Winbond Electronics Corp.

60" Voice/Melody/LCD Controller (ViewTalk™ Series)

Publication Release Date: March 1999

GENERAL DESCRIPTION

The W53322/W53342 are a high-performance 4-bit microcontroller (μC) with built-in speech, melody and 32*48/32*64 LCD driver which includes internal pump circuit. The 4-bit uc core contains dual clock source, 4-bit ALU, two 8-bit timers, one divider, 20 pin Input or output, 7 interrupt sources , 8-level subroutine nesting for interrupt applications. Speech unit can be implemented with Winbond 60-sec Power Speech using ADPCM algorithm. Melody unit provides dual tone output and can store up to 1k notes. Power reduction mode is also built in to minimize power dissipation. It is ideal for games, educational toys, remote controllers, watches, clocks and the other application's products which may incorporate both LCD display and melody.

FEATURES

- Operating voltage: 2.4volt ~ 5.5volt
- Dual clock operating system
  - –RC/Crystal (400 KHz to 4 MHz) for main clock
  - – 32.768 KHz crystal oscillation circuit for sub-oscillator
- Memory
  - – 16k × 20 bit program ROM
  - – 896x4/1024 × 4 bit general data RAM (384x4/512x4 shared wih LCD)
- 20 input/output pins
  - –Ports for input only: 2 ports/8 pins
  - –Input/output ports: 2 ports/8 pins
  - –Port for output only: 1 port /4 pins
- Power-down mode
  - –Hold function: no operation (except for 32kHz oscillator)
  - –Stop mode function: no operation (include 32kHz oscillator)
- Seven types of interrupts
  - –Five internal interrupts (Divider ,Timer 0, Timer 1, Speech, Melody)
  - –Two external interrupts (Port RC, Port RD)
- One built-in 14-bit clock frequency divider circuit
- Two built-in 8-bit programmable countdown timers
  - –Timer 0: one of the two internal clock frequencies (FOSC/4 or FOSC/1024) can be selected
  - –Timer 1: built-in auto-reload function includes internal timer with FOSC, FOSC/64 and 8KHz clock source option or TONE output function which can be used as IR carrier output if main clock is 455kHz)
- Built-in 18/14-bit watchdog timer for system reset by mask code option
- Powerful instruction sets
- 8-level subroutine (including interrupt) nesting
- LCD driver output
  - –32 common × 48/64 segment
  - – 1/16 duty or 1/32 duty, 1/5 or 1/7 bias, internal pump circuit option by special register
- Speech function
  - –Provide 1.4M bits dedicated speech ROM
  - –With direct driving output for speaker

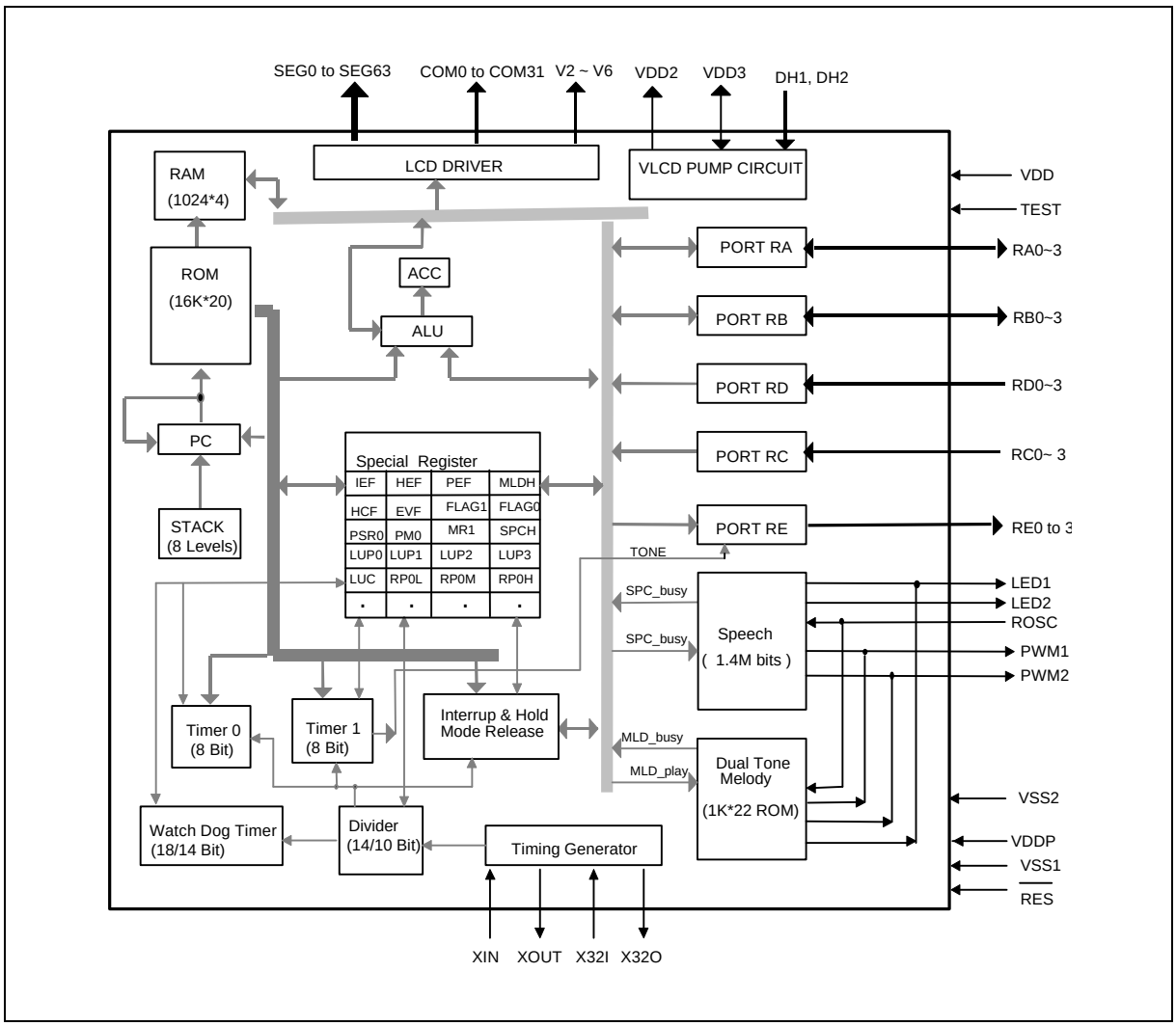
Publication Release Date: March 1999

Winbond Electronics Corp.

–Maximum 256 sections available

- Melody function
  - –Provide 22 kbits dedicated melody ROM
  - –Provide 6 kinds of beat, 16 kinds of tempo, and pitch rang from G3# to C7
  - –Tremolo, triple frequency and 3 kinds of percussion available
  - –With direct driving output for speaker
  - –Maximum 32 scores available
- Mix speech with melody available
- Multi-engine controller
- PWM output current option
- Chip On Board available

BLOCK DIAGRAM



PIN DESCRIPTION

SYMBOL	I/O	FUNCTION
XIN	I	Input pin for oscillator. It can be connected to crystal, or can connect a resistor to VDD to generate main system clock. Oscillator can be stopped when SCR.1 is set to logic 1.
XOUT	O	Output pin for oscillator which is connected to another crystal pin.
X32I	I	32.768 KHz crystal input pin.
X32O	O	32.768 KHz crystal output pin.
RA0 ~ RA3	I/O	General Input/Output port specified by PM1 register. If output mode is selected, PM0 register can be used to specify CMOS/NMOS driving capability option. Initial state is input mode.
RB0 ~ RB3	I/O	General Input/Output port specified by PM2 register. If output mode is selected, PM0 register can be used to specify CMOS/NMOS driving capability option. Initial state is input mode.
RC0 ~ RC3	I	4-bit schmitt input with internal pull high option specified by PM0 register. Each pin has an independent interrupt capability specified by PEFL special register.
RD0 ~ RD3	I	4-bit schmitt input port with internal pull high option specified by PM0 register. Each pin has an independent interrupt capability specified by PEFH special register.
RE0~RE3/TONE	O	Output port only. RE3 may use as TONE if bit 0 of MR0 special register is set to logic 1.
$\overline{\text{RES}}$	I	System reset pin with internal pull-high resistor is active low.
TEST	I	Test pin. Connected to low for normal use.
ROSC	I	Connect resistor to VDD to generate speech or melody clock source.
VDDP	I	Power source for PWM output.
LED1	O	Synchronous LED1 output while speech play/melody is active.
LED2	O	Synchronous LED2 output only while speech play is active.
PWM1	O	Speaker direct driving output 1 while speech or melody is active.
PWM2	O	Speaker direct driving output 2 while speech or melody is active.
SEG0–SEG31/47/63	O	LCD segment output pins.
COM0–COM31	O	LCD common signal output pins. The LCD alternating frequency is fixed at 64Hz.
DH1, DH2	I	Connection terminals for voltage doubler capacitor.
VDD2	O	Connect a 1uF capacitor to VSS1 to double VDD voltage output if triple pump option is enabled. Otherwise, VDD2 connects to VDD directly if double pump option is enabled.
VDD3	O/I	An output if internal pump circuit is enabled. It connects a 1uF capacitor to VSS. Triple VDD voltage will be output if triple pump option is enabled. Otherwise, double VDD voltage will be output if double pump option is enabled. An input if internal pump voltage is disabled.
V2 ~ V5	O	LCD COM/SEG output driving voltage. If internal shunt resistor is disabled, external resistors need to be supplied to V2, V3, V4, V5 . A capacitor is suggested for stable LCD voltage level.
V6	I	External variable resistor connects between VDD3 and V6 to adjust LCD maximun voltage level.
VDD	I	Microcontroller Positive power supply (+).
VSS1	I	Negative power supply (-).
VSS2	I	Negative power supply (-).

FUNCTIONAL DESCRIPTION

Four main units are included in the W533X2. They are 4bit uc, power speech, dual tone melody and 32 com * 32/48/64 seg LCD driver. The 4bit uc is modified from winbond W741C260 that many features are enhanced such as ROM space, RAM space and addressing capability, more and more instruction sets , 7 interrupt sources, controlling speech and melody playing to drive speaker directly and so on. We separate three parts PART A , PARTB and PART C to explain function more detailly.

PART A: UC FUNCTION

Program Counter (PC)

Organized as an 14-bit binary counter (PC0 to PC13), the program counter generates the addresses of the 16K x20 on-chip ROM containing the program instructions. When the jump , subroutine call instructions , the interrupt , initial reset conditions are executed, the address corresponding to the instruction will be loaded into the program counter. The format used is shown Table 1:

ITEM	ADDRESS	INTERRUPT PRIORITY
Initial Reset	0000H	-
INT 0 (DIV)	0004H	1st
INT 1 (TM 0)	0008H	2nd
INT 2 (RC)	000CH	3rd
INT 3 (RD)	0010H	4th
INT 4 (Reserved)	-	-
INT 5 (SPEECH)	0018H	5th
INT 6 (MELODY)	001CH	6th
INT 7 (TM 1)	0020H	7th
JP Instruction	XXXXH	-
Subroutine Call	XXXXH	-

Table 1: Interrup Address Assignment & Priority

Stack Register (STACK)

The stack register is organized as 14bits × 8 levels (first-in, last-out). When either a call subroutine or an interrupt is executed, the program counter will be pushed onto the stack register automatically. At the end of a call subroutine or an interrupt service subroutine, the RTN instruction must be executed to pop the contents of the stack register into the program counter. When the stack register is pushed over the eighth level, the contents of the first level will be lost. In other words, the stack register is maximun with 8 subroutine nesting.

Program Memory (ROM)

1. Architecture

The read-only memory 16K x20 bit is used to store program codes addressed PC from 0000h~3FFFH. Location from 000H to 0020H are reserved for interrupt service as shown Figure 1. All instruction sets are one word, one cycle. Look-up table function is provided to access ROM code in all 16k ROM spaces. The organization of the program memory is shown in Figure 1.

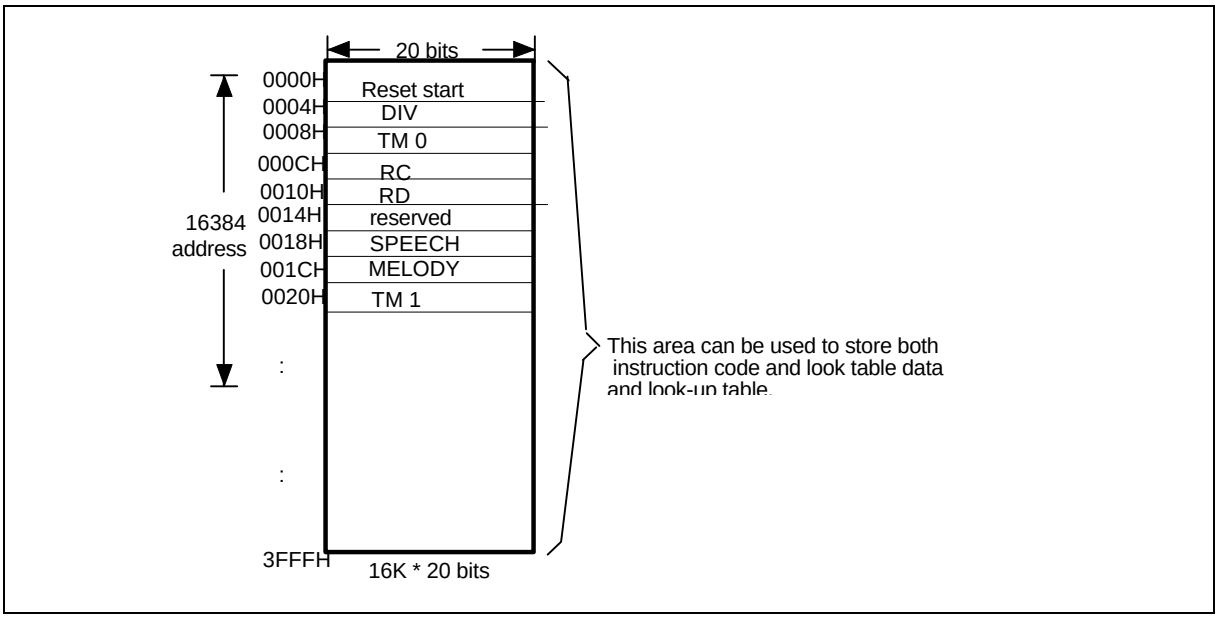


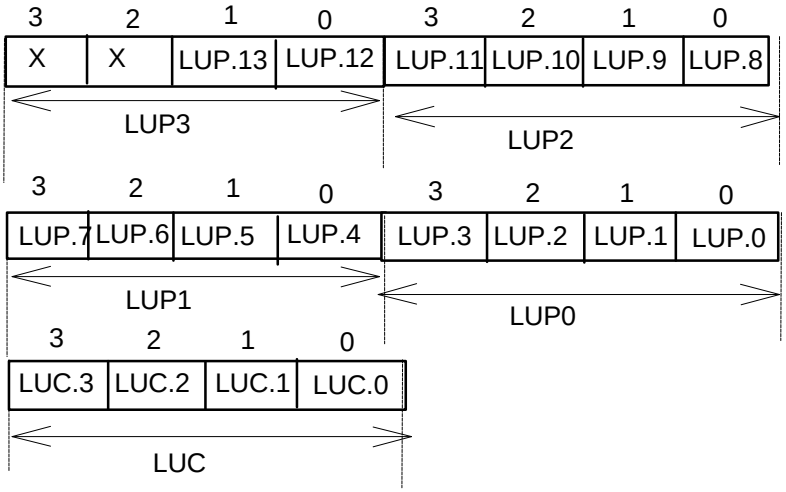
Figure 1. Program Memory Organization

2. Look-Up Table Pointer Register(LUP3, LUP2, LUP1, LUP0 and LUC)

The LUPC (Look-up table address Pointer Counter) symbol in the instruction set is used to access data in the 16K ROM space. It includes 5 registers LUP3 (Look-Up table Pointer), LUP2, LUP1, LUP0, and LUC(Look-Up table data Counter) . LUP3~LUP0 store the 14 bits ROM address to access any data in the 16K word ROM, and each ROM word 20 bit is separated as 5 nibbles that LUC counts from 0 to 4 cyclical. The instruction MOV LUPn, ACC can write LUPC initial address pointer of look-up table, and reset LUC register to 0. So the following equation is described.

LUPC=LUPC.13~LUPC.0 + LUC.3~LUC.0

LUPC.13~LUPC.0 from 0000~3FFFH is used as word address of 16K ROM and LUC uses to select which nibble of the word data and counts from 0 to 4 cyclical. When LUPC is incresased by 1 LUC is firstly increase by 1 , and LUP0 will be inreased by 1 while LUC is counted from 4 to 0. The LUP1 is increased by 1 if LUP0 is counted from F to 0, then LUP3, LUP2 will follow the same rule of LUP1. The LUC will be increased by 1 automatically while symbol @LUPC++ is used. All registers LUP3~LUP0 can read/write by user, but LUC register is reald only. At initial reset, all registers is 0000B.



Data Memory (RAM)

1. Architecture

The static data memory (RAM) is arranged as maximun 512+(384/512) × 4 bits. The data memory can be addressed directly or indirectly. The organization of the data memory is shown in Figure 2 using W53322 as example. The first 512 nibbles RAM from 000 to 1FFH is dedicated for general data memory. Data memory from 200H to 37FH/3FFH has two roles either LCD dedicated pattern data memory as Table 5 mapping or general data memory because they have the same addressing capability as 000H to 1FFH. There are two data memory address point RP0 (RP0H+RP0M+RP0L) and RP1 (RP1H+RP1M+RP1L) that programmer can use indirect addressing instruction such as MOV ACC, @RP0 or MOV @RP1, @RP0 to move data between different data memory range and ACC. We also provide instruction between ROM and RAM such as MOV @RP0, @LUPC that user can move look-up table data in ROM to general RAM easily. The instruction MOV @RP0++, @LUPC++ also provides point counter is incresaed by 1 automatically after instruction is executed. Please refer to instruction sets description for more detail.

The first sixteen addresses (00H to 3FH) in the data memory are known as the page 0 working registers. Only working register can operate directly with immediate data. There is one special register WRPAGE from 0h to 0DH to select working register page

2. Working Register Page (WRPAGE with SR=30H)

The special register WRPAGE is organized as a 4-bit and it counts from 0 to 0DH to separate 896 nibbles RAM as 14 pages. Every page is included 64 nibbles. The bit descriptions are as follows:

	3	2	1	0
WRPAGE	R/W	R/W	R/W	R/W

Bit 3~0: 0000~1011 Page 0 to Page 0DH
Bit3~0: 1100~1111 is inhibited.
All bits are read/write by user. At initial reset, the WRPAGE is 0000B.

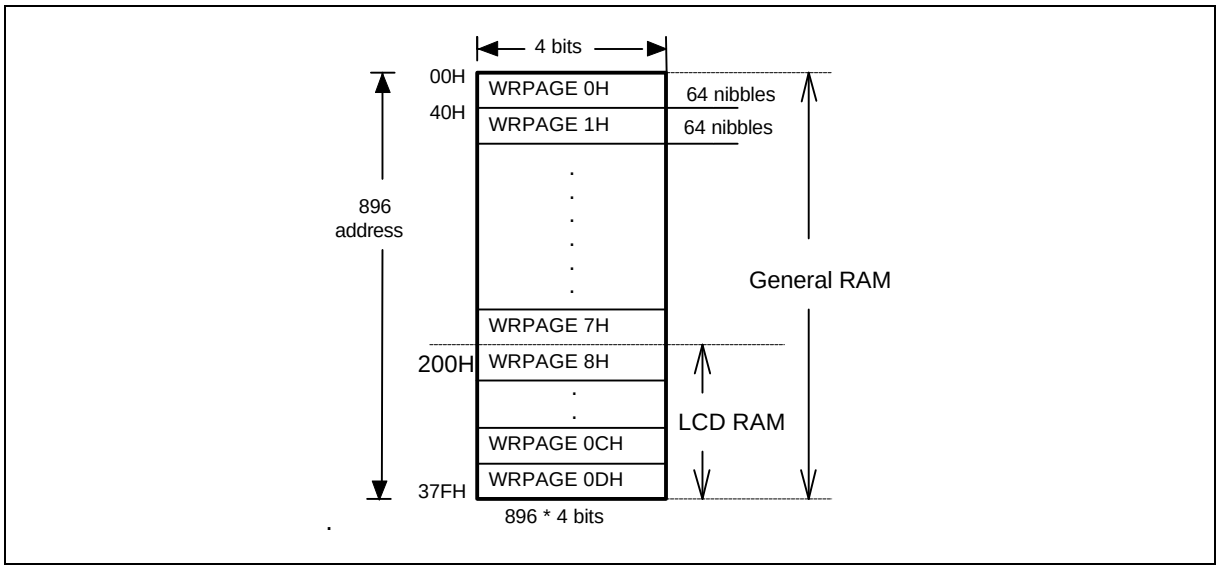
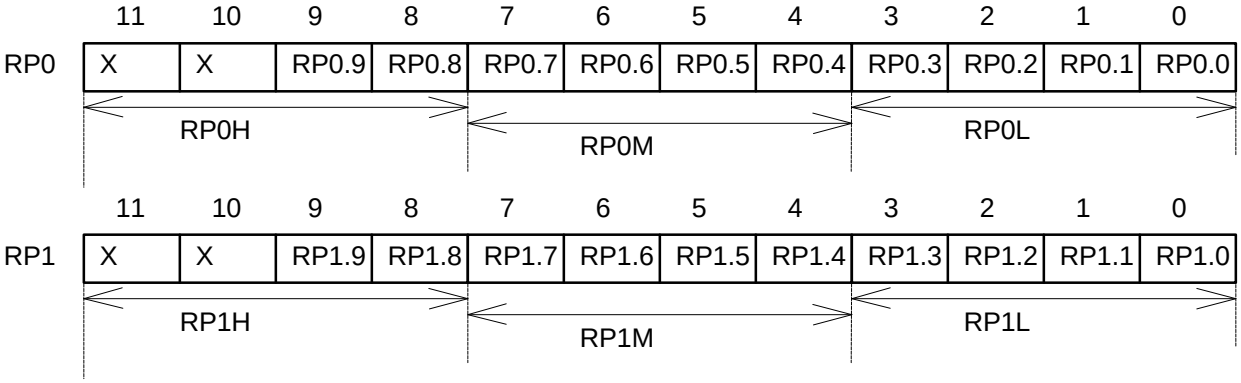


Figure 2. Data Memory Organization

3. RAM Point Register (RP0L, RP0M, RP0H,, RP1L, RP1M, RP1L)

There are two RAM points 0 and 1 that user uses it to access data easily by direct or indirect addressing. RAM Point 0 (RP0) is organized as 10 bit RP0.9~RP0.0 that 3 special registers are used RP0L, RP0M and RP0H. RAM Point 1 (RP1) has the same structure as RP0, so RP1L, RP1M and RP1H are needed.



All bits in RP0, RP1 is read/written by user. At initial reset, all RPn data is 0000B.

Special Rgister and Special Register Pair(SR & SRP)

There are some special registers formatted as 4 bit per register shown as Table 2 that chip operating condition is depended of special register value. Programmer can use such as "MOV SR, #I" or "CLR SR" or "SET SR" command to write suitable value to control chip operatiing state. Some special register such HEF, IEF and HCF can write 8 bit immediate data simultaneously by Special Register Pair (SRP) command that we format as MOV SRP, #I. All special register function will be described detailly while close relation function is introduced.

