ABOV SEMICONDUCTOR Co., Ltd.
8-BIT SINGLE-CHIP MICROCONTROLLERS

MC80F7208
MC80C7208

User's Manual (Ver. 1.15)

ABOV semiconductor

MC80F7208
MC80C7208

CMOS SINGLE-CHIP 8-BIT MICROCONTROLLER WITH LCD CONTROLLER/DRIVER

1. OVERVIEW

1.1 Description

The MC80X7208 is an advanced CMOS 8-bit microcontroller with 8K bytes of FLASH/ROM. This device is one of the HMS800 and MC800 families and a powerful microcontroller which provides a high flexibility and cost effective solution to many LCD applications. The MC80X7208 provides the following standard features: 8K bytes of FLASH/ROM, 256 bytes of RAM, 27 nibbles of segment LCD display RAM, 8/16-bit timer/counter, 8-bit A/D converter, 7-bit watch dog timer, 8-bit remote control carrier generator, 21-bit watch timer with 7-bit auto reload counter, on-chip oscillator and clock circuitry. In addition, this device supports power down modes to reduce power consumption. So the MC80X7208 is the best controller solution in system which charatered LCD display, ADC and battery backup. This document is only explained for the base of MC80X7208, the compact functions are same as below.

Device name	Memory (Bytes)		ADC	PWM	REMOUT	UART	ISP	LCD	Operating Voltage	Package
	ROM	RAM								
MC80F7208[1]	8K	256	6ch.	1ch.	1ch.	1ch.	O	24SEG x 4COM	2.2 ~ 5.5V	64MQFP 64LQFP
MC80C7208[1]	8K	256	6ch.	1ch.	1ch.	-	-	24SEG x 4COM	1.8 ~ 5.5V	64MQFP 64LQFP

1. F: FLASH ROM, C:MASK ROM

1.2 Features

- **8K Bytes On-chip ROM**
- **FLASH Memory**
  - Endurance : 1000 cycles
  - Data Retention : 10 years
- **256 Bytes On-chip Data RAM**
- **27 Nibbles Display RAM**
- **Instruction Cycle Time:**
  - 333ns at 12MHz (2 cycle NOP instruction)
- **LCD display/controller (LCDC)**
  - Static Mode (27Seg × 1Com, 1/3 Bias)
  - 1/2 Duty Mode (26Seg × 2Com, 1/3 Bias)
  - 1/3 Duty Mode (25Seg × 3Com, 1/3 Bias)
  - 1/4 Duty Mode (24Seg × 4Com, 1/3 Bias)
- **Four 8-bit Timer/Counter**
  (They can be used as two 16-bit Timer/Counter)
- **One 7-bit Watch Dog Timer**
- **One 21-bit Watch Timer**
  - 1 minute interrupt available
- **One 8-bit Basic Interval Timer**
- **One 6-bit Buzzer Driving Port**
- **Dual Clock Operation**
  - Main Clock : 1MHz ~ 12MHz
  - Sub Clock : 32.768kHz
- **Main Clock Oscillation**
  - Crystal
  - Ceramic Resonator
  - External RC/R Oscillator(Built-in Capacitor)
- **Operating Temperature : -40~85 °C**
- **Built-in Noise Immunity Circuit**
  - Noise Filter
  - Low Voltage Detector(LVD) : 1.8V(Typ.)
  - Power Fail Detector(PFDL or PFDH) :

1.8V(Typ.) or 2.65V(Typ.)

- **Power Down Mode**
  - Main Clock : STOP, SLEEP mode
- **1MHz to 12MHz Wide Operating Frequency**
  - QFP type:1MHz ~ 12MHz
  - COB type:400kHz ~ 12MHz
- **LCD Display Voltage Booster**
- **39 Programmable I/O Pins**
  I/O:15, I/O with SEG:16, I with SEG:8
- **6-channel 8-bit On-chip A/D Converter**
- **One 10-bit High Speed PWM Output**
- **13 Interrupt sources**
  - 2 External interrupts (INT0 ~ 1)
  - 11 Internal interrupts (WT, WDT, BIT, ADC, Carrier Generator, TIMER*4, Keyscan, RESET)
- **One Universal Asynchronous Receiver/Transmitter (UART) at FLASH MCU**

MC80F7208	**One UART**
MC80C7208	**N/A**

- **1.8V to 5.5V Wide Operating Voltage Range Except FLASH MCU**

MC80F7208	**2.2V to 5.5V**
MC80C7208	**1.8V to 5.5V**

- **Four Key Scan Interrupts**
- **One Carrier Generator for Remote Controller**
- **64MQFP, 64LQFP Package Types**
  - Available Pb free package

1.3 Development Tools

The MC80X7208 is supported by a full-featured macro assembler, an in-circuit emulator CHOICE-Dr.™ and FLASH programmers. There are two different type of programmers such as single type and gang type. For mode detail, Macro assembler operates under the MS-Windows 95 and upversioned Windows OS.

Please contact sales part of ABOV semiconductor.

Software	- MS-Windows based assembler - MS-Windows based Debugger - HMS800 C compiler
Hardware (Emulator)	- CHOICE-Dr. - CHOICE-Dr. EVA80C7x B/D
POD Name	- POD80C72D-64LQ-1010 - POD80C72D-64MQ-1420
FLASH Writer	- PGM Plus USB (Single writer) - Stand Alone GANG4 USB (Gang writer) - CHOICE - SIGMA II(Single writer)

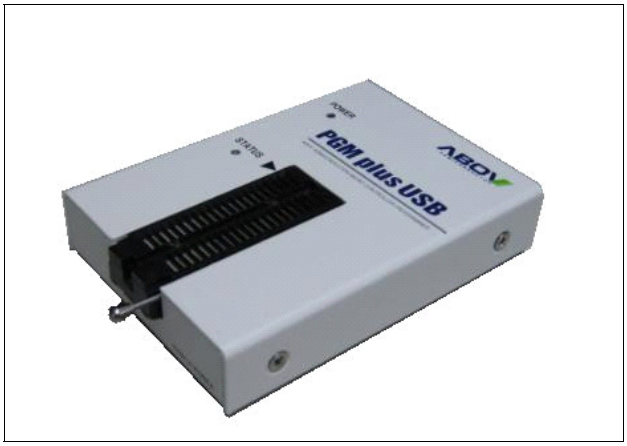


Figure 1-1 PGM plus USB (Single Writer)

Figure 1-2 Choice-Dr. (Emulator, USB Interface)

Figure 1-3 Stand Alone Gang4 USB (Gang Writer)

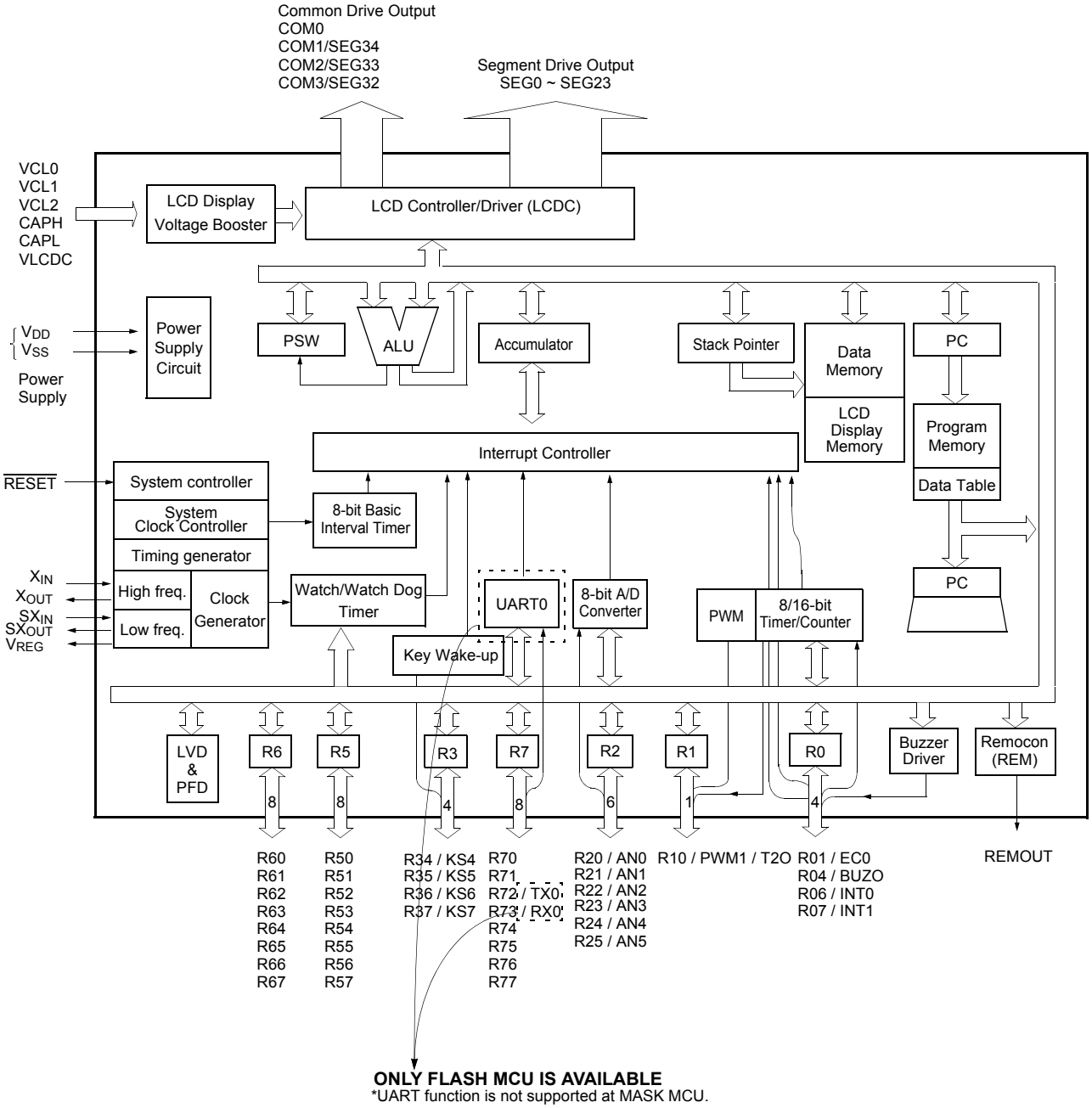
1.4 Ordering Information

	Device name	ROM Size	RAM size	Package
FLASH ROM version	MC80F7208Q MC80F7208L	8K bytes FLASH 8K bytes FLASH	256 bytes 256 bytes	64MQFP 64LQFP
Mask ROM version	MC80C7208 Q MC80C7208 L	8K bytes 8K bytes	256 bytes 256 bytes	64MQFP 64LQFP

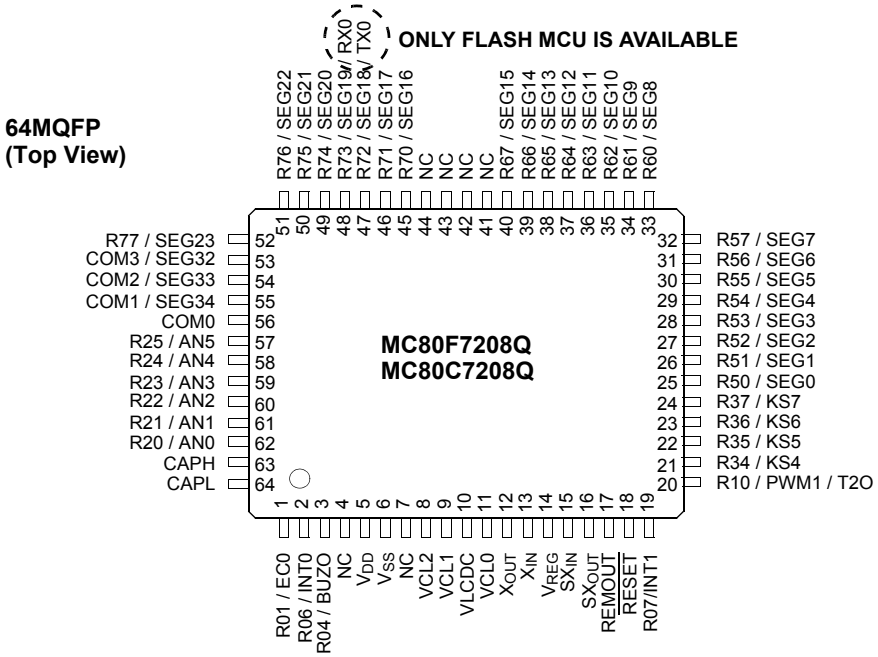
- Pb free package;

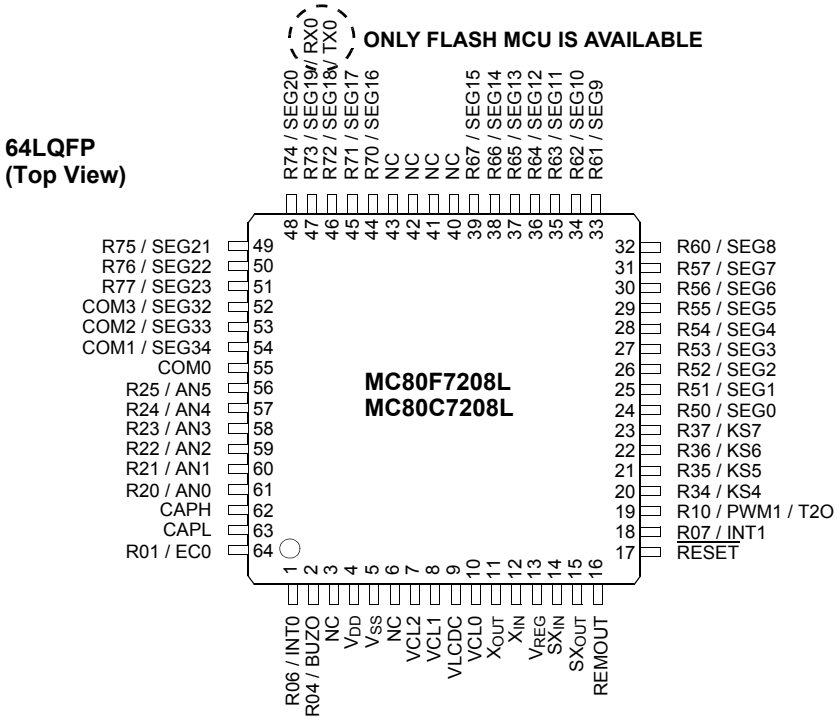
The “P” suffix will be added at original part number.
For example; MC80C7208 Q(Normal package), MC80C7208 QP(Pb free package)

2. BLOCK DIAGRAM



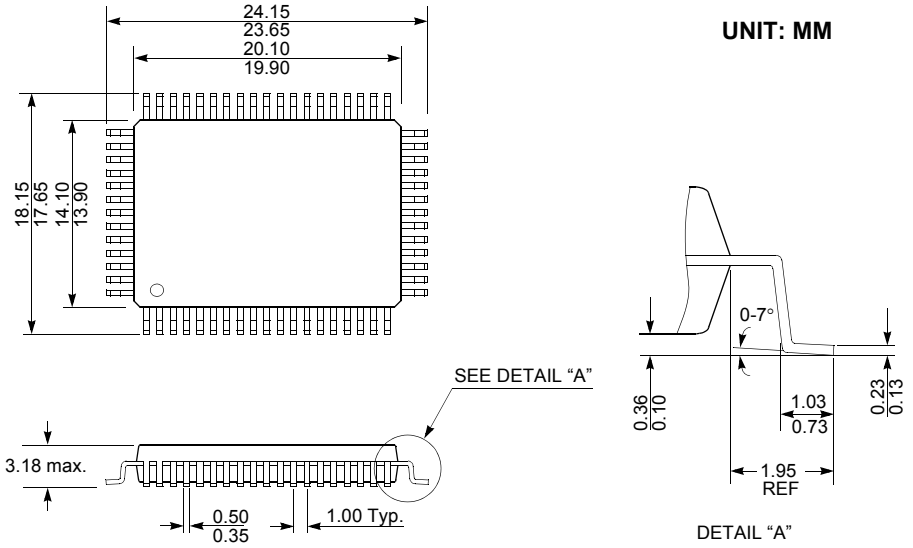
3. PIN ASSIGNMENT



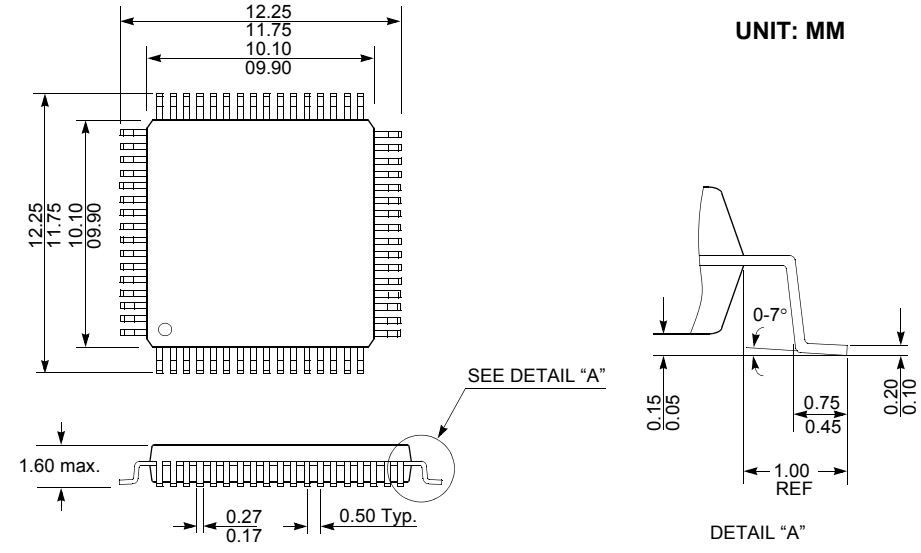


4. PACKAGE DIAGRAM

64MQFP



64LQFP



5. PIN FUNCTION

V_{DD}: Supply Voltage.

V_{SS}: Circuit ground.

$\overline{RESET}$: Reset the MCU Reset.

REMOUT: Signal output of an infrared remote controller.

X_{IN}: Input to the inverting oscillator amplifier and input to the internal main clock operating circuit.

X_{OUT}: Output from the inverting oscillator amplifier.

SX_{IN}: Input to the internal sub system clock operating circuit.

SX_{OUT}: Output from the inverting subsystem oscillator amplifier.

VCL0~VCL2: Power supply pins(bias) for the LCD driver. The voltage on each pin is VCL2 > VCL1 > VCL0. See "19. LCD DRIVER" on page 76 for details.

VLCDC: Booster reference voltage to drive LCD

CAPH, CAPL: Booster capacitor for voltage booster circuit to drive LCD.

V_{REG}: Output of the voltage regulator for the sub clock oscillation circuit. Connect external 0.1uF capacitor to this pin when using the sub system clock.

SEG0~SEG23, SEG32~SEG34: Segment signal output pins for the LCD display. See "19. LCD DRIVER" on page 76 for details. Also SEG0~SEG23 are shared with a normal I/O ports and SEG32~34 are multiplexed with COM3~COM1.

COM0~COM3: Common signal output pins for the LCD display. See "19. LCD DRIVER" on page 76 for details.

SEG32~SEG34 and COM1~COM3 are selected by LCDD[1:0] of the LCR register.

Port pin	Alternate function
COM0	-
COM1	SEG34
COM2	SEG33
COM3	SEG32

R01, R04, R06, R07: R0 is a 4-bit CMOS bidirectional I/O port. R0 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R0 serves the functions of the various following special features.

