Sitronix

ST2202

PRELIMINARY

8 BIT Integrated Microcontroller with 256K Bytes ROM

Notice: This is not a final specification. Some parameters are subject to change.

1. FEATURES

- **Totally static 8-bit CPU**
- **ROM: 256K x 8-bit**
- **RAM: 4K x 8-bit**
- **Stack: Up to 128-level deep**
- **Operation voltage: 2.4V ~ 5.5V**
- **Operation frequency:**
  - 3.0Mhz@2.4V(Min.)
  - 4.0Mhz@2.7V(Min.)
- **Low Voltage Detector (LVD)**
- **Memory interface to ROM, RAM, Flash**
- **Memory configuration**
  - Three kinds of bank for program, data and interrupts
  - 12-bit bank register supports up to 44M bytes
  - 6 programmable chip-selects with 4 modes
  - Maximum single device of 16M bytes at $\overline{CS5}$
- **General-Purpose I/O (GPIO) ports**
  - 48 multiplexed CMOS bidirectional bit programmable I/Os
  - Hardware de-bounce option for Port-A
  - Bit programmable pull-up for input pins
  - Bit programmable pull-up/down and open-drain/CMOS for Port-C
- **Programmable Watchdog Timer (WDT)**
- **Timer/Counter**
  - Two 8-bit timer, one can be a 16-bit event counter
  - One 8-bit Base timer with 5 coexistent interrupt time settings
- **Three clocking outputs**
  - Clock sources including Timer0/1, baud rate generator
- **11 prioritized interrupts with dedicated exception vectors**
  - External interrupt (edge triggered)
  - TIMER0 interrupt
  - TIMER1 interrupt
  - BASE timer interrupt
  - PORTA interrupt (transition triggered)
  - DAC reload interrupt
  - LCD frame interrupt
  - SPI interrupts (x2)
  - UART interrupts (x2)
- **Dual clock sources with warm-up timer**
  - Low frequency crystal oscillator (OSCX) ·····32768 Hz
  - RC oscillator (OSC) ····· 500K ~ 4M Hz
  - High frequency crystal/resonator oscillator (Bonding option)····· 455K~4M Hz
- **Direct Memory Access (DMA)**
  - Block-to-Block transfer
  - Block to Single port
- **LCD Controller (LCDC)**
  - Software programmable screen size up to 240X120 (including 160x160, 160x80, etc.)
  - Support 1-, 4-bit LCD data bus
  - Share system memory with display memory
  - Unique internal bus for memory sharing with no loss of the CPU time
  - Diverse functions including virtual screen , panning , scrolling , contrast control and alternating signal generator
  - Support software 16 gray levels
- **Universal Asynchronous Receiver/Transmitter (UART)**
  - Full-duplex operation
  - Baud rate generator with one digital PLL
  - Standard baud rates of 600 bps to 115.2 kbps
  - Direct glueless support of IrDA physical layer protocol
  - Two sets of I/Os (TX,RX) for two independent devices
- **Serial Peripheral Interface (SPI)**
  - Master and slave modes
  - 5 serial signals including enable and data-ready
  - One stage buffer for transmitter and receiver for continuous data exchange
  - Programmable data length from 7-bit to 16-bit
- **Programmable Sound Generator (PSG)**
  - Two channels with three playing modes
  - Tone/noise generator
  - 16-level volume control
  - 8-bit PWM DAC for speech/voice
  - Two dedicated outputs for directly driving and large current
- **Three power down modes**
  - WAI0 mode
  - WAI1 mode
  - STP mode

2. GENERAL DESCRIPTION

The ST2202 is a 8-bit integrated microcontroller designed with CMOS silicon gate technology. The true static CPU core, power down modes and dual oscillators design makes the ST2202 suitable for power saving and long battery life designs. The ST2202 integrates various logic to support functions on-chip which are needed by system designers. This is also important for lower system complexity, small board size and, of course, shorter time to market and less cost.

The ST2202 features the capacity of memory access of maximum 44M bytes which is needed by products with large data bases, and also DMA function for fast memory transfer. Six chip selects are equipped for direct connection to external ROM, SRAM, Flash memory or other devices. Maximum one single device of 16M bytes is possible.

The ST2202 has 48 I/Os grouped into 6 ports, Port-A ~ Port-E and Port-L. Each pin can be programmed to input or output. There are two options: pull-up/down for inputs of Port-C and only pull-up for inputs of the other ports. In case of output, there are open-drain/CMOS options for outputs of PortC and only CMOS for the other ports. Port-A/B is designed for keyboard scan with de-bounce and transition triggered interrupt at Port-A, while Port-C/D/E/L are shared with other system functions. All the properties of I/O pins are still programmable when they are assigned to another function. This enlarges the flexibility of the usage of function signals.

The ability of driving large LCD panels, up to 240x120, and software 16-gray-level support may rich the display information and the diversity of contents as well. This is done with no need of external display RAM because of the internal memory sharing design.

The ST2202 equips serial communication ports of one UART and one SPI to perform different communications, ex.: RS-232 and IrDA, with system components or other products such as PC, Notebook, and popular PDA. Three clocking outputs can produce synthesized PWM signals or high frequency carrier for IR remote control. This helps products become more useful in our daily life.

The built-in two channel PSG/one channel PWM DAC are for the production of key tone, melody, voice, and speech. Two dedicated pins can drive a buzzer directly for minimum cost.

With these integrated functions inside, the ST2202 single chip microcontroller is a right solution for PDA, translator, databank and other consumer products.

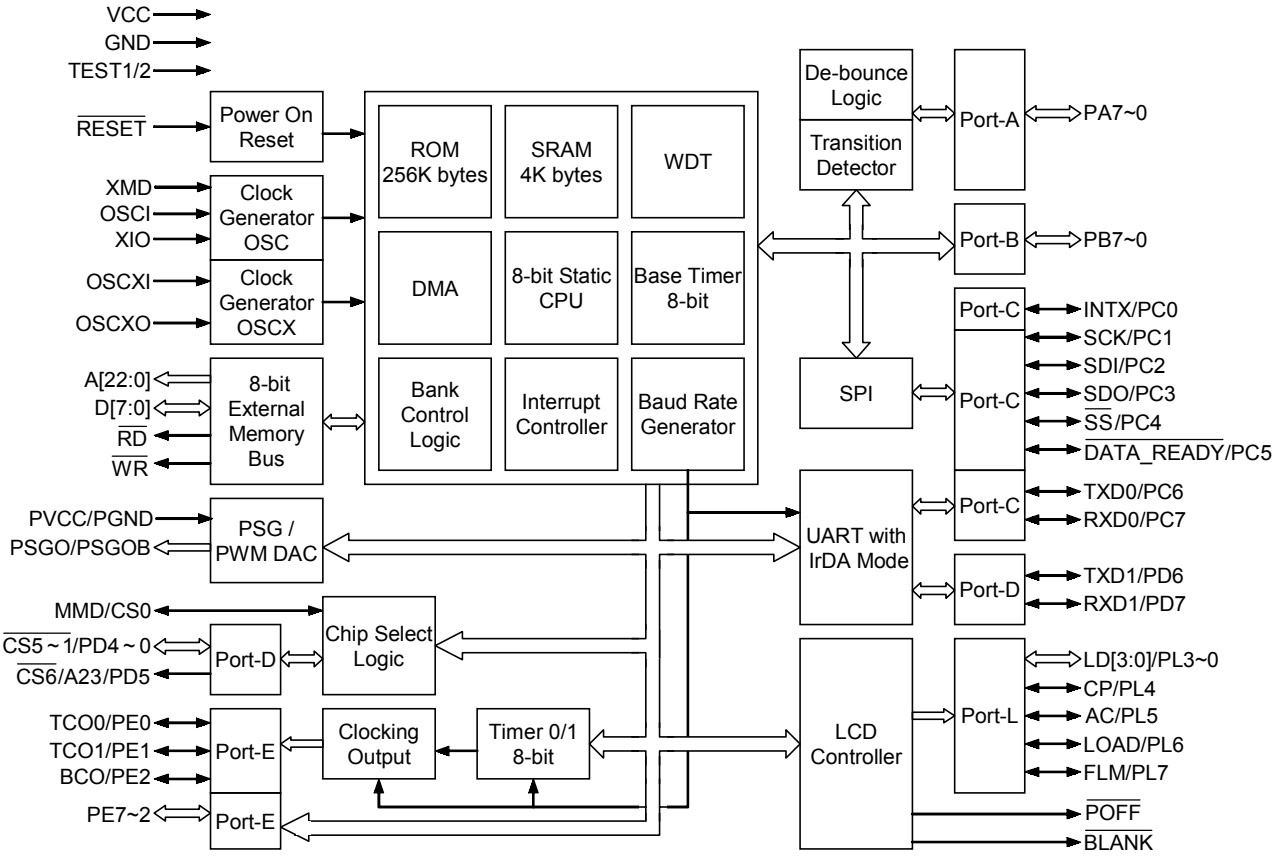


FIGURE 2-1 ST2202 Block Diagram

3. SIGNAL DESCRIPTIONS

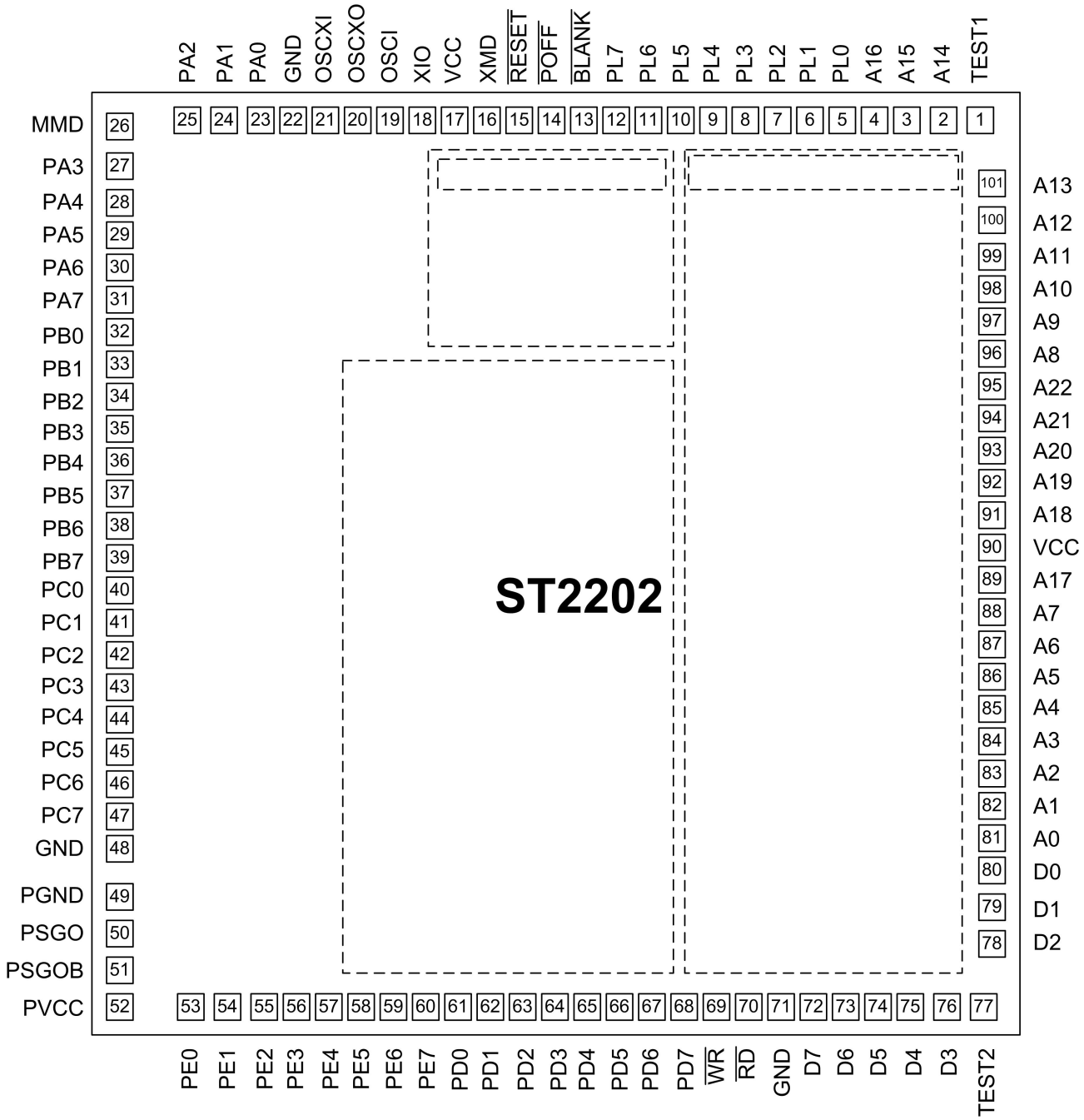
TABLE 3-1 Signal Function Groups

Function Group	Pad No.	Designation	Description
Power	17, 52, 90	VCC , PVCC	**VCC:** Power supply for system **PVCC:** Power supply for PSGO and PSGOB
Ground	22, 48, 49, 71	GND , PGND	**GND:** System power ground **PGND:** Power ground for PSGO and PSGOB
System control	15, 1, 77, 26	$\overline{\text{RESET}}$, TEST1/2 MMD/$\overline{\text{CS0}}$	$\overline{\textbf{RESET}}$ **:** Active low system reset signal input **TEST1/2:** Leave them open when normal operation **MMD/CS0:** Memory modes selection pin **Normal mode:** Enable internal ROM. MMD/$\overline{\text{CS0}}$ connects to GND. **Emulation mode:** Disable internal ROM. MMD/$\overline{\text{CS0}}$ connects to chip-select pin of external ROM. One resistor should be added between VCC and this pin. After reset cycles, MMD/$\overline{\text{CS0}}$ changes to be an output, and outputs signal $\overline{\text{CS0}}$.
Clock	16, 18~21	OSCXO, OSCXI, OSCI, XIO, XMD	**XMD:** High frequency oscillator (OSC) mode selection input **Low:** Crystal mode. One crystal or resonator should be connected between OSCI and XIO **High:** Resistor oscillator mode. One resistor should be connected between OSCI and VCC **OSCXI, OSCXO:** Connect one 32768Hz crystal between these two pins when using low frequency oscillator
External memory bus signals	69, 70	$\overline{\text{WR}}$, $\overline{\text{RD}}$	External memory R/W control signals
	2~4, 81~89, 91~101	A[22:0]	External memory address bus
	72~76, 78~80	D[7:0]	External memory data bus
PSG/PWM DAC	50, 51	PSGO, PSGOB	PSG outputs. Connect to one buzzer or speaker
Keyboard scan signal (return line)	23~25, 27~31	PA7~0	I/O port A
GPIO	32~39	PB7~0	I/O port B
Chip selects	61~66	$\overline{\text{CS6}}$ /A23/PD5 , $\overline{\text{CS5}}$ ~1/PD4~0	I/O port D and chip-select outputs
UART	46, 47, 67, 68	RXD0/PC7 , TXD0/PC6 , RXD1/PD7 , TXD1/PD6	UART signals and I/Os
SPI	41~45	$\overline{\text{DATA\_READY}}$ /PC5 , $\overline{\text{SS}}$ /PC4 , SDO/PC3 , SDI/PC2 , SCK/PC1	SPI signals and I/Os

TABLE 3-2 Signal Function Groups (continued)

Function Group	Pad No.	Designation	Description
External interrupt	40	INTX/PC0	External interrupt inputs
Clocking output	53~55	BCO/PE2 , TCO1/PE1 , TCO0/PE0	Clocking outputs
GPIO	56~60	PE7~3	I/O port E
LCD controller	5~14	FLM/PL7, LOAD/PL6, AC/PL5 , CP/PL4, $\overline{\text{LD[3:0]}}$/PL3~0, $\overline{\text{POFF}}$, $\overline{\text{BLANK}}$	LCD control signals

4. PAD DIAGRAM



5. DEVICE INFORMATION

1. **Pad size: 90um x 90um**
2. **Substrate: GND**
3. **Chip size: 3160um x 3210um**

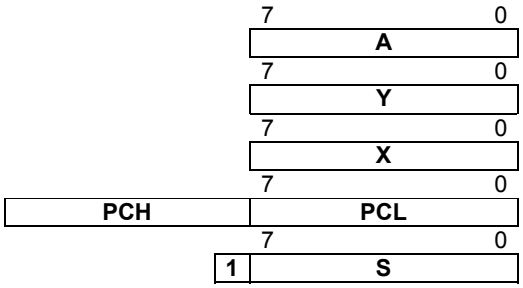
PAD No.	Symbol	X	Y
1	TEST1	1445	1510
2	A14	1320	1510
3	A15	1195	1510
4	A16	1085	1510
5	PL0	975	1510
6	PL1	865	1510
7	PL2	755	1510
8	PL3	645	1510
9	PL4	535	1510
10	PL5	425	1510
11	PL6	315	1510
12	PL7	205	1510
13	$\overline{\text{BLANK}}$	95	1510
14	$\overline{\text{POFF}}$	-15	1510
15	$\overline{\text{RESET}}$	-125	1510
16	XMD	-235	1510
17	VCC	-345	1510
18	XIO	-455	1510
19	OSCI	-565	1510
20	OSCXO	-675	1510
21	OSCXI	-785	1510
22	GND	-895	1510
23	PA0	-1005	1510
24	PA1	-1130	1510
25	PA2	-1255	1510
26	MMD	-1485	1489
27	PA3	-1485	1354
28	PA4	-1485	1229
29	PA5	-1485	1119
30	PA6	-1485	1009
31	PA7	-1485	899
32	PB0	-1485	789
33	PB1	-1485	679
34	PB2	-1485	569

PAD No.	Symbol	X	Y
35	PB3	-1485	459
36	PB4	-1485	349
37	PB5	-1485	239
38	PB6	-1485	129
39	PB7	-1485	19
40	PC0	-1485	-91
41	PC1	-1485	-201
42	PC2	-1485	-311
43	PC3	-1485	-421
44	PC4	-1485	-531
45	PC5	-1485	-641
46	PC6	-1485	-751
47	PC7	-1485	-861
48	GND	-1485	-971
49	PGND	-1485	-1135
50	PSGO	-1485	-1260
51	PSGOB	-1485	-1385
52	PVCC	-1485	-1510
53	PE0	-1243.4	-1510
54	PE1	-1118.4	-1510
55	PE2	-993.4	-1510
56	PE3	-883.4	-1510
57	PE4	-773.4	-1510
58	PE5	-663.4	-1510
59	PE6	-553.4	-1510
60	PE7	-443.4	-1510
61	PD0	-333.4	-1510
62	PD1	-223.4	-1510
63	PD2	-113.4	-1510
64	PD3	-3.4	-1510
65	PD4	106.6	-1510
66	PD5	216.6	-1510
67	PD6	326.6	-1510
68	PD7	436.6	-1510

PAD No.	Symbol	X	Y
69	$\overline{\text{WR}}$	546.6	-1510
70	$\overline{\text{RD}}$	656.6	-1510
71	GND	766.6	-1510
72	D7	876.6	-1510
73	D6	986.6	-1510
74	D5	1096.6	-1510
75	D4	1206.6	-1510
76	D3	1331.6	-1510
77	TEST2	1456.6	-1510
78	D2	1485	-1294.9
79	D1	1485	-1169.9
80	D0	1485	-1044.9
81	A0	1485	-934.9
82	A1	1485	-824.9
83	A2	1485	-714.9
84	A3	1485	-604.9
85	A4	1485	-494.9
86	A5	1485	-384.9
87	A6	1485	-274.9
88	A7	1485	-164.9
89	A17	1485	-54.9
90	VCC	1485	55.1
91	A18	1485	165.1
92	A19	1485	275.1
93	A20	1485	385.1
94	A21	1485	495.1
95	A22	1485	605.1
96	A8	1485	715.1
97	A9	1485	825.1
98	A10	1485	935.1
99	A11	1485	1045.1
100	A12	1485	1170.1
101	A13	1485	1295.1

6. CPU

■ Register Model



Accumulator A

Index Register Y

Index Register X

Program Counter PC

Stack Pointer S

■ Accumulator (A)

The Accumulator is a general-purpose 8-bit register that stores the results of most arithmetic and logic operations. In addition, the accumulator usually contains one of the two data words used in these operations.

■ Index Registers (X,Y)

There are two 8-bit Index Registers (**X** and **Y**), which may be used to count program steps or to provide and index value to be used in generating an effective address. When executing an instruction, which specifies indexed addressing, the CPU fetches the OP code and the base address, and modifies the address by adding the index register to it prior to performing the desired operation. Pre or post-indexing of indirect addresses is possible.

■ Stack Pointer (S)

The Stack Pointer is an 8-bit register, which is used to control the addressing of the variable-length stack. It's range from 100H to 1FFH total for 256 bytes (128 level deep). The stack pointer is automatically increment and decrement under control of the microprocessor to perform stack manipulations under direction of either the program or interrupts (IRQ). The stack allows simple implementation of nested subroutines and multiple level interrupts. The stack pointer is initialized by the user's software.

■ Program Counter (PC)

The 16-bit Program Counter register provides the address, which step the microprocessor through sequential program instructions. Each time the microprocessor fetches and instruction from program memory, the lower byte of the program counter (**PCL**) is placed on the low-order bits of the address bus and the higher byte of the program counter (**PCH**) is placed on the high-order 8 bits. The counter is increment each time an instruction or data is fetched from program memory.

■ Status Register (P)

The 8-bit Processor Status Register contains seven status flags. Some of these flags are controlled by program; others may be also controlled by the CPU as well. The instruction set contains a member of conditional branch instructions that are designed to allow testing of these flags. Refer to TABLE 6-1

TABLE 6-1 Status Register (P)

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
N	V	1	B	D	I	Z	C
Bit 7: **N :** Signed flag by arithmetic 1 = Negative 0 = Positive				Bit 3: **D :** Decimal mode flag 1 = Decimal mode 0 = Binary mode			
Bit 6: **V :** Overflow of signed Arithmetic flag 1 = Negative 0 = Positive				Bit 2: **I :** Interrupt disable flag 1 = Interrupt disable 0 = Interrupt enable			
				Bit 1: **Z :** Zero flag 1 = Zero 0 = Non zero			
Bit 4: **B : BRK** interrupt flag 1 = **BRK** interrupt occur 0 = Non **BRK** interrupt occur				Bit 0: **C :** Carry flag 1 = Carry 0 = Non carry			

7. MEMORY CONFIGURATION

7.1 Memory map

The logical memory space of ST2202 is divided into 3 parts: $0000~$0FFF (4K), $4000~$7FFF (16K), and $8000~FFFF (32K). First is for control registers, stack and system memory. Second and third are banked areas. Logical address in these two areas combines two bank registers, **PRR** and **DRR** respectively, and then be mapped to a physical address. **PRR** is the Program ROM Bank Register and is 12-bit long, while **DRR** is the Data ROM Bank Register of the length of 11 bits. Both can refer to a maximum space of 64M bytes.

Only 44M (28M when **CSM0**="0") bytes is addressable by chip selects.

Refer to FIGURE 7-1 for memory mapping of ST2202.

