megawin
make you win

8051-Based MCU

MG82F6B08/
MG82F6B001/
MG82F6B104
Data Sheet

Version: 1.05

2023/06 version 1.05

Features

- 1-T 80C51 Central Processing Unit
  - Stack Pointer warning indicator
- **MG82F6B08 / 6B001/ 6B104** with **8K** Bytes Flash ROM & 512 Bytes EEPROM
  - ISP memory zone could be optioned as **0.5KB/1.0KB~3.5KB**
  - 512 Bytes EEPROM write/erase cycle: **100,000**
  - Flexible IAP size by software configured
  - Code protection for Flash memory access
  - Flash write/erase cycle: **20,000**
  - Flash data retention: 100 years at 25°C
  - **Default MG82F6B08 / 6B001/ 6B104 Flash space mapping**
    - AP Flash default mapping (6.5KB, 0000h~19FFh)
    - IAP Flash default mapping (Disabled)
    - ISP Flash default mapping (1.5KB, 1A00h~1FFFh), ISP Boot code
- Data RAM: **1K** Bytes
  - On-chip 256 bytes scratch-pad RAM
  - **768** bytes expanded RAM (XRAM)
  - Support page select on XRAM access
- Dual data pointer
- Interrupt controller
  - **12** sources, four-level-priority interrupt capability
  - **Two** external interrupt inputs, nINT0, nINT1 with glitch filter
  - All external interrupts support High/Low level or Rising/Falling edge trigger
- Total **7/8** (with split mode) timers in **MG82F6B08 / 6B001/ 6B104**
  - RTC Timer and WDT Timer
  - Timer 0, Timer 1, Timer 2
  - PCA0, Program Counter Array 0
  - S0 BRG
  - If Timer 2 in split mode, **MG82F6B08 / 6B001/ 6B104** has total **8** timers
- **Three** 16-bit timer/counters, Timer 0, Timer 1, Timer 2
  - Synchronous control of Run-Enable, Stop and Reload on Timer 0~2
  - New **6** operating modes in Timer 2 with 8 clock sources and 8 capture sources
  - Timer 2 can be split to two 8-bit timers
  - Clock Count Output (CCO) on T2CKO
  - Timer 0~2 support PWM mode
  - Timer 2 support Duty Capture function
- **One** Programmable 16-bit counter/timer Arrays (PCA0) with **4** Compare/PWM modules
  - PCA0 has **4** CCP (Capture/Compare/PWM) modules
  - Reloadable 16-bit base counter to support variable length PWM
  - Capture mode, 16-bit software timer mode and High speed output mode
  - Buffered capture mode to monitor narrow pulse input
  - Support POEM0
  - Variable 8/10/12/16-bit PWM mode, the PCA can be configured to:
    - Up to **4** channels un-buffered 10/12/16-bit PWM, or
    - Up to **4** channels buffered 2~8-bit PWM, or
    - Up to **2** channels buffered 9~16-bit PWM
  - PCA0 PWM module 0~3 with dead-time control, break control and central-aligned option
- 8 Inputs Keypad Interrupt (KBI)
- 10-Bit Single-ended ADC

  - Programmable throughput up to **666K** sps
  - **MG82F6B08 / 6B001/ 6B104** has **6** channel external inputs and 2 channel for internal reference voltage (IVR/1.4V) and internal VSS
  - Support window detect function on ADC result
- On-chip voltage reference (IVR14)
- Analog Comparator 0
  - Selectable internal voltage reference (IVR/1.4V) on ACNI0
  - Wake-up from power-down and idle
  - Glitch filter option and output to internal timer capture
- Enhanced UART (S0)
  - Framing Error Detection
  - Automatic Address Recognition
  - Max. UART baud rate up to **2MHz**
  - Support SPI Master in Mode 4, up to **4MHz** on SPICLK
  - Built-in baud rate generator (S0BRG) to support TX or RX on different baud rate
  - Support LIN bus protocol with auto baud rate detection in mode 5
- One Master/Slave SPI serial interface
  - Up to **2** SPI masters including S0 in mode 4
  - Support daisy-chain function in SPI slave mode
- **Two** Master/Slave two wire serial interfaces: TWI0/I2C0 and STWI (SI2C)
  - **One** Master/Slave hardware engine: TWI0/I2C0
  - Max. 1MHz on **I2C0** master mode and Max. 400KHz on **I2C0** slave mode
  - One software TWI/I2C, STWI/SI2C, Start/Stop serial interface detection (SID)
  - Multiple slave address recognition on I2C0
- Programmable Watchdog Timer (WDT), clock sourced from ILRCO or SYSCLK/12
  - One time enabled by CPU or power-on
  - Interrupt CPU or Reset CPU on WDT overflow
  - Support WDT function in power down mode (watch mode) for auto-wakeup function
- Real-Time-Clock (RTC) module, clock sourced from ECKI, ILRCO, WDTPS, WDTOF, SYSCLK or SYSCLK/12
  - Programmable interrupt period from mini-second wakeup to minute wakeup
  - 21-bit length system timer
- Beeper function
- General purpose logic (GPL/CRC)
  - Bit order reversed function
  - 16-bit CRC engine (CCITT-16 polynomial)
  - Support automatic CRC of Flash content
  - Programmable initial seed function of CRC
- On-Chip-Debug interface (OCD)
- Maximum **8/6** GPIOs in **10/8**-pin package
  - P3 can be configured to quasi-bidirectional, push-pull output, open-drain output and input only
  - P1 and P4 can be configured to open-drain output or push-pull output
  - P4.7 shared with RST
  - Programmable GPIO driving strength and driving speed
  - On chip pull-up enabled on each pin
- Clock Sources
  - Internal **16MHz/22.12MHz** oscillator (IHRCO): factory calibrated to ±2%, typical
  - Internal Low power 32KHz RC Oscillator (ILRCO)
  - External clock input (ECKI) on **P4.5** up to **24MHz**
  - Internal RC Oscillator output on **P4.5**
- Two Brown-Out Detectors

  - BOD0: detect **2.35V**
  - BOD1: selected detection level on **4.2V/3.6V/2.4V & 2.7V**
  - Interrupt CPU or reset CPU
  - Wake up CPU in Power-Down mode (BOD1)
- Multiple power control modes: idle mode, power-down mode, slow mode, sub-clock mode, RTC mode, watch mode and monitor mode.
  - All interrupts can wake up IDLE mode
  - **10** sources with **7/6 (MSOP10/SOP8)** pins to wake up Power-Down mode
  - Slow mode and sub-clock mode support low speed MCU operation
  - RTC mode supports RTC to resume CPU in power down
  - Watch mode supports WDT to resume CPU in power down
  - Monitor mode supports BOD1 to resume CPU in power down
- Operating voltage range: 2.4 V – 5.5V
- Operation frequency range: **22.12MHz** (max)
  - SYSCLK up to **24MHz** @ 2.4V – 5.5V
  - CPU up to **12MHz** @ 2.4V – 5.5V, **0 –16MHz** @ 2.7V – 5.5V
- 16-Bytes Unique ID code
- Operating Temperature:
  - Industrial (-40°C to +85°C)*
- Package Types:
  - SOP8: MG82F6B08AS8 (8K)
  - SOP8: MG82F6B001AS8 (8K)
  - SOP8: MG82F6B104AS8 (8K)
  - MSOP10: MG82F6B08AG10 (8K)

*: Tested by sampling.

List of Contents

List of Figures

List of Tables

1. General Description

The **MG82F6B08 / 6B001/ 6B104** is a single-chip microcontroller based on a high performance 1-T architecture 80C51 CPU that executes instructions in 1~7 clock cycles (about 6~7 times the rate of a standard 8051 device), and has an 8051 compatible instruction set. Therefore at the same performance as the standard 8051, the **MG82F6B08 / 6B001/ 6B104** can operate at a much lower speed and thereby greatly reduce the power consumption.

The **MG82F6B08 / 6B001/ 6B104** has **8K** bytes of embedded Flash memory for code and data. The Flash memory can be programmed either in serial writer mode (via ICP, In-Circuit Programming) or in In-System Programming mode. And, it also provides the In-Application Programming (IAP) capability. ICP and ISP allow the user to download new code without removing the microcontroller from the actual end product; IAP means that the device can write non-volatile data in the Flash memory while the application program is running. It also provides on-chip 512 bytes EEPROM for software to store the user data. The EEPROM is independent from embedded 8K bytes Flash memory and would be programmed by CPU software or ICP. There needs no external high voltage for programming due to its built-in charge-pumping circuitry.

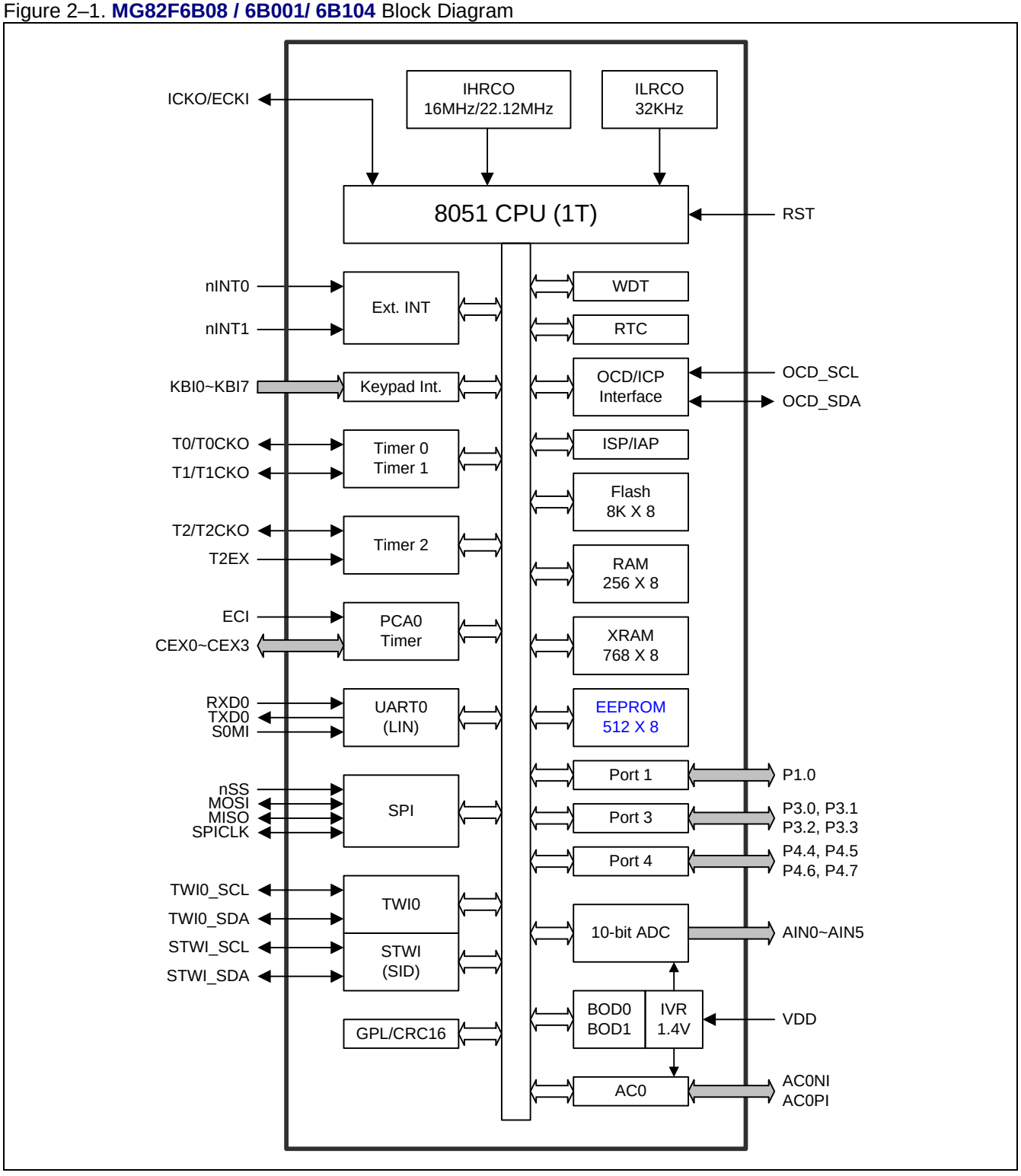
The **MG82F6B08 / 6B001/ 6B104**, **two** external interrupts with High/Low trigger option, a multi-source 4-level interrupt controller and **three** timer/counters. In addition, the **MG82F6B08 / 6B001/ 6B104** has **6** I/O pins, one XRAM of **768** bytes, **400K sps** 10-bit ADC, three 16-bit timer, one 4-channel PCA with dead-time controlled PWM, one 8-bit SPI, two TWI/I2C (TWI0/I2C0 and STWI/ SI2C), keypad interrupt, one Analog Comparators, Watchdog Timer, Real-Time-Clock module, two Brown-out Detectors, an ECKI external clock input (P4.5), an internal high precision oscillator (IHRCO), an internal low speed RC oscillator (ILRCO) and serial ports (UART0) which UART0 has enhanced serial function that facilitates multiprocessor communication, LIN bus mode and a speed improvement mechanism (X2/X4 mode).

The **MG82F6B08 / 6B001/ 6B104** has multiple operating modes to reduce the power consumption: idle mode, power down mode, slow mode, sub-clock mode, RTC mode, watch mode and monitor mode. In the Idle mode the CPU is frozen while the peripherals and the interrupt system are still operating. In the Power-Down mode the RAM and SFRs' value are saved and all other functions are inoperative; most importantly, in the Power-down mode the device can be waked up by many interrupt or reset sources. In slow mode, the user can further reduce the power consumption by using the 8-bit system clock pre-scaler to slow down the operating speed. Or select sub-clock mode which clock source is derived from internal low speed oscillator (ILRCO) for CPU to perform an ultra-low speed operation. The RTC module supports Real-Time-Clock function in all operating modes. In watch mode, it keeps WDT running in power-down or idle mode and resumes CPU as an auto-wakeup timer when WDT overflows. Monitor mode provides the Brown-Out detection in power down mode and resumes CPU when chip VDD reaches the specific detection level.

Additionally, the **MG82F6B08 / 6B001/ 6B104** is equipped with the Megawin proprietary On-Chip Debug (OCD) interface for In-Circuit Emulator (ICE). The OCD interface provides on-chip and in-system non-intrusive debugging without any target resource occupied. Several operations necessary for an ICE are supported such as Reset, Run, Stop, Step, Run to Cursor and Breakpoint Setting. The user has no need to prepare any development board during firmware developing or the socket adapter used in the traditional ICE probe head. All the thing the user needs to do is to prepare a connector for the dedicated OCD interface. This powerful feature makes the developing very easy for any user.

2. Block Diagram

Figure 2–1. **MG82F6B08 / 6B001/ 6B104** Block Diagram



3. Special Function Register

3.1. SFR Map (Page 0~F)

Table 3–1. SFR Map (Page 0~F)

		0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8	0	--	CH	CCAP0H	CCAP1H	CCAP2H	CCAP3H	--	--
	1			--	--				
	~F			--	--				
F0	0	*B*	PAOE	PCAPWM0	PCAPWM1	PCAPWM2	PCAPWM3	--	--
	1			--	--				
	~F			--	--				
E8	0	*P4*	CL	CCAP0L	CCAP1L	CCAP2L	CCAP3L	--	--
	1			--	--				
	~F			--	--				
E0	0~F	*ACC*	WDTCR	IFD	IFADRH	IFADRL	IFMT	SCMD	ISPCR
D8	0	*CCON*	CMOD	CCAPM0	CCAPM1	CCAPM2	CCAPM3	--	--
	1			--	--				
	~F			--	--				
D0	0	*PSW*	SIADR	SIDAT	SISTA	SICON	KBPATN	KBCON	KBMASK
	1		--	--	--	--			
	2		SIA2	SIA2M	--	--			
	~F		--	--	--	--			
C8	0	*T2CON*	T2MOD	RCAP2L	RCAP2H	TL2	TH2	CLRL	CHRL
	1	--	--	--	--	--	--		
	~F	--	--	--	--	--	--		
C0	0	--	XICFG	--	ADCFG0	ADCON0	ADCDL	ADCDH	CKCON0
	1		XICFG1		ADCFG1				
	2		--		ADCFG2				
	3				ADCFG3				
	4				ADCFG4				
	B		AUXIE0		ADCFG11				
	C		--		ADCFG12				
	D				ADCFG13				
	E				ADCFG14				
	F								
B8	0	*IP0L*	SADEN/ S0CR1	--	--	PWMCR	CRC0DA	RTCCR	CKCON1
	1					PDTCRA			
	~F		--			--			
B0	0	*P3*	P3M0	P3M1	P4M0	--	--	RTCTM	IP0H
	1				--	--			
	2				--	PDRVC0			
	3				--	PDRVC1			
	~F				--				
A8	0~F	*IE*	SADDR	--	--	SFRPI	EIE1	EIP1L	EIP1H
A0	0	--	AUXR0	AUXR1	AUXR2	AUXR3	--	--	--
	1					AUXR4			
	2					AUXR5			
	3					AUXR6			
	4					AUXR7			
	5					--			
	6					AUXR9			
	7					AUXR10			
	8					AUXR11			
	~F					--			
98	0	*S0CON*	S0BUF	S0BRT	S0BRC	S0CFG	S0CFG1	AC0CON	AC0MOD
	1~F	--	--	--	--	--	--	--	--
90	0	*P1*	P1M0	P1M1	--	--	--	BOREV	PCON1
	1			--	T2MOD1		TREN0		
	2			P4M1	--		TRLC0		
	3			--	--		TSPC0		
	8			P1FDC	--		--		
	9			--	--		--		
	A			P4FDC	--		--		
	~F			--	--		--		
88	0~F	*TCON*	TMOD	TL0	TL1	TH0	TH1	SFIE	XRPS
80	0~F	--	SP	DPL	DPH	SPSTAT	SPCON	SPDAT	PCON0
		0/8	**1/9**	**2/A**	**3/B**	**4/C**	**5/D**	**6/E**	**7/F**

***: User needs to set SFRPI as SFRPI=0x00 ~ 0x0F for SFR page access.**
（MCU will not keep SFRPI value in interrupt. User need to keep SFRPI value in software flow.）

SFRPI: SFR Page Index Register

SFR Page = 0~F

SFR Address = 0xAC RESET = xxxx-0000

7	6	5	4	3	2	1	0
--	--	--	--	IDX3	IDX2	IDX1	IDX0
W	**W**	**W**	**W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~4: Reserved. Software must write “0” on these bits when SFRPI is written.

Bit 3~0: SFR Page Index.

IDX[3:0]	Selected Page
0000	Page 0
0001	Page 1
0010	Page 2
0011	Page 3
……	……
……	……
……	……
1111	Page F

3.2. SFR Bit Assignment (Page 0~F)

Table 3–2. SFR Bit Assignment (Page 0~F)

SYMBOL	DESCRIPTION	ADDR (HEX)	PAGE (HEX)	BIT ADDRESS AND SYMBOL								RESET VALUE
				Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0	
SP	Stack Pointer	81	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000111
DPL	Data Pointer Low	82	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
DPH	Data Pointer High	83	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
SPSTAT	SPI Status Register	84	*0~F*	SPIF	WCOL	THRF	SPIBSY	MODF	--	--	SPR2	00000000
SPCON	SPI Control Register	85	*0~F*	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	00000100
SPDAT	SPI Data Register	86	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
PCON0	Power Control 0	87	*0~F*	SMOD1	SMOD0	GF	POF0	GF1	GF0	PD	IDL	00010000
TCON	*Timer Control*	*88*	*0~F*	*TF1*	*TR1*	*TF0*	*TR0*	*IE1*	*IT1*	*IE0*	*IT0*	*00000000*
TMOD	Timer Mode	89	*0~F*	T1GATE	T1C/T	T1M1	T1M0	T0GATE	T0C/T	T0M1	T0M0	00000000
TL0	Timer Low 0	8A	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
TL1	Timer Low 1	8B	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
TH0	Timer High 0	8C	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
TH1	Timer High 1	8D	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
SFIE	System Flag INT En.	8E	*0~F*	SIDFIE	1	1	RTCFIE	SPWIE	BOF1IE	BOF0IE	WDTFIE	01100000
XRPS	XRAM Page Select	8F	*0~F*	0	0	0	0	0	0	.1	.0	xxxxxx00
P1	*Port 1*	*90*	*0~F*	--	--	--	--	--	--	--	*P1.0*	*11111111*
P1M0	P1 Mode Register 0	91	*0~F*	--	--	--	--	--	--	--	P1M0.0	00000000
P1M1	P1 Mode Register 1	92	0	--	--	--	--	--	--	--	P1M1.0	11111111
P4M1	P4 Mode Register 1	92	2	P4M1.7	P4M1.6	P4M1.5	P4M1.4	--	--	--	--	11111111
P3FDC	P3 Fast Drv. Ctrl.	92	7	--	--	--	--	P3FDC.3	P3FDC.2	P3FDC.1	P3FDC.0	00000000
P1FDC	P1 Fast Drv. Ctrl.	92	8	--	--	--	--	--	--	--	P1FDC.0	00000000
P4FDC	P4 Fast Drv. Ctrl.	92	A	--	P4FDC.6	P4FDC.5	P4FDC.4	--	--	--	--	00000000
T2MOD1	Timer2 mode 1 Reg.	93	1	TL2CS	TF2IG	TL2IS	T2CKS	T2MS1	CP2S2	CP2S1	CP2S0	00000000
TREN0	Timer Run Enable Register 0	95	1	--	--	TR2LE	--	--	TR2E	TR1E	TR0E	00000000
TRLC0	Timer Reload Control Register 0	95	2	--	--	TL2RLC	--	--	T2RLC	T1RLC	T0RLC	00000000
TSPC0	Timer Stop Control Register 0	95	3	--	--	TL2SC	--	--	T2SC	T1SC	T0SC	00000000
BOREV	Bit Order Reversed	96	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
PCON1	Power Control 1	97	0~F	SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF	00000000
S0CON	*Serial 0 Control*	*98*	*0*	*SM00 /FE*	*SM10*	*SM20*	*REN0*	*TB80*	*RB80*	*TI0*	*RI0*	*00000000*
S0BUF	Serial 0 Buffer	99	0	.7	.6	.5	.4	.3	.2	.1	.0	xxxxxxxx
S0BRT	S0 Baud-Rate Timer	9A	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
S0BRC	S0 Baud-Rate Counter	9B	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
S0CFG	S0 Configuration	9C	0	URTS	SMOD2	URM0X3	SM30	S0DOR	BTI	UTIE	SMOD3	00001000
S0CFG1	S0 Configuration 1 (LINCFG)	9D	0	SBF0	TXER0	S0SB16	ATBR0	TXRX0	SYNC0	--	--	000000xx
AC0CON	AC0 Control Reg.	9E	0	AC0LP	AC0PDX	AC0OUT	AC0F	AC0EN	AC0INV	AC0M1	AC0M0	00x00000
AC0MOD	AC0 Mode Reg.	9F	0	0	0	0	0	NVRL	AC0FLT	0	0	00000000
AUXR0	Auxiliary Register 0	A1	*0~F*	P45OC1	P45OC0	0	PBKF	0	0	INT1H	INT0H	00000000
AUXR1	Auxiliary Register 1	A2	*0~F*	0	0	CRCDS1	CRCDS0	0	0	0	DPS	00000000
AUXR2	Auxiliary Register 2	A3	*0~F*	STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE	00000000
AUXR3	Auxiliary Register 3	A4	0	0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL	00000000
AUXR4	Auxiliary Register 4	A4	1	T2PS1	T2PS0	0	T1PS0	0	0	AC0OE	AC0FLT1	00000000
AUXR5	Auxiliary Register 5	A4	2	0	C0IC2S0	0	0	C0PS1	C0PS0	ECIPS0	C0COPS	00000000
AUXR6	Auxiliary Register 6	A4	3	0	0	0	0	0	T2FCS	SnMIPS	S0COPS	00000000
AUXR7	Auxiliary Register 7	A4	4	1	1	C0CKOE	SPI0M0	0	0	0	0	11000000
AUXR9	Auxiliary Register 9	A4	6	0	SIDPS0	T1G1	T0G1	C0FDC1	C0FDC0	0	0	00000000
AUXR10	Aux. Register 10	A4	7	1	AC0HC0	0	SPIPS0	S0PS1	SPFACE	TWICF	0	11000000
AUXR11	Aux. Register 11	A4	8	P30AM	P33AM	0	0	0	POEM0	C0M0	C0OFS	00000000
IE	*Interrupt Enable*	*A8*	*0~F*	*EA*	*0*	*ET2*	*ES0*	*ET1*	*EX1*	*ET0*	*EX0*	*00000000*
SADDR	Slave Address	A9	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
SFRPI	SFR Page Index	AC	*0~F*	--	--	--	--	IDX3	IDX2	IDX1	IDX0	xxxx0000
EIE1	Extended INT Enable 1	AD	*0~F*	EAC0	ETWI0	EKB	--	ESF	EPCA	EADC	ESPI	00000000
EIP1L	Ext. INT Priority 1 Low	AE	*0~F*	PAC0L	PTWI0L	PKBL	--	PSFL	PPCAL	PADCL	PSPIL	00000000
EIP1H	Ext. INT Priority 1 High	AF	*0~F*	PAC0H	PTWI0H	PKBH	--	PSFH	PPCAH	PADCH	PSPIH	00000000
P3	*Port 3*	*B0*	*0~F*	--	--	--	--	*P3.3*	*P3.2*	*P3.1*	*P3.0*	*11111111*
P3M0	P3 Mode Register 0	B1	*0~F*	--	--	--	--	P3M0.3	P3M0.2	P3M0.1	P3M0.0	00000000
P3M1	P3 Mode Register 1	B2	*0~F*	--	--	--	--	P3M1.3	P3M1.2	P3M1.1	P3M1.0	00000000
P4M0	P4 Mode Register 0	B3	0	P4M0.7	P4M0.6	P4M0.5	P4M0.4	--	--	--	--	10110000
PDRVC0	Port Driving Control 0	B4	2	--	P3DC0	--	--	--	P1DC0	--	--	00000000
PDRVC1	Port Driving Control 1	B4	3	0	--	--	--	--	--	P4DC1	--	00000000
RTCTM	RTC Timer Register	B6	*0~F*	RTCCS1	RTCCS0	RTCCT5	RTCCT4	RTCCT3	RTCCT2	RTCCT1	RTCCT0	01111111
IP0H	Interrupt Priority 0 High	B7	*0~F*	--	--	PT2H	PSH	PT1H	PX1H	PT0H	PX0H	00000000
IP0L	*Interrupt Priority Low*	*B8*	*0~F*	--	--	*PT2L*	*PSL*	*PT1L*	*PX1L*	*PT0L*	*PX0L*	*00000000*

SYMBOL	DESCRIPTION	ADDR (HEX)	PAGE (HEX)	BIT ADDRESS AND SYMBOL								RESET VALUE
				Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0	
SADEN	Slave Address Mask	B9	*0~F*	.7	.6	.5	.4	.3	.2	.1	.0	00000000
S0CR1	S0 Control 1	B9	*0~F*	S0TR	S0TX12	S0TCK	S0RCK	S0CKOE	ARTE	--	S0BC0	00000000
PWMCR	PWM Control Reg.	BC	0	PCAE	EXDT	PBKM	PBKE1.1	PBKE1.0	PBKE0.2	PBKE0.1	PBKE0.0	00000000
PDTCRA	PWM Dead-Time Control Reg. –A	BC	1	DTPS1	DTPS0	DT.5	DT.4	DT.3	DT.2	DT.1	DT.0	00000000
CRC0DA	CRC0 Data Port	BD	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
RTCCR	RTC Control Reg.	BE	0~F	RTCE	RTCO	RTCRL5	RTCRL4	RTCRL3	RTCRL2	RTCRL1	RTCRL0	00111111
CKCON1	Clock Control 1	BF	0~F	--	--	--	--	OSCSTA3	OSCSTA2	OSCSTA1	OSCSTA0	00000000
XICFG	Ext. INT. Configured	C1	0	INT1IS1	INT1IS0	INT0IS1	INT0IS0	--	--	X1FLT	X0FLT	00000000
XICFG1	Ext. INT. Configured 1	C1	1	--	--	--	--	--	--	X1FLT1	X0FLT1	00000000
AUXIE0	Auxiliary INT Enable 0	C1	B	0	0	EPIE	CFIE	0	0	0	0	00000000
ADCFG0	ADC Configuration 0	C3	0	ADCKS2	ADCKS1	ADCKS0	ADRJ	ACHS	SMPF	ADTM1	ADTM0	00000000
ADCFG1	ADC Configuration 1	C3	1	IGADCI	EADCWI	SMPFIE	SIGN	AOS.3	AOS.2	AOS.1	AOS.0	00000000
ADCFG2	ADC Configuration 2	C3	2	SHT.7	SHT.6	SHT.5	SHT.4	SHT.3	SHT.2	SHT.1	SHT.0	00000000
ADCFG3	ADC Configuration 3	C3	3	ADPS1	ADPS0	--	--	ARES1	ARES0	--	--	01000000
ADCFG4	ADC Configuration 4	C3	4	--	ADWM0	ADTM3	ADTM2	--	--	--	--	00000000
ADCFG11	ADC Configuration 11	C3	B	WHB.3	WHB.2	WHB.1	WHB.0	1	1	1	1	11111111
ADCFG12	ADC Configuration 12	C3	C	WHB.11	WHB.10	WHB.9	WHB.8	WHB.7	WHB.6	WHB.5	WHB.4	11111111
ADCFG13	ADC Configuration 13	C3	D	WLB.3	WLB.2	WLB.1	WLB.0	0	0	0	0	00000000
ADCFG14	ADC Configuration 14	C3	E	WLB.11	WLB.10	WLB.9	WLB.8	WLB.7	WLB.6	WLB.5	WLB.4	00000000
ADCON0	ADC Control 0	C4	0~F	ADCEN	ADCWI	CHS3	ADCI	ADCS	CHS2	CHS1	CHS0	00000000
ADCDL	ADC Data Low	C5	0~F	ADCV.3	ADCV.2	ADCV.1	ADCV.0	--	--	--	--	0000xxxx
ADCDH	ADC Data High	C6	0~F	ADCV.11	ADCV.10	ADCV.9	ADCV.8	ADCV.7	ADCV.6	ADCV.5	ADCV.4	00000000
CKCON0	Clock Control 0	C7	0~F	AFS	--	--	--	CCKS	SCKS2	SCKS1	SCKS0	00010000
T2CON	*Timer 2 Control Reg.*	*C8*	*0*	*TF2*	*EXF2*	*RCLK/ TF2L*	*TCLK/ TL2IE*	*EXEN2*	*TR2*	*C/T2*	*CP/RL2*	*00000000*
T2MOD	Timer 2 mode Reg.	C9	0	T2SPL	TL2X12/ T2EIP	T2EXH	T2X12	TR2L	TR2LC	T2OE	T2MS0	00000000
RCAP2L	Timer2 Capture Low	CA	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
RCAP2H	Timer2 Capture High	CB	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
TL2	Timer Low 2	CC	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
TH2	Timer High 2	CD	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CLRL	CL Reload register	CE	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CHRL	CH Reload register	CF	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
PSW	*Program Status Word*	*D0*	0~F	*CY*	*AC*	*F0*	*RS1*	*RS0*	*OV*	*F1*	*P*	*00000000*
SIADR	TWI0 Address Reg.	D1	0	.7	.6	.5	.4	.3	.2	.1	GC	00000000
SIA2	TWI0 2nd Addr Reg.	D1	2	.7	.6	.5	.4	.3	.2	.1	A2E	00000000
SIDAT	TWI0 Data Reg.	D2	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
SIA2M	SIA2 Mask Reg.	D2	2	SIA2M.7	SIA2M.6	SIA2M.5	SIA2M.4	SIA2M.3	SIA2M.2	SIA2M.1	1	11111111
SISTA	TWI0 Status Reg.	D3	0	.7	.6	.5	.4	.3	.2	.1	.0	11111000
SICON	TWI0 Control Reg.	D4	0	CR2	ENSI	STA	STO	SI	AA	CR1	CR0	00000000
KBPATN	Keypad Pattern	D5	0~F	.7	.6	.5	.4	.3	.2	.1	.0	11111111
KBCON	Keypad Control	D6	0~F	KBCS1	KBCS0	KBES	--	0	0	PATN_ SEL	KBIF	00000000
KBMASK	Keypad Int. Mask	D7	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCON	*PCA Control Reg.*	*D8*	0~F	*CF*	*CR*	*0*	*0*	*CCF3*	*CCF2*	*CCF1*	*CCF0*	*00000000*
CMOD	PCA Mode Reg.	D9	0~F	CIDL	0	BME2	BME0	CPS2	CPS1	CPS0	ECF	00000000
CCAPM0	PCA Module0 Mode	DA	0	DTE0	ECOM0	CAPP0	CAPN0	MAT0	TOG0	PWM0	ECCF0	00000000
CCAPM1	PCA Module1 Mode	DB	0	0	ECOM1	CAPP1	CAPN1	MAT1	TOG1	PWM1	ECCF1	00000000
CCAPM2	PCA Module2 Mode	DC	0~F	DTE2	ECOM2	CAPP2	CAPN2	MAT2	TOG2	PWM2	ECCF2	00000000
CCAPM3	PCA Module3 Mode	DD	0~F	0	ECOM3	CAPP3	CAPN3	MAT3	TOG3	PWM3	ECCF3	00000000
ACC	*Accumulator*	*E0*	0~F	*ACC.7*	*ACC.6*	*ACC.5*	*ACC.4*	*ACC.3*	*ACC.2*	*ACC.1*	*ACC.0*	*00000000*
WDTCR	WDT Control register	E1	0~F	WREN	NSW	ENW	CLRW	WIDL	PS2	PS1	PS0	00000000
IFD	ISP Flash data	E2	0~F	.7	.6	.5	.4	.3	.2	.1	.0	11111111
IFADRH	ISP Flash Addr. High	E3	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
IFADRL	ISP Flash Addr. Low	E4	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
IFMT	ISP Mode Table	E5	0~F	MS.7	MS.6	MS.5	MS.4	MS.3	MS.2	MS.1	MS.0	00000000
SCMD	ISP Serial Command	E6	0~F	.7	.6	.5	.4	.3	.2	.1	.0	xxxxxxxx
ISPCR	ISP Control Register	E7	0~F	ISPEN	SWBS	SRST	CFAIL	0	0	PBSY	EEPF	00000000
P4	*Port 4*	*E8*	0~F	*P4.7*	*P4.6*	*P4.5*	*P4.4*	*1*	*1*	*1*	*1*	*11111111*
CL	PCA base timer Low	E9	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP0L	PCA module0 capture Low	EA	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP1L	PCA module1 capture Low	EB	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP2L	PCA module2 capture Low	EC	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP3L	PCA module3 capture Low	ED	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
B	*B Register*	*F0*	0~F	*B.7*	*B.6*	*B.5*	*B.4*	*B.3*	*B.2*	*B.1*	*B.0*	*00000000*
PAOE	PWM Additional Output Enable	F1	0~F	POE3	0	0	POE2	POE1	0	0	POE0	10011001
PCAPWM0	PCA PWM0 Mode	F2	0	P0RS1	P0RS0	0	0	0	P0INV	ECAP0H	ECAP0L	00000000
PCAPWM1	PCA PWM1 Mode	F3	0	P1RS1	P1RS0	0	0	0	P1INV	ECAP1H	ECAP1L	00000000

SYMBOL	DESCRIPTION	ADDR (HEX)	PAGE (HEX)	BIT ADDRESS AND SYMBOL								RESET VALUE
				Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0	
PCAPWM2	PCA PWM2 Mode	F4	0~F	P2RS1	P2RS0	0	0	0	P2INV	ECAP2H	ECAP2L	00000000
PCAPWM3	PCA PWM3 Mode	F5	0~F	P3RS1	P3RS0	0	0	0	P3INV	ECAP3H	ECAP3L	00000000
CH	PCA base timer High	F9	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP0H	PCA Module0 capture High	FA	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP1H	PCA Module1 capture High	FB	0	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP2H	PCA Module2 capture High	FC	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000
CCAP3H	PCA Module3 capture High	FD	0~F	.7	.6	.5	.4	.3	.2	.1	.0	00000000

3.3. Auxiliary SFR Map (Page P)

MG82F6B08 / 6B001/ 6B104 has an auxiliary SFR page which is indexed by page P and the SFRs' write is a different way from standard 8051 SFR page. The registers in auxiliary SFR map are addressed by IFMT and SCMD like ISP/IAP access flow. Page P has 256 bytes space that can target to **11 physical bytes** and **5 logical bytes.** The 11 physical bytes include IAPLB, CKCON2, CKCON3, CKCON4, PCON2, PCON3, PCON4, SPCON0, DCON0, RTCTM and RTCCR. The 6 logical bytes include PCON0, PCON1, CKCON0, WDTCR and P4. Access on the 5 logical bytes gets the coherence content with the same SFR in Page 0~F. Please refer Section "28 Page P SFR Access" for more detail information.

Table 3–3. Auxiliary SFR Map (Page P)

		0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8		--	--	--	--	--	--	--	--
F0		--	--	--	--	--	--	--	--
E8		P4	--	--	--	--	--	--	--
E0		--	WDTCR	--	--	--	--	--	--
D8		--	--	--	--	--	--	--	--
D0		--	--	--	--	--	--	--	--
C8		--	--	--	--	--	--	--	--
C0		--	--	--	--	--	--	--	CKCON0
B8		--	--	--	--	--	--	--	--
B0		--	--	--	--	--	--	--	--
A8		--	--	--	--	--	--	--	--
A0		--	--	--	--	--	--	--	--
98		--	--	--	--	--	--	--	--
90		--	--	--	--	--	--	--	PCON1
88		--	--	--	--	--	--	--	--
80		--	--	--	--	--	--	--	PCON0
78		--	--	--	--	--	--	--	--
70		--	--	--	--	--	--	--	--
68		--	--	--	--	--	--	--	--
60		--	--	--	--	--	--	--	--
58		--	--	--	--	--	--	--	--
50		--	--	--	SPHB	RTCCR	RTCTM	--	--
48		SPCON0	--	--	--	DCON0	--	--	--
40		CKCON2	CKCON3	CKCON4	--	PCON2	PCON3	PCON4	--
38		--	--	--	--	--	--	--	--
30		--	--	--	--	--	--	--	--
28		--	--	--	--	--	--	--	--
20		--	--	--	--	--	--	--	--
18		--	--	--	--	--	--	--	--
10		--	--	--	--	--	--	--	--
08		--	--	--	--	--	--	--	--
00		--	--	--	IAPLB	--	--	--	--
		0/8	**1/9**	**2/A**	**3/B**	**4/C**	**5/D**	**6/E**	**7/F**

3.4. Auxiliary SFR Bit Assignment (Page P)

Table 3–4. Auxiliary SFR Bit Assignment (Page P)

SYMBOL	DESCRIPTION	ADDR	BIT ADDRESS AND SYMBOL								RESET VALUE
			Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0	
Physical Bytes											
IAPLB	IAP Low Boundary	03H	IAPLB6	IAPLB5	IAPLB4	IAPLB3	IAPLB2	IAPLB1	IAPLB0	0	
CKCON2	Clock Control 2	40H	0	1	0	IHRCOE	0	0	OSCS1	OSCS0	01010000
CKCON3	Clock Control 3	41H	WDTCS1	WDTCS0	FWKP	WDTFS	MCKD1	MCKD0	1	0	00000110
CKCON4	Clock Control 4	42H	RCSS2	RCSS1	RCSS0	RPSC2	RPSC1	RPSC0	RTCCS3	RTCCS2	00000000
PCON2	Power Control 2	44H	AWBOD1	0	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	1	0000x1x1
PCON3	Power Control 3	45H	IVREN	0	0	SPWRE	0	0	0	0	00000000
PCON4	Power Control 4	46H	0	0	0	BO1S2	0	0	0	0	00000000
SPCON0	SFR Page Control 0	48H	0	0	P4CTL	WRCTL	0	CKCTL0	PWCTL1	PWCTL0	00000000
DCON0	Device Control 0	4CH	HSE	IAPO	0	0	0	IORCTL	RSTIO	OCDE	10000011
SPHB	SP High Boundary	53H	1	1	1	1	SPHB.3	SPHB.2	SPHB.1	SPHB.0	11111111
RTCCR	RTC Control Reg.	54H	RTCE	RTCO	RTCRL5	RTCRL4	RTCRL3	RTCRL2	RTCRL1	RTCRL0	00111111
RTCTM	RTC Timer Register	55H	RTCCS1	RTCCS0	RTCCT5	RTCCT4	RTCCT3	RTCCT2	RTCCT1	RTCCT0	01111111
Logical Bytes											
PCON0	Power Control 0	87H	SMOD1	SMOD0	GF	POF0	GF1	GF0	PD	IDL	00010000
PCON1	Power Control 1	97H	SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF	00000000
CKCON0	Clock Control 0	C7H	AFS	0	0	1	CCKS	SCKS2	SCKS1	SCKS0	00010000
WDTCR	Watch-dog-timer Control register	E1H	WREN	NSW	ENW	CLRW	WIDL	PS2	PS1	PS0	00000000
P4	Port 4	E8H	P4.7	P4.6	P4.5	P4.4	1	1	1	1	11111111

Sample Code of Page-P SFR write：

```
Check PBSY =0;
IFADRH = 0x00;
ISPCR = ISPEN;                  //enable IAP/ISP
IFMT = MS2;                     // Page-P write，IFMT =0x04
IFADRL = SPCON0;                //Set Page-P SFR address
IFD |= CKCTL0;                  // set CKCTL0
SCMD = 0x46;                    //
SCMD = 0xB9;                    //
IFMT = Flash_Standby;           // IAP/ISP standby，IFMT =0x00
ISPCR &= ~ISPEN;
```

4. Pin Configurations

4.1. Package Instruction

Figure 4–1. **MG82F6B08AG10** MSOP10 Top View

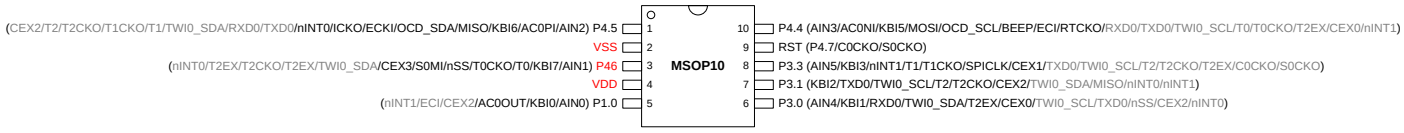


Figure 4–2. **MG82F6B08AS8** SOP8 Top View

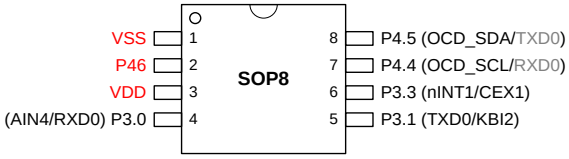


Figure 4–3. **MG82F6B001AS8** SOP8 Top View

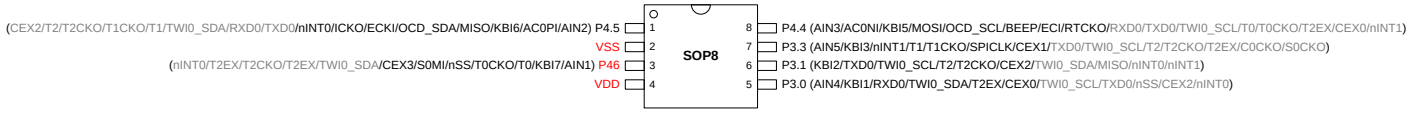
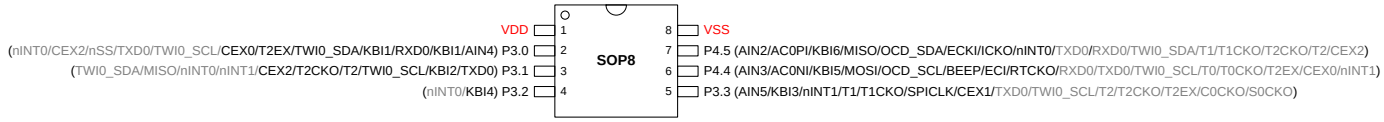


Figure 4–4. **MG82F6B104AS8** SOP8 Top View (Pin Compatible with MG86FE/L104)



4.2. Pin Description

Table 4–1. **MG82F6B08** Pin Description

MNEMONIC	PIN NUMBER		I/O TYPE	DESCRIPTION
	10-Pin MSOP	8-Pin SOP		
P1.0 (AIN0) (KBI0) (AC0OUT)	5	--	I/O	* Port 1.0. * AIN0: ADC channel-0 analog input. * KBI0: keypad input 0. * AC0OUT: Analog Comparator 0 output.
P3.0 (AIN4) (KBI1) (RXD0) (TWI0_SDA) (T2EX) (CEX0)	6	4	I/O	* Port 3.0. * AIN4: ADC channel-4 analog input. * KBI1: keypad input 1. * RXD0: UART0 serial input port. * TWI0_SCL: serial clock of TWI0/I2C0. * T2EX: Timer/Counter 2 external control input. * CEX0: PCA0 module-0 external I/O.
P3.1 (KBI2) (TXD0) (TWI0_SCL) (T2) (T2CKO) (CEX2)	7	5	I/O	* Port 3.1. * KBI2: keypad input 2. * TXD0: UART0 serial output port. * TWI0_SCL: serial clock of TWI0/I2C0. * T2: Timer/Counter 2 external clock input. * T2CKO: Timer 2 programmable clock output. * CEX2: PCA0 module-2 external I/O.
P3.3 (AIN5) (KBI3) (nINT1) (T1) (T1CKO) (SPICLK) (CEX1)	8	6	I/O	* Port 3.3. * AIN5: ADC channel-5 analog input. * KBI3: keypad input 3. * nINT1: external interrupt 1 input. * T1: Timer/Counter 1 external input. * T1CKO: Timer 1 programmable clock output. * SPICLK: SPI clock, output for master and input for slave. * CEX1: PCA0 module-1 external I/O.
P4.4 (AIN3) (KBI5) (AC0NI) (MOSI) (OCD_SCL) (BEEP) (ECI) (RTCKO)	10	7	I/O	* Port 4.4. * AIN3: ADC channel-3 analog input. * KBI5: keypad input 5. * AC0NI: Analog Comparator 0 negative input. * MOSI: SPI master out & slave in. * OCD_SCL: OCD interface, serial clock. * BEEP: Beeper output. * ECI: PCA external clock input. * RTCKO: RTC programmable clock output.
P4.5 (AIN2) (KBI6) (AC0PI) (MISO) (OCD_SDA) (ECKI) (ICKO) (nINT0)	1	8	I/O	* Port 4.5. * AIN2: ADC channel-2 analog input. * KBI6: keypad input 6. * AC0PI: Analog Comparator 0 positive input. * MISO: SPI master in & slave out. * OCD_SDA: OCD interface, serial data. * ECKI: In external clock input mode, this is clock input pin. * ICKO: Internal Clock (MCK) Output. * nINT0: external interrupt 0 input.
P4.6 (AIN1) (KBI7) (T0) (T0CKO) (nSS) (S0MI) (CEX3)	3	2	I/O	* Port 4.6. * AIN1: ADC channel-1 analog input. * KBI7: keypad input 7. * T0: Timer/Counter 0 external clock input. * T0CKO: Timer 0 programmable clock output. * nSS: SPI slave select. * S0MI: Serial Port 0 SPI Master mode data Input. * CEX3: PCA0 module-3 external I/O.
RST (P4.7) (C0CKO) (S0CKO)	9	--	I O	* RST: External RESET input, high active. * Port 4.7 output only. *Note: When P4.7/RST use as port pin, it is not suggest to program it as Input to avoid MCU is locked in reset in bootup period when level high send into this pin.* * C0CKO: Programmable clock output of PCA base counter. * S0CKO: S0BRT programmable clock output.
VDD	4	3	P	Power supply input.
VSS	2	1	G	Ground, 0 V reference.

Table 4–2. **MG82F6B001** Pin Description

MNEMONIC	PIN NUMBER 8-Pin SOP	I/O TYPE	DESCRIPTION
P3.0 (AIN4) (KBI1) (RXD0) (TWI0_SDA) (T2EX) (CEX0)	5	I/O	* Port 3.0. * AIN4: ADC channel-4 analog input. * KBI1: keypad input 1. * RXD0: UART0 serial input port. * TWI0_SCL: serial clock of TWI0/I2C0. * T2EX: Timer/Counter 2 external control input. * CEX0: PCA0 module-0 external I/O.
P3.1 (KBI2) (TXD0) (TWI0_SCL) (T2) (T2CKO) (CEX2)	6	I/O	* Port 3.1. * KBI2: keypad input 2. * TXD0: UART0 serial output port. * TWI0_SCL: serial clock of TWI0/I2C0. * T2: Timer/Counter 2 external clock input. * T2CKO: Timer 2 programmable clock output. * CEX2: PCA0 module-2 external I/O.
P3.3 (AIN5) (KBI3) (nINT1) (T1) (T1CKO) (SPICLK) (CEX1)	7	I/O	* Port 3.3. * AIN5: ADC channel-5 analog input. * KBI3: keypad input 3. * nINT1: external interrupt 1 input. * T1: Timer/Counter 1 external input. * T1CKO: Timer 1 programmable clock output. * SPICLK: SPI clock, output for master and input for slave. * CEX1: PCA0 module-1 external I/O.
P4.4 (AIN3) (KBI5) (AC0NI) (MOSI) (OCD_SCL) (BEEP) (ECI) (RTCKO)	8	I/O	* Port 4.4. * AIN3: ADC channel-3 analog input. * KBI5: keypad input 5. * AC0NI: Analog Comparator 0 negative input. * MOSI: SPI master out & slave in. * OCD_SCL: OCD interface, serial clock. * BEEP: Beeper output. * ECI: PCA external clock input. * RTCKO: RTC programmable clock output.
P4.5 (AIN2) (KBI6) (AC0PI) (MISO) (OCD_SDA) (ECKI) (ICKO) (nINT0)	1	I/O	* Port 4.5. * AIN2: ADC channel-2 analog input. * KBI6: keypad input 6. * AC0PI: Analog Comparator 0 positive input. * MISO: SPI master in & slave out. * OCD_SDA: OCD interface, serial data. * ECKI: In external clock input mode, this is clock input pin. * ICKO: Internal Clock (MCK) Output. * nINT0: external interrupt 0 input.
P4.6 (AIN1) (KBI7) (T0) (T0CKO) (nSS) (S0MI) (CEX3)	3	I/O	* Port 4.6. * AIN1: ADC channel-1 analog input. * KBI7: keypad input 7. * T0: Timer/Counter 0 external clock input. * T0CKO: Timer 0 programmable clock output. * nSS: SPI slave select. * S0MI: Serial Port 0 SPI Master mode data Input. * CEX3: PCA0 module-3 external I/O.
VDD	4	P	Power supply input.
VSS	2	G	Ground, 0 V reference.

Table 4–3. **MG82F6B104** Pin Description

MNEMONIC	PIN NUMBER 8-Pin SOP	I/O TYPE	DESCRIPTION
P3.0 (AIN4) (KBI1) (RXD0) (TWI0_SDA) (T2EX) (CEX0)	2	I/O	* Port 3.0. * AIN4: ADC channel-4 analog input. * KBI1: keypad input 1. * RXD0: UART0 serial input port. * TWI0_SCL: serial clock of TWI0/I2C0. * T2EX: Timer/Counter 2 external control input. * CEX0: PCA0 module-0 external I/O.
P3.1 (KBI2) (TXD0) (TWI0_SCL) (T2) (T2CKO) (CEX2)	3	I/O	* Port 3.1. * KBI2: keypad input 2. * TXD0: UART0 serial output port. * TWI0_SCL: serial clock of TWI0/I2C0. * T2: Timer/Counter 2 external clock input. * T2CKO: Timer 2 programmable clock output. * CEX2: PCA0 module-2 external I/O.
P3.2 (KBI4)	4	I/O	* Port 3.2. * KBI4: keypad input 4.
P3.3 (AIN5) (KBI3) (nINT1) (T1) (T1CKO) (SPICLK) (CEX1)	5	I/O	* Port 3.3. * AIN5: ADC channel-5 analog input. * KBI3: keypad input 3. * nINT1: external interrupt 1 input. * T1: Timer/Counter 1 external input. * T1CKO: Timer 1 programmable clock output. * SPICLK: SPI clock, output for master and input for slave. * CEX1: PCA0 module-1 external I/O.
P4.4 (AIN3) (KBI5) (AC0NI) (MOSI) (OCD_SCL) (BEEP) (ECI) (RTCKO)	6	I/O	* Port 4.4. * AIN3: ADC channel-3 analog input. * KBI5: keypad input 5. * AC0NI: Analog Comparator 0 negative input. * MOSI: SPI master out & slave in. * OCD_SCL: OCD interface, serial clock. * BEEP: Beeper output. * ECI: PCA external clock input. * RTCKO: RTC programmable clock output.
P4.5 (AIN2) (KBI6) (AC0PI) (MISO) (OCD_SDA) (ECKI) (ICKO) (nINT0)	7	I/O	* Port 4.5. * AIN2: ADC channel-2 analog input. * KBI6: keypad input 6. * AC0PI: Analog Comparator 0 positive input. * MISO: SPI master in & slave out. * OCD_SDA: OCD interface, serial data. * ECKI: In external clock input mode, this is clock input pin. * ICKO: Internal Clock (MCK) Output. * nINT0: external interrupt 0 input.
VDD	1	P	Power supply input.
VSS	8	G	Ground, 0 V reference.

4.3. Alternate Function Redirection

Many I/O pins, in addition to their normal I/O function, also serve the alternate function for internal peripherals. For the digital peripherals, all GPIOs serve the alternate function in the default state. However, the user may set the corresponding control bits in AXUR0~AUXR3 to serve their alternate function on the relocated ports.

AUXR0: Auxiliary Register 0

SFR Page = 0~F
SFR Address = 0xA1 RESET = 0000-0000

7	6	5	4	3	2	1	0
P45OC1	P45OC0	0	PBKF	0	0	INT1H	INT0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: P4.5 function configured control bit 1 and 0. The two bits only act when internal RC oscillator (IHRCO or ILRCO) is selected for system clock source. In external clock input mode, P4.5 is the dedicated clock input pin. In internal oscillator condition, P4.5 provides the following selections for GPIO or clock source generator. When P45OC[1:0] index to non-P4.5 GPIO function, P4.5 will drive the on-chip RC oscillator output to provide the clock source for other devices.

P45OC[1:0]	P4.5 function	I/O mode
0 0	P4.5	By P4M1.5 & P4M0.5
0 1	MCK	By P4M1.5 & P4M0.5
1 0	MCK/2	By P4M1.5 & P4M0.5
1 1	MCK/4	By P4M1.5 & P4M0.5

Please refer Section "8 System Clock" to get the more detailed clock information. For clock-out on P4.5 function, it is recommended to set {P4M1.5, P4M0.5} to "01" which selects P4.5 as push-push output mode.

AUXR1: Auxiliary Control Register 1

SFR Page = 0~F
SFR Address = 0xA2 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	CRCDS1	CRCDS0	0	0	0	DPS
W	W	R/W	R/W	W	W	W	R/W

AUXR2: Auxiliary Register 2

SFR Page = 0~F
SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	W	W	R/W	R/W	R/W	R/W

AUXR3: Auxiliary Register 3

SFR Page = **0 only**
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: Reserved. Software must write "**0**" on these bits when AUXR3 is written.

Bit 6: T0PS0, Timer 0 Port pin Selection.

T0PS0	T0/T0CKO
0	P4.6
1	P4.4

Bit 5~4: BPOC1~0, Beeper output control bits.

BPOC[1:0]	P4.4 function	I/O mode
0 0	P4.4	By P4M0.4 & P4M1.4
0 1	ILRCO/32	By P4M0.4 & P4M1.4
1 0	ILRCO/16	By P4M0.4 & P4M1.4
1 1	ILRCO/8	By P4M0.4 & P4M1.4

For beeper on P4.4 function, it is recommended to configure P4.4 as push-push output mode.

Bit 3: S0PS0, Serial Port 0 pin Selection 0. (S0PS1 at AUXR10.3)

S0PS1~0	RXD0	TXD0
0 0	P3.0	P3.1
0 1	P4.4	P4.5
1 0	P4.5	P3.0
1 1	P4.5	P4.4

Bit 2~1: TWIPS1~0, TWI0/I2C0 Port pin Selection [1:0].

TWIPS1~0	TWI0_SCL	TWI0_SDA
0 0	P3.1	P3.0
0 1	P4.4	P4.5
1 0	P3.0	P3.1
1 1	P3.3	P4.6

AUXR4: Auxiliary Register 4

SFR Page = **1 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
T2PS1	T2PS0	0	T1PS0	0	0	AC0OE	AC0FLT1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: T2PS1~0, Timer 2 Port pin Selection [1:0].

T2PS1~0	T2/T2CKO	T2EX
0 0	P3.1	P3.0
0 1	P3.3	P4.6
1 0	P4.6	P3.3
1 1	P4.5	P4.4

Bit 5: Reserved. Software must write "**0**" on these bits when AUXR4 is written.

Bit 4: T1PS1~0, Timer 1 Port pin Selection.

T1PS0	T1/T1CKO
0	P3.3
1	P4.5

Bit 1: AC0OE, AC0OUT output enable on port pin.
0: Disable AC0OUT output on port pin.
1: Enable AC0OUT output on **P1.0.**

AUXR5: Auxiliary Register 5

SFR Page = **2 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	C0IC2S0	0	0	C0PS1	C0PS0	ECIPS0	C0COPS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 6: C0IC2S0, PCA0 Input Channel 2 input port pin Selection.

C1IC2S0	CEX2 input
0	CEX2 Port Pin
1	T2EXI

Bit 3~2: C0PS[1:0], PCA0 Port pin Selection 1 & 0.

C0PS1~0	CEX0	CEX1	CEX2	CEX3
0 0	P3.0	P3.3	P3.1	P4.6
0 1	P4.4	P3.3	P4.5	P4.6
1 0	P4.4	P3.3	P3.0	P4.6
1 1	P4.4	P3.3	P1.0	P4.6

Bit 1: ECIPS0, PCA0 ECI Port pin Selection0.

ECIPS0	ECI
0	P4.4
1	P1.0

Bit 0: C0COPS, PCA0 Clock Output (C0CKO) port pin Selection.

C0COPS	C0CKO
0	P4.7
1	P3.3

AUXR6: Auxiliary Register 6

SFR Page = **3 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	T2FCS	SnMIPS	S0COPS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 2: T2FCS, Reserved for chip test.

Bit 0: S0COPS, S0BRG Clock Output (S0CKO) port pin Selection.

S0COPS	S0CKO
0	P4.7
1	P3.3

AUXR7: Auxiliary Register 7

SFR Page = **4 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	1	C0CKOE	SPI0M0	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 5: C0CKOE, PCA0 clock output (C0CKO) enable.
0: Disable PCA0 clock output.
1: Enable PCA0 clock output with PCA0 base timer overflow rate/2.

AUXR9: Auxiliary Register 9

SFR Page = 6 only

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	SIDPS0	T1G1	T0G1	C0FDC1	C0FDC0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 6: SID/STWI Port pin Selection.

SIDPS0	STWI_SCL	STWI_SDA
0	nINT1	S0MI
1	TWI0_SCL	TWI0_SDA

Bit 5: T1G1, Gating source selection of Timer 1.

T1G1, T1GATE	T1 Gate source
0 0	Disable
0 1	INT1 active
1 0	TF2 active
1 1	KBI active

Bit 4: T0G1, Gating source selection of Timer 0.

T0G1, T0GATE	T0 Gate source
0 0	Disable
0 1	INT0 active
1 0	TF2 active
1 1	KBI active

AUXR10: Auxiliary Register 10

SFR Page = **7 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	AC0HC0	0	SPIPS0	S0PS1	SPFACE	TWICF	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: SPIPS0, SPI Port pin Selection.

SPIPS0	nSS	MOSI	MISO	SPICLK
0 0	P4.6	P4.4	P4.5	P3.3
0 1	P3.0	P4.4	P3.1	P3.3

Bit 3: S0PS1, Serial Port 0 pin Selection 1. (Its function is illustrated at AUXR3.3, S0PS0)

Bit 1: TWICF, TWI0/I2C0 serial Clock input Filter.
0: Disable TWICF function.
1: Enable TWICF function.

AUXR11: Auxiliary Register 11

SFR Page = **8 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
P30AM	P33AM	0	0	0	POEM0	C0M0	C0OFS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

XICFG: External Interrupt Configured Register

SFR Page = **0 only**

SFR Address = **0xC1** RESET = 0000-0000

7	6	5	4	3	2	1	0
INT1IS.1	INT1IS.0	INT0IS.1	INT0IS.0	0	0	X1FLT	X0FLT
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: INT1IS.1~0, nINT1 input port pin selection bits which function is defined as following table.

INT1IS.1~0	Selected Port Pin of nINT1
0 0	P3.3
0 1	P3.1
1 0	P4.4
1 1	P1.0

Bit 5~4: INT0IS.1~0, nINT0 input port pin selection bits which function is defined as following table.

INT0IS.1~0	Selected Port Pin of nINT0
0 0	P4.5
0 1	P3.0
1 0	P3.2
1 1	P4.6

XICFG1: External Interrupt Configured 1 Register

SFR Page = **1 only**

SFR Address = **0xC1** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	0	X1FLT1	X0FLT1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

5. 8051 CPU Function Description

5.1. CPU Register

PSW: Program Status Word

SFR Page = 0~F
SFR Address = 0xD0 RESET = 0000-0000

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

CY: Carry bit.
AC: Auxiliary carry bit.
F0: General purpose flag 0.
RS1: Register bank select bit 1.
RS0: Register bank select bit 0.
OV: Overflow flag.
F1: General purpose flag 1.
P: Parity bit.

The program status word (PSW) contains several status bits that reflect the current state of the CPU. The PSW, shown above, resides in the SFR space. It contains the Carry bit, the Auxiliary Carry(for BCD operation), the two register bank select bits, the Overflow flag, a Parity bit and two user-definable status flags.

The Carry bit, other than serving the function of a Carry bit in arithmetic operations, also serves as the “Accumulator” for a number of Boolean operations.

The bits RS0 and RS1 are used to select one of the four register banks shown in Section “6.2 On-Chip Data RAM”. A number of instructions refer to these RAM locations as R0 through R7.

The Parity bit reflects the number of 1s in the Accumulator. P=1 if the Accumulator contains an odd number of 1s and otherwise P=0.

SP: Stack Pointer

SFR Page = 0~F
SFR Address = 0x81 RESET = 0000-0111

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

The Stack Pointer holds the location of the top of the stack. The stack pointer is incremented before every PUSH operation. The SP register defaults to 0x07 after reset.

DPL: Data Pointer Low

SFR Page = 0~F
SFR Address = 0x82 RESET = 0000-0000

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

The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

DPH: Data Pointer High

SFR Page = 0~F
SFR Address = 0x83 RESET = 0000-0000

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

The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

ACC: Accumulator

SFR Page = 0~F
SFR Address = 0xE0 RESET = 0000-0000

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

This register is the accumulator for arithmetic operations.

B: B Register

SFR Page = 0~F
SFR Address = 0xF0 RESET = 0000-0000

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

This register serves as a second accumulator for certain arithmetic operations.

5.2. CPU Timing

The **MG82F6B08 / 6B001/ 6B104** is a single-chip microcontroller based on a high performance 1-T architecture 80C51 CPU that has an 8051 compatible instruction set, and executes instructions in 1~7 clock cycles (about 6~7 times the rate of a standard 8051 device). It employs a pipelined architecture that greatly increases its instruction throughput over the standard 8051 architecture. The instruction timing is different than that of the standard 8051.

In many 8051 implementations, a distinction is made between machine cycles and clock cycles, with machine cycles varying from 2 to 12 clock cycles in length. However, the 1T-80C51 implementation is based solely on clock cycle timing. All instruction timings are specified in terms of clock cycles. For more detailed information about the 1T-80C51 instructions, please refer section "33 Instruction Set" which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

5.3. CPU Addressing Mode

Direct Addressing (DIR)
In direct addressing the operand is specified by an 8-bit address field in the instruction. Only internal data RAM and SFRs can be direct addressed.

Indirect Addressing (IND)
In indirect addressing the instruction specified a register which contains the address of the operand. Both internal and external RAM can be indirectly addressed.

The address register for 8-bit addresses can be R0 or R1 of the selected bank, or the Stack Pointer.
The address register for 16-bit addresses can only be the 16-bit data pointer register – DPTR.

Register Instruction (REG)
The register banks, containing registers R0 through R7, can be accessed by certain instructions which carry a 3-bit register specification within the op-code of the instruction. Instructions that access the registers this way are code efficient because this mode eliminates the need of an extra address byte. When such instruction is executed, one of the eight registers in the selected bank is accessed.

Register-Specific Instruction
Some instructions are specific to a certain register. For example, some instructions always operate on the accumulator or data pointer, etc. No address byte is needed for such instructions. The op-code itself does it.

Immediate Constant (IMM)
The value of a constant can follow the op-code in the program memory.

Index Addressing
Only program memory can be accessed with indexed addressing and it can only be read. This addressing mode is intended for reading look-up tables in program memory. A 16-bit base register (either DPTR or PC) points to the base of the table, and the accumulator is set up with the table entry number. Another type of indexed addressing is used in the conditional jump instruction.
In conditional jump, the destination address is computed as the sum of the base pointer and the accumulator.

6. Memory Organization

Like all 80C51 devices, the **MG82F6B08 / 6B001/ 6B104** has separate address spaces for program and data memory. The logical separation of program and data memory allows the data memory to be accessed by 8-bit addresses, which can be quickly stored and manipulated by the 8-bit CPU.

Program memory (ROM) can only be read, not written to. There can be up to **8K** bytes of program memory. In the **MG82F6B08 / 6B001/ 6B104**, all the program memory are on-chip Flash memory, and without the capability of accessing external program memory because of no External Access Enable (/EA) and Program Store Enable (/PSEN) signals designed.

Data memory occupies a separate address space from program memory. In the **MG82F6B08 / 6B001/ 6B104**, there are 256 bytes of internal scratch-pad RAM and **768** bytes of on-chip expanded RAM (XRAM).

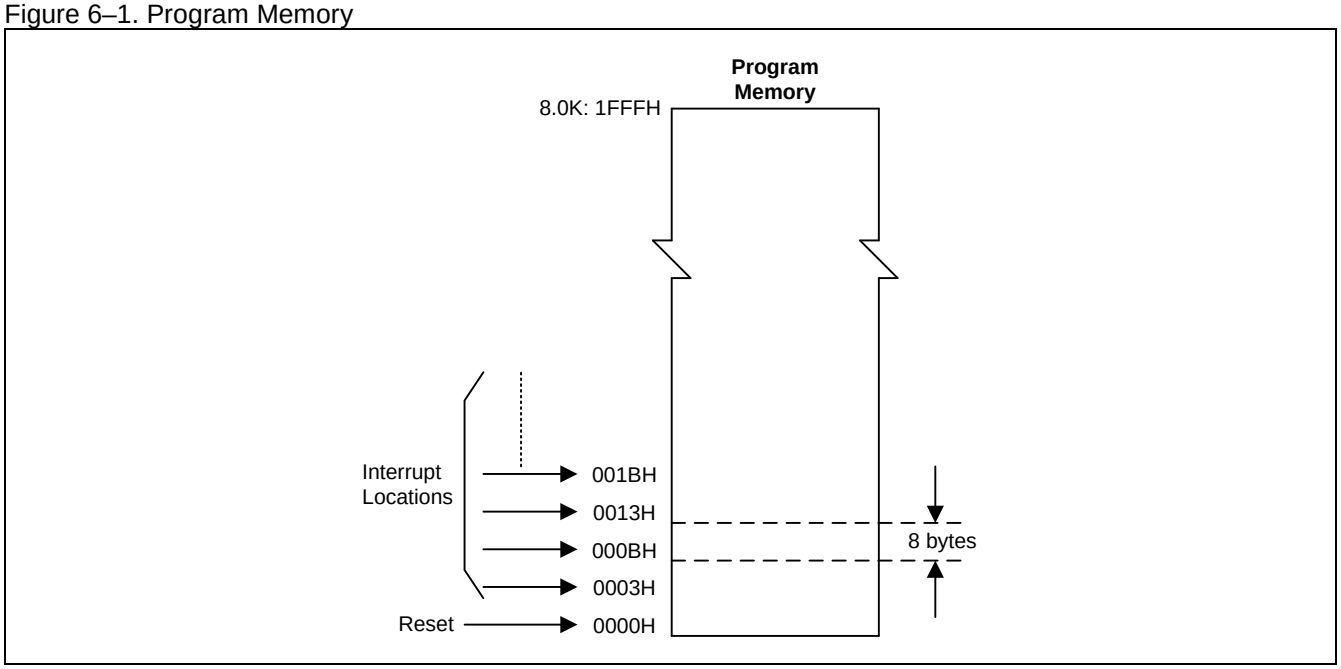
MG82F6B08 / 6B001/ 6B104 provide embedded 512 bytes EEPROM to store the user data in application.

6.1. On-Chip Program Flash

Program memory is the memory which stores the program codes for the CPU to execute, as shown in Figure 6–1. After reset, the CPU begins execution from location 0000H, where should be the starting of the user's application code. To service the interrupts, the interrupt service locations (called interrupt vectors) should be located in the program memory. Each interrupt is assigned a fixed location in the program memory. The interrupt causes the CPU to jump to that location, where it commences execution of the service routine. External Interrupt 0, for example, is assigned to location 0003H. If External Interrupt 0 is going to be used, its service routine must begin at location 0003H. If the interrupt is not going to be used, its service location is available as general purpose program memory.

The interrupt service locations are spaced at an interval of 8 bytes: 0003H for External Interrupt 0, 000BH for Timer 0, 0013H for External Interrupt 1, 001BH for Timer 1, etc. If an interrupt service routine is short enough (as is often the case in control applications), it can reside entirely within that 8-byte interval. Longer service routines can use a jump instruction to skip over subsequent interrupt locations, if other interrupts are in use.

Figure 6–1. Program Memory



6.2. On-Chip Data RAM

Figure 6–2 shows the internal and external data memory spaces available to the **MG82F6B08 / 6B001/ 6B104** user. Internal data memory can be divided into three blocks, which are generally referred to as the lower 128 bytes of RAM, the upper 128 bytes of RAM, and the 128 bytes of SFR space. Internal data memory addresses are always 8-bit wide, which implies an address space of only 256 bytes. Direct addresses higher than 7FH access the SFR space; and indirect addresses higher than 7FH access the upper 128 bytes of RAM. Thus the SFR space and the upper 128 bytes of RAM occupy the same block of addresses, 80H through FFH, although they are physically separate entities.

The lower 128 bytes of RAM are present in all 80C51 devices as mapped in Figure 6–3. The lowest 32 bytes are grouped into 4 banks of 8 registers. Program instructions call out these registers as R0 through R7. Two bits in the Program Status Word (PSW) select which register bank is in use. This allows more efficient use of code space, since register instructions are shorter than instructions that use direct addressing. The next 16 bytes above the register banks form a block of bit-addressable memory space. The 80C51 instruction set includes a wide selection of single-bit instructions, and the 128 bits in this area can be directly addressed by these instructions. The bit addresses in this area are 00H through 7FH.

All of the bytes in the Lower 128 can be accessed by either direct or indirect addressing while the Upper 128 can only be accessed by indirect addressing.

Figure 6–4 gives a brief look at the Special Function Register (SFR) space. SFRs include the Port latches, timers, peripheral controls, etc. These registers can only be accessed by direct addressing. Sixteen addresses in SFR space are both byte- and bit-addressable. The bit-addressable SFRs are those whose address ends in 0H or 8H.

Figure 6–2. Data Memory

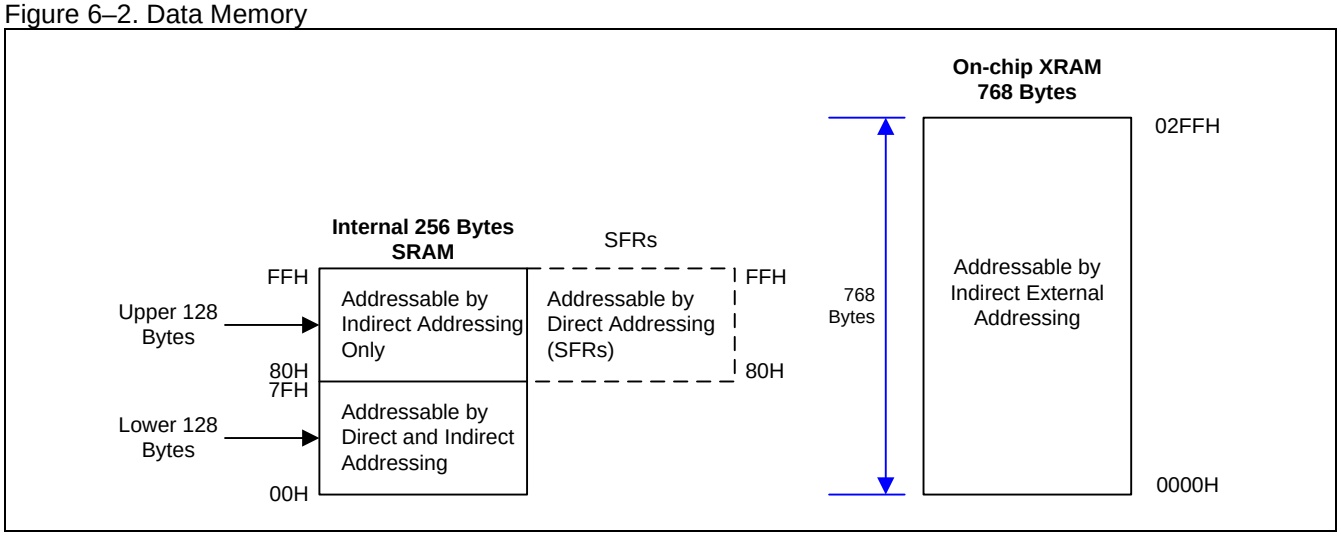


Figure 6–3. Lower 128 Bytes of Internal RAM

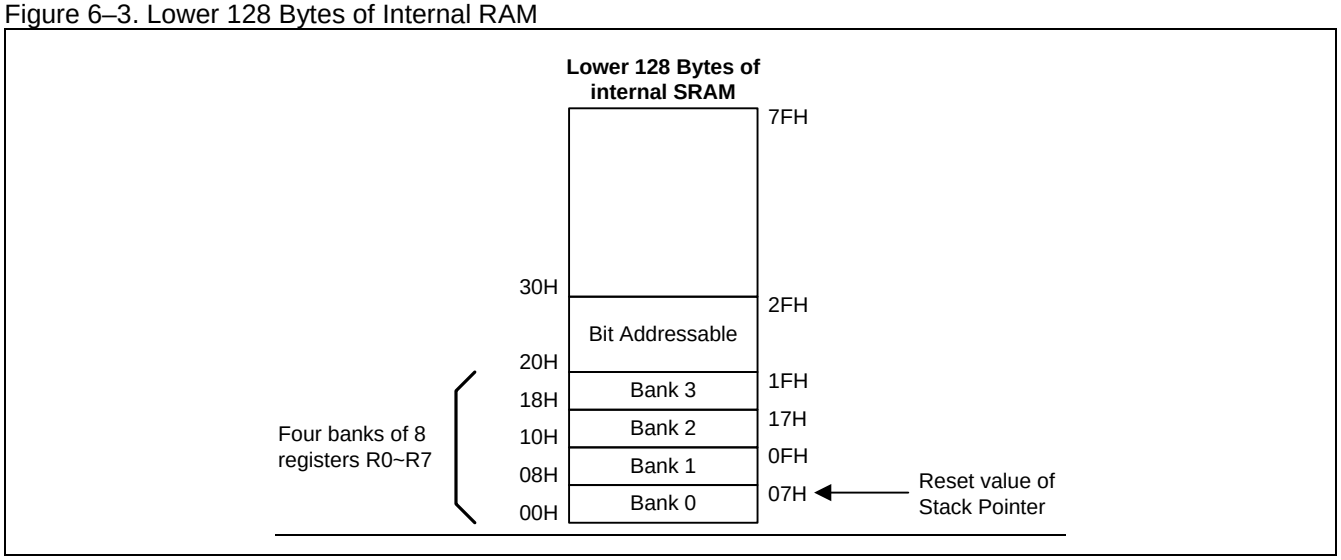
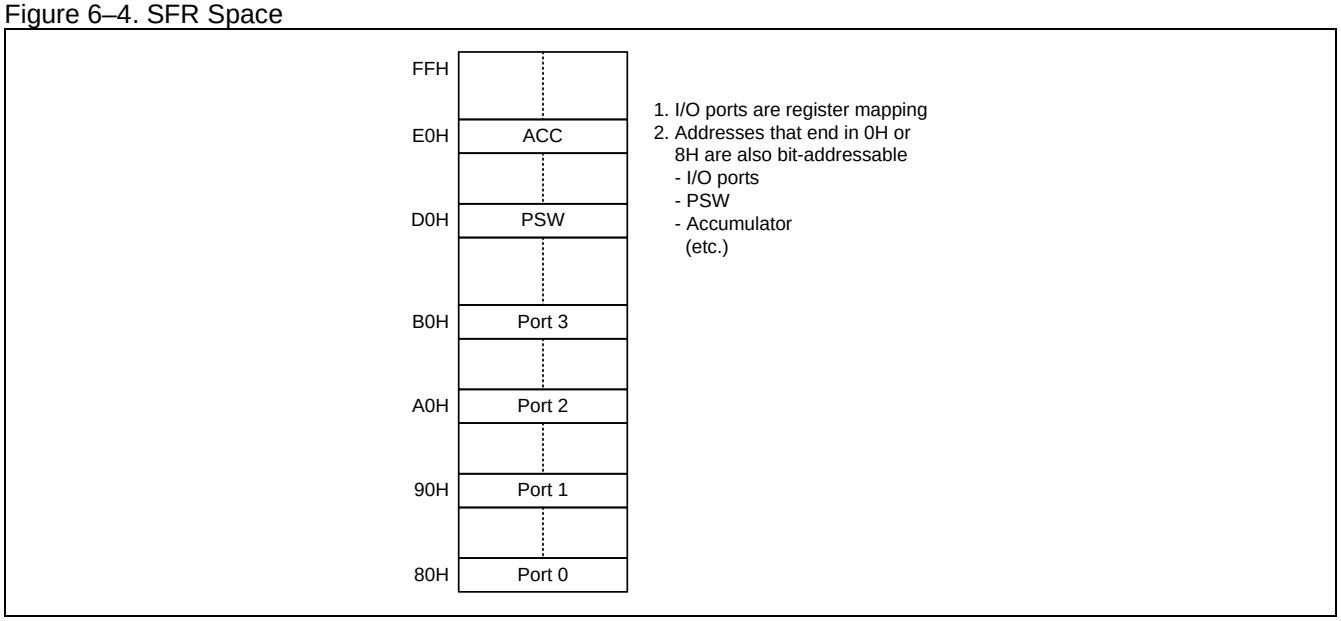


Figure 6–4. SFR Space



6.3. On-chip expanded RAM (XRAM)

To access the on-chip expanded RAM (XRAM), refer to Figure 6–2, the **768** bytes of XRAM (0000H to **02FFH**) are indirectly accessed by move external instruction, “MOVX @Ri” and “MOVX @DPTR”. For C51 compiler, to assign the variables to be located at XRAM, the “pdata” or “xdata” definition should be used. After being compiled, the variables declared by “pdata” and “xdata” will become the memories accessed by “MOVX @Ri” and “MOVX @DPTR”, respectively. Thus the **MG82F6B08 / 6B001/ 6B104** hardware can access them correctly.

6.4. Declaration Identifiers in a C51-Compiler

The declaration identifiers in a C51-compiler for the various **MG82F6B08 / 6B001/ 6B104** memory spaces are as follows:

data
128 bytes of internal data memory space (00h~7Fh); accessed via direct or indirect addressing, using instructions other than MOVX and MOVC. All or part of the Stack may be in this area.

idata
Indirect data; 256 bytes of internal data memory space (00h~FFh) accessed via indirect addressing using instructions other than MOVX and MOVC. All or part of the Stack may be in this area. This area includes the ***data*** area and the 128 bytes immediately above it.

sfr
Special Function Registers; CPU registers and peripheral control/status registers, accessible only via direct addressing.

xdata
External data or on-chip eXpanded RAM (XRAM); duplicates the classic 80C51 64KB memory space addressed via the “MOVX @DPTR” instruction. The **MG82F6B08 / 6B001/ 6B104** has **768** bytes of on-chip xdata memory.

pdata
Paged (256 bytes) external data or on-chip eXpanded RAM; duplicates the classic 80C51 256 bytes memory space addressed via the “MOVX @Ri” instruction. The **MG82F6B08 / 6B001/ 6B104** has 256 bytes of on-chip pdata memory which is shared with on-chip xdata memory.

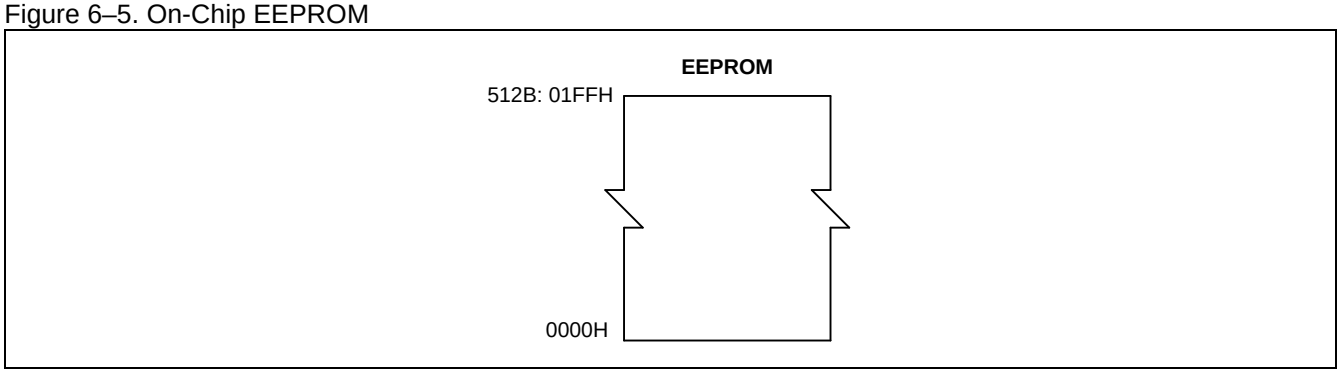
code
8K bytes of program memory space; accessed as part of program execution and via the “MOVC @A+DTPR” instruction. The **MG82F6B08 / 6B001/ 6B104** has **8K** bytes of on-chip code memory.

6.5. On-Chip EEPROM

The **MG82F6B08 / 6B001/ 6B104** contains **512** bytes of data EEPROM memory that has an endurance of at least 100,000 write/erase cycles. And the detailed EEPROM performance, please refer Section "32.7 EEPROM Characteristics".

The EEPROM in **MG82F6B08 / 6B001/ 6B104** is organized as a separate data space, in which single byte can be read and written. The access between the EEPROM and the CPU is through Flash ISP/IAP flow with other IFMT setting. But the difference is CPU will not be halted when EEPROM is programming. A status bit, PBSY (ISCR.1), presents the EEPROM in Process Busy for programming operation. The EEPROM Program Flag is stored on EEPF (ISPCR.0) to indicate the programming flow finished. This flag also support the interrupt capability to share the interrupt vector with System Flag Interrupt. To find the detailed EEPROM access flow, please refer Section "27.2 MG82F6B08 / 6B001/ 6B104 EEPROM Access Flow".

Figure 6–5. On-Chip EEPROM



7. XRAM Access

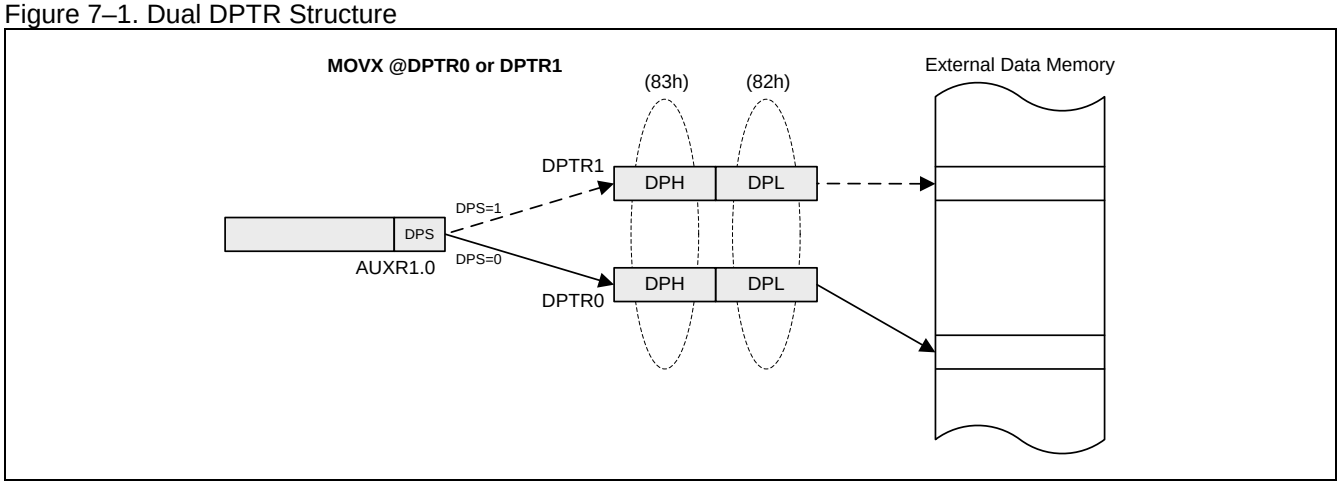
The **MG82F6B08 / 6B001/ 6B104** MCUs include **768** bytes of on-chip RAM mapped into the external data memory space (XRAM). The external memory space may be accessed using the external move instruction (MOVX) and the data pointer (DPTR), or using the MOVX indirect addressing mode using R0 or R1. If the MOVX instruction is used with an 8-bit address operand (such as @R1), then the high byte of the 16-bit address is provided by the XRAM Page Select Register (XRPS).

The **internal** XRAM memory space is accessed using the MOVX instruction. The MOVX instruction has two forms, both of which use an indirect addressing method. The first method uses the Data Pointer, DPTR, a 16-bit register which contains the effective address of the XRAM location to be read or written. The second method uses R0 or R1 in combination with the XRPS register to generate the effective XRAM address.

7.1. MOVX on 16-bit Address with dual DPTR

The dual DPTR structure as shown in Figure 7–1 is a way by which the chip can specify the address of an external data memory location. There are two 16-bit DPTR registers that address the external memory, and a single bit called DPS (AUXR1.0) that allows the program code to switch between them.

Figure 7–1. Dual DPTR Structure



DPTR Instructions

The six instructions that refer to DPTR currently selected using the DPS bit are as follows:

```
INC    DPTR              ; Increments the data pointer by 1
MOV    DPTR,#data16      ; Loads the DPTR with a 16-bit constant
MOV    A,@A+DPTR         ; Move code byte relative to DPTR to ACC
MOVX A,@DPTR             ; Move external RAM (16-bit address) to ACC
MOVX @DPTR,A             ; Move ACC to external RAM (16-bit address)
JMP @A+DPTR              ; Jump indirect relative to DPTR
```

AUXR1: Auxiliary Control Register 1

SFR Page = 0~F
SFR Address = 0xA2 RESET = 0000-0000

7	6	5	4	3	2	1	0
--	--	CRCDS1	CRCDS0	--	--	--	DPS
W	W	R/W	R/W	W	W	W	R/W

Bit 0: DPS, DPTR select bit. Use to switch between DPTR0 and DPTR1.
0: Select DPTR0.
1: Select DPTR1.

DPS	Selected DPTR
0	DPTR0
1	DPTR1

DPL: Data Pointer Low

SFR Page = 0~F
SFR Address = 0x82 RESET = 0000-0000

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

The DPL register is the low byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

DPH: Data Pointer High

SFR Page = 0~F
SFR Address = 0x83 RESET = 0000-0000

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

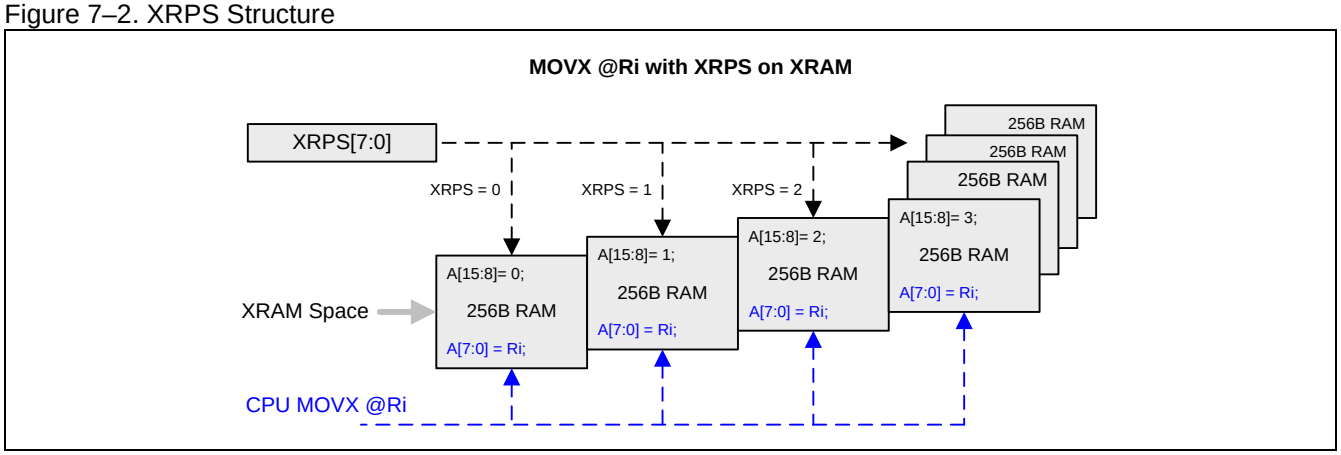
The DPH register is the high byte of the 16-bit DPTR. DPTR is used to access indirectly addressed XRAM and Flash memory.

7.2. MOVX on 8-bit Address with XRPS

The 8-bit form of the MOVX instruction uses the contents of the XRPS SFR to determine the upper 8-bits of the effective address to be accessed and the contents of R0 or R1 to determine the lower 8-bits of the effective address to be accessed.

This function can give the designer to get more efficiency code to access XRAM. To access whole range of XRAM will need to use 2 bytes address. The software complier will compile the 2 bytes address by using DPTR to access specific memory location, it results to add more instructions and will slow down the efficiency. But if use the XRPS with global “pdata” variables, the complier will translate it to MOVX@Ri to reduce many extra instructions to enhance the memory access performance.

Figure 7–2. XRPS Structure



XRPS: XRAM Page Select Register

SFR Page = 0~F

SFR Address = 0x8F RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	0	XRPS.1	XRPS.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~2: Reserved. Software must write “0” on these bits when XRPS is written.

Bit 1~0: XRPS, XRAM Page Select. The XRPS register provides the high byte of the 16-bit external data memory address when using an 8-bit MOVX command, effectively selecting a 256-byte page of RAM. Since the upper (reserved) bits of the register are always zero, the XRPS determines which page of XRAM is accessed. In **MG82F6B08 / 6B001/ 6B104**, XRPS indexes the **3** pages 256-byte RAM (XRPS = 00 ~ 10).

For Example: If XRPS = 0x01, addresses 0x0100 through 0x01FF in XRAM will be accessed.

8. System Clock

There are **three** clock sources for the system clock: Internal High-frequency RC Oscillator (IHRCO), Internal Low-frequency RC Oscillator (ILRCO) and External Clock Input. Figure 8–1 shows the structure of the system clock in **MG82F6B08 / 6B001/ 6B104**.

The **MG82F6B08 / 6B001/ 6B104** always boots from IHRCO on **16MHz**. Software can select the OSCin input on one of the **three** clock sources application required and switches them on the fly. But software needs to settle the clock source stably before clock switching. In external clock input mode (ECKI), the clock source comes from **P4.5** input.

The built-in IHRCO provides two frequencies **16MHz and 22.12MHz** provides high precision frequency for system clock source. To find the detailed IHRCO performance, please refer Section "32.4 IHRCO Characteristics"). In IHRCO or ILRCO mode, **P4.5** can be configured to internal *MCK* output or *MCK*/2 and *MCK*/4 for system application.

The built-in ILRCO provides the low power and low speed frequency about 32KHz to WDT, RTC, and system clock source. MCU can select the ILRCO to system clock source by software for low power operation. To find the detailed IHRCO performance, please refer Section "32.5 ILRCO Characteristics"). In ILRCO mode, **P4.5** can be configured to internal *MCK* output or *MCK*/2 and *MCK*/4 for system application.

The system clock, *SYSCLK*, is obtained from one of these **three** clock sources through the clock divider, as shown in Figure 8–1. The user can program the divider control bits MCKD1~MCKD0 (in CKCON3 register) and SCKS2 ~ SCKS0 (in CKCON0 register) to get the desired system clock.

After MCU reset, the SYSCLK is 8MHz by IHRCO= 16MHz/2 with default value, " 01", on MCKD1~MCKD0.

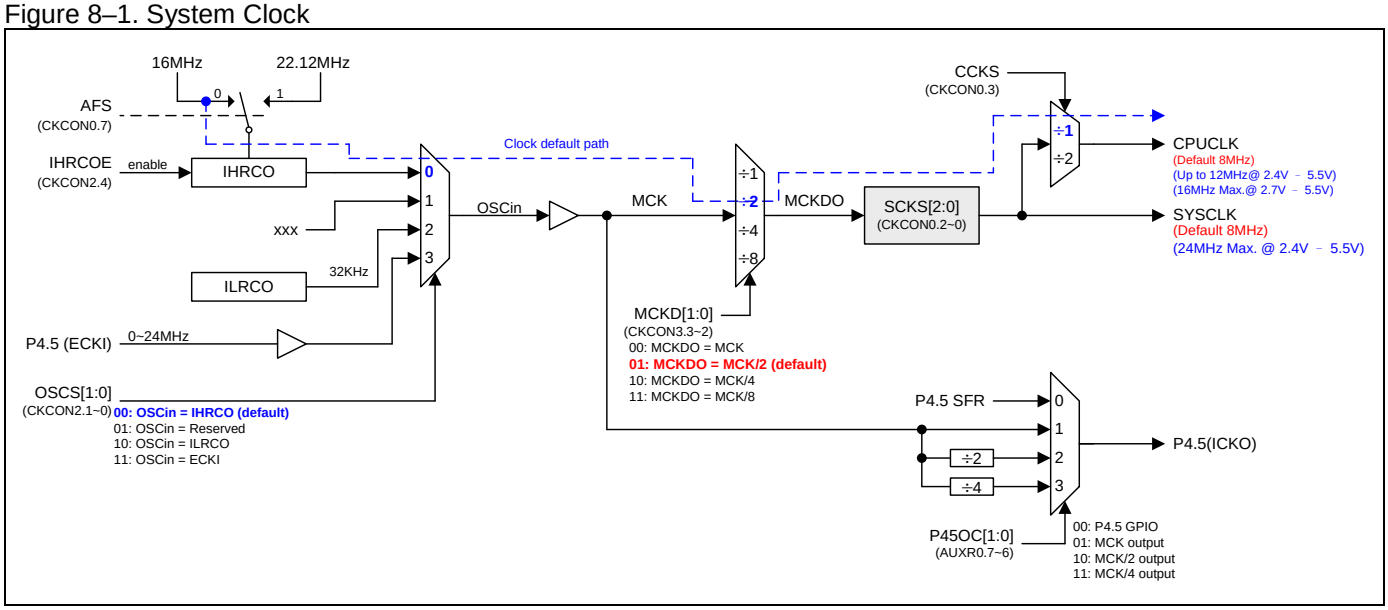
8.1. Clock Structure

Figure 8–1 presents the principal clock systems in the **MG82F6B08 / 6B001/ 6B104**. The initial oscillator source of CPUCLK is set to 8MHz, IHRCO/2 by MCKD divider. It can use the combinations of the clock divider for different frequencies. The maximum CPUCLK is as following:

- SYSCLK up to 24MHz @ 2.4V – 5.5V
- CPUCLK up to 12MHz @ 2.4V – 5.5V; Up to 16MHz @ 2.7V – 5.5V

If the applications need higher performance, then HSE (DCON0 Bit 7) needs to be set when CPUCLK > 48KHz.

Figure 8–1. System Clock



8.2. Clock Source Switching

There are **three** clock sources for the system clock: Internal High-frequency RC Oscillator (IHRCO), Internal Low-frequency RC Oscillator (ILRCO) and External Clock Input. Figure 8–1 shows the structure of the system clock in **MG82F6B08 / 6B001/ 6B104**. The **MG82F6B08 / 6B001/ 6B104** always boots from IHRCO/2 on **8MHz**. OSCS[1:0] are used to select the clock source by software setting, but the software need to wait until the clock be settle before switch the clock source.

8.3. Clock Register

CKCON0: Clock Control Register 0

SFR Page = **0~F & P**

SFR Address = 0xC7 RESET = 0001-0000

7	6	5	4	3	2	1	0
AFS	0	0	1	CCKS	SCKS2	SCKS1	SCKS0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: AFS, Alternated Frequency Selection.
0: Select IHRCO on 16MHz.
1: Select IHRCO on 22.12MHz.

Bit 3: CCKS, CPU Clock Select.
0: Select CPU Clock as SYSCLK.
1: Select CPU Clock as SYSCLK/2.

Bit 2~0: SCKS2 ~ SCKS0, programmable System Clock Selection.

SCKS[2:0]	System Clock (SYSCLK)
0 0 0	MCKDO/1
0 0 1	MCKDO/2
0 1 0	MCKDO/4
0 1 1	MCKDO/8
1 0 0	MCKDO/16
1 0 1	MCKDO/32
1 1 0	MCKDO/64
1 1 1	MCKDO/128

CKCON1: Clock Control Register 1

SFR Page = **0~F**

SFR Address = 0xBF RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	OSCSTA3	OSCSTA2	OSCSTA1	OSCSTA0
R	**R**	**R**	**R**	**R**	**R**	**R**	**R**

Bit 3~0: OSCSTA[3:0]
0001: OSCin MUX is using IHRCO clock source
0010: Reserved
0100: OSCin MUX is using ILRCO clock source
1000: OSCin MUX is using ECKI clock source
Others: OSCin MUX is going on switching clock

CKCON2: Clock Control Register 2

SFR Page = **P Only**

SFR Address = 0x40 RESET = 0101-0000

7	6	5	4	3	2	1	0
0	1	0	IHRCOE	0	0	OSCS1	OSCS0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: IHRCOE, Internal High frequency RC Oscillator Enable.
0: Disable internal high frequency RC oscillator.
1: Enable internal high frequency RC oscillator. If this bit is set by CPU software, it needs **32 us** to have stable output after IHRCOE is enabled.

Bit 1~0: OSCS[1:0], OSCin Source selection.

OSCS[1:0]	OSCin source Selection
0 0	IHRCO
0 1	Reserved
1 0	ILRCO
1 1	ECKI, External Clock Input (P4.5) as OSCin.

CKCON3: Clock Control Register 3

SFR Page = **P only**

SFR Address = 0x41 RESET = 0000-0110

7	6	5	4	3	2	1	0
WDTCS1	WDTCS0	FWKP	WDTFS	MCKD1	MCKD0	1	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: WDTCS1~0. WDT clock source selection.

Bit 5: FWKP, MCU Fast wake up control.
0: Select MCU for normal wakeup time about **90~120us** from power-down mode.
1: Select MCU for fast wakeup time about **15~30us** from power-down mode.

Bit 4: WDTFS. WDT overflow source selection.
0: Select WDT bit-7 overflow as WDT event source.
1: Select WDT bit-0 overflow as WDT event source.

Bit 3~2: MCKD[1:0], MCK Divider Output selection. Default is MCK/2.

MCKD[1:0]	MCKDO Frequency	if MCK = 16MHz	if MCK = 22MHz
0 0	MCKDO = MCK	MCKDO = 16MHz	MCKDO = 22MHz
0 1	MCKDO = MCK/2	MCKDO = 8MHz	MCKDO = 11MHz
1 0	MCKDO = MCK/4	MCKDO = 4MHz	MCKDO = 5.5MHz
1 1	MCKDO = MCK/8	MCKDO = 2MHz	MCKDO = 2.76MHz

AUXR0: Auxiliary Register 0

SFR Page = 0~F
SFR Address = 0xA1 RESET = 0000-0000

7	6	5	4	3	2	1	0
P45OC1	P45OC0	0	PBKF	0	0	INT1H	INT0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: P4.5 function configured control bit 1 and 0. The two bits only act when internal RC oscillator (IHRCO or ILRCO) is selected for system clock source. In external clock input mode, P4.5 is the dedicated clock input pin. In internal oscillator condition, P4.5 provides the following selections for GPIO or clock source generator. When P45OC[1:0] index to non-P4.5 GPIO function, P4.5 will drive the on-chip RC oscillator output to provide the clock source for other devices.

