FT64F0AX

IO Application note

目录

1. GPIO 3
1.1. IO 端口相关寄存器汇总 6
1.2. I/O 配置 10
1.3. PORTx 功能及优先级 12
1.4. PORT 端口变化中断 13
1.5. 关于读端口 PORTx 13
2. 应用范例 16
联系信息 19

FT64F0Ax IO 应用

1. GPIO

本芯片共包含 18 个 GPIO。这些 IO 除了作为普通输入/输出端口以外还通常具备一些与内核周边电路通讯的功能。

每个端口有 8 个标准寄存器供其操作使用。这些寄存包括：

- TRISx 寄存器 (数据方向寄存器)
- PORTx 寄存器 (用于读器件引脚上的电平)
- LATx 寄存器 (输出锁存器)
- WPUx 寄存器 (上拉控制)
- WPDx 寄存器 (下拉控制)
- PSRCx 寄存器 (源电流选择)
- PSINKx 寄存器 (灌电流选择)
- ITYPEx 寄存器 (中断类型选择)

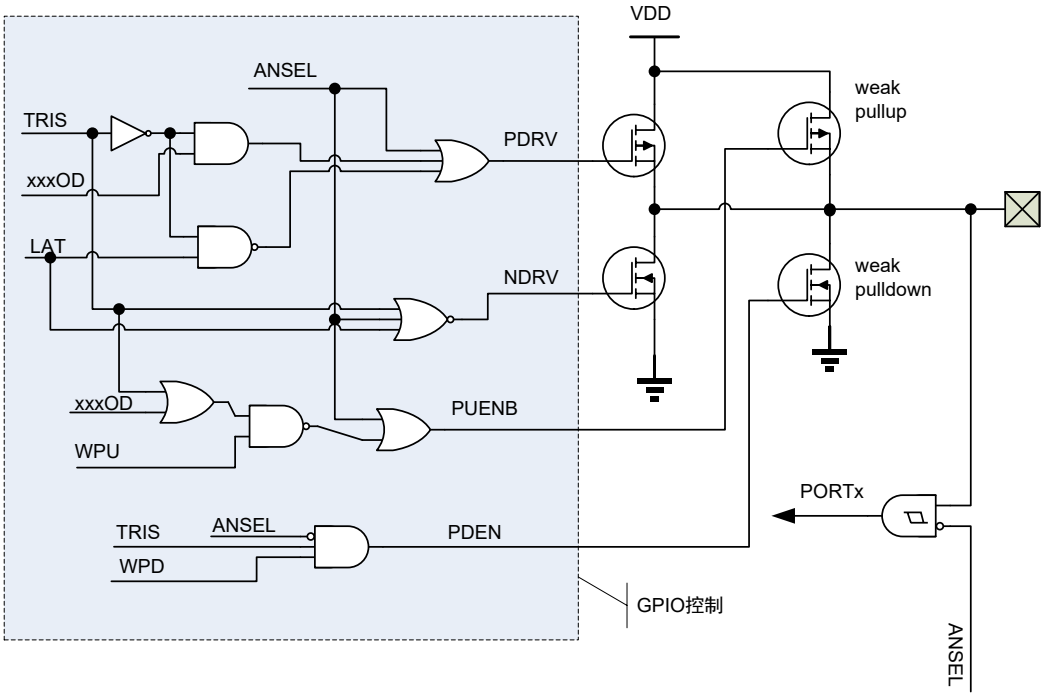

图 1-1 I/O 的结构原理

某些端口可能有以下额外寄存器：

- ANSELAx (模拟选择寄存器)

通常情况下，当某个端口上的外设使能时，其相关引脚可能不能用作通用输出引脚，但可读取该引脚。

数据锁存器 (LATx 寄存器)用于对 I/O 引脚所驱动的值进行“读-修改-写”操作。对 LATx 寄存器的写操作与对相应 PORTx 寄存器的写操作具有相同的效果。读 LATx 寄存器将读取保存在 I/O 端口锁存器中的值，而读 PORTx 寄存器将读取实际的 I/O 引脚值。

支持模拟输入的端口具有相关的 ANSELAx 寄存器。当 ANSELA 位置 1 时，禁止与该位相关的数字输入缓冲器。禁止输入缓冲器可防止逻辑输入电路产生短路电流。