Port pin	Alternate function
R01	EC0 (Timer 0 Event Count Input)
R04	BUZO (Buzzer Output)
R06	INT0 (External Interrupt 0 Request Input)
R07	INT1 (External Interrupt 1 Request input)

R10: R1 is an 1-bit CMOS bidirectional I/O port. R1 pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs or schmitt trigger inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R1 serves the function of the following special feature.

Port pin	Alternate function
R10	PWM1/T2O (Timer3 PWM Output / Timer2 Output)

R20~R25: R2 is a 6-bit CMOS bidirectional I/O port. Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R2 serves the functions of the various following special features.

Port pin	Alternate function
R20	AN0 (Analog Input Port0)
R21	AN1 (Analog Input Port1)
R22	AN2 (Analog Input Port2)
R23	AN3 (Analog Input Port3)
R24	AN4 (Analog Input Port4)
R25	AN5 (Analog Input Port5)

R34~R37: R3 is a 4-bit CMOS bidirectional I/O port. Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. Also, pull-up resistors and open-drain outputs can be assigned by software.

In addition, R3 serves the functions of the various following special features.

Port pin	Alternate function
R34	KS4 (Key scan input 4)
R35	KS5 (Key scan input 5)
R36	KS6 (Key scan input 6)
R37	KS7 (Key scan input 7)

R50~R57: R5 is an 8-bit CMOS bidirectional I/O port or LCD segment output. Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. And each pins can also be set in segment output mode in 1-bit units by R5PSR Register.

Port pin	Alternate function
R50	SEG0 (Segment Output 0)
R51	SEG1 (Segment Output 1)
R52	SEG2 (Segment Output 2)
R53	SEG3 (Segment Output 3)
R54	SEG4 (Segment Output 4)
R55	SEG5 (Segment Output 5)
R56	SEG6 (Segment Output 6)
R57	SEG7 (Segment Output 7)

R60~R67: R6 is an 8-bit CMOS bidirectional I/O port or LCD segment output. Each pins 1 or 0 written to the Port Direction Register can be used as outputs or inputs. And each pins can also be set in segment output mode in 1-bit units by R6PSR Register.

Port pin	Alternate function
R60	SEG8 (Segment Output 8)
R61	SEG9 (Segment Output 9)
R62	SEG10 (Segment Output 10)
R63	SEG11 (Segment Output 11)
R64	SEG12 (Segment Output 12)
R65	SEG13 (Segment Output 13)
R66	SEG14 (Segment Output 14)
R67	SEG15 (Segment Output 15)

R70~R77: R7 is an 8-bit CMOS input port or LCD segment output. Each pins can be set in digital input or segment output mode in 1-bit units by R7PSR Register.

Port pin	Alternate function
R70	SEG16 (Segment Output 0)
R71	SEG17 (Segment Output 1)
R72	SEG18 (Segment Output 2)/TX0
R73	SEG19 (Segment Output 3)/RX0
R74	SEG20 (Segment Output 4)
R75	SEG21 (Segment Output 5)
R76	SEG22 (Segment Output 6)
R77	SEG23 (Segment Output 7)

Especially, R72 and R73 are shared with TX0 and RX0 of UART in the FLASH MCU. See the "Figure 26-3 Example Flow of Reset flow by Power fail" on page 101 for details.

PIN NAME	Pin No.		Primary Function		Secondary Function		State @ Reset	State @ STOP
	MC80X7 208Q	MC80X7 208L	I/O	Description	I/O	Description		
V_{DD}	5	4	-	Supply Voltage	-	-	-	-
V_{SS}	6	5	-	Circuit Ground	-	-	-	-
$\overline{RESET}$	18	17	I	Reset (low active)	-	-	'L' input	'H' input
REMOUT	17	16	O	Remocon output	-	-	'L' output	
X_{IN}, X_{OUT}	13,12	12,11	I,O	Main clock oscillator	-	-	Oscillation	'L', 'H'
SX_{IN}, SX_{OUT}	15,16	14,15	I,O	Sub clock oscillator	-	-	Oscillation	
VCL0~VCL2	11,9,8	10,8,7	-	LCD drive voltage	-	-	Internal VCL0 Connected	State of before STOP
VLCDC	10	9	-	LCD drive voltage booster reference	-	-	-	-

Table 5-1 Port Function Description

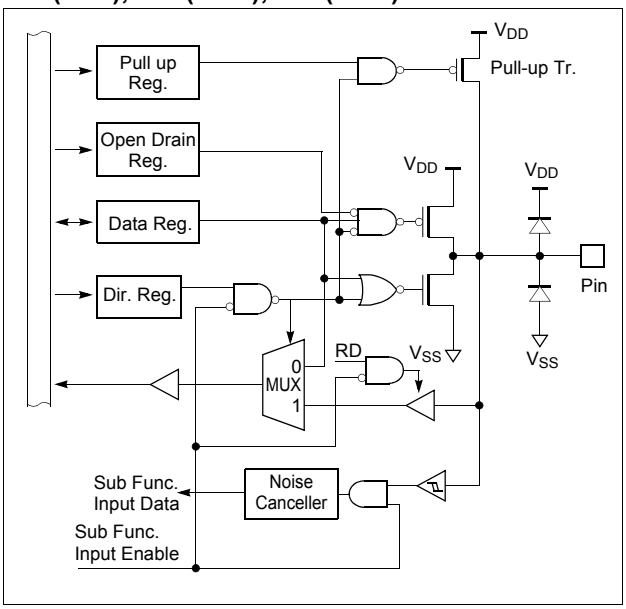
PIN NAME	Pin No.		Primary Function		Secondary Function		State @ Reset	State @ STOP
	MC80X7208Q	MC80X7208L	I/O	Description	I/O	Description		
CAPH,CAPL	63,64	62,63	-	LCD drive voltage booster capacitor	-	-	Internal VCL0 Connected	State of before STOP
V_{REG}	14	13	-	Sub clock voltage	-	-	-	-
R50/SEG0 ~ R57/SEG7	25~32	24~31	I/O	General I/O port	O	LCD segment output	Input port	State of before STOP
R60/SEG8 ~ R67/SEG15	33~40	32~39	I/O	General I/O port	O	LCD segment output	Input port	State of before STOP
R70/SEG16 ~ R77/SEG23	45~52	44~51	I	General Input	O	LCD segment output	Input port	State of before STOP
COM0	56	55	O	LCD common output	-	-	Common output	
COM1/SEG34 COM2/SEG33 COM3/SEG32	55~53	54~52	O	LCD common output.	O	LCD segment output	Common output	State of before STOP
R01/EC0	1	64	I/O	General I/O port	I	Event counter input	Input port	
R04/BUZO	3	2	I/O		O	Buzzer output		
R06/INT0	2	1	I/O		I	Interrupt Input		
R07/INT1	19	18	I/O		I	Interrupt Input		
R10/PWM1/ T2O	20	19	I/O		O	Timer3 PWM Output Timer2 Output		
R20/AN0 ~ R25/AN5	62~57	61~56	I/O		I	A/D Converter Analog Input		
R34/KS4 ~ R37/KS7	21~24	20~23	I/O		I	Key Wake-up Input		
NC	4, 7, 41~44	3, 6, 40~43	-	-	-	No Connection	-	-
R73/SEG19/ *RX0*	48	47	I	General Input/LCD Segment Output	I	UART0 Data Input	Input port	State of before STOP
R72/SEG18/ *TX0*	47	46	I	General Input/LCD Segment Output	O	UART0 Data Output		

Table 5-1 Port Function Description

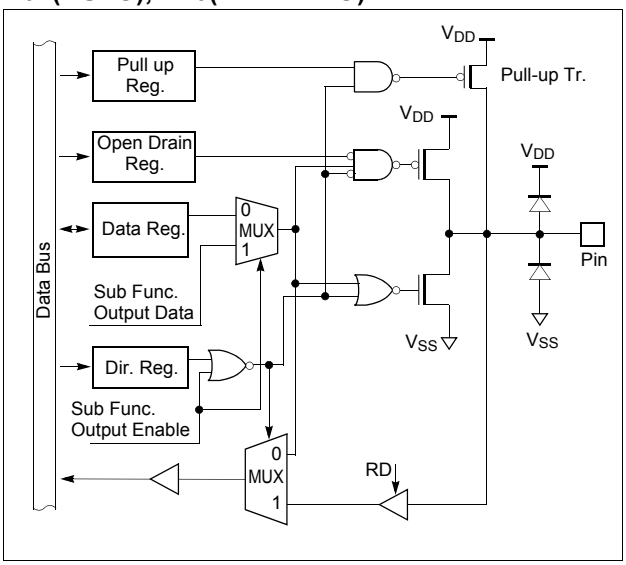
ONLY FLASH MCU IS AVAILABLE

6. PORT STRUCTURES

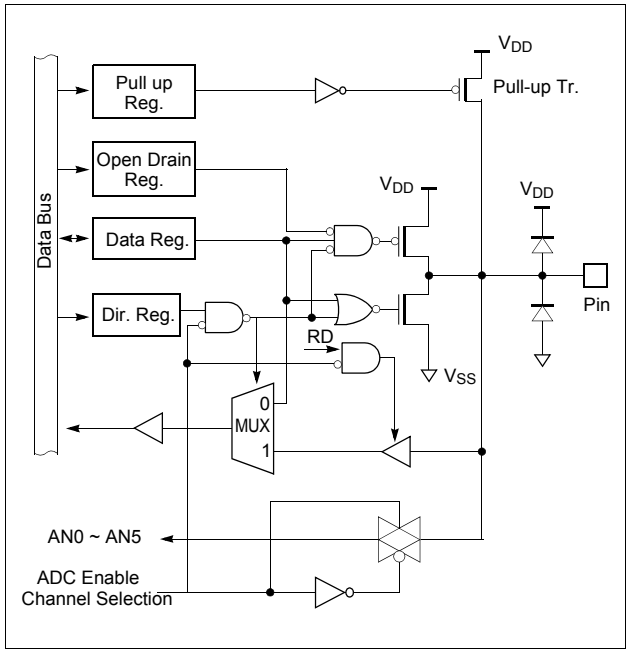
R01(EC0), R06(INT0), R07(INT1)



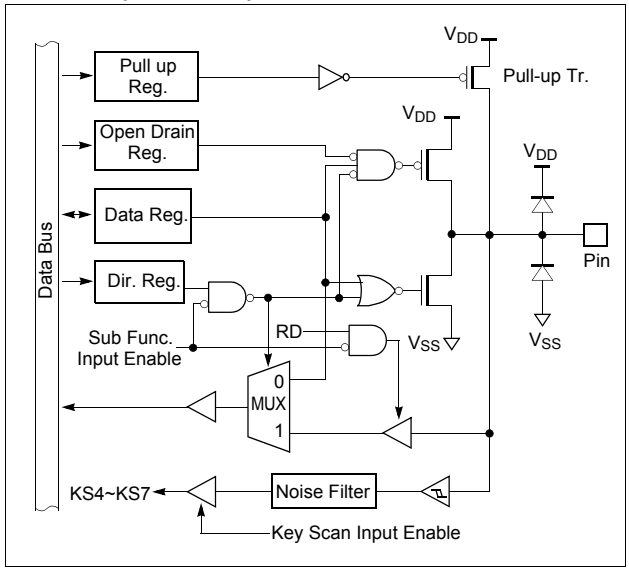
R04(BUZO), R10(PWM1/T2O)



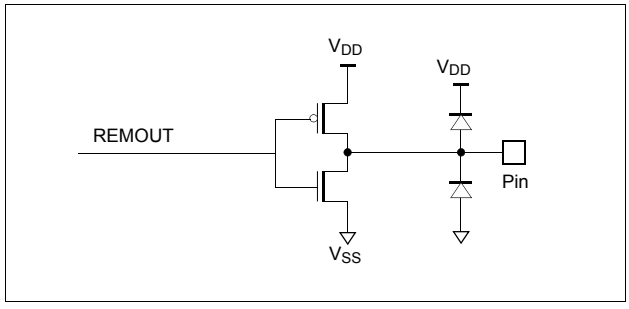
R20~R25/AN0~AN5



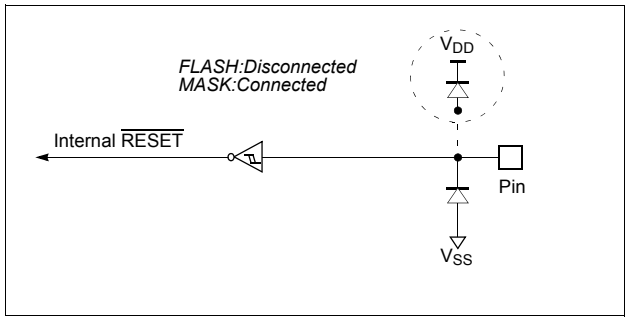
R34~R37(KS4~KS7)



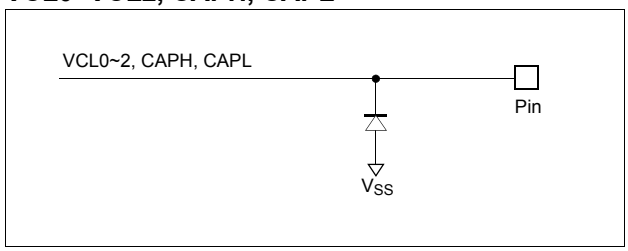
REMOUT



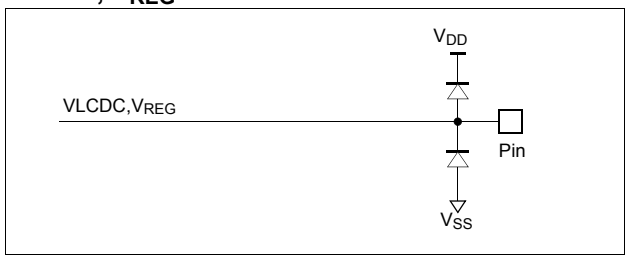
$\overline{\text{RESET}}$



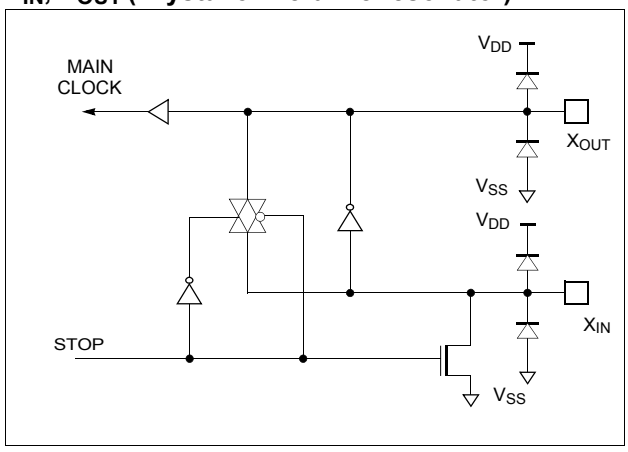
VCL0~VCL2, CAPH, CAPL



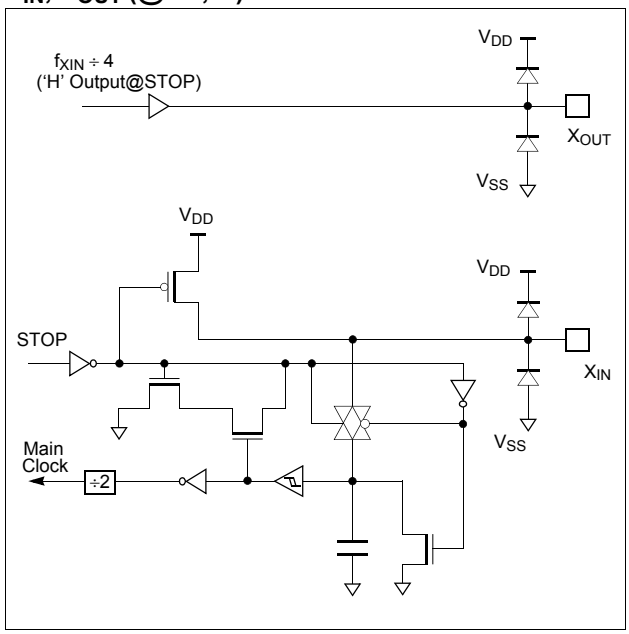
VLCDC, V_{REG}



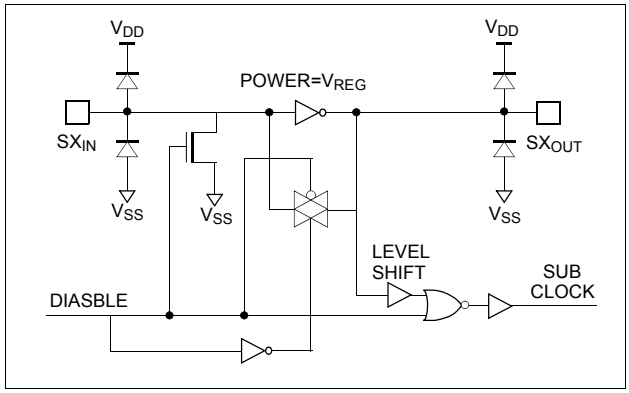
X_{IN}, X_{OUT} (Crystal or Ceramic resonator)



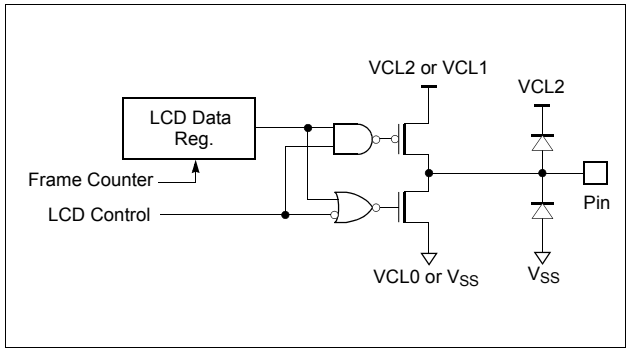
X_{IN}, X_{OUT} (@RC, R)