P45OC[1:0]	P4.5 function	I/O mode
0 0	P4.5	By P4M1.5 & P4M0.5
0 1	MCK	By P4M1.5 & P4M0.5
1 0	MCK/2	By P4M1.5 & P4M0.5
1 1	MCK/4	By P4M1.5 & P4M0.5

For clock-out on P4.5 function, it is recommended to set {P4M1.5, P4M0.5} to “01” which selects P4.5 as push-push output mode.

DCON0: Device Control Register 0

SFR Page = **P Only**
SFR Address = 0x4C POR = 1000-0011

7	6	5	4	3	2	1	0
HSE	IAPO	0	0	0	IORCTL	RSTIO	OCDE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: HSE, High Speed operation Enable.
0: Select CPU running in lower speed mode (F_{CPUCLK} ≤ 48KHz) which is slow down internal circuit to reduce power consumption.

Software flow to disable HSE
Step 1: Switch F_{CPUCLK} ≤ 48KHz.
Step 2: Set HSE = 0.

1: Enable CPU full speed operation if F_{CPUCLK} > 48KHz. Before select high frequency clock (>48KHz) on CPUCLK, software must set HSE to switch internal circuit for high speed operation.

Software flow to enable HSE
Step 1: Set HSE = 1.
Step 2: Switch F_{CPUCLK} > 48KHz.

9. Watch Dog Timer (WDT)

9.1. WDT Structure

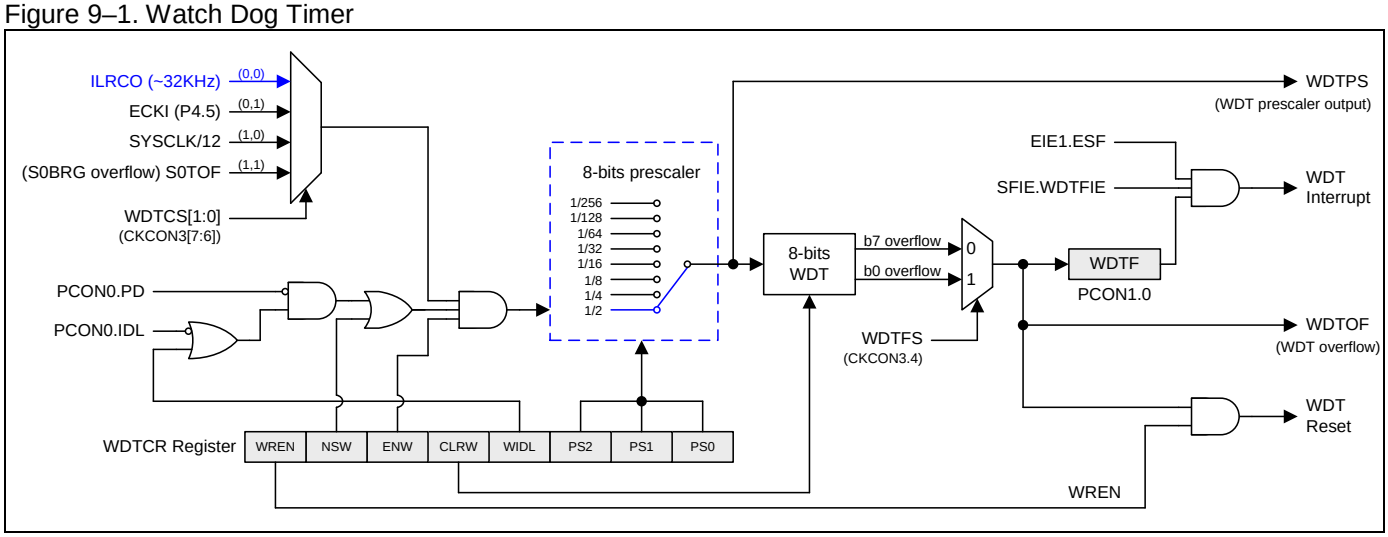
The Watch-dog Timer (WDT) is intended as a recovery method in situations where the CPU may be subjected to software upset. The WDT consists of an 8-bit free-running counter, a 8-bit prescaler and a control register (WDTCR). Figure 9–1 shows the WDT structure in **MG82F6B08 / 6B001/ 6B104**.

There are **four** selections for WDT clock source. The clock source must be configured before WDT enabled. The default WDT clock source is 32KHz ILRCO. The WDT overflow will set the WDTF (PCON1.0) which can be configured to generate an interrupt by enabled WDTFIE (SFIE.0) and enabled ESF (EIE1.3). The overflow can also trigger a system reset when WREN (WDTCR.7) is set. To prevent WDT overflow, software needs to clear it by writing “1” to the CLRW bit (WDTCR.4) before WDT overflows.

Once the WDT is enabled by setting ENW bit, there is no way to disable it except through power-on reset or page-p SFR over-write on ENW, which will clear the ENW bit. The WDTCR register will keep the previous programmed value unchanged after hardware (RST-pin) reset, software reset and WDT reset.

WREN, NSW and ENW are implemented to one-time-enabled function, only writing “1” valid in general SFR page. Page-P SFR Access on WDTCR can disable WREN, NSW and ENW, writing “0” on WDTCR.7~5. Please refer Section “9.4 WDT Register” and Section “28 Page P SFR Access” for more detail information.

Figure 9–1. Watch Dog Timer



9.2. WDT During Idle

In the Idle mode, the WIDL bit (WDTCR.3) determines whether WDT counts or not. Set this bit to let WDT keep counting in the Idle mode. If the hardware option NSWDT is enabled, the WDT always keeps counting regardless of WIDL bit.

9.3. WDT During Power Down (Auto Wake Up)

In the Power down mode, the ILRCO won’t stop if the NSW (WDTCR.6) is enabled. The MCU enters Watch mode to behave an auto-wakeup function. That lets WDT keep counting even in Power down mode (Watch Mode). After WDT overflows, it will wake up the CPU from interrupt or reset by software configured. This function is only active when WDT clock source is come from **ILRCO** or **ECKI** input which can be derived from external input.

9.4. WDT Register

WDTCR: Watch-Dog-Timer Control Register

SFR Page = 0~F & P

SFR Address = 0xE1 POR = XXX0-XXXX (0000-0111)

7	6	5	4	3	2	1	0
WREN	NSW	ENW	CLRW	WIDL	PS2	PS1	PS0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: WREN, WDT Reset Enable. The initial value can be changed by hardware option, WRENO.
0: The overflow of WDT does not set the WDT reset. The WDT overflow flag, WDTF, may be polled by software or trigger an interrupt.
1: The overflow of WDT will cause a system reset. Once WREN has been set, it can not be cleared by software in page 0~F. **In page P, software can modify it to "0" or "1".**

Bit 6: NSW. Non-Stopped WDT. The initial value can be changed by hardware option, NSWDT.
0: WDT stop counting while the MCU is in power-down mode.
1: WDT always keeps counting while the MCU is in power-down mode (Watch Mode) or idle mode. Once NSW has been set, it can not be cleared by software in page 0~F. **In page P, software can modify it to "0" or "1".**

Bit 5: ENW. Enable WDT.
0: Disable WDT running. This bit is only cleared by POR.
1: Enable WDT while it is set. Once ENW has been set, it can not be cleared by software in page 0~F. **In Page P, software can modify it as "0" or "1".**

Bit 4: CLRW. WDT clear bit.
0: Writing "0" to this bit is no operation in WDT.
1: Writing "1" to this bit will clear the 8-bit WDT counter to 00H. Note this bit has no need to be cleared by writing "0".Clear WDT to recount while it is set.

Bit 3: WIDL. WDT idle control.
0: WDT stops counting while the MCU is in idle mode.
1: WDT keeps counting while the MCU is in idle mode.

Bit 2~0: PS2 ~ PS0, select prescaler output for WDT time base input.

When WDTFS (CKCON3.4) = 0, WDT clock source= ILRCO or SYSCLK/12

PS[2:0]	Prescaler Value	WDT Period (WDT clock = ILRCO)	WDT Period (WDT clock = SYSCLK/12) (SYSCLK = IHRCO, 16MHz)
0 0 0	2	16 ms	0.384 ms
0 0 1	4	32 ms	0.768 ms
0 1 0	8	64 ms	1.536 ms
0 1 1	16	128 ms	3.072 ms
1 0 0	32	256 ms	6.144 ms
1 0 1	64	512 ms	12.288 ms
1 1 0	128	1024 ms	24.576 ms
1 1 1	256	2048 ms	49.512 ms

When WDTFS (CKCON3.4) = 1, WDT clock source= ILRCO

PS[2:0]	Prescaler Value	WDT Period [Note] (clock source = ILRCO)
0 0 0	2	245 us= 125 us +120us
0 0 1	4	370 us= 250 us +120us
0 1 0	8	620 us= 500 us +120 us
0 1 1	16	1.12 ms= 1 ms+120 us
1 0 0	32	2.12 ms= 2 ms +120us
1 0 1	64	4.12 ms= 4 ms +120us
1 1 0	128	8.12 ms= 8 ms +120us
1 1 1	256	16.12 ms= 16 ms

Note: When WDT clock source is ILRCO, the WDT internal logic latency is around 120us. Under this condition we suggest to add 120us of WDT period it shorter than 12ms.

CKCON3: Clock Control Register 3

SFR Page = **P only**

SFR Address = 0x41 RESET = 0000-0110

7	6	5	4	3	2	1	0
WDTCS1	WDTCS0	FWKP	WDTFS	MCKD1	MCKD0	1	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: WDTCS1~0, WDT Clock Source selection [1:0].

WDTCS1~0	WDT Clock Source
00	ILRCO
01	ECKI (P4.5)
10	SYSCLK/12
11	S0TOF

Bit 4: WDTFS. WDT overflow source selection.
0: Select WDT bit-7 overflow as WDT event source.
1: Select WDT bit-0 overflow as WDT event source.

PCON1: Power Control Register 1

SFR Page = 0~F & P

SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 1: WDTF, WDT overflow flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing ":0" is no operation.
1: This bit is only set by hardware when WDT overflows. Writing **"1"** on this bit will clear WDTF.

SFIE: System Flag Interrupt Enable Register

SFR Page = 0~F

SFR Address = 0x8E POR = 0110-0000

7	6	5	4	3	2	1	0
SIDFIE	1	1	RTCFIE	SPWIE	BOF1IE	BOF0IE	WDTFIE
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 0: WDTFIE, Enable WDTF (PCON1.0) Interrupt.
0: Disable WDTF interrupt.
1: Enable WDTF interrupt.

9.5. WDT Hardware Option

In addition to being initialized by software, the WDTCR register can also be automatically initialized at power-up by the hardware options WRENO, NSWDT, HWENW, HWWIDL and HWPS[2:0], which should be programmed by a universal Writer or Programmer, as described below.

If HWENW is programmed to “enabled”, then hardware will automatically do the following initialization for the WDTCR register at power-up: (1) set ENW bit, (2) load WRENO into WREN bit, (3) load NSWDT into NSW bit, (4) load HWWIDL into WIDL bit, and (5) load HWPS[2:0] into PS[2:0] bits.

If both of HWENW and WDSFWP are programmed to “enabled”, hardware still initializes the WDTCR register content by WDT hardware option at power-up. Then, any CPU writing on WDTCR bits will be inhibited except writing “1” on WDTCR.4 (CLRW), clear WDT, even though access through Page-P SFR mechanism.

WRENO:
- ☑: Enabled. Set WDTCR.WREN to enable a system reset function by WDTF.
- ☐: Disabled. Clear WDTCR.WREN to disable the system reset function by WDTF.

NSWDT: Non-Stopped WDT
- ☑: Enabled. Set WDTCR.NSW to enable the WDT running in power down mode (watch mode).
- ☐: Disabled. Clear WDTCR.NSW to disable the WDT running in power down mode (disable Watch mode).

HWENW: Hardware loaded for “ENW” of WDTCR.
- ☑: Enabled. Enable WDT and load the content of WRENO, NSWDT, HWWIDL and HWPS2~0 to WDTCR after power-on.
- ☐: Disabled. WDT is not enabled automatically after power-on.

HWWIDL, HWPS2, HWPS1, HWPS0:
When HWENW is enabled, the content on these four fused bits will be loaded to WDTCR SFR after power-on.

WDSFWP:
- ☑: Enabled. The WDT SFRs, WREN, NSW, WIDL, PS2, PS1 and PS0 in WDTCR, will be write-protected.
- ☐: Disabled. The WDT SFRs, WREN, NSW, WIDL, PS2, PS1 and PS0 in WDTCR, are free for writing of software.

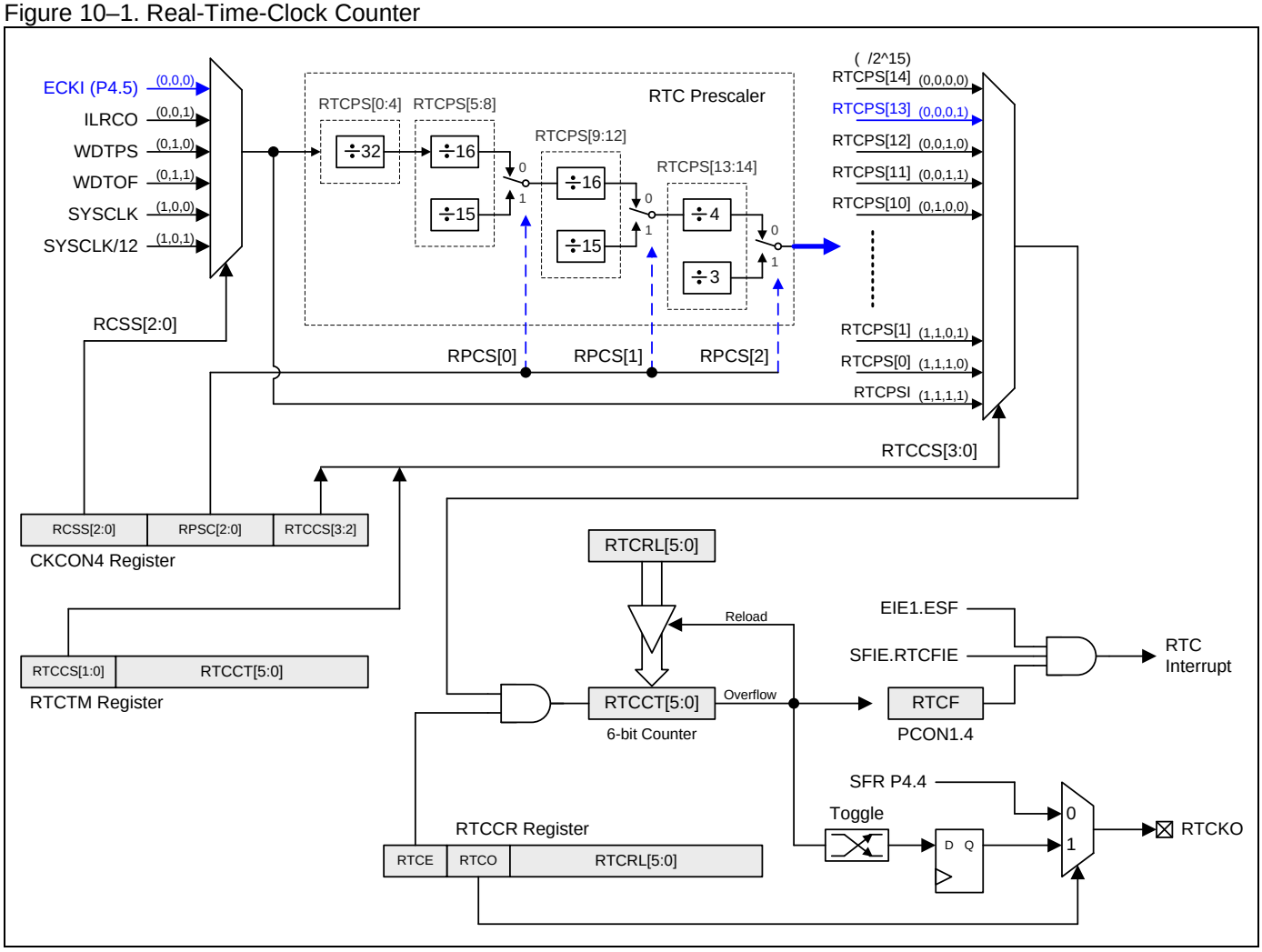
10. Real-Time-Clock (RTC)/System-Timer

The **MG82F6B08 / 6B001/ 6B104** has a simple Real-Time clock that allows a user to continue running an accurate timer while the rest of the device is powered-down. The Real-Time clock can be a wake-up or an interrupt source. The Real-Time clock is a maximum 21-bit up counter comprised of a 0~15-bit prescaler and a 6-bit loadable up counter. When it overflows, the 6-bit counter will be reloaded again and the RTCF flag will be set. The clock source for this prescaler has **6** selections, and needs to set RCSS[2:0] to select one of source before enable WDT. Figure 10–1 shows the RTC structure in **MG82F6B08 / 6B001/ 6B104**.

To input 32.768 KHz ECKI for the RTC module input will provide a programmable overflow period for 0.5S to 64S. The counter also provides a timer function with the clock derived from SYSCLK for a system timer function. The maximum overflow period for the system timer function is SYSCLK/2^21. The ILRCO provides the internal clock source for RTC module. The WDTPS and WDTOF come from WDT prescaler and WDT overflow to provide the extended prescaler source for more long wake-up time requirement. The RTC clock source must be configured before RTCE enabled.

RTCO enables the RTC overflow output on port pin. Only power-on reset will reset the Real-Time clock and its associated SFRs to the default state.

Figure 10–1. Real-Time-Clock Counter



10.1. RTC Register

RTCCR: Real-Time-Clock Control Register

SFR Page = **0~F & P**

SFR Address = **0xBE/0x54** POR = 0011-1111

7	6	5	4	3	2	1	0
RTCE	RTCO	RTCRL.5	RTCRL.4	RTCRL.3	RTCRL.2	RTCRL.1	RTCRL.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: RTCE, RTC Enable.
0: Stop RTC Counter, RTCCT.
1: Enable RTC Counter and set RTCF when RTCCT overflows. When RTCE is set, CPU can not access RTCTM. RTCTM must be accessed in RTCE cleared.

Bit 6: RTCO, RTC Output enabled. The frequency of RTCKO is (RTC overflow rate)/2.
0: Disable the RTCKO output.
1: Enable the RTCKO output on **P4.4**.

Bit 5~0: RTCRL[5:0], RTC counter reload value register. This register is accessed by CPU and the content in the register is reloaded to RTCCT when RTCCT overflows.

RTCTM: Real-Time-Clock Timer Register

SFR Page = **0~F & P**

SFR Address = **0xB6/0x55** POR = 0111-1111

7	6	5	4	3	2	1	0
RTCCS.1	RTCCS.0	RTCCT.5	RTCCT.4	RTCCT.3	RTCCT.2	RTCCT.1	RTCCT.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: RTCCS.1~0, RTC Clock Selection. Default is “01”.

RTCCS.3~0	Clock Source	RTC Interrupt Duration	Min. Step
0 0 0 0	RTCPS[14] (/2^15)	1S ~ 64S when **P4.5** = 32768Hz	1S
0 0 0 1	RTCPS[13] (/2^14)	0.5S ~ 32S when **P4.5** = 32768Hz	0.5S (default)
0 0 1 0	RTCPS[13] (/2^13)	0.25S ~ 16S when **P4.5** = 32768Hz	0.25S
……	……	……	……
1 0 1 0	RTCPS[4] (/2^5)	976us ~ 62.46ms when **P4.5** = 32768Hz	976 us
1 0 1 1	RTCPS[3] (/2^4)		488 us
1 1 0 0	RTCPS[2] (/2^3)		244 us
1 1 0 1	RTCPS[1] (/2^2)	122us ~ 3.9ms when **P4.5** = 32768Hz	122 us
1 1 1 0	RTCPS[0] (/2^1)	61us ~ 1.952ms when **P4.5** = 32768Hz	61 us
1 1 1 1	RTCPSI (/2^0)	30.5us ~ 976us when **P4.5** = 32768Hz	30.5 us

Bit 5~0: RTCCT[5:0], RTC counter register. It is a counter for RTC function or System Timer function by different clock source selection on RTCCS[1:0]. When the counter overflows, it sets the RTCF flag which shares the system flag interrupt when RTCFIE is enabled. The maximum RTC overflow period is 64 seconds.

CKCON4: Clock Control Register 4

SFR Page = **P only**

SFR Address = 0x42 RESET = 0000-0000

7	6	5	4	3	2	1	0
RCSS2	RCSS1	RCSS0	RPSC2	RPSC1	RPSC0	RTCCS3	RTCCS2
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~5: RTC Clock Source selection [2:0]

RCSS2, RCSS1, RCSS0	RTC Clock Selection
0 0 0	ECKI (**P4.5**)
0 0 1	ILRCO
0 1 0	WDTPS
0 1 1	WDTOF
1 0 0	SYSCLK
1 0 1	SYSCLK / 12
1 1 0	Reserved
1 1 1	Reserved

PCON1: Power Control Register 1

SFR Page = 0~F & P

SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: RTCF, RTC overflow flag.

0: This bit must be cleared by software writing **"1"** on it. Software writing "0" is no operation.

1: This bit is only set by hardware when RTCCT overflows. Writing **"1"** on this bit will clear RTCF.

SFIE: System Flag Interrupt Enable Register

SFR Page = 0~F

SFR Address = 0x8E POR = 0110-0000

7	6	5	4	3	2	1	0
SIDFIE	1	1	RTCFIE	SPWIE	BOF1IE	BOF0IE	WDTFIE
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: RTCFIE, Enable RTCF (PCON1.4) Interrupt.

0: Disable RTCF interrupt.

1: Enable RTCF interrupt. If enabled, RTCF will wake up CPU in Idle mode or power-down mode.

11. System Reset

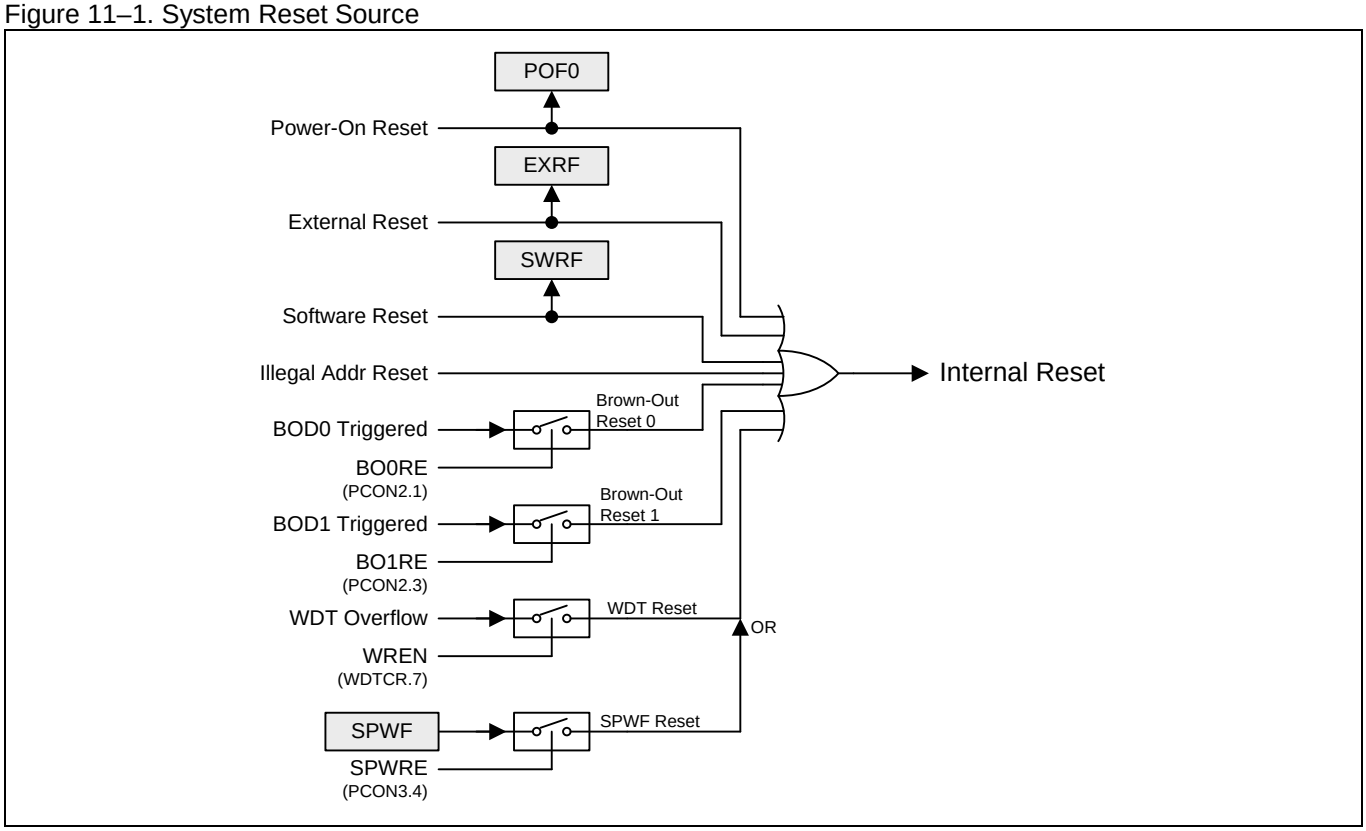
During reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector, 0000H, or ISP start address by OR setting. The **MG82F6B08 / 6B001/ 6B104** has **8** sources of reset: power-on reset, external reset, software reset, illegal address reset, brown-out reset 0, brown-out reset 1, WDT reset and SPWF reset. Figure 11–1 shows the system reset source in **MG82F6B08 / 6B001/ 6B104**.

The following sections describe the reset happened source and corresponding control registers and indicating flags.

11.1. Reset Source

Figure 11–1 presents the reset systems in the **MG82F6B08 / 6B001/ 6B104** and all of its reset sources.

Figure 11–1. System Reset Source



11.2. Power-On Reset

Power-on reset (POR) is used to internally reset the CPU during power-up. The CPU will keep in reset state and will not start to work until the VDD power rises above the voltage of Power-On Reset. And, the reset state is activated again whenever the VDD power falls below the POR voltage. During a power cycle, VDD must fall below the POR voltage before power is reapplied in order to ensure a power-on reset

PCON0: Power Control Register 0

SFR Page = 0~F & P POR = 0001-0000
SFR Address = 0x87 RESET = 000X-0000

7	6	5	4	3	2	1	0
SMOD1	SMOD0	GF	POF0	GF1	GF0	PD	IDL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 4: POF0, Power-On Flag 0.
0: The flag must be cleared by software to recognize next reset type.
1: Set by hardware when VDD rises from 0 to its nominal voltage. POF0 can also be set by software.

The Power-on Flag, POF0, is set to “1” by hardware during power up or when VDD power drops below the POR voltage. It can be clear by firmware and is not affected by any warm reset such as external reset, Brown-Out reset, software reset (ISPCR.5) and WDT reset. It helps users to check if the running of the CPU begins from power up or not. Note that the POF0 must be cleared by firmware.

11.3. External Reset

A reset is accomplished by holding the RESET pin HIGH for at least **24** oscillator periods while the oscillator is running. To ensure a reliable power-up reset, the hardware reset from RST pin is necessary.

PCON1: Power Control Register 1

SFR Page = 0~F & P
SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 6: EXRF, External Reset Flag.
0: This bit must be cleared by software writing **“1”** on it. Software writing “:0” is no operation.
1: This bit is only set by hardware if an External Reset occurs. Writing **“1”** on this bit will clear EXRF.

11.4. Software Reset

Software can trigger the CPU to restart by software reset, writing “1” on SWRST (ISPCR.5), and set the SWRF flag (PCON1.7). SWBS decides the CPU is boot from ISP or AP region after the reset action

ISPCR: ISP Control Register

SFR Page = 0~F
SFR Address = 0xE7 POR = 0000-0000

7	6	5	4	3	2	1	0
ISPEN	SWBS	SWRST	CFAIL	0	0	PBSY	EEPF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 6: SWBS, software boot selection control.
0: Boot from AP-memory after reset.
1: Boot from ISP memory after reset.

Bit 5: SWRST, software reset trigger control.
0: Write “0” is no operation
1: Write “1” to generate software system reset. It will be cleared by hardware automatically.

PCON1: Power Control Register 1

SFR Page = 0~F & P

SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SWRF, Software Reset Flag.
0: This bit must be cleared by software writing “1” on it. Software writing “0” is no operation.
1: This bit is only set by hardware if a Software Reset occurs. Writing “1” on this bit will clear SWRF.

11.5. Brown-Out Reset

In **MG82F6B08 / 6B001/ 6B104**, there are two Brown-Out Detectors (BOD0 & BOD1) to monitor VDD power. BOD0 services the fixed detection level at VDD = 2.35V. BOD1 detects the VDD level by software selecting **4.2V, 3.6V, 2.4V & 2.7V**. If VDD power drops below BOD0 or BOD1 monitor level. Associated flag, BOF0 and BOF1, is set. If BO0RE (PCON2.1) is enabled, BOF0 indicates a BOD0 Reset occurred. If BO1RE (PCON2.3) is enabled, BOF1 indicates a BOD1 Reset occurred.

Notice of BOD1 Reset

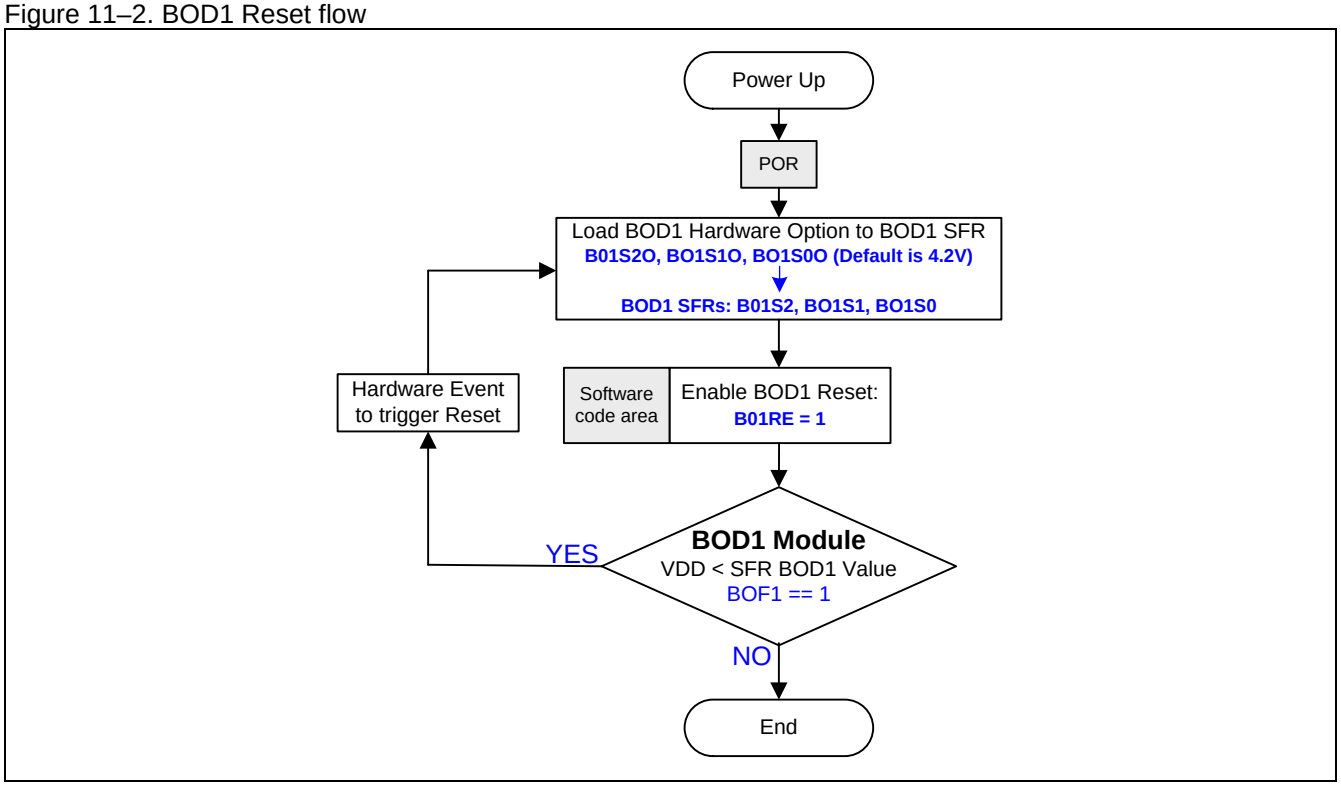
When BOD1 reset, it will reload the BOD1 Hardware Option value (B01S2O, BO1S1O, BO1S0O, please reference Chapter 30 Hardware Option for detail) into BOD1 SFR (B01S2, BO1S1, BO1S0).

There are two ways to control BOD1 Reset:

1. Hardware Option:
   a. Using H/W option to set the BOD1 voltage detect level.
   b. To set H/W option BO1REO to enable BOD1 Reset. User also can reserve the BOD1 Reset and to enable it in the software code when needed.

Figure 11–2 shows the BOD1 reset flow.

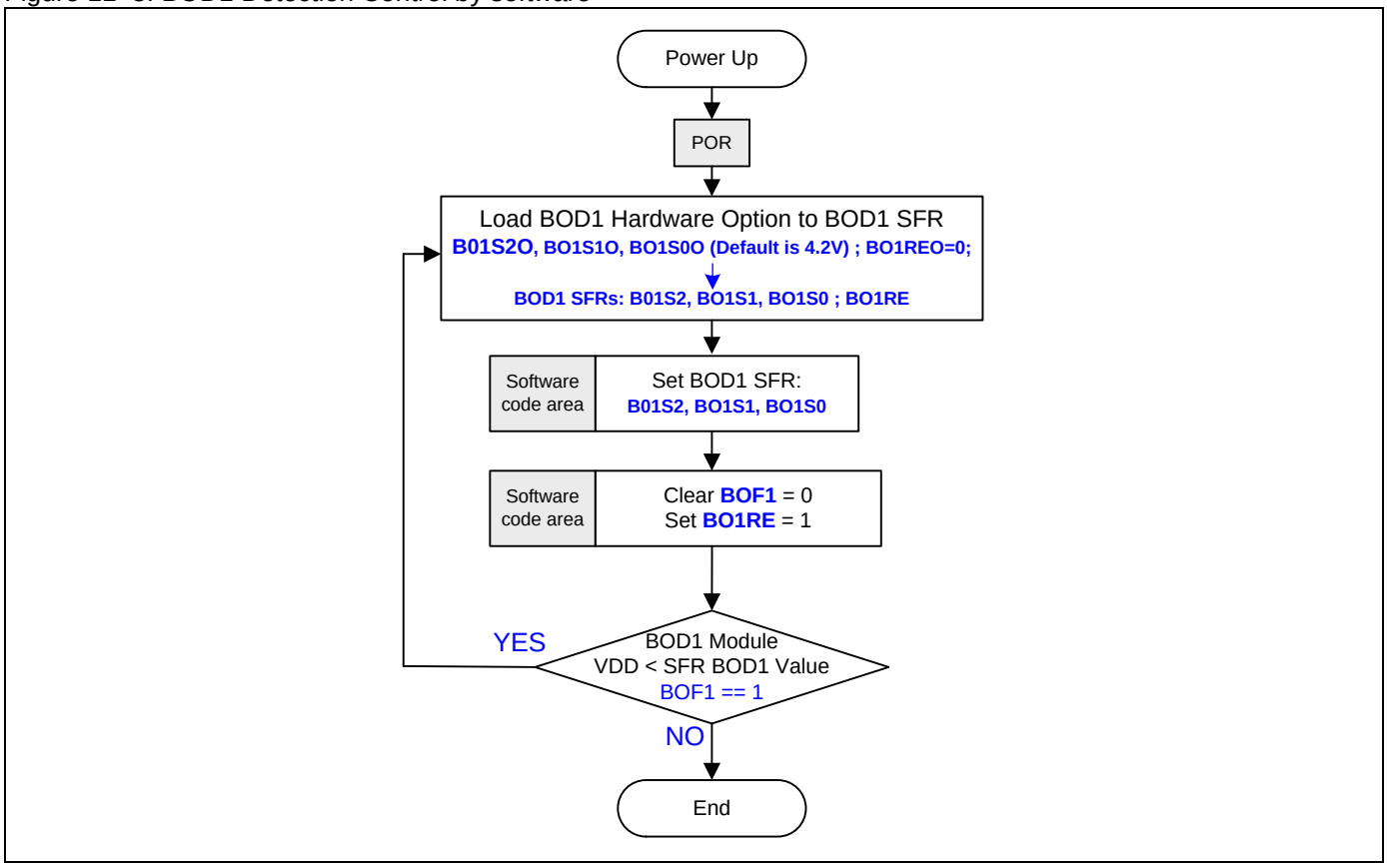
Figure 11–2. BOD1 Reset flow



2. Software control BOD1:
   a. Do not enable BOD1 Reset by Hardware Option BO1REO. (Default is disabled)
   b. Set BOD1 detect voltage by BO1S[2:0]
   c. Clear BOD1 flag, BOF1 = 0
   d. Enable BOD1 Reset

Figure 11–3 is the software control flow for BOD1.

Figure 11–3. BOD1 Detection Control by software



PCON1: Power Control Register 1

SFR Page = 0~F & P

SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 2: BOF1, BOF1 (Reset) Flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing ":0" is no operation.
1: This bit is only set by hardware when VDD meets BOD1 monitored level. Writing **"1"** on this bit will clear BOF1. If BO1RE (PCON2.3) is enabled, BOF1 indicates a BOD1 Reset occurred.

Bit 1: BOF0, BOF0 (Reset) Flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing ":0" is no operation.
1: This bit is only set by hardware when VDD meets BOD0 monitored level. Writing **"1"** on this bit will clear BOF0. If BO0RE (PCON2.1) is enabled, BOF0 indicates a BOD0 Reset occurred.

11.6. WDT Reset

When WDT is enabled to start the counter, WDTF will be set by WDT overflow. If WREN (WDTCR.7) is enabled, the WDT overflow will trigger a system reset that causes CPU to restart. Software can read the WDTF to recognize the WDT reset occurred.

PCON1: Power Control Register 1

SFR Page = 0~F & P
SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 0: WDTF, WDT Overflow/Reset Flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing ":0" is no operation.
1: This bit is only set by hardware when WDT overflows. Writing **"1"** on this bit will clear WDTF. If WREN (WDTCR.7) is set, WDTF indicates a WDT Reset occurred.

11.7. Illegal Address Reset

In **MG82F6B08 / 6B001/ 6B104**, if software program runs to illegal address such as over program ROM limitation, it triggers a RESET to CPU.

11.8. Stack Pointer Warning Reset

SPHB: Stack Pointer High Boundary

SFR Page = P Only
SFR Address = 0x53 RESET = 1111-1111

7	6	5	4	3	2	1	0
1	1	1	1	SPHB.3	SPHB.2	SPHB.1	SPHB.0
R	R	R	R	R/W	R/W	R/W	R/W

SPHB, it is used for the detection boundary of Stack Pointer warning.
If SPHB == 1111-1111, SPWF will be set when SP ≥ 1111-1111.
If SPHB == 1111-0000, SPWF will be set when SP ≥ 1111-0000.

PCON1: Power Control Register 1

SFR Page = 0~F & P
SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 3: SPWF, SP Warning Flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing ":0" is no operation.
1: This bit is only set by hardware when SP ≥ SPHB. Writing **"1"** on this bit will clear SPWF when SP < SPHB.

PCON3: Power Control Register 3

SFR Page = **P Only**
SFR Address = 0x45 POR = 0000-0000

7	6	5	4	3	2	1	0
IVREN	0	0	SPWRE	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 4: SPWRE, SPWF trigger a MCU reset
0: Disable SPWF to trigger a MCU reset.
1: Enable SPWF to trigger a MCU reset.

12. Power Management

The **MG82F6B08 / 6B001/ 6B104** supports two power monitor modules, Brown-Out Detector 0 (BOD0) and Brown-Out Detector 1 (BOD1), and 7 power-reducing modes: Idle mode, Power-down mode, Slow mode, Sub-Clock mode, RTC mode, Watch mode and Monitor mode.

BOD0 and BOD1 report the chip power status on the flags, BOF0 and BOF1, which provide the capability to interrupt CPU or to reset CPU by software configured. The seven power-reducing modes provide the different power-saving scheme for chip application. These modes are accessed through the CKCON0, CKCON2, CKCON3, CKCON4, , PCON0, PCON1, PCON2, PCON3, PCON4, RTCCR and WDTCR register.

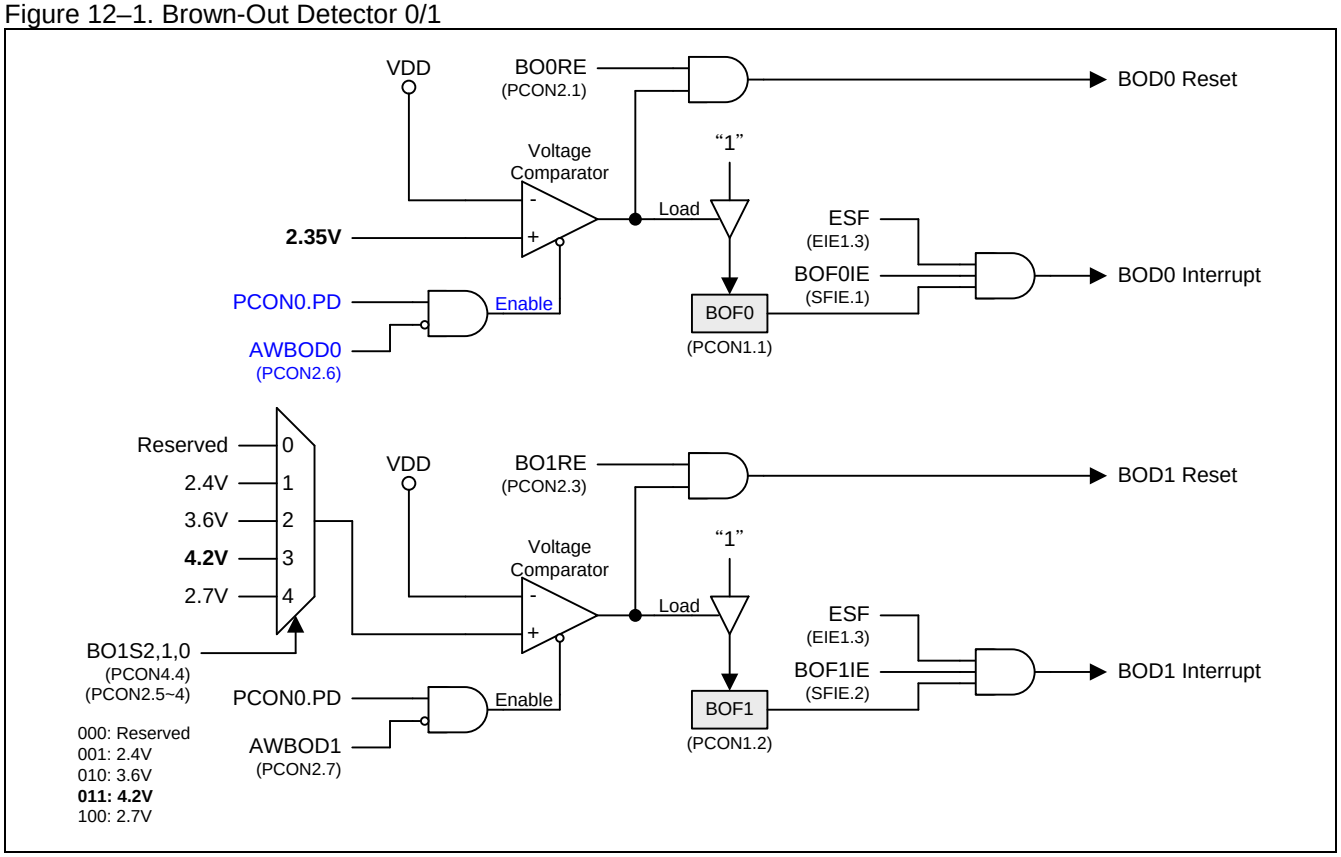
12.1. Brown-Out Detector

In **MG82F6B08 / 6B001/ 6B104**, there are two Brown-Out Detectors (BOD0 & BOD1) to monitor VDD power. Figure 12–1 shows the functional diagram of BOD0 and BOD1. BOD0 services the fixed detection level at VDD=**2.35V** and BOD1 detects the software selection levels (4.2V/3.6V/2.4V/2.7V) on VDD. Associated flag, BOF0 (PCON1.1), is set when BOD0 meets the detection level. If both of ESF (EIE1.3) and BOF0IE (SFIE.1) are enabled, a set BOF0 will generate a system flag interrupt. It can interrupt CPU either CPU in normal mode or idle mode. The BOD1 has the same flag function, BOF1, and same interrupt function. The BOD1 interrupt also wakes up CPU in power down mode if AWBOD1 (PCON2.7) is enabled.

If BO0RE (PCON2.1) is enabled, the BOD0 event will trigger a system reset and set BOF0 to indicate a BOD0 Reset occurred. The BOD0 reset restart the CPU either CPU in normal mode or idle mode. BOD1 also has the same reset capability with associated control bit, BO1RE (PCON2.3). The BOD1 reset also restart CPU in power down mode if AWBOD1 (PCON2.7) is enabled in BOD1 reset operation.

To reduce power consumption, software may clear EBOD1 (PCON2.2) to disable BOD1 if the BOD1 is not applied in user application.

Figure 12–1. Brown-Out Detector 0/1



12.2. Power Saving Mode

12.2.1. Slow Mode

The alternative to save the operating power is to slow the MCU's operating speed by programming SCKS2~SCKS0 bits (in CKCON0 register, see Section "8 System Clock") to a non-0/0/0 value. The user should examine which program segments are suitable for lower operating speed. In principle, the lower operating speed should not affect the system's normal function. Then, restore its normal speed in the other program segments.

12.2.2. Sub-Clock Mode

The alternative to slow down the MCU's operating speed by programming OSCS1~0 can select the ILRCO for system clock. The 32KHz ILRCO provides the MCU to operates in an ultra-low speed and low power operation. Additional programming SCKS2~SCKS0 bits (in CKCON0 register, see Section "8 System Clock"), the user could put the MCU speed down to 250Hz slowest.

12.2.3. RTC Mode

The **MG82F6B08 / 6B001/ 6B104** has a simple RTC module that allows a user to continue running an accurate timer while the rest of the device is powered-down. In RTC mode, the RTC module behaves a "Clock" function and can be a wake-up source from chip power down by RTC overflow rate. Please refer Section "10 Real-Time-Clock (RTC)/System-Timer" for more detail information.

12.2.4. Watch Mode

If Watch-Dog-Timer is enabled and NSW is set, Watch-Dog-Timer will keep running in power down mode to support an auto-wakeup function, which named Watch Mode in **MG82F6B08 / 6B001/ 6B104**. When WDT overflows, set WDTF and wakeup CPU from interrupt or system reset by software configured. The maximum wakeup period is about 2 seconds that is defined by WDT pre-scaler. Please refer Section "9 Watch Dog Timer (WDT)" and Section "14 Interrupt" for more detail information.

12.2.5. Monitor Mode

If AWBOD1 (PCON2.3) is set, BOD1 will keep VDD monitor in power down mode. It is the Monitor Mode in **MG82F6B08 / 6B001/ 6B104**. When BOD1 meets the detection level, set BOF1 and wakeup CPU from interrupt or system reset by software configured. Please refer Section "12.1 Brown-Out Detector" and Section "14 Interrupt" for more detail information.

12.2.6. Idle Mode

Setting the IDL bit in PCON enters idle mode. Idle mode halts the internal CPU clock. The CPU state is preserved in its entirety, including the RAM, stack pointer, program counter, program status word, and accumulator. The Port pins hold the logical states they had at the time that Idle was activated. Idle mode leaves the peripherals running in order to allow them to wake up the CPU when an interrupt is generated. Timer 0, Timer 1, Timer 2, KBI, ADC, AC0, S0, TWI0/I2C0, RTC, BOD0 and BOD1 will continue to function during Idle mode. PCA Timer and WDT are conditional enabled during Idle mode to wake up CPU. Any enabled interrupt source or reset may terminate Idle mode. When exiting Idle mode with an interrupt, the interrupt will immediately be serviced, and following RETI, the next instruction to be executed will be the one following the instruction that put the device into Idle.

The ADC or analog comparator input channels must be set to "***Analog Input Only***" when MCU is in idle mode or power-down mode to reduce power consumption.

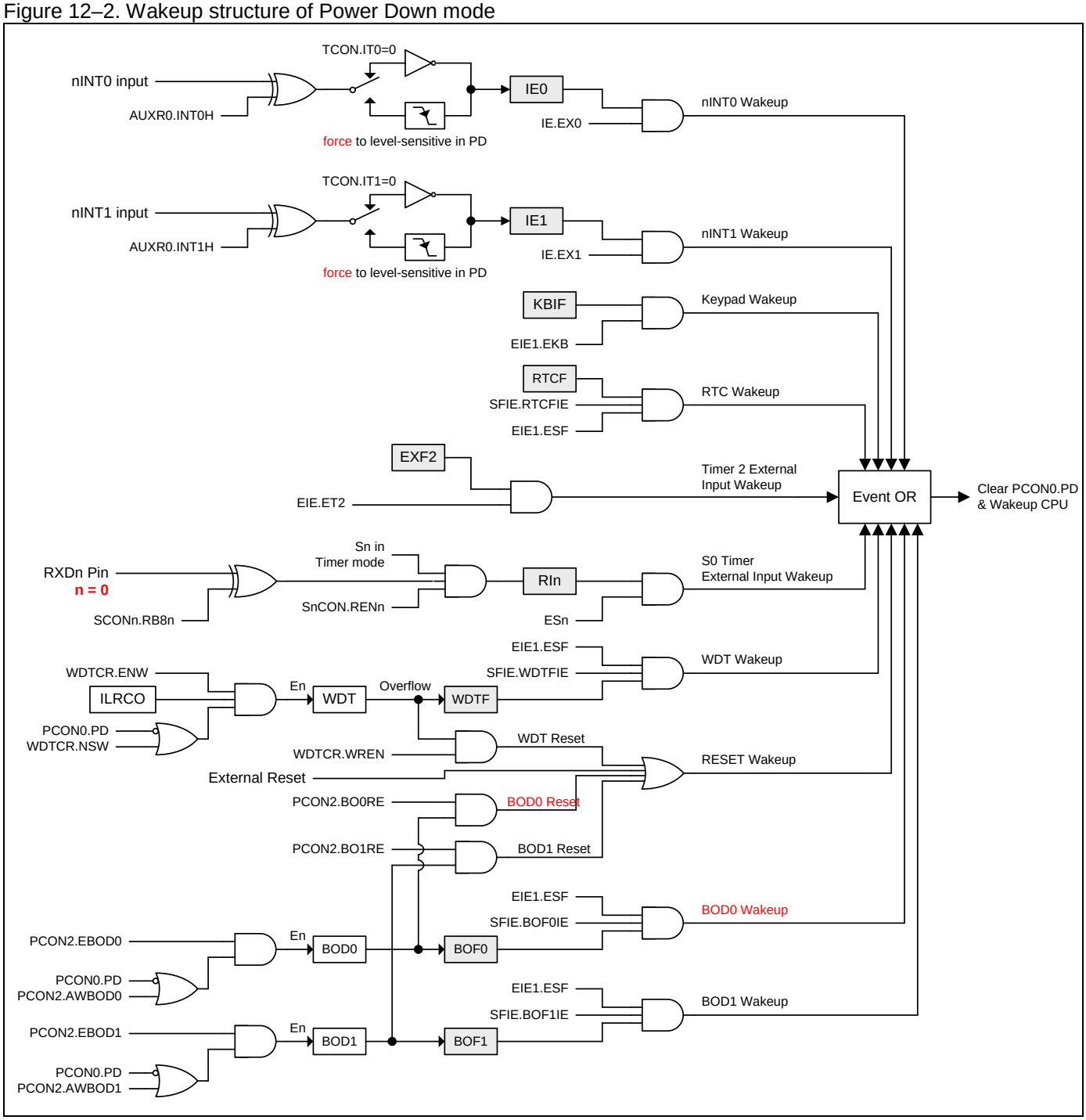
12.2.7. Power-down Mode

Setting the PD bit in PCON0 enters Power-down mode. Power-down mode stops the oscillator and powers down the Flash memory in order to minimize power consumption. Only the power-on circuitry will continue to draw power during Power-down. During Power-down the power supply voltage may be reduced to the RAM keep-alive voltage. The RAM contents will be retained; however, the SFR contents are not guaranteed once VDD has been reduced. Power-down may be exited by external reset, enabled external interrupts, enabled KBI, enabled RTC (RTC mode), enabled BOD1 (monitor mode) or enabled Non-Stop WDT (watch mode).

The user should not attempt to enter (or re-enter) the power-down mode for a minimum of 4 µs until after one of the following conditions has occurred: Start of code execution (after any type of reset), or Exit from power-down mode. To ensure minimum power consumption in power down mode, software must confirm all I/O not in floating state.

Figure 12–2 shows the wakeup mechanism of power-down mode in **MG82F6B08 / 6B001/ 6B104**.

Figure 12–2. Wakeup structure of Power Down mode



12.2.8. Interrupt Recovery from Power-down

Two external interrupts may be configured to terminate Power-down mode. External interrupts nINT0 and nINT1 may be used to exit Power-down. To wake up by external interrupt nINT0 and nINT1, the interrupt must be enabled and configured for level-sensitive operation. If the enabled external interrupts are configured to edge-sensitive operation (Falling or Rising), they will be **forced** to level-sensitive operation (Low level or High level) by hardware in power-down mode.

When terminating Power-down by an interrupt, the wake up period is internally timed. At the falling edge on the interrupt pin, Power-down is exited, the oscillator is restarted, and an internal timer begins counting. The internal clock will not be allowed to propagate and the CPU will not resume execution until after the timer has reached internal counter full. After the timeout period, the interrupt service routine will begin. To prevent the interrupt from re-triggering, the ISR should disable the interrupt before returning. The interrupt pin should be held low until the device has timed out and begun executing.

12.2.9. Reset Recovery from Power-down

Wakeup from Power-down through an external reset is similar to the interrupt. At the rising edge of RST, Power-down is exited, the oscillator is restarted, and an internal timer begins counting. The internal clock will not be allowed to propagate to the CPU until after the timer has reached internal counter full. The RST pin must be held high for longer than the timeout period to ensure that the device is reset properly. The device will begin executing once RST is brought low.

It should be noted that when idle is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

12.2.10. KBI wakeup Recovery from Power-down

The Keypad Interrupt of **MG82F6B08 / 6B001/ 6B104**, KBI.7~0 have wakeup CPU capability that are enabled by the control registers in KBI module.

Wakeup from Power-down through an enabled wakeup KBI is same to the interrupt. At the matched condition of enabled KBI pattern and enabled KBI interrupt (EIE1.5, EKB), Power-down is exited, the oscillator is restarted, and an internal timer begins counting. The internal clock will not be allowed to propagate to the CPU until after the timer has reached internal counter full. After the timeout period, CPU will meet a KBI interrupt and execute the interrupt service routine.

12.3. Power Control Register

PCON0: Power Control Register 0

SFR Page = 0~F & P POR = 0001-0000
SFR Address = 0x87 RESET = 000X-0000

7	6	5	4	3	2	1	0
SMOD1	SMOD0	GF	POF0	GF1	GF0	PD	IDL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 4: POF0, Power-On Flag 0.
0: This bit must be cleared by software writing one to it.
1: This bit is set by hardware if a Power-On Reset occurs.

Bit 1: PD, Power-Down control bit.
0: This bit could be cleared by CPU or any exited power-down event.
1: Setting this bit activates power down operation.

Bit 0: IDL, Idle mode control bit.
0: This bit could be cleared by CPU or any exited Idle mode event.
1: Setting this bit activates idle mode operation.

PCON1: Power Control Register 1

SFR Page = 0~F & P
SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SWRF, Software Reset Flag.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if a Software Reset occurs.

Bit 6: EXRF, External Reset Flag.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if an External Reset occurs.

Bit 4: RTCF, RTC overflow flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing "0" is no operation.
1: This bit is only set by hardware when RTCCT overflows. Writing **"1"** on this bit will clear RTCF.

Bit 3: SPWF, SP Warning Flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing ":0" is no operation.
1: This bit is only set by hardware when SP ≥ SPHB. Writing **"1"** on this bit will clear SPWF when SP < SPHB.

Bit 2: BOF1, Brown-Out Detection flag 1.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if the operating voltage matches the detection level of Brown-Out Detector 1 (4.2V/3.6V/2.7V/2.4V).

Bit 1: BOF0, Brown-Out Detection flag 0.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if the operating voltage matches the detection level of Brown-Out Detector 0 (**2.35V**).

Bit 0: WDTF, WDT overflow flag.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if a WDT overflow occurs.

PCON2: Power Control Register 2

SFR Page = **P Only**
SFR Address = 0x44 POR = 0000-0101

7	6	5	4	3	2	1	0
AWBOD1	0	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: AWBOD1, Awaked BOD1 in PD mode.
0: BOD1 is disabled in power-down mode.
1: BOD1 keeps operation in power-down mode.

Bit 6: Reserved. Software must write “0” on this bit when PCON2 is written.

Bit 5~4: BO1S[1:0]. Brown-Out detector 1 monitored level Selection. (BO1S2 at PCON4.4)

BO1S[2:0]	BOD1 detecting level
0 0 0	Reserved
0 0 1	2.4V
0 1 0	3.6V
0 1 1	4.2V
1 0 0	2.7V
Others	Reserved

Bit 3: BO1RE, BOD1 Reset Enabled.
0: Disable BOD1 to trigger a system reset when BOF1 is set.
1: Enable BOD1 to trigger a system reset when BOF1 is set.

Bit 2: EBOD1, Enable BOD1 that monitors VDD power dropped at a BO1S1~0 specified voltage level.
0: Disable BOD1 to slow down the chip power consumption.
1: Enable BOD1 to monitor VDD power dropped.

Bit 1: BO0RE, BOD0 Reset Enabled.
0: Disable BOD0 to trigger a system reset when BOF0 is set.
1: Enable BOD0 to trigger a system reset when BOF0 is set (VDD meets 2.35V).

Bit 0: Reserved. Software must write “**1**” on this bit when PCON2 is written.

PCON3: Power Control Register 3

SFR Page = **P Only**
SFR Address = 0x45 POR = 0000-0000

7	6	5	4	3	2	1	0
IVREN	0	0	SPWRE	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: IVREN, Internal Voltage Reference Enable.
0: Disable on-chip IVR (1.4V).
1: Enable on-chip IVR (1.4V).

Bit 6~5: Reserved. Software must write “0” on these bits when PCON3 is written.

Bit 4: SPWRE, SPWF trigger a MCU reset.
0: Disable SPWF to trigger a MCU reset.
1: Enable SPWF to trigger a MCU reset.

Bit 3~0: Reserved. Software must write “0” on these bits when PCON3 is written.

PCON4: Power Control Register 4

SFR Page = **P Only**

SFR Address = 0x46 POR = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	BO1S2	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: BOD1 monitored level selection bit 2.

13. Configurable I/O Ports (GPIO)

The **MG82F6B08 / 6B001/ 6B104** has following I/O ports: P1.0, P3.0, P3.1, P3.2, P3.3, P4.4, P4.5, P4.6 and P4.7. If disable external reset function, P4.7 output function is valid. The exact number of I/O pins available depends upon the package types. See Table 13–1.

Table 13–1. Number of I/O Pins Available

Package Type	I/O Pins	Number of I/O ports
10-pin	P1.0, P3.0, P3.1, P3.3, P4.4, P4.5, P4.6, P4.7	8 or 7 (RST selected)
8-pin	P3.0, P3.1, P3.3, P4.4, P4.5, P4.6	6
8-pin (MG82F6B104)	P3.0, P3.1, P3.2, P3.3, P4.4, P4.5	6

13.1. IO Structure

The I/O operating modes are distinguished two groups in **MG82F6B08 / 6B001/ 6B104**. The first group is only for Port 3 to support four configurations on I/O operating. These are: quasi-bidirectional (standard 8051 I/O port), push-pull output, input-only (high-impedance input) and open-drain output. The Port 3 default setting is quasi-bidirectional mode with weakly pull-up resistance.

All other general port pins belong to the second group. They can be programmed to four operating modes, which include analog input only, open-drain output with pull-up resistor, open-drain output and push-pull output. The default setting of this group I/O is analog input only, which means the port pin in high impedance state.

Following sections describe the configuration of the all types I/O mode.

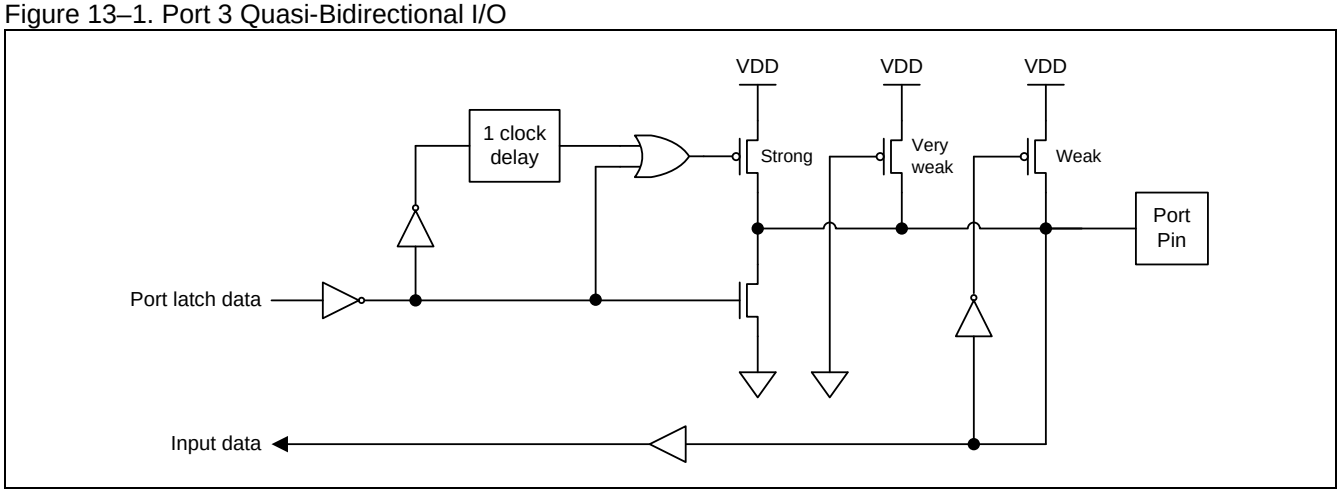
13.1.1. Port 3 Quasi-Bidirectional IO Structure

Port 3 pins in quasi-bidirectional mode are similar to the standard 8051 port pins. A quasi-bidirectional port can be used as an input and output without the need to reconfigure the port. This is possible because when the port outputs a logic high, it is weakly driven, allowing an external device to pull the pin low. When the pin outputs low, it is driven strongly and able to sink a large current. There are three pull-up transistors in the quasi-bidirectional output that serve different purposes.

One of these pull-ups, called the “very weak” pull-up, is turned on whenever the port register for the pin contains a logic “1”. This very weak pull-up sources a very small current that will pull the pin high if it is left floating. A second pull-up, called the “weak” pull-up, is turned on when the port register for the pin contains a logic “1” and the pin itself is also at a logic “1” level. This pull-up provides the primary source current for a quasi-bidirectional pin that is outputting a 1. If this pin is pulled low by the external device, this weak pull-up turns off, and only the very weak pull-up remains on. In order to pull the pin low under these conditions, the external device has to sink enough current to over-power the weak pull-up and pull the port pin below its input threshold voltage. The third pull-up is referred to as the “strong” pull-up. This pull-up is used to speed up low-to-high transitions on a quasi-bidirectional port pin when the port register changes from a logic “0” to a logic “1”. When this occurs, the strong pull-up turns on for one CPU clocks, quickly pulling the port pin high.

The quasi-bidirectional port configuration is shown in Figure 13–1.

Figure 13–1. Port 3 Quasi-Bidirectional I/O

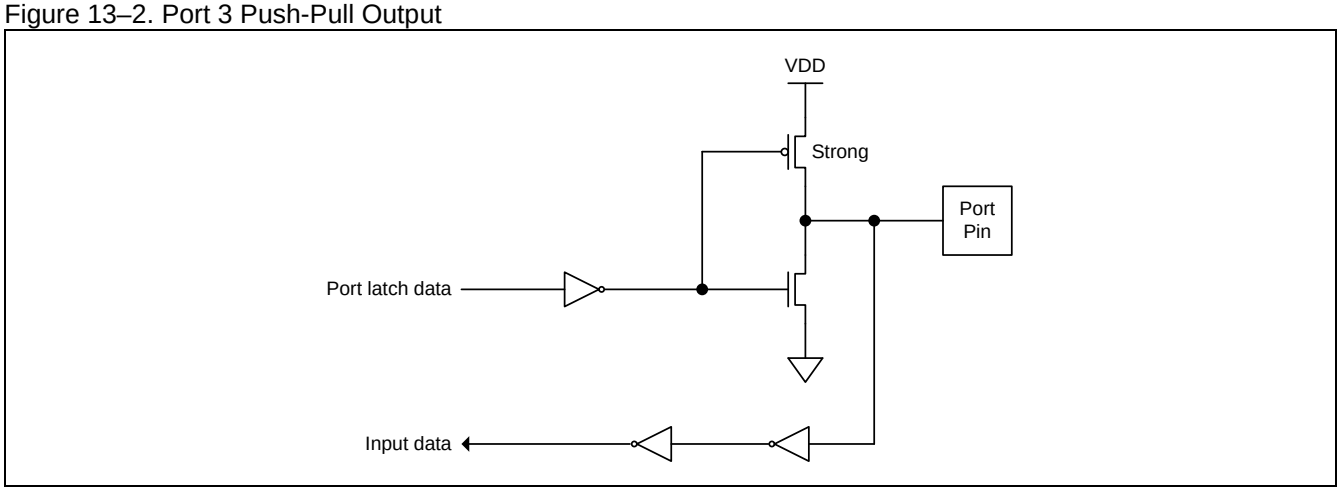


13.1.2. Port 3 Push-Pull Output Structure

The push-pull output configuration on Port 3 has the same pull-down structure as both the open-drain and the quasi-bidirectional output modes, but provides a continuous strong pull-up when the port register contains a logic “1”. The push-pull mode may be used when more source current is needed from a port output. In addition, the input path of the port pin in this configuration is also the same as quasi-bidirectional mode.

The push-pull port configuration is shown in Figure 13–2.

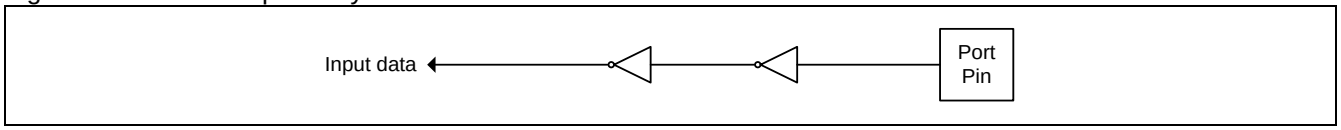
Figure 13–2. Port 3 Push-Pull Output



13.1.3. Port 3 Input-Only (High Impedance Input) Structure

The input-only configuration on Port 3 is an input without any pull-up resistors on the pin, as shown in Figure 13–3.

Figure 13–3. Port 3 Input-Only

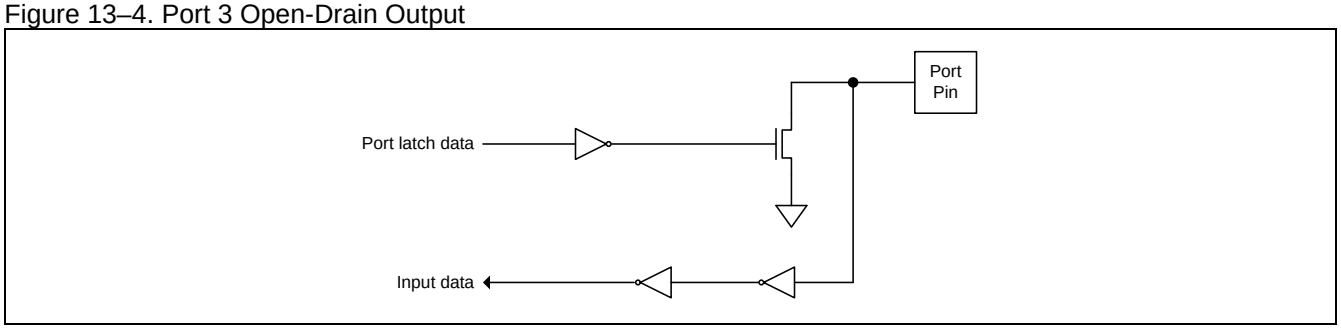


13.1.4. Port 3 Open-Drain Output Structure

The open-drain output configuration on Port 3 turns off all pull-ups and only drives the pull-down transistor of the port pin when the port register contains a logic “0”. To use this configuration in application, a port pin must have an external pull-up, typically a resistor tied to VDD. The pull-down for this mode is the same as for the quasi-bidirectional mode. In addition, the input path of the port pin in this configuration is also the same as quasi-bidirectional mode.

The open-drain port configuration is shown in Figure 13–4.

Figure 13–4. Port 3 Open-Drain Output

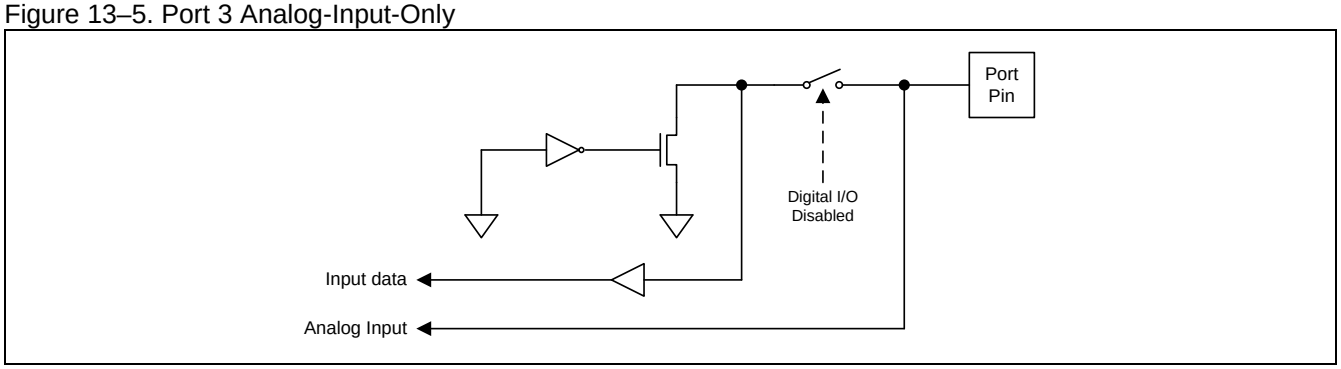


13.1.5. Port 3 Analog Input Structure

The analog-input-only configuration on port 3 is an analog-input-only without any digital function. For ADC or Analog Comparator input application, user may keep the port setting in this configuration. If apply the port pin to digital function, user must program the port pin to associated configuration.

The analog-input-only port configuration is shown in Figure 13–5.

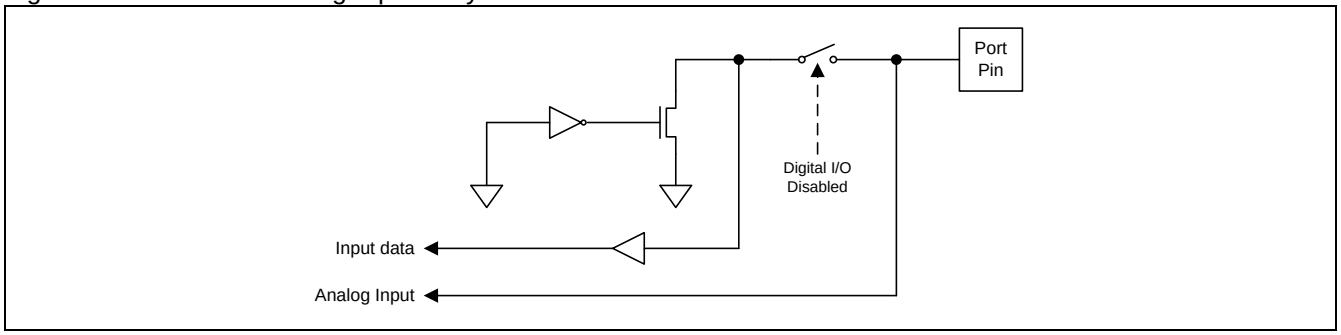
Figure 13–5. Port 3 Analog-Input-Only



13.1.6. General Analog Input Only Structure

The analog-input-only configuration on general port pins is the default setting. For ADC or Analog Comparator input application, user may keep the port setting in this configuration. If apply the port pin to digital function, user must program the port pin to associated configuration. The analog-input-only port configuration is shown in Figure 13–6.

Figure 13–6. General Analog-Input-Only

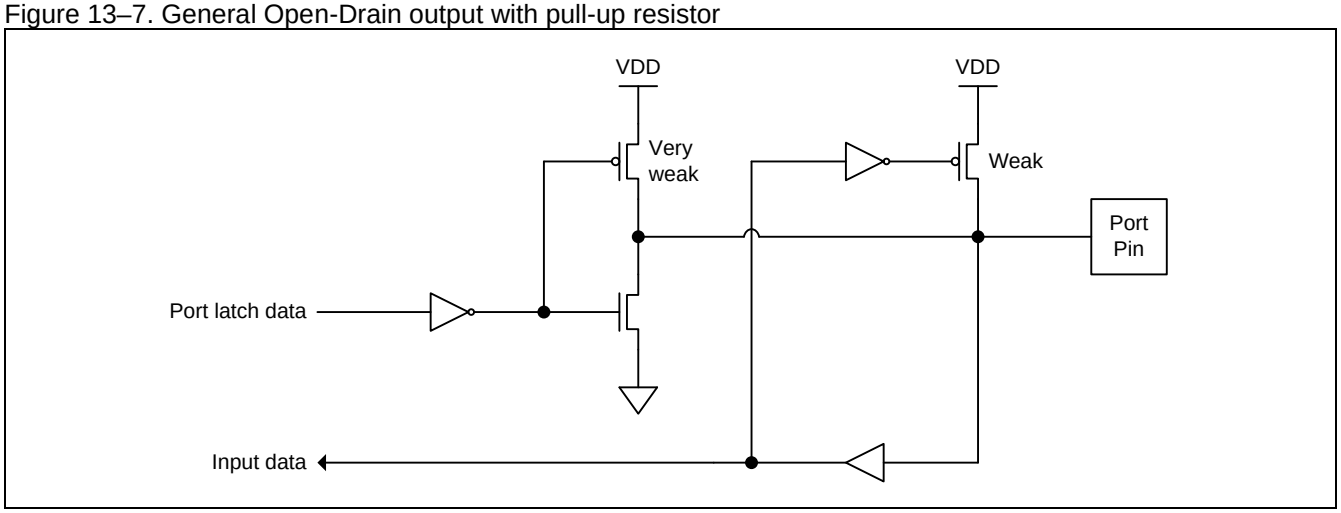


13.1.7. General Open-Drain Output with Pull-up Resistor Structure

The open-drain output with pull-up resistor configuration on general port pins enables the on-chip pull-up resistor in open-drain output mode.

The open-drain output with pull-up resistor port configuration is shown in Figure 13–7.

Figure 13–7. General Open-Drain output with pull-up resistor

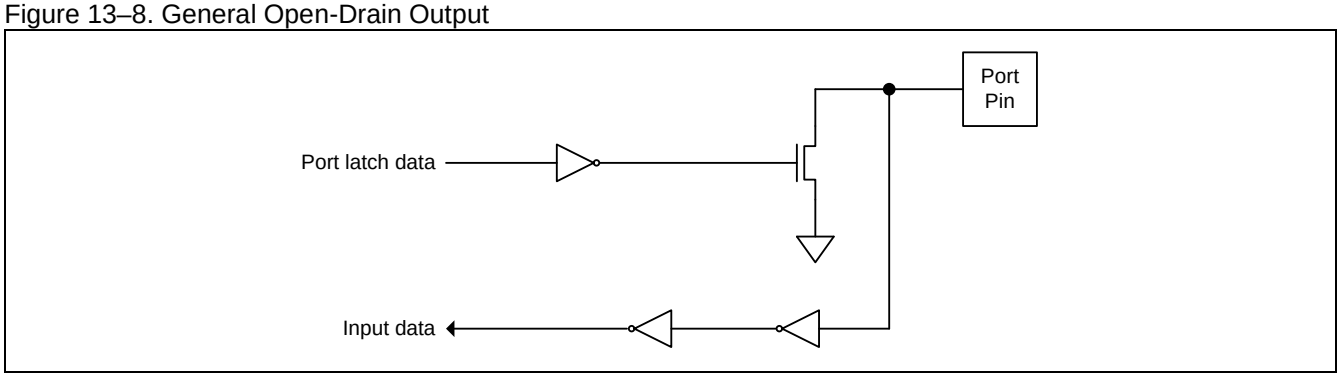


13.1.8. General Open-Drain Output Structure

The open-drain output configuration on general port pins is the same function as port 3 open-drain output mode.

The general open-drain port configuration is shown in Figure 13–8.

Figure 13–8. General Open-Drain Output



13.1.9. General Port Digital Input Configured

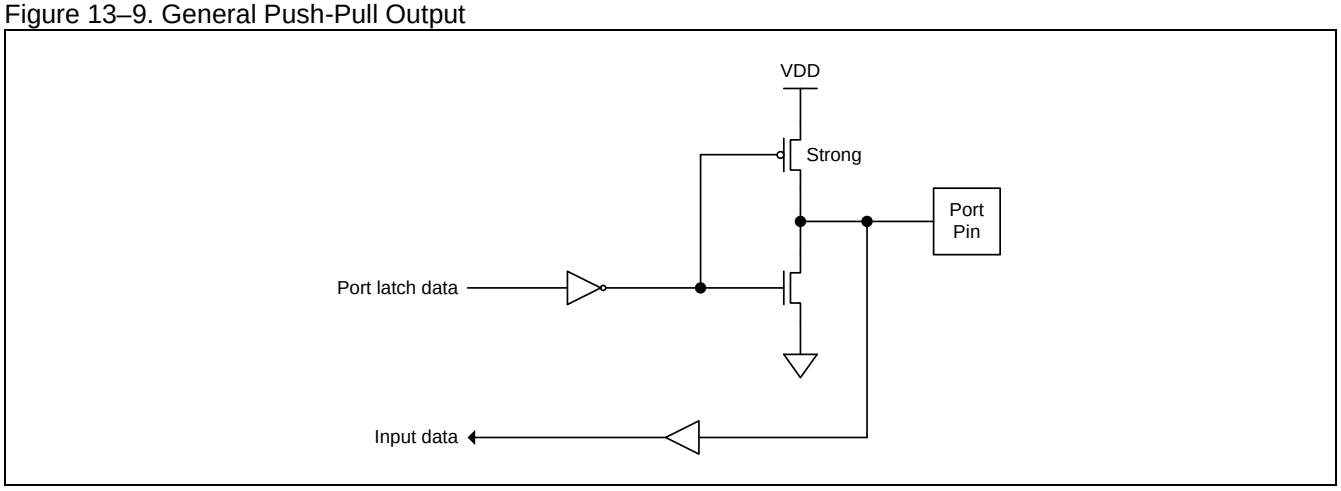
A Port pin is configured as a digital input by setting its output mode to “Open-Drain” and writing a logic “1” to the associated bit in the Port Data register. For example, P1.0 is configured as a digital input by setting P1M0.0 to a logic 0, P1M1.0 to a logic 0 and P1.0 to a logic 1.

13.1.10. General Push-Pull Output Structure

The push-pull output configuration on general port pins has the same function with port 3 push-pull output mode.

The push-pull port configuration is shown in Figure 13–9.

Figure 13–9. General Push-Pull Output



13.1.11. Port Pin Output Driving Strength Selection

The I/O of the **MG82F6B08 / 6B001/ 6B104** has two driving strength can be selected for different kinds of the application to match the output impedance. Please reference 13.2.4 Port Output Driving Strength Control Register.

13.1.12. Port Pin Output Fast Driving Selection

The I/O of the **MG82F6B08 / 6B001/ 6B104** has two driving speed can be selected for different kinds of the I/O frequency. Please reference 13.2.5 Port Output Fast Driving Control Register.

13.2. I/O Port Register

All I/O port pins on the **MG82F6B08 / 6B001/ 6B104** may be individually and independently configured by software to select its operating modes. Port 3 has four operating modes, as shown in Table 13–2. Two mode registers select the output type for each port 3 pin. Only Port 3 supports quasi-bidirectional mode and setting them to quasi-bidirectional mode after system reset.

Table 13–2. Port 3 Configuration Settings

P3xAM	P3M0.y	P3M1.y	Port Mode
0	**0**	**0**	**Quasi-Bidirectional (default)**
0	0	1	Push-Pull Output
0	1	0	Input Only (High Impedance Input)
0	1	1	Open-Drain Output
1	0	0	Analog Input Only

Where y=0~7 (port pin). The registers P3M0 and P3M1 are listed in each port description.
Where x= 0, 3 to define the control bits, P30AM and P33AM, use them to select the P3.0 and P3.3 on analog input mode only.

Other general port pins also support four operating modes, as shown in Table 13–3. Two mode registers select the I/O type for each port pin and setting to analog-input-only on these port pins after system reset.

Table 13–3. General Port Configuration Settings

PxM0.y	PxM1.y	Port Mode
0	**1**	**Analog Input Only (default)**
1	1	Open-Drain with Pull-up resistor
0	0	Open-Drain Output / General Digital Input (Port Pin set to “1”)
1	0	Push-Pull Output

Where x= **1, 4** (port number), and y=0~7 (port pin). The registers PxM0 and PxM1 are listed in each port description

13.2.1. Port 1 Register

P1: Port 1 Register

SFR Page = 0~F
SFR Address = 0x90 RESET = 1111-1111

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	P1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: Port 1 output data latch could be only set/cleared by CPU.

P1M0: Port 1 Mode Register 0

SFR Page = 0~F
SFR Address = 0x91 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	P1M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P1M1: Port 1 Mode Register 1

SFR Page = **0 only**
SFR Address = **0x92** RESET = 1111-1111

7	6	5	4	3	2	1	0
1	1	1	1	1	1	1	P1M1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

13.2.2. Port 3 Register

P3: Port 3 Register

SFR Page = 0~F
SFR Address = 0xB0 RESET = 1111-1111

7	6	5	4	3	2	1	0
1	1	1	1	P3.3	P3.2	P3.1	P3.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: Port 3 output data latch could be only set/cleared by CPU.

P3M0: Port 3 Mode Register 0

SFR Page = 0~F
SFR Address = 0xB1 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	P3M0.3	P3M0.2	P3M0.1	P3M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3M1: Port 3 Mode Register 1

SFR Page = 0~F
SFR Address = 0xB2 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	P3M1.3	P3M1.2	P3M1.1	P3M1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AUXR11: Auxiliary Register 11

SFR Page = **8 only**
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
P30AM	P33AM	0	0	0	POEM0	C0M0	C0OFS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: P30AM, P3.0 Analog input Mode enable.
0: The P3.0 GPIO mode is controlled by P3M0 and P3M1.
1: Force P3.0 to be analog input mode for the AIN4 input of ADC12.

Bit 6: P33AM, P3.3 Analog input Mode enable.
0: The P3.3 GPIO mode is controlled by P3M0 and P3M1.
1: Force P3.3 to be analog input mode for the AIN5 input of ADC12.

13.2.3. Port 4 Register

P4: Port 4 Register

SFR Page = 0~F
SFR Address = 0xE8 RESET = 1111-1111

7	6	5	4	3	2	1	0
P4.7	P4.6	P4.5	P4.4	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: Port 4 output data latch could be set/cleared by CPU.
P4.5 and P4.4 have the alternated function for OCD_SDA and OCD_SCL.
P4.7 has the alternated function for RST input.

P4M0: Port 4 Mode Register 0

SFR Page = 0 only
SFR Address = 0xB3 RESET = 1011-0000

7	6	5	4	3	2	1	0
P4M0.7	P4M0.6	P4M0.5	P4M0.4	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: When P4.7/RST use as port pin, it is not suggest to program it as Input to avoid MCU is locked in reset in bootup period when level high send into this pin.

P4M1: Port 4 Mode Register 1

SFR Page = **2 only**
SFR Address = **0x92** RESET = 1111-1111

7	6	5	4	3	2	1	0
P4M1.7	P4M1.6	P4M1.5	P4M1.4	1	1	1	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Note: When P4.7/RST use as port pin, it is not suggest to program it as Input to avoid MCU is locked in reset in bootup period when level high send into this pin.

13.2.4. Port Output Driving Strength Control Register

In **MG82F6B08 / 6B001/ 6B104,** all port pins have two driving strength selection by software configured except P4.7 Please refer to get the driving strength information on the port pins.

PDRVC0: Port Drive Control Register 0

SFR Page = **2 only**
SFR Address = **0xB4** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	P3DC0	0	0	0	P1DC0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 6: P3DC0, Port 3 output driving strength control on low nibble.
0: Select the P3.3 ~ P3.0 output with high driving strength.
1: Select the P3.3 ~ P3.0 output with low driving strength.

Bit 2: P1DC0, Port 1 output driving strength control on low nibble.
0: Select the P1.3 ~ P1.0 output with high driving strength.
1: Select the P1.3 ~ P1.0 output with low driving strength.

PDRVC1: Port Drive Control Register 1

SFR Page = **3 only**
SFR Address = **0xB4** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	0	P4DC1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~2: Reserved. Software must write “0” on this bit when PDRVC1 is written.

Bit 1: P4DC1, Port 4 output driving strength control on high nibble.
0: Select the P4.6 ~ P4.4 output with high driving strength.
1: Select the P4.6 ~ P4.4 output with low driving strength.

Bit 0: Reserved. Software must write “0” on this bit when PDRVC1 is written.

13.2.5. Port Output Fast Driving Control Register

In **MG82F6B08 / 6B001/ 6B104,** all port pins have two driving speed selection by software configured except P4.7. Please refer to get the driving strength information on the port pins.

P3FDC: Port 3 Fast Driving Control Register

SFR Page = 7 **only**
SFR Address = **0x92** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	P3FDC.3	P3FDC.2	P3FDC.1	P3FDC.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**RW**	**RW**

Bit 3~0: Port 3 output fast driving control could be only set/cleared by CPU.
0: Disable fast driving on port pin output.
1: Enable fast driving on port pin output.

P1FDC: Port 1 Fast Driving Control Register

SFR Page = 8 **only**
SFR Address = **0x92** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	0	0	P1FDC.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**RW**	**RW**

Bit 0: Port 1 output fast driving control could be only set/cleared by CPU.
0: Disable fast driving on port pin output.
1: Enable fast driving on port pin output.

P4FDC: Port 4 Fast Driving Control Register

SFR Page = A **only**
SFR Address = **0x92** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	P4FDC.6	P4FDC.5	P4FDC.4	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: Reserved. Software must write “0” on this bit when P4FDC is written.

Bit 6~4: Port 4 output fast driving control could be only set/cleared by CPU.
0: Disable fast driving on port pin output.
1: Enable fast driving on port pin output.

Bit 3~0: Reserved. Software must write “0” on this bit when P4FDC is written.

14. Interrupt

The **MG82F6B08 / 6B001/ 6B104** has **12** interrupt sources with a four-level interrupt structure. There are several SFRs associated with the four-level interrupt. They are the IE, IP0L, IP0H, EIE1, EIP1L, EIP1H. The IP0H (Interrupt Priority 0 High), EIP1H (Extended Interrupt Priority 1 High) and registers make the four-level interrupt structure possible. The four priority level interrupt structure allows great flexibility in handling these interrupt sources.

14.1. Interrupt Structure

Table 14–1 lists all the interrupt sources. The 'Request Bits' are the interrupt flags that will generate an interrupt if it is enabled by setting the 'Enable Bit'. Of course, the global enable bit EA (in IE0 register) should have been set previously. The 'Request Bits' can be set or cleared by software, with the same result as though it had been set or cleared by hardware. That is, interrupts can be generated or pending interrupts can be cancelled in software. The 'Priority Bits' determine the priority level for each interrupt. The 'Priority within Level' is the polling sequence used to resolve simultaneous requests of the same priority level. The 'Vector Address' is the entry point of an interrupt service routine in the program memory.

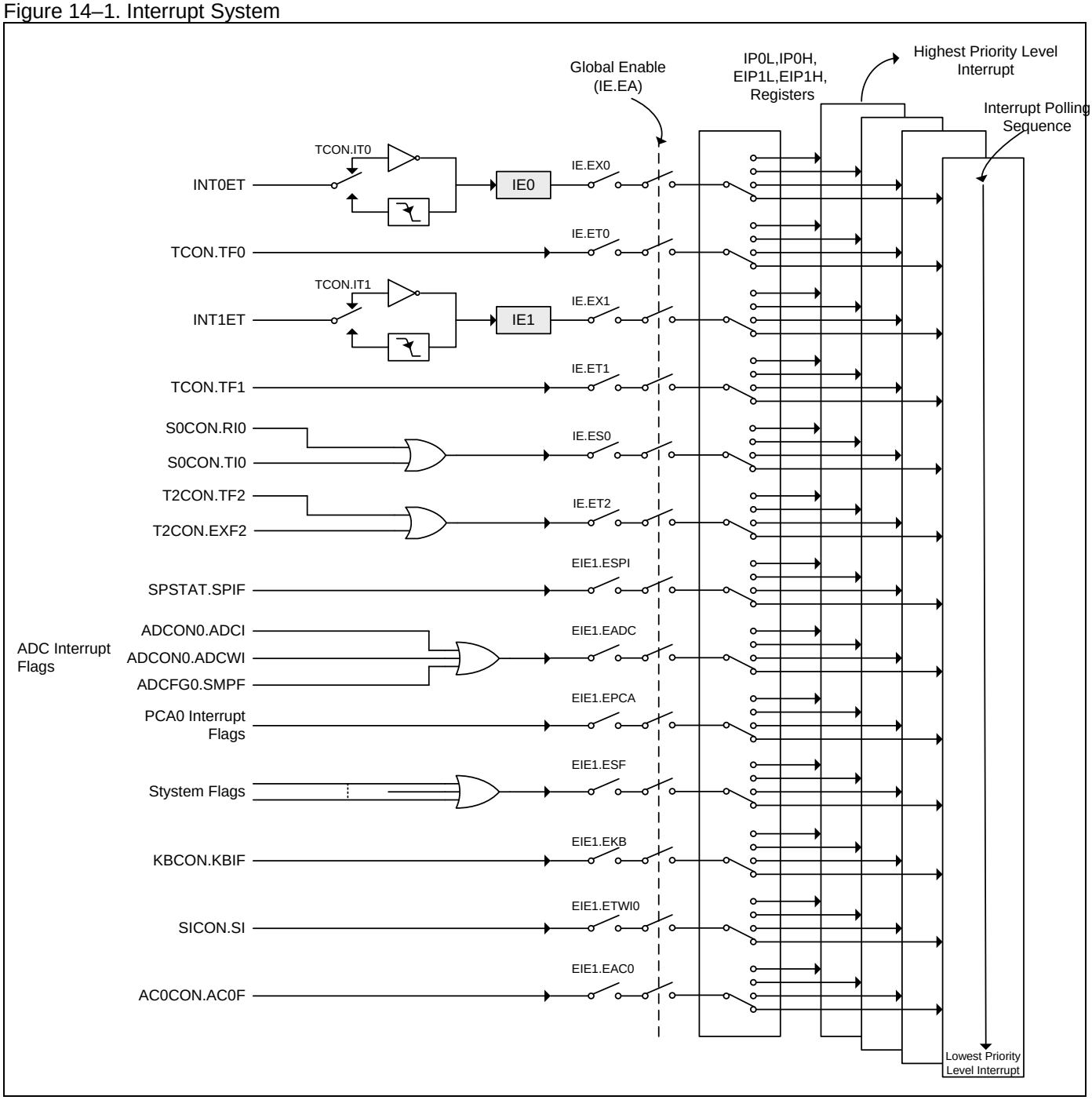
Figure 14–1 shows the interrupt system. Each of these interrupts will be briefly described in the following sections.

Table 14–1. Interrupt Sources

No	Source Name	Enable Bit	Request Bits	Priority Bits	Polling Priority	Vector Address
#0	External Interrupt 0, nINT0	EX0	IE0	[PX0H, PX0L]	(Highest)	0003H
#1	Timer 0	ET0	TF0	[PT0H, PT0L]	…	000Bh
#2	External Interrupt 1, nINT1	EX1	IE1	[PX1H, PX1L]	...	0013H
#3	Timer 1	ET1	TF1	[PT1H, PT1L]	**…**	001BH
#4	Serial Port 0	ES0	RI0, TI0	[PS0H, PS0L]	**…**	0023H
#5	Timer 2	ET2	TF2, EXF2 (TF2L)	[PT2H, PT2L]	**…**	002Bh
#6	Reserved	--	--	--	--	0033H
#7	SPI	ESPI	SPIF	[PSPIH, PSPIL]	**…**	003BH
#8	ADC	EADC	ADCI, ADCWI, SMPF	[PADCH, PADCL]	**…**	0043H
#9	PCA0	EPCA	CF, CCFn (n=0~7)	[PPCAH, PPCAL]	**…**	004Bh
#10	System Flag	ESF	(Note 1)	[PSFH, PSFL]	**…**	0053H
#11	Keypad Interrupt	EKB	KBIF	[PKBH, PKBL]	**…**	005BH
#12	TWI0/I2C0	ETWI0	SI	[PTWI0H, PTWI0L]	**…**	0063H
#13	Analog Comparator 0	EAC0	AC0F	[PAC0H, PAC0L]	**…**	006BH

Note 1: The System Flag interrupt flags include: WDTF, BOF0, BOF1, RTCF, SPWF, CFAIL, EEPF in PCON1, TI0 in S0CON, STAF and STOF in AUXR2.

Figure 14–1. Interrupt System



14.2. Interrupt Source

Table 14–2. Interrupt Source Flag

No	Source Name	Request Bits	Bit Location
#0	External Interrupt 0,nINT0	IE0	TCON.1
#1	Timer 0	TF0	TCON.5
#2	External Interrupt 1,nINT1	IE1	TCON.3
#3	Timer 1	TF1	TCON.7
#4	Serial Port 0	RI0, TI0	S0CON.0 S0CON.1
#5	Timer 2	TF2, EXF2, (TF2L)	T2CON.7 T2CON.6 T2CON.5
#6	Reserved	--	--
#7	SPI	SPIF	SPSTAT.7
#8	ADC	ADCI, ADCWI, SMPF	ADCON0.4 ADCON0.6 ADCFG0.2
#9	PCA0	CF, CCFn (n=0~3),	CCON.7 CCON.3~0
#10	System Flag	WDTF, BOF0, BOF1, SPWF, RTCF, STAF, STOF, CFAIL EEPF (TI0)	PCON1.0 PCON1.1 PCON1.2 PCON1.3 PCON1.4 AUXR2.7 AUXR2.6 ISPCR.4 ISPCR.0 S0CON.1
#11	Keypad Interrupt	KBIF	KBCON.0
#12	TWI0/I2C0	SI	SICON.3
#13	Analog Comparator 0	AC0F	AC0CON.4

The external interrupt nINT0 and nINT1 can each be either level-activated or transition-activated, depending on bits IT0 and IT1 in register TCON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON. When an external interrupt is generated, the flag that generated it is cleared by the hardware when the service routine is vectored to *only if the interrupt was transition –activated*, then the external requesting source is what controls the request flag, rather than the on-chip hardware.

The Timer0 and Timer1 interrupts are generated by TF0 and TF1, which are set by a rollover in their respective Timer/Counter registers in most cases. When a timer interrupt is generated, the flag that generated it is cleared by the on-chip hardware when the service routine is vectored to.

The serial port 0 interrupt is generated by the logical OR of RI0 and TI0. Neither of these flags is cleared by hardware when the service routine is vectored to. The service routine should poll RI0 and TI0 to determine which one to request service and it will be cleared by software.

The timer2 interrupt is generated by the logical OR of TF2 and EXF2. If the timer 2 in split mode, the TL2 overflow will set another interrupt flag, TF2L. Just the same as serial port, neither of these flags is cleared by hardware when the service routine is vectored to.

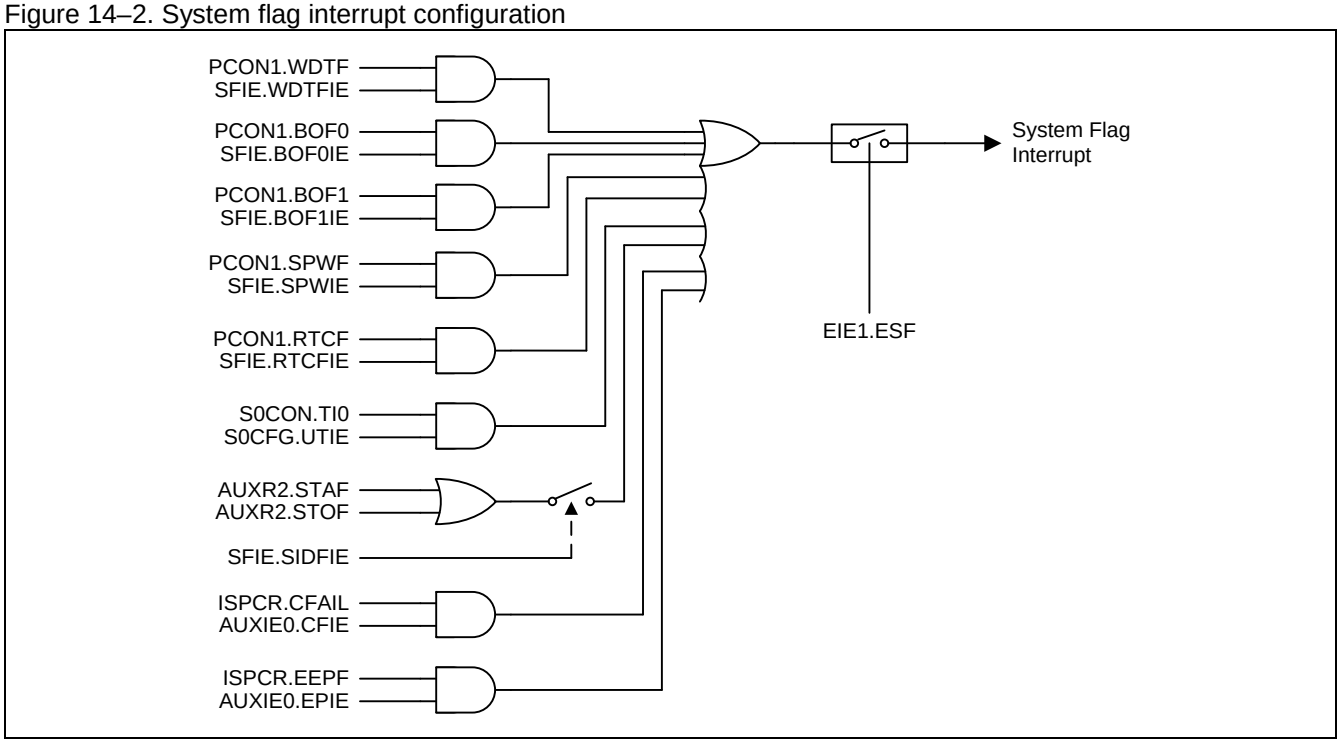
SPI interrupt is generated by SPIF in SPSTAT, which are set by SPI engine finishes a SPI transfer. It will not be cleared by hardware when the service routine is vectored to.

The ADC interrupt is generated by ADCI, ADCWI in ADCON0 and SMPF in ADCFG0. These flags will not be cleared by hardware when the service routine is vectored to.

The PCA0 interrupt is generated by the logical OR of CF, CCF3, CCF2, CCF1 and CCF0 in CCON. Neither of these flags is cleared by hardware when the service routine is vectored to. The service routine should poll these flags to determine which one to request service and it will be cleared by software.

The System Flag interrupt is generated by RTCF, BOF1, BOF0, WDTF, SPWF, CFAIL, EEPF, TI0, STAF and STOF. STAF and STOF are set by serial interface detection and stored in AUXR2. The Serial Port TI flag is optional to locate the interrupt vector shared with system flag interrupt which is enabled by UTIE set. The rest flags are stored in PCON1. RTCF is set by RTC counter overflow. BOF1 and BOF0 are set by on chip Brownout-Detector (BOD1 and BOD0) met the low voltage event. WDTF is set by Watch-Dog-Timer overflow. SPWF is set by SP monitor to indicate the warning for stack pointer overflow coming. ***CFAIL & EEPF***. These flags will not be cleared by hardware when the service routine is vectored to. Figure 14–2 shows the system flag interrupt configuration.

Figure 14–2. System flag interrupt configuration



The keypad interrupt is generated by KBCON.KBIF, which is set by Keypad module meets the input pattern. It will not be cleared by hardware when the service routine is vectored to.

The TWI0/I2C0 interrupt is generated by SI in SICON, which is set by TWI0/I2C0 engine detecting a new bus state updated. It will not be cleared by hardware when the service routine is vectored to.

The AC0 interrupt is generated by AC0F in AC0CON, which is set by AC0OUT changed detecting on rising, falling or dual edge. It will not be cleared by hardware when the service routine is vectored to.

