



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

DATA SHEET

# NY8A054E

---

---

**14 I/O + 5-channel PWM 8-bit EPROM-Based MCU**

**Version 0.2 (Preliminary)**

**Jul. 14, 2020**

---

---

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>
0.1	2020/06/21	Preliminary edition	-
0.2	2020/07/14	Revised "6.6 Recommended Operating Voltage" table	99

## Table of Contents

<b>1. 概述 .....</b>	<b>7</b>
1.1 功能 .....	7
1.2 NY8A054E 與 NY8A054D 的主要差異 .....	9
<b>1. General Description.....</b>	<b>10</b>
1.1 Features .....	10
1.2 Major Differences Table Between NY8A054E and NY8A054D .....	12
1.2 Block Diagram .....	13
1.3 Pin Assignment .....	13
1.4 Pin Description.....	14
<b>2. Memory Organization .....</b>	<b>15</b>
2.1 Program Memory .....	15
2.2 Data Memory .....	16
<b>3. Function Description.....</b>	<b>20</b>
3.1 R-page Special Function Register .....	20
3.1.1 <i>INDF (Indirect Addressing Register)</i> .....	20
3.1.2 <i>TMR0 (Timer0 Register)</i> .....	20
3.1.3 <i>PCL (Low Byte of PC[10:0])</i> .....	20
3.1.4 <i>STATUS (Status Register)</i> .....	21
3.1.5 <i>FSR (Register File Selection Register)</i> .....	21
3.1.6 <i>PortA (PortA Data Register)</i> .....	22
3.1.7 <i>PortB (PortB Data Register)</i> .....	22
3.1.8 <i>PCON (Power Control Register)</i> .....	22
3.1.9 <i>BWUCON (PortB Wake-up Control Register)</i> .....	23
3.1.10 <i>PCHBUF (High Byte of PC)</i> .....	23
3.1.11 <i>ABPLCON (PortA/PortB Pull-Low Resistor Control Register)</i> .....	23
3.1.12 <i>BPHCON (PortB Pull-High Resistor Control Register)</i> .....	24
3.1.13 <i>INTE (Interrupt Enable Register)</i> .....	24
3.1.14 <i>INTF (Interrupt Flag Register)</i> .....	25
3.1.15 <i>AWUCON (PortA Wake-up Control Register)</i> .....	25
3.1.16 <i>INTEDG (Interrupt Edge Register)</i> .....	26
3.1.17 <i>TMRH (Timer High Byte Register)</i> .....	26

3.1.18	<i>RFC (RFC Register)</i> .....	26
3.1.19	<i>TM34RH (Timer3 High Byte Register)</i> .....	27
3.1.20	<i>INTE2 (Interrupt Enable and Flag 2nd. Register)</i> .....	27
3.2	<i>T0MD Register</i> .....	28
3.3	<i>F-page Special Function Register</i> .....	29
3.3.1	<i>IOSTA (PortA I/O Control Register)</i> .....	29
3.3.2	<i>IOSTB (PortB I/O Control Register)</i> .....	29
3.3.3	<i>APHCON (PortA Pull-High Resistor Control Register)</i> .....	29
3.3.4	<i>PS0CV (Prescaler0 Counter Value Register)</i> .....	30
3.3.5	<i>BODCON (PortB Open-Drain Control Register)</i> .....	30
3.3.6	<i>CMPCR (Comparator voltage select Control Register)</i> .....	30
3.3.7	<i>PCON1 (Power Control Register1)</i> .....	31
3.4	<i>S-page Special Function Register</i> .....	32
3.4.1	<i>TMR1 (Timer1 Register)</i> .....	32
3.4.2	<i>T1CR1 (Timer1 Control Register1)</i> .....	32
3.4.3	<i>T1CR2 (Timer1 Control Register2)</i> .....	33
3.4.4	<i>PWM1DUTY (PWM1 Duty Register)</i> .....	34
3.4.5	<i>PS1CV (Prescaler1 Counter Value Register)</i> .....	34
3.4.6	<i>BZ1CR (Buzzer1 Control Register)</i> .....	34
3.4.7	<i>IRCR (IR Control Register)</i> .....	35
3.4.8	<i>TBHP (Table Access High Byte Address Pointer Register)</i> .....	36
3.4.9	<i>TBHD (Table Access High Byte Data Register)</i> .....	36
3.4.10	<i>P2CR1 (PWM2 Control Register1)</i> .....	36
3.4.11	<i>PWM2DUTY (PWM2 Duty Register)</i> .....	37
3.4.12	<i>OSCCR (Oscillation Control Register)</i> .....	37
3.4.13	<i>TMR3 (Timer3 Register)</i> .....	37
3.4.14	<i>T3CR1 (Timer3 Control Register1)</i> .....	38
3.4.15	<i>T3CR2 (Timer3 Control Register2)</i> .....	38
3.4.16	<i>PWM3DUTY (PWM3 Duty Register)</i> .....	39
3.4.17	<i>PS3CV (Prescaler3 Counter Value Register)</i> .....	39
3.4.18	<i>P4CR1 (PWM4 Control Register1)</i> .....	40
3.4.19	<i>PWM4DUTY (PWM4 Duty Register)</i> .....	40
3.4.20	<i>P5CR1 (PWM5 Control Register1)</i> .....	40
3.4.21	<i>PWM5DUTY (PWM5 Duty Register)</i> .....	40
3.4.22	<i>PWM5RH (PWM5 High Byte Register)</i> .....	41

3.5	I/O Port .....	41
3.5.1	<i>Block Diagram of IO Pins</i> .....	43
3.6	Timer0.....	53
3.7	Timer1 / PWM1 / Buzzer1 .....	54
3.8	PWM2.....	57
3.9	Timer3 / PWM3.....	57
3.10	PWM4.....	60
3.11	PWM5.....	60
3.12	RFC Mode .....	61
3.13	IR Carrier .....	62
3.14	Low Voltage Detector (LVD) .....	62
3.15	Voltage Comparator .....	63
3.15.1	Comparator Reference Voltage (Vref) .....	64
3.16	Watch-Dog Timer (WDT).....	66
3.17	Interrupt .....	67
3.17.1	<i>Timer0 Overflow Interrupt</i> .....	68
3.17.2	<i>Timer1 Underflow Interrupt</i> .....	68
3.17.3	<i>Timer3 Underflow Interrupt</i> .....	68
3.17.4	<i>WDT Timeout Interrupt</i> .....	68
3.17.5	<i>PA/PB Input Change Interrupt</i> .....	68
3.17.6	<i>External 0 Interrupt</i> .....	68
3.17.7	<i>External 1 Interrupt</i> .....	68
3.17.8	<i>LVD Interrupt</i> .....	68
3.18	Oscillation Configuration .....	69
3.19	Operating Mode .....	71
3.19.1	<i>Normal Mode</i> .....	73
3.19.2	<i>Slow Mode</i> .....	73
3.19.3	<i>Standby Mode</i> .....	74
3.19.4	<i>Halt Mode</i> .....	74
3.19.5	<i>Wake-up Stable Time</i> .....	75
3.19.6	<i>Summary of Operating Mode</i> .....	75
3.20	Reset Process .....	76
<b>4.</b>	<b>Instruction Set .....</b>	<b>78</b>

<b>5. Configuration Words .....</b>	<b>94</b>
<b>6. Electrical Characteristics .....</b>	<b>95</b>
6.1 Absolute Maximum Rating .....	95
6.2 DC Characteristics .....	95
6.3 OSC Characteristics .....	97
6.4 Comparator / LVD Characteristics .....	97
6.5 Characteristic Graph .....	97
6.5.1 Frequency vs. $V_{DD}$ of $I_{HRC}$ .....	97
6.5.2 Frequency vs. Temperature of $I_{HRC}$ .....	98
6.5.3 Frequency vs. $V_{DD}$ of $I_{LRC}$ .....	98
6.5.4 Frequency vs. Temperature of $I_{LRC}$ .....	98
6.6 Recommended Operating Voltage .....	99
6.7 LVR vs. Temperature .....	99
<b>7. Die Pad Diagram .....</b>	<b>100</b>
<b>8. Package Dimension .....</b>	<b>101</b>
8.1 8-Pin Plastic SOP (150 mil) .....	101
8.2 16-Pin Plastic SOP (150 mil) .....	101
<b>9. Ordering Information .....</b>	<b>102</b>

## 1. 概述

NY8A054E 是以EPROM作為記憶體的 8 位元微控制器，專為多組PWM的應用設計。例如燈控，遙控車應用。採用CMOS製程並同時提供客戶低成本、高性能、及高性價比等顯著優勢。NY8A054E 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，NY8A054E 有 14 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，NY8A054E內建了可選擇頻率的紅外載波發射口。

NY8A054E 有三組計時器，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NY8A054E 提供 5 組 10 位元解析度的PWM輸出，1 組蜂鳴器輸出可用來驅動馬達、LED、或蜂鳴器等等。

NY8A054E 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，NY8A054E 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NY8A054E 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。
- 2Kx14 bits EPROM。
- 128 bytes SRAM。
- 14 根可分別單獨控制輸入輸出方向的I/O腳(GPIO)、PA[7:0]、PB[5:0]。
- PA[5:0] 及 PB[3:0] 可選擇輸入時使用內建下拉電阻。
- PA[7: 0] 及 PB[5:0] 可選擇輸入時使用上拉電阻。
- PB[5:0] 可選擇開漏極輸出(Open-Drain)。
- PA[5] 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。
- 二組 10 位元下數計時器(Timer1, 3)可選重複載入或連續下數計時。

- 五個 10 位元脈衝寬度調變(PWM1, 2, 3, 4, 5) · PWM1/2 共用Timer1; PWM3/4/5 共用Timer3。
- 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。
- 內建準確的低電壓偵測電路(LVD)。
- 內建準確的電壓比較器(Voltage Comparator)。
- 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。
- 內建電阻頻率轉換器(RFC)功能。
- 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)。
- 八種硬體中斷：
  - Timer0 溢位中斷。
  - Timer1 借位中斷。
  - Timer3 借位中斷。
  - WDT 中斷。
  - PA/PB 輸入狀態改變中斷。
  - 兩組外部中斷輸入。
  - 低電壓偵測中斷。
- NY8A054E在待機模式(Standby mode)下的八種喚醒中斷：
  - Timer0 溢位中斷。
  - Timer1 借位中斷。
  - Timer3 借位中斷。
  - WDT 中斷。
  - PA/PB 輸入狀態改變中斷。
  - 兩組外部中斷輸入。
  - 低電壓偵測中斷。
- NY8A054E在睡眠模式(Halt mode)下的四種喚醒中斷：
  - WDT 中斷。



- PA/PB 輸入狀態改變中斷。
- 兩組外部中斷輸入。

## 1.2 NY8A054E 與 NY8A054D 的主要差異

Item	Function	NY8A054E	NY8A054D
1	PortA input pull-high resistor	PA[7:0]	PA[7:6], PA[4:0]
2	PortA input pull-low resistor	PA[5:0]	PA[5], PA[3:0]
3	Low Voltage Detector(LVD) voltage level	16 voltage levels	8 voltage levels
4	Comparator structure	shared with LVD	rail-to-rail comparator
5	PortA, PortB wake-up default status	default: disable	default: enable
6	Input Voltage Schmitt Trigger	Enable/Disable	---
7	Recommended Operating Voltage	NY8A054E LVR default and recommended voltage are lower than NY8A054D.	

## 1. General Description

NY8A054E is an EPROM based 8-bit MCU tailored for I/O based applications like light control and remote car application. NY8A054E adopts advanced CMOS technology to provide customers remarkable solution with low cost, high performance. RISC architecture is applied to NY8A054E 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, NY8A054E is very suitable for those applications that are sophisticated but compact program size is required.

As NY8A054E address I/O type applications, it can provide 14 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, NY8A054E has built-in infrared (IR) carrier generator with selectable IR carrier frequency and polarity for applications which demand remote control feature.

NY8A054E also provides 3 sets of timers which can be used as regular timer based on system oscillation or event counter with external trigger clock. Moreover, NY8A054E provides 5 sets of 10-bit resolution Pulse Width Modulation (PWM) output and 1 set of buzzer output in order to drive motor/LED and buzzer.

NY8A054E 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, NY8A054E 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 NY8A054E operates in Standby mode and Halt mode, kinds of event will issue interrupt requests and can wake-up NY8A054E 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.
- 2K x 14 bits EPROM.
- 128 bytes SRAM.
- 14 general purpose I/O pins (GPIO), PA[7:0], PB[5:0], with independent direction control.
- PA[5:0] and PB[3:0] have features of Pull-Low resistor for input pin.
- PA[7: 0] and PB[5:0] have features of Pull-High resistor.
- PB[5:0] has features of Open-Drain output.

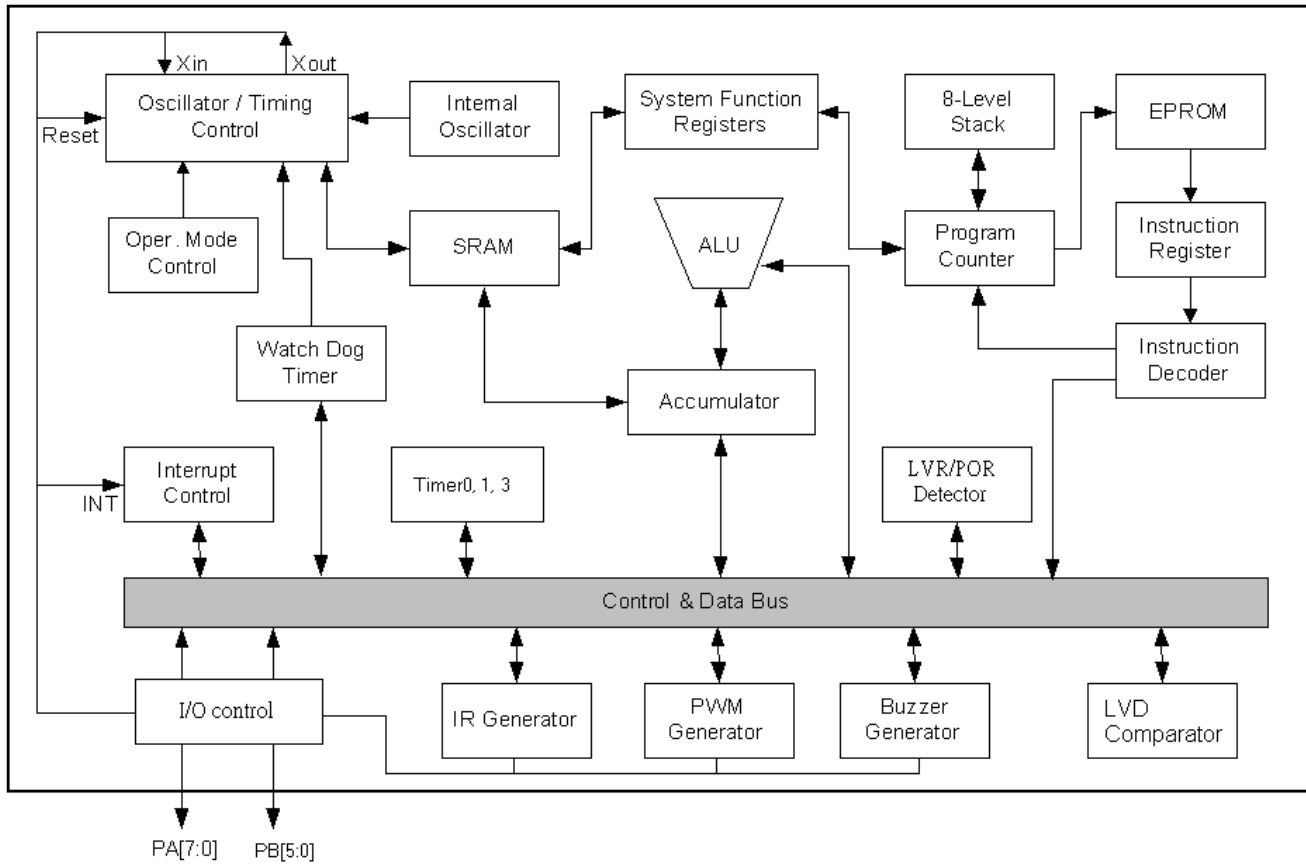
- PA[5] has feature of input or open-drain output.
- 8-level hardware Stack.
- Direct and indirect addressing modes for data access.
- One 8-bit up-count timer (Timer0) with programmable prescaler.
- Five 10-bit reload or continuous down-count timers (Timer1, 3).
- Five 10-bit resolution PWM (PWM1, 2, 3, 4, 5) output. PWM1/2 use Timer1, PWM3/4/5 use Timer3.
- 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.
- Built-in Resistance to Frequency Converter (RFC) function.
- 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.
- Eight hardware interrupt events:
  - Timer0 overflow interrupt.
  - Timer1 underflow interrupt.
  - Timer3 underflow interrupt.
  - WDT timeout interrupt.
  - PA/PB input change interrupt.
  - 2 set External interrupt.
  - LVD interrupt.

- Eight interrupt events to wake-up NY8A054E from Standby mode:
  - Timer0 overflow interrupt.
  - Timer1 underflow interrupt.
  - Timer3 underflow interrupt.
  - WDT timeout interrupt.
  - PA/PB input change interrupt.
  - 2 set External interrupt.
  - LVD interrupt.
- Four interrupt events to wake-up NY8A054E from Halt mode:
  - WDT timeout interrupt.
  - PA/PB input change interrupt.
  - 2 set External interrupt.

## 1.2 Major Differences Table Between NY8A054E and NY8A054D

Item	Function	NY8A054E	NY8A054D
1	PortA input pull-high resistor	PA[7:0]	PA[7:6], PA[4:0]
2	PortA input pull-low resistor	PA[5:0]	PA[5], PA[3:0]
3	Low Voltage Detector(LVD) voltage level	16 voltage levels	8 voltage levels
4	Comparator structure	shared with LVD	rail-to-rail comparator
5	PortA, PortB wake-up default status	default: disable	default: enable
6	Input Voltage Schmitt Trigger	Enable/Disable	---
7	Recommended Operating Voltage	NY8A054E LVR default and recommended voltage are lower than NY8A054D.	

### 1.2 Block Diagram



### 1.3 Pin Assignment

NY8A054E provides THREE kinds of package type which are SOP16 ,SOP14and SOP8 package.

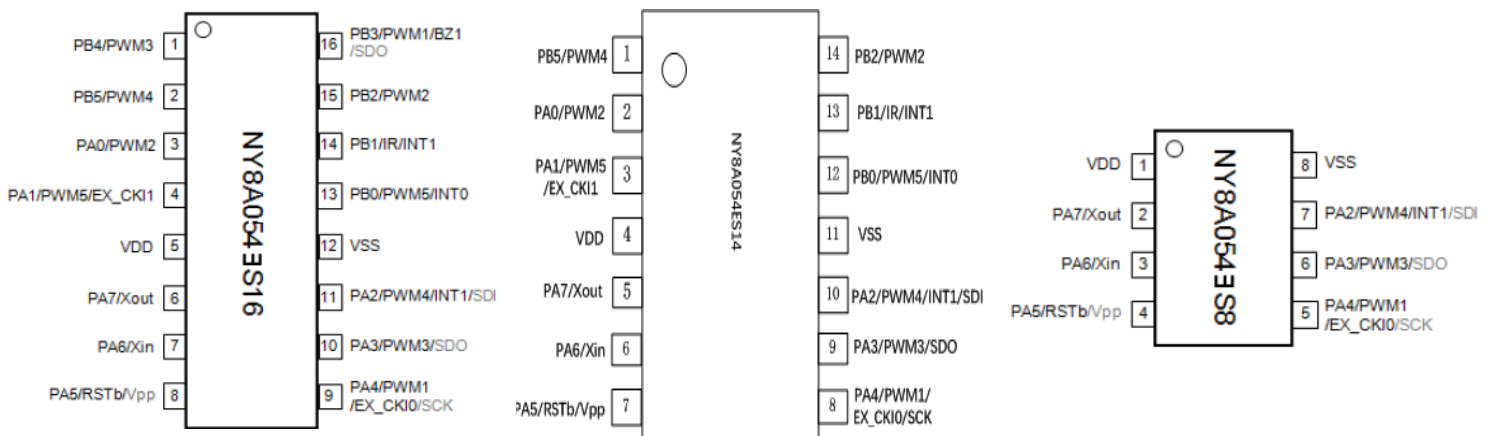


Figure 1 Package pin assignment

**1.4 Pin Description**

Pin Name	I/O	Description
PA0/ PWM2	I/O	PA0 is bidirectional I/O pin, and can be comparator analog input pins. PA0 can be the output of PWM2.
PA1/ PWM5/ EX_CK11	I/O	PA1 is bidirectional I/O pin, and can be comparator analog input pins. PA1 can be the output of PWM5. Moreover, it can be Timer3 clock source EX_CK11.
PA2/ PWM4/ INT1/ SDI	I/O	PA2 is a bidirectional I/O pin, and can be comparator analog input pin. PA2 can be the output of PWM4. PA2 can be the input pin of external interrupt INT1. PA2 can be programming pad SDI.
PA3/ PWM3/ SDO	I/O	PA3 is a bidirectional I/O pin, and can be comparator analog input pin. PA3 can be the output of PWM3 PA3 can be programming pad SDO.
PA4/ PWM1/ EX_CK10/ SCK	I/O	PA4 is a bidirectional I/O pin. PA4 can be the output of PWM1 PA4 can be the Timer0,1 clock source EX_CK10. PA4 can be programming pad SCK.
PA5/ RSTb/ Vpp	I/O	PA5 is an input pin or open-drain output pin. PA5 can be the reset pin RSTb. If this pin is more than 7.75V, it also can make NY8A054E enter EPROM programming mode.
PA6/ Xin	I/O	PA6 is a bidirectional I/O pin. PA6 can be the input pin of crystal oscillator Xin.
PA7/ Xout	I/O	PA7 is a bidirectional I/O pin. PA7 can be the output pin of crystal oscillator Xout. PA7 also can be output of instruction clock.
PB0/ PWM5/ INT0	I/O	PB0 is a bidirectional I/O pin. PB0 can be the output of PWM5. PB0 can be the input pin of external interrupt INT0.
PB1/ IR/ INT1	I/O	PB1 is a bidirectional I/O pin. If IR mode is enabled, this pin is IR carrier output. PB1 can be the input pin of external interrupt INT1.
PB2/ PWM2	I/O	PB2 is a bidirectional I/O pin. PB2 can be the output of PWM2.
PB3/ PWM1/ BZ1/ SDO	I/O	PB3 is a bidirectional I/O pin. PB3 can be the output of Buzzer1 or PWM1 PB3 can be programming pad SDO.
PB4/ PWM3	I/O	PB4 is bidirectional I/O pin. PB4 can be the output of PWM3.
PB5/ PWM4	I/O	PB5 is bidirectional I/O pin. PB5 can be the output of PWM4.
VDD	P	Positive power supply.
VSS	P	Ground.

