DATA SHEET

Generalplus

GPM6C1064A1

Universal Remote Controller with 64KB ROM

Preliminary

APR. 19, 2011

Version 0.1

Table of Contents

PAGE

UNIVERSAL REMOTE CONTROLLER WITH 64KB ROM

1. GENERAL DESCRIPTION

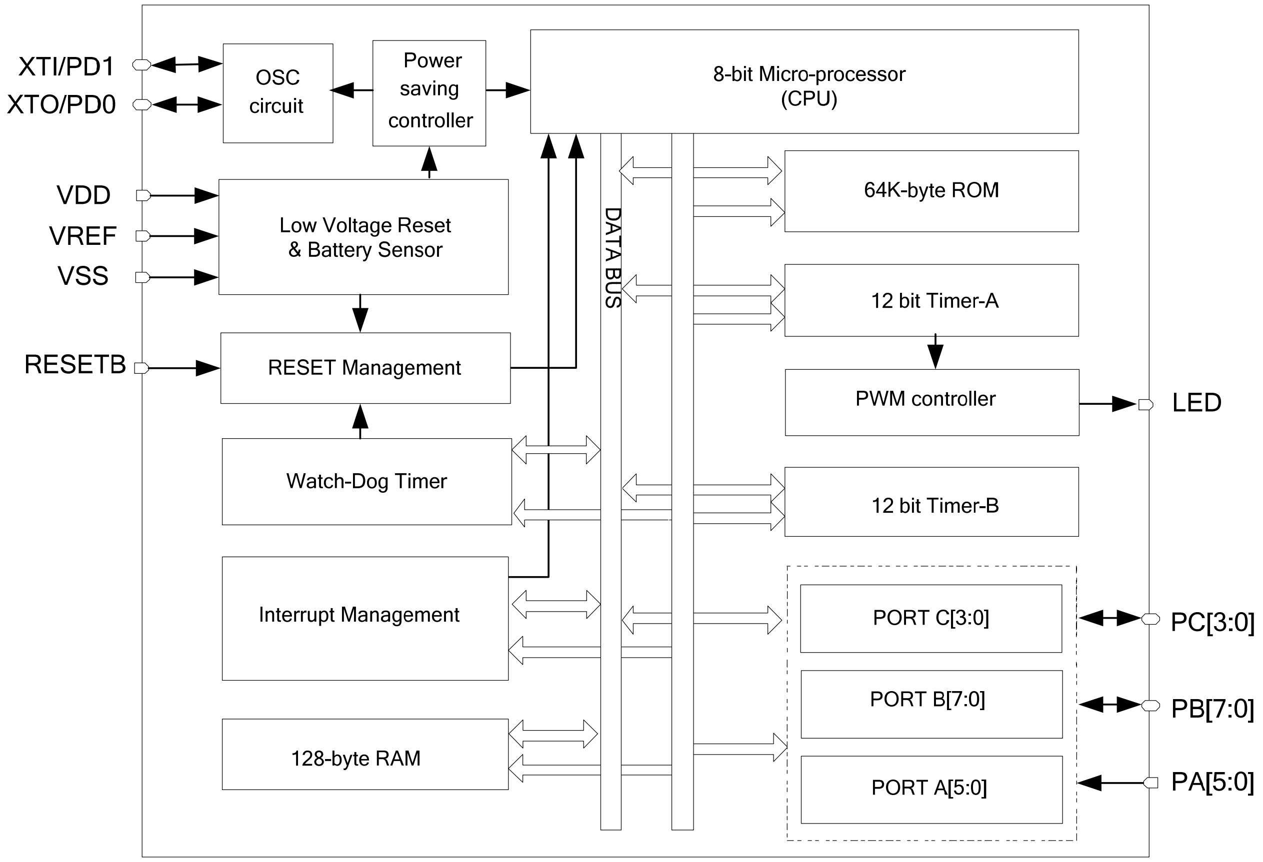
The GPM6C1064A1 is a special chip for remote control with 128 bytes built-in SRAM and 64K bytes built-in ROM. It includes three Timers and up to 20 software selectable general I/Os. Otherwise, it provides one frequency programmable and duty selectable Pulse Width Modulation (PWM) output for remote control it operates over a wide voltage range of 2.0V - 3.6V. It has a clock SLEEP mode for power saving. The power saving mode maintains the RAM contents, but stops the oscillator and causes all other chip functions to be inoperative. The SLEEP mode can be released by using external wakeup sources.

In addition, it provides a FREEZE mode for power savings and key board locking when power-supply voltage is detected lower than V_{LVR}. In FREEZE mode, CPU and peripheral were stopped, and all I/Os maintain floating with input function disabled. The FREEZE mode can not be released by any wakeup or interrupt sources. It is released only when a battery is removed and reinstalled which must be with enough power or external reset occurs. Especially, it has a very accuracy internal OSC, which can match the spec (4MHz±1.5%(typ) @ 2.0V ~3.6V or 8MHz±1.5%(typ) @ 2.0V ~3.6V) and can be used at most application. Meanwhile, the build-in IR transfer module can make IR control and usage easier. Using GPM6C1064A1 does not only share the latest technology, but also enjoy the full commitment and technical support from Generalplus.

2. FEATURES

- **CPU**
  - 151 instructions
  - 13 addressing modes
  - Up to 8MHz clock operation
- **Memories**
  - 64K bytes program memory (ROM)
  - 128 bytes RAM including stack area
- **Reset Management**
  - Enhanced Reset System
  - Power On Reset (POR)
  - Low Voltage Reset (LVR)
  - Watchdog Reset (WDR)
  - External Reset (ERST)
- **Interrupt Management**
  - 6 internal interrupts
- **I/O Ports**
  - Max 14 multifunction bi-directional I/Os
  - 6 Schmitt Trigger pure input I/Os
  - Each incorporate with pull-up resistor, pull-down resistor or floating input, depending on programmer's settings on the corresponding registers
  - I/O ports with LED driving capability
  - 14 I/O ports with 16mA current sink
- **Clock Management**
  - Internal oscillator: 4MHz±1.5%(typ.), @ 2.0V ~3.6V or 8MHz±1.5%(typ.), @ 2.0V ~3.6V
  - Crystal input: 4MHz @ 2.0V ~3.6V or 8MHz @ 2.0V ~3.6V
- **Power Management**
  - Two power saving modes: SLEEP, FREEZE mode
- **Two Analog Peripherals**
  - Battery Sensor for detecting battery connection
  - LVR: Low Voltage Reset (1.85V ± 0.15V)
- **12-bit Timer (Timer A)**
  - Timer mode with clock source selectable
  - PWM output in carrier signal mode with duty and driver current programmable
  - PWM output in no carrier signal mode with driver current programmable
- **12-bit Timer (Timer B)**
  - Timer mode with clock source selectable
  - Overflow signal can be configure as clock source for TimerA
- **Watchdog Timer**
  - Frequency: 0.475Hz @4MHz(System Clock)
  - Frequency: 0.95Hz @8MHz(System Clock)
- **Key wake up**
  - Key change wake-up from SLEEP mode
- **IR**
  - Built-in IR TX can drive IR LED with up to 300mA driving capability @ VDD=3.0V & V_{LED}=3.0V.

3. BLOCK DIAGRAM



4. SIGNAL DESCRIPTIONS

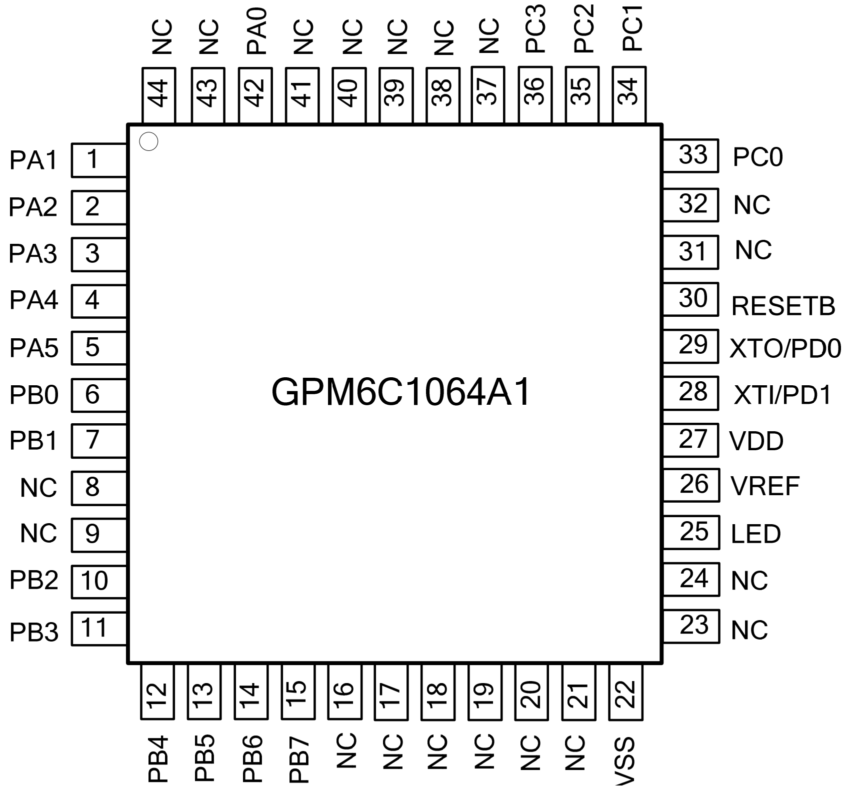
4.1. Pin Description

Type: I = Input, O = Output, S = Supply

Pin Name	Package Pin No.	Dice Pin No.	Type	Main Function	Alternate Function
XTI / PD1	28	15	I/O	**Crystal In**: It is connected with external crystal for a crystal oscillation circuitry in crystal mode. **PortD[1]**: Bi-directional programmable Input/Output port. It can be configured as pull-up (pull-high) resistor, pull-down (pull-low) resistor or floating input, open-drain PMOS output, or CMOS output. The sink current (I_{OL}) of this I/O can reach 18mA (VDD = 3.0V, V_{OL} = 0.2*VDD) enough to drive LED.	
XTO / PD0	29	16	I/O	**Crystal Output**: It is connected with external crystal for a crystal oscillation circuitry in crystal mode. **PortD[0]**: Bi-directional programmable Input/Output port. It can be configured as pull-up resistor, pull-down resistor or floating input, open-drain PMOS output, or CMOS output. The sink current (I_{OL}) of this I/O can reach 18mA (VDD = 3.0V, V_{OL} = 0.2*VDD) enough to drive LED.	
PA5	5	28	I	**PortA[5:0]**: Schmitt Trigger pure input. It can be configured as pull-up resistor, pull-down resistor or floating input. This port is special for key input in IR controller application. (Key Change Wake-up).	
PA4	4	27	I		
PA3	3	26	I		
PA2	2	25	I		
PA1	1	24	I		
PA0	42	23	I		
PB7	15	8	I/O	**PortB[7:0]**: Bi-directional programmable Input/Output port. It can be configured as pull-up res. The sink current (I_{OL}) of this I/O can reach 18mA (VDD = 3.0V, V_{OL} = 0.2*VDD) enough to drive LED.	
PB6	14	7	I/O		
PB5	13	6	I/O		
PB4	12	5	I/O		
PB3	11	4	I/O		
PB2	10	3	I/O		
PB1	7	2	I/O		
PB0	6	1	I/O		
PC3	36	22	I/O	**PortC[3:0]**: Bi-directional programmable Input/Output port. It can be configured as pull-up resistor, pull-down resistor or floating input, open-drain PMOS output, or CMOS output. The sink current (I_{OL}) of this I/O can reach 18mA (VDD = 3.0V, V_{OL} = 0.2*VDD) enough to drive LED.	
PC2	35	21	I/O		
PC1	34	20	I/O		
PC0	33	19	I/O		
LED	25	12	O	IR controller signal transmit pin.	
VDD	27	14	S	Battery supply	
VREF	26	13	S	power supply	
VSS	22	11	S	Ground	
VSS_TX	-	11	S	The Ground for PWM block.	
RESETB	30	17	I	This pin is an active low reset for the chip.	

4.2. PIN Map (Top View)

4.2.1. LQFP44 package



5. FUNCTIONAL DESCRIPTIONS

5.1. Central Processing Unit

5.1.1. CPU Introduction

The CPU inside GPMC1064A1 is a high performance processor equipped with six internal registers: accumulator, program counter, X register, Y register, stack pointer, and processor status register. This CPU is a fully static CMOS design. The oscillation frequency could be varied up to 8.0MHz depending on the application.

5.1.2. CPU register

The CPU has six registers that are the Program Counter (PC), an Accumulator (A), two index registers (X, Y), the Stack Pointer (SP), and the Status register (P). The program counter consists of 16-bit register.

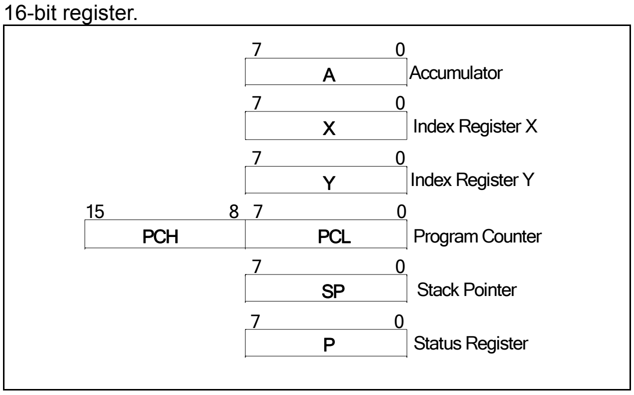


Figure 5-1 System registers

X, Y register

In address mode, X and Y registers can be used as index registers or buffer registers. These register contents are added to the specified address, which becomes the actual address. Some operations such as increment, decrement, comparison and data transfer function can be used in X and Y registers.

Accumulator

The Accumulation is the 8-bit general-purpose register, which can be operated with transfer, temporary saving, condition judgment, etc.

Stack pointer

The CPU has an 8-bit-wide register indicating the location in the stack to be accessed (push or pop) when a subroutine call or interrupt occurs.

When subroutine call is executed or an interrupt occurrence is accepted, the value of stack point is updated automatically.

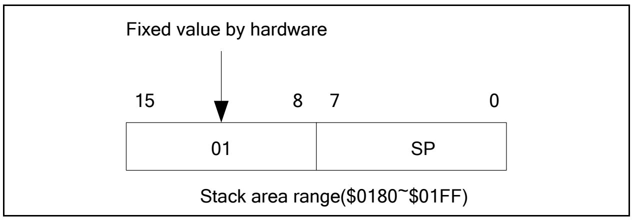


Figure 5-2 Stack point register

[Example] 5-1 Initialized stack point value

```
ldx   #C_STACK_BOTTOM     ; Initial stack pointer at $1FF
txs                       ;Transfer to stack point
```

Program counter (PC)

The program counter is a 16-bit wide register. It consists of two 8-bit registers which registers are PCH and PCL. This register indicates the address of next instruction to be executed. In Reset state, the content of program counter is stored with $FFFC.

Status register (P)

The 8-bit status register contains the interrupt mask and 6 flags representative of the result of the instruction just executed. This register can also be handled by the PHP and PLP instructions. These bits can be individually controlled by specific instructions. The detailed description is shown in following description.

Note: Not all instructions affect status register. A detailed instruction description will be discussed in 6502 instruction manual.

- **Negative flag bit**

This flag indicated the bit7 status of the result of a data or arithmetic operation. Programmer can use this bit to do some operations, e.g. branch condition or bit operation.

- **Overflow flag bit**

This flag indicates whether the overflow has occurred in arithmetic operation. When the result of an addition or subtraction is over +127 or less than –128, this overflow bit is set to ‘1’.

- **Decimal mode flag**

This flag indicates what mode is operated by arithmetic operation. The CPU has two operation modes, binary mode and decimal mode for arithmetic operation. Programmer can use the instruction to alternate them.

- **Interrupt disable flag**

This bit can enable or disable all interrupts except NMI interrupt source. If this bit is set to '1', CPU will ignore interrupt signal. On the contrary, if this bit is set to '0', CPU will accept interrupt signal.

- **Zero flag**

This flag indicates the result of a data or arithmetic operation. If the result is equal to zero, the zero flag is set to '1'. Contrary, this bit is set to '0' by other values.

- **Carry flag**

This bit is set to '1 if the result of addition operation generates a carry, or if the result of subtraction doesn't generate a borrowing. In addition, some shift instructions or rotate instructions also change this bit.