All of the bits that generate interrupts can be set or cleared by software, with the same result as though it had been set or cleared by hardware. In other words, interrupts can be generated or pending interrupts can be canceled in software.

14.3. Interrupt Enable

Table 14–3. Interrupt Enable

No	Source Name	Enable Bit	Bit Location
#0	External Interrupt 0,nINT0	EX0	IE.0
#1	Timer 0	ET0	IE.1
#2	External Interrupt 1,nINT1	EX1	IE.2
#3	Timer 1	ET1	IE.3
#4	Serial Port 0	ES0	IE.4
#5	Timer 2	ET2	IE.5
#6	Reserved	--	--
#7	SPI	ESPI	EIE1.0
#8	ADC	EADC	EIE1.1
#9	PCA	EPCA	EIE1.2
#10	System Flag	ESF	EIE1.3
#11	Keypad Interrupt	EKB	EIE1.5
#12	TWI0/I2C0	ETWI0	EIE1.6
#13	Analog Comparator 0, AC0	EAC0	EIE1.7

There are **12** interrupt sources available in **MG82F6B08 / 6B001/ 6B104**. Each of these interrupt sources can be individually enabled or disabled by setting or clearing an interrupt enable bit in the registers IE, EIE1. IE also contains a global disable bit, EA, which can be cleared to disable all interrupts at once. If EA is set to ‘1’, the interrupts are individually enabled or disabled by their corresponding enable bits. If EA is cleared to ‘0’, all interrupts are disabled.

14.4. Interrupt Priority

The priority scheme for servicing the interrupts is the same as that for the 80C51, except there are four interrupt levels rather than two as on the 80C51. The Priority Bits (see Table 14–1) determine the priority level of each interrupt. IP0L, IP0H, EIP1L, EIP1H are combined to 4-level priority interrupt. Table 14–4 shows the bit values and priority levels associated with each combination.

Table 14–4. Interrupt Priority

{IPnH.x , IPnL.x}	Priority Level
11	1 (highest)
10	2
01	3
00	4

Each interrupt source has two corresponding bits to represent its priority. One is located in SFR named IPnH and the other in IPnL register. Higher-priority interrupt will be not interrupted by lower-priority interrupt request. If two interrupt requests of different priority levels are received simultaneously, the request of higher priority is serviced. If interrupt requests of the same priority level are received simultaneously, an internal polling sequence determine which request is serviced. Table 14–2 shows the internal polling sequence in the same priority level and the interrupt vector address.

14.5. Interrupt Process

Each interrupt flag is sampled at every system clock cycle. The samples are polled during the next system clock. If one of the flags was in a set condition at first cycle, the second cycle (polling cycle) will find it and the interrupt system will generate a hardware LCALL to the appropriate service routine as long as it is not blocked by any of the following conditions.

Block conditions:

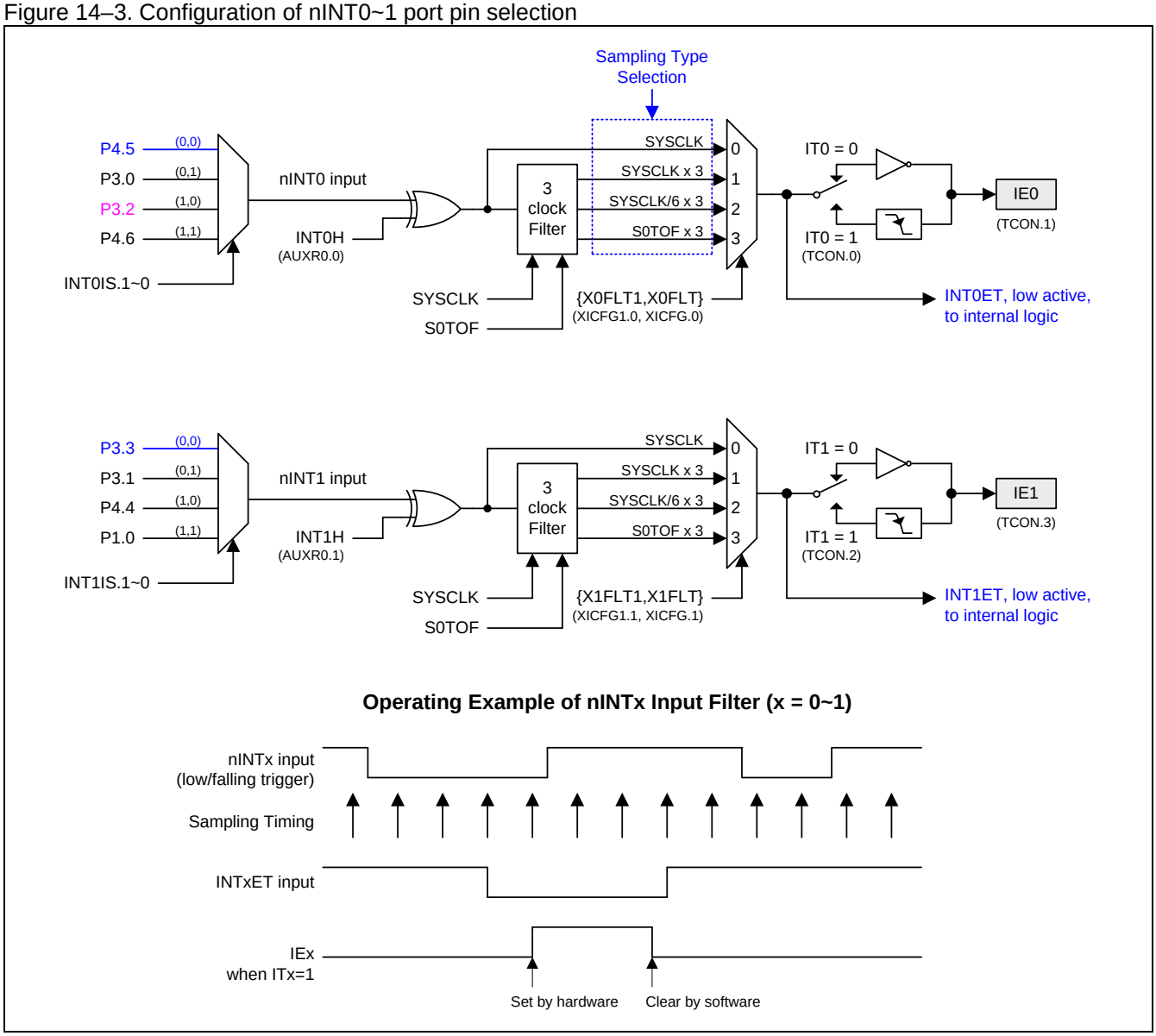
- An interrupt of equal or higher priority level is already in progress.
- The current cycle (polling cycle) is not the final cycle in the execution of the instruction in progress.
- The instruction in progress is RETI or any write to the IE, IP0L, IPH, EIE1, EIP1L, EIP1H registers.

Any of these three conditions will block the generation of the hardware LCALL to the interrupt service routine. Condition 2 ensures that the instruction in progress will be completed before vectoring into any service routine. Condition 3 ensures that if the instruction in progress is RETI or any access to IE or IP, then at least one or more instruction will be executed before any interrupt is vectored to.

14.6. nINTx Input Source Selection and input filter (x=0~1)

The **MG82F6B08 / 6B001/ 6B104** provides flexible nINT0, nINT1 source selection to share the port pin inputs for different system design. These pins support digital filters to filter the noise, it also can be the input trigger for many other modules, such as Timer, ADC, PCA, I2C, etc.

Figure 14–3. Configuration of nINT0~1 port pin selection



14.7. Interrupt Register

TCON: Timer/Counter Control Register

SFR Page = 0~F

SFR Address = 0x88 RESET = 0000-0000

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 3: IE1, Interrupt 1 (nINT1) Edge flag.
0: Cleared when interrupt processed on if transition-activated.
1: Set by hardware when external interrupt 1 (nINT1) edge is detected (transmitted or level-activated).

Bit 2: IT1: Interrupt 1 (nINT1) Type control bit.
0: Cleared by software to specify low level triggered external interrupt 1 (nINT1). If INT1H (AUXR0.1) is set, this bit specifies high level triggered on nINT1.
1: Set by software to specify falling edge triggered external interrupt 1 (nINT1). If INT1H (AUXR0.1) is set, this bit specifies rising edge triggered on nINT1.

Bit 1: IE0, Interrupt 0 (nINT0) Edge flag.
0: Cleared when interrupt processed on if transition-activated.
1: Set by hardware when external interrupt 0 (nINT0) edge is detected (transmitted or level-activated).

Bit 0: IT0: Interrupt 0 (nINT0) Type control bit.
0: Cleared by software to specify low level triggered external interrupt 0 (nINT0). If INT0H (AUXR0.0) is set, this bit specifies high level triggered on nINT0.
1: Set by software to specify falling edge triggered external interrupt 0 (nINT0). If INT0H (AUXR0.0) is set, this bit specifies rising edge triggered on nINT0.

IE: Interrupt Enable Register

SFR Page = 0~F

SFR Address = 0xA8 RESET = 0000-0000

7	6	5	4	3	2	1	0
EA	0	ET2	ES0	ET1	EX1	ET0	EX0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: EA, All interrupts enable register.
0: Global disables all interrupts.
1: Global enables all interrupts.

Bit 5: ET2, Timer 2 interrupt enable register.
0: Disable Timer 2 interrupt.
1: Enable Timer 2 interrupt.

Bit 4: ES, Serial port 0 interrupt (UART0) enable register.
0: Disable serial port 0 interrupt.
1: Enable serial port 0 interrupt.

Bit 3: ET1, Timer 1 interrupt enable register.
0: Disable Timer 1 interrupt.
1: Enable Timer 1 interrupt.

Bit 2: EX1, External interrupt 1 (nINT1) enable register.
0: Disable external interrupt 1.
1: Enable external interrupt 1.

Bit 1: ET0, Timer 0 interrupt enable register.
0: Disable Timer 0 interrupt.
1: Enable Timer 0 interrupt.

Bit 0: EX0, External interrupt 0 (nINT0) enable register.
0: Disable external interrupt 0.
1: Enable external interrupt 0.

AUXR0: Auxiliary Register 0

SFR Page = 0~F
SFR Address = 0xA1 RESET = 0000-0000

7	6	5	4	3	2	1	0
P45OC1	P45OC0	0	PBKF	0	0	INT1H	INT0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 4: PBKF, PWM Break Flag. This bit is set by PWM break source enabled. If this flag is set, the enabled PWM channel 0~5 will be blocked and the output pins keep the original GPIO state.
0: There is no PWM Break event happened. It is only cleared by software.
1: There is a PWM Break event happened or software triggers a PWM Break.

Bit 1: INT1H, INT1 High/Rising trigger enable.
0: Remain INT1 triggered on low level or falling edge on selected port pin input.
1: Set INT1 triggered on high level or rising edge on selected port pin input.

Bit 0: INT0H, INT0 High/Rising trigger enable.
0: Remain INT0 triggered on low level or falling edge on selected port pin input.
1: Set INT0 triggered on high level or rising edge on selected port pin input.

IP0L: Interrupt Priority 0 Low Register

SFR Page = 0~F
SFR Address = 0xB8 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	PT2L	PSL	PT1L	PX1L	PT0L	PX0L
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 5: PT2L, Timer 2 interrupt priority-L register.
Bit 4: PSL, Serial port interrupt priority-L register.
Bit 3: PT1L, Timer 1 interrupt priority-L register.
Bit 2: PX1L, external interrupt 1 priority-L register.
Bit 1: PT0L, Timer 0 interrupt priority-L register.
Bit 0: PX0L, external interrupt 0 priority-L register.

IP0H: Interrupt Priority 0 High Register

SFR Page = 0~F
SFR Address = 0xB7 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	PT2H	PSH	PT1H	PX1H	PT0H	PX0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 5: PT2H, Timer 2 interrupt priority-H register.
Bit 4: PSH, Serial port interrupt priority-H register.
Bit 3: PT1H, Timer 1 interrupt priority-H register.
Bit 2: PX1H, external interrupt 1 priority-H register.
Bit 1: PT0H, Timer 0 interrupt priority-H register.
Bit 0: PX0H, external interrupt 0 priority-H register.

EIE1: Extended Interrupt Enable 1 Register

SFR Page = 0~F
SFR Address = 0xAD RESET = 0000-0000

7	6	5	4	3	2	1	0
EAC0	ETWI0	EKB	0	ESF	EPCA	EADC	ESPI
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: EAC0, Enable Analog Comparator 0 (AC0) Interrupt.
0: Disable AC0 interrupt.
1: Enable AC0 interrupt.

Bit 6: ETWI0, Enable TWI0/I2C0 interrupt.
0: Disable TWI0/I2C0 interrupt.
1: Enable TWI0/I2C0 interrupt.

Bit 5: EKBI, Enable Keypad Interrupt.
0: Disable the interrupt when KBCON.KBIF is set in Keypad control module.
1: Enable the interrupt when KBCON.KBIF is set in Keypad control module.

Bit 4: Reserved. Software must write “0” on this bit when EIE1 is written.

Bit 3: ESF, Enable System Flag interrupt.
0: Disable the interrupt when the group of {RTCF, BOF1, BOF0, WDTF} in PCON1, {STAF, STOF} in AUXR2, {BM1F, BM0F} in AUXR0, or TI0 with UTIE is set.
1: Enable the interrupt of the flags of {RTCF, BOF1, BOF0, WDTF} in PCON1, {STAF, STOF} in AUXR2, {BM1F, BM0F} in AUXR0, or TI0 with UTIE when the associated system flag interrupt is enabled in SFIE.

Bit 2: EPCA, Enable PCA0 interrupt.
0: Disable PCA0 interrupt.
1: Enable PCA0 interrupt.

Bit 1: EADC, Enable ADC Interrupt.
0: Disable the interrupt when ADCON0.ADCI is set in ADC module.
1: Enable the interrupt when ACCON0.ADCI is set in ADC module.

Bit 0: ESPI, Enable SPI Interrupt.
0: Disable the interrupt when SPSTAT.SPIF is set in SPI module.
1: Enable the interrupt when SPSTAT.SPIF is set in SPI module.

EIP1L: Extended Interrupt Priority 1 Low Register

SFR Page = 0~F
SFR Address = 0xAE RESET = 0000-0000

7	6	5	4	3	2	1	0
PAC0L	PTWI0L	PKBL	0	PSFL	PPCAL	PADCL	PSPIL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: PAC0L, AC0 interrupt priority-L register.
Bit 6: PTWI0L, TWI0/I2C0 interrupt priority-L register.
Bit 5: PKBL, keypad interrupt priority-L register.
Bit 4: Reserved. Software must write “0” on this bit when EIP1L is written.
Bit 3: PSFL, system flag interrupt priority-L register.
Bit 2: PPCAL, PCA0 interrupt priority-L register.
Bit 1: PADCL, ADC interrupt priority-L register.
Bit 0: PSPIL, SPI interrupt priority-L register.

EIP1H: Extended Interrupt Priority 1 High Register

SFR Page = 0~F

SFR Address = 0xAF RESET = 0000-0000

7	6	5	4	3	2	1	0
PAC0H	PTWI0H	PKBH	0	PSFH	PPCAH	PADCH	PSPIH
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: PAC0H, AC0 priority-H register.
Bit 6: PTWI0H, TWI0/I2C0 interrupt priority-H register.
Bit 5: PKBH, keypad interrupt priority-H register.
Bit 4: Reserved. Software must write “0” on this bit when EIP1H is written.
.
Bit 3: PSFH, system flag interrupt priority-H register.
Bit 2: PPCAH, PCA0 interrupt priority-H register.
Bit 1: PADCH, ADC interrupt priority-H register.
Bit 0: PSPIH, SPI interrupt priority-H register.

XICFG: External Interrupt Configured Register

SFR Page = **0 only**

SFR Address = 0xC1 RESET = 0000-0000

7	6	5	4	3	2	1	0
INT1IS.1	INT1IS.0	INT0IS.1	INT0IS.0	0	0	X1FLT	X0FLT
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: INT1IS.1~0, nINT1 input port pin selection bits which function is defined as following table.

INT1IS.1~0	Selected Port Pin of nINT1
0 0	P3.3
0 1	P3.1
1 0	P4.4
1 1	P1.0

Bit 5~4: INT0IS.1~0, nINT0 input port pin selection bits which function is defined as following table.

INT0IS.1~0	Selected Port Pin of nINT0
0 0	P4.5
0 1	P3.0
1 0	P3.2
1 1	P4.6

Bit 3~2: Reserved. Software must write “0” on this bit when XICFG is written.

Bit 1: X1FLT, nINT1 Filter mode control. It selects nINT1 input filter mode with X1FLT1 (XICFG1.1)

X1FLT1, X1FLT	nINT1 input filter mode
0 0	Disabled
0 1	SYSCLK x 3
1 0	SYSCLK/6 x 3
1 1	S0TOF x 3

Bit 0: X0FLT, nINT0 Filter mode control. It selects nINT0 input filter mode with X0FLT1 (XICFG1.0)

X0FLT1, X0FLT	nINT0 input filter mode
0 0	Disabled
0 1	SYSCLK x 3
1 0	SYSCLK/6 x 3
1 1	S0TOF x 3

XICFG1: External Interrupt Configured 1 Register

SFR Page = **1 only**

SFR Address = **0xC1** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	0	X1FLT1	X0FLT1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~2: Reserved. Software must write “0” on this bit when XICFG1 is written.

Bit 1: X1FLT1, nINT1 Filter mode control. It selects nINT1 input filter mode with X1FLT (XICFG.1). Refer XICFG description for nINT1 input filter mode definition.

Bit 0: X0FLT1, nINT0 Filter mode control. It selects nINT0 input filter mode with X0FLT (XICFG.0). Refer XICFG description for nINT0 input filter mode definition.

SFIE: System Flag Interrupt Enable Register

SFR Page = 0~F

SFR Address = 0x8E POR = 0110-0000

7	6	5	4	3	2	1	0
SIDFIE	1	1	RTCFIE	SPWIE	BOF1IE	BOF0IE	WDTFIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SIDFIE, Serial Interface (STWI/SI2C) Detection Flag Interrupt Enabled.
0: Disable SIDF(STAF or STOF) interrupt.
1: Enable SIDF(STAF or STOF) interrupt to share the system flag interrupt.

Bit 4: RTCFIE, Enable RTCF (PCON1.4) Interrupt.
0: Disable RTCF interrupt.
1: Enable RTCF interrupt.

Bit 3: SPWIE, Enable SPWF (PCON1.3) Interrupt.
0: Disable SPWF interrupt.
1: Enable SPWF interrupt.

Bit 2: BOF1IE, Enable BOF1 (PCON1.2) Interrupt.
0: Disable BOF1 interrupt.
1: Enable BOF1 interrupt.

Bit 1: BOF0IE, Enable BOF0 (PCON1.1) Interrupt.
0: Disable BOF0 interrupt.
1: Enable BOF0 interrupt.

Bit 0: WDTFIE, Enable WDTF (PCON1.0) Interrupt.
0: Disable WDTF interrupt.
1: Enable WDTF interrupt.

PCON1: Power Control Register 1

SFR Page = 0~F & P

SFR Address = 0x97 POR = 0000-0000

7	6	5	4	3	2	1	0
SWRF	EXRF	0	RTCF	SPWF	BOF1	BOF0	WDTF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SWRF, Software Reset Flag.
0: This bit must be cleared by software writing **“1”** to it.
1: This bit is set by hardware if a Software Reset occurs.

Bit 6: EXRF, External Reset Flag.
0: This bit must be cleared by software writing **“1”** to it.
1: This bit is set by hardware if an External Reset occurs.

Bit 4: RTCF, RTC overflow flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing "0" is no operation.
1: This bit is only set by hardware when RTCCT overflows. Writing **"1"** on this bit will clear RTCF.

Bit 3: SPWF, SP Warning Flag.
0: This bit must be cleared by software writing **"1"** on it. Software writing "0" is no operation.
1: This bit is only set by hardware when SP ≥ SPHB. Writing **"1"** on this bit will clear SPWF when SP < SPHB.

Bit 2: BOF1, Brown-Out Detection flag 1.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if the operating voltage matches the detection level of Brown-Out Detector 1 (4.2V/3.6V/2.7V/2.4V).

Bit 1: BOF0, Brown-Out Detection flag 0.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if the operating voltage matches the detection level of Brown-Out Detector 0 (2.35V).

Bit 0: WDTF, WDT overflow flag.
0: This bit must be cleared by software writing **"1"** to it.
1: This bit is set by hardware if a WDT overflow occurs.

AUXR2: Auxiliary Register 2

SFR Page = 0 only
SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: STAF, Start Flag detection of STWI (SID).
0: Clear by firmware by writing "0" on it. STAF might be held within MCU reset period, so needs to clear STAF in firmware initial.
1: Set by hardware to indicate the START condition occurred on STWI bus.

Bit 6: STOF, Stop Flag detection of STWI (SID).
0: Clear by firmware by writing "0" on it.
1: Set by hardware to indicate the STOP condition occurred on STWI bus. STOF might be held within MCU reset period, so needs to clear STOF in firmware initial.

15. Timers/Counters

MG82F6B08 / 6B001/ 6B104 has **three** 16-bit Timers/Counters: Timer 0, Timer 1, Timer 2. All of them can be configured as timers or event counters.

In the “timer” function, the timer rate is prescaled by 12 clock cycle to increase register value. In other words, it is to count the standard C51 machine cycle. AUXR2.T0X12, AUXR2.T1X12, T2MOD.T2X12 are the function for Timer 0/1/2 to set the timer rate on every clock cycle. It performs at a speed 12 times than standard C51 timer function. Other prescaler values can be selected by combining T0C/T, T0XL and T0X12 for Timer 0 clock input.

In the “counter” function, the register is increased in response to a 1-to-0 transition at its corresponding external input pin, T0, T1, T2. In this function, the external input is sampled by every timer rate cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register at the end of the cycle following the one in which the transition was detected.

15.1. Timer 0 and Timer 1

15.1.1. Timer 0/1 Mode 0

The timer register is configured as a PWM generator. As the count rolls over from all 1s to all 0s, it sets the timer interrupt flag TFx. Timer0 uses the control bits {T0XL, T0X12, T0C/T} to set the clock source to count. And it also uses TR0 and {T0G1, T0GATE} to select the gating sources to block the trigger signal to stop the counting. Timer1 uses the control bits {T1X12, T1C/T} to set the clock source to count. And it uses TR1 and {T1G1, T1GATE} to select the gating sources to block the trigger signal to stop the counting. Mode 0 operation is the same for Timer0 and Timer1. The PWM function of Timer 0/1 is shown in Figure 15–1 and Figure 15–2.

Figure 15–1. Timer 0 Mode 0 Structure

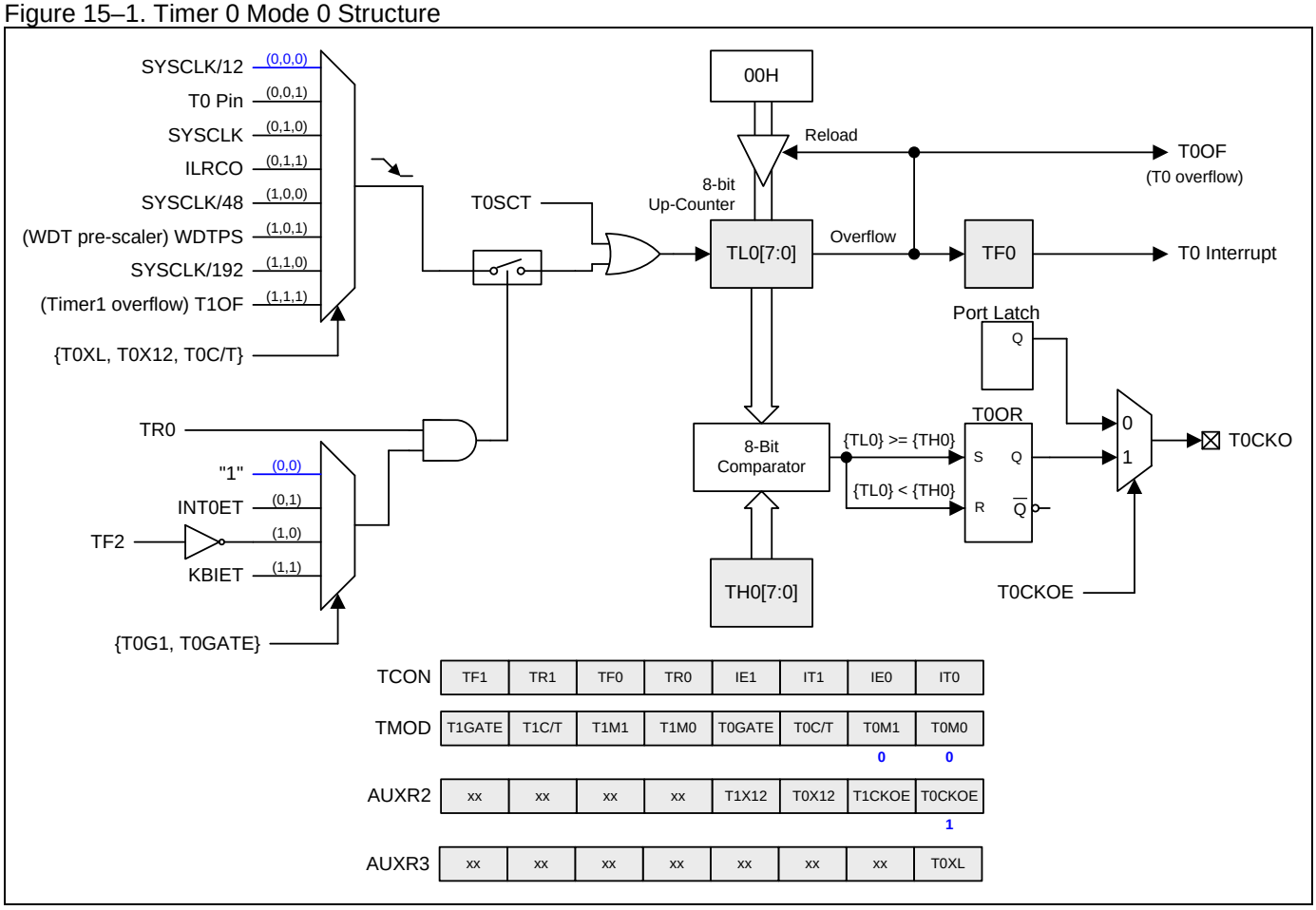
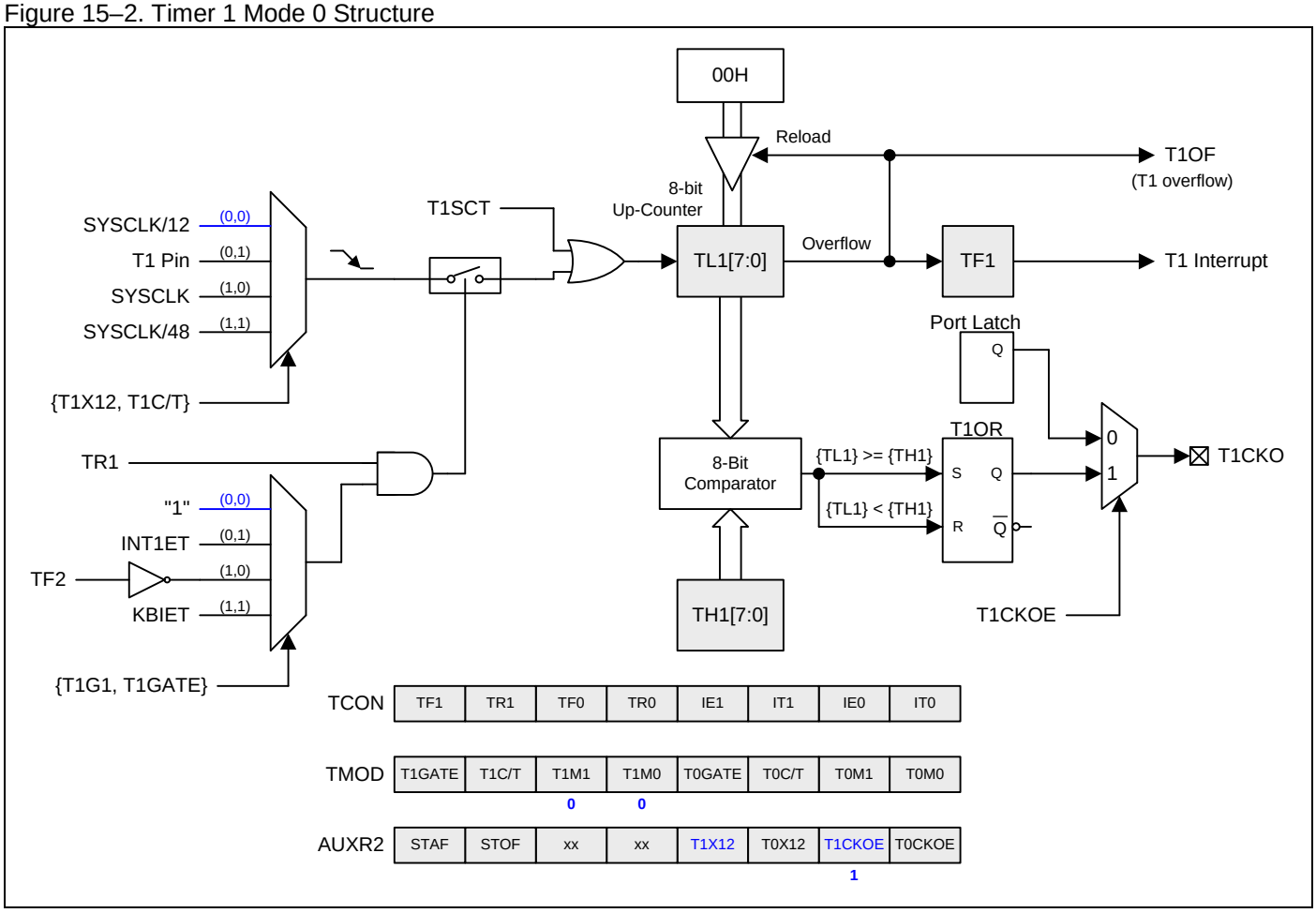


Figure 15–2. Timer 1 Mode 0 Structure



15.1.2. Timer 0/1 Mode 1

Timer 0/1 in Mode1 is configured as a 16 bit timer or counter. The function of GATE, TxG1 and TRx is same as mode 0. Figure 15–3 and Figure 15–4 show the mode 1 structure of Timer 0 and Timer 1.

Figure 15–3. Timer 0 Mode 1 Structure

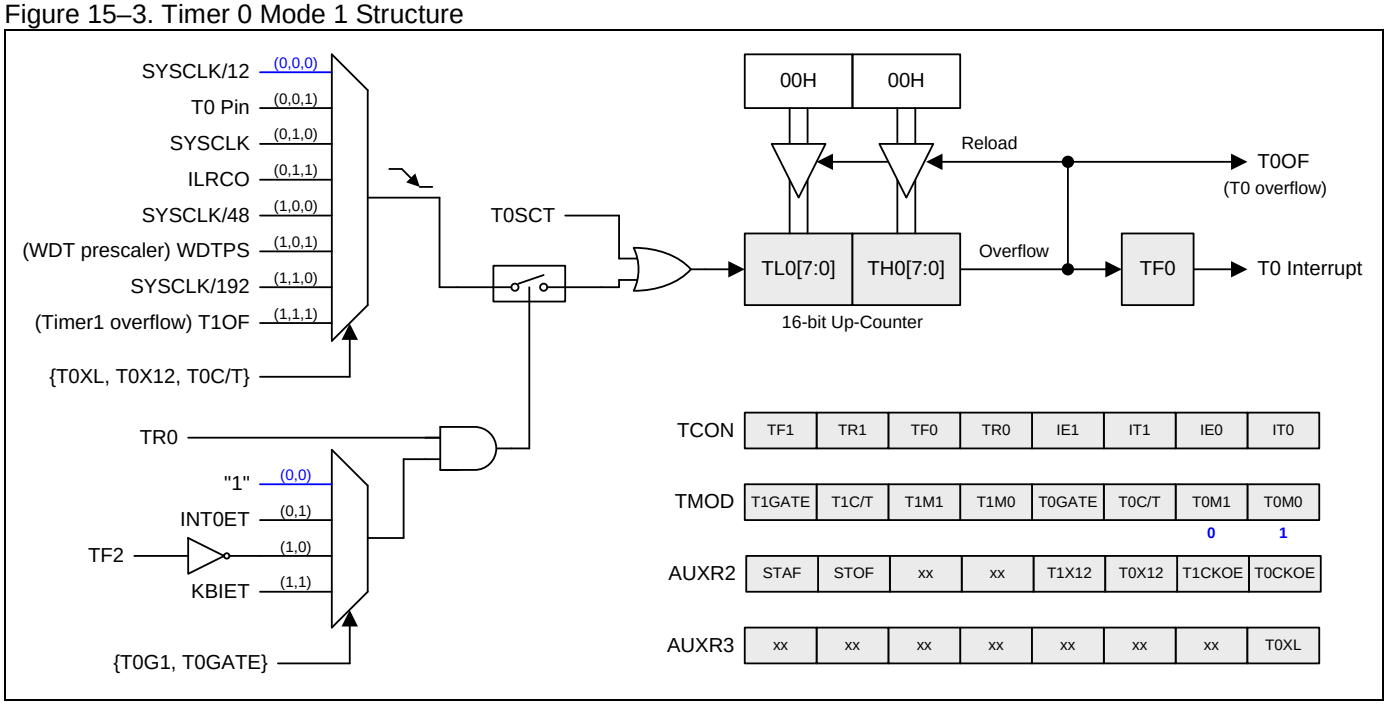
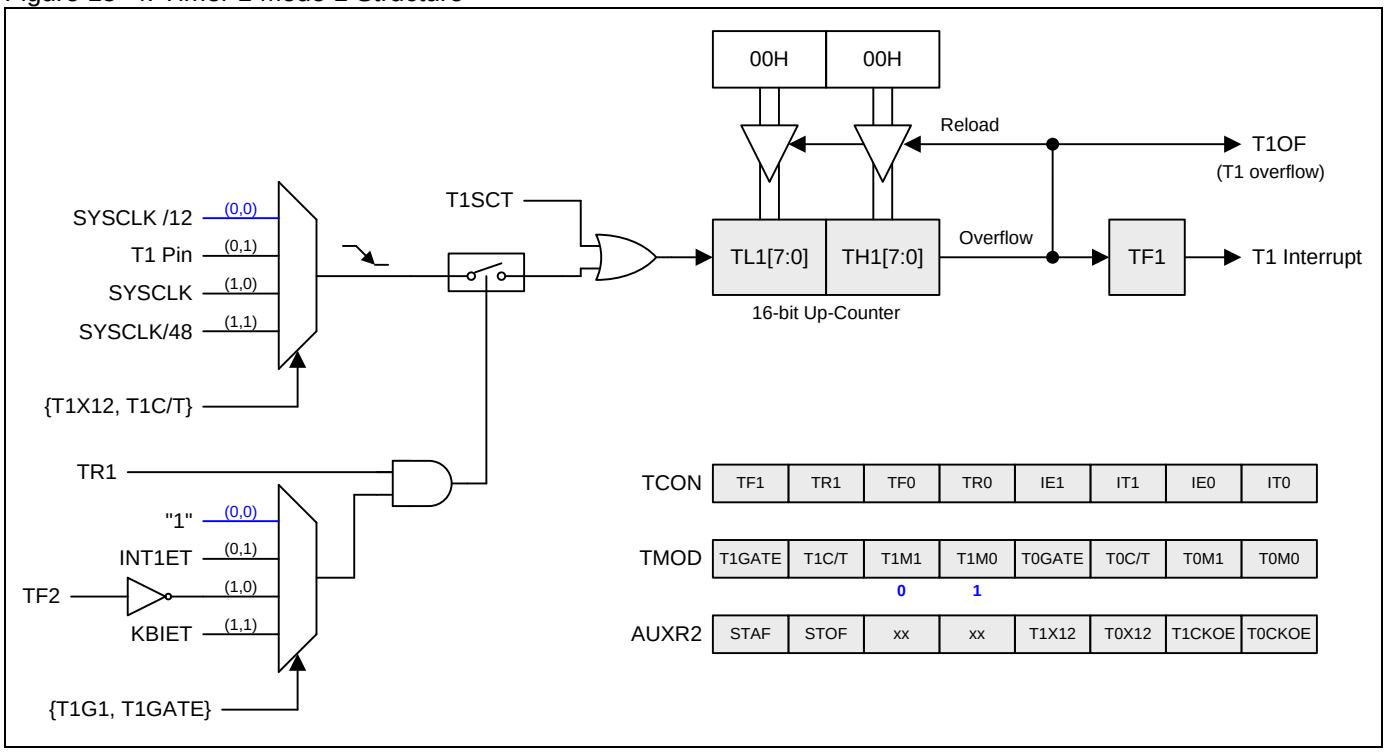


Figure 15–4. Timer 1 Mode 1 Structure



15.1.3. Timer 0/1 Mode 2

Mode 2 configures the timer register as an 8-bit counter (TLx) with automatic reload. Overflow from TLx not only set TFx, but also reload TLx with the content of THx, which is determined by software. The reload leaves THx unchanged. Mode 2 operation is the same for Timer0 and Timer1. Figure 15–5 and Figure 15–6 show the mode 2 structure of Timer 0 and Timer 1.

Figure 15–5. Timer 0 Mode 2 Structure

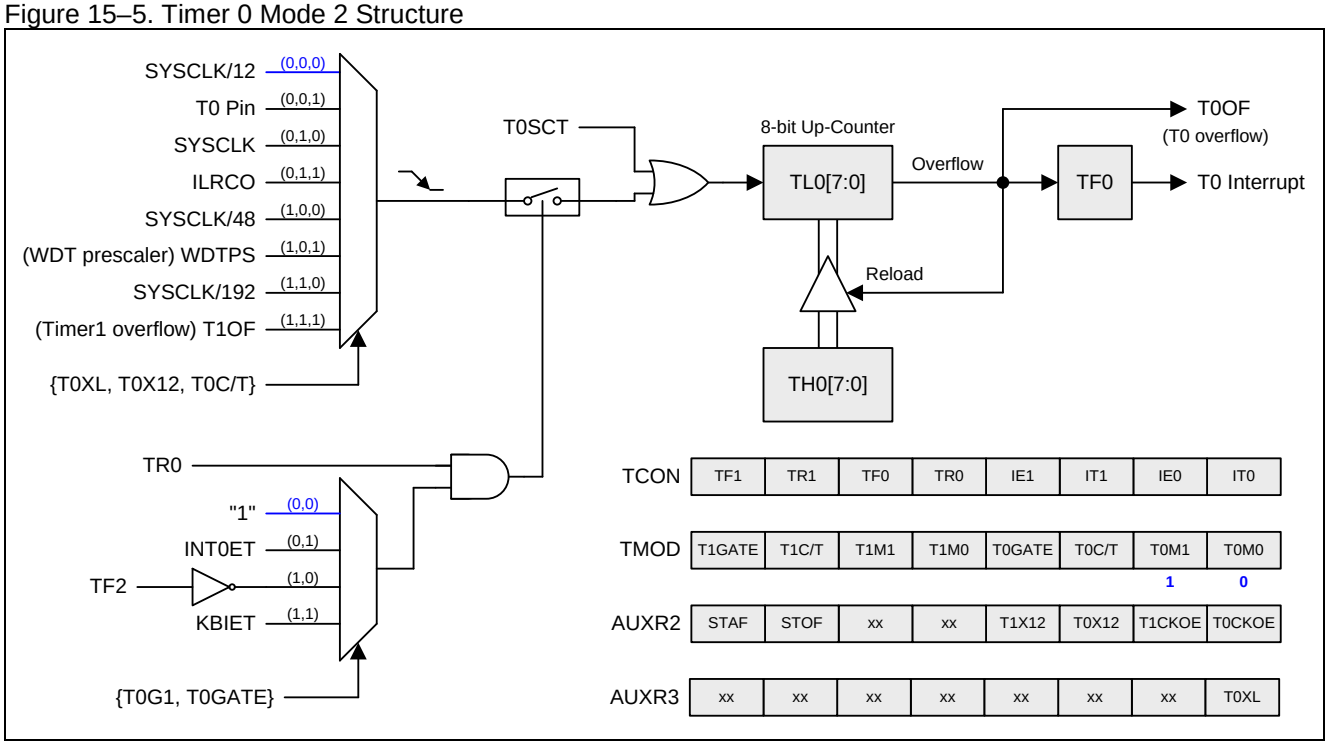
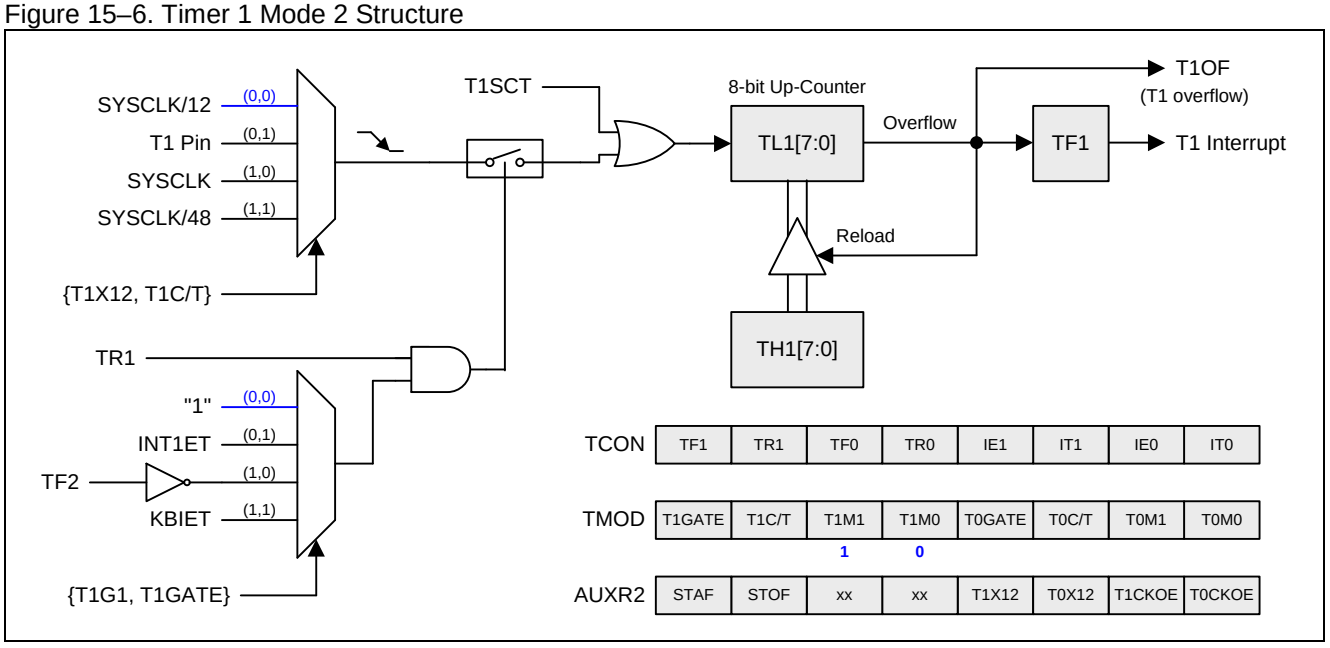


Figure 15–6. Timer 1 Mode 2 Structure



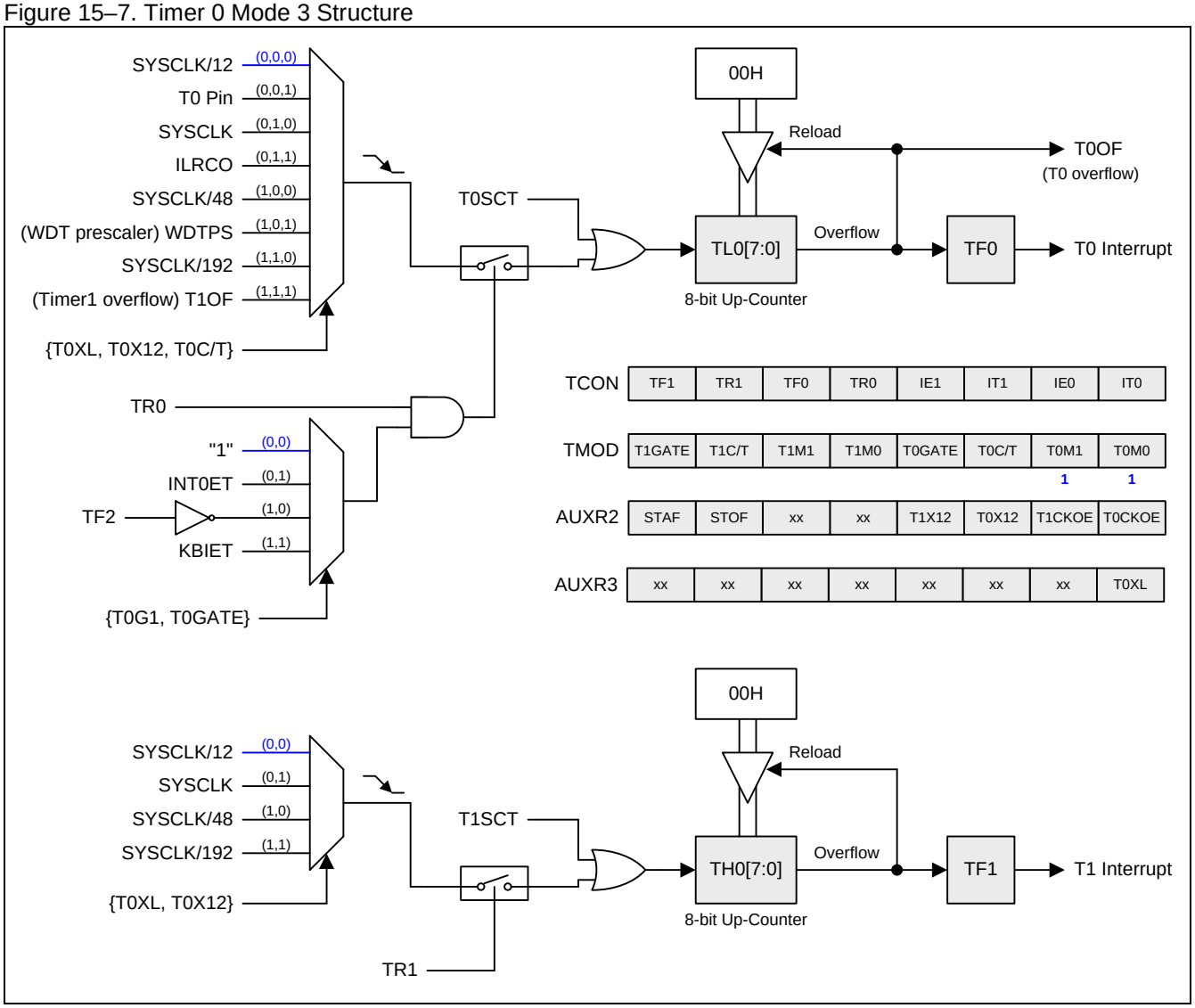
15.1.4. Timer 0/1 Mode 3

Timer1 in Mode3 will be held even if TR1 = 1.
Timer0 in Mode 3 set TL0 and TH0 as two separate 8-bit counters. TL0 is controlled by the Timer0 control bits such like T0XL, T0X12, T0C/T, T0G1, T0GATE, TR0 and TF0. TH0 is locked into a timer function (cannot be external event counter) and TR1 from Timer1 will be taken over to control the TH0. TH0 overflow will set the TF1 to trigger Timer1 interrupt.

In order to ensure TL1 does not malfunction, please set the Timer 1 to mode 3 before Timer 0 to set to mode 3.

Figure 15–7 shows the mode 3 structure of Timer 0.

Figure 15–7. Timer 0 Mode 3 Structure



15.1.5. Timer 0/1 Programmable Clock-Out

Timer 0 and Timer 1 have a Clock-Out Mode (while TxCKOE=1). In this mode, Timer 0 or Timer 1 operates as 8-bit auto-reload timer for a programmable clock generator with 50% duty-cycle. The generated clocks come out on T0CKO (P3.4) and T1CKO (P3.5) individually. The input clock of Timer 0 increases the 8-bit timer, TL0, in Timer 0 module. The input clock of Timer 1 increases the 8-bit timer, TL1, in Timer 1 module. The timer repeatedly counts to overflow from a loaded value. Once overflows occur, the contents of (TH0 and TH1) are loaded into (TL0, TL1) for the consecutive counting. Figure 15–8 and Figure 15–9 formula gives the formula of Timer 0 and Timer 1 clock-out frequency. Figure 15–10 and Figure 15–11 show the clock-out structure and output control.

Figure 15–11 shows the clock-out structure of Timer 1.

Figure 15–8. Timer 0 clock out equation

$$\text{T0 Clock-out Frequency} = \frac{\text{T0 Clock Frequency}}{2 \times (256 - \text{TH0})}$$

Figure 15–9. Timer 1 clock out equation

$$\text{T1 Clock-out Frequency} = \frac{\text{T1 Clock Frequency}}{2 \times (256 - \text{TH1})}$$

Note:
(1) Timer 0/1 overflow flag, TF0/1, will be set when Timer 0/1 overflows
(2) For SYSCLK=12MHz and select SYSCLK/12 as Timer 0/1 clock source, Timer 0/1 has a programmable output frequency range from 1.95KHz to 500KHz.
(3) For SYSCLK=12MHz and select SYSCLK as Timer 0/1 clock source , Timer 0/1 has a programmable output frequency range from 23.44KHz to 6MHz.

Figure 15–10. Timer 0 in Clock Output Mode

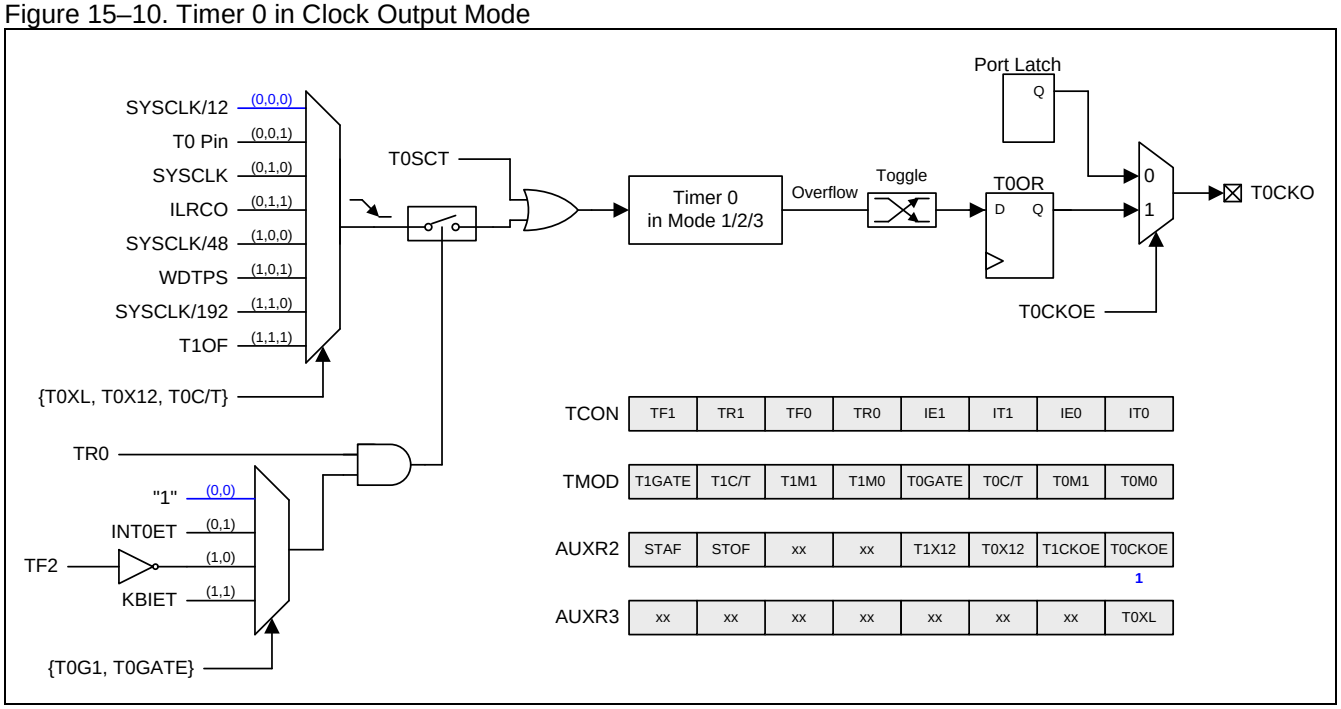
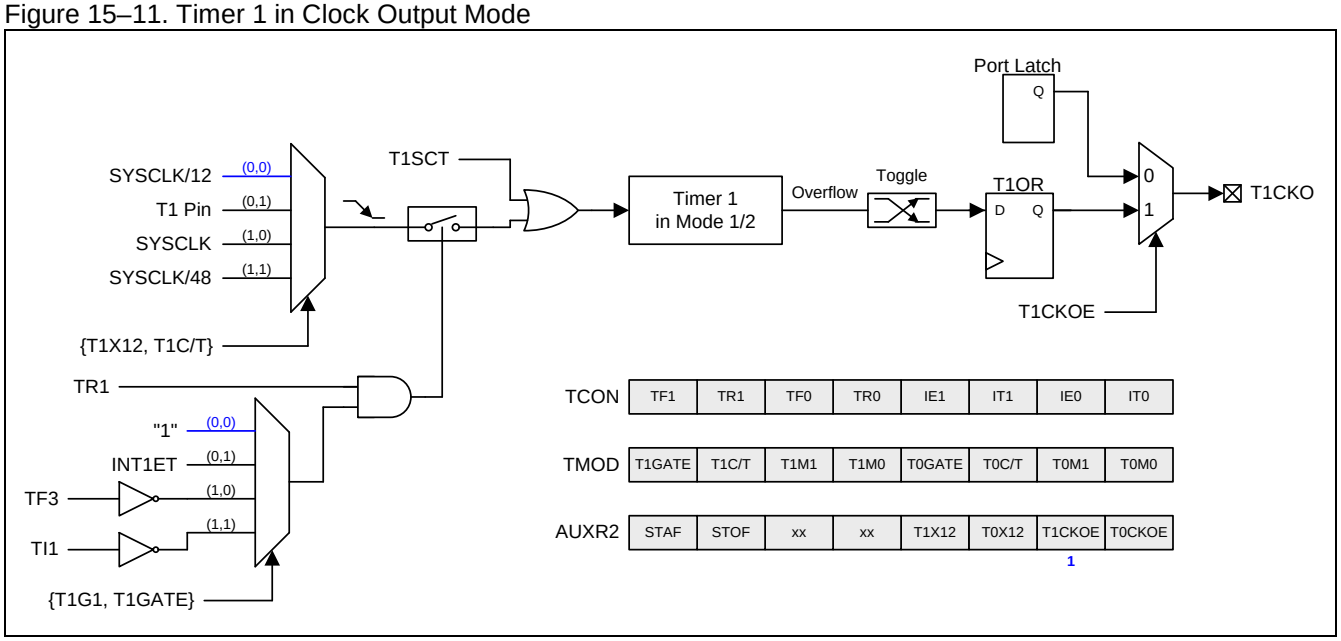


Figure 15–11. Timer 1 in Clock Output Mode



How to Program Timer 0/1 in Clock-out Mode

- Select Timer 0/1 clock source.
- Determine the 8-bit reload value from the formula and enter it in the TH0/TH1 register.
- Enter the same reload value as the initial value in the TL0/TL1 register.
- Set T0CKOE/T1CKOE bit in AUXR2 register.
- Set TR0/TR1 bit in TCON register to start the Timer 0/1.

In the Clock-Out mode, Timer 0/1 rollovers will not generate an interrupt. This is similar to when Timer 1 is used as a baud-rate generator. It is possible to use Timer 1 as a baud rate generator and a clock generator simultaneously. Note, however, that the baud-rate and the clock-out frequency depend on the same overflow rate of Timer 1. So, software usually disables the Timer 0/1 interrupt in this kind of application.

15.1.6. Timer 0/1 Register

TCON: Timer/Counter Control Register

SFR Page = 0~F
SFR Address = 0x88 RESET = 0000-0000

7	6	5	4	3	2	1	0
TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: TF1, Timer 1 overflow flag.
0: Cleared by hardware when the processor vectors to the interrupt routine, or cleared by software.
1: Set by hardware on Timer/Counter 1 overflow, or set by software.

Bit 6: TR1, Timer 1 Run control bit.
0: Disabled to stop Timer/Counter 1.
1: Enabled to start Timer/Counter 1.

Bit 5: TF0, Timer 0 overflow flag.
0: Cleared by hardware when the processor vectors to the interrupt routine, or cleared by software.
1: Set by hardware on Timer/Counter 0 overflow, or set by software.

Bit 4: TR0, Timer 0 Run control bit.
0: Disabled to stop Timer/Counter 0.
1: Enabled to start Timer/Counter 0.

TMOD: Timer/Counter Mode Control Register

SFR Page = 0~F
SFR Address = 0x89 RESET = 0000-0000

7	6	5	4	3	2	1	0
T1GATE	T1C/T	T1M1	T1M0	T0GATE	T0C/T	T0M1	T0M0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**
|←---------- Timer1 ----------→|				|←---------- Timer0 ----------→|		

Bit 7: T1GATE, Gating control for Timer1.

T1G1, T1GATE	T1 Gate source
0 0	Disable
0 1	nINT1 active
1 0	TF2 active
1 1	KBI active

Bit 6: T1C/T, Timer 1 clock source selector. It controls the Timer 1 as timer or counter with 4 clock sources. Refer to T1X12 description in the AUXR2.

Bit 5~4: Operating mode selection.

T1M1	T1M0	Timer 1 Operating Mode
0	0	8-bit PWM generator for Timer1
0	1	16-bit timer/counter for Timer1
1	0	8-bit timer/counter with automatic reload for Timer1
1	1	Timer/Counter1 Stopped

Bit 3: T0GATE, Gating control for Timer0.

T0G1, T0GATE	T0 Gate source
0 0	Disable
0 1	nINT0 active
1 0	TF2 active
1 1	KBI active

.
Bit 2: T0C/T, Timer 0 clock source selector. It controls the Timer 0 as timer or counter with 8 clock sources. Refer to T0X12 description in the AUXR2.

Bit 1~0: Operating mode selection.

T0M1	T0M0	Timer 0 Operating Mode
0	0	8-bit PWM generator for Timer0
0	1	16-bit timer/counter for Timer0
1	0	8-bit timer/counter with automatic reload for Timer0
1	1	TL0 is 8-bit timer/counter, TH0 is locked into 8-bit timer

TL0: Timer 0 Low byte Register

SFR Page = 0~F

SFR Address = 0x8A RESET = 0000-0000

7	6	5	4	3	2	1	0
TL0.7	TL0.6	TL0.5	TL0.4	TL0.3	TL0.2	TL0.1	TL0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TH0: Timer 0 High byte Register

SFR Page = 0~F

SFR Address = 0x8C RESET = 0000-0000

7	6	5	4	3	2	1	0
TH0.7	TH0.6	TH0.5	TH0.4	TH0.3	TH0.2	TH0.1	TH0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TL1: Timer 1 Low byte Register

SFR Page = 0~F

SFR Address = 0x8B RESET = 0000-0000

7	6	5	4	3	2	1	0
TL1.7	TL1.6	TL1.5	TL1.4	TL1.3	TL1.2	TL1.1	TL1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

TH1: Timer 1 High byte Register

SFR Page = 0~F

SFR Address = 0x8D RESET = 0000-0000

7	6	5	4	3	2	1	0
TH1.7	TH1.6	TH1.5	TH1.4	TH1.3	TH1.2	TH1.1	TH1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

AUXR2: Auxiliary Register 2

SFR Page = 0~F

SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	W	W	R/W	R/W	R/W	R/W

Bit 3: T1X12, Timer 1 clock source selection with T1C/T control.

T1X12, T1C/T	Timer 1 Clock Selection
0 0	SYSCLK/12
0 1	T1 Pin
1 0	SYSCLK
1 1	SYSCLK/48

Bit 2: T0X12, Timer 0 clock source selection with T0C/T and T0XL control.

T0XL, T0X12, T0C/T	Timer 0 Clock Selection
0 0 0	SYSCLK/12
0 0 1	T0 Pin
0 1 0	SYSCLK
0 1 1	ILRCO
1 0 0	SYSCLK/48
1 0 1	WDTPS
1 1 0	SYSCLK/192
1 1 1	T1OF

Bit 1: T1CKOE, Timer 1 Clock Output Enable.
0: Disable Timer 1 clock output.
1: Enable Timer 1 clock output on T1CKO Port pin.

Bit 0: T0CKOE, Timer 0 Clock Output Enable.
0: Disable Timer 0 clock output.
1: Enable Timer 0 clock output on T0CKO Port pin.

AUXR3: Auxiliary Register 3

SFR Page = **0 only**
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: Reserved. Software must write "**0**" on these bits when AUXR3 is written.

Bit 6: T0PS0, Timer 0 Port pin Selection.

T0PS0	T0/T0CKO
0	P4.6
1	P4.4

Bit 0: T0XL is the Timer 0 per-scaler control bit. Please refer T0X12 (AUXR2.2) for T0XL function definition.

AUXR4: Auxiliary Register 4

SFR Page = **1 only**
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
T2PS1	T2PS0	0	T1PS0	0	0	AC0OE	AC0FLT1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 5: Reserved. Software must write "**0**" on these bits when AUXR4 is written.

Bit 4: T1PS1~0, Timer 1 Port pin Selection.

T1PS0	T1/T1CKO
0	P3.3
1	P4.5

AUXR9: Auxiliary Register 9

SFR Page = 6 only
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	SIDPS0	T1G1	T0G1	C0FDC1	C0FDC0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 5: T1G1, Gating source selection of Timer 1.

T1G1, T1GATE	T1 Gate source
00	Disable
01	INT1 active
10	TF2 active
11	KBI active

Bit 4: T0G1, Gating source selection of Timer 0.

T0G1, T0GATE	T0 Gate source
00	Disable
01	INT0 active
10	TF2 active
11	KBI active

15.2. Timer 2

Timer 2 is a 16-bit Timer/Counter which can operate either as a timer or an event counter, as selected by T2CKS, T2X12 and C/T2. Timer 2 has serval operating modes: Capture, Auto-Reload (up counting), 8-bit PWM, Baud Rate Generator and Programmable Clock-Out, which are selected by bits in the T2CON, T2MOD and T2MOD1 registers.

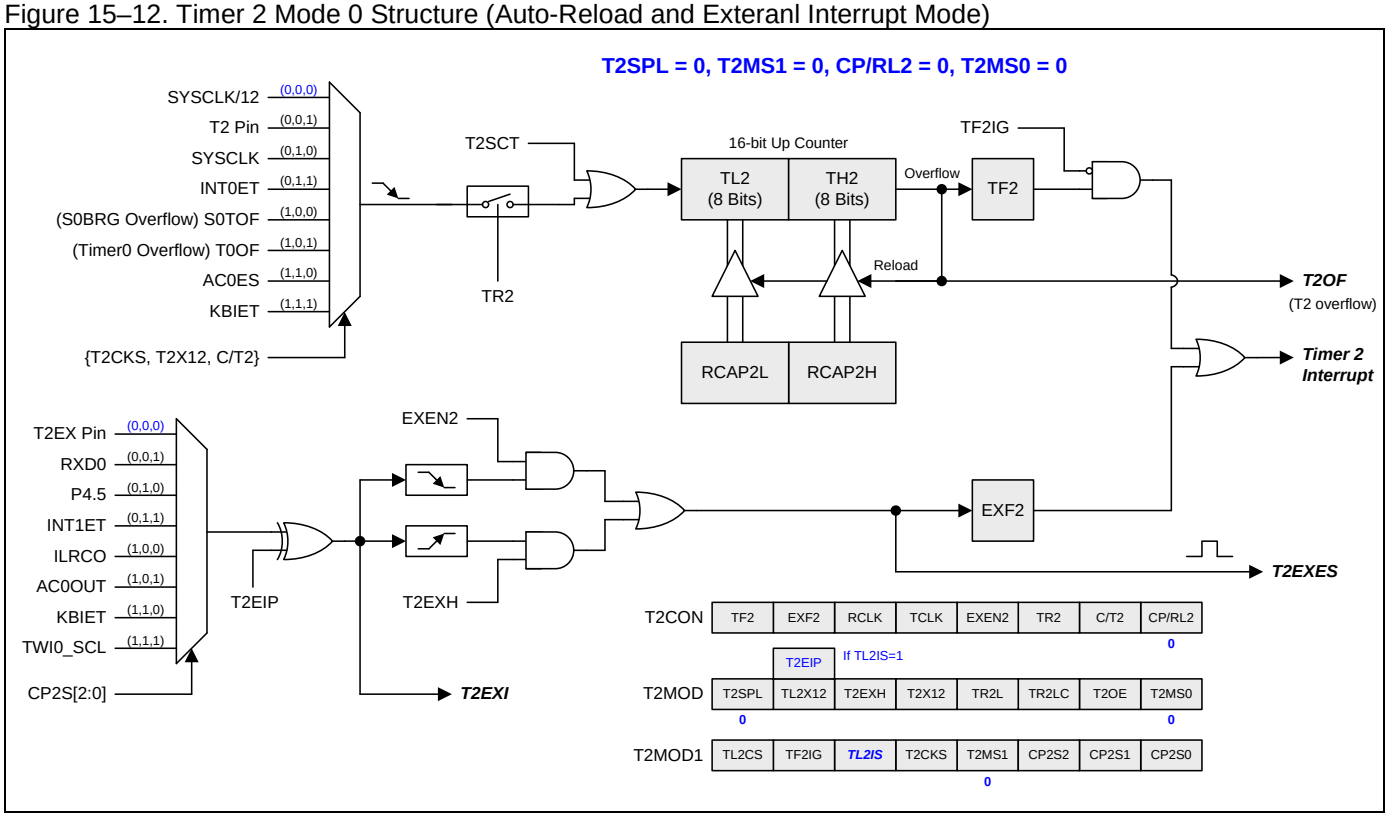
15.2.1. Timer 2 Mode 0 (Auto-Reload and External Interrupt)

In this mode, Timer 2 provides a 16-bit auto-reload timer/counter. The TF2, Timer 2 overflow flag, is one of the Timer 2 interrupt source which interrupt function can be blocked by TF2IG. EXEN2 enables a 1-to-0 transition at T2EXI to set the flag, EXF2, for an external input interrupt to share the Timer 2 interrupt with TF2. T2EXI is the selection result of 8 Timer 2 external inputs. T2EXH performs the same function as EXEN2 but it enables the detecting a 0-to-1 transition at T2EXI input.

The Timer 2 overflow event (T2OF) in this module will be output to other peripheral as clock input or event source.

Timer 2 Mode 0 is illustrated in Figure 15–12.

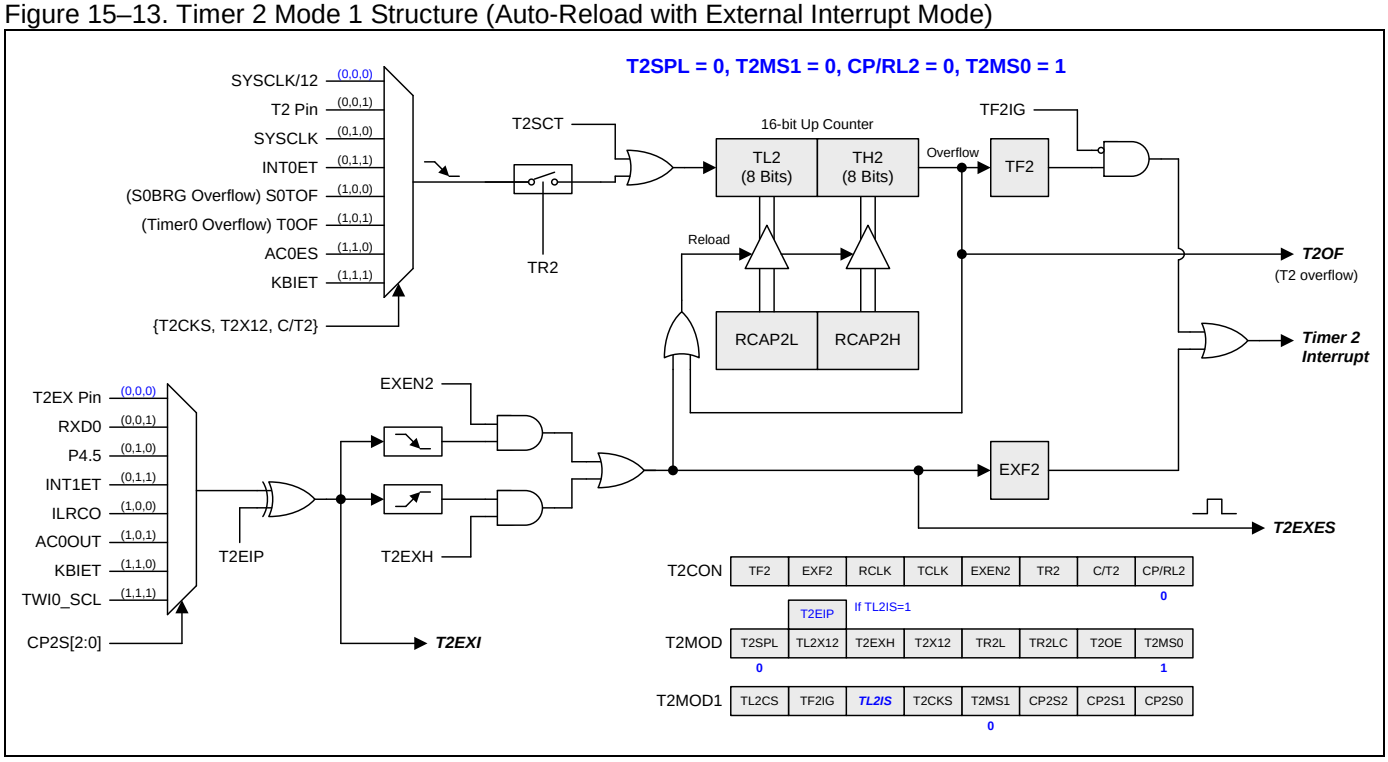
Figure 15–12. Timer 2 Mode 0 Structure (Auto-Reload and Exteranl Interrupt Mode)



15.2.2. Timer 2 Mode 1 (Auto-Reload with External Interrupt)

Figure 15–13 shows Timer 2 Mode 1, which enables Timer 2 to count up automatically. In this mode there are two options selected by bit EXEN2 in T2CON register. If EXEN2=0, then Timer 2 counts up to 0FFFFH and sets the TF2 (Overflow Flag) bit upon overflow. This causes the Timer 2 registers to be reloaded with the 16-bit value in RCAP2L and RCAP2H. The values in RCAP2L and RCAP2H are preset by firmware. If EXEN2=1, then a 16-bit reload can be triggered either by a T2 overflow or by a 1-to-0 transition of T2EXI, which is chosen from one of 8 external trigger inputs. This transition also sets the EXF2 bit. The Timer 2 interrupt, if enabled, can be generated when either TF2 or EXF2 are 1. T2EXH performs the same function as EXEN2 but it enables the detecting a 0-to-1 transition at input T2EXI.

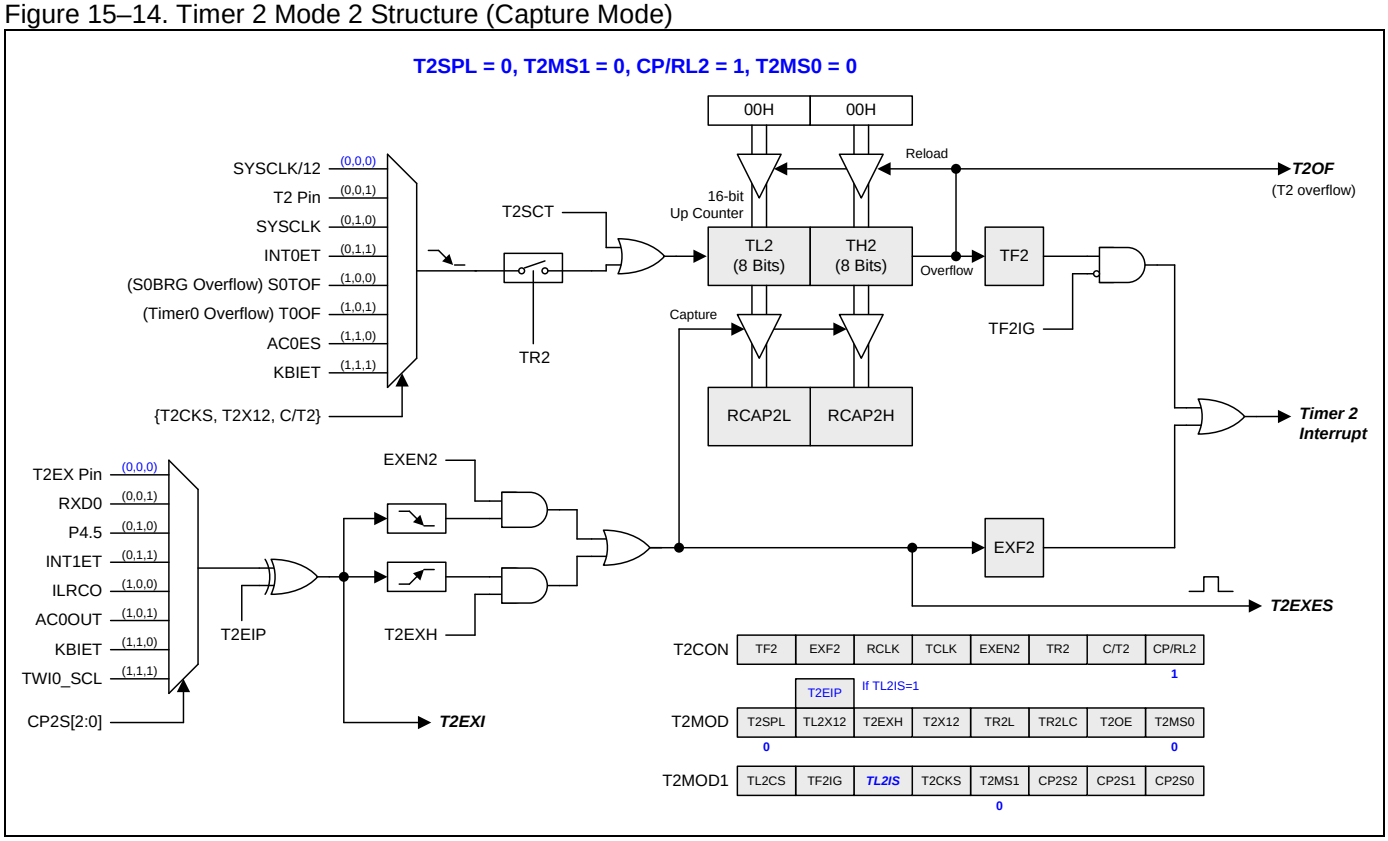
Figure 15–13. Timer 2 Mode 1 Structure (Auto-Reload with External Interrupt Mode)



15.2.3. Timer 2 Mode 2 (Capture)

Figure 15–14 shows the capture mode there are two options selected by bit EXEN2 in T2CON. If EXEN2=0, Timer 2 is a 16-bit timer or counter which, upon overflow, sets bit TF2 (Timer 2 overflow flag). This bit can then be used to generate an interrupt (by enabling the Timer 2 interrupt bit in the IE register). If EXEN2=1, Timer 2 still does the above, but with the added feature that a 1-to-0 transition at T2EXI, one of 8 Timer 2 external inputs, that causes the current value in the Timer 2 registers, TH2 and TL2, to be captured into registers RCAP2H and RCAP2L, respectively. In addition, the transition at T2EXI causes bit EXF2 in T2CON to be set, and the EXF2 bit (like TF2) can generate an interrupt which vectors to the same location as Timer 2 overflow interrupt. T2EXH performs the same function as EXEN2 but it enables the detecting a 0-to-1 transition at T2EXI input.

Figure 15–14. Timer 2 Mode 2 Structure (Capture Mode)

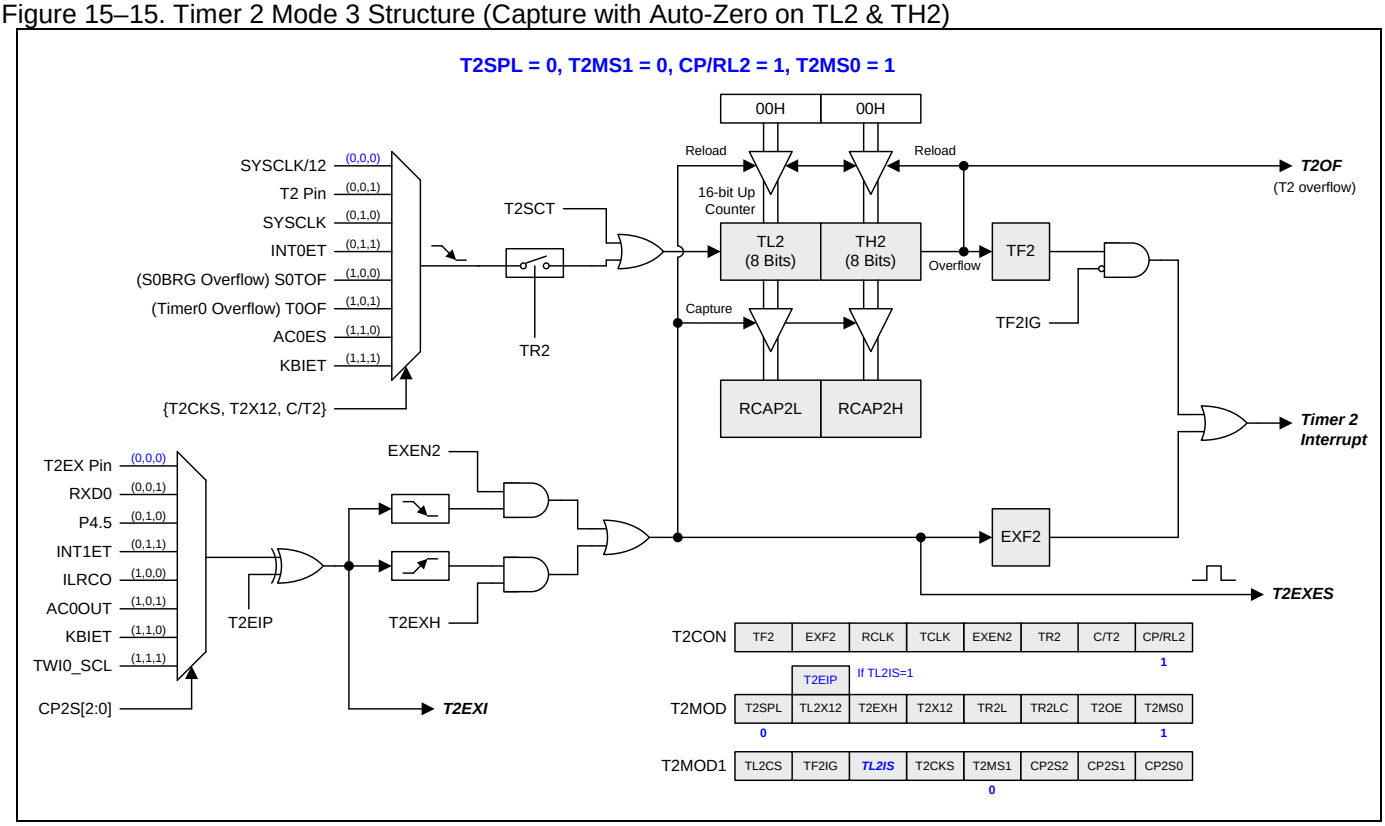


15.2.4. Timer 2 Mode 3 (Capture with Auto-Zero)

Timer 2 Mode 3 is the similar function with Timer 2 Mode 2. There is one difference that the T2EXES, EXF2 event set signal, not only is the capture source of Timer 2 but also clears the content of TL2 and TH2 to 0x0000H.

Timer 2 Mode 3 is illustrated in Figure 15–15.

Figure 15–15. Timer 2 Mode 3 Structure (Capture with Auto-Zero on TL2 & TH2)



15.2.5. Timer 2 Mode 6 (Duty Capture)

Timer 2 Mode 6 provides ability to capture cycle time or duty of the input waveform. Three edges of the signal can calculate the cycle and duty. In the timer duty capture mode it needs to clear the TH2:TL2 to 00H. And then start duty capture mode by setting TR2, but the counter will not be started yet, it will wait until the first edge send into the external trigger channel, for example T2EX Pin. It means the first edge value is 00H. And after first edge triggered, the counter start to count. Please note, the first trigger edge of the T2EXI must be the rising edge. And even you set the T2EXH, the first edge will be ignored to trigger EXF2.

Second, if only want to calculate the pulse width, then can set EXEN2 to pass the second edge to trigger EXF2. In this case, when second edge trigger timer to capture its value into RCAP2H: RCAP2L, it also trigger EXF2 for interrupt to know it has been done.

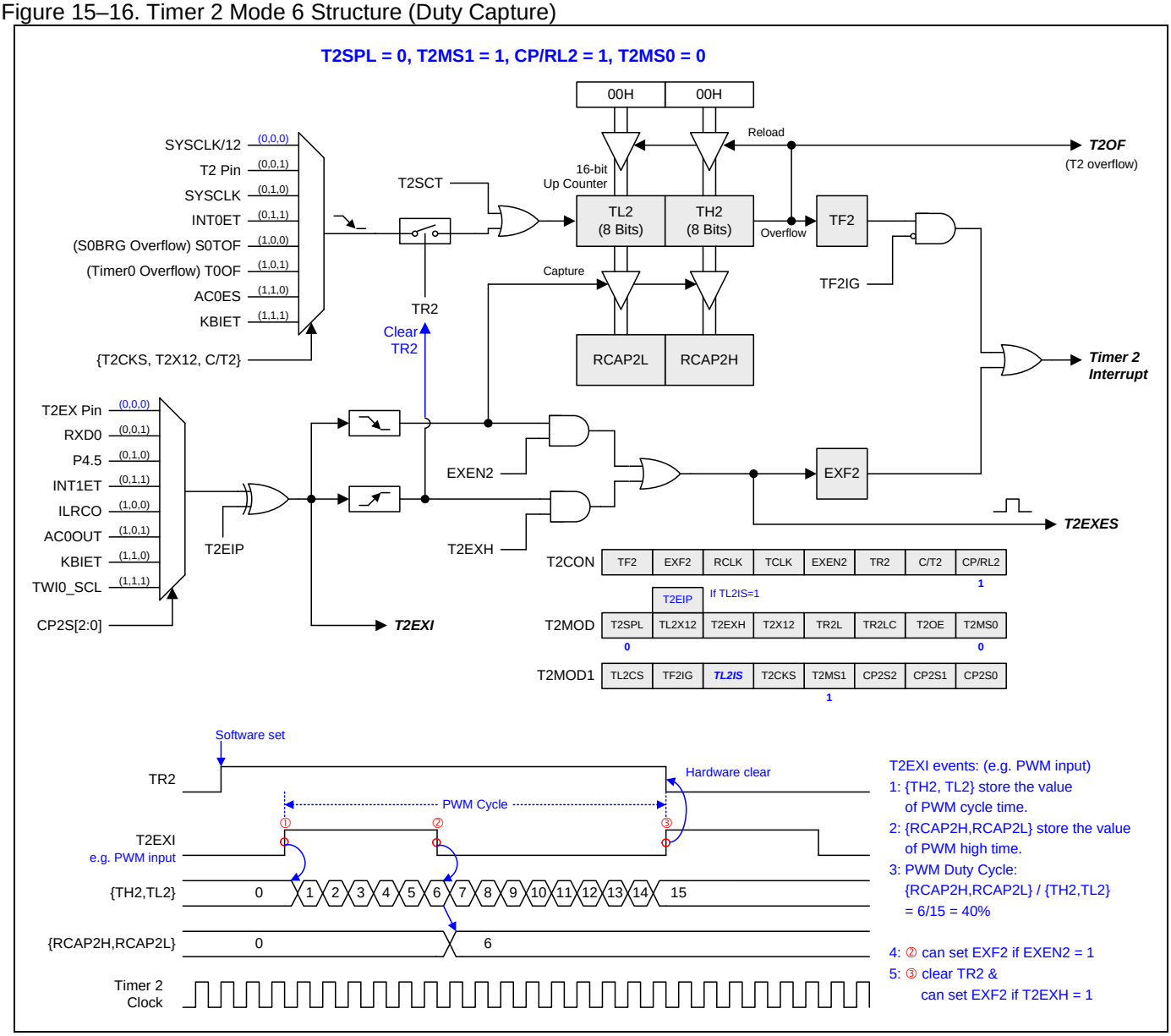
But if you want to get cycle time, then the second edge needs to be blocked by clear EXEN2.

When third edge arrives, it will automatically clear TR2 to stop the counter.

Using the TH2: TL2 (3rd edge), RCAP2H: RCAP2L (2nd edge) and 0 (1set edge) to calculate the cycle time.

Timer 2 Mode 6 is illustrated in Figure 15–16.

Figure 15–16. Timer 2 Mode 6 Structure (Duty Capture)



15.2.6. Split Timer 2 Mode 0 (AR and Ext. INT)

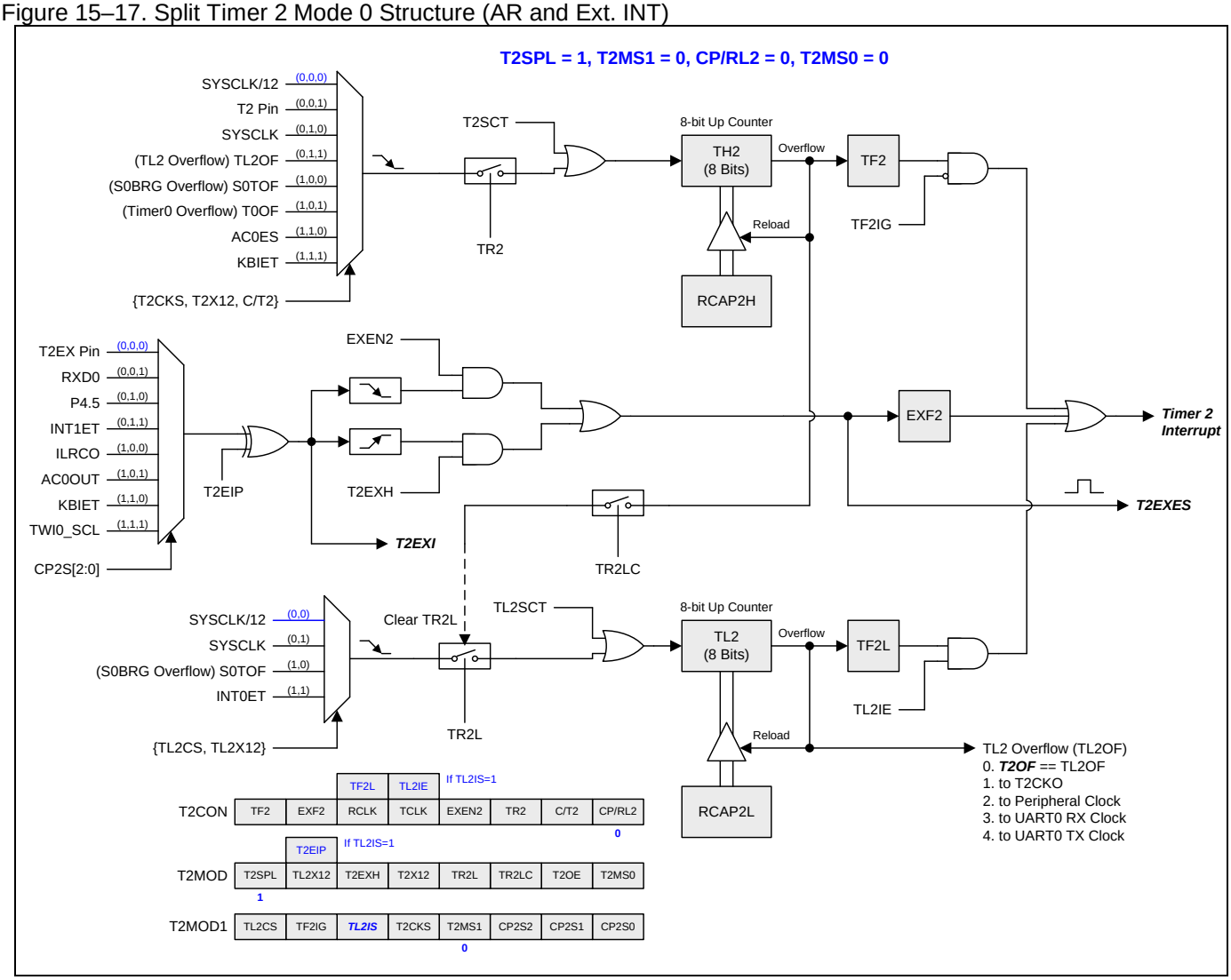
When T2SPL is set in this mode, Timer 2 operates as two 8-bit timers (TH2 and TL2). Both 8-bit timers operate in up-counter as shown in Figure 15–17. TH2 holds the reload value for RCAP2H and keep the same 8 clock source inputs selection as 16-bit mode. It behaves the 8-bit function liked Timer 2 Mode 0 in 16-bit mode. TL2 holds the reload value for RCAP2L with 4 clock inputs selection. The TR2 bit in T2CON handles the run control for TH2. The TR2L bit in T2MOD handles the run control for TL2. And TH2 overflow can stop the TR2L running when TR2LC is set.

There are 3 interrupt flags in split mode, EXF2, TF2 and TF2L. EXF2 has the same function as 16-bit mode to detect the transition on T2EXI. TF2 is set when TH2 overflows from 0xFF to 0x00 with TF2IG control. TF2L is set when TL2 overflows from 0xFF to 0x00 with interrupt enabled by TL2IE. The EXF2, TF2 and TF2L interrupt flags are not cleared by hardware and must be cleared by software.

By the way, the Timer 2 overflow event (T2OF) in 16-bit timer is replaced by TL2 overflow event (TL2OF) in this split mode.

If TL2IS in T2MOD1 is “0”, the bits on T2CON.5~4 and T2MOD.6 are the function of RCLK, TCLK and TL2X12. If TL2IS is “1”, the bits on T2CON.5~4 and T2MOD.6 are the function of TF2L, TL2IE and T2EIP.

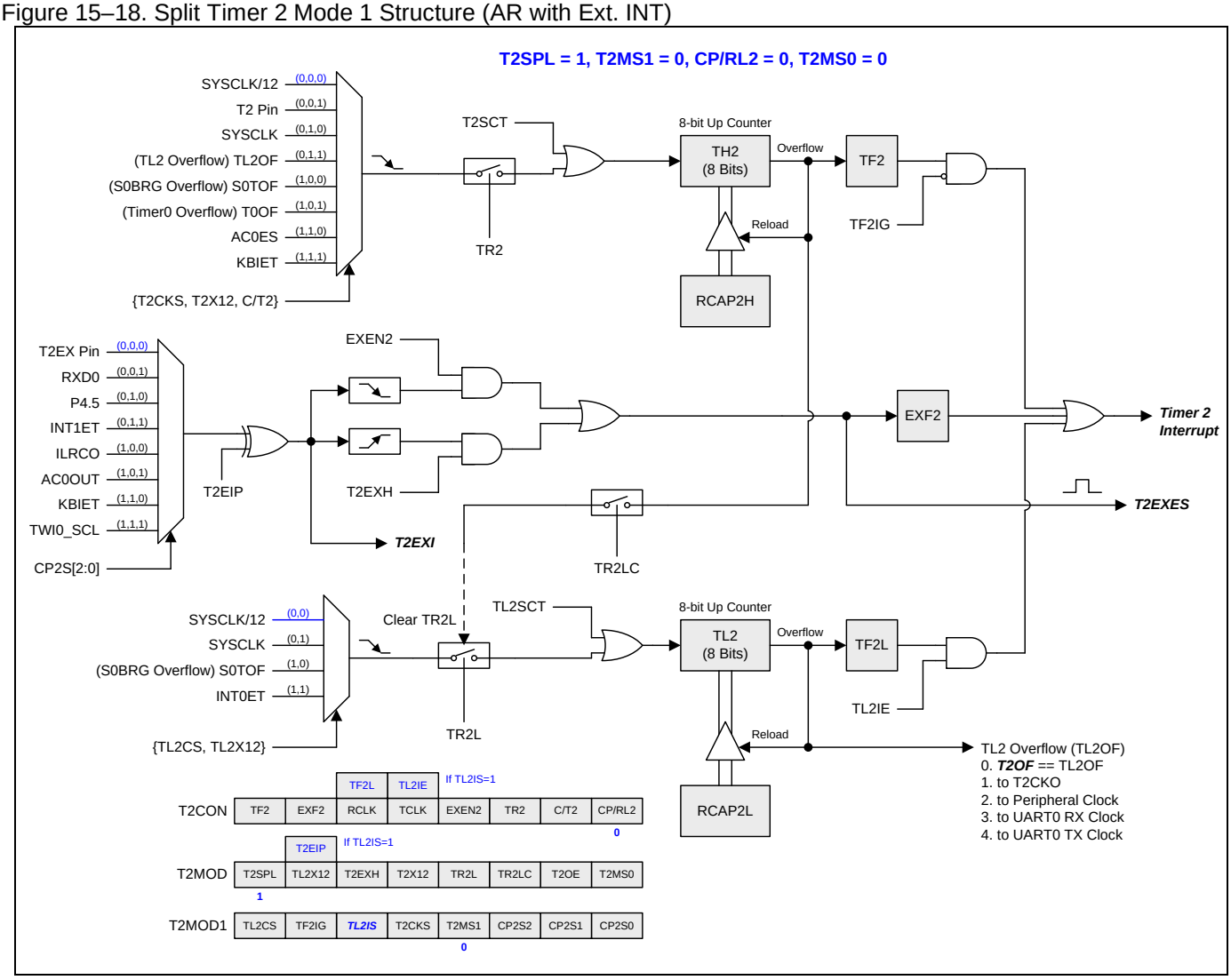
Figure 15–17. Split Timer 2 Mode 0 Structure (AR and Ext. INT)



15.2.7. Split Timer 2 Mode 1 (AR with Ext. INT)

When T2SPL is set in this mode, Timer 2 is split to two 8-bit timers as shown in Figure 15–18. It is similar function as Timer 2 Mode 1 and keeps the same interrupt scheme in Split Timer 2 Mode 0.

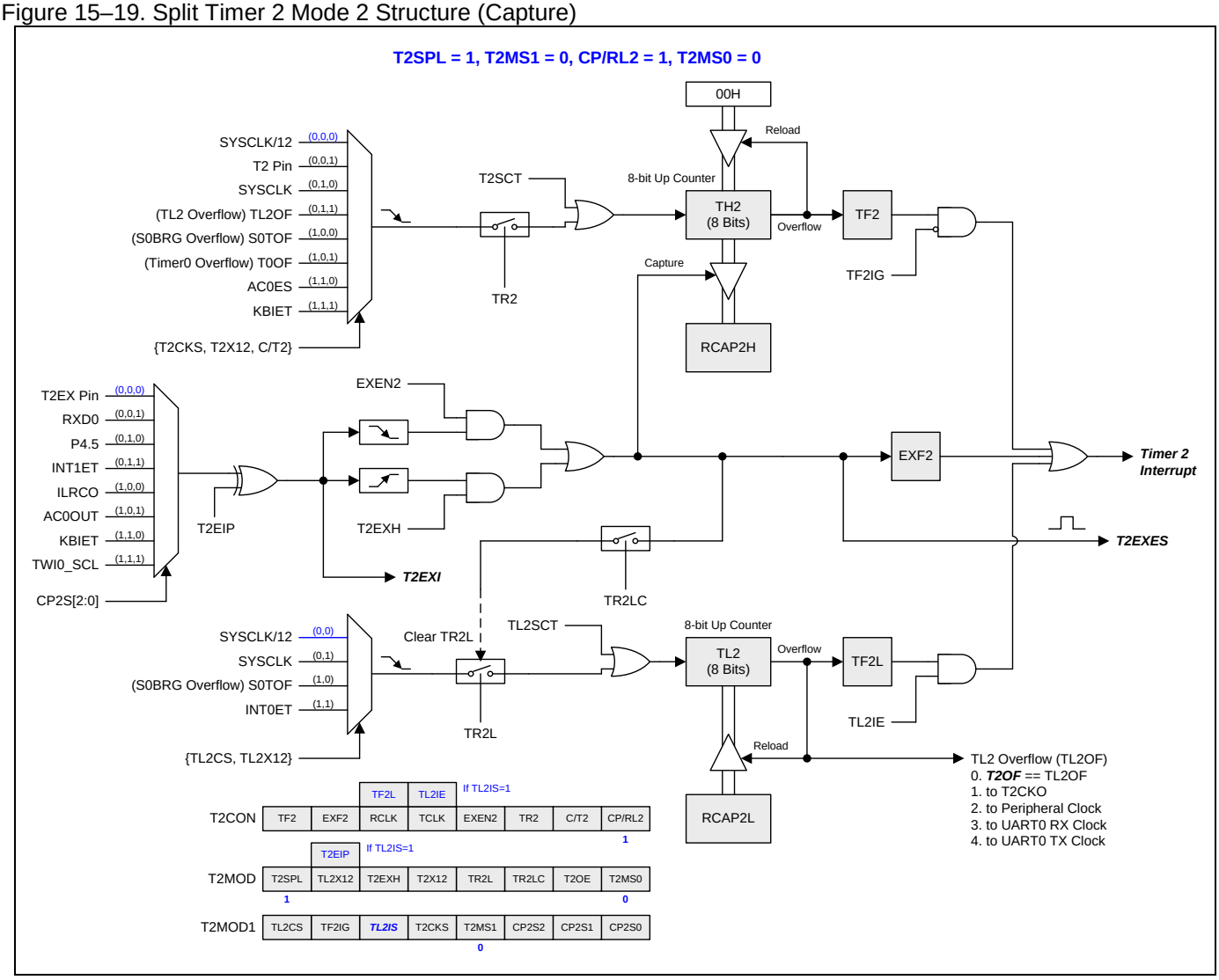
Figure 15–18. Split Timer 2 Mode 1 Structure (AR with Ext. INT)



15.2.8. Split Timer 2 Mode 2 (Capture)

When T2SPL is set in this mode, Timer 2 is split to two 8-bit timers as shown in Figure 15–19. It is similar function as Timer 2 Mode 2 and keeps the same interrupt scheme in Split Timer 2.

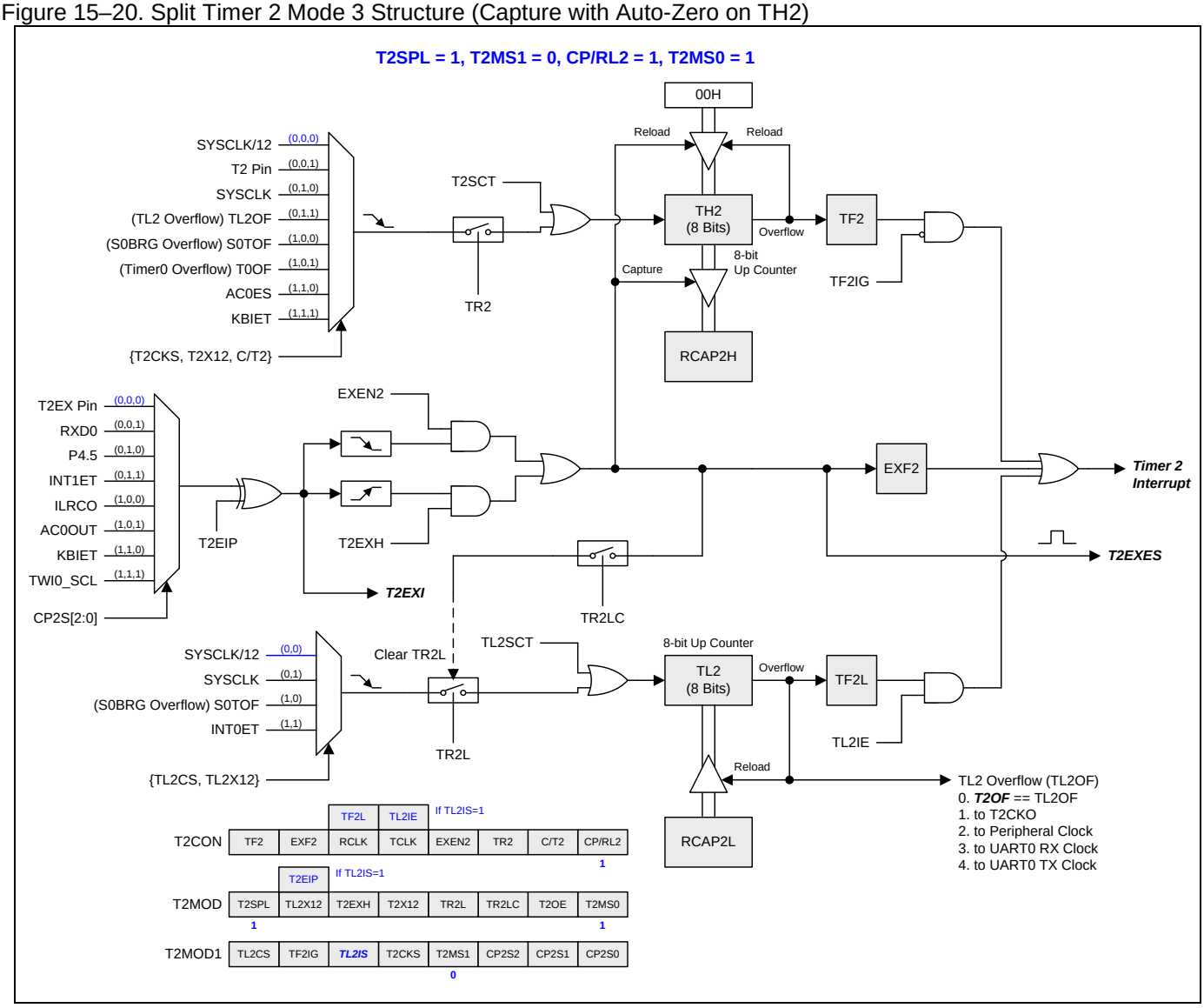
Figure 15–19. Split Timer 2 Mode 2 Structure (Capture)



15.2.9. Split Timer 2 Mode 3 (Capture with Auto-Zero)

When T2SPL is set in this mode, Timer 2 is split to two 8-bit timers as shown in Figure 15–20. It is similar function as Timer 2 Mode 3 and keeps the same interrupt scheme in Split Timer 2 Mode 0.