## 2. Memory Organization

NY8A054E 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 NY8A054E is 2K words. Therefore, the Program Counter (PC) is 11-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.

NY8A054E provides instruction CALL, GTOA, CALLA to address 256 location of program space. NY8A054E provides instruction GOTO to address 512 location of program space. NY8A054E 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.

NY8A054E program ROM address 0x7FE~0x7FF are reserved space, if user tries to write code in these addresses will get unexpected false functions.

NY8A054E program ROM address 0x00E~0x00F are preset rolling code can be released and used as normal program space.

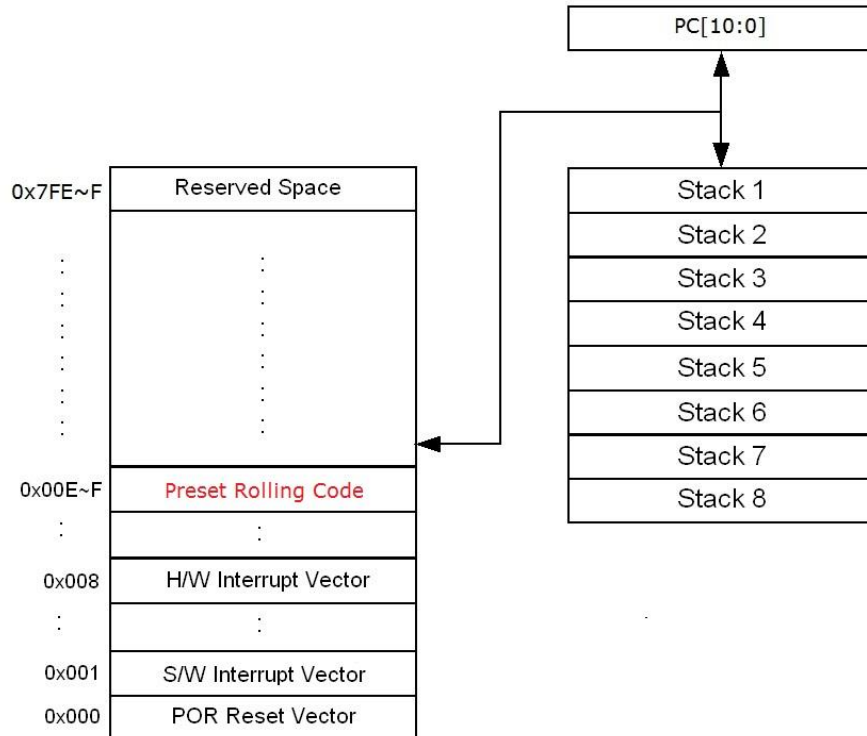


Figure 2 Program Memory Address Mapping

## 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). STATUS [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 STATUS[7:6] and the location selection is from FSR[6:0].

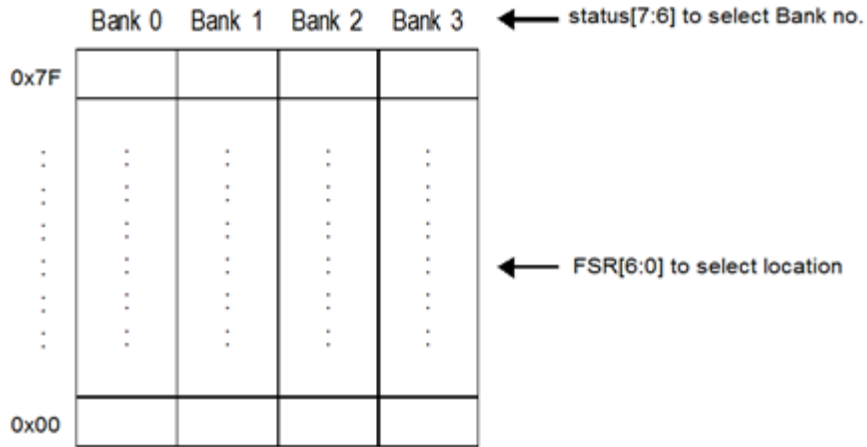


Figure 3 Indirect Addressing Mode of Data Memory Access

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

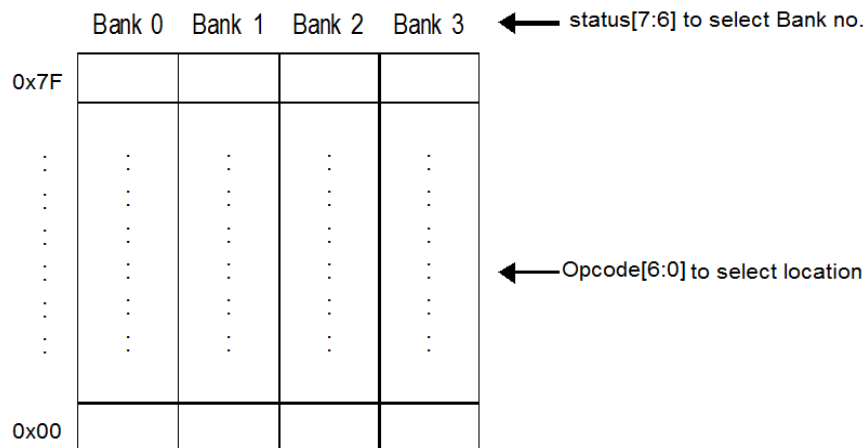


Figure 4 Direct Addressing Mode of Data Memory Access



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 0x1F 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 0x20 to 0x7F of Bank 0 and 0x20 to 0x3F of Bank 1. Other bank in address from 0x20 to 0x7F are mapped back as the Table 1 shows.

The NY8A054E register name and address mapping of R-page SFR are described in the following table.

Status [7:6] Address	00 (Bank 0)	01 (Bank 1)	10 (Bank 2)	11 (Bank 3)
0x0	INDF			
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	-			
0x11	-			
0x12	-			
0x13	-			
0x14	-			
0x15	AWUCON			
0x16	-			
0x17	-			
0x18	INTEDG			
0x19	TMRH			
0x1A	-			

*The same mapping as Bank 0*

Status [7:6] Address	00 (Bank 0)	01 (Bank 1)	10 (Bank 2)	11 (Bank 3)
0x1B	RFC	<i>The same mapping as Bank 0</i>		
0x1C	TM34RH			
0x1D ~ 0x1E	-	-		
0x1F	INTE2	<i>The same mapping as Bank 0</i>		
0x20 ~ 0x3F	General Purpose Register	General Purpose Register	<i>Mapped to bank0</i>	<i>Mapped to Bank1</i>
0x40 ~ 0x7F	General Purpose Register	<i>Mapped to bank0</i>	<i>Mapped to bank0</i>	<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. STATUS[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 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	APHCON	-
0xA	PS0CV	P2CR1
0xB	-	-
0xC	BODCON	PWM2DUTY
0xD	-	-
0xE	CMPCR	-
0xF	PCON1	OSCCR
0x10	-	TMR3
0x11	-	T3CR1
0x12	-	T3CR2
0x13	-	PWM3DUTY
0x14	-	PS3CV
0x15	-	-
0x16	-	P4CR1

Address	SFR Category	F-page SFR	S-page SFR
0x17		-	-
0x18		-	PWM4DUTY
0x19		-	-
0x1A		-	-
0x1B		-	P5CR1
0x1C		-	-
0x1D		-	PWM5DUTY
0x1E		-	-
0x1F		-	PWM5RH

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

### 3. Function Description

This chapter will describe the detailed operations of NY8A054E.

#### 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\_CK10, or from Low Oscillator Frequency according to T0MD and configuration word setting.

##### 3.1.3 PCL (Low Byte of PC[10:0])

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 11-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[10:8], is not directly accessible. Update of PC[10:8] must be done through register PCHBUF.

For GOTO instruction, PC[8:0] is from instruction word and PC[10:9] is loaded from PCHBUF[2:1]. For CALL instruction, PC[7:0] is from instruction word and PC[10:8] is loaded from PCHBUF[2:0]. Moreover the next PC address, i.e. PC+1, will push onto top of Stack. For LGOTO instruction, PC[10:0] is from instruction word.

For LCALL instruction, PC[10:0] 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	BK[1]	BK[0]	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.

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

BK[1:0]: Bank register is 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.

(\*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	GP7	FSR[6:0]						
R/W Property			R/W							
Initial Value			0	X	X	X	X	X	X	X

**FSR[6:0]:** Select one register out of 128 registers of specific Bank.

**GP7:** general register.

### 3.1.6 PortA (PortA Data Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PortA	R	0x5	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
R/W Property			R/W							
Initial Value			Data latch value is xxxxxxxx, read value is xxxxxxxx port value(PA7~PA0)							

While reading PortA, it will get the status of the specific pin if that pin is configured as input pin. However, if that pin is configured as output pin, whether it will get the status of the pin or the value of the corresponding output data latch is depend on the configuration option RD\_OPT. While writing to PortA, data is written to PA's output data latch.

### 3.1.7 PortB (PortB Data Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PortB	R	0x6	GP7	GP6	PB5	PB4	PB3	PB2	PB1	PB0
R/W Property			R/W							
Initial Value			Data latch value is xxxxxxxx, read value is xxxxxxxx port value(PB5~PB0)							

While reading PortB, it will get the status of the specific pin if that pin is configured as input pin. However, if that pin is configured as output pin, whether it will get the status of the pin or the value of the corresponding output data latch is depend on the configuration option RD\_OPT. While writing to PortB, data is written to PB's output data latch.

GP7~6 : general purpose registers.

### 3.1.8 PCON (Power Control Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCON	R	0x8	WDTEN	/PLPA4	LVDEN	/PHPA5	LVREN	CMPEN	GP1	GP0
R/W Property			R/W							
Initial Value			1	1	0	1	1	0	0	0

**GP6, GP4, GP1, GP0:** General read/write register bits.

**LVREN:** Enable/disable LVR.

LVREN=1, enable LVR.

LVREN=0, disable LVR.

**LVDEN:** Enable/disable LVD.

LVDEN=1, enable LVD.

LVDEN=0, disable LVD.

**WDTEN:** Enable/disable WDT.

WDTEN=1, enable WDT.

WDTEN=0, disable WDT.

**COMPEN:** Enable/disable Comparator.

COMPEN=1, enable comparator.

COMPEN=0, disable comparator.

**/PLPA4:** Disable/Enable PA4 pull-low resistor

/PLPA4=1, Disable PA4 pull-low resistor

/PLPA4=0, Enable PA4 pull-low resistor

**/PHPA5:** Disable/Enable PA5 pull-high resistor

/PHPA5=1, Disable PA5 pull-high resistor

/PHPA5=0, Enable PA5 pull-high resistor

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

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

**WUPBx:** Enable/disable PBx wake-up function,  $0 \leq x \leq 5$ .

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	-			PCHBUF[2:0]		
R/W Property			-	W	-					
Initial Value			X	0	X			0		

**PCHBUF[2:0]:** Buffer of the 10<sup>th</sup> ~ 8<sup>th</sup> bit of PC.

**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	-	-	/PHPB5	/PHPB4	/PHPB3	/PHPB2	/PHPB1	/PHPB0
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

**/PHPBx:** Disable/enable PBx Pull-High resistor,  $0 \leq x \leq 5$ .

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

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

**3.1.13 INTE (Interrupt Enable Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTE	R	0xE	INT1IE	WDTIE	-	LVDIE	T1IE	INT0IE	PABIE	T0IE
R/W Property			R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
Initial Value			0	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.

**PABIE:** PortA/PortB input change interrupt enable bit.

PABIE=1, enable PortA/PortB input change interrupt.

PABIE=0, disable PortA/PortB input change interrupt.

**INT0IE:** External interrupt 0 enable bit.

INT0IE=1, enable external interrupt 0.

INT0IE=0, disable external interrupt 0.

**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.

**INT1IE:** External interrupt 1 enable bit.

INT1IE=1, enable external interrupt 1.

INT1IE=0, disable external interrupt 1.



### 3.1.14 INTF (Interrupt Flag Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTF	R	0xF	INT1IF	WDTIF	-	LVDIF	T1IF	INT0IF	PABIF	T0IF
R/W Property			R/W	R/W	-	R/W	R/W	R/W	R/W	R/W
Initial Value(note*)			0	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.

**PABIF:** PortA/PortB input change interrupt flag bit.

PABIF=1, PortA/PortB input change interrupt is occurred.

PABIF must be clear by firmware.

**INT0IF:** External interrupt 0 flag bit.

INT0IF=1, external interrupt 0 is occurred.

INT0IF 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.

**INT1IF:** External interrupt 1 flag bit.

INT1IF=1, external interrupt 1 is occurred.

INT1IF must be clear by firmware.

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

### 3.1.15 AWUCON (PortA Wake-up Control Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
AWUCON	R	0x15	WUPA7	WUPA6	WUPA5	WUPA4	WUPA3	WUPA2	WUPA1	WUPA0
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

**WUPAx:** Enable/disable P<sub>Ax</sub> wake-up function, 0 ≤ x ≤ 7.

WUPAx=1, enable P<sub>Ax</sub> wake-up function.

WUPAx=0, disable P<sub>Ax</sub> wake-up function.

**3.1.16 INTEDG (Interrupt Edge Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTEDG	R	0x18	-	-	EIS1	EIS0	INT1G1	INT1G0	INT0G1	INT0G0
R/W Property			-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value			X	X	0	0	0	1	0	1

**EIS1:** External interrupt 1 select bit

EIS1=1, PB1/PA2 is external interrupt 1.

EIS1=0, PB1/PA2 is GPIO.

**EIS0:** External interrupt 0 select bit

EIS0=1, PB0 is external interrupt 0.

EIS0=0, PB0 is GPIO.

**INT1G1~0:** INT1 edge trigger select bit.

00: reserved, 01: rising edge, 10: falling edge, 11: rising/falling edge.

**INT0G1~0:** INT0 edge trigger select bit.

00: reserved, 01: rising edge, 10: falling edge, 11: rising/falling edge.

**3.1.17 TMRH (Timer High Byte Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TMRH	R	0x19	-	-	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.

**3.1.18 RFC (RFC Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
RFC	R	0x1B	RFCEN	-	-	-	PSEL[3:0]			
R/W Property			R/W	-	-	-	R/W			
Initial Value			0	X	X	X	0			

**RFCEN:** Enable/disable RFC function.

RFCEN=1, enable RFC function.

RFCEN=0, disable RFC function.

**PSEL[3:0]:** Select RFC pad.

PSEL[3:0]	RFC PAD
0000	PA0
0001	PA1
0010	PA2
0011	PA3
0100	PA4
0101	PA5
0110	PA6
0111	PA7
1000	PB0
1001	PB1
1010	PB2
1011	PB3
1100	PB4
1101	PB5

Table 3 RFC pad select

### 3.1.19 TM34RH (Timer3 High Byte Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TM34RH	R	0x1C	-	-	TMR39	TMR38	PWM4 DUTY9	PWM4 DUTY8	PWM3 DUTY9	PWM3 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

**TMR39~8:** Timer3 MSB 2 bits. Write these 2 bits will overwrite the 10-bit Timer3 load value of bit 9 and 8. Read these 2 bits will get the Timer3 bit9-8 current value.

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

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

### 3.1.20 INTE2 (Interrupt Enable and Flag 2nd. Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
INTE2	R	0x1F	-	-	-	T3IF	-	-	-	T3IE
R/W Property			-	-	-	R/W	-	-	-	R/W
Initial Value			-	-	-	0	-	-	-	0

**T3IF:** Timer3 underflow interrupt flag bit.

T3IF=1, Timer3 underflow interrupt is occurred.

T3IF must be clear by firmware.

**T3IE:** Timer3 underflow interrupt enable bit.

T3IE=1, enable Timer3 underflow interrupt.

T3IE=0, disable Timer3 underflow interrupt.

### 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	GP6	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 (Timer0)	PS0WDT=1 (WDT Reset)	PS0WDT=1 (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 4 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\_CK10.

T0CE=0, Timer0 will increase one while low-to-high transition occurs on pin EX\_CK10.

**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\_CK10 or Low Oscillator Frequency (I\_LRC or E\_LXT) is selected.

T0CS=0, Instruction clock  $F_{INST}$  is selected.

**GP6:** General register.

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

T0CS=0, Instruction clock  $F_{INST}$  is selected as Timer0 clock source.

T0CS=1, LCKTM0=0, external clock on pin EX\_CK10 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\_CK10 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	IOPA7	IOPA6	IOPA5	IOPA4	IOPA3	IOPA2	IOPA1	IOPA0
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

**IOPAx:** P<sub>Ax</sub> I/O mode selection,  $0 \leq x \leq 7$ .

IOPAx=1, P<sub>Ax</sub> is input mode.

IOPAx=0, P<sub>Ax</sub> 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	-	-	IOPB5	IOPB4	IOPB3	IOPB2	IOPB1	IOPB0
R/W Property			-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value			X	X	X	X	1	1	1	1

**IOPBx:** P<sub>Bx</sub> I/O mode selection,  $0 \leq x \leq 5$ .

IOPBx=1, P<sub>Bx</sub> is input mode.

IOPBx=0, P<sub>Bx</sub> is output mode.

#### 3.3.3 APHCON (PortA Pull-High Resistor Control Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
APHCON	F	0x9	/PHPA7	/PHPA6	/PLPA5	/PHPA4	/PHPA3	/PHPA2	/PHPA1	/PHPA0
R/W Property			R/W							
Initial Value			1	1	1	1	1	1	1	1

**/PHPAx:** Enable/disable Pull-High resistor of P<sub>Ax</sub>, x=0~4, 6~7.

/PHPAx=1, disable Pull-High resistor of P<sub>Ax</sub>.

/PHPAx=0, enable Pull-High resistor of P<sub>Ax</sub>.

**/PLPA5:** Enable/disable Pull-Low resistor of PA5.

/PLPA5=1, disable Pull-Low resistor of PA5.

/PLPA5=0, enable Pull-Low resistor of PA5.

**Note:** When PA6 and PA7 are used as crystal oscillator pads, the Pull-High resistor should not enable. Or the oscillation may fail.

### 3.3.4 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.

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

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
BODCON	F	0xC	-	-	ODPB5	ODPB4	ODPB3	ODPB2	ODPB1	ODPB0
R/W Property			-	-	R/W	R/W	R/W	R/W	R/W	R/W
Initial Value			X	X	X	X	0	0	0	0

**ODPBx:** Enable/disable open-drain of PBx,  $0 \leq x \leq 5$ .

ODPBx=1, enable open-drain of PBx.

ODPBx=0, disable open-drain of PBx.

### 3.3.6 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]	Non-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 3.15.1)*

### 3.3.7 PCON1 (Power Control Register1)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PCON1	F	0xF	GIE	LVDOUT	LVDS3	LVDS2	LVDS1	LVDS0	GP1	T0EN
R/W Property			R/W(1*)	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 16 LVD voltage.

LVDS[3:0]	Voltage
0000	1.9V
0001	2.0V
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 7 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.

**GP5, 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 10-bit down-count Timer1 at TMR1[9:0]. When writing register TMR1, it will write data from TMRH[5:4] and Timer1 reload register to TMR1[9:0] 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			R/W	R/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[9:0].

T1RL=0, continuous down-count from 0x3FF 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. When underflow is reached, 0x3FF is reloaded and continues down-count.



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

Table 8 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 PB3 or PA4.

PWM1OEN=0, PB3 or PA4 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 9 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\_CK10.

T1CE=0, Timer1 will decrease one while low-to-high transition occurs on pin EX\_CK10.

**T1CS:** Timer1 clock source selection.

T1CS=1, External clock on pin EX\_CKIO 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							

The reload value of 10-bit Timer1 stored on registers TMRH[5:4] and TMR1[7:0] is used to define the PWM1 frame rate, and registers TMRH[1:0] and PWM1DUTY[7:0] 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

BZ1FSEL[3:0]	BZ1 Frequency Selection	
	Clock Source	Dividing Rate
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 10 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 3.18) 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 11 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[10:0] and TBHP[2:0] is the high byte of PC[10:0].

### 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 PB2 or PA0.  
 PWM2OEN=0, PB2 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 OSCCR (Oscillation Control Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
OSCCR	S	0xF	-	-	-	-	OPMD[1:0]	STPHOSC	SELHOSC	
R/W Property			-	-	-	-	R/W	R/W	R/W	
Initial Value			X	X	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
10	Standby mode
11	reserved

Table 12 Selection of Operating Mode by OPMD[1:0]

**3.4.13 TMR3 (Timer3 Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
TMR3	S	0x10	TMR3[7:0]							
R/W Property			R/W							
Initial Value			XXXXXXXX							

When reading register TMR3, it will obtain current value of 10-bit down-count Timer3 at TMR3[7:0]. When writing register TMR3, it will write data from TMR3RH[5:4] and Timer3 reload register to Timer3[9:0] current content.