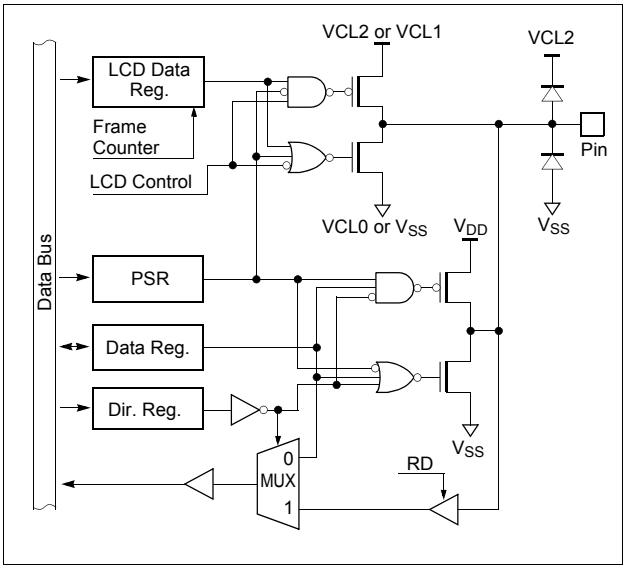
SX_{IN}, SX_{OUT}



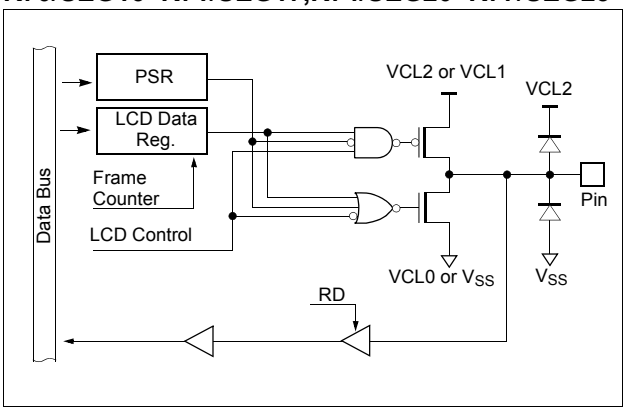
COM0, COM3/SEG32~COM1/SEG34



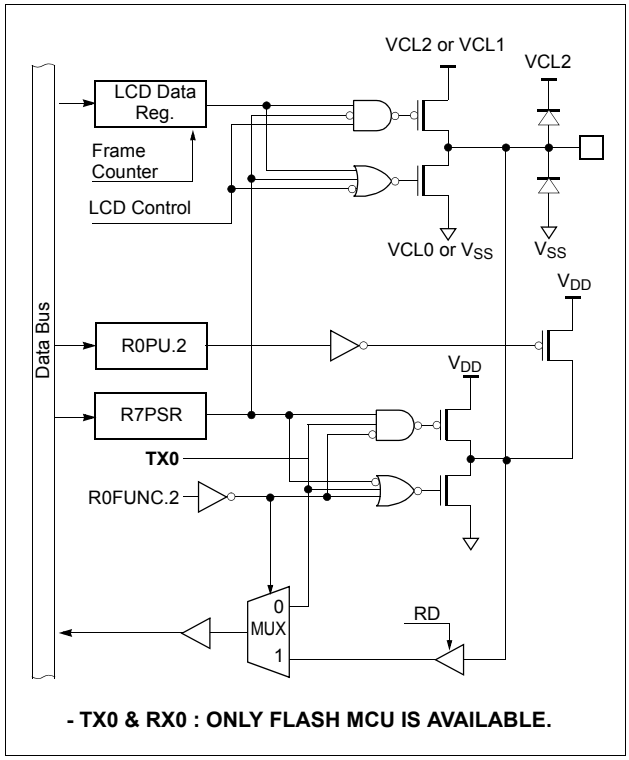
R50/SEG0~R57/SEG7, R60/SEG8~R67/SEG15



R70/SEG16~R71/SEG17,R74/SEG20~R77/SEG23

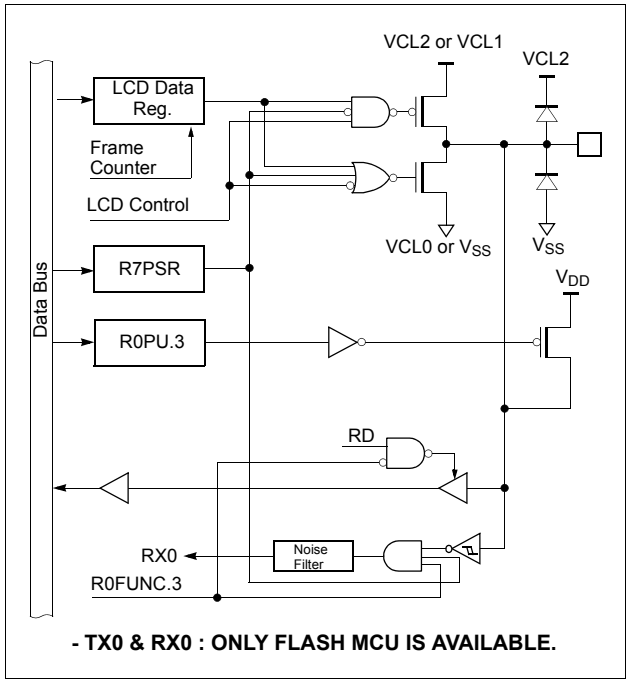


R72/SEG18/TX0



- TX0 & RX0 : ONLY FLASH MCU IS AVAILABLE.

R73/SEG19/RX0



- TX0 & RX0 : ONLY FLASH MCU IS AVAILABLE.

7. ELECTRICAL CHARACTERISTICS

7.1 Absolute Maximum Ratings

Supply voltage ..... -0.3 to +6.0 V

Storage Temperature ..... -45 to +125 °C

Voltage on any pin with respect to Ground (V_{SS}) ..... -0.3 to V_{DD}+0.3

Maximum current sunk by (I_{OL} per I/O Pin) .....20 mA

Maximum output current sourced by (I_{OH} per I/O Pin) .....15 mA

Maximum current (ΣI_{OL}) .....100 mA

Maximum current (ΣI_{OH})..... 60 mA

Note: *Stresses above those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at any other conditions above those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.*

7.2 Recommended Operating Conditions

Parameter	Symbol	Condition	Specifications			Unit
			Min.	Typ.	Max.	
Supply Voltage	V_{DD}	f_{MAIN}=4MHz	1.8	-	5.5	V
Main Operating Frequency	f_{MAIN}	V_{DD}[1]=1.8~5.5V V_{DD}[2]=2.2~5.5V	1[3]	-	4.0	MHz
		V_{DD}=4.5~5.5V	1[3]	-	12.0	
Sub Operating Frequency	f_{SUB}	V_{DD}=V_{DD}	-	32.768	-	kHz
Operating Temperature	T_{OPR}		-40	-	85	°C

1. MASK MCU Operating Voltage.
2. FLASH MCU Operating Voltage.
3. QFP type is 1MHz, COBtype is 0.4MHz.

7.3 DC Electrical Characteristics

(T_A= -40~85°C, V_{DD}=1.8~5.5V, V_{SS}=0V)

Parameter	Symbol	Pin / Condition	Specifications			Unit
			Min.	Typ.	Max.	
Input High Voltage	V_{IH1}	R0~R7	0.7V_{DD}	-	V_{DD}+0.3	V
	V_{IH2}	$\overline{RESET}$, X_{IN}, INT0~1, EC0	0.8V_{DD}	-	V_{DD}+0.3	
	V_{IH3}	SX_{IN}	0.8V_{REG}	-	V_{REG}+0.3	
Input Low Voltage	V_{IL1}	R0~R7	-0.3	-	0.3V_{DD}	V
	V_{IL2}	$\overline{RESET}$, X_{IN}, INT0~1, EC0	-0.3	-	0.2V_{DD}	
	V_{IL3}	SX_{IN}	-0.3	-	0.1V_{DD}	
Output High Voltage	V_{OH1}	R0~R3 (V_{DD}=4.5V, I_{OH1}=-1.6mA)	V_{DD}-0.5	-	-	V
	V_{OH2}	R5~R6 (V_{DD}=4.5V, I_{OH2}=-1.6mA)	V_{DD}-1.0	-	-	
	V_{OH3}[1]	SEG0~23, COM0~3 (V_{DD}=4.5V, VCL2~0=3V, I_{OH3}=-15μA)	VCL2-0.4	-	-	
Output Low Voltage	V_{OL1}	R0~R3 (V_{DD}=4.5V, I_{OL1}=1.6mA)	-	-	0.35	V
	V_{OL2}	R5~R6 (V_{DD}=4.5V, I_{OL2}=1.6mA)			0.4	
	V_{OL3}[2]	SEG0~23, COM0~3 (V_{DD}=4.5V, VCL2~0=3V, I_{OL3}=15μA)			0.12	
Input High Leakage Current	I_{IH}	All input pins including R5~R7 and COM0~COM3 (V_{IN}=V_{DD})	-	-	1	μA
Input Low Leakage Current	I_{IL}	All input pins including R5~R7 and COM0~COM3 (V_{IN}=V_{ss})	-1	-	-	
Output High Current	I_{OH}	REMOUT (V_{DD}=3.0V, V_{OH}=V_{DD}-1V)	-20	-	-5	mA
Output Low Current	I_{OL}	REMOUT (V_{DD}=3.0V, V_{OL}=1V)	0.3	-	3.0	
V_{REG} Voltage	V_{REG}	V_{REG} (V_{DD}=3.0V)	1.5	-	2.6	V
PFD Voltage	V_{PFDH}	V_{DD} (PFDLS<1:0>=00)	2.2	2.8	3.4	
	V_{PFDL}		1.3	1.9	2.5	
Low Voltage Detector	LVD	V_{DD} (TA=25°C)	-	-	2.2(1.8)[3]	V
Hysteresis	VT+ ~ VT-	$\overline{RESET}$, INT0~1, EC0 (V_{DD}=5V)	0.2V_{DD}	-	0.8V_{DD}	V
Pull-up Current	I_{PU}	R0~R3 (V_{DD}=3.0V, V_{PIN}=0V)	20	-	60	μA
Current dissipation in active mode[4]	I_{DD}	V_{DD}(f_{MAIN}=8MHz, V_{DD}=5.5V, f_{SUB}=0 V)	-	5	15	mA
Current dissipation in sleep mode[5]	I_{SLEEP}	V_{DD} (f_{MAIN}=8MHz, V_{DD}=5.5V, f_{SUB}=0 V)	-	2	4	
Current dissipation in stop mode	I_{STOP}	f_{MAIN}=off, V_{DD}=5.5V, f_{SUB}=0 V	-	3	7	μA
External RC Oscillator Frequency	f_{RC-OSC}	f_{XOUT} = f_{RC-OSC}÷4 (R = 10 kΩ, C = 30pF, V_{DD} = 5.5 V)	1	2	3	MHz
	f_{R-OSC}	f_{XOUT} = f_{R-OSC}÷4 (R = 10 kΩ, V_{DD} = 5.5 V)	2	3	4	

1. V_{OH3} is the voltage when VCL2, VCL1 and VCL0 are supplied at pads.
2. V_{OL3} is the voltage when V_{SS} is supplied at pad.

3. LVD maximum value of FLASH is 2.2V, in case of MASK is 1.8V.
4. Current dissipation is proportioned according to operation voltage and frequency.
5. In sleep mode, oscillation continues and peripherals are operated normally but internal CPU clock stops.

7.4 LCD Characteristics

(T_A= -40~85°C, V_{DD}=1.8~5.5V, V_{SS}=0V)

Parameter	Symbol	Condition	Specifications			Unit
			Min.	Typ.	Max.	
LCD Reference Output Voltage	VCL0	External Variable Resistance (0 to 1MΩ)	0.9	-	2.0	V
Double Output Voltage	VCL1	C1~C4=0.47μF	1.9VCL0	2.0VCL0	-	V
Triple Output Voltage	VCL2	C1~C4=0.47μF	2.85VCL0	3.0VCL0	-	
LCD Common Output Current	I_{COM}	Output Voltage Deviation=0.2V	30	-	-	μA
LCD Segment Output Current	I_{SEG}	Output Voltage Deviation=0.2V	5	-	–	

7.5 A/D Converter Characteristics

(TA=25°C, V_{DD}=3.072V@f_{XIN}=4MHz, V_{DD}=5.12V@f_{XIN}=8MHz, V_{SS}=0V)

Parameter	Symbol	Pin/Condition	Specifications			Unit
			Min.	Typ.	Max.	
AV_{DD} input Current	I_{AVDD}[1]	V_{DD}	-	-	200	uA
Analog Power Supply Input Voltage Range	AV_{DD}	V_{DD}	2.7		5.5	V
Analog Input Voltage Range	V_{AIN}	AN0 ~ AN5	V_{SS}-0.3	-	V_{DD}+0.3	
Resolution	N_R	-		8		Bit
Analog Input Capacitance	CAIN	AN0 ~ AN5	-	-	30	pF
Input Impedance	IN[7:0]	AN0 ~ AN5	1	-	-	MΩ
Accuracy	N_{ACC}	-	-	±1.0	±1.5	LSB
Non Linearity Error	N_{NLE}	-	-	±1.0	±1.5	
Differential Non Linearity Error	N_{DNLE}	-	-	±1.0	±1.5	
Zero Offset Error	N_{ZOE}	-	-	±0.5	±1.5	
Full Scale Error	N_{FSE}	-	-	±0.25	±0.5	
Conversion Time	T_{CONV}	f_{XIN} = 8MHz	-	-	10	μS
		f_{XIN} = 4MHz	-	-	20	

1. AVDD input current is measured to V_{DD} pin when all blocks except ADC are disabled.

7.6 AC Characteristics

(TA=25°C, V_{DD}=4V, AV_{DD}=4V, V_{SS}=AV_{SS}=0V)

Parameter	Symbol	Pins	Specifications			Unit
			Min.	Typ.	Max.	
Main Operating Frequency	f_{MCP}	X_{IN}	0.4	-	12	MHz
Sub Operating Frequency	f_{SCP}	SX_{IN}	30	32.768	35	kHz
System Clock Frequency[1]	t_{SYS}	-	166	-	5000	nS
Main Oscillation Stabilization Time (4MHz)	t_{MST}	X_{IN}, X_{OUT}	-	-	20	mS
Main Oscillation Stabilization Time (910kHz)			-	-	60	
Main Oscillation Stabilization Time (455kHz)			-	-	100	
Sub Oscillation Stabilization Time	t_{SST}	SX_{IN}, SX_{OUT}	-	1	2	S
External Clock “H” or “L” Pulse Width	t_{MCPW}	X_{IN}	35	-	-	nS
	t_{SCPW}	SX_{IN}	5	-	-	μS
External Clock Transition Time	t_{RCP}, t_{FCP}	X_{IN}	-	-	20	nS
Interrupt Pulse Width	t_{IW}	INT0, INT1	2	-	-	t_{SYS}
$\overline{\text{RESET}}$ Input Pulse “L” Width	t_{RST}	$\overline{\text{RESET}}$	8	-	-	t_{SYS}
Event Counter Input “H” or “L” Pulse Width	t_{ECW}	EC0	2	-	-	t_{SYS}
Event Counter Transition Time	tREC, tFEC	EC0	-	-	20	nS

1.SCMR=XXXX000X_B that is f_{MAIN}÷2

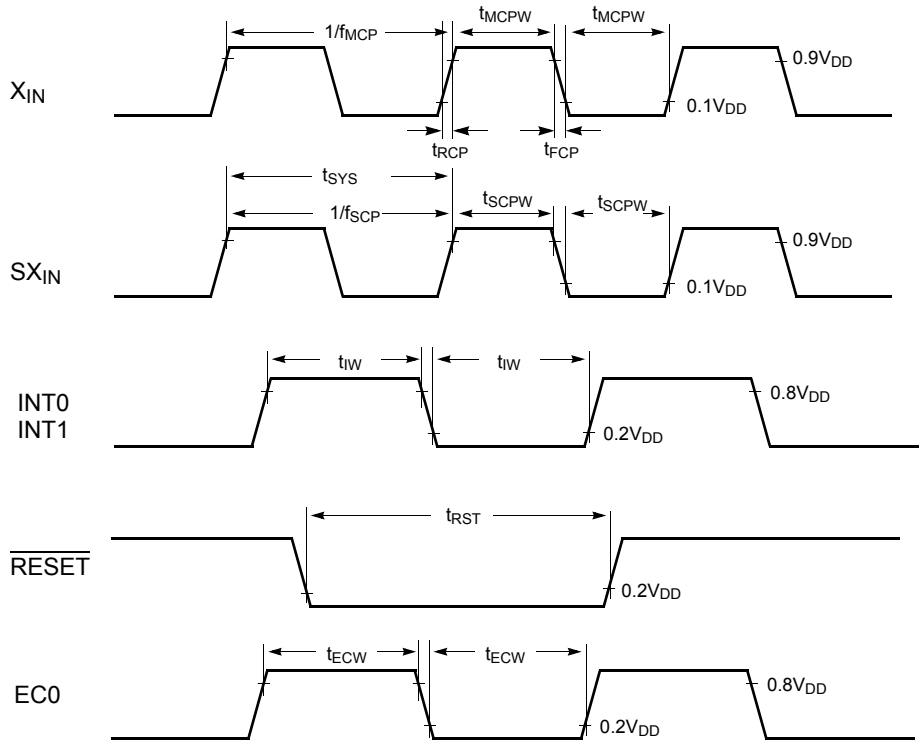


Figure 7-1 AC Timing Chart

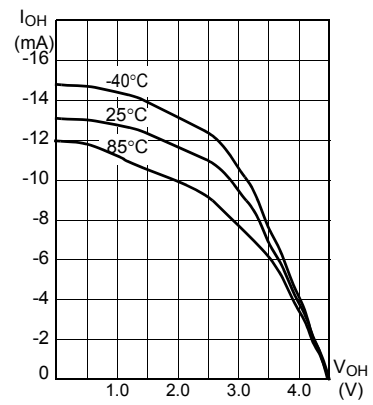
7.7 Typical Characteristics

These graphs and tables are for design guidance only and are not tested or guaranteed.

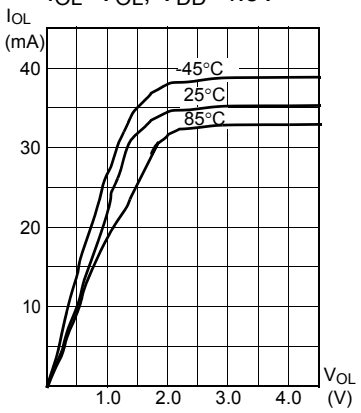
In some graphs or tables, the data presented are outside specified operating range (e.g. outside specified V_{DD} range). This is for information only and devices are guaranteed to operate properly only within the specified range.

