Product Overview

Address Spaces

Addressing Modes

Memory Map

SAM47 Instruction Set

KS57C21832/P21832 PRODUCT OVERVIEW

1

# **PRODUCT OVERVIEW**

## **OVERVIEW**

The KS57C21832 single-chip CMOS microcontroller has been designed for high performance using Samsung's newest 4-bit CPU core, SAM47 (Samsung Arrangeable Microcontrollers).

With an up-to-896-dot LCD direct drive capability, 8-bit timer/counter 0, 16-bit timer/counter 1, and serial I/O, the KS57C21832 offers an excellent design solution for a wide variety of applications which require LCD functions.

Up to 39 pins of the 100-pin QFP package can be dedicated to I/O. Eight vectored interrupts provide fast response to internal and external events. In addition, the KS57C21832's advanced CMOS technology provides for low power consumption and a wide operating voltage range.

## **OTP**

The KS57C21832 microcontroller is also available in OTP (One Time Programmable) version, KS57P21832. KS57P21832 microcontroller has an on-chip 32 K-byte one-time-programable EPROM instead of masked ROM. The KS57P21832 is comparable to KS57C21832, both in function and in pin configuration.



PRODUCT OVERVIEW KS57C21832/P21832

## **FEATURES SUMMARY**

## Memory

- 8,192 × 4-bit RAM (excluding LCD display RAM)
- 32,768 × 8-bit ROM

#### 39 I/O Pins

- I/O: 35 pins
- · Input only: 4 pins

#### LCD Controller/Driver

- 56 segments and 16 common terminals
- 8 and 16 common selectable
- · Internal resistor circuit for LCD bias
- All dot can be switched on/off

#### 8-bit Basic Timer

- 4 interval timer functions
- Watchdog timer

## 8-bit Timer/Counter 0

- Programmable 8-bit timer
- External event counter
- · Arbitrary clock frequency output
- · External clock signal divider
- Serial I/O interface clock generator

## 16-Bit Timer/Counter 1

- Programmable 16-bit timer
- · External event counter
- · Arbitrary clock frequency output
- External clock signal divider

#### 8-bit Serial I/O Interface

- 8-bit transmit/receive mode
- 8-bit receive mode
- · LSB-first or MSB-first transmission selectable
- Internal or external clock source

## Memory-Mapped I/O Structure

Data memory bank 15

#### **Watch Timer**

- Time interval generation: 0.5 s, 3.9 ms at 32768 Hz
- 4 frequency outputs to BUZ pin
- Clock source generation for LCD

## Interrupts

- · Four internal vectored interrupts
- Four external vectored interrupts
- · Two quasi-interrupts

## **Bit Sequential Carrier**

Supports 16-bit serial data transfer in arbitrary format

#### **Power-Down Modes**

- Idle mode (only CPU clock stops)
- Stop mode (main system clock and CPU clock stop)
- Sub-system clock stop mode

## **Oscillation Sources**

- · Crystal, ceramic, or RC for main system clock
- Crystal oscillator for subsystem clock
- Main system clock frequency: 0.4 6 MHz
- Subsystem clock frequency: 32.768 kHz
- CPU clock divider circuit (by 4, 8, or 64)

#### Instruction Execution Times

- 0.67, 1.33, 10.7 µs at 6 MHz
- 0.95, 1.91, 15.3 µs at 4.19 MHz
- 122 µs at 32.768 kHz

#### **Operating Temperature**

- 40 °C to 85 °C

## **Operating Voltage Range**

• 1.8 V to 5.5 V (3.0 MHz @ 1.8 V)

## Package Type

100-pin QFP



KS57C21832/P21832 PRODUCT OVERVIEW

## **BLOCK DIAGRAM**

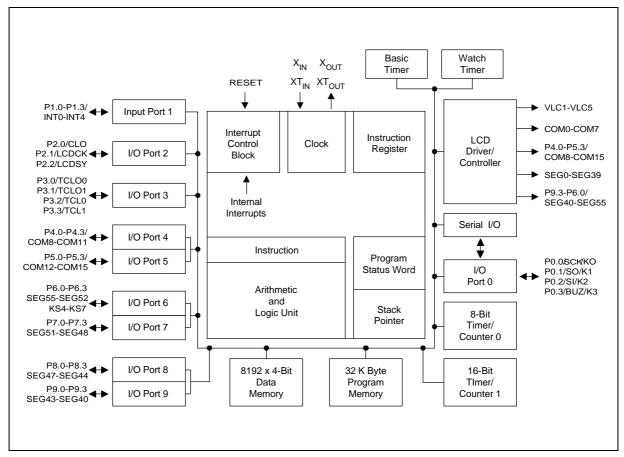


Figure 1-1. KS57C21832 Simplified Block Diagram



PRODUCT OVERVIEW KS57C21832/P21832

## **PIN ASSIGNMENTS**

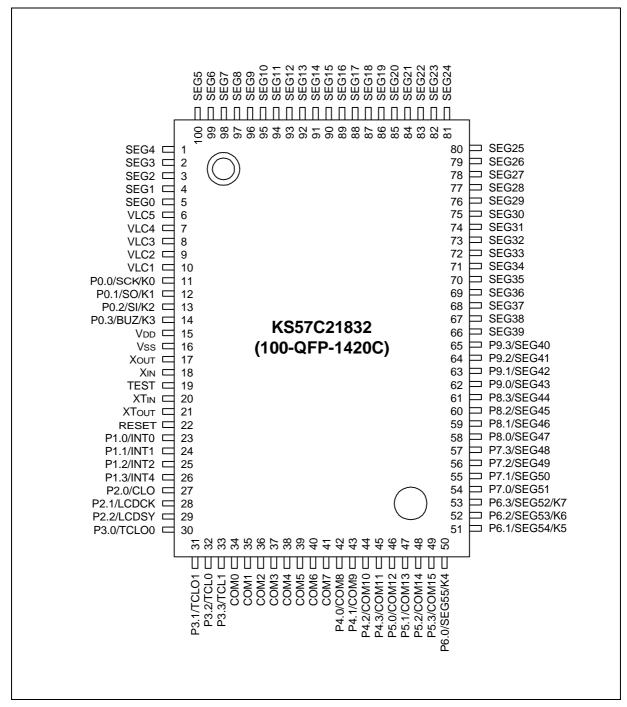


Figure 1-2. KS57C21832 100-QFP Pin Assignment Diagram



KS57C21832/P21832 PRODUCT OVERVIEW

# **PIN DESCRIPTIONS**

Table 1-1. KS57C21832 Pin Descriptions

Pin Name	Pin Type	Description	Number	Share Pin
P0.0 P0.1 P0.2 P0.3	I/O	4-bit I/O port. 1-bit and 4-bit read/write and test are possible. Individual pins are software configurable as input or output. Individual pins are software configurable as opendrain or push-pull output. 4-bit pull-up resistors are software assignable; pull-up resistors are automatically disabled for output pins.	11 12 13 14	SCK/K0 SO/K1 SI/K2 BUZ/K3
P1.0 P1.1 P1.2 P1.3	I	<ul><li>4-bit input port.</li><li>1-bit and 4-bit read and test are possible.</li><li>4-bit pull-up resistors are assignable by software.</li></ul>	23 24 25 26	INTO INT1 INT2 INT4
P2.0 P2.1 P2.2	I/O	Same as port 0 except that port 2 is 3-bit I/O port.	27 28 29	CLO LCDCK LCDSY
P3.0 P3.1 P3.2 P3.3	I/O	Same as port 0.	30 31 32 33	TCLO0 TCLO1 TCL0 TCL1
P4.0-P4.3 P5.0-P5.3	I/O	<ul> <li>4-bit I/O ports.</li> <li>1-, 4-bit or 8-bit read/write and test are possible.</li> <li>Individual pins are software configurable as input or output.</li> <li>4-bit pull-up resistors are software assignable; pull-up resistors are automatically disabled for output pins.</li> </ul>	42-45 46-49	COM8- COM11 COM12- COM15
P6.0-P6.3	I/O	Same as P4, P5.	50-53	SEG55/K4- SEG52/K7
P7.0-P7.3			54-57	SEG51- SEG48
P8.0-P8.3 P9.0-P9.3	I/O	Same as P4, P5.	58-61 62-65	SEG47- SEG44 SEG43- SEG40
SCK	I/O	Serial I/O interface clock signal.	11	P0.0/K0
SO	I/O	Serial data output.	12	P0.1/K1
SI	I/O	Serial data input.	13	P0.2/K2
BUZ	I/O	2 kHz, 4 kHz, 8 kHz or 16 kHz frequency output for buzzer signal.	14	P0.3/K3
INTO, INT1	I	External interrupts. The triggering edge for INT0 and INT1 is selectable.	23, 24	P1.0, P1.1



PRODUCT OVERVIEW KS57C21832/P21832

Table 1-1. KS57C21832 Pin Descriptions (Continued)

Pin Name	Pin Type	Description	Number	Share Pin
INT2	I	Quasi-interrupt with detection of rising or falling edges.	25	P1.2
INT4	I	External interrupt with detection of rising or falling edges.	26	P1.3
CLO	I/O	Clock output .	27	P2.0
LCDCK	I/O	LCD clock output for display expansion.	28	P2.1
LCDSY	I/O	LCD synchronization clock output for display expansion.	29	P2.2
TCLO0	I/O	Timer/counter 0 clock output.	30	P3.0
TCLO1	I/O	Timer/counter 1 clock output.	31	P3.1
TCL0	I/O	External clock input for timer/counter 0.	32	P3.2
TCL1	I/O	External clock input for timer/counter 1.	33	P3.3
COM0-COM7	0	LCD common signal output.	34-41	_
COM8-COM11	I/O		42-45	P4.0-P4.3
COM12-COM15			46-49	P5.0-P5.3
SEG0-SEG39	0	LCD segment signal output.	5-1, 100-66	-
SEG40-SEG43	I/O		65-62	P9.3-P9.0
SEG44-SEG47			61-58	P8.3-P8.0
SEG48-SEG51			57-54	P7.3-P7.0
SEG52-SEG55			53-50	P6.3/K7-P6.0/K4
K0-K3	I/O	External interrupt. The triggering edge is selectable.	11-14	P0.0-P0.3
K4-K7			50-53	P6.0-P6.3
$V_{DD}$	_	Main power supply.	15	-
V <sub>SS</sub>	_	Ground.	16	_
RESET	I	Reset signal.	22	_
V <sub>LC1</sub> -V <sub>LC5</sub>	_	LCD power supply.	10-6	_
X <sub>in</sub> , X <sub>out</sub>	_	Crystal, Ceramic or RC oscillator pins for system clock.	18, 17	-
XT <sub>in,</sub> XT <sub>out</sub>	_	Crystal oscillator pins for subsystem clock.	20, 21	
TEST	I	Test signal input. (must be connected to V <sub>SS</sub> )	19	-

**NOTE:** Pull-up resistors for all I/O ports are automatically disabled if they are configured to output mode.



KS57C21832/P21832 PRODUCT OVERVIEW

Table 1-2. Overview of KS57C21832 Pin Data

Pin Names	Share Pins	I/O Type	Reset Value	Circuit Type
P0.1, P0.3	SO/K1, BUZ/K3	I/O	Input	E-1
P0.0, P0.2	SCK/K0, SI/K2	I/O	Input	E-2
P1.0-P1.3	INT0-INT2, INT4	I	Input	A-3
P2.0-P2.2	CLO, LCDCK, LCDSY	I/O	Input	E
P3.0-P3.1	TCLO0, TCLO1	I/O	Input	Е
P3.2-P3.3	TCL0, TCL1	I/O	Input	E-1
P4.0-P4.3 P5.0-P5.3	COM8-COM11 COM12-COM15	I/O	Input	H-13
P6.0-P6.3	SEG55/K4-SEG52/K7	I/O	Input	H-16
P7.0-P7.3	SEG51-SEG48	I/O	Input	H-13
P8.0-P8.3 P9.0-P9.3	SEG47-SEG44 SEG43-SEG40	I/O	Input	H-13
COM0-COM7	_	0	High	H-3
SEG0-SEG39	_	0	High	H-15
$V_{DD}$	_	_	_	_
V <sub>SS</sub>	_	-	_	_
RESET	_	I	-	В
V <sub>LC1</sub> -V <sub>LC5</sub>	-			
X <sub>IN,</sub> X <sub>OUT</sub>	-	_	_	_
XT <sub>IN,</sub> XT <sub>OUT</sub>	-	-	_	_
TEST	-	I	_	_



PRODUCT OVERVIEW KS57C21832/P21832

## **PIN CIRCUIT DIAGRAMS**

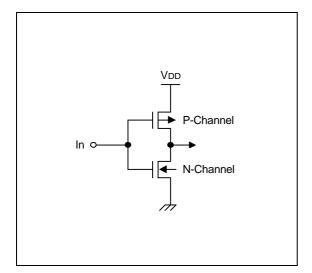


Figure 1-3. Pin Circuit Type A

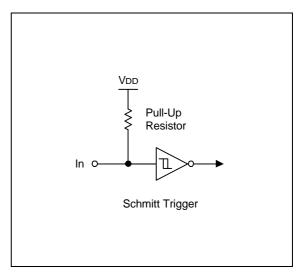


Figure 1-5. Pin Circuit Type B

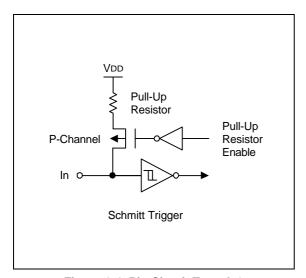


Figure 1-4. Pin Circuit Type A-3

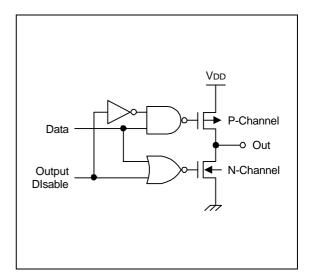


Figure 1-6. Pin Circuit Type C



KS57C21832/P21832 PRODUCT OVERVIEW

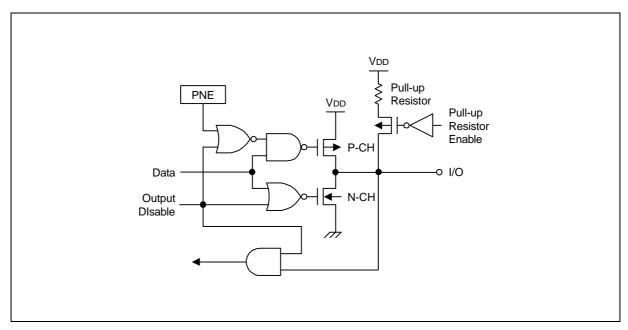


Figure 1-7. Pin Circuit Type E

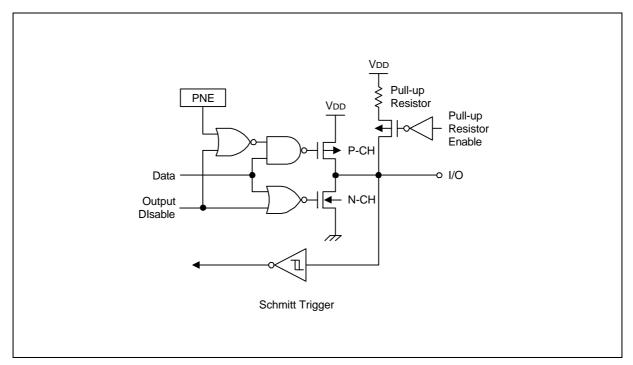


Figure 1-8. Pin Circuit Type E-1

PRODUCT OVERVIEW KS57C21832/P21832

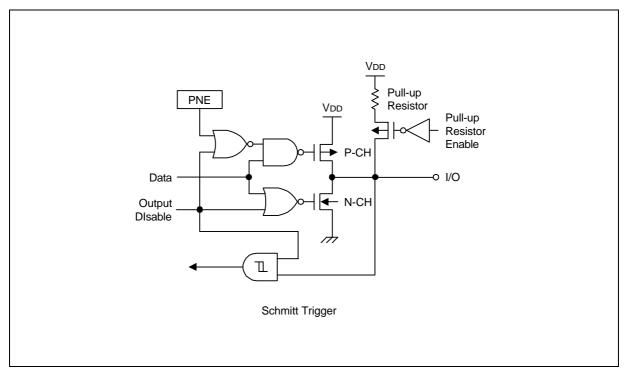


Figure 1-9. Pin Circuit Type E-2

KS57C21832/P21832 PRODUCT OVERVIEW

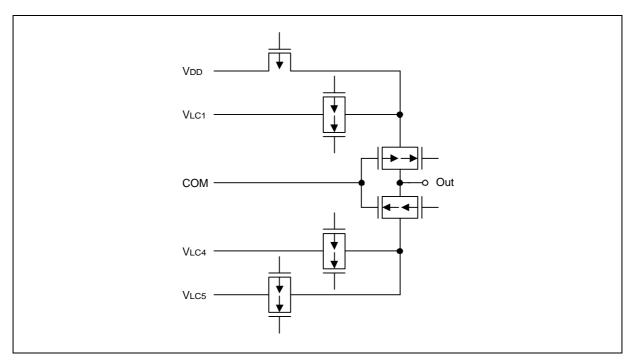


Figure 1-10. Pin Circuit Type H-3

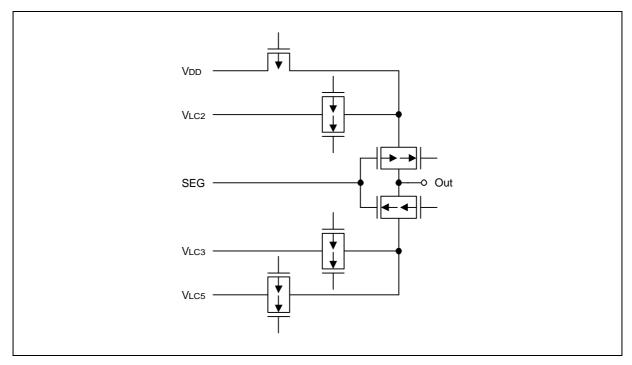


Figure 1-11. Pin Circuit Type H-15



PRODUCT OVERVIEW KS57C21832/P21832

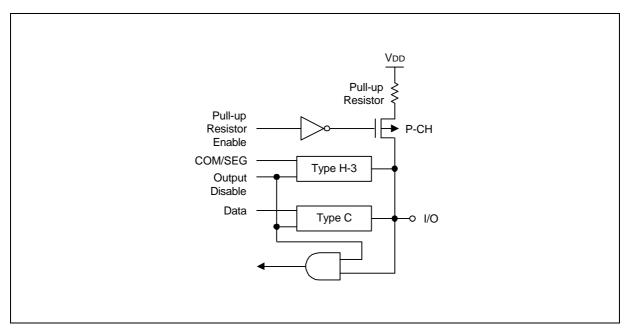


Figure 1-12. Pin Circuit Type H-13

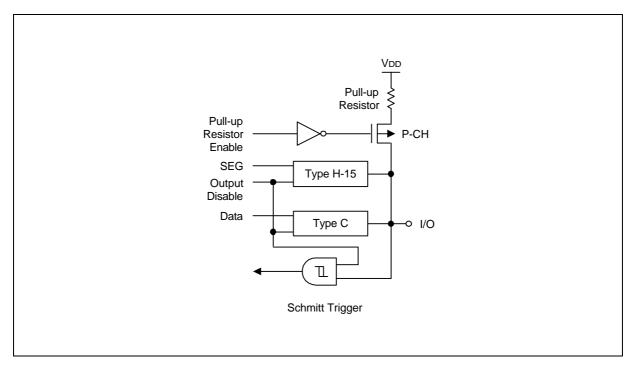


Figure 1-13. Pin Circuit Type H-16



# 2

# ADDRESS SPACES

## **PROGRAM MEMORY (ROM)**

## **OVERVIEW**

ROM maps for KS57C21832 devices are mask programmable at the factory. In its standard configuration, the device's  $32,768 \times 8$ -bit program memory has three areas that are directly addressable by the program counter (PC):

- 16-byte area for vector addresses
- 96-byte instruction reference area
- 16-byte general-purpose area
- 32,640-byte general-purpose area

## **General-purpose Program Memory**

Two program memory areas are allocated for general-purpose use: One area is 16 bytes in size and the other is 32,640 bytes.

#### **Vector Addresses**

A 16-byte vector address area is used to store the vector addresses required to execute system resets and interrupts. Start addresses for interrupt service routines are stored in this area, along with the values of the enable memory bank (EMB) and enable register bank (ERB) flags that are used to set their initial value for the corresponding service routines. The 16-byte area can be used alternately as general-purpose ROM.

## **REF Instructions**

Locations 0020H-007FH are used as a reference area (look-up table) for 1-byte REF instructions. The REF instruction reduces the byte size of instruction operands. REF can reference one 2-byte instruction, two 1-byte instructions, and one 3-byte instructions which are stored in the look-up table. Unused look-up table addresses can be used as general-purpose ROM.

**Table 2-1. Program Memory Address Ranges** 

ROM Area Function	Address Ranges	Area Size (in Bytes)
Vector address area	0000H-000FH	16
General-purpose program memory	0010H-001FH	16
REF instruction look-up table area	0020H-007FH	96
General-purpose program memory	0080H-7FFFH	32,640



#### **GENERAL-PURPOSE MEMORY AREAS**

The 16-byte area at ROM locations 0010H-001FH and the 32,640-byte area at ROM locations 0080H-7FFFH are used as general-purpose program memory. Unused locations in the vector address area and REF instruction look-up table areas can be used as general-purpose program memory. However, care must be taken not to overwrite live data when writing programs that use special-purpose areas of the ROM.

## **VECTOR ADDRESS AREA**

The 16-byte vector address area of the ROM is used to store the vector addresses for executing system resets and interrupts. The starting addresses of interrupt service routines are stored in this area, along with the enable memory bank (EMB) and enable register bank (ERB) flag values that are needed to initialize the service routines. 16-byte vector addresses are organized as follows:

EMB	ERB	PC13	PC12	PC11	PC10	PC9	PC8
PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

To set up the vector address area for specific programs, use the instruction VENTn. The programming tips on the next page explain how to do this.

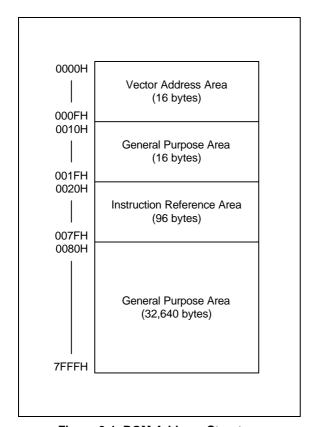


Figure 2-1. ROM Address Structure

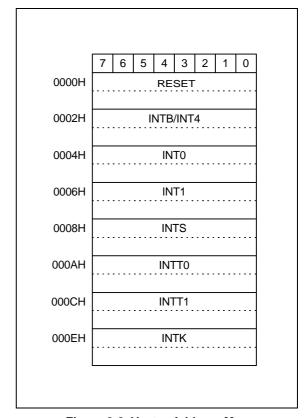


Figure 2-2. Vector Address Map



## PROGRAMMING TIP — Defining Vectored Interrupts

The following examples show you several ways you can define the vectored interrupt and instruction reference areas in program memory:

1. When all vector interrupts are used:

```
ORG
              0000H
                                    ; EMB \leftarrow 1, ERB \leftarrow 0; Jump to RESET address by RESET
VENT0
              1,0,RESET
                                    ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTB address by INTB
VENT1
              0,0,INTB
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INT0 address by INT0
VENT2
              0,0,INT0
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INT1 address by INT1
VENT3
              0,0,INT1
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTS address by INTS
VENT4
              0,0,INTS
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTT0 address by INTT0
VENT5
              0,0,INTT0
                                    ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTT1 address by INTT1
VENT6
              0,0,INTT1
                                    ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTK address by INTK
VENT7
              0,0,INTK
```

2. When a specific vectored interrupt such as INT0, and INTT0 is not used, the unused vector interrupt locations must be skipped with the assembly instruction ORG so that jumps will address the correct locations:

```
ORG
             0000H
                                   ; EMB \leftarrow 1, ERB \leftarrow 0; Jump to RESET address by RESET
VENT0
             1,0,RESET
                                   : EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTB address by INTB
VENT1
             0,0,INTB
ORG
             0006H
                                   ; INT0 interrupt not used
VENT3
             0.0.INT1
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INT1 address by INT1
VENT4
             0,0,INTS
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTS address by INTS
ORG
             000CH
                                   ; INTT0 interrupt not used
VENT6
             0,0,INTT1
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTT1 address by INTT1
VENT7
                                    ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTK address by INTK
             0,0,INTK
             0010H
ORG
```



## PROGRAMMING TIP — Defining Vectored Interrupts (Continued)

3. If an INTO interrupt is not used and if its corresponding vector interrupt area is not fully utilized, or if it is not written by a ORG instruction as in Example 2, a CPU malfunction will occur:

```
ORG
             0000H
                                  ; EMB \leftarrow 1, ERB \leftarrow 0; Jump to RESET address by RESET
VENT0
             1,0,RESET
                                  ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTB address by INTB
VENT1
             0,0,INTB
                                  ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INT1 address by INT0
VENT3
             0,0,INT1
VENT4
                                  ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTS address by INT1
             0,0,INTS
VENT5
                                  ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTT0 address by INTS
             0,0,INTT0
VENT6
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTT1 address by INTT0
             0,0,INTT1
VENT7
                                   ; EMB \leftarrow 0, ERB \leftarrow 0; Jump to INTK address by INTT1
             0,0,INTK
ORG
             0010H
General-purpose ROM area
```

In this example, when an INTS interrupt is generated, the corresponding vector area is not VENT4 INTS, but VENT5 INTT0. This causes an INTS interrupt to jump incorrectly to the INTT0 address and causes a CPU malfunction to occur.



#### **INSTRUCTION REFERENCE AREA**

Using 1-byte REF instructions, you can easily reference instructions with larger byte sizes that are stored in addresses 0020H-007FH of program memory. This 96-byte area is called the REF instruction reference area, or look-up table. Locations in the REF look-up table may contain two 1-byte instructions, one 2-byte instruction, or one 3-byte instruction such as a JP (jump) or CALL. The starting address of the instruction you are referencing must always be an even number. To reference a JP or CALL instruction, it must be written to the reference area in a two-byte format: for JP, this format is TJP; for CALL, it is TCALL. In summary, there are three ways to the REF instruction:

By using REF instructions you can execute instructions larger than one byte. In summary, there are three ways you can use the REF instruction:

- Using the 1-byte REF instruction to execute one 2-byte or two 1-byte instructions,
- Branching to any location by referencing a branch instruction stored in the look-up table,
- Calling subroutines at any location by referencing a call instruction stored in the look-up table.

## PROGRAMMING TIP — Using the REF Look-Up Table

Here is one example of how to use the REF instruction look-up table:

	ORG	0020H		
JMAIN KEYCK WATCH INCHL	TJP BTSF TCALL LD INCS	MAIN KEYFG CLOCK @HL,A HL	;	0, MAIN 1, KEYFG CHECK 2, Call CLOCK 3, (HL) ← A
ABC	LD ORG	EA,#00H 0080	;	47, EA ← #00H
; MAIN	NOP NOP •			
	REF REF REF REF	KEYCK JMAIN WATCH INCHL	;	BTSF KEYFG (1-byte instruction) KEYFG = 1, jump to MAIN (1-byte instruction) KEYFG = 0, CALL CLOCK (1-byte instruction) LD @HL,A INCS HL
	REF	ABC		LD EA,#00H (1-byte instruction)
	•			
	•			



## **DATA MEMORY (RAM)**

#### **OVERVIEW**

In its standard configuration, the 8,448 × 4-bit data memory has five areas:

- 32 × 4-bit working register area in bank 0
- 224 × 4-bit general-purpose area in bank 0 which is also used as the stack area
- 31 pages (00H–1EH pages) with 256 × 4-bit general-purpose area in bank 1
- 32 × 4-bit general-purpose area in bank 2
- 224 × 4-bit area for LCD data in bank 2
- 128 × 4-bit area in bank 15 for memory-mapped I/O addresses

To make it easier to reference, the data memory area has four memory banks — bank 0, bank 1, bank 2 and bank 15. The select memory bank instruction (SMB) and the page selection register (PASR) are used to select the bank and page you want to select as working data memory. Data stored in RAM locations are 1-, 4-, and 8-bit addressable.

Initialization values for the data memory area are not defined by hardware and must therefore be initialized by program software following power RESET. However, when RESET signal is generated in power-down mode, the most of data memory contents are held.

## **BANK 1 PAGE SELECTION REGISTER (PASR)**

PASR is a 5-bit write-only register for selecting the page of bank 1, and is mapped to the RAM address FD2H. It should be written by a 8-bit RAM control instruction only and the MSB 3 bits should be "0". PASR retains the previous value as long as change is not required, and the reset value is 0. Therefore, when it returns to the bank 1 from other bank (bank 0 or bank 15) without changing the contents of PASR, the previously specified bank 1 page is selected. The PASR must not be changed in the interrupt service routine because it's value cannot be recovered as the original value when the routine is finished.



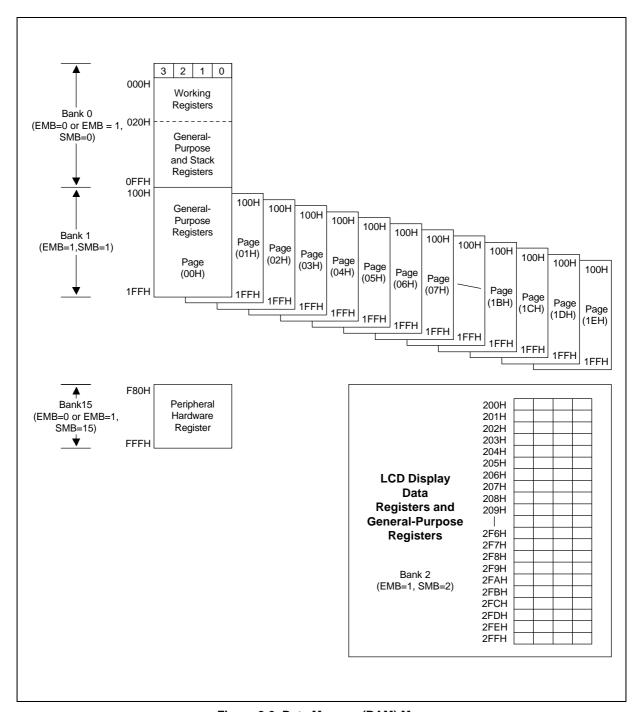


Figure 2-3. Data Memory (RAM) Map



#### Memory Banks 0, 1, 2, and 15

Bank 0	(000H-0FFH)	The lowest 32 nibbles of bank 0 (000H-01FH) are used as working registers; the next 224 nibbles (020H-0FFH) can be used both as stack area and as general-purpose data memory. Use the stack area for implementing subroutine calls and returns, and for interrupt processing.
Bank 1	(100H-1FFH)	Bank 1 has the data memory of 31 pages, the 31 pages for general-purpose data memory are comprised of $256 \times 4$ -bits, respectively. The KS57C21832 uses specially the page selection register (PASR) for selecting one of these 31 pages in bank 1.
Bank 2	(200H-2FFH)	The 224 nibbles of bank 2 are for display registers or general-purpose use; locations $2xE$ and $2xF$ ( $x=0-F$ ) are for general-purpose use in bank 2. Detailed map on bank 2 is shown in Section 12 LCD Controller/Driver.
Bank 15	(F80H-FFFH)	The microcontroller uses bank 15 for memory-mapped peripheral I/O. Fixed RAM locations for each peripheral hardware address are mapped into this area.

## **Data Memory Addressing Modes**

The enable memory bank (EMB) flag controls the addressing mode for data memory banks 0, 1, 2, or 15. When the EMB flag is logic zero, the addressable area is restricted to specific locations, depending on whether direct or indirect addressing is used. With direct addressing, you can access locations 000H-07FH of bank 0 and bank 15. With indirect addressing, only bank 0 (000H-0FFH) can be accessed. When the EMB flag is set to logic one, all four data memory banks can be accessed according to the current SMB value.

For 8-bit addressing, two 4-bit registers are addressed as a register pair. Also, when using 8-bit instructions to address RAM locations, remember to use the even-numbered register address as the instruction operand.

## **Working Registers**

The RAM working register area in data memory bank 0 is further divided into four *register* banks (bank 0, 1, 2, and 3). Each register bank has eight 4-bit registers and paired 4-bit registers are 8-bit addressable.

Register A is used as a 4-bit accumulator and register pair EA as an 8-bit extended accumulator. The carry flag bit can also be used as a 1-bit accumulator. Register pairs WX, WL, and HL are used as address pointers for indirect addressing. To limit the possibility of data corruption due to incorrect register addressing, it is advisable to use register bank 0 for the main program and banks 1, 2, and 3 for interrupt service routines.

## **LCD Data Register Area**

Bit values for LCD segment data are stored in data memory bank 2. Register locations in this area that are not used to store LCD data can be assigned to general-purpose use.



Table 2-2. Data Memory Organization and Addressing

Addresses	Register Areas	Bank	EMB Value	SMB Value
000H-01FH	Working registers	0	0, 1	0
020H-0FFH	Stack and general-purpose registers			
100H-1FFH	General-purpose registers (31 pages)	1	1	1
200H-2FFH	FFH Display registers and general-purpose registers		1	2
F80H-FFFH	I/O-mapped hardware registers	15	0, 1	15

# PROGRAMMING TIP — Clearing Data Memory Bank 0 and Page 00H in Bank 1

Clear the bank 0 and the page 00H in bank 1 of the data memory area:

RAMCLR RMCL0	SMB LD LD LD INCS JR	0 HL,#10H A,#0H @HL,A HL RMCL0	;	RAM (010H-0FFH) clear in bank 0
;	SMB LD LD	15 EA, #00H PASR, EA		
RMCL1	SMB LD LD INCS JR	1 HL,#00H @HL,A HL RMCL1	;	RAM (100H-1FFH) clear page 00H in bank 1



## **WORKING REGISTERS**

Working registers, mapped to RAM address 000H-01FH in data memory bank 0, are used to temporarily store intermediate results during program execution, as well as pointer values used for indirect addressing. Unused registers may be used as general-purpose memory. Working register data can be manipulated as 1-bit units, 4-bit units or, using paired registers, as 8-bit units.

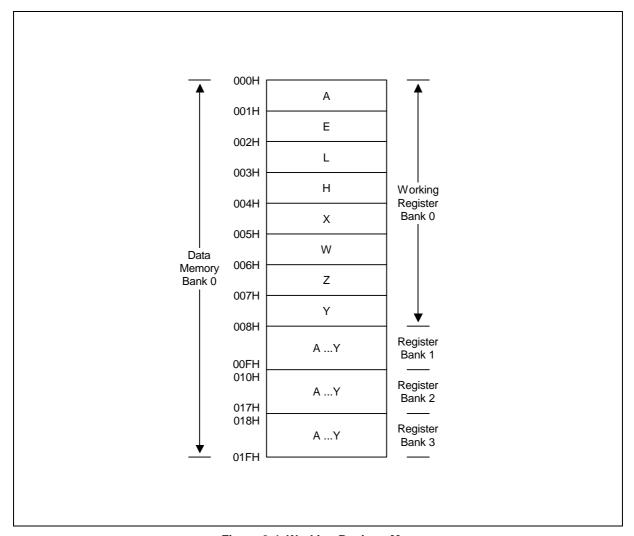


Figure 2-4. Working Register Map



## Working Register Banks

For addressing purposes, the working register area is divided into four register banks — bank 0, bank 1, bank 2, and bank 3. Any one of these banks can be selected as the working register bank by the register bank selection instruction (SRB n) and by setting the status of the register bank enable flag (ERB).

Generally, working register bank 0 is used for the main program, and banks 1, 2, and 3 for interrupt service routines. Following this convention helps to prevent possible data corruption during program execution due to contention in register bank addressing.

ERB Setting		SRB S	Selected Register Bank		
	3	2	1	0	
0	0	0	х	х	Always set to bank 0
1	0	0	0	0	Bank 0
			0	1	Bank 1
			1	0	Bank 2
			1	1	Bank 3

Table 2-3. Working Register Organization and Addressing

NOTE: 'x' means don't care.

## **Paired Working Registers**

Each of the register banks is subdivided into eight 4-bit registers. These registers, named Y, Z, W, X, H, L, E and A, can either be manipulated individually using 4-bit instructions, or together as register pairs for 8-bit data manipulation.

The names of the 8-bit register pairs in each register bank are EA, HL, WX, YZ and WL. Registers A, L, X and Z always become the lower nibble when registers are addressed as 8-bit pairs. This makes a total of eight 4-bit registers or four 8-bit double registers in each of the four working register banks.

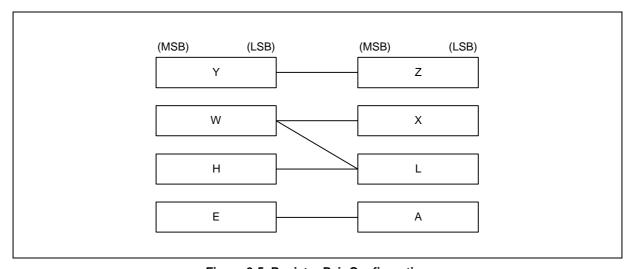


Figure 2-5. Register Pair Configuration



## **Special-Purpose Working Registers**

Register A is used as a 4-bit accumulator and double register EA as an 8-bit accumulator. The carry flag can also be used as a 1-bit accumulator.

8-bit double registers WX, WL and HL are used as data pointers for indirect addressing. When the HL register serves as a data pointer, the instructions LDI, LDD, XCHI, and XCHD can make very efficient use of working registers as program loop counters by letting you transfer a value to the L register and increment or decrement it using a single instruction.

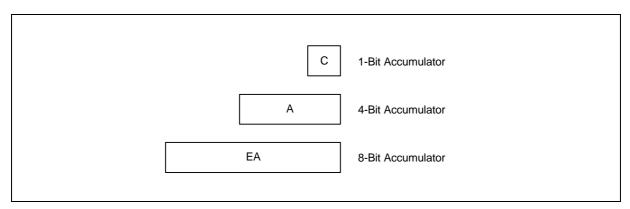


Figure 2-6. 1-Bit, 4-Bit, and 8-Bit Accumulator

## **Recommendation for Multiple Interrupt Processing**

If more than four interrupts are being processed at one time, you can avoid possible loss of working register data by using the PUSH RR instruction to save register contents to the stack before the service routines are executed in the same register bank. When the routines have executed successfully, you can restore the register contents from the stack to working memory using the POP instruction.



## PROGRAMMING TIP — Selecting the Working Register Area

The following examples show the correct programming method for selecting working register area:

1. When ERB = "0":

```
VENT2
                        1,0,INT0
                                                ; EMB \leftarrow 1, ERB \leftarrow 0, Jump to INT0 address
INT0
            PUSH
                        SB
                                                  PUSH current SMB, SRB
            SRB
                                                   Instruction does not execute because ERB = "0"
                        2
            PUSH
                        HL
                                                   PUSH HL register contents to stack
            PUSH
                        WX
                                                  PUSH WX register contents to stack
            PUSH
                        YΖ
                                                ; PUSH YZ register contents to stack
            PUSH
                        EΑ
                                                ; PUSH EA register contents to stack
            SMB
            LD
                        EA,#00H
            LD
                        80H,EA
            LD
                        HL,#40H
            INCS
                        HL
            LD
                        WX,EA
            LD
                        YZ,EA
            POP
                        EΑ
                                                ; POP EA register contents from stack
            POP
                        YΖ
                                                ; POP YZ register contents from stack
                                                ; POP WX register contents from stack
            POP
                        WX
                                                ; POP HL register contents from stack
            POP
                        HL
            POP
                        SB
                                                ; POP current SMB, SRB
            IRET
```

The POP instructions execute alternately with the PUSH instructions. If an SMB n instruction is used in an interrupt service routine, a PUSH and POP SB instruction must be used to store and restore the current SMB and SRB values, as shown in Example 2 below.

## 2. When ERB = "1":

```
VENT2
                        1,1,INT0
                                                 ; EMB \leftarrow 1, ERB \leftarrow 1, Jump to INT0 address
INT0
            PUSH
                        SB
                                                 ; Store current SMB, SRB
            SRB
                                                 ; Select register bank 2 because of ERB = "1"
                        2
            SMB
                        0
                        EA,#00H
            LD
            LD
                        80H,EA
            LD
                        HL,#40H
            INCS
                        HL
                        WX,EA
            LD
            LD
                        YZ,EA
            POP
                                                 ; Restore SMB, SRB
                        SB
            IRET
```



## STACK OPERATIONS

## **STACK POINTER (SP)**

The stack pointer (SP) is an 8-bit register that stores the address used to access the stack, an area of data memory set aside for temporary storage of data and addresses. The SP can be read or written by 8-bit control instructions. When addressing the SP, bit 0 must always remain cleared to logic zero.

F80H	SP3	SP2	SP1	"0"
F81H	SP7	SP6	SP5	SP4

There are two basic stack operations: writing to the top of the stack (push), and reading from the top of the stack (pop). A push decrements the SP and a pop increments it so that the SP always points to the top address of the last data to be written to the stack.

The program counter contents and program status word are stored in the stack area prior to the execution of a CALL or a PUSH instruction, or during interrupt service routines. Stack operation is a LIFO (Last In-First Out) type. The stack area is located in general-purpose data memory bank 0.

During an interrupt or a subroutine, the PC value and the PSW are saved to the stack area. When the routine has completed, the stack pointer is referenced to restore the PC and PSW, and the next instruction is executed. The SP can address stack registers in bank 0 (addresses 000H-0FFH) regardless of the current value of the enable memory bank (EMB) flag and the select memory bank (SMB) flag. Although general-purpose register areas can be used for stack operations, be careful to avoid data loss due to simultaneous use of the same register(s). Since the reset value of the stack pointer is not defined in firmware, we recommend that you initialize the stack pointer by program code to location 00H. This sets the first register of the stack area to 0FFH.

#### NOTE

A subroutine call occupies six nibbles in the stack; an interrupt requires six. When subroutine nesting or interrupt routines are used continuously, the stack area should be set in accordance with the maximum number of subroutine levels. To do this, estimate the number of nibbles that will be used for the subroutines or interrupts and set the stack area correspondingly.

