits of Interrupt Priority (IP) registers: 1 bit from IP and another 1 bit from IP1.

Interrupt Service Routine serves an interrupt having higher priority first. If two requests of different priority levels are received simultaneously, the request with higher priority level is served prior to the lower one.

## 6.1 External interrupt

External interrupts on pins of INT0 to INT5 receive various interrupt requests in accordance with the external interrupt polarity 0 register (EIPOL0) and external interrupt polarity 1 register (EIPOL1) as shown in figure 12. Each external interrupt source has enable/disable bits. An external interrupt flag register (EIFLAG) provides status of the external interrupts.

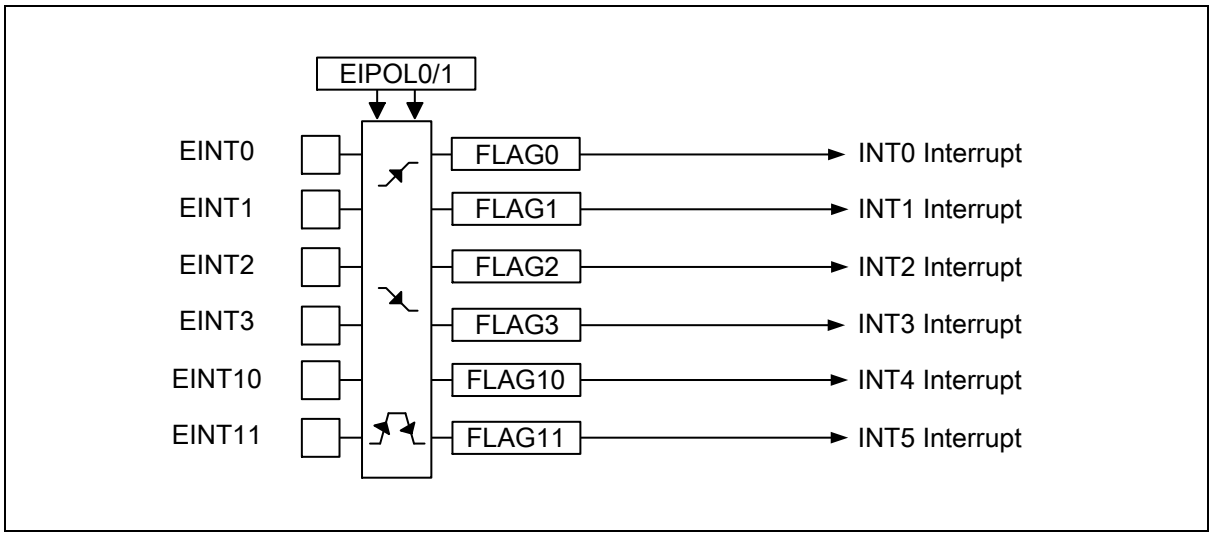



**Figure 12. External Interrupt Description**

## 6.2 Interrupt controller block diagram

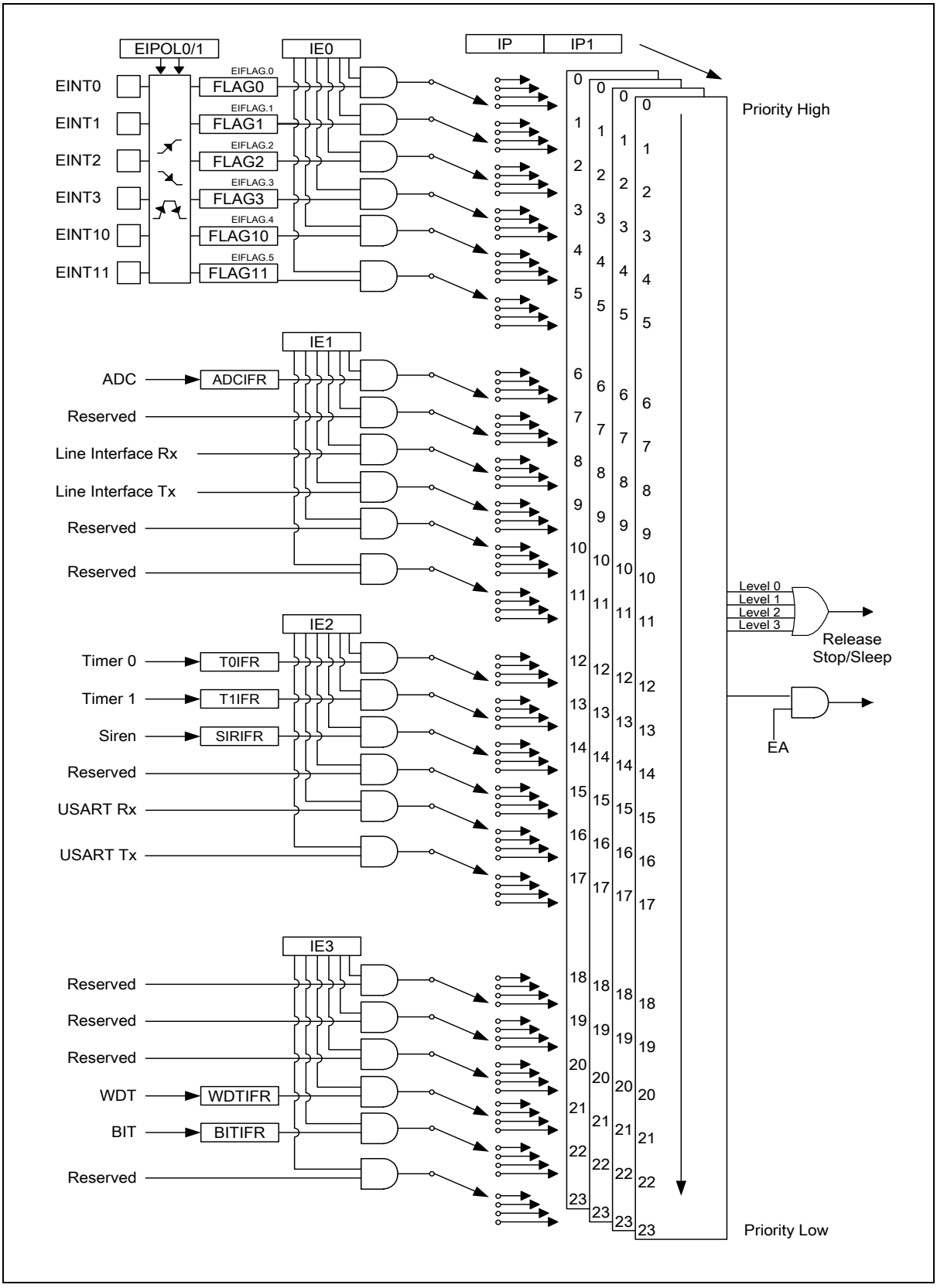



Figure 13. Interrupt Controller Block Diagram

In figure 13, release signal for STOP and IDLE mode can be generated by all interrupt sources which are enabled without reference to priority level. An interrupt request will be delayed while data is written to one of the registers IE, IE1, IE2, IE3, IP, IP1, and PCON.

## 6.3 Interrupt vector table

When a certain interrupt occurs, a LCALL (Long Call) instruction pushes the contents of the PC (Program Counter) onto the stack, and loads the appropriate vector address. CPU pauses from its current task for some time and processes the interrupt at the vector address.

Interrupt controller supports 24 interrupt sources and each interrupt source has a determined priority order as shown in table 8.

**Table 8. Interrupt Vector Address Table**

| Interrupt source | Symbol | Interrupt Enable Bit | Priority | Mask | Vector address |
|---|---|---|---|---|---|
| Hardware RESET | RESETB | - | 0 | Non-Maskable | 0000H |
| External Interrupt 0 | INT0 | IE.0 | 1 | Maskable | 0003H |
| External Interrupt 1 | INT1 | IE.1 | 2 | Maskable | 000BH |
| External Interrupt 2 | INT2 | IE.2 | 3 | Maskable | 0013H |
| External Interrupt 3 | INT3 | IE.3 | 4 | Maskable | 001BH |
| External Interrupt 10 | INT4 | IE.4 | 5 | Maskable | 0023H |
| External Interrupt 11 | INT5 | IE.5 | 6 | Maskable | 002BH |
| ADC Interrupt | INT6 | IE1.0 | 7 | Maskable | 0033H |
| – | INT7 | IE1.1 | 8 | Maskable | 003BH |
| Line Interface Rx | INT8 | IE1.2 | 9 | Maskable | 0043H |
| Line Interface Tx | INT9 | IE1.3 | 10 | Maskable | 004BH |
| – | INT10 | IE1.4 | 11 | Maskable | 0053H |
| – | INT11 | IE1.5 | 12 | Maskable | 005BH |
| T0 Interrupt | INT12 | IE2.0 | 13 | Maskable | 0063H |
| T1 Interrupt | INT13 | IE2.1 | 14 | Maskable | 006BH |
| Siren Interrupt | INT14 | IE2.2 | 15 | Maskable | 0073H |
| – | INT15 | IE2.3 | 16 | Maskable | 007BH |
| USART Rx Interrupt | INT16 | IE2.4 | 17 | Maskable | 0083H |
| USART Tx Interrupt | INT17 | IE2.5 | 18 | Maskable | 008BH |
| – | INT18 | IE3.0 | 19 | Maskable | 0093H |
| – | INT19 | IE3.1 | 20 | Maskable | 009BH |
| – | INT20 | IE3.2 | 21 | Maskable | 00A3H |
| WDT Interrupt | INT21 | IE3.3 | 22 | Maskable | 00ABH |

Table 8. Interrupt Vector Address Table (continued)

| Interrupt source | Symbol | Interrupt Enable Bit | Priority | Mask | Vector address |
|---|---|---|---|---|---|
| BIT Interrupt | INT22 | IE3.4 | 23 | Maskable | 00B3H |
| – | INT23 | IE3.5 | 24 | Maskable | 00BBH |

To execute the maskable interrupts, both EA bit and a corresponding bit of IEx associated with a specific interrupt source must be set to '1'. When an interrupt request is received, a particular interrupt request flag is set to '1' and maintains its status until CPU accepts the interrupt. After the interrupt acceptance, the interrupt request flag will be cleared automatically.

## 6.4 Interrupt sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to '0' by a reset or an instruction. The interrupt acceptance always happens at the last cycle of the instruction process. So rather than fetching the current instruction, CPU executes internally LCALL instruction and saves a PC onto the stack.

To begin an ISR (Interrupt Service Routine), the interrupt controller uses a branch instruction LJMP (Long Jump). The interrupt controller gives address of LJMP instruction to CPU. Since the end of the execution of current instruction, it needs 3~9 machine cycles to go to the interrupt service routine. The interrupt service task is terminated by the interrupt return instruction [RETI]. Once an interrupt request is generated, the following process is performed.

Table 9 introduces LJMP example code.

Table 9. LJMP Description and Example Code

<table>
<tr><td>Instruction</td><td>LJMP</td></tr>
<tr><td>Example code</td><td>LJMP 4000H
<table>
<tr><th>Address</th><th>Data</th><th>Instruction</th></tr>
<tr><td>1280H</td><td>02</td><td>LJMP 4000H</td></tr>
<tr><td>1281H</td><td>40</td><td></td></tr>
<tr><td>1282H</td><td>00</td><td></td></tr>
<tr><td>1283H</td><td>E4</td><td>CLR A</td></tr>
<tr><td>⋮</td><td>⋮</td><td>⋮</td></tr>
<tr><td>4000H</td><td>00</td><td>NOP</td></tr>
<tr><td>4001H</td><td>23</td><td>RL A</td></tr>
<tr><td></td><td></td><td></td></tr>
</table>
NOTE:<br>After finishing LJMP, NOP located at the address 400H will be executed as the next instruction.</td></tr>
</table>

Figure 14 shows a flow diagram of an ISR process.

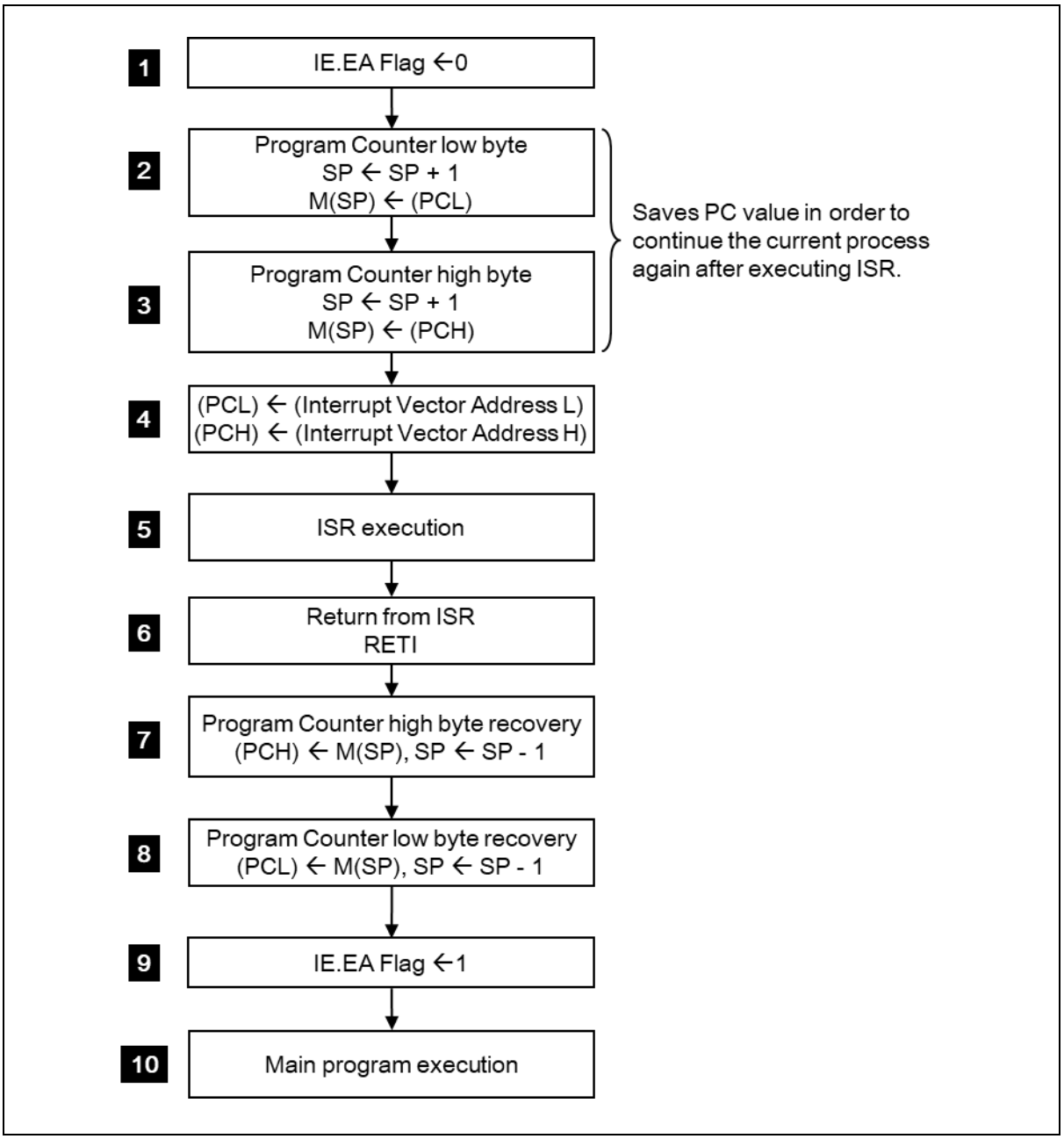



**Figure 14. Interrupt Sequence Flow**

## 6.5 Effective timing after controlling interrupt bit

Case A in figure 15 shows an effective time of Control Interrupt Enable Register (IE, IE1, IE2, IE3).

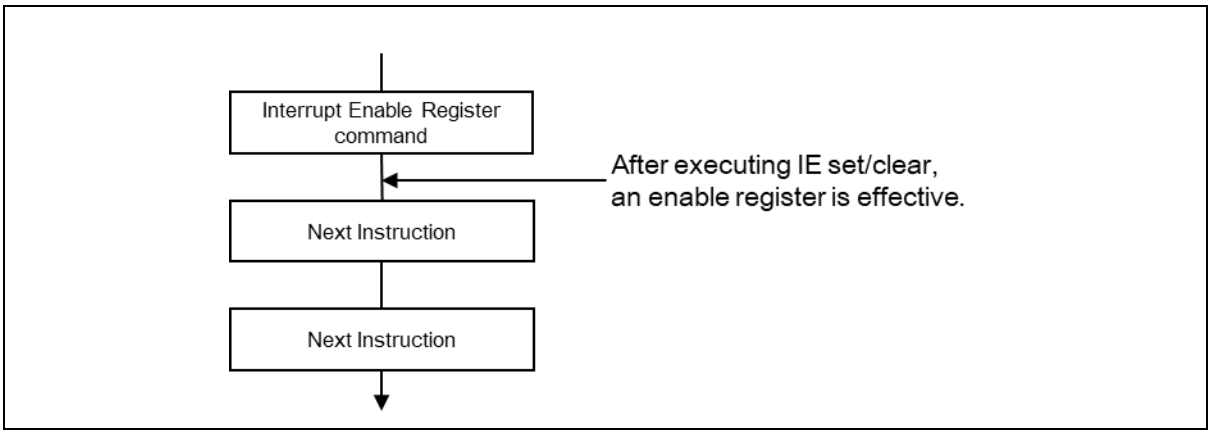



**Figure 15. Case A: Effective Timing of Interrupt Enable Register**

Case B in figure 16 shows an effective time of Interrupt Flag Register.

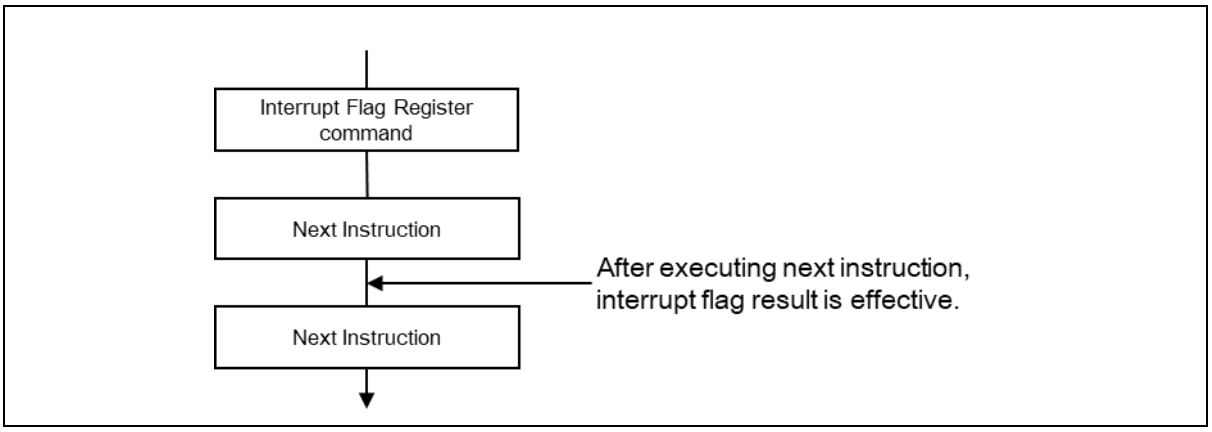



**Figure 16. Case B: Effective Timing of Interrupt Flag Register**

## 6.6 Multi interrupt

If two requests of different priority levels are received simultaneously, the request with higher priority level is served first. If more than one interrupt request are received, the interrupt polling sequence determines which request is served first by hardware. However, for special features, multi-interrupt processing can be executed by software.

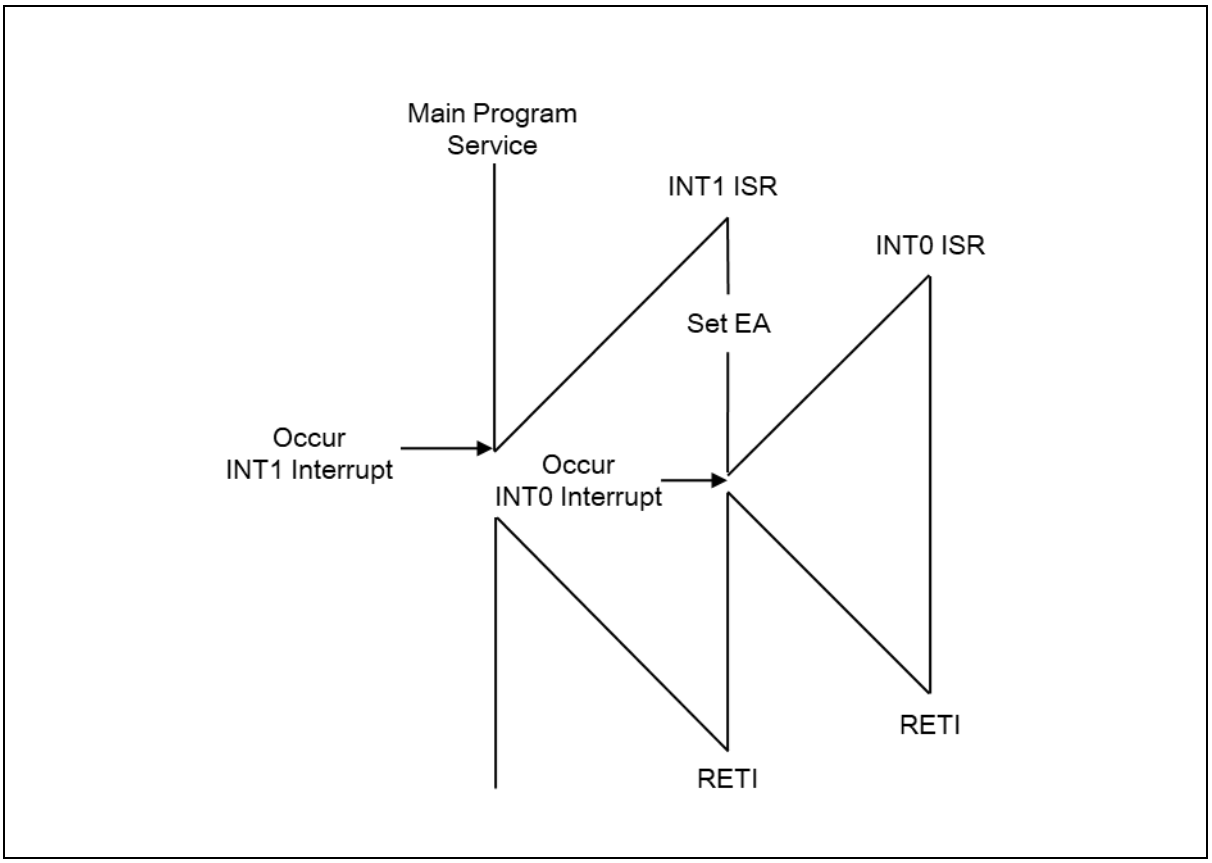



**Figure 17. Effective Timing of Multi Interrupt**

Figure 17 shows an example of multi-interrupt processing. While INT1 is served, INT0 which has higher priority than INT1 is occurred. Then INT0 is served immediately, then remain part of INT1 service routine is executed. If the priority level of INT0 is same or lower than INT1, INT0 will be served after the INT1 service has completed.

An interrupt service routine can be interrupted only by an interrupt with higher priority, and if two interrupts of different priority occur at the same time, the interrupt with higher priority level will be served first. An interrupt cannot be interrupted by another interrupt with the same or a lower priority level. If two interrupts having the same priority level occur simultaneously, the service order for those interrupts will be determined by the scan order.

## 6.7 Interrupt enable accept timing

Figure 18 implies that some period of time is required to response to the latched interrupt signal. In figure 18, 4 machine cycles will be taken for the processes of LCALL and LJMP.

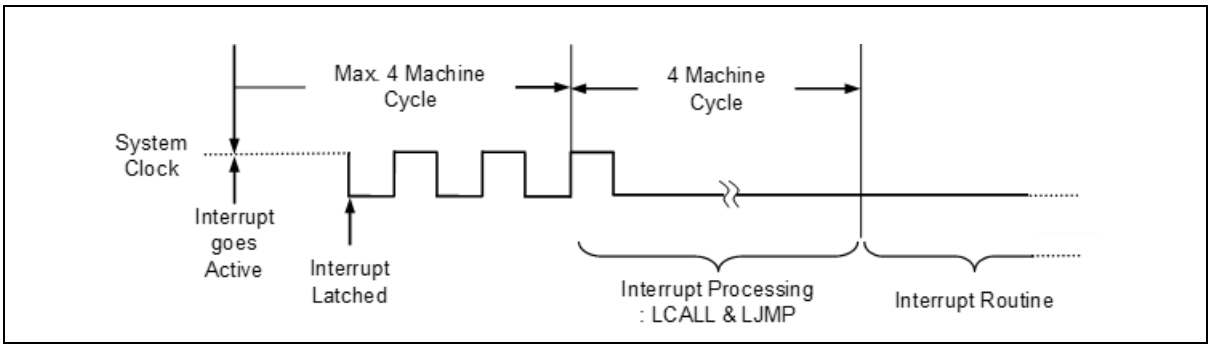



Figure 18. Interrupt Response Timing Diagram

## 6.8 Interrupt Service Routine Address

As seen in figure 19, ISR can be placed at any other location in program memory, and program memory must provide an unconditional jump to the starting address of ISR from the corresponding vector address.

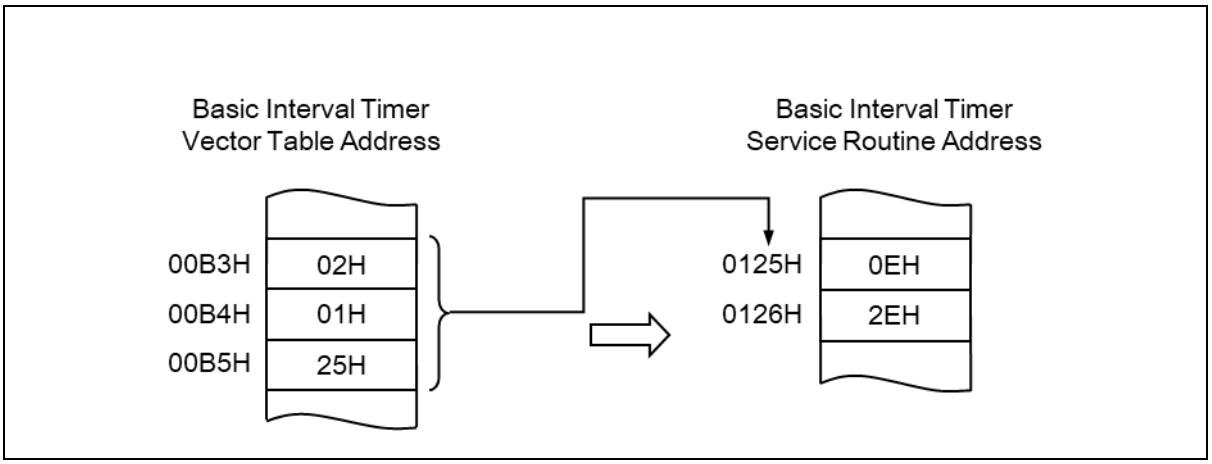



Figure 19. Correspondence between Vector Table Address and ISR Entry Address

## 6.9 Saving/ restore general-purpose registers

Let's assume there occurs an urgent condition. CPU needs to pause from its current task (Main Task in figure 20) for some time to execute something else (Interrupt Service Task in figure 20). After finishing the something else, CPU will return to the current task (Main Task).

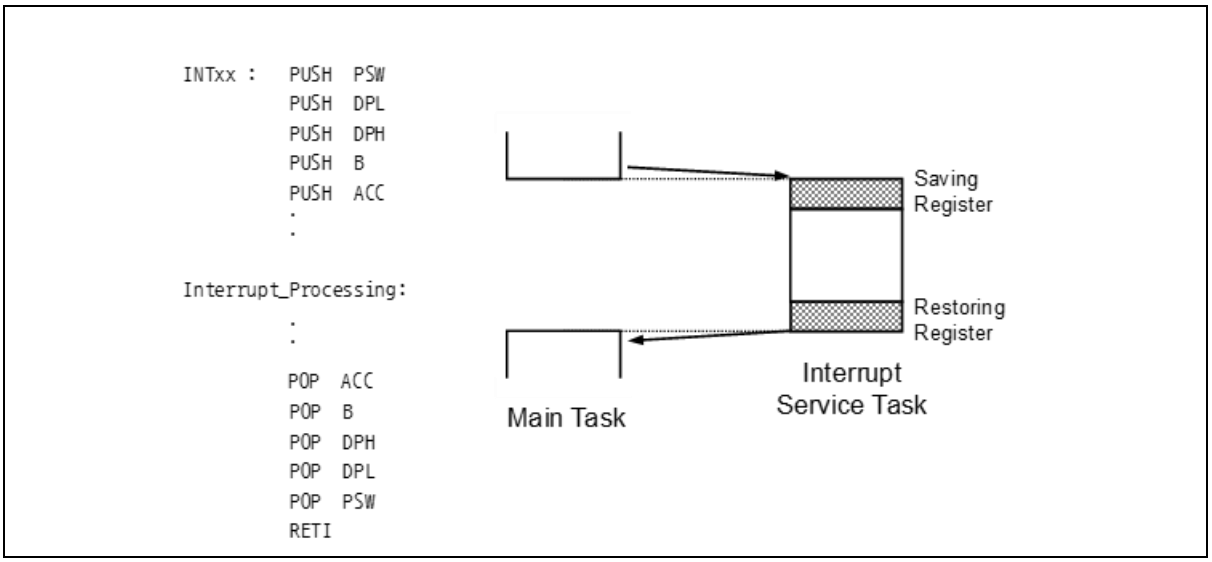



**Figure 20. Saving and Restore Process Diagram and Example Code**

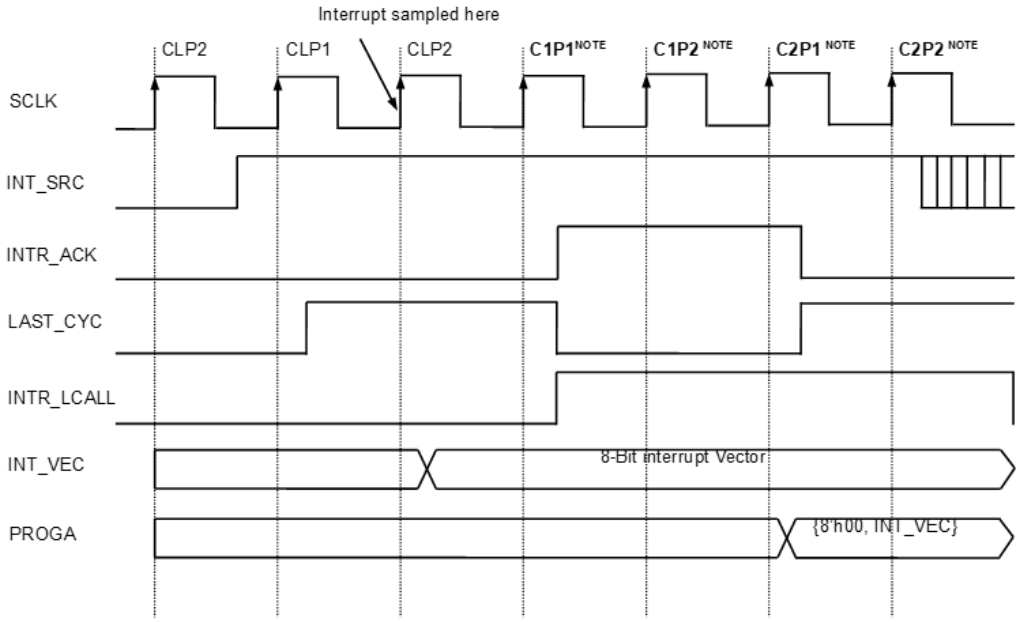
Example code in figure 20 performs the followings:

1. Interrupt INTXX occurs.
2. PUSH PSW: the SP is incremented by one, and the value of the specified byte operand is stored at the internal RAM address indirectly referenced by the SP.
3. PUSH DPL, PUSH DPH: PSW in memory stack by help of PUSH instruction.
4. CPU stores low
5. CPU pops the value of flag register and stores it in register H by help of POP Instruction.

## 6.10 Interrupt timing

As seen in figure 21 below, an interrupt source is sampled at the last cycle of a command. Upon the sampling, low 8-bit of interrupt vector is decided.

M8051W core makes the interrupt acknowledge at the first cycle of a command, executes LCALL instruction to jump interrupt routine at the address referenced by INT_VEC.




**NOTES**: Command cycle CLPx imply the followings:

1. L ➔ Last cycle
2. 1 ➔ 1st cycle or 1st phase
3. 2 ➔ 2nd cycle or 2nd phase

**Figure 21. Timing Chart of Interrupt Acceptance and Interrupt Return Instruction**

## 6.11 Interrupt register

Interrupt registers are memory space used to control interrupt functions. As shown in table 10, the interrupt registers consist of Interrupt Enable Registers, Interrupt Priority Registers, External Interrupt Flag Registers, and External Interrupt Polarity Register.

### 6.11.1 Interrupt Enable registers (IE, IE1, IE2, IE3)

Interrupt enable register consists of global interrupt control bit (EA) and peripheral interrupt control bits. Total 24 peripherals are able to control interrupts.

### 6.11.2 Interrupt Priority registers (IP, IP1)

24 interrupts are divided into 6 groups where each group has 4 interrupt sources respectively. A group can be assigned 4 levels of interrupt priority by using an interrupt priority register. Level 3 is the highest priority, while level 0 is the lowest priority. After a reset, IP and IP1 are cleared to '00H'. If interrupts have the same priority level, lower number interrupt is served first.

### 6.11.3 External Interrupt Flag register (EIFLAG)

External interrupt flag (EIFLAG) is set to '1' when the external interrupt generating condition is satisfied.

The flag is cleared when the interrupt service routine is executed. Alternatively, the flag can be cleared by writing '0' to it.

### 6.11.4 External Interrupt Polarity registers (EIPOL0, EIPOL1)

External interrupt polarity 0 register (EIPOL0) and external interrupt polarity 1 register (EIPOL1) determine one from rising edge, falling edge, and both edges for interrupt. No interrupt is at any edge by default.

### 6.11.5 Register map

Table 10. Interrupt Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| IE | A8H | R/W | 00H | Interrupt Enable Register |
| IE1 | A9H | R/W | 00H | Interrupt Enable Register 1 |
| IE2 | AAH | R/W | 00H | Interrupt Enable Register 2 |
| IE3 | ABH | R/W | 00H | Interrupt Enable Register 3 |
| IP | B8H | R/W | 00H | Interrupt Priority Register |
| IP1 | F8H | R/W | 00H | Interrupt Priority Register 1 |
| EIFLAG | A0H | R/W | 00H | External Interrupt Flag Register |
| EIPOL0 | A4H | R/W | 00H | External Interrupt Polarity 0 Register |
| EIPOL1 | A5H | R/W | 00H | External Interrupt Polarity 1 Register |

### 6.11.6 Interrupt register description

Interrupt registers are used to control interrupt functions. In addition to external interrupt control registers, these interrupt registers consist of interrupt enable register (IE), interrupt enable register 1 (IE1), interrupt enable register 2 (IE2) and interrupt enable register 3 (IE3).

For external interrupt, there are external interrupt flag register (EIFLAG), external interrupt polarity 0/1 registers (EIPOL0/1), and external interrupt flag register (EIFLAG).

### IE (Interrupt Enable Register): A8H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EA | – | INT5E | INT4E | INT3E | INT2E | INT1E | INT0E |
| R/W | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

EA Enable or Disable All Interrupt bits
0 All Interrupt disable
1 All Interrupt enable

INT5E Enable or Disable External Interrupt 11(EINT11)
0 Disable
1 Enable

INT4E Enable or Disable External Interrupt 10(EINT10)
0 Disable
1 Enable

INT3E Enable or Disable External Interrupt 3(EINT3)
0 Disable
1 Enable

INT2E Enable or Disable External Interrupt 2(EINT2)
0 Disable
1 Enable

INT1E Enable or Disable External Interrupt 1(EINT1)
0 Disable
1 Enable

INT0E Enable or Disable External Interrupt 0 (EINT0)
0 Disable
1 Enable

### IE1 (Interrupt Enable Register 1): A9H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| –- | – | – | – | INT9E | INT8E | – | INT6E |
| – | – | – | – | R/W | R/W | – | R/W |

Initial value: 00H

INT9E Enable or Disable Line Interface Tx interrupt
0 Disable
1 Enable

INT8E Enable or Disable Line Interface Rx interrupt
0 Disable
1 Enable

INT6E Enable or Disable ADC interrupt
0 Disable
1 Enable

### IE2 (Interrupt Enable Register 2): AAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | INT17E | INT16E | – | INT14E | INT13E | INT12E |
| – | – | R/W | R/W | – | R/W | R/W | R/W |

Initial value: 00H

| | | |
|---|---|---|
| INT17E | Enable or Disable USART Tx Interrupt | |
| | 0 | Disable |
| | 1 | Enable |
| INT16E | Enable or Disable USART Rx Interrupt | |
| | 0 | Disable |
| | 1 | Enable |
| INT14E | Enable or Disable Siren Interrupt | |
| | 0 | Disable |
| | 1 | Enable |
| INT13E | Enable or Disable Timer 1 Match Interrupt | |
| | 0 | Disable |
| | 1 | Enable |
| INT12E | Enable or Disable Timer 0 Match Interrupt | |
| | 0 | Disable |
| | 1 | Enable |

### IE3 (Interrupt Enable Register 3): ABH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | INT22E | INT21E | – | – | – |
| – | – | – | R/W | R/W | – | – | – |

Initial value: 00H

| | | |
|---|---|---|
| INT22E | Enable or Disable BIT Interrupt | |
| | 0 | Disable |
| | 1 | Enable |
| INT21E | Enable or Disable WDT Interrupt | |
| | 0 | Disable |
| | 1 | Enable |

### IP (Interrupt Priority Register): B8H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | IP5 | IP4 | IP3 | IP2 | IP1 | IP0 |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

### IP1 (Interrupt Priority Register 1): F8H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | IP15 | IP14 | IP13 | IP12 | IP11 | IP10 |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

IP[5:0], IP1[5:0] Select Interrupt Group Priority

| IP1x | IPx | Description |
|---|---|---|
| 0 | 0 | level 0 (lowest) |
| 0 | 1 | level 1 |
| 1 | 0 | level 2 |
| 1 | 1 | level 3 (highest) |

### EIFLAG (External Interrupt Flag Register): A0H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| –- | – | FLAG11 | FLAG10 | FLAG3 | FLAG2 | FLAG1 | FLAG0 |
| – | – | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

EIFLAG[5:0] When an external interrupt (EINT0/1/2/3/10/11) is occurred, the flag becomes '1'. The flag is cleared by writing a '0' to the bit or automatically cleared by INT_ACK signal. Writing "1" has no effect.

0 External Interrupt 0/1/2/3/10/11 not occurred

1 External Interrupt 0/1/2/3/10/11 occurred

### EIPOL0 (External Interrupt Polarity 0 Register): A4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| POL3 | | POL2 | | POL1 | | POL0 | |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

EIPOL0[7:0] External interrupt (EINT0, EINT1, EINT2, EINT3) polarity selection

| POLn[1:0] | | Description |
|---|---|---|
| 0 | 0 | No interrupt at any edge |
| 0 | 1 | Interrupt on rising edge |
| 1 | 0 | Interrupt on falling edge |
| 1 | 1 | Interrupt on both of rising and falling edge |

Where n = 0, 1, 2 and 3

### EIPOL1 (External Interrupt Polarity 1 Register): A5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| –- | – | –- | – | POL11 | | POL10 | |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

EIPOL1[3:0] External interrupt (EINT10 and EINT11) polarity selection

| POLn[1:0] | | Description |
|---|---|---|
| 0 | 0 | No interrupt at any edge |
| 0 | 1 | Interrupt on rising edge |
| 1 | 0 | Interrupt on falling edge |
| 1 | 1 | Interrupt on both of rising and falling edge |

Where n = 10 and 11

# 7 Clock generator

As shown in figure 22, a clock generator produces basic clock pulses which provide a CPU and peripherals with a system clock. A default system clock is a 1MHz INT-RC oscillator and default division rate is one. To stabilize system internally, it is used 1MHz INT-RC oscillator on POR.

A96L322 incorporates two types of oscillators:

- Calibrated Internal RC Oscillator (1MHz)
  - INT-RC OSC/8 (0.125MHz)
  - INT-RC OSC/4 (0.25MHz)
  - INT-RC OSC/2 (0.5MHz)
  - INT-RC OSC/1 (1MHz, default system clock)
- Internal WDTRC Oscillator (1KHz)

## 7.1 Block diagram

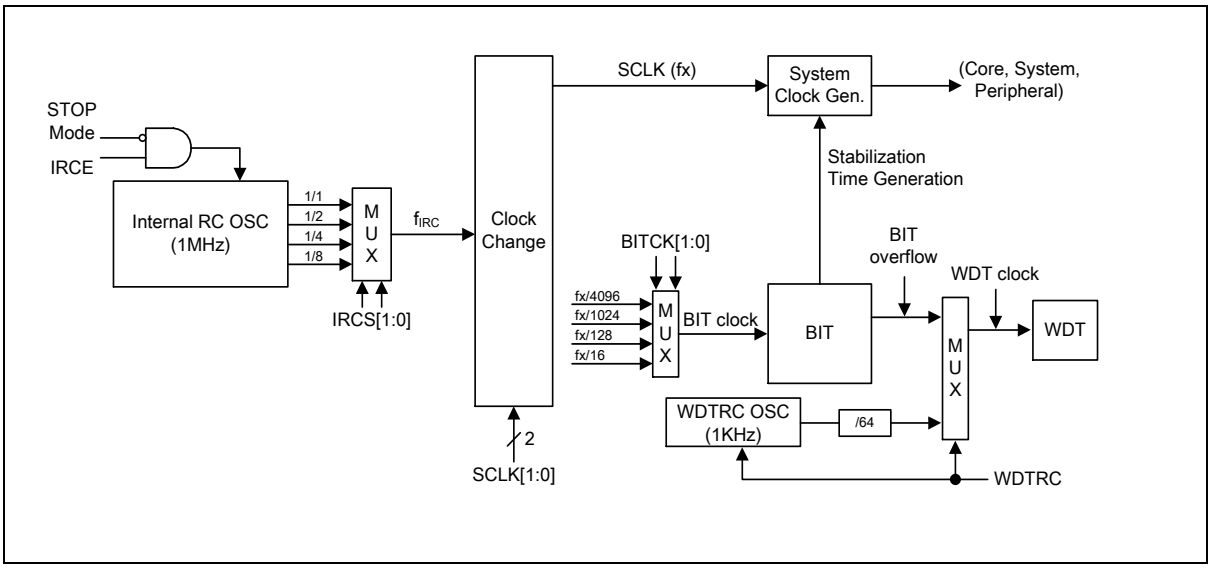



Figure 22. Clock Generator in Block Diagram

## 7.2 Register map

Table 11. Clock Generator Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| SCCR | 8AH | R | 00H | System and Clock Control Register |
| OSCCR | C8H | R/W | 18H | Oscillator Control Register |
| IRCTCR | 8FH | R/W | 00H | Internal RC Trim Control Register |
| IRCTRM | 97H | R/W | xxH | Internal RC Trim Register |
| IRCIDR | 9FH | R/W | 00H | Internal RC Identification Register |

## 7.3 Register description

Clock generator registers use the clock control for system operation. The clock generator consists of System and clock control register, oscillator control register internal RC trim control register, internal RC trim register, and internal RC identification register.

**SCCR (System and Clock Control Register): 8AH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | SCLK1 | SCLK0 |
| – | – | – | – | – | – | R | R |

Initial value: 00H

SCLK[1:0] System Clock Selection Bit

| SCLK1 | SCLK0 | Description |
|---|---|---|
| 0 | 0 | INT-RC OSC (fIRC) for system clock |

**OSCCR (Oscillator Control Register): C8H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | IRCS1 | IRCS0 | – | – | – |
| – | – | – | R/W | R/W | – | – | – |

Initial value: 18H

IRCS[1:0] Internal RC Oscillator Post-divider Selection

| IRCS1 | IRCS0 | Description |
|---|---|---|
| 0 | 0 | fIRC/8 (0.125MHz) |
| 0 | 1 | fIRC/4 (0.25MHz) |
| 1 | 0 | fIRC/2 (0.5MHz) |
| 1 | 1 | fIRC/1 (1MHz) |

**IRCIDR (Internal RC Trim Identification Register): 9FH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| IRCID7 | IRCID6 | IRCID5 | IRCID4 | IRCID3 | IRCID2 | IRCID1 | IRCID0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

IRCID[7:0] Internal RC Trim Identification.

| | |
|---|---|
| Others | No identification value |
| 01000110b | Identification value for IRC Trim |

(These bits are automatically cleared to logic '00H' immediately after one time operation.)

**IRCTRM (Internal RC Trim Register): 97H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ITRM7 | ITRM6 | ITRM5 | ITRM4 | ITRM3 | ITRM2 | ITRM1 | ITRM0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: xxH

ITRM[7:0] Internal RC Trim bits.
These bits are read from "Configure Area" when a system reset occurs. These bits provide a user programmable trimming value on operation. The range is -128 to +127. The ITRM7 is sign bit. The IRC frequency is faster by minus value and slower by plus. The frequency is changed by about 0.5[kHz] to 14[kHz] step-by-step. This register can be written with valid ID value and IRCTCR=0xB3.

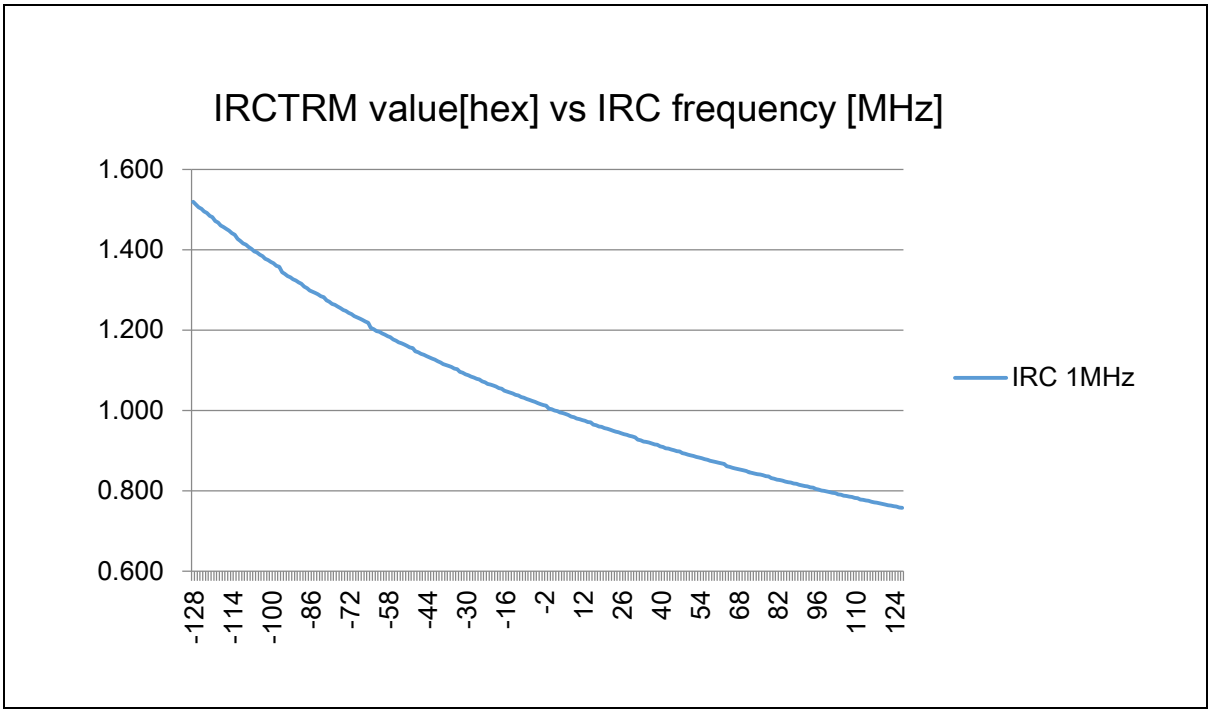



Figure 23. IRCTRM Value vs. IRC Frequency Graph

**IRCTCR (Internal RC Trim Control Register): 8FH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ITCR7 | ITCR6 | ITCR5 | ITCR4 | ITCR3 | ITCR2 | ITCR1 | ITCR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

ITCR[7:0] Internal RC Trim Control Register.

| | |
|---|---|
| Others | No effect |
| 10110011b | IRCTRM register is used for IRC frequency. This register can be written with valid ID value. |

# 8 Basic interval timer

A96L322 has a free running 8-bit Basic Interval Timer (BIT). BIT generates the time base for watchdog timer counting, and provides a basic interval timer interrupt (BITIFR).

BIT of A96L322 features the followings:

- During Power On, BIT gives a stable clock generation time
- On exiting Stop mode, BIT gives a stable clock generation time
- As a timer, BIT generates a timer interrupt.

## 8.1 Block diagram

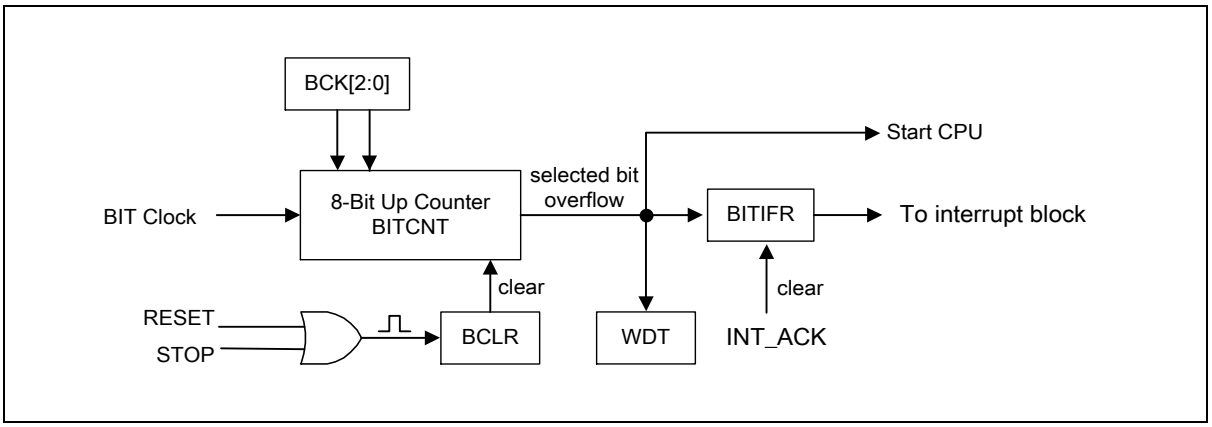



Figure 24. Basic Interval Timer in Block Diagram

## 8.2 Register map

Table 12. Basic Interval Timer Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| BITCNT | 8CH | R | 00H | Basic Interval Timer Counter Register |
| BITCR | 8BH | R/W | 01H | Basic Interval Timer Control Register |

## 8.3 Register description

BIT registers consist of Basic Interval Timer Counter register (BITCNT) and Basic Interval Timer Control register (BITCR). If BCLR bit in BITCR register is set to '1', BITCNT becomes '0' and then counts up. When 1 machine cycle is completed, BCLR bit is cleared to '0' automatically.

### BITCNT (Basic Interval Timer Counter Register): 8CH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BITCNT7 | BITCNT6 | BITCNT5 | BITCNT4 | BITCNT3 | BITCNT2 | BITCNT1 | BITCNT0 |
| R | R | R | R | R | R | R | R |

Initial value: 00H

BITCNT[7:0] BIT Counter

### BITCR (Basic Interval Timer Control Register): 8BH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| BITIFR | BITCK1 | BITCK0 | – | BCLR | BCK2 | BCK1 | BCK0 |
| R/W | R/W | R/W | – | R/W | R/W | R/W | R/W |

Initial value: 01H

BITIFR When BIT Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal. Writing "1" has no effect.

0 BIT interrupt no generation

1 BIT interrupt generation

BITCK[1:0] Select BIT clock source

| BITCK1 | BITCK 0 | Description |
|---|---|---|
| 0 | 0 | fx/4096 |
| 0 | 1 | fx/1024 |
| 1 | 0 | fx/128 |
| 1 | 1 | fx/16 |

BCLR If this bit is written to '1', BIT Counter is cleared to '0'

0 Free Running

1 Clear Counter

BCK[2:0] Select BIT overflow period

| BCK2 | BCK1 | BCK0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Bit 0 overflow (BIT Clock * 2) |
| 0 | 0 | 1 | Bit 1 overflow (BIT Clock * 4) (default) |
| 0 | 1 | 0 | Bit 2 overflow (BIT Clock * 8) |
| 0 | 1 | 1 | Bit 3 overflow (BIT Clock * 16) |
| 1 | 0 | 0 | Bit 4 overflow (BIT Clock * 32) |
| 1 | 0 | 1 | Bit 5 overflow (BIT Clock * 64) |
| 1 | 1 | 0 | Bit 6 overflow (BIT Clock * 128) |
| 1 | 1 | 1 | Bit 7 overflow (BIT Clock * 256) |

# 9 Watchdog timer

Watchdog timer is used to rapidly detect the CPU malfunctions such as endless looping caused by noise. In addition, it is used to resume the CPU in a normal state. Watchdog timer signal for malfunction detection can be used as either a CPU reset or an interrupt request. When the Watchdog Timer is not being used for the detection of the CPU malfunctions, it can be used as a timer generating an interrupt at fixed intervals.

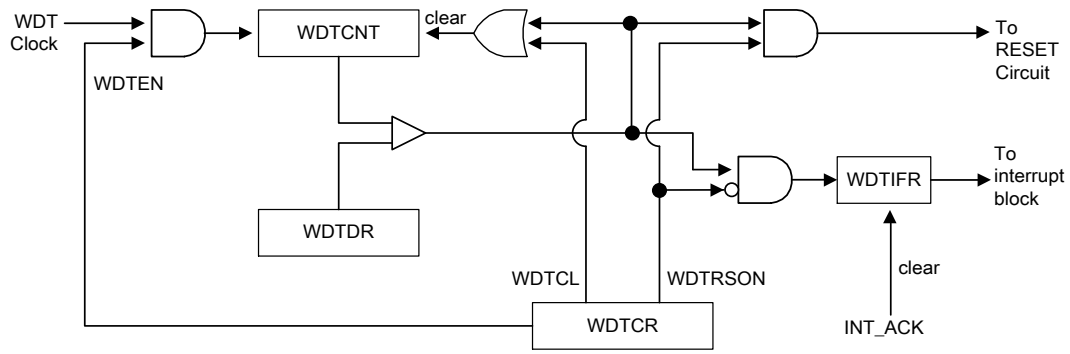
Watchdog timer can be used in a free running 8-bit timer mode or in a watch dog timer mode by setting WDTRSON bit, which is WDTCR[5]. If '1' is written to WDTCR[5], WDT counter value is cleared and counts up. After 1 machine cycle, this bit is cleared to '0' automatically.

Watchdog timer consists of an 8-bit binary counter and a watchdog timer data register. When value of an 8-bit binary counter is equal to the 8 bits of WDTCNT, an interrupt request flag is generated. This can be used as a watchdog timer interrupt or a reset of CPU in accordance with a bit WDTRSON.

The input clock source of watch dog timer is the BIT overflow. Interval of watchdog timer interrupt is decided by the BIT overflow period and WDTDR set value. Equation of the WDT interrupt interval is described in the followings:

WDT Interrupt Interval = (BIT Interrupt Interval) X (WDTDR Value+1)

## 9.1 Block diagram




**NOTE**: Watchdog timer clock depends on WDTCK bit in WDTCR register. For detailed information of the WDTCR register, please refer to Watchdog timer register description later part in this chapter.

**Figure 25. Watchdog Timer in Block Diagram**

## 9.2 WDT interrupt timing waveform

Figure 26 shows a timing diagram when a watchdog timer generates system reset signal and an interrupt signal.

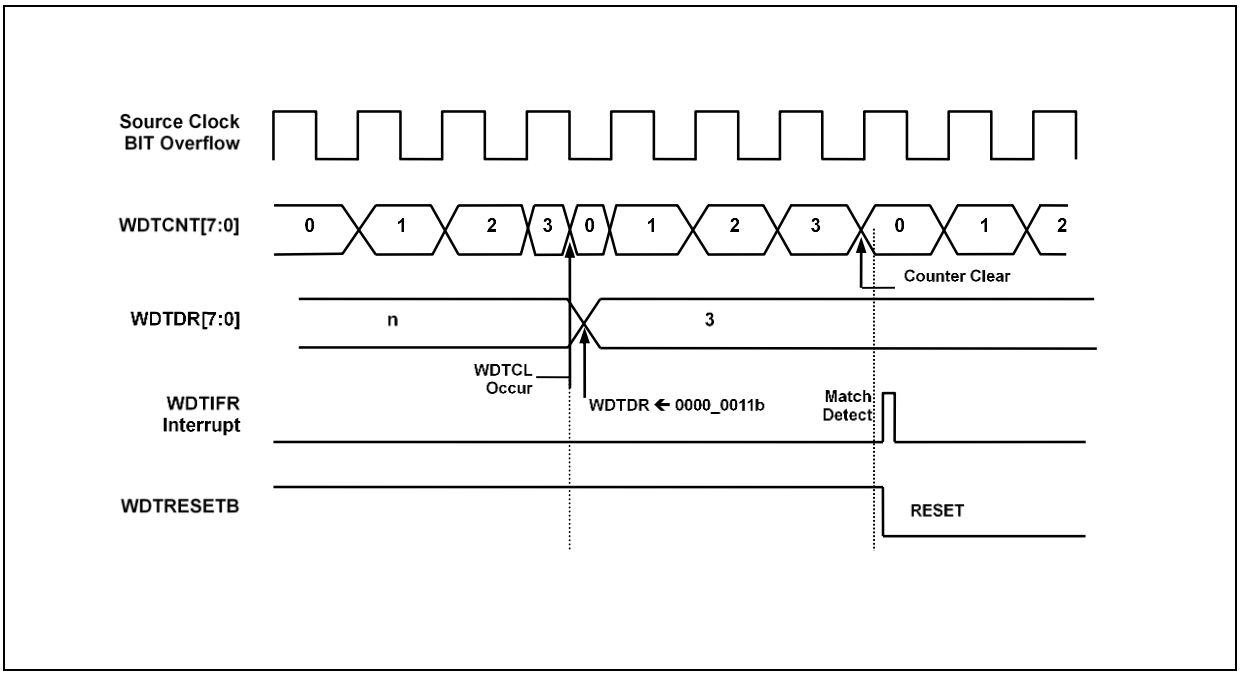



Figure 26. Watchdog Timer Interrupt Timing Waveform

## 9.3 Register map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| WDTCNT | 8EH | R | 00H | Watch Dog Timer Counter Register |
| WDTDR | 8EH | W | FFH | Watch Dog Timer Data Register |
| WDTCR | 8DH | R/W | 00H | Watch Dog Timer Control Register |

## 9.4 Register description

Watch dog timer registers consist of watch dog timer counter register (WDTCNT), watch dog timer data register (WDTDR) and watch dog timer control register (WDTCR).