SRP	SR	SR Symbol	Function	Bit 3 ~ 0 assignment
	00	-----		
	01	-----		
	02	TM0L(w)	low nibble of Timer 0	TM0.3~TM0.0
	03	TM0H(w)	high nibble of Timer 0	TM0.7~TM0.4
	04	TM1L(w)	low nibble of Timer 1	TM1.3~TM1.0
	05	TM1H(w)	high nibble of Timer 1	TM1.7~TM1.4
	06	TMC1L(r)	low nibble of Timer 1	TM1.3~TM1.0
	07	TMC1H(r)	high nibble of Timer 1	TM1.7~TM1.4
	08	-----		
	09	-----		
05H	0A	EVFL(r,c)	Event Flag (set by chip hardware	RD,RC,TM0,DIV
	0B	EVFH(r,c)	if interrupt is occurred)	TM1,SPEECH,MELODY,X
06H	0C	HEFL(r/w,s/c)	Hold mode release Enable Flag	RD,RC,TM0,DIV
	0D	HEFH(r/w,s/c)		TM1,SPEECH,MELODY,X
07H	0E	IEFL(r/w,s/c)	Interrup Enable Flag	RD,RC,TM0,DIV
	0F	IEFH(r/w,s/c)		TM1,SPEECH,MELODY,X
	10	HCFL(r)	Hold mode release Condition Flag	RD,RC,TM0,DIV
	11	HCFH(r)	(set by H/W if hold mode is released)	TM1,SPEECH,MELODY,X
	12	LDIV(w)	Divider of LCD fundamental frequency	LDIV.3 ~ LDIV.0
	13	-----		
	14	PEFL(r/w,s/c)	Port Enable Flag for hold mode	RC.3, RC.2, RC.1, RC.0
	15	PEFH(r/w,s/c)	release or interrupt function	RD.3, RD.2, RD.1, RD.0
	16	RP0L(r/w)	RAM address Pointer 0 Low nibble	RP0.3~RP0.0
	17	RP0M(r/w)	RAM address Pointer 0 Middle nibble	RP0.7~RP0.4
	18	RP1L(r/w)	RAM address Pointer 1 Low nibble	RP1.3~RP1.0
	19	RP1M(r/w)	RAM address Pointer 1 Middle nobble	RP1.7~RP1.4
	1A	RP0H(r/w)	RAM address Pointer 0 High nibble	X,X, RP0.9,RP0.8
	1B	RP1H(r/w)	RAM address Pointer 1 High nibble	X,X, RP1.9,RP1.8
	1C	MLDL(w)	MeLoDy score address Low nibble	MLD.3~MLD.0
	1D	MLDH(w)	MeLoDy score address High nibble	MLED1,MLED0, OSB,MLD.4
	1E	SPCL(w)	SPeeCh section address Low nibbel	SPC.3~SPC.0
	1F	SPCH(w)	SPeeCh section address High nibbel	SPC.7~SPC.4
	20	CF(r,s/c)	Carrier Flag	X,X,X,CF
	21	-----		
	22	FLAG0(r/w,s/c)	melody/speech busy and play flag	MLD_busy,SPC_busy,MLD_play, SPC_play
	23	FLAG1(c)	reset flag for Divider/WatchDog	X,DIVR,WDTR,X
	24	-----		
	25	-----		

	26	MR2(r/w,s/c)	special register for interrupt enable	X,X,X,INTEN
	27	MR3(w)	Mode register 3	Vlcd, FsENB, PWM1, PWM0
	28	MR0(r/w,s/c)	Mode Register 0 for timer	TM0EN,TM1EN,LCDEN,TONE
	29	MR1(w)	Mode Register 1 for timer	WDTCK,TM0CK,TM1SR,TM1CK
	2A	LCDM1(w)	LCD Mode register 1	COM32B,BIAS7B,PUPV3B,INTSRB
	2B	SCR(r/w,s/c)	System Control Register	DIV5MB,FMRCB,FMEN,F32IN
	2C	LUP0(r/w)	Look UP table address pointer first nibble	LUPC.3~LUPC.0
	2D	LUP1(r/w)	Look UP table address pointer 2nd nibble	LUPC.7~LUPC.4
	2E	LUP2(r/w)	Look UP table address pointer 3th nibble	LUPC.11~LUPC.8
	2F	LUP3(r/w)	Look UP table address pointer 4th nibble	X,X,LUPC.13,LUPC.12
	30	WRPAGE(r/w)	Working Register PAGE register	0000~1101H
	31	LUC(r)	LUPC nibble counter	0000~0100H
	32	PM0(r/w,s/c)	Port Mode Register 0	RD_PH,RC_PH,RB_NM,RA_NM
	33	-----		
	34	PSR0(r,clr-all)	Port RC Status change Register	RC3EG~RC0EG
	35	PSR1(r,clr-all)	Port RD Status chabge Register	RD3EG~RD0EG
	36	PM1(r/w,s/c)	Port RA I/O Mode select Register	RA3IN~RA0IN
	37	PM2(r/w,s/c)	Port RB I/O Mode select Register	RB3IN~RB0IN
	38	PORTA(r/w)	PORT data of RA	RA3~RA0
	39	PORTB(r/w)	PORT data of RB	RB3~RB0
	3A	PORTC(r)	PORT data of RC	RC3~RC0
	3B	PORTD(r)	PORT data of RD	RD3~RD0
	3C	PORTE(w)	PORT data of RE	RE3~RE0
	3D~3F	------		

Note 1: "r, w, s,c " means "read, write, set, and clear" separately)

Note 2: "clr-all " means all 4 bit will be clear simultaneously.

Note 3: X means don't care bit

Table 2: Special Register address mapping

Accumulator (ACC)

The accumulator (ACC) is a 4-bit register used to hold results from the ALU and transfer data between the data memory, I/O ports, and special registers.

Arithmetic and Logic Unit (ALU)

This is a circuit which performs arithmetic and logic operations. The ALU provides the following functions:

- Logic operations: ANL, XRL, ORL
- Branch decisions: JB0, JB1, JB2, JB3, JNZ, JZ, JC, JNC, DSKZ, DSKNZ, SKB0, SKB1, SKB2, SKB3, JNB0, JNB1, JNB2, JNB3, SKNB0, SKNB1, SKNB2, SKNB3
- Shift operations: SHRC, RRC, SHLC, RLC,
- Binary additions/subtractions: ADDC, ADD, ADDU, SUB, SUBB, DEC, INC

After any of the above instructions are executed, the status of the carry flag (CF) and zero signal (ZF) will be influenced. The CF will be stored to internal register, read out by executing MOVA R, CF or MOV CF, R.

Carrie Flag Register (CF with SR=20H)

The CF register is only stored the CF signal state. Please refer to instruction sets to know CF signal status.

	3	2	1	0
CF	X	X	X	CF

Clock Generator

The W533X2 provides two oscillation circuits- main-oscillator (FM) and sub-oscillator (FS). The SCR (System Control Register) uses to select clock operation condition. Either main-oscillator or sub-clock can be the system clock (FOSC) by F32IN option bit (bit 0 of SCR special register) . The main-oscillator starts oscillation if FMEN (bit 1 of SCR) is written to 1. Main-oscillator can select crystal or RC oscillation by special register FMRCB bit (bit 2 of SCR) through external connections. If a crystal oscillator is used, a crystal or a ceramic resonator must be connected between XIN and XOUT, and a capacitor must be connected if an accurate frequency is needed. The oscillator is range form 400 KHz to 4 MHz. A 455 KHz ceramic resonator can be selected if a IR carrier output from RE3/TONE is needed. If the RC oscillator is used, a resistor must be connected between XIN and VDD. The sub-oscillator must be connected to a 32.768 KHz crystal between X32I and X32O. The connection is shown in Figure 3. One machine cycle consists of a four-state system clock sequence and can run up to 1 μS with a 4 MHz system clock.

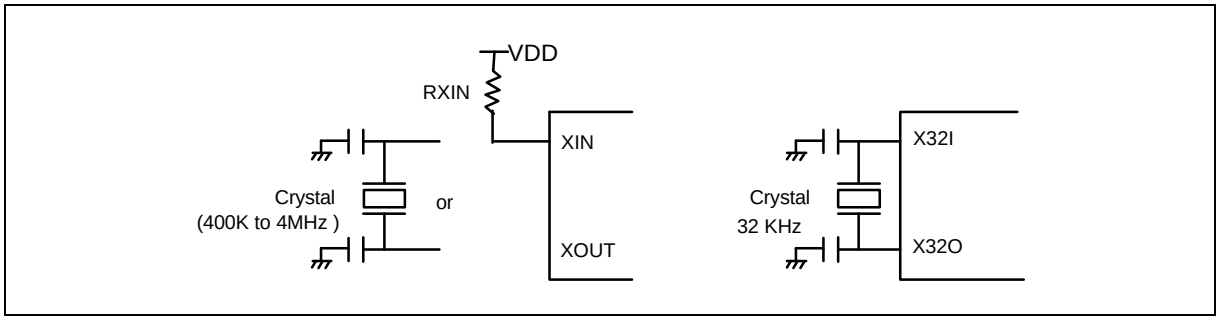


Figure 3. Oscillator Configuration

Dual-clock operation

This operation mode is dual-clock mode while FMEN bit is enable, and LCD operation clock source should be the sub-oscillator clock (32768 Hz) only. Sub-clock is used as system clock in initial reset such power on or reset pin active because SCR special register is 0001B. Programmer firstly needs to write F32IN suitable value at program start to change system clock to main--clock if high frequency clock is needed.

The exchange of the main-clock and sub-clock operation is performed by resetting or setting F32IN. If he F32IN is reset to 0, the clock source of the system clock generator is the main-oscillator clock; if the F32IN is set to 1, the clock source of the system clock generator is the sub-oscillator clock. The main-oscillator can stop oscillating when FMEN is reset to 0.

When the SCR is set or reset, we must pay attention to the following:

1. XX10B → XX01B: Disable the main-oscillator (FM) should not be done simultaneously with changing the system clock source(FOSC) from FM to FS. The FOSC should be changed first from FM to FS before the main-oscillator (FM) is disable. The correct seqence is: XX10B→XX11B→XX01B.

2. XX01B → XX10B: Enabling the main-oscillator (FM) should not be done simultaneously with changing from FS into FM. The main-oscillator (FM) should be enabled first before a delay subroutine is called to allow the main-oscillator to oscillate stably. The FOSC can now be changed from FS into FM. The correct sequence is therefore XX01B→XX11B→delay subroutine→XX10B. The suggested delay for Fm is 3.5 mS for 455 KHz ceramic resonator and 0.8 mS for 4 MHz crystal.

We must remember that the XX00B state which FM is stopped and FOSC is come from FM is inhibitive, because it will induce a system shutdown. The organization of the dual-clock operation mode is shown in below.

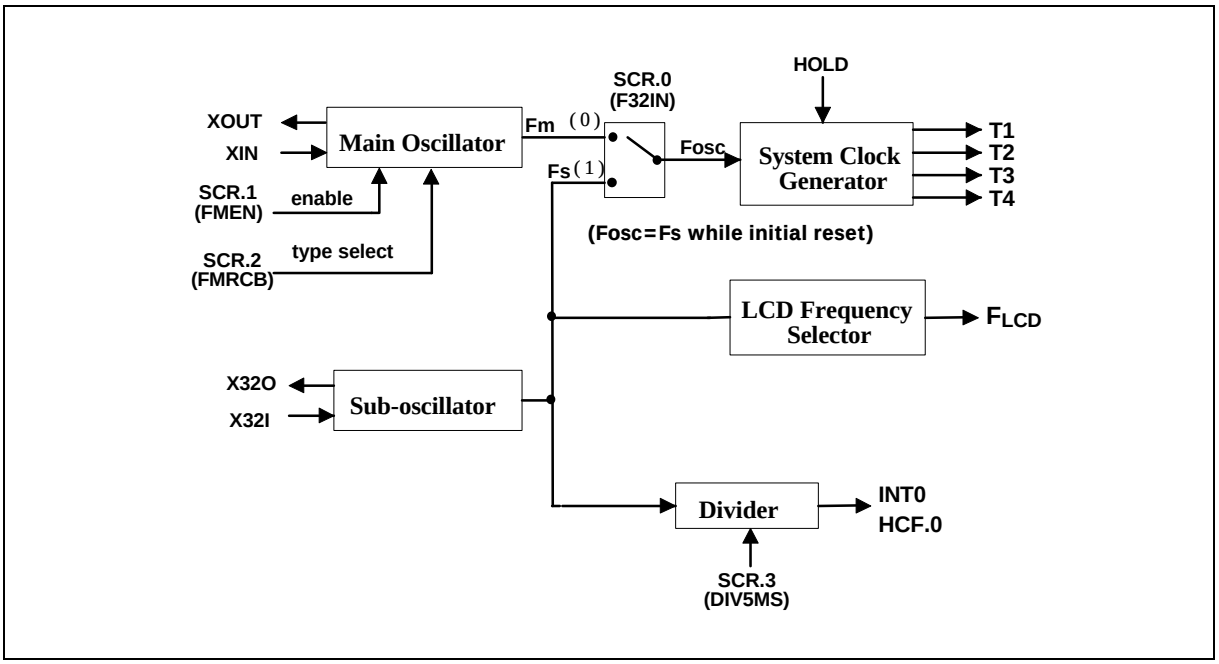


Figure 4. The Dual Clock Operation Mode Control Diagram

System Control Register (SCR with SR=2BH)

The SCR register is organized as 4 bit register SCR.3~SCR.0. Tha function of bit assignment is shown as following.

	3	2	1	0
SCR	DIV5MB	FMRCB	FMEN	F32IN

F32IN =0: F_M is used as F_{OSC} input
=1: F_S is used as F_{OSC} input
FMEN =0: F_M oscillation is disable
=1: F_M oscillation is enable
FMRCB =0: F_M type is RC oscillation
=1: F_M type is XTAL oscillation
DIV5MB =0 : Divider per 0.5sec will be overflow periodically
=1: Divider per 0.125sec will be overflow periodically.
All bit are possible to read/write, set/clear by user. At initial reset, the SCR is 0001B.

Divider

There is one divider as 14-bit/12bit binary up-counter designed to generate periodic interrupts. The divider is incremented by each clock (Fs). When an overflow is occurred, the divider event flag is set to 1 (EVF.0 = 1). The interrupt is executed if the divider interrupt enable flag has been set (IEF.0 = 1), or the hold state is terminated if the hold release enable flag has been set (HEF.0 = 1). There are two time periods (500mS & 125 mS) that can be selected by DIV5MB bit. When DIV5MB is reset to 0 (default), the 500 mS period time is selected; others DIV5MB is set to 1 to select 125 mS.

Watchdog Timer (WDT)

The watchdog timer (WDT) is used to prevent the program from unknown errors. The WDT function can be enable by mask option and the clock source is Fosc/1024 or Fosc/16384 by WDTCK (bit 3 of MR1 special register) . At initial reset, the WDTCK is come from F_{OSC}/1024. The WDT overflows is occurred while chip operation is not under control and will be reset. The contents of the WDT can be reset by the instruction CLR FLAG1, #0010B (CLR WDT). The input clock of

the WDT can be switched to Fosc/16384 (or Fosc/1024) while WDTCK is written 1 (or 0). In normal operation, the application program must reset WDT (by CLR WDT) before it overflows. The WDT minimun overflow period is 500mS when the system clock (Fosc) is 32 KHz and WDT clock input is Fosc/1024. The organization of watchdog timer is shown in Figure 5

FLAG1 Register (FLAG1 with SR=23H)

Divider and watchdog counter can be reset by CLR FLAG1, #I instruction. Both CLR DIV and CLR WDT instructions can be use to clear DIVR bit and WDTR bit separately. The bit descriptions are as following.

	3	2	1	0
FLAG1	X	DIVR	WDTR	X

DIVR =0 no influence
=1 Divider counter is clear
WDTR=0 no influence
=1 Watchdog timer is clear
X means don't care value
All bit can be cleared only. At initial reset, FLAG1 is 0000B.

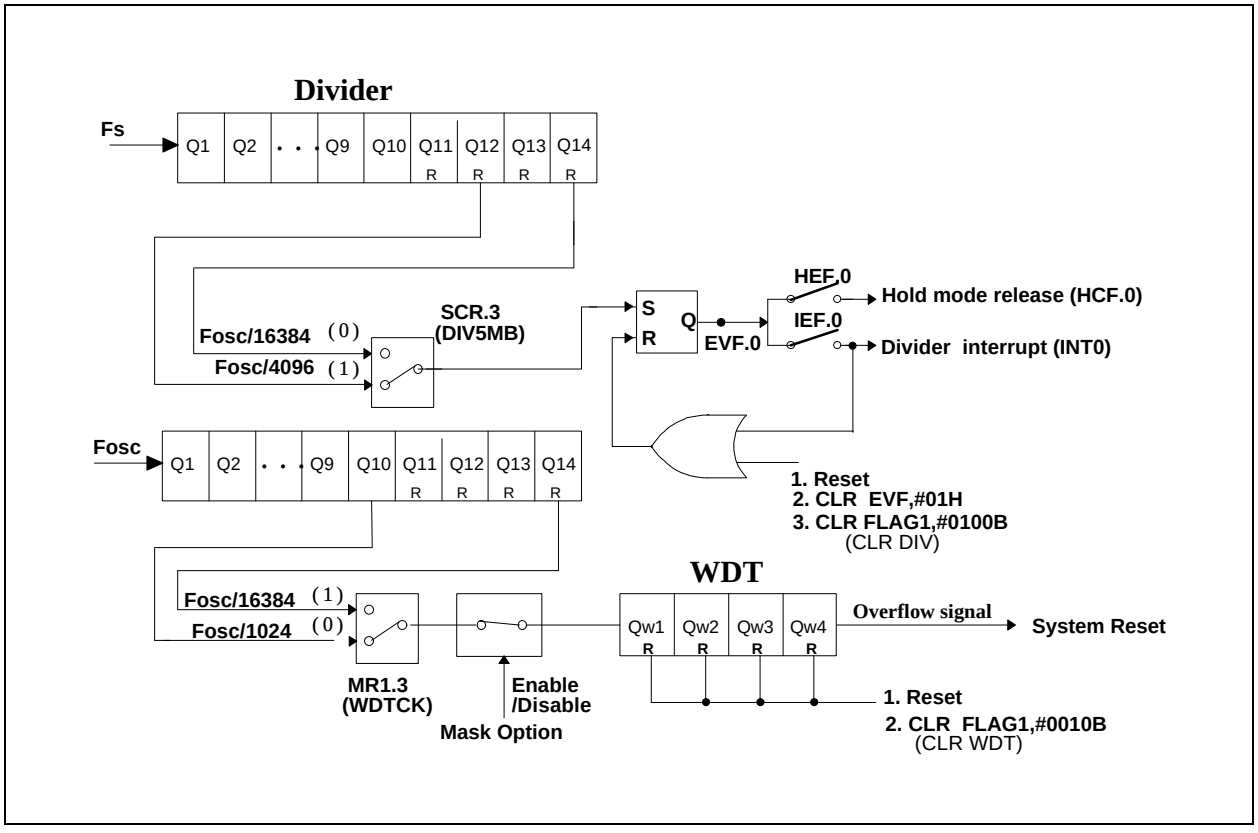


Figure 5. Organization of Divider and Watchdog Timer

Timer/Counter

1. Timer 0 (TM0)

Timer 0 (TM0) is a programmable 8-bit binary down-counter. The specified value can be loaded into TM0 by executing the

MOV TM0L(TM0H),R instructions. To execute MOV TM0L(TM0H),R instructions will stop TM0 down-counting if the TM0 is processing down-counting, reset TM0EN option bit (bit 3 of MR0 special register) to 0, and load specified value to TM0. When TM0EN is set to 1, the event flag 1 (EVF.1) is reset and the TM0 starts to count. Timer 0 stops operating and generates an underflow (EVF.1 = 1) while it decrements to FFH. The interrupt is executed if the Timer 0 interrupt enable flag has been set (IEF.1 = 1); and the hold state is terminated if the hold release enable flag 1 has been set (HEF.1 = 1). The Timer 0 clock input can select either FOSC/1024 or FOSC/4 by setting TM1CK (bit 2 of MR1 special register) to 1 or resetting TM1CK to 0. The organization of Timer 0 is shown in Figure 6.

Example:

If the Timer 0 clock input is FOSC/4, then:

Desired Time 0 interval = (preset value +1) × 4 × 1/FOSC

If the Timer 0 clock input is FOSC/1024, then:

Desired Time 0 interval = (preset value +1) × 1024 × 1/FOSC

Preset value: Decimal number of Timer 0 preset value

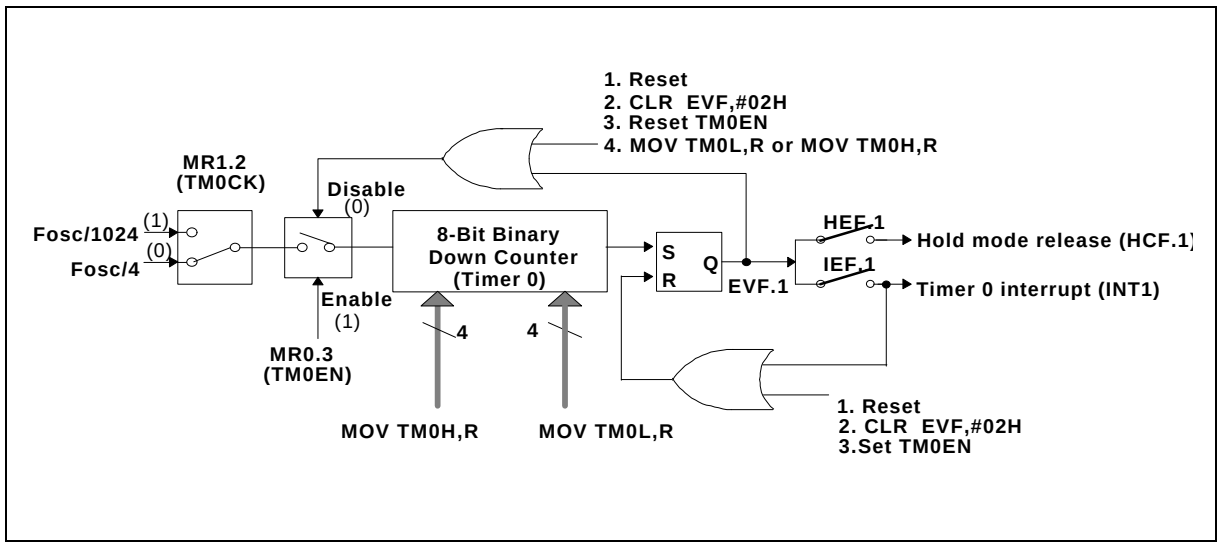


Figure 6. Organization of Timer 0

2. Timer 1 (TM1)

Timer 1 (TM1) is also a programmable 8-bit binary down counter, as shown in Figure 7. Timer 1 can be used as a counter to count external events or to output an arbitrary frequency to the RE3/TONE pin. The input clock source of Timer 1 can be internal or sub-frequency/4 (32768/4) Hz clock by TM1SR option bit (bit 1 of MR1 special register). The internal clock can be selected FOSC/64 or FOSC by TM1CK option bit (bit 0 of MR1 special register) At initial reset, the Timer 1 clock input is FOSC. If an external clock is selected as the clock source of Timer 1, the content of Timer 1 is decreased by 1 at the falling edge of RC.0. To execute MOV TM1L, R or MOV TM1H,R instruction will load specified data to the auto-reload buffer and disable TM1 down-counting (i.e. TM1EN is reset to 0). If TM1EN is set 1 , the contents of the auto-reload buffer will be loaded into the TM1 down counter to start counting and reset the event flag 7 (EVF.7 = 0). When the timer decrements to FFH, it will generate an underflow (EVF.7 = 1) and be auto-reloaded with the specified data, after which it will continue to count down. An interrupt is executed if the interrupt enable flag 7 has been set to 1 (IEF.7 = 1), and the hold state is terminated if the hold mode release enable flag 7 is set to 1 (HEF.7 = 1).