## PROGRAMMING TIP — Initializing the Stack Pointer

To initialize the stack pointer (SP):

1. When EMB = "1":

SMB 15 ; Select memory bank 15

LD EA,#00H ; Bit 0 of SP is always cleared to "0"

LD SP,EA ; Stack area initial address (0FFH)  $\leftarrow$  (SP) - 1

2. When EMB = "0":

LD EA,#00H

LD SP,EA ; Memory addressing area (00H-7FH, F80H-FFFH)



#### **PUSH OPERATIONS**

Three kinds of push operations reference the stack pointer (SP) to write data from the source register to the stack: PUSH instructions, CALL instructions, and interrupts. In each case, the SP is *decremented* by a number determined by the type of push operation and then points to the next available stack location.

#### **PUSH Instructions**

A PUSH instruction references the SP to write two 4-bit data nibbles to the stack. Two 4-bit stack addresses are referenced by the stack pointer: one for the upper register value and another for the lower register. After the PUSH has executed, the SP is decremented by two and points to the next available stack location.

#### **CALL Instructions**

When a subroutine call is issued, the CALL instruction references the SP to write the PC's contents to six 4-bit stack locations. Current values for the enable memory bank (EMB) flag and the enable register bank (ERB) flag are also pushed to the stack. Since six 4-bit stack locations are used per CALL, you may nest subroutine calls up to the number of levels permitted in the stack.

#### **Interrupt Routines**

An interrupt routine references the SP to push the contents of the PC and the program status word (PSW) to the stack. Six 4-bit stack locations are used to store this data. After the interrupt has executed, the SP is decremented by six and points to the next available stack location. During an interrupt sequence, subroutines may be nested up to the number of levels which are permitted in the stack area.

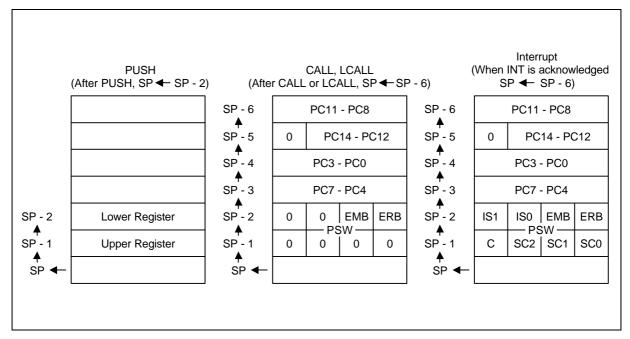


Figure 2-7. Push-Type Stack Operations



#### **POP OPERATIONS**

For each push operation there is a corresponding pop operation to write data from the stack back to the source register or registers: for the PUSH instruction it is the POP instruction; for CALL, the instruction RET or SRET; for interrupts, the instruction IRET. When a pop operation occurs, the SP is *incremented* by a number determined by the type of operation and points to the next free stack location.

## **POP Instructions**

A POP instruction references the SP to write data stored in two 4-bit stack locations back to the register pairs and SB register. The value of the lower 4-bit register is popped first, followed by the value of the upper 4-bit register. After the POP has executed, the SP is incremented *by two* and points to the next free stack location.

#### **RET and SRET Instructions**

The end of a subroutine call is signaled by the return instruction, RET or SRET. The RET or SRET uses the SP to reference the six 4-bit stack locations used for the CALL and to write this data back to the PC, the EMB, and the ERB. After the RET or SRET has executed, the SP is incremented *by six* and points to the next free stack location.

#### **IRET Instructions**

The end of an interrupt sequence is signaled by the instruction IRET. IRET references the SP to locate the six 4-bit stack addresses used for the interrupt and to write this data back to the PC and the PSW. After the IRET has executed, the SP is incremented *by six* and points to the next free stack location.

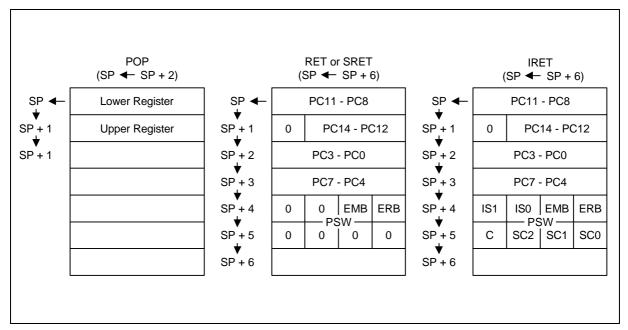


Figure 2-8. Pop-Type Stack Operations



## **BIT SEQUENTIAL CARRIER (BSC)**

The bit sequential carrier (BSC) is a 16-bit general register that can be manipulated using 1-, 4-, and 8-bit RAM control instructions. RESET clears all BSC bit values to logic zero.

Using the BSC, you can specify sequential addresses and bit locations using 1-bit indirect addressing (memb.@L). (Bit addressing is independent of the current EMB value.) In this way, programs can process 16-bit data by moving the bit location sequentially and then incrementing or decrementing the value of the L register.

BSC data can also be manipulated using direct addressing. For 8-bit manipulations, the 4-bit register names BSC0 and BSC2 must be specified and the upper and lower 8 bits manipulated separately.

If the values of the L register are 0H at BSC0.@L, the address and bit location assignment is FC0H.0. If the L register content is FH at BSC0.@L, the address and bit location assignment is FC3H.3.

Name	Address	Bit 3	Bit 2	Bit 1	Bit 0
BSC0	FC0H	BSC0.3	BSC0.2	BSC0.1	BSC0.0
BSC1	FC1H	BSC1.3	BSC1.2	BSC1.1	BSC1.0
BSC2	FC2H	BSC2.3	BSC2.2	BSC2.1	BSC2.0
BSC3	FC3H	BSC3.3	BSC3.2	BSC3.1	BSC3.0

Table 2-4. BSC Register Organization

# PROGRAMMING TIP — Using the BSC Register to Output 16-Bit Data

To use the bit sequential carrier (BSC) register to output 16-bit data (5937H) to the P3.0 pin:

	BITS SMB LD LD LD LD SMB LD	EMB 15 EA,#37H BSC0,EA EA,#59H BSC2,EA 0 L,#0H	; ; BSC0 ← A, BSC1 ← E ; ; BSC2 ← A, BSC3 ← E ;
AGN	LDB LDB INCS JR RET	C,BSC0.@L P3.0,C L AGN	; ; P3.0 ← C



## **PROGRAM COUNTER (PC)**

A 14-bit program counter (PC) stores addresses for instruction fetches during program execution. Whenever a reset operation or an interrupt occurs, bits PC13 through PC0 are set to the vector address.

Usually, the PC is incremented by the number of bytes of the instruction being fetched. One exception is the 1-byte REF instruction which is used to reference instructions stored in the ROM.

## **PROGRAM STATUS WORD (PSW)**

The program status word (PSW) is an 8-bit word that defines system status and program execution status and which permits an interrupted process to resume operation after an interrupt request has been serviced. PSW values are mapped as follows:

	(MSB)			(LSB)
FB0H	IS1	IS0	EMB	ERB
FB1H	С	SC2	SC1	SC0

The PSW can be manipulated by 1-bit or 4-bit read/write and by 8-bit read instructions, depending on the specific bit or bits being addressed. The PSW can be addressed during program execution regardless of the current value of the enable memory bank (EMB) flag.

Part or all of the PSW is saved to stack prior to execution of a subroutine call or hardware interrupt. After the interrupt has been processed, the PSW values are popped from the stack back to the PSW address. When a RESET is generated, the EMB and ERB values are set according to the RESET vector address, and the carry flag is left undefined (or the current value is retained). PSW bits ISO, IS1, SCO, SC1, and SC2 are all cleared to logical zero.

Table 2-5. Program Status Word Bit Descriptions

PSW Bit Identifier	Description	Bit Addressing	Read/Write
IS1, IS0	Interrupt status flags	1, 4	R/W
EMB	Enable memory bank flag	1	R/W
ERB	Enable register bank flag	1	R/W
С	Carry flag	1	R/W
SC2, SC1, SC0	Program skip flags	8	R



## **INTERRUPT STATUS FLAGS (ISO, IS1)**

PSW bits ISO and IS1 contain the current interrupt execution status values. You can manipulate ISO and IS1 flags directly using 1-bit RAM control instructions.

By manipulating interrupt status flags in conjunction with the interrupt priority register (IPR), you can process multiple interrupts by anticipating the next interrupt in an execution sequence. The interrupt priority control circuit determines the ISO and IS1 settings in order to control multiple interrupt processing. When both interrupt status flags are set to "0", all interrupts are allowed. The priority with which interrupts are processed is then determined by the IPR.

When an interrupt occurs, ISO and IS1 are pushed to the stack as part of the PSW and are automatically incremented to the next higher priority level. Then, when the interrupt service routine ends with an IRET instruction, ISO and IS1 values are restored to the PSW. Table 2-6 shows the effects of ISO and IS1 flag settings.

IS1 Value	IS0 Value	Status of Currently Executing Process	Effect of IS0 and IS1 Settings on Interrupt Request Control
0	0	0	All interrupt requests are serviced.
0	1	1	Only high-priority interrupt(s) as determined in the interrupt priority register (IPR) are serviced.
1	0	2	No more interrupt requests are serviced.
1	1	_	Not applicable; these bit settings are undefined.

Table 2-6. Interrupt Status Flag Bit Settings

Since interrupt status flags can be addressed by write instructions, programs can exert direct control over interrupt processing status. Before interrupt status flags can be addressed, however, you must first execute a DI instruction to inhibit additional interrupt routines. When the bit manipulation has been completed, execute an EI instruction to re-enable interrupt processing.

## PROGRAMMING TIP — Setting ISx Flags for Interrupt Processing

The following instruction sequence shows how to use the ISO and IS1 flags to control interrupt processing:

BITS ISO ; Allow interrupts according to IPR priority level

El ; Enable interrupt



## **EMB FLAG (EMB)**

The EMB flag is used to allocate specific address locations in the RAM by modifying the upper 4 bits of 12-bit data memory addresses. In this way, it controls the addressing mode for data memory banks 0, 1, 2, or 15.

When the EMB flag is "0", the data memory address space is restricted to bank 15 and addresses 000H-07FH of memory bank 0, regardless of the SMB register contents.

When the EMB flag is set to "1", the general-purpose areas of bank 0, 1, 2, and 15 can be accessed by using the appropriate SMB value. If it is selected the general purpose area of bank 1 using SMB 1, a page in bank 1 can be selected by means of the page selection register (PASR).

## PROGRAMMING TIP — Using the EMB Flag to Select Memory Banks

EMB flag settings for memory bank selection:

1. When EMB = "0":

```
SMB
            15
LD
            EA,#00H
LD
            PASR,EA
SMB
                                     ; Non-essential instruction since EMB = "0"
LD
            A,#9H
LD
            90H,A
                                     ; (F90H) \leftarrow A, bank 15 is selected
                                        (034H) ← A, bank 0 is selected
LD
            34H,A
SMB
                                     ; Non-essential instruction since EMB = "0"
            O
LD
                                        (F90H) ← A, bank 15 is selected
            90H,A
            34H,A
LD
                                        (034H) ← A, bank 0 is selected
                                     ; Non-essential instruction, since EMB = "0"
SMB
            15
LD
            20H.A
                                     ; (020H) ← A, bank 0 is selected
LD
            90H,A
                                     ; (F90H) ← A, bank 15 is selected
```

#### 2. When EMB = "1":

SMB	15	
LD	EA,#00H	
LD	PASR,EA	
SMB	1	; Select memory bank 1
LD	A,#9H	·
LD	90H,A	; (190H) $\leftarrow$ A, page 00H in bank 1 is selected
LD	34H,A	; (134H) ← A, page 00H in bank 1 is selected
SMB	0	; Select memory bank 0
LD	90H,A	; (090H) ← A, bank 0 is selected
LD	34H,A	; (034H) ← A, bank 0 is selected
SMB	15	; Select memory bank 15
LD	20H,A	; Program error, but assembler does not detect i
LD	90H,A	; (F90H) ← A, bank 15 is selected



## **ERB FLAG (ERB)**

The 1-bit register bank enable flag (ERB) determines the range of addressable working register area. When the ERB flag is "1", the working register area from register banks 0 to 3 is selected according to the register bank selection register (SRB). When the ERB flag is "0", register bank 0 is the selected working register area, regardless of the current value of the register bank selection register (SRB).

When an internal RESET is generated, bit 6 of program memory address 0000H is written to the ERB flag. This automatically initializes the flag. When a vectored interrupt is generated, bit 6 of the respective address table in program memory is written to the ERB flag, setting the correct flag status before the interrupt service routine is executed.

During the interrupt routine, the ERB value is automatically pushed to the stack area along with the other PSW bits. Afterwards, it is popped back to the FB0H.0 bit location. The initial ERB flag settings for each vectored interrupt are defined using VENTn instructions.

# PROGRAMMING TIP — Using the ERB Flag to Select Register Banks

ERB flag settings for register bank selection:

1. When ERB = "0":

```
SRB
            1
                                      ; Register bank 0 is selected (since ERB = "0", the
                                        SRB is configured to bank 0)
ΙD
            EA,#34H
                                      ; Bank 0 EA ← #34H
                                      ; Bank 0 HL \leftarrow EA
ΙD
            HL,EA
                                      ; Register bank 0 is selected
SRB
            2
                                      ; Bank 0 YZ \leftarrow EA
LD
            YZ,EA
                                      ; Register bank 0 is selected
SRB
            3
                                      ; Bank 0 WX ← EA
LD
            WX,EA
```

2. When ERB = "1":

```
SRB
                                   Register bank 1 is selected
LD
           EA.#34H
                                    Bank 1 EA ← #34H
LD
                                     Bank 1 HL ← Bank 1 EA
           HL.EA
SRB
           2
                                    Register bank 2 is selected
LD
           YZ,EA
                                     Bank 2 YZ ← BANK2 EA
SRB
                                    Register bank 3 is selected
           3
LD
           WX,EA
                                  ; Bank 3 WX ← Bank 3 EA
```



## SKIP CONDITION FLAGS (SC2, SC1, SC0)

The skip condition flags SC2, SC1, and SC0 in the PSW indicate the current program skip conditions and are set and reset automatically during program execution. Skip condition flags can only be addressed by 8-bit read instructions. Direct manipulation of the SC2, SC1, and SC0 bits is not allowed.

## **CARRY FLAG (C)**

The carry flag is used to save the result of an overflow or borrow when executing arithmetic instructions involving a carry (ADC, SBC). The carry flag can also be used as a 1-bit accumulator for performing Boolean operations involving bit-addressed data memory.

If an overflow or borrow condition occurs when executing arithmetic instructions with carry (ADC, SBC), the carry flag is set to "1". Otherwise, its value is "0". When a RESET occurs, the current value of the carry flag is retained during power-down mode, but when normal operating mode resumes, its value is undefined.

The carry flag can be directly manipulated by predefined set of 1-bit read/write instructions, independent of other bits in the PSW. Only the ADC and SBC instructions, and the instructions listed in Table 2-7, affect the carry flag.

Operation Type	Instructions	Carry Flag Manipulation
Direct manipulation	SCF	Set carry flag to "1".
	RCF	Clear carry flag to "0" (reset carry flag).
	CCF	Invert carry flag value (complement carry flag).
	BTST C	Test carry and skip if C = "1".
Bit transfer	LDB (operand) (1),C	Load carry flag value to the specified bit.
	LDB C,(operand) (1)	Load contents of the specified bit to carry flag.
Boolean manipulation	BAND C,(operand) (1)	AND the specified bit with contents of carry flag and save the result to the carry flag.
	BOR C,(operand) (1)	OR the specified bit with contents of carry flag and save the result to the carry flag.
	BXOR C,(operand) (1)	XOR the specified bit with contents of carry flag and save the result to the carry flag.
Interrupt routine	INTn (2)	Save carry flag to stack with other PSW bits.
Return from interrupt	IRET	Restore carry flag from stack with other PSW bits.

**Table 2-7. Valid Carry Flag Manipulation Instructions** 

## NOTES:

- 1. The operand has three bit addressing formats: mema.a, memb.@L, and @H + DA.b.
- 2. 'INTn' refers to the specific interrupt being executed and is not an instruction.



# PROGRAMMING TIP — Using the Carry Flag as a 1-Bit Accumulator

1. Set the carry flag to logic one:

SCF ; C ← 1

EA,#0C3H HL,#0AAH ; EA ← #0C3H LD ; HL ← #0AAH LD

ADC EA,HL ; EA  $\leftarrow$  #0C3H + #0AAH + #1H, C  $\leftarrow$  1

2. Logical-AND bit 3 of address 3FH with P3.3 and output the result to P5.0:

LD ; Set the upper four bits of the address to the H register H,#3H

; value

C,@H+0FH.3 LDB  $; \quad C \leftarrow \ \, \text{bit 3 of 3FH}$ BAND C,P3.3 ;  $C \leftarrow C$  AND P3.3

LDB P5.0,C ; Output result from carry flag to P5.0 ADDRESS SPACES KS57C21832/P21832

## **NOTES**



KS57C21832/P21832 ADDRESSING MODES

3

## **ADDRESSING MODES**

#### **OVERVIEW**

The enable memory bank flag, EMB, controls the two addressing modes for data memory. When the EMB flag is set to logic one, you can address the entire RAM area; when the EMB flag is cleared to logic zero, the addressable area in the RAM is restricted to specific locations.

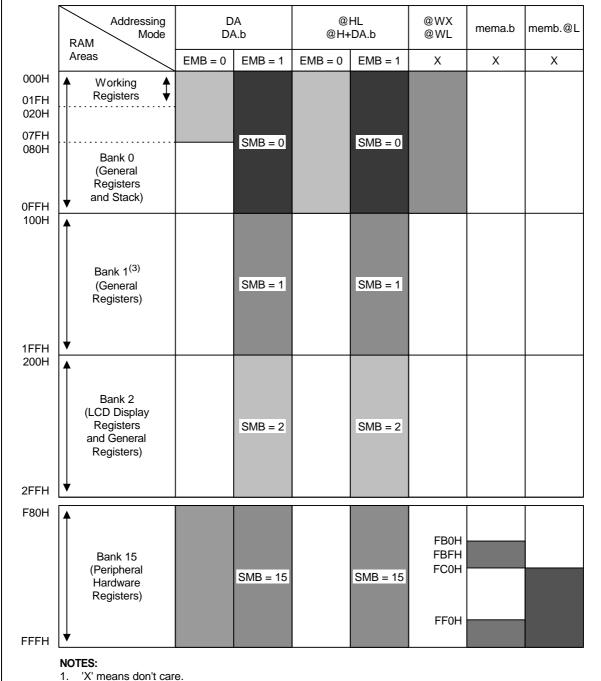
The EMB flag works in connection with the select memory bank instruction, SMBn. You will recall that the SMBn instruction is used to select RAM bank 0, 1, 2, or 15. The SMB setting is always contained in the upper four bits of a 12-bit RAM address. For this reason, both addressing modes (EMB = "0" and EMB = "1") apply specifically to the memory bank indicated by the SMB instruction, and any restrictions to the addressable area within banks 0, 1, 2, or 15. Direct and indirect 1-bit, 4-bit, and 8-bit addressing methods can be used. Several RAM locations are addressable at all times, regardless of the current EMB flag setting.

Here are a few guidelines to keep in mind regarding data memory addressing:

- When you address peripheral hardware locations in bank 15, the mnemonic for the memory-mapped hardware component can be used as the operand in place of the actual address location.
- Always use an even-numbered RAM address as the operand in 8-bit direct and indirect addressing.
- With direct addressing, use the RAM address as the instruction operand; with indirect addressing, the instruction specifies a register which contains the operand's address.



**ADDRESSING MODES** KS57C21832/P21832



- Blank columns indicate RAM areas that are not addressable, given the addressing method and enable memory bank (EMB) flag setting shown in the column headers.
- The bank 1 has 31 pages (00H-1EH), and a page in bank 1 can be selected by means of the page selection register (PASR).

Figure 3-1. RAM Address Structure



KS57C21832/P21832 ADDRESSING MODES

#### **EMB AND ERB INITIALIZATION VALUES**

The EMB and ERB flag bits are set automatically by the values of the RESET vector address and the interrupt vector address. When a RESET is generated internally, bit 7 of program memory address 0000H is written to the EMB flag, initializing it automatically. When a vectored interrupt is generated, bit 7 of the respective vector address table is written to the EMB. This automatically sets the EMB flag status for the interrupt service routine. When the interrupt is serviced, the EMB value is automatically saved to stack and then restored when the interrupt routine has completed.

At the beginning of a program, the initial EMB and ERB flag values for each vectored interrupt must be set by using VENT instruction. The EMB and ERB can be set or reset by bit manipulation instructions (BITS, BITR) despite the current SMB setting.

#### PROGRAMMING TIP — Initializing the EMB and ERB Flags

The following assembly instructions show how to initialize the EMB and ERB flag settings:

```
ORG
           H0000
                         ; ROM address assignment
VENT0
           1,0,RESET ; EMB \leftarrow 1, ERB \leftarrow 0; Jump to RESET address by RESET
VENT1
           0,1,INTB
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INTB address by INTB
VENT2
           0,1,INT0
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INT0 address by INT0
VENT3
           0,1,INT1
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INT1 address by INT1
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INTS address by INTS
VENT4
           0,1,INTS
VENT5
           0,1,INTT0
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INTT0 address by INTT0
VENT6
           0,1,INTT1
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INTT1 address by INTT1
VENT7
           0,1,INTK
                         ; EMB \leftarrow 0, ERB \leftarrow 1; Jump to INTK address by INTK
```

RESET BITR EMB



ADDRESSING MODES KS57C21832/P21832

#### **ENABLE MEMORY BANK SETTINGS**

#### **EMB** = "1"

When the enable memory bank flag EMB is set to logic one, you can address the data memory bank specified by the select memory bank (SMB) value (0, 1, 2, or 15) using 1-, 4-, or 8-bit instructions. You can use both direct and indirect addressing modes. The addressable RAM areas when EMB = "1" are as follows:

If SMB = 0, 000H-0FFH (bank 0 is selected)

If SMB = 1, 100H-1FFH (bank 1 is selected: bank 1 has the data memory of 31 pages (00H-1EH))

If SMB = 2, 200H-2FFH (bank 2 is selected)

If SMB = 15, F80H-FFFH (bank 15 is selected)

#### EMB = "0"

When the enable memory bank flag EMB is set to logic zero, the addressable area is defined independently of the SMB value, and is restricted to specific locations depending on whether a direct or indirect address mode is used.

If EMB = "0", the addressable area is restricted to locations 000H-07FH in bank 0 and to locations F80H-FFFH in bank 15 for direct addressing. For indirect addressing, only locations 000H-0FFH in bank 0 are addressable, regardless of SMB value.

To address the peripheral hardware register (bank 15) using indirect addressing, the EMB flag must first be set to "1" and the SMB value to "15". When a RESET occurs, the EMB flag is set to the value contained in bit 7 of ROM address 0000H.

#### **EMB-Independent Addressing**

At any time, several areas of the data memory can be addressed independent of the current status of the EMB flag. These exceptions are described in Table 3-1.

Table 3-1. RAM Addressing Not Affected by the EMB Value

Address	Addressing Method	Affected Hardware	Program Examples
000H-0FFH	4-bit indirect addressing using WX and WL register pairs; 8-bit indirect addressing using SP	Not applicable	LD A,@WX PUSH POP
FB0H-FBFH FF0H-FFFH	1-bit direct addressing	PSW, SCMOD, IEx, IRQx, I/O	BITS EMB BITR IE4
FC0H-FFFH	1-bit indirect addressing using the L register	BSC, I/O	BTST FC3H.@L BAND C,P3.@L



KS57C21832/P21832 ADDRESSING MODES

#### **SELECT BANK REGISTER (SB)**

The select bank register (SB) is used to assign the memory bank and register bank. The 8-bit SB register consists of the 4-bit select register bank register (SRB) and the 4-bit select memory bank register (SMB), as shown in Figure 3-2.

During interrupts and subroutine calls, SB register contents can be saved to stack in 8-bit units by the PUSH SB instruction. You later restore the value to the SB using the POP SB instruction.

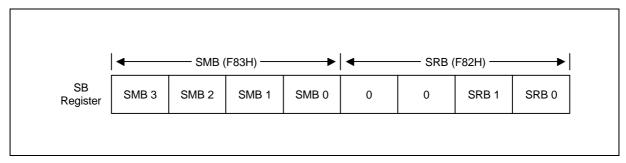


Figure 3-2. SMB and SRB Values in the SB Register

#### **SELECT REGISTER BANK (SRB) INSTRUCTION**

The select register bank (SRB) value specifies which register bank is to be used as a working register bank. The SRB value is set by the 'SRB n' instruction, where n = 0, 1, 2, 3.

One of the four register banks is selected by the combination of ERB flag status and the SRB value that is set using the 'SRB n' instruction. The current SRB value is retained until another register is requested by program software. PUSH SB and POP SB instructions are used to save and restore the contents of SRB during interrupts and subroutine calls. RESET clears the 4-bit SRB value to logic zero.

#### **SELECT MEMORY BANK (SMB) INSTRUCTION**

To select one of the four available data memory banks, you must execute an SMB n instruction specifying the number of the memory bank you want (0, 1, 2, or 15). For example, the instruction 'SMB 1' selects bank 1 and 'SMB 15' selects bank 15. (And remember to enable the selected memory bank by making the appropriate EMB flag setting).

The upper four bits of the 12-bit data memory address are stored in the SMB register. If the SMB value is not specified by software (or if a RESET does not occur) the current value is retained. RESET clears the 4-bit SMB value to logic zero.

The PUSH SB and POP SB instructions save and restore the contents of the SMB register to and from the stack area during interrupts and subroutine calls.



ADDRESSING MODES KS57C21832/P21832

#### **DIRECT AND INDIRECT ADDRESSING**

1-bit, 4-bit, and 8-bit data stored in data memory locations can be addressed directly using a specific register or bit address as the instruction operand.

Indirect addressing specifies a memory location that contains the required direct address. The KS57 instruction set supports 1-bit, 4-bit, and 8-bit indirect addressing. For 8-bit indirect addressing, an even-numbered RAM address must always be used as the instruction operand.

#### 1-BIT ADDRESSING

Table 3-2. 1-Bit Direct and Indirect RAM Addressing

Operand Notation	Addressing Mode Description	EMB Flag Setting	Addressable Area	Memory Bank	Hardware I/O Mapping
DA.b	Direct: bit is indicated by the	0	000H-07FH	Bank 0	_
	RAM address (DA), memory bank selection, and specified bit number (b).		F80H-FFFH	Bank 15	All 1-bit addressable peripherals (SMB = 15)
		1	000H-FFFH	SMB = 0, 1, 2, 15	
mema.b	Direct: bit is indicated by addressable area (mema) and bit number (b).	х	FB0H-FBFH FF0H-FFFH	Bank 15	IS0, IS1, EMB, ERB, IEx, IRQx, Pn.n
memb.@L	Indirect: address is indicated by the upper 6 bits of RAM area (memb) and the upper 2 bits of register L, and bit is indicated by the lower 2 bits of register L.	x	FC0H-FFFH	Bank 15	BSCn.x Pn.n
@H + DA.b	Indirect: bit is indicated by the lower 4 bits of the address (DA), memory bank selection, and the H register identifier.	0	000H-0FFH	Bank 0	_
		1	000H-FFFH	SMB = 0, 1, 2, 15	All 1-bit addressable peripherals (SMB = 15)

NOTE: 'x' means don't care.



KS57C21832/P21832 ADDRESSING MODES

## PROGRAMMING TIP — 1-Bit Addressing Modes

#### 1-Bit Direct Addressing

```
1. If EMB = "0":
AFLAG
           EQU
                      34H.3
BFLAG
           EQU
                      85H.3
CFLAG
           EQU
                      0BAH.0
           SMB
           BITS
                      AFLAG
                                           ; 34H.3 ← 1
                                           ; F85H.3 ← 1
           BITS
                      BFLAG
                                           ; If FBAH.0 = 1, skip
           BTST
                      CFLAG
                                           ; Else if, FBAH.0 = 0, F85H.3 (BMOD.3) \leftarrow 1
                      BFLAG
           BITS
                                            ; FF3H.0 (P3.0) ← 1
           BITS
                      P3.0
2. If EMB = "1":
                      34H.3
AFLAG
           EQU
BFLAG
                      85H.3
           EQU
CFLAG
           EQU
                      0BAH.0
           SMB
                      0
                                           ; 34H.3 ← 1
           BITS
                      AFLAG
                                            ; 85H.3 ← 1
           BITS
                      BFLAG
                                            ; If 0BAH.0 = 1, skip
                      CFLAG
           BTST
                                            ; Else if 0BAH.0 = 0, 085H.3 \leftarrow 1
           BITS
                      BFLAG
           BITS
                      P3.0
                                            ; FF3H.0 (P3.0) ← 1
```

ADDRESSING MODES KS57C21832/P21832

#### PROGRAMMING TIP — 1-Bit Addressing Modes (Continued)

#### 1-Bit Indirect Addressing

```
1. If EMB = "0":
AFLAG
           EQU
                       34H.3
BFLAG
           EQU
                       85H.3
CFLAG
           EQU
                       0BAH.0
           SMB
                       0
           LD
                       H,#0BH
                                              ; H ← #0BH
                                              ; If 0BAH.0 = 1, 0BAH.0 \leftarrow 0 and skip
           BTSTZ
                       @H+CFLAG
                                              ; Else if 0BAH.0 = 0, FBAH.0 \leftarrow 1
           BITS
                       CFLAG
2. If EMB = "1":
AFLAG
           EQU
                       34H.3
BFLAG
                       85H.3
           EQU
                       0BAH.0
CFLAG
           EQU
           SMB
                       0
           LD
                       H,#0BH
                                              ; H ← #0BH
           BTSTZ
                       @H+CFLAG
                                              ; If 0BAH.0 = 1, 0BAH.0 \leftarrow 0 and skip
           BITS
                       CFLAG
                                              ; Else if 0BAH.0 = 0, 0BAH.0 \leftarrow 1
```



KS57C21832/P21832 ADDRESSING MODES

#### **4-BIT ADDRESSING**

Table 3-3. 4-Bit Direct and Indirect RAM Addressing

Operand Notation	Addressing Mode Description	EMB Flag Setting	Addressable Area	Memory Bank	Hardware I/O Mapping
DA	Direct: 4-bit address indicated	0	000H-07FH	Bank 0	_
	by the RAM address (DA) and the memory bank selection		F80H-FFFH	Bank 15	All 4-bit addressable peripherals
		1	000H-FFFH	SMB = 0, 1, 2, 15	(SMB = 15)
@HL	Indirect: 4-bit address indicated by the memory bank selection and register HL	0	000H-0FFH	Bank 0	_
		1	000H-FFFH	SMB = 0, 1, 2, 15	All 4-bit addressable peripherals (SMB = 15)
@WX	Indirect: 4-bit address indicated by register WX	х	000H-0FFH	Bank 0	-
@WL	Indirect: 4-bit address indicated by register WL	Х	000H-0FFH	Bank 0	

NOTE: 'x' means don't care.

## PROGRAMMING TIP — 4-Bit Addressing Modes

#### 4-Bit Direct Addressing

1. If EMB = "0":

**ADATA** EQU 46H **BDATA** EQU 8EH ; Non-essential instruction, since EMB = "0" SMB 15 ; A ← (P3) LD A,P3 ; Non-essential instruction, since EMB = "0" SMB ADATA,A LD (046H) ← A LD ;  $(F8EH (LCON)) \leftarrow A$ BDATA,A

2. If EMB = "1":

EQU 46H **ADATA BDATA** EQU 8EH SMB 15 LD A,P3 ; A ← (P3) SMB 0 ADATA,A LD ; (046H) ← A LD BDATA,A ; (08EH)  $\leftarrow$  A



ADDRESSING MODES KS57C21832/P21832

#### PROGRAMMING TIP — 4-Bit Addressing Modes (Continued)

#### 4-Bit Indirect Addressing (Example 1)

1. If EMB = "0", compare bank 0 locations 040H-046H with bank 0 locations 060H-066H:

```
ADATA
            EQU
                         46H
BDATA
            EQU
                         66H
            SMB
                                                  ; Non-essential instruction, since EMB = "0"
                         1
            LD
                         HL,#BDATA
            LD
                         WX,#ADATA
                                                  ; A \leftarrow bank 0 (040H-046H) ; If bank 0 (060H-066H) = A, skip
COMP
                         A,@WL
            LD
            CPSE
                         A,@HL
            SRET
            DECS
                         COMP
            JR
            RET
```

2. If EMB = "1", compare bank 0 locations 040H-046H to bank 1 locations 160H-166H:

```
ADATA
          EQU
                     46H
BDATA
          EQU
                     66H
          SMB
                     HL,#BDATA
          LD
          LD
                     WX,#ADATA
COMP
                     A,@WL
                                          ; A \leftarrow bank 0 (040H-046H)
          LD
          CPSE
                                           ; If bank 1 (160H-166H) = A, skip
                     A,@HL
          SRET
          DECS
           JR
                     COMP
          RET
```

KS57C21832/P21832 ADDRESSING MODES

#### PROGRAMMING TIP — 4-Bit Addressing Modes (Concluded)

#### 4-Bit Indirect Addressing (Example 2)

1. If EMB = "0", exchange bank 0 locations 040H-046H with bank 0 locations 060H-066H:

ADATA EQU 46H BDATA EQU 66H

SMB 1 ; Non-essential instruction, since EMB = "0"

LD HL,#BDATA LD WX,#ADATA

TRANS LD A,@WL ; A  $\leftarrow$  bank 0 (040H-046H)

XCHD A,@HL ; Bank 0 (060H-066H)  $\leftrightarrow$  A

JR TRANS

2. If EMB = "1", exchange bank 0 locations 040H-046H to bank 1 locations 160H-166H:

ADATA EQU 46H BDATA EQU 66H

SMB 1

LD HL,#BDATA LD WX,#ADATA

TRANS LD A,@WL ; A  $\leftarrow$  bank 0 (040H-046H)

XCHD A,@HL ; Bank 1 (160H-166H)  $\leftrightarrow$  A

JR TRANS

ADDRESSING MODES KS57C21832/P21832

#### **8-BIT ADDRESSING**

Table 3-4. 8-Bit Direct and Indirect RAM Addressing

Instruction Notation	Addressing Mode Description	EMB Flag Setting	Addressable Area	Memory Bank	Hardware I/O Mapping
DA	Direct: 8-bit address indicated	0	000H-07FH	Bank 0	_
	by the RAM address ( <i>DA</i> = even number) and memory bank selection		F80H-FFFH	Bank 15	All 8-bit addressable peripherals
		1	000H-FFFH	SMB = 0, 1, 2, 15	(SMB = 15)
@HL	Indirect: the 8-bit address indicated by the memory bank selection and register HL; (the 4-bit L register value must be an even number)	0	000H-0FFH	Bank 0	-
		1	000H-FFFH	SMB = 0, 1, 2, 15	All 8-bit addressable peripherals (SMB = 15)

## PROGRAMMING TIP — 8-Bit Addressing Modes

#### 8-Bit Direct Addressing

1. If EMB = "0":

ADATA EQU 46H
BDATA EQU 8EH
SMB 15
LD EA,P4
SMB 0

; Non-essential instruction, since EMB = "0"

;  $E \leftarrow (P5), A \leftarrow (P4)$ 

2. If EMB = "1":

ADATA EQU 46H BDATA EQU 8EH SMB 15 LD EA,P4

LD ADATA, EA ;  $(046H) \leftarrow A$ ,  $(047H) \leftarrow E$ LD BDATA, EA ;  $(08EH) \leftarrow A$ ,  $(08FH) \leftarrow E$ 



KS57C21832/P21832 ADDRESSING MODES

#### PROGRAMMING TIP — 8-Bit Addressing Modes (Continued)

#### 8-Bit Indirect Addressing

1. If EMB = "0":

ADATA EQU 46H

SMB 1 ; Non-essential instruction, since EMB = "0"

LD HL,#ADATA

LD EA,@HL ; A  $\leftarrow$  (046H), E  $\leftarrow$  (047H)

2. If EMB = "1":

ADATA EQU 46H

SMB 1

LD HL,#ADATA

LD EA,@HL ; A  $\leftarrow$  (146H), E  $\leftarrow$  (147H)

ADDRESSING MODES KS57C21832/P21832

## **NOTES**



# 4 MEMORY MAP

#### **OVERVIEW**

To support program control of peripheral hardware, I/O addresses for peripherals are memory-mapped to bank 15 of the RAM. Memory mapping lets you use a mnemonic as the operand of an instruction in place of the specific memory location.

Access to bank 15 is controlled by the select memory bank (SMB) instruction and by the enable memory bank flag (EMB) setting. If the EMB flag is "0", bank 15 can be addressed using direct addressing, regardless of the current SMB value. 1-bit direct and indirect addressing can be used for specific locations in bank 15, regardless of the current EMB value.

#### I/O MAP FOR HARDWARE REGISTERS

Table 4-1 contains detailed information about I/O mapping for peripheral hardware in bank 15 (register locations F80H-FFFH). Use the I/O map as a quick-reference source when writing application programs. The I/O map gives you the following information:

- Register address
- Register name (mnemonic for program addressing)
- Bit values (both addressable and non-manipulable)
- Read-only, write-only, or read and write addressability
- 1-bit, 4-bit, or 8-bit data manipulation characteristics



Table 4-1. I/O Map for Memory Bank 15

		Memory	Bank 15				Add	dressing N	lode
Address	Register	Bit 3	Bit 2	Bit 1	Bit 0	R/W	1-Bit	4-Bit	8-Bit
F80H	SP	.3	.2	.1	"0"	R/W	No	No	Yes
F81H		.7	.6	.5	.4				
		Loc	cations F82	2H-F84H ar	e not map	ped.			
F85H	BMOD	.3	.2	.1	.0	W	.3	Yes	No
F86H	BCNT					R	No	No	Yes
F87H									
F88H	WMOD	.3	.2	.1	.0	W	.3 (1)	No	Yes
F89H		.7	"0"	.5	.4				
		Loc	ations F8A	H-F8BH a	re not map	ped.			
F8CH	LMOD	.3	.2	.1	.0	W	No	No	Yes
F8DH		.7	.6	.5	.4				
F8EH	LCON	"0"	.2	.1	.0	W	No	Yes	No
			Locations	F8FH is no	ot mapped	i.			
F90H	TMOD0	.3	.2	"0"	"0"	W	.3	No	Yes
F91H		"0"	.6	.5	.4				
F92H		TOE1	TOE0	"U" <sup>(4)</sup>	"0"	R/W	Yes	Yes	No
			Locations	F93H is no	ot mapped			1	1
F94H	TCNT0					R	No	No	Yes
F95H									
F96H	TREF0					W	No	No	Yes
F97H									
F98H	WDMOD	.3	.2	.1	.0	W	No	No	Yes
F99H		.7	.6	.5	.4				
F9AH	WDFLAG (2)	WDTCF	"0"	"0"	"0"	W	.3	Yes	No
	•	Loc	ations F9E	BH-F9FH a	re not map	ped.			
FA0H	TMOD1	.3	.2	"0"	"0"	W	.3	No	Yes
FA1H		"0"	.6	.5	.4				
		Loc	ations FA2	2H-FA3H a	re not map	ped.			
FA4H	TCNT1A					R	No	No	Yes
FA5H									
FA6H	TCNT1B								
FA7H									



Table 4-1. I/O Map for Memory Bank 15 (Continued)

		Memory	Bank 15				Add	dressing N	lode
Address	Register	Bit 3	Bit 2	Bit 1	Bit 0	R/W	1-Bit	4-Bit	8-Bit
FA8H	TREF1A					W	No	No	Yes
FA9H									
FAAH	TREF1B								
FABH									
,		Loc	cations FAC	CH-FAFH a	are not map	ped.			
FB0H	PSW	IS1	IS0	EMB	ERB	R/W	Yes	Yes	Yes
FB1H		C (3)	SC2	SC1	SC0	R	No	No	
FB2H	IPR	IME	.2	.1	.0	W	IME	Yes	No
FB3H	PCON	.3	.2	.1	.0	W	No	Yes	No
FB4H	IMOD0	"0"	"0"	.1	.0	W	No	Yes	No
FB5H	IMOD1	"0"	"0"	"0"	.0	W	No	Yes	No
FB6H	IMODK	"0"	.2	.1	.0	W	No	Yes	No
FB7H	SCMOD	.3	.2	"0"	.0	W	Yes	No	No
FB8H		IE4	IRQ4	IEB	IRQB	R/W	Yes	Yes	No
			Locations	FB9H is n	ot mapped				
FBAH		"0"	"0"	IEW	IRQW	R/W	Yes	Yes	No
FBBH		IEK	IRQK	IET1	IRQT1				
FBCH		"0"	"0"	IET0	IRQT0				
FBDH		"0"	"0"	IES	IRQS				
FBEH		IE1	IRQ1	IE0	IRQ0				
FBFH		"0"	"0"	IE2	IRQ2				
FC0H	BSC0					R/W	Yes	Yes	Yes
FC1H	BSC1								
FC2H	BSC2								
FC3H	BSC3								
		Loc	cations FC	1H-FCFH a	are not map	ped.			



