



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

DATA SHEET

# NY8A053E

---

## 12 I/O 8-bit EPROM-Based MCU

**Version 1.1**

**Sept. 30, 2021**

---

NYQUEST TECHNOLOGY CO. reserves the right to change this document without prior notice. Information provided by NYQUEST is believed to be accurate and reliable. However, NYQUEST makes no warranty for any errors which may appear in this document. Contact NYQUEST to obtain the latest version of device specifications before placing your orders. No responsibility is assumed by NYQUEST for any infringement of patent or other rights of third parties which may result from its use. In addition, NYQUEST products are not authorized for use as critical components in life support devices/systems or aviation devices/systems, where a malfunction or failure of the product may reasonably be expected to result in significant injury to the user, without the express written approval of NYQUEST.

## Revision History

| <i>Version</i> | <i>Date</i> | <i>Description</i>   | <i>Modified Page</i> |
|----------------|-------------|--|----------------------|
| 1.0            | 2020/11/27  | Preliminary edition.   | -                    |
| 1.1            | 2021/09/30  | <ol style="list-style-type: none"><li>1. Revised PWM channels</li><li>2. RBIAS_L and RBIAS_H must set as "00" before sleep or halt mode</li><li>3. Revised I<sub>HALT</sub> @ VDD=3V, Halt mode, WDT enabled</li></ol> | 11<br>26, 53<br>82   |

## Table of Contents

|  |           |
|--|-----------|
| <b>1. 概述 .....</b>   | <b>7</b>  |
| 1.1 功能.....  | 7         |
| 1.2 NY8A053E 與 NY8A053D 的主要差異.....   | 9         |
| <b>1. General Description.....</b>   | <b>10</b> |
| 1.1 Features.....  | 10        |
| 1.2 Block Diagram.....   | 12        |
| 1.3 Pin Assignment .....   | 12        |
| 1.4 Pin Description.....   | 13        |
| <b>2. Memory Organization .....</b>  | <b>14</b> |
| 2.1 Program Memory .....   | 14        |
| 2.2 Data Memory .....  | 15        |
| <b>3. Function Description.....</b>  | <b>18</b> |
| 3.1 R-page Special Function Register .....                                   | 18        |
| 3.1.1 <i>INDF (Indirect Addressing Register)</i> .....                       | 18        |
| 3.1.2 <i>TMR0 (Timer0 Register)</i> .....                                    | 18        |
| 3.1.3 <i>PCL (Low Byte of PC[9:0])</i> .....                                 | 18        |
| 3.1.4 <i>STATUS (Status Register)</i> .....                                  | 19        |
| 3.1.5 <i>FSR (Register File Selection Register)</i> .....                    | 19        |
| 3.1.6 <i>PortA (PortA Data Register)</i> .....                               | 20        |
| 3.1.7 <i>PortB (PortB Data Register)</i> .....                               | 20        |
| 3.1.8 <i>PCON (Power Control Register)</i> .....                             | 20        |
| 3.1.9 <i>BWUCON (PortB Wake-up Control Register)</i> .....                   | 21        |
| 3.1.10 <i>PCHBUF (High Byte of PC)</i> .....                                 | 21        |
| 3.1.11 <i>ABPLCON (PortA/PortB Pull-Low Resistor Control Register)</i> ..... | 21        |
| 3.1.12 <i>BPHCON (PortB Pull-High Resistor Control Register)</i> .....       | 22        |
| 3.1.13 <i>INTE (Interrupt Enable Register)</i> .....                         | 22        |
| 3.1.14 <i>INTF (Interrupt Flag Register)</i> .....                           | 23        |
| 3.2 T0MD Register .....  | 23        |
| 3.3 F-page Special Function Register .....                                   | 25        |
| 3.3.1 <i>IOSTA (PortA I/O Control Register)</i> .....                        | 25        |
| 3.3.2 <i>IOSTB (PortB I/O Control Register)</i> .....                        | 25        |

|        |   |    |
|--------|---|----|
| 3.3.3  | <i>PS0CV (Prescaler0 Counter Value Register)</i> .....              | 25 |
| 3.3.4  | <i>BODCON (PortB Open-Drain Control Register)</i> .....             | 25 |
| 3.3.5  | <i>CMPCR (Comparator voltage select Control Register)</i> .....     | 26 |
| 3.3.6  | <i>PCON1 (Power Control Register1)</i> .....                        | 26 |
| 3.4    | S-page Special Function Register .....                              | 27 |
| 3.4.1  | <i>TMR1 (Timer1 Register)</i> .....                                 | 27 |
| 3.4.2  | <i>T1CR1 (Timer1 Control Register1)</i> .....                       | 28 |
| 3.4.3  | <i>T1CR2 (Timer1 Control Register2)</i> .....                       | 28 |
| 3.4.4  | <i>PWM1DUTY (PWM1 Duty Register)</i> .....                          | 29 |
| 3.4.5  | <i>PS1CV (Prescaler1 Counter Value Register)</i> .....              | 30 |
| 3.4.6  | <i>BZ1CR (Buzzer1 Control Register)</i> .....                       | 30 |
| 3.4.7  | <i>IRCR (IR Control Register)</i> .....                             | 31 |
| 3.4.8  | <i>TBHP (Table Access High Byte Address Pointer Register)</i> ..... | 32 |
| 3.4.9  | <i>TBHD (Table Access High Byte Data Register)</i> .....            | 32 |
| 3.4.10 | <i>P2CR1 (PWM2 Control Register1)</i> .....                         | 32 |
| 3.4.11 | <i>PWM2DUTY (PWM2 Duty Register)</i> .....                          | 32 |
| 3.4.12 | <i>TMRH (Timer High Byte Register)</i> .....                        | 33 |
| 3.4.13 | <i>TM34RH (Timer3 High Byte Register)</i> .....                     | 33 |
| 3.4.14 | <i>OSCCR (Oscillation Control Register)</i> .....                   | 33 |
| 3.4.15 | <i>P3CR1 (PWM3 Control Register1)</i> .....                         | 34 |
| 3.4.16 | <i>PWM3DUTY (PWM3 Duty Register)</i> .....                          | 34 |
| 3.4.17 | <i>P4CR1 (PWM4 Control Register1)</i> .....                         | 35 |
| 3.4.18 | <i>PWM4DUTY (PWM4 Duty Register)</i> .....                          | 35 |
| 3.5    | I/O Port .....  | 35 |
| 3.5.1  | <i>Block Diagram of IO Pins</i> .....                               | 37 |
| 3.6    | Timer0.....   | 44 |
| 3.7    | Timer1/PWM1/Buzzer1 .....   | 45 |
| 3.8    | PWM2/3/4 .....  | 47 |
| 3.9    | IR Carrier .....  | 48 |
| 3.10   | Low Voltage Detector (LVD) .....                                    | 49 |
| 3.11   | Voltage Comparator .....  | 50 |
| 3.11.1 | <i>Comparator Reference Voltage (Vref)</i> .....                    | 51 |
| 3.12   | Watch-Dog Timer (WDT).....  | 53 |
| 3.13   | Interrupt .....   | 54 |

|           |  |           |
|-----------|--|-----------|
| 3.13.1    | <i>Timer0 Overflow Interrupt</i> .....                                 | 54        |
| 3.13.2    | <i>Timer1 Underflow Interrupt</i> .....                                | 55        |
| 3.13.3    | <i>WDT Timeout Interrupt</i> .....                                     | 55        |
| 3.13.4    | <i>PB Input Change Interrupt</i> .....                                 | 55        |
| 3.13.5    | <i>External Interrupt</i> .....  | 55        |
| 3.13.6    | <i>LVD Interrupt</i> .....   | 55        |
| 3.14      | Oscillation Configuration .....  | 55        |
| 3.15      | Operating Mode .....   | 58        |
| 3.15.1    | <i>Normal Mode</i> .....   | 59        |
| 3.15.2    | <i>Slow Mode</i> .....   | 59        |
| 3.15.3    | <i>Standby Mode</i> .....  | 59        |
| 3.15.4    | <i>Halt Mode</i> .....   | 60        |
| 3.15.5    | <i>Wake-up Stable Time</i> .....                                       | 61        |
| 3.15.6    | <i>Summary of Operating Mode</i> .....                                 | 61        |
| 3.16      | Reset Process.....   | 61        |
| <b>4.</b> | <b>Instruction Set</b> .....   | <b>63</b> |
| <b>5.</b> | <b>Configuration Words</b> .....                                       | <b>79</b> |
| <b>6.</b> | <b>Electrical Characteristics</b> .....                                | <b>81</b> |
| 6.1       | Absolute Maximum Rating .....  | 81        |
| 6.2       | DC Characteristics .....   | 81        |
| 6.3       | OSC Characteristics.....   | 83        |
| 6.4       | Comparator / LVD Characteristics .....                                 | 83        |
| 6.5       | Characteristic Graph .....   | 83        |
| 6.5.1     | <i>Frequency vs. <math>V_{DD}</math> of <math>I_{HRC}</math></i> ..... | 83        |
| 6.5.2     | <i>Frequency vs. Temperature of <math>I_{HRC}</math></i> .....         | 84        |
| 6.5.3     | <i>Frequency vs. <math>V_{DD}</math> of <math>I_{LRC}</math></i> ..... | 84        |
| 6.5.4     | <i>Frequency vs. Temperature of <math>I_{LRC}</math></i> .....         | 84        |
| 6.6       | Recommended Operating Voltage .....                                    | 85        |
| 6.7       | LVR vs. Temperature.....   | 85        |
| <b>7.</b> | <b>Die Pad Diagram</b> .....   | <b>85</b> |
| <b>8.</b> | <b>Package Dimension</b> .....   | <b>86</b> |
| 8.1       | 8-Pin Plastic SOP (150 mil).....                                       | 86        |

---

|           |                                    |           |
|-----------|------------------------------------|-----------|
| 8.2       | 14-Pin Plastic SOP (150 mil) ..... | 86        |
| 8.3       | 14-Pin Plastic DIP (300 mil) ..... | 87        |
| <b>9.</b> | <b>Ordering Information.....</b>   | <b>87</b> |

## 1. 概述

NY8A053E是以EPROM作為記憶體的 8 位元微控制器，專為多IO產品的應用而設計，例如遙控器、風扇/燈光控制或是遊樂器周邊等等。採用CMOS製程並同時提供客戶低成本、高性能等顯著優勢。NY8A053E核心建立在RISC精簡指令集架構可以很容易地做編輯和控制，共有 55 條指令。除了少數指令需要 2 個時序，大多數指令都是 1 個時序即能完成，可以讓使用者輕鬆地以程式控制完成不同的應用。因此非常適合各種中低記憶容量但又複雜的應用。

在I/O的資源方面，NY8A053E有 12 根彈性的雙向I/O腳，每個I/O腳都有單獨的暫存器控制為輸入或輸出腳。而且每一個I/O腳位都有附加的程式控制功能如上拉或下拉電阻或開漏極(Open-Drain) 輸出。此外針對紅外線搖控的產品方面，NY8A053E內建了可選擇頻率的紅外載波發射口。

NY8A053E有兩組計時器，可用系統頻率當作一般的計時的應用或者從外部訊號觸發來計數。另外NY8A053E提供 4 組 10 位元解析度的PWM輸出或者蜂鳴器輸出，可用來驅動馬達、LED、或蜂鳴器等等。

NY8A053E採用雙時鐘機制，高速振盪或者低速振盪都可以分別選擇內部RC振盪或外部Crystal輸入。在雙時鐘機制下，NY8A053E可選擇多種工作模式如正常模式(Normal)、慢速模式(Slow mode)、待機模式(Standby mode) 與睡眠模式(Halt mode)可節省電力消耗延長電池壽命。並且微控制器在使用內部RC高速振盪時，低速振盪可以同時使用外部精準的Crystal計時。可以維持高速處理同時又能精準計算真實時間。

在省電的模式下如待機模式(Standby mode)與睡眠模式(Halt mode)中，有多種事件可以觸發中斷喚醒NY8A053E進入正常操作模式(Normal) 或 慢速模式(Slow mode) 來處理突發事件。

### 1.1 功能

- 寬廣的工作電壓：
  - 2.0V ~ 5.5V @系統頻率 $\leq$ 8MHz。
  - 2.2V ~ 5.5V @系統頻率 $>$ 8MHz。
- 寬廣的工作溫度：-40°C ~ 85°C。
- 1Kx14 bits EPROM。
- 64 bytes SRAM。
- 內建準確的低電壓偵測電路(LVD)。14 階 ( 2.2V ~ 4.15V)
- 內建準確的電壓比較器(Voltage Comparator)。
- 12 根可分別單獨控制輸入輸出方向的I/O腳(GPIO)、PA[3:0]、PB[7:0]。
- PA[3:0]及PB[3:0]可選擇輸入時使用內建下拉電阻。
- PB[7:0]可選擇上拉電阻或開漏極輸出(Open-Drain)。
- PB[3]可選擇當作輸入或開漏極輸出(Open-Drain)。
- 8 層程式堆棧(Stack)。
- 存取資料有直接或間接定址模式。
- 一組 8 位元上數計時器(Timer0)包含可程式化的頻率預除線路。

- 一組 10 位元下數計時器(Timer1)可選重複載入或連續下數計時。
- 4 個 10 位元的脈衝寬度調變輸出(PWM1~PWM4),PWM1/2/3/4 共用Timer1。
- 一個蜂鳴器輸出(BZ1)。
- 38/57KHz紅外線載波頻率可供選擇，同時載波之極性也可以根據數據作選擇。
- 紅外線載波發射口。
- 內建上電復位電路(POR)。
- 內建低壓復位功能(LVR)。
- 內建看門狗計時(WDT)，可由程式韌體控制開關。
- 雙時鐘機制，系統可以隨時切換高速振盪或者低速振盪。
  - 高速振盪： E\_HXT (超過 6MHz外部高速石英振盪)  
E\_XT (455K~6MHz外部石英振盪)  
I\_HRC (1~20MHz內部高速RC振盪)
  - 低速振盪： E\_LXT (32KHz外部低速石英振盪)  
I\_LRC (內部 32KHz低速RC振盪)
- 四種工作模式可隨系統需求調整電流消耗：正常模式(Normal)、慢速模式(Slow mode)、待機模式(Standby mode) 與 睡眠模式(Halt mode)。
- 六種硬體中斷：
  - Timer0 溢位中斷。
  - Timer1 借位中斷。
  - WDT 中斷。
  - PB 輸入狀態改變中斷。
  - 外部中斷輸入。
  - 低電壓偵測中斷。
- NY8A053E在待機模式(Standby mode)下的六種喚醒中斷：
  - Timer0 溢位中斷。
  - Timer1 借位中斷。
  - WDT 中斷。
  - PB 輸入狀態改變中斷。
  - 外部中斷輸入。
  - 低電壓偵測中斷。
- NY8A053E在睡眠模式(Halt mode)下的三種喚醒中斷：
  - WDT 中斷。
  - PB 輸入狀態改變中斷。
  - 外部中斷輸入。



**1.2 NY8A053E 與 NY8A053D 的主要差異**

| Item | Function                                | NY8A053E          | NY8A053D         |
|------|---|-------------------|------------------|
| 1    | PortB input pull-high resistor          | PB[7:0]           | PB[7:4], PB[2:0] |
| 2    | Comparator                              | Yes               | ---              |
| 3    | Low Voltage Detector(LVD) voltage level | 14 voltage levels | ---              |
| 4    | Comparator structure                    | shared with LVD   | ---              |
| 5    | PortB wake-up default status            | default: disable  | default: enable  |
| 6    | Input Schmitt Trigger                   | Enable/Disable    | ---              |
| 7    | PWM                                     | 10-bit PWM x4     | 8-bit PWM x1     |

## 1. General Description

NY8A053E is an EPROM based 8-bit MCU tailored for I/O based applications like remote controllers, fan/light controller, game controllers, toy and various controllers. NY8A053E adopts advanced CMOS technology to provide customers remarkable solution with low cost and high performance. RISC architecture is applied to NY8A053E and it provides 55 instructions. All instructions are executed in single instruction cycle except program branch and skip instructions which will take two instruction cycles. Therefore, NY8A053E is very suitable for those applications that are sophisticated but compact program size is required.

As NY8A053E address I/O type applications, it can provide 12 I/O pins for applications which require abundant input and output functionality. Moreover, each I/O pin may have additional features, like Pull-High/Pull-Low resistor and open-drain output type through programming. Moreover, NY8A053E has built-in infrared (IR) carrier generator with selectable IR carrier frequency and polarity for applications which demand remote control feature.

NY8A053E also provides 2 sets of timers which can be used as regular timer based on system oscillation or event counter with external trigger clock. Moreover, NY8A053E provides 4 set of 10-bit resolution Pulse Width Modulation (PWM) output and buzzer output in order to drive motor/LED and buzzer.

NY8A053E employs dual-clock oscillation mechanism, either high oscillation or low oscillation can be derived from internal resistor/capacitor oscillator or external crystal oscillator. Moreover, based on dual-clock mechanism, NY8A053E provides kinds of operation mode like Normal mode, Slow mode, Standby mode and Halt mode in order to save power consumption and lengthen battery operation life. Moreover, it is possible to use internal high-frequency oscillator as CPU operating clock source and external 32KHz crystal oscillator as timer clock input, so as to accurate count real time and maintain CPU working power.

While NY8A053E operates in Standby mode and Halt mode, kinds of event will issue interrupt requests and can wake-up NY8A053E to enter Normal mode and Slow mode in order to process urgent events.

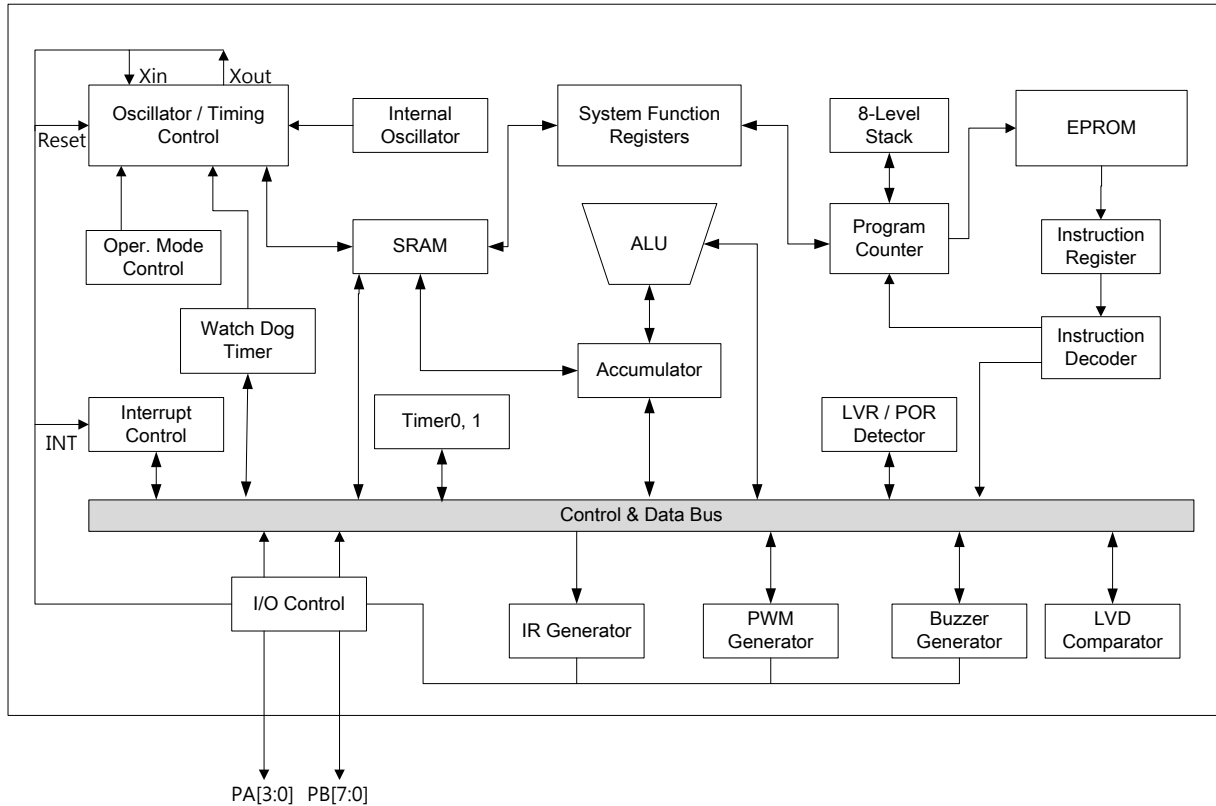
### 1.1 Features

- Wide operating voltage range:
  - 2.0V ~ 5.5V @system clock  $\leq$  8MHz.
  - 2.2V ~ 5.5V @system clock  $>$  8MHz.
- Wide operating temperature: -40°C ~ 85°C.
- 1K x 14 bits EPROM.
- 64 bytes SRAM.
- 12 general purpose I/O pins (GPIO), PA[3:0], PB[7:0], with independent direction control.
- PA[3:0] and PB[3:0] have features of Pull-Low resistor for input pin.
- PB[7:0] have features of Pull-High resistor, and open-drain output.
- PB[3] have feature of input or open-drain output.

- 8-level hardware Stack.
- Direct and indirect addressing modes for data access.
- One 8-bit up-count timer (Timer0) with programmable prescaler.
- One 10-bit reload or continuous down-count timers (Timer1).
- **Four 10-bit resolution PWM (PWM1~4) outputs.**
- One buzzer (BZ1) output.
- Selectable 38/57KHz IR carrier frequency and high/low polarity according to data value.
- Built-in high-precision Low-Voltage Detector (LVD).
- Built-in high-precision Voltage Comparator.
- Built-in Power-On Reset (POR).
- Built-in Low-Voltage Reset (LVR).
- Built-in Watch-Dog Timer (WDT) enabled/disabled by firmware control.
- Dual-clock oscillation: System clock can switch between high oscillation and low oscillation.
  - High oscillation: E\_HXT (External High Crystal Oscillator, above 6MHz)  
E\_XT (External Crystal Oscillator, 455K~6MHz)  
I\_HRC (Internal High Resistor/Capacitor Oscillator ranging from 1M~20MHz)
  - Low oscillation: E\_LXT (External Low Crystal Oscillator, about 32KHz)  
I\_LRC (Internal 32KHz oscillator)
- Four kinds of operation mode to reduce system power consumption:
  - Normal mode, Slow mode, Standby mode and Halt mode.
- Six hardware interrupt events:
  - Timer0 overflow interrupt.
  - Timer1 underflow interrupt.
  - WDT timeout interrupt.
  - PB input change interrupt.
  - External interrupt.
  - LVD interrupt.
- Six interrupt events to wake-up NY8A053E from Standby mode:
  - Timer0 overflow interrupt.
  - Timer1 underflow interrupt.
  - WDT timeout interrupt.
  - PB input change interrupt.
  - External interrupt.
  - LVD interrupt

- Three interrupt events to wake-up NY8A053E from Halt mode:
  - WDT timeout interrupt.
  - PB input change interrupt.
  - External interrupt.

### 1.2 Block Diagram



### 1.3 Pin Assignment

NY8A053E provides three kinds of package type which are SOP14, DIP14 and SOP8.

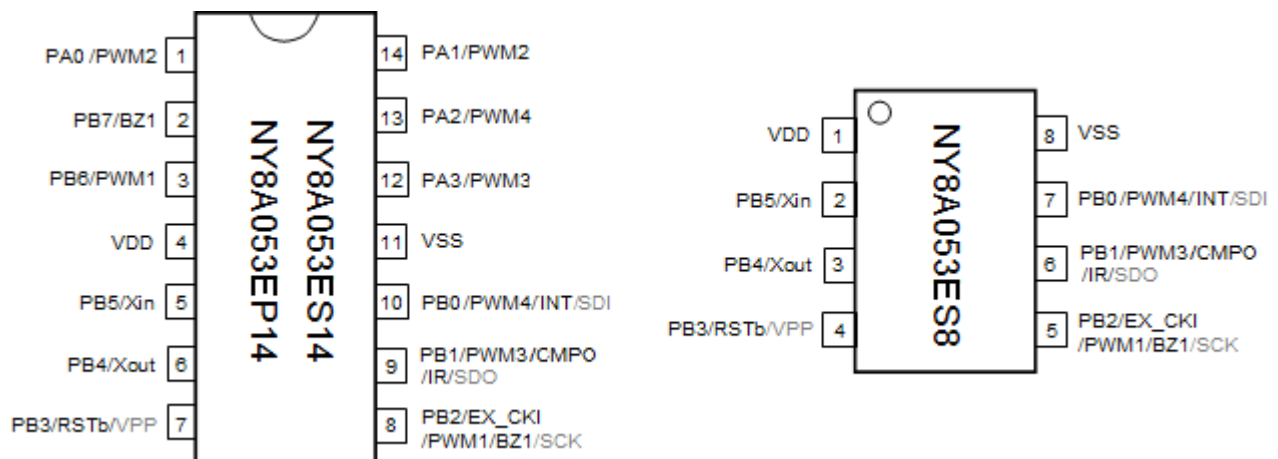


Figure 1 Package pin assignment

**1.4 Pin Description**

| Pin Name                       | I/O | Description   |
|--------------------------------|-----|---|
| PA0/ PWM2                      | I/O | PA0 is a bidirectional I/O pin.<br>PA0 is output of PWM2 signal by configuration words.   |
| PA1/ PWM2                      | I/O | PA1 is a bidirectional I/O pin.<br>PA1 is output of PWM2 signal by configuration words.   |
| PA2/ PWM4                      | I/O | PA2 is a bidirectional I/O pin.<br>PA2 is output of PWM4 signal by configuration words.   |
| PA3/ PWM3                      | I/O | PA3 is a bidirectional I/O pin.<br>PA3 is output of PWM3 signal by configuration words.   |
| PB0/ PWM4 / INT/<br>SDI        | I/O | PB0 is a bidirectional I/O pin.<br>PB0 is output of PWM4 signal by configuration words.<br>PB0 is input pin of external interrupt when EIS=1 & INTIE=1.<br>PB0 can be programming pad SDI.  |
| PB1/ PWM3/ IR/<br>CMPO/ SDO    | I/O | PB1 is a bidirectional I/O pin.<br>PB1 can be the output of PWM3, IR or comparator.<br>PB1 can be programming pad SDO.  |
| PB2/ EX_CK1/<br>PWM1/ BZ1/ SCK | I/O | PB2 is a bidirectional I/O pin.<br>It also can be timer clock source EX_CK1.<br>PB2 can be programming pad SCK.<br>PB2 is output of PWM1 signal by configuration words.<br>PB2 is output of Buzzer signal by configuration words.<br>PWM has higher priority over Buzzer functions. |
| PB3/ RSTb/ VPP                 | I/O | PB3 is an input pin or open-drain output pin.<br>It can be reset pin RSTb. If RSTb pin is low, it will reset NY8A053E.<br>If this pin is more than 7.75V, it also can make NY8A053E enter EPROM programming mode.   |
| PB4/ Xout                      | I/O | PB4 is a bidirectional I/O pin if I_HRC and I_LRC are adopted. PB4 also can be output of instruction clock.<br>PB4 is output of external crystal if E_HXT, E_XT or E_LXT is adopted.  |
| PB5/ Xin                       | I/O | PB5 is a bidirectional I/O pin if I_HRC and I_LRC are adopted.<br>PB5 is input of external crystal if E_HXT, E_XT or E_LXT is adopted.  |
| PB6/ PWM1                      | I/O | PB6 is a bidirectional I/O pin.<br>PB6 is output of PWM signal by configuration words.  |
| PB7/ BZ1                       | I/O | PB7 is a bidirectional I/O pin.<br>PB7 is output of Buzzer signal by configuration words.   |
| VDD                            | -   | Positive power supply.  |
| VSS                            | -   | Ground.   |

## 2. Memory Organization

NY8A053E memory is divided into two categories: one is program memory and the other is data memory.

### 2.1 Program Memory

The program memory space of NY8A053E is 1K words. Therefore, the Program Counter (PC) is 10-bit wide in order to address any location of program memory.

Some locations of program memory are reserved as interrupt entrance. Power-On Reset vector is located at 0x000. Software interrupt vector is located at 0x001. Internal and external hardware interrupt vector is located at 0x008.

NY8A053E provides instruction CALL, GOTOA, CALLA to address 256 location of program space. NY8A053E provides instruction GOTO to address 512 location of program space. NY8A053E also provides instructions LCALL and LGOTO to address any location of program space.

When a call or interrupt is happening, next ROM address is written to top of the stack, when RET, RETIA or RETIE instruction is executed, the top of stack data is read and load to PC.

NY8A053E program ROM address 0x3FE~0x3FF are reserved space, if user tries to write code in these addresses will get unexpected false functions.

NY8A053E program ROM address 0x00E~0x00F are preset rolling code can be released and used as normal program space.