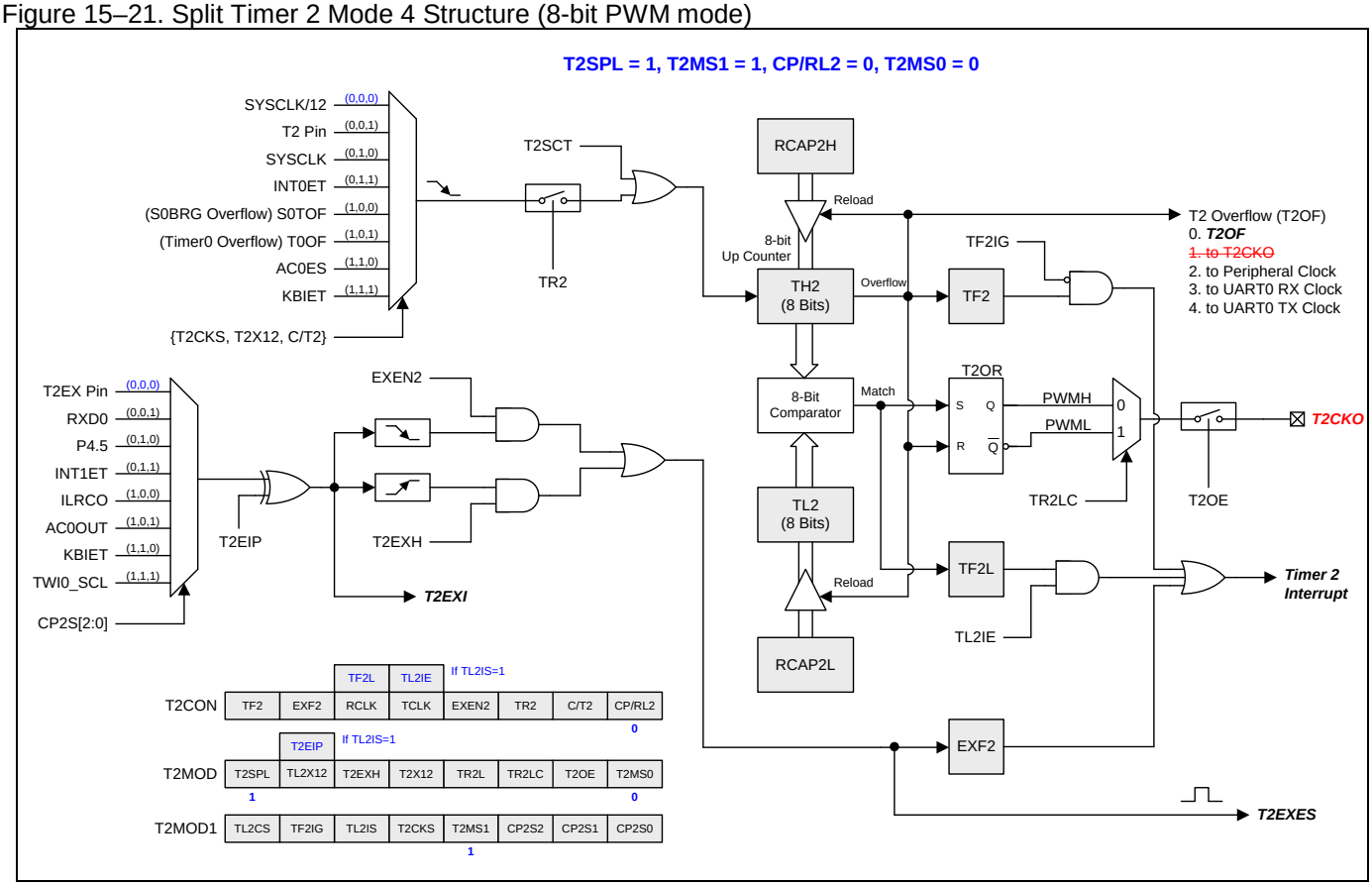
Figure 15–20. Split Timer 2 Mode 3 Structure (Capture with Auto-Zero on TH2)



15.2.10. Split Timer 2 Mode 4 (8-bit PWM Mode)

In this mode, Timer 2 is an 8-bit PWM mode as shown in Figure 15–21. TH2 and RCAP2H are combined to an 8-bit auto-reload counter. Software configures these two registers to decide the PWM cycle time. TL2 is the PWM compare register to generate PWM waveform. RCAP2L is the PWM buffer register and software will update PWM data in this register. Each TH2 overflow event will set TF2 and load RCAP2L value into TL2. The PWM signal will be output on T2CKO function pin and the output is gated by T2OE in T2MOD register.

Figure 15–21. Split Timer 2 Mode 4 Structure (8-bit PWM mode)



15.2.11. Baud-Rate Generator Mode (BRG)

Bits TCLK and/or RCLK in T2CON register allow the serial port transmit and receive baud rates to be derived from either Timer 1 or Timer 2. When TCLK=0, Timer 1 is used as the serial port transmit baud rate generator. When TCLK= 1, Timer 2 is used as the serial port transmit baud rate generator. RCLK has the same effect for the serial port receive baud rate. With these two bits, the serial port can have different receive and transmit baud rates – one generated by Timer 1, the other by Timer 2.

Figure 15–22 shows the Timer 2 in baud rate generation mode to generate RX Clock and TX Clock into UART engine (See Figure 17–6.). The baud rate generation mode is like the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by firmware.

The Timer 2 as a baud rate generator mode is valid only if RCLK and/or TCLK=1 in T2CON register. Note that a rollover in TH2 does set TF2. Thus, the Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode by setting TF2IG to block TF2 interrupt. Also if the EXEN2 (T2 external enable bit) is set, a 1-to-0 transition in T2EX (Timer/counter 2 trigger input) will set EXF2 (T2 external flag) but will not cause a reload from (RCAP2H, RCAP2L) to (TH2,TL2). Therefore when Timer 2 is in use as a baud rate generator, T2EX can be used as an additional external interrupt, if needed.

When Timer 2 is in the baud rate generator mode, one should not try to read or write TH2 and TL2. As a baud rate generator, Timer 2 is incremented at 1/2 the system clock or asynchronously from pin T2; under these conditions, a read or write of TH2 or TL2 may not be accurate. The RCAP2 registers may be read, but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

Note:

Refer to Section "17.8.4 Baud Rate in Mode 1 & 3" to get baud rate setting value when using Timer 2 as the baud rate generator.

If Timer 2 in Split Mode 0, TL2 and RCAP2L are combined to an 8-bit baud-rate generator as shown in Figure 15–23. TL2 overflow sets the TF2L which interrupt is enabled by TL2IE. TH2 and RCAP2H act as an 8-bit auto-reload timer/counter function with Timer 2 interrupt capability.

Figure 15–22. Timer 2 in Baud-Rate Generator Mode

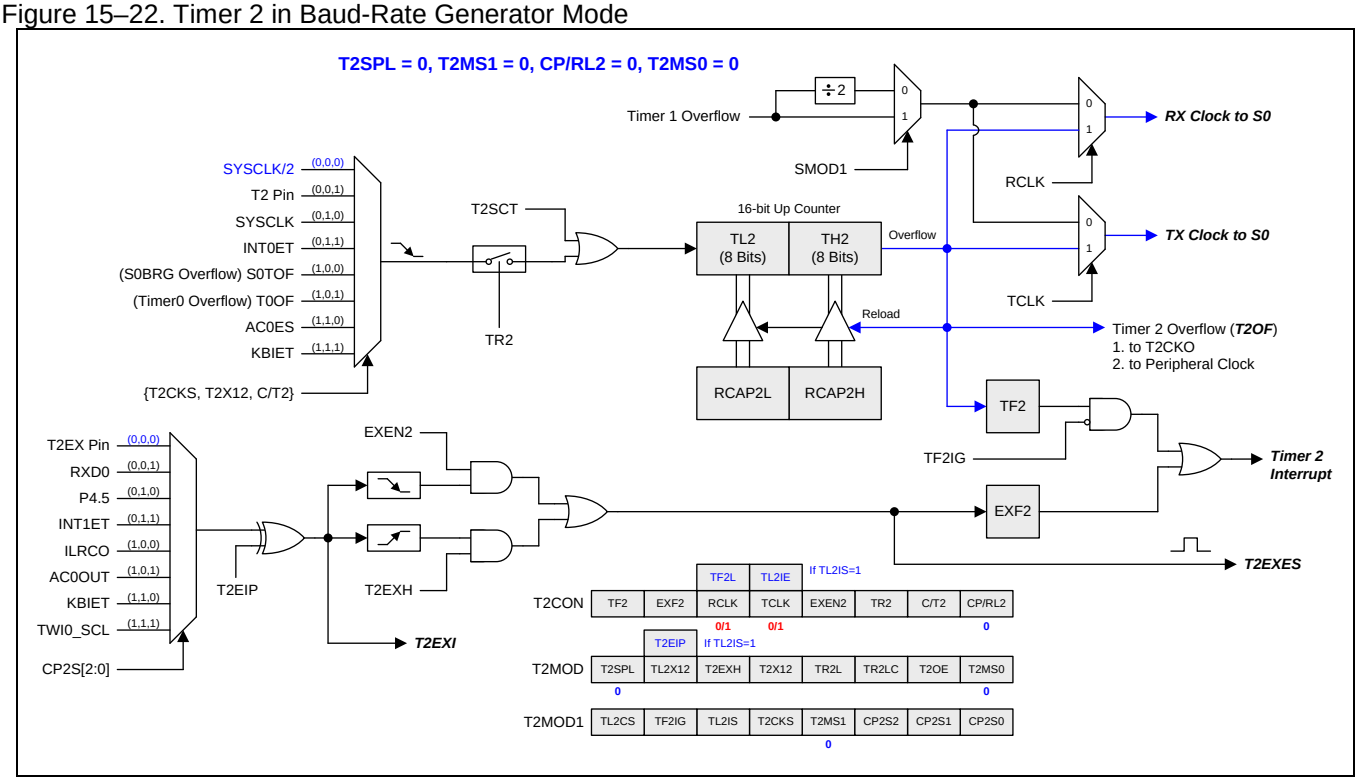
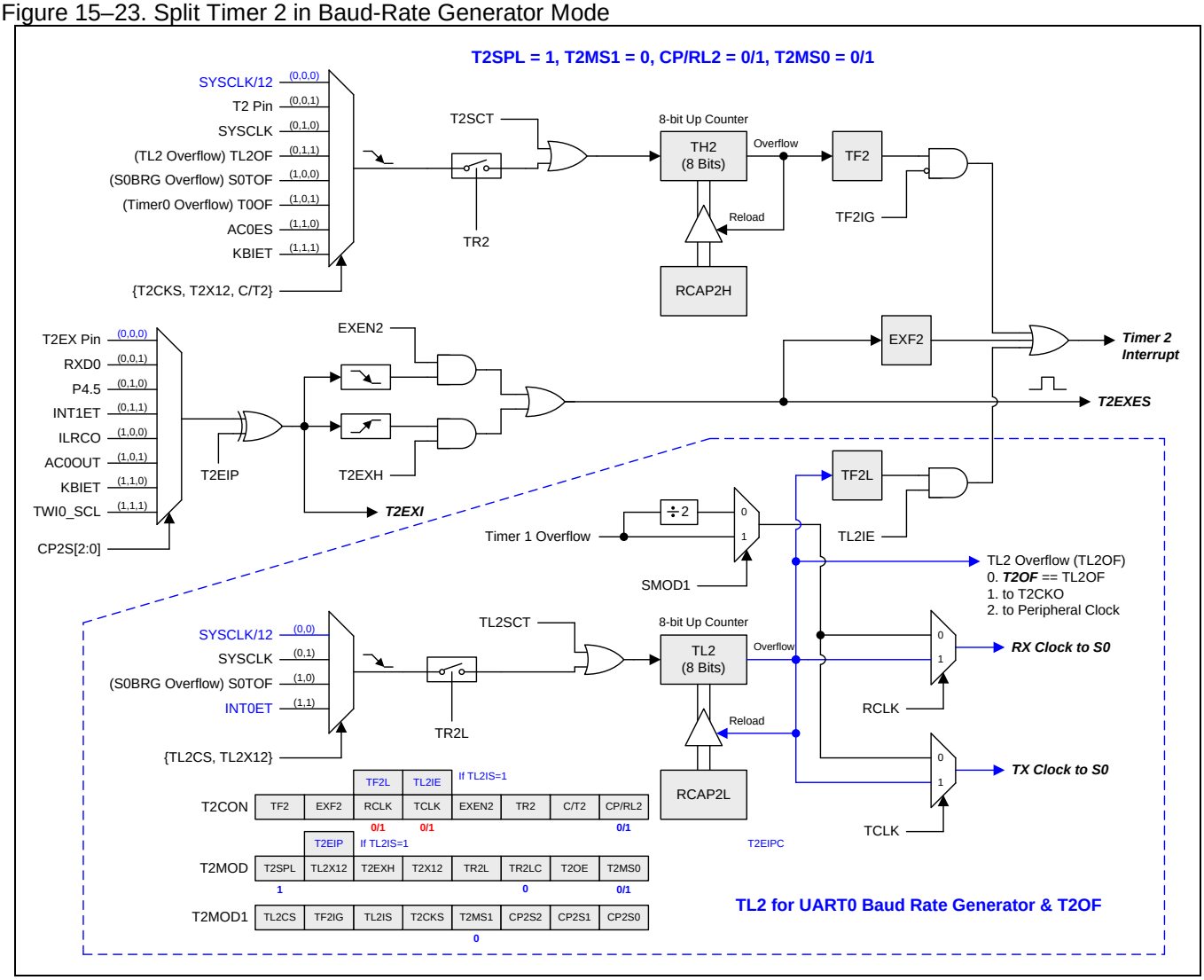


Figure 15–23. Split Timer 2 in Baud-Rate Generator Mode



15.2.12. Timer 2 Programmable Clock Output

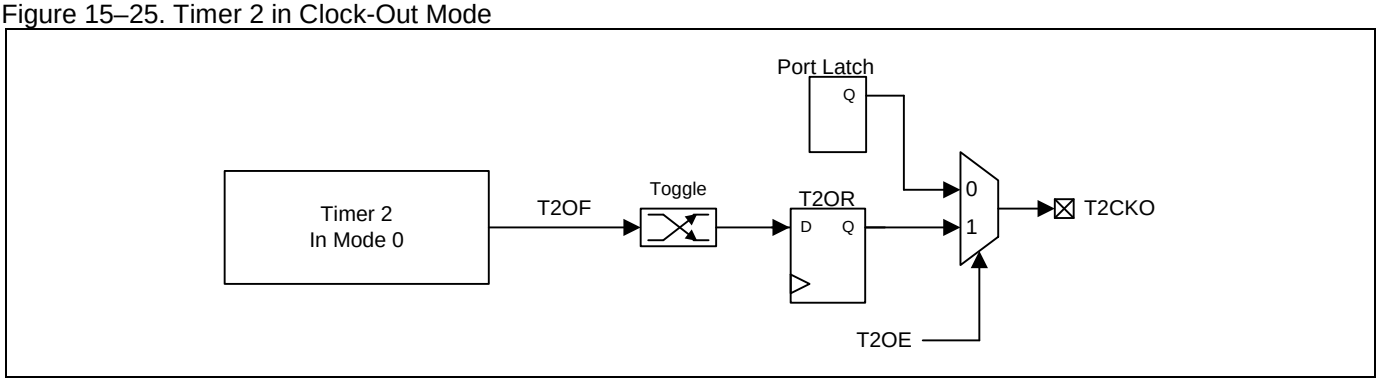
Timer 2 has a Clock-Out Mode (while CP/RL2=0 & T2OE=1). In this mode, Timer 2 operates as a programmable clock generator with 50% duty-cycle. The generated clocks come out on T2CKO. The input clock (SYSCLK/2, SYSCLK/12 or SYSCLK) increments the 16-bit timer (TH2, TL2). The timer repeatedly counts to overflow from a loaded value. Once overflows occur, the contents of (RCAP2H, RCAP2L) are loaded into (TH2, TL2) for the consecutive counting. Figure 15–24 gives the formula of Timer 2 clock-out frequency: Figure 15–25 shows the clock structure of Timer 2.

Figure 15–24. Timer 2 clock out equation

$$\text{T2 Clock-out Frequency} = \frac{\text{T2 Clock Frequency}}{2 \times (65536 - (\text{RCAP2H, RCAP2L}))}$$

Note:
(1) Timer 2 overflow flag, TF2, will be set when Timer 2 overflows to generate interrupt. But, the TF2 interrupt can be blocked by TF2IG in T2MOD1 register.
(2) For SYSCLK=12MHz and select SYSCLK/12 as Timer 2 clock source, Timer 2 has a programmable output frequency range from 45.7Hz to 3MHz.
(3) For SYSCLK=12MHz and select SYSCLK as Timer 2 clock source, Timer 2 has a programmable output frequency range from 91.5Hz to 6MHz.

Figure 15–25. Timer 2 in Clock-Out Mode



How to Program Timer 2 in Clock-out Mode

- Select Timer 2 clock source.
- Determine the 16-bit reload value from the formula and enter it in the RCAP2H and RCAP2L registers.
- Enter the same reload value as the initial value in the TH2 and TL2 registers.
- Set T2OE bit in T2MOD register.
- Set TR2 bit in T2CON register to start the Timer 2.

In the Clock-Out mode, Timer 2 rollovers will also generate a TF2 interrupt. This is similar to when Timer 2 is used as a baud-rate generator. It is possible to use Timer 2 as a baud rate generator and a clock generator simultaneously. Note, however, that the baud-rate and the clock-out frequency depend on the same overflow rate of Timer 2 and its interrupt will be blocked by TF2IG.

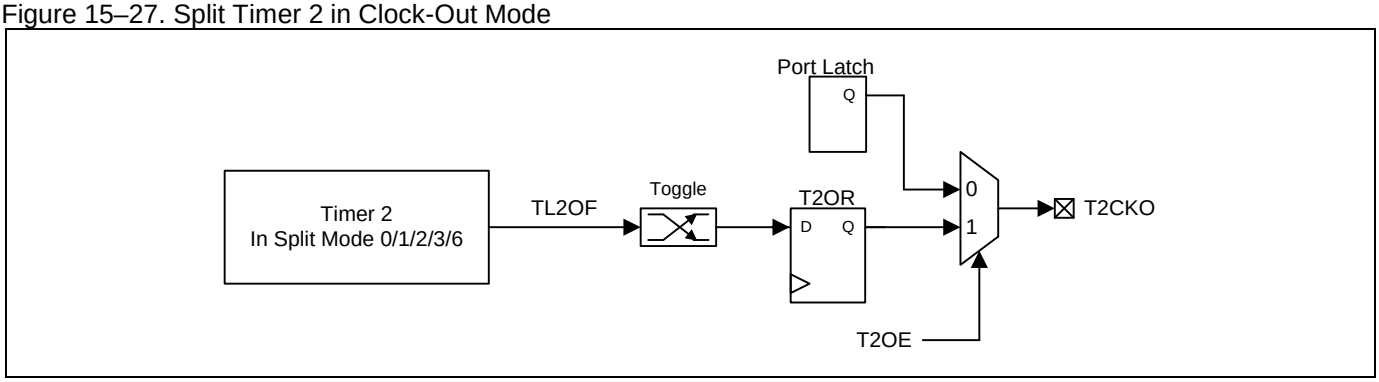
If Timer 2 in split mode, the clock output function is generated by TL2 overflow and the output clock frequency is TL2 overflow rate /2. RCAP2L is the TL2's reload value when TL2 overflow. There are four clock source selections for TL2. Before enable split Timer 2 clock output function, software must finish the TL2 clock source configuration. Figure 15–26 gives the formula of TL2 clock-out frequency: Figure 15–27 shows the clock structure of Split Timer 2.

Figure 15–26. Split Timer 2 clock out equation

$$\text{Split T2 Clock-out Frequency} = \frac{\text{TL2 Clock Frequency}}{2 \times (256 - \text{RCAP2L})}$$

Note:
(1) TL 2 overflow flag, TF2L, will be set when TL2 overflows to generate interrupt. But, the TF2L interrupt is enabled by TL2IE in T2CON register.
(2) For SYSCLK=12MHz and select SYSCLK/12 as TL2 clock source, TL2 has a programmable output frequency range from 1.95KHz to 500KHz.
(3) For SYSCLK=12MHz and select SYSCLK as TL2 clock source, TL2 has a programmable output frequency range from 23.44Hz to 6MHz.

Figure 15–27. Split Timer 2 in Clock-Out Mode



How to Program Split Timer 2 in Clock-out Mode

- Select TL2 clock source.
- Determine the 8-bit reload value from the formula and enter it in the RCAP2L register.
- Enter the same reload value as the initial value in the TL2 register.
- Set T2OE bit in T2MOD register.
- Set TR2L bit in T2CON register to start the Timer 2.

In the Clock-Out mode, TL2 rollovers will also generate an interrupt, TF2L. This is similar to when TL2 is used as a baud-rate generator. It is possible to use TL2 as a baud rate generator and a clock generator simultaneously. Note, however, that the baud-rate and the clock-out frequency depend on the same overflow rate of TL2 in split Timer 2. The TF2L interrupt is enabled by TL2IE in T2CON register.

15.2.13. Timer 2 Register

T2CON: Timer 2 Control Register

SFR Page = **0 Only**

SFR Address = 0xC8 RESET = 0000-0000

7	6	5	4	3	2	1	0
TF2	EXF2	RCLK/ TF2L	TCLK/ TL2IE	EXEN2	TR2	C/T2	CP/RL2
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: TF2, Timer 2 overflow flag.
0: TF2 must be cleared by software.
1: TF2 is set by a Timer 2 overflow happens. TF2 will not be set when either RCLK=1 or TCLK=1.

Bit 6: EXF2, Timer 2 external flag.
0: EXF2 must be cleared by software.
1: Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX pin and EXEN2=1 or a positive transition on T2EX and T2EXH=1. When Timer 2 interrupt is enabled, EXF2=1 will cause the CPU to vector to the Timer 2 interrupt routine.

TL2IS (T2MOD1.5) must be cleared to enable access to the RCLK bit.
Bit 5: RCLK, Receive clock flag.
0: Causes Timer 1 overflow to be used for the receive clock.
1: Causes the serial port to use Timer 2 overflow pulses for its receive clock in modes 1 and 3.

TL2IS (T2MOD1.5) must be set to enable access to the TF2L bit.
Bit 5: TF2L, TL2 overflow flag in Timer 2 split mode.
0: TF2L must be cleared by software.
1: TF2L is set by TL2 overflow happened in Timer 2 split mode.

TL2IS (T2MOD1.5) must be cleared to enable access to the TCLK bit.
Bit 4: TCLK, Transmit clock flag.
0: Causes Timer 1 overflows to be used for the transmit clock.
1: Causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3.

TL2IS (T2MOD1.5) must be set to enable access to the TL2IE bit.
Bit 4: TL2IE, TF2L interrupt enable.
0: Disable TF2L interrupt.
1: Enable TF2L interrupt to share the Timer 2 interrupt vector.

Bit 3: EXEN2, Timer 2 external enable flag on a negative transition of T2EX pin.
0: Cause Timer 2 to ignore negative transition events at T2EX pin.
1: Allows a capture or reload to occur as a result of a 1-to-0 transition on T2EX pin if Timer 2 is not being used to clock the serial port 0. If Timer 2 is configured to clock the serial port 0, the T2EX remains the external transition detection and reports on EXF2 flag with Timer 2 interrupt.

Bit 2: TR2, Timer 2 Run control bit. If in Timer 2 split mode, it only controls the TH2.
0: Disabled to stop the Timer/Counter 2.
1: Enabled to start the Timer/Counter 2.

Bit 1: C/T2, Timer 2 clock or counter source selector. The function is active with T2X12 and T2CKS as following definition:

T2CKS, T2X12, C/T2	Timer 2 Clock Selection	TH2 Clock Selection in split mode
0 0 0	SYSCLK/12	SYSCLK/12
0 0 1	T2 Pin	T2 Pin
0 1 0	SYSCLK	SYSCLK
0 1 1	INT0ET	TL2OF
1 0 0	S0TOF	S0TOF
1 0 1	T0OF	T0OF
1 1 0	AC0ES	AC0ES
1 1 1	KBIET	KBIET

Bit 0: CP/RL2, Timer 2 mode control bit. Refer T2MOD.T2MS0 description for the function definition.

T2MOD: Timer 2 Mode Register

SFR Page = **0 Only**

SFR Address = 0xC9 RESET = 0000-0000

7	6	5	4	3	2	1	0
T2SPL	TL2X12/ T2EIP	T2EXH	T2X12	TR2L	TR2LC	T2OE	T2MS0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: T2SPL, Timer 2 split mode control.
0: Disable Timer 2 to split mode.
1: Enable Timer 2 to split mode. Mode 6 split mode is reserved.

TL2IS (T2MOD1.5) must be cleared to enable access to the TL2X12 bit.
Bit 6: TL2X12, the clock control bit of TL2 in Timer 2 split mode.

TL2CS, TL2X12	TL2 Clock Selection
0 0	SYSCLK/12
0 1	SYSCLK
1 0	S0TOF
1 1	INT0ET

TL2IS (T2MOD1.5) must be set to enable access to the T2EIP bit.
Bit 6: T2EIP, T2EXI input signal inversion control bit.
0: T2EXI input signal is not inverted.
1: T2EXI input signal is inverted.

Bit 5: T2EXH, Timer 2 external enable flag on a positive transition of T2EX pin.
0: Cause Timer 2 to ignore positive transition events at T2EX pin.
1: Allows a capture or reload to occur as a result of a 0-to1 transition on T2EX pin if Timer 2 is not being used to clock the serial port 0. If Timer 2 is configured to clock the serial port 0, the T2EX remains the external transition detection and reports on EXF2 flag with Timer 2 interrupt.

Bit 4: T2X12, Timer 2 clock source selector. Refer to C/T2 description for the function defined.

Bit 3: TR2L, TL2 Run control bit in Timer 2 split mode.
0: Disabled to stop the TL2.
1: Enabled to start the TL2.

Bit 2: TR2LC, TR2L Cleared control.
0: Disabled the TR2L cleared by hardware event.
1: Enabled the TR2L cleared by the TH2 overflow (Timer 2 in mode 0/1) or capture input (Timer 2 in mode 2/3).

Bit 1: T2OE, Timer 2 clock-out enable bit.
0: Disable Timer 2 clock output.
1: Enable Timer 2 clock output.

Bit 0: T2MS0, Timer 2 mode select bit 0.

T2MS1, CP/RL2, T2MS0	Timer 2 Mode Selection
0 0 0	Mode 0: Auto-Reload and External Interrupt
0 0 1	Mode 1: Auto-Reload with External Interrupt
0 1 0	Mode 2: Capture mode
0 1 1	Mode 3: Capture with Auto-Zero
1 0 0	Mode 4: 8-bit PWM if T2SPL = 1
1 1 0	Mode 6: Duty Capture
Others	Reserved

T2MOD1: Timer 2 Mode Register 1

SFR Page = **1 Only**

SFR Address = 0x93 RESET = 0000-0000

7	6	5	4	3	2	1	0
TL2CS	TF2IG	TL2IS	T2CKS	T2MS1	CP2S2	CP2S1	CP2S0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: TL2CS. TL2 clock selection in Timer 2 split mode. Refer to T2MOD.TL2X12 description for the function defined.

Bit 6: TF2IG, TF2 interrupt Ignored.
0: Enabled TF2 interrupt. Default is enabled.
1: Disable TF2 interrupt.

Bit 5: TL2IS, TF2L, TL2IE and T2EIP access control.
0: Enable RCLK and TCLK access function on T2CON.5~4, TL2X12 access on T2MOD.6.
1: Enable TF2L and TL2IE access function on T2CON.5~4, T2EIP access on T2MOD.6.

Bit 4: T2CKS, Timer 2 clock selection. Refer to C/T2 description for the function defined.

Bit 3: T2MS1, Timer 2 mode selection bit 1. Refer T2MOD.T2MS0 description for the function definition.

Bit 2~0: CP2S.2~0. These bits define the capture source selector of Timer 2.

CP2S.2~0	Timer 2 Capture Source Selection
0 0 0	T2EX Pin
0 0 1	RXD0
0 1 0	P4.5 Pin
0 1 1	INT1ET
1 0 0	ILRCO
1 0 1	AC0OUT
1 1 0	KBIET
1 1 1	TWI0_SCL

TL2: Timer 2 Low byte Register

SFR Page = **0 Only**

SFR Address = 0xCC RESET = 0000-0000

7	6	5	4	3	2	1	0
TL2.7	TL2.6	TL2.5	TL2.4	TL2.3	TL2.2	TL2.1	TL2.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

TH2: Timer 2 High byte Register

SFR Page = **0 Only**

SFR Address = 0xCD RESET = 0000-0000

7	6	5	4	3	2	1	0
TH2.7	TH2.6	TH2.5	TH2.4	TH2.3	TH2.2	TH2.1	TH2.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

RCAP2L: Timer 2 Capture Low byte Register

SFR Page = **0 Only**

SFR Address = 0xCA RESET = 0000-0000

7	6	5	4	3	2	1	0
RCAP2L.7	RCAP2L.6	RCAP2L.5	RCAP2L.4	RCAP2L.3	RCAP2L.2	RCAP2L.1	RCAP2L.1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

RCAP2H: Timer 2 Capture High byte Register

SFR Page = **0 Only**

SFR Address = 0xCB RESET = 0000-0000

7	6	5	4	3	2	1	0
RCAP2H.7	RCAP2H.6	RCAP2H.5	RCAP2H.4	RCAP2H.3	RCAP2H.2	RCAP2H.1	RCAP2H.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

AUXR4: Auxiliary Register 4

SFR Page = **1 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
T2PS1	T2PS0	0	T1PS0	0	0	AC0OE	AC0FLT1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: T2PS1~0, Timer 2 Port pin Selection [1:0].

T2PS1~0	T2/T2CKO	T2EX
0 0	P3.1	P3.0
0 1	P3.3	P4.6
1 0	P4.6	P3.3
1 1	P4.5	P4.4

15.3. Timer Global Control

When the applications are asking all timers work together in sync mode, it can set the registers to Start, Reload and Stop the timers.

15.3.1. Global Enable for all Timer Run

When the applications are asking all timers work together in sync mode, just need to set the TRxE or TRxLE in TREN0 to start the timer at the same time. Those registers will be auto cleared by hardware after writing “1” into it.

TREN0: Timer Run Enable Register 0

SFR Page = **1 Only**

SFR Address = 0x95 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	TR2LE	0	0	TR2E	TR1E	TR0E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7 ~ 6: Reserved. Software must write “0” on this bit when TREN0 is written.

Bit 5, TR2LE, write “1” on this bit to set TR2L enabled (TR2L=1) when Timer 2 in split mode. This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 4 ~ 3: Reserved. Software must write “0” on this bit when TREN0 is written.

Bit 2, TR2E, write “1” on this bit to set TR2 enabled (TR2=1). This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 1, TR1E, write “1” on this bit to set TR1 enabled (TR1=1). This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 0, TR0E, write “1” on this bit to set TR0 enabled (TR0=1). This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

15.3.2. Global Control for all Timer Reload

TRLC0: Timer Reload Control Register 0

SFR Page = **2 Only**

SFR Address = 0x95 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	TL2RLC	0	0	T2RLC	T1RLC	T0RLC
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7 ~ 6: Reserved. Software must write “0” on this bit when TRLC0 is written.

Bit 5, TL2RLC, write “1” on this bit to force TL2 reload condition happened when Timer 2 in split mode. This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 4 ~ 3: Reserved. Software must write “0” on this bit when TRLC0 is written.

Bit 2, T2RLC, write “1” on this bit to force TH2 and TL2 reload condition happened when Timer 2 not in split mode. Or force TH2 reload condition happened when Timer 2 in split mode. The force reload is not available if the timer in duty capture mode. This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 1, T1RLC, write “1” on this bit to force TH1/TL1 reload condition happened. This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 0, T0RLC, write “1” on this bit to force TH0/TL0 reload condition happened. This bit is auto-cleared by hardware

after writing “1” operation. Write “0” on this bit is no action.

15.3.3. Global Control for all Timer Stop

TSPC0: Timer Stop Control Register 0

SFR Page = **3 Only**

SFR Address = 0x95 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	TL2SC	0	0	T2SC	T1SC	T0SC
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7 ~ 6: Reserved. Software must write “0” on this bit when TSPC0 is written.

Bit 5, TL2SC, write “1” on this bit to set TR2L disabled (TR2L=0) when Timer 2 in split mode. This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 4 ~ 3: Reserved. Software must write “0” on this bit when TSPC0 is written.

Bit 2, T2SC, write “1” on this bit to set TR2 disabled (TR2=0). This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 1, T1SC, write “1” on this bit to set TR1 disabled (TR1=0). This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

Bit 0, T0SC, write “1” on this bit to set TR0 disabled (TR0=0). This bit is auto-cleared by hardware after writing “1” operation. Write “0” on this bit is no action.

16. Programmable Counter Array (PCA0)

The **MG82F6B08 / 6B001/ 6B104** is equipped with a Programmable Counter Array (PCA0), which provides more timing capabilities with less CPU intervention than the standard timer/counters. Its advantages include reduced software overhead and improved accuracy.

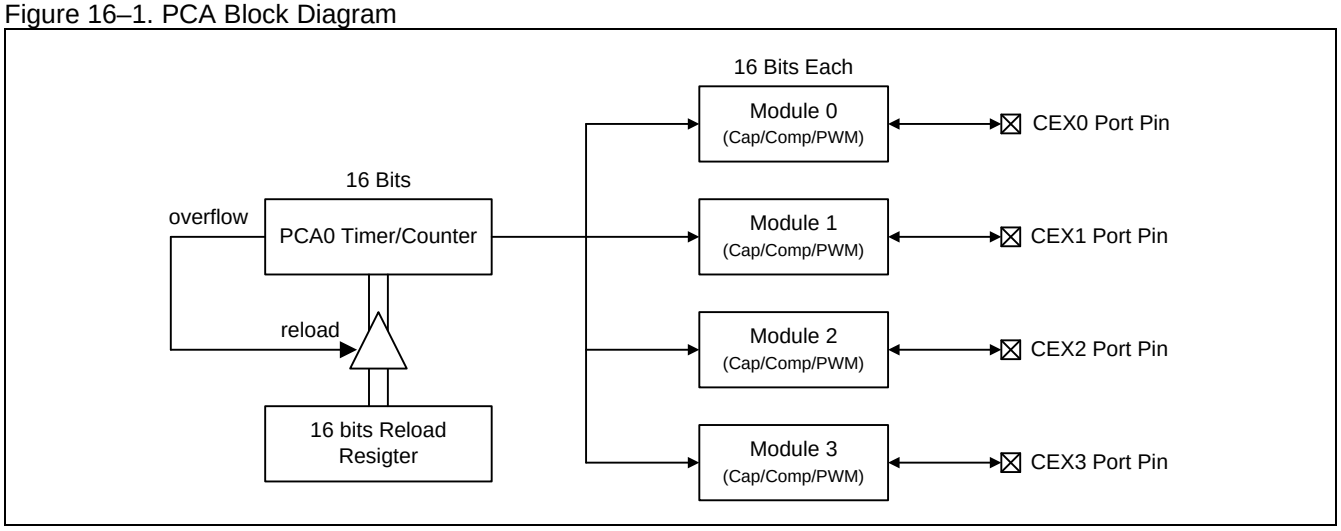
16.1. PCA Overview

The PCA consists of a dedicated timer/counter which serves as the time base for an array of **Four** capture /compare/PWM modules. Figure 16–1 shows a block diagram of the PCA. Notice that the PCA timer and modules are all 16-bits. If an external event is associated with a module, that function is shared with the corresponding Port pin. If the module is not using the port pin, the pin can still be used for standard I/O.

Module 0~3 can be programmed in any one of the following modes:
- Rising and/or Falling Edge Capture
- Software Timer (Compare)
- High Speed Output (Compare Output)
- Pulse Width Modulator Output (PWM)
- Compare Output on PWM Match case (COPM)

All of these modes will be discussed later in detail. However, let's first look at how to set up the PCA timer and modules.

Figure 16–1. PCA Block Diagram



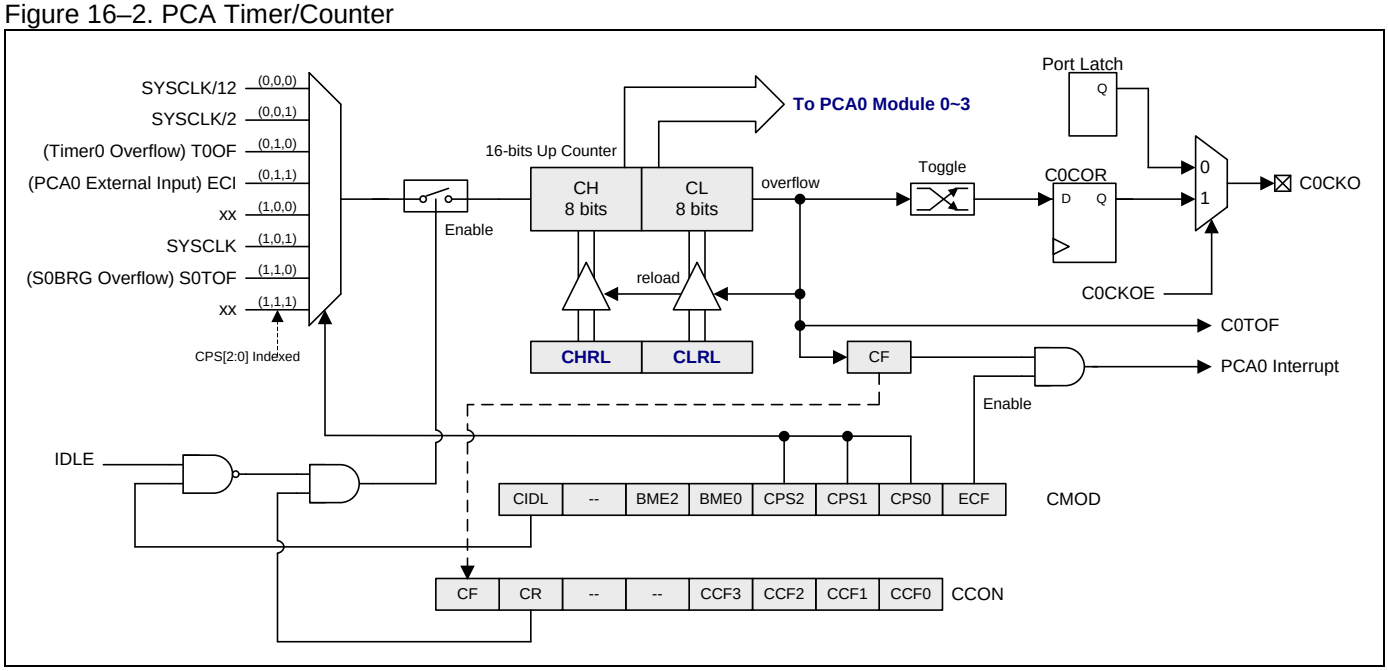
16.2. PCA Timer/Counter

The timer/counter for the PCA is a auto-reload 16-bit timer consisting of registers CH and CL (the high and low bytes of the count values), CHRL, CLRL (the high and low bytes reload registers), as shown in Figure 16–2. CHRL and CLRL are reloaded to CH and CL at each time overflow on {CH+CL} counter which can change the PCA cycle time for variable PWM resolution, such as 7-bit or 9-bit PWM.

{CH + CL} is the common time base for all modules and its clock input can be selected from the following source:
- 1/12 the system clock frequency,
- 1/2 the system clock frequency,
- The Timer 0 overflow, which allows for a range of slower clock inputs to the timer,
- External clock input, 1-to-0 transitions, on ECI pin,
- Directly from the system clock frequency,
- The S0BRG overflow, S0TOF,

Special Function Register CMOD contains the Count Pulse Select bits (CPS2, CPS1 and CPS0) to specify the PCA timer input. This register also contains the ECF bit which enables an interrupt when the counter {CH+CL} overflows. And the counter overflow toggles C0COR, it will output on port pin when C0CKOE is enabled. In addition, the user has the option of turning off the PCA timer during Idle Mode by setting the Counter Idle bit (CIDL). This can further reduce power consumption during Idle mode.

Figure 16–2. PCA Timer/Counter



CMOD: PCA Counter Mode Register

SFR Page = 0 Only

SFR Address = 0xD9 RESET = 0000-0000

7	6	5	4	3	2	1	0
CIDL	0	BME2	BME0	CPS2	CPS1	CPS0	ECF
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: CIDL, PCA counter Idle control.
0: Lets the PCA counter continue functioning during Idle mode.
1: Lets the PCA counter be gated off during Idle mode.

Bit 5: BME2, Buffer Mode Enable on PCA module 2/3. It is only valid on both of PCA module 2 and module 3 in capture mode, PWM mode or COPM mode.
0: PCA Module 2/3 buffer mode disabled.
1: PCA Module 2/3 buffer mode enabled.

Bit 4: BME0, Buffer Mode Enable on PCA module 0/1. It is only valid on both of PCA module 0 and module 1 in capture mode, PWM mode or COPM mode.
0: PCA Module 0/1 buffer mode disabled.
1: PCA Module 0/1 buffer mode enabled.

Bit 3~1: CPS2-CPS0, PCA counter clock source select bits.

CPS2	CPS1	CPS0	PCA Clock Source
0	0	0	Internal clock, (system clock)/12
0	0	1	Internal clock, (system clock)/2
0	1	0	Timer 0 overflow
0	1	1	External clock at the ECI pin
1	0	0	Reserved
1	0	1	Internal clock, (system clock)/1
1	1	0	S0BRT overflow
1	1	1	Reserved

Bit 0: ECF, Enable PCA counter overflow interrupt.
0: Disables an interrupt when CF bit (in CCON register) is set.
1: Enables an interrupt when CF bit (in CCON register) is set.

The CCON register shown below contains the run control bit for the PCA and the flags for the PCA timer and each module. To run the PCA the CR bit (CCON.6) must be set by software. The PCA is shut off by clearing this bit. The CF bit (CCON.7) is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in the CMOD register is set. The CF bit can only be cleared by software. CCF0 to CCF3 are the interrupt flags for module 0 to module 3, respectively, and they are set by hardware when either a match or a capture occurs. These flags also can only be cleared by software. The PCA interrupt system is shown Figure 16–3.

CCON: PCA Counter Control Register

SFR Page = 0 only
SFR Address = 0xD8 RESET = 0000-0000

7	6	5	4	3	2	1	0
CF	CR	0	0	CCF3	CCF2	CCF1	CCF0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: CF, PCA Counter Overflow flag.
0: Only be cleared by software.
1: Set by hardware when the counter rolls over. CF flag can generate an interrupt if bit ECF in CMOD is set. CF may be set by either hardware or software.

Bit 6: CR, PCA Counter Run control bit.
0: Must be cleared by software to turn the PCA counter off.
1: Set by software to turn the PCA counter on.

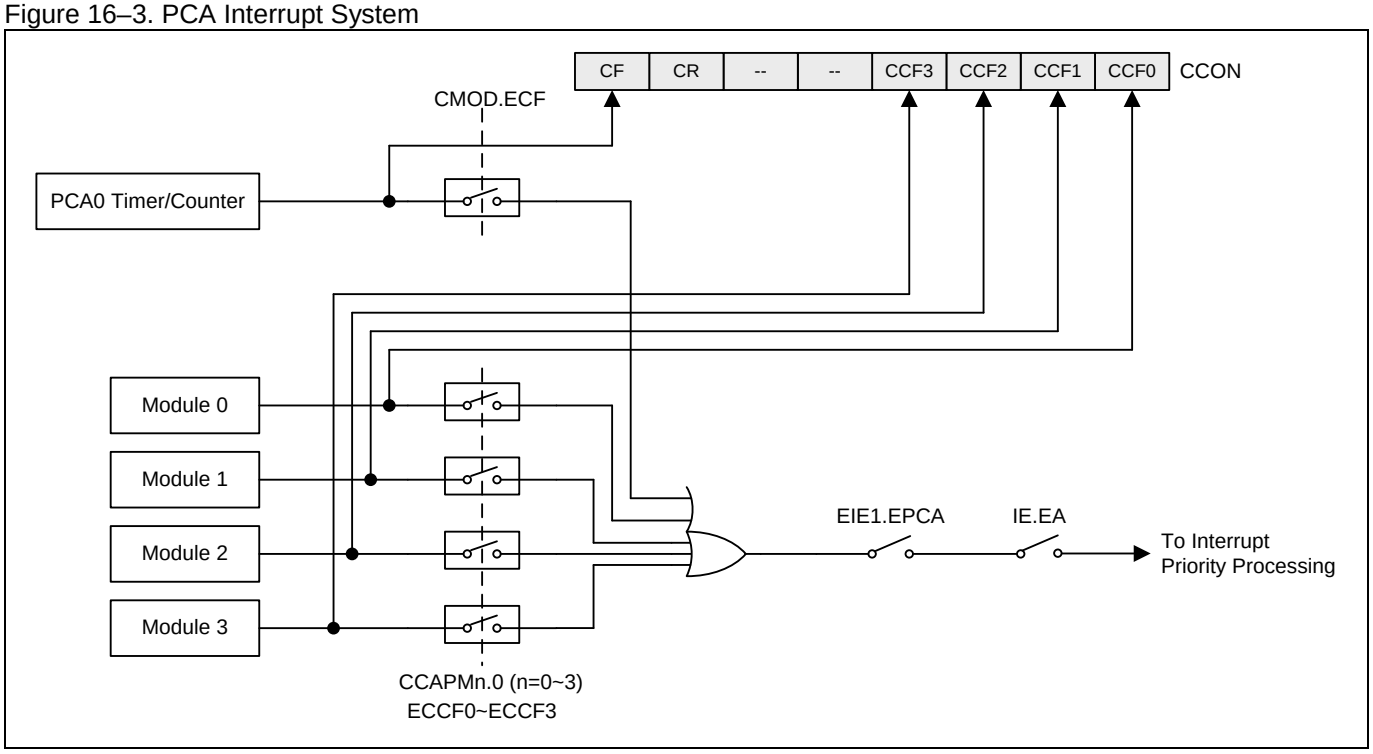
Bit 3: CCF3, PCA Module 3 interrupt flag.
0: Must be cleared by software.
1: Set by hardware when a match or capture occurs.

Bit 2: CCF2, PCA Module 2 interrupt flag.
0: Must be cleared by software.
1: Set by hardware when a match or capture occurs.

Bit 1: CCF1, PCA Module 1 interrupt flag.
0: Must be cleared by software.
1: Set by hardware when a match or capture occurs.

Bit 0: CCF0, PCA Module 0 interrupt flag.
0: Must be cleared by software.
1: Set by hardware when a match or capture occurs.

Figure 16–3. PCA Interrupt System



CH: PCA base timer High

SFR Page = 0 ~ F
SFR Address = 0xF9 RESET = 0000-0000

7	6	5	4	3	2	1	0
CH.7	CH.6	CH.5	CH.4	CH.3	CH.2	CH.1	CH.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

CL: PCA base timer Low

SFR Page = 0 ~ F
SFR Address = 0xE9 RESET = 0000-0000

7	6	5	4	3	2	1	0
CL.7	CL.6	CL.5	CL.4	CL.3	CL.2	CL.1	CL.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

CHRL: PCA CH Reload Register

SFR Page = 0 ~ F
SFR Address = 0xCF RESET = 0000-0000

7	6	5	4	3	2	1	0
CHRL.7	CHRL.6	CHRL.5	CHRL.4	CHRL.3	CHRL.2	CHRL.1	CHRL.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~0: CHRL, reload value of CH.

CLRL: PCA CL Reload Register

SFR Page = 0 ~ F
SFR Address = 0xCE RESET = 0000-0000

7	6	5	4	3	2	1	0
CLRL.7	CLRL.6	CLRL.5	CLRL.4	CLRL.3	CLRL.2	CLRL.1	CLRL.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~0: CLRL, reload value of CL.

16.3. Compare/Capture Modules

Each of the compare/capture module 0~3 has a mode register called CCAPMn (n = 0,1,2,3) to select which function it will perform. Note the ECCFn bit which enables an interrupt to occur when a module's interrupt flag is set.

CCAPMn: PCA Module Compare/Capture Register, n=0~3
SFR Page = 0 only for n= 0~1 (n=2~*3* for all page)
SFR Address = 0xDA~0xDD RESET = 0000-0000

7	6	5	4	3	2	1	0
DTEn	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: DTEn. Enable Dead-Time control on PWMHn/PWMLn output pair. This bit is only valid on n= 0, 2 and the dead-time function is active when PWM channel is operating in buffer mode. The channel buffer mode is enabled by BME0, BME2 in CMOD.
0: Disable the Dead-Time control on PWMn output.
1: Enable the Dead-Time control on PWMn output.

Bit 6: ECOMn, Enable Comparator.
0: Disable the digital comparator function.
1: Enables the digital comparator function.

Bit 5: CAPPn, Capture Positive enabled.
0: Disable the PCA capture function on CEXn positive edge detected.
1: Enable the PCA capture function on CEXn positive edge detected.

Bit 4: CAPNn, Capture Negative enabled.
0: Disable the PCA capture function on CEXn negative edge detected.
1: Enable the PCA capture function on CEXn negative edge detected.

Bit 3: MATn, Match control.
0: Disable the digital comparator match event to set CCFn.
1: A match of the PCA counter with this module’s compare/capture register causes the CCFn bit in CCON to be set.

Bit 2: TOGn, Toggle control.
0: Disable the digital comparator match event to toggle CEXn.
1: A match of the PCA counter with this module’s compare/capture register causes the CEXn pin to toggle.

Bit 1: PWMn, PWM control.
0: Disable the PWM mode in PCA module.
1: Enable the PWM function and cause CEXn pin to be used as a pulse width modulated output.

Bit 0: ECCFn, Enable CCFn interrupt.
0: Disable compare/capture flag CCFn in the CCON register to generate an interrupt.
1: Enable compare/capture flag CCFn in the CCON register to generate an interrupt.

Note: The bits CAPNn (CCAPMn.4) and CAPPn (CCAPMn.5) determine the edge on which a capture input will be active. If both bits are set, both edges will be enabled and a capture will occur for either transition.

Each module also has a pair of 8-bit compare/capture registers (CCAPnH, CCAPnL) associated with it. These registers are used to store the time when a capture event occurred or when a compare event should occur.
When a module is used in the PWM mode, in addition to the above two registers, an extended register PCAPWMn is used to improve the range of the duty cycle of the output. The improved range of the duty cycle starts from 0%, up to 100%, with a step of 1/256. About 10/12/16 bit PWM please reference 16.4.6 and 16.4.7.

CCAPnH: PCA Module n Capture High Register, n=0~3
SFR Page = 0 only for n= 0~1 (n=2~*3* for all page)
SFR Address = 0xFA~0xFD RESET = 0000-0000

7	6	5	4	3	2	1	0
CCAPnH.7	CCAPnH.6	CCAPnH.5	CCAPnH.4	CCAPnH.3	CCAPnH.2	CCAPnH.1	CCAPnH.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

CCAPnL: PCA Module n Capture Low Register, n=0~3
SFR Page = 0 only for n= 0~1 (n=2~*3* for all page)
SFR Address = 0xEA~0xED RESET = 0000-0000

7	6	5	4	3	2	1	0
CCAPnL.7	CCAPnL.6	CCAPnL.5	CCAPnL.4	CCAPnL.3	CCAPnL.2	CCAPnL.1	CCAPnL.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

PCAPWMn: PWM Mode Auxiliary Register, n=0~3
SFR Page = 0 only for n= 0~1 (n=2~*3* for all page)
SFR Address = 0xF2~0xF5 RESET = 0000-0000

7	6	5	4	3	2	1	0
PnRS1	PnRS0	0	0	0	PnINV	ECAPnH	ECAPnL
R/W	**R/W**	**W**	**W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: PnRS1~0, PWMn Resolution Setting 1~0.
00: 8 bit PWMn, the overflow is active when [CH, CL] counts XXXX-XXXX-1111-1111 → XXXX-XXXX-0000-0000.
01: 10 bit PWMn, the overflow is active when [CH, CL] counts XXXX-XX11-1111-1111 → XXXX-XX00-0000-0000.
10: 12 bit PWMn, the overflow is active when [CH, CL] counts XXXX-1111-1111-111 → XXXX-0000-0000-0000.
11: 16 bit PWMn, the overflow is active when [CH, CL] counts 1111-1111-1111-1111→ 0000-0000-0000-0000.

Bit 5~4: Reserved. Software must write “0” on these bits when PCAPWMn is written.

Bit 2: PnINV, Invert Compare/PWM output (C0PnOR) on CEXn pin.
0: Non-inverted Compare/PWM output (C0PnOR).
1: Inverted Compare/PWM output (C0PnOR).

Bit 1: ECAPnH, Extended 9th bit (MSB bit), associated with CCAPnH to become a 9-bit register used in PWM mode.

Bit 0: ECAPnL, Extended 9th bit (MSB bit), associated with CCAPnL to become a 9-bit register used in PWM mode.

16.4. Operation Modes of the PCA

Table 16–1 shows the CCAPMn register settings for the various PCA functions.

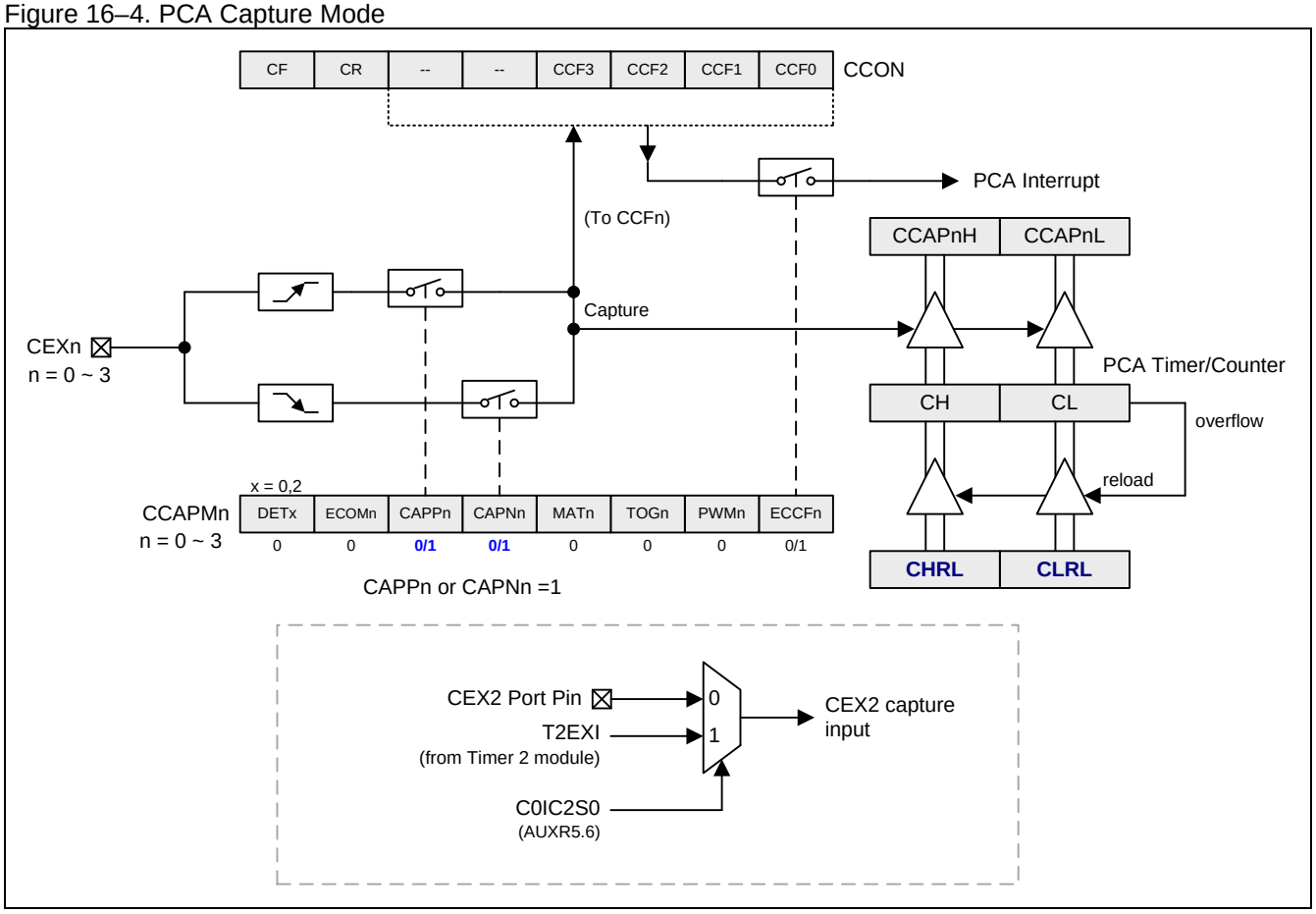
Table 16–1. PCA Module Modes

ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	Module Function
0	0	0	0	0	0	0	No operation
X	1	0	0	0	0	X	16-bit capture by a positive-edge trigger on CEXn
X	0	1	0	0	0	X	16-bit capture by a negative-edge trigger on CEXn
X	1	1	0	0	0	X	16-bit capture by a transition on CEXn
1	0	0	1	0	0	X	16-bit Software Timer (Compare)
1	0	0	1	1	0	X	16-bit High Speed Output (HSO)
1	0	0	0/1	0	1	X	Pulse Width Modulator (PWM)
1	0	0	0	1	1	X	Compare Output on PWM match case (COPM)
1	0	1	0	0	1	X	FIFO Data Mode

16.4.1. Capture Mode

To use one of the PCA modules in the capture mode, either one or both of the bits CAPN and CAPP for that module must be set. The external CEX input for the module is sampled for a transition. When a valid transition occurs the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn and the ECCFn bits for the module are both set, an interrupt will be generated.

Figure 16–4. PCA Capture Mode



16.4.2. Buffered Capture Mode

To capture narrow input signal, buffered capture mode is necessary. If enabled, it put the odd module capture data registers (CCAPnH, CCAPnL, n= 1, 3) to be the buffer register of even module capture data registers (channel 0, 2). There is no influence on module 0/2 capture operation. BME0 enables the buffer operation of channel 0 and channel 1. BME2 control the module 2/3.

Figure 16–5. PCA Buffered Capture Mode (BMEn=1, n= 0, 2)

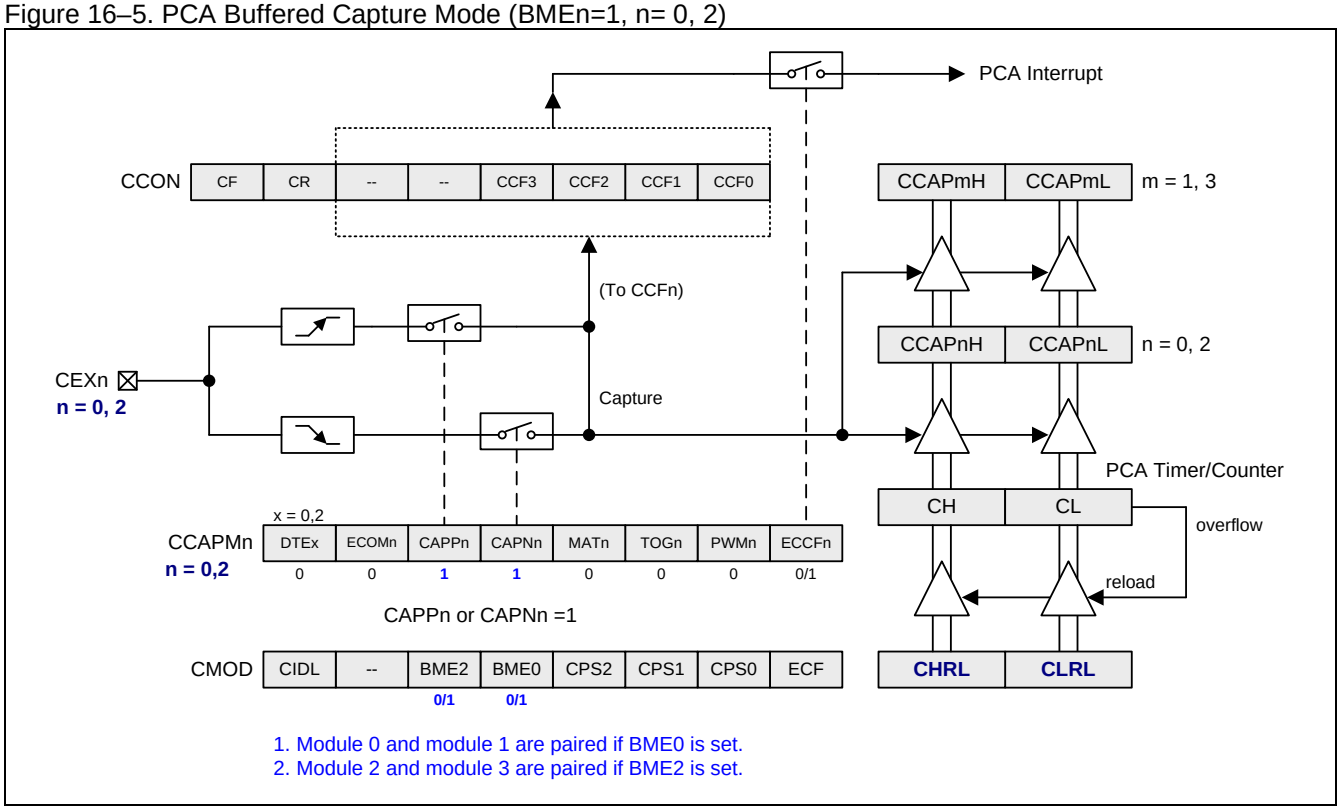
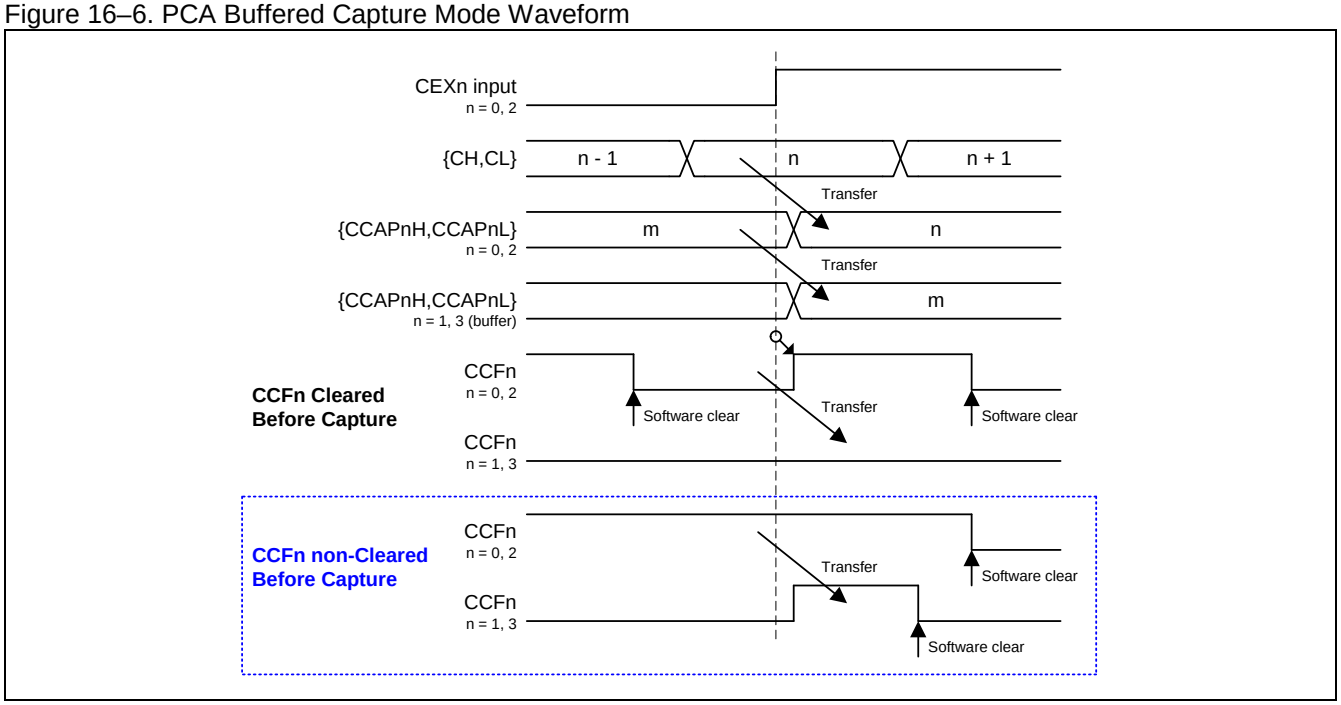


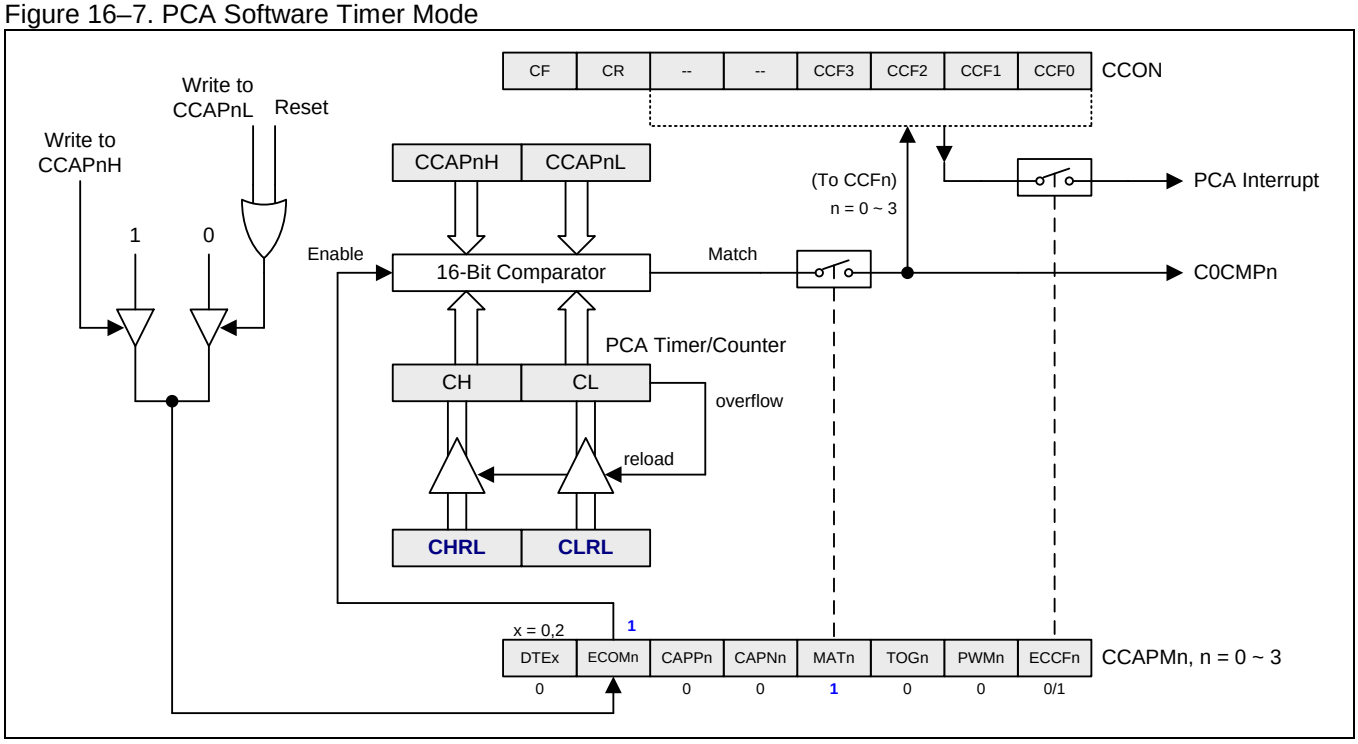
Figure 16–6. PCA Buffered Capture Mode Waveform



16.4.3. 16-bit Software Timer Mode (Compare mode)

The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the module's CCAPMn register. The PCA timer will be compared to the module's capture registers, and when a match occurs an interrupt will occur if the CCFn and the ECCFn bits for the module are both set.

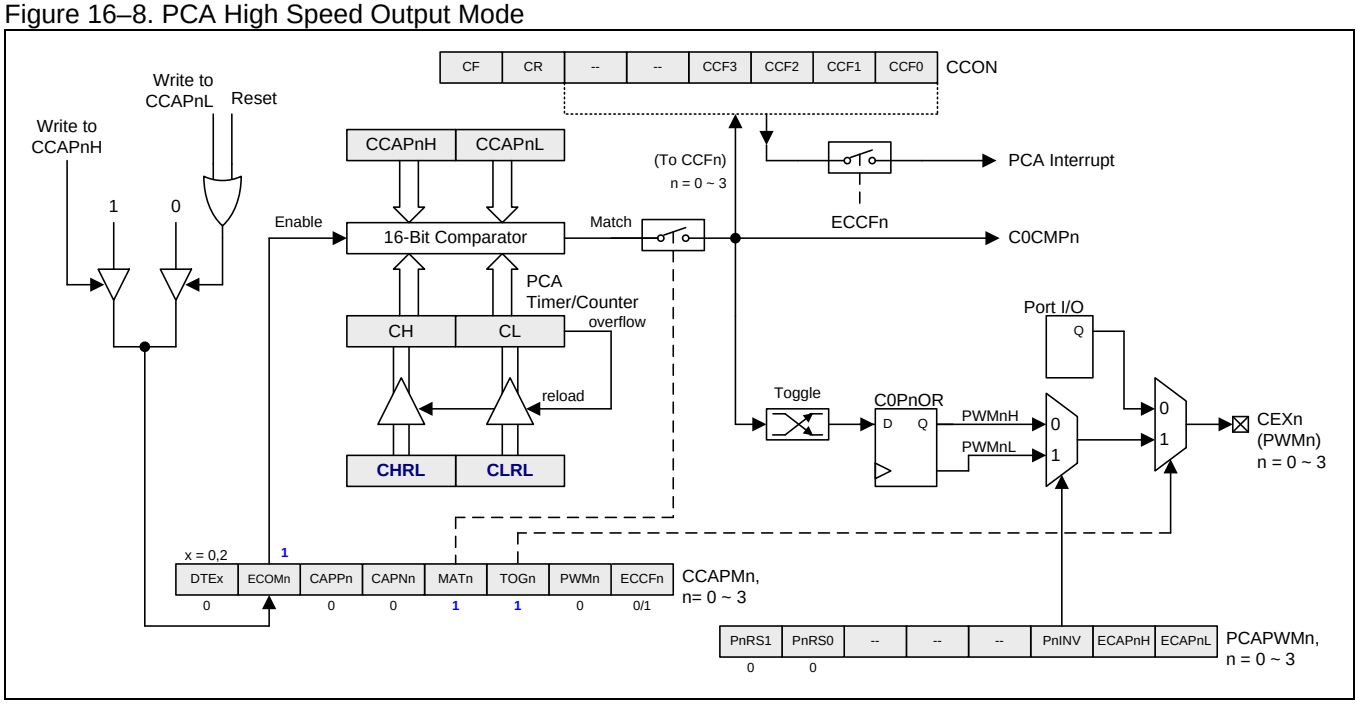
Figure 16–7. PCA Software Timer Mode



16.4.4. High Speed Output Mode (Compare Output mode)

In this mode the CEX output associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode, the TOG, MAT and ECOM bits in the module's CCAPMn register must be set.

Figure 16–8. PCA High Speed Output Mode



16.4.5. Buffered 8-bit PWM Mode

All of the PCA modules can be used as PWM outputs. The frequency of the output depends on the clock source for the PCA timer. All of the modules will have the same frequency of output because they all share the PCA timer.

The duty cycle of each module is determined by the module's capture register CCAPnL and the extended 9th bit, ECAPnL. When the 9-bit value of { 0, [CL] } is *less than* the 9-bit value of { ECAPnL, [CCAPnL] } the output will be low, and if *equal to or greater than* the output will be high.

When CL overflows from 0xFF to 0x00, { ECAPnL, [CCAPnL] } is reloaded with the value of { ECAPnH, [CCAPnH] }. This allows updating the PWM without glitches. The PWMn and ECOMn bits in the module's CCAPMn register must be set to enable the PWM mode.

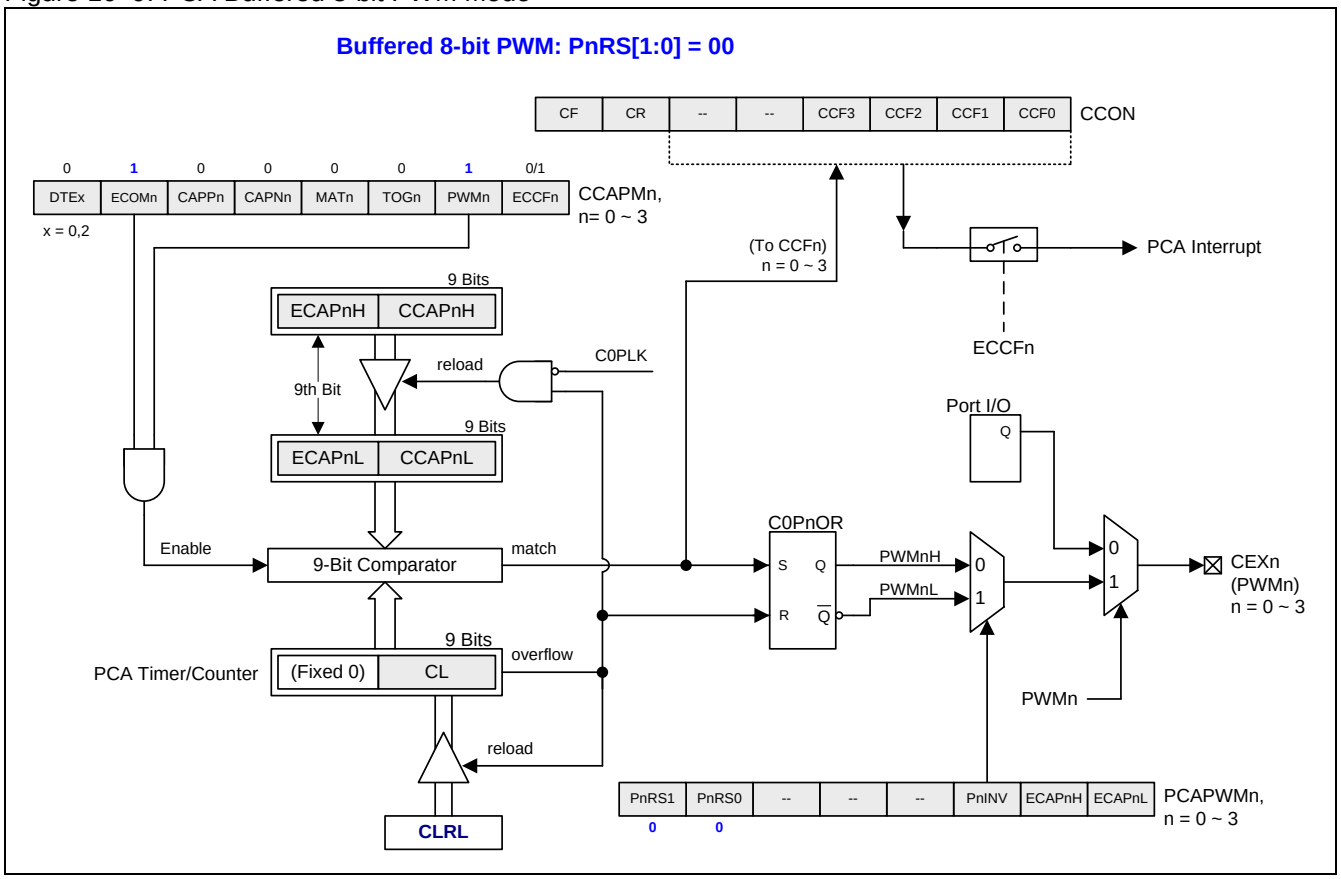
Using the 9-bit comparison, the duty cycle of the output can be improved to really start from 0%, and up to 100%. The formula for the duty cycle is:

Duty Cycle = 1 – { ECAPnH, [CCAPnH] } / 256.

Where, *[CCAPnH]* is the 8-bit value of the CCAPnH register, and *ECAPnH* (bit-1 in the PCAPWMn register) is 1-bit value. So, *{ ECAPnH, [CCAPnH] }* forms a 9-bit value for the 9-bit comparator.

For examples,
a. If ECAPnH=0 & CCAPnH=0x00 (i.e., 0x000), the duty cycle is 100%.
b. If ECAPnH=0 & CCAPnH=0x40 (i.e., 0x040) the duty cycle is 75%.
c. If ECAPnH=0 & CCAPnH=0xC0 (i.e., 0x0C0), the duty cycle is 25%.
d. If ECAPnH=1 & CCAPnH=0x00 (i.e., 0x100), the duty cycle is 0%.

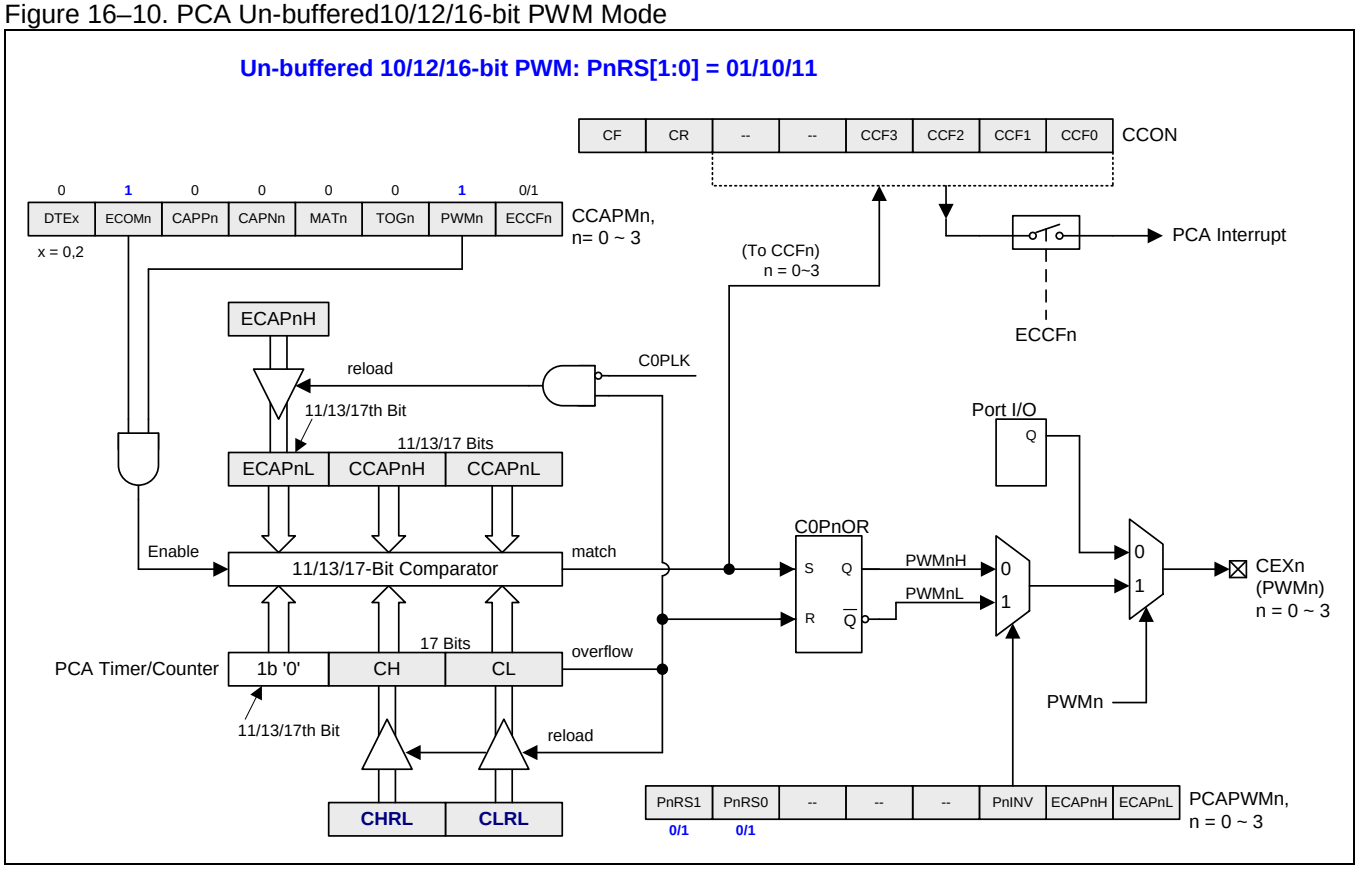
Figure 16–9. PCA Buffered 8-bit PWM Mode



16.4.6. Un-buffered 10/12/16-bit PWM Mode

The PCA provides the variable PWM mode to enhance the control capability on PWM application. There are additional un-buffered 10/12/16 bits PWM can be assigned in each channel and each PWM channel with different resolution can operate concurrently.

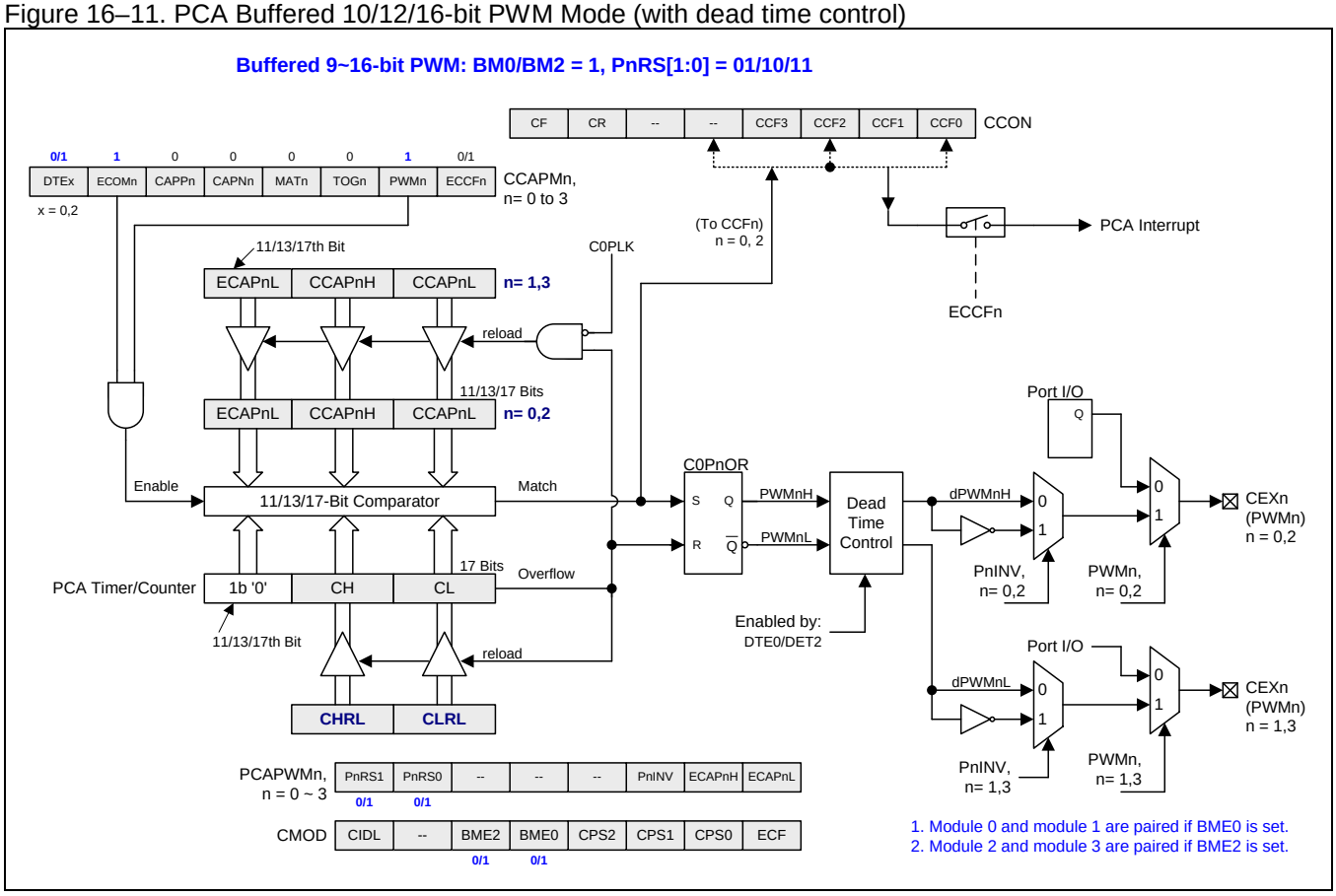
Figure 16–10. PCA Un-buffered10/12/16-bit PWM Mode



16.4.7. Buffered 10/12/16-bit PWM Mode

To use 10/12/16-bit PWM mode might cause unexpected duty cycle when change the duty cycle setting by writing data into CCAPnH and CCAPnL, because the 8 bit CPU can only write one byte at a time. To finish fully setting it will take two write cycles, and the comparator will output unexpected duty cycle when the first byte have been written. If the applications need accurate control when change the duty cycle, it needs to use the Buffered PWM mode.

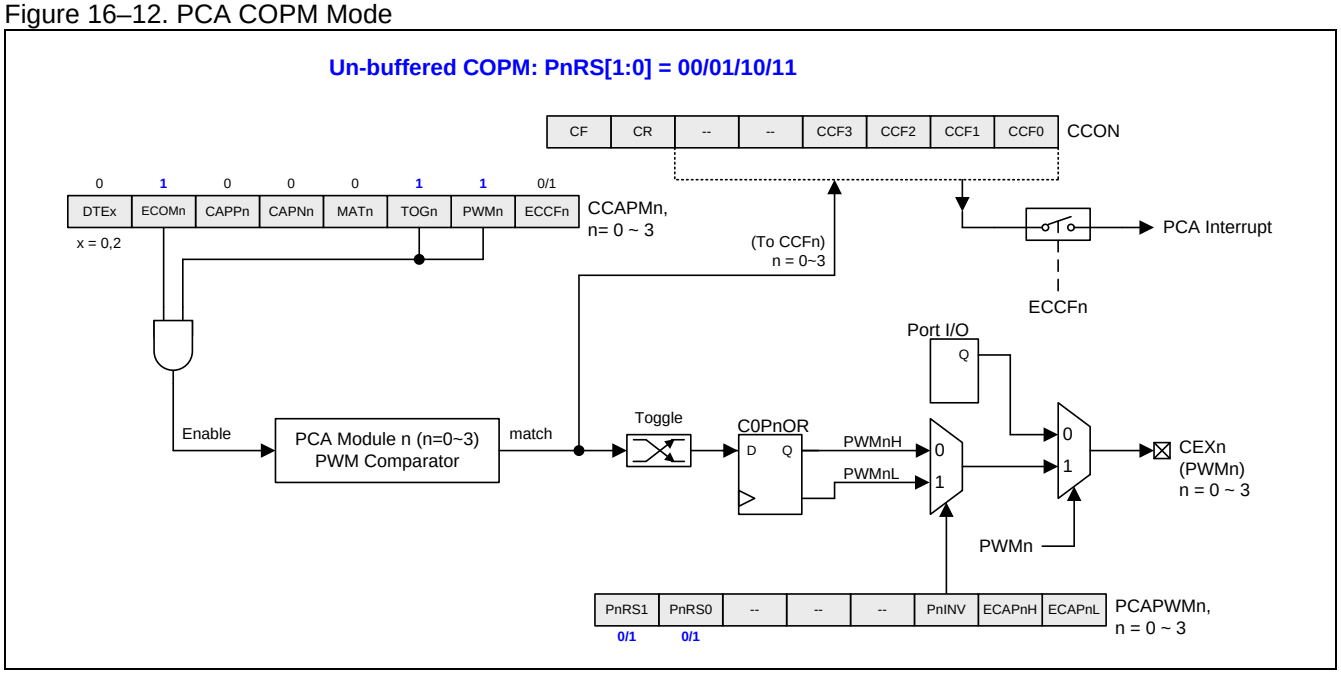
Figure 16–11. PCA Buffered 10/12/16-bit PWM Mode (with dead time control)



16.4.8. COPM Mode

Compare Output on PWM Match mode is similar to High Speed Output Mode, but it uses PCA0 PWM comparators instead of fixed 16-bit comparators. It gives more flexibility to the applications. For example, if it uses 8-Bit PWM for the PCA0 comparator, the output toggles frequency can higher than High Speed Output Mode.

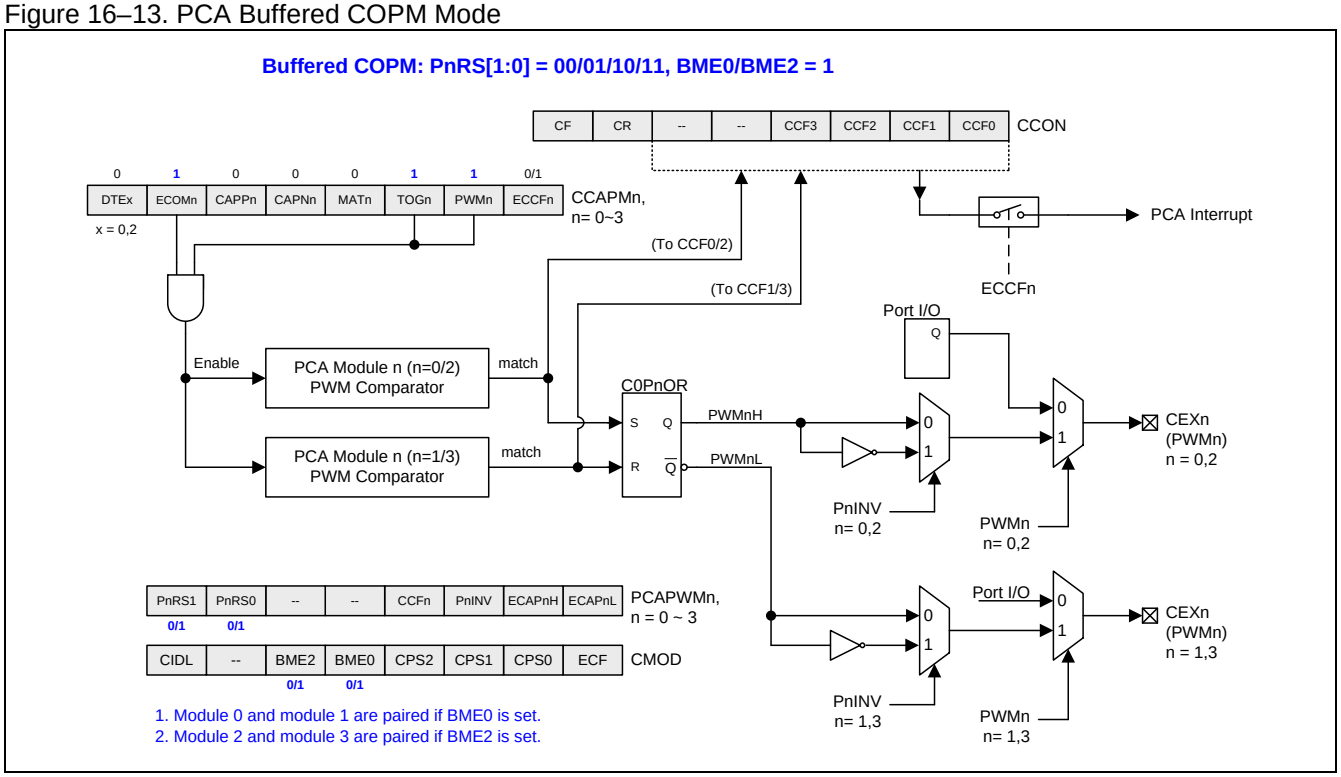
Figure 16–12. PCA COPM Mode



16.4.9. Buffered COPM Mode

If the applications need to have any phase control of the PWM signals, it needs to set the PCA0 modules in buffered COPM mode. One pair of the PCA0 module (n=0&1 / 2&3) can program the time delay of the two edges of one cycle of the PWM signal. It means you can set the start and end point of the waveform. This is useful when the 2 or 3 correlation PWM signals can set the phase shift between each other.

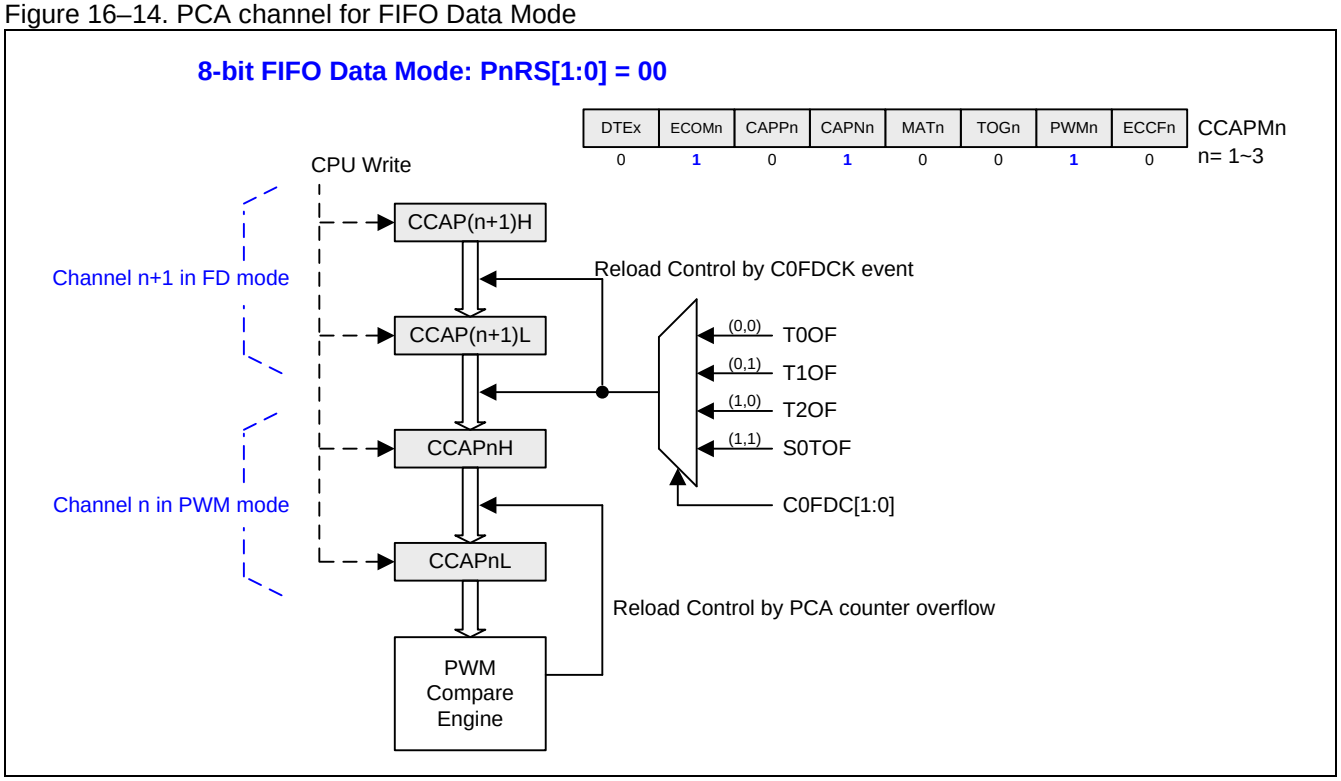
Figure 16–13. PCA Buffered COPM Mode



16.4.10. FIFO Data Mode

In this mode the user can set the CCAPnL, CCAPnH, CCAP(n+1)L and CCAP(n+1)H as a buffer chain. After all these buffers are set, it can change the duty sequentially trigger by T0OF, T1OF, T2OF or S0TOF. This function is enabled, the CPU can leave it to run by itself to earn more time slot to do other operations. For example, when the power converter start to raise the voltage from light load to heave load, it might useful to set the duty larger than the target it the beginning period, and then reduce the duty step by step close to the target duty. It can just set all duties in the buffer and leave it to finish.

Figure 16–14. PCA channel for FIFO Data Mode



Channel FIFO data mode that is moved on C0FDCK.
C0FDCK source selection, updated clock selection of PCA0 FIFO Data mode.

C0FDC1~0	C0FDCK
00	T0OF
01	T1OF
10	T2OF
11	S0TOF

AUXR9: Auxiliary Register 9
SFR Page = **6 only**
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	SIDPS0	T1G0	T0G1	C0FDC1	C0FDC0	0	0
W	**W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

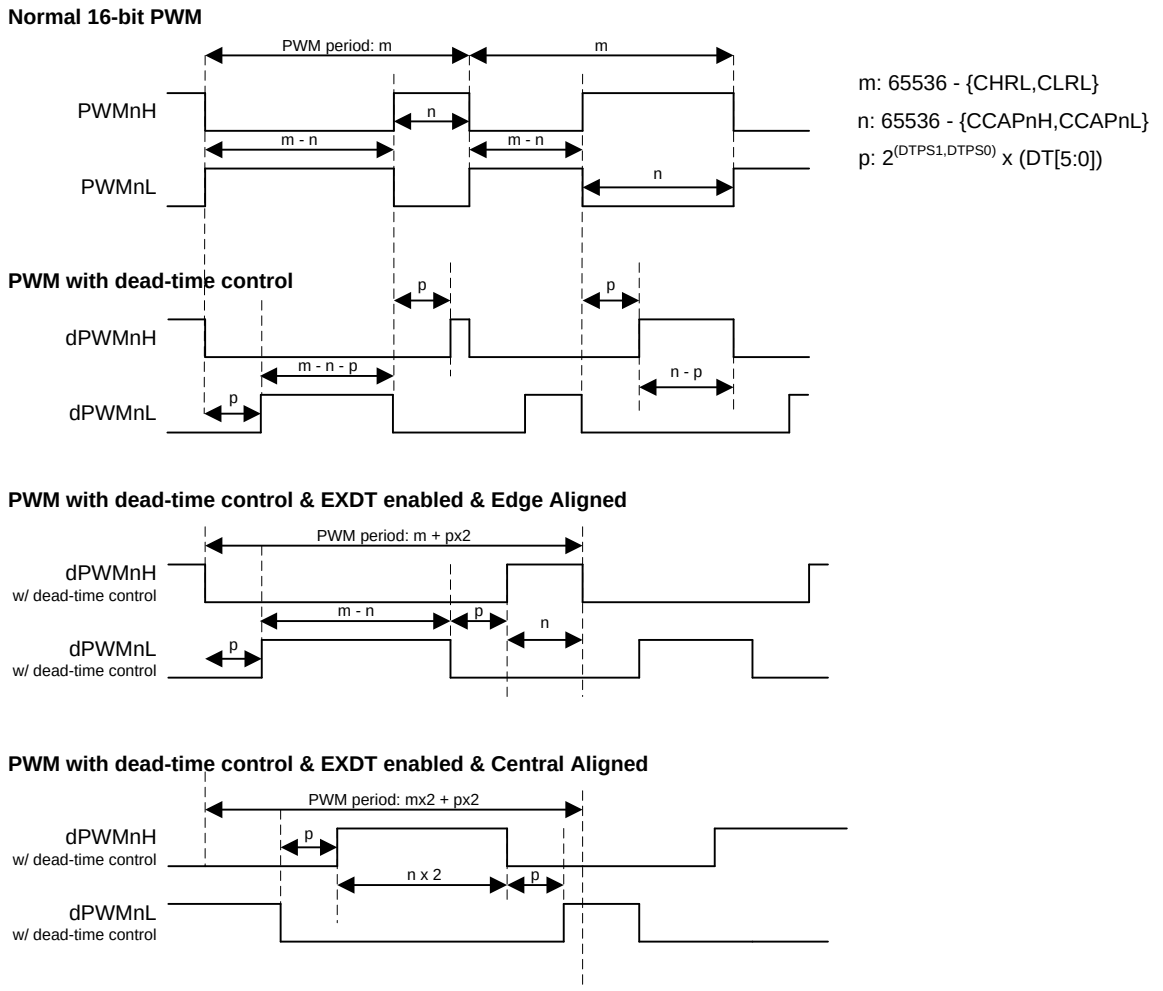
Bit 3~2: C0FDC1~0, C0FDCK Selection [1:0].