所有 I/O 引脚均具有以下功能（**表 1-2，表 1-4**）：

- 数字输出
- 数字输入
- 开漏（SPI, I2C, USART 相应端口）
- 弱上拉
- 弱下拉

此外，部分 I/O 具有以下特殊功能：

1. 烧录调试引脚（ISP-Data，ISP-CLK），硬件内部连接，不需设置。

2. 通过 IDE 界面配置，且在芯片初始化配置时加载的功能（**表 1-3**）：

   - 外部时钟/晶振输入（OSC1, OSC2）
   - 系统外部复位（/MCLRB）

3. 通过指令对相应 I/O 引脚进行配置的其他功能，可分为 5 类：

   a) 数字输出

   - PWM
   - LVD 输出
   - 内部时钟输出

   b) 数字输入

   - PWM 故障刹车
   - Timer1 触发
   - GPIO 端口变化中断
   - ADC 触发（ADC_ETR）

   c) 模拟输入

   - LVD / BOR
   - OP0+
   - OP0−
   - ADC
   - V_{REF}+
   - V_{REF}−

   d) 模拟输出

   - OP0 输出

   e) 通信接口

   - SPI
   - I2C
   - USART

引脚名	ISP 调试	时钟	ADC	SPI	I2C	USART1	USART2	中断	LVD	OPAMP	PWM	数字 I/O 上拉/下拉	开漏	源电流 (mA)	灌电流 (mA)
PA0			(V_{REF}−)	MOSI				√			PWM1	√	√	4, 29	48, 61
PA1			(V_{REF}+)	MISO				√			PWM2	√	√	4, 29	48, 61
PA2	CLK				[SCL]	[RX]		√			PWM4	√	√	4, 29	48, 61
PA3			AN1				[TX]	√			PWM7	√		4, 29	48, 61
PA4			AN2/ Trigger				[RX]	√			PWM6	√		4, 29	48, 61
PA5						CK		√	LVDOUT	OP0+	PWM5	√		4, 29	48, 61
PA6			AN3			TX		√		OP0-		√	√	4, 29	48, 61
PA7		输出	AN4			RX	[TX]	√	ELVD0	OP0OUT	PWM4	√	√	4 ,29	48, 61
PB0				SCK				√			[PWM2]/ PWM3N/ [PWM5]	√		8, 29	48, 61
PB1		输出	AN0					√			PWM2N/ PWM4	√		8, 29	48, 61
PB2				[SCK]				√			PWM3/ [PWM1N]	√		8, 29	48, 61
PB3			[Trigger]		SCL		CK	√	[LVDOUT]		TIM1_ETR	√	√	8, 29	48, 61
PB4					SDA		TX	√	[LVDOUT]		BKIN	√	√	8, 29	48, 61
PB5				NSS			RX	√	[LVDOUT]		PWM7	√		8, 29	48, 61
PB6	DATA		AN6		[SDA]	[TX]		√				√	√	8, 29	48, 61
PB7		OSC−		[MOSI]				√	ELVD3			√	√	8, 29	48, 61
PC0			AN5					√	ELVD1		PWM1N	√		4, 8, 29	48, 61
PC1		OSC+		[MISO]				√	ELVD2			√	√	4, 8, 29	48, 61
注								/MCLRB = PC0						V_{DD}=5, V_{DS}=0.5	

表 1-1 I/O 端口功能

注： 所有 IO 支持 3 档可配置源电流驱动能力 (参阅 “PSRCx”，**表 1-4**)，和 2 档可配置灌电流驱动能力 (参阅 “PSINKx”，**表 1-4**)。

1.1. IO 端口相关寄存器汇总

名称	地址	bit 7	bit 6	bit 5	bit 4	bit 3	bit 2	bit 1	bit 0	复位值
ANSELA	0x197	–	ANSELA[6:0]							-000 0000
TRISA	0x8C	PORTA 方向控制								1111 1111
TRISB	0x8D	PORTB 方向控制								1111 1111
TRISC	0x8E	–	–	–	–	–	–	PORTC 方向控制		---- --11
PORTA	0x0C	PORTA 输出寄存器								xxxx xxxx
PORTB	0x0D	PORTB 输出寄存器								xxxx xxxx
PORTC	0x0E	–	–	–	–	–	–	PORTC 输出寄存器		---- --xx
LATA	0x10C	PORTA 数据锁存器								xxxx xxxx
LATB	0x10D	PORTB 数据锁存器								xxxx xxxx
LATC	0x10E	–	–	–	–	–	–	PORTC 数据锁存器		---- --xx
WPUA	0x18C	PORTA 弱上拉								0000 0000
WPUB	0x18D	PORTB 弱上拉								0000 0000
WPUC	0x18E	–	–	–	–	–	–	PORTC 弱上拉		---- --00
WPDA	0x20C	PORTA 弱下拉								0000 0000
WPDB	0x20D	PORTB 弱下拉								0000 0000
WPDC	0x20E	–	–	–	–	–	–	PORTC 弱下拉		---- --00
ODCON0	0x21F	–	–	–	–	UR2OD	SPIOD	I2COD	UR1OD	---- 0000
PSRC0	0x11A	PORTA 源电流设置								1111 1111
PSRC1	0x11B	PORTB 源电流设置								1111 1111
PSRC2	0x11C	–	–	–	–	PORTC 源电流设置				---- 1111
PSINK0	0x19A	PORTA 灌电流设置								0000 0000
PSINK1	0x19B	PORTB 灌电流设置								0000 0000
PSINK2	0x19C	–	–	–	–	–	–	PORTC 灌电流设置		---- --00
ITYPE0	0x11E	PORTx[1:0] (x = A, B, C)和 PORTy[3:2] (y = A, B)外部管脚中断类型设置								0000 0000
ITYPE1	0x11F	PORTy[7:4] (y = A, B) 外部管脚中断类型设置								0000 0000
AFP0	0x19E	管脚重映射寄存器 0								0000 0000
AFP1	0x19F	管脚重映射寄存器 1								0000 0000
AFP2	0x198	–	–	管脚重映射寄存器 2						--00 0000
EPS0	0x118	外部中断 EINT3~0 管脚选择 0								0000 0000
EPS1	0x119	外部中断 EINT7~4 管脚选择 1								0000 0000
EPIE0	0x94	外部管脚中断使能位								0000 0000
EPIF0	0x14	外部管脚中断标志位								0000 0000