### WDTCNT (Watch Dog Timer Counter Register: Read Case): 8EH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTCNT7 | WDTCNT6 | WDTCNT5 | WDTCNT4 | WDTCNT3 | WDTCNT2 | WDTCNT1 | WDTCNT0 |
| R | R | R | R | R | R | R | R |

Initial value: 00H

WDTCNT[7:0] WDT Counter

### WDTDR (Watch Dog Timer Data Register: Write Case): 8EH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTDR7 | WDTDR6 | WDTDR5 | WDTDR4 | WDTDR3 | WDTDR2 | WDTDR1 | WDTDR0 |
| W | W | W | W | W | W | W | W |

Initial value: FFH

WDTDR[7:0] Set a period
WDT Interrupt Interval=(BIT Interrupt Interval) x(WDTDR Value+1)
**NOTE:** Do not write "0" in the WDTDR register.

### WDTCR (Watch Dog Timer Control Register): 8DH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTEN | WDTRSON | WDTCL | – | – | – | WDTCK | WDTIFR |
| R/W | R/W | R/W | – | – | – | R/W | R/W |

Initial value: 00H

WDTEN Control WDT Operation
0 Disable
1 Enable

WDTRSON Control WDT RESET Operation
0 Free Running 8-bit timer
1 Watch Dog Timer RESET ON

WDTCL Clear WDT Counter
0 Free Run
1 Clear WDT Counter (auto clear after 1 Cycle)

WDTCK Control WDT Clock Selection Bit
0 BIT overflow for WDT clock (WDTRC disable)
1 WDTRC for WDT clock (WDTRC enable)

WDTIFR When WDT Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal. Writing "1" has no effect.
0 WDT Interrupt no generation
1 WDT Interrupt generation

# 10 TIMER 0

A 16-bit timer TIMER 0 incorporates a multiplexer and six registers such as timer0A data register high/low, timer0B data register high/low, and timer0 control register high/low (T0ADRH, T0ADRL, T0BDRH, T0BDRL, T0CRH, T0CRL).

TIMER 0 operates in one of four operating modes:

- 16-bit capture mode
- 16-bit timer/ counter mode
- 16-bit PPG output mode (one-shot mode)
- 16-bit PPG output mode (repeat mode)

Specifically in capture mode, data is captured into input capture data register (T0BDRH/T0BDRL) by EINT10. TIMER 0 outputs the comparison result between counter and data register through T0O port in timer/counter mode. TIMER 0 outputs PWM wave form through PWM0O port in the PPG mode.

A timer/counter 0 uses an internal clock or an external clock (EC0) as an input clock source. The clock sources are introduced below, and one is selected by clock selection logic which is controlled by clock selection bits (T0CK[2:0]).

- TIMER 0 clock sources: $f_X$/1, 2, 4, 8, 64, 512, 2048 and EC0

**Table 13. TIMER 0 Operating Modes**

| T0EN | P1FSRL[2](T0) | T0MS[1:0] | T0CK[2:0] | Timer 0 |
|---|---|---|---|---|
| 1 | 1 | 00 | XXX | 16 Bit Timer/Counter Mode |
| 1 | 0 | 01 | XXX | 16 Bit Capture Mode |
| 1 | 1 | 10 | XXX | 16 Bit PPG Mode(one-shot mode) |
| 1 | 1 | 11 | XXX | 16 Bit PPG Mode(repeat mode) |

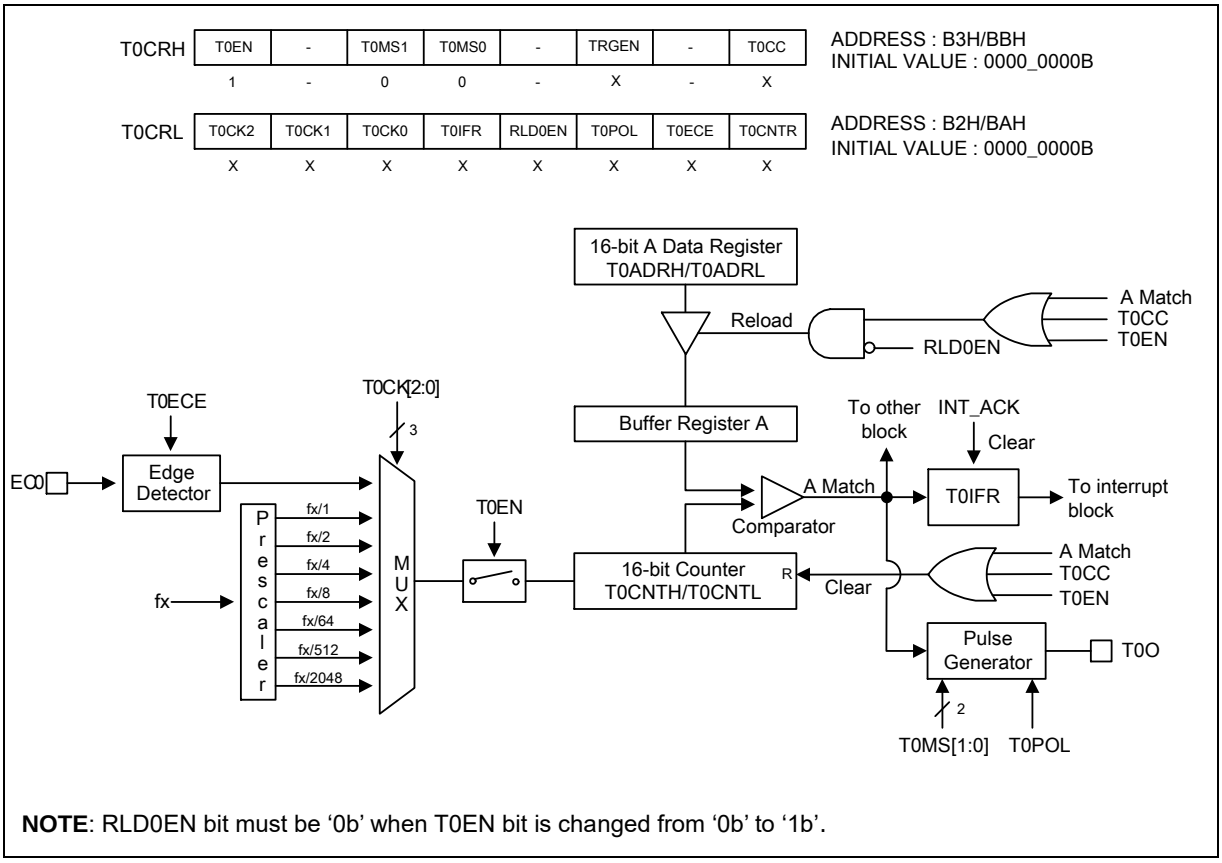
## 10.1 16-bit timer/ counter mode

16-bit timer/counter mode is selected by control register as shown in figure 27. As shown in figure 27, a 16-bit timer has a counter and data registers. Counter registers have increasing values by internal or external clock input. TIMER 0 can use the input clock with one of 1, 2, 4, 8, 64, 512 and 2048 prescaler division rates (T0CK[2:0]).

When the value of T0CNTH, T0CNTL and the value of T0ADRH, T0ADRL are identical each other in Timer 0, a match signal is generated and the interrupt of Timer 0 occurs.

The T0CNTH, T0CNTL value is automatically cleared by match signal. It can be cleared by software (T0CC) too.

The external clock (EC0) counts up the timer at the rising edge. If the EC0 is selected as a clock source by T0CK[2:0], EC0 port should be set to the input port by P12IO bit.




**NOTE**: RLD0EN bit must be '0b' when T0EN bit is changed from '0b' to '1b'.

**Figure 27. 16-bit Timer/ Counter Mode of TIMER 0**

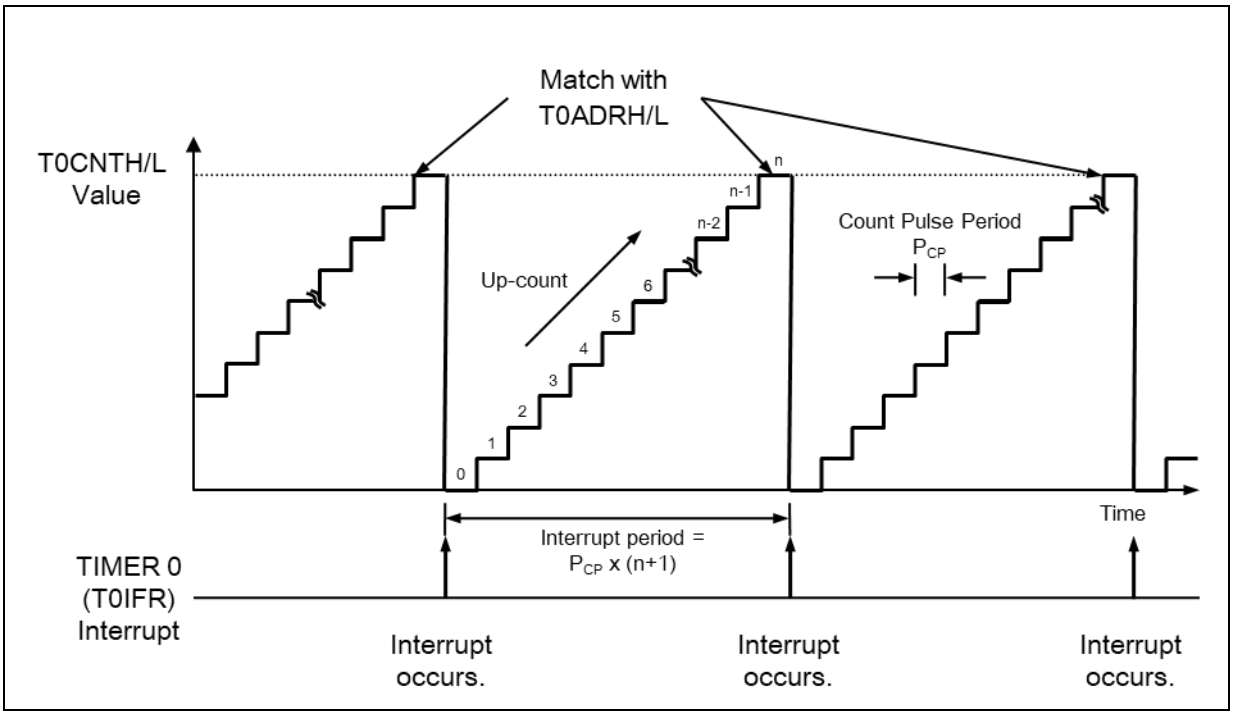
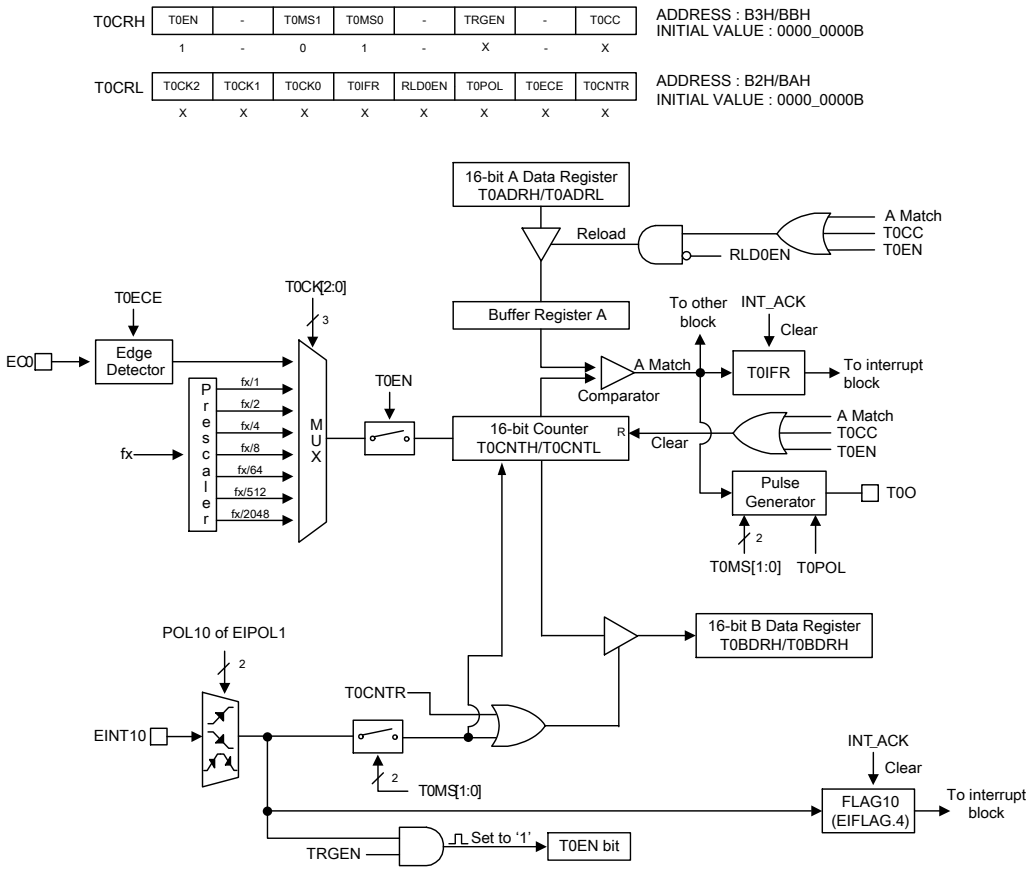



**Figure 28. 16-bit Timer/ Counter 0 Interrupt Example**

## 10.2 16-bit capture mode

16-bit timer 0 capture mode is set by configuring T0MS[1:0] as '01'. It uses an internal/external clock as a clock source. Basically, the 16-bit timer 0 capture mode has the same function as the 16-bit timer/counter mode, and the interrupt occurs when T0CNTH/T0CNTL is equal to T0ADRH/T0ADRL. The T0CNTH, T0CNTL values are automatically cleared by match signal. It can be cleared by software (T0CC) too.

A timer interrupt in capture mode is very useful when the pulse width of captured signal is wider than the maximum period of timer. Capture result is loaded into T0BDRH/T0BDRL. According to EIPOL1 registers settings, the external interrupt EINT10 function is selected. EINT10 pin must be set as an input port.




**NOTE**: RLD0EN bit must be '0b' when T0EN bit is changed from '0b' to '1b'.

**Figure 29. 16-bit Capture Mode of TIMER 0**

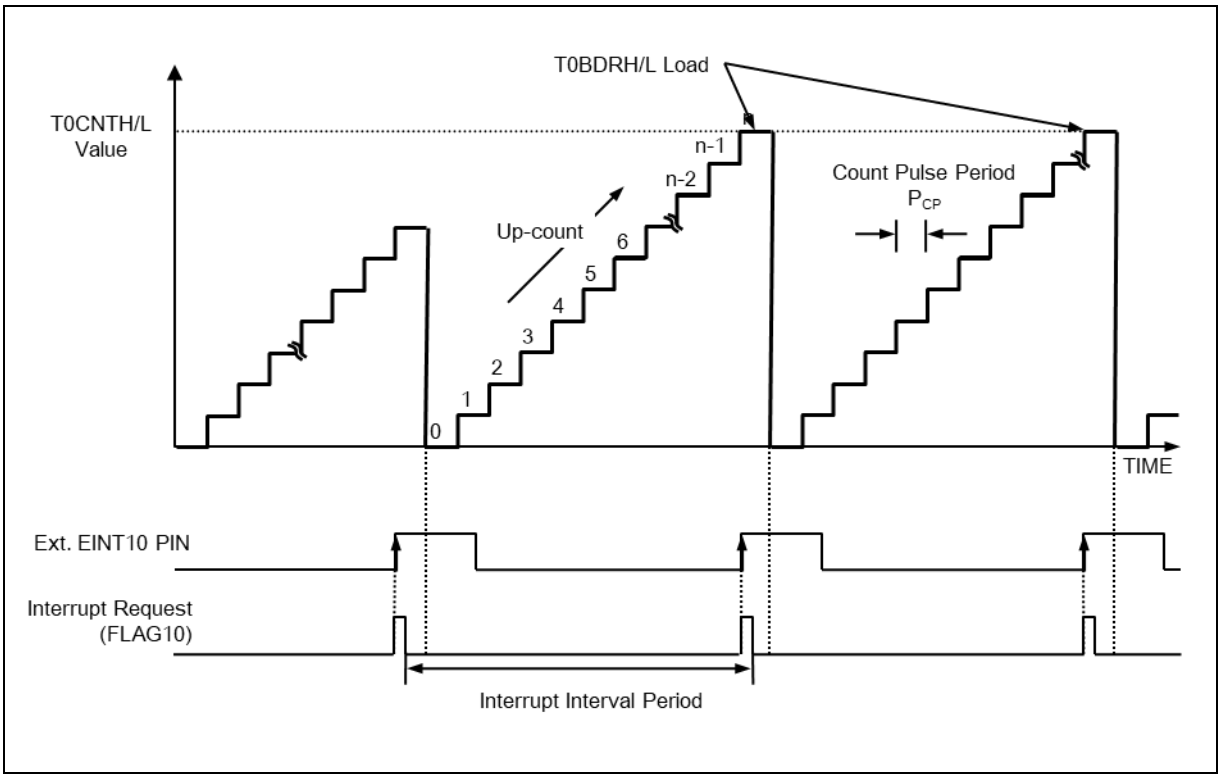



Figure 30. Input Capture Mode Operation of TIMER 0

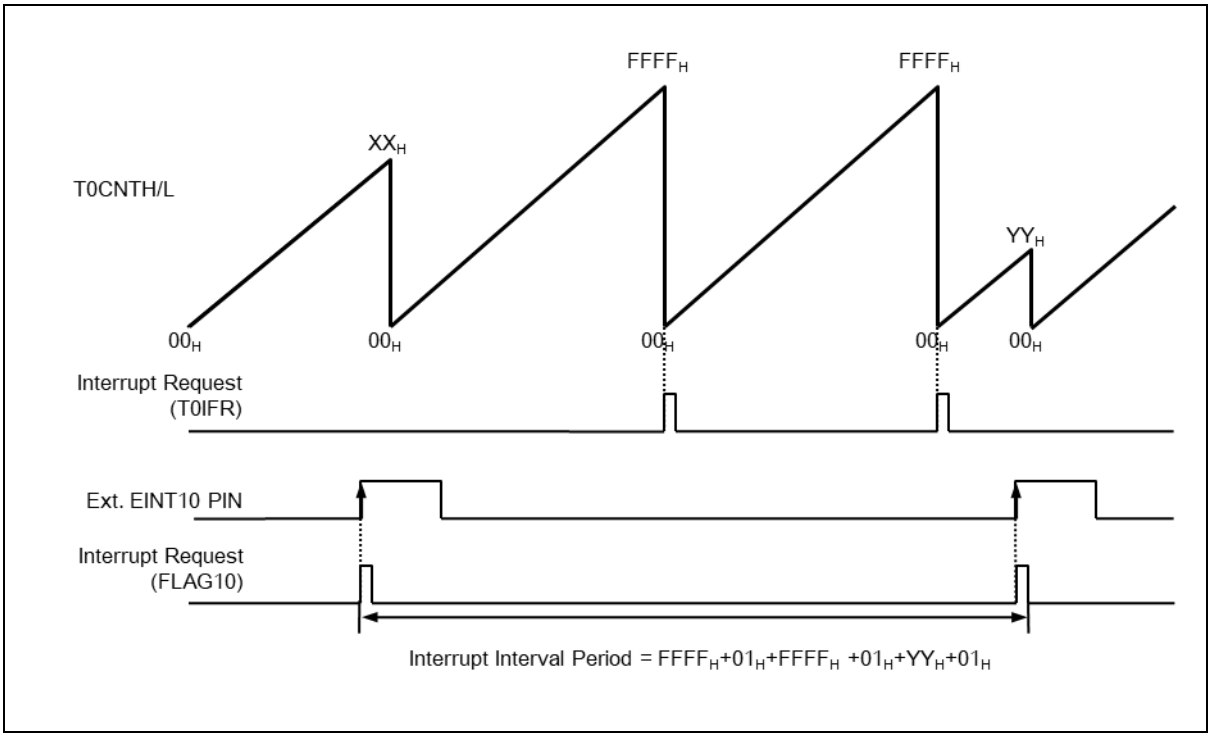
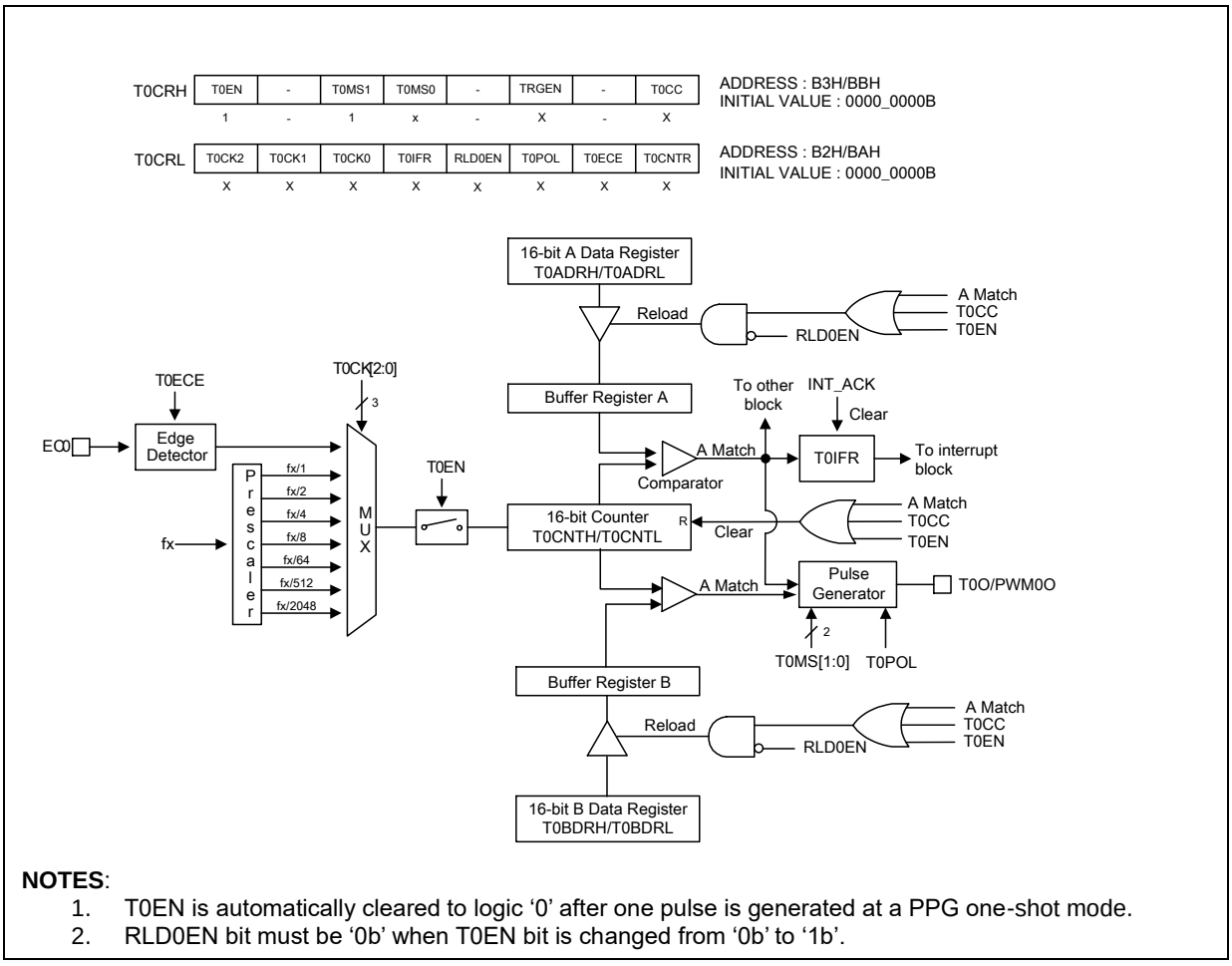



Figure 31. Express Timer Overflow in Capture Mode

## 10.3 16-bit PPG mode

TIMER 0 has a PPG (Programmable Pulse Generation) function. In PPG mode, T0O/PWM0O pin outputs up to 16-bit resolution PWM output.

For this function, T0O/PWM0O pin must be configured as a PWM output by setting P1FSRL[2] to '1'(T0). Period of the PWM output is determined by T0ADRH/T0ADRL, and duty of the PWM output is determined by T0BDRH/T0BDRL.




**NOTES**:

1. T0EN is automatically cleared to logic '0' after one pulse is generated at a PPG one-shot mode.
2. RLD0EN bit must be '0b' when T0EN bit is changed from '0b' to '1b'.

**Figure 32. 16-bit PPG Mode of TIMER 0**

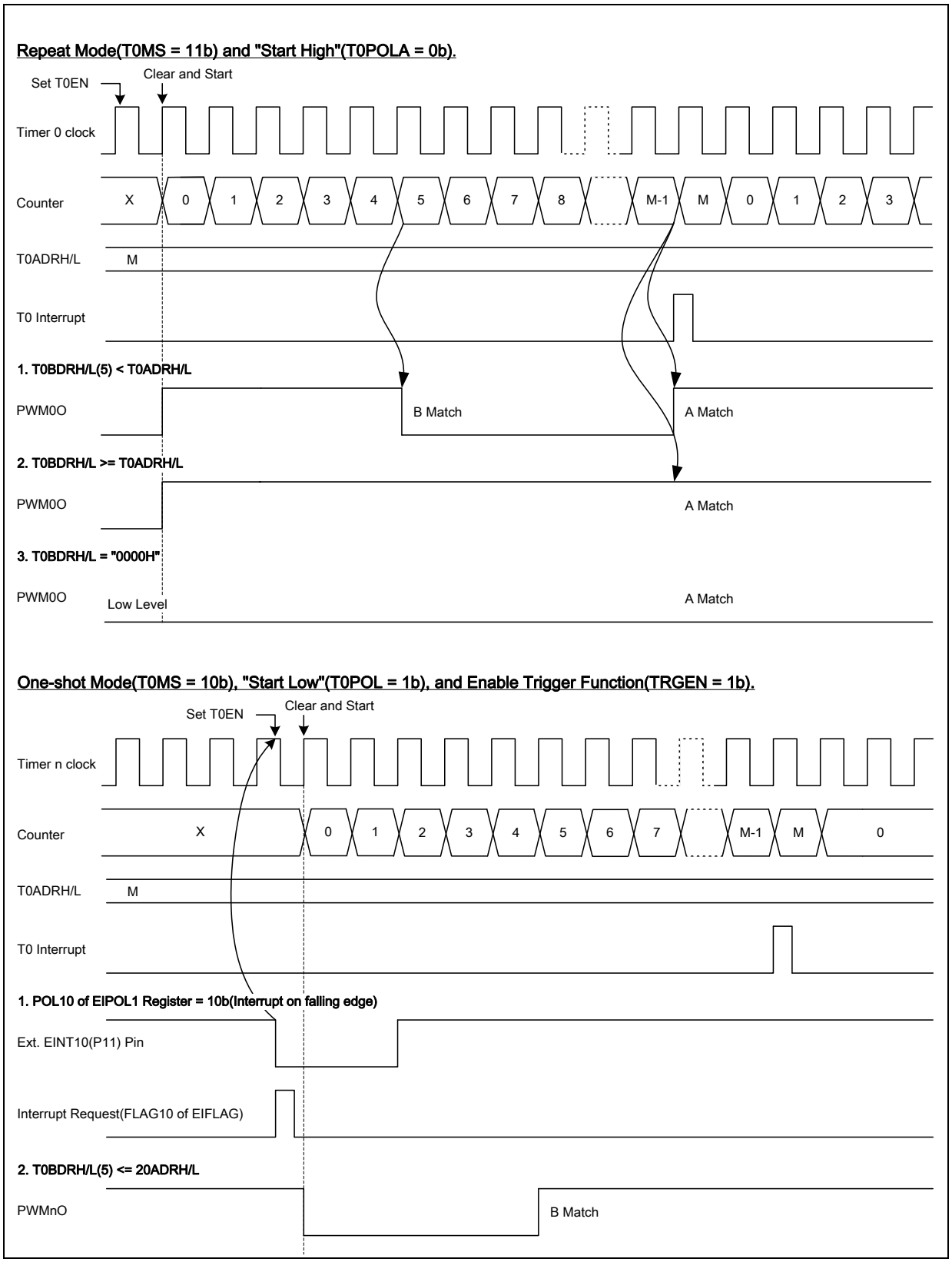
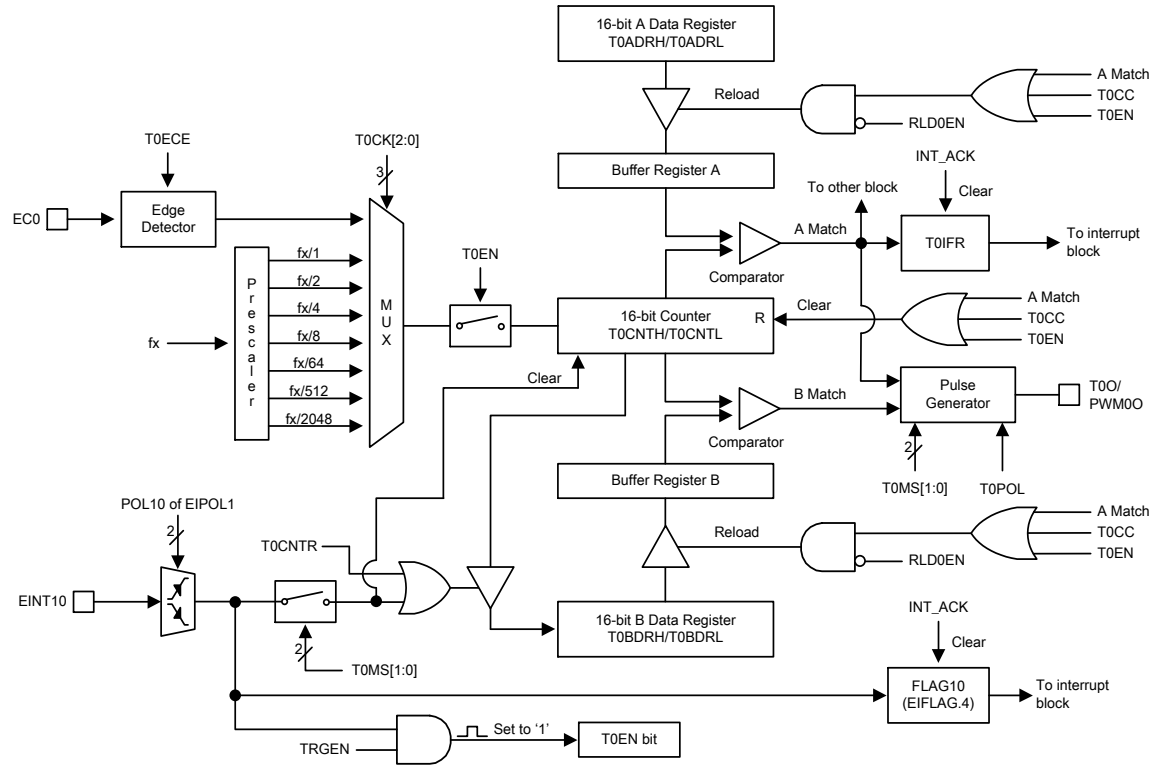



Figure 33. 16-bit PPG Mode Timing Chart of TIMER 0

## 10.4 Block diagram




**NOTES**:

1. T0EN is automatically cleared to logic '0' after one pulse is generated at a PPG one-shot mode.
2. RLD0EN bit must be '0b' when T0EN bit is changed from '0b' to '1b'.

**Figure 34. 16-bit Timer 0 in Block Diagram**

## 10.5 Register map

**Table 14. TIMER 0 Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| T0CRH | B3H | R/W | 00H | Timer 0 Control High Register |
| T0CRL | B2H | R/W | 00H | Timer 0 Control Low Register |
| T0ADRH | B5H | R/W | FFH | Timer 0 A Data High Register |
| T0ADRL | B4H | R/W | FFH | Timer 0 A Data Low Register |
| T0BDRH | B7H | R/W | FFH | Timer 0 B Data High Register |
| T0BDRL | B6H | R/W | FFH | Timer 0 B Data Low Register |

## 10.6 Timer/counter 0 Register description

**T0ADRH (Timer 0 A data High Register): B5H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0ADRH7 | T0ADRH6 | T0ADRH5 | T0ADRH4 | T0ADRH3 | T0ADRH2 | T0ADRH1 | T0ADRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T0ADRH[7:0] T0 A Data High Byte

**T0DRL (Timer 0 A Data Low Register): B4H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0ADRL7 | T0ADRL6 | T0ADRL5 | T0ADRL4 | T0ADRL3 | T0ADRL2 | T0ADRL1 | T0ADRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T0ADRL[7:0] T0 A Data Low Byte
**NOTE:** Do not write "0000H" in the T0ADRH/T0ADRL register when PPG mode

**T0BDRH (Timer 0 B Data High Register): B7H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0BDRH7 | T0BDRH6 | T0BDRH5 | T0BDRH4 | T0BDRH3 | T0BDRH2 | T0BDRH1 | T0BDRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T0BDRH[7:0] T0 B Data High Byte

**T0BDRL (Timer 0 B Data Low Register): B6H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0BDRL7 | T0BDRL6 | T0BDRL5 | T0BDRL4 | T0BDRL3 | T0BDRL2 | T0BDRL1 | T0BDRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T0BDRL[7:0] T0 B Data Low Byte

### T0CRH (Timer 0 Control High Register): B3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0EN | – | T0MS1 | T0MS0 | – | TRGEN | – | T0CC |
| R/W | – | R/W | R/W | – | R/W | – | R/W |

Initial value: 00H

| | | | |
|---|---|---|---|
| T0EN | Control Timer 0 | | |
| | 0 | Timer 0 disable | |
| | 1 | Timer 0 enable (Counter clear and start) | |
| T0MS[1:0] | Control Timer 0 Operation Mode | | |
| | T0MS1 | T0MS | Description |
| | | 0 | |
| | 0 | 0 | Timer/counter mode (T0O: toggle at A match) |
| | 0 | 1 | Capture mode (The A match interrupt can occur) |
| | 1 | 0 | PPG one-shot mode (PWM0O) |
| | 1 | 1 | PPG repeat mode (PWM0O) |
| TRGEN | Control Trigger Function | | |
| | 0 | Disable trigger function | |
| | 1 | Enable trigger function by valid edge | |
| T0CC | Clear Timer 0 Counter | | |
| | 0 | No effect | |
| | 1 | Clear the Timer 0 counter (When write, automatically cleared "0" after being cleared counter) | |

**NOTE:** Refer to the EIPOL1 register to select a valid edge for an external trigger or capture signal. If the TRGEN bit is set to "1b", the timer 0 will be started or restarted by a valid edge.

**T0CRL (Timer 0 Control Low Register): B2H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T0CK2 | T0CK1 | T0CK0 | T0IFR | RLD0EN | T0POL | T0ECE | T0CNTR |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

T0CK[2:0] Select Timer 0 clock source. fx is main system clock frequency

| T0CK2 | T0CK1 | T0CK0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | fx/2048 |
| 0 | 0 | 1 | fx/512 |
| 0 | 1 | 0 | fx/64 |
| 0 | 1 | 1 | fx/8 |
| 1 | 0 | 0 | fx/4 |
| 1 | 0 | 1 | fx/2 |
| 1 | 1 | 0 | fx/1 |
| 1 | 1 | 1 | External clock (EC0) |

T0IFR When T0 Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal. Writing "1" has no effect.
- 0 T0 Interrupt no generation
- 1 T0 Interrupt generation

RLD0EN Control Timer 0 Reload Signal
- 0 Enable Timer 0 reload signal
- 1 Disable Timer 0 reload signal

T0POL T0O/PWM0O Polarity Selection
- 0 Start High (T0O/PWM0O is low level at disable)
- 1 Start Low (T0O/PWM0O is high level at disable)

T0ECE Timer 0 External Clock Edge Selection
- 0 External clock falling edge
- 1 External clock rising edge

T0CNTR Timer 0 Counter Read Control
- 0 No effect
- 1 Load the counter value to the B data register (When write, automatically cleared "0" after being loaded)

# 11 TIMER 1

A 16-bit timer TIMER 1 incorporates a multiplexer and nineteen registers such as timer1A data register high/low, timer1B data register high/low, timer1 control register high/low, siren control register, siren max data high/low register, siren min data high/low register, siren up/down match times register, siren up/down bundle times register, siren down decrement data register, siren down increment match times register, siren up increment data register and siren up decrement match times register (T1ADRH, T1ADRL, T1BDRH, T1BDRL, T1CRH, T1CRL, SIRENCR, MAXDRL, MAXDRH, MINDRL, MINDRH, DWMAT, DWBNDL, DWDECD, DWINCM, UPMAT, UPBNDL, UPINCD, UPDECM).

TIMER 1 operates in one of five operating modes:

- 16-bit capture mode
- 16-bit timer/ counter mode
- 16-bit PPG output mode (one-shot mode)
- 16-bit PPG output mode (repeat mode)
- Siren

Specifically in capture mode, data is captured into input capture data register (T1BDRH/T1BDRL) by EINT10/EINT11. TIMER 1 outputs the comparison result between counter and data register through T1O port in timer/counter mode. TIMER 1 outputs PWM wave form through PWM1O port in the PPG mode.

A timer/counter 1 uses an internal clock or an external clock (EC1) as an input clock source. The clock sources are introduced below, and one is selected by clock selection logic which is controlled by clock selection bits (T1CK[2:0]).

- TIMER 1 clock sources: $f_X$/1, 2, 4, 8, 64, 512, 2048 and EC1

**Table 15. TIMER 1 Operating Modes**

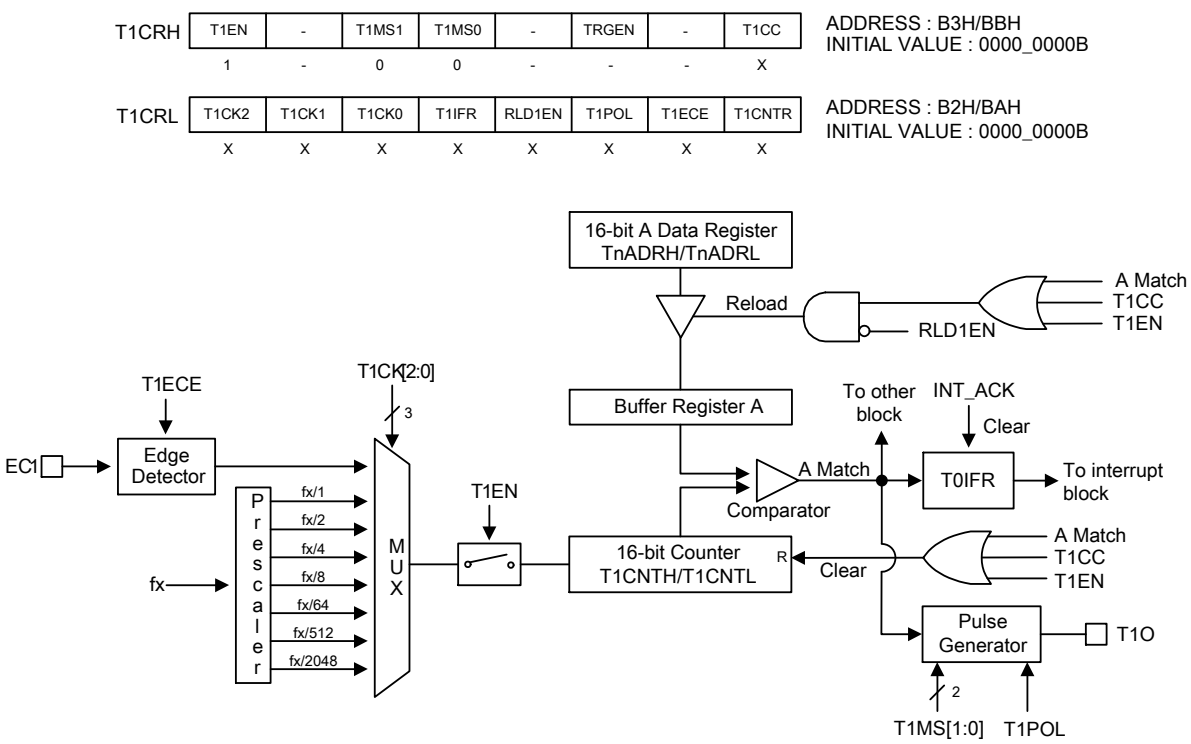
| T1EN | P1FSRH[1:0](T1) | T1MS[1:0] | T1CK[2:0] | Timer 1 |
|---|---|---|---|---|
| 1 | 01 | 00 | XXX | 16 Bit Timer/Counter Mode |
| 1 | 00 | 01 | XXX | 16 Bit Capture Mode |
| 1 | 01 | 10 | XXX | 16 Bit PPG Mode(one-shot mode) |
| 1 | 01 | 11 | XXX | 16 Bit PPG Mode(repeat mode) |

## 11.1 16-bit timer/ counter mode

16-bit timer/counter mode is selected by control register as shown in figure 35. As shown in figure 35, a 16-bit timer has a counter and data registers.

Counter registers have increasing values by internal or external clock input. TIMER 1 can use the input clock with one of 1, 2, 4, 8, 64, 512 and 2048 prescaler division rates (T1CK[2:0]). When the value of T1CNTH, T1CNTL and the value of T1ADRH, T1ADRL are identical each other in Timer 1, a match signal is generated and the interrupt of Timer 1 occurs. T1CNTH, T1CNTL value is automatically cleared by match signal. It can be cleared by software (T1CC) too.

The external clock (EC1) counts up the timer at the rising edge. If the EC1 is selected as a clock source by T1CK[2:0], EC1 port should be set to the input port by P13IO bit.




**NOTE**: RLD1EN bit must be '0b' when T1EN bit is changed from '0b' to '1b'.

**Figure 35. 16-bit Timer/ Counter Mode of TIMER 1**

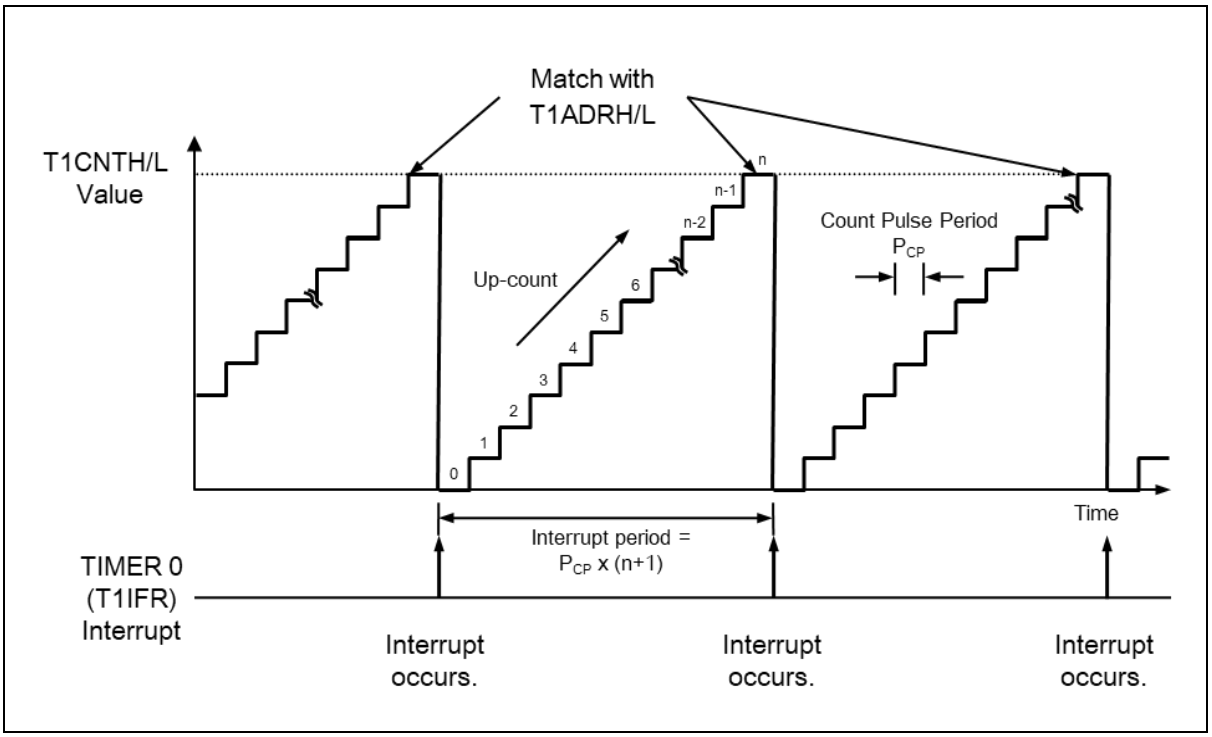
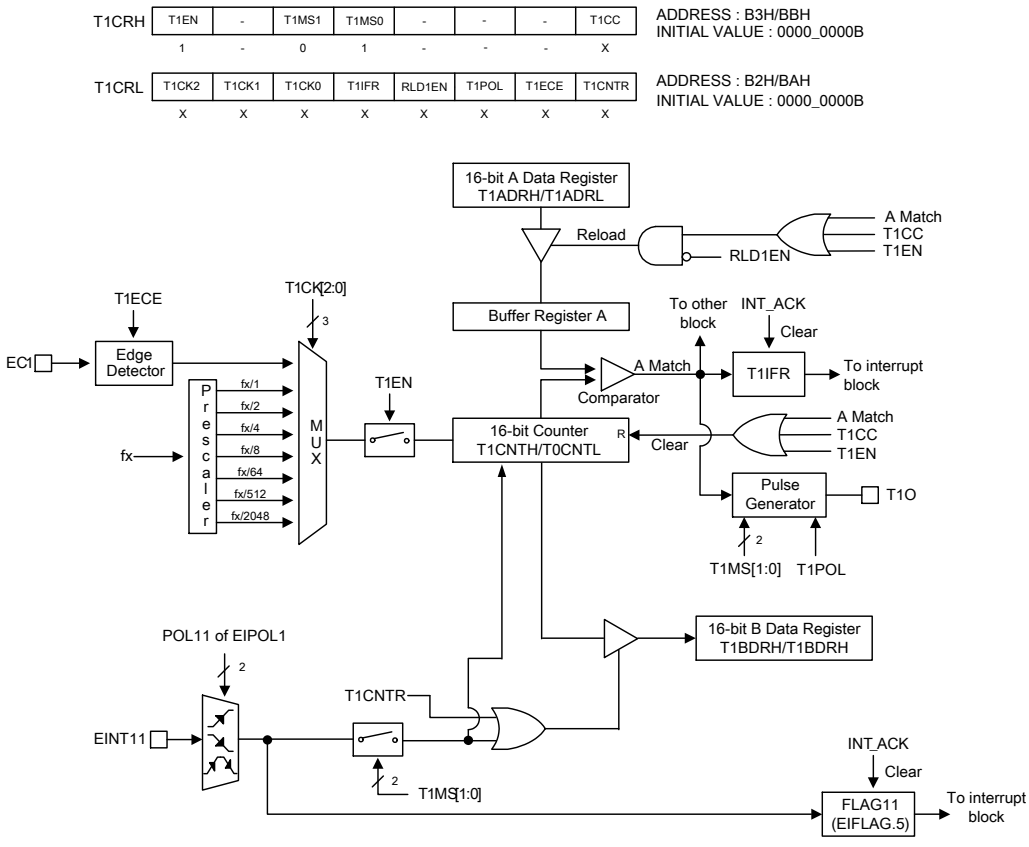



**Figure 36. 16-bit Timer/ Counter 1 Interrupt Example**

## 11.2 16-bit capture mode

16-bit timer 1 capture mode is set by configuring T1MS[1:0] as '01'. It uses an internal/external clock as a clock source. Basically, the 16-bit timer 1 capture mode has the same function as the 16-bit timer/counter mode, and the interrupt occurs when T1CNTH/T1CNTL is equal to T1ADRH/T1ADRL. The T1CNTH, T1CNTL values are automatically cleared by match signal. It can be cleared by software (T1CC) too.

A timer interrupt in capture mode is very useful when the pulse width of captured signal is wider than the maximum period of timer. Capture result is loaded into T0BDRH/T0BDRL. According to EIPOL1 registers settings, the external interrupt EINT10/EINT11 function is selected. EINT10/EINT11 pin must be set as an input port.




**NOTE**: RLD1EN bit must be '0b' when T1EN bit is changed from '0b' to '1b'.

**Figure 37. 16-bit Capture Mode of TIMER 1**

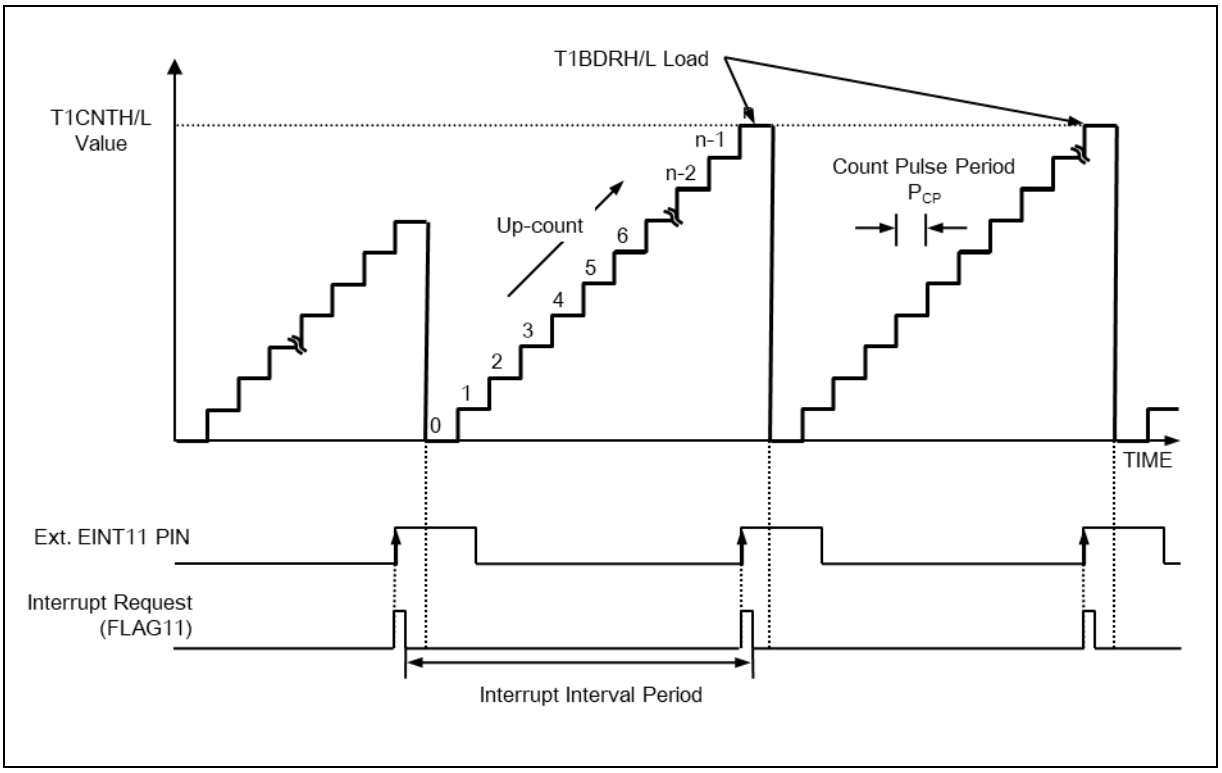



Figure 38. Input Capture Mode Operation of TIMER 1

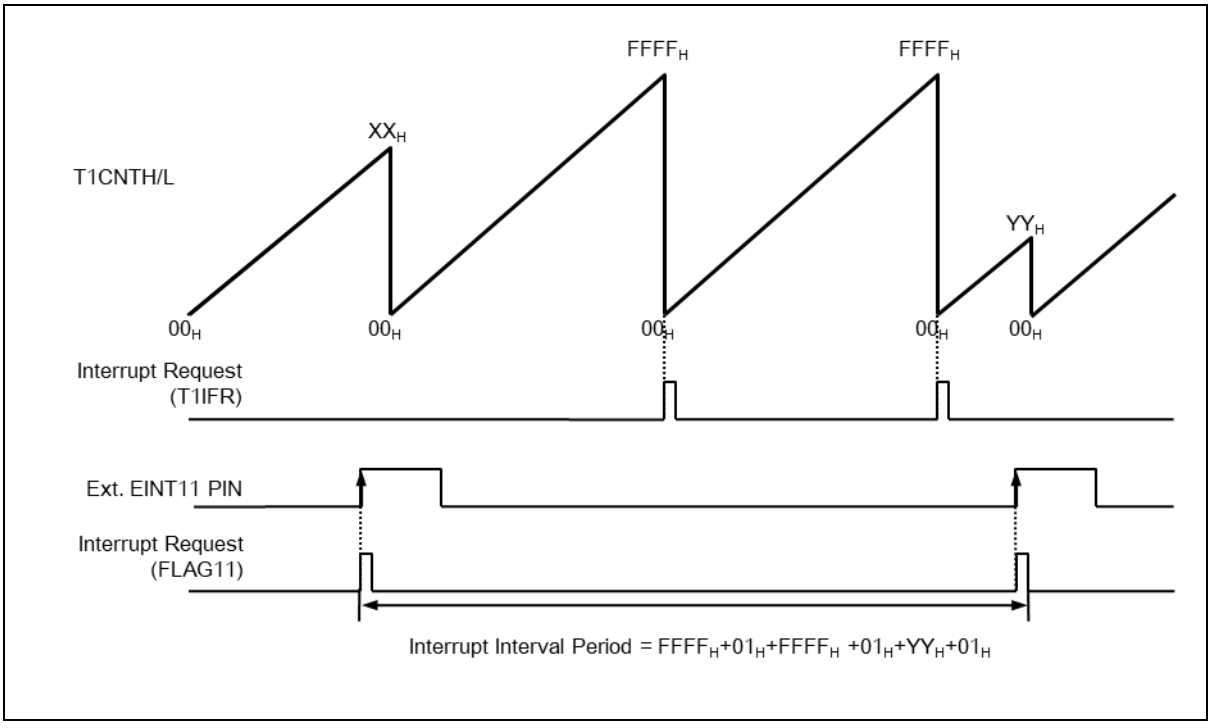
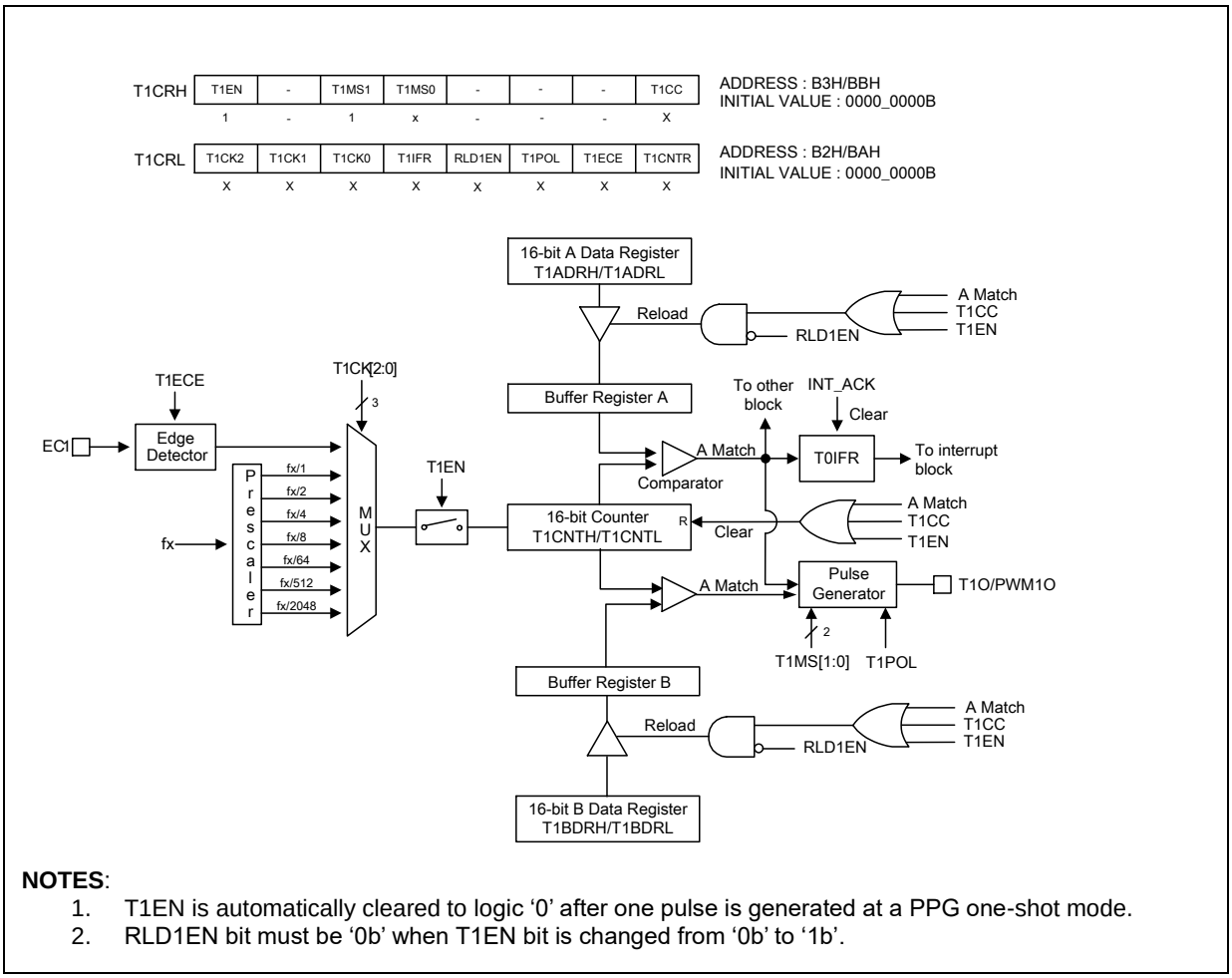



Figure 39. Express Timer Overflow in Capture Mode

## 11.3 16-bit PPG mode

TIMER 1 has a PPG (Programmable Pulse Generation) function. In PPG mode, T1O/PWM1O pin outputs up to 16-bit resolution PWM output.