C0FDC1~0	C0FDCK
00	T0OF
01	T1OF
10	T2OF
11	S0TOF

16.4.11. Enhanced PWM Control

The PCA provides the variable PWM mode to enhance the control capability on PWM application. There are additional 10/12/16 bits PWM can be assigned in each channel and each PWM channel with different resolution and different phase delay can operate concurrently.

Figure 16–15. PWM Waveform with Dead-Time Control



CCAPMn: PCA Module Compare/Capture Register, n=0~3

SFR Page = 0 only for n= 0~1 (n=2~3 for all page)

SFR Address = 0xDA~0xDD RESET = 0000-0000

7	6	5	4	3	2	1	0
DTEn	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: DTEn. Enable Dead-Time control on PWMHn/PWMLn output pair. This bit is only valid on n= 0, 2 and the dead-time function is active when PWM channel is operating in buffer mode. The channel buffer mode is enabled by BME0, BME2 in CMOD.
0: Disable the Dead-Time control on PWMn output.
1: Enable the Dead-Time control on PWMn output.

PDTCRA: PWM Dead-Time Control Register -A

SFR Page = **1 only**

SFR Address = **0xBC** RESET = 0000-0000

7	6	5	4	3	2	1	0
DTPS1	**DTPS0**	**DT5**	**DT4**	**DT3**	**DT2**	**DT1**	**DT0**
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: DTPS1~0, Clock Pre-Scaler of Dead-Time counter.

DTPS[1:0]	Pre-Scaler Selection
00	SYSCLK
01	SYSCLK/2
10	SYSCLK/4
11	SYSCLK/8

Bit 5~0: DT5~0, Dead-Time period control bits.

DT[5:0]	Dead-Time Period
000000	Dead-Time Disabled
000001	Pre-Scaler Clock X 1
000010	Pre-Scaler Clock X 2
000011	Pre-Scaler Clock X 3
…..	……
111110	Pre-Scaler Clock X 62
111111	Pre-Scaler Clock X 63

PWMCR: PWM Control Register

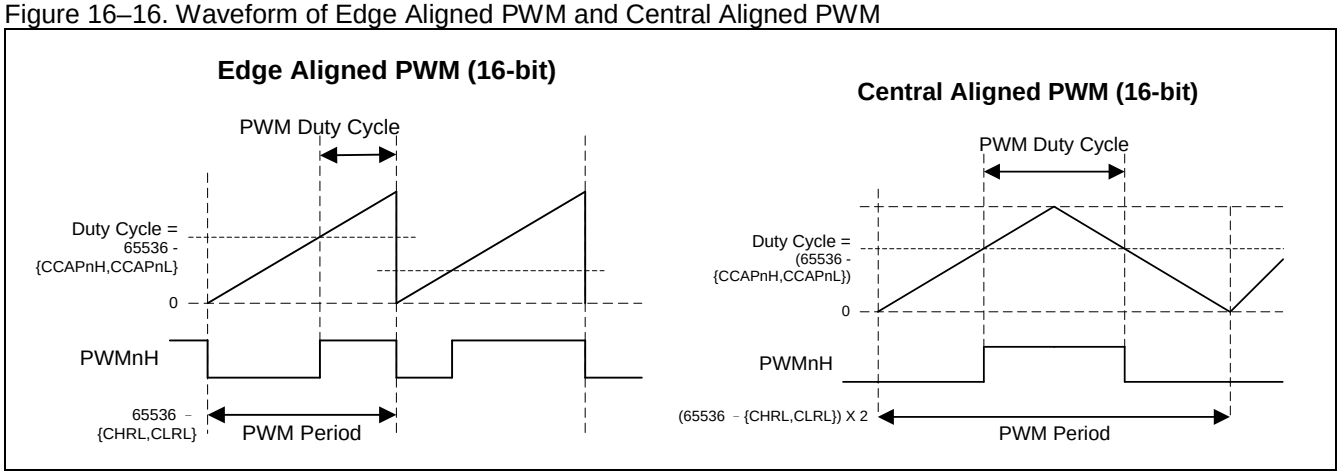
SFR Page = **0 only**

SFR Address = **0xBC** RESET = 0000-0000

7	6	5	4	3	2	1	0
PCAE	**EXDT**	**PBKM**	**PBKE1.1**	**PBKE1.0**	**PBKE0.2**	**PBKE0.1**	**PBKE0.0**
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: PCAE, PWM Central Aligned Enabled. PCAE controls the enabled PWM channels to central aligned modulation including buffer mode PWM or non-buffer mode PWM. In this PWM mode, the PWM frequency is the half of edge aligned mode. This function is only active on PWMO0~3.
0: Set the PWM function with edge aligned modulation.
1: Enable the PWM function with central aligned modulation. It only supports 8/10/12/16-bit resolution on CHRL and CLRL setting.

Figure 16–16. Waveform of Edge Aligned PWM and Central Aligned PWM



Bit 6: EXDT: Extend Dead-Time in PWM Period. This function will corrupt the non-PWM channel function. Such as capture mode, software timer mode and high speed output mode.
0: Disable M + **2P**.
1: Enable M + **2P** on enabled PWM channel.

Bit 5: PBKM, PWM Break Mode selection.
0: Latched Mode.
1: Cycle-by-cycle Mode.

Figure 16–17. Latch Mode Waveform of PWM Break control

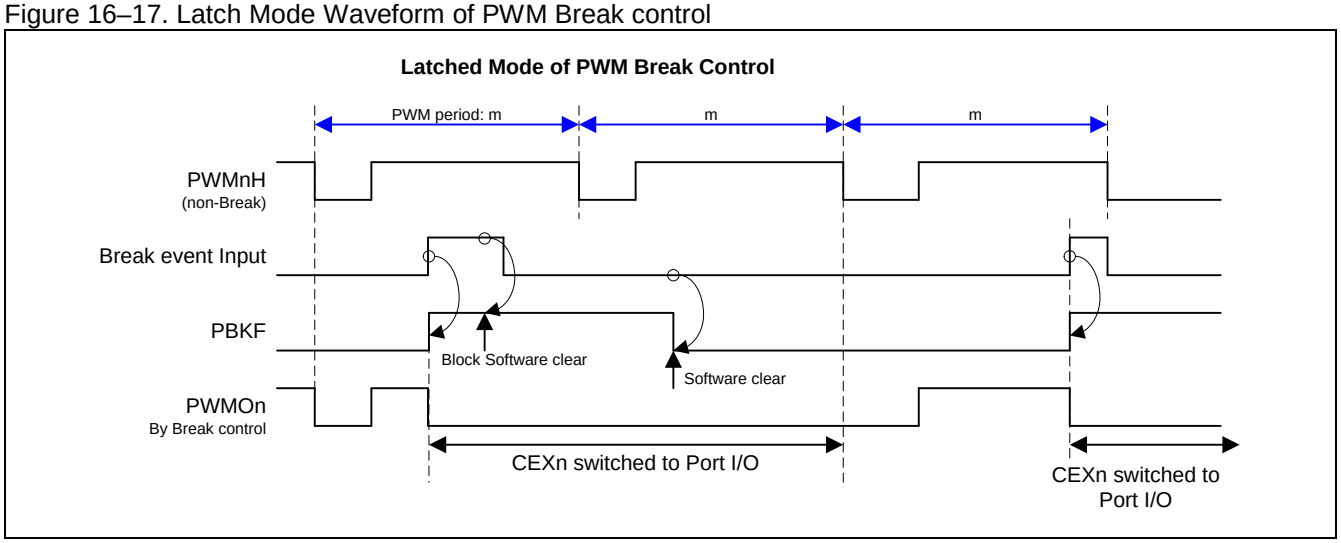
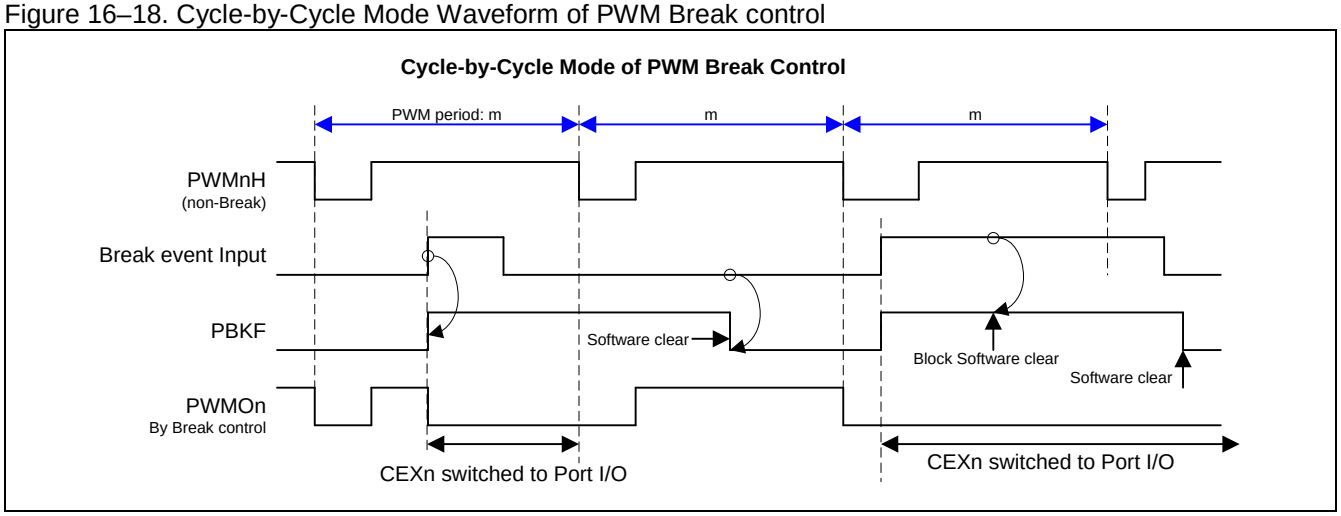


Figure 16–18. Cycle-by-Cycle Mode Waveform of PWM Break control



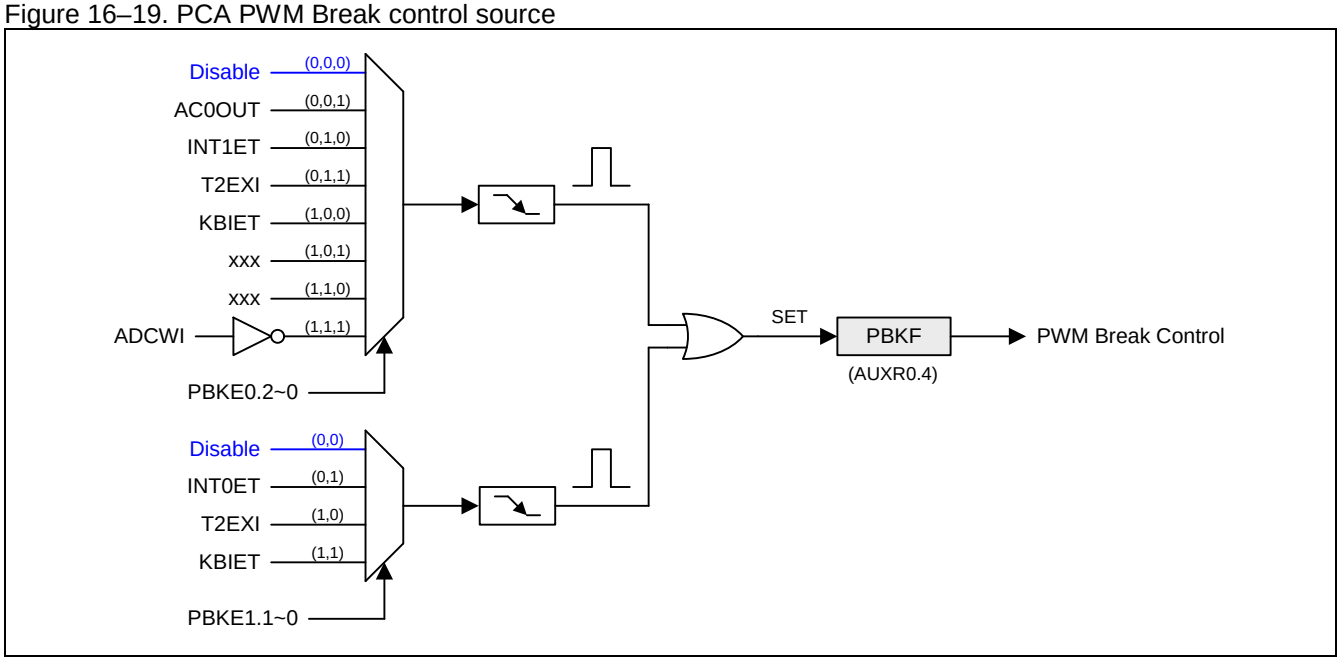
Bit 4~3: PBKE1.1~0, PWM Break Enable 1 selection. This function is only active on CEXn output mode (n=0~3).

PBKE1[1:0]	PWM Break Source
0 0	Disable PWM break source 1
0 1	INT0ET
1 0	T2EXI
1 1	KBIET, KBI match active

Bit 2~0: PBKE0.2~0, PWM Break Enable 0 selection. This function is only active on CEXn output mode (n=0~3).

PBKE0[2:0]	PWM Break Source
0 0 0	Disable PWM break source 0
0 0 1	AC0OUT
0 1 0	INT1ET, nINT1 active
0 1 1	T2EXI
1 0 0	KBIET, KBI match active
1 0 1	Reserved
1 1 0	Reserved
1 1 1	ADCWI active

Figure 16–19. PCA PWM Break control source



AUXR0: Auxiliary Register 0

SFR Page = 0~F

SFR Address = **0xA1** RESET = 0000-0000

7	6	5	4	3	2	1	0
P45OC1	P45OC0	0	PBKF	0	0	INT1H	INT0H
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

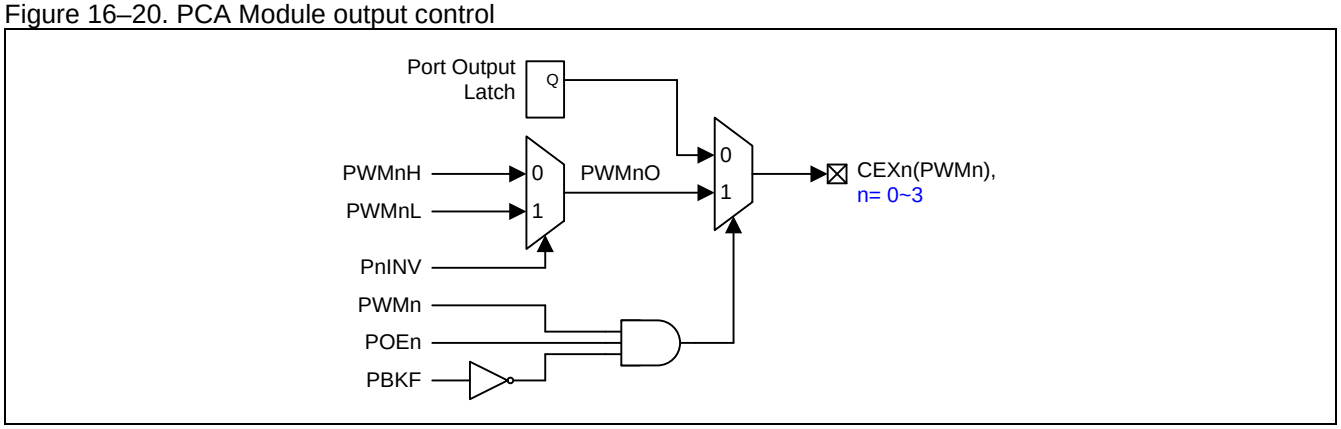
Bit 4: PBKF, PWM Break Flag. This bit is set by PWM break source enabled. If this flag is set, the enabled PWM channel 0~3 will be blocked and the output pins keep the original GPIO state.
0: There is no PWM Break event happened. It is only cleared by software.
1: There is a PWM Break event happened or software triggers a PWM Break.

16.4.12. PCA Module Output Control

PCA0 modules have multi output control mode can be selected for different applications. The CEXn (n=0~3) can be programed as general I/O port or the output of the PCA0 module 0~ 3. When PWM has been assigned to the CEXn, the PnINV can switch between the normal PWM signal or inverted PWM signal. POEn can be used to enable or disable the PWM output to the port pin.

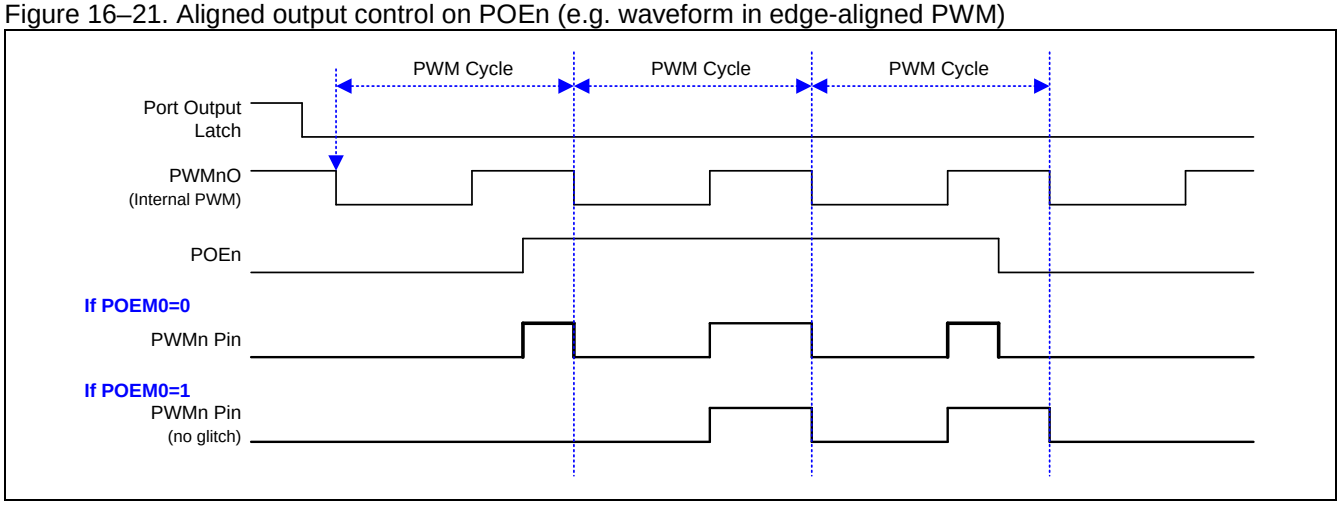
The CEXn (n=0~3) also can use PBKF, PWM Break Flag, to break PWM output. If this flag is set, the enabled PWM channel 0~3 will be blocked and the output pins keep the original GPIO state.

Figure 16–20. PCA Module output control



POEM0 in **MG82F6B08 / 6B001/ 6B104** controls the POEn output timing to align with PWM cycle. The configuration and waveform of the aligned function is shown in Figure 16–21.

Figure 16–21. Aligned output control on POEn (e.g. waveform in edge-aligned PWM)



PAOE: PWM Additional Output Enable Register

SFR Page = **0~F**

SFR Address = **0xF1** RESET = **1001-1001**

7	6	5	4	3	2	1	0
POE3	0	0	POE2	POE1	0	0	POE0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: POE3, PCA0 PWM3 main channel (PWM3O) output control.
0: Disable PWM3O output on port pin.
1: Enable PWM3O output on port pin. **Default is enabled.**

Bit 4: POE2, PCA0 PWM2 main channel (PWM2O) output control.
0: Disable PWM2O output on port pin.
1: Enable PWM2O output on port pin. **Default is enabled.**

Bit 3: POE1, PCA0 PWM1 main channel (PWM1O) output control.
0: Disable PWM1O output on port pin.
1: Enable PWM1O output on port pin. **Default is enabled.**

Bit 0: POE0, PCA0 PWM0 main channel (PWM0O) output control.
0: Disable PWM0O output on port pin.
1: Enable PWM0O output on port pin. **Default is enabled.**

AUXR2: Auxiliary Register 2

SFR Page = 0~F

SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: C0PLK, PCA0 buffered PWM/COPM update control.
0: Buffered PWM/COPM is auto-updated on PCA0 base timer overflow.
1: Disable the buffered PWM/COPM auto-updated.

AUXR7: Auxiliary Register 7

SFR Page = 4 Only

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	1	C0CKOE	SPI0M0	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 5: C0CKOE, PCA0 clock output enable.
0: Disable PCA0 clock output.
1: Enable PCA0 clock output with PCA0 base timer overflow rate/2.

AUXR11: Auxiliary Register 11

SFR Page = **8 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
P30AM	P33AM	0	0	0	POEM0	C0M0	C0OFS
R/W	**R/W**	**R/W**	**R/W**	**W**	**R/W**	**R/W**	**R/W**

Bit 2: POEM0, PCA0 POEn control 0.
0: POEn function is active immediately after CPU writing.
1: POEn function is aligned to PWM cycle.

AUXR5: Auxiliary Register 5

SFR Page = **2 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	C0IC2S0	0	0	C0PS1	C0PS0	ECIPS0	C0COPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 6: C0IC2S0, PCA0 Input Channel 2 input port pin Selection.

C1IC2S0	CEX2 input
0	CEX2 Port Pin
1	T2EXI

Bit 3~2: C0PS[1:0], PCA0 Port pin Selection 1 & 0.

C0PS1~0	CEX0	CEX1	CEX2	CEX3
0 0	P3.0	P3.3	P3.1	P4.6
0 1	P4.4	P3.3	P4.5	P4.6
1 0	P4.4	P3.3	P3.0	P4.6
1 1	P4.4	P3.3	P1.0	P4.6

Bit 1: ECIPS0, PCA0 ECI Port pin Selection0.

ECIPS0	ECI
0	P4.4
1	P1.0

Bit 0: C0COPS, PCA0 Clock Output (C0CKO) port pin Selection.

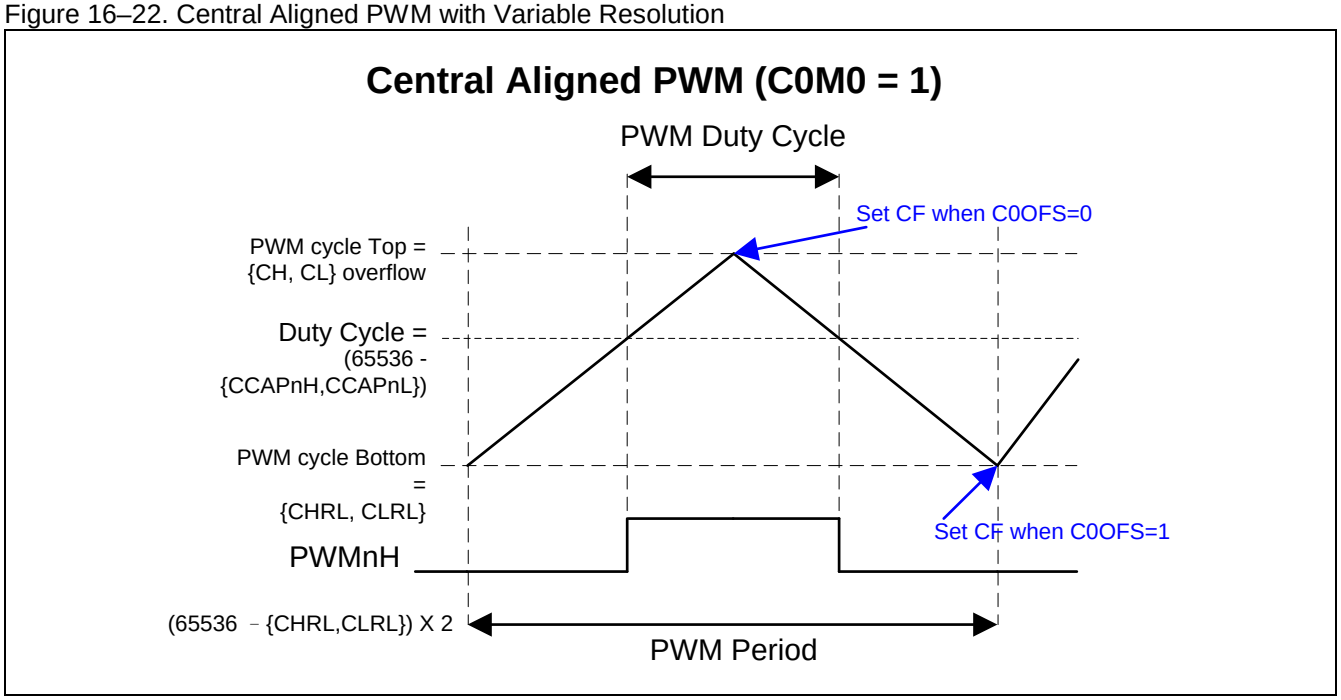
C0COPS	C0CKO
0	P4.7
1	P3.3

16.4.13. Variable Resolution on Central Aligned PWM

In Section “Enhanced PWM Control”, it defines the central aligned PWM only support the 8/10/12/16-bit resolution. And in that mode, all of PCA functions, capture or compare, on other non-PWM modules are still available.

If it is necessary to apply the variable resolution on central aligned PWM, software must set C0M0 to enable the PCA0 to support this function operating. In this mode, PCA0 can support all compare or PWM modes. Otherwise, not support. Please note when using Central Aligned PWM with C0M0 = 1, Please note when using Central Aligned PWM with C0M0 = 1, we suggest to set the PWM module under 16-bit mode and the base timer need to use 16-bit 0xFFFF to minus the value to prevent unexpected error.

Figure 16–22. Central Aligned PWM with Variable Resolution



AUXR11: Auxiliary Register 11

SFR Page = **8 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
P30AM	P33AM	0	0	0	POEM0	C0M0	C0OFS
R/W	**R/W**	**R/W**	**R/W**	**W**	**R/W**	**R/W**	**R/W**

Bit 1: C0M0, PCA0 Mode control 0.
0: Not support variable resolution on central aligned PWM.
1: Enable PCA0 variable resolution central aligned PWM. To enable this function, the PCAE also needs to be set.

Bit 0: C0OFS, PCA0 overflow flag selection when C0M0 is enabled.
0: CF is set on the top of central aligned PWM cycle.
1: CF is set on the bottom of central aligned PWM cycle.

17. Serial Port 0 (UART0)

The serial port 0 of **MG82F6B08 / 6B001/ 6B104** supports full-duplex transmission, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the register. However, if the first byte still hasn't been read by the time reception of the second byte is complete, one of the bytes will be lost. The serial port receive and transmit registers are both accessed at special function register S0BUF. Writing to S0BUF loads the transmit register, and reading from S0BUF accesses a physically separate receive register.

17.1. Serial Port 0 Mode Selection

The serial port can operate in **5** standard modes and **8** enhance modes: Mode 0 provides *synchronous* communication while Modes 1, 2, and 3 provide *asynchronous* communication. The asynchronous communication operates as a full-duplex Universal Asynchronous Receiver and Transmitter (UART), which can transmit and receive simultaneously and at different baud rates. Mode 4 in UART0 supports SPI master operation which data rate setting is same as Mode 0. For the enhance modes please reference 17.11 Serial Port 0 Enhance function.

Table 17–1. Serial Port 0 Mode Selection

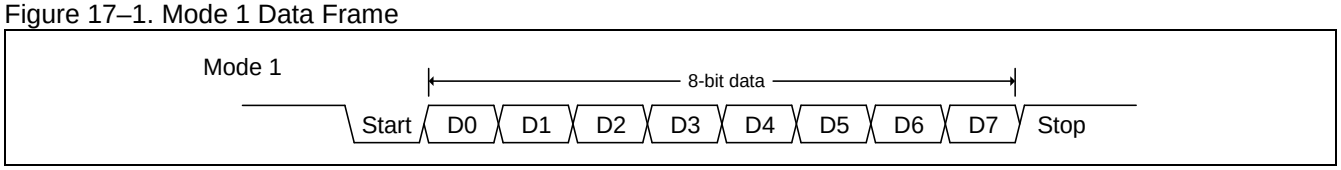
SM30,SM00, SM10	S0RCK	S0TCK	MODE	Function	Baud Rate Time Base	Note
000	0	0	0	shift register	SYSCLK/12 or SYSCLK/4 (URM0X3=1)	
001	0	0	1	8-bit UART	Timer 1 or Timer 2 overflow	When SMOD1 & SMOD2 =1，counter cannot be Full-1 or Full-2 (e.g. 254, 255, 65534, 65535)
010	0	0	2	9-bit UART	SYSCLK/64, /32, /16, or /8	
011	0	0	3	9-bit UART	Timer 1 or Timer 2 overflow	When SMOD1 & SMOD2 =1，counter cannot be Full-1 or Full-2 (e.g. 254, 255, 65534, 65535)
100	0	0	4	SPI Master	SYSCLK/12 or SYSCLK/4 (URM0X3=1)	
000	0	1	Enhanced	shift register	S0BRG overflow	S0BRT cannot be 255
001	0/1	0/1	Enhanced	8-bit UART	Selectable S0BRG overflow on TX or RX	SMOD1 & SMOD2 cannot be 1 at the same time
010	0	1	Enhanced	9-bit UART	TX: S0BRG overflow RX: SYSCLK/64, /32 or /16	SMOD1 & SMOD2 cannot be 1 at the same time
010	1	0	Enhanced	9-bit UART	TX: SYSCLK/64, /32 or /16 RX: S0BRG overflow	SMOD1 & SMOD2 cannot be 1 at the same time
010	1	1	Enhanced	Pure Timer	Only Timer function	
011	0/1	0/1	Enhanced	9-bit UART	Selectable S0BRG overflow on TX or RX	SMOD1 & SMOD2 cannot be 1 at the same time
100	0	1	Enhanced	SPI Master	S0BRG overflow	S0BRT cannot be 255
101	1	1	Enhanced	LIN Bus	S0BRG overflow and auto baud rate	SMOD1 & SMOD2 cannot be 1 at the same time
Others				Reserved.		

Note: Mode 0 ~ 4 use the default value of S0RCK and S0TCK, for the reset enhance mode please reference 17.11 Serial Port 0 Enhance function for detail description.

Mode 0: 8 data bits (LSB first) are transmitted or received through RXD0. TXD0 always outputs the shift clock. The baud rate can be selected to 1/12 or 1/4 the system clock frequency by URM0X3 setting in S0CFG register. In **MG82F6B08 / 6B001/ 6B104**, the clock polarity of serial port Mode 0 can be selected by software. It is decided by P3.1 state before serial data shift in or shift out. Figure 17–4 and Figure 17–5 show the clock polarity waveform in Mode 0.

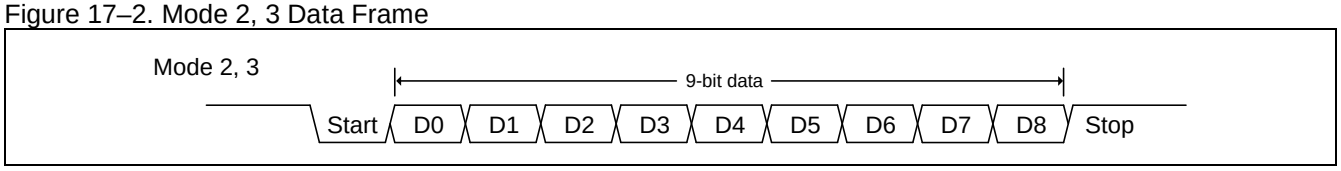
Mode 1: 10 bits are transmitted through TXD0 or received through RXD0. The frame data includes a start bit (0), 8 data bits (LSB first), and a stop bit (1), as shown in Figure 17–1. On receive, the stop bit would be loaded into RB80 in S0CON register. The baud rate is variable.

Figure 17–1. Mode 1 Data Frame



Mode 2: 11 bits are transmitted through TXD0 or received through RXD0. The frame data includes a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1), as shown in Figure 17–2. On Transmit, the 9th data bit comes from TB80 in S0CON register can be assigned the value of 0 or 1. On receive, the 9th data bit would be loaded into RB80 in S0CON register, while the stop bit is ignored. The baud rate can be configured to 1/32 or 1/64 the system clock frequency.

Figure 17–2. Mode 2, 3 Data Frame



Mode 3: Mode 3 is the same as Mode 2 except the baud rate is variable.

In all four modes, transmission is initiated by any instruction that uses S0BUF as a destination register. In Mode 0, reception is initiated by the condition RI0=0 and REN0=1. In the other modes, reception is initiated by the incoming start bit with 1-to-0 transition if REN0=1.

In addition to the standard operation, the UART0 can perform framing error detection by looking for missing stop bits, and automatic address recognition.

17.2. Serial Port 0 Mode 0

Serial data enters and exits through RXD0. TXD0 outputs the shift clock. 8 bits are transmitted/received: 8 data bits (LSB first). The shift clock source can be selected to 1/12 or 1/4 the system clock frequency by URM0X3 setting in S0CFG register.
Figure 17–3 shows a simplified functional diagram of the serial port 0 in Mode 0.

Transmission is initiated by any instruction that uses S0BUF as a destination register. The “write to S0BUF” signal triggers the UART0 engine to start the transmission. The data in the S0BUF would be shifted into the RXD0(P3.0) pin by each raising edge shift clock on the TXD0(P3.1) pin. After eight raising edge of shift clocks passing, TI0 would be asserted by hardware to indicate the end of transmission and its interrupt vector can be switched to System Flag interrupt by BTI and UTIE gated. Figure 17–4 shows the transmission waveform in Mode 0.

Reception is initiated by the condition REN0=1 and RI0=0. At the next instruction cycle, the Serial Port 0 Controller writes the bits 11111110 to the receive shift register, and in the next clock phase activates Receive.

Receive enables Shift Clock which directly comes from RX Clock to the alternate output function of TXD0 pin. When Receive is active, the contents on the RXD0 pin would be sampled and shifted into shift register by falling edge of shift clock. After eight falling edge of shift clock, RI0 would be asserted by hardware to indicate the end of reception. Figure 17–5 shows the reception waveform in Mode 0.

Figure 17–3. Serial Port 0 Mode 0

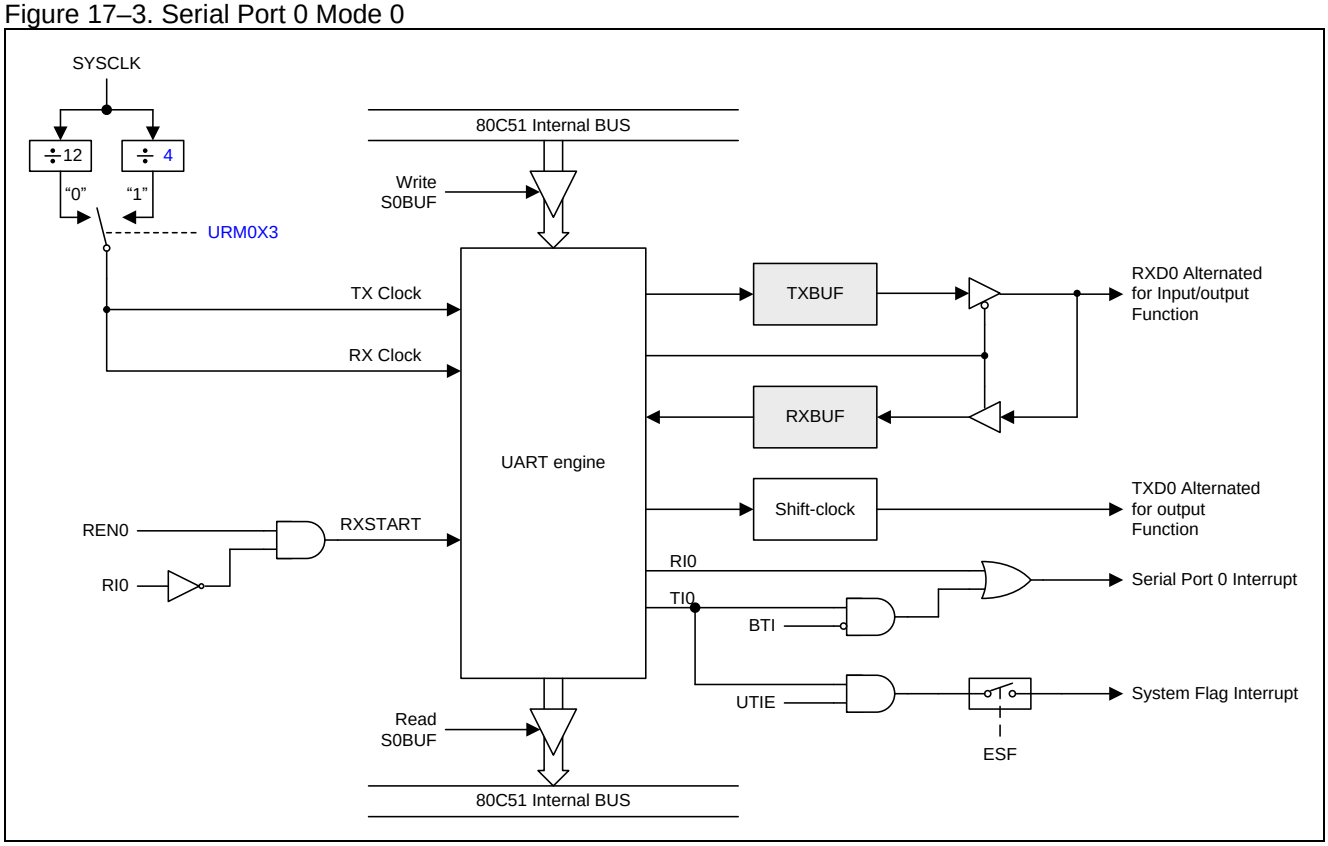


Figure 17–4. Mode 0 Transmission Waveform

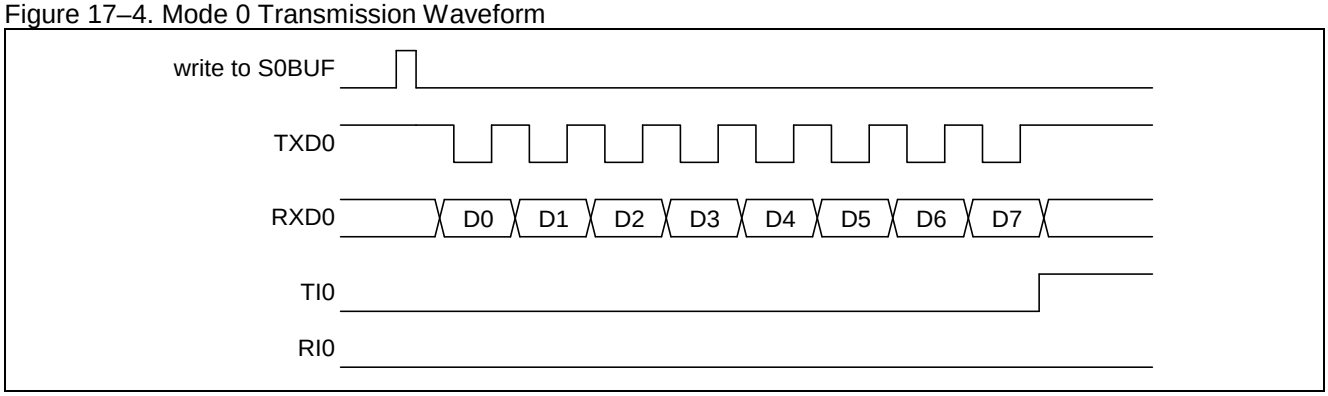
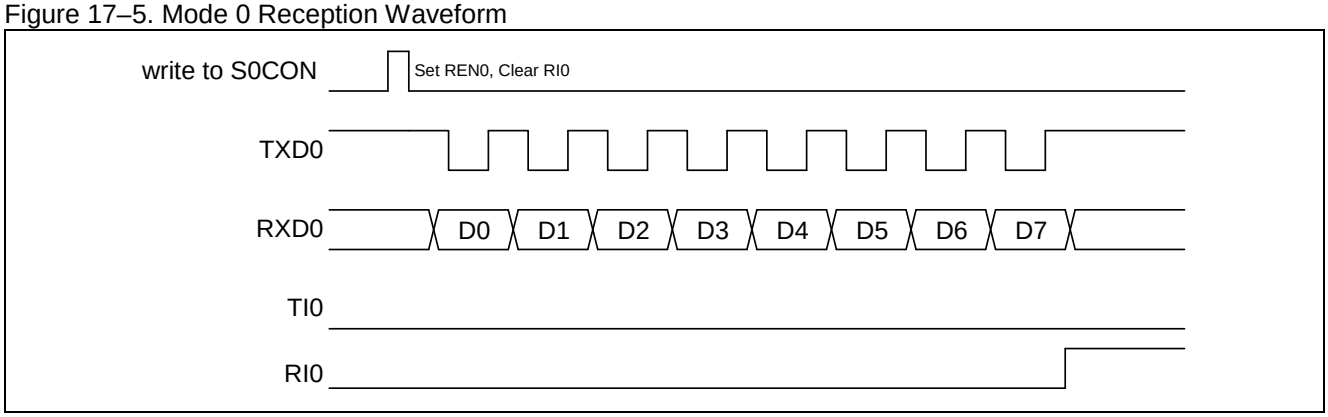


Figure 17–5. Mode 0 Reception Waveform



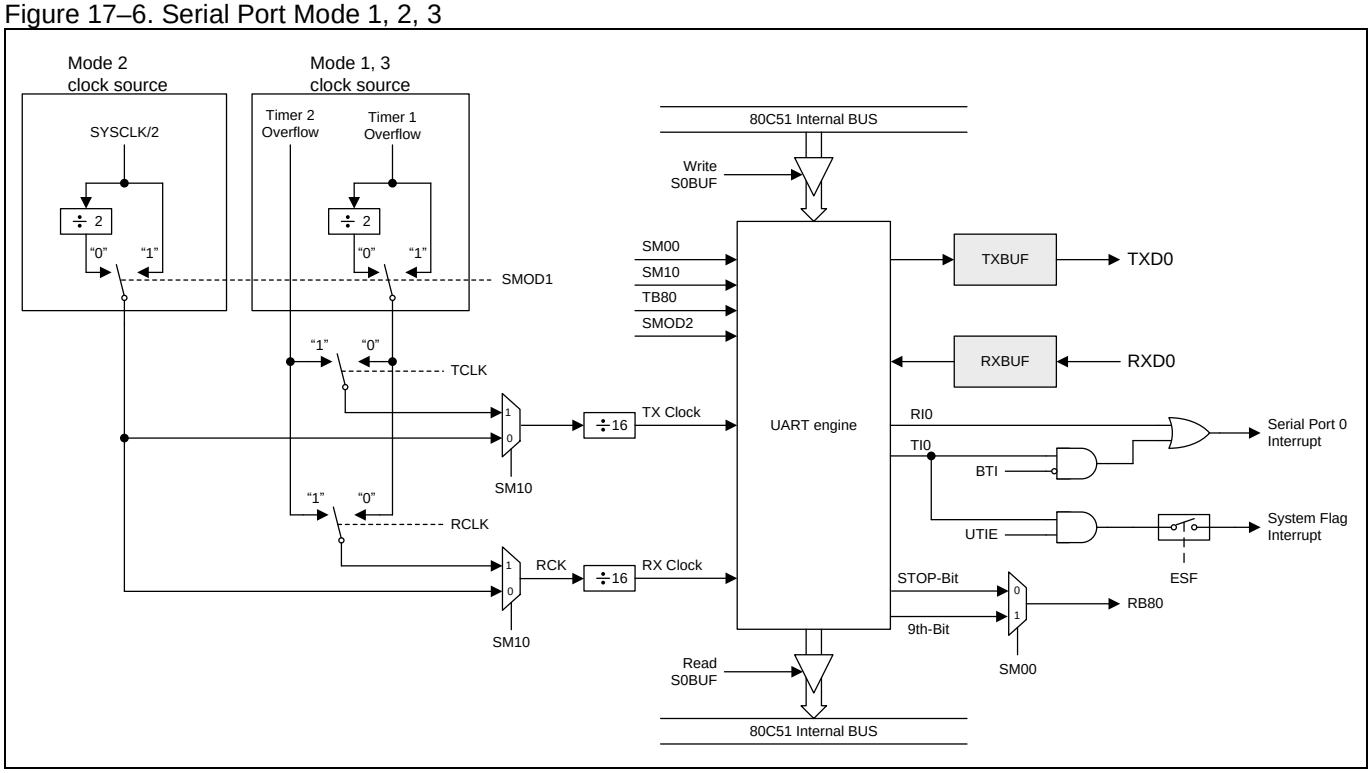
17.3. Serial Port 0 Mode 1

10 bits are transmitted through TXD0, or received through RXD0: a start bit (0), 8 data bits (LSB first), and a stop bit (1). On receive, the stop bit goes into RB80 in S0CON. The baud rate is determined by the Timer 1 or Timer 2 overflow rate. Figure 17–1 shows the data frame in Mode 1 and Figure 17–6 shows a simplified functional diagram of the serial port in Mode 1.

Transmission is initiated by any instruction that uses S0BUF as a destination register. The “write to S0BUF” signal requests the UART0 engine to start the transmission. After receiving a transmission request, the UART0 engine would start the transmission at the raising edge of TX Clock. The data in the S0BUF would be serial output on the TXD0 pin with the data frame as shown in Figure 17–1 and data width depend on TX Clock. After the end of 8th data transmission, TI0 would be asserted by hardware to indicate the end of data transmission and its interrupt vector can be switched to System Flag interrupt by BTI and UTIE gated.

Reception is initiated when Serial Port 0 Controller detected 1-to-0 transition at RXD0 sampled by RCK. The data on the RXD0 pin would be sampled by Bit Detector in Serial Port 0 Controller. After the end of STOP-bit reception, RI0 would be asserted by hardware to indicate the end of data reception and load STOP-bit into RB80 in S0CON register.

Figure 17–6. Serial Port Mode 1, 2, 3



17.4. Serial Port 0 Mode 2 and Mode 3

11 bits are transmitted through TXD0, or received through RXD0: a start bit (0), 8 data bits (LSB first), a programmable 9th data bit, and a stop bit (1). On transmit, the 9th data bit (TB80) can be assigned the value of 0 or 1. On receive, the 9th data bit goes into RB80 in S0CON. The baud rate is programmable to select one of 1/16, 1/32 or 1/64 the system clock frequency in Mode 2. Mode 3 may have a variable baud rate generated from Timer 1 or Timer 2.

Figure 17–2 shows the data frame in Mode 2 and Mode 3. Figure 17–5 shows a functional diagram of the serial port in Mode 2 and Mode 3. The receive portion is exactly the same as in Mode 1. The transmit portion differs from Mode 1 only in the 9th bit of the transmit shift register.

The “write to S0BUF” signal requests the Serial Port 0 Controller to load TB80 into the 9th bit position of the transmit shit register and starts the transmission. After receiving a transmission request, the UART0 engine would start the transmission at the raising edge of TX Clock. The data in the S0BUF would be serial output on the TXD0 pin with the data frame as shown in Figure 17–2 and data width depend on TX Clock. After the end of 9th data transmission, TI0 would be asserted by hardware to indicate the end of data transmission and its interrupt vector can be switched to System Flag interrupt by BTI and UTIE gated.

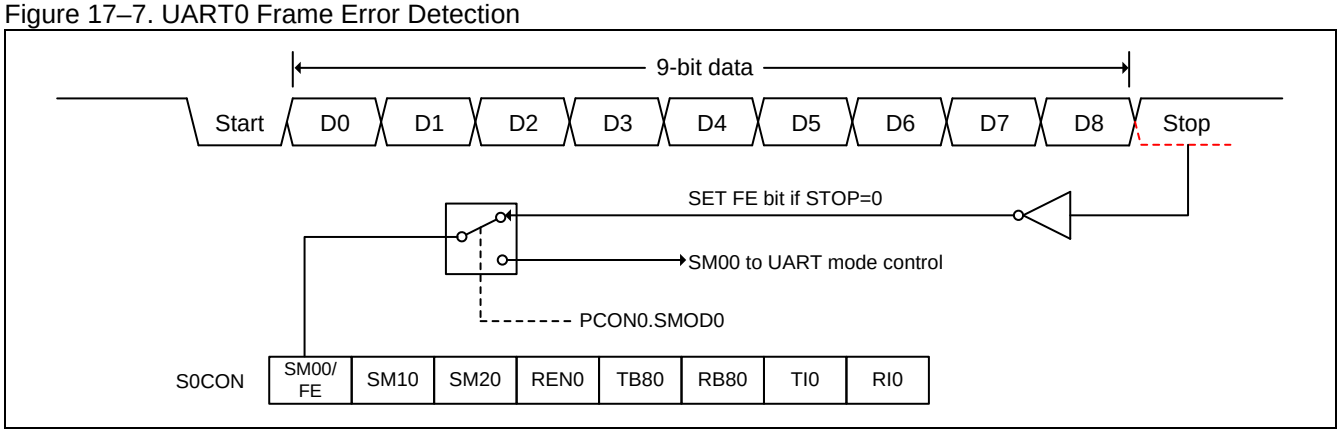
Reception is initiated when the UART0 engine detected 1-to-0 transition at RXD0 sampled by RCK. The data on the RXD0 pin would be sampled by Bit Detector in UART0 engine. After the end of 9th data bit reception, RI0 would be asserted by hardware to indicate the end of data reception and load the 9th data bit into RB80 in S0CON register.

In all four modes, transmission is initiated by any instruction that use S0BUF as a destination register. Reception is initiated in mode 0 by the condition RI0 = 0 and REN0 = 1. Reception is initiated in the other modes by the incoming start bit with 1-to-0 transition if REN0=1.

17.5. Frame Error Detection

When used for framing error detection, the UART0 looks for missing stop bits in the communication. A missing stop bit will set the FE bit in the S0CON register. The FE bit shares the S0CON.7 bit with SM00 and the function of S0CON.7 is determined by SMOD0 bit (PCON.6). If SMOD0 is set then S0CON.7 functions as FE. S0CON.7 functions as SM00 when SMOD0 is cleared. When S0CON.7 functions as FE, it can only be cleared by firmware. Refer to Figure 17–7.

Figure 17–7. UART0 Frame Error Detection



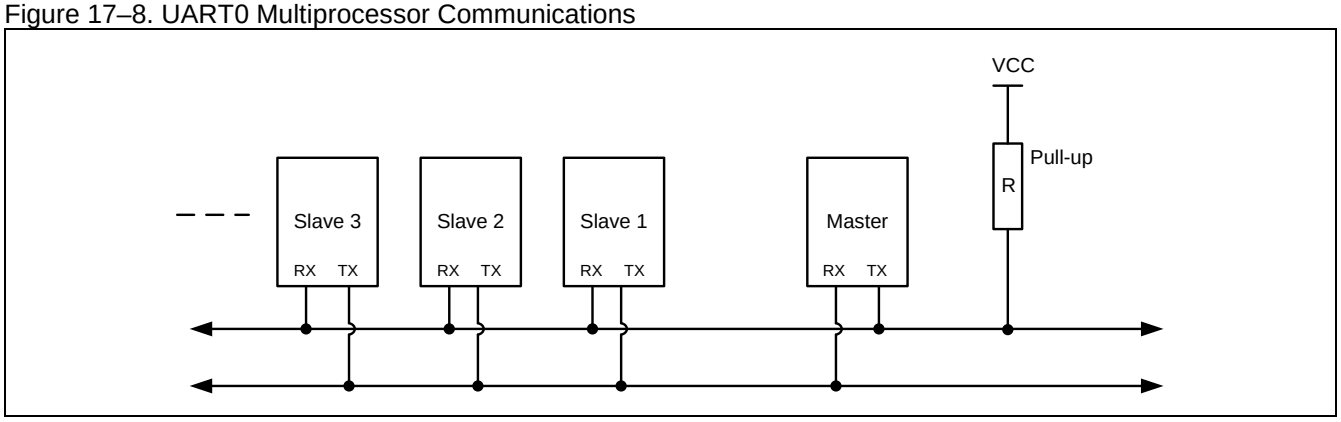
17.6. Multiprocessor Communications

Modes 2 and 3 have a special provision for multiprocessor communications as shown in Figure 17–8. In these two modes, 9 data bits are received. The 9th bit goes into RB80. Then comes a stop bit. The port can be programmed such that when the stop bit is received, the serial port interrupt will be activated only if RB80=1. This feature is enabled by setting bit SM20 (in S0CON register). A way to use this feature in multiprocessor systems is as follows:

When the master processor wants to transmit a block of data to one of several slaves, it first sends out an address byte which identifies the target slave. An address byte differs from a data byte in that the 9th bit is 1 in an address byte and 0 in a data byte. With SM20=1, no slave will be interrupted by a data byte. An address byte, however, will interrupt all slaves, so that each slave can examine the received byte and check if it is being addressed. The addressed slave will clear its SM20 bit and prepare to receive the data bytes that will be coming. The slaves that weren't being addressed leave their SM20 set and go on about their business, ignoring the coming data bytes.

SM20 has no effect in Mode 0, and in Mode 1 can be used to check the validity of the stop bit. In a Mode 1 reception, if SM20=1, the receive interrupt will not be activated unless a valid stop bit is received.

Figure 17–8. UART0 Multiprocessor Communications

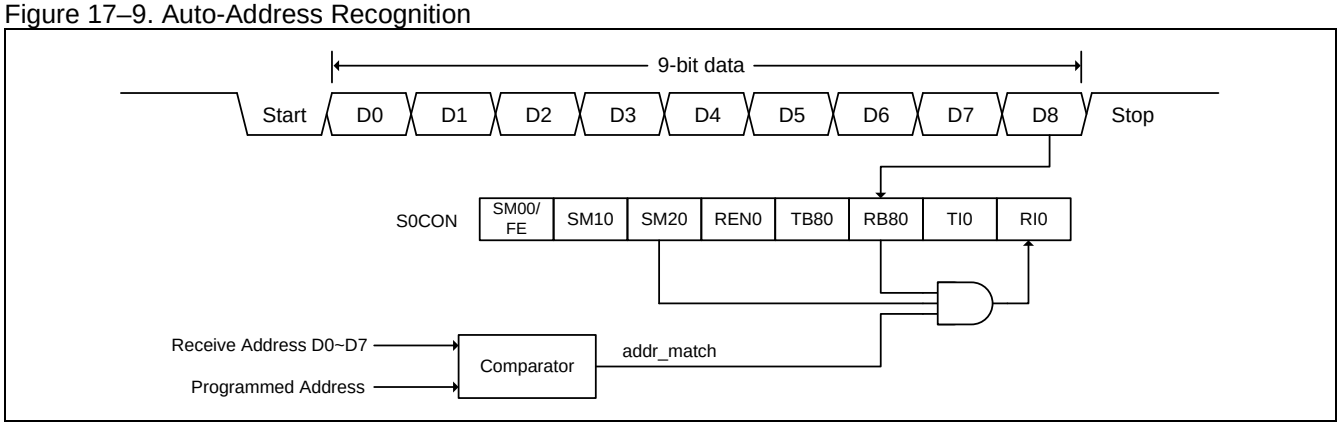


17.7. Automatic Address Recognition

Automatic Address Recognition is a feature which allows the UART0 to recognize certain addresses in the serial bit stream by using hardware to make the comparisons. This feature saves a great deal of firmware overhead by eliminating the need for the firmware to examine every serial address which passes by the serial port. This feature is enabled by setting the SM20 bit in S0CON.

In the 9 bit UART modes, mode 2 and mode 3, the Receive Interrupt flag (RI0) will be automatically set when the received byte contains either the "Given" address or the "Broadcast" address. The 9-bit mode requires that the 9th information bit is a 1 to indicate that the received information is an address and not data. Automatic address recognition is shown in Figure 17–9.

Figure 17–9. Auto-Address Recognition



Note:

(1) After address matching (addr_match=1), Clear SM20 to receive data bytes

(2) After all data bytes have been received, Set SM20 to wait for next address.

The 8 bit mode is called Mode 1. In this mode the RI flag will be set if SM20 is enabled and the information received has a valid stop bit following the 8 address bits and the information is either a Given or Broadcast address. Mode 0 is the Shift Register mode and SM20 is ignored.

Using the Automatic Address Recognition feature allows a master to selectively communicate with one or more slaves by invoking the Given slave address or addresses. All of the slaves may be contacted by using the Broadcast address. Two special Function Registers are used to define the slave's address, SADDR, and the address mask, SADEN.

SADEN is used to define the bits in the SADDR which bits are available, and the remaining bits are "don't care". To use the SADEN mask, perform a logical AND operation on SADDR to create a "Given" address that will be used as the address of slave devices and the master can send the "Given" address on bus to identify the slave out from multiple slaves.

The following examples will help to show the versatility of this scheme:

Slave 0	**Slave 1**
SADDR = 1100 0000	SADDR = 1100 0000
SADEN = 1111 1101	SADEN = 1111 1110
Given = 1100 00X**0**	Given = 1100 00**0**X

In the above example SADDR is the same and the SADEN data is used to differentiate between the two slaves. Slave 0 requires a 0 in bit 0 and it ignores bit 1. Slave 1 requires a 0 in bit 1 and bit 0 is ignored. A unique address for Slave 0 would be 1100 0010 since slave 1 requires a 0 in bit 1. A unique address for slave 1 would be 1100 0001 since a 1 in bit 0 will exclude slave 0. Both slaves can be selected at the same time by an address which has bit 0 = 0 (for slave 0) and bit 1 = 0 (for slave 1). Thus, both could be addressed with 1100 0000.

In a more complex system the following could be used to select slaves 1 and 2 while excluding slave 0:

Slave 0	**Slave 1**	**Slave 2**
SADDR = 1100 0000	SADDR = 1110 0000	SADDR = 1110 0000
SADEN = 1111 1001	SADEN = 1111 1010	SADEN = 1111 1100
Given = 1100 0XX**0**	Given = 1110 0X**0**X	Given = 1110 0**0**XX

In the above example the differentiation among the 3 slaves is in the lower 3 address bits. Slave 0 requires that bit 0 = 0 and it can be uniquely addressed by 1110 0110. Slave 1 requires that bit 1 = 0 and it can be uniquely addressed by 1110 0101. Slave 2 requires that bit 2 = 0 and its unique address is 1110 0011. To select Slaves 0 and 1 and exclude Slave 2 use address 1110 0100, since it is necessary to make bit 2 = 1 to exclude slave 2.

The Broadcast Address for each slave is created by taking the logical OR of SADDR and SADEN. Zeros in this result are treated as don't-cares. In most cases, interpreting the don't-cares as ones, the broadcast address will be FF hexadecimal.

When RESET, the SADDR (SFR address 0xA9) and SADEN (SFR address 0xB9) are loaded with 0s. This generates a "Given" address as "don't cares" ("XXXXXXXXb") and a "Broadcast" address as "don't cares" ("XXXXXXXXb"). This effectively disables the Automatic Addressing mode and allows the micro-controller to use standard 80C51 type UART drivers which do not make use of this feature.

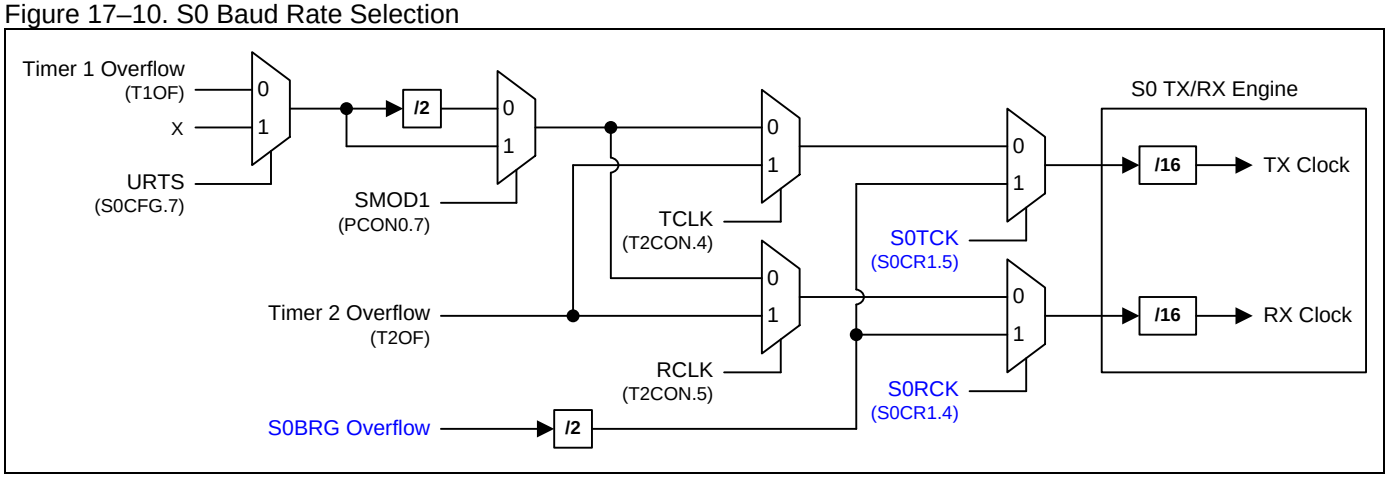
17.8. Baud Rate Setting

Bits T2X12 (T2MOD.4), T1X12 (AUXR2.3), URM0X3 (S0CFG.5) and SMOD2 (S0CFG.6) provide a new option for the baud rate setting, as listed below. For the Enhance mode, please reference "17.11 Serial Port 0 Enhance function" for detail.

17.8.1. Baud Rate Selection in S0

In the Mode 1 and Mode 3 operation of the UART0, the software can select Timer 1 as the Baud Rate Generator by clearing bits TCLK and RCLK in T2CON register. In other words, the user can adopt S1BRG as the Baud Rate Generator for Mode 1 or Mode 3 of the UART0 as long as RCLK=0, TCLK=0 and URTS=1. In this condition, Timer 1 is free for other application. Of course, if UART1 (Mode 1 or Mode 3) is also operated at this time, these two UARTs will have the same baud rates.

Figure 17–10. S0 Baud Rate Selection



17.8.2. Baud Rate in Shift Register Mode (Mode 0 and Enhanced Mode)

$$\text{Mode 0 Baud Rate} = \frac{F_{SYSCLK}}{n}$$; n=12, if URM0X3=0; n=4, if URM0X3=1

Note:
If URM0X3=0, the baud rate formula is as same as standard 8051.

Enhance Mode:

$$\frac{2^{(SMOD2+2)}}{32} \times \frac{F_{SYSCLK}}{12 \times (256 - S0BRT)}$$;S0TX12 = 0

$$\frac{2^{(SMOD2+2)}}{32} \times \frac{F_{SYSCLK}}{1 \times (256 - S0BRT)}$$;S0TX12 = 1

17.8.3. Baud Rate in Mode 2

When S0BC0 = 0,

$$\text{Mode 2 Baud Rate} = \frac{2^{SMOD1} \times 2^{(SMOD2 \times 2)}}{64} \times F_{SYSCLK}$$

When S0BC0 = 1,

$$\text{Mode 2 Baud Rate} = \frac{2^{SMOD1} \times 2^{(SMOD2 \times 2)}}{48} \times F_{SYSCLK}$$

Note:

If SMOD2=0, the baud rate formula is as same as standard 8051. If SMOD2=1, there is an enhanced function for baud rate setting. Table 17–2 defines the Baud Rate setting with SMOD2 factor in Mode 2 baud rate generator.

Table 17–2. SMOD2 application criteria in Mode 2

SMOD2	SMOD1	Baud Rate	Note	Recommended Max. Receive Error (%)
0	0	Default Baud Rate	Standard function	± 3%
0	1	Double Baud Rate	Standard function	± 3%
1	0	Double Baud Rate **X2**	Enhanced function	± 2%
1	1	Double Baud Rate **X4**	Enhanced function	± 1%

Note: When Timer 1 in Double Baud Rate x4 (SMOD1=1 & SMOD2=1) mode, the TH1 can not equal to 254 & 255.

Table 17–3. S0 Mode 2 Baud Rates @ F_{SYSCLK}=16MHz & *S0BC0 = 0*

Baud Rate	SMOD2	SMOD1	Error
250,000	0	0	0.0%
500,000	0	1	0.0%
1,000,000	1	0	0.0%
2,000,000	1	1	0.0%

Table 17–4. S0 Mode 2 Baud Rates @ F_{SYSCLK}=16MHz & *S0BC0 = 1*

Baud Rate	SMOD2	SMOD1	Error
333,333	0	0	0.0%
666,667	0	1	0.0%
1,333,333	1	0	0.0%
2,666,667	1	1	0.0%

17.8.4. Baud Rate in Mode 1 & 3 & Enhance Mode

17.8.4.1. Using Timer 1 as the Baud Rate Generator

When S0BC0 = 0,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{SMOD1} \times 2^{(SMOD2 \times 2)}}{32} \times \frac{F_{SYSCLK}}{12 \times (256 - TH1)} \; ; \text{T1X12=0}$$

$$\text{or} = \frac{2^{SMOD1} \times 2^{(SMOD2 \times 2)}}{32} \times \frac{F_{SYSCLK}}{1 \times (256 - TH1)} \; ; \text{T1X12=1}$$

When S0BC0 = 1,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{SMOD1} \times 2^{(SMOD2 \times 2)}}{24} \times \frac{F_{SYSCLK}}{12 \times (256 - TH1)} \; ; \text{T1X12=0}$$

$$\text{or} = \frac{2^{SMOD1} \times 2^{(SMOD2 \times 2)}}{24} \times \frac{F_{SYSCLK}}{1 \times (256 - TH1)} \; ; \text{T1X12=1}$$

Note:

If SMOD2=0, T1X12=0, the baud rate formula is as same as standard 8051. If SMOD2=1, there is an enhanced function for baud rate setting. Table 17–5 defines the Baud Rate setting with SMOD2 factor in Timer 1 baud rate generator.

Table 17–5. SMOD2 application criteria in Mode 1 & 3 using Timer 1

SMOD2	SMOD1	Baud Rate	Note	Recommended Max. Receive Error (%)
0	0	Default Baud Rate	Standard function	± 3%
0	1	Double Baud Rate	Standard function	± 3%
1	0	Double Baud Rate **X2**	Enhanced function	± 2%
1	1	Double Baud Rate **X4**	Enhanced function	± 1%

Note: When Timer 1 in Double Baud Rate x4 (SMOD1=1 & SMOD2=1) mode, the TH1 can not equal to 254 & 255.

Figure 17–6 ~ Table 17–9 list various commonly used baud rates and how they can be obtained from Timer 1 in its 8-Bit Auto-Reload Mode. For the non-standard Baud Rate, the maximum frequency is 3MHz when F_{SYSCLK} = 24MHz).

Table 17–6. Timer 1 Generated Commonly Used Baud Rates @ F_{SYSCLK}=16.0MHz & *S0BC0 = 0*

Baud Rate	**TH1**, the Reload Value					
	T1X12=0 & SMOD2=0			**T1X12=1 & SMOD2=0**		
	SMOD=0	SMOD=1	Error	SMOD1=0	SMOD1=1	Error
1200	221(0.79%)	187(0.64%)	0.79%	--	--	--
2400	239(2.12%)	221(0.79%)	2.12%	48	--	0.16%
4800	--	239	2.12%	152	48	0.16%
9600	--	--	--	204	152	0.16%
14400	--	--	--	221	186	0.79%
19200	--	--	--	230	204	0.16%
28800	--	--	--	239(2.21%)	221(0.79%)	2.12%
38400	--	--	--	243	230	0.16%
57600	--	--	--	--	239	2.12%
115200	--	--	--	--	--	--

Table 17–7. Timer 1 Generated High Baud Rates @ F_{SYSCLK}=16.0MHz & *S0BC0 = 0*

Baud Rate	**TH1**, the Reload Value					
	T1X12=0 & SMOD2=1			**T1X12=1 & SMOD2=1**		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
115.2K	--	--	--	239(2.21%)	221(0.79%)	2.12%
230.4K	--	--	--	--	239	2.12%
460.8K	--	--	--	--	--	--

Table 17–8. Timer 1 Generated Commonly Used Baud Rates @ F_{SYSCLK}=16.0MHz & *S0BC0 = 1*

Baud Rate	**TH1**, the Reload Value					
	T1X12=0 & SMOD2=0			**T1X12=1 & SMOD2=0**		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
1200	210(0.64%)	163(0.44%)	0.64%	--	--	--
2400	233	210	0.64%	--	--	--
4800	--	233	0.64%	117	--	0.08%
9600	--	--	--	187(0.64%)	117(0.08%)	0.64%
14400	--	--	--	210(0.64%)	163(0.44%)	0.64%
19200	--	--	--	221(0.79%)	187(0.64%)	0.79%
28800	--	--	--	233	210	0.64%
38400	--	--	--	239(2.21%)	221(0.79%)	2.12%
57600	--	--	--	--	233	0.64%
115200	--	--	--	--	--	--

Table 17–9. Timer 1 Generated High Baud Rates @ F_{SYSCLK}=16.0MHz & *S0BC0 = 1*

Baud Rate	**TH1**, the Reload Value					
	T1X12=0 & SMOD2=1			**T1X12=1 & SMOD2=1**		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
115.2K	--	--	--	233	--	0.64%
230.4K	--	--	--	--	--	--
460.8K	--	--	--	--	--	--

17.8.4.2. Using Timer 2 as the Baud Rate Generator

When Timer 2 is used as the baud rate generator (either TCLK or RCLK in T2CON is '1'), the baud rate is as follows.

When S0BC0 = 0,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{\text{SMOD2 X (SMOD1 + 1)}} \times F_{SYSCLK}}{32 \times (65536 - (\text{RCAP2H, RCAP2L}))} \text{ ; T2X12=0}$$

$$\text{or} = \frac{2^{\text{SMOD2 X (SMOD1 +1)}} \times F_{SYSCLK}}{16 \times (65536 - (\text{RCAP2H, RCAP2L}))} \text{ ; T2X12=1}$$

When S0BC0 = 1,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{\text{SMOD2 X (SMOD1 + 1)}} \times F_{SYSCLK}}{24 \times (65536 - (\text{RCAP2H, RCAP2L}))} \text{ ; T2X12=0}$$

$$\text{or} = \frac{2^{\text{SMOD2 X (SMOD1 + 1)}} \times F_{SYSCLK}}{12 \times (65536 - (\text{RCAP2H, RCAP2L}))} \text{ ; T2X12=1}$$

Note:

If SMOD2=0, the baud rate formula is as same as standard 8051. If SMOD2=1, there is an enhanced function for baud rate setting. Table 17–10 defines the Baud Rate setting with SMOD2 factor in Timer 2 baud rate generator.

Table 17–10. SMOD2 application criteria in Mode 1 & 3 using Timer 2

SMOD2	SMOD1	Baud Rate	Note	Recommended Max. Receive Error (%)
0	X	Default Baud Rate	Standard function	± 3%
1	0	Double Baud Rate	Enhanced function	± 3%
1	1	Double Baud Rate **X2**	Enhanced function	± 2%

Note: When Timer 2 in Double Baud Rate x2 (SMOD1=1 & SMOD2=1) mode, the RCAP2H & RPAC2L can not equal to 65534 & 65535.

Figure 17–11 ~ Table 17–14 list various commonly used baud rates and how they can be obtained from Timer 2 in its Baud-Rate Generator Mode.

Table 17–11. Timer 2 Generated Commonly Used Baud Rates @ F_{SYSCLK}=16.0MHz *& S0BC0 = 0*

Baud Rate	**[RCAP2H, RCAP2L]**, the Reload Value					
	T2X12=0 & SMOD2=0			**T2X12=1 & SMOD2=0**		
	SMOD=0	SMOD=1	Error	SMOD=0	SMOD=1	Error
1200	65120	65120	0.16%	64704	64704	0.16%
2400	65328	65328	0.16%	65120	65120	0.16%
4800	65432	65432	0.16%	65328	65328	0.16%
9600	65484	65484	0.16%	65432	65432	0.16%
14400	65501	65501	0.79%	65466	65466	0.79%
19200	65510	65510	0.16%	65484	65484	0.16%
28800	65519	65519	2.12%	65501	65501	0.79%
38400	65523	65523	0.16%	65510	65510	0.16%
57600	--	--	--	65519	65519	2.12%
115200	--	--	--	--	--	--

Table 17–12. Timer 2 Generated High Baud Rates @ F_{SYSCLK}=16.0MHz *& S0BC0 = 0*

Baud Rate	**[RCAP2H, RCAP2L]**, the Reload Value					
	T2X12=0 & SMOD2=1			**T2X12=1 & SMOD2=1**		
	SMOD1=0	SMOD1=1	Error	SMOD=0	SMOD=1	Error
115.2K	--	65519	2.12%	65519(2.21%)	65515(0.79%)	2.12%
230.4K	--	--	--	--	65519	2.12%

Table 17–13. Timer 2 Generated Commonly Used Baud Rates @ F_{SYSCLK}=16.0MHz & *S0BC0 = 1*

Baud Rate	**[RCAP2H, RCAP2L]**, the Reload Value					
	T2X12=0 & SMOD2=0			**T2X12=1 & SMOD2=0**		
	SMOD=0	SMOD=1	Error	SMOD=0	SMOD=1	Error
1200	64980	64980	0.08%	64424	64424	0.08%
2400	65258	65258	0.08%	64980	64980	0.08%
4800	65397	65397	0.08%	65258	65258	0.08%
9600	65467	65467	0.64%	65397	65397	0.08%
14400	65490	65490	0.64%	65443	65443	0.44%
19200	65501	65501	0.79%	65467	65467	0.64%
28800	65513	65513	0.64%	65490	65490	0.64%
38400	65519	65519	2.21%	65501	65501	0.79%
57600	--	--	--	65513	65513	0.64%
115200	--	--	--	--	--	--

Table 17–14. Timer 2 Generated High Baud Rates @ F_{SYSCLK}=24.0MHz

Baud Rate	**[RCAP2H, RCAP2L]**, the Reload Value					
	T2X12=0 & SMOD2=1			**T2X12=1 & SMOD2=1**		
	SMOD1=0	SMOD1=1	Error	SMOD=0	SMOD=1	Error
115.2K	--	65513	0.64%	--	--	--
230.4K	--	--	--	--	--	--

17.8.4.3. Using Split Timer 2 as the Baud Rate Generator

When Split Timer 2 is used as the baud rate generator (either TCLK or RCLK in T2CON is '1'), the baud rate is as follows.

When S0BC0 = 0,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{\text{SMOD2 X (SMOD1 + 1)}} \times F_{SYSCLK}}{16 \times 12 \times (256 - \text{RCAP2L})} \text{ ; TL2X12=0}$$

$$\text{or} = \frac{2^{\text{SMOD2 X SMOD1 +1)}} \times F_{SYSCLK}}{16 \times 1 \times (256 - \text{RCAP2L})} \text{ ; TL2X12=1}$$

When S0BC0 = 1,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{\text{SMOD2 X (SMOD1 + 1)}} \times F_{SYSCLK}}{12 \times 12 \times (256 - \text{RCAP2L})} \text{ ; TL2X12=0}$$

$$\text{or} = \frac{2^{\text{SMOD2 X SMOD1 +1)}} \times F_{SYSCLK}}{12 \times 1 \times (256 - \text{RCAP2L})} \text{ ; TL2X12=1}$$

Note:
Table 17–15 defines the Baud Rate setting with SMOD2 and SMOD1 factors in Split Timer 2 baud rate generator.

Table 17–15. SMOD2 application criteria in Mode 1 & 3 using Split Timer 2

SMOD2	SMOD1	Baud Rate	Note	Recommended Max. Receive Error (%)
0	X	Default Baud Rate	Standard function	± 3%
1	0	Double Baud Rate	Enhanced function	± 3%
1	1	Double Baud Rate **X2**	Enhanced function	± 2%

Note: When Timer 2 in Double Baud Rate x2 (SMOD1=1 & SMOD2=1) mode, the RPAC2L can not equal to 254 & 255.

17.8.4.4. Using S0 Baud Rate Timer as the Baud Rate Generator (S0BRG under Enhance Mode)

The S0 of **MG82F6B08 / 6B001/ 6B104** has embedded a dedicated baud rate generator (S0BRG), which detailed function is described in Section "17.11 Serial Port 0 Enhance function". When S0BRG is used as the baud rate generator of S0, the baud rate is as follows.

When S0BC0 = 0,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{(\text{SMOD2})}}{32} \times \frac{F_{SYSCLK}}{12 \times (256 - \text{S0BRT})} \text{ ; S0TX12=0, SMOD1=0}$$

$$\text{or} = \frac{2^{(\text{SMOD2})}}{32} \times \frac{F_{SYSCLK}}{1 \times (256 - \text{S0BRT})} \text{ ; S0TX12=1, SMOD1=0}$$

When S0BC0 = 1,

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{(\text{SMOD2})}}{24} \times \frac{F_{SYSCLK}}{12 \times (256 - \text{S0BRT})} \text{ ; S0TX12=0, SMOD1=0}$$

$$\text{or} = \frac{2^{(\text{SMOD2})}}{24} \times \frac{F_{SYSCLK}}{1 \times (256 - \text{S0BRT})} \text{ ; S0TX12=1, SMOD1=0}$$

Note:
Table 17–16 defines the Baud Rate setting with SMOD2 and SMOD1 factors in S0BRG.

Table 17–16. SMOD2 application criteria in Mode 1 & 3 using S0BRG

SMOD2	SMOD1	Baud Rate	Note	Recommended Max. Receive Error (%)
0	0	Default Baud Rate	Standard function	± 3%
1	0	Double Baud Rate	Enhanced function	± 3%
X	1	Not support		

17.9. Serial Port 0 Mode 4 (SPI Master)

The Serial Port 0 of **MG82F6B08 / 6B001/ 6B104** is embedded an additional Mode 4 to support SPI master engine. The Mode 4 is selected by SM30, SM00 and SM10. Please reference "Table 17–1. Serial Port 0 Mode Selection".

URM0X3 also controls the SPI transfer speed. If URM0X3 = 0, the SPI clock frequency is SYSCLK/12. If URM0X3 = 1, the SPI clock frequency is SYSCLK/4.

The SPI master in **MG82F6B08 / 6B001/ 6B104** uses the TXD0 as SPICLK, RXD0 as MOSI, and S0MI as MISO. nSS is selected by MCU software on other port pin. Figure 17–11 shows the SPI connection. It also can support the configuration for multiple slave communication in Figure 17–12.

Figure 17–11. Serial Port 0 Mode 4, Single Master and Single Slave configuration (n = 0)

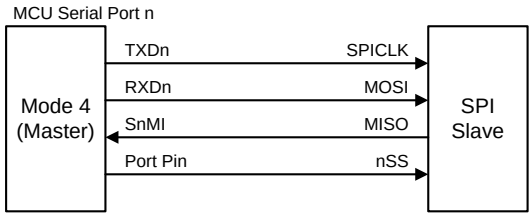
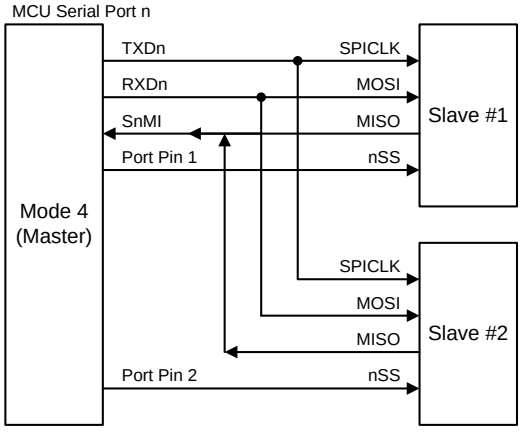


Figure 17–12. Serial Port 0 Mode 4, Single Master and Multiple Slaves configuration (n = 0)



The SPI master satisfies the transfer with the full function SPI module of Megawin MG82/84 series MCU with CPOL, CPHA and DORD selection. For CPOL and CPHA condition, **MG82F6B08 / 6B001/ 6B104** uses an easy way by initialize SPI clock assigned port pin (TXD0) polarity to fit them. Table 17–17 shows the serial port Mode 4 mapping with the four SPI operating mode.

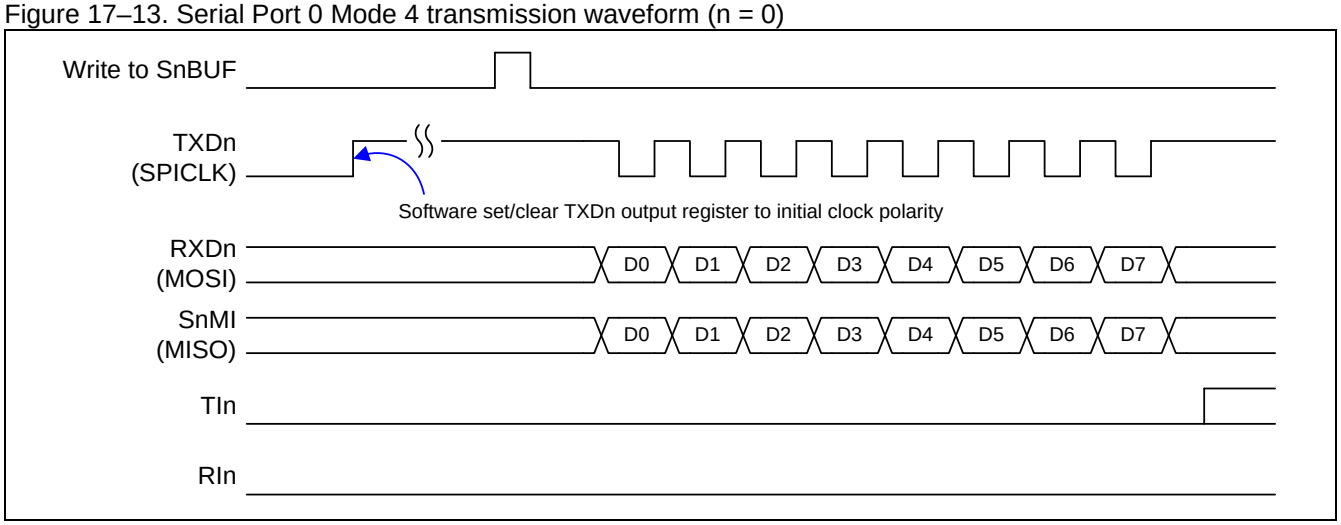
Table 17–17. SPI mode mapping with Serial Port Mode 4 configuration

SPI Mode	CPOL	CPHA	Configuration in TXD0
0	0	0	Clear TXD0 output register to "0"
1	0	1	Clear TXD0 output register to "0"
2	1	0	Set TXD0 output register to "1"
3	1	1	Set TXD0 output register to "1"

For bit order control (DORD) on SPI serial transfer, **MG82F6B08 / 6B001/ 6B104** provides a control bit, S0DOR, to control the data bit order by software program. S0DOR default is “1”, LSB first.

Transmission is initiated by any instruction that uses S0BUF as a destination register. The “write to S0BUF” signal triggers the UART engine to start the transmission. The data in the S0BUF would be shifted into the RXD0 pin as MOSI serial data. The SPI shift clock is built on the TXD0 pin for SPICLK output. After eight raising edge of shift clocks passing, TI0 would be asserted by hardware to indicate the end of transmission. And the contents on the S0MI pin would be sampled and shifted into shift register. Then, “read S0BUF” can get the SPI shift-in data. Figure 17–13 shows the transmission waveform in Mode 0. RI0 will not be asserted in Mode 4.

Figure 17–13. Serial Port 0 Mode 4 transmission waveform (n = 0)



17.10. Serial Port 0 Register

All the four operation modes of the serial port are the same as those of the standard 8051 except the baud rate setting. Three registers, PCON, AUXR2 and S0CFG, are related to the baud rate setting:

S0CON: Serial port 0 Control Register

SFR Page = **0 only**

SFR Address = 0x98 RESET = 0000-0000

7	6	5	4	3	2	1	0
SM00/FE	SM10	SM20	REN0	TB80	RB80	TI0	RI0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: FE, Framing Error bit. The SMOD0 bit must be set to enable access to the FE bit.
0: The FE bit is not cleared by valid frames but should be cleared by software.
1: This bit is set by the receiver when an invalid stop bit is detected.

Bit 7: Serial port 0 mode bit 0, (SMOD0 must = 0 to access bit SM00)
Bit 6: Serial port 0 mode bit 1.

SM30	SM00	SM10	Mode	Description	Baud Rate
0	0	0	0	shift register	SYSCLK/12 or SYSCLK/4
0	0	1	1	8-bit UART	variable
0	1	0	2	9-bit UART	SYSCLK/64, /32, /16, /8
0	1	1	3	9-bit UART	variable
1	0	0	4	**SPI Master**	SYSCLK/12 or SYSCLK/4
1	0	1	5	Reserved	Reserved
1	1	0	6	Reserved	Reserved
1	1	1	7	Reserved	Reserved

Bit 5: Serial port 0 mode bit 2.
0: Disable SM20 function.
1: Enable the automatic address recognition feature in Modes 2 and 3. If SM20=1, RI0 will not be set unless the received 9th data bit is 1, indicating an address, and the received byte is a Given or Broadcast address. In mode1, if SM20=1 then RI0 will not be set unless a valid stop Bit was received, and the received byte is a Given or Broadcast address. In Mode 0, SM20 should be 0.

Bit 4: REN0, Enable serial reception.
0: Clear by software to disable reception.
1: Set by software to enable reception.

Bit 3: TB80, The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.

Bit 2: RB80, In Modes 2 and 3, the 9th data bit that was received. In Mode 1, if SM20 = 0, RB80 is the stop bit that was received. In Mode 0, RB80 is not used.

Bit 1: TI0. Transmit interrupt flag.
0: Must be cleared by software.
1: Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. This bit is also set in mode 4 (SPI master) and mode 6 (SPI slave) after a SPI transfer finished.

Bit 0: RI0. Receive interrupt flag.
0: Must be cleared by software.
1: Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM20).

S0BUF: Serial port 0 Buffer Register

SFR Page = **0 only**
SFR Address = 0x99 RESET = XXXX-XXXX

7	6	5	4	3	2	1	0
S0BUF.7	S0BUF.6	S0BUF.5	S0BUF.4	S0BUF.3	S0BUF.2	S0BUF.1	S0BUF.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: It is used as the buffer register in transmission and reception.

SADDR: Slave Address Register

SFR Page = 0~F
SFR Address = 0xA9 RESET = 0000-0000

7	6	5	4	3	2	1	0
SADDR.7	SADDR.6	SADDR.5	SADDR.4	SADDR.3	SADDR.2	SADDR.1	SADDR.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SADEN: Slave Address Mask Register (SMOD3 = 0)

SFR Page = 0~F
SFR Address = 0xB9 RESET = 0000-0000

7	6	5	4	3	2	1	0
SADEN.7	SADEN.6	SADEN.5	SADEN.4	SADEN.3	SADEN.2	SADEN.1	SADEN.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SADDR register is combined with SADEN register to form Given/Broadcast Address for automatic address recognition. In fact, SADEN functions as the “mask” register for SADDR register. The following is the example for it.

SADDR = 1100 0000
SADEN = 1111 1101
Given = 1100 00x0 ⟶ The Given slave address will be checked except bit 1 is treated as “don’t care”

The Broadcast Address for each slave is created by taking the logical OR of SADDR and SADEN. Zero in this result is considered as “don’t care”. Upon reset, SADDR and SADEN are loaded with all 0s. This produces a Given Address of all “don’t care” and a Broadcast Address of all “don’t care”. This disables the automatic address detection feature.

PCON0: Power Control Register 0

SFR Page = 0~F POR = 0001-0000
SFR Address = 0x87 RESET = 0000-0000

7	6	5	4	3	2	1	0
SMOD1	SMOD0	GF	POF	GF1	GF0	PD	IDL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SMOD1, double Baud rate control bit.
0: Disable double Baud rate of the UART.
1: Enable double Baud rate of the UART in mode 1, 2, or 3.

Bit 6: SMOD0, Frame Error select.
0: S0CON.7 is SM0 function.
1: S0CON.7 is FE function. Note that FE will be set after a frame error regardless of the state of SMOD0.

S0CFG: Serial Port 0 Configuration Register

SFR Page = **0 only**

SFR Address = **0x9C** RESET = 0000-1000

7	6	5	4	3	2	1	0
URTS	SMOD2	URM0X3	SM30	S0DOR	BTI	UTIE	SMOD3
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: URTS, UART0 Timer Selection.
0: Timer 1 or Timer 2 can be used as the Baud Rate Generator in Mode 1 and Mode 3.
1: Timer 1 overflow signal is replaced by the UART1 Baud Rate Timer overflow signal when Timer 1 is selected as the Baud Rate Generator in Mode1 or Mode 3 of the UART0. (Refer Section "17.8.4 Baud Rate in Mode 1 & 3".)

Bit 6: SMOD2, UART0 extra double baud rate selector.
0: Disable extra double baud rate for UART0.
1: Enable extra double baud rate for UART0.

Bit 5: URM0X3, this bit control the baud rate in S0 mode 0 and mode 4.
S1 in mode 0 and mode4:
0: Clear to select SYSCLK/12 as the baud rate for S0 Mode 0 and Mode 4.
1: Set to select SYSCLK/4 as the baud rate for S0 Mode 0 and Mode 4.

Bit 4: SM30, Serial Port Mode control bit 3.

Bit 3: S0DOR, Serial Port 0 data order control in all operating modes.
If S0 is not in Timer mode:
0: The MSB of the data byte is transmitted first.
1: The LSB of the data byte is transmitted first. S0DOR is set to "1" in default.
If S0 is in Timer mode:
0: Set the S0BRG to 8-bit reload timer/counter mode.
1: Set the S0BRG to 16-bit timer/counter mode.

Bit 2: BTI, Block TI0 in Serial Port 0 Interrupt.
0: Retain the TI0 to be a source of Serial Port 0 Interrupt.
1: Block TI0 to be a source of Serial Port 0 Interrupt.

Bit 1: UTIE, S0 TI0 Enabled in system flag interrupt.
0: Disable the interrupt vector sharing for TI0 in system flag interrupt.
1: Set TI0 flag will share the interrupt vector with system flag interrupt.

Bit 0: SMOD3, S0CR1 access control.
0: Disable S0CR1 access. CPU accesses SFR address 0xB9 to read/write SADEN.
1: Enable S0CR1 access. CPU accesses SFR address 0xB9 to read/write S0CR1.

AUXR2: Auxiliary Register 2

SFR Page = 0~F

SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	**R/W**	**W**	**W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 3: T1X12, Timer 1 clock source selector while C/T=0.
0: Clear to select SYSCLK/12.
1: Set to select SYSCLK as the clock source. If set, the UART0 baud rate by Timer 1 in Mode 1 and Mode 3 is 12 times than standard 8051 function.

AUXR3: Auxiliary Register 3

SFR Page = **0 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 3: S0PS0, Serial Port 0 pin Selection 0. (S0PS1 at AUXR10.3)

S0PS1~0	RXD0	TXD0
00	P3.0	P3.1
01	P4.4	P4.5
10	P4.5	P3.0
11	P4.5	P4.4

AUXR6: Auxiliary Register 6

SFR Page = **3 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	T2FCS	SnMIPS	S0COPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 0: S0COPS, S0BRG Clock Output (S0CKO) port pin Selection.

S0COPS	S0CKO
0	P4.7
1	P4.4

17.11. Serial Port 0 Enhance function

If SMOD3 (S0CFG.0) is set, SFR address 0xB9 will be accessed on S0CR1. S0CR1 control the enhanced function of serial port 0 including :

- Enable S0 embedded baud rate generator, S0BRG
- Enable the S0 TX or RX to select the baud rate time base by S0BRG
- Enable S0BRG to behave a general timer
- Enable S0 to enter LIN bus mode

S0CR1: Serial Port 0 Control Register 1 (SMOD3 = 1)

SFR Page = 0~F
SFR Address = 0xB9 RESET = 0000-0000

7	6	5	4	3	2	1	0
S0TR	S0TX12	S0TCK	S0RCK	S0CKOE	ARTE	0	S0BC0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: S0TR, UART0 Baud Rate Generator control bit.
0: Clear to stop S0BRG operation.
1: Set to start S0BRG operation.

Bit 6: S0TX12, S0BRG clock source selection.
0: Clear to select SYSCLK/12 as the clock source of S0BRG.
1: Set to select SYSCLK as the clock source of S0BRG.

Bit 5: S0TCK, S0 control bit to select S0BRG overflow for UART0 transmit clock.
0: Cause Timer 1 or Timer 2 overflow to be used for the transmit clock.
1: Cause the S0 to use S0BRG overflow for it's transmit clock and operating mode control.

Bit 4: S0RCK, S0 control bit to select S0BRG overflow for UART0 receive clock.
0: Cause Timer 1 or Timer 2 overflow to be used for the receive clock.
1: Cause the S0 to use S0BRG overflow for it's receive clock and operating mode control.

Bit 3: S0CKOE, S0BRG clock output control.
0: Disable S0BRG clock output on S0CKO.
1: Enable S0BRG clock output on S0CKO.

Bit 2: ARTE, Auto Repeat Transmit Enable.
0: Disable auto repeat transmit.
1: Auto repeat transmit enable.

Bit 0: S0BC0, S0 Baud rate Control 0.
0: Select UART oversampling by 16 in S0 mode 1/2/3/5.
1: Select UART oversampling by 12 in S0 mode 1/2/3/5.

S0BRT: Serial port 0 Baud Rate Timer Reload Register

SFR Page = **0 only**
SFR Address = 0x9A RESET = 0000-0000

7	6	5	4	3	2	1	0
S0BRT.7	S0BRT.6	S0BRT.5	S0BRT.4	S0BRT.3	S0BRT.2	S0BRT.1	S0BRT.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: It is used as the reload value register for baud rate timer generator that works in a similar manner as Timer 1.

S0BRC: Serial port 0 Baud Rate Counter Register

SFR Page = **0 only**

SFR Address = 0x9B RESET = 0000-0000

7	6	5	4	3	2	1	0
S0BRC.7	S0BRC.6	S0BRC.5	S0BRC.4	S0BRC.3	S0BRC.2	S0BRC.1	S0BRC.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~0: It is used as the reload value register for baud rate timer generator that works in a similar manner as Timer 1. This register can be always read/written by software. If S0TR (S0CR1.7) = 0, software writing S0BRT will store the data content to S0BRT and S0BRC concurrently. If S0TR = 1, software writing S0BRT will not store the data to S0BRC.

17.11.1. S0 Baud Rate Generator (S0BRG)

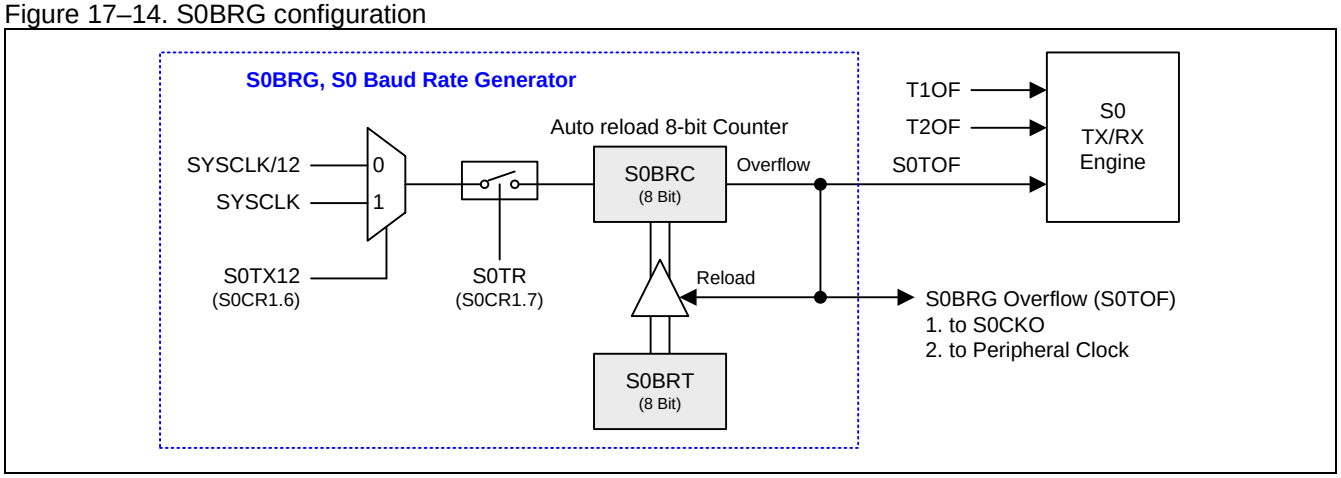
The **MG82F6B08 / 6B001/ 6B104** has an embedded Baud Rate Generator to generate the clock for serial port 0 operation. It is constructed by an 8-bit up-counter, S0BRC, and an 8-bit reload register, S0BRT. The overflow (S0TOF) of S0BRC is the time base of UART0 serial engine in all operation modes and triggers the S0BRT content reloaded into S0BRC for the consecutive counting.

If S0TR = 0, software writing S0BRT will modify S0BRC simultaneously. After S0TR enabled to start the S0BRC counting, it is no influence on S0BRC when S0BRT is writing. Modifying S0BRC is always independent with S0BRT content.

This baud rate generator can also provide the time base for clock output, S0CKO, from the S0BRC overflow rate by 2 (S0TOF/2). S0TOF also supplies the toggle source for other peripherals' clock input. Regardless S0 engine is running or pending, S0BRG always serves the time base function for these peripherals.

The configuration of the Serial Port 0 Baud Rate Generator is shown in Figure 17–14.

Figure 17–14. S0BRG configuration



17.11.2. Independent Baud Rate Generator S0BRG for S0

To give S0 more flexibility, S0 Baud Rate Generator S0BRG can be selected as Baud Rate source.
The configuration of the Serial Port 0 baud rate selection please reference Figure 17–10. S0 Baud Rate Selection.

17.11.3. S0 LIN Bus Register

S0CFG1: Serial Port 0 Configuration Register 1

SFR Page = 0 **only**

SFR Address = **0x9D** RESET = 0000-00xx

7	6	5	4	3	2	1	0
SBF0	TXER0	S0SB16	ATBR0	TXRX0	SYNC0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**W**	**W**

Bit 7: SBF0, Sync-Break Flag on S0.
0: Must be cleared by software.
1: Set by hardware at the end of the break event detection on LIN bus. In master mode, it is set combined with TI0 flag. In slave mode, it is set combined with RI0.

Bit 6: TXER0, LIN Transmit Error on S0.
0: Must be cleared by software.
1: In TX mode, set by hardware at the transmit error detection on LIN bus.

Bit 5: S0SB16, Sync-Break 16 Bit enable on S0.
0: Select 13-bit Sync-Break transmitting in master mode.
1: Select 16-bit Sync-Break transmitting in master mode.

Bit 4: ATBR0, Auto Baud Rate on S0.
0: Auto cleared by hardware at the end of SYNC field.
1: Before SYNC field, set by software to perform auto baud rate adjustment on LIN bus SYNC field in slave RX mode.

Bit 3: TXRX0, TX/RX selection on S0 LIN bus.
0: Select the LIN bus interface engine to RX function.
1: Select the LIN bus interface engine to TX function.

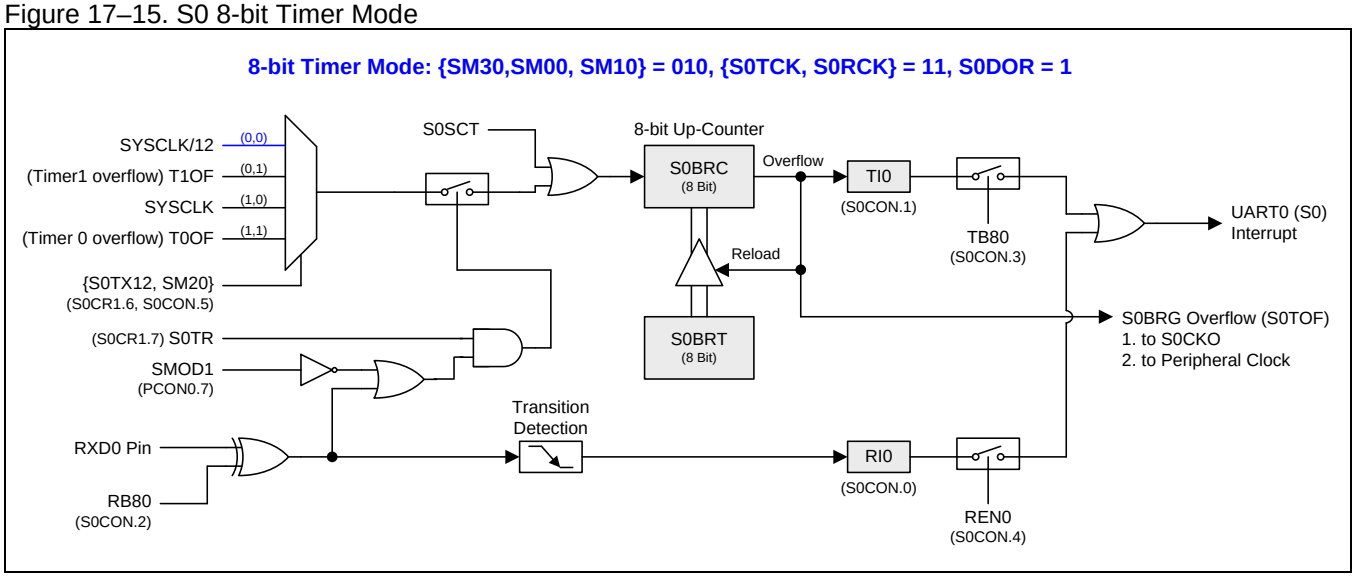
Bit 2: SYNC, Sync-break Control bit on S0.
0: Auto cleared when Sync-Break is sent in master mode or received in slave mode.
1: Set by software. If set in master mode, next writing S0BUF will send a Sync-Break on LIN bus. If set in slave mode, the LIN interface engine will wait to receive a Sync-Break.