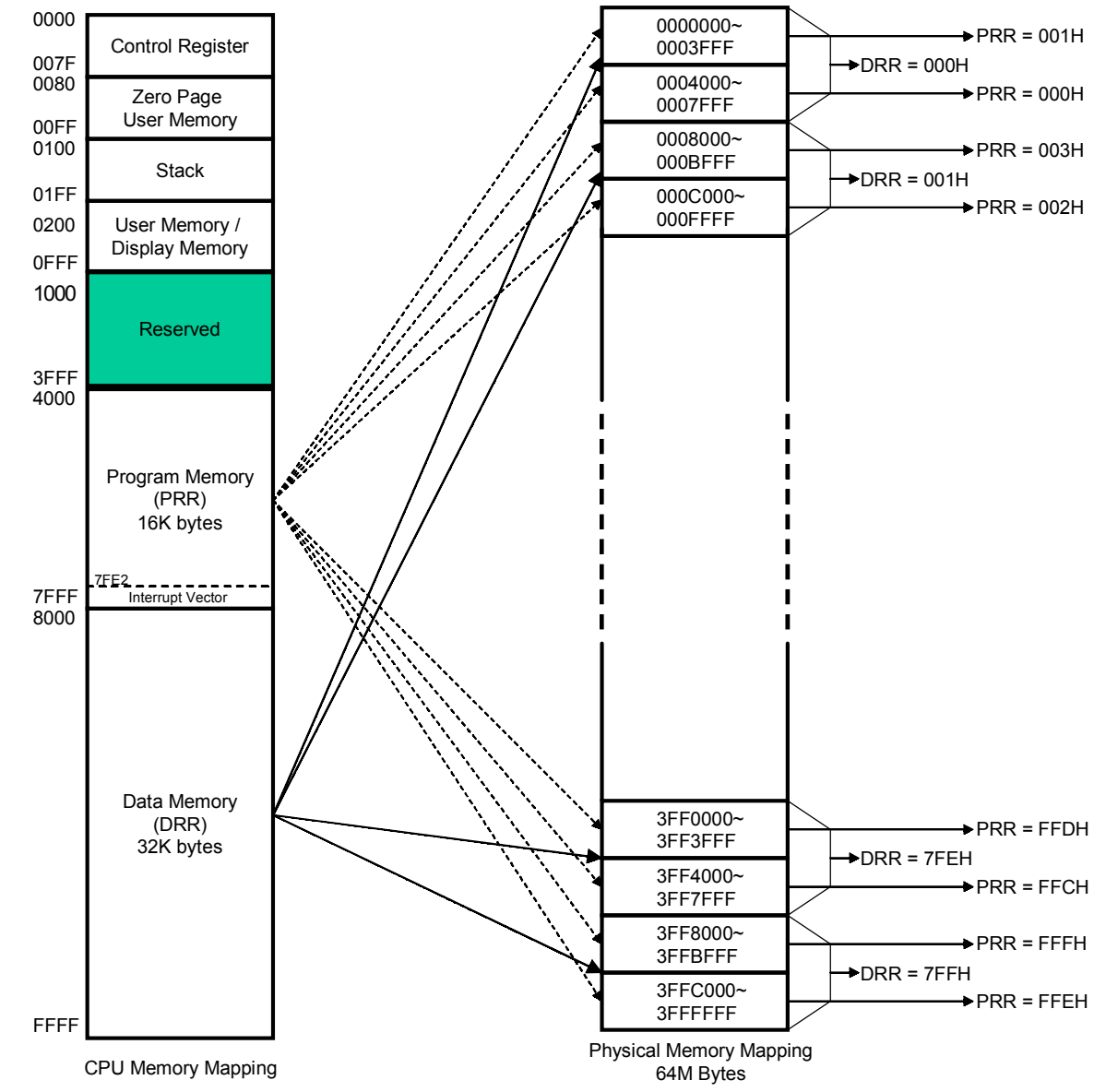


FIGURE 7-1 Memory Mapping

7.2 Control Registers

Address $000~$07F is for control registers. Refer to TABLE 7-1 for summary of all registers. There are more details of registers in the related sections.

TABLE 7-1 Control Registers Summary

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$000	PA*	R/W	PA[7]	PA[6]	PA[5]	PA[4]	PA[3]	PA[2]	PA[1]	PA[0]	1111 1111
$001	PB*	R/W	PB[7]	PB[6]	PB[5]	PB[4]	PB[3]	PB[2]	PB[1]	PB[0]	1111 1111
$002	PC*	R/W	PC[7]	PC[6]	PC[5]	PC[4]	PC[3]	PC[2]	PC[1]	PC[0]	1111 1111
$003	PD*	R/W	PD[7]	PD[6]	PD[5]	PD[4]	PD[3]	PD[2]	PD[1]	PD[0]	1111 1111
$004	PE*	R/W	PE[7]	PE[6]	PE[5]	PE[4]	PE[3]	PE[2]	PE[1]	PE[0]	1111 1111
$005	PSC	R/W	PSC[7]	PSC[6]	PSC[5]	PSC[4]	PSC[3]	PSC[2]	PSC[1]	PSC[0]	1111 1111
$008	PCA	R/W	PCA[7]	PCA[6]	PCA[5]	PCA[4]	PCA[3]	PCA[2]	PCA[1]	PCA[0]	0000 0000
$009	PCB	R/W	PCB[7]	PCB[6]	PCB[5]	PCB[4]	PCB[3]	PCB[2]	PCB[1]	PCB[0]	0000 0000
$00A	PCC	R/W	PCC[7]	PCC[6]	PCC[5]	PCC[4]	PCC[3]	PCC[2]	PCC[1]	PCC[0]	0000 0000
$00B	PCD	R/W	PCD[7]	PCD[6]	PCD[5]	PCD[4]	PCD[3]	PCD[2]	PCD[1]	PCD[0]	0000 0000
$00C	PCE	R/W	PCE[7]	PCE[6]	PCE[5]	PCE[4]	PCE[3]	PCE[2]	PCE[1]	PCE[0]	0000 0000
$00D	PFC	R/W	RXD0	TXD0	SRDY	SS	MOSI	MISO	SCK	INTX	0000 0000
$00E	PFD	R/W	RXD1	TXD1	CS6	CS5	CS4	CS3	CS2	CS1	0000 0000
$00F	PMCR	R/W	PULL	PDBN	INTEG	CSM1	CSM0	BCO	TCO1	TCO0	1000 -000
$010	PSG0L	R/W	PSG0[7]	PSG0[6]	PSG0[5]	PSG0[4]	PSG0[3]	PSG0[2]	PSG0[1]	PSG0[0]	0000 0000
$011	PSG0H	R/W	-	-	-	-	PSG0[11]	PSG0[10]	PSG0[9]	PSG0[8]	- - - - 0000
$012	PSG1L	R/W	PSG1[7]	PSG1[6]	PSG1[5]	PSG1[4]	PSG1[3]	PSG1[2]	PSG1[1]	PSG1[0]	0000 0000
$013	PSG1H	R/W	-	-	-	-	PSG1[11]	PSG1[10]	PSG1[9]	PSG1[8]	- - - - 0000
$014	DAC	R/W	DAC[7]	DAC[6]	DAC[5]	DAC[4]	DAC[3]	DAC[2]	DAC[1]	DAC[0]	0000 0000
$016	PSGC	R/W	-	PCK[2]	PCK[1]	PCK[0]	PRBS	C1EN	C0EN	DACE=0	-000 0000
		R/W	-	PCK[2]	PCK[1]	PCK[0]	DMD[1]	DMD[0]	INH	DACE=1	-000 0000
$017	VOL	R/W	VOL1[3]	VOL1[2]	VOL1[1]	VOL1[0]	VOL0[3]	VOL0[2]	VOL0[1]	VOL0[0]	0000 0000
$020	BTEN	R/W	-	-	-	BTEN[4]	BTEN[3]	BTEN[2]	BTEN[1]	BTEN[0]	- - -0 0000
$021	BTSR*	R	-	-	-	BTSR[4]	BTSR[3]	BTSR[2]	BTSR[1]	BTSR[0]	- - -0 0000
		W	BTCLR	-	-	-	-	-	-	-	0- - - - - - -
$023	PRS*	R	PRS[7]	PRS[6]	PRS[5]	PRS[4]	PRS[3]	PRS[2]	PRS[1]	PRS[0]	0000 0000
		W	SRES	SENA	SENT	-	-	-	-	-	000 - - - - -
$024	T0M	R/W	-	-	T0M[5]	T0M[4]	-	T0M[2]	T0M[1]	T0M[0]	- -00 -000
$025	T0C	R/W	T0C[7]	T0C[6]	T0C[5]	T0C[4]	T0C[3]	T0C[2]	T0C[1]	T0C[0]	0000 0000
$026	T1M	R/W	-	-	-	T1M[4]	T1M[3]	T1M[2]	T1M[1]	T1M[0]	- - -0 0000
$027	T1C	R/W	T1C[7]	T1C[6]	T1C[5]	T1C[4]	T1C[3]	T1C[2]	T1C[1]	T1C[0]	0000 0000
$028	DMSL*	W	DMS[7]	DMS[6]	DMS[5]	DMS[4]	DMS[3]	DMS[2]	DMS[1]	DMS[0]	???? ????
$029	DMSH*	W	DMS[15]	DMS[14]	DMS[13]	DMS[12]	DMS[11]	DMS[10]	DMS[9]	DMS[8]	???? ????
$02A	DMDL*	W	DMD[7]	DMD[6]	DMD[5]	DMD[4]	DMD[3]	DMD[2]	DMD[1]	DMD[0]	???? ????
$02B	DMDH*	W	DMD[15]	DMD[14]	DMD[13]	DMD[12]	DMD[11]	DMD[10]	DMD[9]	DMD[8]	???? ????
$02C	DCNTL*	W	DCNT[7]	DCNT[6]	DCNT[5]	DCNT[4]	DCNT[3]	DCNT[2]	DCNT[1]	DCNT[0]	???? ????
$02D	DCNTH*	W	-	-	-	DMAM	DCNT[11]	DCNT[10]	DCNT[9]	DCNT[8]	- - -? ????
$030	SYS*	R	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	HIGH	0000 0001
		W	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	LVDEN	0000 0000
$031	IRR	R/W	IRR[7]	IRR[6]	IRR[5]	IRR[4]	IRR[3]	IRR[2]	IRR[1]	IRR[0]	0000 0000
$032	PRRL	R/W	PRR[7]	PRR[6]	PRR[5]	PRR[4]	PRR[3]	PRR[2]	PRR[1]	PRR[0]	0000 0000
$033	PRRH	R/W	-	-	-	-	PRR[11]	PRR[10]	PRR[9]	PRR[8]	- - - - 0000
$034	DRRL	R/W	DRR[7]	DRR[6]	DRR[5]	DRR[4]	DRR[3]	DRR[2]	DRR[1]	DRR[0]	0000 0000
$035	DRRH	R/W	-	-	-	-	-	DRR[10]	DRR[9]	DRR[8]	- - - - -000
$036	DMRL	R/W	DMR[7]	DMR[6]	DMR[5]	DMR[4]	DMR[3]	DMR[2]	DMR[1]	DMR[0]	0000 0000
$037	DMRH	R/W	-	-	-	-	-	DMR[10]	DMR[9]	DMR[8]	- - - - -000
$038	MISC	R/W	-	-	-	-	WDTEN	WDTPS	TEST	TEST	- - - - 0000
		W	Reset WDT								
$03C	IREQL	R/W	-	IRLCD	IRBT	IRPT	IRT1	IRT0	IRDAC	IRX	- 000 0000
$03D	IREQH	R/W	-	-	-	-	IRURX	IRUTX	IRSRX	IRSTX	- - - - 0000
$03E	IENAL	R/W	-	IELCD	IEBT	IEPT	IET1	IET0	IEDAC	IEX	- 000 0000
$03F	IENAH	R/W	-	-	-	-	IEURX	IEUTX	IESRX	IESTX	- - - - 0000

$040	**LSSAL***	W	SSA[7]	SSA[6]	SSA[5]	SSA[4]	SSA[3]	SSA[2]	SSA[1]	SSA[0]	0000 0000
$041	**LSSAH***	W	SSA[15]	SSA[14]	SSA[13]	SSA[12]	SSA[11]	SSA[10]	SSA[9]	SSA[8]	0000 0000
$042	**LVPW***	W	VP[7]	VP[6]	VP[5]	VP[4]	VP[3]	VP[2]	VP[1]	VP[0]	0000 0000
$043	**LXMAX**	R/W	XM[7]	XM[6]	XM[5]	XM[4]	XM[3]	XM[2]	XM[1]	XM[0]	0000 0000
$044	**LYMAX**	R/W	YM[7]	YM[6]	YM[5]	YM[4]	YM[3]	YM[2]	YM[1]	YM[0]	0000 0000
$045	**LPAN**	R/W	-	-	-	-	-	PAN[2]	PAN[1]	PAN[0]	- - - - -000
$047	**LCTR**	R/W	LPWR	BLNK	REV	-	-	-	-	-	100- - - - -
$048	**LCKR***	W	-	-	-	LMOD	LCK[3]	LCK[2]	LCK[1]	LCK[0]	- - -0 0000
$049	**LFRA***	W	-	-	FRA[5]	FRA[4]	FRA[3]	FRA[2]	FRA[1]	FRA[0]	- - 00 0000
$04A	**LAC**	R/W	-	-	-	AC[4]	AC[3]	AC[2]	AC[1]	AC[0]	- - -0 0000
$04B	**LPWM**	R/W	-	-	LPWM[5]	LPWM[4]	LPWM[3]	LPWM[2]	LPWM[1]	LPWM[0]	- - 00 0000
$04C	**PL***	R/W	PL[7]	PL[6]	PL[5]	PL[4]	PL[3]	PL[2]	PL[1]	PL[0]	1111 1111
$04E	**PCL***	W	PCL[7]	PCL[6]	PCL[5]	PCL[4]	PCL[3]	PCL[2]	PCL[1]	PCL[0]	0000 0000
$050	**SDAT0AL**	R/W	SD[7]	SD[6]	SD[5]	SD[4]	SD[3]	SD[2]	SD[1]	SD[0]	???? ????
$051	**SDATAH**	R/W	SD[15]	SD[14]	SD[13]	SD[12]	SD[11]	SD[10]	SD[9]	SD[8]	???? ????
$052	**SCTR**	R/W	SPIEN	RXIEN	ERIEN	MEREN	DRINV	POL	PHA	SMOD	0000 0000
$053	**SCKR**	R/W	-	SCK[2]	SCK[1]	SCK[0]	BC[3]	BC[2]	BC[1]	BC[0]	-000 0000
$054	**SSR***	R	-	RXRDY	TXEMP	SBZ	-	MDERR	OERR	BCERR	-000 -000
		W	Write any value to clear SSR								
$060	**UCTR**	R/W	-	-	-	-	PEN	PMOD	UMOD	BRK	- - - - 0000
$061	**USTR***	R	-	FER	PER	OER	RXBZ	RXEN	TXBZ	TXEN	-000 0000
		W	-	-	-	-	RXTRG	RXEN	TXTRG	TXEN	- - - - 0000
$062	**IRCTR**	R/W	RXINV	TXINV	-	-	-	PW1	PW0	IREN	00- - -000
$063	**BCTR**	R/W	TEST	-	-	-	-	BSTR	BMOD	BGREN	0- - - -000
$064	**UDATA**	R/W	UD[7]	UD[6]	UD[5]	UD[4]	UD[3]	UD[2]	UD[1]	UD[0]	???? ????
$066	**BRS**	R/W	BRS[7]	BRS[6]	BRS[5]	BRS[4]	BRS[3]	BRS[2]	BRS[1]	BRS[0]	???? ????
$067	**BDIV**	R/W	BDIV[7]	BDIV[6]	BDIV[5]	BDIV[4]	BDIV[3]	BDIV[2]	BDIV[1]	BDIV[0]	???? ????

Note: 1. Undefined bytes and bits should not be used.

* **Do not use read-modify-write instructions, RMBx and SMBx, to write-only registers.**

7.3 Bank Registers

There are four kinds of bank registers, interrupt bank register (**IRR**), program ROM bank register (**PRR**), data ROM bank register (**DRR**), and DMA source data bank register (**DMR**). **IRR**, **PRR** refer to logic address range of $4000~$7FFF, while **DRR**, **DMR** refer to the range of $8000~$FFFF. The register length, addressable range, and size are listed in TABLE 7-2. When normal process is running, address falls in one of the two areas will activate either **PRR** or **DRR**.

In the case of interrupts, bit[11:8] of **PRR** will be masked to zero and bit[7:0] will be replaced by **IRR**. This replacement lasts until instruction RTI is met. That is, the interrupt vectors and service routines will all base on **IRR**. Operation of **IRR** is also enabled by **IRREN** of **SYS**.

Although a maximum number of 64M bytes can be addressed, the physical size is lower than that because of the limit of chip selects. Please refer to section 10 for more details.

TABLE 7-2 Bank Registers and Addressable Range

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Addressable Range	Size
$031	**IRR**	R/W	IRR[7]	IRR[6]	IRR[5]	IRR[4]	IRR[3]	IRR[2]	IRR[1]	IRR[0]	$0000000~ $03FFFFF	4M
$032	**PRRL**	R/W	PRR[7]	PRR[6]	PRR[5]	PRR[4]	PRR[3]	PRR[2]	PRR[1]	PRR[0]	$0000000~ $3FFFFFF	64M*
$033	**PRRH**	R/W	-	-	-	-	PRR[11]	PRR[10]	PRR[9]	PRR[8]		
$034	**DRRL**	R/W	DRR[7]	DRR[6]	DRR[5]	DRR[4]	DRR[3]	DRR[2]	DRR[1]	DRR[0]	$0000000~ $3FFFFFF	64M*
$035	**DRRH**	R/W	-	-	-	-	-	DRR[10]	DRR[9]	DRR[8]		
$036	**DMRL**	R/W	DMR[7]	DMR[6]	DMR[5]	DMR[4]	DMR[3]	DMR[2]	DMR[1]	DMR[0]	$0000000~ $3FFFFFF	64M*
$037	**DMRH**	R/W	-	-	-	-	-	DMR[10]	DMR[9]	DMR[8]		

Note: * Please refer to section 10 for the limit of addressable size.

TABLE 7-3 System Control Register SYS

Address	Name		Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$030	**SYS**	R	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	HIGH	0000 0001
		W	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	LVDEN	0000 0000

Bit 1: **IRREN** : Enable/Disable Bank register IRR
0 = Disable IRR
1 = Enable IRR

7.4 RAM

Internal static RAM can be divided into 3 parts in function. First is the zero page memory, second is for stack, and third can be used as LCD frame buffer or for general purpose.

■ Zero Page Data RAM ($0080~$00FF)

Total 128 bytes of data RAM in zero page is very useful for programmers. They provide short instruction codes and cycles. Use zero page addressing mode on the variables in this area usually speeds up the overall performance.

■ Stack RAM ($0100~$01FF)

The ST2202 has 256 bytes stack from $0100 to $01FF. It provides a maximum of 128 levels for subroutines. By setting stack pointer carefully, stack memory can also be used as data memory.

■ User Memory and LCD Frame Buffer ($0200~$0FFF)

The ST2202 shares memory for both user memory and LCD frame buffer. The range of LCD frame buffer will be fixed after initialization of LCD control registers. Memory beyond is user memory. Read and write operations can be applied to LCD frame buffer to maintain display content, and almost none of the CPU time is affected. This is contributed by one special memory transfer technique of display data from LCD frame buffer to the LCD controller.

8. INTERRUPT CONTROLLER

The ST2202 supports 11 hardware internal/external interrupts as well as one software interrupt Brk. There are 12 exception vectors for these interrupts and another one for reset. All interrupts are controlled by interrupt disable flag "**I**" (bit2 of status register **P**), and initiate if "**I**" equals "0". Hardware interrupts are further controlled by interrupt enable register **IENA**. Setting bits of **IENA** enables respective interrupts.

The interrupt controller owns one priority arbitrator. When more than one interrupts happen at the same time, the one with lower priority number will be executed first. Refer to TABLE 8-1 for priorities of interrupts.

Once an interrupt event was enabled and then happens, the CPU wakes up (if in either wait mode), and associated bit of interrupt request register (**IREQ**) will be set. If "**I**" flag is cleared, the related vector will be fetched and then the interrupt service routine (ISR) will be executed. Interrupt request flag can be cleared by two methods. One is to write "0" to **IREQ**, the other is to initiate related interrupt service routine. Hardware will automatically clear the Interrupt request flag. All interrupt vectors are listed in TABLE 8-1.

TABLE 8-1 Interrupt Vectors

Name	Signal Source	Vector Address	Priority	Description
BRK	Internal	$7FFF,$7FFE	1	Software BRK operation vector
RESET	External	$7FFD,$7FFC	0	Reset vector
-	-	$7FFB,$7FFA	-	Reserved
INTX	External	$7FF9,$7FF8	6	PC0 edge interrupt
DAC	Internal	$7FF7,$7FF6	7	Reload DAC data interrupt
T0	Internal/External	$7FF5,$7FF4	8	Timer0 interrupt
T1	Internal/External	$7FF3,$7FF2	9	Timer1 interrupt
PT	External	$7FF1,$7FF0	10	Port-A transition interrupt
BT	Internal	$7FEF,$7FEE	11	Base Timer interrupt
LCD	Internal	$7FED,$7FEC	12	LCD Frame interrupt
-	-	$7FEB,$7FEA	-	Reserved
STX	External	$7FE9,$7FE8	2	SPI transmit buffer empty interrupt
SRX	External	$7FE7,$7FE6	3	SPI receive buffer ready interrupt
UTX	External	$7FE5,$7FE4	4	UART receiver interrupt
URX	External	$7FE3,$7FE2	5	UART transmitter interrupt

TABLE 8-2 Interrupt Request Register (IREQ)

Address	Name	R/W	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8	Default
$03C	**IREQL**	R/W	-	IRLCD	IRBT	IRPT	IRT1	IRT0	IRDAC	IRX	-000 0000
$03D	**IREQH**	R/W	-	-	-	-	IRURX	IRUTX	IRSRX	IRSTX	- - - - 0000

Bit 0: **IRX:** INTX interrupt request bit 1 = INTX edge interrupt occurs **0** = INTX edge interrupt doesn't occur	Bit 1: **IRDAC:** DAC reload interrupt request bit 1 = DAC time out interrupt occurs **0** = DAC time out interrupt doesn't occur
Bit 2: **IRT0:** Timer0 interrupt request bit 1 = Timer0 overflow interrupt occurs **0** = Timer0 overflow interrupt doesn't occur	Bit 3: **IRT1:** Timer1 interrupt request bit 1 = Timer1 overflow interrupt occurs **0** = Timer1 overflow interrupt doesn't occur
Bit 4: **IRPT:** Port-A interrupt request bit 1 = Port-A transition interrupt occurs **0** = Port-A transition interrupt doesn't occur	Bit 5: **IRBT:** Base Timer interrupt request bit 1 = Time base interrupt occurs **0** = Time base interrupt doesn't occur
Bit 6: **IRLCD:** LCD frame Interrupt request bit 1 = LCD Frame interrupt occurs **0** = LCD Frame interrupt doesn't occur	Bit 8: **IRSTX:** SPI transmitter interrupt request bit 1 = SPI transmit buffer is empty **0** = SPI transmit buffer is occupied
Bit 9: **IRSRX:** SPI receiver interrupt request bit 1 = SPI receive buffer is ready **0** = SPI receive buffer is not ready	Bit 10: **IRUTX:** UART transmitter interrupt request bit 1 = UART data transmission completes **0** = UART data transmission not completes
Bit 11: **IRURX:** UART receiver interrupt request bit 1 = UART data receiving completes **0** = UART data receiving not completes	

TABLE 8-3 Interrupt Enable Register (IENA)

Address	Name	R/W	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8	Default
$03E	**IENAL**	R/W	-	IELCD	IEBT	IEPT	IET1	IET0	IEDAC	IEX	- 000 0000
$03F	**IENAH**	R/W	-	-	-	-	IEURX	IEUTX	IESRX	IESTX	- - - - 0000

Bitx: 1 = Enable respective interrupt
0 = Disable respective interrupt

8.1 Interrupt Description

■ Brk

Instruction 'BRK' will cause software interrupt when interrupt disable flag (**I**) is cleared. Hardware will push '**PC**', '**P** ' registers to stack and then sets interrupt disable flag (**I**). Program counter will be loaded with the BRK vector from locations $7FFE and $7FFF.

■ Reset

A positive transition of $\overline{\text{RESET}}$ pin will make an initialization sequence to begin. After the system has been operating, one low level signal on this line of at least two clock cycles will cease ST2202 activity. When a positive edge is detected, there is an initialization sequence lasting six clock cycles. Then the interrupt disable flag is set, the decimal mode is cleared and the program counter will be loaded with the reset vector from locations $7FFC (low byte) and $7FFD (high byte). This is the start location for program flow. This input should be high in normal operation.

■ INTX Interrupt

The **IRX** (INTX interrupt request) flag will be set while INTX edge signal occurs. The INTX interrupt will be active when IEX (INTX interrupt enable) is set, and interrupt disable flag is cleared. Hardware will push 'PC', 'P ' registers to stack and sets interrupt disable flag (**I**). Program counter will be loaded with the INTX vector from locations $7FF8 and $7FF9.