Table 4-1. I/O Map for Memory Bank 15 (Continued)

		Memory	Bank 15				Add	dressing N	lode
Address	Register	Bit 3	Bit 2	Bit 1	Bit 0	R/W	1-Bit	4-Bit	8-Bit
FD0H	CLMOD	.3	"0"	.1	.0	W	No	Yes	No
			Location	FD1H is no	ot mapped.				
FD2H	PASR	.3	.2	.1	.0	W	No	No	Yes
FD3H		"0"	"0"	"0"	.4				
		Loc	cations FD4	IH-FD5H a	re not map	ped.			
FD6H	PNE1	.3	.2	.1	.0	W	No	No	Yes
FD7H		"0"	.6	.5	.4				
FD8H	PNE2	.3	.2	.1	.0		No	Yes	No
			Locations	FD9H is n	ot mapped	-			
FDAH	IMOD2	"0"	"0"	"0"	.0	W	No	Yes	No
			Locations	FDBH is n	ot mapped				
FDCH	PUMOD1	PUR3	PUR2	PUR1	PUR0	W	No	No	Yes
FDDH		PUR7	PUR6	PUR5	PUR4				
FDEH	PUMOD2	"0"	"0"	PUR9	PUR8		No	Yes	No
		•	Locations	FDFH is n	ot mapped				
FE0H	SMOD	.3	.2	.1	.0	W	.3	No	Yes
FE1H		.7	.6	.5	"0"				
		Loc	cations FE2	2H-FE3H a	re not map	ped.			
FE4H	SBUF					R/W	No	No	Yes
FE5H									
FE6H	PMG1	PM0.3	PM0.2	PM0.1	PM0.0	W	No	No	Yes
FE7H		"0"	PM2.2	PM2.1	PM2.0				
FE8H	PMG2	PM3.3	PM3.2	PM3.1	PM3.0				Yes
FE9H		"0"	"0"	"0"	"0"				
FEAH	PMG3	PM4.3	PM4.2	PM4.1	PM4.0				Yes
FEBH		PM5.3	PM5.2	PM5.1	PM5.0				
FECH	PMG4	PM6.3	PM6.2	PM6.1	PM6.0				Yes
FEDH		PM7.3	PM7.2	PM7.1	PM7.0				
FEEH	PMG5	PM8.3	PM8.2	PM8.1	PM8.0				Yes
FEFH		PM9.3	PM9.2	PM9.1	PM9.0	1			



Table 4-1. I/O Map for Memory Bank 15 (Concluded)

	Memory Bank 15							ressing N	lode
Address	Register	Bit 3	Bit 2	Bit 1	Bit 0	R/W	1-Bit	4-Bit	8-Bit
FF0H	Port 0	.3	.2	.1	.0	R/W	Yes	Yes	No
FF1H	Port 1	.3	.2	.1	.0	R			
FF2H	Port 2	"0"	.2	.1	.0	R/W			
FF3H	Port 3	.3	.2	.1	.0	R/W			
FF4H	Port 4	.3	.2	.1	.0	R/W			Yes
FF5H	Port 5	.3 / .7	.2 / .6	.1 / .5	.0 / .4				
FF6H	Port 6	.3	.2	.1	.0	R/W			
FF7H	Port 7	.3 / .7	.2 / .6	.1 / .5	.0 / .4				
FF8H	Port 8	.3	.2	.1	.0	R/W			
FF9H	Port 9	.3 / .7	.2 / .6	.1 / .5	.0 / .4				
	Locations FFAH-FFFH are not mapped.								

#### NOTES:

- 1. Bit 3 in the WMOD register is read only.
- 2. F9AH.0, F9AH.1 and F9AH.2 are fixed to "0".
- 3. The carry flag can be read or written by specific bit manipulation instructions only.
- 4. The "U" means that the bit is undefined.

#### **REGISTER DESCRIPTIONS**

In this section, register descriptions are presented in a consistent format to familiarize you with the memory-mapped I/O locations in bank 15 of the RAM. Figure 4-1 describes features of the register description format. Register descriptions are arranged in alphabetical order. Programmers can use this section as a quick-reference source when writing application programs.

Counter registers, buffer registers, and reference registers, as well as the stack pointer and port I/O latches, are not included in these descriptions. More detailed information about how these registers are used is included in Part II of this manual, "Hardware Descriptions," in the context of the corresponding peripheral hardware module descriptions.



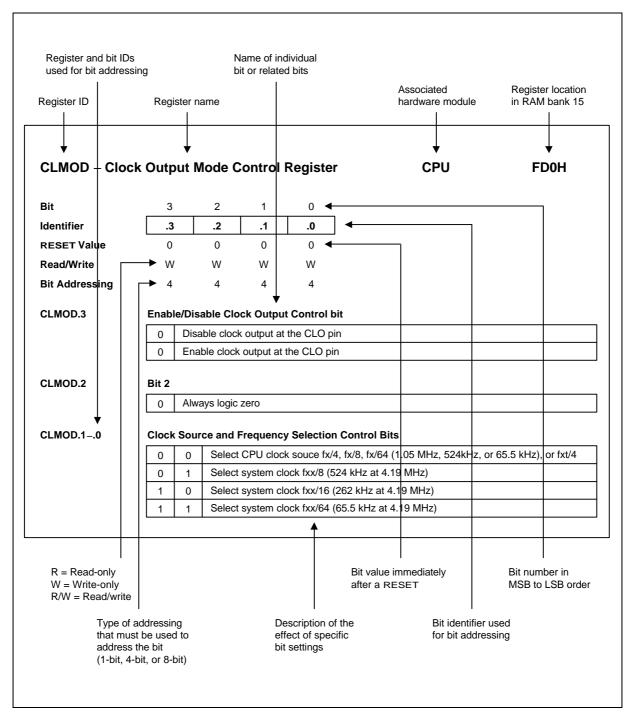


Figure 4-1. Register Description Format



# **BMOD** — Basic Timer Mode Register

BT

F85H

Bit	3	2	1	0
Identifier	.3	.2	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	1/4	4	4	4

#### BMOD.3 Basic Timer Restart Bit

1 Restart basic timer, then clear IRQB flag, BCNT and BMOD.3 to logic zero

#### BMOD.2 - .0 Input Clock Frequency and Interrupt Interval Time

0	0	0	Input clock frequency: Interrupt interval time (wait time)	fxx/2 <sup>12</sup> (1.02 kHz) 2 <sup>20</sup> /fxx (250 ms)
0	1	1	Input clock frequency: Interrupt interval time (wait time)	fxx/2 <sup>9</sup> (8.18 kHz) 2 <sup>17</sup> /fxx (31.3 ms)
1	0	1	Input clock frequency: Interrupt interval time (wait time)	fxx/2 <sup>7</sup> (32.7 kHz) 2 <sup>15</sup> /fxx (7.82 ms)
1	1	1	Input clock frequency: Interrupt interval time (wait time)	fxx/2 <sup>5</sup> (131 kHz) 2 <sup>13</sup> /fxx (1.95 ms)

#### NOTES:

- 1. When a RESET occurs, the oscillator stabilization wait time is 31.3 ms (2<sup>17</sup>/fxx) at 4.19 MHz.
- 2. 'fxx' is the system clock rate given a clock frequency of 4.19 MHz.



# **CLMOD** — Clock Output Mode Register

**CPU** 

FD0H

Bit	3	2	1	0
Identifier	.3	"0"	.1	.0
<b>RESET Value</b>	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

CLMOD.3 Enable/Disable Clock Output Control Bit

0	Disable clock output at the CLO pin
1	Enable clock output at the CLO pin

CLMOD.2 Bit 2

0 Always logic zero

#### CLMOD.1 - .0 Clock Source and Frequency Selection Control Bits

0	0	Select CPU clock source fx/4, fx/8, fx/64 (1.05 MHz, 524 kHz, or 65.5 kHz) or fxt/4
0	1	Select system clock fxx/8 (524 kHz)
1	0	Select system clock fxx/16 (262 kHz)
1	1	Select system clock fxx/64 (65.5 kHz)

**NOTE**: 'fxx' is the system clock, given a clock frequency of 4.19 MHz.



# IEO, IRQO — INTO Interrupt Enable/Request Flags

CPU FBEH

# IE1, IRQ1 — INT1 Interrupt Enable/Request Flags

CPU FBEH

Bit	3	2	1	0
Identifier	IE1	IRQ1	IE0	IRQ0
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

IE1 INT1 Interrupt Enable Flag

0	Disable interrupt requests at the INT1 pin
1	Enable interrupt requests at the INT1 pin

#### IRQ1 INT1 Interrupt Request Flag

 Generate INT1 interrupt (This bit is set and cleared by hardware when rising or falling edge detected at INT1 pin.)

IEO INTO Interrupt Enable Flag

	- · · · · · · · · · · · · · · · · · · ·
0	Disable interrupt requests at the INT0 pin
1	Enable interrupt requests at the INT0 pin

#### IRQ0 INT0 Interrupt Request Flag

 Generate INT0 interrupt (This bit is set and cleared automatically by hardware when rising or falling edge detected at INT0 pin.)



# IE2, IRQ2 — INT2 Interrupt Enable/Request Flags

**CPU** 

**FBFH** 

Bit	3	2	1	0
Identifier	"0"	"0"	IE2	IRQ2
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

.3 - .2 Bits 3-2

0 Always logic zero

IE2 INT2 Interrupt Enable Flag

0	Disable INT2 interrupt requests at the INT2 pin
1	Enable INT2 interrupt requests at the INT2 pin

#### IRQ2 INT2 Interrupt Request Flag

 Generate INT2 quasi-interrupt (This bit is set and is <u>not</u> cleared automatically by hardware when a rising or falling edge is detected at INT2. Since INT2 is a quasi-interrupt, IRQ2 flag must be cleared by software.)



# IE4, IRQ4 — INT4 Interrupt Enable/Request Flags

**CPU** 

FB8H

# IEB, IRQB — INTB Interrupt Enable/Request Flags

CPU

FB8H

Bit
Identifier
<b>RESET Value</b>
Read/Write
Bit Addressing

3	2	1	0
IE4	IRQ4	IEB	IRQB
0	0	0	0
R/W	R/W	R/W	R/W
1/4	1/4	1/4	1/4

IE4

#### **INT4 Interrupt Enable Flag**

(	)	Disable interrupt requests at the INT4 pin
1	1	Enable interrupt requests at the INT4 pin

IRQ4

#### **INT4 Interrupt Request Flag**

 Generate INT4 interrupt (This bit is set and cleared automatically by hardware when rising and falling signal edge detected at INT4 pin.)

**IEB** 

#### **INTB Interrupt Enable Flag**

0	Disable INTB interrupt requests
1	Enable INTB interrupt requests

**IRQB** 

#### **INTB Interrupt Request Flag**

 Generate INTB interrupt (This bit is set and cleared automatically by hardware when reference interval signal received from basic timer.)



# IES, IRQS — INTS Interrupt Enable/Request Flags CPU FBDH

Bit 3 2 Identifier "0" "0" **IES IRQS** 0 0 **RESET Value** 0 0 Read/Write R/W R/W R/W R/W **Bit Addressing** 1/4 1/4 1/4 1/4

.3 - .2 Bits 3-2

0 Always logic zero

IES INTS Interrupt Enable Flag

0	Disable INTS interrupt requests
1	Enable INTS interrupt requests

IRQS INTS Interrupt Request Flag

 Generate INTS interrupt (This bit is set and cleared automatically by hardware when serial data transfer completion signal received from serial I/O interface.)



# IETO, IRQTO — INTTO Interrupt Enable/Request Flags CPU FBCH

Bit	3	2	1	0
Identifier	"0"	"0"	IET0	IRQT0
RESET Value	0	0	0	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

.3 - .2 Bits 3-2

0 Always logic zero

IETO INTTO Interrupt Enable Flag

0	Disable INTT0 interrupt requests	
1	Enable INTT0 interrupt requests	

IRQT0 INTT0 Interrupt Request Flag

 Generate INTT0 interrupt (This bit is set and cleared automatically by hardware when contents of TCNT0 and TREF0 registers match.)



# IET1, IRQT1 — INTT1 Interrupt Enable/Request Flags

**CPU** 

**FBBH** 

# IEK, IRQK — INTK Interrupt Enable/Request Flags

**CPU** 

**FBBH** 

Bit
Identifier
<b>RESET Value</b>
Read/Write
Bit Addressing

3	2	1	0
IEK	IRQK	IET1	IRQT1
0	0	0	0
R/W	R/W	R/W	R/W
1/4	1/4	1/4	1/4

**IEK** 

#### **INTK Interrupt Enable Flag**

ſ	0	Disable interrupt requests at the K0-K7 pins
	1	Enable interrupt requests at the K0-K7 pins

**IRQK** 

#### **INTK Interrupt Request Flag**

Generate INTK interrupt (This bit is set and cleared automatically by hardware when rising or falling edge detected at K0-K7 pins.)

IET1

#### **INTT1 Interrupt Enable Flag**

0	Disable INTT1 interrupt requests
1	Enable INTT1 interrupt requests

#### **IRQT1**

#### **INTT1 Interrupt Request Flag**

Generate INTT1 interrupt (This bit is set and cleared automatically by hardware when contents of TCNT1 and TREF1 registers match.)



IEW, IRQW — INTW Interrupt Enable/Request Flags CPU FBAH							
Bit	3	2	1	0			
Identifier	"0"	"0"	IEW	IRQW			
RESET Value	0	0	0	0	_l		
Read/Write	R/W	R/W	R/W	R/W			
Bit Addressing	1/4	1/4	1/4	1/4			
.32	Bits 3-2						
	0 /	Always logic ze	ero				
IEW	INTW	Interrupt Ena	ble Flag				
	0 Disable INTW interrupt requests						
	1 Enable INTW interrupt requests						
					<del>-</del>		
IRQW	INTW	INTW Interrupt Request Flag					
	<ul> <li>Generate INTW interrupt (This bit is set when the timer interval is set to 0.5</li> </ul>						

**NOTE**: Since INTW is a quasi-interrupt, the IRQW flag must be cleared by software.

seconds or 3.91 milliseconds.)



# IMOD0 — External Interrupt 0 (INT0) Mode Register

CPU

FB4H

Bit	3	2	1	0
Identifier	"0"	"0"	.1	.0
<b>RESET Value</b>	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

IMOD0.3 - .2

#### Bits 3- 2

^	A I		
U	Always	iogic	zero

#### IMOD0.1 - .0

#### **External Interrupt Mode Control Bits**

0	0	Interrupt request is triggered by a rising signal edge
0	1	Interrupt request is triggered by a falling signal edge
1	0	Interrupt request is triggered by both rising and falling signal edges
1	1	Interrupt request flag (IRQ0) cannot be set to logic one

IMOD1 — External Interrupt 1	(INT1) Mode Register	CPU
	(III I I / IVIOGO I LOGISTOI	<b>U. U</b>

Bit 2 1 0 IMOD1.0 Identifier "0" "0" "0" **RESET Value** 0 0 0 0 Read/Write W W W W **Bit Addressing** 4 4 4 4

IMOD1.3 - .1 Bits 3-1

0 Always logic zero

IMOD1.0 External Interrupt 1 Edge Detection Control Bit

0	Rising edge detection
1	Falling edge detection



FB5H

# IMOD2 — External Interrupt 2 (INT2) Mode Register

CPU

**FDAH** 

Bit	3	2	1	0
Identifier	"0"	"0"	"0"	IMOD2.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

IMOD2.3 - .1

#### Bits 3-1

Λ	ΔΙνιανιο	logic	70r0
U	Always	logic	2610

#### IMOD2.0

### **External Interrupt 2 Edge Detection Selection Bit**

0	Interrupt request at INT2 pin trigged by rising edge
1	Interrupt request at INT2 pin trigged by falling edge



## **IMODK** — External Key Interrupt Mode Register

**CPU** 

FB6H

Bit	3	2	1	0
Identifier	"0"	IMODK.2	IMODK.1	IMODK.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

IMODK.3

Bit 3

0 Always logic zero

#### IMODK.2

#### **External Key Interrupt Edge Detection Selection Bit**

0	Falling edge detection		
1	Rising edge detection		

#### IMODK.1 - .0

#### **External Key Interrupt Mode Control Bits**

0	0	Disable key interrupt
0	1	Enable edge detection at K0-K3 pins
1	0	Enable edge detection at K4-K7 pins
1	1	Enable edge detection at K0-K7 pins

#### NOTES:

- 1. To generate a key interrupt, all of the selected pins must be configured to input mode. If any one of the selected pins is configured to output mode, only falling edge can be detected.
- 2. To generate a key interrupt, all of the selected pins must be at input high state for falling edge detection, or all of the selected pins must be at input low state for rising edge detection. If any one of them or more is at input low state or input high state, the interrupt may be not occurred at falling edge or rising edge.
- 3. To generate a key interrupt, first, configure pull-up resistors or external pull-down resistors. And then, select edge detection and pins by setting IMODK register.



# IPR — Interrupt Priority Register

**CPU** 

FB2H

Bit	3	2	1	0
Identifier	IME	.2	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	1/4	4	4	4

IME

#### **Interrupt Master Enable Bit**

0	Disable all interrupt processing
1	Enable processing for all interrupt service requests

#### IPR.2 - .0

#### **Interrupt Priority Assignment Bits**

0	0	0	Normal interrupt handling according to default priority settings
0	0	1	Process INTB and INT4 interrupts at highest priority
0	1	0	Process INT0 interrupt at highest priority
0	1	1	Process INT1 interrupt at highest priority
1	0	0	Process INTS interrupt at highest priority
1	0	1	Process INTT0 interrupt at highest priority
1	1	0	Process INTT1 interrupt at highest priority
1	1	1	Process INTK interrupt at highest priority

## LCON — LCD Output Control Register

LCD

F8EH

Bit	3	2	1	0
Identifier	"0"	.2	.1	.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

LCON.3

Bit 3

0 Always logic zero

#### LCON.2 LCD Clock Output Disable/Enable Bit

0	Disable LCDCK and LCDSY signal outputs.
1	Enable LCDCK and LCDSY signal outputs.

#### LCON.1 - .0 LCD Output Control Bit

0	0	LCD display off; cut off current to dividing resistor
0	1	LCD display on; application without contrast control
1	0	LCD display on; application with contrast control
1	1	LCD dispaly on; application without contrast control

#### NOTES:

- 1. The function of LCON.0 is applied in case of using the internal GND for LCD power; the function of LCON.1 is used for contrast control application.
- 2. The table for LCON.1-LCON.0 also shows the case that internal bias resistors are built-in by mask option. For the case that external bias resistors are configured by mask option, refer to Chapter 12.



## LMOD — LCD Mode Register

LCD F8DH, F8CH

Bit	3	2	1	0	3	2	1	0
Identifier	.7	.6	.5	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

#### LMOD.7 - .5 LCD Output Segment and Pin Configuration Bits

0	0	0	Segments 40-43, 44-47, 48-51 and 52-55
0	0	1	Segments 40-43, 44-47 and 48-51; normal I/O at port 6
0	1	0	Segments 40-43 and 44-47; normal I/O at port 6 and port 7
0	1	1	Segments 40-43; normal I/O at port 6, 7 and 8
1	0	0	Normal I/O at port 6, 7, 8 and 9

**NOTE:** Segment pins that also can used for normal I/O should be configured to output mode when the SEG function is used.

#### LMOD.4 - .3 LCD Clock (LCDCK) Frequency Selection Bits

0	0	When 1/8 duty: fxx/2 <sup>7</sup> (256 Hz); when 1/16 duty: fxx/2 <sup>6</sup> (512 Hz)
0	1	When 1/8 duty: fxx/2 <sup>6</sup> (512 Hz); when 1/16 duty: fxx/2 <sup>5</sup> (1024 Hz)
1	0	When 1/8 duty: fxx/2 <sup>5</sup> (1024 Hz); when 1/16 duty: fxx/2 <sup>4</sup> (2048 Hz)
1	1	When 1/8 duty: fxx/2 <sup>4</sup> (2048 Hz); when 1/16 duty: fxx/2 <sup>3</sup> (4096 Hz)

**NOTE:** LCDCK is supplied only when the watch timer operates. To use the LCD controller, bit 2 in the watch mode register WMOD should be set to 1.

#### LMOD.2 LCD Duty and Selection Bits

0	1/8 duty (COM0-COM7 select)
1	1/16 duty (COM0-COM15 select)

**NOTE:** When 1/16 duty is selected, ports 4 and 5 should be configured as output mode; when 1/8 duty is selected, ports 4 and 5 can be used as normal I/O ports.

#### LMOD.1 - .0 LCD Display Mode Selection Bits

		· ·
0	0	All LCD dots off
0	1	All LCD dots on
1	1	Normal display



## **PASR** — Page Selection Register

FD2H

Bit	7	6	5	4	3	2	1	0
Identifier	"0"	"0"	"0"	.4	.3	.2	.1	.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8

#### PASR.7 - .5

#### Bits 7-4

0	Always	logic	zero
•	, livvayo	10910	2010

#### PASR.4 - .0

#### Page Selection Register in the Bank 1

0	0	0	0	0	00H page in the bank 1		
0	0	0	0	1	H page in the bank 1		
		•			"		
		•					
1	1	1	1	0	1EH page in the bank 1		

**NOTE:** The 00H-1EH pages can be used in the KS57C21832.



## **PCON** — Power Control Register

**CPU** 

FB3H

Bit	3	2	1	0
Identifier	.3	.2	.1	.0
<b>RESET Value</b>	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

#### PCON.3 - .2 CPU Operating Mode Control Bits

	•	<del>-</del>
0	0	Enable normal CPU operating mode
0	1	Initiate idle power-down mode
1	0	Initiate stop power-down mode

#### PCON.1 - .0 CPU Clock Frequency Selection Bits

0	0	If SCMOD.0 = "0", fx/64; if SCMOD.0 = "1", fxt/4
1	0	If SCMOD.0 = "0", fx/8; if SCMOD.0 = "1", fxt/4
1	1	If SCMOD.0 = "0", fx/4; if SCMOD.0 = "1", fxt/4

**NOTE**: 'fx' is the main system clock; 'fxt' is the subsystem clock.

PMG1 — Port I/O Mode Register 1 (Group 1: Port 0, 2) I/O FE7H, FE6								7H, FE6H	
Bit		7	6	5	4	3	2	1	0
Identifier	ıı	0"	PM2.2	PM2.1	PM2.0	PM0.3	PM0.2	PM0.1	PM0.0
RESET Value		0	0	0	0	0	0	0	0
Read/Write	١	W	W	W	W	W	W	W	W
Bit Addressing		8	8	8	8	8	8	8	8
.7	Bit	7							
	0	Alwa	ays logic ze	ero					
PM2.2	P2.2	2 1/0 1	Mode Sele	ction Flag					
	0	Set	P2.2 to inp	ut mode					
	1	Set	P2.2 to out	put mode					
PM2.1	P2.	1 I/O I	Mode Sele	ction Flag					
	0	1	P2.1 to inp						
	1	Set	P2.1 to out	put mode					
P.40 0									
PM2.0		1	Mode Sele						
	1	+	P2.0 to inp						
	'	Set	P2.0 to out	put mode					
PM0.3	P0.3	3 I/O I	Mode Sele	ction Flag					
	0	Set	P0.3 to inp	ut mode					
	1	Set	P0.3 to out	put mode					
PM0.2		T	Mode Sele						
	0		P0.2 to inp						
	1	Set	P0.2 to out	put mode					
PM0.1	P0.	1 1/0 1	Mode Sele	ction Flag					
	0	Set	P0.1 to inp	ut mode					
	1	Set	P0.1 to out	put mode					
P.440 0				<del>-</del> -					·
PM0.0			Mode Sele						
	0	Set	P0.0 to inp	ut mode					

Set P0.0 to output mode



PMG2 — Port I/O Mode Register 2 (Group 2: Port 3) I/O FE9H, FE							9H, FE8H	
Bit	7	6	5	4	3	2	1	0
Identifier	"0"	"0"	"0"	"0"	PM3.3	PM3.2	PM3.1	PM3.0
RESET Value	0	0	0	0	0	0	0	0
Read/Write	W	W	W	W	W	W	W	W
Bit Addressing	8	8	8	8	8	8	8	8
.74	Bits 7-4							
	0 Alwa	ays logic ze	ero					
PM3.3	P3.3 I/O I	Mode Sele	ction Flag					
	0 Set	P3.3 to inp	ut mode					
	1 Set	P3.3 to out	put mode					
PM3.2	P3.2 I/O I	Mode Sele	ction Flag					
	0 Set	P3.2 to inp	ut mode					
	1 Set	P3.2 to out	put mode					
PM3.1	P3.1 I/O I	Mode Sele	ction Flag					
		P3.1 to inp						
		P3.1 to out						
PM3.0	P3.0 I/O I	Mode Sele	ction Flag					

0 Set P3.0 to input mode

Set P3.0 to output mode

1



PMG3 — Port I/	О Мо	de R	egister 3	(Group 3	3: Port 4,	5)	I/O	FE	FEBH, FEAH		
Bit		7	6	5	4	3	2	1	0		
Identifier	PN	<b>/</b> 15.3	PM5.2	PM5.1	PM5.0	PM4.3	PM4.2	PM4.1	PM4.0		
RESET Value		0	0	0	0	0	0	0	0		
Read/Write	,	W	W	W	W	W	W	W	W		
Bit Addressing		8	8	8	8	8	8	8	8		
PM5.3	P5.	3 1/0 1	Mode Sele	ction Flag							
	0	Set	P5.3 to inp	ut mode							
	1	Set	P5.3 to out	put mode							
PM5.2	P5.2	2 1/0 1	Mode Sele	ction Flag							
	0	Set	P5.2 to inp	ut mode							
	1	Set	P5.2 to out	put mode							
PM5.1	P5.	1 1/0 1	Mode Sele	ction Flag							
	0	Set	P5.1 to inp	ut mode							
	1	Set	P5.1 to out	put mode							
PM5.0	P5.0	0 1/0 1	Mode Sele	ction Flag							
	0	Set	P5.0 to inp	ut mode							
	1	Set	P5.0 to out	put mode							
PM4.3	P4.:	3 1/0 1	Mode Sele	ction Flag							
	0	1	P4.3 to inp								
	1	Set	P4.3 to out	put mode							
PM4.2	P4.2	2 1/0 1	Mode Sele	ction Flag							
	0	Set	P4.2 to inp	ut mode							
	1	Set	P4.2 to out	put mode							
PM4.1	P4.	1 1/0 1	Mode Sele	ction Flag							
	0	Set	P4.1 to inp	ut mode							
	1	Set	P4.1 to out	put mode							
PM4.0	P4.0	0  /0	Mode Sele	ction Flag							
-	0	1	P4.0 to inp								

Set P4.0 to output mode



PMG4 — Port I/O	Mod	de Re	egister 4	(Group 4	4: Port 6,	7)	I/O FEDH, FE		
Bit		7	6	5	4	3	2	1	0
Identifier	PN	17.3	PM7.2	PM7.1	PM7.0	PM6.3	PM6.2	PM6.1	PM6.0
RESET Value		0	0	0	0	0	0	0	0
Read/Write	١	N	W	W	W	W	W	W	W
Bit Addressing		8	8	8	8	8	8	8	8
PM7.3	P7.3	3 I/O N	/lode Seled	ction Flag					
	0	Set I	7.3 to inp	ut mode					
	1	Set I	P7.3 to out	put mode					
PM7.2	P7.2	2 I/O N	/lode Seled	ction Flag					
	0	1	P7.2 to inpu						
	1		27.2 to out						
PM7.1	P7 1	I/O N	/lode Seled	ction Flag					
	0		7.1 to inpu						
	1		27.1 to out						
PM7.0	P7.0	) I/O N	/lode Seled	ction Flag					
	0	Set I	7.0 to inpu	ut mode					
	1	Set I	P7.0 to out	put mode					
PM6.3	P6.3	3 I/O N	/lode Seled	ction Flag					
	0	Set I	P6.3 to inpu	ut mode					
	1	Set I	P6.3 to out	put mode					
PM6.2	P6.2	2 I/O N	/lode Seled	ction Flag					
	0	Set I	P6.2 to inpu	ut mode					
	1	Set I	P6.2 to out	put mode					
PM6.1	P6.1	I I/O N	/lode Seled	ction Flag					
	0	Set I	P6.1 to inpu	ut mode					
	1	Set I	P6.1 to out	put mode					
PM6.0	P6.0	) I/O N	/lode Seled	ction Flag					
	0	Set I	P6.0 to inpu	ut mode					
	1	Set I	P6.0 to out	put mode					



Bit         7         6         5         4         3         2         1         0           Identifier         PM9.3         PM9.2         PM9.1         PM9.0         PM8.3         PM8.2         PM8.1         PM8           RESET Value         0         0         0         0         0         0         0         0           Read/Write         W         W         W         W         W         W         W           Bit Addressing         8         8         8         8         8         8         8           PM9.3           P9.3 I/O Mode Selection Flag           0         Set P9.3 to input mode	
Identifier         PM9.3         PM9.2         PM9.1         PM9.0         PM8.3         PM8.2         PM8.1         PM8           RESET Value         0 <t< th=""><th></th></t<>	
Read/Write         W	)
Bit Addressing         8	
PM9.3 P9.3 I/O Mode Selection Flag  0 Set P9.3 to input mode	
0 Set P9.3 to input mode	
1 Set P9.3 to output mode	
PM9.2 P9.2 I/O Mode Selection Flag	
0 Set P9.2 to input mode	
1 Set P9.2 to output mode	
PM9.1 P9.1 I/O Mode Selection Flag	
0 Set P9.1 to input mode	
1 Set P9.1 to output mode	
PM9.0 P9.0 I/O Mode Selection Flag	
0 Set P9.0 to input mode	
1 Set P9.0 to output mode	
PM8.3 P8.3 I/O Mode Selection Flag	
0 Set P8.3 to input mode	
1 Set P8.3 to output mode	
PM8.2 P8.2 I/O Mode Selection Flag	
0 Set P8.2 to input mode	
1 Set P8.2 to output mode	
PM8.1 P8.1 I/O Mode Selection Flag	
0 Set P8.1 to input mode	
1 Set P8.1 to output mode	
	_
PM8.0 P8.0 I/O Mode Selection Flag  0 Set P8.0 to input mode	



1

Set P8.0 to output mode

Bit 7 6 5 4	3			
Identifier "0" PNE1.6 PNE1.5 PNE1.4	PNE1.3	2 <b>PNE1.2</b>	1 PNE1.1	0 <b>PNE1.0</b>
RESET Value         0         0         0         0           Read/Write         W         W         W	0 W	0 W	0 W	0 W
Bit Addressing 8 8 8 8	8	8	8	8
.7 Bit 7				
0 Always logic 0				
PNE1.6 P2.2 N-Channel Open-Drain Configurable	Bit			
0 Configure P2.2 as a push-pull				
1 Configure P2.2 as a n-channel open-di	rain			
PNE1.5 P2.1 N-Channel Open-Drain Configurable	Bit			
0 Configure P2.1 as a push-pull				
1 Configure P2.1 as a n-channel open-di	rain			
				_
PNE1.4 P2.0 N-Channel Open-Drain Configurable	Bit			
0 Configure P2.0 as a push-pull				
1 Configure P2.0 as a n-channel open-di	rain			
PNE1.3 P0.3 N-Channel Open-Drain Configurable	Bit			
0 Configure P0.3 as a push-pull				
1 Configure P0.3 as a n-channel open-di	rain			
PNE1.2 P0.2 N-Channel Open-Drain Configurable	Bit			
0 Configure P0.2 as a push-pull				
1 Configure P0.2 as a n-channel open-di	rain			
PNE1.1 P0.1 N-Channel Open-Drain Configurable	Bit			
0 Configure P0.1 as a push-pull				
1 Configure P0.1 as a n-channel open-di	rain			
PNE1.0 P0.0 N-Channel Open-Drain Configurable	Bit			
0 Configure P0.0 as a push-pull				
1 Configure P0.0 as a n-channel open-di	rain			



## PNE2 — N-Channel Open-Drain Mode Register 2

I/O

FD8H

Bit	3	2	1	0
Identifier	PNE2.3	PNE2.2	PNE2.1	PNE2.0
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

#### PNE2.3 P3.3 N-Channel Open-Drain Configurable Bit

0	Configure P3.3 as a push-pull
1	Configure P3.3 as a n-channel open-drain

#### PNE2.2 P3.2 N-Channel Open-Drain Configurable Bit

0	)	Configure P3.2 as a push-pull
1		Configure P3.2 as a n-channel open-drain

#### PNE2.1 P3.1 N-Channel Open-Drain Configurable Bit

0	Configure P3.1 as a push-pull
1	Configure P3.1 as a n-channel open-drain

#### PNE2.0 P3.0 N-Channel Open-Drain Configurable Bit

0	Configure P3.0 as a push-pull
1	Configure P3.0 as a n-channel open-drain



## **PSW** — Program Status Word

CPU FB1H, FB0H

7	6	5	4	3	2	1	0
С	SC2	SC1	SC0	IS1	IS0	EMB	ERB
(1)	0	0	0	0	0	0	0
R/W	R	R	R	R/W	R/W	R/W	R/W
(2)	8	8	8	1/4/8	1/4/8	1/4/8	1/4/8

C Carry Flag

0	No overflow or borrow condition exists
1	An overflow or borrow condition does exist

#### SC2 - SC0 Skip Condition Flags

0	No skip condition exists; no direct manipulation of these	bits is allowed
1	A skip condition exists; no direct manipulation of these b	its is allowed

#### IS1, IS0 Interrupt Status Flags

0	0	Service all interrupt requests
0	1	Service only the high-priority interrupt(s) as determined in the interrupt priority register (IPR)
1	0	Do not service any more interrupt requests
1	1	Undefined

#### EMB Enable Data Memory Bank Flag

0	Restrict program access to data memory to bank 15 (F80H-FFFH) and to the locations 000H-07FH in the bank 0 only
1	Enable full access to data memory banks 0, 1, 2, and 15

#### ERB Enable Register Bank Flag

0	Select register bank 0 as working register area
1	Select register banks 0, 1, 2, or 3 as working register area in accordance with the select register bank (SRB) instruction operand

#### NOTES:

- 1. The value of the carry flag after a RESET occurs during normal operation is undefined. If a RESET occurs during power-down mode (IDLE or STOP), the current value of the carry flag is retained.
- 2. The carry flag can only be addressed by a specific set of 1-bit manipulation instructions. See Section 2 for detailed information.



PUMOD1 — Pull-up Resistor Mode Register 1 I/O FDD					OH, FDCH				
Bit	7		6	5	4	3	2	1	0
Identifier	PUI	R7	PUR6	PUR5	PUR4	PUR3	PUR2	PUR1	PUR0
RESET Value	0	1	0	0	0	0	0	0	0
Read/Write	W	1	W	W	W	W	W	W	W
Bit Addressing	8		8	8	8	8	8	8	8
PUR7	Conr	nect/l	Disconnec	t Port 7 P	ull-up Resi	istor Contr	ol Bit		
	0	Disco	onnect por	t 7 pull-up	resistor				
				pull-up res					
PUR6	Conr	nect/l	Disconnec	t Port 6 P	ull-up Resi	istor Contr	ol Bit		
	0	Disco	onnect por	t 6 pull-up	resistor				
	1	Conr	nect port 6	pull-up res	istor				
PUR5	Conr	nect/l	Disconnec	t Port 5 P	ull-up Resi	istor Contr	ol Bit		
	0	Disc	onnect por	t 5 pull-up	resistor				
	1	Conr	nect port 5	pull-up res	istor				
PUR4	Conr	nect/l	Disconnec	t Port 4 P	ull-up Resi	istor Contr	ol Bit		
	0	Disco	onnect por	t 4 pull-up	resistor				
	1 Connect port 4 pull-up resistor								
PUR3	Conr	nect/l	Disconnec	of Dort 3 D	ull-un Pasi	istor Contr	ol Rit		
FORS				t 3 pull-up	-	istor Conti	OI BIL		
	-			pull-up res					
PUR2	Conr	nect/l	Disconnec	et Port 2 Pi	ull-un Resi	istor Contr	ol Rit		
1 0112				t 2 pull-up	<del>-</del>	10101 001111	O. D		
			•	pull-up res					
PUR1									
	0 Disconnect port 1 pull-up resistor								
	1 Connect port 1 pull-up resistor								
PUR0	Conr	nect/l	Disconnec	t Port 0 P	ull-up Resi	istor Contr	ol Bit		

Disconnect port 0 pull-up resistor

Connect port 0 pull-up resistor



## **PUMOD2** — Pull-up Resistor Mode Register 2

I/O

**FDEH** 

Bit	3	2	1	0
Identifier	"0"	"0"	PUR9	PUR8
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	4	4	4	4

.3 - .2 Bits 3-2

0 Always cleared to logic zero

#### PUR9 Connect/Disconnect Port 9 Pull-up Resistor Control Bit

0	Disconnect port 9 pull-up resistor
1	Connect port 9 pull-up resistor

#### PUR8 Connect/Disconnect Port 8 Pull-up Resistor Control Bit

0	Disconnect port 8 pull-up resistor
1	Connect port 8 pull-up resistor



SCMOD - Syste	lock		CPU	FB7H				
Bit	;	3	2	1	0			
Identifier		3	.2	"0"	.0			
RESET Value	(	0	0	0	0	_		
Read/Write	١	V	W	W	W			
Bit Addressing		1	1	1	1			
SCMOD.3	<b>Bit</b> 3	1	le main sy	stem clock	(fx)			
	1	Disab	ole main s	ystem clock	(fx)			
SCMOD.2	Bit 2	2						
	0	Enab	le sub sys	tem clock (	fxt)			
	1	Disab	ole sub sy	stem clock	(fxt)			
SCMOD.1	Bit <sup>2</sup>	1						
	0 Always logic zero							

#### NOTES:

SCMOD.0

Bit 0

- Sub-oscillation goes into stop mode only by SCMOD.2. PCON which revokes stop mode cannot stop the sub-oscillation.
- You can use SCMOD.2 as follows (ex; after data bank was used, a few minutes have passed):
   Main operation → sub-operation → sub-idle (LCD on, after a few minutes later without any external input) → sub-operation → main operation → SCMOD.2 = 1 → main stop mode (LCD off).
- 3. SCMOD bits 3-0 cannot be modified simultaneously by a 4-bit instruction; they can only be modified by separate 1-bit instructions.

Select main system clock (fx)
Select sub system clock (fxt)



SMOD — Seria	SMOD — Serial I/O Mode Register											
Bit		7		6	5	4	3	2	1	0		
Identifier	.7			.6	.5	"0"	.3	.2	.1	.0		
RESET Value	(	0		0	0	0	0	0	0	0		
Read/Write	٧	V	١	W	W	W	W	W	W	W		
Bit Addressing	;	8 8			8	8	1/8	8	8	8		
SMOD.75	Serial I/O Clock Selection and SBUF R/W Status Control Bits											
	0	0	0	Ena	ble SBUF w		peration is h		en SCK go	es high		
	0	0	1				mer/counter peration is h		ien SCK go	es high		
	0	1	х				k (fxx/4, 8, orite operatio		-			
	1	0	0	4.09	kHz clock	(fxx/2 <sup>10</sup> )						
	1	1	1				e: You canr d a CPU clo		ot select a fxx/2 <sup>4</sup> clock fre- ck of fxx/64			
SMOD.4	NOT		l kHz	freque	ncy ratings a	assume a sys	stem clock of	4.19 MHz.				
OMOD.4	0	1	avs Ic	gic ze	ero.							
SMOD.3	Initia 1	ate Se	erial ar IRC sion. '	<b>I/O O</b>	peration B	clock count	er to logic z					
SMOD.2	<b>Ena</b> 0	when serial transmission is completed										
SMOD.1	0	al I/O	Trar	nsmis only m	sion Mode	Selection	Bit					
	1 Transmit-and-receive mode											

LSB/MSB Transmission Mode Selection Bit

Transmit the most significant bit (MSB) first
Transmit the least significant bit (LSB) first



SMOD.0

TMOD0 — Timer	TMOD0 — Timer/Counter 0 Mode Register T/C0 F91H, F90H										
Bit		7	6	5	4	3	2	1	0		
Identifier	"(	0"	.6	.5	.4	.3	.2	"0"	"0"		
RESET Value	(	0	0	0	0	0	0	0	0		
Read/Write	١	V	W	W	W	W	W	W	W		
Bit Addressing	;	8	8	8	8	1/8	8	8	8		
TMOD0.7	Bit 7	7									
	0	Alwa	ays logic	zero							
TMOD0.64	Timer/Counter 0 Input Clock Selection Bits  0 0 0 External clock input at TCL0 pin on rising edge										
	0	0		ternal clock i	•						
	1	0		(/2 <sup>10</sup> (4.09 kł		Lo pin on ic	annig cage				
	1	0		/2 <sup>6</sup> (65.5 kH	,						
	1	1		(/2 <sup>4</sup> (262 kHz	<u> </u>						
	1	1		(4.19 MHz)	,						
	NOT	E: "fx	x" is selec	ted system clo	ck of 4.19 N	ИНz					
TMOD0.3	NOTE: "fxx" is selected system clock of 4.19 MHz  Clear Counter and Resume Counting Control Bit  Clear TCNT0, IRQT0, and TOL0 and resume counting immediately (This bit is cleared automatically when counting starts.)										
TMOD0.2	Ena	ble/D	isable Ti	mer/Counte	r 0 Bit						
	0	Disa	Disable timer/counter 0; retain TCNT0 contents								
	1	Enal	ole timer/	counter 0							
TMOD0.1	<b>Bit</b> '	1	ays logic	zero							



TMOD0.0

Bit 0

Always logic zero

TMOD1 — Timer/Counter 1 Mode Register T/C										1H, FA0H	
Bit	;	3	2	2	1	0	3	2	1	0	
Identifier	"(	0"		6	.5	.4	.3	.2	"0"	"0"	
RESET Value	(	0	(	)	0	0	0	0	0	0	
Read/Write	٧	٧	٧	٧	W	W	W	W	W	W	
Bit Addressing	8	8	8		8	8	1/8	8	8	8	
TMOD1.7	Bit 7	7									
	0	Alwa	ays lo	gic ze	ro						
TMOD1.64	Time	er/Co	unter	1 Inp	out Clock S	Selection	Bit				
0 0 External clock input at TCL1 pin on rising edge											
	0	0	1	Exte	rnal clock i	nput at TC	L1 pin on fa	alling edge			
	1	0	0	fxx/2	<sup>10</sup> (4.09 kH	Hz)					
	1	0	1	fxx/2	<sup>8</sup> (16.4 kH:	z)					
	1	1	0	fxx/2	x/2 <sup>6</sup> (65.5 kHz)						
	1	1	1	fxx/2	<sup>4</sup> (262 kHz	:)					
	NOT	E: "fxx	" is se	lected	system cloc	ck of 4.19 M	Hz				
		_			_						
TMOD1.3					Resume C				Р		
	1						d resume co en counting		ediately		
		(	<i>.</i>	o o o o o	ou automo	thouny will	on ocurring	otario.,			
TMOD1.2	Ena	ble/D	isable	e Time	er/Counte	r 1 Bit					
	0	Disa	ble tir	mer/co	ounter 1; re	tain TCNT	1 contents				
	1	Enal	ole tin	ner/co	unter 1						
TMOD1.1	Bit 1	1									
	0	Alwa	ays lo	gic ze	ro						
	L	1	<u> '</u>	-							



TMOD1.0

Bit 0

Always logic zero

### **TOE** — Timer Output Enable Flag Register

T/C

F92H

Bit	3	2	1	0
Identifier	TOE1	TOE0	"U"	"0"
RESET Value	0	0	U	0
Read/Write	R/W	R/W	R/W	R/W
Bit Addressing	1/4	1/4	1/4	1/4

TOE1

#### **Timer/Counter 1 Output Enable Flag**

(	)	Disable timer/counter 1 clock output at the TCLO1 pin
1	1	Enable timer/counter 1 clock output at the TCLO1 pin

TOE0

#### Timer/Counter 0 Output Enable Flag

0	Disable timer/counter 0 clock output at the TCLO0 pin
1	Enable timer/counter 0 clock output at the TCLO0 pin

.1

#### Bits 1

U	This bit is undefined

.0

#### Bits 0

0 Always logic zero

**NOTE:** The "U" means that the bit is undefined.



## WDFLAG — Watchdog Timer's Counter Clear Flag

WT

F9AH.3

Bit	3	2	1	0
Identifier	WDTCF	"0"	"0"	"0"
RESET Value	0	0	0	0
Read/Write	W	W	W	W
Bit Addressing	1/4	1/4	1/4	1/4

WDTCF Watchdog Timer's Counter Clear Bit

0	-
1	Clear the WDT's counter to zero and restart the WDT's counter

#### WDFLAG.2 - .0 Bit2-0

0	Always logic zero	
---	-------------------	--



WDMOD — Watchdog Timer Mode Control Register WT F99H, F98H												
Bit	3		2	1	1	(	)	3	2	1	0	
Identifier	.7		6	.!	5		4	.3	.2	.1	.0	
RESET Value	1		0	1	1	(	)	0	1	0	1	
Read/Write	W	١	V	V	V	V	V	W	W	W	W	
Bit Addressing	8		8	8	3	8	3	8	8	8	8	
WMOD.70	WMOD.70 Watchdog Timer Enable/Disable Control											
	0	1 0	1	1	0	1	0	Disable v	atchdog tim	ner function	n	

Enable watchdog timer function

Other Values



WMOD — Watch	Tim	er M	lode Regi	ister			WT	F	F89H, F88H
Bit		7	6	5	4	3	2	1	0
Identifier		.7	"0"	.5	.4	.3	.2	.1	.0
RESET Value		0	0	0	0	(note)	0	0	0
Read/Write	١	Ν	W	W	W	R	W	W	W
Bit Addressing		8	8	8	8	1	8	8	8
WMOD.7	Ena	ble/D	isable Buz	zer Outpu	t Bit				
	0	Disa	ble buzzer	(BUZ) sigr	nal output a	t the BUZ p	oin		
	1	Ena	ble buzzer (	(BUZ) sign	al output a	t the BUZ p	in		
WMOD.6	Bit	6							
	0	Alwa	ays logic ze	ro					
WMOD.54	Out	put B	uzzer Freq	uency Se	lection Bits	s			
	0	0	2 kHz buz	zer (BUZ)	signal outp	ut			
	0	1			signal outp				
	1	0			signal outp				
	1	1	16 kHz bu	zzer (BUZ	) signal out	put			
WMOD.3	ΧΤ <sub>ΙΝ</sub>	Inpu	ıt Level Co	ntrol Bit					
	0	Inpu	t level to X	Γ <sub>IN</sub> pin is lo	ow; 1-bit rea	ad-only add	lressable fo	r tests	
	1	Inpu	t level to X7	Γ <sub>IN</sub> pin is h	igh; 1-bit re	ead-only ad	dressable f	or tests	
WMOD.2	Enable/Disable Watch Timer Bit								
	0	Disa	ble watch t	imer and c	lear freque	ncy dividing	g circuits		
	1	Ena	ble watch ti	mer					
WMOD.1	Watch Timer Speed Control Bit								
	0	Normal speed; set IRQW to 0.5 seconds							
	1	High	n-speed ope	eration; set	IRQW to 3	3.91 ms			
WMOD.0	Wat	ch Ti	mer Clock	Selection	Bit				
	0	Sele	ect main sys	tem clock	(fx)/128 as	the watch	timer clock		
	1	1 Select a subsystem clock as the watch timer clock							

**NOTE**: RESET sets WMOD.3 to the current input level of the subsystem clock, XT<sub>IN</sub>. If the input level is high, WMOD.3 is set to logic one; if low, WMOD.3 is cleared to zero along with all the other bits in the WMOD register.