The data is a statistical summary of data collected on units from different lots over a period of time. "Typical" represents the mean of the distribution while "max" or "min" represents (mean + 3σ) and (mean − 3σ) respectively where σ is standard deviation

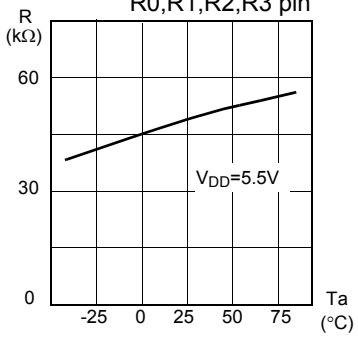
I_{OH}–V_{OH}, V_{DD}=4.5V



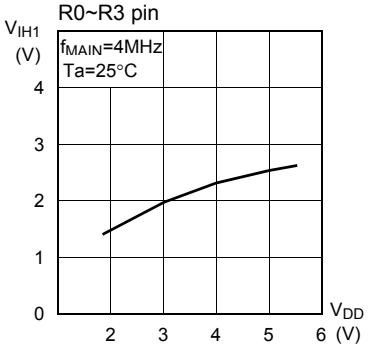
I_{OL}–V_{OL}, V_{DD}=4.5V



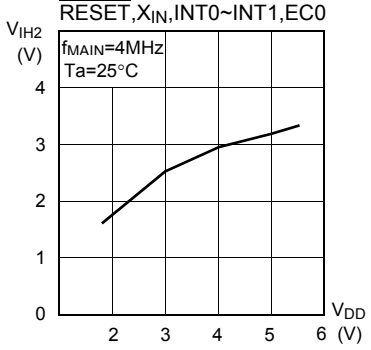
R–Ta R0,R1,R2,R3 pin



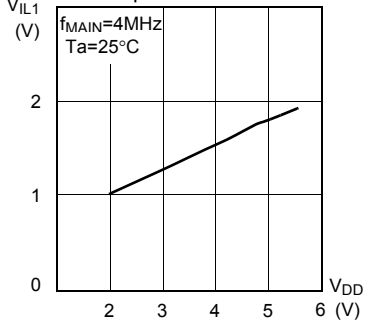
V_{DD}–V_{IH1}
R0~R3 pin



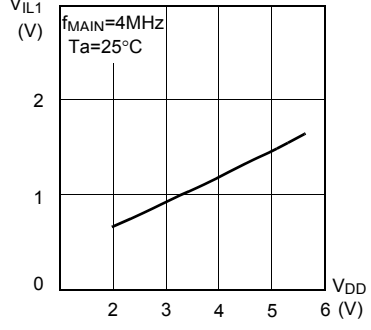
V_{DD}–V_{IH2}
$\overline{RESET}$,X_{IN},INT0~INT1,EC0



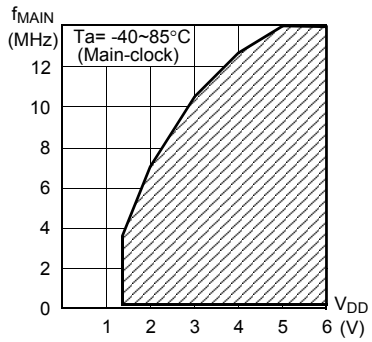
V_{DD}–V_{IL1}
R0~R3 pin



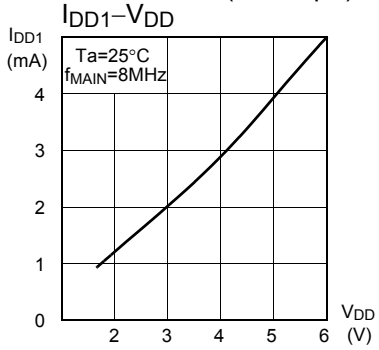
V_{DD}–V_{IL2}
$\overline{RESET}$,X_{IN},INT0~INT1,EC0



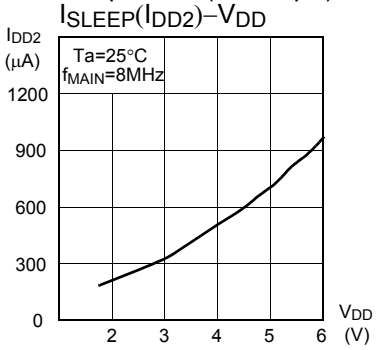
Operating Area



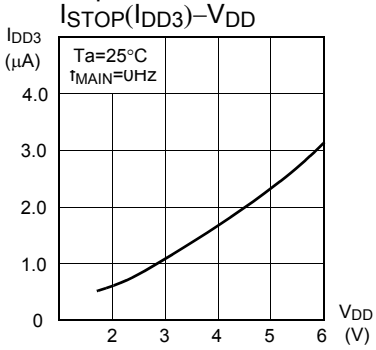
Normal Mode (Main opr.)
I_{DD1}–V_{DD}



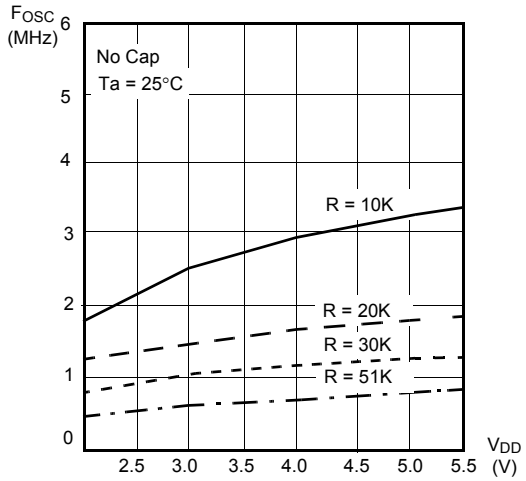
Sleep Mode (Main opr.)
$I_{SLEEP}(I_{DD2})$–V_{DD}



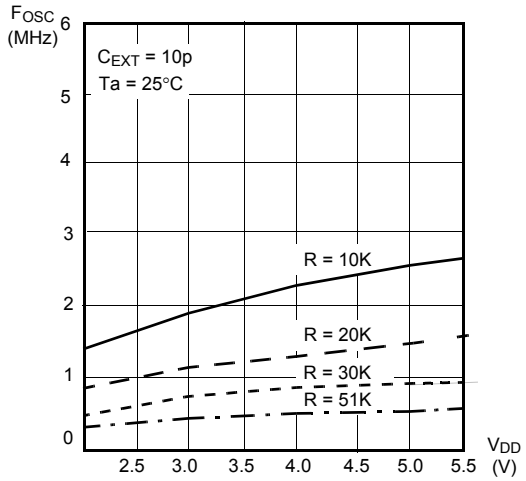
Stop Mode
$I_{STOP}(I_{DD3})$–V_{DD}



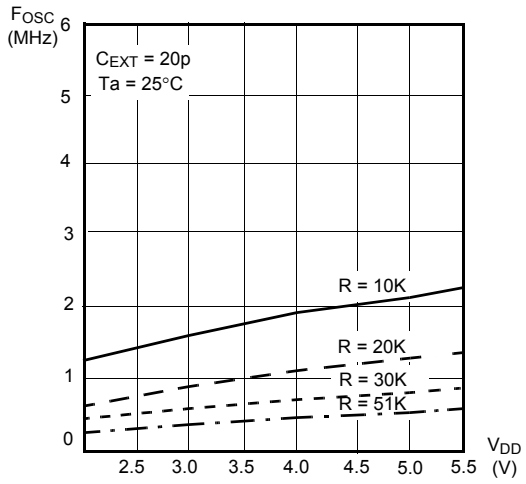
Typical RC Oscillator
Frequency vs V_{DD}



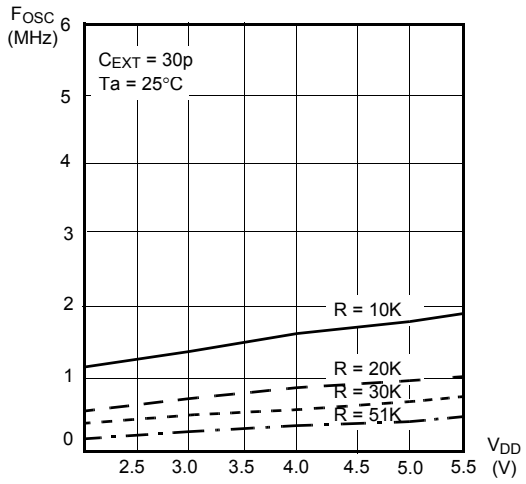
Typical RC Oscillator
Frequency vs V_{DD}



Typical RC Oscillator
Frequency vs V_{DD}



Typical RC Oscillator
Frequency vs V_{DD}



Note: *The external RC oscillation frequencies shown in above are provided for design guidance only and not tested or guaranteed. The user needs to take into account that the external RC oscillation frequencies generated by the same circuit design may be not the same. Because there are variations in the resistance and capacitance due to the tolerance of external R and C components. The parasitic capacitance difference due to the different wiring length and layout may change the external RC oscillation frequencies.*

Note: *The external RC oscillation frequencies of the MC80F7208 and MC80C7208 may be different as to the influence of environment. The user should modify the value of R and C components to get the proper frequency in exchanging FLASH and Mask device.*

8. MEMORY ORGANIZATION

The MC80X7208 has separate address spaces for Program memory, Data Memory and Display memory. Program memory can only be read, not written to. It can be up to 8K bytes of Program memory. Data memory can be read and written to up to 1024 bytes including the stack area. Display memory has prepared 27 nibbles for LCD.

8.1 Registers

This device has six registers that are the Program Counter (PC), a Accumulator (A), two index registers (X, Y), the Stack Pointer (SP), and the Program Status Word (PSW). The Program Counter consists of 16-bit register.

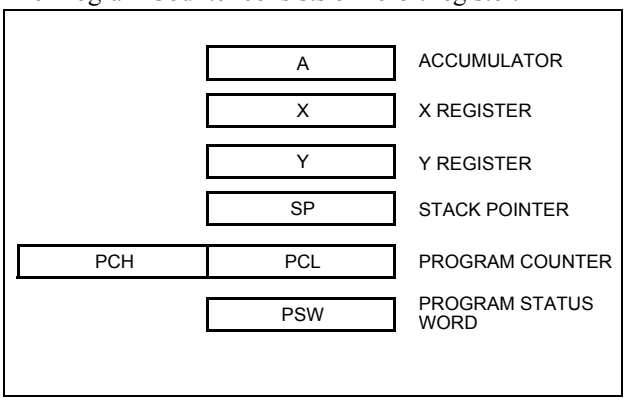


Figure 8-1 Configuration of Registers

Accumulator: The Accumulator is the 8-bit general purpose register, used for data operation such as transfer, temporary saving, and conditional judgement, etc.

The Accumulator can be used as a 16-bit register with Y Register as shown below.

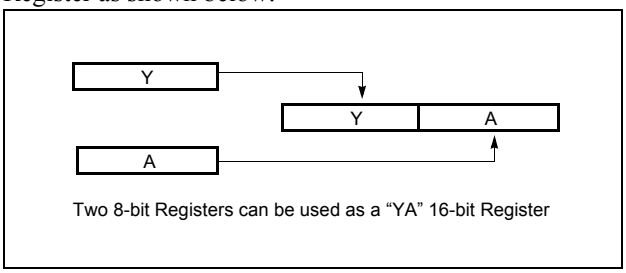


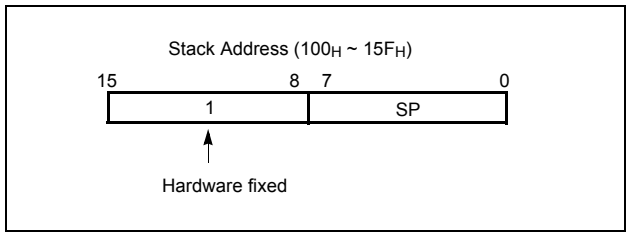
Figure 8-2 Configuration of YA 16-bit Register

X, Y Registers: In the addressing mode which uses these index registers, the register contents are added to the specified address, which becomes the actual address. These modes are extremely effective for referencing subroutine tables and memory tables. The index registers also have increment, decrement, comparison and data transfer functions, and they can be used as simple accumulators.

Stack Pointer: The Stack Pointer is an 8-bit register used for occurrence interrupts and calling out subroutines. Stack Pointer identifies the location in the stack to be accessed (save or restore).

Generally, SP is automatically updated when a subroutine call is executed or an interrupt is accepted. However, if it is used in excess of the stack area permitted by the data memory allocating configuration, the user-processed data may be lost.

The stack can be located at any position within 100_H to $15F_H$ of the internal data memory. The SP is not initialized by hardware, requiring to write the initial value (the location with which the use of the stack starts) by using the initialization routine. Normally, the initial value of "$15F_H$" is used.



Caution:

The Stack Pointer must be initialized by software because its value is undefined after RESET.

Example: To initialize the SP

```
LDX   #05FH      ;
TXSP              ;SP ← 05FH
```

Program Counter: The Program Counter is a 16-bit wide which consists of two 8-bit registers, PCH and PCL. This counter indicates the address of the next instruction to be executed. In reset state, the program counter has reset routine address (PC_H:$0FF_H$, PC_L:$0FE_H$).

Program Status Word: The Program Status Word (PSW) contains several bits that reflect the current state of the CPU. The PSW is described in . It contains the Negative flag, the Overflow flag, the Break flag the Half Carry (for BCD operation), the Interrupt enable flag, the Zero flag, and the Carry flag.

[Carry flag C]

This flag stores any carry or $\overline{\text{borrow}}$ from the ALU of CPU after an arithmetic operation and is also changed by the Shift Instruction or Rotate Instruction.

[Zero flag Z]

This flag is set when the result of an arithmetic operation or data transfer is “0” and is cleared by any other result.

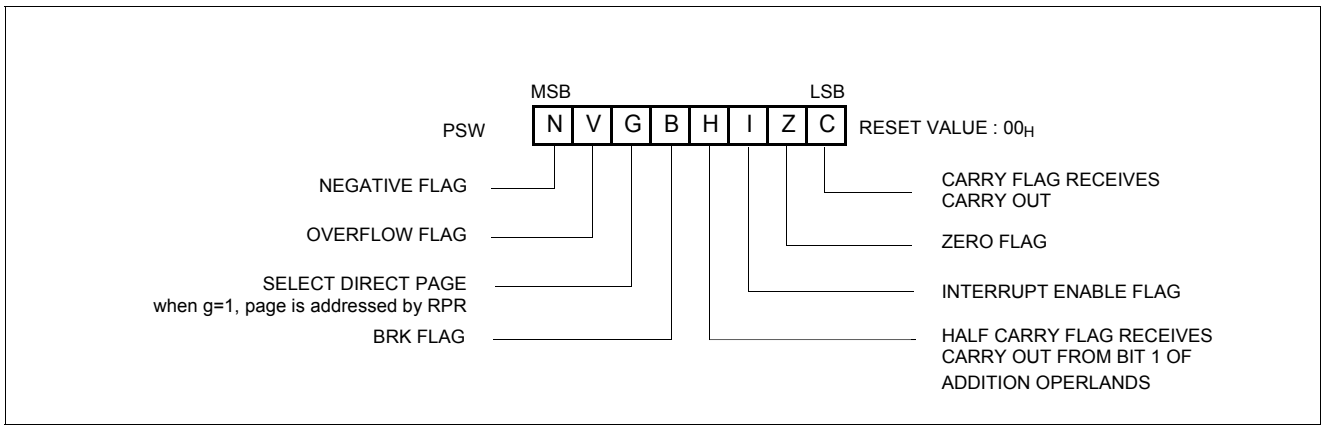


Figure 8-3 PSW (Program Status Word) Register

[Interrupt disable flag I]

This flag enables/disables all interrupts except interrupt caused by Reset or software BRK instruction. All interrupts are disabled when cleared to “0”. This flag immediately becomes “0” when an interrupt is served. It is set by the EI instruction and cleared by the DI instruction.

[Half carry flag H]

After operation, this is set when there is a carry from bit 3 of ALU or there is no borrow from bit 4 of ALU. This bit can not be set or cleared except CLRV instruction with Overflow flag (V).

[Break flag B]

This flag is set by software BRK instruction to distinguish BRK from TCALL instruction with the same vector address.

[Direct page flag G]

This flag assigns RAM page for direct addressing mode. In the direct addressing mode, addressing area is from zero page 00_H to $0FF_H$ when this flag is "0". If it is set to "1", addressing area is assigned by RPR register (address $0F3_H$). It is set by SETG instruction and cleared by CLRG.

[Overflow flag V]

This flag is set to “1” when an overflow occurs as the result of an arithmetic operation involving signs. An overflow occurs when the result of an addition or subtraction exceeds +127 ($7F_H$) or −128 (80_H). The CLRV instruction clears the overflow flag. There is no set instruction. When the BIT instruction is executed, bit 6 of memory is copied to this flag.

[Negative flag N]

This flag is set to match the sign bit (bit 7) status of the result of a data or arithmetic operation. When the BIT instruction is executed, bit 7 of memory is copied to this flag.

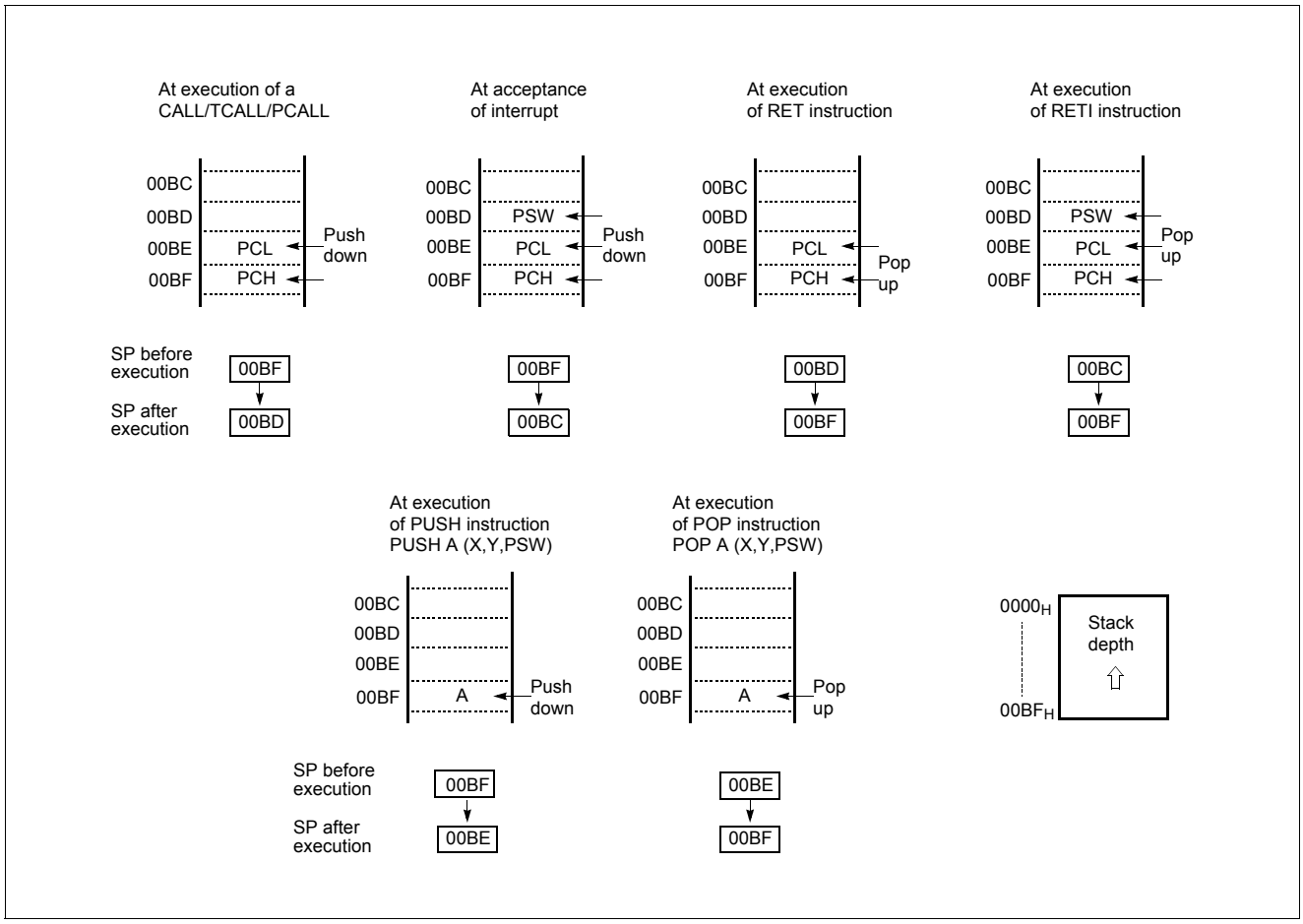


Figure 8-4 Stack Operation

8.2 Program Memory

A 16-bit program counter is capable of addressing up to 64K bytes, but this device has 8K bytes program memory space only physically implemented. Accessing a location above $FFFF_H$ will cause a wrap-around to 0000_H.

shows a map of Program Memory. After reset, the CPU begins execution from reset vector which is stored in address $FFFE_H$ and $FFFF_H$ as shown in .

As shown in , each area is assigned a fixed location in Program Memory. Program Memory area contains the user program.

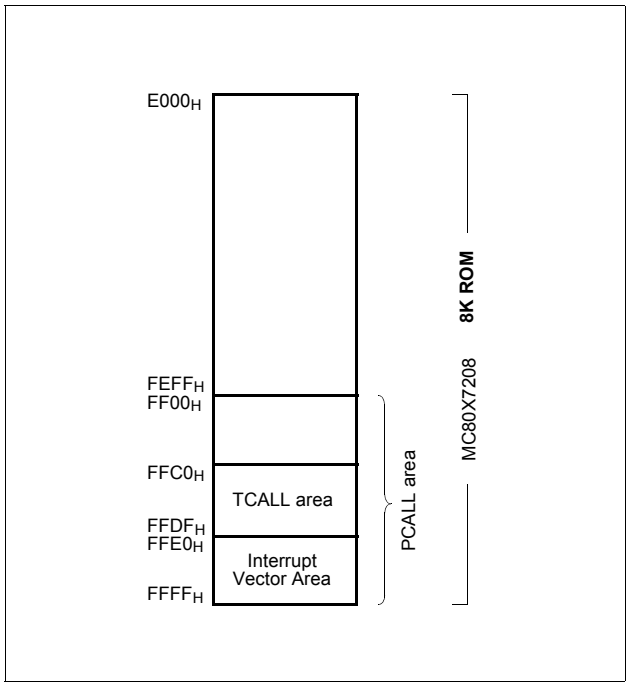


Figure 8-5 Program Memory Map

Page Call (PCALL) area contains subroutine program to reduce program byte length by using 2 bytes PCALL instead of 3 bytes CALL instruction. If it is frequently called, it is more useful to save program byte length.

