

# **MG84FG516**

## **Data Sheet**

**Version: A1.5**



## Features

- 1-T 80C51 Central Processing Unit
- MG84FG516 with 64K Bytes flash ROM
  - ISP memory zone could be optioned as 1KB/1.5KB~4KB
  - Flexible IAP size.
  - Code protection for flash memory access
  - Flash erase/program cycle: 10,000 times
  - Flash data retention: 100 years at 25°C
  - **MG84FG516 Flash space mapping**
    - ◆ **AP Flash (60KB, 0000h~EFFFh)**
    - ◆ **IAP Flash (1.5KB, F000h~F5FFh)**
    - ◆ **USB DFU boot-Code (2.5KB, F600h~FFFFh) (USB boot-code space)**
- Data RAM
  - On-chip 256 bytes scratch-pad RAM
  - 4096 bytes expanded RAM (XRAM)
  - Up to 5120 bytes expanded RAM (XRAM) by sharing USB buffer
- Dual data pointer
- Variable length MOVX for slow SRAM/Peripherals
- Interrupt controller
  - 17 sources, four-level-priority interrupt capability
  - Four external interrupt inputs, nINT0, nINT1, nINT2 and nINT3
  - All external interrupts support High/Low level or Rising/Falling edge trigger
- Four 16-bit timer/counters, Timer 0, Timer 1, Timer 2 and Timer 3
  - T0CKO on P34, T1CKO on P35, T2CKO on P10 and T3CKO on P01
  - X12 mode enabled for T0/T1/T2/T3
- Programmable 16-bit counter/timer Array (PCA) with 6 compare/capture modules
  - Capture mode
  - 16-bit software timer mode
  - High speed output mode
  - 8/10/12/16-bit PWM (Pulse Width Modulator) mode with phase shift function
- Keypad Interrupt (P0/P2/P5/P6)
- 12-Bit ADC
  - Programmable throughput up to 250 ksp/s
  - Up to 8 channel single-ended inputs or 4 channel differential inputs
- Enhanced UART (S0)
  - Framing Error Detection
  - Automatic Address Recognition
  - Speed improvement mechanism (X2/X4 mode)
- Secondary UART (S1)
  - Dedicated Baud Rate Generator
  - S1 shares baud rate generator to S0.
- Master/Slave SPI serial interface
- Master/Slave two wire serial interface (TWSI)
- USB Device Controller
  - USB Full speed (12Mbps) operation and USB specification 2.0 compliant
  - Intel 8X931 like USB control flow

- Built-in USB transceiver and 3.3V regulator
- Six pairs flexible endpoints for control/interrupt/isochronous/bulk IN/OUT transaction
- 1K bytes FIFO for USB endpoint buffer
- Software-controlled USB connection/disconnection mechanism
- Programmable Watchdog Timer
  - One time enabled by CPU or power-on
  - Sourced from on-chip low frequency oscillator
  - Interrupt CPU or Reset CPU on WDT overflow
  - Watch mode support
- On-Chip-Debug interface (OCD)
- Maximum 55 GPIOs in LQFP64 package
  - P3 can be configured to quasi-bidirectional, push-pull output, open-drain output and input only
  - P0, P1, P2, P4, P5 and P6 can be configured to open-drain output or push-pull output
  - P6.0 and P6.1 shared with XTAL2 and XTAL1
- Multiple power control modes: idle mode, power-down mode, slow mode, sub-clock mode, watch mode and monitor mode.
  - All interrupts can wake up IDLE mode
  - 12 sources to wake up Power-Down mode
  - Slow mode and sub-clock mode support low speed MCU operation
  - Watch mode supports WDT to resume CPU in power down
  - Monitor mode supports BOD1 to resume CPU in power down
- Two Brown-Out Detectors
  - BOD0: detect 2.2V
  - BOD1: selected detection level on 4.2V/3.7V/2.4V/2.0V
  - Interrupt CPU or reset CPU
  - Wake up CPU in Power-Down mode
- Operating voltage range: 2.0V – 5.5V
  - Minimum 2.2V requirement in flash write operation (ISP/IAP/ICP)
- Operating frequency range: 36MHz(max)
  - External crystal mode, 2 – 12MHz @ 2.0V – 5.5V and 2 – 25MHz @ 2.7V – 5.5V.
  - CPU up to 12MHz @ 2.0V – 5.5V, up to 25MHz @ 2.4V – 5.5V and up to 36MHz @ 2.7V – 5.5V
- Clock Sources
  - Internal 12MHz oscillator (IHRCO): factory calibrated to  $\pm 1\%$ , typical
  - External crystal mode
  - Internal Low frequency RC Oscillator (ILRCO) support: about 32KHz
  - External clock input (ECKI) on XTAL2/P6.0, up to 36MHz.
  - Internal Oscillator output on XTAL2/P6.0
  - On-chip Clock Multiplier (CKM) to provide high speed clock source
- Operating Temperature:
  - Industrial (-40°C to +125°C)\*
- Package Types:
  - LQFP64 (7mm x 7mm): MG84FG516AD64
  - LQFP48 (7mm x 7mm): MG84FG516AD48

\*: Tested by sampling.

# Content

Features.....	3
<b>Content .....</b>	<b>5</b>
1. General Description.....	11
2. Ordering Information.....	12
3. Block Diagram.....	13
4. Special Function Register.....	14
4.1. SFR Map (Page 0~F).....	14
4.2. SFR Bit Assignment (Page 0~F).....	16
4.3. Auxiliary SFR Map (Page P).....	19
4.4. Auxiliary SFR Bit Assignment (Page P).....	20
5. Pin Configurations.....	21
5.1. Package Instruction.....	21
5.2. Pin Description.....	23
5.3. Alternate Function Redirection.....	26
6. 8051 CPU Function Description.....	28
6.1. CPU Register.....	28
6.2. CPU Timing.....	29
6.3. CPU Addressing Mode.....	29
7. Memory Organization.....	31
7.1. On-Chip Program Flash.....	31
7.2. On-Chip Data RAM.....	32
7.3. On-chip expanded RAM (XRAM).....	36
7.4. External Data Memory access.....	37
7.4.1. Multiplexed Mode for 8-bit MOVX.....	38
7.4.2. Multiplexed Mode for 16-bit MOVX.....	39
7.4.3. No Address Phase Mode for MOVX.....	40
7.5. Declaration Identifiers in a C51-Compiler.....	41
8. Dual Data Pointer Register (DPTR).....	42
9. System Clock.....	43
9.1. Clock Structure.....	43
9.2. Clock Register.....	44
9.3. Clock Sample Code.....	47
10. Watch Dog Timer (WDT).....	50
10.1. WDT Structure.....	50
10.2. WDT During Idle and Power Down.....	50
10.3. WDT Register.....	51
10.4. WDT Hardware Option.....	52
10.5. WDT Sample Code.....	53
11. System Reset.....	55
11.1. Reset Source.....	55
11.2. Power-On Reset.....	55
11.3. External Reset.....	56
11.4. Software Reset.....	56
11.5. Illegal Address Reset.....	56
11.6. Brown-Out Reset.....	56
11.7. WDT Reset.....	57

11.8.	Reset Sample Code .....	58
12.	Power Management .....	59
12.1.	Brown-Out Detector.....	59
12.2.	Power Saving Mode .....	60
12.2.1.	Slow Mode .....	60
12.2.2.	Sub-Clock Mode.....	60
12.2.3.	Watch Mode .....	60
12.2.4.	Monitor Mode .....	60
12.2.5.	Idle Mode .....	60
12.2.6.	Power-down Mode .....	60
12.2.7.	Interrupt Recovery from Power-down .....	62
12.2.8.	Reset Recovery from Power-down .....	62
12.2.9.	KBI wakeup Recovery from Power-down.....	62
12.2.10.	USB wakeup Recovery from Power-down .....	62
12.3.	Power Control Register .....	63
12.4.	Power Control Sample Code.....	65
13.	Configurable I/O Ports .....	69
13.1.	IO Structure .....	69
13.1.1.	Port 3 Quasi-Bidirectional IO Structure .....	69
13.1.2.	Port 3 Push-Pull Output Structure .....	70
13.1.3.	Port 3 Input-Only (High Impedance Input) Structure .....	70
13.1.4.	Port 3 Open-Drain Output Structure .....	70
13.1.5.	General Open-Drain Output Structure .....	71
13.1.6.	General Push-Pull Output Structure .....	71
13.1.7.	General Port Input Configured.....	72
13.2.	I/O Port Register .....	72
13.2.1.	Port 0 Register .....	72
13.2.2.	Port 1 Register .....	73
13.2.3.	Port 2 Register .....	73
13.2.4.	Port 3 Register .....	74
13.2.5.	Port 4 Register .....	74
13.2.6.	Port 5 Register .....	75
13.2.7.	Port 6 Register .....	75
13.2.8.	Pull-Up Control Register.....	75
13.3.	GPIO Sample Code.....	77
14.	Interrupt.....	78
14.1.	Interrupt Structure .....	78
14.2.	Interrupt Source .....	80
14.3.	Interrupt Enable.....	81
14.4.	Interrupt Priority.....	81
14.5.	Interrupt Process.....	82
14.6.	nINT2/nINT3 Input Source Selection.....	83
14.7.	Interrupt Register .....	84
14.8.	Interrupt Sample Code .....	90
15.	Timers/Counters .....	91
15.1.	Timer 0 and Timer 1 .....	91
15.1.1.	Timer 0/1 Mode 0 .....	91
15.1.2.	Timer 0/1 Mode 1 .....	91
15.1.3.	Timer 0/1 Mode 2 .....	92
15.1.4.	Timer 0/1 Mode 3 .....	92
15.1.5.	Timer 0/1 Programmable Clock-Out .....	92
15.1.6.	Timer 0/1 Register.....	93
15.2.	Timer 2 .....	95

15.2.1. Capture Mode (CP) .....	95
15.2.2. Auto-Reload Mode (AR) .....	96
15.2.3. Baud-Rate Generator Mode (BRG) .....	97
15.2.4. Timer 2 Programmable Clock Output .....	98
15.2.5. Timer 2 Register.....	99
15.3. Timer 3.....	101
15.3.1. 16-bit Timer with Auto-Reload .....	101
15.3.2. Two 8-bit Timers with Auto-Reload.....	102
15.3.3. Timer 3 Programmable Clock Output .....	102
15.3.4. Timer 3 Register.....	104
15.4. Timer0/1 Sample Code .....	106
16. Serial Port 0 (UART0).....	109
16.1. Serial Port 0 Mode 0.....	110
16.2. Serial Port 0 Mode 1.....	111
16.3. Serial Port 0 Mode 2 and Mode 3.....	112
16.4. Frame Error Detection.....	112
16.5. Multiprocessor Communications .....	113
16.6. Automatic Address Recognition .....	113
16.7. Baud Rate Setting .....	115
16.7.1. Baud Rate in Mode 0.....	115
16.7.2. Baud Rate in Mode 2.....	115
16.7.3. Baud Rate in Mode 1 & 3 .....	115
16.8. Serial Port 0 Register .....	121
17. Serial Port 1 (UART1).....	124
17.1. Serial Port 1 Baud Rate Generator (S1BRG).....	124
17.2. Serial Port 1 Baud Rate Setting .....	124
17.2.1. Baud Rate in Mode 0.....	124
17.2.2. Baud Rate in Mode 2.....	124
17.2.3. Baud Rate in Mode 1 & 3 .....	125
17.3. Pure Timer Mode for S1BRG .....	127
17.4. S1BRT Programmable Clock Output.....	127
17.5. S1 Baud Rate Timer for S0 .....	128
17.6. Serial Port 1 Register .....	129
17.7. Serial Port Sample Code.....	131
18. Programmable Counter Array (PCA) .....	132
18.1. PCA Overview.....	132
18.2. PCA Timer/Counter.....	133
18.3. Compare/Capture Modules .....	135
18.4. Operation Modes of the PCA .....	136
18.4.1. Capture Mode .....	136
18.4.2. 16-bit Software Timer Mode .....	137
18.4.3. High Speed Output Mode .....	137
18.4.4. PWM Mode .....	138
18.4.5. Enhance PWM Mode .....	139
18.5. PCA Sample Code .....	141
19. Serial Peripheral Interface (SPI) .....	142
19.1. Typical SPI Configurations .....	143
19.1.1. Single Master & Single Slave .....	143
19.1.2. Dual Device, where either can be a Master or a Slave .....	143
19.1.3. Single Master & Multiple Slaves .....	143
19.2. Configuring the SPI .....	144
19.2.1. Additional Considerations for a Slave .....	144
19.2.2. Additional Considerations for a Master .....	144

19.2.3. Mode Change on nSS-pin .....	145
19.2.4. Transmit Holding Register Full Flag.....	145
19.2.5. Write Collision .....	145
19.2.6. SPI Clock Rate Select .....	145
19.3. Data Mode.....	146
19.4. SPI Register .....	148
19.5. SPI Sample Code.....	150
20. Two Wire Serial Interface (TWSI/TWI).....	151
20.1. Operating Modes.....	152
20.1.1. Master Transmitter Mode .....	152
20.1.2. Master Receiver Mode .....	152
20.1.3. Slave Transmitter Mode .....	153
20.1.4. Slave Receiver Mode .....	153
20.2. Miscellaneous States .....	154
20.3. Using the TWSI .....	154
20.4. TWSI Register.....	160
20.5. TWSI Sample Code.....	163
21. Keypad Interrupt (KBI).....	166
21.1. Keypad Register.....	166
21.2. Keypad Interrupt Sample Code .....	168
22. 12-Bit ADC .....	169
22.1. ADC Structure .....	169
22.2. ADC Operation.....	169
22.2.1. ADC Input Channels.....	169
22.2.2. Starting a Conversion.....	170
22.2.3. ADC Conversion Time.....	170
22.2.4. I/O Pins Used with ADC Function.....	170
22.2.5. Idle and Power-Down Mode .....	170
22.3. ADC Register .....	171
22.4. ADC Sample Code.....	174
23. Universal Serial Bus (USB).....	175
23.1. Features .....	175
23.2. Block Diagram.....	176
23.3. FIFO Management .....	176
23.4. Access USB 1K FIFO.....	177
23.5. USB Initial .....	178
23.6. Access USB SFR .....	179
23.6.1. Through MOVX Instruction .....	179
23.6.2. Through USBADR/USB DAT .....	179
23.7. USB Interrupt .....	181
23.8. USB Special Function Registers .....	182
23.8.1. USB Function SFR Mapping.....	182
23.8.2. USB Function SFR Bit Assignment.....	183
24. ISP and IAP .....	192
24.1. Flash Memory Configuration .....	192
24.2. ISP Operation.....	193
24.2.1. ISP/IAP Register .....	193
24.2.2. Description for ISP Operation.....	195
24.3. Sample Code for ISP.....	196
24.4. IAP Operation.....	197
24.4.1. IAP-memory Boundary/Range.....	197
24.4.2. Update data in IAP-memory .....	197



25. Page P SFR Access .....	198
25.1. Page-P Sample Code.....	201
26. Auxiliary SFRs .....	203
27. Hardware Option .....	206
28. Application Notes .....	208
28.1. Power Supply Circuit .....	208
28.2. Reset Circuit.....	208
28.3. XTAL Oscillating Circuit.....	209
28.4. ICP and OCD Interface Circuit .....	210
28.5. In-Chip-Programming Function .....	211
28.6. On-Chip-Debug Function .....	212
29. Electrical Characteristics .....	213
29.1. Absolute Maximum Rating .....	213
29.2. DC Characteristics .....	214
29.3. External Clock Characteristics .....	216
29.4. IHRCO Characteristics .....	216
29.5. ILRCO Characteristics.....	216
29.6. CKM Characteristics.....	217
29.7. Flash Characteristics.....	217
29.8. ADC Characteristics .....	218
29.9. Serial Port Timing Characteristics .....	219
29.10. SPI Timing Characteristics .....	220
29.11. External Memory Cycle Timing Characteristics .....	222
29.12. USB Transceiver Electrical Characteristics .....	224
30. Instruction Set .....	225
31. Package Dimension .....	228
31.1. LQFP-64 (7mm X 7mm) .....	228
31.2. LQFP-48 (7mm X 7mm) .....	229
32. Revision History .....	230



# 1. General Description

The **MG84FG516** 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 **MG84FG516** can operate at a much lower speed and thereby greatly reduce the power consumption.

The **MG84FG516** has **64K** 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 (via USB DFU). 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. There needs no external high voltage for programming due to its built-in charge-pumping circuitry.

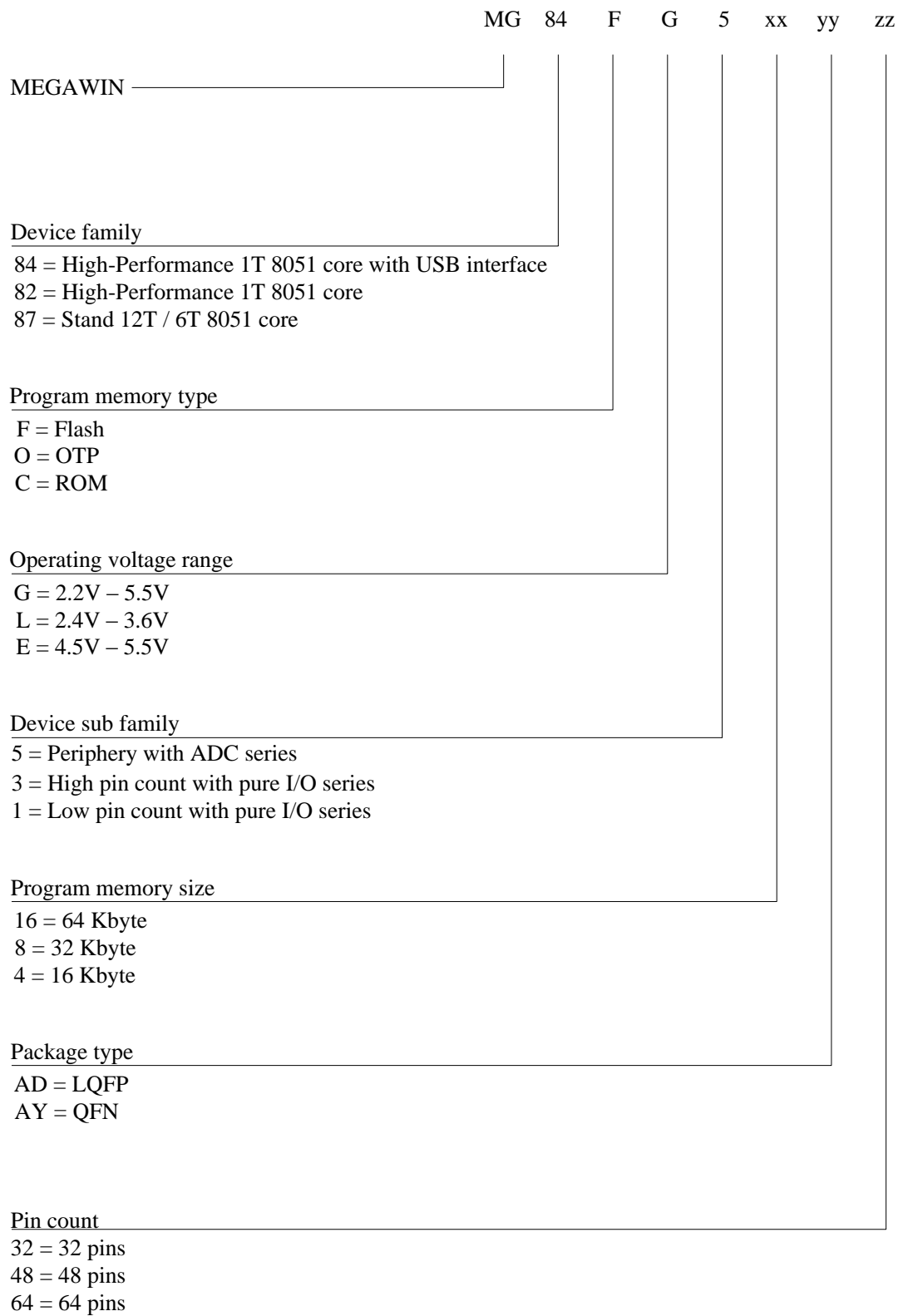
The **MG84FG516** retains all features of the standard 80C52 with 256 bytes of scratch-pad RAM, four 8-bit I/O ports, two external interrupts, a multi-source 4-level interrupt controller, a serial port (UART0) and three timer/counters. In addition, the **MG84FG516** has a full speed USB device function, three extra I/O ports (P4[6:0], P5, P6), one XRAM of 4096 bytes, two extra external interrupts with High/low trigger option, 12-bit ADC, a 6-channel PCA, SPI, TWSI, secondary serial port (UART1), keypad interrupt, Watchdog Timer, 4th 16-bit timer, two Brown-out Detectors, an on-chip crystal oscillator(shared with P6.0 and P6.1), an internal high precision oscillator to fit USB full speed application, an internal low speed RC oscillator (ILRCO) and an enhanced serial function in UART0 that facilitates multiprocessor communication and a speed improvement mechanism (X2/X4 mode).

The **MG84FG516** has multiple operating modes to reduce the power consumption: idle mode, power down mode, slow mode, sub-clock 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. In watch mode, it keeps WDT running in power-down or idle mode and resumes CPU 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 **MG84FG516** 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.

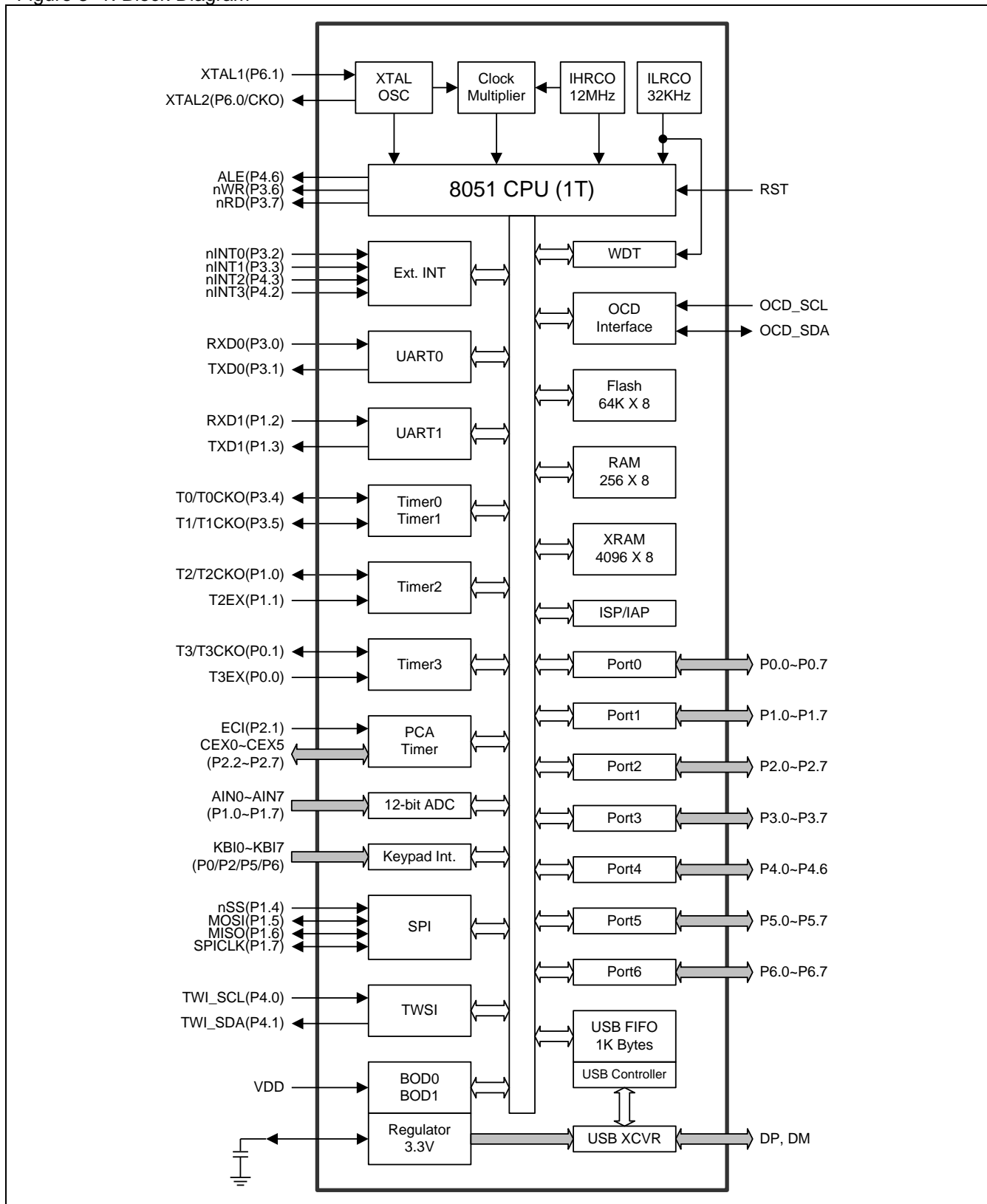
## 2. Ordering Information

Figure 2–1. ordering information



### 3. Block Diagram

Figure 3–1. Block Diagram



## 4. Special Function Register

### 4.1. SFR Map (Page 0~F)

Table 4-1. SFR Map (Page 0~F)

	Page	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8	0*	P5*	CH	CCAP0H	CCAP1H	CCAP2H	CCAP3H	CCAP4H	CCAP5H
	1*	P6*							
F0	0	B	--	PCAPWM0	PCAPWM1	PCAPWM2	PCAPWM3	PCAPWM4	PCAPWM5
	1								
E8	0	P4	CL	CCAP0L	CCAP1L	CCAP2L	CCAP3L	CCAP4L	CCAP5L
	1								
E0	0	ACC	WDTCR	IFD	IFADRH	IFADRL	IFMT	SCMD	ISPCR
	1								
D8	0	CCON	CMOD	CCAPM0	CCAPM1	CCAPM2	CCAPM3	CCAPM4	CCAPM5
	1								
D0	0	PSW	SIADR	SIDAT	SISTA	SICON	KBPATN	KBCON	KBMASK
	1								
C8	0*	T2CON*	T2MOD*	RCAP2L*	RCAP2H*	TL2*	TH2*	--	--
	1*	T3CON*	T3MOD*	RCAP3L*	RCAP3H*	TL3*	TH3*		
C0	0	XICON	--	--	ADCFG0	ADCON0	ADCDL	ADCDH	CKCON0
	1								
B8	0	IP0L	SADEN	--	--	--	--	--	CKCON1
	1								
B0	0*	P3	P3M0	P3M1	P4M0	PUCON0*	P5M0*	--	IP0H
	1*					PUCON1*	P6M0*		
A8	0	IE	SADDR	--	--	SFRPI*	EIE1	EIP1L	EIP1H
	1								
A0	0	P2	AUXR0	AUXR1	AUXR2	--	EIE2	EIP2L	EIP2H
	1								
98	0*	S0CON*	S0BUF*	S0CFG*	S1CFG*	--	--	--	--
	1*	S1CON*	S1BUF*	S1BRT*	S1BRC*				
90	0	P1	P1M0	P1AIO	P0M0	--	P2M0	--	PCON1
	1								
88	0	TCON	TMOD	TL0	TL1	TH0	TH1	SFIE	STRETCH
	1								
80	0	P0	SP	DPL	DPH	SPSTAT	SPCON	SPDAT	PCON0
	1								
		0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F

\*: User needs to set SFRPI as SFRPI=0x00, or SFRPI=0x01 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
--	--	--	--	PIDX3	PIDX2	PIDX1	PIDX0
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. The available pages are only page "0" and "1".

There are 13 register sets in Page 0, S0CON(98H), S0BUF(99H), S0CFG(9AH), S1CFG(9BH), PUCON0(B4H), P5M0(B5H), T2CON(C8H), T2MOD(C9H), RCAP2L(CAH), RCAP2H(CBH),

TL2(CCH), TH2(CDH) and P5(F8H).

13 register sets in Page 1, S1CON(98H), S1BUF(99H) and S1BRT(9AH), S1BRC(9BH), PUCON1(B4H), P6M0(B5H), T3CON(C8H), T3MOD(C9H), RCAP3L(CAH), RCAP3H(CBH), TL3(CCH), TH3(CDH) and P6(F8H).

PIDX[3:0]	Selected Page
0000	Page 0
0001	Page 1
0010	Page 2
0011	Page 3
.....	.....
.....	.....
.....	.....
1111	Page F

## 4.2. SFR Bit Assignment (Page 0~F)

Table 4-2. SFR Bit Assignment (Page 0~F)

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	
<i>P0</i>	<i>Port 0</i>	<i>80H</i>	<i>P0.7</i>	<i>P0.6</i>	<i>P0.5</i>	<i>P0.4</i>	<i>P0.3</i>	<i>P0.2</i>	<i>P0.1</i>	<i>P0.0</i>	11111111
SP	Stack Pointer	81H									00000111
DPL	Data Pointer Low	82H									00000000
DPH	Data Pointer High	83H									00000000
SPSTAT	SPI Status Register	84H	SPIF	WCOL	THRF	SPIBSY	--	--	--	SPR2	0000XXXX
SPCON	SPI Control Register	85H	SSIG	SPEN	DORD	MSTR	CPOL	CPHA	SPR1	SPR0	00000100
SPDAT	SPI Data Register	86H									00000000
PCON0	Power Control 0	87H	SMOD1	SMOD0	--	POF0	GF1	GF0	PD	IDL	00010000
<i>TCON</i>	<i>Timer Control</i>	<i>88H</i>	<i>TF1</i>	<i>TR1</i>	<i>TF0</i>	<i>TR0</i>	<i>IE1</i>	<i>IT1</i>	<i>IE0</i>	<i>IT0</i>	00000000
TMOD	Timer Mode	89H	GATE	C/T	M1	M0	GATE	C/T	M1	M0	00000000
TL0	Timer Low 0	8AH									00000000
TL1	Timer Low 1	8BH									00000000
TH0	Timer High 0	8CH									00000000
TH1	Timer High 1	8DH									00000000
SFIE	System Flag INT En.	8EH	--	--	--	--	--	BOF1IE	BOF0IE	WDTFIE	XXXXX000
STRETCH	MOVX Timing Stretch	8FH	EMA11	--	ALES1	ALES0	RWSH	RWS2	RWS1	RWS0	0X000000
<i>P1</i>	<i>Port 1</i>	<i>90H</i>	<i>P1.7</i>	<i>P1.6</i>	<i>P1.5</i>	<i>P1.4</i>	<i>P1.3</i>	<i>P1.2</i>	<i>P1.1</i>	<i>P1.0</i>	11111111
P1M0	P1 Mode Register 0	91H	P1M0.7	P1M0.6	P1M0.5	P1M0.4	P1M0.3	P1M0.2	P1M0.1	P1M0.0	00000000
P1AIO	P1 Analog Input Only	92H	P1AIO	P1AIO	P1AIO	P1AIO	P1AIO	P1AIO	P1AIO	P1AIO	00000000
P0M0	P0 Mode Register 0	93H	P0M0.7	P0M0.6	P0M0.5	P0M0.4	P0M0.3	P0M0.2	P0M0.1	P0M0.0	00000000
P2M0	P2 Mode Register 0	95H	P2M0.7	P2M0.6	P2M0.5	P2M0.4	P2M0.3	P2M0.2	P2M0.1	P2M0.0	00000000
PCON1	Power Control 1	97H	SWRF	EXRF	--	--	--	BOF1	BOF0	WDTF	00XXX000
<i>S0CON</i>	<i>Serial 0 Control</i>	<i>98H</i>	<i>SM00/FE</i>	<i>SM10</i>	<i>SM20</i>	<i>REN0</i>	<i>TB80</i>	<i>RB80</i>	<i>T10</i>	<i>R10</i>	00000000
<i>S1CON</i>	<i>Serial 1 Control</i>	<i>98H</i>	<i>SM01</i>	<i>SM11</i>	<i>SM21</i>	<i>REN1</i>	<i>TB81</i>	<i>RB81</i>	<i>T11</i>	<i>R11</i>	00000000
S0BUF	Serial 0 Buffer	99H									XXXXXXXXXX
S1BUF	Serial 1 Buffer	99H									XXXXXXXXXX
S0CFG	Serial 0 Configuration	9AH	URTS	SMOD2	URM0X6	--	--	--	--	--	000XXXXX
S1BRT	S1 Baud-Rate Timer	9AH									00000000
S1CFG	Serial 1 Configuration	9BH	--	--	--	S1TR	S1MOD1	S1TX12	S1CKOE	S1TME	XXXX0000
S1BRC	S1 Baud-Rate Counter	9BH									00000000
<i>P2</i>	<i>Port 2</i>	<i>A0H</i>	<i>P2.7</i>	<i>P2.6</i>	<i>P2.5</i>	<i>P2.4</i>	<i>P2.3</i>	<i>P2.2</i>	<i>P2.1</i>	<i>P2.0</i>	11111111
AUXR0	Auxiliary Register 0	A1H	P60OC1	P60OC0	P60FD	--	P4FS1	P4FS0	INT1H	INT0H	000X0000
AUXR1	Auxiliary Register 1	A2H	KBIPS1	KBIPS0	P5SPI	P5S1	P5T2	P6PCA	EXTRAM	DPS	00000000
AUXR2	Auxiliary Register 2	A3H	INT3IS1	INT3IS0	INT2IS1	INT2IS0	T1X12	T0X12	T1CKOE	T0CKOE	00000000
EIE2	Extended INT Enable 2	A5H	--	--	--	--	--	--	--	ET3	XXXXXXXXX0
EIP2L	Ext. INT Priority 2 Low	A6H	--	--	--	--	--	--	--	PT3L	XXXXXXXXX0
EIP2H	Ext. INT Priority 2 High	A7H	--	--	--	--	--	--	--	PT3H	XXXXXXXXX0
IE	Interrupt Enable	A8H	EA	GF4	ET2	ES0	ET1	EX1	ET0	EX0	00000000
SADDR	Slave Address	A9H									00000000
USBDAT	USB Data Register	AAH	UDAT7	UDAT6	UDAT5	UDAT4	UDAT3	UDAT2	UDAT1	UDAT0	XXXXXXXXXX
USBADR	USB Indirect Address	ABH	UBSY	--	USFRA5	USFRA4	USFRA3	USFRA2	USFRA1	USFRA0	0X000000
SFRPI	SFR Page Index	ACH	--	--	--	--	IDX3	IDX2	IDX1	IDX0	XXXX0000
EIE1	Extended INT Enable 1	ADH	EUSB	ETWSI	EKB	ES1	ESF	EPCA	EADC	ESPI	00000000
EIP1L	Ext. INT Priority 1 Low	AEH	PUSBL	PTWIL	PKBL	PS1L	PSFL	PPCAL	PADCL	PSPIL	00000000
EIP1H	Ext. INT Priority 1 High	AFH	PUSBH	PTWIH	PKBH	PS1H	PSFH	PPCAH	PADCH	PSPIH	00000000
<i>P3</i>	<i>Port 3</i>	<i>B0H</i>	<i>P3.7</i>	<i>P3.6</i>	<i>P3.5</i>	<i>P3.4</i>	<i>P3.3</i>	<i>P3.2</i>	<i>P3.1</i>	<i>P3.0</i>	11111111
P3M0	P3 Mode Register 0	B1H	P3M0.7	P3M0.6	P3M0.5	P3M0.4	P3M0.3	P3M0.2	P3M0.1	P3M0.0	00000000
P3M1	P3 Mode Register 1	B2H	P3M1.7	P3M1.6	P3M1.5	P3M1.4	P3M1.3	P3M1.2	P3M1.1	P3M1.0	00000000
P4M0	P4 Mode Register 0	B3H	--	P4M0.6	P4M0.5	P4M0.4	P4M0.3	P4M0.2	P4M0.1	P4M0.0	X00000XX
PUCON0	Port Pull-Up Control 0	B4H	P4PU1	P4PU0	P2PU1	P2PU0	P1PU1	P1PU0	P0PU1	P0PU0	00000000
PUCON1	Port Pull-Up Control 1	B4H	--	--	--	--	P6PU1	P6PU0	P5PU1	P5PU0	XXXX0000
P5M0	P5 Mode Register 0	B5H	P5M0.7	P5M0.6	P5M0.5	P5M0.4	P5M0.3	P5M0.2	P5M0.1	P5M0.0	00000000
P6M0	P6 Mode Register 0	B5H	P6M0.7	P6M0.6	P6M0.5	P6M0.4	P6M0.3	P6M0.2	P6M0.1	P6M0.0	00000000
IP0H	Interrupt Priority 0 High	B7H	PX3H	PX2H	PT2H	PSH	PT1H	PX1H	PT0H	PX0H	00000000
<i>IPOL</i>	<i>Interrupt Priority Low</i>	<i>B8H</i>	<i>PX3L</i>	<i>PX2L</i>	<i>PT2L</i>	<i>PSL</i>	<i>PT1L</i>	<i>PX1L</i>	<i>PT0L</i>	<i>PX0L</i>	00000000
SADEN	Slave Address Mask	B9H									00000000
CKCON1	Clock Control 1	BFH	--	--	XCKS5	XCKS4	XCKS3	XCKS2	XCKS1	XCKS0	XX001011
XICON	External INT Control	C0H	INT3H	EX3	IE3	IT3	INT2H	EX2	IE2	IT2	00000000
ADCFG0	ADC Configuration 0	C3H	ADCKS2	ADCKS1	ADCKS0	ADRJ	--	--	ADTM1	ADTM0	0000XX00
ADCON0	ADC Control 0	C4H	ADCEN	ADCMS	AZEN	ADCI	ADCS	CHS2	CHS1	CHS0	00000000
ADCDL	ADC Data Low	C5H	ADCV.3	ADCV.2	ADCV.1	ADCV.0	--	--	--	--	0000XXXX
ADCDH	ADC Data High	C6H	ADCV.11	ADCV.10	ADCV.9	ADCV.8	ADCV.7	ADCV.6	ADCV.5	ADCV.4	00000000



CKCON0	Clock Control 0	C7H	ENUSB	ENCKM	CKMIS1	CKMIS0		SCKS2	SCKS1	SCKS0	0001X000
T2CON	Timer 2 Control	C8H	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2	00000000
T3CON	Timer 3 Control	C8H	TF3	EXF3	--	--	EXEN3	TR3	C/T3	--	00XX000X
T2MOD	Timer2 mode	C9H	--	--	T2EXH	T2X12	--	--	T2OE	DCEN2	XX00XX00
T3MOD	Timer3 mode	C9H	T3SPL	TL3X12	T3EXH	T3X12	--	--	T3OE	--	0000XX0X
RCAP2L	Timer2 Capture Low	CAH									00000000
RCAP3L	Timer3 Capture Low	CAH									00000000
RCAP2H	Timer2 Capture High	CBH									00000000
RCAP3H	Timer3 Capture High	CBH									00000000
TL2	Timer Low 2	CCH									00000000
TL3	Timer Low 3	CCH									00000000
TH2	Timer High 2	CDH									00000000
TH3	Timer High 3	CDH									00000000
PSW	Program Status Word	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	00000000
SIADR	TWSI Address Reg.	D1H								GC	00000000
SIDAT	TWSI Data Reg.	D2H									00000000
SISTA	TWSI Status Reg.	D3H									11111000
SICON	TWSI Control Reg.	D4H	CR2	ENSI	STA	STO	SI	AA	CR1	CR0	00000000
KBPATN	Keypad Pattern	D5H									11111111
KBCON	Keypad Control	D6H							PATNS	KBIF	XXXXXX00
KBMASK	Keypad Int. Mask	D7H									00000000
CCON	PCA Control Reg.	D8H	CF	CR	CCF5	CCF4	CCF3	CCF2	CCF1	CCF0	00000000
CMOD	PCA Mode Reg.	D9H	CIDL	--	--	--	--	CPS1	CPS0	ECF	0xxxx000
CCAPM0	PCA Module0 Mode	DAH	--	ECOM0	CAPP0	CAPN0	MAT0	TOG0	PWM0	ECCF0	X0000000
CCAPM1	PCA Module1 Mode	DBH	--	ECOM1	CAPP1	CAPN1	MAT1	TOG1	PWM1	ECCF1	X0000000
CCAPM2	PCA Module2 Mode	DCH	--	ECOM2	CAPP2	CAPN2	MAT2	TOG2	PWM2	ECCF2	X0000000
CCAPM3	PCA Module3 Mode	DDH	--	ECOM3	CAPP3	CAPN3	MAT3	TOG3	PWM3	ECCF3	X0000000
CCAPM4	PCA Module4 Mode	DEH	--	ECOM4	CAPP4	CAPN4	MAT4	TOG4	PWM4	ECCF4	X0000000
CCAPM5	PCA Module5 Mode	DFH	--	ECOM5	CAPP5	CAPN5	MAT5	TOG5	PWM5	ECCF5	X0000000
ACC	Accumulator	E0H	ACC.7	ACC.6	ACC.5	ACC.4	ACC.3	ACC.2	ACC.1	ACC.0	00000000
WDTCR	Watch-dog-timer Control register	E1H	WREN	NSW	ENW	CLW	WIDL	PS2	PS1	PS0	00000000
IFD	ISP Flash data	E2H									11111111
IFADRH	ISP Flash address High	E3H									00000000
IFADRL	ISP Flash Address Low	E4H									00000000
IFMT	ISP Mode Table	E5H	--	--	--	--	--	MS.2	MS.1	MS.0	XXXXX000
SCMD	ISP Serial Command	E6H									XXXXXXXX
ISPCR	ISP Control Register	E7H	ISPEN	SWBS	SWRST	CFAIL	MISPF	--	--	--	00000XXX
P4	Port 4	E8H	--	P4.6	P4.5	P4.4	P4.3	P4.2	P4.1	P4.0	X1111111
CL	PCA base timer Low	E9H									00000000
CCAP0L	PCA module0 capture Low	EAH									00000000
CCAP1L	PCA module1 capture Low	EBH									00000000
CCAP2L	PCA module2 capture Low	ECH									00000000
CCAP3L	PCA module3 capture Low	EDH									00000000
CCAP4L	PCA module4 capture Low	EEH									00000000
CCAP5L	PCA module5 capture Low	EFH									00000000
B	B Register	F0H	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0	00000000
PCAPWM0	PCA PWM0 Mode	F2H	P0RS1	P0RS0	P0PS2	P0PS1	P0PS0	P0INV	EPC0H	EPC0L	00000000
PCAPWM1	PCA PWM1 Mode	F3H	P1RS1	P1RS0	P1PS2	P1PS1	P1PS0	P1INV	EPC1H	EPC1L	00000000
PCAPWM2	PCA PWM2 Mode	F4H	P2RS1	P2RS0	P2PS2	P2PS1	P2PS0	P2INV	EPC2H	EPC2L	00000000
PCAPWM3	PCA PWM3 Mode	F5H	P3RS1	P3RS0	P3PS2	P3PS1	P3PS0	P3INV	EPC3H	EPC3L	00000000
PCAPWM4	PCA PWM4 Mode	F6H	P4RS1	P4RS0	P4PS2	P4PS1	P4PS0	P4INV	EPC4H	EPC4L	00000000
PCAPWM5	PCA PWM5 Mode	F7H	P5RS1	P5RS0	P5PS2	P5PS1	P5PS0	P5INV	EPC5H	EPC5L	00000000
P5	Port 5	F8H	P5.7	P5.6	P5.5	P5.4	P5.3	P5.2	P5.1	P5.0	11111111
P6	Port 6	F8H	P6.7	P6.6	P6.5	P6.4	P6.3	P6.2	P6.1	P6.0	11111111
CH	PCA base timer High	F9H									00000000
CCAP0H	PCA Module0 capture High	FAH									00000000
CCAP1H	PCA Module1 capture High	FBH									00000000
CCAP2H	PCA Module2 capture High	FCH									00000000
CCAP3H	PCA Module3 capture High	FDH									00000000

CCAP4H	PCA Module4 capture High	FEH									00000000
CCAP5H	PCA Module5 capture High	FFH									00000000

### 4.3. Auxiliary SFR Map (Page P)

**MG84FG516** 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 5 physical bytes and 7 logical bytes. The 5 physical bytes include IAPLB, CKCON2, PCON2, PCON3 and SPCON0. The 7 logical bytes include PCON0, PCON1, CKCON0, CKCON1, WDTCR, P4 and P6. Write on the 7 logical bytes gets the coherence content with the same SFR in Page 0~F. Please refer Section “[25 Page P SFR Access](#)” for more detail information.

Table 4–3. Auxiliary SFR Map (Page P)

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F
F8	P6							
F0								
E8	P4							
E0		WDTCR						
D8								
D0								
C8								
C0								CKCON0
B8								CKCON1
B0								
A8								
A0								
98								
90								PCON1
88								
80								PCON0
78								
70								
68								
60								
58								
50								
48	SPCON0							
40	CKCON2				PCON2	PCON3		
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

#### 4.4. Auxiliary SFR Bit Assignment (Page P)

Table 4–4. 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	
IAPLB	IAP Low Boundary	03H	IAPLB6	IAPLB5	IAPLB4	IAPLB3	IAPLB2	IAPLB1	IAPLB0	--	
CKCON2	Clock Control 2	40H	XTGS1	XTGS0	XTALE	IHRCOE	MCKS1	MCKS0	OSCS1	OSCS0	01010000
PCON2	Power Control 2	44H	HSE	IAPO	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	1	0000X1X1
PCON3	Power Control 3	45H	0	0	0	0	AWBOD1	0	0	OCDE	00000001
SPCON0	SFR Page Control 0	48H	--	P6CTL	P4CTL	WRCTL	CKCTL1	CKCTL0	PWCTL1	PWCTL0	X0000000

#### Sample Code of Page-P SFR write:

```

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;

```

## 5. Pin Configurations

### 5.1. Package Instruction

Figure 5–1. MG84FG516AD64 Top View

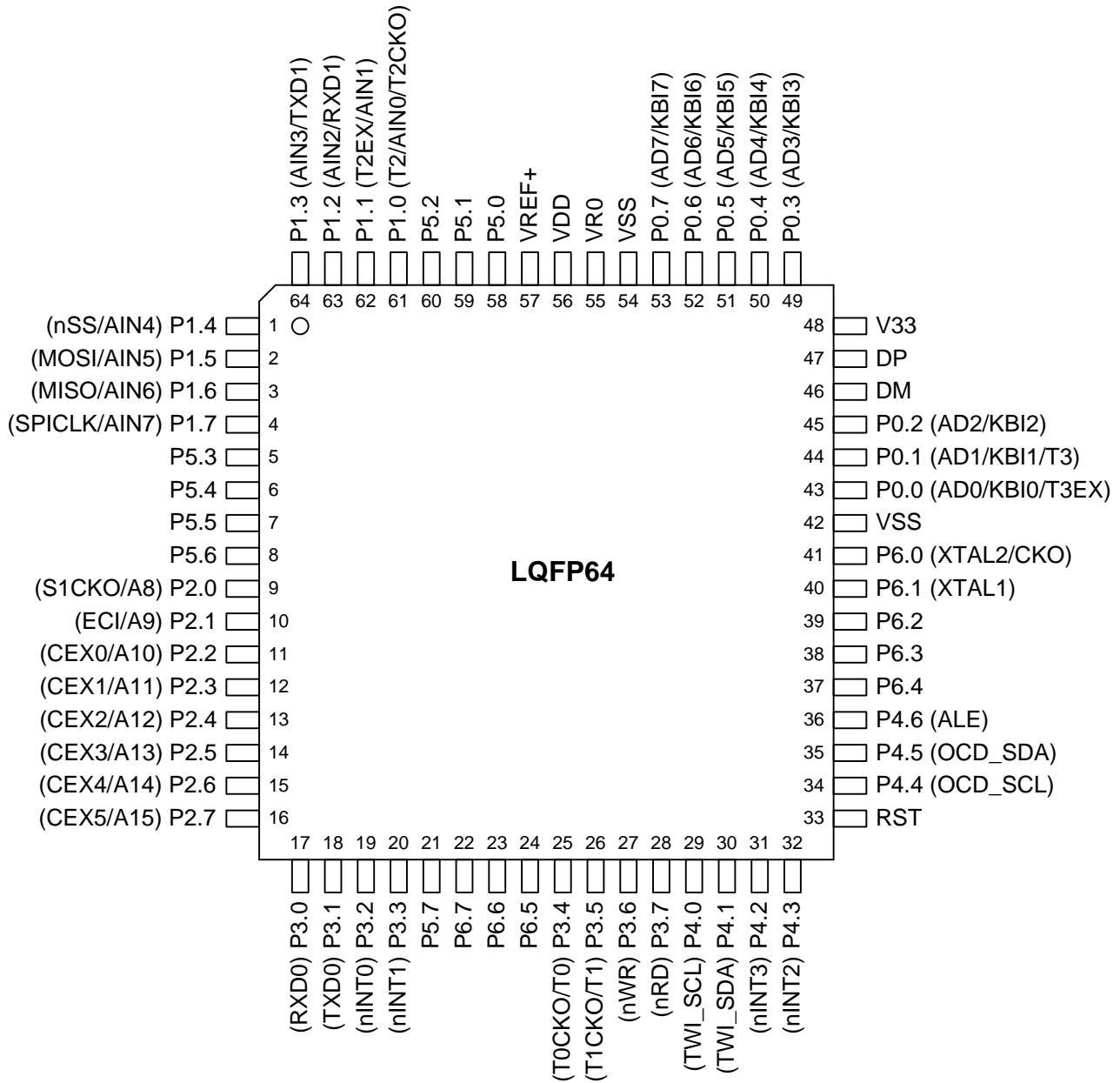
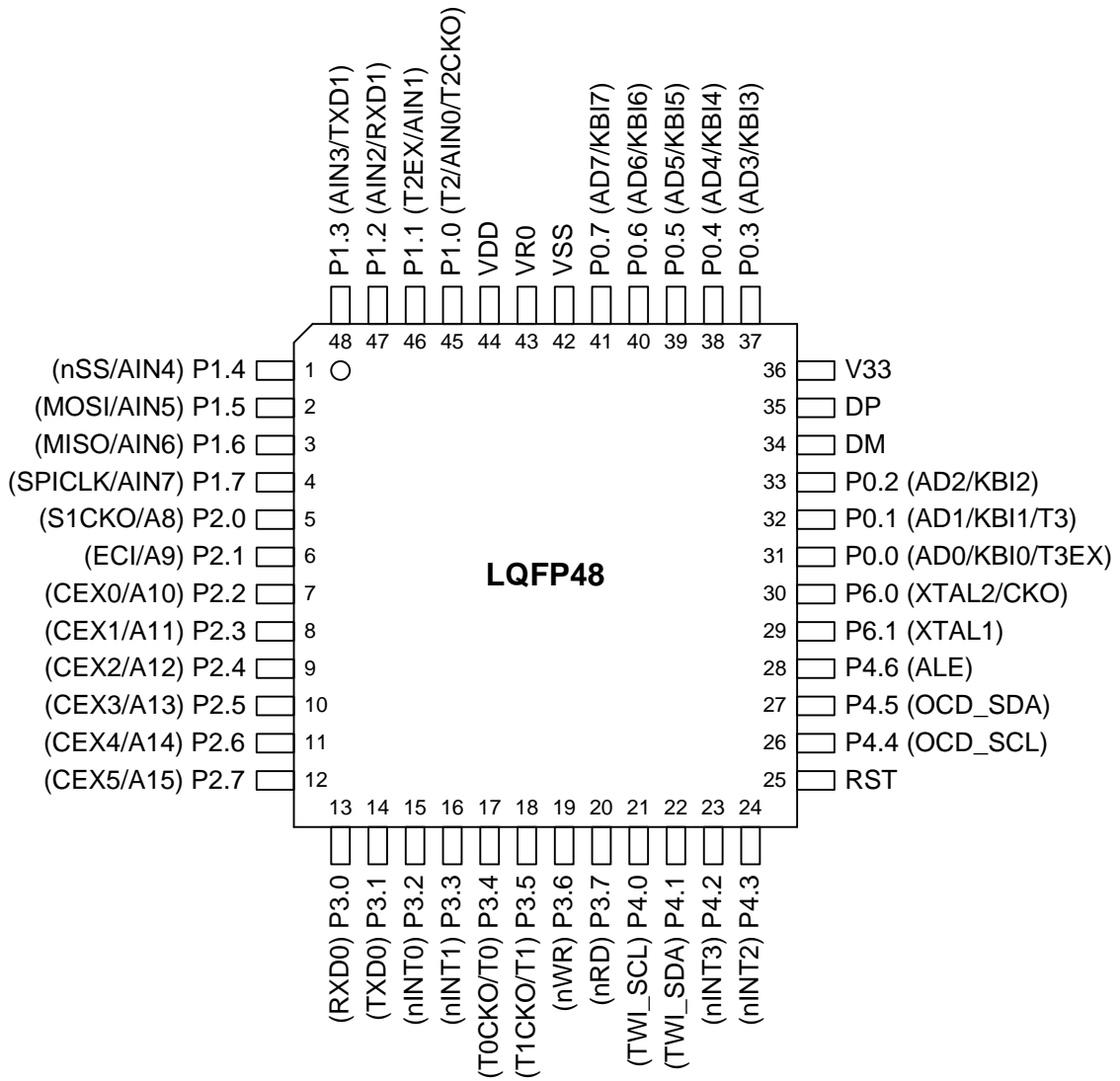


Figure 5–2. MG84FG516AD48 Top View



## 5.2. Pin Description

Table 5–1. Pin Description

MNEMONIC	PIN NUMBER		I/O TYPE	DESCRIPTION
	64-Pin LQFP	48-Pin LQFP		
P0.0 (AD0) (KB10) (T3EX)	43	31	I/O	* Port 0.0. * AD0: multiplexed A0/D0 during external data memory access. * KB10: keypad input 0. * T3EX: Timer/Counter 3 external control input.
P0.1 (AD1) (KB11) (T3/T3CKO)	44	32	I/O	* Port 0.1. * AD1: multiplexed A1/D1 during external data memory access. * KB11: keypad input 1. * T3/T3CKO: Timer/Counter 3 external clock input or programmable clock output.
P0.2 (AD2) (KB12)	45	33	I/O	* Port 0.2. * AD2: multiplexed A2/D2 during external data memory access. * KB12: keypad input 2.
P0.3 (AD3) (KB13)	49	37	I/O	* Port 0.3. * AD3: multiplexed A3/D3 during external data memory access. * KB13: keypad input 3.
P0.4 (AD4) (KB14)	50	38	I/O	* Port 0.4. * AD4: multiplexed A4/D4 during external data memory access. * KB14: keypad input 4.
P0.5 (AD5) (KB15)	51	39	I/O	* Port 0.5. * AD5: multiplexed A5/D5 during external data memory access. * KB15: keypad input 5.
P0.6 (AD6) (KB16)	52	40	I/O	* Port 0.6. * AD6: multiplexed A6/D6 during external data memory access. * KB16: keypad input 6.
P0.7 (AD7) (KB17)	53	41	I/O	* Port 0.7. * AD7: multiplexed A7/D7 during external data memory access. * KB17: keypad input 7.
P1.0 (T2/T2CKO) (AIN0)	61	45	I/O	* Port 1.0. * T2/T2CKO: Timer/Counter 2 external clock input or programmable clock output. * AIN0: ADC channel-0 analog input.
P1.1 (T2EX) (AIN1)	62	46	I/O	* Port 1.1. * T2EX: Timer/Counter 2 external control input. * AIN1: ADC channel-1 analog input.
P1.2 (AIN2) (RXD1)	63	47	I/O	* Port 1.2. * AIN2: ADC channel-2 analog input. * RXD1: UART1 serial input port.
P1.3 (AIN3) (TXD1)	64	48	I/O	* Port 1.3. * AIN3: ADC channel-3 analog input. * TXD1: UART1 serial output port.
P1.4 (AIN4) (nSS)	1	1	I/O	* Port 1.4. * AIN4: ADC channel-4 analog input. * nSS: SPI Slave select.
P1.5 (AIN5) (MOSI)	2	2	I/O	* Port 1.5. * AIN5: ADC channel-5 analog input. * MOSI: SPI master out & slave in.
P1.6 (AIN6) (MISO)	3	3	I/O	* Port 1.6. * AIN6: ADC channel-6 analog input. * MISO: SPI master in & slave out.
P1.7 (AIN7) (SPICLK)	4	4	I/O	* Port 1.7. * AIN7: ADC channel-7 analog input. * SPICLK: SPI clock, output for master and input for slave.
P2.0 (A8) (S1CKO)	9	5	I/O	* Port 2.0. * A8: A8 output during external data memory access. * S1CKO: S1BRT Clock Output.
P2.1 (A9) (ECI)	10	6	I/O	* Port 2.1. * A9: A9 output during external data memory access. * ECI: PCA external clock input.
P2.2 (A10)	11	7	I/O	* Port 2.2. * A10: A10 output during external data memory access.

(CEX0)				* CEX0: PCA module-0 external I/O.
P2.3 (A11) (CEX1)	12	8	I/O	* Port 2.3. * A11: A11 output during external data memory access. * CEX1: PCA module-1 external I/O.
P2.4 (A12) (CEX2)	13	9	I/O	* Port 2.4. * A12: A12 output during external data memory access. * CEX2: PCA module-2 external I/O.
P2.5 (A13) (CEX3)	14	10	I/O	* Port 2.5. * A13: A13 output during external data memory access. * CEX3: PCA module-3 external I/O.
P2.6 (A14) (CEX4)	15	11	I/O	* Port 2.6. * A14: A14 output during external data memory access. * CEX4: PCA module-4 external I/O.
P2.7 (A15) (CEX5)	16	12	I/O	* Port 2.7. * A15: A15 output during external data memory access. * CEX5: PCA module-5 external I/O.
P3.0 (RXD0)	17	13	I/O	* Port 3.0. * RXD0: UART0 serial input port.
P3.1 (TXD0)	18	14	I/O	* Port 3.1. * TXD0: UART0 serial output port.
P3.2 (nINT0)	19	15	I/O	* Port 3.2. * nINT0: external interrupt 0 input.
P3.3 (nINT1)	20	16	I/O	* Port 3.3. * nINT1: external interrupt 1 input.
P3.4 (T0) (T0CKO)	25	17	I/O	* Port 3.4. * T0: Timer/Counter 0 external input. * T0CKO: programmable clock-out from Timer 0.
P3.5 (T1) (T1CKO)	26	18	I/O	* Port 3.5. * T1: Timer/Counter 1 external input. * T1CKO: programmable clock-out from Timer 1.
P3.6 (nWR)	27	19	I/O	* Port 3.6. * nWR: external data memory write strobe.
P3.7 (nRD)	28	20	I/O	* Port 3 bit-7. * nRD: external data memory read strobe.
P4.0 (TWI_SCL)	29	21	I/O	* Port 4.0. * TWI_SCL: serial clock of TWI.
P4.1 (TWI_SDA)	30	22	I/O	* Port 4.1. * TWI_SDA: serial data of TWI.
P4.2 (nINT3)	31	23	I/O	* Port 4.2. * nINT3: external interrupt 3 input.
P4.3 (nINT2)	32	24	I/O	* Port 4.3. * nINT2: external interrupt 2 input.
P4.4 (OCD_SCL)	34	26	I/O	* Port 4.4. * OCD_SCL: OCD interface, serial clock.
P4.5 (OCD_SDA)	35	27	I/O	* Port 4.5. * OCD_SDA: OCD interface, serial data.
P4.6 (ALE)	36	28	I/O	* Port 4.6. * ALE: Address Latch Enable, output pulse for latching the low byte of the address during an access cycle to external data memory.
P5.0	58	--	I/O	* Port 5.0. It is only accessed in SFR page "0"
P5.1	59	--	I/O	* Port 5.1. It is only accessed in SFR page "0"
P5.2	60	--	I/O	* Port 5.2. It is only accessed in SFR page "0"
P5.3	5	--	I/O	* Port 5.3. It is only accessed in SFR page "0"
P5.4	6	--	I/O	* Port 5.4. It is only accessed in SFR page "0"
P5.5	7	--	I/O	* Port 5.5. It is only accessed in SFR page "0"
P5.6	8	--	I/O	* Port 5.6. It is only accessed in SFR page "0"
P5.7	21	--	I/O	* Port 5.7. It is only accessed in SFR page "0"
P6.0 (CKO) (ECKI) (XTAL2)	41	30	I/O O I O	* Port 6.0. It is only accessed in SFR page "1". * CKO: Enable IHRCO/ILRCO output. * ECKI: In external clock input mode, this is clock input pin. * XTAL2: Output of on-chip crystal oscillating circuit.
P6.1 (XTAL1)	40	29	I/O I	* Port 6.1. It is only accessed in SFR page "1". * XTAL1: Input of on-chip crystal oscillating circuit.
P6.2	39	--	I/O	* Port 6.2. It is only accessed in SFR page "1"
P6.3	38	--	I/O	* Port 6.3. It is only accessed in SFR page "1"
P6.4	37	--	I/O	* Port 6.4. It is only accessed in SFR page "1"



P6.5	24	--	I/O	* Port 6.5. It is only accessed in SFR page "1".
P6.6	23	--	I/O	* Port 6.6. It is only accessed in SFR page "1".
P6.7	22	--	I/O	* Port 6.7. It is only accessed in SFR page "F".
RST	33	25	I	* RST: External RESET input, high active.
DM	46	34	I/O	* DM. USB DM.
DP	47	35	I/O	* DP. USB DP.
V33	48	36	I/O	* V33. 3.3V Regulator Output. Connect a 4.7uF to VSS.
VR0	55	43	I/O	* VR0. Voltage Reference 0. Connect 0.1uF and 4.7uF to VSS.
VREF+	57	--	I	* VREF+. ADC Voltage Reference + input.
VDD	56	44	P	Power supply input.
VSS	54	42	G	Ground, 0 V reference.

### 5.3. 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 peripherals Keypad interrupt, PCA, SPI, UART0, UART1, Timer 2 and Timer3, Port 0, Port 1, Port 2 and Port 3 serve the alternate function in the default state. However, the user may select Port 4 and Port 5 to serve their alternate function by setting the corresponding control bits P4KB, P4PCA, P5SPI and P4S1 in AUXR1 register. It is especially useful when the packages more than 40 pins are adopted. Note that only one of the four control bits can be set at any time.

#### AUXR0: Auxiliary Register 0

SFR Page = 0~F

SFR Address = 0xA1

RESET = 000X-0000

7	6	5	4	3	2	1	0
P60FC1	P60FC0	P60FD	--	P4FS1	P4FS0	INT1H	INT0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: P6.0 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 crystal mode, XTAL2 and XTAL1 are the alternated function of P6.0 and P6.1. In external clock input mode, P6.0 is the dedicated clock input pin. In internal oscillator condition, P6.0 provides the following selections for GPIO or clock source generator. When P60OC[1:0] index to non-P6.0 GPIO function, P6.0 will drive the on-chip RC oscillator output to provide the clock source for other devices.

P60OC[1:0]	P60 function	I/O mode
00	P60	By P6M0.0
01	MCK	By P6M0.0
10	MCK/2	By P6M0.0
11	MCK/4	By P6M0.0

Please refer Section “9 System Clock” to get the more detailed clock information. For clock-out on P6.0 function, it is recommended to set P6M0.0 to “1” which selects P6.0 as push-push output mode.

Bit 5: P60FD, P6.0 Fast Driving.

0: P6.0 output with default driving.

1: P6.0 output with fast driving enabled. If P6.0 is configured to clock output, enable this bit when P6.0 output frequency is more than 12MHz at 5V application or more than 6MHz at 3V application.

Bit 3~2: P4.4 and P4.5 alternated function selection.

P4FS[1:0]	P4.4	P4.5
00	P4.4	P4.5
01	Input for RXD0	Output for TXD0
10	Input for nINT2	Input for nINT3
11	Input for T3EX	Input for T3 or Output for T3CKO

#### AUXR1: Auxiliary Control Register 1

SFR Page = 0~F

SFR Address = 0xA2

RESET = 0000-0000

7	6	5	4	3	2	1	0
KBIPS1	KBIPS0	P5SPI	P5S1	P5T2	P6PCA	EXTRAM	DPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: KBIPS1~0, KBI Port Selection [1:0].

KBIPS1~0	KBI7~0
00	P0.7~P0.0
01	P2.7~P2.0
10	P5.7~P5.0
11	P6.7~P6.0

Bit 5: P5SPI, SPI interface on P5.7~P5.4.

0: Disable SPI function moved to P5.  
 1: Set SPI function on P5 as following definition.  
 'nSS' function in P1.4 is moved to P5.4.  
 'MOSI' function in P1.5 is moved to P5.5.  
 'MISO' function in P1.6 is moved to P5.6.  
 'SPICLK' function in P1.7 is moved to P5.7.

Bit 4: P5S1, Serial Port 1 (UART1) on P5.2/P5.3.  
 0: Disable UART1 function moved to P5.  
 1: Set UART1 RXD1/TXD1 on P5.2/P5.3 following definition.  
 'RXD1' function in P1.2 is moved to P5.2.  
 'TXD1' function in P1.3 is moved to P5.3.

Bit 3: P5T2, T2(T2CKO)/T2EX function on P5.0/P5.1.  
 0: Disable T2 function moved to P5.  
 1: Set UART1 T2(T2CKO)/T2EX on P5.0/P5.1 following definition.  
 'T2(T2CKO)' function in P1.0 is moved to P5.0.  
 'T2EX' function in P1.1 is moved to P5.1.

Bit 2: P6PCA, PCA function on P6.  
 0: Disable PCA function moved to P6.  
 1: Set PCA function on P6 as following definition.  
 'ECI' function in P2.1 is moved to P6.1.  
 'CEX0' function in P2.2 is moved to P6.2.  
 'CEX1' function in P2.3 is moved to P6.3.  
 'CEX2' function in P2.4 is moved to P6.4.  
 'CEX3' function in P2.5 is moved to P6.5.  
 'CEX4' function in P2.6 is moved to P6.6.  
 'CEX5' function in P2.7 is moved to P6.7.

**AUXR2: Auxiliary Register 2**

SFR Page = 0~F

SFR Address = 0xA3

RESET = 0000-0000

7	6	5	4	3	2	1	0
INT3IS1	INT3IS0	INT2IS1	INT2IS0	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: INT3IS1~0, nINT3 input selection bits which function is defined as following table.

INT3IS1~0	nINT3	Selected Port Pin
00	nINT3 Port Pin	P4.2 or P4.5
01	RXD1 Port Pin	P1.2 or P5.2
10	TWSI SDA Port Pin	P4.1
11	SPI nSS Port Pin	P1.4 or P5.4

Bit 5~4: INT2IS1~0, nINT2 input selection bits which function is defined as following table.

INT2IS1~0	nINT2	Selected Port Pin
00	nINT2 Port Pin	P4.3 or P4.4
01	RXD0 Port Pin	P3.0 or P4.4
10	TWSI SDA Port Pin	P4.1
11	SPI nSS Port Pin	P1.4 or P5.4

## 6. 8051 CPU Function Description

### 6.1. 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 “7.2 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 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.

## 6.2. CPU Timing

The **MG84FG516** 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 “[30 Instruction Set](#)” which includes the mnemonic, number of bytes, and number of clock cycles for each instruction.

## 6.3. 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.

## 7. Memory Organization

Like all 80C51 devices, the **MG84FG516** 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 64K bytes of program memory. In the **MG84FG516**, 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.

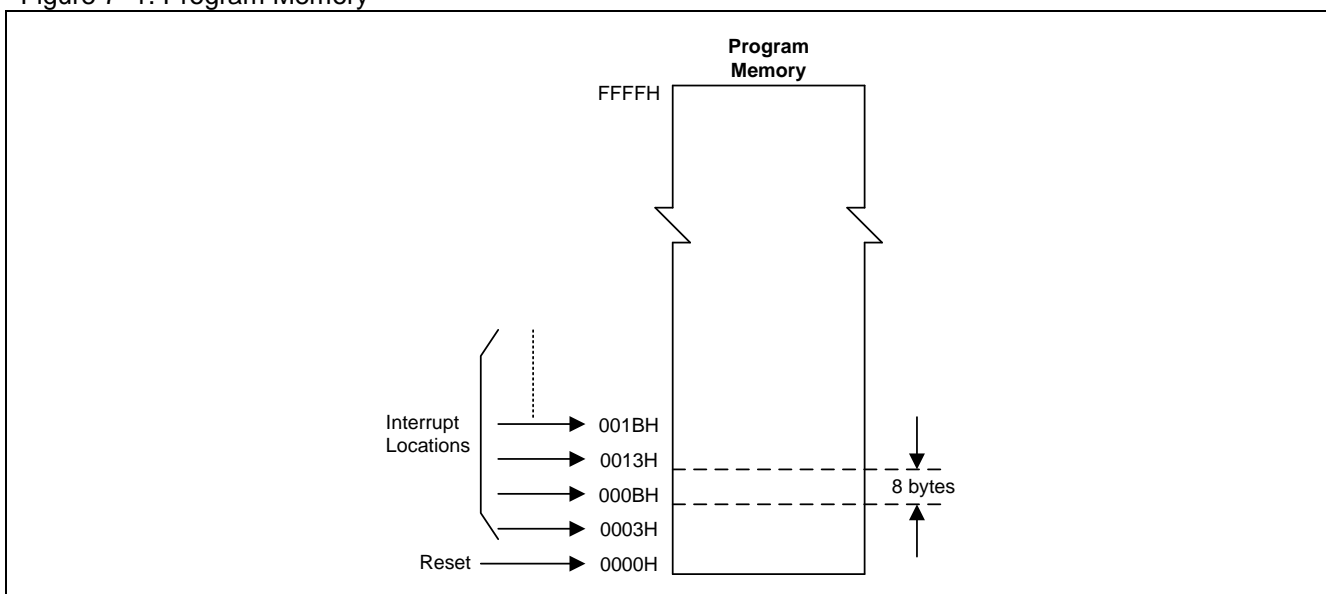
Data memory occupies a separate address space from program memory. In the **MG84FG516**, there are 256 bytes of internal scratch-pad RAM and **4096** bytes of on-chip expanded RAM(XRAM). If USB function is disabled, the USB buffer is moved to on-chip expanded RAM region which will provide **5120** bytes XRAM in **MG84FG516**. If USB function is enabled, the un-used USB buffer can be access by USB re-directed mechanism to extend MCU memory. Please refer Section “**23.4 Access USB 1K FIFO**” for more detail description.