■ DAC Interrupt

The IRDAC (DAC interrupt request) flag will be set while reload signal of DAC occurs. Then the DAC interrupt will be executed if IEDAC (DAC interrupt enable) is set, and interrupt disable flag is cleared. Hardware will push 'PC', 'P ' Register to stack and set interrupt mask flag (I). Program counter will be loaded with the DAC vector from locations $7FF6 and $7FF7.

■ T0 Interrupt

The IRT0 (TIMER0 interrupt request) flag will be set while Timer0 overflows. With IET0 (TIMER0 interrupt enable) being set, the T0 interrupt will execute, and interrupt mask flag will be cleared. Hardware will push 'PC', 'P ' Register to stack and set interrupt mask flag (I). Program counter will be loaded with the T0 vector from locations $7FF4 and $7FF5.

■ T1 Interrupt

The IRT1 (TIMER1 interrupt request) flag will be set while T1 overflows. With IET1 (TIMER1 interrupt enable) being set, the T1 interrupt will execute, and interrupt mask flag will be cleared. Hardware will push 'PC', 'P ' Register to stack and set interrupt mask flag (I). Program counter will be loaded with the T1 vector from locations $7FF2 and $7FF3.

■ PT Interrupt

The IRPT (Port-A interrupt request) flag will be set while Port-A transition signal occurs. With IEPT (PT interrupt enable) being set, the PT interrupt will be execute, and interrupt mask flag will be cleared. Hardware will push 'PC', 'P ' Register to stack and set interrupt mask flag (I). Program counter will be loaded with the PT vector from locations $7FF0 and $7FF1.

■ BT Interrupt

The IRBT (Base timer interrupt request) flag will be set when Base Timer overflows. The BT interrupt will be executed once the IEBT (BT interrupt enable) is set and the interrupt mask flag is cleared. Hardware will push 'PC', 'P ' Register to stack and set interrupt mask flag (I). Program counter will be loaded with the BT vector from locations $7FEE and $7FEF.

■ LCD Frame Interrupt

The **IRLCD** (LCD frame interrupt request) flag will be set when one new display frame cycle starts. This interrupt is very useful for software grayscale design. The LCD frame interrupt will be executed once the **IELCD** (LCD frame interrupt enable) is set and the interrupt mask flag is cleared. Hardware will push **PC** and **P** registers to stack and set interrupt disable flag "**I**". Program counter **PC** will be loaded with the LCD vector from locations $7FEC and $7FED.

■ SPI Interrupt

There are two interrupts for SPI transmitter and receiver respectively. **IRSTX** (SPI transmitter interrupt request) flag will be set when SPI transmit buffer is empty. **IRSRX** (SPI receiver interrupt request) flag will be set when SPI completes one receiving data and the receive buffer is ready. The SPI interrupts will be executed once the related enable flag **IESRX**, **IESTX** are set and the interrupt disable flag "**I**" is cleared. Hardware will push '**PC**', '**P** ' registers to stack and set "**I**" flag. Program counter will be loaded with the SPI vector from locations $7FE7, $7FE6, and $7FE9, $7FE8.

■ UART Interrupts

There are 2 interrupts for UART: receiver interrupt (URX), and transmitter interrupt (UTX). URX happens when receive-data is ready and the receiver needs to be serviced. UTX happens when current transmission is completed. Errors are indicated by bits of UART status register (**USTR**). Other sequences of UART interrupts are the same with those descriptions above.

9. GPIO

The ST2202 consists of 48 general-purpose I/O (GPIO) which are divided into six I/O ports: Port-A/B/C/D/E and Port-L. Control registers of GPIO are shown as following and in TABLE 9-1.

- Port data registers: **PA~PE, PL**
- Port direction control registers: **PCA~PCE, PCL**
- Port type select registers: **PSC**
- Port function select registers: **PFC** and **PFD**
- Port miscellaneous control register: **PMCR**

TABLE 9-1 Summary Of Control Registers Of GPIO

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$000	**PA**	R/W	PA[7]	PA[6]	PA[5]	PA[4]	PA[3]	PA[2]	PA[1]	PA[0]	1111 1111
$001	**PB**	R/W	PB[7]	PB[6]	PB[5]	PB[4]	PB[3]	PB[2]	PB[1]	PB[0]	1111 1111
$002	**PC**	R/W	PC[7]	PC[6]	PC[5]	PC[4]	PC[3]	PC[2]	PC[1]	PC[0]	1111 1111
$003	**PD**	R/W	PD[7]	PD[6]	PD[5]	PD[4]	PD[3]	PD[2]	PD[1]	PD[0]	1111 1111
$004	**PE**	R/W	PE[7]	PE[6]	PE[5]	PE[4]	PE[3]	PE[2]	PE[1]	PE[0]	1111 1111
$04C	**PL**	R/W	PL[7]	PL[6]	PL[5]	PL[4]	PL[3]	PL[2]	PL[1]	PL[0]	1111 1111
$005	**PSC**	R/W	PSC[7]	PSC[6]	PSC[5]	PSC[4]	PSC[3]	PSC[2]	PSC[1]	PSC[0]	1111 1111
$008	**PCA**	R/W	PCA[7]	PCA[6]	PCA[5]	PCA[4]	PCA[3]	PCA[2]	PCA[1]	PCA[0]	0000 0000
$009	**PCB**	R/W	PCB[7]	PCB[6]	PCB[5]	PCB[4]	PCB[3]	PCB[2]	PCB[1]	PCB[0]	0000 0000
$00A	**PCC**	R/W	PCC[7]	PCC[6]	PCC[5]	PCC[4]	PCC[3]	PCC[2]	PCC[1]	PCC[0]	0000 0000
$00B	**PCD**	R/W	PCD[7]	PCD[6]	PCD[5]	PCD[4]	PCD[3]	PCD[2]	PCD[1]	PCD[0]	0000 0000
$00C	**PCE**	R/W	PCE[7]	PCE[6]	PCE[5]	PCE[4]	PCE[3]	PCE[2]	PCE[1]	PCE[0]	0000 0000
$04E	**PCL**	W	PCL[7]	PCL[6]	PCL[5]	PCL[4]	PCL[3]	PCL[2]	PCL[1]	PCL[0]	0000 0000
$00D	**PFC**	R/W	RXD0	TXD0	SRDY	SS	MOSI	MISO	SCK	INTX	0000 0000
$00E	**PFD**	R/W	RXD1	TXD1	CS6	CS5	CS4	CS3	CS2	CS1	0000 0000
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	BCO	TCO1	TCO0	1000 0000

Each single pin can be programmed to be input or output. This is controlled by port direction control registers **PCx**. Setting bit of **PCx** makes respective pin to output, and clearing this bit for input. There are two options: pull-up/down for inputs of Port-C but only pull-up for inputs of the other ports. In case of output, there are open-drain/CMOS options for outputs of PortC but only CMOS for the other ports. Refer to TABLE 9-2.

TABLE 9-2 I/O Types Of GPIO Ports

I/O Mode	I/O Types	
	Port-A/B/D/E/L	Port-C
Input	Pull-up/Pure	Pull-up/Pull-down/Pure
Output	CMOS	Open-drain/CMOS

■ Input Mode

In case of input function, port data registers **Px** reflect the values on associated pins. Besides read instruction for data of signals input, writing to register **Px** selects I/O types of pins, pull-up or pull-down. Setting bits of all port data register **Px** to select pull-up type. Clearing bits of only **PC** to select pull-down type for pins of Port-C. There are no pull-down resistors for Port-A/B/D/E and Port-L, thereby no pull-down resistors will be enabled if clearing bits of **PA**, **PB**, **PD**, **PE** and **PL**. Pull-up resistors of Port-A/B/D/E/L are also controlled by PULL bit (bit7 of port miscellaneous register **PMCR**), "0" is to disable, while "1" is to enable them. The pull-up/pull-down resistors of Port-C are further controlled by bits of port type select registers **PSC**. They work in the same way with PULL bit of **PMCR** but only on single pin, "0" is to disable, while "1" is to enable. Refer to FIGURE 9-1.

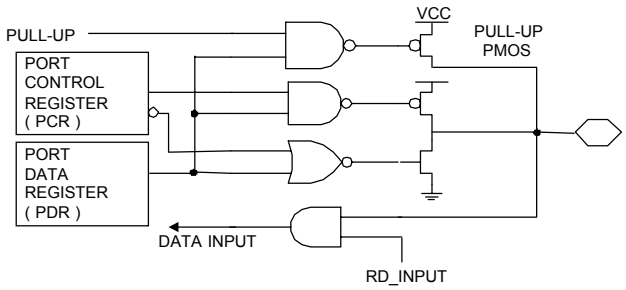


FIGURE 9-1 Configuration Of Inputs

■ Output Mode

In case of output function, Write to port data registers **Px** makes pins to output desired value. This value can also be read back by read instruction. Besides Port-C, the output pins are CMOS type. Port-C have two options of output types: open-drain and CMOS, and is controlled by port type select registers **PSC**. Clearing bits of registers **PSC** is for that disable PMOS of output stage and left only NMOS, while setting bits is for CMOS. Refer to FIGURE 9-2.

FIGURE 9-2 Configuration Of Outputs

Port-A is designed for keyboard scan with de-bounce and transition triggered interrupt, while Port-C/D/E are multiplexed with other system functions, and are controlled by **PFC**, **PFD**, and **PMCR[2:0]**. Port-L is shared with LCD specific signals of LCDC. Turning off LCDC by setting **LPWR** (**LCTR[7]**) reserves Port-L for GPIO.

Selecting respective pins to be GPIO or signals of system function will not affect original settings of I/O directions and types. This extends the flexibility of the usage of function signals.

Note: All the properties of pins are still programmable and must be ascertained before they are assigned to system functions, especially the direction of pins.

TABLE 9-3 Port Control Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$008~$00C / $07E	**PCA~PCE, PCL**	R/W	PCx[7]	PCx[6]	PCx[5]	PCx[4]	PCx[3]	PCx[2]	PCx[1]	PCx[0]	0000 0000

Bit 7~0: **PCx[7:0] :** Port-x direction control bits
0 = Input mode
1 = Output mode

TABLE 9-4 Port Data Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$000~$004 / $07C	**PA~PE, PL**	R/W	Px[7]	Px[6]	Px[5]	Px[4]	Px[3]	Px[2]	Px[1]	Px[0]	1111 1111

Bit 7~0: **Px[7:0] :** Port data / pull-resistor control bits

R/W	I/O Modes	
	Input Mode	Output Mode
Read	Input data	Output data
Write	0 = Disable pull-up resistor Select pull-down resistor (Port-C only) **1** = Select pull-up resistor	

TABLE 9-5 Port I/O Type Select Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$005	**PSC**	R/W	PSC[7]	PSC[6]	PSC[5]	PSC[4]	PSC[3]	PSC[2]	PSC[1]	PSC[0]	1111 1111

Bit 7~0: **PSC[7:0] :** Port I/O types selection bits

Input Mode	Output Mode
0 = Disable pull-up/down resisters	0 = Open-drain
1 = Enable pull-up/down resisters	**1** = CMOS

TABLE 9-6 Port Function Select Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00D	**PFC**	R/W	RXD0	TXD0	SRDY	SS	MOSI	MISO	SCK	INTX	0000 0000
$00E	**PFD**	R/W	RXD1	TXD1	CS6	CS5	CS4	CS3	CS2	CS1	0000 0000

Bit 7~0: **PFC/D[7:0] :** Port function select bits
0 = GPIO
1 = Indicated function signal is connected

TABLE 9-7 Port Miscellaneous Control Register (PMCR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	BCO	TCO1	TCO0	1000 0000

Bit 7: **PULL :** Enable/disable all pull-up resisters of Port-A/B/D/E/L
1 = Enable pull-up resisters
0 = Disable pull-up resisters

9.1 Port-A Transistion Interrupt

Port-A is designed for the return line inputs of keyboard scan with transition triggered interrupt and de-bounce option. Difference between current value and the data kept previously of Port-A will generate an interrupt request. The last state of Port-A must be latched before transition, and this can be done by one read instruction to Port-A. Steps and program example are shown below.

Operate Port-A interrupt steps:

1. Set input mode.
2. Read Port-A.
3. Clear interrupt request flag (IRPT).
4. Set interrupt enable flag (IEPT).
5. Clear CPU interrupt disable flag (I).
6. Read Port-A before 'RTI' instruction in ISR

Example:

```
      .
      .
   STZ    <PCA        ; Set input mode.
   LDA    #$FF
   STA    <PA         ; PA be PULL-UP.
   LDA    <PA         ; Keep last state.
   RMB4   <IREQ       ; Clear IRQ flag.
   SMB4   <IENA       ; Enable INT.
   CLI
      .
      .
```

Interrupt subroutine

```
      .
      .
   LDA    <PA         ; Keep last state.
   RTI
```

9.1.1 Port-A Interrupt De-bounce

The ST2202 has a hardware de-bounce block for Port-A interrupt. It is enabled with “1” and disable with “0” of **PDBN (PMCR[6])**. The de-bounce function is activated after first Port-A transition is detected. It uses OSCX as the sampling clock. The de-bounce time is OSCX x 512 cycles (about 15.6 ms). Data filtered by de-bounce presents a stable state, then the interrupt can be issued.

TABLE 9-8 Port Miscellaneous Control Register (PMCR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	TO2	TO1	TO0	1000 0000

Bit 6: **PDBN :** Enable Port-A interrupt de-bounce
1 = De-bounce for Port-A interrupt
0 = No de-bounce for Port-A interrupt

9.2 External Interrupt

PC0 plays another function of external edge-sensitive interrupt source. Falling or rising edge is controlled by INTEG(PMCR[5]).

Steps for INTX interrupt operation:

1. Set PC0 to input mode. (PCC[0])
2. Set PF0 to “1”
3. Select edge level. (INTEG)
4. Clear INTX interrupt request flag. (IRX)
5. Set INTX interrupt enable bits. (IEX)
6. Clear CPU interrupt mask flag (I).

Steps and program example are shown below.

Example:

```
.
.
RMB0    <PCC        ; Set input mode.
SMB0    <PFC        ; Enable INTX function
SMB5    <PMCR       ; Rising edge.
RMB0    <IREQ       ; Clear IRQ flag.
SMB0    <IENA       ; Enable INTX interrupt.
CLI
.
```

TABLE 9-9 Port Miscellaneous Control Register (PMCR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	TO2	TO1	TO0	1000 0000

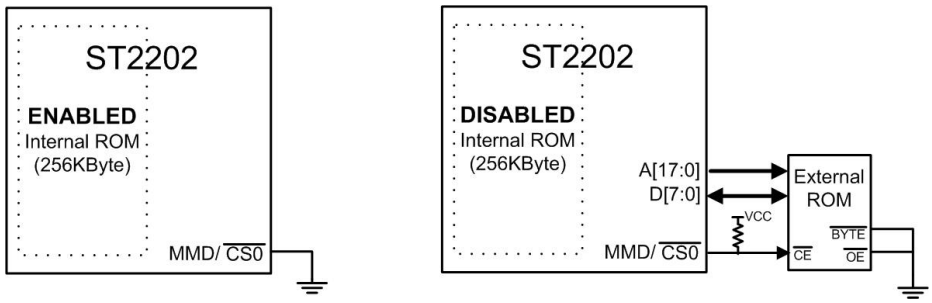
Bit 5: **INTEG :** Edge options of external interrupt
1 = External interrupt is rising edge triggered
0 = External interrupt is falling edge triggered

10. CHIP-SELECT LOGIC (CSL)

The ST2202 builds in one chip-select signal ($\overline{CS0}$) for embedded 256K bytes mask ROM and six chip-select signals multiplexed with PD5~0 of Port-D which are used to select external devices on the address and data bus. There are two options for the first 256K bytes memory which are controlled by MMD pin. Tie MMD to ground to select normal mode and enable internal ROM for the first 256K bytes memory. Connect MMD to chip-select of an external device to select emulation mode and disable internal ROM. After reset cycles, MMD changes to an output and outputs chip-select signal $\overline{CS0}$. Refer to FIGURE 10-1 for two connections of different modes.

Two bits **CSM[1:0]** of port miscellaneous register (**PMCR**) select four modes of CSL which define the memory size of each external chip-select. If **CSM0** equals "1", chip-select signal $\overline{CS6}$ changes to be address signal A23 to make one single device of 16M bytes at $\overline{CS5}$ possible. The address range of $\overline{CSx}$ of higher number follows the range of previous one of lower number. Refer to TABLE 10-2 for configurations of all chip-selects in different modes.

Note: Write "1" to bit of port direction control register **PCD**, then to bit of port function-select register **PFD** to activate the designated chip-select signal.



A. Normal Mode **B. Emulation Mode**

FIGURE 10-1 Connections Of MMD/ $\overline{CS0}$

TABLE 10-2 Memory Configurations Of Chip-selects

First 256K	External Chip-select Modes	Memory Range and Size of Chip-selects							Total Support Memory Size
$\overline{CS0}$, MMD/ $\overline{CS0}$	**CSM[1:0]**	$\overline{CS1}$	$\overline{CS2}$	$\overline{CS3}$	$\overline{CS4}$	$\overline{CS5}$	$\overline{CS6}$ /A23		
$0000000~ $003FFFF (**256K**byte)	0 0	$0400000~ $04FFFFF (**1M** bytes)	$0500000~ $05FFFFF (**1M** bytes)	$0600000~ $07FFFFF (**2M** bytes)	$0800000~ $0FFFFFF (**8M** bytes)	$1000000~ $17FFFFF (**8M** bytes)	$1800000~ $1FFFFFF (**8M** bytes)		**28M + 256K** Bytes
	0 1					$1000000~ $1FFFFFF (**16M**bytes)	A23		
	1 0	$0400000~ $07FFFFF (**4M** bytes)	$0800000~ $0FFFFFF (**8M** bytes)	$1000000~ $17FFFFF (**8M** bytes)	$1800000~ $1FFFFFF (**8M** bytes)	$2000000~ $27FFFFF (**8M** bytes)	$2800000~ $2FFFFFF (**8M** bytes)		**44M + 256K** Bytes
	1 1					$2000000~ $2FFFFFF (**16M**bytes)	A23		

TABLE 10-3 Port Function Select Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00E	**PFD**	R/W	RX1	TX1	CS6	CS5	CS4	CS3	CS2	CS1	0000 0000

Bit 7~0: **PFD[5:0] :** Port function select bits
0 = GPIO
1 = Chip-select signal is connected

TABLE 10-4 Port Miscellaneous Control Register (PMCR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	TO2	TO1	TO0	1000 0000

Bit 1~0: **CSM[1:0] :** External chip-select mode selection bits
See TABLE 10-2 for more information

11. CLOCK GENERATOR

The ST2202 has two oscillators OSC and OSCX for both high and low frequency needed. When oscillator mode selection pin, XMD, is inputted high level, the high frequency oscillator OSC adopts only one external resistor to generate a high frequency clock OSCK which is used by almost every block in chip. OSC can also change to be a resonator/crystal oscillator by input low level to XMD.

The low frequency oscillator OSCX needs a 32768Hz crystal and one capacitor to generator a precise frequency CLK32 for Base timer, Timer1 and the reference clock of baud rate generator (BGR).

Other clocks are sourced from either OSCK or CLK32 and are listed below:

- System clock: **SYSCK**
- LCD controller clock: **LCDCK**
- PSG and PWM DAC clock: **PSGCK**
- BGR output clock: **BGRCK**
- SPI transmission clock: **SPICK**

■ SYSCK

The system clock can be switched between OSCK and CLK32 by resetting or setting **XSEL** (**SYS[7]**). After **XSEL** is set (or reset), warm-up cycles will be initiated at the same time. The original clock is still connected until the end of warm-up cycles. Clock being used can be reported by reading **XSEL** back.

Note: Test **XSEL** to confirm SYSCK is switched over successfully before turning down the original clock.

There are two options for warm-up cycles: 16 / 256 cycles, which are controlled by **WSKP** (**SYS[3]**). Usually 16 cycles are enough for OSC and OSCX.

■ LCDCK

The LCD controller has one four-bit divider to generate LCDCK directly from OSCK for pixel clock and other operations. This divider is controlled by **LCKR[3:0]** and the data mode selection bit **LMOD**(**LCKR[4]**). Refer to TABLE 11-3 for settings of LCDCK.

■ PSGCK

PSGCK is the clock used by PSG and PWM DAC. It is sourced from OSCK to make sure of one right and high enough base frequency and to keep it unchanged. Bits of **PSGC[6:4]** control the options of PSGCK. Refer to TABLE 11-4 for these options.

■ BGRCK

The ST2202 equips a baud rate generator (BGR), which is controlled by BGR control register **BCTR**, locked frequency selection register **BRS**, and divider control register **BDIV**. The BGR utilizes digital PLL technique to lock a high frequency F_{HIGH} around OSCK/2. This high frequency is further scaled down via an integer divider to a desired frequency BGRCK. The BGR uses CLK32 as reference clock for the modulation of OSCK. There are two modulation modes which can be selected by **BMOD** (**BCTR[1]**). The modulation strength is also controllable by setting or resetting **BSTR** (**BCTR[2]**).

The relation between locked frequency and **BRS** can be found in the following equation.

$$F_{HIGH} = CLK32 \cdot \mathbf{BRS} \qquad \text{Equation9-1}$$

OSCK and F_{HIGH} are close related. Value of F_{HIGH} limits the frequency range of the OSCK applied, which is also the locking range of BGR, and is given by the following equation, where α is the modulation strength coefficient.

$$F_{HIGH} \cdot \frac{\alpha}{\alpha+1} \le \frac{OSCK}{2} \le F_{HIGH} \cdot \frac{\alpha}{\alpha-1} \qquad \text{Equation9-2}$$

Although the locked frequency is limited to be around OSCK, lower frequency can still be obtained by one 8-bit integer divider, which is assigned by **BDIV**. Thus BGRCK can be expressed by Equation9-3.

$$BGRCK = \frac{F_{HIGH}}{\mathbf{BDIV}} \qquad \text{Equation9-3}$$

■ SPICK

The SPI block has one three-bit divider to generate SPICK directly from OSCK for transmission and other operations. This divider is controlled by **SCKR[6:4]**. Refer to TABLE 11-7 for settings of SPICK.