### 3.4.14 T3CR1 (Timer3 Control Register1)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T3CR1	S	0x11	PWM3OEN	PWM3OAL	-	-	-	T3OS	T3RL	T3EN
R/W Property			R/W	R/W	-	-	-	R/W	R/W	R/W
Initial Value			0	0	X	X	X	0	0	0

This register is used to configure Timer3 functionality.

**T3EN:** Enable/disable Timer3.

T3EN=1, enable Timer3.

T3EN=0, disable Timer3.

**T3RL:** Configure Timer3 down-count mechanism while Non-Stop mode is selected (T3OS=0).

T3RL=1, initial value is reloaded from reload register TMR3.

T3RL=0, continuous down-count from 0x3FF when underflow is occurred.

**T3OS:** Configure Timer3 operating mode while underflow is reached.

T3OS=1, One-Shot mode. Timer3 will count once from the initial value to 0x00.

T3OS=0, Non-Stop mode. Timer3 will keep down-count after underflow.

T3OS	T3RL	Timer3 Down-Count Functionality
0	0	Timer3 will count from reload value down to 0x00. When underflow is reached, 0x3FF is reloaded and continues down-count.
0	1	Timer3 will count from reload value down to 0x00. When underflow is reached, reload value is reloaded and continues to down-count.
1	x	Timer3 will count from initial value down to 0x00. When underflow is reached, Timer3 will stop down-count.

Table 13 Timer3 Functionality

**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 PB4.

PWM3OEN=0, PA3 or PB4 is GPIO.

### 3.4.15 T3CR2 (Timer3 Control Register2)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
T3CR2	S	0x12	-	-	T3CS	T3CE	/PS3EN	PS3SEL[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 Timer3 functionality.

**PS3SEL[2:0]:** Prescaler3 dividing rate selection.

PS3SEL[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 14 Prescaler3 Dividing Rate

**Note:** Always set PS3SEL[2:0] at /PS3EN=1, or interrupt may be falsely triggered.

**/PS3EN:** Disable/enable Prescaler3.

/PS3EN=1, disable Prescaler3.

/PS3EN=0, enable Prescaler3.

**T3CE:** Timer3 external clock edge selection.

T3CE=1, Timer3 will decrease one while high-to-low transition occurs on pin EX\_CK11.

T3CE=0, Timer3 will decrease one while low-to-high transition occurs on pin EX\_CK11.

**T3CS:** Timer3 clock source selection.

T3CS=1, External clock on pin EX\_CK11 is selected.

T3CS=0, Instruction clock is selected.

### 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 Timer3 stored on registers TM34RH[5:4] and TMR3[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 PS3CV (Prescaler3 Counter Value Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PS3CV	S	0x14	PS3CV[7:0]							
R/W Property			R							
Initial Value			1	1	1	1	1	1	1	1

While reading PS3CV, it will get current value of Prescaler3 counter.

**3.4.18 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 PB5.

PWM4OEN=0, PA2 or PB5 is GPIO.

**3.4.19 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 Timer3 stored on registers TM34RH[5:4] and TMR3[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.

**3.4.20 P5CR1 (PWM5 Control Register1)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
P5CR1	S	0x1B	PWM5OEN	PWM5OAL	-	-	-	-	-	-
R/W Property			R/W	R/W	-	-	-	-	-	-
Initial Value			0	0	X	X	X	X	X	X

**PWM5OAL:** Define PWM5 output active state.

PWM5OAL=1, PWM5 output is active low.

PWM5OAL=0, PWM5 output is active high.

**PWM5OEN:** Enable/disable PWM5 output.

PWM5OEN=1, PWM5 output will be present on PB0 or PA1.

PWM5OEN=0, PB0 or PA1 is GPIO.

**3.4.21 PWM5DUTY (PWM5 Duty Register)**

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM5DUTY	S	0x1D	PWM5DUTY[7:0]							



R/W Property	W
Initial Value	XXXXXXXX

The reload value of 10-bit Timer3 stored on registers TM34RH[5:4] and TMR3[7:0] is used to define the PWM5 frame rate, and registers PWM5RH[1:0] and PWM5DUTY[7:0] is used to define the duty cycle of PWM5.

### 3.4.22 PWM5RH (PWM5 High Byte Register)

Name	SFR Type	Addr.	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
PWM5RH	S	0x1F	-	-	-	-	-	-	PWM5 DUTY9	PWM5D UTY8
R/W Property			-	-	-	-	-	-	R/W	R/W
Initial Value			X	X	X	X	X	X	X	X

**PWM5DUTY9~8:** PWM5 duty data MSB 2 bits.

## 3.5 I/O Port

NY8A054E provides 14 I/O pins which are PA[7:0] and PB[5:0]. 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[7:0] define the input/output direction of PA[7:0]. Register IOSTB[5:0] define the input/output direction of PB[5:0].

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 APHCON[7:6, 4:0] and PCON[4] are used to enable or disable Pull-High resistor of PA[7:0]. Register APHCON[5], PCON[6] and ABPLCON[3:0] are used to enable or disable Pull-Low resistor of PA[5:0]. Register BPHCON[5:0] are used to enable or disable Pull-High resistor of PB[5:0]. Register ABPLCON[7:4] are used to enable or disable Pull-Low resistor of PB[3:0].

When an PortB I/O pin is configured as output pin, there is a corresponding and individual register to select as Open-Drain output pin. Register BODCON[5:0] determine PB[5:0] is Open-Drain or not.

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

Feature		PA[4:0]	PA[7:6]	PA[5]	PB[3:0]	PB[5:4]
Input	Pull-High Resistor	V	V	V	V	V
	Pull-Low Resistor	V	X	V	V	X
Output	Open-Drain	X	X	always	V	V

Table 15 Summary of Pad I/O Feature

The level change on each I/O pin of PA and PB may generate interrupt request. Register AWUCON[7:0] and BWUCON[5:0] will select which I/O pin of PA and PB may generate this interrupt. As long as any pin of PA and

PB is selected by corresponding bit of AWUCON and BWUCON, the register bit PABIF (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 PABIE (INTE[1]) and GIE (PCON1[7]) are both set to 1.

There is two external interrupt provided by NY8A054E. When register bit EIS0 (INTEDG[4]) is set to 1, PB0 is used as input pin for external interrupt 0. When register bit EIS1 (INTEDG[5]) is set to 1, PB1 is used as input pin for external interrupt 1.

***Note: When PB0 or PB1 is both set as level change operation and external interrupt, the external interrupt will have higher priority, and the PB0 or PB1 level change operation will be disabled. But PB5~PB2 level change function are not affected.***

NY8A054E provides IR carrier generation output. It is depended both on register bit IREN (IRCR[0]) and carrier output will be present on PB1 pad. When IREN=1, the IR sink current=20mA. When IREN=0, the IR carrier will not be generated.

PA5 can be used as external reset input determined by a configuration word. When an active-low signal is applied to PA5, it will cause NY8A054E 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, PA6 will be used as crystal input pin (Xin) and PA7 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 PA7 if a configuration word is enabled.

Moreover, PA4 can be timer 0 external clock source EX\_CK10 if T0MD T0CS=1 and LCK\_TM0=0. PA4 can be timer 1 external clock source EX\_CK10 if T1CS=1. PA1 can be Timer3 external clock source EX\_CK11 if T3CS=1.

Moreover, PB3 or PA4 can be PWM1 output If T1CR1[7] PWM1OEN=1. PB3 can be Buzzer1 output if BZ1CR[7] BZ1EN=1. The output priority of PB3 is PWM1 output > Buzzer1 output.

PB2 or PA0 can be PWM2 output If P2CR1[7] PWM2OEN=1.

### 3.5.1 Block Diagram of IO Pins

IO\_SEL: set pad attribute as input or output

WRITE\_EN: write data to pad.

READ\_EN: read pad.

PULLUP100K\_ENB: enable Pull-High.

PULLDOWN\_EN: enable Pull-Low.

VPEN: enable pad to comparator non-inverting input.

VNEN: enable pad to comparator inverting input.

CMPVP, CMPVN: comparator non-inverting and inverting input.

RD\_TYPE: select read pin or read latch.

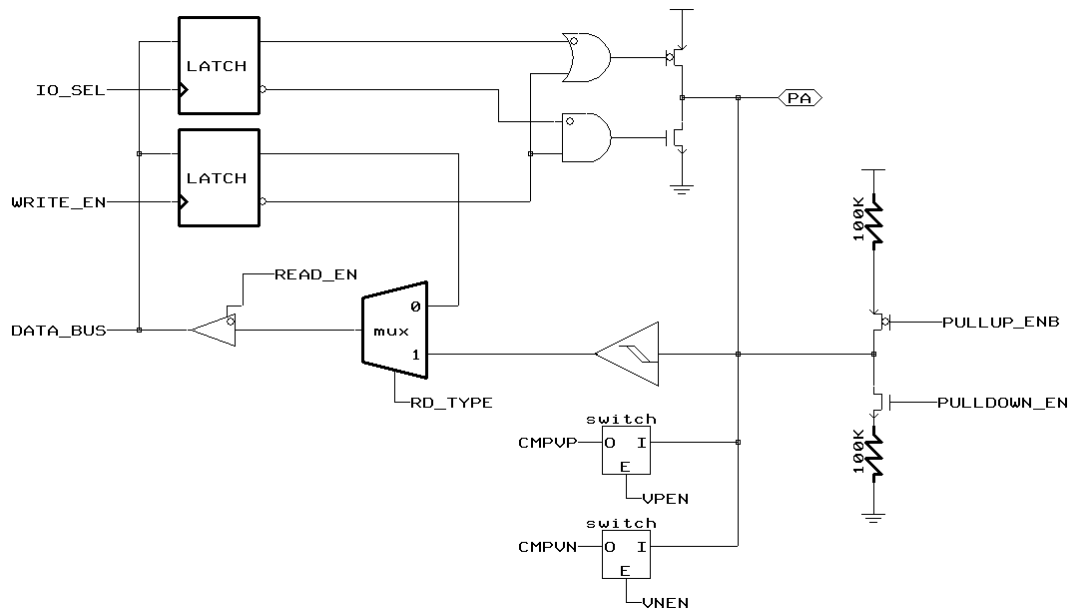


Figure 5 Block Diagram of PA[3:2]&PA0

IO\_SEL: set pad attribute as input or output.

WRITE\_EN: write data to pad.

READ\_EN: read pad.

PULLUP100K\_ENB: enable Pull-High.

PULLDOWN\_EN: enable Pull-Low.

VPEN: enable pad to comparator non-inverting input.

VNEN: enable pad to comparator inverting input.

CMPVP, CMPVN: comparator non-inverting and inverting input.

RD\_TYPE: select read pin or read latch.

EX\_CK11: external clock for Timer 3.

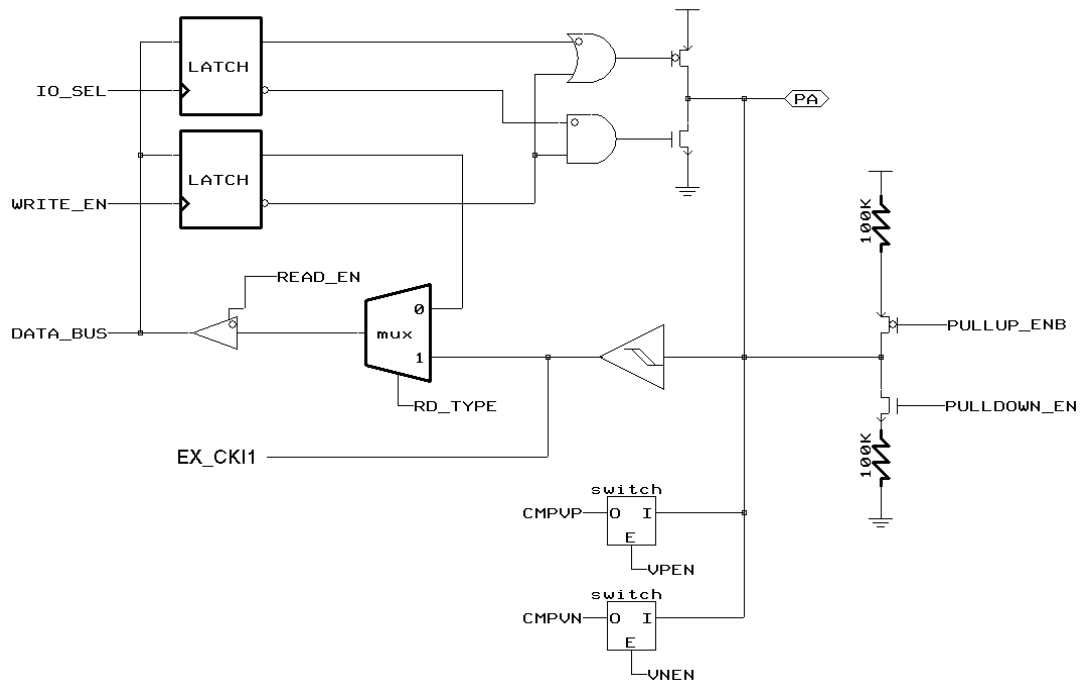


Figure 6 Block Diagram of PA1

IO\_SEL: set pad attribute as input or output.

WRITE\_EN: write data to pad.

READ\_EN: read pad.

PULLUP100K\_ENB: enable Pull-High.

PULLDOWN\_EN: enable Pull-Low.

RD\_TYPE: select read pin or read latch.

EX\_CKIO: external clock for Timer0, 1.

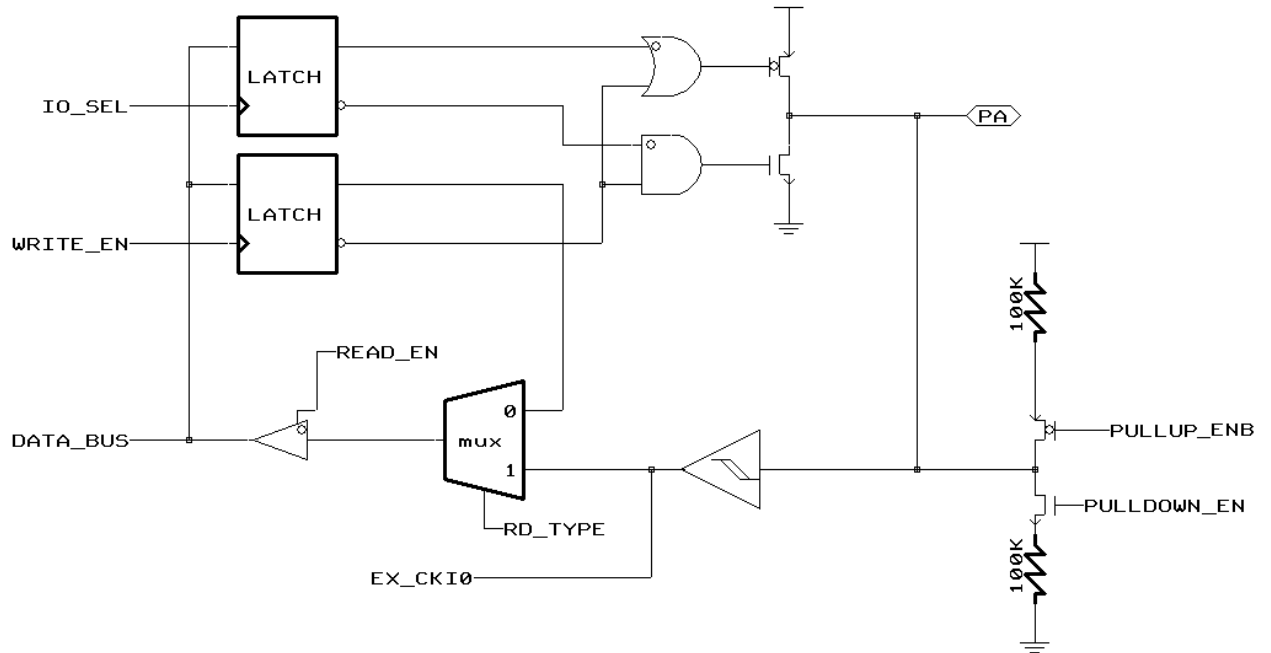


Figure 7 Block Diagram of PA4

- RSTPAD\_EN: enable PA5 as reset pin.
- RSTB\_IN: reset signal input.
- IO\_SEL: set pad attribute as input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad.
- PULLUP100K\_ENB: enable Pull-High.
- PULLDOWN\_EN: enable Pull-Low.
- RD\_TYPE: select read pin or read latch.

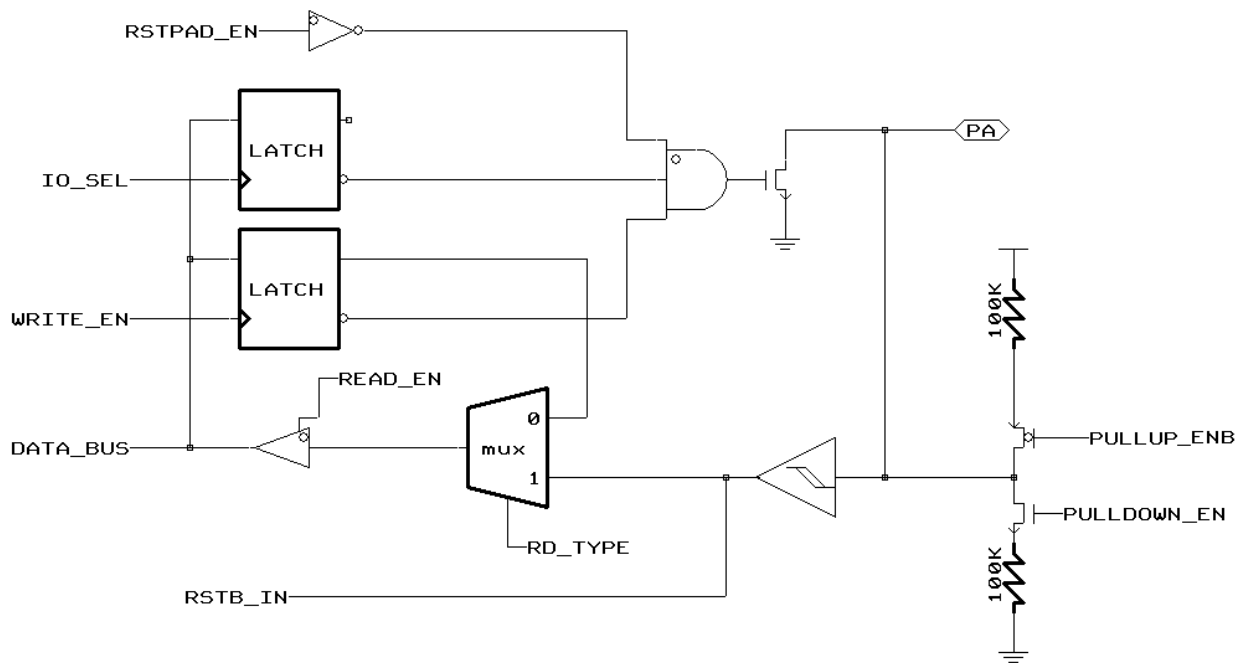


Figure 8 Block Diagram of PA5

- XTL\_EN: enable crystal oscillation mode.
- IO\_SEL: set pad attribute as input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad.
- PULLUP100K\_ENB: enable Pull-High.
- RD\_TYPE: select read pin or read latch.

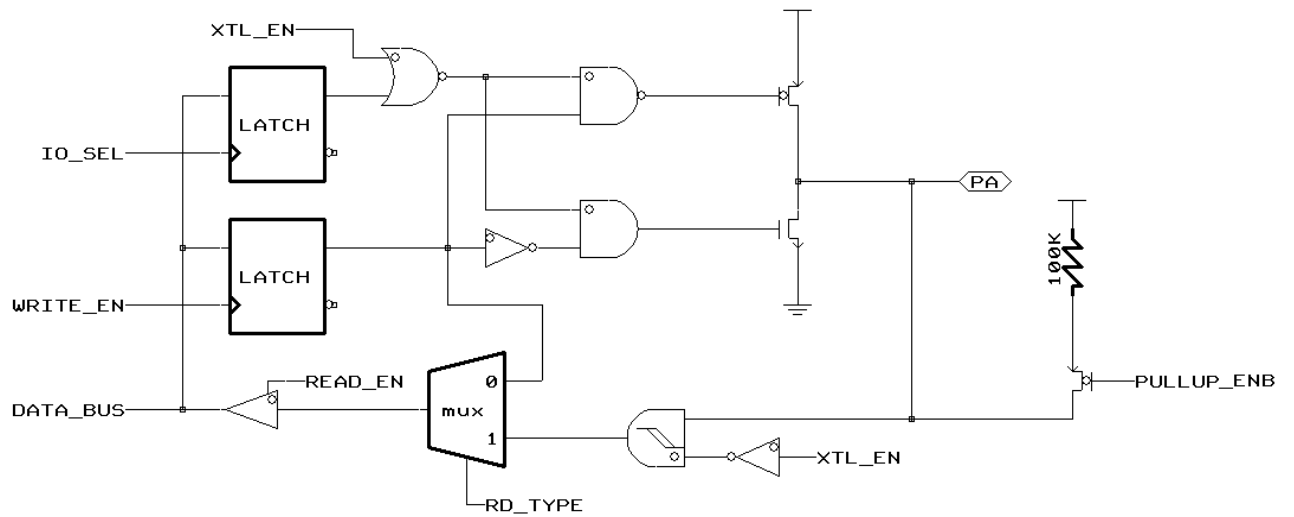


Figure 9 Block Diagram of PA6, PA7





- IO\_SEL: set pad attribute as input or output.
- WRITE\_EN: write data to pad.
- READ\_EN: read pad.
- OD\_EN: enable open-Drain.
- PULLUP100K\_ENB: enable Pull-High.
- PULLDOWN\_EN: enable Pull-Low.
- RD\_TYPE: select read pin or read latch.
- EIS1: external interrupt function enable.
- INTEDG[3:2]: external interrupt edge select.
- EX\_INT1: external interrupt signal.
- WUB: port B wake-up enable.
- SET\_PBIF: port B wake-up flag.