Table Call (TCALL) causes the CPU to jump to each TCALL address, where it commences the execution of the service routine. The Table Call service area spaces 2-byte for every TCALL: $0FFC0_H$ for TCALL15, $0FFC2_H$ for TCALL14, etc., as shown in .

Example: Usage of TCALL

```
LDA       #5
          TCALL  0FH          ;1BYTE INSTRUCTION
          :                   ;INSTEAD OF 2 BYTES
          :                   ;NORMAL CALL
;
;TABLE CALL ROUTINE
;
FUNC_A:   LDA    LRG0
          RET
;
FUNC_B:   LDA    LRG1     ②  ①
          RET
;
;TABLE CALL ADD. AREA
;
          ORG    0FFC0H       ;TCALL ADDRESS AREA
          DW     FUNC_A
          DW     FUNC_B
```

The interrupt causes the CPU to jump to specific location, where it commences the execution of the service routine. The External interrupt 0, for example, is assigned to location $0FFFA_H$. The interrupt service locations spaces 2-byte interval: $0FFF8_H$ and $0FFF9_H$ for External Interrupt 1, $0FFFA_H$ and $0FFFB_H$ for External Interrupt 0, etc.

Any area from $0FF00_H$ to $0FFFF_H$, if it is not going to be used, its service location is available as general purpose Program Memory.

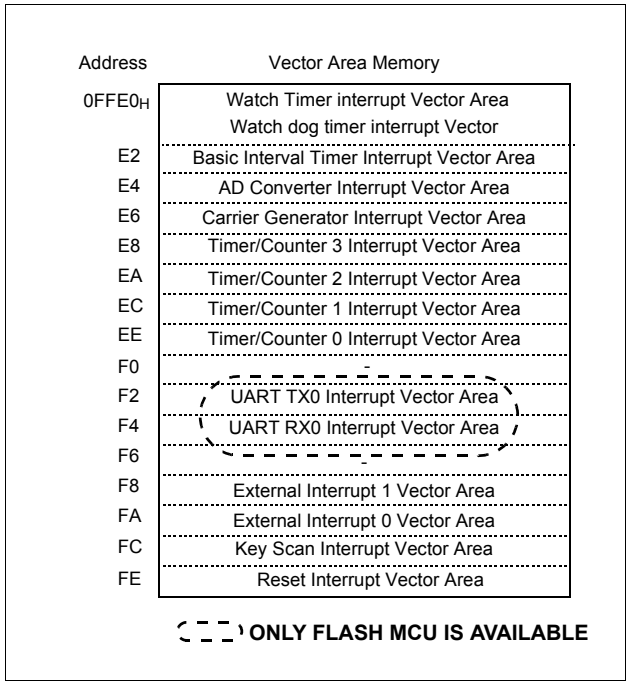


Figure 8-6 Interrupt Vector Area

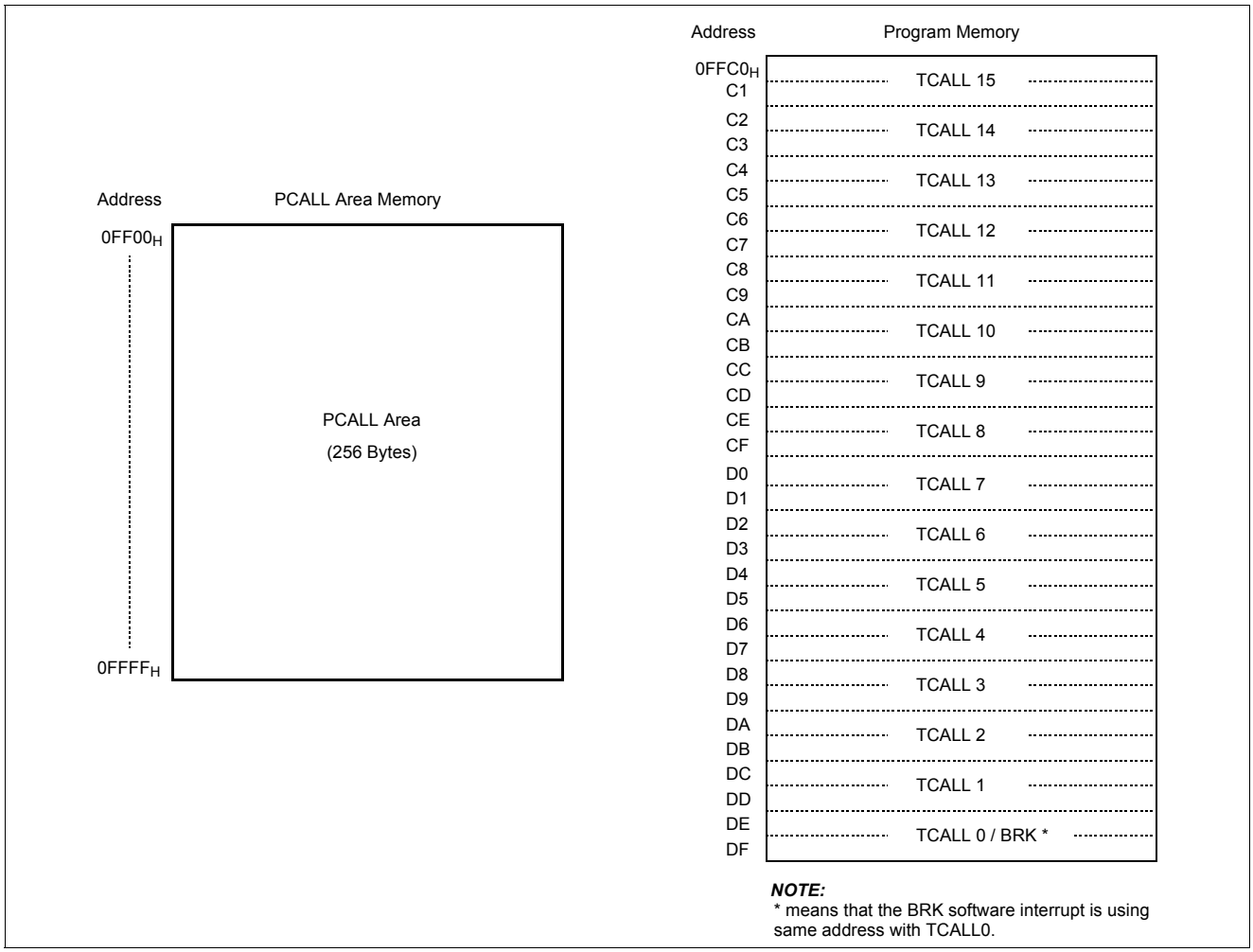
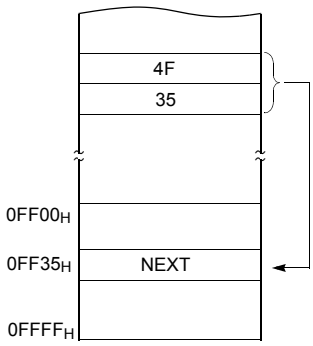


Figure 8-7 PCALL and TCALL Memory Area

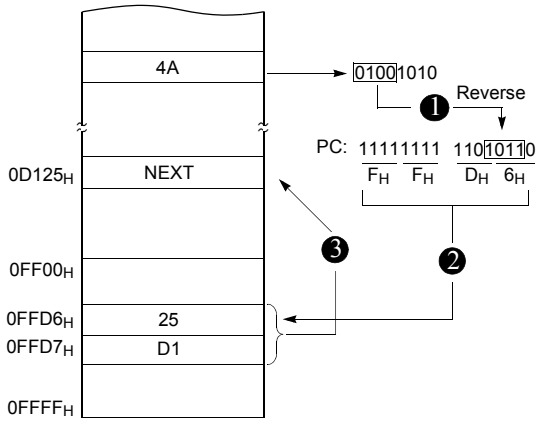
PCALL→ rel

```
4F35      PCALL  35H
```



TCALL→ n

```
4A        TCALL  4
```



Example: The usage software example of Vector address and the initialize part.

```
        ORG    0FFE0H      ; Device : MC80X7208, at X = F,C

        DW     WT_INT      ; Watch Timer / Watch Dog Timer
        DW     BIT_INT     ; Basic Interval Timer
        DW     AD_Con      ; AD converter
        DW     Carrier_INT ; Carrier
        DW     TMR3_INT    ; Timer-3
        DW     TMR2_INT    ; Timer-2
        DW     TMR1_INT    ; Timer-1
        DW     TMR0_INT    ; Timer-0
        DW     NOT_USED    ; Not Used
        DW     TX0_INT     ; UART TX0(ONLY FLASH MCU IS AVAILABLE)
        DW     RX0_INT     ; UART RX0(ONLY FLASH MCU IS AVAILABLE)
        DW     NOT_USED    ; Not Used
        DW     EX1_INT     ; Int.1
        DW     EX0_INT     ; Int.0
        DW     KEY_INT     ; Key Scan
        DW     RESET       ; Reset


;*****************************************
;           MAIN     PROGRAM            *
;*****************************************

        ORG    0E000H

RESET:  DI                 ;Disable All Interrupts
        CLRG
        LDX    #0
        LDA    #0
RAM_CLR1:STA   {X}+        ;Page0 RAM Clear(!0000H->!00BFH)
        CMPX   #0A0H
        BNE    RAM_CLR1

        LDM    RPR,#0000_0001B;Page1 RAM Clear(!0100H->!00FFH)
        SETG
        LDX    #0
        LDA    #0
RAM_CLR2:STA   {X}+
        CMPX   #060H
        BNE    RAM_CLR2
        CLRG

        LDX    #05FH       ;Stack Pointer Initialize
        TXSP

        LDM    RPR,#0000_0000B;Page0 selection

        CALL   LCD_CLR     ;Clear LCD display memory

        LDM    R0, #0      ;Normal Port 0
        LDM    R0IO,#1000_0010B;Normal Port Direction
        LDM    R0PU,#1000_0010B;Pull Up Selection Set
        LDM    R0OD,#0000_0001B;R0 port Open Drain control
        :
        :
        LDM    SCMR,#1111_0000B;System clock control
        :
        :
```

8.3 Data Memory

shows the internal Data Memory space available. Data Memory is divided into four groups, a user RAM, control registers, Stack, and LCD memory.

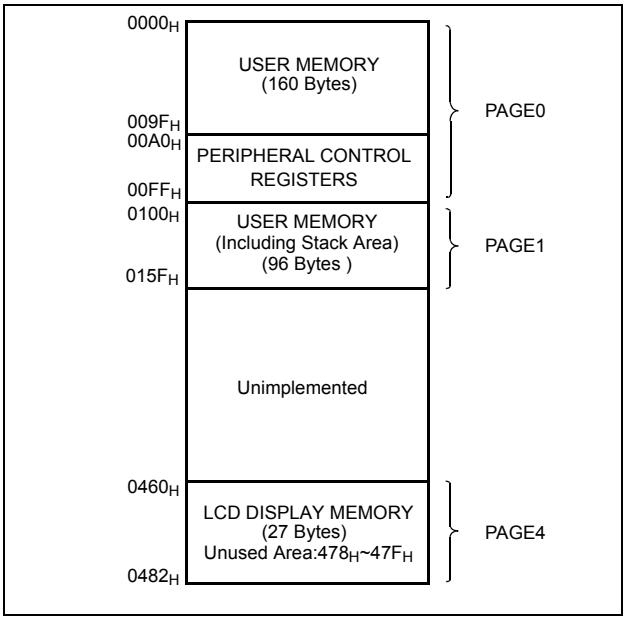


Figure 8-8 Data Memory Map

User Memory

The MC80X7208 has 256 × 8 bits for the user data memory (RAM). There are three pages internal RAM. Page is selected by G-flag and RAM page selection register RPR. When G-flag is cleared to "0", always page 0 is selected regardless of RPR value. If G-flag is set to "1", page will be selected according to RPR value.

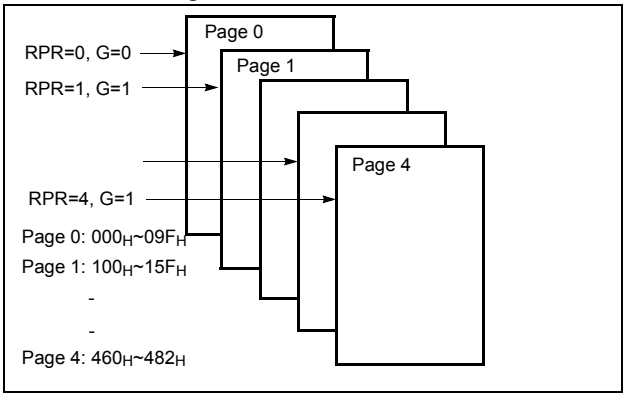


Figure 8-9 RAM page configuration

RPR (RAM Page Selection Register)

ADDRESS: $0CF_H$
INITIAL VALUE: ----_-001_B

MSB							LSB
					R/W	R/W	R/W
-	-	-	-	-	RPR2	RPR1	RPR0

RPR[2:0] (RAM Page Selection)

RPR2	RPR1	RPR0	RAM Page Selection
0	0	0	PAGE 0
0	0	1	PAGE 1
0	1	0	-
0	1	1	-
1	0	0	PAGE 4
1	0	1	-
1	1	0	-
1	1	1	-

Caution1 : After setting RPR, be sure to execute SETG instruction.
Caution2 : When executing CLRG instruction, be selected PAGE0 regardless of RPR.

Figure 8-10 RAM Page Selection Register

Control Registers

The control registers are used by the CPU and Peripheral function blocks for controlling the desired operation of the device. Therefore these registers contain control and status bits for the interrupt system, the timer/counters, analog to digital converters and I/O ports. The control registers are in address range of $0A0_H$ to $0FF_H$.

Note that unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

More detailed informations of each register are explained in each peripheral section.

Note: *Write only registers can not be accessed by bit manipulation instruction. Do not use read-modify-write instruction. Use byte manipulation instruction.*

Example; To write at CKCTLR

```
LDM    CKCTLR,#05H ;Divide ratio ÷2^8
```

Stack Area

The stack provides the area where the return address is saved before a jump is performed during the processing

routine at the execution of a subroutine call instruction or the acceptance of an interrupt.

When returning from the processing routine, executing the subroutine return instruction [RET] restores the contents of the program counter from the stack; executing the interrupt return instruction [RETI] restores the contents of the program counter and flags.

The save/restore locations in the stack are determined by the stack pointer (SP). The SP is automatically decreased after the saving, and increased before the restoring. This means the value of the SP indicates the stack location number for the next save. Refer to Figure 8-4 on page 23.

LCD Display Memory

LCD display data area is handled in LCD section.

See "19.3 LCD Display Memory" on page 79.

Address	Register Name	Symbol	R/W	Initial Value								Addressing Mode
				7	6	5	4	3	2	1	0	
00A0H	R0 Open Drain Control Register	R0OD	W	0	0	-	0	-	-	0	-	byte[1]
00A1H	R1 Open Drain Control Register	R1OD	W	-	-	-	-	-	-	-	0	byte
00A2H	R2Open Drain Control Register	R2OD	W	-	-	0	0	0	0	0	0	byte
00A3H	R3 Open Drain Control Register	R3OD	W	0	0	0	0	-	-	-	-	byte
00A5H	R0 Pull-up Register	R0PU	W	0	0	-	0	**0**	**0**	0	-	byte
00A6H	R1 Pull-up Register	R1PU	W	-	-	-	-	-	-	-	0	byte
00A7H	R2 Pull-up Register	R2PU	W	-	-	0	0	0	0	0	0	byte
00A8H	R3 Pull-up Register	R3PU	W	0	0	0	0	-	-	-	-	byte
00AAH	R0 Function Selection Register	R0FUNC	W	0	0	-	0	**0**	**0**	0	-	byte
00ABH	R1 Function Selection Register	R1FUNC	W	-	-	-	-	-	-	-	0	byte
00ACH	R5 Port Selection Register	R5PSR[2]	R / W	1	1	1	1	1	1	1	1	byte, bit[3]
00ADH	R6 Port Selection Register	R6PSR[2]	R / W	1	1	1	1	1	1	1	1	byte, bit
00AEH	R7 Port Selection Register	R7PSR[2]	R / W	1	1	1	1	1	1	1	1	byte, bit
00B0H	R7 Data Register	R7	R / W	0	0	0	0	0	0	0	0	byte
00B2H	LCD Control Register	LCR	R / W	0	0	0	0	0	0	0	0	byte, bit
00B8H	Asynchronous Serial Mode Register0	ASIMR0	R / W	**0**	**0**	**0**	**0**	**-**	**0**	**0**	**-**	byte, bit
00B9H	Asynchronous Serial Status Register0	ASISR0	R	**-**	**-**	**-**	**-**	**-**	**0**	**0**	**0**	byte
00BAH	Baud Rate Generator Control Register0	BRGCR0	R / W	**-**	**0**	**0**	**1**	**0**	**0**	**0**	**0**	byte, bit
00BBH	Receive Buffer Register0	RXBR0	R	**0**	**0**	**0**	**0**	**0**	**0**	**0**	**0**	byte
	Transmit Shift Register0	TXSR0	W	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**	byte
00C0H	R0 port data register	R0	R / W	0	0	-	0	-	-	0	-	byte, bit
00C1H	R0 Direction Register	R0IO	W	0	0	-	0	-	-	0	-	byte
00C2H	R1 port data register	R1	R / W	-	-	-	-	-	-	-	0	byte, bit
00C3H	R1 Direction Register	R1IO	W	-	-	-	-	-	-	-	0	byte
00C4H	R2 port data register	R2	R / W	-	-	0	0	0	0	0	0	byte, bit
00C5H	R2 Direction Register	R2IO	W	-	-	0	0	0	0	0	0	byte
00C6H	R3 port data register	R3	R/W	0	0	0	0	-	-	-	-	byte, bit
00C7H	R3 Direction Register	R3IO	W	0	0	0	0	-	-	-	-	byte
00CAH	R5 port data register	R5	R/W	0	0	0	0	0	0	0	0	byte, bit