Bit 1~0: Reserved. Software must write “0” on these bits when S0CFG1 is written.

17.11.4. S0 acts as 8-bit Timer Mode

S0 8-bit Timer Mode is shown in Figure 17–15.

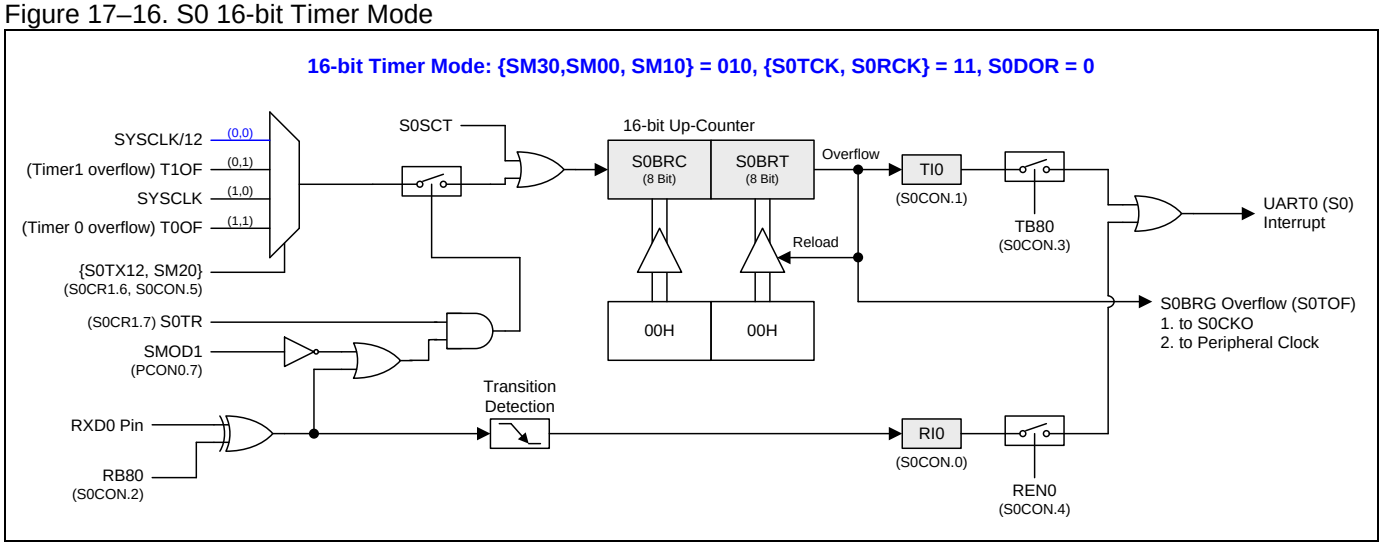
Figure 17–15. S0 8-bit Timer Mode



17.11.5. S0 acts as 16-bit Timer Mode

S0 16-bit Timer Mode is shown in Figure 17–16.

Figure 17–16. S0 16-bit Timer Mode



17.11.6. S0BRG Programmable Clock Output

S0BRG has a clock output mode is shown in Figure 17–17 and Figure 17–18.

Figure 17–17. S0BRG Clock Output (S0BRG in 8-bit Timer Mode)

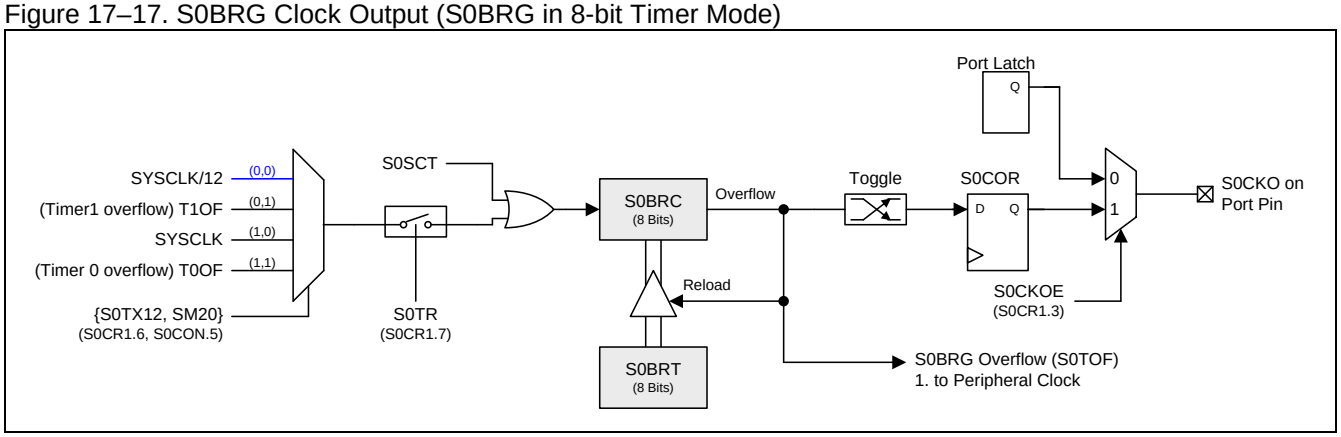
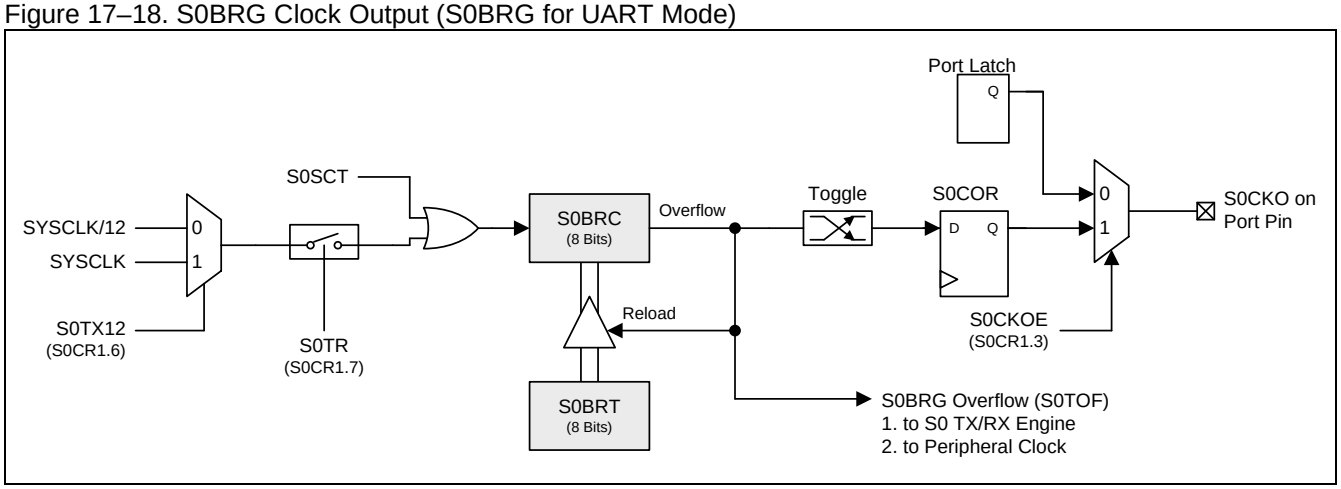


Figure 17–18. S0BRG Clock Output (S0BRG for UART Mode)



AUXR6: Auxiliary Register 6

SFR Page = **3 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	T2FCS	SnMIPS	S0COPS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

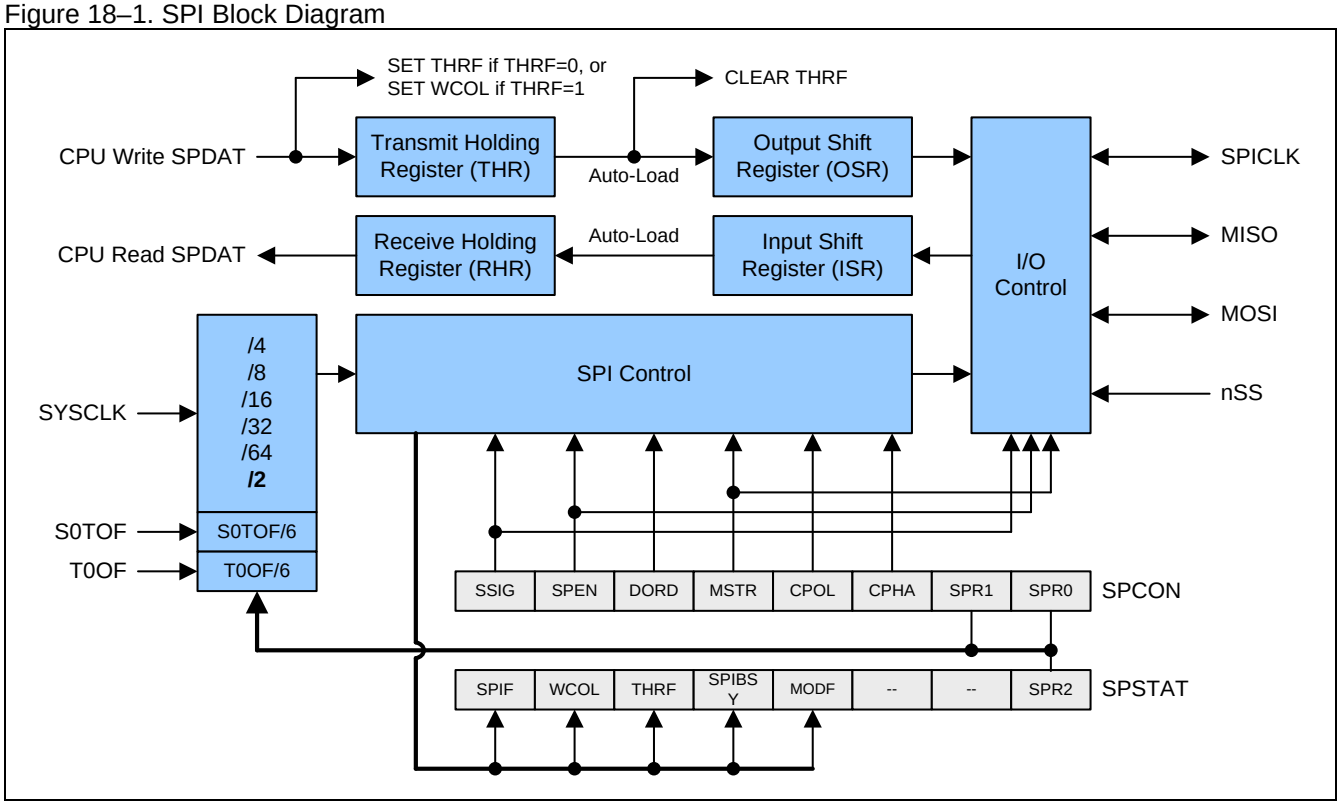
Bit 0: S0COPS, S0BRG Clock Output (S0CKO) port pin Selection.

S0COPS	S0CKO
0	P4.7
1	P4.4

18. Serial Peripheral Interface (SPI)

The **MG82F6B08 / 6B001/ 6B104** provides a high-speed serial communication interface, the SPI interface. SPI is a full-duplex, high-speed and synchronous communication bus with two operation modes: Master mode and Slave mode. Up to **12** Mbps can be supported in Master or **6**MHz in Slave mode under a 24MHz system clock. It has a Transfer Completion Flag (SPIF), Write Collision Flag (WCOL) and Mode Fault flag (MODF) in the SPI status register (SPSTAT). And a specially designed Transmit Holding Register (THR) improves the transmit performance compared to the conventional SPI and THRF flag indicates the THR is full or empty. SPIBSY read-only flag reports the Busy state in SPI engine.

Figure 18–1. SPI Block Diagram



The SPI interface has four pins: MISO, MOSI, SPICLK and nSS:

• SPICLK, MOSI and MISO are typically tied together between two or more SPI devices. Data flows from master to slave on the MOSI pin (Master Out / Slave In) and flows from slave to master on the MISO pin (Master In / Slave Out). The SPICLK signal is output in the master mode and is input in the slave mode. If the SPI system is disabled, i.e., SPEN (SPCTL.6) = 0, these pins function as normal I/O pins.

• nSS (/SS) is the optional slave select pin. In a typical configuration, an SPI master asserts one of its port pins to select one SPI device as the current slave. An SPI slave device uses its nSS pin to determine whether it is selected. The nSS is ignored if any of the following conditions are true:

- If the SPI system is disabled, i.e. SPEN (SPCTL.6) = 0 (reset value).
- If the SPI is configured as a master, i.e., MSTR (SPCTL.4) = 1, and nSS GPIO is configured as an output.
- If the nSS pin is ignored, i.e. SSIG (SPCTL.7) bit = 1, this pin is configured for port functions.

Note: See the AUXR10 in Section "4.3 Alternate Function Redirection", for its alternate pin-out option.

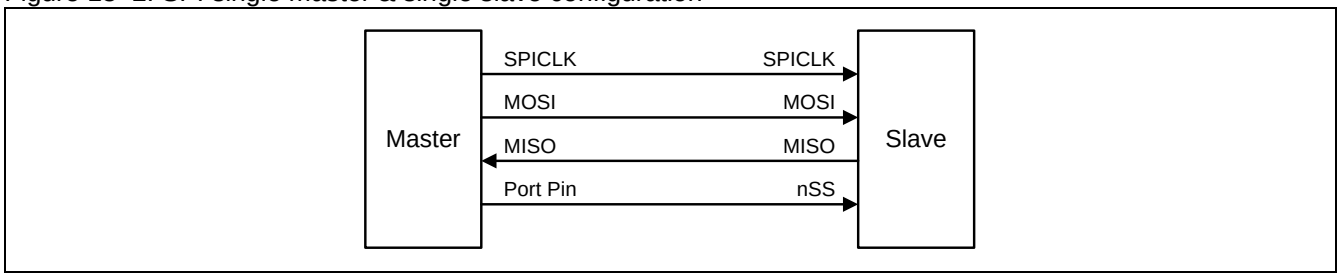
Note that even if the SPI is configured as a master (MSTR=1), it can still be converted to slave mode by the logic low of nSS pin input (if SSIG=0). Should this happen, the SPIF bit (SPSTAT.7) will be set and SPEN will be cleared. (See Section "18.2.3 Mode Change on nSS-pin")

18.1. Typical SPI Configurations

18.1.1. Single Master & Single Slave

For the master: any port pin, including nSS GPIO, can be used to drive the nSS pin of the slave.
For the slave: SSIG is '0', and nSS pin is used to determine whether it is selected.

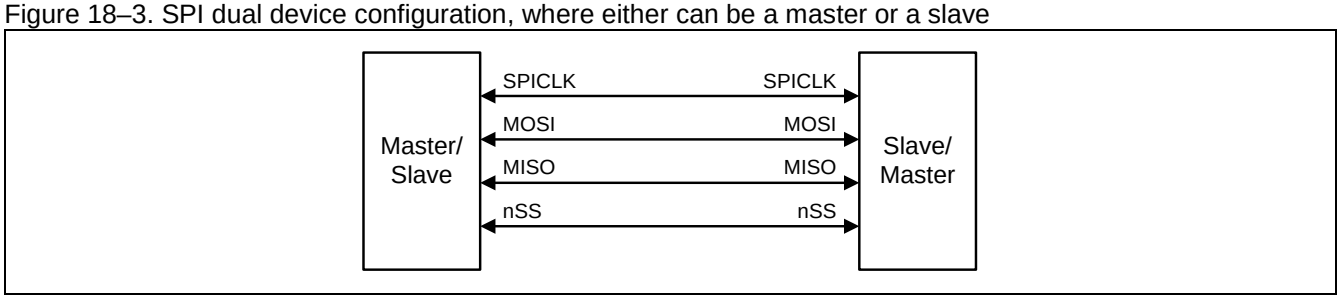
Figure 18–2. SPI single master & single slave configuration



18.1.2. Dual Device, where either can be a Master or a Slave

Two devices are connected to each other and either device can be a master or a slave. When no SPI operation is occurring, both can be configured as masters with MSTR=1, SSIG=0 and nSS port pin configured in quasi-bidirectional mode or in open-drain mode with pull-up resistor. When any device initiates a transfer, it can configure nSS port pin as an output and drive it low to force a "mode change to slave" in the other device. (See Section "18.2.3 Mode Change on nSS-pin")

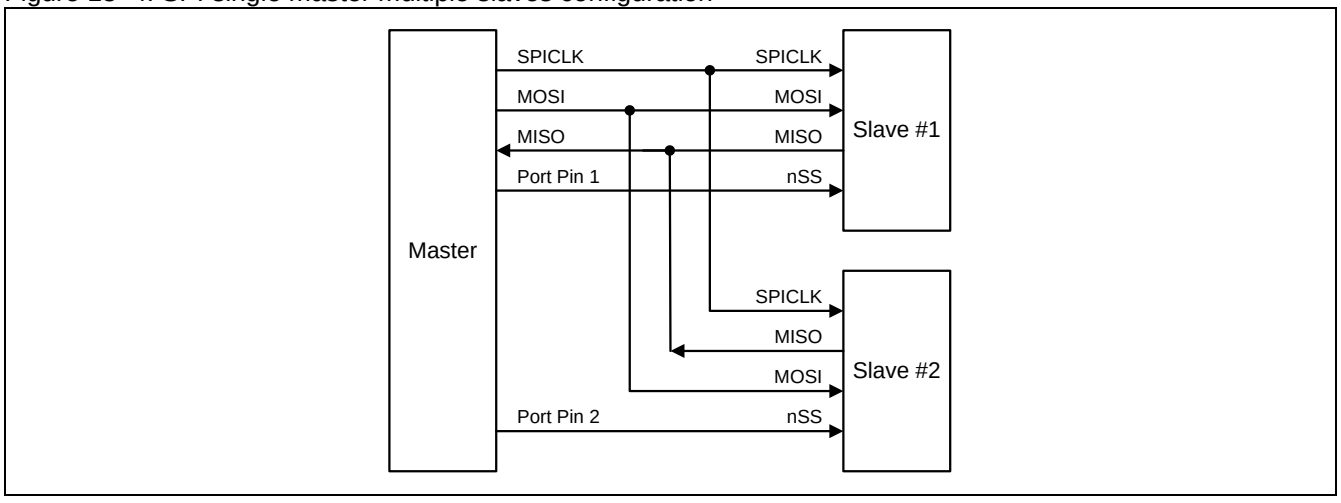
Figure 18–3. SPI dual device configuration, where either can be a master or a slave



18.1.3. Single Master & Multiple Slaves

For the master: any port pin, including nSS GPIO, can be used to drive the nSS pins of the slaves. For all the slaves: SSIG is '0', and nSS pin are used to determine whether it is selected.

Figure 18–4. SPI single master multiple slaves configuration



18.2. Configuring the SPI

Table 18–1 shows configuration for the master/slave modes as well as usages and directions for the modes.

Table 18–1. SPI Master and Slave Selection

SPEN (SPCON.6)	SSIG (SPCON.7)	nSS -pin	MSTR (SPCON.4)	Mode	MISO -pin	MOSI -pin	SPICLK -pin	Remarks
0	X	X	X	SPI disabled	input	input	input	SPI assigned port pint are used as general port pins.
1	0	0	0	Salve (selected)	output	input	input	Selected as slave.
1	0	1	0	Slave (not selected)	Hi-Z	input	input	Not selected.
1	0	0	1 → 0	Slave (by mode change)	output	input	input	Mode change to slave if nSS pin is driven low, then MSTR will be cleared to '0' by H/W automatically, and SPEN is cleared, MODF is set.
1	0	1	1	Master (idle)	input	Hi-Z	Hi-Z	MOSI and SPICLK are at high impedance to avoid bus contention when the Master is idle.
				Master (active)		output	output	MOSI and SPICLK are push-pull when the Master is active.
1	1	X	0	Slave	output	input	input	
1	1	X	1	Master	input	output	output	

"X" means "don't care".

18.2.1. Additional Considerations for a Slave

When CPHA is 0, SSIG must be 0 and nSS pin must be negated and reasserted between each successive serial byte transfer. Note the SPDAT register cannot be written while nSS pin is active (low), and the operation is undefined if CPHA is 0 and SSIG is 1.

When CPHA is 1, SSIG may be 0 or 1. If SSIG=0, the nSS pin may remain active low between successive transfers (can be tied low at all times). This format is sometimes preferred for use in systems having a single fixed master and a single slave configuration.

18.2.2. Additional Considerations for a Master

In SPI, transfers are always initiated by the master. If the SPI is enabled (SPEN=1) and selected as master, writing to the SPI data register (SPDAT) by the master starts the SPI clock generator and data transfer. The data will start to appear on MOSI about one half SPI bit-time to one SPI bit-time after data is written to SPDAT.

Before starting the transfer, the master may select a slave by driving the nSS pin of the corresponding device low. Data written to the SPDAT register of the master is shifted out of MOSI pin of the master to the MOSI pin of the slave. And, at the same time the data in SPDAT register of the selected slave is shifted out on MISO pin to the MISO pin of the master.

After shifting one byte, the SPI clock generator stops, setting the transfer completion flag (SPIF) and an interrupt will be created if the SPI interrupt is enabled. The two shift registers in the master CPU and slave CPU can be considered as one distributed 16-bit circular shift register. When data is shifted from the master to the slave, data is also shifted in the opposite direction simultaneously. This means that during one shift cycle, data in the master and the slave are interchanged.

18.2.3. Mode Change on nSS-pin

If SPEN=1, SSIG=0, MSTR=1 and nSS pin=1, the SPI is enabled in master mode. In this case, another master can drive this pin low to select this device as an SPI slave and start sending data to it. To avoid bus contention, the SPI becomes a slave. As a result of the SPI becoming a slave, the MOSI and SPICLK pins are forced to be an input and MISO becomes an output. The SPIF flag in SPSTAT is set, and if the SPI interrupt is enabled, an SPI interrupt will occur. User software should always check the MSTR bit. If this bit is cleared by a slave select and the user wants to continue to use the SPI as a master, the user must set the MSTR bit again, otherwise it will stay in slave mode.

18.2.4. Transmit Holding Register Full Flag

To speed up the SPI transmit performance, a specially designed Transmit Holding Register (THR) improves the latency time between byte to byte transmitting in CPU data moving. And a set THR-Full flag, THRF (SPSTAT.5), indicates the data in THR is valid and waiting for transmitting. If THR is empty (THRF=0), software writes one byte data to SPDAT will store the data in THR and set the THRF flag. If Output Shift Register (OSR) is empty, hardware will move THR data into OSR immediately and clear the THRF flag. In SPI mater mode, valid data in OSR triggers a SPI transmit. In SPI slave mode, valid data in OSR is waiting for another SPI master to shift out the data. If THR is full (THRF=1), software writes one byte data to SPDAT will set a write collision flag, WCOL (SPSTAT.6).

18.2.5. Write Collision

The SPI in **MG82F6B08 / 6B001/ 6B104** is double buffered data both in the transmit direction and in the receive direction. New data for transmission cannot be written to the THR until the THR is empty. The read-only flag, THRF, indicates the THR is full or empty. The WCOL (SPSTAT.6) bit is set to indicate data collision when the data register is written during set THRF. In this case, the SPDAT writing operation is ignored.

While write collision is detected for a master or a slave, it is uncommon for a master because the master has full control of the transfer in progress. The slave, however, has no control over when the master will initiate a transfer and therefore collision can occur.

WCOL can be cleared in software by writing ‘1’ to the bit.

18.2.6. SPI Clock Rate Select

The SPI clock rate selection (in master mode) uses the SPR1 and SPR0 bits in the SPCON register and SPR2 in the SPSTAT register, as shown in Table 18–2.

Table 18–2. SPI Serial Clock Rates

SPR2	SPR1	SPR0	SPI Clock Selection	SPI Clock Rate @ SYSCLK= 16 MHz	SPI Clock Rate @ SYSCLK=24MHz
0	0	0	SYSCLK/4	4 MHz	6 MHz
0	0	1	SYSCLK/8	2 MHz	3 MHz
0	1	0	SYSCLK/16	1 KHz	1.5 MHz
0	1	1	SYSCLK/32	500 KHz	750 KHz
1	0	0	SYSCLK/64	250 KHz	375 KHz
1	0	1	SYSCLK/2	8 MHz	12 MHz
1	1	0	**S0TOF/6**	Variable	Variable
1	1	1	**T0OF/6**	Variable	Variable

Note:

1. *SYSCLK is the system clock.*
2. *S0TOF is UART0 Baud-Rate Generator Overflow.*
3. *T0OF is Timer 0 Overflow.*

18.3. Data Mode

Clock Phase Bit (CPHA) allows the user to set the edges for sampling and changing data. The Clock Polarity bit, CPOL, allows the user to set the clock polarity. The following figures show the different settings of Clock Phase Bit, CPHA.

Table 18–3. SPI mode definition

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

Figure 18–5. SPI Slave Transfer Format with CPHA=0

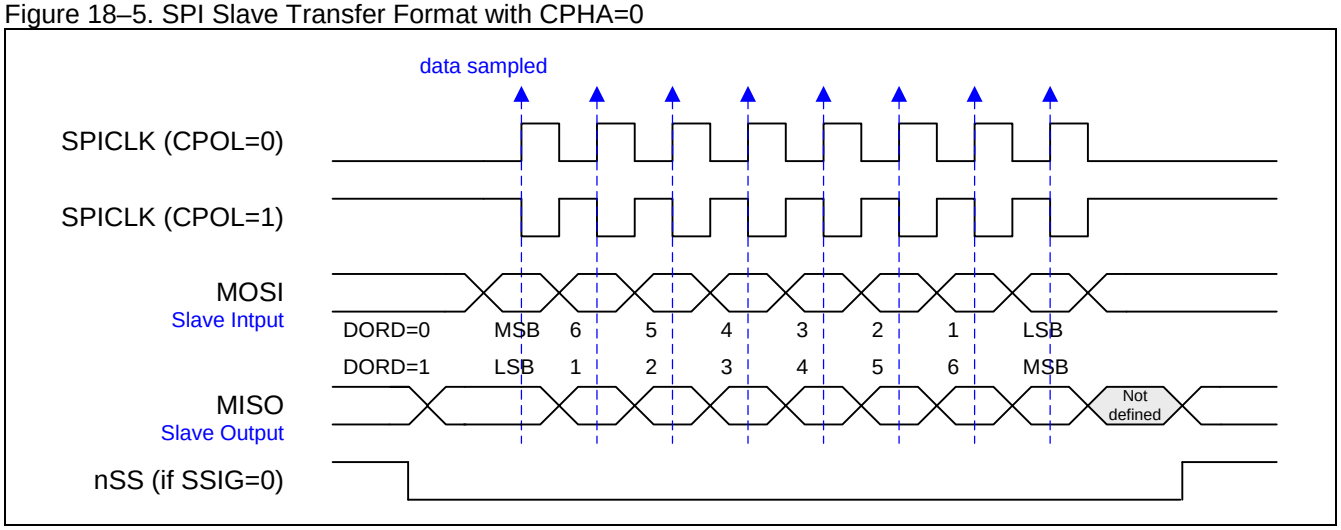


Figure 18–6. Slave Transfer Format with CPHA=1

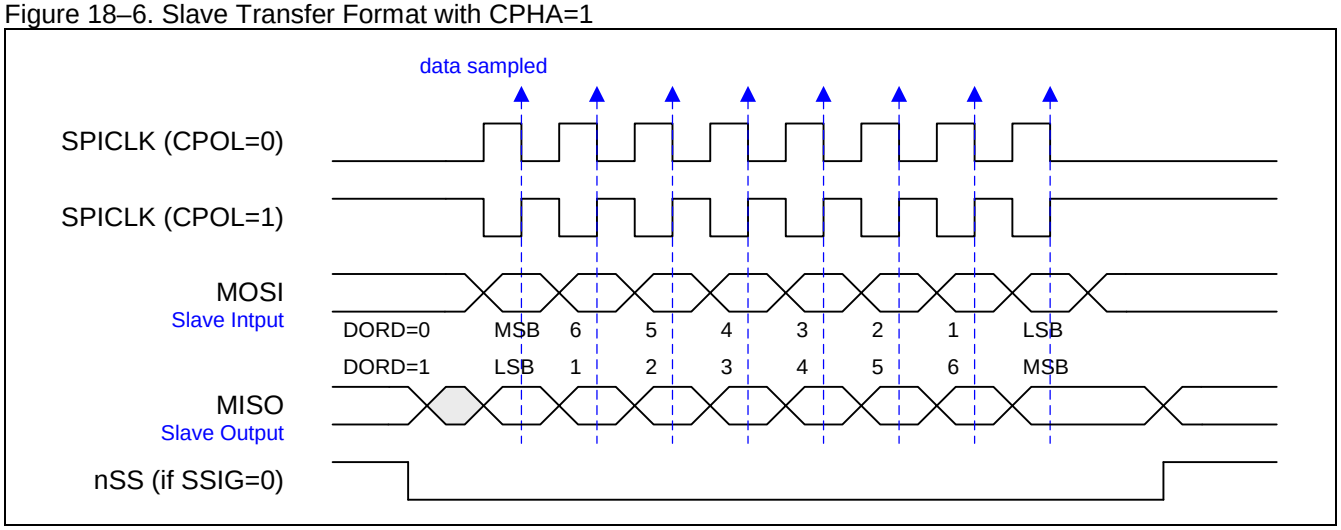


Figure 18–7. SPI Master Transfer Format with CPHA=0

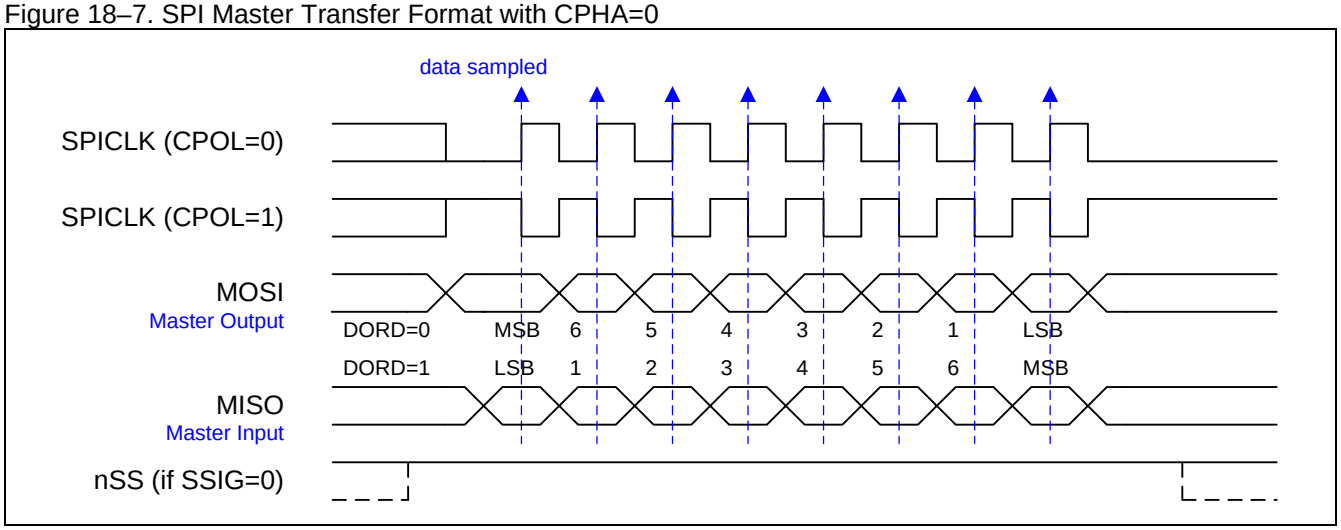
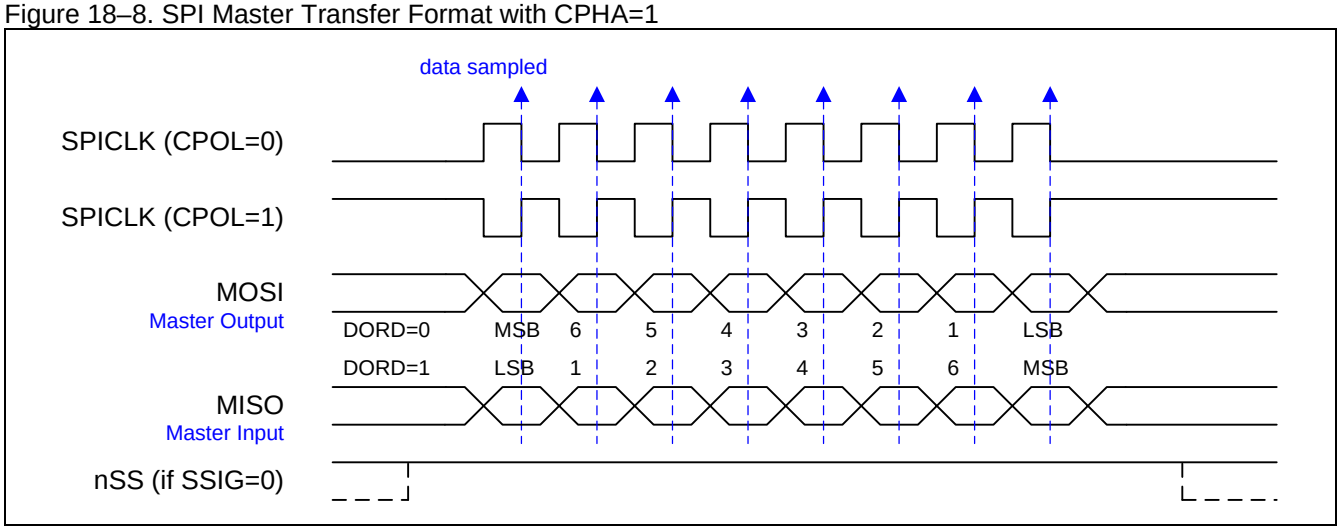


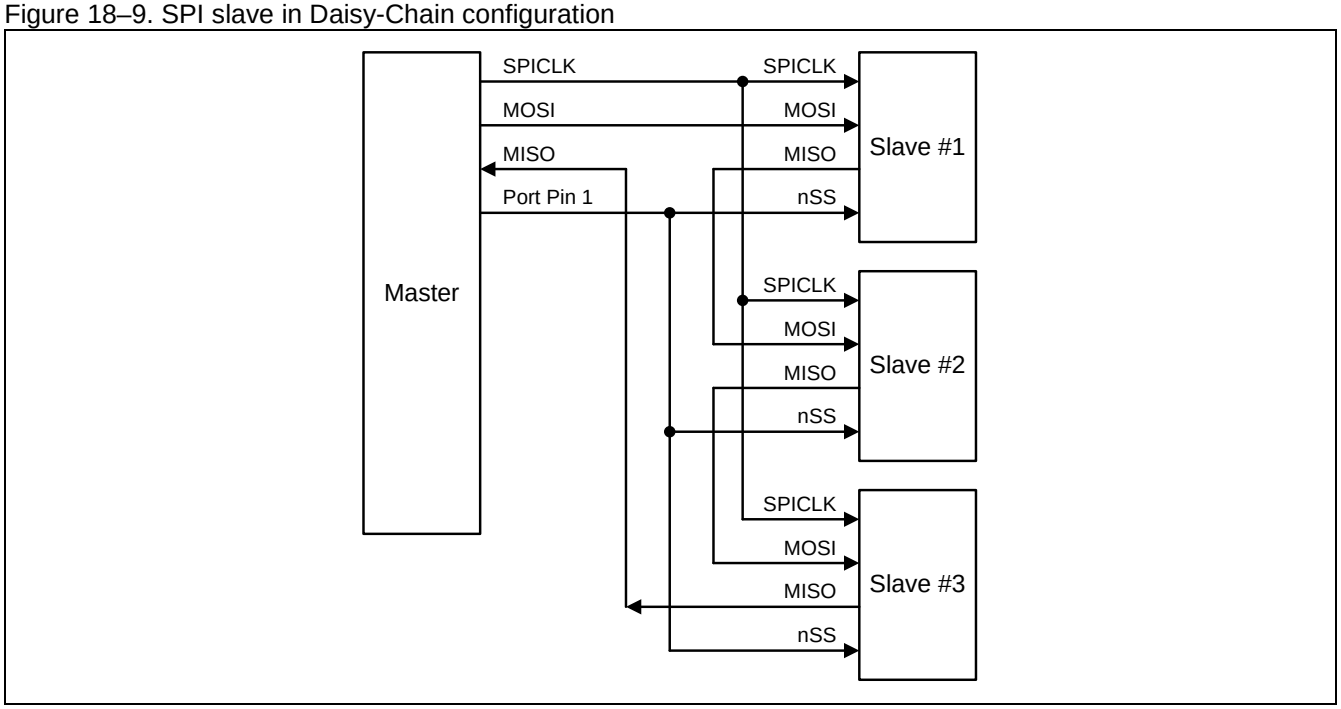
Figure 18–8. SPI Master Transfer Format with CPHA=1



18.4. Daisy-Chain Connection

If SPI0 is defined in slave mode, it can be connected in a daisy-chain configuration. The first slave output is connected to the second slave input, the second slave output is connected to the third slave input, and so on. The final slave output is connected to the master input. Each slave sends out, during a second group of clock pulses, an exact copy of what was received during the first group of clock pulses. The whole chain acts as one large communication shift register. The daisy-chain feature only requires a single Slave Select line (nSS) from the master device.

Figure 18–9. SPI slave in Daisy-Chain configuration



18.4.1. Configuring the Daisy-Chain

How to Configure SPI Slave in Daisy-Chain

- Configure SPCON to define the data mode and select SPI0 in slave mode.
- Set SPI0M0 (AUXR7.4) to enable SPI0 in Daisy-Chain mode.
- Service SPIF to get daisy-chain communication.

18.5. SPI Register

The following special function registers are related to the SPI operation:

SPCON: SPI Control Register

SFR Page = 0~F

SFR Address = 0x85 RESET = 0000-0100

7	6	5	4	3	2	1	0
SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: SSIG, nSS is ignored.
0: The nSS pin decides whether the device is a master or slave.
1: MSTR decides whether the device is a master or slave.

Bit 6: SPEN, SPI enable.
0: The SPI interface is disabled and all SPI pins will be general-purpose I/O ports.
1: The SPI is enabled.

Bit 5: DORD, SPI data order.
0: The MSB of the data byte is transmitted first.
1: The LSB of the data byte is transmitted first.

Bit 4: MSTR, Master/Slave mode select
0: Selects slave SPI mode.
1: Selects master SPI mode.

Bit 3: CPOL, SPI clock polarity select
0: SPICLK is low when Idle. The leading edge of SPICLK is the rising edge and the trailing edge is the falling edge.
1: SPICLK is high when Idle. The leading edge of SPICLK is the falling edge and the trailing edge is the rising edge.

Bit 2: CPHA, SPI clock phase select
0: Data is driven when nSS pin is low (SSIG=0) and changes on the trailing edge of SPICLK. Data is sampled on the leading edge of SPICLK.
1: Data is driven on the leading edge of SPICLK, and is sampled on the trailing edge.
(Note: If SSIG=1, CPHA must not be 1, otherwise the operation is not defined.)

Bit 1~0: SPR1-SPR0, SPI clock rate select 0 & 1 (associated with SPR2, when in master mode)

SPR2	SPR1	SPR0	SPI Clock Selection	SPI Clock Rate @ SYSCLK= 16 MHz	SPI Clock Rate @ SYSCLK=24MHz
0	0	0	SYSCLK/4	4 MHz	6 MHz
0	0	1	SYSCLK/8	2 MHz	3 MHz
0	1	0	SYSCLK/16	1 KHz	1.5 MHz
0	1	1	SYSCLK/32	500 KHz	750 KHz
1	0	0	SYSCLK/64	250 KHz	375 KHz
1	**0**	**1**	**SYSCLK/2**	8 MHz	12 MHz
1	1	0	**S0TOF/6**	Variable	Variable
1	1	1	**T0OF/6**	Variable	Variable

Note:

1. SYSCLK is the system clock.
2. S0TOF is UART0 Baud-Rate Generator Overflow.
3. T0OF is Timer 0 Overflow.

SPSTAT: SPI Status Register

SFR Page = 0~F

SFR Address = 0x84 RESET = 0000-XXX0

7	6	5	4	3	2	1	0
SPIF	WCOL	THRF	SPIBSY	MODF	0	0	SPR2
R/W	R/W	R	R	R/W	R/W	R/W	R/W

Bit 7: SPIF, SPI transfer completion flag

0: **The SPIF is cleared in software by writing "1" to this bit.**

1: When a serial transfer finishes, the SPIF bit is set and an interrupt is generated if SPI interrupt is enabled. If nSS pin is driven low when SPI is in master mode with SSIG=0, SPIF will also be set to signal the "mode change".

Bit 6: WCOL, SPI write collision flag.

0: The WCOL flag is cleared in software by writing "1" to this bit.

1: The WCOL bit is set if the SPI data register, SPDAT, is written during a data transfer (see Section "18.2.5 Write Collision").

Bit 5: THRF, Transmit Holding Register (THR) Full flag. Read only.

0: Means the THR is "empty". This bit is cleared by hardware when the THR is empty. That means the data in THR is loaded (by H/W) into the Output Shift Register to be transmitted, and now the user can write the next data byte to SPDAT for next transmission.

1: Means the THR is "full". This bit is set by hardware just when SPDAT is written by software.

Bit 4, SPIBSY, SPI Busy flag. Read only.

0: It indicates SPI engine is idle and all shift registers are empty.

1: It is set to logic 1 when a SPI transfer is in progress (Master or slave Mode).

Bit 3: Mode Fault Flag. This bit is set to logic 1 by hardware when a master mode collision is detected (nSS is low, MSTEN = 1, and SSIG = 0). If SPI interrupts are enabled, an interrupt will be generated. This bit is not automatically cleared by hardware, and must be cleared by software writing "1".

Bit 2 ~ 1: Reserved. Software must write "0" on this bit when SPSTAT is written.

Bit 0: SPR2, SPI clock rate select 2 (associated with SPR1 and SPR0).

SPDAT: SPI Data Register

SFR Page = 0~F

SFR Address = 0x86 RESET = 0000-0000

7	6	5	4	3	2	1	0
(MSB)							(LSB)
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SPDAT has two physical buffers for writing to and reading from during transmit and receive, respectively.

AUXR7: Auxiliary Register 7

SFR Page = **4 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	1	C0CKOE	SPI0M0	0	0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 4: SPI0M0, SPI0 model control bit 0. It controls the SPI application with daisy-chain connection.

0: Disable the mode control.

1: Enable the mode control.

AUXR10: Auxiliary Register 10

SFR Page = **7 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	AC0HC0	0	SPIPS0	S0PS1	SPFACE	TWICF	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: SPIPS0, SPI Port pin Selection.

SPIPS0	nSS	MOSI	MISO	SPICLK
0	P4.6	P4.4	P4.5	P3.3
1	P3.0	P4.4	P3.1	P3.3

Bit 2: SPFACE, SPIF Auto-Cleared Enable.

0: Disable, SPIF is only cleared by CPU software.

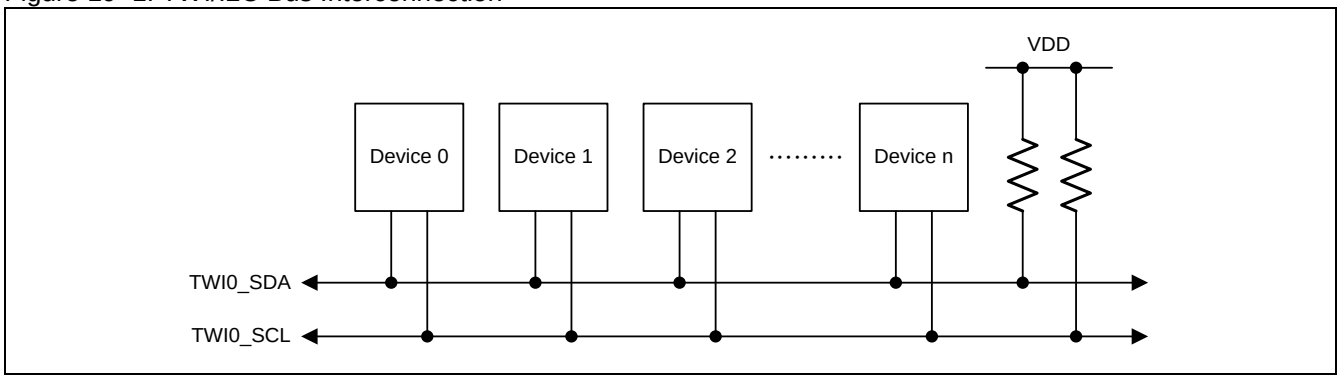
1: Enable. SPIF is also cleared by CPU read/write SPDAT operation.

19. Two Wire serial Interface (TWI0/I2C0)

The Two-Wire serial Interface is a two-wire, bi-directional serial bus. It is ideally suited for typical microcontroller applications. There are one TWI/I2C embedded in the **MG82F6B08 / 6B001/ 6B104**, TWI0/I2C0. It support the multiple slave address recognition.

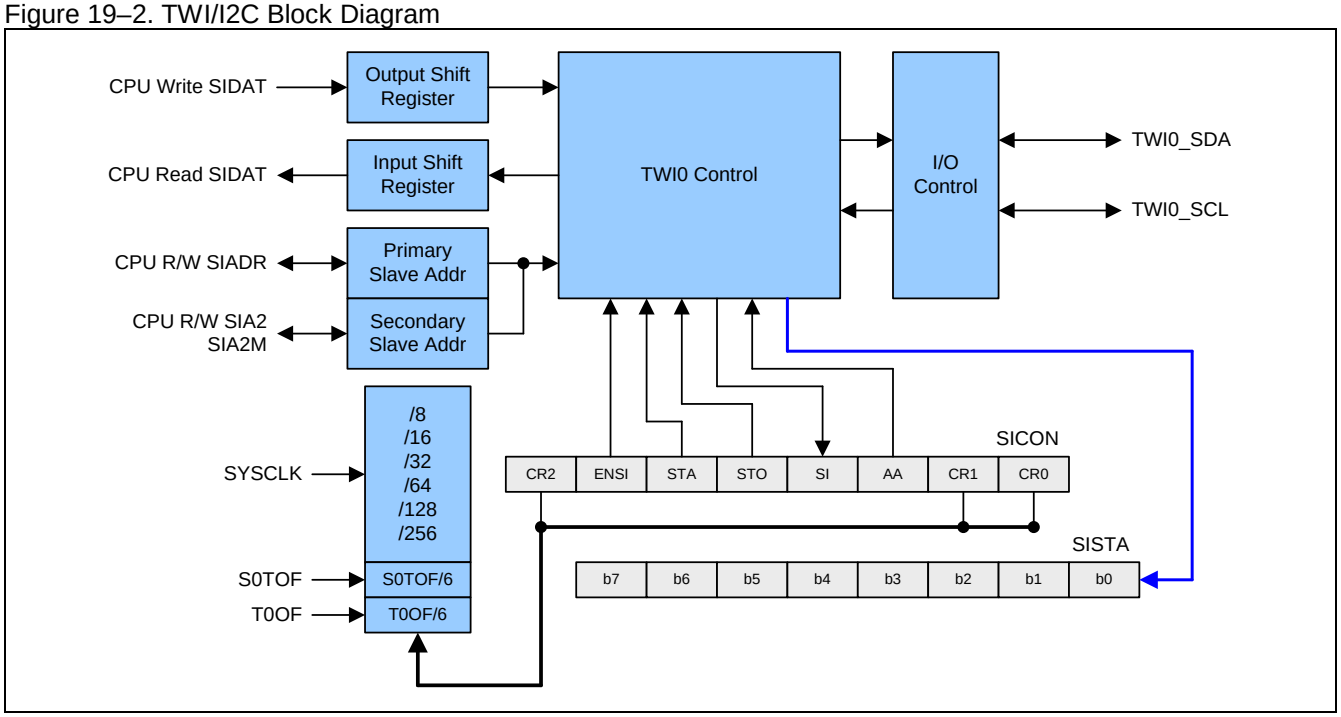
The TWI/I2C protocol allows the systems designer to interconnect up to 128 different devices using only two bi-directional bus lines, one for clock (TWI0_SCL) and one for data (TWI0_SDA). The TWI bus provides control of TWI0_SDA (serial data), TWI0_SCL (serial clock) generation and synchronization, arbitration logic, and START/STOP control and generation. The only external hardware needed to implement this bus is a single pull-up resistor for each of the TWI bus lines. All devices connected to the bus have individual addresses, and mechanisms for resolving bus contention are inherent in the TWI/I2C protocol.

Figure 19–1. TWI/I2C Bus Interconnection



The TWI/I2C bus may operate as a master and/or slave, and may function on a bus with multiple masters. The CPU interfaces to the TWI/I2C through the following four special function registers: SICON configures the TWI/I2C bus; SISTA reports the status code of the TWI/I2C bus; and SIDAT is the data register, used for both transmitting and receiving TWI/I2C data. SIADR and SIA2 are the slave address registers. And, the TWI/I2C hardware interfaces to the serial bus via two lines: SDA (serial data line) and SCL (serial clock line).

Figure 19–2. TWI/I2C Block Diagram



19.1. Operating Modes

There are four operating modes for the TWI/I2C: 1) Master/Transmitter mode, 2) Master/Receiver mode, 3) Slave/Transmitter mode and 4) Slave/Receiver mode. Bits STA, STO and AA in SICON decide the next action which the TWI hardware will take after SI is cleared by software. When the next action is completed, a new status code in SISTA will be updated and SI will be set by hardware in the same time. Now, the interrupt service routine is entered (if the TWI/I2C interrupt is enabled), and the new status code can be used to determine which appropriate routine the software is to branch to.

19.1.1. Master Transmitter Mode

In the master transmitter mode, a number of data bytes are transmitted to a slave receiver. Before the master transmitter mode can be entered, SICON must be initialized as follows:

SICON

7	6	5	4	3	2	1	0
CR2	ENSI	STA	STO	SI	AA	CR1	CR0
Bit rate	1	0	0	0	x	Bit rate	

CR0, CR1, and CR2 define the serial bit rate. ENSI must be set to logic 1 to enable TWI/I2C. If the AA bit is reset, TWI/I2C will not acknowledge its own slave address or the general call address in the event of another device becoming master of the bus. In other words, if AA is reset, TWI/I2C cannot enter a slave mode. STA, STO, and SI must be reset.

The master transmitter mode may now be entered by software setting the STA bit. The TWI/I2C logic will now test the serial bus and generate a START condition as soon as the bus becomes free. When a START condition is transmitted, the serial interrupt flag (SI) is set, and the status code in the status register (SISTA) will be 08H. This status code must be used to vector to an interrupt service routine that loads SIDAT with the slave address and the data direction bit (SLA+W). The SI bit in SICON must then be reset before the serial transfer can continue.

When the slave address and the direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in SISTA are possible. There are 18H, 20H, or 38H for the master mode and also 68H, 78H, or B0H if the slave mode was enabled (AA=1). The appropriate action to be taken for each of these status codes is detailed in the following operating flow chart. After a repeated START condition (state 10H), TWI/I2C may switch to the master receiver mode by loading SIDAT with SLA+R.

19.1.2. Master Receiver Mode

In the master receiver mode, a number of data bytes are received from a slave transmitter. SICON must be initialized as in the master transmitter mode. When the start condition has been transmitted, the interrupt service routine must load SIDAT with the 7-bit slave address and the data direction bit (SLA+R). The SI bit in SICON must then be cleared before the serial transfer can continue.

When the slave address and the data direction bit have been transmitted and an acknowledgment bit has been received, the serial interrupt flag (SI) is set again, and a number of status codes in SISTA are possible. They are 40H, 48H, or 38H for the master mode and also 68H, 78H, or B0H if the slave mode was enabled (AA=1). The appropriate action to be taken for each of these status codes is detailed in the following operating flow chart. After a repeated start condition (state 10H), TWI/I2C may switch to the master transmitter mode by loading SIDAT with SLA+W.

19.1.3. Slave Transmitter Mode

In the slave transmitter mode, a number of data bytes are transmitted to a master receiver. To initiate the slave transmitter mode, SIADR and SICON must be loaded as follows:

SIADR

7	6	5	4	3	2	1	0
X	X	X	X	X	X	X	GC

|<---------- Own Slave Address ---------->|

The upper 7 bits are the address to which TWI/I2C will respond when addressed by a master. If the LSB (GC) is set, TWI/I2C will respond to the general call address (00H); otherwise it ignores the general call address.

SICON

7	6	5	4	3	2	1	0
CR2	ENSI	STA	STO	SI	AA	CR1	CR0
x	1	0	0	0	1	x	x

CR0, CR1, and CR2 do not affect TWI/I2C in the slave mode. ENSI must be set to “1” to enable TWI/I2C. The AA bit must be set to enable TWI/I2C to acknowledge its own slave address or the general call address. STA, STO, and SI must be cleared to “0”.

When SIADR and SICON have been initialized, TWI/I2C waits until it is addressed by its own slave address followed by the data direction bit which must be “1” (R) for TWI/I2C to operate in the slave transmitter mode. After its own slave address and the “R” bit have been received, the serial interrupt flag (SI) is set and a valid status code can be read from SISTA. This status code is used to vector to an interrupt service routine, and the appropriate action to be taken for each of these status codes is detailed in the following operating flow chart. The slave transmitter mode may also be entered if arbitration is lost while TWI/I2C is in the master mode (see state B0H).

If the AA bit is reset during a transfer, TWI/I2C will transmit the last byte of the transfer and enter state C0H or C8H. TWI/I2C is switched to the not-addressed slave mode and will ignore the master receiver if it continues the transfer. Thus the master receiver receives all 1s as serial data. While AA is reset, TWI/I2C does not respond to its own slave address or a general call address. However, the serial bus is still monitored, and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate TWI/I2C from the bus.

19.1.4. Slave Receiver Mode

In the slave receiver mode, a number of data bytes are received from a master transmitter. Data transfer is initialized as in the slave transmitter mode.

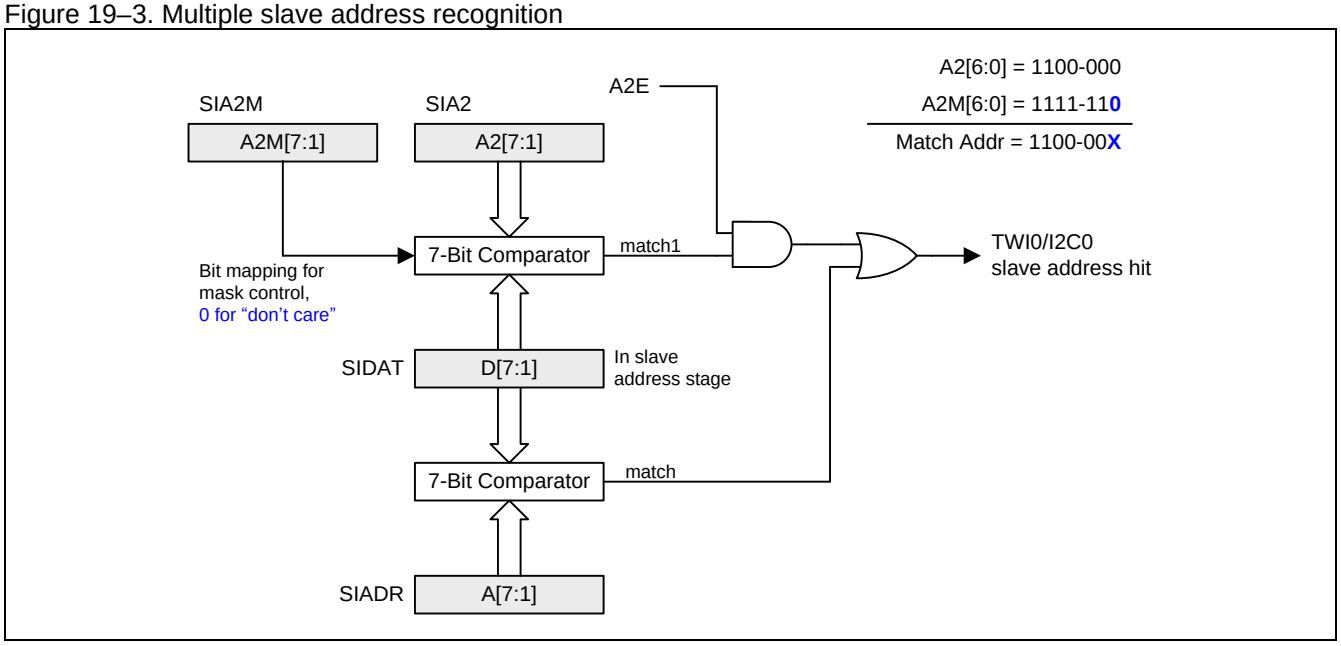
When SIADR and SICON have been initialized, TWI/I2C waits until it is addressed by its own slave address followed by the data direction bit which must be “0” (W) for TWI/I2C to operate in the slave receiver mode. After its own slave address and the W bit have been received, the serial interrupt flag (SI) is set and a valid status code can be read from SISTA. This status code is used to vector to an interrupt service routine, and the appropriate action to be taken for each of these status codes is detailed in the following operating flow chart. The slave receiver mode may also be entered if arbitration is lost while TWI/I2C is in the master mode (see status 68H and 78H).

If the AA bit is reset during a transfer, TWI/I2C will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, TWI/I2C does not respond to its own slave address or a general call address. However, the serial bus is still monitored and address recognition may be resumed at any time by setting AA. This means that the AA bit may be used to temporarily isolate from the bus.

19.1.5. Multiple slave address recognition

SIADR defines the primary slave address in **MG82F6B08 / 6B001/ 6B104** TWI/I2C. **MG82F6B08 / 6B001/ 6B104** also provide the secondary slave address with mask function that is implemented on SIA2 and SIA2M. A 1 in bit positions of the slave address mask SIA2M[7:1] enable a comparison between the received slave address and the secondary hardware's slave address SIA2[7:1] for those bits. A 0 in a bit of the slave address mask means that bit will be treated as a "don't care" for comparison purposes. In this case, either a 1 or a 0 value are acceptable on the incoming slave address.

Figure 19–3. Multiple slave address recognition



19.2. Miscellaneous States

There are two SISTA codes that do not correspond to a defined TWI/I2C hardware state, as described below.

S1STA = F8H:
This status code indicates that no relevant information is available because the serial interrupt flag, SI, is not yet set. This occurs between other states and when TWI/I2C is not involved in a serial transfer.

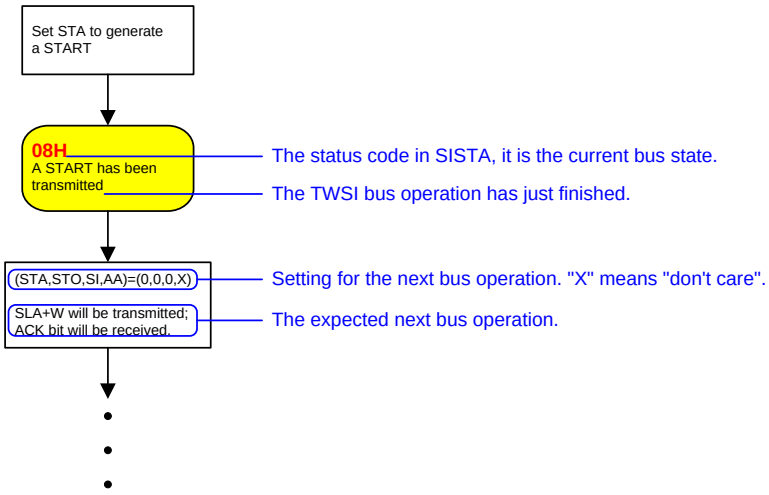
S1STA = 00H:
This status code indicates that a bus error has occurred during a TWI/I2C serial transfer. A bus error is caused when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte, or an acknowledge bit. A bus error may also be caused when external interference disturbs the internal TWI/I2C signals. When a bus error occurs, SI is set. To recover from a bus error, the STO flag must be set and SI must be cleared by software. This causes TWI/I2C to enter the "not-addressed" slave mode (a defined state) and to clear the STO flag (no other bits in SICON are affected). The TWI0_SDA and TWI0_SCL lines are released (a STOP condition is not transmitted).

19.3. Using the TWI/I2C

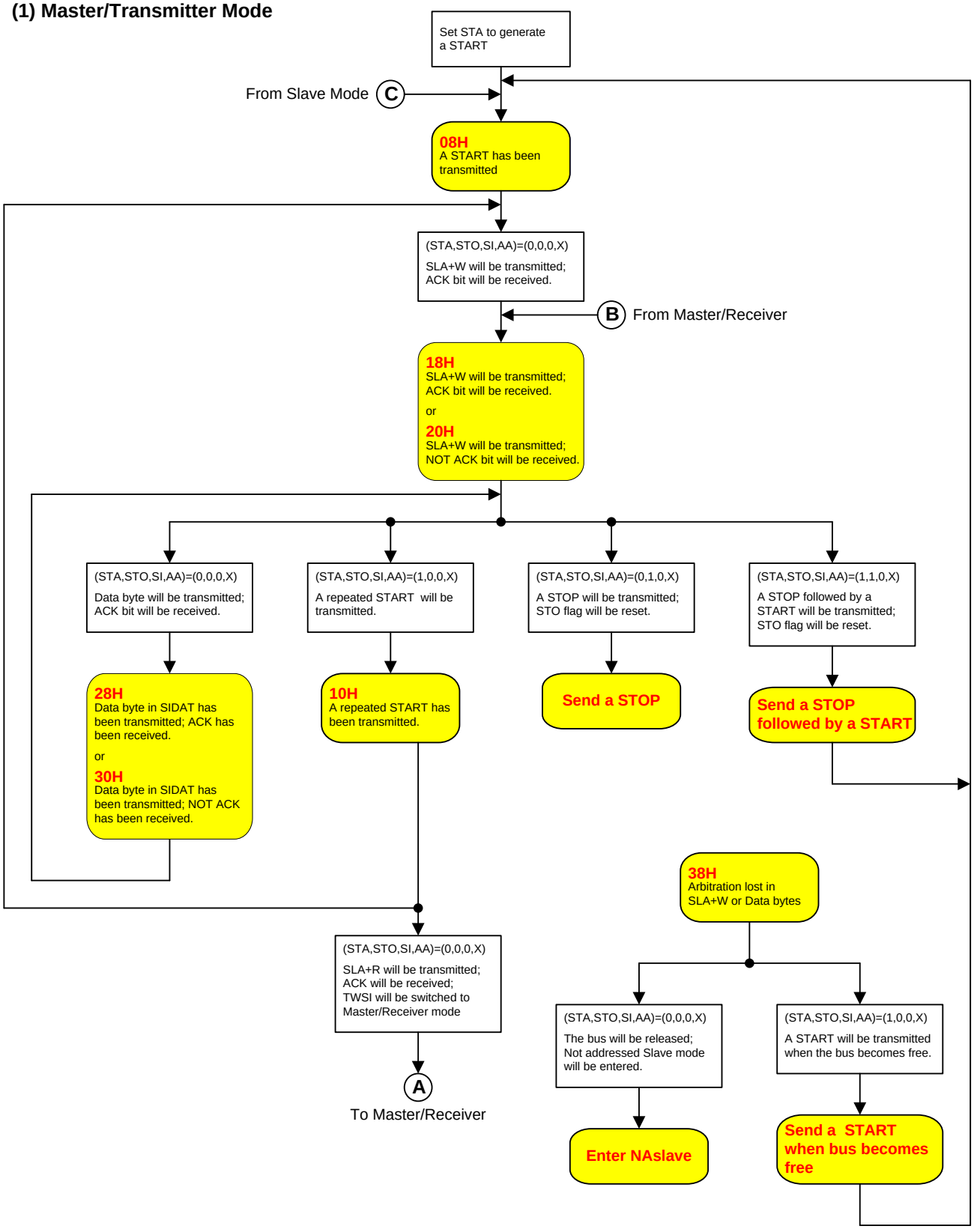
The TWI/I2C is byte-oriented and interrupt based. Interrupts are issued after all bus events, like reception of a byte or transmission of a START condition. Because the TWI/I2C is interrupt-based, the application software is free to carry on other operations during a TWI/I2C byte transfer. Note that the TWI0/I2C0 interrupt enable bit ETWI/I2C0 bit (EIE1.6) together with the EA bit allow the application to decide whether or not assertion of the SI Flag should generate an interrupt request. When the SI flag is asserted, the TWI/I2C has finished an operation and awaits application response. In this case, the status register SISTA contains a status code indicating the current state of the TWI/I2C bus. The application software can then decide how the TWI/I2C should behave in the next TWI/I2C bus operation by properly programming the STA, STO and AA bits (in SICON).

The following operating flow charts will instruct the user to use the TWI/I2C using state-by-state operation. First, the user should fill SIADR with its own Slave address (refer to the previous description about SIADR). To act as a master, after initializing the SICON, the first step is to set “STA” bit to generate a START condition to the bus. To act as a slave, after initializing the SICON, the TWI/I2C waits until it is addressed. And then follow the operating flow chart for a number a next actions by properly programming (STA,STO,SI,AA) in the SICON. Since the TWI/I2C hardware will take next action when SI is just cleared, it is recommended to program (STA,STO,SI,AA) by two steps, first STA, STO and AA, then clear SI bit (may use instruction “CLR SI”) for safe operation. “don’t care”

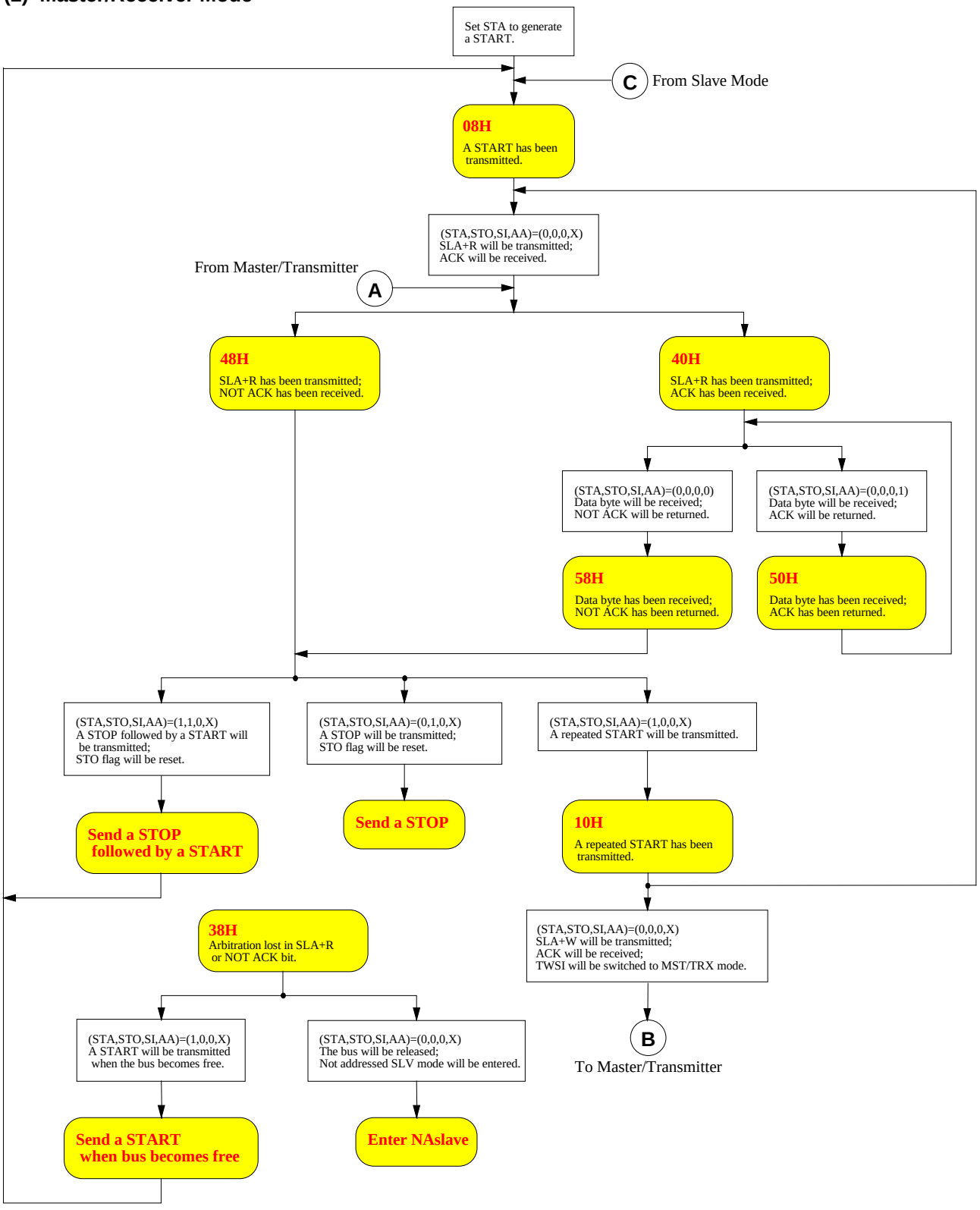
The figure below shows how to read the flow charts.



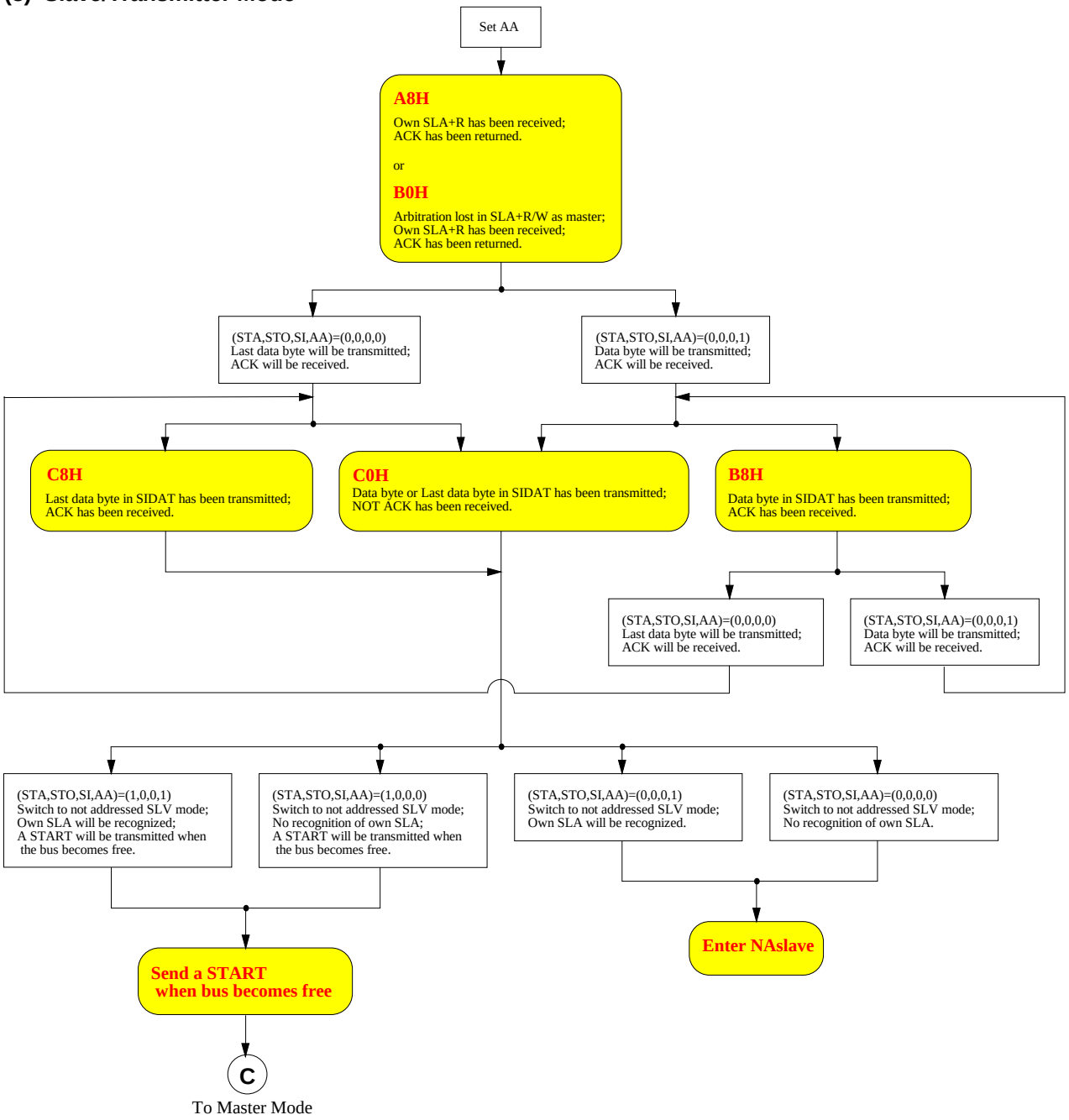
(1) Master/Transmitter Mode



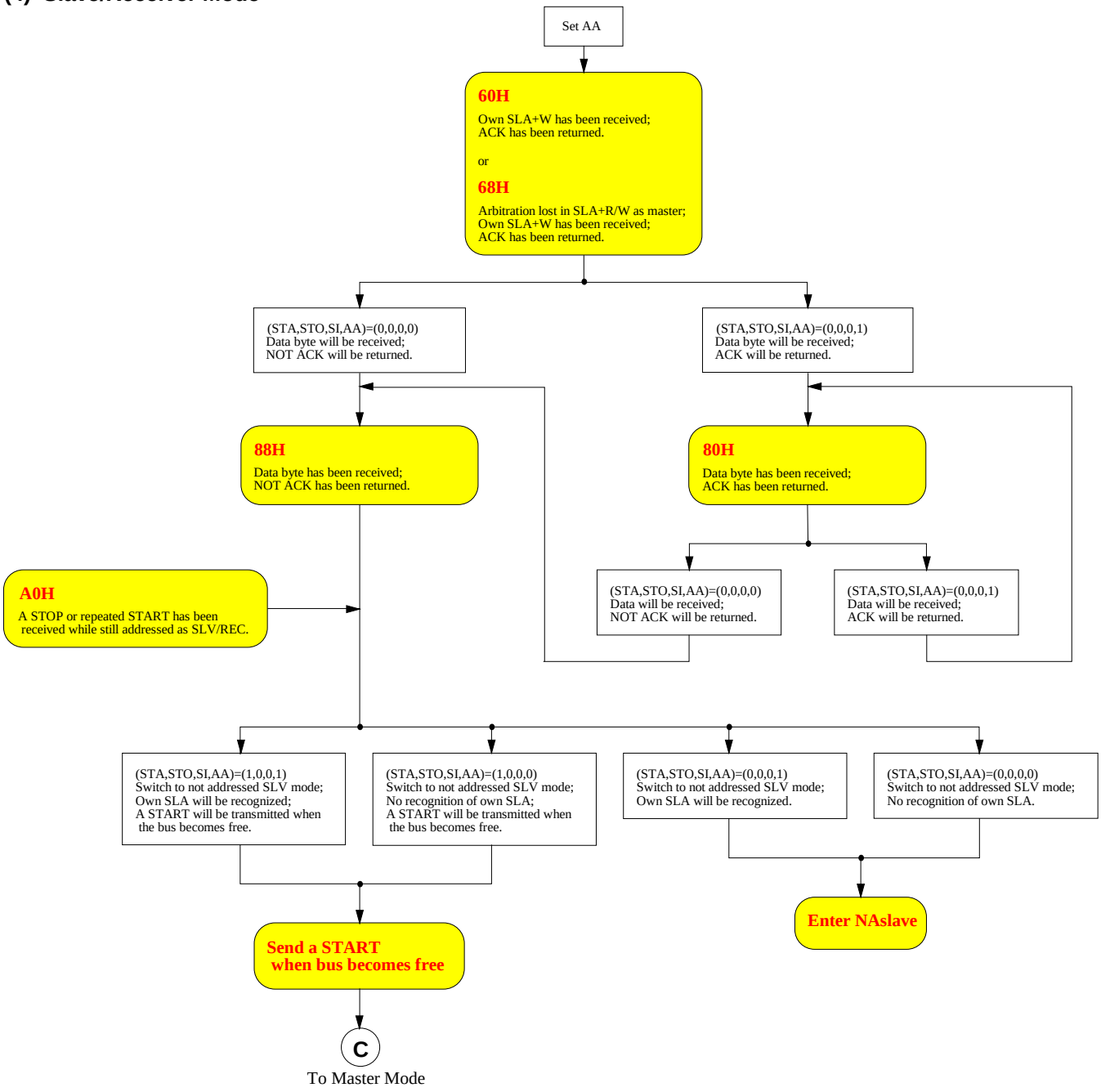
(2) Master/Receiver Mode



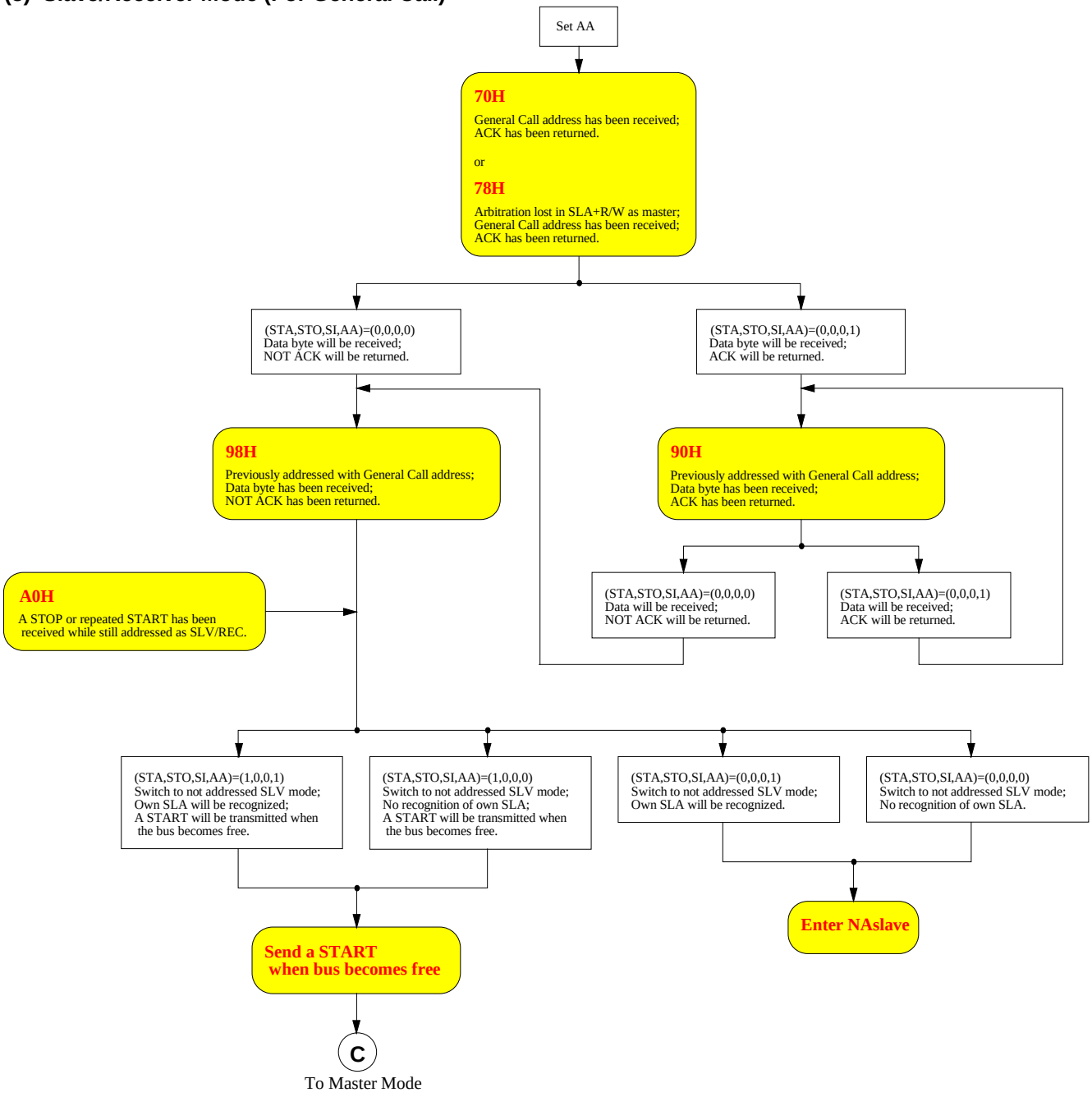
(3) Slave/Transmitter Mode



(4) Slave/Receiver Mode



(5) Slave/Receiver Mode (For General Call)



19.4. TWI0/I2C0 Register

SIADR: TWI0/I2C0 Address Register

SFR Page = **0 Only**

SFR Address = 0xD1 RESET = 0000-0000

7	6	5	4	3	2	1	0
A6	A5	A4	A3	A2	A1	A0	GC
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

The CPU can read from and write to this register directly. SIADR is not affected by the TWI0/I2C0 hardware. The contents of this register are irrelevant when TWI0/I2C0 is in a master mode. In the slave mode, the seven most significant bits must be loaded with the microcontroller's own slave address, and, if the least significant bit (GC) is set, the general call address (00H) is recognized; otherwise it is ignored. The most significant bit corresponds to the first bit received from the TWI0/I2C0 bus after a START condition.

SIDAT: TWI0/I2C0 Data Register

SFR Page = **0 Only**

SFR Address = 0xD2 RESET = 0000-0000

7	6	5	4	3	2	1	0
SD7	SD6	SD5	SD4	SD3	SD2	SD1	SD0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

This register contains a byte of serial data to be transmitted or a byte which has just been received. The CPU can read from or write to this register directly while it is not in the process of shifting a byte. This occurs when TWI0/I2C0 is in a defined state and the serial interrupt flag (SI) is set. Data in SIDAT remains stable as long as SI is set. While data is being shifted out, data on the bus is simultaneously being shifted in; SIDAT always contains the last data byte present on the bus. Thus, in the event of lost arbitration, the transition from master transmitter to slave receiver is made with the correct data in SIDAT.

SIDAT and the ACK flag form a 9-bit shift register which shifts in or shifts out an 8-bit byte, followed by an acknowledge bit. The ACK flag is controlled by the TWI0/I2C0 hardware and cannot be accessed by the CPU. Serial data is shifted through the ACK flag into SIDAT on the rising edges of serial clock pulses on the TWI0_SCL line. When a byte has been shifted into SIDAT, the serial data is available in SIDAT, and the acknowledge bit is returned by the control logic during the 9th clock pulse. Serial data is shifted out from SIDAT on the falling edges of clock pulses on the TWI0_SCL line.

When the CPU writes to SIDAT, the bit SD7 is the first bit to be transmitted to the SDA line. After nine serial clock pulses, the eight bits in SIDAT will have been transmitted to the SDA line, and the acknowledge bit will be present in the ACK flag. Note that the eight transmitted bits are shifted back into SIDAT.

SICON: TWI0/I2C0 Control Register

SFR Page = **0 Only**

SFR Address = 0xD4 RESET = 0000-0000

7	6	5	4	3	2	1	0
CR2	ENSI	STA	STO	SI	AA	CR1	CR0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

The CPU can read and write to this register directly. Two bits are affected by the TWI0/I2C0 hardware: the SI will be set when a serial interrupt occurred, and the STO will be cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENSI="0".

Bit 7: CR2, TWI0/I2C0 Clock Rate select bit 2 (associated with CR1 and CR0).

Bit 6: ENSI, the TWI0/I2C0 Hardware Enable Bit
When ENSI is "0", the TWI0 _SDA and TWI0_SCL outputs are in a high impedance state, and it will ignore the input signals. Under this condition, the TWI0/I2C0 is in the not-addressed slave state, and STO is forced to "0". No other bits are affected, and the TWI0_SDA and TWI0_SCL can be used as general purpose I/O pins. When ENSI is "1", TWI0 is enabled, the TWI0_SDA and TWI0_SCL assign to port pin latch, such as P4.1 and P4.0. The port pin latch must be set to logic 1 and I/O mode must be configured to open-drain mode for the serial communication.

Bit 5: STA, the START Flag
When sets the STA to enter master mode, the TWI0/I2C0 hardware will check the status of the serial bus. It will generate a START condition if the bus is free. Otherwise TWI0/I2C0 will wait for a STOP condition and generates a START condition after a delay. If STA is set while TWI0/I2C0 is already in a master mode and one or more bytes are transmitting or receiving, TWI0/I2C0 will send a repeated START condition. STA may be set at any time. STA may also be set when TWI0/I2C0 is an addressed slave mode. When the STA bit is reset, no START condition or repeated START condition will be generated.

Bit 4: STO, the STOP Flag
When the STO is set while TWI0/I2C0 is in a master mode, a STOP condition is transmitted to the serial bus. When the STOP condition is detected on the bus, the TWI0/I2C0 hardware clears the STO flag. In a slave mode, the STO flag may be set to recover from a bus error condition. In this case, no STOP condition is transmitted to the bus. However, the TWI0/I2C0 hardware behaves as if a STOP condition has been received and switches to the defined not addressed slave receiver mode. The STO flag is automatically cleared by hardware. If the STA and STO bits are both set, then a STOP condition is transmitted to the bus if TWI0/I2C0 is in a master mode (in a slave mode, TWI0/I2C0 generates an internal STOP condition which is not transmitted), and then transmits a START condition.

Bit 3: SI, the Serial Interrupt Flag
When a new TWI0/I2C0 state is present in the SISTA register, the SI flag is set by hardware. And, if the TWI0/I2C0 interrupt is enabled, an interrupt service routine will be serviced. The only state that does not cause SI to be set is state F8H, which indicates that no relevant state information is available. When SI is set, the low period of the serial clock on the TWI0_SCL line is stretched, and the serial transfer is suspended. A high level on the TWI0_SCL line is unaffected by the serial interrupt flag. SI must be cleared by software writing “0” on this bit. When the SI flag is reset, no serial interrupt is requested, and there is no stretching on the serial clock on the TWI0_SCL line.

Bit 2: AA, the Assert Acknowledge Flag
If the AA flag is set to “1”, an Acknowledge (low level to TWI0_SDA) will be returned during the acknowledge clock pulse on the TWI0_SCL line when:
1) The own slave address has been received.
2) A data byte has been received while TWI0/I2C0 is in the master/receiver mode.
3) A data byte has been received while TWI0/I2C0 is in the addressed slave/receiver mode.

If the AA flag is reset to “0”, a not acknowledge (high level to TWI0_SDA) will be returned during the acknowledge clock pulse on TWI0_SCL when:
1) A data has been received while TWI0/I2C0 is in the master/receiver mode.
2) A data byte has been received while TWI0/I2C0 is in the addressed slave/receiver mode.

Bit 7, 1~0: CR2, CR1 and CR0, the Clock Rate select Bits
These three bits determine the serial clock frequency when TWI0/I2C0 is in a master mode. The highest master mode clock frequency is limited to 1MHz. In slave mode, it is no need to select the clock rate. TWI0/I2C0 will automatically synchronize with any clock frequency from master, which is up to 400KHz. The various serial clock rates are shown in Table 19–1.

Table 19–1. TWI0/I2C0 Serial Clock Rates

CR2	CR1	CR0	TWI0/I2C0 Clock Selection	TWI0/I2C0 Clock Rate @ SYSCLK=16MHz
0	0	0	SYSCLK/8	2 MHz Note1
0	0	1	SYSCLK/16	1 MHz
0	1	0	SYSCLK/32	500 KHz
0	1	1	SYSCLK/64	250 KHz
1	0	0	SYSCLK/128	125 KHz
1	0	1	SYSCLK/256	75 KHz
1	1	0	S0TOF/6	Variable
1	1	1	T0OF/6	Variable

Note:1. The Maximum TWI0/I2C0 *clock Rate should under 1MHz, to set SYSCLK = 8MHz to generate 1MHz.*
2. SYSCLK is the system clock.
3. S0TOF is UART0 Baud-Rate Generator Overflow.
4. T0OF is Timer 0 Overflow.

SISTA: TWI0/I2C0 Status Register

SFR Page = **0 Only**
SFR Address = 0xD3 RESET = 1111-1000

7	6	5	4	3	2	1	0
SIS7	SIS6	SIS5	SIS4	SIS3	SIS2	SIS1	SIS0
R	R	R	R	R	R	R	R

SISTA is an 8-bit read-only register. The three least significant bits are always 0. The five most significant bits contain the status code. There are a number of possible status codes. When SISTA contains F8H, no serial interrupt is requested. All other SISTA values correspond to defined TWI0/I2C0 states. When each of these states is entered, a status interrupt is requested (SI=1). A valid status code is present in SISTA when SI is set by hardware.

In addition, state 00H stands for a Bus Error. A Bus Error occurs when a START or STOP condition is present at an illegal position, such as inside an address/data byte or just on an acknowledge bit.

SIA2: TWI0/I2C0 2nd Address Register

SFR Page = **2 Only**
SFR Address = 0xD1 RESET = 0000-0000

7	6	5	4	3	2	1	0
A2.6	A2.5	A2.4	A2.3	A2.2	A2.1	A2.0	A2E
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~1: 2nd slave address content of TWI0/I2C0.

Bit 0: A2E, Enable control of 2nd slave address recognition.
0: Disable 2nd slave address recognition.
1: Enable 2nd slave address recognition.

SIA2M: TWI0/I2C0 2nd Address Mask Register

SFR Page = 2 Only
SFR Address = 0xD2 RESET = 1111-1111

7	6	5	4	3	2	1	0
A2M.6	A2M.5	A2M.4	A2M.3	A2M.2	A2M.1	A2M.0	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SIA2 register is combined with ISA2M register for 2nd address recognition. In fact, SIA2M functions as the “mask” register for SIA2 register. The following is the example for it.

SIA2[7:1] = 1100 000
SIA2M[7:1] = 1111 110
2nd ADR[7:1] = 1100 00x ⟶ The 2nd slave address will be checked except bit 1 is treated as “don’t care”

Bit 0: Reserved. Software must write “1” on this bit when SIA2M is written.

AUXR3: Auxiliary Register 3

SFR Page = **0 only**
SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 2~1: TWIPS1~0, TWI0/I2C0 Port Selection [1:0].

TWIPS1~0	TWI0_SCL	TWI0_SDA
0 0	P3.1	P3.0
0 1	P4.4	P4.5
1 0	P3.0	P3.1
1 1	P3.3	P4.6

AUXR10: Auxiliary Register 10

SFR Page = **7 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
0	AC0HC0	0	SPIPS0	S0PS1	SPFACE	TWICF	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 1: TWICF, TWI0/I2C0 serial Clock input Filter.
0: Disable TWICF function.
1: Enable TWICF function.

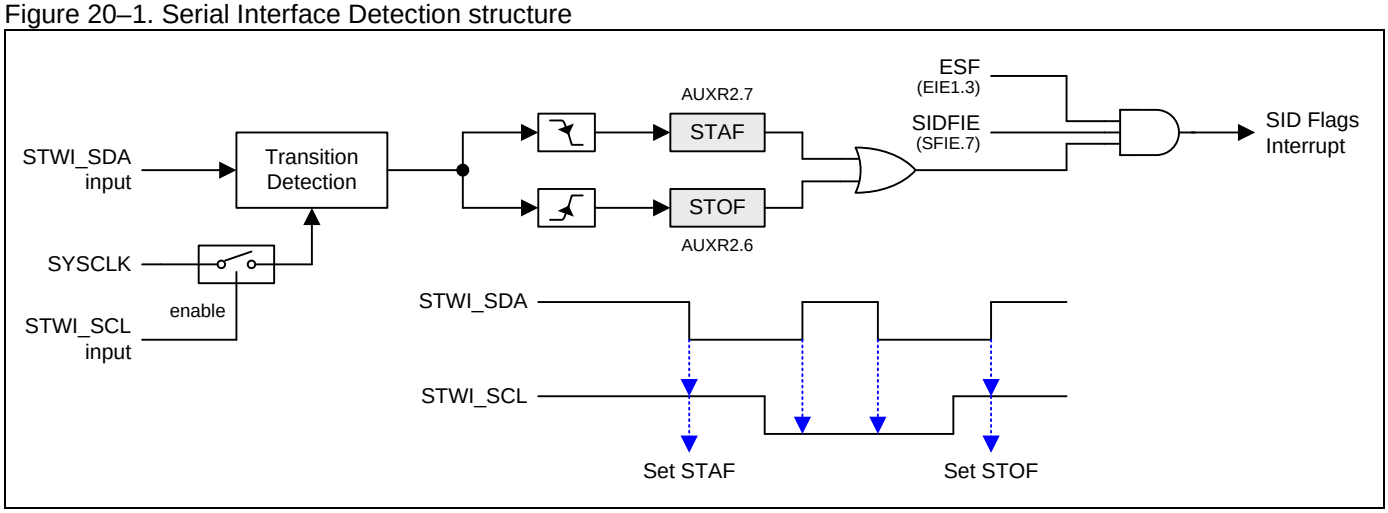
20. Serial Interface Detection (STWI/SI2C)

The serial interface detection module (SID) is always monitoring the “Start” and “Stop” condition on software two-wire-interface (STWI/SI2C). STWI_SCL is the serial clock signal and STWI_SDA is the serial data signal. If any matched condition is detected, hardware set the flag on STAF and STOF. Software can poll these two flags or set SIDFIE (SFIE.7) to share the interrupt vector on System Flag. And STWI_SCL is located on nINT1 which helps MCU to strobe the serial data by nINT1 interrupt. Software can use these resources to implement a variable TWI slave device.

20.1. SID Structure

Figure 20–1 shows the configuration of STAF and STOF detection, interrupt architecture and event detecting waveform.

Figure 20–1. Serial Interface Detection structure



20.2. SID Register

AUXR2: Auxiliary Register 2

SFR Page = 0~F

SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	W	W	R/W	R/W	R/W	R/W

Bit 7: STAF, Start Flag detection of STWI (SID).
0: Clear by firmware by writing "0" on it. STAF might be held within MCU reset period, so needs to clear STAF in firmware initial.
1: Set by hardware to indicate the START condition occurred on STWI bus.

Bit 6: STOF, Stop Flag detection of STWI (SID).
0: Clear by firmware by writing "0" on it.
1: Set by hardware to indicate the STOP condition occurred on STWI bus. STOF might be held within MCU reset period, so needs to clear STOF in firmware initial.

SFIE: System Flag Interrupt Enable Register

SFR Page = 0~F

SFR Address = 0x8E POR = 0110-0000

7	6	5	4	3	2	1	0
SIDFIE	1	1	RTCFIE	SPWIE	BOF1IE	BOF0IE	WDTFIE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SIDFIE, Serial Interface (STWI/SI2C) Detection Flag Interrupt Enabled.
0: Disable SID Flags (STAF or STOF) interrupt.
1: Enable SID Flags (STAF or STOF) interrupt.

AUXR9: Auxiliary Register 9

SFR Page = 6 only

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	SIDPS0	T1G1	T0G1	C0FDC1	C0FDC0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: Reserved. Software must write "**0**" on these bits when AUXR9 is written.

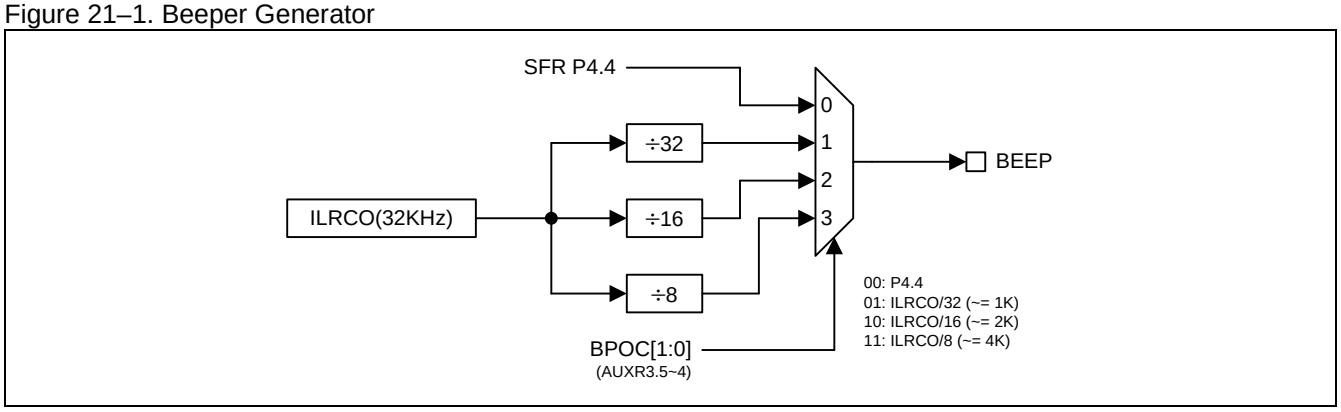
Bit 6: SID/STWI Port pin Selection.

SIDPS0	STWI_SCL	STWI_SDA
0	nINT1	S0MI
1	TWI0_SCL	TWI0_SDA

21. Beeper

The beeper function outputs a signal on the BEEP pin for sound generation. The signal is in the range about 1, 2 or 4 kHz which is divided from ILRCO. Figure 21–1 shows the beeper generator circuit. But ILRCO is not the precision clock source. Please refer Section "32.5 ILRCO Characteristics" for more detailed ILRCO frequency deviation range.

Figure 21–1. Beeper Generator



21.1. Beeper Register

AUXR3: Auxiliary Register 3

SFR Page = **0 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 5~4: BPOC1~0, Beeper output control bits.

BPOC[1:0]	P4.4 function	I/O mode
00	P4.4	By P4M0.4 & P4M1.4
01	ILRCO/32	By P4M0.4 & P4M1.4
10	ILRCO/16	By P4M0.4 & P4M1.4
11	ILRCO/8	By P4M0.4 & P4M1.4

For beeper on P4.4 function, it is recommended to configure P4.4 as push-push output mode.

Beeper will use P4.4, and please disable OCD function before enable Beeper function.

DCON0: Device Control 0

SFR Page = P Only

SFR Address = 0x4C RESET = 1000-0011

7	6	5	4	3	2	1	0
HSE	IAPO	0	0	0	IORCTL	RSTIO	OCDE
R/W	**R/W**	**R/W**	**W**	**W**	**W**	**R/W**	**R/W**

Bit 0: OCDE, OCD enable.
0: Disable OCD interface on P4.4 and P4.5
1: Enable OCD interface on P4.4 and P4.5.

22. Keypad Interrupt (KBI)

The Keypad Interrupt function is intended primarily to allow a single interrupt to be generated when KBI.7~0 is equal to or not equal to a certain pattern. This function can be used for bus address recognition or keypad recognition.

There are three SFRs used for this function. The Keypad Interrupt Mask Register (KBMASK) is used to define which KBI input pins are enabled to trigger the interrupt. The Keypad Pattern Register (KBPATN) is used to define a pattern that is compared to the value of keypad input. The Keypad Interrupt Flag (KBIF) in the Keypad Interrupt Control Register (KBCON) is set by hardware when the condition is matched. An interrupt will be generated if it has been enabled by setting the EKBI bit in EIE1 register and EA=1. The PATN_SEL bit in the Keypad Interrupt Control Register (KBCON) is used to define “equal” or “not-equal” for the comparison. The keypad input can be assigned on the different port pins, please refer Section “4.3 Alternate Function Redirection” for more detailed information.

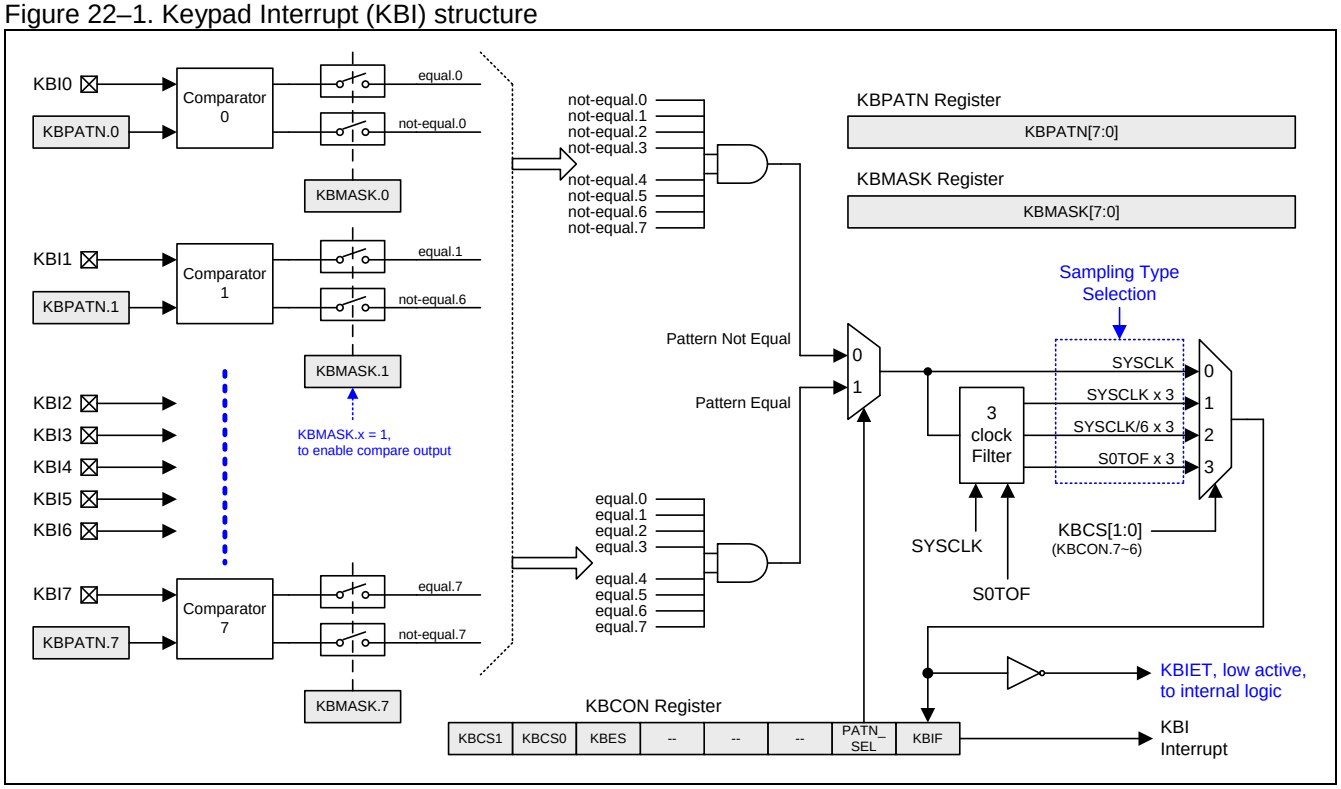
In order to use the Keypad Interrupt as the “Keyboard” Interrupt, the user needs to set KBPATN=0xFF and PATN_SEL=0 (not equal), then any key connected to keypad input which is enabled by KBMASK register will cause the hardware to set the interrupt flag KBIF and generate an interrupt if it has been enabled. To rewrite KBPATN to clear internal logic to allow next key press event.