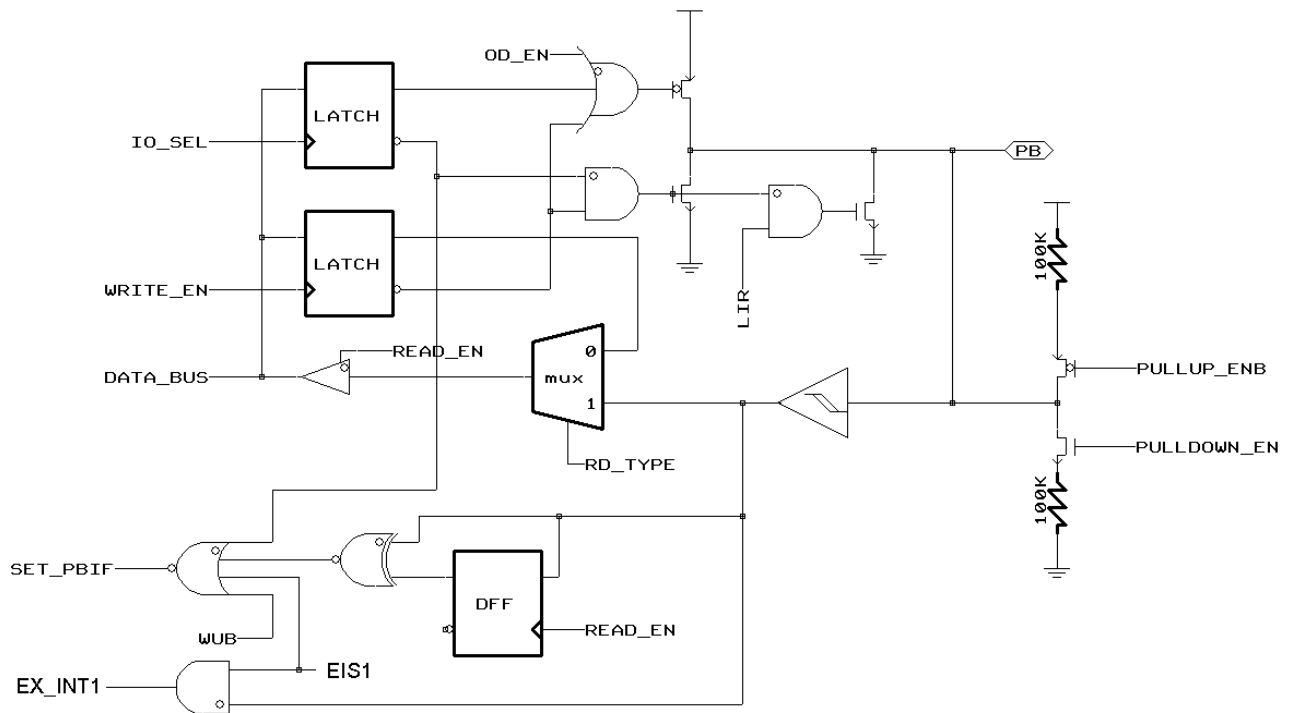


Figure 11 Block Diagram of PB1

IO\_SEL: set pad attribute as input or output.

WRITE\_EN: write data to pad.

READ\_EN: read pad.

OD\_EN: enable open-Drain.

PULLUP100K\_ENB: enable Pull-High.

PULLDOWN\_EN: enable Pull-Low.

RD\_TYPE: select read pin or read latch.

WUB: port B wake-up enable.

SET\_PBIF: port B wake-up flag.

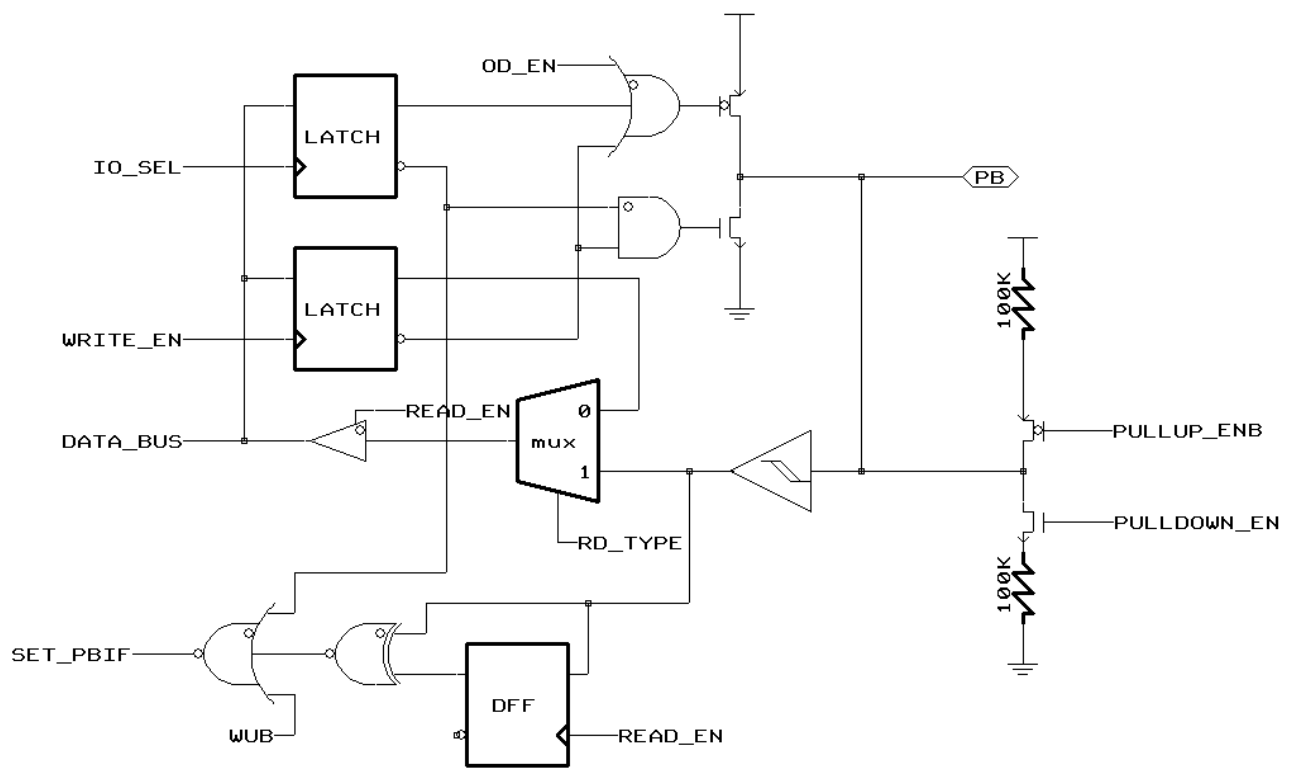


Figure 12 Block Diagram of PB2

IO\_SEL: set pad attribute as input or output.

WRITE\_EN: write data to pad.

READ\_EN: read pad.

OD\_EN: enable open-Drain.

PULLUP100K\_ENB: enable Pull-High.

PULLDOWN\_EN: enable Pull-Low.

RD\_TYPE: select read pin or read latch.

WUB: port B wake-up enable.

SET\_PBIF: port B wake-up flag.

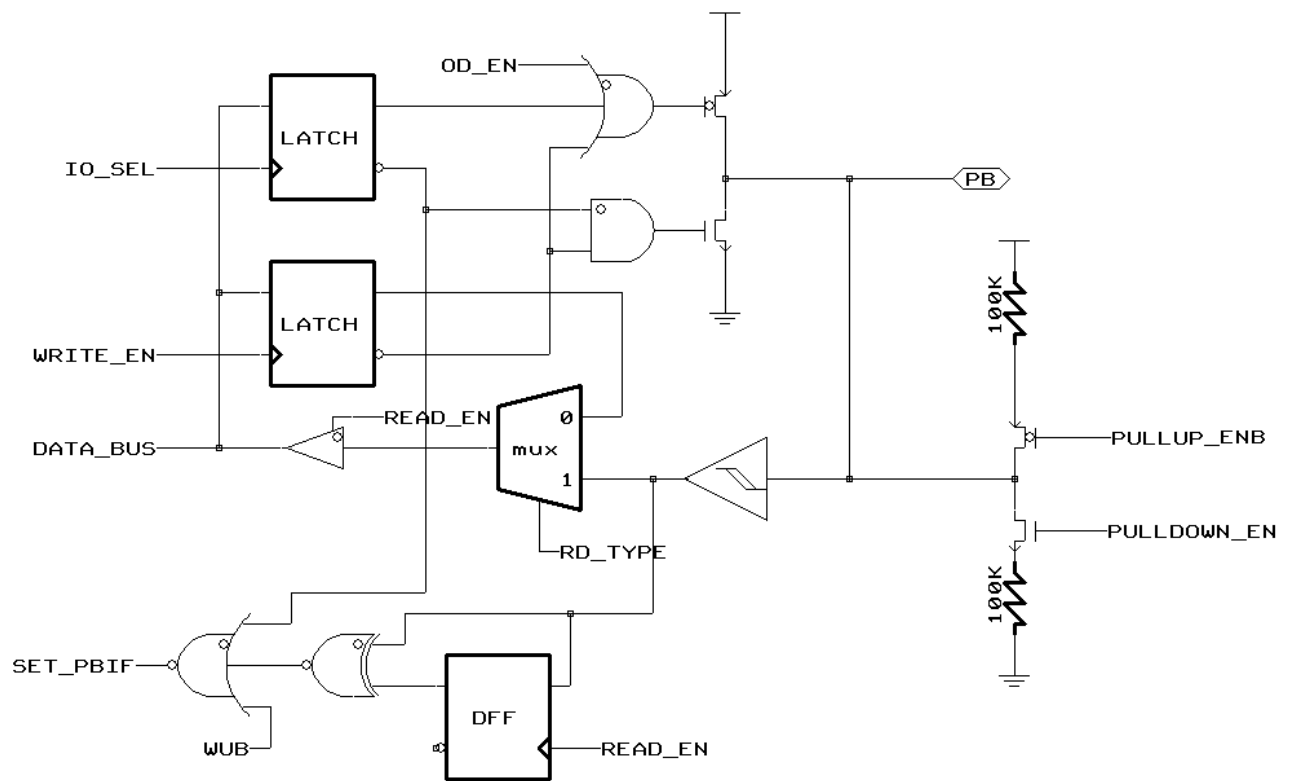


Figure 13 Block Diagram of PB3

IO\_SEL: set pad attribute as input or output.

WRITE\_EN: write data to pad.

READ\_EN: read pad.

OD\_EN: enable open-Drain.

PULLUP100K\_ENB: enable Pull-High.

RD\_TYPE: select read pin or read latch.

WUB: port B wake-up enable.

SET\_PBIF: port B wake-up flag.

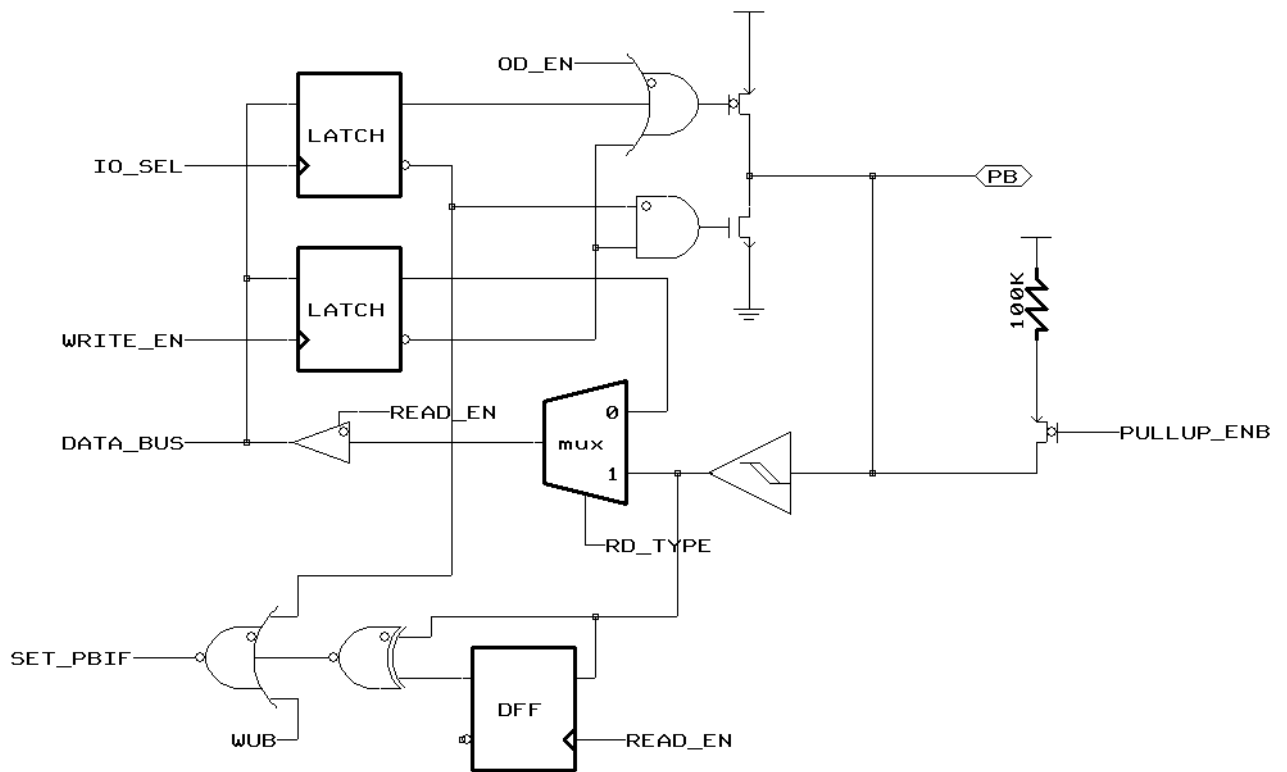


Figure 14 Block Diagram of PB4, PB5

### 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\_CK10 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\_CK10 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 Table 15)

Timer0 clock source	T0CS	LCKTM0	Timer0 source	Low Oscillator Frequency
Instruction clock	0	X	X	X
EX_CK10	1	0	X	X
		X	0	
E_LXT	1	1	1	1
I_LRC	1	1	1	0

Table 16 Summary of Timer0 clock source control

Moreover the active edge of EX\_CK10 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\_CK10 or Low Oscillator Frequency will increase Timer0. When T0CE is 0, low-to-high transition on EX\_CK10 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.

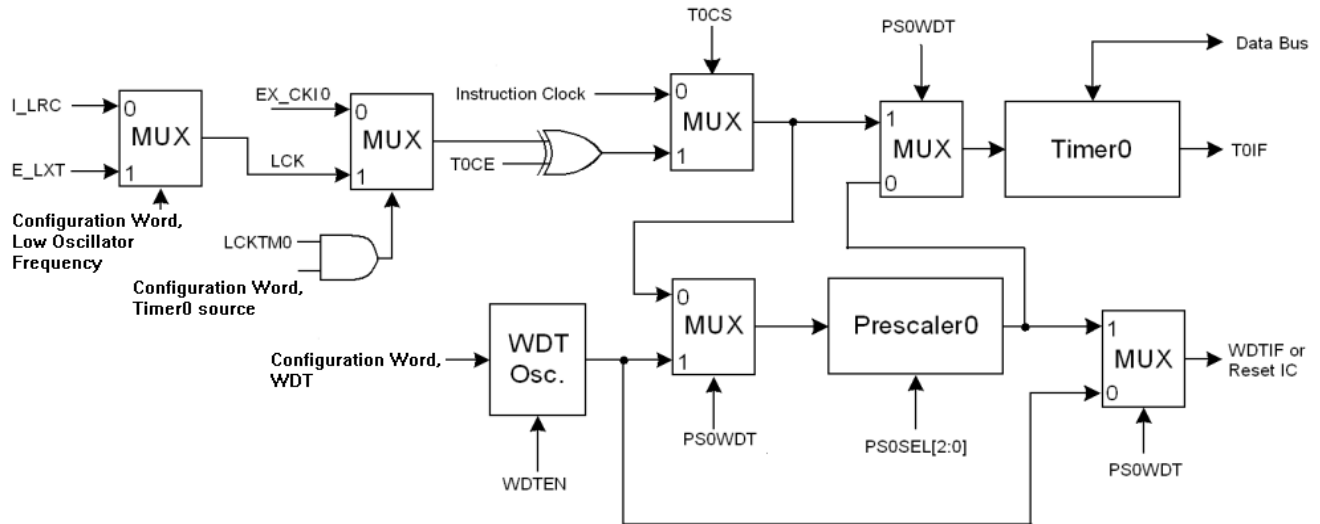


Figure 15 Block Diagram of Timer0 and WDT

### 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. Timer1 builds in auto-reload function and Timer1 reload register stores reload data with double buffers. When user write Timer1 reload register, write Timer1 MSB 2 bits(TMRH[5:4]) first and write TMR1 second, Timer1 reload register will be updated to Timer1 counter after Timer1 overflow occurs when T1EN=1. If T1EN=0, Timer1 reload register will be updated to Timer1 counter after write TMR1 immediately. 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.

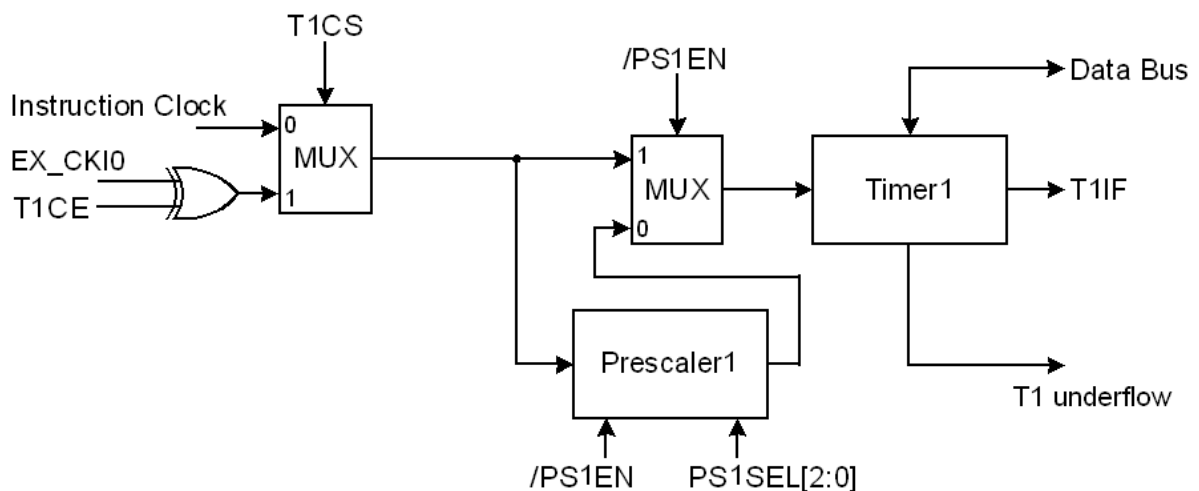


Figure 16 Block Diagram of Timer1

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\_CKIO which is determined by register bit T1CS (T1CR2[5]). When T1CS is 1, EX\_CKIO is selected as clock source. When T1CS is 0, instruction clock is selected as clock source. When EX\_CKIO 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\_CKIO will decrease Timer1. When T1CE is 0, low-to-high transition on EX\_CKIO 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 provides 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[9:0] 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[9:0] will be restored and start next down-count from this initial value. When T1RL is 0, Timer1 will start next down-count from 0x3FF.

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.

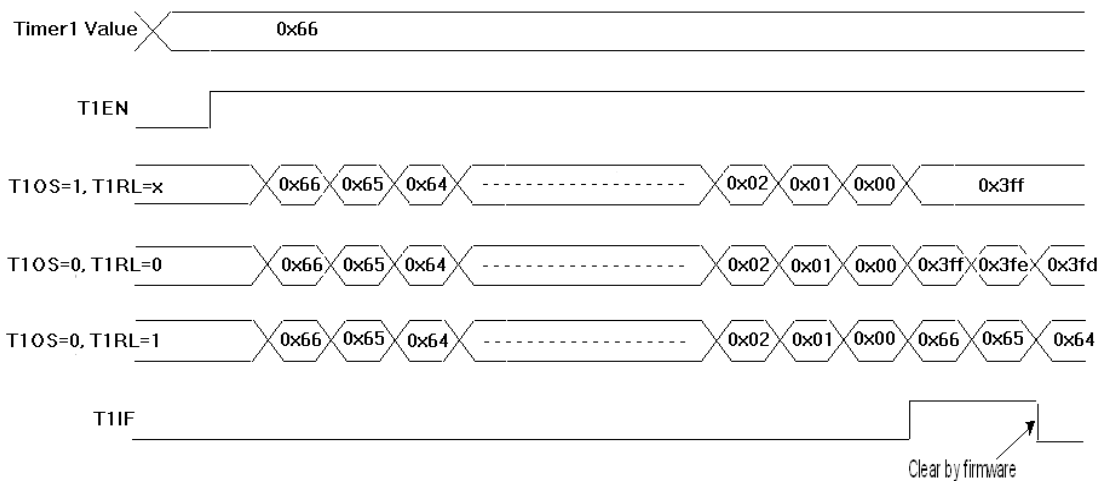


Figure 17 Timer1 Timing Chart

The PWM1 output can be available on I/O pin PB3 or PA4 when register bit PWM1OEN (T1CR1[7]) is set to 1. Moreover, PB3 or PA4 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.