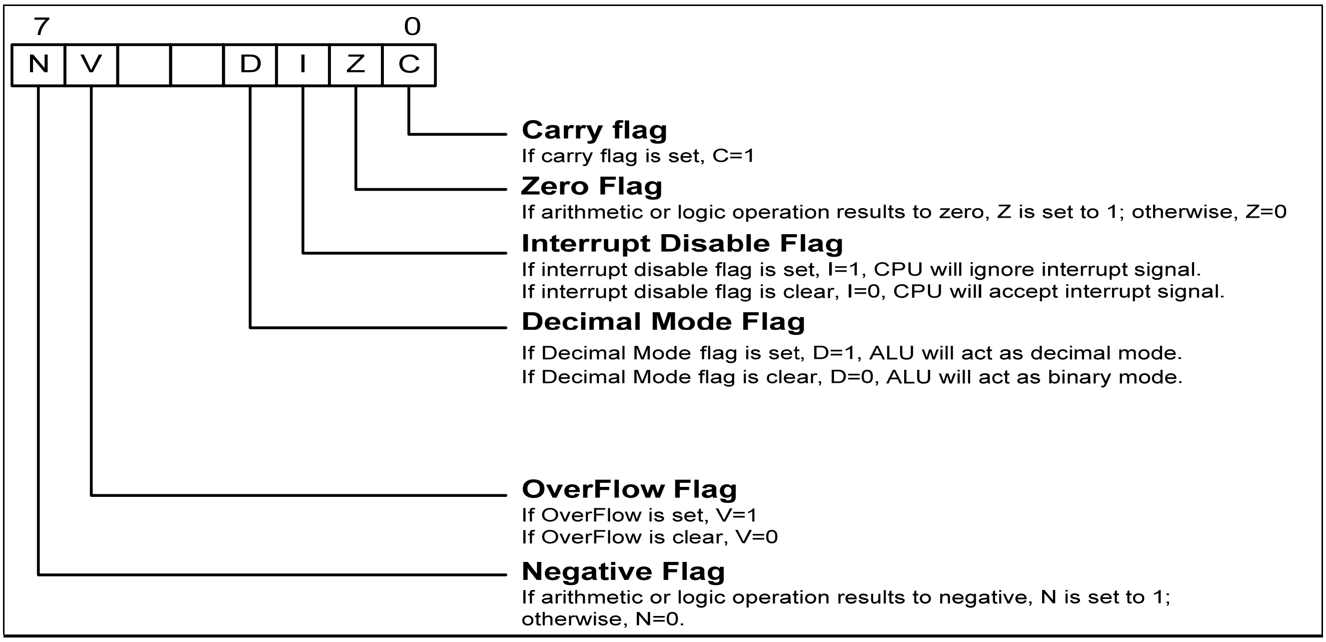


Figure 5-3 Status register

5.2. Memory Organization

5.2.1. Introduction

The GPM6C1064A1 has separated address spaces for program memory and data memory. Program memory can be read only. It contains up to 64K bytes of program memory. Data memory that contains 128 bytes of RAM including stack area can be read and written.

5.2.2. Memory Space

Memory address allocations on the GPM6C1064A1 are divided into several parts. Figure 5-5 shows GPM6C1064A1 memory map.

The first 128 addresses are allocated for special function registers, including function control registers and I/O control registers, which allow programmer to use the first page instruction in setting this register and help for program size reduction.

The total RAM consists of 128 bytes (including Stack) on the locations from $080 through $FF and double mapping to $180 ~ $1FF (see Figure 5-5).

GPM6C1064A1 supports 64K bytes of ROM. The address for ROM is located on $4000 ~ $FFFF (see Figure 5-5).

The address of NMI, RESET and IRQ exception vectors are located from $FFFA to $FFFF. The exception vectors should be specified in the program to have proper operation.

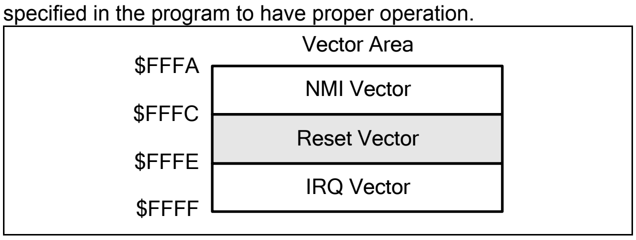


Figure 5-4 Interrupt vector area

[Example] 5-2 Interrupt vector table in software

```
VECTOR:         .SECTION
                DW      V_NMI
                DW      V_Reset
                DW      V_IRQ
```

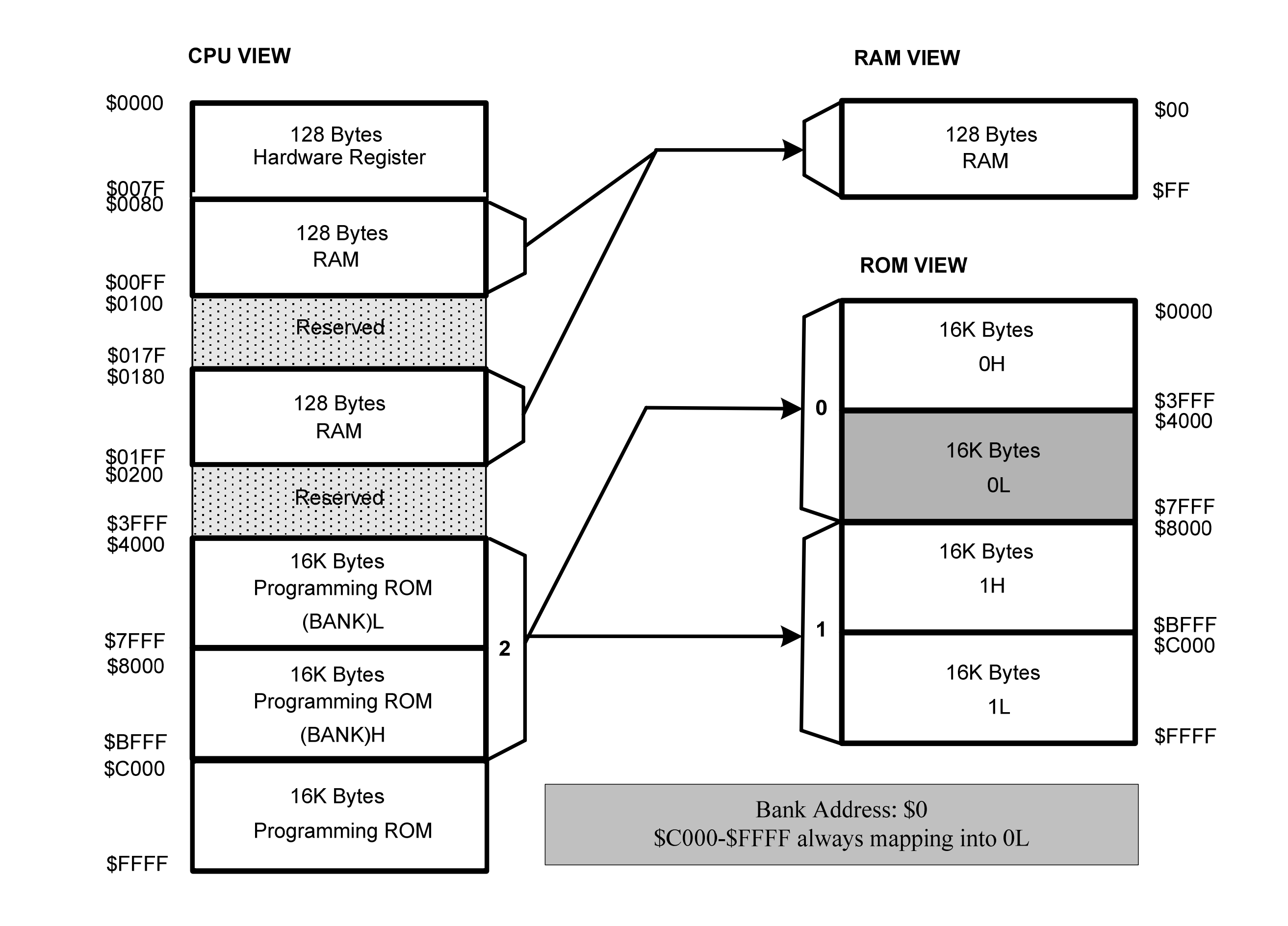


Figure 5-5 Memory map for GPM6C1064A1

5.2.3. Configuration Option Register

The configuration option register is used to setup the operation condition. And its CPU view address is $FFF8. It is map to the special reserved ROM address $7FF8.

The GPM6C1064A1 has the following configuration options.

- It supports crystal resonator or internal oscillator clock source.
- It supports LVR enable or disable option.
- It supports watch dog enable or disable option.

Users can refer to the Device Configuration Register and set it in [Project/ Setting/ Mask Option] of Fortis IDE as Figure 5-6.

Device Configuration Register (OPCODE, $FFF8)

BIT	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	OPTCHK3	OPTCHK2	OPTCHK1	OPTCHK0	-	WDTENB	LVRENB	SYSCLKS
Access	R	R	R	R	-	R	R	R
Default	0	0	0	0	-	0	0	0

Bit [7:4] **OPTCHK [3:0]:** Configuration Option Check bits must fill 1010.

Bit [3] Reserved

Bit [2] **WDTENB:** disable/enable watchdog
0= WDT is enabled
1= WDT is disabled

Bit [1] **LVRENB:** disable/enable LVR
0= LVR is enabled
1= LVR is disabled

Bit [0] **SYSCLKS:** IOSC (internal) / Crystal selection
0= IOSC
1= Crystal

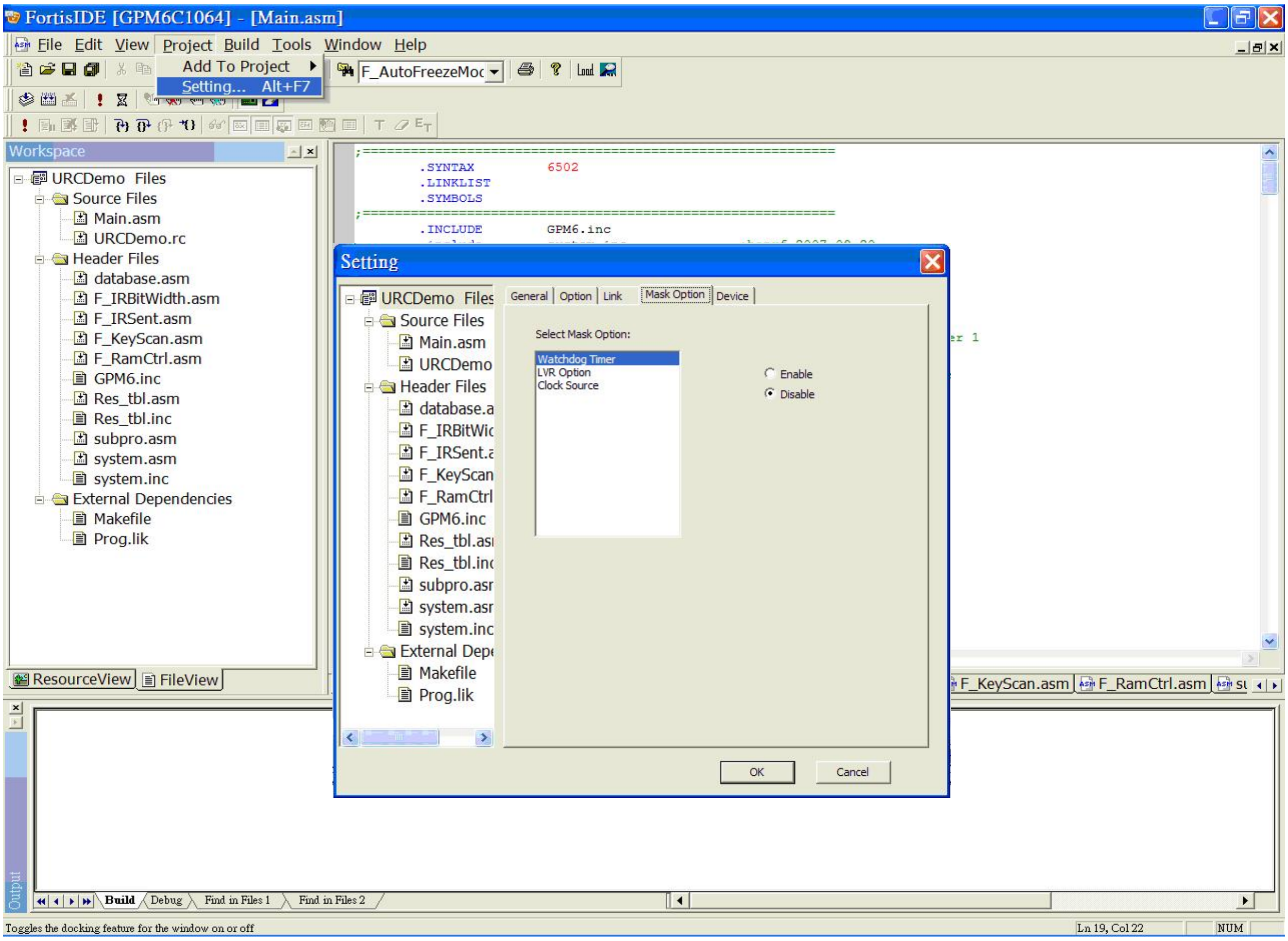


Figure 5-6 Device Configuration Register set in Fortis IDE

5.2.4. Special function Registers

GPM6C1064A1 device have up to twenty three control registers. All of the control registers are used by MCU and peripheral function block for controlling the desired operation of the device. Some of the control registers contain control and statue bits for peripheral module such as Timer unit, Interrupt control unit, etc. Note that the reserved addresses are not implemented on the chip.

Some of bits in control register are read only. When writing to them, there are not any effects on the corresponding bits. The following table shows the summary of the control registers. The detailed information of each control registers are explained in each peripheral section.

$0000~$000A: I/O port

Address	Register	Reset Value	R/W	Bit7	Bit6$	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$00	P_IOB_DIR	00h	R/W	Port B Direction control							
$01	P_IOC_DIR	00h	R/W	-	-	-	-	Port C Direction control			
$02	P_IOD_DIR	00h	R/W	-	-	-	-	-	-	Port D Direction control	
$03	P_IOA_PULL	00h	R/W	-	-	Port A input control					
$04	P_IOB_ATT	00h	R/W	Port B attribute register							
$05	P_IOC_ ATT	00h	R/W	-	-	-	-	Port B attribute register			

$0000~$000A: I/O port

Address	Register	Reset Value	R/W	Bit7	Bit6$	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$06	P_IOD_ ATT	00h	R/W	-	-	-	-	-	-	Port D attribute register	
$07	P_IOA_DAT	00h	R/W	-	-	Write data into the Port A data register and read data from the I/O pad.					
$08	P_IOB_DAT	00h	R/W	Write data into the Port B data register and read data from the I/O pad.							
$09	P_IOC_DAT	00h	R/W	-	-	-	-	Write data into the Port C data register and read data from the I/O pad.			
$0A	P_IOD_DAT	00h	R/W	-	-	-	-	-	-	Write data into the Port D data register and read data from the I/O pad.	

$0010~$0014：INT Flag & other special register

Address	Register	Reset Value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$10	P_BANK_CTRL	00h	R/W	-	-	-	-	-	-	-	BANK0
	P_PWM_DRV	00h	W	-	-	-	PWMDRV1	PWMDRV0	-	-	-
$12	P_SYS_SLEEP	00h	W	C_SYS_SLEEP= AAH Write other data system reset							
$13	P_INT_CTRL	00h	R/W	-	TMAOIE	-	TMBOIE	F1KIE	F4KIE	F32KIE	F2MIE
$14	P_INT_FLAG	00h	R/W	-	TMAOIF	-	TMBOIF	F1KIF	F4KIF	F32KIF	F2MIF

$0020~0026: Timer control

Address	Register	Reset Value	R/W	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
$20	P_WDT_CTRL	00h	W	C_WDT_CLR= AAH Write other data system reset							
$21	P_TMA_CTRL	00h	R/W	TMAES	-	TMACLK1	TMACLK0	TMADUT1	TMADUT0	-	TMAMOD0
$22	P_TMB_CTRL	00h	R/W	TMBES	-	TMBCLK1	TMBCLK0	-	-	-	-
$23	P_TMA_CNTL	00h	R	TMA Counter Low 8 Bits Data							
	P_TMA_PWML		W	TMA PWM Low 8 Bits Register							
$24	P_TMA_CNTH	00h	R	-	-	-	-	TMA Counter High 4 Bits Data			
	P_TMA_PWMH		W	-	-	-	-	TMA PWM High 4 Bits Register			
$25	P_TMB_CNTL	00h	R	TMB Counter Low 8 Bits Data							
	P_TMB_REGL		W	TMB Low 8 Bits Register							
$26	P_TMB_CNTH	00h	R	-	-	-	-	TMB Counter High 4 Bits Data			
	P_TMB_REGH		W	-	-	-	-	TMB High 4 Bits Register			

Note: If the bits of the register is not defined, it will not be implemented in the real chip, its readout value should be random. However, it is defined as 0 in this table for simplification.