### 7.1. On-Chip Program Flash

Program memory is the memory which stores the program codes for the CPU to execute, as shown in **Figure 7–1**. 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 7–1. Program Memory



## 7.2. On-Chip Data RAM

Figure 7–2 shows the internal and external data memory spaces available to the **MG84FG516** 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 7–3. 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 7–4 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.

To access the external data memory, the EXTRAM bit should be set to 1. Accesses to external data memory can use either a 16-bit address (using 'MOVX @DPTR') or an 8-bit address (using 'MOVX @Ri'), as described below.

### Accessing by an 8-bit address

8-bit addresses are often used in conjunction with one or more other I/O lines to page the RAM. If an 8-bit address is being used, the contents of the Port 2 SFR remain at the Port 2 pins throughout the external memory cycle. This will facilitate paging access. Figure 7–5 shows an example of a hardware configuration for accessing up to 2K bytes of external RAM. In multiplexed mode, Port 0 serves as a multiplexed address/data bus to the RAM, and 3 lines of Port 2 are being used to page the RAM. The CPU generates nRD and nWR (alternate functions of P3.7 and P3.6) to strobe the memory. Of course, the user may use any other I/O lines instead of P2 to page the RAM.

### Accessing by a 16-bit address

16-bit addresses are often used to access up to 64k bytes of external data memory. Figure 7–6 shows the hardware configuration for accessing 64K bytes of external RAM. Whenever a 16-bit address is used, in addition to the functioning of P0, nRD and nWR, the high byte of the address comes out on Port 2 and it is held during the read or write cycle.

### Accessing by a non-address mode

Non-address mode is provided to access the external data memory without MCU address limitation. Figure 7–7 shows the hardware configuration for accessing the external RAM. It also supports the FIFO structure memory, such as NAND flash. Whenever a non-address mode is selected, in addition to the functioning of P0, nRD and nWR, the address phase is skipped to speed up the access performance.

In multiplexed case, the low byte of the address is time-multiplexed with the data byte on Port 0. ALE (Address Latch Enable) should be used to capture the address byte into an external latch. The address byte is valid at the negative transition of ALE. Then, in a write cycle, the data byte to be written appears on Port 0 just before nWR is activated, and remains there until after nWR is deactivated. In a read cycle, the incoming byte is accepted at Port 0 just before the read strobe is deactivated. During any access to external memory, the CPU writes 0FFH to the Port 0 latch (the Special Function Register), thus obliterating whatever information the Port 0 SFR may have been holding.

To access the on-chip expanded RAM (XRAM), the EXTRAM bit should be cleared to 0. Refer to Figure 7–2, the 4096/5120 bytes of XRAM (0000H to 0FFFH/13FFFH) are indirectly accessed by move external instruction, MOVX.



An access to XRAM will have not any outputting of address, address latch enable and read/write strobe. That means P0, P2, P4.6(ALE), P3.6 (nWR) and P3.7 (nRD) will keep unchanged during access of on-chip XRAM.

Figure 7–2. Data Memory

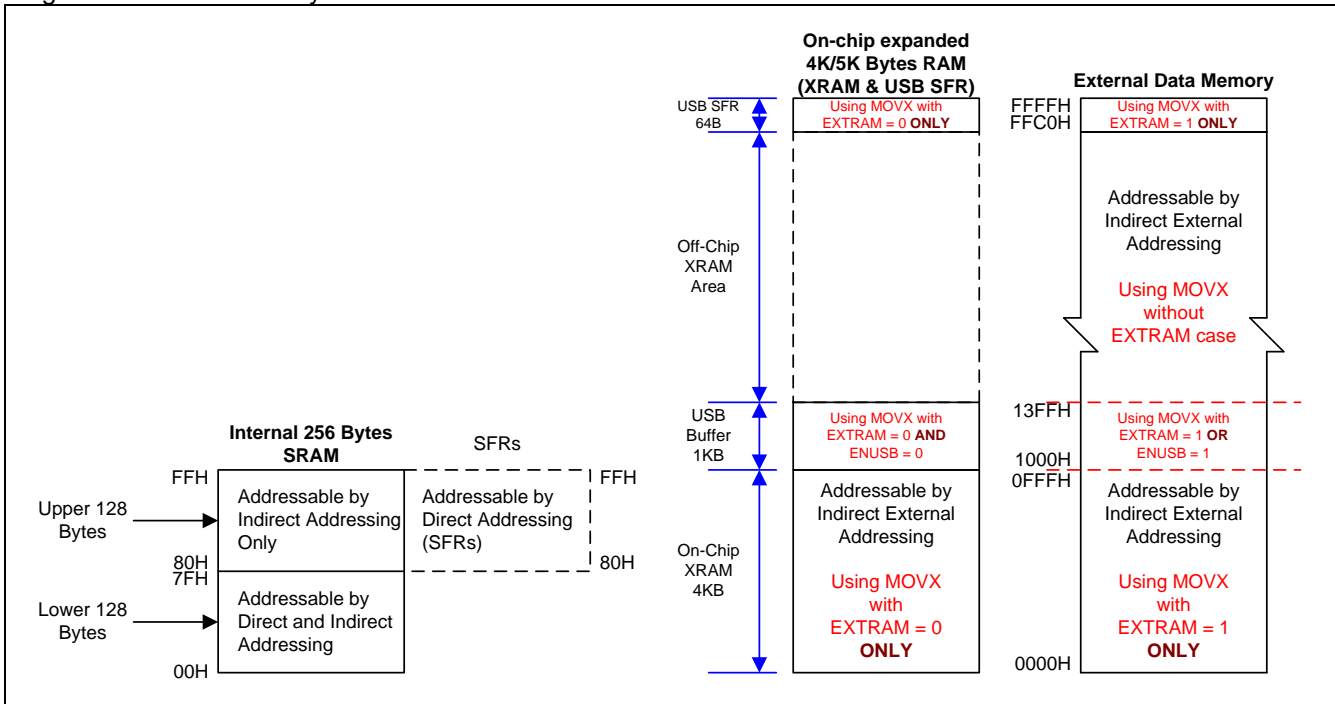


Figure 7–3. Lower 128 Bytes of Internal RAM

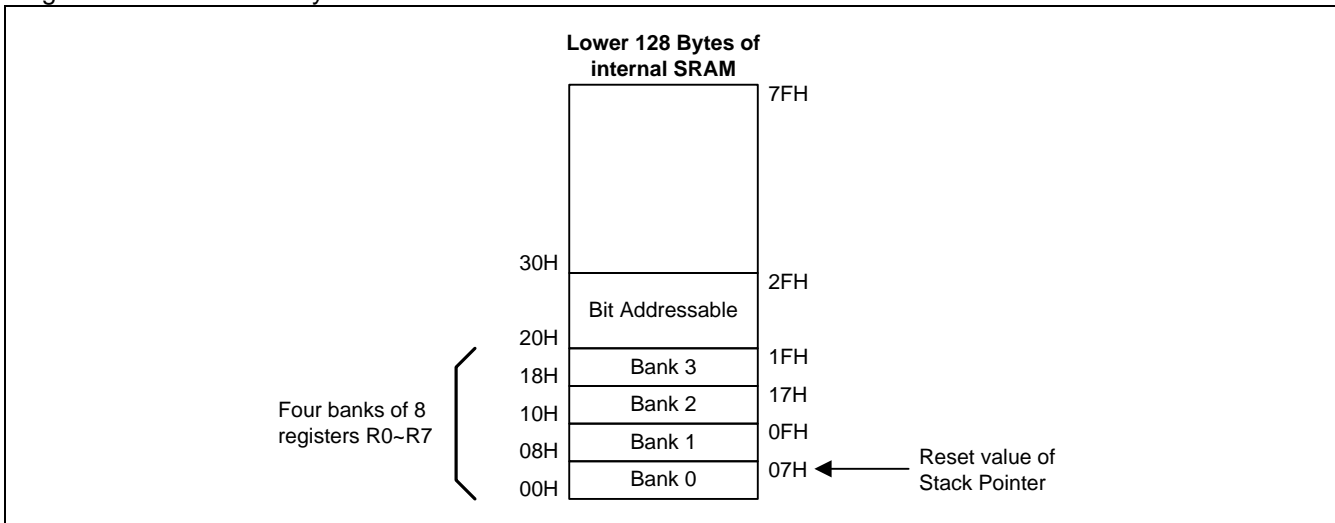


Figure 7–4. SFR Space

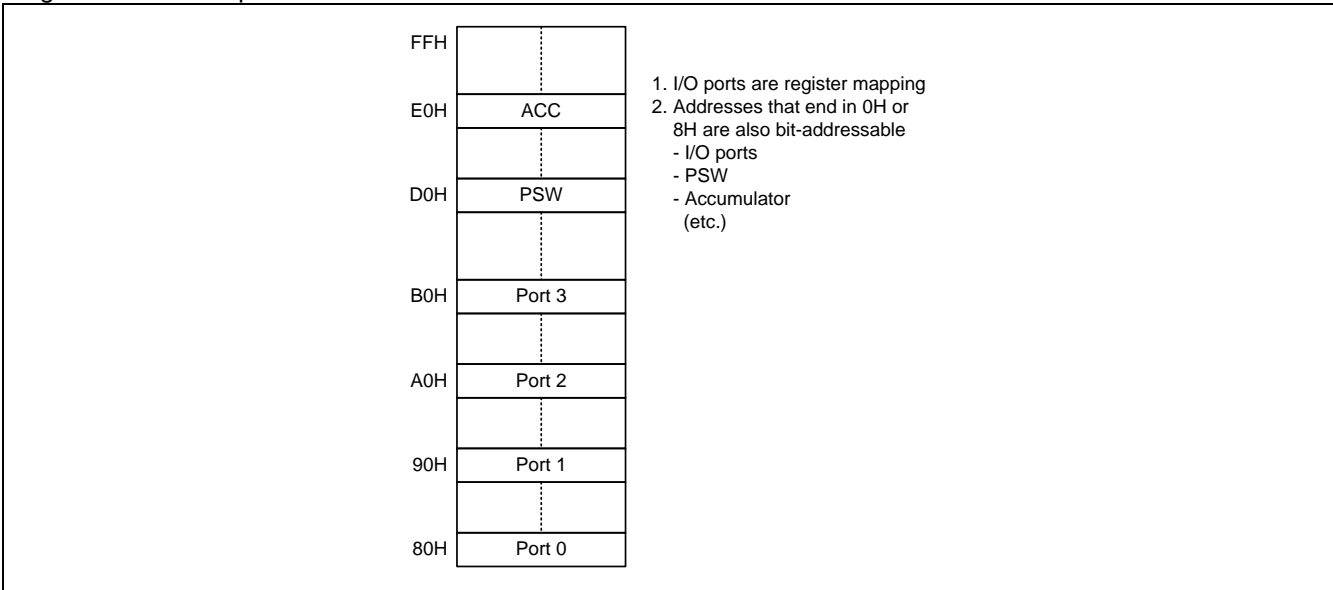
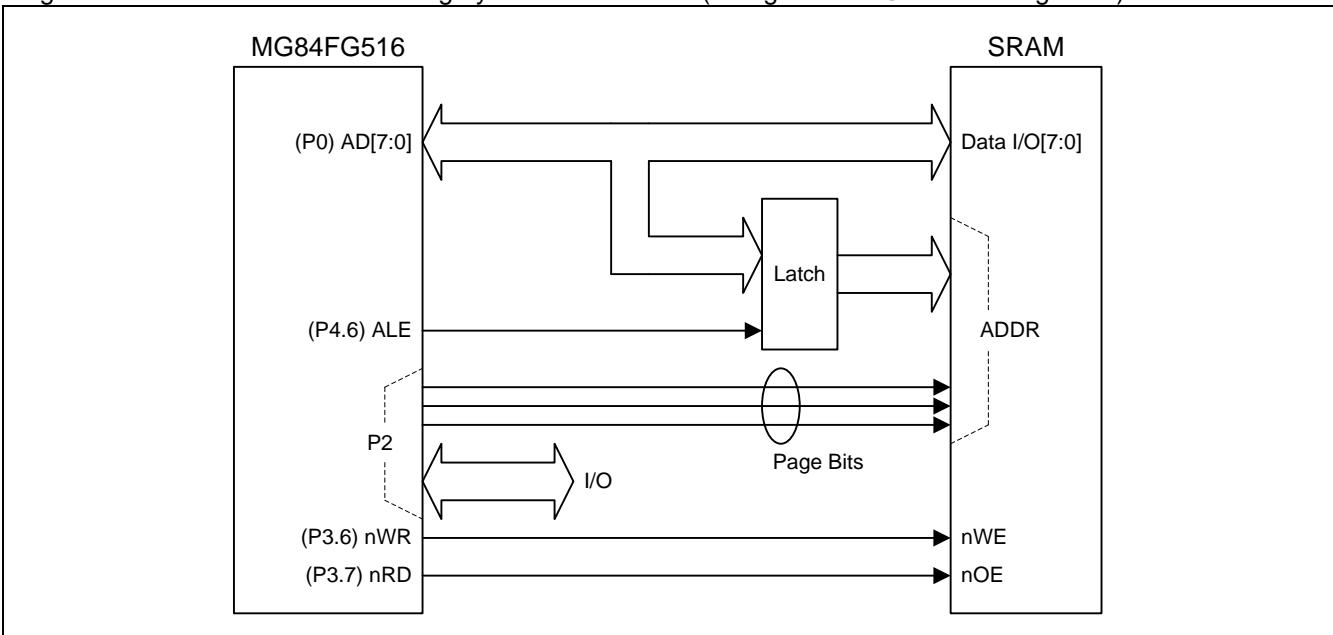
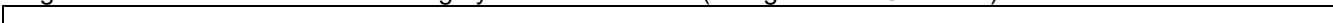


Figure 7–5. External RAM Accessing by an 8-Bit Address (Using ‘MOVX @ Ri’ and Page Bits)



Note that in this case, the other bits of P2 are available as general I/O pins.

Figure 7–6. External RAM Accessing by a 16-Bit Address (Using ‘MOVX @ DPTR’)



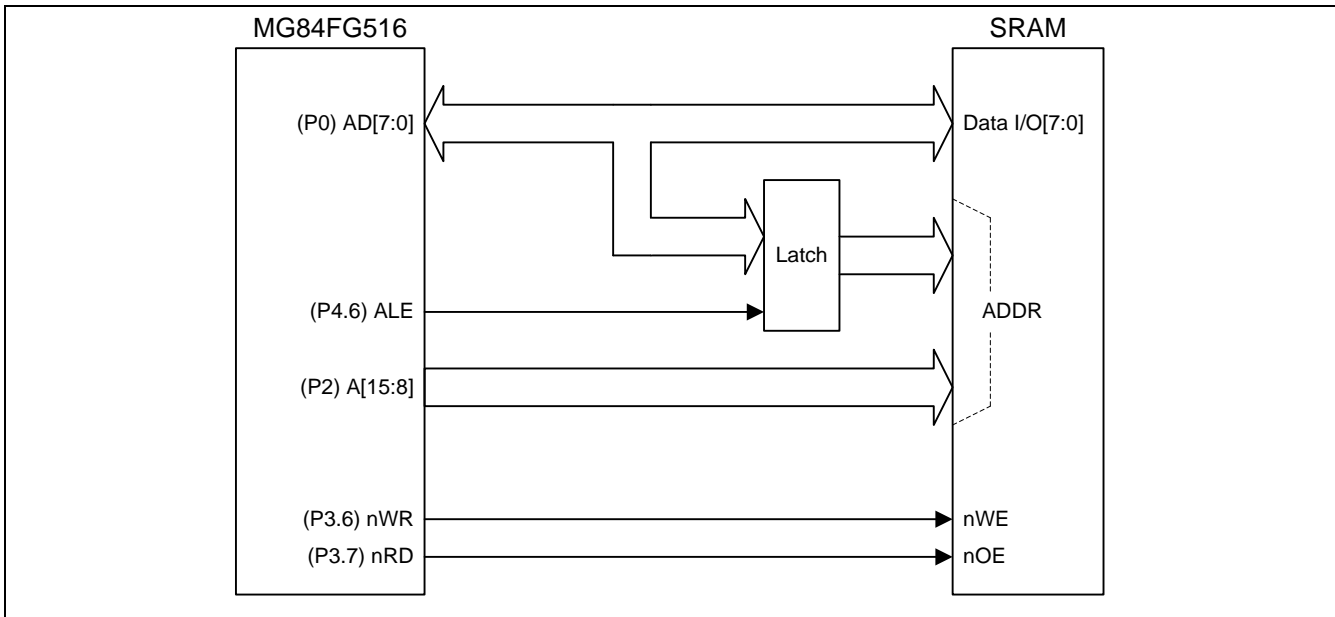
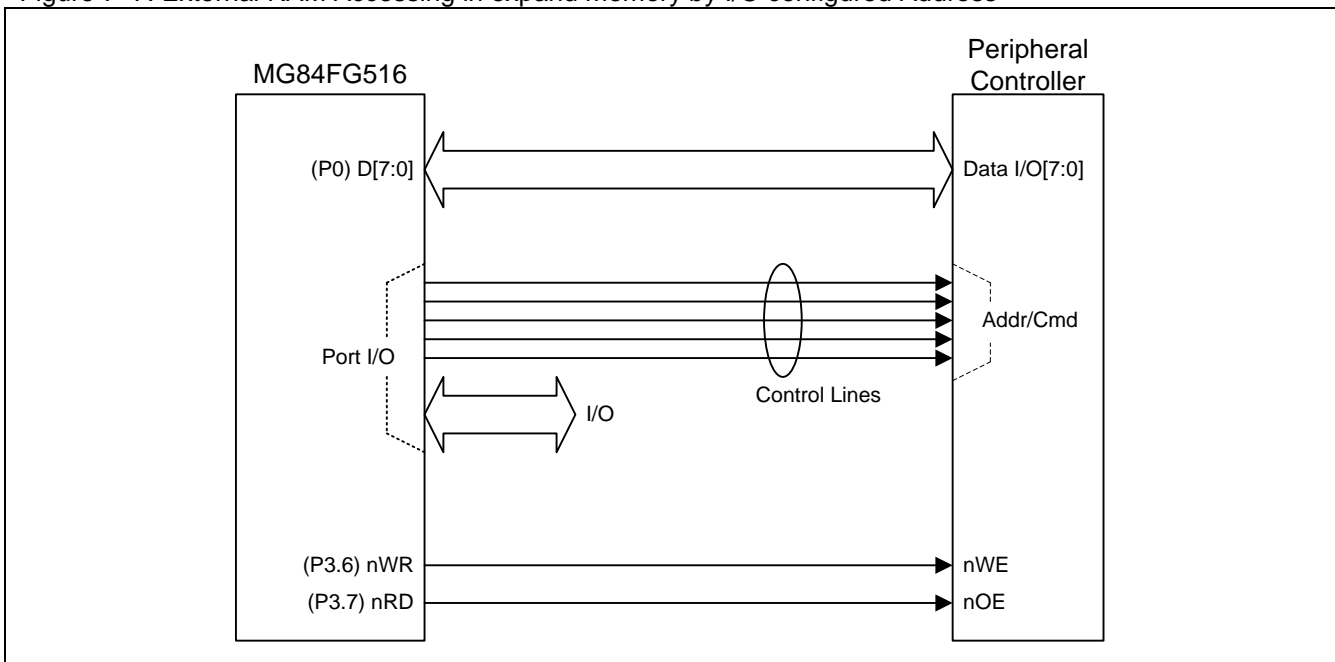


Figure 7-7. External RAM Accessing in expand memory by I/O configured Address



It also fits the FIFO Architecture Accessing as a NAND type flash application.

### 7.3. On-chip expanded RAM (XRAM)

To access the on-chip expanded RAM (XRAM), refer to [Figure 7-2](#), the **4906/5120** bytes of XRAM (0000H to **0FFFH/13FF**) are indirectly accessed by move external instruction, "MOVX @Ri" and "MOVX @DPTR". For KEIL-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 **MG84FG516** hardware can access them correctly.

If ENUSB (CKCON0.7) is cleared to disable USB function, the USB 1024 bytes buffer is moved to on-chip expanded RAM area which will provide **5120** bytes XRAM in **MG84FG516**. If ENUSB is set to enable USB function, **MG84FG516** just provides **4096** bytes XRAM in software application. In most application case, software does not spend all 1K bytes USB buffer, the unused USB buffer can provide MCU additional data RAM which is access through USB re-directed mechanism. Please refer Section "[23.4 Access USB 1K FIFO](#)" to get the detail information for the buffer access.

## 7.4. External Data Memory access

### AUXR1: Auxiliary Control Register 1

SFR Page = 0~F

SFR Address = 0xA2

RESET = 0000-0000

7	6	5	4	3	2	1	0
KBIPS1	KBIPS0	P5SPI	P5S1	P5T2	P6PCA	<b>EXTRAM</b>	DPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 1: EXTRAM, External data RAM enable.

0: Enable on-chip expanded data RAM (XRAM 4096/5120 bytes) and enable USB 64 bytes SFR on XRAM access.

1: Disable on-chip expanded data RAM and disable USB 64 bytes SFR on XRAM access.

### STRETCH: MOVX Stretch Register

SFR Page = 0~F

SFR Address = 0x8F

RESET = 0X00-0000

7	6	5	4	3	2	1	0
EMAI1	--	ALES1	ALES0	RWSH	RWS2	RWS1	RWS0
R/W	W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: EMAI1, EMAI1 configures the External data Memory Access Interface mode as following:

0: Multiplexed address/data.

1: No Address phase access

Bit 6: Reserved. Software must write "0" on this bit when STRETCH is written.

Bit 5~4: ALES[1:0], EMAI ALE pulse width select bits. It only has effect when EMAI in Multiplexed mode.

00: ALE high and ALE low pulse width = 1 SYSCLK cycle.

01: ALE high and ALE low pulse width = 2 SYSCLK cycle.

10: ALE high and ALE low pulse width = 3 SYSCLK cycle.

11: ALE high and ALE low pulse width = 4 SYSCLK cycle.

Bit 3: RWSH, EMAI Read/Write pulse Setup/Hold time control.

0: /RD and /WR command Setup/Hold Time = 1 SYSCLK cycle.

1: /RD and /WR command Setup/Hold Time = 2 SYSCLK cycle.

Bit 2~0: RWS[2:0], EMAI Read/Write command pulse width select bits.

000: /RD and /WR pulse width = 1 SYSCLK cycle.

001: /RD and /WR pulse width = 2 SYSCLK cycle.

010: /RD and /WR pulse width = 3 SYSCLK cycle.

011: /RD and /WR pulse width = 4 SYSCLK cycle.

100: /RD and /WR pulse width = 5 SYSCLK cycle.

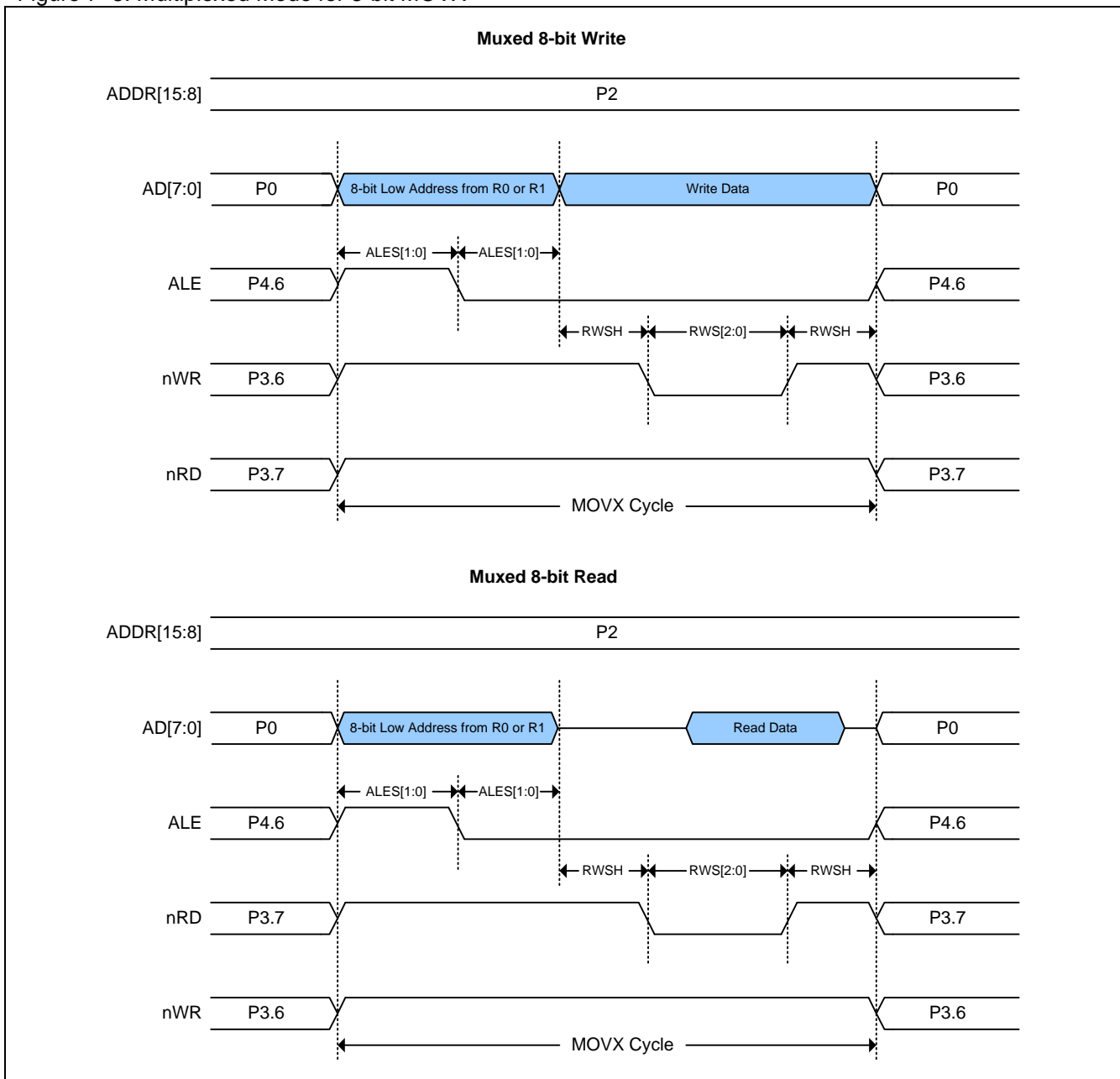
101: /RD and /WR pulse width = 6 SYSCLK cycle.

110: /RD and /WR pulse width = 7 SYSCLK cycle.

111: /RD and /WR pulse width = 8 SYSCLK cycle.

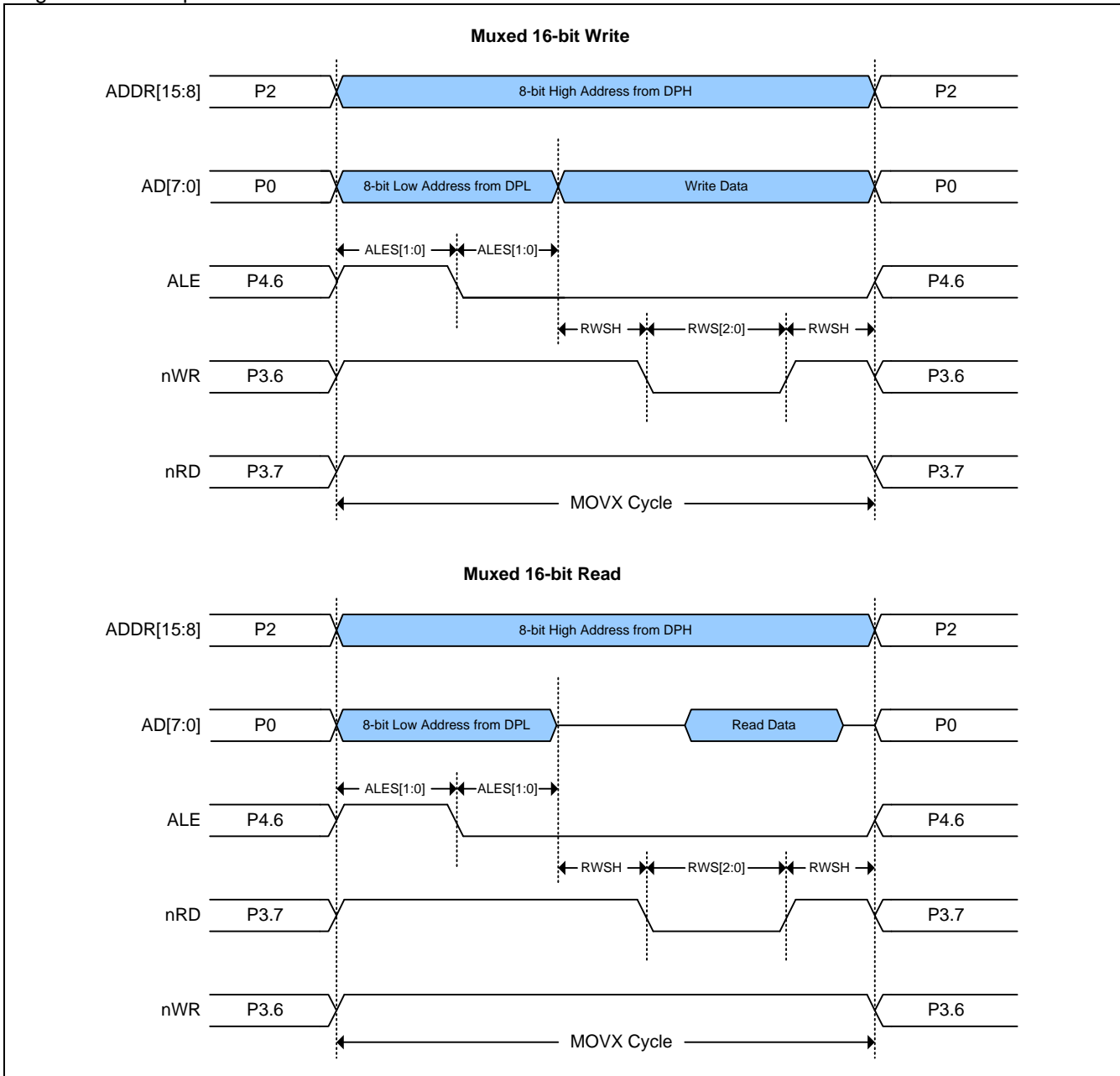
### 7.4.1. Multiplexed Mode for 8-bit MOVX

Figure 7–8. Multiplexed Mode for 8-bit MOVX



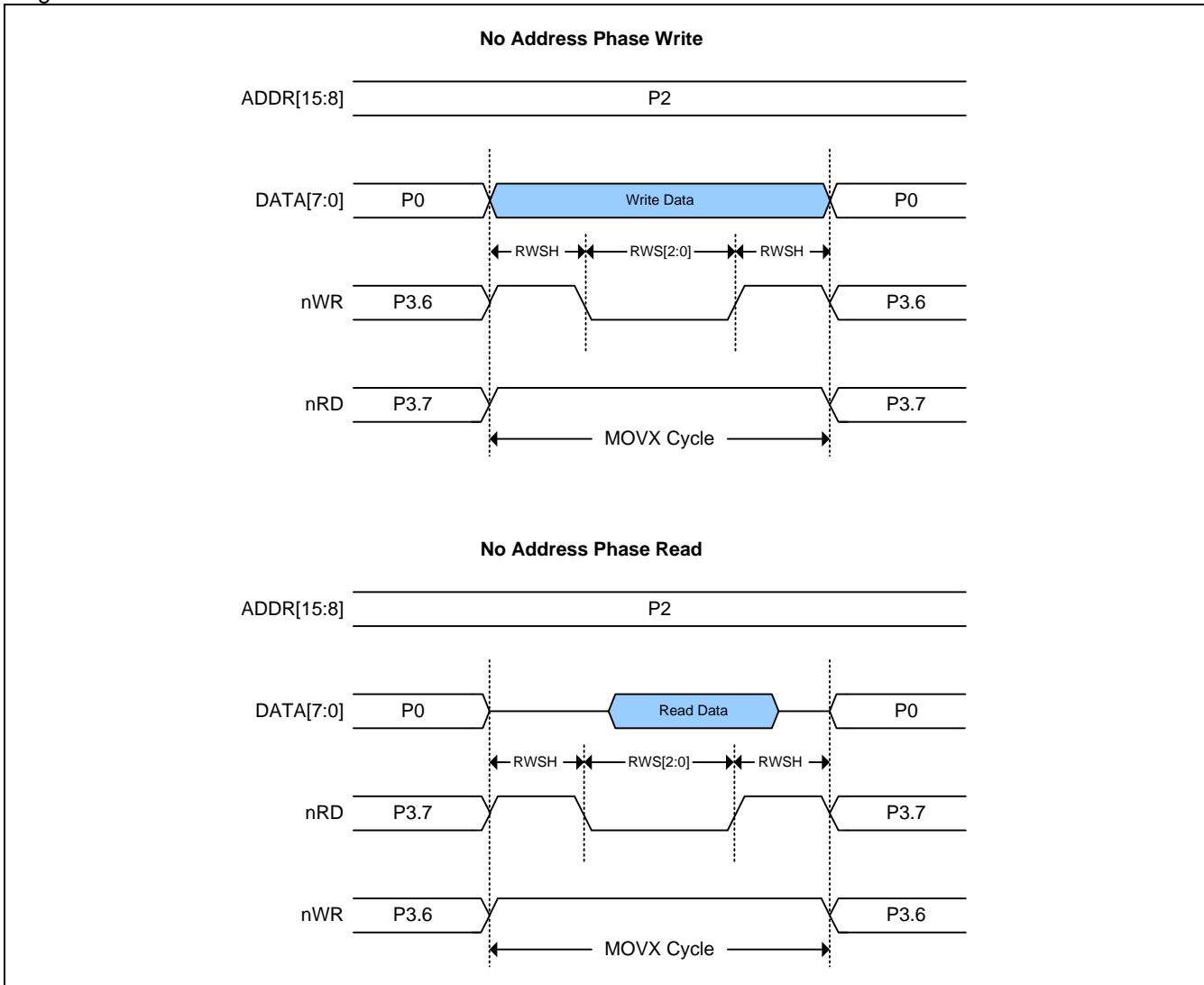
### 7.4.2. Multiplexed Mode for 16-bit MOVX

Figure 7–9. Multiplexed Mode for 16-bit MOVX



### 7.4.3. No Address Phase Mode for MOVX

Figure 7–10. No Address Phase Mode for MOVX





## 7.5. Declaration Identifiers in a C51-Compiler

The declaration identifiers in a C51-compiler for the various **MG84FG516** 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 **MG84FG516** has **4096/5120** 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 **MG84FG516** has 256 bytes of on-chip pdata memory which is shared with on-chip xdata memory.

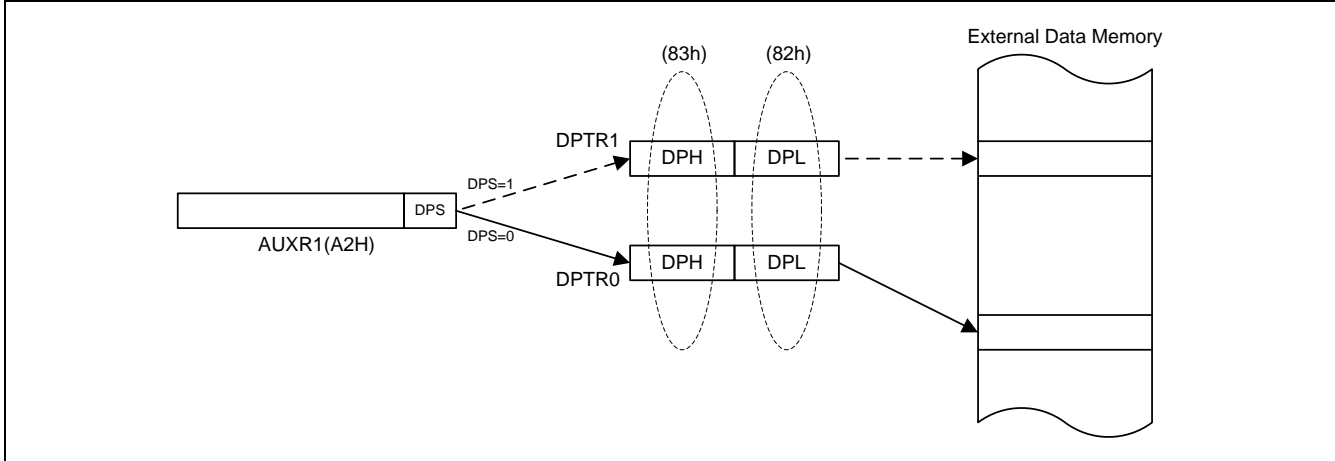
### ***code***

64K bytes of program memory space; accessed as part of program execution and via the "MOVC @A+DTPR" instruction. The **MG84FG516** has 64K bytes of on-chip code memory.

## 8. Dual Data Pointer Register (DPTR)

The dual DPTR structure as shown in Figure 8–1 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 8–1. Dual DPTR



### 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
KBIPS1	KBIPS0	P5SPI	P5S1	P5T2	P6PCA	EXTRAM	<b>DPS</b>
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 0: DPTR select bit, used to switch between DPTR0 and DPTR1.

0: Select DPTR0.

1: Select DPTR1.

DPS	Selected DPTR
0	DPTR0
1	DPTR1

## 9. System Clock

There are four clock sources for the system clock: Internal High-frequency RC Oscillator (IHRCO), external crystal oscillator, Internal Low-frequency RC Oscillator (ILRCO) and External Clock Input. Figure 9–1 shows the structure of the system clock in **MG84FG516**.

The **MG84FG516** always boots from IHRCO on 12MHz and reserves crystal pads as P6.0/P6.1 GPIO function. Software can select the one of the four clock sources by application required and switches them on the fly. But software needs to settle the clock source stably before clock switching. If software selects external crystal mode, port pin of P6.0 and P6.1 will be assigned to XTAL2 and XTAL1. And P6.0/P6.1 GPIO function will be inhibited. In external clock input mode (ECKI), the clock source comes from P6.0 input and P6.1 still reserves GPIO function.

The built-in IHRCO provides the high precision frequency at 12.0MHz with  $\pm 2\%$  deviation over all operating voltage and temperature. In IHRCO or ILRCO mode, P6.0 can be configured to internal *MCK* output or *MCK/2* and *MCK/4* for system application.

The **MG84FG516** device includes a Clock Multiplier to generate the high speed clock for system clock source. It generates 4/5.33/8 times frequency of CKMI, CKMI is shown in Figure 9–1 and its typical input is 6MHz. This function provides the high speed operation on MCU without external high-frequency crystal. To find the detailed CKM performance, please refer Section “29.6 CKM Characteristics”.

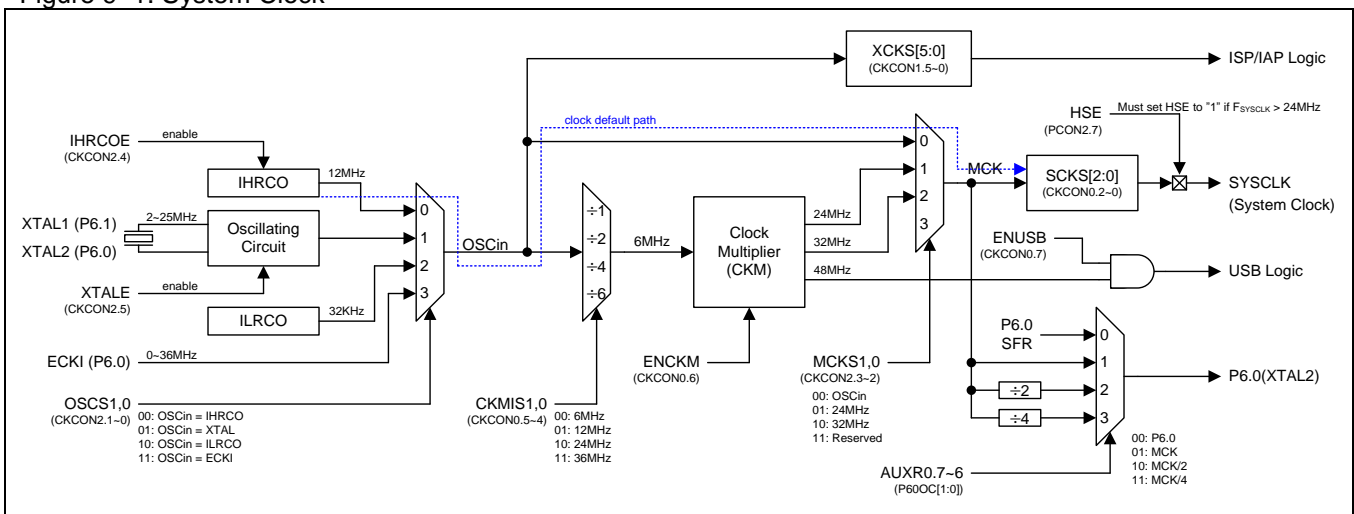
The built-in ILRCO provides the low power and low speed frequency about 32KHz to WDT and system clock source. MCU can select the ILRCO to system clock source by software for low power operation. To find the detailed IHRCO performance, please refer Section “29.5 ILRCO Characteristics”. In ILRCO mode, P6.0 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 four clock sources through the clock divider, as shown in Figure 9–1. The user can program the divider control bits *SCKS2*–*SCKS0* (in *CKCON0* register) to get the desired system clock.

### 9.1. Clock Structure

Figure 9–1 presents the principal clock systems in the **MG84FG516**. The system clock can be sourced by the external oscillator circuit or either internal oscillator.

Figure 9–1. System Clock



## 9.2. Clock Register

### CKCON0: Clock Control Register 0

SFR Page = 0-F & P

SFR Address = 0xC7

RESET = 0001-x000

7	6	5	4	3	2	1	0
ENUSB	ENCKM	CKMIS1	CKMIS0	-	SCKS2	SCKS1	SCKS0
R/W	R/W	R/W	R/W	W	R/W	R/W	R/W

Bit 7: ENUSB, Enable USB clock and whole USB function.

0: Disable USB clock and USB function.

1: Enable USB clock and USB function.

Bit 6: ENCKM, Enable clock multiplier (X8)

0: Disable the X8 clock multiplier.

1: Enable the X8 clock multiplier.

Bit 5~4: CKMIS1 ~ CKMIS0, Clock Multiplier Input Selection.

CKMIS[1:0]	Clock Multiplier Input Selection
0 0	6MHz input
0 1	12MHz input
1 0	24MHz input
1 1	36MHz input

Bit 3: Reserved. Software must write "0" on this bit when CKCON0 is written.

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

SCKS[2:0]	System Clock
0 0 0	MCK
0 0 1	MCK/2
0 1 0	MCK/4
0 1 1	MCK/8
1 0 0	MCK/16
1 0 1	MCK/32
1 1 0	MCK/64
1 1 1	MCK/128

### CKCON1: Clock Control Register 1

SFR Page = 0-F & P

SFR Address = 0xBF

RESET = xx00-1011

7	6	5	4	3	2	1	0
--	--	XCKS5	XCKS4	XCKS3	XCKS2	XCKS1	XCKS0
W	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 CKCON1 is written.

Bit 5~0: This is set the OSCin frequency value to define the time base of ISP/IAP programming. Fill with a proper value according to OSCin, as listed below.

**[XCKS5~XCKS0] = OSCin - 1, where OSCin=1~40 (MHz).**

For examples,

(1) If OSCin=12MHz, then fill [XCKS5~XCKS0] with 11, i.e., 00-1011B.

(2) If OSCin=6MHz, then fill [XCKS5~XCKS0] with 5, i.e., 00-0101B.

OSCin	XCKS[4:0]
1MHz	00-0000
2MHz	00-0001

3MHz	00-0010
4MHz	00-0011
.....	.....
.....	.....
38MHz	10-0101
39MHz	10-0110
40MHz	10-0111

The default value of XCKS= 00-1011 for OSCin= 12MHz.

**CKCON2: Clock Control Register 2**

SFR Page = P

SFR Address = 0x40

RESET = 0101-0000

7	6	5	4	3	2	1	0
XTGS1	XTGS0	XTALE	IHRCOE	MCKS1	MCKS0	OSCS1	OSCS0
W	W	W	W	W	W	W	W

Bit 7~6: XTGS1~XTGS0, XTAL oscillator Gain control Register. Software must writ "01" on the two bits.

Bit 5: XTALE, external Crystal(XTAL) Enable.

0: Disable XTAL oscillating circuit. In this case, XTAL2 and XTAL1 behave as Port 6.0 and Port 6.1.

1: Enable XTAL oscillating circuit. If this bit is set by CPU software, it needs **3 ms** to have stable output after XTALE enabled.

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: MCKS[1:0], MCK Source Selection.

MCKS[1:0]	MCK Source Selection
0 0	OSCin
0 1	24MHz (ENCKM must be enabled)
1 0	32MHz (ENCKM must be enabled)
1 1	Reserved for 48MHz

Bit 1~0: OSC[1:0], OSCin source selection.

CKMIS[1:0]	OSCin source Selection
0 0	IHRCO
0 1	XTAL
1 0	ILRCO
1 1	ECKI, External Clock Input (P6.0) as OSCin.

**AUXR0: Auxiliary Register 0**

SFR Page = 0~F

SFR Address = 0xA1

RESET = 000x-0000

7	6	5	4	3	2	1	0
P60OC1	P60OC0	P60FD	--	P4FS1	P4FS0	INT1H	INT0H
R/W	R/W	R/W	W	R/W	R/W	R/W	R/W

Bit 7~6: P6.0 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 crystal mode, XTAL2 and XTAL1 are the alternated function of P6.0 and P6.1. In external clock input mode, P6.0 is the dedicated clock input pin. In internal oscillator condition, P6.0 provides the following selections for GPIO or clock source generator. When P60OC[1:0] index to non-P6.0 GPIO function, P6.0 will drive the on-chip RC oscillator (IHRCO or ILRCO) output to provide the clock source for other devices.

P60OC[1:0]	P6.0 function	P6.0 I/O mode
------------	---------------	---------------

00	P60	By P6M0.0
01	MCK	By P6M0.0
10	MCK/2	By P6M0.0
11	MCK/4	By P6M0.0

For clock-out on P6.0 function, it is recommended to set P6M0.0 to "1" which selects P6.0 as push-push output mode.

Bit 5: P60FD, P6.0 Fast Driving.

0: P6.0 output with default driving.

1: P6.0 output with fast driving enabled. If P6.0 is configured to clock output, enable this bit when P6.0 output frequency is more than 12MHz at 5V application or more than 6MHz at 3V application.

**PCON2: Power Control Register 2**

SFR Page = P Only

SFR Address = 0x44

POR = 0000-0101

7	6	5	4	3	2	1	0
HSE	IAPO	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	1
W	W	W	W	W	W	W	W

Bit 7: HSE, High Speed operation Enable.

0: Disable high speed operation for MCU.

1: Enable high speed operation for MCU (SYSCLK > 24MHz). Before select high frequency clock (>24MHz) on SYSCLK, software must set HSE to switch internal circuit for high speed operation. It may cause more power consumption on chip operation.

### 9.3. Clock Sample Code

(1). Required Function: Select XTAL as OSCin source when MCU using IHRCO or ILRCO (default is IHRCO)

Assembly Code Example:

```

MOV    IFADRL,#(CKCON2)      ; Index Page-P address to CKCON2
CALL   _page_p_sfr_read      ; Read CKCON2 data

ORL    IFD,#( XTALE)         ; Enable XTALE
                                ; For XTAL,
CALL   _page_p_sfr_write     ; Write data to CKCON2, SYSCLK must be less than 25MHz

CALL   Delay_10ms           ;delay 10ms Wait XTAL stable

ANL    IFD,#~(OSCS1 | OSCS0) ; Switch OSCin source to XTAL.
ORL    IFD,#(OSCS0)
CALL   _page_p_sfr_write     ; Write data to CKCON2

ANL    IFD,#~(IHRCOE)        ; Disable IHRCO if MCU is switched from IHRCO
CALL   _page_p_sfr_write     ; Write data to CKCON2

```

C Code Example:

```

IFADRL = CKCON2;           // Index Page-P address to CKCON2
page_p_sfr_read();        // Read CKCON2 data

IFD |= XTALE;              // Enable XTALE
                                // For XTAL,
page_p_sfr_write ();      // Write data to CKCON2, SYSCLK must be less than 25MHz

Dealy_10mS();             // ;delay 10ms Wait XTAL stable

IFD &= ~(OSCS1 | OSCS0);   // Switch OSCin source to XTAL.
IFD |= OSCS0;
page_p_sfr_write ();      // Write data to CKCON2

IFD &= ~IHRCOE;           // Disable IHRCO if MCU is switched from IHRCO
page_p_sfr_write();       // Write data to CKCON2

```

(2). Required Function: Select ILRCO as OSCin source when MCU using IHRCO, ECKI or XTAL (default is IHRCO)

Assembly Code Example:

```

MOV    IFADRL,#(CKCON2)      ; Index Page-P address to CKCON2
CALL   _page_p_sfr_read      ; Read CKCON2 data

ANL    IFD,#~(OSCS1 | OSCS0) ; Switch OSCin source to ILRCO
ORL    IFD,#(OSCS1)
CALL   _page_p_sfr_write     ; Write data to CKCON2

ANL    IFD,#~(XTALE | IHRCOE) ; Disable XTAL and IHRCO
CALL   _page_p_sfr_write     ; Write data to CKCON2

MOV    IFADRL,#(PCON2)      ; Index Page-P address to PCON2
CALL   _page_p_sfr_read      ; Read PCON2 data

ANL    IFD,#~(HSE)          ; Disable HSE when SYSCLK ≤ 6MHz for power saving
CALL   _page_p_sfr_write     ; Write data to PCON2

```

C Code Example:

```

IFADRL = CKCON2;           // Index Page-P address to CKCON2
page_p_sfr_read();        // Read CKCON2 data.

IFD = ~(OSCS1 | OSCS0);   // Switch OSCin source to ILRCO
IFD |= OSCS1;

```

```

page_p_sfr_write();           // Write data to CKCON2

IFD &= ~(XTALE | IHRCOE);    // Disable XTAL and IHRCO
page_p_sfr_write();          // Write data to CKCON2

IFADRL = PCON2;              // Index Page-P address to PCON2
page_p_sfr_read();           // Read PCON2 data

IFD &= ~HSE;                  // Disable HSE when SYSCLK ≤ 6MHz for power saving
page_p_sfr_write();          // Write data to PCON2

```

**(3). Required Function: Select 32M as OSCin source when MCU using CKM (default is IHRCO)**

Assembly Code Example:

```

MOV    CKCON0,#(ENCKM)      ; Enable CKM
CALL   Delay_3mS;           ;delay 3mS

MOV    IFADRL,#(PCON2)     ; Index Page-P address to PCON2
CALL   _page_p_sfr_read    ; Read PCON2 data

ORL    IFD,#(HSE)           ; Set HSE =1, Due to SYSCLK=32M more than 25MHz (HSE=1)
CALL   _page_p_sfr_write   ; Write data to PCON2,

MOV    IFADRL,#(CKCON2)    ; Index Page-P address to CKCON2
CALL   _page_p_sfr_read    ; Read CKCON2 data

ORL    IFD,#(MCKS1)        ; Switch MCK source = 32M
CALL   _page_p_sfr_write   ; Write data to CKCON2,

```

C Code Example:

```

CKCON0 |= ENCKM;           //Enable CKM
Dealy_3mS();

IFADRL = PCON2;           // Index Page-P address to PCON2
page_p_sfr_read();        // Read PCON2 data.

IFD |= HSE ;              // Set HSE=1, SYSCLK=32M
page_p_sfr_write ();      // Write data to PCON2, SYSCLK must be less than 25MHz(HSE=1)

IFADRL = CKCON2;         // Index Page-P address to CKCON2
page_p_sfr_read();        // Read CKCON2 data.

IFD |= MCKS1 ;           // Switch MCK source to 32M
page_p_sfr_write ();      // Write data to CKCON2, SYSCLK more than 25MHz

```

**(4). Required Function: Select IHRCO as OSCin source when MCU using ILRCO, ECKI or XTAL**

Assembly Code Example:

```

MOV    IFADRL,#(CKCON2)    ; Index Page-P address to CKCON2
CALL   _page_p_sfr_read    ; Read CKCON2 data

ORL    IFD,#(IHRCOE)       ; Enable IHRCO
CALL   _page_p_sfr_write   ; Write data to CKCON2

Delay_32us

ANL    IFD,#~(OSCS1 | OSCS0) ; Switch OSCin source to IHRCO
CALL   _page_p_sfr_write   ; Write data to CKCON2

```



**C Code Example:**

```

IFADRL = CKCON2;           // Index Page-P address to CKCON2
page_p_sfr_read();        // Read CKCON2 data.

IFD |= IHRCOE;            // Enable IHRCO
page_p_sfr_write();       // Write data to CKCON2

Delay 32us

IFD &= ~(OSCS1 | OSCS0);  // Switch OSCin source to IHRCO
page_p_sfr_write();       // Write data to CKCON2

```

**(5). Required Function: Output IHRCO frequency on P6.0****Assembly Code Example:**

```

MOV    SFRPI,#01h          ; Set SFRPI=1
MOV    P6M0,#P6M0         ; Set P6.0 to push-pull output mode
MOV    SFRPI,#0h          ; Set SFRPI=0
ANL    AUXR0,#~(P60OC1|P60OC0) ; Switch P6.0 to GPIO function
ORL    AUXR0,#(P60OC0|P6FD) ; P6.0 = IHRCO Frequency + Pin fast driving
                                ; P60OC[1:0] | P6.0
                                ; 00      | GPIO
                                ; 01      | IHRCO/1
                                ; 10      | IHRCO/2
                                ; 11      | IHRCO/4

```

**C Code Example:**

```

SFRPI = 1;                // SFRPI=1.
P6M0 |= P6M0;             // P6.0 select push-pull output mode.
SFRPI = 0;                // SFRPI=0.
AUXR0 &= ~(P60OC0 | P60OC1); // Switch P6.0 to GPIO function
AUXR0 |= (P60OC0 | P6FD);  // P6.0 output IHRCO/1
// AUXR0 = P60OC1|P6FD;    // P6.0 output IHRCO/2
// AUXR0 = P60OC1|P60OC0|P6FD; // P6.0 output IHRCO/4

```

## 10. Watch Dog Timer (WDT)

### 10.1. 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 a 9-bit free-running counter, a 7-bit prescaler and a control register (WDTCR). Figure 10–1 shows the WDT structure in **MG84FG516**.

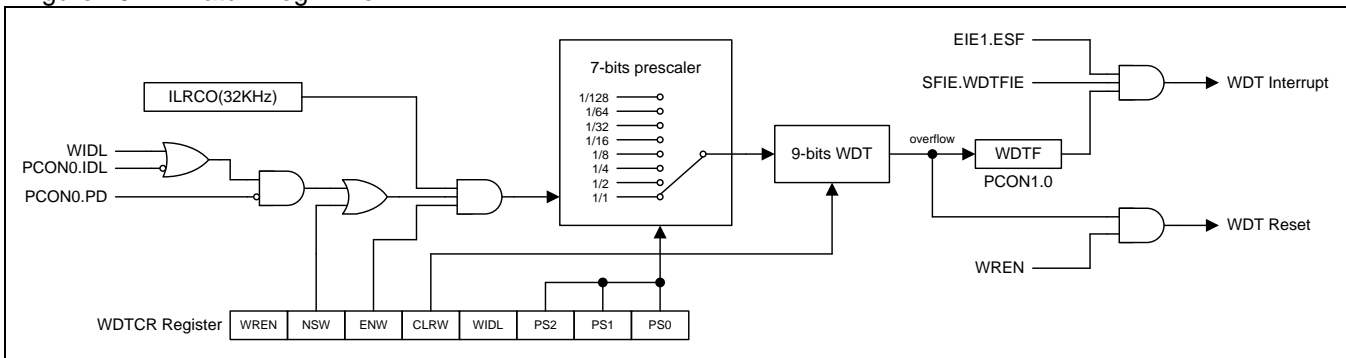
When WDT is enabled, it derives its time base from the 32KHz ILRCO. The WDT overflow will set the WDTF on PCON1.0 which can be configured to generate an interrupt by enabled WDTFIE (SFIE.0) and enabled ESF (EIE1.3). The overflow can also trigger a system reset when WREN (WDTCR.7) is set. To prevent WDT overflow, software needs to clear it by writing “1” to the CLRW bit (WDTCR.4) before WDT overflows.

*Note: the WDTFIE function is under verifying.*

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 “10.3 WDT Register” and Section “25 Page P SFR Access” for more detail information.

Figure 10–1. Watch Dog Timer



### 10.2. WDT During Idle and Power Down

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.

In the Power down mode, the ILRCO won't stop if the NSW (WDTCR.6) is enabled. The MUC enters Watch mode. 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.

### 10.3. 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 9-bit WDT counter to 000H. 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.

PS[2:0]	Prescaler Value	WDT Period
0 0 0	1	15 ms
0 0 1	2	31 ms
0 1 0	4	62 ms
0 1 1	8	124 ms
1 0 0	16	248 ms
1 0 1	32	496 ms
1 1 0	64	992 ms
1 1 1	128	1.984 S

#### PCON1: Power Control Register 1

SFR Page = 0~F & P

SFR Address = 0x97

POR = 00xx-x000

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

## 10.4. 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.

## 10.5. WDT Sample Code

(1) Required function: Enable WDT and select WDT period to 248ms

Assembly Code Example:		
ORL	PCON1,#(WDTF)	; Clear WDTF flag (write "1")
MOV	WDTCR,#(ENW   CLRW   PS2)	; Enable WDT counter and set WDT period to 248ms
C Code Example:		
PCON1  = WDTF;		// Clear WDT flag (write "1")
WDTCR = (ENW   CLRW   PS2);		// Enable WDT counter and set WDT period to 248ms
		// PS[2:0]   WDT period selection
		// 0   15ms
		// 1   31ms
		// 2   62ms
		// 3   124ms
		// 4   248ms
		// 5   496ms
		// 6   992ms
		// 7   1.984s

(2) Required function: How to Disable WDT

Assembly Code Example:		
MOV	IFD,WDTCR	; Read WDTCR data
ANL	IFD,#~(ENW)	; Clear ENW to disable WDT
MOV	IFADRL,#(WDTCR_P)	; Index Page-P address to WDTCR_P
CALL	_page_p_sfr_write	; Write data to WDTCR
C Code Example:		
IFD = WDTCR;		// Read WDTCR data
IFD &= ~ENW;		// Clear ENW to disable WDT
IFADRL = WDTCR_P;		// Index Page-P address to WDTCR_P
page_p_sfr_write();		// Write data to WDTCR

(3). Required Function: Enable WDT reset function and select WDT period to 62ms

Assembly Code Example:		
ORL	PCON1,#(WDTF)	; Clear WDTF flag (write "1")
MOV	WDTCR,#(WREN   CLRW   PS1)	; Enable WDT reset function and set WDT period to 62ms
ORL	WDTCR,#(ENW)	; Enable WDT counter, WDT running
C Code Example:		
PCON1  = WDTF;		// Clear WDTF flag (write "1")
WDTCR = WREN   CLRW   PS1;		// Enable WDT reset function and set WDT period to 62ms
WDTCR  = ENW;		// Enable WDT counter, WDT running.

(4). Required Function: Enable protected write for WDTCR

Assembly Code Example:

```
ORL    PCON1,#(WDTF)           ; Clear WDTF flag (write "1")
MOV    WDTCR,#(ENW | CLRW | PS2) ; Enable WDT counter and set WDT period to 248ms

MOV    IFADRL,#(SPCON0)        ; Index Page-P address to SPCON0
CALL   _page_p_sfr_read        ; Read SPCON0 data

ORL    IFD,#(WRCTL)           ; Enable protected write for WDTCR
CALL   _page_p_sfr_write       ; Write data to SPCON0

MOV    IFD,WDTCR               ; Read WDTCR data
ORL    IFD,#(CLRW)            ; Enable CLRW

MOV    IFADRL,#(WDTCR_P)      ; Index Page-P address to WDTCR_P
CALL   _page_p_sfr_write       ; Write data to WDTCR to clear WDT counter
```

C Code Example:

```
PCON1 |= WDTF;                // Clear WDTF flag (write "1")
WDTCR = ENW | CLRW | PS2;     // Enable WDT counter and set WDT period to 248ms

IFADRL = SPCON0;              // Index Page-P address to SPCON0
page_p_sfr_read();           // Read SPCON0 data

IFD |= WRCTL;                 // Enable protected write for WDTCR
page_p_sfr_write();          // Write data to SPCON0

IFD = WDTCR;                  // Read WDTCR data
IFD |= CLRW;                  // Enable CLRW

IFADRL = WDTCR_P;            // Index Page-P address to WDTCR_P
page_p_sfr_write();          // Write data to WDTCR to clear WDT counter
```

# 11. System Reset

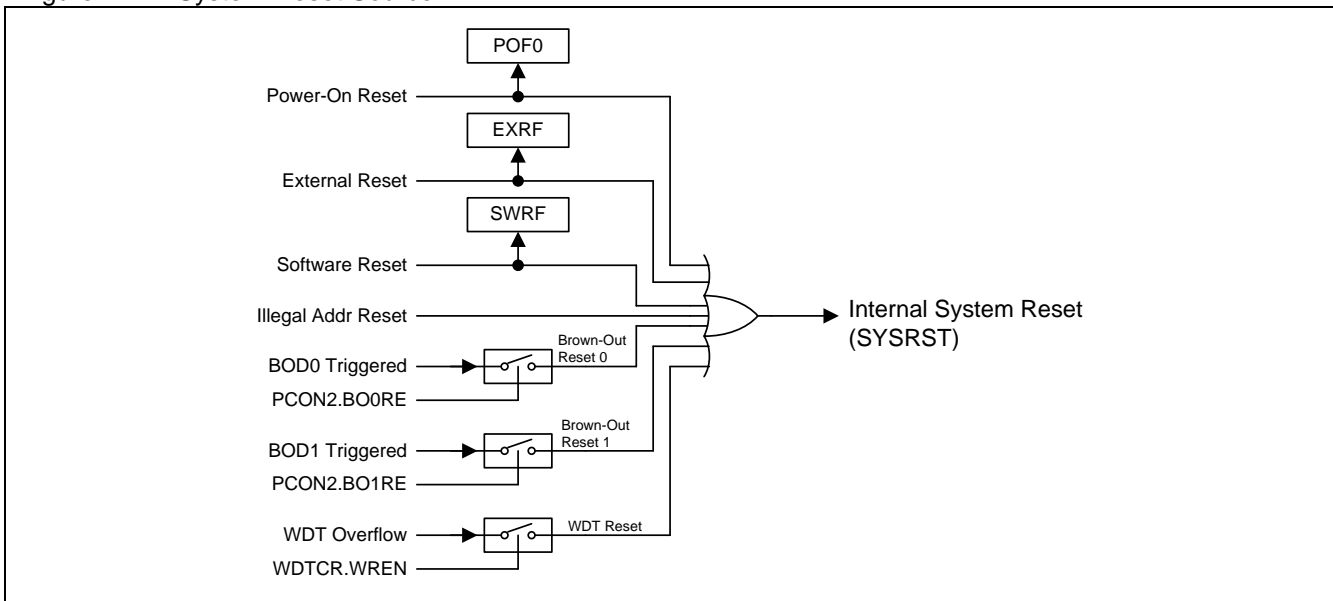
During reset, all I/O Registers are set to their initial values, and the program starts execution from the Reset Vector, 0000H, or ISP start address by OR setting. The **MG84FG516** has seven sources of reset: power-on reset, external reset, software reset, illegal address reset, brown-out reset 0, brown-out reset 1 and WDT reset. **Figure 11–1** shows the system reset source in **MG84FG516**.

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 **MG84FG516** 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

SFR Address = 0x87

POR = 0001-0000, RESET = 000X-0000

7	6	5	4	3	2	1	0
SMOD1	SMOD0	GF	<b>POF0</b>	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 (ISP.CR.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 = 00xx-x000

7	6	5	4	3	2	1	0
SWRF	EXRF	--	--	--	BOF1	BOF0	WDTF
R/W	R/W	W	W	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+RESET = 0000-0XXX

7	6	5	4	3	2	1	0
ISPEN	SWBS	SWRST	CFAIL	MISPF	--	--	--
R/W	R/W	R/W	R/W	R/W	W	W	W

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 = 00xx-x000

7	6	5	4	3	2	1	0
SWRF	EXRF	--	--	--	BOF1	BOF0	WDTF
R/W	R/W	W	W	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. Illegal Address Reset

In **MG84FG516**, if software program runs to illegal address such as over program ROM limitation, it triggers a RESET to CPU.

### 11.6. Brown-Out Reset

In **MG84FG516**, there are two Brown-Out Detectors (BOD0 & BOD1) to monitor VDD power. BOD0 services the fixed detection level at VDD=2.2V. BOD1 detects the VDD level by software selecting 4.2V, 3.7V, 2.4V or 2.0V. 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.



**PCON1: Power Control Register 1**

SFR Page = 0~F & P

SFR Address = 0x97 POR = 00xx-x000

7	6	5	4	3	2	1	0
SWRF	EXRF	--	--	--	<b>BOF1</b>	<b>BOF0</b>	<b>WDTF</b>
R/W	R/W	W	W	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.

**11.7. 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 = 00xx-x000

7	6	5	4	3	2	1	0
SWRF	EXRF	--	--	--	BOF1	BOF0	<b>WDTF</b>
R/W	R/W	W	W	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.

## 11.8. Reset Sample Code

(1) Required function: Trigger a software reset

Assembly Code Example:	
ORL	ISPCR,#SWRST ; Trigger Software Reset
C Code Example:	
ISPCR  = SWRST;	// Trigger Software Reset

(2) Required Function: Enable BOD0 reset

Assembly Code Example:	
MOV	IFADRL,#PCON2 ; Index Page-P address to PCON2
CALL	_page_p_sfr_read ; Read PCON2 data
ORL	IFD,#BO0RE ; Enable BOD0 reset function
CALL	_page_p_sfr_write ; Write data to PCON2
C Code Example:	
IFADRL = PCON2;	// Index Page-P address to PCON2
page_p_sfr_read();	// Read PCON2 data
IFD  = BO0RE;	// Enable BOD0 reset function
page_p_sfr_write();	// Write data to PCON2

## 12. Power Management

The **MG84FG516** supports two power monitor modules, Brown-Out Detector 0 (BOD0) and Brown-Out Detector 1 (BOD1), and 6 power-reducing modes: Idle mode, Power-down mode, Slow mode, Sub-Clock mode, Watch mode and Monitor mode.

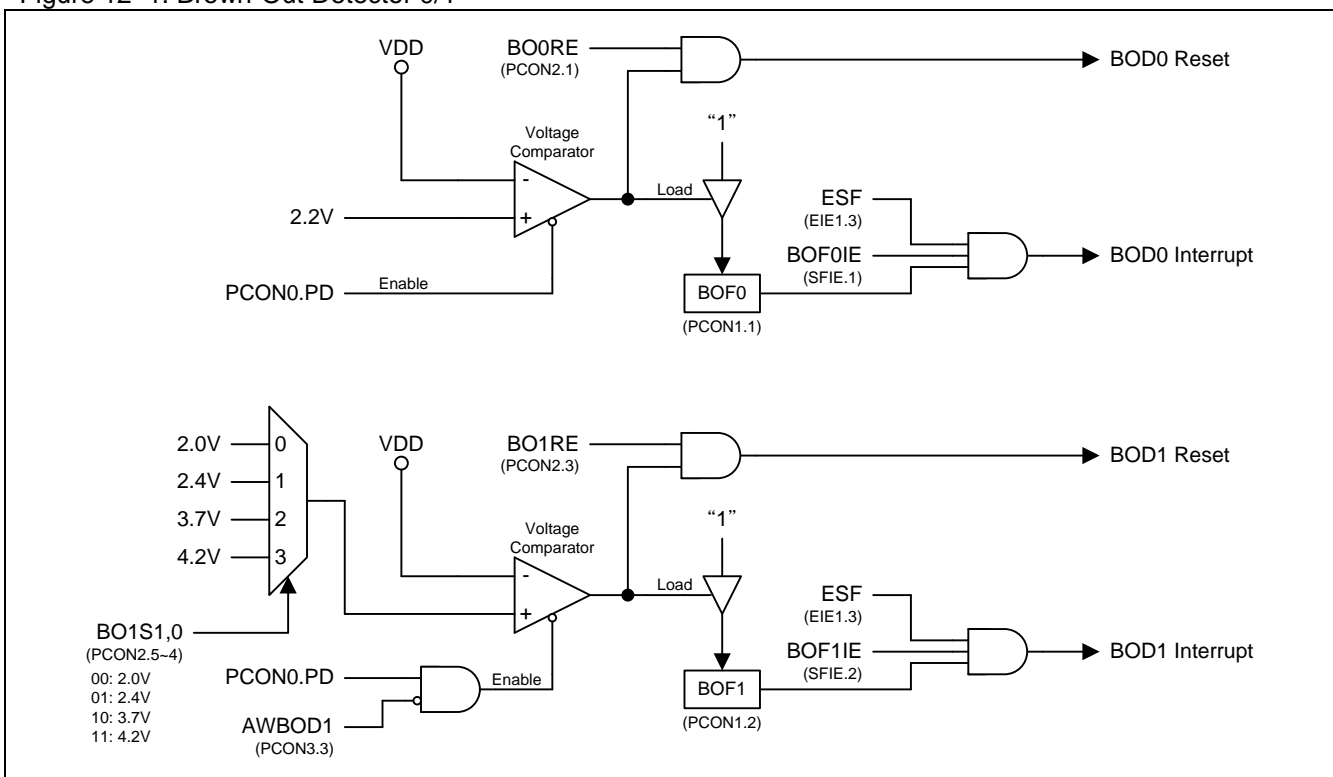
BOD0 and BOD1 report the chip power status on the flags, BOF0 and BOF1, which provide the capability to interrupt CPU or to reset CPU by software configured. The six power-reducing modes provide the different power-saving scheme for chip application. These modes are accessed through the CKCON0, CKCON2,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

### 12.1. Brown-Out Detector

In **MG84FG516**, 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=2.2V and BOD1 detects the software selection levels (4.2V/3.7V/2.4V/2.0V) 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 (PCON3.3) 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 (PCON3.3) is enabled in BOD1 reset operation.