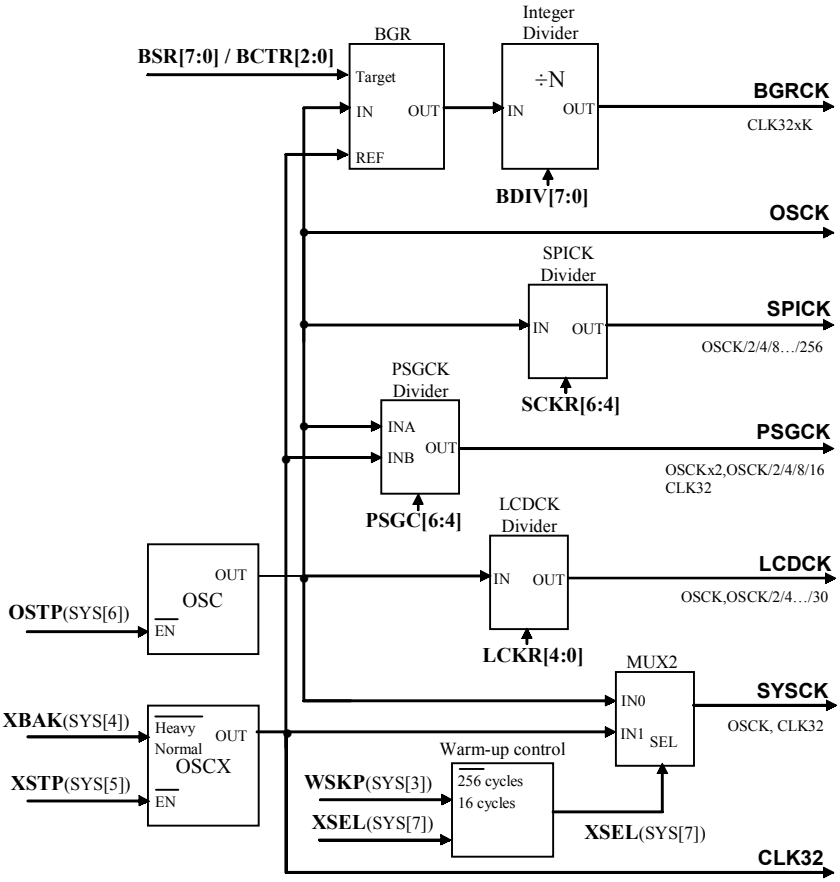


FIGURE 11-1 Clock Generator Diagram

TABLE 11-2 System Control Register (SYS)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$030	SYS	R	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	HIGH	0000 0001
		W	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	LVDEN	0000 0000

Bit 7: **XSEL :** Write: Select source of system clock (SYSCK) / Read: report of clock source being used
0 = OSC
1 = OSCX

Bit 6: **OSTP :** OSC stop control bit
0 = Enable OSC
1 = Disable OSC

Bit 5: **XSTP :** OSCX stop control bit
0 = Enable OSCX
1 = Disable OSCX

Bit 4: **XBAK :** OSCX driver heavy load bit
0 = OSCX heavy load
1 = OSCX normal load

Bit 3: **WSKP :** System warm-up cycles selection bit
0 = 256 warm-up cycles
1 = 16 warm-up cycles

TABLE 11-3 LCD Clock Control Register (LCKR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$048	**LCKR**	W	-	-	-	LMOD	LCK[3]	LCK[2]	LCK[1]	LCK[0]	- - -0 0000

Bit 4: **LMOD** : LCD data bus mode selection
0 = 1-bit mode
1 = 4-bit mode

Bit 3~0: **LCKR[3:0]** : LCD clock selection

LCKR[3:0]	LCDCK	
	1-bit mode (**LMOD=0**)	4-bit mode (**LMOD**=1)
0000	OSCK	OSCK
0001		OSCK/2
0010		/4
0011		/6
0100	OSCK/2	/8
0101		/10
0110		/12
0111		/14
1000	OSCK/4	/16
1001		/18
1010		/20
1011		/22
1100	OSCK/6	/24
1101		/26
1110		/28
1111		/30

TABLE 11-4 PSG Control Register (PSGC)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$016	**PSGC**	R/W	-	PCK[2]	PCK[1]	PCK[0]	PRBS	C1EN	C0EN	DACE=0	-000 0000
			-	PCK[2]	PCK[1]	PCK[0]	DMD[1]	DMD[0]	INH	DACE=1	-000 0000

Bit 3~0: **PSGC[6:4]** : PSG clock selection

PCK[2:0]	PSGCK
000	SYSCK/2
001	SYSCK/4
010	SYSCK/8
011	SYSCK/16
1xx	SYSCK*2
111	CLK32

TABLE 11-5 BGR Control Register (BCTR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$063	**BCTR**	R/W	TEST	-	-	-	-	BSTR	BMOD	BGREN	0- - - -000

Bit 7: **TEST** : Test bit, must be “0”

Bit 2: **BSTR** : Modulation strength selection bit
0 = Full modulation strength
1 = Half modulation strength

Bit 1: **BMOD** : Modulation mode selection bit
0 = Coarse modulation mode
1 = Fine modulation mode

Bit 0: **BGREN** : BGR enable/disable bit
0 = Disable BGR
1 = Enable BGR

TABLE 11-6 BGR Configuration Registers (BRS/BDIV)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$066	**BRS**	R/W	BRS[7]	BRS[6]	BRS[5]	BRS[4]	BRS[3]	BRS[2]	BRS[1]	BRS[0]	???? ????
$067	**BDIV**	R/W	BDIV[7]	BDIV[6]	BDIV[5]	BDIV[4]	BDIV[3]	BDIV[2]	BDIV[1]	BDIV[0]	???? ????

BGR output frequency settings. See Equation9-1 ~ 9-3

TABLE 11-7 SPI Clock Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$053	**SCKR**	R/W	-	SCK[2]	SCK[1]	SCK[0]	BC[3]	BC[2]	BC[1]	BC[0]	-000 0000

Bit 6~4: **SCK[2:0]** : SPI clock selection

SCK[2:0]	SPICK
000	SYSCK/2
001	SYSCK/4
010	SYSCK/8
011	SYSCK/16
100	SYSCK/32
101	SYSCK/64
110	SYSCK/128
111	SYSCK/256

12. TIMER/EVENT COUNTER

12.1 Prescaler

12.1.1 Function Description

The ST2202 has three timers, Base timer, Timer 0 and Timer 1, and two prescalers PRES and PREW. There are two clock sources, SYSCK and INTX, for PRES and one clock source, CLK32, for PREW. Refer to FIGURE 12-1

TABLE 12-1 Summary of Timer Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$020	**BTEN**	R/W	-	-	-	BTEN[4]	BTEN[3]	BTEN[2]	BTEN[1]	BTEN[0]	- - -0 0000
$021	**BTR**	R/W	-	-	-	BTR[4]	BTR[3]	BTR[2]	BTR[1]	BTR[0]	- - -0 0000
$023	**PRS**	R	PRS[7]	PRS[6]	PRS[5]	PRS[4]	PRS[3]	PRS[2]	PRS[1]	PRS[0]	0000 0000
		W	SRES	SENA	SENT	-	-	-	-	-	000 - - - - -
$024	**T0M**	R/W	-	-	T0M[5]	T0M[4]	-	T0M[2]	T0M[1]	T0M[0]	- -00 -000
$025	**T0C**	R/W	T0C[7]	T0C[6]	T0C[5]	T0C[4]	T0C[3]	T0C[2]	T0C[1]	T0C[0]	0000 0000
$026	**T1M**	R/W	-	-	-	T1M[4]	T1M[3]	T1M[2]	T1M[1]	T1M[0]	- - -0 0000
$027	**T1C**	R/W	T1C[7]	T1C[6]	T1C[5]	T1C[4]	T1C[3]	T1C[2]	T1C[1]	T1C[0]	0000 0000
$030	**SYS**	R	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	HIGH	0000 0001
		W	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	LVDEN	0000 0000
$03C	**IREQ**	R/W	-	-	IRBT	IRPT	IRT1	IRT0	IRDAC	IRX	- -00 0000
$03E	**IENA**	R/W	-	-	IEBT	IEPT	IET1	IET0	IEDAC	IEX	- -00 0000

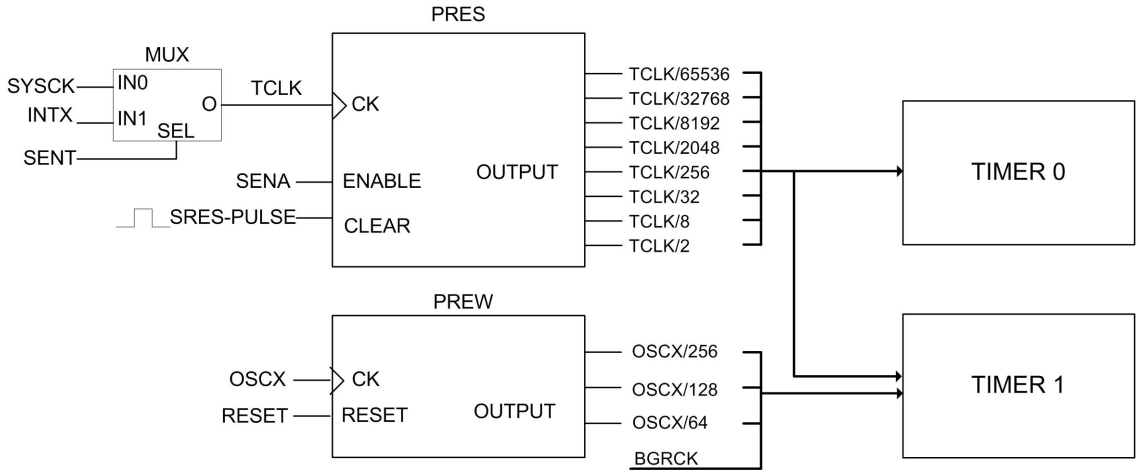


FIGURE 12-1 Structure Of Two Prescalers

12.1.2 PRES

The prescaler PRES is an 8-bits counter as shown in FIGURE 12-1. Which provides four clock sources for base timer and timer1, and it is controlled by register PRS. The instruction read toward PRS will bring out the content of PRES and the Instruction write toward PRS will reset, enable or select clock sources for PRES.

When user set external interrupt as the input of PRES for event counter, combining PRES and Timer1 will get a 16bit-event counter.

TABLE 12-2 Prescaler Control Register (PRS)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$023	PRS	R	PRS[7]	PRS[6]	PRS[5]	PRS[4]	PRS[3]	PRS[2]	PRS[1]	PRS[0]	0000 0000
		W	SRES	SENA	SENT	-	-	-	-	-	000 - - - - -

READ

Bit 7~0: **PRS[7~0] :** Value of PRES counter

WRITE

Bit 7: **SRES :** Prescaler Reset bit
Write “1” to reset the prescaler (PRS[7~0])

Bit 6: **SENA :** Prescaler enable bit
0 = Disable prescaler counting
1 = Enable prescaler counting

Bit 5: **SENT :** Clock source(TCLK) selection for prescaller PRES
0 = Clock source from system clock “SYSCK”
1 = Clock source from external events “INTX”

12.1.3 PREW

The prescaler PREW is an 8-bits counter as shown in Figure 11-6. PREW provides four clocks source for base timer and timer1. It stops counting only if OSCX stops or hardware reset occurs.

12.2 Base Timer

The base timer supports one interrupt, which occurs at five different rates. Applications base on the base timer interrupt can chose an appropriate interrupt rate from five time bases for their specific needs. These real-time applications may include digitizer sampling, keyboard debouncing, or communication polling. Block diagram of base timer is shown in FIGURE 12-2.

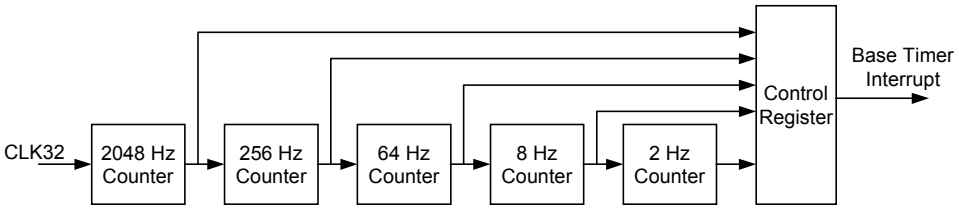


FIGURE 12-2 Base Timer Block Diagram

12.2.1 Base Timer Operations

The base timer consists of five sub-counters to produce five predefined rates. The connections between overflow signals of these sub-counters and the base timer interrupt are controlled by respective bit fields of base timer enable register (**BTEN**). The enabled overflow signals are ORed to generate the base timer interrupt request. Related bits of base timer status register (**BTSR**) will show which rates of interrupts should be serviced. Write "1" to **BTCLR** (bit 7 of **BTSR)** may clear this register.

Note: Make sure **BTSR** is cleared after the interrupt was serviced, so that the request can be set next time.

12.2.2 Base Timer Control/Status Registers

Summary of base timer control/status registers is shown in TABLE 12-3.

TABLE 12-3 Summary Of Base Timer Control Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$020	**BTEN**	R/W	-	-	-	BTEN[4]	BTEN[3]	BTEN[2]	BTEN[1]	BTEN[0]	- - -0 0000
$021	**BTSR**	R	-	-	-	BTSR[4]	BTSR[3]	BTSR[2]	BTSR[1]	BTSR[0]	- - -0 0000
		W	BTCLR	-	-	-	-	-	-	-	0- - - - - - -
$03C	**IREQ**	R/W	-	IRLCD	IRBT	IRPT	IRT1	IRT0	IRDAC	IRX	- 000 0000
$03E	**IENA**	R/W	-	IELCD	IEBT	IEPT	IET1	IET0	IEDAC	IEX	- 000 0000

■ **Base Timer Control Register**

TABLE 12-4 Base Timer Control Register (BTEN)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$020	**BTEN**	R/W	-	-	-	BTEN[4]	BTEN[3]	BTEN[2]	BTEN[1]	BTEN[0]	- - -0 0000

Bit 0: **BTEN0 :** 2 Hz interrupt control bit
0 = Disable 2 Hz interrupt
1 = Enable 2 Hz interrupt

Bit 1: **BTEN1 :** 8 Hz interrupt control bit
0 = Disable 8 Hz interrupt
1 = Enable 8 Hz interrupt

Bit 2: **BTEN2 :** 64 Hz interrupt control bit
0 = Disable 64 Hz interrupt
1 = Enable 64 Hz interrupt

Bit 3: **BTEN3 :** 256 Hz interrupt control bit
0 = Disable 256 Hz interrupt
1 = Enable 256 Hz interrupt

Bit 4: **BTEN4 :** 2048 Hz interrupt control bit
0 = Disable 2048 Hz interrupt
1 = Enable 2048 Hz interrupt

■ Base Timer Status Register

TABLE 12-5 Base Timer Status Register (BTSR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$021	BTSR	R	-	-	-	BTSR[4]	BTSR[3]	BTSR[2]	BTSR[1]	BTSR[0]	0- -0 0000
		W	BTCLR	-	-	-	-	-	-	-	0- - - - - - -

Bit 0: **BTSR0 :** 2 Hz interrupt status bit
0 = No 2 Hz interrupt occurred
1 = 2 Hz interrupt occurred

Bit 1: **BTSR1 :** 8 Hz interrupt status bit
0 = No 8 Hz interrupt occurred
1 = 8 Hz interrupt occurred

Bit 2: **BTSR2 :** 64 Hz interrupt status bit
0 = No 64 Hz interrupt occurred
1 = 64 Hz interrupt occurred

Bit 3: **BTSR3 :** 256 Hz interrupt status bit
0 = No 256 Hz interrupt occurred
1 = 256 Hz interrupt occurred

Bit 4: **BTSR4 :** 2048 Hz interrupt status bit
0 = No 2048 Hz interrupt occurred
1 = 2048 Hz interrupt occurred

Bit 7: **BTCLR :** Write “1” to clear all status bit

12.3 Timer 0

12.3.1 Function Description

The Timer0 is an 8-bit up counter. It can be used as a timer or an event counter. T0C($25) is a real time read/write counter. When an overflow from $FF to $00, a timer interrupt request IRT0 will be generated. Timer0 will stop counting when system clock stops. Please refer to FIGURE 12-3.

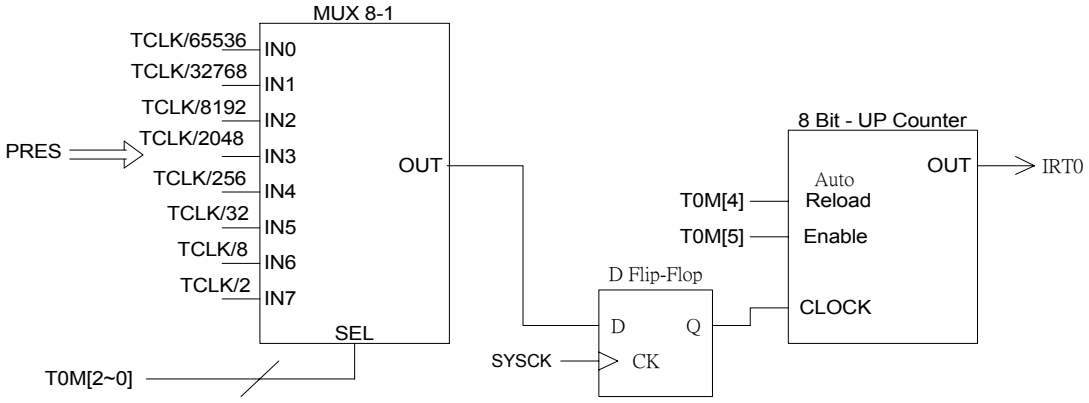


FIGURE 12-3 Timer0 Structure

12.3.2 Timer0 Clock Source Control

Several clock sources can be chosen from for Timer0. It's very important that Timer0 can keep counting as long as SYSCK stays active. Refer to TABLE 12-6.

TABLE 12-6 Clock Sources Of Timer0

T0M[2]	T0M[1]	T0M[0]	T0 Timer Clock Source
0	0	0	TCLK/65536
0	0	1	TCLK/32768
0	1	0	TCLK/8192
0	1	1	TCLK/2048
1	0	0	TCLK/256
1	0	1	TCLK/32
1	1	0	TCLK/8
1	1	1	TCLK/2

T0M[4] : Control automatic reload operation
0 : No auto reload
1 : Auto reload
T0M[5] : Control Timer 0 enable/disable
0 : Disable counting
1 : Enable counting
SENA : Prescaler enable bit
0 : TCLK stop
1 : TCLK counting

TABLE 12-7 Timer0 Register (T0C)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$025	**T0C**	R/W	T0C[7]	T0C[6]	T0C[5]	T0C[4]	T0C[3]	T0C[2]	T0C[1]	T0C[0]	0000 0000
Bit 7-0: **T0C[7-0] :** Timer0 up counter register											

12.4 Timer 1

The Timer1 is an 8-bit up counter. It used as timer/counter as program specified. The difference between base timer is that Timer1 will halt during CPU SBY, but base timer will not. It is shown in FIGURE 12-4.

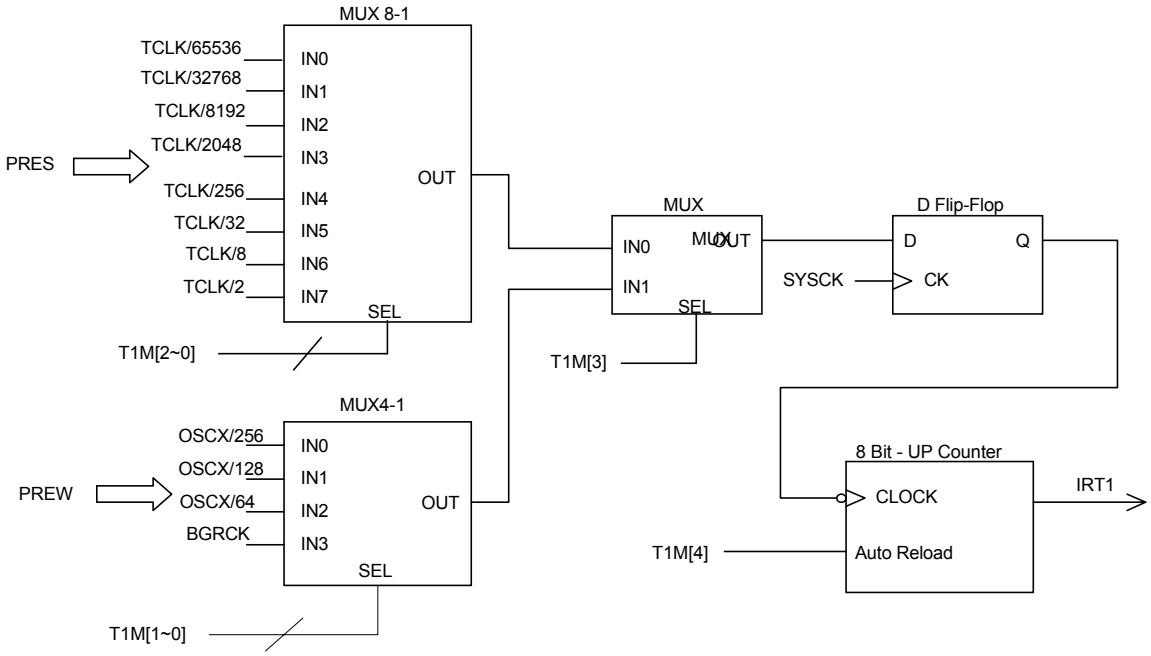


FIGURE 12-4 Timer1 Structure

TABLE 12-8 Timer1 Register (T1C)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$027	**T1C**	R/W	T1C[7]	T1C[6]	T1C[5]	T1C[4]	T1C[3]	T1C[2]	T1C[1]	T1C[0]	0000 0000
Bit 7-0: **T1C[7-0] :** Timer1 up counter register											

TABLE 12-9 Clock Sources Of Timer1

T1M[3]	T1M[2]	T1M[1]	T1M[0]	T1 Timer Clock Source
0	0	0	0	TCLK/65536
0	0	0	1	TCLK/32768
0	0	1	0	TCLK/8192
0	0	1	1	TCLK/2048
0	1	0	0	TCLK/256
0	1	0	1	TCLK/32
0	1	1	0	TCLK/8
0	1	1	1	TCLK/2
1	0	0	0	OSCX/256
1	0	0	1	OSCX/128
1	0	1	0	OSCX/64
1	0	1	1	BGRCK

T1M[4]: Control automatic reload operation
0: No auto reload
1: auto reload
SENA : Prescaler enable bit
0 : TCLK stop
1 : TCLK counting

13. CLOCKING OUTPUTS

Three clocking outputs PE0, PE1 and PE2 are supported by the ST2202. These signals are very useful for outputs of high frequency, such as PWM base signal or carrier of remote control. Timer0, Timer1 overflow signals are clock sources for PE0 and PE1, while BGRCK are for PE2.

■ Clocking Outputs: PE0 and PE1

Overflow states of Timers will be connected to toggle data of **PE[0:1]** when setting function selection bits **TCO0/TCO1** (**PMCR[0:1]**). Meanwhile PE0/PE1 output clocked data of half the frequency of Timers. After resetting **TCO0/TCO1**, the toggle operation ceases. Then PE0/PE1 return to the original logic level of **PE[0:1]**.

■ Clocking Output: PE2

BGRCK will output through PE2 when setting function selection bit **BCO** (**PMCR[2]**). If **BCO** is cleared, PE2 returns to the original logic level of **PE[2]**.

Summary of clocking outputs registers is shown in TABLE 13-1. The clocking outputs enable bits can be found in TABLE 13-2.

TABLE 13-1 Summary Of Clocking Outputs Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$004	**PE**	R/W	PE[7]	PE[6]	PE[5]	PE[4]	PE[3]	PE[2]	PE[1]	PE[0]	1111 1111
$00C	**PCE**	R/W	PCE[7]	PCE[6]	PCE[5]	PCE[4]	PCE[3]	PCE[2]	PCE[1]	PCE[0]	0000 0000
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	BCO	TCO1	TCO0	1000 -000

TABLE 13-2 Port Miscellaneous Control Register (PMCR)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$00F	**PMCR**	R/W	PULL	PDBN	INTEG	CSM1	CSM0	BCO	TCO1	TCO0	1000 0000

Bit 0: **TCO0 :** Clocking output PE0 control bit (sourced from Timer0)
0 = Disable clocking output of PE0
1 = Enable clocking output of PE0

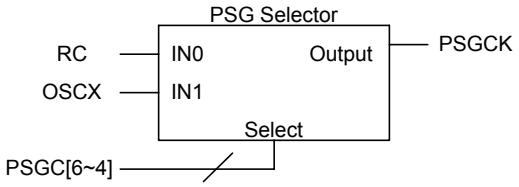
Bit 1: **TCO1 :** Clocking output PE1 control bit (sourced from Timer1)
0 = Disable clocking output of PE1
1 = Enable clocking output of PE1

Bit 2: **BCO :** Clock signal output PE2 control bit (sourced from BGRCK)
0 = Disable clock signal output of PE2
1 = Enable clock signal output of PE2

14. PSG

14.1 Function description

The built-in dual channel Programmable Sound Generator (PSG) is controlled by register file directly. Its flexibility makes it useful in applications such as music synthesis, sound effects generation, audible alarms and tone signaling. In order to generate sound effects while allowing the processor to perform other tasks, the PSG can continue to produce sound after the initial commands have been given by the CPU. The structure of PSG was shown in FIGURE 14-2 and the PSG clock source is shown in FIGURE 14-1. The ST2202 has three PSG playing type. One for channel0(C0) & channel1(C1) square type tone sound playing. Second for ch0 square tone sound and ch1 noise sound. The third sound playing type is DAC PCM playing.