5.2.5. BANK Control

The GPM6C1064A1 support max 2 banks: each bank has 32K address space. Bank control register can control bank switch. It is show as below.

Bank control register (P_BANK_CTRL, $0010)

BIT	Bit7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Name	-	-	-	-	-	-	-	BANK0
Access	-	-	-	-	-	-	-	R/W
Default	-	-	-	-	-	-	-	0

Bit [7:1] Reverse

Bit [0] **BANK0:** Bank number set.

0 = Bank number 0 (C_BANK_00)

1 = Bank number 1 (C_BANK_01)

5.3. Clock Source

The GPM6C1064A1 supports Crystal / Ceramic or Internal oscillator, as shown in the following diagram, Figure 5-7. They can be selected by device configuration option at address ($FFF8.0) and be set in Fortis IDE like as Figure 5-6. The detailed configuration option setting of device is given in section 5.2.3 configuration option register.

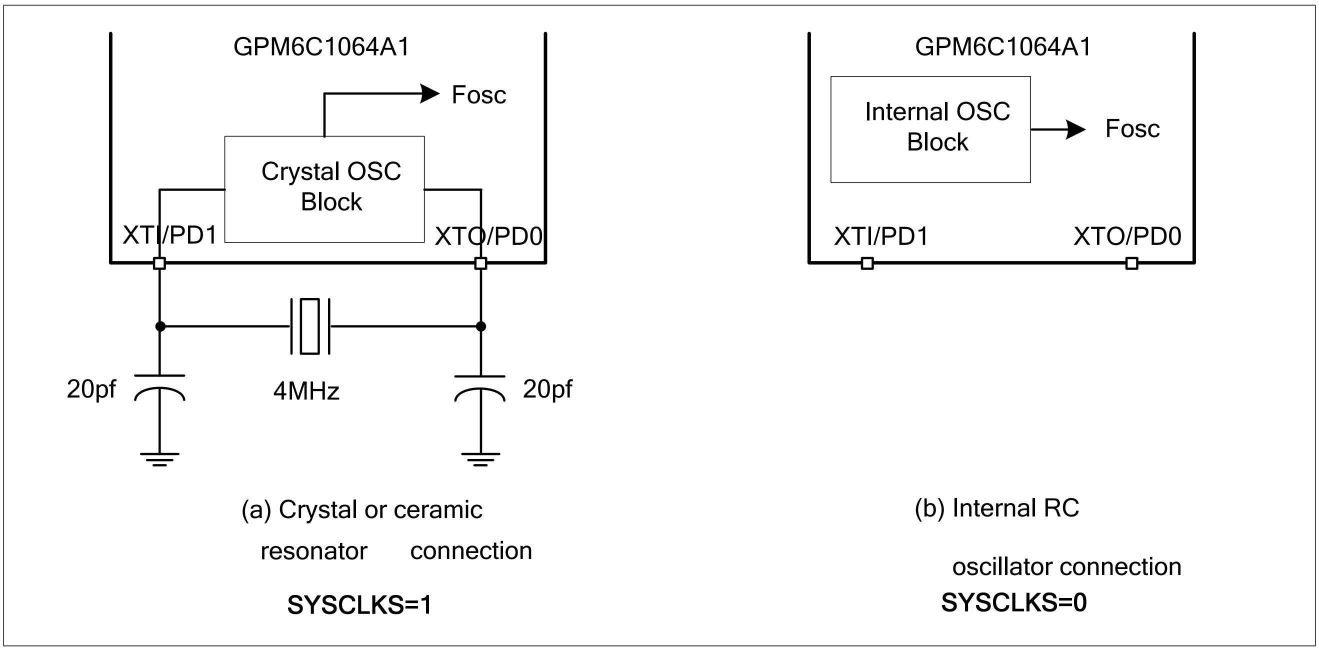


Figure 5-7 Two types of clock sources

5.4. Power Saving Mode

5.4.1. Introduction

To reduce the current consumption when system does not need to be active, SLEEP mode and FREEZE mode can be utilized. These two modes are able to reduce power consumption to save power. They also feature different wakeup time. Programmer must write corresponding value to SLEEP Control Register to enter SLEEP mode. And system will enter FREEZE mode automatically when power down. For more information about SLEEP and FREEZE modes, please see figure 5-8 and they will be depicted in the next two sections.

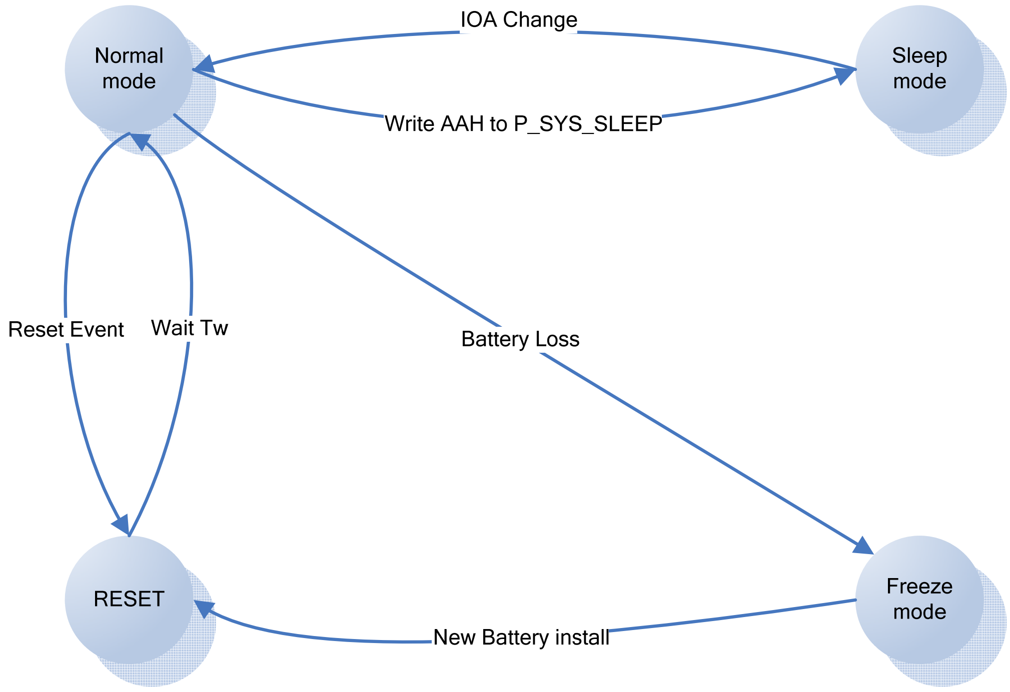


Figure 5-8 Power saving mode operation

5.4.2. SLEEP Mode

The SLEEP mode function will disable all system clocks, including the clock generation circuit. Once the system enters the SLEEP mode, only the activated IOA change wakeup events (from I/Os) can recover the normal operation from SLEEP mode.

In such a mode, LVR function is disabled, RAM and I/Os will remain in their previous states until being awakened. The system will be waked up by any change on port A. After the GPM6C1064 is awakened, the internal CPU will remain on previous State until Tw ≧ 65536 x T1 (Tw = waiting time & T1 = system clock cycle); and then continue processing the program. (See Figure 5-9).

$$T1 = 1 / (F_{CPU}),\ Tw \geqq 65536 \times T1$$

To enter SLEEP mode, programmer must write #C_SYS_SLEEP ($AA) to SLEEP control register (P_SYS_SLEEP).

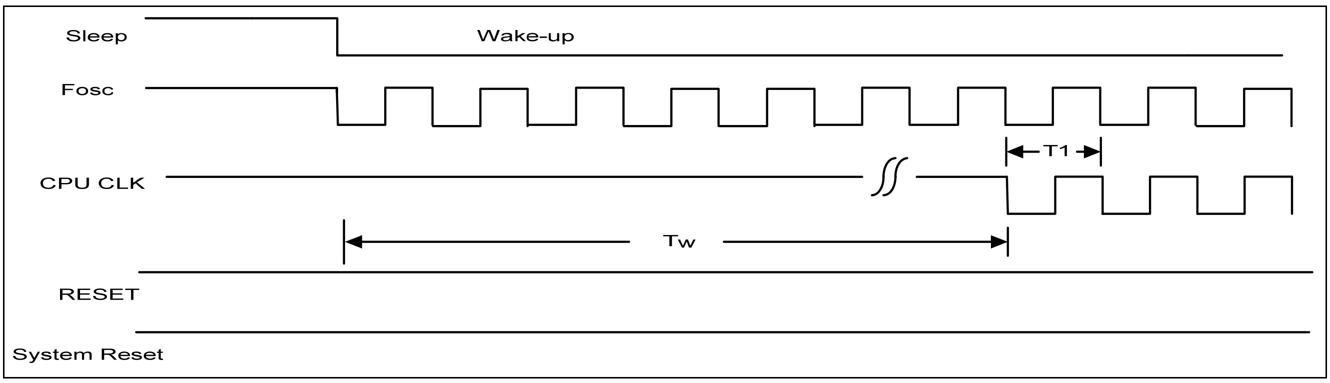


Figure 5-9 SLEEP mode

5.4.3. FREEZE mode

If the power-supply voltage drops down (See Figure 5-10), system will go into FREEZE mode. Low Voltage Reset (LVR) will reset all functions into the initial operational (stable) state. In FREEZE mode, system clock and CPU is stopped; RAM remains on its previous states; all I/Os are floating with input function disabled; LVR function is disabled. The FREEZE mode would not release by any external interrupts unless the battery is reinstalled which voltage is higher then V_{LVR}. The system watch dog action is not occurred in FREEZE mode. Two methods of freeze can be selected by mask code. Mode 1 needs to bond VDD VREF together, which can save one pin connect. Mode 2 needs to bond VDD VREF separately, which has longer sram hold time.

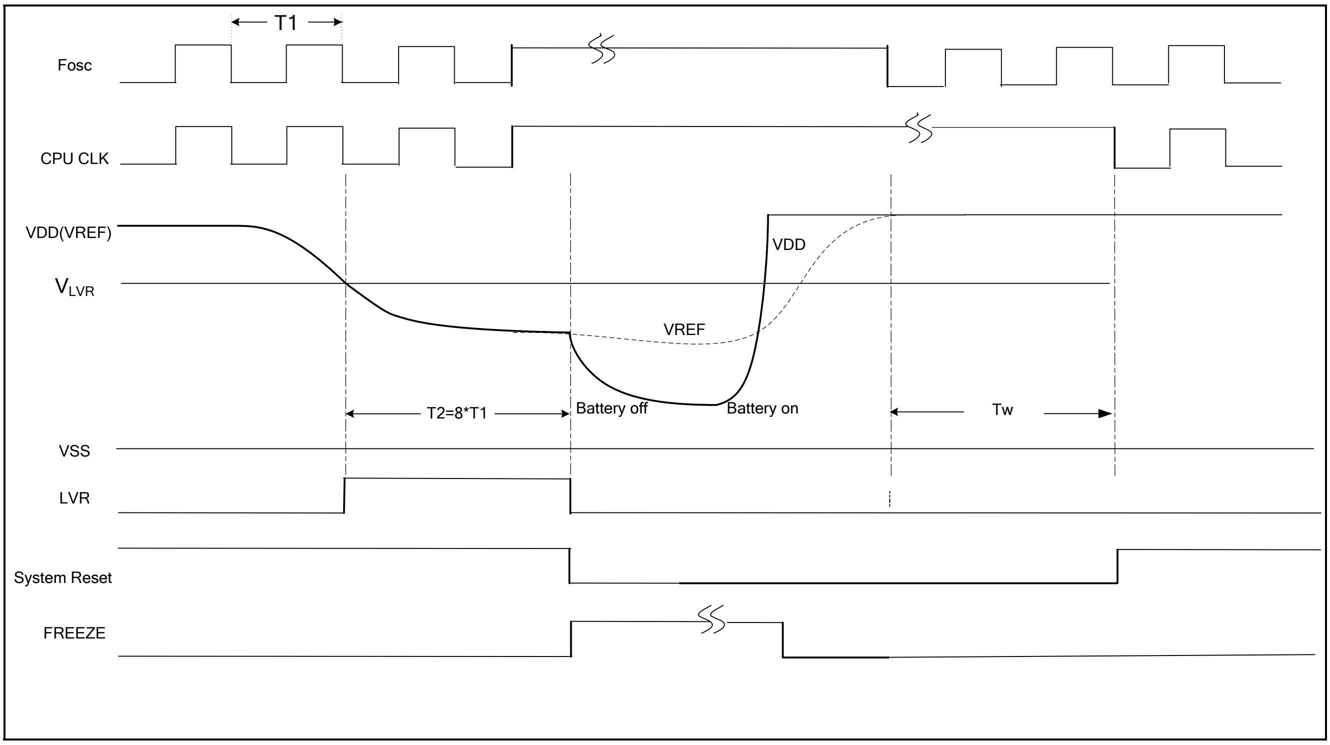


Figure 5-10 FREEZE mode

SLEEP Control Register (P_SYS_SLEEP, $0012)

BIT	**Bit7**	**Bit6**	**Bit5**	**Bit4**	**Bit3**	**Bit2**	**Bit1**	**Bit0**
NAME	SLEEPCTRL7	SLEEPCTRL6	SLEEPCTRL5	SLEEPCTRL4	SLEEPCTRL3	SLEEPCTRL2	SLEEPCTRL1	SLEEPCTRL0
ACCESS	W	W	W	W	W	W	W	W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **SLEEPCTRL** [7:0]: Operation mode control register

$AA = write to enter SLEEP mode (C_SYS_SLEEP)

Other data = reset system

[Example] 5-3 Let MCU enter SLEEP mode

```
    lda     P_IOA_DAT                       ; latch PortA
    lda     #C_SYS_SLEEP                    ; SLEEP command $AA
    sta     P_SYS_ SLEEP                    ; goto sleep mode
```

5.5. Interrupt

5.5.1. Introduction

The GPM6C1064A1 provides six types of interrupt sources with the same normal interrupt level. The six types of interrupt sources are timer A overflow interrupt, timer B overflow interrupt, time Fosc/1024 interrupt, time Fosc/4096 interrupt, time Fosc/32768 interrupt, time Fosc/2097152 interrupt.

These interrupts have individual status (occurred or not) and control (enable or not) registers. In general, once an interrupt event occurs, the corresponding flag bit will be set. If the related interrupt control bit is set to enable interrupt, an interrupt request signal will be generated and then CPU executes service routine. If the related interrupt control bit is disabled, programmer still can observe the corresponding flag bit, but no interrupt request signal will be generated. The interrupt flag bits must be cleared in the interrupt service routine to prevent program from deadlock in interrupt service routine. With any instruction, interrupts pending during the previous instruction is served.

Before entering interrupt service routine, the system saves the current PC address into bottom of the stack such as address $1FF and $1FE in Figure 5-11. And abstract the interrupt service routine first address from $FFF6 and $FFF7. In a corresponding way, the system abstract the return PC address from the bottom of the stack when finished the interrupt service (See Figure 5-12). These interrupt sources are listed as [Table] 5-1 and will be described in corresponding section.