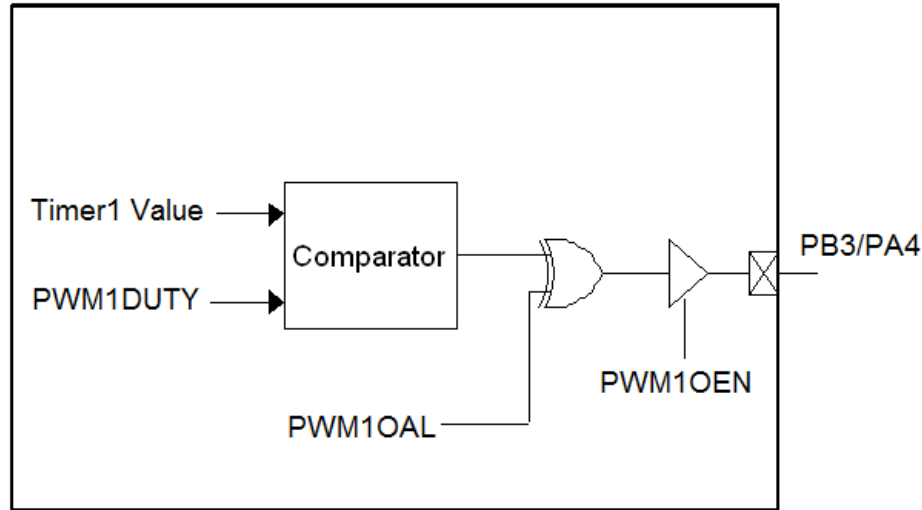


Figure 18 PWM1 Block Diagram

The Buzzer1 output (BZ1) can be available on I/O pin PB3 when register bit BZ1EN (BZ1CR[7]) is set to 1. Moreover, PB3 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.

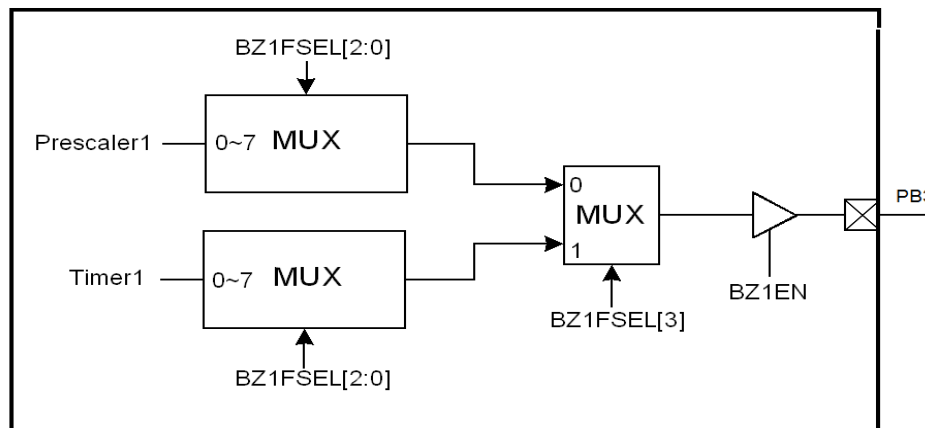


Figure 19 Buzzer1 Block Diagram

**Note: When PWM1 and Buzzer1 are both enabled, PWM1 will have the higher priority for PB3 output.**



### 3.8 PWM2

The PWM2 output can be available on I/O pin PB2 or PA0 when register bit PWM2OEN (P2CR1[7]) is set to 1. Moreover, PB2 or PA0 will become output pin automatically. The active state of PWM2 output is determined by register bit PWM2OAL (P2CR1[6]). When PWM2OAL is 1, PWM2 output is active low. When PWM2OAL is 0, PWM2 output is active high. Moreover, the duty cycle and frame rate of PWM2 are both programmable. The duty cycle is determined by register TMRH[3:2], PWM2DUTY[7:0]. When PWM2DUTY is 0, PWM2 output will be never active. When PWM2DUTY is 0x3FF, PWM2 output will be active for 1023 Timer1 input clocks. The frame rate is determined by TMRH[5:4], TMR1[7:0] initial value. Therefore, PWM2DUTY value must be less than or equal to TMR1[9:0]. Besides, user needs to set P2CR1 (T1EN, T1RL, T1OS data is as the same as T1CR1) because PWM2 and PWM1 are both shared with Timer1. When user write PWM2DUTY, write PWM2DUTY[9:8] MSB 2 bits(TMRH[3:2]) first and write PWM2DUTY[7:0] second, PWM2 duty register will be updated after Timer1 overflow occurs. The block diagram of PWM2 is illustrated in the following figure.

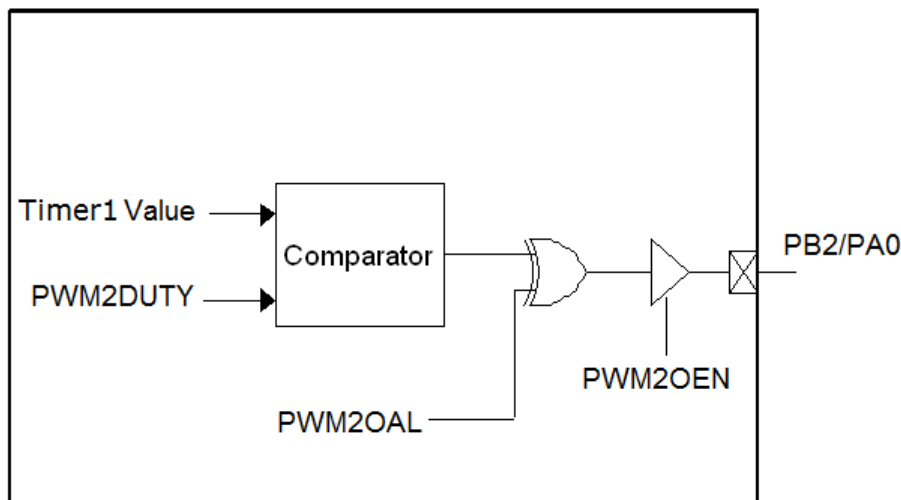


Figure 20 PWM2 Block Diagram

### 3.9 Timer3 / PWM3

Timer3 is an 10-bit down-count timer with Prescaler3 whose dividing rate is programmable. The output of Timer3 can be used to generate PWM3 output. Timer3 builds in auto-reload function and Timer3 reload register stores reload data with double buffers. When users write Timer3 reload register, write Timer3 MSB 2 bits(TM34RH[5:4]) first and write TMR3 second, Timer3 reload register will be updated to Timer3 counter after Timer3 overflow occurs when T3EN=1. If T3EN=0, Timer3 reload register will be updated to Timer3 counter after write TMR3 immediately. A read to the Timer3 will show the content of the Timer3 current count value.

The block diagram of Timer3 is shown in the figure below.

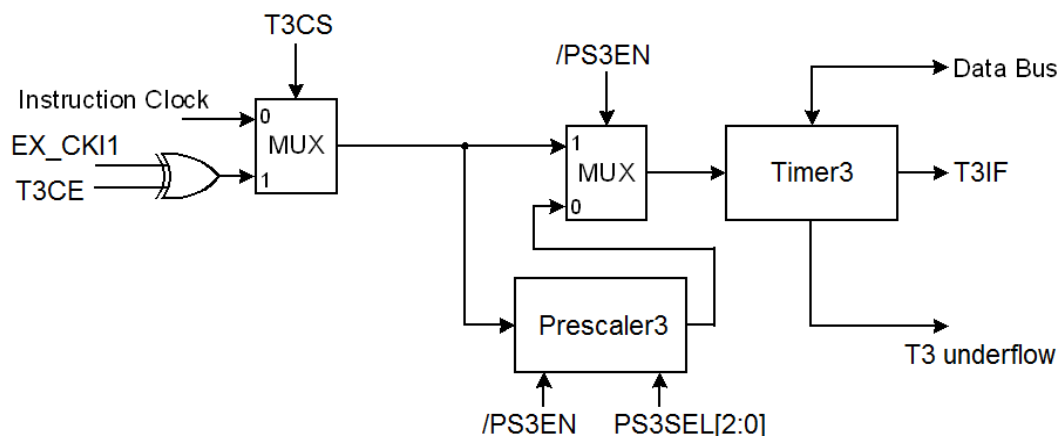


Figure 21 Block Diagram of Timer3

The operation of Timer3 can be enabled or disabled by register bit T3EN (T3CR1[0]). After Timer3 is enabled, its clock source can be instruction clock or pin EX\_CK11 which is determined by register bit T3CS (T3CR2[5]). When T3CS is 1, EX\_CK11 is selected as clock source. When T3CS is 0, instruction clock is selected as clock source. When EX\_CK11 is selected, the active edge to decrease Timer3 is determined by register bit T3CE (T3CR2[4]). When T3CE is 1, high-to-low transition on EX\_CK11 will decrease Timer3. When T3CE is 0, low-to-high transition on EX\_CK11 will decrease Timer3.

The selected clock source can be divided further by Prescaler3 before it is applied to Timer3. Prescaler3 is enabled by writing 0 to register bit /PS3EN (T3CR2[3]) and the dividing rate is from 1:2 to 1:256 determined by register bits PS3SEL[2:0] (T3CR2[2:0]). Current value of Prescaler3 can be obtained by reading register PS3CV.

Timer3 provides two kinds of operating mode: one is One-Shot mode and the other is Non-Stop mode. When register bit T3OS (T3CR1[2]) is 1, One-Shot mode is selected. Timer3 will count down once from initial value stored on register TMR3[9:0] to 0x00, i.e. underflow is occurred. When register bit T3OS (T3CR1[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 T3RL (T3CR1[1]). When T3RL is 1, the initial value stored on register TMR3[9:0] will be restored and start next down-count from this initial value. When T3RL is 0, Timer3 will start next down-count from 0x3FF.

When Timer3 is underflow, the register bit T3IF (INTE2[4]) will be set to 1 to indicate Timer3 underflow event is occurred. If register bit T3IE (INTE2[0]) and GIE are both set to 1, interrupt request will occur and interrupt service routine will be executed. T3IF will not be clear until firmware writes 0 to T3IF.

The timing chart of Timer3 is shown in the following figure.

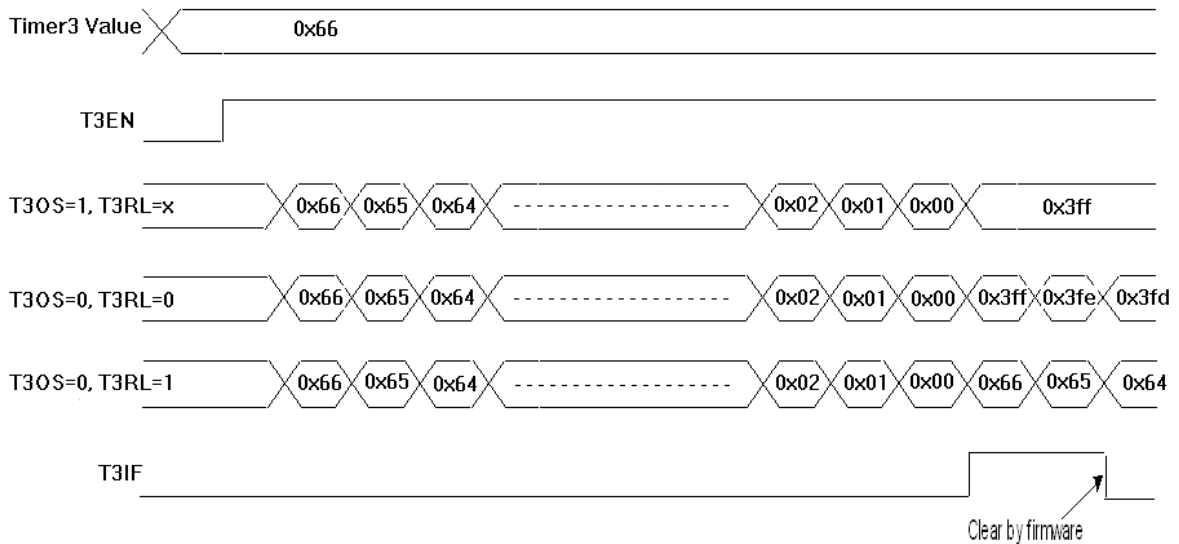


Figure 22 Timer3 Timing Chart

The PWM3 output can be available on I/O pin PA3 or PB4 when register bit PWM3OEN (T3CR1[7]) is set to 1. Moreover, PA3 or PB4 will become output pin automatically. The active state of PWM3 output is determined by register bit PWM3OAL (T3CR1[6]). When PWM3OAL is 1, PWM3 output is active low. When PWM3OAL is 0, PWM3 output is active high. Moreover, the duty cycle and frame rate of PWM3 are both programmable. The duty cycle is determined by register TM34RH[1:0], PWM3DUTY[7:0]. When PWM3DUTY is 0, PWM3 output will be never active. When PWM3DUTY is 0x3FF, PWM3 output will be active for 1023 Timer3 input clocks. The frame rate is determined by TM34RH[5:4], TMR3[7:0] initial value. Therefore, PWM3DUTY value must be less than or equal to TMR3[9:0]. When user write PWM3DUTY, write PWM3DUTY[9:8] MSB 2 bits(TM34RH[1:0]) first and write PWM3DUTY[7:0] second, PWM3 duty register will be updated after Timer3 overflow occurs. The block diagram of PWM3 is illustrated in the following figure.

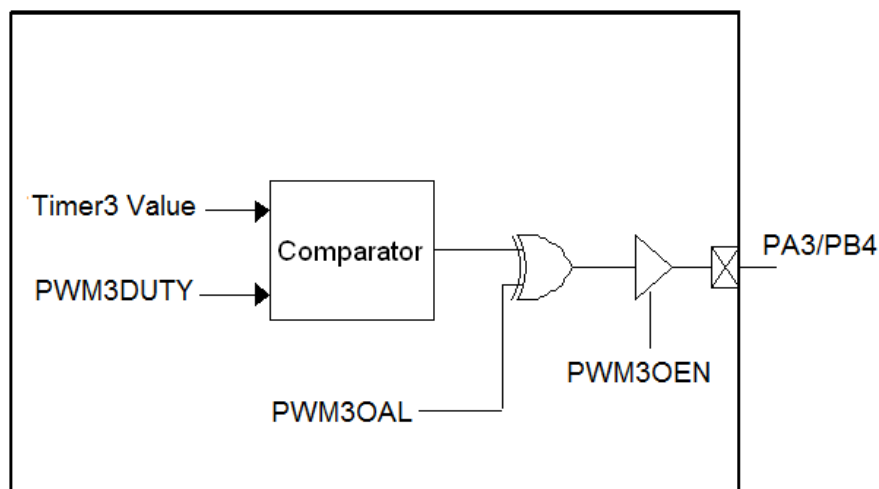


Figure 23 PWM3 Block Diagram

### 3.10 PWM4

The PWM4 output can be available on I/O pin PB5 or PA2 when register bit PWM4OEN (P4CR1[7]) is set to 1. Moreover, PB5 or PA2 will become output pin automatically. The active state of PWM4 output is determined by register bit PWM4OAL (P4CR1[6]). When PWM4OAL is 1, PWM4 output is active low. When PWM4OAL is 0, PWM4 output is active high. Moreover, the duty cycle and frame rate of PWM4 are both programmable. The duty cycle is determined by register TM34RH[3:2],PWM4DUTY[7:0]. When PWM4DUTY is 0, PWM4 output will be never active. When PWM4DUTY is 0x3FF, PWM4 output will be active for 1023 Timer3 input clocks. The frame rate is determined by TM34RH[5:4],TMR3[7:0] initial value. Therefore, PWM4DUTY value must be less than or equal to TMR3[9:0]. Besides, user needs to set P4CR1 (T3EN, T3RL, T3OS data is as the same as T3CR1) because PWM4 and PWM3 are both shared with Timer3. When user write PWM4DUTY, write PWM4DUTY[9:8] MSB 2 bits(TM34RH[3:2]) first and write PWM4DUTY[7:0] second, PWM4 duty register will be updated after Timer3 overflow occurs. The block diagram of PWM4 is illustrated in the following figure.

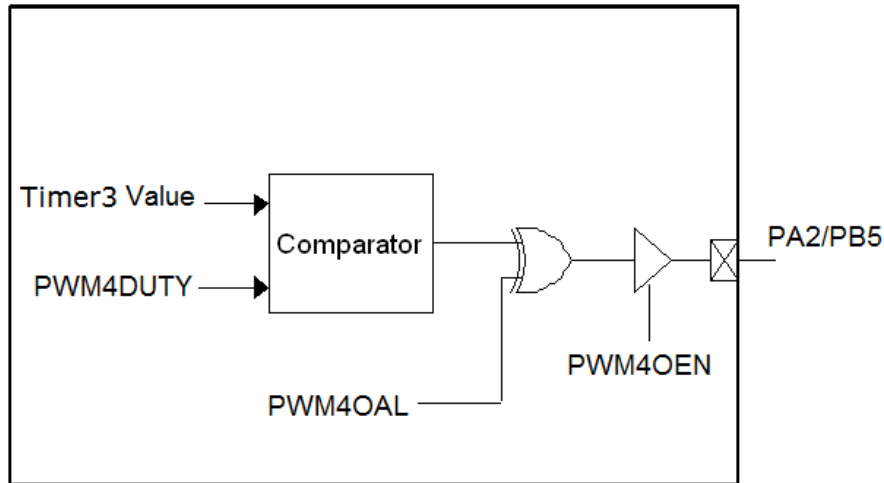


Figure 24 PWM4 Block Diagram

### 3.11 PWM5

The PWM5 output can be available on I/O pin PA1 or PB0 when register bit PWM5OEN (P5CR1[7]) is set to 1. Moreover, PA1 or PB0 will become output pin automatically. The active state of PWM5 output is determined by register bit PWM5OAL (P5CR1[6]). When PWM5OAL is 1, PWM5 output is active low. When PWM5OAL is 0, PWM5 output is active high. Moreover, the duty cycle and frame rate of PWM5 are both programmable. The duty cycle is determined by register PWM5RH[1:0],PWM5DUTY[7:0]. When PWM5DUTY is 0, PWM5 output will be never active. When PWM5DUTY is 0x3FF, PWM5 output will be active for 1023 Timer3 input clocks. The frame rate is determined by TM34RH[5:4],TMR3[7:0] initial value. Therefore, PWM5DUTY value must be less than or equal to TMR3[9:0]. Besides, user needs to set P5CR1 (T3EN, T3RL, T3OS data is as the same as T3CR1) because PWM5 and PWM3 are both shared with Timer3. When user write PWM5DUTY, write PWM5DUTY[9:8] MSB 2 bits(PWM5RH[1:0]) first and write PWM5DUTY[7:0] second, PWM5 duty register will be updated after Timer3 overflow occurs. The block diagram of PWM5 is illustrated in the following figure.

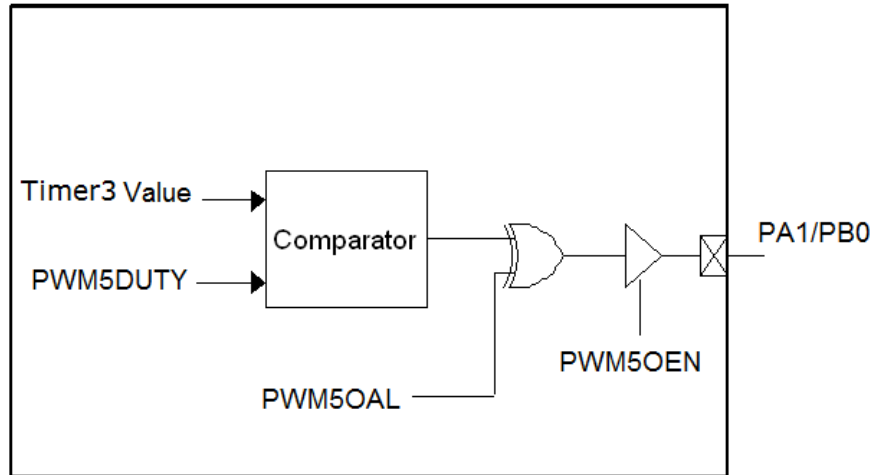


Figure 25 PWM5 Block Diagram

### 3.12 RFC Mode

NY8A054E has built-in RFC mode. Once RFC mode is enabled, the selected input pad state will take control of the Timer1 counting. When the selected input pad is recognized as 0 state (The input pad voltage is smaller than  $V_{IL}$ ), Timer1 keeps counting. When this selected pad is recognized as 1 (The input pad voltage is larger than  $V_{IH}$ ), Timer1 stops counting. The following figure shows how RFC mode operates: PSEL3~0 is used to select one RFC input pad out of 14 NY8A054E pads. RFCEN is used to switch the Timer1 enable signal between the normal enable signal T1EN and RFC selected input state.

One application of RFC mode is to measure the capacitor-resistor charging time, As the figure shows, when PSEL3~0=0x01, PA1 is selected as RFC input pad. At first the PA1 is set as output low (the voltage of PA1 is discharged to 0). Next step, clear Timer1 content, set PA1 as input and enable RFC mode. Then Timer1 will start counting, and the RC circuit will start charging PA1. As PA1 is charged to the  $V_{IH}$  voltage, the Timer1 counting is stopped because PA1 input is high. The Timer1 content will show the RC circuit charging time. (Note: Timer1 is down-count.)