PSGC			PSGCK
B6	B5	B4	
0	0	0	SYSCK
X	0	1	SYSCK/2
X	1	0	SYSCK/4
0	1	1	SYSCK/8
1	0	0	SYSCK x 2
1	1	1	OSCX

FIGURE 14-1 PSG Clock Source Control

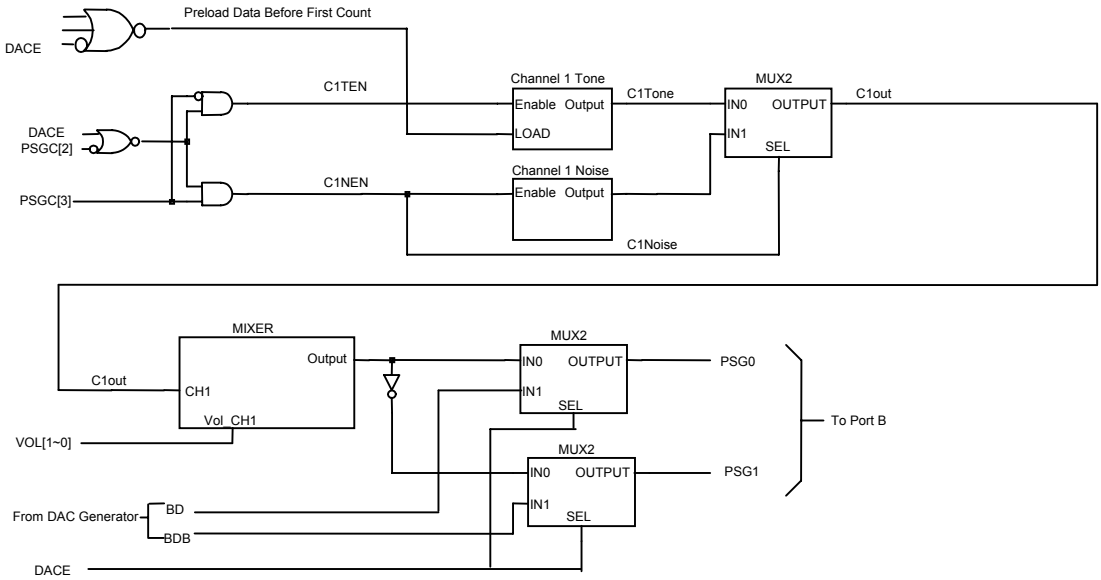


FIGURE 14-2 PSG Block Diagram

TABLE 14-1 Summary Of PSG Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$010	**PSG0L**	W	PSG0[7]	PSG0[6]	PSG0[5]	PSG0[4]	PSG0[3]	PSG0[2]	PSG0[1]	PSG0[0]	0000 0000
$011	**PSG0H**	W	-	-	-	-	PSG0[11]	PSG0[10]	PSG0[9]	PSG0[8]	- - - - 0000
$012	**PSG1L**	W	PSG1[7]	PSG1[6]	PSG1[5]	PSG1[4]	PSG1[3]	PSG1[2]	PSG1[1]	PSG1[0]	0000 0000
$013	**PSG1H**	W	-	-	-	-	PSG1[11]	PSG1[10]	PSG1[9]	PSG1[8]	- - - - 0000
$016	**PSGC**	W	-	PCK[2]	PCK[1]	PCK[0]	PRBS	C1EN	C0EN	DACE=0	- 000 0000
		W	-	PCK[2]	PCK[1]	PCK[0]	DMD[1]	DMD[0]	INH	DACE=1	- 000 0000
$017	**VOL**	W	VOL1[3]	VOL1[2]	VOL1[1]	VOL1[0]	VOL0[3]	VOL0[2]	VOL0[1]	VOL0[0]	0000 0000

TABLE 14-2 PSG Volume Control Register (VOL)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$017	**VOL**	W	VOL1[3]	VOL1[2]	VOL1[1]	VOL1[0]	VOL0[3]	VOL0[2]	VOL0[1]	VOL0[0]	0000 0000

Bit 3~0: **VOL0[3~0] :** PSG channel 0 volume control bit
0000 = No sound output
0001 = 1/16 volume (PSGCK must >= 320K Hz)
:
0100 = 4/16 volume
:
1000 = 8/16 volume
:
1111 = Maximum volume (PSGCK must >= 20K Hz)

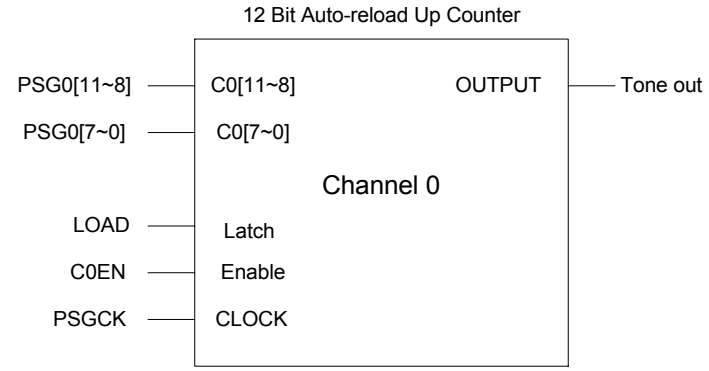
Bit 7~4: **VOL1[3~0] :** PSG channel 1 volume control bit
0000 = No sound output
0001 = 1/16 volume (PSGCK must >= 320K Hz)
:
0100 = 4/16 volume
:
1000 = 8/16 volume
:
1111 = Maximum volume (PSGCK must >= 20K Hz)

Note: If single channel is enable, then PSG volume control can be double. (16 + 16 = 32 level volume control)

14.2 Tone Generator

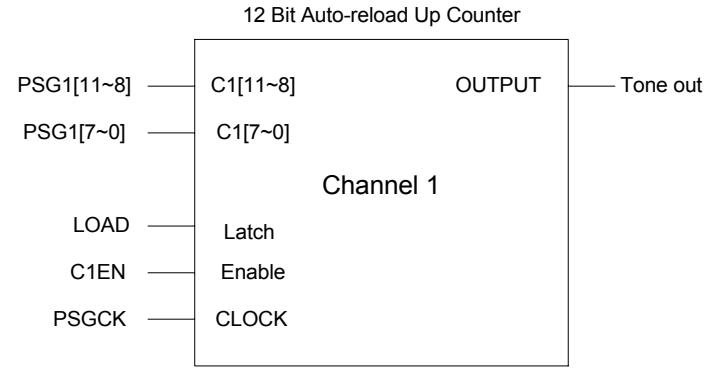
14.2.1 General Description

The tone frequency is decided by PSGCK and 12-bit programmable divider (PSG[11~0]). Please refer to FIGURE 14-3 and.FIGURE 14-4.



Frequency of Channel 0 Tone = PSGCK/(1000H-PSG0[11~0])/2

FIGURE 14-3 Tone Generator Channel 0



Frequency of Channel 1 Tone = PSGCK/(1000H-PSG1[11~0])/2

FIGURE 14-4 Tone Generator Channel 1

14.2.2 PSG Tone Programming

Setting PSG control bit **DACE** (**PSGC[0]**) will make PSG block functions as a sound generator of 2 channels. Setting **C1EN** will enable tone generator when PSG is in tone function. Noise or tone function is selected by **PRBS**.

TABLE 14-3 PSG Control Register (PSGC)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$016	PSGC	W	-	PCK[2]	PCK[1]	PCK[0]	PRBS	C1EN	C0EN	DACE=0	- 000 0000
		W	-	PCK[2]	PCK[1]	PCK[0]	DMD[1]	DMD[0]	INH	DACE=1	- 000 0000

Bit 0: **DACE :** Tone(Noise) or DAC Generator selection bit
1 = PSG is used as the DAC generator
0 = PSG is used as the Tone (Noise) generator

Bit 1: **C0EN :** PSG channel 0 (Tone) enable bit
1 = PSG0 (Tone) enable
0 = PSG0 (Tone) disable

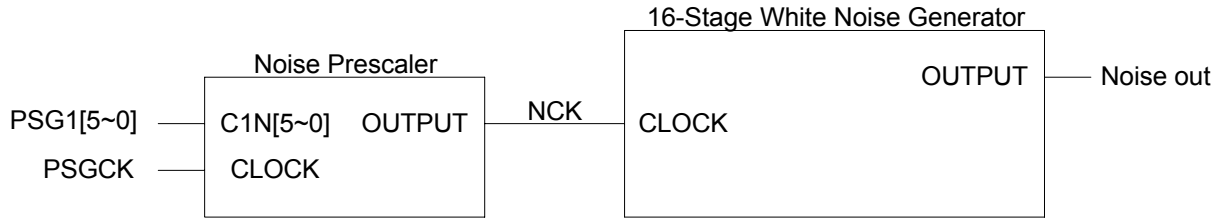
Bit 2: **C1EN :** PSG channel 1 (Tone or Noise) enable bit
1 = PSG1 (Tone or Noise) enable
0 = PSG1 (Tone or Noise) disable

Bit 3: **PRBS :** Tone or Noise generator selection bit
1 = Noise generator
0 = Tone generator

14.3 Noise Generator Control

14.3.1 General description

Noise generator is shown in FIGURE 14-5, which base frequency is controlled by PSG1[5~0].



NCK Frequency = PSGCK/(40H-PSG1[5~0])

FIGURE 14-5 Noise Generator

14.3.2 Noise Generator Programming

DACE defines noise or DAC function. Writing a “1” to C1EN will enable noise generator when PSG is in noise mode.

15. PWM DAC

A built-in PWM DAC is for analog sampling data or voice signals. The structure of DAC is shown in TABLE 15-1. There is an interrupt signal from DAC to CPU whenever DAC data update is needed and the same signal will decide the sampling rate of voice. In DAC mode, the frequency of RC oscillator can't less 2M Hz.

TABLE 15-1 Summary Of DAC Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$012	**PSG1L**	W	PSG1[7]	PSG1[6]	PSG1[5]	PSG1[4]	PSG1[3]	PSG1[2]	PSG1[1]	PSG1[0]	0000 0000
$013	**PSG1H**	W	-	-	-	-	PSG1[11]	PSG1[10]	PSG1[9]	PSG1[8]	- - - - 0000
$014	**DAC**	W	DAC[7]	DAC[6]	DAC[5]	DAC[4]	DAC[3]	DAC[2]	DAC[1]	DAC[0]	0000 0000
$016	**PSGC**	W	-	PCK[2]	PCK[1]	PCK[0]	DMD[1]	DMD[0]	INH	DACE=1	-000 0000

TABLE 15-2 DAC Data Register (DAC)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$014	**DAC**	W	DAC[7]	DAC[6]	DAC[5]	DAC[4]	DAC[3]	DAC[2]	DAC[1]	DAC[0]	0000 0000

Bit 7~0: **DAC[7~0] :** DAC output data

Note: For Single-Pin Single Ended mode, the effective output resolution is 7 bit.

TABLE 15-3 DAC Control Register (PSGC)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$016	**PSGC**	W	-	PCK[2]	PCK[1]	PCK[0]	DMD[1]	DMD[0]	INH	DACE=1	- 000 0000

Bit 0: **DACE :** PSG play as Tone (Noise) or DAC Generator selection bit
1 = PSG is used as DAC Generator
0 = PSG is used as Tone (Noise) Generator

Bit 1: **INH :** DAC output inhibit control bit
1 = DAC output inhibit
0 = DAC output enable

Bit 3~2: **DMD[1~0] :** DAC output mode selection
00 = Single-Pin mode : 7 bit resolution
01 = Two-Pin Two Ended mode : 8 bit resolution
10 = Reserved
11 = Two-Pin Push Pull mode : 8 bit resolution

15.1 Sample Rate Control

PSG1L and PSG1H control the sample rate. PSG1[11~6] controls PWM repeat times (usually set=111100 for four times of DAC reload) and PSG1[5~0] usually set '1'. The input clock source is controlled by PCK[2~0]. The block diagram is shown as the following:

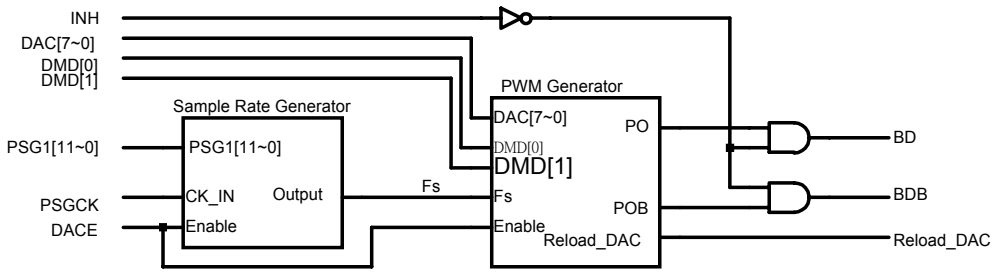
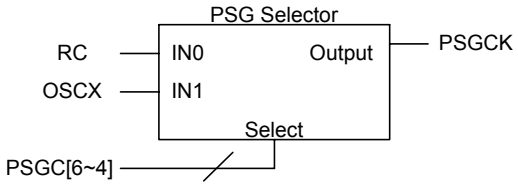


FIGURE 15-1 DAC Diagram



PSGC			PSGCK
B6	B5	B4	
0	0	0	SYSCK
X	0	1	SYSCK/2
X	1	0	SYSCK/4
0	1	1	SYSCK/8
1	0	0	SYSCK x 2
1	1	1	OSCX

FIGURE 15-2 DAC Clock Source Control

TABLE 15-4 DAC Sample Rate Description (RC_{OSC} = 2MHz)

DAC interrupt frequency	PWM frequency	PSGC b6, b5, b4	PSG1H, PSG1L
8K	32K	100	00001111, 00111111
16K	32K	100	00001111, 10111111

15.2 PWM DAC Mode Options

The PWM DAC generator has three modes, Single-pin mode, Two-pin two-ended mode and Two-pin push pull mode. They are depended on the application used. The DAC mode is controlled by DMD[1~0]. (TABLE 13-3)

15.2.1 Single-Pin Mode (7-bit Accuracy)

Single-pin mode is designed for use with a single-transistor amplifier. It has 7 bits of resolution. The duty cycle of the PSG1 is proportional to the output value. If the output value is 0, the duty cycle is 50%. As the output value increases from 0 to 63, the duty cycle goes from being high 50% of the time up to 100% high. As the value goes from 0 to -64, the duty cycle decreases from 50% high to 0%. PSG0 is inverse of PSG1's waveform. Figure 13-3 shows the PSG1 waveforms.

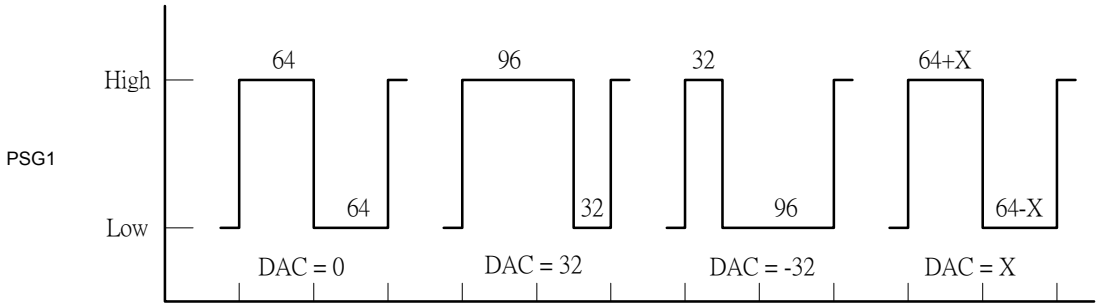


FIGURE 15-3 Single-Pin Mode Wave Form

15.2.2 Two-Pin Two Ended Mode (8-bit Accuracy)

Two-Pin Two-Ended mode is designed for use with a single transistor amplifier. It requires two pin that **PSG0** and **PSG1**. When the DAC value is positive, **PSG1** goes high with a duty cycle proportional to the output value, while **PSG0** stays high. When the DAC value is negative, **PSG0** goes low with a duty cycle proportional to the output value, while **PSG1** stays low. This mode offers a resolution of 8 bits.

Figure 13-5 shows examples of DAC output waveforms with different output values. Each pulse of the DAC is divided into 128 segments per sample period. For a positive output value x=0 to 127, **PSG1** goes high for X segments while **PSG0** stays high. For a negative output value x=0 to -127, **PSG0** goes low for |X| segments while **PSG1** stays low.

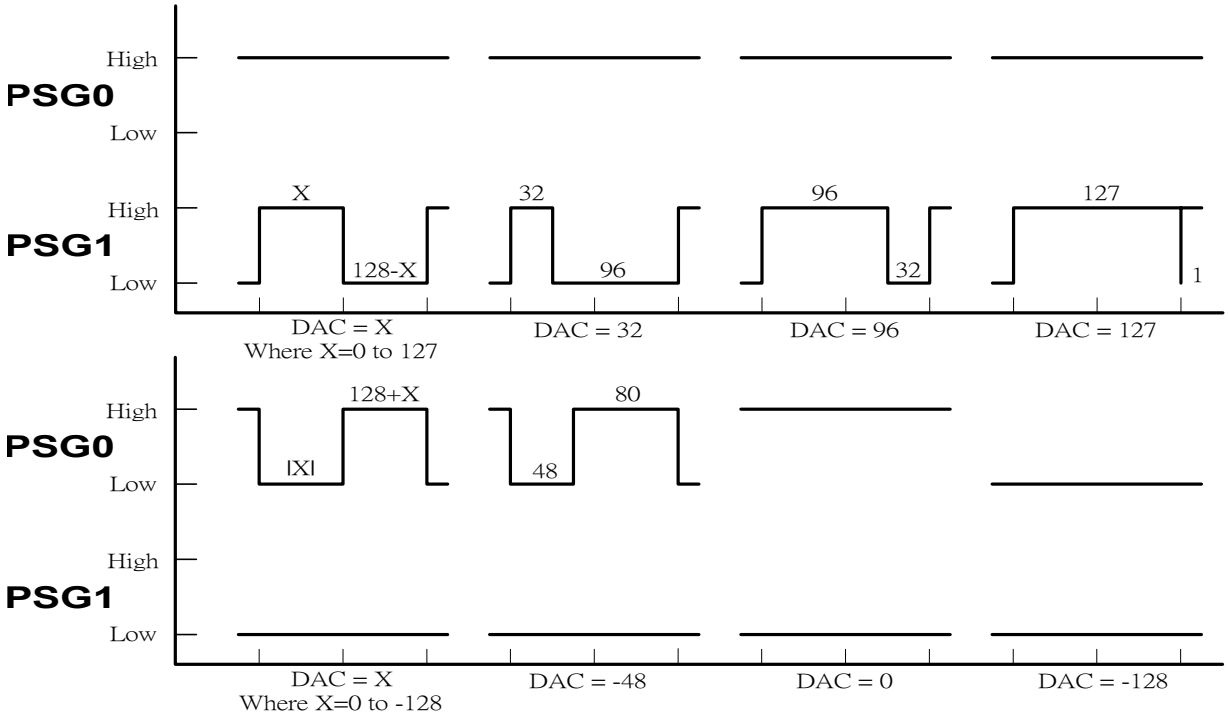


FIGURE 15-4 Two-Pin Two Ended Mode Wave-Form

15.2.3 Two-Pin Push Pull Mode (8-bit Accuracy)

Two-Pin Push Pull mode is designed for buzzer. It requires two pin that **PSG0** and **PSG1**. When the DAC value is 0, both pins are low. When the DAC value is positive, **PSG1** goes high with a duty cycle proportional to the output value, while **PSG0** stays low. When the DAC value is negative, **PSG0** goes high with a duty cycle proportional to the output value, while **PSG1** stays low. This mode offers a resolution of 8 bits.

Figure 13-7 shows examples of DAC output waveforms with different output values. Each pulse of the DAC is divided into 128 segments per sample period. For a positive output value x=0 to 127, **PSG1** goes high for X segments while **PSG0** stays low. For a negative output value x=0 to -127, **PSG0** goes high for |X| segments while **PSG1** stays low.

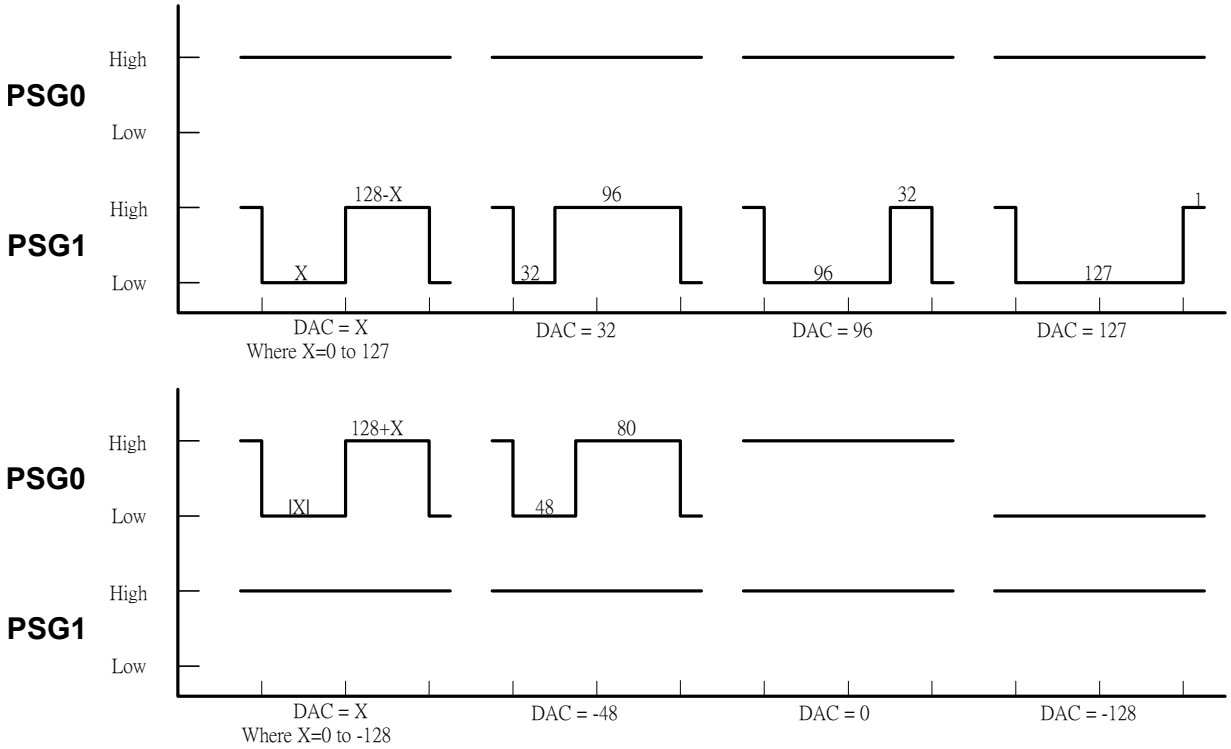


FIGURE 15-5 Two-Pin Push Pull Mode Wave Form

16. LCD CONTROLLER (LCDC)

The LCD controller (LCDC) provides display data and specific signals for external LCD drivers to drive the STN LCD panels. The LCDC fetches display data directly from internal system memory through one unique memory bus. The special designed internal bus shares almost none of the CPU resources to make both fast display data process and high speed CPU operation possible. The ST2202 builds in 4K bytes SRAM, so the maximum panel size can be 240x120. The LCDC also supports software grayscale to rich the display information and the diversity of contents as well.

LCDCK is for LCDC to generate timings and the pixel clock. It is from OSCK instead of SYSCK, therefore frame content retains while SYSCK slows down. Refer to TABLE 11-3 for frequency settings of LCDCK.

The ST2202 supports 1- and 4-bit data bus for the compatibility of most popular LCD drivers. The LCD output signals are shared with Port-L., and are controlled by LCD power control bit **LPWR** (**LCTL[7]**) and data bus selection bit **LMOD** (**LCK[4]**). In case of 1-bit mode, PL3~1 of Port-L can still be used for general purpose.

Note: The LCD signals will be disconnected and Port-L will output values assigned by **PL** after clearing **LPWR**.

Various functions are also supported to rich the display information, including virtual screen, panning, scrolling, contrast control and an alternating signal generator. Control registers used by LCDC are listed below.