The specified frequency of Timer 1 can also be output to the RE3/TONE pin by TONE option bit(bit 0 of MR0).

Example:

If the Timer 1 clock input is FT, then:

Desired Timer 1 interval = (preset value +1) / FT

Desired frequency for RE3/TONE output pin = FT ÷ (preset value + 1) ÷ 2 (Hz)

Preset value: Decimal number of Timer 1 preset value

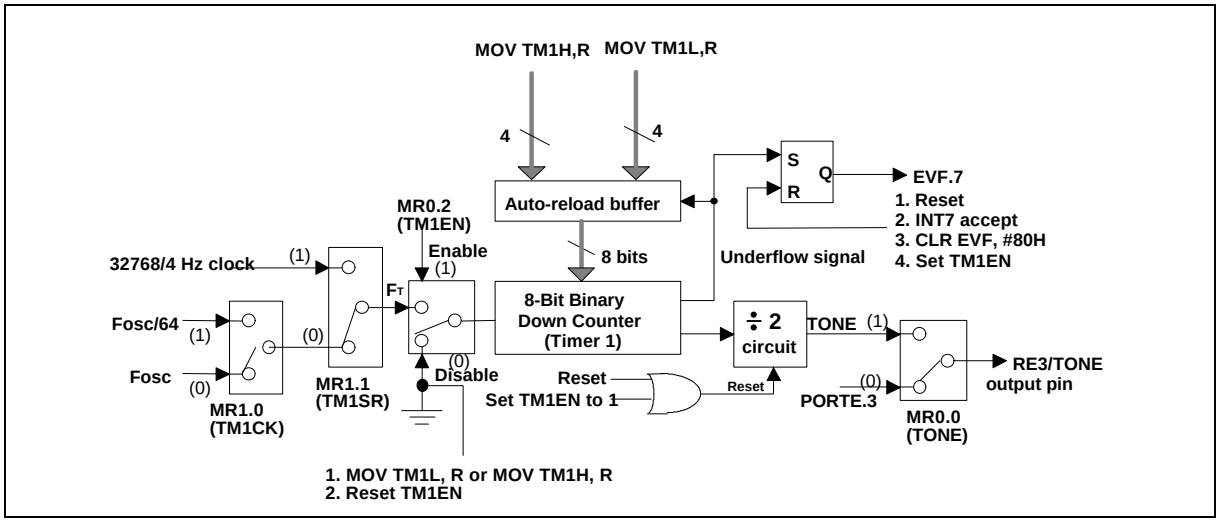


Figure 7. Organization of Timer 1

For example, when FT equals 32768 Hz, depending on the preset value of TM1, the RE3/TONE pin will output a single tone signal in the tone frequency range from 64 Hz to 16384 Hz. The relation between the tone frequency and the preset value of TM1 is shown in the Table 3.

Mode Register 0 (MR0 with SR=28H)

Mode Register 0 is organized as a 4-bit binary register (MR0.0 to MR0.3) . The bit descriptions are as following: (Initial value=0000B)

	3	2	1	0
MR0	TM0EN	TM1EN	LCDEN	TONE

TONE = 0 RE3 as the data output of PORTE.3.
= 1 RE3 will be as TONE signal output generated from Timer 1
LCDEN =0 LCD display OFF
=1 LCD display ON
TM1EN =0 Timer 1 counting is disable
=1 Timer 1 counting is enable
TM0EN=0 Timer 0 counting is disable
=1 Timer 0 counting is enable
User can read/write and set/clear all bits. At initial reset, MR0 is 0000B.

Mode 1 Register (MR1 with SR=29H)

Mode Register 1 is organized as a 4-bit binary register (MR1.0 to MR1.3) . The bit descriptions are as following: (Initial value=0000B)

	3	2	1	0
MR1	WDTCK	TM0CK	TM1SR	TM1CK

TM1CK= 0 The internal Timer 1 clock rate is FOSC.
= 1 The internal Timer1 clock rate is FOSC/64.
TM1SR=0 The Timer 1 with internal clock source (depened on TM1CK)
=1 The Timer 1 with sub-frequency/4 (32768/4) clock source
TM0CK= 0 The internal Timer 0 clock rate is Fosc/4

= 1 The internal Timer0 clock rate is Fosc/1024
WDTCK= 0 The watchdog timer clock rate is Fosc/1K
= 1 The watchdog timer clock rate is Fosc/16K
User can read/write and set/clear all bits. At initial reset, MR1 is 0000B.

		3			4			5		
		Tone frequency	TM1 preset value & MFP frequency		Tone frequency	TM1 preset value & MFP frequency		Tone frequency	TM1 preset value & MFP frequency	
TONE	C	130.81	7CH	131.07	261.63	3EH	260.06	523.25	1EH	528.51
	C #	138.59	75H	138.84	277.18	3AH	277.69	554.37	1CH	564.96
	D	146.83	6FH	146.28	293.66	37H	292.57	587.33	1BH	585.14
	D #	155.56	68H	156.03	311.13	34H	309.13	622.25	19H	630.15
	E	164.81	62H	165.49	329.63	31H	327.68	659.26	18H	655.36
	F	174.61	5DH	174.30	349.23	2EH	372.36	698.46	16H	712.34
	F #	185.00	58H	184.09	369.99	2BH	390.09	739.99	15H	744.72
	G	196.00	53H	195.04	392.00	29H	420.10	783.99	14H	780.19
	G #	207.65	4EH	207.39	415.30	26H	443.81	830.61	13H	819.20
	A	220.00	49H	221.40	**440.00**	**24H**	**442.81**	880.00	12H	862.84
	A #	233.08	45H	234.05	466.16	22H	468.11	932.23	11H	910.22
	B	246.94	41H	248.24	493.88	20H	496.48	987.77	10H	963.76

Table 3: TONE output with central tone A4(440HZ)

Mode Register 3 (MR3 with SR=27H)

Mode Register 3 is organized as a 4-bit binary register (MR3.3 to MR3.0) . The bit descriptions are as following: (Initial value=0000B)

	3	2	1	0
MR3	VLCD	FsENB	P1	P0

VLCD = 0 Use internal LCD supplying voltage generated by pump circuit.
= 1 Use external LCD supplying voltage.
FsENB = 0 Enable 32768 Hz crystal
=1 Disable 32768 Hz crystal
P1 = PWM volumn control bit 1
P0 = PWM volumn control bit 0

Note that any one pin of RC port in low state will force the bit MR3.2 low. It means once any one pin of RC port in low state, the setting action for this bit is invalid.

Interrupts

The W533X2 provides five internal interrupt sources (Divider, TM0, SPEECH, MELODY and TM1) and two external interrupt source (port RC and port RD). Vector addresses for each of the interrupts are located in the range of program memory (ROM) addresses 004H to 020H. The flags IEF, PEF, and EVF are used to control the interrupts. When EVF is set to "1" by hardware and the corresponding bits of IEF and PEF have been set by software, an interrupt is generated. When an interrupt occurs, all of the interrupts are inhibited until the EN INT or MOV IEF,#I instruction is invoked. The

interrupts can also be disabled by executing the DIS INT instruction. When an interrupt is generated in hold mode, the hold mode will be released momentarily and interrupt subroutine will be executed. After the RTN instruction is executed in an interrupt subroutine, the μC will enter hold mode again. The control circuit diagram and operation flow chart are shown in Figure 8, and Figure 9 separately.

Mode Register 2 (MR2 with SR=26H)

Mode Register 2 is organized as a 1-bit only register . This INTEN bit uses to disable/enable interrupt function. Instruction of DIS EN uses to reset INTEN bit logic 0, and EN INT set INTEN bit to 1.

	3	2	1	0
MR2	X	X	X	INTEN

INTEN = 0 Disable any interrupt process.

= 1 Enable interrupt process which IEF.n is set by 1.

X means do't care.

User can read/write and set/clear INTEN. At initial reset, MR2 is 0001B.

Interrupt Enable Flag Register (IEF with SRP=07H)

The interrupt enable flag is organized as a 8-bit binary register (IEF.0 to IEF.7) that IEFL and IEFH registers store IEF.0~IEF.3 and IEF.4~IEF.7 separately. These bits are used to control the interrupt conditions. It is controlled by the MOV IEF, #I instruction with 8 bit immediate data. Of course, MOV IEFH, #I and MOV IEFL, #I instructions can be used with 4 bit immediate data. When one of these interrupts is accepted, the corresponding to the bit of the event flag will be reset by hardware, but the other bits are unaffected. In interrupt subroutine, these interrupts will be disable till the instruction MOV IEF, #I or EN INT is executed again. Therefore, to enable these interrupts, the instructions MOV IEF, #I or EN INT must be executed again. Otherwise, these interrupts can be disable by executing DIS INT instruction. The bit descriptions are as follows:

	7	6	5	4	3	2	1	0
IEF	TM1	Melody	Speech	X	RD	RC	TM0	DIV

IEFH: 7 6 5 4 — IEFL: 3 2 1 0

IEF.0 = 1 Interrupt 0 is accepted by overflow from the Divider

IEF.1 = 1 Interrupt 1 is accepted by underflow from the Timer 0.

IEF.2 = 1 Interrupt 2 is accepted by a signal change on port RC.

IEF.3 = 1 Interrupt 3 is accepted by a signal change on port RD

IEF.4 Reserved

IEF.5 = 1 Interrupt 5 is accepted by speech play ending with SPC_busy falling edge

IEF.6 = 1 Interrupt 6 is accepted by melody play ending with MLD_busy falling edge

IEF.7 = 1 Interrupt 7 is accepted by underflow from Timer 1.

All bits can be read/write and set/clear by user.

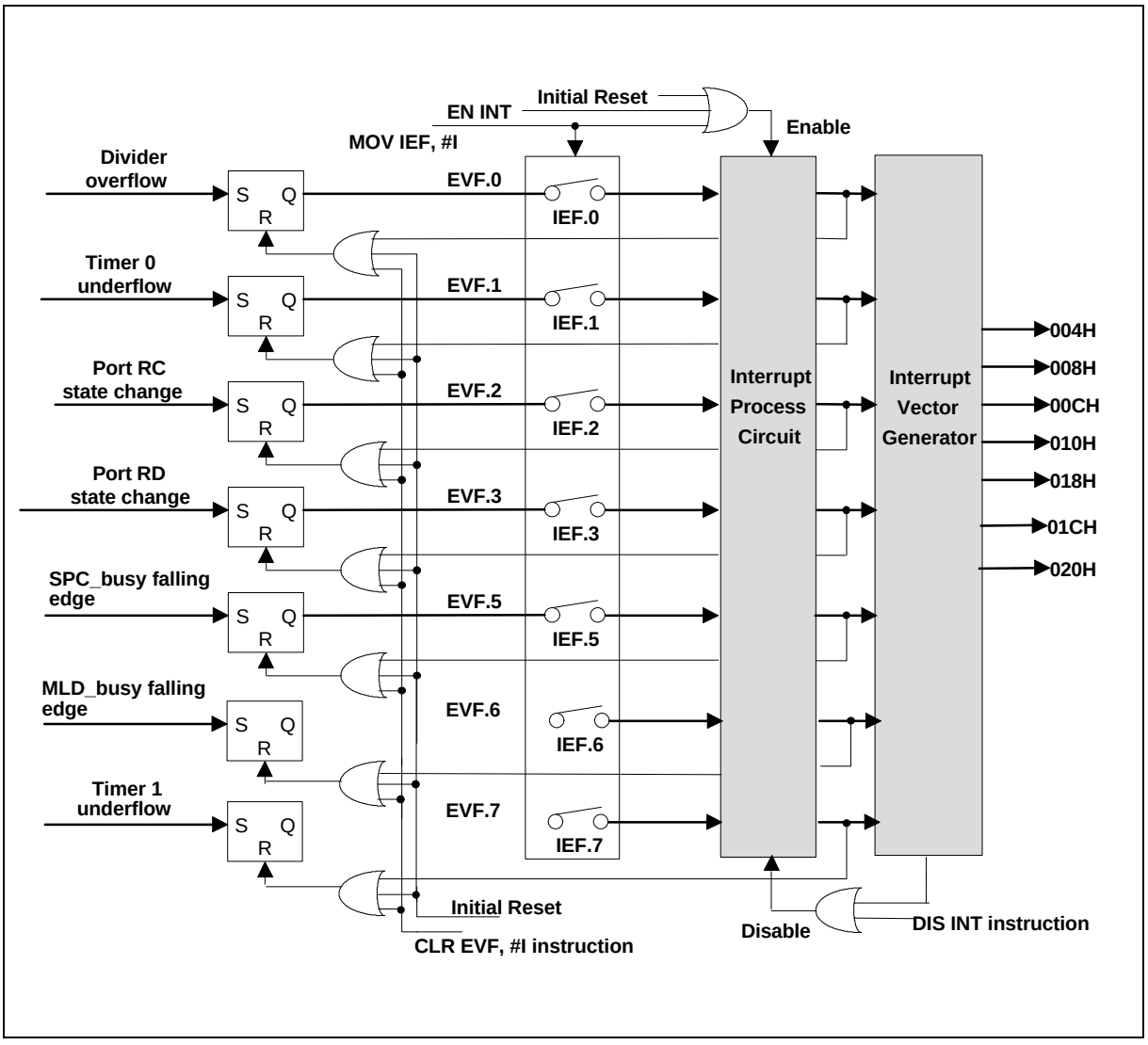


Figure 8. Interrupt Event Control Diagram

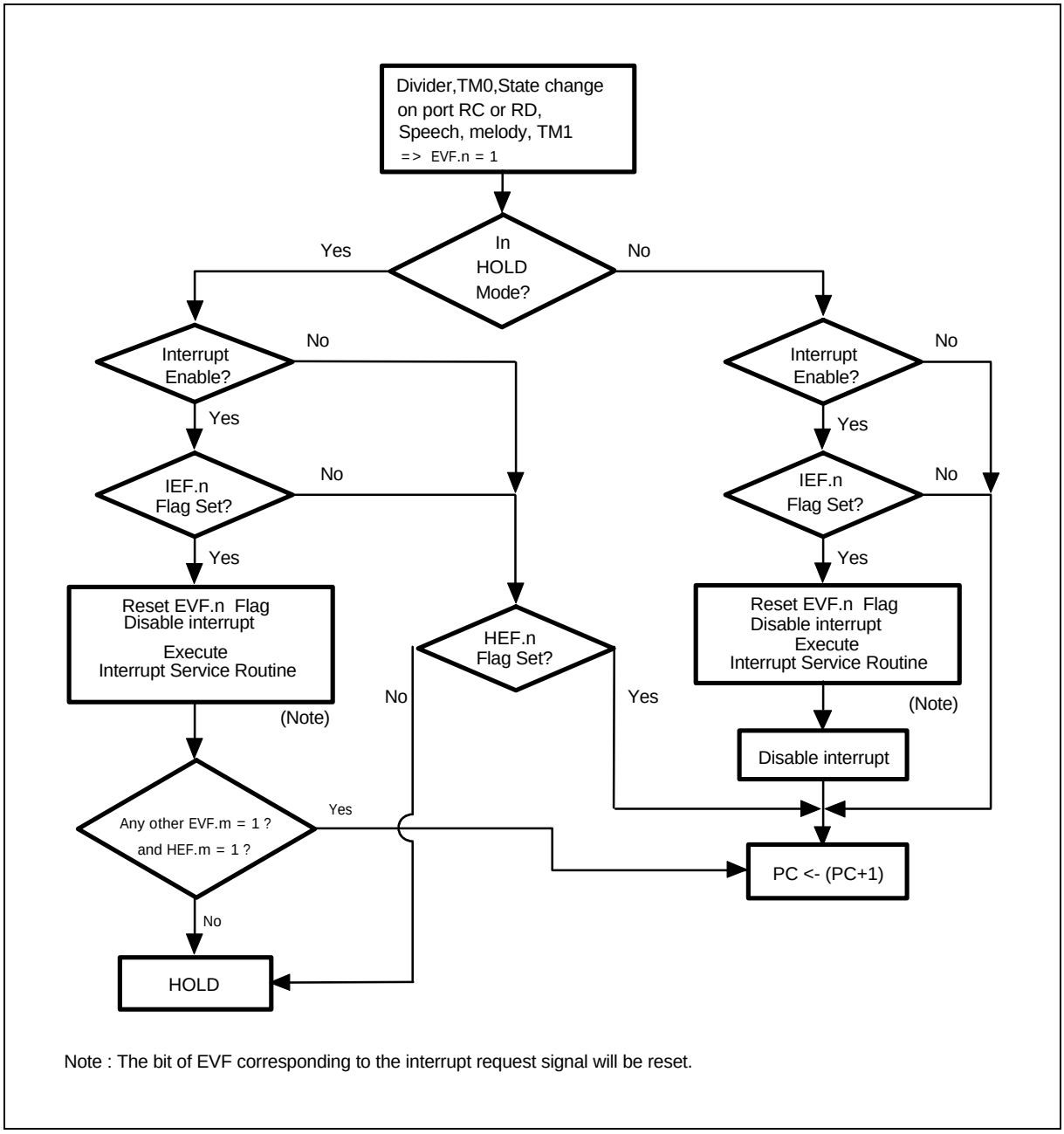


Figure 9. Hold Mode and Interrupt Operation Flow Chart

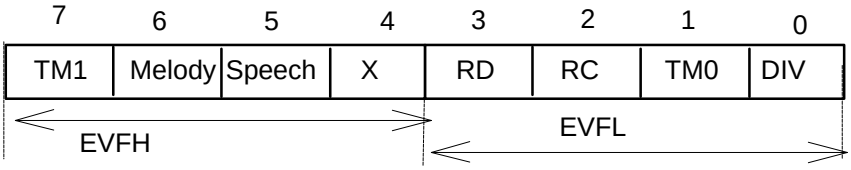
Hold Mode Operation

All operations of the μC cease in hold mode except for oscillator , timer, divider and LCD driver . The μC enters hold mode while the HOLD instruction is executed. The hold mode can be released by one of seven ways which are timer 0 underflow, timer 1 underflow, divider overflow, speech playing finished, melody playing finished, RC port pin state changed and RD port pin state changed. Before the device enters the hold mode, the HEF, PEF, and IEF flags must be set to define the hold mode release conditions. For more details, refer to the instruction sets and Figure 9.

Event Flag Register (EVF with SRP=05H)

The event flag is organized as an 8-bit binary EVF0 to EVF7 that EVFL and EVFH registers store EVF.0 ~ EVF.3 and

EVF.4 ~ EVF.7 separately. It is set by hardware and reset by CLR EVFL,#I and MOV EVFH,#I instruction or the occurrence of an interrupt. The bit descriptions are as follows:



EVF.0 = 1 Overflow from Divider occurred.

EVF.1 = 1 Underflow from Timer 0 occurred.

EVF.2 = 1 Statel change on port RC occurred.

EVF.3 = 1 State change on port RD occurred.

EVF.4 Reserved

EVF.5 = 1 Speech play ending with SPC_busy flag falling edge occurred.

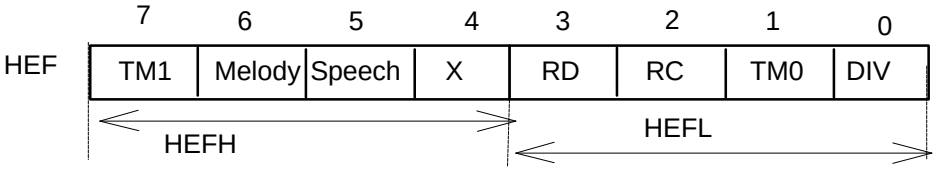
EVF.6 = 1 Speech play ending with SPC_busy flag falling edge occurred.

EVF.7 = 1 Underflow from Timer 1 occurred.

All bits can be read and clear only by user.

Hold Mode Release Enable Flag Register (HEF with SRP=06H)

The hold mode release enable flag is organized as an 8-bit binary register (HEF.0 to HEF.7) that HEFL and HEFH register store HEF.0~HEF.3 and HEF.4~ HEF.7 separately. The HEF is used to control the hold mode release conditions. It is controlled by the MOV HEF, #I instruction with 8 bit immediate data , or MOV HEFH,#I and MOV HEFL,#I with 4 bit immediate data.. The bit descriptions are as follows:



HEF.0 = 1 Overflow from the Divider causes hold mode to be released.

HEF.1 = 1 Underflow from Timer 0 causes hold mode to be released.

HEF.2 = 1 Statel change on port RC causes hold mode to be released.

HEF.3 =1 Statel change on port RD causes hold mode to be released

HEF.5 = 1 Speech play ending with SPC_busy flag falling edge causes hold mode to be released

HEF.6 =1 Melody play ending with MLD_busy flag falling edge causes hold mode to be released

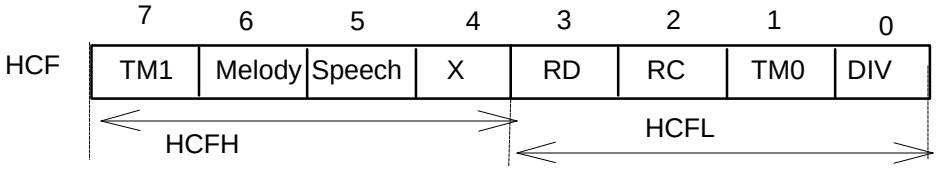
HEF.7 = 1 Underflow from Timer 1 causes hold mode to be released.

All bits can be read/write and set/clear by user

Hold mode release Condition Flag Register (HCFL, HCFH with SR=10H & 11H)

The hold mode release condition flag is organized as a 8-bit binary register (HCF0 to HCF7) that HCFL and HCFH registers store HCF.0~HCF.3 and HCF.4~HCF.7 separately. The hold mode has been released, and is loaded by

hardware. The HCF can be read out by the MOVA R, HCFL and MOVA R, HCFH instructions. When any of the HCF bits is "1," the hold mode will be released and the HOLD instruction is invalid. The HCF can be reset by the CLR EVFL/EVFH,#I (EVF.n = 0) When EVF.n or HEF.n have been reset, the corresponding bit of HCF is reset simultaneously by hardware. The bit descriptions are as follows:



HCF.0 = 1 Hold mode was released by overflow from the Divider.

HCF.1 = 1 Hold mode was released by underflow from the Timer 0.

HCF.2 = 1 Hold mode was released by a state change on port RC

HCF.3 = 1 Hold mode was released by a state change on port RD

HCF.4 reserved

HCF.5 = 1 Hold mode was released by speech play ending with SPC_busy falling edge

HCF.6 = 1 Hold mode was released by melody play ending with MLD_busy falling edge

HCF.7 = 1 Hold mode was released by underflow from the Timer 1

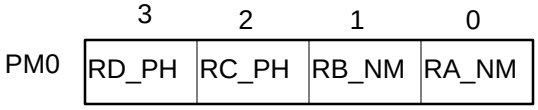
All bits are read only by user and set/clear by chip hardware.