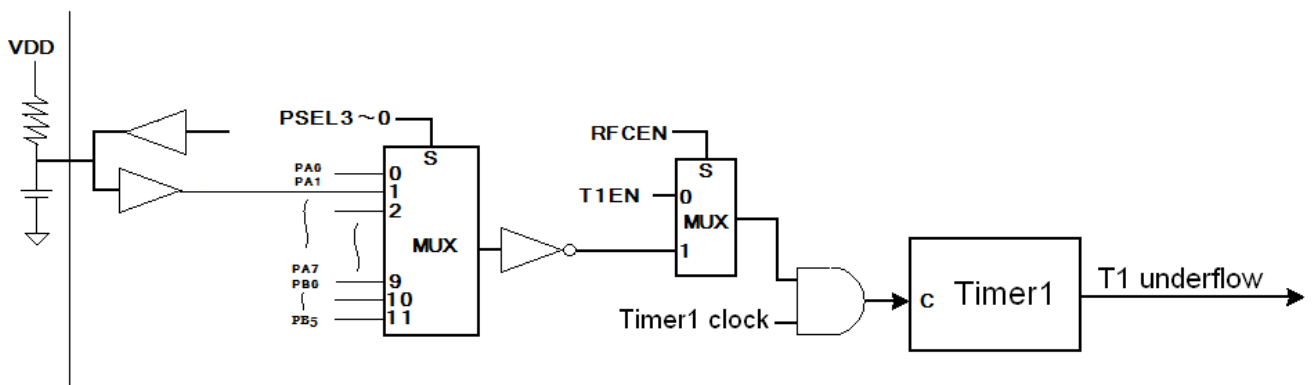


Figure 26 RFC Block Diagram

### 3.13 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 NY8A054E 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.

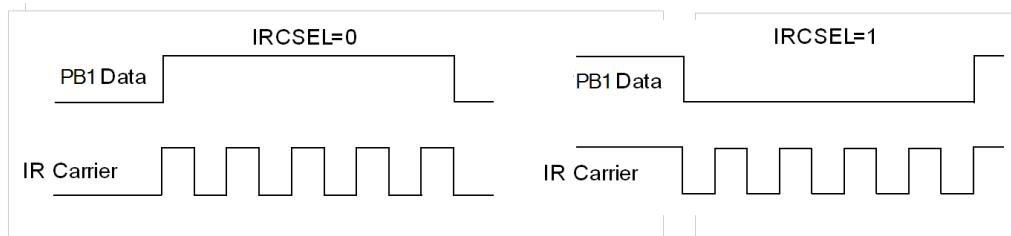


Figure 27 Polarity of IR Carrier vs. Output Data

### 3.14 Low Voltage Detector (LVD)

NY8A054E 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:

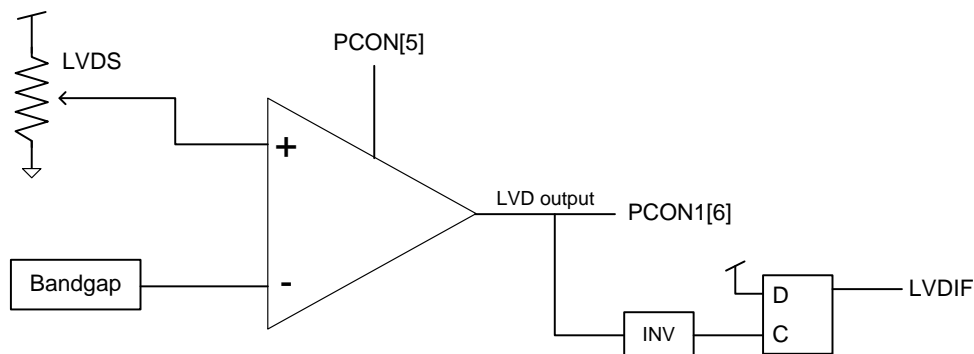


Figure 28 LVD block diagram

The following table is LVD voltage select table.

LVDS[3:0]	Voltage
0000	1.9V
0001	2.0V
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 17 LVD voltage select

The LVD control flow is as the following:

- Step1: Select LVD voltage by LVDS[3:0]*
- Step2: Set CMPCR = \$0A*
- Step3: Set PCOM[5]=1 (enable LVD)*
- Step4: Check LVD status by PCON1[6]*

### 3.15 Voltage Comparator

NY8A054E provides voltage comparator and internal reference voltage with various analog comparing mode. The comparator non-inverting and inverting input can share with GPIO.

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

The structure of comparator is shown in the following figure:

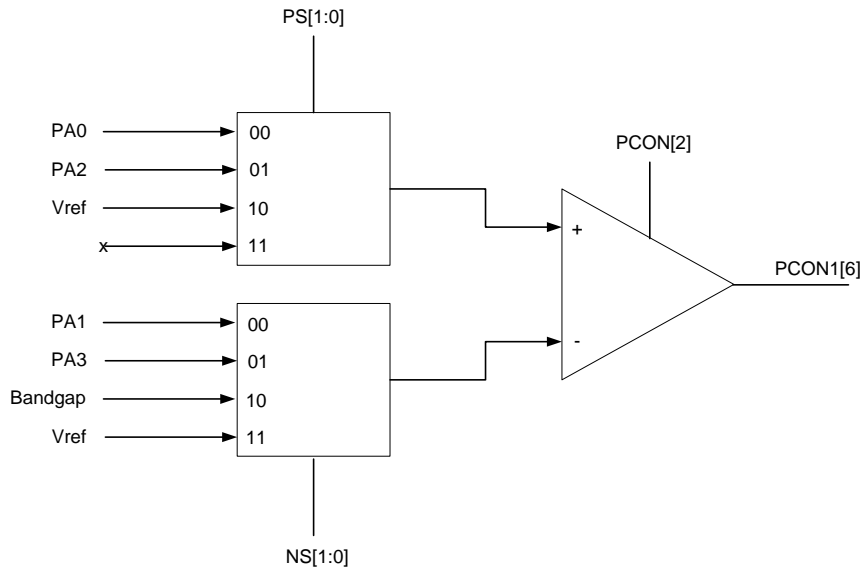


Figure 29 Comparator block diagram

### 3.15.1 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.

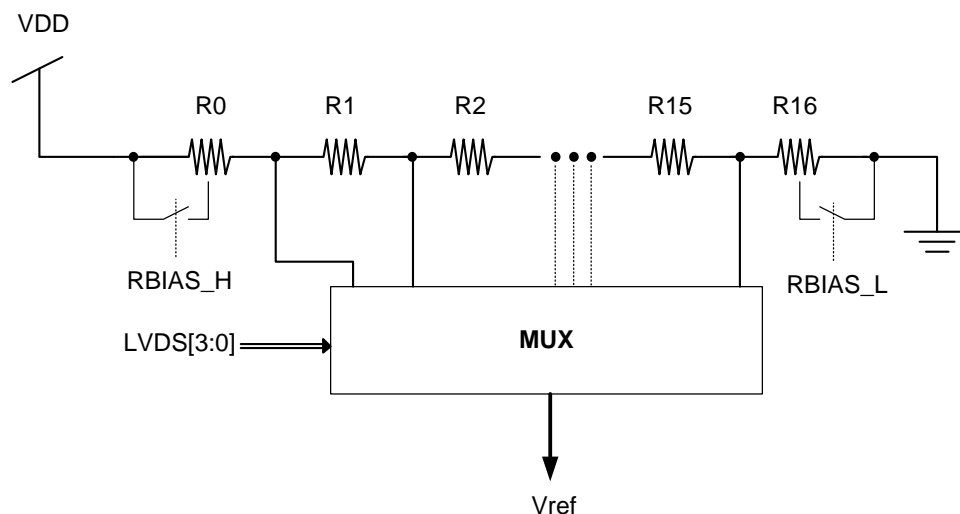


Figure 30 Vref hardware connection



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 RBIAS_L=0	RBIAS_H=0 RBIAS_L=1	RBIAS_H=1 RBIAS_L=1
0000	65/128 V <sub>DD</sub>	31/128 V <sub>DD</sub>	54/128 V <sub>DD</sub>
0001	62/128 V <sub>DD</sub>	29/128 V <sub>DD</sub>	50/128 V <sub>DD</sub>
0010	56/128 V <sub>DD</sub>	25/128 V <sub>DD</sub>	43/128 V <sub>DD</sub>
0011	51/128 V <sub>DD</sub>	22/128 V <sub>DD</sub>	38/128 V <sub>DD</sub>
0100	48/128 V <sub>DD</sub>	19/128 V <sub>DD</sub>	33/128 V <sub>DD</sub>
0101	44/128 V <sub>DD</sub>	17/128 V <sub>DD</sub>	29/128 V <sub>DD</sub>
0110	43/128 V <sub>DD</sub>	16/128 V <sub>DD</sub>	27/128 V <sub>DD</sub>
0111	41/128 V <sub>DD</sub>	15/128 V <sub>DD</sub>	26/128 V <sub>DD</sub>
1000	39/128 V <sub>DD</sub>	14/128 V <sub>DD</sub>	23/128 V <sub>DD</sub>
1001	37/128 V <sub>DD</sub>	12/128 V <sub>DD</sub>	21/128 V <sub>DD</sub>
1010	36/128 V <sub>DD</sub>	11/128 V <sub>DD</sub>	19/128 V <sub>DD</sub>
1011	34/128 V <sub>DD</sub>	10/128 V <sub>DD</sub>	18/128 V <sub>DD</sub>
1100	33/128 V <sub>DD</sub>	9/128 V <sub>DD</sub>	16/128 V <sub>DD</sub>
1101	32/128 V <sub>DD</sub>	8/128 V <sub>DD</sub>	14/128 V <sub>DD</sub>
1110	31/128 V <sub>DD</sub>	8/128 V <sub>DD</sub>	13/128 V <sub>DD</sub>
1111	30/128 V <sub>DD</sub>	7/128 V <sub>DD</sub>	12/128 V <sub>DD</sub>

Table 18 The reference voltage select

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 19 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 (0.6V)
11	Vref

Table 20 Inverting input select

Comparator output can be polled by PCON1[6].

### 3.16 Watch-Dog Timer (WDT)

There is an on-chip free-running oscillator in NY8A054E 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 WD TEN (PCON[7]) during program execution. Moreover, the mechanism after WDT time-out can reset NY8A054E 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 NY8A054E and dividing rate is from 1:2 to 1:256 if WDT time-out will interrupt NY8A054E.

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.

### 3.17 Interrupt

NY8A054E 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 8 hardware interrupts:

- Timer0 overflow interrupt.
- Timer1 underflow interrupt.
- Timer3 underflow interrupt.
- WDT timeout interrupt.
- PA/PB input change interrupt.
- External 0 interrupt.
- External 1 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 NY8A054E 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 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 the corresponding bit of interrupt flag. Note that only when the corresponding interrupt enable bit is set to 1, will the corresponding interrupt flag be read. And if the corresponding interrupt enable bit 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 NY8A054E 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 NY8A054E 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.

### 3.17.1 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.

### 3.17.2 Timer1 Underflow Interrupt

Timer1 underflow (from 0x3FF to 0x00) will set register bit T1IF. This interrupt request will be serviced if T1IE and GIE are set to 1.

### 3.17.3 Timer3 Underflow Interrupt

Timer3 underflow (from 0x3FF to 0x00) will set register bit T3IF. This interrupt request will be serviced if T3IE and GIE are set to 1.

### 3.17.4 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.

### 3.17.5 PA/PB Input Change Interrupt

When  $PA_x$ ,  $0 \leq x \leq 7$ ,  $PB_y$ ,  $0 \leq y \leq 5$  is configured as input pin and corresponding register bit  $WUPA_x$ ,  $WUPB_x$  is set to 1, a level change on these selected I/O pin(s) will set register bit PABIF. This interrupt request will be serviced if PABIE and GIE are set to 1. Note when PB0 or PB1 is both set as level change interrupt and external interrupt, the external interrupt enable  $EIS0$  or  $EIS1=1$  will disable PB0 or PB1 level change operation.

### 3.17.6 External 0 Interrupt

According to the configuration of  $EIS0=1$  and INTEDG, the selected active edge on I/O pin PB0 will set register bit INT0IF and this interrupt request will be served if INT0IE and GIE are set to 1.

### 3.17.7 External 1 Interrupt

According to the configuration of  $EIS1=1$  and INTEDG, the selected active edge on I/O pin PB1 will set register bit INT1IF and this interrupt request will be served if INT1IE and GIE are set to 1.

### 3.17.8 LVD Interrupt

When  $V_{DD}$  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.

### 3.18 Oscillation Configuration

Because NY8A054E is a dual-clock IC, there are high oscillation ( $F_{HOSC}$ ) and low oscillation ( $F_{LOSC}$ ) that can be selected as system oscillation ( $F_{OSC}$ ). The oscillators which could be used as  $F_{HOSC}$  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  $F_{LOSC}$  are internal low RC oscillator (I\_LRC) and external low crystal oscillator (E\_LXT).

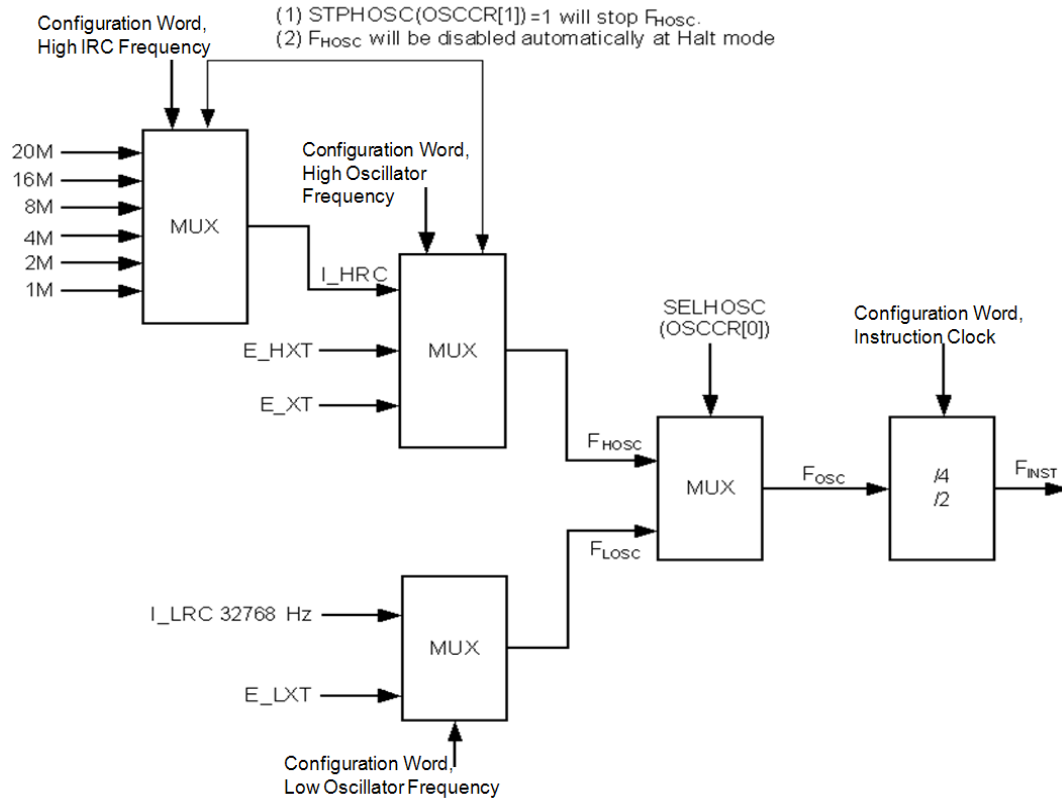


Figure 31 Oscillation Configuration of NY8A054E

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 PA6 and PA7 can be used as I/O pins. On the other hand, PA7 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, PA6/PA7 cannot be used as I/O pins. They must be used as crystal output pin and input pin. PA7 is crystal output pin (Xout) and PA6 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, PA6/PA7 cannot be used as I/O pins. They must be used as crystal output pin and input pin. PA7 is crystal output pin (Xout) and PA6 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 21 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	5 ~ 30

Table 22 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.

To get precise and stable 32.768k frequency, choosing the right C1 and C2 value is important. You need to match the C1 / C2 capacitance to the specific crystal you chose. Every crystal datasheet lists something called the Load Capacitance (CL), C1 and C2 value is chosen with the following formula:

$$C1=C2=2*CL-Cbt$$

Where Cbt is the NY8A054E crystal pad built-in capacitance, which is about 5pF. For example, for crystal CL=12.5P, C1=C2=20pF is recommended.

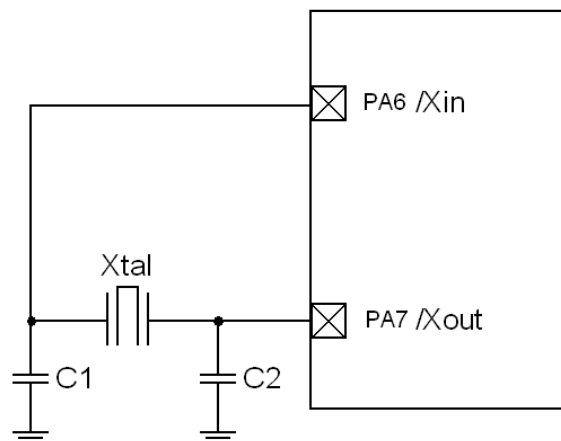


Figure 32 Connection for External Crystal Oscillation

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.

### 3.19 Operating Mode

NY8A054E 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, NY8A054E will stop almost all operations except Timer0/Timer1/ Timer3 WDT in order to wake-up periodically. At Halt mode, NY8A054E 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.

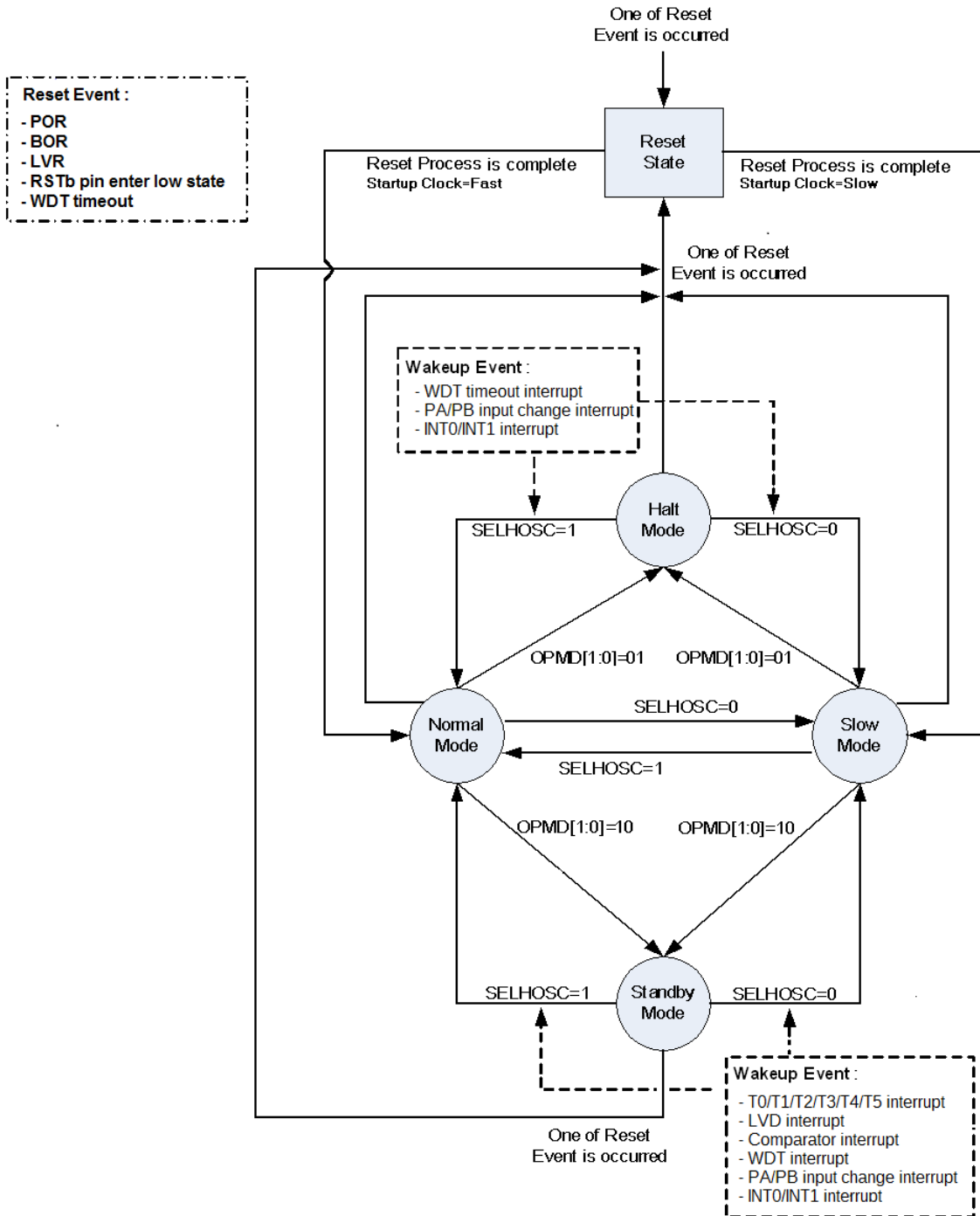


Figure 33 Four Operating Modes



### 3.19.1 Normal Mode

After any Reset Event is occurred and Reset Process is completed, NY8A054E 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, NY8A054E will enter Normal mode, if Startup Clock=Slow, NY8A054E 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, NY8A054E 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 NY8A054E 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.

### 3.19.2 Slow Mode

NY8A054E 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 NY8A054E. 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.