表 1-2 I/O 相关用户寄存器的地址和复位值

名称	功能	默认
MCLRE	外部 I/O 复位	关闭
FOSC	• LP：PC1 (+) 和 PB7 (−) 接外部低速晶振 • XT：PC1 (+) 和 PB7 (−) 接外部高速晶振 • EC：PC1 (+) 接外部时钟输入，PB7 为 I/O • INTOSCIO：PC1 和 PB7 为 I/O	INTOSCIO

表 1-3 I/O 相关初始化配置寄存器

名称	状态		寄存器	地址	复位值
TRISA	PORTA	PORT 端口数字输出 (方向控制) 1 = 关闭 0 = 使能 (关闭上拉/下拉)	TRISA[7:0]	0x8C	RW−1111 1111
TRISB	PORTB		TRISB[7:0]	0x8D	RW−1111 1111
TRISC	PORTC		TRISC[1:0]	0x8E	RW−11
ANSELA	1 = 关闭上拉/下拉，及数字输入 (仅适用于 7 个 ADC 通道) 0 = (无动作)		ANSELA[6:0]	0x197	RW−000 0000
WPUA	PORTA	弱上拉 1 = 使能 0 = 关闭	WPUA[7:0]	0x18C	RW−0000 0000
WPUB	PORTB		WPUB[7:0]	0x18D	RW−0000 0000
WPUC	PORTC		WPUC[1:0]	0x18E	RW−00
WPDA	PORTA	弱下拉 1 = 使能 0 = 关闭	WPDA[7:0]	0x20C	RW−0000 0000
WPDB	PORTB		WPDB[7:0]	0x20D	RW−0000 0000
WPDC	PORTC		WPDC[1:0]	0x20E	RW−00
PORTA	PORTA	数据输出寄存器 读：返回 IO 引脚上的电平 写：写入相应的 LATx 寄存器	PORTA[7:0]	0x0C	RW−xxxx xxxx
PORTB	PORTB		PORTB[7:0]	0x0D	RW−xxxx xxxx
PORTC	PORTC		PORTC[1:0]	0x0E	RW−xx
LATA	PORTA	数据锁存器寄存器	LATA[7:0]	0x10C	RW−xxxx xxxx
LATB	PORTB		LATB[7:0]	0x10D	RW−xxxx xxxx
LATC	PORTC		LATC[1:0]	0x10E	RW−xx
UR2OD	USART2_TX	开漏 1 = 使能 0 = 关闭	ODCON0[3]	0x21F	RW−0
SPIOD	SPI_MISO, SPI_MOSI		ODCON0[2]		RW−0
I2COD	I2C_SDA, I2C_SCL		ODCON0[1]		RW−0
UR1OD	USART1_TX		ODCON0[0]		RW−0
AFP0	USART1_TX 复用管脚选择 00 = PA6　10 = PA7 01 = PB6　11 = PA2		AFP0[7:6]	0x19E	RW−00
	I2C_SCL 复用管脚选择	1 = PA2　0 = PB3	AFP0[5]		RW−0
	TIM2_CH1 复用管脚选择	1 = PB0　0 = PA5	AFP0[4]		RW−0

名称	状态		寄存器	地址	复位值
	TIM2_CH3 复用管脚选择	1 = PA3 0 = PB5	AFP0[3]		RW-0
	TIM1_CH1N 复用管脚选择	1 = PB2 0 = PC0	AFP0[2]		RW-0
	ADC_ETR 复用管脚选择	1 = PB3 0 = PA4	AFP0[1]		RW-0
	I2C_SDA 复用管脚选择	1 = PB6 0 = PB4	AFP0[0]		RW-0
AFP1	USART1_RX 复用管脚选择 00 = PA7 10 = PA6 10 = PA2 11 = PB6		AFP1[7:6]	0x19F	RW-00
	TIM1_CH4 复用管脚选择 00/01= PB1 10 = PA7 11 = PA2		AFP1[5:4]		RW-00
	TIM1_CH2 复用管脚选择	1 = PB0 0 = PA1	AFP1[3]		RW-0
	SPI_SCK 复用管脚选择	1 = PB2 0 = PB0	AFP1[2]		RW-0
	SPI_MOSI 复用管脚选择	1 = PB7 0 = PA0	AFP1[1]		RW-0
	SPI_MISO 复用管脚选择	1 = PC1 0 = PA1	AFP1[0]		RW-0
AFP2	USART2_RX 复用管脚选择 000 = PB5 011 = PA3 001 = PA4 1xx = PA7 010 = PB4		AFP2[5:3]	0x198	RW-000
	USART2_TX 复用管脚选择 000 = PB4 011 = PA4 001 = PA3 1xx = PA7 010 = PB5		AFP2[2:0]		RW-000
PSINK0	PA7-PA0	灌电流 (mA) 1 = 62 0 = 48	PSINK0[7:0]	0x19A	RW-0000 0000
PSINK1	PB7-PB0		PSINK1[7:0]	0x19B	RW-0000 0000
PSINK2	PC1-PC0		PSINK2[1:0]	0x19C	RW-00
PSRC0	PA7-PA0	源电流 (mA) 1 = 29 0 = 4	PSRC0[7:0]	0x11A	RW-1111 1111
PSRC1	PB7-PB0	源电流 (mA) 1 = 29 0 = 8	PSRC1[7:0]	0x11B	RW-1111 1111
PSRC2	PC1	源电流能力(mA) (00) = 4 (01) = 8 / (10) = 8 (11) = 29	PSRC2[3:2]	0x11C	RW-11
	PC0		PSRC2[1:0]		RW-11