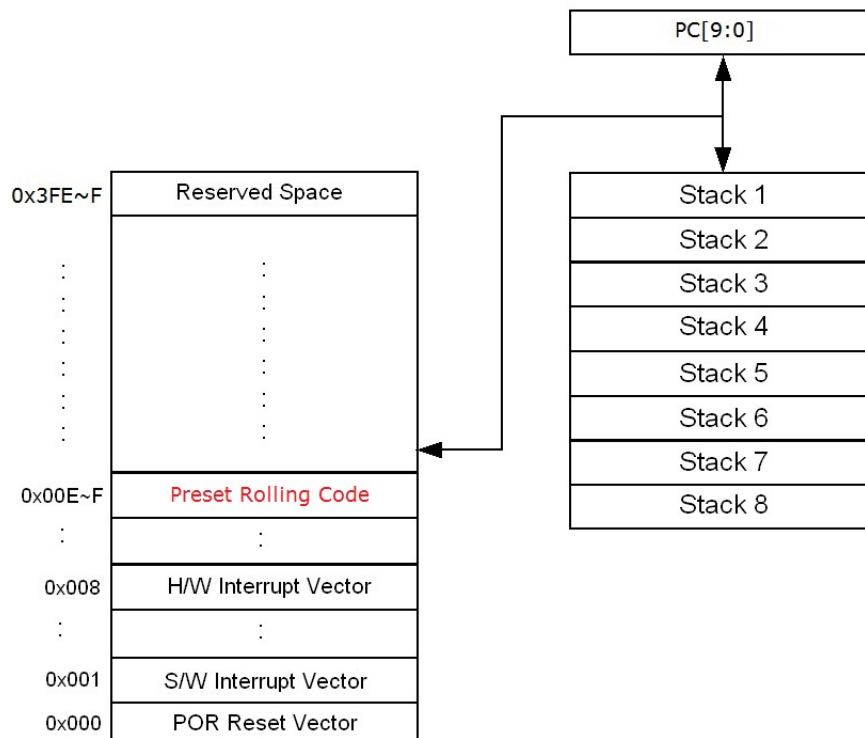


Figure 2 Program Memory Address Mapping

## 2.2 Data Memory

According to instructions used to access data memory, the data memory can be divided into three kinds of categories: one is R-page Special-function Register (SFR) + General Purpose Register (GPR), another is F-page SFR and the other is S-page SFR. GPR are made of SRAM and user can use them to store variables or intermediate results.

R-page data memory is divided into 4 banks and can be accessed directly or indirectly through a SFR register which is File Select Register (FSR). FSR[7:6] are used as Bank register BK[1:0] to select one bank out of the 4 banks.

R-page register can be divided into addressing mode: direct addressing mode and indirect addressing mode.

The indirect addressing mode of data memory access is described in the following graph. This indirect addressing mode is implied by accessing register INDF. The bank selection is determined by FSR[7:6] and the location selection is from FSR[5:0].

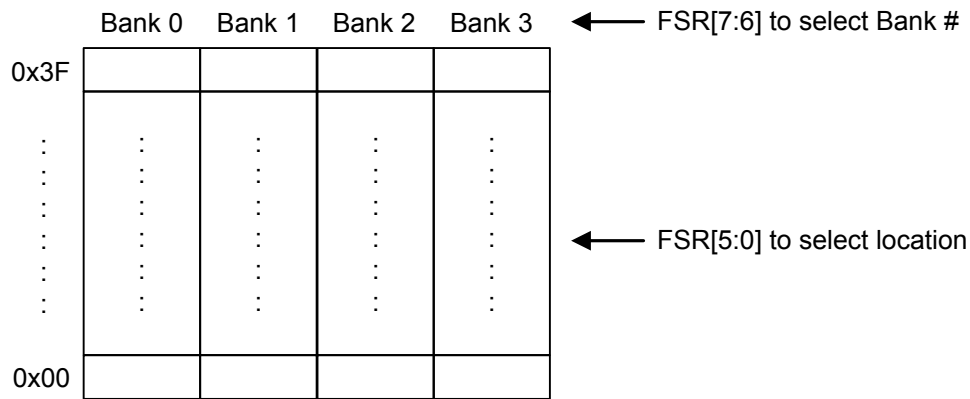


Figure 3 Indirect Addressing Mode of Data Memory Access

The direct addressing mode of data memory access is described below. The bank selection is determined by FSR[7:6] and the location selection is from instruction op-code[5:0] immediately.

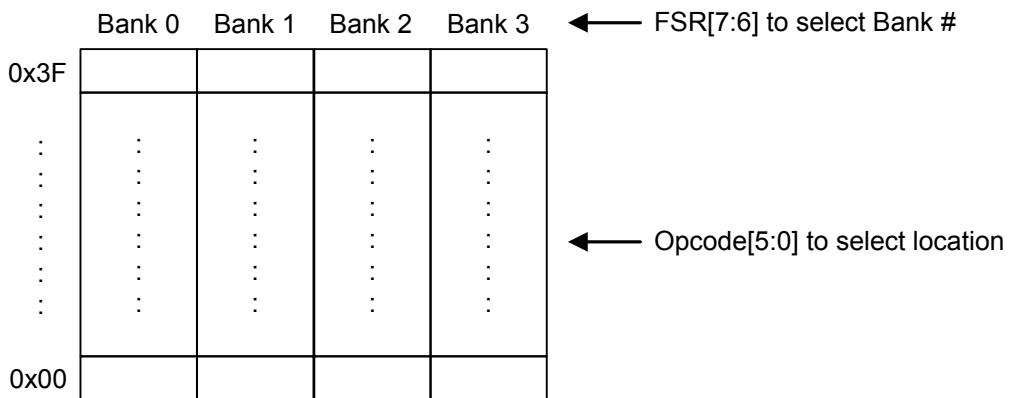


Figure 4 Direct Addressing Mode of Data Memory Access

R-page SFR can be accessed by general instructions like arithmetic instructions and data movement instructions. The R-page SFR occupies address from 0x0 to 0xF of Bank 0. However, the same address range of Bank 1, Bank 2 and Bank 3 are mapped back to Bank 0. In other words, R-page SFR physically existed at Bank 0. The GPR physically occupy address from 0x10 to 0x3F of Bank 0 and 0x10 to 0x1F of Bank 1. Other banks in address from 0x10 to 0x3F are mapped back as the Table 1 shows.

The NY8A053E register name and address mapping of R-page SFR are described in the following table.

| <b>FSR[7:6]<br/>Address</b> | <b>00<br/>(Bank 0)</b>   | <b>01<br/>(Bank 1)</b>            | <b>10<br/>(Bank 2)</b> | <b>11<br/>(Bank 3)</b> |
|-----------------------------|--------------------------|-----------------------------------|------------------------|------------------------|
| 0x0                         | INDF                     | <i>The same mapping as Bank 0</i> |                        |                        |
| 0x1                         | TMR0                     |                                   |                        |                        |
| 0x2                         | PCL                      |                                   |                        |                        |
| 0x3                         | STATUS                   |                                   |                        |                        |
| 0x4                         | FSR                      |                                   |                        |                        |
| 0x5                         | PORTA                    |                                   |                        |                        |
| 0x6                         | PORTB                    |                                   |                        |                        |
| 0x7                         | -                        |                                   |                        |                        |
| 0x8                         | PCON                     |                                   |                        |                        |
| 0x9                         | BWUCON                   |                                   |                        |                        |
| 0xA                         | PCHBUF                   |                                   |                        |                        |
| 0xB                         | ABPLCON                  |                                   |                        |                        |
| 0xC                         | BPHCON                   |                                   |                        |                        |
| 0xD                         | -                        |                                   |                        |                        |
| 0xE                         | INTE                     |                                   |                        |                        |
| 0xF                         | INTF                     |                                   |                        |                        |
| 0x10 ~ 0x1F                 | General Purpose Register | General Purpose Register          | <i>Mapped to bank0</i> | <i>Mapped to bank1</i> |
| 0x20 ~ 0x3F                 | General Purpose Register | <i>Mapped to bank0</i>            |                        |                        |

Table 1 R-page SFR Address Mapping

F-page SFR can be accessed only by instructions IOST and IOSTR. S-page SFR can be accessed only by instructions SFUN and SFUNR. FSR[7:6] bank select bits are ignored while F-page and S-page register is accessed. The register name and address mapping of F-page and S-page are depicted in the following table.



| SFR Category<br>Address | F-page SFR | S-page SFR |
|-------------------------|------------|------------|
| 0x0                     | -          | TMR1       |
| 0x1                     | -          | T1CR1      |
| 0x2                     | -          | T1CR2      |
| 0x3                     | -          | PWM1DUTY   |
| 0x4                     | -          | PS1CV      |
| 0x5                     | IOSTA      | BZ1CR      |
| 0x6                     | IOSTB      | IRCR       |
| 0x7                     | -          | TBHP       |
| 0x8                     | -          | TBHD       |
| 0x9                     | -          | -          |
| 0xA                     | PS0CV      | P2CR1      |
| 0xB                     | -          | -          |
| 0xC                     | BODCON     | PWM2DUTY   |
| 0xD                     | -          | TMRH       |
| 0xE                     | CMPCR      | TM34RH     |
| 0xF                     | PCON1      | OSCCR      |
| 0x10                    |            |            |
| 0x11                    |            | P3CR1      |
| 0x12                    |            |            |
| 0x13                    |            | PWM3DUTY   |
| 0x14                    |            |            |
| 0x15                    |            |            |
| 0x16                    |            | P4CR1      |
| 0x17                    |            |            |
| 0x18                    |            | PWM4DUTY   |

Table 2 F-page and S-page SFR Address Mapping

### 3. Function Description

This chapter will describe the detailed operations of NY8A053E.

#### 3.1 R-page Special Function Register

##### 3.1.1 INDF (Indirect Addressing Register)

| Name          | SFR Type | Addr. | Bit7      | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|-----------|------|------|------|------|------|------|------|
| INDF          | R        | 0x0   | INDF[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | R/W       |      |      |      |      |      |      |      |
| Initial Value |          |       | xxxxxxxx  |      |      |      |      |      |      |      |

The register INDF is not physically existed and it is used as indirect addressing mode. Any instruction accessing INDF actually accesses the register pointed by register FSR

##### 3.1.2 TMR0 (Timer0 Register)

| Name          | SFR Type | Addr. | Bit7      | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|-----------|------|------|------|------|------|------|------|
| TMR0          | R        | 0x1   | TMR0[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | R/W       |      |      |      |      |      |      |      |
| Initial Value |          |       | xxxxxxxx  |      |      |      |      |      |      |      |

When read the register TMR0, it actually read the current running value of Timer0.

Write the register TMR0 will change the current value of Timer0.

Timer0 clock source can be from instruction clock  $F_{INST}$ , or from external pin EX\_CK1, or from Low Oscillator Frequency according to T0MD and configuration word setting.

##### 3.1.3 PCL (Low Byte of PC[9:0])

| Name          | SFR Type | Addr. | Bit7     | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|----------|------|------|------|------|------|------|------|
| PCL           | R        | 0x2   | PCL[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | R/W      |      |      |      |      |      |      |      |
| Initial Value |          |       | 0x00     |      |      |      |      |      |      |      |

The register PCL is the least significant byte (LSB) of 10-bit PC. PCL will be increased by one after one instruction is executed except some instructions which will change PC directly. The high byte of PC, i.e. PC[9:8], is not directly accessible. Update of PC[9:8] must be done through register PCHBUF.

For GOTO instruction, PC[8:0] is from instruction word and PC[9] is loaded from PCHBUF[1]. For CALL instruction, PC[7:0] is from instruction word and PC[9:8] is loaded from PCHBUF[1:0]. Moreover the next PC address, i.e. PC+1, will push onto top of Stack. For LGOTO instruction, PC[9:0] is from instruction word.

For LCALL instruction, PC[9:0] is from instruction word. Moreover the next PC address, i.e. PC+1, will push onto top of Stack.

### 3.1.4 STATUS (Status Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6 | Bit5 | Bit4    | Bit3    | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|------|------|------|---------|---------|------|------|------|
| STATUS        | R        | 0x3   | GP7  | GP6  | GP5  | /TO     | /PD     | Z    | DC   | C    |
| R/W Property  |          |       | R/W  | R/W  | R/W  | R/W(*2) | R/W(*1) | R/W  | R/W  | R/W  |
| Initial Value |          |       | 0    | 0    | 0    | 1       | 1       | X    | X    | X    |

The register STATUS contains result of arithmetic instructions and reasons to cause reset.

**C:** Carry/Borrow bit

C=1, carry is occurred for addition instruction or borrow is not occurred for subtraction instruction.

C=0, carry is not occurred for addition instruction or borrow is occurred for subtraction instruction.

**DC:** Half Carry/half Borrow bit

DC=1, carry from the 4th LSB is occurred for addition instruction or borrow from the 4th LSB is not occurred for subtraction instruction.

DC=0, carry from the 4th LSB is not occurred for addition instruction or borrow from the 4th LSB is occurred for subtraction instruction.

**Z:** Zero bit

Z=1, result of logical operation is zero.

Z=0, result of logical operation is not zero.

**/PD:** Power down flag bit

/PD=1, after power-up or after instruction CLRWDT is executed.

/PD=0, after instruction SLEEP is executed.

**/TO:** Time overflow flag bit

/TO=1, after power-up or after instruction CLRWDT or SLEEP is executed.

/TO=0, WDT timeout is occurred.

**GP7, GP6, GP5:** General purpose read/write register bit.

(\*1) can be cleared by sleep instruction.

(\*2) can be set by clrwtd instruction.

### 3.1.5 FSR (Register File Selection Register)

| Name          | SFR Type | Addr. | Bit7    | Bit6 | Bit5     | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------|------|----------|------|------|------|------|------|
| FSR           | R        | 0x4   | BK[1:0] |      | FSR[5:0] |      |      |      |      |      |
| R/W Property  |          |       | R/W     |      |          |      |      |      |      |      |
| Initial Value |          |       | 0       | 0    | X        | X    | X    | X    | X    | X    |

**FSR[5:0]:** Select one register out of 64 registers of specific Bank.

**BK[1:0]:** Bank register used to select one specific bank of data memory. BK[1:0]=00b, Bank 0 is selected.

BK[1:0]=01b, Bank 1 is selected. BK[1:0]=10b, Bank 2 is selected. BK[1:0]=11b, Bank 3 is selected.

### 3.1.6 PortA (PortA Data Register)

| Name          | SFR Type | Addr. | Bit7   | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|--|------|------|------|------|------|------|------|
| PortA         | R        | 0x5   | GP7  | GP6  | GP5  | GP4  | PA3  | PA2  | PA1  | PA0  |
| R/W Property  |          |       | R/W  |      |      |      |      |      |      |      |
| Initial Value |          |       | Data latch value is xxxx, read value is xxxx port value(PA3~PA0) |      |      |      |      |      |      |      |

While reading PortA, it will get the status of pins of PA regardless that pin is configured as input or output pin.

While writing to PortA, data is written to PA's data latch.

GP7 ~ GP4: general register.

### 3.1.7 PortB (PortB Data Register)

| Name          | SFR Type | Addr. | Bit7   | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|--|------|------|------|------|------|------|------|
| PortB         | R        | 0x6   | PB7  | PB6  | PB5  | PB4  | PB3  | PB2  | PB1  | PB0  |
| R/W Property  |          |       | R/W  |      |      |      |      |      |      |      |
| Initial Value |          |       | Data latch value is xxxxxxxx, read value is xxxxxxxx port value(PB7~PB0) |      |      |      |      |      |      |      |

While reading PortB, it will get the status of pins of PB regardless that pin is configured as input or output pin.

While writing to PortB, data is written to PB's data latch.

### 3.1.8 PCON (Power Control Register)

| Name          | SFR Type | Addr. | Bit7  | Bit6 | Bit5   | Bit4 | Bit3  | Bit2  | Bit1 | Bit0 |
|---------------|----------|-------|-------|------|--------|------|-------|-------|------|------|
| PCON          | R        | 0x8   | WDTEN | EIS  | LV DEN | GP4  | LVREN | CMPEN | GP1  | GP0  |
| R/W Property  |          |       | R/W   |      |        |      |       |       |      |      |
| Initial Value |          |       | 1     | 0    | 0      | 0    | 1     | 0     | 0    | 0    |

**GP4, GP1, GP0:** General read/write register bits.

**CMPEN:** Enable/disable Comparator.

CMPEN=1, enable comparator.

CMPEN=0, disable comparator.

**LVREN:** Enable/disable LVR.

LVREN=1, enable LVR.

LVREN=0, disable LVR.

**LV DEN:** Enable/disable LVD.

LV DEN=1, enable LVD.

LV DEN=0, disable LVD.

**EIS:** External interrupt select bit

EIS=1, PB0 is external interrupt.

EIS=0, PB0 is GPIO.

**WDTEN:** Enable/disable WDT.

WDTEN=1, enable WDT.

WDTEN=0, disable WDT.

### 3.1.9 BWUCON (PortB Wake-up Control Register)

| Name          | SFR Type | Addr. | Bit7  | Bit6  | Bit5  | Bit4  | Bit3  | Bit2  | Bit1  | Bit0  |
|---------------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| BWUCON        | R        | 0x9   | WUPB7 | WUPB6 | WUPB5 | WUPB4 | WUPB3 | WUPB2 | WUPB1 | WUPB0 |
| R/W Property  |          |       | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial Value |          |       | 0     | 0     | 0     | 0     | 0     | 0     | 0     | 0     |

**WUPBx:** Enable/disable PBx wake-up function,  $0 \leq x \leq 7$ .

WUPBx=1, enable PBx wake-up function.

WUPBx=0, disable PBx wake-up function.

### 3.1.10 PCHBUF (High Byte of PC)

| Name          | SFR Type | Addr. | Bit7 | Bit6     | Bit5 | Bit4 | Bit3 | Bit2 | Bit1        | Bit0 |
|---------------|----------|-------|------|----------|------|------|------|------|-------------|------|
| PCHBUF        | R        | 0xA   | -    | XSPD_STP | -    | -    | -    | GP5  | PCHBUF[1:0] |      |
| R/W Property  |          |       | -    | W        | -    | -    | -    | R/W  | W           |      |
| Initial Value |          |       | X    | 0        | X    | X    | X    | 0    | 00          |      |

**PCHBUF[1:0]:** Buffer of the 9<sup>th</sup> bit, 8<sup>th</sup> bit of PC.

**GP5:** General read/write register bit.

**XSPD\_STP:** Write 1 to stop crystal 32.768K speed-up function, write-only.

### 3.1.11 ABPLCON (PortA/PortB Pull-Low Resistor Control Register)

| Name          | SFR Type | Addr. | Bit7   | Bit6   | Bit5   | Bit4   | Bit3   | Bit2   | Bit1   | Bit0   |
|---------------|----------|-------|--------|--------|--------|--------|--------|--------|--------|--------|
| ABPLCON       | R        | 0xB   | /PLPB3 | /PLPB2 | /PLPB1 | /PLPB0 | /PLPA3 | /PLPA2 | /PLPA1 | /PLPA0 |
| R/W Property  |          |       | R/W    |        |        |        |        |        |        |        |
| Initial Value |          |       | 1      | 1      | 1      | 1      | 1      | 1      | 1      | 1      |

**/PLPAx:** Disable/enable PAx Pull-Low resistor,  $0 \leq x \leq 3$ .

/PLPAx=1, disable PAx Pull-Low resistor.

/PLPAx=0, enable PAx Pull-Low resistor.

**/PLPBx:** Disable/enable PBx Pull-Low resistor,  $0 \leq x \leq 3$ .

/PLPBx=1, disable PBx Pull-Low resistor.

/PLPBx=0, enable PBx Pull-Low resistor.

**3.1.12 BPHCON (PortB Pull-High Resistor Control Register)**

| Name          | SFR Type | Addr. | Bit7   | Bit6   | Bit5    | Bit4    | Bit3   | Bit2   | Bit1   | Bit0   |
|---------------|----------|-------|--------|--------|---------|---------|--------|--------|--------|--------|
| BPHCON        | R        | 0xC   | /PHPB7 | /PHPB6 | /PHPB5* | /PHPB4* | /PHPB3 | /PHPB2 | /PHPB1 | /PHPB0 |
| R/W Property  |          |       | R/W    |        |         |         |        |        |        |        |
| Initial Value |          |       | 1      | 1      | 1       | 1       | 1      | 1      | 1      | 1      |

**/PHPBx:** Disable/enable PBx Pull-High resistor,  $0 \leq x \leq 7$ .

/PHPBx=1, disable PBx Pull-High resistor.

/PHPBx=0, enable PBx Pull-High resistor.

**\*Note:** When PB4 and PB5 are used as crystal oscillator pads, the Pull-High resistor should not be enabled, or the oscillation may fail.

**3.1.13 INTE (Interrupt Enable Register)**

| Name          | SFR Type | Addr. | Bit7 | Bit6  | Bit5 | Bit4  | Bit3 | Bit2  | Bit1 | Bit0 |
|---------------|----------|-------|------|-------|------|-------|------|-------|------|------|
| INTE          | R        | 0xE   | -    | WDTIE | -    | LVDIE | T1IE | INTIE | PBIE | T0IE |
| R/W Property  |          |       | -    | R/W   | -    | R/W   | R/W  | R/W   | R/W  | R/W  |
| Initial Value |          |       | X    | 0     | X    | 0     | 0    | 0     | 0    | 0    |

**T0IE:** Timer0 overflow interrupt enable bit.

T0IE=1, enable Timer0 overflow interrupt.

T0IE=0, disable Timer0 overflow interrupt.

**PBIE:** PortB input change interrupt enable bit.

PBIE=1, enable PortB input change interrupt.

PBIE=0, disable PortB input change interrupt.

**INTIE:** External interrupt enable bit.

INTIE=1, enable external interrupt.

INTIE=0, disable external interrupt.

**T1IE:** Timer1 underflow interrupt enable bit.

T1IE=1, enable Timer1 underflow interrupt.

T1IE=0, disable Timer1 underflow interrupt.

**LVDIE:** Low-voltage detector interrupt enable bit.

LVDIE=1, enable low-voltage detector interrupt.

LVDIE=0, disable low-voltage detector interrupt.

**WDTIE:** WDT timeout interrupt enable bit.

WDTIE=1, enable WDT timeout interrupt.

WDTIE=0, disable WDT timeout interrupt.

### 3.1.14 INTF (Interrupt Flag Register)

| Name                 | SFR Type | Addr. | Bit7 | Bit6  | Bit5 | Bit4  | Bit3 | Bit2  | Bit1 | Bit0 |
|----------------------|----------|-------|------|-------|------|-------|------|-------|------|------|
| INTF                 | R        | 0xF   | -    | WDTIF | -    | LVDIF | T1IF | INTIF | PBIF | T0IF |
| R/W Property         |          |       | -    | R/W   | -    | R/W   | R/W  | R/W   | R/W  | R/W  |
| Initial Value(note*) |          |       | X    | 0     | X    | 0     | 0    | 0     | 0    | 0    |

**T0IF:** Timer0 overflow interrupt flag bit.

T0IF=1, Timer0 overflow interrupt is occurred.

T0IF must be clear by firmware.

**PBIF:** PortB input change interrupt flag bit.

PBIF=1, PortB input change interrupt is occurred.

PBIF must be clear by firmware.

**INTIF:** External interrupt flag bit.

INTIF=1, external interrupt is occurred.

INTIF must be clear by firmware.

**T1IF:** Timer1 underflow interrupt flag bit.

T1IF=1, Timer1 underflow interrupt is occurred.

T1IF must be clear by firmware.

**LVDIF:** Low-voltage detector interrupt flag bit.

LVDIF=1, Low-voltage detector interrupt is occurred.

LVDIF must be clear by firmware.

**WDTIF:** WDT timeout interrupt flag bit.

WDTIF=1, WDT timeout interrupt is occurred.

WDTIF must be clear by firmware.

**Note:** When corresponding *INTE* bit is not enabled, the read interrupt flag is 0.

### 3.2 T0MD Register

T0MD is a readable/writeable register which is only accessed by instruction T0MD / T0MDR.

| Name                 | SFR Type | Addr. | Bit7   | Bit6   | Bit5 | Bit4 | Bit3   | Bit2        | Bit1 | Bit0 |
|----------------------|----------|-------|--------|--------|------|------|--------|-------------|------|------|
| T0MD                 | -        | -     | LCKTM0 | INTEDG | T0CS | T0CE | PS0WDT | PS0SEL[2:0] |      |      |
| R/W Property         |          |       | R/W    |        |      |      |        |             |      |      |
| Initial Value(note*) |          |       | 0      | 0      | 1    | 1    | 1      | 111         |      |      |

**PS0SEL[2:0]:** Prescaler0 dividing rate selection. The rate depends on Prescaler0 is assigned to Timer0 or WDT. When Prescaler0 is assigned to WDT, the dividing rate is dependent on which timeout mechanism is selected.

| PS0SEL[2:0] | Dividing Rate        |                         |                             |
|-------------|----------------------|-------------------------|-----------------------------|
|             | PS0WDT=0<br>(Timer0) | PS0WDT=1<br>(WDT Reset) | PS0WDT=1<br>(WDT Interrupt) |
| 000         | 1:2                  | 1:1                     | 1:2                         |
| 001         | 1:4                  | 1:2                     | 1:4                         |
| 010         | 1:8                  | 1:4                     | 1:8                         |
| 011         | 1:16                 | 1:8                     | 1:16                        |
| 100         | 1:32                 | 1:16                    | 1:32                        |
| 101         | 1:64                 | 1:32                    | 1:64                        |
| 110         | 1:128                | 1:64                    | 1:128                       |
| 111         | 1:256                | 1:128                   | 1:256                       |

Table 3 Prescaler0 Dividing Rate

**PS0WDT:** Prescaler0 assignment.

PS0WDT=1, Prescaler0 is assigned to WDT.

PS0WDT=0, Prescaler0 is assigned to Timer0.

**Note:** Always set PS0WDT and PS0SEL[2:0] before enabling watchdog or timer interrupt, or reset or interrupt may be falsely triggered.

**T0CE:** Timer0 external clock edge selection.

T0CE=1, Timer0 will increase one while high-to-low transition occurs on pin EX\_CK1.

T0CE=0, Timer0 will increase one while low-to-high transition occurs on pin EX\_CK1.

**Note:** T0CE is also applied to Low Oscillator Frequency as timer0 clock source condition.

**T0CS:** Timer0 clock source selection.

T0CS=1, External clock on pin EX\_CK1 or Low Oscillator Frequency (I\_LRC or E\_LXT) is selected.

T0CS=0, Instruction clock F<sub>INST</sub> is selected.

**INTEDG:** Edge selection of external interrupt.

INTEDG=1, INTIF will be set while rising edge occurs on pin PB0.

INTEDG=0, INTIF will be set while falling edge occurs on pin PB0.

**LCKTM0:** When T0CS=1, timer 0 clock source can be optionally selected to be low-frequency oscillator.

T0CS=0, Instruction clock F<sub>INST</sub> is selected as timer0 clock source.

T0CS=1, LCKTM0=0, external clock on pin EX\_CK1 is selected as timer0 clock source.

T0CS=1, LCKTM0=1, Low Oscillator Frequency (I\_LRC or E\_LXT, depends on configuration word Low Oscillator Frequency) output replaces pin EX\_CK1 as timer0 clock source.

**Note:** For more detail descriptions of timer0 clock source select, please see timer0 section.



### 3.3 F-page Special Function Register

#### 3.3.1 IOSTA (PortA I/O Control Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6 | Bit5 | Bit4 | Bit3  | Bit2  | Bit1  | Bit0  |
|---------------|----------|-------|------|------|------|------|-------|-------|-------|-------|
| IOSTA         | F        | 0x5   | -    | -    | -    | -    | IOPA3 | IOPA2 | IOPA1 | IOPA0 |
| R/W Property  |          |       | -    | -    | -    | -    | R/W   | R/W   | R/W   | R/W   |
| Initial Value |          |       | X    | X    | X    | X    | 1     | 1     | 1     | 1     |

**IOPAx:** PAx I/O mode selection,  $0 \leq x \leq 3$ .

IOPAx=1, PAx is input mode.

IOPAx=0, PAx is output mode.

#### 3.3.2 IOSTB (PortB I/O Control Register)

| Name          | SFR Type | Addr. | Bit7  | Bit6  | Bit5  | Bit4  | Bit3  | Bit2  | Bit1  | Bit0  |
|---------------|----------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| IOSTB         | F        | 0x6   | IOPB7 | IOPB6 | IOPB5 | IOPB4 | IOPB3 | IOPB2 | IOPB1 | IOPB0 |
| R/W Property  |          |       | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   | R/W   |
| Initial Value |          |       | 1     | 1     | 1     | 1     | 1     | 1     | 1     | 1     |

**IOPBx:** PBx I/O mode selection,  $0 \leq x \leq 7$ .

IOPBx=1, PBx is input mode.

IOPBx=0, PBx is output mode.

#### 3.3.3 PS0CV (Prescaler0 Counter Value Register)

| Name          | SFR Type | Addr. | Bit7       | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|------------|------|------|------|------|------|------|------|
| PS0CV         | F        | 0xA   | PS0CV[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | R          |      |      |      |      |      |      |      |
| Initial Value |          |       | 1          | 1    | 1    | 1    | 1    | 1    | 1    | 1    |

While reading PS0CV, it will get current value of Prescaler0 counter.

#### 3.3.4 BODCON (PortB Open-Drain Control Register)

| Name          | SFR Type | Addr. | Bit7  | Bit6  | Bit5  | Bit4  | Bit3 | Bit2  | Bit1  | Bit0  |
|---------------|----------|-------|-------|-------|-------|-------|------|-------|-------|-------|
| BODCON        | F        | 0xC   | ODPB7 | ODPB6 | ODPB5 | ODPB4 | GP3  | ODPB2 | ODPB1 | ODPB0 |
| R/W Property  |          |       | R/W   |       |       |       |      |       |       |       |
| Initial Value |          |       | 0     | 0     | 0     | 0     | 0    | 0     | 0     | 0     |

**ODPBx:** Enable/disable open-drain of PBx,  $x=0,1, 2, 4, 5, 6, 7$ .

ODPBx=1, enable open-drain of PBx.

ODPBx=0, disable open-drain of PBx.

**GP3:** General purpose register bit.

### 3.3.5 CMPCR (Comparator voltage select Control Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6    | Bit5    | Bit4     | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|------|---------|---------|----------|------|------|------|------|
| CMPCR         | F        | 0xE   | GP7  | RBIAS_H | RBIAS_L | CMPF_INV | PS1  | PS0  | NS1  | NS0  |
| R/W Property  |          |       | R/W  |         |         |          |      |      |      |      |
| Initial Value |          |       | 0    | 0       | 0       | 0        | 0    | 0    | 0    | 0    |

**NS[1:0]:** Comparator inverting input select.

| NS[1:0] | Inverting input |
|---------|-----------------|
| 00      | PA1             |
| 01      | PA3             |
| 10      | Bandgap (0.6V)  |
| 11      | Vref            |

**PS[1:0]:** Comparator non-inverting input select

| PS[1:0] | Non-inverting input |
|---------|---------------------|
| 00      | PA0                 |
| 01      | PA2                 |
| 10      | Vref                |
| 11      | ---                 |

**CMPF\_INV:** Comparator output inverse control bit.

CMPF\_INV = 1, Inverse comparator output.