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 "9 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 operate in an ultra low speed and low power operation. Additional programming SCKS2~SCKS0 bits (in CKCON0 register, see Section "9 System Clock"), the user could put the MCU speed down to 250Hz slowest.

### 12.2.3. Watch Mode

If Watch-Dog-Timer is enabled and NSW is set, Watch-Dog-Timer will keep running in power down mode, which named Watch Mode in **MG84FG516**. 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 "10 Watch Dog Timer (WDT)" and Section "14 Interrupt" for more detail information.

### 12.2.4. Monitor Mode

If AWBOD1 (PCON3.3) is set, BOD1 will keep VDD monitor in power down mode. It is the Monitor Mode in **MG84FG516**. 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.

### 12.2.5. 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, Timer 3, SPI, KBI, ADC, UART0, UART1, TWSI, USB, 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 input channels must be set to "Analog Input Only" in P1AIO SFR when MCU is in idle mode or power-down mode.

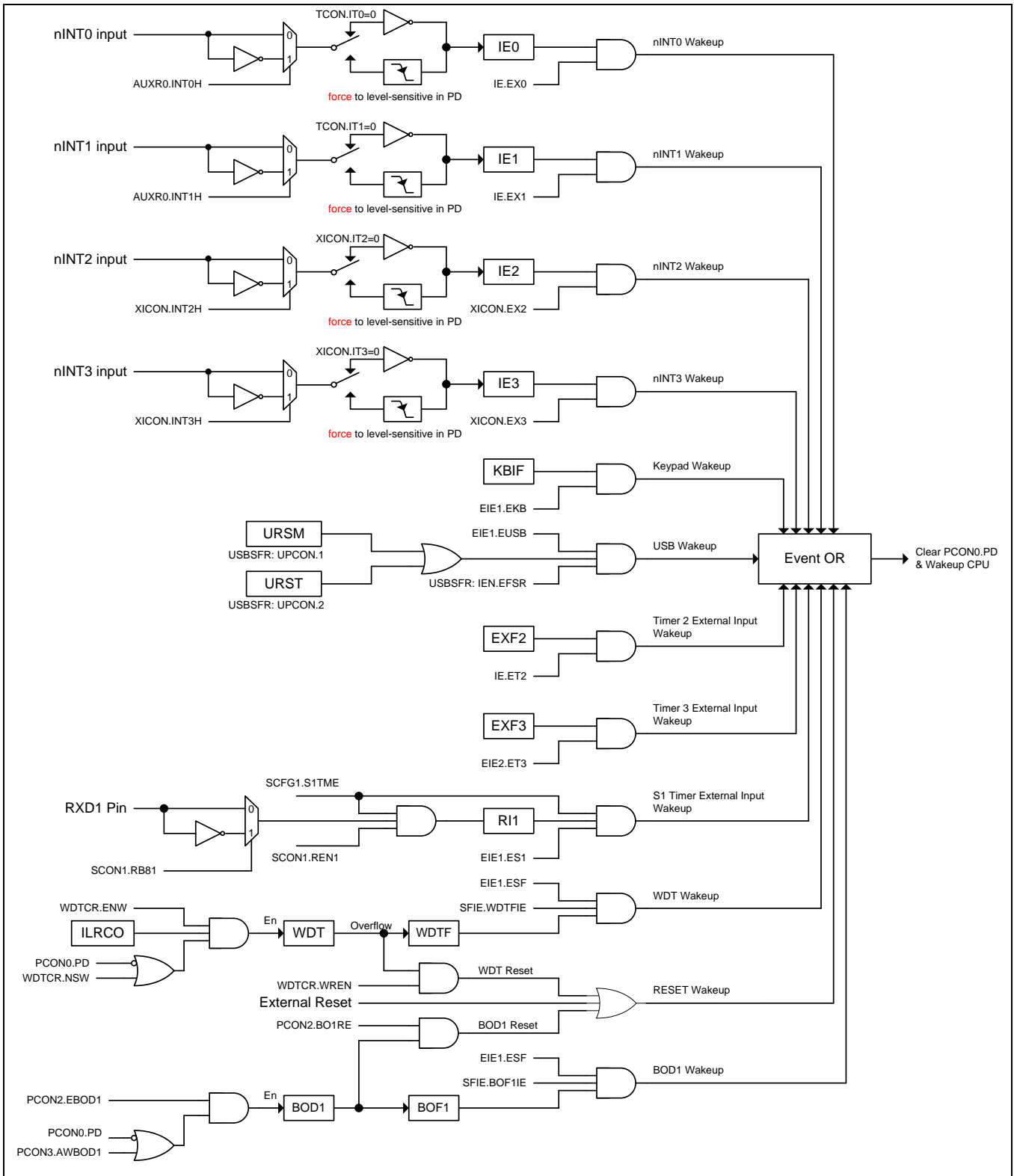
### 12.2.6. Power-down Mode

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, power-on reset, enabled external interrupts, enabled KBI, enabled USB, enabled BOD1 or enabled Non-Stop WDT.

The user should not attempt to enter (or re-enter) the power-down mode for a minimum of 4  $\mu$ s 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, including the port I/Os un-appearance on package pins. For example, P5.7~P5.0 and P6.7~P6.2 are un-appearance in MG84FG516AD48 (LQFP48) package pins. Software may configure P5/P6 corresponding bit SFR to "0" (output low) to avoid pin floating in power-down mode.

Figure 12–2 shows the wakeup mechanism of power-down mode in **MG84FG516**.

Figure 12–2. Wakeup structure of Power Down mode



### 12.2.7. Interrupt Recovery from Power-down

Four external interrupts may be configured to terminate Power-down mode. External interrupts nINT0 (P3.2), nINT1 (P3.3), nINT2 (P4.3) and nINT3 (P4.2) may be used to exit Power-down. To wake up by external interrupt nINT0, nINT1, nINT2 or nINT3, 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.

### 12.2.8. 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.

### 12.2.9. KBI wakeup Recovery from Power-down

The Keypad Interrupt of **MG84FG516**, P0.7 ~ P0.0 have wakeup CPU capability that are enabled by the control registers in KBI module. The KBI function can be set on Port 2, Port 5 or Port 6 by software configured.

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.

### 12.2.10. USB wakeup Recovery from Power-down

If USB function is enabled and USB host is connected to **MG84FG516**, USB host reset and resume event will wakeup CPU from power down mode. Wakeup from Power-down through enabled USB function (CKCON0.7) and enabled USB interrupt (IEN.2 in USB SFR) is same to the interrupt. At the active USB interrupt, 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

SFR Address = 0x87

POR = 0001-0000, RESET = 000X-0000

7	6	5	4	3	2	1	0
SMOD1	SMOD0	--	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 = 00xx-x000

7	6	5	4	3	2	1	0
SWRF	EXRF	--	--	--	BOF1	BOF0	WDTF
R/W	R/W	W	W	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 5~3: Reserved. Software must write "0" on these bits when PCON1 is written.

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/3.7/2.4/2.0).

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 (2.2V).

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
HSE	IAP0	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	1
W	W	W	W	W	W	W	W

Bit 5-4: BO1S[1:0]. Brown-Out detector 1 monitored level Selection.

BO1S[1:0]	BOD1 detecting level
0 0	2.0V
0 1	2.4V
1 0	3.7V
1 1	4.2V

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 2.2V).

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 = xxxx-00x1

7	6	5	4	3	2	1	0
0	0	0	0	AWBOD1	0	0	OCDE
W	W	W	W	W	W	W	R/W

Bit 7-4: Reserved. Software must write "0" on these bits when PCON3 is written.

Bit 3: AWBOD1, Awaked BOD1 in PD mode.

- 0: BOD1 is disabled in power-down mode.
- 1: BOD1 keeps operation in power-down mode.

Bit 2-1: Reserved. Software must write "0" on these bits when PCON3 is written.

Bit 0: OCDE, OCD enable. The initial value is loaded from OR and reset by POR.

- 0: Disable OCD interface on P4.4 and P4.5
- 1: Enable OCD interface on P4.4 and P4.5.



## 12.4. Power Control Sample Code

(1) Required function: Select Slow mode with OSCin/128 (default is OSCin)

Assembly Code Example:

```

ORL    CKCON0,#(SCKSO | SCKS1 | SCKS2)    ; OSCin/128

MOV    IFADRL,#PCON2                      ; Index Page-P address to PCON2
CALL   _page_p_sfr_read                   ; Read PCON2 data

ANL    IFD,#~(HSE)                        ; Disable HSE when SYSCLK ≤ 6MHz for power saving
CALL   _page_p_sfr_write                  ; Write data to PCON2

```

C Code Example:

```

CKCON0 |= (SCKS2 | SCKS1 | SCKS0);      // Select system clock divider to OSCin/128.

IFADRL = PCON2;                          // Index Page-P address to PCON2
page_p_sfr_read();                       // Read PCON2 data.

IFD &= ~HSE;                             // Disable HSE when SYSCLK ≤ 6MHz for power saving
page_p_sfr_write();                      // Write data to PCON2

```

(2) Required function: Select Sub-Clock mode with OSCin ( OSCin=32KHz)

Assembly Code Example:

```

MOV    IFADRL,#CKCON2                      ; Index Page-P address to CKCON2
CALL   _page_p_sfr_read                   ; Read CKCON2 data

ANL    IFD,#~(OSCS1|OSCS0)                ; Switch OSCin source to ILRCO
ORL    IFD,#OSCS1
CALL   _page_p_sfr_write                  ; Write data to CKCON2

ANL    IFD,#~(IHRCOE|XTALE)               ; Disable IHRCO & XTAL
CALL   _page_p_sfr_write                  ; Write data to CKCON2

MOV    IFADRL,#PCON2                      ; Index Page-P address to PCON2
CALL   _page_p_sfr_read                   ; Read PCON2 data

ANL    IFD,#~(HSE)                        ; Disable HSE when SYSCLK ≤ 6MHz for power saving
CALL   _page_p_sfr_write                  ; Write data to PCON2

MOV    A,CKCON0                            ; Select system clock = OSCin/1
ANL    A,#~(SCKS2|SCKS1|SCKS0)
ORL    A,#SCKSO
MOV    CKCON0,A

```

C Code Example:

```

IFADRL = CKCON2;                          // Index Page-P address to CKCON2
page_p_sfr_read();                       // Read CKCON2 data

IFD &= ~(OSCS1 | OSCS0);                  // Switch OSCin source to ILRCO
IFD |= OSCS1;
page_p_sfr_write();                      // Write data to CKCON2

IFD = IFD & ~(IHRCOE|XTALE);             // Disable IHRCO & XTAL
page_p_sfr_write();                      // Write data to CKCON2

IFADRL = PCON2;                          // Index Page-P address to PCON2
page_p_sfr_read();                       // Read PCON2 data

IFD = IFD & ~(HSE);                      // Disable HSE when SYSCLK ≤ 6MHz for power saving
page_p_sfr_write();                      // Write data to PCON2

```

```

ACC = CKCON0; // Select system clock = OSCin
ACC &= ~(SCK2 | SCK1 | SCK0);
ACC |= SCK0;
CKCON0 = ACC;

```

**(3). Required Function: Switch MCU running with 32.768KHz XTAL mode**

**Assembly Code Example:**

```

MOV    IFADRL,#CKCON2           ; Index Page-P address to CKCON2
CALL   _page_p_sfr_read         ; Read CKCON2 data

ORL    IFD,#(XTALE)             ; Enable XTAL oscillating
CALL   _page_p_sfr_write        ; Write data to CKCON2
CALL   Delay_10mS

ANL    IFD,#~(OSCS1|OSCS0)      ; Switch OSCin source to XTAL 32.768KHz
ORL    IFD,#OSCS0
CALL   _page_p_sfr_write        ; Write data to CKCON2

ANL    IFD,#~(IHRCOE)          ; Disable IHRCO
CALL   _page_p_sfr_write        ; Write data to CKCON2

MOV    IFADRL,#PCON2           ; Index Page-P address to PCON2
CALL   _page_p_sfr_read         ; Read PCON2 data

ANL    IFD,#~(HSE)             ; Disable HSE when SYSCLK ≤ 6MHz for power saving
CALL   _page_p_sfr_write        ; Write data to PCON2

ANL    CKCON0,#~(SCK2|SCK1|SCK0); SYSCLK = OSCin/1 = 32.768KHz

```

**C Code Example:**

```

IFADRL = CKCON2; // Index Page-P address to CKCON2
page_p_sfr_read(); // Read CKCON2 data

IFD |= XTALE; // Enable XTAL oscillating
page_p_sfr_write(); // Write data to CKCON2

Dealy_10mS();

IFD &= ~(OSCS1 | OSCS0); // Switch OSCin source to XTAL.
IFD |= OSCS0;
page_p_sfr_write (); // Write data to CKCON2

IFD &= ~IHRCOE; // Disable IHRCO if MCU is switched from IHRCO
page_p_sfr_write(); // Write data to CKCON2.

IFADRL = PCON2; // Index Page-P address to PCON2
page_p_sfr_read(); // Read PCON2 data.

IFD &= ~HSE; // Disable HSE when SYSCLK ≤ 6MHz for power saving
page_p_sfr_write(); // Write data to PCON2

CKCON0 &= ~(SCK2 | SCK1 | SCK0); // SYSCLK = OSCin/1 = 32.768KHz

```

**(4). Required Function: Enter Watch mode with 2S wake-up duration**

**Assembly Code Example:**

```

ORG    0003Bh
SystemFlag_ISR:
ANL    PCON1,#(WDTF)           ; Clear WDT flag (write "1")
RETI

main:
ANL    PCON1,#WDTF             ; Clear WDTF flag (write "1")

```

```

    ORL    WDTCR,#(NSW|ENW|PS2|PS1|PS0)
           ;Enable WDT and NSW (for watch mode)
           ;Set PS[2:0] = 7 to select WDT period for 1.984s

    ORL    SFIE,#WDTFIE           ; Enable WDT interrupt
    ORL    EIE1,#ESF             ; Enable SystemFlag interrupt
    SETB   EA                     ; Enable Global interrupt

    ORL    PCON0,#PD             ; Set MCU to power down

;   MCU wait for wake-up

```

C Code Example:

```

void SystemFlag_ISR (void) interrupt 7
{
    PCON1 &= WDTF;                // Clear WDT flag (write "1")
}

void main (void)
{
    PCON1 &= WDTF;                // Clear WDT flag (write "1")
    WDTCR |= (NSW | ENW | PS2 | PS1 | PS0); // Enable WDT and NSW (for watch mode)
                                           // Set PS[2:0] = 7 to select WDT period for 1.984s

    SFIE |= WDTFIE;              // Enable WDT interrupt
    EIE1 |= ESF;                 // Enable SystemFlag interrupt
    EA = 1;                       // Enable global interrupt

    PCON0 |= PD;                 // Set MCU to power down

//   MCU wait for wake-up
}

```

*(5). Required Function: Monitor Mode*

Assembly Code Example:

```

    ORG    0003Bh
SystemFlag_ISR:
    ANL    PCON1,#(BOF1)         ; Clear BOD1 flag (write "1")
    RETI

main:
    MOV    IFADRL,#PCON3        ; Index Page-P address to PCON3
    CALL   _page_p_sfr_read      ; Read PCON3 data

    ORL    IFD,#AWBOD1          ; Enable BOD1 operating in power-down mode
    CALL   _page_p_sfr_write     ; Write data to PCON3

    ORL    SFIE,#BOF1IE         ; Enable BOF1 interrupt
    ORL    EIE1,#ESF            ; Enable SystemFlag interrupt
    SETB   EA                    ; Enable global interrupt

    ORL    PCON0,#PD            ; Set MCU to power down

;   MCU wait for wake-up

```

C Code Example:

```

void SystemFlag_ISR() interrupt 7
{
    PCON1 &= BOF1;                // Clear BOD1 flag (write "1")
}

void main()
{

```

```
IFADRL = PCON3;           // Index Page-P address to PCON3
page_p_sfr_read();       // Read PCON3 data

IFD |= AWBOD1;           // Enable BOD1 operating in power-down mode
page_p_sfr_write();      // Write data to PCON3

SFIE |= BOF1IE;         // Enable BOD1 interrupt
EIE1 |= ESF;            // Enable SystemFlag interrupt
EA = 1;                 // Enable global interrupt

PCON0 |= PD;           // Set MCU to power down

// MCU wait for wake-up
}
```

## 13. Configurable I/O Ports

The **MG84FG516** has following I/O ports: P0.0~P0.7, P1.0~P1.7, P2.0~P2.7, P3.0~P3.7, P4.0~P4.6, P5.0~P5.7 and P6.0~P6.7. If select external crystal oscillator as system clock input, Port 6.0 and Port 6.1 are configured to XTAL2 and XTAL1. 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
64-pin LQFP	P0.0~P0.7, P1.0~P1.7, P2.0~P2.7, P3.0~P3.7, P4.0~P4.6, P5.0~P5.7, P6.0~P6.7	55
48-pin LQFP	P0.0~P0.7, P1.0~P1.7, P2.0~P2.7, P3.0~P3.7, P4.0~P4.6, P6.0, P6.1	41

### 13.1. IO Structure

The I/O operating modes are distinguished two groups in **MG84FG516**. 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 two output modes, push-pull output and open-drain output with pull-up resistor control. The default setting of this group I/O is open-drain mode with output high, which means input mode with high impedance state.

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

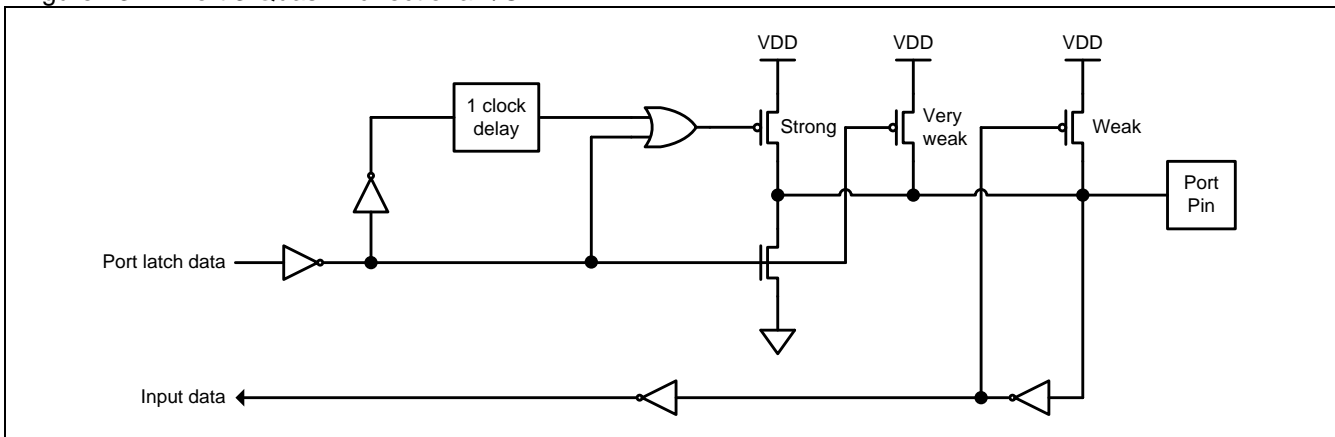
#### 13.1.1. Port 3 Quasi-Bidirectional IO Structure

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

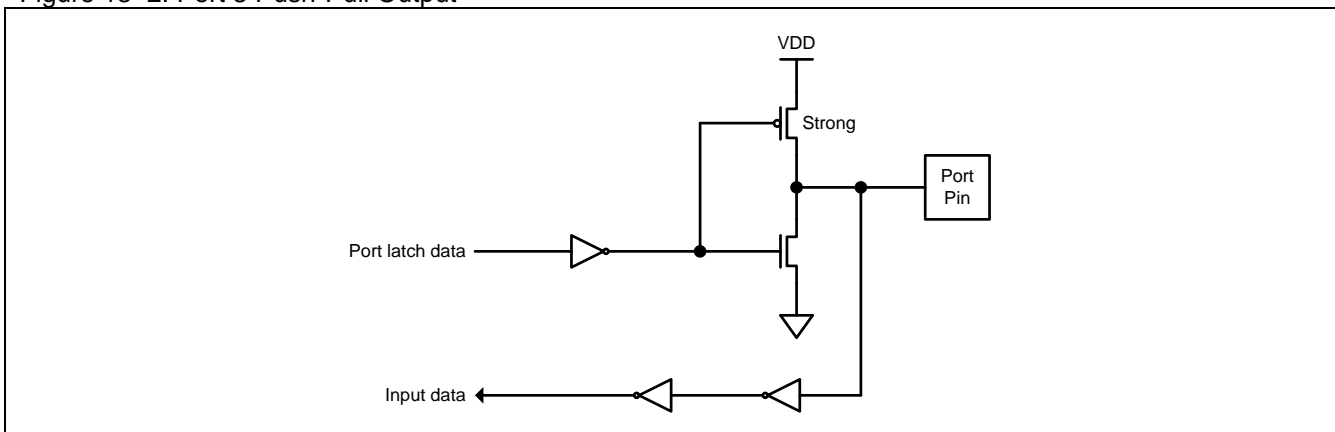


### 13.1.2. Port 3 Push-Pull Output Structure

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](#).

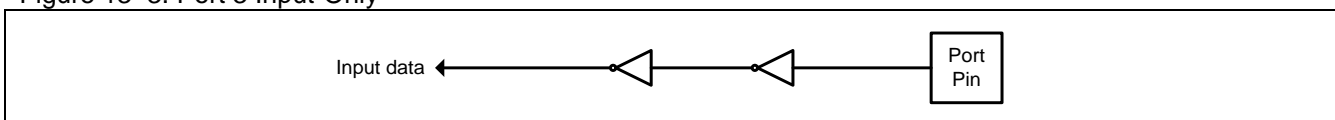
Figure 13–2. Port 3 Push-Pull Output



### 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



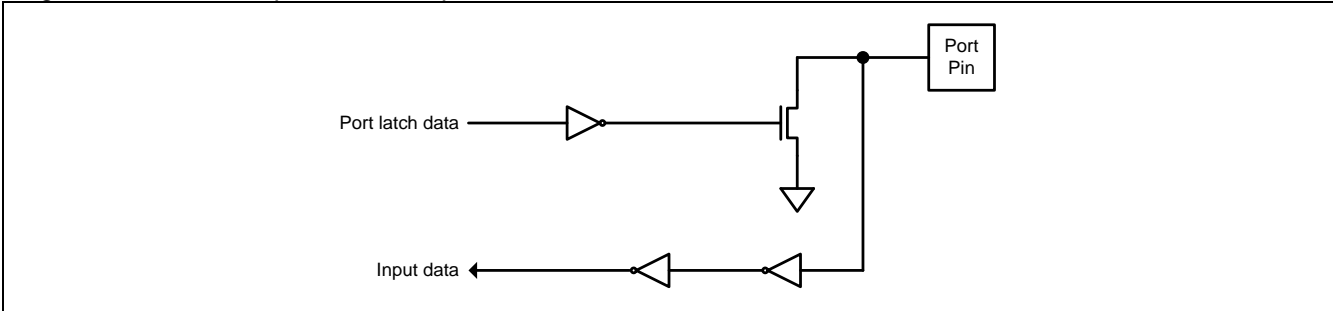
### 13.1.4. Port 3 Open-Drain Output Structure

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

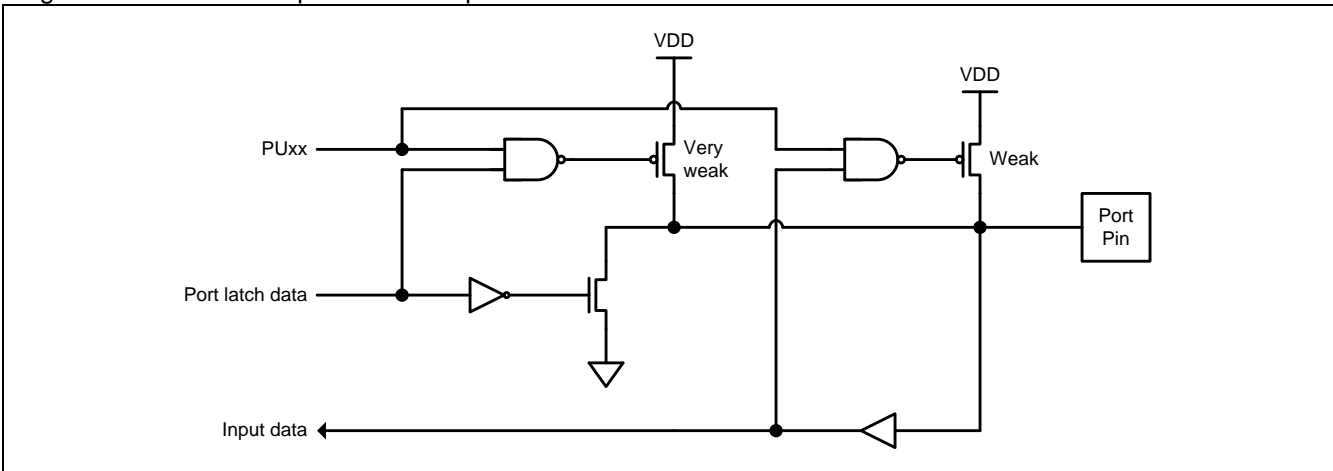


### 13.1.5. General Open-Drain Output Structure

The open-drain output configuration on general port pins only drives the pull-down transistor of the port pin when the Port Data register contains a logic “0”. To use this configuration in application, a port pin can select an external pull-up, or an on-chip pull-up by software enabled in PUCON0 and PUCON1.

The general open-drain port configuration is shown in [Figure 13-5](#).

Figure 13-5. General Open-Drain Output

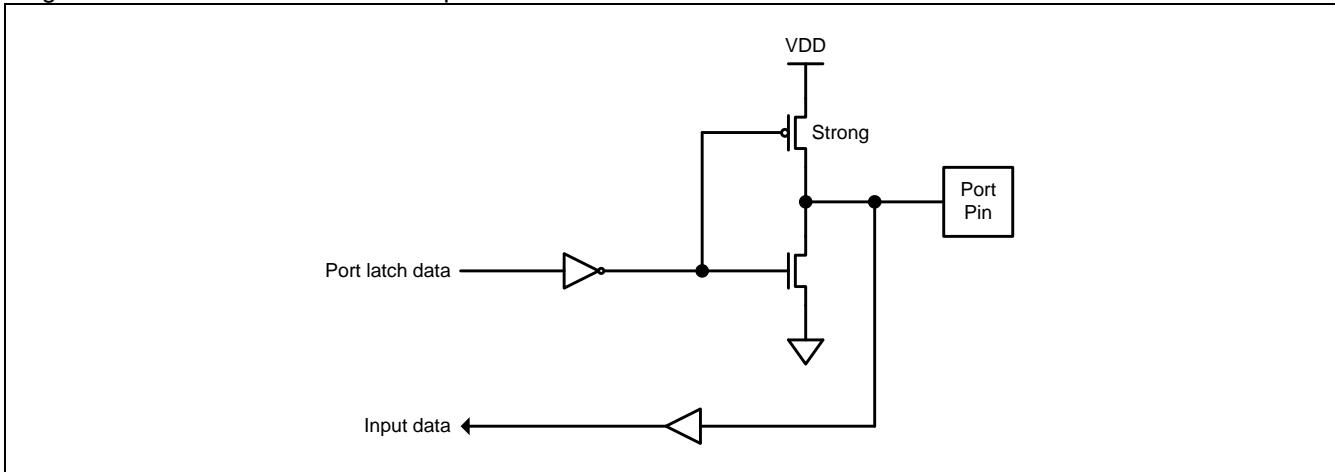


### 13.1.6. General Push-Pull Output Structure

The push-pull output configuration on general port pins has the same pull-down structure as the open-drain 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 open-drain mode.

The push-pull port configuration is shown in [Figure 13-6](#).

Figure 13–6. General Push-Pull Output



### 13.1.7. General Port 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, P1.7 is configured as a digital input by setting P1M0.7 to a logic 0 and P1.7 to a logic 1.

## 13.2. I/O Port Register

All I/O port pins on the **MG84FG516** may be individually and independently configured by software to select its operating mode. Only Port 3 has four operating modes, as shown in [Table 13–2](#). Two mode registers select the output type for each port 3 pin.

Table 13–2. Port 3 Configuration Settings

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

Where y=0~7 (port pin). The registers P3M0 and P3M1 are listed in each port description.

Other general port pins support two operating modes, as shown in [Table 13–3](#). One mode register selects the output type for each port pin.

Table 13–3. General Port Configuration Settings

PxM0.y	Port Mode
0	Open-Drain Output
1	Push-Pull Output

Where x= **0, 1, 2, 4, 5, 6** (port number), and y=0~7 (port pin). The registers PxM0 are listed in each port description.

### 13.2.1. Port 0 Register

#### P0: Port 0 Register

SFR Page = 0~F

SFR Address = 0x80

RESET = 1111-1111

7	6	5	4	3	2	1	0
P0.7	P0.6	P0.5	P0.4	P0.3	P0.2	P0.1	P0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: P0.7~P0.0 could be set/cleared by CPU.



**P0M0: Port 0 Mode Register 0**

SFR Page = 0~F

SFR Address = 0x93 RESET = 0000-0000

7	6	5	4	3	2	1	0
P0M0.7	P0M0.6	P0M0.5	P0M0.4	P0M0.3	P0M0.2	P0M0.1	P0M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin output mode is configured to open-drain.

1: Port pin output mode is configured to push-pull.

**13.2.2. 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
P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: P1.7~P1.0 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
P1M0.7	P1M0.6	P1M0.5	P1M0.4	P1M0.3	P1M0.2	P1M0.1	P1M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin output mode is configured to open-drain.

1: Port pin output mode is configured to push-pull.

**P1AIO: Port 1 Analog Input Only**

SFR Page = 0~F

SFR Address = 0x92 RESET = 0000-0000

7	6	5	4	3	2	1	0
P17AIO	P16AIO	P15AIO	P14AIO	P13AIO	P12AIO	P11AIO	P10AIO
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin has digital and analog input capability.

1: Port pin only has analog input only for ADC input application. The corresponding Port PIN Register bit will always read as "0" when this bit is set.

**13.2.3. Port 2 Register**

**P2: Port 2 Register**

SFR Page = 0~F

SFR Address = 0xA0 RESET = 1111-1111

7	6	5	4	3	2	1	0
P2.7	P2.6	P2.5	P2.4	P2.3	P2.2	P2.1	P2.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: P2.7~P2.0 could be only set/cleared by CPU.

**P2M0: Port 2 Mode Register 0**

SFR Page = 0~F

SFR Address = 0x95 RESET = 0000-0000

7	6	5	4	3	2	1	0
P2M0.7	P2M0.6	P2M0.5	P2M0.4	P2M0.3	P2M0.2	P2M0.1	P2M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin output mode is configured to open-drain.  
 1: Port pin output mode is configured to push-pull.

### 13.2.4. 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
P3.7	P3.6	P3.5	P3.4	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: P3.7~P3.0 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
P3M0.7	P3M0.6	P3M0.5	P3M0.4	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
P3M1.7	P3M1.6	P3M1.5	P3M1.4	P3M1.3	P3M1.2	P3M1.1	P3M1.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

P3.7 and P3.6 have alternated function for /RD and /WR in off-chip memory access cycle.

### 13.2.5. Port 4 Register

#### P4: Port 4 Register

SFR Page = 0~F  
 SFR Address = 0xE8 RESET = X111-1111

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

Bit 7: Reserved. Software must write "1" on this bit when P4 is written.

Bit 6~0: P4.6~P4.0 could be only set/cleared by CPU.  
 P4.6 has an alternated function for ALE in off-chip memory access cycle.

#### P4M0: Port 4 Mode Register 0

SFR Page = 0~F  
 SFR Address = 0xB3 RESET = X000-0000

7	6	5	4	3	2	1	0
--	P4M0.6	P4M0.5	P4M0.4	P4M0.3	P4M0.2	P4M0.1	P4M0.0
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 P4M0 is written.

Bit 6~0:  
 0: Port pin output mode is configured to open-drain.  
 1: Port pin output mode is configured to push-pull.

### 13.2.6. Port 5 Register

#### P5: Port 5 Register

SFR Page = 0 only

SFR Address = 0xF8

RESET = 1111-1111

7	6	5	4	3	2	1	0
P5.3	P5.3	P5.3	P5.3	P5.3	P5.2	P5.1	P5.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: P5.7~P5.0 could be only set/cleared by CPU.

#### P5M0: Port 5 Mode Register 0

SFR Page = 0 only

SFR Address = 0xB5

RESET = 0000-0000

7	6	5	4	3	2	1	0
P5M0.7	P5M0.6	P5M0.5	P5M0.4	P5M0.3	P5M0.2	P5M0.1	P5M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin output mode is configured to open-drain.

1: Port pin output mode is configured to push-pull.

### 13.2.7. Port 6 Register

#### P6: Port 6 Register

SFR Page = 1 only

SFR Address = 0xF8

RESET = 1111-1111

7	6	5	4	3	2	1	0
P6.7	P6.6	P6.5	P6.4	P6.3	P6.2	P6.1	P6.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: P6.7~P6.0 could be only set/cleared by CPU.

P6.1 and P6.0 have the alternated function for crystal oscillating circuit, XTAL1 and XTAL2.

#### P6M0: Port 6 Mode Register 0

SFR Page = 1 only

SFR Address = 0xB5

RESET = X000-0000

7	6	5	4	3	2	1	0
P6M0.7	P6M0.6	P6M0.5	P6M0.4	P6M0.3	P6M0.2	P6M0.1	P6M0.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin output mode is configured to open-drain.

1: Port pin output mode is configured to push-pull.

### 13.2.8. Pull-Up Control Register

#### PUCON0: Port Pull-up Control Register 0

SFR Page = 0 only

SFR Address = 0xB4

RESET = 0000-0000

7	6	5	4	3	2	1	0
P4PU1	P4PU0	P2PU1	P2PU0	P1PU1	P1PU0	P0PU1	P0PU0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: Port 4 pull-up enable control on high nibble.

0: Disable the P4.6 ~ P4.4 pull-up resistor in open-drain output mode.

1: Enable the P4.6 ~ P4.4 pull-up resistor in open-drain output mode.

Bit 6: Port 4 pull-up enable control on low nibble.

0: Disable the P4.3 ~ P4.0 pull-up resistor in open-drain output mode.

1: Enable the P4.3 ~ P4.0 pull-up resistor in open-drain output mode.

Bit 5: Port 2 pull-up enable control on high nibble.  
 0: Disable the P2.7 ~ P2.4 pull-up resistor in open-drain output mode.  
 1: Enable the P2.7 ~ P2.4 pull-up resistor in open-drain output mode.

Bit 4: Port 2 pull-up enable control on low nibble.  
 0: Disable the P2.3 ~ P2.0 pull-up resistor in open-drain output mode.  
 1: Enable the P2.3 ~ P2.0 pull-up resistor in open-drain output mode.

Bit 3: Port 1 pull-up enable control on high nibble.  
 0: Disable the P1.7 ~ P1.4 pull-up resistor in open-drain output mode.  
 1: Enable the P1.7 ~ P1.4 pull-up resistor in open-drain output mode.

Bit 2: Port 1 pull-up enable control on low nibble.  
 0: Disable the P1.3 ~ P1.0 pull-up resistor in open-drain output mode.  
 1: Enable the P1.3 ~ P1.0 pull-up resistor in open-drain output mode.

Bit 1: Port 0 pull-up enable control on high nibble.  
 0: Disable the P0.7 ~ P0.4 pull-up resistor in open-drain output mode.  
 1: Enable the P0.7 ~ P0.4 pull-up resistor in open-drain output mode.

Bit 0: Port 0 pull-up enable control on low nibble.  
 0: Disable the P0.3 ~ P0.0 pull-up resistor in open-drain output mode.  
 1: Enable the P0.3 ~ P0.0 pull-up resistor in open-drain output mode.

**PUCON1: Port Pull-up Control Register 1**

SFR Page = 1 only

SFR Address = 0xB4

RESET = XXXX-0000

7	6	5	4	3	2	1	0
--	--	--	--	P6PU1	P6PU0	P5PU1	P5PU0
W	W	W	W	R/W	R/W	RW	RW

Bit 7 ~ 4: Reserved. Software must write "0" on this bit when PUCON1 is written.

Bit 3: Port 6 pull-up enable control on high nibble.  
 0: Disable the P6.7 ~ P6.4 pull-up resistor in open-drain output mode.  
 1: Enable the P6.7 ~ P6.4 pull-up resistor in open-drain output mode.

Bit 2: Port 6 pull-up enable control on low nibble.  
 0: Disable the P6.3 ~ P6.0 pull-up resistor in open-drain output mode.  
 1: Enable the P6.3 ~ P6.0 pull-up resistor in open-drain output mode.

Bit 1: Port 5 pull-up enable control on high nibble.  
 0: Disable the P5.7 ~ P5.4 pull-up resistor in open-drain output mode.  
 1: Enable the P5.7 ~ P5.4 pull-up resistor in open-drain output mode.

Bit 0: Port 5 pull-up enable control on low nibble.  
 0: Disable the P5.3 ~ P5.0 pull-up resistor in open-drain output mode.  
 1: Enable the P5.3 ~ P5.0 pull-up resistor in open-drain output mode.

### 13.3. GPIO Sample Code

(1). Required Function: Set P1.0 to input mode with on-chip pull-up resistor enabled

Assembly Code Example:

```
ANL    P1M0,#~P1M00      ; Configure P1.0 to open drain mode
SETB   P10                ; Set P1.0 data latch to "1" to enable input mode
ORL    PUCON0,#PU10      ; Enable the P1.3~P1.0 on-chip pull-up resistor
```

C Code Example:

```
P1M0 &= P1M00;          // Configure P1.0 to open drain mode
P10 = 1;                // Set P1.0 data latch to "1" to enable input mode
PUCON0 |= PU10;        // Enable the P1.3~P1.0 on-chip pull-up resistor
```

## 14. Interrupt

The **MG84FG516** has 17 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, EIE2, EIP2L, EIP2H and XICON. The IP0H (Interrupt Priority 0 High), EIP1H (Extended Interrupt Priority 1 High) and EIP2H (Extended Interrupt Priority 2 High) 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.

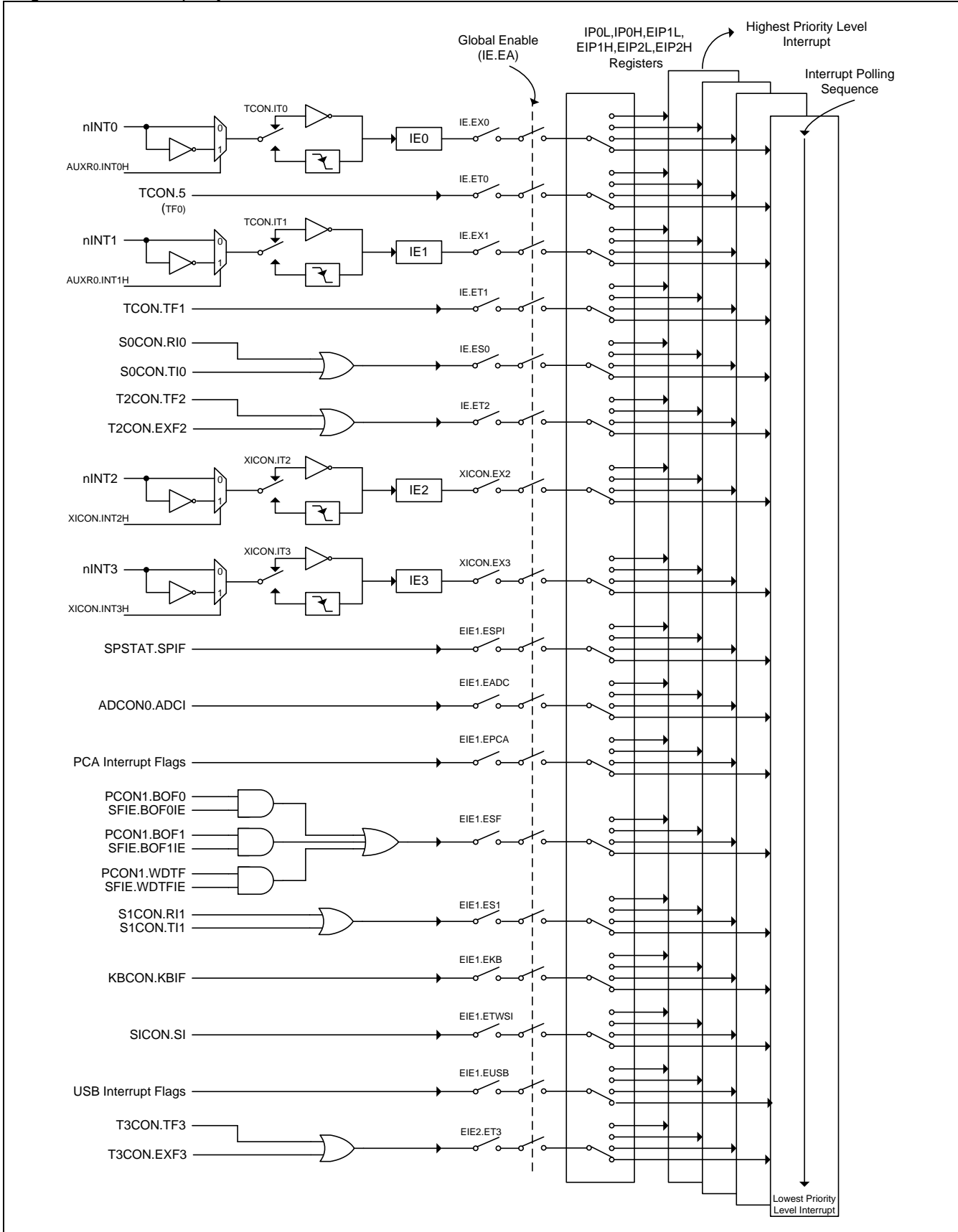
Table 14–1. Interrupt Sources

No	Source Name	Enable Bit	Request Bits	Priority Bits	Polling Priority	Vector Address
#1	External Interrupt 0, nINT0	EX0	IE0	[ PX0H, PX0L ]	(Highest)	0003H
#2	Timer 0	ET0	TF0	[ PT0H, PT0L ]	...	000Bh
#3	External Interrupt 1, nINT1	EX1	IE1	[ PX1H, PX1L ]	...	0013H
#4	Timer 1	ET1	TF1	[ PT1H, PT1L ]	...	001BH
#5	Serial Port 0	ES0	RI0, TI0	[ PS0H, PS0L ]	...	0023H
#6	Timer 2	ET2	TF2, EXF2	[ PT2H, PT2L ]	...	002Bh
#7	External Interrupt 2, nINT2	EX2	IE2	[ PX2H, PX2L ]	...	0033H
#8	External Interrupt 3, nINT3	EX3	IE3	[ PX3H, PX3L ]	...	003BH
#9	SPI	ESPI	SPIF	[ PSPIH, PSPIL ]	...	0043H
#10	ADC	EADC	ADCI	[ PADCH, PADCL ]	...	004Bh
#11	PCA	EPCA	CF, CCF <sub>n</sub> (n=0~5)	[ PPCAH, PPCAL ]	...	0053H
#12	System Flag	ESF	BOF1, BOF0, WDTF	[ PSFH, PSFL ]	...	005BH
#13	Serial Port 1	ES1	RI1, TI1	[ PS1H, PS1L ]	...	0063H
#14	Keypad Interrupt	EKB	KBIF	[ PKBH, PKBL ]	...	006BH
#15	TWSI	ETWSI	SI	[ PTWIH, PTWIL ]	...	0073H
#16	USB	EUSB	(Note 1)	[ PUSBH, PUSBL ]	...	007BH
#17	Timer 3	ET3	TF3, EXF3	[ PT3H, PT3L ]	(Lowest)	0083H

Note1: The USB interrupt flags include:

- (1) URST, URSM and USUS: contained in USB register UPCON.
- (2) UTXD0, URXD0, UTXD1, URXD1, UTXD2, URXD2 and UOFIF: contained in USB register UIFLG.
- (3) UTXD3, URXD3, UTXD4, URXD4, UTXD5 and URXD5: contained in USB register UIFLG1

Figure 14–1. Interrupt System



## 14.2. Interrupt Source

Table 14–2. Interrupt Source Flag

No	Source Name	Request Bits	Bit Location
#1	External Interrupt 0,nINT0	IE0	TCON.1
#2	Timer 0	TF0	TCON.5
#3	External Interrupt 1,nINT1	IE1	TCON.3
#4	Timer 1	TF1	TCON.7
#5	Serial Port 0	RI0, TI0	S0CON.0 S0CON.1
#6	Timer 2	TF2, EXF2	T2CON.7 T2CON.6
#7	External Interrupt 2,nINT2	IE2	XICON.1
#8	External Interrupt 3,nINT3	IE3	XICON.5
#9	SPI	SPIF	SPSTAT.7
#10	ADC	ADCI	ADCON0.4
#11	PCA	CF, CCFn (n=0~5)	CCON.7 CCON.5~0
#12	System Flag	BOF1, BOF0, WDTF	PCON1.2~0
#13	Serial Port 1	RI1, TI1	S1CON.0 S1CON.1
#14	Keypad Interrupt	KBIF	KBCON.0
#15	TWSI	SI	SICON.3
#16	USB	(Note 1)	(Note 1)
#17	Timer 3	TF3, EXF3	T3CON.7 T3CON.6

Note1: The USB interrupt flags include:

- (1) URST, URSM and USUS: contained in USB register UPCON.
- (2) UTXD0, URXD0, UTXD1, URXD1, UTXD2, URXD2 and UOFIF: contained in USB register UIFLG.
- (3) UTXD3, URXD3, UTXD4, URXD4, UTXD5 and URXD5: contained in USB register UIFLG1

The external interrupt nINT0, nINT1, nINT2 and nINT3 can each be either level-activated or transition-activated, depending on bits IT0 and IT1 in register TCON, IT2 and IT3 in register XICON. The flags that actually generate these interrupts are bits IE0 and IE1 in TCON, IE2 and IE3 in XICON. 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. 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 in ADCON0. It will not be cleared by hardware when the service routine is vectored to.

The PCA interrupt is generated by the logical OR of CF, CCF5, CCF4, 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 BOF1, BOF0 and WDTF in PCON1, which is set by on chip Brownout-Detectors (BOD1 & BOD0) meet the low voltage event and Watch-Dog-Timer overflow. They will not be cleared by hardware when the service routine is vectored to.

*Note: the WDTFIE function is under verifying.*

The serial port 1 interrupt is generated by the logical OR of RI1 and TI1. Neither of these flags is cleared by hardware when the service routine is vectored to. The service routine should poll RI1 and TI1 to determine which one to request service and it will be cleared by software.

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

The USB interrupt is generate by a grouping of USB event flags in USB SFR, which are set by USB engine detecting a new bus state or USB function event happened. They will not be cleared by hardware when the service routine is vectored to.

The timer3 interrupt is generated by the logical OR of TF3 and EXF3. Just the same as serial port, neither of these flags is 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
#1	External Interrupt 0,nINT0	EX0	IE.0
#2	Timer 0	ET0	IE.1
#3	External Interrupt 1,nINT1	EX1	IE.2
#4	Timer 1	ET1	IE.3
#5	Serial Port 0	ES0	IE.4
#6	Timer 2	ET2	IE.5
#7	External Interrupt 2,nINT2	EX2	XICON.2
#8	External Interrupt 3,nINT3	EX3	XICON.3
#9	SPI	ESPI	EIE1.0
#10	ADC	EADC	EIE1.1
#11	PCA	EPCA	EIE1.2
#12	System Flag	ESF	EIE1.3
#13	Serial Port 1	ES1	EIE1.4
#14	Keypad Interrupt	EKB	EIE1.5
#15	TWSI	ETWSI	EIE1.6
#16	USB	EUSB	EIE1.7
#17	Timer 3	ET3	EIE2.0

There are 17 interrupt sources available in **MG84FG516**. Each of these interrupt sources can be individually enabled or disabled by setting or clearing an interrupt enable bit in the registers IE, EIE1, EIE2 and XICON. 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, EIP2L and EIP2H 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 an hardware LCALL to the appropriate service routine as long as it is not blocked by any of the following conditions.

Block conditions:

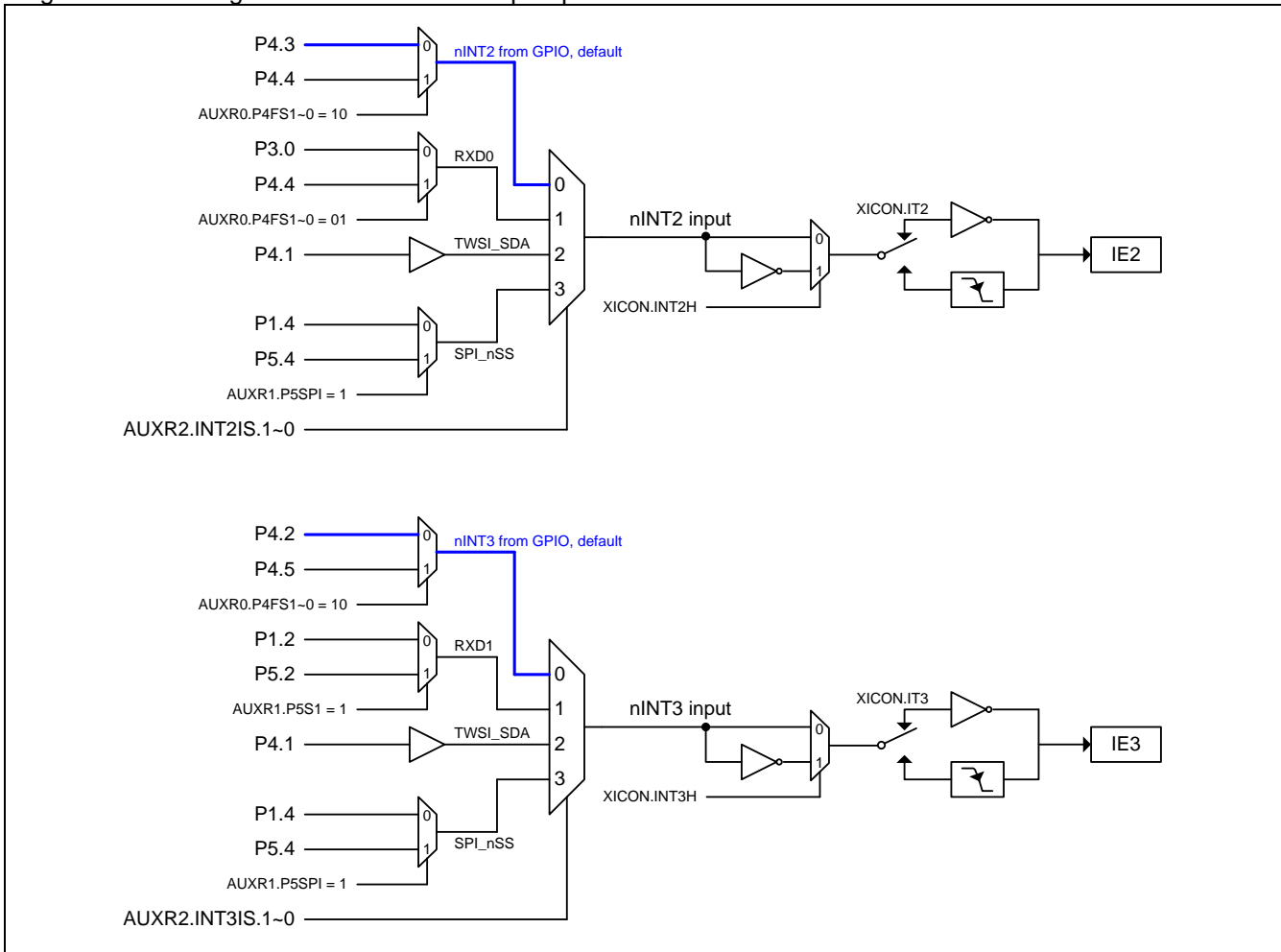
- An interrupt of equal or higher priority level is already in progress.
- The current cycle (polling cycle) is not the final cycle in the execution of the instruction in progress.
- The instruction in progress is RETI or any write to the IE, IP0L, IPH, EIE1, EIP1L and EIP1H registers.

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 IP0L, IP0H, EIE1, EIP1L or EIP1H, then at least one or more instruction will be executed before any interrupt is vectored to.

## 14.6. nINT2/nINT3 Input Source Selection

The **MG84FG516** provides flexible nINT2 and nINT3 source selection to share the port pin input of on-chip serial interface. That will support the additional remote wakeup function for communication peripheral in power-down mode. The nINT2/nINT3 input can be routed to the interface pin to catch port change and set them as an interrupt input event to wakeup MCU. INT3H (XICON.7) and INT2H (XICON.3) configure the port change detection level on low/falling or high/rising event. In MCU power-down mode, both of the falling edge or rising edge configurations of the external interrupts are forced to level-sensitive operation.

Figure 14–2. Configuration of nINT2/nINT3 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 Edge flag.

0: Cleared when interrupt processed on if transition-activated.

1: Set by hardware when external interrupt 1 edge is detected (transmitted or level-activated).

Bit 2: IT1: Interrupt 1 Type control bit.

0: Cleared by software to specify low level triggered external interrupt 1. 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. If INT1H (AUXR0.1) is set, this bit specifies rising edge triggered on nINT1.

Bit 1: IE0, Interrupt 0 Edge flag.

0: Cleared when interrupt processed on if transition-activated.

1: Set by hardware when external interrupt 0 edge is detected (transmitted or level-activated).

Bit 0: IT0: Interrupt 0 Type control bit.

0: Cleared by software to specify low level triggered external interrupt 0. 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. 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 = 0X00-0000

7	6	5	4	3	2	1	0
EA	--	ET2	ES0	ET1	EX1	ET0	EX0
R/W	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 6: Reserved. Software must write "0" on this bit when IE is written.

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 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 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 1 interrupt.

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

**XICON: External Interrupt Control Register**

SFR Page = 0~F  
 SFR Address = 0xC0 RESET = 0000-0000

7	6	5	4	3	2	1	0
INT3H	EX3	IE3	IT3	INT2H	EX2	IE2	IT2
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: INT3H, nINT3 High/Rising trigger enable.  
 0: Maintain nINT3 triggered on low level or falling edge on P4.2.  
 1: Set nINT3 triggered on high level or rising edge on P4.2.

Bit 6: EX3, external interrupt 3 enable register.  
 0: Disable external interrupt 3.  
 1: Enable external interrupt 3.  
 When CPU in IDLE and PD mode, nINT3 event will trigger IE3 and have wake-up CPU capability if EX3 is enabled. If EX3 is disabled, IE3 on nINT3 will not wake-up CPU from IDLE or PD mode.

Bit 5: IE3, External interrupt 3 Edge flag.  
 0: Cleared by hardware when the interrupt is starting to be serviced. It also could be cleared by CPU.  
 1: Set by hardware when external interrupt edge detected. It also could be set by CPU.

Bit 4: IT3, Interrupt 3 type control bit.  
 0: Cleared by CPU to specify low level triggered on nINT3. If INT3H is set, this bit specifies high level triggered on nINT3.  
 1: Set by CPU to specify falling edge triggered on nINT3. If INT3H is set, this bit specifies rising edge triggered on nINT3.

Bit 3: INT2H, nINT2 High/Rising trigger enable.  
 0: Maintain nINT2 triggered on low level or falling edge on P4.3.  
 1: Set nINT2 triggered on high level or rising edge on P4.3.

Bit 2: EX2, external interrupt 2 enable register.  
 0: Disable external interrupt 2.  
 1: Enable external interrupt 2.  
 When CPU in IDLE and PD mode, nINT2 event will trigger IE2 and have wake-up CPU capability if EX2 is enabled. If EX2 is disabled, IE2 on nINT2 will not wake-up CPU from IDLE or PD mode.

Bit 1: IE2, External interrupt 2 Edge flag.  
 0: Cleared by hardware when the interrupt is starting to be serviced. It also could be cleared by CPU.  
 1: Set by hardware when external interrupt edge detected. It also could be set by CPU.

Bit 0: IT2, Interrupt 2 type control bit.  
 0: Cleared by CPU to specify low level triggered on nINT2. If INT2H is set, this bit specifies high level triggered on nINT2.  
 1: Set by CPU to specify falling edge triggered on nINT2. If INT2H is set, this bit specifies rising edge triggered on nINT2.

**EIE1: Extended Interrupt Enable 1 Register**

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

7	6	5	4	3	2	1	0
EUSB	ETWSI	EKBI	ES1	ESF	EPCA	EADC	ESPI
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: EUSB, Enable USB Interrupt.  
 0: Disable USB interrupt.  
 1: Enable USB interrupt.

Bit 6: ETWSI, Enable TWSI interrupt.  
 0: Disable TWSI interrupt.  
 1: Enable TWSI 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: ES1, Enable Serial Port 1 (UART1) interrupt.  
 0: Disable Serial Port 1 interrupt.  
 1: Enable Serial Port 1 interrupt.

Bit 3: ESF, Enable System Flag interrupt.  
 0: Disable the interrupt when the group of {BOF1, BOF0, WDTF} in PCON1 is set  
 1: Enable the interrupt of the flags of {BOF1, BOF0, WDTF} in PCON1 when the associated system flag interrupt is enabled in SFIE.

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

**EIE2: Extended Interrupt Enable 2 Register**

SFR Page = 0~F  
 SFR Address = 0xA5 RESET = XXXX-XXX0

7	6	5	4	3	2	1	0
--	--	--	--	--	--	--	ET3
W	W	W	W	W	W	W	R/W

Bit 7 ~1: Reserved. Software must write “0” on these bits when EIE2 is written.

Bit 0: ET3, Enable Timer 3 Interrupt.  
 0: Disable Timer 3 interrupt.  
 1: Enable Timer 3 interrupt.

**SFIE: System Flag Interrupt Enable Register**

SFR Page = 0~F  
 SFR Address = 0x8E RESET = XXXX-X000

7	6	5	4	3	2	1	0
--	--	--	--	--	BOF1IE	BOF0IE	WDTFIE
W	W	W	W	W	R/W	R/W	R/W

Bit 7 ~3: Reserved. Software must write “0” on these bits when EIE2 is written.

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.

*Note: the WDTFIE function is under verifying.*

**IP0L: Interrupt Priority 0 Low Register**

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

7	6	5	4	3	2	1	0
PX3L	PX2L	PT2L	PSL	PT1L	PX1L	PT0L	PX0L
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: PX3L, external interrupt 3 priority-L register.  
 Bit 6: PX2L, external interrupt 2 priority-L register.  
 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
PX3H	PX2H	PT2H	PSH	PT1H	PX1H	PT0H	PX0H
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: PX3H, external interrupt 3 priority-H register.  
 Bit 6: PX2H, external interrupt 2 priority-H register.  
 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.

**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
PUSBL	PTWIL	PKBL	PS1L	PSFL	PPCAL	PADCL	PSPIL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: PUSBL, USB interrupt priority-L register.  
 Bit 6: PTWIL, TWSI interrupt priority-L register.  
 Bit 5: PKBL, keypad interrupt priority-L register.  
 Bit 4: PS1L, UART1 interrupt priority-L register.  
 Bit 3: PSFL, system flag interrupt priority-L register.  
 Bit 2: PPCAL, PCA 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
PUSBH	PTWIH	PKBH	PS1H	PSFH	PPCAH	PADCH	PSPIH
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: PUSBH, USB interrupt priority-H register.

Bit 6: PTWIH, TWSI interrupt priority-H register.

Bit 5: PKBH, keypad interrupt priority-H register.

Bit 4: PS1H, UART1 interrupt priority-H register.

Bit 3: PSFH, system flag interrupt priority-H register.

Bit 2: PPCAH, PCA interrupt priority-H register.

Bit 1: PADCH, ADC interrupt priority-H register.

Bit 0: PSPIH, SPI interrupt priority-H register.

**EIP2L: Extended Interrupt Priority 2 Low Register**

SFR Page = 0~F

SFR Address = 0xA6 RESET = XXXX-XXX0

7	6	5	4	3	2	1	0
--	--	--	--	--	--	--	PT3L
W	W	W	W	W	W	W	R/W

Bit 7 ~ 1: Reserved. Software must write "0" on these bits when EIP2L is written.

Bit 0: PT3L, Timer 3 interrupt priority-L register.

**EIP2H: Extended Interrupt Priority 2 High Register**

SFR Page = 0~F

SFR Address = 0xA7 RESET = XXXX-XXX0

7	6	5	4	3	2	1	0
--	--	--	--	--	--	--	PT3H
W	W	W	W	W	W	W	R/W

Bit 7 ~ 1: Reserved. Software must write "0" on these bits when EIP2H is written.

Bit 0: PT3H, Timer 3 interrupt priority-H register.

**AUXR0: Auxiliary Register 0**

SFR Page = 0~F

SFR Address = 0xA1 RESET = 000X-0000

7	6	5	4	3	2	1	0
P6OC1	P6OC0	P60FD	--	P4FS1	P4FS0	INT1H	INT0H
R/W	R/W	R/W	W	R/W	R/W	R/W	R/W

Bit 1: INT1H, INT1 High/Rising trigger enable.

0: Remain INT1 triggered on low level or falling edge on P3.3.

1: Set INT1 triggered on high level or rising edge on P3.3.

Bit 0: INT0H, INT0 High/Rising trigger enable.

0: Remain INT0 triggered on low level or falling edge on P3.2.

1: Set INT0 triggered on high level or rising edge on P3.2.

**AUXR2: Auxiliary Register 2**

SFR Page = 0~F

SFR Address = 0xA3 RESET = 0000-0000

7	6	5	4	3	2	1	0
INT3IS1	INT3IS0	INT2IS1	INT2IS0	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W



Bit 7~6: INT3IS1~0, nINT3 input selection bits which function is defined as following table.

INT3IS1~0	nINT3 Input	Selected Port Pin	Switch Condition
00	nINT3 Port Pin	P4.2 or P4.5	AUXR0.P4FS1~0
01	RXD1 Port Pin	P1.2 or P5.2	AUXR1.P5S1
10	TWSI SDA Port Pin	P4.1	None
11	SPI nSS Port Pin	P1.4 or P5.4	AUXR1.P5SPI

Bit 5~4: INT2IS1~0, nINT2 input selection bits which function is defined as following table.

INT2IS1~0	nINT2	Selected Port Pin	Switch Condition
00	nINT2 Port Pin	P4.3 or P4.4	AUXR0.P4FS1~0
01	RXD0 Port Pin	P3.0 or P4.4	AUXR0.P4FS1~0
10	TWSI SDA Port Pin	P4.1	None
11	SPI nSS Port Pin	P1.4 or P5.4	AUXR1.P5SPI

## 14.8. Interrupt Sample Code

(1). Required Function: Set INT0 high level wake-up MCU in power-down mode

Assembly Code Example:

```
    ORG    00003h
ext_int0_isr:
    to do.....
    RETI

main:

    SETB   P32                ;
                                ;
    ORL    IP0L,#PX0L        ; Select INT0 interrupt priority
    ORL    IP0H,#PX0H        ;
                                ;
    ORL    AUXR0,#INT0H      ; Set INT0 High level active

    JB     P32,$              ; Confirm P3.2 input low

    SETB   EX0                ; Enable INT0 interrupt
    CLR    IE0                ; Clear INT0 flag
    SETB   EA                ; Enable global interrupt

    ORL    PCON0,#PD         ; Set MCU into Power Down mode
```

C Code Example:

```
void ext_int0_isr(void) interrupt 0
{
    To do.....
}

void main(void)
{
    P32 = 1;

    IP0L |= PX0L;              // Select INT0 interrupt priority
    IP0H |= PX0H;

    AUXR0 |= INT0H;           // Set INT0 High level active

    while(P32);               // Confirm P3.2 input low

    EX0 = 1;                  // Enable INT0 interrupt
    IE0 = 0;                  // Clear INT0 flag
    EA = 1;                   // Enable global interrupt

    PCON0 |= PD;              // Set MCU into Power Down mode
}
```

## 15. Timers/Counters

**MG84FG516** has four 16-bit Timers/Counters: Timer 0, Timer 1, Timer 2 and Timer 3. 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 increment register value. In other words, it is to count the standard C51 machine cycle. AUXR2.T0X12, AUXR2.T1X12, T2MOD.T2X12 and T3MOD.T3X12 are the function for Timer 0/1/2/3 to set the timer rate on every clock cycle. It behaves X12 times speed than standard C51 timer function.

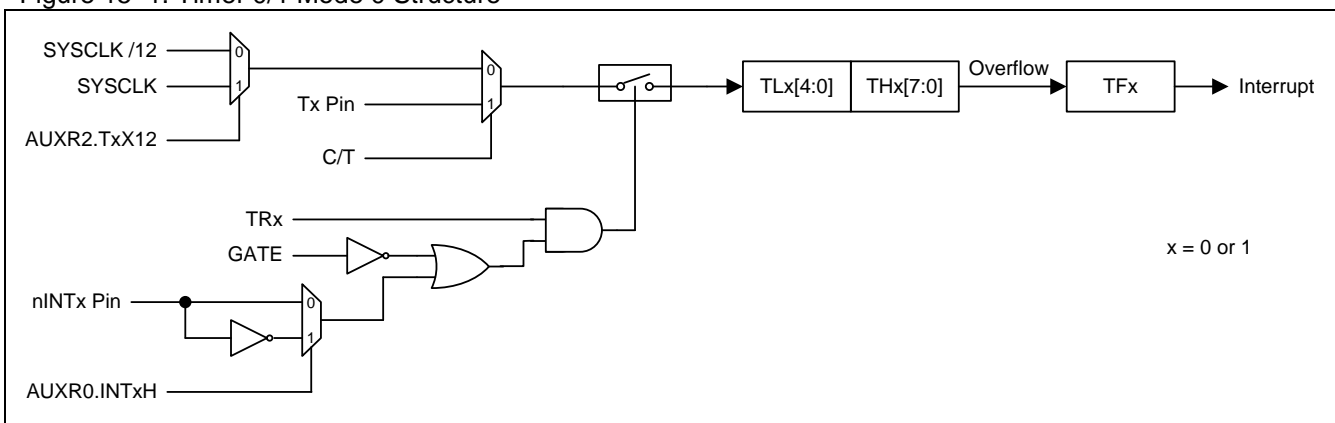
In the “counter” function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T0, T1, T2 or T3. In this function, the external input is sampled by every timer rate cycle. When the samples show a high in one cycle and a low in the next cycle, the count 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 13-bit register. As the count rolls over from all 1s to all 0s, it sets the timer interrupt flag TFX. The counted input is enabled to the timer when TRx = 1 and either GATE=0 or nINTx = 1. Mode 0 operation is the same for Timer0 and Timer1.

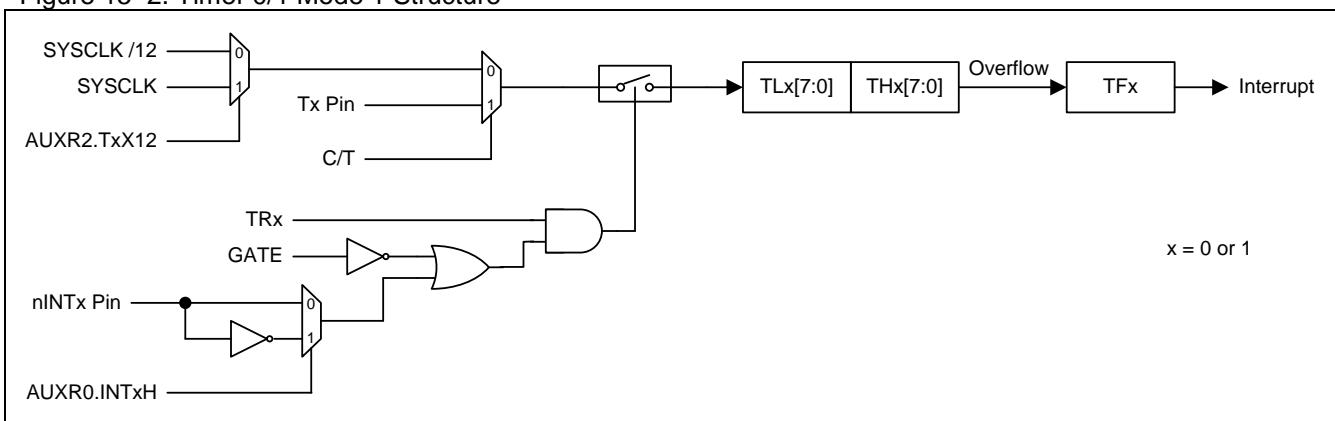
Figure 15–1. Timer 0/1 Mode 0 Structure



#### 15.1.2. Timer 0/1 Mode 1

Mode1 is the same as Mode0, except that the timer register is being run with all 16 bits.