[Table] 5-1 Interrupt source list

Source	Interrupt flag register	Interrupt control register	Source	Interrupt flag register	Interrupt control register
Timer A overflow	TMAOIF($0014.6)	TMAOIE($0013.6)	Time Fosc/4096	F4KIF($0014.2)	F4KIE($0013.2)
Timer B overflow	TMBOIF($0014.4)	TMBOIE($0013.4)	Time Fosc/32768	F32KIF($0014.1)	F32KIE($0013.1)
Time Fosc/1024	F1KIF($0014.3)	F1KIE($0013.3)	Time Fosc/2097152	F2MIF($0014.0)	F2MIE($0013.0)

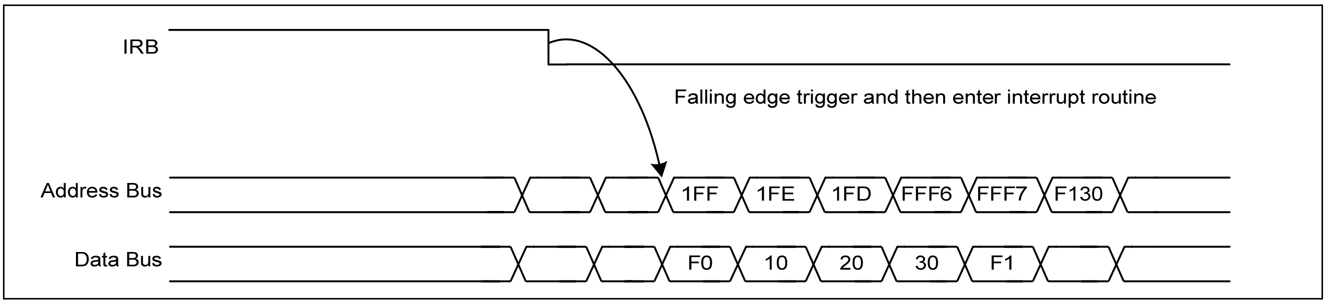


Figure 5-11 Interrupt triggered by IRB

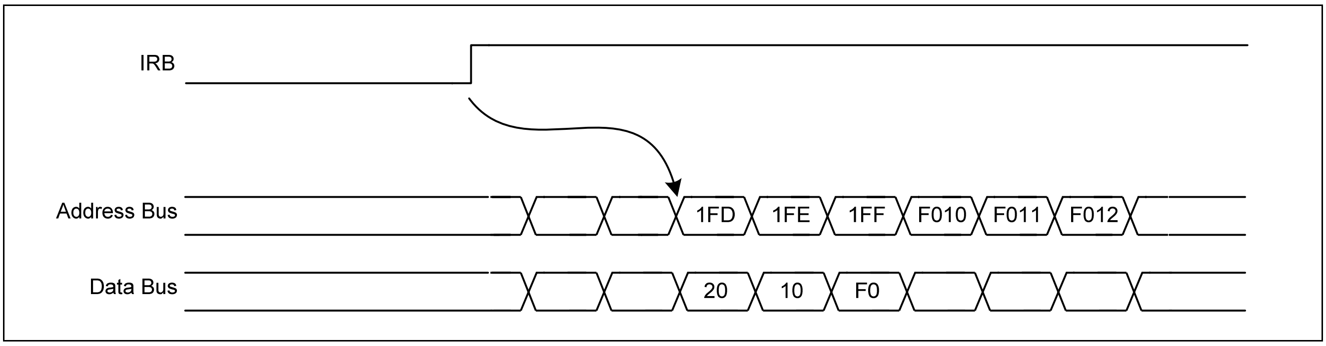


Figure 5-12 Leave interrupt routine

5.5.2. Interrupt register

Interrupt Control Register (P_INT_CTRL, $0013)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	TMAOIE	-	TMBOIE	F1KIE	F4KIE	F32KIE	F2MIE
ACCESS	-	R/W	-	R/W	R/W	R/W	R/W	R/W
DEFAULT	-	0	-	0	0	0	0	0

Bit [7] Reserved

Bit [6] **TMAOIE**: Timer A overflow interrupt enable bit
0 = interrupt disable
1 = interrupt enable (C_INT_TMAOIE)

Bit [5] Reserved

Bit [4] **TMBOIE**: Timer B overflow interrupt enable bit
0 = interrupt disable
1 = interrupt enable (C_INT_TMBOIE)

Bit [3] **F1KIE**: Time Fosc/1024 interrupt enable bit
0 = interrupt disable
1 = interrupt enable (C_INT_F1KIE)

Bit [2] **F4KIE**: Time Fosc/4096 interrupt enable bit
0 = interrupt disable
1 = interrupt enable (C_INT_F4KIE)

Bit [1] **F32KIE**: Time Fosc/32768 interrupt enable bit
0 = interrupt disable
1 = interrupt enable (C_INT_F32KIE)

Bit [0] **F2MIE**: Time Fosc/2097152 interrupt enable bit
0 = interrupt disable
1 = interrupt enable (C_INT_F2MIE)

Interrupt Flag Register (P_INT_FLAG, $0014)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	TMAOIF	-	TMBOIF	F1KIF	F4KIF	F32KIF	F2MIF
ACCESS	-	R/W	-	R/W	R/W	R/W	R/W	R/W
DEFAULT	-	0	-	0	0	0	0	0

This flag is cleared by writing the corresponding bit by “1”.

Bit [7] Reserved

Bit [6] **TMAOIF**: Timer A overflow interrupt flag
0 = no event
1 = event has occurred (C_INT_TMAOIF).

Bit [5] Reserved

Bit [4] **TMBOIF**: Timer B overflow interrupt flag
0 = no event
1 = event has occurred (C_INT_TMBOIF).

Bit [3] **F1KIF**: Time Fosc/1024 interrupt flag
0 = no event
1 = event has occurred (C_INT_F1KIF).

Bit [2] **F4KIF**: Time Fosc/4096 interrupt flag
0 = no event
1 = event has occurred (C_INT_F4KIF).

Bit [1] **F32KIF**: Time Fosc/32768 interrupt flag
0 = no event
1 = event has occurred (C_INT_F32KIF).

Bit [0] **F2MIF**: Time Fosc/2097152 interrupt flag
0 = no event
1 = event has occurred (C_INT_F2MIF).

[Example] 5-4 Enable Timer A overflow interrupt

```
;=========================;
; main loop routine
;=========================;
lda     #C_INT_TMAOIE
sta      P_INT_CTRL                         ; enable Timer A overflow INT
cli                                         ; enable INT
;=========================;
;IRQ interrupt service routine
;=========================;
lda     #C_INT_TMAOIF
sta     P_INT_FLAG                          ; clear INT request flag
sta     P_INT_CTRL                          ; enable Timer A overflow INT
```

5.6. Reset Sources

5.6.1. Introduction

There are four types of reset sources for the system, Power-On Reset (POR), External Reset (ERST), Low Voltage Reset (LVR), Watchdog Timer Reset (WDR). These reset sources can be concluded as external events and internal events. The external events come from power line or external trigger event. The internal events come from the program run away. Figure 5-13 shows the affected region for each reset source.

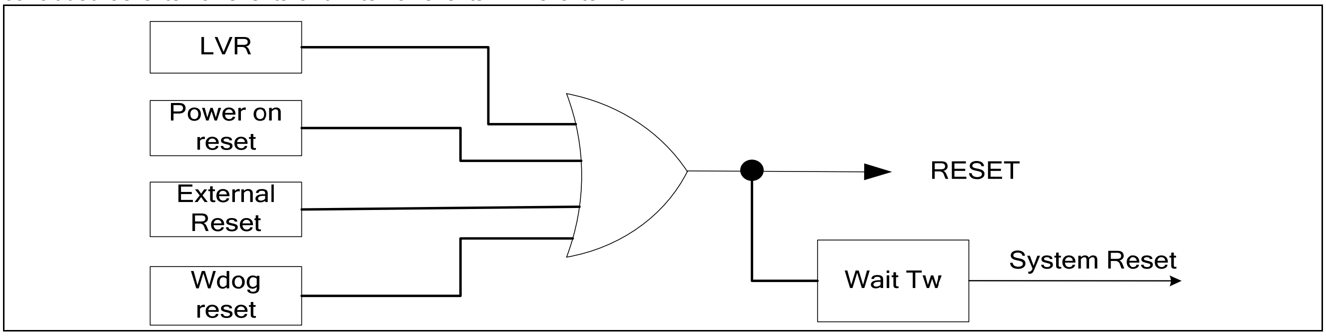


Figure 5-13 Reset sources

5.6.2. Power-On Reset (POR)

A POR is generated when VDD is rising from 0v. When VDD rises to an acceptable level (~1.45V), the power on reset circuit will start a power-on sequence. After that, the system will operate in target speed and start to activate.

5.6.3. External Reset (ERST)

The GPM6C1064A1 provides an external pin to force the system returning to the initial status. The RESETB pin is used to connect a capacitor to ground, shown in Figure 5-14 Reset Circuit. This pin is a low active signal. When the RESETB pin falls below 0.3 x VDD, the system will be forced to enter reset state.

The external reset pulse width must be larger than 1000ns at least. Any pulse shorter than 1000ns will be filtered and taken no effect on the system. If a reset pulse that is long enough to take effect, the reset will be extended to 16ms.

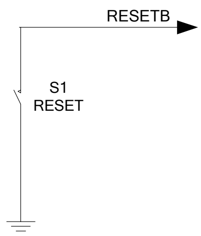


Figure 5-14 Reset circuit

5.6.4. Low Voltage Reset (LVR)

The on-chip Low Voltage Reset (LVR) circuitry forces the system entering FREEZE mode when the MCU voltage falls below the specific LVR trigger voltage. This function prevents MCU from working at an invalid operating voltage range.

A device configuration option at address ($FFF8.1, can be set in Fortis IDE as Figure 5-6) is used to enable or disable this function. If this function is enabled, the LVR circuit will monitor power level while chip is operating. If the power is lower than the specific level for a specific period, the system reset will enter FREEZE mode and all I/Os will be locked.

5.6.5. Watchdog Timer Reset (WDR)

On-chip watchdog circuitry makes the device entering reset when MCU goes into an unknown state and has no watchdog cleared information. This function prevents the MCU to be stuck in an abnormal condition. The WDT can be disabled or enabled through configuration option address ($FFF8.2, can be set in Fortis IDE as Figure 5-6). The internal reset of WDT will be generated by a time-out event of the WDT automatically when watchdog is enabled.

These reset signals will reset the CPU and restart the program. To avoid a WDT time-out reset, programmer has to write # C_WDT_CLR (=$AA) to P_WDT_CTRL periodically. If a reset signal is generated, it will also clear the WDT counter and restart the WDT.

Watchdog Control Register (P_WDT_CTRL, $0020)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	WDTCTRL7	WDTCTRL6	WDTCTRL5	WDTCTRL4	WDTCTRL3	WDTCTRL2	WDTCTRL1	WDTCTRL0
ACCESS	W	W	W	W	W	W	W	W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **WDTCTRL [7:0]**: Operation mode control register
$AA = write to clear watch dog counter (C_WDT_CLR)
Other data = reset system

[Example] 5-5 Clear watch dog counter

```
lda     # C_WDT_CLR                     ; Clear watch dog command $AA
sta     P_WDT_ CTRL
```

5.7. I/O PORTS

5.7.1. Introduction

The GPM6C1064A1 has four ports, Port A, Port B, Port C and Port D. These port pins may be multiplexed with an alternate function for the peripheral features on the device. In general, when an initial reset state occurs, all ports are used as a general purpose input port. There are two parts, data and pull registers, in Port A's and three parts, data, direction and attribution registers, in the others' IO structures. Each corresponding bit in these ports should be given a value.

The setting rules are as follows:

- The direction setting determines whether this pin is an input or an output.
- The data register is used to read the value on the port, which can be different when programmer sets the port to input pull-high/ pull- low.
- The pull register for IOA setting affects if the pure input pin with or without a pull resistor.

Please refer to the [Table] 5-2 for IOA's and [Table] 5-3 for IOB, IOC IOD's setting.

[Table] 5-2 I/O configurations (for IOA)

Pull (P_IOA_PULL)	Data (P_IOA_DAT)	Function	Description
0	0	Floating	Input with float
0	1	Floating	Input with float
1	0	Pull Low	Input with pull-low
1	1	Pull High	Input with pull-high

[Table] 5-3 I/O configurations (for IOB, IOC and IOD)

Attribution (P_IOX_ATT)	Direction (P_IOX_DIR)	Data (P_IOX_DAT)	Function	Description
0	0	0	Floating	Input with float
0	0	1	Pull low	Input with pull-low
0	1	0	Driving low	Output Data
0	1	1	Driving High	Output Data
1	0	0	Floating	Input with float
1	0	1	Pull high	Input with pull-high
1	1	0	Driving High	Output Data
1	1	1	Driving low	Output Data

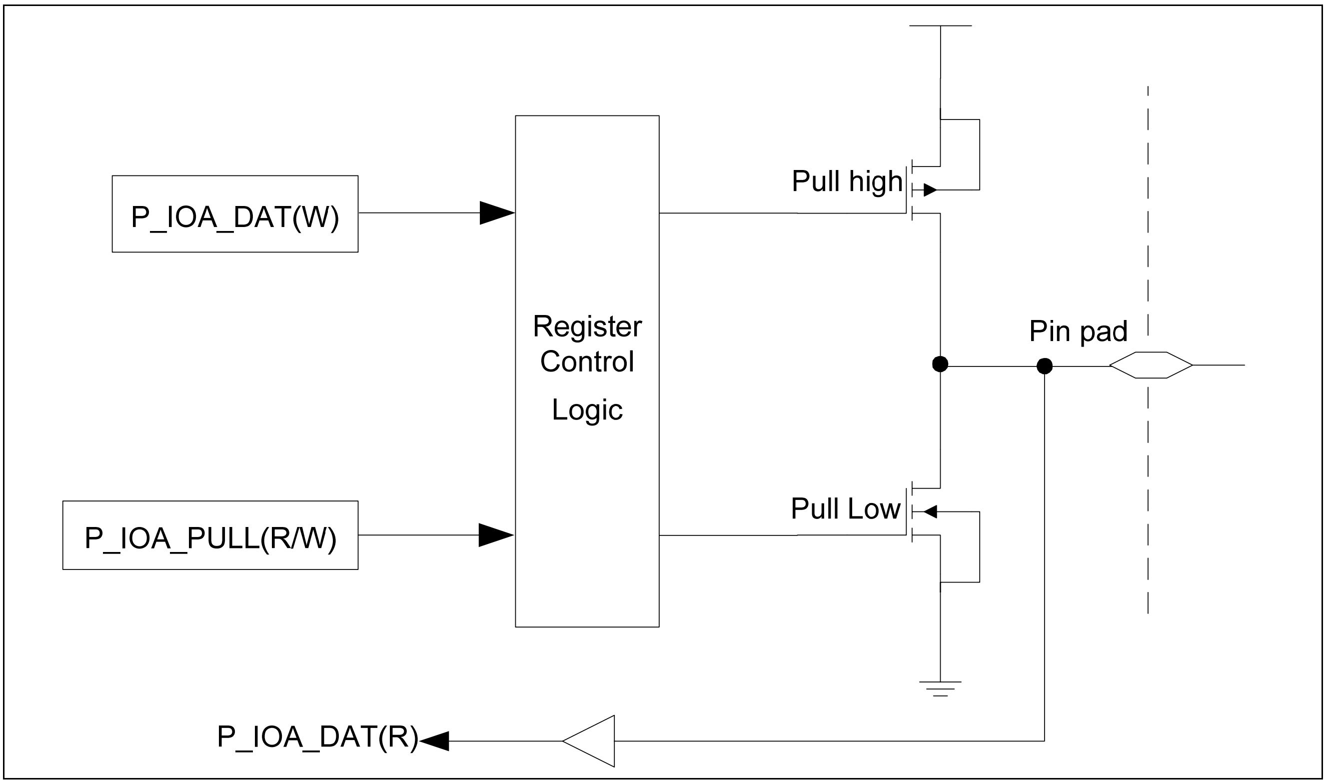


Figure 5-15 Block diagram of I/O port (IOA)

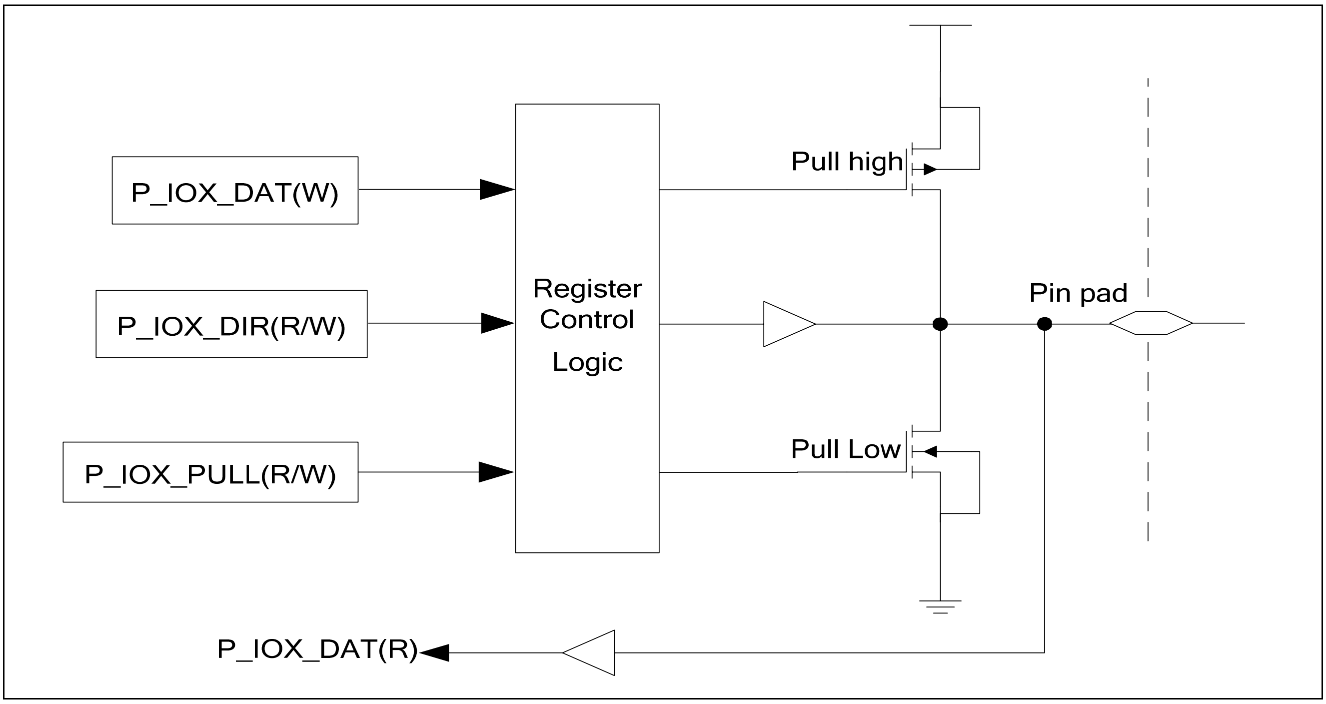


Figure 5-16 Block diagram of I/O port (IOB, IOC and IOD)

5.7.2. Port A

Port A is a 6-bit pure input I/O port. The I/O Port A has 6 programmable I/Os that are controlled by data register P_IOA_DAT, and pull control register P_IOA_PULL. P_IOA_PULL is used to disable or enable pull resistor. P_IOA_DAT is used to control the input pin with pull low or pull high resistor. To read the real IO value, programmer has to read P_IOA_DAT.