CMPF\_INV = 0, Non-inverse comparator output.

**RBIAS\_L, RBIAS\_H:** Set corresponding voltage reference levels

*(please refer to chapter 3.11.1)*

**Note:** *RBIAS\_L and RBIAS\_H MUST be set as "00" before sleep mode or halt mode to prevent current leakage.*

### 3.3.6 PCON1 (Power Control Register1)

| Name          | SFR Type | Addr. | Bit7               | Bit6    | Bit5  | Bit4  | Bit3  | Bit2  | Bit1 | Bit0 |
|---------------|----------|-------|--------------------|---------|-------|-------|-------|-------|------|------|
| PCON1         | F        | 0xF   | GIE                | LVDOOUT | LVDS3 | LVDS2 | LVDS1 | LVDS0 | GP1  | T0EN |
| R/W Property  |          |       | R/W <sup>(1)</sup> | R       | R/W   | R/W   | R/W   | R/W   | R/W  | R/W  |
| Initial Value |          |       | 0                  | X       | 1     | 1     | 1     | 1     | 0    | 1    |

**T0EN:** Enable/disable Timer0.

T0EN=1, enable Timer0.

T0EN=0, disable Timer0.

**LVDS3~0:** Select one of the 14 LVD voltage.

| LVDS[3:0] | Voltage |
|-----------|---------|
| 0000      | ---     |
| 0001      | ---     |
| 0010      | 2.2V    |
| 0011      | 2.4V    |
| 0100      | 2.6V    |
| 0101      | 2.8V    |
| 0110      | 2.9V    |
| 0111      | 3.0V    |
| 1000      | 3.15V   |
| 1001      | 3.30V   |
| 1010      | 3.45V   |
| 1011      | 3.60V   |
| 1100      | 3.75V   |
| 1101      | 3.90V   |
| 1110      | 4.05V   |
| 1111      | 4.15V   |

Table 4 LVD voltage select

**LVDOOUT:** Low voltage detector output, read-only.

**GIE:** Global interrupt enable bit.

GIE=1, enable all unmasked interrupts.

GIE=0, disable all interrupts.

**GP1:** General purpose read/write register.

(1\*) : set by instruction ENI, clear by instruction DISI, read by instruction IOSTR.

### 3.4 S-page Special Function Register

#### 3.4.1 TMR1 (Timer1 Register)

| Name          | SFR Type | Addr. | Bit7      | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|-----------|------|------|------|------|------|------|------|
| TMR1          | S        | 0x0   | TMR1[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | R/W       |      |      |      |      |      |      |      |
| Initial Value |          |       | XXXXXXXX  |      |      |      |      |      |      |      |

When reading register TMR1, it will obtain current value of 8-bit down-count Timer1. When writing register TMR1, it will both write data to timer1 reload register and update Timer1 current content.

### 3.4.2 T1CR1 (Timer1 Control Register1)

| Name          | SFR Type | Addr. | Bit7    | Bit6    | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------|---------|------|------|------|------|------|------|
| T1CR1         | S        | 0x1   | PWM1OEN | PWM1OAL | -    | -    | -    | T1OS | T1RL | T1EN |
| R/W Property  |          |       | W       | W       | -    | -    | -    | R/W  | R/W  | R/W  |
| Initial Value |          |       | 0       | 0       | X    | X    | X    | 0    | 0    | 0    |

This register is used to configure Timer1 functionality.

**T1EN:** Enable/disable Timer1.

T1EN=1, enable Timer1.

T1EN=0, disable Timer1.

**T1RL:** Configure Timer1 down-count mechanism while Non-Stop mode is selected (T1OS=0).

T1RL=1, initial value is reloaded from reload register TMR1.

T1RL=0, continuous down-count from 0xFF when underflow is occurred.

**T1OS:** Configure Timer1 operating mode while underflow is reached.

T1OS=1, One-Shot mode. Timer1 will count once from the initial value to 0x00.

T1OS=0, Non-Stop mode. Timer1 will keep down-count after underflow.

| T1OS | T1RL | Timer1 Down-Count Functionality   |
|------|------|---|
| 0    | 0    | Timer1 will count from reload value down to 0x00.<br>When underflow is reached, 0xFF is reloaded and continues down-count.            |
| 0    | 1    | Timer1 will count from reload value down to 0x00.<br>When underflow is reached, reload value is reloaded and continues to down-count. |
| 1    | x    | Timer1 will count from initial value down to 0x00.<br>When underflow is reached, Timer1 will stop down-count.                         |

Table 5 Timer1 Functionality

**PWM1OAL:** Define PWM1 output active state.

PWM1OAL=1, PWM1 output is active low.

PWM1OAL=0, PWM1 output is active high.

**PWM1OEN:** Enable/disable PWM1 output.

PWM1OEN=1, PWM1 output will be present on PB6 or PB2.

PWM1OEN=0, PB6 or PB2 is GPIO.

### 3.4.3 T1CR2 (Timer1 Control Register2)

| Name  | SFR Type | Addr. | Bit7 | Bit6 | Bit5 | Bit4 | Bit3   | Bit2        | Bit1 | Bit0 |
|-------|----------|-------|------|------|------|------|--------|-------------|------|------|
| T1CR2 | S        | 0x2   | -    | -    | T1CS | T1CE | /PS1EN | PS1SEL[2:0] |      |      |

|               |   |   |     |     |     |     |     |     |
|---------------|---|---|-----|-----|-----|-----|-----|-----|
| R/W Property  | - | - | R/W | R/W | R/W | R/W | R/W | R/W |
| Initial Value | X | X | 1   | 1   | 1   | 1   | 1   | 1   |

This register is used to configure Timer1 functionality.

**PS1SEL[2:0]:** Prescaler1 dividing rate selection.

| PS1SEL[2:0] | Dividing Rate |
|-------------|---------------|
| 000         | 1:2           |
| 001         | 1:4           |
| 010         | 1:8           |
| 011         | 1:16          |
| 100         | 1:32          |
| 101         | 1:64          |
| 110         | 1:128         |
| 111         | 1:256         |

Table 6 Prescaler1 Dividing Rate

**Note:** Always set **PS1SEL[2:0]** at **/PS1EN=1**, or interrupt may be falsely triggered.

**/PS1EN:** Disable/enable Prescaler1.

/PS1EN=1, disable Prescaler1.

/PS1EN=0, enable Prescaler1.

**T1CE:** Timer1 external clock edge selection.

T1CE=1, Timer1 will decrease one while high-to-low transition occurs on pin EX\_CK1.

T1CE=0, Timer1 will decrease one while low-to-high transition occurs on pin EX\_CK1.

**T1CS:** Timer1 clock source selection.

T1CS=1, External clock on pin EX\_CK1 is selected.

T1CS=0, Instruction clock is selected.

### 3.4.4 PWM1DUTY (PWM1 Duty Register)

| Name          | SFR Type | Addr. | Bit7          | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------------|------|------|------|------|------|------|------|
| PWM1DUTY      | S        | 0x3   | PWM1DUTY[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | W             |      |      |      |      |      |      |      |
| Initial Value |          |       | XXXXXXXX      |      |      |      |      |      |      |      |

This register is write-only. After Timer1 is enabled and start down-count, PWM1 output will keep at inactive state. While Timer1 value is equal to PWM1DUTY, PWM1 output will become active state until underflow is occurred.

Moreover, the reload value of Timer1 stored on register TMR1 is used to define the PWM1 frame rate and register PWM1DUTY is used to define the duty cycle of PWM1.

### 3.4.5 PS1CV (Prescaler1 Counter Value Register)

| Name          | SFR Type | Addr. | Bit7       | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|------------|------|------|------|------|------|------|------|
| PS1CV         | S        | 0x4   | PS1CV[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | R          |      |      |      |      |      |      |      |
| Initial Value |          |       | 1          | 1    | 1    | 1    | 1    | 1    | 1    | 1    |

While reading PS1CV, it will get current value of Prescaler1 counter.

### 3.4.6 BZ1CR (Buzzer1 Control Register)

| Name          | SFR Type | Addr. | Bit7  | Bit6 | Bit5 | Bit4 | Bit3         | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|-------|------|------|------|--------------|------|------|------|
| BZ1CR         | S        | 0x5   | BZ1EN | -    | -    | -    | BZ1FSEL[3:0] |      |      |      |
| R/W Property  |          |       | W     | -    | -    | -    | W            |      |      |      |
| Initial Value |          |       | 0     | X    | X    | X    | 1            | 1    | 1    | 1    |

**BZ1FSEL[3:0]**: Frequency selection of BZ1 output.

| BZ1FSEL[3:0] | BZ1 Frequency Selection |               |
|--------------|-------------------------|---------------|
|              | Clock Source            | Dividing Rate |
| 0000         | Prescaler1 output       | 1:2           |
| 0001         |                         | 1:4           |
| 0010         |                         | 1:8           |
| 0011         |                         | 1:16          |
| 0100         |                         | 1:32          |
| 0101         |                         | 1:64          |
| 0110         |                         | 1:128         |
| 0111         |                         | 1:256         |
| 1000         | Timer1 output           | Timer1 bit 0  |
| 1001         |                         | Timer1 bit 1  |
| 1010         |                         | Timer1 bit 2  |
| 1011         |                         | Timer1 bit 3  |
| 1100         |                         | Timer1 bit 4  |
| 1101         |                         | Timer1 bit 5  |
| 1110         |                         | Timer1 bit 6  |
| 1111         |                         | Timer1 bit 7  |

Table 7 Buzzer1 Output Frequency Selection

**BZ1EN:** Enable/Disable BZ1 output.

BZ1EN=1, enable Buzzer1.

BZ1EN=0, disable Buzzer1.

### 3.4.7 IRCR (IR Control Register)

| Name          | SFR Type | Addr. | Bit7      | Bit6 | Bit5 | Bit4 | Bit3 | Bit2   | Bit1   | Bit0 |
|---------------|----------|-------|-----------|------|------|------|------|--------|--------|------|
| IRCR          | S        | 0x6   | IROSC358M | -    | -    | -    | -    | IRCSEL | IRF57K | IREN |
| R/W Property  |          |       | W         | -    | -    | -    | -    | W      | W      | W    |
| Initial Value |          |       | 0         | X    | X    | X    | X    | 0      | 0      | 0    |

**IREN:** Enable/Disable IR carrier output.

IREN=1, enable IR carrier output.

IREN=0, disable IR carrier output.

**IRF57K:** Selection of IR carrier frequency.

IRF57K=1, IR carrier frequency is 57KHz.

IRF57K=0, IR carrier frequency is 38KHz.

**IRCSEL:** Polarity selection of IR carrier.

IRCSEL=0, IR carrier will be generated when I/O pin data is 1.

IRCSEL=1, IR carrier will be generated when I/O pin data is 0.

**IROSC358M:** When external crystal is used, this bit is determined according to what kind of crystal is used.

This bit is ignored if internal high frequency oscillation is used.

IROSC358M=1, crystal frequency is 3.58MHz.

IROSC358M=0, crystal frequency is 455KHz.

**Note:**

**1. Only high oscillation ( $F_{HOSC}$ ) (See section 3.14) can be used as IR clock source.**

**2. Division ratio for different oscillation type.**

| OSC. Type      | 57KHz | 38KHz | Conditions   |
|----------------|-------|-------|--|
| High IRC(4MHz) | 64    | 96    | HIRC mode (the input to IR module is set to 4MHz no matter what system clock is) |
| Xtal 3.58MHz   | 64    | 96    | Xtal mode & IROSC358M=1  |
| Xtal 455KHz    | 8     | 12    | Xtal mode & IROSC358M=0  |

Table 8 Division ratio for different oscillation type

### 3.4.8 TBHP (Table Access High Byte Address Pointer Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2  | Bit1  | Bit0  |
|---------------|----------|-------|------|------|------|------|------|-------|-------|-------|
| TBHP          | S        | 0x7   | -    | -    | -    | -    | -    | TBHP2 | TBHP1 | TBHP0 |
| R/W Property  |          |       | -    | -    | -    | -    | -    | R/W   | R/W   | R/W   |
| Initial Value |          |       | X    | X    | X    | X    | X    | X     | X     | X     |

When instruction CALLA, GOTOA or TABLEA is executed, the target address is constituted by TBHP[2:0] and ACC. ACC is the Low Byte of PC[9:0] and TBHP[1:0] is the high byte of PC[9:0]. TBHP[2] is general register for NY8A053E.

### 3.4.9 TBHD (Table Access High Byte Data Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6 | Bit5  | Bit4  | Bit3  | Bit2  | Bit1  | Bit0  |
|---------------|----------|-------|------|------|-------|-------|-------|-------|-------|-------|
| TBHD          | S        | 0x8   | -    | -    | TBHD5 | TBHD4 | TBHD3 | TBHD2 | TBHD1 | TBHD0 |
| R/W Property  |          |       | -    | -    | R     | R     | R     | R     | R     | R     |
| Initial Value |          |       | X    | X    | X     | X     | X     | X     | X     | X     |

When instruction TABLEA is executed, high byte of content of addressed ROM is loaded into TBHD[5:0] register. The Low Byte of content of addressed ROM is loaded to ACC.

### 3.4.10 P2CR1 (PWM2 Control Register1)

| Name          | SFR Type | Addr. | Bit7    | Bit6    | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------|---------|------|------|------|------|------|------|
| P2CR1         | S        | 0xA   | PWM2OEN | PWM2OAL | -    | -    | -    | -    | -    | -    |
| R/W Property  |          |       | R/W     | R/W     | -    | -    | -    | -    | -    | -    |
| Initial Value |          |       | 0       | 0       | X    | X    | X    | X    | X    | X    |

**PWM2OAL:** Define PWM2 output active state.

PWM2OAL=1, PWM2 output is active low.

PWM2OAL=0, PWM2 output is active high.

**PWM2OEN:** Enable/disable PWM2 output.

PWM2OEN=1, PWM2 output will be present on PA1 or PA0.

PWM2OEN=0, PA1 or PA0 is GPIO.

### 3.4.11 PWM2DUTY (PWM2 Duty Register)

| Name          | SFR Type | Addr. | Bit7          | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------------|------|------|------|------|------|------|------|
| PWM2DUTY      | S        | 0xC   | PWM2DUTY[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | W             |      |      |      |      |      |      |      |
| Initial Value |          |       | XXXXXXXX      |      |      |      |      |      |      |      |

The reload value of 10-bit Timer1 stored on registers TMRH[5:4] and TMR1[7:0] is used to define the PWM2 frame rate, and registers TMRH[3:2] and PWM2DUTY[7:0] is used to define the duty cycle of PWM2.



### 3.4.12 TMRH (Timer High Byte Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6 | Bit5  | Bit4  | Bit3       | Bit2       | Bit1       | Bit0       |
|---------------|----------|-------|------|------|-------|-------|------------|------------|------------|------------|
| TMRH          | S        | 0xD   | -    | -    | TMR19 | TMR18 | PWM2 DUTY9 | PWM2 DUTY8 | PWM1 DUTY9 | PWM1 DUTY8 |
| R/W Property  |          |       | -    | -    | R/W   | R/W   | R/W        | R/W        | R/W        | R/W        |
| Initial Value |          |       | X    | X    | X     | X     | X          | X          | X          | X          |

**TMR19~8:** Timer1 MSB 2 bits. Write these 2 bits will overwrite the 10-bit Timer1 load value of bit 9 and 8.

Read these 2 bits will get the Timer1 bit9-8 current value.

**PWM2DUTY9~8:** PWM2 duty data MSB 2 bits.

**PWM1DUTY9~8:** PWM1 duty data MSB 2 bits.

### 3.4.13 TM34RH (Timer3 High Byte Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6 | Bit5 | Bit4 | Bit3       | Bit2       | Bit1       | Bit0       |
|---------------|----------|-------|------|------|------|------|------------|------------|------------|------------|
| TM34RH        | S        | 0xE   | -    | -    | -    | -    | PWM4 DUTY9 | PWM4 DUTY8 | PWM3 DUTY9 | PWM3 DUTY8 |
| R/W Property  |          |       | -    | -    | -    | -    | R/W        | R/W        | R/W        | R/W        |
| Initial Value |          |       | X    | X    | X    | X    | X          | X          | X          | X          |

**PWM3DUTY9~8:** PWM3 duty data MSB 2 bits.

**PWM4DUTY9~8:** PWM4 duty data MSB 2 bits.

### 3.4.14 OSCCR (Oscillation Control Register)

| Name          | SFR Type | Addr. | Bit7 | Bit6   | Bit 5 | Bit 4 | Bit3      | Bit2    | Bit1    | Bit0 |
|---------------|----------|-------|------|--------|-------|-------|-----------|---------|---------|------|
| OSCCR         | S        | 0xF   | -    | CMPOEN | -     | -     | OPMD[1:0] | STPHOSC | SELHOSC |      |
| R/W Property  |          |       | -    | R/W    | -     | -     | R/W       | R/W     | R/W     |      |
| Initial Value |          |       | X    | 0      | X     | X     | 00        | 0       | 1       |      |

**SELHOSC:** Selection of system oscillation ( $F_{osc}$ ).

SELHOSC=1,  $F_{osc}$  is high-frequency oscillation ( $F_{Hosc}$ ).

SELHOSC=0,  $F_{osc}$  is low-frequency oscillation ( $F_{Losc}$ ).

**STPHOSC:** Disable/enable high-frequency oscillation ( $F_{Hosc}$ ).

STPHOSC=1,  $F_{Hosc}$  will stop oscillation and be disabled.

STPHOSC=0,  $F_{Hosc}$  keep oscillation.

**OPMD[1:0]:** Selection of operating mode.

| OPMD[1:0] | Operating Mode |
|-----------|----------------|
| 00        | Normal mode    |
| 01        | Halt mode      |

| OPMD[1:0] | Operating Mode |
|-----------|----------------|
| 10        | Standby mode   |
| 11        | reserved       |

Table 9 Selection of Operating Mode by OPMD[1:0]

**Note:** *STPHOSC cannot be changed with SELHOSC or OPMD at the same time. STPHOSC cannot be changed with OPMD at the same time during SELHOSC=1.*

**CMPOEN:** Enable/Disable comparator output to pad PB1.

CMPOEN=1, Enable comparator output to pad PB1.

CMPOEN=0, Disable comparator output to pad PB1.

**Note:** *Comparator output to pad PB1 has higher priority than PWM3/IR.*

### 3.4.15 P3CR1 (PWM3 Control Register1)

| Name          | SFR Type | Addr. | Bit7    | Bit6    | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------|---------|------|------|------|------|------|------|
| P3CR1         | S        | 0x11  | PWM3OEN | PWM3OAL | -    | -    | -    | -    | -    | -    |
| R/W Property  |          |       | R/W     | R/W     | -    | -    | -    | -    | -    | -    |
| Initial Value |          |       | 0       | 0       | X    | X    | X    | X    | X    | X    |

**PWM3OAL:** Define PWM3 output active state.

PWM3OAL=1, PWM3 output is active low.

PWM3OAL=0, PWM3 output is active high.

**PWM3OEN:** Enable/disable PWM3 output.

PWM3OEN=1, PWM3 output will be present on PA3 or PB1.

PWM3OEN=0, PA3 or PB1 is GPIO.

### 3.4.16 PWM3DUTY (PWM3 Duty Register)

| Name          | SFR Type | Addr. | Bit7          | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------------|------|------|------|------|------|------|------|
| PWM3DUTY      | S        | 0x13  | PWM3DUTY[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | W             |      |      |      |      |      |      |      |
| Initial Value |          |       | XXXXXXXX      |      |      |      |      |      |      |      |

The reload value of 10-bit Timer1 stored on registers TMRH[5:4] and TMR1[7:0] is used to define the PWM3 frame rate, and registers TM34RH[1:0] and PWM3DUTY[7:0] is used to define the duty cycle of PWM3.

### 3.4.17 P4CR1 (PWM4 Control Register1)

| Name          | SFR Type | Addr. | Bit7    | Bit6    | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------|---------|------|------|------|------|------|------|
| P4CR1         | S        | 0x16  | PWM4OEN | PWM4OAL | -    | -    | -    | -    | -    | -    |
| R/W Property  |          |       | R/W     | R/W     | -    | -    | -    | -    | -    | -    |
| Initial Value |          |       | 0       | 0       | X    | X    | X    | X    | X    | X    |

**PWM4OAL:** Define PWM4 output active state.

PWM4OAL=1, PWM4 output is active low.

PWM4OAL=0, PWM4 output is active high.

**PWM4OEN:** Enable/disable PWM4 output.

PWM4OEN=1, PWM4 output will be present on PA2 or PB0.

PWM4OEN=0, PA2 or PB0 is GPIO.

### 3.4.18 PWM4DUTY (PWM4 Duty Register)

| Name          | SFR Type | Addr. | Bit7          | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 | Bit0 |
|---------------|----------|-------|---------------|------|------|------|------|------|------|------|
| PWM4DUTY      | S        | 0x18  | PWM4DUTY[7:0] |      |      |      |      |      |      |      |
| R/W Property  |          |       | W             |      |      |      |      |      |      |      |
| Initial Value |          |       | XXXXXXXX      |      |      |      |      |      |      |      |

The reload value of 10-bit Timer1 stored on registers TMRH[5:4] and TMR1[7:0] is used to define the PWM4 frame rate, and registers TM34RH[3:2] and PWM4DUTY[7:0] is used to define the duty cycle of PWM4.

## 3.5 I/O Port

NY8A053E provides 12 I/O pins which are PA[3:0] and PB[7:0]. User can read/write these I/O pins through registers PORTA and PORTB respectively. Each I/O pin has a corresponding register bit to define it is input pin or output pin. Register IOSTA[3:0] define the input/output direction of PA[3:0]. Register IOSTB[7:0] define the input/output direction of PB[7:0].

When an I/O pin is configured as input pin, it may have Pull-High resistor or Pull-Low resistor which is enabled or disabled through registers. Register ABPLCON[3:0] are used to enable or disable Pull-Low resistor of PA[3:0]. Register BPHCON[7:0] are used to enable or disable Pull-High resistor of PB[7:0]. Register ABPLCON[7:4] are used to enable or disable Pull-Low resistor of PB[3:0].

When an I/O pin is configured as output pin, there is a corresponding and individual register to select as Open-Drain output pin. Register BODCON[7:0] determine PB[7:0] is Open-Drain or not. (Except PB[3], which is always in open-drain mode when configured as output port.)

The summary of Pad I/O feature is listed in the table below.

| Feature |                    | PA[3:0] | PB[2:0] | PB[3]  | PB[7:4] |
|---------|--------------------|---------|---------|--------|---------|
| Input   | Pull-High Resistor | X       | V       | V      | V       |
|         | Pull-Low Resistor  | V       | V       | V      | X       |
| Output  | Open-Drain         | X       | V       | always | V       |

Table 10 Summary of Pad I/O Feature

The level change on each I/O pin of PB may generate interrupt request. Register BWUCON[7:0] will select which I/O pin of PB may generate this interrupt. As long as any pin of PB is selected by corresponding bit of BWUCON, the register bit PBIF (INTF[1]) will set to 1 if there is a level change occurred on any selected pin. An interrupt request will occur and interrupt service routine will be executed if register bit PBIE (INTE[1]) and GIE (PCON1[7]) are both set to 1.

There is one external interrupt provided by NY8A053E. When register bit EIS (PCON[6]) is set to 1, PB0 is used as input pin for external interrupt.

**Note: When PB0 is both set as level change operation and external interrupt, the external interrupt will have higher priority, and the PB0 level change operation will be disabled. But PB7~PB1 level change function are not affected.**

NY8A053E can provide IR carrier generation. IR carrier generation is enabled by register bit IREN (IRCR[0]) and carrier will be present on a PB1 pin. Configuration word IR Current determines sink current value of IR carrier.

PB3 can be used as external reset input determined by a configuration word. When an active-low signal is applied to PB3, it will cause NY8A053E to enter reset process.

When external crystal (E\_HXT, E\_XT or E\_LXT) is adopted for high oscillation or low oscillation according to setting of configuration words, PB5 will be used as crystal input pin (Xin) and PB4 will be used as crystal output pin (Xout).

When I\_HRC or I\_LRC mode is selected as system oscillation and E\_HXT, E\_XT or E\_LXT is not adopted, instruction clock is observable on PB4 if a configuration word is enabled.

Moreover, PB2 can be timer 0 external clock source EX\_CK1 if T0MD T0CS=1 and LCK\_TM0=0. PB2 can be timer 1 external clock source if T1CS=1.

Moreover, PB6 or PB2 can be PWM1 output and PB7 or PB2 can be Buzzer1 output. If T1CR1[7] PWM1OEN=1, PB6 or PB2 can be PWM1 output. And if BZ1CR[7] BZ1EN=1, PB7 or BP2 can be Buzzer1 output.

### 3.5.1 Block Diagram of IO Pins

- IO\_SEL: set pad input or output
- WRITE\_EN: write data to pad.
- READ\_EN: read pad.
- PULLDOWN\_EN: enable Pull-Low.
- RD\_TYPE : select read pin or read latch.

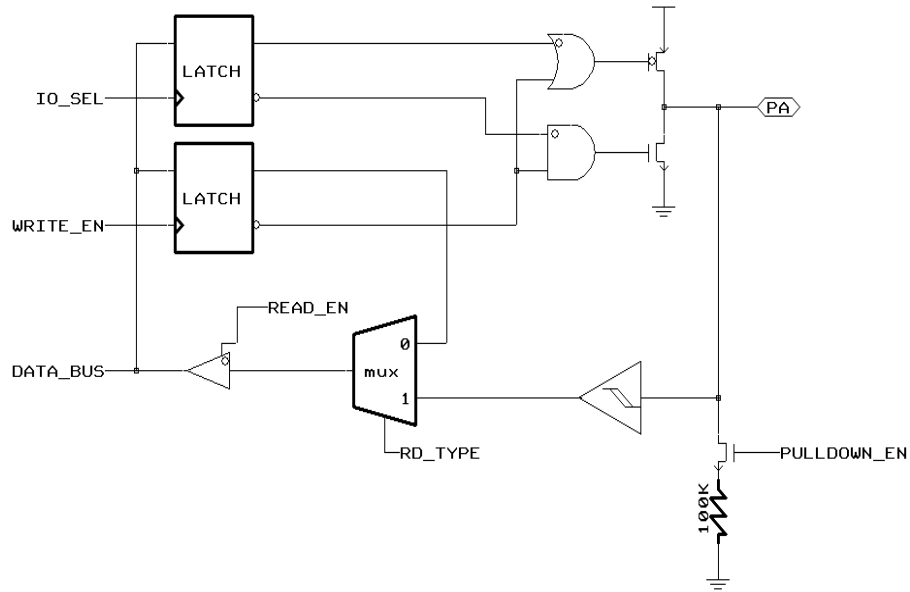


Figure 5 Block Diagram of PA[3:0]

- OD\_EN: open-drain enable.
- IO\_SEL: set pad input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad to DATA\_BUS.
- PULLDOWN\_EN: enable Pull-Low.
- PULLUP\_ENB: enable pull high.
- WUBx: wake-up enable.
- INTEDGE: external interrupt edge select
- SET\_PBIF: set port change interrupt flag.
- RD\_TYPE: select read pin or read latch.

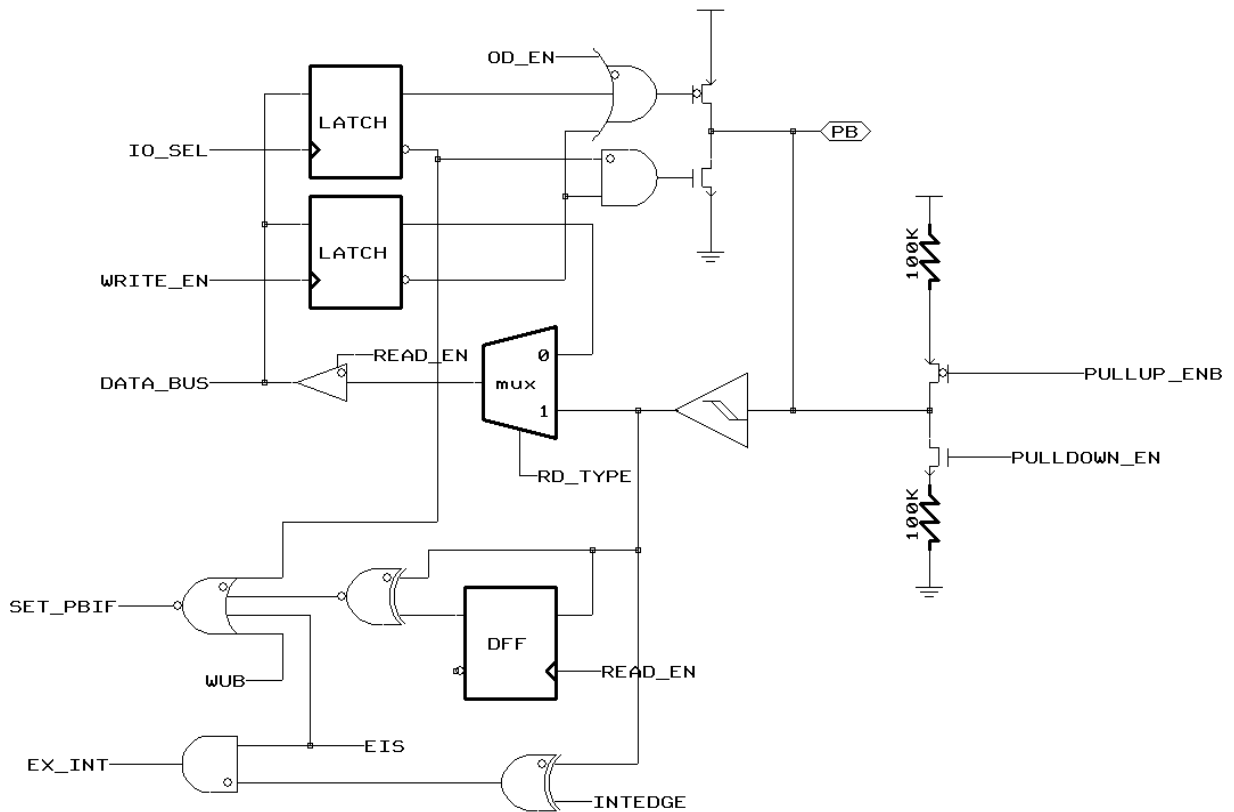


Figure 6 Block Diagram of PB0

- OD\_EN: open-drain enable.
- IREN: Enable IR.
- IO\_SEL: set pad input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad to DATA\_BUS.
- PULLDOWN\_EN: enable Pull-Low.
- PULLUP\_ENB: enable pull high.
- WUBx: wake-up enable.
- RD\_TYPE: read pin or read latch.
- SET\_PBIF: set port change interrupt flag.