表 1-4 I/O 相关用户寄存器

名称	状态		寄存器	地址	复位值
ITYPE0[1:0]	PORTx.0	外部管脚中断触发类型 00 = 低电平 01 = 上升沿 10 = 下降沿 11 = 双边沿	ITYPE0[1:0]	0x11E	RW-00
ITYPE0[3:2]	PORTx.1		ITYPE0[3:2]		RW-00
ITYPE0[5:4]	PORTy.2		ITYPE0[5:4]		RW-00
ITYPE0[7:6]	PORTy.3		ITYPE0[7:6]		RW-00
ITYPE1[1:0]	PORTy.4		ITYPE1[1:0]	0x11F	RW-00
ITYPE1[3:2]	PORTy.5		ITYPE1[3:2]		RW-00
ITYPE1[5:4]	PORTy.6		ITYPE1[5:4]		RW-00
ITYPE1[7:6]	PORTy.7		ITYPE1[7:6]		RW-00

表 1-5 I/O 中断触发寄存器 (x = A, B, C; y = A, B)

名称	状态		寄存器	地址	复位值
EINT0	EINT0 管脚选择	00 = PA0 10 = PC0 01 = PB0 11 = 保留	EPS0[1:0]	0x118	RW-00
EINT1	EINT1 管脚选择	00 = PA1 10 = PC1 01 = PB1 11 = 保留	EPS0[3:2]		RW-00
EINT2	EINT2 管脚选择	00 = PA2 1x = 保留 01 = PB2	EPS0[5:4]		RW-00
EINT3	EINT3 管脚选择	00 = PA3 1x = 保留 01 = PB3	EPS0[7:6]		RW-00
EINT4	EINT4 管脚选择	00 = PA4 1x = 保留 01 = PB4	EPS1[1:0]	0x119	RW-00
EINT5	EINT5 管脚选择	00 = PA5 1x = 保留 01 = PB5	EPS1[3:2]		RW-00
EINT6	EINT6 管脚选择	00 = PA6 1x = 保留 01 = PB6	EPS1[5:4]		RW-00
EINT7	EINT7 管脚选择	00 = PA7 1x = 保留 01 = PB7	EPS1[7:6]		RW-00

表 1-6 外部中断管脚选择寄存器

名称	状态		寄存器	地址	复位值
EPIE0	外部中断使能位	1 = 使能 0 = 禁止	EPIE0[7:0]	0x94	RW-00000000
EPIF0 [1]	外部中断标志位	1 = Yes (锁存) 0 = No	EPIF0[7:0]	0x14	R_W1C-00000000

表 1-7 外部中断使能和中断标志寄存器

[1] 写 1 清 0，写 0 无效。建议只使用 STR、MOVWI 指令进行写操作，而不要用 BSR 或 IOR 指令。

1.2. I/O 配置

每个 PORT 端口，均需根据其相应功能配置以下 4 个模块（**表 1-4**）：

- 弱上拉
- 弱下拉
- 开漏
- 数字输入
- 数字输出

功能	数字输入	上拉/下拉	数字输出	设置
ISP-DATA	On	Off	On	(硬件内置，忽略指令)
ISP-CLK	On	Off	Off	(硬件内置，忽略指令)
/MCLRB	On	上拉	Off	(初始化配置，忽略指令)
时钟输出	(忽略)	Off	On	(初始化配置，忽略指令)
OSC+ (EC)	On	(可选)	Off	(初始化配置，忽略指令)
OSC+ / OSC− (LP, XT)	Off	Off	Off	(初始化配置，忽略指令)
ADC	Off	Off	Off	TRISx = 1; ANSELAx = 1
OP0OUT	Off	Off	Off	OP0ON = 1; OPTOIO = 1
SPI 输出	On	Off	On	TRISx = 0
I2C 输出	On	Off	On	TRISx = 0
USART 输出	On	Off	On	TRISx = 0
LVD	Off (5)	Off	Off	TRISx = 1; ANSELAx = 1
V_{REF}+ / V_{REF}−	Off	Off	Off	TRISx = 1
ADC 触发	On	(可选)	Off	TRISx = 1
OP0+ 输入	Off	Off	Off	TRISx = 1; ANSELAx = 1
OP0− 输入	Off	Off	Off	TRISx = 1; ANSELAx = 1
SPI 输入	On	(可选)	Off	TRISx = 1
I2C 输入	On	(可选)	Off	TRISx = 1
USART 输入	On	(可选)	Off	TRISx = 1
端口变化中断	On	(可选)	Off	TRISx = 1
BKIN	On	(可选)	Off	TRISx = 1
数字输入	On	(可选)	Off	TRISx = 1
PWM	On	Off	On	TRISx = 0
数字输出	On	Off	On	TRISx = 0