Port A Pull Register (P_IOA_PULL $0003)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	P_IOA_PULL					
ACCESS	-	-	R/W					
DEFAULT	-	-	00h					

Bit [7:6] Reserved

Bit [5:0] **P_IOA_PULL**: Writing to disable or enable pull resistor.

0 = disable pull resistor

1 = enable pull resistor

Port A Data Register (P_IOA_DAT, $0007)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	P_IOA_DAT					
ACCESS	-	-	R/W					
DEFAULT	-	-	00h					

Bit [7:6] Reserved

Bit [5:0] **P_IOA_DAT**: Read to get Port A value. Writing to configure input with pull low or pull high resistor.

Read to get Port A value

Write to configure input with pull low or pull high resistor if the corresponding P_IOA_PULL register bit is set as “1”.

0= input with pull low resistor

1= input with pull high resistor

[Example] 5-6 Set Port A [5:0] as input with pull low resistor.

```
lda   #$3F          ; store accumulator with $3F
sta   P_IOA_PULL    ; Enable pull resistor
lda   #$00          ; store accumulator with $00
sta   P_IOA_DAT     ; set IOA as input with pull low resistor
```

[Example] 5-7 Set Port A [5:0] as Input with pull high resistor.

```
lda   #$3F          ; store accumulator with $3F
sta   P_IOA_PULL    ; Enable pull resistor
sta   P_IOA_DAT     ; set IOA as input with pull high resistor
```

5.7.3. Port B

Port B is an 8-bit bi-directional I/O port. The I/O Port B has 8 programmable I/Os, controlled by direction control register P_IOB_DIR, and attribution register P_IOB_ATT. Reading P_IOB_DAT will get the real IO value.

Port B Direction Register (P_IOB_DIR, $0000)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_DIR							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_DIR**: Port B direction register.

0 = input

1 = output

Port B Attribution Register (P_IOB_ATT, $0004)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_ATT							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_ATT**: Port B attribution register

Port B Data Register (P_IOB_ DAT, $0008)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	P_IOB_ DAT							
ACCESS	R/W							
DEFAULT	00h							

Bit [7:0] **P_IOB_ DAT**: Port B Data value.
Read to get Port B value
Write to configure input with pull low or pull high resistor

[Example] 5-8 Set Port B [3:0] as output with low data and Port B [7:4] as input with pulling high.

```
lda     #$0F            ; store accumulator with $0F
sta     P_IOB_DIR       ; set direction
lda     #$00            ; store accumulator with $00
sta     P_IOB_ATT       ; set attribute
lda     #$F0            ; store accumulator with $F0
sta     P_IOB_DAT       ; set Port Data
```

[Example] 5-9 Set Port B [7:0] as input with float.

```
lda     #$00            ; store accumulator with $00
sta     P_IOB_ATT       ; set direction
sta     P_IOB_DIR       ; set attribute
sta     P_IOB_DAT       ; set Port Data
```

5.7.4. Port C

Port C is a 4-bit bi-directional I/O port. The I/O Port C has 4 programmable I/Os, controlled by direction control register P_IOC_DIR, and attribution register P_IOC_ATT. Reading P_IOC_DAT will get the real IO value.

Port C Direction Register (P_IOC_DIR, $0001)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	P_IOC_DIR			
ACCESS	-	-	-	-	R/W			
DEFAULT	00h							

Bit [7:4] Reserved
Bit [3:0] **P_IOC_DIR**: Port C direction register.
0 = input
1 = output

Port C Attribution Register (P_IOC_ATT, $0005)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	P_IOC_ATT			
ACCESS	-	-	-	-	R/W			
DEFAULT	00h							

Bit [7:4] Reserved

Bit [3:0] **P_IOC_ATT**: Port C attribution register

Port C Data Register (P_IOC_ DAT, $0009)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	P_IOC_ DAT			
ACCESS	-	-	-	-	R/W			
DEFAULT	00h							

Bit [7:4] Reserved

Bit [3:0] **P_IOC_ DAT**: Port C Data value.

Read to get Port C value

Write to configure input with pull low or pull high resistor

[Example] 5-10 Set Port C [1:0] as output with high data and Port C [3:2] as input with pulling high.

```
lda     #$03            ; store accumulator with $03
sta     P_IOC_DIR       ; set direction
lda     #$0F            ; store accumulator with $0F
sta     P_IOC_ATT       ; set attribute
lda     #$0B            ; store accumulator with $0B
sta     P_IOC_DAT       ; set Port Data
```

[Example] 5-11 Set Port C [3:0] as input with pulling low.

```
lda     #$00            ; store accumulator with $00
sta     P_IOC_DIR       ; set direction
sta     P_IOC_ATT       ; set attribute
lda     #$0F            ; store accumulator with $0F
sta     P_IOC_DAT       ; set Port Data
```

5.7.5. Port D

Port D is a 2-bit bi-directional I/O port. The I/O Port D has 2 programmable I/Os, controlled by direction control register P_IOD_DIR, and attribution register P_IOD_ATT. Reading P_IOD_DAT will get the real IO value. In addition, Port D is multiplexed with various special functions. After reset, the default setting for port D is used as general I/O ports.

[Table] 5-4 Port D function list

Port D Pin	BIT	Shared function
PD0	Bit0	Crystal output (XTO)
PD1	Bit1	Crystal input (XTI)

Port D Direction Register (P_IOD_DIR, $0002)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	-	P_IOD_DIR	
ACCESS	-	-	-	-	-	-	R/W	
DEFAULT	00h							

Bit [7:2] Reserved

Bit [1:0] **P_IOD_DIR**: Port D direction register.

0 = input

1 = output

Port D Attribution Register (P_IOD_ATT, $0006)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	-	P_IOD_ATT	
ACCESS	-	-	-	-	-	-	R/W	
DEFAULT	00h							

Bit [7:2] Reserved

Bit [1:0] **P_IOD_ATT**: Port D attribution register

Port D Data Register (P_IOD_ DAT, $000A)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	-	-	P_IOD_ DAT	
ACCESS	-	-	-	-	-	-	R/W	
DEFAULT	00h							

Bit [7:2] Reserved

Bit [1:0] **P_IOD_ DAT**: Port D Data value.

Read to get Port D value

Write to configure input with pull low or pull high resistor

[Example] 5-12 Set Port D[1:0] as output with low data.

```
lda   #$03          ; store accumulator with $03
sta   P_IOD_DIR     ; set direction
lda   #$00          ; store accumulator with $00
sta   P_IOD_ATT     ; set attribute
sta   P_IOD_DAT     ; set port data
```

5.8. Timer Module

5.8.1. Introduction

GPM6C1064A1 is equipped with 2 of Timers. Both Timer A and Timer B are 12-bit timers. They are up-count timers. Timer A contains one powerful PWM function and is controlled by corresponding control registers. This function can be easily configured. Each timer's function summary is shown as [Table] 5-5.

[Table] 5-5 Summary of timer function for GPM6C1064A1

	Timer Counter	PWM with carrier signal	Envelop PWM
Timer A	YES	YES	YES
Timer B	YES	None	None

5.9. Timer A

Timer A is special for generating carrier signal in IR control application. The 12-bit timer is an up counter with input clock selectable (Fosc/1, Fosc/4, Fosc/16), which can be configured the corresponding bits of the control register (P_TMA_CTRL [5:4]). The timer A provides with two PWM mode, and the PWM signal is send to IR TX (LED) pin. The driver current of these two kinds of PWM are programmable by configuring TX PWM driving current control source register (P_PWM_DRV [4:3]).

The Timer A module has the following features:

Readable and writable

Clock source selectable

Interrupt-on-overflow from #$FFF to #$000

It supports PWM with carrier signal mode

It supports PWM without carrier signal mode

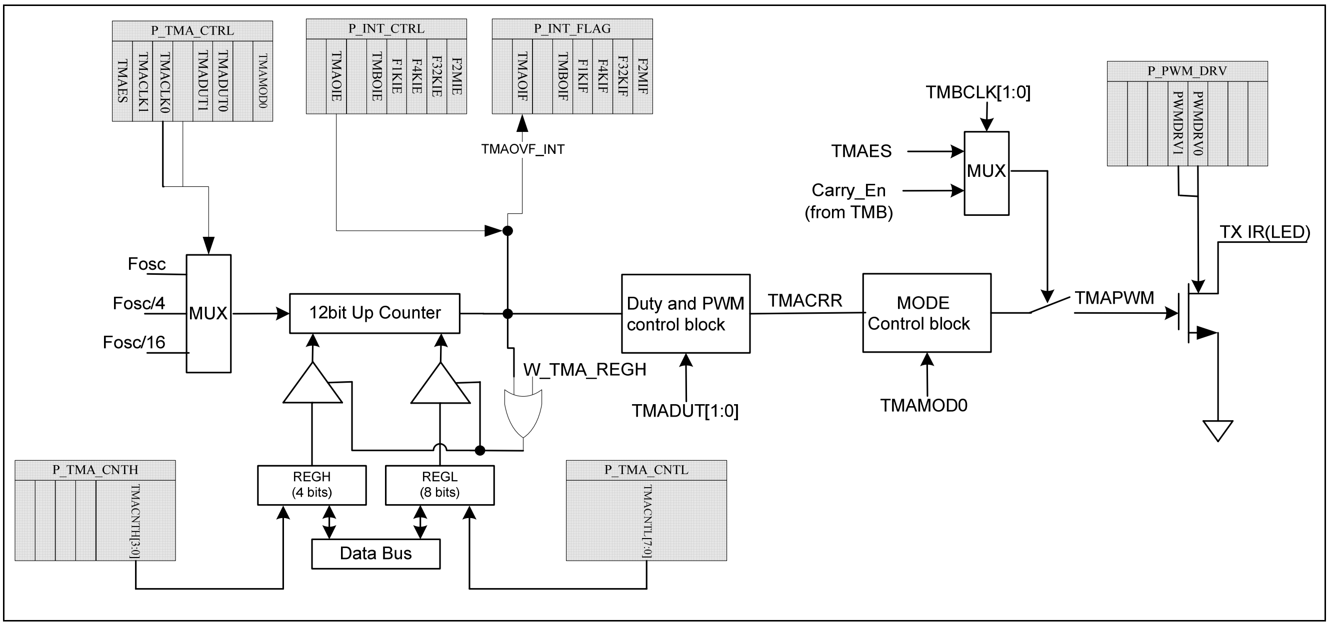


Figure 5-17 Timer A block diagram

5.9.1. PWM with carrier signal mode

Timer A can be configured as PWM mode for generating carrier signal. In PWM with carrier signal mode, the 12-bit timer is an up counter with input clock selectable (Fosc/1, Fosc/4, Fosc/16). When the timer A is started, the value of 4-bit high-byte (low-nibble) register and 8-bit low-byte register would firstly be loaded into the 12-bit counter and then the counter starts count up from the loaded value. If an overflow occurs, the value of high-byte (low-nibble) register (P_TMA_CNTH) and low-byte register (P_TMA_CNTL) would be reloaded into the counter automatically and the counter starts count up again. So the carrier signal with frequency programmable can be generated by this PWM mode via configuring these two registers. Also users can select PWM duty cycle (1/3, 1/4, 1/5, 1/6) via configuring the corresponding bits of the control register (P_TMA_CTRL[3:2]). The carrier signal's enabled or disabled bit can be controlled by two methods depended on which clock source is selected by timer B. If timer B is selected one of the first three clock source (Fosc, Fosc/4 or Fosc/64) by P_TMB_CTRL [5:4] (TMBCLK [1:0]), the timer A's carrier signal on/ off is controlled by timer A's enable/ disable control bit (TMAES) directly. But if the timer B is selected the forth clock source (timer A carrier signal), the timer A's carrier signal is off or on only when timer B overflow events occur one by one.

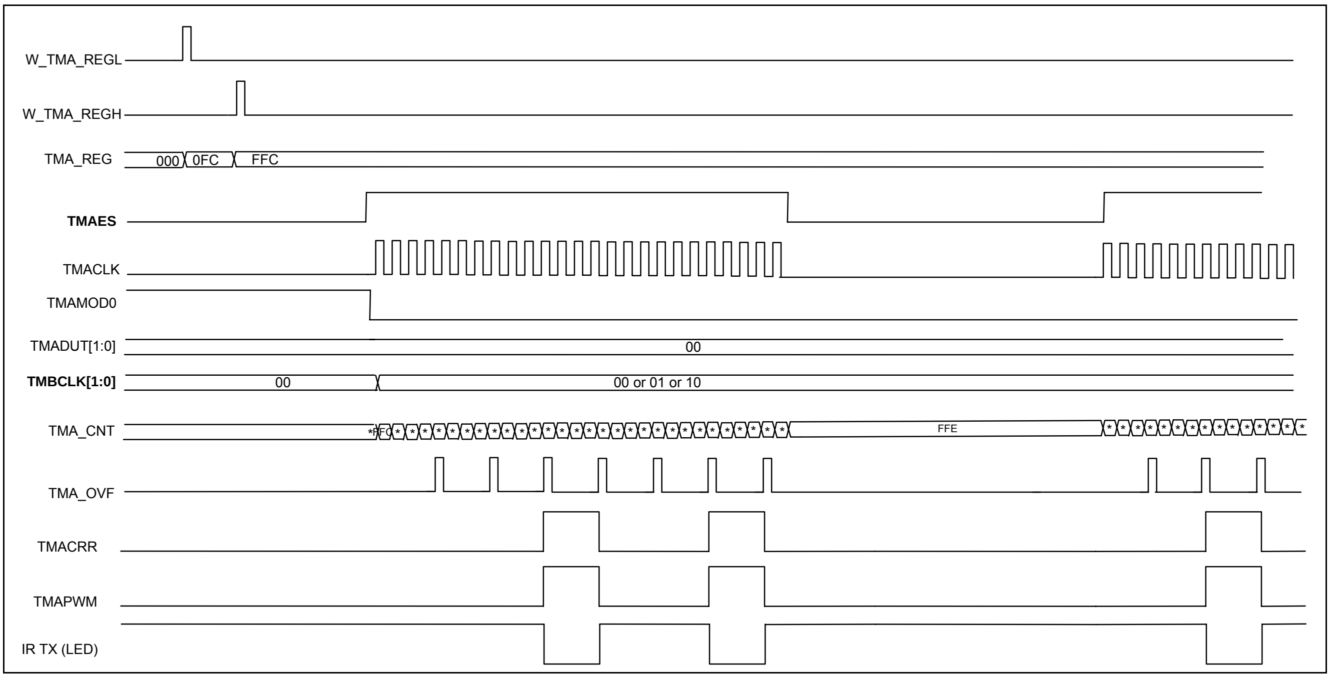


Figure 5-18 The Waveform of PWM with carrier signal mode (1/3 duty, on/off control by TMAES)