### 3.19.3 Standby Mode

NY8A054E 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 NY8A054E and user has to enter slow mode and write 1 to register bit STPHOSC in order to stop  $F_{HOSC}$  oscillation. Most of NY8A054E peripheral modules are disabled but Timer can be still active if register bit T0EN/T1EN/T3EN is set to 1. Therefore NY8A054E can wake-up after Timer0/Timer1/Timer3 is expired. The expiration period is determined by the register TMR0/TMR1[9:0]/TMR3[9:0],  $F_{INST}$  and other configurations for Timer0/Timer1/Timer3.

- 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 the following conditions:

- (a) Timer0/Timer1/Timer3 (overflow/underflow) interrupt
  - (b) WDT timeout interrupt
  - (c) PA/PB input change interrupt
  - (d) INT0/INT1 external interrupt is happened.
- 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.

### 3.19.4 Halt Mode

NY8A054E 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 NY8A054E can only wake-up by some specific events. Therefore, Halt mode is the most power saving mode provided by NY8A054E.

- 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 the following conditions:

- (a) WDT timeout interrupt
- (b) PA/PB input change interrupt
- (c) INT0/INT1 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.

### 3.19.5 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 NY8A054E enter Standby mode or Halt mode, user may execute instruction ENI. At this condition, NY8A054E will branch to address 0x008 in order to execute interrupt service routine after wake-up. If instruction DISI is executed before entering Standby mode or Halt mode, the next instruction will be executed after wake-up.

### 3.19.6 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
Timer0/1/3	TxEN	TxEN	TxEN	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	-	-	- Timer0/1/3 overflow - WDT timeout - PA/PB input change - INT0/1 - LVD interrupt	- WDT timeout - PA/PB input change - INT0/1

Table 23 Summary of Operating Modes

### 3.20 Reset Process

NY8A054E 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 24 Summary of /TO & /PD Value and its Associated Event

After Reset Event is released, NY8A054E 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 three-bit configuration words which can be 140us, 4.5ms, 18ms, 72ms or 288ms. After power-up reset time, NY8A054E will wait for further oscillator start-up time (OST) before it starts to execute program. OST=1 clock cycle of  $F_{OSC}$  if the previous power-up time is 140us, OST=16 clock cycles of  $F_{OSC}$  if the previous power-up time is 4.5ms, 18ms, 72ms or 288ms.

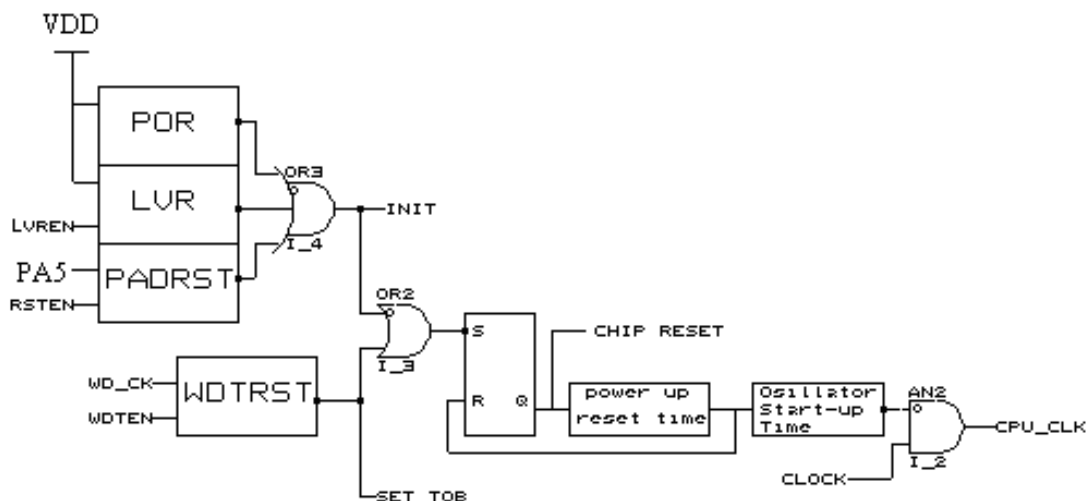


Figure 34 Block diagram of on-chip reset circuit

For slow  $V_{DD}$  power-up, it is recommended to use RSTb reset, as the following figure.

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

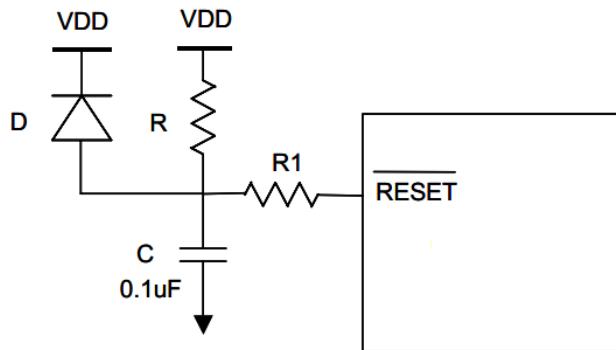


Figure 35 Block Diagram of Reset Application

#### 4. Instruction Set

NY8A054E 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 = ~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 + (~ACC)	1	Z, DC, C
ADCAR	R	d	dest = R + ACC + C	1	Z, DC, C
SBCAR	R	d	dest = R + (~ACC) + C	1	Z, DC, C
ADDIA	i		ACC = I + ACC	1	Z, DC, C
SUBIA	i		ACC = i + (~ACC)	1	Z, DC, C
ADCIA	i		ACC = i + ACC + C	1	Z, DC, C
SBCIA	i		ACC = i + (~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 25 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 0x0 ~0x7F.

S: S-page SFR, S is 0x0 ~ 0x1F.

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:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	$R + ACC + C \rightarrow dest$
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 before executing instruction: ACC=0x12, R=0x34, C=1, d=1, after executing instruction: R=0x47, ACC=0x12, C=0.

<b>ADDAR</b>	<b>Add ACC and R</b>
Syntax:	ADDAR R, d
Operand:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	$ACC + R \rightarrow dest$
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 before executing instruction: ACC=0x12, R=0x34, C=1, d=1, after executing instruction: 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 before executing instruction: ACC=0x12, i=0x34, C=1, after executing instruction: 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 before executing instruction: ACC=0x12, i=0x34, C=1, after executing instruction: ACC=0x46, C=0.



<b>ANDAR</b>	<b>AND ACC and R</b>
Syntax:	ANDAR R, d
Operand:	$0 \leq R \leq 127$ $d = 0, 1.$
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 before executing instruction: ACC=0x5A, R=0XaF, d=1. after executing instruction: R=0x0A, ACC=0x5A, Z=0.

<b>BCR</b>	<b>Clear Bit in R</b>
Syntax:	BCR R, bit
Operand:	$0 \leq R \leq 127$ $0 \leq \text{bit} \leq 7$
Operation:	$0 \rightarrow R[\text{bit}]$
Status affected:	--
Description:	Clear the bit <sup>th</sup> position in R.
Cycle:	1
Example:	BCR R, B2 before executing instruction: R=0x5A, B2=0x3, after executing instruction: 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 before executing instruction: ACC=0x5A, i=0XaF, after executing instruction: ACC=0x0A, Z=0.

<b>BSR</b>	<b>Set Bit in R</b>
Syntax:	BSR R, bit
Operand:	$0 \leq R \leq 127$ $0 \leq \text{bit} \leq 7$
Operation:	$1 \rightarrow R[\text{bit}]$
Status affected:	--
Description:	Set the bit <sup>th</sup> position in R.
Cycle:	1
Example:	BSR R, B2 before executing instruction: R=0x5A, B2=0x2, after executing instruction: R=0x5E.

<b>BTRSC</b>	<b>Test Bit in R and Skip if Clear</b>
Syntax:	BTRSC R, bit
Operand:	$0 \leq R \leq 127$ $0 \leq \text{bit} \leq 7$
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 Instruction1 Instruction2 before executing instruction: R=0x5A, B2=0x2, after executing instruction: 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 → Top of Stack {PCHBUF, adr} → 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[2:0] is loaded into PC[10:8].
Cycle:	2
Example:	CALL SUB before executing instruction: PC=A0. Stack pointer=1 after executing instruction: 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:	$0 \leq R \leq 127$ $0 \leq \text{bit} \leq 7$
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 Instruction2 Instruction3 before executing instruction: R=0x5A, B2=0x3, after executing instruction: 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 → Top of Stack {TBHP, ACC} → PC
Status affected:	--
Description:	The return address (PC + 1) is pushed onto top of Stack. The contents of TBHP[2:0] is loaded into PC[10:8] and ACC is loaded into PC[7:0].
Cycle:	2
Example:	CALLA before executing instruction: TBHP=0x02, ACC=0x34. PC=A0. Stack pointer=1. after executing instruction: PC=0x234, Stack[1]=A0+1, Stack pointer=2

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

<b>CLRWDT</b>	<b>Clear Watch-Dog Timer</b>
Syntax:	CLRWDT
Operand:	--
Operation:	00h → WDT, 00h → WDT prescaler 1 → /TO 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 before executing instruction: /TO=0 after executing instruction: /TO=1

<b>CLRR</b>	<b>Clear R</b>
Syntax:	CLRR R
Operand:	$0 \leq R \leq 127$
Operation:	00h → R 1 → Z
Status affected:	Z
Description:	The content of R is clear and Z is set to 1.
Cycle:	1
Example:	CLRR R before executing instruction: R=0x55, Z=0. after executing instruction: R=0x00, Z=1.

<b>COMR</b>	<b>Complement R</b>
Syntax:	COMR R, d
Operand:	$0 \leq R \leq 127$ 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 before executing instruction: R=0xA6, d=1, Z=0. after executing instruction: R=0x59, Z=0.

<b>CMPAR</b>	<b>Compare ACC and R</b>
Syntax:	CMPAR R
Operand:	$0 \leq R \leq 127$
Operation:	R ← ACC → (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 before executing instruction: R=0x34, ACC=12, Z=0, C=0. after executing instruction: R=0x34, ACC=12, Z=0, C=1.

<b>DECR</b>	<b>Decrease R</b>
Syntax:	DECR R, d
Operand:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	R ← 1 → 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 before executing instruction: R=0x01, d=1, Z=0. after executing instruction: R=0x00, Z=1.

<b>DAA</b>	<b>Convert ACC Data Format from Hexadecimal to Decimal</b>
Syntax:	DAA
Operand:	--
Operation:	ACC(hex) → 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 ADDAR R,d DAA ENI before executing instruction: ACC=0x28, R=0x25, d=0. after executing instruction: ACC=0x53, C=0.

<b>DECRSZ</b>	<b>Decrease R, Skip if 0</b>
Syntax:	DECRSZ R, d
Operand:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	R ← 1 → dest, 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 instruction2 instruction3 before executing instruction: R=0x1, d=1, Z=0. after executing instruction: 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 before executing instruction: GIE=1, After executing instruction: 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[2:1] is loaded into PC[10:9].
Cycle:	2
Example:	GOTO Level before executing instruction: PC=A0. after executing instruction: 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 before executing instruction: GIE=0, After executing instruction: 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[2:0] is loaded into PC[10:8] and ACC is loaded into PC[7:0].
Cycle:	2
Example:	GOTOA before executing instruction: PC=A0. TBHP=0x02, ACC=0x34. after executing instruction: PC=0x234

<b>INCR</b>	<b>Increase R</b>
Syntax:	INCR R, d
Operand:	$0 \leq R \leq 127$ 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 before executing instruction: R=0XfF, d=1, Z=0. after executing instruction: R=0x00, Z=1.

<b>INT</b>	<b>Software Interrupt</b>
Syntax:	INT
Operand:	--
Operation:	$PC + 1 \rightarrow \text{Top of Stack,}$ $001h \rightarrow PC$
Status affected:	--
Description:	Software interrupt. First, return address (PC + 1) is pushed onto the Stack. The address 0x001 is loaded into PC[10:0].
Cycle:	3
Example:	INT before executing instruction: PC=address of INT code after executing instruction: PC=0x01

<b>INCRSZ</b>	<b>Increase R, Skip if 0</b>
Syntax:	INCRSZ R, d
Operand:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	$R + 1 \rightarrow \text{dest,}$ 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. 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 instruction2, instruction3. before executing instruction: R=0XfF, d=1, Z=0. after executing instruction: 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:	$0 \leq R \leq 127$ 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 before executing instruction: R=0x50, ACC=0XaA, d=1, Z=0. after executing instruction: R=0XfA, ACC=0XaA, Z=0.

<b>IORIA</b>	<b>OR Immediate with ACC</b>
Syntax:	IORIA i
Operand:	$0 \leq i < 255$
Operation:	ACC   i → ACC
Status affected:	Z
Description:	OR ACC with 8-bit immediate data i. The result is stored in ACC.
Cycle:	1
Example:	IORIA i before executing instruction: i=0x50, ACC=0XaA, Z=0. after executing instruction: 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 → ACC
Status affected:	--
Description:	Move F-page SFR F to ACC.
Cycle:	1
Example:	IOSTR F before executing instruction: F=0x55, ACC=0XaA. after executing instruction: F=0x55, ACC=0x55.

<b>IOST</b>	<b>Load F-page SFR from ACC</b>
Syntax:	IOST F
Operand:	$5 \leq F \leq 15$
Operation:	ACC → F-page SFR
Status affected:	--
Description:	F-page SFR F is loaded by content of ACC.
Cycle:	1
Example:	IOST F before executing instruction: F=0x55, ACC=0XaA. after executing instruction: F=0XaA, ACC=0XaA.

<b>LCALL</b>	<b>Call Subroutine</b>
Syntax:	LCALL adr
Operand:	$0 \leq \text{adr} \leq 2047$
Operation:	PC + 1 → Top of Stack, adr → PC[10:0]
Status affected:	--
Description:	The return address (PC + 1) is pushed onto top of Stack. The 11-bit immediate address adr is loaded into PC[10:0].
Cycle:	2
Example:	LCALL SUB before executing instruction: PC=A0, Stack level=1 after executing instruction: PC=address of SUB, Stack[1]= A0+1, Stack pointer =2.

<b>LGOTO</b>	<b>Unconditional Branch</b>
Syntax:	LGOTO adr
Operand:	$0 \leq \text{adr} \leq 2047$
Operation:	$\text{adr} \rightarrow \text{PC}[10:0]$ .
Status affected:	--
Description:	LGOTO is an unconditional branch instruction. The 11-bit immediate address adr is loaded into PC[10:0].
Cycle:	2
Example:	LGOTO Level before executing instruction: PC=A0. after executing instruction: 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 before executing instruction: i=0x55, ACC=0XaA. after executing instruction: ACC=0x55.

<b>MOVAR</b>	<b>Move ACC to R</b>
Syntax:	MOVAR R
Operand:	$0 \leq R \leq 127$
Operation:	$\text{ACC} \rightarrow R$
Status affected:	--
Description:	Move content of ACC to R.
Cycle:	1
Example:	MOVAR R before executing instruction: R=0x55, ACC=0XaA. after executing instruction: R=0XaA, ACC=0XaA.

<b>MOVR</b>	<b>Move R to ACC or R</b>
Syntax:	MOVR R, d
Operand:	$0 \leq R \leq 127$ $d = 0, 1.$
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 before executing instruction: R=0x0, ACC=0XaA, Z=0, d=0. after executing instruction: 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:	<p>NOP  before executing instruction:  PC=A0  after executing instruction:  PC=A0+1</p>

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

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

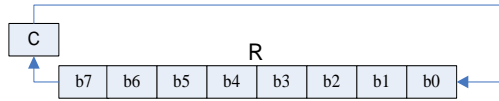
<b>RET</b>	<b>Return from Subroutine</b>
Syntax:	RET
Operand:	--
Operation:	Top of Stack $\rightarrow \text{PC}$
Status affected:	--
Description:	PC is loaded from top of Stack as return address.
Cycle:	2
Example:	<p>RET  before executing instruction:  Stack level=2.  after executing instruction:  PC=Stack[2],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

**RLR Rotate Left R Through Carry**

**Syntax:** RLR R, d

**Operand:**  $0 \leq R \leq 127$   
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

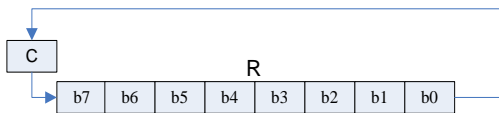
**Example:** RLR R, d  
before executing instruction:  
R=0XA5, d=1, C=0.  
after executing instruction:  
R=0x4A, C=1.

**RRR Rotate Right R Through Carry**

**Syntax:** RRR R, d

**Operand:**  $0 \leq R \leq 127$   
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:**  $0 \leq R \leq 127$   
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, ACC=0x06, d=1,  
C=0,  
after executing instruction:  
R=0xfE, C=0. (-2)

(b) before executing instruction:  
R=0x05, ACC=0x06, d=1,  
C=1,  
after executing instruction:  
R=0xfF, C=0. (-1)| before  
executing instruction:  
R=0x06, ACC=0x05, d=1,  
C=0,  
after executing instruction:  
R=0x00, C=1. (-0), Z=1.

(d) before executing instruction:  
R=0x06, 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:  
            $i=0x05, \text{ACC}=0x06, C=0,$   
           after executing instruction:  
            $\text{ACC}=0xFE, C=0. (-2)$

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

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

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

---

**SFUN          Load S-page SFR from ACC**


---

Syntax:            SFUN    S

Operand:           $0 \leq S \leq 31$

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:  
            $S=0x55, \text{ACC}=0xAA.$   
           after executing instruction:  
            $S=0xAA, \text{ACC}=0xAA.$

---

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


---

Syntax:            SFUNR   S

Operand:           $0 \leq S \leq 31$

Operation:        S-page SFR  $\rightarrow$  ACC

Status affected:   --

Description:      Move S-page SFR S to ACC.

Cycle:             1

Example:          SFUNR S

          before executing instruction:  
            $S=0x55, \text{ACC}=0xAA.$   
           after executing instruction:  
            $S=0x55, \text{ACC}=0x55.$

---

**SLEEP        Enter Halt Mode**


---

Syntax:            SLEEP

Operand:          --

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

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:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	R – ACC → 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:	SBCAR R, d (a) before executing instruction: R=0x05, ACC=0x06, d=1, after executing instruction: R=0xFF, C=0. (-1) (b) before executing instruction: R=0x06, ACC=0x05, d=1, after executing instruction: R=0x01, C=1. (+1)

<b>SWAPR</b>	<b>Swap High/Low Nibble in R</b>
Syntax:	SWAPR R, d
Operand:	$0 \leq R \leq 127$ d = 0, 1.
Operation:	R[3:0] → dest[7:4]. R[7:4] → 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 before executing instruction: R=0xA5, d=1. after executing instruction: R=0x5A.

<b>SUBIA</b>	<b>Subtract ACC from Immediate</b>
Syntax:	SUBIA i
Operand:	$0 \leq i < 255$
Operation:	i – ACC → 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 (a) before executing instruction: i=0x05, ACC=0x06. after executing instruction: ACC=0xFF, C=0. (-1) (b) before executing instruction: i=0x06, ACC=0x05, d=1, after executing instruction: ACC=0x01, C=1. (+1)

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

<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 before executing instruction: T0MD=0x55, ACC=0xAA. after executing instruction: t0MD=0xAA.