For this function, T1O/PWM1O pin must be configured as a PWM output by setting P1FSRH[1:0](T1) to '01'. Period of the PWM output is determined by T1ADRH/T1ADRL, and duty of the PWM output is determined by T1BDRH/T1BDRL.




**NOTES**:

1. T1EN is automatically cleared to logic '0' after one pulse is generated at a PPG one-shot mode.
2. RLD1EN bit must be '0b' when T1EN bit is changed from '0b' to '1b'.

**Figure 40. 16-bit PPG Mode of TIMER 1**

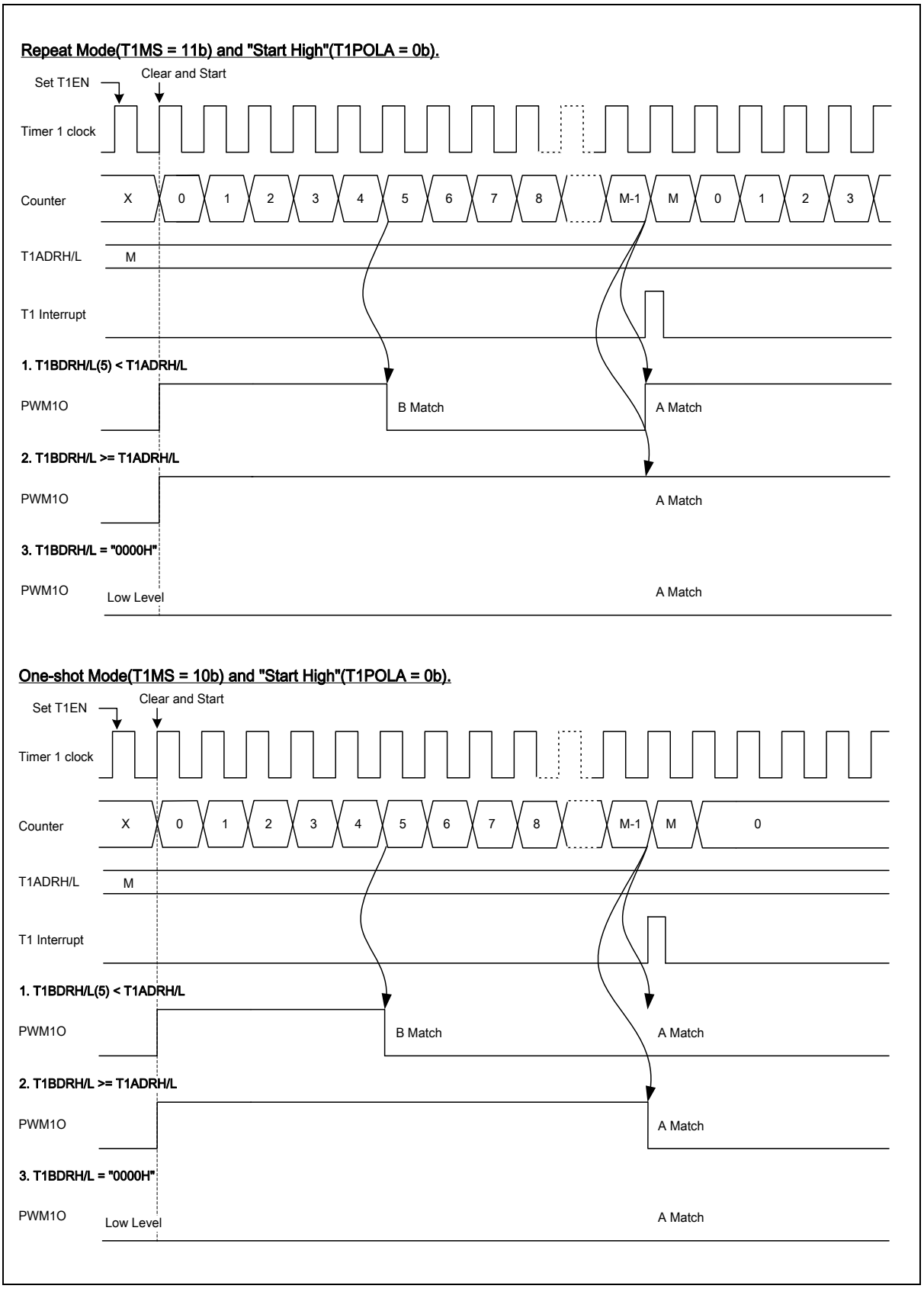


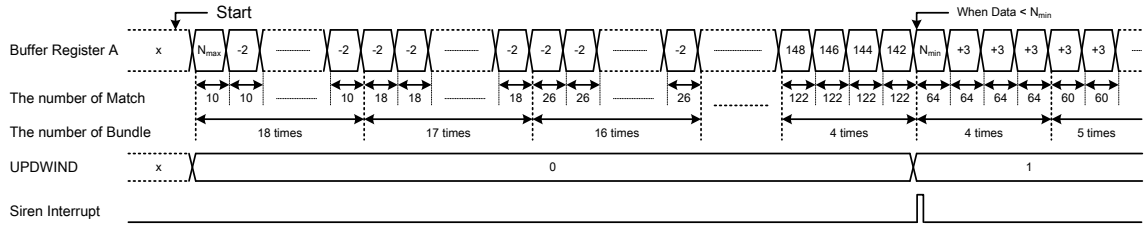

Figure 41. 16-bit PPG Mode Timing Chart of TIMER 1

## 11.4 Siren signal timing chart

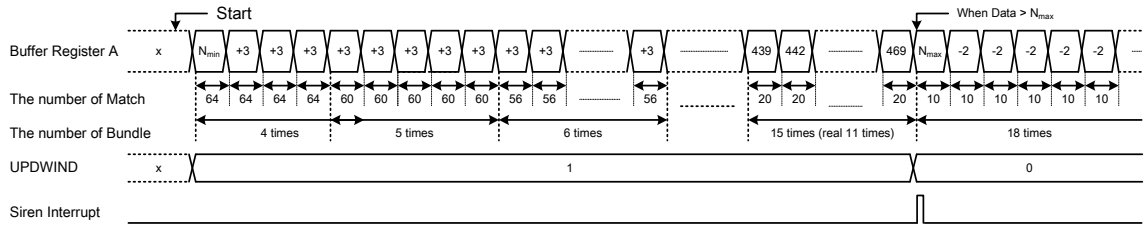
Value of the siren related registers are as follows:

- MAXDR: $N_{max}$
- MINDR: $N_{min}$
- DWMAT: 10, DWBNDL: 18, DWDECD: 2, DWINCM: 8
- UPMAT: 64, UPBNDL: 4, UPINCD: 3, UPDECM: 4

1. On down data (up frequency) with UPDW = 0




2. On up data (down frequency) with UPDW = 1

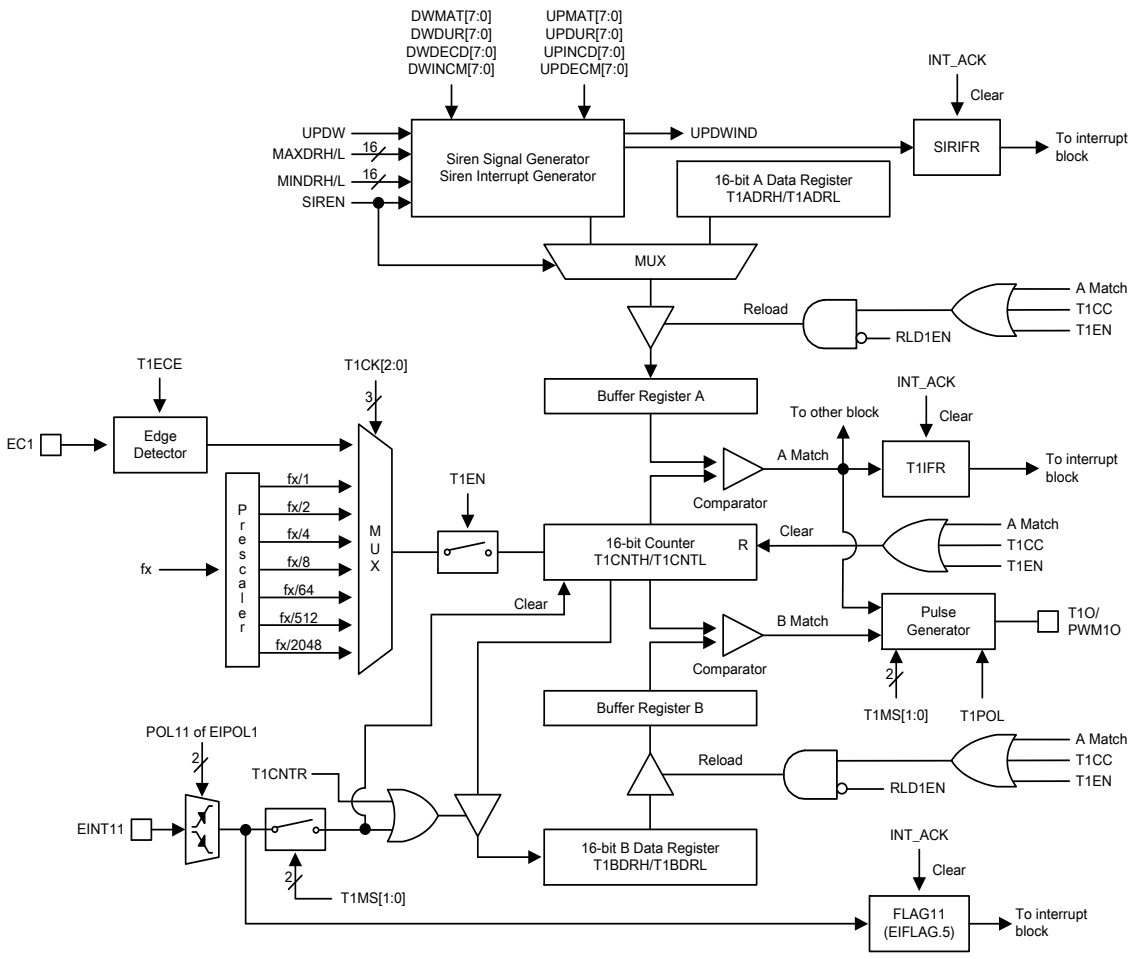



**NOTES**:

1. Where $N_{max}$ = 470 and $N_{min}$ = 142.
2. The match times and bundle times in the siren generator is equal to or greater than 1.

**Figure 42. Siren Signal Timing Chart**

## 11.5 Block diagram




**NOTES**:

1. T1EN is automatically cleared to logic '0' after one pulse is generated at a PPG one-shot mode.
2. RLD1EN bit must be '0b' when T1EN bit is changed from '0b' to '1b'.

**Figure 43. 16-bit Timer 1 in Block Diagram**

## 11.6 Register map

**Table 16. TIMER 1 Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| T1CRH | BBH | R/W | 00H | Timer 1 Control High Register |
| T1CRL | BAH | R/W | 00H | Timer 1 Control Low Register |
| T1ADRH | BDH | R/W | FFH | Timer 1 A Data High Register |
| T1ADRL | BCH | R/W | FFH | Timer 1 A Data Low Register |
| T1BDRH | BFH | R/W | FFH | Timer 1 B Data High Register |
| T1BDRL | BEH | R/W | FFH | Timer 1 B Data Low Register |
| SIRENCR | 1008H (XSFR) | R/W | 00H | Siren Control Register |
| MAXDRL | 100CH (XSFR) | R/W | FFH | Siren Max Data Low Register |
| MAXDRH | 100DH (XSFR) | R/W | FFH | Siren Max Data High Register |
| MINDRL | 100EH (XSFR) | R/W | 00H | Siren Min Data Low Register |
| MINDRH | 100FH (XSFR) | R/W | 00H | Siren Min Data High Register |
| DWMAT | 1010H (XSFR) | R/W | 00H | Siren down match times register |
| DWBNDL | 1011H (XSFR) | R/W | 00H | Siren down bundle times register |
| DWDECD | 1012H (XSFR) | R/W | 00H | Siren down decrement data register |
| DWINCM | 1013H (XSFR) | R/W | 00H | Siren down increment match times register |
| UPMAT | 1014H (XSFR) | R/W | 00H | Siren up match times register |
| UPBNDL | 1015H (XSFR) | R/W | 00H | Siren up bundle times register |
| UPINCD | 1016H (XSFR) | R/W | 00H | Siren up increment data register |
| UPDECM | 1017H (XSFR) | R/W | 00H | Siren up decrement match times register |

## 11.7 Timer/counter 1 Register description

**T1ADRH (Timer 1 A data High Register): BDH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1ADRH7 | T1ADRH6 | T1ADRH5 | T1ADRH4 | T1ADRH3 | T1ADRH2 | T1ADRH1 | T1ADRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T1ADRH[7:0] T1 A Data High Byte

### T1DRL (Timer 1 A Data Low Register): BCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1ADRL7 | T1ADRL6 | T1ADRL5 | T1ADRL4 | T1ADRL3 | T1ADRL2 | T1ADRL1 | T1ADRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T1ADRL[7:0] T1 A Data Low Byte

**NOTE:** Do not write "0000H" in the T1ADRH/T1ADRL register when PPG mode

### T1BDRH (Timer 1 B Data High Register): BFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1BDRH7 | T1BDRH6 | T1BDRH5 | T1BDRH4 | T1BDRH3 | T1BDRH2 | T1BDRH1 | T1BDRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T1BDRH[7:0] T1 B Data High Byte

### T1BDRL (Timer 1 B Data Low Register): BEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1BDRL7 | T1BDRL6 | T1BDRL5 | T1BDRL4 | T1BDRL3 | T1BDRL2 | T1BDRL1 | T1BDRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

T1BDRL[7:0] T1 B Data Low Byte

### T1CRH (Timer 1 Control High Register): BBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1EN | – | T1MS1 | T1MS0 | – | – | – | T1CC |
| R/W | – | R/W | R/W | – | – | – | R/W |

Initial value: 00H

T1EN Control Timer 1
- 0 Timer 1 disable
- 1 Timer 1 enable (Counter clear and start)

T1MS[1:0] Control Timer 1 Operation Mode

| T1MS1 | T1MS0 | Description |
|---|---|---|
| 0 | 0 | Timer/counter mode (T1O: toggle at A match) |
| 0 | 1 | Capture mode (The A match interrupt can occur) |
| 1 | 0 | PPG one-shot mode (PWM1O) |
| 1 | 1 | PPG repeat mode (PWM1O) |

T1CC Clear Timer 1 Counter
- 0 No effect
- 1 Clear the Timer 1 counter (When write, automatically cleared "0" after being cleared counter)

### T1CRL (Timer 1 Control Low Register): BAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| T1CK2 | T1CK1 | T1CK0 | T1IFR | RLD1EN | T1POL | T1ECE | T1CNTR |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

T1CK[2:0] Select Timer 1 clock source. fx is main system clock frequency

| T1CK2 | T1CK1 | T1CK0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | fx/2048 |
| 0 | 0 | 1 | fx/512 |
| 0 | 1 | 0 | fx/64 |
| 0 | 1 | 1 | fx/8 |
| 1 | 0 | 0 | fx/4 |
| 1 | 0 | 1 | fx/2 |
| 1 | 1 | 0 | fx/1 |
| 1 | 1 | 1 | External clock (EC1) |

T1IFR When T1 Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal. Writing "1" has no effect.
- 0 T1 Interrupt no generation
- 1 T1 Interrupt generation

RLD1EN Control Timer 1 Reload Signal
- 0 Enable timer n reload signal
- 1 Disable timer n reload signal

T1POL T1O/PWM1O Polarity Selection
- 0 Start High (T1O/PWM1O is low level at disable)
- 1 Start Low (T1O/PWM1O is high level at disable)

T1ECE Timer 1 External Clock Edge Selection
- 0 External clock falling edge
- 1 External clock rising edge

T1CNTR Timer 1 Counter Read Control
- 0 No effect
- 1 Load the counter value to the B data register (When write, automatically cleared "0" after being loaded)

### MAXDRH (Siren Max Data High Register): 100DH (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MAXDRH7 | MAXDRH6 | MAXDRH5 | MAXDRH4 | MAXDRH3 | MAXDRH2 | MAXDRH1 | MAXDRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

MAXDRH[7:0] Siren Max Data High Byte

### MAXDRL (Siren Max Data Low Register): 100CH (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MAXDRL7 | MAXDRL6 | MAXDRL5 | MAXDRL4 | MAXDRL3 | MAXDRL2 | MAXDRL1 | MAXDRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: FFH

MAXDRH[7:0] Siren Max Data High Byte

**NOTE:** The value of the siren max data register (MAXDR) should be greater than the value of the siren min data register (MINDR). The MAXDR register has the value for the lowest frequency in frequency modulation. The higher the value, the lower the frequency.

### MINDRH (Siren Min Data High Register): 100FH (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MINDRH7 | MINDRH6 | MINDRH5 | MINDRH4 | MINDRH3 | MINDRH2 | MINDRH1 | MINDRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

MINDRH[7:0] Siren Min Data High Byte

### MINDRL (Siren Min Data Low Register): 100EH (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MINDRL7 | MINDRL6 | MINDRL5 | MINDRL4 | MINDRL3 | MINDRL2 | MINDRL1 | MINDRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

MINDRL[7:0] Siren Max Data Low Byte

**NOTE**: The MINDR register has the value for the highest frequency in frequency modulation.

### DWMAT (Siren Down Match Times Register): 1010H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DWMAT7 | DWMAT6 | DWMAT5 | DWMAT4 | DWMAT3 | DWMAT2 | DWMAT1 | DWMAT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

DWMAT[7:0] The match times during same data when down data (up frequency). The range is 01H to FFH.

### DWBNDL (Siren Down Bundle Times Register): 1011H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DWBNDL7 | DWBNDL6 | DWBNDL5 | DWBNDL4 | DWBNDL3 | DWBNDL2 | DWBNDL1 | DWBNDL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

DWBNDL[7:0] The bundle times during same match times when down data (up frequency). The range is 01H to FFH.

### DWDECD (Siren Down Decrement Data Register): 1012H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DWDECD7 | DWDECD6 | DWDECD5 | DWDECD4 | DWDECD3 | DWDECD2 | DWDECD1 | DWDECD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

DWDECD[7:0] The decrement values of data every match times when down data (up frequency). The range is 00H to FFH.

### DWINCM (Siren Down Increment Match Times Register): 1013H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DWINCM7 | DWINCM6 | DWINCM5 | DWINCM4 | DWINCM3 | DWINCM2 | DWINCM1 | DWINCM0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

DWINCM[7:0] The increment values of match times every bundle times when down data (up frequency). The range is 00H to FFH.

### UPMAT (Siren Up Match Times Register): 1014H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UPMAT7 | UPMAT6 | UPMAT5 | UPMAT4 | UPMAT3 | UPMAT2 | UPMAT1 | UPMAT0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| UPMAT [7:0] | The match times during same data when up data (down frequency). The range is 01H to FFH. |

### UPBNDL (Siren Up Duration Times Register): 1015H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UPBNDL7 | UPBNDL6 | UPBNDL5 | UPBNDL4 | UPBNDL3 | UPBNDL2 | UPBNDL1 | UPBNDL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| UPBNDL [7:0] | The bundle times during same match times when up data (down frequency). The range is 01H to FFH. |

### UPINCD (Siren Up Increment Data Register): 1016H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UPINCD7 | UPINCD6 | UPINCD5 | UPINCD4 | UPINCD3 | UPINCD2 | UPINCD1 | UPINCD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| UPINCD [7:0] | The increment values of data every match times when up data (down frequency). The range is 00H to FFH. |

### UPDECM (Siren Up Decrement Match Times Register): 1017H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| UPDECM7 | UPDECM6 | UPDECM5 | UPDECM4 | UPDECM3 | UPDECM2 | UPDECM1 | UPDECM0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| UPDECM [7:0] | The decrement values of match times every bundle times when up data (down frequency). The range is 00H to FFH |

**SIRENCR (Siren Control Register): 1008H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | SIRIFR | UPDWIND | – | UPDW | – | SIREN |
| – | – | R/W | R | – | R/W | – | R/W |

Initial value: 00H

SIRIFR When the UPDWIND bit is changed (0/1 to 1/0) after the enable of siren, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal.
- 0 Siren Interrupt no generation
- 1 Siren Interrupt generation

UPDWIND Siren Data Up/Down Indicator
- 0 On down data (up frequency)
- 1 On up data (down frequency)

UPDW Siren Up/Down Initial Setting bit
- 0 Down data at start (up frequency)
- 1 Up data at start (down frequency)

SIREN Control Siren bit
- 0 Disable siren (frequency modulation)
- 1 Enable siren (frequency modulation)

**NOTES:**

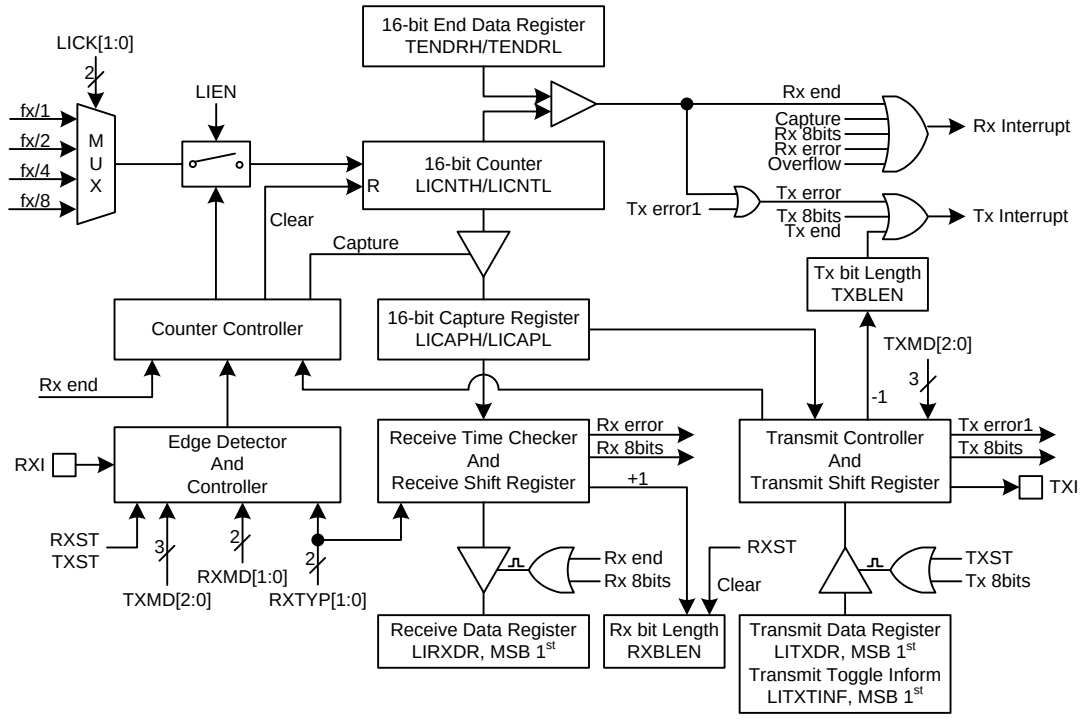
1. Siren function can be operated during the enable of the timer 1 and the timer/counter mode should be configured by T1CRH, T1MS[1:0] bits. After all set of the siren related register, the timer 1 should be enabled.
2. To load new data to the siren related register, the timer 1 should be enabled.
3. Clear the SIREN bit of SIRENCR register to "0b" to disable siren.
4. Wait for 2 clocks or more of timer's clock.
   Ex) If the timer's clock is fx/64, 2x64 = 128 clocks.
5. Clear the T1EN bit of T1CRH register to "0b" to disable the timer 1.
6. Load new data to the registers.
7. Enable siren, and then timer 1.

# 12 Line interface

A96L322 offers two operating modes for line interfaces as shown in the followings:

- Receive mode (RX Types 0~2)
- Transmit mode (TX Mode 0~4)

## 12.1 Block diagram

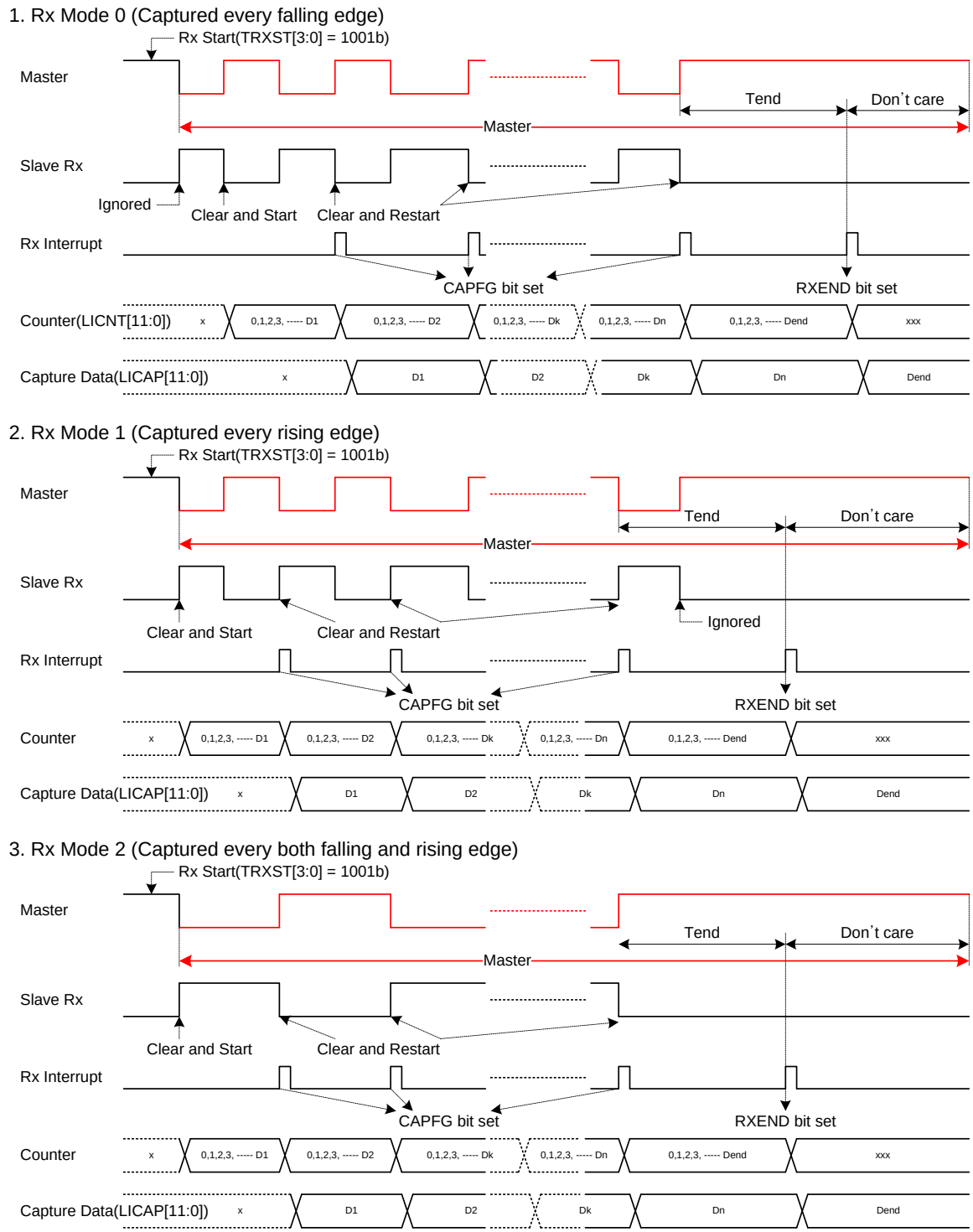



**NOTES**:

1. TENDR register must have larger number than TMAXR register has.
2. A96L322 transmits and receives MSB 1st always.
3. Time calculation registers are calculated by n+1.

**Figure 44. Line Interface in Block Diagram**

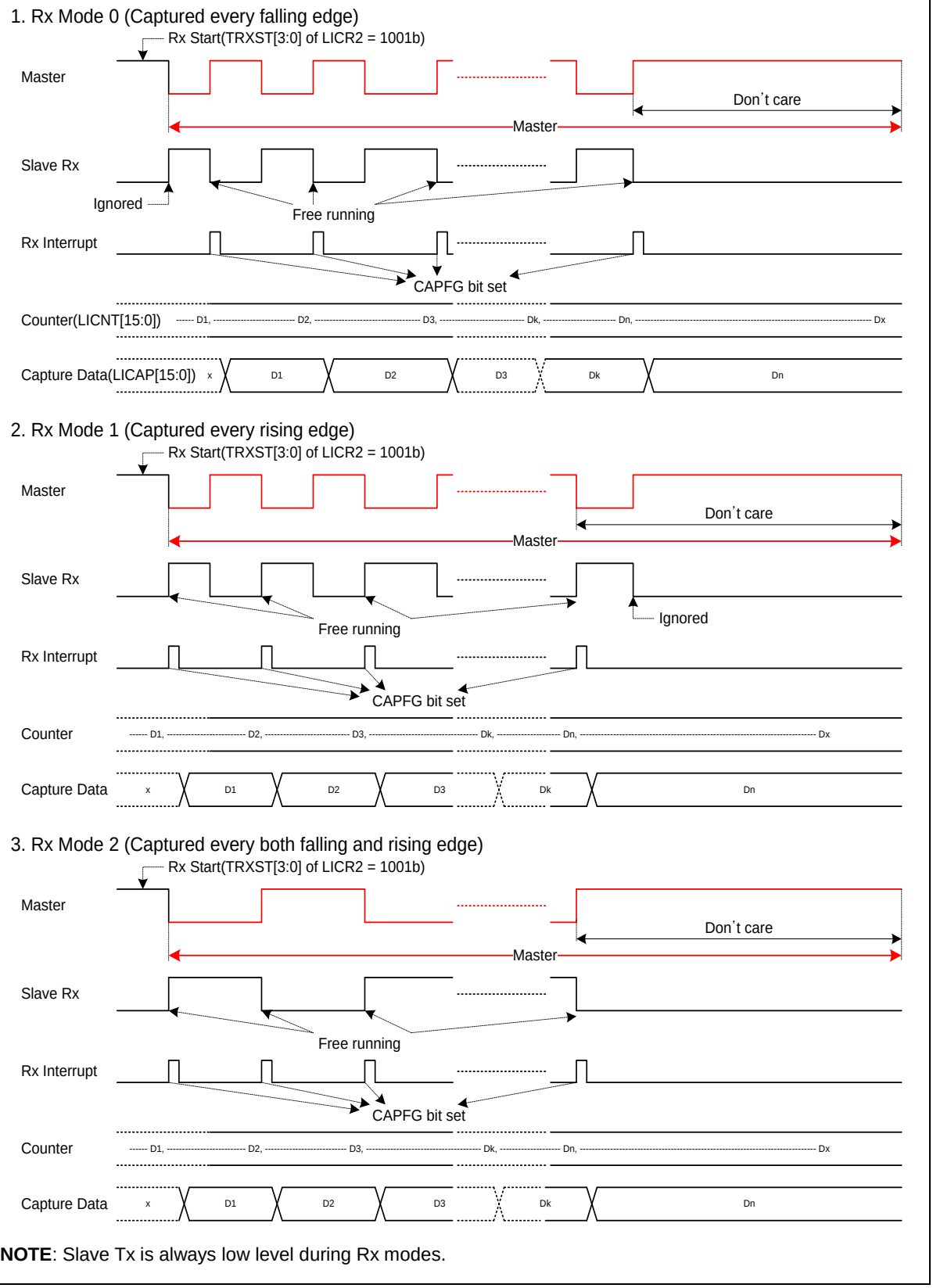
## 12.2 Line interface timing chart




**NOTES**:

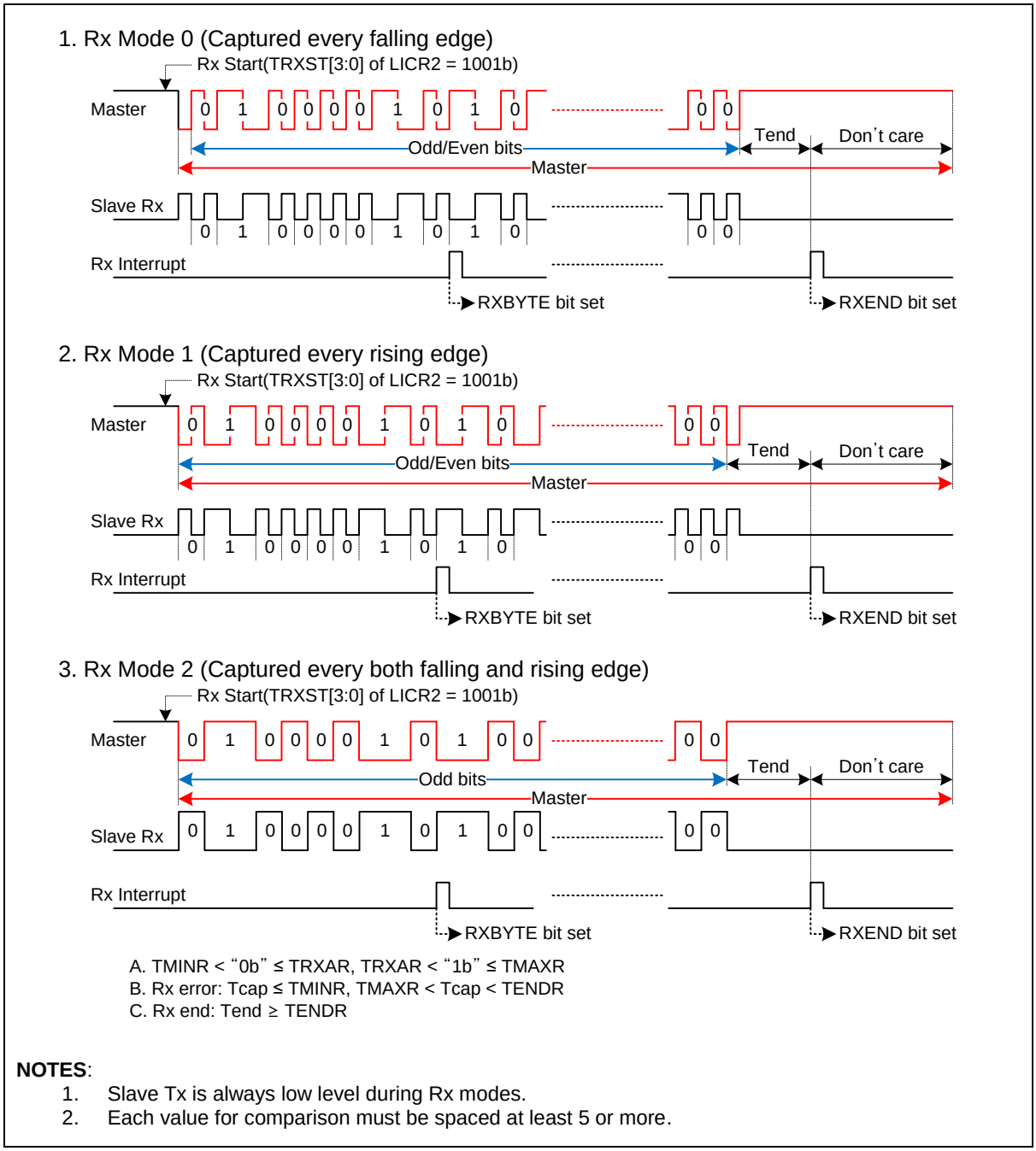
1. Rx end: Tend ≥ TENDR
2. The slave Tx is always low level during Rx modes.
3. The TENDR register should be greater than or equal to 0x000A.

**Figure 45. Rx Type 0 Timing Chart (Counter Clear/ Restart at Valid Edge)**




**NOTE**: Slave Tx is always low level during Rx modes.

**Figure 46. Rx Type 1 Timing Chart (Counter Free Running)**




**NOTES**:

1. Slave Tx is always low level during Rx modes.
2. Each value for comparison must be spaced at least 5 or more.

**Figure 47. Rx Type 2 Timing Chart (Receive Bits by H/W)**

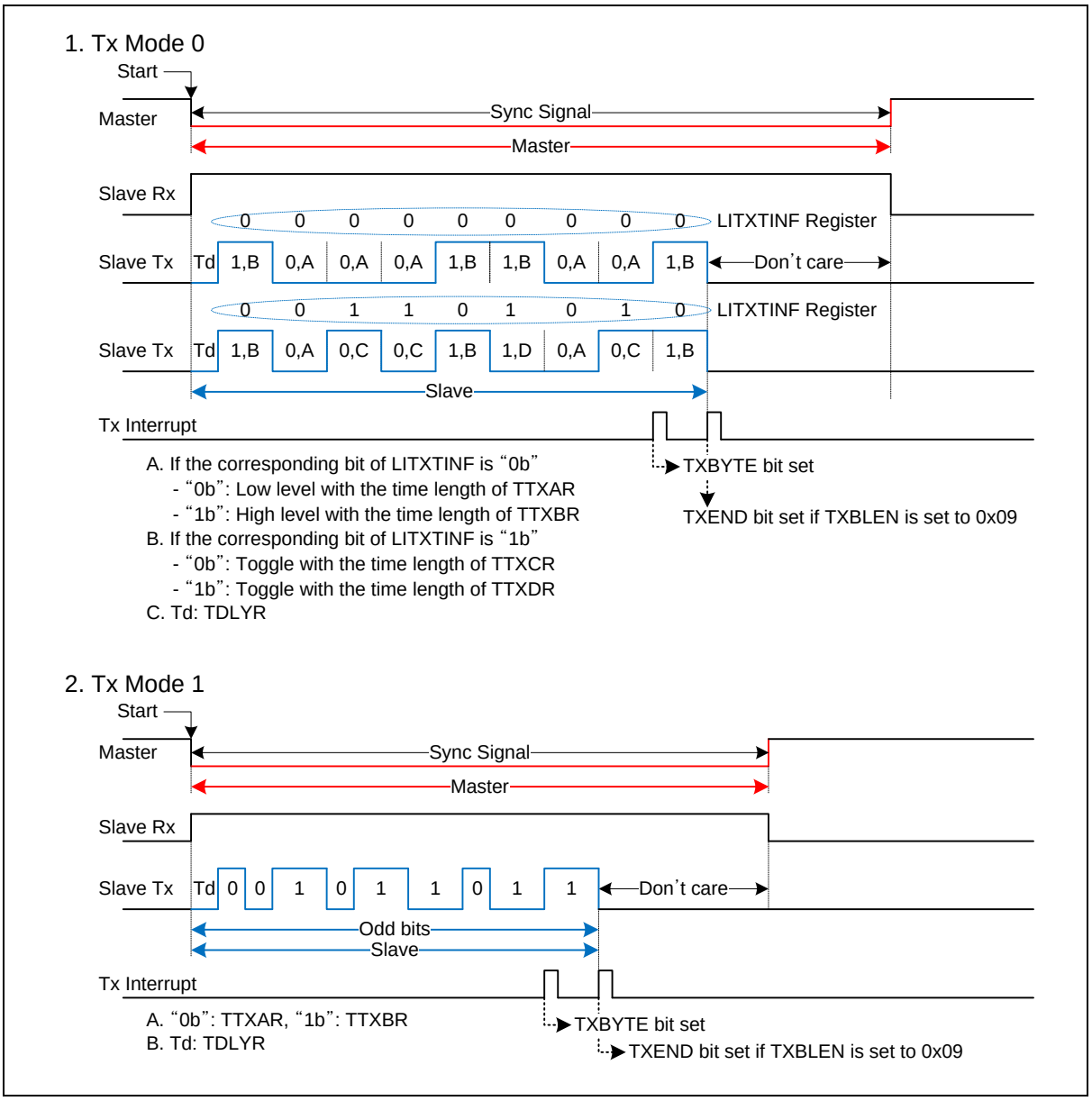
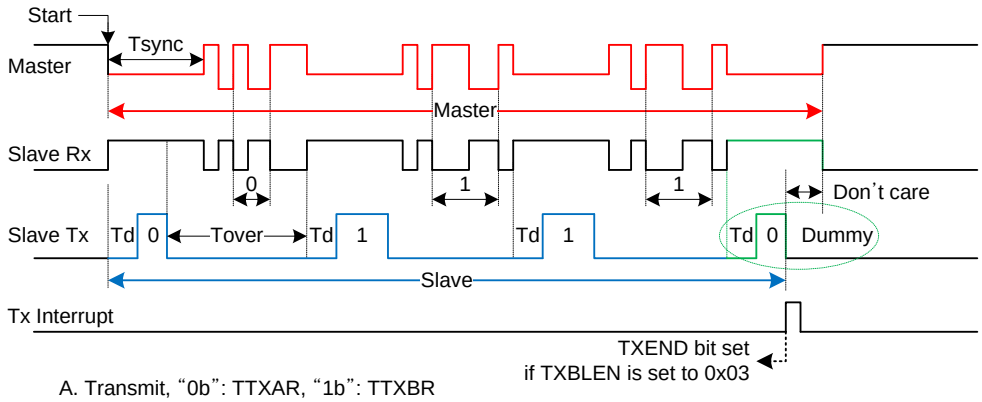


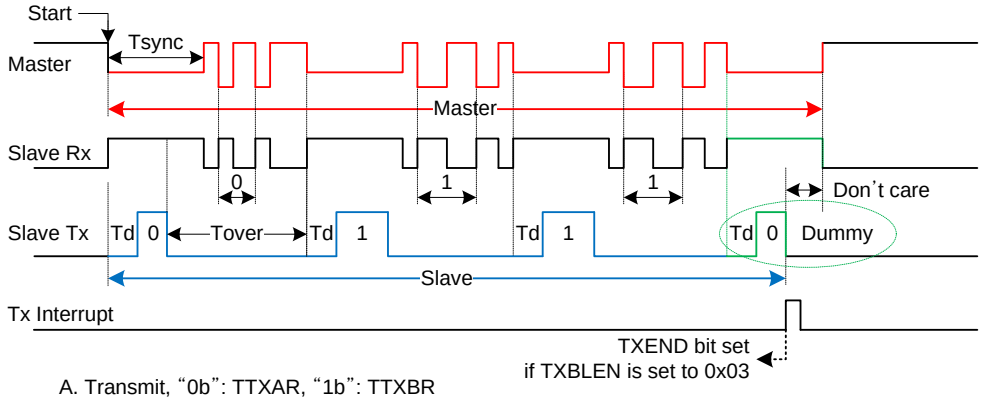

**Figure 48. Tx Mode Timing Chart (Mode 0 and Mode 1)**

3. Tx Mode 2 (Check response at falling edge)



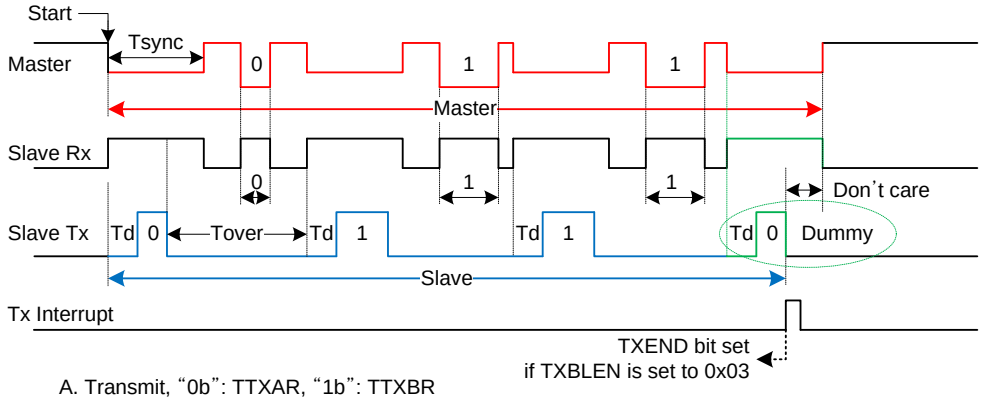

A. Transmit, "0b": TTXAR, "1b": TTXBR
B. Response, TMINR < "0b" ≤ TTXRR, TTXRR < "1b" ≤ TMAXR
C. Td: TDLYR
D. Tx error: Tbit ≤ TMINR, Tbit > TMAXR, Tover ≥ TENDR, Tx bit ≠ Response bit

4. Tx Mode 3 (Check response at rising edge)




A. Transmit, "0b": TTXAR, "1b": TTXBR
B. Response, TMINR < "0b" ≤ TTXRR, TTXRR < "1b" ≤ TMAXR
C. Td: TDLYR
D. Tx error: Tbit ≤ TMINR, Tbit > TMAXR, Tover ≥ TENDR, Tx bit ≠ Response bit

5. Tx Mode 4 (Check response at both falling and rising edge)




A. Transmit, "0b": TTXAR, "1b": TTXBR
B. Response, TMINR < "0b" ≤ TTXRR, TTXRR < "1b" ≤ TMAXR
C. Td: TDLYR
D. Tx error: Tbit ≤ TMINR, Tbit > TMAXR, Tover ≥ TENDR, Tx bit ≠ Response bit

**NOTE**: Each value for comparison must be spaced at least 5 or higher.

**Figure 49. Tx Modes Timing Chart (Mode 2, Mode 3, and Mode 4)**

## 12.3 Register map

Table 17. Line Interface Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| LICR0 | C1H | R/W | 00H | Line Interface Control Register 0 |
| LICR1 | C2H | R/W | 00H | Line Interface Control Register 1 |
| LICR2 | C3H | R/W | 00H | Line Interface Control Register 2 |
| LICAPH | C5H | R | 00H | Line Interface Capture Data High Register |
| LICAPL | C4H | R | 00H | Line Interface Capture Data Low Register |
| TDLYRH | C7H | R/W | 00H | Delay Time Data High Register |
| TDLYRL | C6H | R/W | 00H | Delay Time Data Low Register |
| TTXARH | D3H | R/W | 00H | Transmit Time A Data High Register |
| TTXARL | D2H | R/W | 00H | Transmit Time A Data Low Register |
| TTXBRH | D5H | R/W | 00H | Transmit Time B Data High Register |
| TTXBRL | D4H | R/W | 00H | Transmit Time B Data Low Register |
| TTXCRH | DBH | R/W | 00H | Transmit Time C Data High Register |
| TTXCRL | DAH | R/W | 00H | Transmit Time C Data Low Register |
| TTXDRH | DDH | R/W | 00H | Transmit Time D Data High Register |
| TTXDRL | DCH | R/W | 00H | Transmit Time D Data Low Register |
| TTXRRH | DFH | R/W | 00H | Transmit Time Rx Data High Register |
| TTXRRL | DEH | R/W | 00H | Transmit Time Rx Data Low Register |
| TRXARH | E3H | R/W | 00H | Receive Time A Data High Register |
| TRXARL | E2H | R/W | 00H | Receive Time A Data Low Register |
| TMINRH | EBH | R/W | 00H | Minimum Time Data High Register |
| TMINRL | EAH | R/W | 00H | Minimum Time Data Low Register |
| TMAXRH | EDH | R/W | 00H | Maximum Time Data High Register |
| TMAXRL | ECH | R/W | 00H | Maximum Time Data Low Register |
| TENDRH | EFH | R/W | 00H | End Time Data High Register |
| TENDRL | EEH | R/W | 00H | End Time Data Low Register |
| RXBLEN | E9H | R | 00H | Receive bits Length Counter |
| LIRXDR | E1H | R | 00H | Line Interface Receive Data Register |
| TXBLEN | D9H | R/W | 00H | Transmit bits Length Counter |
| LITXDR | D1H | R/W | 00H | Line Interface Transmit Data Register |
| LITXTINF | C9H | R/W | 00H | Line Interface Transmit Toggle Information Register |
| LISTATR | C0H | R/W | 00H | Line Interface Status Register |

## 12.4 Register description

### LITXDR (Line Interface Transmit Data Register): D1H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LITXDR7 | LITXDR6 | LITXDR5 | LITXDR4 | LITXDR3 | LITXDR2 | LITXDR1 | LITXDR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| LITXDR[7:0] | Line Interface Transmit Data Byte<br>The last bits should be written to the MSB. Ex) If the last 5-bits are to be transmitted, the bits should be written to LITXDR[7:3]. |

### TXBLEN (Transmit Bits Length Counter): D9H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TXBLEN7 | TXBLEN6 | TXBLEN5 | TXBLEN4 | TXBLEN3 | TXBLEN2 | TXBLEN1 | TXBLEN0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| TXBLEN[7:0] | Transmit Bits Length Counter Byte<br>Write 'n' in this register to transmit n-bits. The register is decreased by one every transmitted bit. The transmission is over if the register reaches 0x00. |

### LIRXDR (Line Interface Receive Data Register): E1H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LIRXDR7 | LIRXDR6 | LIRXDR5 | LIRXDR4 | LIRXDR3 | LIRXDR2 | LIRXDR1 | LIRXDR0 |
| R | R | R | R | R | R | R | R |

Initial value: 00H

| | |
|---|---|
| LIRXDR[7:0] | Line Interface Receive Data Byte<br>The LSB is valid when the last bits are smaller than 8-bits.<br>Ex) If the last 3-bits are received, the LIRXDR[2:0] is available. |

### RXBLEN (Receive Bits Length Counter): E9H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| RXBLEN7 | RXBLEN6 | RXBLEN5 | RXBLEN4 | RXBLEN3 | RXBLEN2 | RXBLEN1 | RXBLEN0 |
| R | R | R | R | R | R | R | R |

Initial value: 00H

| | |
|---|---|
| RXBLEN[7:0] | Receive Bits Length Counter Byte<br>This register is automatically cleared to '0' by writing "1001b" to TRXST[3:0] bits in a line interface control register. The register is increased by one every received bit. This register has "0x15" if the 21-bits are received. |

### LICAPH (Line Interface Capture Data High Register): C5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | LICAPH3 | LICAPH2 | LICAPH1 | LICAPH0 |
| – | – | – | – | R | R | R | R |

Initial value: 00H

| | |
|---|---|
| LICAPH[3:0] | Line Interface Capture Data High Byte |

### LICAPL (Line Interface Capture Data Low Register): C4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LICAPL7 | LICAPL6 | LICAPL5 | LICAPL4 | LICAPL3 | LICAPL2 | LICAPL1 | LICAPL0 |
| R | R | R | R | R | R | R | R |

Initial value: 00H

LICAPL[7:0] Line Interface Capture Data Low Byte

### TDLYRH (Delay Time Data High Register): C7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TDLYRH3 | TDLYRH2 | TDLYRH1 | TDLYRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TDLYRH[3:0] Delay Time Data High Byte

### TDLYRL (Delay Time Data Low Register): C6H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TDLYRL7 | TDLYRL6 | TDLYRL5 | TDLYRL4 | TDLYRL3 | TDLYRL2 | TDLYRL1 | TDLYRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TDLYRL[7:0] Delay Time Data Low Byte

### TTXARH (Transmit Time A Data High Register): D3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TTXARH3 | TTXARH2 | TTXARH1 | TTXARH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXARH[3:0] Transmit Time A Data High Byte

### TTXARL (Transmit Time A Data Low Register): D2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTXARL7 | TTXARL6 | TTXARL5 | TTXARL4 | TTXARL3 | TTXARL2 | TTXARL1 | TTXARL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXARL[7:0] Transmit Time A Data Low Byte

### TTXBRH (Transmit Time B Data High Register): D5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TTXBRH3 | TTXBRH2 | TTXBRH1 | TTXBRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXBRH[3:0] Transmit Time B Data High Byte

### TTXBRL (Transmit Time B Data Low Register): D4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTXBRL7 | TTXBRL6 | TTXBRL5 | TTXBRL4 | TTXBRL3 | TTXBRL2 | TTXBRL1 | TTXBRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXBRL[7:0] Transmit Time B Data Low Byte

### TTXCRH (Transmit Time C Data High Register): DBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TTXCRH3 | TTXCRH2 | TTXCRH1 | TTXCRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXCRH[3:0] Transmit Time C Data High Byte

### TTXCRL (Transmit Time C Data Low Register): DAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTXCRL7 | TTXCRL6 | TTXCRL5 | TTXCRL4 | TTXCRL3 | TTXCRL2 | TTXCRL1 | TTXCRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXCRL[7:0] Transmit Time C Data Low Byte

### TTXDRH (Transmit Time D Data High Register): DDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TTXDRH3 | TTXDRH2 | TTXDRH1 | TTXDRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXDRH[3:0] Transmit Time D Data High Byte

### TTXDRL (Transmit Time D Data Low Register): DCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTXDRL7 | TTXDRL6 | TTXDRL5 | TTXDRL4 | TTXDRL3 | TTXDRL2 | TTXDRL1 | TTXDRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXDRL[7:0] Transmit Time D Data Low Byte

### TTXRRH (Transmit Time Rx Data High Register): DFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TTXRRH3 | TTXRRH2 | TTXRRH1 | TTXRRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXRRH[3:0] Transmit Time Rx Data High Byte

### TTXRRL (Transmit Time Rx Data Low Register): DEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TTXRRL7 | TTXRRL6 | TTXRRL5 | TTXRRL4 | TTXRRL3 | TTXRRL2 | TTXRRL1 | TTXRRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TTXRRL[7:0] Transmit Time D Data Low Byte

### TRXARH (Receive Time A Data High Register): E3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TRXARH3 | TRXARH2 | TRXARH1 | TRXARH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXARH[3:0] Receive Time A Data High Byte

**TRXARL (Receive Time A Data Low Register): E2H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TRXARL7 | TRXARL6 | TRXARL5 | TRXARL4 | TRXARL3 | TRXARL2 | TRXARL1 | TRXARL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXARL[7:0] Receive Time A Data Low Byte

**TRXBRH (Receive Time B Data High Register): E5H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TRXBRH3 | TRXBRH2 | TRXBRH1 | TRXBRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXBRH[3:0] Receive Time B Data High Byte

**TRXBRL (Receive Time B Data Low Register): E4H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TRXBRL7 | TRXBRL6 | TRXBRL5 | TRXBRL4 | TRXBRL3 | TRXBRL2 | TRXBRL1 | TRXBRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXBRL[7:0] Receive Time B Data Low Byte

**TRXCRH (Receive Time C Data High Register): E7H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TRXCRH3 | TRXCRH2 | TRXCRH1 | TRXCRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXCRH[3:0] Receive Time C Data High Byte

**TRXCRL (Receive Time C Data Low Register): E6H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TRXCRL7 | TRXCRL6 | TRXCRL5 | TRXCRL4 | TRXCRL3 | TRXCRL2 | TRXCRL1 | TRXCRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXCRL[7:0] Receive Time C Data Low Byte

**TMINRH (Minimum Time Data High Register): EBH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TMINRH3 | TMINRH2 | TMINRH1 | TMINRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TMINRH[3:0] Minimum Time Data High Byte

**TMINRL (Minimum Time Data Low Register): EAH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TMINRL7 | TMINRL6 | TMINRL5 | TMINRL4 | TMINRL3 | TMINRL2 | TMINRL1 | TMINRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TMINRL[7:0] Minimum Time Data Low Byte

### TMAXRH (Maximum Time Data High Register): EDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TMAXRH3 | TMAXRH2 | TMAXRH1 | TMAXRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TMAXRH[3:0] Maximum Time Data High Byte

### TMAXRL (Maximum Time Data Low Register): ECH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TMAXRL7 | TMAXRL6 | TMAXRL5 | TMAXRL4 | TMAXRL3 | TMAXRL2 | TMAXRL1 | TMAXRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TMAXRL[7:0] Maximum Time Data Low Byte

### TENDRH (End Time Data High Register): EFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TENDRH3 | TENDRH2 | TENDRH1 | TENDRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TENDRH[3:0] End Time Data High Byte

### TENDRL (End Time Data Low Register): EEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| TENDRL7 | TENDRL6 | TENDRL5 | TENDRL4 | TENDRL3 | TENDRL2 | TENDRL1 | TENDRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

TENDRL[7:0] End Time Data Low Byte

### LICR0 (Line Interface Control Register 0): C1H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LIEN | LICK1 | LICK0 | – | RXTYP1 | RXTYP0 | RXMD1 | RXMD0 |
| R/W | R/W | R/W | – | R/W | R/W | R/W | R/W |

Initial value: 00H

LIEN — Line Interface Control bit

| | |
|---|---|
| 0 | Disable line interface block (Clear TRXST[3:0] of LICR2) |
| 1 | Enable line interface block |

LICK[1:0] — Line Interface Clock Selection bits

| LICK1 | LICK0 | Description |
|---|---|---|
| 0 | 0 | fx/1 |
| 0 | 1 | fx/2 |
| 1 | 0 | fx/4 |
| 1 | 1 | fx/8 |

RXTYP[1:0] — Receive Type Selection bits

| RXTYP1 | RXTYP0 | Description |
|---|---|---|
| 0 | 0 | Rx type 0, Counter clear/restart at valid edge<br>(Rx end and capture interrupts can occur) |
| 0 | 1 | Rx type 1, Counter free running<br>(Capture/overflow interrupt can occur) |
| 1 | 0 | Rx type 2, Receive bits by H/W<br>(Rx end, error, and 8bits interrupts can occur) |
| 1 | 1 | Not available |

RXMD[1:0] — Receive Mode Selection bits

| RXMD1 | RXMD0 | Description |
|---|---|---|
| 0 | 0 | Rx mode 0, Captured every falling edge |
| 0 | 1 | Rx mode 1, Captured every rising edge |
| 1 | 0 | Rx mode 2, Captured every both falling and rising edge |
| 1 | 1 | Not available |

### LICR1 (Line Interface Control Register 1): C2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | TXMD2 | TXMD1 | TXMD0 |
| – | – | – | – | – | R/W | R/W | R/W |

Initial value: 00H

TXMD[2:0] Transmit Mode Selection bits

| TXMD2 | TXMD1 | TXMD0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Tx mode 0, level/toggle (Tx end/8bits interrupts can occur) |
| 0 | 0 | 1 | Tx mode 1, time length (Tx end/8bits interrupts can occur) |
| 0 | 1 | 0 | Tx mode 2, time length and check response at falling edge (Tx end/8bits interrupts can occur) |
| 0 | 1 | 1 | Tx mode 3, time length and check response at rising edge (Tx end/8bits interrupts can occur) |
| 1 | 0 | 0 | Tx mode 4, time length and check response at both falling edge and rising edge (Tx end/8bits interrupts can occur) |
| Other values | | | Not available |

### LICR2 (Line Interface Control Register 2): C3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | TRXST3 | TRXST2 | TRXST1 | TRXST0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

TRXST[3:0] Transmit and Receive Start Signal

| | |
|---|---|
| Others | No effect |
| 0110b | Start transmission and load the LITXDR to transmit shift register (These bits are automatically cleared '0' after Tx end or Tx error) |
| 1001b | Start reception and clear the RXBLEN register These bits are automatically cleared '0' after Rx end or Rx error) |

### LISTATR (Line Interface Status Register): C0H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| OVERFG | TXBYTE | TXERR | TXEND | CAPFG | RXBYTE | RXERR | RXEND |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

OVERFG When the counter overflows, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 No overflow
- 1 Overflow occurs

TXBYTE When 8bits have been transmitted, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 On transmission or No transmission
- 1 8bits have been transmitted

TXERR When Tx error occurs on transmission, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 No transmission error
- 1 Transmission error

TXEND When all bits have been transmitted, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 On transmission or No transmission
- 1 Transmission is over

CAPFG When the corresponding edge occurs on Rx mode 2 and 3, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 No capture
- 1 Capture occurs

RXBYTE When 8bits have been received, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 On reception or No reception
- 1 8bits have been received

RXERR When Rx error occurs on reception, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 No reception error
- 1 Reception error

RXEND When all bits have been received, this bit becomes '1'. For clearing bit, write '0' to this bit. So, the bit should be cleared by software.
- 0 On reception or No reception
- 1 Reception is over

# 13 10-bit A/D Converter

Analog-to-digital (A/D) converter allows conversion of an analog input signal to a corresponding 10-bit digital output. The A/D module has 11 analog inputs, and the output of the multiplexer becomes the input into the converter, which generates a result via successive approximation.