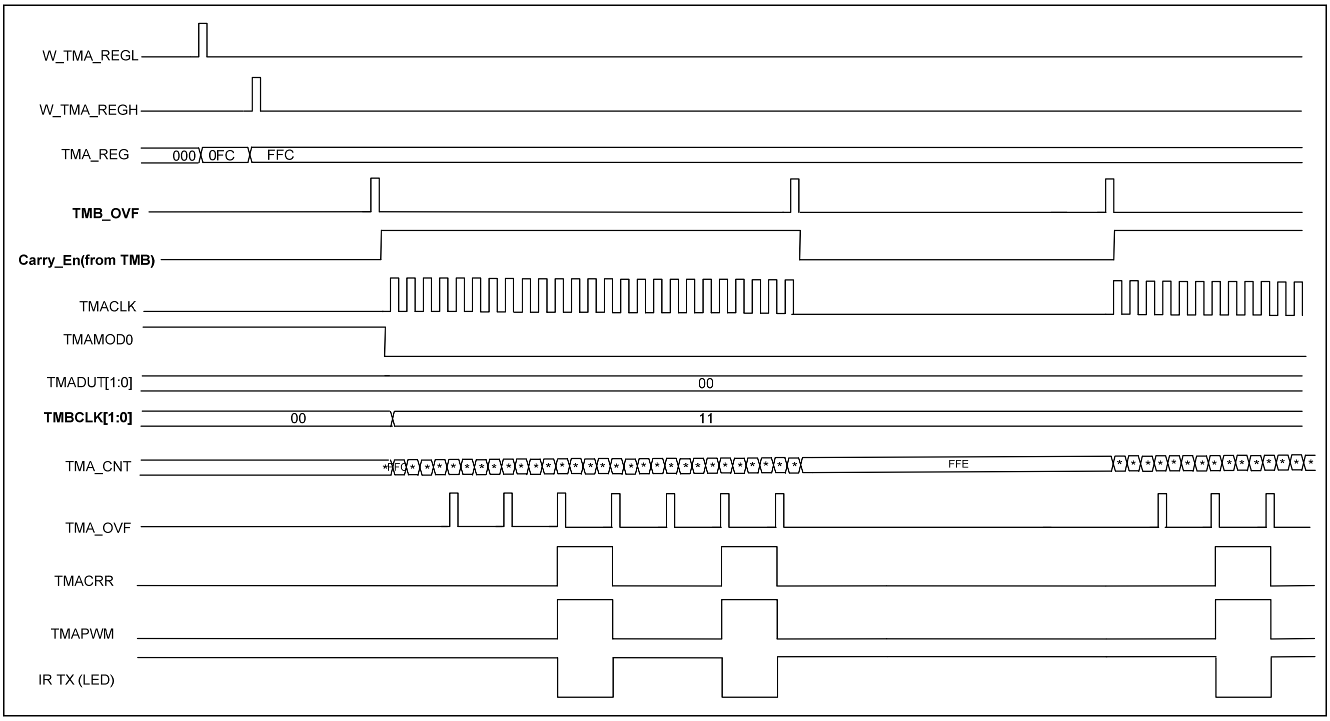


Figure 5-19 The Waveform of PWM with carrier signal mode (1/3 duty, on/off control by timer B overflow events)

5.9.2. PWM without carrier signal mode

PWM without carrier signal mode is used to generate envelop PWM signal without carrier signal. In this mode, IR TX (LED) pin is just output high or low signal and is controlled by timer A's enable or disable control bit or timer B's overflow events in turn. The same as PWM with carrier signal mode, the 12-bit timer is an up counter with input clock selectable (Fosc/1, Fosc/4, Fosc/16). When the timer A is started, the value of high-byte (low-nibble) Register and low-byte Register would firstly be loaded into the 12-bit counter and then the counter starts count up from the loaded value. If an overflow occurs, the value of high-byte (low-nibble) register and low-byte register would be reloaded into the counter automatically and the counter starts count up again. The internal carrier signal is generated but does not be sent to IR TX pin.

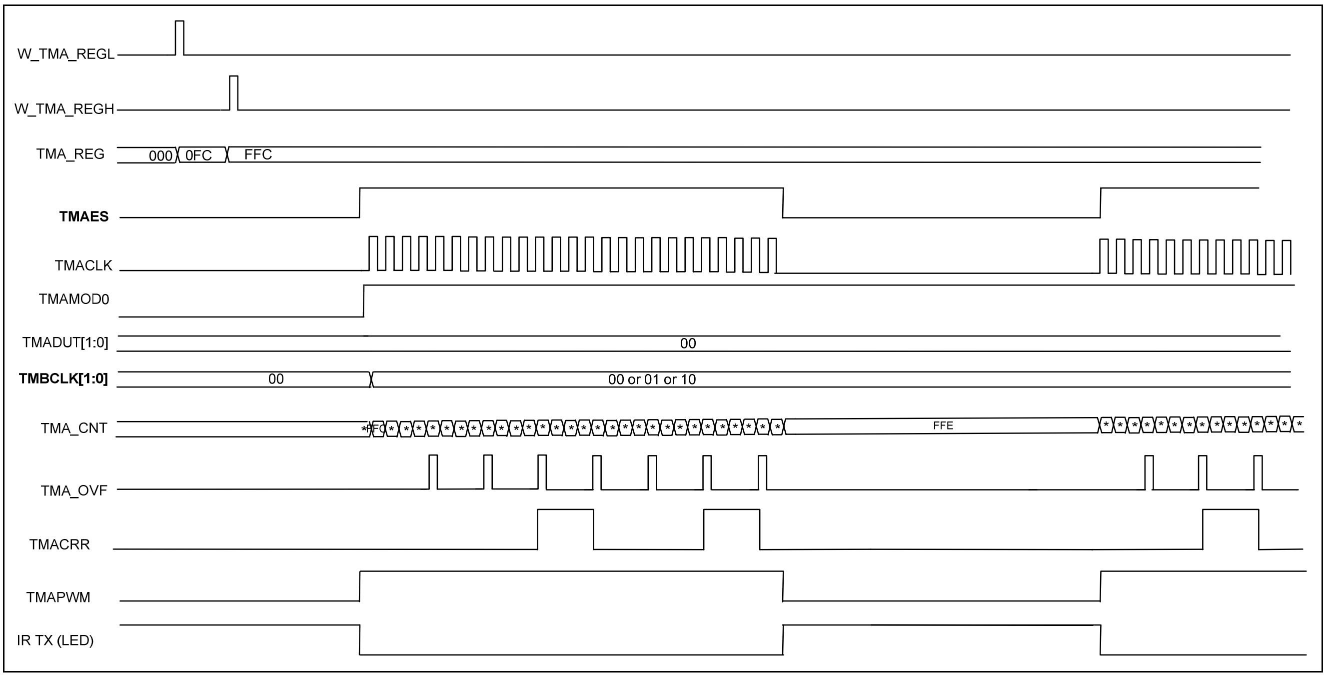


Figure 5-20 The Waveform of PWM without carrier signal mode (on/off control by TMAES)

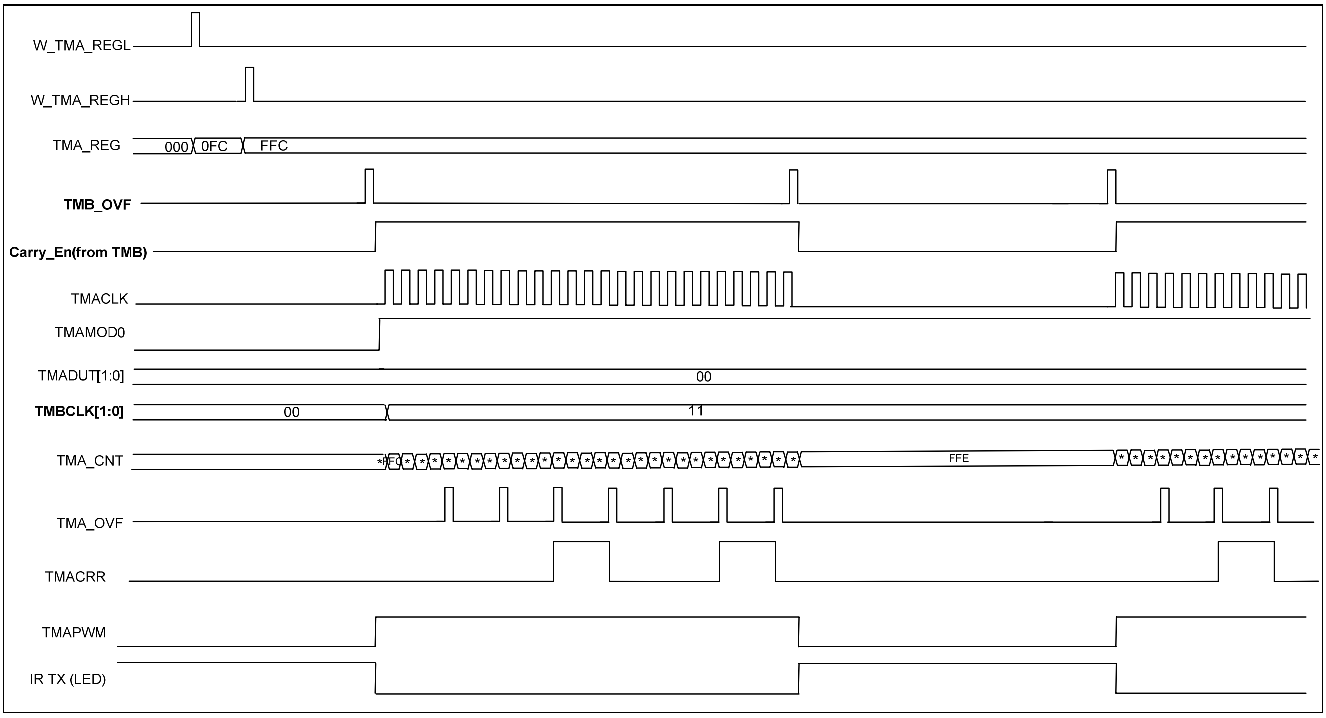


Figure 5-21 The Waveform of PWM without carrier signal mode (on/off control by timer B overflow events)

Timer A Control Register (P_TMA_CTRL, $0021)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	TMAES	-	TMACLK1	TMACLK0	TMADUT1	TMADUT0	-	TMAMOD0
ACCESS	R/W	-	R/W	R/W	R/W	R/W	-	R/W
DEFAULT	0	-	0	0	0	0	-	0

Bit [7] **TMAES**: Timer A enable/disable control.
0 = disable (C_TMAES_DIS)
1 = enable. (C_TMAES_EN)

Bit [6] Reserved

Bit [5:4] **TMACLK**[1：0]: Timer A clock source select bits
11 = Fosc/16 (C_TMACLK _16)
10 = Fosc/4 (C_TMACLK _4)
01 = Fosc (C_TMACLK _1)
00 = Fosc

Bit [5:4] **TMADUT**[1：0]: PWM duty selected bits
11 = 1/6 duty (C_PWMDUT_6)
10 = 1/5 duty (C_PWMDUT_5)
01 = 1/4 duty (C_PWMDUT_4)
00 = 1/3 duty (C_PWMDUT_3)

Bit [1] Reserved

Bit [0] **TMAMOD0**: Timer A work mode configuration bit.
0 = PWM with carrier signal mode (C_TMAMOD_WTC)
1 = PWM without carrier signal mode (C_TMAMOD_WOC)

Timer A low 8-bit data Register (P_TMA_CNTL, $0023)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	TMACNTL7	TMACNTL6	TMACNTL5	TMACNTL4	TMACNTL3	TMACNTL2	TMACNTL1	TMACNTL0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **TMACNTL**[7：0]: Timer A low 8-bit pre-value for the counter.

Timer A high 4-bit data Register (P_TMA_CNTH, $0024)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	TMACNTH3	TMACNTH2	TMACNTH1	TMACNTH0
ACCESS	-	-	-	-	R/W	R/W	R/W	R/W
DEFAULT	-	-	-	-	0	0	0	0

Bit [3:0] **TMACNTH**[3：0]: Timer A high 4-bit pre-value for the counter.

[Example] 5-13 Set Timer A as PWM with carrier signal mode.

```
lda   #$FC                                        ; Before starting timer, set Timer A counter initial value first
sta   P_TMA_CNTL                                  ; set low 8-bit pre-value
lda   #$0F
sta   P_TMA_CNTH                                  ;set high 4-bit pre-value
lda   #C_TMAES_EN + #C_TMACLK _4 +#C_TMADUT_3 + #C_TMAMOD_WTC
sta   P_TMA_CTRL                                  ;Set clock source Fosc/4, 1/3duty, PWM with carrier signal mode
```

5.9.3. PWM Carrier Signal Algorithm

The frequency of PWM carrier signal (**Fpwm**) generated by Timer A depends on three factors.

1. The initial value (**Vreg)** filled into high-byte (low-nibble) register (**TMACNTH [3：0]**) and low-byte register (**TMACNTL [7：0]**)
2. The frequency of timer A clock source (**Ftimer**)
3. The duty of the carrier signal (**DUT**).

➢ If clock source F_{timer} (=F_{osc}/1) is selected as timer clock, then user can calculate the register value (V_{reg}= { **TMACNTH [3：0]**, **TMACNTL [7：0]**}) flow below equation.

$$V_{reg} = 4097 - \frac{F_{timer}}{F_{PWM}} \times DUT$$

For example, if user needs to generate 38 KHz 1/3 duty PWM carrier frequency and TIMER clock source is 4MHz/1 (system clock is 4MHz).

Condition: $\mathbf{F_{PWM}}$ = 38 KHz, $\mathbf{F_{OSC}}$=4MHz, **DUT**=1/3

$\mathbf{V_{reg}}$ = 4097 – (4M/38K)*1/3 = 4062 =FDEH

Then the result FDEH can be written into the PWM high/low register, and the 38 KHz PWM signal is generated.

- If not base system clock F_{timer} (**F_{OSC}/4** or **F_{OSC}/16**) is selected as timer clock, then user can calculate the register value (**V_{reg}**) flow below equation.

$$V_{reg} = 4096 - \frac{F_{timer}}{F_{PWM}} \times DUT$$

For example, if user need to generate 38 KHz 1/3 duty PWM carrier frequency, and system frequency is 4MHz. And **F_{OSC}/4** is selected as timer clock.

Condition: $\mathbf{F_{PWM}}$ = 38 KHz, $\mathbf{F_{timer}}$=4MHz/4, **DUT**=1/3

$\mathbf{V_{reg}}$ = 4096 – (1M/38K)*1/3 = 4087 =FF7H

Then the result FF7H can be written into the PWM high/low register, and the 38 KHz PWM signal is generated.