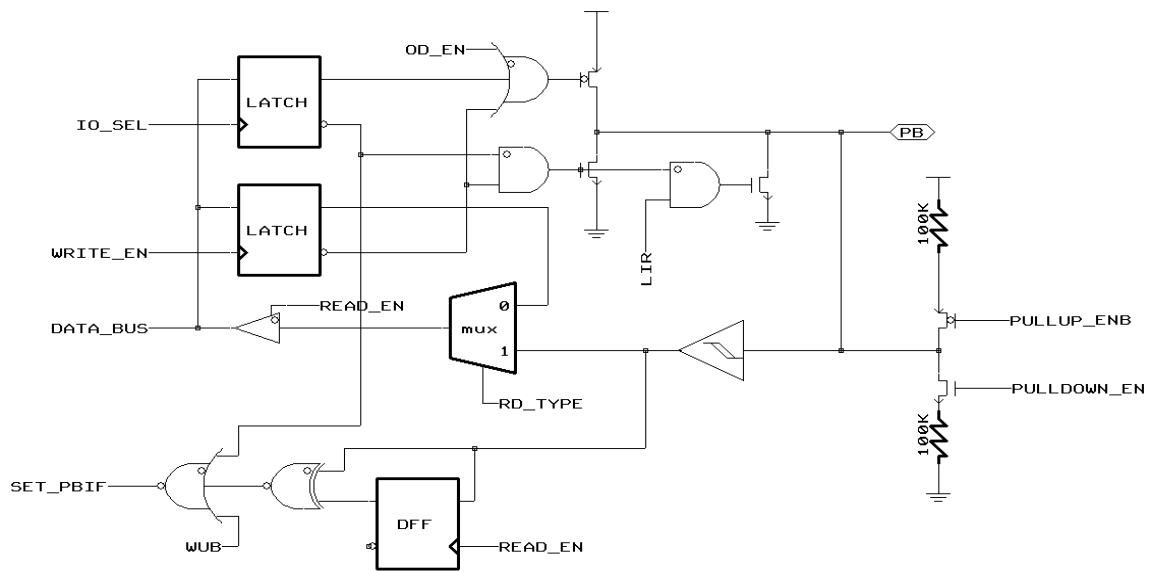


Figure 7 Block Diagram of PB1

- OD\_EN: open-drain enable.
- IO\_SEL: set pad input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad to DATA\_BUS.
- PULLDOWN\_EN: enable Pull-Low.
- PULLUP\_ENB: enable pull high.
- WUBx: wake-up enable.
- SET\_PBIF: set port change interrupt flag.
- RD\_TYPE: read pin or read latch.
- EX\_CKI: Timer0 EX CLOCK IN.

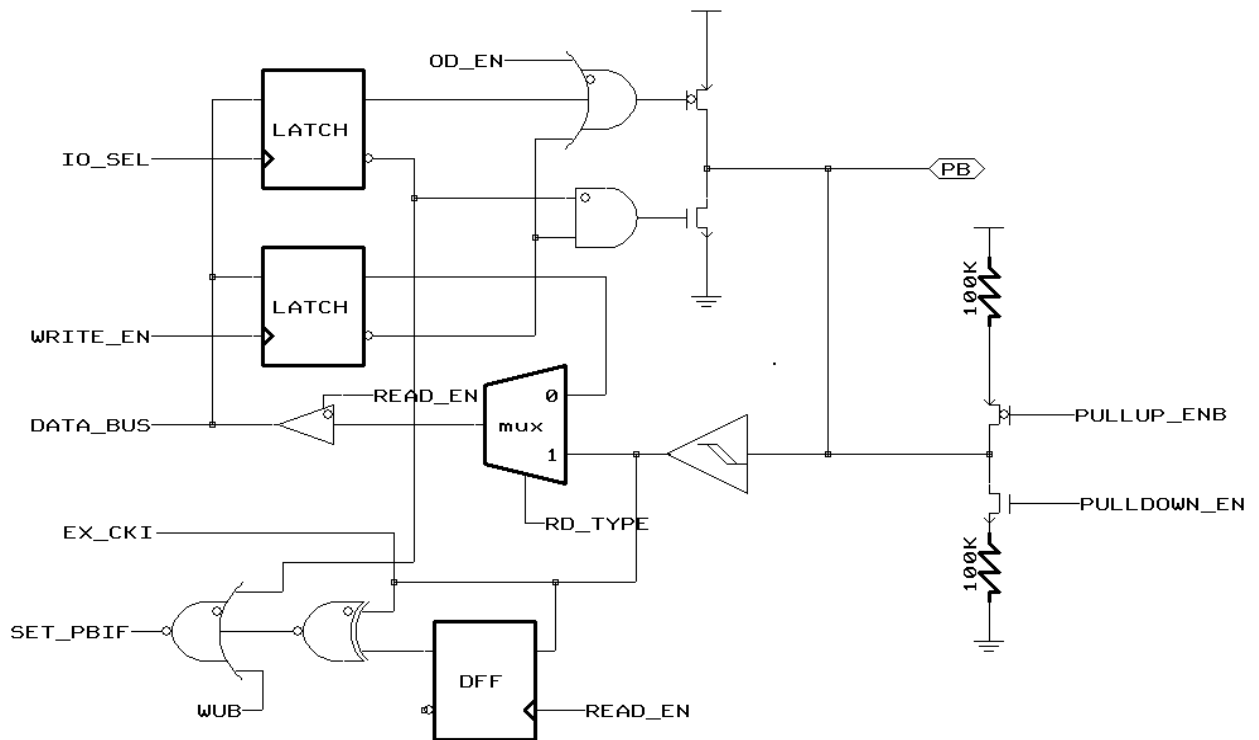


Figure 8 Block Diagram of PB2



- PB3\_RSTPAD: PB3 is set as Reset Pin by configuration word.
- IO\_SEL: set pad input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad to DATA\_BUS.
- PULLDOWN\_EN: enable Pull-Low.
- PULLUP\_ENB: enable pull high.
- WUBx: wake-up enable.
- SET\_PBIF: set port change interrupt flag.
- RD\_TYPE: read pin or read latch.
- RSTB\_IN: Reset signal with Schmitt trigger.

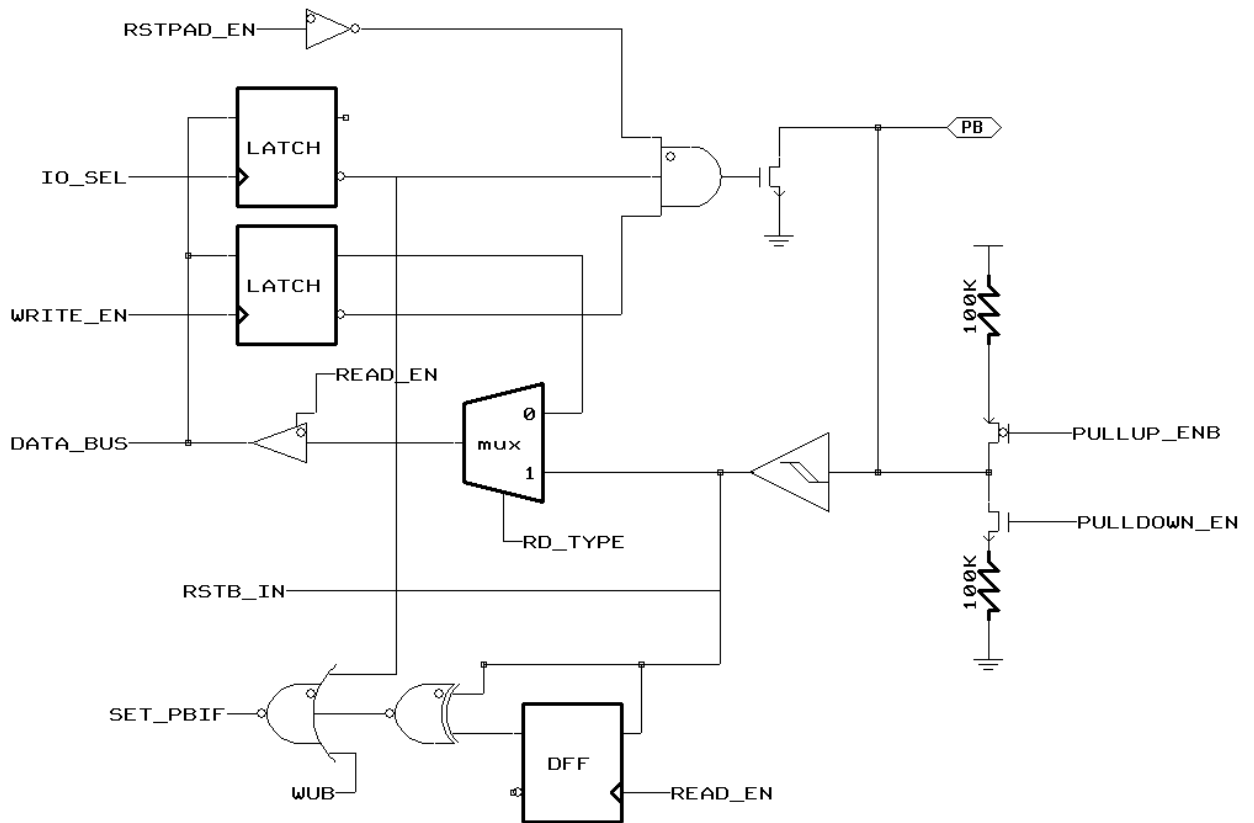


Figure 9 Block Diagram of PB3

- OD\_EN: open-drain enable.
- XTL\_EN: PB4, PB5 is set as Xtal Pin by configuration word.
- IO\_SEL: set pad input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad to DATA\_BUS.
- PULLUP\_ENB: enable pull high.
- WUBx: wake-up enable.
- RD\_TYPE: read pin or read latch
- SET\_PBIF: set port change interrupt flag.

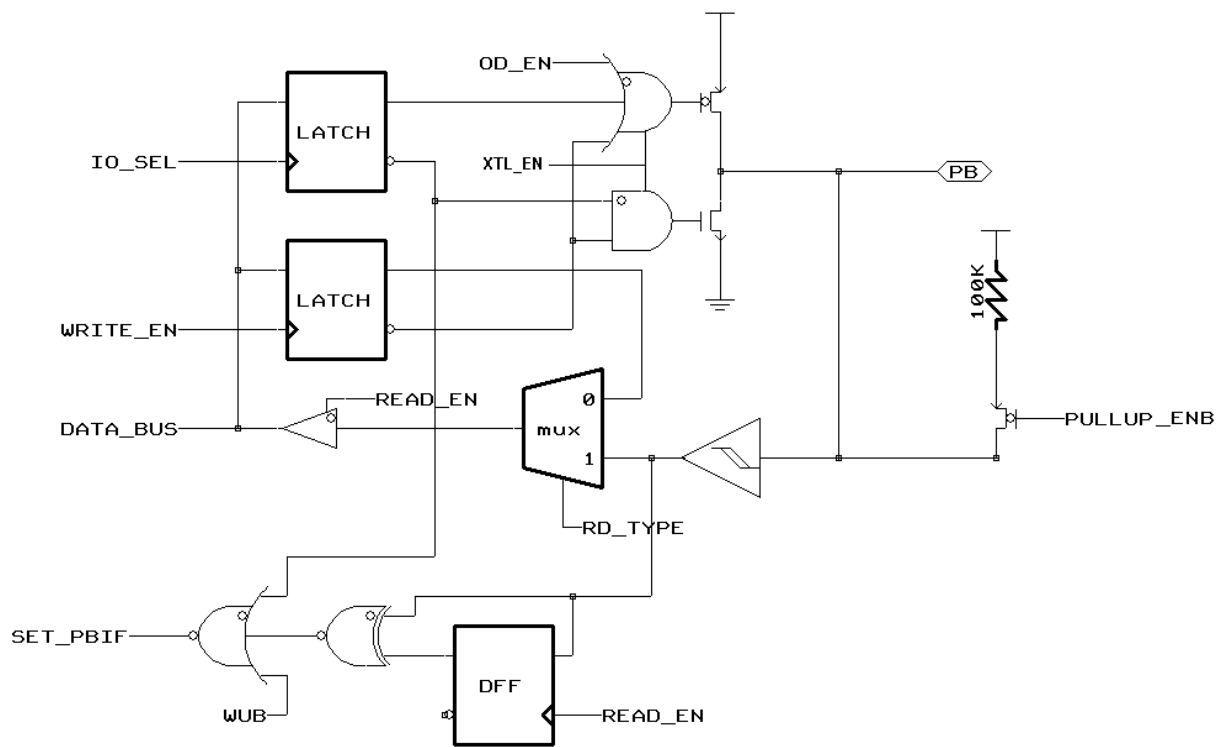


Figure 10 Block Diagram of PB5, PB4

- OD\_EN: open-drain enable.
- DATA\_BUS: PWM or Buzzer data.
- IO\_SEL: set pad input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad to DATA\_BUS.
- PULLUP\_ENB: enable pull high.
- WUB: wake-up enable.
- RD\_TYPE: read pin or read latch.
- SET\_PBIF: set port change interrupt flag.

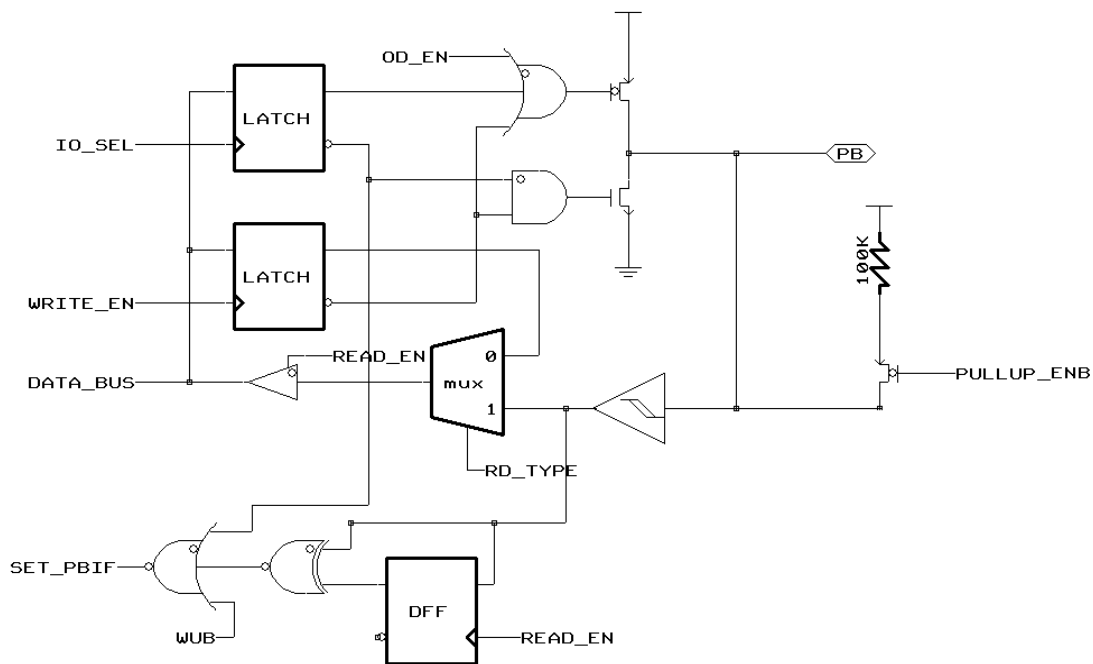


Figure 11 Block Diagram of PB7, PB6

### 3.6 Timer0

Timer0 is an 8-bit up-count timer and its operation is enabled by register bit T0EN (PCON1[0]). Writing to Timer0 will set its initial value. Reading from Timer0 will show its current count value.

The clock source to Timer0 can be from instruction clock, external pin EX\_CK1 or low speed clock Low Oscillator Frequency according to register bit T0CS and LCK\_TM0 (T0MD[5] and T0MD[7]). When T0CS is 0, instruction clock is selected as Timer0 clock source. When T0CS is 1 and LCK\_TM0 is 0, EX\_CK1 is selected as Timer0 clock source. When T0CS is 1 and LCK\_TM0 is 1 (and Timer0 source must set to 1), Low Oscillator Frequency (I\_LRC or E\_LXT, depends on configuration word) output is selected. Summarized table is shown below. (Also check Figure. 12)

| Timer0 clock source | T0CS | LCKTM0 | Timer0 source | Low Oscillator Frequency |
|---------------------|------|--------|---------------|--------------------------|
| Instruction clock   | 0    | X      | X             | X                        |
| EX_CK1              | 1    | 0      | X             | X                        |
|                     |      | X      | 0             |                          |
| E_LXT               | 1    | 1      | 1             | 1                        |
| I_LRC               | 1    | 1      | 1             | 0                        |

Table 11 Summary of Timer0 clock source control

Moreover the active edge of EX\_CK1 or Low Oscillator Frequency to increase Timer0 can be selected by register bit T0CE (T0MD[4]). When T0CE is 1, high-to-low transition on EX\_CK1 or Low Oscillator Frequency will increase Timer0. When T0CE is 0, low-to-high transition on EX\_CK1 or Low Oscillator Frequency will increase Timer0. When using Low Oscillator Frequency as timer0 clock source, it is suggested to use prescaler0 (see below descriptions) and the ratio set to more than 4, or missing count may happen.

Before Timer0 clock source is supplied to Timer0, it can be divided by Prescaler0 if register bit PS0WDT (T0MD[3]) is clear to 0. When writing 0 to PS0WDT by instruction, Prescaler0 is assigned to Timer0 and Prescaler0 will be clear after this instruction is executed. The dividing rate of Prescaler0 is determined by register bits PS0SEL[2:0] which is from 1:2 to 1:256.

When Timer0 is overflow, the register bit T0IF (INTF[0]) will be set to 1 to indicate Timer0 overflow event is occurred. If register bit T0IE (INTE[0]) and GIE are both set to 1, interrupt request will occur and interrupt service routine will be executed. T0IF will not be clear until firmware writes 0 to T0IF.

The block diagram of Timer0 and WDT is shown in the figure below.

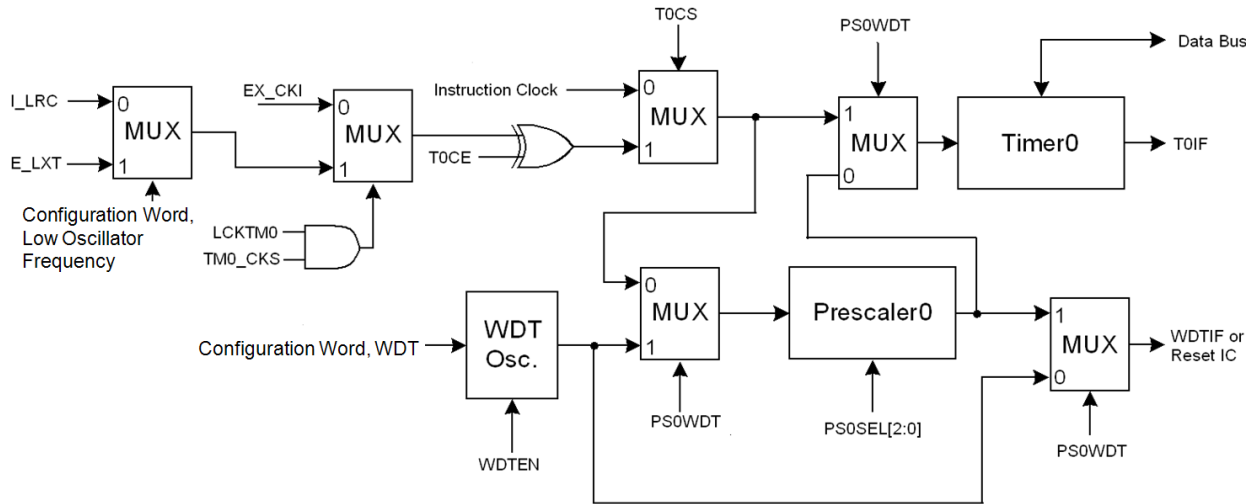


Figure 12 Block Diagram of Timer0 and WDT

### 3.7 Timer1/PWM1/Buzzer1

Timer1 is an 10-bit down-count timer with Prescaler1 whose dividing rate is programmable. The output of Timer1 can be used to generate PWM1 output and Buzzer1 output. A write to the Timer1 will both write to a timer1 reload register (T1rld) and timer1 counter. A read to the timer1 will show the content of the Timer1 current count value.

The block diagram of Timer1 is shown in the figure below.

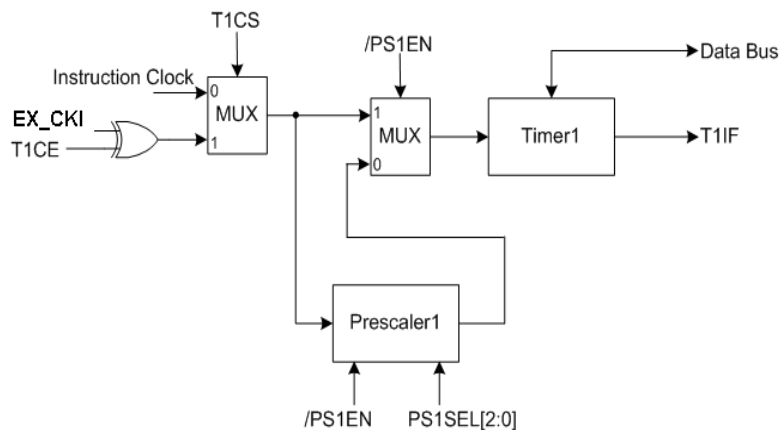


Figure 13 Block Diagram of Timer1

The operation of Timer1 can be enabled or disabled by register bit T1EN (T1CR1[0]). After Timer1 is enabled, its clock source can be instruction clock or pin EX\_CKI which is determined by register bit T1CS (T1CR2[5]). When T1CS is 1, EX\_CKI is selected as clock source. When T1CS is 0, instruction clock is selected as clock source. When EX\_CKI is selected, the active edge to decrease Timer1 is determined by register bit T1CE (T1CR2[4]). When T1CE is 1, high-to-low transition on EX\_CKI will decrease Timer1. When T1CE is 0, low-to-high transition

on EX\_CK1 will decrease Timer1. The selected clock source can be divided further by Prescaler1 before it is applied to Timer1. Prescaler1 is enabled by writing 0 to register bit /PS1EN (T1CR2[3]) and the dividing rate is from 1:2 to 1:256 determined by register bits PS1SEL[2:0] (T1CR2[2:0]). Current value of Prescaler1 can be obtained by reading register PS1CV.

Timer1 provide two kinds of operating mode: one is One-Shot mode and the other is Non-Stop mode. When register bit T1OS (T1CR1[2]) is 1, One-Shot mode is selected. Timer1 will count down once from initial value stored on register TMR1 to 0x00, i.e. underflow is occurred. When register bit T1OS (T1CR1[2]) is 0, Non-Stop mode is selected. When underflow is occurred, there are two selections to start next down-count which is determined by register bit T1RL (T1CR1[1]). When T1RL is 1, the initial value stored on register TMR1 will be restored and start next down-count from this initial value. When T1RL is 0, Timer1 will start next down-count from 0xFF.

When Timer1 is underflow, the register bit T1IF (INTF[3]) will be set to 1 to indicate Timer1 underflow event is occurred. If register bit T1IE (INTE[3]) and GIE are both set to 1, interrupt request will occur and interrupt service routine will be executed. T1IF will not be clear until firmware writes 0 to T1IF.

The timing chart of Timer1 is shown in the following figure.

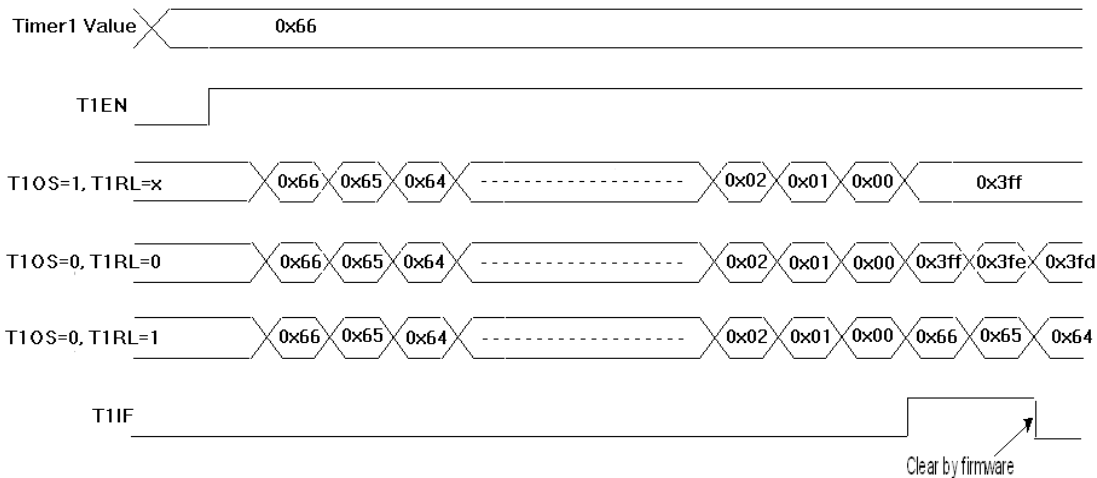


Figure 14 Timer1 Timing Chart

The PWM1 output can be available on I/O pin PB2 or PB6 when register bit PWM1OEN (T1CR1[7]) is set to 1. Moreover, PB2 or PB6 will become output pin automatically. The active state of PWM1 output is determined by register bit PWM1OAL (T1CR1[6]). When PWM1OAL is 1, PWM1 output is active low. When PWM1OAL is 0, PWM1 output is active high. Moreover, the duty cycle and frame rate of PWM1 are both programmable. The duty cycle is determined by registers TMRH[1:0] and PWM1DUTY[7:0]. When PWM1DUTY is 0, PWM1 output will be never active. When PWM1DUTY is 0x3FF, PWM1 output will be active for 1023 Timer1 input clocks. The frame rate is determined by TMRH[5:4] + TMR1[7:0] initial value. Therefore, PWM1DUTY value must be less than or equal to TMRH[5:4] + TMR1[7:0]. When user write PWM1DUTY, write PWM1DUTY[9:8] MSB 2

bits(TMRH[1:0]) first and write PWM1DUTY[7:0] second, PWM1 duty register will be updated after Timer1 overflow occurs. The block diagram of PWM1 is illustrated in the following figure.

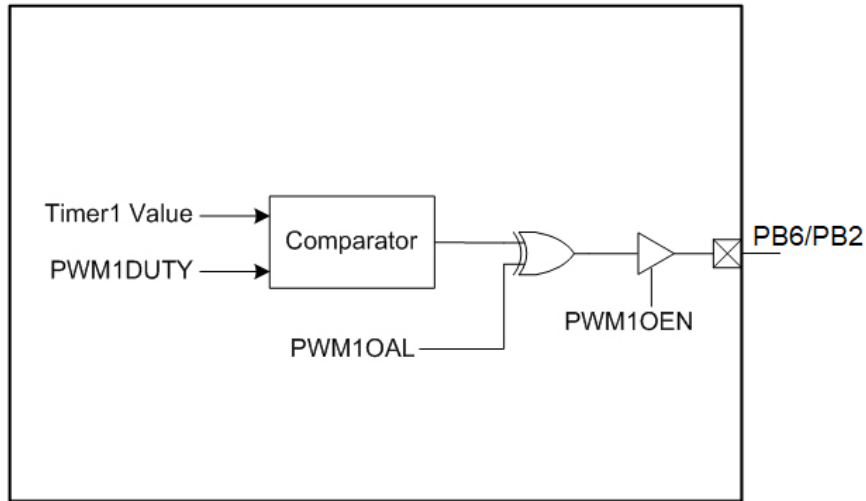


Figure 15 PWM1 Block Diagram

The Buzzer1 output (BZ1) can be available on I/O pin PB7 or PB2 when register bit BZ1EN (BZ1CR1[7]) is set to 1. Moreover, PB7 or PB2 will become output pin automatically. The frequency of BZ1 can be derived from Timer1 output or Prescaler1 output and dividing rate is determined by register bits BZ1FSEL[3:0] (BZ1CR[3:0]). When BZ1FSEL[3] is 0, Prescaler1 output is selected to generate BZ1 output. When BZ1FSEL[3] is 1, Timer1 output is selected to generate BZ1 output. The dividing rate can be from 1:2 to 1:256 in order to generate all kinds of frequency. The block diagram of Buzzer1 is illustrated in the following figure.