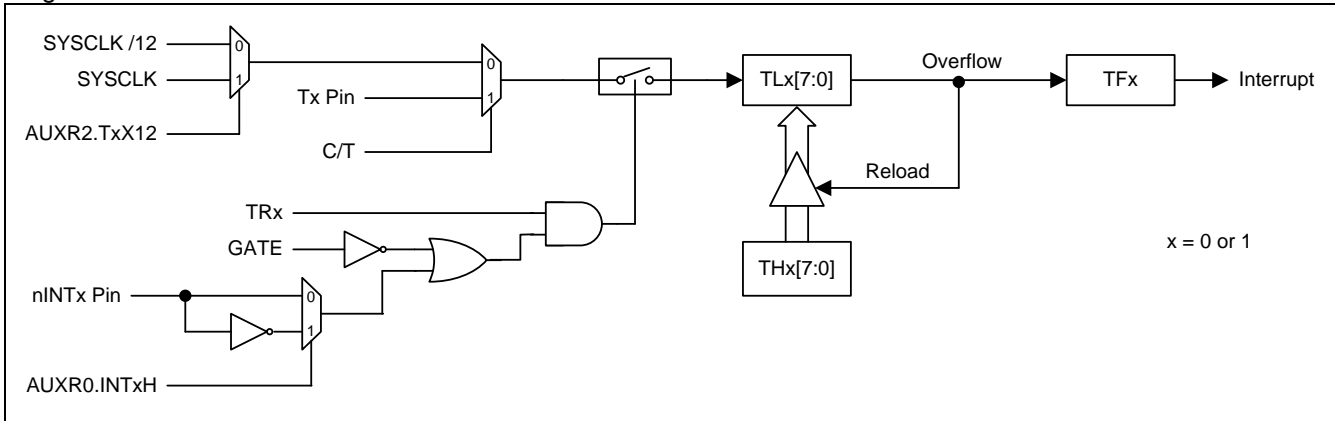
Figure 15–2. Timer 0/1 Mode 1 Structure



### 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.

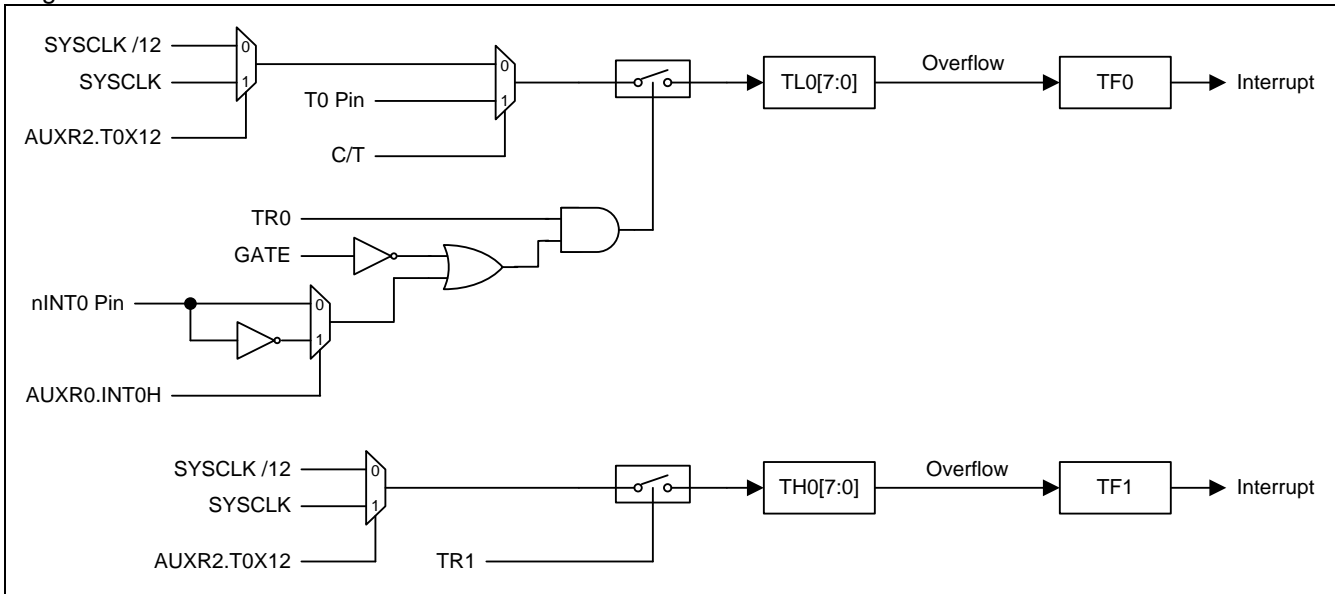
Figure 15–3. Timer 0/1 Mode 2 Structure



### 15.1.4. Timer 0/1 Mode 3

Timer1 in Mode3 simply holds its count, the effect is the same as setting TR1 = 1. Timer0 in Mode 3 enables TL0 and TH0 as two separate 8-bit counters. TL0 uses the Timer0 control bits such like C/T, GATE, TR0, INT0 and TF0. TH0 is locked into a timer function (can not be external event counter) and take over the use of TR1, TF1 from Timer1. TH0 now controls the Timer1 interrupt.

Figure 15–4. Timer 0/1 Mode 3 Structure



### 15.1.5. Timer 0/1 Programmable Clock-Out

Timer 0 and Timer 1 have a Clock-Out Mode (while C/Tx=0 & 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 P3.4 (T0CKO) and P3.5 (T1CKO) individually. The input clock (SYSCLK/12 or SYSCLK) increments the 8-bit timer (TL0 or TL1). 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. The following formula gives the clock-out frequency:

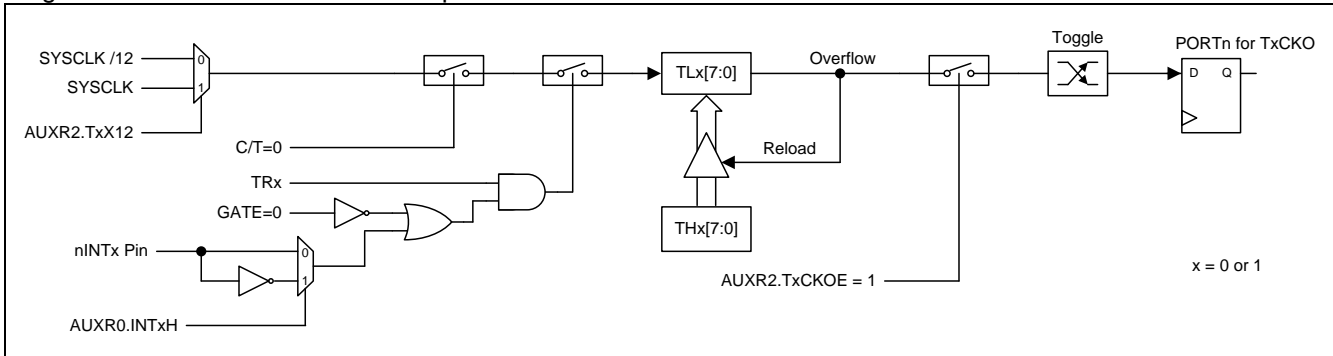
$$\text{T0/T1 Clock-out Frequency} = \frac{\text{SYSCLK Frequency}}{n \times (256 - \text{THx})}$$

; n=24, if TxX12=0  
; n=2, if TxX12=1  
; x = 0 or 1 & C/T = 0

**Note:**

- (1) Timer 0/1 overflow flag, TF0/1, will be set when Timer 0/1 overflows but not generate interrupt.
- (2) For SYSCLK=12MHz & TxX12=0, Timer 0/1 has a programmable output frequency range from 1.95KHz to 500KHz.
- (3) For SYSCLK=12MHz & TxX12=1, Timer 0/k1 has a programmable output frequency range from 23.43KHz to 6MHz.

Figure 15–5. Timer 0/1 in Clock Output Mode



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

- Select T0X12/T1X12 bit in AUXR2 register to decide the Timer 0/1 clock source.
- Set T0CKOE/T1CKOE bit in AUXR2 register.
- Clear C/T bit in TMOD register.
- 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 registers.
- 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.

**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: Cleared by software to turn Timer/Counter 1 off.

1: Set by software to turn Timer/Counter 1 on.

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: Cleared by software to turn Timer/Counter 0 off.  
 1: Set by software to turn Timer/Counter 0 on.

**TMOD: Timer/Counter Mode Control Register**

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

7	6	5	4	3	2	1	0
GATE	C/T	M1	M0	GATE	C/T	M1	M0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

|←-----Timer1-----→|←-----Timer0-----→|

Bit 7/3: Gate, Gating control for Timer1/0.  
 0: Disable gating control for Timer1/0.  
 1: Enable gating control for Timer1/0. When set, Timer1/0 or Counter1/0 is enabled only when /INT1 or /INT0 pin is high and TR1 or TR0 control bit is set.

Bit 6/2: C/T, Timer for Counter function selector.  
 0: Clear for Timer operation, input from internal system clock.  
 1: Set for Counter operation, input form T1 input pin.

Bit 5~4/1~0: Operating mode selection.

M1	M0	Operating Mode
0	0	13-bit timer/counter for Timer0 and Timer1
0	1	16-bit timer/counter for Timer0 and Timer1
1	0	8-bit timer/counter with automatic reload for Timer0 and Timer1
1	1 (Timer0)	TL0 is 8-bit timer/counter, TH0 is locked into 8-bit timer
1	1 (Timer1)	Timer/Counter1 Stopped

**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 = All  
 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
INT3IS1	INT3IS0	INT2IS1	INT2IS0	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	R/W	R/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.

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 P3.5.

Bit 0: T0CKOE, Timer 0 Clock Output Enable.  
 0: Disable Timer 0 clock output.  
 1: Enable Timer 0 clock output on P3.4.

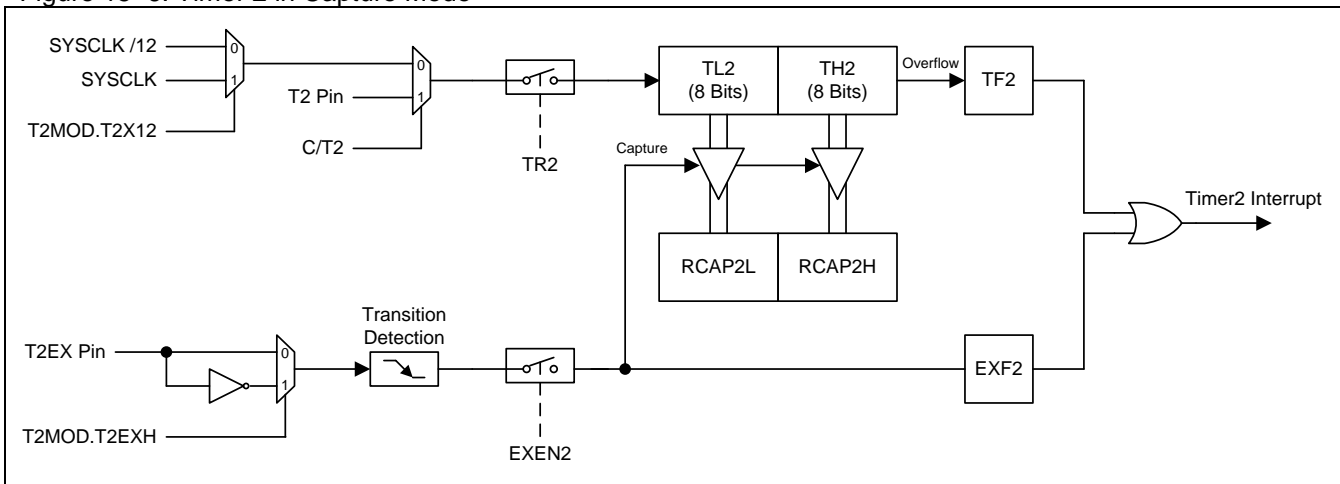
**15.2. Timer 2**

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

**15.2.1. Capture Mode (CP)**

In 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 external input T2EX 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 T2EX 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. The capture mode is illustrated in Figure 15-6.

Figure 15-6. Timer 2 in Capture Mode



### 15.2.2. Auto-Reload Mode (AR)

Figure 15–7 shows DCEN=0, 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 an overflow or by a 1-to-0 transition at input T2EX. This transition also sets the EXF2 bit. The Timer 2 interrupt, if enabled, can be generated when either TF2 or EXF2 are 1.

Figure 15–7. Timer 2 in Auto-Reload Mode (DCEN=0)

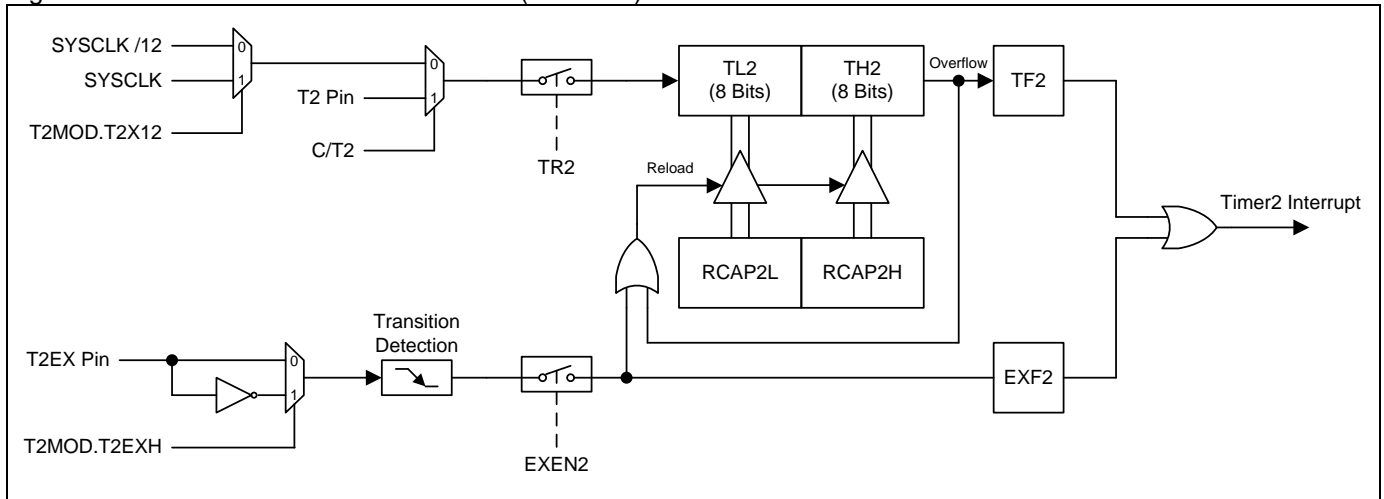
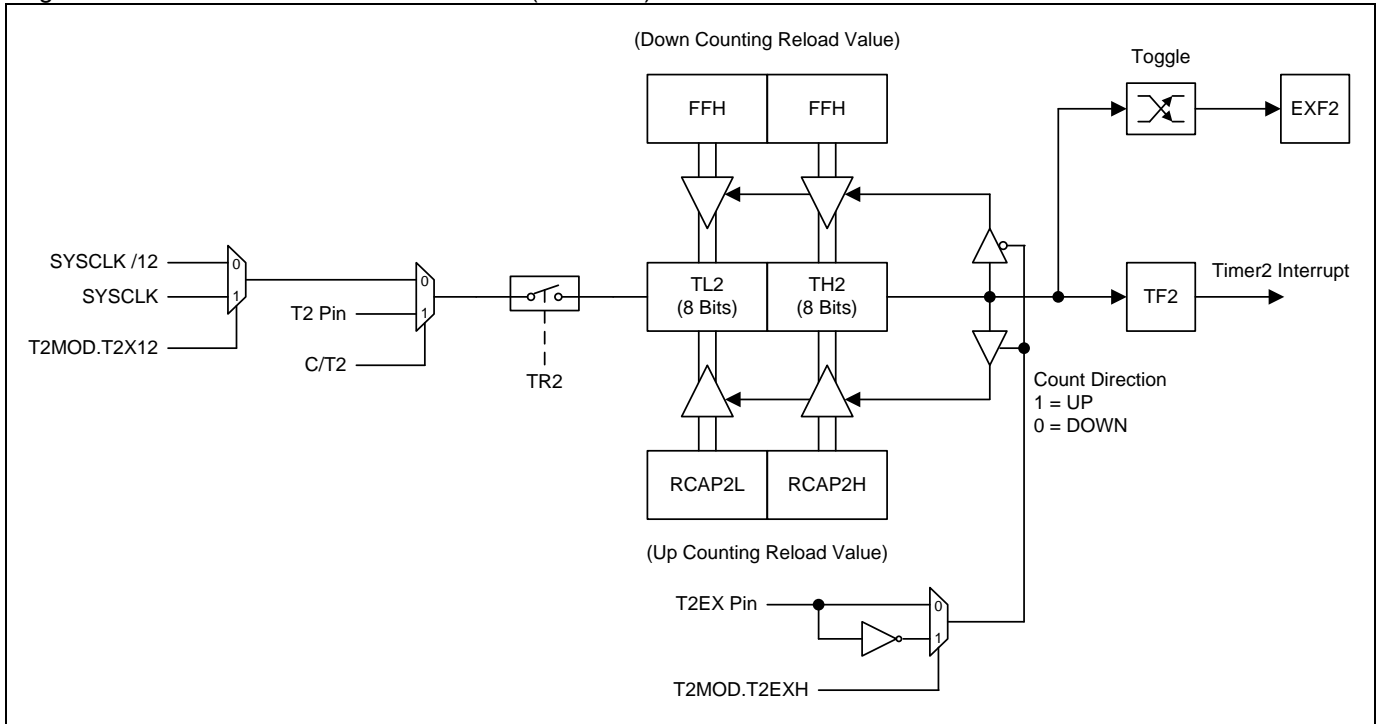


Figure 15–8 shows DCEN=1, which enables Timer 2 to count up or down. This mode allows pin T2EX to control the counting direction. When a logic 1 is applied at pin T2EX, Timer 2 will count up. Timer 2 will overflow at 0FFFFH and set the TF2 flag, which can then generate an interrupt if the interrupt is enabled. This overflow also causes the 16-bit value in RCAP2L and RCAP2H to be reloaded into the timer registers TL2 and TH2. A logic 0 applied to pin T2EX causes Timer 2 to count down. The timer will underflow when TL2 and TH2 become equal to the value stored in RCAP2L and RCAP2H. This underflow sets the TF2 flag and causes 0FFFFH to be reloaded into the timer registers TL2 and TH2.

The external flag EXF2 toggles when Timer 2 underflows or overflows. This EXF2 bit can be used as a 17th bit of resolution if needed. The EXF2 flag does not generate an interrupt in this mode.



Figure 15–8. Timer 2 in Auto-Reload Mode (DCEN=1)



### 15.2.3. 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 15–9 shows the Timer 2 in baud rate generation mode to generate RX Clock and TX Clock into UART engine (See Figure 16–6.). 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.

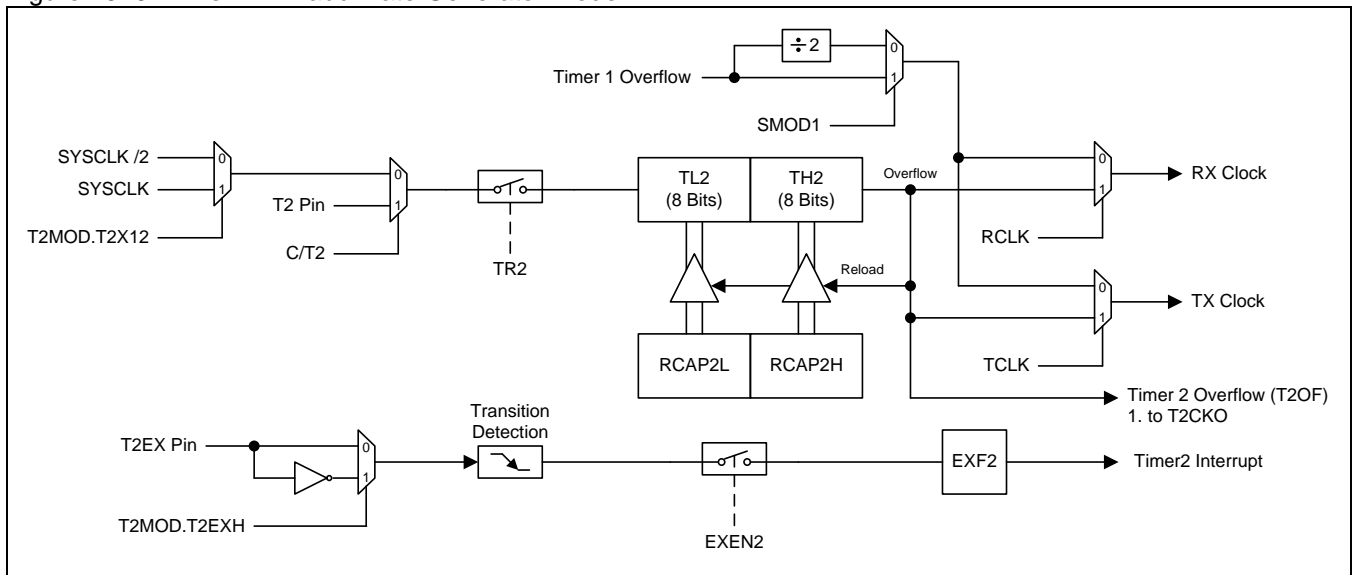
The Timer 2 as a baud rate generator mode is valid only if RCLK and/or TCLK=1 in T2CON register. Note that a rollover in TH2 does not set TF2, and will not generate an interrupt. Thus, the Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode. Also if the EXEN2 (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 “16.7.3 Baud Rate in Mode 1 & 3” to get baud rate setting value when using Timer 2 as the baud rate generator.*

Figure 15–9. Timer 2 in Baud-Rate Generator Mode



### 15.2.4. 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 P1.0. The input clock (SYSCLK/2 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. The following formula gives the clock-out frequency:

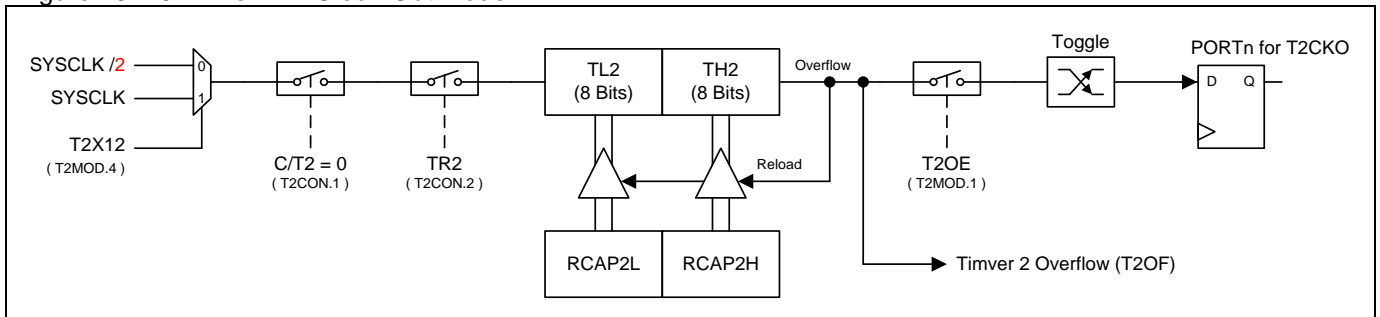
$$T2 \text{ Clock-out Frequency} = \frac{\text{SYSCLK Frequency}}{n \times (65536 - (\text{RCAP2H}, \text{RCAP2L}))} \quad ; n=4, \text{ if } T2X12=0$$

$$; n=2, \text{ if } T2X12=1$$

Note:

- (1) Timer 2 overflow flag, TF2, will be set when Timer 2 overflows but not generate interrupt.
- (2) For SYSCLK=12MHz & T2X12=0, Timer 2 has a programmable output frequency range from 45.7Hz to 3MHz.
- (3) For SYSCLK=12MHz & T2X12=1, Timer 2 has a programmable output frequency range from 91.5Hz to 6MHz.

Figure 15–10. Timer 2 in Clock-Out Mode



### How to Program Timer 2 in Clock-out Mode

- Select T2X12 bit in T2MOD register to decide the Timer 2 clock source.
- Set T2OE bit in T2MOD register.
- Clear C/T2 bit in T2CON register.
- 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 TR2 bit in T2CON register to start the Timer 2.

In the Clock-Out mode, Timer 2 rollovers will not generate an 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.

### 15.2.5. 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	TCLK	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. When Timer 2 interrupt is enabled, EXF2=1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 does not cause an interrupt in up/down mode (DCEN = 1).

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.

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.

Bit 3: EXEN2, Timer 2 external enable flag.

0: Cause Timer 2 to ignore events at T2EX pin.

1: Allows a capture or reload to occur as a result of a negative 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.

0: Stop the Timer 2.

1: Start the Timer 2.

Bit 1: C/T2, Timer or counter selector.

0: Select Timer 2 as internal timer function.

1: Select Timer 2 as external event counter (falling edge triggered).

Bit 0: CP/-RL2, Capture/Reload flag.

0: Auto-reloads will occur either with Timer 2 overflows or negative transitions at T2EX pin when EXEN2=1.

1: Captures will occur on negative transitions at T2EX pin if EXEN2=1.

When either RCLK=1 or TCLK=1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

#### **T2MOD: Timer 2 Mode Register**

SFR Page = 0 Only

SFR Address = 0xC9

RESET= XX00-XX00

7	6	5	4	3	2	1	0
--	--	T2EXH	T2X12	--	--	T2OE	DCEN2
W	W	R/W	R/W	W	W	R/W	R/W

Bit 7-6: Reserved. Software must write "0" on these bits when T2MOD is written.

Bit 5: T2EXH, Detecting T2EX input transition to High enable.



**RCAP2H: Timer 2 Capture High byte Register**

SFR Page = 0 Only

SFR Address = 0xCB

POR+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

**15.3. Timer 3**

Timer 3 is a 16-bit Timer/Counter which can operate either as a timer or an event counter, as selected by C/T3 in T3CON register. Timer 3 may operate in 16-bit auto-reload mode, (split) 8-bit auto-reload mode and Programmable Clock-Out, which are selected by bits in the T3CON and T3MOD registers.

**15.3.1. 16-bit Timer with Auto-Reload**

In default, Timer 3 operates as a 16-bit timer/counter with auto-reload. As the 16-bit timer register increments and overflows from 0xFFFF to 0x0000, the 16-bit value in the Timer 3 reload registers (RCAP3H and RCAP3L) is loaded into the Timer 3 register as shown in Figure 15–11, and the Timer 3 High Byte Overflow Flag (TF3, T3CON.7) is set. If Timer 3 interrupt is enabled, an interrupt will be generated on each Timer 3 overflow. C/T3=1 sets the Timer 3 as counter mode which clock source comes from T3 pin. In timer mode, T3X12 selects SYSCLK/12 or SYSCLK for Timer 3 clock source. TR3 control the timer running or stopped.

A port change detection on T3EX is built in Timer 3 module. If EXEN3=1 and T3EXH=0, EXF3 is set by a 1-to-0 transition at input T3EX. If EXEN3=1 and T3EXH=1, EXF3 is set by a 0-to-1 transition at input T3EX. If Timer 3 interrupt is enabled, and interrupt will be generate on each time EXF3 is set. When EXEN3 is enabled, software must check the TF3 and EXF3 flags to determine the source of the Timer 3 interrupt. The TF3 and EXF3 interrupt flags are not cleared by hardware and must be manually cleared by software.

In power-down mode, EXF3 is forced to level-sensitive operation and has the capability to wake up CPU if Timer 3 interrupt is enabled. This function provides an additional wake-up CPU path like external interrupt inputs. Detection level of EXF3 is decided by T3EXH. If T3EXH=1, detecting T3EX high level sets EXF3 in power down mode. Otherwise, detecting T3EX low sets EXF3.

Figure 15–11. Timer 3 in 16 bit Auto-Reload Mode (T3SPL=0)

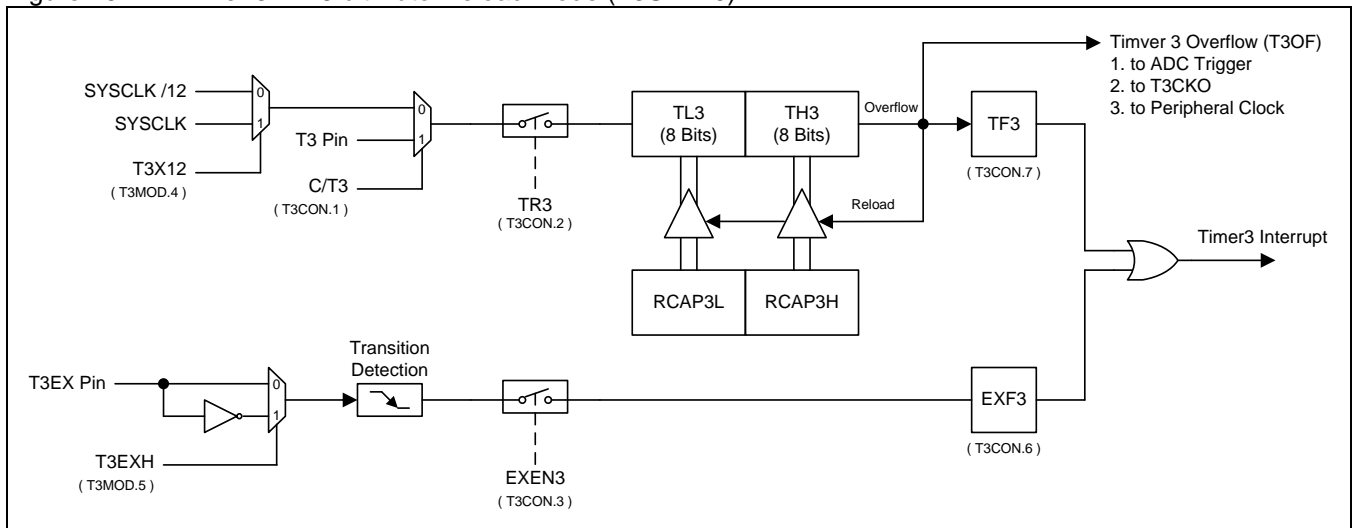
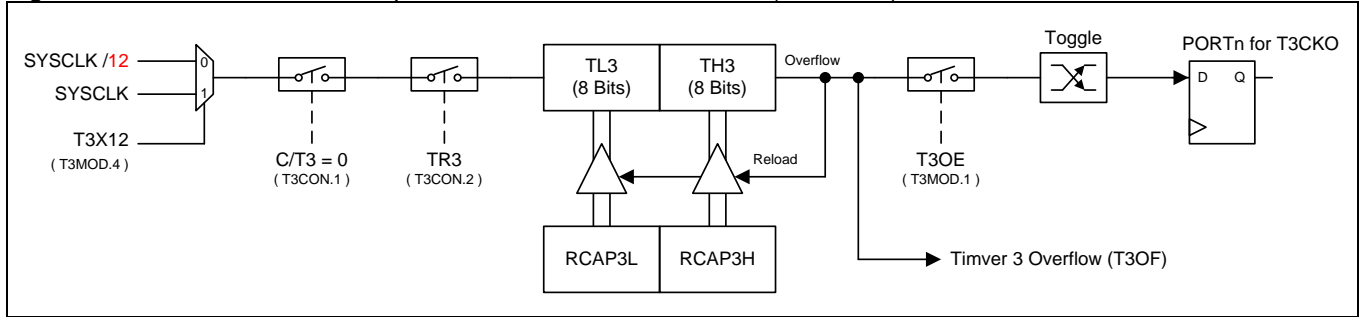




Figure 15–13. Timer 3 Clock Output in 16-bit Auto-Reload Mode (T3SPL=0)



**How to Program 16-bit Timer 3 in Clock-out Mode**

- Select T3X12 bit in T3MOD register to decide the Timer 3 clock source.
- Clear C/T3 bit in T3CON register.
- Determine the 16-bit reload value from the formula and enter it in the RCAP3H and RCAP3L registers.
- Enter the same reload value as the initial value in the TH3 and TL3 registers.
- Set T3OE bit in T3MOD register.
- Set TR3 bit in T3CON register to start the Timer 3.

In the Clock-Out mode, Timer 3 rollovers will not generate an interrupt. However, that the clock-out frequency depends on the overflow rate of Timer 3.

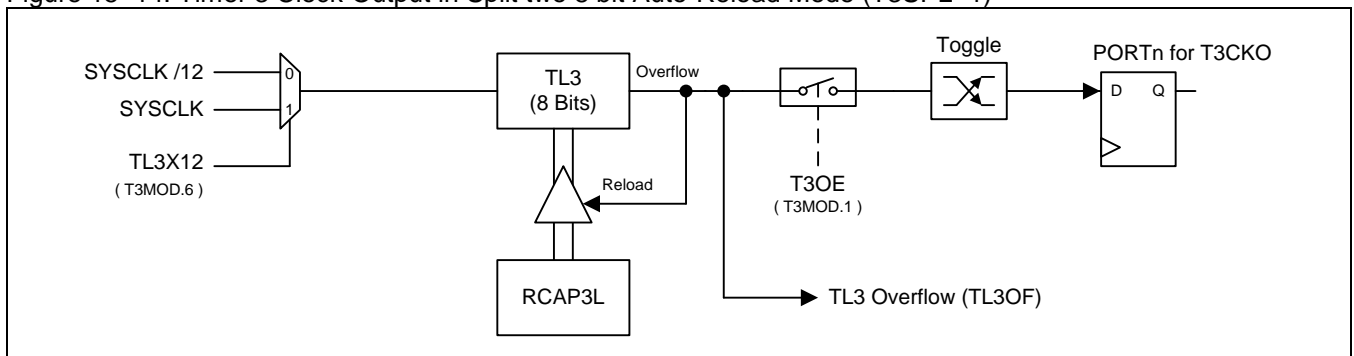
When Timer 3 is configured to split two 8-bit timers, the TL3 overflow triggers the Timer 3 clock output scheme. The input clock (SYSCLK/12 or SYSCLK) increments the 8-bit timer (TL3). The timer repeatedly counts to overflow from a loaded value. Once overflows occur, the content of RCAP3L is loaded in to TL3 for the consecutive counting. Figure 15–14 shows the block diagram for the Timer 3 Clock Output mode. The following formula gives the clock-out frequency.

$\text{T3 Clock-out Frequency} = \frac{\text{SYSCLK Frequency}}{n \times (256 - \text{RCAP3L})}$ <p style="text-align: right; margin: 0;">; n=24, if TL3X12=0 ; n=2, if TL3X12=1</p>
--

**Note:**

- (1) Timer 3 overflow flag, TF3, will not be set by TL3 overflow. TL3 overflow will not influence TF3.
- (2) For SYSCLK=12MHz & TL3X12=0, TL3 has a programmable output frequency range from 1.95KHz to 500KHz.
- (3) For SYSCLK=12MHz & TL3X12=1, TL3 has a programmable output frequency range from 23.43KHz to 6MHz.

Figure 15–14. Timer 3 Clock Output in Split two 8 bit Auto-Reload Mode (T3SPL=1)



**How to Program 8-bit Timer 3 in Clock-out Mode**

- Select TL3X12 bit in T3MOD register to decide the TL3 clock source.
- Determine the 8-bit reload value from the formula and enter it in the RCAP3L registers.
- Enter the same reload value as the initial value in the TL3 registers.
- Set T3OE bit in T3MOD register.
- Set T3SPL bit in T3MOD register to select Timer 3 as split 8-bit mode and start the TL3 timer.

In the Clock-Out mode, Timer 3 rollovers will not generate an interrupt. However, that the clock-out frequency depends on the overflow rate of TL3.

### 15.3.4. Timer 3 Register

#### **T3CON: Timer 3 Control Register**

SFR Page = 1 Only

SFR Address = 0xC8 RESET = 00xx-000x

7	6	5	4	3	2	1	0
TF3	EXF3	--	--	EXEN3	TR3	C/T3	--
R/W	R/W	W	W	R/W	R/W	R/W	W

Bit 7: TF3, Timer 3 overflow flag.

0: TF3 must be cleared by software.

1: TF3 is set by a Timer 3 overflow happens. TF3 will not be set when either RCLK=1 or TCLK=1.

Bit 6: EXF3, Timer 3 external flag.

0: EXF3 must be cleared by software.

1: Timer 3 external flag set when either a capture or reload is caused by a negative transition on T3EX pin and EXEN3=1. When Timer 3 interrupt is enabled, EXF3=1 will cause the CPU to vector to the Timer 3 interrupt routine. EXF3 does not cause an interrupt in up/down mode (DCEN3 = 1).

Bit 5-4: Reserved. Software must write "0" on these bits when T3CON is written.

Bit 3: EXEN3, Timer 3 external enable flag.

0: Cause Timer 3 to ignore events at T3EX pin.

1: Allows a capture or reload to occur as a result of a negative transition on T3EX pin if Timer 3 is not being used to clock the serial port.

Bit 2: TR3, Timer 3 Run control bit.

0: Stop the Timer 3.

1: Start the Timer 3.

Bit 1: C/T3, Timer or counter selector.

0: Select Timer 3 as internal timer function.

1: Select Timer 3 as external event counter (falling edge triggered).

Bit 0: Reserved. Software must write "0" on this bit when T3CON is written.

#### **T3MOD: Timer 3 Mode Register**

SFR Page = 1 Only

SFR Address = 0xC9 RESET= 0000-xx0x

7	6	5	4	3	2	1	0
T3SPL	TL3X12	T3EXH	T3X12	--	--	T3OE	--
R/W	R/W	R/W	R/W	W	W	R/W	W

Bit 7: T3SPL, Timer 3 Split Mode Control.

0: Disable Timer 3 Split Mode. Timer 3 operates as a 16-bit timer in this mode.

1: Enable Timer 3 Split Mode. Timer 3 operates as two 8-bit timers in this mode.

Bit 6: TL3X12, Low byte Timer 3 clock source selector.

0: Select SYSClk/12 as TL3 clock input in Timer 3 split mode.

1: Select SYSClk as TL3 clock input in Timer 3 split mode.

Bit 5: T3EXH, Detecting T3EX input transition to High enable.

0: Detecting T3EX trigger on T3EX falling.

1: Detecting T3EX trigger on T3EX Rising.

Bit 4: T3X12, Timer 3 clock source selector.

0: Select SYSClk/12 as Timer 3 clock source while T3CON.C/T3 = 0.

1: Select SYSClk as Timer 3 clock source while T3CON.C/T3 = 0.



Bit 3~2: Reserved. Software must write “0” on these bits when T3MOD is written.

Bit 1: T3OE, Timer 3 clock-out enable bit.

0: Disable Timer 3 clock output.

1: Enable Timer 3 clock output.

Bit 0: Reserved. Software must write “0” on this bit when T3MOD is written.

Table 15–2 shows the operation modes of Timer 3.

Table 15–2. T3 Mode

TR3	T3OE	T3SPL	Mode
0	0	x	(off)
1	0	0	16-bit auto-reload (TH3 + TL3)
1	1	0	Clock output (TH3 + TL3)
1	0	1	Two 8-bit auto-reload (TH3 and TL3)
1	1	1	8-bit auto-reload (TH3) and clock output (TL3)

**TL3: Timer 3 Low byte Register**

SFR Page = 1 Only

SFR Address = 0xCC POR+RESET = 0000-0000

7	6	5	4	3	2	1	0
TL3.7	TL3.6	TL3.5	TL3.4	TL3.3	TL3.2	TL3.1	TL3.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**TH3: Timer 3 High byte Register**

SFR Page = 1 Only

SFR Address = 0xCD POR+RESET = 0000-0000

7	6	5	4	3	2	1	0
TH3.7	TH3.6	TH3.5	TH3.4	TH3.3	TH3.2	TH3.1	TH3.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**RCAP3L: Timer 3 Capture Low Register**

SFR Page = 1 Only

SFR Address = 0xCA POR+RESET = 0000-0000

7	6	5	4	3	2	1	0
RCAP3L.7	RCAP3L.6	RCAP3L.5	RCAP3L.4	RCAP3L.3	RCAP3L.2	RCAP3L.1	RCAP3L.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

**RCAP3H: Timer 3 Capture High Register**

SFR Page = 1 Only

SFR Address = 0xCB POR+RESET = 0000-0000

7	6	5	4	3	2	1	0
RCAP3H.7	RCAP3H.6	RCAP3H.5	RCAP3H.4	RCAP3H.3	RCAP3H.2	RCAP3H.1	RCAP3H.0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

## 15.4. Timer0/1 Sample Code

(1). Required Function: IDLE mode with T0 wake-up frequency 320Hz, SYSCLK = ILRCO

Assembly Code Example:

```

    ORG    0000Bh
time0_isr:
    to do...
    RETI

main:
    ; (unsigned short value)
    //Switch Sysclk to ILRCO
    MOV    IFADRL,#(CKCON2)    ; Index Page-P address to CKCON2
    CALL   _page_p_sfr_read    ; Read CKCON2 data

    ANL   IFD,#~(OSCS1 | OSCS0) ; Switch OSCin source to ILRCO
    ORL   IFD,#(OSCS1)
    CALL   _page_p_sfr_write    ; Write data to CKCON2

    ANL   IFD,#~(XTALE | IHRCOE) ; Disable XTAL and IHRCOE
    CALL   _page_p_sfr_write    ; Write data to CKCON2

    MOV    IFADRL,#(PCON2)    ; Index Page-P address to PCON2
    CALL   _page_p_sfr_read    ; Read PCON2 data

    ANL   IFD,#~(HSE)         ; Disable HSE when SYSCLK ≤ 6MHz for power saving
    CALL   _page_p_sfr_write    ; Write data to PCON2

    ANL   CKCON0,#(AFS)       ; Select SCKS[2:0] = 0 = OSCin/1

    ORL   AUXR2,#T0X12        ; Select SYSCLK/1 for Timer 0 clock input
    ANL   AUXR0,#~T0XL        ;

    MOV    TH0,#(256-100)     ; Set Timer 0 overflow rate = SYSCLK x 100
    MOV    TL0,#(256-100)     ;
    ANL   TMOD,#(0F0h|T0M1)   ; Set Timer 0 to Mode 2
    ORL   TMOD,#T0M1         ;
    CLR   TF0                 ; Clear Timer 0 Flag

    ORL   IP0L,#PT0L          ; Select Timer 0 interrupt priority
    ORL   IP0H,#PT0H          ;

    SETB  ET0                 ; Enable Timer 0 interrupt
    SETB  EA                   ; Enable global interrupt

    SETB  TR0                 ; Start Timer 0 running

    ORL   PCON0,#IDL          ; Set MCU into IDLE mode

```

C Code Example:

```

void time0_isr(void) interrupt 1
{
    To do...
}

void main(void)
{
    IFADRL = CKCON2;           // Index Page-P address to CKCON2
    page_p_sfr_read();        // Read CKCON2 data.

    IFD = ~(OSCS1 | OSCS0);   // Switch OSCin source to ILRCO
    IFD |= OSCS1;
    page_p_sfr_write();       // Write data to CKCON2

    IFD &= ~(XTALE | IHRCOE); // Disable XTAL and IHRCOE
    page_p_sfr_write();       // Write data to CKCON2
}

```

```

IFADRL = PCON2;           // Index Page-P address to PCON2
page_p_sfr_read();       // Read PCON2 data

IFD &= ~HSE;             // Disable HSE when SYSCLK ≤ 6MHz for power saving
page_p_sfr_write();      // Write data to PCON2

CKCON0 &= AFS;           // Select SCKS[2:0] = 0 = OSCin/1

AUXR2 |= T0X12;          // Select SYSCLK/1 for Timer 0 clock input
AUXR0 &= ~T0XL;

TH0 = TL0 = (256-100);   // Set Timer 0 overflow rate = SYSCLK x 100
TMOD &= 0xF0;           // Set Timer 0 to Mode 2
TMOD |= T0M1;
TF0 = 0;                 // Clear Timer 0 Flag

IP0L |= PT0L;           // Select Timer 0 interrupt priority
IP0H |= PT0H;

ET0 = 1;                 // Enable Timer 0 interrupt
EA = 1;                  // Enable global interrupt

TR0 = 1;                 // Start Timer 0 running

PCON0=IDL;              // Set MCU into IDLE mode
}

```

*(2). Required Function: Set Timer 0 clock output by SYSCLK/48 input*

**Assembly Code Example:**

```

CLR    TR0                ;

ANL    P3M0,#0EFh         ; Set P3.4(T0CKO) to push-pull output
ORL    P3M1,#010h         ;
ORL    AUXR2,#T0CKOE      ; Enable T0CKO

ANL    AUXR2,#~T0X12      ; Select SYSCLK/48 for Timer 0 clock input
ORL    AUXR0,#T0XL        ;

MOV    TH0,#0FFh         ;
MOV    TL0,#0FFh         ;

ANL    TMOD,#0F0h         ; Set Timer 0 to Mode 2
ORL    TMOD,#T0M1        ;

SETB   TR0                ; Start Timer 0 running

```

**C Code Example:**

```

TR0 = 0;

P3M0 &= 0xEF;           // Set P3.4(T0CKO) to push-pull output
P3M1 |= 0x10;
AUXR2 |= T0CKOE;       // Enable T0CKO

AUXR2 &= ~T0X12;       // Select SYSCLK/48 for Timer 0 clock input
AUXR0 |= T0XL;

TH0 = TL0 = 0xFF;

TMOD &= 0xF0;           // Set Timer 0 to Mode 2
TMOD |= T0M1;

TR0 = 1;                // Start Timer 0 running

```

*(3). Required Function: Set Timer 1 clock output by SYSCLK input*

**Assembly Code Example:**

```
ORL    P3M1,#020h           ; Set P3.5(T1CKO) to push-pull output
ANL    P3M0,#0DFh           ;
;
ORL    AUXR2,#(T1X12|T1CKOE) ; Select SYSCLK for Timer 1 clock input
; Enable T1CKO
;
MOV    TH1,#0FFh           ;
MOV    TL1,#0FFh           ;
;
ANL    TMOD,#00Fh           ; Set Timer 1 to Mode 2
ORL    TMOD,#T1M1           ;
;
SETB   TR1                  ; Start Timer 1 running
```

**C Code Example:**

```
P3M1 |= 0x20;                // Set P3.5(T1CKO) to push-pull output
P3M0 &= 0xDF;
;
AUXR2 |= (T1X12|T1CKOE);    // Select SYSCLK for Timer 1 clock input
; Enable T1CKO
;
TH1 = TL1 = 0xFF;
;
TMOD &= 0x0F;                // Set Timer 1 to Mode 2
TMOD |= T1M1;
;
TR1 = 1;                     // Start Timer 1 running
```

## 16. Serial Port 0 (UART0)

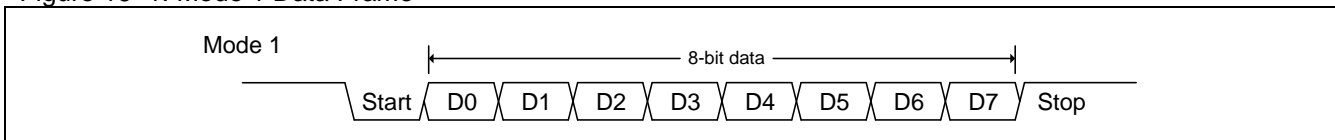
The serial port 0 of **MG84FG516** support full-duplex transmission, meaning it can transmit and receive simultaneously. It is also receive-buffered, meaning it can commence reception of a second byte before a previously received byte has been read from the register. However, if the first byte still 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.

The serial port can operate in 4 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 0:** 8 data bits (LSB first) are transmitted or received through RXD0(P3.0). TXD0(P3.1) always outputs the shift clock. The baud rate can be selected to 1/12 or 1/2 the system clock frequency by URM0X6 setting in S0CFG register.

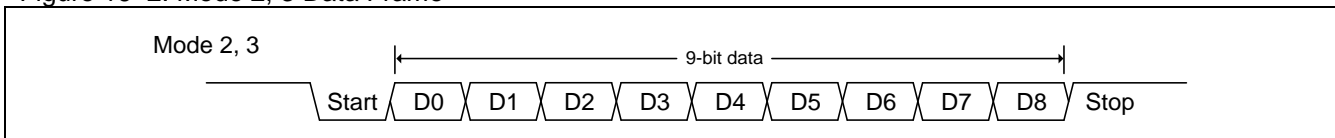
**Mode 1:** 10 bits are transmitted through TXD0 or received through RXD0. The frame data includes a start bit (0), 8 data bits (LSB first), and a stop bit (1), as shown in [Figure 16–1](#). On receive, the stop bit would be loaded into RB80 in S0CON register. The baud rate is variable.

Figure 16–1. 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 16–2](#). 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 16–2. 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.

## 16.1. 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 1/2 the system clock frequency by URM0X6 setting in S0CFG register. Figure 16–3 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, TI would be asserted by hardware to indicate the end of transmission. Figure 16–4 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 P3.1 pin. When Receive is active, the contents on the RXD0(P3.0) 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 16–5 shows the reception waveform in Mode 0.

Receive enables Shift Clock which directly comes from RX Clock to the alternate output function of P3.1 pin. When Receive is active, the contents on the RXD0(P3.0) 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 16–5 shows the reception waveform in Mode 0.

Figure 16–3. Serial Port 0 Mode 0

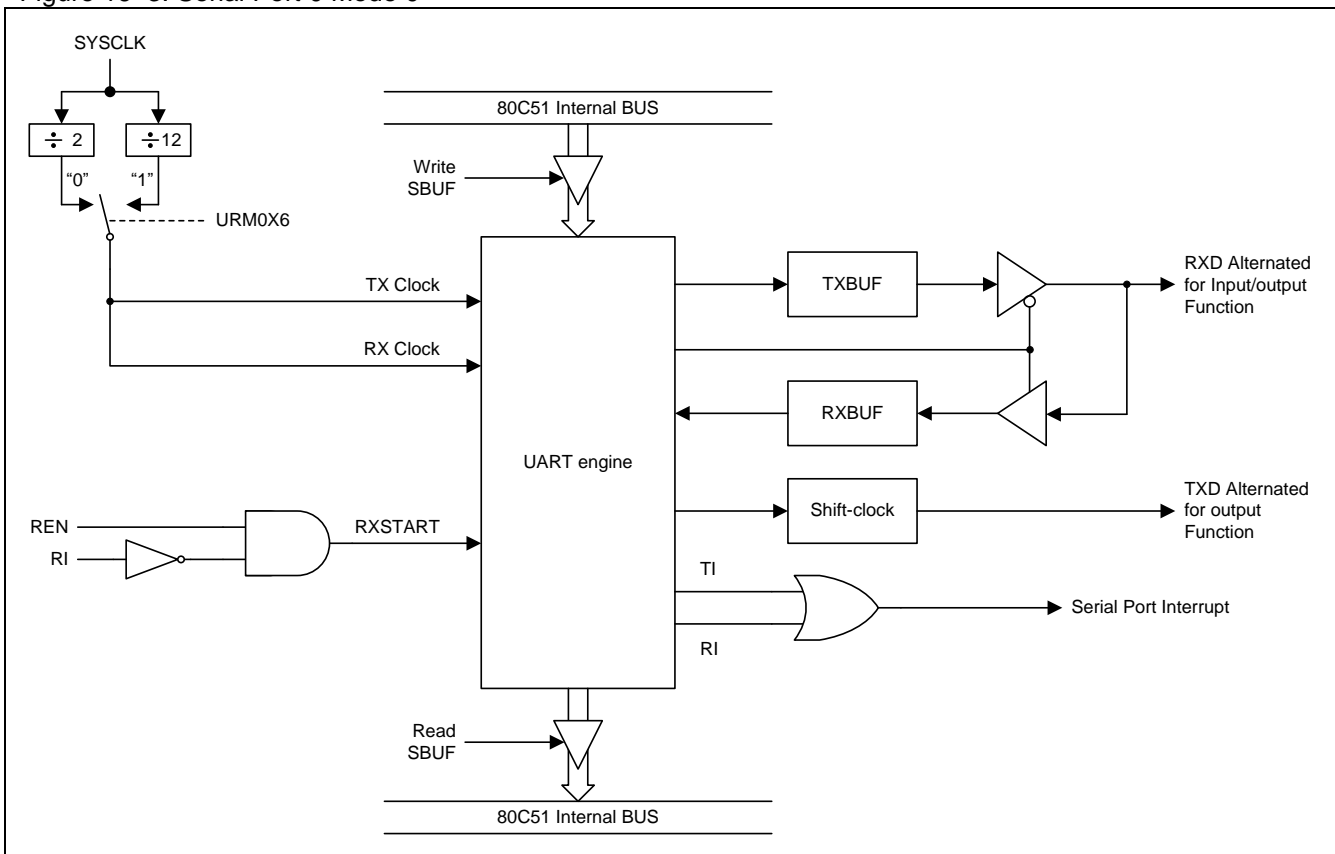


Figure 16–4. Mode 0 Transmission Waveform

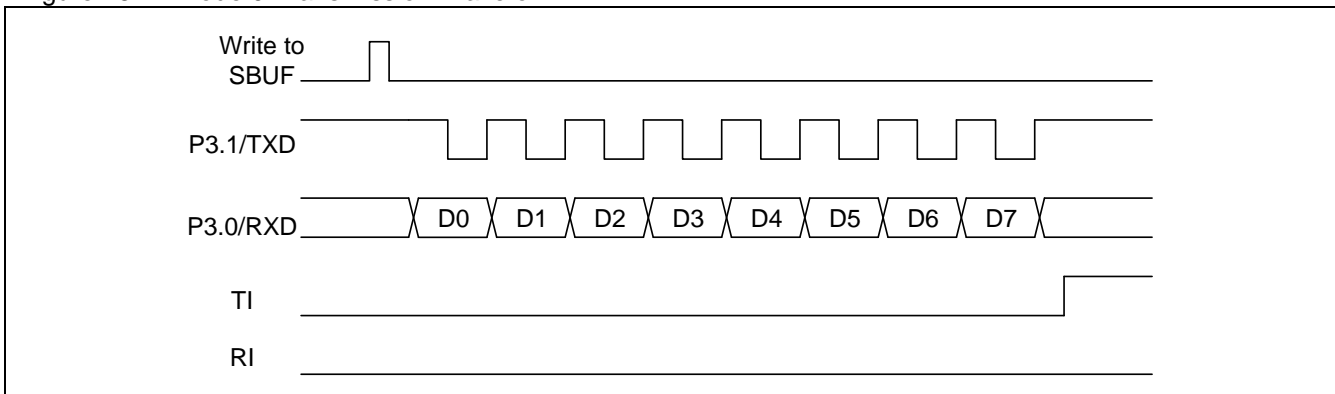
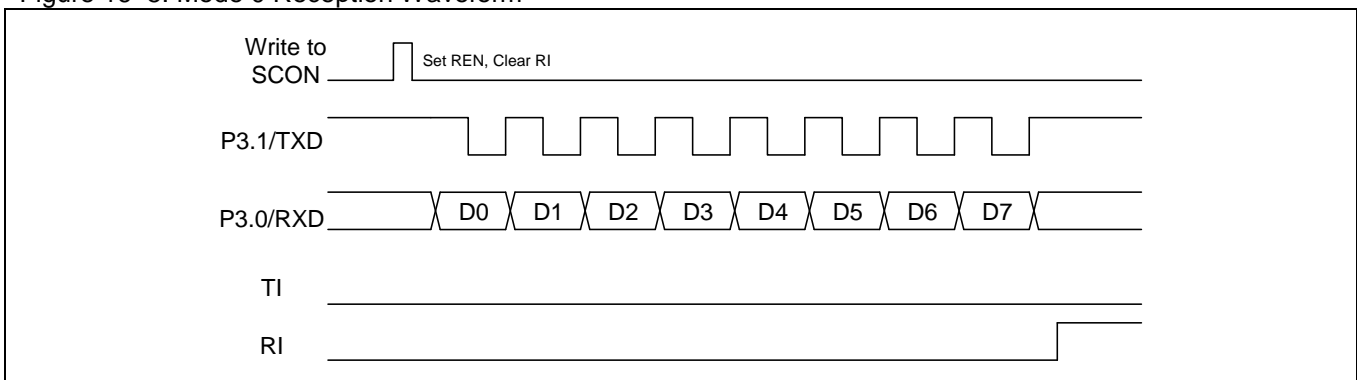


Figure 16–5. Mode 0 Reception Waveform



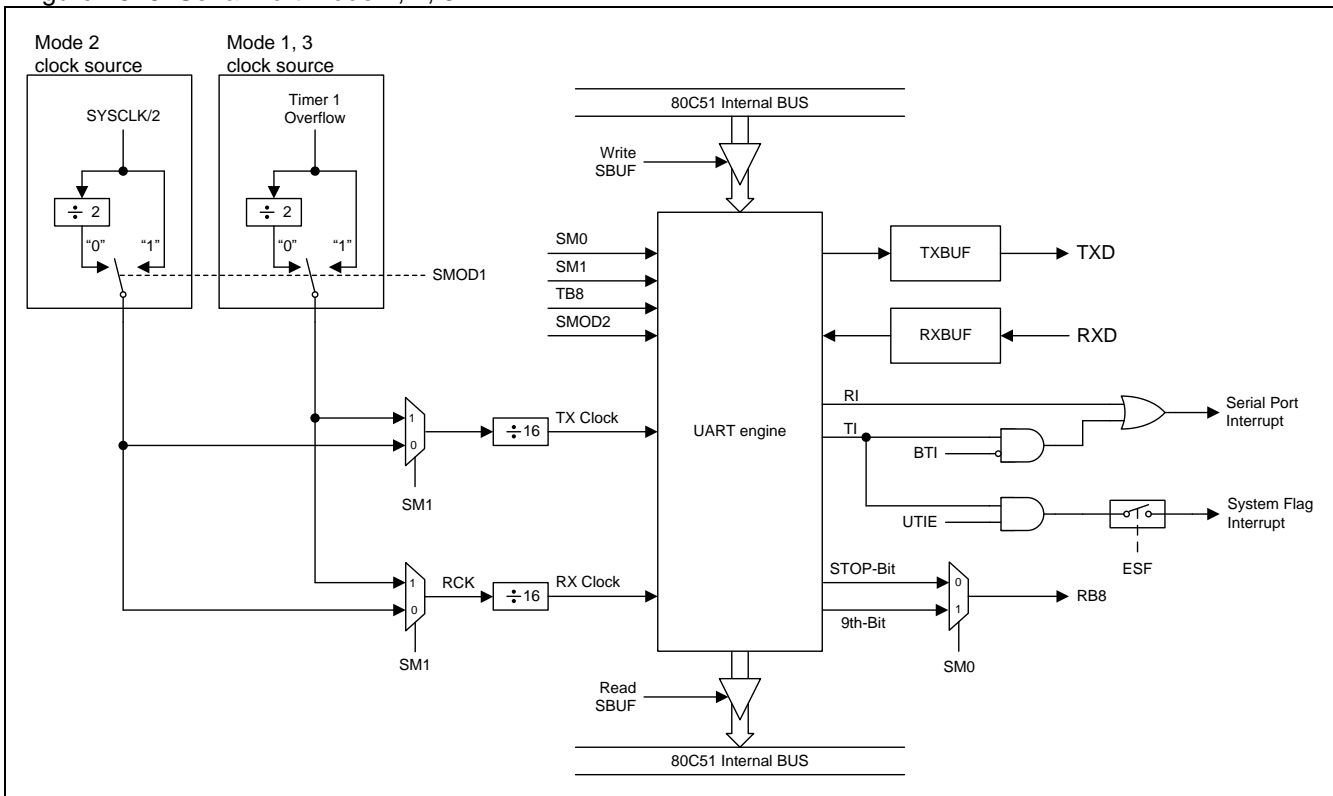
## 16.2. Serial Port 0 Mode 1

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 16–1](#) shows the data frame in Mode 1 and [Figure 16–6](#) 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 16–1](#) 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.

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 16–6. Serial Port Mode 1, 2, 3



### 16.3. 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 16–2 shows the data frame in Mode 2 and Mode 3. Figure 16–5 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 shift 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 16–2 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.

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.

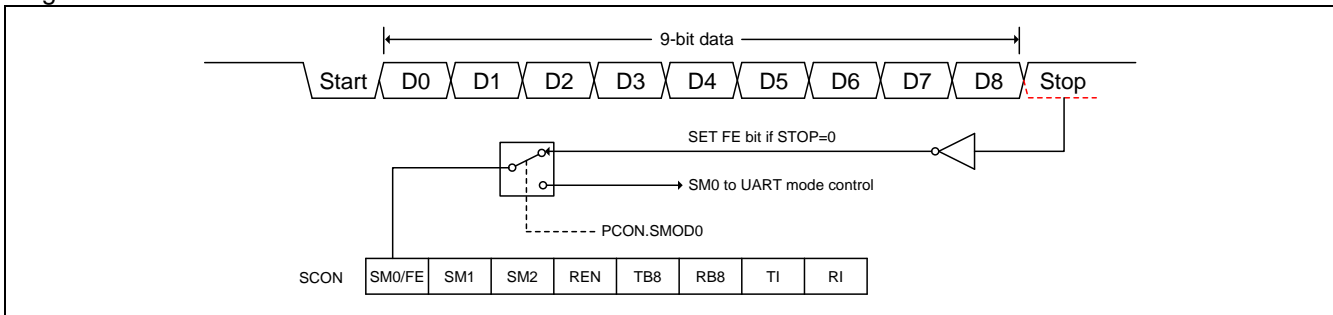
### 16.4. Frame Error Detection

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 16–7](#).

Figure 16–7. UART0 Frame Error Detection



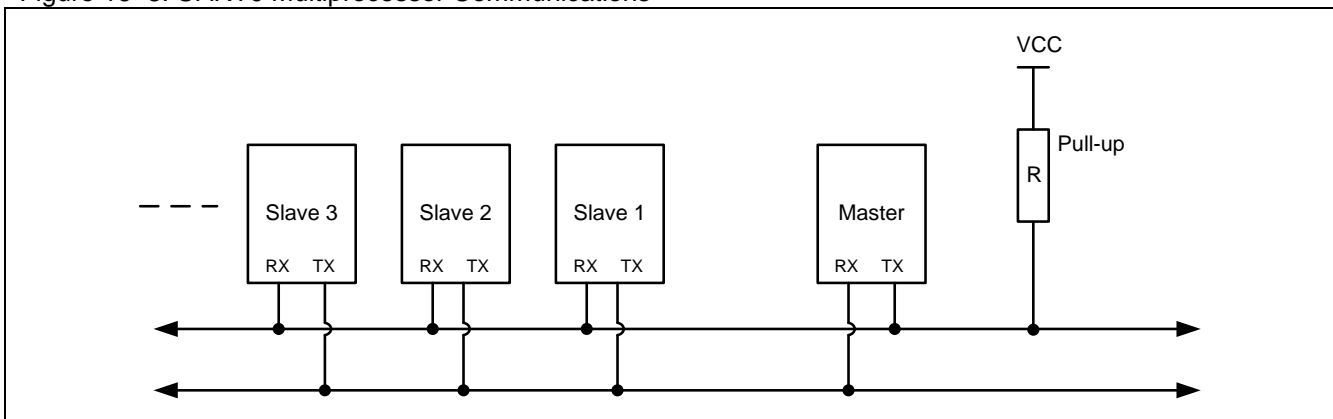
## 16.5. Multiprocessor Communications

Modes 2 and 3 have a special provision for multiprocessor communications as shown in [Figure 16–8](#). 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 16–8. UART0 Multiprocessor Communications



## 16.6. 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 16–9](#). 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 which bits in the SADDR are to be used and which bits are "don't care". The SADEN mask can be logically ANDed with the SADDR to create the "Given" address which the master will use for addressing each of the slaves. Use of the Given address allows multiple slaves to be recognized while excluding others.

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 00X0	Given = 1100 000X

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:

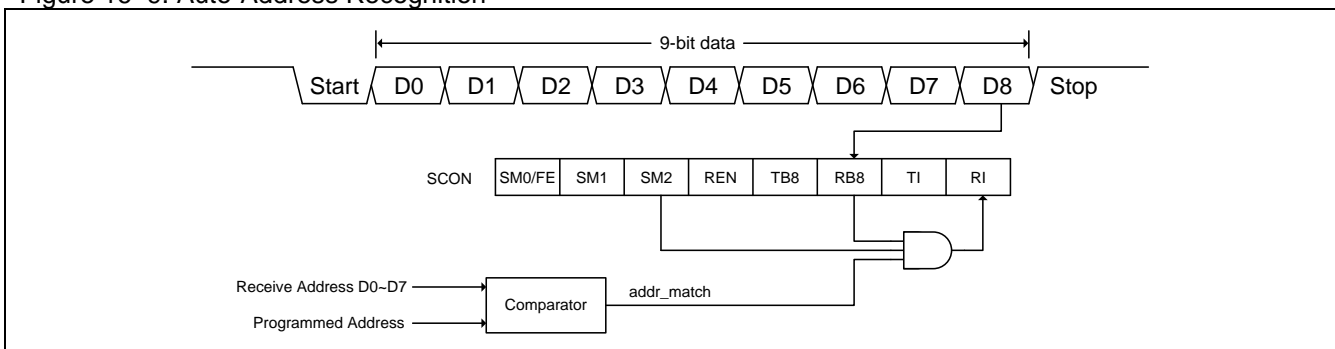
Slave 0	Slave 1	Slave 2
SADDR = 1100 0000	SADDR = 1110 0000	SADDR = 1110 0000
SADEN = 1111 1001	SADEN = 1111 1010	SADEN = 1111 1100
Given = 1100 0XX0	Given = 1110 0X0X	Given = 1110 00XX

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.

Upon reset SADDR (SFR address 0xA9) and SADEN (SFR address 0xB9) are loaded with 0s. This produces a given address of all "don't cares" as well as a Broadcast address of all "don't cares". 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.

Figure 16–9. 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.

## 16.7. Baud Rate Setting

Bits AUXR2.T1X12, URM0X6 and SMOD2 in S0CFG register provide a new option for the baud rate setting, as listed below.

### 16.7.1. Baud Rate in Mode 0

$$\text{Mode 0 Baud Rate} = \frac{F_{\text{SYSCLK}}}{n} \quad ; n=12, \text{ if URM0X6}=0 \\ ; n=2, \text{ if URM0X6}=1$$

Note:

If URM0X6=0, the baud rate formula is as same as standard 8051.

### 16.7.2. Baud Rate in Mode 2

$$\text{Mode 2 Baud Rate} = \frac{2^{\text{SMOD1}} \times 2^{(\text{SMOD2} \times 2)}}{64} \times F_{\text{SYSCLK}}$$

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 16–1 defines the Baud Rate setting with SMOD2 factor in Mode 2 baud rate generator.

Table 16–1. 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 <b>X2</b>	Enhanced function	± 2%
1	1	Double Baud Rate <b>X4</b>	Enhanced function	± 1%

### 16.7.3. Baud Rate in Mode 1 & 3

#### Using Timer 1 as the Baud Rate Generator

$$\text{Mode 1, 3 Baud Rate} = \frac{2^{\text{SMOD1}} \times 2^{(\text{SMOD2} \times 2)}}{32} \times \frac{F_{\text{SYSCLK}}}{12 \times (256 - \text{TH1})} ; \text{T1X12}=0 \\ \text{or} = \frac{2^{\text{SMOD1}} \times 2^{(\text{SMOD2} \times 2)}}{32} \times \frac{F_{\text{SYSCLK}}}{1 \times (256 - \text{TH1})} ; \text{T1X12}=1$$

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 16–2 defines the Baud Rate setting with SMOD2 factor in Timer 1 baud rate generator.

Table 16–2. 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 <b>X2</b>	Enhanced function	± 2%
1	1	Double Baud Rate <b>X4</b>	Enhanced function	± 1%

Table 15-3 ~ Table 15-10 list various commonly used baud rates and how they can be obtained from Timer 1 in its 8-Bit Auto-Reload Mode.

Table 16–3. Timer 1 Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=11.0592\text{MHz}$

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	232	208	0.0%	--	--	--
2400	244	232	0.0%	112	--	0.0%
4800	250	244	0.0%	184	112	0.0%
9600	253	250	0.0%	220	184	0.0%
14400	254	252	0.0%	232	208	0.0%
19200	--	253	0.0%	238	220	0.0%
28800	255	254	0.0%	244	232	0.0%
38400	--	--	--	247	238	0.0%
57600	--	255	0.0%	250	244	0.0%
115200	--	--	--	253	250	0.0%
230400	--	--	--	--	253	0.0%

Table 16–4. Timer 1 Generated High Baud Rates @  $F_{\text{SYSCLK}}=11.0592\text{MHz}$

Baud Rate	TH1, the Reload Value					
	T1X12=0 & SMOD2=1			T1X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
230.4K	--	255	0.0%	250	244	0.0%
460.8K	--	--	--	253	250	0.0%
691.2K	--	--	--	254	252	0.0%
921.6K	--	--	--	--	253	0.0%
1.3824M	--	--	--	255	254	0.0%
2.7648M	--	--	--	--	255	0.0%

Table 16–5. Timer 1 Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=22.1184\text{MHz}$

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	208	160	0.0%	--	--	--
2400	232	208	0.0%	--	--	0.0%
4800	244	232	0.0%	112	--	0.0%
9600	250	244	0.0%	184	112	0.0%
14400	252	248	0.0%	208	160	0.0%
19200	253	250	0.0%	220	184	0.0%
28800	254	252	0.0%	232	208	0.0%
38400	--	253	0.0%	238	220	0.0%
57600	255	254	0.0%	244	232	0.0%
115200	--	255	0.0%	250	244	0.0%
230400	--	--	--	253	250	0.0%
460800	--	--	--	--	253	0.0%

Table 16–6. Timer 1 Generated High Baud Rates @  $F_{\text{SYSCLK}}=22.1184\text{MHz}$

Baud Rate	TH1, the Reload Value					
	T1X12=0 & SMOD2=1			T1X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
460.8K	--	255	0.0%	250	244	0.0%
691.2K	--	--	--	252	248	0.0%
921.6K	--	--	--	253	250	0.0%
1.3824M	--	--	--	254	252	0.0%
1.8432M	--	--	--	--	253	0.0%
2.7648M	--	--	--	255	254	0.0%
5.5296M	--	--	--	--	255	0.0%

Table 16–7. Timer 1 Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=12.0\text{MHz}$

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	230	204	0.16%	--	--	--
2400	243	230	0.16%	100	--	0.16%
4800	--	243	0.16%	178	100	0.16%
9600	--	--	--	217	178	0.16%
14400	--	--	--	230	204	0.16%
19200	--	--	--	--	217	0.16%
28800	--	--	--	243	230	0.16%
38400	--	--	--	246	236	2.34%
57600	--	--	--	--	243	0.16%
115200	--	--	--	--	--	--

Table 16–8. Timer 1 Generated High Baud Rates @  $F_{\text{SYSCLK}}=12.0\text{MHz}$

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	--	--	--	243	230	0.16%
230.4K	--	--	--	--	243	0.16%
460.8K	--	--	--	--	--	--

Table 16–9. Timer 1 Generated Commonly Used Baud Rates @ F<sub>SYSClk</sub>=24.0MHz

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	204	152	0.16%	--	--	--
2400	230	204	0.16%	--	--	--
4800	243	230	0.16%	100	--	0.16%
9600	--	243	0.16%	178	100	0.16%
14400	--	--	--	204	152	0.16%
19200	--	--	--	217	178	0.16%
28800	--	--	--	230	204	0.16%
38400	--	--	--	--	217	0.16%
57600	--	--	--	243	230	0.16%
115200	--	--	--	--	243	0.16%

Table 16–10. Timer 1 Generated High Baud Rates @ F<sub>SYSClk</sub>=24.0MHz

Baud Rate	TH1, the Reload Value					
	T1X12=0 & SMOD2=1			T1X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
230.4K	--	--	--	243	230	0.16%
460.8K	--	--	--	--	243	0.16%
691.2K	--	--	--	--	--	--
921.6K	--	--	--	--	--	--

**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.

$\text{Mode 1, 3 Baud Rate} = \frac{2^{\text{SMOD2} \times (\text{SMOD1} + 1)} \times F_{\text{SYSClk}}}{32 \times (65536 - (\text{RCAP2H}, \text{RCAP2L}))}; \text{T2X12}=0$ <p style="text-align: center;">or =</p> $\frac{2^{\text{SMOD2} \times (\text{SMOD1} + 1)} \times F_{\text{SYSClk}}}{16 \times (65536 - (\text{RCAP2H}, \text{RCAP2L}))}; \text{T2X12}=1$
--

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 16–11 defines the Baud Rate setting with SMOD2 factor in Timer 2 baud rate generator.