表 1-8 I/O 配置标志和用户寄存器

注：

1. TRISx = 0：“数字输出”使能，“上拉/下拉”自动关闭 (忽略 WPDx, WPUx)。
2. TRISx = 1：“数字输出”关闭。

3. ANSELAx = 1：“上拉”、“下拉”、“数字输入”自动关闭 (忽略 WPDx, WPUx)。
4. 可关闭“数字输入”的唯一指令为“ANSELAx = 1”。
5. 将 PORT 端口设置为 LVD 输入时，其“数字输入”、“上拉”和“下拉”功能被自动关闭。当 LVD 输入需要在不同的通道之间切换使用时，通过设置“ANSELAx = 1”可关闭当前未被选择通道的“数字输入”。但 PC1 和 PB7 无 ANSELAx 控制，无法关闭“数字输入”，因此不应仅在部分时间作为 LVD 输入
6. /MCLR 使能：PC0 的弱上拉功能自动使能 (忽略 WPUC[0])；读 PORTC[0] 的值为“0”。
7. 对 PORTx 数据输出寄存器进行写操作，I/O 端口将输出相应的逻辑电平。每组多达 8 个 I/O 的数据寄存器共用相同的地址，写操作实际执行‘读-修改-写’的过程，即先读取该组 PORTx 端口锁存器值 (输出或输入)，然后修改，再写回 PORTx 数据寄存器。
8. 数字输出和数字输入功能可以共存，有些应用需要同时使能数字输出和数字输入。
9. 当 TRISx = 0 时，通过软件可读取 PORTx 输出或输入锁存器 LATx 的值。
10. ODCON0x = 1：“ SPI_MISO, SPI_MOSI ”, “ I2C_SCL, I2C_SDA ”, “ USARTx_TX ”管脚开漏输出。管脚的开漏功能和内部上拉功能可以同时打开。
11. 完全复位或系统复位时，PORTx 寄存器不会复位，但 TRISx 将被重置为“1”，从而关闭输出。
12. 部分管脚输入/输出支持重映射功能，AFPx 寄存器可在多个管脚之间选择。

1.3. PORTx 功能及优先级

每个 I/O 管脚均复用了多个功能，当某管脚复用的功能模块都使能输出的情况下，就存在优先级的问题。

因为输入是连到各个功能模块的，故输入不存在优先级问题，例如 PB0 作为 GPIO 输入功能时，同时也作为 TIM2 的捕捉输入。

管脚名称	功能优先级 0	功能优先级 1	功能优先级 2	功能优先级 3	功能优先级 4	功能优先级 5	功能优先级 6	功能优先级 7
PA0	PA0	SPI_MISO	TIM1_CH1	VREFN	–	–	–	–
PA1	PA1	SPI_MOSI	TIM1_CH2	VREFP	–	–	–	–
PA2	PA2	TIM1_CH4	I2C_SCL	USART1_RX	ISPCLK (调试模式)	–	–	–
PA3	PA3	AN1	TIM2_CH3	USART2_TX	–	–	–	–
PA4	PA4	AN2	TIM2_CH2	ADC_ETR	USART2_RX	–	–	–
PA5	PA5	LVDOUT	TIM2_CH1	USART1_CK	OP0+	–	–	–
PA6	PA6	AN3	USART1_TX	OP0-	–	–	–	–
PA7	PA7	AN4	CLKO	TIM1_CH4	USART1_RX	ELVD0	OP0OUT	UASRT2_TX
PB0	PB0	TIM1_CH2	SPI_SCK	TIM1_CH3N	TIM2_CH1	–	–	–
PB1	PB1	AN0	TIM1_CH4	CLKO	TIM1_CH2N	–	–	–
PB2	PB2	TIM1_CH3	SPI_SCK	TIM1_CHIN	–	–	–	–
PB3	PB3	LVDOUT	I2C_SCL	ADC_ETR	TIM1_ETR	USART2_CK	–	–
PB4	PB4	LVDOUT	I2C_SDA	TIM1_BKIN	USART2_TX	–	–	–
PB5	PB5	LVDOUT	TIM2_CH3	SPI_NSS	USART2_RX	–	–	–
PB6	PB6	AN6	I2C_SDA	USART1_TX	ISPDAT (调试模式)	–	–	–
PB7	PB7	SPI_MOSI	ELVD3	OSC2 (XT 模式)	–	–	–	–
PC0	PC0	AN5	MCLRB (复位脚)	TIM1_CH1N	ELVD1	–	–	–
PC1	PC1	SPI_MISO	ELVD2	OSC1 (XT 模式)	–	–	–	–