The A/D module incorporates four registers as listed in the followings. Each register can be selected for the corresponding channel by setting ADSEL[3:0]. When conversion is completed, two registers ADCDRH and ADCDRL contain the results of the conversion, the conversion status bit AFLAG is set to '1', and A/D interrupt is set. During the A/D conversion, AFLAG bit is read as '0'.

- A/D converter control high register (ADCCRH)
- A/D converter control low register (ADCCRL)
- A/D converter data high register (ADCDRH)
- A/D converter data low register (ADCDRL)

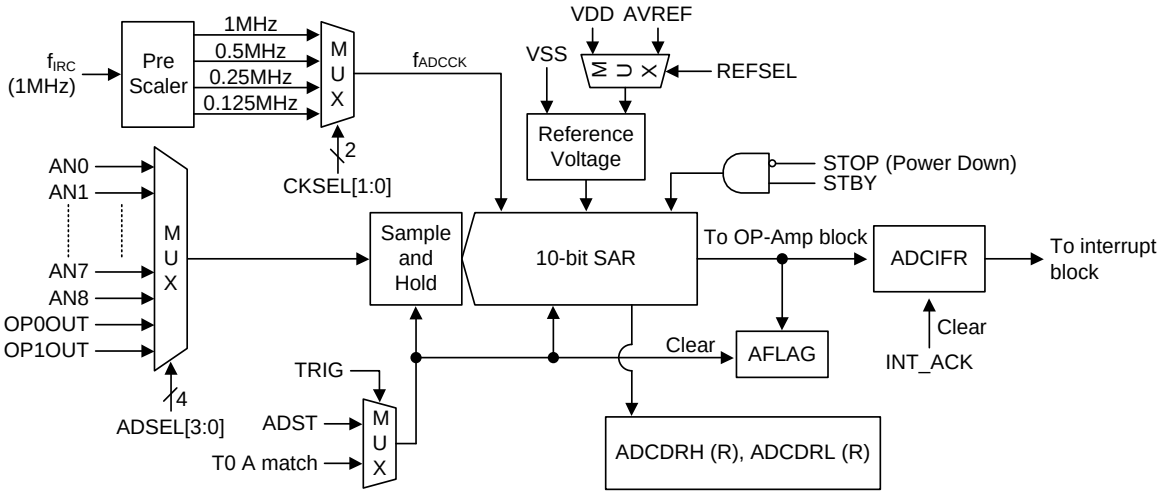
## 13.1 Conversion timing

A/D conversion process requires 6 clocks to sample and hold, 2 steps (2 clock edges) to convert each bit, and 2 clocks to set up A/D conversion. Therefore, total of 28 clocks are required to complete a 10-bit conversion.

For example, in a case that conversion clock operates with a 1MHz ADC clock frequency, one clock cycle is 1 us. Each bit conversion requires 2 clocks, the conversion rate is calculated as follows:

'6 clocks for S&H' + '2 clocks/bit × 10 bits' + set-up time = 28 clocks
28 clock × 1 us = 28 us at 1 MHz

## 13.2 Block diagram




**NOTE**: The conversion clock is 26-clock including ADC sample and hold.

**Figure 50. 10-bit ADC Block Diagram**

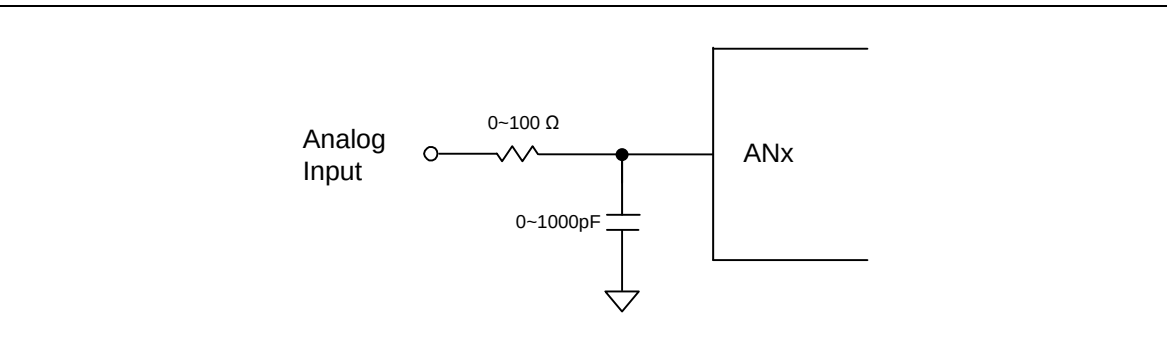



**Figure 51. AD Analog Input Pin with Capacitor**

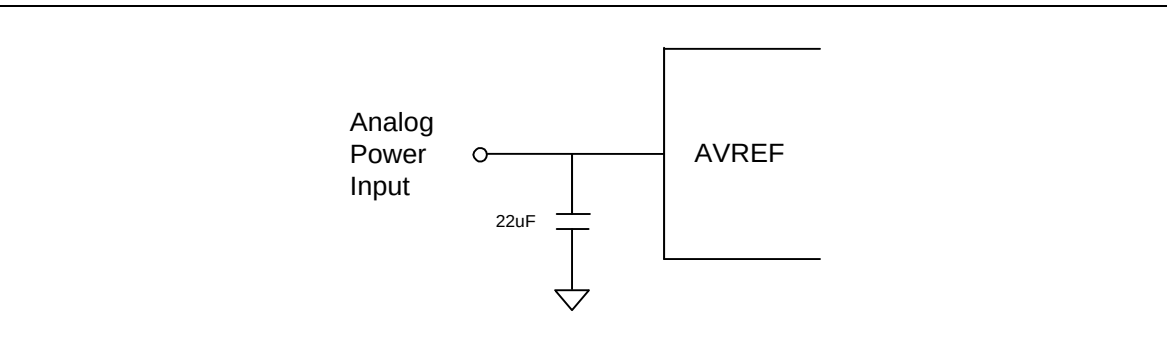



**Figure 52. AD Power (AVREF) Pin with Capacitor**

## 13.3 ADC operation

In this section, ADC operation is described through figures 4 to 6. As shown in figure 53, ADC conversion starts after configuring ADC Control High/ Low registers. By checking AFLAG, it is defined whether the conversion is completed or not. If AFLG is '1', the conversion is completed and ADC Data High/ Low registers are read to finish ADC operation.

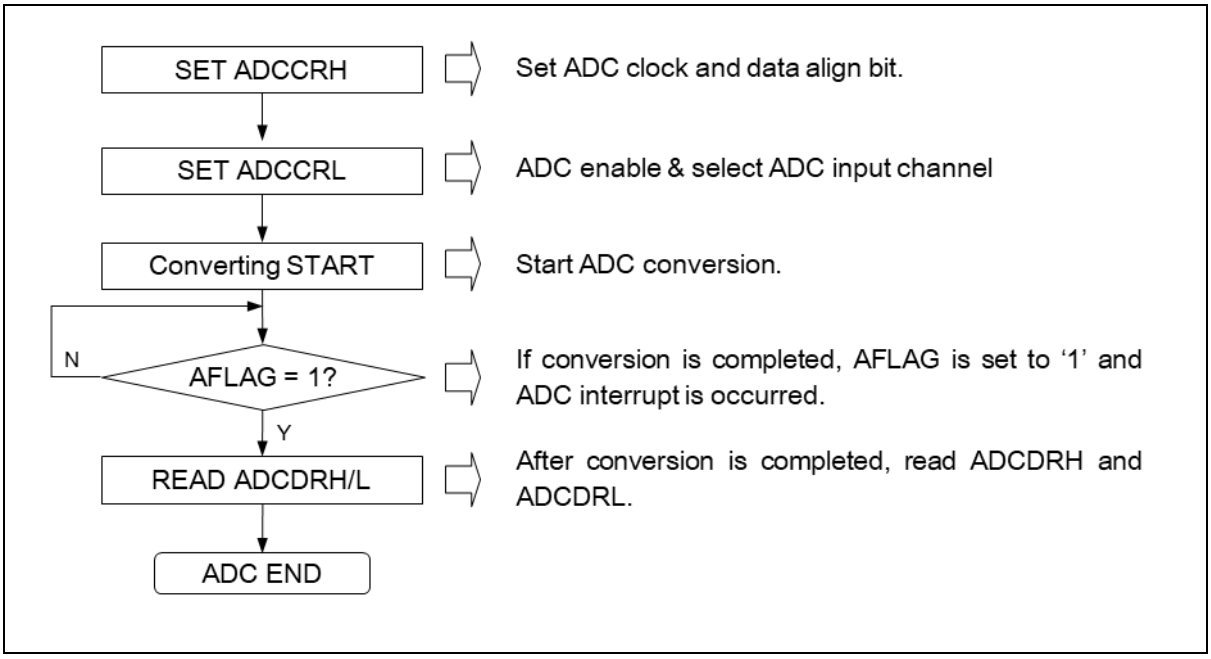



Figure 53. ADC Operation Flow

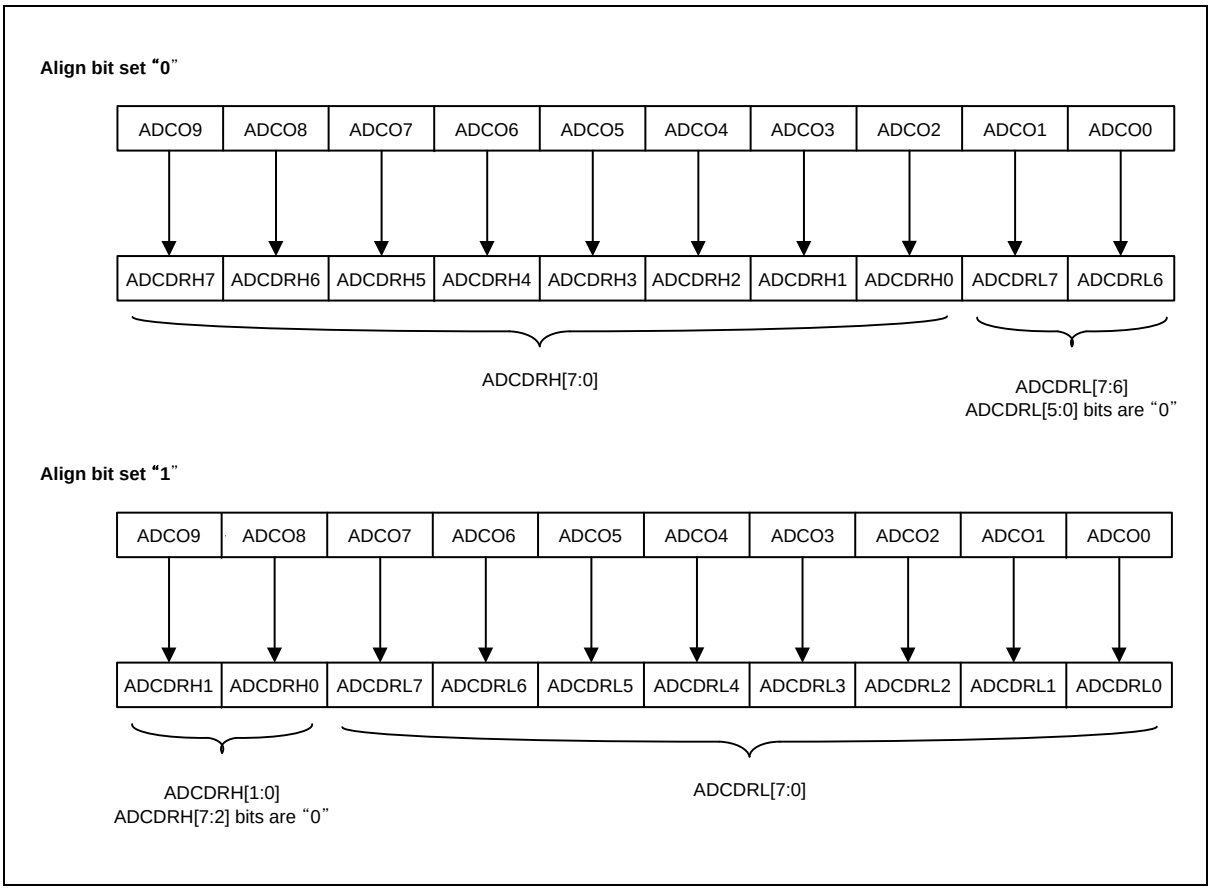



Figure 54. ADC Operation for Align Bit

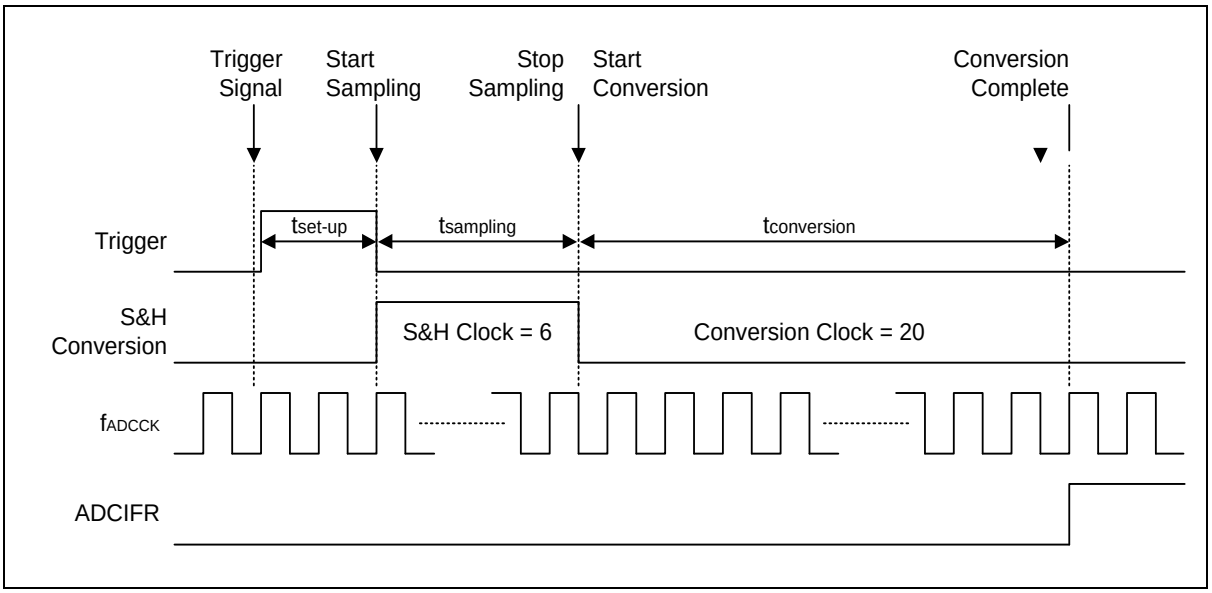



Figure 55. ADC Timing Chart

## 13.4 Register map

**Table 18. 10-bit ADC Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| ADCCRH | CBH | R/W | 00H | A/D Converter Control High Register |
| ADCCRL | CAH | R/W | 00H | A/D Converter Control Low Register |
| ADCDRH | CDH | R | xxH | A/D Converter Data High Register |
| ADCDRL | CCH | R | xxH | A/D Converter Data Low Register |

## 13.5 Register description

### ADCDRH (A/D Converter Data High Register): CDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADDM9 | ADDM8 | ADDM7 | ADDM6 | ADDM5 | ADDM4 | ADDM3<br>ADDL9 | ADDM2<br>ADDL8 |
| R | R | R | R | R | R | R | R |

Initial value: xxH

ADDM[9:2] MSB align, A/D Converter High Result (8-bit)
ADDL[9:8] LSB align, A/D Converter High Result (2-bit)

### ADCDRL (A/D Converter Data Low Register): CCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADDM1<br>ADDL7 | ADDM0<br>ADDL6 | ADDL5 | ADDL4 | ADDL3 | ADDL2 | ADDL1 | ADDL0 |
| R | R | R | R | R- | R | R | R |

Initial value: xxH

ADDM[1:0] MSB align, A/D Converter Low Result (2-bit)
ADDL[7:0] LSB align, A/D Converter Low Result (8-bit)

### ADCCRH (A/D Converter Control High Register): CBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| ADCIFR | – | – | – | TRIG | ALIGN | CKSEL1 | CKSEL0 |
| R/W | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

ADCIFR — When ADC Interrupt occurs, this bit becomes '1'. For clearing bit, write '0' to this bit or auto clear by INT_ACK signal. Writing "1" has no effect.

| | |
|---|---|
| 0 | ADC Interrupt no generation |
| 1 | ADC Interrupt generation |

TRIG — A/D Trigger Signal Selection

| | |
|---|---|
| 0 | ADST |
| 1 | Timer 0 A match signal |

ALIGN — A/D Converter data align selection.

| | |
|---|---|
| 0 | MSB align (ADCDRH[7:0], ADCDRL[7:6]) |
| 1 | LSB align (ADCDRH[1:0], ADCDRL[7:0]) |

CKSEL[1:0] — A/D Converter Clock selection

| CKSEL1 | CKSEL0 | Description |
|---|---|---|
| 0 | 0 | 1MHz |
| 0 | 1 | 0.5MHz |
| 1 | 0 | 0.25MHz |
| 1 | 1 | 0.125MHz |

### ADCCRL (A/D Converter Control Low Register): CAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| STBY | ADST | REFSEL | AFLAG | ADSEL3 | ADSEL2 | ADSEL1 | ADSEL0 |
| R/W | R/W | R/W | R | R/W | R/W | R/W | R/W |

Initial value: 00H

STBY — Control Operation of A/D
(The ADC module is automatically disabled at stop mode)
0 ADC module disable
1 ADC module enable

ADST — Control Trigger Signal for Conversion Start.
0 No effect
1 Trigger signal generation for conversion start

REFSEL — A/D Converter Reference Selection
0 Internal Reference (VDD)
1 External Reference (AVREF)

AFLAG — A/D Converter Operation State (This bit is cleared to '0' when the STBY bit is set to '0' or when the CPU is at STOP mode)
0 During A/D Conversion
1 A/D Conversion finished

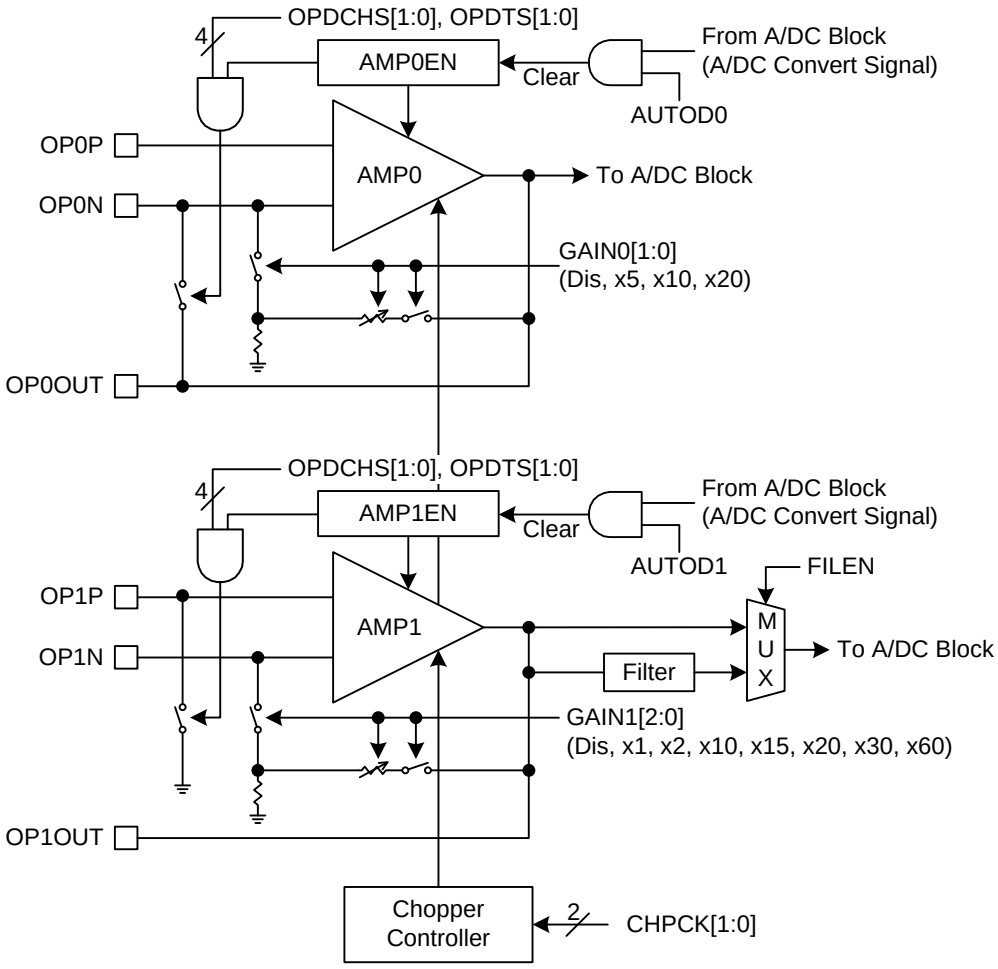
ADSEL[3:0] — A/D Converter input selection

| ADSEL3 | ADSEL2 | ADSEL1 | ADSEL0 | Description |
|---|---|---|---|---|
| 0 | 0 | 0 | 0 | AN0 |
| 0 | 0 | 0 | 1 | AN1 |
| 0 | 0 | 1 | 0 | AN2 |
| 0 | 0 | 1 | 1 | AN3 |
| 0 | 1 | 0 | 0 | AN4 |
| 0 | 1 | 0 | 1 | AN5 |
| 0 | 1 | 1 | 0 | AN6 |
| 0 | 1 | 1 | 1 | AN7 |
| 1 | 0 | 0 | 0 | AN8 |
| 1 | 0 | 0 | 1 | Output of OP-AMP 0 |
| 1 | 0 | 1 | 0 | Output of OP-AMP 1 |
| Other values | | | | Not available |

# 14 Operational amplifier

A96L322 offers two channels of an operational amplifier (OP-Amp). OP-Amp consists of three registers such as OP-AMP control register 0 (AMPCR0), OP-AMP control register 1 (AMPCR1), and Chopper control register (CHPCR).

## 14.1 Block diagram




**NOTE**: Feedback resistors for gain on each Amp are disconnected during one or more Amp and gain are disabled.

**Figure 56. OP Amp Block Diagram**

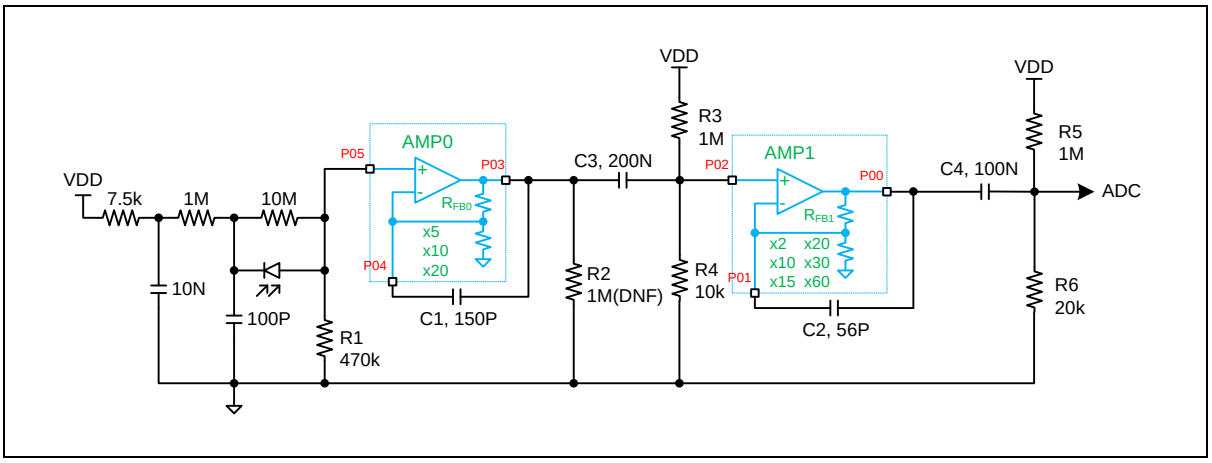



Figure 57. Recommend circuit for internal gain.

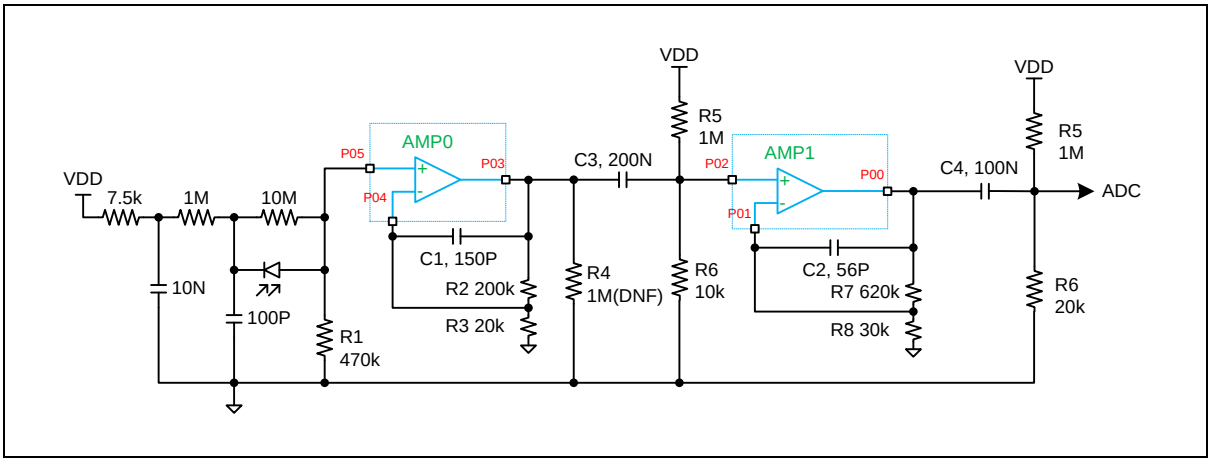



Figure 58. Recommend circuit for external gain.

## 14.2 Register map

Table 19. OP Amp Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| CHPCR | ADH | R/W | 00H | Chopper Control Register |
| AMPCR0 | AEH | R/W | 00H | OP-AMP Control Register 0 |
| AMPCR1 | AFH | R/W | 00H | OP-AMP Control Register 1 |

## 14.3 Register description

**AMPCR0 (Operational Amplifier Control Register 0): AEH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | FILEN | OPDCHS1 | OPDCHS0 | OPDTS1 | OPDTS0 | AUTOD1 | AUTOD0 |
| – | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

FILEN — Filter Control bit

| | |
|---|---|
| 0 | Disable filter and select no filtered OP1OUT for ADC |
| 1 | Enable filter and select the filtered OP1OUT for ADC |

OPDCHS[1:0] — OP-AMP Discharge Channel Selection bits.

| OPDCHS1 | OPDCHS0 | Description |
|---|---|---|
| 0 | 0 | No select |
| 0 | 1 | Select AMP0 |
| 1 | 0 | Select AMP1 |
| 1 | 1 | Select AMP0 and AMP1 |

OPDTS[1:0] — OP-AMP Discharge Time Selection bits. The discharge switch is on for duration time to reduce amp circuit stabilization time when the selected AMP is enabled.

| OPDTS1 | OPDTS0 | Description |
|---|---|---|
| 0 | 0 | Disable discharge |
| 0 | 1 | Enable discharge during 100usec |
| 1 | 0 | Enable discharge during 200usec |
| 1 | 1 | Enable discharge during 300usec |

AUTOD1 — Control disable of OP-AMP1 Block

| | |
|---|---|
| 0 | Not automatically disable |
| 1 | Automatically disable by A/DC convert signal |

AUTOD0 — Control disable of OP-AMP0 Block.

| | |
|---|---|
| 0 | Not automatically disable |
| 1 | Automatically disable by A/DC convert signal |

**NOTE**: The FILEN, OPDCHS[1:0], OPDTS[1:0] bits should always be '0'.

### AMPCR1 (Operational Amplifier Control Register 1): AFH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| AMP1EN | GAIN12 | GAIN11 | GAIN10 | AMP0EN | – | GAIN01 | GAIN00 |
| R/W | R/W | R/W | R/W | R/W | – | R/W | R/W |

Initial value: 00H

AMP1EN Control operation of OP-AMP1 Block, This bit is automatically cleared by A/DC convert signal when the AUTOD1 bit is "1".

| | |
|---|---|
| 0 | OP-AMP1 block disable |
| 1 | OP-AMP1 block enable |

GAIN1[2:0] Select Gain of OP-AMP1

| GAIN12 | GAIN11 | GAIN10 | Description |
|---|---|---|---|
| 0 | 0 | 0 | Disable gain |
| 0 | 0 | 1 | x1 |
| 0 | 1 | 0 | x2 |
| 0 | 1 | 1 | x10 |
| 1 | 0 | 0 | x15 |
| 1 | 0 | 1 | x20 |
| 1 | 1 | 0 | x30 |
| 1 | 1 | 1 | x60 |

AMP0EN Control operation of OP-AMP0 Block, This bit is automatically cleared by A/DC convert signal when the AUTOD0 bit is "1".

| | |
|---|---|
| 0 | OP-AMP0 block disable |
| 1 | OP-AMP0 block enable |

GAIN0[1:0] Select Gain of OP-AMP0

| GAIN01 | GAIN00 | Description |
|---|---|---|
| 0 | 0 | Disable gain |
| 0 | 1 | x5 |
| 1 | 0 | x10 |
| 1 | 1 | x20 |

**NOTE**: The AMP0 and AMP1 must be enabled at same time if the OPDCHS[1:0] bits of AMPCR0 are '11b'.

### CHPCR (Chopper Control Register): ADH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | CHPCK1 | CHPCK0 |
| – | – | – | – | – | – | R/W | R/W |

Initial value: 00H

CHPCK[1:0] Chopper Clock Selection bits

| CHPCK1 | CHPCK0 | Description |
|---|---|---|
| 0 | 0 | 125 kHz |
| 0 | 1 | 167 kHz |
| 1 | 0 | 250 kHz |
| 1 | 1 | 500 kHz |

**NOTE**: The CHPCK[1:0] bits should always be '0'.

# 15 USART

USART (Universal Synchronous/Asynchronous Receiver/Transmitter) is a microchip that facilitates communication through a computer's serial port using RS-232C protocol. A96L322 incorporates a USART function block inside. The USART function block consists of USART control register1/2/3/4, USART status register, USART baud-rate generation register and USART data register.

Operation mode is selected by the operation mode of USART selection bits (USTMS[1:0]).

It has three operating modes as listed in the followings:

- Asynchronous mode (UART)
- Synchronous mode (USART)
- SPI mode

## 15.1 USART UART mode

Universal Asynchronous serial Receiver and Transmitter (UART) is a highly flexible serial communication device. Its main features are listed below:

- Full Duplex Operation (Independent Serial Receive and Transmit Registers)
- Baud Rate Generator
- Supports Serial Frames with 5, 6, 7, 8, or 9 Data Bits and 1 or 2 Stop Bits
- Odd or Even Parity Generation and Parity Check Supported by Hardware
- Data Overrun Detection
- Framing Error Detection
- Three Separate Interrupts on TX Complete, TX Data Register Empty and RX Complete

The UART has a baud rate generator, a transmitter and a receiver. A baud rate generator is used for asynchronous operation. A transmitter consists of a single write buffer, a serial shift register, parity generator and control logic, and is used for handling different serial frame formats. A write buffer allows continuous transfer of data without any delay between frames. A receiver is the most complex part of the UART module because of its clock and data recovery units. A recovery unit is used for asynchronous data reception. In addition to the recovery unit, the receiver includes a parity checker, a shift register, a two-level receive FIFO (USTDR) and control logic. The receiver supports identical frame formats to the transmitter's and can detect frame error, data overrun and parity errors.

## 15.2 UART block diagram

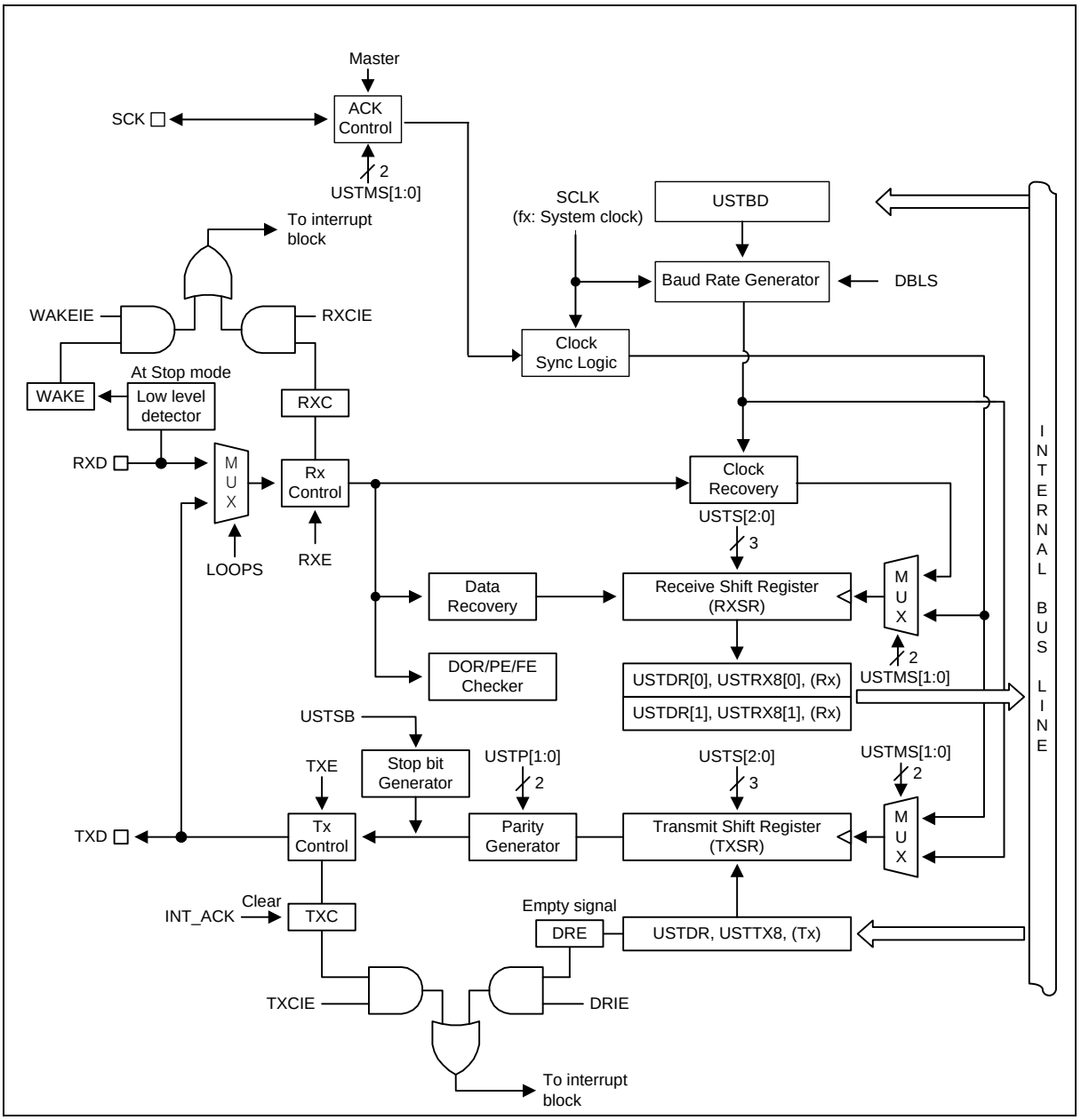



**Figure 59. UART Block Diagram**

## 15.3 Clock generator

A clock generation logic generates a base clock for the transmitter and the receiver. The USART supports four modes of clock operation such as normal asynchronous mode, double speed asynchronous mode, master synchronous mode and slave synchronous mode.

A clock generation scheme for master SPI and slave SPI mode is the same as master synchronous and slave synchronous operation mode. By configuring USTMS[1:0] bits in USTCR1 register, asynchronous operation or synchronous operation can be selected. Asynchronous double speed mode is controlled by the DBLS bit in the USTCR2 register.

MASTER bit in USTCR3 register controls whether the clock source is internal (master mode, output

pin) or external (slave mode, input pin). The SCK pin is active only when the USART operates in synchronous or SPI mode.

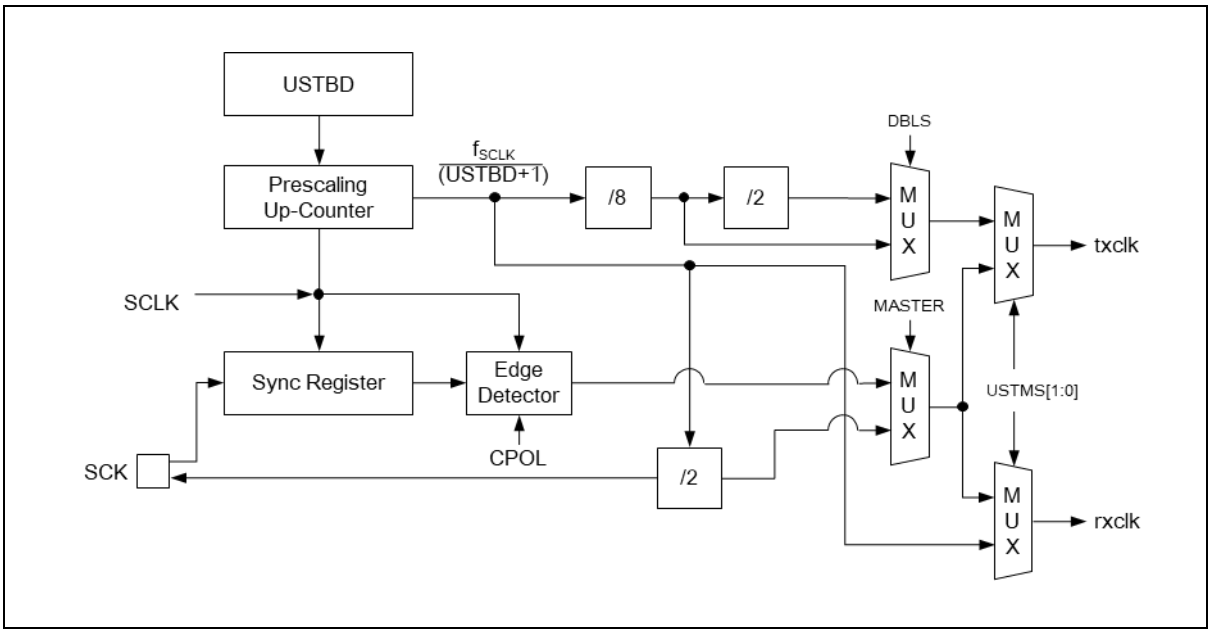



**Figure 60. Clock Generator Block Diagram**

Table 20 introduces equations for calculating baud rate (in bps).

**Table 20. Equations for Baud Rate Register Settings**

| Operating mode | Equation for calculating baud rate |
|---|---|
| Normal Mode(DBLS=0) | Baud Rate= fx/(16(USTBD+1)) |
| Double Speed Mode(DBLS=1) | Baud Rate= fx/(8(USTBD+1)) |
| Synchronous or SPI Master Mode | Baud Rate= fx/(2(USI0BD+1)) |

## 15.4 External clock (SCK)

External clocking is used in the synchronous mode of operation.

External clock input from the SCK pin is sampled by a synchronization logic to remove meta-stability. The output from the synchronization logic must be passed through an edge detector before it is used by the transmitter and the receiver. This process brings two CPU clock period delays. The maximum frequency of the external SCK pin is limited up to 1MHz.

## 15.5 Synchronous mode operation

When synchronous or SPI mode is used, the SCK pin will be used as either a clock input (slave) or a clock output (master). Data is sampled and transmitter is issued on different edges of SCK clock respectively.

For example, if data input on RXD (MISO in SPI mode) pin is sampled on the rising edge of SCK clock, data output on TXD (MOSI in SPI mode) pin is altered on the falling edge.

By configuring CPOL bit in USTCR1 register, an edge of SCK clock for data sampling and for data change can be selected. As shown in the figure 59 below, when CPOL is zero, the data will be changed at rising SCK edge and sampled at falling SCK0 edge.

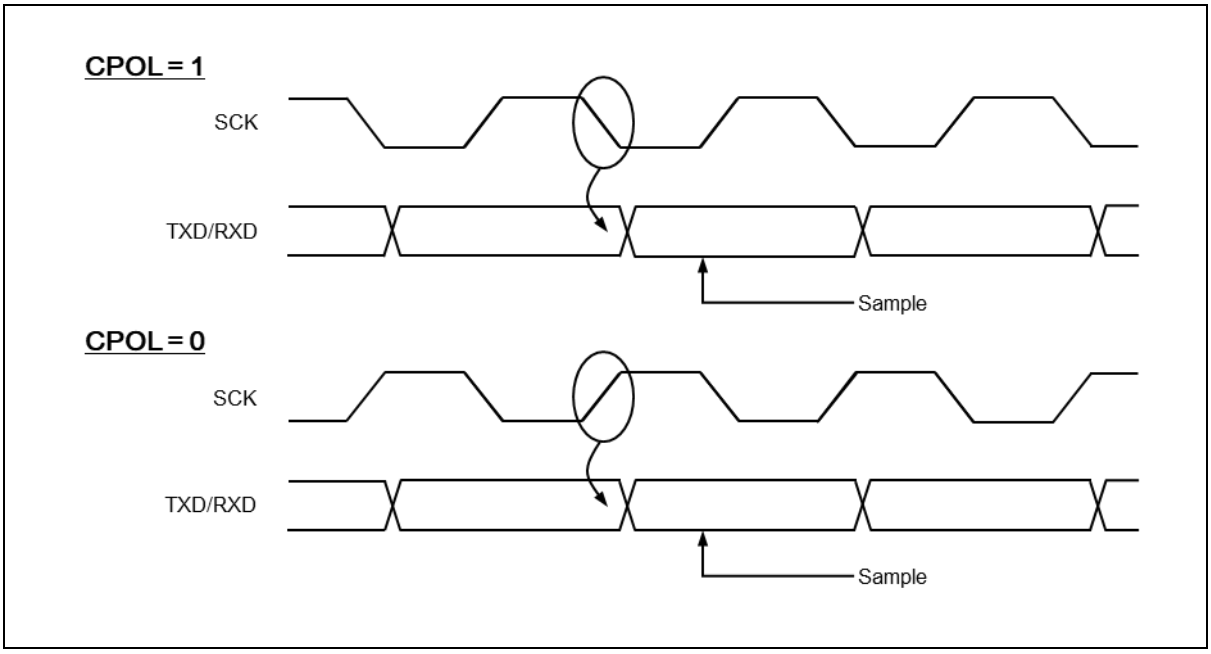



Figure 61. Synchronous Mode SCK Timing (USART)

## 15.6 Data format

A serial frame is defined as a single character of data bits that consist of synchronization bits (start and stop bits), and optionally a parity bit for error detection.

The UART supports 30 combinations of the followings as valid frame formats.

- 1 start bit
- 5, 6, 7, 8 or 9 data bits
- no, even or odd parity bit
- 1 or 2 stop bits

A frame starts with a start bit, followed by the least significant data bit (LSB). Then the next data bits, up to nine, are succeeding, ending with the most significant bit (MSB). If parity function is enabled, the parity bit is inserted between the last data bit and the stop bit. A high-to-low signal transition on a data pin is considered as start bit. When a complete frame is transmitted, it can be directly followed by a new frame, or the communication line enters to an idle state. The idle means high state of data pin.

The following figure 62 shows possible combinations of the frame format. Bits inside brackets are optional.

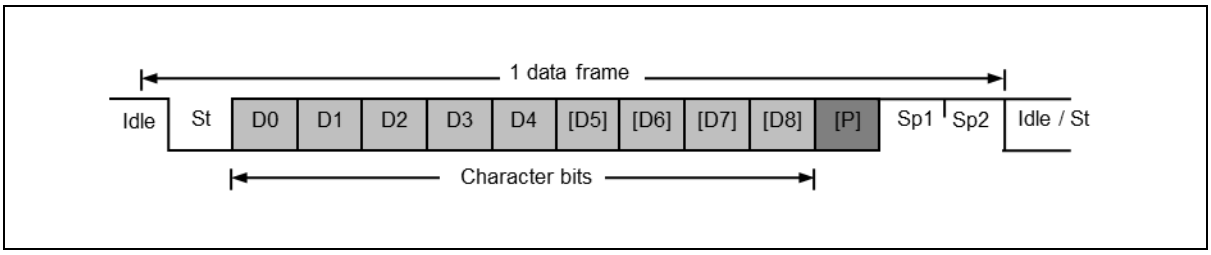



**Figure 62. Frame Format Diagram**

Every single frame will have the following bits:

- Idle: no communication on communication line (TxD/RxD)
- St: starting the frame (Start bit: Low)
- Dn: data bits (0 to 8)
- Parity bit: even parity, odd parity, no parity
- Stop bit(s): end of the frame (1 bit or 2 bits)

Frame format of the UART is defined by configuring USTS [2:0], USTP [1:0] and USTSB in registers USTCR1 and USTCR3. The transmitter and the receiver use the same settings.

## 15.7 Parity bit

The parity bit is calculated by doing XOR of all data bits. If odd parity is used, result of the XOR is inverted. The parity bit is located between the MSB and the first stop bit of a serial frame.

$P_{even} = D_{n-1} \wedge \ldots \wedge D_3 \wedge D_2 \wedge D_1 \wedge D_0 \wedge 0$

$P_{odd} = D_{n-1} \wedge \ldots \wedge D_3 \wedge D_2 \wedge D_1 \wedge D_0 \wedge 1$

$P_{even}$: Parity bit using even parity

$P_{odd}$: Parity bit using odd parity

$D_n$: Data bit n of the character

## 15.8 UART transmitter

UART transmitter is enabled by setting TXE bit in USTCR2 register. When the Transmitter is enabled, TXD pin must be set to TXD function for the serial output pin of UART. This can be done by configuring P0FSRL[1:0]. Baud-rate, operation mode and frame format must be set once before starting any transmission.

### 15.8.1 Sending Tx data

Data transmission is initiated by loading a transmit buffer (USTDR register I/O location) with data to be transmitted.

The data written in transmit buffer is moved to the shift register when the shift register is ready to send a new frame. The shift register is loaded with the new data if it is in idle state or immediately after the last stop bit of the previous frame is transmitted. When the shift register is loaded with new data, it will transfer one complete frame according to the settings of control registers. If the 9-bit characters are used, the ninth bit must be written to the TX8 bit in USTCR3 register before it is loaded to the transmit buffer (USTDR register).

### 15.8.2 Transmitter flag and interrupt

The UART transmitter has 2 flags indicating its status. One is a UART data register empty flag (UDRE) and the other is transmit complete flag (TXC). Both flags can be interrupt sources. UDRE flag indicates whether the transmit buffer is ready to receive new data. This bit is set when the transmit buffer is empty, and cleared when the transmit buffer contains data to be transmitted but has not yet been moved into the shift register. This flag can be cleared by writing '0' to this bit position too. Writing '1' to this bit position is prevented.

When the data register empty interrupt enable (UDRIE) bit in USTCR2 register is set and the global interrupt is enabled, UART data register empty interrupt is generated while UDRE flag is set.

The transmit complete (TXC) flag bit is set when the entire frame in the transmit shift register has been shifted out and there is no more data in the transmit buffer. The TXC flag is automatically cleared when the transmit complete interrupt service routine is executed, or it can be cleared by writing '0' to TXC bit in USTST register.

When the transmit complete interrupt enable (TXCIE) bit in USTCR2 register is set and the global interrupt is enabled, UART transmit complete interrupt is generated while TXC flag is set.

### 15.8.3 Parity generator

A parity generator calculates a parity bit for the serial frame data to be sent. When parity bit is enabled (USTP[1] = 1), the transmitter control logic inserts the parity bit between the MSB and the first stop bit of the frame to be sent.

### 15.8.4 Disabling transmitter

Disabling the transmitter by clearing TXE bit will not be effective until ongoing transmission is completed. When the Transmitter is disabled, the TXD pin can be used as a normal general purpose I/O (GPIO).

## 15.9 UART receiver

UART receiver is enabled by setting RXE bit in USTCR2 register. When the receiver is enabled, the RXD pin must be set to RXD function for the serial input pin of UART. This can be done by configuring by P0FSRL[3:2]. Baud-rate, mode of operation and frame format must be set before starting serial reception.

### 15.9.1 Receiving Rx data

The receiver starts data reception when it detects a valid start bit (LOW) on RXD pin. Each bit after start bit is sampled at pre-defined baud-rate (asynchronous) and shifted into the receive shift register until the first stop bit of a frame is received. Even if there's 2nd stop bit in the frame, the 2nd stop bit is ignored by the receiver. That is, receiving the first stop bit means that a complete serial frame is present in the receiver shift register and contents of the shift register are to be moved into the receive buffer. The receive buffer is read by reading the USTDR register.

If 9-bit characters are used (USTS[2:0] = "111"), the ninth bit is stored in the RX8 bit position in the USTCR3 register. The 9th bit must be read from the RX8 bit before reading the low 8 bits from the USTDR register. Likewise, the error flags FE, DOR, PE must be read before reading the data from USTDR register. It's because the error flags are stored in the same FIFO position of the receive buffer.

### 15.9.2 Receiver flag and interrupt

The UART receiver has one flag that indicates the receiver's status. Receive complete (RXC) flag indicates whether there are unread data in the receive buffer. This flag is set when there are unread data in the receive buffer, and cleared when the receive buffer is empty. If the receiver is disabled (RXE=0), the receiver buffer is flushed and the RXC flag is cleared.

When receive complete interrupt enable (RXCIE) bit in register USTCR2 is set and global interrupt is enabled, the UART receiver complete interrupt is generated while RXC flag is set.

The UART receiver has three error flags which are frame error (FE), data overrun (DOR) and parity error (PE). These error flags can be read from the USTST register. As received data are stored in the 2-level receive buffer, these error flags are also stored in the same position of receive buffer. So, before reading received data from USTDR register, read the USTST register first which contains error flags.

The frame error (FE) flag indicates the state of the first stop bit. The FE flag is '0' when the stop bit was correctly detected as '1', and the FE flag is '1' when the stop bit was incorrect, i.e. detected as '0'. This flag can be used for detecting out-of-sync conditions between data frames.

The data overrun (DOR) flag indicates data loss due to a receive buffer full condition. DOR occurs when the receive buffer is full, and another new data is present in the receive shift register which are to be stored into the receive buffer. After the DOR flag is set, all the incoming data are lost. To prevent data loss or clear this flag, read the receive buffer.

The parity error (PE) flag indicates that the frame in the receive buffer had a parity error when received. If parity check function is not enabled (USTP[1] = 0), the PE bit is always read '0'.

### 15.9.3 Parity checker

Parity checker calculates the parity of the data bits in incoming frame and compares the result with the parity bit from the received serial frame, if parity bit is enabled (USTP[1]=1).

### 15.9.4 Disabling receiver

Unlike the transmitter, disabling the Receiver by clearing RXE bit makes the receiver inactive immediately. When the receiver is disabled, the receiver flushes the receive buffer, the remaining data in the buffer is all reset, and the RXD pin can be used as a normal general purpose I/O (GPIO).

### 15.9.5 Asynchronous data reception

To receive asynchronous data frame, the UART has a clock and data recovery unit. The clock recovery logic is used for synchronizing the internally generated baud-rate clock to the incoming asynchronous serial frame on the RXD pin. The data recovery logic samples and low pass filters the incoming bits, to remove the noise of RXD pin.

Figure 63 illustrates the sampling process of the start bit of an incoming frame. The sampling rate is 16 times the baud-rate for normal mode (DBLS=0) and 8 times the baud-rate for double speed mode (DBLS=1). The horizontal arrows show the synchronization variation due to the asynchronous sampling process. Note that larger time variation is shown when using the double speed mode.

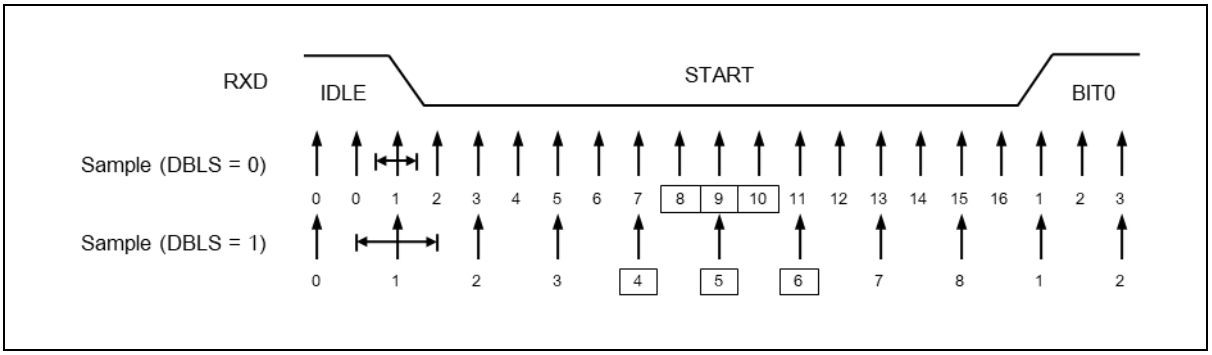



Figure 63. Start Bit Sampling

When the receiver is enabled (RXE=1), the clock recovery logic tries to find a high-to-low transition on the RXD line, the start bit condition.

After detecting high to low transition on RXD line, the clock recovery logic uses samples 8, 9, and 10 for normal mode, and samples 4, 5, and 6 for double speed mode to decide if a valid start bit is received. If more than 2 samples have logical low level, it is considered that a valid start bit is detected and the internally generated clock is synchronized to the incoming data frame. And the data recovery can begin. The synchronization process is repeated for each start bit.

As described in figure 63, when the receiver clock is synchronized to the start bit, the data recovery can begin. Data recovery process is almost similar to the clock recovery process. The data recovery logic samples 16 times for each incoming bits for normal mode and 8 times for double speed mode. And uses sample 8, 9, and 10 to decide data value for normal mode, and samples 4, 5, and 6 for double speed mode. If more than 2 samples have low levels, the received bit is considered to a logic '0' and if more than 2 samples have high levels, the received bit is considered to a logic '1'. The data recovery process is then repeated until a complete frame is received including the first stop bit. The decided bit value is stored in the receive shift register in order.

Note that the Receiver only uses the first stop bit of a frame. Internally, after receiving the first stop bit, the Receiver is in idle state and waiting to find start bit.

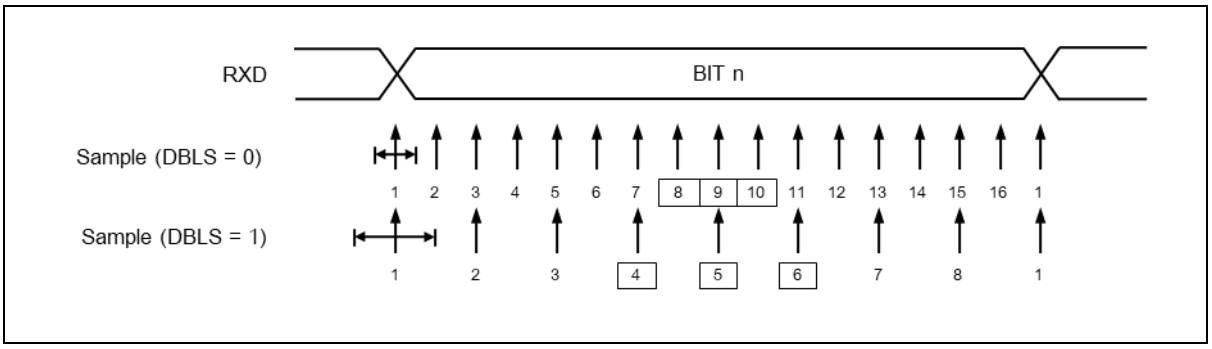



**Figure 64. Data and Parity Bit Sampling**

The process for detecting stop bit is like clock and data recovery process. That is, if 2 or more samples of 3 center values have high level, correct stop bit is detected, else a frame error (FE) flag is set. After deciding whether the first stop bit is valid or not, the Receiver goes to idle state and monitors the RXD line to check a valid high to low transition is detected (start bit detection).