Table 16–11. 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 <b>X2</b>	Enhanced function	± 2%

Table 15-12 ~ Table 15-19 list various commonly used baud rates and how they can be obtained from Timer 2 in its Baud-Rate Generator Mode.

Table 16–12. Timer 2 Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=11.0592\text{MHz}$

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=0			T2X12=1 & SMOD2=0		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
1200	65248	65248	0.0%	64960	64960	0.0%
2400	65392	65392	0.0%	65248	65248	0.0%
4800	65464	65464	0.0%	65392	65392	0.0%
9600	65500	65500	0.0%	65464	65464	0.0%
14400	65512	65512	0.0%	65488	65488	0.0%
19200	65518	65518	0.0%	65500	65500	0.0%
28800	65524	65524	0.0%	65512	65512	0.0%
38400	65527	65527	0.0%	65518	65518	0.0%
57600	65530	65530	0.0%	65524	65524	0.0%
115200	65533	65533	0.0%	65530	65530	0.0%
230400	--	--	--	65533	65533	0.0%

Table 16–13. Timer 2 Generated High Baud Rates @  $F_{\text{SYSCLK}}=11.0592\text{MHz}$

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=1			T2X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
230.4K	65533	65530	0.0%	65530	65524	0.0%
460.8K	--	65533	0.0%	65533	65530	0.0%
691.2K	65535	65534	0.0%	65534	65532	0.0%
921.6K	--	--	--	--	65533	0.0%
1.3824M	--	65535	0.0%	65535	65534	0.0%
2.7648M	--	--	--	--	65535	0.0%

Table 16–14. Timer 2 Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=22.1184\text{MHz}$

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=0			T2X12=1 & SMOD2=0		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
1200	64960	64960	0.0%	64384	64384	0.0%
2400	65248	65248	0.0%	64960	64960	0.0%
4800	65392	65392	0.0%	65248	65248	0.0%
9600	65464	65464	0.0%	65392	65392	0.0%
14400	65488	65488	0.0%	65440	65440	0.0%
19200	65500	65500	0.0%	65464	65464	0.0%
28800	65512	65512	0.0%	65488	65488	0.0%
38400	65518	65518	0.0%	65500	65500	0.0%
57600	65524	65524	0.0%	65512	65512	0.0%
115200	65530	65530	0.0%	65524	65524	0.0%
230400	65533	65533	0.0%	65530	65530	0.0%
460800	--	--	--	65533	65533	0.0%

Table 16–15. Timer 2 Generated High Baud Rates @  $F_{\text{SYSCLK}}=22.1184\text{MHz}$

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=1			T2X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
460.8K	65533	65530	0.0%	65530	65524	0.0%
691.2K	65534	65532	0.0%	65532	65528	0.0%
921.6K	--	65533	0.0%	65533	65530	0.0%
1.3824M	65535	65534	0.0%	65534	65532	0.0%
1.8432M	--	--	--	--	65533	0.0%
2.7648M	--	65535	0.0%	65535	65534	0.0%
5.5296M	--	--	--	--	65535	0.0%

Table 16–16. Timer 2 Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=12.0\text{MHz}$

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=0			T2X12=1 & SMOD2=0		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
1200	65224	65224	0.16%	64912	64912	0.16%
2400	65380	65380	0.16%	65224	65224	0.16%
4800	65458	65458	0.16%	65380	65380	0.16%
9600	65497	65497	0.16%	65458	65458	0.16%
14400	65510	65510	0.16%	65484	65484	0.16%
19200	65516	65516	2.34%	65497	65497	0.16%
28800	65523	65523	0.16%	65510	65510	0.16%
38400	--	--	--	65516	65516	2.34%
57600	--	--	--	65523	65523	0.16%
115200	--	--	--	--	--	--

Table 16–17. Timer 2 Generated High Baud Rates @  $F_{\text{SYSCLK}}=12.0\text{MHz}$

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=1			T2X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
115.2K	--	65523	0.16%	65523	65510	0.16%
230.4K	--	--	--	--	65523	0.16%
460.8K	--	--	--	--	--	--



Table 16–18. Timer 2 Generated Commonly Used Baud Rates @ F<sub>SYSCLK</sub>=24.0MHz

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=0			T2X12=1 & SMOD2=0		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
1200	64912	64912	0.16%	64288	64288	0.16%
2400	65224	65224	0.16%	64912	64912	0.16%
4800	65380	65380	0.16%	65224	65224	0.16%
9600	65458	65458	0.16%	65380	65380	0.16%
14400	65484	65484	0.16%	65432	65432	0.16%
19200	65497	65497	0.16%	65458	65458	0.16%
28800	65510	65510	0.16%	65484	65484	0.16%
38400	65516	65516	2.34%	65497	65497	0.16%
57600	65523	65523	0.16%	65510	65510	0.16%
115200	--	--	--	65523	65523	0.16%

Table 16–19. Timer 2 Generated High Baud Rates @ F<sub>SYSCLK</sub>=24.0MHz

Baud Rate	[RCAP2H, RCAP2L], the Reload Value					
	T2X12=0 & SMOD2=1			T2X12=1 & SMOD2=1		
	SMOD1=0	SMOD1=1	Error	SMOD1=0	SMOD1=1	Error
230.4K	--	65523	0.16%	65523	65510	0.16%
460.8K	--	--	--	--	65523	0.16%
691.2K	--	--	--	--	--	--
921.6K	--	--	--	--	--	--

### Using S1 Baud Rate Timer as the Baud Rate Generator

The secondary UART (S1) in **MG84FG516** has an independent baud-rate generator. S0 can set URTS (SOCFG.7) to select the S1BRT as the timer source for UART Mode 1 and Mode 3. See Section “17.5 S1 Baud Rate Timer for S0” on for details for the S0 baud rate select.

## 16.8. 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.

SM00	SM10	Mode	Description	Baud Rate
0	0	0	shift register	SYSCCLK/12 or /2
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	SYSCCLK/64, /32, /16 or /8
1	1	3	9-bit UART	variable

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 9<sup>th</sup> 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 9<sup>th</sup> 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 8<sup>th</sup> bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission.

Bit 0: RI0. Receive interrupt flag.

0: Must be cleared by software.

1: Set by hardware at the end of the 8<sup>th</sup> 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
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

#### **SADEN: Slave Address Mask Register**

SFR Page = 0~F

SFR Address = 0xB9

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

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

SFR Address = 0x87

POR = 00X1-0000, RESET = 00X0-0000

7	6	5	4	3	2	1	0
SMOD1	SMOD0	--	POF	GF1	GF0	PD	IDL
R/W	R/W	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 = 0x9A

RESET = 0000-XX00

7	6	5	4	3	2	1	0
URTS	SMOD2	URM0X6	--	--	--	--	--
R/W	R/W	R/W	W	W	W	W	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 “16.7.3 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: URM0X6, Serial Port mode 0 baud rate selector.

0: Clear to select SYSCLK/12 as the baud rate for UART Mode 0.

1: Set to select SYSCLK/2 as the baud rate for UART Mode 0.

### AUXR2: Auxiliary Register 2

SFR Page = 0~F

SFR Address = 0xA3

RESET = 0000-0000

7	6	5	4	3	2	1	0
INT3IS1	INT3IS0	INT2IS1	INT2IS0	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	R/W	R/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.

## 17. Serial Port 1 (UART1)

The **MG84FG516** is equipped with a secondary UART (hereafter, called UART1), which also has four operation modes the same as the first UART except the following differences:

- (1) The UART1 has no enhanced functions: Framing Error Detection and Auto Address Recognition.
- (2) The UART1 use the dedicated Baud Rate Timer as its Baud Rate Generator (S1BRG).
- (3) The UART1 uses port pin **P1.3** (TXD1) and **P1.2** (RXD1) for transmit and receive, respectively.
- (4) The Baud Rate Generator provide the toggle source for S1CKO and peripheral clock.
- (5) S1 + S1BRG can be configured to an 8-bit auto-reload timer with port change detection.

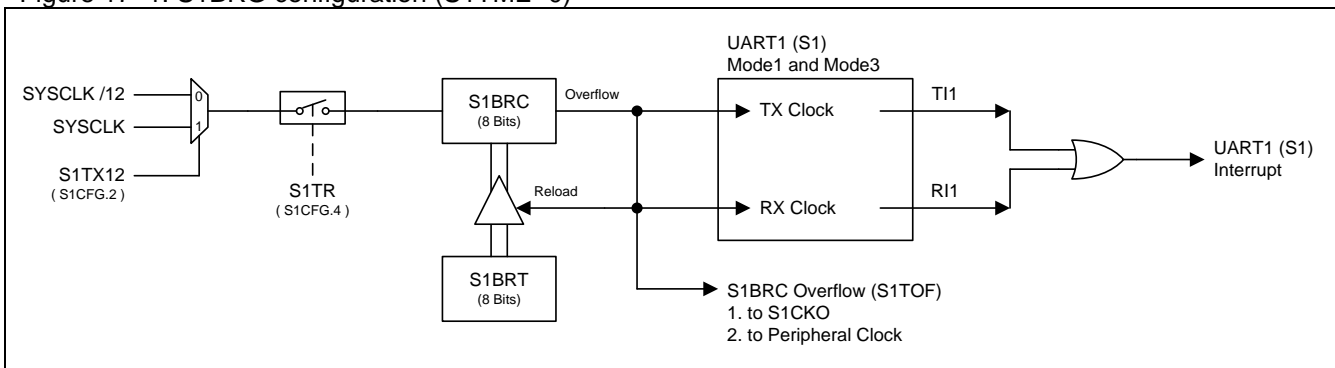
The UART1 and UART0 in **MG84FG516** can operate simultaneously in identical or different modes and communication speeds.

### 17.1. Serial Port 1 Baud Rate Generator (S1BRG)

The **MG84FG516** has an embedded Baud Rate Generator to generate the UART clock for serial port 1 operation in mode 1 and mode 3. This baud rate generator can also provide the time base for serial port 0 by software configured. There is an addition clock output, S1CKO, from the S1BRC overflow rate by 2 (S1TOF/2). S1TOF also supplies the toggle source for ADC, SPI, TWSI clock input.

The configuration of the Serial Port 1 Baud Rate Generator is shown in [Figure 17–1](#).

Figure 17–1. S1BRG configuration (S1TME=0)



### 17.2. Serial Port 1 Baud Rate Setting

#### 17.2.1. Baud Rate in Mode 0

$$\text{S1 Mode 0 Baud Rate} = \frac{F_{\text{SYSCLK}}}{12}$$

#### 17.2.2. Baud Rate in Mode 2

$$\text{S1 Mode 2 Baud Rate} = \frac{2^{\text{S1MOD1}}}{64} \times F_{\text{SYSCLK}}$$

### 17.2.3. Baud Rate in Mode 1 & 3

$$\text{S1 Mode 1, 3 Baud Rate} = \frac{2^{\text{S1MOD1}}}{32} \times \frac{F_{\text{SYSCLK}}}{12 \times (256 - \text{S1BRT})} ; \text{S1TX12}=0$$

$$\text{or} = \frac{2^{\text{S1MOD1}}}{32} \times \frac{F_{\text{SYSCLK}}}{1 \times (256 - \text{S1BRT})} ; \text{S1TX12}=1$$

Table 16-1 ~ Table 16-4 list various commonly used baud rates and how they can be obtained from S1BRG, serial port 1 baud rate generator.

Table 17–1. S1BRG Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=11.0592\text{MHz}$

Baud Rate	S1BRT, the Reload Value					
	S1TX12=0			S1TX12=1		
	S1MOD1=0	S1MOD1=1	Error	S1MOD1=0	S1MOD1=1	Error
1200	232	208	0.0%	--	--	--
2400	244	232	0.0%	112	--	0.0%
4800	250	244	0.0%	184	112	0.0%
9600	253	250	0.0%	220	184	0.0%
14400	254	252	0.0%	232	208	0.0%
19200	--	253	0.0%	238	220	0.0%
28800	255	254	0.0%	244	232	0.0%
38400	--	--	--	247	238	0.0%
57600	--	255	0.0%	250	244	0.0%
115200	--	--	--	253	250	0.0%
230400	--	--	--	--	253	0.0%

Table 17–2. S1BRG Generated Commonly Used Baud Rates @  $F_{\text{SYSCLK}}=22.1184\text{MHz}$

Baud Rate	S1BRT, the Reload Value					
	S1TX12=0			S1TX12=1		
	S1MOD1=0	S1MOD1=1	Error	S1MOD1=0	S1MOD1=1	Error
1200	208	160	0.0%	--	--	--
2400	232	208	0.0%	--	--	0.0%
4800	244	232	0.0%	112	--	0.0%
9600	250	244	0.0%	184	112	0.0%
14400	252	248	0.0%	208	160	0.0%
19200	253	250	0.0%	220	184	0.0%
28800	254	252	0.0%	232	208	0.0%
38400	--	253	0.0%	238	220	0.0%
57600	255	254	0.0%	244	232	0.0%
115200	--	255	0.0%	250	244	0.0%
230400	--	--	--	253	250	0.0%
460800	--	--	--	--	253	0.0%

Table 17–3. S1BRG Generated Commonly Used Baud Rates @ F<sub>SYSClk</sub>=12.0MHz

Baud Rate	S1BRT, the Reload Value					
	S1TX12=0			S1TX12=1		
	S1MOD1=0	S1MOD1=1	Error	S1MOD1=0	S1MOD1=1	Error
1200	230	204	0.16%	--	--	--
2400	243	230	0.16%	100	--	0.16%
4800	--	243	0.16%	178	100	0.16%
9600	--	--	--	217	178	0.16%
14400	--	--	--	230	204	0.16%
19200	--	--	--	--	217	0.16%
28800	--	--	--	243	230	0.16%
38400	--	--	--	246	236	2.34%
57600	--	--	--	--	243	0.16%
115200	--	--	--	--	--	--

Table 17–4. S1BRG Generated Commonly Used Baud Rates @ F<sub>SYSClk</sub>=24.0MHz

Baud Rate	S1BRT, the Reload Value					
	S1TX12=0			S1TX12=1		
	S1MOD1=0	S1MOD1=1	Error	S1MOD1=0	S1MOD1=1	Error
1200	204	152	0.16%	--	--	--
2400	230	204	0.16%	--	--	--
4800	243	230	0.16%	100	--	0.16%
9600	--	243	0.16%	178	100	0.16%
14400	--	--	--	204	152	0.16%
19200	--	--	--	217	178	0.16%
28800	--	--	--	230	204	0.16%
38400	--	--	--	--	217	0.16%
57600	--	--	--	243	230	0.16%
115200	--	--	--	--	243	0.16%

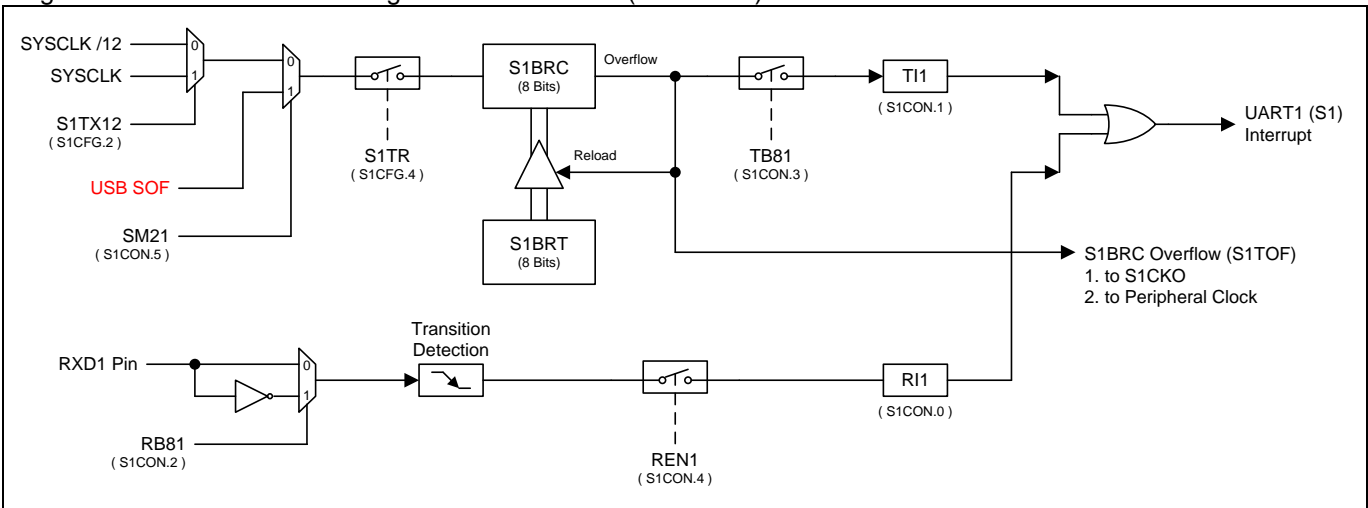
### 17.3. Pure Timer Mode for S1BRG

If the UART1 is not necessary or pending by software, setting S1TME=1 in the **MG84FG516** provides the pure timer operating mode on S1 Baud Rate Generator (S1BRG). This timer operates as an 8-bit auto-reload timer and provides the overflow flag which is set on the TI1 of S1CON.1. The RI1 of S1CON.0 serves the port change detector on RXD1 port pin. Both of TI1 and RI1 in this mode keep the interrupt capability on UART1 interrupt resource and have the individual interrupt enabled control (TB81 & REN1). RB81 selects the RI1 detection level on RXD1 port input. If RB81=0, RI1 will be set by REN1=1 and RXD1 pin falling edge detecting. Otherwise, RI1 will detect the rising edge on RXD1 port pin. In MCU power-down mode, the RI1 is forced to level-sensitive operation and has the capability to wake up CPU if UART1 interrupt is enabled.

This timer provides additional clock input from USB engine detecting host SOF token, normally, the frequency of SOF is around 1ms. The clock source function for ADC, SPI, TW5I or output on port pin is also valid in this mode. S1CKOE=1 enables the S1CKO output on port pin and masks the RI1 interrupt.

The configuration of the Pure Timer mode of S1BRG is shown in [Figure 17–2](#).

Figure 17–2. Timer Mode Configuration for S1BRG (S1TME=1)



### 17.4. S1BRT Programmable Clock Output

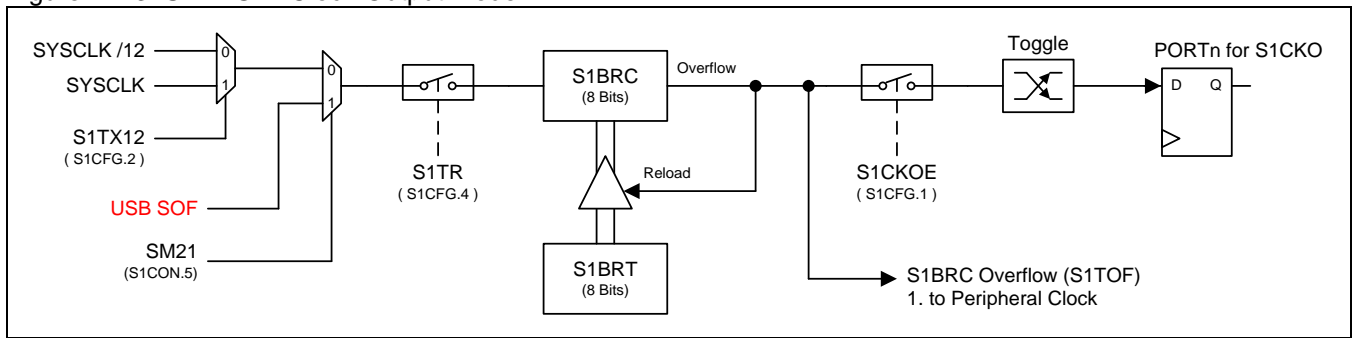
When S1BRC overflows, the overflow flag, S1TOF, provides the toggle source for S1CKO and peripheral clock. The input clock (SYSCLK/12 or SYSCLK) increments the 8-bit timer (S1BRC). The timer repeatedly counts to overflow from a loaded value. Once overflows occur, the content of S1BRT is loaded in to S1BRC for the consecutive counting. [Figure 17–3](#) shows the block diagram for the Clock Output mode of S1 Baud Rate Generator. The following formula gives the clock-out frequency.

$\text{S1T Clock-out Frequency} = \frac{\text{SYSCLK Frequency}}{n \times (256 - \text{S1BRT})}$	; n=24, if S1TX12=0 ; n=2, if S1TX12=1
--	---

Note:

- (1) For SYSCLK=12MHz & S1TX12=0, S1BRG has a programmable output frequency range from 1.95KHz to 500KHz.
- (2) For SYSCLK=12MHz & S1TX12=1, S1BRG has a programmable output frequency range from 23.43KHz to 6MHz.

Figure 17–3. S1BRG in Clock Output Mode



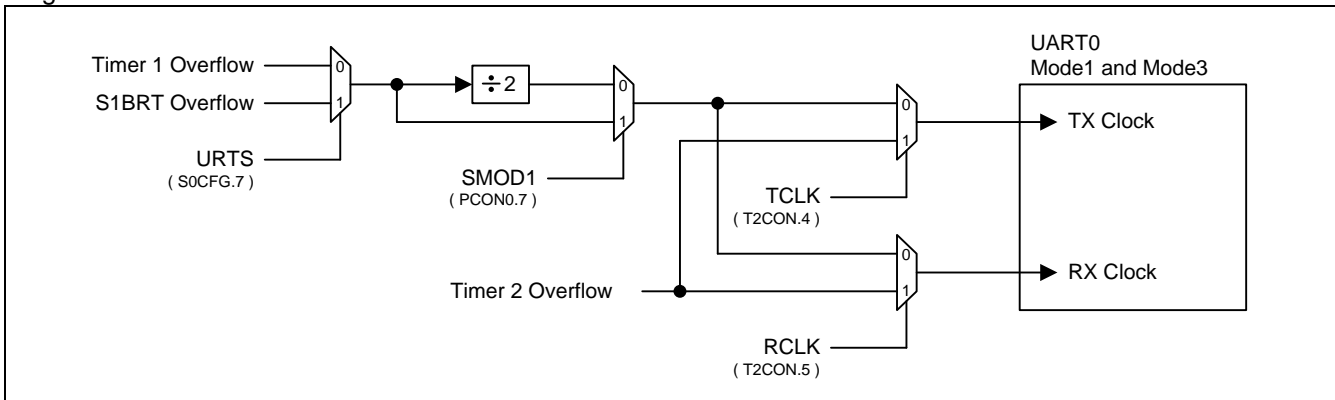
**How to Program 8-bit S1BRG in Clock-out Mode**

- Select S1CFG.S1TX12 bit and S1CON.SM21 bit to decide the S1BRG clock source.
- Determine the 8-bit reload value from the formula and enter it in the S1BRT registers.
- Set S1CKOE bit in S1CFG register.
- Set S1TR to start the S1 BRC timer.

**17.5. S1 Baud Rate Timer for 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. At this time, if URTS bit (in S0CFG register) is set, then Timer 1 overflow signal will be replaced by the overflow signal of the UART1 Baud Rate Timer. In other words, the user can adopt UART1 Baud Rate Timer 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 17–4. Additional Baud Rate Source for the UART0





## 17.6. Serial Port 1 Register

The following special function registers are related to the operation of the UART1:

### **S1CON: Serial port 1 Control Register**

SFR Page = 1 only

SFR Address = 0x98

RESET = 0000-0000

7	6	5	4	3	2	1	0
SM01	SM11	SM21	REN1	TB81	RB81	TI1	RI1
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: SM01, Serial port 1 mode bit 0.

Bit 6: SM11, Serial port 1 mode bit 1.

SM01	SM11	Mode	Description	Baud Rate
0	0	0	shift register	SYSCLK/12
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	SYSCLK/64, /32
1	1	3	9-bit UART	variable

Bit 5: Serial port 0 mode bit 2.

0: Disable SM21 function.

1: Enable the automatic address recognition feature in Modes 2 and 3. If SM21=1, RI1 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 SM21=1 then RI1 will not be set unless a valid stop Bit was received, and the received byte is a Given or Broadcast address. In Mode 0, SM21 should be 0.

Bit 4: REN1, Enable serial reception.

0: Clear by software to disable reception.

1: Set by software to enable reception.

Bit 3: TB81, The 9<sup>th</sup> data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.

Bit 2: RB81, In Modes 2 and 3, the 9<sup>th</sup> data bit that was received. In Mode 1, if SM21 = 0, RB81 is the stop bit that was received. In Mode 0, RB81 is not used.

Bit 1: TI1. Transmit interrupt flag.

0: Must be cleared by software.

1: Set by hardware at the end of the 8<sup>th</sup> bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission.

Bit 0: RI1. Receive interrupt flag.

0: Must be cleared by software.

1: Set by hardware at the end of the 8<sup>th</sup> bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM21).

### **S1BUF: Serial port 1 Buffer Register**

SFR Page = 1 only

SFR Address = 0x99

RESET = XXXX-XXXX

7	6	5	4	3	2	1	0
S1BUF.7	S1BUF.6	S1BUF.5	S1BUF.4	S1BUF.3	S1BUF.2	S1BUF.1	S1BUF.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.

**S1BRT: Serial port 1 Baud Rate Timer Reload Register**

SFR Page = 1 only

SFR Address = 0x9A RESET = 0000-0000

7	6	5	4	3	2	1	0
S1BRT.7	S1BRT.6	S1BRT.5	S1BRT.4	S1BRT.3	S1BRT.2	S1BRT.1	S1BRT.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.

**S1BRC: Serial port 1 Baud Rate Counter Register**

SFR Page = 1 only

SFR Address = 0x9B RESET = 0000-0000

7	6	5	4	3	2	1	0
S1BRC.7	S1BRC.6	S1BRC.5	S1BRC.4	S1BRC.3	S1BRC.2	S1BRC.1	S1BRC.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 S1CFG.S1TME = 0, software writing S1BRT will store the data content to S1BRT and S1BRC concurrently.

**S1CFG: Serial Port 1 Configuration Register**

SFR Page = 0 only

SFR Address = 0x9B RESET = xxx0-0000

7	6	5	4	3	2	1	0
--	--	--	S1TR	S1MOD1	S1TX12	S1CKOE	S1TME
W	W	W	R/W	R/W	R/W	R/W	R/W

Bit 7~5: Reserved. Software must write "0" on these bits when S1CFG is written.

Bit 4: S1TR, UART1 Baud Rate Timer control bit.

0: Clear to turn off the S1BRT.

1: Set to turn on S1BRT.

Bit 3: S1MOD1, UART1 double baud rate enable bit.

0: Disable the double baud rate function for UART1.

1: Enable the double baud rate function for UART1.

Bit 2: S1TX12, UART1 Baud Rate Timer clock source select

0: Clear to select SYSCLK/12 as the clock source for S1BRT.

1: Set to select SYSCLK as the clock source for S1BRT.

Bit 1: S1CKOE, Serial Port 1 BRT Clock Output Enable.

0: Disable the S1CKO output on the port pin.

1: Enable the S1CKO output on the port pin

Bit 0: S1TME, Serial port 1 BRT Timer Mode Enabled.

0: Keep S1BRT to service Serial Port 1 (UART1).

1: Disable Serial Port 1 function and release the S1BRT as an 8-bit auto-reload timer. In this mode, there is an additional function for RXD1 port pin change detector.

## 17.7. Serial Port Sample Code

### (1). Required Function: IDLE mode with RI wake-up capability

#### Assembly Code Example:

```
    ORG    00023h
uart_ri_idle_isr:
    JB     RI,RI_ISR           ;
    JB     TI,TI_ISR          ;
    RETI                          ;

RI_ISR:
; Process
    CLR    RI                 ;
    RETI                          ;

TI_ISR:
; Process
    CLR    TI                 ;
    RETI                          ;

main:
    CLR    TI                 ;
    CLR    RI                 ;
    SETB   SM1                ;
    SETB   REN                ; 8bit Mode2, Receive Enable

    MOV    IP0L,#PSL          ; Select S0 interrupt priority
    MOV    IP0H,#PSH          ;

    SETB   ES                 ; Enable S0 interrupt
    SETB   EA                 ; Enable global interrupt

    ORL    PCON0,#IDL;        ; Set MCU into IDLE mode
```

#### C Code Example:

```
void uart_ri_idle_isr(void) interrupt 4
{
    if(RI)
    {
        RI=0;
        // to do ...
    }

    if(TI)
    {
        TI=0;
        // to do ...
    }
}

void main(void)
{
    TI = RI = 0;
    SM1 = REN = 1;                // 8bit Mode2, Receive Enable

    IP0L = PSL;                   // Select S0 interrupt priority
    IP0H = PSH;                   //

    ES = 1;                       // Enable S0 interrupt
    EA = 1;                       // Enable global interrupt

    PCON |= IDL;                 // Set MCU into IDLE mode
}
```

## 18. Programmable Counter Array (PCA)

The **MG84FG516** is equipped with a Programmable Counter Array (PCA), which provides more timing capabilities with less CPU intervention than the standard timer/counters. Its advantages include reduced software overhead and improved accuracy.

### 18.1. PCA Overview

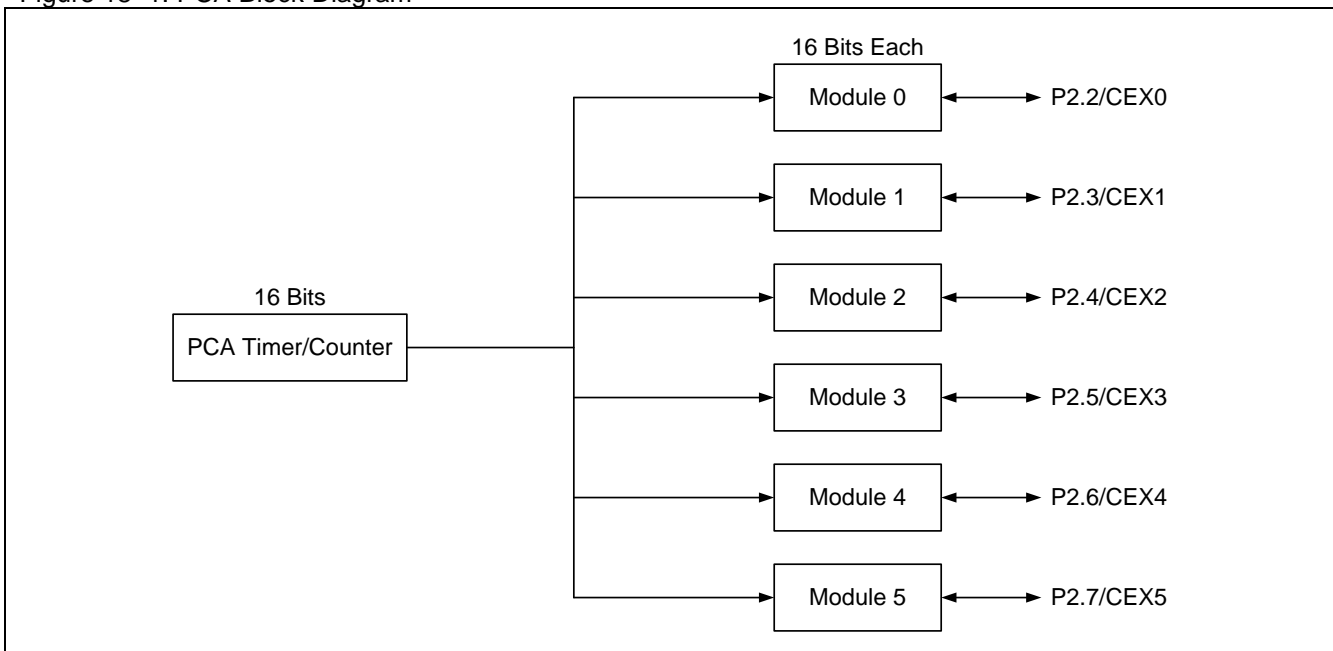
The PCA consists of a dedicated timer/counter which serves as the time base for an array of six compare/capture modules. [Figure 18–1](#) shows a block diagram of the PCA. Notice that the PCA timer and modules are all 16-bits. If an external event is associated with a module, that function is shared with the corresponding Port 2 pin. If the module is not using the port pin, the pin can still be used for standard I/O.

Each of the six modules can be programmed in any one of the following modes:

- Rising and/or Falling Edge Capture
- Software Timer
- High Speed Output
- Pulse Width Modulator (PWM) Output

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 18–1. PCA Block Diagram



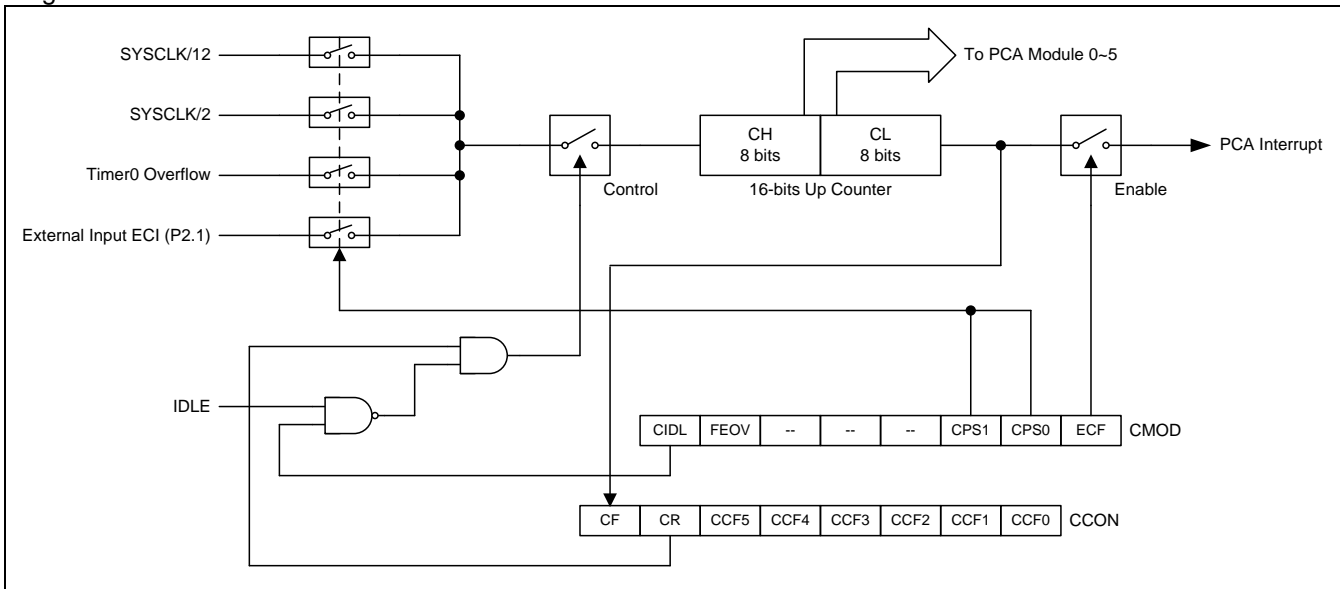
## 18.2. PCA Timer/Counter

The timer/counter for the PCA is a free-running 16-bit timer consisting of registers CH and CL (the high and low bytes of the count values), as shown in Figure 18–2. It 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 (P2.1).

Special Function Register CMOD contains the Count Pulse Select bits (CPS1 and CPS0) to specify the PCA timer input. This register also contains the ECF bit which enables an interrupt when the counter overflows. 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 18–2. PCA Timer/Counter



### CMOD: PCA Counter Mode Register

SFR Page = All

SFR Address = 0xD9

RESET = 00xx-x000

7	6	5	4	3	2	1	0
CIDL	FEOV	--	--	--	CPS1	CPS0	ECF
R/W	R/W	W	W	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 6: FEOV: CL overflows on FEH enabled.

0: CL overflows on FFH.

1: CL overflows on FEH.

Bit 5–3: Reserved. Software must write “0” on these bits when CMOD is written.

Bit 2–1: CPS1-CPS0, PCA counter clock source select bits.

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

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 CCF5 are the interrupt flags for module 0 to module 5, 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 18–3](#).

**CCON: PCA Counter Control Register**

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

7	6	5	4	3	2	1	0
CF	CR	CCF5	CCF4	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 5: CCF5, PCA Module 5 interrupt flag.  
 0: Must be cleared by software.  
 1: Set by hardware when a match or capture occurs.

Bit 4: CCF4, PCA Module 4 interrupt flag.  
 0: Must be cleared by software.  
 1: Set by hardware when a match or capture occurs.

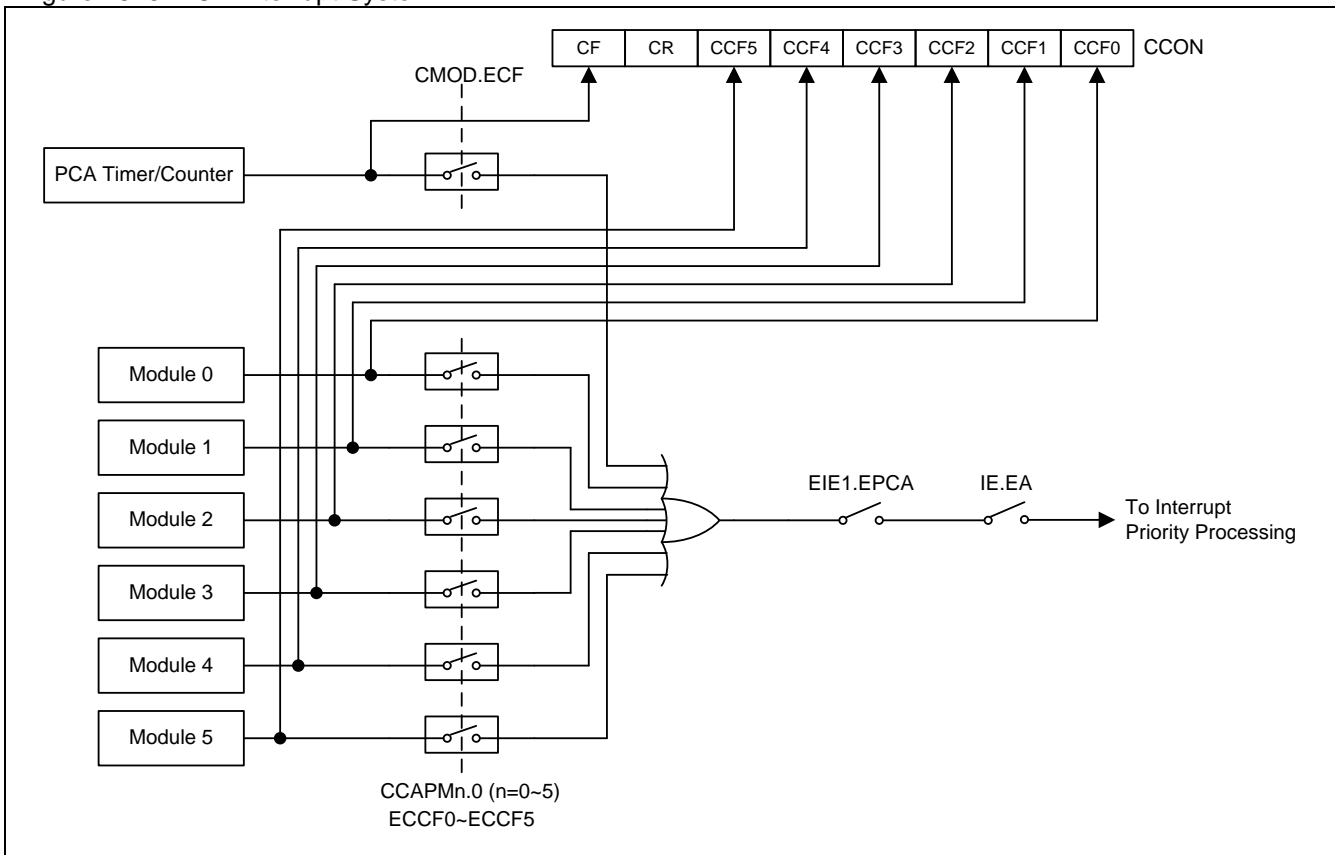
Bit 3: CCF3, PCA Module 3 interrupt flag.  
 0: Must be cleared by software.  
 1: Set by hardware when a match or capture occurs.

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 18–3. PCA Interrupt System



### 18.3. Compare/Capture Modules

Each of the six compare/capture modules has a mode register called CCAPMn (n = 0,1,2,3,4 or 5) 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=0~5**

SFR Page = All

SFR Address = 0xDA~0xDF                      RESET = x000-0000

7	6	5	4	3	2	1	0
--	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
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 the CCAPMn is written.

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 positive 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.

**PCAPWMn: PWM Mode Auxiliary Register, n=0~5**

SFR Page = All  
 SFR Address = 0xF2~0xF7                      RESET = 0000-0000

7	6	5	4	3	2	1	0
PnRS1	PnRS0	PnPS2	PnPS1	PnPS0	PnINV	ECAPnH	ECAPnL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

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.

**18.4. Operation Modes of the PCA**

Table 18–1 shows the CCAPMn register settings for the various PCA functions.

Table 18–1. PCA Module Modes

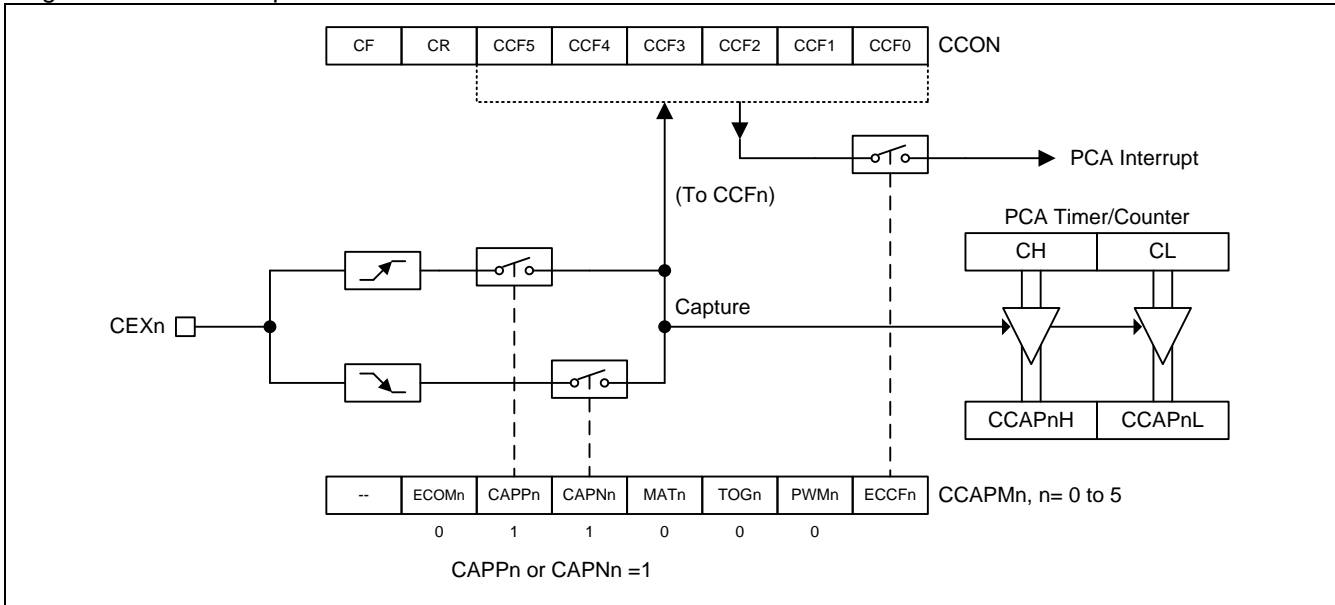
ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	Module Function
0	0	0	0	0	0	0	No operation
X	1	0	0	0	0	X	16-bit capture by a positive-edge trigger on CEXn
X	0	1	0	0	0	X	16-bit capture by a negative-edge trigger on CEXn
X	1	1	0	0	0	X	16-bit capture by a transition on CEXn
1	0	0	1	0	0	X	16-bit Software Timer
1	0	0	1	1	0	X	16-bit High Speed Output
1	0	0	0	0	1	0	8-bit Pulse Width Modulator (PWM)

**18.4.1. 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.



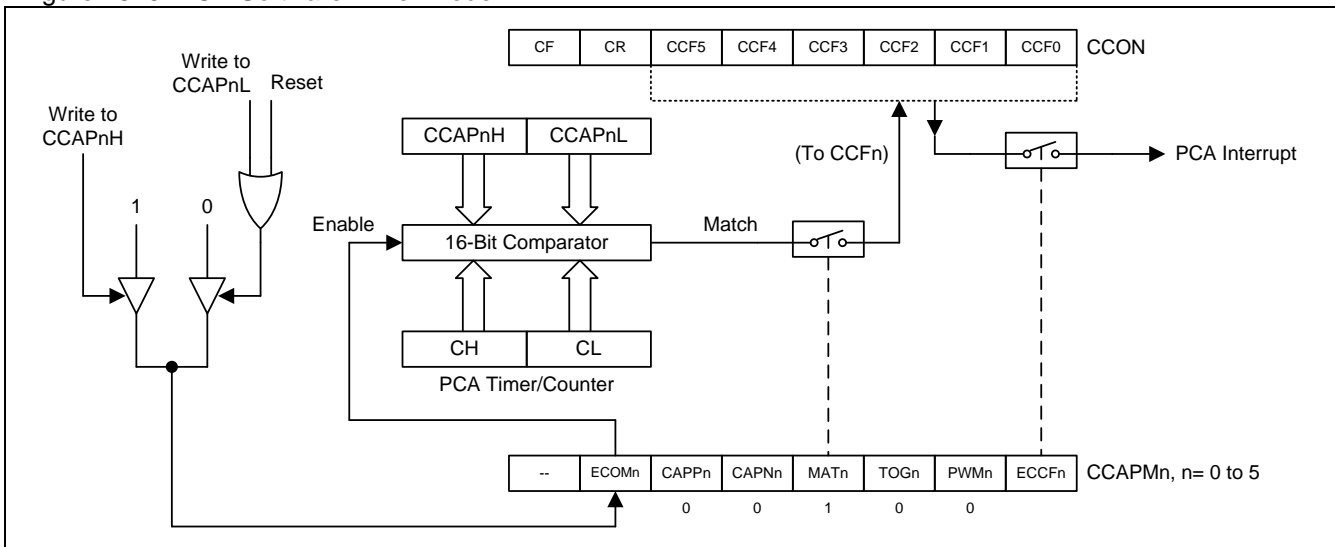
Figure 18–4. PCA Capture Mode



### 18.4.2. 16-bit Software Timer 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.

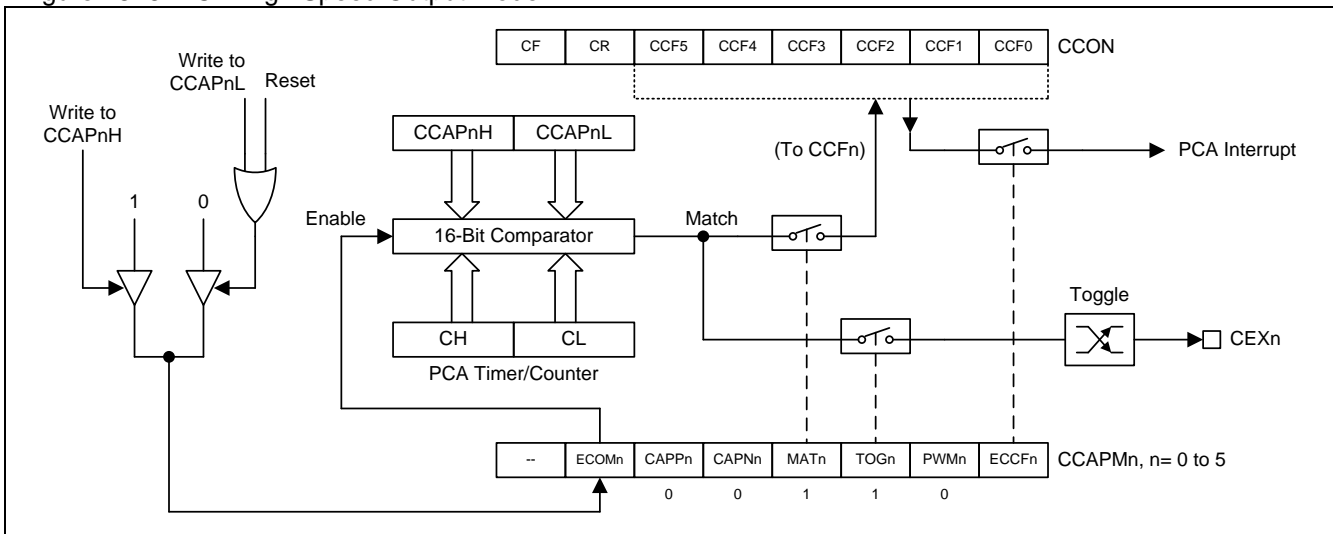
Figure 18–5. PCA Software Timer Mode



### 18.4.3. High Speed 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 18–6. PCA High Speed Output Mode



#### 18.4.4. 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 9<sup>th</sup> 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:

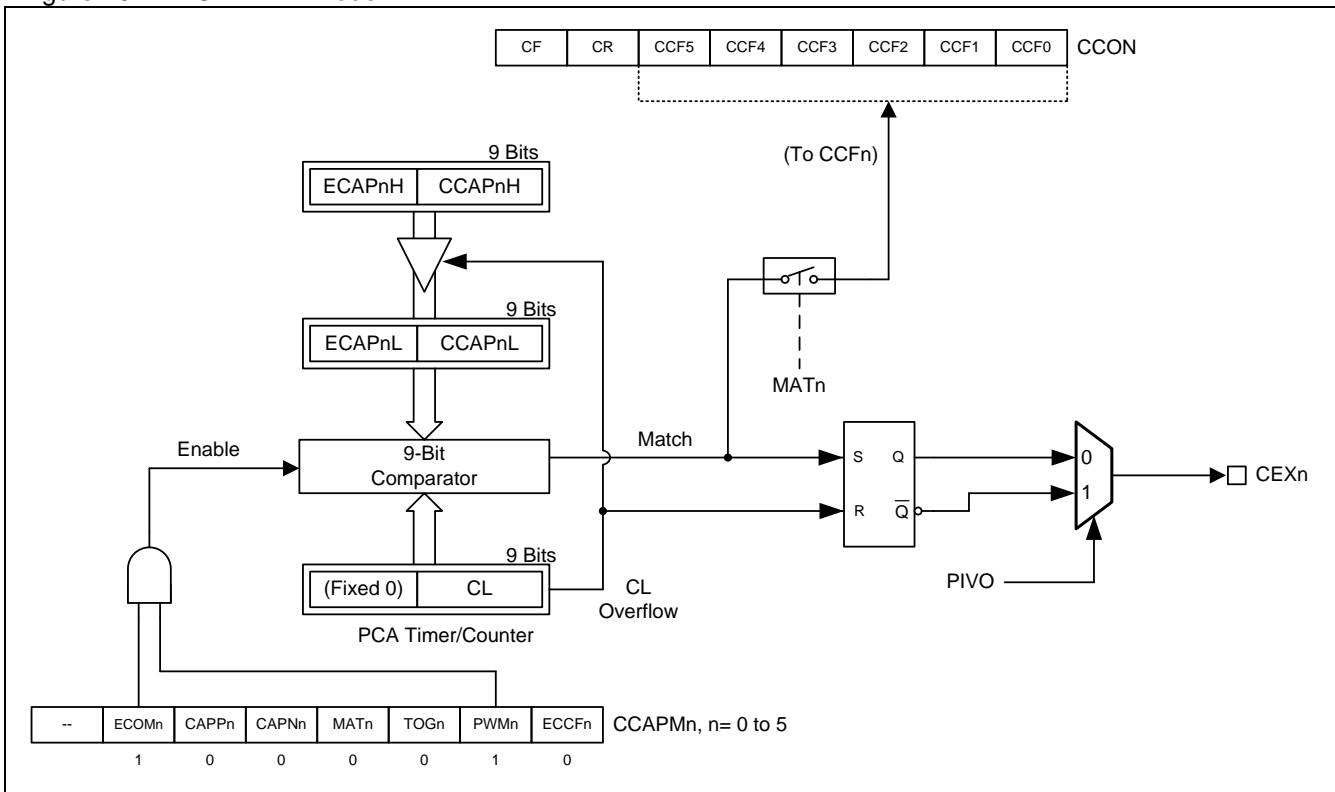
$$\text{Duty Cycle} = 1 - \{ \text{ECAPnH}, [\text{CCAPnH}] \} / 256.$$

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,

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

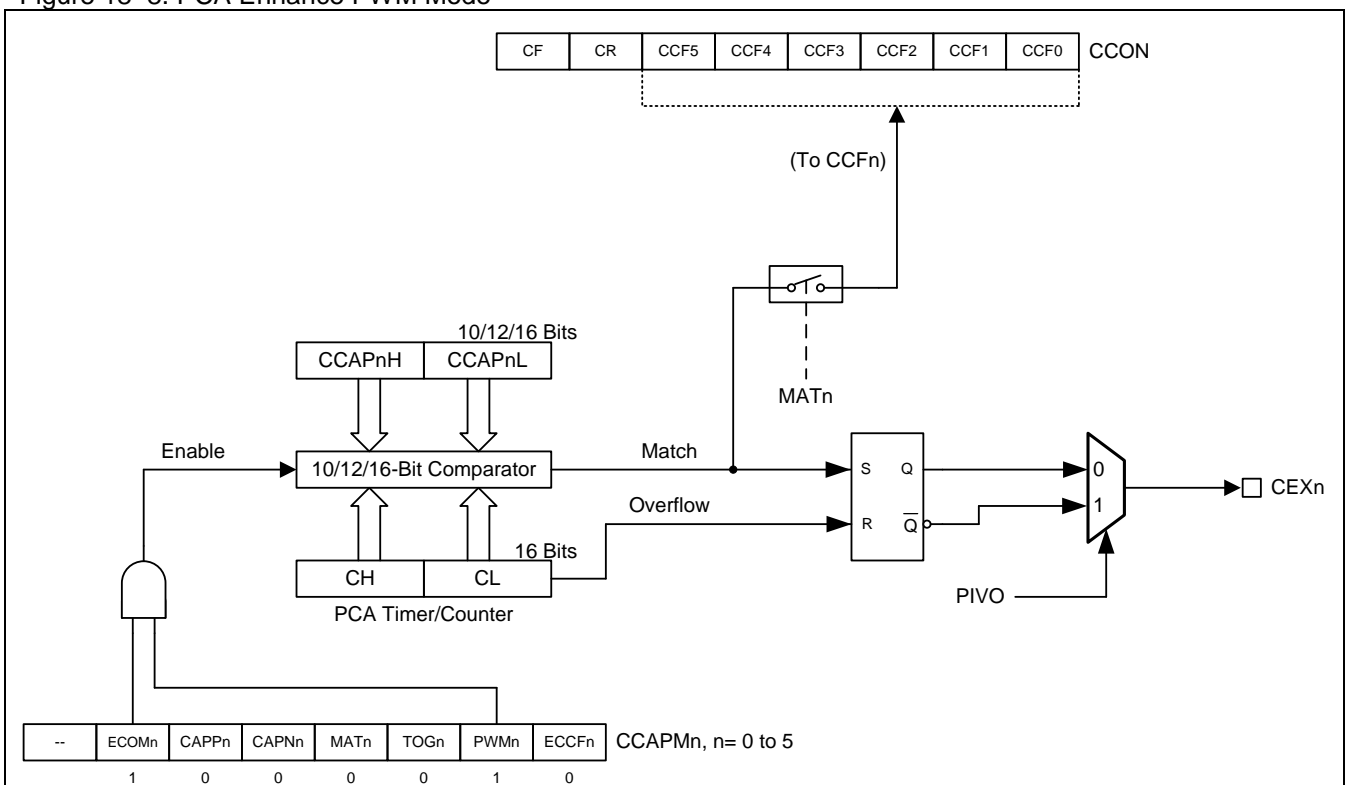
Figure 18–7. PCA PWM Mode



### 18.4.5. Enhance PWM Mode

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 18–8. PCA Enhance PWM Mode



**PCAPWMn: PWM Mode Auxiliary Register, n=0~5**

SFR Page = 0~F

SFR Address = 0xF2~0xF7 RESET = 0000-0000

7	6	5	4	3	2	1	0
PnRS1	PnRS0	PnPS2	PnPS1	PnPS0	PnINV	ECAPnH	ECAPnL
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: 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 5~3: PWMn Phase Setting 2~0.

000: The enabled PWM channel starts at 0 degree.

001: The enabled PWM channel starts at 90 degree.

010: The enabled PWM channel starts at 180 degree.

011: The enabled PWM channel starts at 270 degree.

100: The enabled PWM channel starts at 120 degree.

101: The enabled PWM channel starts at 240 degree.

110: The enabled PWM channel starts at 60 degree.

111: The enabled PWM channel starts at 300 degree.

In default PCA PWM mode, all PWM outputs are cleared on CL overflow (See Figure 18-7). All PWM outputs go to low simultaneously and are set to high by the match event from individual CCAPnL setting and CL counter. This mode PWM behaves a same phase PWM because the PWM outputs always start at the same time. The PCA enhanced PWM mode provides the phase delay function on each PWM channel with different PWM resolution. The following table indicates the counter value to clear PWM output if comparator result is matched. The set condition of PWM outputs keeps the original matched event by {CCFnH, CCFnL} and {CH, CL}. So after setting the phase delay parameter, software only take care the value of the PWM END count (PWM output SET) to implement the variable phase delay PWM.

Phase	0°/360°	90°	180°	270°	120°	240°	60°	300°
PWM8	00	40	80	C0	55	AA	2A	D5
PWM10	{00}00	{01}00	{10}00	{11}00	{01}55	{10}AA	{00}AA	{11}55
PWM12	000	400	800	C00	555	AAA	2AA	D55
PWM16	0000	4000	8000	C000	5555	AAAA	2AAA	D555

Bit 2: Invert PWM output on CEXn.

0: Non-inverted PWM output.

1: Inverted PWM output.

Bit 1: ECAPnH: Extended MSB bit, associated with CCAPnH to become a 9th-bit register used in 8-bit PWM mode. As well as for 10/12/16 bit PWM, it will become a 11th/13th/17<sup>th</sup> bit register.

Bit 0: ECAPnL: Extended MSB bit, associated with CCAPnL to become a 9th-bit register used in 8-bit PWM mode. As well as for 10/12/16 bit PWM, it will become a 11th/13th/17<sup>th</sup> bit register.

**CMOD: PCA Counter Mode Register**

SFR Page = 0~F

SFR Address = 0xD9 RESET = 00xx-x000

7	6	5	4	3	2	1	0
CIDL	FEOV	--	--	--	CPS1	CPS0	ECF
R/W	R/W	W	W	W	R/W	R/W	R/W

Bit 6: FEOV: CL overflows on FEH enabled.

0: CL overflows on FFH.

1: CL overflows on FEH.

## 18.5. PCA Sample Code

(1). Required Function: Set PWM2/PWM3 output with 25% & 75% duty cycle

Assembly Code Example:

```
PWM2_PWM3:
  MOV     CCON,#00H           ; stop CR
  MOV     CMOD,#02H          ; PCA clock source = system clock / 2

  MOV     CH,#00H            ; initial state
  MOV     CL,#00H
  ;
  MOV     PCAPWM2,#PWM2      ; enable PCA module 2 (PWM mode)
  MOV     CCAP2H,#0C0H       ; 25%
  MOV     CCAP2L,#0C0H

  MOV     PCAPWM3,#PWM3      ; enable PCA module 3 (PWM mode)
  MOV     CCAP3H,#40H        ; 75%
  MOV     CCAP3L,#40H
  ;
  MOV     P2M0,#00110000B    ; enable P2.5 & P2.4 push-pull
  SETB   CR                  ; start PCA
```

C Code Example:

```
void main(void)
{
  // set PCA
  CCON = 0x00;           // disable PCA & clear CCF0, CCF1, CF flag
  CMOD = 0x02;          // PCA clock source = system clock / 2

  CL = 0x00; CH = 0x00; // PCA counter range
  //-----
  PCAPWM2 = PWM2;       // module 2 (Non-inverted)
  CCAP2H = 0xC0; CCAP2L = 0xC0; // 25%

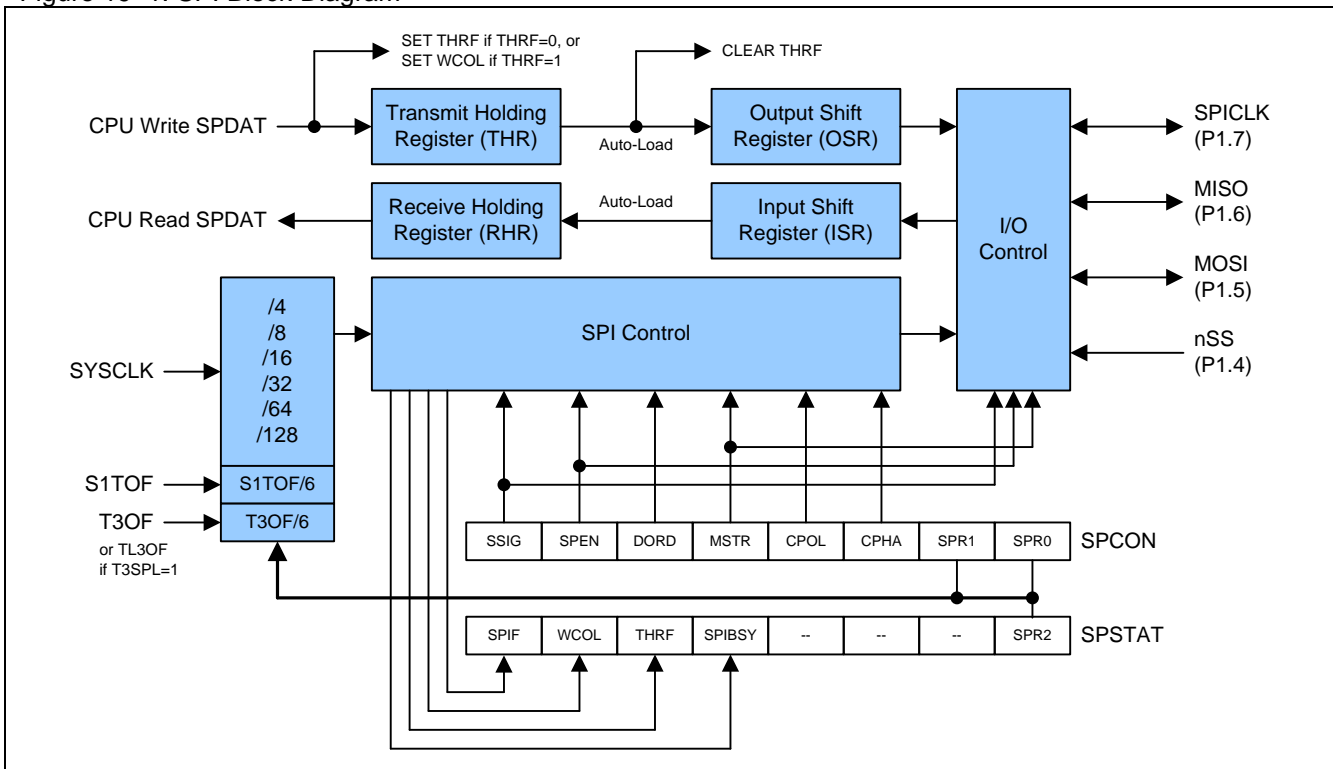
  PCAPWM3 = PWM3;       // module 3
  CCAP3H = 0x40; CCAP3L = 0x40; // 75 %
  //-----
  P2M0 = 0x30;
  CR = 1;                // start PCA's PWM output

  while (1);
}
```

## 19. Serial Peripheral Interface (SPI)

The **MG84FG516** 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 3 Mbps can be supported in either Master or Slave mode under a 12MHz system clock. It has a Transfer Completion Flag (SPIF) and Write Collision Flag (WCOL) in the SPI status register (SPSTAT). And a specially designed Transmit Holding Register (THR) improves the transmit performance compared to the conventional SPI. SPIBSY read-only flag reports the Busy state in SPI engine.

Figure 19–1. SPI Block Diagram



The SPI interface has four pins: MISO (P1.6), MOSI (P1.5), SPICLK (P1.7) and /SS (P1.4):

- 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.
- /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 /SS pin to determine whether it is selected. The /SS 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 P1.4 (/SS) is configured as an output.
  - If the /SS pin is ignored, i.e. SSIG (SPCTL.7) bit = 1, this pin is configured for port functions.

*Note: See the AUXR1 in Section “5.3 Alternate Function Redirection”, for its alternate pin-out option.*

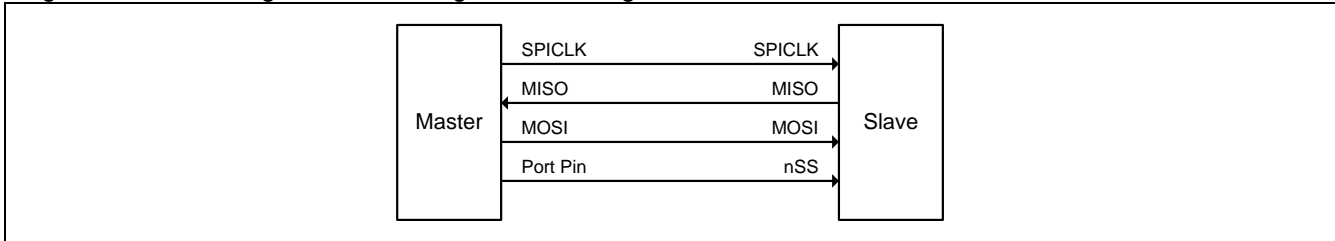
Note that even if the SPI is configured as a master (MSTR=1), it can still be converted to a slave by driving the /SS pin low (if SSIG=0). Should this happen, the SPIF bit (SPSTAT.7) will be set. (See Section “19.2.3 Mode Change on nSS-pin”)

## 19.1. Typical SPI Configurations

### 19.1.1. Single Master & Single Slave

For the master: any port pin, including P1.4 (/SS), can be used to drive the /SS pin of the slave.  
 For the slave: SSIG is '0', and /SS pin is used to determine whether it is selected.

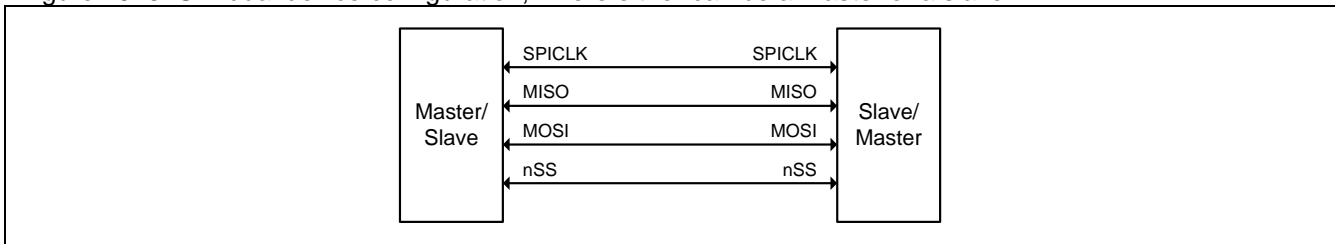
Figure 19–2. SPI single master & single slave configuration



### 19.1.2. 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 P1.4 (/SS) configured in quasi-bidirectional mode. When any device initiates a transfer, it can configure P1.4 as an output and drive it low to force a “mode change to slave” in the other device. (See Section “19.2.3 Mode Change on nSS-pin”)

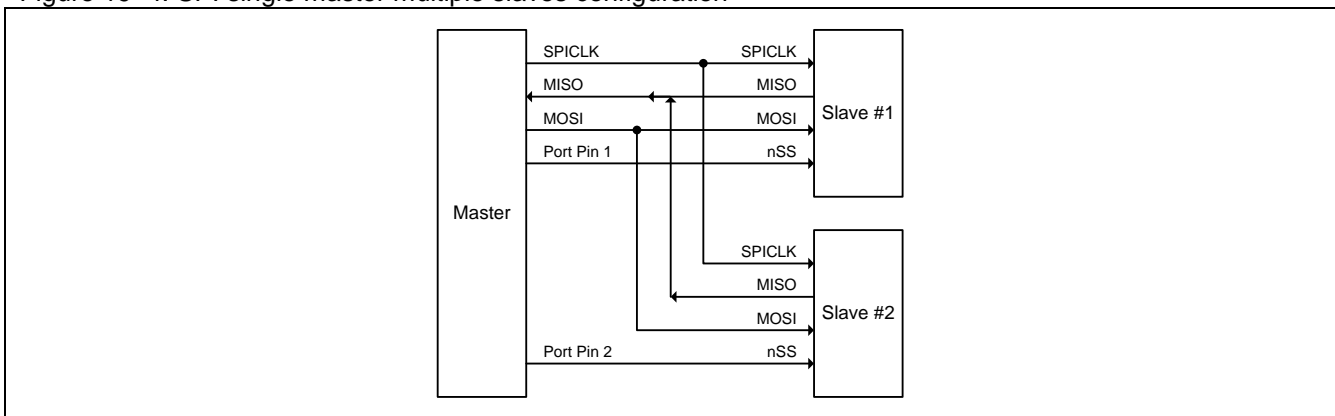
Figure 19–3. SPI dual device configuration, where either can be a master or a slave



### 19.1.3. Single Master & Multiple Slaves

For the master: any port pin, including P1.4 (/SS), can be used to drive the /SS pins of the slaves. For all the slaves: SSIG is '0', and /SS pin are used to determine whether it is selected.

Figure 19–4. SPI single master multiple slaves configuration



## 19.2. Configuring the SPI

Table 19–1 shows configuration for the master/slave modes as well as usages and directions for the modes.