Table 8-1 Control Registers

Address	Register Name	Symbol	R/W	Initial Value							Addressing Mode
				7	6	5	4	3	2	1	0	
00CBH	R5 Direction Register	R5IO	W	0	0	0	0	0	0	0	0	byte
00CCH	R6 port data register	R6	R/W	0	0	0	0	0	0	0	0	byte, bit
00CDH	R6 Direction Register	R6IO	W	0	0	0	0	0	0	0	0	byte
00CEH	Buzzer Driver Register	BUZR	W	1	1	1	1	1	1	1	1	byte
00CFH	Ram Page Selection Register	RPR	R / W	-	-	-	-	-	0	0	1	byte, bit
00D0H	Timer 0 Mode Control Register	TM0	R / W	-	-	0	0	0	0	0	0	byte, bit
00D1H	Timer 0 Register	T0	R	0	0	0	0	0	0	0	0	byte
	Timer 0 Data Register	TDR0	W	1	1	1	1	1	1	1	1	byte
	Timer 0 Capture Data Register	CDR0	R	0	0	0	0	0	0	0	0	byte
00D2	Timer 1 Mode Control Register	TM1	R / W	0	0	-	0	0	0	0	0	byte, bit
00D3H	Timer 1 Data Register	TDR1	W	1	1	1	1	1	1	1	1	byte
00D4H	Timer 1 Register	T1	R	0	0	0	0	0	0	0	0	byte
	Timer 1 Capture Data Register	CDR1	R	0	0	0	0	0	0	0	0	byte
00D6H	Timer 2 Mode Control Register	TM2	R / W	-	-	0	0	0	0	0	0	byte, bit
00D7H	Timer 2 Register	T2	R	0	0	0	0	0	0	0	0	byte
	Timer 2 Data Register	TDR2	W	1	1	1	1	1	1	1	1	byte
	Timer 2 Capture data Register	CDR2	R	0	0	0	0	0	0	0	0	byte
00D8H	Timer 3 Mode Control Register	TM3	R / W	0	0	0	0	0	0	0	0	byte, bit
00D9H	Timer 3 Data Register	TDR3	W	1	1	1	1	1	1	1	1	byte
	Timer 3 PWM Period Register	T3PPR	W	1	1	1	1	1	1	1	1	byte
00DAH	Timer 3 Register	T3	R	0	0	0	0	0	0	0	0	byte
	Timer 3 PWM Duty Register	T3PDR	R / W	0	0	0	0	0	0	0	0	byte, bit
	Timer 3 Capture Data Register	CDR3	R	0	0	0	0	0	0	0	0	byte
00DBH	Timer 3 PWM High Register	T3PWHR	W	-	-	-	-	0	0	0	0	byte
00E2H	8bit A/D Converter Mode Control Register	ADCM[4]	R / W	-	0	-	0	0	0	0	1	byte, bit
00E3H	8bit A/D Converter Result Register	ADCR	R	Undefined								byte
00E5H	PFD Control Register	PFDR	R / W	0	0	0	0	0	0	0	0	byte, bit
00E6H	Basic Interval Timer Register	BITR	R	Undefined								byte
	Clock Control Register	CKCTLR	W	-	-	0	1	0	1	1	1	byte
00E7H	System Clock Mode Register	SCMR	R / W	-	-	-	-	-	0	0	0	byte
00E8H	Watch Dog Timer Register	WDTR	W	0	1	1	1	1	1	1	1	byte
	Watch Dog Timer Data Register	WDTDR	R	Undefined								byte
	Watch Timer Register	WTR	W	0	1	1	1	1	1	1	1	byte
00E9H	Stop & Sleep Mode Control Register	SSCR	W	0	0	0	0	0	0	0	0	byte
00EAH	Watch Timer Mode Register	WTMR	R / W	0	0	-	-	0	0	0	0	byte, bit
00EBH	Key Scan Mode Register	KSMR	R / W	0	0	0	0	-	-	-	-	byte, bit
00ECH	Carrier Frequency High Selection	CFHS	W	-	-	1	1	1	1	1	1	byte

Table 8-1 Control Registers

Address	Register Name	Symbol	R/W	Initial Value							Addressing Mode
				7	6	5	4	3	2	1	0	
00EDH	Carrier Frequency Low Selection	CFLS	W	-	-	1	1	1	1	1	1	byte
00EEH	Remocon Mode Register	RMR	R / W	-	0	0	0	0	0	0	0	byte, bit
00EFH	Remocon Data High Register	RDHR	W	1	1	1	1	1	1	1	1	byte
00F0H	Remocon Data Low Register	RDLR	W	1	1	1	1	1	1	1	1	byte
00F1H	Remocon Data Counter	RDC	R	0	0	0	0	0	0	0	0	byte
00F2H	Remocon Output Data Register	RODR	R / W	-	-	-	-	-	-	-	0	byte, bit
00F3H	Remocon Output Buffer	ROB	R / W	-	-	-	-	-	-	-	0	byte, bit
00F5H	Interrupt Generation Flag Register Low	INTFL	R / W	0	0	-	-	-	0	0	0	byte, bit
00F6H	Interrupt Enable Register High	IENH	R / W	0	0	0	-	**0**	**0**	-	-	byte, bit
00F7H	Interrupt Enable Register Middle	IENM	R / W	0	0	0	0	-	-	0	0	byte, bit
00F8H	Interrupt Enable Register Low	IENL	R / W	-	0	0	0	-	-	-	-	byte, bit
00F9H	Interrupt Request Register High	IRQH	R / W	0	0	0	-	**0**	**0**	-	-	byte, bit
00FAH	Interrupt Request Register Middle	IRQM	R / W	0	0	0	0	-	-	0	0	byte, bit
00FBH	Interrupt Request Register Low	IRQL	R / W	-	0	0	0	-	-	-	-	byte, bit
00FCH	Interrupt Edge Selection Register	IEDS	R / W	-	-	-	-	0	0	0	0	byte, bit

Table 8-1 Control Registers

1. “byte”, “bit” means that register can be addressed by not only bit but byte manipulation instruction.
2. The initial value for MCUs and Emulator is just the opposite to prevent glitter from LCD panel at reset state.
3. “byte” means that register can be addressed by only byte manipulation instruction. On the other hand, do not use any read-modify-write instruction such as bit manipulation.
4. Bit 0 of ADCM is read only.

ONLY FLASH MCU IS AVAILABLE

8.4 Addressing Mode

The MC80X7208 uses six addressing modes;

- **Register addressing**
- **Immediate addressing**
- **Direct page addressing**
- **Absolute addressing**
- **Indexed addressing**
- **Register-indirect addressing**

(1) Register Addressing

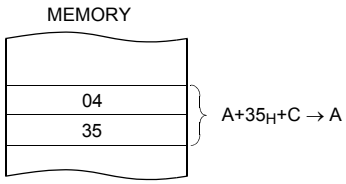
Register addressing accesses the A, X, Y, C and PSW.

(2) Immediate Addressing → #imm

In this mode, second byte (operand) is accessed as a data immediately.

Example:

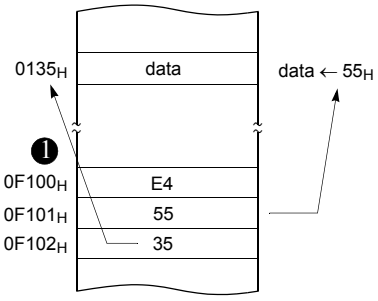
```
0435    ADC    #35H
```



When G-flag is 1, then RAM address is defined by 16-bit address which is composed of 8-bit RAM paging register (RPR) and 8-bit immediate data.

Example: G=1, RPR=01H

```
E45535  LDM    35H,#55H
```

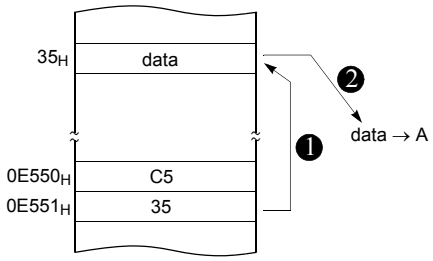


(3) Direct Page Addressing → dp

In this mode, a address is specified within direct page.

Example; G=0

```
C535    LDA    35H          ;A ←RAM[35H]
```



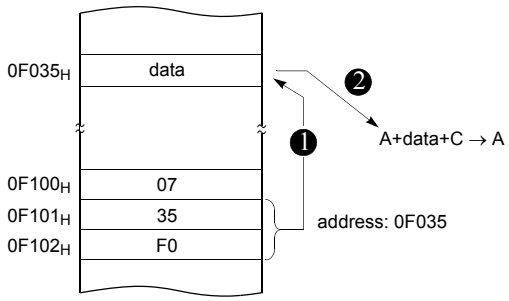
(4) Absolute Addressing → !abs

Absolute addressing sets corresponding memory data to Data, i.e. second byte (Operand I) of command becomes lower level address and third byte (Operand II) becomes upper level address.
With 3 bytes command, it is possible to access to whole memory area.

ADC, AND, CMP, CMPX, CMPY, EOR, LDA, LDX, LDY, OR, SBC, STA, STX, STY

Example;

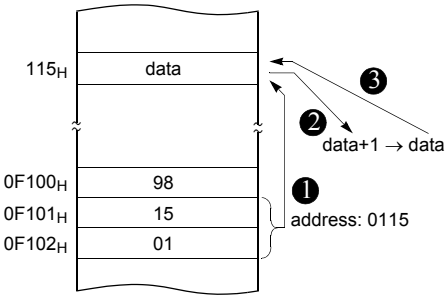
```
0735F0  ADC    !0F035H      ;A ←ROM[0F035H]
```



The operation within data memory (RAM)
ASL, BIT, DEC, INC, LSR, ROL, ROR

Example; Addressing accesses the address 0135_H regardless of G-flag and RPR.

```
981501   INC    !0115H        ;A ←ROM[115H]
```



(5) Indexed Addressing

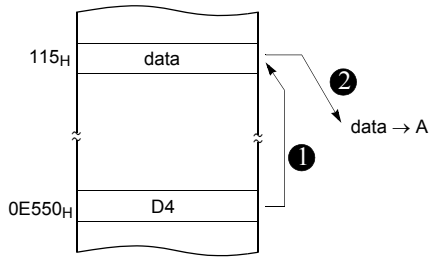
X indexed direct page (no offset) → {X}

In this mode, a address is specified by the X register.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA, XMA

Example; $X=15_H$, G=1, $RPR=01_H$

```
D4       LDA    {X}          ;ACC←RAM[X].
```



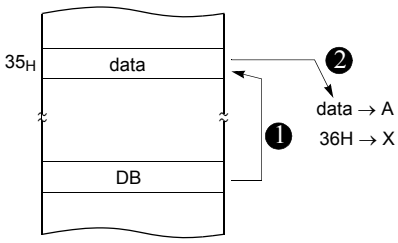
X indexed direct page, auto increment→ {X}+

In this mode, a address is specified within direct page by the X register and the content of X is increased by 1.

LDA, STA

Example; G=0, $X=35_H$

```
DB       LDA    {X}+
```



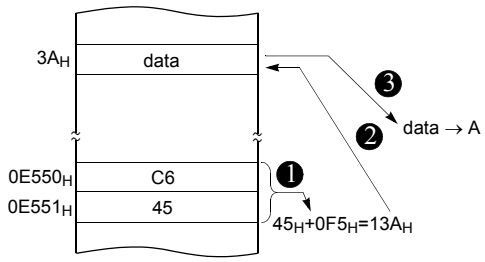
X indexed direct page (8 bit offset) → dp+X

This address value is the second byte (Operand) of command plus the data of X-register. And it assigns the memory in Direct page.

ADC, AND, CMP, EOR, LDA, LDY, OR, SBC, STA STY, XMA, ASL, DEC, INC, LSR, ROL, ROR

Example; G=0, $X=0F5_H$

```
C645     LDA    45H+X
```



Y indexed direct page (8 bit offset) → dp+Y

This address value is the second byte (Operand) of command plus the data of Y-register, which assigns Memory in Direct page.

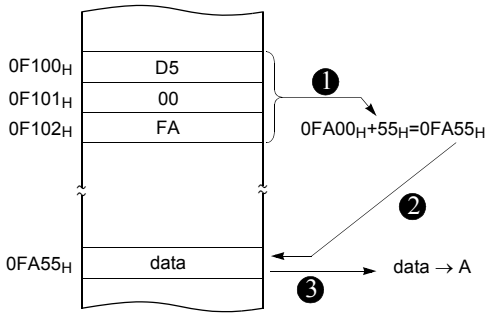
This is same with above. Use Y register instead of X.

Y indexed absolute → !abs+Y

Sets the value of 16-bit absolute address plus Y-register data as Memory. This addressing mode can specify memory in whole area.

Example; Y=55_H

```
D500FA   LDA    !0FA00H+Y
```



(6) Indirect Addressing

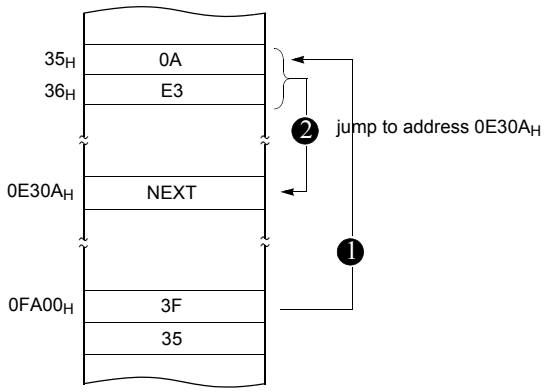
Direct page indirect → [dp]

Assigns data address to use for accomplishing command which sets memory data (or pair memory) by Operand. Also index can be used with Index register X,Y.

JMP, CALL

Example; G=0

```
3F35     JMP    [35H]
```



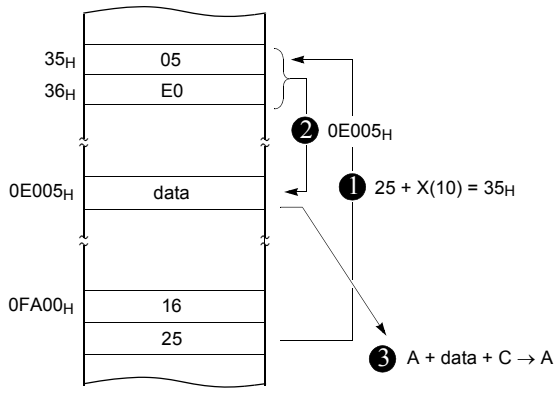
X indexed indirect → [dp+X]

Processes memory data as Data, assigned by 16-bit pair memory which is determined by pair data [dp+X+1][dp+X] Operand plus X-register data in Direct page.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA

Example; G=0, X=10_H

```
1625     ADC    [25H+X]
```



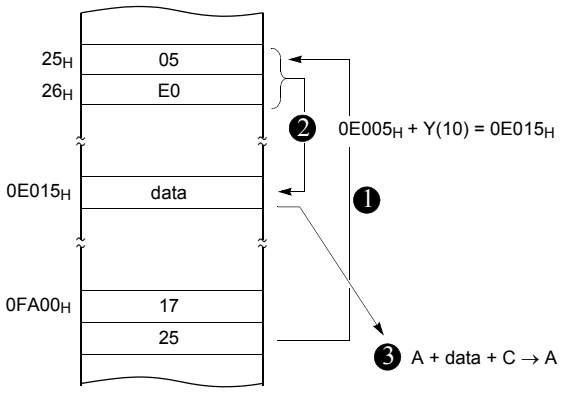
Y indexed indirect → [dp]+Y

Processes memory data as Data, assigned by the data [dp+1][dp] of 16-bit pair memory paired by Operand in Direct page plus Y-register data.

ADC, AND, CMP, EOR, LDA, OR, SBC, STA

Example; G=0, Y=10_H

```
1725     ADC    [25H]+Y
```



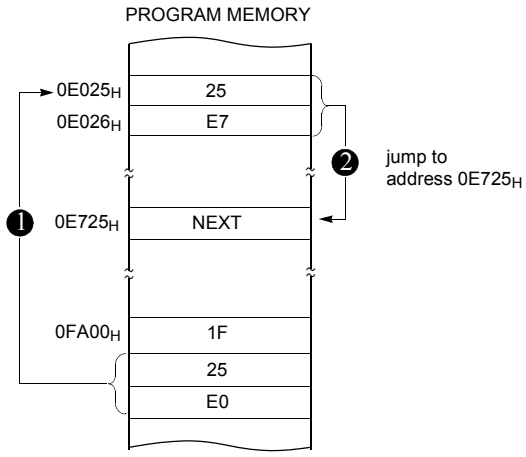
Absolute indirect → [!abs]

The program jumps to address specified by 16-bit absolute address.

JMP

Example; G=0

```
1F25E0   JMP    [!0E025H]
```



9. I/O PORTS

The MC80X7208 has seven I/O ports, LCD segment ports (R0, R1, R2, R3, R5/SEG0 ~ R7/SEG23) and LCD common ports (COM0, COM1/SEG34, COM2/SEG33 and COM3/SEG32).

These ports pins may be multiplexed with an alternate function for the peripheral features on the device.

9.1 Registers for Ports

Port Data Registers

The Port Data Registers (R0, R1, R2, R3) are represented as a D-Type flip-flop, which will clock in a value from the internal bus in response to a "write to data register" signal from the CPU. The Q output of the flip-flop is placed on the internal bus in response to a "read data register" signal from the CPU. The level of the port pin itself is placed on the internal bus in response to "read data register" signal from the CPU. Some instructions that read a port activating the "read register" signal, and others activating the "read pin" signal.

Port Direction Registers

All pins have data direction registers which can define these ports as output or input. A "1" in the port direction register configure the corresponding port pin as output. Conversely, write "0" to the corresponding bit to specify it as input pin. For example, to use the even numbered bit of R0 as output ports and the odd numbered bits as input ports, write "55_H" to address $0C1_H$ (R0 port direction register) during initial setting as shown in .

All the port direction registers in the MC80X7208 have 0 written to them by reset function. Therefore, its initial status is input.

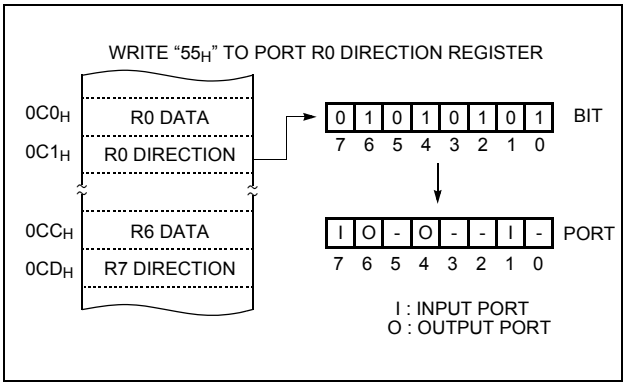


Figure 9-1 Example of port I/O assignment

Pull-up Control Registers

The R0, R1, R2 and R3 ports have internal pull-up resistors. shows a functional diagram of a typical pull-up port. It is connected or disconnected by Pull-up Control register (R*n*PU). The value of that resistor is typically 100kΩ. Refer to DC characteristics for more details.

When a port is used as key input, input logic is firmly either low or high, therefore external pull-down or pull-up resisters are required practically. The MC80X7208 has internal pull-up, it can be logic high by pull-up that can be able to configure either connect or disconnect individually by pull-up control registers R*n*PU.

When ports are configured as inputs and pull-up resistor is selected by software, they are pulled to high.

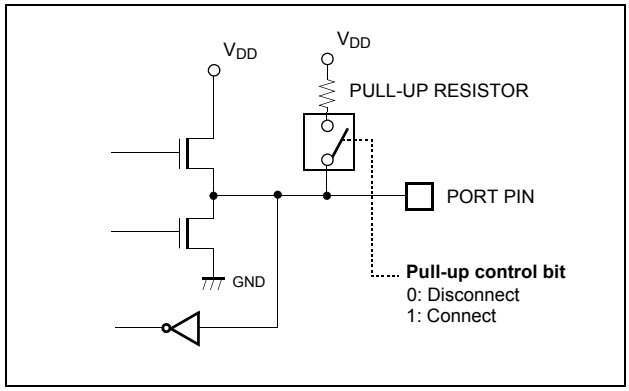


Figure 9-2 Pull-up Port Structure

Open drain port Registers

The R0, R1, R2 and R3 ports have open drain port resistors R0OD~R3OD.
 shows an open drain port configuration by control register. It is selected as either push-pull port or open-drain port by R0OD, R1OD, R2OD and R3OD.