[Example] 5-14 Set Timer A as PWM with carrier signal mode and the carrier frequency is 38 KHz with 1/3 duty (clock source=Fosc/1).

```
lda   #$DE                                        ; Before starting timer, set Timer A counter initial value first
sta   P_TMA_CNTL                                  ; set low 8-bit pre-value
lda   #$0F
sta   P_TMA_CNTH                                  ;set high 4-bit pre-value
lda   #C_TMAES_EN + #C_TMACLK _1 +#C_TMADUT_3 + #C_TMAMOD_WTC
sta   P_TMA_CTRL                                  ;Set clock source Fosc/1, 1/3duty, PWM with carrier signal mode
```

[Example] 5-15 Set Timer A as PWM with carrier signal mode and the carrier frequency is 38 KHz with 1/3 duty (clock source=Fosc/4).

```
lda   #$F7                                        ; Before starting timer, set Timer A counter initial value first
sta   P_TMA_CNTL                                  ; set low 8-bit pre-value
lda   #$0F
sta   P_TMA_CNTH                                  ;set high 4-bit pre-value
lda   #C_TMAES_EN + #C_TMACLK _4 +#C_TMADUT_3 + #C_TMAMOD_WTC
sta   P_TMA_CTRL                                  ;Set clock source Fosc/4, 1/3duty, PWM with carrier signal mode
```

5.10. Timer B

The Timer B is special for envelope signal generation in IR controller application. The 12-bit timer is an up counter with input clock selectable (Fosc/1, Fosc/4, Fosc/64, TMACAR) via configuring the corresponding bits of the control register by P_TMB_CTRL [5:4] (TMBCLK [1:0]). And the value of low-byte register (P_TMB_CNTL) and high-byte (low-nibble) register (P_TMB_CNTH) would be reloaded into the 12-bit up counter and an interrupt (TMBOIF) would be generated whenever an overflow occurs. The interrupt frequency can be freely selected by selecting different clock source and configuring the low-byte register and high-byte (low-nibble) register with different values. The Timer B module has the following features:

Readable and writable

Clock source selectable

Interrupt-on-overflow from #$FFF to #$000

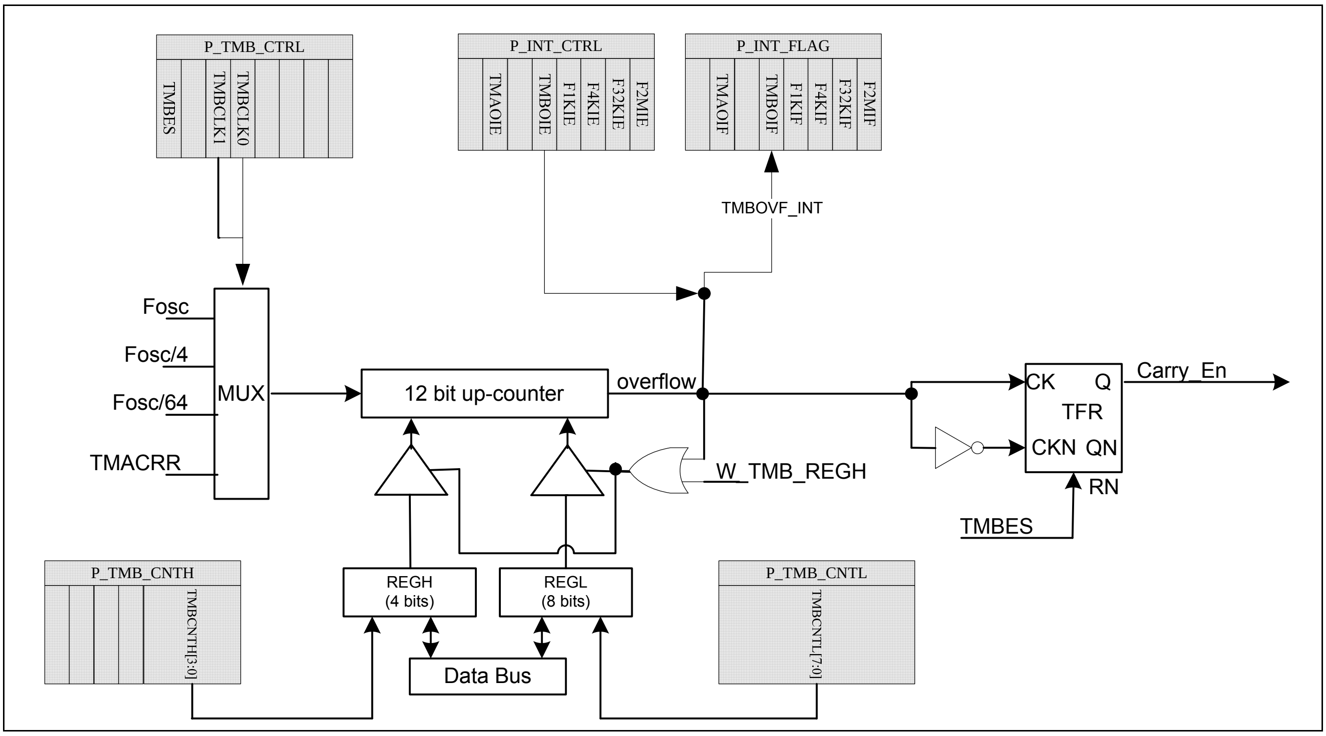


Figure 5-22 Timer B block diagram

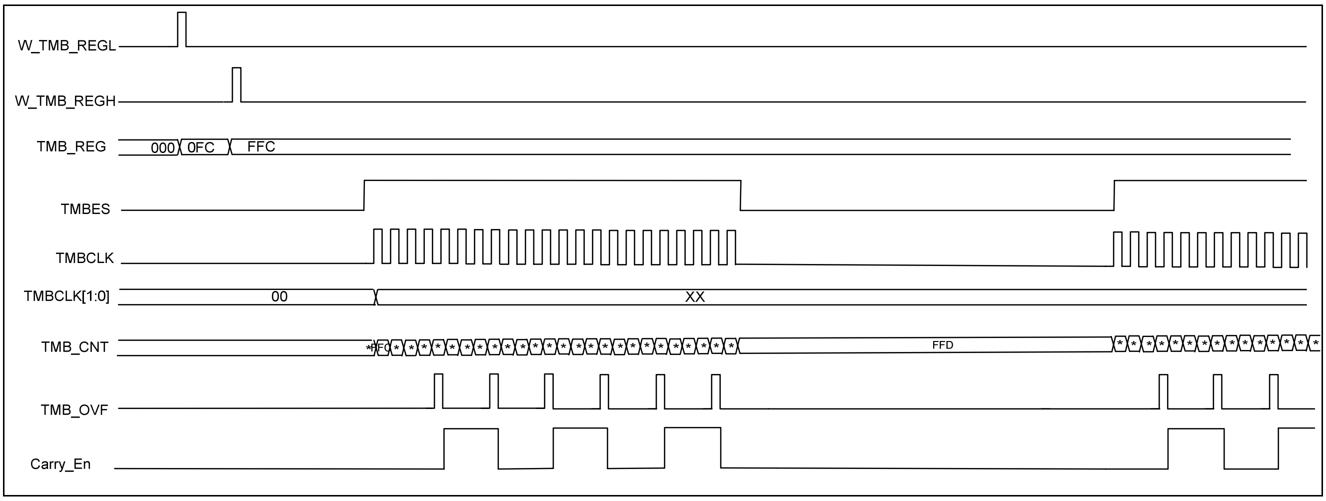


Figure 5-23 The Waveform of timer B

Timer B Control Register (P_TMB_CTRL, $0022)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	TMBES	-	TMBCLK1	TMBCLK0	-	-	-	-
ACCESS	R/W	-	R/W	R/W	-	-	-	-
DEFAULT	0	-	0	0	-	-	-	-

Bit [7] **TMBES**: Timer B enable/disable control.

0 = disable (C_TMBES_DIS)

1 = enable (C_TMBES_EN)

Bit [6] Reserved

Bit [5:4] **TMBCLK**[1：0]: Timer B clock source select bits

11 = TMACRR (C_TMBCLK_TMACRR)

10 = Fosc/64 (C_TMBCLK_64)

01 = Fosc/4 (C_TMBCLK_4)

00 = Fosc (C_TMBCLK_1)

Bit [3:0] Reserved

Timer B low 8-bit data Register (P_TMB_CNTL, $0025)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	TMBCNTL7	TMBCNTL6	TMBCNTL5	TMBCNTL4	TMBCNTL3	TMBCNTL2	TMBCNTL1	TMBCNTL0
ACCESS	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
DEFAULT	0	0	0	0	0	0	0	0

Bit [7:0] **TMBCNTL**[7：0]: Timer B low 8-bit pre-value for the counter.

Timer B high 4-bit data Register (P_TMB_CNTH, $0026)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	-	TMBCNTH3	TMBCNTH2	TMBCNTH1	TMBCNTH0
ACCESS	-	-	-	-	R/W	R/W	R/W	R/W
DEFAULT	-	-	-	-	0	0	0	0

Bit [3:0] **TMBCNTH**[3：0]: Timer B high 4-bit pre-value for the counter.

[Example] 5-16 Set Timer B selects timer A carrier signal as counter clock.

```
lda   #$FC                          ; Before starting timer, set Timer B counter initial value first
sta   P_TMB_CNTL                    ; set low 8-bit pre-value
lda   #$0F
sta   P_TMB_CNTH                    ; set high 4-bit pre-value
lda   #C_TMBES_EN + #C_TMBCLK_TMACRR
sta   P_TMB_CTRL                    ;Set clock source for TMA_Carrier
```

5.11. IR Transfer Module

IR TX is an analog block of GPM6C1064A1, which can driver LED by TX. **TX PWM driving current control** register can control this block. Users can adjust PWM driving ability by setting the value of P_PWM_DRV [4:3] (PWMDRV [1:0]), and 2/4 driving current is suggested to drive IR LED. TMAPWM signal as showed in Figure 5-24 controls the LED driving MOS. When Timer A is in PWM mode, it can generate PWM signal, and the PWM duty, frequency, on/off switch can be accuracy controlled by timer A. The Envelope PWM signal can be generated by timer A & timer B. And it has been illustrated in timer instruction.

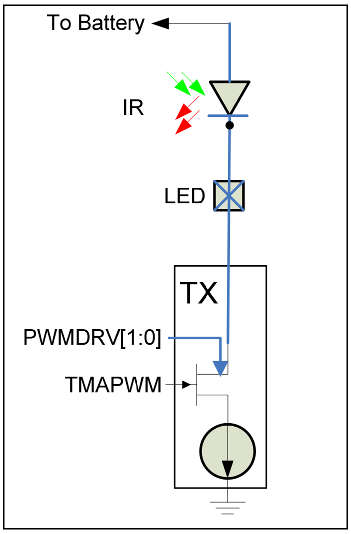


Figure 5-24 IR TX module diagram

TX PWM Driving Current Control Register (P_PWM_DRV, $0011)

BIT	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
NAME	-	-	-	PWMDRV1	PWMDRV0	-	-	-
ACCESS	-	-	-	W	W	-	-	-
DEFAULT	-	-	-	0	0	-	-	-

Bit [7:5] Reserved

Bit [4:3] **PWMDRV** [1：0]: PWM driving current select.

00 = 1/4 driving current (C_PWMDRV_1)

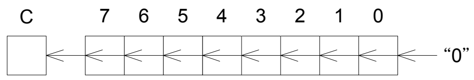
01 = 2/4 driving current (C_PWMDRV_2)

10 = 3/4 driving current (C_PWMDRV_3)

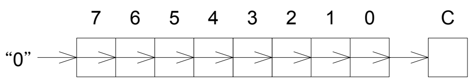
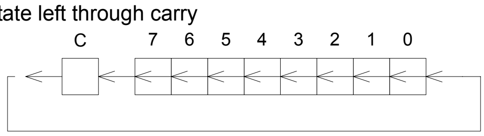
11 = full driving current (C_PWMDRV_4)

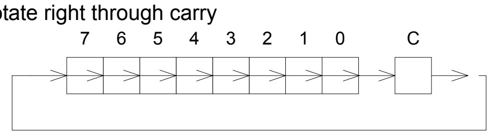
Bit [2:0] Reserved

5.12. Alphabetical List of Instruction Set

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
1.	ADC #dd	69	2	2	Add to accumulator with carry.	
2.	ADC aa	65	2	3	A ← (A) + (M) + C	
3.	ADC aa, X	75	2	4		
4.	ADC aaaa	6D	3	4	If D-flag set to 1, the ADC performs decimal operation.	NV--D-ZC
5.	ADC aaaa,X	7D	3	4(A)		
6.	ADC aaaa,Y	79	3	4(A)		
7.	ADC (aa,X)	61	2	6		
8.	ADC (aa), Y	71	2	5(A)		
9.	AND #dd	29	2	2	And memory data with accumulator.	
10.	AND aa	25	2	3	A ← (A) ^ (M)	
11.	AND aa, X	35	2	4		
12.	AND aaaa	2D	3	4		N-----Z-
13.	AND aaaa,X	3D	3	4(A)		
14.	AND aaaa,Y	39	3	4(A)		
15.	AND (aa,X)	21	2	6		
16.	AND (aa), Y	31	2	5(A)		
17.	ASL A	0A	1	2	Arithmetic Shift Left	
18.	ASL aa	06	2	5		
19.	ASL aa,X	16	2	6		N-----ZC
20.	ASL aaaa	0E	3	6		
21.	ASL aaaa,X	1E	3	6(A)		
22.	BCC aa	90	2	2(C)	Branch if carry bit clear If (C) = 0, then pc ← (pc) + ??	--------
23.	BCS aa	B0	2	2(C)	Branch if carry bit set If (C) = 1, then pc ← (pc) + ??	--------
24.	BEQ aa	F0	2	2(C)	Branch if equal If (Z) = 1, then pc ← (pc) + ??	--------
25.	BIT aa	24	2	3	Test bit in memory with accumulator	NV----Z-
26.	BIT aaaa	2C	3	4	Z ← (A) ^ (M), N ← (M_7), V ← (M_6)	
27.	BMI aa	30	2	2(C)	Branch if minus If (N) = 1, then pc ← (pc) + ??	--------
28.	BNE aa	D0	2	2(C)	Branch if not equal If (Z) = 0, then pc ← (pc) + ??	--------
29.	BPL aa	10	2	2(C)	Branch if plus If (N) = 0, then pc ← (pc) + ??	--------
30.	BRK	00	1	7	Software interrupt If (B) = 1, then pc ← (pc) + 1	---B-I--
31.	BVC aa	50	2	2(C)	Branch if overflow bit clear If (V) = 0, then pc ← (pc) + ??	--------
32.	BVS aa	70	2	2(C)	Branch if overflow bit set If (V) = 1, then pc ← (pc) + ??	--------
33.	CLC	18	1	2	Clear C-flag : C ← “0”	-------0
34.	CLD	D8	1	2	Clear D-flag : D ← “0”	----0---

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
35.	CLI	58	1	2	Clear I-flag : I ← "0"	-----0--
36.	CLV	B8	1	2	Clear V-flag : V ← "0"	-0------
37.	CMP #dd	C9	2	2	Compare memory data with accumulator,	N-----ZC
38.	CMP aa	C5	2	3	(A) – (M)	
39.	CMP aa, X	D5	2	4		
40.	CMP aaaa	CD	3	4		
41.	CMP aaaa,X	DD	3	4(A)		
42.	CMP aaaa,Y	D9	3	4(A)		
43.	CMP (aa,X)	C1	2	6		
44.	CMP (aa), Y	D1	2	5(A)		
45.	CPX #dd	E0	2	2	Compare memory data with X-register,	N-----ZC
46.	CPX aa	E4	2	3	(X) – (M)	
47.	CPX aaaa	EC	3	4		
48.	CPY #dd	C0	2	2	Compare memory data with Y-register,	N-----ZC
49.	CPY aa	C4	2	3	(Y) – (M)	
50.	CPY aaaa	CC	3	4		
51.	DEC aa	C6	2	5	Decrement	N-----Z-
52.	DEC aa, X	D6	2	6	M ← (M) - 1	
53.	DEC aaaa	CE	3	6		
54.	DEC aaaa,X	DE	3	7		
55.	DEX	CA	1	2		
56.	DEY	88	1	2		
57.	EOR #dd	49	2	2	Exclusive OR	N-----Z-
58.	EOR aa	45	2	3	A ← (A) $\oplus$ (M)	
59.	EOR aa, X	55	2	4		
60.	EOR aaaa	4D	3	4		
61.	EOR aaaa,X	5D	3	4(A)		
62.	EOR aaaa,Y	59	3	4(A)		
63.	EOR (aa,X)	41	2	6		
64.	EOR (aa), Y	51	2	5(A)		
65.	INC aa	E6	2	5	Increment	N-----Z-
66.	INC aa, X	F6	2	6	M ← (M) + 1	
67.	INC aaaa	EE	3	6		
68.	INC aaaa,X	FE	3	7		
69.	INX	E8	1	2	X ← X + 1	N-----Z-
70.	INY	C8	1	2	Y ← Y + 1	N-----Z-
71.	JMP aaaa	4C	3	3	Unconditional jump	--------
72.	JMP (aaaa)	6C	3	6	Pc ← jump address	
73.	JSR aaaa	20	3	6	Jump to subroutine (sp) ← (pc_H), sp ← sp – 1, (sp) ← (pc_L), sp ← sp – 1, pc ← aaaa	--------

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
74.	LDA #dd	A9	2	2	Load accumulator A ← (M)	N-----Z-
75.	LDA aa	A5	2	3		
76.	LDA aa, X	B5	2	4		
77.	LDA aaaa	AD	3	4		
78.	LDA aaaa,X	BD	3	4(A)		
79.	LDA aaaa,Y	B9	3	4(A)		
80.	LDA (aa,X)	A1	2	6		
81.	LDA (aa), Y	B1	2	5(A)		
82.	LDX #dd	A2	2	2	Load X-register X ← (M)	N-----Z-
83.	LDX aa	A6	2	3		
84.	LDX aa, Y	B6	2	4		
85.	LDX aaaa	AE	3	4		
86.	LDX aaaa,Y	BE	3	4(A)		
87.	LDY #dd	A0	2	2	Load Y-register Y ← (M)	N-----Z-
88.	LDY aa	A4	2	3		
89.	LDY aa, X	B4	2	4		
90.	LDY aaaa	AC	3	4		
91.	LDY aaaa,X	BC	3	4(A)		
92.	LSR A	4A	1	2	Logical shift right  	N-----ZC
93.	LSR aa	46	2	5		
94.	LSR aa, X	56	2	6		
95.	LSR aaaa	4E	3	6		
96.	LSR aaaa,X	5E	3	6(A)		
97.	NOP	EA	1	2	No operation	--------
98.	ORA #dd	09	2	2	Logical OR A ← (A) v (M)	N-----Z-
99.	ORA aa	05	2	3		
100.	ORA aa, X	15	2	4		
101.	ORA aaaa	0D	3	4		
102.	ORA aaaa,X	1D	3	4(A)		
103.	ORA aaaa,Y	19	3	4(A)		
104.	ORA (aa,X)	01	2	6		
105.	ORA (aa), Y	11	2	5(A)		
106.	PHA	48	1	3	(sp) ← A, sp ← sp - 1	--------
107.	PHP	08	1	3	(sp) ← P status, sp ← sp -1	
108.	PLA	68	1	4	sp ← sp +1, A ← (sp)	--------
109.	PLP	28	1	4	Sp ← sp +1, P status ← (sp)	restored
110.	ROL A	2A	1	2	Rotate left through carry  	N-----ZC
111.	ROL aa	26	2	5		
112.	ROL aa, X	36	2	6		
113.	ROL aaaa	2E	3	6		
114.	ROL aaaa,X	3E	3	6(A)		