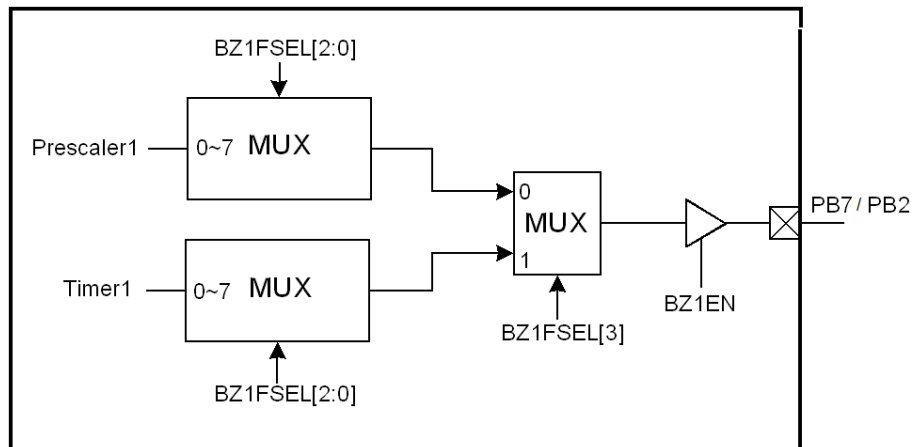


Figure 16 Buzzer1 Block Diagram

### 3.8 PWM2/3/4

The PWM2/3/4 output can be available on PortA or PortB when register bit PWMxOEN (x=2,3,4) is set to 1. Moreover, PortA or PortB will become output pin automatically. The active state of PWM2/3/4 output is determined by register bit PWMxOAL. When PWMxOAL is 1, PWM2/3/4 output is active low. When PWMxOAL is 0, PWM2/3/4 output is active high. Moreover, the duty cycle and frame rate of PWM2/3/4 are both

programmable. The duty cycle is determined by register PWMxDUTY. When PWMxDUTY is 0, PWM2/3/4 output will be never active. When PWMxDUTY is 0x3FF, PWM2/3/4 output will be active for 1023 Timer1 input clocks. The frame rate is determined by TMRH[5:4],TMR1[7:0] initial value. Therefore, PWMxDUTY value must be less than or equal to TMR1[9:0]. Besides, user needs to set P2CR1 because PWM2/3/4 are both shared with Timer1. When user write PWMxDUTY, write PWMxDUTY[9:8] MSB 2 bits first and write PWMxDUTY[7:0] second, PWM2/3/4 duty register will be updated after Timer1 overflow occurs. The block diagram of PWM2/3/4 is illustrated in the following figure.

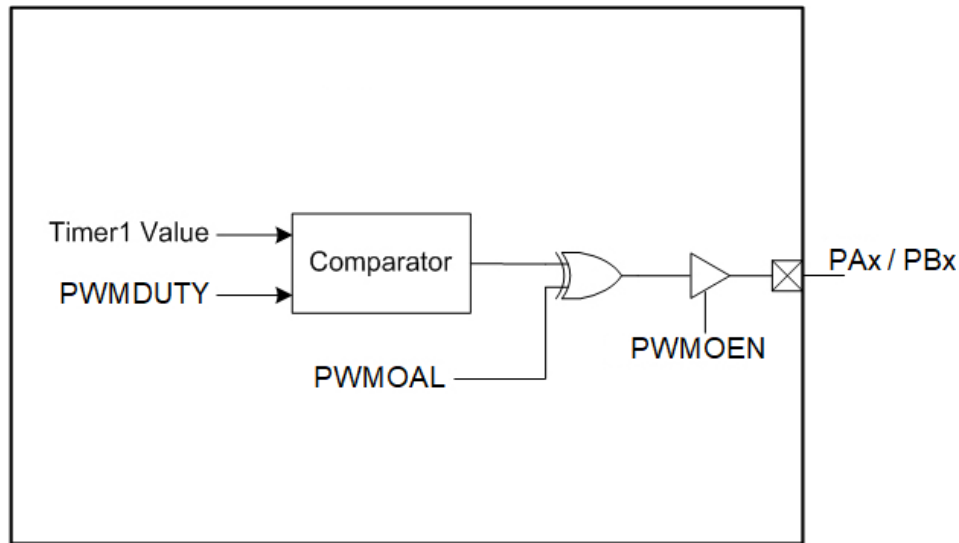


Figure 17 PWM2/3/4 Block Diagram

### 3.9 IR Carrier

The IR carrier will be generated after register bit IREN (IRCR[0]) is set to 1. Moreover, PB1 will become output pin automatically. When IREN is clear to 0, PB1 will become general I/O pin as it was configured.

The IR carrier frequency is selectable by register bit IRF57K (IRCR[1]). When IRF57K is 1, IR carrier frequency is 57KHz. When IRF57K is 0, IR carrier frequency is 38KHz. Because IR carrier frequency is derived from high frequency system oscillation  $F_{HOSC}$ , it is necessary to specify what frequency is used as system oscillation when external crystal is used. Register bit IROSC358M (IRCR[7]) is used to provide NY8A053E this information. When IROSC358M is 1, frequency of external crystal is 3.58MHz and when IROSC358M is 0, frequency of external crystal is 455KHz. When internal high frequency oscillation is adopted, this register will be ignored, and it will provide 4MHz clock to IR module.

The active state (polarity) of IR carrier is selectable according to PB1 output data. When register bit IRCSEL (IRCR[2]) is 1, IR carrier will be present on pin PB1 when its output data is 0. When register bit IRCSEL (IRCR[2]) is 0, IR carrier will be present on pin PB1 when its output data is 1. The polarity of IR carrier is shown in the following figure.



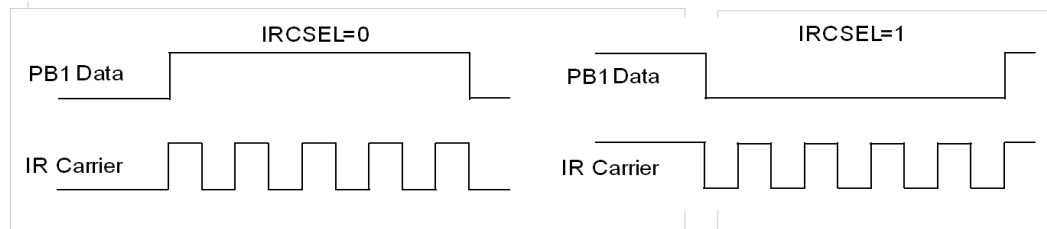


Figure 18 Polarity of IR Carrier vs. Output Data

### 3.10 Low Voltage Detector (LVD)

NY8A053E low voltage detector (LVD) built-in precise band-gap reference for accurately detecting  $V_{DD}$  level. If LVDEN (register PCON[5]) =1 and  $V_{DD}$  voltage value falls below LVD voltage which is selected by LVDS[3:0] as table shown below, the LVD output will become low. If the LVD interrupt is enabled, the LVD interrupt flag will be high and if GIE=1 it will force the program to execute interrupt service routine. Moreover, LVD real-state output can be polled by register PCON1[6]. The following is LVD block diagram:

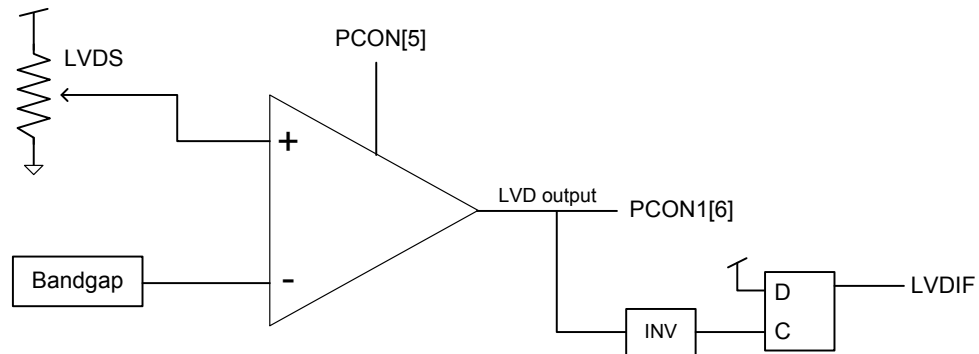


Figure 19 LVD block diagram

The following table is LVD voltage select table.

| LVDS[3:0] | Voltage |
|-----------|---------|
| 0000      | --      |
| 0001      | --      |
| 0010      | 2.2V    |
| 0011      | 2.4V    |
| 0100      | 2.6V    |
| 0101      | 2.8V    |
| 0110      | 2.9V    |
| 0111      | 3.0V    |
| 1000      | 3.15V   |
| 1001      | 3.30V   |
| 1010      | 3.45V   |
| 1011      | 3.60V   |
| 1100      | 3.75V   |
| 1101      | 3.90V   |

| LVDS[3:0] | Voltage |
|-----------|---------|
| 1110      | 4.05V   |
| 1111      | 4.15V   |

Table 12 LVD voltage select

**Note:**

**The hysteresis voltage (from low to high) of LVD is about 0.1V.**

**In battery charging applications (detected voltage is from low to high),**

**the LVD voltage select table should be as followed:**

| LVDS[3:0] | Voltage      |
|-----------|--------------|
| 0000      | --           |
| 0001      | --           |
| 0010      | (2.2+0.1) V  |
| 0011      | (2.4+0.1) V  |
| 0100      | (2.6+0.1) V  |
| 0101      | (2.8+0.1) V  |
| 0110      | (2.9+0.1) V  |
| 0111      | (3.0+0.1) V  |
| 1000      | (3.15+0.1) V |
| 1001      | (3.30+0.1) V |
| 1010      | (3.45+0.1) V |
| 1011      | (3.60+0.1) V |
| 1100      | (3.75+0.1) V |
| 1101      | (3.90+0.1) V |
| 1110      | (4.05+0.1) V |
| 1111      | (4.15+0.1) V |

The LVD control flow is as the following:

*Step1: Select LVD voltage by LVDS[3:0]*

*Step2: Set CMPCR = 0x0A*

*Step3: Set PCON[5]=1 (enable LVD)*

*Step4: Check LVD status by PCON1[6]*

**Note: If LVD voltage LVDS[3:0] is changed, user must wait at least 50us(@F<sub>Hosc</sub>=1MHz) to get correct LVD status by PCON1[6]**

### 3.11 Voltage Comparator

NY8A053E provides voltage comparator and internal reference voltage with various analog comparing mode.

The comparator non-inverting and inverting input can share with GPIO.

CMPEN (register PCON[2]) is used to enable and disable comparator. When CMPEN=0(default), comparator is disabled. When CMPEN=1, the comparator is enabled. In halt mode the comparator is disabled automatically.

The structure of comparator is shown in the following figure:

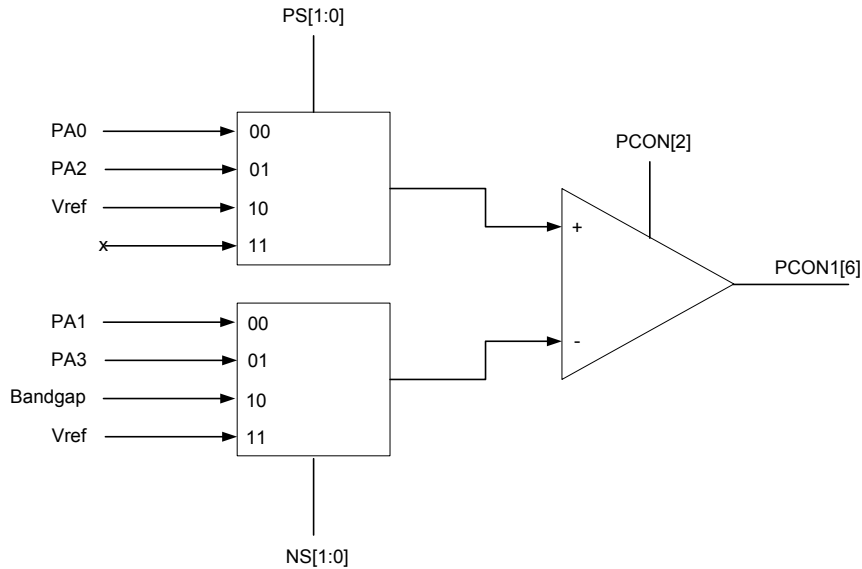


Figure 20 Comparator block diagram

### 3.11.1 Comparator Reference Voltage (Vref)

The internal reference voltage Vref is built by series resistance to provide different level of reference voltage. RBIAS\_H and RBIAS\_L are used to select the maximum and minimum values of Vref, and LVDS[3:0] are used to select one of 16 voltage levels.

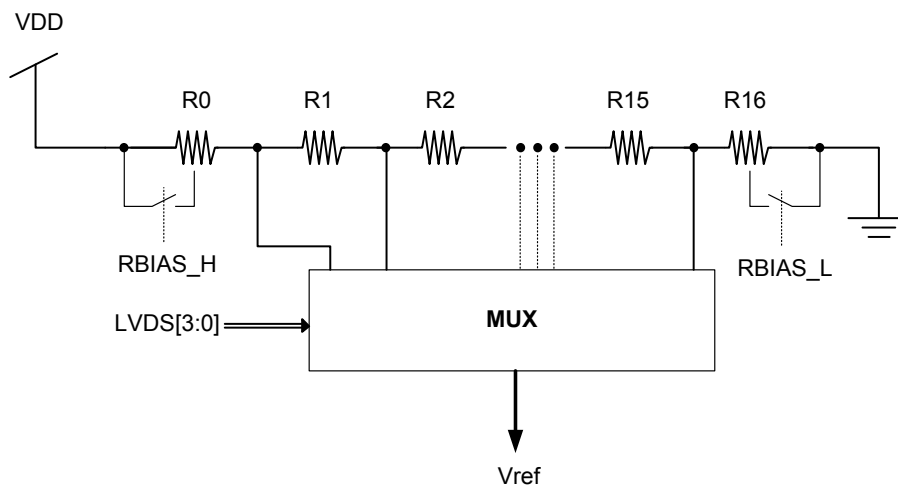


Figure 21 Vref hardware connection

The **Vref** is determined by RBIAS\_H, RBIAS\_L and LVDS[3:0]. The LVDS[3:0] is used to select one out of 16 reference voltages, the table shown below.

| LVDS[3:0] | RBIAS_H=1<br>RBIAS_L=0 | RBIAS_H=0<br>RBIAS_L=1 |
|-----------|------------------------|------------------------|
| 0000      | 67/128 V <sub>DD</sub> | 34/128 V <sub>DD</sub> |
| 0001      | 64/128 V <sub>DD</sub> | 32/128 V <sub>DD</sub> |
| 0010      | 59/128 V <sub>DD</sub> | 28/128 V <sub>DD</sub> |
| 0011      | 54/128 V <sub>DD</sub> | 25/128 V <sub>DD</sub> |
| 0100      | 50/128 V <sub>DD</sub> | 22/128 V <sub>DD</sub> |
| 0101      | 47/128 V <sub>DD</sub> | 20/128 V <sub>DD</sub> |
| 0110      | 45/128 V <sub>DD</sub> | 19/128 V <sub>DD</sub> |
| 0111      | 44/128 V <sub>DD</sub> | 17/128 V <sub>DD</sub> |
| 1000      | 42/128 V <sub>DD</sub> | 16/128 V <sub>DD</sub> |
| 1001      | 40/128 V <sub>DD</sub> | 15/128 V <sub>DD</sub> |
| 1010      | 38/128 V <sub>DD</sub> | 14/128 V <sub>DD</sub> |
| 1011      | 37/128 V <sub>DD</sub> | 13/128 V <sub>DD</sub> |
| 1100      | 35/128 V <sub>DD</sub> | 12/128 V <sub>DD</sub> |
| 1101      | 34/128 V <sub>DD</sub> | 11/128 V <sub>DD</sub> |
| 1110      | 33/128 V <sub>DD</sub> | 10/128 V <sub>DD</sub> |
| 1111      | 32/128 V <sub>DD</sub> | 10/128 V <sub>DD</sub> |

Table 13 The reference voltage Vref selection table

**Note: The deviation of Vref is ±0.1V.**

**Note: RBIAS\_L and RBIAS\_H MUST be set as "00" before sleep mode or halt mode to prevent current leakage.**

The non-inverting input of the comparator is determined by PS[1:0] (register CMPCR[3:2]).

The table is shown below

| PS[1:0] | Non-inverting input |
|---------|---------------------|
| 00      | PA0                 |
| 01      | PA2                 |
| 10      | Vref                |
| 11      | ---                 |

Table 14 Non-inverting input select

The inverting input of the comparator is determined by NS[1:0] (register CMPCR[1:0]).

The table is shown below

| NS[1:0] | Inverting input |
|---------|-----------------|
| 00      | PA1             |
| 01      | PA3             |
| 10      | Bandgap (0.65V) |
| 11      | Vref            |

Table 15 Inverting input select

There are two ways to get the comparator output result: one is through register polling, the other is through probing output pad.

Comparator output can be polled by LVDOOUT (register PCON1[6]).

To probe comparator output at output pad, set CMPOE (register OSCCR[6]) to 1, then PB1 will be the real-time state of the comparator output. It is noted that when CMPOE=1, the PWM3 function will be disabled if it is enabled.

### 3.12 Watch-Dog Timer (WDT)

There is an on-chip free-running oscillator in NY8A053E which is used by WDT. As this oscillator is independent of other oscillation circuits, WDT may still keep working during Standby mode and Halt mode.

WDT can be enabled or disabled by a configuration word. When WDT is enabled by configuration word, its operation still can be controlled by register bit WDTEN (PCON[7]) during program execution. Moreover, the mechanism after WDT time-out can reset NY8A053E or issue an interrupt request which is determined by another configuration word. At the same time, register bit /TO (STATUS[4]) will be clear to 0 after WDT time-out.

The baseline of WDT time-out period can be 3.5 ms, 15 ms, 60 ms or 250 ms which is determined by two configuration words. The time-out period can be lengthened if Prescaler0 is assigned to WDT. Prescaler0 will be assigned to WDT by writing 1 to register bit PS0WDT. The dividing rate of Prescaler0 for WDT is determined by register bits PS0SEL[2:0] and depends on WDT time-out mechanism. The dividing rate is from 1:1 to 1:128 if WDT time-out will reset NY8A053E and dividing rate is from 1:2 to 1:256 if WDT time-out will interrupt NY8A053E.

When Prescaler0 is assigned to WDT, the execution of instruction CLRWDT will clear WDT, Prescaler0 and set /TO flag to 1.

If user selects interrupt for WDT time-out mechanism, register bit WDTIF (INTF[6]) will set to 1 after WDT is expired. It may generate an interrupt request if register bit WDTIE (INTE[6]) and GIE both set to 1. WDTIF will not be clear until firmware writes 0 to WDTIF.

### 3.13 Interrupt

NY8A053E provides two kinds of interrupt: one is software interrupt and the other is hardware interrupt. Software interrupt is caused by execution of instruction INT. There are 6 hardware interrupts:

- Timer0 overflow interrupt.
- Timer1 underflow interrupt.
- WDT timeout interrupt.
- PB input change interrupt.
- External interrupt.
- LVD interrupt

GIE is global interrupt enable flag. It has to be 1 to enable hardware interrupt functions. GIE can be set by ENI instruction and clear to 0 by DISI instruction.

After instruction INT is executed, no matter GIE is set or clear, the next instruction will be fetched from address 0x001. At the same time, GIE will be clear to 0 by NY8A053E automatically. This will prevent nested interrupt from happening. The last instruction of interrupt service routine of software interrupt has to be RETIE. Execution of this instruction will set GIE to 1 and return to original execution sequence.

While any of hardware interrupts is occurred, the corresponding bit of Interrupt Flag Register INTF will be set to 1. This bit will not be clear until firmware writes 0 to this bit. Therefore user can obtain information of which event causes hardware interrupt by polling register INTF. Note that only when the corresponding bit of Interrupt Enable register INTE is set to 1, will the corresponding interrupt flag be read. And if the corresponding bit of Interrupt Enable Register INTE is set to 1 and GIE is also 1, hardware interrupt will occur and next instruction will be fetched from 0x008. At the same time, the register bit GIE will be clear by NY8A053E automatically. If user wants to implement nested interrupt, instruction ENI can be used as the first instruction of interrupt service routine which will set GIE to 1 again and allow other interrupt events to interrupt NY8A053E again. Instruction RETIE has to be the last instruction of interrupt service routine which will set GIE to 1 and return to original execution sequence.

It should be noted that ENI instruction cannot be placed right before RETIE instruction because ENI instruction in interrupt service routine will trigger nested interrupt, but RETIE will clear internal interrupt processing after jump out of ISR, so it is possible for interrupt flag to be falsely cleared.

#### 3.13.1 Timer0 Overflow Interrupt

Timer0 overflow (from 0x00 to 0xFF) will set register bit T0IF. This interrupt request will be serviced if T0IE and GIE are set to 1.

### 3.13.2 Timer1 Underflow Interrupt

Timer1 underflow (from 0xFF to 0x00) will set register bit T1IF. This interrupt request will be serviced if T1IE and GIE are set to 1.

### 3.13.3 WDT Timeout Interrupt

When WDT is timeout and the configuration word selects WDT timeout will generate interrupt request, it will set register bit WDTIF. This interrupt request will be serviced if WDTIE and GIE are set to 1.

### 3.13.4 PB Input Change Interrupt

When PB<sub>x</sub>,  $0 \leq x \leq 7$ , is configured as input pin and corresponding register bit WUPB<sub>x</sub> is set to 1, a level change on these selected I/O pin(s) will set register bit PBIF. This interrupt request will be serviced if PBIE and GIE are set to 1. Note when PB0 is both set as level change interrupt and external interrupt, the external interrupt enable EIS=1 will disable PB0 level change operation.

### 3.13.5 External Interrupt

According to the configuration of EIS=1 and INTEDG, the selected active edge on I/O pin PB0 will set register bit INTIF and this interrupt request will be served if INTIE and GIE are set to 1.

### 3.13.6 LVD Interrupt

When V<sub>DD</sub> level falls below LVD voltage, LVD flag will go from high to low, and set the register bit LVDIF=1. This interrupt request will be serviced if LVDIE and GIE are set to 1.

## 3.14 Oscillation Configuration

Because NY8A053E is a dual-clock IC, there are high oscillation (F<sub>HOSC</sub>) and low oscillation (F<sub>LOSC</sub>) that can be selected as system oscillation (F<sub>OSC</sub>). The oscillators which could be used as F<sub>HOSC</sub> are internal high RC oscillator (I\_HRC), external high crystal oscillator (E\_HXT) and external crystal oscillator (E\_XT). The oscillators which could be used as F<sub>LOSC</sub> are internal low RC oscillator (I\_LRC) and external low crystal oscillator (E\_LXT).

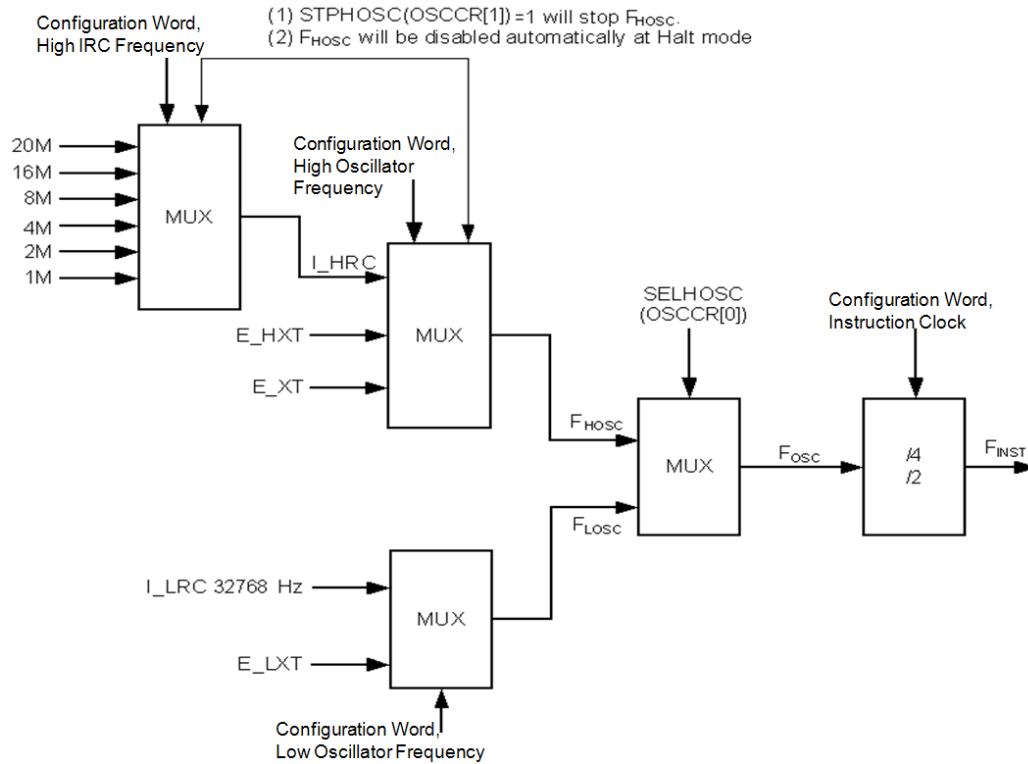


Figure 22 Oscillation Configuration of NY8A053E

There are two configuration words to determine which oscillator will be used as  $F_{HOSC}$ . When  $I\_HRC$  is selected as  $F_{HOSC}$ ,  $I\_HRC$  output frequency is determined by three configuration words and it can be 1M, 2M, 4M, 8M, 16M or 20MHz. Moreover, external crystal oscillator pads PB4 and PB5 can be used as I/O pins. On the other hand, PB4 can be the output pin of instruction clock according to a configuration word's setting. If  $F_{HOSC}$  required external crystal whose frequency ranges from 8MHz to 20MHz,  $E\_HXT$  is recommended. If  $F_{HOSC}$  required external crystal whose frequency ranges from 455KHz to 6MHz,  $E\_XT$  is recommended. When  $E\_HXT$  or  $E\_XT$  is adopted, PB4/PB5 cannot be used as I/O pins. They must be used as crystal output pin and input pin. PB4 is crystal output pin (Xout) and PB5 is crystal input pin (Xin).

There is one configuration word to determine which oscillator will be used as  $F_{LOSC}$ . When  $I\_LRC$  is selected, its frequency is centered on 32768Hz. If  $F_{LOSC}$  required external crystal,  $E\_LXT$  is selected and only 32768Hz crystal is allowed. When  $E\_LXT$  is adopted, PB4/PB5 cannot be used as I/O pins. They must be used as crystal output pin and input pin. PB4 is crystal output pin (Xout) and PB5 is crystal input pin (Xin). The dual-clock combinations of  $F_{HOSC}$  and  $F_{LOSC}$  are listed below:

| No. | FHOSC         | FLOSC |
|-----|---------------|-------|
| 1   | I_HRC         | I_LRC |
| 2   | E_HXT or E_XT | I_LRC |
| 3   | I_HRC         | E_LXT |

Table 16 Dual-clock combinations



When E\_HXT, E\_XT or E\_LXT is used as one of oscillations, the crystal or resonator is connected to Xin and Xout to provide oscillation. Moreover, a resistor and two capacitors are recommended to connect as following figure in order to provide reliable oscillation, refer to the specification of crystal or resonator to adopt appropriate C1 or C2 value. The recommended value of C1 and C2 are listed in the table below.

| Oscillation Mode | Crystal Frequency(Hz) | C1, C2 (pF) |
|------------------|-----------------------|-------------|
| E_HXT            | 16M                   | 5 ~ 10      |
|                  | 10M                   | 5 ~ 30      |
|                  | 8M                    | 5 ~ 20      |
| E_XT             | 4M                    | 5 ~ 30      |
|                  | 1M                    | 5 ~ 30      |
|                  | 455K                  | 10 ~ 100    |
| E_LXT            | 32768                 | 10 ~ 30     |

Table 17 Recommended C1 and C2 Value for Different Kinds of Crystal Oscillation

For 20MHZ resonator in 2 clock CPU cycle mode, An 18pF C2 capacitor is a must.

Moreover, for precision 32.768k crystal, it is recommended to set C1=C2=20pF capacitor.

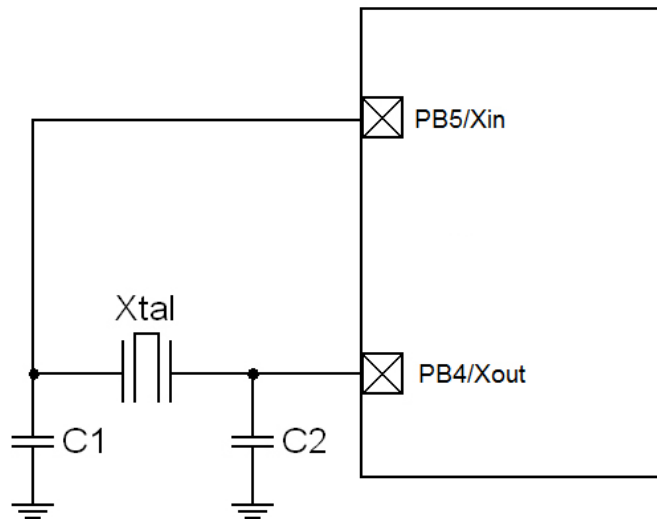


Figure 23 Connection for External Crystal Oscillation

Either  $F_{HOSC}$  or  $F_{LOSC}$  can be selected as system oscillation  $F_{OSC}$  according to the value of register bit SELHOSC (OSCCR[0]). When SELHOSC is 1,  $F_{HOSC}$  is selected as  $F_{OSC}$ . When SELHOSC is 0,  $F_{LOSC}$  is selected as  $F_{OSC}$ . Once  $F_{OSC}$  is determined, the instruction clock  $F_{INST}$  can be  $F_{OSC}/2$  or  $F_{OSC}/4$  according to value of a configuration word.