表 1-9 IO 功能优先级

注：

A~C 版芯片的输出映射有以下限制，使用时应注意：

TIM1_CH2 捕捉输入没有映射到 PB0，即在 PB0 管脚上：TIM1_CH2 只具备比较输出和 PWM 输出；

对于 D 版芯片，则无该限制。

1.4. PORT 端口变化中断

所有 I/O 都可选择为外部中断源 (通过 EPS0 和 EPS1 设置)， 同一时刻最多只有 8 个 IO 可以作为外部中断管脚，中断类型可通过寄存器 ITYPE0 和 ITYPE1 设置。它们具备以下特性：

- 上升沿中断
- 下降沿中断
- 双边沿中断
- 低电平中断

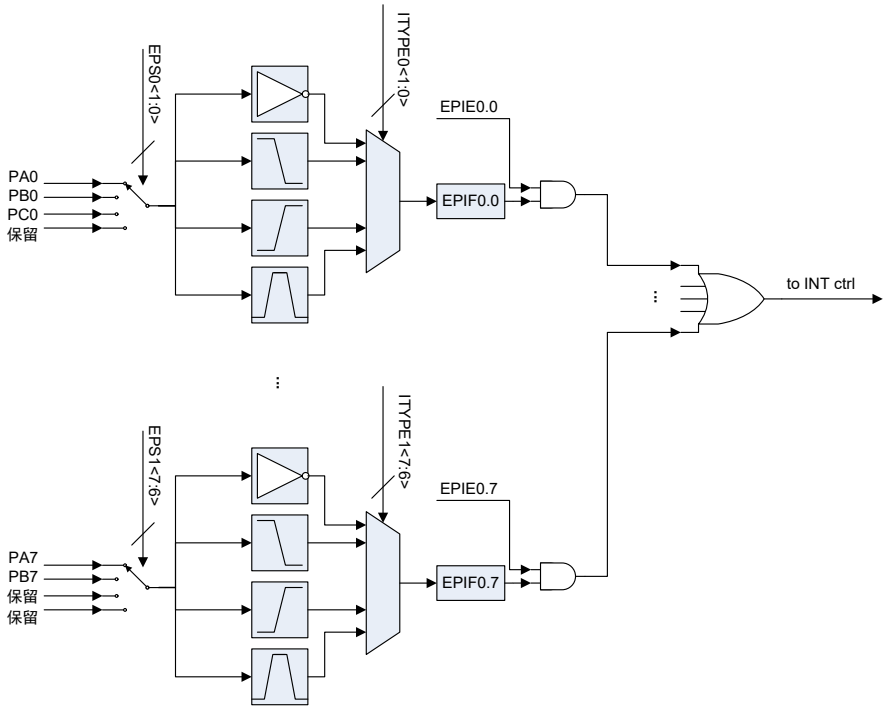

图 1-2 端口变化中断结构框图

1.5. 关于读端口 PORTx

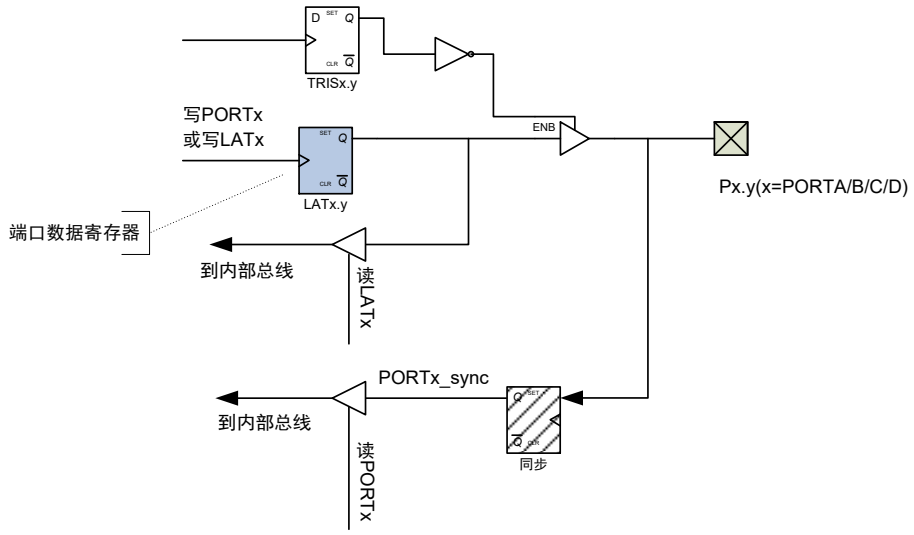

图 1-3 端口读操作原理框图

在该系列芯片中，操作 GPIO 有两种方式：访问 PORTx 寄存器或者 LATx 寄存器，它们有不同的 SFR 地址。

对于读操作，“读 PORTx”返回的是管脚经过同步寄存器后的值，而“读 LATx”返回的是端口数据寄存器的值；换言之，软件对端口数据寄存器写操作之后，至少要经过一个系统时钟之后，才能通过“读 PORTx”的方式得到新值，而“读 LATx”则无需等待；