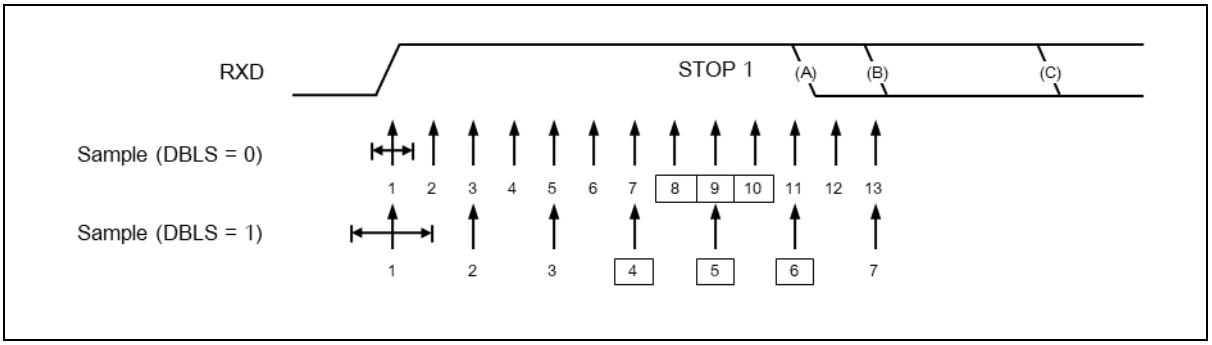



**Figure 65. Stop Bit Sampling and Next Stop Bit Sampling**

## 15.10 USART SPI mode

USART can be configured to operate in industrial standard SPI compliant mode. The SPI mode has the following features.

- Full Duplex, Three-wire synchronous data transfer
- Mater and Slave Operation
- Supports all four SPI modes of operation (mode 0, 1, 2, and 3)
- Selectable LSB first or MSB first data transfer
- Double buffered transmit and receive
- Programmable transmit bit rate

When SPI mode is enabled (USTMS[1:0]="11"), the slave select (SSn) pin becomes active LOW input in slave mode operation, or can be output in master mode operation if USTSSEN bit is set to '0'.

Note that during SPI mode of operation, the pin RXD is renamed as MISO and TXD is renamed as MOSI for compatibility to other SPI devices.

## 15.11 SPI block diagram

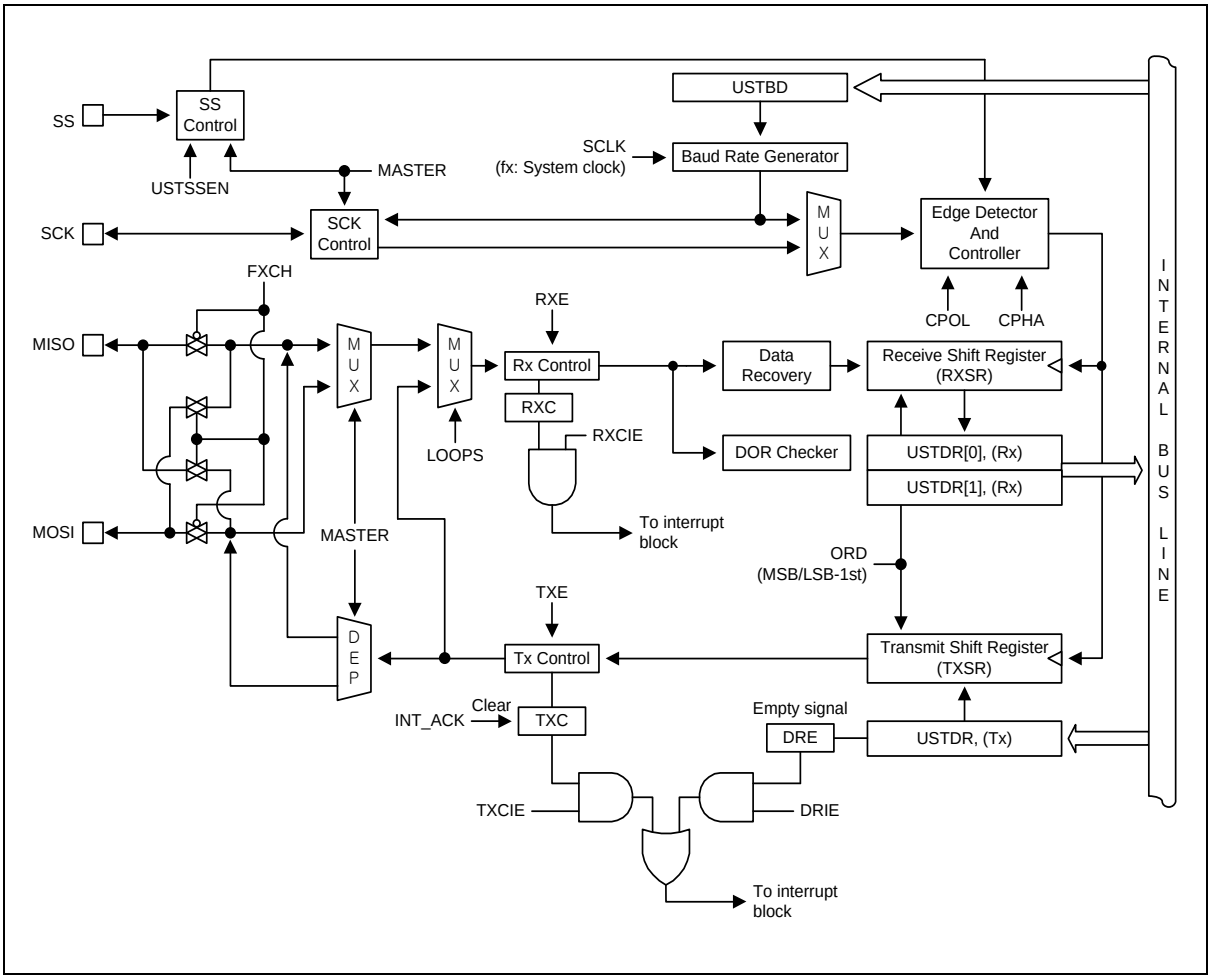



**Figure 66. SPI Block Diagram**

## 15.12 SPI clock formats and timing

To accommodate a wide variety if synchronous serial peripherals from different manufacturers, the USART has a clock polarity bit (CPOL) and a clock phase control bit (CPHA) to select one of four clock formats for data transfers. CPOL selectively insert an inverter in series with the clock. CPHA chooses between two different clock phase relationships between the clock and data. Note that CPHA and CPOL bits in USTCR1 register have different meanings according to the USTMS[1:0] bits which decides the operating mode of USART.

Table 21 introduces four combinations of CPOL and CPHA for SPI mode 0, 1, 2, and 3.

Table 21. CPOL Functionality

| SPI Mode | CPOL | CPHA | Leading Edge | Trailing Edge |
|---|---|---|---|---|
| 0 | 0 | 0 | Sample (Rising) | Setup (Falling) |
| 1 | 0 | 1 | Setup (Rising) | Sample (Falling) |
| 2 | 1 | 0 | Sample (Falling) | Setup (Rising) |
| 3 | 1 | 1 | Setup (Falling) | Sample (Rising) |

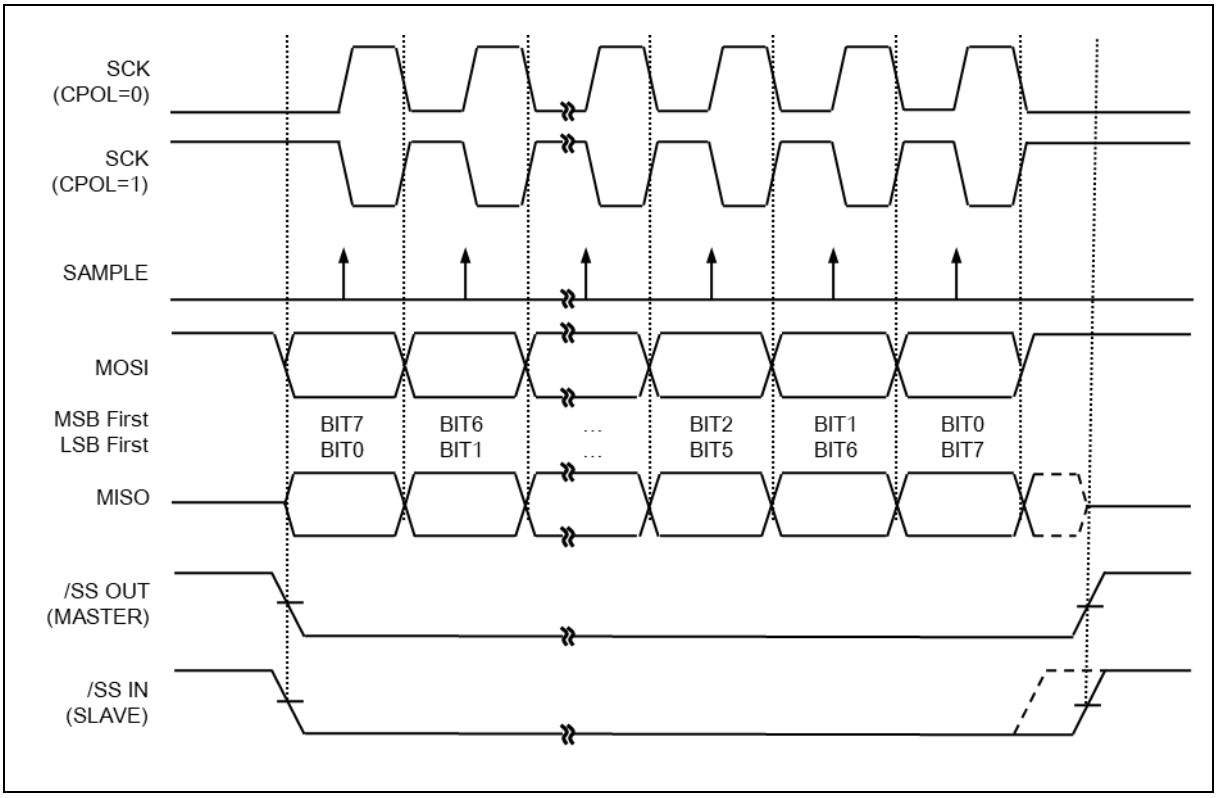



Figure 67. SPI Clock Formats when CPHA=0

When CPHA=0, the slave begins to drive its MISO output with the first data bit value when SS goes to active low. The first SCK edge causes both the master and the slave to sample the data bit value on their MISO and MOSI inputs, respectively. At the second SCK edge, the USART shifts the second data bit value out to the MOSI and MISO outputs of the master and slave, respectively. Unlike the case of CPHA=1, when CPHA=0, the slave's SS input must go to its inactive high level between transfers. This is because the slave can prepare the first data bit when it detects falling edge of SS input.

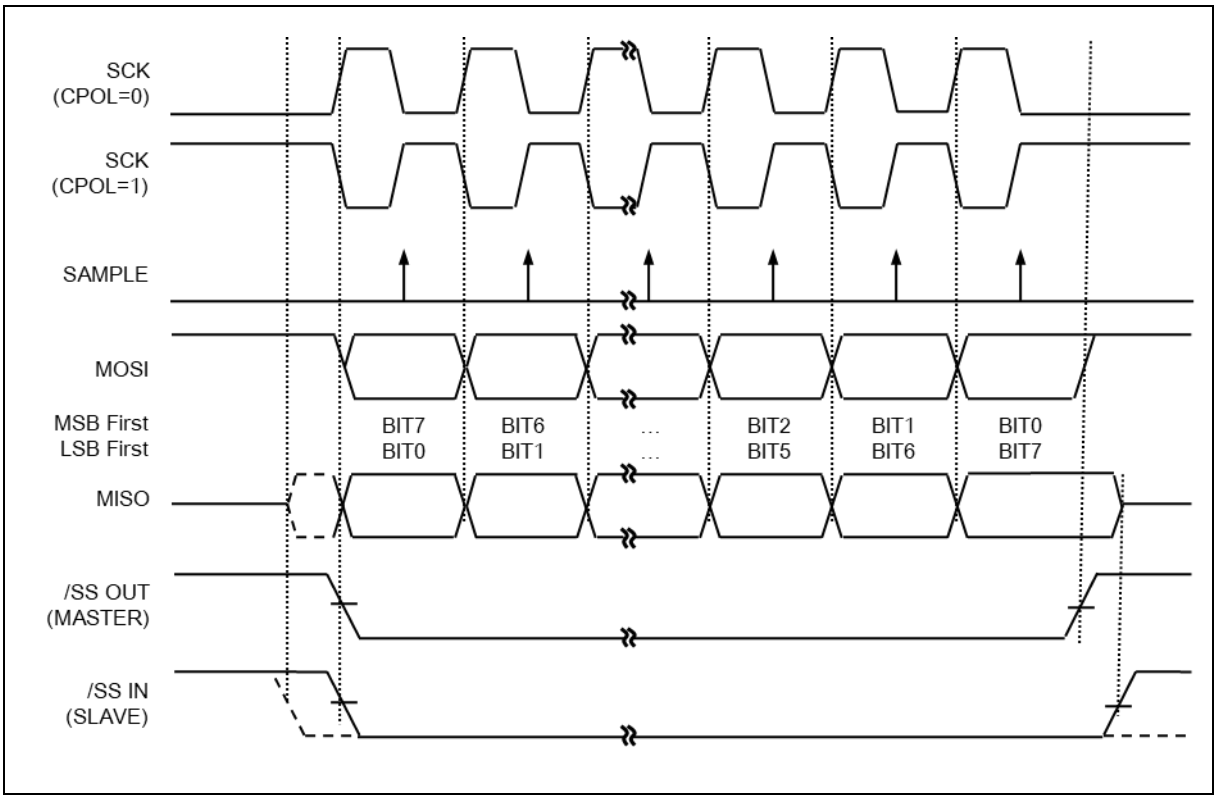



Figure 68. SPI Clock Formats when CPHA=1

When CPHA=1, the slave begins to drive its MISO output when SS goes active low, but the data is not defined until the first SCK edge. The first SCK edge shifts the first bit of data from the shifter onto the MOSI output of the master and the MISO output of the slave. The next SCK edge causes both the master and slave to sample the data bit value on their MISO and MOSI inputs, respectively. At the third SCK edge, the USART shifts the second data bit value out to the MOSI and MISO output of the master and slave respectively. When CPHA=1, the slave's SS input is not required to go to its inactive high level between transfers.

Because the SPI logic reuses the USART resources, SPI mode of operation is similar to that of synchronous or asynchronous operation. An SPI transfer is initiated by checking for the USART Data Register Empty flag (DRE=1) and then writing a byte of data to the USTDR Register. In master mode of operation, even if transmission is not enabled (TXE=0), writing data to the USTDR register is necessary because the clock SCK is generated from transmitter block.

## 15.13 Register map

Table 22. USART Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| USTCR1 | 1000H (XSFR) | R/W | 00H | USART Control Register 1 |
| USTCR2 | 1001H (XSFR) | R/W | 00H | USART Control Register 2 |
| USTCR3 | 1002H (XSFR) | R/W | 00H | USART Control Register 3 |
| USTST | 1003H (XSFR) | R/W | 80H | USART Status Register |
| USTBD | 1004H (XSFR) | R/W | FFH | USART Baud Rate Generation Register |
| USTDR | 1005H (XSFR) | R/W | 00H | USART Data Register |

## 15.14 Register description

### USTBD (USART Baud-Rate Generation Register): 1004H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| USTBD7 | USTBD6 | USTBD5 | USTBD4 | USTBD3 | USTBD2 | USTBD1 | USTBD0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

USTBD[7:0] The value in this register is used to generate internal baud rate in UART mode or to generate SCK clock in SPI mode. To prevent malfunction, do not write '0' in UART mode and do not write '0' or '1' in synchronous or SPI mode.

### USTDR (USART Data Register): 1005H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| USTDR7 | USTDR6 | USTDR5 | USTDR4 | USTDR3 | USTDR2 | USTDR1 | USTDR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

USTDR[7:0] The USART Transmit buffer and Receive buffer share the same I/O address with this DATA register. The Transmit Data Buffer is the destination for data written to the USTDR register. Reading the USTDR register returns the contents of the Receive Buffer. Write to this register only when the DRE flag is set. In SPI master mode, the SCK clock is generated when data are written to this register.

### USTCR1 (USART Control Register 1): 1000H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| USTMS1 | USTMS0 | USTP1 | USTP0 | USTS2 | USTS1<br>ORD | USTS0<br>CPHA | CPOL |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

USTMS[1:0] Selects Operation Mode of USART

| USTMS1 | USTMS0 | Operation mode |
|---|---|---|
| 0 | 0 | Asynchronous Mode (UART) |
| 0 | 1 | Synchronous Mode |
| 1 | 0 | Reserved |
| 1 | 1 | SPI mode |

USTP[1:0] Selects Parity Generation and Check method (only UART mode)

| USTPM1 | USTPM0 | Parity |
|---|---|---|
| 0 | 0 | No Parity |
| 0 | 1 | Reserved |
| 1 | 0 | Even Parity |
| 1 | 1 | Odd Parity |

USTS[2:0] When in Asynchronous or Synchronous mode of operation, selects the length of data bits in a frame.

| USTS2 | USTS1 | USTS0 | Data Length |
|---|---|---|---|
| 0 | 0 | 0 | 5 bit |
| 0 | 0 | 1 | 6 bit |
| 0 | 1 | 0 | 7 bit |
| 0 | 1 | 1 | 8 bit |
| 1 | 1 | 1 | 9 bit |
| Other values | | | Reserved |

ORD This bit is in the same bit position with USTS1. The MSB of the data byte is transmitted first when set to '1' and the LSB when set to '0' (only SPI mode)

| | |
|---|---|
| 0 | LSB-first |
| 1 | MSB-first |

CPOL This bit determines the clock polarity of ACK in synchronous or SPI mode

| | |
|---|---|
| 0 | TXD Change @Rising Edge, RXD Change @Falling Edge |
| 1 | TXD Change @Falling Edge, RXD Change @Rising Edge |

CPHA This bit is in the same bit position with USTS0. This bit determines if data are sampled on the leading or trailing edge of SCK (only SPI mode)

| CPOL | CPHA | Leading edge | Trailing edge |
|---|---|---|---|
| 0 | 0 | Sample (Rising) | Setup (Falling) |
| 0 | 1 | Setup (Rising) | Sample (Falling) |
| 1 | 0 | Sample (Falling) | Setup (Rising) |
| 1 | 1 | Setup (Falling) | Sample (Rising) |

### USTCR2 (USART Control Register 2): 1001H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DRIE | TXCIE | RXCIE | WAKEIE | TXE | RXE | USTEN | DBLS |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | | |
|---|---|---|
| DRIE | Interrupt enable bit for Data Register | |
| | 0 | Interrupt from DRE is inhibited (use polling) |
| | 1 | When DRE is set, request an interrupt |
| TXCIE | Interrupt enable bit for Transmit Complete | |
| | 0 | Interrupt from TXC is inhibited (use polling) |
| | 1 | When TXC is set, request an interrupt |
| RXCIE | Interrupt enable bit for Receive Complete | |
| | 0 | Interrupt from RXC is inhibited (use polling) |
| | 1 | When RXC is set, request an interrupt |
| WAKEIE | Interrupt enable bit for Asynchronous Wake in STOP mode. When device is in stop mode, if RXD goes to Low level, an interrupt can be requested to wake-up system (only UART mode) | |
| | 0 | Interrupt from Wake is inhibited |
| | 1 | When WAKE is set, request an interrupt |
| TXE | Enables the Transmitter unit | |
| | 0 | Transmitter is disabled |
| | 1 | Transmitter is enabled |
| RXE | Enables the Receiver unit | |
| | 0 | Receiver is disabled |
| | 1 | Receiver is enabled |
| USTEN | Activate USART Function Block by supplying. | |
| | 0 | USART is disabled |
| | 1 | USART is enabled |
| DBLS | This bit selects receiver sampling rate (only UART mode) | |
| | 0 | Normal asynchronous operation |
| | 1 | Double speed asynchronous operation |

### USTCR3 (USART Control Register 3): 1002H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| MASTER | LOOPS | DISSCK | USTSSEN | FXCH | USTSB | USTTX8 | USTRX8 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | | |
|---|---|---|
| MASTER | Selects master or slave in SPI or Synchronous mode operation and controls the direction of SCK pin. | |
| | 0 | Slave operation (External clock for SCK) |
| | 1 | Master operation (Internal clock for SCK) |
| LOOPS | Control the Loop Back mode of USART for test mode | |
| | 0 | Normal operation |
| | 1 | Loop Back mode |
| DISSCK | In synchronous mode operation, selects the waveform of SCK output. | |
| | 0 | SCK is free-running while UART is enabled in synchronous master mode |
| | 1 | SCK is active while any frame is on transferring |
| USTSSEN | This bit controls the SS pin operation (only SPI mode) | |
| | 0 | Disable |
| | 1 | Enable (The SS pin should be a normal input) |
| FXCH | SPI port function exchange control bit (only SPI mode) | |
| | 0 | No effect |
| | 1 | Exchange MOSI and MISO function |
| USTSB | Selects the length of stop bit in Asynchronous or Synchronous mode of operation. | |
| | 0 | 1 Stop bit |
| | 1 | 2 Stop bit |
| USTTX8 | The ninth bit of data frame in Asynchronous or Synchronous mode of operation. Write this bit first before loading the USTDR register. | |
| | 0 | MSB (9th bit) to be transmitter is '0' |
| | 1 | MSB (9th bit) to be transmitter is '1' |
| USTRX8 | The ninth bit of data frame in Asynchronous or Synchronous mode of operation. Read this bit first before reading the receive buffer (only UART mode) | |
| | 0 | MSB (9th bit) to be received is '0' |
| | 1 | MSB (9th bit) to be received is '1' |

### USTST (USART Status Register): 1003H (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| DRE | TXC | RXC | WAKE | USTRST | DOR | FE | PE |
| R/W | R/W | R | R/W | R/W | R | R/W | R/W |

Initial value: 80H

DRE — The DRE flag indicates if the transmit buffer (USTDR) is ready to receive new data. If DRE is '1', the buffer is empty and ready to be written. The flag can generate a DRE interrupt.

- 0 Transmit buffer is not empty
- 1 Transmit buffer is empty

TXC — This flag is set when the entire frame in the transmit shift register has been shifted out and there is no new data currently present in the transmit buffer. This flag is automatically cleared when the interrupt service routine of a TXC interrupt is executed. This flag can generate a TXC interrupt.

- 0 Transmission is ongoing
- 1 Transmit buffer is empty and the data in transmit shift register are shifted out completely

RXC — This flag is set when there are unread data in the receive buffer and cleared when all the data in the receive buffer are read. The RXC flag can be used to generate a RXC interrupt.

- 0 There is no data unread in the receive buffer
- 1 There are more than 1 data in the receive buffer

WAKE — This flag is set when the RXD pin is detected low while the CPU is in STOP mode. This flag can be used to generate a WAKE interrupt (only UART mode)

- 0 No WAKE interrupt is generated
- 1 WAKE interrupt is generated

USTRST — This is an internal reset and only has effect on USART. Writing '1' to this bit initializes the internal logic of USART and is automatically cleared to '0'.

- 0 No effect
- 1 Reset USART

DOR — This bit is set if data OverRun occurs. While this bit is set, the incoming data frame is ignored. This flag is valid until the receive buffer is read.

- 0 No Data OverRun
- 1 Data OverRun detected

FE — This bit is set if the first stop bit of next character in the receive buffer is detected as '0'. This bit is valid until the receive buffer is read (only UART mode)

- 0 No Frame Error
- 1 Frame Error detected

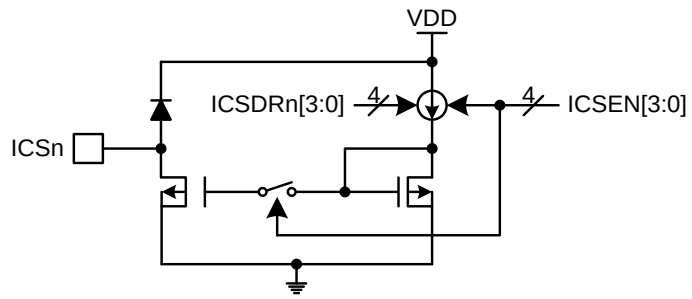
PE — This bit is set if the next character in the receive buffer has a Parity Error while Parity Checking is enabled. This bit is valid until the receive buffer is read (only UART mode)

- 0 No Parity Error
- 1 Parity Error detected

# 16 Constant sink current generator

Constant sink current generator supplies constant current regardless of variable $I_{CS}$ voltage ranging from 1.8V to 3.6V. The constant current value is controlled by registers ICSDR0 and ICSDR1, and the sink current ranges from 49mA to 274mA.

## 16.1 Block diagram




**NOTE**: If a sink current generator of an ICSn pin is disabled by ICSCR[3:0], the corresponding ICSn pin is high and the current flowing is zero.

**Figure 69. Constant Sink Current Generator Block Diagram (n=0 and 1)**

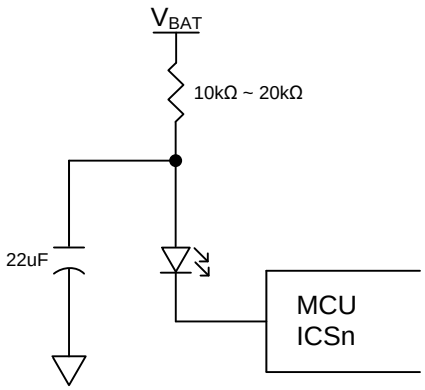



**Figure 70. Constant Sink Current Generator Pin with Capacitor**

## 16.2 Register map

**Table 23. Constant Sink Current Generator Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| ICSCR | E5H | R/W | 00H | Constant Sink Current Control Register |
| ICSDR0 | E6H | R/W | 00H | Constant Sink Current Data Register 0 |
| ICSDR1 | E7H | R/W | 00H | Constant Sink Current Data Register 1 |

## 16.3 Register description

### ICSCR (Constant Sink Current Control Register): E5H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | ICSEN3 | ICSEN2 | ICSEN1 | ICSEN0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

| | | |
|---|---|---|
| ICSEN[3:0] | Constant Sink Current Enable bits | |
| | 0101b | Enable for the ICS0 pin and disable for the ICS1 pin |
| | 1010b | Enable for the ICS1 pin and disable for the ICS0 pin |
| | Others | Disable sink current generator for the ICS0 and ICS1 pins |

### ICSDR0 (Constant Sink Current Data Register 0): E6H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | ICSD03 | ICSD02 | ICSD01 | ICSD00 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| ICSD0[3:0] | ICS0 pin Constant Sink Current Data bits<br>ICS0 pin current [mA] = 50 + 15 x ICSD0[3:0] |

### ICSDR1 (Constant Sink Current Data Register 1): E7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | ICSD13 | ICSD12 | ICSD11 | ICSD10 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| ICSD1[3:0] | ICS1 pin Constant Sink Current Data bits<br>ICS1 pin current [mA] = 50 + 15 x ICSD1[3:0] |

# 17 Flash CRC and Checksum generator

Flash CRC (Cyclic Redundancy Check) generator of A96L322 generates 16-bit CRC code bits from flash and a generator polynomial. The CRC code for each input data frame is appended to the frame.

Specifically CRC-based technique is used to verify data transmission or storage integrity. In the scope of the functional safety standards, this technique offers a means of verifying the Flash memory integrity. The flash CRC generator helps compute a signature of software during runtime, to be compared with a reference signature.

The CRC generator has following features:

- Auto CRC and User CRC Mode
- CRC Clock : fIRC, fIRC/2, fIRC/4, fIRC/8 and fx (System clock)
- CRC-16 polynomial: 0x8C81 ($X^{16} + X^{15} + X^{11} + X^{10} + X^{7} + 1$)

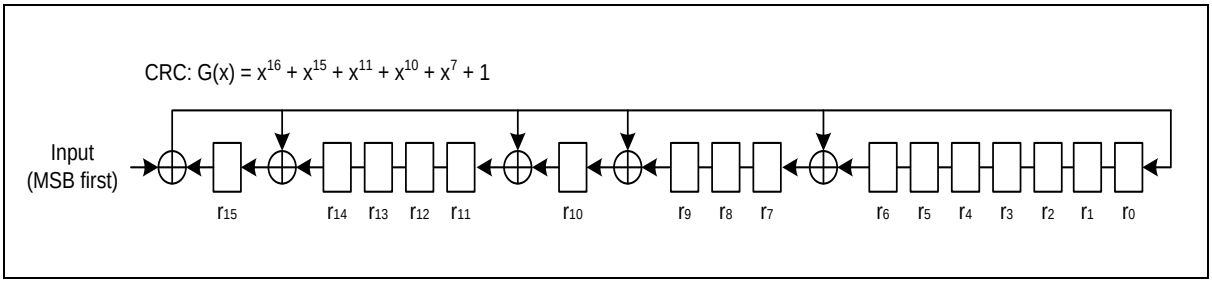



Figure 71. CRC-16 Polynomial Structure

## 17.1 Block diagram

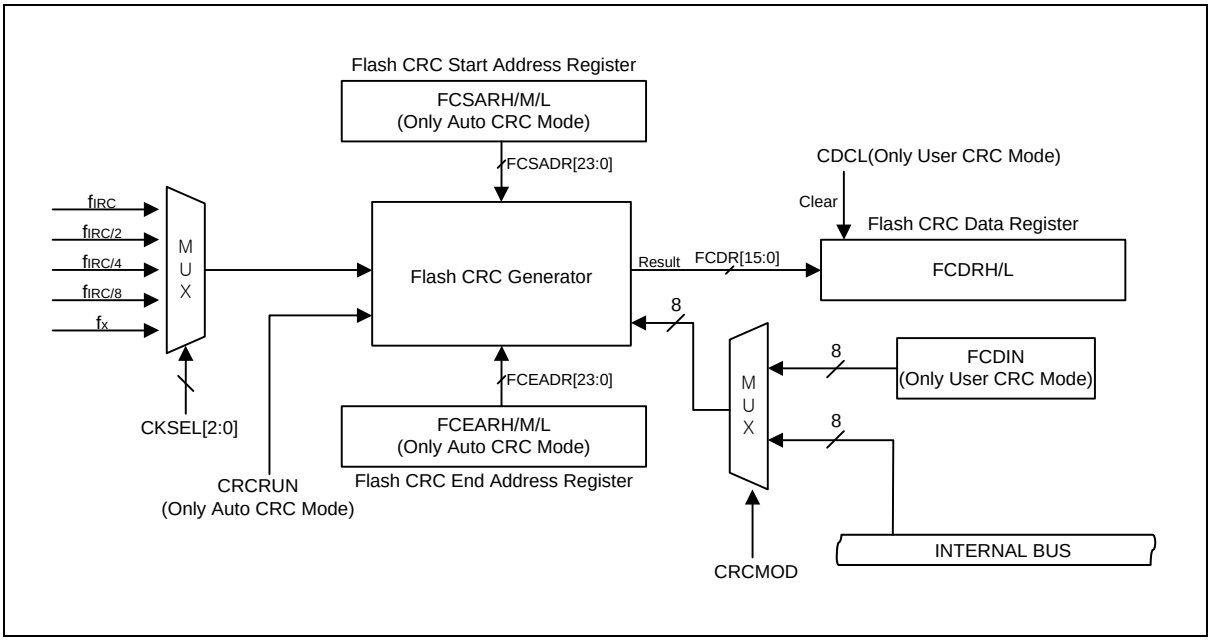



**Figure 72. Flash CRC/Checksum Generator Block Diagram**

## 17.2 Operation procedure and example code of CRC and Checksum

The CRC operation procedure in Auto CRC/Checksum mode is introduced in the following list, and figure 73 shows example program tip:

1. Global interrupt Disable (EA = 0)
2. Select Auto CRC/Checksum Mode and CRC
3. Select CRC Clock
4. Set CRC start address register (FCSARH/FCSARM/FCSARL)
5. Set CRC end address register (FCEARH/FCEARM/FCEARL)
6. CRC operation starts (CRCRUN = 1)
7. Read the CRC result
8. Global interrupt Enable (EA = 1)

```
//**** Global interrupt Disable
       EA = 0;

//**** Flash CRC Auto CRC/Checksum Mode and CRC
       FCCR &= _0101_1111;

       OSCCR &= _1111_1011;       // IRC Enable
       FCCR &= _1111_0001;        // CRC clk = fIRC/1

//**** CRC start address set
       FCSARH = 0x00;
       FCSARM = 0x00;
       FCSARL = 0x00;

//**** CRC end address set
       FCEARH = 0x00;
       FCEARM = 0x3F;
       FCEARL = 0xFF;

//**** CRC start
       FCCR |= _0000_0001;
     _nop_();              //Dummy instruction, This instruction must be
needed.
     _nop_();              //Dummy instruction, This instruction must be
needed.
     _nop_();              //Dummy instruction, This instruction must be
needed.

//**** Read CRC result
     Temp0 = FCDRH;
     Temp1 = FCDRL;

//**** Global interrupt Enable
       EA = 1;
```

**NOTES**:

1. Three or more NOP instructions must immediately follow the CRC start operation in auto CRC/Checksum mode.
2. During a CRC operation (when CRCRUN bit is Running state) in auto CRC/Checksum mode, the CPU is hold and the global interrupt is on disable state regardless of the IE.7 (EA) bit. But should be set the global interrupt is disabled (EA = 0) before the CRC operation is started in use auto CRC/Checksum mode, recommend.

**Figure 73. Program Tip for CRC Operation in Auto CRC/Checksum Mode**

The CRC operation procedure in User CRC/Checksum mode is introduced in the following list, and figure 74 shows example program tip:

1. Select User CRC/Checksum Mode and CRC
2. Clear Flash CRC data register (FCDRH/FCDRL)
3. Read data from the Flash
4. Write the data to FCDIN Register
5. Read the CRC result

```
unsigned char code *rom_addr=0x0000;
unsigned int i=0;

       FCCR |= _1000_0000; // Flash CRC User CRC/Checksum Mode
       FCCR &= _1101_1111; // Flash CRC CRC Mode
       FCCR |= _0100_0000; // Flash CRC data register clear

       for(i=0x0000; i <= 0x3FFF; i++)   // 0000H~3FFFH
       {
              FCDIN = rom_addr[i];
              WDTCR |= _0010_0000;       // Clear WDT counter
       }

//**** Read CRC result
       Temp0 = FCDRH;
       Temp1 = FCDRL;
```

**Figure 74. Program Tip for CRC Operation in User CRC/Checksum Mode**

The Checksum operation procedure in Auto CRC/Checksum mode is introduced in the following list, and figure 75 shows example program tip:

1. Global interrupt Disable (EA = 0)
2. Select Auto CRC/Checksum Mode and Checksum
3. Select CRC Clock
4. Set CRC start address register (FCSARH/FCSARM/FCSARL)
5. Set CRC end address register (FCEARH/FCEARM/FCEARL)
6. CRC operation starts (CRCRUN = 1)
7. Read the Checksum result
8. Global interrupt Enable (EA = 1)

```
//**** Global interrupt Disable
EA = 0;

//**** Flash CRC Auto CRC/Checksum Mode and Checksum
FCCR &= _0111_1111;
FCCR |= _0010_0000;          // Checksum mode

OSCCR &= _1111_1011;         // IRC Enable
FCCR &= _1111_0001;          // CRC clk = fIRC/1

//**** Checksum start address set
FCSARH = 0x00;
    FCSARM = 0x00;
    FCSARL = 0x00;

//**** Checksum end address set
    FCEARH = 0x00;
    FCEARM = 0x3F;
    FCEARL = 0xFF;

//**** Checksum start
FCCR |= _0000_0001;
_nop_();            //Dummy instruction, This instruction must be needed.
_nop_();            //Dummy instruction, This instruction must be needed.
_nop_();            //Dummy instruction, This instruction must be needed.

//**** Read Checksum result
Temp0 = FCDRH;
Temp1 = FCDRL;

//**** Global interrupt Enable
EA = 1;
```

**NOTES**:

1. Three or more NOP instructions must immediately follow the Checksum start operation in auto CRC/Checksum mode.
2. During a checksum operation (when CRCRUN bit is Running state) in auto CRC/Checksum mode, the CPU is hold and the global interrupt is on disable state regardless of the IE.7 (EA) bit. But should be set the global interrupt is disabled (EA = 0) before the Checksum operation is started in use auto CRC/Checksum mode, recommend.

**Figure 75. Program Tip for Checksum Operation in Auto CRC/Checksum Mode**

The Checksum operation procedure in User CRC/Checksum mode is introduced in the following list, and figure 76 shows example program tip:

1. Select User CRC/Checksum Mode and Checksum
2. Clear Flash CRC data register (FCDRH/FCDRL)
3. Read data from the Flash
4. Write the data to FCDIN Register

5. Read the Checksum result

```
unsigned char code *rom_addr=0x0000;
unsigned int i=0;

FCCR |= _1000_0000;                  // Flash CRC User CRC/Checksum Mode
FCCR &= ~0010_0000;                  // Flash CRC Checksum
FCCR |= _0100_0000;                  // Flash CRC data register clear

for(i=0x0000; i <= 0x3FFF; i++)      // 0000H~3FFFH
{
    FCDIN = rom_addr[i];
    WDTCR |= _0010_0000;             // Clear WDT counter
}

//**** Read Checksum result
       Temp0 = FCDRH;
       Temp1 = FCDRL;
```

**Figure 76. Program Tip for Checksum Operation in User CRC/Checksum Mode**

## 17.3 Register map

**Table 24. Flash CRC/Checksum Generator Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| FCSARH | 5050H (XSFR) | R/W | 00H | Flash CRC Start Address High Register |
| FCEARH | 5051H (XSFR) | R/W | 00H | Flash CRC End Address High Register |
| FCSARM | 5052H (XSFR) | R/W | 00H | Flash CRC Start Address Middle Register |
| FCEARM | 5053H (XSFR) | R/W | 00H | Flash CRC End Address Middle Register |
| FCSARL | 5054H (XSFR) | R/W | 00H | Flash CRC Start Address Low Register |
| FCEARL | 5055H (XSFR) | R/W | 0FH | Flash CRC End Address Low Register |
| FCCR | 5056H (XSFR) | R/W | 00H | Flash CRC Control Register |
| FCDRH | 5057H (XSFR) | R | FFH | Flash CRC Data High Register |
| FCDRL | 5058H (XSFR) | R | FFH | Flash CRC Data Low Register |
| FCDIN | D7H | R/W | 00H | Flash CRC Data In Register |

## 17.4 Register description

### FCSARH (Flash CRC Start Address High Register): 5050H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | FCSARH0 |
| – | – | – | – | – | – | – | R/W |

Initial value: 00H

| | |
|---|---|
| FCSARH0 | Flash CRC Start Address High<br>**NOTE:** Used only to Auto CRC Mode. |

### FCSARM (Flash CRC Start Address Middle Register): 5052H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCSARM7 | FCSARM6 | FCSARM5 | FCSARM4 | FCSARM3 | FCSARM2 | FCSARM1 | FCSARM0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| FCSARM[7:0] | Flash CRC Start Address Middle<br>**NOTE:** Used only to Auto CRC Mode. |

### FCSARL (Flash CRC Start Address Low Register): 5054H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCSARL7 | FCSARL6 | FCSARL5 | FCSARL4 | FCSARL3 | FCSARL2 | FCSARL1 | FCSARL0 |
| R/W | R/W | R/W | R/W | – | – | – | – |

Initial value: 00H

| | |
|---|---|
| FCSARL[7:4] | Flash CRC Start Address Low<br>**NOTE**: Used only to Auto CRC Mode. |
| FCSARL[3:0] | These bits are always "0000b". |

### FCEARH (Flash CRC End Address High Register): 5051H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | – | FCEARH0 |
| – | – | – | – | – | – | – | R/W |

Initial value: 00H

| | |
|---|---|
| FCEARH0 | Flash CRC End Address High<br>**NOTE:** Used only to Auto CRC Mode. |

### FCEARM (Flash CRC End Address Middle Register): 5053H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCEARM7 | FCEARM6 | FCEARM5 | FCEARM4 | FCEARM3 | FCEARM2 | FCEARM1 | FCEARM0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

| | |
|---|---|
| FCEARM[7:0] | Flash CRC End Address Middle<br>**NOTE:** Used only to Auto CRC Mode. |

### FCEARL (Flash CRC End Address Low Register): 5055H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCEARL7 | FCEARL6 | FCEARL5 | FCEARL4 | FCEARL3 | FCEARL2 | FCEARL1 | FCEARL0 |
| R/W | R/W | R/W | R/W | – | – | – | – |

Initial value: 0FH

FCEARL[7:4] Flash CRC End Address Low
**NOTE:** Used only to Auto CRC Mode.

FCEARL[3:0] These bits are always "1111b".

### FCDRH (Flash CRC Data High Register): 5057H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCDRH7 | FCDRH6 | FCDRH5 | FCDRH4 | FCDRH3 | FCDRH2 | FCDRH1 | FCDRH0 |
| R | R | R | R | R | R | R | R |

Initial value: FFH

FCDRH[7:0] Flash CRC Data High

### FCDRL (Flash CRC Data Low Register): 5058H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCDRL7 | FCDRL6 | FCDRL5 | FCDRL4 | FCDRL3 | FCDRL2 | FCDRL1 | FCDRL0 |
| R | R | R | R | R | R | R | R |

Initial value: FFH

FCDRL[7:0] Flash CRC Data Low

### FCDIN (Flash CRC Data IN Register): D7H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FCDIN7 | FCDIN6 | FCDIN5 | FCDIN4 | FCDIN3 | FCDIN2 | FCDIN1 | FCDIN0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

FCDIN[7:0] Flash CRC Data In
**NOTE:** Used only to User CRC Mode.

### FCCR (Flash CRC Control Register): 5056H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| CRCMOD | CDCL | MDSEL | – | CKSEL2 | CKSEL1 | CKSEL0 | CRCRUN |
| R/W | R/W | R/W | – | R/W | R/W | R/W | R/W |

Initial value: 00H

CRCMOD Select CRC/Checksum Mode
- 0 Auto CRC/Checksum Mode
- 1 User CRC/Checksum Mode

CDCL Flash CRC Data Register Clear
- 0 No effect
- 1 Clear Flash CRC Data register

**NOTE:** This bit is cleared to '0' automatically, after Flash CRC Data register is cleared. The FCDRH/L is set to "FFH" if the MDSEL is set to "0b" and "00H" if the MDSEL is set to "1b". Used only to User CRC/Checksum Mode.

MDSEL CRC/Checksum Selection
- 0 Select CRC
- 1 Select Checksum

CKSEL[2:0] Select Flash CRC/Checksum Clock

| CKSEL2 | CKSEL1 | CKSEL0 | Description |
|---|---|---|---|
| 0 | 0 | 0 | $f_{IRC}$ |
| 0 | 0 | 1 | $f_{IRC}/2$ |
| 0 | 1 | 0 | $f_{IRC}/4$ |
| 0 | 1 | 1 | $f_{IRC}/8$ |
| 1 | 0 | 0 | $f_X$(system clock) |
| Other values | | | Not used |

CRCRUN CRC/Checksum Start Signal & Busy Flag, Used only to Auto CRC/Checksum mode.
- 0 Indicates that CRC/Checksum operation is not running or has finished. When written "0", CRC/Checksum operation is finished by force even if CRC/Checksum operation is running. It has no effect to write "0" if CRC/Checksum is not running currently.
- 1 When written "1", CRC/Checksum operation starts and this bit remains "1" as long as CRC/Checksum operation is on-going. This bit is cleared to "0" automatically after CRC/Checksum operation finishes.

# 18 Power down operation

A96L322 offers two power-down modes to minimize power consumption of itself. Programs under the two power saving modes IDLE and STOP, are stopped and power consumption is reduced considerably.

## 18.1 Peripheral operation in IDLE/STOP mode

Peripheral's operations during IDLE/STOP mode is introduced in table 25.

**Table 25. Peripheral Operation during Power-down Mode**

| Peripheral | IDLE mode | STOP mode |
|---|---|---|
| CPU | ALL CPU Operation are Disable | ALL CPU Operation are Disable |
| RAM | Retain | Retain |
| Basic Interval Timer | Operates Continuously | Stop |
| Watch Dog Timer | Operates Continuously | Stop (Can be operated with WDTRC OSC) |
| Timer0~1 | Operates Continuously | Halted (Only when the Event Counter Mode is Enabled, Timer operates Normally) |
| ADC | Operates Continuously | Stop |
| USART | Operates Continuously | Stop |
| Siren | Operates Continuously | Stop |
| Line Interface | Operates Continuously | Stop |
| Internal OSC | Oscillation | Stop |
| WDTRC OSC (1KHz) | Can be operated with setting value | Can be operated with setting value |
| Constant Sink Current | Retain | Retain |
| I/O Port | Retain | Retain |
| Control Register | Retain | Retain |
| Address Data Bus | Retain | Retain |
| Release Method | By RESET, all Interrupts | By RESET, Timer Interrupt (EC0, EC1),External Interrupt, WDT, USART |

## 18.2 IDLE mode

The power control register is set to '01h' to enter the IDLE Mode. In this mode, the internal oscillation circuits remain active. Oscillation continues and peripherals are operated normally but CPU stops. It is released by reset or interrupt. To be released by interrupt, interrupt should be enabled before IDLE mode. If using reset, because the device becomes initialized state, the registers have reset value.

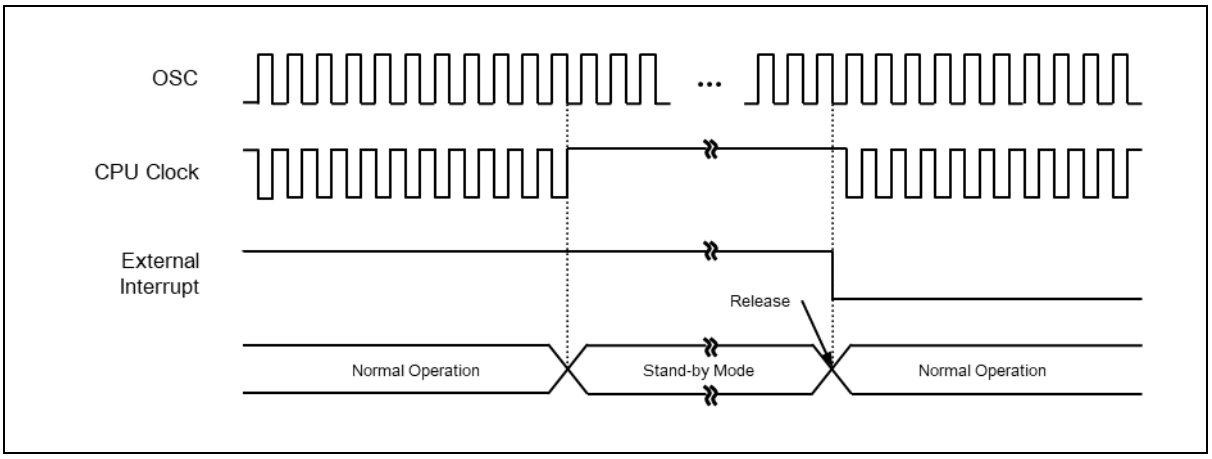



Figure 77. IDLE Mode Release Timing by External Interrupt

## 18.3 STOP mode

The power control register is set to '03H' to enter the STOP Mode. In the stop mode, the selected oscillator, system clock and peripheral clock is stopped. With the clock frozen, all functions are stopped, but the on-chip RAM and control registers are held.

The source for exit from STOP mode is hardware reset and interrupts. The reset re-defines all the control registers.

When exit from STOP mode, enough oscillation stabilization time is required to normal operation. Figure 78 shows the timing diagram. When released from STOP mode, the Basic interval timer is activated on wake-up. Therefore, before STOP instruction, user must be set its relevant prescale divide ratio to have long enough time. This guarantees that oscillator has started and stabilized.

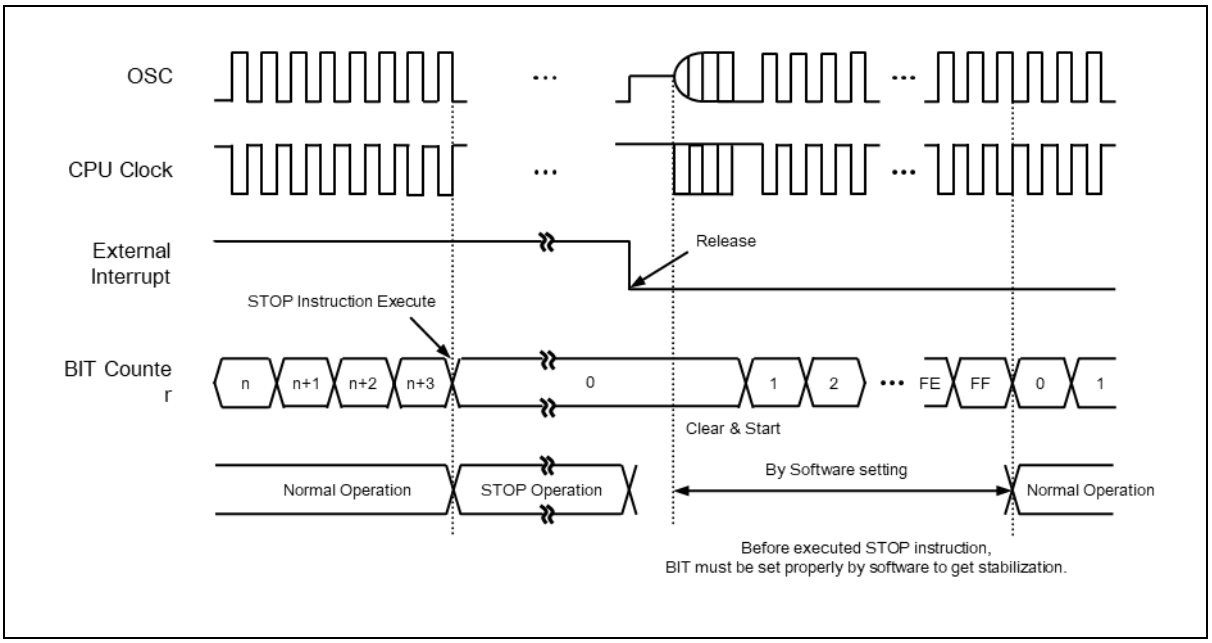



**Figure 78. STOP Mode Release Timing by External Interrupt**

## 18.4 Release operation of STOP mode

After STOP mode is released, the operation begins according to content of related interrupt register just before STOP mode start (figure 79).

If the global interrupt Enable Flag (IE.EA) is set to `1`, the STOP mode is released by the interrupt which each interrupt enable flag = `1` and the CPU jumps to the relevant interrupt service routine. Even if the IE.EA bit is cleared to '0', the STOP mode is released by the interrupt of which the interrupt enable flag is set to '1'.

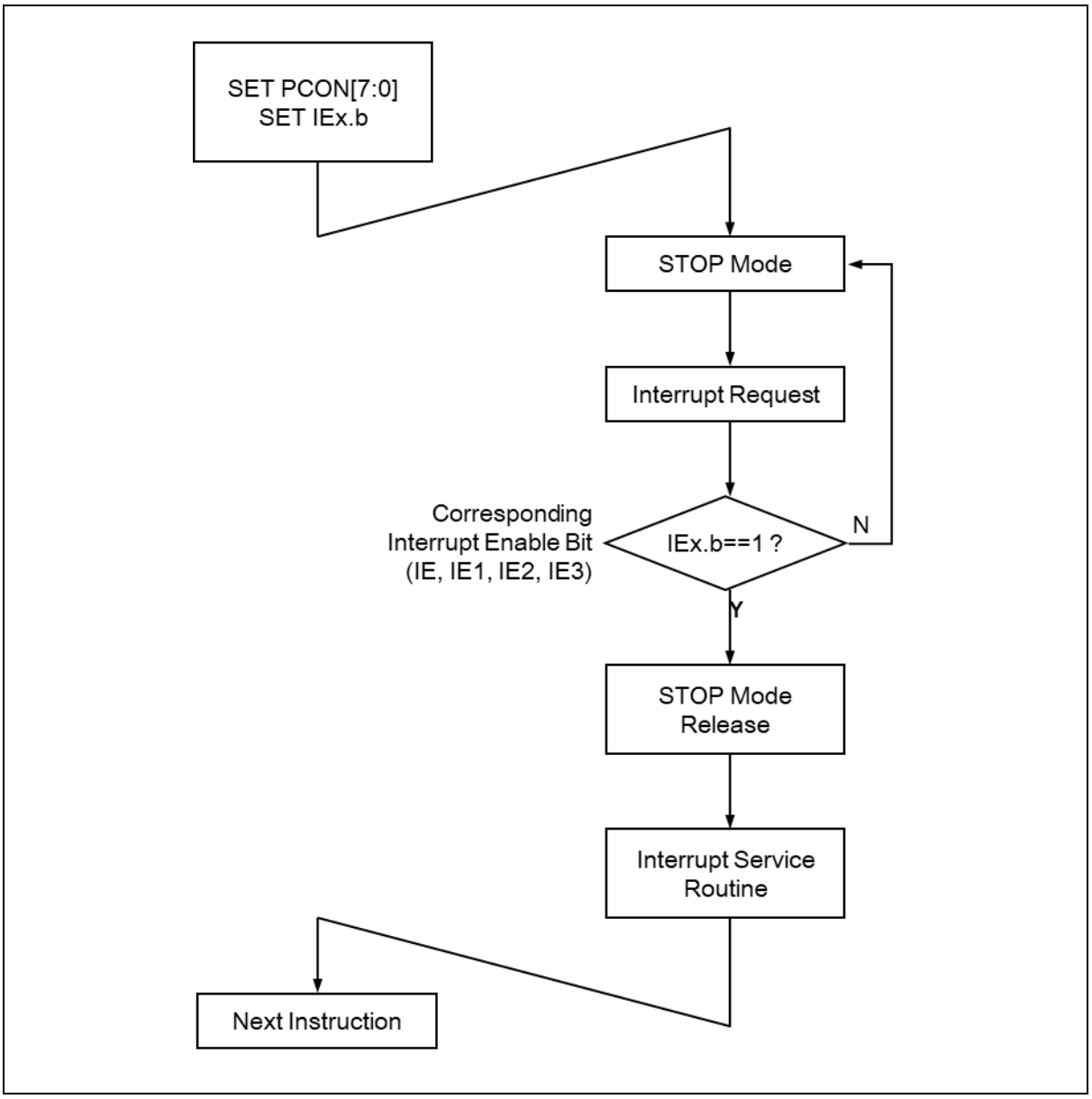



Figure 79. STOP Mode Release Flow

## 18.5 Register map

**Table 26. Power-down Operation Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| PCON | 87H | R/W | 00H | Power control register |

## 18.6 Register description

**PCON (Power Control Register): 87H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | – | PCON1 | PCON0 |
| – | – | – | – | – | – | R/W | R/W |

Initial value: 00H

| PCON[1:0] | Power Control |
|---|---|
| 01H | IDLE mode enable |
| 03H | STOP mode enable |
| Other Values | Normal operation |

Before configuring a register PCON, please be aware of the followings:

1. To enter IDLE mode, PCON must be set to '01H'.
2. To enter STOP mode, PCON must be set to '03H'.
3. The PCON register is automatically cleared by a release signal in STOP/IDLE mode.
4. Three or more NOP instructions must follow immediately after the instruction that makes the device enter in STOP/IDLE mode. Refer to the following example code in table 27.

**Table 27. Example Code with 3 or more NOP Instructions**

| Example code 1 | Example code 2 |
|---|---|
| MOV PCON, #01H ; IDLE mode<br>NOP<br>NOP<br>NOP<br>•<br>•<br>• | MOV PCON, #03H ; STOP mode<br>NOP<br>NOP<br>NOP<br>•<br>•<br>• |

# 19 Reset

When a reset event occurs, an internal register is selected to be initialized in accordance with a reset value. Each reset value introduced in table 28 indicates a corresponding On Chip Hardware that is to be initialized.

**Table 28. Reset Value and the Relevant On Chip Hardware**

| On Chip Hardware | Reset Value |
|---|---|
| Program Counter (PC) | 0000H |
| Accumulator | 00H |
| Stack Pointer (SP) | 07H |
| Peripheral clock | On |
| Control register | Refer to peripheral registers. |

A96L322 has 5 types of reset sources as listed in the followings:

- External RESETB
- Power On RESET (POR)
- WDT overflow reset (in a case of WDTEN='1')
- Low voltage reset (in a case of LVREN='0')
- OCD RESET

## 19.1 Reset block diagram

Figure 80 shows a reset block of A96L322.

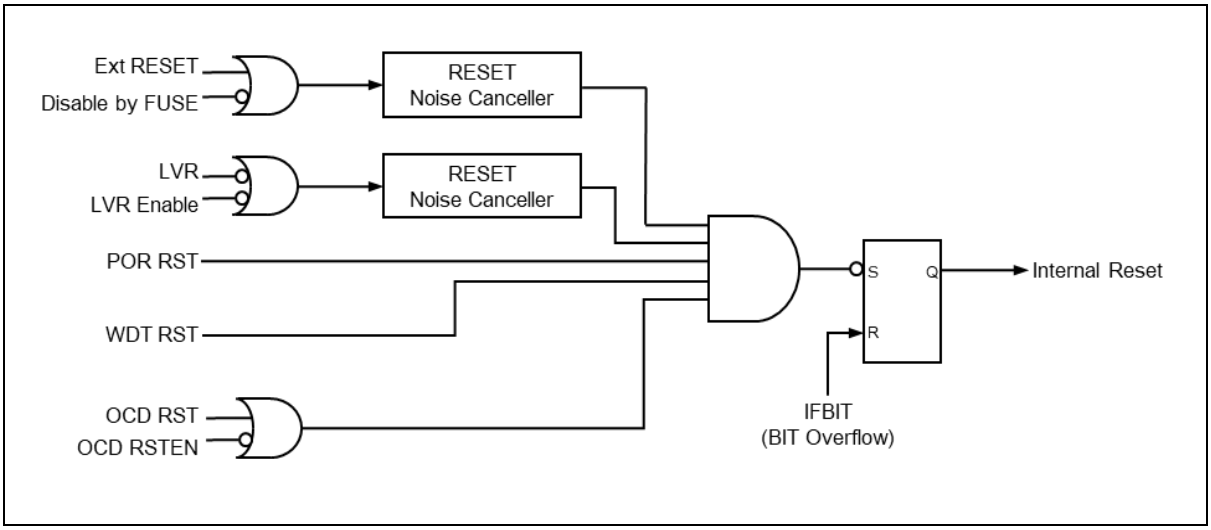



**Figure 80. Reset Block Diagram**

## 19.2 Reset noise canceller

Figure 81 is a noise canceller timing diagram for noise cancellation of RESET. It has the noise cancellation value of about 2us (@VDD=5V) to the low input of system reset.