Input/Output Ports RA, RB

Port RA consists of pins RA0 to RA3 and port RB consists of pins RB0 to RB3. At initial reset, input/output ports RA and RB are both in input mode. When RA and RB are used as output ports, CMOS or NMOS open drain output type can be selected by the PM0 special register. Each pin of port RA or RB can be specified as input or output mode independently by the PM1 and PM2 special registers. The MOVA R, PORTA or MOVA R, RORTB instructions operate the input functions and the MOV PORTA, R or MOV PORTB, R operate the output functions. For more details, refer to the instruction table and Figure 10

Port Mode 0 Register (PM0 with SR=32H)

The port mode 0 register is organized as a 4-bit binary register (PM0.0 to PM0.3). PM0 can be used to determine the structure of the input/output ports; it is controlled by the MOV PM0, #I instruction. The bit descriptions are as follows:



RA_NM = 0 RA port is CMOS output type.
= 1 RA port is NMOS open drain output type.

RB_NM = 0 RB port is CMOS output type.
= 1 RB port is NMOS open drain output type.

RC_PH = 0 RC port pull-high resistor is disabled.
= 1 RC port pull-high resistor is enabled.

RD_PH = 0 RD port pull-high resistor is disabled.
= 1 RD port pull-high resistor is enabled.

All bit can be read/write and set/clear by user. At initial reset, PM0 is equal to "0000" that port RA , RB are CMOS type input mode , and port RC, RD are disable pull-high resistor.

Port Mode Register 1 and 2 (PM1, PM2 with SR=36H, 37H)

The port mode 1, 2 registers are organized as a 4-bit binary PM1.0 to PM1.3 and PM2.0~PM2.3. PM1 (PM2) can be used to control the input/output mode of port RA (RB) . PM1 (PM2) is controlled by the MOV PM1, #I (MOV PM2, #I) instruction. The bit descriptions are as follows:

	3	2	1	0
PM1	RA3IN	RA2IN	RA1IN	RA0IN

RA0IN = 0 RB0 works as output pin;
= 1 RB.0 works as input pin
RA1IN = 0 RB1 works as output pin;
= 1 RB.1 works as input pin
RA2IN = 0 RB2 works as output pin;
=1 RB.2 works as input pin
RA3IN = 0 RB3 works as output pin;
= 1 RB.3 works as input pin

	3	2	1	0
PM2	RB3IN	RB2IN	RB1IN	RB0IN

RB0IN = 0 RB0 works as output pin;
= 1 RB.0 works as input pin
RB1IN = 0 RB1 works as output pin;
= 1 RB.1 works as input pin
RB2IN = 0 RB2 works as output pin;
=1 RB.2 works as input pin
RB3IN = 0 RB3 works as output pin;
= 1 RB.3 works as input pin

All bit can be read/write and set/clear by user. At initial reset, port RA, RB is input mode (PM1, PM2 = 1111B).

Port A Register (PORTA with SR=38H)

This register stores the current port RA pin state by MOV PORTA, R and MOV R, PORTA instructions. When port A is input, the register is read only. Otherwise PORTA is written during port RA output mode.

	3	2	1	0
PORTA	PA3	PA2	PA1	PA0

Port B Register (PORTB with SR=39H)

This register stores the current port RB pin state by MOV PORTB, R and MOV R, PORTB instructions. When port RB is input, the register is read only. Otherwise PORTB is writen during port RB output mode.

	3	2	1	0
PORTB	PB3	PB2	PB1	PB0

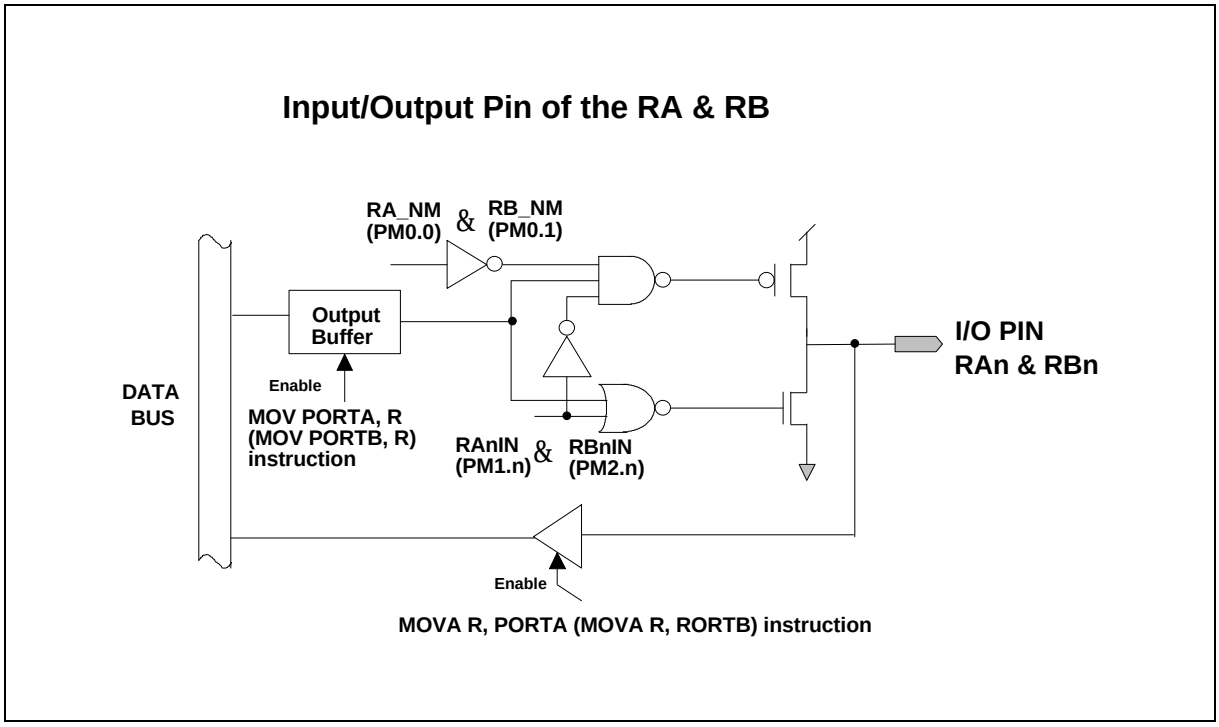


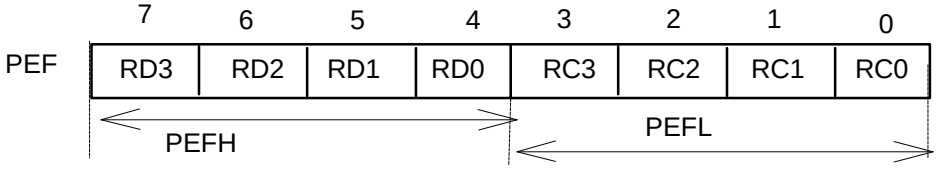
Figure 10. Architecture of RA (RB) Input/Output Pins

Input Ports RC, RD

Port RC consists of pins RC0 to RC3, and port RD consists of pins RD0 to RD3. Each pin of port RC and port RD can be connected to a pull-up resistor, which is controlled by the port mode 0 register (PM0). When the PEF, HEF, and IEF corresponding to the RC (RD) port are set, a statel change at the specified pins of port RC (RD) will execute the hold mode release or interrupt subroutine. Port status register 0 and 1 (PSR0, PSR1) record the signal changing status on the port RC and RD. PSR0(PSR1) can be read out and cleared by the MOVA R, PSR0(MOVA R, PSR0) and CLR PSR0 (CLR PSR1) instructions. Refer to Figure 11 and the instruction sets for more details.

Port Enable Flag for hold mode (PEFL, PEFH with SR=14H, 15H)

The port enable flag is organized as 8-bit binary register (PEF.0 to PEF.7) that PEFL and PEFH registers store PEF.0~PEF.3 and PEF.4~PEF.7 separately. PEFL controls port RC status, and PEFH is responsible port RD status. Before port RC, RD may be used to release the hold mode or preform interrupt function, the content of the PEF must be set first. If PEF is wirtten to "1", the function will be enable. The PEF is controlled by the MOV PEF, #I instruction with 8 bit immediate data. Both MOV PEFH,#I and MOV PEFH,#I can also be used with 4 bit immediate data. The bit descriptions are as follows:



PEF.0 =1 : State change on pin RC0 to release hold mode or perform interrupt

PEF.1 =1 : State change on pin RC1 to release hold mode or perform interrupt

PEF.2 =1 : State change on pin RC2 to release hold mode or perform interrupt

PEF.3 =1 : State change on pin RC31 to release hold mode or perform interrupt

PEF.4 =1 : State change on pin RD0 to release hold mode or perform interrupt

PEF.5 =1 : State change on pin RD1 to release hold mode or perform interrupt

PEF.6 =1 : State change on pin RD2 to release hold mode or perform interrupt

PEF.7 =1 : State change on pin RD3 to release hold mode or perform interrupt

All bit can be read/write and set/clear by user.

Port Status Register 0 and 1 (PSR0, PSR1 with 34H, 35H)

Port status register 0 and 1 are organized as 4-bit binary PSR0.0 to PSR0.3 and PSR1.0 to PSR1.3. PSR0 (PSR1) will have the chance to be set to "1" if the PEF.n is enable and RCn (RDn) input state is changed. Then hold mode or interupt will be occurred. Refer to Figure 10. PSR0 (PSR1) can be read or cleared by the MOVA R, PSR0 (MOVA R, PSR0), and CLR PSR0(CLR PSR0) instructions. The bit descriptions are as follows:

	3	2	1	0
PSR0	RC3EG	RC2EG	RC1EG	RC0EG

Bit 0 = 1 : RC0 input signal state is changed
= 0 : RC0 input signal state isn't changed
Bit 1 = 1 : RC1 input signal state is changed
= 0 : RC1 input signal state isn't changed
Bit 2 = 1 : RC2 input signal state is changed
= 0 : RC2 input signal state isn't changed
Bit 3 = 1 : RC3 input signal state is changed
= 0 : RC3 input signal state isn't changed

All bit can be read only, and clear 4bit simultaneously. At initial reset , PSR1 is 0000B

	3	2	1	0
PSR1	RD3EG	RD2EG	RD1EG	RD0EG

Bit 0 = 1 : RD0 input signal state is changed
= 0 : RD0 input signal state isn't changed
Bit 1 = 1 : RD1 input signal state is changed
= 0 : RD1 input signal state isn't changed
Bit 2 = 1 : RD2 input signal state is changed
= 0 : RD2 input signal state isn't changed
Bit 3 = 1 : RD3 input signal state is changed
= 0 : RD3 input signal state isn't changed

All bit can be read only, and clear 4bit simultaneously. At initial reset , PSR1 is 0000B

Port C Register (PORTC with SR=3AH)

This register stores the port RC current input state by MOV R, PORTC instructions.

	3	2	1	0
PORTC	PC3	PC2	PC1	PC0

Port D Register (PORTD with SR=3BH)

This register stores the port RD current input state by MOV R, PORTD instructions.

	3	2	1	0
PORTD	PD3	PD2	PD1	PD0

Output Port RE

When the MOV PORTE, R instruction is executed, the data in the RAM will be output to port RE . The RE3 pin can use to output TONE from Timer 1 if TONE option bit is set to 1.

Port E Register (PORTE with SR=3CH)

This register stores the current Port RE output state by MOV PORTE, R instructions.

	3	2	1	0
PORTE	PE3	PE2	PE1	PE0

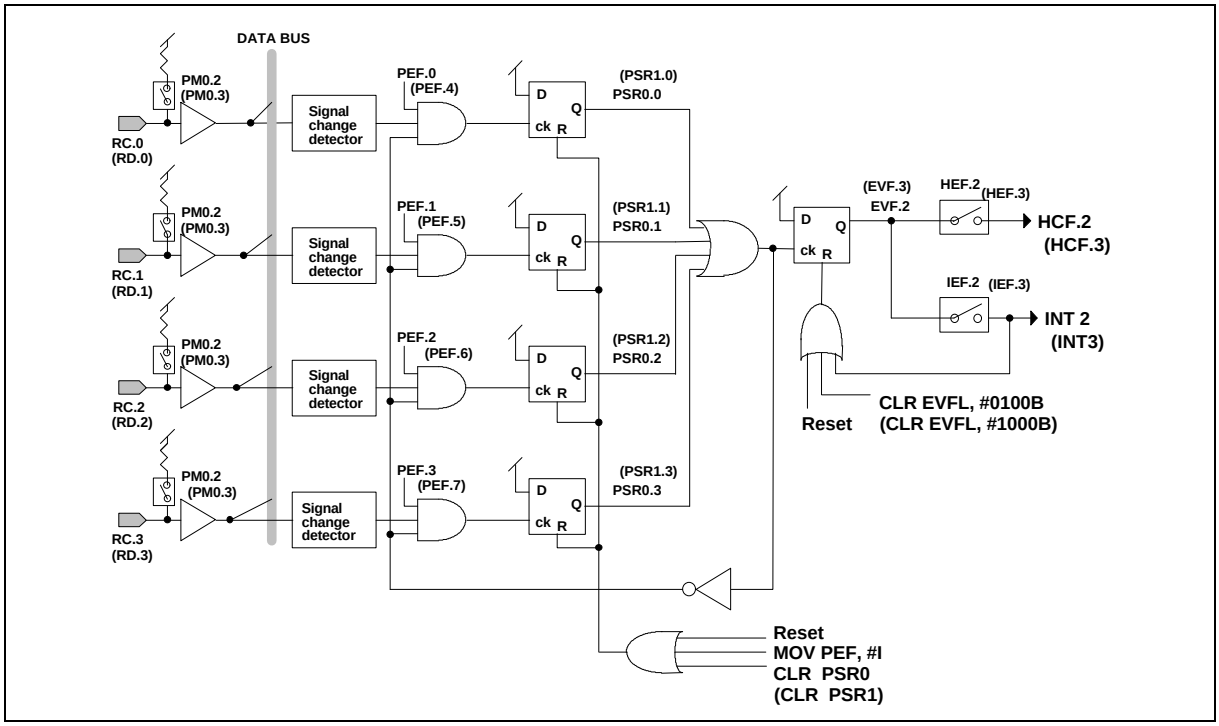


Figure 11. Architecture of Input Ports RC (RD)

Reset Function

The W533X2 is reset either by a power-on reset or $\overline{RES}$ active low pulse. The initial reset state of internal special register and Input/Output are shown as Table 4.

Program Counter (PC)	0000B
Input/output ports RA, RB	Input mode
Output port RE	0000B
RA & RB ports output type	CMOS type
RC & RD ports pull-high resistors	Disable
System Clock Input	Fs (32768HZ)
Timer 0 input clock	FOSC/4
Timer 1 input clock	FOSC
Input clock of the watchdog timer	FOSC/1024
LCD display	OFF
LCD Bias	1/7 bias
LCD Duty	1/32 duty
LCD Internal Pump Circuit	Enable
LCD Pump Voltage	Triple pump
SCR register	0001B
MR2 register (INTEN flag)	0001B
PM1,PM2 register	1111B
Others Registers	0000B

Table 4: Default value at initial Reset

PART B: SPEECH and MELODY FUNCTION

Both speech and melody use the same clock source from ROSC pin and these two functions can be played at the same time. When speech or melody is playing, the ROSC clock is enable, otherwise clock is disable for power saving. Either speech synthesiaer or melody sound tone can be output to PWM1 and PWM2 and direct driving speaker. Speech coding can select whether two LED output pin will be active. And LED1 can also use to active depended on melody output

volume.

FLAG0 Register (FLAG0 with SR=22H)

FLAG0 is organized as a 4-bit register and used to control the speech and melody synthesizers. FLAG0.1~0 are read/write and set/clear by user., but FLAG0.3 ~ FLAG0.2 are set/clear by chip hardware. At initial reset, FLAG0 is 0000B. The bit description are as following.

	3	2	1	0
FLAG0	MLD_busy	SPC_busy	MLD_play	SPC_play

SPC_paly =0 : Speech play is disable
=1 : Speech play is enable.
MLD_paly =0 : Melody play is disbale.
=1 : Melody play is enable.
SPC_busy =0 : Speech play is finished
=1 : Speech play is processing
MLD_busy =0 : Melody play is finished.
=1 : melody play is processing

SPEECH Function

There are 1.4M bits dedicated speech ROM for speech synthesizer, and can be sepatated as 255 sections different voice maximun by WINBOND ADPCM power speech coding system. Uc needs to write play section number in SPCH, SPCL , and set SPC_play option bit to "1" (bit 0 of FLAG0 special register) to play speech voice . Then SPC_busy bit (bit 2 of FLAG0) will be changed from low to high and keeps high till speech play is ending. If interrupt flag or hold mode flag IEF.5, HEF.5, HCF.5 are set, interrupt or hold mode release will be processed while SPC_busy falling edge occurred. The circuit structure is shown in Figure 12. SPC_play bit can be set to "1" again, after section number had been finished parallel to serial of previous SPC_play edge. There are minimun 8 instruction delay of two continuous SPC_play rising shown in Figure 12.
Two LED output with 3HZ frequency can be used to drive external LED during speech playing. The SPCH, SPCL will be latched during SPC_play risng edge. The speech synthesizer is disabled when MLD_busy bit (bit 3 of FLAG0) is 1, and so is the melody synthesizer when SPC_busy bit is 1.
The SPC_play is set to 1 to activate the speech synthesizer. The speech synthesizer receives the rising edge of SPC_play then plays the voice section pointed by SPC.7~SPC.0 and pull the voltage level of SPC_busy to logic 1. The SPC_busy is cleared by hardware when:
1. the speech synthesizer finishes its tasks and executes an END command;
2.the speech synthesizer receives a rising edge of SPC_play again and the content of SPC.7~SPC.0 is 00H, which forces the speech synthesizer into STANDBY mode whether the tasks is finished or not.

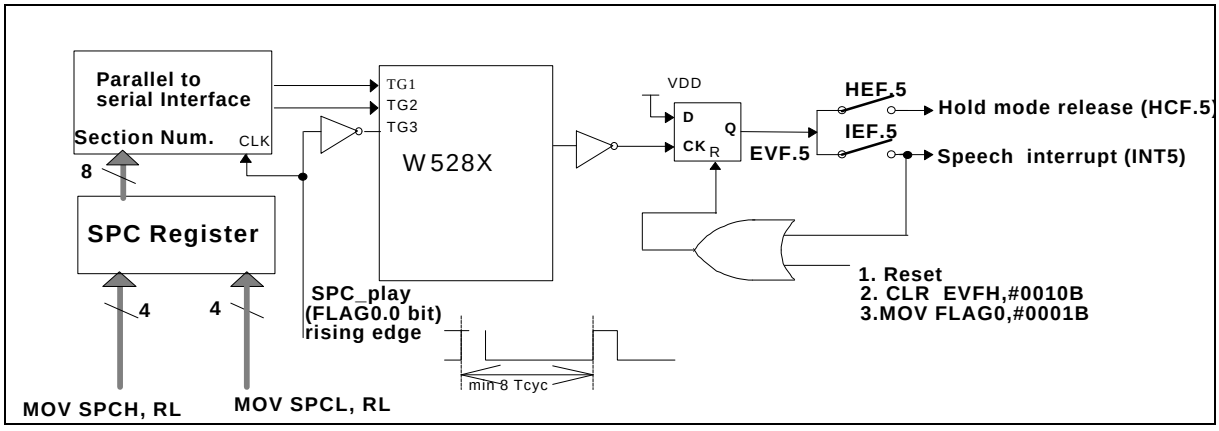
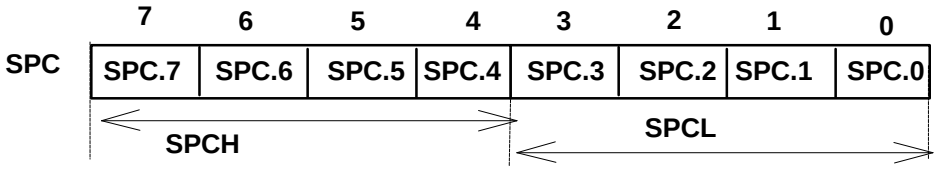


Figure 12. Speech Circuit Diagram

Speech Section Register (SPCL, SPCH with SR=1E, 1F)

The SPCH and SPCL registers named as SPC.7 ~SPC.0 define the speech section that the speech synthesizer is required to play. The SPCH represents the high nibble SPC.7 ~ SPC.4 while the SPCL represents the low nibble SPC.3 ~ SPC.0 . When the speech synthesizer is actived, it plays the voice section pointed by the SPC.7~SPC.0 with maximun 255 sections (01h to FFh). If the content of the SPC register is set to 0, a speech-play command becomes a speech-stop command.



Melody Function

There are 1k notes (22 bits per note) dedicated ROM for dual tone melody code , can be separated as 31 different scores maximun. Uc controls the dual tone melody by the same methodology as speech playing. The melody scores can be write to MLDH, MLDL register. Then MLD_play is enable high to play melody, and the MLD_busy bit will be changed from low to high and keeps high till melody play is ending. If interrupt flag or hold mode flag IEF.6, HEF.6, HCF.6 are set, interrupt or hold mode release will be processed while MLD_busy falling edge occurred. The MLDH, MLDL will be latched during MLD_play . User can select melody play mode by OSB bit (bit 2 of MLDH). In one-shot trigger mode (OSB=0) , the melody synthesizer receives a rising edge of MLD_play then plays the score pointed by MLD5~MLD.0 and pull the voltage level of the MLD_busy to logic 1. When the melody synthesizer finishes its tasks or it receives a rising edge of MLD_play with the score number 00H , the melody synthesizer enters the standby mode and MLD_busy is pulled to logic 0. In level-trig mode(OSB=1) , the melody synthesizer plays the pointed score when MLD_busy is set to 1. The MLD pointed score is repeatedly played and the MLD_busy is pulled high until the MLD_play is cleared by user.

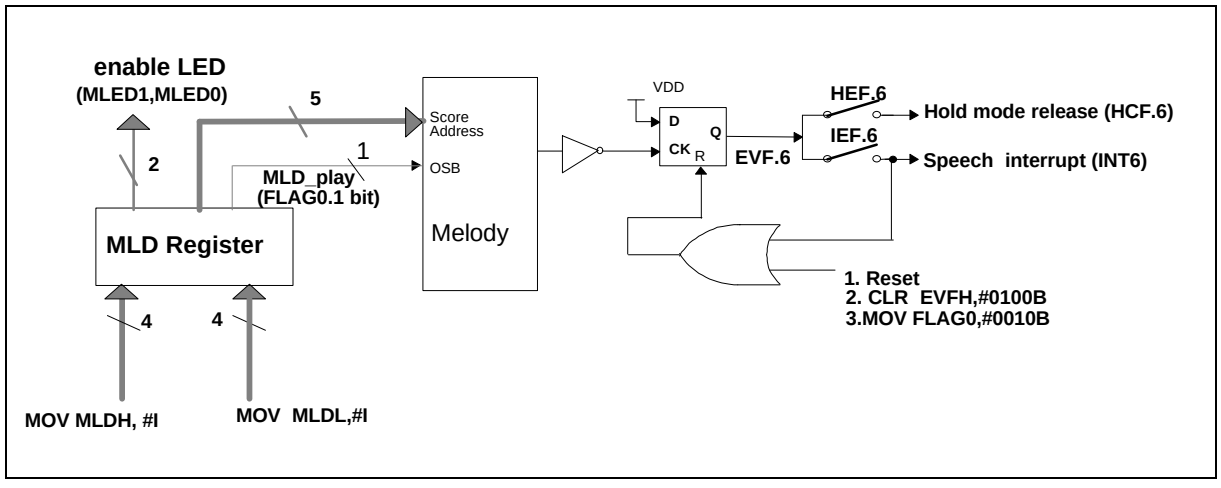
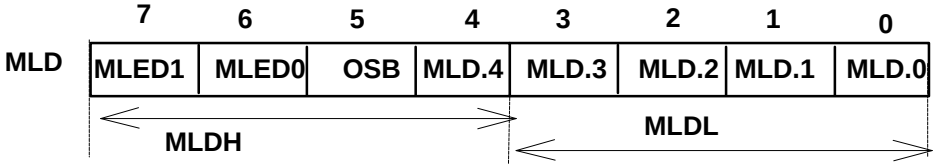


Figure 13. Melody Circuit Diagram

Melody scores Register (MLDL, MLDH with SR=1CH, 1DH)

MLD register is organized by two 4-bit registers, MLDH and MLDL. The MLDH represents the high nibble of MLED1, MLED0 , OSB, MLD.4 while the MLDL represents the low nibble MLD.3 ~ MLD.0. The MLD.4 ~ MLD.0 performs a 5-bit pointer of scores, and MLED1~0 use to control LED1 pin active type during melody playing. When the melody synthesizer is actived, it plays the score section pointed by the MLD.4 ~ MLD.0 . From score 01H to score 1FH, 31 scores can be

pointed by the MLD register. When the melody synthesizer is one-shot trigger mode and MLD.4 ~ MLD.0 is set to 00H, a melody-play command becomes a melody-stop command.