#### **NOTES**



**Oscillator Circuits** 

**Interrupts** 

**Power-Down** 

**RESET** 

I/O Ports

**Timers and Timer/Counters** 

**LCD Controller/Driver** 

Serial I/O Interface

**Electrical Data** 

**Mechanical Data** 

KS57P21832 OTP

**Development Tools** 

KS57C21832/P21832 OSCILLATOR CIRCUITS

# 6

#### **OSCILLATOR CIRCUITS**

#### **OVERVIEW**

The KS57C21832 microcontroller have two oscillator circuits: a main system clock circuit, and a subsystem clock circuit. The CPU and peripheral hardware operate on the system clock frequency supplied through these circuits. Specifically, a clock pulse is required by the following peripheral modules:

- LCD controller
- Basic timer
- Timer/counter 0 and 1
- Watch timer
- Serial I/O interface
- Clock output circuit

#### **CPU Clock Notation**

In this document, the following notation is used for descriptions of the CPU clock:

fx Main-system clock fxt Sub-system clock fxx Selected system clock

#### **Clock Control Registers**

When the system clock mode register, SCMOD, and the power control register, PCON, are both cleared to zero after RESET, the normal CPU operating mode is enabled, a main system clock is selected as fx/64, and main system clock oscillation is initiated.

The PCON is used to select normal CPU operating mode or one of two power down mode-stop or idle. Bits 3 and 2 of the PCON register can be manipulated by STOP of IDLE instruction to engage stop or idle power down mode.

The SCMOD, lets you select the main system clock (fx) or a subsystem clock (fxt) as the CPU clock and start (or stop) main/sub system clock oscillation. The resulting clock source, either main system clock or subsystem clock, is referred to the selected system clock (fxx).

The main system clock is selected and oscillation started when all SCMOD bits are cleared to "0". By setting SCMOD.3, SCMOD.2 and SCMOD.0 to different values, you can select a subsystem clock source and start or stop main/sub system clock oscillation. To stop main system clock oscillation, you must use the STOP instruction (assuming the main system clock is selected) or manipulate SCMOD.3 (assuming the sub system clock is selected).

The main system clock frequencies can be divided by 4, 8, or 64 and a subsystem clock frequencies can only be divided by 4. By manipulating PCON bits 1 and 0, you select one of the following frequencies as the CPU clock.

fx/4, fxt/4, fx/8, fx/64



OSCILLATOR CIRCUITS KS57C21832/P21832

#### **Using a Subsystem Clock**

If a subsystem clock is being used as the selected system clock, the idle power-down mode can be initiated by executing an IDLE instruction.

The watch timer, buzzer and LCD display operate normally with a subsystem clock source, since they operate at very low speed (as low as 122  $\mu$ s at 32.768 kHz) and with very low power consumption.

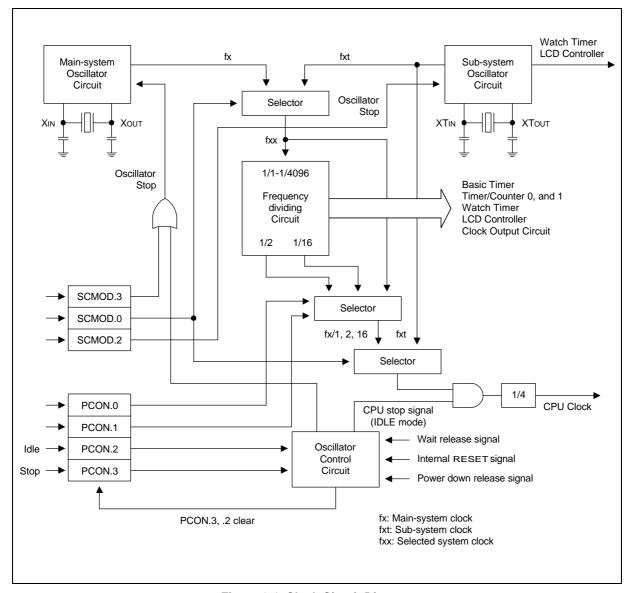


Figure 6-1. Clock Circuit Diagram



KS57C21832/P21832 OSCILLATOR CIRCUITS

#### MAIN SYSTEM OSCILLATOR CIRCUITS

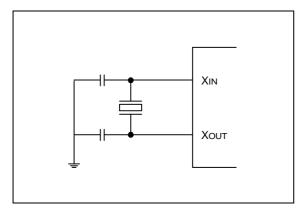


Figure 6-2. Crystal/Ceramic Oscillator (fx)

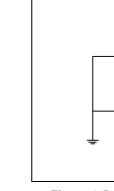


Figure 6-5. Crystal/Ceramic Oscillator (fxt)

32.768 kHz

XTIN

ХТоит

SUB SYSTEM OSCILLATOR CIRCUITS

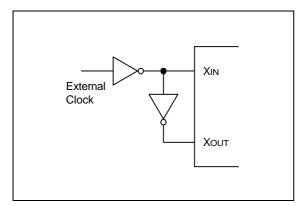


Figure 6-3. External Oscillator (fx)

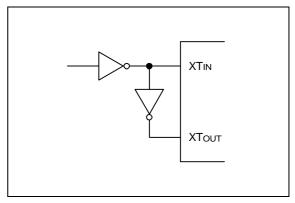


Figure 6-6. External Oscillator (fxt)

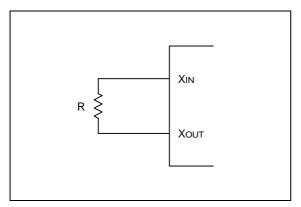


Figure 6-4. RC Oscillator (fx)

OSCILLATOR CIRCUITS KS57C21832/P21832

#### **POWER CONTROL REGISTER (PCON)**

The power control register, PCON, is a 4-bit register that is used to select the CPU clock frequency and to control CPU operating and power-down modes. PCON can be addressed directly by 4-bit write instructions or indirectly by the instructions IDLE and STOP.

FB3H PC	ON.3 PCON.:	PCON.1	PCON.0
---------	-------------	--------	--------

PCON bits 3 and 2 are addressed by the STOP and IDLE instructions, respectively, to engage the idle and stop power-down modes. Idle and stop modes can be initiated by these instruction despite the current value of the enable memory bank flag (EMB). PCON bits 1 and 0 are used to select a specific system clock frequency. There are two basic choices:

- Main system clock (fx) or subsystem clock (fxt);
- Divided fx/4, 8, 64 or fxt/4 clock frequency.

PCON.1 and PCON.0 settings are also connected with the system clock mode control register, SCMOD. If SCMOD.0 = "0" the main system clock is always selected by the PCON.1 and PCON.0 setting; if SCMOD.0 = "1" the subsystem clock is selected.

RESET sets PCON register values (and SCMOD) to logic zero: SCMOD.3 and SCMOD.0 select the main system clock (fx) and start clock oscillation; PCON.1 and PCON.0 divide the selected fx frequency by 64, and PCON.3 and PCON.2 enable normal CPU operating mode.

Table 6-1. Power Control Register (PCON) Organization

<b>PCON Bit Settings</b>		Resulting CPU Operating Mode	
PCON.3	PCON.2		
0	0	Normal CPU operating mode	
0	1	Idle power-down mode	
1	0	Stop power-down mode	

PCON Bit	t Settings	Resulting CPU (	Clock Frequency
PCON.1	PCON.0	If SCMOD.0 = "0"	If SCMOD.0 = "1"
0	0	fx/64	fxt/4
1	0	fx/8	
1	1	fx/4	

#### PROGRAMMING TIP — Setting the CPU Clock

To set the CPU clock to 0.95 µs at 4.19 MHz:

BITS EMB
SMB 15
LD A,#3H
LD PCON.A



KS57C21832/P21832 OSCILLATOR CIRCUITS

#### **INSTRUCTION CYCLE TIMES**

The unit of time that equals one machine cycle varies depending on whether the main system clock (fx) or a subsystem clock (fxt) is used, and on how the oscillator clock signal is divided (by 4, 8, or 64). Table 6-2 shows corresponding cycle times in microseconds.

Table 6-2. Instruction Cycle Times for CPU Clock Rates

Selected CPU Clock	Resulting Frequency	Oscillation Source	Cycle Time (µsec)
fx/64	65.5 kHz	fx = 4.19  MHz	15.3
fx/8	524.0 kHz		1.91
fx/4	1.05 MHz		0.95
fxt/4	8.19 kHz	fxt = 32.768 kHz	122.0



OSCILLATOR CIRCUITS KS57C21832/P21832

#### SYSTEM CLOCK MODE REGISTER (SCMOD)

The system clock mode register, SCMOD, is a 4-bit register that is used to select the CPU clock and to control main and sub-system clock oscillation. The SCMOD is mapped to the RAM address FB7H.

The main clock oscillation is stopped by setting SCMOD.3 when the clock source is subsystem clock and subsystem clock can be stopped by setting SCMOD.2 when the clock source is main system clock. SCMOD.0, SCMOD.3 cannot be simultaneously modified.

The subsystem clock is stopped only by setting SCMOD.2, and PCON which revokes stop mode cannot stop the subsystem clock. The stop of subsystem clock is released by RESET when the selected system clock is main system clock or subsystem clock and is released by setting SCMOD.2 when the selected system clock is main system clock.

RESET clears all SCMOD values to logic zero, selecting the main system clock (fx) as the CPU clock and starting clock oscillation. The reset value of the SCMOD is "0"

SCMOD.0, SCMOD.2, and SCMOD.3 bits can be manipulated by 1-bit write instructions (In other words, SCMOD.0, SCMOD.2, and SCMOD.3 cannot be modified simultaneously by a 4-bit write). Bit 1 is always logic zero.

FB7H SCMOD.3	SCMOD.2	"0"	SCMOD.0
--------------	---------	-----	---------

A subsystem clock (fxt) can be selected as the system clock by manipulating the SCMOD.3 and SCMOD.0 bit settings. If SCMOD.3 = "0" and SCMOD.0 = "1", the subsystem clock is selected and main system clock oscillation continues. If SCMOD.3 = "1" and SCMOD.0 = "1", fxt is selected, but main system clock oscillation stops.

Even if you have selected fx as the CPU clock, setting SCMOD.3 to "1" will stop main system clock oscillation, and malfunction may be occured. To operate safely, main system clock should be stopped by a stop instruction is main system clock mode.

Table 6-3. System Clock Mode Register (SCMOD) Organization

SCMOD Regist	ter Bit Settings	Resulting Clock Selection		
SCMOD.3	SCMOD.0	<b>CPU Clock Source</b>	fx Oscillation	
0	0	fx	On	
0	1	fxt	On	
1	1	fxt	Off	

SCMOD.2	Sub-oscillation on/off	
0	Enable sub system clock	
1	Disable sub system clock	

**NOTE:** You can use SCMOD.2 as follows (ex; after data bank was used, a few minutes have passed): Main operation  $\rightarrow$  sub-operation  $\rightarrow$  sub-idle (LCD on, after a few minutes later without any external input)  $\rightarrow$  sub-operation  $\rightarrow$  main operation  $\rightarrow$  SCMOD.2 = 1  $\rightarrow$  main stop mode (LCD off).



KS57C21832/P21832 OSCILLATOR CIRCUITS

Table 6-4. Main/Sub Oscillation Stop Mode

Mode	Condition	Method to issue Osc Stop	Osc Stop Release Source (2)
Main Oscillation STOP Mode	Main oscillator runs. Sub oscillator runs (stops). System clock is the main oscillation clock.	STOP instruction: Main oscillator stops. CPU is in idle mode. Sub oscillator still runs (stops).	Interrupt and RESET: After releasing stop mode, main oscillation starts and oscillation stabilization time is elapsed. And then the CPU operates. Oscillation stabilization time is 1/ {256 x BT clock (fx)}.
		When SCMOD.3 is set to "1" <sup>(1)</sup> , main oscillator stops, halting the CPU operation. Sub oscillator still runs (stops).	RESET: Interrupt can't start the main oscillation. Therefore, the CPU operation can never be restarted.
	Main oscillator runs. Sub oscillator runs. System clock is the sub oscillation clock.	STOP instruction (1): Main oscillator stops. CPU is in idle mode. Sub oscillator still runs (stops). Sub oscillator still runs.	BT overflow, interrupt, and RESET: After the overflow of basic timer [1/ {256 x BT clock (fxt)}], CPU operation and main oscillation automatically start.
		When SCMOD.3 is set to "1", main oscillator stops. The CPU, however, would still operate. Sub oscillator still runs.	Set SCMOD.3 to "0" or RESET
Sub Oscillation STOP Mode	Main oscillator runs. Sub oscillator runs. System clock is the main oscillation clock.	When SCMOD.2 to "1", sub oscillator stops, while main oscillator and the CPU would still operate.	Set SCMOD.2 to "0" or RESET
	Main oscillator runs (stops). Sub oscillator runs. System clock is the sub oscillation clock.	When SCMOD.2 to "1", sub oscillator stops, halting the CPU operation. Main oscillator still runs (stops).	RESET

#### NOTES:

- 1. This mode must not be used.
- 2. Oscillation stabilization time by interrupt is 1/(256 x BT clocks). Oscillation stabilization time by a reset is 31.3ms at 4.19Mhz, main oscillation clock.

OSCILLATOR CIRCUITS KS57C21832/P21832

#### SWITCHING THE CPU CLOCK

Together, bit settings in the power control register, PCON, and the system clock mode register, SCMOD, determine whether a main system or a subsystem clock is selected as the CPU clock, and also how this frequency is to be divided. This makes it possible to switch dynamically between main and subsystem clocks and to modify operating frequencies.

SCMOD.3, SCMOD.2, and SCMOD.0 select the main system clock (fx) or a subsystem clock (fxt) and start or stop main system and sub system clock oscillation. PCON.1 and PCON.0 control the frequency divider circuit, and divide the selected fx clock by 4, 8, or 64,or fxt clock by 4.

#### NOTE

A clock switch operation does not go into effect immediately when you make the SCMOD and PCON register modifications — the previously selected clock continues to run for a certain number of machine cycles.

For example, you are using the default CPU clock (normal operating mode and a main system clock of fx/64) and you want to switch from the fx clock to a subsystem clock and to stop the main system clock. To do this, you first need to set SCMOD.0 to "1". This switches the clock from fx to fxt but allows main system clock oscillation to continue. Before the switch actually goes into effect, a certain number of machine cycles must elapse. After this time interval, you can then disable main system clock oscillation by setting SCMOD.3 to "1".

This same 'stepped' approach must be taken to switch from a subsystem clock to the main system clock: first, clear SCMOD.3 to "0" to enable main system clock oscillation. Then, after a certain number of machine cycles has elapsed, select the main system clock by clearing all SCMOD values to logic zero.

Following a RESET, CPU operation starts with the lowest main system clock frequency of 15.3  $\mu$ s at 4.19 MHz after the standard oscillation stabilization interval of 31.3 ms has elapsed. Table 6-5 details the number of machine cycles that must elapse before a CPU clock switch modification goes into effect.

	AFTER	SCMOD.0 = 0					SCMOD.0 = 1	
BEFORE		PCON.1 = 0 PCON.0 = 0		PCON.1 = 1	PCON.0 = 0	PCON.1 = 1	PCON.0 = 1	
	PCON.1 = 0	N/A		1 MACHINE CYCLE		1 MACHINE CYCLE		N/A
	PCON.0 = 0							
SCMOD.0 = 0	PCON.1 = 1	8 MACHINE CYCLES		N/A		1 MACHINE CYCLES		N/A
	PCON.0 = 0							
	PCON.1 = 1	16 MACHINE CYCLES		1 MACHINE CYCLES		N/	Ά	fx / 4fxt
	PCON.0 = 1	-						
SCMOD.0 = 1		N.	/A	N	N/A 1MAC		CYCLES	N/A

Table 6-5. Elapsed Machine Cycles During CPU Clock Switch

#### NOTES:

- 1. Even if oscillation is stopped by setting SCMOD.3 during main system clock operation, the stop mode is not entered.
- 2. Since the X<sub>IN</sub> input is connected internally to V<sub>SS</sub> to avoid current leakage due to the crystal oscillator in stop mode, do not set SCMOD.3 to "1" or do not use stop instruction when an external clock is used as the main system clock.
- 3. When the system clock is switched to the subsystem clock, it is necessary to disable any interrupts which may occur during the time intervals shown in Table 6-5.
- 4. 'N/A' means 'not available'.
- 5. fx: Main-system clock, fxt: Sub-system clock. When fx is 4.19 MHz, and fxt is 32.768 kHz.



KS57C21832/P21832 OSCILLATOR CIRCUITS

#### PROGRAMMING TIP — Switching Between Main System and Subsystem Clock

1. Switch from the main system clock to the subsystem clock:

MA2SUB **BITS** SCMOD.0 ; Switches to subsystem clock DLY80 ; Delay 80 machine cycles CALL BITS SCMOD.3 ; Stop the main system clock RET DLY80 LD A,#0FH DEL1 NOP NOP **DECS** DEL1 JR **RET** 

2. Switch from the subsystem clock to the main system clock:

SUB2MA BITR SCMOD.3 ; Start main system clock oscillation CALL DLY80 ; Delay 160 machine cycles CALL DLY80 BITR SCMOD.0 ; Switch to main system clock RET



OSCILLATOR CIRCUITS KS57C21832/P21832

#### **CLOCK OUTPUT MODE REGISTER (CLMOD)**

The clock output mode register, CLMOD, is a 4-bit register that is used to enable or disable clock output to the CLO pin and to select the CPU clock source and frequency. CLMOD is addressable by 4-bit write instructions only.

FD0H	CLMOD.3	"0"	CLMOD.1	CLMOD.0
------	---------	-----	---------	---------

RESET clears CLMOD to logic zero, which automatically selects the CPU clock as the clock source (without initiating clock oscillation), and disables clock output.

CLMOD.3 is the enable/disable clock output control bit; CLMOD.1 and CLMOD.0 are used to select one of four possible clock sources and frequencies: normal CPU clock, fxx/8, fxx/16, or fxx/64.

Table 6-6. Clock Output Mode Register (CLMOD) Organization

CLMOD B	it Settings	Resulting Clock Output		
CLMOD.1	CLMOD.0	Clock Source	Frequency	
0	0	CPU clock (fx/4, fx/8, fx/64, fxt/4)	1.05 MHz, 524 kHz, 65.5 kHz, 8.2 kHz	
0	1	fxx/8	524 kHz	
1	0	fxx/16	262 kHz	
1	1	fxx/64	65.5 kHz	

CLMOD.3	Result of CLMOD.3 Setting	
0	Disable clock output at the CLO pin	
1	Enable clock output at the CLO pin	

**NOTE**: Frequencies assume that fxx, fx = 4.19 MHz and fxt = 32.768 kHz.



KS57C21832/P21832 OSCILLATOR CIRCUITS

#### **CLOCK OUTPUT CIRCUIT**

The clock output circuit, used to output clock pulses to the CLO pin, has the following components:

- 4-bit clock output mode register (CLMOD)
- Clock selector
- Output latch
- Port mode flag
- CLO output pin (P2.0)

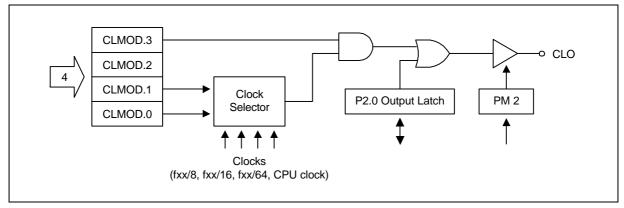


Figure 6-7. CLO Output Pin Circuit Diagram

#### **CLOCK OUTPUT PROCEDURE**

The procedure for outputting clock pulses to the CLO pin may be summarized as follows:

- 1. Disable clock output by clearing CLMOD.3 to logic zero.
- 2. Set the clock output frequency (CLMOD.1, CLMOD.0).
- 3. Load a "0" to the output latch of the CLO pin (P2.0).
- 4. Set the P2.0 mode flag (PM2.0) to output mode.
- 5. Enable clock output by setting CLMOD.3 to logic one.

#### PROGRAMMING TIP — CPU Clock Output to the CLO Pin

To output the CPU clock to the CLO pin:

BITS	EMB		
SMB	15		
LD	EA,#10H		
LD	PMG1,EA	;	P2.0 ← Output mode
BITR	P2.0	;	Clear P2.0 output latch
LD	A,#9H		
LD	CLMOD,A		



OSCILLATOR CIRCUITS KS57C21832/P21832

#### **NOTES**



# 7 INTERRUPTS

# **OVERVIEW**

The KS57C21832 interrupt control circuit has five functional components:

- Interrupt enable flags (IEx)
- Interrupt request flags (IRQx)
- Interrupt master enable register (IME)
- Interrupt priority register (IPR)
- Power-down release signal circuit

Three kinds of interrupts are supported:

- Internal interrupts generated by on-chip processes
- External interrupts generated by external peripheral devices
- Quasi-interrupts used for edge detection and as clock sources

Table 7-1. Interrupt Types and Corresponding Port Pin(s)

Interrupt Type	Interrupt Name	Corresponding Port Pins
External interrupts	INTO, INT1, INT4, INTK	P1.0, P1.1, P1.3, K0-K7
Internal interrupts	INTB, INTT0, INTT1, INTS	Not applicable
Quasi-interrupts	INT2	P1.2
	INTW	Not applicable



#### **Vectored Interrupts**

Interrupt requests may be processed as vectored interrupts in hardware, or they can be generated by program software. A vectored interrupt is generated when the following flags and register settings, corresponding to the specific interrupt (INTn) are set to logic one:

- Interrupt enable flag (IEx)
- Interrupt master enable flag (IME)
- Interrupt request flag (IRQx)
- Interrupt status flags (IS0, IS1)
- Interrupt priority register (IPR)

If all conditions are satisfied for the execution of a requested service routine, the start address of the interrupt is loaded into the program counter and the program starts executing the service routine from this address.

EMB and ERB flags for RAM memory banks and registers are stored in the vector address area of the ROM during interrupt service routines. The flags are stored at the beginning of the program with the VENT instruction. The initial flag values determine the vectors for resets and interrupts. Enable flag values are saved during the main routine, as well as during service routines. Any changes that are made to enable flag values during a service routine are not stored in the vector address.

When an interrupt occurs, the EMB and the ERB flags before the interrupt is initiated are saved along with the program status word (PSW), and the EMB and the ERB flag for the interrupt are fetched from the respective vector address. Then, if necessary, you can modify the enable flags during the interrupt service routine. When the interrupt service routine is returned to the main routine by the IRET instruction, the original values saved in the stack are restored and the main program continues program execution with these values.

#### **Software-Generated Interrupts**

To generate an interrupt request from software, the program manipulates the appropriate IRQx flag. When the interrupt request flag value is set, it is retained until all other conditions for the vectored interrupt have been met, and the service routine can be initiated.

#### **Multiple Interrupts**

By manipulating the two interrupt status flags (IS0 and IS1), you can control service routine initialization and thereby process multiple interrupts simultaneously.

If more than four interrupts are being processed at one time, you can avoid possible loss of working register data by using the PUSH RR instruction to save register contents to the stack before the service routines are executed in the same register bank. When the routines have executed successfully, you can restore the register contents from the stack to working memory by using the POP instruction.

#### **Power-Down Mode Release**

An interrupt can be used to release power-down mode (stop or idle). Interrupts for power-down mode release are initiated by setting the corresponding interrupt enable flag. Even if the IME flag is cleared to zero, power-down mode will be released by an interrupt request signal when the interrupt enable flag has been set. In such cases, the interrupt routine will not be executed since IME = "0".



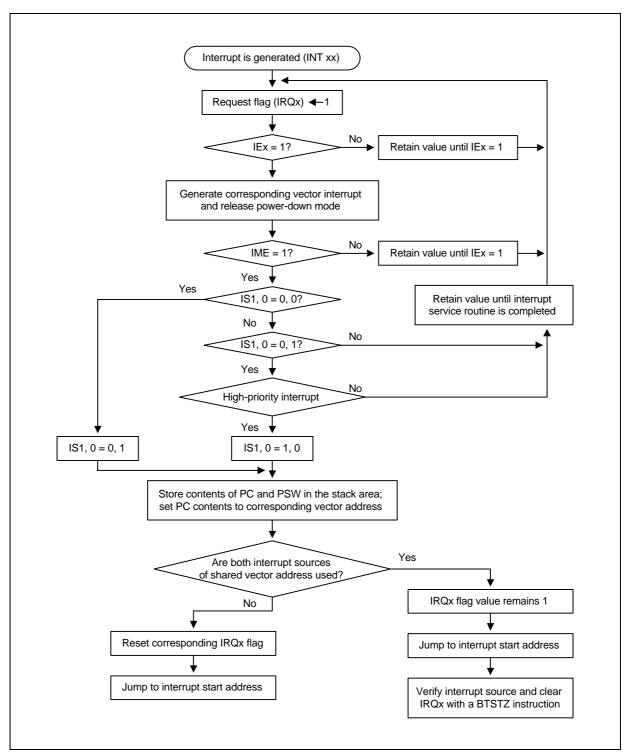


Figure 7-1. Interrupt Execution Flowchart



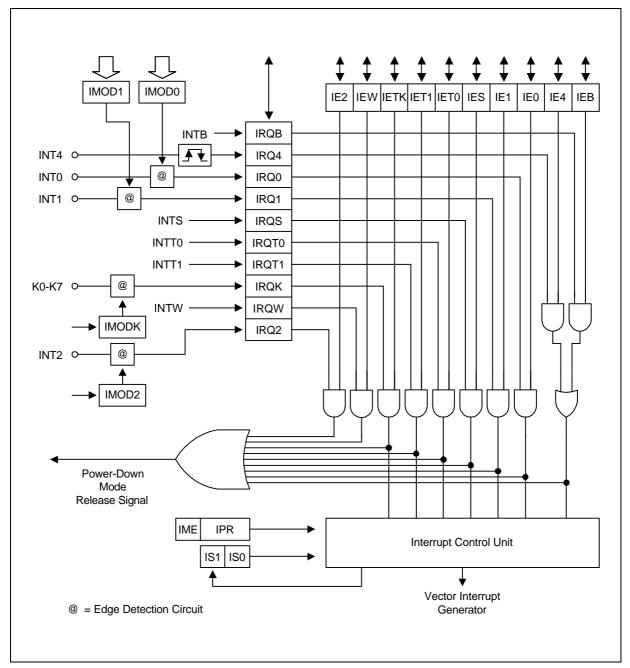


Figure 7-2. Interrupt Control Circuit Diagram

#### **MULTIPLE INTERRUPTS**

The interrupt controller can service multiple interrupts in two ways: as two-level interrupts, where either all interrupt requests or only those of highest priority are serviced, or as multi-level interrupts, when the interrupt service routine for a lower-priority request is accepted during the execution of a higher priority routine.

#### **Two-Level Interrupt Handling**

Two-level interrupt handling is the standard method for processing multiple interrupts. When the IS1 and IS0 bits of the PSW (FB0H.3 and FB0H.2, respectively) are both logic zero, program execution mode is normal and all interrupt requests are serviced (see Figure 7-3).

Whenever an interrupt request is accepted, IS1 and IS0 are incremented by one and the values are stored in the stack along with the other PSW bits. After the interrupt routine has been serviced, the modified IS1 and IS0 values are automatically restored from the stack by an IRET instruction.

ISO and IS1 can be manipulated directly by 1-bit write instructions, regardless of the current value of the enable memory bank flag (EMB). Before you can modify an interrupt service flag, however, you must first disable interrupt processing with a DI instruction.

When IS1 = "0" and IS0 = "1", all interrupt service routines are inhibited except for the highest priority interrupt currently defined by the interrupt priority register (IPR).

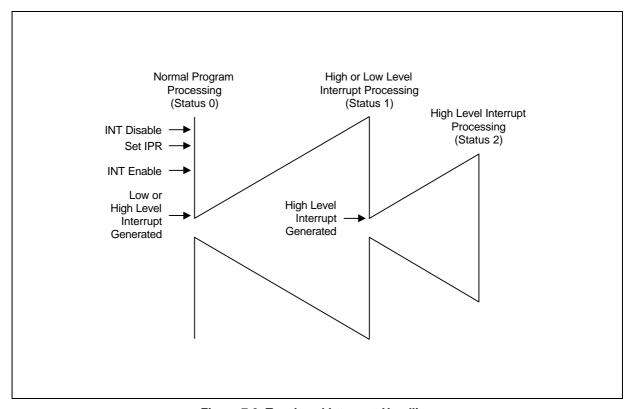


Figure 7-3. Two-Level Interrupt Handling



#### **Multi-Level Interrupt Handling**

With multi-level interrupt handling, a lower-priority interrupt request can be executed while a high-priority interrupt is being serviced. This is done by manipulating the interrupt status flags, ISO and IS1 (see Table 7-2).

When an interrupt is requested during normal program execution, interrupt status flags ISO and IS1 are set to "1" and "0", respectively. This setting allows only highest-priority interrupts to be serviced. When a high-priority request is accepted, both interrupt status flags are then cleared to "0" by software so that a request of any priority level can be serviced. In this way, the high- and low-priority requests can be serviced in parallel (see Figure 7-4).

Process Status	Before INT		Effect of ISx Bit Setting	After INT ACK	
	IS1	IS0		IS1	IS0
0	0	0	All interrupt requests are serviced.	0	1
1	0	1	Only high-priority interrupts as determined by the current settings in the IPR register are serviced.	1	0
2	1	0	No additional interrupt requests will be serviced.	_	_
_	1	1	Value undefined	_	_

Table 7-2. IS1 and IS0 Bit Manipulation for Multi-Level Interrupt Handling

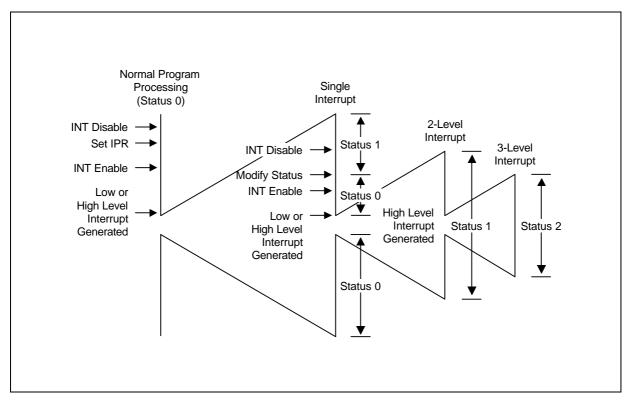


Figure 7-4. Multi-Level Interrupt Handling



#### **INTERRUPT PRIORITY REGISTER (IPR)**

The 4-bit interrupt priority register (IPR) is used to control multi-level interrupt handling. Its reset value is logic zero. Before the IPR can be modified by 4-bit write instructions, all interrupts must first be disabled by a DI instruction.

FB2H	IME	IPR.2	IPR.1	IPR.0

By manipulating the IPR settings, you can choose to process all interrupt requests with the same priority level, or you can select one type of interrupt for high-priority processing. A low-priority interrupt can itself be interrupted by a high-priority interrupt, but not by another low-priority interrupt. A high-priority interrupt cannot be interrupted by any other interrupt source.

Table 7-3. Standard Interrupt Priorities

Interrupt	Default Priority
INTB, INT4	1
INT0	2
INT1	3
INTS	4
INTT0	5
INTT1	6
INTK	7

The MSB of the IPR, the interrupt master enable flag (IME), enables and disables all interrupt processing. Even if an interrupt request flag and its corresponding enable flag are set, a service routine cannot be executed until the IME flag is set to logic one. The IME flag can be directly manipulated by EI and DI instructions, regardless of the current enable memory bank (EMB) value.

**Table 7-4. Interrupt Priority Register Settings** 

IPR.2	IPR.1	IPR.0	Result of IPR Bit Setting
0	0	0	Normal interrupt handling according to default priority settings (note)
0	0	1	Process INTB and INT4 interrupts at highest priority
0	1	0	Process INT0 interrupts at highest priority
0	1	1	Process INT1 interrupts at highest priority
1	0	0	Process INTS interrupts at highest priority
1	0	1	Process INTT0 interrupts at highest priority
1	1	0	Process INTT1 interrupts at highest priority
1	1	1	Process INTK interrupts at highest priority

NOTE: During normal interrupt processing, interrupts are processed in the order in which they occur. If two or more interrupt requests are received simultaneously, the priority level is determined according to the standard interrupt priorities in Table 7-3 (the default priority assigned by hardware when the lower three IPR bits = "0"). In this case, the higher-priority interrupt request is serviced and the other interrupt is inhibited. Then, when the high-priority interrupt is returned from its service routine by an IRET instruction, the inhibited service routine is started.



# PROGRAMMING TIP — Setting the INT Interrupt Priority

The following instruction sequence sets the INT1 interrupt to high priority:

BITS EMB SMB 15

DI ; IPR.3 (IME)  $\leftarrow$  0 LD A,#3H

LD IPR,A

EI ; IPR.3 (IME)  $\leftarrow$  1

### EXTERNAL INTERRUPT 0, 1 AND 2 MODE REGISTERS (IMOD0, IMOD1 AND IMOD2)

The following components are used to process external interrupts at the INT0, INT1 and INT2 pins:

- Edge detection circuit
- Three mode registers, IMOD0, IMOD1 and IMOD2

The mode registers are used to control the triggering edge of the input signal. IMOD0, IMOD1 and IMOD2 settings let you choose either the rising or falling edge of the incoming signal as the interrupt request trigger. The INT4 interrupt is an exception since its input signal generates an interrupt request on both rising and falling edges. Since INT2 is a qusi-interrupt, the interrupt request flag (IRQ2) must be cleared by software.

FB4H	"0"	"0"	IMOD0.1	IMOD0.0
FB5H	"0"	"0"	"0"	IMOD1.0
	`			
FDAH	"0"	"0"	"0"	IMOD2.0

IMOD0, IMOD1 and IMOD2 are addressable by 4-bit write instructions. RESET clears all IMOD values to logic zero, selecting rising edges as the trigger for incoming interrupt requests.

Table 7-5. IMOD0, 1 and 2 Register Organization

IMOD0	0	0	IMOD0.1	IMOD0.0	Effect of IMOD0 Settings
			0	0	Rising edge detection
	0		0	1	Falling edge detection
			1	0	Both rising and falling edge detection
			1	1	IRQ0 flag cannot be set to "1"
IMOD1 IMOD2	0	0	0	IMOD1.0 IMOD2.0	Effect of IMOD1 and IMOD2 Settings
				0	Rising edge detection
				1	Falling edge detection



### **EXTERNAL INTERRUPT 0, 1 and 2 MODE REGISTERS (Continued)**

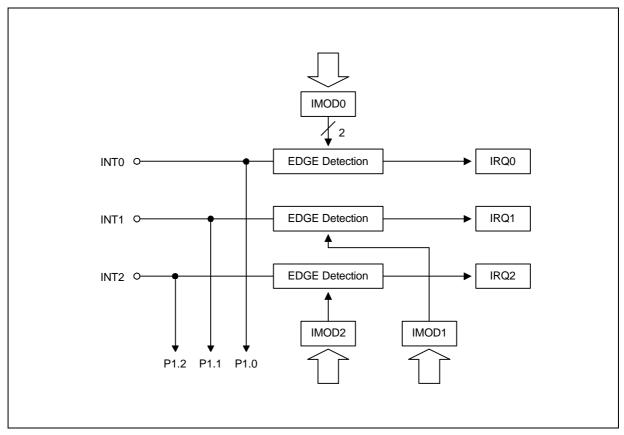


Figure 7-5. Circuit Diagram for INT0, INT1 and INT2 Pins

When modifying the IMOD registers, it is possible to accidentally set an interrupt request flag. To avoid unwanted interrupts, take these precautions when writing your programs:

- 1. Disable all interrupts with a DI instruction.
- 2. Modify the IMOD register.
- 3. Clear all relevant interrupt request flags.
- 4. Enable the interrupt by setting the appropriate IEx flag.
- 5. Enable all interrupts with an El instructions.

### **EXTERNAL KEY INTERRUPT MODE REGISTER (IMODK)**

The mode register for external key interrupts at the K0-K7 pins, IMODK, is addressable only by 4-bit write instructions. RESET clears all IMODK bits to logic zero.

FB6H "0"	IMODK.2	IMODK.1	IMODK.0
----------	---------	---------	---------

Rising or falling edge can be detected by bit IMODK.2 settings. If a rising or falling edge is detected at any one of the selected K pin by the IMODK register, the IRQK flag is set to logic one and a release signal for power-down mode is generated.

# Table 7-6. IMODK Register Bit Settings

IIVIODN	
---------	--

0	IMODK.2	IMODK.1	IMODK.0	Effect of IMODK Settings
	0, 1	0	0	Disable key interrupt
		0	1	Enable edge detection at the K0-K3 pins
		1	0	Enable edge detection at the K4-K7 pins
		1	1	Enable edge detection at the K0-K7 pins

IM	O	D	K.	.2

0	Falling edge detection
1	Rising edge detection

#### NOTES:

- 1. To generate a key interrupt, the selected pins must be configured to input mode. If any one pin of the selected pins is configured to output mode, only falling edge can be detected.
- 2. To generate a key interrupt, all of the selected pins must be at input high state for falling edge detection, or all of the selected pins must be at input low state for rising edge detection. If any one of them or more is at input low state or input high state, the interrupt may be not occurred at falling edge or rising edge.
- 3. To generate a key interrupt, first, configure pull-up resistors or external pull-down resistors. And then, select edge detection and pins by setting IMODK register.