对于写操作，无论是写 PORTx 还是 LATx，都是对端口数据寄存器进行写；

由于以上特性，当软件使用“读-修改-写” 指令对 PORTx 进行写操作时，需要特别注意以下情形：

```
BSR     PORTx, n    ;    对 PORTx 第 n 位置 1
BSR     PORTx, m    ;    对 PORTx 第 m 位置 1
...
```

软件期望的波形如下：

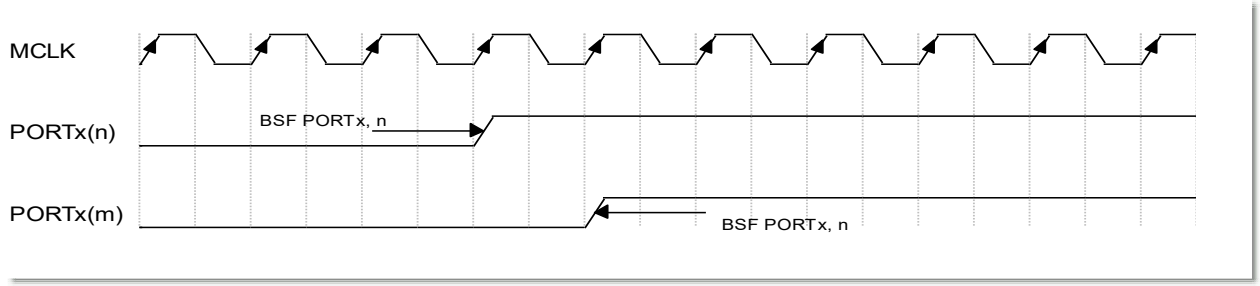

图 1-4 连续使用 RMW 指令对 PORTx 写操作的期望时序

实际输出波形如下：

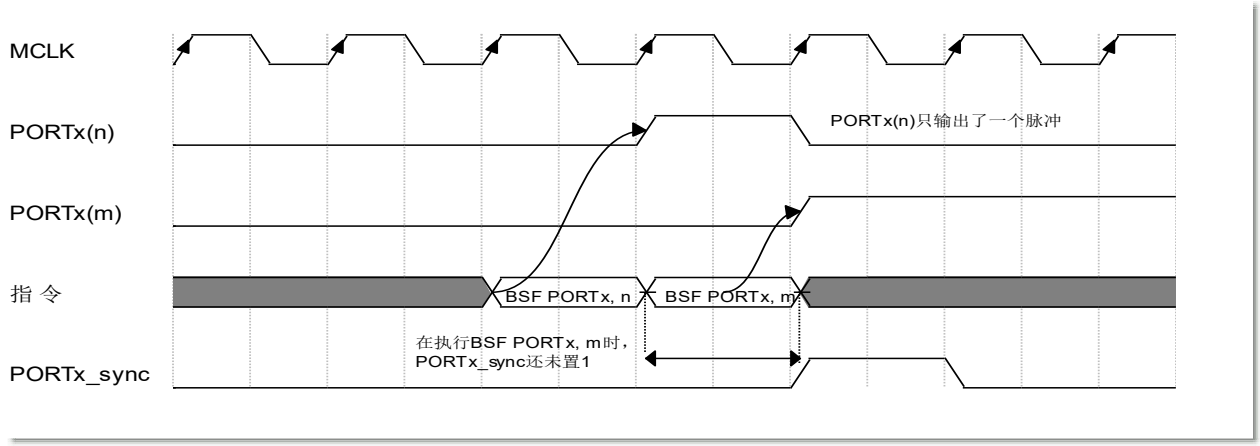

图 1-5 连续使用 RMW 指令对 PORTx 写操作的输出时序

出现这个现象的原因是在执行“BSR PORTx, m” (回顾一下 RMW 指令的执行流程：先读取 PORTx，修改数据，写 PORTx(LATx)) 时，由于同步的原因，PORTx_sync 还保持为 0，那么写回 PORTx 时刻，这一位的“0”又被写回到 LATx，导致管脚 PORTx.n 只有一个高脉冲。

有以下两种方式解决这一问题：

a) 在 PORTx 连续写操作中间插入一个 NOP；

```
BSR PORTx, n    ;    对 PORTx 第 n 位置 1
NOP             ;    插入 NOP 等待
BSR PORTx, m    ;    对 PORTx 第 m 位置 1
```

b) 或者，写操作用 LATx 寄存器而不是 PORTx；

```
BSR LATx, n     ;    直接操作端口数据寄存器 LATx
BSR LATx, m     ;    直接操作端口数据寄存器 LATx
```

注意：只有 1T 速度模式下才有该现象，不存在于其它 2T/4T 模式，原因是处于 2T/4T 模式下，执行后续指令时，PORTx_sync 已经同步到最新的值。