MLD.4 ~ MLD.0 are the melody score number with 31 scores maximun.
OSB=0 : Melody play mode by one shot trgger
=1 : Melody play mode by level trigger
MLED1~0: Selct LED1 output pin active type while melody is playing.
00: LED1 is disable during melody is playing
01: LED1 will be active during melody volume high than low level
10: LED1 will be active during melody volume high than middle level
11: LED1 will be active during melody volume high than high level

PART C: LCD FUNCTION

The W53322/W53342 can directly drive an LCD panel with 32 common output pins and 48/64 segment output pins for a total of 32 × 48/64 dots and the frame updating rate is 64 Hz. Two registers LCDM1 and LCDM2 can use to select different LCD operating type such as duty cycle, bias ratio, maximun pump voltage, internal shunt resistor and enable LCD pump voltage circuit by instruction MOV LCDM1, #I ; MOV LCDM1, RL (where RL is thelow 9 bit of RAM address) and MOV LCDM1, ACC. For power saving issue, LCDEN bit (bit 1 of MR0 register) can select LCD panel on or off ; it is controlled by the LCDON and LCDOFF instructions. The LCDON instruction turns the LCD display on (even in HOLD mode), and the LCDOFF instruction turns the LCD display off. At initial reset, the LCDM1 is 0000B that LCD operating condition is 1/32 duty, 1/7 bias, triple pump voltage with internal shunt resistor, and all the LCD segments are lit. When the initial reset state ends, the LCD display is turned off automatically. The circuit architecture is shown as Figure 14. Many different application condition are shown from Figure 15 to 22.

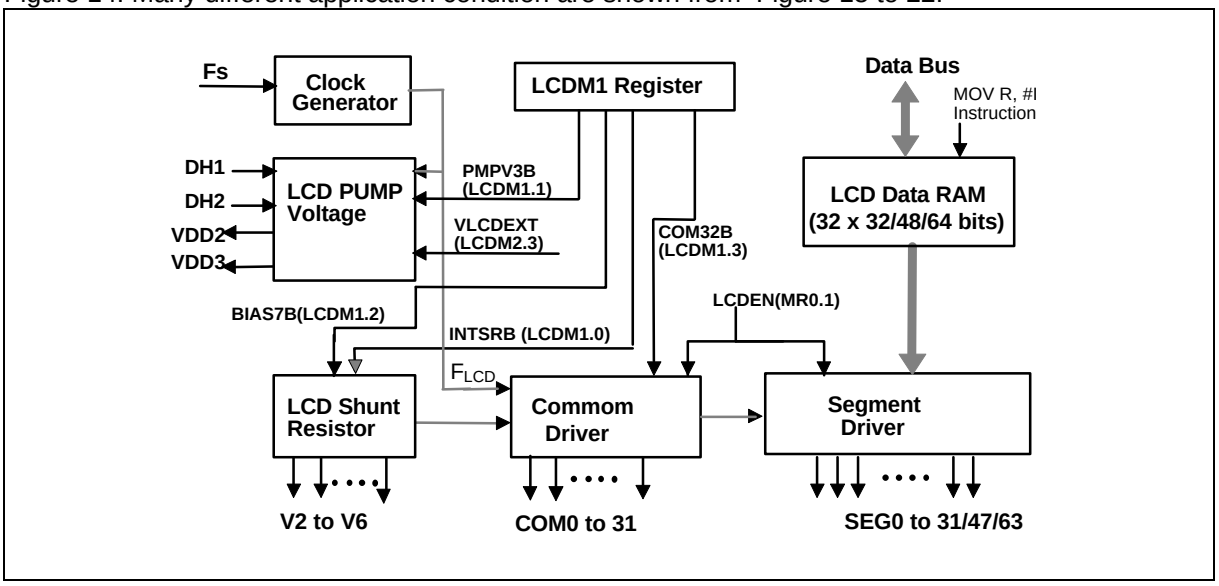


Figure 14. LCD Driver Circuit Diagram

LCD pattern RAM (LCDR000H~LCDR0FFH/LCDR17FH/LCDR1FFH)

Corresponding to the 48/64 LCD drive output pins, there are 384/512 LCD data RAM from 200H to 37FH/3FFH or named as LCDR000H to LCDR17FH/LCDR1FFH. In fact, they are also general purpose RAM, all the operatin instruction is same as RAM area 00H~1FFH . But instructions such as MOV LCDR,#I, MOV WR, LCDR ; MOV LCDR,WR and MOV LCDR, ACC are also available to control the LCD data RAM because LCDR will be added 1FFH in cross assembler automatically. When the bit value of the LCD data RAM is written "1" , the LCD dot is turned on. Otherwise LCD dot is turnned off if RAM bit data is written "0". The contents of the LCD data RAM (LCDR) are sent out to the SEG0~SEG47/SEG63 pins by a direct memory access. The relation between the LCD data RAM and segment/common pins is shown Table 5

LCD DATA RAM	OUTPUT PIN	BIT3	BIT 2	BIT 1	BIT 0
LCDR000 (RAM200)		COM3	COM2	COM1	COM0
LCDR001 (RAM201)		COM7	COM6	COM5	COM4
LCDR002 (RAM202)		COM11	COM10	COM9	COM8
LCDR003 (RAM203)		COM15	COM14	COM13	COM12

LCDR004 (RAM204)	SEG0	COM19	COM18	COM17	COM16
LCDR005 (RAM205)		COM23	COM22	COM21	COM20
LCDR006 (RAM206)		COM27	COM26	COM25	COM24
LCDR007 (RAM207)		COM31	COM30	COM29	COM28
LCDR008 (RAM208)	SEG1	COM3	COM2	COM1	COM0
LCDR009 (RAM209)		COM7	COM6	COM5	COM4
LCDR00A (RAM20A)		COM11	COM10	COM9	COM8
LCDR00B (RAM20B)		COM15	COM14	COM13	COM12
LCDR00C (RAM20C)		COM19	COM18	COM17	COM16
LCDR00D (RAM20D)		COM23	COM22	COM21	COM20
LCDR00E (RAM20E)		COM27	COM26	COM25	COM24
LCDR00F (RAM20F)		COM31	COM30	COM29	COM28
: :	: :	: :	: :	: :	: :
LCDR1F8 (RAM3F8)	SEG63	COM3	COM2	COM1	COM0
LCDR1F9 (RAM3F9)		COM7	COM6	COM5	COM4
LCDR1FA (RAM3FA)		COM11	COM10	COM9	COM8
LCDR1FB (RAM3FB)		COM15	COM14	COM13	COM12
LCDR1FC (RAM3FC)		COM19	COM18	COM17	COM16
LCDR1FD (RAM3FD)		COM23	COM22	COM21	COM20
LCDR1FE (RAM3FE)		COM27	COM26	COM25	COM24
LCDR1FF (RAM3FF)		COM31	COM30	COM29	COM28

Table5. W53342 LCD RAM mapping to segment and common output pins

LCD Mode Register 1 (LCDM1 with SR=2AH)

The LCDM1 register is organized as 4 bit LCDM1.0~LCDM1.3 that LCD duty cycle, bias ratio , pump voltage, internal shunt resistor can be selected by instruction MOV LCDM1, #I ; MOV LCDM1, RL (where RL is thelow 9 bit of RAM address) and MOV LCDM1, ACC.

The COM32B defines the duty cycle . The BIAS7B controls bias ratio to match the characteristic of LCD panel. The PMPV3B is used to choose COM/SEG output maximun voltage either doubler or tripler when the build-in LCD voltage pump circuit is enable. The voltage tripler should be enabled for 3V operating voltage, and the voltage doubler shoule be enabled for 4.5V operating voltage. The INTSRB is used to select the internal shunt reistoe for V2~V6 output power. Please refer to following application circuits. The output waveforms for the five LCD driving modes are shown in Figure

14 to Figure XX

	3	2	1	0
LCDM1	COM32B	BIAS7B	PMPV3B	INTSRB

INTSRB = 0 Internal shunt resistor is available between V2~V6
= 1 External shunt resistor is needed between V2 ~V6.
PMPV3B = 0 Triple pump voltage available (suggeset while VDD=3v)
= 1 Double pump voltage available (suggest while VDD=4.5v)
BIAS7B = 0 1/7 bias available (suggeset for 32 common)
= 1 1/5 bias available (suggest for 16 common)
COM32B =0 1/32 duty, COM0~COM31 output available
=1 1/16 duty, COM0~ COM15 output available
All bit can write only. At initial reset, LCDM1 is 0000B

LCD frame rate divider (LDIV with SR=12H)

The LDIV register is used to define the frame rate of LCD driver. The relationship between the frame rate of LCD driver and the LDIV value is : FLCD = 32768 / [(LDIV+1)*64] . If LDIV value is set to 7 (default value), the frame rate of LCD driver is 64Hz. For 1/16 duty (while LCDM1 bit 3=0), please MOV LDIV, #1111B to get 64 hz frame rate. Otherwise the fame rate will be 128HZ.

.