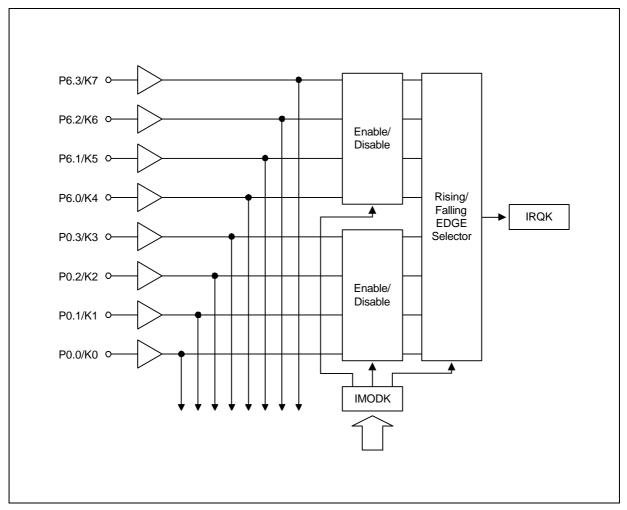


Figure 7-6. Circuit Diagram for INTK

# PROGRAMMING TIP — Using INTK as a Key Input Interrupt

When the key interrupt is used, the selected key interrupt source pin must be set to input:

1. When K0-K7 are selected (eight pins):

BITS EMB
SMB 15
LD A,#3H

LD IMODK,A ; (IMODK)  $\leftarrow$  #3H, K0-K7 falling edge select

LD EA,#41H

LD PUMOD1,EA ; Enable P0 and P6 pull-up resistors

2. When K0-K3 are selected (four pins):

BITS EMB SMB 15 LD A,#1H

LD IMODK,A ; (IMODK)  $\leftarrow$  #1H, K0-K3 falling edge select

LD EA,#00H

LD PUMOD1,EA ; Enable P0 pull-up resistors

#### **INTERRUPT FLAGS**

There are three types of interrupt flags: interrupt request and interrupt enable flags that correspond to each interrupt, the interrupt master enable flag, which enables or disables all interrupt processing.

#### Interrupt Master Enable Flag (IME)

The interrupt master enable flag, IME, enables or disables all interrupt processing. Therefore, even when an IRQx flag is set and its corresponding IEx flag is enabled, the interrupt service routine is not executed until the IME flag is set to logic one.

The IME flag is located in the IPR register (IPR.3). It can be directly be manipulated by EI and DI instructions, regardless of the current value of the enable memory bank flag (EMB).

IME	IPR.2	IPR.1	IPR.0	Effect of Bit Settings
0		,		Inhibit all interrupts
1				Enable all interrupts

### Interrupt Enable Flags (IEx)

IEx flags, when set to logical one, enable specific interrupt requests to be serviced. When the interrupt request flag is set to logical one, an interrupt will not be serviced until its corresponding IEx flag is also enabled.

Interrupt enable flags can be read, written, or tested directly by 1-bit instructions. IEx flags can be addressed directly at their specific RAM addresses, despite the current value of the enable memory bank (EMB) flag.

	•	•		
Address	Bit 3	Bit 2	Bit 1	Bit 0
FB8H	IE4	IRQ4	IEB	IRQB
FBAH	"0"	"0"	IEW	IRQW
FBBH	IEK	IRQK	IET1	IRQT1
FBCH	"0"	"0"	IET0	IRQT0
FBDH	"0"	"0"	IES	IRQS
FBEH	IE1	IRQ1	IE0	IRQ0
FBFH	"0"	"0"	IE2	IRQ2

Table 7-7. Interrupt Enable and Interrupt Request Flag Addresses

#### NOTES:

- 1. IEx refers to all interrupt enable flags.
- IRQx refers to all interrupt request flags.
- 3. IEx = 0 is interrupt disable mode.
- 4. IEx = 1 is interrupt enable mode.

#### Interrupt Request Flags (IRQx)

Interrupt request flags are read/write addressable by 1-bit or 4-bit instructions. IRQx flags can be addressed directly at their specific RAM addresses, regardless of the current value of the enable memory bank (EMB) flag.

When a specific IRQx flag is set to logic one, the corresponding interrupt request is generated. The flag is then automatically cleared to logic zero when the interrupt has been serviced. Exceptions are the watch timer interrupt request flags, IRQW, and the external interrupt 2 flag IRQ2, which must be cleared by software after the interrupt service routine has executed. IRQx flags are also used to execute interrupt requests from software. In summary, follow these guidelines for using IRQx flags:

- 1. IRQx is set to request an interrupt when an interrupt meets the set condition for interrupt generation.
- 2. IRQx is set to "1" by hardware and then cleared by hardware when the interrupt has been serviced (with the exception of IRQW and IRQ2).
- 3. When IRQx is set to "1" by software, an interrupt is generated.

When two interrupts share the same service routine start address, interrupt processing may occur in one of two ways:

- When only one interrupt is enabled, the IRQx flag is cleared automatically when the interrupt has been serviced.
- When two interrupts are enabled, the request flag is not automatically cleared so that the user has an opportunity to locate the source of the interrupt request. In this case, the IRQx setting must be cleared manually using a BTSTZ instruction.

Table 7-8. Interrupt Request Flag Conditions and Priorities

Interrupt Source	Internal / External	Pre-condition for IRQx Flag Setting	Interrupt Priority	IRQ Flag Name
INTB	1	Reference time interval signal from basic timer	1	IRQB
INT4	Е	Both rising and falling edges detected at INT4	1	IRQ4
INT0	Е	Rising or falling edge detected at INT0 pin	2	IRQ0
INT1	Е	Rising or falling edge detected at INT1 pin	3	IRQ1
INTS	I	Completion signal for serial transmit-and-receive or receive-only operation	4	IRQS
INTT0	I	Signals for TCNT0 and TREF0 registers match	5	IRQT0
INTT1	I	Signals for TCNT1 and TREF1 registers match	6	IRQT1
INTK	E	When a rising or falling edge detected at any one of the K0-K7 pins	7	IRQK
INT2	Е	Rising or falling edge detected at INT2	_	IRQ2
INTW	I	Time interval of 0.5 secs or 3.19 msecs	_	IRQW



# PROGRAMMING TIP — Enabling the INTB and INT4 Interrupts

To simultaneously enable INTB and INT4 interrupts:

INTB	DI BTSTZ JR • • EI IRET	IRQB INT4	;	IRQB = 1? If no, INT4 interrupt; if yes, INTB interrupt is processed
; INT4	BITR  •  •  EI IRET	IRQ4	;	INT4 is processed

# **NOTES**



KS57C21832/P21832 POWER-DOWN

# 8 POWER-DOWN

#### **OVERVIEW**

The KS57C21832 microcontroller has two power-down modes to reduce power consumption: idle and stop. Idle mode is initiated by the IDLE instruction and stop mode by the instruction STOP. (Several NOP instructions must always follow an IDLE or STOP instruction in a program.) In idle mode, the CPU clock stops while peripherals and the oscillation source continue to operate normally.

When RESET occurs during normal operation or during a power-down mode, a reset operation is initiated and the CPU enters idle mode. When the standard oscillation stabilization time interval (31.3 ms at 4.19 MHz) has elapsed, normal CPU operation resumes.

In stop mode, main system clock oscillation is halted (assuming it is currently operating), and peripheral hardware components are powered-down. The effect of stop mode on specific peripheral hardware components — CPU, basic timer, serial I/O, timer/counters 0 and 1, watch timer, and LCD controller — and on external interrupt requests, is detailed in Table 8-1.

Idle or stop modes are terminated either by a RESET, or by an interrupt which is enabled by the corresponding interrupt enable flag, IEx. When power-down mode is terminated by RESET, a normal reset operation is executed. Assuming that both the interrupt enable flag and the interrupt request flag are set to "1", power-down mode is released immediately upon entering power-down mode.

When an interrupt is used to release power-down mode, the operation differs depending on the value of the interrupt master enable flag (IME):

- If the IME flag = "0"; If the power down mode release signal is generated, after releasing the power-down mode, program execution starts immediately under the instruction to enter power down mode without execution of interrupt service routine. The interrupt request flag remains set to logic one.
- If the IME flag = "1"; If the power down mode release signal is generated, after releasing the power down mode, two instructions following the instruction to enter power down mode are executed first and the interrupt service routine is executed, finally program is resumed.
  However, when the release signal is caused by INT2 or INTW, the operation is identical to the IME = "0" condition because INT2 and INTW are a quasi-interrupt.

### **NOTE**

Do not use stop mode if you are using an external clock source because  $X_{\text{IN}}$  input must be restricted internally to  $V_{\text{SS}}$  to reduce current leakage.



POWER-DOWN KS57C21832/P21832

Table 8-1. Hardware Operation During Power-Down Modes

Operation	Stop Mode (STOP)	Idle Mode (IDLE)
System clock status	STOP mode can be used only if the main system clock is selected as system clock (CPU clock)	Idle mode can be used if the main system clock or subsystem clock is selected as system clock (CPU clock)
Clock oscillator	Main system clock oscillation stops	CPU clock oscillation stops (main and subsystem clock oscillation continues)
Basic timer	Basic timer stops	Basic timer operates (with IRQB set at each reference interval)
Serial I/O interface	Operates only if external SCK input is selected as the serial I/O clock	Operates if a clock other than the CPU clock is selected as the serial I/O clock
Timer/counter 0	Operates only if TCL0 is selected as the counter clock	Timer/counter 0 operates
Timer/counter 1	Operates only if TCL1 is selected as the counter clock	Timer/counter 1 operates
Watch timer	Operates only if subsystem clock (fxt) is selected as the counter clock	Watch timer operates
LCD controller	Operates only if a subsystem clock is selected as LCDCK	LCD controller operates
External interrupts	INTO, INT1, INT2, INT4, and INTK are acknowledged	INT0, INT1, INT2, INT4 and INTK are acknowledged
CPU	All CPU operations are disabled	All CPU operations are disabled
Mode release signal	Interrupt request signals are enabled by an interrupt enable flag or by RESET input	Interrupt request signals are enabled by an interrupt enable flag or by RESET input



KS57C21832/P21832 POWER-DOWN

**Table 8-2. System Operating Mode Comparison** 

Mode	Condition	STOP/IDLE Mode Start Method	Current Consumption
Main operating mode	Main oscillator runs. Sub oscillator runs (stops). System clock is the main oscillation clock.	_	A
Main Idle mode	Main oscillator runs. Sub oscillator runs (stops). System clock is the main oscillation clock.	IDLE instruction	В
Main Stop mode	Main oscillator runs. Sub oscillator runs. System clock is the main oscillation clock.	STOP instruction	D
Sub operating mode	Main oscillator is stopped by SCMOD.3. Sub oscillator runs. System clock is the sub oscillation clock.	_	C
Sub Idle Mode	Main oscillator is stopped by SCMOD.3. Sub oscillator runs. System clock is the sub oscillation clock.	IDLE instruction	D
Sub Stop mode	Main oscillator is stopped by SCMOD.3. Sub oscillator runs. System clock is the sub oscillation clock.	Setting SCMOD.2 to "1": This mode can be released only by an external RESET.	E
Main/Sub Stop mode	Main oscillator runs. Sub oscillator is stopped by SCMOD.2. System clock is the main oscillation clock.	STOP instruction: This mode can be released by an interrupt and RESET.	E

**NOTE:** The current consumption is: A > B > C > D > E.

POWER-DOWN KS57C21832/P21832

### **IDLE MODE TIMING DIAGRAMS**

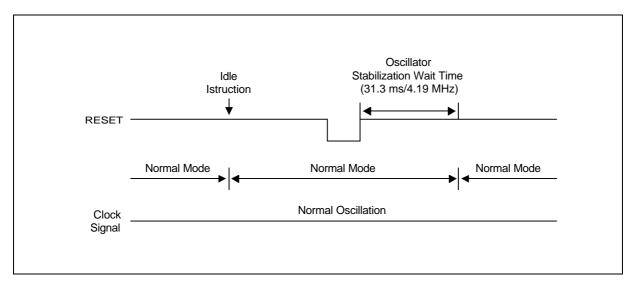


Figure 8-1. Timing When Idle Mode is Released by RESET

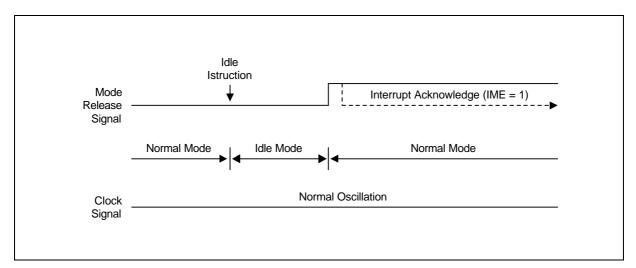


Figure 8-2. Timing When Idle Mode is Released by an Interrupt



KS57C21832/P21832 POWER-DOWN

### **STOP MODE TIMING DIAGRAMS**

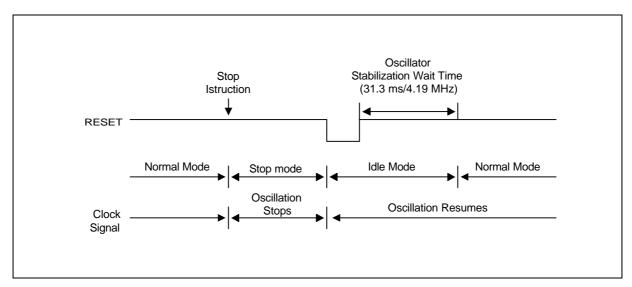


Figure 8-3. Timing When Stop Mode is Released by RESET

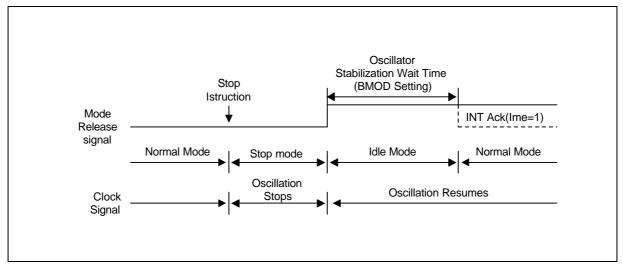


Figure 8-4. Timing When Stop Mode is Released by an Interrupt

POWER-DOWN KS57C21832/P21832

# PROGRAMMING TIP — Reducing Power Consumption for Key Input Interrupt Processing

The following code shows real-time clock and interrupt processing for key inputs to reduce power consumption. In this example, the system clock source is switched from the main system clock to a subsystem clock and the LCD display is turned on:

KEYCLK	DI			
	CALL	MA2SUB	; Main system $\rightarrow$ clock subsystem clock switch subrouting	ne
	SMB	15		
	LD	EA,#00H		
	LD	P4,EA	; All key strobe outputs to low level	
	LD	A,#3H	2	
	LD	IMODK,A	; Select K0-K7 enable	
	SMB	0		
	BITR BITR	IRQW IRQK		
	BITS	IEW		
	BITS	IEK		
CLKS1	CALL	WATDIS	; Execute clock and display changing subroutine	
	BTSTZ	IRQK	,	
	JR	CIDLE		
	CALL	SUB2MA	; Subsystem clock → main system clock switch subrouti	ne
	EI			
	RET			
CIDLE	IDLE		; Engage idle mode	
	NOP			
	NOP			
	JPS	CLKS1		



KS57C21832/P21832 POWER-DOWN

# RECOMMENDED CONNECTIONS FOR UNUSED PINS

To reduce overall power consumption, please configure unused pins according to the guidelines described in Table 8-3.

Table 8-3. Unused Pin Connections for Reducing Power Consumption

Pin/Share Pin Names	Recommended Connection
P0.0/SCK/K0 P0.1/SO/K1 P0.2/SI/K2 P0.3/BUZ/K3	Input mode: Connect to V <sub>DD</sub> Output mode: No connection
P1.0/INT0-P1.2/INT2	Connect to V <sub>DD</sub>
P1.3/INT4	Connect to V <sub>DD</sub>
P2.0/CLO P2.1/LCDCK P2.2/LCDSY P3.0/TCLO0 P3.1/TCLO1 P3.2/TCL0 P3.3/TCL1 P4.0/COM8-P4.3/COM11 P5.0/COM12-P5.3/COM15 P6.0/SEG55/K4-P6.3/ SEG52/K7 P7.0/SEG51-P7.3/SEG48 P8.0/SEG47-P8.3/SEG44 P9.0/SEG43-P9.3/SEG40	Input mode: Connect to V <sub>DD</sub> Output mode: No connection
SEG0-SEG39 COM0-COM7	No connection
V <sub>LC1</sub> -V <sub>LC5</sub>	No connection
XT <sub>IN</sub> (note)	Stop sub-oscillator by setting the SCMOD.2 to logic "1"
XT <sub>OUT</sub>	No connection
TEST	Connect to V <sub>SS</sub>

**NOTE:** You can stop the sub-oscillator by setting the SCMOD.2 to one.



POWER-DOWN KS57C21832/P21832

# **NOTES**



KS57C21832/P21832 RESET

# 9 RESET

#### **OVERVIEW**

When a RESET signal is input during normal operation or power-down mode, a hardware reset operation is initiated and the CPU enters idle mode. Then, when the standard oscillation stabilization interval of 31.3 ms at 4.19 MHz has elapsed, normal system operation resumes.

Regardless of when the RESET occurs — during normal operating mode or during a power-down mode — most hardware register values are set to the reset values described in Table 9-1. The current status of several register values is, however, always retained when a RESET occurs during idle or stop mode; If a RESET occurs during normal operating mode, their values are undefined. Current values that are retained in this case are as follows:

- Carry flag
- Data memory values
- General-purpose registers E, A, L, H, X, W, Z, and Y
- Serial I/O buffer register (SBUF)

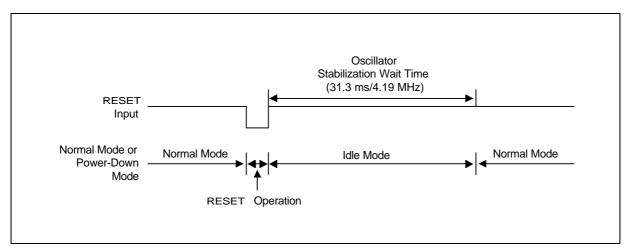


Figure 9-1. Timing for Oscillation Stabilization after RESET

### HARDWARE REGISTER VALUES AFTER RESET

Table 9-1 gives you detailed information about hardware register values after a RESET occurs during power-down mode or during normal operation.



RESET KS57C21832/P21832

Table 9-1. Hardware Register Values after RESET

Hardware Component or Subcomponent	If RESET Occurs During Power-Down Mode	If RESET Occurs During Normal Operation	
Program counter (PC)	Lower six bits of address 0000H are transferred to PC13-8, and the contents of 0001H to PC7-0.	Lower six bits of address 0000H are transferred to PC13-8, and the contents of 0001H to PC7-0.	
Program Status Word (PSW):			
Carry flag (C)	Retained	Undefined	
Skip flag (SC0-SC2)	0	0	
Interrupt status flags (IS0, IS1)	0	0	
Bank enable flags (EMB, ERB)	Bit 6 of address 0000H in program memory is transferred to the ERB flag, and bit 7 of the address to the EMB flag.	Bit 6 of address 0000H in program memory is transferred to the ERB flag, and bit 7 of the address to the EMB flag.	
Stack pointer (SP)	Undefined	Undefined	
Data Memory (RAM):			
General registers E, A, L, H, X, W, Z, Y	Values retained	Undefined	
General-purpose registers	Values retained (note)	Undefined	
Bank selection registers (SMB, SRB)	0, 0	0, 0	
BSC register (BSC0-BSC3)	0	0	
Clocks:			
Power control register (PCON)	0	0	
Clock output mode register (CLMOD)	0	0	
System clock mode register (SCMOD)	0	0	
Interrupts:			
Interrupt request flags (IRQx)	0	0	
Interrupt enable flags (IEx)	0	0	
Interrupt priority flag (IPR)	0	0	
Interrupt master enable flag (IME)	0	0	
INT0 mode register (IMOD0)	0	0	
INT1 mode register (IMOD1)	0	0	
INT2 mode register (IMOD2)	0	0	
INTK mode register (IMODK)	0	0	

**NOTE:** The values of the 0F8H-0FDH are not retained when a RESET signal is input.



**KS57C21832/P21832** 

Table 9-1. Hardware Register Values after RESET (Continued)

Hardware Component or Subcomponent	If RESET Occurs During Power-Down Mode	If RESET Occurs During Normal Operation	
I/O Ports:			
Output buffers	Off	Off	
Output latches	0	0	
Port mode flags (PM)	0	0	
Pull-up resistor mode reg (PUMOD1/2)	0	0	
Basic Timer:			
Count register (BCNT)	Undefined	Undefined	
Mode register (BMOD)	0	0	
Mode register (WDMOD)	A5H	A5H	
Counter clear flag (WDTCF)	0	0	
Timer/Counter 0 and 1:			
Count registers (TCNT0/1)	0	0	
Reference registers (TREF0/1)	FFH, FFFFH	FFH, FFFFH	
Mode registers (TMOD0/1)	0	0	
Output enable flags (TOE0/1)	0	0	
Watch Timer:			
Watch timer mode register (WMOD)	0	0	
LCD Driver/Controller:			
LCD mode register (LMOD)	0	0	
LCD control register (LCON)	0	0	
Display data memory	Values retained	Undefined	
Output buffers	Off	Off	
Serial I/O Interface:			
SIO mode register (SMOD)	0	0	
SIO interface buffer (SBUF)	Values retained	Undefined	
N-Channel Open-Drain Mode Register			
PNE0/3	0	0	



RESET KS57C21832/P21832

# **NOTES**



# **10** 1/0 PORTS

### **OVERVIEW**

The KS57C21832 has 10 ports. There are total of 4 input pins and 35 configurable I/O pins, for a maximum number of 39 pins.

Pin addresses for all ports are mapped to bank 15 of the RAM. The contents of I/O port pin latches can be read, written, or tested at the corresponding address using bit manipulation instructions.

### **Port Mode Flags**

Port mode flags (PM) are used to configure I/O ports to input or output mode by setting or clearing the corresponding I/O buffer.

# Pull-up Resistor Mode Register (PUMOD)

The pull-up resistor mode registers (PUMOD1, 2) are used to assign internal pull-up resistors by software to specific ports. When a configurable I/O port pin is used as an output pin, its assigned pull-up resistor is automatically disabled, even though the pin's pull-up is enabled by a corresponding PUMOD bit setting.



I/O PORTS KS57C21832/P21832

Table 10-1. I/O Port Overview

Port	I/O	Pins	Pin Names	Address	Function Description
0	I/O	4	P0.0-P0.3	FF0H	4-bit I/O port.  1-bit and 4-bit read/write and test is possible. Individual pins are software configurable as input or output.  Individual pins are software configurable as open-drain or push-pull output.  4-bit pull-up resistors are software assignable; pull-up resistors are automatically disabled for output pins.
1	I	4	P1.0-P1.3	FF1H	<ul><li>4-bit input port.</li><li>1-bit and 4-bit read and test is possible.</li><li>4-bit pull-up resistors are assignable.</li></ul>
2	I/O	3	P2.0-P2.2	FF2H	Same as port 0 except that port 2 is 3-bit I/O port.
3	I/O	4	P3.0-P3.3	FF3H	Same as port 0.
4, 5	I/O	8	P4.0-P4.3 P5.0-P5.3	FF4H FF5H	<ul> <li>4-bit I/O ports.</li> <li>1-, 4-bit or 8-bit read/write and test is possible.</li> <li>Individual pins are software configurable as input or output.</li> <li>4-bit pull-up resistors are software assignable; pull-up resistors are automatically disabled for output pins.</li> </ul>
6, 7	I/O	8	P6.0-P6.3 P7.0-P7.3	FF6H FF7H	Same as P4 and P5.
8, 9	I/O	8	P8.0-P8.3 P9.0-P9.3	FF8H FF9H	Same as P4 and P5.

Table 10-2. Port Pin Status During Instruction Execution

Instruction Type	Example		Input Mode Status	Output Mode Status
1-bit test 1-bit input 4-bit input 8-bit input	BTST LDB LD LD	P0.1 C,P1.3 A,P7 EA,P4	Input or test data at each pin	Input or test data at output latch
1-bit output	BITR	P2.3	Output latch contents undefined	Output pin status is modified
4-bit output 8-bit output	LD LD	P2,A P6,EA	Transfer accumulator data to the output latch	Transfer accumulator data to the output pin



KS57C21832/P21832 VO PORTS

#### **PORT MODE FLAGS (PM FLAGS)**

Port mode flags (PM) are used to configure I/O ports to input or output mode by setting or clearing the corresponding I/O buffer.

For convenient program reference, PM flags are organized into five groups – PMG1, PMG2, PMG3, PMG4 and PMG5 as shown in Table 10-3. They are addressable by 8-bit write instructions only.

When a PM flag is "0", the port is set to input mode; when it is "1", the port is enabled for output. RESET clears all port mode flags to logical zero, automatically configuring the corresponding I/O ports to input mode.

**Table 10-3. Port Mode Group Flags** 

PM Group ID	Address	Bit 3	Bit 2	Bit 1	Bit 0
PMG1	FE6H	PM0.3	PM0.2	PM0.1	PM0.0
	FE7H	"0"	PM2.2	PM2.1	PM2.0
PMG2	FE8H	PM3.3	PM3.2	PM3.1	PM3.0
	FE9H	"0"	"0"	"0"	"0"
PMG3	FEAH	PM4.3	PM4.2	PM4.1	PM4.0
	FEBH	PM5.3	PM5.2	PM5.1	PM5.0
PMG4	FECH	PM6.3	PM6.2	PM6.1	PM6.0
	FEDH	PM7.3	PM7.2	PM7.1	PM7.0
PMG5	FEEH	PM8.3	PM8.2	PM8.1	PM8.0
	FEFH	PM9.3	PM9.2	PM9.1	PM9.0

NOTE: If bit = "0", the corresponding I/O pin is set to input mode. If bit = "1", the pin is set to output mode: PM0.0 for P0.0, PM0.1 for P0.1, etc., All flags are cleared to "0" following RESET.

## PROGRAMMING TIP — Configuring I/O Ports to Input or Output

Configure ports 0 and 2 as an output port:

BITS EMB

SMB 15

LD EA,#7FH

LD PMG1,EA ; P0 and P2  $\leftarrow$  Output

I/O PORTS KS57C21832/P21832

#### **PULL-UP RESISTOR MODE REGISTER (PUMOD)**

The pull-up resistor mode registers (PUMOD1 and PUMOD2) are used to assign internal pull-up resistors by software to specific ports. When a configurable I/O port pin is used as an output pin, its assigned pull-up resistor is automatically disabled, even though the pin's pull-up is enabled by a corresponding PUMOD bit setting.

PUMOD1 is addressable by 8-bit write instructions only, and PUMOD2 by 4-bit write instruction only. RESET clears PUMOD register values to logic zero, automatically disconnecting all software-assignable port pull-up resistors.

Table 10-4. Pull-Up Resistor Mode Register (PUMOD) Organization

PUMOD ID	Address	Bit 3	Bit 2	Bit 1	Bit 0
PUMOD1	FDCH	PUR3	PUR2	PUR1	PUR0
	FDDH	PUR7	PUR6	PUR5	PUR4
PUMOD2	FDEH	"0"	"0"	PUR9	PUR8

**NOTE:** When bit = "1", a pull-up resistor is assigned to the corresponding I/O port: PUR3 for port 3, PUR2 for port 2, and so on.

# PROGRAMMING TIP — Enabling and Disabling I/O Port Pull-Up Resistors

P6 and P7 enable pull-up resistors.

BITS EMB

SMB 15

LD EA,#0C0H

LD PUMOD1,EA : P6 and P7 enable

# N-CHANNEL OPEN-DRAIN MODE REGISTER (PNE)

The n-channel, open-drain mode register (PNE) is used to configure ports 0, 2 and 3 to n-channel, open-drain or as push-pull outputs. When a bit in the PNE register is set to "1", the corresponding output pin is configured to n-channel, open-drain; when set to "0", the output pin is configured to push-pull. The PNE register consists of an 8-bit register and a 4-bit register; PNE0 can be addressed by 8-bit write instructions only and PNE3 by 4-bit write instructions only.

FD6H	PNE0.3	PNE0.2	PNE0.1	PNE0.0	PNE1
FD7H	PNE2.3	PNE2.2	PNE2.1	PNE2.0	
					Ī
FD8H	PNE3.3	PNE3.2	PNE3.1	PNE3.0	PNE2



### **PORT 0 CIRCUIT DIAGRAM**

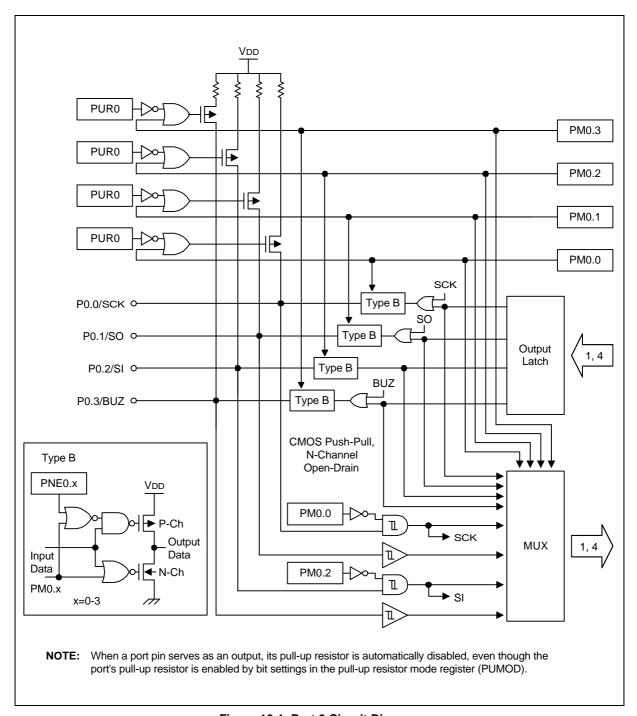


Figure 10-1. Port 0 Circuit Diagram



I/O PORTS KS57C21832/P21832

### **PORT 1 CIRCUIT DIAGRAM**

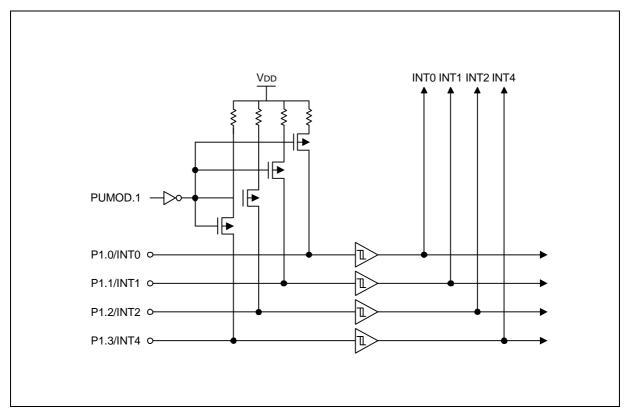


Figure 10-2. Port 1 Circuit Diagram



### **PORT 2 CIRCUIT DIAGRAM**

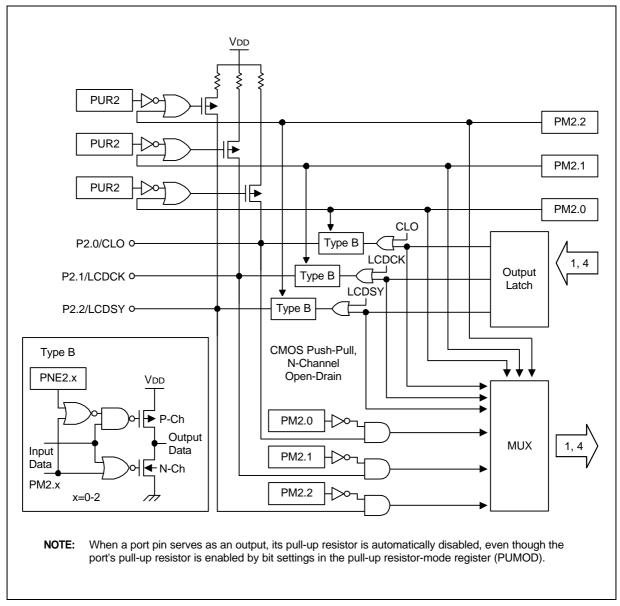


Figure 10-3. Port 2 Circuit Diagram

I/O PORTS KS57C21832/P21832

#### **PORT 3 CIRCUIT DIAGRAM**

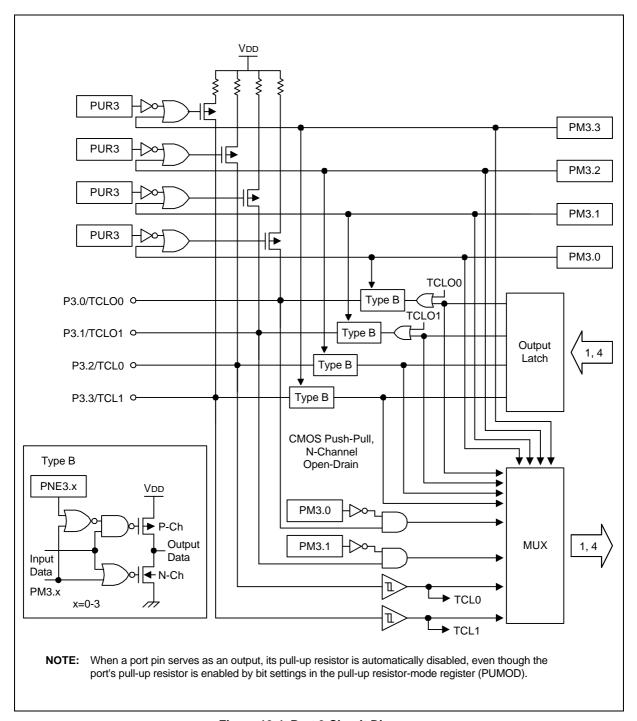


Figure 10-4. Port 3 Circuit Diagram



## **PORT 4, 5, 6, 7, 8, 9 CIRCUIT DIAGRAM**

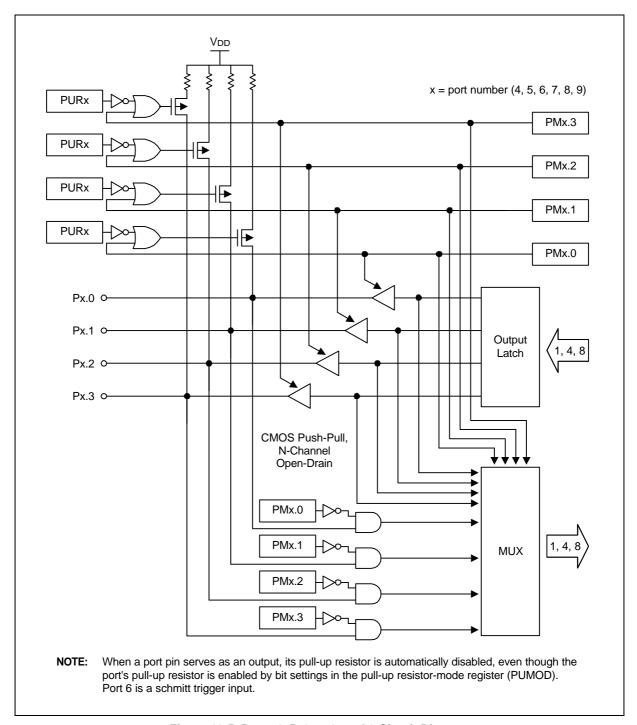


Figure 10-5. Ports 4, 5, 6, 7, 8, and 9 Circuit Diagram



I/O PORTS KS57C21832/P21832

## **NOTES**



11

# **TIMERS and TIMER/COUNTERS**

#### **OVERVIEW**

The KS57C21832 microcontroller has four timer and timer/counter modules:

- 8-bit basic timer (BT)
- 8-bit timer/counter (TC0)
- 16-bit timer/counter (TC1)
- Watch timer (WT)

The 8-bit basic timer (BT) is the microcontroller's main interval timer and watch-dog timer. It generates an interrupt request at a fixed time interval when the appropriate modification is made to its mode register. The basic timer is also used to determine clock oscillation stabilization time when stop mode is released by an interrupt and after a RESET.

The 8-bit timer/counter (TC0) and the 16-bit timer/counter (TC1) are programmable timer/counters that are used primarily for event counting and for clock frequency modification and output. In addition, TC0 generates a clock signal that can be used by the serial I/O interface.

The watch timer (WT) module consists of an 8-bit watch timer mode register, a clock selector, and a frequency divider circuit. Watch timer functions include real-time and watch-time measurement, main and subsystem clock interval timing, buzzer output generation. It also generates a clock signal for the LCD controller.



## **BASIC TIMER (BT)**

#### **OVERVIEW**

The 8-bit basic timer (BT) has six functional components:

- Clock selector logic
- 4-bit mode register (BMOD)
- 8-bit counter register (BCNT)
- 8-bit watchdog timer mode register (WDMOD)
- Watchdog timer counter clear flag (WDTCF)

The basic timer generates interrupt requests at precise intervals, based on the frequency of the system clock. You can use the basic timer as a "watchdog" timer for monitoring system events or use BT output to stabilize clock oscillation when stop mode is released by an interrupt and following RESET. Bit settings in the basic timer mode register BMOD turns the BT module on and off, selects the input clock frequency, and controls interrupt or stabilization intervals.

#### **Interval Timer Function**

The basic timer's primary function is to measure elapsed time intervals. The standard time interval is equal to 256 basic timer clock pulses.

To restart the basic timer, one bit setting is required: bit 3 of the mode register BMOD should be set to logic one. The input clock frequency and the interrupt and stabilization interval are selected by loading the appropriate bit values to BMOD.2-BMOD.0.

The 8-bit counter register, BCNT, is incremented each time a clock signal is detected that corresponds to the frequency selected by BMOD. BCNT continues incrementing as it counts BT clocks until an overflow occurs (≥ 255). An overflow causes the BT interrupt request flag (IRQB) to be set to logic one to signal that the designated time interval has elapsed. An interrupt request is than generated, BCNT is cleared to logic zero, and counting continues from 00H.

#### **Watchdog Timer Function**

The basic timer can also be used as a "watchdog" timer to signal the occurrence of system or program operation error. For this purpose, instruction that clear the watchdog timer (BITS WDTCF) should be executed at proper points in a program within given period. If an instruction that clears the watchdog timer is not executed within the given period and the watchdog timer overflows, reset signal is generated and the system restarts with reset status. An operation of watchdog timer is as follows:

- Write some values (except #5AH) to watchdog timer mode register, WDMOD.
- If WDCNT overflows, system reset is generated.



### **Oscillation Stabilization Interval Control**

Bits 2-0 of the BMOD register are used to select the input clock frequency for the basic timer. This setting also determines the time interval (also referred to as 'wait time') required to stabilize clock signal oscillation when stop mode is released by an interrupt. When a RESET signal is inputted, the standard stabilization interval for system clock oscillation following the RESET is 31.3 ms at 4.19 MHz.

Table 11-1. Basic Timer Register Overview

Register Name	Туре	Description	Size	RAM Address	Addressing Mode	Reset Value
BMOD	Control	Controls the clock frequency (mode) of the basic timer; also, the oscillation stabilization interval after stop mode release or RESET	4-bit	F85H	4-bit write-only; BMOD.3: 1-bit writeable	"0"
BCNT	Counter	Counts clock pulses matching the BMOD frequency setting	8-bit	F86H-F87H	8-bit read-only	U (note)
WDMOD	Control	Controls watchdog timer operation.	8-bit	F98H-F99H	8-bit write-only	A5H
WDTCF	Control	Clears the watchdog timer's counter.	1-bit	F9AH.3	1-, 4-bit write	"0"

NOTE: 'U' means the value is undetermined after a RESET.



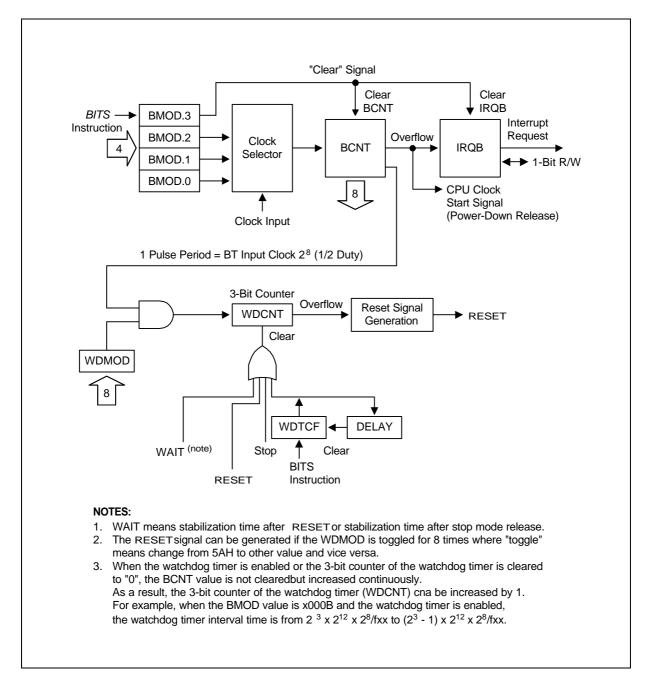


Figure 11-1. Basic Timer Circuit Diagram



### **BASIC TIMER MODE REGISTER (BMOD)**

The basic timer mode register, BMOD, is a 4-bit write-only register. Bit 3, the basic timer start control bit, is also 1-bit addressable. All BMOD values are set to logic zero following RESET and interrupt request signal generation is set to the longest interval. (BT counter operation cannot be stopped.) BMOD settings have the following effects:

- Restart the basic timer;
- Control the frequency of clock signal input to the basic timer;
- Determine time interval required for clock oscillation to stabilize following the release of stop mode by an interrupt.

By loading different values into the BMOD register, you can dynamically modify the basic timer clock frequency during program execution. Four BT frequencies, ranging from fxx/2<sup>12</sup> to fxx/2<sup>5</sup>, are selectable. Since BMOD's reset value is logic zero, the default clock frequency setting is fxx/2<sup>12</sup>.

The most significant bit of the BMOD register, BMOD.3, is used to restart the basic timer. When BMOD.3 is set to logic one by a 1-bit write instruction, the contents of the BT counter register (BCNT) and the BT interrupt request flag (IRQB) are both cleared to logic zero, and timer operation restarts.

The combination of bit settings in the remaining three registers — BMOD.2, BMOD.1, and BMOD.0 — determine the clock input frequency and oscillation stabilization interval.