The interrupt may wake up the CPU from Idle mode or Power-Down mode. This feature is particularly useful in handheld, battery powered systems that need to carefully manage power consumption but also need to be convenient to use.

When using “Not Equal” (PATN_SEL = 0) with edge trigger, it should rewrite KBPATN with 0xFF to enable next edge event to trigger interrupt.

22.1. KBI Structure

Figure 22–1. Keypad Interrupt (KBI) structure



22.2. KBI Register

The following special function registers are related to the KBI operation:

KBPATN: Keypad Pattern Register

SFR Page = 0~F
SFR Address = 0xD5 RESET = 1111-1111

7	6	5	4	3	2	1	0
KBPATN.7	KBPATN.6	KBPATN.5	KBPATN.4	KBPATN.3	KBPATN.2	KBPATN.1	KBPATN.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: KBPATN.7~0: The keypad pattern, reset value is 0xFF.

KBCON: Keypad Control Register

SFR Page = 0~F
SFR Address = 0xD6 RESET = 0000-0000

7	6	5	4	3	2	1	0
KBCS1	KBCS0	KBES	--	0	0	PATN_SEL	KBIF
R/W	R/W	R/W	R	R/W	R/W	R/W	R/W

Bit 7~6: KBCS1~0, KBI Filter mode control.

KBCS1~0	KBI input filter mode
00	Disabled
01	SYSCLK x 3
10	SYSCLK/6 x 3
11	S0TOF x 3

Bit 5: KBES, KBI Edge mode select.
0: Set KBI module to level detection mode.
1: Set KBI module to edge detection mode.

Bit 3 ~ 2: Reserved. Software must write “0” on this bit when KBCON is written.

Bit 1: PATN_SEL, Pattern Matching Polarity selection.
0: The keypad input has to be not equal to user-defined keypad pattern in KBPATN to generate the interrupt.
1: The keypad input has to be equal to the user-defined keypad pattern in KBPATN to generate the interrupt.

Bit 0: KBIF, Keypad Interrupt Flag. The default value of KBIF is set to “1”.
0: Must be cleared by software by writing “0”.
1: Set when keypad input matches user defined conditions specified in KBPATN, KBMASK, and PATN_SEL.

KBMASK: Keypad Interrupt Mask Register

SFR Page = 0~F
SFR Address = 0xD7 RESET = 0000-0000

7	6	5	4	3	2	1	0
KBMASK.7	KBMASK.6	KBMASK.5	KBMASK.4	KBMASK.3	KBMASK.2	KBMASK.1	KBMASK.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

KBMASK.7: When set, enables KBI7 input as a cause of a Keypad Interrupt.
KBMASK.6: When set, enables KBI6 input as a cause of a Keypad Interrupt.
KBMASK.5: When set, enables KBI5 input as a cause of a Keypad Interrupt.
KBMASK.4: When set, enables KBI4 input as a cause of a Keypad Interrupt.
KBMASK.3: When set, enables KBI3 input as a cause of a Keypad Interrupt.
KBMASK.2: When set, enables KBI2 input as a cause of a Keypad Interrupt.
KBMASK.1: When set, enables KBI1 input as a cause of a Keypad Interrupt.
KBMASK.0: When set, enables KBI0 input as a cause of a Keypad Interrupt.

23. General Purpose Logic (GPL-CRC)

The **MG82F6B08 / 6B001/ 6B104** builds in a general purpose logic cyclic redundancy check function with CCITT16 (CRC16 0x1021) polynomial. The CRC accepts a stream of 8-bit data written to the CRC0DI. Its initial value (seed value) is programmable for multi-purpose applications. The 16-bit initial value (seed value) is set to high byte CRC0SH (CRCDS0~1=01) and low byte CRC0SL (CRCDS0~1=00). The result is stored in CRC0RH (CRCDS0~1=01) and CRC0RL (CRCDS0~1=00).

The GPL-CRC has another data path direct from Flash memory by the Flash Auto-Reload Engine to dynamically check the data correctness in the Flash.

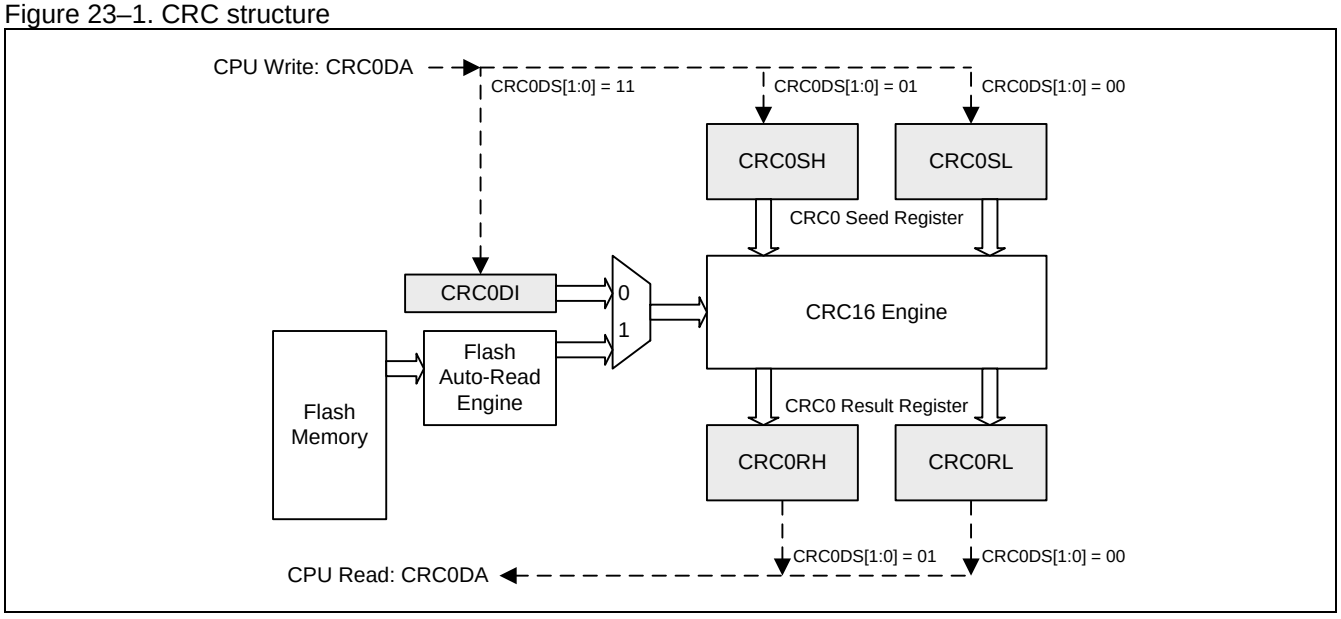
The GPL-CRC can also combine the data inverse function. To write the data byte into BOREV register and it will be flipped automatically when read it back from BOREV. The MSB becomes the LSB.

23.1. GPL-CRC Structure

In the normal mode, it needs to set the seed in CRC0SH and CRC0SL and then write the data into CRC0DI to start the conversion.

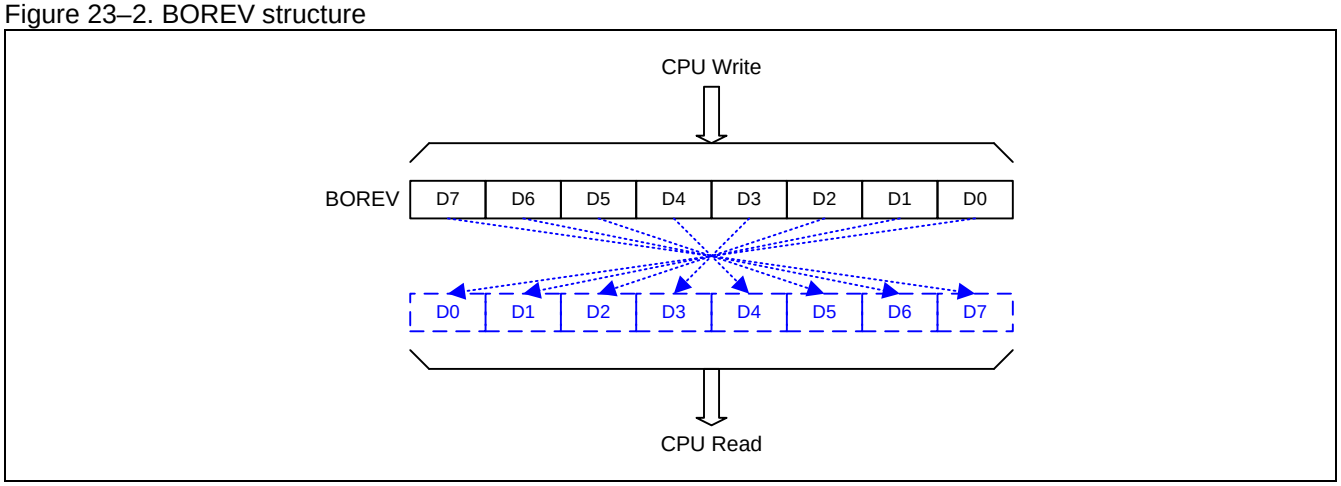
In the Flash Auto-Read mode, it needs to keep CRCDS1~0 at “0x11”. And follow the steps show in below:
1. To set the start address of the reload sector, this is defined in IFADRH and IFADRL.
2. To set its end-address is combined the IAPLB (7 bits) and 9’b1-1111-1111.
3. Set IFMT register (ISP/IAP Flash Mode) to 0x80 for Flash Auto-Read mode.
4. Sequentially write 0x46h then 0xB9h to SCMD register to trigger CRC calculation.

Figure 23–1. CRC structure



23.2. GPL-BOREV Structure

Figure 23–2. BOREV structure



23.3. GPL Register

The following special function registers are related to the CRC operation:

CRC0DA: CRC0 Data Port

SFR Page = 0~F

SFR Address = 0xB6 RESET = 0000-0000

7	6	5	4	3	2	1	0
CRC0DA.7	CRC0DA.6	CRC0DA.5	CRC0DA.4	CRC0DA.3	CRC0DA.2	CRC0DA.1	CRC0DA.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: CRC0 Data Port. The CRC0 data access is defined as following table:

CRCDS1~0	CPU R/W	CRC0 Data Selection	Description
00	Write	CRC0SL	CRC0 Data Seed register-L.
01	Write	CRC0SH	CRC0 Data Seed register-H.
10	Write	--	Reserved.
11	Write	CRC0DI	CRC0 Data Input register.
00	Read	CRC0RL	CRC0 Result register-L.
01	Read	CRC0RH	CRC0 Result register-H.
10	Read	--	Reserved.
11	Read	--	Reserved.

AUXR1: Auxiliary Control Register 1

SFR Page = 0~F

SFR Address = 0xA2 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	CRCDS1	CRCDS0	0	0	0	DPS
W	W	R/W	R/W	W	W	W	R/W

Bit 5~4: CRCDS1~0. CRC0 Data port Selection bit 1~0.

BOREV: Bit Order Reversed data register

SFR Page = 0~F

SFR Address = 0x96 RESET = 0000-0000

7	6	5	4	3	2	1	0
BOREV.7	BOREV.6	BOREV.5	BOREV.4	BOREV.3	BOREV.2	BOREV.1	BOREV.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~0: BOREV7~0, data read/write for Bit-Order-Reversed function.
Any byte written to BOREV is read back in a bit-reversed order, i.e., the written LSB becomes the MSB. For example:
If 0xA0 is written to BOREV, the data read back will be 0x05.
If 0x01 is written to BOREV, the data read back will be 0x80.

IFMT: ISP/IAP/EEPROM Mode Table

SFR Page = 0~F

SFR Address = 0xE5 RESET = xxxx-x000

7	6	5	4	3	2	1	0
MS.7	MS.6	MS.5	MS.4	MS.3	MS.2	MS.1	MS.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~4: Reserved. Software must write “0000_0” on these bits when IFMT is written.

Bit 3~0: ISP/IAP/EEPROM/Page-P operating mode selection

MS[7:0]	Mode
0 0 0 0-0 0 0 0	Standby
0 0 0 0-0 0 0 1	Flash byte read of AP/IAP-memory
0 0 0 0-0 0 1 0	Reserved
0 0 0 0-0 0 1 1	Reserved
0 0 0 0-0 1 0 0	Page P SFR Write
0 0 0 0-0 1 0 1	Page P SFR Read
1 0 0 0-0 0 0 0	Automatic Flash read for CRC.
1 0 0 0-0 0 0 1	Flash byte read with address increased function
1 0 0 0-0 0 1 0	Flash byte write with address increased function. Always stay in target page.
1 1 0 0-0 0 0 0	EEPROM Byte read
1 1 0 0-0 0 0 1	Flash Page Program
1 1 0 0-0 0 1 0	Flash Byte Write in 64 Byte data latch
1 1 0 0-0 0 1 1	Flash Page Erase
1 1 0 0-1 0 0 0	EEPROM Byte Program
Others	Reserved

IFMT is used to select the Flash mode for performing numerous ISP/IAP function or to select page P SFR access.

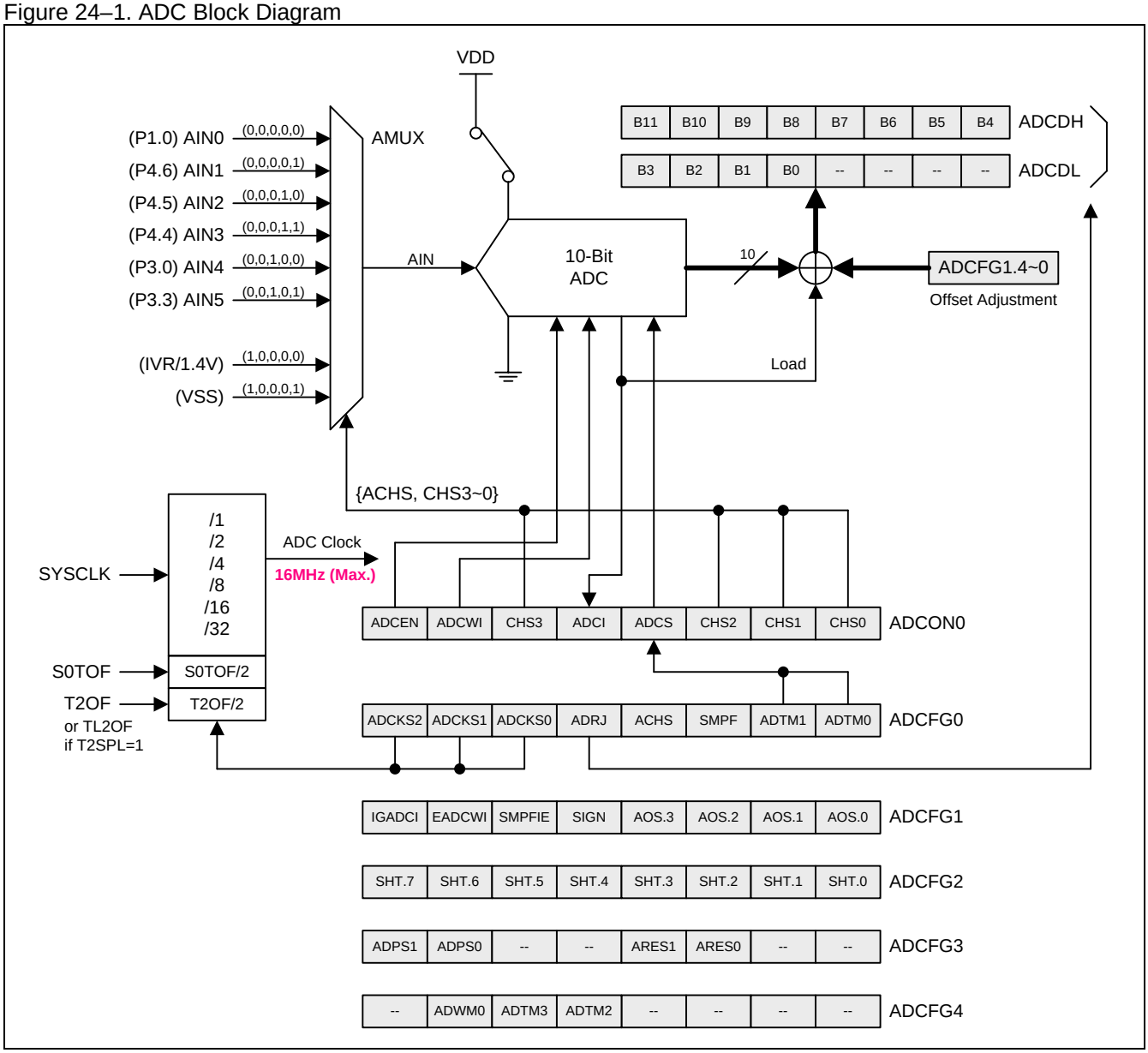
If software selects the mode on automatic Flash read for CRC, the Flash start-address is defined in IFARDH and IFADRL. The Flash end-address is defined at {IAPLB + 9’b1-1111-1111}.

24. 10-Bit ADC

The ADC subsystem for the **MG82F6B08 / 6B001/ 6B104** consists of an analog multiplexer (AMUX), and a **666K**sps, **10**-bit successive-approximation-register ADC. The AMUX can be configured via the Special Function Registers shown in Figure 24–1. ADC operates in Single-ended mode, and may be configured to measure any of the pins on AIN0 ~ AIN5 or internal reference. The ADC subsystem is enabled only when the ADEN bit in the ADC Control register (ADCON0) is set to logic 1. The ADC subsystem is in low power shutdown when this bit is logic 0.

24.1. ADC Structure

Figure 24–1. ADC Block Diagram



24.2. ADC Operation

ADC has a maximum conversion speed of **666K** sps. The ADC conversion clock is a divided version of the system clock, S0 BRG overflow or Timer 2 overflow, determined by the ADCKS2~0 bits in the ADCFG0 register. The ADC conversion clock should be no more than16MHz.

After the conversion is complete (ADCI is high), the conversion result can be found in the ADC Result Registers (ADCDH, ADCDL).For single ended conversion, and the result is

$$\text{ADC Result} = \frac{V_{IN} \times 1024}{\text{VDD Voltage}}$$

24.2.1. ADC Input Channels

The analog multiplexer (AMUX) selects the inputs to the ADC, allowing any of the pins on AIN5~0 to be measured in single-ended mode and one internal voltage reference (IVR, 1.4V). The ADC input channels are configured and selected by **CHS3~0**in theADCON0 register and ACHS in the ADCFG0 register as shown in Figure 24–1. The selected pin is measured with respect to GND.

24.2.2. ADC Internal Voltage Reference

The default ADC reference is VDD. If the VDD is not fixed at a certain voltage, then use the following steps to read voltage:

1) To set the analog multiplexer (AMUX) to IVR.
2) Convert and store the IVR value by ADC. (Hint: Different VDD voltage will get different IVR read back value, but IVR is fixed at 1.4V. So this read back value can be tread as the reference value.)
3) To use the IVR read back reference value to calculate the VDD value. Now the VDD get a certain value, and can be treated as the reference voltage.
4) To use the reference voltage converts the input voltage.

24.2.3. Starting a Conversion

Prior to using the ADC function, the user should:

1) Set ADC Data Resolution to 10-bit or 8-bit Data mode
2) Turn on the ADC hardware by setting the ADCEN bit
3) Configure the ADC input clock by bits ADCKS2, ADCKS1 and ADCKS0
4) Select the analog input channel by bits ACHS,CHS3, CHS2, CHS1 and CHS0
5) Configure the selected port input to the Analog-Input-Only mode, and
6) Configure ADC result arrangement using ADRJ bit.

Now, user can set the ADCS bit to start the A-to-D conversion. The conversion time is controlled by the bits ADCKS2, ADCKS1 and ADCKS0. Once the conversion is completed, the hardware will automatically clear the ADCS bit, set the interrupt flag ADCI and load the **10** bits of conversion result into ADCDH and ADCDL (according to ADRJ bit) simultaneously. If user sets the ADCS and selects the ADC trigger mode to **S0BRG/Timer2** over flow or free-run, then the ADC will keep conversion continuously unless ADCEN is cleared or configure ADC to manual mode.

As described above, the interrupt flag ADCI, when set by hardware, shows a completed conversion. Thus two ways may be used to check if the conversion is completed: (1) Always polling the interrupt flag ADCI by software; (2) Enable the ADC interrupt by setting bits EADC (in EIE1 register) and EA (in IE register), and then the CPU will jump into its Interrupt Service Routine when the conversion is completed. Regardless of (1) or (2), the ADCI flag should be cleared by software before next conversion.

24.2.4. ADC Conversion Rate

The user can select the appropriate conversion speed according to the frequency of the analog input signal. The maximum input clock of the ADC is **16MHz** and it operates a minimum conversion time with **24** ADC clocks. User can configure the ADCKS2~0 (ADCFG0.7~5), SHT (ADCFG2.7~0) and HA (ADCFG3.5) to specify the conversion rate. The following equation is the clock number of one ADC conversion:

$$\text{ADC Conversion Rate} = \frac{\text{ADC Clock Freq.}}{(24 + X)} \quad ; X = SHT, 0\sim255$$

Please note is the input signal is AC signal, f_N, and assume the sample rate is f_S, based on Nyquist theorem, f_S should large than 2 times f_N to ensure the measurement accuracy.

For example,

1. To get 666K Sample Rate:
   If SYSCLK= 16MHz and the ADCKS = SYSCLK is selected, SHT = 0,
   Then conversion ratef_S= 16MHz/(24+0) = 666K sps.
   (In this case, the AC input signal f_N frequency should lower than 333KHz to ensure the measurement accuracy.)
2. To get 100K Sample Rate:
   If SYSCLK= 16MHz and the ADCKS = SYSCLK/4is selected, SHT = 10,
   Then conversion rate f_S = 16MHz/4/(24+16) = 100K sps.
   (In this case, the AC input signal f_N frequency should lower than 50KHz to ensure the measurement accuracy.)

24.2.5. ADC Interrupts

The ADC interrupt of **MG82F6B08 / 6B001/ 6B104** includes 3 sources:

1. ADCI, when an A/D conversion is completed, ADCI will be set to invoke an interrupt. The interrupt on this flag can be blocked by IGADCI (ADCFG1.7).
2. SMPF, it is set when an ADC channel sample & hold is completed to invoke an interrupt. The interrupt on this flag can be blocked by SMPFIE (ADCFG1.5).
3. ADCWI, under ADC Window Compare mode, this Interrupt flag will be held when Window Comparison Data match has occurred. An interrupt is invoked if it is enabled. The interrupt on this flag can be enabled by EADCWI. (ADCFG1.6)

Figure 24–2. ADC Interrupt

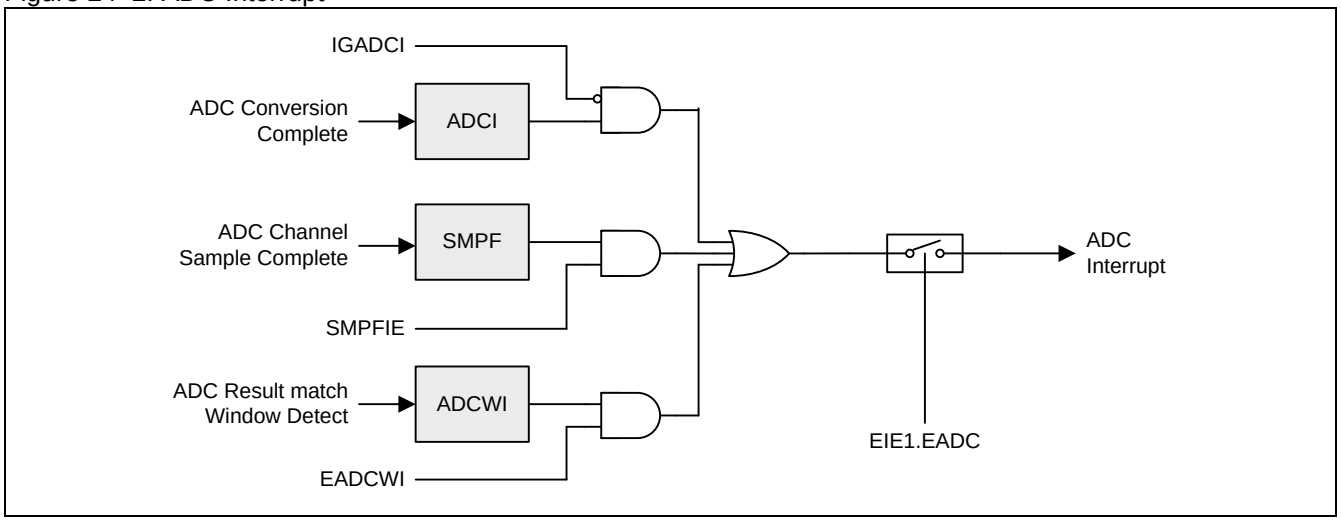
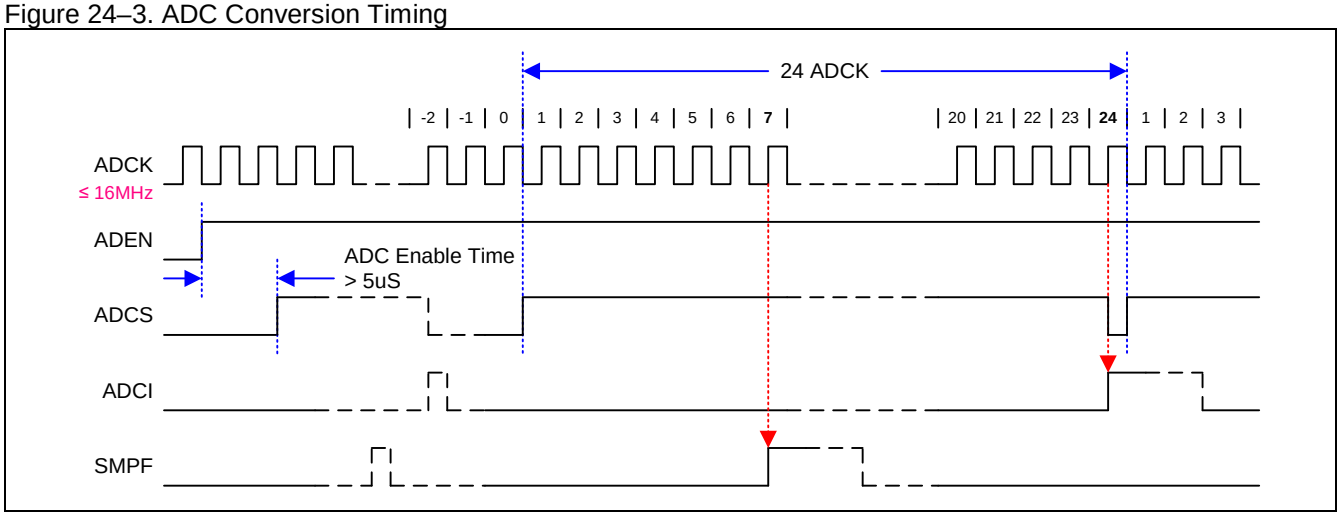


Figure 24–3. ADC Conversion Timing

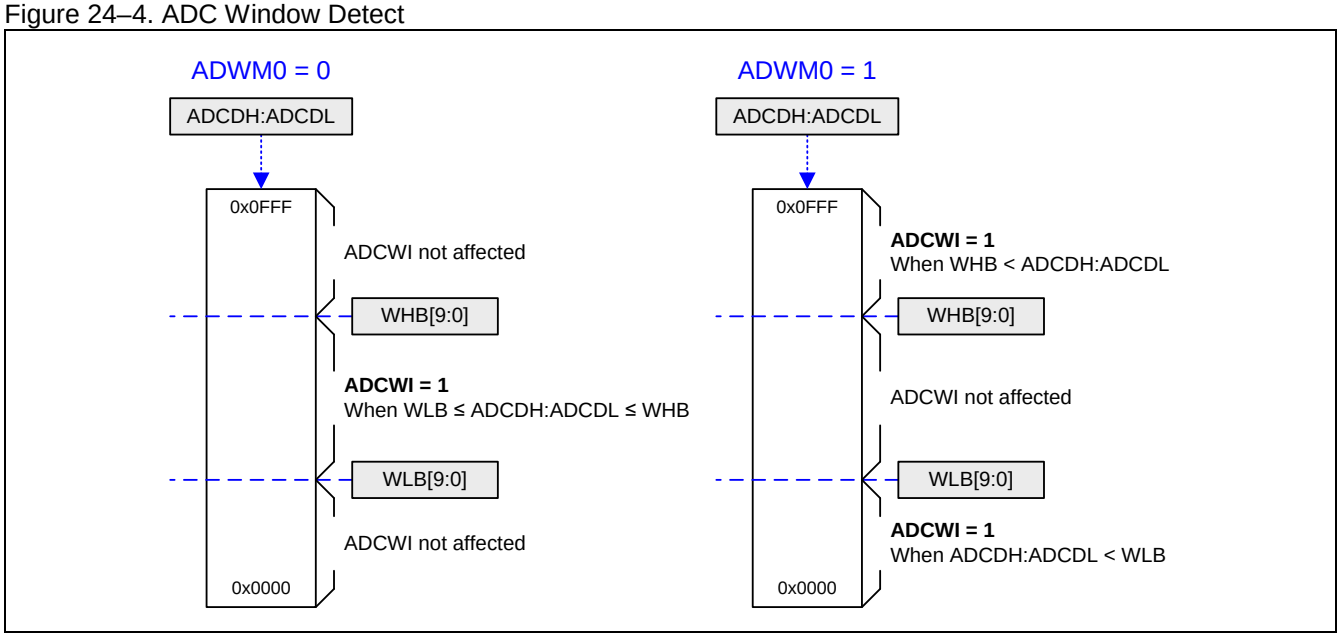


24.2.6. ADC Window Detect

The **MG82F6B08 / 6B001/ 6B104**ADC's programmable window detector continuously compares the ADC output registers with user-programmed limits, and notifies the system when a desired condition is detected. This is especially effective in an interrupt driven system, saving code space and CPU bandwidth while delivering faster response times. The window detector interrupt flag (ADCWI) can also be used in polled mode. The Window-High-Boundary (WHB[9:0] , {ADCFG12, ADCFG11}) and Window-Low-Boundary (WLB[9:0], {ADCFG14, ADCFG13}) registers hold the boundary values. The Window-Boundary flags can be programmed to catch the ADC convert value (ADCDH:ADCDL) when it is inside or outside of the user-defined boundary. The following figure shows the two window detect modes:

1. ADWM0 = 0: When ADC convert value is “inside” the boundary the interrupt flag ADCWI will be held. Which means the condition WLB[9:0] ≤ ADCDH:ADCDL ≤ WHB[9:0] is true, ADCWI will be held.
2. ADWM0 = 1: When ADC convert value is “outside” the boundary the interrupt flag ADCWI will be held. Which means the condition WLB[9:0] > ADCDH:ADCDL or ADCDH:ADCDL< WHB[9:0] is true, ADCWI will be held.

Figure 24–4. ADC Window Detect



Another application of ADC Window Detect is to specify the voltage is lager or less than a specific voltage. For example:

1. The target voltage ≥ the condition: ADWM0 = 0, to set condition value in WLB and set WHB = 0x3FF
2. The voltage less ≤ the condition: ADWM0 =0, to set the condition value in WHB and set WLB = 0
3. The target voltage > the condition: ADWM0 = 1, to set condition value in WHB and set WLB = 0
4. The target voltage < the condition: ADWM0 = 1, to set condition value in WLB and set WHB = 0x3FF

24.2.7. I/O Pins Used with ADC Function

The analog input pins used for the A/D converters also have its I/O port's digital input and output function. In order to give the proper analog performance, a pin that is being used with the ADC should have its digital output as disabled. It is done by putting the port pin into the analog-input-only mode to AIN5 ~ 0. The port pin configuration for analog input function is described in "Table 13–3. General Port Configuration Settings" and regarding the AIN port pin setting please reference Section "13.2 I/O Port Register".

24.2.8. Idle and Power-Down Mode

If the ADC is turned on in Idle mode and Power-Down mode, it will consume a little power. So, power consumption can be reduced by turning off the ADC hardware (ADCEN=0) before entering Idle mode and Power-Down mode.

In Power-Down mode, the ADC does not function. If software triggers the ADC operation in Idle mode, the ADC will finish the conversion and set the ADC interrupt flag, ADCI. When the ADC interrupt enable (EADC, EIE1.1) is set, the ADC interrupt will wake up CPU from Idle mode.

24.2.9. How to improve ADC Accuracy

To use ADC measure the voltage, its accuracy might be affected by many factors, for example, the power noise of the MCU VDD or tolerance of the reference voltage. **MG82F6B08 / 6B001/ 6B104** has trimmed the internal reference voltage – IVR under VDD equals to 3.3V, and use the ADC to read its ADC value to store in flash ROM as the Presorted value. To use this value by following formulas to calculate the AIN voltage instead of measuring VDD to calculate the 1 LSB voltage.

- To push back the IVR voltage (which was measured under VDD=3.3V)

$$IVR\ Voltage = \frac{IVR_{ADC\_PreStored\_Value} * 3300}{1024}(mV) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(1)$$

- To use the proportional relationship calculate the I/O pin voltage

$$AIN\ Voltage = \frac{IVR\ Voltage * AIN_{ADC\_Value}}{IVR_{ADC\_Value}}(mV) \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots(2)$$

Note: To read the IVR ADC Presorted value please reference 26.3 How to read IVR (2.4V) ADC Pre-stored value.

24.3. ADC Register

ADCON0: ADC Control Register 0

SFR Page = 0~F

SFR Address =**0xC4** RESET=0000-0000

7	6	5	4	3	2	1	0
ADCEN	ADCWI	CHS3	ADCI	ADCS	CHS2	CHS1	CHS0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: ADCEN, ADC Enable.
0: Clear to turn off the ADC block.
1: Set to turn on the ADC block. At least 5us ADC enabled time is required before set ADCS.

Bit 6: ADCWI, ADC Window Compare Interrupt flag.
0: ADC0 Window Comparison Data match has not occurred since this flag was last cleared. The flag must be cleared by software.
1: This flag is set when ADC Window Comparison Data match has occurred. An interrupt is invoked if it is enabled. The interrupt on this flag can be enabled by EADCWI. (ADCFG1.6)

Bit 5: CHS3. Combined CH2~0 to select ADC input channel.

Bit 4: ADCI, ADC Interrupt Flag.
0: The flag must be cleared by software.
1: This flag is set when an A/D conversion is completed. An interrupt is invoked if it is enabled. The interrupt on this flag can be blocked by IGADCI (ADCFG1.7).

Bit 3: ADCS. ADC Start of conversion.
0: ADCS cannot be cleared by software.
1: Setting this bit by software starts an A/D conversion. On completion of the conversion, the ADC hardware will clear ADCS and set the ADCI. A new conversion may not be started while either ADCS or ADCI is high.

Bit 2~0: CHS2 ~ CHS1,Input Channel Selection for ADC analog multiplexer.

In Single-ended mode:

ACHS	CHS3~0	Selected Channel
0	0 0 0 0	AIN0 (P1.0)
0	0 0 0 1	AIN1 (P4.6)
0	0 0 1 0	AIN2 (P4.5)
0	0 0 1 1	AIN3 (P4.4)
0	0 1 0 0	AIN4 (P3.0)
0	0 1 0 1	AIN5 (P3.3)
1	0 0 0 0	Int. VREF (IVR/1.4V)
1	0 0 0 1	AVSS
	Others	Reserved

ADCFG0: ADC Configuration Register 0

SFR Page = **0 Only**

SFR Address = **0xC3** RESET = 0000-0000

7	6	5	4	3	2	1	0
ADCKS2	ADCKS1	ADCKS0	ADRJ	ACHS	SMPF	ADTM1	ADTM0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~5: ADC Conversion Clock Select bits.

ADCKS[2:0]	ADC Clock Selection
0 0 0	SYSCLK
0 0 1	SYSCLK/2
0 1 0	SYSCLK/4
0 1 1	SYSCLK/8
1 0 0	SYSCLK/16
1 0 1	SYSCLK/32
1 1 0	S0TOF/2
1 1 1	T2OF/2

Note:

1. *SYSCLK is the system clock.*
2. *S0TOF is UART0 Baud-Rate Generator Overflow.*
3. *T2OF is Timer2 Overflow.*

Bit 4: ADRJ, ADC result Right-Justified selection.

0: The most significant 8 bits of conversion result are saved in ADCDH[7:0], while the least significant 2 bits in ADCDL[7:6].

1: The most significant 2 bits of conversion result are saved in ADCDH[1:0], while the least significant 8 bits in ADCDL[7:0].

If ADRJ = 0

ADCDH: ADC Date High Byte Register

SFR Page = 0~F

SFR Address = 0xC6 RESET = xxxx-xxxx

7	6	5	4	3	2	1	0
(B11)	(B10)	(B9)	(B8)	(B7)	(B6)	(B5)	(B4)
R	R	R	R	R	R	R	R

ADCDL: ADC Data Low Byte Register

SFR Page = 0~F

SFR Address =**0xC5** RESET = xxxx-xxxx

7	6	5	4	3	2	1	0
(B3)	(B2)	--	--	--	--	--	--
R	R	R	R	R	R	R	R

If ADRJ = 1

ADCDH

7	6	5	4	3	2	1	0
--	--	--	--	--	--	(B11)	(B10)
R	R	R	R	R	R	R	R

ADCDL

7	6	5	4	3	2	1	0
(B9)	(B8)	(B7)	(B6)	(B5)	(B4)	(B3)	(B2)
R	R	R	R	R	R	R	R

When in Single-ended Mode, conversion codes are represented as 10-bit unsigned integers. Inputs are measured from ‘0’ to VDD(VREF) x 1023/1024. Example codes are shown below for both right-justified and left-justified data. Unused bits in the ADCDH and ADCDL registers are set to ‘0’.

Input Voltage (Single-Ended)	ADCDH:ADCDL (ADRJ = 0)	ADCDH:ADCDL (ADRJ = 1)
VREF+ x 1023/1024	0xFFC0	0x03FF
VREF+ x 512/1024	0x8000	0x0200
VREF+ x 256/1024	0x4000	0x0100
VREF+ x 128/1024	0x2000	0x0080
0	0x0000	0x0000

Bit 3: ACHS,ADC Auxiliary Channel Select. Decode ACHS and CHS3~0 to select ADC input channel.

Bit 2: SMPF. ADC channel sample & hold flag.
0: The flag must be cleared by software.
1: This flag is set when an ADC channel sample & hold is completed. An interrupt is invoked if it is enabled. The interrupt on this flag can be enabled by SMPFIE (ADCFG1.5).

Bit 1~0: ADC Trigger Mode selection.

ADTM[1:0]	ADC Conversion Start Selection
0 0	Set ADCS
0 1	Timer 0 overflow
1 0	Free running mode
1 1	S0 BRG overflow

ADCFG1: ADC Configuration Register 1

SFR Page = **1 Only**
SFR Address = **0xC3** RESET = 0000-0000

7	6	5	4	3	2	1	0
IGADCI	EADCWI	SMPFIE	SIGN	AOS.3	AOS.2	AOS.1	AOS.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: IGADCI, Ignore ADCI interrupt.
0: Enabled ADCI interrupt. Default is enabled.
1: Disable ADCI interrupt.

Bit 6: EADCWI, ADCWI interrupt enable.
0: Disable ADCWI interrupt.
1: Enable ADCWI interrupt to share the ADC interrupt vector.

Bit 5: SMPFIE, SMPF interrupt enable.
0: Disable SMPF interrupt.
1: Enable SMPF interrupt to share the ADC interrupt vector.

Bit 4~0: SIGN and AOS.3~0. The register value adjusts the ADC result in {ADCDH, ADCDL} for offset cancellation. Software can dynamically collect the ADC offset value. Software can also store the value in **MG82F6B08 / 6B001/ 6B104** IAP zone to use it as a constant parameter for ADC offset cancellation. The following table lists the adjustment value for ADC transfer result.

{Sign, AOS.[3:0]}	Value in {ADCDH, ADCDL}
0_1111	ADC transfer value + 15
0_1110	ADC transfer value + 14
……	……
0_0010	ADC transfer value + 2
0_0001	ADC transfer value + 1
0_0000	ADC transfer value + 0
1_1111	ADC transfer value – 1
1_1110	ADC transfer value – 2
……	……
1_0001	ADC transfer value – 15
1_0000	ADC transfer value – 16

ADCFG2: ADC Configuration Register 2

SFR Page =**2 only**
SFR Address = **0xC3** RESET = 0000-0000

7	6	5	4	3	2	1	0
SHT.7	SHT.6	SHT.5	SHT.4	SHT.3	SHT.2	SHT.1	SHT.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~0: SHT[7:0], extend ADC sample time. The value of SHT is 0~255 ADC clocks.

ADCFG3: ADC Configuration Register 3

SFR Page =**3 only**

SFR Address = **0xC3** RESET = 0100-0000

7	6	5	4	3	2	1	0
ADPS1	ADPS0	0	0	ARES1	ARES0	0	0
R/W	**R/W**	**W**	**W**	**R/W**	**R/W**	**R/W**	**W**

Bit 7~6: ADPS1~0, ADC Trigger Mode selection bit 3~2.

ADPS[1:0]	ADC Power Saving control
0 0	High power, high speed
0 1	Medium high power, medium high speed (default)
1 0	Medium low power, medium low speed
1 1	Low power, low speed

Bit 5~4: Reserved. Software must write “0” on these bits when ADCFG3 is written.

Bit 3~2: ARES1~0, ADC data Resolution selection bit 1~0.

ARES[1:0]	ADC Data Resolution Selection
0 0	Reserved
0 1	10-bit Data
1 0	8-bit Data
1 1	Reserved

Note: Hardware default is set to “00b”, please set it to “01b” before start ADC function.

Bit 1 ~ 0: Reserved. Software must write “0” on this bit when ADCFG3 is written.

ADCFG4: ADC Configuration Register 4

SFR Page = **4 only**

SFR Address = **0xC3** RESET = 0000-0000

7	6	5	4	3	2	1	0
0	ADWM0	ADTM3	ADTM2	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: Reserved. Software must write “0” on this bit when ADCFG4 is written.

Bit 6: ADWM0. Mode selection of ADC Window Detector.
0: ADCWI will be set when ADCDH:ADCDL value is within the range defined by WHB and WLB.
1: ADCWI will be set when ADCDH:ADCDL value is outside of the range defined by WHB and WLB.

Bit 5~4: ADC Trigger Mode selection bit 3~2.

ADTM[3:0]	ADC Conversion Start Selection	Source
0 0 0 0	Set ADCS	Software
0 0 0 1	Timer 0 overflow (T0OF)	Timer 0
0 0 1 0	Free running mode	ADC
0 0 1 1	S0 BRG overflow (S0TOF)	S0 BRG
0 1 0 0	KBIET	KBI
0 1 0 1	INT1ET	nINT1
0 1 1 0	Reserved	Reserved
0 1 1 1	INT0ET	nINT0
1 0 0 0	T2EXES	Timer 2
1 0 0 1	AC0ES	AC0
1 0 1 0	Reserved	Reserved
1 0 1 1	Reserved	Reserved
1 1 0 0	PCA0 Overflow (C0TOF)	PCA0 Counter
1 1 0 1	Reserved	Reserved
1 1 1 0	Reserved	Reserved
1 1 1 1	Reserved	Reserved

Bit 3~0: Reserved. Software must write “0” on these bits when ADCFG4 is written.

ADCFG11: ADC Configuration Register 11

SFR Page = **B only**

SFR Address = **0xC3** RESET = 1111-1111

7	6	5	4	3	2	1	0
WHB.3	WHB.2	WHB.1	WHB.0	1	1	1	1
R/W	**R/W**	**R/W**	**R/W**	**W**	**W**	**W**	**W**

ADCFG12: ADC Configuration Register 12

SFR Page = **C only**

SFR Address = **0xC3** RESET = 1111-1111

7	6	5	4	3	2	1	0
WHB.11	WHB.10	WHB.9	WHB.8	WHB.7	WHB.6	WHB.5	WHB.4
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

WHB.9~0: ADC Window High Boundary value.

ADCFG13: ADC Configuration Register 13

SFR Page = **D only**

SFR Address = **0xC3** RESET = 1111-1111

7	6	5	4	3	2	1	0
WLB.3	WLB.2	WLB.1	WLB.0	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**W**	**W**	**W**	**W**

ADCFG14: ADC Configuration Register 14

SFR Page = **E only**

SFR Address = **0xC3** RESET = 0000-0000

7	6	5	4	3	2	1	0
WLB.11	WLB.10	WLB.9	WLB.8	WLB.7	WLB.6	WLB.5	WLB.4
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

WLB.9~0: ADC Window Low Boundary value.

PCON3: Power Control Register 3

SFR Page = **P Only**

SFR Address = 0x45 POR = 0000-0000

7	6	5	4	3	2	1	0
IVREN	0	0	SPWRE	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: IVREN, Internal Voltage Reference Enable.
0: Disable on-chip IVR (1.4V).
1: Enable on-chip IVR (1.4V).

Bit 6~5: Reserved. Software must write “0” on these bits when PCON3 is written.

Bit 3~0: Reserved. Software must write “0” on these bits when PCON3 is written.

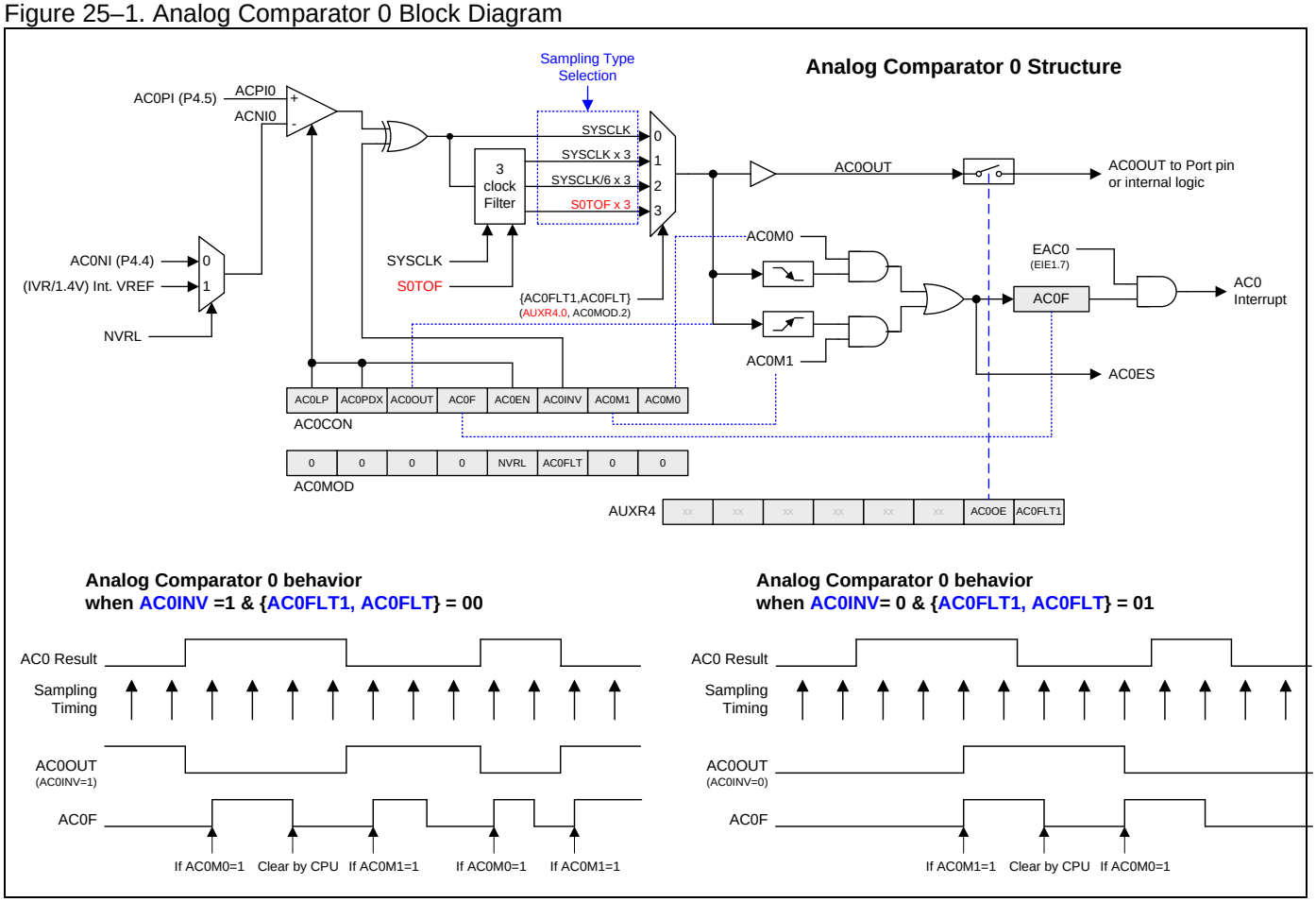
25. Analog Comparator 0 (AC0)

The **MG82F6B08 / 6B001/ 6B104** consist **one** analog comparator modules. It is useful to transfer analog signal into digital information by comparing the voltage level between V_{IN}+ and V_{IN}-. The result of these modules can send to port pin of internal logics.

1. Input Signals on V_{IN}+: It has single I/O input.
2. There are 2 kinds of the reference voltage on V_{IN}- can be used:
   a. From port pin AC0NI: If the application need a much precise voltage, then it can apply a precise voltage source into this I/O pin.
   b. IVR: Internal Voltage Reference, 1.4V.
3. Combine clock filter, which can set 3 different sample rates to filer different noise, which can reduce the software de-bounced effort to improve whole system efficiency.
4. Interrupt Mode selection: The interrupt of AC0 can be triggered by raising edge, falling edge and dual edge.

25.1. AC0 Structure

Figure 25–1. Analog Comparator 0 Block Diagram



25.2. AC0 Register

AC0CON: Analog Comparator 0 Control & Status Register

SFR Page = **0 Only**

SFR Address = 0x9E RESET = 00X0-0000

7	6	5	4	3	2	1	0
AC0LP	AC0PDX	AC0OUT	AC0F	AC0EN	AC0INV	AC0M1	AC0M0
R/W	**R/W**	**R**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: AC0LP, Analog Comparator 0 Low Power Enable.
0: Disable AC0 low power mode.
1: Enable AC0 low power mode.

Bit 6: AC0PDX, Analog Comparator 0 control in PD mode.
0: Program the Analog Comparator 0 to be gated off during PD mode.
1: Program the Analog Comparator 0 to continue its function during PD mode.
If AC0EN, AC0PDX and EAC0 have been set, the comparator in PD function can only wake up CPU in low level or high level mode.

Bit 5: AC0OUT, this is a read only bit from comparator output.

AC0 Input	AC0INV = 0	AC0INV = 1
ACPI0(+) > ACNI0(-)	AC0OUT = 1	AC0OUT = 0
ACPI0(+) < ACNI0(-)	AC0OUT = 0	AC0OUT = 1

Bit 4: AC0F. Analog Comparator 0 Interrupt Flag.
0: The flag must be cleared by software.
1: Set when the comparator output meets the conditions specified by the AC0M [1:0] bits and AC0EN is set. The interrupt may be enabled/disabled by setting/clearing bit 7 of EIE1.

Bit 3: AC0EN. Analog Comparator 0 Enable.
0: Clearing this bit will force the comparator output low and prevent further events from setting AC0F.
1: Set this bit to enable the comparator.

Bit 2: AC0INV, Analog Comparator 0 output inversion bit.
0: AC0 output not inverted.
1: AC0 output inverted.

Bit 1~0: AC0M[1:0], Analog Comparator 0 Interrupt Mode.

AC0M[1:0]	AC0 Interrupt Mode
0 0	Reserved
0 1	Comparator 0 detects output Falling edge
1 0	Comparator 0 detects output Rising edge
1 1	Comparator 0 detects output Toggle

AC0MOD: Analog Comparator 0 Mode Register

SFR Page = **0 Only**

SFR Address = 0x9F RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	NVRL	AC0FLT	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7 ~ 5: Reserved. Software must write “0” on this bit when AC0MOD is written.

Bit 3: NVRL, Negative Voltage Reference Low range select.

Bit 2: AC0FLT, Analog Comparator 0 output Filter control. It selects AC0OUT filter mode with AC0FLT1 (AUXR4.0)

AC0FLT1, AC0FLT	AC0OUT filter mode
0 0	Disabled
0 1	SYSCLK x 3
1 0	SYSCLK/6 x 3
1 1	S0TOF x 3

Bit 1 ~ 0: Reserved. Software must write “0” on this bit when AC0MOD is written.

AUXR10: Auxiliary Register 10

SFR Page = **7 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	AC0HC0	0	SPIPS0	S0PS1	SPFACE	TWICF	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 6: AC0HC0, AC0 Hysteresis Control 0.
0: Disable Hysteresis input on AC0.
1: Enable Hysteresis input on AC0. Default is enabled.

AUXR4: Auxiliary Register 4

SFR Page = **1 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
T2PS1	T2PS0	0	T1PS0	0	0	AC0OE	AC0FLT1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 1: AC0OE, AC0OUT output enable on port pin.
0: Disable AC0OUT output on port pin.
1: Enable AC0OUT output on P1.0.

Bit 0: AC0FLT1, AC0 output Filter control 1.

PCON3: Power Control Register 3

SFR Page = **P Only**

SFR Address = 0x45 POR = 0000-0000

7	6	5	4	3	2	1	0
IVREN	0	0	SPWRE	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

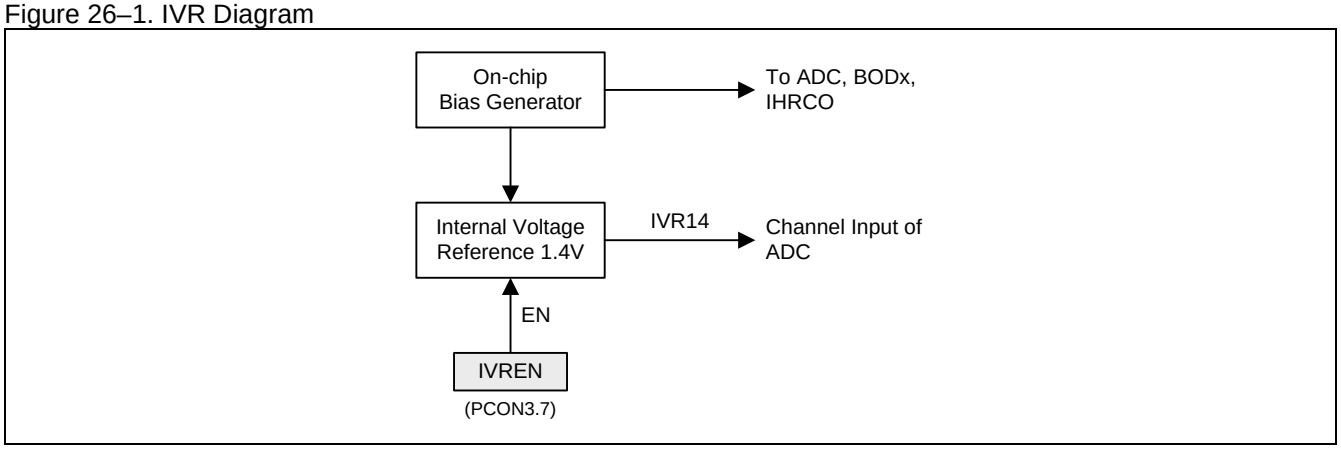
Bit 7: IVREN, Internal Voltage Reference Enable.
0: Disable on-chip IVR (1.4V).
1: Enable on-chip IVR (1.4V).

26. Internal Voltage Reference (IVR, 1.4V)

The IVR can be used as the reference voltage of the AC0 and ADC. The typical output is 1.4V. It can be disabling by IVREN.

26.1. IVR (1.4V) Structure

Figure 26–1. IVR Diagram



26.2. IVR Register

PCON3: Power Control Register 3

SFR Page = **P Only**

SFR Address = 0x45 POR = 0000-0000

7	6	5	4	3	2	1	0
IVREN	0	0	SPWRE	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**W**	**W**	**W**	**W**

Bit 7: IVREN, Internal Voltage Reference Enable.
0: Disable on-chip IVR (1.4V).
1: Enable on-chip IVR (1.4V).

26.3. How to read IVR (1.4V) ADC Pre-stored value

IVR had been trimmed @VDD=3.3V in factory. And its ADC value had been stored in reserved area in Flash ROM for customer calculation the voltage value from the ADC value. It means customer don't needs to do the calibration of the ADC in the production line. It can save the test time and cost. Please reference the following sample code to read the prestored IVR ADC value. And reference "24.2.9 How to improve ADC Accuracy" to understand how to improve ADC measurement accuracy.

Please note this IVR ADC pre-stored value is based on the factory trimmed ADC offset. When using AOS[3:0] to change the offset setting, it should be added to the IVR ADC Pre-stored value to keep the accuracy.

```
void Get_Prestored_IVR(void)
{
	while(PBSY != 0);

	ISPCR = ISP_ENABLE;
	IFMT = 0x06;
	IFADRH = 0x00;
	IFADRL = 0x4C;

	SCMD = 0x46;
	SCMD = 0xB9;
	Trim_IVR_ADC_Value.B[0] = IFD;
	IFADRL ++;

	SCMD = 0x46;
	SCMD = 0xB9;
	Trim_IVR_ADC_Value.B[1] = IFD;

	ISPCR = ISP_DISABLE;
}
```

27. ISP and IAP/EEPROM

The Flash memory of **MG82F6B08 / 6B001/ 6B104** is partitioned into AP-memory, IAP-memory and ISP-memory. AP-memory is used to store user's application program; IAP-memory is used to store the non-volatile application data; and, ISP-memory is used to store the boot loader program for In-System Programming. When MCU is running in ISP region, MCU could modify the AP and IAP memory for software upgraded. If MCU is running in AP region, software could only modify the IAP memory for storage data updated.

MG82F6B08 / 6B001/ 6B104 also provide EEPROM as another storage for the application data. It can be do the Byte program without erase in advance. EEPROM provides more endurance up to 100,000 times.

27.1. MG82F6B08 / 6B001/ 6B104 Flash Memory Configuration

There are total **8K** bytes of Flash Memory in **MG82F6B08 / 6B001/ 6B104.** Figure 27–1 shows the device Flash configuration of **MG82F6B08 / 6B001/ 6B104**. The ISP-memory can be configured as disabled or up to **3.5K** bytes space by hardware option setting with 0.5KB step. The Flash size of IAP memory is located between the IAP low boundary and IAP high boundary. The IAP low boundary is defined by the value of IAPLB register. The IAP high boundary is associated with ISP start address which decides ISP memory size by hardware option. The IAPLB register value is configured by hardware option or AP software programming. All of the AP, IAP and ISP memory are shared the total **8K** bytes Flash memory.

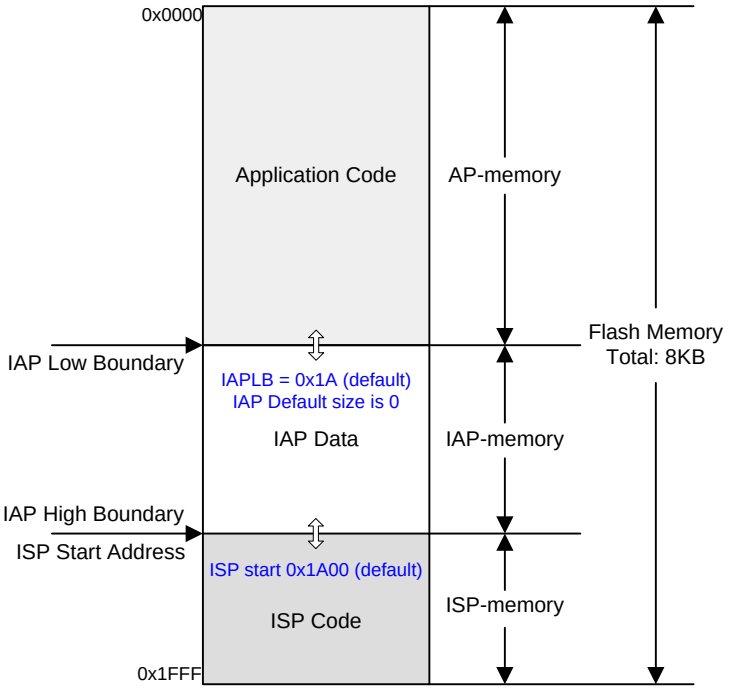
Figure 27–1. **MG82F6B08 / 6B001/ 6B104** Flash Memory Configuration

Note:
(1) ISP Start Address:
0x1A00 if ISP Size = 1.5KB
0x1C00 if ISP Size = 1KB
0x1E00 if ISP Size = 0.5KB

(2) IAP Size :
IAPLB = IAP Low Boundary (ROM High-Byte address)
IAP Start Address = {IAPLB, 00H}
IAP Size = ISP Start Address – IAP Start Address
Set LAPLB = Change IAP Size

(3) If ISP is enabled:
IAP High Boundary = ISP Start Address – 1
IAP Low Boundary = ISP Start Address – IAP Size

(4) If ISP is disabled:
IAP High Boundary = 0x1FFF
IAP Low Boundary = 0x1FFF – IAP Size + 1



Note:

*In default, the **MG82F6B08 / 6B001/ 6B104** that Megawin shipped had configured the Flash memory for 1.5K ISP and Lock enabled. The 1.5K ISP region is inserted Megawin proprietary COMBO ISP code to perform In-System-Programming through Megawin 1-Line ISP protocol and COM port ISP.*

27.2. MG82F6B08 / 6B001/ 6B104 EEPROM Access Flow

There are **2** EEPROM access modes are provided in **MG82F6B08 / 6B001/ 6B104** for application: byte program mode and read mode. MCU software uses these modes to update new data into EEPROM storage and get EEPROM content.
During the access period of EEPROM, MCU can process other task without halting. When access EEPROM, the PBSY will be halt. To check PBSY before access EEPROM can prevent confliction. After the byte read or program has been finished, the EEPF will be set, and it should clear by software. If want to program serval bytes to EEPROM sequentially, EEPF should be checked between bytes.

This section shows the flow chart and demo code for the various EEPROM modes.

To do EEPROM Byte Program

Step 1: Check PBSY = 0 to confirm the EEPROM engine in standby state.
Step 2: Set ISPEN in ISPCR to enable the ISP/IAP/EEPROM flow.
Step 3: Set MS= **0xC8** in IFMT register to select “EE Byte Program Mode”.
Step 4: Fill byte address in IFADRH & IFADRL registers.
Step 5: Fill data to be programmed in IFD register.
Step 6: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an EEPROM processing.
Step 7: Wait EEPF = 1, the process is finished. Then, clear EEPF = 0.
Step 8: Clear ISPEN to 0 and MS= 0x00 to close the ISP/IAP flow.

To do EEPROM Byte Read

Step 1: Check PBSY = 0 to confirm the EEPROM engine in standby state.
Step 2: Set ISPEN in ISPCR to enable the ISP/IAP/EEPROM flow.
Step 3: Set MS= **0xC0** in IFMT register to select “EE Read Mode”.
Step 4: Fill byte address in IFADRH & IFADRL registers.
Step 5: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an EEPROM processing.
Step 6: Now, the EEPROM data is in IFD register.
Step 7: Clear ISPEN to 0 and MS= 0x00 to close the ISP/IAP flow.

The detailed descriptions of Flash page erase, byte program and Flash read in **MG82F6B08 / 6B001/ 6B104** is listed in the following sections:

27.2.1. Notes for EEPROM

Interrupts during EEPROM

Different from ISP/IAP Flash access, when access EEPROM MCU will keep running. It means the interrupt can be used as normal.

Double confirm after data program in EEPROM

For applications involving data correctness, we recommend reading back the data just programmed into the EEPROM for comparison to ensure that it has been programmed correctly.

Accessing Destination of EEPROM

MG82F6B08 has 512 bytes EEPROM. Its address is defined by IFADRH:IFADRL, once the accessing destination is excess 0x1FF, the hardware will automatically neglect the address over 0x200.
For example:
1. If the target address is 0x2FF, then real address of EEPROM will become 0x0FF.
2. If it has 260 byte data is needed to store in EEPROM, it's not suggest to use IFADRL to be the counter, because when IFADRL equal 0xFF, it will not generate overflow to IFADRH.

Endurance for EEPROM

The endurance of the embedded EEPROM is 100,000 erase/write cycles, that is to say, the erase-then-write cycles shouldn't exceed 100,000 times. Thus the user should pay attention to it in the application which needs to frequently update the EEPROM.

27.2.2. EEPROM Byte Program Mode

The any bit in EEPROM data of **MG82F6B08 / 6B001/ 6B104** can be read and write. The targeted EEPROM address is defined in IFADRH and IFADRL. Figure 27–2 shows the EEPROM byte program flow in ISPIAP operation.

Figure 27–2. EEPROM Byte Program Flow

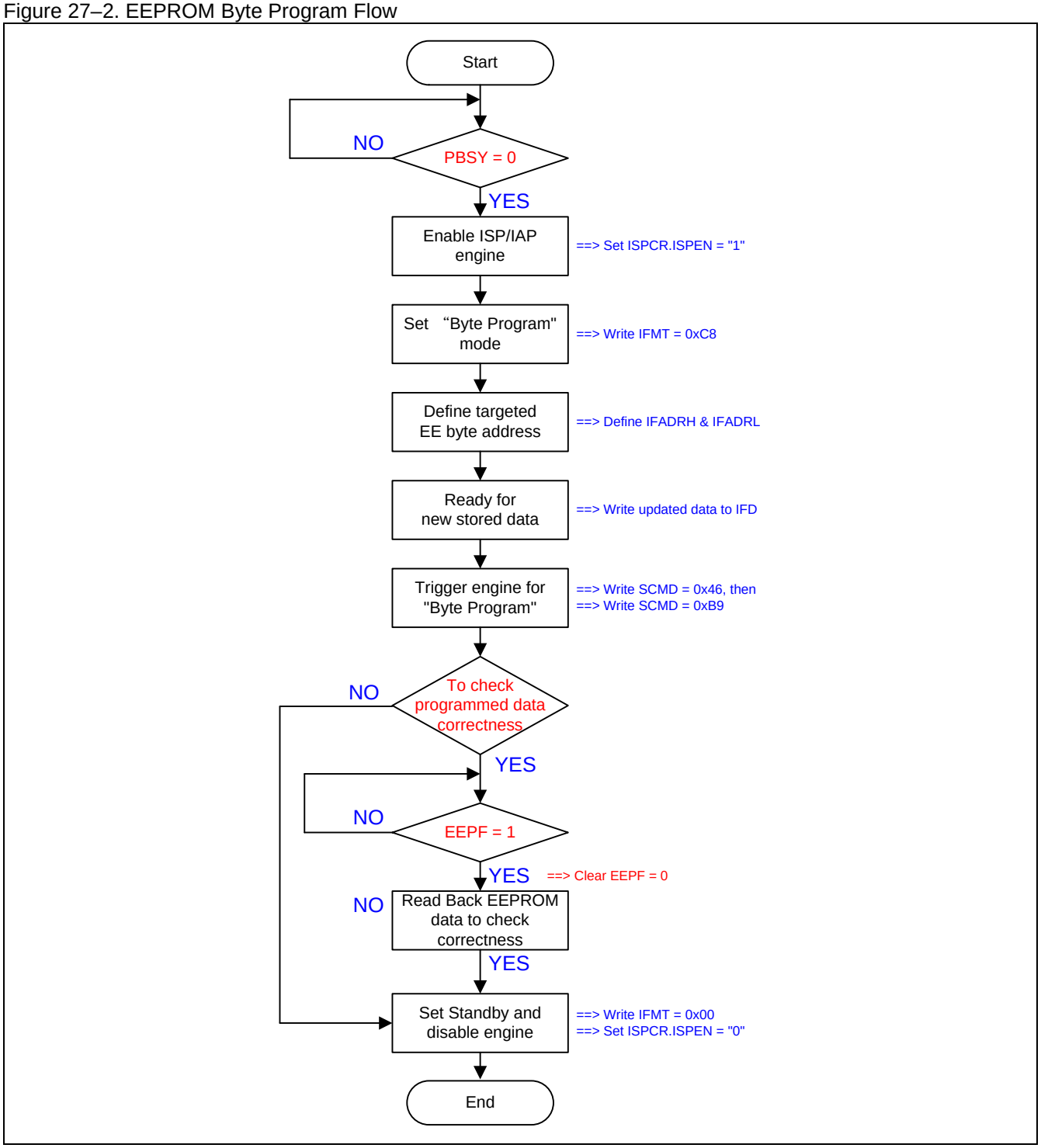


Figure 27–3 shows the demo code of the EEPROM byte program operation.

Figure 27–3. Demo Code for EEPROM Byte Program

```
CHK     PBSY=0              ; Confirm the engine in standby mode

MOV     ISPCR,#10000000b  ; ISPCR.7 = 1, enable ISP
MOV     IFMT,#0C8h          ; select EEPROM Byte Program Mode
MOV     IFADRH,??           ; fill [IFADRH,IFADRL] with byte address
MOV     IFADRL,??           ;
MOV     IFD,??              ;fill the data to be programmed in IFD
MOV     SCMD,#46h           ; trigger ISP/IAP processing
MOV     SCMD,#0B9h          ;

CHK     EEPF=1              ; Confirm the engine in standby mode
ORL     ISPCR,#01h          ;clear EEPF flag and byte program procedure is finish
;

MOV     IFMT,#00h           ; select Standby Mode
MOV     ISPCR,#00000000b ; ISPCR.7 = 0, disable ISP
```

27.2.3. EEPROM Byte Read Mode

The “read” mode of **MG82F6B08 / 6B001/ 6B104** provides the byte read operation from EEPROM memory to get the stored data. The IFADRH and IFADRL point to the physical EEPROM byte address. IFD stores the data which is read from the EEPROM content. It is recommended to verify the EEPROM data by read mode after data programmed.
Figure 27–4 shows the EEPROM byte read flow.

Figure 27–4. EEPROM Byte Read Flow

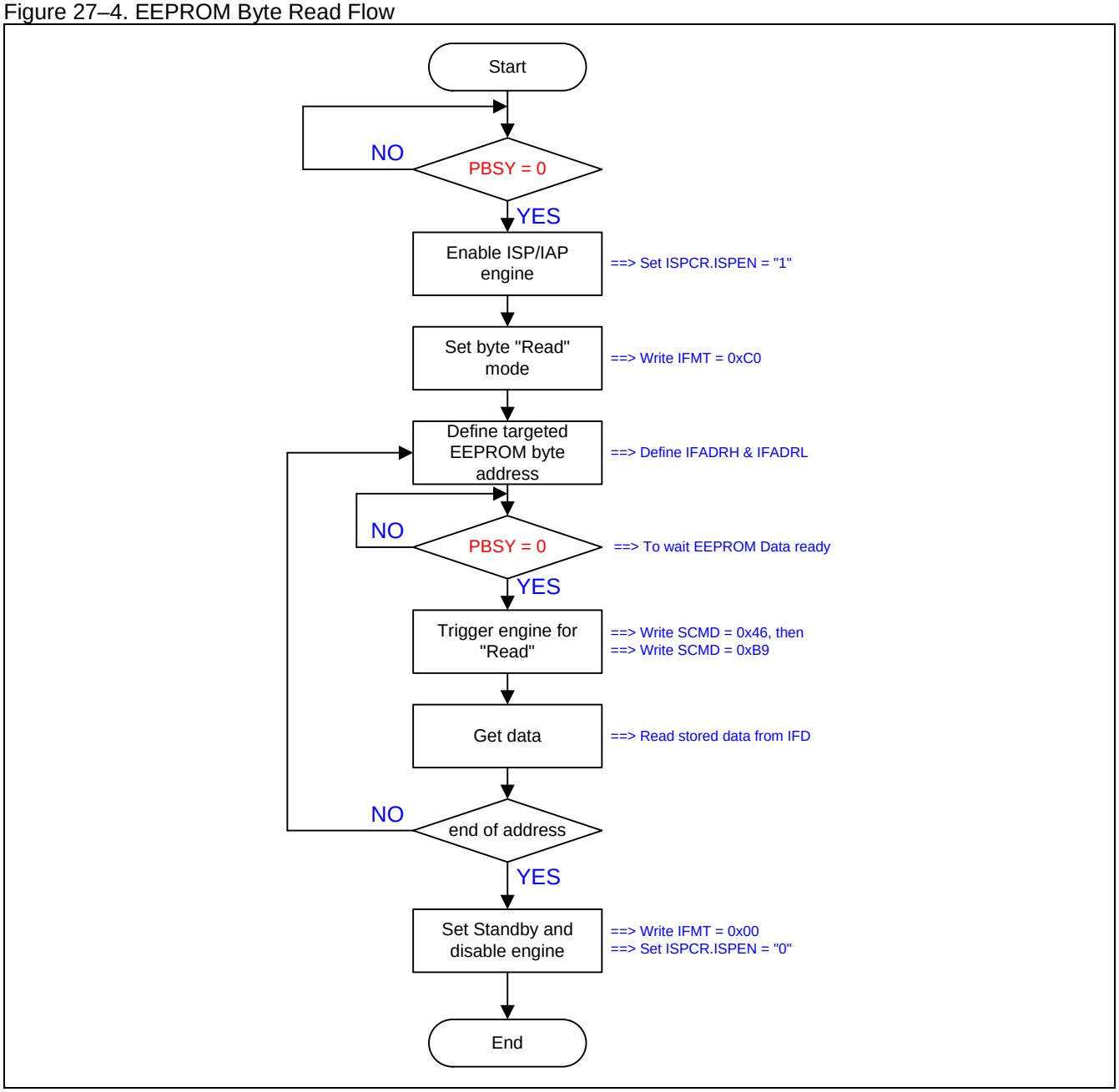


Figure 27–5 shows the demo code of the EEPROM byte read operation.

Figure 27–5. Demo Code for EEPROM Byte Read

```
CHK     PBSY=0              ; Confirm the engine in standby mode

MOV     ISPCR,#10000011b    ; ISPCR.7=1, enable ISP
MOV     IFMT,#0C0h          ; select Read Mode
MOV     IFADRH,??           ; fill [IFADRH,IFADRL] with byte address
MOV     IFADRL,??           ;
MOV     SCMD,#46h           ; trigger ISP/IAP processing
MOV     SCMD,#0B9h          ;

CHK     PBSY=0              ; Confirm the data is ready in IFD
;
MOV     A,IFD               ; now, the read data exists in IFD
MOV     IFMT,#00h           ; select Standby Mode
MOV     ISPCR,#00000000b    ; ISPCR.7 = 0, disable ISP
```

27.3. MG82F6B08 / 6B001/ 6B104 Flash Access in ISP/IAP

There are 3 Flash access modes are provided in **MG82F6B08 / 6B001/ 6B104** for ISP and IAP application: page erase mode, page program mode and read mode. MCU software uses these three modes to update new data into Flash storage and get Flash content.

When the EEPROM is operating, the CPU will not be halt. Once user read or write the ISP/IAP Flash during the EEPROM operation, the CPU will be halt by ISP/IAP related commands and will not continue to execute the ISP/IAP Flash related operations until the EEPROM completes the current operation. For example, if the EEPROM is in the byte programming, it usually takes a few millisecond to complete the action. If to execute the ISP/IAP Flash access at this moment, the CPU will be halt until the ISP/IAP Flash access is completed. Therefore, it is recommended to check PBSY before performing Flash-related accesses to avoid unnecessary CPU waiting.

This section shows the flow chart and demo code for the various Flash modes.

To do Page Erase (64 Bytes per Page)

Step 1: Check PBSY = 0 to confirm the EEPROM engine in standby state.
Step 2: Set ISPEN in ISPCR to enable the ISP/IAP flow.
Step 3: Set MS= **0xC3** in IFMT register to select Page Erase Mode.
Step 4: Fill page address in IFADRH & IFADRL registers.
Step 5: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.
Step 6: Clear ISPEN to 0 and MS= 0x00 to close the ISP/IAP flow.

To do Page Program (64 Bytes per Page)

Step 1: Check PBSY = 0 to confirm the EEPROM engine in standby state.
Step 2: Set ISPEN in ISPCR to enable the ISP/IAP flow.
Step 3: Set MS= **0xC2** in IFMT register to select **Byte Write** Mode.
Step 4: Fill byte address in IFADRH & IFADRL registers.
Step 5: Fill data to be programmed in IFD register.
Step 6: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.
Step 7: Go to “Step 3” to fill the next byte data, repeat 64 cycles for one page data write.
Step 8: Set MS= **0xC1** in IFMT register to select **Page Program** Mode.
Step 9: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.
Step 10: Clear ISPEN to 0 and MS= 0x00 to close the ISP/IAP flow.

To do Byte Read

Step 1: Check PBSY = 0 to confirm the EEPROM engine in standby state.
Step 2: Set ISPEN in ISPCR to enable the ISP/IAP flow.
Step 3: Set MS= **0x01** in IFMT register to select Read Mode.
Step 4: Fill byte address in IFADRH & IFADRL registers.
Step 5: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.
Step 6: Now, the Flash data is in IFD register.
Step 7: Clear ISPEN to 0 and MS= 0x00 to close the ISP/IAP flow.

The detailed descriptions of Flash page erase, byte program and Flash read in **MG82F6B08 / 6B001/ 6B104** is listed in the following sections:

27.3.1. ISP/IAP Flash Page Erase Mode

The any bit in Flash data of **MG82F6B08 / 6B001/ 6B104** only can be programmed to “0”. If user would like to write a “1” into Flash data, the Flash erase is necessary. But the Flash erase in **MG82F6B08 / 6B001/ 6B104** ISP/IAP operation only support “page erase” mode, a page erase will write all data bits to “1” in one page. There are 64 bytes in one page of **MG82F6B08 / 6B001/ 6B104** and the page start address is aligned to A8~A0 = 0x000. The targeted Flash address is defined in IFADRH and IFADRL. So, in Flash page erase mode, the IFADRH.0(A8) and IFADRL.7~0(A7~A0) must be written to “0” for right page address selection. Figure 27–6 shows the Flash page erase flow in ISPIAP operation.

Figure 27–6. ISP/IAP Page Erase Flow

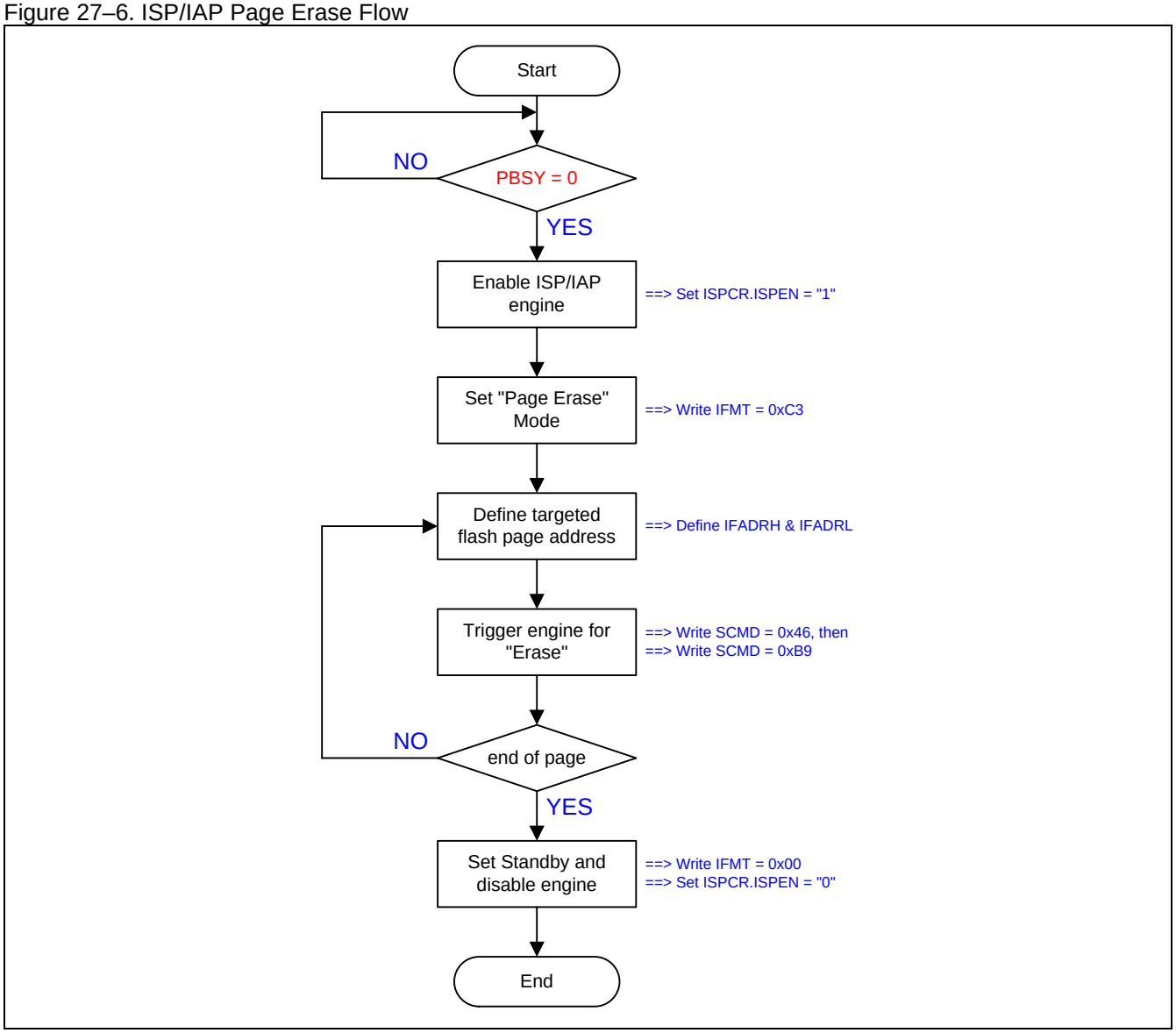


Figure 27–7 shows the demo code of the ISP/IAP page erase operation.

Figure 27–7. Demo Code for ISP/IAP Page Erase

```
CHK     PBSY=0                  ; Confirm the engine in standby mode

MOV     ISPCR,#10000000b ; ISPCR.7 = 1, enable ISP
MOV     IFMT,#0C3h               ; select Page Erase Mode
MOV     IFADRH,??           ; fill [IFADRH,IFADRL] with page address
MOV     IFADRL,??           ;

MOV     SCMD,#46h            ; trigger ISP/IAP processing
MOV     SCMD,#0B9h           ;

;Now, MCU will halt here until processing completed

MOV     IFMT,#00h            ; select Standby Mode
MOV     ISPCR,#00000000b ; ISPCR.7 = 0, disable ISP
```

27.3.2. ISP/IAP Flash Page Program Mode

The “program” mode of **MG82F6B08 / 6B001/ 6B104** provides the buffered byte write operation into Flash memory for new data updated. The IFADRH and IFADRL point to the physical Flash byte address. IFD stores the content which will be programmed into the Flash. Before write the data into Flash, it should be stored in the 64 bytes data latch (To set IFMT = 0xC2 to access the data latch). Figure 27–8 shows the Flash Page program flow in ISP/IAP operation.

Figure 27–8. ISP/IAP Page Program Flow

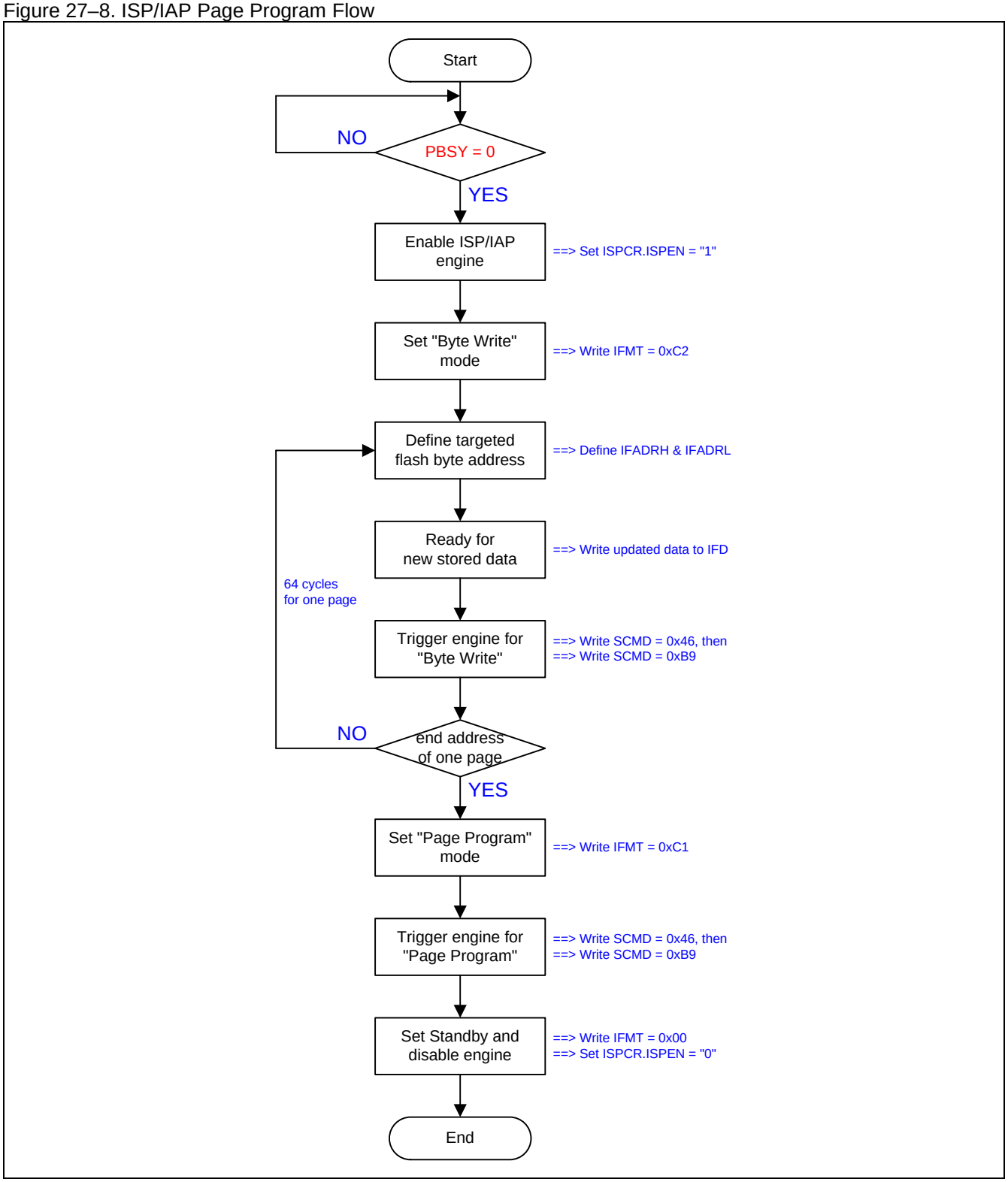


Figure 27–9 shows the demo code of the ISP/IAP page program operation.

Figure 27–9. Demo Code for ISP/IAP Page Program

```
CHK     PBSY=0                  ; Confirm the engine in standby mode

MOV     ISPCR,#10000011b ; ISPCR.7=1, enable ISP
MOV     IFMT,#0C2h               ; select Program Mode
MOV     IFADRH,??           ; fill [IFADRH,IFADRL] with byte address
MOV     IFADRL,??           ;
MOV     IFD,??              ; fill IFD with the data to be programmed
MOV     SCMD,#46h           ;trigger ISP/IAP processing
MOV     SCMD,#0B9h          ;

;Now, MCU will halt here until processing completed

INC     A
CJNE    A,#40h, Byte_Write_Loop     ;64 cycles for one page

MOV     IFMT,#0C1h              ;MS[7: 0]=[1,1,0,0,0,0,0,1], select Page Program Mode
MOV     SCMD,#46h               ;trigger ISP processing
MOV     SCMD,#0B9h              ;
;Now in processing...(CPU will halt here until complete)

MOV     IFMT,#00h               ; select Standby Mode
MOV     ISPCR,#00000000b ; ISPCR.7 = 0, disable ISP
```

27.3.3. How to do ISP/IAP Flash Byte Program

MG82F6B08/6B001/6B104 is not support Flash Byte Program mode. User can allocate 64 bytes data array in RAM to store the data. And then do the page write from RAM to ISP/IAP Flash. Or using EEPROM to store byte data.

27.3.4. ISP/IAP Flash Read Mode

The “read” mode of **MG82F6B08 / 6B001/ 6B104** provides the byte read operation from Flash memory to get the stored data. The IFADRH and IFADRL point to the physical Flash byte address. IFD stores the data which is read from the Flash content. It is recommended to verify the Flash data by read mode after data programmed or page erase.
Figure 27–10 shows the Flash byte read flow in ISP/IAP operation.

Figure 27–10. ISP/IAP Byte Read Flow

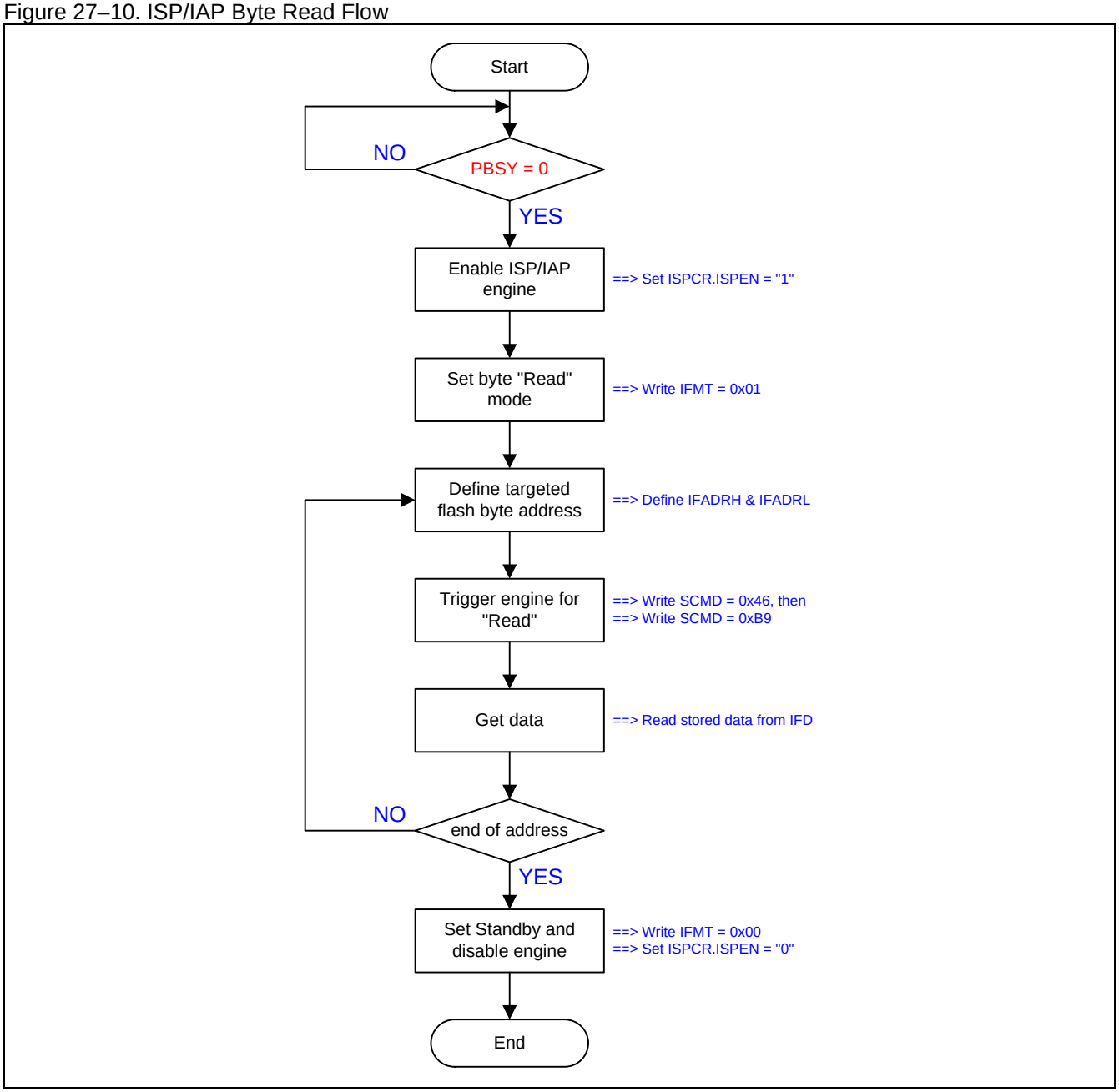


Figure 27–11 shows the demo code of the ISP/IAP byte read operation.

Figure 27–11. Demo Code for ISP/IAP Byte Read

```
CHK     PBSY=0                  ; Confirm the engine in standby mode

MOV     ISPCR,#10000011b ; ISPCR.7=1, enable ISP
MOV     IFMT,#01h                ; select Read Mode
MOV     IFADRH,??                ; fill [IFADRH,IFADRL] with byte address
MOV     IFADRL,??               ;
MOV     SCMD,#46h                 ; trigger ISP/IAP processing
MOV     SCMD,#0B9h              ;
;Now, MCU will halt here until processing completed
MOV     A,IFD                    ; now, the read data exists in IFD
MOV     IFMT,#00h                ; select Standby Mode
MOV     ISPCR,#00000000b ; ISPCR.7 = 0, disable ISP
```

27.4. ISP Operation

ISP means In-System-Programming which makes it possible to update the user's application program (in AP-memory) and non-volatile application data (in IAP-memory) without removing the MCU chip from the actual end product. This useful capability makes a wide range of field-update applications possible. The ISP mode is used in the *loader program* to program both the AP-memory and IAP-memory.

Note:

(1) Before using the ISP feature, the user should configure an ISP-memory space and pre-program the ISP code (boot loader program) into the ISP-memory by a universal Writer/Programmer or Megawin proprietary Writer/Programmer.

(2) ISP code in the ISP-memory can only program the AP-memory and IAP-memory.

After ISP operation has been finished, software writes "001" on ISPCR.7 ~ ISPCR.5 which triggers an software RESET and makes CPU reboot into application program memory (AP-memory) on the address 0x0000.

As we have known, the purpose of the ISP code is to program both AP-memory and IAP-memory. Therefore, ***the MCU must boot from the ISP-memory in order to execute the ISP code***. There are two methods to implement In-System Programming according to how the MCU boots from the ISP-memory.

27.4.1. Hardware approached ISP

To make the MCU directly boot from the ISP-memory when it is just powered on, the MCU's hardware options *HWBS* and *ISP Memory* must be enabled. The ISP entrance method by hardware option is named hardware approached. Once *HWBS* and *ISP Memory* are enabled, the MCU will always boot from the ISP-memory to execute the ISP code (boot loader program) when it is just powered on. The first thing the ISP code should do is to check if there is an ISP request. If there is no ISP requested, the ISP code should trigger a software reset (setting ISPCR.7~5 to "101" simultaneously) to make the MCU re-boot from the AP-memory to run the user's application program..

If the additional hardware option, HWBS2, is enabled with HWBS and ISP Memory, the MCU will always boot from ISP memory after power-on or **external reset finished**. It provides another hardware approached way to enter ISP mode by external reset signal. After first time power-on, **MG82F6B08 / 6B001/ 6B104** can perform ISP operation by external reset trigger and doesn't wait for next time power-on, which suits the non-power-off system to apply the hardware approached ISP function.

27.4.2. Software approached ISP

The software approached ISP to make the MCU boot from the ISP-memory is to trigger a software reset while the MCU is running in the AP-memory. In this case, neither HWBS nor HWBS2 is enabled. The only way for the MCU to boot from the ISP-memory is to trigger a software reset, setting ISPCR.7~5 to "111" simultaneously, when running in the AP-memory. Note: the ISP memory must be configured a valid space by hardware option to reserve ISP mode for software approached ISP application.

27.4.3. Notes for ISP

Developing of the ISP Code

Although the ISP code is programmed in the ISP-memory that has an *ISP Start Address* in the MCU's Flash (see Figure 27–1 for **MG82F6B08 / 6B001/ 6B104**, it doesn't mean you need to put this offset (= *ISP Start Address*) in your source code. The code offset is automatically manipulated by the hardware. User just needs to develop it like an application program in the AP-memory.

Interrupts during ISP

After triggering the ISP/IAP Flash processing, the MCU will halt for a while for internal ISP processing until the processing is completed. At this time, the interrupt will queue up for being serviced if the interrupt is enabled previously. Once the processing is completed, the MCU continues running and the interrupts in the queue will be serviced immediately if the interrupt flag is still active. The user, however, should be aware of the following:

(1) Any interrupt can not be in-time serviced when the MCU halts for ISP processing.
(2) The low/high-level triggered external interrupts, nINTx, should keep activated until the ISP is completed, or they will be neglected.

ISP and Idle mode

MG82F6B08 / 6B001/ 6B104 does not make use of idle-mode to perform ISP function. Instead, it freezes CPU running to release the Flash memory for ISP/IAP engine operating. Once ISP/IAP operation finished, CPU will be resumed and advanced to the instruction which follows the previous instruction that invokes ISP/AP activity.

Accessing Destination of ISP

As mentioned previously, the ISP is used to program both the AP-memory and the IAP-memory. Once the accessing destination address is beyond that of the last byte of the IAP-memory, the hardware will automatically neglect the triggering of ISP processing. That is the triggering of ISP is invalid and the hardware does nothing.

Flash Endurance for ISP

The endurance of the embedded Flash is 20,000 erase/write cycles, that is to say, the erase-then-write cycles shouldn't exceed 20,000 times. Thus the user should pay attention to it in the application which needs to frequently update the AP-memory and IAP-memory.

27.5. In-Application-Programming (IAP)

Although **MG82F6B08 / 6B001/ 6B104** has embedded EEPROM, but if the size is not enough or consider to share the code between other series MCU, user can use the *In Application Programmable* (IAP) to store the Data. Which allows the Flash memory to be used as non-volatile data storage while the application program is running. This useful feature can be applied to the application where the data must be kept after power off.

In fact, the operating of IAP is the same as that of ISP except the Flash range to be programmed is different. The programmable Flash range for ISP operating is located within the AP and IAP memory, while the range for IAP operating is only located within the configured IAP-memory.

Note:

*(1) For **MG82F6B08 / 6B001/ 6B104** IAP feature, the software should specify an IAP-memory space by writing IAPLB in IFMT defined. The IAP-memory space can be also configured by a universal Writer/Programmer or Megawin proprietary Writer/Programmer which configuration is corresponding to IAPLB initial value.*

*(2) The program code to execute IAP is located in the AP-memory and **just only** program IAP-memory **not** ISP-memory.*

27.5.1. IAP-memory Boundary/Range for MG82F6B08 / 6B001/ 6B104

If ISP-memory is specified, the range of the IAP-memory is determined by IAP and the ISP starts address as listed below.

IAP high boundary = ISP start address –1.
IAP low boundary = ISP start address - IAP.

If ISP-memory is not specified, the range of the IAP-memory is determined by the following formula.

*IAP high boundary = **0x1FFF**.*
*IAP low boundary = **0x1FFF** – IAP + 1.*

For example, if ISP-memory is 1K, so that ISP start address is **0x1C00**, and IAP-memory is 1K, then the IAP-memory range is located at **0x1800 ~ 0x1BFF**. The IAP low boundary in **MG82F6B08 / 6B001/ 6B104** is defined by IAPLB register which can be modified by software to adjust the IAP size in user's AP program.

In this chip, the IAP function is disabled in default (IAPLB = 0x1A). If software will access IAP function, only specify the IAPLB to the available Flash zone to enable the function.

27.5.2. Update data in IAP-memory

The special function registers are related to ISP/IAP would be shown in Section "27.6 ISP/IAP/EEPROM Register".

Because the IAP-memory is a part of Flash memory, only ***Page Erase, no Byte Erase***, is provided for Flash erasing. To update "one byte" in the IAP-memory, users can not directly program the new datum into that byte. The following steps show the proper procedure:

Step 1: Save the whole page Flash data (with 64 bytes) into XRAM buffer which contains the data to be updated.
Step 2: Erase this page ***(using ISP/IAP Flash Page Erase mode)***.
Step 3: Modify the new data on the byte(s) in the XRAM buffer.
Step 4: Program the updated data out of the XRAM buffer into this page ***(using ISP/IAP Flash Program mode)***.