	3	2	1	0
LDIV	LDIV.3	LDIV.2	LDIV.1	LDIV.0

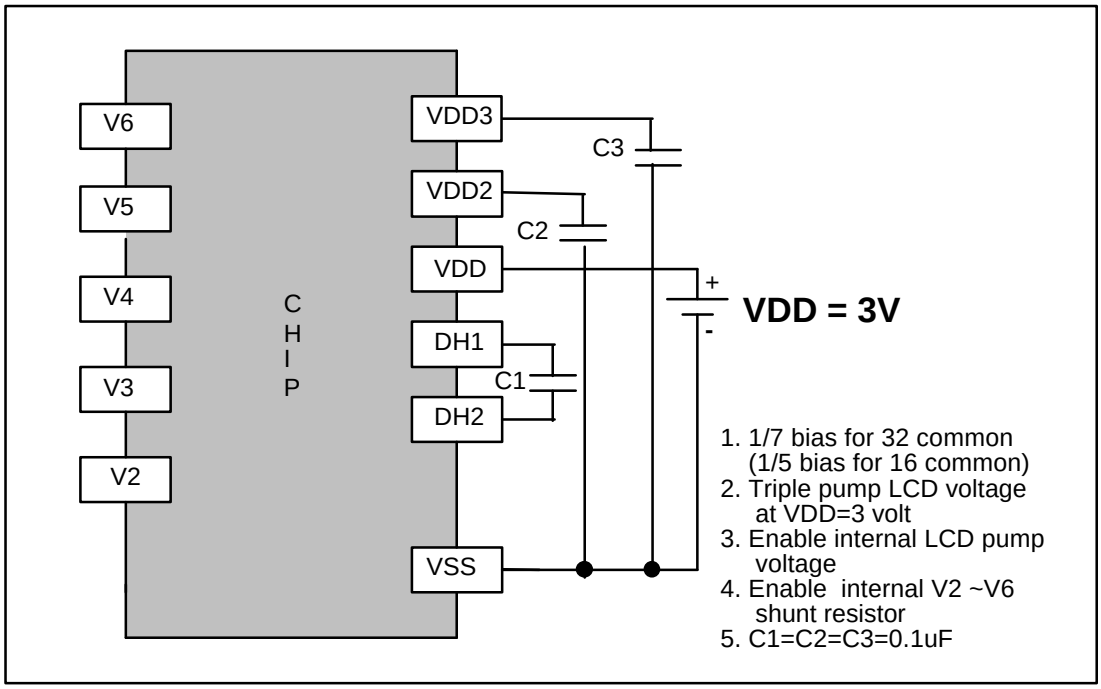


Figure 15. Triple **pump voltage and** internal **shunt resistor**

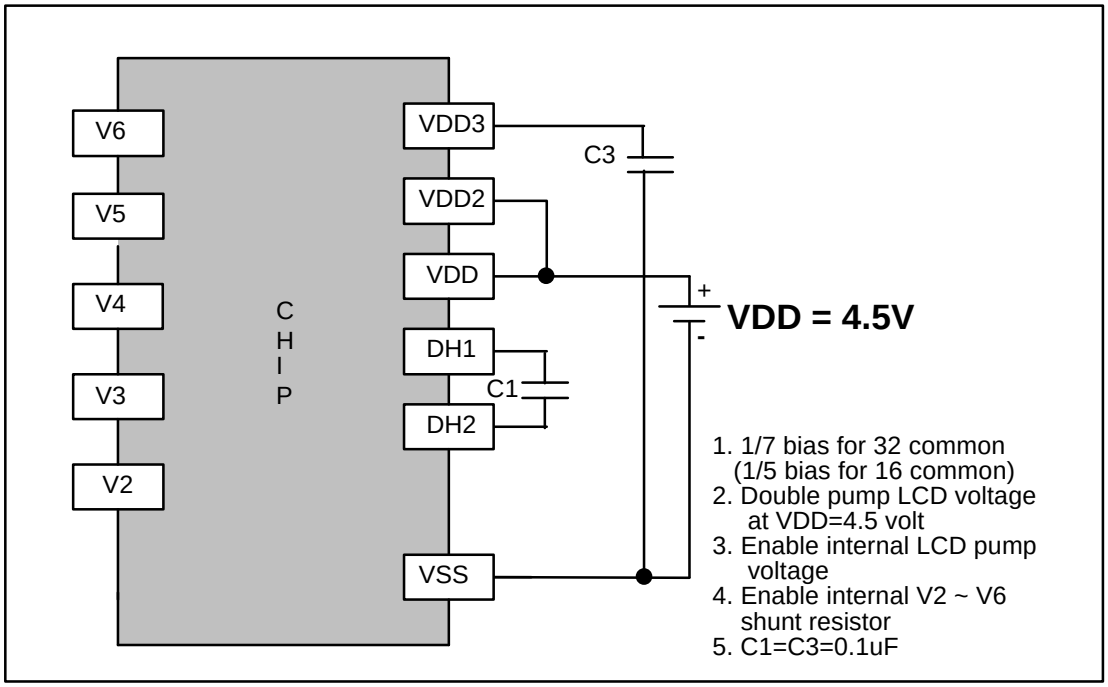


Figure 16. Double **pump voltage and** internal **shunt resistor**

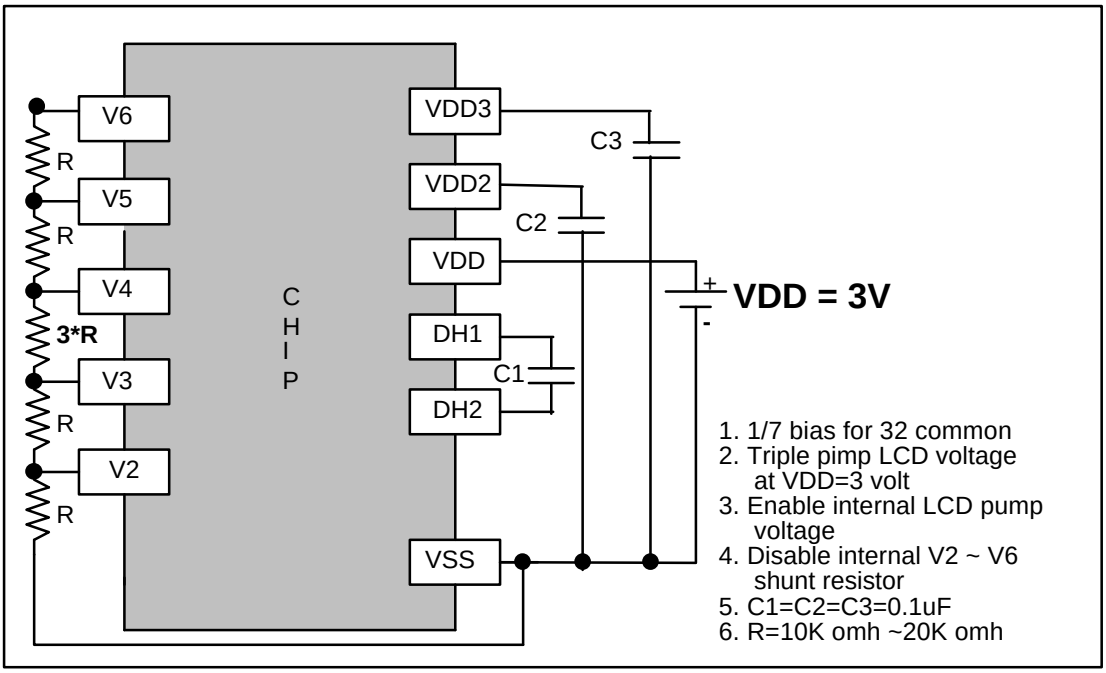


Figure 17. 1/7 bias, Triple **pump voltage and** external **shunt resistor**

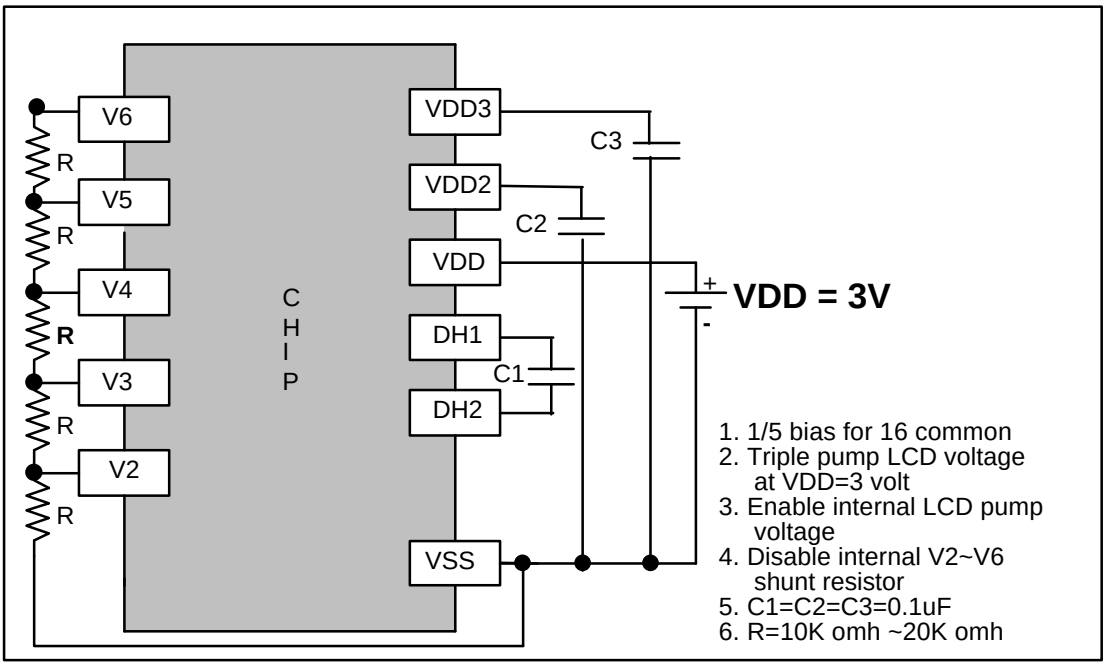


Figure 18. 1/5 bias, Triple pump voltage and external shunt resistor

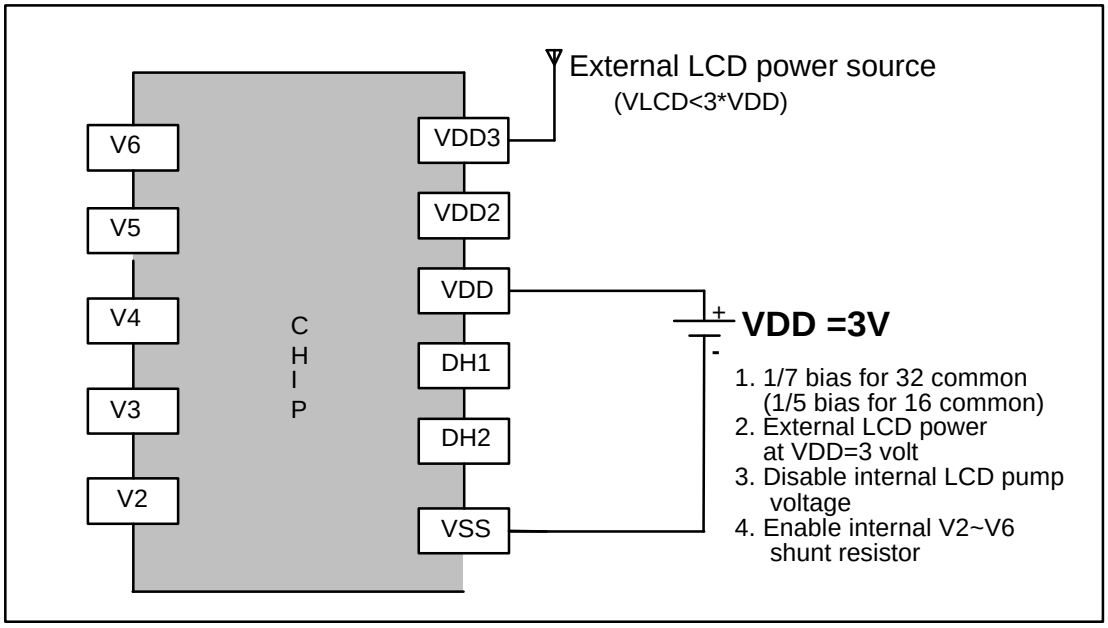


Figure 19. External LCD voltage and external shunt resistor at VDD=3v

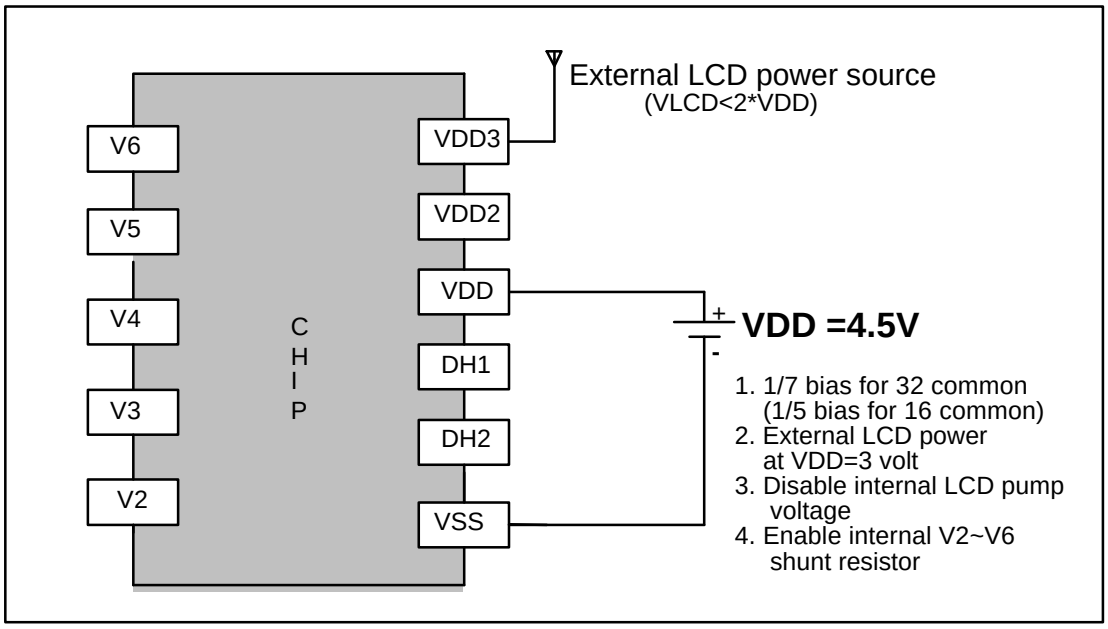


Figure 20. External LCD voltage and external shunt resistor at VDD=4.5v

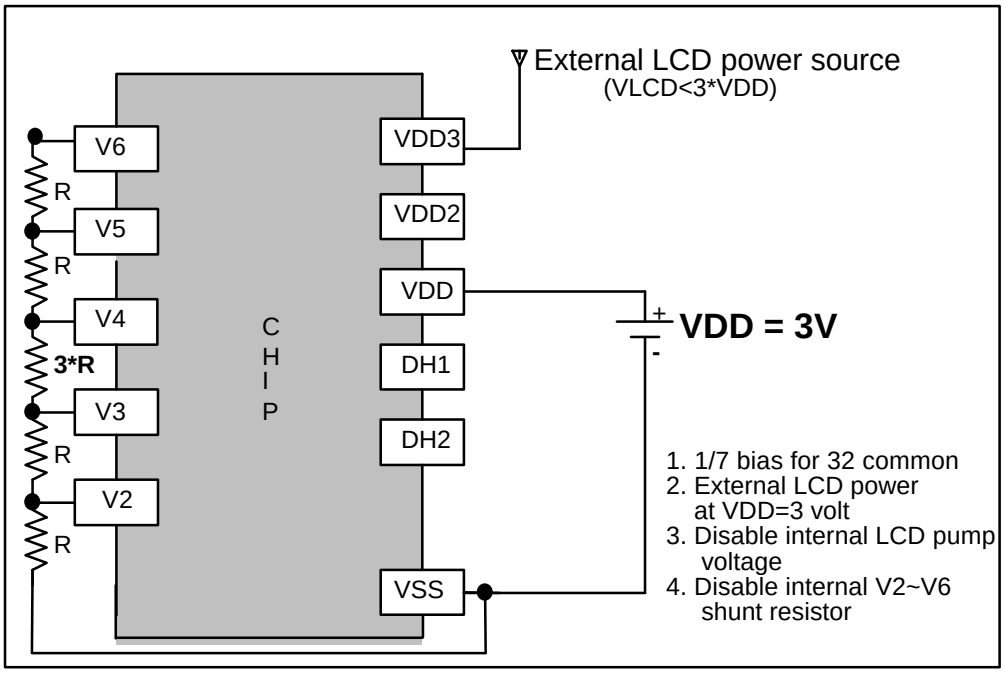


Figure 21. 1/7 bias, External LCD voltage and external shunt resistor

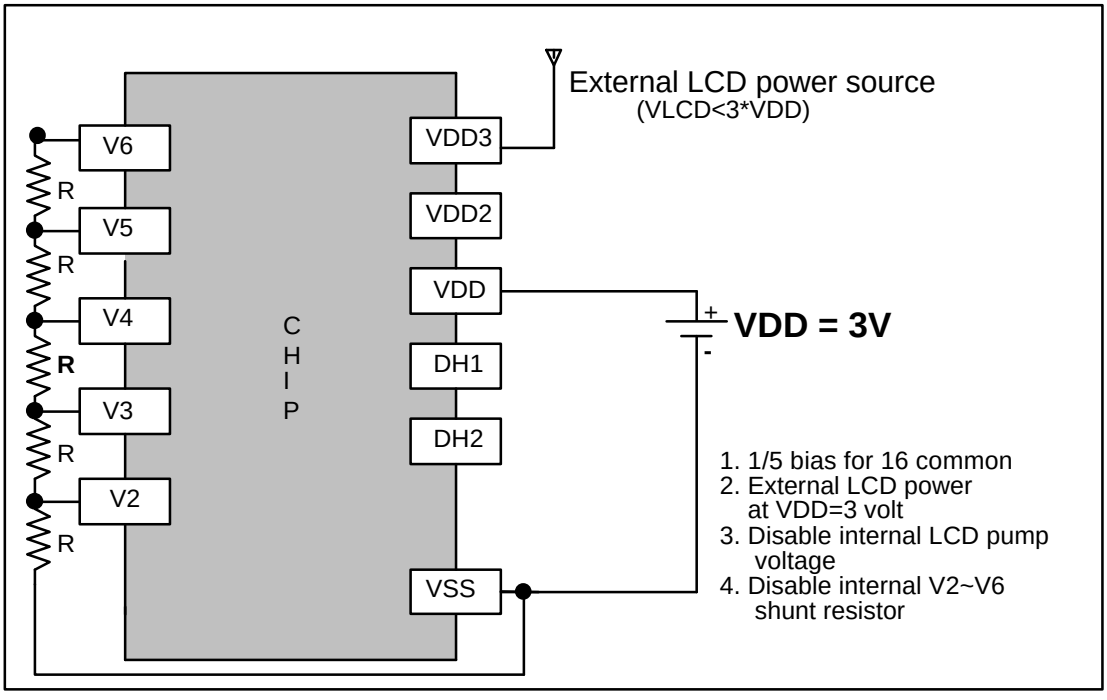


Figure 22. 1/5 bias, External LCD voltage and external shunt resistor

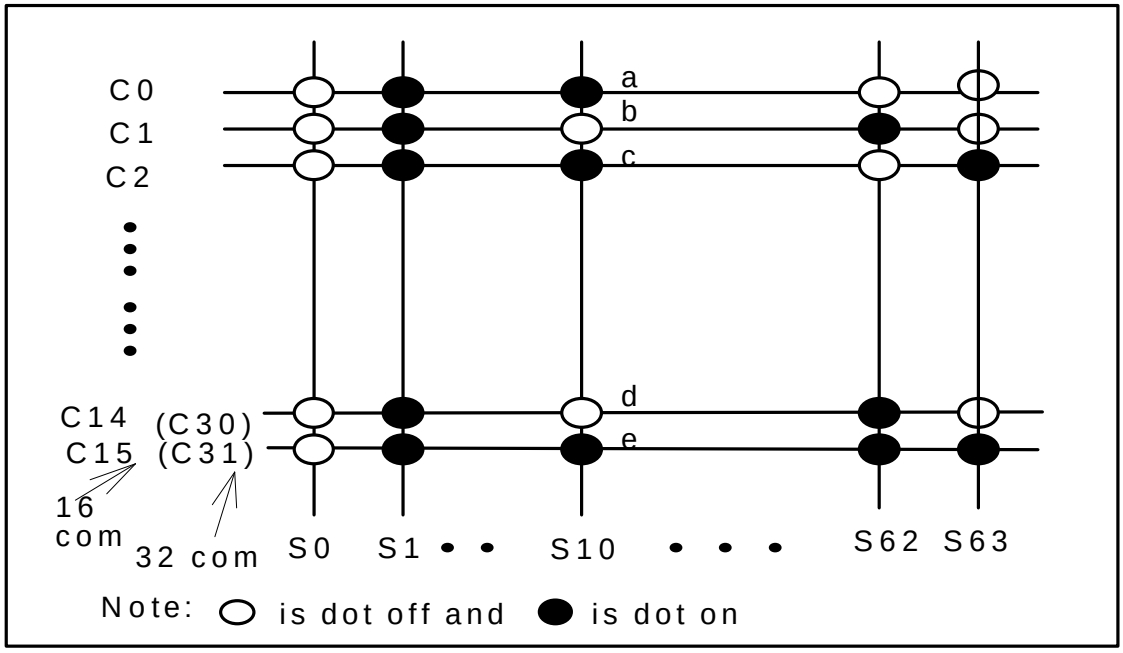


Figure 23. W53342 Common/Segment driving pattern

Accoding Figure 23 pattern assignment, we can get the common, segment output waveform as Figure 24 for 1/7 bias, and Figure 25 for 1/5 bias.

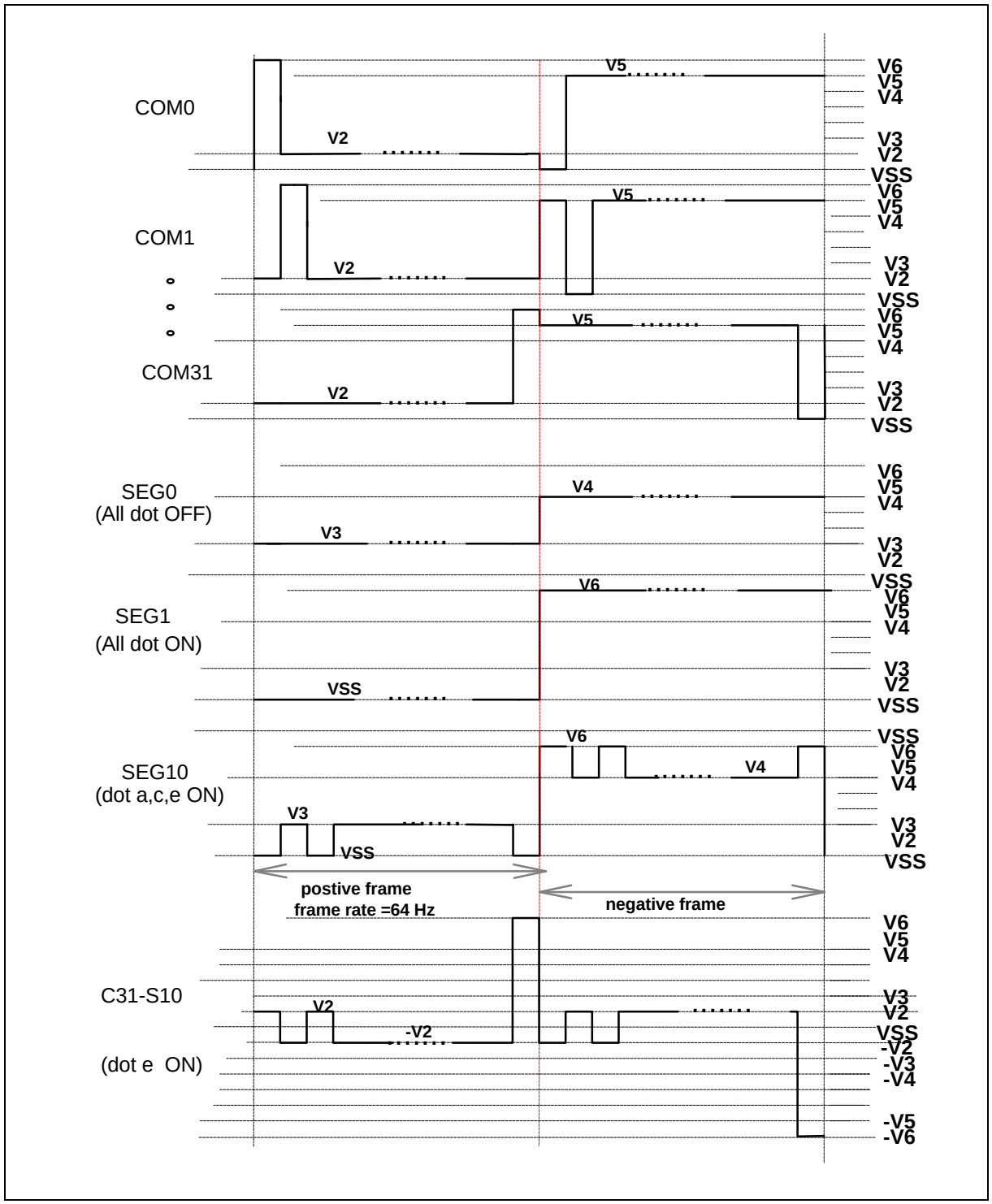


Figure 24. 1/7 bias, 1/32 duty driving waveform

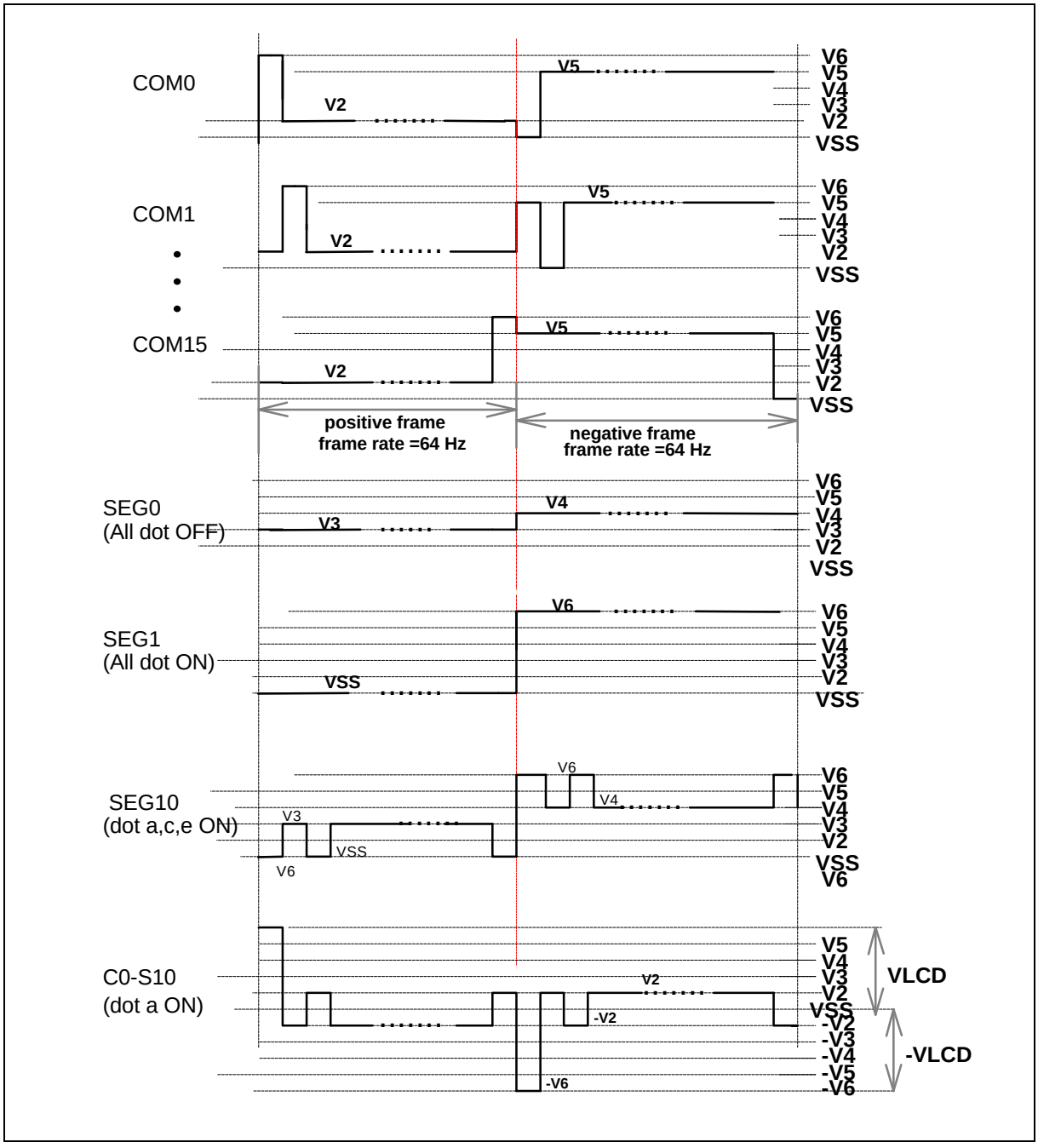


Figure 25. 1/5 bias, 1/16 duty driving waveform

ABSOLUTE MAXIMUM RATINGS

PARAMETER	RATING	UNIT
Supply Voltage to Ground Potential	-0.3 to +7.0	V
Applied Input/Output Voltage	-0.3 to +7.0	V
Power Dissipation	120	mW
Ambient Operating Temperature	0 to +70	°C
Storage Temperature	-55 to +150	°C
VDD3 Input Voltage	12	V

Note: Exposure to conditions beyond those listed under Absolute Maximum Ratings may adversely affect the life and reliability of the device.

DC CHARACTERISTICS

(VDD–VSS = 3.0V, Fm = 1 MHz, Fs = 32.768 KHz, TA = 25° C, LCD on and dot size is 0.5mmm*0.5mm ; unless otherwise specified)

PARAMETER	SYM.	CONDITIONS	MIN.	TYP.	MAX.	UNIT
Op. Voltage	VDD		2.4		5.5	V
Op. Current (No Load)	IOP1	Dual clock with crystal	-	250	300	uA
		Dual clock with RC type		250	300	uA
		Single Clock, LCD ON		60	100	uA
Hold Mode Current (No Load, LCD OFF)	Iop2	Dual clock with crystal		120	150	uA
		Dual clock with RC type		120	150	
		Single clock		6	10	
Stop Mode Current	Iop3	LCD OFF			1	uA
Input Low Voltage	VIL	-	VSS	-	0.3*VDD	V
Input High Voltage	VIH	-	0.7	-	1	VDD
Port RA, RB Output Low Voltage	VABL	IOL = 2.0 mA	-	-	0.4	V
Port RA, RB Output High Voltage	VABH	IOH = -2.0 mA	2.4	-	-	V
Port RE Sink Current	IEL	VOL = 0.4V	2	-	-	mA
Port RE Source Current	IEH	VOH = 2.4V	-2	-	-	mA
Pull-up Resistor	RCD	Port RC, RD	100	350	1000	KΩ
RES Pull-up Resistor	RRES	-	20	100	500	KΩ
LED1/LED2 Sink Current	ILED	VO=1 volt		8		mA
PWM1/2 Source Current	ISPH	VOL = 2.4V CUR1~0=00	-30			mA
		VOL = 2.4V CUR1~0=01	-60			
		VOL = 2.4V CUR1~0=10	-90			
		VOL = 2.4V CUR1~0=11	-120			

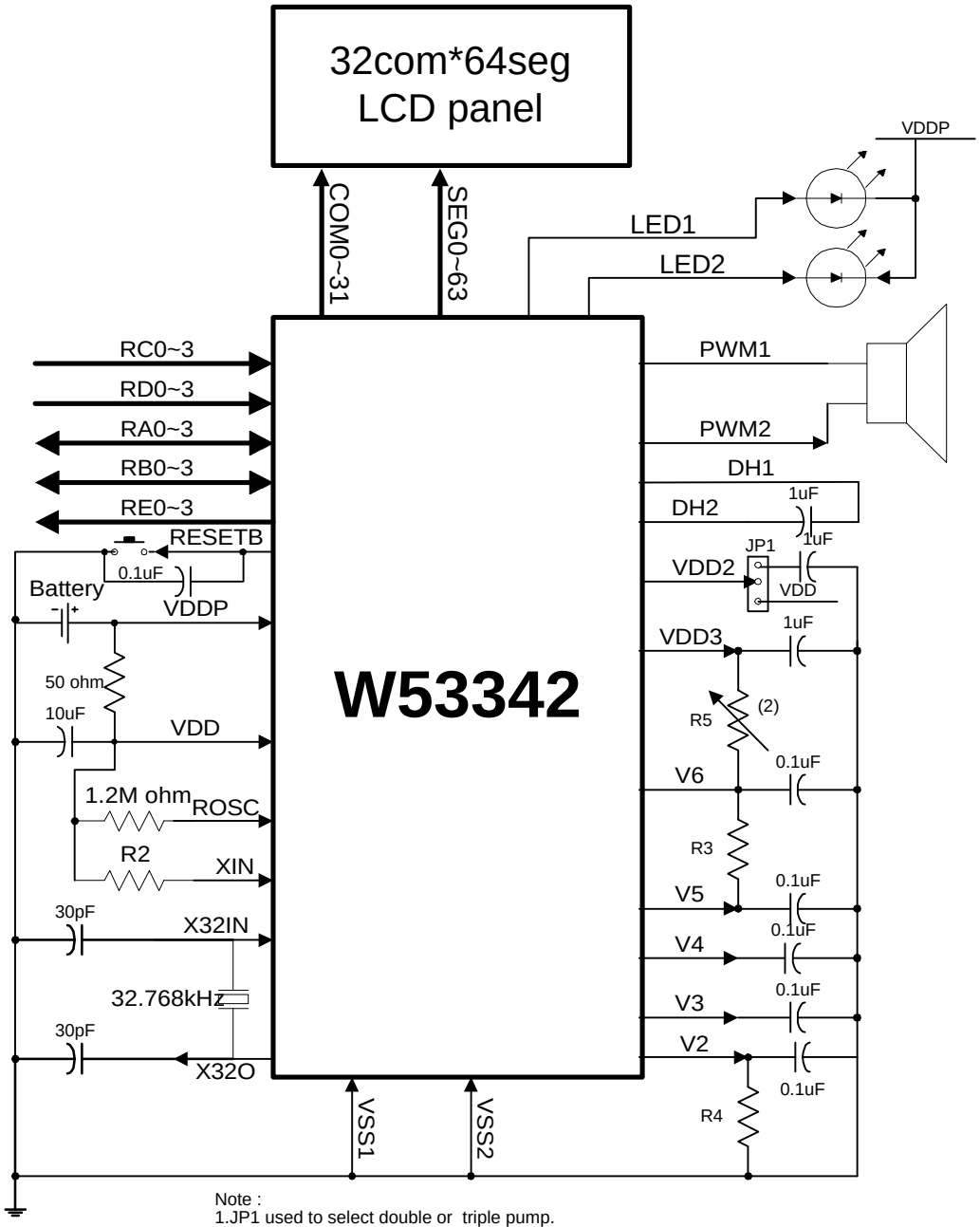
PWM1/2 Sink Current	ISPL	VOL = 0.6V CUR1~0=00	30			mA
		VOL = 0.6V CUR1~0=01	60			
		VOL = 0.6V CUR1~0=10	90			
		VOL = 0.6V CUR1~0=11	120			
LCD Supply Current	ILCD	dot size 0.5mm*0.5mmm, All Seg. ON	-	50	-	μA
COM/SEG On Resistor	Ron	IOH = ± 50 μA		5K	10K	Ω
PARAMETER	**SYM.**	**CONDITIONS**	**MIN.**	**TYP.**	**MAX.**	**UNIT**
VDD2 output voltage	VDOB	VLCDEXT=0 & PMPV3B=0		2		VDD
		VLCDEXT=0 & PMPV3B=1		1		
VDD3 output Voltage	VTRI	VLCDEXT=0 & PMPV3B=0		3		VDD
		VLCDEXT=0 & PMPV3B=1		2		
VDD3 Input Voltage	VLCD	VLCDEXT=1		7	10	V

AC CHARATERISTICS

(VDD–VSS = 3.0V, Fm = 1 MHz, Fs = 32.768 KHz, TA = 25° C, LCD on; unless otherwise specified)

PARAMETER	SYM.	CONDITIONS	MIN.	TYP.	MAX.	UNIT
Sub-clock Frequency	FS	Crystal type		32768		Hz
Main-clock Frequency	FM	RC type/Crystal type	400	-	4190	KHz
Op. Frequency	FOSC	SCR.0=1		32768		KHz
		SCR.0=0	400	-	4190	
Instruction Cycle Time	TCYC	One machine cycle	-	4/FOSC	-	S
Reset Active Width	TRAW	FOSC = 32.768 KHz	1	-	-	μS
Interrupt Active Width	TIAW	FOSC = 32.768 KHz	1	-	-	μS
Main clock RC frequency	FRXIN	RXIN =2.4 MΩ		400K		Hz
		RXIN =1.2 MΩ		800K		
		RXIN =910 KΩ		1M		
		RXIN =160 KΩ		4M		
Frequency Deviation of main-clock FRXIN =1MHz	$\frac{\Delta f}{f}$	$\frac{f(3V)-f(2.4V)}{f(3V)}$	-	-	10	%
ROSC Frequency	FROSC	ROSC =1.2MΩ		3.23		MHz
Frequency Deviation of FROSC = 3MHz	$\frac{\Delta f}{f}$	$\frac{f(3V)-f(2.4V)}{f(3V)}$	-	-	10	%
Frame frequency	FLCD	LDIV = 0111b &1/32 duty		64		Hz
		LDIV= 1111b & 1/16 duty		64		Hz

TYPICAL APPLICATION CIRCUIT



Note :
1.JP1 used to select double or triple pump.
2.R5=0 if double pump is active.
3.R3 and R4 are optional.
4.LCD duty and bias are programmed by registers .