Table 19–1. SPI Master and Slave Selection

SPEN (SPCTL.6)	SSIG (SPCTL.7)	/SS -pin	MSTR (SPCTL.4)	Mode	MISO -pin	MOSI -pin	SPICLK -pin	Remarks
0	X	X	X	SPI disabled	input	input	input	P1.4~P1.7 are used as general port pins.
1	0	0	0	Slave (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 /SS pin is driven low, and MSTR will be cleared to '0' by H/W automatically.
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".

### 19.2.1. Additional Considerations for a Slave

When CPHA is 0, SSIG must be 0 and /SS pin must be negated and reasserted between each successive serial byte transfer. Note the SPDAT register cannot be written while /SS 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 /SS 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.

### 19.2.2. 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 /SS 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.



### 19.2.3. Mode Change on nSS-pin

If SPEN=1, SSIG=0, MSTR=1 and /SS 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.

### 19.2.4. 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, 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 master 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).

### 19.2.5. Write Collision

The SPI in **MG84FG516** is double buffered data both in the transmit direction and in the receive direction. New data for transmission can not 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.

### 19.2.6. 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 19–2](#).

Table 19–2. SPI Serial Clock Rates

SPR2	SPR1	SPR0	SPI Clock Selection	SPI Clock Rate @ SYSCLK=12MHz
0	0	0	SYSClk/4	3 MHz
0	0	1	SYSClk/8	1.5 MHz
0	1	0	SYSClk/16	750 KHz
0	1	1	SYSClk/32	375 KHz
1	0	0	SYSClk/64	187.5 KHz
1	0	1	SYSClk/128	93.75 KHz
1	1	0	<b>S1TOF/6</b>	Variable
1	1	1	<b>T3OF/6</b>	Variable

Note:

1. SYSCLK is the system clock.
2. S1TOF is UART1 Baud-Rate Timer Overflow.
3. T3OF is Timer 3 Overflow.
4. In Timer 3 split mode, T3OF is replaced by TL3OF.

### 19.3. 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.

Figure 19–5. SPI Slave Transfer Format with CPHA=0

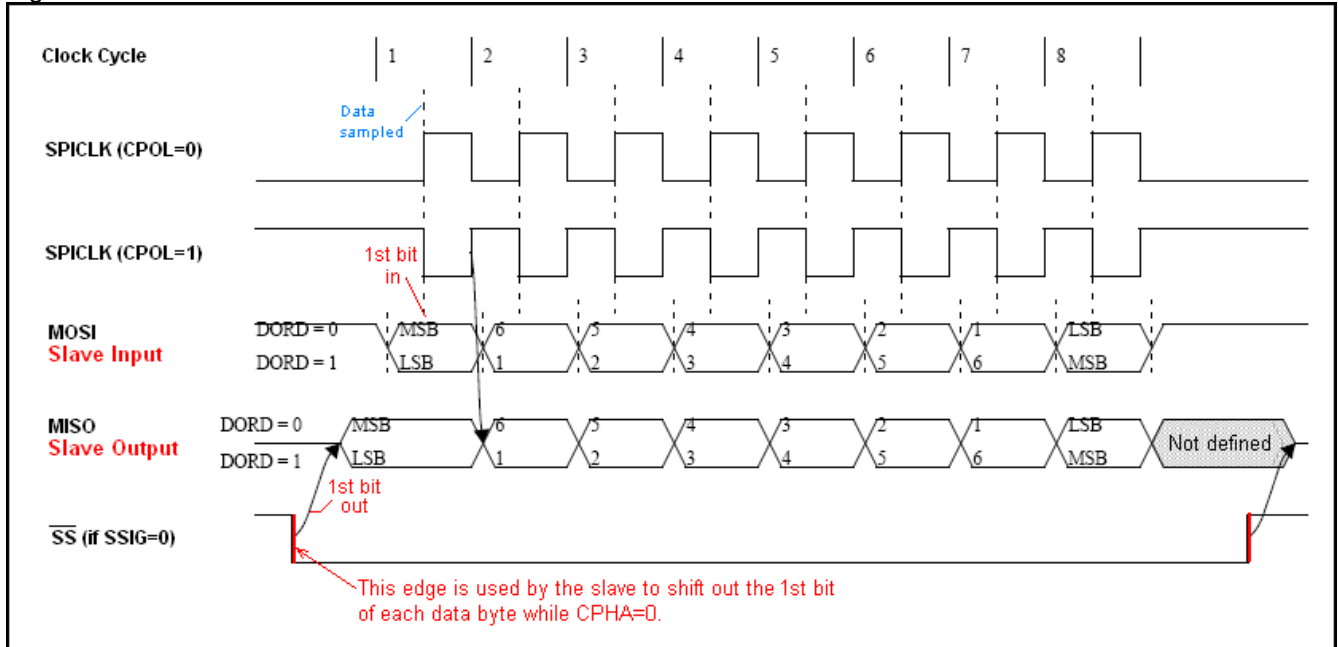


Figure 19–6. Slave Transfer Format with CPHA=1

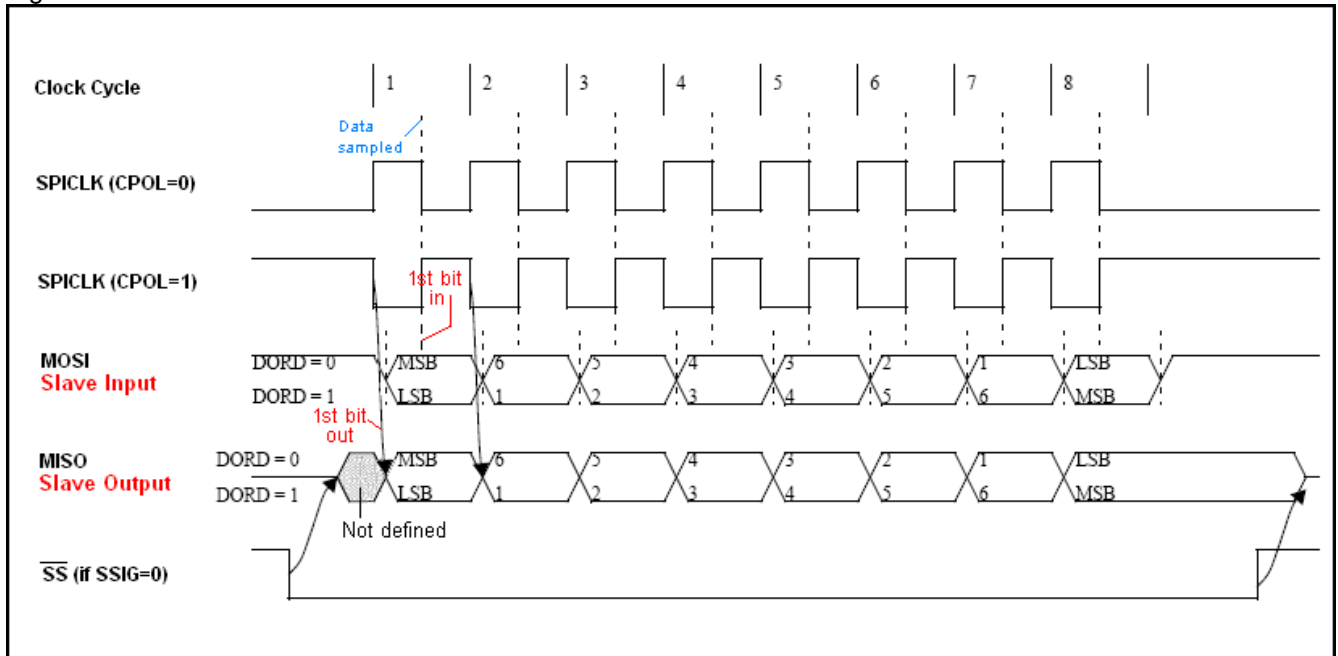


Figure 19–7. SPI Master Transfer Format with CPHA=0

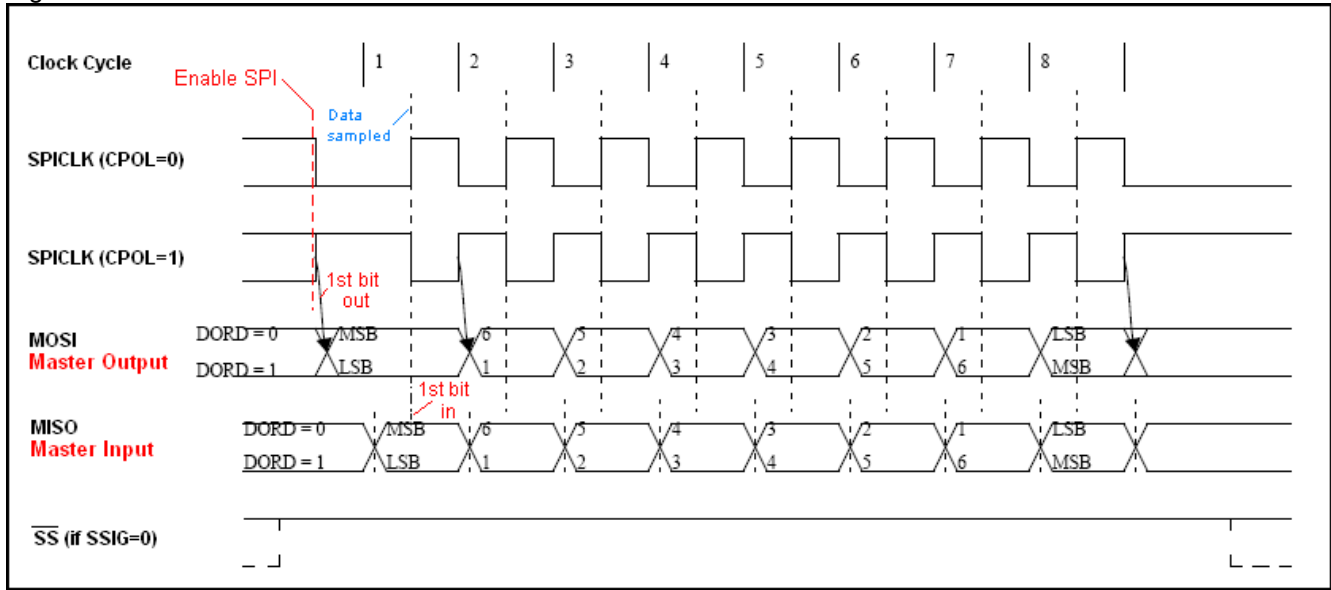
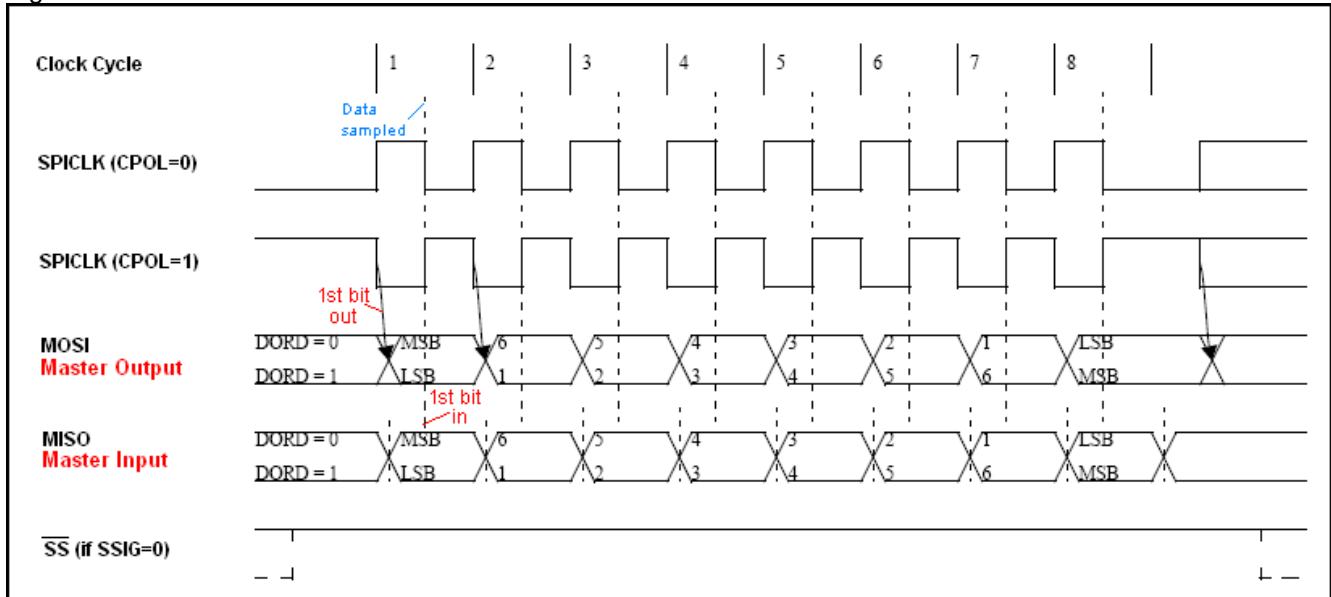


Figure 19–8. SPI Master Transfer Format with CPHA=1



## 19.4. 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 /SS 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.

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=12MHz
0	0	0	SYSClk/4	3 MHz
0	0	1	SYSClk/8	1.5 MHz
0	1	0	SYSClk/16	750 KHz
0	1	1	SYSClk/32	375 KHz
1	0	0	SYSClk/64	187.5 KHz
1	0	1	SYSClk/128	93.75 KHz
1	1	0	<b>S1TOF/6</b>	Variable
1	1	1	<b>T3OF/6</b>	Variable

Note:

1. SYSCLK is the system clock.
2. S1TOF is UART1 Baud-Rate Timer Overflow.
3. T3OF is Timer 3 Overflow.
4. In Timer 3 split mode, T3OF is replaced by TL3OF.

**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	--	--	--	SPR2
R/W	R/W	R	R	W	W	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 “19.2.5 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 3~1: Reserved. Software must write “0” on these bits 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.

## 19.5. SPI Sample Code

### (1). Required Function: Set SPI Master write/read

#### Assembly Code Example:

```

MOV     SPCON,#( SPEN | SSIG | MSTR) ;enable SPI and set sampling data at rising edge,
                                           ;SPICLK is sysclk/ 4.
MOV     P1M0,#0B0H                    ; set P14 to push-pull
CLR     P14                            ; enable slave device select
MOV     SPDAT,#55H                    ; SPI send Addr=0x55 to slave
MOV     a,#20H
check_THRF_0:
ANL     a,SPSTAT
JNZ     check_THRF_0

MOV     SPDAT,#0AAH                    ; SPI send Data=0xAA to slave;
MOV     a,#10H
check_SPIBSY_0:
ANL     a,SPSTAT
JNZ     check_SPIBSY_0
SETB    P14                            ; disable slave device select

CLR     P14                            ; enable slave device select
MOV     SPDAT,#55H                    ; SPI send Addr=0x55 to slave
MOV     a,#20H
check_THRF_0:
ANL     a,SPSTAT
JNZ     check_THRF_0

MOV     SPDAT,#0FFH                    ; SPI send Data=0xff dummy data, and read back data
MOV     a,#10H
check_SPIBSY_0:
ANL     a,SPSTAT
JNZ     check_SPIBSY_0
SETB    P14                            ; disable slave device select

MOV     A,SPDAT
;SPDAT=read back Data

```

#### C Code Example:

```

#define    nCS                P14
void main(void)
{
    Unsigned char SPI_read_Data;

    SPCON = ( SPEN | SSIG | MSTR);    //enable SPI and set sampling data at rising edge, SPICLK is sysclk / 4.
    P1M0 = 0xB0;                      //set P14 to push-pull
    nCS = 0;                           //enable slave device select
    SPDAT = 0x55;                      // SPI send Addr=0x55 to slave;
    while(SPSTAT & THRF);
    SPDAT = 0xAA;                      //SPI send Data=0xAA to slave;
    while(SPSTAT & SPIBSY);
    nCS = 1;                            //disable slave device select
    //;
    nCS = 0;                            //enable slave device select
    SPDAT = 0x55;                      // SPI send Addr=0x55 to slave;
    while(SPSTAT & THRF);
    SPDAT = 0xFF;                      // SPI send Data=0xff dummy data, and read back data
    while(SPSTAT & SPIBSY);
    nCS = 1;                            //disable slave device select

    SPI_read_Data = SPDAT;

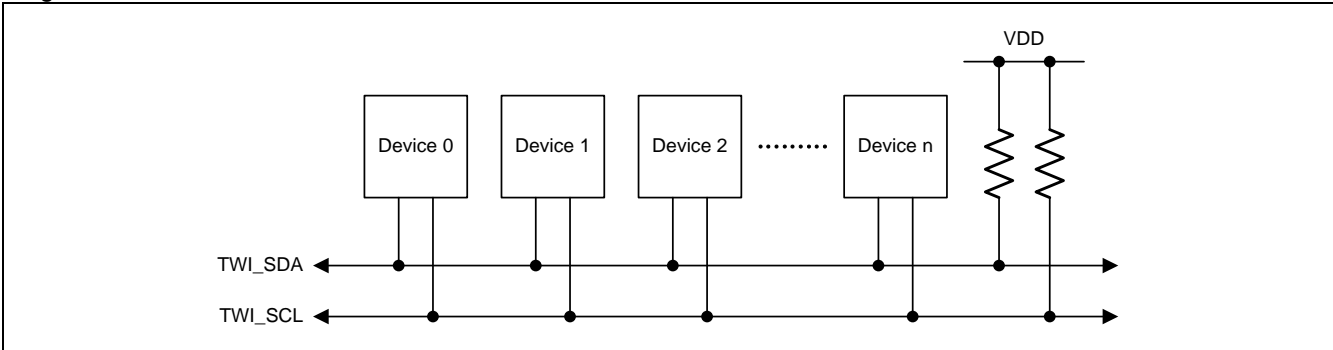
    while (1);
}

```

## 20. Two Wire Serial Interface (TWSI/TWI)

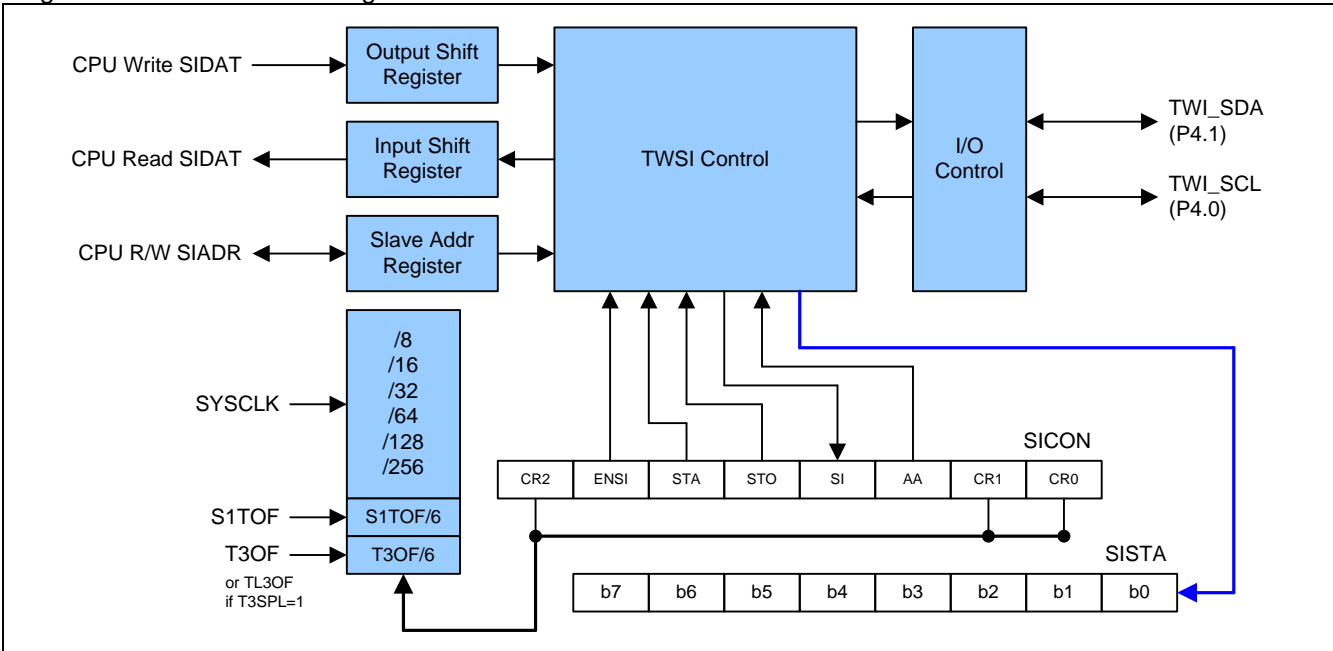
The Two-Wire Serial interface is a two-wire, bi-directional serial bus. It is ideally suited for typical microcontroller applications. The TWSI protocol allows the systems designer to interconnect up to 128 different devices using only two bi-directional bus lines, one for clock (SCL) and one for data (SDA). The TWSI bus provides control of SDA (serial data, P4.1), SCL (serial clock, P4.0) 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 TWSI bus lines. All devices connected to the bus have individual addresses, and mechanisms for resolving bus contention are inherent in the TWSI protocol.

Figure 20–1. TWSI Bus Interconnection



The TWSI bus may operate as a master and/or slave, and may function on a bus with multiple masters. The CPU interfaces to the TWSI through the following four special function registers: SICON configures the TWSI bus; SISTA reports the status code of the TWSI bus; and SIDAT is the data register, used for both transmitting and receiving TWSI data. SIADR is the slave address register. And, the TWSI hardware interfaces to the serial bus via two lines: SDA (serial data line, P4.1) and SCL (serial clock line, P4.0).

Figure 20–2. TWSI Block Diagram



## 20.1. Operating Modes

There are four operating modes for the TWSI: 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 TWSI 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 TWSI interrupt is enabled), and the new status code can be used to determine which appropriate routine the software is to branch to.

### 20.1.1. 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 TWSI. If the AA bit is reset, TWSI 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, TWSI 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 TWSI 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), TWSI may switch to the master receiver mode by loading SIDAT with SLA+R.

### 20.1.2. 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), TWSI may switch to the master transmitter mode by loading SIDAT with SLA+W.



### 20.1.3. 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 TWSI will respond when addressed by a master. If the LSB (GC) is set, TWSI 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 TWSI in the slave mode. ENSI must be set to “1” to enable TWSI. The AA bit must be set to enable TWSI 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, TWSI waits until it is addressed by its own slave address followed by the data direction bit which must be “1” (R) for TWSI 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 TWSI is in the master mode (see state B0H).

If the AA bit is reset during a transfer, TWSI will transmit the last byte of the transfer and enter state C0H or C8H. TWSI 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, TWSI 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 TWSI from the bus.

### 20.1.4. 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, TWSI waits until it is addressed by its own slave address followed by the data direction bit which must be “0” (W) for TWSI 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 TWSI is in the master mode (see status 68H and 78H).

If the AA bit is reset during a transfer, TWSI will return a not acknowledge (logic 1) to SDA after the next received data byte. While AA is reset, TWSI 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.

## 20.2. Miscellaneous States

There are two SISTA codes that do not correspond to a defined TWSI 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 TWSI is not involved in a serial transfer.

### S1STA = 00H:

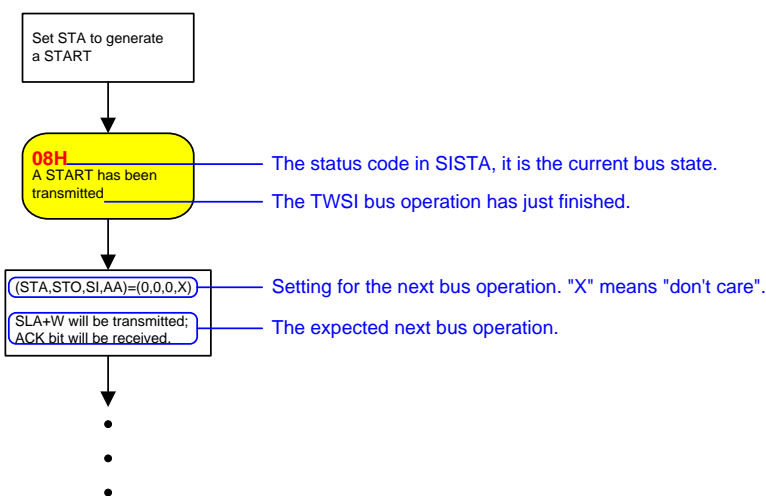
This status code indicates that a bus error has occurred during an TWSI serial transfer. A bus error is caused when a START or STOP condition occurs at an illegal position in the format frame. Examples of such illegal positions are during the serial transfer of an address byte, a data byte, or an acknowledge bit. A bus error may also be caused when external interference disturbs the internal TWSI 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 TWSI to enter the "not-addressed" slave mode (a defined state) and to clear the STO flag (no other bits in SICON are affected). The SDA and SCL lines are released (a STOP condition is not transmitted).

## 20.3. Using the TWSI

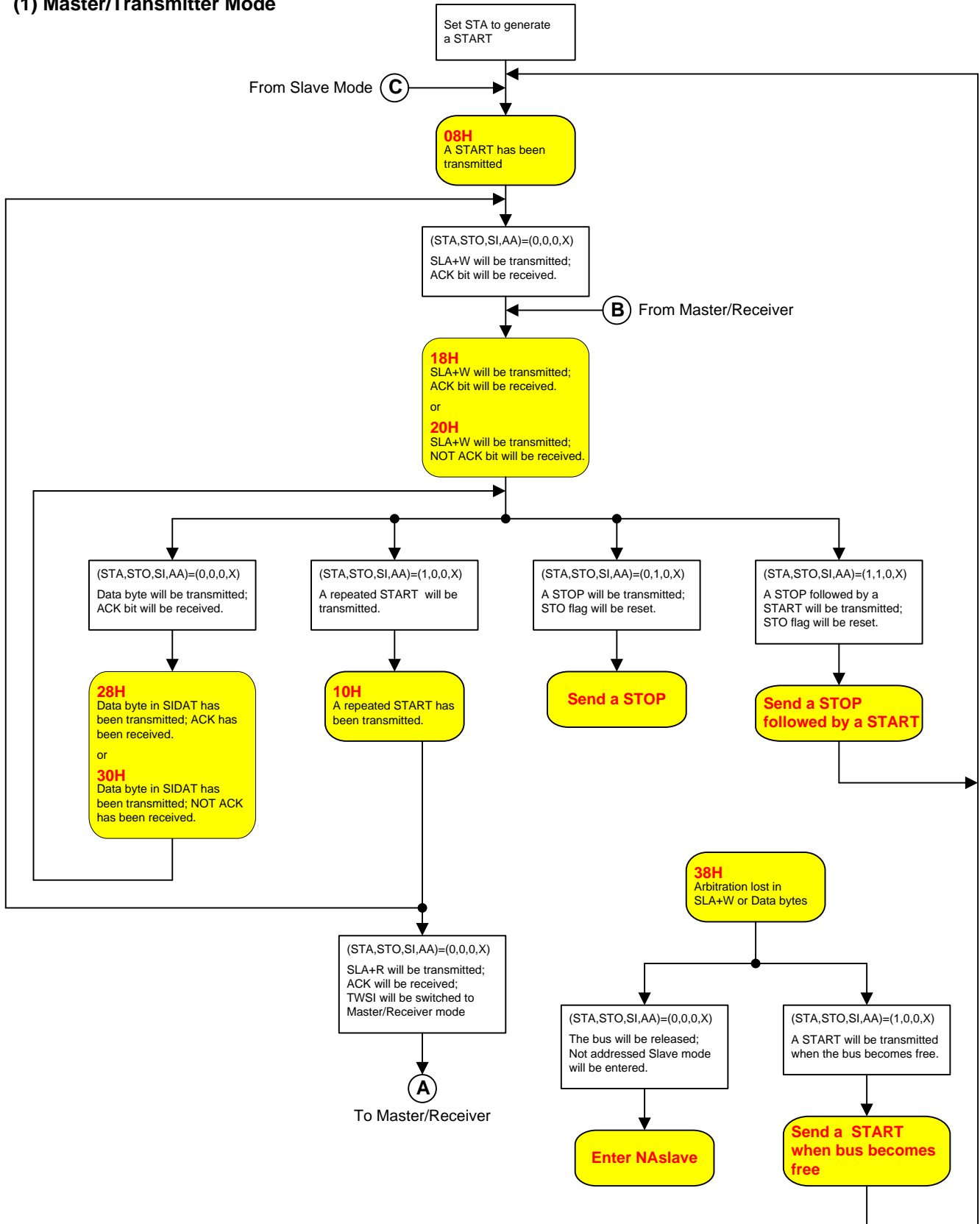
The TWSI is byte-oriented and interrupt based. Interrupts are issued after all bus events, like reception of a byte or transmission of a START condition. Because the TWSI is interrupt-based, the application software is free to carry on other operations during a TWSI byte transfer. Note that the TWSI interrupt enable bit ETWSI bit (AUXIE.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 TWSI 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 TWSI bus. The application software can then decide how the TWSI should behave in the next TWSI 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 TWSI 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 TWSI 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 TWSI 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"

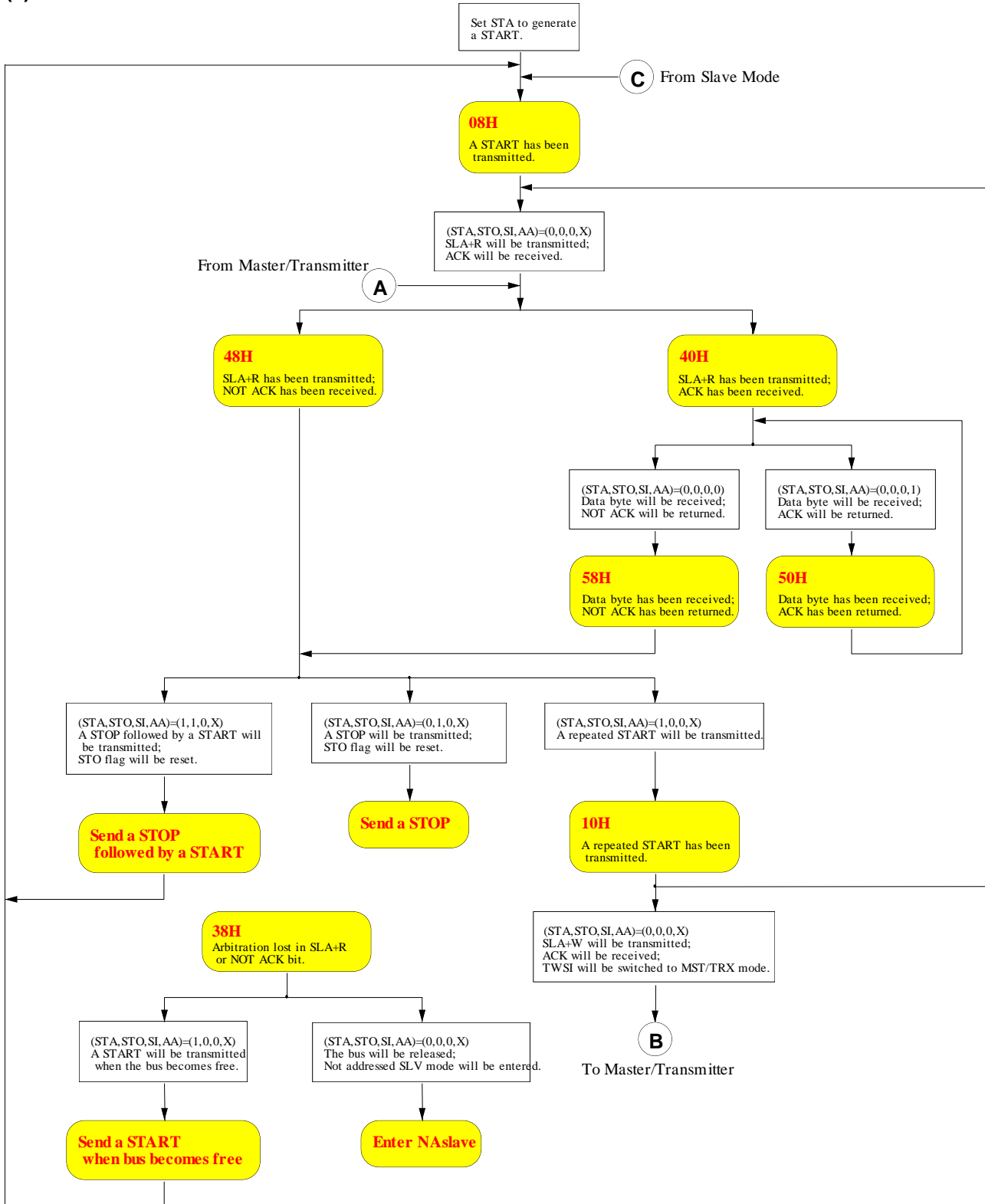
The figure below shows how to read the flow charts.



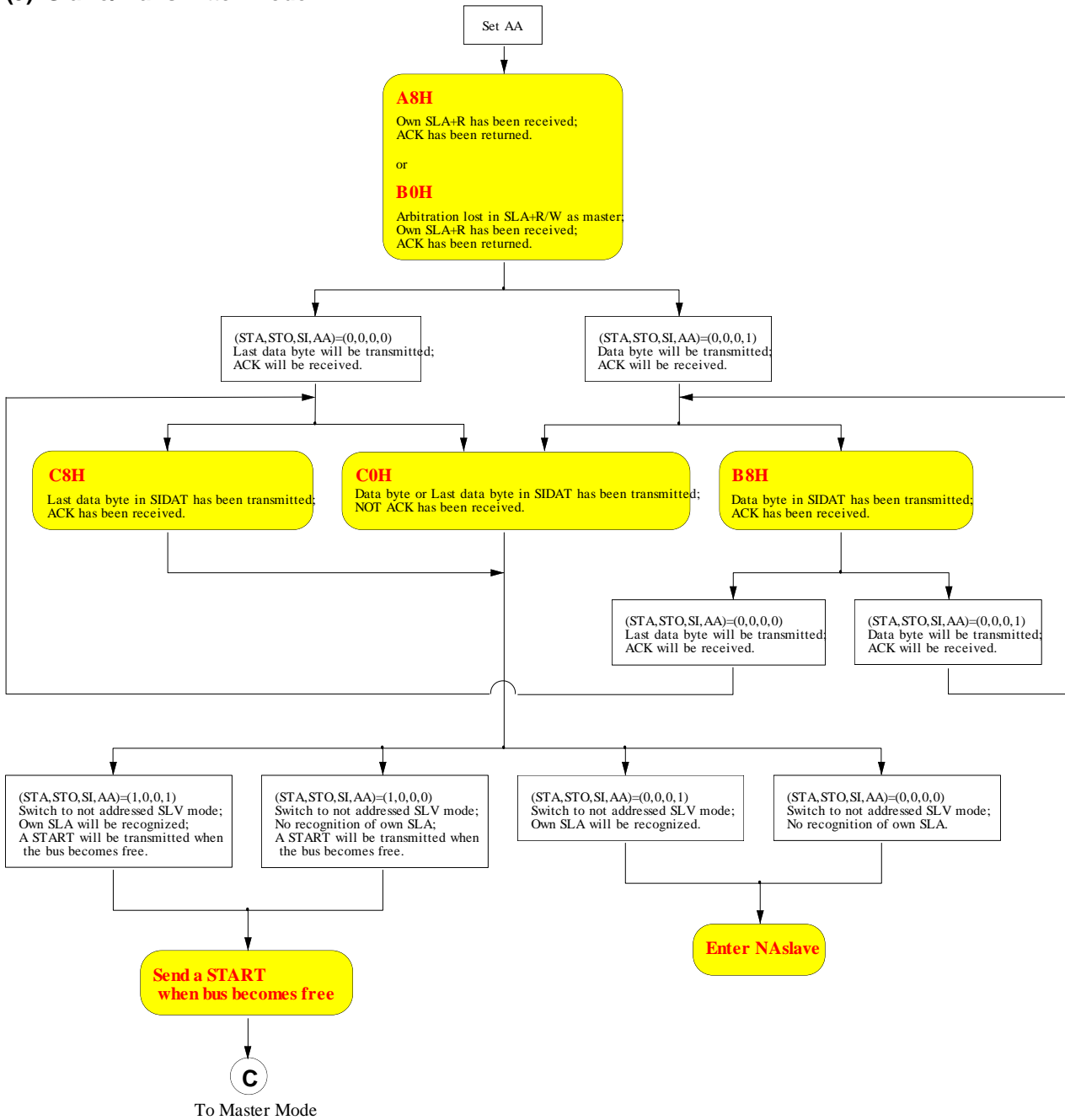
## (1) Master/Transmitter Mode



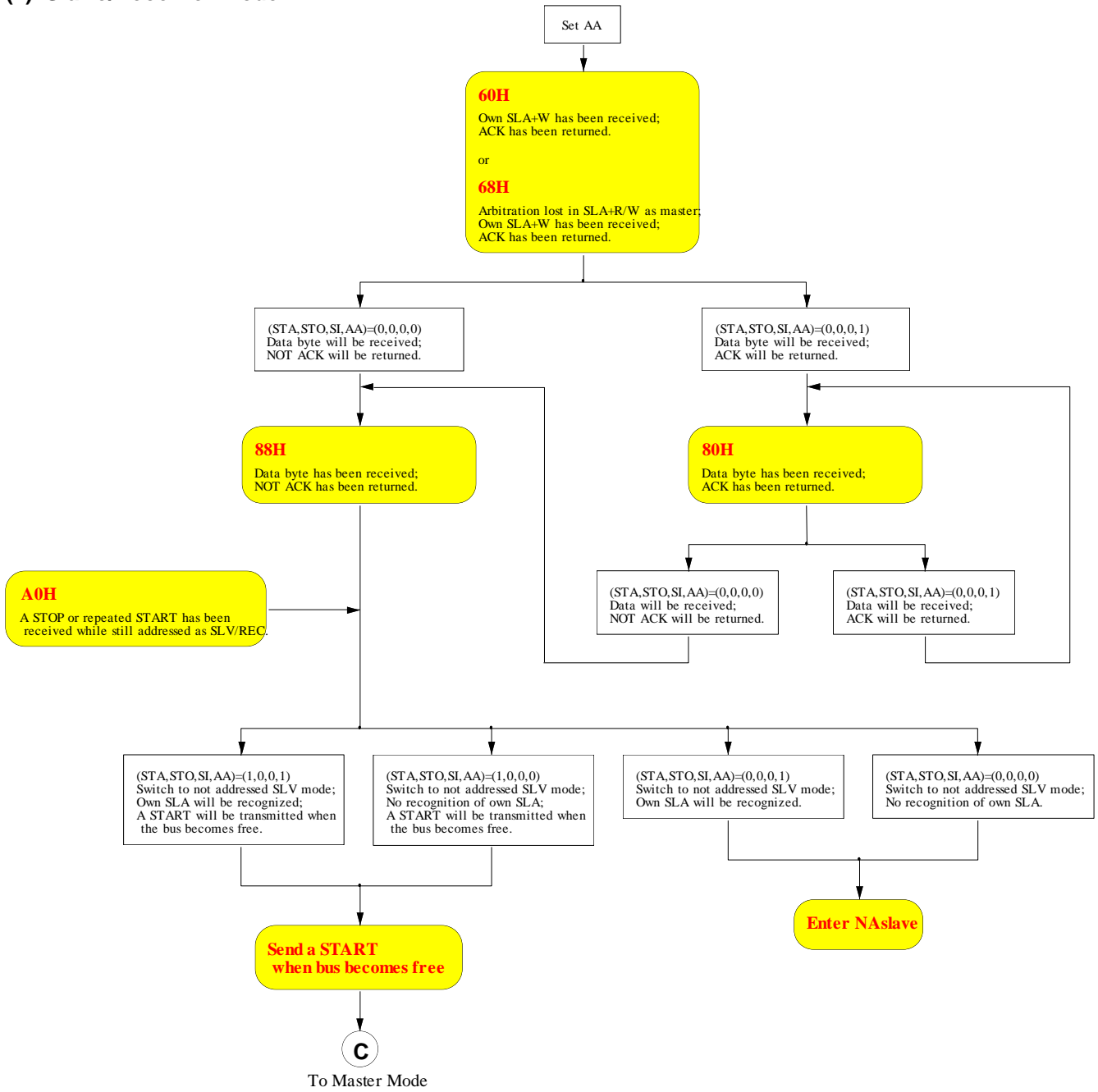
## (2) Master/Receiver Mode



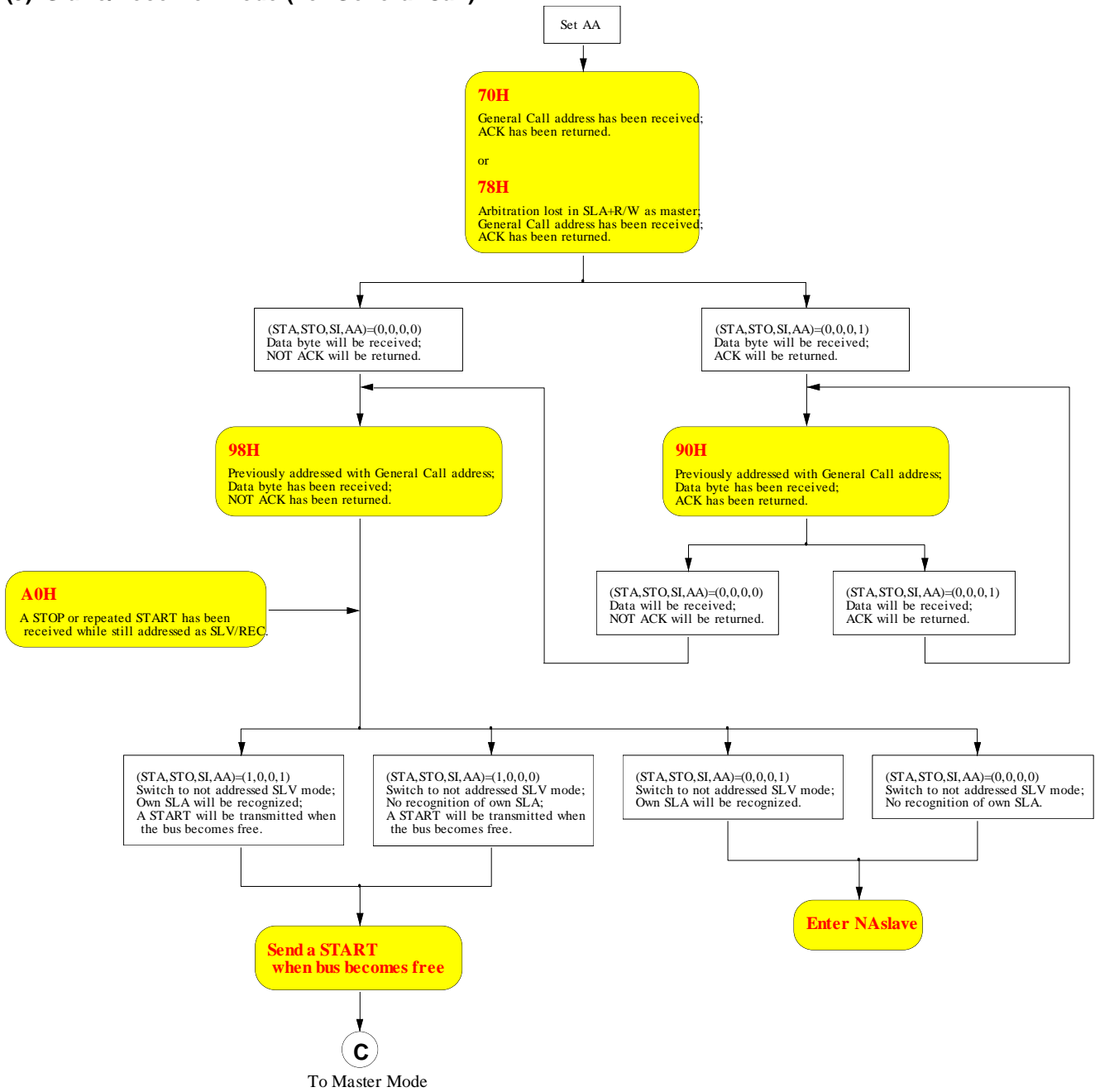
### (3) Slave/Transmitter Mode



#### (4) Slave/Receiver Mode



### (5) Slave/Receiver Mode (For General Call)



## 20.4. TWSI Register

### SIADR: 2-wire Serial Interface Address Register

SFR Page = 0~F

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 TWSI hardware. The contents of this register are irrelevant when TWSI 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 TWSI bus after a START condition.

### SIDAT: 2-wire Serial Interface Data Register

SFR Page = 0~F

SFR Address = 0xD2

RESET= 0000-0000

7	6	5	4	3	2	1	0
D7	D6	D5	D4	D3	D2	D1	D0
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 TWSI 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 TWSI 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 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 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: 2-wire Serial Interface Control Register

SFR Page = 0~F

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 from and write to this register directly. Two bits are affected by the TWSI hardware: the SI bit is set when a serial interrupt is requested, and the STO bit is cleared when a STOP condition is present on the bus. The STO bit is also cleared when ENSI="0".

Bit 7: CR2, TWSI Clock Rate select bit 2 (associated with CR1 and CR0).

Bit 6: ENSI, the TWSI Hardware Enable Bit

When ENSI is "0", the SDA and SCL outputs are in a high impedance state. SDA and SCL input signals are ignored, TWSI is in the not-addressed slave state, and STO bit in SICON is forced to "0". No other bits are affected, and, P4.1 (SDA) and P4.0 (SCL) may be used as general purpose I/O pins. When ENSI is "1", TWSI is enabled, and, the P4.1 and P4.0 port latches must be set to logic 1 and I/O mode must be configured to open-drain mode for the following serial communication.



**Bit 5: STA, the START Flag**

When the STA bit is set to enter a master mode, the TWSI hardware checks the status of the serial bus and generates a START condition if the bus is free. If the bus is not free, then TWSI waits for a STOP condition and generates a START condition after a delay. If STA is set while TWSI is already in a master mode and one or more bytes are transmitted or received, TWSI transmits a repeated START condition. STA may be set at any time. STA may also be set when TWSI is an addressed slave. 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 bit is set while TWSI is in a master mode, a STOP condition is transmitted to the serial bus. When the STOP condition is detected on the bus, the TWSI 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 TWSI 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 TWSI is in a master mode (in a slave mode, TWSI 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 TWSI state is present in the SISTA register, the SI flag is set by hardware. And, if the TWSI 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 SCL line is stretched, and the serial transfer is suspended. A high level on the 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 SCL line.

**Bit 2: AA, the Assert Acknowledge Flag**

If the AA flag is set to “1”, an acknowledge (low level to SDA) will be returned during the acknowledge clock pulse on the SCL line when:

- 1) The own slave address has been received.
- 2) A data byte has been received while TWSI is in the master/receiver mode.
- 3) A data byte has been received while TWSI is in the addressed slave/receiver mode.

If the AA flag is reset to “0”, a not acknowledge (high level to SDA) will be returned during the acknowledge clock pulse on SCL when:

- 1) A data has been received while TWSI is in the master/receiver mode.
- 2) A data byte has been received while TWSI 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 TWSI is in a master mode. The clock rate is not important when TWSI is in a slave mode because TWSI will automatically synchronize with any clock frequency, which is from a master, up to 100KHz. The various serial clock rates are shown in Table 20–1.

Table 20–1. TWSI Serial Clock Rates

CR2	CR1	CR0	TWSI Clock Selection	TWSI Clock Rate @ SYSCLK=12MHz
0	0	0	SYSCLK/8	1.5 MHz
0	0	1	SYSCLK/16	750 KHz
0	1	0	SYSCLK/32	375 KHz
0	1	1	SYSCLK/64	187.5 KHz
1	0	0	SYSCLK/128	93.75 KHz
1	0	1	SYSCLK/256	46.875 KHz
1	1	0	<b>S1TOF/6</b>	Variable
1	1	1	<b>T3OF/6</b>	Variable

Note:

1. SYSCLK is the system clock.
2. S1TOF is UART1 Baud-Rate Timer Overflow.
3. T3OF is Timer 3 Overflow.
4. In Timer 3 split mode, T3OF is replaced by TL3OF.

**SISTA: 2-wire Serial Interface Status Register**

SFR Page = 0~F

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 TWSI 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.

## 20.5. TWSI Sample Code

(1). Required Function: Set TWSI Master write/read

Assembly Code Example:

```
uCHAR I2C_Read(uCHAR Dev_Addr, uCHAR Reg_Addr)
{
    uCHAR    usData = 0;

    SICON |= STA;
    SICON &= ~SI;
    while(( SICON & SI ) != SI );
    SICON &= ~STA;

    SIDAT = Dev_Addr;                // send device address
    SICON &= ~SI;
    while(( SICON & SI ) != SI );

    SIDAT = Reg_Addr;                // send register address
    SICON &= ~SI;
    while(( SICON & SI ) != SI );

    SICON |= STA;                    // restart
    SICON &= ~SI;
    while(( SICON & SI ) != SI );
    SICON &= ~STA;

    SIDAT = Dev_Addr | 0x01;         // send device address
    SICON &= ~SI;
    while(( SICON & SI ) != SI );

    SICON &= ~SI;
    while(( SICON & SI ) != SI );
    usData = SIDAT;

    SICON |= STO;
    SICON &= ~SI;
    while(( SICON & STO ) == STO );

    return usData;
}

Main:

    MOV    SICON,#( CR2 | ENSI | CR1 | CR0 );           ;enable TWSI and clock source Timer3 overflow
;Timer3_Initial();                                   ;I2C freq is 100K @ MCU run 12MHz.
    MOV    SFRPI,#01h                                ;select SFR page index to "1"

    MOV    T3MOD,#01h
    CLR    T3CON
    MOV    RCAP3H,#0FFh
    MOV    TH3,#0FFh
    MOV    RCAP3L,#0ECh                                ;0x10000 - 0xFFEC = 0x14h = 20d
    MOV    TL3,#0ECh                                  ;20*6 = 120 * 83ns = 9.96us

    SETB   TR3
    MOV    SFRPI,#0h

;I2C_Write(0xA0, 0x30, 0x55);
    ORL    SICON,#STA;
    ANL    SICON,#~SI;

while(( SICON & SI ) != SI );
    ANL    SICON,#~STA;
```

```

    MOV  SIDAT,#0A0h      ;Dev_Addr;           // send device address
    ANL  SICON,#~SI;
while(( SICON & SI ) != SI );

    MOV  SIDAT,#30h      ; Reg_Addr           // send register address
    ANL  SICON,#~SI;
while(( SICON & SI ) != SI );

    MOV  SIDAT,#55h      ; = ucData           // send data
    ANL  SICON,#~SI;
while(( SICON & SI ) != SI );

    ORL  SICON,#STO;
    ANL  SICON,#~SI;
while(( SICON & STO ) == STO );

    CALL delay_10ms;
    P0 = I2C_Read(0xA0, 0x30);

    JMP  $;
}

```

#### C Code Example:

```

uCHAR I2C_Read(uCHAR Dev_Addr, uCHAR Reg_Addr)
{
    uCHAR    usData = 0;

    SICON |= STA;
    SICON &= ~SI;
    while(( SICON & SI ) != SI );
    SICON &= ~STA;

    SIDAT = Dev_Addr;           // send device address
    SICON &= ~SI;
    while(( SICON & SI ) != SI );

    SIDAT = Reg_Addr;          // send register address
    SICON &= ~SI;
    while(( SICON & SI ) != SI );

    SICON |= STA;              // restart
    SICON &= ~SI;
    while(( SICON & SI ) != SI );
    SICON &= ~STA;

    SIDAT = Dev_Addr | 0x01;    // send device address
    SICON &= ~SI;
    while(( SICON & SI ) != SI );

    SICON &= ~SI;
    while(( SICON & SI ) != SI );
    usData = SIDAT;

    SICON |= STO;
    SICON &= ~SI;
    while(( SICON & STO ) == STO );

    return usData;
}

void I2C_Write(uCHAR Dev_Addr, uCHAR Reg_Addr, uCHAR ucData)
{
    SICON |= STA;
    SICON &= ~SI;

```

```

while(( SICON & SI ) != SI );
SICON &= ~STA;

SIDAT = Dev_Addr; // send device address
SICON &= ~SI;
while(( SICON & SI ) != SI );

SIDAT = Reg_Addr; // send register address
SICON &= ~SI;
while(( SICON & SI ) != SI );

SIDAT = ucData; // send data
SICON &= ~SI;
while(( SICON & SI ) != SI );

SICON |= STO;
SICON &= ~SI;
while(( SICON & STO ) == STO );
}

void Timer3_Initial(void)
{
    SFRPI = 1; //select SFR page index to "1"

    T3MOD = 0x10;
    T3CON = 0;
    RCAP3H = 0xFF;
    TH3 = 0xFF;
    RCAP3L = 0xEC; //0x10000 - 0xFFEC = 0x14h = 20d
    TL3 = 0xEC; //20*6 = 120 * 83ns = 9.96us

    TR3 = 1;

    SFRPI = 0;
}

void main()
{
    SICON |= ( CR2 | ENSI | CR1 | CR0 ); //enable TWSI and clock source Timer3 overflow
    Timer3_Initial(); //I2C freq is 100K @ MCU run 12MHz.

    I2C_Write(0xA0, 0x30, 0x55);
    delay_ms(10);
    P0 = I2C_Read(0xA0, 0x30);

    while(1);
}

```

## 21. Keypad Interrupt (KBI)

The Keypad Interrupt function is intended primarily to allow a single interrupt to be generated when Port 2 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 input pins connected to Port 2 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 selected from the port pins on Port 0, Port 2, Port 5 and Port 6 by KBIPS1~0, AUXR1.7~6. The default keypad input is indexed on Port 0.

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. 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.

### 21.1. Keypad 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= XXXX-XX00

7	6	5	4	3	2	1	0
--	--	--	--	--	--	PATN_SEL	KBIF
W	W	W	W	W	W	R/W	R/W

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

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 Px.7 as a cause of a Keypad Interrupt (KBI7).  
 KBMASK.6: When set, enables Px.6 as a cause of a Keypad Interrupt (KBI6).  
 KBMASK.5: When set, enables Px.5 as a cause of a Keypad Interrupt (KBI5).  
 KBMASK.4: When set, enables Px.4 as a cause of a Keypad Interrupt (KBI4).  
 KBMASK.3: When set, enables Px.3 as a cause of a Keypad Interrupt (KBI3).  
 KBMASK.2: When set, enables Px.2 as a cause of a Keypad Interrupt (KBI2).  
 KBMASK.1: When set, enables Px.1 as a cause of a Keypad Interrupt (KBI1).  
 KBMASK.0: When set, enables Px.0 as a cause of a Keypad Interrupt (KBI0).  
*x = 0, 2, 5 or 6.*

**AUXR1: Auxiliary Control Register 1**

SFR Page = 0~F

SFR Address = 0xA2

POR+RESET = 0000-0000

7	6	5	4	3	2	1	0
<b>KBIPS1</b>	<b>KBIPS0</b>	P5SPI	P5S1	P5T2	P6PCA	EXTRAM	DPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: KBIPS1~0, KBI Port Selection [1:0].

KBIPS1~0	KBI7~0
00	P0.7~P0.0
01	P2.7~P2.0
10	P5.7~P5.0
11	P6.7~P6.0

## 21.2. Keypad Interrupt Sample Code

(1). Required Function: Implement a KBI function on P0

Assembly Code Example:

```
ORG    0003Bh
KBI_INT:
    MOV KBCON, #00h           ;Clear KP Interrupt Flag
    MOV KBMASK, #00h         ;Will Disable KP Interrupt

    RETI

main:
    MOV PUCON0, #0Fh         ;enable P0, P1 internal pull high
    ORL EIE1, #20h
    SETBEA

    delay_ms 5

    MOV KBPATN, #0FFh
    MOV KBCON, #00h
    MOV KBMASK, #0FFh       ;Will Enable KP Interrupt

    CLR P1.0
    ORL PCON0, #02h         ;into power down

    CLR P1.1                 ;pull low any P0.x will wake up MCU.

Loop:
    JMP Loop
```

C Code Example:

```
void KBI_ISR(void) interrupt 7
{
    KBCON=0;
    KBMASK=0;
}

void main(void)
{
    PUCON0 = 0x0F;           // Enable P0 ~P1 on-chip pull-up resistor
    EIE1 |= EKB;           // Enable KBI interrupt
    EA = 1;                 // Enable global interrupt

    Delay_5mS();

    KBPATN=0xFF;
    KBCON=0;
    KBMASK=0xFF;
    P10=0;

    PCON0 |= PD;           // Set MCU into power-down mode
    P11=0;
    While(1);
}
```

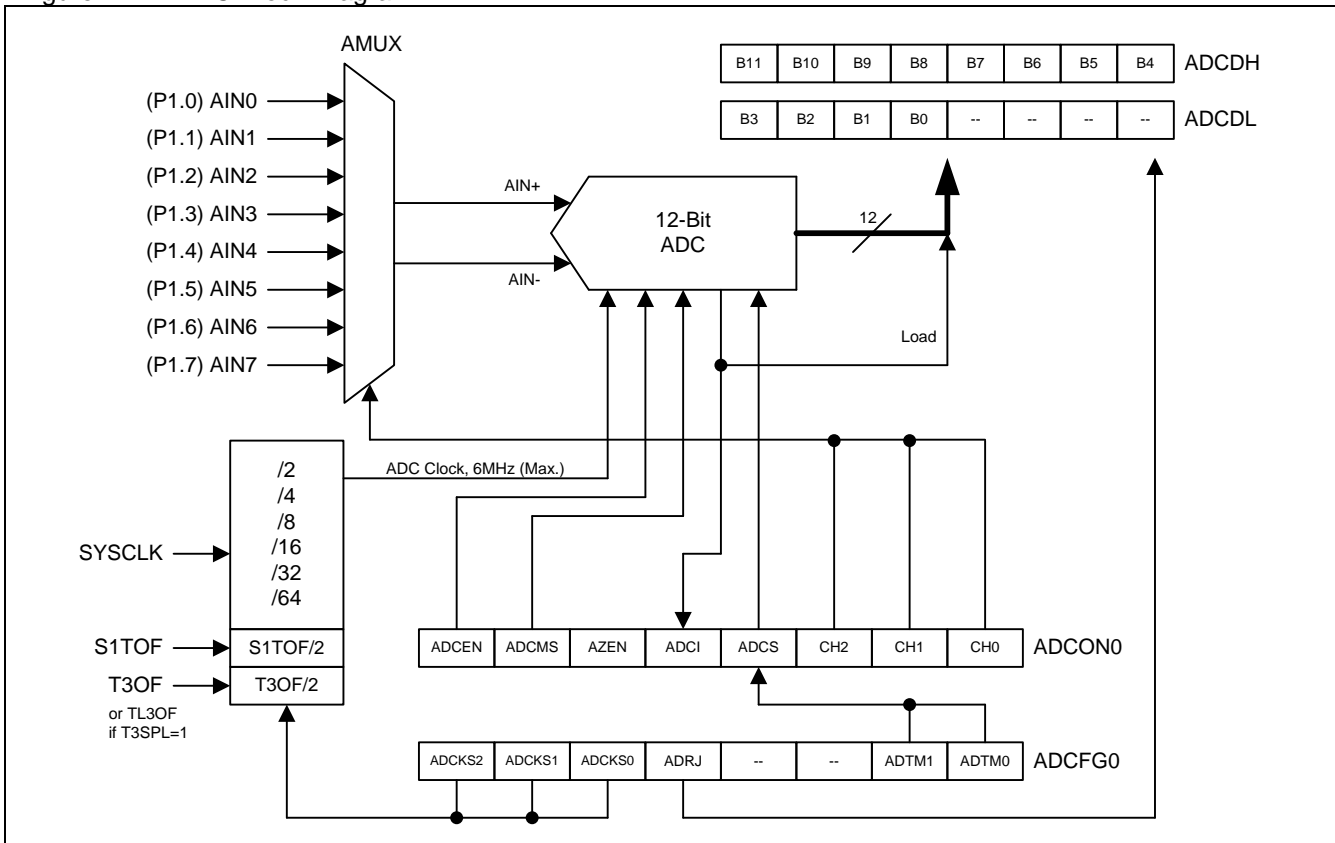


## 22. 12-Bit ADC

The ADC subsystem for the **MG84FG516** consists of an analog multiplexer (AMUX), and a **250 kbps**, **12-bit** successive-approximation-register ADC. The AMUX can be configured via the Special Function Registers shown in [Figure 22–1](#). ADC operates in Single-ended mode or Fully differential mode, and may be configured to measure any of the pins on Port 1. The ADC subsystem is enabled only when the ADCEN 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.

### 22.1. ADC Structure

Figure 22–1. ADC Block Diagram



### 22.2. ADC Operation

ADC has a maximum conversion speed of 250 kbps. The ADC conversion clock is a divided version of the system clock or the timer overflow rate of S1BRG and Timer 3, determined by the ADCKS2~0 bits in the ADCFG0 register. The ADC conversion clock should be no more than 6 MHz.

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, the result is

$$\text{ADC Result} = \frac{V_{IN} \square 4096}{VDD \text{ Voltage}}$$

#### 22.2.1. ADC Input Channels

The analog multiplexer (AMUX) selects the inputs to the ADC, allowing any of the pins on Port 1 to be measured in single-ended mode. The ADC input channels are configured and selected by CHS.2~0 in the ADCON0 register as described in [Figure 22–1](#). The selected pin is measured with respect to GND. In Fully-differential mode, ADC will support 4 channels differential input on Port 1 and output the result value with signed 2's complement format.

### 22.2.2. Starting a Conversion

Prior to using the ADC function, the user should:

- 1) Turn on the ADC hardware by setting the ADCEN bit,
- 2) Select ADCMS to configure ADC for single-ended mode or fully-differential mode
- 3) Configure the ADC input clock by bits ADCKS2, ADCKS1 and ADCKS0,
- 4) Select the analog input channel by bits CHS2, CHS1 and CHS0,
- 5) Configure the selected input (shared with P1) to the Analog-Input-Only mode by P1, P1M0 and P1AIO registers, 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 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 12 bits of conversion result into ADCH and ADCL (according to ADRJ bit) simultaneously. If user sets the ADCS and selects the ADC trigger mode to timer0/3 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.

### 22.2.3. ADC Conversion Time

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 6MHz and it operates a fixed conversion time with 24 ADC clocks. User can configure the ADCKS2~0 in ADCFG0 to specify the conversion rate. For example, if SYSCLK =12MHz and the ADCKS = SYSCLK/2 is selected, then the frequency of the analog input should be no more than 250KHz to maintain the conversion accuracy. (Conversion rate =  $12\text{MHz}/2/24 = 250\text{KHz}$ .)

### 22.2.4. 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 input-only mode. And when an analog signal is applied to the ADCI7~0 pin and the digital input from this pin is not needed, software could set the corresponding pin to analog-input-only in P1AIO to reduce power consumption in the digital input buffer. The port pin configuration for analog input function is described in the Section "[13.2.2 Port 1 Register](#)".

### 22.2.5. 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.

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.

## 22.3. 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	ADCMS	AZEN	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: ADC Conversion Mode Select.

0: Set the ADC as single-ended mode.

1: Set the ADC as differential mode.

Bit 5: AZEN. ADC auto-zero function enable.

0: Disable the ADC auto-zero function.

1: Enable the ADC auto-zero function.

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.

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:

CHS[2:0]	Selected Channel
0 0 0	AIN0 (P1.0)
0 0 1	AIN1 (P1.1)
0 1 0	AIN2 (P1.2)
0 1 1	AIN3 (P1.3)
1 0 0	AIN4 (P1.4)
1 0 1	AIN5 (P1.5)
1 1 0	AIN6 (P1.6)
1 1 1	AIN7 (P1.7)

In Fully-differential mode:

CHS[2:1]	Selected Channel
0 0	AIN0P (P1.0) AIN0M (P1.1)
0 1	AIN1P (P1.2) AIN1M (P1.3)
1 0	AIN2P (P1.4) AIN2M (P1.5)
1 1	AIN3P (P1.6) AIN3M (P1.7)

Note:

1. AIN0P, AIN1P, AIN2P and AIN3P are the positive inputs in fully-differential mode.
2. AIN0M, AIN1M, AIN2M and AIN3M are the negative inputs in fully-differential mode

**ADCFG0: ADC Configuration Register 0**

SFR Page = 0~F

SFR Address = 0xC3 RESET = 0000-XX00

7	6	5	4	3	2	1	0
ADCKS2	ADCKS1	ADCKS0	ADRJ	--	--	ADTM1	ADTM0
R/W	R/W	R/W	R/W	W	W	R/W	R/W

Bit 7~5: ADC Conversion Clock Select bits.

ADCKS[1:0]	ADC Clock Selection
0 0 0	SYSCLK/2
0 0 1	SYSCLK/4
0 1 0	SYSCLK/8
0 1 1	SYSCLK/16
1 0 0	SYSCLK/32
1 0 1	SYSCLK/64
1 1 0	S1TOF/2
1 1 1	T3OF/2

Note:

1. SYSCLK is the system clock.
2. S1TOF is UART1 Baud-Rate Timer Overflow.
3. T3OF is Timer 3 Overflow.
4. In Timer 3 split mode, T3OF is replaced by TL3OF.

Bit 4: ADRJ, ADC result Right-Justified selection.

0: The most significant 8 bits of conversion result are saved in ADCH[7:0], while the least significant 4 bits in ADCL[7:4].

1: The most significant 4 bits of conversion result are saved in ADCH[3:0], while the least significant 8 bits in ADCL[7:0].

**If ADRJ = 0**

**ADCDH: ADC Data 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

**ADCL: 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)	(B1)	(B0)	--	--	--	--
R	R	R	R	R	R	R	R

**If ADRJ = 1**

**ADCDH**

7	6	5	4	3	2	1	0
--	--	--	--	(B11)	(B10)	(B9)	(B8)
R	R	R	R	R	R	R	R

**ADCDL**

7	6	5	4	3	2	1	0
(B7)	(B6)	(B5)	(B4)	(B3)	(B2)	(B1)	(B0)
R	R	R	R	R	R	R	R

When in Single-ended Mode, conversion codes are represented as 12-bit unsigned integers. Inputs are measured from '0' to VREF x 4095/4096. 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 4095/4096	0xFFFF	0x0FFF
VREF x 2048/4096	0x8000	0x0800
VREF x 1024/4096	0x4000	0x0400
VREF x 512/4096	0x2000	0x0200
0	0x0000	0x0000

When in Differential Mode, conversion codes are represented as 12-bit signed 2's complement numbers. Inputs are measured from  $-V_{REF}$  to  $V_{REF} \times 2047/2048$ . 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 (Differential)	ADCDH:ADCDL (ADRJ = 0)	ADCDH:ADCDL (ADRJ = 1)
VREF x 2047/2048	0x7FF0	0x07FF
VREF x 1024/2048	0x4000	0x0400
0	0x0000	0x0000
$-V_{REF} \times 1024/2048$	0xC000	0x0C00
$-V_{REF}$	0x8000	0x0800

Bit 3~2: Reserved. Software must write "0" on these bits when ADCFG0 is written.

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	Timer 3 overflow

**P1AIO: Port 1 Analog Input Only**

SFR Page = 0~F

SFR Address = 0x92

RESET = 0000-0000

7	6	5	4	3	2	1	0
P17AIO	P16AIO	P15AIO	P14AIO	P13AIO	P12AIO	P11AIO	P10AIO
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

0: Port pin has digital and analog input capability.

1: Port pin only has analog input only. The corresponding Port PIN Register bit will always read as zero when this bit is set.

## 22.4. ADC Sample Code

```
--- (8051 Assembly) -----
ADCR0_H DATA 0x40 ;high 4 bit
ADCR0_L DATA 0x41 ;LSB

// initial ADC
MOV ADCON0,#81H ; Enable ADCEN, Select single-end mode, Select AIN1 (P1.1)
MOV ADCFG0,#10H ; ADC clock = SYSCLK/2, ADRJ=1, Set ADCS to Start ADC conversion
ORL P1AIO,#02H ; configure P1.1 as Input-Only Mode

// start ADC conversion
ORL ADCON0,#08H ; Start of conversion
wait_loop: ; wait until ADCl=1 conversion completed
MOV ACC,ADCON0
JNB ACC.4,wait_loop

ANL ADCON0,#0E7H ; clear ADC interrupt flag

MOV ADCR0_H, ADCDH ;now, the 12-bit ADC result is in the ADCH and ADCL.
MOV ADCR0_L, ADCDL

--- (C language) -----
unsigned int ADCR0;

// initial ADC
ADCON0 = 0x81; // Enable ADCEN, Select single-end mode, Select AIN1 (P1.1)
ADCFG0 = 0x10; // ADC clock = SYSCLK/2, ADRJ=1 (right-justified), Set ADCS to Start ADC conversion
P1AIO |= 0x02; // configure P1.1 as Input-Only Mode

// start ADC conversion
ADCON0 |= 0x08; // Start of conversion
while ((ADCON0 & 0x10) == 0x00); // wait until ADCl=1 conversion completed
ADCON0 &= ~0x10; // clear ADC interrupt flag

ADCR0 = ADCDH << 8; // for right-justified
ADCR0 |= ADCDL;
```

## 23. Universal Serial Bus (USB)

**MG84FG516** implements a USB full-speed function which is fully compliant with USB specification 2.0 and 1.1 to support various USB applications. The USB block contains a on-chip 3.3V regulator, a USB transceiver which transmits and receives differential USB signal, a 1K bytes FIFO which is a temporary store data unit, and a USB Core to perform NRZI encoding and decoding, bit stuffing, CRC generation and checking, serial-parallel data transforming, data flow between 1K bytes FIFO and CPU, USB special function register and setting, and communication with CPU by using MOVX instruction or accessing USBADR/USB DAT in CPU SFRs directly.

Before using **MG84FG516** USB function, we assume that user has a comprehensive understanding on USB protocol and application. So, the following descriptions in this chapter would not focus on the detail of USB specification. If user is interesting in USB specification, user can download the latest version of USB specification document from the USB official website <http://www.usb.org/home>.

Megawin Inc. also offer a development kit which contain sample code, C language library and application note on the website <http://www.megawin.com.tw> to help user to implement design more quickly and easily.