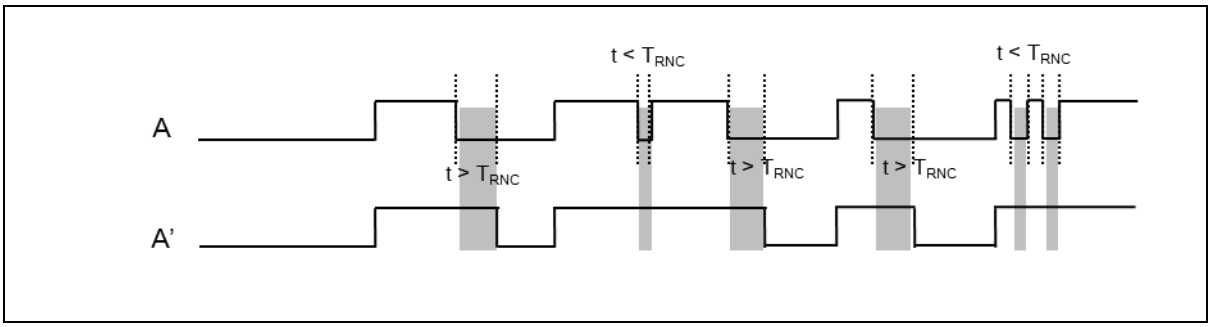



Figure 81. Reset Noise Canceller Timing Diagram

## 19.3 Power on Reset

When device power is increasing, POR (Power On Reset) executes a function to reset the device. If POR is used to reset the device, it executes the device reset function instead of RESET IC or RESET Circuit.

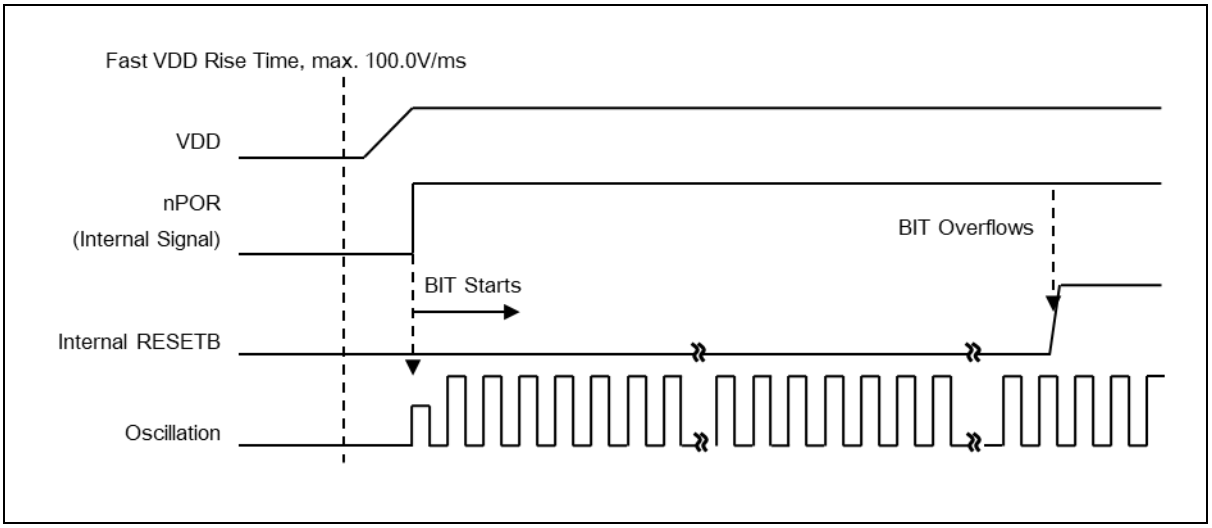



Figure 82. Fast VDD Rising Time

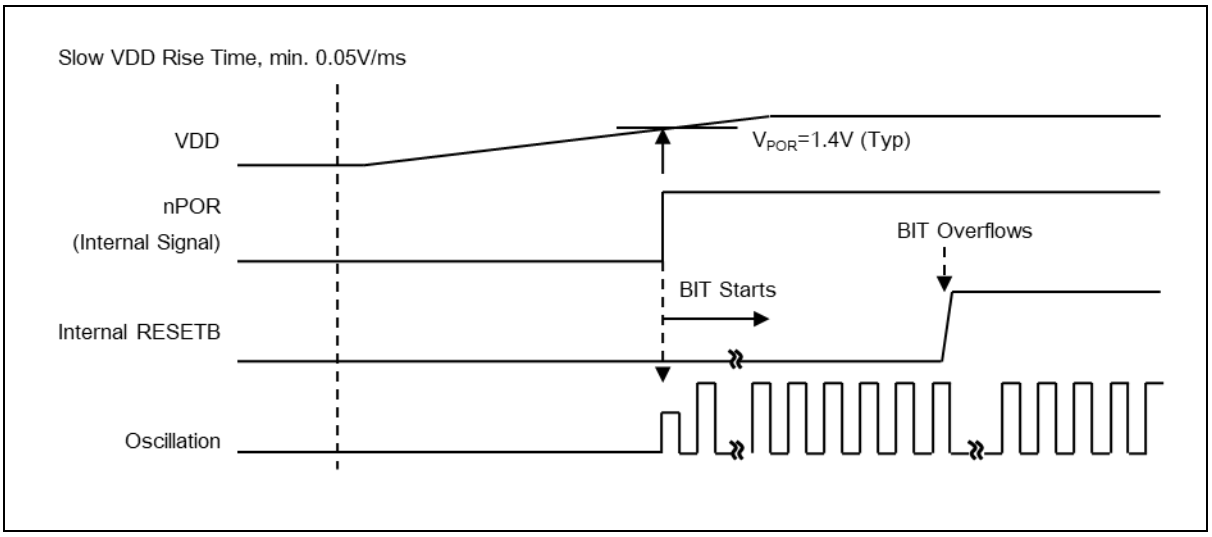



Figure 83. Internal Reset Release Timing on Power-Up

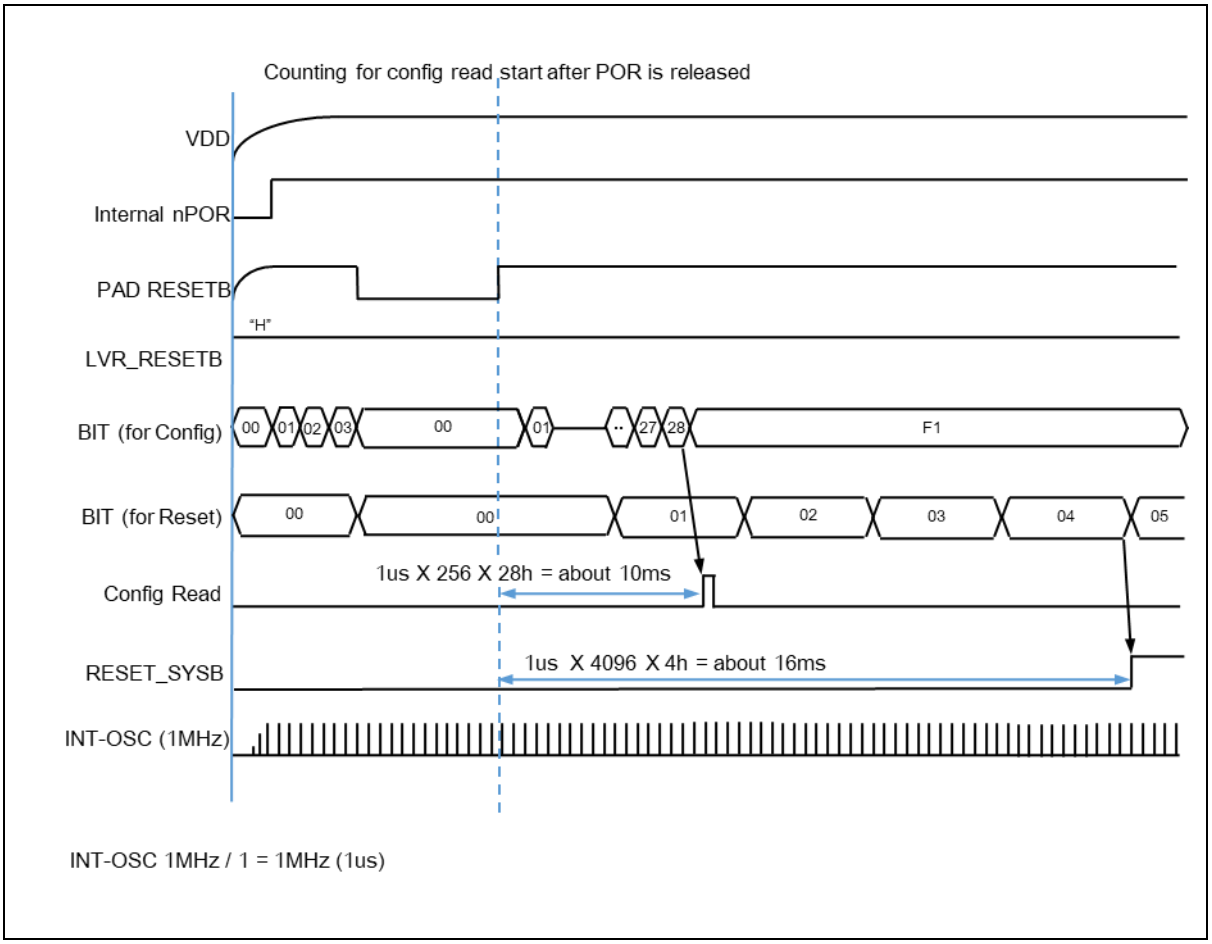



Figure 84. Configuration Timing when Power-On

Relationship between VDD input and internal oscillator is described in figure 85 and table 29.

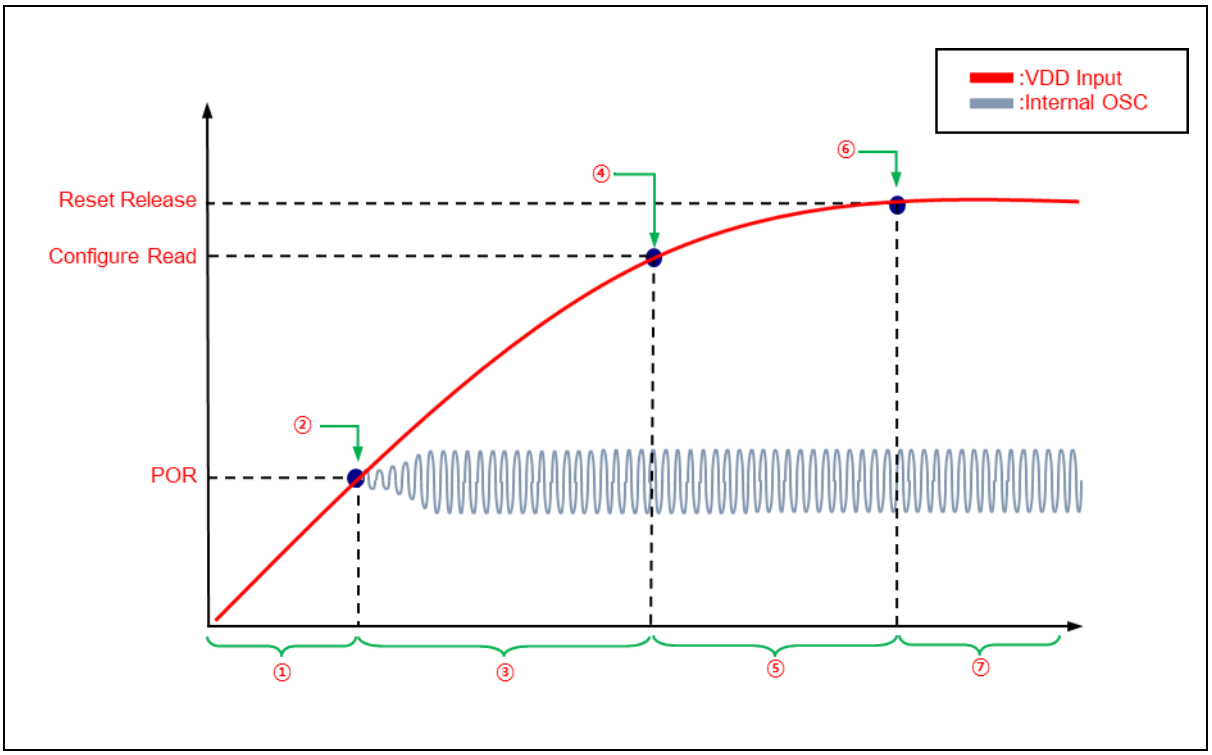



Figure 85. Boot Process Waveform

Table 29. Boot Process Description

| Process | Description | Remark |
|---|---|---|
| ① | No operation | |
| ② | 1st POR level detection | About 1.4V |
| ③ | • (INT-OSC 1MHz/1) x 256 x 28h Delay section (=10ms)<br>• VDD input voltage must rise over than flash operating voltage for Config read. | Slew rate >= 0.05V/ms |
| ④ | Config read point | • About 1.5V ~ 1.6V<br>• Config Value is determined by Writing Option. |
| ⑤ | Rising section to reset release level | 16ms point after POR or Ext_reset release |
| ⑥ | Reset release section (BIT overflow)<br>• After16ms, after external reset release (external reset)<br>• 16ms point after POR (POR only) | BIT is used for peripheral stability. |
| ⑦ | Normal operation | |

## 19.4 External RESETB input

External RESETB is the input to a Schmitt trigger. If RESETB pin is held with low for at least 10us over within the operating voltage range and stable oscillation, it is applied and the internal state is initialized. When reset state becomes '1', it needs stabilization time with 16ms and after the stable state, the internal RESET becomes '1'. The Reset process step needs 5 oscillator clocks. And the program execution starts at the vector address stored at address 0000H.

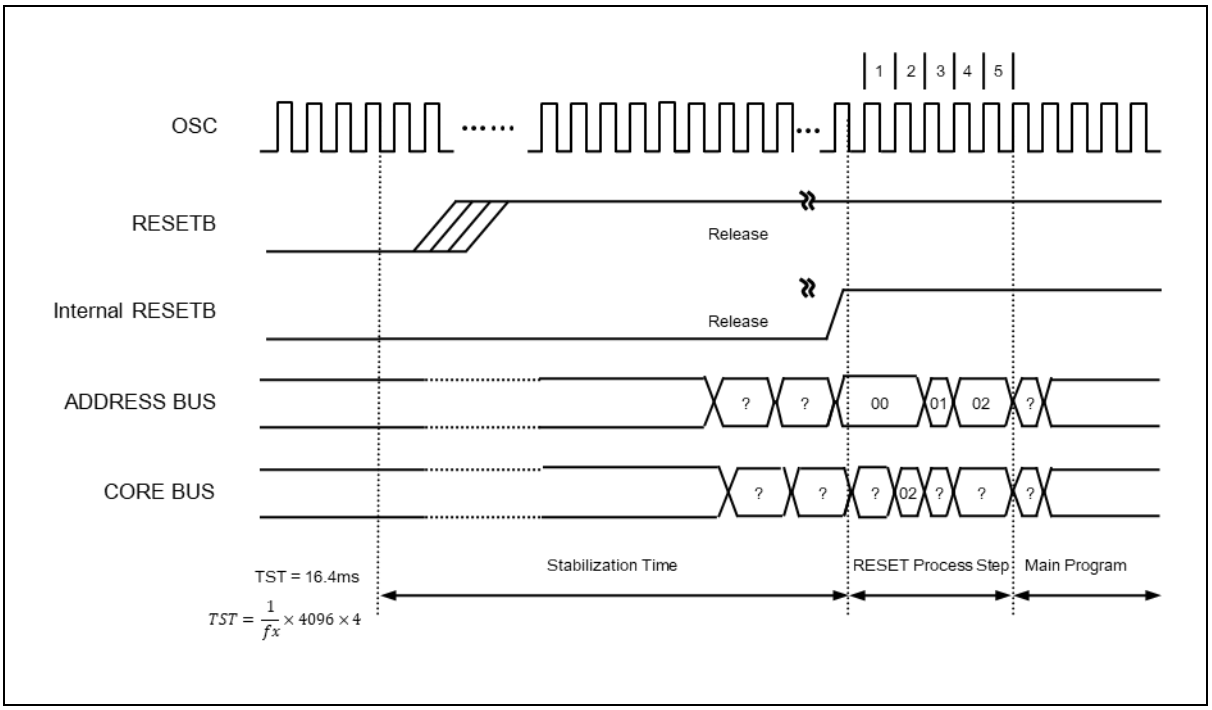
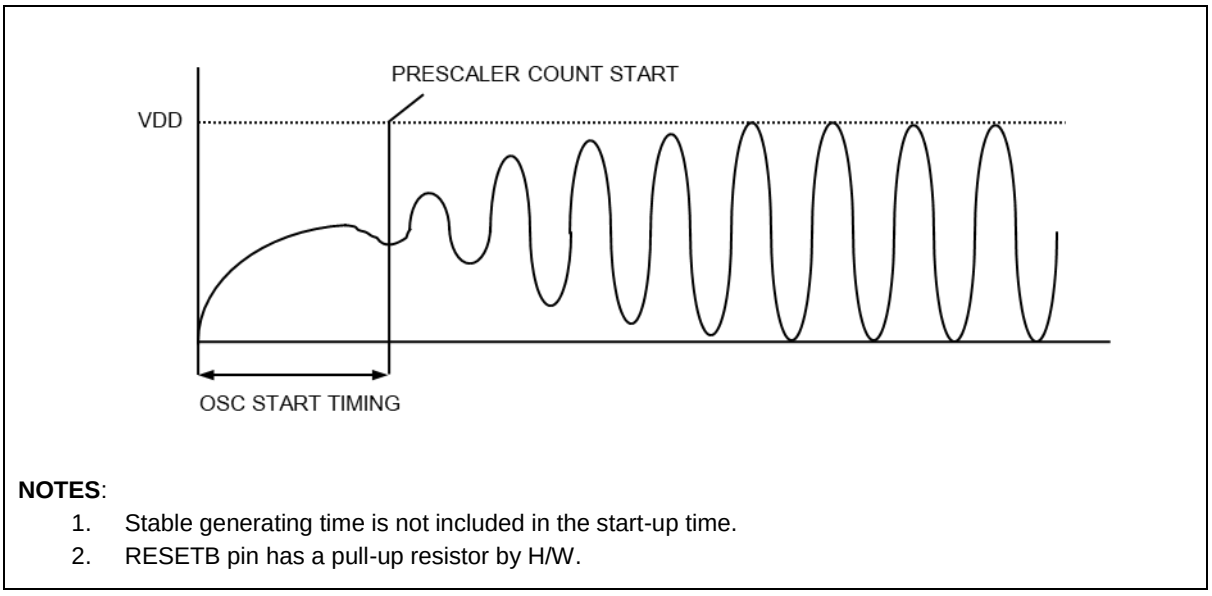



Figure 86. Timing Diagram after RESET




**NOTES**:

1. Stable generating time is not included in the start-up time.
2. RESETB pin has a pull-up resistor by H/W.

Figure 87. Oscillator Generating Waveform Example

## 19.5 Brown out detector processor

A96L322 has an On-chip brown-out detection circuit (BOD) to monitor VDD level during its operation. It compares VDD level to a fixed trigger level which can be selected to be one of 1.60V, 2.20V, and 2.70V by LVRVS[1:0] bits. In STOP mode, since the BOD will contribute significantly to the total current consumption, the LVREN bit is set to off by software to minimize the current consumption.

### 19.5.1 Block diagram

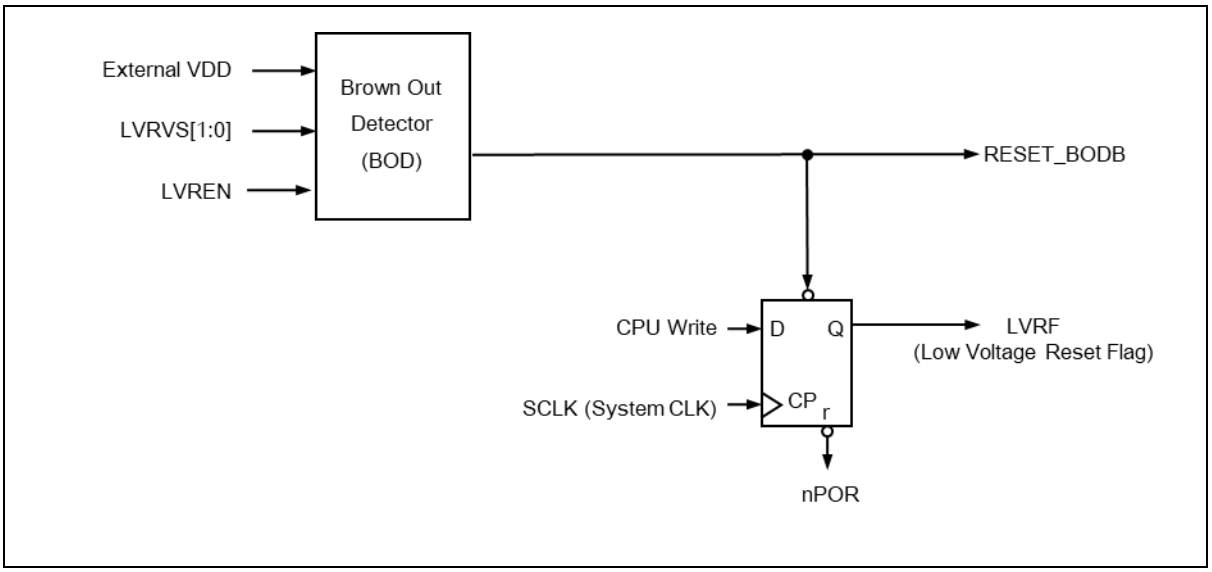



Figure 88. BOD Block Diagram

### 19.5.2 Internal reset and BOD reset in timing diagram

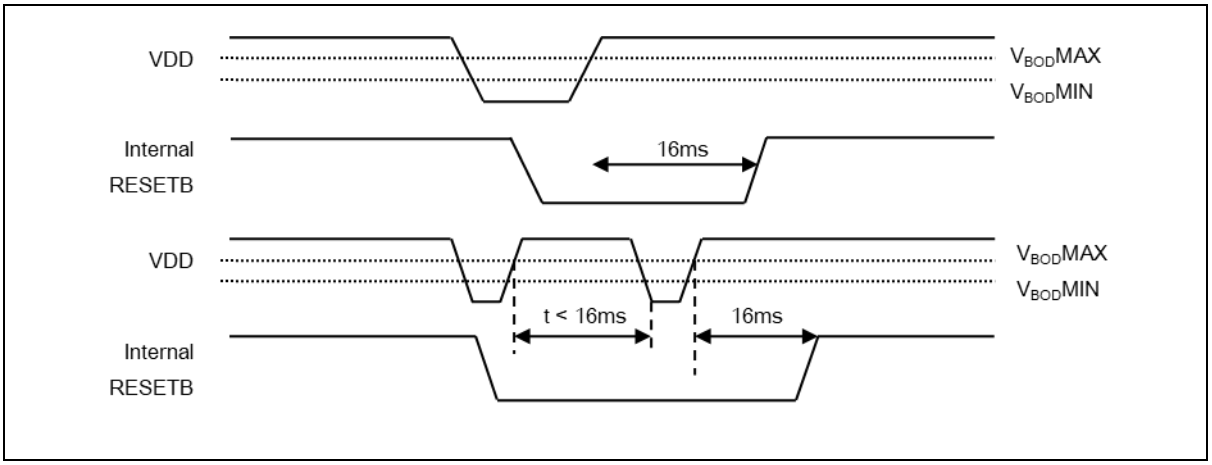



Figure 89. Internal Reset at Power Fail Situation

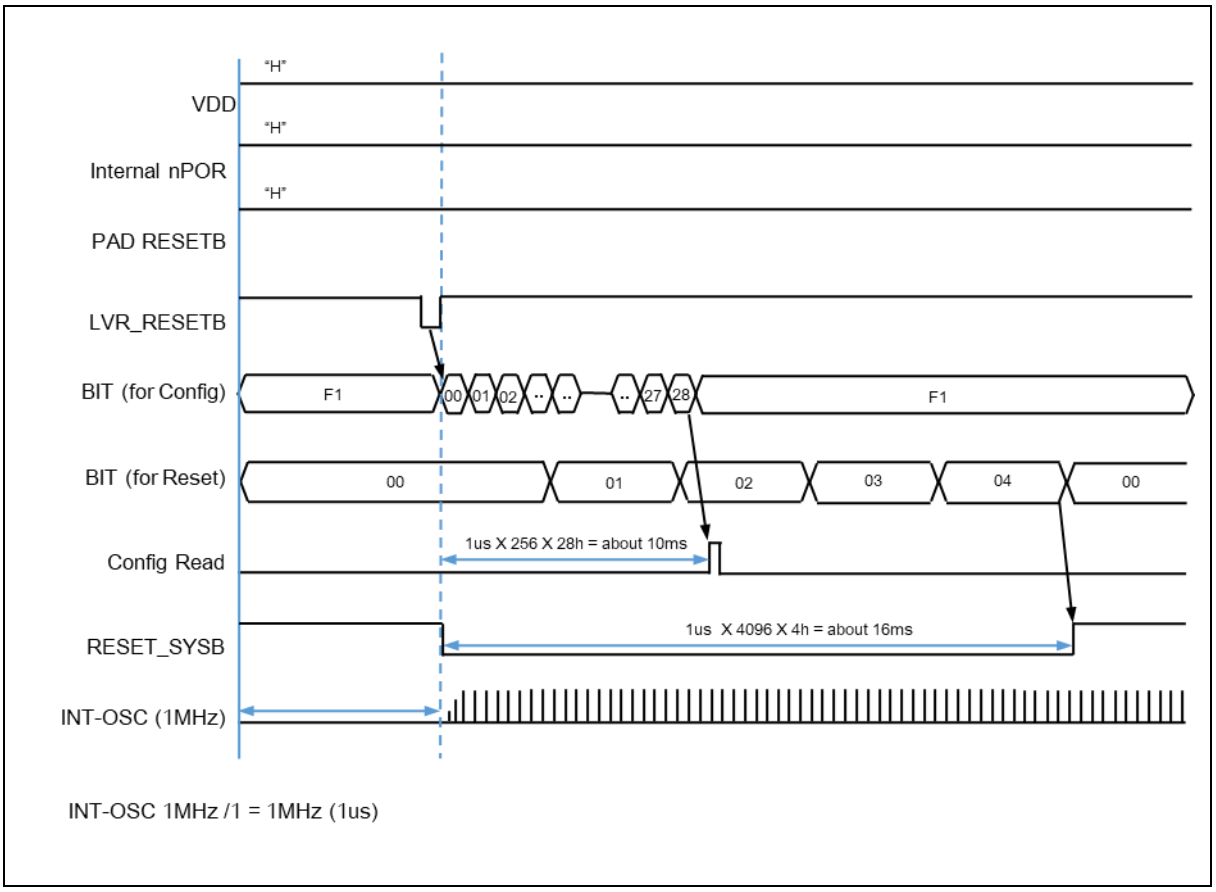



Figure 90. Configuration Timing when BOD Reset

## 19.6 Register map

Table 30. Reset Operation Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| RSTFR | E8H | R/W | 80H | Reset Flag Register |
| LVRCR | D8H | R/W | 00H | Low Voltage Reset Control Register |
| LVRIDR | 505FH (XSFR) | R/W | 00H | LVR Write Identification Register |

## 19.7 Register description

**RSTFR (Reset Flag Register): E8H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| PORF | EXTRF | WDTRF | OCDRF | LVRF | – | – | – |
| R/W | R/W | R/W | R/W | R/W | – | – | – |

Initial value: 80H

PORF Power-On Reset flag bit. The bit is reset by writing '0' to this bit.
0 No detection
1 Detection

EXTRF External Reset (RESETB) flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
0 No detection
1 Detection

WDTRF Watch Dog Reset flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
0 No detection
1 Detection

OCDRF On-Chip Debug Reset flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
0 No detection
1 Detection

LVRF Low Voltage Reset flag bit. The bit is reset by writing '0' to this bit or by Power-On Reset.
0 No detection
1 Detection

**NOTES**

1. When the Power-On Reset occurs, the PORF bit is only set to "1", the WDTRF/OCDRF bits are cleared to "0".
2. When the Power-On Reset occurs, the EXTRF bit is unknown, at that time, the EXTRF bit can be set to "1" when External Reset (RESETB) occurs.
3. When the Power-On Reset occurs, the LVRF bit is unknown, at that time, the LVRF bit can be set to "1" when LVR Reset occurs.
4. When a reset except the POR occurs, the corresponding flag bit is only set to "1", the other flag bits are kept in the previous values.

### LVRCR (Low Voltage Reset Control Register): D8H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LVRST | – | – | – | – | LVRVS1 | LVRVS0 | LVREN |
| R/W | – | – | – | – | R/W | R/W | R/W |

Initial value: 00H

LVRST LVR Enable when Stop Release
0 Not effect at stop release
1 LVR enable at stop release
**NOTES:**
When this bit is '1', the LVREN bit is cleared to '0' by stop mode release. (LVR enable)
When this bit is '0', the LVREN bit is not effect by stop mode release.

LVRVS[1:0] LVR Voltage Select

| LVRVS1 | LVRVS0 | Description |
|---|---|---|
| 0 | 0 | 1.60V |
| 0 | 1 | 2.20V |
| 1 | 0 | 2.70V |
| 1 | 1 | Not available |

LVREN LVR Operation
0 LVR Enable
1 LVR Disable

**NOTES:**
1. The LVRST and LVRVS[1:0] bits are cleared by a power-on reset but are retained by other reset signals.
2. The LVRVS[1:0] bits should be set to '00b' while LVREN bit is "1".
3. This register can be written with valid ID value (LVRIDR == 0x59).

### LVRIDR (LVR Write Identification Register): 505FH (XSFR)

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| LVRID7 | LVRID6 | LVRID5 | LVRID4 | LVRID3 | LVRID2 | LVRID1 | LVRID0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

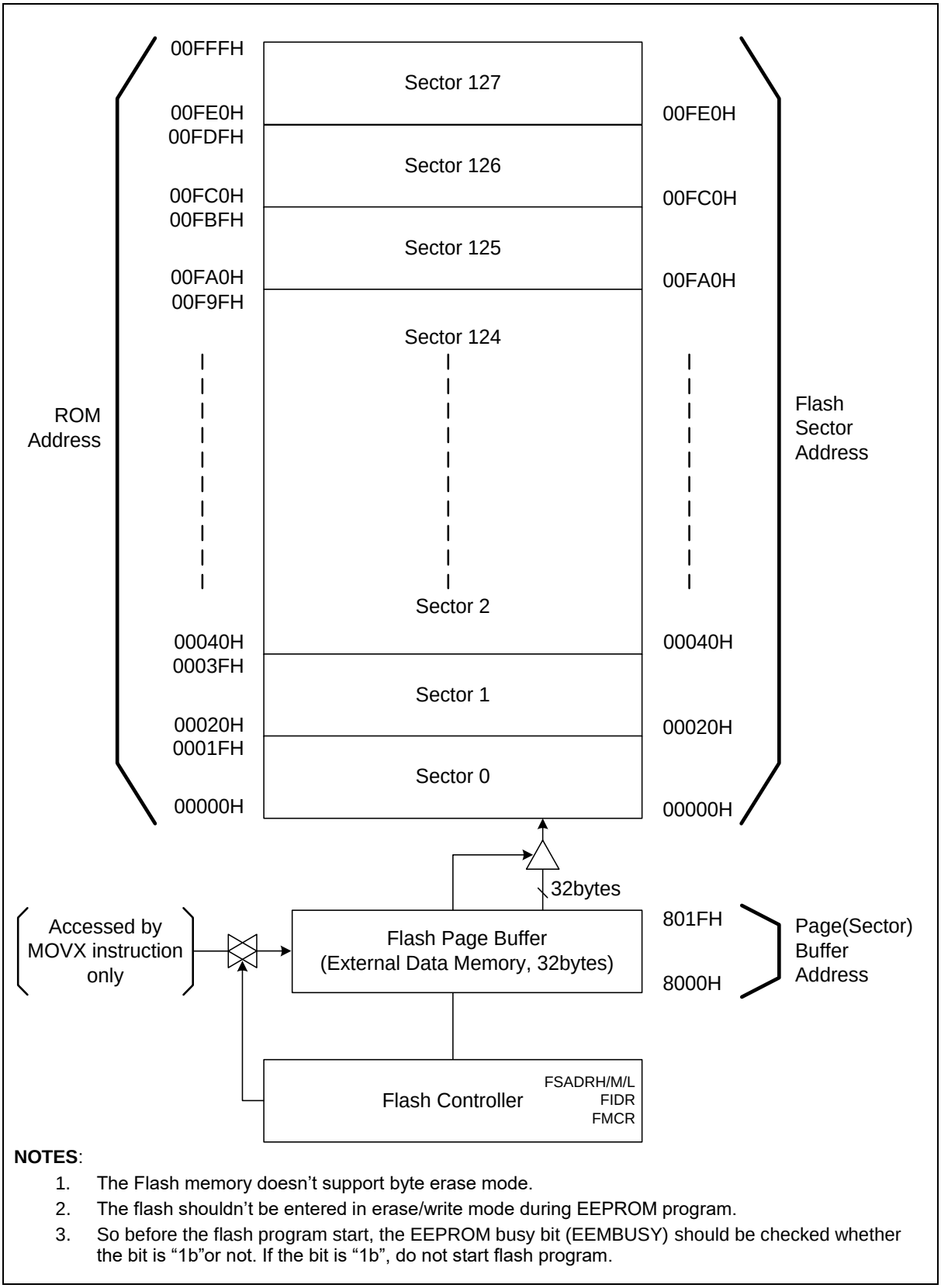
LVRID[7:0] LVR Write Identification
Others No identification value
01011001b Identification value for LVR register write
(These bits are automatically cleared to logic '00H' immediately after one time operation)

# 20 Flash memory

A96L322 incorporates flash memory inside. Program can be written, erased, and overwritten on the flash memory while it is mounted on a board. The flash memory can be read by 'MOVC' instruction and programmed in OCD, serial ISP mode or user program mode. Followings are features summary of flash memory.

- Flash Size : 4Kbytes
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 10,000 program/erase cycles at typical voltage and temperature for flash memory

## 20.1 Flash program ROM structure




**NOTES**:

1. The Flash memory doesn't support byte erase mode.
2. The flash shouldn't be entered in erase/write mode during EEPROM program.
3. So before the flash program start, the EEPROM busy bit (EEMBUSY) should be checked whether the bit is "1b"or not. If the bit is "1b", do not start flash program.

**Figure 91. Flash program ROM structure**

## 20.2 Register map

**Table 31. Flash Memory Register Map**

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| FSADRH | FAH | R/W | 00H | Flash Sector Address High Register |
| FSADRM | FBH | R/W | 00H | Flash Sector Address Middle Register |
| FSADRL | FCH | R/W | 00H | Flash Sector Address Low Register |
| FIDR | FDH | R/W | 00H | Flash Identification Register |
| FMCR | FEH | R/W | 00H | Flash Mode Control Register |

## 20.3 Register description

### FSADRH (Flash Sector Address High Register): FAH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | FSADRH3 | FSADRH 2 | FSADRH1 | FSADRH0 |
| – | – | – | – | R/W | R/W | R/W | R/W |

Initial value: 00H

FSADRH[3:0] Flash Sector Address High

### FSADRM (Flash Sector Address Middle Register): FBH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FSADRM7 | FSADRM6 | FSADRM5 | FSADRM4 | FSADRM3 | FSADRM2 | FSADRM1 | FSADRM0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

FSADRM[7:0] Flash Sector Address Middle

### FSADRL (Flash Sector Address Low Register): FCH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FSADRL7 | FSADRL6 | FSADRL5 | FSADRL4 | FSADRL3 | FSADRL2 | FSADRL1 | FSADRL0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

FSADRL[7:0] Flash Sector Address Low

### FIDR (Flash Identification Register): FDH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FIDR7 | FIDR6 | FIDR5 | FIDR4 | FIDR3 | FIDR2 | FIDR1 | FIDR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

FIDR[7:0] Flash Identification
Others No identification value
10100101 Identification value for a flash mode
(These bits are automatically cleared to logic '00H' immediately after one time operation except "flash page buffer reset mode")

### FMCR (Flash Mode Control Register): FEH

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| FMBUSY | – | – | – | – | FMCR2 | FMCR1 | FMCR0 |
| R | – | – | – | – | R/W | R/W | R/W |

Initial value: 00H

FMBUSY Flash Mode Busy Bit. This bit will be used for only debugger.
0 No effect when "1" is written
1 Busy

FMCR[2:0] Flash Mode Control Bits. During a flash mode operation, the CPU is hold and the global interrupt is on disable state regardless of the IE.7 (EA) bit.

| FMCR2 | FMCR1 | FMCR0 | Description |
|---|---|---|---|
| 0 | 0 | 1 | Select flash page buffer reset mode and start regardless of the FIDR value (Clear all 32bytes to '0') |
| 0 | 1 | 0 | Select flash sector erase mode and start operation when the FIDR="10100101b' |
| 0 | 1 | 1 | Select flash sector write mode and start operation when the FIDR="10100101b' |
| 1 | 0 | 0 | Select flash hard lock and start operation when the FIDR="10100101b' |

Others Values: No operation
(These bits are automatically cleared to logic '00H' immediately after one time operation)

## 20.4 Serial In-System Program (ISP) mode

Serial in-system program uses the interface of debugger which uses two wires. Refer to Chapter 23. Development tools in details about debugger.

## 20.5 Protection area (user program mode)

A user can program flash memory (protection area) of A96L322. The protection area cannot be erased or programmed if any protection area is enabled by the configure option 2. If the protection area is disabled (PAEN ='0'), this area can be erased or programmed.

The user can choose size of protection area can by using configure option 2. For more information about configure option 2, please refer to **Appendix A. Configure option**.

Table 32 introduces protection area size and relative information.

**Table 32. Protection Area Size and its Relative Information**

| Protection area size select | | Size of protection area | Address of protection area |
|---|---|---|---|
| PASS1 | PASS0 | | |
| 0 | 0 | 0.7Kbytes | 0100H – 03FFH |
| 0 | 1 | 1.7Kbytes | 0100H – 07FFH |
| 1 | 0 | 2.7Kbytes | 0100H – 0BFFH |
| 1 | 1 | 3.6Kbytes | 0100H – 0F7FH |

**NOTE**: Please refer to **Appendix A. Configure option**.

## 20.6 Erase mode

The sector erase program procedure in user program mode

1. Page buffer clear (FMCR=0x01).
2. Write '0' to the page buffer.
3. Set flash sector address register (FSADRH/FSADRM/FSADRL).
4. Set flash identification register (FIDR).
5. Check User ID to prevent invalid work[NOTE].
6. Set flash mode control register (FMCR).
7. Erase verify

**NOTE:** Please refer to a subsection **20.8 Protection for invalid erase/write** introduced later part in this chapter.

Figure 92 shows example program tip regarding sector erase.

```
        ANL     EO,#0xF8 ;Set DPTR0
        MOV     FMCR,#0x01      ;page buffer clear
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.

        MOV     A,#0
        MOV     R0,#SectorSize  ;Sector size of Device
        MOV     DPH,#0x80       ;Page Buffer Address is 8000H
        MOV     DPL,#0

    Pgbuf_clr:
        MOVX    @DPTR,A
        INC     DPTR
        DJNZ    R0,Pgbuf_clr    ;Write '0' to all page buffer



        MOV     FSADRH,#SAH     ;Sector Address High Byte.
        MOV     FSADRM,#SAM     ;Sector Address Middle Byte
        MOV     FSADRL,#SAL     ;Sector Address Low Byte
        MOV     FIDR,#0xA5      ;Identification value

        MOV     A,#ID_DATA_1    ;Check the UserID(written by user)
        CJNE    A,UserID1,No_WriteErase;This routine for UserID must be needed.
        MOV     A,#ID_DATA_2
        CJNE    A,UserID2,No_WriteErase

        MOV     FMCR,#0x02      ;Start flash erase mode
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.

        LJMP    Erase_verify
                ---
    No_WriteErase:
        MOV     FIDR,#00H
        MOV     UserID1,#00H
        MOV     UserID2,#00H
                ---
    Erase_verify:
                ---
    Verify_error:
                ---
```

**Figure 92. Program Tip: Sector Erase**

## 20.7 Write mode

The sector Write program procedure in user program mode

1. Page buffer clear (FMCR=0x01)
2. Write data to page buffer
3. Set flash sector address register (FSADRH/FSADRM/FSADRL).
4. Set flash identification register (FIDR).
5. Check the UserID for to prevent the invalid work NOTE1.
6. Set flash mode control register (FMCR).
7. Write verify

**NOTES**

1. Please refer to "20.8. Protection for Invalid Erase/Write".
2. All data of the sector should be "00H" before writing data to a sector

Figure 93 shows example program tip regarding sector write.

```
        ANL     EO,#0xF8;Set DPTR0
        MOV     FMCR,#0x01      ;page buffer clear
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.

        MOV     A,#0
        MOV     R0,#SectorSize  ;Sector size of Device
        MOV     DPH,#0x80       ;Page Buffer Address is 8000H
        MOV     DPL,#0

    Pgbuf_WR:   MOVX    @DPTR,A
        INC     A
        INC     DPTR
        DJNZ    R0,Pgbuf_WR     ;Write data to all page buffer

        MOV     FSADRH,#SAH     ;Sector Address High Byte.
        MOV     FSADRM,#SAM     ;Sector Address Middle Byte
        MOV     FSADRL,#SAL     ;Sector Address Low Byte
        MOV     FIDR,#0xA5      ;Identification value

        MOV     A,#ID_DATA_1    ;Check the UserID(written by user)
        CJNE    A,UserID1,No_WriteErase;This routine for UserID must be needed.
        MOV     A,#ID_DATA_2
        CJNE    A,UserID2,No_WriteErase

        MOV     FMCR,#0x03      ;Start flash write mode
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.

        LJMP    Write_verify
                ---
    No_WriteErase:
        MOV     FIDR,#00H
        MOV     UserID1,#00H
        MOV     UserID2,#00H
                ---
    Write_verify:
                ---
    Verify_error:
                ---
```

**Figure 93. Program Tip: Sector Write**

The Byte Write program procedure in user program mode

1. Page buffer clear (FMCR=0x01)
2. Write data to page buffer
3. Set flash sector address register (FSADRH/FSADRM/FSADRL).
4. Set flash identification register (FIDR).
5. Check the UserID for to prevent the invalid work NOTE1.
6. Set flash mode control register (FMCR).

7. Write verify

**NOTES**:

1. Please refer to "20.8. Protection for invalid erase/write".
2. Data of the address should be "00H" before writing data to an address

Figure 94 shows example program tip regarding byte write.

```
    ANL     EO,#0xF8;Set DPTR0
    MOV     FMCR,#0x01      ;page buffer clear
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

    MOV     A,#5
    MOV     DPH,#0x80
    MOV     DPL,#0
    MOVX    @DPTR,A         ;Write data to page buffer

    MOV     A,#6
    MOV     DPH,#0x80
    MOV     DPL,#0x05
    MOVX    @DPTR,A         ;Write data to page buffer

   MOV      FSADRH,#SAH     ;Sector Address High Byte.
   MOV      FSADRM,#SAM     ;Sector Address Middle Byte
   MOV      FSADRL,#SAL     ;Sector Address Low Byte
   MOV      FIDR,#0xA5      ;Identification value

    MOV     A,#ID_DATA_1    ;Check the UserID(written by user)
    CJNE    A,UserID1,No_WriteErase;This routine for UserID must be needed.
    MOV     A,#ID_DATA_2
    CJNE    A,UserID2,No_WriteErase

    MOV     FMCR,#0x03      ;Start flash write mode
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

    LJMP    Write_verify
            ---
No_WriteErase:
    MOV     FIDR,#00H
    MOV     UserID1,#00H
    MOV     UserID2,#00H
            ---
Write_verify:
            ---
Verify_error:
            ---
```

**Figure 94. Program Tip: Byte Write**

## 20.8 Protection for invalid erase/ write

It needs to be careful when programming flash erase/write operation in code. In addition, it needs preparations for invalid jump to the flash erase/write code occurred by malfunction, noise, and power off.

**NOTE:** For more information about the invalid erase and write operation, please refer to Appendix: Flash protection for invalid erase/write.

Following procedure instructs to protect for invalid erase and write operation:

1. User ID check routine for the flash erase/write code

```
ErWt_rtn:
     ---
     MOV     FIDR,#10100101B          ;ID Code
     MOV     A,#ID_DATA_1             ;Ex) ID_DATA_1: 93H, ID_DATA_2: 85H, ID_DATA_3:
     5AH
     CJNE    A,UserID1,No_WriteErase
     MOV     A,#ID_DATA_2
     CJNE    A,UserID2,No_WriteErase
     MOV     A,#ID_DATA_3
     CJNE    A,UserID3,No_WriteErase
     MOV     FMCR,#0x??               ;0x03 if write, 0x02 if erase
     ---
     ---
     RET

No_WriteErase:
     MOV     FIDR,#00H
     MOV     UserID1,#00H
     MOV     UserID2,#00H
     MOV     UserID3,#00H
     MOV     Flash_flag,#00H
     RET
```

**Figure 95. User ID Check Routine for Flash Erase/Write Code**

With codes in figure 95 invalid flash erase/write can be avoided.

2. It is important the location where the UserID1/2/3 will be written. The invalid flash erase/write problem will remain if the UserID1/2/3 is written at the above line of the instruction "MOV FIDR,#10100101B". Therefore, it is recommended to write the UserID1/2/3 in different routine after returning.
   Figure 96 shows example code regarding the recommendation.

```
Decide_ErWt:
     ---
     MOV     Flash_flag1,#38H  ;Random value for example, in case of erase/write needs
     MOV     FSADRL,#20H       ;Here 20H is example,
     MOV     Flash_flag2,#75H
     RET
```

Figure 96. User ID Check Routine for Flash Erase/Write Code

3. The flash sector address (FSADRH/FSADRM/FSADRL) must always keep the address of the flash which is used for data area. For example, The FSADRH/FSADRM is always 0x00/0x0f" if 0x0f00 to 0x0fff is used for data.

4. Overview of main

```
     ---
     CALL    Work1
     CALL    Decide_ErWt
     CALL    Work2
     CALL    ID_write
     CALL    Work3
     CALL    Flash_erase
     CALL    Flash_write
     ---
     ---
     ---

ID_wire:
     MOV     A,#38H
     CJNE    A,Flash_flag1,No_write_ID
     MOV     A,#75H
     CJNE    A,Flash_flag2,No_write_ID
     MOV     UserID1,#ID_DATA_1        ;Write Uiser ID1
     MOV     A,#38H
     CJNE    A,Flash_flag1,No_write_ID
     MOV     A,#75H
     CJNE    A,Flash_flag2,No_write_ID
     MOV     UserID2,#ID_DATA_2        ;Write Uiser ID2
     MOV     A,#38H
     CJNE    A,Flash_flag1,No_write_ID
     MOV     A,#75H
     CJNE    A,Flash_flag2,No_write_ID
     MOV     UserID3,#ID_DATA_3        ;Write Uiser ID3
     RET

No_write_ID:
     MOV     UserID1,#00H
     MOV     UserID2,#00H
     MOV     UserID3,#00H
     RET
```

Figure 97. Overview of Main

### 20.8.1 Protection flow of invalid erase/write

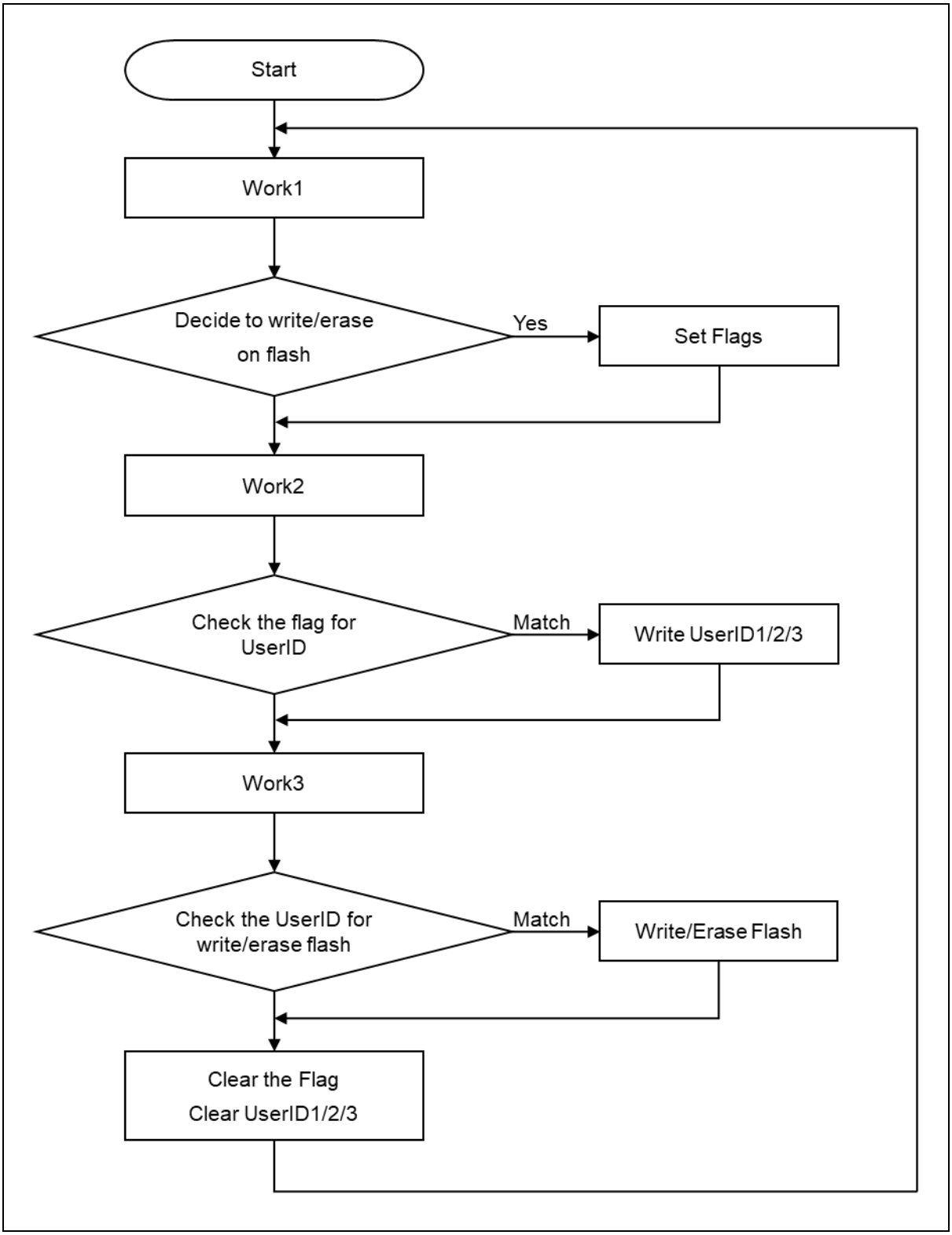



Figure 98. Protection Flow of Invalid Erase/ Write

## 20.9 Read mode

The Reading program procedure in user program mode is shown in the followings:

- Load received data from flash memory on MOVC instruction by indirectly addressing mode.

```
    MOV     A,#0
    MOV     DPH,#0x0F
    MOV     DPL,#0xA0               ;flash memory address

    MOVC    A,@A+DPTR               ;read data from flash memory
```

**Figure 99. Program Tip: Reading**

## 20.10 Code write protection mode

The code write protection program procedure in user program mode

1. Set flash identification register (FIDR).
2. Check the UserID for to prevent the invalid work NOTE.
3. Set flash mode control register (FMCR).

**NOTE:** Please refer to 20.8 Protection for Invalid Erase/Write.

```
    MOV     FIDR,#0xA5      ;Identification value

    MOV     A,#ID_DATA_1    ;Check the UserID(written by user)
    CJNE    A,UserID1,No_WriteErase;This routine for UserID must be needed.
    MOV     A,#ID_DATA_2
    CJNE    A,UserID2,No_WriteErase

    MOV     FMCR,#0x04      ;Start flash Code Write Protection mode
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

No_WriteErase:
    MOV     FIDR,#00H
    MOV     UserID1,#00H
    MOV     UserID2,#00H
    ---
```

**Figure 100. Program Tip: Code Write Protection**

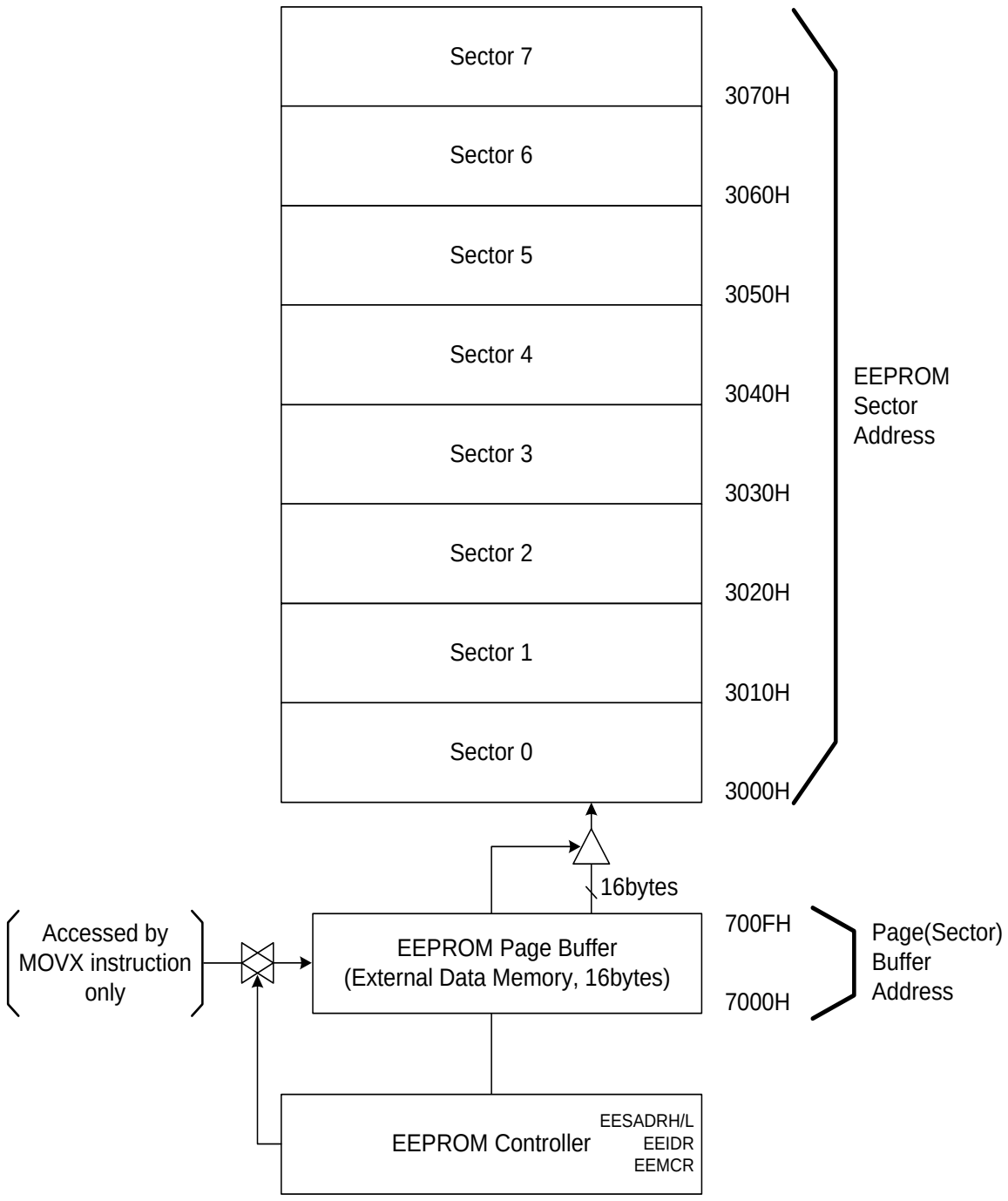
# 21 EEPROM memory

The A96L322 includes EEPROM memory of 128bytes. It can be written, erased, and overwritten. The EEPROM memory can be read by 'MOVX' instruction.

- EEPROM Size: 128bytes
- Single power supply program and erase
- Command interface for fast program and erase operation
- Up to 100,000 program/erase cycles at typical voltage and temperature for memory

The write/erase cycles of the internal EEPROM can be increased significantly if it is divided into smaller and used in turn. If 128bytes are divided into 4 areas with 32bytes and the each area from 1st to 4th is used up to 100,000 cycles, the total erase/write is for 400,000 cycles.

Figure 101 describes the relationship between EEPROM page buffer, EEPROM controller, and EEPROM sector addresses.




**NOTES**:

1. The EEPROM supports byte and sector erase mode.
2. The EEPROM supports byte and sector write mode.
3. The EEPROM is in extended data memory area.
4. If Idle and Stop are executed on EEPROM erase/write, the erase/write is forcibly terminated. So before entering Idle and Stop mode, the EEPROM busy bit (EEMBUSY) should be checked whether the bit is "1b"or not. If the bit is "1b", do not enter Idle and Stop mode.