Table 11-2. Basic Timer Mode Register (BMOD) Organization

BMOD.3	Basic Timer Start Control Bit
1	Start basic timer; clear IRQB, BCNT, and BMOD.3 to "0"

BMOD.2	BMOD.1	BMOD.0
0	0	0
0	1	1
1	0	1
1	1	1

Basic Timer Input Clock	Interrupt Interval Time (Wait Time)
fxx/2 <sup>12</sup> (1.02 kHz)	2 <sup>20</sup> /fxx (250 ms)
fxx/2 <sup>9</sup> (8.18 kHz)	2 <sup>17</sup> /fxx (31.3 ms)
fxx/2 <sup>7</sup> (32.7 kHz)	2 <sup>15</sup> /fxx (7.82 ms)
fxx/2 <sup>5</sup> (131 kHz)	2 <sup>13</sup> /fxx (1.95 ms)

#### NOTES:

- 1. Clock frequencies and interrupt interval time assume a system oscillator clock frequency (fxx) of 4.19 MHz.
- 2. fxx = selected system clock frequency.
- 3. Wait time is the time required to stabilize clock signal oscillation after stop mode is released. The data in the table column "Interrupt Interval Time" can also be interpreted as "Oscillation Stabilization."
- 4. The standard stabilization time for system clock oscillation following a RESET is 31.3 ms at 4.19 MHz.



### **BASIC TIMER COUNTER (BCNT)**

BCNT is an 8-bit counter for the basic timer. It can be addressed by 8-bit read instructions. RESET leaves the BCNT counter value undetermined. BCNT is automatically cleared to logic zero whenever the BMOD register control bit (BMOD.3) is set to "1" to restart the basic timer. It is incremented each time a clock pulse of the frequency determined by the current BMOD bit settings is detected.

When BCNT has incrementing to hexadecimal 'FFH' (≥ 255 clock pulses), it is cleared to '00H' and an overflow is generated. The overflow causes the interrupt request flag, IRQB, to be set to logic one. When the interrupt request is generated, BCNT immediately resumes counting incoming clock signals.

#### **NOTE**

Always execute a BCNT read operation twice to eliminate the possibility of reading unstable data while the counter is incrementing. If, after two consecutive reads, the BCNT values match, you can select the latter value as valid data. Until the results of the consecutive reads match, however, the read operation must be repeated until the validation condition is met.

#### **BASIC TIMER OPERATION SEQUENCE**

The basic timer's sequence of operations may be summarized as follows:

- 1. Set BMOD.3 to logic one to restart the basic timer.
- 2. BCNT is then incremented by one after each clock pulse corresponding to BMOD selection.
- 3. BCNT overflows if BCNT ≥ 255 (BCNT = FFH).
- 4. When an overflow occurs, the IRQB flag is set by hardware to logic one.
- 5. The interrupt request is generated.
- 6. BCNT is then cleared by hardware to logic zero.
- 7. Basic timer resumes counting clock pulses.



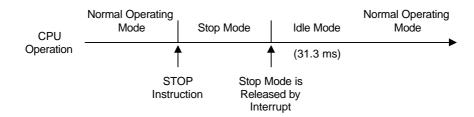
## PROGRAMMING TIP — Using the Basic Timer

1. To read the basic timer count register (BCNT):

	BITS	EMB
	SMB	15
BCNTR	LD	EA,BCNT
	LD	YZ,EA
	LD	EA,BCNT
	CPSE	EA,YZ
	JR	BCNTR

2. When stop mode is released by an interrupt, set the oscillation stabilization interval to 31.3 ms:

BITS	EMB	
SMB	15	
LD	A,#0BH	
LD	BMOD,A	; Wait time is 31.3 ms
NOP		
STOP		; Set stop power-down mode
NOP		
NOP		
NOP		



3. To set the basic timer interrupt interval time to 1.95 ms (at 4.19 MHz):

BITS	EMB	
SMB	15	
LD	A,#0FH	
LD	BMOD,A	
El		
BITS	IEB	: Basic timer interrupt enable flag is set to "1"

4. Clear BCNT and the IRQB flag and restart the basic timer:

BITS	EMB
SMB	15
BITS	BMOD.3



### **WATCHDOG TIMER MODE REGISTER (WDMOD)**

The watchdog timer mode register, WDMOD, is a 8-bit write-only register. WDMOD register controls to enable or disable the watchdog function. WDMOD values are set to logic "A5H" following RESET and this value enables the watchdog timer. Watchdog timer is set to the longest interval because BT overflow signal is generated with the longest interval.

WDMOD	Watchdog Timer Enable/Disable Control
5AH	Disable watchdog timer function
Any other value	Enable watchdog timer function

### **WATCHDOG TIMER COUNTER (WDCNT)**

The watchdog timer counter, WDCNT, is a 3-bit counter. WDCNT is automatically cleared to logic zero, and restarts whenever the WDTCF register control bit is set to "1". RESET, stop, and wait signal clears the WDCNT to logic zero also.

WDCNT increments each time a clock pulse of the overflow frequency determined by the current BMOD bit setting is generated. When WDCNT has incremented to hexadecimal '07H', it is cleared to '00H' and an overflow is generated. The overflow causes the system RESET. When the interrupt request is generated, BCNT immediately resumes counting incoming clock signals.

### WATCHDOG TIMER COUNTER CLEAR FLAG (WDTCF)

The watchdog timer counter clear flag, WDTCF, is a 1-bit write instruction. When WDTCF is set to one, it clears the WDCNT to zero and restarts the WDCNT. WDTCF register bits 2-0 are always logic zero.

BMOD	BT Input Clock	WDT Interval Time (3)	
x000b	fxx/2 <sup>12</sup>	$2^3 \times 2^{12} \times 2^8 / \text{fxx} - (2^3 - 1) \times 2^{12} \times 2^8 / \text{fxx}$	1.75-2.0 sec
x011b	fxx/2 <sup>9</sup>	$2^{3} \times 2^{9} \times 2^{8}$ /fxx - (2 <sup>3</sup> -1) × 2 <sup>9</sup> × 2 <sup>8</sup> /fxx	218.7-250 ms
x101b	fxx/2 <sup>7</sup>	$2^3 \times 2^7 \times 2^8 / \text{fxx} - (2^3 - 1) \times 2^7 \times 2^8 / \text{fxx}$	54.6-62.5ms
x111b	fxx/2 <sup>5</sup>	$2^3 \times 2^5 \times 2^8 / \text{fxx} - (2^3 - 1) \times 2^5 \times 2^8 / \text{fxx}$	13.6-15.6 ms

Table 11-3. Watchdog Timer Interval Time

#### NOTES:

- 1. Clock frequencies assume a system oscillator clock frequency (fx) of 4.19 MHz
- 2. fxx = system clock frequency.
- 3. When the watchdog timer is enabled or the 3-bit counter of the watchdog timer is cleared to "0", the BCNT value is not cleared but increased continuously. As a result, the 3-bit counter of the watchdog timer (WDCNT) can be increased by 1. For example, when the BMOD value is x000b and the watchdog timer is enabled, the watchdog timer interval time is from 2<sup>3</sup> × 2<sup>12</sup> × 2<sup>8</sup>/fxx to (2<sup>3</sup>-1) × 2<sup>12</sup> × 2<sup>8</sup>/fxx.



## 

RESET	DI LD LD	EA,#00H SP,EA		
	• LD LD	A,#0DH BMOD,A	;	WDCNT input clock is 7.82 ms
	•	J. 100 J. 1		
MAIN	BITS •	WDTCF	;	Main routine operation period must be shorter than watchdog-timer's period
	• JP	MAIN		



## 8-BIT TIMER/COUNTER 0 (TC0)

#### **OVERVIEW**

Timer/counter 0 (TC0) is used to count system 'events' by identifying the transition (high-to-low or low-to-high) of incoming square wave signals. To indicate that an event has occurred, or that a specified time interval has elapsed, TC0 generates an interrupt request. By counting signal transitions and comparing the current counter value with the reference register value, TC0 can be used to measure specific time intervals.

TC0 has a reloadable counter that consists of two parts: an 8-bit reference register (TREF0) into which you write the counter reference value, and an 8-bit counter register (TCNT0) whose value is automatically incremented by counter logic.

An 8-bit mode register, TMOD0, is used to activate the timer/counter and to select the basic clock frequency to be used for timer/counter operations. To dynamically modify the basic frequency, new values can be loaded into the TMOD0 register during program execution.

#### **TC0 FUNCTION SUMMARY**

8-bit programmable timer	Generates interrupts at specific time intervals based on the selected clock frequency.
External event counter	Counts various system "events" based on edge detection of external clock signals at the TC0 input pin, TCL0. To start the event counting operation, TMOD0.2 is set to "1" and TMOD0.6 is cleared to "0".
Arbitrary frequency output	Outputs selectable clock frequencies to the TC0 output pin, TCLO0.
External signal divider	Divides the frequency of an incoming external clock signal according to a modi-

fiable reference value (TREF0), and outputs the modified frequency to the

Serial I/O clock source Outputs a modifiable clock signal for use as the SCK clock source.

TCLO0 pin.



#### **TC0 COMPONENT SUMMARY**

Mode register (TMOD0) Activates the timer/counter and selects the internal clock frequency or the

external clock source at the TCL0 pin.

Reference register (TREF0) Stores the reference value for the desired number of clock pulses between in-

terrupt requests.

Counter register (TCNT0) Counts internal or external clock pulses based on the bit settings in TMOD0

and TREF0.

Clock selector circuit Together with the mode register (TMOD0), lets you select one of four internal

clock frequencies or an external clock.

8-bit comparator Determines when to generate an interrupt by comparing the current value of

the counter register (TCNT0) with the reference value previously programmed

into the reference register (TREF0).

Output latch (TOL0) Where a clock pulse is stored pending output to the serial I/O circuit or to the

TC0 output pin, TCLO0.

When the contents of the TCNT0 and TREF0 registers coincide, the

timer/counter interrupt request flag (IRQT0) is set to "1", the status of TOL0 is

inverted, and an interrupt is generated.

Output enable flag (TOE0) Must be set to logic one before the contents of the TOL0 latch can be output to

TCLO0.

Interrupt request flag (IRQT0) Cleared when TC0 operation starts and the TC0 interrupt service routine is

executed and set to 1 whenever the counter value and reference value

coincide.

Interrupt enable flag (IET0) Must be set to logic one before the interrupt requests generated by

timer/counter 0 can be processed.

Table 11-4. TC0 Register Overview

Register Name	Туре	Description	Size	RAM Address	Addressing Mode	Reset Value
TMOD0	Control	Controls TC0 enable/disable (bit 2); clears and resumes counting operation (bit 3); sets input clock and clock frequency (bits 6-4)	8-bit	F90H-F91H	8-bit write- only; (TMOD0.3 is also 1-bit writeable)	"0"
TCNT0	Counter	Counts clock pulses matching the TMOD0 frequency setting	8-bit	F94H-F95H	8-bit read-only	"0"
TREF0	Reference	Stores reference value for the timer/counter 0 interval setting	8-bit	F96H-F97H	8-bit write-only	FFH
TOE0	Flag	Controls timer/counter 0 output to the TCLO0 pin	1-bit	F92H.2	1/4-bit read/write	"0"



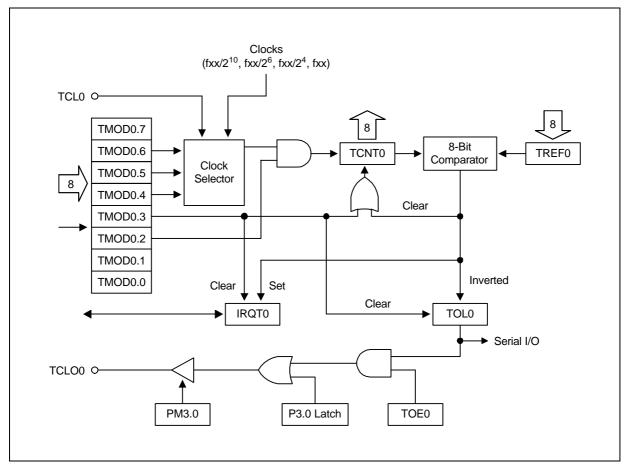


Figure 11-2. TC0 Circuit Diagram

## TC0 ENABLE/DISABLE PROCEDURE

## **Enable Timer/Counter 0**

- Set TMOD0.2 to logic one.
- Set the TC0 interrupt enable flag IET0 to logic one.
- Set TMOD0.3 to logic one.

TCNT0, IRQT0, and TOL0 are cleared to logic zero, and timer/counter operation starts.

### **Disable Timer/Counter 0**

Set TMOD0.2 to logic zero.

Clock signal input to the counter register TCNT0 is halted. The current TCNT0 value is retained and can be read if necessary.



#### TC0 PROGRAMMABLE TIMER/COUNTER FUNCTION

Timer/counter 0 can be programmed to generate interrupt requests at various intervals based on the selected system clock frequency. Its 8-bit TC0 mode register TMOD0 is used to activate the timer/counter and to select the clock frequency. The reference register TREF0 stores the value for the number of clock pulses to be generated between interrupt requests. The counter register, TCNT0, counts the incoming clock pulses, which are compared to the TREF0 value as TCNT0 is incremented. When there is a match (TREF0 = TCNT0), an interrupt request is generated.

To program timer/counter 0 to generate interrupt requests at specific intervals, choose one of four internal clock frequencies (divisions of the system clock, fxx) and load a counter reference value into the TREF0 register. TCNT0 is incremented each time an internal counter pulse is detected with the reference clock frequency specified by TMOD0.4-TMOD0.6 settings. To generate an interrupt request, the TC0 interrupt request flag (IRQT0) is set to logic one, the status of TOL0 is inverted, and the interrupt is generated. The content of TCNT0 is then cleared to 00H and TC0 continues counting. The interrupt request mechanism for TC0 includes an interrupt enable flag (IET0) and an interrupt request flag (IRQT0).

#### **TC0 OPERATION SEQUENCE**

The general sequence of operations for using TC0 can be summarized as follows:

- 1. Set TMOD0.2 to "1" to enable TC0.
- 2. Set TMOD0.6 to "1" to enable the system clock (fxx) input.
- 3. Set TMOD0.5 and TMOD0.4 bits to desired internal frequency (fxx/2<sup>n</sup>).
- 4. Load a value to TREF0 to specify the interval between interrupt requests.
- 5. Set the TC0 interrupt enable flag (IET0) to "1".
- 6. Set TMOD0.3 bit to "1" to clear TCNT0, IRQT0, and TOL0, and start counting.
- 7. TCNT0 increments with each internal clock pulse.
- 8. When the comparator shows TCNT0 = TREF0, the IRQT0 flag is set to "1" and an interrupt request is generated.
- 9. Output latch (TOL0) logic toggles high or low.
- 10. TCNT0 is cleared to 00H and counting resumes.
- 11. Programmable timer/counter operation continues until TMOD0.2 is cleared to "0".



### **TC0 EVENT COUNTER FUNCTION**

Timer/counter 0 can monitor or detect system 'events' by using the external clock input at the TCL0 pin as the counter source. The TC0 mode register selects rising or falling edge detection for incoming clock signals. The counter register TCNT0 is incremented each time the selected state transition of the external clock signal occurs.

With the exception of the different TMOD0.4-TMOD0.6 settings, the operation sequence for TC0's event counter function is identical to its programmable timer/counter function. To activate the TC0 event counter function,

- Set TMOD0.2 to "1" to enable TC0.
- Clear TMOD0.6 to "0" to select the external clock source at the TCL0 pin.
- Select TCL0 edge detection for rising or falling signal edges by loading the appropriate values to TMOD0.5 and TMOD0.4.
- P3.2 must be set to input mode.

Table 11-5. TMOD0 Settings for TCL0 Edge Detection

TMOD0.5	TMOD0.4	TCL0 Edge Detection
0	0	Rising edges
0	1	Falling edges



#### **TC0 CLOCK FREQUENCY OUTPUT**

Using timer/counter 0, a modifiable clock frequency can be output to the TC0 clock output pin, TCLO0. To select the clock frequency, load the appropriate values to the TC0 mode register, TMOD0. The clock interval is selected by loading the desired reference value into the reference register TREF0. To enable the output to the TCLO0 pin, the following conditions must be met:

- TC0 output enable flag TOE0 must be set to "1".
- I/O mode flag for P3.0 (PM3.0) must be set to output mode ("1").
- Output latch value for P3.0 must be set to "0".

In summary, the operational sequence required to output a TC0-generated clock signal to the TCLO0 pin is as follows:

- 1. Load a reference value to TREF0.
- 2. Set the internal clock frequency in TMOD0.
- 3. Initiate TC0 clock output to TCLO0 (TMOD0.2 = "1").
- 4. Set P3.0 mode flag (PM3.0) to "1".
- 5. Set P3.0 output latch to "0".
- 6. Set TOE0 flag to "1".

Each time TCNT0 overflows and an interrupt request is generated, the state of the output latch TOL0 is inverted and the TC0-generated clock signal is output to the TCLO0 pin.

## PROGRAMMING TIP — TC0 Signal Output to the TCLO0 Pin

Output a 30 ms pulse width signal to the TCLO0 pin:

BITS	EMB
SMB	15
LD	EA,#79H
LD	TREF0,EA
LD	EA,#4CH
LD	TMOD0,EA
LD	EA,#01H
LD	PMG2,EA

LD PMG2,EA ; P3.0 ← output mode

BITR P3.0 ; P3.0 clear

BITS TOE0



#### TC0 SERIAL I/O CLOCK GENERATION

Timer/counter 0 can supply a clock signal to the clock selector circuit of the serial I/O interface for data shifter and clock counter operations. (These internal SIO operations are controlled in turn by the SIO mode register, SMOD). This clock generation function enables you to adjust data transmission rates across the serial interface.

Use TMOD0 and TREF0 register settings to select the frequency and interval of the TC0 clock signals to be used as SCK input to the serial interface. The generated clock signal is then sent directly to the serial I/O clock selector circuit (the TOE0 flag may be disabled).

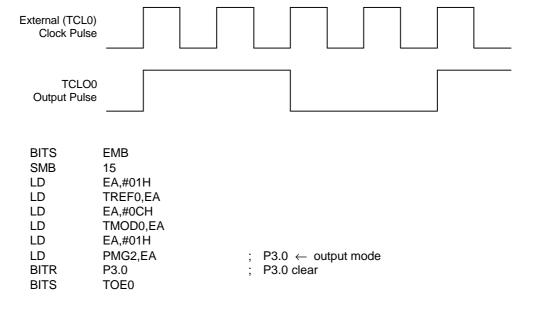
#### **TC0 EXTERNAL INPUT SIGNAL DIVIDER**

By selecting an external clock source and loading a reference value into the TC0 reference register, TREF0, you can divide the incoming clock signal by the TREF0 value and then output this modified clock frequency to the TCLO0 pin. The sequence of operations used to divide external clock input can be summarized as follows:

- 1. Load a signal divider value to the TREF0 register.
- 2. Clear TMOD0.6 to "0" to enable external clock input at the TCL0 pin.
- 3. Set TMOD0.5 and TMOD0.4 to desired TCL0 signal edge detection.
- 4. Set port 3.0 mode flag (PM3.0) to output ("1").
- 5. Set P3.0 output latch to "0".
- 6. Set TOE0 flag to "1" to enable output of the divided frequency to the TCLO0 pin.

## PROGRAMMING TIP — External TCL0 Clock Output to the TCL00 Pin

Output external TCL0 clock pulse to the TCLO0 pin (divided by four):





### TC0 MODE REGISTER (TMOD0)

TMOD0 is the 8-bit mode control register for timer/counter 0. It is addressable by 8-bit write instructions. One bit, TMOD0.3, is also 1-bit writeable. RESET clears all TMOD0 bits to logic zero and disables TC0 operations.

F90H	TMOD0.3	TMOD0.2	"0"	"0"
F91H	"0"	TMOD0.6	TMOD0.5	TMOD0.4

TMOD0.2 is the enable/disable bit for timer/counter 0. When TMOD0.3 is set to "1", the contents of TCNT0, IRQT0, and TOL0 are cleared, counting starts from 00H, and TMOD0.3 is automatically reset to "0" for normal TC0 operation. When TC0 operation stops (TMOD0.2 = "0"), the contents of the TC0 counter register TCNT0 are retained until TC0 is re-enabled.

The TMOD0.6, TMOD0.5, and TMOD0.4 bit settings are used together to select the TC0 clock source. This selection involves two variables:

- Synchronization of timer/counter operations with either the rising edge or the falling edge of the clock signal input at the TCL0 pin, and
- Selection of one of four frequencies, based on division of the incoming system clock frequency, for use in internal TC0 operation.

Table 11-6. TC0 Mode Register (TMOD0) Organization

Bit Name	Setting	Resulting TC0 Function	Address
TMOD0.7	0	Always logic zero	F91H
TMOD0.6	0,1	Specify input clock edge and internal frequency	
TMOD0.5			
TMOD0.4			
TMOD0.3	1	Clear TCNT0, IRQT0, and TOL0 and resume counting immediately (This bit is automatically cleared to logic zero immediately after counting resumes.)	F90H
TMOD0.2	0	Disable timer/counter 0; retain TCNT0 contents	
	1	Enable timer/counter 0	
TMOD0.1	0	Always logic zero	
TMOD0.0	0	Always logic zero	



Table 11-7. TMOD0.6, TMOD0.5, and TMOD0.4 Bit Settings

TMOD0.6	TMOD0.5	TMOD0.4	Resulting Counter Source and Clock Frequency
0	0	0	External clock input (TCL0) on rising edges
0	0	1	External clock input (TCL0) on falling edges
1	0	0	fxx/2 <sup>10</sup> (4.09 kHz)
1	0	1	fxx /2 <sup>6</sup> (65.5 kHz)
1	1	0	fxx/2 <sup>4</sup> (262 kHz)
1	1	1	fxx (4.19 MHz)

**NOTE:** 'fxx' = selected system clock of 4.19 MHz.

## PROGRAMMING TIP — Restarting TC0 Counting Operation

1. Set TC0 timer interval to 4.09 kHz:

BITS EMB SMB 15 LD EA,#4CH LD TMOD0,EA

ΕI

BITS IET0

2. Clear TCNT0, IRQT0, and TOL0 and restart TC0 counting operation:

BITS EMB SMB 15

BITS TMOD0.3



### **TC0 COUNTER REGISTER (TCNT0)**

The 8-bit counter register for timer/counter 0, TCNT0, is read-only and can be addressed by 8-bit RAM control instructions. RESET sets all TCNT0 register values to logic zero (00H).

Whenever TMOD0.3 is enabled, TCNT0 is cleared to logic zero and counting resumes. The TCNT0 register value is incremented each time an incoming clock signal is detected that matches the signal edge and frequency setting of the TMOD0 register (specifically, TMOD0.6, TMOD0.5, and TMOD0.4).

Each time TCNT0 is incremented, the new value is compared to the reference value stored in the TC0 reference buffer, TREF0. When TCNT0 = TREF0, an overflow occurs in the TCNT0 register, the interrupt request flag, IRQT0, is set to logic one, and an interrupt request is generated to indicate that the specified timer/counter interval has elapsed.

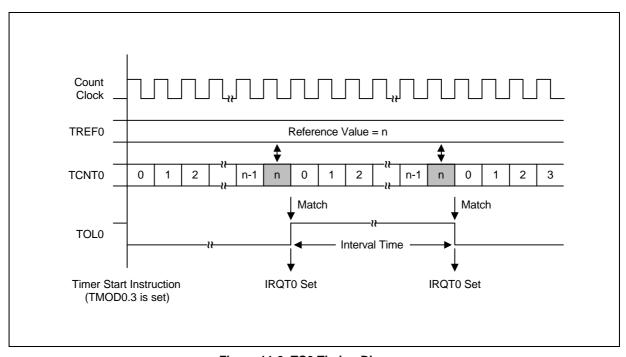


Figure 11-3. TC0 Timing Diagram



#### **TC0 REFERENCE REGISTER (TREF0)**

The TC0 reference register TREF0 is an 8-bit write-only register. It is addressable by 8-bit RAM control instructions. RESET initializes the TREF0 value to 'FFH'.

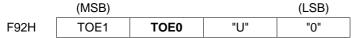
TREF0 is used to store a reference value to be compared to the incrementing TCNT0 register in order to identify an elapsed time interval. Reference values will differ depending upon the specific function that TC0 is being used to perform — as a programmable timer/counter, event counter, clock signal divider, or arbitrary frequency output source.

During timer/counter operation, the value loaded into the reference register is compared to the TCNT0 value. When TCNT0 = TREF0, the TC0 output latch (TOL0) is inverted and an interrupt request is generated to signal the interval or event. The TREF0 value, together with the TMOD0 clock frequency selection, determines the specific TC0 timer interval. Use the following formula to calculate the correct value to load to the TREF0 reference register:

TC0 timer interval = 
$$(TREF0 \text{ value} + 1) \times \frac{1}{TMOD0 \text{ frequency setting}}$$
  
 $(TREF0 \text{ value} \neq 0)$ 

#### TC0 OUTPUT ENABLE FLAG (TOE0)

The 1-bit timer/counter 0 output enable flag TOE0 controls output from timer/counter 0 to the TCLO0 pin. TOE0 is addressable by 1-bit and 4-bit read/write instruction.



NOTE: The "U" means that the bit is undefined.

When you set the TOE0 flag to "1", the contents of TOL0 can be output to the TCL00 pin. Whenever a RESET occurs, TOE0 is automatically set to logic zero, disabling all TC0 output. Even when the TOE0 flag is disabled, timer/counter 0 can continue to output an internally-generated clock frequency, via TOL0, to the serial I/O clock selector circuit.

### TC0 OUTPUT LATCH (TOL0)

TOL0 is the output latch for timer/counter 0. When the 8-bit comparator detects a correspondence between the value of the counter register TCNT0 and the reference value stored in the TREF0 register, the TOL0 value is inverted — the latch toggles high-to-low or low-to-high. Whenever the state of TOL0 is switched, the TC0 signal is output. TC0 output may be directed to the TCL00 pin, or it can be output directly to the serial I/O clock selector circuit as the SCK signal.

Assuming TC0 is enabled, when bit 3 of the TMOD0 register is set to "1", the TOL0 latch is cleared to logic zero, along with the counter register TCNT0 and the interrupt request flag, IRQT0, and counting resumes immediately. When TC0 is disabled (TMOD0.2 = "0"), the contents of the TOL0 latch are retained and can be read, if necessary.



## PROGRAMMING TIP — Setting a TC0 Timer Interval

To set a 30 ms timer interval for TC0, given fxx = 4.19 MHz, follow these steps.

- 1. Select the timer/counter 0 mode register with a maximum setup time of 62.5 ms (assume the TC0 counter clock =  $fxx/2^{10}$ , and TREF0 is set to FFH):
- 2. Calculate the TREF0 value:

$$30 \text{ ms} = \frac{\text{TREF0 value} + 1}{4.09 \text{ kHz}}$$

$$\text{TREF0} + 1 = \frac{30 \text{ ms}}{244 \text{ µs}} = 122.9 = 7\text{AH}$$

$$\text{TREF0 value} = 7\text{AH} - 1 = 79\text{H}$$

3. Load the value 79H to the TREF0 register:

BITS	EMB
SMB	15
LD	EA,#79H
LD	TREF0,EA
LD	EA,#4CH
LD	TMOD0,EA



#### **16-BIT TIMER/COUNTER 1**

#### **OVERVIEW**

Timer/counter 1 (TC1) is used to count system 'events' by identifying the transition (high-to-low or low-to-high) of incoming square wave signals. To indicate that an event has occurred, or that a specified time interval has elapsed, TC1 generates an interrupt request. By counting signal transitions, it can be used to measure time intervals. The TC1 circuit also has 16-bit comparator logic.

TC1 has a reloadable counter that consists of two parts: a 16-bit reference register (TREF1) into which you can write data for use as a reference value, and a 16-bit counter register (TCNT1) whose contents are automatically incremented by counter logic.

The 8-bit mode register, TMOD1, is used to activate the timer/counter and to select the basic clock frequency to be used for timer/counter operations. You can modify the basic frequency dynamically by loading new values into TMOD1 during program execution.

The only functional differences between TC0 and TC1 are the size of the counter and reference value registers (8-bit versus 16-bit), and the fact that only TC0 can generate a clock signal for the serial I/O interface.

#### **TIMER/COUNTER 1 FUNCTION SUMMARY**

16-bit programmable timer	Generates interrupts at specific time intervals based on the selected clock frequency.
External event counter	Counts various system "events" based on edge detection of external clock signals at the TC1 input pin, TCL1.
Arbitrary frequency output	Outputs selectable clock frequencies to the TC1 output pin, TCLO1.
External signal divider	Divides the frequency of an incoming external clock signal according to the modifiable reference value (TREF1), and outputs the modified frequency to the TCLO1 pin.



#### **TIMER/COUNTER 1 COMPONENT SUMMARY**

Mode register (TMOD1) Activates the timer/counter and selects the internal clock frequency or the

external clock source at the TCL1 pin.

Reference register (TREF1) Stores the reference value for the desired number of clock pulses between

interrupt requests.

Counter register (TCNT1) Counts internal clock pulses that are generated based on bit settings in the

mode register and reference register.

Clock selector circuit Together with the mode register (TMOD1), lets you select one of four internal

clock frequencies, or the external system clock source.

16-bit comparator Determines when to generate an interrupt by comparing the current value of

the counter (TCNT1) with the reference value previously programmed into the

reference register (TREF1).

Output latch (TOL1) Where a TC1 clock pulse is stored pending output to the TC1 output pin,

TCLO1. When the contents of the TCNT1 and TREF1 registers coincide, the timer/counter interrupt request flag (IRQT1) is set to "1", the status of TOL1 is

inverted, and an interrupt is generated.

Output enable flag (TOE1) Must be set to logic one before the contents of the TOL1 latch can be output to

TCLO1.

Interrupt request flag (IRQT1) Cleared when TC1 operation starts and set to logic one whenever the counter

value and reference value match.

Interrupt enable flag (IET1) Must be set to logic one before the interrupt requests generated by

timer/counter 1 can be processed.

Table 11-8. TC1 Register Overview

Register Name	Туре	Description	Size	RAM Address	Addressing Mode	Reset Value
TMOD1	Control	Controls TC1 enable/disable (bit 2); clears and resumes counting operation (bit 3); sets input clock and the clock frequency (bits 6-4)	8-bit	FA0H-FA1H	8-bit write- only; (TMOD1.3 is also 1-bit writeable)	"0"
TCNT1	Counter	Counts clock pulses matching the TMOD1 frequency setting	16-bit	FA4H-FA5H, FA6H-FA7H	8-bit read-only	"0"
TREF1	Reference	Stores reference value for TC1 interval setting	16-bit	FA8H-FA9H, FAAH-FABH	8-bit write-only	FFFFH
TOE1	Flag	Controls TC1 output to the TCLO1 pin	1-bit	F92H.3	1/4-bit read/write	"0"



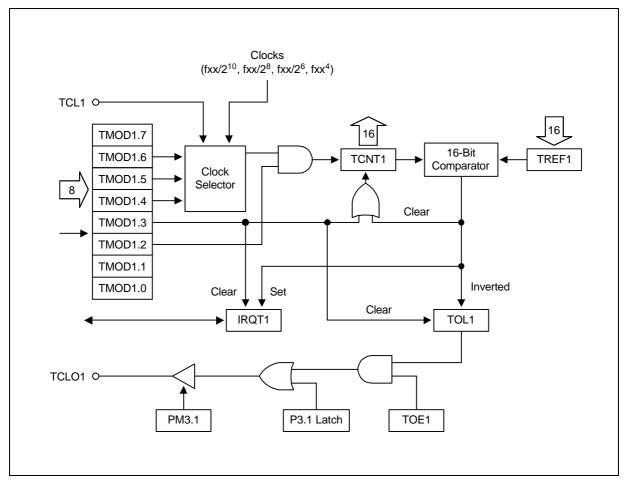


Figure 11-4. TC1 Circuit Diagram

### TC1 ENABLE/DISABLE PROCEDURE

### **Enable Timer/Counter 1**

- Set the TC1 interrupt enable flag IET1 to logic one.
- Set TMOD1.3 to logic one.

TCNT1, IRQT1, and TOL1 are cleared to logic zero, and timer/counter operation starts.

### **Disable Timer/Counter 1**

Set TMOD1.2 to logic zero.

Clock signal input to the counter register TCNT1 is halted. The current TCNT1 value is retained and can be read if necessary.



#### TC1 PROGRAMMABLE TIMER/COUNTER FUNCTION

Timer/counter 1 can be programmed to generate interrupt requests at variable intervals, based on the system clock frequency you select. The 8-bit TC1 mode register, TMOD1, is used to activate the timer/counter and to select the clock frequency; the 16-bit reference register, TREF1, is used to store the value for the desired number of clock pulses between interrupt requests. The 16-bit counter register, TCNT1, counts the incoming clock pulses, which are compared to the TREF1 value. When there is a match, an interrupt request is generated.

To program timer/counter 1 to generate interrupt requests at specific intervals, select one of the four internal clock frequencies (divisions of the system clock, fxx) and load a counter reference value into the TREF1 register. TCNT1 is incremented each time an internal counter pulse is detected with the reference clock frequency specified by TMOD1.4-TMOD1.6 settings. To generate an interrupt request, the TC1 interrupt request flag (IRQT1) is set to logic one, the status of TOL1 is inverted, and the interrupt is output. The content of TCNT1 is then cleared to 0000H, and TC1 continues counting. The interrupt request mechanism for TC1 includes an interrupt enable flag (IET1) and an interrupt request flag (IRQT1).

#### TC1 TIMER/COUNTER OPERATION SEQUENCE

The general sequence of operations for using TC1 can be summarized as follows:

- 1. Set TMOD1.2 to "1" to enable TC1.
- 2. Set TMOD1.6 to "1" to enable the system clock (fxx) input.
- 3. Set TMOD1.5 and TMOD1.4 bits to desired internal frequency (fxx/2<sup>n</sup>).
- 4. Load a value to TREF1 to specify the interval between interrupt requests.
- 5. Set the TC1 interrupt enable flag (IET1) to "1".
- 6. Set TMOD1.3 bit to "1" to clear TCNT1, IRQT1, and TOL1, and start counting.
- 7. TCNT1 increments with each internal clock pulse.
- 8. When the comparator shows TCNT1 = TREF1, the IRQT1 flag is set to "1" and an interrupt request is generated.
- 9. Output latch (TOL1) logic toggles high or low.
- 10. TCNT1 is cleared to 0000H and counting resumes.
- 11. Programmable timer/counter operation continues until TMOD1.2 is cleared to "0".



### **TC1 EVENT COUNTER FUNCTION**

Timer/counter 1 can monitor system 'events' by using the external clock input at the TCL1 pin as the counter source. The TC1 mode register selects rising or falling edge detection for incoming clock signals. The counter register TCNT1 is incremented each time the selected state transition of the external clock signal occurs.

With the exception of the different TMOD1.4-TMOD1.6 settings, the operation sequence for TC1's event counter function is identical to its programmable timer/counter function. To activate the TC1 event counter function,

- Set TMOD1.2 to "1" to enable TC1.
- Clear TMOD1.6 to "0" to select the external clock source at the TCL1 pin.
- Select TCL1 edge detection for rising or falling signal edges by loading the appropriate values to TMOD1.5 and TMOD1.4.
- Pin P3.3 must be set to input mode.

Table 11-9. TMOD1 Settings for TCL1 Edge Detection

TMOD1.5	TMOD1.4	TCL1 Edge Detection
0	0	Rising edges
0	1	Falling edges



#### TC1 CLOCK FREQUENCY OUTPUT

Using timer/counter 1, a modifiable clock frequency can be output to the TC1 clock output pin, TCLO1. To select the clock frequency, load the appropriate values to the TC1 mode register, TMOD1. The clock interval is selected by loading the desired reference value into the 16-bit reference register TREF1. To enable the output to the TCLO1 pin at I/O port 3.1, the following conditions must be met:

- TC1 output enable flag TOE1 must be set to "1".
- I/O mode flag for P3.1 (PM3.1) must be set to output mode ("1").
- P3.1 output latch must be cleared to "0".

In summary, the operational sequence required to output a TC1-generated clock signal to the TCLO1 pin is as follows:

- 1. Load your reference value to TREF1.
- 2. Set the internal clock frequency in TMOD1.
- 3. Initiate TC1 clock output to TCLO1 (TMOD1.2 = "1").
- 4. Set port 3.1 mode flag (PM3.1) to "1".
- 5. Clear the P3.1 output latch.
- 6. Set TOE1 flag to "1".

Each time TCNT1 overflows and an interrupt request is generated, the state of the output latch TOL1 is inverted and the TC1-generated clock signal is output to the TCLO1 pin.

## PROGRAMMING TIP — TC1 Signal Output to the TCLO1 Pin

Output a 30 ms pulse width signal to the TCLO1 pin:

BITS	EMB
SMB	15
LD	EA,#79H
LD	TREF1A,EA
LD	EA,#00H
LD	TREF1B,EA
LD	EA,#4CH
LD	TMOD1,EA
LD	EA,#02H
LD	PMG2.EA

.D PMG2,EA ; P3.1 ← output mode

BITR P3.1 ; P3.1 clear

BITS TOE1



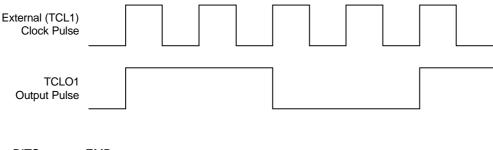
### TC1 EXTERNAL INPUT SIGNAL DIVIDER

By selecting an external clock source and loading a reference value into the TC1 reference register, TREF1, you can divide the incoming clock signal by the TREF1 value and then output this modified clock frequency to the TCLO1 pin. The sequence of operations used to divide external clock input and output the signals to the TCLO1 pin can be summarized as follows:

- 1. Load a signal divider value to the TREF1 register.
- 2. Clear TMOD1.6 to "0" to enable external clock input at the TCLO1 pin.
- 3. Set TMOD1.5 and TMOD1.4 to desired TCL signal edge detection.
- 4. Set P3.1 mode flag (PM3.1) to output ("1").
- 5. Clear the P3.1 output latch.
- 6. Set TOE1 flag to "1" to enable output of the divided frequency.

## PROGRAMMING TIP — External TCL1 Clock Output to the TCL01 Pin

Output the external TCL1 clock source to the TCLO1 pin (divide by four):



BITS	EMB		
SMB	15		
LD	EA,#01H		
LD	TREF1A,EA		
LD	EA,#00H		
LD	TREF1B,EA		
LD	EA,#0CH		
LD	TMOD1,EA		
LD	EA,#02H		
LD	PMG2,EA	;	P3.1 ← output mode
BITR	P3.1	;	P3.1 clear
RITS	TOF1		



## TC1 MODE REGISTER (TMOD1)

TMOD1 is the 8-bit mode register for timer/counter 1. It is addressable by 8-bit write instructions. The TMOD1.3 bit is also 1-bit write addressable. RESET clears all TMOD1 bits to logic zero. Following a RESET, timer/counter 1 is disabled.

FA0H	TMOD1.3	TMOD1.2	"0"	"0"
FA1H	"0"	TMOD1.6	TMOD1.5	TMOD1.4

TMOD1.2 is the enable/disable bit for timer/counter 1. When TMOD1.3 is set to "1", the contents of TCNT1, IRQT1, and TOL1 are cleared, counting starts from 0000H, and TMOD1.3 is automatically reset to "0" for normal TC1 operation. When TC1 operation stops (TMOD1.2 = "0"), the contents of the TC1 counter register, TCNT1, are retained until TC1 is re-enabled.

The TMOD1.6, TMOD1.5, and TMOD1.4 bit settings are used together to select the TC1 clock source. This selection involves two variables:

- Synchronization of timer/counter operations with either the rising edge or the falling edge of the clock signal input at the TCL1 pin, and
- Selection of one of four frequencies, based on division of the incoming system clock frequency, for use in internal TC1 operations.

Table 11-10. TC1	Mode Register	(TMOD1)	Organization
------------------	---------------	---------	--------------

Bit Name	Setting	Resulting TC1 Function	Address
TMOD1.7	0	Always logic zero	
TMOD1.6	0,1	Specify input clock edge and internal frequency FA	
TMOD1.5			
TMOD1.4			
TMOD1.3	1	Clear TCNT1, IRQT1, and TOL1 and resume counting immediately (This bit is automatically cleared to logic zero immediately after counting resumes).	FA0H
TMOD1.2	0	Disable timer/counter 1; retain TCNT1 contents	
	1	Enable timer/counter 1	
TMOD1.1	0	Always logic zero	
TMOD1.0	0	Always logic zero	



Table 11-11. TMOD1.6, TMOD1.5, and TMOD1.4 Bit Settings

TMOD1.6	TMOD1.5	TMOD1.4	Resulting Counter Source and Clock Frequency
0	0	0	External clock input (TCL1) on rising edges
0	0	1	External clock input (TCL1) on falling edges
1	0	0	fxx/2 <sup>10</sup> (4.09 kHz)
1	0	1	fxx/2 <sup>8</sup> (16.4 kHz)
1	1	0	fxx/2 <sup>6</sup> (65.5 kHz)
1	1	1	fxx/2 <sup>4</sup> (262 kHz)

**NOTE:** 'fxx' = selected system clock of 4.19 MHz.

## PROGRAMMING TIP — Restarting TC1 Counting Operation

1. Set TC1 timer interval to 4.09 kHz:

**BITS EMB** SMB 15 EA,#4CH LD LD TMOD1,EA ΕI

BITS IET1

2. Clear TCNT1, IRQT1, and TOL1 and restart TC1 counting operation:

SBITS **EMB** SMB 15 **BITS** TMOD1.3



### TC1 COUNTER REGISTER (TCNT1)

The 16-bit counter register for timer/counter 1, TCNT1, is mapped to RAM addresses FA5H-FA4H (TCNT1A) and FA7H-FA6H (TCNT1B). The two 8-bit registers are read-only and can be addressed by 8-bit RAM control instructions. RESET sets all TCNT1 register values to logic zero (00H).

Whenever TMOD1.2 and TMOD1.3 are enabled, TCNT1 is cleared to logic zero and counting begins. The TCNT1 register value is incremented each time an incoming clock signal is detected that matches the signal edge and frequency setting of the TMOD1 register (specifically, TMOD1.6, TMOD1.5, and TMOD1.4).

Each time TCNT1 is incremented, the new value is compared to the reference value stored in the TC1 reference register, TREF1. When TCNT1 = TREF1, an overflow occurs in the TCNT1 register, the interrupt request flag, IRQT1, is set to logic one, and an interrupt request is generated to indicate that the specified timer/counter interval has elapsed.