TABLE 16-1 Summary Of LCD Control Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$040	**LSSAL**	W	SSA[7]	SSA[6]	SSA[5]	SSA[4]	SSA[3]	SSA[2]	SSA[1]	SSA[0]	0000 0000
$041	**LSSAH**	W	SSA[15]	SSA[14]	SSA[13]	SSA[12]	SSA[11]	SSA[10]	SSA[9]	SSA[8]	0000 0000
$042	**LVPW**	W	VP[7]	VP[6]	VP[5]	VP[4]	VP[3]	VP[2]	VP[1]	VP[0]	0000 0000
$043	**LXMAX**	R/W	XM[7]	XM[6]	XM[5]	XM[4]	XM[3]	XM[2]	XM[1]	XM[0]	0000 0000
$044	**LYMAX**	R/W	YM[7]	YM[6]	YM[5]	YM[4]	YM[3]	YM[2]	YM[1]	YM[0]	0000 0000
$045	**LPAN**	R/W	-	-	-	-	-	PAN[2]	PAN[1]	PAN[0]	- - - - -000
$047	**LCTR**	R/W	LPWR	BLNK	REV	-	-	-	-	-	100- - - - -
$048	**LCK**	W	-	-	-	LMOD	LCK[3]	LCK[2]	LCK[1]	LCK[0]	- - -0 0000
$049	**LFRA**	W	-	-	FRA[5]	FRA[4]	FRA[3]	FRA[2]	FRA[1]	FRA[0]	- - 00 0000
$04A	**LAC**	R/W	-	-	-	AC[4]	AC[3]	AC[2]	AC[1]	AC[0]	- - -0 0000
$04B	**LPWM**	R/W	-	-	LPWM[5]	LPWM[4]	LPWM[3]	LPWM[2]	LPWM[1]	LPWM[0]	- - 00 0000
$04C	**PL**	R/W	PL[7]	PL[6]	PL[5]	PL[4]	PL[3]	PL[2]	PL[1]	PL[0]	1111 1111
$04E	**PCL**	W	PCL[7]	PCL[6]	PCL[5]	PCL[4]	PCL[3]	PCL[2]	PCL[1]	PCL[0]	0000 0000

16.1 LCD Specific Signals

The following signals are generated by LCDC to connect the ST2202 and an LCD panel. Two of them are dedicated output pins, while the rest eight pins are multiplexed with Port-L.

■ FLM (PL7)

The LCD frame marker signal indicates the start of a new display frame. FLM becomes active after the last line pulse of the frame and remains active until the next line pulse, at which point it de-asserts and remains inactive until the next frame.

■ LOAD (PL6)

The LCD line pulse signal is used to latch a line of shifted data to the segment drivers' outputs and is also used to shift the line enable signal of common driver. All the driver outputs then control the liquid crystal to form the desired frame on panel.

■ AC (PL5)

The LCD alternate signal toggles the polarity of liquid crystal on the panel. This signal can be programmed to toggle for a period of 1 to 31 lines or one frame. See section 16.4.7 for register settings.

■ CP (PL4)

The LCD shift clock pulse signal is the clock output to which the output data to the LCD panel is synchronized. Data for segment drivers is shifted into the internal line buffer at each falling edge of CP.

■ LD3~0 (PL3~0)

The LCD data bus lines transfer pixel data to the LCD panel so that it can be displayed. Two kinds of data busses, 1- and 4-bit, are supported and are controlled by **LMOD** (**LCK[4]**). In case of 1-bit mode, **LMOD** should be cleared and the LCDC uses only LD0 to transfer data. LD3~1 can still be programmed to be normal inputs or outputs. The output pixel data can be inverted through programming. Setting **REV** (**LCTR**) will reverse the output data on data bus.

■ $\overline{\text{POFF}}$ (Power control)

The LCD power control signal is used to turn on/off the

external DC-DC converter, which generates a high voltage for driving liquid crystal. $\overline{POFF}$ outputs “1” when clearing **LPWR** (**LCTR**), and outputs “0” by setting this bit, which is also the default value.

■ **$\overline{BLANK}$ (Contrast control)**

The LCD blank signal is used to control the contrast of display by setting contrast level in **LPWM[5:0]** with “00000” (default) represents a maximum level and “11111” is for minimum. The $\overline{BLANK}$ signal achieves this function by outputting a PWM signal according to the settings of contrast. Refer to section 16.4.10 for more information.

Besides contrast control, $\overline{BLANK}$ signal plays another role of turning display off. This is controlled by register bit **BLNK** (**LCTR**). Setting **BLNK** bit will make $\overline{BLANK}$ signal to output “0” to blank the display regardless of contrast control. Setting **BLNK** bit will enable the PWM contrast control and of course the $\overline{BLANK}$ signal. If **LPWMTR[5:0]** are all zeros, $\overline{BLANK}$ signal will stay at high level with no PWM modulation.

16.2 Mapping the Display Data

The screen width and height of the LCD panel are programmable through software. FIGURE 16-1 illustrates the relationship between the portion of a large graphics file displayed on the screen and the actual page. Although the maximum screen size can be up to 1024x512, the actual supported resolution is limited by the display buffer size, which is also the internal RAM size, 4K bytes. Each bit in the display memory corresponds to a pixel in the LCD panel. FIGURE 16-1 also shows the mapping of the display data to the LCD.

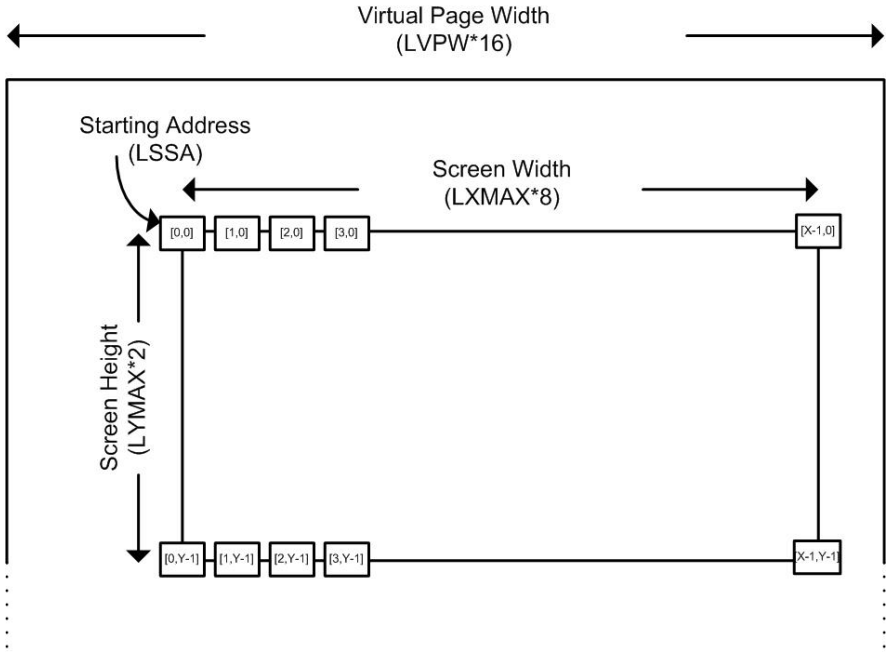


FIGURE 16-1 LCD Screen Format

16.3 LCD Interface Timing

The LCD controller continuously pumps the pixel data into the LCD panel via the LCD data bus. The bus is timed by the CP, LOAD, and FLM signals. Two kinds of data width, 1- and 4-bit, are supported for most monochrome LCD panels. Refer to FIGURE 16-2 for both 1- and 4-bit interface timing.

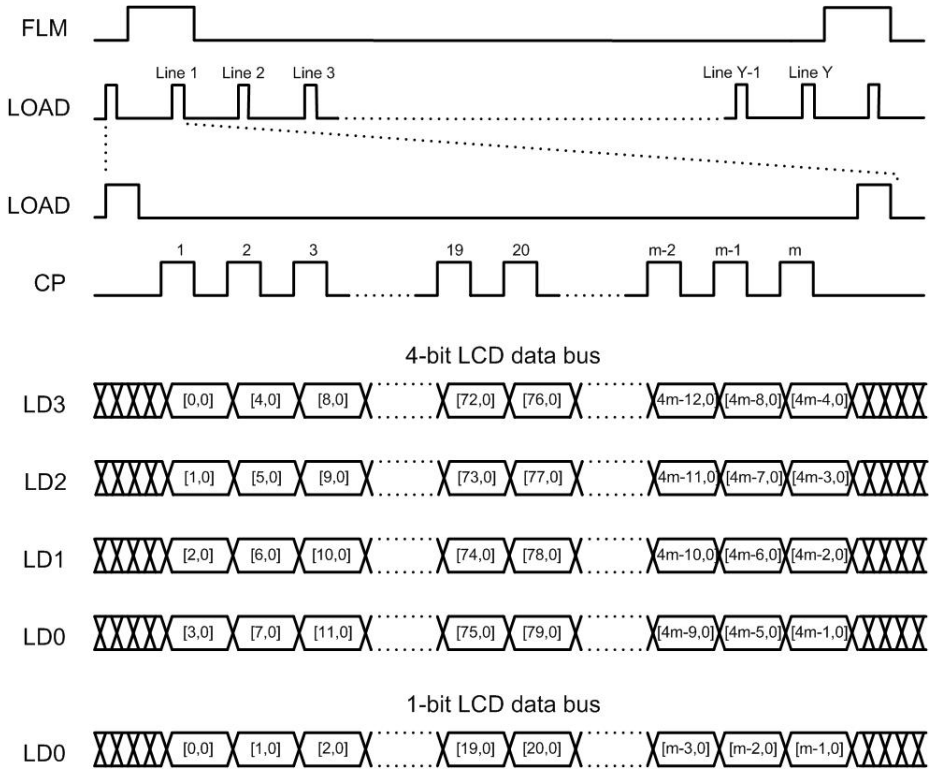


FIGURE 16-2 LCD Interface Timing for 1-/4-Bit Data

16.4 Control Registers

16.4.1 LCD Screen Starting Address Register

The LCD screen starting address register (**LSSA**) is used to inform the starting address of current display buffer. Different LCD frames can be switched quickly by simply modifying content of **LSSA**. The LCD controller will start fetching pixel data from system memory at this address.

TABLE 16-2 LCD Screen Starting Address Register

Address	Name	R/W	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8	Default
$040	**LSSAL**	W	SSA[7]	SSA[6]	SSA[5]	SSA[4]	SSA[3]	SSA[2]	SSA[1]	SSA[0]	0000 0000
$041	**LSSAH**	W	SSA[15]	SSA[14]	SSA[13]	SSA[12]	SSA[11]	SSA[10]	SSA[9]	SSA[8]	0000 0000
Bit 15~0: **LSSA[15:0]** : 16-bit starting address of display buffer											

16.4.2 LCD Virtual Page Width Register

The LCD virtual page width register (**LVPW**) contains the width of a virtual screen that may be wider than real setting. This field is used for calculating the starting point of next line.

TABLE 16-3 LCD Virtual Page Width Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$042	**LVPW**	W	VP[7]	VP[6]	VP[5]	VP[4]	VP[3]	VP[2]	VP[1]	VP[0]	0000 0000
Bit 7~0: **VP[7:0]** : Width of virtual page width Virtual page with = **LVPW** * 16											

16.4.3 LCD Screen Width Register

The LCD screen width register (**LXMAX**) is used to specify the width of the LCD panel in pixels. Every bit of display data maps to one pixel of LCD panel. **LXMAX** represents number of data in byte of each line.

TABLE 16-4 LCD Screen Width Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$043	**LXMAX**	R/W	XM[7]	XM[6]	XM[5]	XM[4]	XM[3]	XM[2]	XM[1]	XM[0]	0000 0000

Bit 7~0: **XM[7:0]** : LCD screen width
LCD screen width = **LXMAX** * 8

16.4.4 LCD Screen Height Register

The LCD screen height register (**LYMAX**) is used to specify the weight of the LCD panel in pixels.

TABLE 16-5 LCD Screen Height Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$044	**LYMAX**	R/W	YM[7]	YM[6]	YM[5]	YM[4]	YM[3]	YM[2]	YM[1]	YM[0]	0000 0000

Bit 7~0: **YM[7:0]** : LCD screen height
LCD screen height = **LYMAX** * 2

16.4.5 LCD Panning Offset Register

The LCD panning offset register (**LPAN**) is used to control how many pixels the picture is shifted to the left. Values from 0 to 7 can be filled into this register to denote the offset, while 0 means no panning control.

TABLE 16-6 LCD Panning Offset Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$045	**LPAN**	R/W	-	-	-	-	-	PAN[2]	PAN[1]	PAN[0]	- - - - -000

Bit 2~0: **PAN[2:0]** : LCD panning offset from zero to 7 pixels max.

16.4.6 LCD Control Register

The LCD control register (**LCTR**) controls the enabling switch of LCDC, display panel on/off or reverse and the PWM contrast control block.

TABLE 16-7 LCD Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$047	**LCTR**	R/W	LPWR	BLNK	REV	-	-	-	-	-	100- - - - -

Bit 7: **LPWR** : LCDC enable/disable bit
0 = Enable LCDC ($\overline{\text{POFF}}$ signal outputs high level)
1 = Disable LCDC ($\overline{\text{POFF}}$ signal outputs low level)

Bit 6: **BLNK** : LCD display ON/OFF bit
0 = LCD display on ($\overline{\text{BLANK}}$ signal outputs contrast control signal)
1 = LCD display off ($\overline{\text{BLANK}}$ signal outputs low level)

Bit 5: **REV** : LCD display reverse
0 = Normal display
1 = Reverse display

16.4.7 LCD Frame Rate Adjust Register

The LCD frame rate adjust register (**LFRA**) specifies the extended time of each scan line. Thus the frame rate slows down to be the desired value.

Note: **LFRA** must be a number greater than 4.

The adjusted frame rate for 1- and 4-bit modes can be found in the following equations

■ **1-bit Mode**

$$Frame\ Rate = \frac{LCDCK}{16 \cdot (LXMAX + LFRA + 1.5) \cdot LYMAX}$$

Equation14-1

■ **4-bit Mode**

$$Frame\ Rate = \frac{LCDCK}{4 \cdot (LXMAX + LFRA + 1.5) \cdot LYMAX}$$

Equation14-2

16.4.8 LCD Frame Rate Adjust Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$049	**LFRA**	W	-	-	FRA[5]	FRA[4]	FRA[3]	FRA[2]	FRA[1]	FRA[0]	- -00 0000

Bit 5~0: **LFRA[5:0]** : Extended time of each scan line

16.4.9 LCD AC Signal Rate Register

The LCD alternating signal rate register (**LAC**) specifies the time of horizontal lines the alternating signal toggles.

TABLE 16-8 LCD AC Signal Rate Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$04A	**LAC**	R/W	-	-	-	AC[4]	AC[3]	AC[2]	AC[1]	AC[0]	- - -0 0000

Bit 2~0: **AC[4:0]** : Time of horizontal lines the AC signal toggles

AC[4:0]	AC signal
00000	Every frame
00001	Every 3 lines
00010	Every 5 lines
00011	Every 7 lines
:	:
11101	Every 59 lines
11110	Every 61 lines
11111	Every 63 lines

16.4.10 LCD PWM Contrast Control Register

The ST2202 achieves contrast control function by outputting a PWM signal from $\overline{BLANK}$. The duty ratio of PWM signal, also is the contrast level, is controlled by **LPWM[5:0]** with 64 steps.

TABLE 16-9 LCD PWM Contrast Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$04B	**LPWM**	R/W	-	-	LPWM[5]	LPWM[4]	LPWM[3]	LPWM[2]	LPWM[1]	LPWM[0]	- - 00 0000

Bit 5~0: **LPWM[5~0] :** LCD contrast control

LPWM[5:0]	Contrast Level
00000	64 (maximum)
00001	63
00010	62
:	:
11101	3
11110	2
11111	1 (minimum)

17. SERIAL PERIPHERAL INTERFACE

The ST2202 contains one serial peripheral interface (SPI) module to interface with external devices, such as Flash memory, analog-to-digital converter, and other peripherals, including another ST2202. The SPI consists of a master- or slave-configurable interface so that connections of both master and slave devices are allowable. Five signals multiplexed with Port-C are used by SPI. With equipped $\overline{\text{DATA\_READY}}$ and $\overline{\text{SS}}$ (slave-select) control signals and transmit/receive buffers, faster data exchange with fewer software interrupts is easy to be made. Data length is widely supported from 7-bit up to 16-bit to satisfy various applications. One clock generator is provided for the synchronous communication clock SCK, which is sourced from OSCK. FIGURE 17-1 illustrates the block diagram of SPI.

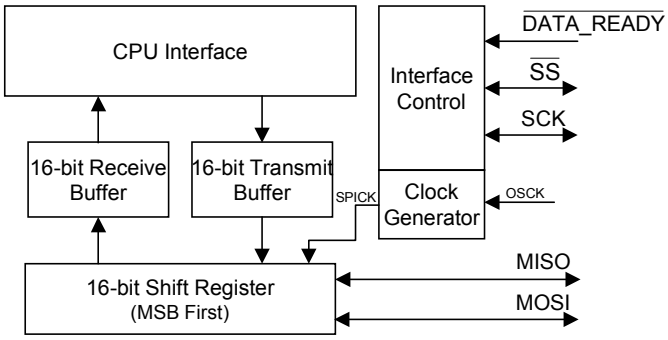


FIGURE 17-1 SPI Block Diagram

17.1 SPI Operations

The SPI contains one 16-bit shift register and two 16-bit buffers for transmission and receiving respectively. Data with variable length from 7-bit to 16-bit can be exchanged with external devices through two data lines. Data length is controlled by bit count register **BC[3:0]** (bit3~0 of SPI clock control register **SCKR)**. The current exchange will be over while the exchanged bit number reaches bit count setting.

The synchronous communication clock SCK is used to synchronize two devices and transfer data in and out of the shift register. Data is clocked by SCK with a programmable data rate, which is assigned by **SCK[2:0]** (bit6~4 of SPI clock control register **SCKR**). Refer to TABLE 11-7 for all clock rate settings.

The SPI block is controlled by **SPIEN** (**SCTR[7]**). Setting **SPIEN** will enable SPI function and the clock divider. Then the internal states of SPI will be reset to initial values. After that, write data to **SDATAL** will initiate an exchange. While exchanging, the busy flag will be set and is reported in **SBZ** (bit 4 of SPI status register **SSR**).

A slave select signal $\overline{\text{SS}}$ (multiplexed with PC4) is used to identify individual selection of a slave SPI device. Slave devices that are not selected do not interfere with SPI bus activities. For a master SPI device, $\overline{\text{SS}}$ can be used to indicate a multiple-master bus contention which can be reported in mode fault bit **MDERR** (bit3 of SPI status register **SSR**).

17.1.1 Clock Phase and Polarity Controls

Four combinations of serial clock (SCK) phase and polarity are selectable by two control bits **PHA** and **POL** (bit 2~1 of SPI control register **SCTR**). FIGURE 17-2 and FIGURE 17-3 show the transmission format of two phase settings.

Note:
The clock settings should be identical for master and the communicating slave device.

■ **Transmission Format – PHA = 0**

In this mode, both master and the communicating slave should present MSB after the falling edge of $\overline{\text{SS}}$. Then the first edge of SCK will be the first capture strobe of input data. If **POL**=0, this first edge is rising edge; if **POL**=1, it will be a falling edge.

■ **Transmission Format – PHA = 1**

In this mode, both master and the communicating slave will be ready after the falling edge of $\overline{\text{SS}}$. The two output MSB at the first edge of SCK. Then the second edge will be the capture strobe. If **POL**=0, the first edge is rising edge; if **POL**=1, it will be a falling one.

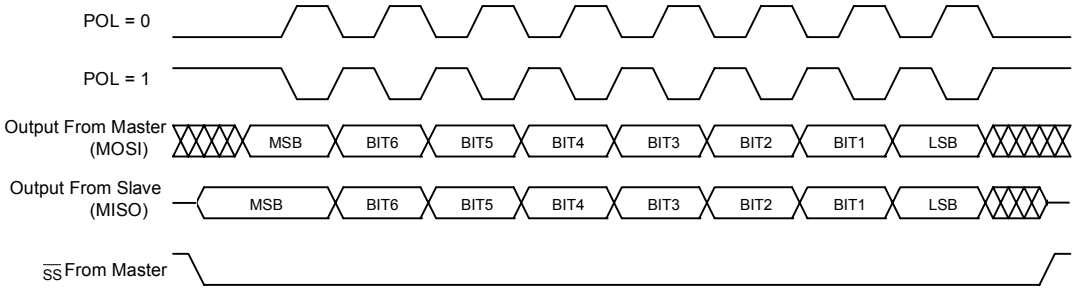


FIGURE 17-2 Transmission Format (PHA = 0)

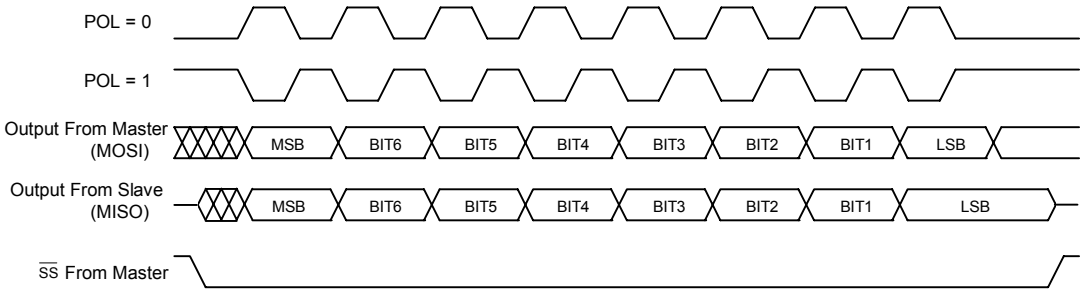


FIGURE 17-3 Transmission Format (PHA = 1)

17.1.2 Transmit Buffer and Receive Buffer

Operations of transmit and receive buffers are discussed below.

■ Transmit Buffer

The transmit buffer is 16-bit long, and is write-only. This buffer is empty after the SPI was enabled at the beginning. In the meantime, the transmit buffer empty flag **TXEMP** (**SSR[5]**) will be set to indicate the status of buffer. Up to 16 bits of data can be filled with writes to SPI data registers (**SDATAL** and **SDATAH**). **TXEMP** will be cleared after **SDATAL** is wrote a value (Writing **SDATAH** will not affect **TXEMP**). Once the shift register proceeds to exchange, data in buffer will be loaded into shift register and **TXEMP** will be set again. Meanwhile a SPI transmitter interrupt will be issued and the transmit buffer can be filled with new data for next transmission.

■ Receive Buffer

The receive buffer is 16-bit long, and is read-only. This buffer is empty after the SPI was enabled first. In the meantime, the receive buffer ready flag **RXRDY** (**SSR[6]**) will be cleared to indicate status of buffer. Two bytes of data can be read from **SDATAL** and **SDATAH**. After completing exchange, data in shift register will be loaded into receive buffer, and then **RXRDY** will be set to indicate that the received data is available. Next, **RXRDY** should be cleared by one read instruction to **SDATAL** (Reading **SDATAH** will not affect **RXRDY)**. In case of master mode, if one completed data is moving into receive buffer and **RXRDY** is still set, the moving activity will no stop but the receive buffer overrun flag **OERR** (**SSR[1]**) will be set to indicate that an old data is overwrote. If it is in slave mode, the receive buffer will not be overwrote while **OERR** equals “1”. **OERR** can be cleared by reading **SDATAL** or by any write to **SSR**.

17.1.3 Master, Slave Modes and The Shift Register

The SPI can operate in master or slave mode according to **SMOD** (**SCTR[0]**). These two modes and operations of the shift register for each are discussed below.

■ Master Mode

The SPI operates as a master device when setting **SMOD**. In this mode, communication clock is provided by ST2202 with SCK (PC1). If there may have more than one master connected, bus contention can be detected by setting mode fault detection bit **MEREN** (**SCTR[4]**). $\overline{SS}$ signal should be input and pulled high temporarily during this detection. Once $\overline{SS}$ inputs low level, a mode fault status can be reported at **MDERR** (**SSR[2]**).

Some SPI devices have $\overline{DATA\_READY}$ output to suspend the incoming transmission. Setting **SRDY** (**PFC[5]**) may include timing of $\overline{DATA\_READY}$, while clearing this bit to discard it. Communication clock and data transmission only starts after $\overline{DATA\_READY}$ returns to low level. The active level of $\overline{DATA\_READY}$ can be inverted to be high level active by setting inversion control bit **DRINV** (**SCTR[3]**).

When transmission, data in shift register will be shifted to master data output MOSI (PC3) with most significant bit (MSB) first, while data from serial data input MISO (PC2)

will be shifted in as well. After the exchanged bits reach bit count setting, current data is complete and then moves to receive buffer.