### 3.15 Operating Mode

NY8A053E provides four kinds of operating mode to tailor all kinds of application and save power consumptions. These operating modes are Normal mode, Slow mode, Standby mode and Halt mode. Normal mode is designated for high-speed operating mode. Slow mode is designated for low-speed mode in order to save power consumption. At Standby mode, NY8A053E will stop almost all operations except Timer0/Timer1/WDT in order to wake-up periodically. At Halt mode, NY8A053E will sleep until external event or WDT trigger IC to wake-up.

The block diagram of four operating modes is described in the following figure.

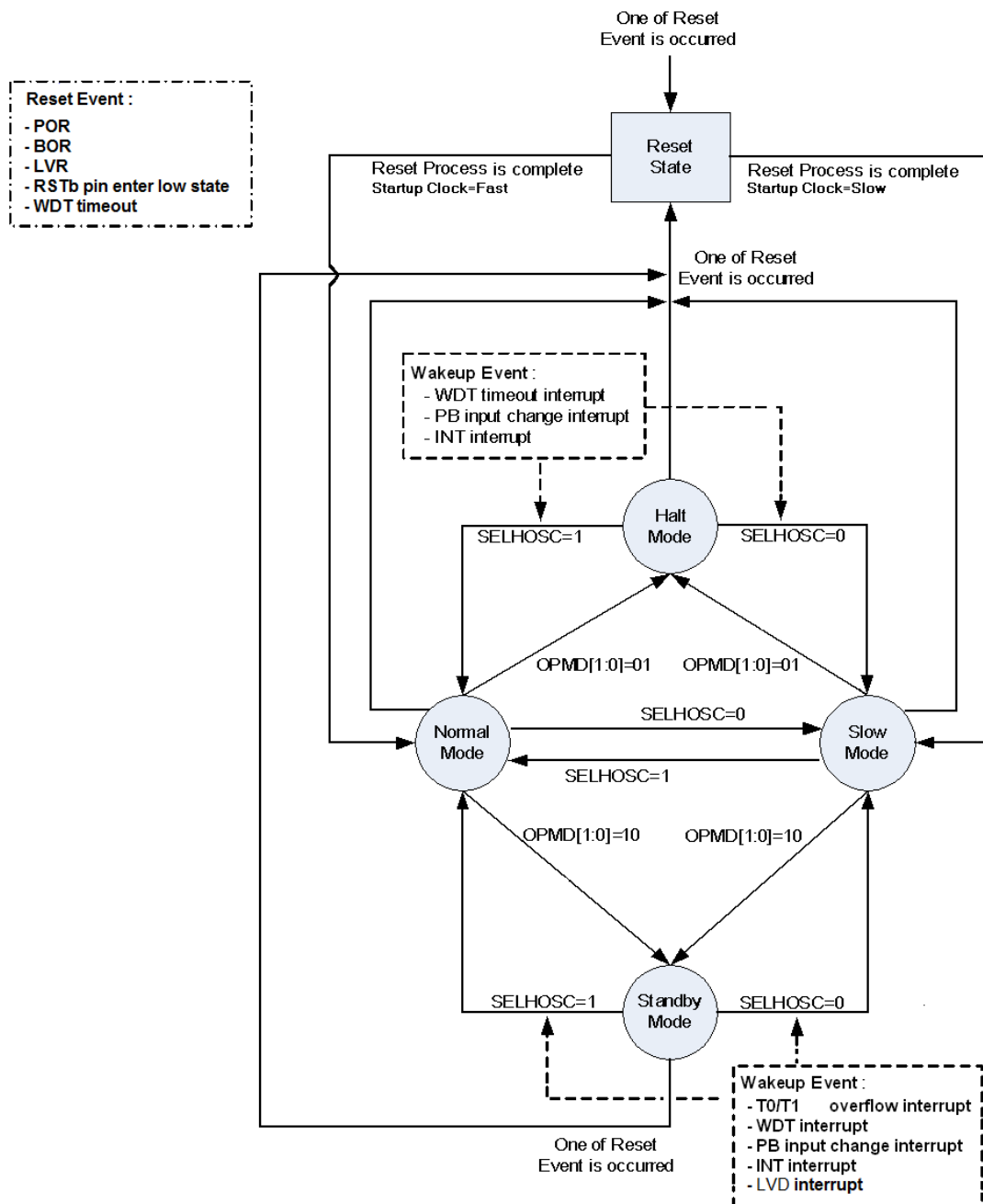


Figure 24 Four Operating Modes

### 3.15.1 Normal Mode

After any Reset Event is occurred and Reset Process is completed, NY8A053E will begin to execute program under Normal mode or Slow mode. Which mode is selected after Reset Process is determined by the Startup Clock configuration word. If Startup Clock=fast, NY8A053E will enter Normal mode, if Startup Clock=Slow, NY8A053E will enter Slow mode. At Normal mode,  $F_{HOSC}$  is selected as system oscillation in order to provide highest performance and its power consumption will be the largest among four operating modes. After power on or any reset trigger is released, NY8A053E will enter Normal mode after reset process is completed.

- Instruction execution is based on  $F_{HOSC}$  and all peripheral modules may be active according to corresponding module enable bit.
- The  $F_{LOSC}$  is still active and running.
- IC can switch to Slow mode by writing 0 to register bit SELHOSC (OSCCR[0]).
- IC can switch to Standby mode or Halt mode by programming register bits OPMD[1:0] (OSCCR[3:2]).
- For real time clock applications, the NY8A053E can run in normal mode, at the same time the low-frequency clock Low Oscillator Frequency connects to timer0 clock. This is made possible by setting LCKTM0 to 1 and corresponding configuration word Timer0 source setting to 1.

### 3.15.2 Slow Mode

NY8A053E will enter Slow mode by writing 0 to register bit SELHOSC. At Slow mode,  $F_{LOSC}$  is selected as system oscillation in order to save power consumption but still keep IC running. However,  $F_{HOSC}$  will not be disabled automatically by NY8A053E. Therefore user can write 0 to register bit STPHOSC (OSCCR[1]) in slow mode to reduce power consumption further. But it is noted that it is forbidden to enter slow mode and stop  $F_{HOSC}$  at the same time, one must enter slow mode first, then disable  $F_{HOSC}$ , or the program may hang on.

- Instruction execution is based on  $F_{LOSC}$  and all peripheral modules may be active according to corresponding module enable bit.
- $F_{HOSC}$  can be disabled by writing 1 to register bit STPHOSC.
- IC can switch to Standby mode or Halt mode by programming register bits OPMD[1:0].
- IC can switch to Normal mode by writing 1 to SELHOSC.

### 3.15.3 Standby Mode

NY8A053E will enter Standby mode by writing 10b to register bits OPMD[1:0]. At Standby mode, however,  $F_{HOSC}$  will not be disabled automatically by NY8A053E and user has to enter slow mode and write 1 to register bit STPHOSC in order to stop  $F_{HOSC}$  oscillation. Most of NY8A053E peripheral modules are disabled but Timer can be still active if register bit T0EN/T1EN is set to 1. Therefore NY8A053E can wake-up after Timer0/Timer1 is expired. The expiration period is determined by the register TMR0/TMR1,  $F_{INST}$  and other configurations for Timer0/Timer1.

- Instruction execution is stop and some peripheral modules may be active according to corresponding module enable bit.
- $F_{HOSC}$  can be disabled by writing 1 to register bit  $STPHOSC$ .
- The  $F_{LOSC}$  is still active and running.
- IC can wake-up from Standby mode if any of
  - (a) Timer0/Timer1 (overflow/underflow) interrupt
  - (b) WDT timeout interrupt
  - (c) PB input change interrupt
  - (d) INT external interrupt is happened.
  - (e) LVD interrupt.
- After wake-up from Standby mode, IC will return to Normal mode if  $SELHOSC=1$ , IC will return to Slow mode if  $SELHOSC=0$ .
- It is not recommended to change oscillator mode (normal to slow / slow to normal) and enter standby mode at the same time.

### 3.15.4 Halt Mode

NY8A053E will enter Halt mode by executing instruction SLEEP or writing 01b to register bits  $OPMD[1:0]$ . After entering Halt mode, register bit  $/PD$  ( $STATUS[3]$ ) will be clear to 0, register bit  $/TO$  ( $STATUS[4]$ ) will be set to 1 and WDT will be clear but keep running.

At Halt mode, all of peripheral modules are disabled, instruction execution is stop and NY8A053E can only wake-up by some specific events. Therefore, Halt mode is the most power saving mode provided by NY8A053E.

- Instruction execution is stop and all peripheral modules are disabled.
- $F_{HOSC}$  and  $F_{LOSC}$  are both disabled automatically.
- IC can wake-up from Halt mode if any of
  - (a) WDT timeout interrupt
  - (b) PB input change interrupt
  - (c) INT or external interrupt is happened.
- After wake-up from Halt mode, IC will return to Normal mode if  $SELHOSC=1$ , IC will return to Slow mode if  $SELHOSC=0$ .

**Note: Users can change  $STPHOSC$  and enter Halt mode in the same instruction.**

- It is not recommended to change oscillator mode (normal to slow or slow to normal) and enter halt mode at the same time.

### 3.15.5 Wake-up Stable Time

The wake-up stable time of Halt mode is determined by Configuration word: High Oscillator Frequency or Low Oscillator Frequency. If one of E\_HXT, E\_XT and E\_LXT is selected, the wake-up period would be  $512 \cdot F_{osc}$ . And if no XT mode are selected,  $16 \cdot F_{osc}$  would be set as wake-up period. On the other hand, there is no need of wake-up stable time for Standby mode because either  $F_{HOSC}$  or  $F_{LOSC}$  is still running at Standby mode.

Before NY8A053E enter Standby mode or Halt mode, user may execute instruction ENI. At this condition, NY8A053E will branch to address 0x008 in order to execute interrupt service routine after wake-up. If instruction DISI is executed or 0 is written to register bit GIE before entering Standby mode or Halt mode, the next instruction will be executed after wake-up.

### 3.15.6 Summary of Operating Mode

The summary of four operating modes is described in the following table.

| Mode                  | Normal            | Slow              | Standby   | Halt  |
|-----------------------|-------------------|-------------------|---|---|
| $F_{HOSC}$            | Enabled           | STPHOSC           | STPHOSC   | Disabled                                    |
| $F_{LOSC}$            | Enabled           | Enabled           | Enabled   | Disabled                                    |
| Instruction Execution | Executing         | Executing         | Stop  | Stop  |
| Timer0/1              | T0EN / T1EN       | T0EN / T1EN       | T0EN / T1EN   | Disabled                                    |
| WDT                   | Option and WDTEN  | Option and WDTEN  | Option and WDTEN  | Option and WDTEN                            |
| Other Modules         | Module enable bit | Module enable bit | Module enable bit   | All disabled                                |
| Wake-up Source        | -                 | -                 | - Timer0 overflow<br>- Timer1 underflow<br>- WDT timeout<br>- PB input change<br>- INT<br>- LVD interrupt | - WDT timeout<br>- PB input change<br>- INT |

Table 18 Summary of Operating Modes

### 3.16 Reset Process

NY8A053E will enter Reset State and start Reset Process when one of following Reset Event is occurred:

- Power-On Reset (POR) is occurred when  $V_{DD}$  rising is detected.
- Low-Voltage Reset (LVR) is occurred when operating  $V_{DD}$  is below pre-defined voltage.
- Pin RSTb is low state.
- WDT timeout reset.

Moreover, value of all registers will be initialized to their initial value or unchanged if its initial value is unknown. The status bits /TO and /PD could be initialized according to which event causes reset. The /TO and /PD value and its associated event is summarized in the table below.

| Event                         | /TO       | /PD       |
|-------------------------------|-----------|-----------|
| POR, LVR                      | 1         | 1         |
| RSTb reset from non-Halt mode | unchanged | unchanged |
| RSTb reset from Halt mode     | 1         | 1         |
| WDT reset from non-Halt mode  | 0         | 1         |
| WDT reset from Halt mode      | 0         | 0         |
| SLEEP executed                | 1         | 0         |
| CLRWDT executed               | 1         | 1         |

Table 19 Summary of /TO & /PD Value and its Associated Event

After Reset Event is released, NY8A053E will start Reset Process. It will wait certain amount of period for oscillation stable no matter what kind of oscillator is adopted. This period is called power-up reset time and is determined by two-bit configuration words which can be 140us, 4.5ms, 18ms, 72ms or 288ms. After oscillator is stable, NY8A053E will wait further 16 clock cycles of Fosc (oscillator start-up time, OST) and Reset Process is complete.

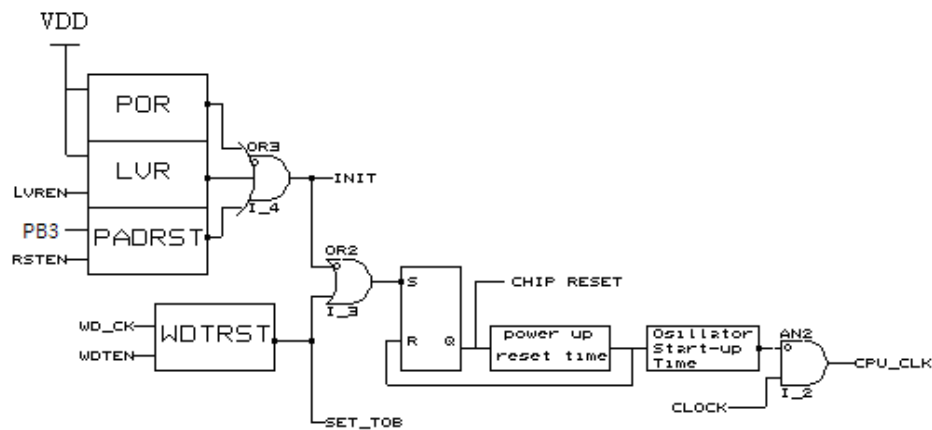


Figure 25 Block Diagram of On-Chip Reset Circuit

For slow V<sub>DD</sub> power-up, it is recommended to use RSTb reset, as the following figure.

- It is recommended the R value should be not greater than 40kΩ.
- The R1 value=100Ω to 1kΩ will prevent high current, ESD or Electrical overstress flowing into reset pin.
- The diode helps discharge quickly when power down.

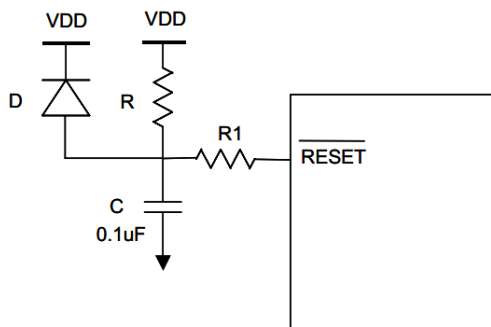


Figure 26 Block Diagram of Reset Application

#### 4. Instruction Set

NY8A053E provides 55 powerful instructions for all kinds of applications.

| Inst.                             | OP |     | Operation                    | Cyc.   | Flag |
|-----------------------------------|----|-----|------------------------------|--------|------|
|                                   | 1  | 2   |                              |        |      |
| <b>Arithmetic Instructions</b>    |    |     |                              |        |      |
| ANDAR                             | R  | d   | dest = ACC & R               | 1      | Z    |
| IORAR                             | R  | d   | dest = ACC   R               | 1      | Z    |
| XORAR                             | R  | d   | dest = ACC $\oplus$ R        | 1      | Z    |
| ANDIA                             | i  |     | ACC = ACC & i                | 1      | Z    |
| IORIA                             | i  |     | ACC = ACC   i                | 1      | Z    |
| XORIA                             | i  |     | ACC = ACC $\oplus$ i         | 1      | Z    |
| RRR                               | R  | d   | Rotate right R               | 1      | C    |
| RLR                               | R  | d   | Rotate left R                | 1      | C    |
| BSR                               | R  | bit | Set bit in R                 | 1      | -    |
| BCR                               | R  | bit | Clear bit in R               | 1      | -    |
| INCR                              | R  | d   | Increase R                   | 1      | Z    |
| DECR                              | R  | d   | Decrease R                   | 1      | Z    |
| COMR                              | R  | d   | dest = $\sim$ R              | 1      | Z    |
| <b>Conditional Instructions</b>   |    |     |                              |        |      |
| BTRSC                             | R  | bit | Test bit in R, skip if clear | 1 or 2 | -    |
| BTRSS                             | R  | bit | Test bit in R, skip if set   | 1 or 2 | -    |
| INCRSZ                            | R  | d   | Increase R, skip if 0        | 1 or 2 | -    |
| DECRSZ                            | R  | d   | Decrease R, skip if 0        | 1 or 2 | -    |
| <b>Data Transfer Instructions</b> |    |     |                              |        |      |
| MOVAR                             | R  |     | Move ACC to R                | 1      | -    |
| MOVR                              | R  | d   | Move R                       | 1      | Z    |
| MOVIA                             | i  |     | Move immediate to ACC        | 1      | -    |
| SWAPR                             | R  | d   | Swap halves R                | 1      | -    |
| IOST                              | F  |     | Load ACC to F-page SFR       | 1      | -    |
| IOSTR                             | F  |     | Move F-page SFR to ACC       | 1      | -    |
| SFUN                              | S  |     | Load ACC to S-page SFR       | 1      | -    |
| SFUNR                             | S  |     | Move S-page SFR to ACC       | 1      | -    |
| T0MD                              |    |     | Load ACC to T0MD             | 1      | -    |
| T0MDR                             |    |     | Move T0MD to ACC             | 1      | -    |
| TABLEA                            |    |     | Read ROM                     | 2      | -    |

| Inst.                          | OP  |   | Operation                                  | Cyc. | Flag     |
|--------------------------------|-----|---|--|------|----------|
|                                | 1   | 2 |  |      |          |
| <b>Arithmetic Instructions</b> |     |   |  |      |          |
| ADDAR                          | R   | d | dest = R + ACC                             | 1    | Z, DC, C |
| SUBAR                          | R   | d | dest = R + ( $\sim$ ACC)                   | 1    | Z, DC, C |
| ADCAR                          | R   | d | dest = R + ACC + C                         | 1    | Z, DC, C |
| SBCAR                          | R   | d | dest = R + ( $\sim$ ACC) + C               | 1    | Z, DC, C |
| ADDIA                          | i   |   | ACC = i + ACC                              | 1    | Z, DC, C |
| SUBIA                          | i   |   | ACC = i + ( $\sim$ ACC)                    | 1    | Z, DC, C |
| ADCIA                          | i   |   | ACC = i + ACC + C                          | 1    | Z, DC, C |
| SBCIA                          | i   |   | ACC = i + ( $\sim$ ACC) + C                | 1    | Z, DC, C |
| DAA                            |     |   | Decimal adjust for ACC                     | 1    | C        |
| CMPAR                          | R   |   | Compare R with ACC                         | 1    | Z, C     |
| CLRA                           |     |   | Clear ACC                                  | 1    | Z        |
| CLRR                           |     |   | Clear R                                    | 1    | Z        |
| <b>Other Instructions</b>      |     |   |  |      |          |
| NOP                            |     |   | No operation                               | 1    | -        |
| SLEEP                          |     |   | Go into Halt mode                          | 1    | /TO, /PD |
| CLRWDT                         |     |   | Clear Watch-Dog Timer                      | 1    | /TO, /PD |
| ENI                            |     |   | Enable interrupt                           | 1    | -        |
| DISI                           |     |   | Disable interrupt                          | 1    | -        |
| INT                            |     |   | Software Interrupt                         | 3    | -        |
| RET                            |     |   | Return from subroutine                     | 2    | -        |
| RETIE                          |     |   | Return from interrupt and enable interrupt | 2    | -        |
| RETIA                          | i   |   | Return, place immediate in ACC             | 2    | -        |
| CALLA                          |     |   | Call subroutine by ACC                     | 2    | -        |
| GOTOA                          |     |   | unconditional branch by ACC                | 2    | -        |
| CALL                           | adr |   | Call subroutine                            | 2    | -        |
| GOTO                           | adr |   | unconditional branch                       | 2    | -        |
| LCALL                          | adr |   | Call subroutine                            | 2    | -        |
| LGOTO                          | adr |   | unconditional branch                       | 2    | -        |

Table 20 Instruction Set

ACC: Accumulator.

adr: immediate address.

bit: bit address within an 8-bit register R.

**C**: Carry/Borrow bit

C=1, carry is occurred for addition instruction or borrow is **NOT** occurred for subtraction instruction.

C=0, carry is not occurred for addition instruction or borrow **IS** occurred for subtraction instruction.

d: Destination

If d is "0", the result is stored in the ACC.

If d is "1", the result is stored back in register R.

DC: Digital carry flag.

dest: Destination.

F: F-page SFR, F is 0x5 ~ 0xF.

i: 8-bit immediate data.

PC: Program Counter.

PCHBUF: High Byte Buffer of Program Counter.

**/PD**: Power down flag bit

/PD=1, after power-up or after instruction CLRWDT is executed.

/PD=0, after instruction SLEEP is executed.

Prescaler: Prescaler0 dividing rate.

R: R-page SFR, R is 0x00 ~0x3F.

S: S-page SFR, S is 0x0 ~ 0x18.

T0MD: T0MD register.

TBHP: The high-Byte at target address in ROM.

TBHD: Store the high-Byte data at target address in ROM.

**/TO**: Time overflow flag bit

/TO=1, after power-up or after instruction CLRWDT or SLEEP is executed.

/TO=0, WDT timeout is occurred.

WDT: Watchdog Timer Counter.

Z: Zero flag.



| <b>ADCAR</b>     | <b>Add ACC and R with Carry</b>   |
|------------------|---|
| Syntax:          | ADCAR R, d  |
| Operand:         | $0 \leq R \leq 63$<br>d = 0, 1.   |
| Operation:       | $R + ACC + C \rightarrow \text{dest}$   |
| Status affected: | Z, DC, C  |
| Description:     | Add the contents of ACC and register R with Carry. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R. |
| Cycle            | 1   |
| Example:         | ADCAR R, d<br>before executing instruction:<br>ACC=0x12, R=0x34, C=1, d=1,<br>after executing instruction:<br>R=0x47, ACC=0x12, C=0.  |

| <b>ADDAR</b>     | <b>Add ACC and R</b>   |
|------------------|--|
| Syntax:          | ADDAR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>d = 0, 1.  |
| Operation:       | $ACC + R \rightarrow \text{dest}$  |
| Status affected: | Z, DC, C   |
| Description:     | Add the contents of ACC and R. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R.                    |
| Cycle:           | 1  |
| Example:         | ADDAR R, d<br>before executing instruction:<br>ACC=0x12, R=0x34, C=1, d=1,<br>after executing instruction:<br>R=0x46, ACC=0x12, C=0. |

| <b>ADCIA</b>     | <b>Add ACC and Immediate with Carry</b>  |
|------------------|--|
| Syntax:          | ADCIA i  |
| Operand:         | $0 \leq i < 255$   |
| Operation:       | $ACC + i + C \rightarrow ACC$  |
| Status affected: | Z, DC, C   |
| Description:     | Add the contents of ACC and the 8-bit immediate data i with Carry. The result is placed in ACC.                      |
| Cycle:           | 1  |
| Example:         | ADCIA i<br>before executing instruction:<br>ACC=0x12, i=0x34, C=1,<br>after executing instruction:<br>ACC=0x47, C=0. |

| <b>ADDIA</b>     | <b>Add ACC and Immediate</b>   |
|------------------|--|
| Syntax:          | ADDIA i  |
| Operand:         | $0 \leq i < 255$   |
| Operation:       | $ACC + i \rightarrow ACC$  |
| Status affected: | Z, DC, C   |
| Description:     | Add the contents of ACC with the 8-bit immediate data i. The result is placed in ACC.                                |
| Cycle:           | 1  |
| Example:         | ADDIA i<br>before executing instruction:<br>ACC=0x12, i=0x34, C=1,<br>after executing instruction:<br>ACC=0x46, C=0. |

| <b>ANDAR</b>     | <b>AND ACC and R</b>  |
|------------------|---|
| Syntax:          | ANDAR R, d  |
| Operand:         | $0 \leq R \leq 63$ .<br>$d = 0, 1$ .  |
| Operation:       | ACC & R $\rightarrow$ dest  |
| Status affected: | Z   |
| Description:     | The content of ACC is AND'ed with R. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R.         |
| Cycle:           | 1   |
| Example:         | ANDAR R, d<br>before executing instruction:<br>ACC=0x5A, R=0xAF, d=1.<br>after executing instruction:<br>R=0x0A, ACC=0x5A, Z=0. |

| <b>BCR</b>       | <b>Clear Bit in R</b>   |
|------------------|---|
| Syntax:          | BCR R, bit  |
| Operand:         | $0 \leq R \leq 63$<br>$0 \leq \text{bit} \leq 7$  |
| Operation:       | $0 \rightarrow R[\text{bit}]$   |
| Status affected: | --  |
| Description:     | Clear the bit <sup>th</sup> position in R.  |
| Cycle:           | 1   |
| Example:         | BCR R,B2<br>before executing instruction:<br>R=0x5A, B2=0x3,<br>after executing instruction:<br>R=0x52. |

| <b>ANDIA</b>     | <b>AND Immediate with ACC</b>   |
|------------------|---|
| Syntax:          | ANDIA i   |
| Operand:         | $0 \leq i < 255$  |
| Operation:       | ACC & i $\rightarrow$ ACC   |
| Status affected: | Z   |
| Description:     | The content of ACC register is AND'ed with the 8-bit immediate data i. The result is placed in ACC.             |
| Cycle:           | 1   |
| Example:         | ANDIA i<br>before executing instruction:<br>ACC=0x5A, i=0xAF,<br>after executing instruction:<br>ACC=0x0A, Z=0. |

| <b>BSR</b>       | <b>Set Bit in R</b>   |
|------------------|---|
| Syntax:          | BSR R, bit  |
| Operand:         | $0 \leq R \leq 63$<br>$0 \leq \text{bit} \leq 7$  |
| Operation:       | $1 \rightarrow R[\text{bit}]$   |
| Status affected: | --  |
| Description:     | Set the bit <sup>th</sup> position in R.  |
| Cycle:           | 1   |
| Example:         | BSR R,B2<br>before executing instruction:<br>R=0x5A, B2=0x2,<br>after executing instruction:<br>R=0x5E. |

| <b>BTRSC</b>     | <b>Test Bit in R and Skip if Clear</b>   |
|------------------|--|
| Syntax:          | BTRSC R, bit   |
| Operand:         | $0 \leq R \leq 63$<br>$0 \leq \text{bit} \leq 7$   |
| Operation:       | Skip next instruction, if R[bit] = 0.  |
| Status affected: | --   |
| Description:     | If R[bit] = 0, the next instruction which is already fetched is discarded and a NOP is executed instead. Therefore it makes this instruction a two-cycle instruction.  |
| Cycle:           | 1 or 2(skip)   |
| Example:         | BTRSC R, B2<br>Instruction1<br>Instruction2<br>before executing instruction:<br>R=0x5A, B2=0x2,<br>after executing instruction:<br>because R[B2]=0, instruction1 will not be executed, the program will start execute instruction from instruction2. |

| <b>CALL</b>      | <b>Call Subroutine</b>  |
|------------------|---|
| Syntax:          | CALL adr  |
| Operand:         | $0 \leq \text{adr} < 255$   |
| Operation:       | PC + 1 $\rightarrow$ Top of Stack<br>{PCHBUF, adr} $\rightarrow$ PC   |
| Status affected: | --  |
| Description:     | The return address (PC + 1) is pushed onto top of Stack. The 8-bit immediate address adr is loaded into PC[7:0] and PCHBUF[1:0] is loaded into PC[9:8].     |
| Cycle:           | 2   |
| Example:         | CALL SUB<br>before executing instruction:<br>PC=A0. Stack pointer=1<br>after executing instruction:<br>PC=address of SUB, Stack[1] = A0+1, Stack pointer=2. |

| <b>BTRSS</b>     | <b>Test Bit in R and Skip if Set</b>   |
|------------------|--|
| Syntax:          | BTRSS R, bit   |
| Operand:         | $0 \leq R \leq 63$<br>$0 \leq \text{bit} \leq 7$   |
| Operation:       | Skip next instruction, if R[bit] = 1.  |
| Status affected: | --   |
| Description:     | If R[bit] = 1, the next instruction which is already fetched is discarded and a NOP is executed instead. Therefore it makes this instruction a two-cycle instruction.  |
| Cycle:           | 1 or 2(skip)   |
| Example:         | BTRSS R, B2<br>Instruction2<br>Instruction3<br>before executing instruction:<br>R=0x5A, B2=0x3,<br>after executing instruction:<br>because R[B2]=1, instruction2 will not be executed, the program will start execute instruction from instruction3. |

| <b>CALLA</b>     | <b>Call Subroutine</b>  |
|------------------|---|
| Syntax:          | CALLA   |
| Operand:         | --  |
| Operation:       | PC + 1 $\square$ Top of Stack<br>{TBHP, ACC} $\rightarrow$ PC   |
| Status affected: | --  |
| Description:     | The return address (PC + 1) is pushed onto top of Stack. The contents of TBHP[1:0] is loaded into PC[9:8] and ACC is loaded into PC[7:0].                             |
| Cycle:           | 2   |
| Example:         | CALLA<br>before executing instruction:<br>TBHP=0x02, ACC=0x34.<br>PC=A0. Stack pointer=1.<br>after executing instruction:<br>PC=0x234, Stack[1]=A0+1, Stack pointer=2 |

| <b>CLRA</b>      | <b>Clear ACC</b>  |
|------------------|---|
| Syntax:          | CLRA  |
| Operand:         | --  |
| Operation:       | 00h → ACC<br>1 → Z  |
| Status affected: | Z   |
| Description:     | ACC is clear and Z is set to 1.   |
| Cycle:           | 1   |
| Example:         | CLRA<br>before executing instruction:<br>ACC=0x55, Z=0.<br>after executing instruction:<br>ACC=0x00, Z=1. |