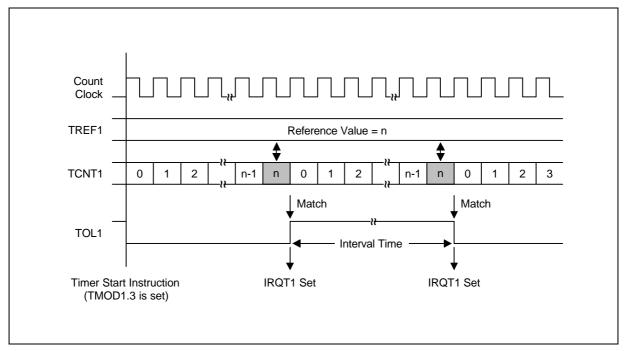


Figure 11-5. TC1 Timing Diagram



#### TC1 REFERENCE REGISTER (TREF1)

The TC1 reference register TREF1 is a 16-bit write-only register that is mapped to RAM locations FA9H-FA8H (TREF1A) and FABH-FAAH (TREF1B). It is addressable by 8-bit RAM control instructions. RESET clears the TREF1 value to 'FFFFH'.

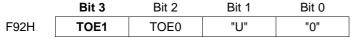
TREF1 is used to store a reference value to be compared to the incrementing TCNT1 register in order to identify an elapsed time interval. Reference values will differ depending upon the specific function that TC1 is being used to perform — as a programmable timer/counter, event counter, clock signal divider, or arbitrary frequency output source.

During timer/counter operation, the value loaded into the reference register compared to the TCNT1 value. When TCNT1 = TREF1, the TC1 output latch (TOL1) is inverted and an interrupt request is generated to signal the interval or event. The TREF1 value, together with the TMOD1 clock frequency selection, determines the specific TC1 timer interval. Use the following formula to calculate the correct value to load to the TREF1 reference register:

TC1 timer interval = (TREF1 value + 1) 
$$\times \frac{1}{\text{TMOD1 frequency setting}}$$
(TREF1 value  $\neq 0$ )

### TC1 OUTPUT ENABLE FLAG (TOE1)

The 1-bit timer/counter 1 output enable flag TOE1 flag controls output from timer/counter 1 to the TCLO1 pin. TOE1 is addressable by 1-bit read and write instructions.



**NOTE:** The "U" means that the bit is undefined.

When you set the TOE1 flag to "1", the contents of TOL1 can be output to the TCLO1 pin. Whenever a RESET occurs, TOE1 is automatically set to logic zero, disabling all TC1 output.

#### TC1 OUTPUT LATCH (TOL1)

TOL1 is the output latch for timer/counter 1. When the 16-bit comparator detects a correspondence between the value of the counter register TCNT1 and the reference value stored in the TREF1 register, the TOL1 logic toggles high-to-low or low-to-high. Whenever the state of TOL1 is switched, the TC1 signal exits the latch for output. TC1 output is directed (if TOE1 = "1") to the TCLO1 pin at I/O port 3.1.

When timer/counter 1 is started, (TMOD1.3 = "0"), the contents of the output latch are cleared automatically. However, when TC1 is disabled (TMOD1.2 = "0"), the contents of the TOL1 latch are retained and can be read, if necessary.



## PROGRAMMING TIP — Setting a TC1 Timer Interval

To set a 30 ms timer interval for TC1, given fxx = 4.19 MHz, follow these steps:

- 1. Select the timer/counter 1 mode register with a maximum setup time of 16 seconds; assume the TC1 counter clock =  $fxx/2^{10}$  and TREF1 is set to FFFFH.
- 2. Calculate the TREF1 value:

$$30 \text{ ms} = \frac{\text{TREF1 value} + 1}{4.09 \text{ kHz}}$$

$$\text{TREF1} + 1 = \frac{30 \text{ ms}}{244 \text{ µs}} = 122.9 = 7\text{AH}$$

$$\text{TREF1 value} = 7\text{AH} - 1 = 79\text{H}$$

3. Load the value 79H to the TREF1 register:

BITS	EMB
SMB	15
LD	EA,#79H
LD	TREF1A,EA
LD	EA,#00H
LD	TREF1B,EA
LD	EA,#4CH
LD	TMOD1,EA



#### **WATCH TIMER**

#### **OVERVIEW**

The watch timer is a multi-purpose timer which consists of three basic components:

- 8-bit watch timer mode register (WMOD)
- Clock selector
- Frequency divider circuit

Watch timer functions include real-time and watch-time measurement and interval timing for the main and subsystem clock. It is also used as a clock source for the LCD controller and for generating buzzer (BUZ) output.

### **Real-Time and Watch-Time Measurement**

To start watch timer operation, set bit 2 of the watch timer mode register (WMOD.2) to logic one. The watch timer starts, the interrupt request flag IRQW is automatically set to logic one, and interrupt requests commence in 0.5-second intervals.

Since the watch timer functions as a quasi-interrupt instead of a vectored interrupt, the IRQW flag should be cleared to logic zero by program software as soon as a requested interrupt service routine has been executed.

#### Using a Main System or Subsystem Clock Source

The watch timer can generate interrupts based on the main system clock frequency or on the subsystem clock. When the zero bit of the WMOD register is set to "1", the watch timer uses the subsystem clock signal (fxt) as its source; if WMOD.0 = "0", the main system clock (fx) is used as the signal source, according to the following formula:

Watch timer clock (fw) = 
$$\frac{\text{Main system clock (fx)}}{128}$$
 = 32.768 kHz (fx = 4.19 MHz)

This feature is useful for controlling timer-related operations during stop mode. When stop mode is engaged, the main system clock (fx) is halted, but the subsystem clock continues to oscillate. By using the subsystem clock as the oscillation source during stop mode, the watch timer can set the interrupt request flag IRQW to "1", thereby releasing stop mode.

#### **Clock Source Generation for LCD Controller**

The watch timer supplies the clock frequency for the LCD controller (f<sub>LCD</sub>). Therefore, if the watch timer is disabled, the LCD controller does not operate.



#### **Buzzer Output Frequency Generator**

The watch timer can generate a steady 2 kHz, 4 kHz, 8 kHz, or 16 kHz signal to the BUZ pin. To select the desired BUZ frequency, load the appropriate value to the WMOD register. This output can then be used to actuate an external buzzer sound. To generate a BUZ signal, three conditions must be met:

- The WMOD.7 register bit is set to "1"
- The output latch for I/O port 0.3 is cleared to "0"
- The port 0.3 output mode flag (PM0.3) set to 'output' mode

#### **Timing Tests in High-Speed Mode**

By setting WMOD.1 to "1", the watch timer will operate in high-speed mode, generating an interrupt every 3.91 ms. At its normal speed (WMOD.1 = '0'), the watch timer generates an interrupt request every 0.5 seconds. High-speed mode is useful for timing events for program debugging sequences.

#### **Check Subsystem Clock Level Feature**

The watch timer can also check the input level of the subsystem clock by testing WMOD.3. If WMOD.3 is "1", the input level at the  $XT_{IN}$  pin is high; if WMOD.3 is "0", the input level at the  $XT_{IN}$  pin is low.



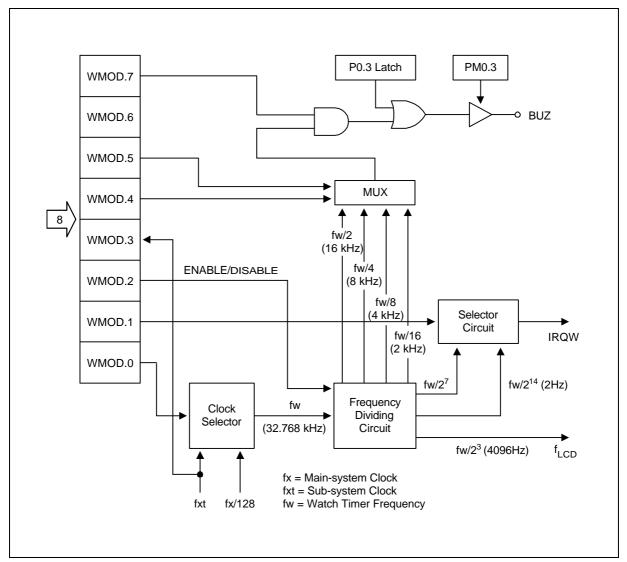


Figure 11-6. Watch Timer Circuit Diagram



#### WATCH TIMER MODE REGISTER (WMOD)

The watch timer mode register WMOD is used to select specific watch timer operations. It is 8-bit write-only addressable. An exception is WMOD bit 3 (the  $XT_{IN}$  input level control bit) which is 1-bit read-only addressable. A RESET automatically sets WMOD.3 to the current input level of the subsystem clock,  $XT_{IN}$  (high, if logic one; low, if logic zero), and all other WMOD bits to logic zero.

F88H	WMOD.3	WMOD.2	WMOD.1	WMOD.0
F89H	WMOD.7	"0"	WMOD.5	WMOD.4

In summary, WMOD settings control the following watch timer functions:

Watch timer clock selection (WMOD.0)
 Watch timer speed control (WMOD.1)
 Enable/disable watch timer (WMOD.2)
 XT<sub>IN</sub> input level control (WMOD.3)

Buzzer frequency selection (WMOD.4 and WMOD.5)

Enable/disable buzzer output (WMOD.7)

Table 11-12. Watch Timer Mode Register (WMOD) Organization

Bit Name	Values		Function	Address	
WMOD.7	0		Disable buzzer (BUZ) signal output at the BUZ pin	F89H	
			1		Enable buzzer (BUZ) signal output at the BUZ pin
WMOD.6	(	)	Always logic zero		
WMOD.54	0	0	2 kHz buzzer (BUZ) signal output		
	0	1	4 kHz buzzer (BUZ) signal output		
	1	0	8 kHz buzzer (BUZ) signal output		
	1	1	16 kHz buzzer (BUZ) signal output		
WMOD.3	(	)	Input level to XT <sub>IN</sub> pin is low	F88H	
		I	Input level to XT <sub>IN</sub> pin is high		
WMOD.2	0		Disable watch timer; clear frequency dividing circuits		
	•	1	Enable watch timer		
WMOD.1	0		Normal mode; sets IRQW to 0.5 seconds		
	1		High-speed mode; sets IRQW to 3.91 ms		
WMOD.0	0		Select (fx/128 ) as the watch timer clock (fw)		
	,	1	Select subsystem clock as watch timer clock (fw)		

NOTE: Main system clock frequency (fx) is assumed to be 4.19 MHz; subsystem clock (fxt) is assumed to be 32.768 kHz.



## PROGRAMMING TIP — Using the Watch Timer

1. Select a subsystem clock as the LCD display clock, a 0.5 second interrupt, and 2 kHz buzzer enable:

BITS EMB SMB 15 LD EA,#8H LD PMG1,EA

; P0.3  $\leftarrow$  output mode

BITR P0.3 LD EA,#85H LD WMOD,EA BITS IEW

2. Sample real-time clock processing method:

CLOCK BTSTZ IRQW ; 0.5 second check

RET ; No, return

• Yes, 0.5 second interrupt generation

; Increment HOUR, MINUTE, SECOND



# **12**

# LCD CONTROLLER/DRIVER

#### **OVERVIEW**

The KS57C21832 microcontroller can directly drive an up-to-896-dot (56 segments x 16 commons) LCD panel. Its LCD block has the following components:

- LCD controller/driver
- Display RAM for storing display data
- 56 segment output pins (SEG0-SEG55)
- 16 common output pins (COM0-COM15)
- Five LCD operating power supply pins (V<sub>LC1</sub>-V<sub>LC5</sub>)
- V<sub>LC5</sub> pin for controlling the driver and bias voltage

To use the LCD controller, bit 2 in the watch mode register WMOD must be set to 1, because LCDCK is supplied by the watch timer.

The frame frequency, duty and bias, and the segment pins used for display output, are determined by bit settings in the LCD mode register, LMOD.

The LCD control register, LCON, is used to turn the LCD display on and off, to switch current to the dividing resistors for the LCD display, and to output LCD clock (LCDCK) and synchronizing signal (LCDSY) for LCD display expansion. Data written to the LCD display RAM can be transferred to the segment signal pins automatically without program control.

When a subsystem clock is selected as the LCD clock source, the LCD display is enabled even during main clock stop and idle modes.

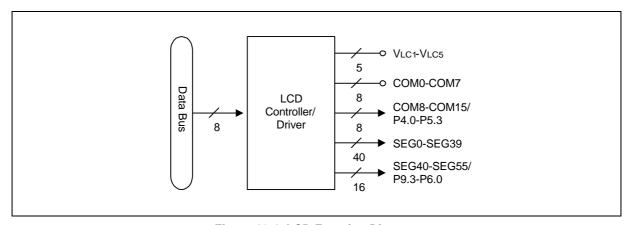


Figure 12-1. LCD Function Diagram



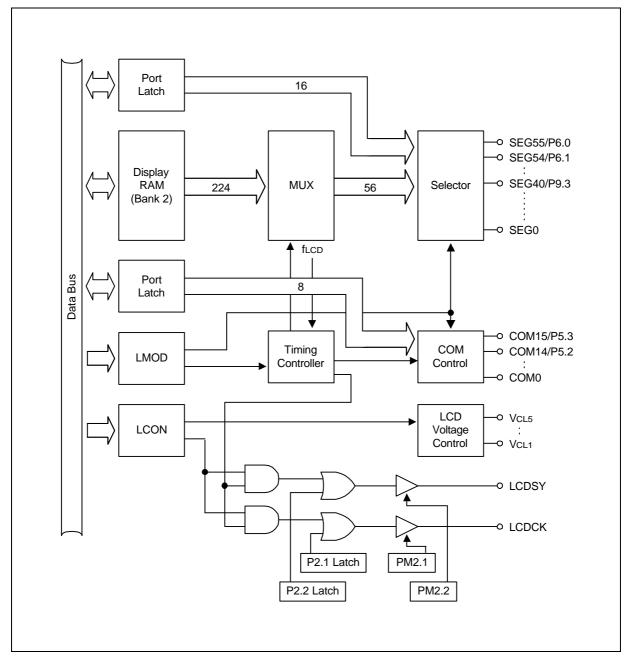


Figure 12-2. LCD Circuit Diagram



#### **LCD RAM ADDRESS AREA**

RAM addresses of bank 2 are used as LCD data memory. These locations can be addressed by 1-bit, 4-bit, or 8-bit instructions. When the bit value of a display segment is "1", the LCD display is turned on; when the bit value is "0", the display is turned off.

Display RAM data are sent out through segment pins SEG0-SEG55 using a direct memory access (DMA) method that is synchronized with the  $f_{LCD}$  signal. RAM addresses in this location that are not used for LCD display can be allocated to general-purpose use.

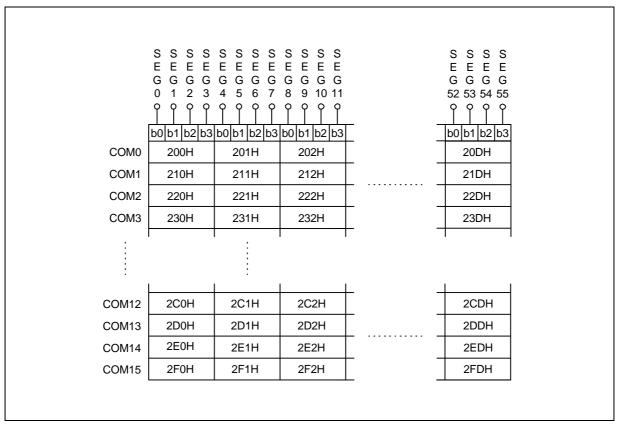


Figure 12-3. LCD Display Data RAM Organization

Table 12-1. Common and Segment Pins per Duty Cycle

Duty	Common Pins	Segment Pins	Dot Number
1/16	COM0-COM15	40 pins-56 pins	640 dots-896 dots
1/8	COM0-COM7		320 dots-448 dots

NOTE: When 1/8 duty is selected, COM8-COM15 (P4.0-P5.3) can be used for normal I/O pins.



#### **LCD CONTROL REGISTER (LCON)**

The LCD control register (LCON) is used to turn the LCD display on and off, to output LCD clock (LCDCK) and synchronizing signal (LCDSY) for LCD display expansion, and to control the flow of current to dividing resistors in the LCD circuit. Following a RESET, all LCON values are cleared to "0". This turns the LCD display off and stops the flow of current to the dividing resistors.

F8EH	"0"	LCON.2	LCON.1	LCON.0
-	-			

The effect of the LCON.0 setting is dependent upon the current setting of bits LMOD.0 and LMOD.1. Bit 1 in the LCON is used for contrast control application.

Table 12-2. LCD Control Register (LCON) Organization

LCON Bit	Setting		Description
LCON.3	(	)	Always logic zero.
LCON.2	0		Disable LCDCK and LCDSY signal outputs.
	1		Enable LCDCK and LCDSY signal outputs.
LCON.1	0	0	LCD display off; cut off current to dividing resistor
LCON.0	0	1	LCD display on; application without contrast control
	1	0	LCD display on; application with contrast control
	1	1	LCD display on; application without contrast control

**NOTE:** The function of LCON.0 is applied in case of using the internal GND for LCD power; the function of LCON1 is used for contrast control application.

Table 12-3. LMOD.1-0 Bits Settings

LMOD.1-0	COM0-COM15	SEG0-SEG55	SEG40/P9.3-SEG55/P6.0	Power Supply to the Dividing Resistor
0, 0	All of the LCD dots	off	Normal I/O port function	On
0, 1	All of the LCD dots	on		
1, 1	Common and segn corresponds to disp display mode)			

NOTE: 'x' means 'don't care.'



#### **LCD MODE REGISTER (LMOD)**

The LCD mode control register LMOD is used to control display mode; LCD clock, segment or port output, and display on/off. LMOD can be manipulated using 8-bit write instructions.

F8CH	LMOD.3	LMOD.2	LMOD.1	LMOD.0
F8DH	LMOD.7	LMOD.6	LMOD.5	LMOD.4

The LCD clock signal, LCDCK, determines the frequency of COM signal scanning of each segment output. This is also referred to as the 'frame frequency. Since LCDCK is generated by dividing the watch timer clock (fw), the watch timer must be enabled when the LCD display is turned on. RESET clears the LMOD register values to logic zero.

The LCD display can continue to operate during idle and stop modes if a subsystem clock is used as the watch timer source. The LCD mode register LMOD controls the output mode of the 16 pins used for normal outputs (P9.3-P6.0). Bits LMOD.7-5 define the segment output and normal bit output configuration.

Table 12-4. LCD Clock Signal (LCDCK) Frame Frequency

LCDCK	256 Hz	512 Hz	1024 Hz	2048 Hz	4096 Hz
Display Duty Cycle					
1/8	32	64	128	256	_
1/16	_	32	64	128	256

#### NOTE:

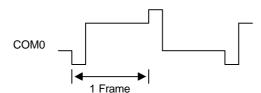


Table 12-5. LCD Mode Register (LMOD) Organization

#### **Segment/Port Output Selection Bits**

LMOD.7	LMOD.6	LMOD.5	SEG40-43	SEG44-47	SEG48-51	SEG52-55	Total Number of Segment
0	0	0	SEG port	SEG port	SEG port	SEG port	56
0	0	1	SEG port	SEG port	SEG port	Normal port	52
0	1	0	SEG port	SEG port	Normal port	Normal port	48
0	1	1	SEG port	Normal port	Normal port	Normal port	44
1	0	0	Normal port	Normal port	Normal port	Normal port	40

**NOTE:** Segment pins that also can used for normal I/O should be configured to output mode when the SEG function is used.

#### **LCD Clock Selection Bits**

LMOD.4	LMOD.3	LCD Clock (LCDCK)		
		1/8 duty (COM0-COM7)	1/16 duty (COM0-COM15)	
0	0	fxx/2 <sup>7</sup> (256 Hz)	fxx/2 <sup>6</sup> (512 Hz)	
0	1	fxx/2 <sup>6</sup> (512 Hz)	fxx/2 <sup>5</sup> (1024 Hz)	
1	0	fxx/2 <sup>5</sup> (1024 Hz)	fxx/2 <sup>4</sup> (2048 Hz)	
1	1	fxx/2 <sup>4</sup> (2048 Hz)	fxx/2 <sup>3</sup> (4096 Hz)	

**NOTE:** LCDCK is supplied only when the watch timer is operating. To use the LCD controller, you must set bit 2 in the watch mode register WMOD to "1".

## **Duty Selection Bits**

LMOD.2	Duty
0	1/8 duty (COM0-COM7 select)
1	1/16 duty (COM0-COM15 select)

**NOTE:** When 1/16 duty is selected, ports 4 and 5 should be configured as output mode; when 1/8 duty is selected, ports 4 and 5 can be used as normal I/O ports.

#### **Display Mode Selection Bits**

LMOD.1	LMOD.0	Function
0	0	All LCD dots off
0	1	All LCD dots on
1	1	Normal display



#### LCD VOLTAGE DIVIDING RESISTORS

On-chip voltage dividing resistors for the LCD drive power supply are fixed to the  $V_{LC1}$ - $V_{LC5}$  pins. Power can be supplied without an external dividing resistor. Figure 12-4 shows the bias connections for the KS57C21832 LCD drive power supply. To cut off the flow of current through the dividing resistor, clear bits 0 and 1 of the LCON register.

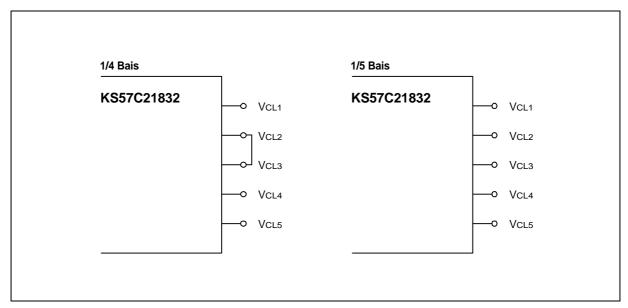


Figure 12-4. LCD Bias Circuit Connection



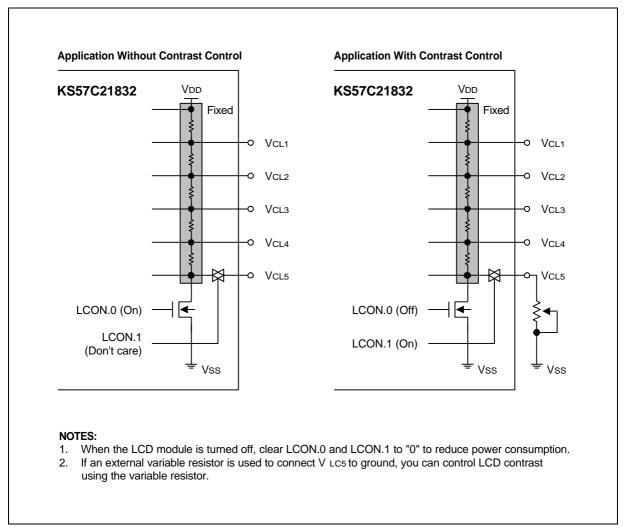


Figure 12-5. Internal Voltage Dividing Resistor Connection (1/5 Bias, Display On)

## **COMMON (COM) SIGNALS**

The common signal output pin selection (COM pin selection) varies according to the selected duty cycle.

- In 1/8 duty mode, COM0-COM7 pins are selected.
- In 1/16 duty mode, COM0-COM15 pins are selected.

When 1/8 duty is selected by clearing LMOD.2 to zero, COM8-COM15 (P4.0-P5.3) can be used for normal I/O port.

#### **SEGMENT (SEG) SIGNALS**

The 56 LCD segment signal pins are connected to corresponding display RAM locations at bank 2. Bits of the display RAM are synchronized with the common signal output pins.

When the bit value of a display RAM location is "1", a select signal is sent to the corresponding segment pin. When the display bit is "0", a 'no-select' signal is sent to the corresponding segment pin.



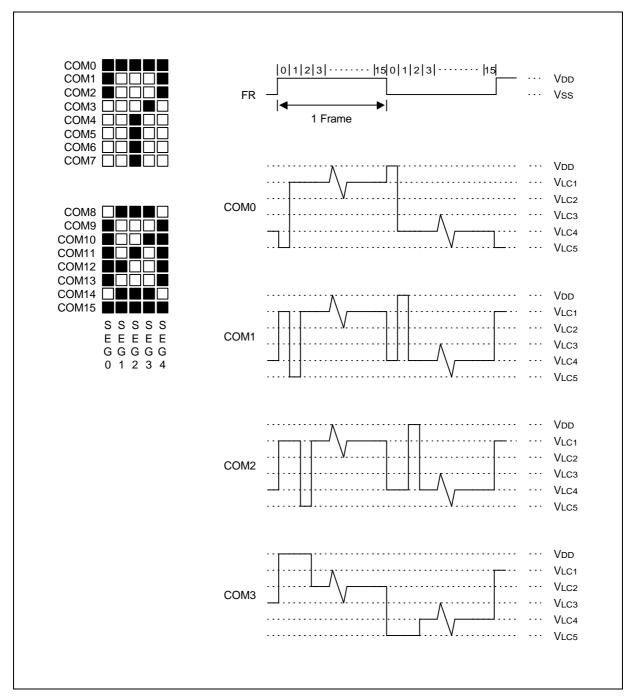


Figure 12-6. LCD Signal Waveforms (1/16 Duty, 1/5 Bias)



KS57C21832/P21832 LCD CONTROLLER/DRIVER

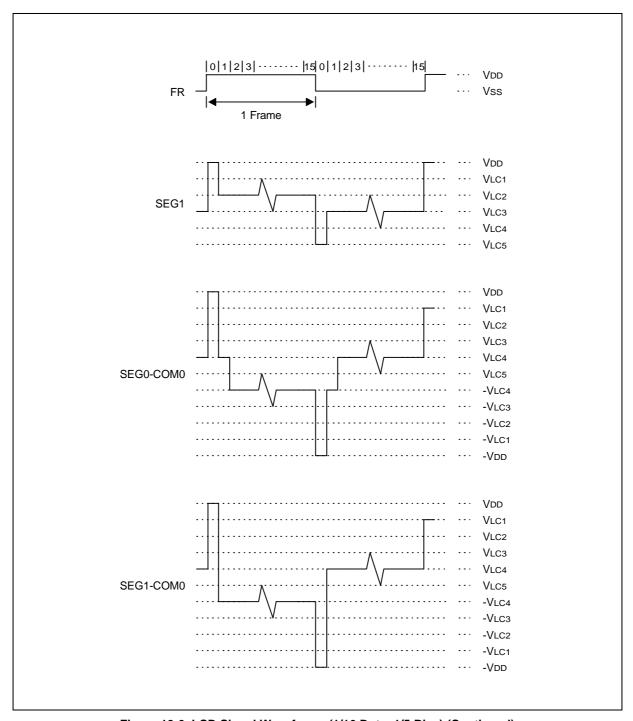


Figure 12-6. LCD Signal Waveforms (1/16 Duty, 1/5 Bias) (Continued)



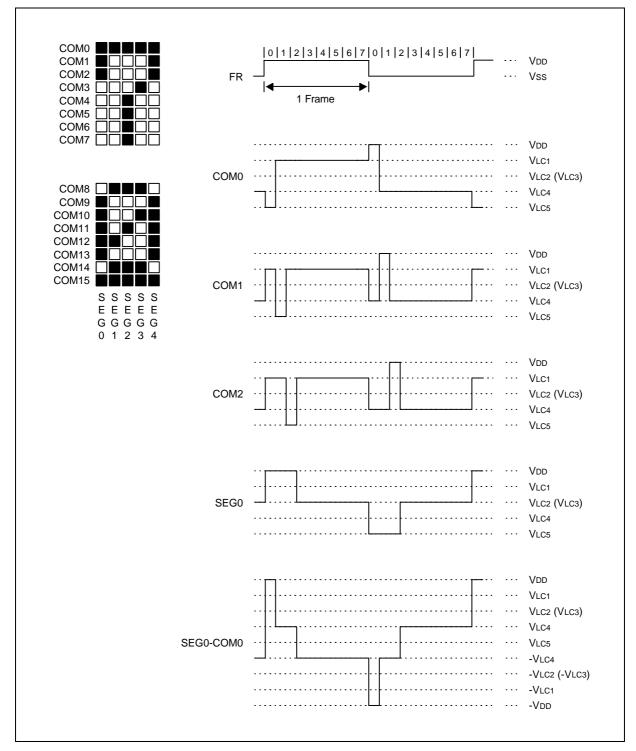


Figure 12-7. LCD Signal Waveforms (1/8 Duty, 1/4 Bias)



KS57C21832/P21832 LCD CONTROLLER/DRIVER

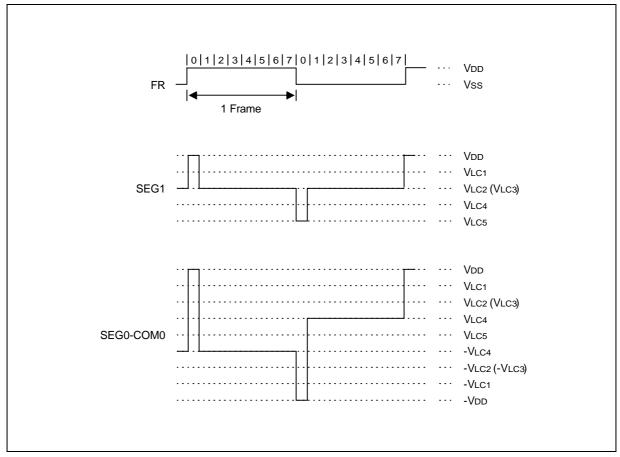


Figure 12-7. LCD Signal Waveforms (1/8 Duty, 1/4 Bias) (Continued)

# **NOTES**



KS57C21832/P21832 SERIAL I/O INTERFACE

13

# **SERIAL I/O INTERFACE**

#### **OVERVIEW**

The serial I/O interface (SIO) has the following functional components:

- 8-bit mode register (SMOD)
- Clock selector circuit
- 8-bit buffer register (SBUF)
- 3-bit serial clock counter

Using the serial I/O interface, 8-bit data can be exchanged with an external device. The transmission frequency is controlled by making the appropriate bit settings to the SMOD register.

The serial interface can run off an internal or an external clock source, or the TOL0 signal that is generated by the 8-bit timer/counter, TC0. If the TOL0 clock signal is used, you can modify its frequency to adjust the serial data transmission rate.

## **SERIAL I/O OPERATION SEQUENCE**

The general operation sequence of the serial I/O interface can be summarized as follows:

- 1. Set SIO mode to transmit-and-receive or to receive-only.
- 2. Select MSB-first or LSB-first transmission mode.
- 3. Set the SCK clock signal in the mode register, SMOD.
- 4. Set SIO interrupt enable flag (IES) to "1".
- 5. Initiate SIO transmission by setting bit 3 of the SMOD to "1".
- 6. When the SIO operation is complete, IRQS flag is set and an interrupt is generated.



SERIAL I/O INTERFACE KS57C21832/P21832

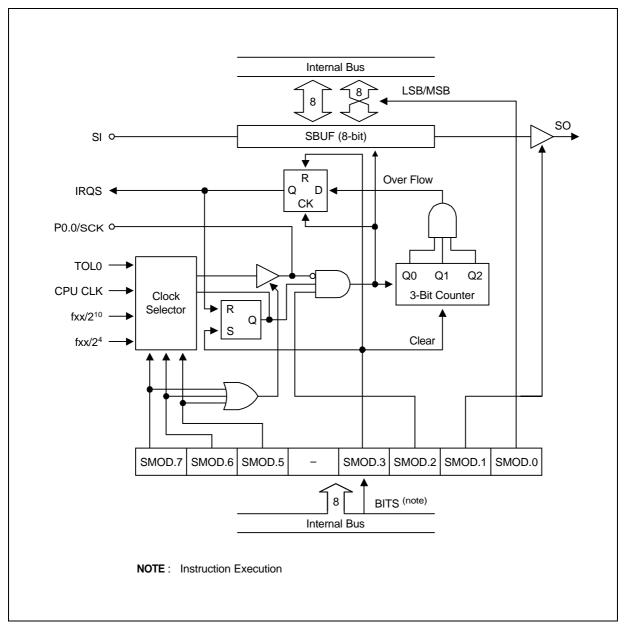


Figure 13-1. Serial I/O Interface Circuit Diagram

KS57C21832/P21832 SERIAL I/O INTERFACE

#### **SERIAL I/O MODE REGISTER (SMOD)**

The serial I/O mode register, SMOD, is an 8-bit register that specifies the operation mode of the serial interface. Its reset value is logical zero. SMOD is organized in two 4-bit registers, as follows:

FE0H	SMOD.3	SMOD.2	SMOD.1	SMOD.0
FE1H	SMOD.7	SMOD.6	SMOD.5	0

SMOD register settings enable you to select either MSB-first or LSB-first serial transmission, and to operate in transmit-and-receive mode or receive-only mode. SMOD is a write-only register and can be addressed only by 8-bit RAM control instructions. One exception to this is SMOD.3, which can be written by a 1-bit RAM control instruction. When SMOD.3 is set to 1, the contents of the serial interface interrupt request flag, IRQS, and the 3-bit serial clock counter are cleared, and SIO operations are initiated. When the SIO transmission starts, SMOD.3 is cleared to logical zero.

Table 13-1. SIO Mode Register (SMOD) Organization

SMOD.0	0	Most significant bit (MSB) is transmitted first
	1	Least significant bit (LSB) is transmitted first
SMOD.1	0	Receive-only mode
	1	Transmit-and-receive mode
SMOD.2	0	Disable the data shifter and clock counter; retain contents of IRQS flag when serial transmission is halted
	1	Enable the data shifter and clock counter; set IRQS flag to "1" when serial transmission is halted
SMOD.3	1	Clear IRQS flag and 3-bit clock counter to "0"; initiate transmission and then reset this bit to logic zero
SMOD.4	0	Bit not used; value is always "0"

SMOD.7	SMOD.6	SMOD.5	Clock Selection	R/W Status of SBUF
0	0	0	External clock at SCK pin	SBUF is enabled when SIO operation is halted or when SCK goes high.
0	0	1	Use TOL0 clock from TC0	
0	1	х	CPU clock: fxx/4, fxx/8, fxx/64	Enable SBUF read/write
1	0	0	4.09 kHz clock: fxx/2 <sup>10</sup>	SBUF is enabled when SIO operation is halted or when SCK goes high.
1	1	1	262 kHz clock: fxx/2 <sup>4</sup>	

#### NOTES:

- 1. 'fxx' = system clock; 'x' means 'don't care.'
- 2. kHz frequency ratings assume a system clock (fxx) running at 4.19 MHz.
- 3. The SIO clock selector circuit cannot select a fxx/2<sup>4</sup> clock if the CPU clock is fxx/64.
- 4. It must be selected MSB-first or LSB-first transmission mode before loading the data to SBUF.



SERIAL I/O INTERFACE KS57C21832/P21832

#### **SERIAL I/O TIMING DIAGRAMS**

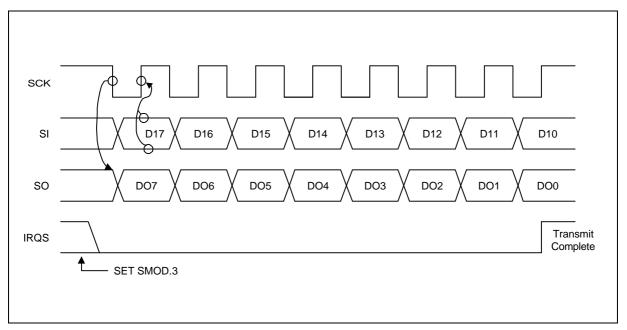


Figure 13-2. SIO Timing in Transmit/Receive Mode

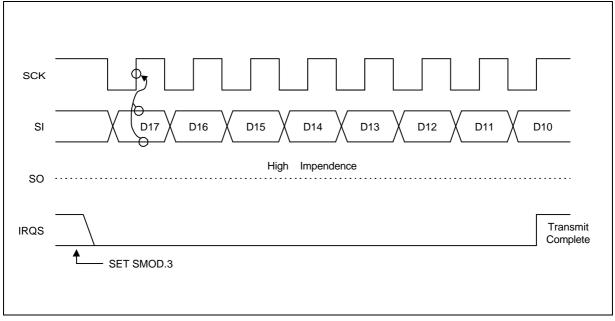


Figure 13-3. SIO Timing in Receive-Only Mode



KS57C21832/P21832 SERIAL I/O INTERFACE

#### **SERIAL I/O BUFFER REGISTER (SBUF)**

The serial I/O buffer register ,SBUF, can be read or written using 8-bit RAM control instructions. Following a RESET, the value of SBUF is undetermined.

When the serial interface operates in transmit-and-receive mode (SMOD.1 = "1"), transmit data in the SIO buffer register are output to the SO pin (P0.1) at the rate of one bit for each falling edge of the SIO clock. Receive data are simultaneously input from the SI pin (P0.2) to SBUF at the rate of one bit for each rising edge of the SIO clock. When receive-only mode is used, incoming data are input to the SIO buffer at the rate of one bit for each rising edge of the SIO clock.

# PROGRAMMING TIP — Setting Transmit/Receive Modes for Serial I/O

1. Transmit the data value 48H through the serial I/O interface using an internal clock frequency of fxx/2 and in MSB-first mode:

```
BITS
           EMB
SMB
           15
LD
           EA,#03H
LD
           PMG1,EA
LD
           EA,#0E6H
LD
           SMOD, EA
                                     P0.0/SCK and P0.1/SO ← Output
LD
           EA,#48H
LD
           SBUF, EA
BITS
           SMOD.3
                                     SIO data transfer
   SCK/P0.0
                                External
    SO/P0.1
                                Device
```

2. Use CPU clock to transfer and receive serial data at high speed:

[KS57C21832]

```
BITS
                       EMB
            SMB
                       15
           LD
                       EA.#03H
           LD
                       PMG1.EA
                                               ; P0.0/SCK and P0.1/SO \leftarrow Output, P0.2/SI \leftarrow Input
           LD
                       EA.#47H
           LD
                       SMOD.EA
            LD
                       EA,TDATA
           LD
                       SBUF,EA
            BITS
                       SMOD.3
                                               : SIO start
           BITR
                       IES
STEST
           BTSTZ
                       IRQS
            JR
                       STEST
           LD
                       EA,SBUF
            SMB
           LD
                       RDATA, EA
```

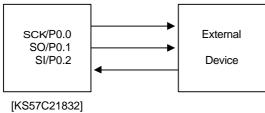


SERIAL I/O INTERFACE KS57C21832/P21832

# PROGRAMMING TIP — Setting Transmit/Receive Modes for Serial I/O (Continued)

3. Transmit and receive an internal clock frequency of 4.09 kHz (at 4.19 MHz) in LSB-first mode:

	BITS SMB LD LD LD LD LD LD ED LD BITS EI BITS .	EMB 15 EA,#03H PMG1,EA EA,#87H SMOD,EA EA,TDATA SBUF,EA SMOD.3	; P0.0/SCK and P0.1/SO ← Output, P0.2/SI ← Input ; SIO start
INTS	PUSH PUSH LD SMB XCH SMB LD BITS POP POP IRET	SB EA EA,TDATA 15 EA,SBUF 0 RDATA,EA SMOD.3 EA SB	; Store SMB, SRB ; Store EA ; EA ← Transmit data ; EA ← Receive data ; RDATA ← Receive data ; SIO start

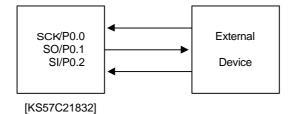


KS57C21832/P21832 SERIAL I/O INTERFACE

# PROGRAMMING TIP — Setting Transmit/Receive Modes for Serial I/O (Continued)

4. Transmit and receive an external clock in LSB-first mode:

**BITS EMB** SMB 15 EA,#02H LD LD PMG1,EA LD EA,#07H LD SMOD,EA ; P0.1/SO  $\leftarrow$  Output, P0.0/SCK and P0.2/SI  $\leftarrow$  Input LD EA,TDATA LD SBUF,EA ; SIO start **BITS** SMOD.3 ΕI **BITS IES** INTS **PUSH** SB ; Store SMB, SRB **PUSH** Store EA EΑ LD EA,TDATA ; EA ← Transmit data SMB 15 EA,SBUF XCH ; EA ← Receive data SMB LD RDATA,EA ; RDATA ← Receive data **BITS** SMOD.3 ; SIO start POP EΑ POP SB **IRET** 



High Speed SIO Transmission

SERIAL I/O INTERFACE KS57C21832/P21832

# PROGRAMMING TIP — Setting Transmit/Receive Modes for Serial I/O (Concluded)

Use CPU clock to transfer and receive serial data at high speed:

	BITS	EMB		
	SMB	15		
	LD	EA,#03H		
	LD	PMG1,EA		
	LD	EA,#47H		
	LD	SMOD, EA	;	P0.0/SCK and P0.1/SO ← Output, P0.2/SI ← Input
	LD	EA,TDATA		•
	LD	SBUF,EA		
	BITS	SCMOD.3	;	SIO start
	BITR	IES		
STEST	BTSTZ	IRQS		
	JR	STEST		
	LD	EA,SBUF		
	SMB	0		
	LD	RDATA.EA		

KS57C21832/P21832 ELECTRICAL DATA

# 14

# **ELECTRICAL DATA**

#### **OVERVIEW**

In this section, information on KS57C21832 electrical characteristics is presented as tables and graphics. The information is arranged in the following order:

#### **Standard Electrical Characteristics**

- Absolute maximum ratings
- D.C. electrical characteristics
- Main system clock oscillator characteristics
- Subsystem clock oscillator characteristics
- I/O capacitance
- A.C. electrical characteristics
- Operating voltage range

#### **Miscellaneous Timing Waveforms**

- A.C timing measurement point
- Clock timing measurement at X<sub>IN</sub>
- Clock timing measurement at XT<sub>IN</sub>
- TCL timing
- Input timing for RESET
- Input timing for external interrupts
- Serial data transfer timing

#### **Stop Mode Characteristics and Timing Waveforms**

- RAM data retention supply voltage in stop mode
- Stop mode release timing when initiated by RESET
- Stop mode release timing when initiated by an interrupt request



ELECTRICAL DATA KS57C21832/P21832

**Table 14-1. Absolute Maximum Ratings** 

 $(T_A = 25 \,^{\circ}C)$ 

Parameter	Symbol	Conditions	Rating	Units	
Supply Voltage	$V_{DD}$	_	- 0.3 to + 6.5	V	
Input Voltage	V <sub>I</sub>	Ports 0-9	- 0.3 to V <sub>DD</sub> + 0.3	V	
Output Voltage	Vo	_	- 0.3 to V <sub>DD</sub> + 0.3	V	
Output Current High	I <sub>OH</sub>	One I/O pin active	- 15	mA	
		All I/O pins active	- 35		
Output Current Low	I <sub>OL</sub>	One I/O pin active	+ 30 (Peak value)	mA	
			+ 15 <sup>(note)</sup>		
		Total for ports 0, 2-9	+ 100 (Peak value)		
			+ 60 <sup>(note)</sup>		
Operating Temperature	T <sub>A</sub>	_	- 40 to +85	°C	
Storage Temperature	T <sub>stg</sub>	_	- 65 to + 150	°C	

**NOTE:** The values for Output Current Low (  $I_{OL}$  ) are calculated as Peak Value  $\times$   $\sqrt{\text{Duty}}$  .