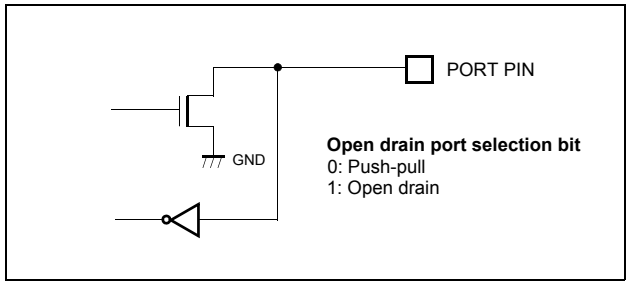


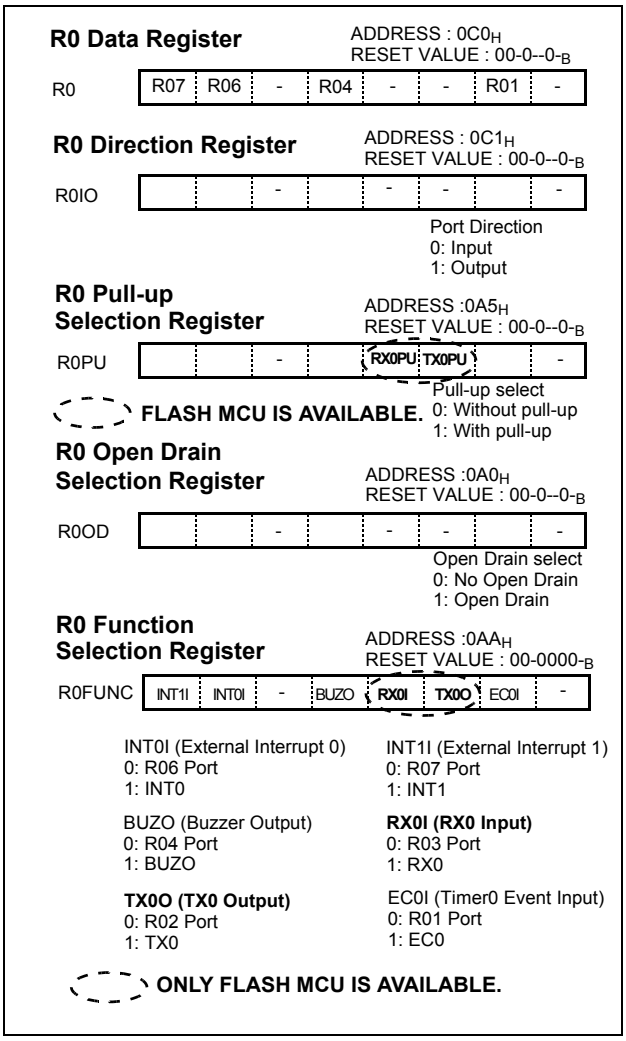
Figure 9-3 Open-drain Port Structure

9.2 I/O Ports Configuration

R0 Port

R0 is a 4-bit CMOS bidirectional I/O port (address $0C0_H$). Each I/O pin can independently used as an input or an output through the R0IO register (address $0C1_H$).

R0 has internal pull-ups that is independently connected or disconnected by R0PU. The control registers for R0 are shown below.



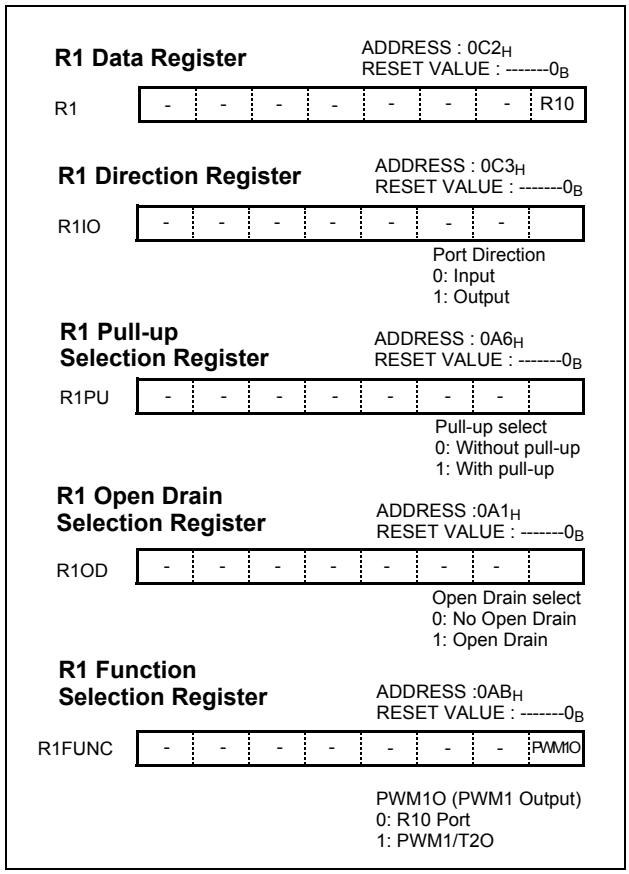
In addition, Port R0 is multiplexed with various special features. The control register R0FUNC (address 0AAH) controls the selection of alternate function. After reset, this value is "0", port may be used as normal I/O port.
To use alternate function such as External Interrupt rather than normal I/O, write "1" in the corresponding bit of F0FUNC.

Port Pin	Alternate Function
R01	EC0 (Timer0 Event Input)
-	**TX0 (TX0 Output) : FLASH MCU AVAILABLE**
-	**RX0 (RX0 Input) : FLASH MCU AVAILABLE**
R04	BUZO (Buzzer Output)
R06	INT0 (External Interrupt 0)
R07	INT1 (External Interrupt 1)

R1 Ports

R1 is an 1-bit CMOS bidirectional I/O port (address $0C2_H$). Each I/O pin can independently used as an input or an output through the R1IO register (address $0C3_H$).

R1 has internal pull-up that is independently connected or disconnected by register R1PU. The control registers for R1 are shown below.



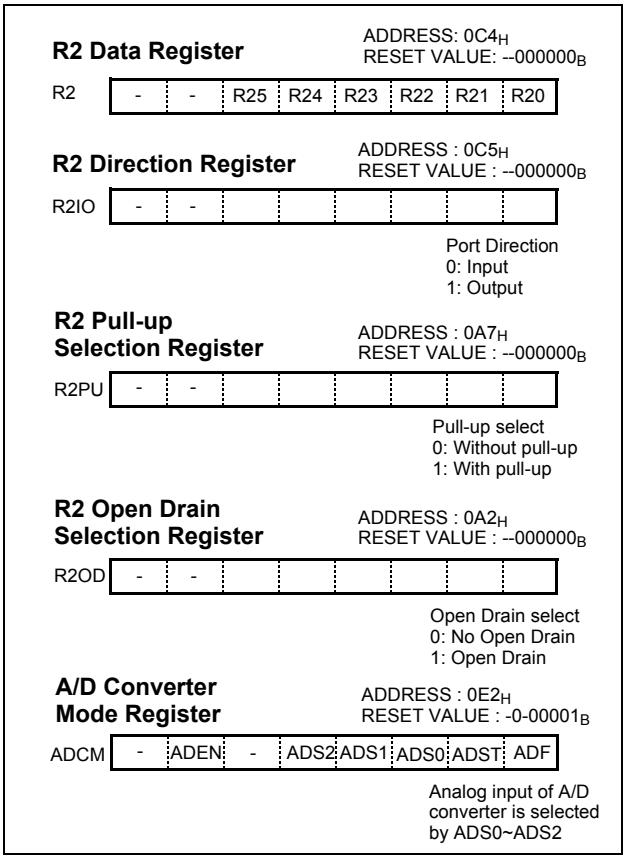
Port R1 is multiplexed with two special features. The control register controls the selection of alternate function. After reset, this value is "0", port may be used as normal I/O port. The way to select alternate function such as PWM1

or Timer Output Wave will be shown in each peripheral section.

R2 Ports

R2 is a 6-bit CMOS bidirectional I/O port (address $0C4_H$). Each I/O pin can independently used as an input or an output through the R2IO register (address $0C5_H$).

R2 has internal pull-ups that are independently connected or disconnected by R2PU (address $0A7_H$). The control registers for R2 are shown as below.

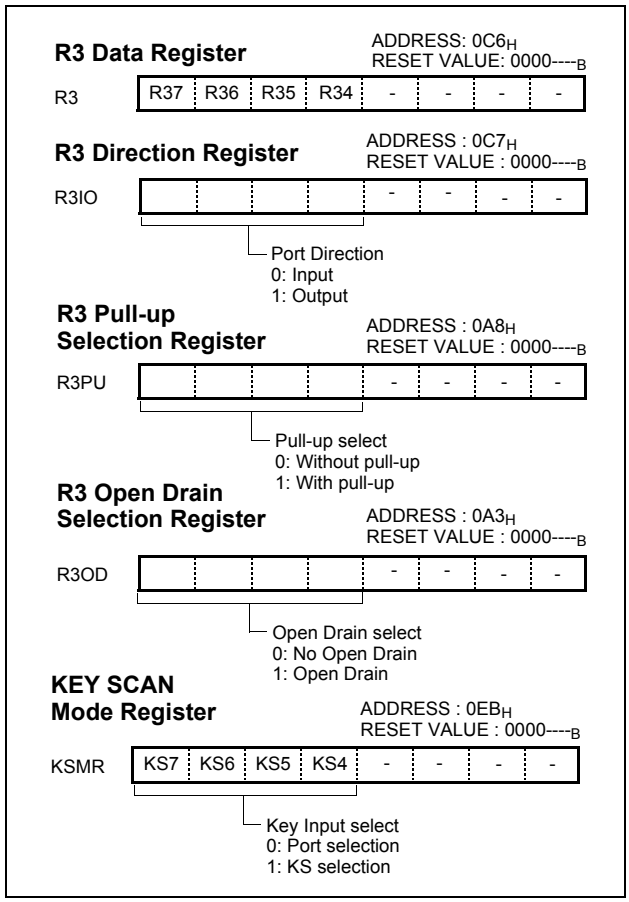


R3 Ports

R3 is a 4-bit CMOS bidirectional I/O port (address $0C6_H$). Each I/O pin can independently used as an input or an output through the R3IO register (address $0C7_H$).

In addition, R3 port is used as key scan function which operate with normal input port.

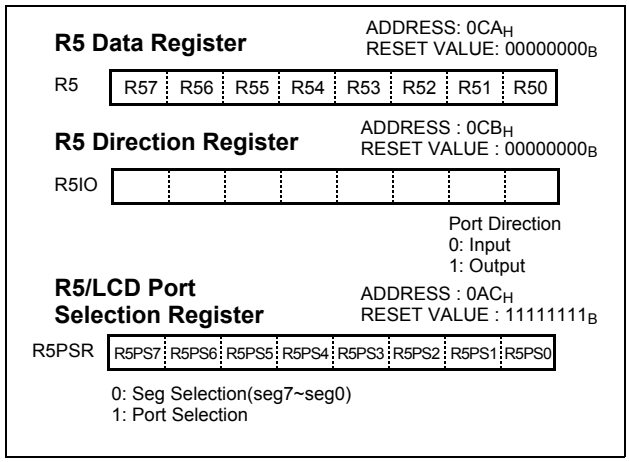
Input or output is configured automatically by each function register(KSMR) regardless of R3IO.



R5 Ports

R5 is an 8-bit CMOS bidirectional I/O port (address $0CA_H$). Each I/O pin can independently used as an input or an output through the R5IO register (address $0CB_H$).

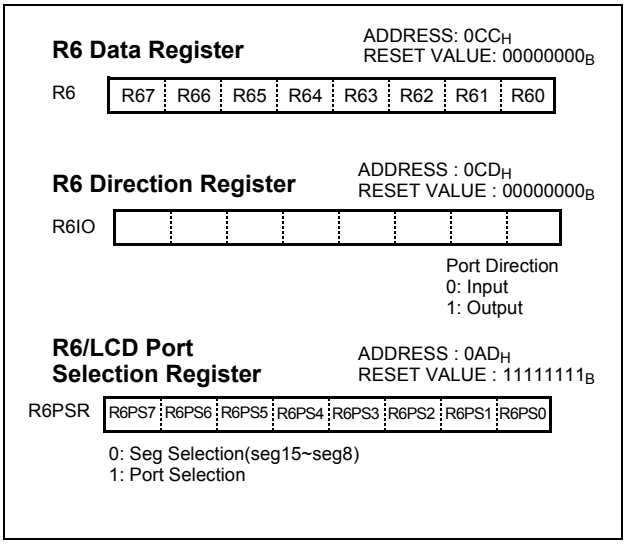
After Reset, R5 port is used as LCD segment output SEG7~SEG0. To use general Input ports user should be written appropriate value into the R5PSR(address $0AC_H$).



R6 Ports

R6 is an 8-bit CMOS bidirectional I/O port (address $0CC_H$). Each I/O pin can independently used as an input or an output through the R6IO register (address $0CD_H$).

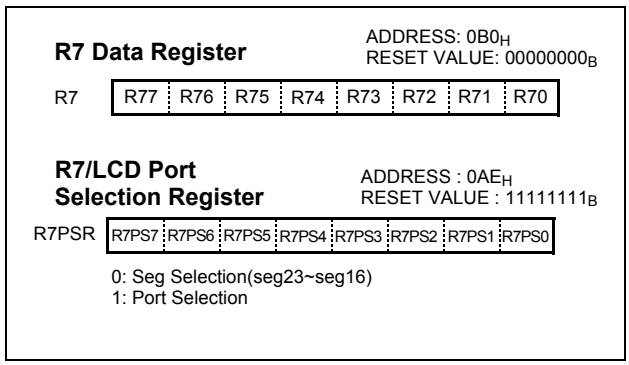
After Reset, R6 port is used as LCD segment output SEG15~SEG8. To use general Input ports user should be written appropriate value into the R6PSR(address $0AD_H$).



R7 Ports

R7 is an 8-bit CMOS general Input port (address $0B0_H$).

After Reset, R7 port is used as LCD segment output SEG23~SEG16. To use general Input ports user should be written appropriate value into the R7PSR(address $0AE_H$).



SEG0~SEG23

Segment signal output pins for the LCD display. See "19. LCD DRIVER" on page 76 for details.

COM0~COM3

Common signal output pins for the LCD display. See "19. LCD DRIVER" on page 76 for details.

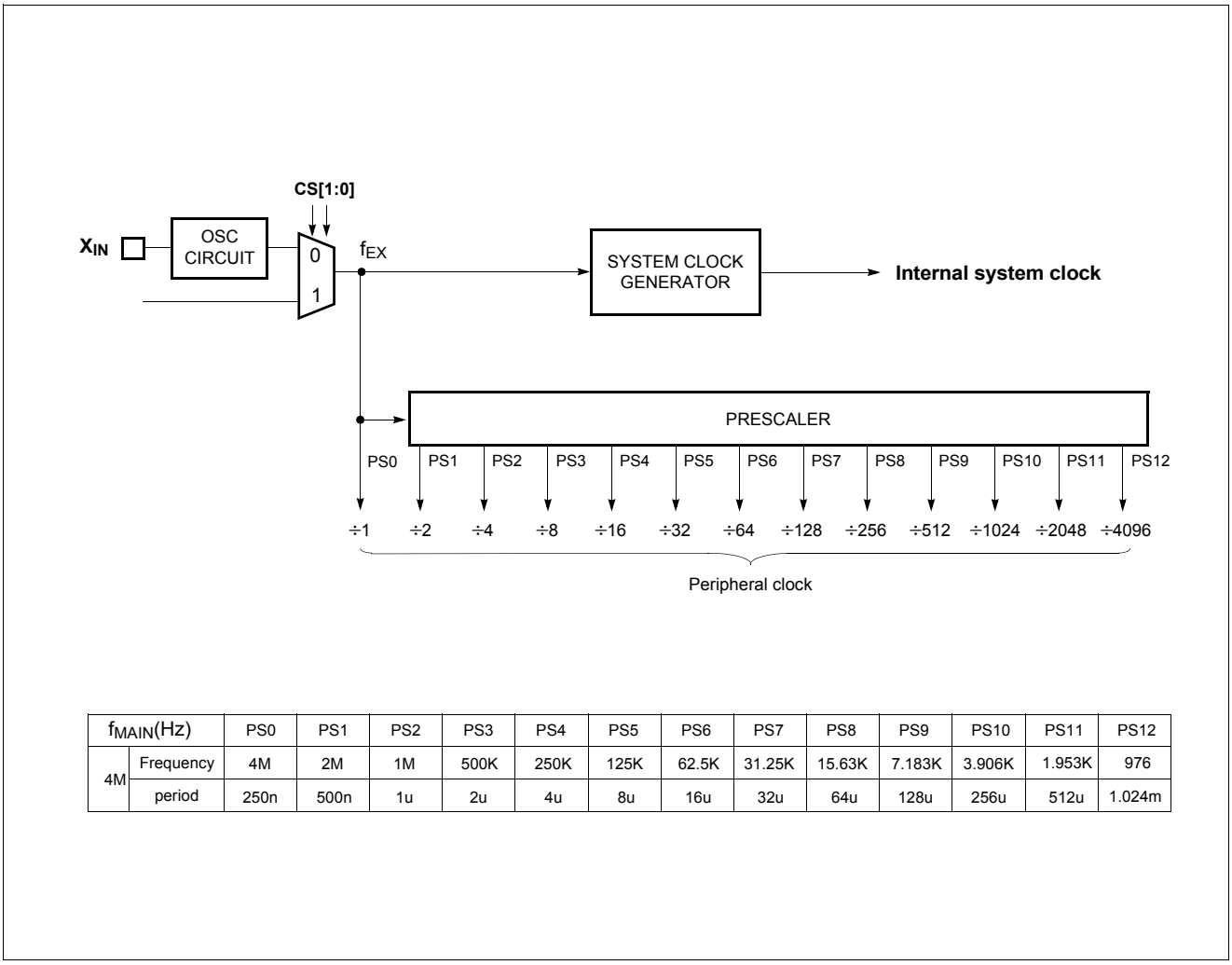
SEG34~SEG32 and COM1~COM3 are selected by LCDD of the LCR register.

10. CLOCK GENERATOR

As shown in , the clock generator produces the basic clock pulses which provide the system clock to be supplied to the CPU and the peripheral hardware. It contains two oscillators which are main-frequency clock oscillator and a sub-frequency clock oscillator. Power consumption can be reduced by switching them to the low power operation frequency clock that can be easily obtained by attaching a resonator between the X_{IN} and X_{OUT} pin and the SX_{IN} and SX_{OUT} pin, respectively. The system clock can also be obtained from the external oscillator.

The clock generator produces the system clocks forming clock pulse, which are supplied to the CPU and the peripheral hardware. The internal system clock can be selected by bit1, and bit0 of the system clock mode register (SCMR). The registers are shown in .

To the peripheral block, the clock among the not-divided original clocks, divided by 2, 4,..., up to 4096 can be provided. Peripheral clock is enabled or disabled by STOP instruction. The peripheral clock is controlled by clock control register (CKCTLR). See "11. BASIC INTERVAL TIMER" on page 41 for details.



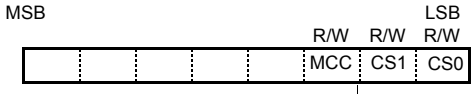
f_{MAIN}(Hz)		PS0	PS1	PS2	PS3	PS4	PS5	PS6	PS7	PS8	PS9	PS10	PS11	PS12
4M	Frequency	4M	2M	1M	500K	250K	125K	62.5K	31.25K	15.63K	7.183K	3.906K	1.953K	976
	period	250n	500n	1u	2u	4u	8u	16u	32u	64u	128u	256u	512u	1.024m

Figure 10-1 Block Diagram of Clock Generator

The system clock is selected by bit1 of the system clock mode register, SCMR. In selection Sub clock, to oscillate or stop the Main clock is decided by bit0 of SCMR.

On the initial reset, internal system clock is PS1 which is the fastest and other clock can be provided by bit2 and bit3 of SCMR.

SCMR (System Clock Mode Register)



ADDRESS: $0E7_H$
INITIAL VALUE: -----000_B

CS[1:0] (System clock selection)
00: main clock
01: main clock
10: Reserved
11: setting prohibited

MCC (Main Clock Oscillation Control)
0: main clock oscillation possible
1: main clock oscillation stopped

Note 1. When the CPU is operating on the subsystem clock, MCC should be used to stop the main system oscillation. A STOP instruction should not be used.