| <b>CLRWDT</b>    | <b>Clear Watch-Dog Timer</b>  |
|------------------|---|
| Syntax:          | CLRWDT  |
| Operand:         | --  |
| Operation:       | 00h → WDT,<br>00h → WDT prescaler<br>1 → /TO<br>1 → /PD   |
| Status affected: | /TO, /PD  |
| Description:     | Executing CLRWDT will reset WDT, Prescaler0 if it is assigned to WDT. Moreover, status bits /TO and /PD will be set to 1. |
| Cycle:           | 1   |
| Example:         | CLRWDT<br>before executing instruction:<br>/TO=0<br>after executing instruction:<br>/TO=1                                 |

| <b>CLRR</b>      | <b>Clear R</b>  |
|------------------|---|
| Syntax:          | CLRR R  |
| Operand:         | $0 \leq R \leq 63$  |
| Operation:       | 00h → R<br>1 → Z  |
| Status affected: | Z   |
| Description:     | The content of R is clear and Z is set to 1.  |
| Cycle:           | 1   |
| Example:         | CLRR R<br>before executing instruction:<br>R=0x55, Z=0.<br>after executing instruction:<br>R=0x00, Z=1. |

| <b>COMR</b>      | <b>Complement R</b>  |
|------------------|--|
| Syntax:          | COMR R, d  |
| Operand:         | $0 \leq R \leq 63$<br>d = 0, 1.  |
| Operation:       | $\sim R \rightarrow \text{dest}$   |
| Status affected: | Z  |
| Description:     | The content of R is complemented. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R. |
| Cycle:           | 1  |
| Example:         | COMR , d<br>before executing instruction:<br>R=0xA6, d=1, Z=0.<br>after executing instruction:<br>R=0x59, Z=0.       |

| <b>CMPAR</b>     | <b>Compare ACC and R</b>   |
|------------------|--|
| Syntax:          | CMPAR R  |
| Operand:         | $0 \leq R \leq 63$   |
| Operation:       | $R - ACC \rightarrow$ (No restore)   |
| Status affected: | Z, C   |
| Description:     | Compare ACC and R by subtracting ACC from R with 2's complement representation. The content of ACC and R is not changed.           |
| Cycle:           | 1  |
| Example:         | CMPAR R<br>before executing instruction:<br>R=0x34, ACC=12, Z=0, C=0.<br>after executing instruction:<br>R=0x34, ACC=12, Z=0, C=1. |

| <b>DECR</b>      | <b>Decrease R</b>   |
|------------------|---|
| Syntax:          | DECR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$   |
| Operation:       | $R - 1 \rightarrow dest$  |
| Status affected: | Z   |
| Description:     | Decrease R. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R.                  |
| Cycle:           | 1   |
| Example:         | DECR R, d<br>before executing instruction:<br>R=0x01, d=1, Z=0.<br>after executing instruction:<br>R=0x00, Z=1. |

| <b>DAA</b>       | <b>Convert ACC Data Format from Hexadecimal to Decimal</b>   |
|------------------|--|
| Syntax:          | DAA  |
| Operand:         | --   |
| Operation:       | $ACC(hex) \rightarrow ACC(dec)$  |
| Status affected: | C  |
| Description:     | Convert ACC data format from hexadecimal to decimal after addition operation and restore result to ACC. DAA instruction must be placed immediately after addition operation if decimal format is required. Please note that interrupt should be disabled before addition instruction and enabled after DAA instruction to avoid unexpected result. |
| Cycle:           | 1  |
| Example:         | DISI<br>ADDAR R,d<br>DAA<br>ENI<br>before executing instruction:<br>ACC=0x28, R=0x25, d=0.<br>after executing instruction:<br>ACC=0x53, C=0.   |

| <b>DECRSZ</b>    | <b>Decrease R, Skip if 0</b>  |
|------------------|---|
| Syntax:          | DECRSZ R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$   |
| Operation:       | $R - 1 \rightarrow dest,$<br>Skip if result = 0   |
| Status affected: | --  |
| Description:     | Decrease R first. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R. If result is 0, the next instruction which is already fetched is discarded and a NOP is executed instead. Therefore it makes this instruction a two-cycle instruction. |
| Cycle:           | 1 or 2(skip)  |
| Example:         | DECRSZ R, d<br>instruction2<br>instruction3<br>before executing instruction:<br>R=0x1, d=1, Z=0.<br>after executing instruction:<br>R=0x0, Z=1, and instruction will skip instruction2 execution because the operation result is zero.                                      |

| <b>DISI</b>      | <b>Disable Interrupt Globally</b>   |
|------------------|---|
| Syntax:          | DISI  |
| Operand:         | --  |
| Operation:       | Disable Interrupt, 0 → GIE  |
| Status affected: | --  |
| Description:     | GIE is clear to 0 in order to disable all interrupt requests.                             |
| Cycle:           | 1   |
| Example:         | DISI<br>before executing instruction:<br>GIE=1.<br>After executing instruction:<br>GIE=0. |

| <b>GOTO</b>      | <b>Unconditional Branch</b>   |
|------------------|---|
| Syntax:          | GOTO adr  |
| Operand:         | $0 \leq \text{adr} < 511$   |
| Operation:       | {PCHBUF, adr} → PC  |
| Status affected: | --  |
| Description:     | GOTO is an unconditional branch instruction. The 9-bit immediate address adr is loaded into PC[8:0] and PCHBUF[1] is loaded into PC[9]. |
| Cycle:           | 2   |
| Example:         | GOTO Level<br>before executing instruction:<br>PC=A0.<br>after executing instruction:<br>PC=address of Level.                           |

| <b>ENI</b>       | <b>Enable Interrupt Globally</b>   |
|------------------|--|
| Syntax:          | ENI  |
| Operand:         | --   |
| Operation:       | Enable Interrupt, 1 → GIE  |
| Status affected: | --   |
| Description:     | GIE is set to 1 in order to enable all interrupt requests.                               |
| Cycle:           | 1  |
| Example:         | ENI<br>before executing instruction:<br>GIE=0.<br>After executing instruction:<br>GIE=1. |

| <b>GOTOA</b>     | <b>Unconditional Branch</b>   |
|------------------|---|
| Syntax:          | GOTOA   |
| Operand:         | --  |
| Operation:       | {TBHP, ACC} → PC  |
| Status affected: | --  |
| Description:     | GOTOA is an unconditional branch instruction. The content of TBHP[1:0] is loaded into PC[9:8] and ACC is loaded into PC[7:0]. |
| Cycle:           | 2   |
| Example:         | GOTOA<br>before executing instruction:<br>PC=A0. TBHP=0x02,<br>ACC=0x34.<br>after executing instruction:<br>PC=0x234.         |

| <b>INCR</b>      | <b>Increase R</b>   |
|------------------|---|
| Syntax:          | INCR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$   |
| Operation:       | $R + 1 \rightarrow \text{dest}.$  |
| Status affected: | Z   |
| Description:     | Increase R. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R.                  |
| Cycle:           | 1   |
| Example:         | INCR R, d<br>before executing instruction:<br>R=0xFF, d=1, Z=0.<br>after executing instruction:<br>R=0x00, Z=1. |

| <b>INT</b>       | <b>Software Interrupt</b>  |
|------------------|--|
| Syntax:          | INT  |
| Operand:         | --   |
| Operation:       | $PC + 1 \rightarrow \text{Top of Stack},$<br>$001h \rightarrow PC$   |
| Status affected: | --   |
| Description:     | Software interrupt. First, return address ( $PC + 1$ ) is pushed onto the Stack. The address 0x001 is loaded into PC[9:0]. |
| Cycle:           | 3  |
| Example:         | INT<br>before executing instruction:<br>PC=address of INT code<br>after executing instruction:<br>PC=0x01                  |

| <b>INCRSZ</b>    | <b>Increase R, Skip if 0</b>   |
|------------------|--|
| Syntax:          | INCRSZ R, d  |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$  |
| Operation:       | $R + 1 \rightarrow \text{dest},$<br>Skip if result = 0   |
| Status affected: | --   |
| Description:     | Increase R first. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R.<br>If result is 0, the next instruction which is already fetched is discarded and a NOP is executed instead. Therefore it makes this instruction a two-cycle instruction. |
| Cycle:           | 1 or 2(skip)   |
| Example:         | INCRSZ R, d<br>instruction2,<br>instruction3.<br>before executing instruction:<br>R=0xFF, d=1, Z=0.<br>after executing instruction:<br>R=0x00, Z=1. And the program will skip instruction2 execution because the operation result is zero.                                     |

| <b>IORAR</b>     | <b>OR ACC with R</b>   |
|------------------|--|
| Syntax:          | IORAR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$  |
| Operation:       | $ACC   R \rightarrow \text{dest}$  |
| Status affected: | Z  |
| Description:     | OR ACC with R. If d is 0, the result is stored in ACC. If d is 1, the result is stored back to R.                                    |
| Cycle:           | 1  |
| Example:         | IORAR R, d<br>before executing instruction:<br>R=0x50, ACC=0xAA, d=1, Z=0.<br>after executing instruction:<br>R=0xFA, ACC=0xAA, Z=0. |

| <b>INT</b> | <b>Software Interrupt</b> |
|------------|---------------------------|
|------------|---------------------------|

| <b>IORIA</b>     | <b>OR Immediate with ACC</b>   |
|------------------|--|
| Syntax:          | IORIA i  |
| Operand:         | $0 \leq i < 255$   |
| Operation:       | $ACC \mid i \rightarrow ACC$   |
| Status affected: | Z  |
| Description:     | OR ACC with 8-bit immediate data i. The result is stored in ACC.   |
| Cycle:           | 1  |
| Example:         | IORIA i<br>before executing instruction:<br>i=0x50, ACC=0xAA, Z=0.<br>after executing instruction:<br>ACC=0xFA, Z=0. |

| <b>IOSTR</b>     | <b>Move F-page SFR to ACC</b>  |
|------------------|--|
| Syntax:          | IOSTR F  |
| Operand:         | $5 \leq F \leq 15$   |
| Operation:       | F-page SFR $\rightarrow$ ACC   |
| Status affected: | --   |
| Description:     | Move F-page SFR F to ACC.  |
| Cycle:           | 1  |
| Example:         | IOSTR F<br>before executing instruction:<br>F=0x55, ACC=0xAA.<br>after executing instruction:<br>F=0x55, ACC=0x55. |

| <b>IOST</b>      | <b>Load F-page SFR from ACC</b>   |
|------------------|---|
| Syntax:          | IOST F  |
| Operand:         | $0 \leq F \leq 15$  |
| Operation:       | $ACC \rightarrow$ F-page SFR  |
| Status affected: | --  |
| Description:     | F-page SFR F is loaded by content of ACC.   |
| Cycle:           | 1   |
| Example:         | IOST F<br>before executing instruction:<br>F=0x55, ACC=0xAA.<br>after executing instruction:<br>F=0xAA, ACC=0xAA. |

| <b>LCALL</b>     | <b>Call Subroutine</b>  |
|------------------|---|
| Syntax:          | LCALL adr   |
| Operand:         | $0 \leq \text{adr} \leq 1023$   |
| Operation:       | $PC + 1 \rightarrow$ Top of Stack,<br>$\text{adr} \rightarrow PC[9:0]$  |
| Status affected: | --  |
| Description:     | The return address (PC + 1) is pushed onto top of Stack. The 10-bit immediate address adr is loaded into PC[9:0].   |
| Cycle:           | 2   |
| Example:         | LCALL SUB<br>before executing instruction:<br>PC=A0. Stack level=1<br>after executing instruction:<br>PC=address of SUB, Stack[1]=<br>A0+1, Stack pointer =2. |



| <b>LGOTO</b>     | <b>Unconditional Branch</b>  |
|------------------|--|
| Syntax:          | LGOTO adr  |
| Operand:         | $0 \leq \text{adr} \leq 1023$  |
| Operation:       | $\text{adr} \rightarrow \text{PC}[9:0]$ .  |
| Status affected: | --   |
| Description:     | LGOTO is an unconditional branch instruction. The 10-bit immediate address adr is loaded into PC[9:0].         |
| Cycle:           | 2  |
| Example:         | LGOTO Level<br>before executing instruction:<br>PC=A0.<br>after executing instruction:<br>PC=address of Level. |

| <b>MOVIA</b>     | <b>Move Immediate to ACC</b>   |
|------------------|--|
| Syntax:          | MOVIA i  |
| Operand:         | $0 \leq i < 255$   |
| Operation:       | $i \rightarrow \text{ACC}$   |
| Status affected: | --   |
| Description:     | The content of ACC is loaded with 8-bit immediate data i.  |
| Cycle:           | 1  |
| Example:         | MOVIA i<br>before executing instruction:<br>i=0x55, ACC=0xAA.<br>after executing instruction:<br>ACC=0x55. |

| <b>MOVAR</b>     | <b>Move ACC to R</b>   |
|------------------|--|
| Syntax:          | MOVAR R  |
| Operand:         | $0 \leq R \leq 63$   |
| Operation:       | $\text{ACC} \rightarrow R$   |
| Status affected: | --   |
| Description:     | Move content of ACC to R.  |
| Cycle:           | 1  |
| Example:         | MOVAR R<br>before executing instruction:<br>R=0x55, ACC=0xAA.<br>after executing instruction:<br>R=0xAA, ACC=0xAA. |

| <b>MOVR</b>      | <b>Move R to ACC or R</b>   |
|------------------|---|
| Syntax:          | MOVR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1$ .  |
| Operation:       | $R \rightarrow \text{dest}$   |
| Status affected: | Z   |
| Description:     | The content of R is move to destination. If d is 0, destination is ACC. If d is 1, destination is R and it can be used to check whether R is zero according to status flag Z after execution. |
| Cycle:           | 1   |
| Example:         | MOVR R, d<br>before executing instruction:<br>R=0x0, ACC=0xAA, Z=0, d=0.<br>after executing instruction:<br>R=0x0, ACC=0x00, Z=1.   |

| <b>NOP</b>       | <b>No Operation</b>  |
|------------------|--|
| Syntax:          | NOP  |
| Operand:         | --   |
| Operation:       | No operation.  |
| Status affected: | --   |
| Description:     | No operation.  |
| Cycle:           | 1  |
| Example:         |  |
|                  | NOP<br>before executing instruction:<br>PC=A0<br>after executing instruction:<br>PC=A0+1 |

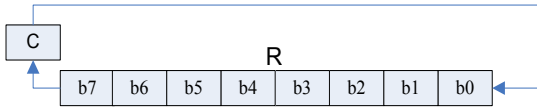
| <b>RETIA</b>     | <b>Return with Data in ACC</b>  |
|------------------|---|
| Syntax:          | RETIA i   |
| Operand:         | $0 \leq i < 255$  |
| Operation:       | $i \rightarrow \text{ACC}$ ,<br>Top of Stack $\rightarrow$ PC   |
| Status affected: | --  |
| Description:     | ACC is loaded with 8-bit immediate data i and PC is loaded from top of Stack as return address and GIE is set to 1.   |
| Cycle:           | 2   |
| Example:         | RETIA i<br>before executing instruction:<br>GIE=0, Stack pointer =2. i=0x55,<br>ACC=0xAA.<br>after executing instruction:<br>GIE=1, PC=Stack[2], Stack<br>pointer =1. ACC=0x55. |

| <b>RETIE</b>     | <b>Return from Interrupt and Enable Interrupt Globally</b>   |
|------------------|--|
| Syntax:          | RETIE  |
| Operand:         | --   |
| Operation:       | Top of Stack $\rightarrow$ PC<br>$1 \rightarrow$ GIE   |
| Status affected: | --   |
| Description:     | The PC is loaded from top of Stack as return address and GIE is set to 1.  |
| Cycle:           | 2  |
| Example:         | RETIE<br>before executing instruction:<br>GIE=0, Stack level=2.<br>after executing instruction:<br>GIE=1, PC=Stack[2], Stack<br>pointer=1. |

| <b>RET</b>       | <b>Return from Subroutine</b>   |
|------------------|---|
| Syntax:          | RET   |
| Operand:         | --  |
| Operation:       | Top of Stack $\rightarrow$ PC   |
| Status affected: | --  |
| Description:     | PC is loaded from top of Stack as return address.   |
| Cycle:           | 2   |
| Example:         | RET<br>before executing instruction:<br>Stack level=2.<br>after executing instruction:<br>PC=Stack[2], Stack level=1. |

**RLR Rotate Left R Through Carry**

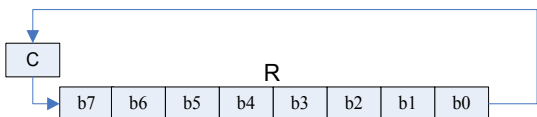
Syntax: RLR R, d  
 Operand:  $0 \leq R \leq 63$   
 $d = 0, 1.$   
 Operation:  $C \rightarrow \text{dest}[0], R[7] \rightarrow C,$   
 $R[6:0] \rightarrow \text{dest}[7:1]$



Status affected: C  
 Description: The content of R is rotated one bit to the left through flag Carry. If d is 0, the result is placed in ACC. If d is 1, the result is stored back to R.  
 Cycle: 1  
 Example: RLR R, d  
 before executing instruction:  
 $R=0xA5, d=1, C=0.$   
 after executing instruction:  
 $R=0x4A, C=1.$

**RRR Rotate Right R Through Carry**

Syntax: RRR R, d  
 Operand:  $0 \leq R \leq 63$   
 $d = 0, 1.$   
 Operation:  $C \rightarrow \text{dest}[7], R[7:1] \rightarrow \text{dest}[6:0],$   
 $R[0] \rightarrow C$



Status affected: C  
 Description: The content of R is rotated one bit to the right through flag Carry. If d is 0, the result is placed in ACC. If d is 1, the result is stored back to R.  
 Cycle: 1  
 Example: RRR R, d  
 before executing instruction:  
 $R=0xA5, d=1, C=0.$   
 after executing instruction:  
 $R=0x52, C=1.$

**SBCAR Subtract ACC and Carry from R**

Syntax: SBCAR R, d  
 Operand:  $0 \leq R \leq 63$   
 $d = 0, 1.$   
 Operation:  $R + (\sim\text{ACC}) + C \rightarrow \text{dest}$   
 Status affected: Z, DC, C

Description: Subtract ACC and Carry from R with 2's complement representation. If d is 0, the result is placed in ACC. If d is 1, the result is stored back to R.

Cycle: 1

Example: SBCAR R, d

- (a) before executing instruction:  
 $R=0x05, \text{ACC}=0x06, d=1,$   
 $C=0.$   
 after executing instruction:  
 $R=0xFE, C=0. (-2)$
- (b) before executing instruction:  
 $R=0x05, \text{ACC}=0x06, d=1,$   
 $C=1.$   
 after executing instruction:  
 $R=0xFF, C=0. (-1)$
- (c) before executing instruction:  
 $R=0x06, \text{ACC}=0x05, d=1,$   
 $C=0.$   
 after executing instruction:  
 $R=0x00, C=1. (-0), Z=1.$
- (d) before executing instruction:  
 $R=0x06, \text{ACC}=0x05, d=1,$   
 $C=1.$   
 after executing instruction:  
 $R=0x1, C=1. (+1)$

---

**SBCIA      Subtract ACC and Carry from Immediate**


---

Syntax:            SBCIA    i

Operand:           $0 \leq i < 255$

Operation:         $i + (\sim\text{ACC}) + C \rightarrow \text{dest}$

Status affected:   Z, DC, C

Description:      Subtract ACC and Carry from 8-bit immediate data i with 2's complement representation. The result is placed in ACC.

Cycle:            1

Example:          SBCIA i

(a) before executing instruction:  
i=0x05, ACC=0x06, C=0.  
after executing instruction:  
ACC=0xFE, C=0. (-2)

(b) before executing instruction:  
i=0x05, ACC=0x06, C=1.  
after executing instruction:  
ACC=0xFF, C=0. (-1)

(c) before executing instruction:  
i=0x06, ACC=0x05, C=0.  
after executing instruction:  
ACC=0x00, C=1. (-0), Z=1.

(d) before executing instruction:  
i=0x06, ACC=0x05, C=1.  
after executing instruction:  
ACC=0x1, C=1. (+1)

---

**SFUN      Load S-page SFR to ACC**


---

Syntax:            SFUN    S

Operand:           $0 \leq S \leq 15$

Operation:         $\text{ACC} \rightarrow \text{S-page SFR}$

Status affected:   --

Description:      S-page SFR S is loaded by content of ACC.

Cycle:            1

Example:          SFUN S

before executing instruction:  
S=0x55, ACC=0xAA.

after executing instruction:  
S=0xAA, ACC=0xAA.

---

**SFUNR      Move S-page SFR from ACC**


---

Syntax:            SFUNR   S

Operand:           $0 \leq S \leq 15$

Operation:        S-page SFR  $\rightarrow$  ACC

Status affected:   --

Description:      Move S-page SFR S to ACC.

Cycle:            1

Example:          SFUNR S

before executing instruction:  
S=0x55, ACC=0xAA.

after executing instruction:  
S=0x55, ACC=0x55.

---

**SLEEP      Enter Halt Mode**


---

Syntax:            SLEEP

Operand:          --

Operation:        00h  $\rightarrow$  WDT,  
00h  $\rightarrow$  WDT prescaler  
1  $\rightarrow$  /TO  
0  $\rightarrow$  /PD

Status affected:   /TO, /PD

Description:      WDT and Prescaler0 are clear to 0. /TO is set to 1 and /PD is clear to 0.  
IC enter Halt mode.

Cycle:            1

Example:          SLEEP

before executing instruction:  
/PD=1, /TO=0.

after executing instruction:  
/PD=0, /TO=1.

| <b>SUBAR</b>     | <b>Subtract ACC from R</b>   |
|------------------|--|
| Syntax:          | SUBAR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$  |
| Operation:       | $R - ACC \rightarrow dest$   |
| Status affected: | Z, DC, C   |
| Description:     | Subtract ACC from R with 2's complement representation. If d is 0, the result is placed in ACC. If d is 1, the result is stored back to R.   |
| Cycle:           | 1  |
| Example:         | SUBAR R, d<br>(a) before executing instruction:<br>R=0x05, ACC=0x06, d=1.<br>after executing instruction:<br>R=0xFF, C=0. (-1)<br>(b) before executing instruction:<br>R=0x06, ACC=0x05, d=1.<br>after executing instruction:<br>R=0x01, C=1. (+1) |

| <b>SWAPR</b>     | <b>Swap High/Low Nibble in R</b>   |
|------------------|--|
| Syntax:          | SWAPR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$  |
| Operation:       | $R[3:0] \rightarrow dest[7:4].$<br>$R[7:4] \rightarrow dest[3:0]$  |
| Status affected: | --   |
| Description:     | The high nibble and low nibble of R is exchanged. If d is 0, the result is placed in ACC. If d is 1, the result is stored back to R. |
| Cycle:           | 1  |
| Example:         | SWAPR R, d<br>before executing instruction:<br>R=0xA5, d=1.<br>after executing instruction:<br>R=0x5A.                               |

| <b>SUBIA</b>     | <b>Subtract ACC from Immediate</b>   |
|------------------|--|
| Syntax:          | SUBIA i  |
| Operand:         | $0 \leq i < 255$   |
| Operation:       | $i - ACC \rightarrow ACC$  |
| Status affected: | Z, DC, C   |
| Description:     | Subtract ACC from 8-bit immediate data i with 2's complement representation. The result is placed in ACC.  |
| Cycle:           | 1  |
| Example:         | SUBIA i<br>(a) before executing instruction:<br>i=0x05, ACC=0x06.<br>after executing instruction:<br>ACC=0xFF, C=0. (-1)<br>(b) before executing instruction:<br>i=0x06, ACC=0x05, d=1, .<br>after executing instruction:<br>ACC=0x01, C=1. (+1) |

| <b>TABLEA</b>    | <b>Read ROM data</b>  |
|------------------|---|
| Syntax:          | TABLEA  |
| Operand:         | --  |
| Operation:       | ROM data{ TBHP, ACC } [7:0]<br>$\rightarrow ACC$<br>ROM data{TBHP, ACC} [13:8]<br>$\rightarrow TBHD.$   |
| Status affected: | --  |
| Description:     | The 8 least significant bits of ROM pointed by {TBHP[2:0], ACC} is placed to ACC.<br>The 6 most significant bits of ROM pointed by {TBHP[2:0], ACC} is placed to TBHD[5:0]. |
| Cycle:           | 2   |
| Example:         | TABLEA<br>before executing instruction:<br>TBHP=0x02, CC=0x34.<br>TBHD=0x01.<br>ROM data[0x234]=0x35AA.<br>after executing instruction:<br>TBHD=0x35, ACC=0xAA.             |

| <b>SWAPR</b> | <b>Swap High/Low Nibble in R</b> |
|--------------|----------------------------------|
|--------------|----------------------------------|

| <b>T0MD</b>      | <b>Load ACC to T0MD</b>   |
|------------------|---|
| Syntax:          | T0MD  |
| Operand:         | --  |
| Operation:       | ACC → T0MD  |
| Status affected: | --  |
| Description:     | The content of T0MD is loaded by ACC.   |
| Cycle:           | 1   |
| Example:         | T0MD<br>before executing instruction:<br>T0MD=0x55, ACC=0xAA.<br>after executing instruction:<br>T0MD=0xAA. |

| <b>XORAR</b>     | <b>Exclusive-OR ACC with R</b>   |
|------------------|--|
| Syntax:          | XORAR R, d   |
| Operand:         | $0 \leq R \leq 63$<br>$d = 0, 1.$  |
| Operation:       | $ACC \oplus R \rightarrow dest$  |
| Status affected: | Z  |
| Description:     | Exclusive-OR ACC with R. If d is 0, the result is placed in ACC. If d is 1, the result is stored back to R.      |
| Cycle:           | 1  |
| Example:         | XORAR R, d<br>before executing instruction:<br>R=0xA5, ACC=0xF0, d=1.<br>after executing instruction:<br>R=0x55. |

| <b>T0MDR</b>     | <b>Move T0MD to ACC</b>   |
|------------------|---|
| Syntax:          | T0MDR   |
| Operand:         | --  |
| Operation:       | T0MD → ACC  |
| Status affected: | --  |
| Description:     | Move the content of T0MD to ACC.  |
| Cycle:           | 1   |
| Example:         | T0MDR<br>before executing instruction<br>T0MD=0x55, ACC=0xAA.<br>after executing instruction<br>ACC=0x55. |

| <b>XORIA</b>     | <b>Exclusive-OR Immediate with ACC</b>   |
|------------------|--|
| Syntax:          | XORIA i  |
| Operand:         | $0 \leq i < 255$   |
| Operation:       | $ACC \oplus i \rightarrow ACC$   |
| Status affected: | Z  |
| Description:     | Exclusive-OR ACC with 8-bit immediate data i. The result is stored in ACC.                                 |
| Cycle:           | 1  |
| Example:         | XORIA i<br>before executing instruction:<br>i=0xA5, ACC=0xF0.<br>after executing instruction:<br>ACC=0x55. |

**5. Configuration Words**

| Item | Name                      | Options   |          |   |          |          |
|------|---------------------------|---|----------|---|----------|----------|
| 1    | High Oscillator Frequency | 1. I_HRC  | 2. E_HXT | 3. E_XT   |          |          |
| 2    | Low Oscillator Frequency  | 1. I_LRC  | 2. E_LXT |   |          |          |
| 3    | High IRC Frequency        | 1. 1MHz   | 2. 2MHz  | 3. 4MHz   |          |          |
|      |                           | 4. 8MHz   | 5. 16MHz | 6. 20MHz  |          |          |
| 4    | High Crystal Oscillator   | 1. $6\text{MHz} < F_{\text{HOSC}} \leq 8\text{MHz}$   |          | 2. $8\text{MHz} < F_{\text{HOSC}} \leq 10\text{MHz}$  |          |          |
|      |                           | 3. $10\text{MHz} < F_{\text{HOSC}} \leq 12\text{MHz}$ |          | 4. $12\text{MHz} < F_{\text{HOSC}} \leq 16\text{MHz}$ |          |          |
|      |                           | 5. $16\text{MHz} < F_{\text{HOSC}} \leq 20\text{MHz}$ |          |   |          |          |
| 5    | Instruction Clock         | 1. 2 oscillator period                                |          | 2. 4 oscillator period                                |          |          |
| 6    | WDT                       | 1. Watchdog Enable (Software control)                 |          |   |          |          |
|      |                           | 2. Watchdog Disable (Always disable)                  |          |   |          |          |
| 7    | WDT Event                 | 1. Watchdog Reset                                     |          | 2. Watchdog Interrupt                                 |          |          |
| 8    | Timer0 source             | 1. EX_CK1   |          | 2. Low Oscillator                                     |          |          |
| 9    | PB.3                      | 1. PB.3 is I/O  |          | 2. PB.3 is reset                                      |          |          |
| 10   | PB.4                      | 1. PB.4 is I/O  |          | 2. PB.4 is instruction clock output                   |          |          |
| 11   | Startup Time              | 1. 140us  | 2. 4.5ms | 3. 18ms   | 4. 72ms  | 5. 288ms |
| 12   | WDT Time Base             | 1. 3.5ms  | 2. 15ms  | 3. 60ms   | 4. 250ms |          |
| 13   | LVR Setting               | 1. Register Control                                   |          | 2. LVR Always On                                      |          |          |
| 14   | LVR Voltage               | 1. 1.6V   | 2. 1.8V  | 3. 2.0V   | 4. 2.2V  | 5. 2.4V  |
|      |                           | 6. 2.7V   | 7. 3.0V  | 8. 3.3V   | 9. 3.6V  | 10. 4.2V |
| 15   | VDD Voltage               | 1. 3.0V   | 2. 4.5V  | 3. 5.0V   |          |          |
| 16   | Read Output Data          | 1. I/O Port   |          | 2. Register   |          |          |
| 17   | E_LXT Backup Control      | 1. Auto Off   |          | 2. Register Off                                       |          |          |
| 18   | EX_CK1 to Inst. Clock     | 1. Sync   |          | 2. Async  |          |          |
| 19   | Startup Clock             | 1. Fast (I_HRC/E_HXT/E_XT)                            |          | 2. Slow (I_LRC/E_LXT)                                 |          |          |
| 20   | PWM1 Output Pin           | 1. PB.6   |          | 2. PB.2   |          |          |
| 21   | PWM2 Output Pin           | 1. PA.1   |          | 2. PA.0   |          |          |
| 22   | PWM3 Output Pin           | 1. PB.1   |          | 2. PA.3   |          |          |
| 23   | PWM4 Output Pin           | 1. PB.0   |          | 2. PA.2   |          |          |
| 24   | PWM Resolution            | 1. 10-bit   |          | 2. 8-bit  |          |          |

| Item | Name                            | Options   |                      |
|------|---------------------------------|-----------|----------------------|
| 25   | Buzzer Output Pin               | 1. PB.7   | 2. PB.2              |
| 26   | Input Schmitt Trigger           | 1. Enable | 2. Disable (1/2 VDD) |
| 27   | Input High Voltage ( $V_{IH}$ ) | 1. 0.7VDD | 2. 0.5VDD            |
| 28   | Input Low Voltage ( $V_{IL}$ )  | 1. 0.3VDD | 2. 0.2VDD            |