INSTRUCTION SET TABLE

SYMBOL DESCRIPTION

WR:	Working RAM
SR:	special register
SRP:	special register pair
ACC:	Accumulator
ACC.n:	Accumulator bit n
R:	Memory (RAM) addressed by 10 bit direct address R
R.n:	Bit n of memory (RAM) addressed by 10 bit direct address R
RL:	Lower-half memory (RAM) addressed by 9 bit direct address RL
RL.n:	Bit n of lower-half memory (RAM) addressed by 9 bit direct address RL
P:	RAM pointer RP0/RP1
@P:	Memory (RAM) addressed by pointer RP0/RP1
@P.n:	Memory (RAM) bit n addressed by pointer RP0/RP1
LUPC:	ROM pointer, for use of look-up-table
@LUPC:	Memory (ROM) addressed by pointer LUPC
I:	Constant parameter
L:	Branch or jump address
CF:	Carry Flag
ZF:	Zero Flag
PC:	Program Counter

! =:	Not equal
&:	AND
^:	OR
EX:	Exclusive OR
←:	Transfer direction, result

COMPLETE INSTRUCTION SET TABLE 2

MNEMONIC		FUNCTION	FLAG AFFECTED	CYCLE
Arithmetic & Logic Operations				
ADDC	R, ACC	ACC ← R + ACC + CF	CF & ZF	1
ADDC	@RP0, ACC	ACC ← @RP0 + ACC + CF	CF & ZF	1
ADDC	@RP1, ACC	ACC ← @RP1 + ACC + CF	CF & ZF	1
ADDC	WR, #I	ACC ← WR + I + CF	CF & ZF	1
ADDC	@RP0, #I	ACC ← @RP0 + I + CF	CF & ZF	1
ADDC	@RP1, #I	ACC ← @RP1 + I + CF	CF & ZF	1
ADDC	ACC, #I	ACC ← ACC + I + CF	CF & ZF	1
ADDCR	R, ACC	ACC, R ← R + ACC + CF	CF & ZF	1
ADDCR	@RP0, ACC	ACC, @RP0 ← @RP0 + ACC + CF	CF & ZF	1
ADDCR	@RP1, ACC	ACC, @RP1 ← @RP1 + ACC + CF	CF & ZF	1
ADDCR	WR, #I	ACC, WR ← WR + I + CF	CF & ZF	1
ADDCR	@RP0, #I	ACC, @RP0 ← @RP0 + I + CF	CF & ZF	1
ADDCR	@RP1, #I	ACC, @RP1 ← @RP1 + I + CF	CF & ZF	1
ADD	R, ACC	ACC ← R + ACC	CF & ZF	1
ADD	@RP0, ACC	ACC ← @RP0 + ACC	CF & ZF	1
ADD	@RP1, ACC	ACC ← @RP1 + ACC	CF & ZF	1
ADD	WR, #I	ACC ← WR + I	CF & ZF	1
ADD	@RP0, #I	ACC ← @RP0 + I	CF & ZF	1
ADD	@RP1, #I	ACC ← @RP1 + I	CF & ZF	1
ADD	ACC, #I	ACC ← ACC + I	CF & ZF	1
ADDR	R, ACC	ACC, R ← R + ACC	CF & ZF	1
ADDR	@RP0, ACC	ACC, @RP0 ← @RP0 + ACC	CF & ZF	1
ADDR	@RP1, ACC	ACC, @RP1 ← @RP1 + ACC	CF & ZF	1
ADDR	WR, #I	ACC, WR ← WR + I	CF & ZF	1
ADDR	@RP0, #I	ACC, @RP0 ← @RP0 + I	CF & ZF	1
ADDR	@RP1, #I	ACC, @RP1 ← @RP1 + I	CF & ZF	1

ADDU	R, ACC	ACC ← R + ACC	ZF	1
ADDU	@RP0, ACC	ACC ← @RP0 + ACC	ZF	1
ADDU	@RP1, ACC	ACC ← @RP1 + ACC	ZF	1
ADDU	WR, #I	ACC ← WR + I	ZF	1
ADDU	@RP0, #I	ACC ← @RP0 + I	ZF	1
ADDU	@RP1, #I	ACC ← @RP1 + I	ZF	1
ADDU	ACC, #I	ACC ← ACC + I	ZF	1
ADDUR	R, ACC	ACC, R ← R + ACC	ZF	1
ADDUR	@RP0, ACC	ACC, @RP0 ← @RP0 + ACC	ZF	1
ADDUR	@RP1, ACC	ACC, @RP1 ← @RP1 + ACC	ZF	1
ADDUR	WR, #I	ACC, WR ← WR + I	ZF	1
ADDUR	@RP0, #I	ACC, @RP0 ← @RP0 + I	ZF	1
ADDUR	@RP1, #I	ACC, @RP1 ← @RP1 + I	ZF	1
SUBB	R, ACC	ACC ← R - ACC - CF	CF & ZF	1
SUBB	@RP0, ACC	ACC ← @RP0 - ACC - CF	CF & ZF	1
SUBB	@RP1, ACC	ACC ← @RP1 - ACC - CF	CF & ZF	1
SUBB	WR #I	ACC ← WR - I - CF	CF & ZF	1
SUBB	@RP0, #I	ACC ← @RP0 - I - CF	CF & ZF	1
SUBB	@RP1, #I	ACC ← @RP1 - I - CF	CF & ZF	1
SUBB	ACC, #I	ACC ← ACC - I - CF	CF & ZF	1
SUBBR	R, ACC	ACC, R ← R - ACC - CF	CF & ZF	1
SUBBR	@RP0, ACC	ACC, @RP0 ← @RP0 - ACC - CF	CF & ZF	1
SUBBR	@RP1, ACC	ACC, @RP1 ← @RP1 - ACC - CF	CF & ZF	1
SUBBR	WR, #I	ACC, WR ← WR - I - CF	CF & ZF	1
SUBBR	@RP0, #I	ACC, @RP0 ← @RP0 - I - CF	CF & ZF	1
SUBBR	@RP1, #I	ACC, @RP1 ← @RP1 - I - CF	CF & ZF	1
SUB	R, ACC	ACC ← R - ACC	CF & ZF	1
SUB	@RP0, ACC	ACC ← @RP0 - ACC	CF & ZF	1
SUB	@RP1, ACC	ACC ← @RP1 - ACC	CF & ZF	1
SUB	WR, #I	ACC ← WR - I	CF & ZF	1
SUB	@RP0, #I	ACC ← @RP0 - I	CF & ZF	1
SUB	@RP1, #I	ACC ← @RP1 - I	CF & ZF	1

SUB	ACC, #I	ACC ← ACC - I	CF & ZF	1
SUBR	R, ACC	ACC, R ← R - ACC	CF & ZF	1
SUBR	@RP0, ACC	ACC, @RP0 ← @RP0 - ACC	CF & ZF	1
SUBR	@RP1, ACC	ACC, @RP1 ← @RP1 - ACC	CF & ZF	1
SUBR	WR, #I	ACC, WR ← WR - I	CF & ZF	1
SUBR	@RP0, #I	ACC, @RP0 ← @RP0 - I	CF & ZF	1
SUBR	@RP1, #I	ACC, @RP1 ← @RP1 - I	CF & ZF	1
ANL	R, ACC	ACC ← R & ACC	ZF	1
ANL	@RP0, ACC	ACC ← @RP0 & ACC	ZF	1
ANL	@RP1, ACC	ACC ← @RP1 & ACC	ZF	1
ANL	WR, #I	ACC ← WR & I	ZF	1
ANL	@RP0, #I	ACC ← @RP0 & I	ZF	1
ANL	@RP1, #I	ACC ← @RP1 & I	ZF	1
ANL	ACC, #I	ACC ← ACC & I	ZF	1
ANLR	R, ACC	ACC, R ← R & ACC	ZF	1
ANLR	@RP0, ACC	ACC, @RP0 ← @RP0 & ACC	ZF	1
ANLR	@RP1, ACC	ACC, @RP1 ← @RP1 & ACC	ZF	1
ANLR	WR, #I	ACC, WR ← WR & I	ZF	1
ANLR	@RP0, #I	ACC, @RP0 ← @RP0 & I	ZF	1
ANLR	@RP1, #I	ACC, @RP1 ← @RP1 & I	ZF	1
XRL	R, ACC	ACC ← R EX ACC	ZF	1
XRL	@RP0, ACC	ACC ← @RP0 EX ACC	ZF	1
XRL	@RP1, ACC	ACC ← @RP1 EX ACC	ZF	1
XRL	WR, #I	ACC ← WR EX I	ZF	1
XRL	@RP0, #I	ACC ← @RP0 EX I	ZF	1
XRL	@RP1, #I	ACC ← @RP1 EX I	ZF	1
XRL	ACC, #I	ACC ← ACC EX I	ZF	1
XRLR	R, ACC	ACC, R ← R EX ACC	ZF	1
XRLR	@RP0, ACC	ACC, @RP0 ← @RP0 EX ACC	ZF	1
XRLR	@RP1, ACC	ACC, @RP1 ← @RP1 EX ACC	ZF	1
XRLR	WR, #I	ACC, WR ← WR EX I	ZF	1

XRLR	@RP0, #I	ACC, @RP0 ← @RP0 EX I	ZF	1
XRLR	@RP1, #I	ACC, @RP1 ← @RP1 EX I	ZF	1
ORL	R, ACC	ACC ← R ∧ ACC	ZF	1
ORL	@RP0, ACC	ACC ← @RP0 ∧ ACC	ZF	1
ORL	@RP1, ACC	ACC ← @RP1 ∧ ACC	ZF	1
ORL	WR , #I	ACC ← WR ∧ I	ZF	1
ORL	@RP0, #I	ACC ← @RP0 ∧ I	ZF	1
ORL	@RP1, #I	ACC ← @RP1 ∧ I	ZF	1
ORL	ACC, #I	ACC ← ACC ∧ I	ZF	1
ORLR	R, ACC	ACC, R ← R ∧ ACC	ZF	1
ORLR	@RP0, ACC	ACC, @RP0 ← @RP0 ∧ ACC	ZF	1
ORLR	@RP1, ACC	ACC, @RP1 ← @RP1 ∧ ACC	ZF	1
ORLR	WR , #I	ACC, WR ← WR ∧ I	ZF	1
ORLR	@RP0, #I	ACC, @RP0 ← @RP0 ∧ I	ZF	1
ORLR	@RP1, #I	ACC, @RP1 ← @RP1 ∧ I	ZF	1
SKB0	R	PC ← PC + 2; if R.0 = 1	ZF	1
SKB1	R	PC ← PC + 2; if R.1 = 1	ZF	1
SKB2	R	PC ← PC + 2; if R.2 = 1	ZF	1
SKB3	R	PC ← PC + 2; if R.3 = 1	ZF	1
SKB0	@RP0	PC ← PC + 2; if @RP0.0 = 1	ZF	1
SKB1	@RP0	PC ← PC + 2; if @RP0.1 = 1	ZF	1
SKB2	@RP0	PC ← PC + 2; if @RP0.2 = 1	ZF	1
SKB3	@RP0	PC ← PC + 2; if @RP0.3 = 1	ZF	1
SKB0	@RP1	PC ← PC + 2; if @RP1.0 = 1	ZF	1
SKB1	@RP1	PC ← PC + 2; if @RP1.1 = 1	ZF	1
SKB2	@RP1	PC ← PC + 2; if @RP1.2 = 1	ZF	1
SKB3	@RP1	PC ← PC + 2; if @RP1.3 = 1	ZF	1
SKB0	ACC	PC ← PC + 2; if ACC.0 = 1	ZF	1
SKB1	ACC	PC ← PC + 2; if ACC.1 = 1	ZF	1
SKB2	ACC	PC ← PC + 2; if ACC.2 = 1	ZF	1
SKB3	ACC	PC ← PC + 2; if ACC.3 = 1	ZF	1
SKNB0	R	PC ← PC + 2; if R.0 = 0	ZF	1

SKNB1	R	PC ← PC + 2; if R.1 = 0	ZF	1
SKNB2	R	PC ← PC + 2; if R.2 = 0	ZF	1
SKNB3	R	PC ← PC + 2; if R.3 = 0	ZF	1
SKNB0	@RP0	PC ← PC + 2; if @RP0.0 = 0	ZF	1
SKNB1	@RP0	PC ← PC + 2; if @RP0.1 = 0	ZF	1
SKNB2	@RP0	PC ← PC + 2; if @RP0.2 = 0	ZF	1
SKNB3	@RP0	PC ← PC + 2; if @RP0.3 = 0	ZF	1
SKNB0	@RP1	PC ← PC + 2; if @RP1.0 = 0	ZF	1
SKNB1	@RP1	PC ← PC + 2; if @RP1.1 = 0	ZF	1
SKNB2	@RP1	PC ← PC + 2; if @RP1.2 = 0	ZF	1
SKNB3	@RP1	PC ← PC + 2; if @RP1.3 = 0	ZF	1
SKNB0	ACC	PC ← PC + 2; if ACC.0 = 0	ZF	1
SKNB1	ACC	PC ← PC + 2; if ACC.1 = 0	ZF	1
SKNB2	ACC	PC ← PC + 2; if ACC.2 = 0	ZF	1
SKNB3	ACC	PC ← PC + 2; if ACC.3 = 0	ZF	1
SHLC	R	ACC.n, R.n ← R.n-1; ACC.0, R.0 ← 0; CF ← R.3	CF & ZF	1
SHRC	R	ACC.n, R.n ← R.n+1; ACC.3, R.3 ← 0; CF ← R.0	CF & ZF	1
RLC	R	ACC.n, R.n ← R.n-1; CF ← R.3; ACC.0, R.0 ← CF	CF & ZF	1
RRC	R	ACC.n, R.n ← R.n+1; CF ← R.0; ACC.3, R.3 ← CF	CF & ZF	1
SHLC	@RP0	ACC.n, @RP0.n ← @RP0.n-1; ACC.0, @RP0.0 ← 0; CF ← @RP0.3	CF & ZF	1
SHRC	@RP0	ACC.n, @RP0.n ← @RP0.n+1; ACC.3, @RP0.3 ← 0; CF ← @RP0.0	CF & ZF	1
RLC	@RP0	ACC.n, @RP0.n ← @RP0.n-1; CF ← @RP0.3; ACC.0, @RP0.0 ← CF	CF & ZF	1
RRC	@RP0	ACC.n, @RP0.n ← @RP0.n+1; CF ← @RP0.0; ACC.3, @RP0.3 ← CF	CF & ZF	1

SHLC @RP1	ACC.n, @RP1.n ← @RP1.n-1; ACC.0, @RP1.0 ← 0; CF ← @RP1.3	CF & ZF	1
SHRC @RP1	ACC.n, @RP1.n ← @RP1.n+1; ACC.3, @RP1.3 ← 0; CF ← @RP1.0	CF & ZF	1
RLC @RP1	ACC.n, @RP1.n ← @RP1.n-1; CF ← @RP1.3; ACC.0, @RP1.0 ← CF	CF & ZF	1
RRC @RP1	ACC.n, @RP1.n ← @RP1.n+1; CF ← @RP1.0; ACC.3, @RP1.3 ← CF	CF & ZF	1
SHLC ACC	ACC.n ← (ACC.n-1); ACC.0 ← 0; CF ← ACC.3	CF & ZF	1
SHRC ACC	ACC.n ← (ACC.n+1); ACC.3 ← 0; CF ← ACC.0	CF & ZF	1
RLC ACC	ACC.n ← (ACC.n-1); ACC.0 ← CF; CF ← ACC.3	CF & ZF	1
RRC ACC	ACC.n ← (ACC.n+1); ACC.3 ← CF; CF ← ACC.0	CF & ZF	1
DSKZ R	ACC, R ← R - 1; PC ← PC + 2 if ACC = 0	ZF	1
DSKNZ R	ACC, R ← R - 1; PC ← PC + 2 if ACC ! = 0	ZF	1
DSKZ @RP0	ACC, @RP0 ← @RP0 - 1; PC ← PC + 2 if ACC = 0	ZF	1
DSKNZ @RP0	ACC, @RP0 ← @RP0 - 1; PC ← PC + 2 if ACC ! = 0	ZF	1
DSKZ @RP1	ACC, @RP1 ← @RP1 - 1; PC ← PC + 2 if ACC = 0	ZF	1
DSKNZ @RP1	ACC, @RP1 ← @RP1 - 1; PC ← PC + 2 if ACC ! = 0	ZF	1
DSKZ ACC	ACC ← ACC - 1; PC ← PC + 2 if ACC = 0	ZF	1
DSKNZ ACC	ACC ← ACC - 1; PC ← PC + 2 if ACC ! = 0	ZF	1
DEC R	ACC, R ← R - 1	CF & ZF	1
INC R	ACC, R ← R + 1	CF & ZF	1
DEC @RP0	ACC, @RP0 ← @RP0 - 1	CF & ZF	1
INC @RP0	ACC, @RP0 ← @RP0 + 1	CF & ZF	1
DEC @RP1	ACC, @RP1 ← @RP1 - 1	CF & ZF	1

INC @RP1	ACC, @RP1 ← @RP1 + 1	CF & ZF	1
DEC ACC	ACC ← ACC - 1	CF & ZF	1
INC ACC	ACC ← ACC + 1	CF & ZF	1
Branch			
CALL L	STACK ← PC+1; PC13 ~ PC0 ← L13 ~ L0		1
JP L	PC13 ~ PC0 ← L13 ~ L0		1
JC L	PC13 ~ PC0 ← L13 ~ L0; if CF = "1"		1
JNC L	PC13 ~ PC0 ← L13 ~ L0; if CF = "0"		1
JZ L	PC13 ~ PC0 ← L13 ~ L0; if ACC = 0		1
JNZ L	PC13 ~ PC0 ← L13 ~ L0; if ACC ! = 0		1
JB0 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.0 = "1"		1
JB1 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.1 = "1"		1
JB2 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.2="1"		1
JB3 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.3 = "1"		1
JNB0 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.0 = "0"		1
JNB1 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.1 = "0"		1
JNB2 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.2="0"		1
JNB3 L	PC13 ~ PC0 ← L13 ~ L0; if ACC.3 = "0"		1
SET/CLR Special Registers			
SET HEFH, #I	HEFH.0 = 1, if I0 =1 HEFH.1 = 1, if I1 =1 HEFH.2 = 1, if I2 =1 HEFH.3 = 1, if I3 =1		1
SET HEFL, #I	HEFL.0 = 1, if I0 =1 HEFL.1 = 1, if I1 =1 HEFL.2 = 1, if I2 =1 HEFL.3 = 1, if I3 =1		1
SET IEFH, #I	IEFH.0 = 1, if I0 =1 IEFH.1 = 1, if I1 =1 IEFH.2 = 1, if I2 =1 IEFH.3 = 1, if I3 =1		1

SET IEFL, #I	IEFL.0 = 1, if I0 =1 IEFL.1 = 1, if I1 =1 IEFL.2 = 1, if I2 =1 IEFL.3 = 1, if I3 =1		1
SET PEFH, #I	PEFH.0 = 1, if I0 =1 PEFH.1 = 1, if I1 =1 PEFH.2 = 1, if I2 =1 PEFH.3 = 1, if I3 =1		1
SET PEFL, #I	PEFL.0 = 1, if I0 =1 PEFL.1 = 1, if I1 =1 PEFL.2 = 1, if I2 =1 PEFL.3 = 1, if I3 =1		1
SET CF	set carry flag =1		1
SET FLAG0, #I	FLAG0.3 and FLAG0.2 can not be set FLAG0.1 = 1, if I1 =1 FLAG0.0 = 1, if I0 =1		1
SET MR0, #I	MR0.0 = 1, if I0 =1 MR0.1 = 1, if I1 =1 MR0.2 = 1, if I2 =1 MR0.3 = 1, if I3 =1		1
SET MR2, #I	MR2.0 = 1, if I0 =1 MR2.1 = 1, if I1 =1 MR2.2 = 1, if I2 =1 MR2.3 = 1, if I3 =1		1
SET SCR, #I	SCR.0 = 1, if I0 =1 SCR.1 = 1, if I1 =1 SCR.2 = 1, if I2 =1 SCR.3 = 1, if I3 =1		1
SET PM0, #I	PM0.0 = 1, if I0 =1 PM0.1 = 1, if I1 =1 PM0.2 = 1, if I2 =1 PM0.3 = 1, if I3 =1		1
SET PM2, #I	PM2.0 = 1, if I0 =1 PM2.1 = 1, if I1 =1 PM2.2 = 1, if I2 =1 PM2.3 = 1, if I3 =1		1
SET PM1, #I	PM1.0 = 1, if I0 =1 PM1.1 = 1, if I1 =1 PM1.2 = 1, if I2 =1 PM1.3 = 1, if I3 =1		1
CLR EVFH, #I	EVFH.0 = 0, if I0 =1 EVFH.1 = 0, if I1 =1 EVFH.2 = 0, if I2 =1 EVFH.3 = 0, if I3 =1		1