但如果 I/O 直接驱动 LED 或三极管，或者其它会导致 I/O 被拉低的电路时，无论是在何种速度模式下，软件读到的 PORTx 值将是 0，这种情况下，对 I/O 数据寄存器只能使用 LATx，而不能是 PORTx。

2. 应用范例

```
//*********************************************************
/* 文件名：TEST_64F0Ax_IO.c
* 功能：    FT64F0Ax_IO 功能演示
*  IC：     FT64F0A5 TSSOP20
* 内部：    16M/2T
* 说明：    当 DemoPortIn 悬空或者高电平时，DemoPortOut 输出 50Hz 占空比 50%的波形
*           当 DemoPortIn 接地时，DemoPortOut 输出高电平
*
*             FT64F0A5   TSSOP20
*                 ------------------------
* NC-------------|1(PA5)     (PA4)20|----------------NC
* NC-------------|2(PA6)     (PA3)19|----------------NC
* NC-------------|3(PA7)     (PA2)18|----------------NC
* NC-------------|4(PC0)     (PA1)17|----------------NC
* DemoPortIn--|5(PC1)     (PA0)16|----------------NC
* NC-------------|6(PB7)     (PB0)15|----------------NC
* GND------------|7(GND)     (PB1)14|----------------NC
* NC-------------|8(PB6)     (PB2)13|----------------NC
* VDD------------|9(VDD)     (PB3)12|--DemoPortOut
* NC-------------|10(PB5)    (PB4)11|----------------NC
*                 ------------------------
*/
//*********************************************************
#include   "SYSCFG.h";
#include   "FT64F0AX.h";
//**********************宏定义*****************************
#define   uchar   unsigned   char
#define   uint    unsigned   int
#define   ulong   unsigned   long

#define   DemoPortOut       PB3
#define   DemoPortIn        PC1
/*-----------------------------------------------
 * 函数名：POWER_INITIAL
 * 功能：   上电系统初始化
 * 输入：   无
 * 输出：   无
 -----------------------------------------------*/
 void POWER_INITIAL(void)
 {
     OSCCON=0B01110001;          //系统时钟选择为内部振荡器 16MHz,分频比为 1:1
     INTCON=0;                   //禁止所有中断
```

```
    PORTA=0B00000000;
    PORTB=0B00000000;
    PORTC=0B00000000;

    WPUA=0B00000000;          //弱上拉的开关，0-关，1-开
    WPUB=0B00000000;
    WPUC=0B00000010;          //PC1 上拉

    WPDA=0B00000000;          //弱下拉的开关，0-关，1-开
    WPDB=0B00000000;
    WPDC=0B00000000;

    TRISA=0B00000000;         //输入输出设置，0-输出，1-输入
    TRISB=0B00000000;         //PB3-输出  PC1-输入
    TRISC=0B00000011;

    PSRC0=0B11111111;         //源电流设置最大
    PSRC1=0B11111111;
    PSRC2=0B00001111;

    PSINK0=0B11111111;        //灌电流设置最大
    PSINK1=0B11111111;
    PSINK2=0B00000011;

    ANSELA=0B00000000;        //设置对应的 IO 为数字 IO
 }
/*---------------------------------------------
 * 函数名：DelayUs
 * 功能：   短延时函数
 * 输入：   Time 延时时间长度  延时时长 Timeμs
 * 输出：   无
 ---------------------------------------------*/
 void DelayUs(uchar Time)
 {
     uchar a;
     for(a=0;a<Time;a++)
     {
         NOP();
     }
}
/*---------------------------------------------
 * 函数名：DelayMs
 * 功能：   短延时函数
 * 输入：   Time 延时时间长度  延时时长 Time ms
```

```
 * 输出：    无
 ----------------------------------------------------*/
 void DelayMs(uchar Time)
 {
      uchar a,b;
      for(a=0;a<Time;a++)
      {
           for(b=0;b<5;b++)
           {
                DelayUs(197);
           }
      }
 }
/*----------------------------------------------
 * 函数名：main
 * 功能：    主函数
 * 输入：    无
 * 输出：    无
 ----------------------------------------------------*/
void main(void)
{
     POWER_INITIAL();              //系统初始化
     while(1)
     {
          DemoPortOut=1;
          DelayMs(10);             //10ms
          if(DemoPortIn==1)        //判断输入是否为高电平
          {
               DemoPortOut=0;
          }
          DelayMs(10);
     }
}
```

联系信息

Fremont Micro Devices Corporation

#5−8, 10/F, Changhong Building
Ke-Ji Nan 12 Road, Nanshan District,
Shenzhen, Guangdong, PRC 518057

Tel: (+86 755) 8611 7811
Fax: (+86 755) 8611 7810

Fremont Micro Devices (HK) Limited

#16, 16/F, Block B, Veristrong Industrial Centre,
34−36 Au Pui Wan Street, Fotan, Shatin, Hong Kong SAR

Tel: (+852) 2781 1186
Fax: (+852) 2781 1144

http://www.fremontmicro.com