Note:

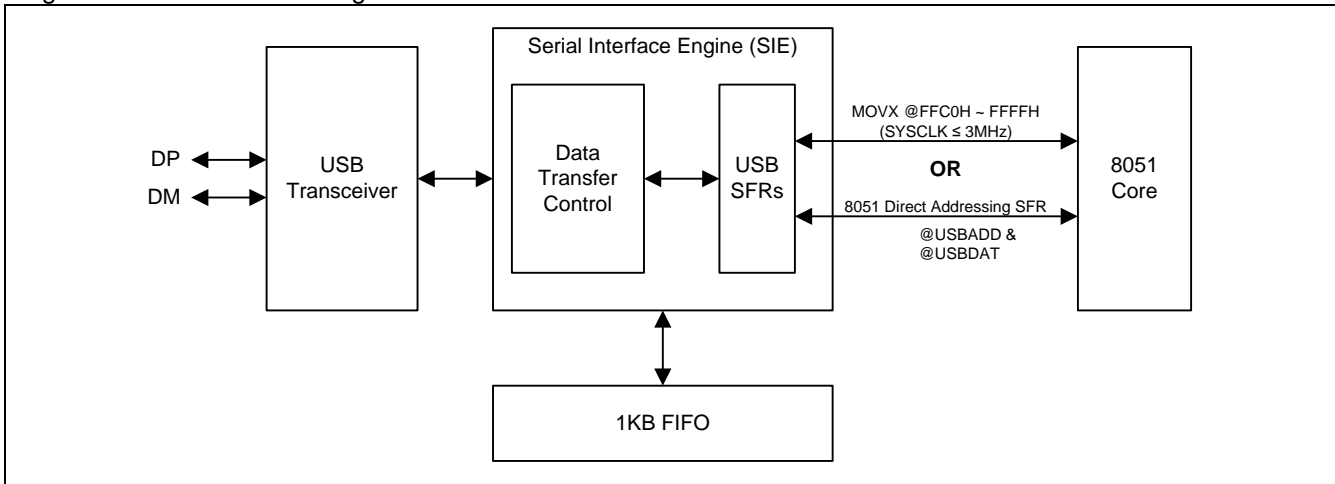
**MG84FG516** can't be used as a USB HOST device or USB OTG device.

### 23.1. Features

- Compliant with USB specification v1.1/v2.0.
- Supports USB full speed 12M bps serial data transmission
- Supports USB suspend/resume and remote wake-up
- 1K bytes FIFO for USB endpoint-shared buffer
- Support ping-pong mode (dual bank)
- 64 bytes FIFO for EP0 Control In/Out buffer
- 64 bytes FIFO for EP1 Interrupt/Bulk IN buffer
- 64 bytes FIFO for EP1 Interrupt/Bulk OUT buffer
- 64 bytes FIFO for EP2 Interrupt/Bulk IN buffer (Ping-Pong Pair 1)
- 64 bytes FIFO for EP3 Interrupt/Bulk IN buffer (Ping-Pong Pair 1)
- 64 bytes FIFO for EP2 Interrupt/Bulk OUT buffer (Ping-Pong Pair 2)
- 64 bytes FIFO for EP3 Interrupt/Bulk OUT buffer (Ping-Pong Pair 2)
- 96 bytes FIFO for EP4 Interrupt/Bulk/Isochronous IN buffer (Ping-Pong Pair 3)
- 96 bytes FIFO for EP5 Interrupt/Bulk/Isochronous IN buffer (Ping-Pong Pair 3)
- 192 bytes FIFO for EP4 Interrupt/Bulk/Isochronous OUT buffer (Ping-Pong Pair 4)
- 192 bytes FIFO for EP5 Interrupt/Bulk/Isochronous OUT buffer (Ping-Pong Pair 4)

### 23.2. Block Diagram

Figure 23–1. USB Block Diagram



### 23.3. FIFO Management

There are 1K bytes FIFO for temporary USB data store unit accessed by USB core and a total of 11 endpoints are available in **MG84FG516** as shown in [Figure 23–2](#). Endpoint 0 supports a bi-direction control transfer. Endpoint 1/2/3/4/5 supports Interrupt/Bulk IN/OUT transaction. Endpoint 4/5 also has additional function to support Isochronous IN/OUT transaction. The maximum data packet size can be up to 64 bytes for endpoint 0/1/2/3 IN/OUT function, 96 bytes for endpoint 4/5 IN function, and 192 bytes for endpoint 4/5 OUT function.

[Figure 23–3](#) shows a USB Ping-Pong Pair FIFO configuration. In this configuration, two 64/96 bytes FIFO combined to a ping-pong pair will be dedicated to endpoint 2/4 IN function to improve data throughput performance. Endpoint 3/5 has two 64/192 bytes FIFO for OUT function.

*Note: the Ping-Pong FIFO function is under verifying.*

Figure 23–2. Non Ping-Pong Pair

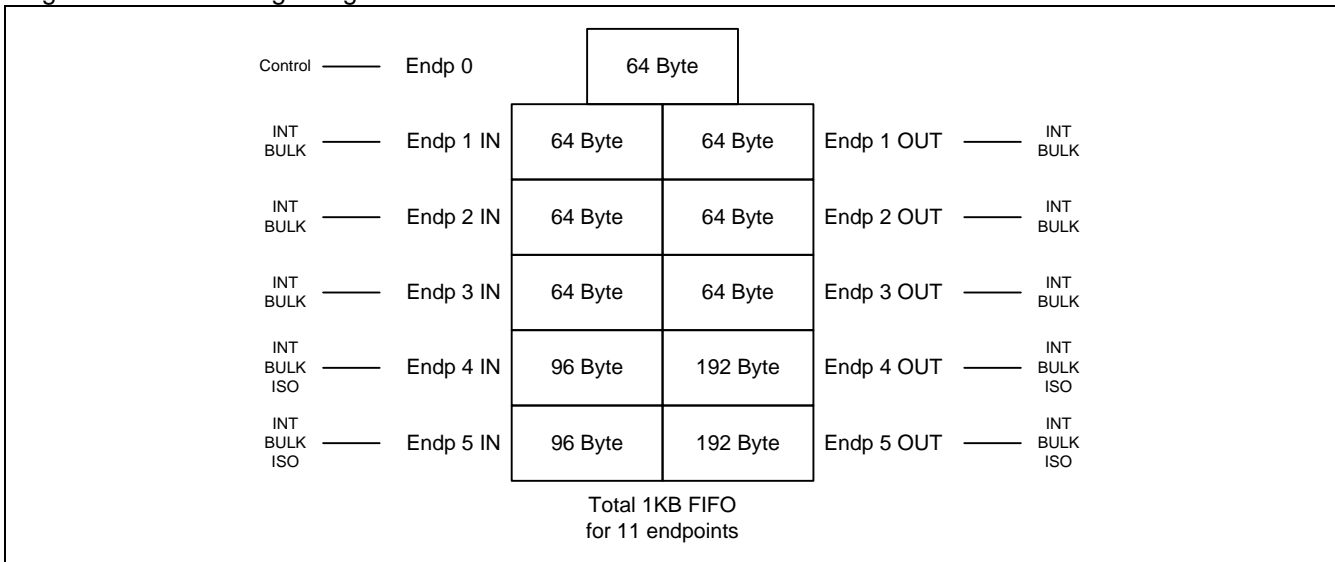
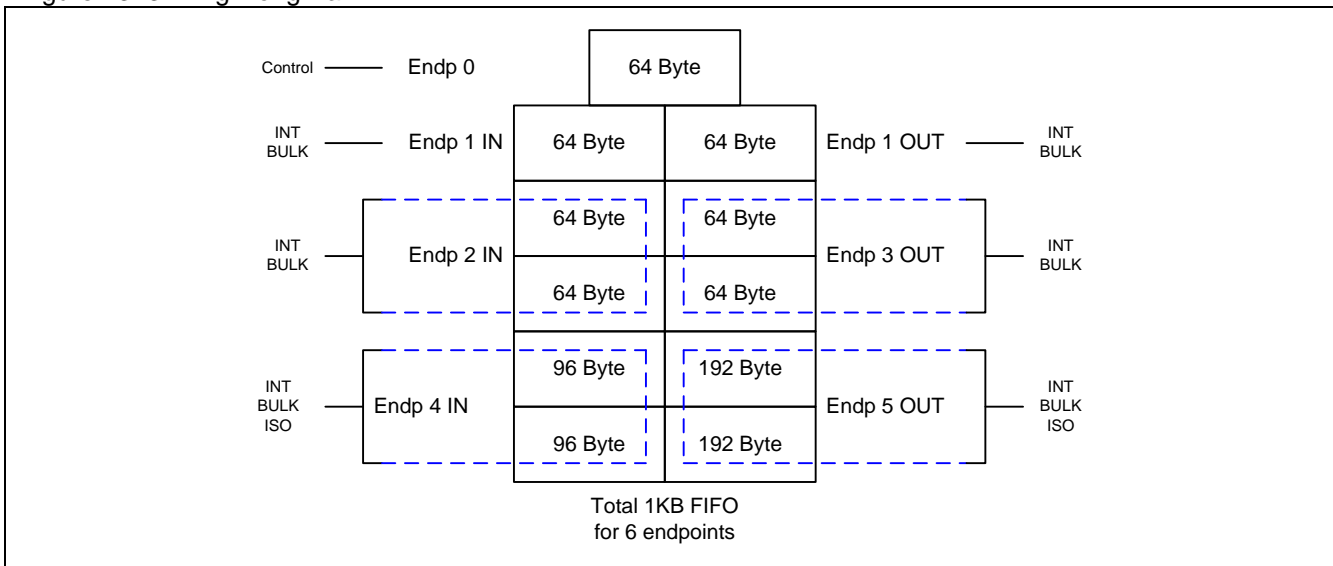




Figure 23–3. Ping-Pong Pair



## 23.4. Access USB 1K FIFO

If ENUSB is set to enable USB function, a 1K bytes FIFO would be a dedicated buffer for USB application. But in most application case, it is not necessary and wasted that all 1K bytes FIFO would reserve for USB buffer, the unused USB buffer can provide MCU additional data RAM which is access through USB re-directed mechanism. Figure 23–4 shows the USB FIFO access mechanism to share the USB FIFO for MCU application.

### Access USB 1K buffer through USB SFR re-direct flow:

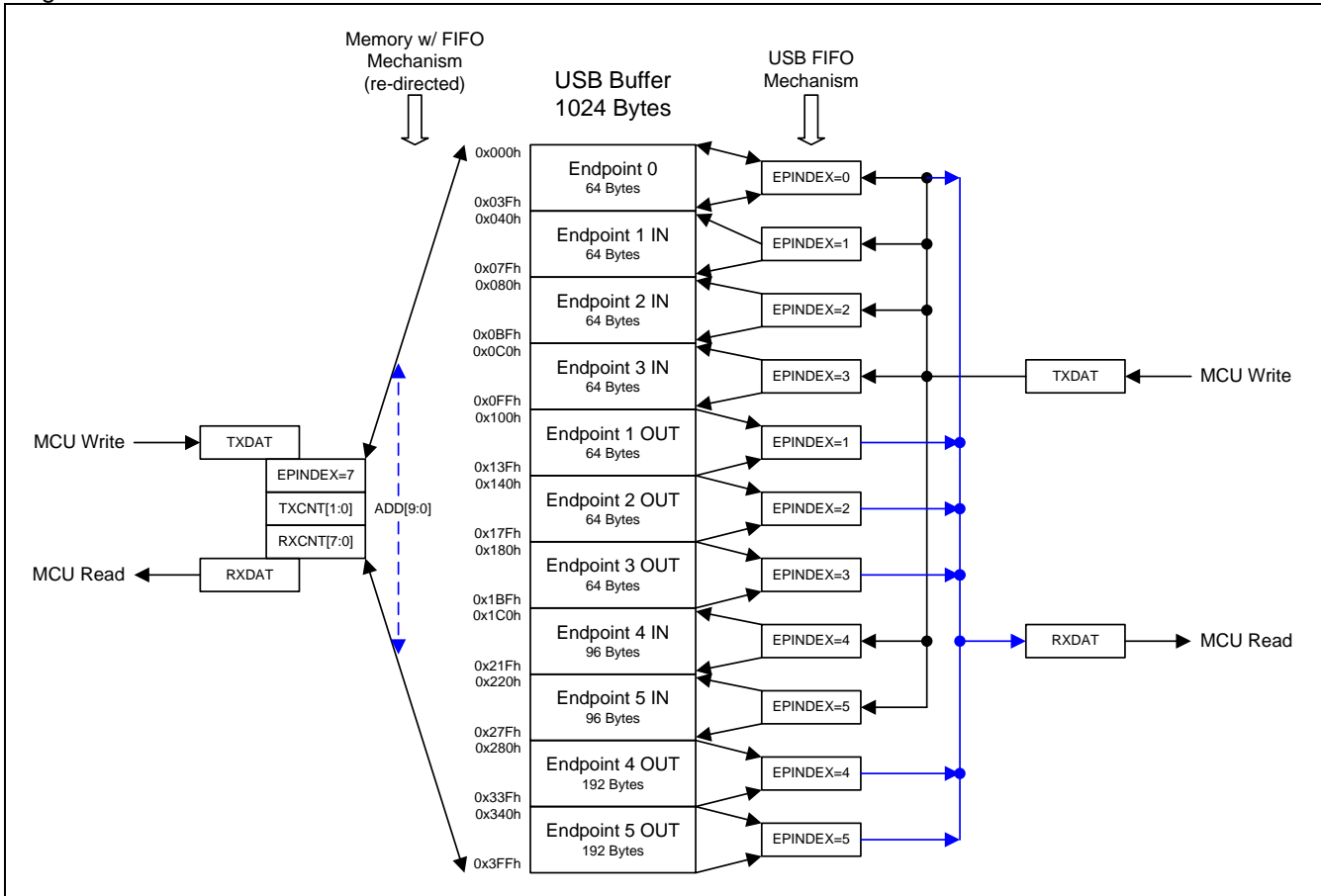
#### Write data into 1K buffer

1. Write EPINDEX=7
2. Write TXCNT
3. Write RXCNT (directly mapping {TXCNT[1:0], RXCNT[7:0]} to 1K buffer Address space)
4. Write data into TXDAT
5. Repeat to step 2 (step 2 & 3 can be skipped, if the next buffer address is increased)

#### Read data from 1K buffer

1. Write EPINDEX=7
2. Write TXCNT
3. Write RXCNT (directly mapping {TXCNT[1:0], RXCNT[7:0]} to 1K buffer Address space)
4. Read data from RXDAT
5. Repeat to step 2 (step 2 & 3 can be skipped, if the next buffer address is increased)

Figure 23–4. USB FIFO access mechanism



### 23.5. USB Initial

To activate the USB operation, the user should enable Clock Multiplier (CKM) unit by setting ENCKM bit, wait 100us for CKM ready to work, and then enable USB function by setting ENUSB bit. Clearing bit 'ENUSB' will deactivate the USB operation and let the USB function enter its power-down mode. These relevant control bits are contained in the CKCON0 register, as follows.

#### CKCON0: Clock Control Register 0

SFR Page = 0~F & P

SFR Address = 0xC7

RESET = 0001-x000

7	6	5	4	3	2	1	0
ENUSB	ENCKM	CKMIS1	CKMIS0	-	SCKS2	SCKS1	SCKS0
R/W	R/W	R/W	R/W	W	R/W	R/W	R/W

Bit 7: ENUSB, Enable USB clock and whole USB function.

0: Disable USB clock and USB function.

1: Enable USB clock and USB function.

Bit 6: ENCKM, Enable clock multiplier (X8)

0: Disable the X8 clock multiplier.

1: Enable the X8 clock multiplier.

Bit 5~4: CKMIS1 ~ CKMIS0, Multiplier Input Clock Selection.

CKMIS[1:0]	Multiplier Input Clock Selection
0 0	6MHz input
0 1	12MHz input
1 0	24MHz input
1 1	36MHz input

Note:

Before using USB function, the setting of CKM must be correct to provide proper clock for USB communication. Please refer Section “9 System Clock” to get the information about the setting of CKM.

## 23.6. Access USB SFR

There are two ways that software can use to read/write USB SFR in **MG84FG516**. One is using MOVX instruction to access USB SFR like MG84FL54B. Another is using USB read/write procedure by accessing USBADR/USBDAT in CPU SFR.

Note:

**If using MOVX instruction to access USB SFR, the system clock (SYSCLK) must not be more than 3MHz. When using USBADR/USBDAT access procedure, it has no the criteria on SYSCLK speed.**

### 23.6.1. Through MOVX Instruction

The special function registers which are dedicated to the USB operation are grouped in the external memory address space and share the addresses 0xFF00 to 0xFFFF with the physical external data memory. The bit ‘EXTRAM’ (AUXR1.1) determines accessing to the USB SFRs or the physical external data memory. That is, before accessing the USB SFRs, the user must clear EXTRAM bit and then use the instruction “**MOVX @DPTR**” to access these SFRs.

#### AUXR1: Auxiliary Control Register 1

SFR Page = 0~F

SFR Address = 0xA2

POR+RESET = 0000-0000

7	6	5	4	3	2	1	0
KBIPS1	KBIPS0	P5SPI	P5S1	P5T2	P6PCA	<b>EXTRAM</b>	DPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 1: EXTRAM, External data RAM enable.

0: Enable on-chip expanded data RAM (XRAM 4096 bytes)

1: Disable on-chip expanded data RAM.

### 23.6.2. Through USBADR/USBDAT

Table 23–1 shows the USB SFR and their indirect address from 00H~FFH. USB SFR can be indirectly accessed according to a 6-bit address hold in USBADR. Read/Write USBDAT will target the register indicated by USBADR.

#### USB Write SFR procedure

1. Wait for UBSY=0
2. Write USB SFR address into USBADR
3. Write data into USBDAT
4. Repeat to step1 to write next data(step 2 can be skipped, when writing to the same USB SFR address)

#### USB Read SFR procedure

1. Wait for UBSY=0
2. Write USB SFR address into USBADR
3. Read data from USBDAT
4. Repeat to step 1 to read next data(step 2 can be skipped, when reading from the same USB SFR address)

**USBADR: USB indirect Address Register**

SFR Page = 0~F

SFR Address = 0xAB POR+RESET = 0x00-0000

7	6	5	4	3	2	1	0
UBSY	--	USFRA5	USFRA4	USFRA3	USFRA2	USFRA1	USFRA0
R	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7: UBSY, USB core BUSY flag.

0: The data access request in USB core is finished.

1: USB core is **BUSY** on accessing software read/write request.

Bit 5~0: USFRA[5:0], USB SFR indirect address.

**USBDAT: USB Data Register**

SFR Page = 0~F

SFR Address = 0xAA POR+RESET = xxxx-xxxx

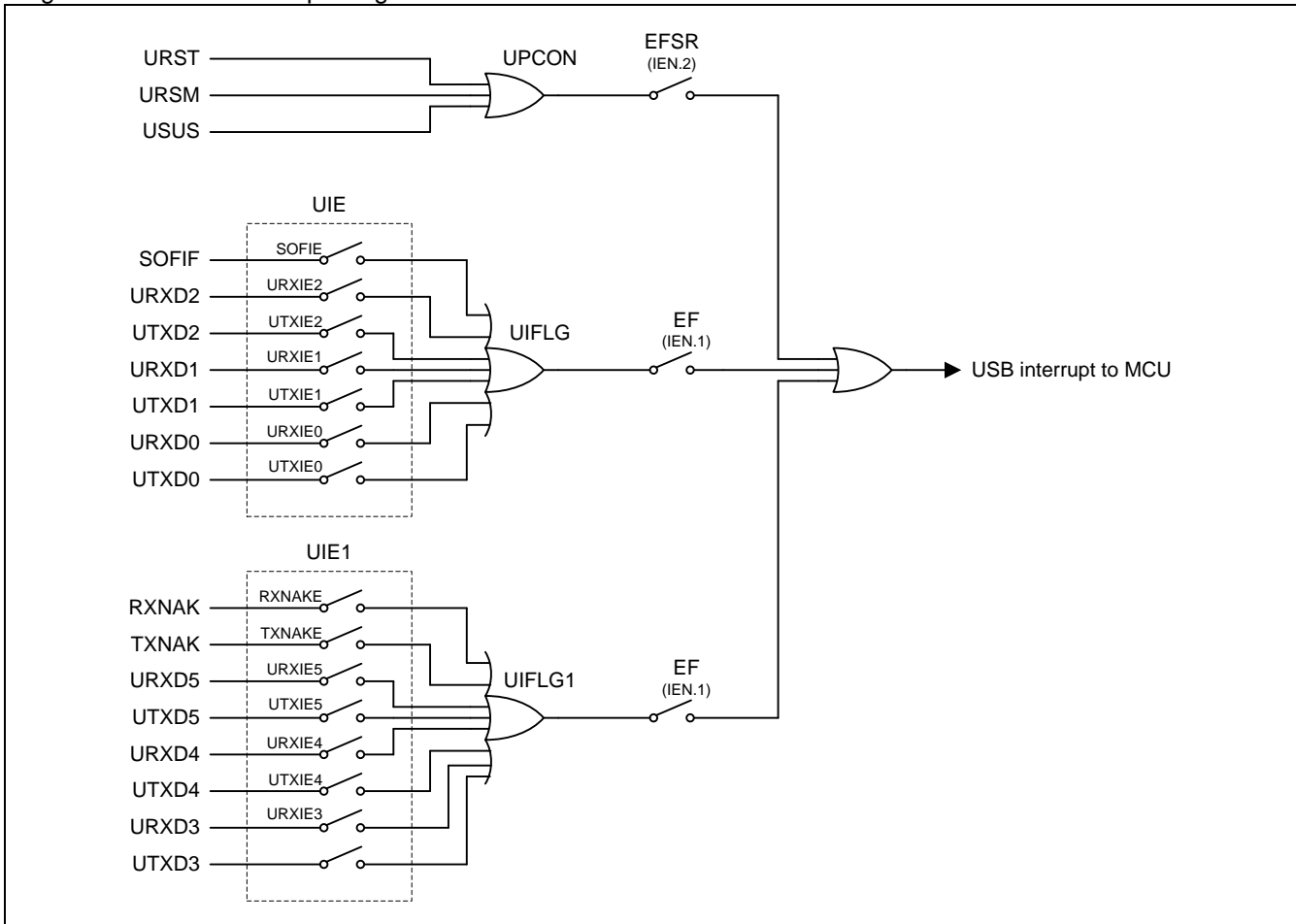
7	6	5	4	3	2	1	0
UDAT7	UDAT6	UDAT5	UDAT4	UDAT3	UDAT2	UDAT1	UDAT0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~0: UDAT[7:0], USB SFR Data.

### 23.7. USB Interrupt

Figure 23–5 shows the USB interrupt structure and there are 18 interrupt flags which are located in USB SFRs shown in Section “23.8.2 USB Function SFR Bit Assignment”. The USB interrupt is generated on the combination of USB event flags and USB endpoint flags contained in USB SFRs. The USB event flags include USB reset flag (URST), USB resume flag (URSM) and USB suspend flag (USUS) can indicate that the upstream host has sent the USB reset, resume or suspend event on USB bus to device. The USB endpoint flags, as UTXDx and URXDx (x=0~5), show the USB data transmission or reception of respective endpoint had been done by USB transceiver. The associated interrupt enable bits are located in UIE, UIE1 and IEN registers.

Figure 23–5. USB Interrupt Diagram



## 23.8. USB Special Function Registers

All USB SFRs would be reset by the reset sources as listed in Section “11 System Reset” (SYSRST) and the most of USB SFRs would be reset when device receives the USB reset event (USBRST) except DCON0, DCON1, IEN, SIOCTL registers and CONEN bit in UPCON register.

### 23.8.1. USB Function SFR Mapping

Table 23–1. USB Function SFR Mapping

	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	
FFF8H	--	--	--	--	--	--	--	--	FFFFH
FFF0H	--	<b>EPINDEX</b>	<b>TXSTAT</b>	<b>TXDAT</b>	<b>TXCON</b>	--	<b>TXCNT</b>	--	FFF7H
FFE8H	--	--	--	--	--	--	--	--	FFE7H
FFE0H	--	<b>EPCON</b>	<b>RXSTAT</b>	<b>RXDAT</b>	<b>RXCON</b>	--	<b>RXCNT</b>	--	FFE7H
FFD8H	<b>UADDR</b>	<b>IEN</b>	<b>UIE</b>	<b>UIFLG</b>	<b>UIE1</b>	<b>UIFLG1</b>			FFDFH
FFD0H	--	--	--	--	--	--	--	--	FFD7H
FFC8H	--	<b>UPCON</b>	--	--	--	--	--	--	FFCFH
FFC0H	<b>DCON0</b>	<b>DCON1</b>	<b>SIOCTL</b>	--	--	--	--	--	FFC7H
	0/8	1/9	2/A	3/B	4/C	5/D	6/E	7/F	

## 23.8.2. USB Function SFR Bit Assignment

Table 23–2. USB Function SFR Bit Assignment

SYMBOL	DESCRIPTION	"MOVX" ADDR	BIT SYMBOL								RESET VALUE
			Bit-7	Bit-6	Bit-5	Bit-4	Bit-3	Bit-2	Bit-1	Bit-0	
DCON0	Device Control Register 0	C0H	EP5O96	EP4O96	<b>SCWKP</b>	--	--	XCVPD	--	--	0000000B
<b>DCON1</b>	Device Control Register 1	C1H	--	--	SETNO	STLDEN	NAKEP1	NAKEP0	RPD1M	--	0000000B
UADDR	USB Address Register	D8H	--	UADD6	UADD5	UADD4	UADD3	UADD2	UADD1	UADD0	x0000000B
UPCON	USB Power Control Register	C9H	CONEN	--	URWU	--	<b>FRST</b>	URST	URSM	USUS	0x0x0000B
IEN	Interrupt Enable Register	D9H	--	--	--	--	--	EFSR	EF	--	xxxxx00xB
UIE	USB Interrupt Enable Register	DAH	SOFIE	--	URXIE2	UTXIE2	URXIE1	UTXIE1	URXIE0	UTXIE0	0000000B
UIFLG	USB Interrupt Flag Register	DBH	SOFIF	--	URXD2	UTXD2	URXD1	UTXD1	URXD0	UTXD0	0000000B
UIE1	USB Interrupt Enable Register 1	DCH	RXNAKE	TXNAKE	URXIE5	UTXIE5	URXIE4	UTXIE4	URXIE3	UTXIE3	0000000B
UIFLG1	USB Interrupt Flag Register 1	DDH	RXNAK	TXNAK	URXD5	UTXD5	URXD4	UTXD4	URXD3	UTXD3	0000000B
EPINDEX	Endpoint Index Register	F1H	--	--	--	--	--	EPINX2	EPINX1	EPINX0	xxxxx000B
EPCON	Endpoint Control Register	E1H	RXSTL	TXSTL	RXDBM	TXDBM	RXISO	RXEPEN	TXISO	TXEPEN	0000000B
RXSTAT	Endpoint Receive Status Register	E2H	RXSEQ	RXSETUP	STOVW	EDOVW	RXSOVW	ISOVW	--	--	000000xxB
RXDAT	FIFO Receive Data Register	E3H	RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0	xxxxxxxxxB
RXCON	FIFO Receive Control Register	E4H	RXCLR	--	--	RXFFRC	--	--	--	--	0xx0xxxxB
RXCNT	FIFO Receive Byte Count Register	E6H	RXBC7	RXBC6	RXBC5	RXBC4	RXBC3	RXBC2	RXBC1	RXBC0	00000000B
TXSTAT	Endpoint Transmit Status Register	F2H	TXSEQ	--	--	--	TXSOVW	--	--	--	0xxx0xxxB
TXDAT	FIFO Transmit Data Register	F3H	TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0	xxxxxxxxxB
TXCON	FIFO Transmit Control Register	F4H	TXCLR	--	--	TXFFRC	--	--	--	--	0xx0xxxxB
TXCNT	FIFO Transmit Byte Count Register	F6H	TXBC7	TXBC6	TXBC5	TXBC4	TXBC3	TXBC2	TXBC1	TXBC0	00000000B
SIOCTL	Serial I/O Control Register	C2H	DPI	DMI	--	--	--	--	--	--	xxxxxxxxxB

**DCON0: Device Control Register 0**

SFR Address = 0xC0H SYSRST= 0000-x00x

7	6	5	4	3	2	1	0
EP5O96	EP4O96	SCWKP	--	--	XCVPD	--	--
R/W	R/W	R/W	W	W	R/W	W	W

Bit 7: EP5O96-- Endpoint 5 OUT 96byte buffer size.

0: 96bytes for endpoint 5 IN buffer and 192bytes for endpoint 5 OUT buffer.

1: 192bytes for endpoint 5 IN buffer and 96bytes for endpoint 5 OUT buffer.

Bit 6: EP4O96-- Endpoint 4 OUT 96byte buffer size.

0: 96bytes for endpoint 4 IN buffer and 192bytes for endpoint 4 OUT buffer.

1: 192bytes for endpoint 4 IN buffer and 96bytes for endpoint 4 OUT buffer.

Bit 5: SCWKP-- Software Control Remote-wakeup.

0: Remote-wakeup length will be decided by hardware setting.

1: Remote-wakeup length will be controlled by URWU = 1.

Bit 4~3: Reserved. Software must write "0" on these bits when DCON0 is written.

Bit 2: XCVPD-- USB XCVR Power down

Used for test only, set it to "1" to make USB transceiver into power down mode. Default is cleared to "0".

Bit 1~0: Reserved. Software must write "0" on these bits when DCON0 is written.

**DCON1: Device Control Register 1**

SFR Address = 0xC1H SYSRST= 0000-0000

7	6	5	4	3	2	1	0
--	--	SETNO	STLDEN	NAKEP1	NAKEP0	RPD1M	--
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 DCON1 is written.**

Bit 5: SETNO-- Set No-response in EP0 IN/OUT transaction.

0: Device will send ACK/NAK/STALL packet in IN/OUT transaction.

1: Device will be just only response ACK packet with SETUP transaction but no response with EP0 IN/OUT transaction.

Note:

*This bit will be clear by HW when Device receive an SETUP token.*

Bit 4: STLDEN-- STALL Done enable.

0: Disable IN/OUT STALL transaction flag setting.

1: IN/OUT STALL transaction will set TXD0/RXD0 in FIFLG.

Bit 3~2: NAKEP[1:0]-- Endpoint NAK done select.

00: RXNAK and TXNAK dedicated for endpoint 0

01: RXNAK and TXNAK dedicated for endpoint 1

10: RXNAK and TXNAK dedicated for endpoint 2

11: RXNAK and TXNAK dedicated for endpoint 3

Bit 1: RPD1M, two individual pull-down Resistor enabled on DP and DM.

0: Disable the pull-down resistors on DP and DM.

1: Enable the pull-down resistors on DP and DM. The resistance is about 750K~1MΩ

Bit 0: Reserved. Software must write "0" on this bit when DCON1 is written.

**UADDR: USB Function Address Register**

SFR Address = 0xD8H SYSRST/USBRST= X000-0000

7	6	5	4	3	2	1	0
--	UADD6	UADD5	UADD4	UADD3	UADD2	UADD1	UADD0
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 UADDR is written.

Bit 6–0: UADD[6:0]-- USB Function Address.

This register holds the address for the USB function. During bus enumeration, it is written with a unique value assigned by the host.

**UPCON: USB Power Control Register**

SFR Address = 0xC9H

SYSRST/USBRST= 0X0X-X000

7	6	5	4	3	2	1	0
CONEN	--	URWU	--	FRST	URST	URSM	USUS
W	W	R/W	W	W	R/W	R/W	R/W

Bit 7: CONEN-- USB Connect Enable.

Default is cleared to '0' after reset. FW should set '1' to enable connection to upper host/hub.

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

Bit 5: URWU-- USB Remote Wake-Up Trigger.

0: If SWCKP=0, this bit will be cleared by hardware when remote-wakeup is completed. If SWCKP=1, this bit will be cleared by firmware to stop device driving a remote wake-up on the USB bus. Don't set this bit unless the function is suspended

1: This bit is set by the firmware to initiate a remote wake-up on the USB bus when CPU is wake-up by external trigger.

Note:

Set by firmware to make driving resume signaling to the host. Don't set this bit unless the function is suspended (USUS=1 and URSM=0).

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

Bit 3: FRST-- Function interface Reset Flag of USB device.

Set by hardware when the **MG84FG516** detects the USB device interface in USB reset duration. If this bit is set, the flag will **NOT** generate an interrupt to uC. It would be cleared by firmware writing '1' to it.

Bit 2: URST-- USB Reset Flag.

Set by hardware when the function detects the USB bus reset. If this bit is set, the chip will generate an USRT interrupt to uC. It would be cleared by firmware when serving the USB reset interrupt. This bit is cleared when firmware writes '1' to it.

Bit 1: URSM-- USB Resume Flag.

Set by hardware when the function detects the resume state on the USB bus. If this bit is set, the chip will generate an interrupt to uC. It would be cleared by firmware when serving the function resume interrupt. This bit is cleared when firmware writes '1' to it.

Bit 0: USUS-- USB Suspend Flag.

Set by hardware when the function detects the suspend state on the USB bus. If this bit is set, the chip will generate an interrupt to uC. During the function suspend interrupt-service routine, firmware should clear this bit before enter the suspend mode. This bit is cleared when firmware writes '1' to it.

**IEN: Interrupt Enable Register**

SFR Address = 0xD9H

SYSRST= XXXX-X00X

7	6	5	4	3	2	1	0
--	--	--	--	--	EFSR	EF	--
W	W	W	W	W	R/W	R/W	W

Bit 7–3: Reserved. Software must write “0” on these bits when IEN is written.

Bit 2: EFSR-- Enable USB Function's Suspend/Resume interrupt.

If this bit is set, enables function's interrupt of FPCON events. Function suspend/resume/remote-wakeup/USB-reset interrupt enable bit. This bit doesn't be reset USB\_RESET. Default is cleared.

Bit 1: EF-- Enable USB Function's interrupt Flag.

If this bit is set, enables function's interrupt of UIFLG. Transmit/receive done interrupt enable bit for USB function endpoints. This bit doesn't be reset by USB\_RESET. Default is cleared.

Bit 0: Reserved. Software must write "0" on this bit when IEN is written.

**UIE: USB Interrupt Enable Register**

SFR Address = 0xDAH

SYSRST/USBRST= 0000-0000

7	6	5	4	3	2	1	0
SOFIE	--	URXIE2	UTXIE2	URXIE1	UTXIE1	URXIE0	UTXIE0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit7: SOFIE-- Host SOF received Interrupt Enable.

If this bit is set, enables the Host SOF received interrupt. Default is cleared.

Bit 6: Reserved. Software must write "0" on this bit when UIE is written.

Bit5: URXIE2-- USB Function Receive Interrupt Enable 2.

If this bit is set, enables the receive done interrupt for USB endpoint 2 (URXD2). Default is cleared.

Bit4: UTXIE2-- USB Function Transmit Interrupt Enable 2.

If this bit is set, enables the transmit done interrupt for USB endpoint 2 (UTXD2). Default is cleared.

Bit3: URXIE1-- USB Function Receive Interrupt Enable 1.

If this bit is set, enables the receive done interrupt for USB endpoint 1 (URXD1). Default is cleared.

Bit2: UTXIE1-- USB Function Transmit Interrupt Enable 1.

If this bit is set, enables the transmit done interrupt for USB endpoint 1 (UTXD1). Default is cleared.

Bit1: URXIE0-- USB Function Receive Interrupt Enable 0.

If this bit is set, enables the receive done interrupt for USB endpoint 0 (URXD0). Default is cleared.

Bit0: UTXIE0-- USB Function Transmit Interrupt Enable 0.

If this bit is set, enables the transmit done interrupt for USB endpoint 0 (UTXD0). Default is cleared.

**UIFLG: USB Interrupt Flag Register**

SFR Address = 0xDBH

SYSRST/USBRST= 0000-0000

7	6	5	4	3	2	1	0
SOFIF	--	URXD2	UTXD2	URXD1	UTXD1	URXD0	UTXD0
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit7: SOFIF-- Host SOF received Interrupt Flag.

This bit is set by hardware when detected a host SOF. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit 6: Reserved. Software must write "0" on this bit when UIFLG is written.

Bit5: URXD2-- USB Transmit Done Flag for endpoint 2.

This bit is set by hardware when detected a receive done on endpoint 2. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit4: UTXD2-- USB Transmit Done Flag for endpoint 2.

This bit is set by hardware when detected a transmit done on endpoint 2. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit3: URXD1-- USB Receive Done Flag for endpoint 1.

This bit is set by hardware when detected a receive done on endpoint 1. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit2: UTXD1-- USB Transmit Done Flag for endpoint 1.

This bit is set by hardware when detected a transmit done on endpoint 1. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit1: URXD0-- USB Receive Done Flag for endpoint 0.

This bit is set by hardware when detected a receive done on endpoint 0. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit0: UTXD0-- USB Transmit Done Flag for endpoint 0.

This bit is set by hardware when detected a transmit done on endpoint 0. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

### **UIE1: USB Interrupt Enable Register 1**

SFR Address = 0xDCH

SYSRST/USBRST= 0000-0000

7	6	5	4	3	2	1	0
<b>RXNAKE</b>	<b>TXNAKE</b>	URXIE5	UTXIE5	URXIE4	UTXIE4	URXIE3	UTXIE3
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit7: RXNAKE-- Enable RX NAK interrupt on NAKEP[1:0] indexed.

If this bit is set, enables the RXNAK interrupt for NAKEP[1:0] indexed endpoint. Default is cleared.

Bit6: TXNAKE-- Enable TX NAK interrupt on NAKEP[1:0] indexed.

If this bit is set, enables the TXNAK interrupt for NAKEP[1:0] indexed endpoint. Default is cleared.

Bit5: URXIE5-- USB Function Receive Interrupt Enable 5.

If this bit is set, enables the receive done interrupt for USB endpoint 5 (URXD5). Default is cleared.

Bit4: UTXIE5-- USB Function Transmit Interrupt Enable 5.

If this bit is set, enables the transmit done interrupt for USB endpoint 5 (UTXD5). Default is cleared.

Bit3: URXIE4-- USB Function Receive Interrupt Enable 4.

If this bit is set, enables the receive done interrupt for USB endpoint 4 (URXD4). Default is cleared.

Bit2: UTXIE4-- USB Function Transmit Interrupt Enable 4.

If this bit is set, enables the transmit done interrupt for USB endpoint 4 (UTXD4). Default is cleared.

Bit1: URXIE3-- USB Function Receive Interrupt Enable 3.

If this bit is set, enables the receive done interrupt for USB endpoint 3 (URXD3). Default is cleared.

Bit0: UTXIE3-- USB Function Transmit Interrupt Enable 3.

If this bit is set, enables the transmit done interrupt for USB endpoint 3 (UTXD3). Default is cleared.

### **UIFLG1: USB Interrupt Flag Register 1**

SFR Address = 0xDDH

SYSRST/USBRST= 0000-0000

7	6	5	4	3	2	1	0
<b>RXNAK</b>	<b>TXNAK</b>	URXD5	UTXD5	URXD4	UTXD4	URXD3	UTXD3
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit7: RXNAK-- RX NAK Flag on NAKEP[1:0] indexed.

This bit is set by hardware when detected a receive done on the NAK packet for OUT transaction of the NAKEP[1:0] indexed endpoint. This bit is clear when firmware write "1" to it.

Bit6: TXNAK-- TX NAK Flag on NAKEP[1:0] indexed.

This bit is set by hardware when detected a transmit done on the NAK packet for IN transaction of the NAKEP[1:0] indexed endpoint. This bit is clear when firmware write "1" to it.

Bit5: URXD5-- USB Receive Done Flag for endpoint 5.

This bit is set by hardware when detected a receive done on endpoint 5. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit4: UTXD5-- USB Transmit Done Flag for endpoint 5.

This bit is set by hardware when detected a transmit done on endpoint 5. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit3: URXD4-- USB Receive Done Flag for endpoint 4.

This bit is set by hardware when detected a receive done on endpoint 4. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit2: UTXD4-- USB Transmit Done Flag for endpoint 4.

This bit is set by hardware when detected a transmit done on endpoint 4. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit1: URXD3-- USB Receive Done Flag for endpoint 3.

This bit is set by hardware when detected a receive done on endpoint 3. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

Bit0: UTXD3-- USB Transmit Done Flag for endpoint 3.

This bit is set by hardware when detected a transmit done on endpoint 3. UC can read/write-clear on this bit. This bit is cleared when firmware writes '1' to it.

**EPINDEX: Endpoint Index Register**

SFR Address = 0xF1H

SYSRST/USBRST= XXXX-X000

7	6	5	4	3	2	1	0
--	--	--	--	--	EPINX2	EPINX1	EPINX0
W	W	W	W	W	R/W	R/W	R/W

Bit7~3: Reserved. Software must write "0" on these bits when EPINDEX is written.

Bit2~0: EPINX[2:0]-- Endpoint Index Bits [2:0]

3'b000: USB Function Endpoint 0.

3'b001: USB Function Endpoint 1.

3'b010: USB Function Endpoint 2.

3'b011: USB Function Endpoint 3.

3'b100: USB Function Endpoint 4.

3'b101: USB Function Endpoint 5.

3'b111: uC access 1K buffer enable.

**EPCON: Endpoint Control Register (endpoint-indexed)**

SFR Address = 0xE1H

SYSRST/USBRST= 0000-0000

7	6	5	4	3	2	1	0
RXSTL	TXSTL	RXDBM	TXDBM	RXISO	RXEPEN	TXISO	TXEPEN
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit7: RXSTL-- Receive Endpoint Stall.

Set this bit to stall the receive endpoint.

Note:

Clear this bit only when the host has intervened through commands sent down endpoint 0. When this bit is set and RXSETUP is clear, the receive endpoint will respond with a STALL handshake to a valid OUT token. When this bit is set and RXSETUP is set, the receive endpoint will NAK. This bit does not affect the reception of SETUP tokens by a control endpoint.

Bit6: TXSTL-- Transmit Endpoint Stall.

Set this bit to stall the transmit endpoint.

Note:

Clear this bit only when the host has intervened through commands sent down endpoint 0. When this bit is set and RXSETUP is clear, the transmit endpoint will respond with a STALL handshake to a valid IN token. When this bit is set and RXSETUP is set, the transmit endpoint will NAK.

Bit5: RXDBM-- Receive Endpoint Double Buffer Mode.

Set this bit to enable the double buffer transfer for OUT transaction. Default is cleared.

This bit is only valid for endpoint 3 and 5. When this bit is set for endpoint 3, the FIFO of endpoint 2 and the FIFO of endpoint 3 for OUT transaction would be paired to provide double buffer transfer. The OUT function of endpoint 2 is disabled. When this bit is set for endpoint 5, the FIFO of endpoint 4 and the FIFO of endpoint 5 for OUT transaction would be paired to provide double buffer transfer. The OUT function of endpoint 4 is disabled.

*Note: the Ping-Pong FIFO (double buffer mode) function is under verifying.*

Bit4: TXDBM-- Transmit Endpoint Double Buffer Mode.

Set this bit to enable the double buffer transfer for IN transaction. Default is cleared.

This bit is only valid for endpoint 2 and 4. When this bit is set for endpoint 2, the FIFO of endpoint 2 and the FIFO of endpoint 3 for IN transaction would be paired to provide double buffer transfer. The IN function of endpoint 3 is disabled. When this bit is set for endpoint 4, the FIFO of endpoint 4 and the FIFO of endpoint 5 for IN transaction would be paired to provide double buffer transfer. The IN function of endpoint 5 is disabled.

*Note: the Ping-Pong FIFO (double buffer mode) function is under verifying.*

Bit3: RXISO-- Receive Isochronous Type Enable.

Set this bit to configure the endpoint for Isochronous-Out transfer type. When disabled, the endpoint is for Bulk/Interrupt-Out transfer type. The default value is 0. This bit is only valid for endpoint 4 and 5.

Bit2: RXEPEN-- Receive Endpoint Enable.

Set this bit to enable the receive endpoint. When disabled, the endpoint does not respond to a valid OUT or SETUP token. This bit in endpoint 0 is enabled after reset. This bit has the highest priority than RXSTL.

Bit1: TXISO-- Transmit Isochronous Type Enable.

Set this bit to configure the endpoint for Isochronous-In transfer type. When disabled, the endpoint is for Bulk/Interrupt-In transfer type. The default value is 0. This bit is only valid for endpoint 4 and 5.

Bit0: TXEPEN-- Transmit Endpoint Enable.

Set this bit to enable the transmit endpoint. When disabled, the endpoint does not respond to a valid IN token. This bit in endpoint 0 is enabled after reset. This bit has the highest priority than TXSTL.

**RXSTAT: Endpoint Receive Status Register (endpoint-indexed)**

SFR Address = 0xE2H                      SYSRST/USBRST= 0000-00XX

7	6	5	4	3	2	1	0
RXSEQ	RXSETUP	STOVW	EDOVW	RXSOVW	ISOOVW	--	--
R/W	R/W	R/W	R/W	R/W	R/W	W	W

Bit7: RXSEQ-- Receive Endpoint Sequence Bit (read, conditional write).

The bit will be toggled on completion of an ACK handshake in response to an OUT token. This bit can be written by firmware if the RXOVW bit is set when written along with the new RXSEQ value.

Bit6: RXSETUP-- Received Setup Transaction.

This bit is set by hardware when a valid SETUP transaction has been received. Clear this bit upon detection of a SETUP transaction or the firmware is ready to handle the data/status stage of control transfer.

Bit5: STOVW-- Start Overwrite Flag (read-only).

Set by hardware upon receipt of a SETUP token for the control endpoint to indicate that the receive FIFO is being overwritten with new SETUP data. This bit is used only for control endpoints.

Bit4: ED OVW-- End Overwrite Flag.

This flag is set by hardware during the handshake phase of a SETUP transaction. This bit is cleared by firmware to read the FIFO data. This bit is only used for control endpoints.

Bit3: RXSOVW-- Receive Data Sequence Overwrite Bit.

Write '1' to this bit to allow the value of the RXSEQ bit to be overwritten. Writing a '0' to this bit has no effect on RXSEQ. This bit always returns '0' when read.

Bit2: ISOOVW-- Isochronous receive data Overwrite Bit.

This bit is set by hardware as a FIFO access conflict happen when uc read the last data and usb host send the next data in the same time. Firmware can use this bit to make sure whether the data had been overwritten or not. When this bit is set, Firmware should write '0' to clear this bit. This bit is only valid for endpoint 4 and endpoint 5 Isochronous OUT function.

Bit1~0: Reserved. Software must write "0" on these bits when RXSTAT is written.

**RXDAT: Receive FIFO Data Register (endpoint-indexed)**

SFR Address = 0xE3H SYSRST/USBRST= XXXX-XXXX

7	6	5	4	3	2	1	0
RXD7	RXD6	RXD5	RXD4	RXD3	RXD2	RXD1	RXD0
R	R	R	R	R	R	R	R

Bit7~0: RXD[7:0]-- Receive FIFO Data.

Receive FIFO data specified by EPINDEX is stored and read from this register.

**RXCON: Receive FIFO Control Register (endpoint-indexed)**

SFR Address = 0xE4H SYSRST/USBRST= 0XX0-XXXX

7	6	5	4	3	2	1	0
RXCLR	--	--	RXFFRC	--	--	--	--
W	W	W	W	W	W	W	W

Bit7: RXCLR-- Receive FIFO Clear.

Set this bit to flush the entire receive FIFO. All FIFO statuses are reverted to their reset states. Hardware clears this bit when the flush operation is completed.

Bit6~5: Reserved. Software must write "0" on these bits when RXCON is written.

Bit4: RXFFRC-- Receive FIFO Read Complete.

Set this bit to release the receive FIFO when data set read is complete. Hardware clears this bit after the FIFO release operation has been finished.

Bit3~0: Reserved. Software must write "0" on these bits when RXCON is written.

**RXCNT: Receive FIFO Byte Count Register (endpoint-indexed)**

SFR Address = 0xE6H SYSRST/USBRST= 0000-0000

7	6	5	4	3	2	1	0
RXBC7	RXBC6	RXBC5	RXBC4	RXBC3	RXBC2	RXBC1	RXBC0
R	R	R	R	R	R	R	R

Bit7~0: RXBC[7:0]-- Receive Byte Count.

For endpoint 0~5, This register is used to store the byte count for the data packet received in the receive FIFO specified by EPINDEX. If EPINDEX =3'b111, this register is designed to be a low byte address for read/write 1K SRAM function.

**TXSTAT: Endpoint Transmit Status Register (endpoint-indexed)**

SFR Address = 0xF2H SYSRST/USBRST= 0XXX-0XXX

7	6	5	4	3	2	1	0
TXSEQ	--	--	--	TXSOVW	--	--	--
R/W	W	W	W	R/W	W	W	W

Bit7: TXSEQ-- Transmit Endpoint Sequence Bit (read, conditional write).

The bit will be transmitted in the next PID and toggled on a valid ACK handshake of an IN transaction. This bit can be written by firmware if the TXOVW bit is set when written along with the new TXSEQ value.

Bit6~4: Reserved. Software must write "0" on these bits when TXSTAT is written.

Bit3: TXSOVW-- Transmit Data Sequence Overwrite Bit.

Write '1' to this bit to allow the value of the TXSEQ bit to be overwritten. Writing a '0' to this bit has no effect on TXSEQ. This bit always returns '0' when read.

Bit2~0: Reserved. Software must write "0" on these bits when TXSTAT is written.

**TXDAT: Transmit FIFO Data Register** (endpoint-indexed)

SFR Address = 0xF3H

SYSRST/USBRST= XXXX-XXXX

7	6	5	4	3	2	1	0
TXD7	TXD6	TXD5	TXD4	TXD3	TXD2	TXD1	TXD0
W	W	W	W	W	W	W	W

Bit7~0: TXD[7:0]-- Transmit FIFO Data.

Data to be transmitted in the FIFO specified by EPINDEX is written to this register.

**TXCON: Transmit FIFO Control Register** (endpoint-indexed)

SFR Address = 0xF4H

SYSRST/USBRST= 0XXX-0XXX

7	6	5	4	3	2	1	0
TXCLR	--	--	TXFFRC	--	--	--	--
W	W	W	W	W	W	W	W

Bit7: TXCLR-- Transmit FIFO Clear.

Set this bit to flush the entire transmit FIFO. All FIFO statuses are reverted to their reset states. Hardware clears this bit when the flush operation is completed.

Bit6~5: Reserved. Software must write "0" on these bits when TXCON is written.

Bit4: TXFFRC-- Transmit FIFO Write Complete.

Set this bit to release the transmit FIFO when data set write is complete. Hardware clears this bit after the FIFO release operation has been finished. Firmware should write this bit only after firmware finished writing TXCNT register.

Bit2~0: Reserved. Software must write "0" on these bits when TXCON is written.

**TXCNT: Transmit FIFO Byte Count Register** (endpoint-indexed)

SFR Address = 0xF6H

SYSRST/USBRST= XXXX-XXXX

7	6	5	4	3	2	1	0
TXBC7	TXBC6	TXBC5	TXBC4	TXBC3	TXBC2	TXBC1	TXBC0
W	W	W	W	W	W	W	W

Bit7~0: TXBC[7:0]-- Transmit Byte Count.

For endpoint 0~5, this register is used to stored the byte count for the data packet in the transmit FIFO specified by EPINDEX. If EPINDEX = 3'b111, this register is designed to be a high byte address for read/write 1K SRAM function.

**SIOCTL: Serial I/O Control Register High**

SFR Address = 0x2FH

SYSRST = XXXX-XXXX

7	6	5	4	3	2	1	0
DPI	DMI	--	--	--	--	--	--
R	R	R	R	R	R	R	R

Bit7: DPI-- USB DP port state, read only  
Read the port status on USB DP.

Bit6: DMI-- USB DM port state, read only  
Read the port status on USB DM.

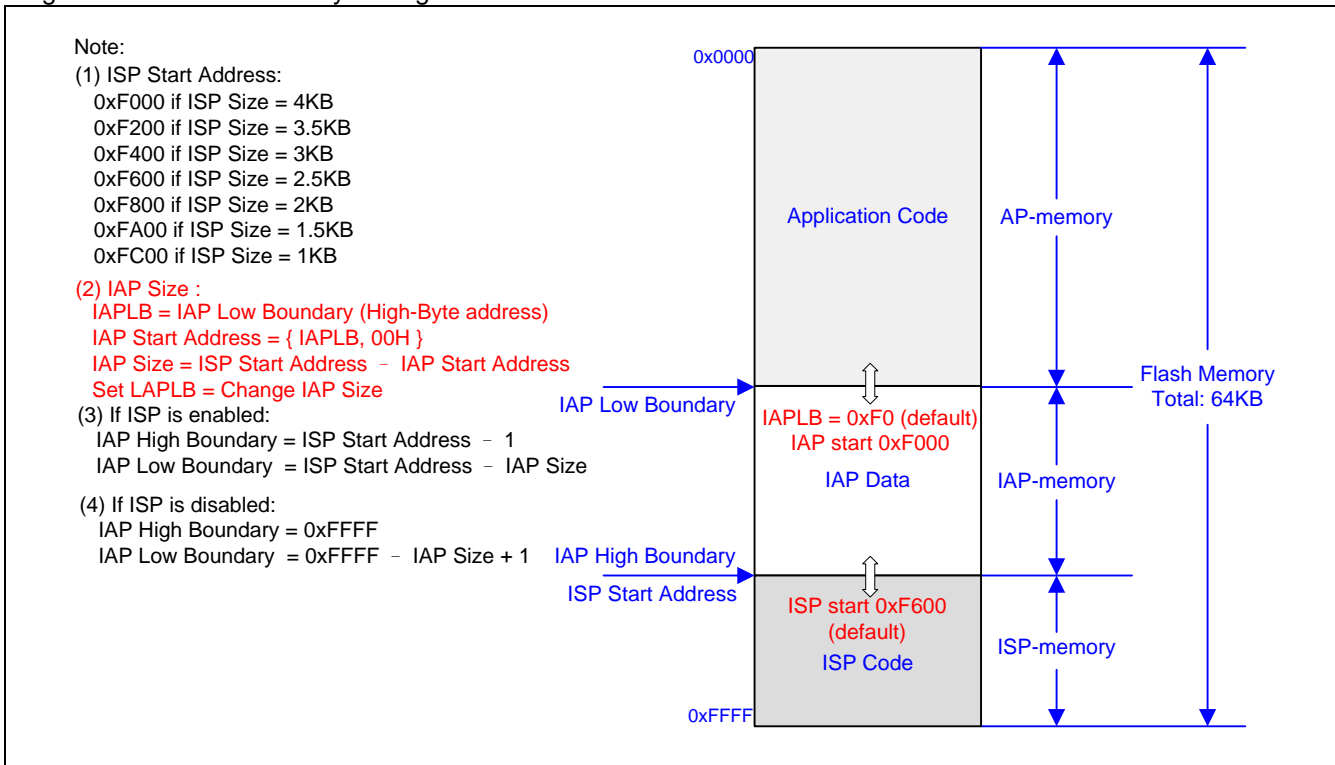
Bit5~0: Reserved. Read only.

## 24. ISP and IAP

### 24.1. Flash Memory Configuration

There are total 64K bytes of Flash Memory in **MG84FG516** and [Figure 24–1](#) shows the device flash configuration. The flash can be 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, MCU could only modify the IAP memory for storage data updated.

Figure 24–1. Flash Memory Configuration



Note:

*In default, the samples that Megawin shipped had configured the flash memory for 2.5K ISP, 1.5K IAP and Lock enabled. The 2.5K ISP region is inserted Megawin proprietary ISP code to perform In-System-Programming through USB DFU operation. For more detail information for USB DFU, please refer [MG84FG516 Development Kit on Megawin web site](#).*

**MG84FG516** does not make use of idle-mode to perform ISP and IAP function. Instead, it freezes CPU running to release the flash memory for ISP or 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. During ISPIAP operation, interrupt service is also blocked until ISP/IAP finished.



## 24.2. ISP Operation

The flash memory of **MG84FG516** can be both programming by the universal Writer/Programmer or the way of In-System Programming (ISP). ISP 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.

Note:

- (1) Before using the ISP feature, the user should configure an ISP-memory space and pre-program the ISP code 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.

### 24.2.1. ISP/IAP 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 Flash Mode Table**

SFR Page = 0~F

SFR Address = 0xE5

RESET = xxxx-x000

7	6	5	4	3	2	1	0
--	--	--	--	--	MS.2	MS.1	MS.0
W	W	W	W	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/Page-P operating mode selection

MS[2:0]	Mode
0 0 0	Standby
0 0 1	Flash byte read of AP/IAP-memory
0 1 0	Flash byte program of AP/IAP-memory
0 1 1	Flash page erase of AP/IAP-memory
<b>1 0 0</b>	<b>Page P SFR Write</b>
<b>Others</b>	<b>Reserved</b>

IFMT is used to select the flash mode for performing numerous ISP/IAP function or to select page P SFR access.

#### **IAPLB: IAP Low Boundary**

SFR Page = P

SFR Address = 0x03

RESET = 1111-111x

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 512 bytes, the IAPLB must be an even number.

To read IAPLB, MCU need to define the IMFT 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 IMFT, 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.

#### **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

RESET = 0000-0xxx

7	6	5	4	3	2	1	0
ISPEN	SWBS	SWRST	CFAIL	MISPF	--	--	--
R/W	R/W	R/W	R/W	R/W	W	W	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: MISPF, Megawin proprietary ISP Flag.

If user wants to execute the Megawin proprietary ISP function (USB DFU), user should not only set SWBS to select booting from ISP-memory, but also set this bit to trigger Megawin proprietary ISP operation before triggering software reset. If user just only sets a soft reset or perform IAP flow, must write "0" on this bit.

Bit 2~0: Reserved. Software must write "0" on these bits when ISPCR is written.

## 24.2.2. Description for ISP Operation

CPU in **MG84FG516** can be vectored into ISP-memory space from two possible ways: When HWBS (in hardware option bits) enabled, **MG84FG516** will always boot from the ISP-memory on the specified "ISP start address" since power-on. This way is named as hardware approach. Another way is software approach that allows **MG84FG516** execution switched into ISP-memory from AP memory by software setting ISPCR.7 ~ ISPCR.5 to "111" simultaneously.

Once CPU in ISP memory region and ISPEN = 1, accurate values in ISP-related registers should be confirmed by ISP software. Then, sequentially writing 0x46h, 0xB9h into SCMD are used to really trigger memory access operations (flash page erase, flash byte programming and flash byte read).

After ISP operation has been finished, software writes "001" on ISPCR.7 ~ ISPCR.5 which triggers a software RESET and makes CPU reboot into application program memory (AP-memory) on the address 0x0000.

Before perform ISP operation, the user should fill the bits XCKS4~XCKS0 in CKCON1 register with a proper value. (Refer to Section "9.2 Clock Register")

### To do Page Erase (512 Bytes per Page)

Step 1: Set MS[2:0]=[0,1,1] in ISPCR register to select Page Erase Mode.

Step 2: Fill page address in IFADRH & IFADRL registers.

Step 3: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.

### To do Byte Program

Step 1: Set MS[2:0]=[0,1,0] in ISPCR register to select Byte Program Mode.

Step 2: Fill byte address in IFADRH & IFADRL registers.

Step 3: Fill data to be programmed in IFD register.

Step 4: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.

### To do Read

Step 1: Set MS[2:0]=[0,0,1] in ISPCR register to select Read Mode.

Step 2: Fill byte address in IFADRH & IFADRL registers.

Step 3: Sequentially write 0x46h then 0xB9h to SCMD register to trigger an ISP processing.

Step 4: Now, the Flash data is in IFD register.

## 24.3. Sample Code for ISP

The following Figure 24–2 shows a sample code for ISP operation.

Figure 24–2. Sample Code for ISP

```
*****
; 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 (512 bytes per page)
;=====
;          ORL    IFMT,#03h      ;MS[2:0]=[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. Byte Program Mode
;=====
;          ORL    IFMT,#02h      ;MS[2:0]=[0,1,0], select Byte Program Mode
;          ANL    ISPCR,#0FAh    ;
;          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)
;
;=====
; 3. Verify using Read Mode
;=====
;          ANL    IFMT,#0F9h     ;MS1[2:0]=[0,0,1], select Byte Read Mode
;          ORL    IFMT,#01h      ;
;          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
;          ...
ISP_error:
;          ...
;
```

## 24.4. IAP Operation

The device is *In Application Programmable* (IAP), which allows some region in 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. Thus, there is no need to use an external serial EEPROM (such as 93C46, 24C01, ..., and so on) for saving the non-volatile data.

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) Before using the IAP feature, the software should specify an IAP-memory space by writing IAPLB in Page-P SFR. 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.

### 24.4.1. IAP-memory Boundary/Range

If ISP-memory is specified, the range of the IAP-memory is determined by IAP and the ISP starts address as listed below.

$$\begin{aligned} \text{IAP high boundary} &= \text{ISP start address} - 1. \\ \text{IAP low boundary} &= \text{ISP start address} - \text{IAP size}.. \end{aligned}$$

If ISP-memory is not specified, the range of the IAP-memory is determined by the following formula.

$$\begin{aligned} \text{IAP high boundary} &= 0xFFFF. \\ \text{IAP low boundary} &= 0xFFFF - \text{IAP size} + 1. \end{aligned}$$

For example, if ISP-memory is 2.5K, so that ISP start address is 0xF600, and IAP-memory is 1.5K, then the IAP-memory range is located at 0xF000 ~ 0xF5FF. The IAP low boundary in **MG84FG516** is defined by IAPLB register which can be modified by software to adjust the IAP size in user's AP program.

### 24.4.2. Update data in IAP-memory

The special function registers are related to ISP/IAP would be shown in Section [24.2.1 ISP/IAP 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 512 bytes) into XRAM buffer which contains the data to be updated.
- Step 2: Erase this page (**using Page Erase mode of ISP**).
- 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 Byte Program mode of ISP**).

To read the data in the IAP-memory, users can use either the "**MOVC A,@A+DPTR**" instruction or the **Read mode of ISP**.

## 25. Page P SFR Access

**MG84FG516** 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. The SFRs in Page-P don’t support read function.

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
W	W	W	W	W	W	W	W

Bit 7~0: The IAPLB determines the IAP-memory lower boundary. Since a Flash page has 512 bytes, the IAPLB must be an even number.

To read IAPLB, MCU need to define the IFADRL for SFR address in Page-P, the IMFT 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 IMFT, 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

SFR Address = 0x40 RESET = 0101-0000

7	6	5	4	3	2	1	0
XTGS1	XTGS0	XTALE	IHRCOE	MCKS1	MCKS0	OSCS1	OSCS0
W	W	W	W	W	W	W	W

Bit 7~6: XTGS1~XTGS0, XTAL oscillator Gain control Register. Software must writ “01” on the two bits.

Bit 5: XTALE, external Crystal(XTAL) Enable.

0: Disable XTAL oscillating circuit. In this case, XTAL2 and XTAL1 behave as Port 6.0 and Port 6.1.

1: Enable XTAL oscillating circuit. If this bit is set by CPU software, it needs **3 ms** to have stable output after XTALE enabled.

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: MCKS[1:0], MCK Source Selection.

MCKS[1:0]	MCK Source Selection
0 0	OSCin
0 1	24MHz (ENCKM must be enabled)
1 0	32MHz (ENCKM must be enabled)
1 1	Reserved.

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

OSCS[1:0]		OSCin source Selection
0	0	IHRCO
0	1	XTAL
1	0	ILRCO
1	1	ECKI, External Clock Input (P6.0) as OSCin.

**PCON2: Power Control Register 2**

SFR Page = **P Only**

SFR Address = 0x44

POR = 0000-0101

7	6	5	4	3	2	1	0
HSE	IAP0	BO1S1	BO1S0	BO1RE	EBOD1	BO0RE	<b>1</b>
W	W	W	W	W	W	W	W

Bit 7: HSE, High Speed operation Enable.

0: Disable high speed operation for MCU.

1: Enable high speed operation for MCU (SYSCLK > 24MHz). Before select high frequency clock (>24MHz) on SYSCLK, software must set HSE to switch internal circuit for high speed operation. It may cause more power consumption on chip operation.

Bit 6: IAP0, 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 5~4: BO1S[1:0]. Brown-Out detector 1 monitored level Selection. The initial values of these two bits are loaded from OR1.BO1S10 and OR1.BO1S00.

BO1S[1:0]		BOD1 detecting level
0	0	2.0V
0	1	2.4V
1	0	3.7V
1	1	4.2V

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 2.2V).

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 = XXXX-0XX1

7	6	5	4	3	2	1	0
<b>0</b>	<b>0</b>	<b>0</b>	<b>0</b>	AWBOD1	<b>0</b>	<b>0</b>	OCDE
W	W	W	W	W	W	W	W

Bit 7~4: Reserved. Software must write "0" on these bits when PCON3 is written.

Bit 3: AWBOD1, Awaked BOD1 in PD mode.

0: BOD1 is disabled in power-down mode.

1: BOD1 keeps operation in power-down mode.

Bit 2~1: Reserved. Software must write “0” on these bits when PCON3 is written.

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.

**SPCON0: SFR Page Control 0**

SFR Page = P Only

SFR Address = 0x48

POR = X000-0000

7	6	5	4	3	2	1	0
--	P6CTL	P4CTL	WRCTL	CKCTL1	CKCTL0	PWCTL1	PWCTL0
W	W	W	W	W	W	W	W

Bit 7: Reserved. Software must write “0” on this bit when SPCON0 is written.

Bit 6: P6CTL. P6 SFR access Control.

If P6CTL is set, it will disable the P6 SFR modified in Page 1. P6 in Page 1 only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

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. WDTCSR SFR access Control.

If WRCTL is set, it will disable the WDTCSR SFR modified in Page 0~F. WDTCSR in Page 0~F only keeps the SFR read function. But software always owns the modification capability in SFR Page P.

Bit 3: CKCTL1. CKCON1 SFR access Control.

If CKCTL1 is set, it will disable the CKCON1 SFR modified in Page 0~F. CKCON1 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.



## 25.1. Page-P Sample Code

### (1). Required Function: General function call of Page-P SFR Read

#### Assembly Code Example:

```
_page_p_sfr_read:
page_p_sfr_read:
    MOV    IFADRH,000h
    MOV    IFMT,#(MS2|MS0)          ; PageP_Read=0x05

    ANL    ISPCR,#CFAIL
    ORL    ISPCR,#ISPEN            ; Enable Function

    MOV    SCMD,#046h
    MOV    SCMD,#0B9h

    MOV    IFMT,#000h
    ANL    ISPCR,#~ISPEN           ; Flash_Standby=0x00
                                           ; Disable Function

    RET
```

#### C Code Example:

```
void page_p_sfr_read (void)
{
    IFADRH = 0x00;                //
    ISPCR = ISPEN;                // Enable Function
    IFMT = (MS0 | MS2);          // PageP_Read=0x05

    SCMD = 0x46;                 //
    SCMD = 0xB9;                 //

    IFMT = Flash_Standby;        // Flash_Standby=0x00
    ISPCR &= ~ISPEN;

}
```

### (2). Required Function: General function call of Page-P SFR Write

#### Assembly Code Example:

```
_page_p_sfr_write:
page_p_sfr_write:
    MOV    IFADRH,000h
    MOV    ISPCR,#ISPEN            ; Enable Function
    MOV    IFMT,#MS2              ; PageP_Write=0x04

    MOV    SCMD,#046h
    MOV    SCMD,#0B9h

    MOV    IFMT,#000h
    ANL    ISPCR,#~ISPEN           ; Flash_Standby=0x00
                                           ; Disable Function

    RET
```

#### C Code Example:

```
void page_p_sfr_write (void)
{
    IFADRH = 0x00;
    ISPCR = ISPEN;                // Enable Function
    IFMT = MS2;                  // PageP_Write=0x04

    SCMD = 0x46;                 //
    SCMD = 0xB9;                 //
}
```

```

IFMT = Flash_Standby;           // Flash_Standby=0x00
ISPCR &= ~ISPEN;
}

```

**(3). Required Function: Enable PWCTL0 for PCON0.PD control in Page-P**

Assembly Code Example:

```

MOV    IFADRL,#SPCON0           ;
CALL   page_p_sfr_read          ;

ORL    IFD,#PWCTL0              ; Set PWCTL0
CALL   page_p_sfr_write         ;

MOV    IFD,PCON0                ; Read PCON0

ORL    IFD,#PD                  ; Write PCON0 and Power-Down
MOV    IFADRL,#PCON0_P          ;
CALL   page_p_sfr_write         ;

```

C Code Example:

```

IFADRL = SPCON0;                //
page_p_sfr_read();              //

IFD |= PWCTL0;                  // Set PWCTL0
page_p_sfr_write();             //

IFD = PCON0;                    // Read PCON0

IFD |= PD;                      // Write PCON0
IFADRL = PCON0_P;               //
page_p_sfr_write();             //

```

**(4). Required Function: Enable CKCTL0 for SYSCLK divider (CKCON0) changed in Page-P**

Assembly Code Example:

```

MOV    IFADRL,#SPCON0           ;
CALL   page_p_sfr_read          ;

ORL    IFD,#CKCTL0              ; Set CKCTL0
CALL   page_p_sfr_write         ;

MOV    IFD,CKCON0               ; Read CKCON0

ORL    IFD,#(AFS | SCKS0)       ; Write CKCON0 and Set AFS
MOV    IFADRL,#CKCON0_P         ; SYSCLK / 2
CALL   page_p_sfr_write         ;

```

C Code Example:

```

IFADRL = SPCON0;                //
page_p_sfr_read ();             //

IFD |= CKCTL0;                  // Set CKCTL
page_p_sfr_write();             //

IFD = CKCON0;                   // read CKCON0

IFD |= (AFS | SCKS0);           //
IFADRL = CKCON0_P;              //
page_p_sfr_write();             // Write CKCON0

```

## 26. Auxiliary SFRs

### Auxiliary Register 0

SFR Page = 0 ~ F

SFR Address = 0xA1

RESET = 000X-0000

7	6	5	4	3	2	1	0
P60OC1	P60OC0	P60FD	--	P4FS1	P4FS0	INT1H	INT0H
R/W	R/W	R/W	W	R/W	R/W	R/W	R/W

Bit 7~6: P6.0 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 crystal mode, XTAL2 and XTAL1 are the alternated function of P6.0 and P6.1. In external clock input mode, P6.0 is the dedicated clock input pin. In internal oscillator condition, P6.0 provides the following selections for GPIO or clock source generator. When P60OC[1:0] index to non-P6.0 GPIO function, P6.0 will drive the on-chip RC oscillator output to provide the clock source for other devices.