**Figure 101. EEPROM Structure**

## 21.1 Register map

Table 33. EEPROM Register Map

| Name | Address | Direction | Default | Description |
|---|---|---|---|---|
| EESADRH | F3H | R/W | 00H | EEPROM Sector Address High Register |
| EESADRL | F2H | R/W | 00H | EEPROM Sector Address Low Register |
| EEIDR | F4H | R/W | 00H | EEPROM Identification Register |
| EEMCR | F5H | R/W | 00H | EEPROM Mode Control Register |

## 21.2 Register description: EEPROM control and status

### EESADRH (EEPROM Sector Address High Register): F3H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EESADRH7 | EESADRH6 | EESADRH5 | EESADRH4 | EESADRH3 | EESADRH2 | EESADRH1 | EESADRH0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

EESADRH[7:0] EEPROM Sector Address High

### EESADRL (EEPROM Sector Address Low Register): F2H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EESADRL7 | EESADRL6 | EESADRL5 | EESADRL4 | – | – | – | – |
| R/W | R/W | R/W | R/W | – | – | – | – |

Initial value: 00H

EESADRL[7:4] Flash Sector Address Low

### EEIDR (EEPROM Identification Register): F4H

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EEIDR7 | EEIDR6 | EEIDR5 | EEIDR4 | EEIDR3 | EEIDR2 | EEIDR1 | EEIDR0 |
| R/W | R/W | R/W | R/W | R/W | R/W | R/W | R/W |

Initial value: 00H

EEIDR[7:0] EEPROM Identification
Others No identification value
01101001 Identification value for a EEPROM mode
(These bits are automatically cleared to logic '00H' immediately after one time operation except "EEPROM page buffer reset mode")

**EEMCR (EEPROM Mode Control Register): F5H**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| EEMBUSY | – | – | – | – | EEMCR2 | EEMCR1 | EEMCR0 |
| R | – | – | – | – | R/W | R/W | R/W |

Initial value: 00H

EEMBUSY EEPROM busy bit.
0 No effect when "1" is written
1 Busy

EEMCR[2:0] EEPROM Mode Control Bits

| EEMCR2 | EEMCR1 | EEMCR0 | Description |
|---|---|---|---|
| 0 | 0 | 1 | Select EEPROM page buffer reset mode and start regardless of the EEIDR value. (Clear all 16bytes to '0') |
| 0 | 1 | 0 | Select EEPROM sector erase mode and start operation when the EEIDR="01101001b" |
| 1 | 0 | 0 | Select EEPROM sector write mode and start operation when the EEIDR=" 01101001b" |
| 1 | 1 | 0 | Select EEPROM bulk erase mode and start operation when the EEIDR=" 01101001b" |

Others Values: No operation
(Automatically cleared to logic '00H' immediately after one time operation)

## 21.3 Erase mode

**The sector erase program procedure in user program mode**

1. Page buffer clear (EEMCR=0x01)
2. Write '0' to page buffer
3. Set EEPROM sector address register (EESADRH/EESADRL).
4. Set EEPROM identification register (EEIDR).
5. Check the UserID for to prevent the invalid work NOTE.
6. Set EEPROM mode control register (EEMCR).
7. Erase verify

**NOTE:** Please refer to 20.8 Protection for Invalid Erase/Write.

```
        ANL     EO,#0xF8 ;Set DPTR0
        MOV     EEMCR,#0x01     ;page buffer clear
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.

        MOV     A,#0
        MOV     R0,#E2P_SectorSize      ;Sector size of EEPROM
        MOV     DPH,#0x70               ;Page Buffer Address is 7000H
        MOV     DPL,#0

    E2P_Pgbuf_clr:
        MOVX    @DPTR,A
        INC     DPTR
        DJNZ    R0,E2P_Pgbuf_clr ;Write '0' to all page buffer

        MOV     EESADRH,#SAH            ;Sector Address High Byte.
        MOV     EESADRL,#SAL            ;Sector Address Low Byte
        MOV     EEIDR,#0x69             ;Identification value

        MOV     A,#E2P_ID_DATA_1 ;Check the UserID(written by user)
        CJNE    A,E2P_UserID1,No_E2PWriteErase ;This routine for UserID must be needed.
        MOV     A,#E2P_ID_DATA_2
        CJNE    A,E2P_UserID2,No_E2PWriteErase

        MOV     EEMCR,#0x02     ;Start EEPROM erase mode
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.
        NOP                     ;Dummy instruction, This instruction must be needed.

        LJMP    E2P_Erase_verify
        ---
    No_E2PWriteErase:
        MOV     EEIDR,#00H
        MOV     E2P_UserID1,#00H
        MOV     E2P_UserID2,#00H
        ---
    E2P_Erase_verify:
        MOV     A,EEMCR
        JNB     ACC.7,E2P_Erase_verify
        ---
    E2P_Verify_error:
        ---
```

**Figure 102. Program Tip: Sector Erase**

## 21.4 Write mode

The sector Write program procedure in user program mode

1. Page buffer clear (EEMCR=0x01)
2. Write data to page buffer
3. Set EEPROM sector address register (EESADRH/EESADRL).
4. Set EEPROM identification register (EEIDR).

5. Check the UserID for to prevent the invalid work NOTE.

6. Set EEPROM mode control register (EEMCR).

7. Write verify

**NOTE:** Data of the address must be "00H" before writing data to an address.

```
    ANL     EO,#0xF8;Set DPTR0
    MOV     EEMCR,#0x01     ;page buffer clear
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

    MOV     A,#0
    MOV     R0,#E2P_SectorSize      ;Sector size of EEPROM
    MOV     DPH,#0x70               ;Page Buffer Address is 7000H
    MOV     DPL,#0

E2P_Pgbuf_WR:
    MOVX    @DPTR,A
    INC     A
    INC     DPTR
    DJNZ    R0,E2P_Pgbuf_WR  ;Write data to all page buffer

    MOV     EESADRH,#SAH            ;Sector Address High Byte.
    MOV     EESADRL,#SAL            ;Sector Address Low Byte
    MOV     EEIDR,#0x69             ;Identification value

    MOV     A,#E2P_ID_DATA_1        ;Check the UserID(written by user)
    CJNE    A,E2P_UserID1,No_E2PWriteErase ;This routine for UserID must be needed.
    MOV     A,#E2P_ID_DATA_2
    CJNE    A,E2P_UserID2,No_E2PWriteErase

    MOV     EEMCR,#0x04     ;Start EEPROM write mode
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

    LJMP    E2P_Write_verify
    ---
No_E2PWriteErase:
    MOV     EEIDR,#00H
    MOV     E2P_UserID1,#00H
    MOV     E2P_UserID2,#00H
    ---
E2P_Write_verify:
    MOV     A,EEMCR
    JNB     ACC.7,E2P_Write_verify
    ---
E2P_Verify_error:
    ---
```

Figure 103. Program Tip: Sector Write

The Byte Write program procedure in user program mode

1. Page buffer clear (EEMCR=0x01)
2. Write data to page buffer
3. Set EEPROM sector address register (EESADRH/EESADRL).
4. Set EEPROM identification register (EEIDR).
5. Check the UserID for to prevent the invalid work NOTE.
6. Set EEPROM mode control register (EEMCR).
7. Write verify

**NOTE:** Data of the address must be "00H" before writing data to an address.

```
    ANL     EO,#0xF8;Set DPTR0
    MOV     EEMCR,#0x01     ;page buffer clear
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

    MOV     A,#5
    MOV     DPH,#0x70
    MOV     DPL,#0
    MOVX    @DPTR,A         ;Write data to page buffer

    MOV     A,#6
    MOV     DPH,#0x70
    MOV     DPL,#0x05
    MOVX    @DPTR,A         ;Write data to page buffer

    MOV     EESADRH,#SAH    ;Sector Address High Byte.
    MOV     EESADRL,#SAL    ;Sector Address Low Byte
    MOV     EEIDR,#0x69     ;Identification value

    MOV     A,#E2P_ID_DATA_1 ;Check the UserID(written by user)
    CJNE    A,E2P_UserID1,No_E2PWriteErase ;This routine for UserID must be needed.
    MOV     A,#E2P_ID_DATA_2
    CJNE    A,E2P_UserID2,No_E2PWriteErase

    MOV     EEMCR,#0x04     ;Start EEPROM write mode
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.
    NOP                     ;Dummy instruction, This instruction must be needed.

    LJMP    E2P_Write_verify
    ---
No_E2PWriteErase:
    MOV     EEIDR,#00H
    MOV     E2P_UserID1,#00H
    MOV     E2P_UserID2,#00H
    ---
E2P_Write_verify:
    MOV     A,EEMCR
    JNB     ACC.7,E2P_Write_verify
    ---
E2P_Verify_error:
    ---
```

Figure 104. Program Tip: Byte Write

## 21.5 Read mode

The Reading program procedure in user program mode

- Load the received data from EEPROM memory on MOVX instruction by indirectly addressing mode.

```
MOV     DPH,#0x30
MOV     DPL,#0x10               ;EEPROM memory address

MOVX    A,@DPTR                 ;read data from EEPROM memory
```

**Figure 105. Program Tip: Reading**

# 22 Electrical characteristics

## 22.1 Absolute maximum ratings

**Table 34. Absolute Maximum Ratings**

| Parameter | Symbol | Rating | Unit | Remark |
|---|---|---|---|---|
| Supply Voltage | VDD | -0.3 ~ +4.0 | V | – |
| Normal Voltage Pin | VI | -0.3 ~ VDD+0.3 | V | Voltage on any pin with respect to VSS |
| | VO | -0.3 ~ VDD+0.3 | V | |
| | IOH | -10 | mA | Maximum current output sourced by ($I_{OH}$ per I/O pin) |
| | ∑IOH | -80 | mA | Maximum current ($\sum I_{OH}$) |
| | IOL | 60 | mA | Maximum current sunk by ($I_{OL}$ per I/O pin) |
| | ∑IOL | 120 | mA | Maximum current ($\sum I_{OL}$) |
| Total Power Dissipation | PT | 600 | mW | – |
| Storage Temperature | TSTG | -65 ~ +150 | °C | – |

**Caution**
Stresses beyond those listed under 'Absolute Maximum Ratings' may cause permanent damage to the device. This is a stress rating only and functional operation of the device at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure above maximum rating conditions for extended periods may affect device reliability.

## 22.2 Operating conditions

The device must be used in operating conditions that comply with the parameters in table 35.

**Table 35. Recommended Operating Conditions**

(TA=-40℃ to +85℃)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Operating voltage | VDD | fx=0.125 to 1.0MHz, internal RC | 2.0 | — | 3.6 | V |
| Operating temperature | $T_{OPR}$ | VDD=2.0 to 3.6V | -40 | — | 85 | ℃ |

## 22.3 ADC characteristics

**Table 36. ADC Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| Resolution | – | – | | – | 10 | – | bit |
| Integral Linear Error | ILE | AVREF= 2.7V to 3.6V fx=1MHz | | – | – | ±3 | LSB |
| Differential Linearity Error | DLE | | | – | – | ±1 | |
| Top Offset Error | TOE | | | – | – | ±5 | |
| Zero Offset Error | ZOE | | | – | – | ±5 | |
| Conversion Time | $t_{CON}$ | AVREF= 2.7V to 3.6V | | 28 | – | – | us |
| Analog Input Voltage | $V_{AN}$ | – | | VSS | – | AVREF | V |
| Analog Reference Voltage | AVREF | NOTE3 | | 2.0 | – | VDD | |
| Sample/Hold Time | $t_{SH}$ | – | | 2 | – | – | us |
| A/DC Input Leakage Current | IAN | AVREF=3.3V | | – | – | 2 | uA |
| A/DC Current | $I_{ADC}$ | Enable | VDD=3.3 V | – | 300 | 500 | uA |
| | | Disable | | – | – | 0.1 | uA |

**NOTES**:

1. Zero offset error is the difference between 0000000000 and the converted output for zero input voltage (VSS).
2. Top offset error is the difference between 1111111111 and the converted output for top input voltage (AVREF).
3. If AVREF is less than 2.7V, the resolution degrades by 1-bit whenever AVREF drops 0.1V. (@ADCLK = 0.5MHz, under 2.7V resolution has no test.)

## 22.4 Power on Reset

**Table 37. Power on Reset Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| RESET Release Level | $V_{POR}$ | – | – | 1.4 | – | V |
| VDD Voltage Rising Time | $t_R$ | 0.2V to 2.0V | 0.05 | – | 100 | V/ms |
| POR Current | $I_{POR}$ | – | – | 0.2 | – | uA |

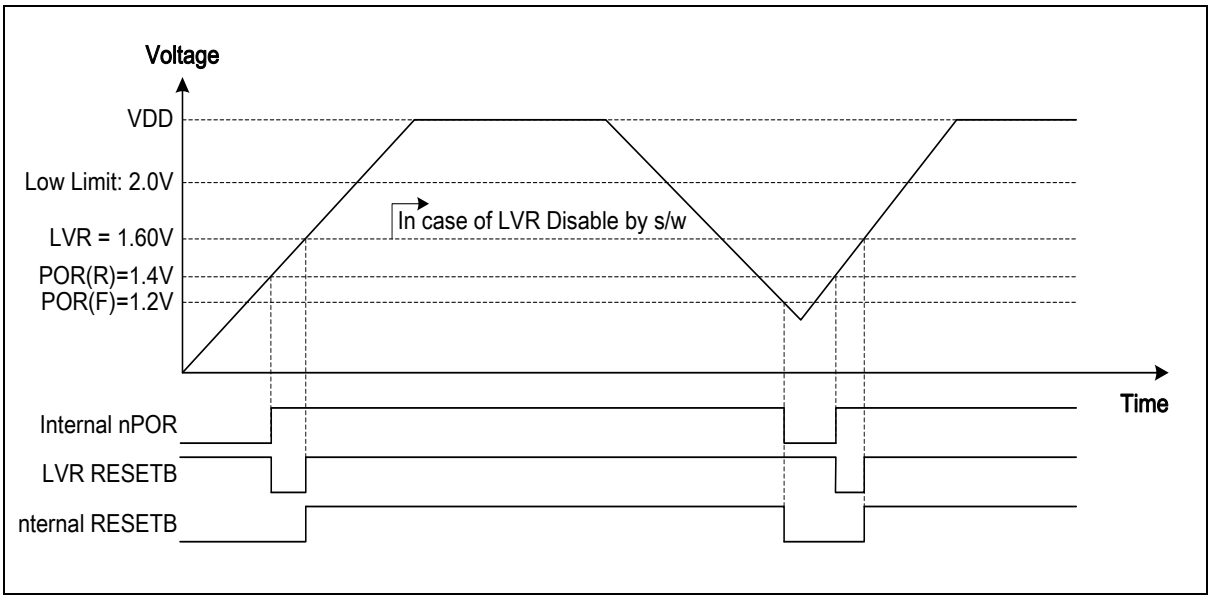



Figure 106. Power-On Reset Timing

## 22.5 Low voltage reset characteristics

Table 38. LVR Characteristics

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| Detection Level | $V_{LVR}$ | The LVR can select all levels. | | – | 1.60 | 1.89 | V |
| | | | | 2.05 | 2.20 | 2.35 | |
| | | | | 2.50 | 2.70 | 2.90 | |
| Hysteresis | △V | – | | – | 10 | 100 | mV |
| Minimum Pulse Width | $t_{LW}$ | – | | 100 | – | – | us |
| LVR Current | $I_{LVR}$ | Enable | VDD= 3V, RUN mode | – | 4.0 | 8.0 | uA |
| | | Disable | | – | – | 0.1 | |

## 22.6 Operational amplifier 0/1 characteristics

**Table 39. Operational Amplifier 0/1 Characteristic**

($T_A$=-40℃ to +85℃, VDD=2.7V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| Input Offset Voltage | $V_{OF}$ | VDD=3.3V | | – | ±10 | ±100 | uV |
| Input Offset Current | $I_{OF}$ | VDD=3.3V, VCM=0V | | – | 15 | 50 | pA |
| Common-mode Rejection Ratio | CMRR | VDD=3.3V, DC VCM=0V to VDD-1.2V | | 80 | 100 | – | dB |
| Power Supply Rejection Ration | PSRR | VDD=3.3V | | 80 | 100 | – | |
| Open Loop Voltage Gain | – | VDD=3.3V | | 100 | 120 | – | dB |
| Gain Error | ERR | VDD=3.3V, VIN ≥ 0.1V, x10 VIN<(Input x Gain) | | – | – | 1 | % |
| Input Common-mode Voltage Range | $V_{IN}$ | VDD=3.3V | | 0 | – | VDD-1.2 | V |
| Output Voltage Range | $V_O$ | VDD=3.3V, RL=10KΩ | | VSS+0.1 | – | VDD-0.1 | V |
| Output Short Circuit Current | ISCH | VDD=3.3V, Absolute | | – | 12 | – | mA |
| | ISCL | | | – | 12 | – | |
| Gain Bandwidth | $f_{GB}$ | VDD=3.3V | | 1 | 2 | – | MHz |
| Voltage Follower Pulse Response | $T_{AR}$ | VDD=3.3V, Small Signal | | – | 5 | 10 | us |
| OP-AMP 0/1 Total Current | $I_{AMP}$ | Enable | VDD=3.3V, No Load | – | 150 | 220 | uA |
| | | Disable | | – | – | 0.1 | |
| Enable Time of AMP0/1 | $t_{ON}$ | VDD=3.3V, Gain=x20/x30, RL=10KΩ with 50pF | | – | – | 150 | us |
| Input Noise Voltage Density | $e_{ni}$ | Input Referred f=1Hz | | – | 0.1 | – | uV/rtHz |
| | | Input Referred f=1KHz | | – | 50 | – | nV/rtHz |
| Slew Rate | $S_R$ | VDD=3.3V, RL=10K, CL=50pF | | – | 0.7 | – | V/us |
| Phase Margin | $P_M$ | VDD=3.3V, RL=10K, CL=50pF | | – | 60 | – | Degrees |
| Chopping Clock | $f_{CHOP}$ | – | | 125 | – | 500 | KHz |

## 22.7 Internal RC oscillator characteristics

**Table 40. Internal RC Oscillator Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Frequency | fIRC | VDD=3.3V | – | 1 | – | MHz |
| Tolerance | – | $T_A$ = -10°C to +40°C, with user (S/W) trim. | – | – | ±1.0 | % |
| | | $T_A$ = -10°C to +40°C | | | ±2.0 | |
| | | $T_A$ = -40°C to +85°C | | | ±3.0 | |
| Clock Duty Ratio | $T_{OD}$ | – | 40 | 50 | 60 | % |
| Stabilization Time | $T_{FS}$ | – | – | – | 100 | us |
| IRC Current | $I_{IRC}$ | Enable | – | 15 | – | uA |
| | | Disable | – | – | 0.1 | uA |

## 22.8 Internal watchdog timer RC oscillator characteristics

**Table 41. Internal WDTRC Oscillator Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Frequency | $f_{WDTRC}$ | – | 0.5 | 1 | 2 | KHz |
| Stabilization Time | $t_{WDTS}$ | – | – | – | 1 | ms |
| WDTRC Current | $I_{WDTRC}$ | Enable | – | 1 | – | uA |
| | | Disable | – | – | 0.1 | |

## 22.9 DC characteristics

**Table 42. DC Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V, $f_{IRC}$=1MHz)

| Parameter | Symbol | Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| Input high voltage | $V_{IH1}$ | P00, P01, P06, P07, P1, RESETB | | 0.8VDD | – | VDD | V |
| | $V_{IH2}$ | All input pins except VIH1 | | 0.7VDD | – | VDD | |
| Input low voltage | $V_{IL1}$ | P00, P01, P06, P07, P1, RESETB | | – | – | 0.2VDD | V |
| | $V_{IL2}$ | All input pins except VIL1 | | – | – | 0.3VDD | |
| Output high voltage | $V_{OH}$ | VDD=3.3V, IOH = –6mA; | | VDD-1.0 | – | – | V |
| Output low voltage | $V_{OL}$ | All output ports | | – | – | 1.0 | V |
| Input high leakage current | $I_{IH}$ | VDD=3.3V, IOL = 8mA; | | – | – | 1.0 | uA |
| Input low leakage current | $I_{IL}$ | All output ports | | -1.0 | – | – | uA |
| Pull-up resistor | $R_{PU1}$ | VI=0V, $T_A$=25℃, All Input ports | VDD= 3.0V | 50 | 100 | 200 | KΩ |
| | $R_{PU2}$ | VI=0V, $T_A$=25℃, RESETB | VDD= 3.0V | 300 | 500 | 700 | KΩ |
| Supply current | $I_{DD1}$ (RUN) | $f_{IRC}$ = 1MHz | VDD= 3V±10 % | – | 240 | 320 | uA |
| | | $f_{IRC}$ = 0.5MHz | | – | 150 | 200 | |
| | $I_{DD2}$ (IDLE) | $f_{IRC}$ = 1MHz | VDD= 3V±10 % | – | 100 | 150 | uA |
| | | $f_{IRC}$ = 0.5MHz | | – | 90 | 140 | |
| | $I_{DD5}$ | STOP, VDD= 3V±10%, $T_A$= 25℃ | | – | 0.5 | 3.0 | uA |

**NOTES**:

1. Where the fx is the selected system clock, the $f_{IRC}$ is an internal RC oscillator.
2. All supply current items don't include the current of an internal Watch-dog timer RC (WDTRC) oscillator and a peripheral block.
3. All supply current include the current of the power-on reset (POR) block.

## 22.10 Constant sink current electrical characteristics

**Table 43. Constant Sink Current Electrical Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|---|
| Constant sink current | $I_{CS}$ | VDD=3V<br>$V_{ICS}$=1.5V<br>$T_A$=25℃ | ICSDR[3:0] = 0 | -7% | 49 | +7% | mA |
| | | | ICSDR[3:0] = 1 | -7% | 65 | +7% | |
| | | | ICSDR[3:0] = 2 | -7% | 80 | +7% | |
| | | | ICSDR[3:0] = 3 | -7% | 96 | +7% | |
| | | | ICSDR[3:0] = 4 | -7% | 111 | +7% | |
| | | | ICSDR[3:0] = 5 | -7% | 127 | +7% | |
| | | | ICSDR[3:0] = 6 | -7% | 142 | +7% | |
| | | | ICSDR[3:0] = 7 | -7% | 158 | +7% | |
| | | | ICSDR[3:0] = 8 | -7% | 173 | +7% | |
| | | | ICSDR[3:0] = 9 | -7% | 188 | +7% | |
| | | | ICSDR[3:0] = 10 | -7% | 203 | +7% | |
| | | | ICSDR[3:0] = 11 | -7% | 218 | +7% | |
| | | | ICSDR[3:0] = 12 | -7% | 232 | +7% | |
| | | | ICSDR[3:0] = 13 | -7% | 246 | +7% | |
| | | | ICSDR[3:0] = 14 | -7% | 260 | +7% | |
| | | | ICSDR[3:0] = 15 | -7% | 274 | +7% | |
| | | VDD=3V<br>$V_{ICS}$=1V to 2.0V<br>$T_A$= -40 to +85°C | ICSDR[3:0] = n<br>n: 0 to 15 | -15% | Typ. | +15% | |
| | | VDD=2.7V to 3.6V<br>$V_{ICS}$=1V to VDD-1.0V<br>$T_A$= -40 to +85°C | ICSDR[3:0] = n<br>n: 0 to 15 | -20% | Typ. | +20% | |

## 22.11 AC characteristics

**Table 44. AC Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| RESETB input low width | $t_{RST}$ | VDD = 3V | 10 | – | – | us |
| Interrupt input high, low width | $t_{IWH}$, $t_{IWL}$ | All interrupt, VDD = 3V | 200 | – | – | ns |
| External counter input high, low pulse width | $t_{ECWH}$, $t_{ECWL}$ | EC0/EC1, VDD = 3V | 200 | – | – | |
| External counter transition time | $t_{REC}$, $t_{FEC}$ | EC0/EC1, VDD = 3V | 20 | – | – | |

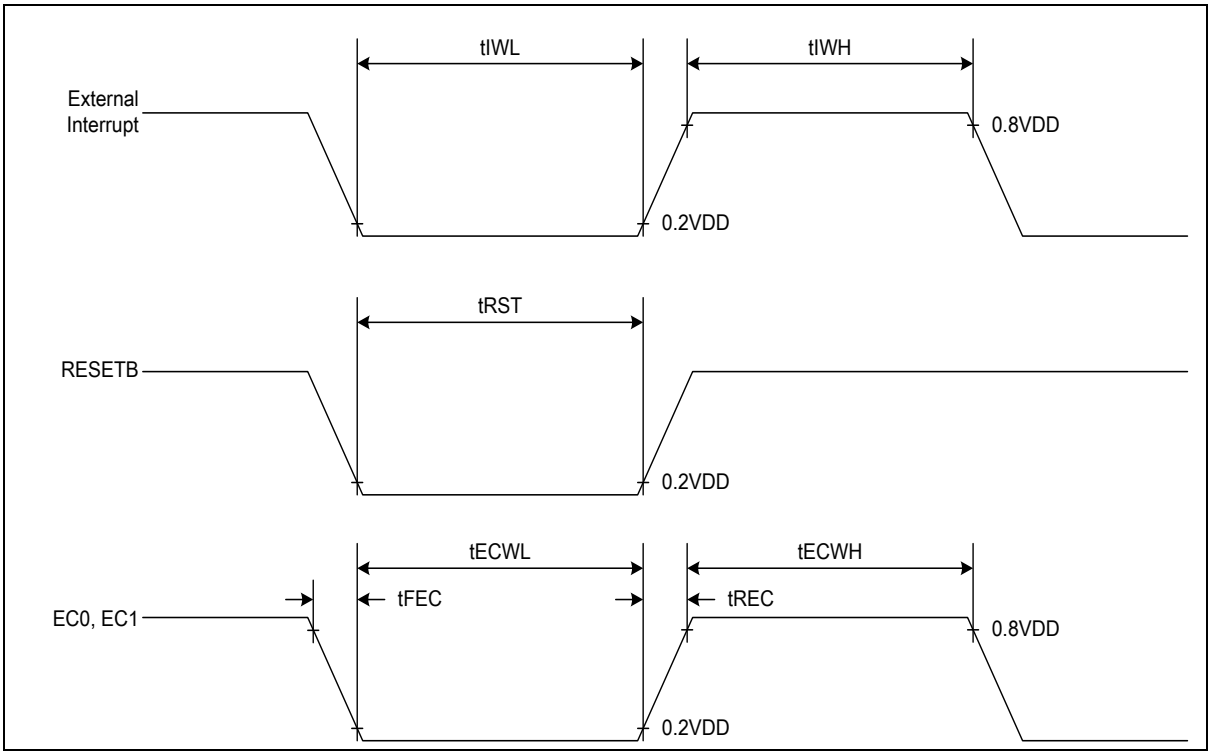



**Figure 107. AC Timing**

## 22.12 SPI characteristics

**Table 45. SPI Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Output clock pulse period | $t_{SCK}$ | Internal SCK source | 2000 | – | – | ns |
| Input clock pulse period | | External SCK source | 2000 | – | – | |
| Output clock high, low pulse width | $t_{SCKH}$ $t_{SCKL}$ | Internal SCK source | 700 | – | – | |
| Input clock high, low pulse width | | External SCK source | 700 | – | – | |
| First output clock delay time | $t_{FOD}$ | Internal/external SCK source | 1000 | – | – | |
| Output clock delay time | $t_{DS}$ | – | – | – | 250 | |
| Input setup time | $t_{DIS}$ | – | 1000 | – | – | |
| Input hold time | $t_{DIH}$ | – | 1000 | – | – | |

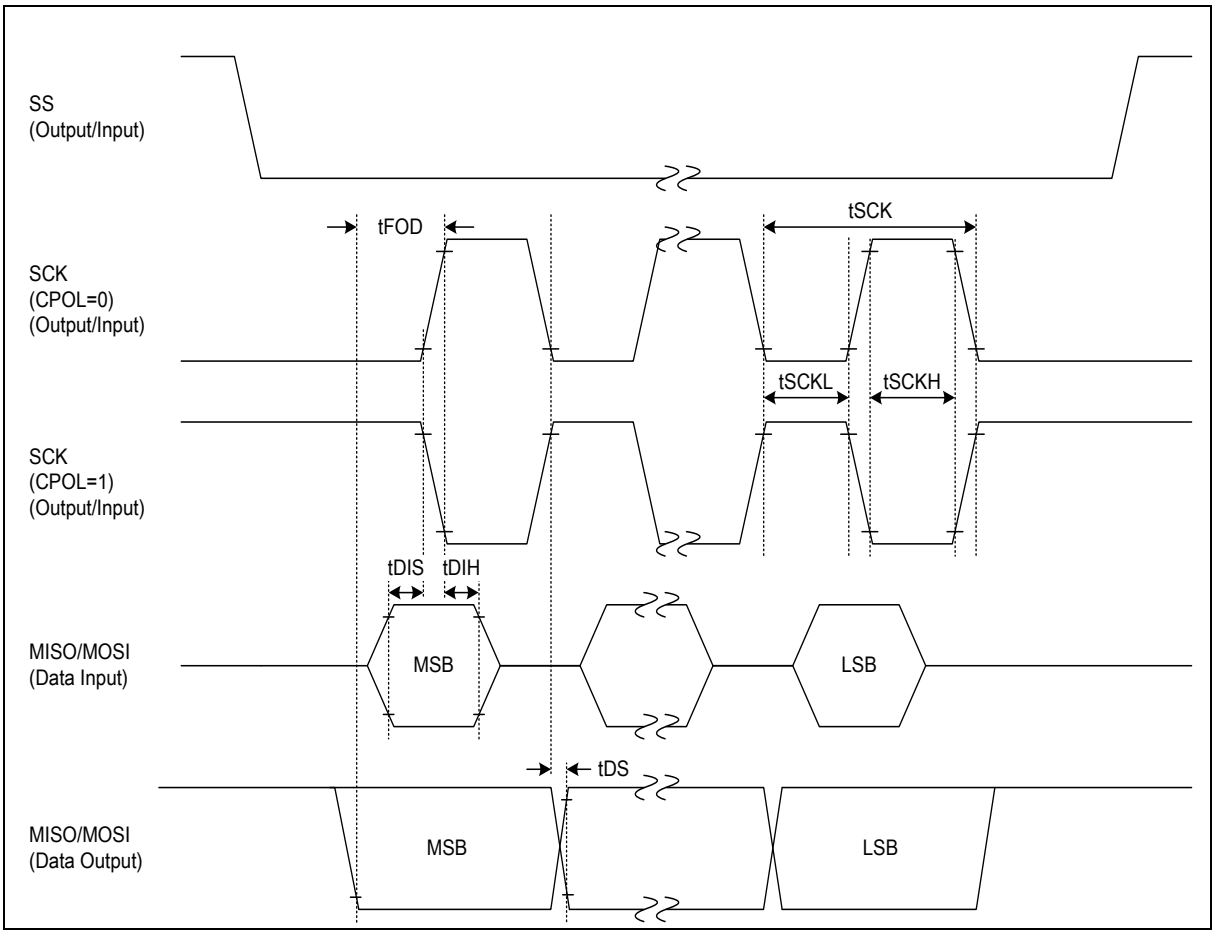



**Figure 108. SPI Timing**

## 22.13 UART timing characteristics

**Table 46. UART Timing Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, $f_{IRC}$=1MHz)

| Parameter | Symbol | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|
| Serial port clock cycle time | $t_{SCK}$ | 13.92 | $t_{CPU}$ x 16 | 18.08 | us |
| Output data setup to clock rising edge | $t_{S1}$ | 6.5 | $t_{CPU}$ x 13 | – | |
| Clock rising edge to input data valid | $t_{S2}$ | – | – | 6.5 | |
| Output data hold after clock rising edge | $t_{H1}$ | $t_{CPU}$ – 0.1 | $t_{CPU}$ | – | |
| Input data hold after clock rising edge | $t_{H2}$ | 0 | – | – | |
| Serial port clock High, Low level width | $t_{HIGH}$, $t_{LOW}$ | 5.5 | $t_{CPU}$ x 8 | 10.5 | |

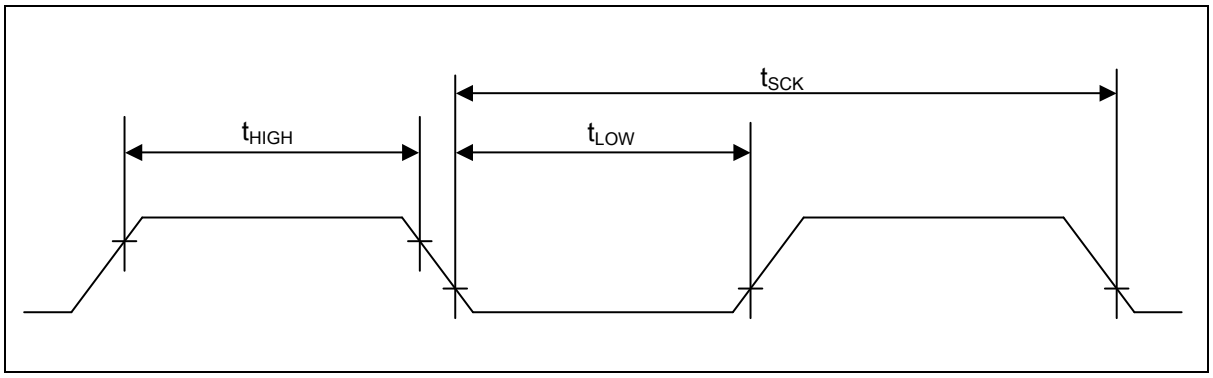



Figure 109. UART Timing Characteristics

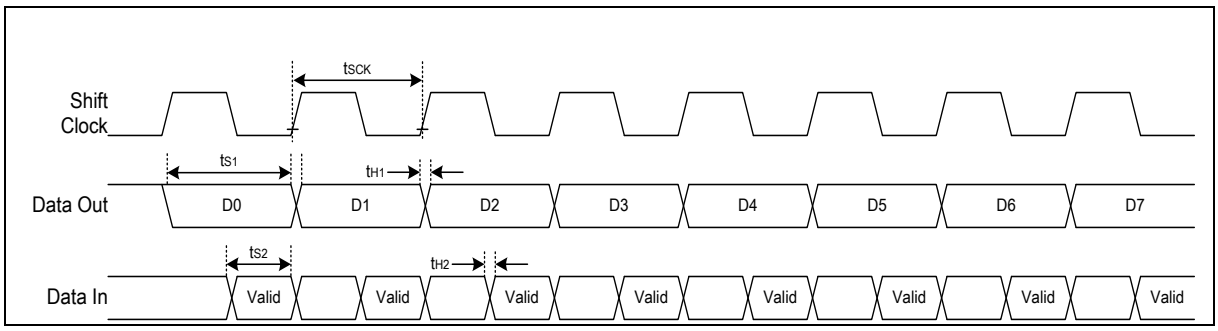


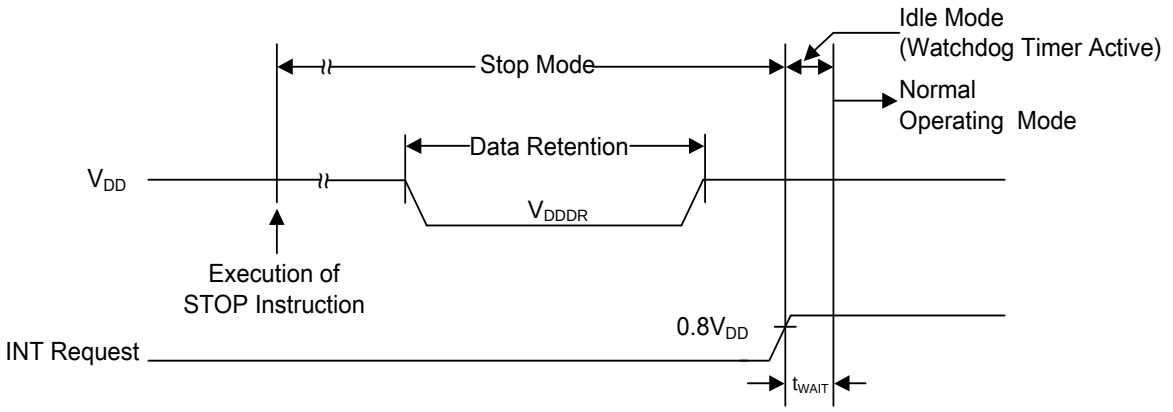

Figure 110. Timing Waveform of UART Module

## 22.14 Data retention voltage in STOP mode

Table 47. Data Retention Voltage in STOP Mode

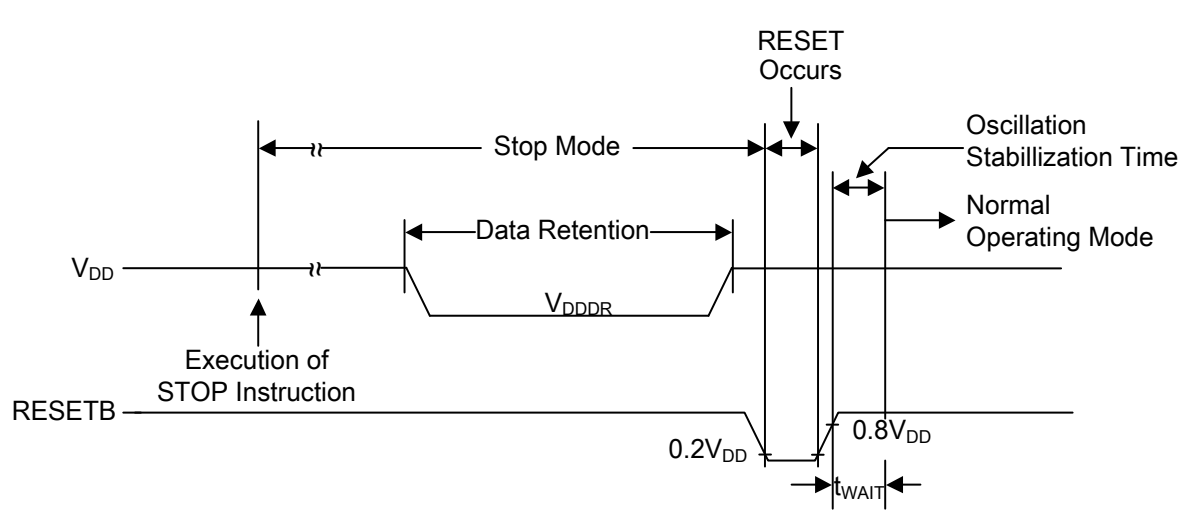
($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Data retention supply voltage | $V_{DDDR}$ | – | 2.0 | – | 3.6 | V |
| Data retention supply current | $I_{DDDR}$ | VDDR= 2.0V<br>($T_A$= 25°C)<br>STOP mode | – | – | 1 | uA |




**NOTE**: $t_{WAIT}$ is the same as (the selected bit overflow of BIT) x 1/ (BIT clock)

**Figure 111. STOP Mode Release Timing when Initiated by an Interrupt**




**NOTE**: $t_{WAIT}$ is the same as (4096 x 4 x $f_{IRC}$) x 1/ (BIT clock) = (16.4 ms at fx=1MHz)

**Figure 112. STOP Mode Release Timing when Initiated by RESETB**

## 22.15 Internal flash characteristics

**Table 48. Internal Flash Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Sector write time | $t_{FSW}$ | – | – | 2.5 | 2.7 | ms |
| Sector erase time | $t_{FSE}$ | – | – | 2.5 | 2.7 | |
| Code write protection time | $t_{FHL}$ | – | – | 2.5 | 2.7 | |
| Page buffer reset time | $t_{FBR}$ | – | – | – | 5 | us |
| Flash programming frequency | $f_{PGM}$ | – | 0.125 | – | – | MHz |
| Endurance of write/erase (sector 0 to 123) | $NF_{WE}$ | Sector erase, byte write | 10,000 | – | – | cycles |
| Endurance of write/erase (sector 124 to 127) | | | 100,000 | — | — | |

## 22.16 Internal EEPROM characteristics

**Table 49. Internal EEPROM Characteristics**

($T_A$=-40℃ to +85℃, VDD=2.0V to 3.6V, VSS=0V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Sector write time | $t_{ESW}$ | – | – | 2.5 | 2.7 | ms |
| Sector erase time | $t_{ESE}$ | – | – | 2.5 | 2.7 | |
| Page buffer reset time | $t_{EBR}$ | – | – | – | 5 | us |
| EEPROM programming frequency | $f_{PGM}$ | – | 0.125 | – | – | MHz |
| Endurance of write/erase | $NE_{WE}$ | Sector erase, byte write | 100,000 | – | – | cycles |

**NOTE**: The write/erase cycles of an internal EEPROM can be increased significantly if it is divided into smaller and used in turn: Ex.) If 128 bytes are divided into 4 areas with 32 bytes and each area from 1st to 4th is used up to 100,000 cycles, the total erase/write is for 400,000 cycles.

## 22.17 Input/output capacitance characteristics

**Table 50. I/O Capacitance Characteristics**

($T_A$=-40℃ to +85℃, VDD=0V)

| Parameter | Symbol | Conditions | Min. | Typ. | Max. | Unit |
|---|---|---|---|---|---|---|
| Input Capacitance | $C_{IN}$ | fx=1MHz unmeasured pins are connected to VSS. | – | – | 10 | pF |
| Output Capacitance | $C_{OUT}$ | | | | | |
| I/O Capacitance | $C_{IO}$ | | | | | |

## 22.18 Recommended circuit and layout

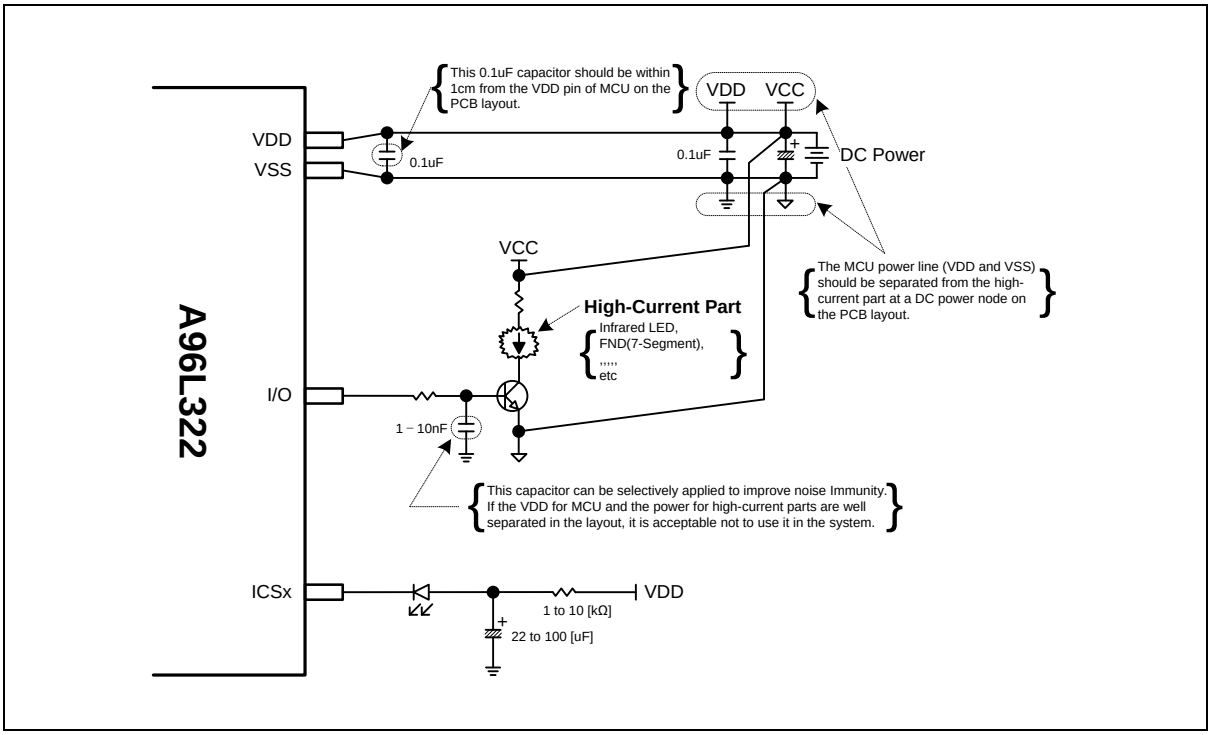



Figure 113. Recommended Circuit and Layout

## 22.19 Typical characteristics

Figures and tables introduced in this chapter can be used only for design guidance, and are not tested or guaranteed. In graphs or tables some data may exceed specified operating range, and can be only for information. The device is guaranteed to operate properly only within the specified range.

The data presented in this chapter is a statistical summary of data collected on units from different lots over a period of time. "Typical" represents the mean of the distribution while "max" or "min" represents (mean + 3σ) and (mean - 3σ) respectively where σ is standard deviation.

# 23 Development tools

This chapter introduces wide range of development tools for A96L322. ABOV offers software tools, debuggers, and programmers to help a user in generating right results to match target applications. ABOV supports entire development ecosystem of the customers.

## 23.1 Compiler

ABOV semiconductor does not provide any compiler for A96L322. However, since A96L322 has Mentor 8051 as its CPU core, you can use all kinds of third party's standard 8051 compiler such as Keil C Compiler. These compilers' output debug information can be integrated with our OCD emulator and debugger. Please visit our website www.abovsemi.com for more information regarding the OCD emulator and debugger.

## 23.2 OCD (On-Chip Debugger) emulator and debugger

The OCD emulator supports ABOV Semiconductor's 8051 series MCU emulation. The OCD uses two wires interfacing between PC and MCU, which is attached to user's system. The OCD can read or change the value of MCU's internal memory and I/O peripherals. In addition, the OCD controls MCU's internal debugging logic. This means OCD controls emulation, step run, monitoring and many more functions regarding debugging.

The OCD debugger program runs underneath MS operating system such as MS-Windows NT/ 2000/ XP/ Vista/ 7/ 8/ 8.1/ 10 (32-bit, 64-bit).

Programming information using the OCD is provided in section 23.5 Circuit design guide later part in this chapter. More detailed information about the OCD, please visit our website www.abovsemi.com and download the debugger S/W and documents.

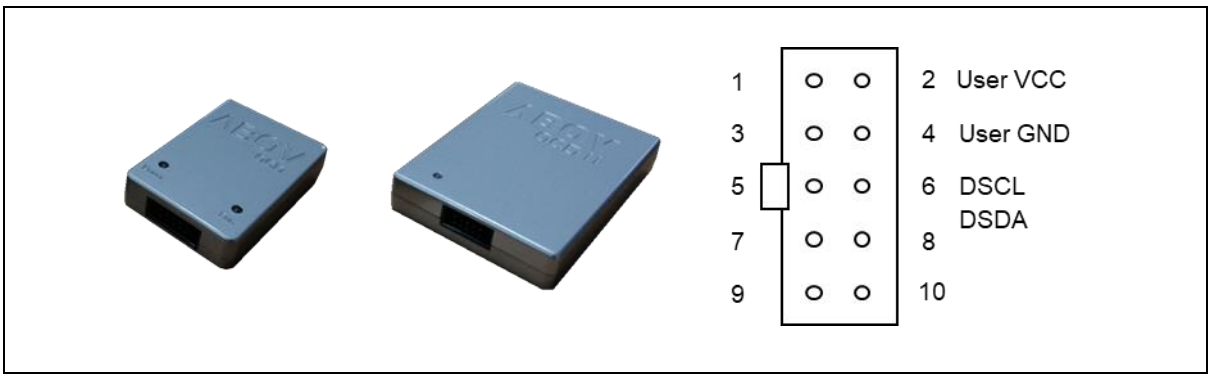



**Figure 114. OCD and Pin Descriptions**

Following is the OCD mode connections:

- DSCL (A96L322 P12 port)
- DSDA (A96L322 P13 port)

## 23.3 Programmer

### E-PGM+

E-PGM+ is a single programmer, and allows a user to program on the device directly.

- Support ABOV / ADAM devices
- 2~5 times faster than S-PGM+
- Main controller : 32-bit MCU @ 72MHz
- Buffer memory : 1MB

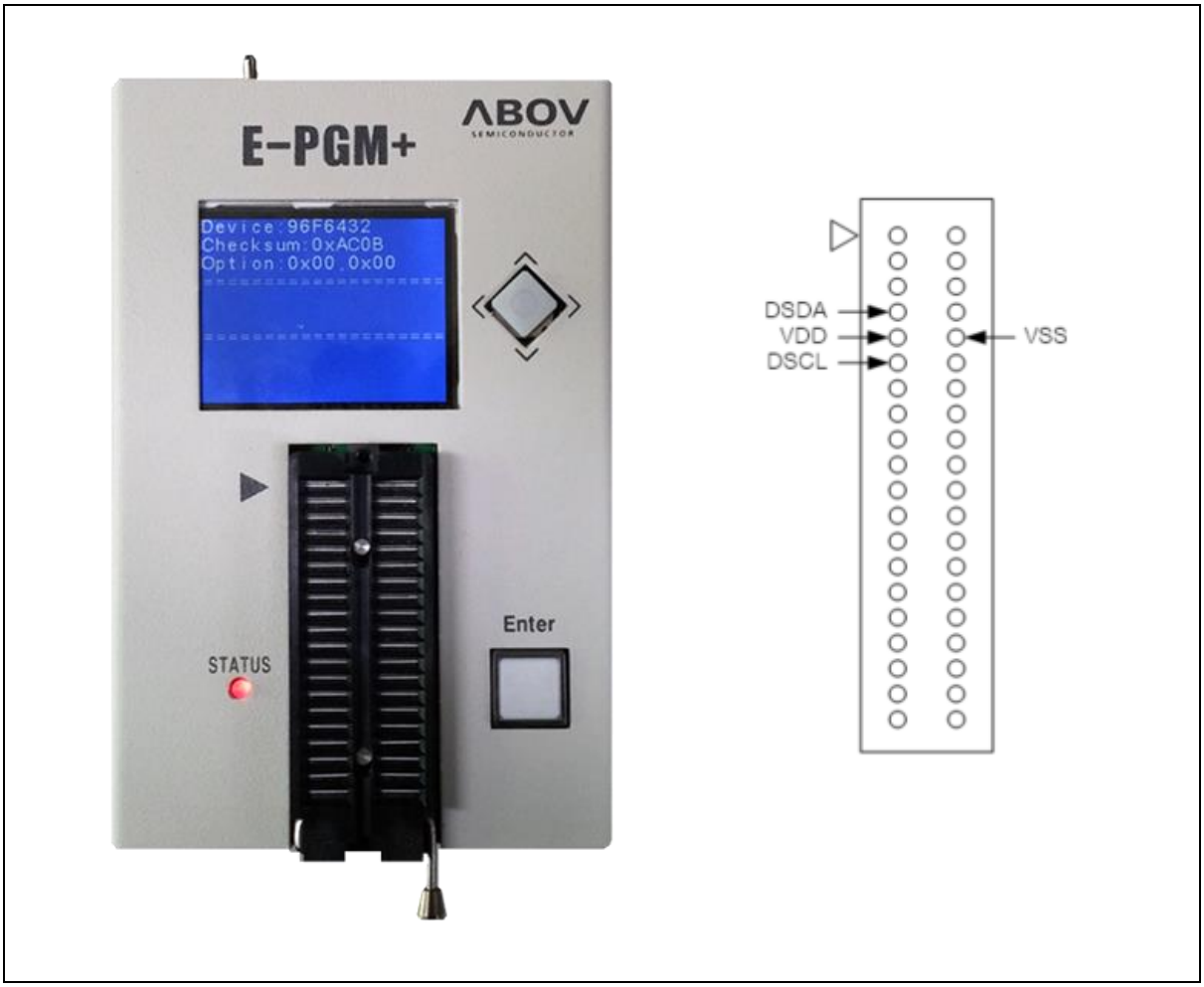



Figure 115. E-PGM+ (Single Writer) and Pin Descriptions

### OCD emulator

OCD emulator allows a user to write code on the device too, since OCD debugger supports ISP (In System Programming). It doesn't require additional H/W, except developer's target system.

Gang programmer

E-Gang4 and E-Gang6 allows a user to program on multiple devices at a time. They run not only in PC controlled mode but also in standalone mode without PC control. USB interface is available and it is easy to connect to the handler.

Table 51. Specification of E-Gang4 and E-Gang6

| Gang programmer | E-Gang4 | E-Gang6 |
|---|---|---|
| Dimension (x, y, h) | 33.5 x 22.5 x35mm | 148.2 x 22.5 x35mm |
| Weight | 2.0kg | 2.8kg |
| Input voltage | DC Adaptor 15V/2A | DC Adaptor 15V/2A |
| Operating temperature | -10 ~ 40℃ | -10 ~ 40℃ |
| Storage temperature | -30 ~ 80℃ | -30 ~ 80℃ |
| Water proof | No | No |

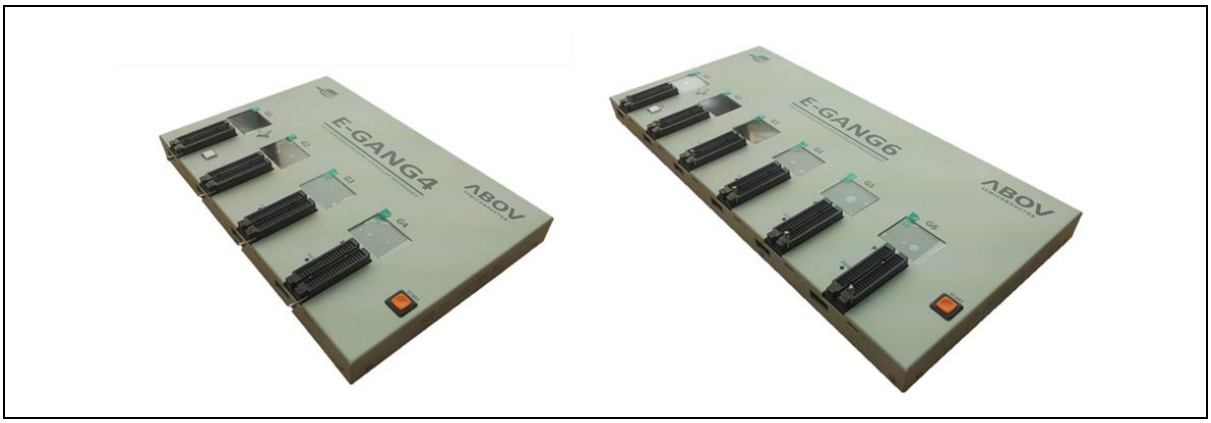



Figure 116. E-Gang4 and E-Gang6 (for Mass Production)

## 23.4 MTP programming

Program memory of A96L322 is an MTP Type. This flash is accessed through four pins such as DSCL, DSDA, VDD, and VSS in serial data format. Table 52 introduces each pin and corresponding I/O status.

Table 52. Pins for MTP Programming

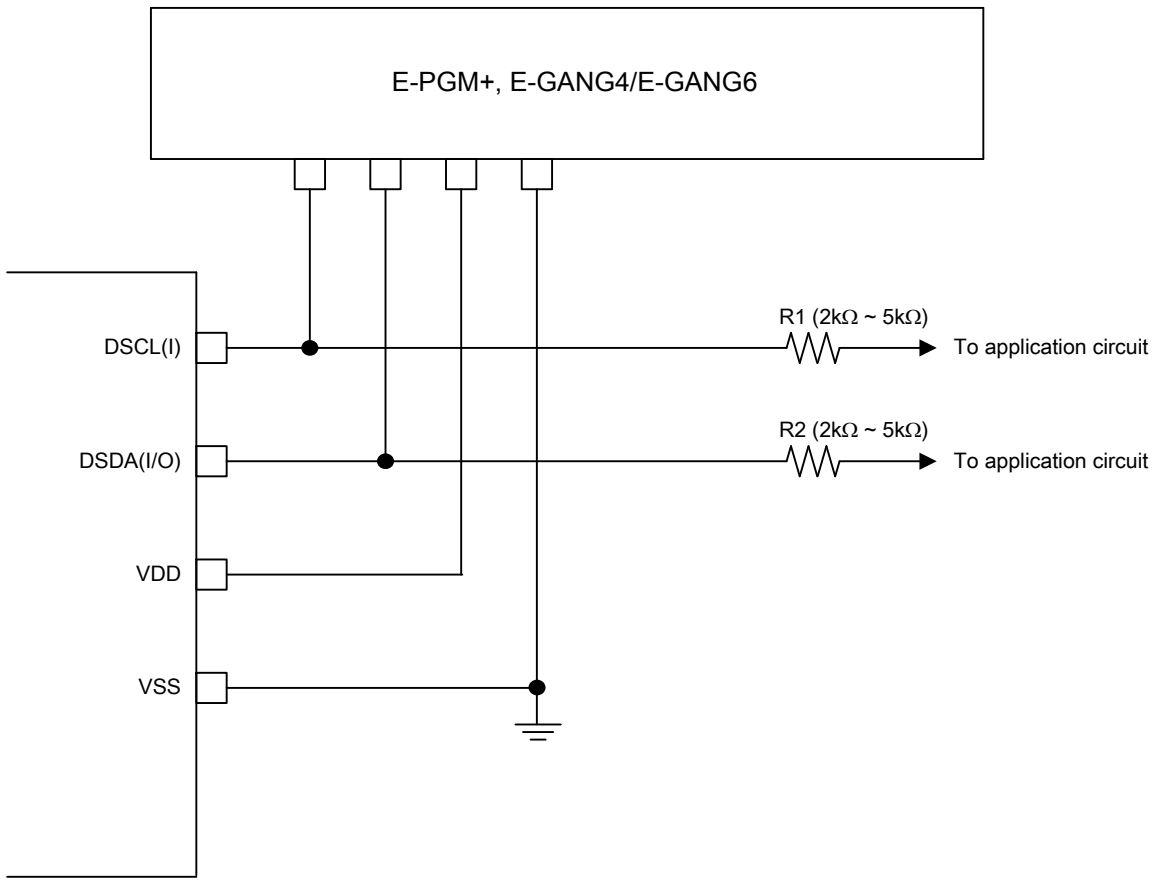
| Pin name | Main chip pin name | During programming | |
|---|---|---|---|
| | | I/O | Description |
| DSCL | P12 | I | Serial clock pin. Input only pin. |
| DSDA | P13 | I/O | Serial data pin. Output port when reading and input port when programming. Can be assigned as input/push-pull output port. |
| VDD, VSS | VDD, VSS | – | Logic power supply pin. |

**On-board programming**

The A96L322 needs only four signal lines including VDD and VSS pins for programming flash with serial protocol. Therefore the on-board programming is possible if the programming signal lines are considered when the PCB of application board is designed.