NO.	MNEMONIC	OP CODE	BYTE NO	CYCLE NO	OPERATION	FLAG NV-BDIZC
115.	ROR A	6A	1	2	Rotate right through carry 	N-----ZC
116.	ROR aa	66	2	5		
117.	ROR aa, X	76	2	6		
118.	ROR aaaa	6E	3	6		
119.	ROR aaaa,X	7E	3	6(A)		
120.	RTI	40	1	6	Return from interrupt Sp ← sp + 1, P status ← (sp), sp ← sp + 1, pc_L← (sp), sp ← sp +1, pc_H ← (sp)	restored
121.	RTS	60	1	6	Return from subroutine Sp ← sp + 1, pc_L← (sp), sp ← sp +1, pc_H ← (sp)	--------
122.	SBC #dd	E9	2	2	Subtract with carry A ← (A) – (M) - ~(C)	NV----ZC
123.	SBC aa	E5	2	3		
124.	SBC aa, X	F5	2	4		
125.	SBC aaaa	ED	3	4		
126.	SBC aaaa,X	FD	3	4(A)		
127.	SBC aaaa,Y	F9	3	4(A)		
128.	SBC (aa,X)	E1	2	6		
129.	SBC (aa), Y	F1	2	5(A)		
130.	SEC	38	1	2	Set C-flag：C ← “1”	-------1
131.	SED	F8	1	2	Set D-flag：D ← “1”	----1---
132.	SEI	78	1	2	Set I-flag：I ← “1”	-----1--
133.	STA aa	85	2	3	Store accumulator in memory (M) ← A	--------
134.	STA aa, X	95	2	4		
135.	STA aaaa	8D	3	4		
136.	STA aaaa,X	9D	3	5		
137.	STA aaaa,Y	99	3	5		
138.	STA (aa,X)	81	2	6		
139.	STA (aa), Y	91	2	6		
140.	STX aa	86	2	3	Store X-register in memory (M) ← X	--------
141.	STX aa, Y	96	2	4		
142.	STX aaaa	8E	3	4		
143.	STY aa	84	2	3	Store Y-register in memory (M) ← Y	--------
144.	STY aa, X	94	2	4		
145.	STY aaaa	8C	3	4		
146.	TAX	AA	1	2	Transfer accumulator to X-register：X ← A	N-----Z-
147.	TAY	A8	1	2	Transfer accumulator to Y-register：Y ← A	N-----Z-
148.	TSX	BA	1	2	Transfer sp to X-register：X ← sp	N-----Z-
149.	TXA	8A	1	2	Transfer X-register to accumulator：A ← X	N-----Z-
150.	TXS	9A	1	2	Transfer X-register to sp：sp ← X	N-----Z-
151.	TYA	98	1	2	Transfer Y-register to accumulator：A ← Y	N-----Z-

Notes:

1. Cycle (A): Cycle+1 when cross a boundary.
2. Cycle(C): Cycle+1 if the branch condition is true; Cycle+2 if the branch condition is true and cross a boundary.

6. ELECTRICAL CHARACTERISTICS

6.1. Absolute Maximum Ratings

Characteristics	Symbol	Ratings
DC Supply Voltage	V_+	< 5.0V
Input Voltage Range	V_{IN}	-0.5V to V_+ + 0.5V
Operating Temperature	T_A	0℃ to +70℃
Storage Temperature	T_{STO}	-50℃ to +150℃
Average PWM MAX Driving Current	I_{LEDM}	150mA
VDD Total MAX Current	I_{VDDM}	100mA
VSS Total MAX Current	I_{VSSM}	120mA

Note: Stresses beyond those given in the Absolute Maximum Rating table may cause operational errors or damage to the device. For normal operational conditions see AC/DC Electrical Characteristics.

6.2. AC Characteristics (T_A =25℃)

Characteristics	Symbol	Limit			Unit	Test Condition
		Min.	Typ.	Max.		
OSC Frequency	F_{OSC}	4.0×(1-3.0%)	4.0×(1±1.5%)	4.0×(1+3.0%)	MHz	VDD = 2.0V - 3.6V, 2-battery
		8.0×(1-3.0%)	8.0×(1±1.5%)	8.0×(1+3.0%)		

6.3. DC Characteristics (VDD = 3.0V, T_A = 25℃)

Characteristics	Symbol	Limit			Unit	Test Condition
		Min.	Typ.	Max.		
Operating Voltage	VDD	2.0	-	3.6	V	For 2-battery
Operating Current	I_{OP}	-	4.0	8.0	mA	F_{CPU} = 8.0MHz @ 3.6V, no load
Standby Current	I_{STBY}	-	-	1.0	uA	VDD = 3.6V
Input High Level	V_{IH}	0.7VDD	-	-	V	VDD = 3.0V
Input Low Level	V_{IL}	-	-	0.3VDD	V	VDD = 3.0V
Output High Level PB, PC, PD	V_{OH}	0.8VDD	-	-	V	VDD = 3.0V I_{OH} = -6mA
Output Low Level PB, PC, PD	V_{OL}	-	-	0.2VDD	V	VDD = 3.0V I_{OL} = 16mA
Input Pull High Resistor PA, PB, PC, PD	R_H	30	50	70	Kohm	Pull High VDD = 3.0V
Input Pull Low Resistor PA, PB, PC, PD	R_L	30	50	70	Kohm	Pull Low VDD = 3.0V
Max PWM driving Current	I_{PWM}	300	-	-	mA	VDD = 3.0V,V_{LED} = 3.0V, PWMDRV[1:0]=11
LVR Active Voltage	V_{LVR}	1.7	1.85	2.0	V	-

7. APPLICATION CIRCUITS

There are two application circuit as below

MODE1: 0ne cap mode for short time sram data hold VDD, VREF must bond together, and connect with 47uF capacitor.

MODE2: two cap modes for long time sram data hold, VDD connects with 0.1uF capacitor, and VREF connects with 47uF capacitor.

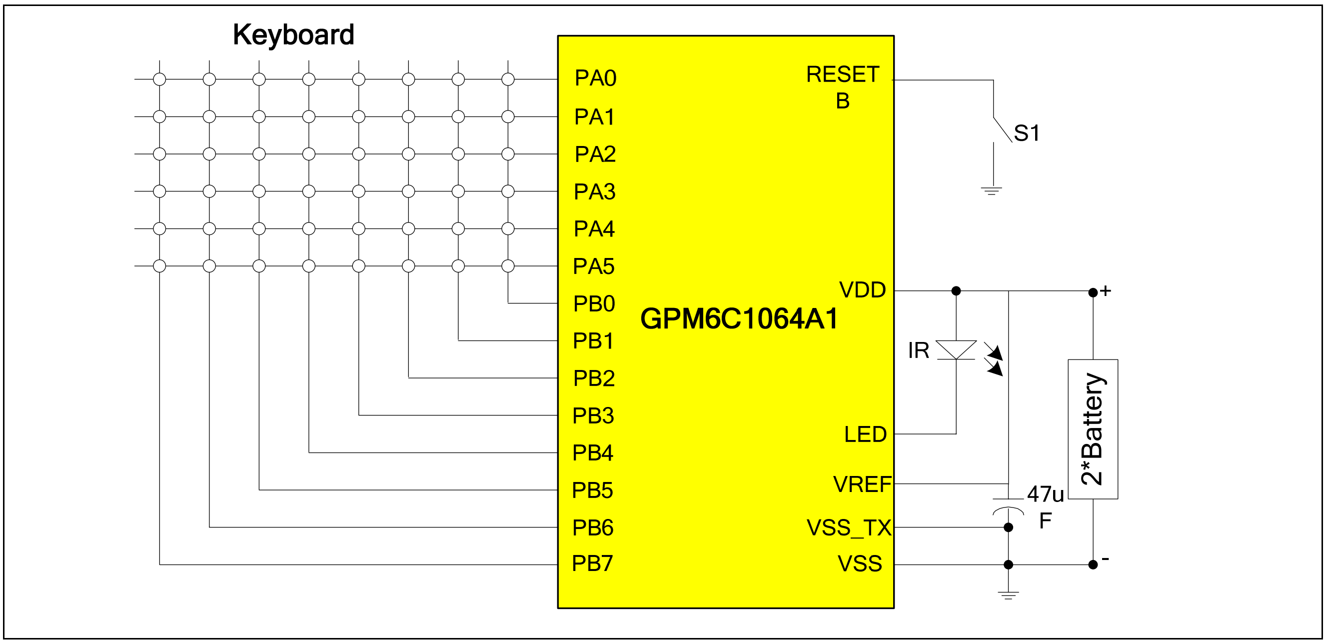


Figure 25 GPM6C1064A1 MODE1 application circuit diagram

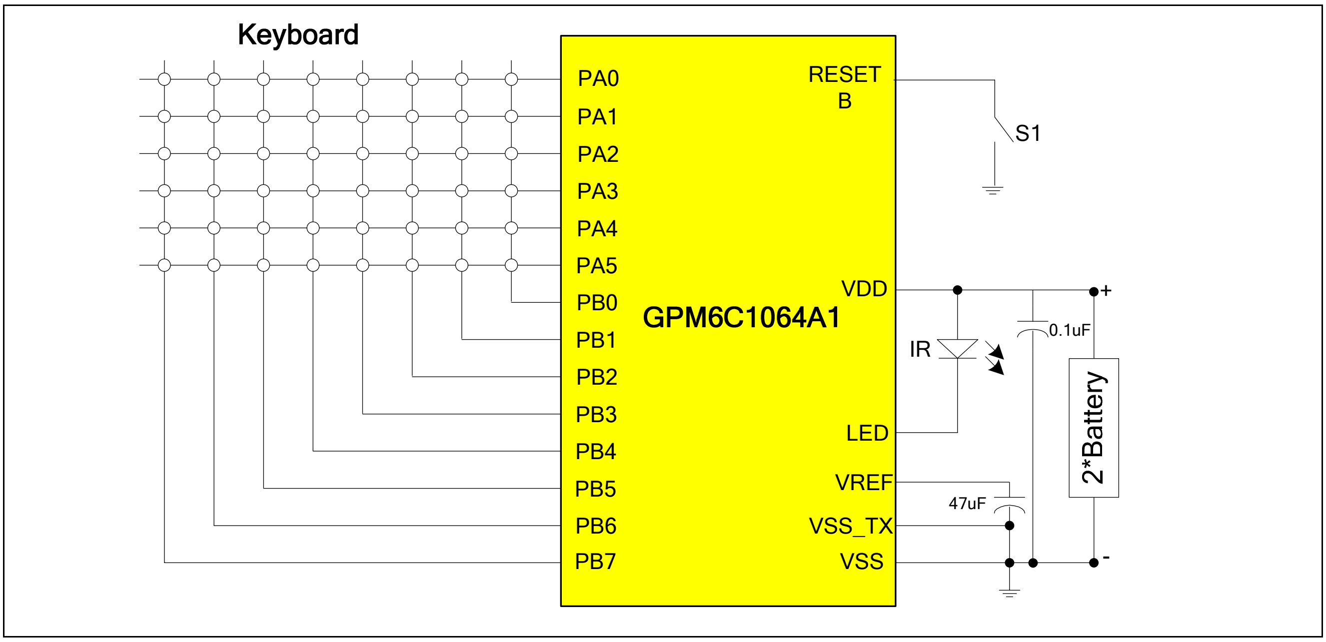


Figure 26 GPM6C1064A1 MODE2 application circuit diagram

8.PACKAGE/PAD LOCATIONS

8.1. Ordering Information

Product Number	Package Type
GPM6C1064A1-NnnV-C	Chip form
GPM6C1064A1-NnnV-QL01x	Halogen Free Package

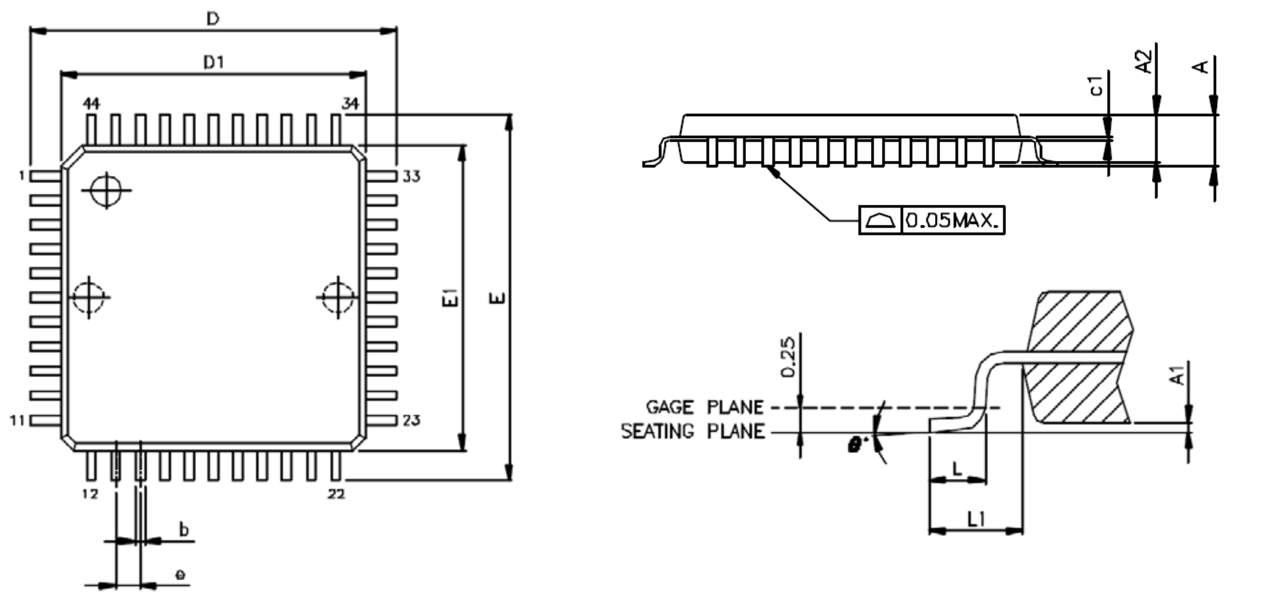
Note1: Code number is assigned for customer.

Note2: Code number (N = A - Z or 0 - 9, nn = 00 - 99); version (V = A - Z).

Note3: Package form number (x = 1 - 9, serial number).

8.2. Package Information

8.2.1. LQFP 44



Symbol	Dimension in Millimeter		
	Min.	Nom.	Max.
A	-	-	1.60
A1	0.05	-	0.15
A2	1.35	1.40	1.45
c1	0.09	-	0.16
D	12.00 BSC		
D1	10.00 BSC		
E	12.00 BSC		
E1	10.00 BSC		
e	0.80 BSC		
b	0.30	0.37	0.45
L	0.45	0.60	0.75
L1	1.00 REF		
θ°	0°	3.5°	7°

9. DISCLAIMER

The information appearing in this publication is believed to be accurate.

Integrated circuits sold by Generalplus Technology are covered by the warranty and patent indemnification provisions stipulated in the terms of sale only. GENERALPLUS makes no warranty, express, statutory implied or by description regarding the information in this publication or regarding the freedom of the described chip(s) from patent infringement. FURTHERMORE, GENERALPLUS MAKES NO WARRANTY OF MERCHANTABILITY OR FITNESS FOR ANY PURPOSE. GENERALPLUS reserves the right to halt production or alter the specifications and prices at any time without notice. Accordingly, the reader is cautioned to verify that the data sheets and other information in this publication are current before placing orders. Products described herein are intended for use in normal commercial applications. Applications involving unusual environmental or reliability requirements, e.g. military equipment or medical life support equipment, are specifically not recommended without additional processing by GENERALPLUS for such applications. Please note that application circuits illustrated in this document are for reference purposes only.

10. REVISION HISTORY

Date	Revision #	Description	Page
APR. 19, 2011	0.1	Original	40