The exchange continues with auto reload function of shift register if **TXEMP** is cleared. That is, MSB of next data will be sent out and be received in right after the LSB of the previous one with no pause.

After the exchange was triggered, the slave-select signal $\overline{SS}$ (PC4) outputs low level to enable the external slave device. It keeps at low level during exchanges of data and data, and returns to high when exchanges cease.

- **Slave Mode**

In slave mode, $\overline{SS}$ (PC5) and SCK (PC1) become input, while $\overline{DATA\_READY}$ (PC5) is not functional. The exchange takes place only when $\overline{SS}$ inputs low level and ends when it returns to high. On the falling edge of $\overline{SS}$, the shift register will be loaded with data in transmit buffer, and then the exchange initiates. During exchanging, data is clocked by external clock from SCK and is shifted in and out the shift register. Exchanged data will be ready when the exchanged bit number matches bit count setting. After data is ready, data transfer between shift register and two buffers will function automatically as it does in master mode. So that the shift register can be ready for the succeeding clock edge. If $\overline{SS}$ rises before enough data bits, current exchange is over anyway, but the bit count violation flag **BERR** (**SSR[0]**) will be set.

17.1.4 SPI Interrupts

Four interrupts are supported by SPI with two interrupt vectors.

Transmit buffer empty interrupt happens when a data exchange starts and the transmit buffer is empty. This status can be read from status bit **TXEMP** (**SSR[5]**).

Receive buffer ready interrupt happens when a data exchange completes and the receive buffer is filled with one new data. This interrupt is enabled by setting control bit **RXIEN** (**SCTR[6]**). The status is reported at status bit **RXRDY** (**SSR[6]**).

The other two interrupts are error interrupts and are both enabled by control bit **ERIEN** (**SCTR[5]**). Receive buffer overrun interrupt and bit count violation interrupt share the interrupt vector with receive buffer ready interrupt. These three interrupts are “OR” together to generate an individual vector. In master mode, receive buffer overrun interrupt happens when moving new data from shift register to receive buffer with **RXRDY** equals “1”. The overrun interrupt is issued and the status bit **OERR** (**SSR[1]**) will be set. In slave mode, old data in receive buffer will not be flushed while other operations are the same with those in master mode.

Bit count violation interrupt only happens in slave mode. If $\overline{SS}$ input rises before enough data bits are reached, current exchange is over anyway, but the bit count violation flag **BERR** (**SSR[0]**) will be set and the interrupt is issued.

17.2 Interface Signals

Five multiplexed signals are used to interface with other SPI devices. With setting related bits of port function select register **PFC**, these signals can be activated. Direction and function select bits should be ascertained before they are used. Refer to section 9 for these settings.

- **SCK (PC1)**

This is a bidirectional SPI synchronous clock I/O, which is multiplexed with PC1. SCK is output in master mode and input in slave mode.

- **MISO (PC2)**

Master In/Slave Out bidirectional signal, which is multiplexed with PC2. External data is inputted to this pin to the shift register in master mode. In slave mode, it is an output of shift register.

- **MOSI (PC3)**

Master Out/Slave In bidirectional signal, which is multiplexed with PC3. Data in shift register is outputted from this pin in master mode. In slave mode, it is an input of external data to the shift register.

- **$\overline{SS}$ (PC4)**

$\overline{SS}$ is a bidirectional slave-select signal, which is multiplexed with PC4. In master mode, $\overline{SS}$ is output to enable a slave device. In slave mode, $\overline{SS}$ is inputted a low level to trigger the exchange.

- **$\overline{DATA\_READY}$ (PC5)**

$\overline{DATA\_READY}$ is an input signal, which is multiplexed with PC5. It is used only in master mode and can be a GPIO in slave mode. The operation of $\overline{DATA\_READY}$ can be enabled by setting **PFC[5]**. The default active level is high, and can be inverted by setting **DRINV** (**SCTR[3]**). Active level is inputted to indicate that the communicating slave is ready for data exchange.

17.3 SPI Control/Status Registers

SPI control and status registers are summarized in TABLE 17-1.

TABLE 17-2 Summary Of SPI Control Registers

Address	Name	R/W	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8	Default
$050	**SDATAL**	R/W	SD[7]	SD[6]	SD[5]	SD[4]	SD[3]	SD[2]	SD[1]	SD[0]	???? ????
$051	**SDATAH**	R/W	SD[15]	SD[14]	SD[13]	SD[12]	SD[11]	SD[10]	SD[9]	SD[8]	???? ????
$052	**SCTR**	R/W	SPIEN	RXIEN	ERIEN	MEREN	DRINV	POL	PHA	SMOD	0000 0000
$053	**SCKR**	R/W	-	SCK[2]	SCK[1]	SCK[0]	BC[3]	BC[2]	BC[1]	BC[0]	-000 0000
$054	**SSR**	R	-	RXRDY	TXEMP	SBZ	-	MDERR	OERR	BCERR	-000 -000
		W	Write any value to reset SSR								
$00A	**PCC**	R/W	PCC[7]	PCC[6]	PCC[5]	PCC[4]	PCC[3]	PCC[2]	PCC[1]	PCC[0]	0000 0000
$00D	**PFC**	R/W	RXD0	TXD0	SRDY	SS	MOSI	MISO	SCK	INTX	0000 0000
$03D	**IREQH**	R/W	-	-	-	-	IRURX	IRUTX	IRSRX	IRSTX	- - - - 0000
$03F	**IENAH**	R/W	-	-	-	-	IEURX	IEUTX	IESRX	IESTX	- - - - 0000

17.3.1 SPI Data Registers

TABLE 17-3 SPI Data Registers

Address	Name	R/W	Bit 7/15	Bit 6/14	Bit 5/13	Bit 4/12	Bit 3/11	Bit 2/10	Bit 1/9	Bit 0/8	Default
$050	**SDATAL**	R/W	SD[7]	SD[6]	SD[5]	SD[4]	SD[3]	SD[2]	SD[1]	SD[0]	0000 0000
$051	**SDATAH**	R/W	SD[15]	SD[14]	SD[13]	SD[12]	SD[11]	SD[10]	SD[9]	SD[8]	0000 0000

Bit 7~0: **Write**: Write low byte data to transmit buffer / clear status bit **TXEMP** / trigger an data exchange
Read: Read low byte data from receive buffer / clear status bit **RXRDY**

Bit 15~8: **Write**: Write high byte data to transmit buffer / **Read**: Read high byte data from receive buffer

17.3.2 SPI Control Register

TABLE 17-4 SPI Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$052	**SCTR**	R/W	SPIEN	RXIEN	ERIEN	MEREN	DRINV	POL	PHA	SMOD	0000 0000

Bit 7: **SPIEN :** SPI control bit
0 = SPI disable
1 = SPI enable

Bit 6: **RXIEN :** Receive buffer ready interrupt control bit
0 = Receive buffer ready interrupt disable
1 = Receive buffer ready interrupt enable

Bit 5: **ERIEN :** Two error interrupts control bit
0 = Two error interrupts disable
1 = Two error interrupts enable

Bit 4: **MEREN :** Mode fault detection control bit
0 = Mode fault detection disable
1 = Mode fault detection enable

Bit 3: **DRINV :** $\overline{\text{DATA\_READY}}$ active level selection bit
0 = Active level is high
1 = Active level is low

Bit 2~1: **SPHA/SPOL :** SPI clock polarity and phase control bits
Refer to section 17.1.1

Bit 0: **SMOD :** Master / Slave modes selection bit
0 = Select slave mode
1 = Select master mode

17.3.3 SPI Status Register

TABLE 17-5 SPI Status Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$053	**SSR**	R	-	RXRDY	TXEMP	SBZ	-	MDERR	OERR	BCERR	-000 -000
		W	Write any value to reset SSR								

Bit 6: **RXRDY :** Receive buffer status flag
0 = Receive buffer is empty
1 = Receive buffer is filled with new data and is ready

Bit 5: **TXEMP :** Transmit buffer status flag
0 = Data in transmit buffer is waiting for exchanging
1 = Transmit buffer is empty

Bit 4: **SBZ :** SPI busy flag
0 = SPI is idle
1 = SPI is busy exchanging data

Bit 2: **MDERR :** Mode fault status flag
0 = $\overline{SS}$ signal is at high level and is normal
1 = $\overline{SS}$ signal inputs low level / a mode fault status detected

Bit 1: **OERR :** Receive buffer overrun error flag
0 = No receive buffer overrun error
1 = Receive buffer overrun error occurs

Bit 0: **BERR :** Bit count violation flag
0 = Exchanged data bit number matches bit count setting in slave mode
1 = Exchanged data bit number is less than bit count setting in slave mode

18. UNIVERSAL ASYNCHRONOUS RECEIVER/TRANSMITTER

The ST2202 integrates one universal asynchronous receiver/transmitter (UART), which can be used to communicate with external serial devices. Serial data is transmitted and received at standard bit rates using the internal baud rate generator (BGR), which is controlled by BGR control register **BCTR**. Settings of clock output of BGR (BGRCK) can be found in section 11. FIGURE 18-1 shows the block diagram of UART. Summary of UART control registers is listed in TABLE 18-1.

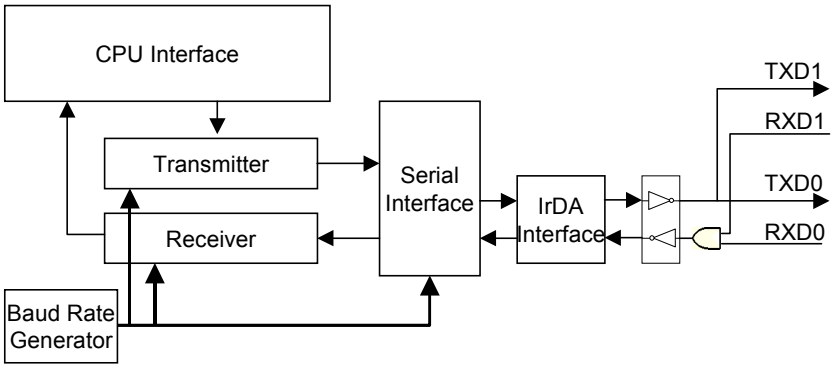


FIGURE 18-1 UART Block Diagram

TABLE 18-1 Summary Of UART Control Registers

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$060	**UCTR**	R/W	-	-	-	-	PEN	PMOD	UMOD	BRK	- - - - 0000
$061	**USTR**	R	-	FER	PER	OER	RXBZ	RXEN	TXBZ	TXEN	-000 0000
		W	-	-	-	-	RXTRG	RXEN	TXTRG	TXEN	- - - - 0000
$062	**IRCTR**	R/W	RXINV	TXINV	-	-	-	PW1	PW0	IREN	00- - -000
$063	**BCTR**	R/W	TEST	-	-	-	-	BSTR	BMOD	BGREN	0- - - -000
$064	**UDATA**	R/W	UD[7]	UD[6]	UD[5]	UD[4]	UD[3]	UD[2]	UD[1]	UD[0]	???? ????
$066	**BRS**	R/W	BRS[7]	BRS[6]	BRS[5]	BRS[4]	BRS[3]	BRS[2]	BRS[1]	BRS[0]	???? ????
$067	**BDIV**	R/W	BDIV[7]	BDIV[6]	BDIV[5]	BDIV[4]	BDIV[3]	BDIV[2]	BDIV[1]	BDIV[0]	???? ????
$00A	**PCC**	R/W	PCC[7]	PCC[6]	PCC[5]	PCC[4]	PCC[3]	PCC[2]	PCC[1]	PCC[0]	0000 0000
$00B	**PCD**	R/W	PCD[7]	PCD[6]	PCD[5]	PCD[4]	PCD[3]	PCD[2]	PCD[1]	PCD[0]	0000 0000
$00D	**PFC**	R/W	RXD0	TXD0	SRDY	SS	MOSI	MISO	SCK	INTX	0000 0000
$00E	**PFD**	R/W	RXD1	TXD1	CS6	CS5	CS4	CS3	CS2	CS1	0000 0000
$03D	**IREQH**	R/W	-	-	-	-	IRURX	IRUTX	IRSRX	IRSTX	- - - - 0000
$03F	**IENAH**	R/W	-	-	-	-	IEURX	IEUTX	IESRX	IESTX	- - - - 0000

18.2 UART Operations

The UART has two modes of operation, NRZ and IrDA, which represent data in different ways for serial communication protocols, RS-232 and IrDA.

18.2.1 NRZ mode

The non-return to zero (NRZ) mode is primarily associated with RS-232. Each character is transmitted as a frame delimited by a start bit at the beginning and a stop bit at the end. Data bits are transmitted least significant bit (LSB) first, and each bit occupies a period of time equal to 1 full bit. If parity is used, the parity bit is transmitted after the most significant bit. Data settings including data length, stop bit number and parity are controlled by bit fields in **UCTR**. FIGURE 18-2 illustrates a character “S” in NRZ mode.

18.2.2 IrDA mode

IrDA mode uses character frames as NRZ mode does, but, instead of driving ones and zeros for a full bit-time period, zeros are transmitted as three-sixteenth (or less) bit-time pulses (which is selected by **PW[1:0]** (**IRCTR[2:1]**), and ones remain low. The polarity of transmitted pulses and expected receive pulses can be inverted so that a direct connection can be made to external IrDA transceiver modules that use active low pulses. This is controlled by **RXINV** and **TXINV** (**IRCTR[7:6]**). IrDA mode is enabled by control bit **IREN** (**IRCTR[0]**). FIGURE 18-3 illustrates a character “S’ in IrDA mode.

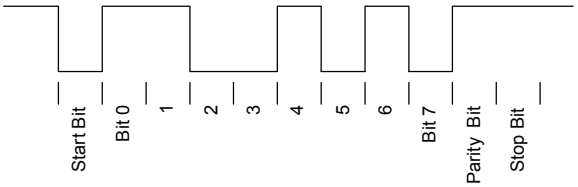


FIGURE 18-2 NRZ ASCII “S” with Odd Parity

FIGURE 18-3 IrDA ASCII “S” with Odd Parity

Two kinds of character, 7-bit and 8-bit, are supported by ST2202. This is controlled by mode selection bit **UMOD** (**UCTR[1]**). Parity options are controlled by parity enable bit **PEN** (**UCTR[3]**) and parity mode selection bit **PMOD** (**UCTR[2]**). Other operations for transmitter and receiver are described below.

18.2.3 Transmitter Operation

Transmitter operation is controlled by control bit **TXEN** (**USTR[0]**). The transmitter accepts a character from the CPU bus, and then transmits it immediately after triggered by writing “1” to control bit **TXTRG** (**USTR[1]**). When a character is available for transmission, the start, stop, and parity (if enabled) bits are added into the character, and then it is serially shifted (LSB first) at the selected bit rate. While transmitter is busy, the busy status is reported at **TXBZ** (**USTR[1]**) with logic value “1”. After all data bits are finished, **IRUTX** (**IREQ[10]**) will be set to issue the interrupt request. Next data transmission may continue with setting trigger bit **TXTRG** again.

If the transmit buffer is empty, the transmitter outputs a continuous idle (which is “1” for normal polarity). Moreover a continuous “0” can also be outputted as a break character by setting **BRK** bit (**UCTR[0]**) and then set the trigger bit.
.

18.2.4 Receiver Operation

Receiver operation is controlled by control bit **RXEN** (**USTR[2]**). Once the receiver is enabled, it searches for a start bit, qualifies it, and then samples the succeeding data bits at the perceived bit center. Jitter tolerance and noise immunity are provided by sampling 16 times per bit and using a voting circuit to enhance sampling. While receiving, the busy status of receiver can be read from **RXBZ** (**USTR[3]**) with logic level “1”.

Receiving activity will be complete after the stop bit is detected. Then **IRURX** (**IREQ[11]**) will be set to issue the interrupt request. The received data can be obtained by reading data register **UDATA**. Then the receive trigger bit **RXTRG** (**USTR[3]**) should be set to indicate that the data register can be overwrote next time.

Three kinds of errors may arise from illegal received data, which are reported at 3 bits of status register **USTR[6:4]** and are discussed below.

1. **Buffer Overrun Error**

This error indicates that the receive trigger bit was not set and the receiver overwrote data in receive buffer, i.e., the previous character was lost. This also means the software is not keeping up with the incoming data rate. Error is updated and reported by reading **OER** (**USTR[4]**) for current received character.

2. **Parity Error**

If parity is enabled, the parity bit of current received character is checked and the status is updated in register bit **PER** (**USTR[5]**).

3. **Framing Error**

This error indicates that a framing error is detected and there may be corrupted data with missing stop bit. Error is updated and reported by reading **FER** (**USTR[6]**) for current received character.

18.3 Interface Signals

Two sets of data lines can be enabled simultaneously for communication, TXD0(PC6), RXD0(PC7) and the auxiliary pins TXD1(PD6), RXD1(PD7). Data can inputs and outputs from and to these pins. With setting related bits of port function select registers (**PFC** and **PFD**), signals of the external devices can be connected. Data in and from these communication I/Os can be inverted by setting polarity control bit **RXINV** and **TXINV** (**IRCTR[7:6]**). Direction settings and function select bits should be ascertained before using signals. Refer to section 9 for these settings.

- **TXD0 (PC6)/TXD1 (PD6)**

The UART transmit data signal is output to one or both of these two pins, which are multiplexed with PC6 and PD6. These pins connect to standard RS-232 or infrared transceiver modules.

- **RXD0 (PC7)/RXD1 (PD7)**

The UART receive data signal is input from one or both of these two pins, which are multiplexed with PC7 and PD7. If RXD0 and RXD1 are enabled at a time, both signals will be gated with AND logic to produce one single signal. These

pins also interface to standard RS-232 and infrared transceiver modules.

18.4 UART Control/Status Registers

18.4.1 UART Control Register

TABLE 18-2 UART Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$060	**UCTR**	R/W	-	-	-	-	PEN	PMOD	UMOD	BRK	- - - - 0000

Bit 3: **PEN :** Parity control bit
0 = Disable parity
1 = Enable parity

Bit 2: **PMOD :** Parity mode selection bit
0 = Even parity
1 = Odd parity

Bit 1: **UMOD :** 7-/8- bit mode selection bit
0 = 7- bit mode (the received data bit 7 will be set to zero)
1 = 8-bit mode

Bit 0: **BRK :** Break character
0 = Normal character
1 = Transmit break character

18.4.2 UART Status Control Register

TABLE 18-3 UART Status Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$061	**USTR**	R	-	FER	PER	OER	RXBZ	RXEN	TXBZ	TXEN	-000 0000
		W	-	-	-	-	RXTRG	RXEN	TXTRG	TXEN	- - - - 0000

Bit 6: **FER :** Received data frame error status bit
0 = Current received data is normal
1 = Frame error occurs

Bit 5: **PER :** Parity error status bit
0 = Current received data is normal
1 = Parity error occurs

Bit 4: **OER :** Overrun error status bit
0 = Current received data is normal
1 = Overrun occurs

Bit 3: **RXBZ :** Receiver busy bit
0 = Receiver is not busy
1 = Receiver is busy

Bit 2: **RXEN :** Receiver control bit
0 = Receiver is disabled
1 = Receiver is enabled

Bit 1: **TXBZ :** Transmitter busy bit
0 = Transmitter is not busy
1 = Transmitter is busy

Bit 0: **TXEN :** Transmitter control bit
0 = Transmitter is disabled
1 = Transmitter is enabled

Bit 3: **RXTRG :** Receiver trigger bit
Writing “1” to make receiver to be ready for next data

Bit 1: **TXTRG :** Transmitter trigger bit
Writing “1” to trigger the transmitter to start transmission

18.4.3 IrDA Control Register

TABLE 18-4 IrDA Control Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$062	**IRCTR**	R/W	RXINV	TXINV	-	-	-	PW1	PW0	IREN	00- - -000

Bit 7: **RXINV :** Receive data inversion bit
0 = Receive data is normal
1 = Receive data is inverted

Bit 6: **TXINV :** Transmit data inversion bit
0 = Transmit data is normal
1 = Transmit data is inverted

Bit 2~1: **PW[1:0] :** IrDA pulse width selection bits

PW[1:0]	Pulse Width
00	1/16
01	2/16
1x	3/16

Bit 0: **IREN :** IrDA mode control bit
0 = Normal mode (NRZ)
1 = IrDA mode

18.4.4 UART Data Register

TABLE 18-5 UART Data Register

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$064	**UDATA**	R/W	UD[7]	UD[6]	UD[5]	UD[4]	UD[3]	UD[2]	UD[1]	UD[0]	???? ????

Write: Write character data to transmitter / **Read**: Read character data from receiver

18.5 Settings For Standard Baud Rates

One clock of 16 times of the communication baud rate is needed by the UART to perform data transmission/receiving, synchronization, and parity/error operations. Settings of **BRS**, **BDIV**, and OSCK ranges for standard baud rates are listed in TABLE 18-6. Besides, fine modulation mode and full modulation strength are suggested when using BGR to generate clock for UART. Store value of $03 to **BCTR** to select these two options.

TABLE 18-6 Settings For Standard Baud Rates

Baud Rate	BRS	BDIV	OSCK(MHz)		Error (%)
			Max.	Min.	
600	27	92	1.893	1.645	0.17
	31	106	2.173	1.889	0.18
	34	116	2.384	2.072	0.05
	53	181	3.716	3.230	0.05
	61	208	4.277	3.717	0.10
	68	232	4.768	4.144	0.05
1200	27	46	1.893	1.645	0.17
	31	53	2.173	1.889	0.18
	34	58	2.384	2.072	0.05
	54	92	3.786	3.290	0.17
	61	104	4.277	3.717	0.10
	68	116	4.768	4.144	0.05
2400	30	26	2.104	1.828	1.54
	34	29	2.384	2.072	0.05
	38	32	2.664	2.316	1.33
	54	46	3.786	3.290	0.17
	61	52	4.277	3.717	0.00
	68	58	4.768	4.144	0.05
4800	26	11	1.822	1.584	0.85
	30	13	2.104	1.828	1.54
	33	14	2.313	2.011	0.57
	54	23	3.786	3.290	0.17
	61	26	4.277	3.717	0.10
	68	29	4.768	4.144	0.05
9600	28	6	1.963	1.707	0.44
	33	7	2.313	2.011	0.57
	37	8	2.594	2.254	1.33
	56	12	3.927	3.413	0.44
	61	13	4.277	3.717	0.10
	66	14	4.628	4.022	0.57
19200	28	3	1.963	1.707	0.44
	37	4	2.594	2.254	1.33
	47	5	3.251	2.825	0.27
	56	6	3.927	3.413	0.44
	65	7	4.557	3.961	0.95
28800	28	2	1.963	1.707	0.44
	42	3	2.945	2.559	0.44
	56	4	3.927	3.413	0.44
	70	5	4.908	4.266	0.44
	72	5	5.048	4.388	2.40
38400	19	1	1.332	1.158	1.33
	37	2	2.594	2.254	1.33
	56	3	3.927	3.413	0.44
	57	3	3.996	3.474	1.33
	75	4	5.259	4.571	0.00
57600	28	1	1.963	1.707	0.44
	55	2	3.856	3.352	2.22
	56	2	3.927	3.413	0.44
	57	2	3.996	3.474	1.33
115200	55	1	3.856	3.352	2.22
	56	1	3.927	3.413	0.44
	57	1	3.996	3.474	1.33

Example:
When Baud Rate=600,BRS=27,BDIV=92,we need to choose a OSCK to be between 1.645MHz and 1.893MHz

19. DIRECT MEMORY ACCESS (DMA)

To speed up the memory access of this system, a sequential direct memory access (DMA) controller is designed-in. DMA can perform memory transfer function more efficient than CPU does. While DMA working, data ROM register (DRR) will disable and DMA use DMA memory bank register (DMR) to access ROM. After DMA complete, ROM bank control still return to DRR.
With the help of DMR can make DMS across bank boundary smoothly, but DMR is only valid for DMS. **The DMR can automatic increases when DMS across bank boundary.**

Note: Location of source data can not fall in the range of internal SRAM(that is the range of 0100H~FFFH).