## 23.5 Circuit design guide

When programming flash memory, the programming tool needs 4 signal lines, DSCL, DSDA, VDD, and VSS. When you design a PCB circuit, you should consider the usage of these 4 signal lines for the on-board programming.




**NOTES**:

1. In on-board programming mode, very high-speed signal will be provided to pin DSCL and DSDA. And it will cause some damages to the application circuits connected to DSCL or DSDA port if the application circuit is designed as high speed response such as relay control circuit. If possible, the I/O configuration of DSDA, DSCL pins had better be set to input mode.
2. The value of R1 and R2 is recommended value. It varies with circuit of system.

**Figure 117. PCB Design Guide for On-Board Programming**

### 23.5.1 On-Chip Debug system

Detail descriptions for programming via the OCD interface can be found in the following figures. Table 53 introduces features of OCD and figure 118 shows a block diagram of the OCD interface and the On-chip Debug system.

**Table 53. Features of OCD**

| | |
|---|---|
| **Two wire external interface** | • 1 for serial clock input<br>• 1 for bi-directional serial data bus |
| **Debugger accesses** | • All internal peripherals<br>• Internal data RAM<br>• Program Counter<br>• Flash memory and data EEPROM memory |
| **Extensive On-Chip Debugging supports for Break Conditions** | • Break instruction<br>• Single step break<br>• Program memory break points on single address<br>• Programming of Flash, EEPROM, Fuses, and Lock bits through the two-wire interface<br>• On-Chip Debugging supported by Dr. Choice® |
| **Operating frequency** | The maximum frequency of a target MCU. |

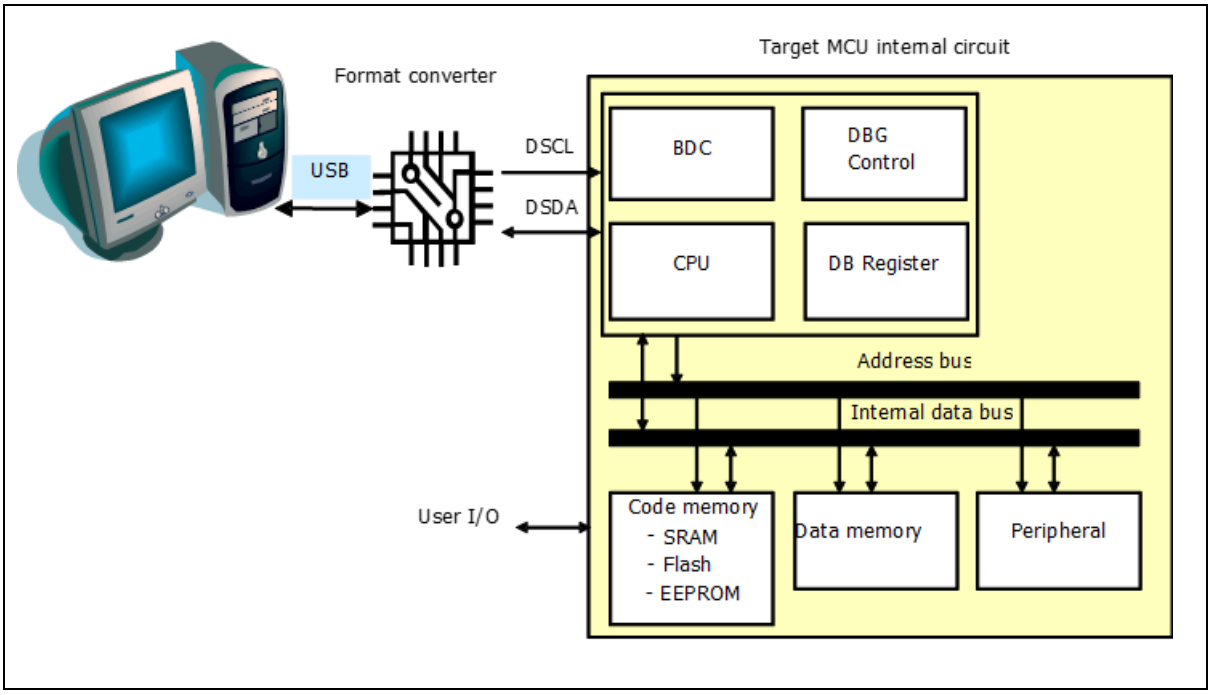



**Figure 118. On-Chip Debugging System in Block Diagram**

### 23.5.2 Two-pin external interface

**Basic transmission packet**

- 10-bit packet transmission using two-pin interface.
- 1-packet consists of 8-bit data, 1-bit parity and 1-bit acknowledge.
- Parity is even of '1' for 8-bit data in transmitter.
- Receiver generates acknowledge bit as '0' when transmission for 8-bit data and its parity has no error.
- When transmitter has no acknowledge (Acknowledge bit is '1' at tenth clock), error process is executed in transmitter.
- When acknowledge error is generated, host PC makes stop condition and transmits command which has error again.
- Background debugger command is composed of a bundle of packet.
- Start condition and stop condition notify the start and the stop of background debugger command respectively.

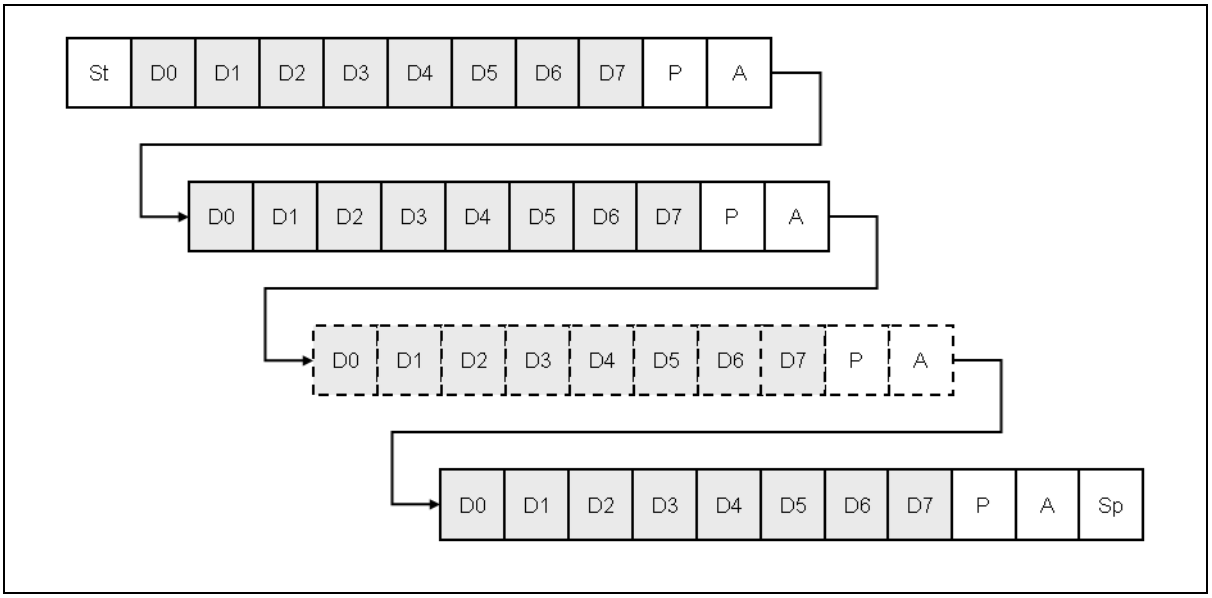



Figure 119. 10-bit Transmission Packet

## Packet transmission timing

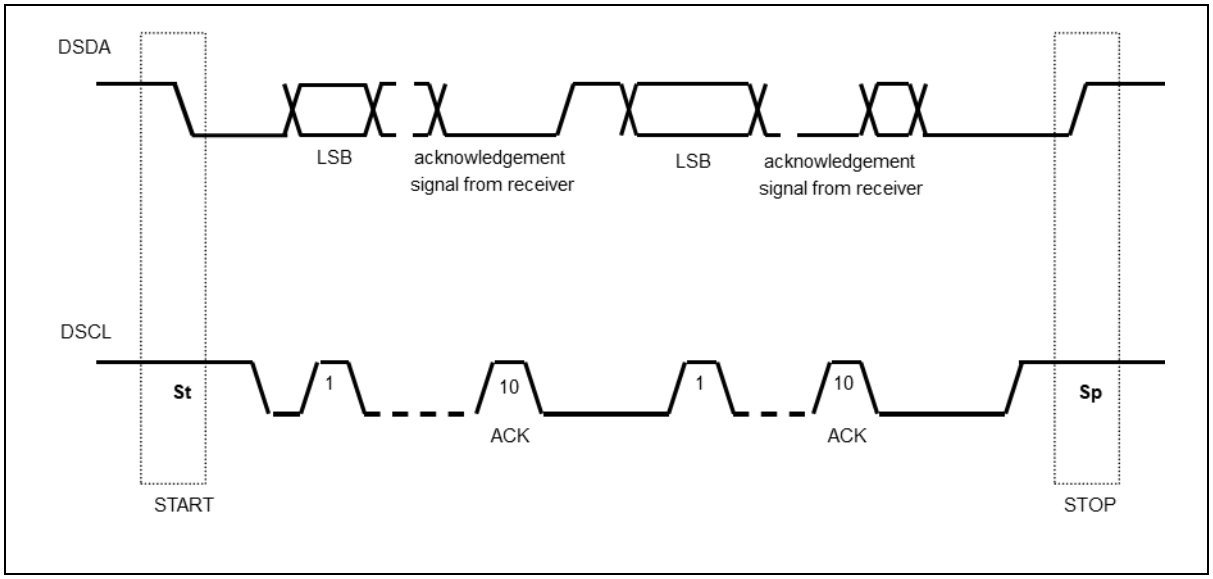



Figure 120. Data Transfer on Twin Bus

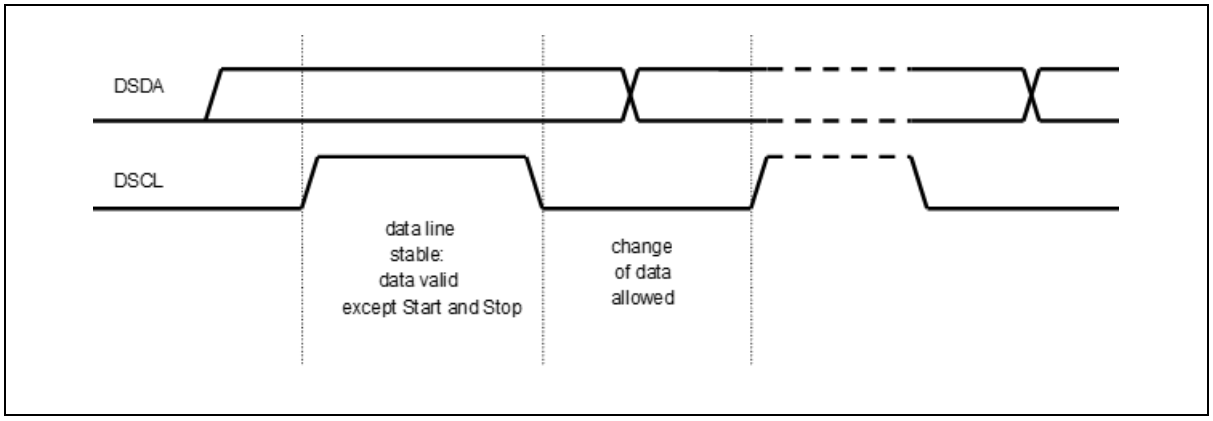



Figure 121. Bit Transfer on Serial Bus

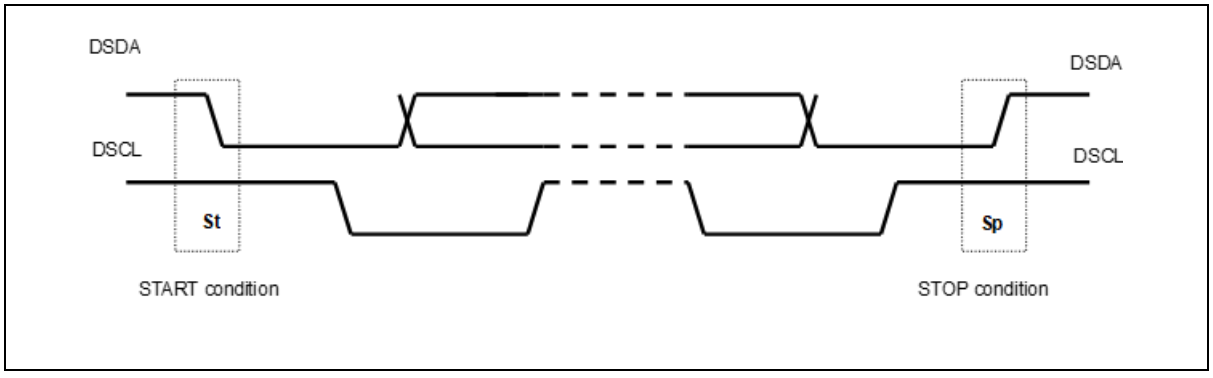



Figure 122. Start and Stop Condition

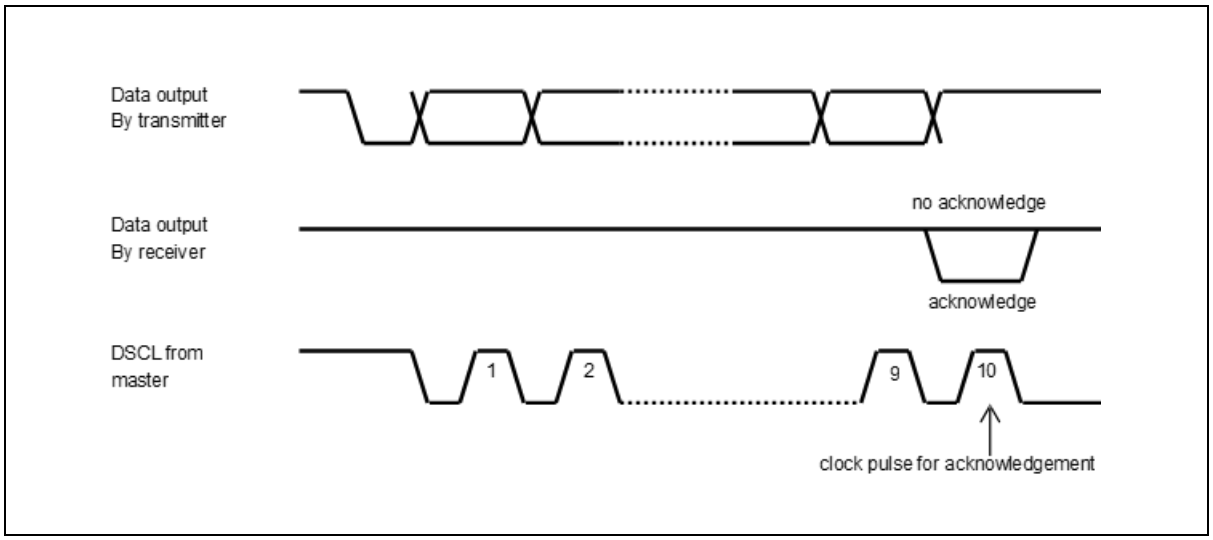



Figure 123. Acknowledge on Serial Bus

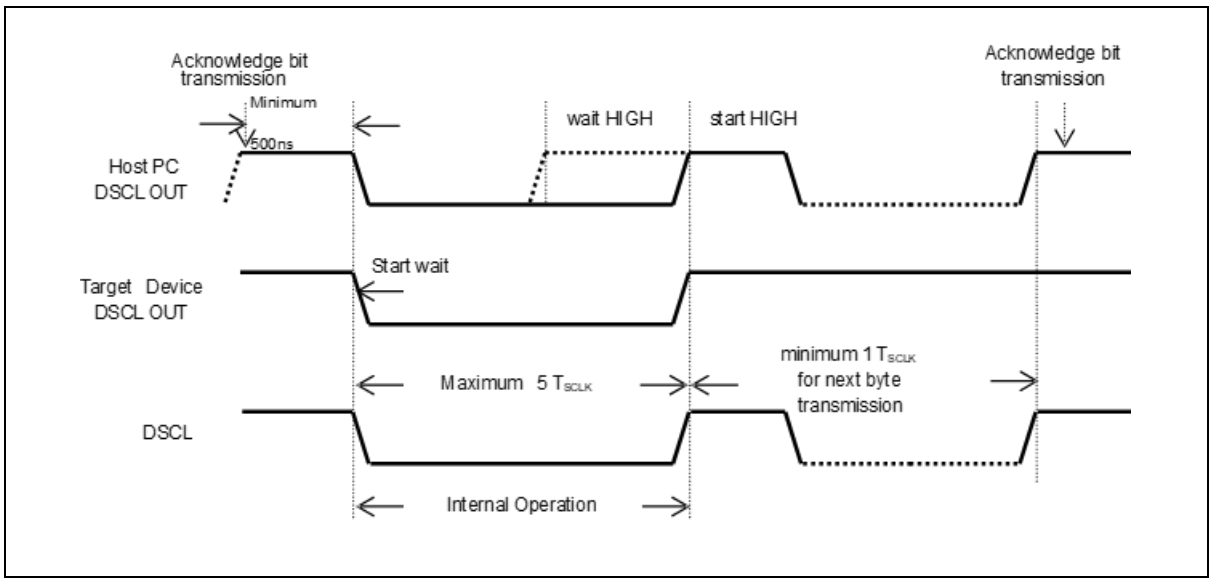



Figure 124. Clock Synchronization during Wait Procedure

### 23.5.3 Connection of transmission

Two-pin interface connection uses open-drain (wire-AND bidirectional I/O).

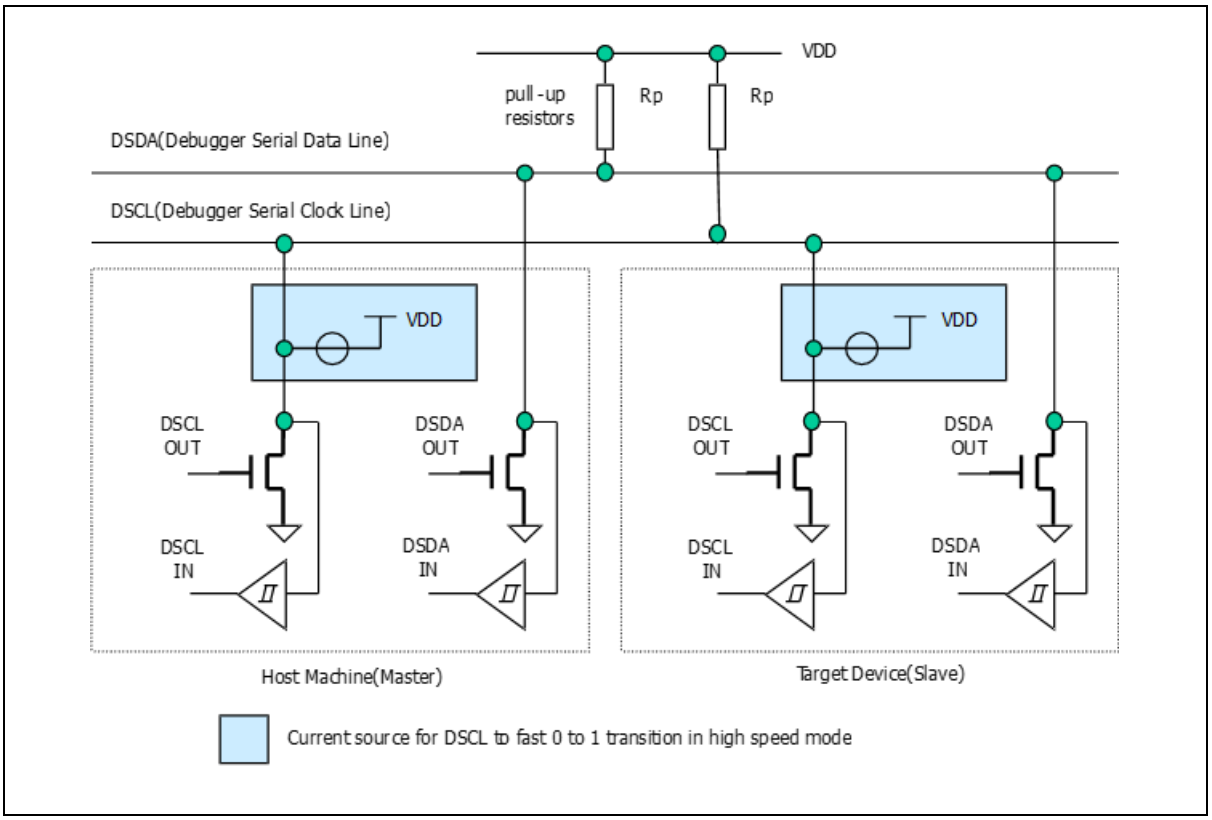



Figure 125. Connection of Transmission

# 24 Package information

ABOV provides A96L322 in 16 SOPN package as shown in figure 126 and table 54.

## 24.1 16 SOPN package information

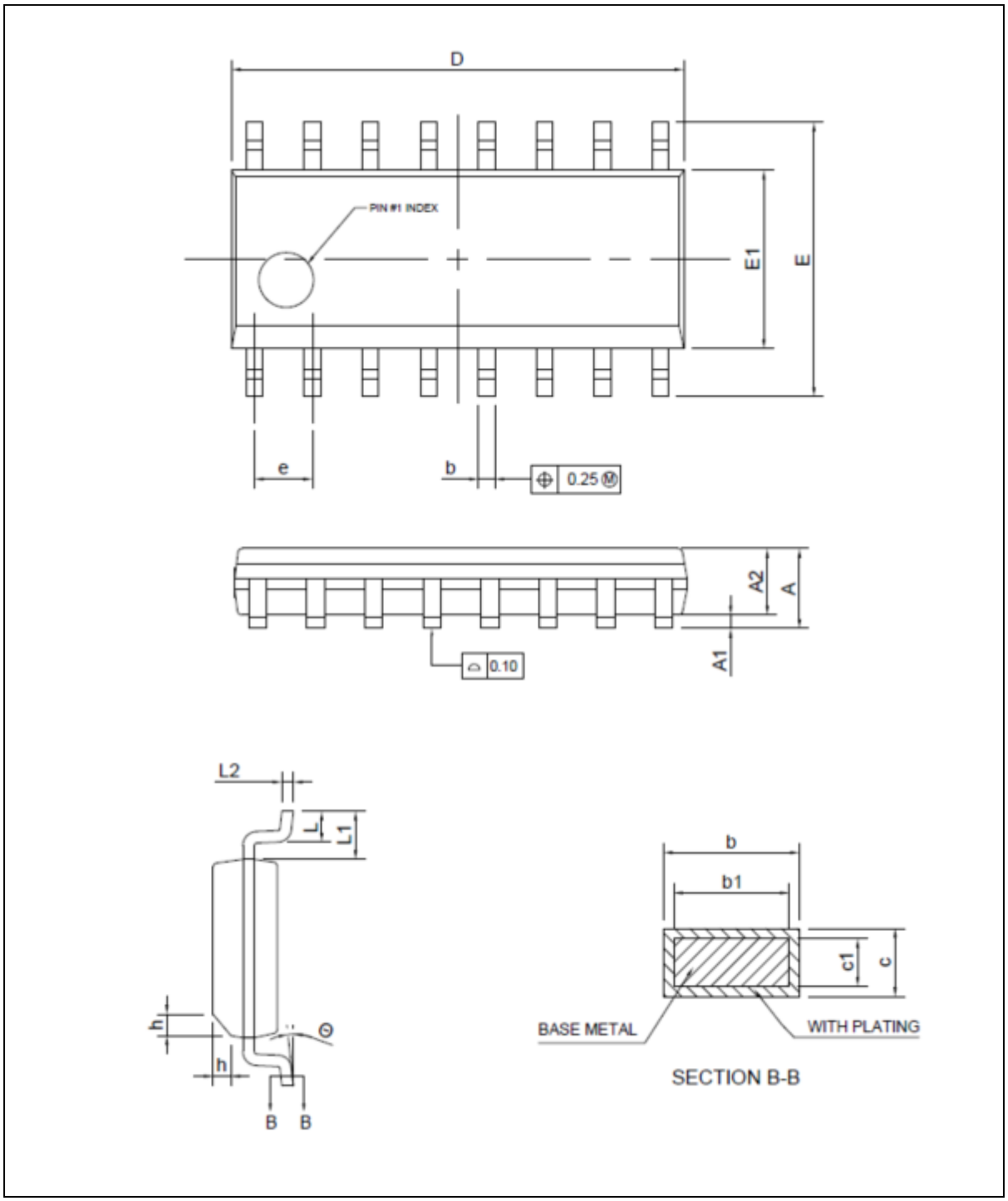



Figure 126. 16 SOPN Package Outline

Table 54. 16 SOPN Package Mechanical Data

| Symbol | Dimension (mm) | | |
|---|---|---|---|
| | Min. | Nom. | Max. |
| A | – | – | 1.75 |
| A1 | 0.10 | – | 0.25 |
| A2 | 1.25 | – | – |
| b | 0.31 | – | 0.51 |
| b1 | 0.28 | – | 0.48 |
| c | 0.10 | – | 0.26 |
| c1 | 0.10 | – | 0.23 |
| D | 9.70 | 9.90 | 10.20 |
| E | 5.80 | 6.00 | 6.20 |
| E1 | 3.70 | 3.90 | 4.20 |
| e | 1.27 BSC | | |
| L | 0.40 | – | 1.27 |
| L1 | 1.04 REF | | |
| L2 | 0.25 BSC | | |
| h | 0.25 | – | 0.50 |
| Θ | 0' | – | 8' |

**NOTES**:

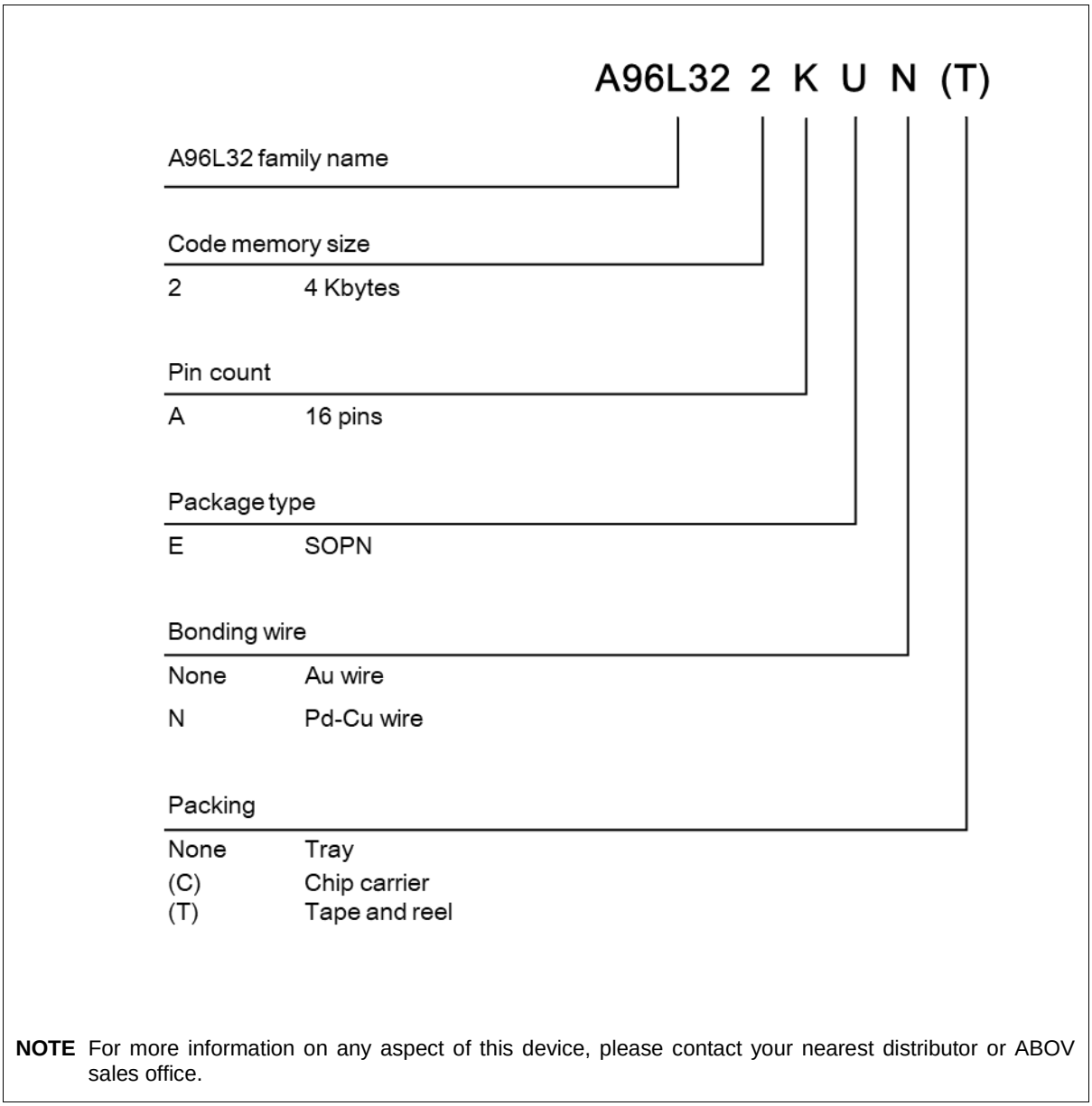
1. All dimension refer to JEDEC standard MS-012-AC.
2. Dimension 'D' does not include MOLD FLASH, PROTRUSIONS or GATE BURR. MOLD FLASH, PROTRUSIONS or GATE BURR shall not exceed 0.15 mm per end. Dimension 'E1' does not include INTERLEAD FLASH or PROTRUSION. INTERLEAD FLASH or PROTRUSION shall not exceed 0.25 mm per side.
3. Dimension 'b' does not include the DAMBAR PROTRUSION. Allowable DAMBAR PROTRUSION shall be 0.10 mm total in excess of the 'b' dimension at maximum material condition.

# 25 Ordering information

Table 55. A96L322 Device Ordering Information

| Device name | Flash | IRAM | EEPROM | ADC | I/O ports | Package type |
|---|---|---|---|---|---|---|
| A96L322AEN | 4 Kbyte | 256 byte | 128 bytes | 9 inputs | 14 | 16 SOPN |

* For available options or further information on the devices with "*" marks, please contact the ABOV Sales Office.




**NOTE** For more information on any aspect of this device, please contact your nearest distributor or ABOV sales office.

Figure 127. A96L322 Device Numbering Nomenclature

# Appendix

## A. Configure option

### Register description: configure option control

**CONFIGURE OPTION 1: ROM Address 001FH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| R_P | HL | – | VAPEN | – | – | – | RSTS |

Initial value: 00H

| | | |
|---|---|---|
| R_P | Code Read Protection | |
| | 0 | Disable |
| | 1 | Enable |
| HL | Code Write Protection | |
| | 0 | Disable |
| | 1 | Enable |
| VAPEN | Vector Area (00H – FFH) Write Protection | |
| | 0 | Disable Protection (Erasable by instruction) |
| | 1 | Enable Protection (Not erasable by instruction) |
| RSTS | Select RESETB pin | |
| | 0 | Disable RESETB pin (P10) |
| | 1 | Enable RESETB pin |

**CONFIGURE OPTION 2: ROM Address 001EH**

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| – | – | – | – | – | PAEN | PASS1 | PASS0 |

Initial value: 00H

| | | | |
|---|---|---|---|
| PAEN | Enable Specific Area Write Protection | | |
| | 0 | Disable (Erasable by instruction) | |
| | 1 | Enable (Not erasable by instruction) | |
| PASS [1:0] | Select Specific Area for Write Protection | | |
| | **NOTE**: When PAEN = '1', it is applied. | | |
| | PASS1 | PASS0 | Description |
| | 0 | 0 | 0.7Kbytes (Address 0100H – 03FFH) |
| | 0 | 1 | 1.7Kbytes (Address 0100H – 07FFH) |
| | 1 | 0 | 2.7Kbytes (Address 0100H – 0BFFH) |
| | 1 | 1 | 3.6KBytes (Address 0100H – 0F7FH) |

## B. Instruction table

- Instructions are either 1, 2 or 3 bytes long as listed in the 'Bytes' column in tables shown below.
- Each instruction takes either 1, 2 or 4 machine cycles to execute as listed in the following tables in this section.
- 1 machine cycle comprises 2 system clock cycles.

**Table 56. Instruction Table: Arithmetic**

| Arithmetic | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| ADD A,Rn | Add register to A | 1 | 1 | 28-2F |
| ADD A,dir | Add direct byte to A | 2 | 1 | 25 |
| ADD A,@Ri | Add indirect memory to A | 1 | 1 | 26-27 |
| ADD A,#data | Add immediate to A | 2 | 1 | 24 |
| ADDC A,Rn | Add register to A with carry | 1 | 1 | 38-3F |
| ADDC A,dir | Add direct byte to A with carry | 2 | 1 | 35 |
| ADDC A,@Ri | Add indirect memory to A with carry | 1 | 1 | 36-37 |
| ADDC A,#data | Add immediate to A with carry | 2 | 1 | 34 |
| SUBB A,Rn | Subtract register from A with borrow | 1 | 1 | 98-9F |
| SUBB A,dir | Subtract direct byte from A with borrow | 2 | 1 | 95 |
| SUBB A,@Ri | Subtract indirect memory from A with borrow | 1 | 1 | 96-97 |
| SUBB A,#data | Subtract immediate from A with borrow | 2 | 1 | 94 |
| INC A | Increment A | 1 | 1 | 04 |
| INC Rn | Increment register | 1 | 1 | 08-0F |
| INC dir | Increment direct byte | 2 | 1 | 05 |
| INC @Ri | Increment indirect memory | 1 | 1 | 06-07 |
| DEC A | Decrement A | 1 | 1 | 14 |
| DEC Rn | Decrement register | 1 | 1 | 18-1F |
| DEC dir | Decrement direct byte | 2 | 1 | 15 |
| DEC @Ri | Decrement indirect memory | 1 | 1 | 16-17 |
| INC DPTR | Increment data pointer | 1 | 2 | A3 |
| MUL AB | Multiply A by B | 1 | 4 | A4 |
| DIV AB | Divide A by B | 1 | 4 | 84 |
| DA A | Decimal Adjust A | 1 | 1 | D4 |

Table 57. Instruction Table: Logical

| Logical | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| ANL A,Rn | AND register to A | 1 | 1 | 58-5F |
| ANL A,dir | AND direct byte to A | 2 | 1 | 55 |
| ANL A,@Ri | AND indirect memory to A | 1 | 1 | 56-57 |
| ANL A,#data | AND immediate to A | 2 | 1 | 54 |
| ANL dir,A | AND A to direct byte | 2 | 1 | 52 |
| ANL dir,#data | AND immediate to direct byte | 3 | 2 | 53 |
| ORL A,Rn | OR register to A | 1 | 1 | 48-4F |
| ORL A,dir | OR direct byte to A | 2 | 1 | 45 |
| ORL A,@Ri | OR indirect memory to A | 1 | 1 | 46-47 |
| ORL A,#data | OR immediate to A | 2 | 1 | 44 |
| ORL dir,A | OR A to direct byte | 2 | 1 | 42 |
| ORL dir,#data | OR immediate to direct byte | 3 | 2 | 43 |
| XRL A,Rn | Exclusive-OR register to A | 1 | 1 | 68-6F |
| XRL A,dir | Exclusive-OR direct byte to A | 2 | 1 | 65 |
| XRL A, @Ri | Exclusive-OR indirect memory to A | 1 | 1 | 66-67 |
| XRL A,#data | Exclusive-OR immediate to A | 2 | 1 | 64 |
| XRL dir,A | Exclusive-OR A to direct byte | 2 | 1 | 62 |
| XRL dir,#data | Exclusive-OR immediate to direct byte | 3 | 2 | 63 |
| CLR A | Clear A | 1 | 1 | E4 |
| CPL A | Complement A | 1 | 1 | F4 |
| SWAP A | Swap Nibbles of A | 1 | 1 | C4 |
| RL A | Rotate A left | 1 | 1 | 23 |
| RLC A | Rotate A left through carry | 1 | 1 | 33 |
| RR A | Rotate A right | 1 | 1 | 03 |
| RRC A | Rotate A right through carry | 1 | 1 | 13 |

Table 58. Instruction Table: Data Transfer

| Data Transfer | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| MOV A,Rn | Move register to A | 1 | 1 | E8-EF |
| MOV A,dir | Move direct byte to A | 2 | 1 | E5 |
| MOV A,@Ri | Move indirect memory to A | 1 | 1 | E6-E7 |
| MOV A,#data | Move immediate to A | 2 | 1 | 74 |
| MOV Rn,A | Move A to register | 1 | 1 | F8-FF |
| MOV Rn,dir | Move direct byte to register | 2 | 2 | A8-AF |
| MOV Rn,#data | Move immediate to register | 2 | 1 | 78-7F |
| MOV dir,A | Move A to direct byte | 2 | 1 | F5 |
| MOV dir,Rn | Move register to direct byte | 2 | 2 | 88-8F |
| MOV dir,dir | Move direct byte to direct byte | 3 | 2 | 85 |
| MOV dir,@Ri | Move indirect memory to direct byte | 2 | 2 | 86-87 |
| MOV dir,#data | Move immediate to direct byte | 3 | 2 | 75 |
| MOV @Ri,A | Move A to indirect memory | 1 | 1 | F6-F7 |
| MOV @Ri,dir | Move direct byte to indirect memory | 2 | 2 | A6-A7 |
| MOV @Ri,#data | Move immediate to indirect memory | 2 | 1 | 76-77 |
| MOV DPTR,#data | Move immediate to data pointer | 3 | 2 | 90 |
| MOVC A,@A+DPTR | Move code byte relative DPTR to A | 1 | 2 | 93 |
| MOVC A,@A+PC | Move code byte relative PC to A | 1 | 2 | 83 |
| MOVX A,@Ri | Move external data(A8) to A | 1 | 2 | E2-E3 |
| MOVX A,@DPTR | Move external data(A16) to A | 1 | 2 | E0 |
| MOVX @Ri,A | Move A to external data(A8) | 1 | 2 | F2-F3 |
| MOVX @DPTR,A | Move A to external data(A16) | 1 | 2 | F0 |
| PUSH dir | Push direct byte onto stack | 2 | 2 | C0 |
| POP dir | Pop direct byte from stack | 2 | 2 | D0 |
| XCH A,Rn | Exchange A and register | 1 | 1 | C8-CF |
| XCH A,dir | Exchange A and direct byte | 2 | 1 | C5 |
| XCH A,@Ri | Exchange A and indirect memory | 1 | 1 | C6-C7 |
| XCHD A,@Ri | Exchange A and indirect memory nibble | 1 | 1 | D6-D7 |

Table 59. Instruction Table: Boolean

| Boolean | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| CLR C | Clear carry | 1 | 1 | C3 |
| CLR bit | Clear direct bit | 2 | 1 | C2 |
| SETB C | Set carry | 1 | 1 | D3 |
| SETB bit | Set direct bit | 2 | 1 | D2 |
| CPL C | Complement carry | 1 | 1 | B3 |
| CPL bit | Complement direct bit | 2 | 1 | B2 |
| ANL C,bit | AND direct bit to carry | 2 | 2 | 82 |
| ANL C,/bit | AND direct bit inverse to carry | 2 | 2 | B0 |
| ORL C,bit | OR direct bit to carry | 2 | 2 | 72 |
| ORL C,/bit | OR direct bit inverse to carry | 2 | 2 | A0 |
| MOV C,bit | Move direct bit to carry | 2 | 1 | A2 |
| MOV bit,C | Move carry to direct bit | 2 | 2 | 92 |

Table 60. Instruction Table: Branching

| Branching | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| ACALL addr 11 | Absolute jump to subroutine | 2 | 2 | 11→F1 |
| LCALL addr 16 | Long jump to subroutine | 3 | 2 | 12 |
| RET | Return from subroutine | 1 | 2 | 22 |
| RETI | Return from interrupt | 1 | 2 | 32 |
| AJMP addr 11 | Absolute jump unconditional | 2 | 2 | 01→E1 |
| LJMP addr 16 | Long jump unconditional | 3 | 2 | 02 |
| SJMP rel | Short jump (relative address) | 2 | 2 | 80 |
| JC rel | Jump on carry = 1 | 2 | 2 | 40 |
| JNC rel | Jump on carry = 0 | 2 | 2 | 50 |
| JB bit,rel | Jump on direct bit = 1 | 3 | 2 | 20 |
| JNB bit,rel | Jump on direct bit = 0 | 3 | 2 | 30 |
| JBC bit,rel | Jump on direct bit = 1 and clear | 3 | 2 | 10 |
| JMP @A+DPTR | Jump indirect relative DPTR | 1 | 2 | 73 |
| JZ rel | Jump on accumulator = 0 | 2 | 2 | 60 |
| JNZ rel | Jump on accumulator ≠0 | 2 | 2 | 70 |
| CJNE A,dir,rel | Compare A,direct jne relative | 3 | 2 | B5 |
| CJNE A,#d,rel | Compare A,immediate jne relative | 3 | 2 | B4 |
| CJNE Rn,#d,rel | Compare register, immediate jne relative | 3 | 2 | B8-BF |
| CJNE @Ri,#d,rel | Compare indirect, immediate jne relative | 3 | 2 | B6-B7 |
| DJNZ Rn,rel | Decrement register, jnz relative | 2 | 2 | D8-DF |
| DJNZ dir,rel | Decrement direct byte, jnz relative | 3 | 2 | D5 |

Table 61. Instruction Table: Miscellaneous

| Miscellaneous | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| NOP | No operation | 1 | 1 | 00 |

Table 62. Instruction Table: Additional Instructions

| Additional instructions (selected through EO[7:4]) | | | | |
|---|---|---|---|---|
| **Mnemonic** | **Description** | **Bytes** | **Cycles** | **Hex code** |
| MOVC @(DPTR++),A | M8051W/M8051EW-specific instruction supporting software download into program memory | 1 | 2 | A5 |
| TRAP | Software break command | 1 | 1 | A5 |

In the above table, an entry such as E8-EF indicates a continuous block of hex opcodes used for 8 different registers, and the register numbers of which are defined by the lowest three bits of the corresponding code. Non-continuous blocks of codes, shown as 11→F1 (for example), are used for absolute jumps and calls, with the top 3 bits of the code being used to store the top three bits of the destination address.

The CJNE instructions use the abbreviation #d for immediate data; other instructions use #data.

## C. Flash protection for invalid erase/ write

Appendix C shows example code to prevent code or data from being changed by abnormal operations such as noise, unstable power, and malfunction.

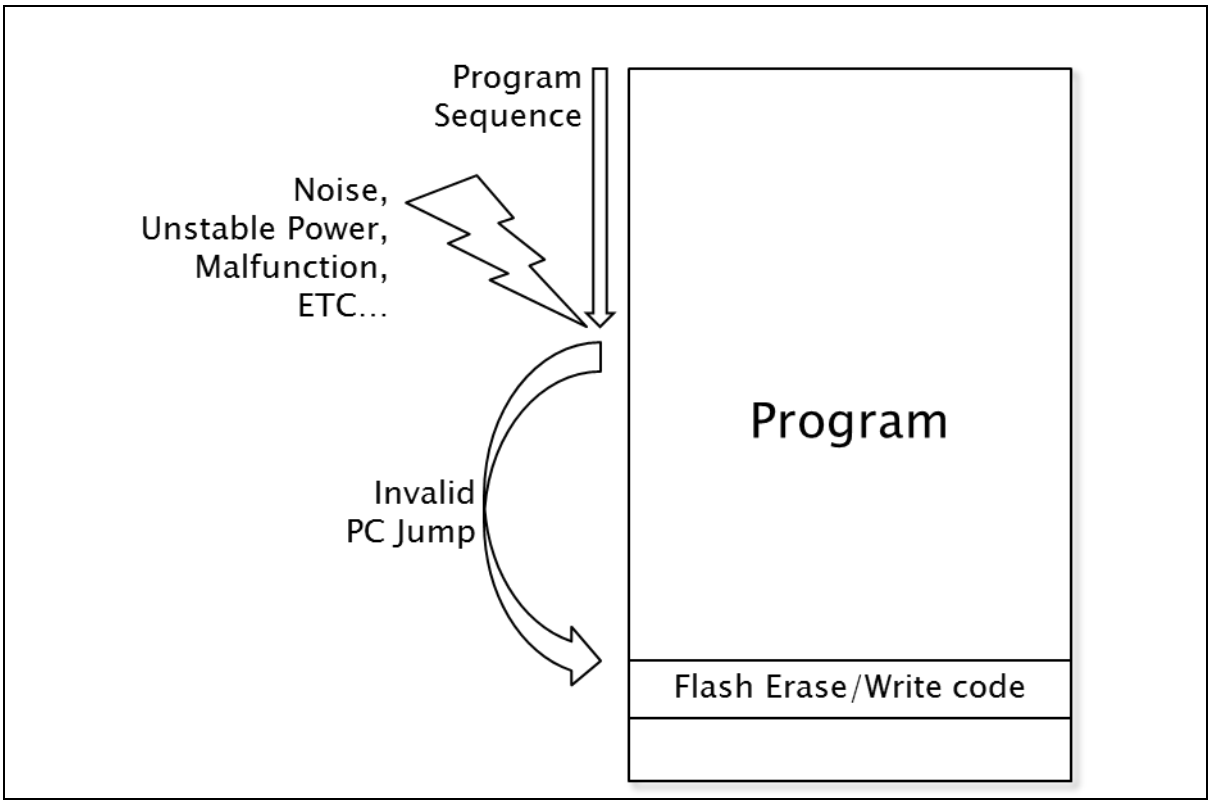



Figure 128. Flash Protection against Abnormal Operations

**How to protect the flash**

- Divide into decision and execution to Erase/Write in flash.
  - Check the program sequence from decision to execution in order of precedence about Erase/Write.
  - Setting the flags in program and check the flags in main loop at the end
  - When the Flash Erase/Write is executed, check the flags. If not matched, do not execute.
- Check the range of Flash Sector Address
  - If the flash sector address is outside of specific area, do not execute.
- Use the Dummy Address
  - Set the flash sector address to dummy address in usually run time.
  - Change the flash sector address to real area range shortly before Erase/Write.

- Even if invalid Erase/Write occurred, it will be Erase/Write in dummy address in flash.

• Use the LVR/LVI

- Unstable or low powers give an adverse effect on MCU. So use the LVR/LVI

**Protection flow description**

The flash protection procedure is described in flowchart in figure 129, and each step in the figure 129 is introduced in the following lists:

1. Initialization
   - Set the LVR/LVI. Check the power by LVR/LVI and do not execute under unstable or low power.
   - Initialize User_ID1/2/3
   - Set Flash Sector Address High/Middle/Low to Dummy address. Dummy address is set to unused area range in flash.
2. Decide to Write
   - When the Erase/Write are determined, set flag. Do not directly Erase/Write in flash.
   - Make the user data.
3. Check and Set User_ID1/2/3
   - In the middle of source, insert code which can check and set the flags.
   - By setting the User_ID 1/2/3 sequentially and identify the flow of the program.
4. Set Flash Sector Address
   - Set address to real area range shortly before Erase/Write in flash.
   - Set to Dummy address after Erase/Write. Even if invalid work occurred, it will be Erase/Write in Dummy address in flash.
5. Check Flags
   — If every flag (User_ID1/2/3, LVI, Flash Address Min/Max) was set, than do Erase/Write.
   — If the Flash Sector Address is outside of Min/Max, do not execute
   — Address Min/Max is set to unused area.
6. Initialize Flags
   — Initialize User_ID1/2/3
   — Set Flash Sector Address to Dummy Address

• Sample Source

— Refer to the ABOV website (www.abovsemi.com).

— It is created based on the MC97F2664.

— Each product should be modified according to the Page Buffer Size and Flash Size

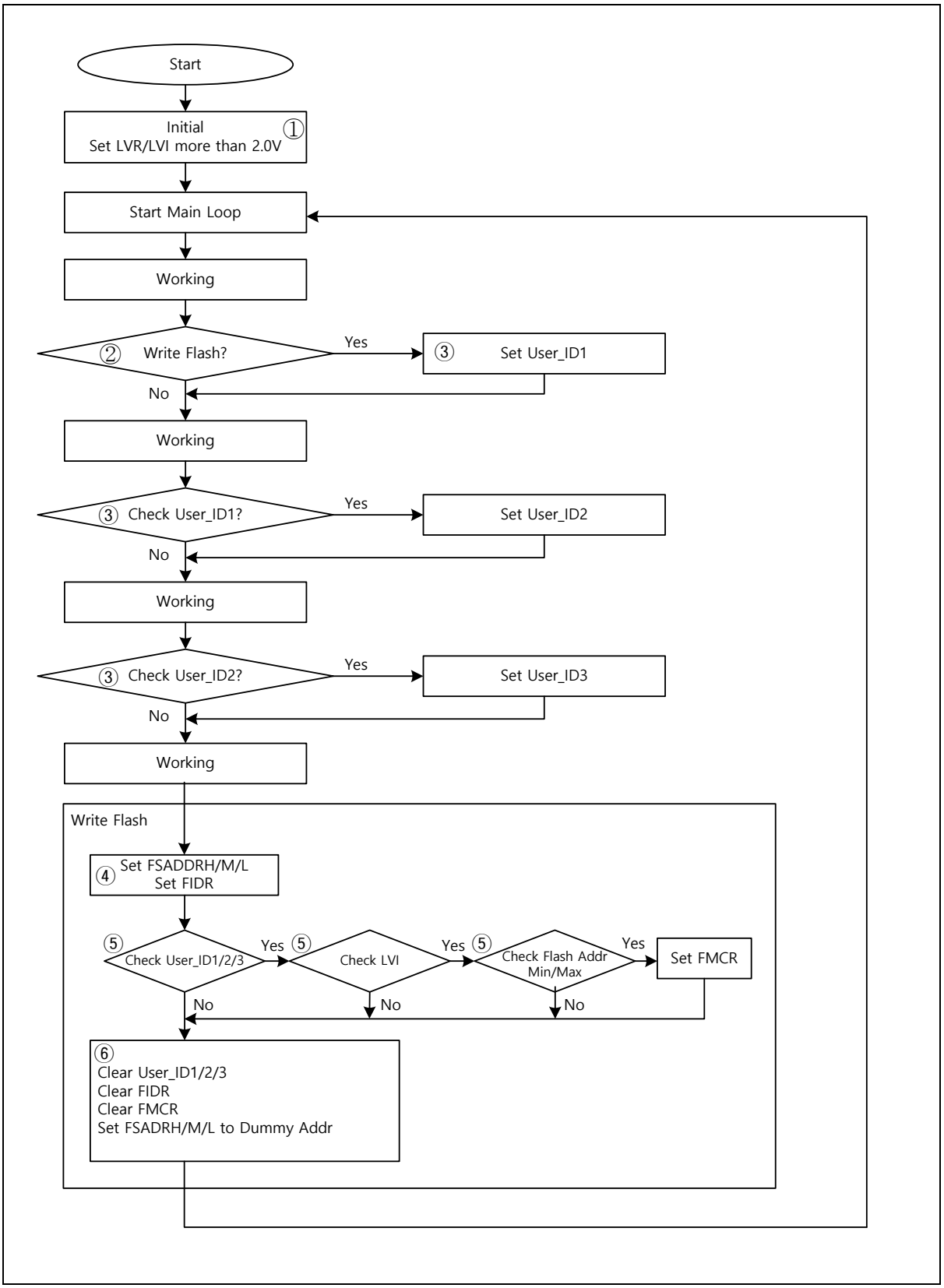



Figure 129. Flowchart of Flash Protection

**Other protection by the configure options**

- Protection by Configure option
  - Set flash protection by MCU Write Tool (OCD, PGM+, etc.)

    Vector Area:

    00H~FFH

    Specific Area:

    0.7KBytes (Address 0100H – 03FFH)

    1.7KBytes (Address 0100H – 07FFH)

    2.7KBytes (Address 0100H – 0BFFH)

    3.6KBytes (Address 0100H – 0F7FH)
  - The range of protection may be different each product.

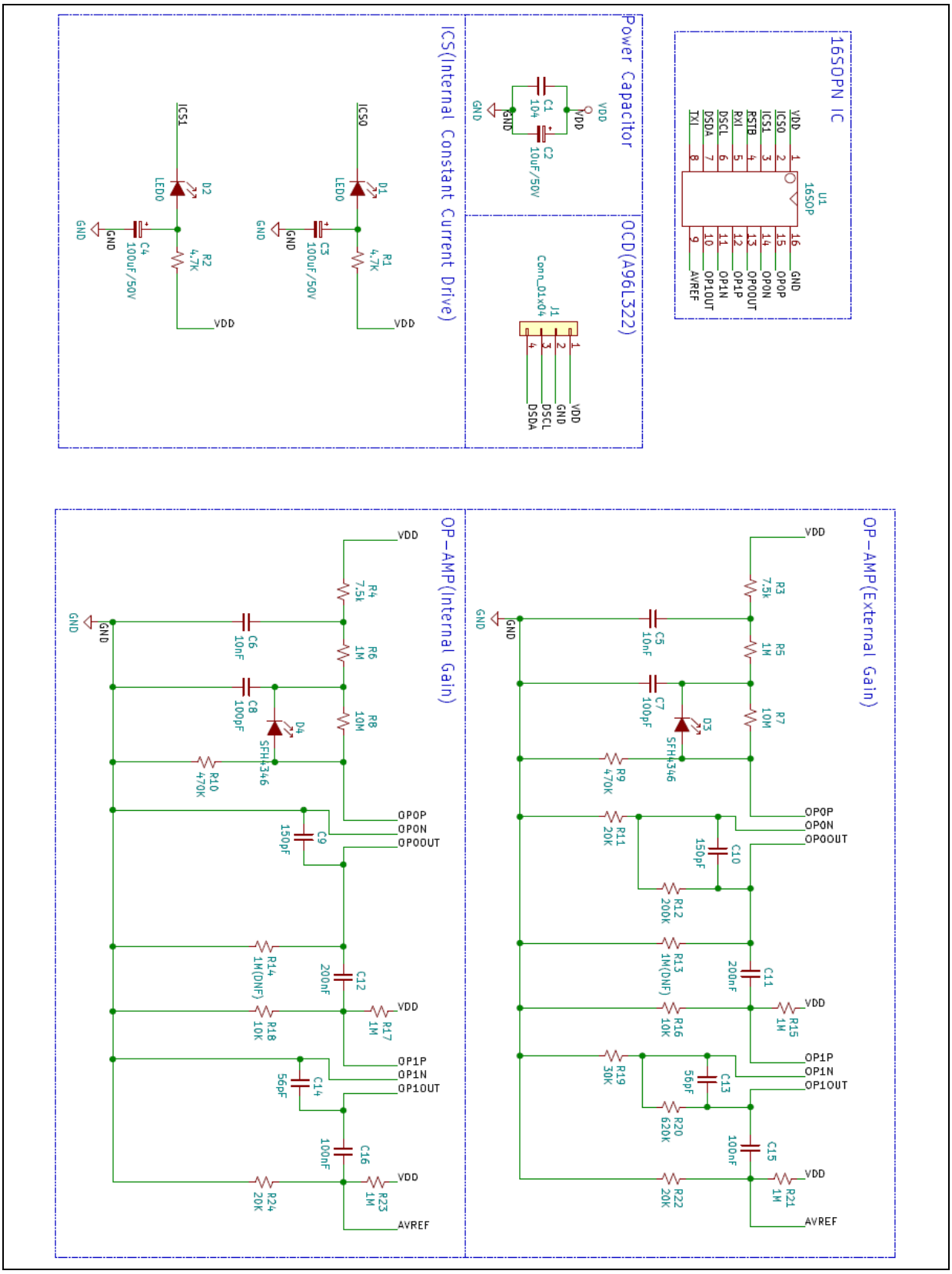



Figure 130. Example Circuit

# Revision history

| Date | Revision | Description |
|---|---|---|
| Sept.5, 2019 | 1.00 | First creation |
| | | |
| | | |

**Korea**

**Regional Office**, Seoul
R&D, Marketing & Sales
8th Fl., 330, Yeongdong-daero,
Gangnam-gu, Seoul,
06177, Korea

Tel: +82-2-2193-2200
Fax: +82-2-508-6903
www.abovsemi.com

**HQ**, Ochang
R&D, QA, and Test Center
93, Gangni 1-gil, Ochang-eup,
Cheongwon-gun,
Chungcheongbuk-do,
28126, Korea

Tel: +82-43-219-5200
Fax: +82-43-217-3534
www.abovsemi.com

**Domestic Sales Manager**
Tel: +82-2-2193-2206
Fax: +82-2-508-6903
Email: sales_kr@abov.co.kr

**Global Sales Manager**
Tel: +82-2-2193-2281
Fax: +82-2-508-6903
Email: sales_gl@abov.co.kr

**China Sales Manager**
Tel: +86-755-8287-2205
Fax: +86-755-8287-2204
Email: sales_cn@abov.co.kr

**ABOV Disclaimer**
**IMPORTANT NOTICE – PLEASE READ CAREFULLY**

ABOV Semiconductor ("ABOV") reserves the right to make changes, corrections, enhancements, modifications, and improvements to ABOV products and/or to this document at any time without notice. ABOV does not give warranties as to the accuracy or completeness of the information included herein. Purchasers should obtain the latest relevant information of ABOV products before placing orders. Purchasers are entirely responsible for the choice, selection, and use of ABOV products and ABOV assumes no liability for application assistance or the design of purchasers' products. No license, express or implied, to any intellectual property rights is granted by ABOV herein. ABOV disclaims all express and implied warranties and shall not be responsible or liable for any injuries or damages related to use of ABOV products in such unauthorized applications. ABOV and the ABOV logo are trademarks of ABOV. All other product or service names are the property of their respective owners. Information in this document supersedes and replaces the information previously supplied in any former versions of this document.

**© 2020 ABOV Semiconductor – All rights reserved**