#### **Table 14-2. D.C. Electrical Characteristics**

 $(T_A = -40 \,^{\circ}C \text{ to } + 85 \,^{\circ}C, V_{DD} = 1.8 \,\text{V} \text{ to } 5.5 \,\text{V})$ 

Parameter	Symbol	Conditions	Min	Тур	Max	Units
Input High Voltage	V <sub>IH1</sub>	All input pins except those specified below for V <sub>IH2</sub> -V <sub>IH3</sub>	0.7 V <sub>DD</sub>	_	V <sub>DD</sub>	V
	V <sub>IH2</sub>	Ports 0, 1, 6, P3.2, P3.3, and RESET	0.8 V <sub>DD</sub>		V <sub>DD</sub>	
	V <sub>IH3</sub>	$X_{IN,} X_{OUT,}$ and $XT_{IN}$	V <sub>DD</sub> - 0.1		$V_{DD}$	
Input Low Voltage	V <sub>IL1</sub>	All input pins except those specified below for V <sub>IL2</sub> -V <sub>IL3</sub>	_	_	0.3 V <sub>DD</sub>	V
	V <sub>IL2</sub>	Ports 0, 1, 6, P3.2, P3.3, and RESET			0.2 V <sub>DD</sub>	
	V <sub>IL3</sub>	X <sub>IN,</sub> X <sub>OUT,</sub> and XT <sub>IN</sub>			0.1	
Output High Voltage	V <sub>OH</sub>	$V_{DD} = 4.5 \text{ V}$ to 5.5 V $I_{OH} = -1 \text{ mA}$ Ports 0, 2-9	V <sub>DD</sub> - 1.0	_	_	V
Output Low Voltage	V <sub>OL</sub>	$V_{DD} = 4.5 \text{ V}$ to 5.5 V $I_{OL} = 15 \text{ mA}$	_	_	2.0	V
		Ports 0, 2-9				



KS57C21832/P21832 ELECTRICAL DATA

Table 14-2. D.C. Electrical Characteristics (Continued)

 $(T_A = -40 \,^{\circ}C \text{ to } + 85 \,^{\circ}C, V_{DD} = 1.8 \,^{\circ}V \text{ to } 5.5 \,^{\circ}V)$ 

Parameter	Symbol	Conditions	Min	Тур	Max	Units
Input High Leakage Current	I <sub>LIH1</sub>	V <sub>I</sub> = V <sub>DD</sub> All input pins except those specified below for I <sub>LIH2</sub>	_	_	3	μΑ
	I <sub>LIH2</sub>	$V_I = V_{DD}$ $X_{IN, X_{OUT, }} XT_{IN, }$ and RESET			20	
Input Low Leakage	I <sub>LIL1</sub>	V <sub>I</sub> = 0 V All input pins except those specified below for I <sub>LIH2</sub>	-	_	- 3	μA
Current	I <sub>LIL2</sub>	$V_I = 0 V$ $X_{IN}$ , $X_{OUT}$ , and $XT_{IN}$			- 20	
Output High Leakage Current	I <sub>LOH</sub>	$V_O = V_{DD}$ All output pins	-	-	3	μΑ
Output Low Leakage Current	I <sub>LOL</sub>	V <sub>O</sub> = 0 V All output pins	-	_	- 3	μΑ
Pull-Up Resistor	R <sub>LI</sub>	V <sub>I</sub> = 0 V; V <sub>DD</sub> = 5 V Port 0-9	25	47	100	kΩ
		$V_{DD} = 3 V$	50	95	200	
	R <sub>L2</sub>	$V_I = 0 \text{ V}; V_{DD} = 5 \text{ V}, \text{RESET}$	100	220	400	
		V <sub>DD</sub> = 3 V	200	450	800	
LCD Voltage Dividing Resistor	R <sub>LCD</sub>	T <sub>A</sub> = 25 °C	25	55	80	kΩ
V <sub>DD</sub> -COMi  Voltage Drop (i = 0-15)	V <sub>DC</sub>	- 15 μA per common pin	-	-	120	mV
V <sub>DD</sub> -SEGx  Voltage Drop (x = 0-55)	V <sub>DS</sub>	- 15 μA per segment pin	-	_	120	
V <sub>LC1</sub> Output Voltage	V <sub>LC1</sub>	LCD clock = 0 Hz, V <sub>LC5</sub> = 0 V	0.8 V <sub>DD</sub> - 0.2	0.8 V <sub>DD</sub>	0.8 V <sub>DD</sub> + 0.2	V
V <sub>LC2</sub> Output Voltage	V <sub>LC2</sub>		0.6 V <sub>DD</sub> - 0.2	0.6 V <sub>DD</sub>	0.6 V <sub>DD</sub> + 0.2	
V <sub>LC3</sub> Output Voltage	V <sub>LC3</sub>		0.4 V <sub>DD</sub> - 0.2	0.4 V <sub>DD</sub>	0.4 V <sub>DD</sub> + 0.2	
V <sub>LC4</sub> Output Voltage	V <sub>LC4</sub>		0.2 V <sub>DD</sub> - 0.2	0.2 V <sub>DD</sub>	0.2 V <sub>DD</sub> + 0.2	



ELECTRICAL DATA KS57C21832/P21832

Table 14-2. D.C. Electrical Characteristics (Concluded)

 $(T_A = -40 \,^{\circ}\text{C} \text{ to } + 85 \,^{\circ}\text{C}, V_{DD} = 1.8 \,\text{V} \text{ to } 5.5 \,\text{V})$ 

Parameter	Symbol	Condition	ıs	Min	Тур	Max	Units
Supply Current	I <sub>DD1</sub> <sup>(2)</sup>	$V_{DD} = 5 V \pm 10\%$ Crystal oscillator C1 = C2 = 22 pF	6.0 MHz 4.19 MHz	_	5.1 3.9	10.0 7.5	mA
		$V_{DD} = 3 V \pm 10\%$	6.0 MHz 4.19 MHz		2.5 1.8	4.0 3.0	
	I <sub>DD2</sub> <sup>(2)</sup>	Idle mode; $V_{DD} = 5 \text{ V} \pm 10\%$ Crystal oscillator C1 = C2 = 22  pF	6.0 MHz 4.19 MHz		1.3 1.2	2.5 1.8	
		$V_{DD} = 3 V \pm 10\%$	6.0 MHz 4.19 MHz		0.5 0.44	1.5 1.0	
	I <sub>DD3</sub> (3)	$V_{DD} = 3 V \pm 10\%$ 32 kHz crystal oscillat	or	_	22.8	35	μΑ
	I <sub>DD4</sub> (3)	Idle mode; V <sub>DD</sub> = 3 V 32 kHz crystal oscillat			6.4	15	
	I <sub>DD5</sub>	Stop mode; V <sub>DD</sub> = 5 V ± 10%	SCMOD = 0000B		2.5	5	
		Stop mode; V <sub>DD</sub> = 3 V ± 10%	XT = 0V		0.5	3	
		Stop mode; V <sub>DD</sub> = 5 V ± 10%	SCMOD = 0100B		0.2	3	
		Stop mode; V <sub>DD</sub> = 3 V ± 10%			0.1	2	

#### NOTES:

- 1. Data includes power consumption for subsystem clock oscillation.
- 2. When the system clock control register, SCMOD, is set to 1001B, main system clock oscillation stops and the subsystem clock is used.
- 3. Currents in the following circuits are not included; on-chip pull-up resistors, internal LCD voltage dividing resistors, output port drive currents.



KS57C21832/P21832 ELECTRICAL DATA

**Table 14-3. Main System Clock Oscillator Characteristics** 

 $(T_A = -40 \,^{\circ}C + 85 \,^{\circ}C, V_{DD} = 1.8 \,^{\circ}V \text{ to } 5.5 \,^{\circ}V)$ 

Oscillator	Clock Configuration	Parameter	Test Condition	Min	Тур	Max	Units
Ceramic Oscillator	XIN XOUT  C1 C2	Oscillation frequency (1)	-	0.4	_	6.0	MHz
		Stabilization time (2)	Stabilization occurs when V <sub>DD</sub> is equal to the minimum oscillator voltage range; V <sub>DD</sub> = 3.0 V.	ı	_	4	ms
Crystal Oscillator	XIN XOUT  C1 C2	Oscillation frequency (1)	_	0.4	-	6.0	MHz
		Stabilization time (2)	V <sub>DD</sub> = 3.0 V	1	_	10	ms
			$V_{DD} = 2.0 \text{ V to } 5.5 \text{ V}$	-	_	30	
External Clock	XIN XOUT	X <sub>IN</sub> input frequency <sup>(1)</sup>	-	0.4	_	6.0	MHz
		X <sub>IN</sub> input high and low level width (t <sub>XH,</sub> t <sub>XL</sub> )	-	83.3	_	1250	ns
RC Oscillator	XIN XOUT	Frequency	$R = 20 \text{ k}\Omega,$ $V_{DD} = 5 \text{ V}$	_	2	_	MHz
			$R = 39 \text{ k}\Omega,$ $V_{DD} = 3 \text{ V}$	_	1	_	

#### NOTES:

- 1. Oscillation frequency and  $X_{\text{IN}}$  input frequency data are for oscillator characteristics only.
- 2. Stabilization time is the interval required for oscillator stabilization after a power-on occurs, or when stop mode is terminated.



**ELECTRICAL DATA** KS57C21832/P21832

**Table 14-4. Recommended Oscillator Constants** 

 $(T_A = -40 \,^{\circ}C + 85 \,^{\circ}C, V_{DD} = 1.8 \,^{\circ}V \text{ to } 5.5 \,^{\circ}V)$ 

Manufacturer	Series Number <sup>(1)</sup>	Frequency Range	Load Cap (pF)		Load Cap (pF)			r Voltage je (V)	Remarks
			C1	C2	MIN	MAX			
TDK	FCR ðÿM5	3.58 MHz-6.0 MHz	33	33	2.0	5.5	Leaded Type		
	FCR ðÿMC5	3.58 MHz-6.0 MHz	(2)	(2)	2.0	5.5	On-chip C Leaded Type		
	CCR ðÿMC3	3.58 MHz-6.0 MHz	(3)	(3)	2.0	5.5	On-chip C SMD Type		

#### NOTES:

Please specify normal oscillator frequency.
 On-chip C: 30pF built in.
 On-chip C: 38pF built in.



KS57C21832/P21832 ELECTRICAL DATA

**Table 14-5. Subsystem Clock Oscillator Characteristics** 

$$(T_A = -40 \,^{\circ}C + 85 \,^{\circ}C, V_{DD} = 1.8 \,^{\circ}V \text{ to } 5.5 \,^{\circ}V)$$

Oscillator	Clock Configuration	Parameter	Test Condition	Min	Тур	Max	Units
Crystal Oscillator	XTIN XTOUT  C1 C2	Oscillation frequency (1)	_	32	32.768	35	kHz
		Stabilization time (2)	V <sub>DD</sub> = 2.7 V to 5.5 V	_	1.0	2	S
			$V_{DD} = 2.0 \text{ V to } 5.5 \text{ V}$	_	_	10	
External Clock	XTIN XTOUT	XT <sub>IN</sub> input frequency <sup>(1)</sup>	_	32	_	100	kHz
		XT <sub>IN</sub> input high and low level width (t <sub>XTL</sub> , t <sub>XTH</sub> )	-	5	-	15	μs

#### NOTES:

- 1. Oscillation frequency and  $\mathsf{XT}_{\mathsf{IN}}$  input frequency data are for oscillator characteristics only.
- 2. Stabilization time is the interval required for oscillating stabilization after a power-on occurs.

## Table 14-6. Input/Output Capacitance

$$(T_A = 25 \,^{\circ}C, V_{DD} = 0 \,V)$$

Parameter	Symbol	Condition	Min	Тур	Max	Units
Input Capacitance	C <sub>IN</sub>	f = 1 MHz; Unmeasured pins are returned to V <sub>SS</sub>	-	_	15	pF
Output Capacitance	C <sub>OUT</sub>		_	_	15	pF
I/O Capacitance	C <sub>IO</sub>		-	_	15	pF

ELECTRICAL DATA KS57C21832/P21832

Table 14-7. A.C. Electrical Characteristics

(T<sub>A</sub> = -40 °C to +85 °C,  $V_{DD}$  = 1.8 V to 5.5 V)

Parameter	Symbol	Conditions	Conditions Min Typ		Max	Units
Instruction Cycle Time (note)	t <sub>CY</sub>	$V_{DD} = 2.7 \text{ V} \text{ to } 5.5 \text{ V}$	0.67	_	64	μs
		V <sub>DD</sub> = 2.0 V to 5.5 V	0.95		64	
TCL0, TCL1 Input Frequency	f <sub>TIO,</sub> f <sub>TI1</sub>	V <sub>DD</sub> = 2.7 V to 5.5 V	0	_	1.5	MHz
		V <sub>DD</sub> = 2.0 V to 5.5 V			1	
TCL0, TCL1 Input High, Low Width	t <sub>TIH0,</sub> t <sub>TIL0</sub>	$V_{DD} = 2.7 \text{ V} \text{ to } 5.5 \text{ V}$	0.48	_	_	μs
		V <sub>DD</sub> = 2.0 V to 5.5 V	1.8			
SCK Cycle Time	t <sub>KCY</sub>	$V_{DD} = 2.7 \text{ V to } 5.5 \text{ V; Input}$	800	_	_	ns
		Internal SCK source; Output	650			
		$V_{DD} = 2.0 \text{ V} \text{ to } 5.5 \text{ V}; \text{Input}$	3200			
		Internal SCK source; Output	3800			
SCK High, Low Width	t <sub>KH</sub> , t <sub>KL</sub>	$V_{DD} = 2.7 \text{ V}$ to 5.5 V; Input	325	_	_	ns
		Internal SCK source; Output	t <sub>KCY</sub> /2- 50			
		$V_{DD} = 2.0 \text{ V}$ to 5.5 V; Input	1600			
		Internal SCK source; Output	t <sub>KCY</sub> /2- 150			
SI Setup Time to SCK High	t <sub>SIK</sub>	$V_{DD} = 2.7 \text{ V}$ to 5.5 V; Input	100	_	_	ns
		$V_{DD} = 2.7 \text{ V}$ to 5.5 V; Output	150			
		V <sub>DD</sub> = 2.0 V to 5.5 V; Input	150			
		$V_{DD} = 2.0 \text{ V}$ to 5.5 V; Output	500			
SI Hold Time to SCK High	t <sub>KSI</sub>	$V_{DD} = 2.7 \text{ V}$ to 5.5 V; Input	400	_	_	ns
		$V_{DD} = 2.7 \text{ V}$ to 5.5 V; Output	400			
		V <sub>DD</sub> = 2.0 V to 5.5 V; Input	600			
		V <sub>DD</sub> = 2.0 V to 5.5 V; Output	500			

**NOTE:** Unless otherwise specified, Instruction Cycle Time condition values assume a main system clock (fx) source.



KS57C21832/P21832 ELECTRICAL DATA

Table 14-7. A.C. Electrical Characteristics (Continued)

 $(T_A = -40 \,^{\circ}C \text{ to } + 85 \,^{\circ}C, V_{DD} = 1.8 \,\text{V to } 5.5 \,\text{V})$ 

Parameter	Symbol	Conditions	Min	Тур	Max	Units
Output Delay for SCK to SO	t <sub>KSO</sub>	$V_{DD} = 2.7 \text{ V to } 5.5 \text{ V; Input}$	_	-	300	ns
		$V_{DD} = 2.7 \text{ V}$ to 5.5 V; Output			250	
		$V_{DD} = 2.0 \text{ V to } 5.5 \text{ V; Input}$			1000	
		$V_{DD} = 2.0 \text{ V to } 5.5 \text{ V; Output}$			1000	
Interrupt Input High, Low Width	t <sub>INTH</sub> , t <sub>INTL</sub>	INT0, INT1, INT2, INT4, K0-K7	10	_	_	μs
RESET Input Low Width	t <sub>RSL</sub>	Input	10	_	_	μs

 $\textbf{NOTE:} \quad \text{Minimum value for INT0 is based on a clock of } 2t_{CY} \quad \text{or } 128/\text{fx} \quad \text{as assigned by the IMOD0 register setting.}$ 

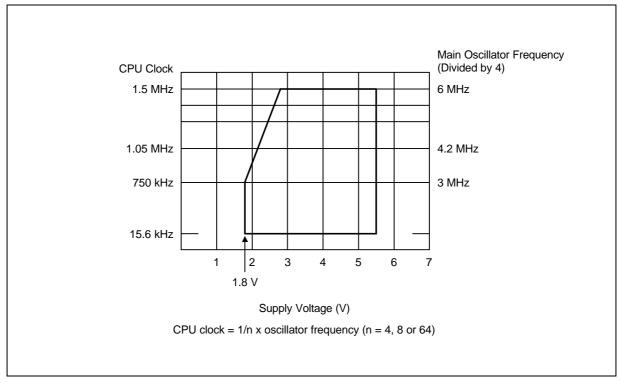


Figure 14-1. Standard Operating Voltage Range

ELECTRICAL DATA KS57C21832/P21832

Table 14-8. RAM Data Retention Supply Voltage in Stop Mode

 $(T_A = -40 \,^{\circ}C \text{ to } + 85 \,^{\circ}C)$ 

Parameter	Symbol	Conditions	Min	Тур	Max	Unit
Data retention supply voltage	$V_{DDDR}$	_	1.8	_	5.5	V
Data retention supply current	I <sub>DDDR</sub>	V <sub>DDDR</sub> = 1.8 V	_	0.1	10	μΑ
Release signal set time	t <sub>SREL</sub>	_	0	_	_	μs
Oscillator stabilization wait time (1)	t <sub>WAIT</sub>	Released by RESET	_	2 <sup>17</sup> /fx	_	ms
		Released by interrupt	_	(2)	_	

#### NOTES:

- During oscillator stabilization wait time, all CPU operations must be stopped to avoid instability during oscillator start-up.
- 2. Use the basic timer mode register (BMOD) interval timer to delay execution of CPU instructions during the wait time.



KS57C21832/P21832 ELECTRICAL DATA

#### **TIMING WAVEFORMS**

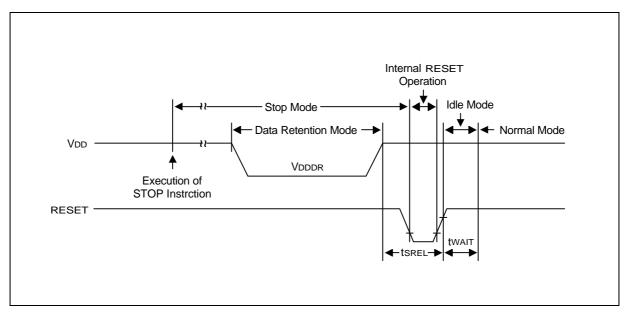


Figure 14-2. Stop Mode Release Timing When Initiated by RESET

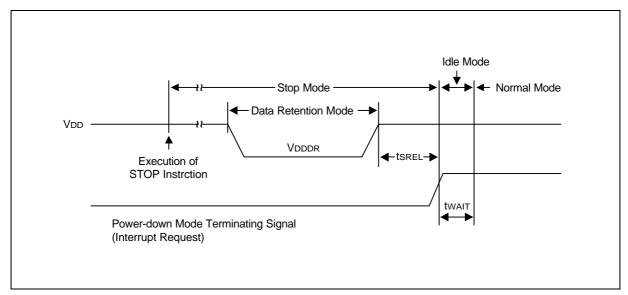


Figure 14-3. Stop Mode Release Timing When Initiated by Interrupt Request



ELECTRICAL DATA KS57C21832/P21832

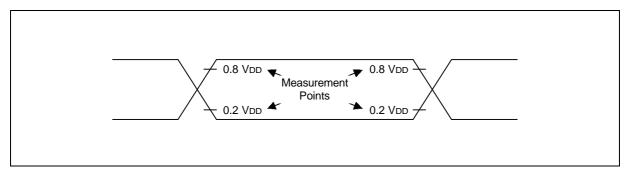


Figure 14-4. A.C. Timing Measurement Points (Except for  $X_{\text{IN}}$  and  $XT_{\text{IN}}$ )

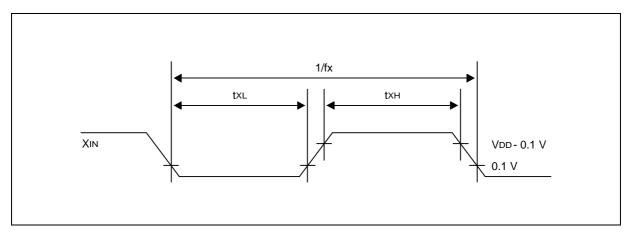


Figure 14-5. Clock Timing Measurement at  $\mathbf{X}_{\text{IN}}$ 

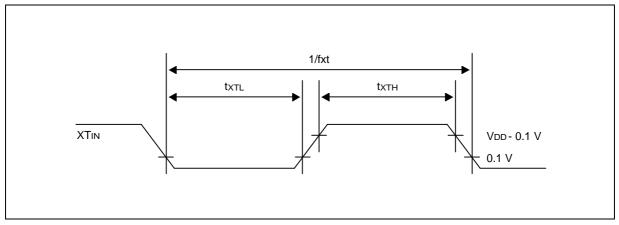


Figure 14-6. Clock Timing Measurement at XT<sub>IN</sub>



KS57C21832/P21832 ELECTRICAL DATA

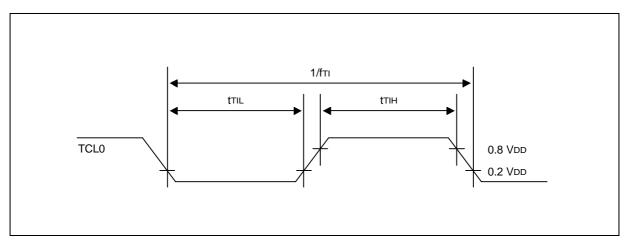


Figure 14-7. TCL Timing

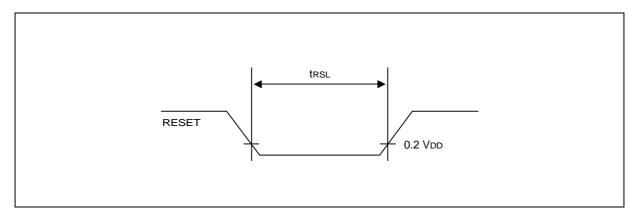


Figure 14-8. Input Timing for RESET Signal

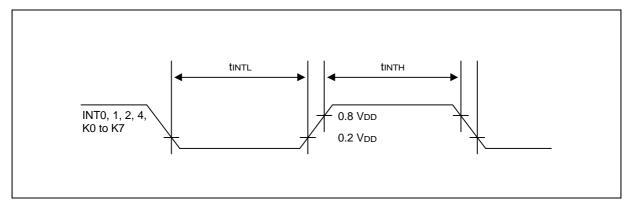


Figure 14-9. Input Timing for External Interrupts and Quasi-Interrupts

ELECTRICAL DATA KS57C21832/P21832

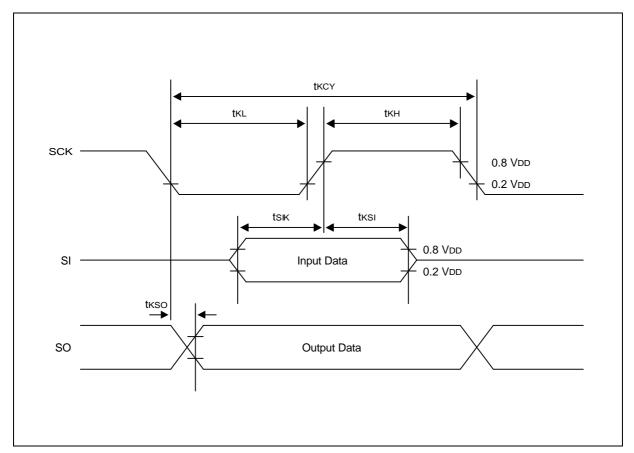


Figure 14-10. Serial Data Transfer Timing

KS57C21832/P21832 MICHANICAL DATA

# 15

# **MICHANICAL DATA**

#### **OVERVIEW**

The KS57C21832 microcontrollers are available in a 100-QFP-1420C package.

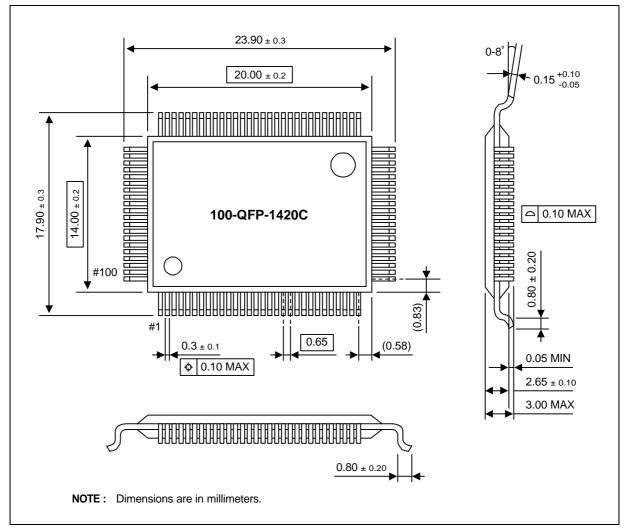


Figure 15-1. 100-QFP Package Dimension



MICHANICAL DATA KS57C21832/P21832

## **NOTES**



KS57C21832/P21832 KS57P21832 OTP

16

# KS57P21832 OTP

#### **OVERVIEW**

The KS57P21832 single-chip CMOS microcontroller is the OTP (One Time Programmable) version of the KS57C21832 microcontroller. It has an on-chip OTP ROM instead of masked ROM. The EPROM is accessed by serial data format.

The KS57P21832 is fully compatible with the KS57C21832, both in function and in pin configuration. Because of its simple programming requirements, the KS57P21832 is ideal for use as an evaluation chip for the KS57C21832.



KS57P21832 OTP KS57C21832/P21832

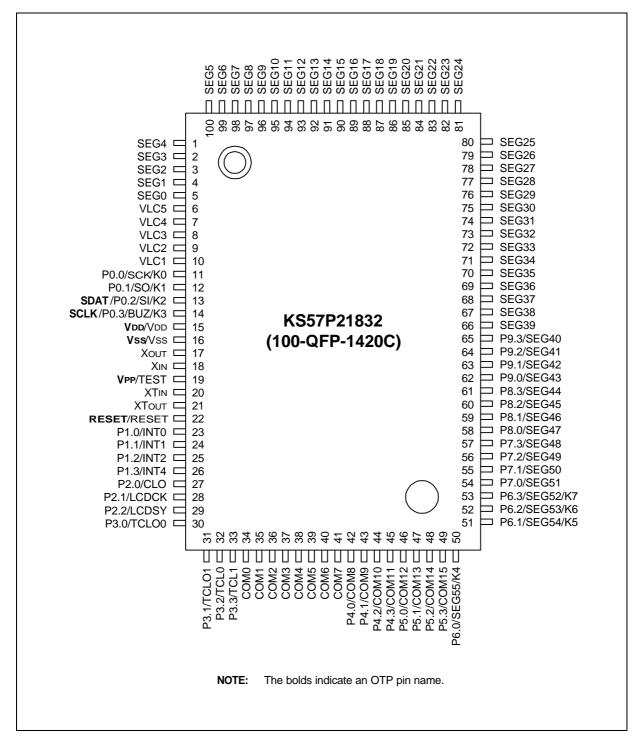


Figure 16-1. KS57P21832 Pin Assignments (100-QFP Package)



KS57C21832/P21832 KS57P21832 OTP

Table 16-1. Descriptions of Pins Used to Read/Write the EPROM

Main Chip	During Programming					
Pin Name	Pin Name	Pin No.	I/O	Function		
P0.2	SDAT	13	I/O	Serial data pin. Output port when reading and input port when writing. Can be assigned as a Input / push-pull output port.		
P0.3	SCLK	14	I/O	Serial clock pin. Input only pin.		
TEST	V <sub>PP</sub> (TEST)	19	1	Power supply pin for EPROM cell writing (indicates that OTP enters into the writing mode). When 12.5 V is applied, OTP is in writing mode and when 5 V is applied, OTP is in reading mode. (Option)		
RESET	RESET	22	- 1	Chip initialization		
V <sub>DD</sub> /V <sub>SS</sub>	V <sub>DD</sub> /V <sub>SS</sub>	15/16	I	Logic power supply pin. V <sub>DD</sub> should be tied to + 5 V during programming.		

Table 16-2. Comparison of KS57P21832 and KS57C21832 Features

Characteristic	KS57P21832	KS57C21832		
Program Memory	32 Kbyte EPROM	32 Kbyte mask ROM		
Operating Voltage (V <sub>DD</sub> )	1.8 V to 5.5 V	1.8 V to 5.5 V		
OTP Programming Mode	V <sub>DD</sub> = 5 V, V <sub>PP</sub> (TEST) = 12.5 V			
Pin Configuration	100 QFP	100 QFP		
EPROM Programmability	User Program 1 time	Programmed at the factory		

#### **OPERATING MODE CHARACTERISTICS**

When 12.5 V is supplied to the  $V_{PP}(TEST)$  pin of the KS57P21832, the EPROM programming mode is entered. The operating mode (read, write, or read protection) is selected according to the input signals to the pins listed in Table 16-3 below.

**Table 16-3. Operating Mode Selection Criteria** 

V <sub>DD</sub>	V <sub>PP</sub> (TEST)	REG/ MEM	Address (A15-A0)	R/W	Mode	
5 V	5 V	0	0000H	1	EPROM read	
	12.5 V	0	0000H	0	EPROM program	
	12.5 V	0	0000H	1	EPROM verify	
	12.5 V	1	0E3FH	0	EPROM read protection	

NOTE: "0" means Low level; "1" means High level.



KS57P21832 OTP KS57C21832/P21832

Table 16-4. D.C. Electrical Characteristics

 $(T_A = -40 \,^{\circ}\text{C} \text{ to } + 85 \,^{\circ}\text{C}, V_{DD} = 1.8 \,^{\circ}\text{V} \text{ to } 5.5 \,^{\circ}\text{V})$ 

Parameter	Symbol	Conditions		Min	Тур	Max	Units
Supply Current	I <sub>DD1</sub> (2)	$V_{DD} = 5 V \pm 10\%$ Crystal oscillator C1 = C2 = 22 pF	6.0 MHz 4.19 MHz	-	5.1 3.9	10.0 7.5	mA
		$V_{DD} = 3 V \pm 10\%$	6.0 MHz 4.19 MHz		2.5 1.8	4.0 3.0	
	I <sub>DD2</sub> (2)	Idle mode; $V_{DD} = 5 \text{ V} \pm 10\%$ Crystal oscillator C1 = C2 = 22  pF	6.0 MHz 4.19 MHz		1.3 1.2	2.5 1.8	
		$V_{DD} = 3 V \pm 10\%$	6.0 MHz 4.19 MHz		0.5 0.44	1.5 1.0	
	I <sub>DD3</sub> (3)	V <sub>DD</sub> = 3 V ± 10% 32 kHz crystal oscillator		_	22.8	35	μΑ
	I <sub>DD4</sub> (3)	Idle mode; V <sub>DD</sub> = 3 V ± 10% 32 kHz crystal oscillator			6.4	15	
	I <sub>DD5</sub>	Stop mode; V <sub>DD</sub> = 5 V ± 10%	SCMOD = 0000B		2.5	5	
		Stop mode; V <sub>DD</sub> = 3 V ± 10%	XT = 0V		0.5	3	
		Stop mode; V <sub>DD</sub> = 5 V ± 10%	SCMOD = 0100B		0.2	3	
		Stop mode; V <sub>DD</sub> = 3 V ± 10%	1		0.1	2	

#### NOTES:

- 1. Data includes power consumption for subsystem clock oscillation.
- 2. When the system clock control register, SCMOD, is set to 1001B, main system clock oscillation stops and the subsystem clock is used.
- 3. Currents in the following circuits are not included; on-chip pull-up resistors, internal LCD voltage dividing resistors, output port drive currents.



KS57C21832/P21832 KS57P21832 OTP

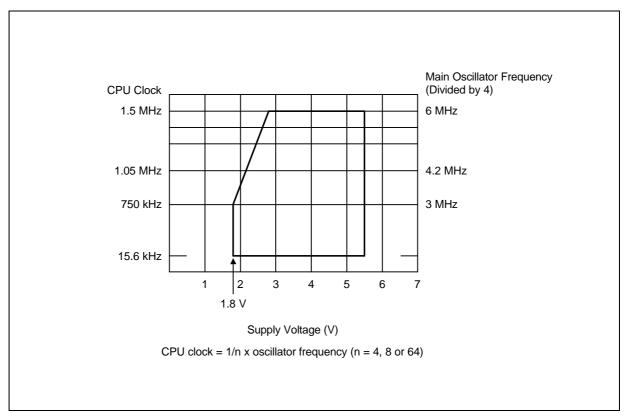


Figure 16-2. Standard Operating Voltage Range

KS57P21832 OTP KS57C21832/P21832

### **NOTES**



KS57C21832/P21832 DEVELOPMENT TOOLS

**17** 

## **DEVELOPMENT TOOLS**

#### **OVERVIEW**

Samsung provides a powerful and easy-to-use development support system in turnkey form. The development support system is configured with a host system, debugging tools, and support software. For the host system, any standard computer that operates with MS-DOS as its operating system can be used. One type of debugging tool including hardware and software is provided: the sophisticated and powerful in-circuit emulator, SMDS2+, for KS57, KS86, KS88 families of microcontrollers. The SMDS2+ is a new and improved version of SMDS2. Samsung also offers support software that includes debugger, assembler, and a program for setting options.

#### SHINE

Samsung Host Interface for In-Circuit Emulator, SHINE, is a multi-window based debugger for SMDS2+. SHINE provides pull-down and pop-up menus, mouse support, function/hot keys, and context-sensitive hyper-linked help. It has an advanced, multiple-windowed user interface that emphasizes ease of use. Each window can be sized, moved, scrolled, highlighted, added, or removed completely.

#### SAMA ASSEMBLER

The Samsung Arrangeable Microcontroller (SAM) Assembler, SAMA, is a universal assembler, and generates object code in standard hexadecimal format. Assembled program code includes the object code that is used for ROM data and required SMDS program control data. To assemble programs, SAMA requires a source file and an auxiliary definition (DEF) file with device specific information.

#### SASM57

The SASM57 is an relocatable assembler for Samsung's KS57-series microcontrollers. The SASM57 takes a source file containing assembly language statements and translates into a corresponding source code, object code and comments. The SASM57 supports macros and conditional assembly. It runs on the MS-DOS operating system. It produces the relocatable object code only, so the user should link object file. Object files can be linked with other object files and loaded into memory.

#### **HEX2ROM**

HEX2ROM file generates ROM code from HEX file which has been produced by assembler. ROM code must be needed to fabricate a microcontroller which has a mask ROM. When generating the ROM code (.OBJ file) by HEX2ROM, the value 'FF' is filled into the unused ROM area upto the maximum ROM size of the target device automatically.

#### **TARGET BOARDS**

Target boards are available for all KS57-series microcontrollers. All required target system cables and adapters are included with the device-specific target board.

#### **OTPs**

One time programmable microcontroller (OTP) for the KS57C21832 microcontroller and OTP programmer (Gang) are now available.



DEVELOPMENT TOOLS KS57C21832/P21832

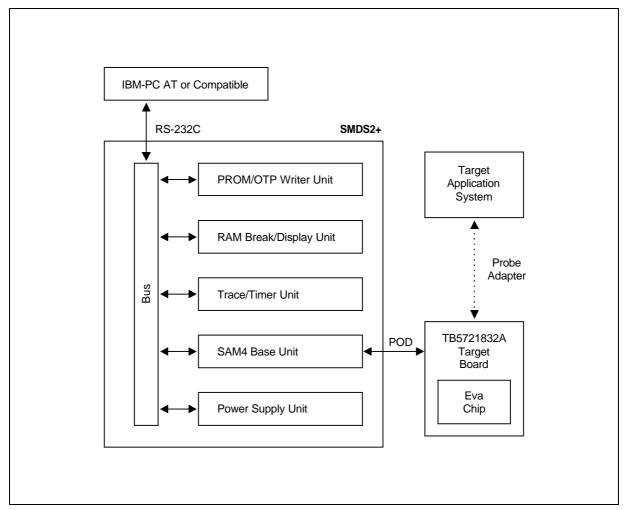


Figure 17-1. SMDS Product Configuration (SMDS2+)

KS57C21832/P21832 DEVELOPMENT TOOLS

#### **TB5721832A TARGET BOARD**

The TB5721832A target board is used for the KS57C21832 microcontroller. It is supported by the SMDS2+ development system.

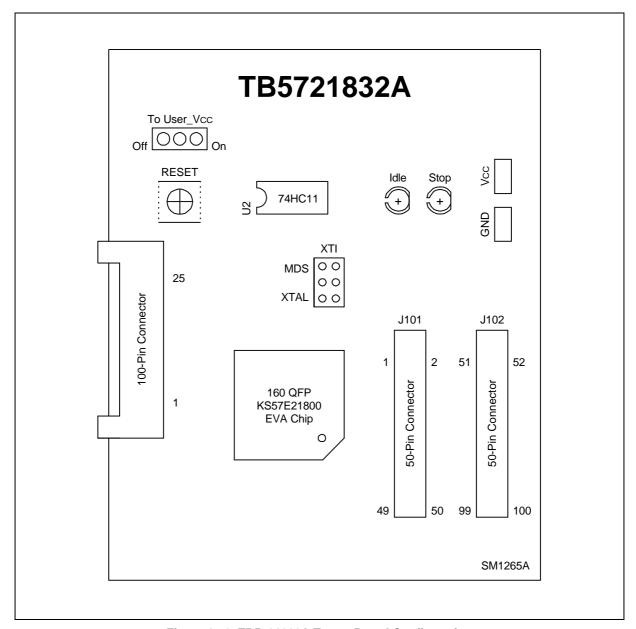


Figure 17-2. TB5721832A Target Board Configuration



DEVELOPMENT TOOLS KS57C21832/P21832

'To User\_Vcc' Settings **Operating Mode** Comments The SMDS2/SMDS2+ To User\_Vcc supplies V<sub>CC</sub> to the target Target Off O On board (evaluation chip) and TB5721832A System the target system. Vss Vcc SMDS2/SMDS2+ The SMDS2/SMDS2+ To User\_Vcc supplies  $V_{CC}$  only to the target External Target board (evaluation chip). The TB5721832A Vcc-System target system must have its Vss own power supply. Vcc I SMDS2+

Table 17-1. Power Selection Settings for TB5721832A



KS57C21832/P21832 DEVELOPMENT TOOLS

**Sub Clock Setting Operating Mode** Comments Set the  $\mathrm{XT}_{\mathrm{IN}}$  switch to "MDS" EVA Chip TXI when the target board is KS57E21800 connected to the MDS C XTAL SMDS2/SMDS2+. No Connection 100 Pin Connector SMDS2/SMDS2+ Set the  $\mathrm{XT}_{\mathrm{IN}}$  switch to "XTAL" when the target board is used EVA Chip MDS O O XTAL as a standalone unit, and is KS57E21800 not connected to the SMDS2/SMDS2+. **XTout XTAL** Target Board

Table 17-2. Sub-Clock Selection Settings for TB5721832A

#### **IDLE LED**

This LED is ON when the evaluation chip (KS57E21800) is in idle mode.

#### **STOP LED**

This LED is ON when the evaluation chip (KS57E21800) is in stop mode.



DEVELOPMENT TOOLS KS57C21832/P21832

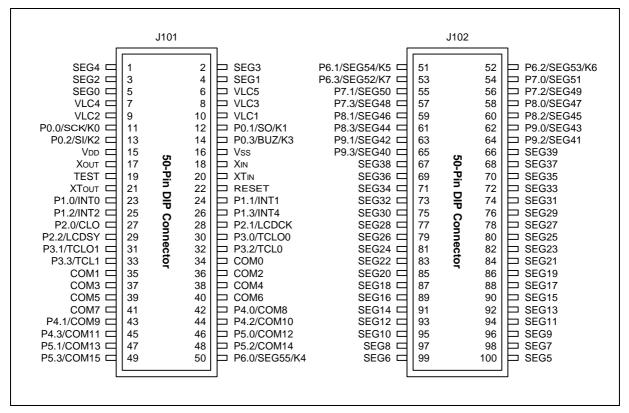


Figure 17-3. 50-Pin Connectors for TB5721832A

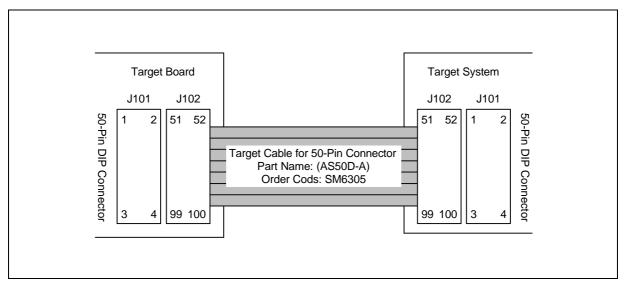


Figure 17-4. TB5721832A Adapter Cable for 100 QFP Package (KS57C21832)