To read the data in the IAP-memory, users can use the ***ISP/IAP Flash Read mode*** to get the targeted data.

27.5.3. Notes for IAP

Interrupts during IAP

After triggering the ISP/IAP Flash processing for In-Application Programming, the MCU will halt for a while for internal IAP processing until the processing is completed. At this time, the interrupt will queue up for being serviced if the interrupt is enabled previously. Once the processing is completed, the MCU continues running and the interrupts in the queue will be serviced immediately if the interrupt flag is still active. Users, however, should be aware of the following:

(1) Any interrupt can not be in-time serviced during the MCU halts for IAP processing.
(2) The low/high-level triggered external interrupts, nINTx, should keep activated until the IAP is completed, or they will be neglected.

IAP and Idle mode

MG82F6B08 / 6B001/ 6B104 does not make use of idle-mode to perform IAP function. Instead, it freezes CPU running to release the Flash memory for ISP/IAP engine operating. Once ISP/IAP operation finished, CPU will be resumed and advanced to the instruction which follows the previous instruction that invokes ISP/AP activity.

Double confirm after data program in IAP

For applications involving data correctness, we recommend reading back the data just programmed into the IAP for comparison to ensure that it has been programmed correctly.

Accessing Destination of IAP

As mentioned previously, the IAP is used to program only the IAP-memory. Once the accessing destination is not within the IAP-memory, the hardware will automatically neglect the triggering of IAP processing. That is the triggering of IAP is invalid and the hardware does nothing.

An Alternative Method to Read IAP Data

To read the Flash data in the IAP-memory, in addition to using the Flash Read Mode, the alternative method is using the instruction “MOVC A,@A+DPTR”. Where, DPTR and ACC are filled with the wanted address and the offset, respectively. And, the accessing destination must be within the IAP-memory, or the read data will be indeterminate. Note that using ‘MOVC’ instruction is much faster than using the Flash Read Mode.

Flash Endurance for IAP

The endurance of the embedded Flash is 20,000 erase/write cycles, that is to say, the erase-then-write cycles shouldn’t exceed 20,000 times. Thus the user should pay attention to it in the application which needs to frequently update the IAP-memory.

27.6. ISP/IAP/EEPROM Register

The following special function registers are related to the access of ISP, IAP and Page-P SFR:

IFD: ISP/IAP Flash Data Register

SFR Page = 0~F
SFR Address = 0xE2 RESET = 1111-1111

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IFD is the data port register for ISP/IAP/Page-P operation. The data in IFD will be written into the desired address in operating ISP/IAP/Page-P write and it is the data window of readout in operating ISP/IAP read.

IFADRH: ISP/IAP Address for High-byte addressing

SFR Page = 0~F
SFR Address = 0xE3 RESET = 0000-0000

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IFADRH is the high-byte address port for all ISP/IAP modes. It is not defined in Page-P mode.

IFADRL: ISP/IAP Address for Low-byte addressing

SFR Page = 0~F
SFR Address = 0xE4 RESET = 0000-0000

7	6	5	4	3	2	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

IFADRL is the low byte address port for all ISP/IAP/Page-P modes. In Flash page erase operation, it is ignored.

IFMT: ISP/IAP/EEPROM Mode Table

SFR Page = 0~F
SFR Address = 0xE5 RESET = xxxx-x000

7	6	5	4	3	2	1	0
MS.7	MS.6	MS.5	MS.4	MS.3	MS.2	MS.1	MS.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: ISP/IAP/EEPROM/Page-P operating mode selection

MS[7:0]	Mode
0 0 0 0-0 0 0 0	Standby
0 0 0 0-0 0 0 1	Flash byte read of AP/IAP-memory
0 0 0 0-0 0 1 0	Reserved
0 0 0 0-0 0 1 1	Reserved
0 0 0 0-0 1 0 0	Page P SFR Write
0 0 0 0-0 1 0 1	Page P SFR Read
1 0 0 0-0 0 0 0	Automatic Flash read for CRC.
1 0 0 0-0 0 0 1	Flash byte read with address increased function
1 0 0 0-0 0 1 0	Flash byte write with address increased function. Always stay in target page.
1 1 0 0-0 0 0 0	EEPROM Byte read
1 1 0 0-0 0 0 1	Flash Page Program
1 1 0 0-0 0 1 0	Flash Byte Write in 64 Byte data latch
1 1 0 0-0 0 1 1	Flash Page Erase
1 1 0 0-1 0 0 0	EEPROM Byte Program
Others	Reserved

IFMT is used to select the Flash mode for performing numerous ISP/IAP function or to select page P SFR access. If software selects the mode on automatic Flash read for CRC, the Flash start-address is defined in IFARDH and IFADRH. The Flash end-address is defined at {IAPLB + 9'b1-1111-1111}.

SCMD: Sequential Command Data register

SFR Page = 0~F
SFR Address = 0xE6 RESET = xxxx-xxxx

7	6	5	4	3	2	1	0
SCMD							
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

SCMD is the command port for triggering ISP/IAP/Page-P activity. If SCMD is filled with sequential 0x46h, 0xB9h and if ISPCR.7 = 1, ISP/IAP/Page-P activity will be triggered.

ISPCR: ISP Control Register

SFR Page = 0~F
SFR Address = 0xE7 POR = 0000-0000

7	6	5	4	3	2	1	0
ISPEN	SWBS	SWRST	CFAIL	0	0	PBSY	EEPF
R/W	R/W	R/W	R/W	R/W	R/W	R	R/W

Bit 7: ISPEN, ISP/IAP/Page-P operation enable.
0: Global disable all ISP/IAP/Page-P program/erase/read function.
1: Enable ISP/IAP/Page-P program/erase/read function.

Bit 6: SWBS, software boot selection control.
0: Boot from main-memory after reset.
1: Boot from ISP memory after reset.

Bit 5: SWRST, software reset trigger control.
0: No operation
1: Generate software system reset. It will be cleared by hardware automatically.

Bit 4: CFAIL, Command Fail indication for ISP/IAP operation.
0: The last ISP/IAP command has finished successfully.
1: The last ISP/IAP command fails. It could be caused since the access of Flash memory was inhibited.

Bit 3~2: Reserved. Software must write “0” on these bits when ISPCR is written.

Bit 1: PBSY, Process Busy in EE access flow. Read only.
0: The EEPROM engine is in standby state.
1: The EEPROM engine is in Process Busy state.

Bit 0: EEPF, EEPROM data Program finished Flag.
0: EEPF must be cleared by software.
1: If the EEPROM Program operation is finished, this bit is set by hardware.

IAPLB: IAP Low Boundary

SFR Page = **P Only**

SFR Address = 0x03 RESET = 0111-000x

7	6	5	4	3	2	1	0
IAPLB							0
W	**W**	**W**	**W**	**W**	**W**	**W**	**W**

Bit 7~0: The IAPLB determines the IAP-memory lower boundary. Since a Flash page has 64 bytes, the IAPLB must be an even number.
To read IAPLB, MCU need to define the IFMT for mode selection on IAPLB Read and set ISPCR.ISPEN. And then write 0x46h & 0xB9h sequentially into SCMD. The IAPLB content is available in IFD. If write IAPLB, MCU will put new IAPLB setting value in IFD firstly. And then select IFMT, enable ISPCR.ISPEN and then set SCMD. The IAPLB content has already finished the updated sequence.

The range of the IAP-memory is determined by IAPLB and the ISP start address as listed below.

IAP lower boundary = IAPLB[7:0] x 256, and
IAP higher boundary *= ISP start address – 1.*

For example, if IAPLB=0x16 and ISP start address is 0x1A00, then the IAP-memory range is located at 0x1600 ~ 0x19FF.

Additional attention point, the IAP low boundary address must not be higher than ISP start address.

27.6.1. ISP/IAP Sample Code

The following Figure 27–12 shows a sample code for ISP operation.

Figure 27–12. Sample Code for ISP/IAP

```
;*****************************************************************************************
; Demo Program for the ISP
;*****************************************************************************************
IFD         DATA    0E2h
IFADRH      DATA    0E3h
IFADRL      DATA    0E4h
IFMT        DATA    0E5h
SCMD        DATA    0E6h
ISPCR       DATA    0E7h
;
            MOV     ISPCR,#10000000b    ;ISPCR.7=1, enable ISP

;=========================================================================
; 1. Page Erase Mode (64 bytes per page)
;=========================================================================
            CALL    WAIT_PBSY_FREE
            MOV     IFMT,#0C3h ;MS[7: 0]=[1,1,0,0,0,0,1,1], select Page Erase Mode
            MOV     IFADRH,??    ;fill page address in IFADRH & IFADRL
            MOV     IFADRL,??    ;
            MOV     SCMD,#46h    ;trigger ISP processing
            MOV     SCMD,#0B9h ;
            ;Now in processing...(CPU will halt here until complete)

;=========================================================================
; 2. Page Program Mode (64 bytes per page)
;=========================================================================
            CALL    WAIT_PBSY_FREE
            MOV     IFMT,#0C2h ;MS[7: 0]=[1,1,0,0,0,0,1,0], select Byte Write Mode
            CLR     A
Byte_Write_Loop:
            MOV     IFADRH,??    ;fill byte address in IFADRH & IFADRL
            MOV     IFADRL,??    ;
            MOV     IFD,??       ;fill the data to be programmed in IFD
            MOV     SCMD,#46h    ;trigger ISP processing
            MOV     SCMD,#0B9h ;
            ;Now in processing...(CPU will halt here until complete)
            INC     A
            CJNE    A,#40h, Byte_Write_Loop

            MOV     IFMT,#0C1h ;MS[7: 0]=[1,1,0,0,0,0,0,1], select Page Program Mode
            MOV     SCMD,#46h    ;trigger ISP processing
            MOV     SCMD,#0B9h ;
            ;Now in processing...(CPU will halt here until complete)

;=========================================================================
; 3. Verify using Read Mode
;=========================================================================
            CALL    WAIT_PBSY_FREE
            MOV     IFMT,#01h ;MS[7: 0]=[0,0,0,0,0,0,0,1], select Byte Read Mode
            MOV     IFADRH,??    ;fill byte address in IFADRH & IFADRL
            MOV     IFADRL,?? ;
            MOV     SCMD,#46h    ;trigger ISP processing
            MOV     SCMD,#0B9h ;
            ;Now in processing...(CPU will halt here until complete)
            MOV     A,IFD        ;data will be in IFD
CJNE    A,wanted,ISP_error ;compare with the wanted value
...
WAIT_PBSY_FREE:
            MOV     A,ISPCR ;
            JB      ACC.1,WAIT_PBSY_FREE ;
            RET
ISP_error:
...
;
```

27.6.2. EEPROM Sample Code

The following Figure 27–13 shows a sample code for EEPROM operation.

Figure 27–13. Sample Code for EEPROM

```
;*****************************************************************************
; Demo Program for the EEPROM
;*****************************************************************************
IFD         DATA    0E2h
IFADRH      DATA    0E3h
IFADRL      DATA    0E4h
IFMT        DATA    0E5h
SCMD        DATA    0E6h
ISPCR       DATA    0E7h
;
            MOV     ISPCR,#10000000b    ;ISPCR.7=1, enable ISP


;===========================================================================
; 1. Byte Program Mode
;===========================================================================
            CALL    WAIT_PBSY_FREE
            MOV     IFMT,#0C8h ;MS[7: 0]=[1,1,0,0,1,0,0,0], select EEPROM Byte Program Mode
            MOV     IFADRH,??   ;fill byte address in IFADRH & IFADRL
            MOV     IFADRL,??   ;
            MOV     IFD,??      ;fill the data to be programmed in IFD
            MOV     SCMD,#46h   ;trigger ISP processing
            MOV     SCMD,#0B9h ;
            CALL    WAIT_EEPF_SET
            ORL     ISPCR,#01h ;clear EEPF flag and byte program procedure is finish

;===========================================================================
; 2. Verify using Read Mode
;===========================================================================
            CALL    WAIT_PBSY_FREE
            MOV     IFMT,#0C0h ;MS[7: 0]=[1,1,0,0,0,0,0,0], select EEPROM Byte Read Mode
            MOV     IFADRH,??   ;fill byte address in IFADRH & IFADRL
            MOV     IFADRL,?? ;
            MOV     SCMD,#46h   ;trigger ISP processing
            MOV     SCMD,#0B9h ;
            CALL    WAIT_PBSY_FREE
            MOV     A,IFD       ;data will be in IFD
CJNE    A,wanted,EE_error ;compare with the wanted value
...

WAIT_PBSY_FREE:
            MOV     A,ISPCR ;
            JB      ACC.1,WAIT_PBSY_FREE ;
            RET

WAIT_EEPF_SET:
            MOV     A,ISPCR ;
            JNB     ACC.0, WAIT_EEPF_SET ;
            RET


EE_error:
...
;
```

28. Page P SFR Access

MG82F6B08 / 6B001/ 6B104 builds a special SFR page (Page P) to store the control registers for MCU operation. These SFRs can be accessed by the ISP/IAP operation with different IFMT. In page P access, IFADRH must set to "00" and IFADRL indexes the SFR address in page P. If IFMT= 04H for Page P writing, the content in IFD will be loaded to the SFR in IFADRL indexed after the SCMD triggered. If IFMT = 05H for Page P reading, the content in IFD is stored the SFR value in IFADRL indexed after the SCMD triggered.

Following descriptions are the SFR function definition in Page P:

IAPLB: IAP Low Boundary

SFR Page = **P**

SFR Address = 0x03 RESET = 1111-111x

7	6	5	4	3	2	1	0
IAPLB							0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: The IAPLB determines the IAP-memory lower boundary. Since a Flash page has 64 bytes, the IAPLB must be an even number.
To read IAPLB, MCU need to define the IFADRL for SFR address in Page-P, the IFMT for mode selection on Page-P Read and set ISPCR.ISPEN. And then write 0x46h & 0xB9h sequentially into SCMD. The IAPLB content is available in IFD. If write IAPLB, MCU will put new IAPLB setting value in IFD firstly. And index IFADRL, select IFMT, enable ISPCR.ISPEN and then set SCMD. The IAPLB content has already finished the updated sequence.

The range of the IAP-memory is determined by IAPLB and the ISP Start address as listed below.

IAP lower boundary = IAPLBx256, and
IAP higher boundary = ISP start address – 1.

For example, if IAPLB=0xE0 and ISP start address is 0xF000, then the IAP-memory range is located at 0xE000 ~ 0xEFFF.

Additional attention point, the IAP low boundary address must not be higher than ISP start address.

CKCON2: Clock Control Register 2

SFR Page = **P Only**

SFR Address = 0x40 RESET = 0101-0000

7	6	5	4	3	2	1	0
0	1	0	IHRCOE	0	0	OSCS1	OSCS0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~5: Reserved. Software must write "0" on these bits when CKCON2 is written.

Bit 4: IHRCOE, Internal High frequency RC Oscillator Enable.
0: Disable internal high frequency RC oscillator.
1: Enable internal high frequency RC oscillator. If this bit is set by CPU software, it needs **32 us** to have stable output after IHRCOE is enabled.

Bit 3~2: Reserved. Software must write "0" on these bits when CKCON2 is written.

Bit 1~0: OSCS[1:0], OSCin Source selection.

OSCS[1:0]	OSCin source Selection
0 0	IHRCO
0 1	Reserved
1 0	ILRCO
1 1	ECKI, External Clock Input (P4.5) as OSCin.

CKCON3: Clock Control Register 3

SFR Page = **P only**

SFR Address = 0x41 RESET = 0000-0110

7	6	5	4	3	2	1	0
WDTCS1	WDTCS0	FWKP	WDTFS	MCKD1	MCKD0	1	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: WDTCS1~0. WDT clock source selection.

Bit 5: FWKP, MCU Fast wake up control.
0: Select MCU for normal wakeup time about **90~120us** from power-down mode.
1: Select MCU for fast wakeup time about **15~30us** from power-down mode.

Bit 4: WDTFS. WDT overflow source selection.
0: Select WDT bit-7 overflow as WDT event source.
1: Select WDT bit-0 overflow as WDT event source.

Bit 3~2: MCKD[1:0], MCK Divider Output selection. (MCKDO default is 8MHz)

MCKD[1:0]	MCKDO Frequency	if MCK = 16MHz	if MCK = 22MHz
0 0	MCKDO = MCK	MCKDO = 16MHz	MCKDO = 22MHz
0 1	MCKDO = MCK/2	MCKDO = 8MHz	MCKDO = 11MHz
1 0	MCKDO = MCK/4	MCKDO = 4MHz	MCKDO = 5.5MHz
1 1	MCKDO = MCK/8	MCKDO = 2MHz	MCKDO = 2.75MHz

Bit 1~0: MCDS[1:0], Reserve for test. Software must write “10” on these two bits when CKCON3 is written.

CKCON4: Clock Control Register 4

SFR Page = **P only**

SFR Address = 0x42 RESET = 0000-0000

7	6	5	4	3	2	1	0
RCSS2	RCSS1	RCSS0	RPSC2	RPSC1	RPSC0	RTCCS3	RTCCS2
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~5: RTC Clock Source selection [2:0]

RCSS2, RCSS1, RCSS0	RTC Clock Selection
0 0 0	ECKI (P4.5)
0 0 1	ILRCO
0 1 0	WDTPS
0 1 1	WDTOF
1 0 0	SYSCLK
1 0 1	SYSCLK / 12
1 1 0	Reserved
1 1 1	Reserved

PCON2: Power Control Register 2

SFR Page = **P Only**

SFR Address = 0x44 POR = 0011-0101

7	6	5	4	3	2	1	0
AWBOD1	0	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: AWBOD1, Awaked BOD1 in PD mode.
0: BOD1 is disabled in power-down mode.
1: BOD1 keeps operation in power-down mode.

Bit 6: Reserved. Software must write “0” on this bit when PCON2 is written.

Bit 5~4: BO1S[1:0]. Brown-Out detector 1 monitored level Selection. The initial values of these two bits are loaded from BO1S1O, and BO1S0O in “Hardware Option”. (BO1S2 at PCON4.4)

BO1S[2:0]	BOD1 detecting level
0 0 0	Reserved
0 0 1	2.4V
0 1 0	3.6V
0 1 1	4.2V
1 0 0	2.7V
Others	Reserved

Bit 3: BO1RE, BOD1 Reset Enabled.
0: Disable BOD1 to trigger a system reset when BOF1 is set.
1: Enable BOD1 to trigger a system reset when BOF1 is set.

Bit 2: EBOD1, Enable BOD1 that monitors VDD power dropped at a BO1S1~0 specified voltage level.
0: Disable BOD1 to slow down the chip power consumption.
1: Enable BOD1 to monitor VDD power dropped.

Bit 1: BO0RE, BOD0 Reset Enabled.
0: Disable BOD0 to trigger a system reset when BOF0 is set.
1: Enable BOD0 to trigger a system reset when BOF0 is set (VDD meets 2.35V).

Bit 0: Reserved. Software must write “**1**” on this bit when PCON2 is written.

PCON3: Power Control Register 3

SFR Page = **P Only**
SFR Address = 0x45 POR = 0000-0000

7	6	5	4	3	2	1	0
IVREN	0	0	SPWRE	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: IVREN, Internal Voltage Reference Enable.
0: Disable on-chip IVR (1.4V).
1: Enable on-chip IVR (1.4V).

Bit 6~5: Reserved. Software must write “0” on these bits when PCON3 is written.

Bit 4: SPWRE, SPWF trigger a MCU reset (soft).
0: Disable SPWF to trigger a MCU reset.
1: Enable SPWF to trigger a MCU reset.

Bit 3~0: Reserved. Software must write “0” on these bits when PCON3 is written.

PCON4: Power Control Register 4

SFR Page = **P Only**
SFR Address = 0x46 POR = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	BO1S2	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 4: BOD1 monitored level selection bit 2.

SPCON0: SFR Page Control 0

SFR Page = **P Only**

SFR Address = 0x48 POR = 0000-0000

7	6	5	4	3	2	1	0
0	0	P4CTL	WRCTL	0	CKCTL0	PWCTL1	PWCTL0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: Reserved. Software must write “0” on these bits when SPCON is written.

Bit 5: P4CTL. P4 SFR access Control.
If P4CTL is set, it will disable the P4 SFR modified in Page 0~F. P4 in Page 0~F only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

Bit 4: WRCTL. WDTCR SFR access Control.
If WRCTL is set, it will disable the WDTCR SFR modified in Page 0~F. WDTCR in Page 0~F only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

Bit 2: CKCTL0. CKCON0 SFR access Control.
If CKCTL0 is set, it will disable the CKCON0 SFR modified in Page 0~F. CKCON0 in Page 0~F only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

Bit 1: PWCTL1. PCON1 SFR access Control.
If PWCTL1 is set, it will disable the PCON1 SFR modified in Page 0~F. PCON1 in Page 0~F only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

Bit 0: PWCTL0. PCON0 SFR access Control.
If PWCTL0 is set, it will disable the PCON0 SFR modified in Page 0~F. PCON0 in Page 0~F only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

DCON0: Device Control 0

SFR Page = P Only

SFR Address = 0x4C RESET = 1000-0011

7	6	5	4	3	2	1	0
HSE	IAPO	0	0	0	IORCTL	RSTIO	OCDE
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**W**	**R/W**	**W**

Bit 7: HSE, High Speed operation Enable.
0: Select CPU running in lower speed mode (F_{CPUCLK} ≤ 48KHz) which is slow down internal circuit to reduce power consumption.

Software flow to disable HSE
Step 1: Switch F_{CPUCLK} ≤ 48KHz.
Step 2: Set HSE = 0.

1: Enable CPU full speed operation if F_{CPUCLK} > 48KHz. Before select high frequency clock (> 48KHz) on CPUCLK, software must set HSE to switch internal circuit for high speed operation.

Software flow to enable HSE
Step 1: Set HSE = 1.
Step 2: Switch F_{CPUCLK} > 48KHz.

Bit 6: IAPO, IAP function Only.
0: Maintain IAP region to service IAP function and code execution.
1: Disable the code execution in IAP region and the region only service IAP function.

Bit 5~3: Reserved. Software must write “**0**” on these bits when DCON0 is written.

Bit 2: IORCTL, GPIO Reset Control.
0: Port 6 keeps reset condition for all reset events.
1: If this bit is set, Port 6 is only reset by POR/LVR/Ext Reset/BOR0/BOR1 (if BOR0/1 is enabled).

Bit 1: RSTIO, RST function on I/O,
0: Select I/O pad function for P47.
1: Select I/O pad function for external reset input, RST.

Bit 0: OCDE, OCD enable.
0: Disable OCD interface on P4.4 and P4.5
1: Enable OCD interface on P4.4 and P4.5.

SPHB: Stack Pointer High Boundary

SFR Page = P Only
SFR Address = 0x53 RESET = 1111-1111

7	6	5	4	3	2	1	0
1	1	1	1	SPHB.3	SPHB.2	SPHB.1	SPHB.0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~4: Reserved. Software must write "**1**" on these bits when SPHB is written.

SPHB, it is used for the detection boundary of Stack Pointer warning.
If SPHB == 1111-1111, SPWF will be set when SP ≥ 1111-1111.
If SPHB == 1111-0000, SPWF will be set when SP ≥ 1111-0000.

29. Auxiliary SFRs

AUXR0: Auxiliary Register 0

SFR Page = 0~F
SFR Address = 0xA1 RESET = 0000-0000

7	6	5	4	3	2	1	0
P45OC1	P45OC0	0	PBKF	0	0	INT1H	INT0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: P4.5 function configured control bit 1 and 0. The two bits only act when internal RC oscillator (IHRCO or ILRCO) is selected for system clock source. In external clock input mode, P4.5 is the dedicated clock input pin. In internal oscillator condition, P4.5 provides the following selections for GPIO or clock source generator. When P45OC[1:0] index to non-P4.5 GPIO function, P4.5 will drive the on-chip RC oscillator output to provide the clock source for other devices.

P45OC[1:0]	P4.5 function	I/O mode
0 0	P4.5	By P4M1.5 & P4M0.5
0 1	MCK	By P4M1.5 & P4M0.5
1 0	MCK/2	By P4M1.5 & P4M0.5
1 1	MCK/4	By P4M1.5 & P4M0.5

Please refer Section "8 System Clock" to get the more detailed clock information. For clock-out on P4.5 function, it is recommended to set {P4M1.5, P4M0.5} to "01" which selects P4.5 as push-push output mode.

Bit 5: Reserved. Software must write "**0**" on these bits when AUXR0 is written.

Bit 4: PBKF, PWM Break Flag. This bit is set by PWM break source enabled. If this flag is set, the enabled PWM channel 0~3 will be blocked and the output pins keep the original GPIO state.
0: There is no PWM Break event happened. It is only cleared by software.
1: There is a PWM Break event happened or software triggers a PWM Break.

Bit 3~2: Reserved. Software must write "**0**" on these bits when AUXR0 is written.

Bit 1: INT1H, INT1 High/Rising trigger enable.
0: Remain nINT1 triggered on low level or falling edge on nINT1 port pin.
1: Set nINT1 triggered on high level or rising edge on nINT1 port pin.

Bit 0: INT0H, INT0 High/Rising trigger enable.
0: Remain nINT0 triggered on low level or falling edge on nINT0 port pin.
1: Set nINT0 triggered on high level or rising edge on nINT0 port pin.

AUXR1: Auxiliary Control Register 1

SFR Page = 0~F
SFR Address = 0xA2 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	CRCDS1	CRCDS0	0	0	0	DPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: Reserved. Software must write "**0**" on these bits when AUXR1 is written.

Bit 5~4: CRCDS1~0. CRC0 Data port Selection bit 1~0.

Bit 3~1: Reserved. Software must write "**0**" on these bits when AUXR1 is written.

Bit 0: DPS, DPTR select bit. Use to switch between DPTR0 and DPTR1.
0: Select DPTR0.
1: Select DPTR1.

DPS	Selected DPTR
0	DPTR0
1	DPTR1

AUXR2: Auxiliary Register 2

SFR Page = 0~F

SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
STAF	STOF	0	C0PLK	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: STAF, Start Flag detection of STWI (SID).
0: Clear by firmware by writing "0" on it. STAF might be held within MCU reset period, so needs to clear STAF in firmware initial.
1: Set by hardware to indicate the START condition occurred on STWI bus.

Bit 6: STOF, Stop Flag detection of STWI (SID).
0: Clear by firmware by writing "0" on it.
1: Set by hardware to indicate the STOP condition occurred on STWI bus. STOF might be held within MCU reset period, so needs to clear STOF in firmware initial.

Bit 5: Reserved. Software must write "**0**" on this bit when AUXR2 is written.

Bit 4: C0PLK, PCA0 buffered PWM/COPM update control.
0: Buffered PWM/COPM is auto-updated on PCA0 base timer overflow.
1: Disable the buffered PWM/COPM auto-updated.

Bit 3: T1X12, Timer 1 clock source selector while C/T=0.
0: Clear to select SYSCLK/12.
1: Set to select SYSCLK as the clock source.

Bit 2: T0X12, Timer 1 clock source selector while C/T=0.
0: Clear to select SYSCLK/12.
1: Set to select SYSCLK as the clock source.

Bit 1: T1CKOE, Timer 1 Clock Output Enable.
0: Disable Timer 1 clock output.
1: Enable Timer 1 clock output on T1CKO port pin.

Bit 0: T0CKOE, Timer 0 Clock Output Enable.
0: Disable Timer 0 clock output.
1: Enable Timer 0 clock output on T0CKO port pin.

AUXR3: Auxiliary Register 3

SFR Page = **0 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	T0PS0	BPOC1	BPOC0	S0PS0	TWIPS1	TWIPS0	T0XL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: Reserved. Software must write "**0**" on these bits when AUXR3 is written.

Bit 6: T0PS0, Timer 0 Port pin Selection.

T0PS0	T0/T0CKO
0	P4.6
1	P4.4

Bit 5~4: BPOC1~0, Beeper output control bits.

BPOC[1:0]	P4.4 function	I/O mode
0 0	P4.4	By P4M0.4 & P4M1.4
0 1	ILRCO/32	By P4M0.4 & P4M1.4
1 0	ILRCO/16	By P4M0.4 & P4M1.4
1 1	ILRCO/8	By P4M0.4 & P4M1.4

For beeper on P4.4 function, it is recommended to configure P4.4 as push-push output mode.

Bit 3: S0PS0, Serial Port 0 pin Selection 0. (S0PS1 at AUXR10.3, SnMIPS at AUXR6.1)

SnMIPS, S0PS1~0	RXD0/ S0MOSI	TXD0/ S0SPCK	S0MISO	S0nSS
0 0 0	P3.0	P3.1	P33	P46
0 0 1	P4.4	P4.5	P33	P46
0 1 0	P4.5	P3.0	P33	P46
0 1 1	P4.5	P4.4	P33	P46
1 0 0 (1XX)	P4.4	P3.3	P31	P30

Bit 2~1: TWIPS1~0, TWI0/I2C0 Port pin Selection [1:0].

TWIPS1~0	TWI0_SCL	TWI0_SDA
0 0	P3.1	P3.0
0 1	P4.4	P4.5
1 0	P3.0	P3.1
1 1	P3.3	P4.6

Bit 0: T0XL is the Timer 0 clock source selection bit. Please refer T0X12 for T0XL function definition.

AUXR4: Auxiliary Register 4

SFR Page = **1 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
T2PS1	T2PS0	0	T1PS0	0	0	AC0OE	AC0FLT1
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~6: T2PS1~0, Timer 2 Port pin Selection [1:0].

T2PS1~0	T2/T2CKO	T2EX
0 0	P3.1	P3.0
0 1	P3.3	P4.6
1 0	P4.6	P3.3
1 1	P4.5	P4.4

Bit 5: Reserved. Software must write "**0**" on these bits when AUXR4 is written.

Bit 4: T1PS1~0, Timer 1 Port pin Selection.

T1PS0	T1/T1CKO
0	P3.3
1	P4.5

Bit 1: AC0OE, AC0OUT output enable on port pin.
0: Disable AC0OUT output on port pin.
1: Enable AC0OUT output on P1.0.

Bit 0: AC0FLT1, AC0 output Filter control 1.

AUXR5: Auxiliary Register 5

SFR Page = **2 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	C0IC2S0	0	0	C0PS1	C0PS0	ECIPS0	C0COPS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: Reserved. Software must write "**0**" on these bits when AUXR5 is written.

Bit 6: C0IC2S0, PCA0 Input Channel 2 input port pin Selection.

C1IC2S0	CEX2 input
0	CEX2 Port Pin
1	T2EXI

Bit 5 ~ 4: Reserved. Software must write "**0**" on these bits when AUXR5 is written.

Bit 3 ~ 2: C0PS[1:0], PCA0 Port pin Selection 1 & 0.

C0PS1~0	CEX0	CEX1	CEX2	CEX3
0 0	P3.0	P3.3	P3.1	P4.6
0 1	P4.4	P3.3	P4.5	P4.6
1 0	P4.4	P3.3	P3.0	P4.6
1 1	P4.4	P3.3	P1.0	P4.6

Bit 1: ECIPS0, PCA0 ECI Port pin Selection0.

ECIPS0	ECI
0	P4.4
1	P1.0

Bit 0: C0COPS, PCA0 Clock Output (C0CKO) port pin Selection.

C0COPS	C0CKO
0	P4.7
1	P3.3

AUXR6: Auxiliary Register 6

SFR Page = **3 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	0	0	0	0	T2FCS	SnMIPS	S0COPS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7 ~ 3: Reserved. Software must write "**0**" on these bits when AUXR6 is written.

Bit 3: T3FCS, Reserved for chip test.

Bit 2: T2FCS, Reserved for chip test.

Bit 1: SnMIP, Serial Port 0 pin Selection 0.

Bit 0: S0COPS, S0BRG Clock Output (S0CKO) port pin Selection.

S0COPS	S0CKO
0	P4.7
1	P3.3

AUXR7: Auxiliary Register 7

SFR Page = **4 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	1	C0CKOE	SPI0M0	0	0	0	0
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 5: C0CKOE, PCA0 clock output (C0CKO) enable.
0: Disable PCA0 clock output.
1: Enable PCA0 clock output with PCA0 base timer overflow rate/2.

Bit 4: SPI0M0, SPI0 model control bit 0. It controls the SPI application with daisy-chain connection.
0: Disable the mode control.
1: Enable the mode control.

AUXR9: Auxiliary Register 9

SFR Page = 6 only

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
0	SIDPS0	T1G1	T0G1	C0FDC1	C0FDC0	0	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: Reserved. Software must write "**0**" on these bits when AUXR9 is written.

Bit 6: SID/STWI Port pin Selection.

SIDPS0	STWI_SCL	STWI_SDA
0	nINT1	S0MI
1	TWI0_SCL	TWI0_SDA

Bit 5: T1G1, Gating source selection of Timer 1.

T1G1, T1GATE	T1 Gate source
0 0	Disable
0 1	INT1 active
1 0	TF2 active
1 1	KBI active

Bit 4: T0G1, Gating source selection of Timer 0.

T0G1, T0GATE	T0 Gate source
0 0	Disable
0 1	INT0 active
1 0	TF2 active
1 1	KBI active

Bit 3~2: C0FDC1~0, C0FDCK Selection [1:0].

C0FDC1~0	C0FDCK
0 0	T0OF
0 1	T1OF
1 0	T2OF
1 1	S0TOF

Bit 1~0: Reserved. Software must write "**0**" on these bits when AUXR9 is written.

AUXR10: Auxiliary Register 10

SFR Page = **7 only**

SFR Address = 0xA4 RESET = 1100-0000

7	6	5	4	3	2	1	0
1	AC0HC0	0	SPIPS0	S0PS1	SPFACE	TWICF	0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 6: AC0HC0, AC0 Hysteresis Control 0.
0: Disable Hysteresis input on AC0.
1: Enable Hysteresis input on AC0. Default is enabled.

Bit 3: S0PS1, Serial Port 0 pin Selection 1. (Its function is illustrated at AUXR3.3, S0PS0)

Bit 2: SPFACE, SPIF Auto-Cleared Enable.
0: Disable, SPIF is only cleared by CPU software.
1: Enable. SPIF is also cleared by CPU read/write SPDAT operation.

Bit 1: TWICF, TWI0/I2C0 serial Clock input Filter.
0: Disable TWICF function.
1: Enable TWICF function.

AUXR11: Auxiliary Register 11

SFR Page = **8 only**

SFR Address = 0xA4 RESET = 0000-0000

7	6	5	4	3	2	1	0
P30AM	P33AM	0	0	0	POEM0	C0M0	C0OFS
R/W	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7: P30AM, P3.0 Analog input Mode enable.
0: The P3.0 GPIO mode is controlled by P3M0 and P3M1.
1: Force P3.0 to be analog input mode for the AIN4 input of ADC12.

Bit 6: P33AM, P3.3 Analog input Mode enable.
0: The P3.3 GPIO mode is controlled by P3M0 and P3M1.
1: Force P3.3 to be analog input mode for the AIN5 input of ADC12.

Bit 2: POEM0, PCA0 POEn control 0.
0: POEn function is active immediately after CPU writing.
1: POEn function is aligned to PWM cycle.

Bit 1: C0M0, PCA0 Mode control 0.
0: Not support variable resolution on central aligned PWM.
1: Enable PCA0 to support variable resolution on central aligned PWM. To enable this function, the PCAE (PWMCR.7) also needs to be set.

Bit 0: C0OFS, PCA0 overflow flag selection when C0M0 is enabled.
0: CF is set on the bottom of PWM cycle.
1: CF is set on the top of PWM cycle.

SFRPI: SFR Page Index Register

SFR Page = **0~F**

SFR Address = 0xAC RESET = xxxx-0000

7	6	5	4	3	2	1	0
--	--	--	--	IDX3	IDX2	IDX1	IDX0
W	**W**	**W**	**W**	**R/W**	**R/W**	**R/W**	**R/W**

Bit 7~4: Reserved. Software must write “0” on these bits when SFRPI is written.

Bit 3~0: SFR Page Index.

IDX[3:0]	Selected Page
0000	Page 0
0001	Page 1
0010	Page 2
0011	Page 3
……	……
……	……
……	……
1111	Page F

30. Hardware Option

The MCU's Hardware Option defines the device behavior which cannot be programmed or controlled by software. The hardware options can only be programmed by a Universal Programmer, the "Megawin 8051 Writer U1" or the "Megawin 8051 ICE Adapter" (The ICE adapter also supports ICP programming function. Refer Section "31.4 In-Chip-Programming Function"). After whole-chip erased, all the hardware options are left in "disabled" state and there is no ISP-memory and IAP-memory configured. The **MG82F6B08 / 6B001/ 6B104** has the following Hardware Options:

LOCK:
☑: Enabled. Code dumped on a universal Writer or Programmer is locked to 0xFF for security.
☐: Disabled. Not locked.

ISP-memory Space:
The ISP-memory space is specified by its starting address. And, its higher boundary is limited by the Flash end address, i.e., **0x1FFF**. The following table lists the ISP space option in this chip. In default setting, **MG82F6B08 / 6B001/ 6B104** ISP space is configured to **1.5K** that had been embedded Megawin proprietary COMBO ISP code to perform device firmware upgrade through Megawin 1-Line ISP protocol and COM port ISP.

ISP-memory Size	**MG82F6B08 / 6B001/ 6B104** ISP Start Address
3.5K bytes	1200
3.0K bytes	1400
2.5K bytes	1600
2.0K bytes	1800
1.5K bytes	1A00
1.0K bytes	1C00
0.5K bytes	1E00
No ISP Space	--

HWBS:
☑: Enabled. When powered up, MCU will boot from ISP-memory if ISP-memory is configured.
☐: Disabled. MCU always boots from AP-memory.

HWBS2:
☑: Enabled. Not only power-up but also any reset will cause MCU to boot from ISP-memory if ISP-memory is configured.
☐: Disabled. Where MCU boots from is determined by HWBS.

IAP-memory Space:
The IAP-memory space specifies the user defined IAP space. The IAP-memory Space can be configured by hardware option or MCU software by modifying IAPLB. In default, it is configured to 0 bytes.

BO1S2O, BO1S1O, BO1S0O:
☑,☑,☐: Select BOD1 to detect 2.4V.
☑,☐,☑: Select BOD1 to detect 3.6V.
☑,☐,☐: Select BOD1 to detect 4.2V.
☐,☑,☑: Select BOD1 to detect 2.7V.

BO0REO:
☑: Enabled. BOD0 will trigger a RESET event to CPU on AP program start address. (1.7V)
☐: Disabled. BOD0 can not trigger a RESET to CPU.

BO1REO:
☑: Enabled. BOD1 will trigger a RESET event to CPU on AP program start address. (4.2V, 3.6V, 2.4V or 2.7V)
☐: Disabled. BOD1 can not trigger a RESET to CPU. If want to use software to set BOD1 detected voltage, do not enable BO1REO. Please reference "11.5 Brown-Out Reset" for detail.

WRENO:
☑: Enabled. Set WDTCR.WREN to enable a system reset function by WDTF.
☐: Disabled. Clear WDTCR.WREN to disable the system reset function by WDTF.

NSWDT: Non-Stopped WDT
☑: Enabled. Set WDTCR.NSW to enable the WDT running in power down mode (watch mode).
☐: Disabled. Clear WDTCR.NSW to disable the WDT running in power down mode (disable Watch mode).

HWENW: Hardware loaded for "ENW" of WDTCR.
☑: Enabled. Enable WDT and load the content of WRENO, NSWDT, HWWIDL and HWPS2~0 to WDTCR after power-on.
☐: Disabled. WDT is not enabled automatically after power-on.

HWWIDL, HWPS2, HWPS1, HWPS0:
When HWENW is enabled, the content on these four fused bits will be loaded to WDTCR SFR after power-on.

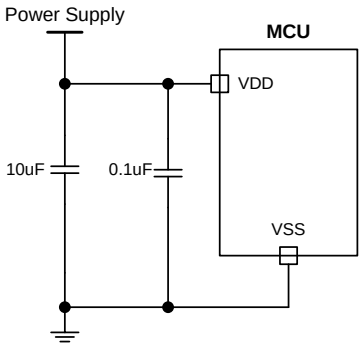
WDSFWP:
☑: Enabled. The WDT SFRs, WREN, NSW, WIDL, PS2, PS1 and PS0 in WDTCR, will be write-protected.
☐: Disabled. The WDT SFRs, WREN, NSW, WIDL, PS2, PS1 and PS0 in WDTCR, are free for writing of software.

31. Application Notes

31.1. Power Supply Circuit

To have the **MG82F6B08 / 6B001/ 6B104** work with power supply varying from 2.4V to 5.5V, adding some external decoupling and bypass capacitors is necessary, as shown in Figure 31–1.

Figure 31–1. Power Supplied Circuit



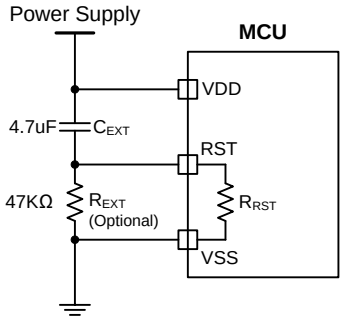
31.2. Reset Circuit

Normally, the power-on reset can be successfully generated during power-up. However, to further ensure the MCU a reliable reset during power-up, the external reset is necessary. Figure 31–2 shows the external reset circuit, which consists of a capacitor C_{EXT} connected to VDD (power supply) and a resistor R_{EXT} connected to VSS (ground).

In general, R_{EXT} is optional because the RST pin has an internal pull-down resistor (R_{RST}). This internal diffused resistor to VSS permits a power-up reset using only an external capacitor C_{EXT} to VDD.

See Section "32.2 DC Characteristics" for R_{RST} value.

Figure 31–2. Reset Circuit



31.3. ICP and OCD Interface Circuit

MG82F6B08 / 6B001/ 6B104 devices include an on-chip Megawin proprietary debug interface to allow In-Chip-Programming (ICP) and in-system On-Chip-Debugging (OCD) with the production part installed in the end application. The ICP and OCD share the same interface to use a clock signal (ICP_SCL/OCD_SCL) and a bi-directional data signal (ICP_SDA/OCD_SDA) to transfer information between the device and a host system.

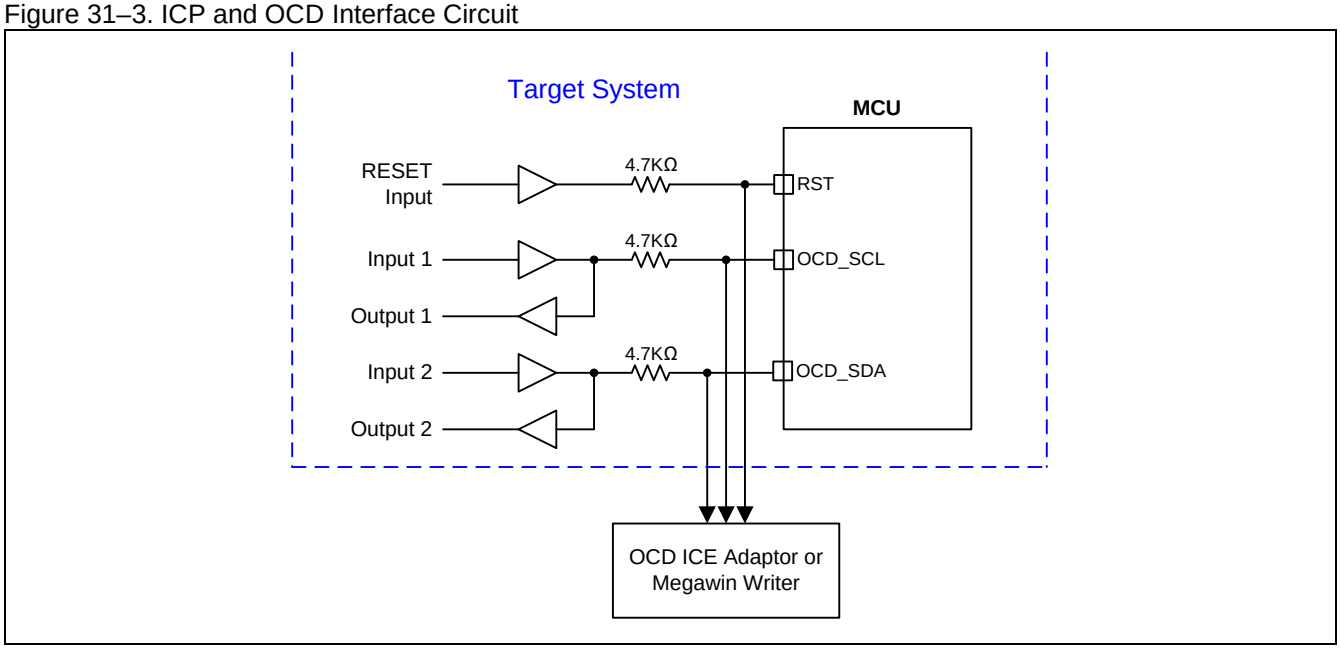
The ICP interface allows the ICP_SCL/ICP_SDA pins to be shared with user functions so that In-Chip Flash Programming function could be performed. This is practicable because ICP communication is performed when the device is in the halt state, where the on-chip peripherals and user software are stalled. In this halted state, the ICP interface can safely 'borrow' the ICP_SCL (P4.4) and ICP_SDA (P4.5) pins. In most applications, external resistors are required to isolate ICP interface traffic from the user application. A typical isolation configuration is shown in Figure 31–3.

It is strongly recommended to build the ICP interface circuit on target system. It will reserve the whole capability for software programming and device options configured.

After power-on, the P4.4 and P4.5 of **MG82F6B08 / 6B001/ 6B104** are configured to OCD_SCL/OCD_SDA for in-system On-Chip-Debugging function. This is possible because OCD communication is typically performed when the CPU is in the halt state, where the user software is stalled. In this halted state, the OCD interface can safely 'use' the OCD_SCL (P4.4) and OCD_SDA (P4.5) pins. As mentioned ICP interface isolation in Figure 31–3, external resistors are required to isolate OCD interface traffic from the user application.

If user gives up the OCD function, software can configure the OCD_SCL and OCD_SDA to port pins: P4.4 and P4.5 by clearing OCDE on bit 0 of DCON0. When user would like to regain the OCD function, user can predict an event that triggers the software to switch the P4.4 and P4.5 back to OCD_SCL and OCD_SDA by setting OCDE as "1". Or "Erase" the on-chip Flash by ICP which cleans the user software to stop the port pins switching.

Figure 31–3. ICP and OCD Interface Circuit



31.4. In-Chip-Programming Function

The ICP, like the traditional parallel programming method, can be used to program anywhere in the MCU, including the Flash and MCU's Hardware Option. And, owing to its dedicated serial programming interface (via the On-Chip Debug path), the ICP can update the MCU without removing the MCU chip from the actual end product, just like the ISP does.

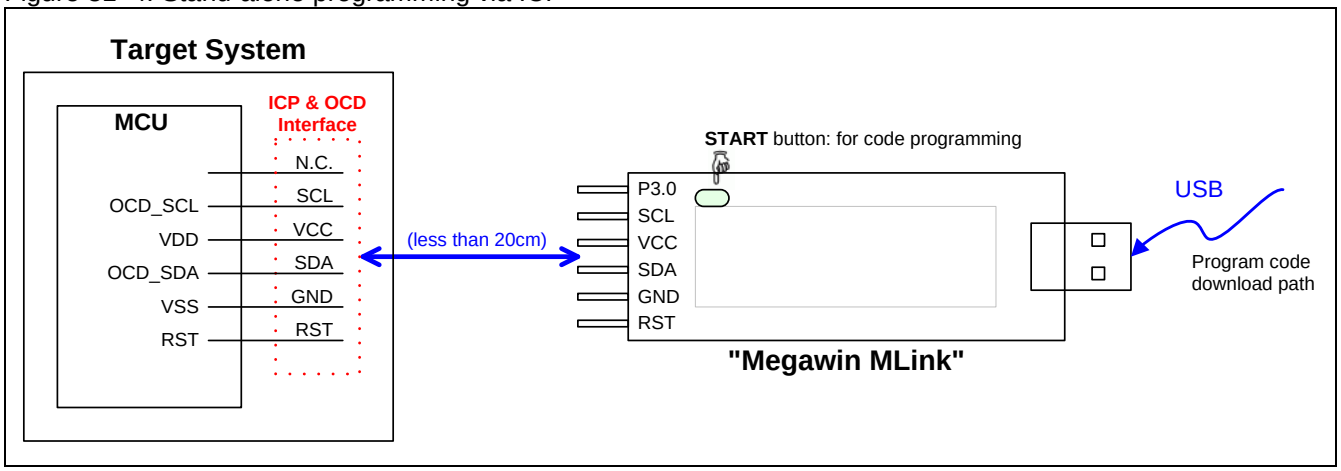
The proprietary 6-pin "Megawin MLink" can support the In-Circuit Programming of **MG82F6B08 / 6B001/ 6B104**. "Megawin MLink" has the in-system storage to store the user program code and device options. So, the tools can perform a portable and stand-alone programming without a host on-line, such as connecting the tool to PC. Following lists the features of the ICP function:

Features

- No need to have a loader program pre-programmed in the target MCU.
- Dedicated serial interface; no port pin is occupied.
- The target MCU needn't be in running state; it just needs to be powered.
- Capable of portable and stand-alone working without host's intervention.

The above valuable features make the ICP function very friendly to the user. Particularly, it is capable of stand-alone working after the programming data is downloaded. This is especially useful in the field without a PC. The system diagrams of the ICP function for the stand-alone programming are shown in Figure 31–4. Only **five** pins are used for the ICP interface: the SDA line and SCL line function as serial data and serial clock, respectively, to transmit the programming data from the 6-pin "Megawin MLink" to the target MCU; the RST line to halt the MCU, and the VCC & GND are the power supply entry of the 6-pin "Megawin MLink" for portable programming application. The USB connector can be directly plugged into the PC's USB port to download the programming data from PC to the 6-pin "Megawin MLink".

Figure 31–4. Stand-alone programming via ICP



31.5. On-Chip-Debug Function

The **MG82F6B08 / 6B001/ 6B104** is equipped with a Megawin proprietary On-Chip Debug (OCD) interface for In-Circuit Emulator (ICE). The OCD interface provides on-chip and in-system non-intrusive debugging without any target resource occupied. Several operations necessary for an ICE are supported, such as Reset, Run, Stop, Step, Run to Cursor and Breakpoint Setting.

Using the OCD technology, Megawin provides the “Megawin MLink” for the user, as shown in Figure 31–5. The user has no need to prepare any development board during developing, or the socket adapter used in the traditional ICE probe. All the thing the user needs to do is to reserve a 6-pin connector on the system for the dedicated OCD interface: P3.0, RST, VCC, OCD_SDA, OCD_SCL and GND as shown in Figure 31–5.

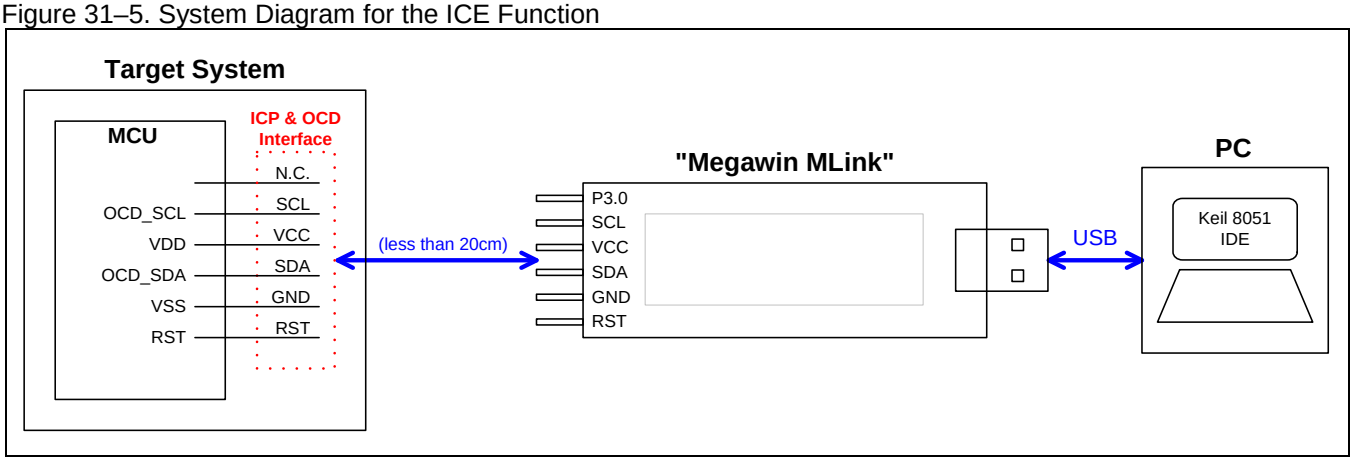
In addition, the most powerful feature is that it can directly connect the user’s target system to the Keil 8051 IDE software for debugging, which directly utilizes the Keil IDE’s dScope-Debugger function. Of course, all the advantages are based on your using Keil 8051 IDE software.

Note: “Keil” is the trade mark of “Keil Elektronik GmbH and Keil Software, Inc.”.

Features

- Megawin proprietary OCD (On-Chip-Debug) technology
- On-chip & in-system real-time debugging
- 5-pin dedicated serial interface for OCD, no target resource occupied
- Directly linked to the debugger function of the Keil 8051 IDE Software
- USB connection between target and host (PC)
- Helpful debug actions: Reset, Run, Stop, Step and Run to Cursor
- Programmable breakpoints, up to 4 breakpoints can be inserted simultaneously
- Several debug-helpful windows: Register/Disassembly/Watch/Memory Windows
- Source-level (Assembly or C-language) debugging capability

Figure 31–5. System Diagram for the ICE Function



Note: For more detailed information about the OCD ICE, please feel free to contact Megawin.

32. Electrical Characteristics

32.1. Absolute Maximum Rating

Parameter	Rating	Unit
Ambient temperature under bias	-40 ~ +85	°C
Storage temperature	-65 ~ + 150	°C
Voltage on any Port I/O Pin or RST with respect to VSS	-0.5 ~ VDD + 0.5	V
Voltage on VDD with respect to VSS	-0.5 ~ +6.0	V
Maximum total current through VDD and VSS	200	mA
Maximum output current sunk by any Port pin	40	mA

*Note: stresses above those listed under “Absolute Maximum Ratings” may cause permanent damage to the device. This is a stress rating only and functional operation of the devices at those or any other conditions above those indicated in the operation listings of this specification is not implied. Exposure to maximum rating conditions for extended periods may affect device reliability.

32.2. DC Characteristics

VDD = 5.0V±10%, VSS = 0V, T_A = 25 °C and execute NOP for each machine cycle, unless otherwise specified

Symbol	Parameter	Test Condition	Limits			Unit
			min	typ	max	
Input/ Output Characteristics						
V_{IH1}	Input High voltage (All I/O Ports)	Except RST	0.6			VDD
V_{IH2}	Input High voltage (RST)		0.75			VDD
V_{IL1}	Input Low voltage (All I/O Ports)	Except RST			0.15	VDD
V_{IL2}	Input Low voltage (RST)				0.2	VDD
I_{IH}	Input High Leakage current (All I/O Ports)	V_{PIN} = VDD		0	±1	uA
I_{IL1}	Logic 0 input current (P3 in quasi-mode)	V_{PIN} = 0.4V		-20	-30	uA
I_{IL2}	Logic 0 input current (All Input only or open-drain Ports)	V_{PIN} = 0.4V		0	-1	uA
I_{H2L}	Logic 1 to 0 input transition current (P3 in quasi-mode) (2)	V_{PIN} =1.8V		-300	-450	uA
I_{OH1}	Output High current (P3 in quasi-Mode)	VDD=5V; V_{PIN} =2.4V	-180	-285		uA
		VDD=3.3V; V_{PIN} =2.4V	-50	-80		uA
		VDD=2.4V; V_{PIN} =2.0V	-20	-28		uA
I_{OH2}	Output High current (All push-pull output ports)	VDD=5V; V_{PIN} =2.4V	-32	-37		mA
		VDD=3.3V; V_{PIN} =2.4V	-9	-12		mA
		VDD=2.4V; V_{PIN} =2.0V	-3	-4.5		mA
I_{OH3}	Output High current (All push-pull output ports on low driving strength, except RST Pin)	VDD=5V; V_{PIN} =2.4V	-7.5	-12.5		mA
		VDD=3.3V; V_{PIN} =2.4V	-3	-4.5		mA
		VDD=2.4V; V_{PIN} =2.0V	-0.8	-1.5		mA
I_{OL1}	Output Low current (All I/O Ports)	VDD=5V; V_{PIN} =0.4V	15	21		mA
		VDD=3.3V; V_{PIN} =0.4V	13	16.5		mA
		VDD=2.4V; VPIN =0.4V	9	12		mA
I_{OL2}	Output Low current (All push-pull output ports on low driving strength, except RST Pin)	VDD=5V; V_{PIN} =0.4V	1.8	2.5		mA
		VDD=3.3V; V_{PIN} =0.4V	1.2	1.99		mA
		VDD=2.4V; VPIN =0.4V	0.8	1.4		mA
R_{RST}	Internal reset pull-down resistance	VDD=5V		130		Kohm
		VDD=3.3V		215		Kohm
		VDD=2.4V		350		Kohm
Power Consumption						
I_{OP1}	Normal mode operating current	SYSCLK = 16MHz @ IHRCO		5.3		mA
I_{OP2}	Normal mode operating current	SYSCLK = 16MHz @ IHRCO, VDD = 5V with ADC 400K sps		6.7		mA
I_{OPS1}	Slow mode operating current	SYSCLK = 16MHz /128 @ IHRCO		0.8		mA
I_{IDLE1}	Idle mode operating current	SYSCLK = 16MHz @ IHRCO		2.5		mA
I_{IDLE2}	Idle mode operating current	SYSCLK = 16MHz /128 @ IHRCO		0.75		mA
I_{IDLE3}	Idle mode operating current	SYSCLK = 32KHz @ ILRCO		33		uA
I_{SUB1}	Sub-clock mode operating current	SYSCLK = 32KHz @ ILRCO, BOD1 disabled		40		uA
I_{SUB2}	Sub-clock mode operating current	SYSCLK = 32KHz/128 @ ILRCO, BOD1 disabled		30		uA
I_{WAT}	Watch mode operating current	WDT = 32KHz @ ILRCO in PD mode		8.6		uA
I_{MON1}	Monitor Mode operating current	BOD1 enabled in PD mode		7.9		uA
I_{RTC1}	RTC Mode operating current	RTC operating in PD mode, VDD = 5.0V		8.6		uA
I_{PD1}	Power down mode current			1.3		uA
BOD0/BOD1 Characteristics						
V_{BOD0}	BOD0 detection level	T_A = -40°C to +85°C		2.35		V

V_{BOD10}	BOD1 detection level for 2.7V	T_A = -40°C to +85°C		2.7		V
V_{BOD10}	BOD1 detection level for 2.4V	T_A = -40°C to +85°C		2.4		V
V_{BOD11}	BOD1 detection level for 3.6V	T_A = -40°C to +85°C		3.6		V
V_{BOD11}	BOD1 detection level for 4.2V	T_A = -40°C to +85°C		4.2		V
I_{BOD1}	BOD1 Power Consumption	T_A = +25°C, VDD=5.0V				uA
		T_A = +25°C, VDD=3.3V				
Operating Condition						
V_{PSR}	Power-on Slope Rate	T_A = -40°C to +85°C	0.05			V/ms
V_{POR1}	Power-on Reset Valid Voltage	T_A = -40°C to +85°C			0.1	V
V_{OP1}	CPU Operating Speed 0-16MHz	T_A = -40°C to +85°C	2.7		5.5	V
V_{OP2}	CPU Operating Speed 0-12MHz	T_A = -40°C to +85°C	2.4		5.5	V

(1) Data based on characterization results, not tested in production.

(2) I/O under Quasi-Bidirectional mode, when input voltage High transfer to Low and across the threshold voltage, the internal “Weak” pull up will be turn off. I_{H2L} indicates the current near the threshold voltage. Please reference “Figure 13–1. Port 3 Quasi-Bidirectional I/O”.

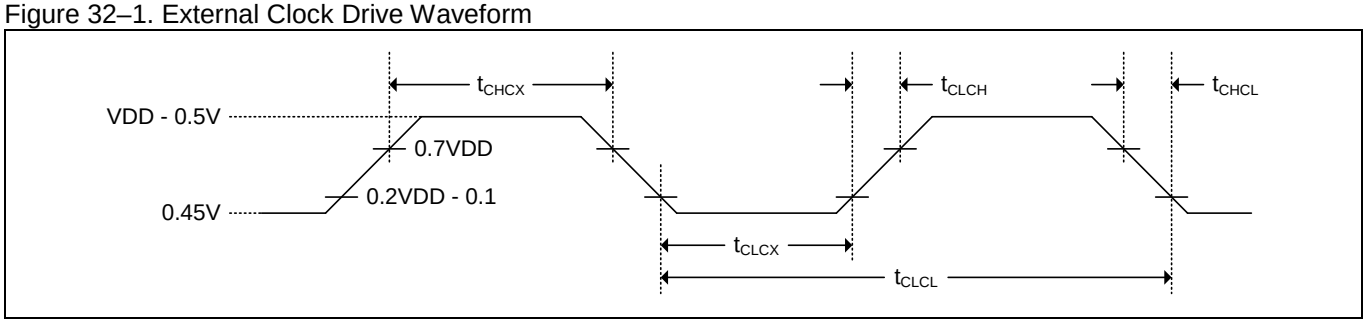
(3) All current flowing into the chip has a positive value, and current flowing out of the chip has negative value.

32.3. External Clock Characteristics

VDD = 2.4V ~ 5.5V, VSS = 0V, T_A = -40°C to +85°C, unless otherwise specified

Symbol	Parameter	Oscillator ECKI Mode		Unit
		Min.	Min.	
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
$1/t_{CLCL}$	Oscillator Frequency (VDD = 2.4V ~ 5.5V)	0	12	MHz
t_{CLCL}	Clock Period	27.7		ns
t_{CHCX}	High Time	0.4T	0.6T	t_{CLCL}
t_{CLCX}	Low Time	0.4T	0.6T	t_{CLCL}
t_{CLCH}	Rise Time		5	ns
t_{CHCL}	Fall Time		5	ns

Figure 32–1. External Clock Drive Waveform



32.4. IHRCO Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage	IHRCO = 16MHz	2.4		5.5	V
	IHRCO = 22.12MHz	2.4		5.5	V
IHRCO Frequency	T_A = +25°C, AFS = 0		16		MHz
	T_A = +25°C, AFS = 1		22.12		MHz
IHRCO = 16MHz Frequency Deviation (factory calibrated)	TA = +25°C	-2.0		+2.0	%
	TA = -40°C to +85°C	-3.8(1)		+3.8(1)	%
	TA = -40°C to +85°C, VDD = 5V	-3.8(1)		+3.8(1)	%
	TA = -40°C to +85°C, VDD = 3.3V	-3.4(1)		+3.4(1)	%
	TA = -40°C to +85°C, VDD = 2.4V	-3.6(1)		+3.6(1)	%
IHRCO = 22.12MHz Frequency Deviation (factory calibrated)	TA = +25°C	-1.2(1)		+1.5(1)	%
	TA = -40°C to +85°C	-3.0(1)		+3.0(1)	%
IHRCO frequency extra deviation when enable MCK/2, MCK/4, MCK/8 (IHRCO= 16MHz)	TA = -40°C to +85°C, VDD = 5V	+1(1)		+1.9(1)	%
	TA = -40°C to +85°C, VDD = 3.3V	-0.8(1)		+0(1)	%
	TA = -40°C to +85°C, VDD = 2.4V	-1.45(1)		+0.5(1)	%
IHRCO frequency extra deviation when enable MCK/2, MCK/4, MCK/8 (IHRCO = 22.12MHz)	TA = -40°C to +85°C, VDD = 5V	-2.4(1)		-0.5(1)	%
IHRCO Start-up Time	TA = -40°C to +85°C			32(1)	us
IHRCO Power Consumption	TA = +25°C, VDD=5.0V		TBD (1)		uA

(1) Data based on characterization results, not tested in production.

32.5. ILRCO Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage		2.4		5.5	V
ILRCO Frequency	TA = +25°C		32		KHz
ILRCO Frequency Deviation	TA = +25°C	-8(1)		+8(1)	%
	TA = -40°C to +85°C	-20(1)		+20(1)	%

(1) Data based on characterization results, not tested in production.

32.6. Flash Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage	TA = -40˚C to +85˚C	2.4		5.5	V
Flash Write (Erase/Program) Voltage	TA = -40˚C to +85˚C	2.4		5.5	V
Flash Erase/Program Cycle	TA = -40˚C to +85˚C	20,000			times
Flash Data Retention	TA = +25˚C	100			year
Flash Page Erase Time	CPUCLK = 8MHz		3.3		ms
Flash Page Program Time	CPUCLK = 8MHz		1.75		ms
Flash Latch Byte Write Time	CPUCLK = 8MHz		1.75		us
	CPUCLK = 16 MHz		880		ns
Flash Byte Read Time	CPUCLK = 8MHz		1.375		us
	CPUCLK = 16MHz		690		ns

32.7. EEPROM Characteristics

Parameter	Test Condition	Limits			Unit
		min	Typ	max	
Supply Voltage	TA = -40˚C to +85˚C	2.4		5.5	V
EEPROM Write Voltage	TA = -40˚C to +85˚C	2.4		5.5	V
EEPROM Write Cycle	TA = -40˚C to +85˚C	100,000			times
EEPROM Data Retention	TA = +25˚C	100			year
EEPROM Byte Read Time	CPUCLK = 8MHz		4.9		us
	CPUCLK = 16MHz		2.5		us
EEPROM Byte Program Time	CPUCLK = 8MHz		4.25		ms

32.8. ADC Characteristics

VDD=5.0V, T_A= -40°C ~ +85°C unless otherwise specified

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage		2.4		5.5	V
Resolution			10		bits
DC Accuracy T_A= -40°C ~ +85°C					
Integral Nonlinearity	VDD = VREF+ ≥ 3.3V, 666K sps, ADC power level = Middle low (ADPS[1:0] = 10b)	-1.3		4.2	LSB
	VDD = VREF+ = 2.4V, 666K sps, ADC power level = Middle low (ADPS[1:0] = 10b)	-1.4		1.5	
Differential Nonlinearity	VDD = VREF+ ≥ 3.3V, 666K sps, ADC power level = Middle low (ADPS[1:0] = 10b)	-1.0		1.2	
	VDD = VREF+ = 2.4V, 666K sps, ADC power level = Middle low (ADPS[1:0] = 10b)	-1.0		1.4	
Offset Error	VDD= 2.4V~5.5V	-3		+3	LSB
Conversion Rate					
SAR Conversion Clock			16		MHz
Conversion Time in SAR Clocks			24		clocks
Suggested Conversion Rate	VDD = VREF+ ≥ 2.4V, ADC power level = Middle low (ADPS[1:0] = 10b)			666	K sps
Analog Inputs					

Input Voltage Range	Single Ended (AIN+ – GND)	0		VDD	V
C_{ADC} Input Capacitance note1			4.3	6	pF
Input Sampling switch resistance note1	VDD = 5V		470 note2	990	Ω
	VDD = 4.2V		530 note2	1.18K	Ω
	VDD = 3.3V		680 note2	1.78K	Ω
	VDD = 2.7V		1K note2	3.43K	Ω
	VDD = 2.4V		1.47K note2	7.16K	Ω
Switch Channel Stable Time					
Original pin and Target pin voltage switch between VDD or GND	CH0(VDD)→CH1(GND)		0		us
	CH0(GND)→CH1(VDD)		0		
Original pin voltage = VDD switch to target pin with pulldown resistor	CH0(VDD)→CH1(51K Pulldown)		6.69		
	CH0(VDD)→CH1(10K Pulldown)		1.06		
Original pin voltage = GND switch to target pin with Pullup resistor	CH0(GND)→CH1(51K Pullup)		7.81		
	CH0(GND)→CH1(10K Pullup)		1.56		
Original pin voltage = VDD switch to resistor divider (VDD/2)	CH0(VDD)→CH1(VDD/2，51K resistor divider)		2.88		
	CH0(VDD)→CH1(VDD/2，10K resistor divider)		3.13		
Original pin voltage = GND switch to resistor divider (VDD/2)	CH0(GND)→CH1(VDD/2，51K resistor divider)		0.38		
	CH0(GND)→CH1(VDD/2，10K resistor divider)		0.44		
Power Consumption					
Power Supply Current	ADPS<1:0>=00		1.160		mA
	ADPS<1:0>=01		1.150		
	ADPS<1:0>=10		1.130		
	ADPS<1:0>=11		1.105		

(1) Data guaranteed by design, not tested in production.
(2) Input voltage < ½VDD

32.9. IVR Characteristics

VDD=5.0V±10%, VSS=0V, T_A =-40°C to +85°C, C_{LOAD}=4.7upF/0.1ohm-ESR unless otherwise specified

Parameter	Test Condition	Limits			Unit
		Min.	Typ.	Max.	
Supply Range					
Supply Voltage		2.4	5.0	5.5	V
Operation Current	Normal Power State	43		67	uA
	Low Power State		0.1		uA
DC Accuracy					
Output Supply Voltage	-40°C ~ +85°C	1.37	1.4	1.43	V
Spread over the temperature range	VDD = 3.3V±10mV				mV

32.10. Analog Comparator AC0 Characteristics

VDD=5.0V, T_A= -40°C ~ +85°C unless otherwise specified

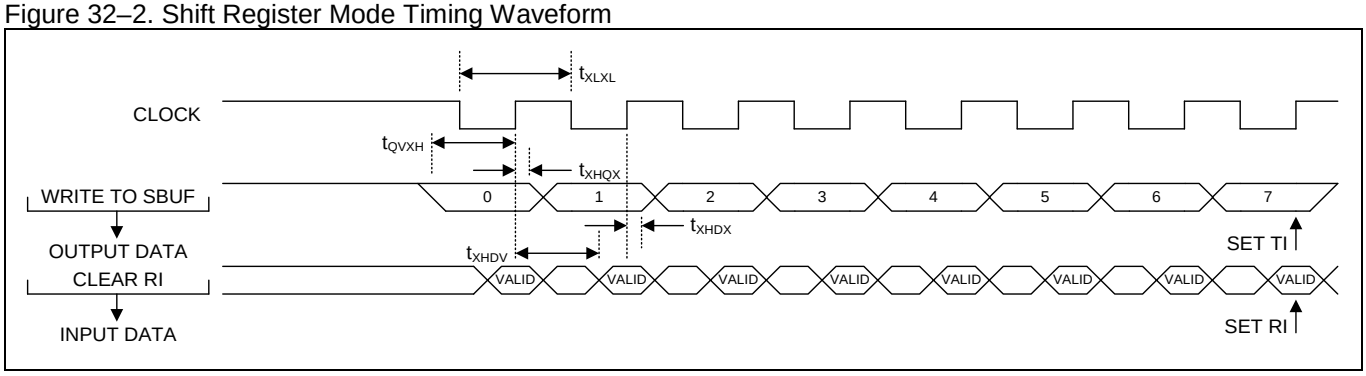
Parameter	Test Condition	Limits			Unit
		Min.	Typ.	Max.	
Supply Range					
Supply Voltage		2.4	5.0	5.5	V
Operation Current	Normal Power State				uA
	Low Power State				uA
DC Accuracy					
Input Voltage Range		50		VDD-900	mV
Input Offset Voltage-AC0 (Normal Mode)	VDD= 5.0V		4	10	mV
	VDD= 3.0V		4	10	mV
Input Offset Voltage-AC0 (Low Power Mode)	VDD= 5.0V		5	11	mV
	VDD= 3.0V		5.5	11	mV
Input Common Mode Voltage		50		VDD-900	mV
Comparator Hysteresi-AC0 (Normal Mode)	VDD= 5.0V		2	4.0	mV
	VDD= 3.0V		1.5	2.5	
Comparator Hysteresi-AC0 (Low Power Mode)	VDD= 5.0V		2	4	mV
	VDD= 3.0V		1.5	2.5	
Response Time (Normal mode)	Rising(V_{DD} = 5V, Ta = 25°C, V_{OD}=50mV, V_{CM} < 50mV)		760		ns
	Falling(V_{DD} = 5V, Ta = 25°C, V_{OD}=50mV, V_{CM} < 50mV)		450		ns
	Rising(V_{DD} ≥ 3V, V_{OD}=500mV, 0.5V ≤ V_{CM} ≤ VDD-0.9V)		780		ns
	Falling(V_{DD} ≥ 3V, V_{OD}=500mV, 0.5V ≤ V_{CM} ≤ VDD-0.9V)		500		ns
Response Time (Low Power mode)	Rising(V_{DD} = 5V, Ta = 25°C, V_{OD}=50mV, V_{CM} < 50mV)		1.2		us
	Falling(V_{DD} = 5V, Ta = 25°C, V_{OD}=50mV, V_{CM} < 50mV)		0.6		us
	Rising(V_{DD} ≥ 3V, V_{OD}=500mV, 0.5V ≤ V_{CM} ≤ VDD-0.9V)		1.27		us
	Falling(V_{DD} ≥ 3V, V_{OD}=500mV, 0.5V ≤ V_{CM} ≤ VDD-0.9V)		0.7		us

32.11. Serial Port Timing Characteristics

VDD = 5.0V±10%, VSS = 0V, T_A = -40°C to +85°C, unless otherwise specified

Symbol	Parameter	URM0X3 = 0		URM0X3 = 1		Unit
		Min.	Max	Min.	Max	
t_{XLXL}	Serial Port Clock Cycle Time	12T		4T		T_{SYSCLK}
t_{QVXH}	Output Data Setup to Clock Rising Edge	10T-20		2T-20		ns
t_{XHQX}	Output Data Hold after Clock Rising Edge	T-10		T-10		ns
t_{XHDX}	Input Data Hold after Clock Rising Edge	5		5		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		2T-10		2T-10	ns

Figure 32–2. Shift Register Mode Timing Waveform



32.12. SPI Timing Characteristics

VDD = 5.0V±10%, VSS = 0V, T_A = -40°C to +85°C, unless otherwise specified

Symbol	Parameter	Min	Max	Units
Master Mode Timing				
$1/(t_{MCKH}+t_{MCKL})$	SPI Clock Frequency @VDD = 2.4V ~ 5.5V		22.12MHz	MHz
t_{MCKH}	SPICLK High Time	1T		T_{SYSCLK}
t_{MCKL}	SPICLK Low Time	1T		T_{SYSCLK}
t_{MIS}	MISO Valid to SPICLK Sample Edge	10		ns
t_{MIH}	SPICLK Shift Edge to MISO Change	0		ns
t_{MOH}	SPICLK Shift Edge to MOSI Change		10	ns
Slave Mode Timing				
$1/(t_{CKH}+t_{CKL})$	SPI Clock Frequency @VDD = 2.4V ~ 5.5V		11.06MHz	MHz
t_{SE}	nSS Falling to First SPICLK Edge	2T		T_{SYSCLK}
t_{SD}	Last SPICLK Edge to nSS Rising	2T		T_{SYSCLK}
t_{SEZ}	nSS Falling to MISO Valid		4T	T_{SYSCLK}
t_{SDZ}	nSS Rising to MISO High-Z		4T	T_{SYSCLK}
t_{CKH}	SPICLK High Time	2T		T_{SYSCLK}
t_{CKL}	SPICLK Low Time	2T		T_{SYSCLK}
t_{SIS}	MOSI Valid to SPICLK Sample Edge	1T		T_{SYSCLK}
t_{SIH}	SPICLK Sample Edge to MOSI Change	1T		T_{SYSCLK}
t_{SOH}	SPICLK Shift Edge to MISO Change		2T	T_{SYSCLK}
t_{SLH}	Last SPICLK Edge to MISO Change (CPHA = 1 ONLY)	1T	2T	T_{SYSCLK}

Figure 32–3. SPI Master Transfer Waveform with CPHA=0

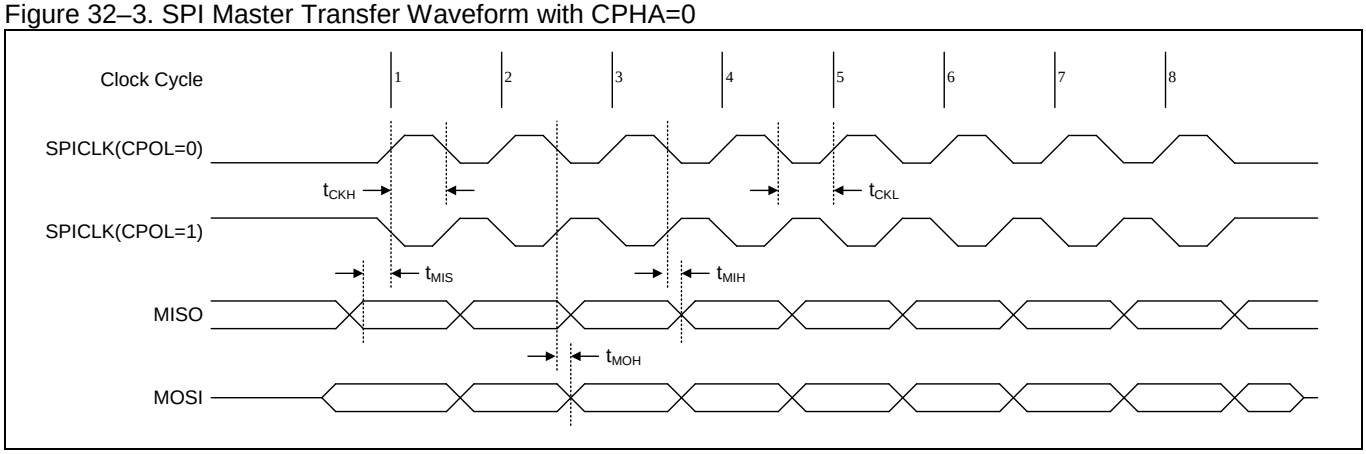


Figure 32–4. SPI Master Transfer Waveform with CPHA=1

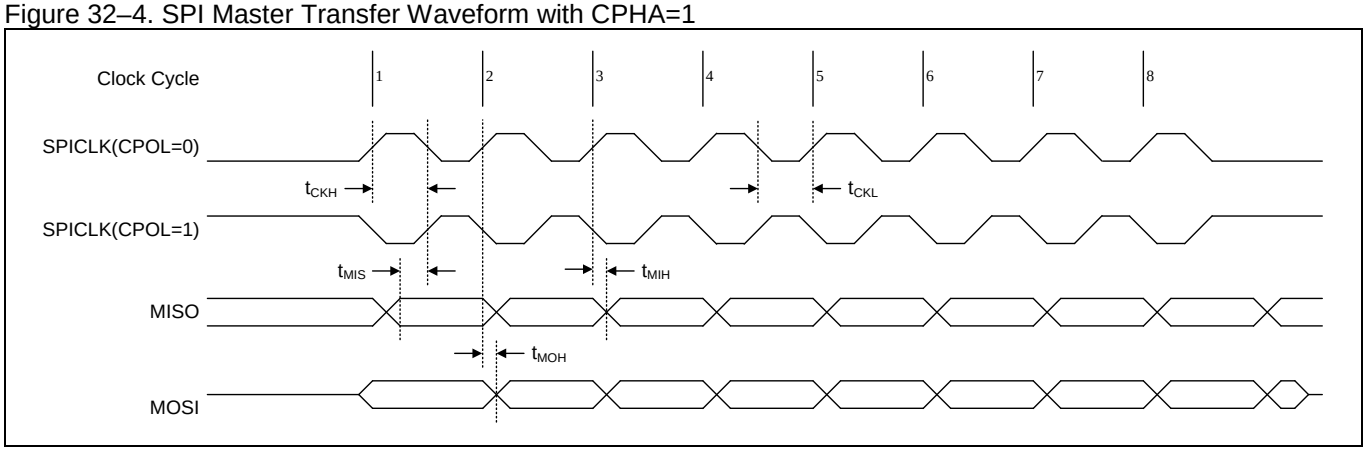


Figure 32–5. SPI Slave Transfer Waveform with CPHA=0

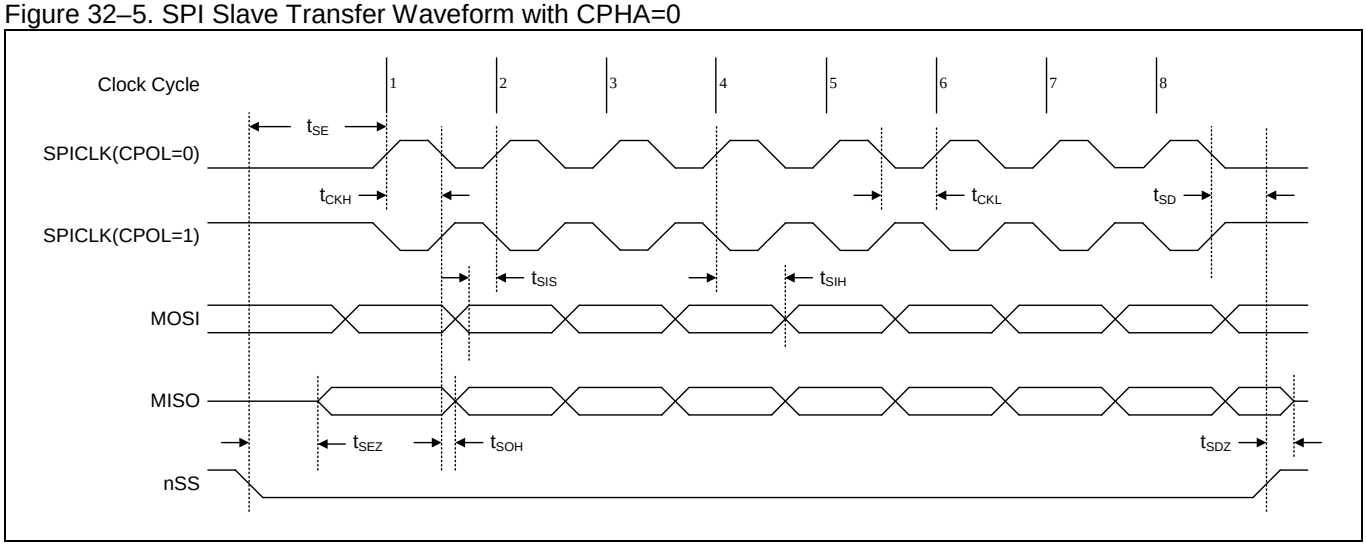
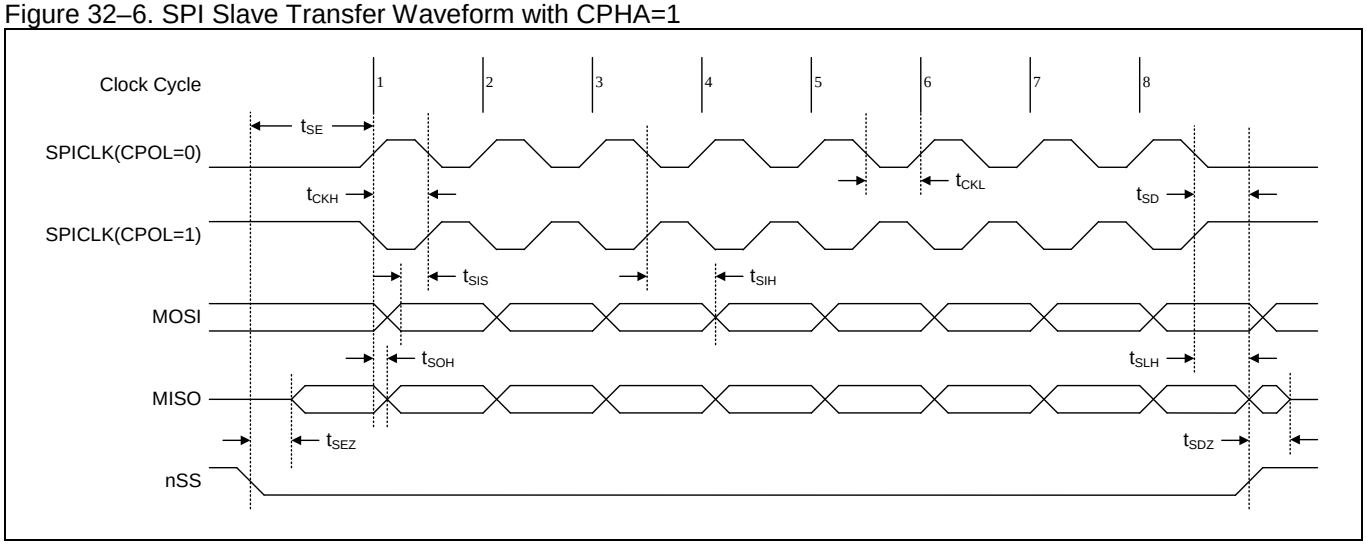


Figure 32–6. SPI Slave Transfer Waveform with CPHA=1



33. Instruction Set

Table 33–1. Instruction Set

MNEMONIC	DESCRIPTION	BYTE	EXECUTION Cycles
DATA TRASFER			
MOV A,Rn	Move register to Acc	1	1
MOV A,direct	Move direct byte o Acc	2	2
MOV A,@Ri	Move indirect RAM to Acc	1	2
MOV A,#data	Move immediate data to Acc	2	2
MOV Rn,A	Move Acc to register	1	2
MOV Rn,direct	Move direct byte to register	2	4
MOV Rn,#data	Move immediate data to register	2	2
MOV direct,A	Move Acc to direct byte	2	3
MOV direct,Rn	Move register to direct byte	2	3
MOV direct,direct	Move direct byte to direct byte	3	4
MOV direct,@Ri	Move indirect RAM to direct byte	2	4
MOV direct,#data	Move immediate data to direct byte	3	3
MOV @Ri,A	Move Acc to indirect RAM	1	3
MOV @Ri,direct	Move direct byte to indirect RAM	2	3
MOV @Ri,#data	Move immediate data to indirect RAM	2	3
MOV DPTR,#data16	Load DPTR with a 16-bit constant	3	3
MOVC A,@A+DPTR	Move code byte relative to DPTR to Acc	1	4
MOVC A,@A+PC	Move code byte relative to PC to Acc	1	4
MOVX A,@Ri	Move on-chip auxiliary RAM(8-bit address) to Acc	1	3
MOVX A,@DPTR	Move on-chip auxiliary RAM(16-bit address) to Acc	1	3
MOVX @Ri,A	Move Acc to on-chip auxiliary RAM(8-bit address)	1	3
MOVX @DPTR,A	Move Acc to on-chip auxiliary RAM(16-bit address)	1	3
PUSH direct	Push direct byte onto Stack	2	4
POP direct	Pop direct byte from Stack	2	3
XCH A,Rn	Exchange register with Acc	1	3
XCH A,direct	Exchange direct byte with Acc	2	4
XCH A,@Ri	Exchange indirect RAM with Acc	1	4
XCHD A,@Ri	Exchange low-order digit indirect RAM with Acc	1	4
ARITHEMATIC OPERATIONS			
ADD A,Rn	Add register to Acc	1	2
ADD A,direct	Add direct byte to Acc	2	3
ADD A,@Ri	Add indirect RAM to Acc	1	3
ADD A,#data	Add immediate data to Acc	2	2
ADDC A,Rn	Add register to Acc with Carry	1	2
ADDC A,direct	Add direct byte to Acc with Carry	2	3
ADDC A,@Ri	Add indirect RAM to Acc with Carry	1	3
ADDC A,#data	Add immediate data to Acc with Carry	2	2
SUBB A,Rn	Subtract register from Acc with borrow	1	2
SUBB A,direct	Subtract direct byte from Acc with borrow	2	3
SUBB A,@Ri	Subtract indirect RAM from Acc with borrow	1	3
SUBB A,#data	Subtract immediate data from Acc with borrow	2	2
INC A	Increment Acc	1	2
INC Rn	Increment register	1	3
INC direct	Increment direct byte	2	4
INC @Ri	Increment indirect RAM	1	4

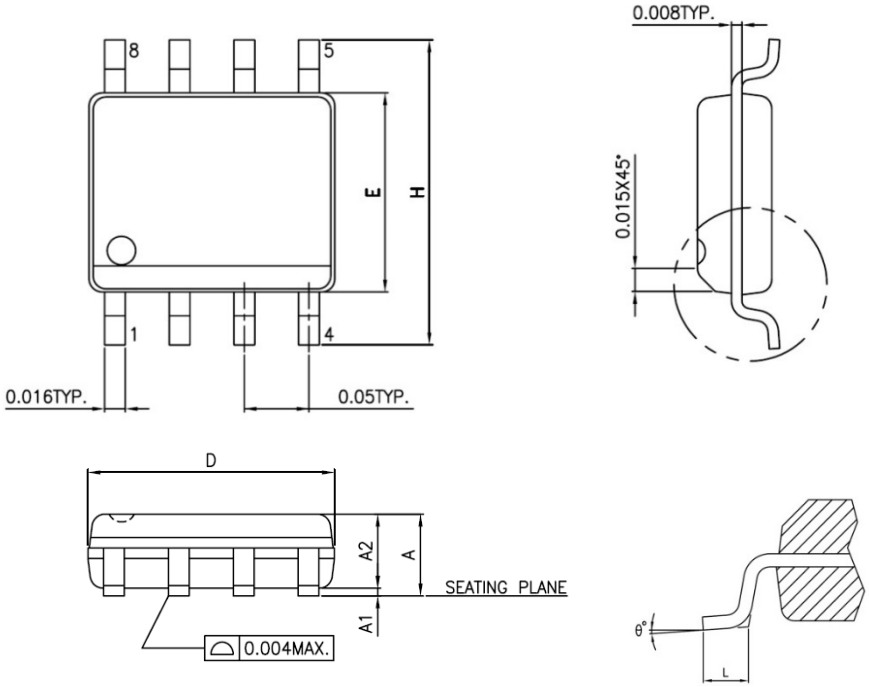
MNEMONIC	DESCRIPTION	BYTE	EXECUTION Cycles
DEC A	Decrement Acc	1	2
DEC Rn	Decrement register	1	3
DEC direct	Decrement direct byte	2	4
DEC @Ri	Decrement indirect RAM	1	4
INC DPTR	Increment DPTR	1	1
MUL AB	Multiply A and B	1	4
DIV AB	Divide A by B	1	5
DA A	Decimal Adjust Acc	1	4
LOGIC OPERATION			
ANL A,Rn	AND register to Acc	1	2
ANL A,direct	AND direct byte to Acc	2	3
ANL A,@Ri	AND indirect RAM to Acc	1	3
ANL A,#data	AND immediate data to Acc	2	2
ANL direct,A	AND Acc to direct byte	2	4
ANL direct,#data	AND immediate data to direct byte	3	4
ORL A,Rn	OR register to Acc	1	2
ORL A,direct	OR direct byte to Acc	2	3
ORL A,@Ri	OR indirect RAM to Acc	1	3
ORL A,#data	OR immediate data to Acc	2	2
ORL direct,A	OR Acc to direct byte	2	4
ORL direct,#data	OR immediate data to direct byte	3	4
XRL A,Rn	Exclusive-OR register to Acc	1	2
XRL A,direct	Exclusive-OR direct byte to Acc	2	3
XRL A,@Ri	Exclusive-OR indirect RAM to Acc	1	3
XRL A,#data	Exclusive-OR immediate data to Acc	2	2
XRL direct,A	Exclusive-OR Acc to direct byte	2	4
XRL direct,#data	Exclusive-OR immediate data to direct byte	3	4
CLR A	Clear Acc	1	1
CPL A	Complement Acc	1	2
RL A	Rotate Acc Left	1	1
RLC A	Rotate Acc Left through the Carry	1	1
RR A	Rotate Acc Right	1	1
RRC A	Rotate Acc Right through the Carry	1	1
SWAP A	Swap nibbles within the Acc	1	1
BOOLEAN VARIABLE MANIPULATION			
CLR C	Clear Carry	1	1
CLR bit	Clear direct bit	2	4
SETB C	Set Carry	1	1
SETB bit	Set direct bit	2	4
CPL C	Complement Carry	1	1
CPL bit	Complement direct bit	2	4
ANL C,bit	AND direct bit to Carry	2	3
ANL C,/bit	AND complement of direct bit to Carry	2	3
ORL C,bit	OR direct bit to Carry	2	3
ORL C,/bit	OR complement of direct bit to Carry	2	3
MOV C,bit	Move direct bit to Carry	2	3
MOV bit,C	Move Carry to direct bit	2	4

MNEMONIC	DESCRIPTION	BYTE	EXECUTION Cycles
BOOLEAN VARIABLE MANIPULATION			
JC rel	Jump if Carry is set	2	3
JNC rel	Jump if Carry not set	2	3
JB bit,rel	Jump if direct bit is set	3	4
JNB bit,rel	Jump if direct bit not set	3	4
JBC bit,rel	Jump if direct bit is set and then clear bit	3	5
PROAGRAM BRACHING			
ACALL addr11	Absolute subroutine call	2	6
LCALL addr16	Long subroutine call	3	6
RET	Return from subroutine	1	4
RETI	Return from interrupt subroutine	1	4
AJMP addr11	Absolute jump	2	3
LJMP addr16	Long jump	3	4
SJMP rel	Short jump	2	3
JMP @A+DPTR	Jump indirect relative to DPTR	1	3
JZ rel	Jump if Acc is zero	2	3
JNZ rel	Jump if Acc not zero	2	3
CJNE A,direct,rel	Compare direct byte to Acc and jump if not equal	3	5
CJNE A,#data,rel	Compare immediate data to Acc and jump if not equal	3	4
CJNE Rn,#data,rel	Compare immediate data to register and jump if not equal	3	4
CJNE @Ri,#data,rel	Compare immediate data to indirect RAM and jump if not equal	3	5
DJNZ Rn,rel	Decrement register and jump if not equal	2	4
DJNZ direct,rel	Decrement direct byte and jump if not equal	3	5
NOP	No Operation	1	1

34. Package Dimension

34.1. SOP-8 (150mil) Package Dimension

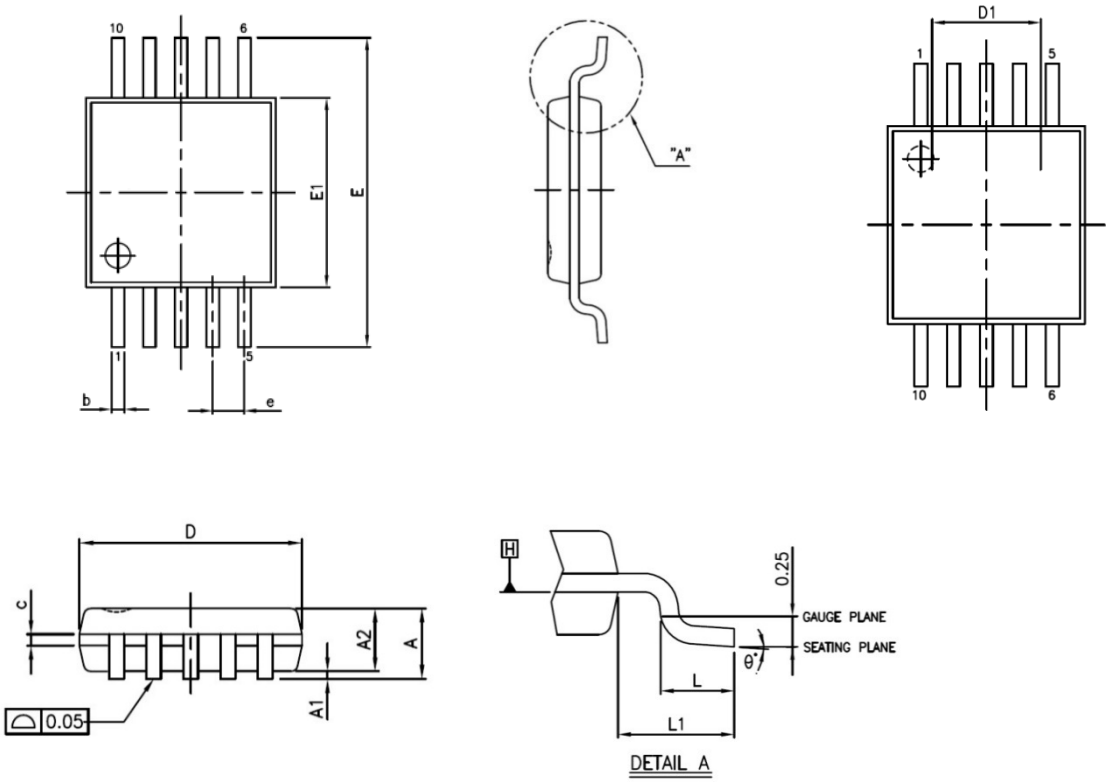
Figure 34-1. SOP-8 (150 mil) Package Dimension



Unit	mm		inch	
Symbols	Min.	Max.	Min.	Max.
A	1.346	1.752	0.052	0.068
A1	0.101	0.254	0.003	0.010
A2	1.346	1.498	0.052	0.058
D	4.800	4.978	0.188	0.195
E	3.810	3.987	0.150	0.156
H	5.791	6.197	0.227	0.243
L	0.406	1.270	0.015	0.050
θ	0°	8°	0	8

34.2. MSOP-10 (150mil) Package Dimension

Figure 34-2. MSOP-10 (3.0x3.0x0.85)



Unit	mm			inch		
Symbols	Min.	Nom.	Max.	Min.	Nom.	Max.
A	----	----	1.10	----	----	0.043
A1	0.00	----	0.15	0.000	----	0.005
A2	0.75	0.85	0.95	0.029	0.033	0.037
b	0.17	----	0.27	0.006	----	0.010
c	0.08	----	0.23	0.003	----	0.009
D	3.00 BSC			0.118 BSC		
E	4.90 BSC			0.192 BSC		
E1	3.00 BSC			0.118 BSC		
e	0.50 BSC			0.019 BSC		
L	0.40	0.60	0.80	0.015	0.023	0.031
L1	0.95 REF			0.037 REF		
θ	0°	----	8°	0°	----	8°

35. Revision History

Table 35–1. Revision History

Rev	Descriptions	Date
0.18	1. Initial version	2021/04/20
0.19	1. Fixed BOF0 and BOF1 voltage in PCON1 2. Fixed P33M to P33AM in Table 13-2 3. Fixed Bit 7~0 to Bit 0 in PDRVC1 4. Fixed Timer 0 to Timer 1 of Figure 15-9 5. Removed PCA clock description in section 16.2 6. Modified CCON description, CCF5 to CCF3. 7. Removed the notice of Table 16-1 8. Modified PCA channel related description from 6 channels to 4 channels 9. Correct the PIN assignment of the RXD0 and TXD0 in Section 17.10 10. Removed QPIEN in section 18.5 11. Fixed Flash page size from 512 to 64 Bytes 12. Fixed the description of the PBSY = 0, from ISP/IAP engine to EEPROM engine in section 27.3 13. Add description to explain why needs to check PBSY before Flash access in section 27.3. 14. Replace the 8051 OCD ICE adapter with MLink in section 31.4 and 31.5. 15. Add 3 Flash access mode in IFMT 16. Add ISP/IAP Byte program flow in section 27.3.3, and modify ISP/IAP page program flow in section 27.3.2	2021/05/30
1.00	1. Add 10 bit ADC related information 2. Add C0PLK description 3. Modified Timer1 mode 3 description 4. Modified KBI description 5. Add the method to improve ADC accuracy. 6. Removed Timer 2 Split mode 6 7. Removed Crystal Mode in section 32.3 8. Modified min supply voltage of IHRCO and ILRCO from 1.8V to 2.4V in section 32.4 and 32.5 9. Modified SPI clock in section 32.12	2022/05/06
1.01	1. Fixed typo of the S0 baud rate formula in section 17.8.4.2 2. Fixed typo of slope rate in section 32.2 3. Fixed typo of RTC clock in chapter 10 4. Correct page information of RTCCR and RTCTM in section 10.1 5. Fixed T2RLC function description in section 15.3.2. Force reload mode only not available in duty capture mode. 6. Fixed WDTFS function description, Bit 0: Select WDT bit-7. 7. Modified IHRCO tolerance from 4.5% to 3%	2022/05/29
1.02	1. Fixed typo of IVR(2.4V) to IVR (1.4V) in section 26.3	2022/06/15
1.03	1. Modify IHRCO spec. 2. Added ADC characteristics. 3. Fixed typo of pin number in Table 4-1 and Table 4-2	2023/05/05
1.04	1. Modify IHRCO spec description 2. Modify maximum CPU operation frequency up to 22.12MHz in the feature and Section 32.2	2023/05/10
1.05	1. Add ADC alternate PIN for MG82F6B104 in Figure 4-4 2. Fixed IHRCO tolerance in the feature section 3. Removed maximum CPU operation frequency up to 22.12MHz in the feature and section 32.2	2023/07/18

36. Disclaimers

Herein, Megawin stands for "***Megawin Technology Co., Ltd.***"

Life Support — This product is not designed for use in medical, life-saving or life-sustaining applications, or systems where malfunction of this product can reasonably be expected to result in personal injury. Customers using or selling this product for use in such applications do so at their own risk and agree to fully indemnify Megawin for any damages resulting from such improper use or sale.

Right to Make Changes — Megawin reserves the right to make changes in the products - including circuits, standard cells, and/or software - described or contained herein in order to improve design and/or performance. When the product is in mass production, relevant changes will be communicated via an Engineering Change Notification (ECN).