<b>XORAR</b>	<b>Exclusive-OR ACC with R</b>
Syntax:	XORAR R, d
Operand:	$0 \leq R \leq 127$ $d = 0, 1.$
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 before executing instruction: R=0xA5, ACC=0xF0, d=1. after executing instruction: 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 before executing instruction T0MD=0x55, ACC=0xAA. after executing instruction 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 before executing instruction: i=0xA5, ACC=0xF0. after executing instruction: 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	
		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_CKIO	2. Low Oscillator (I_LRC/E_LXT)			
9	PA.5	1. PA.5 is I/O	2. PA.5 is reset			
10	PA.7	1. PA.7 is I/O	2. PA.7 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	Analog Input pin select	1. Enable		2. Disable		
17	Read Output Data	1. I/O Port		2. Register		
18	E_LXT Backup Control	1. Auto Off		2. Register Off		
19	EX_CKIO to Inst. Clock	1. Sync		2. Async		
20	Startup Clock	1. Fast (I_HRC/E_HXT/E_XT)		2. Slow (I_LRC/E_LXT)		
21	PWM1 Output Pin	1. PB3		2. PA4		
22	PWM2 Output Pin	1. PB2		2. PA0		
23	PWM3 Output Pin	1. PB4		2. PA3		
24	PWM4 Output Pin	1. PB5		2. PA2		
25	PWM5 Output Pin	1. PB0		2. PA1		
26	Input Schmitt Trigger	1. Enable		2. Disable (1/2 VDD)		
27	Input High Voltage (VIH)	1. 0.7VDD		2. 0.5VDD		
28	Input Low Voltage (VIL)	1. 0.3VDD		2. 0.2VDD		
29	INT1 pad select	1. PB1		2. PA2		

Table 26 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				$F_{INST}=16MHz @ E_{HXT}/2$
			2.0				$F_{INST}=16MHz @ E_{HXT}/4$
			2.0				$F_{INST}=8MHz @ I_{HRC}/4 \& I_{HRC}/2$
			2.0				$F_{INST}=8MHz @ E_{HXT}/4 \& E_{HXT}/2$
			1.8				$F_{INST}=4MHz @ I_{HRC}/4 \& I_{HRC}/2$
			1.8				$F_{INST}=4MHz @ E_{XT}/4 \& E_{XT}/2$
			1.8				$F_{INST}=32KHz @ I_{LRC}/4 \& I_{LRC}/2$
			1.8				$F_{INST}=32KHz @ E_{LXT}/4 \& E_{LXT}/2$
$V_{IH}$	Input high voltage	5V	4.0	--	--	V	RSTb (0.8 $V_{DD}$ )
		3V	2.4	--	--	V	All other I/O pins, EX_CKI, INT CMOS option (0.7 $V_{DD}$ )
		5V	3.5	--	--		
		3V	2.1	--	--	V	All other I/O pins, EX_CKI TTL option (0.5 $V_{DD}$ )
		5V	2.5	--	--		
$V_{IL}$	Input low voltage	5V	--	--	1.0	V	RSTb (0.2 $V_{DD}$ )
		3V	--	--	0.6	V	All other I/O pins, EX_CKI, INT CMOS option (0.3 $V_{DD}$ )
		5V	--	--	1.5		
		3V	--	--	0.9	V	All other I/O pins, EX_CKI TTL option (0.2 $V_{DD}$ )
		5V	--	--	1.0		
3V	--	--	0.6				
$I_{OH}$	Output high current	5V	--	18	--	mA	$V_{OH}=4.0V$
		3V	--	10	--		$V_{OH}=2.0V$
$I_{OL}$	Output low current	5V	--	41	--	mA	$V_{OL}=1.0V$
		3V	--	26	--		
$I_{IR}$	IR sink current	5V	--	43	--	mA	$V_{OL}=1.0V, IR$
		3V	--	27	--		

Symbol	Parameter	V <sub>DD</sub>	Min.	Typ.	Max.	Unit	Condition
I <sub>OP</sub>	Operating current	<b>Normal Mode</b>					
		5V	--	3.1	--	mA	F <sub>HOSC</sub> =20MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2
		3V	--	1.4	--		
		5V	--	2.1	--	mA	F <sub>HOSC</sub> =20MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4
		3V	--	0.9	--		
		5V	--	2.6	--	mA	F <sub>HOSC</sub> =16MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2
		3V	--	1.1	--		
		5V	--	1.8	--	mA	F <sub>HOSC</sub> =16MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4
		3V	--	0.8	--		
		5V	--	1.6	--	mA	F <sub>HOSC</sub> =8MHz @ I <sub>HRC</sub> /2 & E <sub>HXT</sub> /2
		3V	--	0.7	--		
		5V	--	1.2	--	mA	F <sub>HOSC</sub> =8MHz @ I <sub>HRC</sub> /4 & E <sub>HXT</sub> /4
		3V	--	0.5	--		
		5V	--	1.2	--	mA	F <sub>HOSC</sub> =4MHz @ I <sub>HRC</sub> /2 & E <sub>XT</sub> /2
		3V	--	0.5	--		
		5V	--	1.0	--	mA	F <sub>HOSC</sub> =4MHz @ I <sub>HRC</sub> /4 & E <sub>XT</sub> /4
		3V	--	0.4	--		
		5V	--	0.9	--	mA	F <sub>HOSC</sub> =1MHz @ I <sub>HRC</sub> /2 & E <sub>XT</sub> /2
		3V	--	0.4	--		
		5V	--	0.8	--	mA	F <sub>HOSC</sub> =1MHz @ I <sub>HRC</sub> /4 & E <sub>XT</sub> /4
3V	--	0.3	--				
<b>Slow Mode</b>							
I <sub>STB</sub>	Standby current	5V	--	6.9	--	uA	F <sub>HOSC</sub> disabled, F <sub>LOSC</sub> =32KHz @ I <sub>LRC</sub> /2
		3V	--	3.0	--		
I <sub>HALT</sub>	Halt current	5V	--	7.9	--	uA	F <sub>HOSC</sub> disabled, F <sub>LOSC</sub> =32KHz @ E <sub>LXT</sub> /2.
		3V	--	3.2	--		
I <sub>HALT</sub>	Halt current	5V	--	4.8	--	uA	F <sub>HOSC</sub> disabled, F <sub>LOSC</sub> =32KHz @ I <sub>LRC</sub> /4
		3V	--	2.0	--		
I <sub>HALT</sub>	Halt current	5V	--	5.8	--	uA	F <sub>HOSC</sub> disabled, F <sub>LOSC</sub> =32KHz @ E <sub>LXT</sub> /4.
		3V	--	2.2	--		
I <sub>STB</sub>	Standby current	5V	--	3.2	--	uA	Standby mode, F <sub>HOSC</sub> disabled, F <sub>LOSC</sub> =32KHz @ I <sub>LRC</sub> /4
		3V	--	1.2	--		
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	--	--	2.0		
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	%	

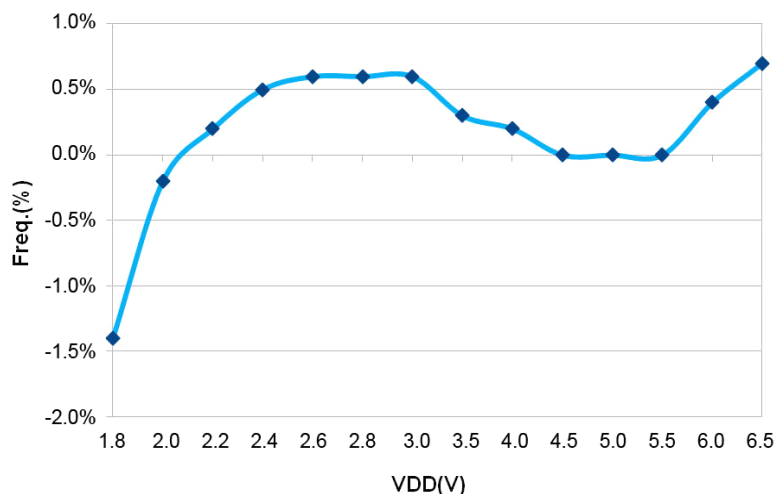
### 6.4 Comparator / LVD Characteristics

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

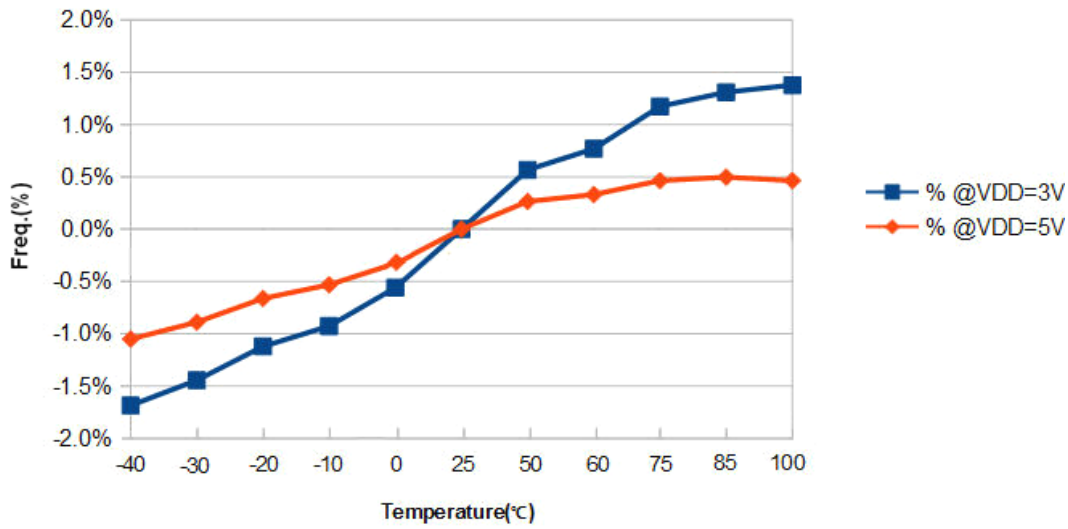
Symbol	Parameter	Min.	Typ.	Max.	Unit	Condition
$V_{IVR}$	Comparator input voltage range	0	--	5	V	$F_{HOSC}=1MHz$
$T_{ENO}$	Comparator enable to output valid	--	20	--	ms	$F_{HOSC}=1MHz$
$I_{CO}$	Operating current of comparator	--	250	--	uA	$F_{HOSC}=1MHz$ , P2V mode
$I_{LVD}$	Operating current of LVD	--	300	--	uA	$F_{HOSC}=1MHz$ , LVD=4.3V
$E_{LVD}$	LVD voltage error	--	--	3	%	$F_{HOSC}=1MHz$ , LVD=4.3V

### 6.5 Characteristic Graph

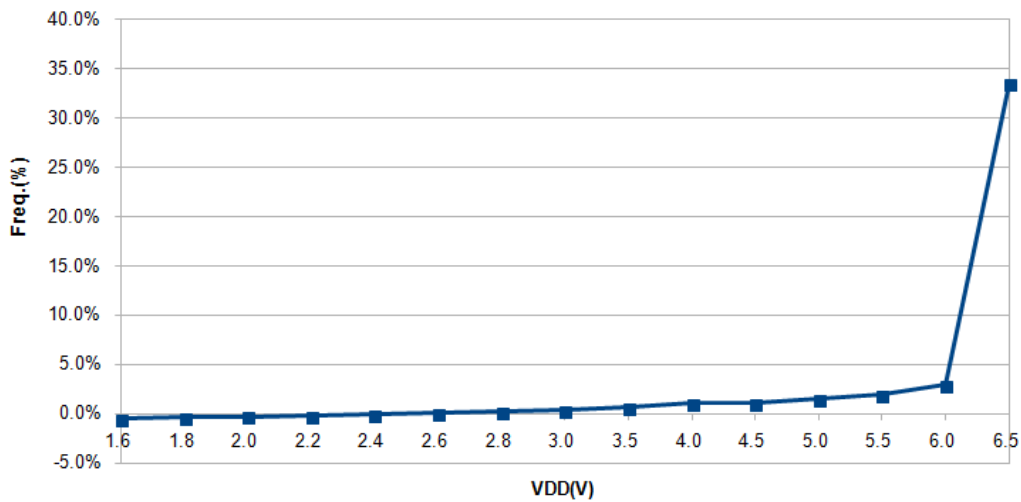
#### 6.5.1 Frequency vs. $V_{DD}$ of I\_HRC



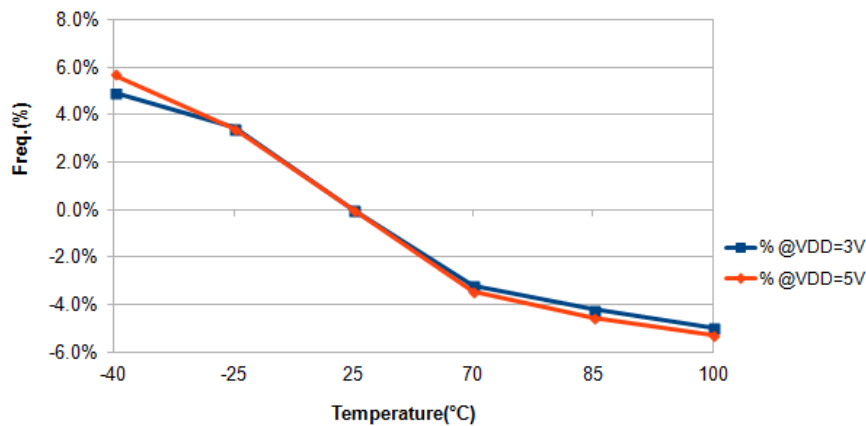
6.5.2 Frequency vs. Temperature of I\_HRC



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



6.5.4 Frequency vs. Temperature of I\_LRC

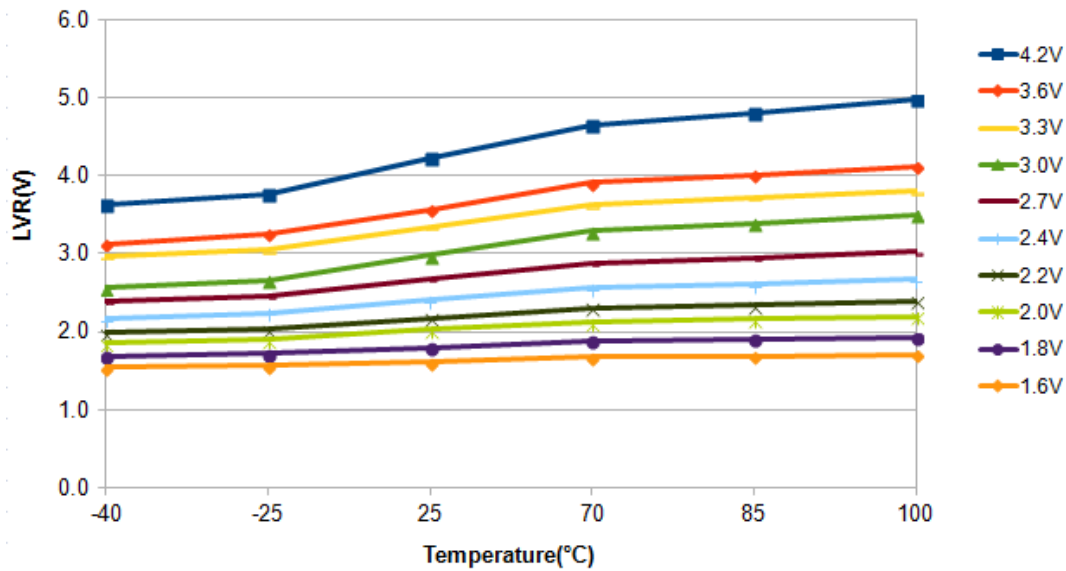


## 6.6 Recommended Operating Voltage

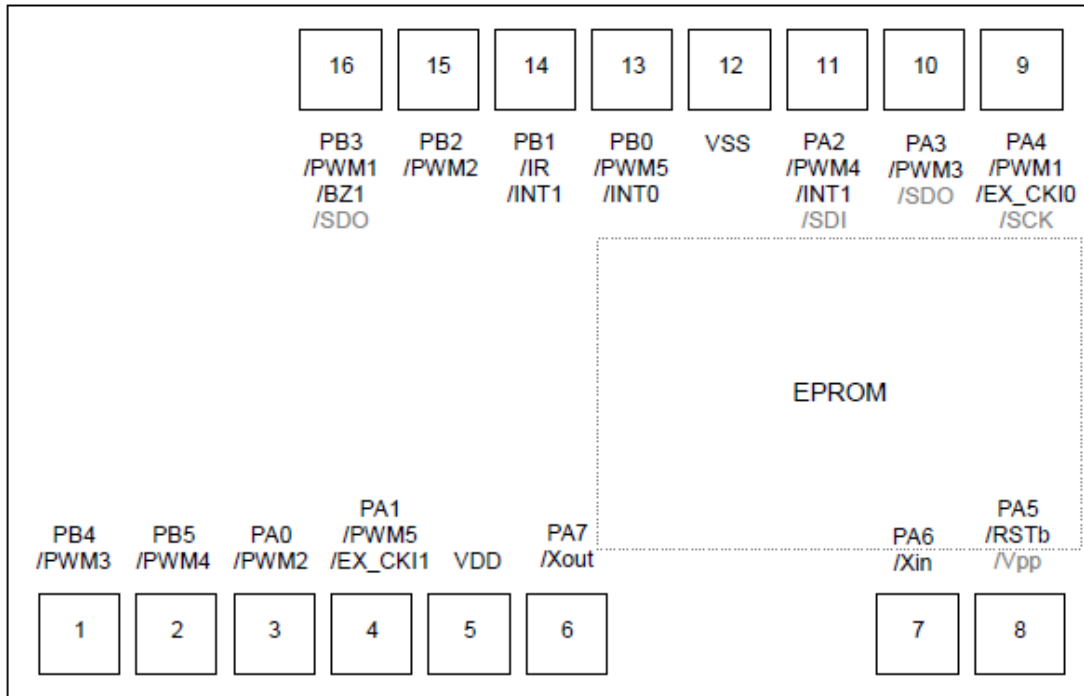
Recommended Operating Voltage (Temperature range: -40°C ~ +85°C)

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

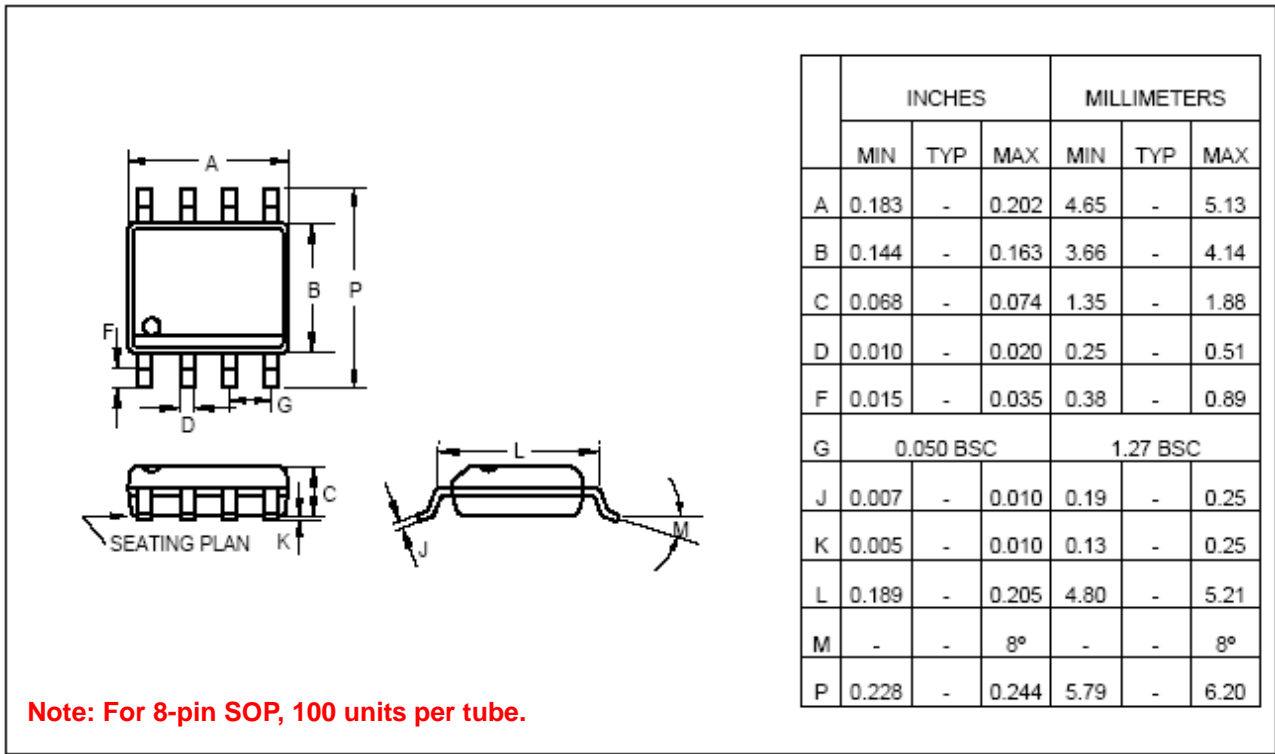


7. Die Pad Diagram

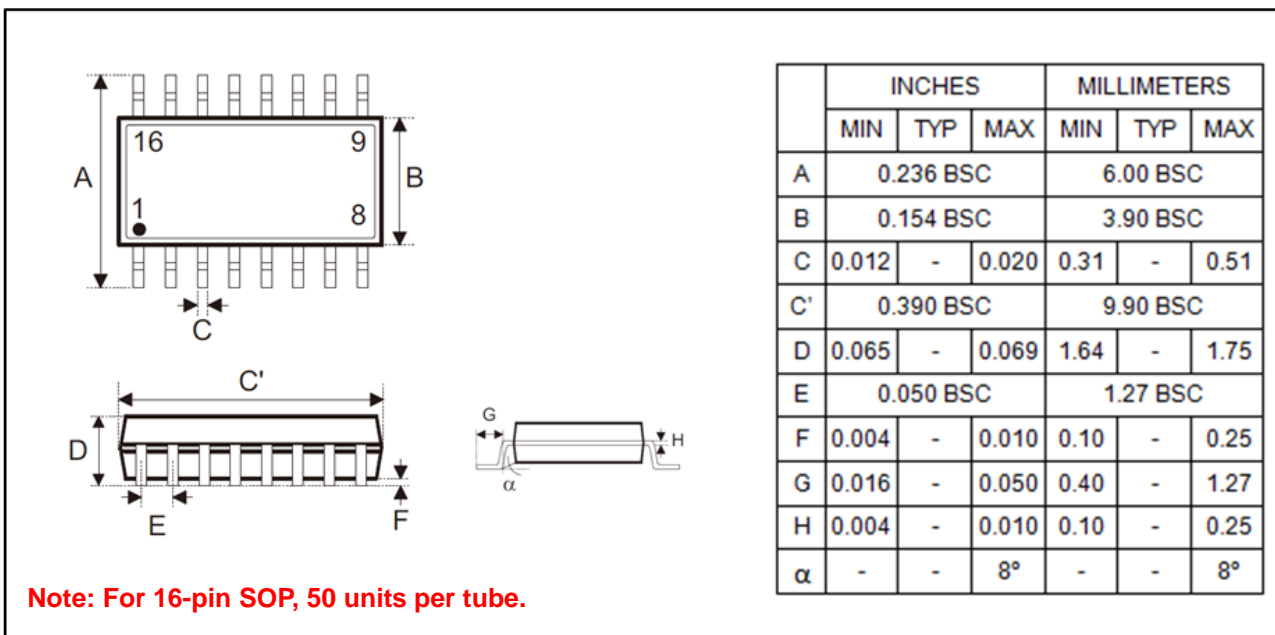


8. Package Dimension

8.1 8-Pin Plastic SOP (150 mil)



8.2 16-Pin Plastic SOP (150 mil)



**9. Ordering Information**

<i>P/N</i>	<i>Package Type</i>	<i>Pin Count</i>	<i>Package Width</i>	<i>Shipping</i>
NY8A054E	Die	--	--	--
NY8A054ES8	SOP	8	150 mil	<u>Tape &amp; Reel</u> : 2.5K pcs per Reel <u>Tube</u> : 100 pcs per Tube
NY8A054ES16	SOP	16	150 mil	<u>Tape &amp; Reel</u> : 2.5K pcs per Reel <u>Tube</u> : 50 pcs per Tube