Table 21 Configuration Words



## 6. Electrical Characteristics

### 6.1 Absolute Maximum Rating

| Symbol            | Parameter             | Rated Value                   | Unit |
|-------------------|-----------------------|-------------------------------|------|
| $V_{DD} - V_{SS}$ | Supply voltage        | -0.5 ~ +6.0                   | V    |
| $V_{IN}$          | Input voltage         | $V_{SS}-0.3V \sim V_{DD}+0.3$ | V    |
| $T_{OP}$          | Operating Temperature | -40 ~ +85                     | °C   |
| $T_{ST}$          | Storage Temperature   | -40 ~ +125                    | °C   |

### 6.2 DC Characteristics

(All refer  $F_{INST}=F_{HOSC}/4$ ,  $F_{HOSC}=16MHz@I\_HRC$ , WDT enabled, ambient temperature  $T_A=25^\circ C$  unless otherwise specified.)

| Symbol                                  | Parameter           | $V_{DD}$ | Min.                                    | Typ. | Max. | Unit | Condition  |
|---|---------------------|----------|---|------|------|------|--|
| $V_{DD}$                                | Operating voltage   | --       | 3.3                                     | --   | 5.5  | V    | $F_{INST}=20MHz @ I\_HRC/2$                          |
|   |                     |          | 2.2                                     |      |      |      | $F_{INST}=20MHz @ I\_HRC/4$                          |
|   |                     |          | 2.7                                     |      |      |      | $F_{INST}=16MHz @ E\_HXT/2$                          |
|   |                     |          | 1.8                                     |      |      |      | $F_{INST}=16MHz @ E\_HXT/4$                          |
|   |                     |          | 1.8                                     |      |      |      | $F_{INST}=8MHz @ I\_HRC/4 \& I\_HRC/2$               |
|   |                     |          | 1.8                                     |      |      |      | $F_{INST}=8MHz @ E\_HXT/4 \& E\_HXT/2$               |
|   |                     |          |   |      |      |      | $F_{INST}=4MHz @ I\_HRC/4 \& I\_HRC/2$               |
|   |                     |          | $F_{INST}=4MHz @ E\_XT/4 \& E\_XT/2$    |      |      |      |  |
|   |                     |          | $F_{INST}=32KHz @ I\_LRC/4 \& I\_LRC/2$ |      |      |      |  |
| $F_{INST}=32KHz @ E\_LXT/4 \& E\_LXT/2$ |                     |          |   |      |      |      |  |
| $V_{IH}$                                | Input high voltage  | 5V       | 4.0                                     | --   | --   | V    | RSTb (0.8 $V_{DD}$ )                                 |
|   |                     | 3V       | 2.4                                     | --   | --   |      |  |
|   |                     | 5V       | 3.5                                     | --   | --   | V    | All other I/O pins, EX_CKI, INT CMOS (0.7 $V_{DD}$ ) |
|   |                     | 3V       | 2.1                                     | --   | --   |      |  |
|   |                     | 5V       | 2.5                                     | --   | --   | V    | All other I/O pins, EX_CKI TTL (0.5 $V_{DD}$ )       |
|   |                     | 3V       | 1.5                                     | --   | --   |      |  |
| $V_{IL}$                                | Input low voltage   | 5V       | --                                      | --   | 1.0  | V    | RSTb (0.2 $V_{DD}$ )                                 |
|   |                     | 3V       | --                                      | --   | 0.6  |      |  |
|   |                     | 5V       | --                                      | --   | 1.5  | V    | All other I/O pins, EX_CKI, INT CMOS (0.3 $V_{DD}$ ) |
|   |                     | 3V       | --                                      | --   | 0.9  |      |  |
|   |                     | 5V       | --                                      | --   | 1.0  | V    | All other I/O pins, EX_CKI TTL (0.2 $V_{DD}$ )       |
|   |                     | 3V       | --                                      | --   | 0.6  |      |  |
| $I_{OH}$                                | Output high current | 5V       | --                                      | -20  | --   | mA   | $V_{OH}=4.0V$  |
|   |                     | 3V       | --                                      | -10  | --   |      | $V_{OH}=2.0V$  |
| $I_{OL}$                                | Output low current  | 5V       | --                                      | 40   | --   | mA   | $V_{OL}=1.0V$  |
|   |                     | 3V       | --                                      | 26   | --   |      |  |
| $I_{IR}$                                | IR sink current     | 5V       | --                                      | 40   | --   | mA   | Normal IR  |
|   |                     | 3V       | --                                      | 26   | --   |      |  |

| Symbol            | Parameter          | V <sub>DD</sub>    | Min. | Typ. | Max. | Unit | Condition   |
|-------------------|--------------------|--------------------|------|------|------|------|---|
| I <sub>OP</sub>   | Operating current  | <b>Normal Mode</b> |      |      |      |      |   |
|                   |                    | 5V                 | --   | 2.7  | --   | mA   | F <sub>HOSC</sub> =20MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2                        |
|                   |                    | 3V                 | --   | 1.3  | --   |      |   |
|                   |                    | 5V                 | --   | 2.1  | --   | mA   | F <sub>HOSC</sub> =20MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4                        |
|                   |                    | 3V                 | --   | 1.0  | --   |      |   |
|                   |                    | 5V                 | --   | 2.4  | --   | mA   | F <sub>HOSC</sub> =16MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2                        |
|                   |                    | 3V                 | --   | 1.1  | --   |      |   |
|                   |                    | 5V                 | --   | 1.9  | --   | mA   | F <sub>HOSC</sub> =16MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4                        |
|                   |                    | 3V                 | --   | 0.8  | --   |      |   |
|                   |                    | 5V                 | --   | 1.5  | --   | mA   | F <sub>HOSC</sub> =8MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2                         |
|                   |                    | 3V                 | --   | 0.7  | --   |      |   |
|                   |                    | 5V                 | --   | 1.3  | --   | mA   | F <sub>HOSC</sub> =8MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4                         |
|                   |                    | 3V                 | --   | 0.5  | --   |      |   |
|                   |                    | 5V                 | --   | 1.1  | --   | mA   | F <sub>HOSC</sub> =4MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2                         |
|                   |                    | 3V                 | --   | 0.5  | --   |      |   |
|                   |                    | 5V                 | --   | 0.9  | --   | mA   | F <sub>HOSC</sub> =4MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4                         |
|                   |                    | 3V                 | --   | 0.4  | --   |      |   |
|                   |                    | 5V                 | --   | 0.9  | --   | mA   | F <sub>HOSC</sub> =1MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2                         |
|                   |                    | 3V                 | --   | 0.4  | --   |      |   |
|                   |                    | 5V                 | --   | 0.8  | --   | mA   | F <sub>HOSC</sub> =1MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4                         |
|                   |                    | 3V                 | --   | 0.4  | --   |      |   |
|                   |                    | <b>Slow mode</b>   |      |      |      |      |   |
|                   |                    | 5V                 | --   | 6.0  | --   | uA   | F <sub>HOSC</sub> disabled,<br>F <sub>LOSC</sub> =32KHz @ I <sub>LRC</sub> /2               |
|                   |                    | 3V                 | --   | 2.7  | --   |      |   |
|                   |                    | 5V                 | --   | 7.0  | --   | uA   | F <sub>HOSC</sub> disabled,<br>F <sub>LOSC</sub> =32KHz @ E <sub>LXT</sub> /2.              |
|                   |                    | 3V                 | --   | 2.9  | --   |      |   |
|                   |                    | 5V                 | --   | 4.3  | --   | uA   | F <sub>HOSC</sub> disabled,<br>F <sub>LOSC</sub> =32KHz @ I <sub>LRC</sub> /4               |
|                   |                    | 3V                 | --   | 1.8  | --   |      |   |
|                   |                    | 5V                 | --   | 5.3  | --   | uA   | F <sub>HOSC</sub> disabled,<br>F <sub>LOSC</sub> =32KHz @ E <sub>LXT</sub> /4.              |
|                   |                    | 3V                 | --   | 2.0  | --   |      |   |
| I <sub>STB</sub>  | Standby current    | 5V                 | --   | 2.7  | --   | uA   | Standby mode, F <sub>HOSC</sub> disabled,<br>F <sub>LOSC</sub> =32KHz @ I <sub>LRC</sub> /4 |
|                   |                    | 3V                 | --   | 1.1  | --   |      |   |
| I <sub>HALT</sub> | Halt current       | 5V                 | --   | --   | 0.5  | uA   | Halt mode, WDT disabled.  |
|                   |                    | 3V                 | --   | --   | 0.2  |      |   |
|                   |                    | 5V                 | --   | --   | 5.0  | uA   | Halt mode, WDT enabled.   |
|                   |                    | 3V                 | --   | --   | 3.0  |      |   |
| R <sub>PH</sub>   | Pull-High resistor | 5V                 | --   | 50   | --   | kΩ   | Pull-High resistor  |
|                   |                    | 3V                 | --   | 100  | --   |      |   |
| R <sub>PL</sub>   | Pull-Low resistor  | 5V                 | --   | 50   | --   | kΩ   | Pull-Low resistor   |
|                   |                    | 3V                 | --   | 100  | --   |      |   |

### 6.3 OSC Characteristics

(Measurement conditions  $V_{DD}$  Voltage,  $T_A$  Temperature are equal to programming conditions.)

| Parameter                  | Min. | Typ. | Max. | Unit | Condition                             |
|----------------------------|------|------|------|------|---------------------------------------|
| I_HRC deviation by socket  |      |      | ±1   | %    | Socket installed directly on writer.  |
| I_HRC deviation by handler |      |      | ±3   | %    | Handler condition with correct setup. |
| I_LRC deviation by handler |      |      | ±5   | %    |                                       |

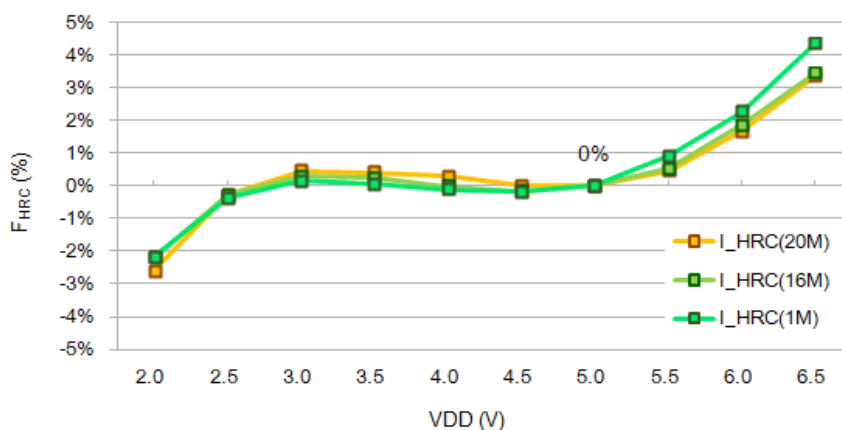
### 6.4 Comparator / LVD Characteristics

( $V_{DD}=5V$ ,  $V_{SS}=0V$ ,  $T_A=25^\circ C$  unless otherwise specified.)

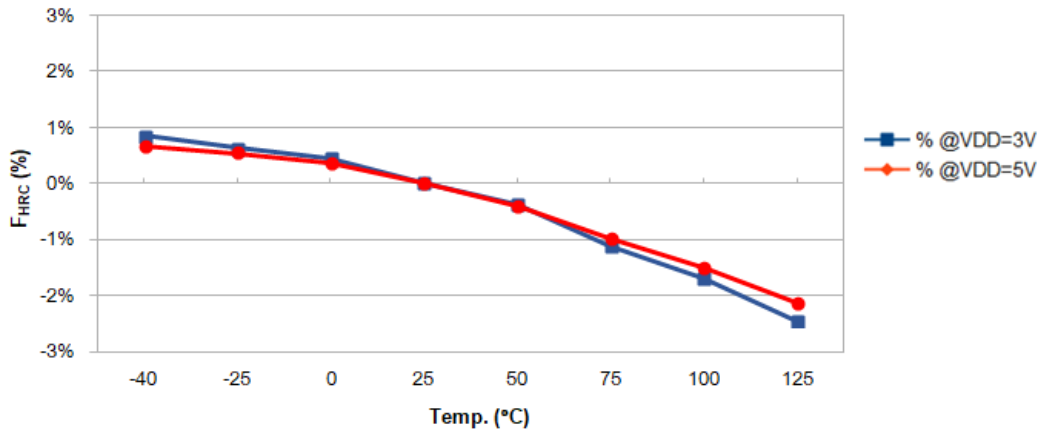
| Symbol    | Parameter                         | Min. | Typ. | Max.         | Unit | Condition                  |
|-----------|-----------------------------------|------|------|--------------|------|----------------------------|
| $V_{IVR}$ | Comparator input voltage range    | 0    | --   | $V_{DD}-1.5$ | V    | $F_{HOSC}=1MHz$            |
| $T_{ENO}$ | Comparator enable to output valid | --   | 20   | --           | us   | $F_{HOSC}=1MHz$            |
| $I_{CO}$  | Operating current of comparator   | --   | 70   | --           | uA   | $F_{HOSC}=1MHz$ , P2V mode |
| $I_{LVD}$ | Operating current of LVD          | --   | 85   | --           | uA   | $F_{HOSC}=1MHz$ , LVD=4.3V |
| $E_{LVD}$ | LVD voltage error                 | --   | --   | 3            | %    | $F_{HOSC}=1MHz$ , LVD=4.3V |

### 6.5 Characteristic Graph

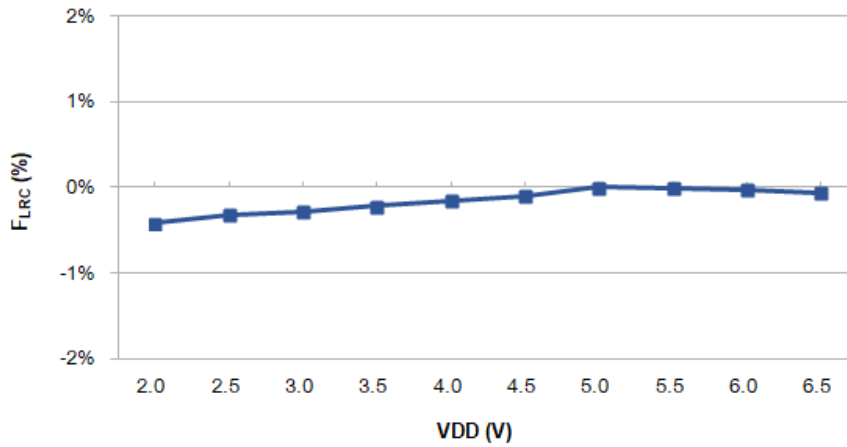
#### 6.5.1 Frequency vs. $V_{DD}$ of I\_HRC



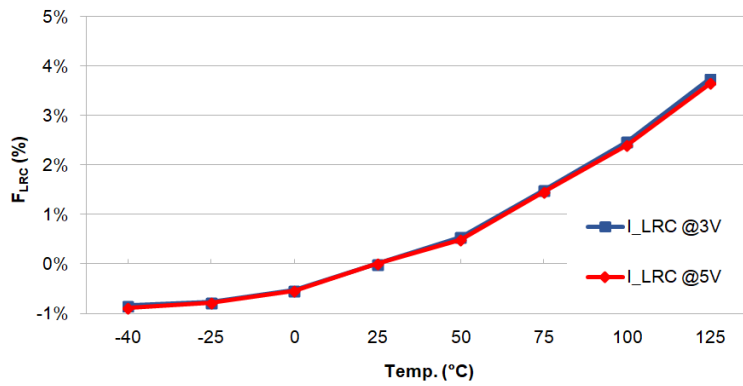
6.5.2 Frequency vs. Temperature of I\_HRC



6.5.3 Frequency vs. V<sub>DD</sub> of I\_LRC



6.5.4 Frequency vs. Temperature of I\_LRC

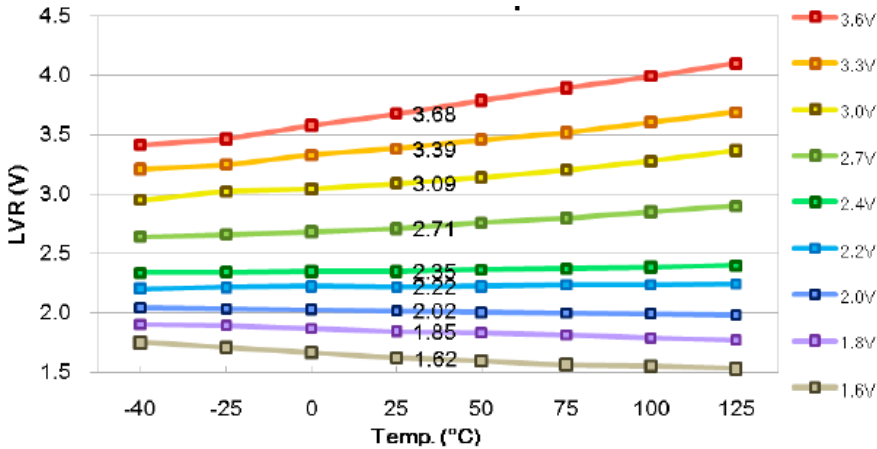


### 6.6 Recommended Operating Voltage

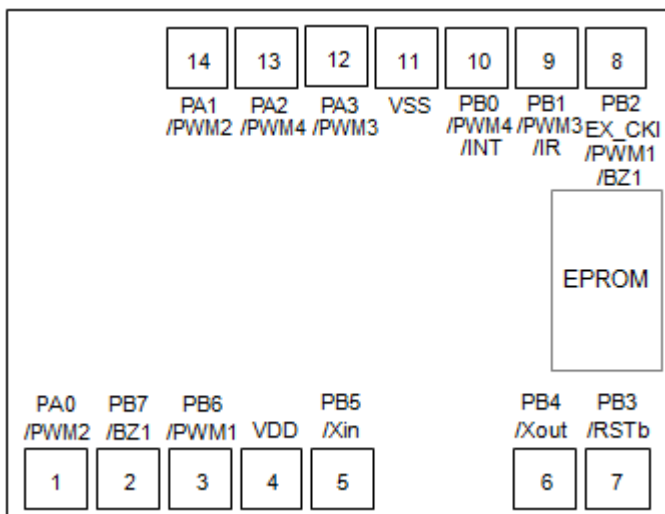
Recommended Operating Voltage (Temperature range: -40 °C ~ +85 °C)

| Frequency     | Min. Voltage | Max. Voltage | LVR : default ( 25 °C) | LVR : Recommended (-40 °C ~ +85 °C) |
|---------------|--------------|--------------|------------------------|-------------------------------------|
| 20M/2T        | 3.3V         | 5.5V         | 3.6V                   | 3.6V                                |
| 16M/2T        | 2.7V         | 5.5V         | 3.0V                   | 3.3V                                |
| 20M/4T        | 2.2V         | 5.5V         | 2.4V                   | 2.7V                                |
| 16M/4T        | 1.8V         | 5.5V         | 2.0V                   | 2.2V                                |
| 8M(2T or 4T)  | 1.8V         | 5.5V         | 2.0V                   | 2.2V                                |
| ≤6M(2T or 4T) | 1.8V         | 5.5V         | 2.0V                   | 2.2V                                |

### 6.7 LVR vs. Temperature

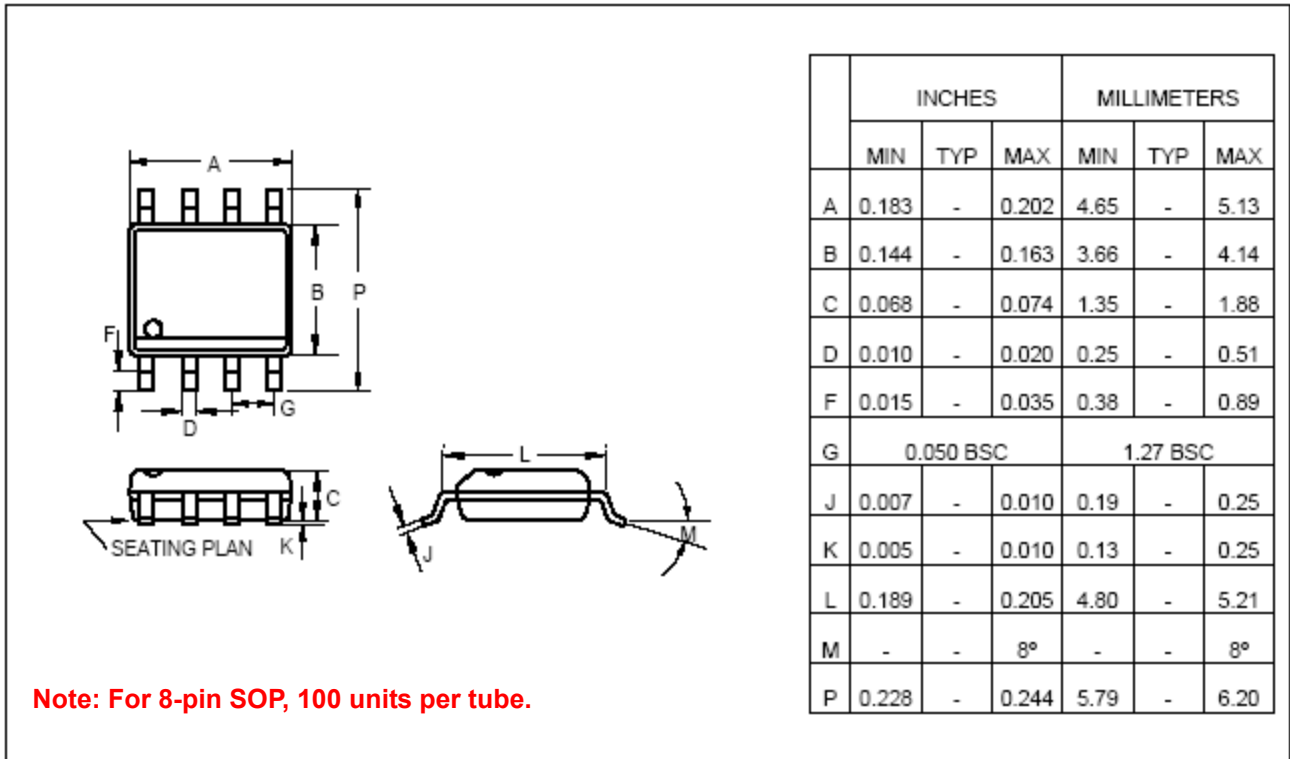


### 7. Die Pad Diagram

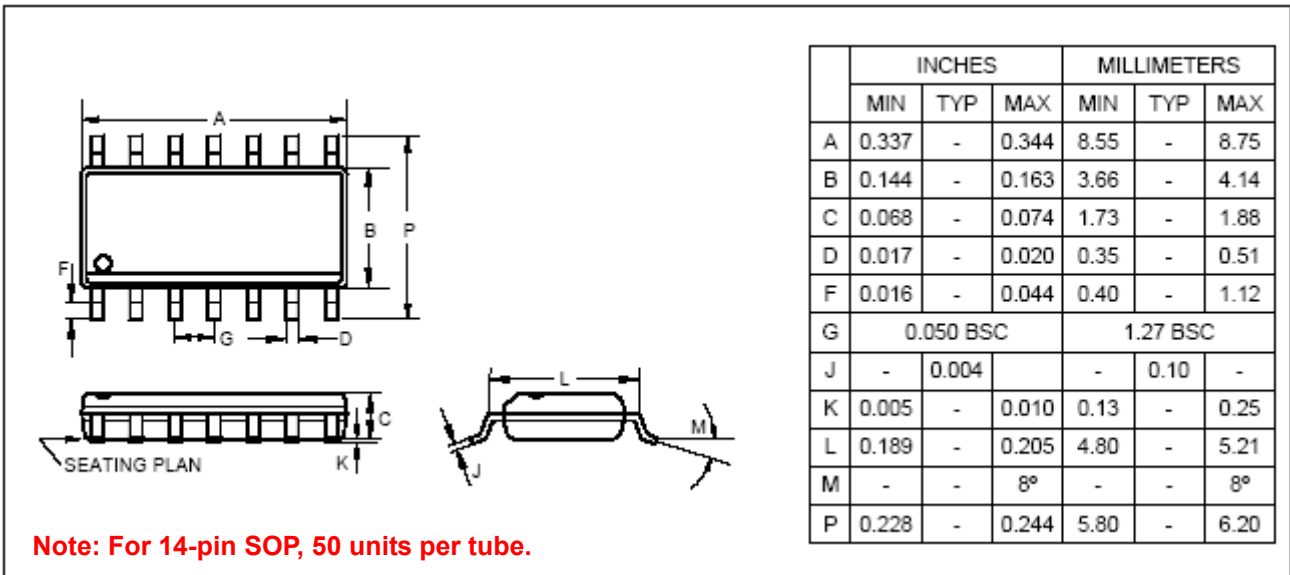


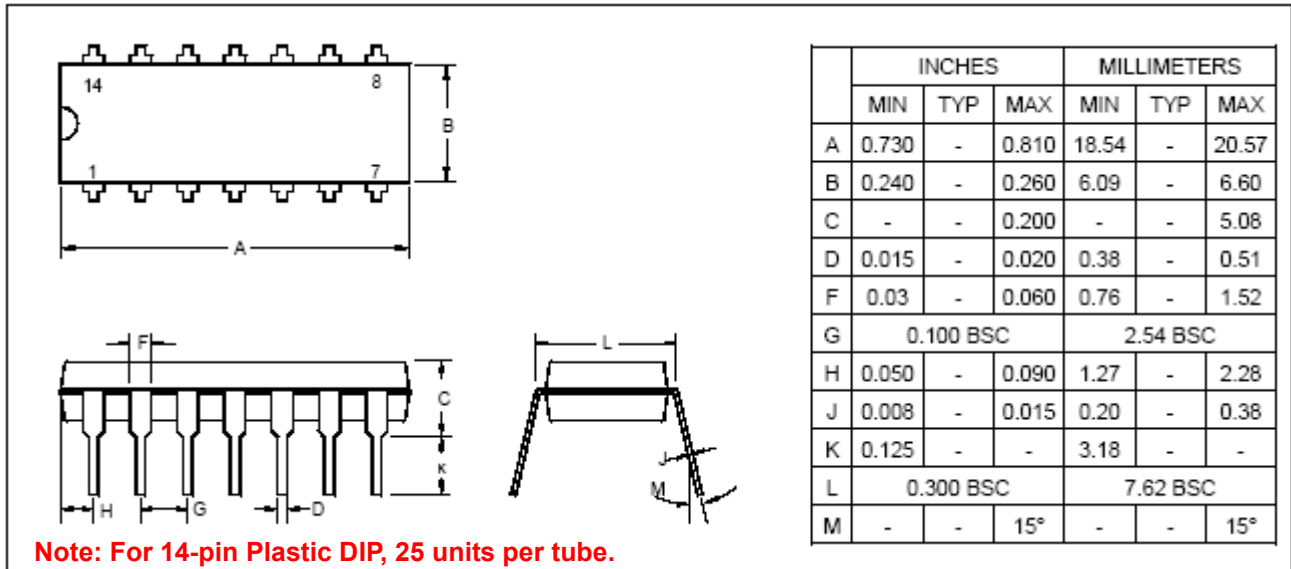
**8. Package Dimension**

**8.1 8-Pin Plastic SOP (150 mil)**



**8.2 14-Pin Plastic SOP (150 mil)**



**8.3 14-Pin Plastic DIP (300 mil)**

**9. Ordering Information**

| <i>P/N</i>  | <i>Package Type</i> | <i>Pin Count</i> | <i>Package Width</i> | <i>Shipping</i>  |
|-------------|---------------------|------------------|----------------------|--|
| NY8A053E    | Die                 | --               | --                   | --   |
| NY8A053ES8  | SOP                 | 8                | 150 mil              | <u>Tape &amp; Reel</u> : 2.5K pcs per Reel<br><u>Tube</u> : 100 pcs per Tube |
| NY8A053ES14 | SOP                 | 14               | 150 mil              | <u>Tape &amp; Reel</u> : 2.5K pcs per Reel<br><u>Tube</u> : 50 pcs per Tube  |
| NY8A053EP14 | PDIP                | 14               | 300 mil              | <u>Tube</u> : 25 pcs per Tube  |