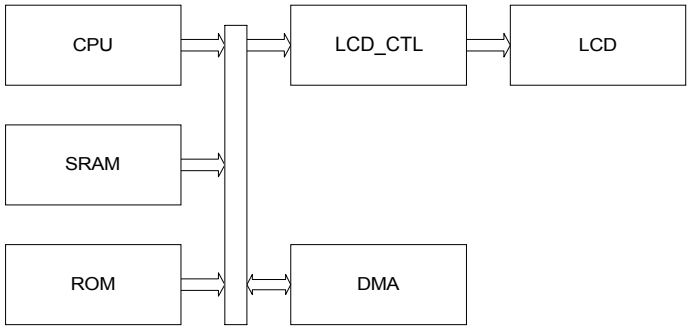


FIGURE 19-1 System Block Diagram

19.1 DMA Control Register

The control register is shown as following:

TABLE 19-2 DMA Control Register (LCTL)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$028	**DMSL**	W	DMS[7]	DMS[6]	DMS[5]	DMS[4]	DMS[3]	DMS[2]	DMS[1]	DMS[0]	???? ????
$029	**DMSH**	W	DMS[15]	DMS[14]	DMS[13]	DMS[12]	DMS[11]	DMS[10]	DMS[9]	DMS[8]	???? ????
$02A	**DMDL**	W	DMD[7]	DMD[6]	DMD[5]	DMD[4]	DMD[3]	DMD[2]	DMD[1]	DMD[0]	???? ????
$02B	**DMDH**	W	DMD[15]	DMD[14]	DMD[13]	DMD[12]	DMD[11]	DMD[10]	DMD[9]	DMD[8]	???? ????
$02C	**DCNTL**	W	DCNT[7]	DCNT[6]	DCNT[5]	DCNT[4]	DCNT[3]	DCNT[2]	DCNT[1]	DCNT[0]	???? ????
$02D	**DCNTH**	W	-	-	-	DMAM	DCNT[11]	DCNT[10]	DCNT[9]	DCNT[8]	- - -? ????
$036	**DMRL**	R/W	DMR[7]	DMR[6]	DMR[5]	DMR[4]	DMR[3]	DMR[2]	DMR[1]	DMR[0]	0000 0000
$037	**DMRH**	R/W	-	-	-	-	-	DMR[10]	DMR[9]	DMR[8]	- - - - -000

DMS[15:0] : DMA source data starting address register

DMD[15:0]: DMA destination data starting address register

DCNT[11:0]: DMA moving data byte counter register

DMR[10:0]: DMA source data bank register (**DMR** works only when **DMS** is in the range from 8000h to FFFFh).

DMAM (**DCNTH[4]**)**:** DMA destination address increasing mode selection bit
0 = Destination address increases automatically.
1 = Destination address is fixed.

The DMA always move (DCNT+1) bytes of data. DMA will start right after CPU write data into register DCNTL. During the DMA operation, the CPU hold, until the DMA transfer completed. The DMR register reset to “$00” on real chip, but Emulation Board is “unknown”, so recommend initial DMR register before use. **Before Read/Write you have to initial the PRR, DRR, DMR register when system reset.**

19.2 DMA Programming Flow

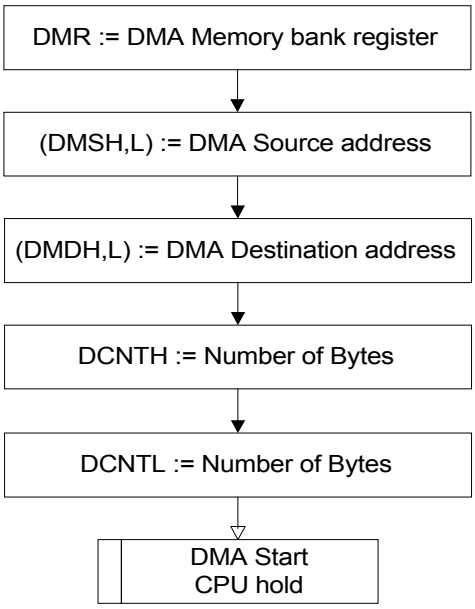


FIGURE 19-2 DMA Programming Flow

19.3 Example Program 1:

This program fills “00” to address $1000~$12FF.

```
STZ     $1000           ;; “00” to $1000
STZ     <DMSL
LDA     #$10
STA     <DMSH           ;; source = $1000
STA     <DMDH
LDA     #$01
STA     <DMDL           ;; destination = $1001
LDA     #$02
STA     <DCNTH
LDA     #$FE
STA     <DCNTL          ;; move $2FF bytes
:
:
```

19.4 Example Program 2:

This program moves data in address $1080~$12FF to $1000~$127F.

```
LDA     #$80
STA     <DMSL
LDA     #$10
STA     <DMSH       ;; source = $1080
STA     <DMDH
STZ     <DMDL       ;; destination = $1000
LDA     #$02
STA     <DCNTH
LDA     #$7F
STA     <DCNTL          ;; move $280 bytes
  :
  :
```

19.5 Application Program 3:

This program moves data in address $8000~$803F one single port at $0200.

```
STZ     <DMSL
LDA     #$80
STA     <DMSH       ;; source = $8000
STZ     <DMDL
LDA     #$02
STA     <DMDH       ;; destination = $0200
LDA     #$10
STA     <DCNTH
LDA     #$3F
STA     <DCNTL          ;; move $40 bytes
  :
  :
```

20. POWER DOWN MODES

ST2202 has three power down modes: WAI-0, WAI-1 and STP. The instruction WAI will enable either WAI-0 or WAI-1, which is controlled by **WAIT** (**SYS[2]**). And the instruction STP will enable **STP** mode in the same manner. WAI-0 and WAI-1 modes can be waked up by interrupt. However, **STP** mode can only be waked up by hardware reset.

TABLE 20-1 System Control Register (SYS)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$030	**SYS**	R	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	HIGH	0000 0001
		W	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	LVDEN	0000 0000

Bit 2: **WAIT :** WAI-0 / WAI-1mode select bit
0 = WAI instruction causes the chip to enter WAI-0 mode
1 = WAI instruction causes the chip to enter WAI-1 mode

20.1 WAI-0 Mode:

If **WAIT** is cleared, WAI instruction makes MCU enter WAI-0 mode. In the mean time, the oscillator, interrupts, timer/counter, and PSG are still working. On the other hand CPU and the related instruction execution stop. All registers, RAM, and I/O pins will retain the same states as those before the MCU entered power down mode. WAI-0 mode can be waked up by reset or interrupt request even If user sets interrupt disable flag **I**. In that case MCU will be waked up but not entering interrupt service routine. If interrupt disable flag is cleared (**I**='0'), the corresponding interrupt vector will be fetched and the service routine will be executed. The sample program is shown below:

```
LDA     #$00
STA     <SYS
WAI                 ; WAI 0 mode
```

20.2 WAI-1 Mode:

If **WAIT** is set, WAI instruction makes MCU enter WAI-1 mode. In this mode, CPU stops, but the PSG, timer/counter keep running if their clock sources are from OSCX. The wake-up procedure is the same as for WAI-0. The difference is that the warm-up cycles occur when waking from WAI-1. Sample program is shown as following:

```
LDA     #$04
STA     <SYS
WAI                 ; WAI 1 mode
```

20.3 STP Mode:

STP instruction will force MCU to enter stop mode. In this mode, MCU stops, but PSG, timer/counter won't stop if the clock source is from OSCX. In power-down mode, MCU can only be waked up by hardware reset, and the warm-up cycles occur at the same time.

FIGURE 20-1 Status Under Power Down Modes

SYSCK source is OSC:

Mode	Timer0,1	SYSCK	LCD	OSC	OSCX	Base Timer	RAM	REG.	I/O	Wake-up condition
WAI-0	Retain									Reset, Any interrupt
WAI-1	Stop	Stop	Stop	Stop	Retain					Reset, Any interrupt
STP	Stop	Stop	Stop	Stop	Retain					Reset

SYSCK source is OSCX:

Mode	Timer0,1	SYSCK	OSC	OSCX	Base Timer	RAM	REG.	I/O	LCD	Wake-up condition
WAI-0	Retain								Wrong Frame	Reset, Any interrupt
WAI-1	Stop	Stop	Retain						Stop	Reset, Any interrupt
STP	Stop	Stop	Retain						Stop	Reset

21. WATCHDOG TIMER

The watchdog timer (WDT) is an added check that a program is running and sequencing properly. When the application software is running, it is responsible for keeping the 2- or 8-second watchdog timer from timing out. If the watchdog timer times out, it is an indication that the software is no longer being executed in the intended sequence. At this time the watchdog timer generates a reset signal to the system.

21.1 WDT Operations

The WDT is enabled by setting the WDT enable flag **WDTEN** (**MISC[3]**). Two time settings, 2 and 8 seconds, are selectable with selection bit **WDTPS** (**MISC[2]**).WDT is clocked by the 2Hz clock from the base timer and therefore has 0.5-second resolution. It is recommended that the watchdog timer be periodically cleared by software once it is enabled. Otherwise, software reset will be generated when the timer reached a binary value of 4 or 16.

Note:The WDT can be reset by writing any value to **MISC** register.

After a system reset, **WDTEN** is cleared. Then the WDT returns to be idle.

TABLE 21-1 System Miscellaneous Register (MISC)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$038	**MISC**	R	-	-	-	-	WDTEN	WDTPS	TEST	TEST	- - - - 0000
		W	Reset WDT								

Bit 2: **WDTPS :** WDT time-out time selection bit
0 = 8 seconds
1 = 2 seconds

Bit 3: **WDTEN :** WDT control bit
0 = Disable WDT
1 = Enable WDT

Bit 1~0: **TEST :** These two bits should be both zero in normal operation

22. LOW VOLTAGE DETECTOR

ST2202 has a built-in low voltage detector for power management. The typical active level of voltage detection is 2.6V. When **LVDEN** (**SYS[0]**) is set, detector circuit is enabled and the detection result will be outputted at the same bit after 3 μs. Using read instruction twice can get this result: first read will enable initial stableness control. Second read equal '0' represents 'low voltage'. Once low voltage detector is enabled, it keeps on consuming power. So it is important that remember to write “0” to LVDET to disable the detector after detection is completed. One sample program is shown below:

```
Start:
    SMB0    <SYS        ; enable detector
        :
      Wait 3 μs
        :
    SEC
    BBS0    <SYS,$+3
    BBS0    <SYS,Normal_Voltage
Low_Voltage:
    CLC
Normal_Voltage:
    RMB0    <SYS        ; disable detector
```

TABLE 22-1 System Control Register (SYS)

Address	Name	R/W	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Default
$030	**SYS**	R	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	HIGH	0000 0001
		W	XSEL	OSTP	XSTP	XBAK	WSKP	WAIT	IRREN	LVDEN	0000 0000

Bit 0: **LVDEN** : Low voltage detector control bit (W)
0 = Disable detector
1 = Enable detector

Bit 0: **HIGH** : Low voltage detector result (R)
0 = Voltage is low
1 = Voltage is normal

23. ELECTRICAL CHARACTERISTICS

23.1 Absolute Maximum Rations

DC Supply Voltage ---------- -0.3V to +6V
Operating Ambient Temperature --------- -10°C to +60°C
Storage Temperature ---------- -10°C to +125°C

***Notice:** Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. All the ranges are stress ratings only. Functional operation of this device at these or any other conditions above those indicated in the operational sections of this specification is not implied or intended. Exposed to the absolute maximum rating conditions for extended periods may affect device reliability.

23.2 DC Electrical Characteristics

Standard operation conditions: VCC = 3.0V, GND = 0V, T_A = 25°C, OSC = 2M Hz, unless otherwise specified

Parameter	Symbol	Min.	Typ.	Max.	Unit	Condition
Operating Voltage	VCC	2.4		5.5	V	
Operating Frequency	F_1			3.0	MHz	VCC = 2.4V ~ 5.5V
Operating Frequency	F_2			4.0	MHz	VCC = 2.7 ~ 5.5V
Operating Current	I_{OP}		770	1100	μA	All I/O port are input and pull-up, execute NOP instruction, LCDC on
Standby Current	I_{SB0}		210	250	μA	All I/O port are input and pull-up, OSCX on, LCDC on (WAIT0 mode)
Standby Current	I_{SB1}		1.2		μA	All I/O port are input and pull-up, OSCX on, LCDC off (WAIT1 mode)
Input High Voltage	V_{IH}	0.7Vcc 0.85Vcc		Vcc+0.3	V V	Port-A/B/C/D/E/L $\overline{\text{RESET}}$
Input Low Voltage	V_{IL}	GND-0.3		0.3Vcc 0.15Vccc	V V	Port-A/B/C/D/E/L $\overline{\text{RESET}}$
Pull-up resistance	R_{IH}	240	245	250	KΩ	Port-A/B/C/D/E/L (input Voltage=0.7VCC)
Output high voltage	V_{OH1}	0.7Vcc			V	Port-A/B/C/D/L (IOH=-3.5mA)
Output high voltage	V_{OH1}	0.7Vcc			V	Port-B (I_{OH}=-5.5mA)
Output low voltage	V_{OL1}			0.3Vcc	V	Port-A/B/C/D/E/L (IOL=7.5mA)
Output high voltage	V_{OH2}	0.7Vcc			V	PSG/DAC, I_{OH} = -30mA.
Output low voltage	V_{OL2}			0.3Vcc	V	PSG/DAC, I_{OL} = 70mA.
Low voltage detector active level	V_{LVD}		2.6		V	
Low voltage detector current	I_{lvdet}		127		uA	

23.3 AC Electrical Characteristics

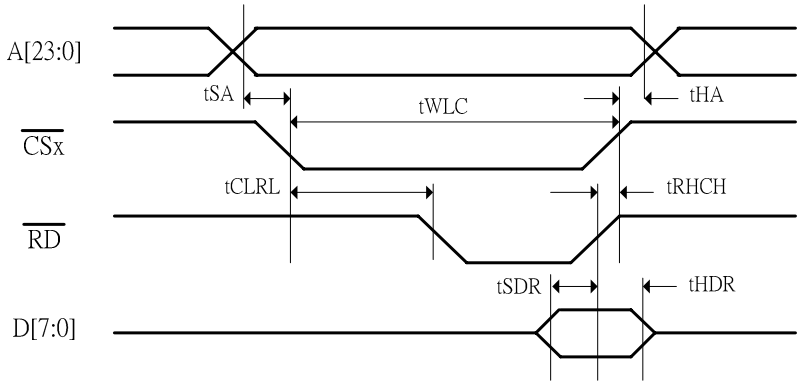


FIGURE 23-1 External Read Timing Diagram

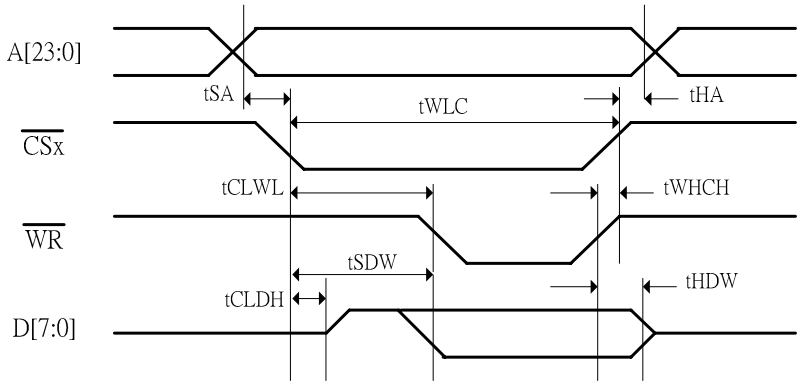


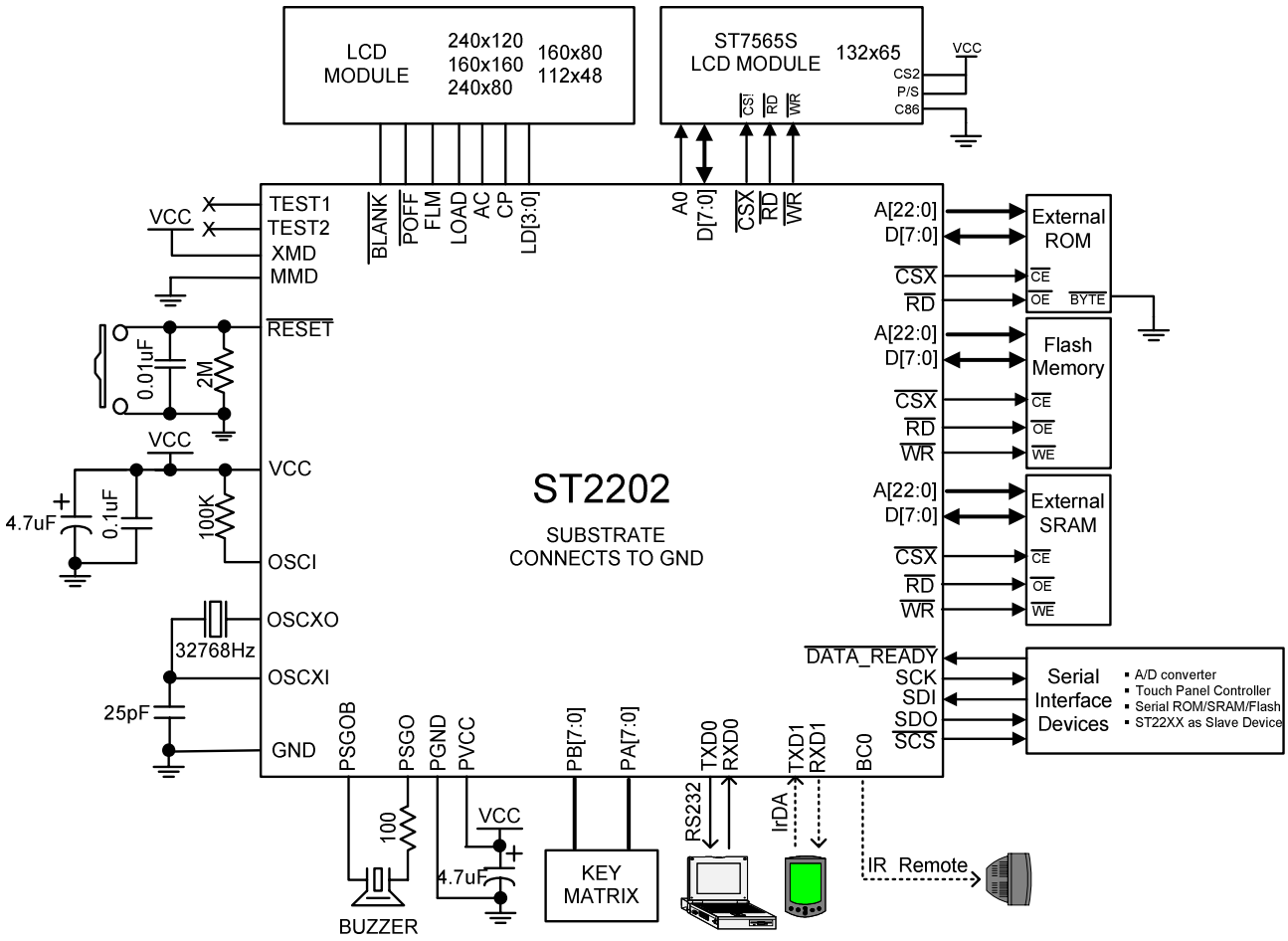
FIGURE 23-2 External Write Timing Diagram

TABLE 23-1 Timing parameters for FIGURE 23-1 and FIGURE 23-2

Standard operation conditions: VCC = 3.0V, GND = 0V, T_A = 25°C

Symbol	Characteristic	Rating			Unit
		Min.	Typ.	Max.	
tSA	Address setup time	—	—	10	ns
tHA	Address hold time	0	—	—	ns
tWLC	CS “L” pulse width	166	—	—	ns
tCLWL	CS asserted to $\overline{WR}$ asserted	—	1/2 tWLC	—	ns
tWHCH	CS negated after $\overline{WR}$ is negated	10	—	—	ns
tCLDH	CS asserted to data outputs high	10	—	—	ns
tSDW	CS asserted to data-out is valid	—	1/2 tWLC	—	ns
tHDW	Data-out hold time after $\overline{WR}$ is negated	20	—		ns
tCLRL	CS asserted to $\overline{RD}$ asserted	—	1/2 tWLC	—	ns
tRHCH	CS negated after $\overline{RD}$ is negated	10	—	—	ns
tSDR	Data-in valid before $\overline{RD}$ is negated	30	—	—	ns
tHDR	Data-in hold time after $\overline{RD}$ is negated	10	—	—	ns
tR	Signal rise time	—	20	—	ns
tF	Signal fall time	—	10	—	ns

24. APPLICATION CIRCUITS



Note: 1. The capacitor connected to $\overline{\text{RESET}}$ should be of the value not greater than 0.01uF.

2. The resistor in parallel with the reset capacitor helps a lot to generate correct reset signal and should not be removed. The drawback is an additional current of about 1.5 uA rises.

3. The resistor in parallel with the oscillator resistor can stabilize the oscillation clock. It can be removed if the oscillation clock is not higher than 4Mhz. Note that the resistance should not be greater than 1000pF.

4. Connect $\overline{\text{OE}}$ signal of an external ROM to $\overline{\text{RD}}$ of ST2202 instead of GND to prevent conflict of data bus when software error of writing to ROM occurs.

25. REVISIONS

REVISION	DESCRIPTION	PAGE	DATE
2.0	■ Change application circuit of connecting an external ROM	63	2003/5/5
	■ Add Section 21.3 AC Electrical Characteristics	62	
	■ Move pad diagram and device information to the front		
1.9	■ Remove the capacitor in parallel with the oscillation resistor.	60	2003/1/2
	■ Modify typical active level of low voltage detector to be 2.6V	58, 59	2003/4/2
1.8	■ Modify application circuit of section 22. See notes below the figure.	60	2002/11/20
1.7	■ Fix errors on addresses of **BRS** and **BDIV** in TABLE 11-6	21	2002/11/4
	■ Modify TABLE 16-6 Settings for Standard Baud Rates in section 16-5	52	2002/11/13
	■ Forbid using SRAM as the source of DMA in section 17	53	2002/11/13
1.6	■ Modify connection of XMD from NC to VCC in section 24	59	2002/9/25
	■ Switch definition of PC4 and PC5, **PFC[4]** and **PFC[5]**	2,3,7,43,44,45	2002/9/25
	■ Modify definition of $\overline{\text{DATA\_READY}}$ active level selection bit in TABLE 17-3	46	2002/9/25
	■ Modify description of oscillator mode selection in first paragraph	18	2002/9/25
1.5	■ Add operation frequency range	1, 58	2002/9/4
	■ Add section 16.2 and 16.3	39,40	2002/9/4
1.4	■ Modify description of pin XMD. Connection “NC” is not allowed	3	2002/6/24
	■ Modify description of pin MMD/ $\overline{\text{CS0}}$ and FIGURE 10-1. One resistor should be added between VCC and this pin when using Emulation mode.	3,16	2002/6/24
	■ Modify selection method of high frequency crystal oscillator from “Code option” to “Bonding option”	1	2002/6/24
	■ Add description of control bit **INTEG** and TABLE 9-9	15	2002/6/24
	■ Change R/W ability of **LSSAL** and **LSSAH** to be write-only	8, 38, 39	2002/6/24
	■ Modify TABLE 15-4 DAC Sample Rate Description	34	2002/6/24
	■ Add chapter 21: DC characteristics	58	2002/9/4
1.3	■ Modify description of pin MMD/ $\overline{\text{CS0}}$	3,16	2002/6/12
	■ Add figures of connection of pin MMD/ $\overline{\text{CS0}}$	16	2002/6/12
1.2	■ Correct description of output levels of $\overline{\text{POFF}}$ in TABLE 16-7	38	2002/6/05
	■ Add application circuit	55	2002/6/10
	■ Change name of TEST2 pin to MMD, change name of TEST3 pin to TEST2. Modify description of MMD	3, 15	2002/6/10
	■ Add section 18.5 “Settings for Standard Baud Rates”	48	2002/6/10
1.1	■ Modify bit0 of system control register **SYS**	6,7,17,20,52,54	2002/4/20
	■ Add two pins XMD and TEST3	2, 3	2002/6/04

	■ Add device information and pad diagram	55,56	2002/6/04
	■ Change PSG outputs' name to PSGO and PSGOB		2002/6/04
	■ Modify LCD frame rate equations 14-1 and 14-2	38	2002/6/04
1.0	Second release		2002/4/15