P60OC[1:0]	P60 function	I/O mode
00	P60	By P6M0.0
01	MCK	By P6M0.0
10	MCK/2	By P6M0.0
11	MCK/4	By P6M0.0

Please refer Section “9 System Clock” to get the more detailed clock information. For clock-out on P6.0 function, it is recommended to set P6M0.0 to “1” which selects P6.0 as push-push output mode.

Bit 5: P60FD, P6.0 Fast Driving.

0: P6.0 output with default driving.

1: P6.0 output with fast driving enabled. If P6.0 is configured to clock output, enable this bit when P6.0 output frequency is more than 12MHz at 5V application or more than 6MHz at 3V application.

Bit 3~2: P4.4 and P4.5 alternated function selection.

P4FS[1:0]	P4.4	P4.5
00	P4.4	P4.5
01	Input for RXD0	Output for TXD0
10	Input for nINT2	Input for nINT3
11	Input for T3EX	Input for T3 or Output for T3CKO

Bit 1: INT1H, INT1 High/Rising trigger enable.

0: Remain INT1 triggered on low level or falling edge on P3.3.

1: Set INT1 triggered on high level or rising edge on P3.3.

Bit 0: INT0H, INT0 High/Rising trigger enable.

0: Remain INT0 triggered on low level or falling edge on P3.2.

1: Set INT0 triggered on high level or rising edge on P3.2.

### AUXR1: Auxiliary Control Register 1

SFR Page = 0 ~ F

SFR Address = 0xA2

RESET = 0000-0000

7	6	5	4	3	2	1	0
KBIPS1	KBIPS0	P5SPI	P5S1	P5T2	P6PCA	EXTRAM	DPS
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: KBIPS1~0, KBI Port Selection [1:0].

KBIPS1~0	KBI7~0
00	P0.7~P0.0
01	P2.7~P2.0
10	P5.7~P5.0
11	P6.7~P6.0

Bit 5: P5SPI, SPI interface on P5.7~P5.4.  
 0: Disable SPI function moved to P5.  
 1: Set SPI function on P5 as following definition.  
 'nSS' function in P1.4 is moved to P5.4.  
 'MOSI' function in P1.5 is moved to P5.5.  
 'MISO' function in P1.6 is moved to P5.6.  
 'SPICLK' function in P1.7 is moved to P5.7.

Bit 4: P5S1, Serial Port 1 (UART1) on P5.2/P5.3.  
 0: Disable UART1 function moved to P5.  
 1: Set UART1 RXD1/TXD1 on P5.2/P5.3 following definition.  
 'RXD1' function in P1.2 is moved to P5.2.  
 'TXD1' function in P1.3 is moved to P5.3.

Bit 3: P5T2, T2(T2CKO)/T2EX function on P5.0/P5.1.  
 0: Disable T2 function moved to P5.  
 1: Set UART1 T2(T2CKO)/T2EX on P5.0/P5.1 following definition.  
 'T2(T2CKO)' function in P1.0 is moved to P5.0.  
 'T2EX' function in P1.1 is moved to P5.1.

Bit 2: P6PCA, PCA function on P6.  
 0: Disable PCA function moved to P6.  
 1: Set PCA function on P6 as following definition.  
 'ECI' function in P2.1 is moved to P6.1.  
 'CEX0' function in P2.2 is moved to P6.2.  
 'CEX1' function in P2.3 is moved to P6.3.  
 'CEX2' function in P2.4 is moved to P6.4.  
 'CEX3' function in P2.5 is moved to P6.5.  
 'CEX4' function in P2.6 is moved to P6.6.  
 'CEX5' function in P2.7 is moved to P6.7.

Bit 1: EXTRAM, External data RAM enable.  
 0: Enable on-chip expanded data RAM (XRAM 4096 bytes)  
 1: Disable on-chip expanded data RAM.

Bit 0: DPS, dual DPTR Selector.  
 0: Select DPTR0.  
 1: Select DPTR1.

**AUXR2: Auxiliary Register 2**

SFR Page = 0 ~ F

SFR Address = 0xA3

RESET = 0000-0000

7	6	5	4	3	2	1	0
INT3IS1	INT3IS0	INT2IS1	INT2IS0	T1X12	T0X12	T1CKOE	T0CKOE
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W

Bit 7~6: INT3IS1~0, nINT3 input selection bits which function is defined as following table.

INT3IS1~0	nINT3 Input	Selected Port Pin	Switch Condition
00	nINT3 Port Pin	P4.2 or P4.5	AUXR0.P4FS1~0
01	RXD1 Port Pin	P1.2 or P5.2	AUXR1.P5S1
10	TWSI SDA Port Pin	P4.1	None
11	SPI nSS Port Pin	P1.4 or P5.4	AUXR1.P5SPI

Bit 5~4: INT2IS1~0, nINT2 input selection bits which function is defined as following table.

INT2IS1~0	nINT2	Selected Port Pin	Switch Condition
00	nINT2 Port Pin	P4.3 or P4.4	AUXR0.P4FS1~0
01	RXD0 Port Pin	P3.0 or P4.4	AUXR0.P4FS1~0
10	TWSI SDA Port Pin	P4.1	None
11	SPI nSS Port Pin	P1.4 or P5.4	AUXR1.P5SPI

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 P3.5.

Bit 0: T0CKOE, Timer 0 Clock Output Enable.  
 0: Disable Timer 0 clock output.  
 1: Enable Timer 0 clock output on P3.4.

**SFRPI: SFR Page Index Register**

SFR Page = 0-F & P

SFR Address = 0xAC

RESET = xxxx-0000

7	6	5	4	3	2	1	0
--	--	--	--	PIDX3	PIDX2	PIDX1	PIDX0
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. The available pages are only page “0” and “1”.

There are 13 register sets in Page 0, S0CON(98H), S0BUF(99H), S0CFG(9AH), S1CFG(9BH), PUCON0(B4H), P5M0(B5H), T2CON(C8H), T2MOD(C9H), RCAP2L(CAH), RCAP2H(CBH), TL2(CCH), TH2(CDH) and P5(F8H).  
 13 register sets in Page 1, S1CON(98H), S1BUF(99H) and S1BRT(9AH), S1BRC(9BH), PUCON1(B4H), P6M0(B5H), T3CON(C8H), T3MOD(C9H), RCAP3L(CAH), RCAP3H(CBH), TL3(CCH), TH3(CDH) and P6(F8H).

PIDX[3:0]	Selected Page
0000	Page 0
0001	Page 1
0010	Page 2
0011	Page 3
.....	.....
.....	.....
.....	.....
1111	Page F

## 27. 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 "28.5 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 **MG84FG516** 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., 0xFFFF. The following table lists the ISP space option in this chip. In default setting, MG84FG516 ISP space is configured to 2.5K that had been embedded Megawin USB DFU boot loader to perform on-USB-line Device Firmware Upgrade.

ISP-memory Size	ISP Start Address
4K bytes	0xF000
3.5K bytes	0xF200
3K bytes	0xF400
2.5K bytes	0xF600
2K bytes	0xF800
1.5K bytes	0xFA00
1K bytes	0xFC00
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 1.5K bytes.

### BO1S10, BO1S00:

- , : Select BOD1 to detect 2.0V.
- , : Select BOD1 to detect 2.4V.
- , : Select BOD1 to detect 3.7V.
- , : Select BOD1 to detect 4.2V.

### BO0REO:

- : Enabled. BOD0 will trigger a RESET event to CPU on AP program start address. (2.2V)
- : 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, 3.7V, 2.4V or 2.0V)
- : Disabled. BOD1 can not trigger a RESET to CPU.

### 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.

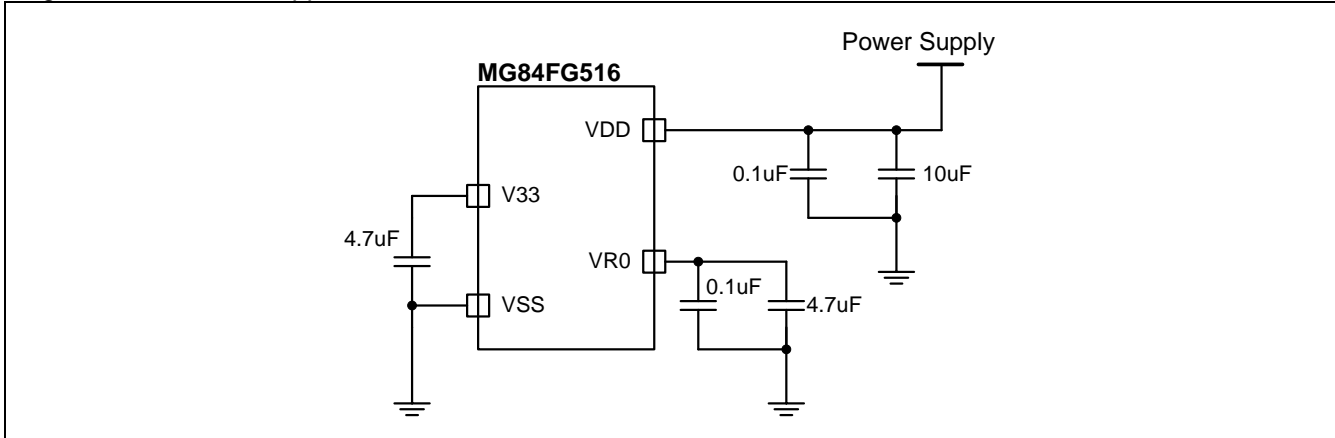
: Disabled. The WDT SFRs, WREN, NSW, WIDL, PS2, PS1 and PS0 in WDTCR, are free for writing of software.

## 28. Application Notes

### 28.1. Power Supply Circuit

To have the **MG84FG516** work with power supply varying from 2.0V to 5.5V, adding some external decoupling and bypass capacitors is necessary, as shown in [Figure 28–1](#).

Figure 28–1. Power Supplied Circuit



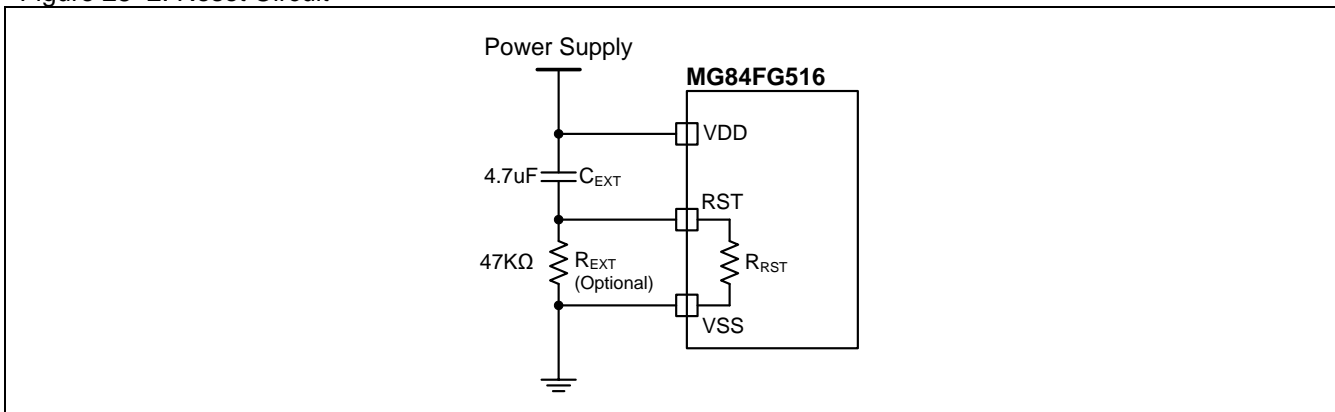
### 28.2. Reset Circuit

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 28–2](#) 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 “[29.2 DC Characteristics](#)” for  $R_{RST}$  value.

Figure 28–2. Reset Circuit





### 28.3. XTAL Oscillating Circuit

To achieve successful and exact oscillating (up to 24MHz), the capacitors C1 and C2 are necessary, as shown in Figure 28–3. Normally, C1 and C2 have the same value. Table 28–1 lists the C1 & C2 value for the different frequency crystal application.

Figure 28–3. XTAL Oscillating Circuit

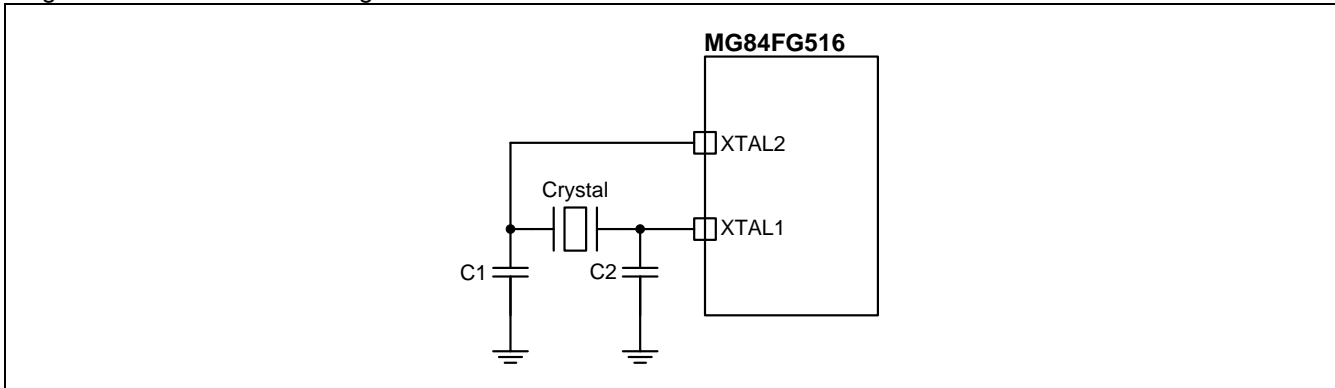


Table 28–1. Reference Capacitance of C1 & C2 for crystal oscillating circuit

Crystal	C1, C2 Capacitance
16MHz ~ 25MHz	10pF
6MHz ~ 16MHz	15pF
2MHz ~ 6MHz	33pF

## 28.4. ICP and OCD Interface Circuit

**MG84FG516** 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.

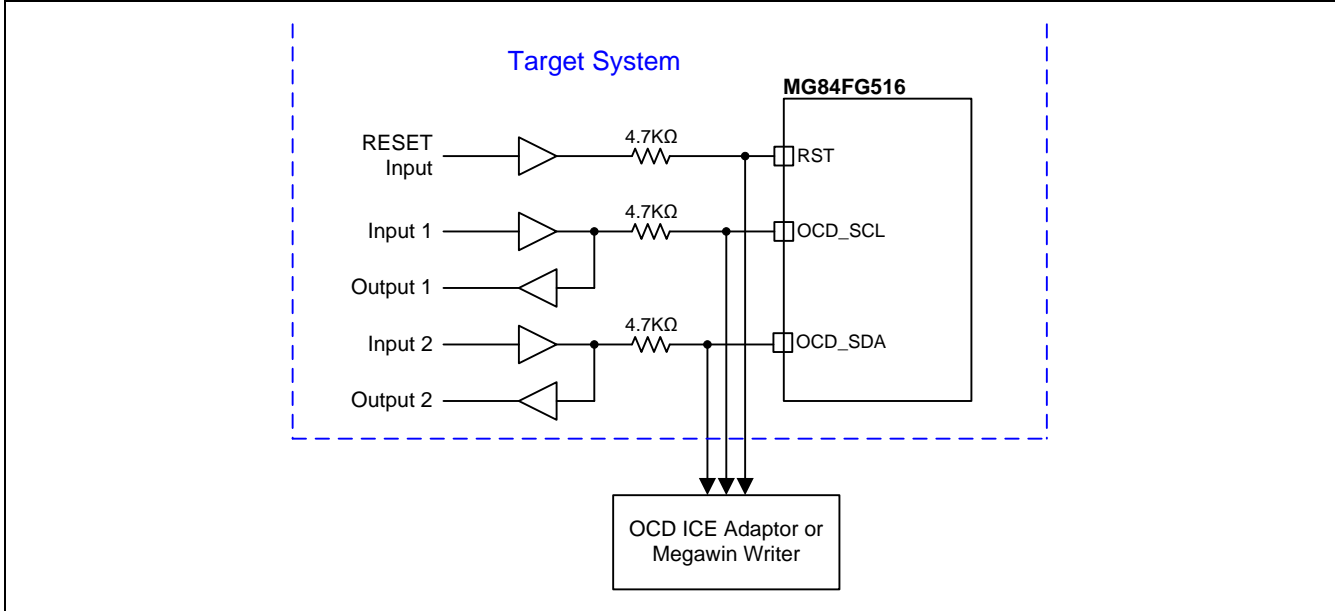
The ICP interface allows the ICP\_SCL/ICP\_SDA pins to be shared with user functions so that In-Chip Flash Programming function could be performed. This is practicable because ICP communication is performed when the device is in the halt state, where the on-chip peripherals and user software are stalled. 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 28–4.

***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 MG84FG516 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 28–4, 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 PCON3. 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 OCED as "1". Or "Erase" the on-chip flash by ICP which cleans the user software to stop the port pins switching.

Figure 28–4. ICP and OCD Interface Circuit



## 28.5. 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.

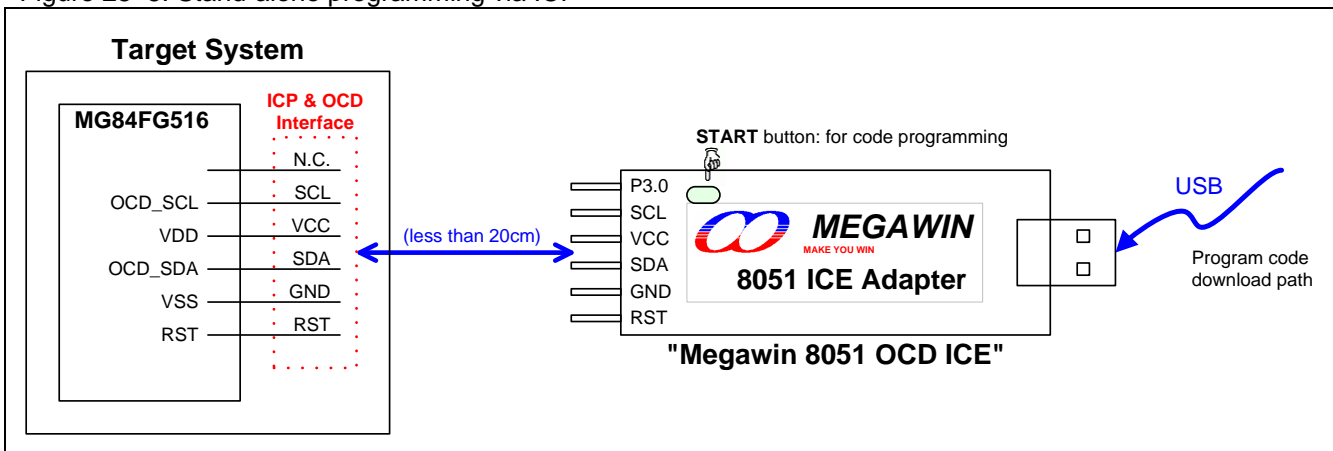
The proprietary 6-pin “Megawin 8051 ICE Adapter” can support the In-Circuit Programming of MG84FG516. “Megawin 8051 ICE Adapter” has the in-system storage to store the user program code and device options. So, the tools can perform a portable and stand-alone programming without a host on-line, such as 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 28–5. 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 8051 ICE Adapter” 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 8051 ICE Adapter” 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 8051 ICE Adapter”.

Figure 28–5. Stand-alone programming via ICP



## 28.6. On-Chip-Debug Function

The **MG84FG516** 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 8051 OCD ICE” for the user, as shown in [Figure 28–6](#). 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 28–6](#).

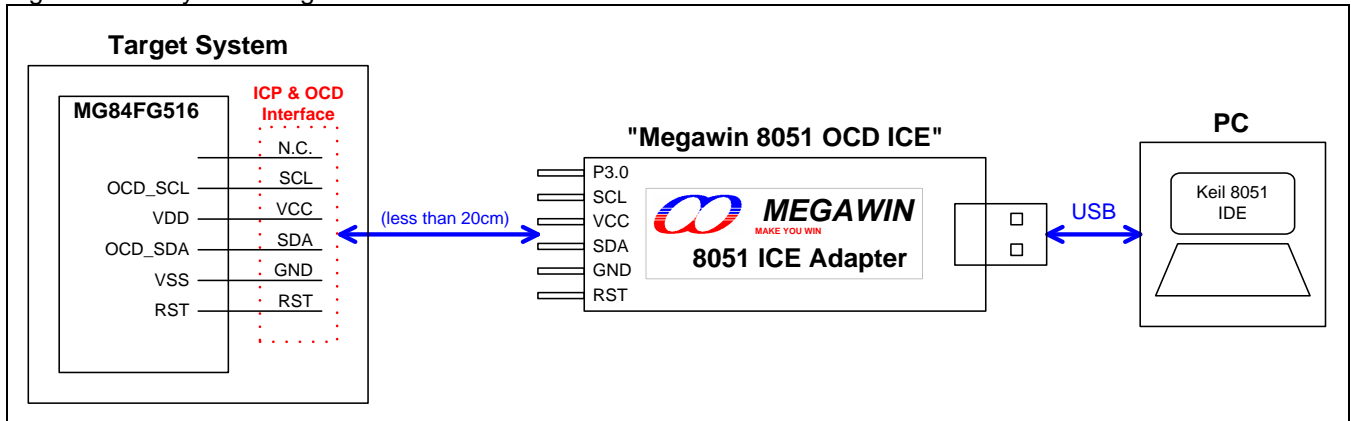
In addition, the most powerful feature is that it can directly connect the user’s target system to the Keil 8051 IDE software for debugging, which directly utilizes the Keil IDE’s dScope-Debugger function. Of course, all the advantages are based on your using Keil 8051 IDE software.

*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 28–6. System Diagram for the ICE Function



*Note: For more detailed information about the OCD ICE, please feel free to contact Megawin.*

## 29. Electrical Characteristics

### 29.1. Absolute Maximum Rating

Parameter	Rating	Unit
Ambient temperature under bias	-40 ~ +125	°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.

## 29.2. DC Characteristics

VDD = 5.0V±10%, VSS = 0V, T<sub>A</sub> = 25 °C and execute NOP for each machine cycle, unless otherwise specified

Symbol	Parameter	Test Condition	Limits			Unit
			min	typ	max	
<b>Input/Output Characteristics</b>						
V <sub>IH1</sub>	Input High voltage (All I/O Ports)	Except P6.0, P6.1	0.6			VDD
V <sub>IH2</sub>	Input High voltage (RST, P6.0, P6.1)		0.75			VDD
V <sub>IL1</sub>	Input Low voltage (All I/O Ports)	Except P6.0, P6.1			0.15	VDD
V <sub>IL2</sub>	Input Low voltage (RST, P6.0, P6.1)				0.2	VDD
I <sub>IH</sub>	Input High Leakage current (All I/O Ports)	V <sub>PIN</sub> = VDD		0	10	µA
I <sub>IL1</sub>	Logic 0 input current (P3 in quasi-mode or other Input port with on-chip pull-up resistor)	V <sub>PIN</sub> = 0.4V		20	50	µA
I <sub>IL2</sub>	Logic 0 input current (All Input only or open-drain Ports)	V <sub>PIN</sub> = 0.4V		0	10	µA
I <sub>H2L</sub>	Logic 1 to 0 input transition current (P3 in quasi-mode or other Input port with on-chip pull-up resistor)	V <sub>PIN</sub> = 1.8V		330	500	µA
I <sub>OH1</sub>	Output High current (P3 in quasi-Mode or other open-drain output port with on-chip pull-up resistor)	V <sub>PIN</sub> = 2.4V	150	200		µA
I <sub>OH2</sub>	Output High current (All push-pull output ports)	V <sub>PIN</sub> = 2.4V	12			mA
I <sub>OL1</sub>	Output Low current (All I/O Ports)	V <sub>PIN</sub> = 0.4V	12			mA
R <sub>RST</sub>	Internal reset pull-down resistance			85		Kohm
<b>Power Consumption</b>						
I <sub>OP1</sub>	Normal mode operating current	SYSCLK = 32MHz @ IHRCO with PLL		10.5		mA
I <sub>OP2</sub>		SYSCLK = 24MHz @ IHRCO with PLL		9		mA
I <sub>OP3</sub>		SYSCLK = 12MHz @ IHRCO		5.3		mA
I <sub>OP4</sub>		SYSCLK = 12MHz @ IHRCO with ADC		9.3		mA
I <sub>OP5</sub>		SYSCLK = 24MHz @ XTAL		11		mA
I <sub>OP6</sub>		SYSCLK = 12MHz @ XTAL		6.4		mA
I <sub>OP7</sub>		SYSCLK = 6MHz @ XTAL		4		mA
I <sub>OP8</sub>		SYSCLK = 2MHz @ XTAL		2.5		mA
I <sub>OP9</sub>	Normal mode operation current on USB connected	SYSCLK = 12MHz @ IHRCO with PLL, USB		12		mA
I <sub>OPS1</sub>	Slow mode operating current	SYSCLK = 12MHz/128 @ IHRCO		1		mA
I <sub>OPS2</sub>		SYSCLK = 12MHz/128 @ XTAL		2		mA
I <sub>IDLE1</sub>	Idle mode operating current	SYSCLK = 12MHz @ IHRCO		2		mA
I <sub>IDLE2</sub>		SYSCLK = 12MHz @ XTAL		3		mA

I <sub>IDLE3</sub>		SYSCCLK = 12MHz/128 @ IHRCO		0.9		mA
I <sub>IDLE4</sub>		SYSCCLK = 12MHz/128 @ XTAL		2		mA
I <sub>IDLE5</sub>		SYSCCLK = 32KHz @ ILRCO		120		uA
I <sub>SUB1</sub>	Sub-clock mode operating current	SYSCCLK = 32KHz @ ILRCO, BOD1 disabled		130		uA
I <sub>SUB2</sub>		SYSCCLK = 32KHz/128 @ ILRCO, BOD1 disabled		120		uA
I <sub>WAT</sub>	Watch mode operating current	WDT = 32KHz @ ILRCO in PD mode		15		uA
I <sub>MON1</sub>	Monitor Mode operating current	BOD1 enabled in PD mode		100		uA
I <sub>PD1</sub>	Power down mode current	Non USB application		5		uA
I <sub>PD2</sub>	USB mode power down current	USB on line		300		uA
<b>BOD0/BOD1 Characteristics</b>						
V <sub>BOD0</sub>	BOD0 detection level	T <sub>A</sub> = -40°C to +125°C	2.1 <sup>(1)</sup>	2.2	2.4 <sup>(1)</sup>	V
V <sub>BOD10</sub>	BOD1 detection level for 2.0V	T <sub>A</sub> = -40°C to +125°C	1.85 <sup>(1)</sup>	2.0	2.15 <sup>(1)</sup>	V
V <sub>BOD11</sub>	BOD1 detection level for 2.4V	T <sub>A</sub> = -40°C to +125°C	2.25 <sup>(1)</sup>	2.37	2.55 <sup>(1)</sup>	V
V <sub>BOD12</sub>	BOD1 detection level for 3.7V	T <sub>A</sub> = -40°C to +125°C	3.55 <sup>(1)</sup>	3.7	3.9 <sup>(1)</sup>	V
V <sub>BOD13</sub>	BOD1 detection level for 4.2V	T <sub>A</sub> = -40°C to +125°C	4.05 <sup>(1)</sup>	4.2	4.4 <sup>(1)</sup>	V
I <sub>BOD1</sub>	BOD1 Power Consumption	T <sub>A</sub> = +25°C, VDD=5.0V		120		uA
<b>Operating Condition</b>						
V <sub>PSR</sub>	Power-on Slop Rate	T <sub>A</sub> = -40°C to +125°C	0.05			V/ms
V <sub>OP1</sub>	XTAL Operating Speed 0-24MHz	T <sub>A</sub> = -40°C to +125°C	2.7		5.5	V
V <sub>OP2</sub>	XTAL Operating Speed 0-12MHz	T <sub>A</sub> = -40°C to +125°C	2.0		5.5	V
V <sub>OP3</sub>	CPU Operating Speed 0-36MHz	T <sub>A</sub> = -40°C to +125°C	3.0		5.5	V
V <sub>OP4</sub>	CPU Operating Speed 0-24MHz	T <sub>A</sub> = -40°C to +125°C	2.4		5.5	V
V <sub>OP5</sub>	CPU Operating Speed 0-12MHz	T <sub>A</sub> = -40°C to +125°C	2.0		5.5	V

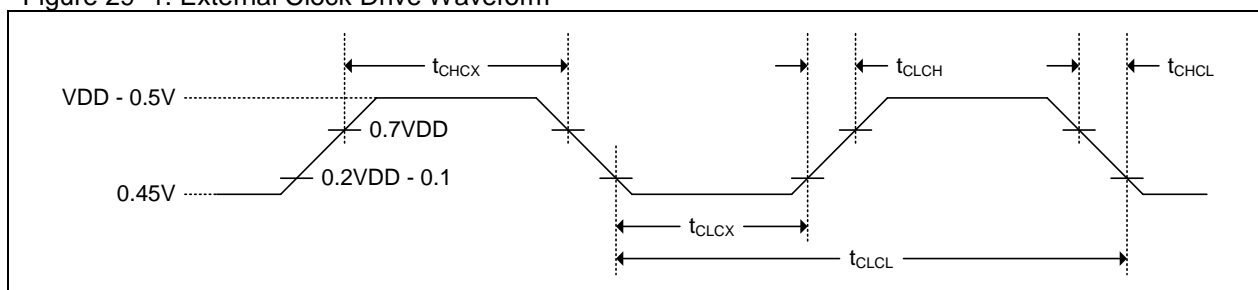
<sup>(1)</sup> Data based on characterization results, not tested in production.

### 29.3. External Clock Characteristics

VDD = 2.7V ~ 5.5V, VSS = 0V, TA = -40°C to +125°C, unless otherwise specified

Symbol	Parameter	Oscillator				Unit
		Crystal Mode		ECKI Mode		
		Min.	Max	Min.	Max	
1/t <sub>CLCL</sub>	Oscillator Frequency	2	24	0	36	MHz
1/t <sub>CLCL</sub>	Oscillator Frequency (VDD = 2.0V ~ 5.5V)	2	12	0	12	MHz
t <sub>CLCL</sub>	Clock Period	41.6		27.7		ns
t <sub>CHCX</sub>	High Time	0.4T	0.6T	0.4T	0.6T	t <sub>CLCL</sub>
t <sub>CLCX</sub>	Low Time	0.4T	0.6T	0.4T	0.6T	t <sub>CLCL</sub>
t <sub>CLCH</sub>	Rise Time		5		5	ns
t <sub>CHCL</sub>	Fall Time		5		5	ns

Figure 29–1. External Clock Drive Waveform



### 29.4. IHRCO Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage		2.0		5.5	V
IHRCO Frequency	TA = +25°C		12		MHz
IHRCO Frequency Deviation (factory calibrated)	TA = +25°C	-1.0		+1.0	%
	TA = -40°C to +85°C	-1.5 <sup>(1)</sup>		+1.5 <sup>(1)</sup>	%
	TA = -40°C to +125°C	-2.0 <sup>(1)</sup>		+2.0 <sup>(1)</sup>	%
IHRCO Start-up Time	TA = -40°C to +125°C			32 <sup>(1)</sup>	us
IHRCO Power Consumption	TA = +25°C, VDD=5.0V		500		uA

<sup>(1)</sup> Data based on characterization results, not tested in production.

### 29.5. ILRCO Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage		2.0		5.5	V
ILRCO Frequency	TA = +25°C		32		KHz
ILRCO Frequency Deviation	TA = +25°C	-20 <sup>(1)</sup>		+20 <sup>(1)</sup>	%
	TA = -40°C to +85°C	-40 <sup>(1)</sup>		+40 <sup>(1)</sup>	%
	TA = -40°C to +125°C	-50 <sup>(1)</sup>		+50 <sup>(1)</sup>	%

<sup>(1)</sup> Data based on characterization results, not tested in production.



## 29.6. CKM Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage	TA = -40°C to +125°C	2.4		5.5	V
Clock Input Range	TA = -40°C to +125°C	5 <sup>(1)</sup>	6	7 <sup>(1)</sup>	MHz
CKM Start-up Time	TA = -40°C to +125°C	20 <sup>(2)</sup>		100 <sup>(2)</sup>	us
CKM Power Consumption	TA = +25°C, VDD=5.0V		1		mA

<sup>(1)</sup> Data guaranteed by design, not tested in production.

<sup>(2)</sup> Data based on characterization results, not tested in production.

## 29.7. Flash Characteristics

Parameter	Test Condition	Limits			Unit
		min	typ	max	
Supply Voltage	TA = -40°C to +125°C	2.0		5.5	V
Flash Write (Erase/Program) Voltage	TA = -40°C to +125°C	2.2		5.5	V
Flash Erase/Program Cycle	TA = -40°C to +125°C	10,000			times
Flash Data Retention	TA = +25°C	100			year

## 29.8. ADC Characteristics

VDD=5.0V, VREF+=3.0, TA= -40°C ~ +85°C unless otherwise specified

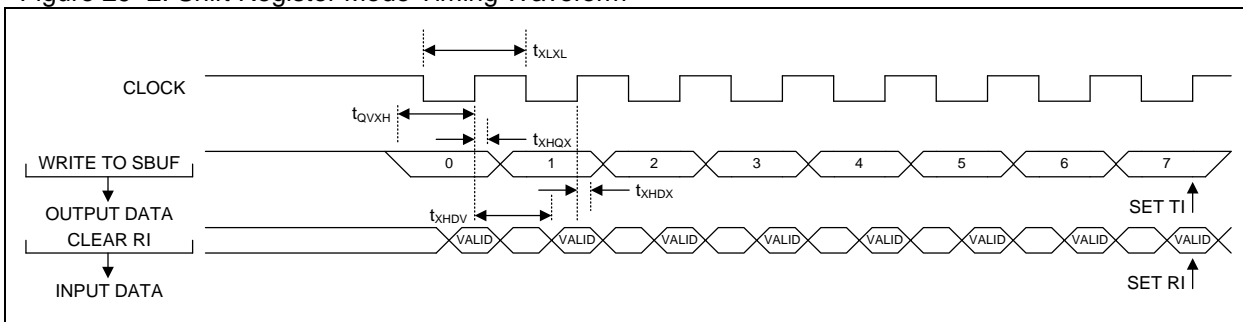
Parameter	Test Condition	Limits			Unit
		min	typ	max	
<b>Supply Range</b>					
Supply Voltage		2.4		5.5	V
<b>DC Accuracy</b>					
Resolution			12		bits
Integral Nonlinearity	VDD= VREF+= 5.0V		±2		LSB
	VDD= VREF+= 2.4V~5.5V			±4	LSB
	VDD > VREF+ & VREF+= 3.0V ~VDD			±4	LSB
Differential Nonlinearity	VDD= VREF+= 2.4V~5.5V			±1	LSB
	VDD > VREF+ & VREF+= 3.0V ~VDD			±1	LSB
Offset Error	VDD= VREF+= 2.4V~5.5V		0	±1	LSB
	VDD > VREF+ & VREF+= 3.0V ~VDD				
<b>Conversion Rate</b>					
SAR Conversion Clock			6		MHz
Conversion Time in SAR Clocks			24		clocks
Throughput Rate			250		ksps
<b>Analog Inputs</b>					
ADC Input Voltage Range	Single Ended (AIN+ – GND)	0		VREF+	V
	Differential (AIN+ – AIN–)	-0.5 * VREF+		+0.5 * VREF+	V
Input Capacitance			25		pF
<b>Power Consumption</b>					
Power Supply Current	Operating Mode, 250 ksps		4		mA

## 29.9. Serial Port Timing Characteristics

VDD = 5.0V±10%, VSS = 0V, T<sub>A</sub> = -40°C to +125°C, unless otherwise specified

Symbol	Parameter	URM0X6 = 0		URM0X6 = 1		Unit
		Min.	Max	Min.	Max	
t <sub>XLXL</sub>	Serial Port Clock Cycle Time	12T		2T		T <sub>SYSCLK</sub>
t <sub>QVXH</sub>	Output Data Setup to Clock Rising Edge	10T-20		T-20		ns
t <sub>XHQX</sub>	Output Data Hold after Clock Rising Edge	T-10		T-10		ns
t <sub>XHDX</sub>	Input Data Hold after Clock Rising Edge	0		0		ns
t <sub>XHDV</sub>	Clock Rising Edge to Input Data Valid		10T-20		2T-20	ns

Figure 29–2. Shift Register Mode Timing Waveform



## 29.10. SPI Timing Characteristics

VDD = 5.0V±10%, VSS = 0V, T<sub>A</sub> = -40°C to +125°C, unless otherwise specified

Symbol	Parameter	Min	Max	Units
<b>Master Mode Timing</b>				
t <sub>MCKH</sub>	SPICLK High Time	2T		T <sub>SYSCLK</sub>
t <sub>MCKL</sub>	SPICLK Low Time	2T		T <sub>SYSCLK</sub>
t <sub>MIS</sub>	MISO Valid to SPICLK Shift Edge	2T+20		ns
t <sub>MIH</sub>	SPICLK Shift Edge to MISO Change	0		ns
t <sub>MOH</sub>	SPICLK Shift Edge to MOSI Change		10	ns
<b>Slave Mode Timing</b>				
t <sub>SE</sub>	nSS Falling to First SPICLK Edge	2T		T <sub>SYSCLK</sub>
t <sub>SD</sub>	Last SPICLK Edge to nSS Rising	2T		T <sub>SYSCLK</sub>
t <sub>SEZ</sub>	nSS Falling to MISO Valid		4T	T <sub>SYSCLK</sub>
t <sub>SDZ</sub>	nSS Rising to MISO High-Z		4T	T <sub>SYSCLK</sub>
t <sub>CKH</sub>	SPICLK High Time	4T		T <sub>SYSCLK</sub>
t <sub>CKL</sub>	SPICLK Low Time	4T		T <sub>SYSCLK</sub>
t <sub>SIS</sub>	MOSI Valid to SPICLK Sample Edge	2T		T <sub>SYSCLK</sub>
t <sub>SIH</sub>	SPICLK Sample Edge to MOSI Change	2T		T <sub>SYSCLK</sub>
t <sub>SOH</sub>	SPICLK Shift Edge to MISO Change		4T	T <sub>SYSCLK</sub>
t <sub>SLH</sub>	Last SPICLK Edge to MISO Change (CPHA = 1 ONLY)	1T	2T	T <sub>SYSCLK</sub>

Figure 29–3. SPI Master Transfer Waveform with CPHA=0

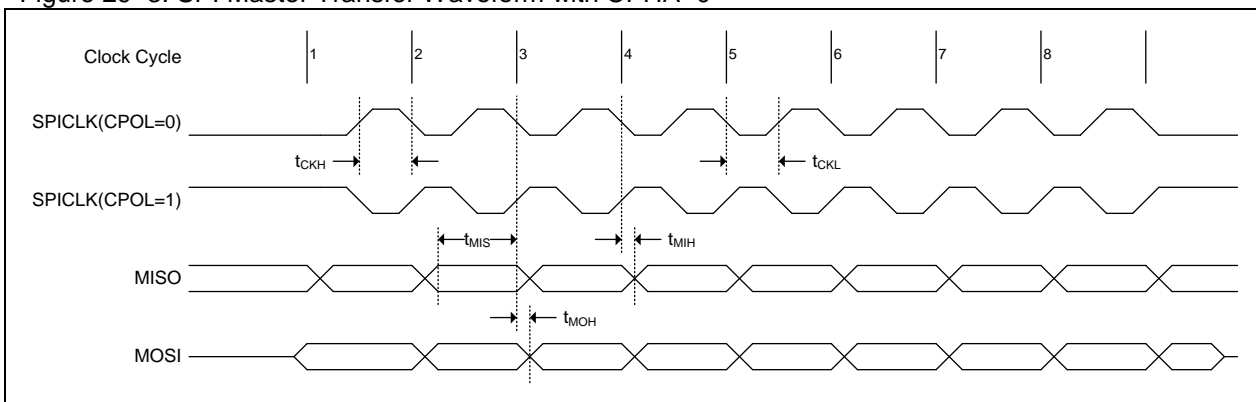


Figure 29–4. SPI Master Transfer Waveform with CPHA=1

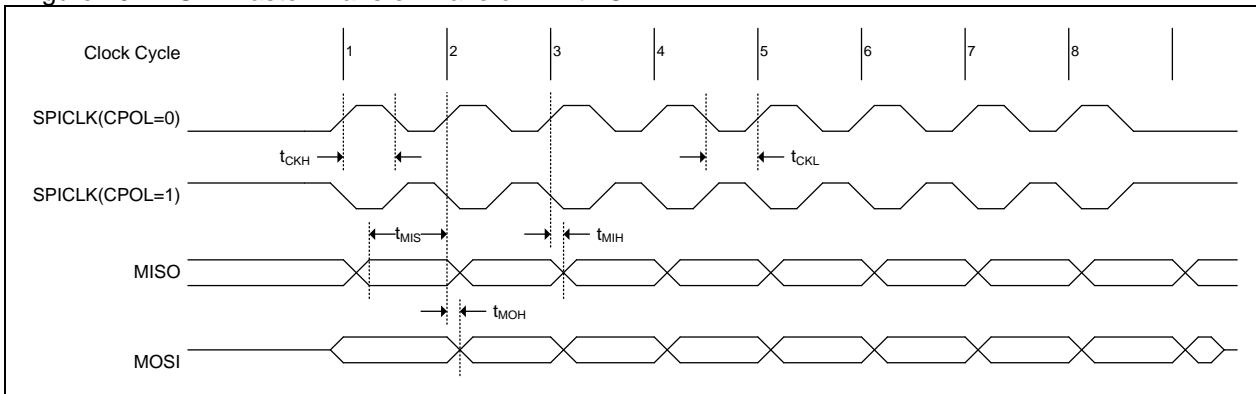


Figure 29–5. SPI Slave Transfer Waveform with CPHA=0

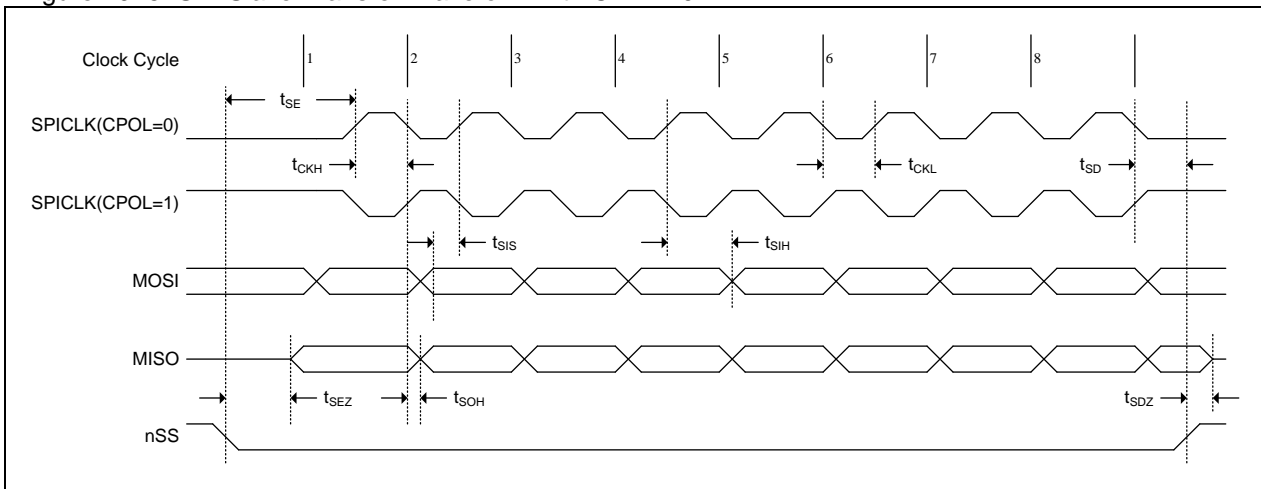
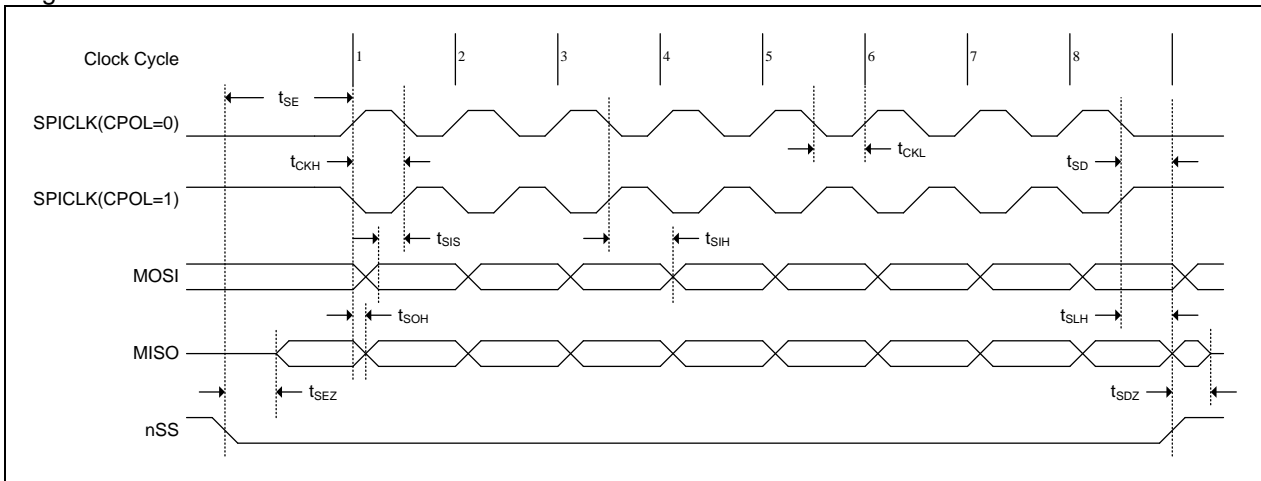


Figure 29–6. SPI Slave Transfer Waveform with CPHA=1



## 29.11. External Memory Cycle Timing Characteristics

Under operating conditions, load capacitance for Port 0, ALE, and PSEN = 100 pF; load capacitance for all other outputs = 80pF.  $T_A = -40^{\circ}\text{C}$  to  $+125^{\circ}\text{C}$ ,  $V_{DD}=5.0\text{V}\pm 10\%$ ,  $V_{SS}=0\text{V}$

T: Clock Cycle

M: Clock number of ALE Stretch,  $M = 0T\sim 3T$

N: Clock number of Read/Write Pulse Width Stretch,  $N = 0T \sim 7T$

L: Clock number of Read/Write pulse Setup/Hold Stretch,  $L = 0T \sim 1T$

Symbol	Parameter	Oscillator				Unit
		36MHz Without Stretched MOVX		36MHz with Stretched MOVX		
		Min.	Max	Min.	Max	
$1/t_{CLCL}$	Oscillator Frequency		36		36	MHz
$t_{LHLL}$	ALE Pulse Width	T-10		T+M-10		ns
$t_{AVLL}$	Address Valid to ALE Low	T-12		T+M-12		ns
$t_{LLAX}$	Address Hold after ALE Low	T-12		T+M-12		ns
$t_{RLRH}$	nRD Pulse Width	T-10		T+N-10		ns
$t_{WLWH}$	nWR Pulse Width	T-10		T+N-10		ns
$t_{RLDV}$	nRD Low to valid Data In		T-20		T+N-20	ns
$t_{RHDX}$	Data Hold After nRD	0		0		ns
$t_{RHDZ}$	Data Float After nRD		10		10	ns
$t_{LLDV}$	ALE Low to Valid Data In		3T-20		3T+M+L+N-20	ns
$t_{AVDV}$	Address to Valid Data In		4T-20		4T+2M+L+N-20	ns
$t_{LLWL}$	ALE Low to nRD or nWR Low	2T-10	2T+10	2T+2M+L-10	2T+2M+L+10	ns
$t_{AVWL}$	Address to nRD or nWR Low	3T-10		3T+2M+L-10		ns
$t_{WHQX}$	Data Hold After nWR	T-10		T+L-10		ns
$t_{QVWH}$	Data Valid to nWR High	2T-10		2T+L+N-10		ns
$t_{QVWX}$	Data Valid to nWR High to Low Transition	T-10		T+L-10		ns
$t_{RLAZ}$	nRD Low to Address Float		0		0	ns
$t_{WHLH}$	nRD or nWR High to ALE High	T-10		T+L-10		ns

**Explanation of Symbols** Each timing symbol has 5 characters. The first character is always a 't' (stands for time). The other characters, depending on their positions, stand for the name of a signal or the logical status of that signal. The following is a list of all the characters and what they stand for.

A: Address

C: Clock

D: Input data

H: Logic level HIGH

L: Logic level LOW or ALE

Q: Output data

R: RD signal

t: Time

V: Valid

W: WR signal

X: No longer a valid logic level

Z: High Impedance (Float)

For example:

$t_{AVLL}$  = Time from Address Valid to ALE Low

$t_{RLRH}$  = nRD Pulse Width

Figure 29–7. External Data Memory Read Cycle

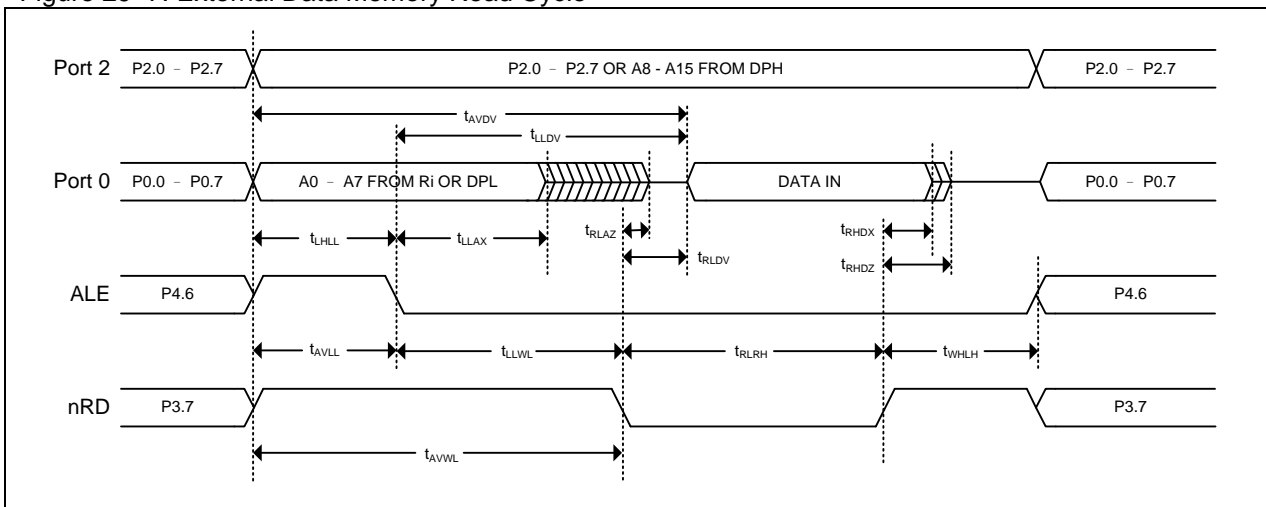
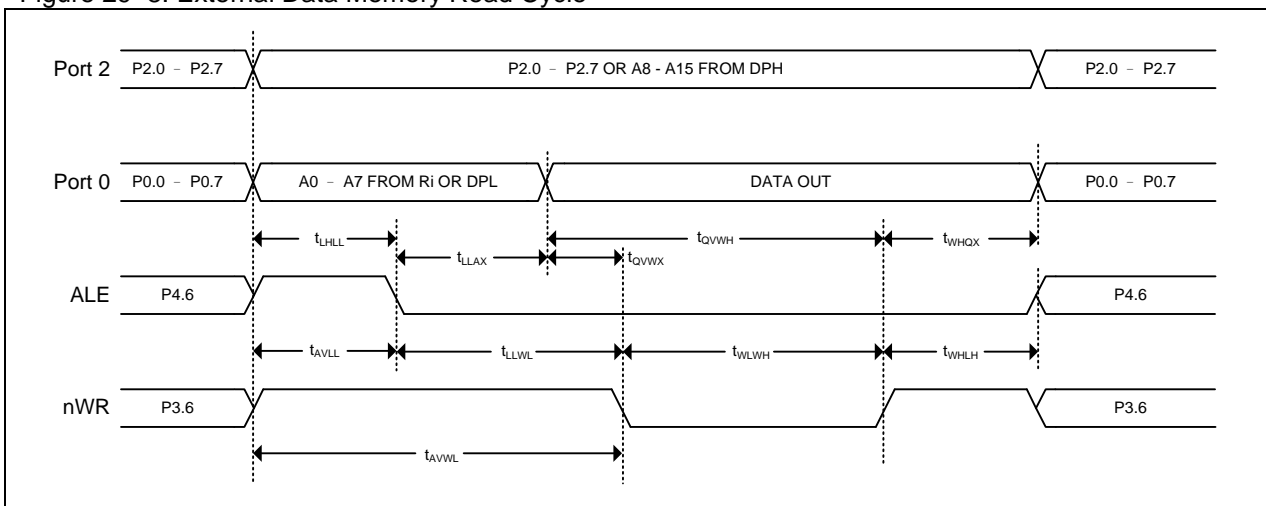


Figure 29–8. External Data Memory Read Cycle



## 29.12. USB Transceiver Electrical Characteristics

VDD = 4.0V ~ 5.5V, VSS = 0V, T<sub>A</sub> = 25°C, unless otherwise specified

Symbol	Parameter	Test Condition	Limits			Unit
			min	typ	max	
V <sub>33</sub>	3.3V regulator output voltage		3.0		3.6	V
R <sub>PU</sub>	Pull-Up Resistance	On DP	0.9	1.2	1.545	Kohm
<b>Transmitter</b>						
V <sub>OH</sub>	Output High Voltage		2.8			V
V <sub>OL</sub>	Output Low Voltage				0.8	V
V <sub>CRS</sub>	Output Cross Over point		1.3		2.0	V
Z <sub>DRVH</sub>	Output Impedance on Driving High		28		44	Ohm
Z <sub>DRVL</sub>	Output Impedance on Driving Low		28		44	Ohm
T <sub>R</sub>	Output Rise Time		4		20	ns
T <sub>F</sub>	Output Fall Time		4		20	ns
<b>Receiver</b>						
V <sub>DI</sub>	Differential Input Sensitivity	DP – DM	0.2			V
V <sub>CM</sub>	Differential Input Common Mode Range		0.8		2.5	V
I <sub>L</sub>	Input Leakage current	Pull-up Disabled		<1.0		uA



## 30. Instruction Set

Table 30–1. Instruction Set

MNEMONIC	DESCRIPTION	BYTE	EXECUTION Cycles
<b>DATA TRASFER</b>			
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
MOVX A,@Ri	Move external RAM(8-bit address) to Acc	1	3 ~ 20 <sup>Note1</sup>
MOVX A,@DPTR	Move external RAM(16-bit address) to Acc	1	3 ~ 20 <sup>Note1</sup>
MOVX @Ri,A	Move Acc to external RAM(8-bit address)	1	3 ~ 20 <sup>Note1</sup>
MOVX @DPTR,A	Move Acc to external RAM(16-bit address)	1	3 ~ 20 <sup>Note1</sup>
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
<b>ARITHMETIC OPERATIONS</b>			
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
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
<b>LOGIC OPERATION</b>			
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
<b>BOOLEAN VARIABLE MANIPULATION</b>			
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
<b>BOOLEAN VARIABLE MANIPULATION</b>			
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
<b>PROAGRAM BRACHING</b>			
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

Note 1: The cycle time for access of external auxiliary RAM is:

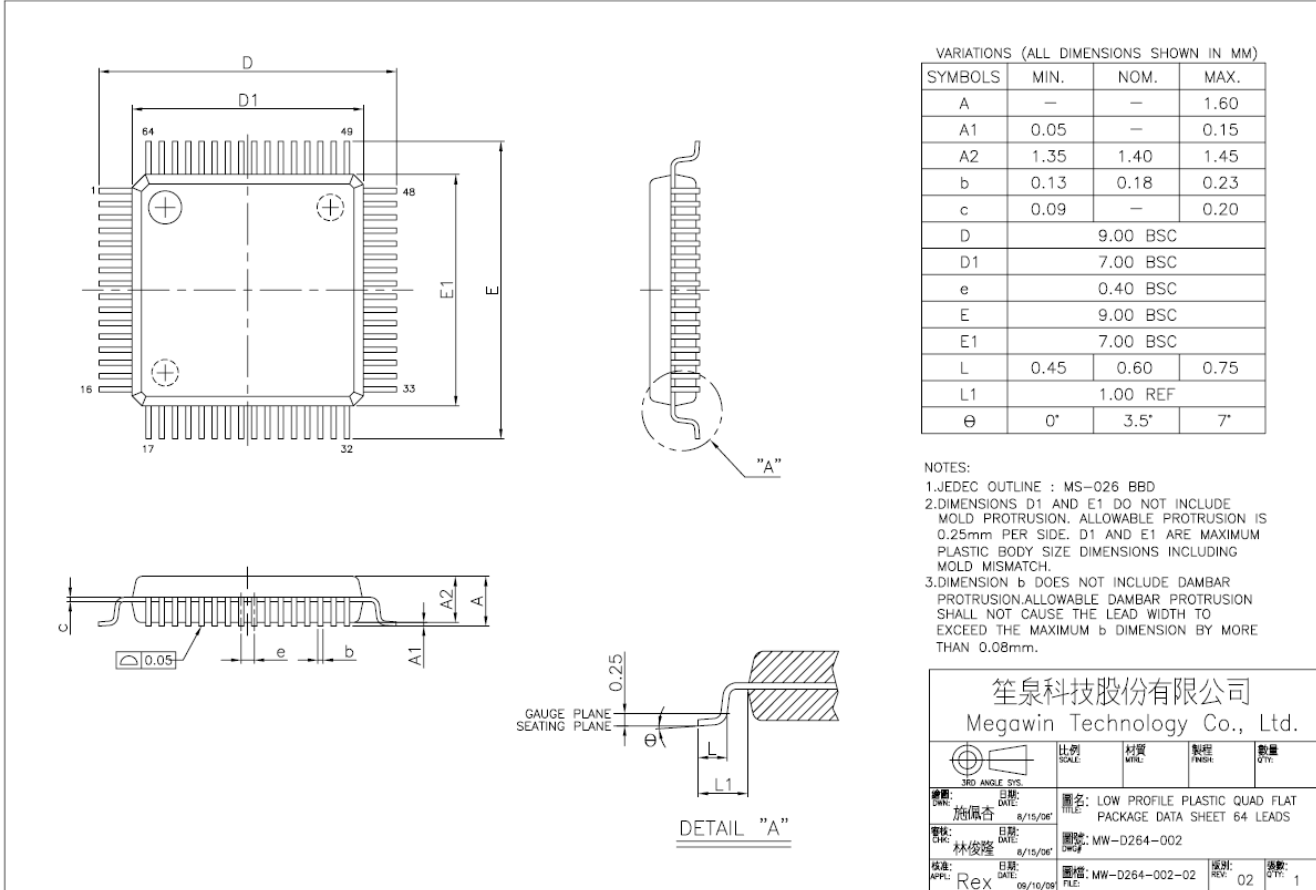
$$EMAI1 = 0: 5 + 2 \times ALE\_Stretch + RW\_Stretch + 2 \times RWSH; (5\sim 20)$$

$$EMAI1 = 1: 3 + RW\_Stretch + 2 \times RWSH; (3\sim 12)$$

# 31. Package Dimension

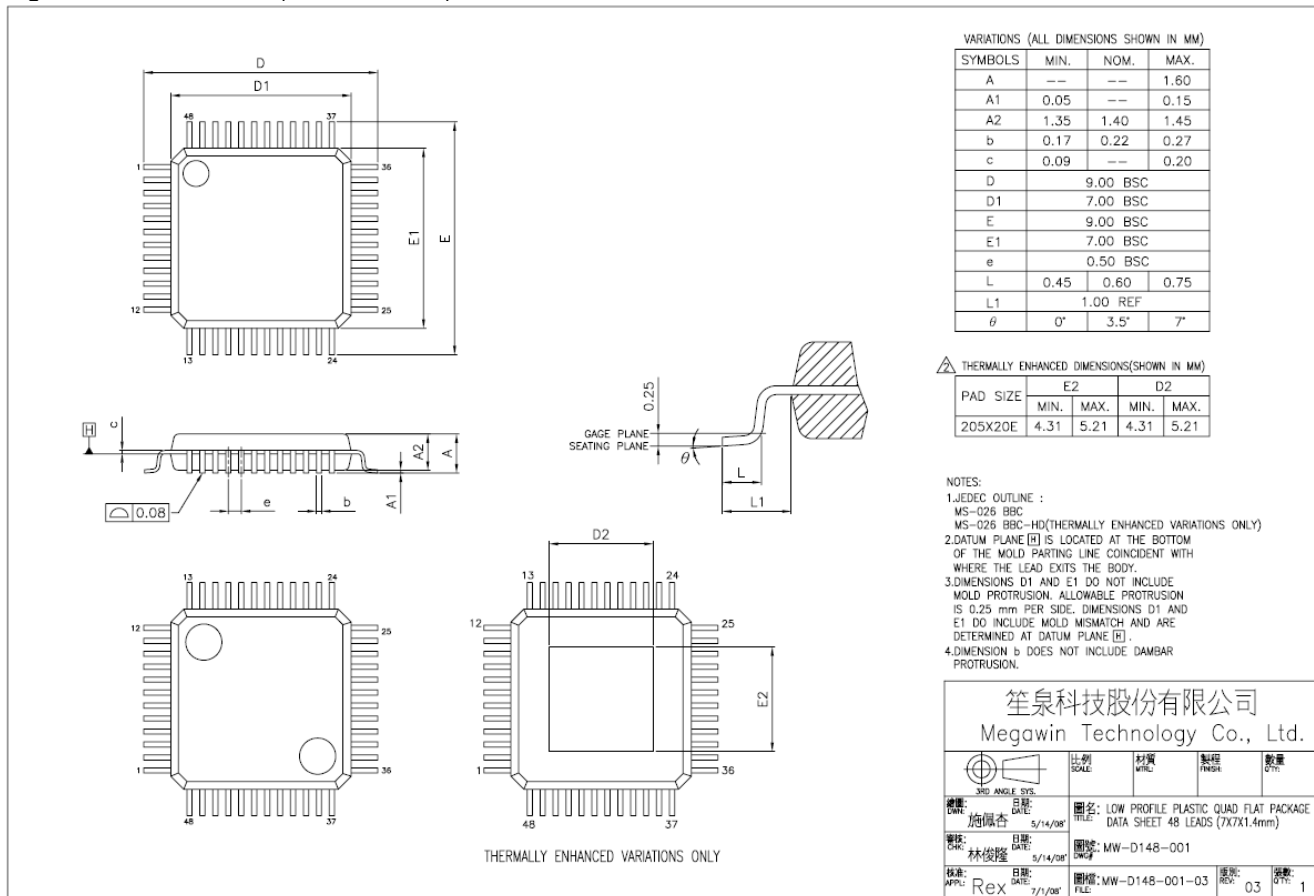
## 31.1. LQFP-64 (7mm X 7mm)

Figure 31-1. LQFP-64 (7mm X 7mm )



### 31.2. LQFP-48 (7mm X 7mm)

Figure 31–2. LQFP-48 (7mm X 7mm)



笙泉科技股份有限公司  
Megawin Technology Co., Ltd.

比例 SCALE	材質 WTL	製程 FINISH	數量 QTY
3RD ANGLE SYS.			
標準: 施隔合	品名: MW-D148-001	圖名: LOW PROFILE PLASTIC QUAD FLAT PACKAGE DATA SHEET 48 LEADS (7X7X1.4mm)	
審核: 林俊隆	日期: 5/14/08	圖號: MW-D148-001	
核准: Rex	日期: 7/1/08	圖號: MW-D148-001-03	版數: 03
		圖號: MW-D148-001-03	數量: 1

## 32. Revision History

Table 32–1. Revision History

Rev	Descriptions	Date
V1.20	1. Preliminary version release.	2011/03/08
v1.21	1. Correct ISPCR SFR address to 0xE7 in section 10.4 and section 23.	2011/10/12
	2. Modify CLRW description in WDT section and update WDT block diagram.	2011/10/12
	3. In section 21.2.2, modify timer1/3 → timer0/3	2011/10/12
	4. Remove DISREG, POF1 & IARF.	2011/10/12
	5. Correct SPEN0 to SPCON0 in Section 3.3.	2011/10/12
	6. Correct SICON address from F8H to D4H in Section 3.2.	2011/10/12
	7. In Section 21, modify description for “Figure 16-1” to “Figure 21-1”.	2011/10/12
	8. Correct bit name in P1AIO SFR in Section 12.2.2 and Section 21.3.	2011/11/10
	9. In Section 12.2.8, correct “P4.6 & P4.4” and “P4.3 & P4.0” to “P4.6 ~ P4.4” and “P4.3 ~ P4.0”.	2011/11/10
	10. Correct SFR name “SCON0, SBUF0, SCFG0, SCON1, SBUF1 and SCFG1” to “S0CON, S0BUF, S0CFG, S1CON, S1BUF and S1CFG.”	2011/11/10
	11. Change bus naming rule from “[n]” to “.n”, n = 0~ 7, such as S0BUIF.7.	2011/11/10
	12. Add description for EXEN2 in Section 14.2.5.	2011/11/10
	13. Update PCA port I/O to Port 2 in Section 17 and <b>PWM diagrams</b> .	2011/11/10
	14. In Section 13.7 and 26, modify bit address to 5~4 for INT2IS1 and INT2SI0.	2011/11/10
	15. In Section 14.3.3, modify description for T3CKO on P0.1.	2011/11/10
	16. Update section 16.2.3 equation for S1TX12.	2011/11/10
	17. Update System Clock Diagram.	2011/11/10
	18. Add tables for Timer 1 baud rate setting.	2011/11/10
	19. Correct the equation for Timer2 baud rate generator and add tables for Timer 2 baud rate setting.	2011/11/10
	20. Add S1 Mode 1 & 3 baud rate setting description in section 16.2.3.	2011/11/10
	21. Combine ISP & IAP to one Section	2011/11/10
	22. Modify Electrical Characteristics	2011/11/14
	23. Modify Hardware Option description and modify Absolute Maximum Rating	2011/11/14
	24. Add Section 27: Application Notes.	2011/11/14
	25. Move Absolute Maximum Rating into Section Electrical Characteristics	2011/11/30
	26. In section 17.1, modify PCA port pin from Port 1 to Port 2.	2011/12/20
	27. In section 17.2, modify ECI from P1.1 to P2.1.	2011/12/20
	28. In section 21.2, modify ADC result from {ADCH, ADCL} to {ADCDH, ADCDL}.	2011/12/20
	29 Add Section 12.1.7 for General Port Input Configured.	2012/01/18
	30. Add Package Dimension & remove 84FG532 in Figure 6-5 ~ 6-7.	2012/01/19
A1	New Datasheet Format	2013/01/11
	1. Remove "Under-Line" for Section 14.1 & Section 14.1.1	2013/05/15
	2. In Section 28.4, IHRCO Characteristics, supply voltage 2.2 --> 2.0	
	3. In the "Feature", modify "Master/Slave tow wire serial interface" to "Master/Slave two wire serial interface"	
	4. In Section 22.2.6, modify ADCE to EADC	

	<p>5. In Section 22.2, ADPS2~0 --&gt; ADCKS2~0</p> <p>6. In Section 22.2, T3CKO --&gt; the timer overflow rate of S1BRG and Timer 3</p> <p>7. In Section 22.2.2, item 5, Input-Only --&gt; Analog-Input-Only</p> <p>8. In Section 28.2</p> <p>8.1. ILL1: add "or other Input port with on-chip pull-up resistor"</p> <p>8.2. IH2L: add "or other Input port with on-chip pull-up resistor"</p> <p>8.3. IOH1: add "or other open-drain output port with on-chip pull-up resistor"</p> <p>8.4. VBOD11, VBOD12, VBOD13 for BOD1 on 2.4V, 3.7V and 4.2V</p> <p>8.5. Modify VBOD12 Min value from 3.65 to 3.55</p> <p>9. In Section 4.2, modify TWSI_SCL and TWSI_SDA to TWI_SCL and TWI_SDA.</p> <p>10. (MCU will not keep SFRPI value in interrupt. User need to keep SFRPI value "by yourself". ) --&gt; in software flow.</p> <p>11. In Section 8, modify description for IHRCO and add description for CKM and ILRCO</p> <p>12. delete "(Note: If SSIG=1, CPHA must not be 1, otherwise the operation is not defined.)" in CPHA description in SPI register.</p> <p>13. In ADC section, Single-ended "modes" &gt;&gt; mode</p> <p>14. In Section 15.7.3, Table16-3 ~ Table16-10 --&gt; Table15-3 ~ Table15-10, Table 16-12 ~ Table 16-19 --&gt; Table15-12 ~ Table15-19</p> <p>15. In Section 22.2.4, modify Fosc to SYSCLK</p> <p>16. In ADC Section, ADEN --&gt; ADCEN</p> <p>17. In SFR bit map, BS --&gt; SWBS, SRST --&gt; SWRST</p> <p>18. In section 13.5, IPH --&gt; IP0H, and add "IE, IP0L, IP0H, EIE1, EIP1L or EIP1H" in the last description</p> <p>19. In SFR bit map, add "x" in IFMT &amp; SCMD reset value</p> <p>20. modify flahs description in Feature</p> <p>21. Rename section from "In-Application-Programming" to "IAP Operation"</p> <p>22. modify Figure 23-1</p> <p>23. In "IAP memory boundary/range" section, modify the content in the equation: IAP --&gt; IAP size</p> <p>24. Link Figure 23-1 in section " Notes for ISP"</p>	
A1.1	Add sample code	2013/10/30
A1.2	Remove CHRL, CLRL description on PCA_PWM sample code	2014/02/06
A1.3	Modify error Disable UART1 function moved to P4 as P5 (Page-204)	2014/02/18
A1.4	Added section of ordering information	2015/08/01
A1.5	Modify the contents of PCON1	2015/09/25

## **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).