CLR EVFL, #I	EVFL.0 = 0, if I0 =1 EVFL.1 = 0, if I1 =1 EVFL.2 = 0, if I2 =1 EVFL.3 = 0, if I3 =1		1
CLR HEFH, #I	HEFH.0 = 0, if I0 =1 HEFH.1 = 0, if I1 =1 HEFH.2 = 0, if I2 =1 HEFH.3 = 0, if I3 =1		1
CLR HEFL, #I	HEFL.0 = 0, if I0 =1 HEFL.1 = 0, if I1 =1 HEFL.2 = 0, if I2 =1 HEFL.3 = 0, if I3 =1		1
CLR IEFH, #I	IEFH.0 = 0, if I0 =1 IEFH.1 = 0, if I1 =1 IEFH.2 = 0, if I2 =1 IEFH.3 = 0, if I3 =1		1
CLR IEFL, #I	IEFL.0 = 0, if I0 =1 IEFL.1 = 0, if I1 =1 IEFL.2 = 0, if I2 =1 IEFL.3 = 0, if I3 =1		1
CLR PEFH, #I	PEFH.0 = 0, if I0 =1 PEFH.1 = 0, if I1 =1 PEFH.2 = 0, if I2 =1 PEFH.3 = 0, if I3 =1		1
CLR PEFL, #I	PEFL.0 = 0, if I0 =1 PEFL.1 = 0, if I1 =1 PEFL.2 = 0, if I2 =1 PEFL.3 = 0, if I3 =1		1
CLR CF	clear carry flag		1
CLR FLAG0, #I	FLAG0.3 and FLAG0.2 can not be cleared FLAG0.1= 0, if I1 =1 FLAG0.0 = 0, if I0 =1		1
CLR FLAG1, #I	FLAG1.0 = don't care, if I0 =1 FLAG1.1 = 0, if I1 =1 FLAG1.2 = 0, if I2 =1 FLAG1.3 = don't care, if I3 =1		1
CLR MR0, #I	MR0.0 = 0, if I0 =1 MR0.1 = 0, if I1 =1 MR0.2 = 0, if I2 =1 MR0.3 = 0, if I3 =1		1
CLR MR2, #I	MR2.0 = 0, if I0 =1 MR2.1 = 0, if I1 =1 MR2.2 = 0, if I2 =1 MR2.3 = 0, if I3 =1		1

CLR	SCR, #I	SCR.0 = 0, if I0 =1 SCR.1 = 0, if I1 =1 SCR.2 = 0, if I2 =1 SCR.3 = 0, if I3 =1		1
CLR	PSR1	PSR1.0 = 0 PSR1.1 = 0 PSR1.2 = 0 PSR1.3 = 0		1
CLR	PSR0	PSR0.0 = 0 PSR0.1 = 0 PSR0.2 = 0 PSR0.3 = 0		1
CLR	PM0, #I	PM0.0 = 0, if I0 =1 PM0.1 = 0, if I1 =1 PM0.2 = 0, if I2 =1 PM0.3 = 0, if I3 =1		1
CLR	PM2, #I	PM2.0 = 0, if I0 =1 PM2.1 = 0, if I1 =1 PM2.2 = 0, if I2 =1 PM2.3 = 0, if I3 =1		1
CLR	PM1, #I	PM1.0 = 0, if I0 =1 PM1.1 = 0, if I1 =1 PM1.2 = 0, if I2 =1 PM1.3 = 0, if I3 =1		1
Data Move				
MOV	WR, R	WR ← R		1
MOVA	WR, R	ACC, WR ← R	ZF	1
MOV	R, WR	R ← WR		1
MOVA	R, WR	ACC, R ← WR	ZF	1
MOV	R, #I	R ← I		1
MOVA	R, #I	ACC, R ← I	ZF	1
MOV	@RP0, #I	@RP0 ← I		1
MOVA	@RP0, #I	ACC, @RP0 ← I	ZF	1
MOV	@RP1, #I	@RP1 ← I		1
MOVA	@RP1, #I	ACC, @RP1 ← I	ZF	1
MOVA	ACC, #I	ACC ← I	ZF	1
MOV	R, ACC	R ← ACC		1
MOV	R, @RP0	R ← @RP0		1
MOV	R, @RP1	R ← @RP1		1

MOV	@RP0, R	@RP0 ← R		1
MOV	@RP1, R	@RP1 ← R		1
MOVA	R, @RP0	ACC, R ← @RP0	ZF	1
MOVA	R, @RP1	ACC, R ← @RP1	ZF	1
MOVA	@RP0, R	ACC, @RP0 ← R	ZF	1
MOVA	@RP1, R	ACC, @RP1 ← R	ZF	1
MOV	@RP1, @RP0	@RP1 ← @RP0		1
MOV	@RP0, @RP1	@RP0 ← @RP1		1
MOVA	@RP1, @RP0	ACC, @RP1 ← @RP0	ZF	1
MOVA	@RP0, @RP1	ACC, @RP0 ← @RP1	ZF	1
MOV	ACC, R	ACC ← R		1
MOV	R, @LUPC	R ← @LUPC		1
MOV	@RP0, @LUPC	@RP0 ← @LUPC		1
MOV	@RP1, @LUPC	@RP1 ← @LUPC		1
MOV	R, @RP0++	R ← @RP0 RP0 ← RP0 + 1		1
MOV	R, @RP1++	R ← @RP1 RP1 ← RP1 + 1		1
MOV	@RP0++, R	@RP0 ← R RP0 ← RP0 + 1		1
MOV	@RP1++, R	@RP1 ← R RP1 ← RP1 + 1		1
MOVA	R, @RP0++	ACC, R ← @RP0 RP0 ← RP0 + 1	ZF	1
MOVA	R, @RP1++	ACC, R ← @RP1 RP1 ← RP1 + 1	ZF	1
MOVA	@RP0++, R	ACC, @RP0 ← R RP0 ← RP0 + 1	ZF	1
MOVA	@RP1++, R	ACC, @RP1 ← R RP1 ← RP1 + 1	ZF	1
INC	RP0	RP0 ← RP0 + 1		1
INC	RP1	RP1 ← RP1 + 1		1
INC	LUPC	LUPC ← LUPC + 1		1

MOV @RP1++, @RP0++	@RP1 ← @RP0 RP0 ← RP0+1 RP1 ← RP1+1		1
MOV @RP0++, @RP1++	@RP0 ← @RP1 RP0 ← RP0+1 RP1 ← RP1+1		1
MOVA @RP1++, @RP0++	ACC, @RP1 ← @RP0 RP0 ← RP0+1 RP1 ← RP1+1	ZF	1
MOVA @RP0++, @RP1++	ACC, @RP0 ← @RP1 RP0 ← RP0+1 RP1 ← RP1+1	ZF	1
MOV R, @LUPC++	R ← @LUPC LUPC ← LUPC + 1		1
MOV @RP0++, @LUPC++	@RP0 ← @LUPC LUPC ← LUPC + 1 RP0 ← RP0 + 1		1
MOV @RP1++, @LUPC++	@RP1 ← @LUPC LUPC ← LUPC + 1 RP1 ← RP1 + 1		1
Special Register Write			
MOV TM0H, ACC	TM0H ← ACC		1
MOV TM0H, RL	TM0H ← RL		1
MOVA TM0H, RL	ACC, TM0H ← RL	ZF	1
MOV TM0H, #I	TM0H ← I		1
MOV TM0L ACC	TM0L ← ACC		1
MOV TM0L RL	TM0L ← RL		1
MOVA TM0L RL	ACC, TM0L ← RL	ZF	1
MOV TM0L #I	TM0L ← I		1
MOV TM1H, ACC	TM1H ← ACC		1
MOV TM1H, RL	TM1H ← RL		1
MOVA TM1H, RL	ACC, TM1H ← RL	ZF	1
MOV TM1H, #I	TM1H ← I		1
MOV TM1L, ACC	TM1L ← ACC		1
MOV TM1L, RL	TM1L ← RL		1
MOVA TM1L, RL	ACC, TM1L ← RL	ZF	1
MOV TM1L, #I	TM1L ← I		1

MOV	HEFH, ACC	HEFH ← ACC		1
MOV	HEFH, RL	HEFH ← RL		1
MOVA	HEFH, RL	ACC, HEFH ← RL	ZF	1
MOV	HEFH, #I	HEFH ← I		1
MOV	HEFL, ACC	HEFL ← ACC		1
MOV	HEFL, RL	HEFL ← RL		1
MOVA	HEFL, RL	ACC, HEFL ← RL	ZF	1
MOV	HEFL, #I	HEFL ← I		1
MOV	IEFH, ACC	IEFH ← ACC		1
MOV	IEFH, RL	IEFH ← RL		1
MOVA	IEFH, RL	ACC, IEFH ← RL	ZF	1
MOV	IEFH, #I	IEFH ← I		1
MOV	IEFL, ACC	IEFL ← ACC		1
MOV	IEFL, RL	IEFL ← RL		1
MOVA	IEFL, RL	ACC, IEFL ← RL	ZF	1
MOV	IEFL, #I	IEFL ← I		1
MOV	LDIV, ACC	LDIV ← ACC		1
MOV	LDIV, RL	LDIV ← RL		1
MOVA	LDIV, RL	ACC, LDIV ← RL	ZF	1
MOV	LDIV, #I	LDIV ← I		1
MOV	PEFH, ACC	PEFH ← ACC		1
MOV	PEFH, RL	PEFH ← RL		1
MOVA	PEFH, RL	ACC, PEFH ← RL	ZF	1
MOV	PEFH, #I	PEFH ← I		1
MOV	PEFL, ACC	PEFL ← ACC		1
MOV	PEFL, RL	PEFL ← RL		1
MOVA	PEFL, RL	ACC, PEFL ← RL	ZF	1
MOV	PEFL, #I	PEFL ← I		1
MOV	RP0M, ACC	RP0M ← ACC		1
MOV	RP0M, RL	RP0M ← RL		1
MOVA	RP0M, RL	ACC, RP0M ← RL	ZF	1

MOV	RP0M, #I	RP0M ← I		1
MOV	RP0L, ACC	RP0L ← ACC		1
MOV	RP0L, RL	RP0L ← RL		1
MOVA	RP0L, RL	ACC, RP0L ← RL	ZF	1
MOV	RP0L, #I	RP0L ← I		1
MOV	RP1M, ACC	RP1M ← ACC		1
MOV	RP1M, RL	RP1M ← RL		1
MOVA	RP1M, RL	ACC, RP1M ← RL	ZF	1
MOV	RP1M, #I	RP1M ← I		1
MOV	RP1L, ACC	RP1L ← ACC		1
MOV	RP1L, RL	RP1L ← RL		1
MOVA	RP1L, RL	ACC, RP1L ← RL	ZF	1
MOV	RP1L, #I	RP1L ← I		1
MOV	RP1H, ACC	RP1H ← ACC		1
MOV	RP1H, RL	RP1H ← RL		1
MOVA	RP1H, RL	ACC, RP1H ← RL	ZF	1
MOV	RP1H, #I	RP1H ← I		1
MOV	RP0H, ACC	RP0H ← ACC		1
MOV	RP0H, RL	RP0H ← RL		1
MOVA	RP0H, RL	ACC, RP0H ← RL	ZF	1
MOV	RP0H, #I	RP0H ← I		1
MOV	MLDH, ACC	MLDH ← ACC		1
MOV	MLDH, RL	MLDH ← RL		1
MOVA	MLDH, RL	ACC, MLDH ← RL	ZF	1
MOV	MLDH, #I	MLDH ← I		1
MOV	MLDL, ACC	MLDL ← ACC		1
MOV	MLDL, RL	MLDL ← RL		1
MOVA	MLDL, RL	ACC, MLDL ← RL	ZF	1
MOV	MLDL, #I	MLDL ← I		1
MOV	SPCH, ACC	SPCH ← ACC		1
MOV	SPCH, RL	SPCH ← RL		1
MOVA	SPCH, RL	ACC, SPCH ← RL	ZF	1

MOV	SPCH, #I	SPCH ← I		1
MOV	SPCL, ACC	SPCL ← ACC		1
MOV	SPCL, RL	SPCL ← RL		1
MOVA	SPCL, RL	ACC, SPCL ← RL	ZF	1
MOV	SPCL, #I	SPCL ← I		1
MOV	FLAG0, ACC	FLAG0 ← ACC, but FLAG0.3 and FLAG0.2 are write-inhibited		1
MOV	FLAG0, RL	FLAG0 ← RL, but FLAG0.3 and FLAG0.2 are write-inhibited		1
MOVA	FLAG0, RL	ACC, FLAG0 ← RL, but FLAG0.3 and FLAG0.2 are write-inhibited	ZF	1
MOV	FLAG0, #I	FLAG0 ← I, but FLAG0.3 and FLAG0.2 are write-inhibited		1
MOV	MR0, ACC	MR0 ← ACC		1
MOV	MR0, RL	MR0 ← RL		1
MOVA	MR0, RL	ACC, MR0 ← RL	ZF	1
MOV	MR0, #I	MR0 ← I		1
MOV	MR1, ACC	MR1 ← ACC		1
MOV	MR1, RL	MR1 ← RL		1
MOVA	MR1, RL	ACC, MR1 ← RL	ZF	1
MOV	MR1, #I	MR1 ← I		1
MOV	MR2, ACC	MR2 ← ACC		1
MOV	MR2, RL	MR2 ← RL		1
MOVA	MR2, RL	ACC, MR2 ← RL	ZF	1
MOV	MR2, #I	MR2 ← I		1
MOV	MR3, ACC	MR3 ← ACC		1
MOV	MR3, RL	MR3 ← RL		1
MOVA	MR3, RL	ACC, MR3 ← RL	ZF	1
MOV	MR3, #I	MR3 ← I		1
MOV	SCR, ACC	SCR ← ACC		1
MOV	SCR, RL	SCR ← RL		1
MOVA	SCR, RL	ACC, SCR ← RL	ZF	1

MOV	SCR, #I	SCR ← I		1
MOV	LCDM1, ACC	LCDM1 ← ACC		1
MOV	LCDM1, RL	LCDM1 ← RL		1
MOVA	LCDM1, RL	ACC, LCDM1 ← RL	ZF	1
MOV	LCDM1, #I	LCDM1 ← I		1
MOV	LCDM2, ACC	LCDM2 ← ACC		1
MOV	LCDM2, RL	LCDM2 ← RL		1
MOVA	LCDM2, RL	ACC, LCDM2 ← RL	ZF	1
MOV	LCDM2, #I	LCDM2 ← I		1
MOV	LUP1, ACC	LUP1 ← ACC		1
MOV	LUP1, RL	LUP1 ← RL		1
MOVA	LUP1, RL	ACC, LUP1 ← RL	ZF	1
MOV	LUP1, #I	LUP1 ← I		1
MOV	LUP0, ACC	LUP0 ← ACC		1
MOV	LUP0, RL	LUP0 ← RL		1
MOVA	LUP0, RL	ACC, LUP0 ← RL	ZF	1
MOV	LUP0, #I	LUP0 ← I		1
MOV	LUP3, ACC	LUP3 ← ACC		1
MOV	LUP3, RL	LUP3 ← RL		1
MOVA	LUP3, RL	ACC, LUP3 ← RL	ZF	1
MOV	LUP3, #I	LUP3 ← I		1
MOV	LUP2, ACC	LUP2 ← ACC		1
MOV	LUP2, RL	LUP2 ← RL		1
MOVA	LUP2, RL	ACC, LUP2 ← RL	ZF	1
MOV	LUP2, #I	LUP2 ← I		1
MOV	WRPAGE, ACC	WRPAGE ← ACC		1
MOV	WRPAGE, RL	WRPAGE ← RL		1
MOVA	WRPAGE, RL	ACC, WRPAGE ← RL	ZF	1
MOV	WRPAGE, #I	WRPAGE ← I		1
MOV	RAMPAGE, ACC	RAMPAGE ← ACC		1
MOV	RAMPAGE, RL	RAMPAGE ← RL		1
MOVA	RAMPAGE, RL	ACC, RAMPAGE ← RL	ZF	1

MOV	RAMPAGE, #I	RAMPAGE ← I		1
MOV	PM0, ACC	PM0 ← ACC		1
MOV	PM0, RL	PM0 ← RL		1
MOVA	PM0, RL	ACC, PM0 ← RL	ZF	1
MOV	PM0, #I	PM0 ← I		1
MOV	PM2, ACC	PM2 ← ACC		1
MOV	PM2, RL	PM2 ← RL		1
MOVA	PM2, RL	ACC, PM2 ← RL	ZF	1
MOV	PM2, #I	PM2 ← I		1
MOV	PM1, ACC	PM1 ← ACC		1
MOV	PM1, RL	PM1 ← RL		1
MOVA	PM1, RL	ACC, PM1 ← RL	ZF	1
MOV	PM1, #I	PM1 ← I		1
MOV	PORTB, ACC	PORTB ← ACC		1
MOV	PORTB, RL	PORTB ← RL		1
MOVA	PORTB, RL	ACC, PORTB ← RL	ZF	1
MOV	PORTB, #I	PORTB ← I		1
MOV	PORTA, ACC	PORTA ← ACC		1
MOV	PORTA, RL	PORTA ← RL		1
MOVA	PORTA, RL	ACC, PORTA ← RL	ZF	1
MOV	PORTA, #I	PORTA ← I		1
MOV	PORTE, ACC	PORTE ← ACC		1
MOV	PORTE, RL	PORTE ← RL		1
MOVA	PORTE, RL	ACC, PORTE ← RL	ZF	1
MOV	PORTE, #I	PORTE ← I		1
Special Register Read				
MOV	RL, TMC1H	RL ← TMC1H		1
MOVA	RL, TMC1H	ACC, RL ← TMC1H	ZF	1
MOV	RL, TMC1L	RL ← TMC1L		1
MOVA	RL, TMC1L	ACC, RL ← TMC1L	ZF	1
MOV	RL, EVFH	RL ← EVFH		1

MOVA	RL, EVFH	ACC, RL ← EVFH	ZF	1
MOV	RL, EVFL	RL ← EVFL		1
MOVA	RL, EVFL	ACC, RL ← EVFL	ZF	1
MOV	RL, HEFH	RL ← HEFH		1
MOVA	RL, HEFH	ACC, RL ← HEFH	ZF	1
MOV	RL, HEFL	RL ← HEFL		1
MOVA	RL, HEFL	ACC, RL ← HEFL	ZF	1
MOV	RL, IEFH	RL ← IEFH		1
MOVA	RL, IEFH	ACC, RL ← IEFH	ZF	1
MOV	RL, IEFL	RL ← IEFL		1
MOVA	RL, IEFL	ACC, RL ← IEFL	ZF	1
MOV	RL, HCFH	RL ← HCFH		1
MOVA	RL, HCFH	ACC, RL ← HCFH	ZF	1
MOV	RL, HCFL	RL ← HCFL		1
MOVA	RL, HCFL	ACC, RL ← HCFL	ZF	1
MOV	RL, PEFH	RL ← PEFH		1
MOVA	RL, PEFH	ACC, RL ← PEFH	ZF	1
MOV	RL, PEFL	RL ← PEFL		1
MOVA	RL, PEFL	ACC, RL ← PEFL	ZF	1
MOV	RL, RP0M	RL ← RP0M		1
MOVA	RL, RP0M	ACC, RL ← RP0M	ZF	1
MOV	RL, RP0L	RL ← RP0L		1
MOVA	RL, RP0L	ACC, RL ← RP0L	ZF	1
MOV	RL, RP1M	RL ← RP1M		1
MOVA	RL, RP1M	ACC, RL ← RP1M	ZF	1
MOV	RL, RP1L	RL ← RP1L		1
MOVA	RL, RP1L	ACC, RL ← RP1L	ZF	1
MOV	RL, RP1H	RL ← RP1H		1
MOVA	RL, RP1H	ACC, RL ← RP1H	ZF	1
MOV	RL, RP0H	RL ← RP0H		1
MOVA	RL, RP0H	ACC, RL ← RP0H	ZF	1
MOV	RL, CF	RL ← CF		1

MOVA	RL, CF	ACC, RL ← CF	ZF	1
MOV	RL, FLAG0	RL ← FLAG0		1
MOVA	RL, FLAG0	ACC, RL ← FLAG0	ZF	1
MOV	RL, MR0	RL ← MR0		1
MOVA	RL, MR0	ACC, RL ← MR0	ZF	1
MOV	RL, MR2	RL ← MR2		1
MOVA	RL, MR2	ACC, RL ← MR2	ZF	1
MOV	RL, SCR	RL ← SCR		1
MOVA	RL, SCR	ACC, RL ← SCR	ZF	1
MOV	RL, LUP0	RL ← LUP0		1
MOVA	RL, LUP0	ACC, RL ← LUP0	ZF	1
MOV	RL, LUP1	RL ← LUP1		1
MOVA	RL, LUP1	ACC, RL ← LUP1	ZF	1
MOV	RL, LUP2	RL ← LUP2		1
MOVA	RL, LUP2	ACC, RL ← LUP2	ZF	1
MOV	RL, LUP3	RL ← LUP3		1
MOVA	RL, LUP3	ACC, RL ← LUP3	ZF	1
MOV	RL, LUC	RL ← LUC		1
MOVA	RL, LUC	ACC, RL ← LUC	ZF	1
MOV	RL, WRPAGE	RL ← WRPAGE		1
MOVA	RL, WRPAGE	ACC, RL ← WRPAGE	ZF	1
MOV	RL, RAMPAGE	RL ← RAMPAGE		1
MOVA	RL, RAMPAGE	ACC, RL ← RAMPAGE	ZF	1
MOV	RL, PM0	RL ← PM0		1
MOVA	RL, PM0	ACC, RL ← PM0	ZF	1
MOV	RL, PSR1	RL ← PSR1		1
MOVA	RL, PSR1	ACC, RL ← PSR1	ZF	1
MOV	RL, PSR0	RL ← PSR0		1
MOVA	RL, PSR0	ACC, RL ← PSR0	ZF	1
MOV	RL, PM2	RL ← PM2		1
MOVA	RL, PM2	ACC, RL ← PM2	ZF	1

MOV RL, PM1	RL ← PM1		1
MOVA RL, PM1	ACC, RL ← PM1	ZF	1
MOV RL, PORTB	RL ← PORTB		1
MOVA RL, PORTB	ACC, RL ← PORTB	ZF	1
MOV RL, PORTA	RL ← PORTA		1
MOVA RL, PORTA	ACC, RL ← PORTA	ZF	1
MOV RL, PORTD	RL ← PORTD		1
MOVA RL, PORTD	ACC, RL ← PORTD	ZF	1
MOV RL, PORTC	RL ← PORTC		1
MOVA RL, PORTC	ACC, RL ← PORTC	ZF	1
Special Register Pair Write			
MOV HEF, #I	HEF ← I		1
MOV IEF, #I	IEF ← I		1
Others			
NOP	No opperation		1
HOLD	Enter the hold mode		1
RTN	PC ← STACK		1
Pseudo Instruction			
EN INT	MR2.0 ← 1		1
DIS INT	MR2.0 ← 0		1
LCDON	MR0.1 ← 1		1
LCDOFF	MR0.1 ← 0		1
CLR WDT	FLAG1.1 ← 0		1
CLR DIV	FLAG1.2 ← 0		1

Winbond Electronics Corp.

Headquarters
No. 4, Creation Rd. III,
Science-Based Industrial Park,
Hsinchu, Taiwan
TEL: 886-3-5770066
FAX: 886-3-5792697
http://www.winbond.com.tw/
Voice & Fax-on-demand: 886-2-7197006

Taipei Office
11F, No. 115, Sec. 3, Min-Sheng East Rd.,
Taipei, Taiwan
TEL: 886-2-7190505
FAX: 886-2-7197502

Winbond Electronics (H.K.) Ltd.
Rm. 803, World Trade Square, Tower II,
123 Hoi Bun Rd., Kwun Tong,
Kowloon, Hong Kong
TEL: 852-27513100
FAX: 852-27552064

Winbond Electronics North America Corp.
Winbond Memory Lab.
Winbond Microelectronics Corp.
Winbond Systems Lab.
2730 Orchard Parkway, San Jose,
CA 95134, U.S.A.
TEL: 1-408-9436666
FAX: 1-408-9436668

Publication Release Date: March 1999