Figure 10-2 SCMR : System Clock Mode Register

11. BASIC INTERVAL TIMER

The MC80X7208 has one 8-bit Basic Interval Timer that is free-run and can not stop. Block diagram is shown in .

The Basic Interval Timer Register (BITR) is increased every internal count pulse which is divided by prescaler. Since prescaler has division ratio from 8 to 1024, the count rate is 1/8 to 1/1024 of the oscillator frequency. After reset, the BCK bits are all set, so the longest oscillation stabilization time is obtained.

It also provides a Basic interval timer interrupt (BITF). The count overflow of BITR from FF_H to 00_H causes the interrupt to be generated. The Basic Interval Timer is controlled by the clock control register (CKCTLR) shown in .

Source clock can be selected by lower 3 bits of CKCTLR. When write "1" to bit BCL of CKCTLR, BITR register is cleared to "0" and restart to count up. The bit BCL becomes "0" automatically after one machine cycle by hardware.

BITR and CKCTLR are located at same address, and address $0E6_H$ is read as a BITR, and written to CKCTLR.

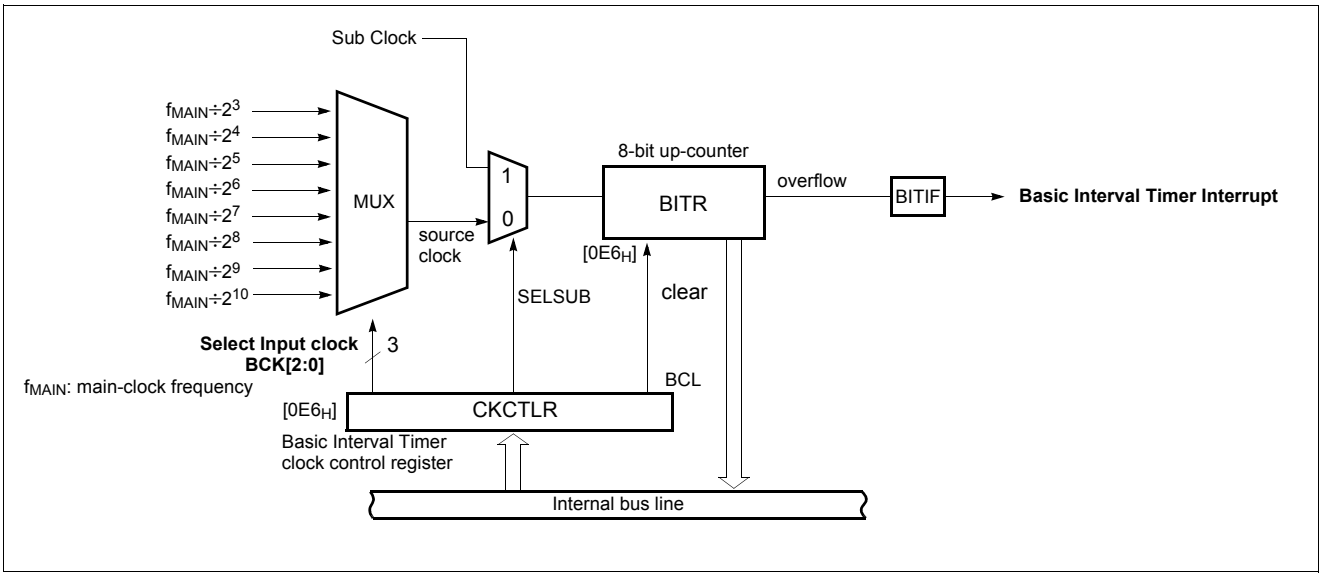


Figure 11-1 Block Diagram of Basic Interval Timer

BCK <2:0>	Source clock	Interrupt (overflow) Period
	SCMR[1:0]= 00 or 01	At f_{MAIN}=4MHz
000	$f_{MAIN}÷2^3$	0.512 ms
001	$f_{MAIN}÷2^4$	1.024
010	$f_{MAIN}÷2^5$	2.048
011	$f_{MAIN}÷2^6$	4.096
100	$f_{MAIN}÷2^7$	8.192
101	$f_{MAIN}÷2^8$	16.384
110	$f_{MAIN}÷2^9$	32.768
111	$f_{MAIN}÷2^{10}$	65.536

Table 11-1 Basic Interval Timer Interrupt Time

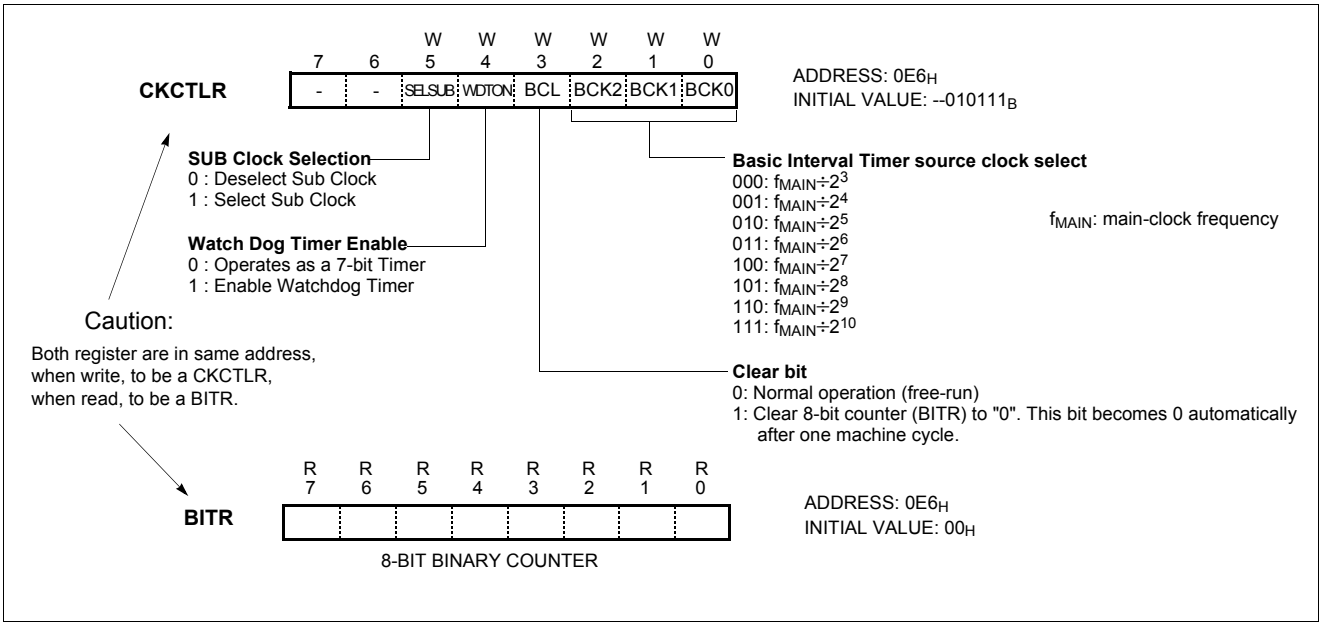


Figure 11-2 BITR: Basic Interval Timer Mode Register

Example 1:

Interrupt request flag is generated every 8.192ms at 4MHz.

```
:
LDM    CKCTLR,#0CH
SET1   BITE
EI
:
```

12. TIMER / COUNTER

Timer/Event Counter consists of prescaler, multiplexer, 8-bit timer data register, 8-bit counter register, mode register, input capture register and Comparator as shown in . And the PWM high register for PWM is consisted separately.

The timer/counter has seven operating modes.
- 8 Bit Timer/Counter Mode
- 8 Bit Capture Mode
- 8 Bit Compare Output Mode
- 16 Bit Timer/Counter Mode
- 16 Bit Capture Mode
- 16 Bit Compare Output Mode
- PWM Mode

In the “timer” function, the register is increased every internal clock input. Thus, one can think of it as counting internal clock input. Since a least clock consists of 2 and most clock consists of 2048 oscillator periods, the count rate is 1/2 to 1/2048 of the oscillator frequency in Timer0. And Timer1 can use the same clock source too. In addition, Timer1 has more fast clock source (1/1 to 1/8).

In the “counter” function, the register is increased in response to a 0-to-1 (rising edge) transition at its corresponding external input pin EC0 (Timer 0).

In addition the “capture” function, the register is increased in response external or internal clock interrupt same with timer/counter function. When external interrupt edge input, the count register is captured into capture data register TMx.

Timer3 is shared with “PWM” function and Timer2 is shared with “Compare output” function.

Example 1:

Timer 0 = 8-bit timer mode, 8ms interval at 4MHz
Timer 1 = 8-bit timer mode, 4ms interval at 4MHz

```
LDM   SCMR,#0        ;Main clock mode
LDM   TDR0,#249
LDM   TM0,#0001_0011B
LDM   TDR1,#124
LDM   TM1,#0000_1111B

SET1  T0E
SET1  T1E
EI
 :
 :
 :
```

Example 2:

Timer0 = 16-bit timer mode, 0.5s at 4MHz

```
LDM   SCMR,#0        ;Main clock mode
LDM   TDR0,#23H
LDM   TDR1,#0F4H
LDM   TM0,#0FH       ;FMAIN÷32, 8us
LDM   TM1,#4CH

SET1  T0E
EI
 :
 :
 :
```

Example 3:

Timer0 = 8-bit event counter

Timer2 = 8-bit capture mode, 2us sampling count.

```
LDM   TDR0,#0FFH       ;don't care
LDM   TM0,#1FH          ;event counter
LDM   R0IO,#1XXX_XX1XB ;R07, R01 input

LDM   IEDS,#XXXX_01XXB  ;FALLING
LDM   R0FUNC,#1XXX_XX1XB;INT1,EC0
LDM   TDR2,#0FFH
LDM   TM2,#0010_1011B   ;2us

SET1  T0E               ;ENABLE TIMER 0
SET1  T2E               ;ENABLE TIMER 1
SET1  INT1E             ;ENABLE INT1
EI
 :
```

X: don’t care.

Example 4:

Timer0 = 16-bit capture mode, 8us sampling count. at 4MHz

```
LDM   TDR0,#0FFH
LDM   TDR1,#0FFH
LDM   TM0,#2FH
LDM   TM1,#5FH

LDM   IEDS,#XXXX_XX01B
LDM   R0FUNC,#X1XX_XXXXB ;AS INT0

SET1  T0E    ;ENABLE TIMER 0
SET1  INT0E  ;ENABLE EXT. INT0
EI
 :
```

X: don’t care.

TM0, TM2 (Timer0, 2 Mode Control Register)

	Bit :	7	6	5	4	3	2	1	0	
		R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
TM0		-	-	CAP0	T0CK2	T0CK1	T0CK0	T0CN	T0ST	ADDRESS: $0D0_H$ INITIAL VALUE:--000000$_B$
TM2		-	-	CAP2	T2CK2	T2CK1	T2CK0	T2CN	T2ST	ADDRESS: $0D6_H$ INITIAL VALUE:--000000$_B$

CAP0,CAP2 (Capture Mode Selection Bit)
0: Timer/Counter Mode
1: Capture Mode

T0CN,T2CN (Timer 0,2 Continue Start)
0: Pause Counting
1: Continue Counting

T0ST,T2ST (Timer 0,2 Start Control)
0: Stop Counting
1: Clear the counter and Start counting again

T0CK[2:0],T2CK[2:0] (Timer 0,2 Input Clock Selection)

T0CK[2:0]	**T2CK[2:0]**
000: $f_{MAIN}÷2$	000: $f_{MAIN}÷2$
001: $f_{MAIN}÷2^2$	001: $f_{MAIN}÷2^2$
010: $f_{MAIN}÷2^3$	010: $f_{MAIN}÷2^4$
011: $f_{MAIN}÷2^5$	011: $f_{MAIN}÷2^6$
100: $f_{MAIN}÷2^7$	100: $f_{MAIN}÷2^8$
101: $f_{MAIN}÷2^9$	101: $f_{MAIN}÷2^{10}$
110: $f_{MAIN}÷2^{11}$	110: $f_{MAIN}÷2^{12}$
111: External Event clock EC0	111: Reserved

f_{MAIN}: main-clock frequency

TM1, TM3 (Timer1, 3 Mode Control Register)

	Bit :	7	6	5	4	3	2	1	0	
			R/W			R/W	R/W	R/W	R/W	
TM1		-	16BIT	-	-	T1CK1	T1CK0	T1CN	T1ST	ADDRESS: $0D2_H$ INITIAL VALUE:-0--0000$_B$
		R/W	R/W	R/W		R/W	R/W	R/W	R/W	
TM3		POL	16BIT	PWM1E	-	T3CK1	T3CK0	T3CN	T3ST	ADDRESS: $0D8_H$ INITIAL VALUE:000-0000$_B$

POL (PWM Output Polarity Selection)
0: Duty Active Low
1: Duty Active High

16BIT (16 Bit Mode Selection)
0: 8-Bit Mode
1: 16-Bit Mode

PWM1E (PWM Enable Bit)
0: PWM1 Disable (T2O Enable)
1: PWM1 Enable (T2O Disable)

T1CK[1:0],T3CK[1:0] (Timer 1,3 Input Clock Selection)

T1CK[1:0]	**T3CK[1:0]**
00: f_{MAIN}	00: f_{MAIN}
01: $f_{MAIN}÷2$	01: $f_{MAIN}÷2$
10: $f_{MAIN}÷2^3$	10: $f_{MAIN}÷2^4$
11: Timer0 Clock	11: Timer2 Clock

T1CN,T3CN (Timer 1,3 Continue Start)
0: Stop Counting
1: Start Counting

T1ST,T3ST (Timer 1,3 Start Control)
0: Stop counting
1: Clear the counter and start count again

*****The counter will be cleared and restarted only when the TxST bit cleared and set again.***
If TxST bit set again when TxST bit is set, the counter can't be cleared but only start again.

T0CK2	T0CK1	T0CK0		4MHz	8MHz	10MHz
0	0	0	($f_{MAIN}÷2$)	500nS	250nS	200nS
0	0	1	($f_{MAIN}÷2^2$)	1uS	500nS	400nS
0	1	0	($f_{MAIN}÷2^3$)	2uS	1uS	800nS
0	1	1	($f_{MAIN}÷2^5$)	8uS	4uS	3.2uS
1	0	0	($f_{MAIN}÷2^7$)	32uS	16uS	12.8uS
1	0	1	($f_{MAIN}÷2^9$)	128uS	64uS	51.2uS
1	1	0	($f_{MAIN}÷2^{11}$)	512uS	256uS	204.8uS

Figure 12-1 Timer0,1,2,3 Registers

T0 (Timer0 Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	T07	T06	T05	T04	T03	T02	T01	T00

ADDRESS: $0D1_H$
INITIAL VALUE:00H

CDR0 (Timer0 Input Capture Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	CDR07	CDR06	CDR05	CDR04	CDR03	CDR02	CDR01	CDR00

ADDRESS: $0D1_H$
INITIAL VALUE:00H

In Timer mode, this register is the value of Timer 0 counter and in Capture mode, this register is the value of input capture.

TDR0 (Timer 0 Data Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	TDR07	TDR06	TDR05	TDR04	TDR03	TDR02	TDR01	TDR00

ADDRESS: $0D1_H$
INITIAL VALUE:FF_H

If the counter of Timer 0 and the data of TDR0 is equal, interrupt is occurred.

T1 (Timer1 Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	T17	T16	T15	T14	T13	T12	T11	T10

ADDRESS: $0D4_H$
INITIAL VALUE:00H

CDR1 (Timer1 Input Capture Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	CDR17	CDR16	CDR15	CDR14	CDR13	CDR12	CDR11	CDR10

ADDRESS: $0D4_H$
INITIAL VALUE:00_H

In Timer mode, this register is the value of Timer 1 counter and in Capture mode, this register is the value of input capture.

TDR1 (Timer1 Data Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	TDR17	TDR16	TDR15	TDR14	TDR13	TDR12	TDR11	TDR10

ADDRESS: $0D3_H$
INITIAL VALUE:FF_H

If the counter of Timer 1 and the data of TDR1 is equal, interrupt is occurred.

Figure 12-2 Related Registers with Timer/Counter0, 1

T2 (Timer2 Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	T27	T26	T25	T24	T23	T22	T21	T20

ADDRESS: $0D7_H$
INITIAL VALUE:00H

CDR2 (Timer2 Input Capture Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	CDR27	CDR26	CDR25	CDR24	CDR23	CDR22	CDR21	CDR20

ADDRESS: $0D7_H$
INITIAL VALUE:00H

In Timer mode, this register is the value of Timer 0 counter and in Capture mode, this register is the value of input capture.

TDR2 (Timer2 Data Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	TDR27	TDR26	TDR25	TDR24	TDR23	TDR22	TDR21	TDR20

ADDRESS: $0D7_H$
INITIAL VALUE:FF_H

If the counter of Timer 0 and the data of TDR0 is equal, interrupt is occurred.

TDR3 (Timer3 Data Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	TDR37	TDR36	TDR35	TDR34	TDR33	TDR32	TDR31	TDR30

ADDRESS: $0D9_H$
INITIAL VALUE:FF_H

If the counter of Timer 1 and the data of TDR1 is equal, interrupt is occurred.

T3PPR (Timer3 PWM Period Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	PWM1PR7	PWM1PR6	PWM1PR5	PWM1PR4	PWM1PR3	PWM1PR2	PWM1PR1	PWM1PR0

ADDRESS: $0D9_H$
INITIAL VALUE:FF_H

The period is decided by PWM.

T3 (Timer3 Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	T37	T36	T35	T34	T33	T32	T31	T30

ADDRESS: $0DA_H$
INITIAL VALUE:00H

CDR3 (Timer1 Input Capture Register)

Bit :	R 7	R 6	R 5	R 4	R 3	R 2	R 1	R 0
	CDR37	CDR36	CDR35	CDR34	CDR33	CDR32	CDR31	CDR30

ADDRESS: $0DA_H$
INITIAL VALUE:00_H

In Timer mode, this register is the value of Timer 1 counter and in Capture mode, this register is the value of input capture.

T3PDR (Timer3 PWM Duty Register)

Bit :	W/R 7	W/R 6	W/R 5	W/R 4	W/R 3	W/R 2	W/R 1	W/R 0
	PWM1DR7	PWM1DR6	PWM1DR5	PWM1DR4	PWM1DR3	PWM1DR2	PWM1DR1	PWM1DR0

ADDRESS: $0DA_H$
INITIAL VALUE:00_H

In PWM mode, decide the pulse duty.

T3PWHR (Timer3 High Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	-	-	-	-	PWM1HR3	PWM1HR2	PWM1HR1	PWM1HR0

ADDRESS: $0DB_H$
INITIAL VALUE:----00000_B

Figure 12-3 Related Registers with Timer/Counter2, 3

16BIT	CAP0	-	T0CK[2:0]	T1CK[1:0]	Timer 0	Timer 1
0	0	-	XXX	XX	8 Bit Timer	8 Bit Timer
0	0	-	111	XX	8 Bit Event Counter	8 Bit Timer
0	1	-	XXX	XX	8 Bit Capture	8 Bit Compare Output
1	0	-	XXX	11	16 Bit Timer	
1	0	-	111	11	16 Bit Event Counter	
1	1	-	XXX	11	16 Bit Capture	
1	0	-	XXX	11	16 Bit Compare Output	

Table 12-1 Operating Modes of Timer 0 and Timer 1

12.1 8-Bit Timer/Counter Mode

The MC80X7208 has four 8-bit Timer/Counters, Timer0, Timer1, Timer2 and Timer3 as shown in .

The “timer” or “counter” function is selected by mode registers TMx (x=0,1,2,3) as shown in and Table 12-1. To use as an 8-bit timer/counter mode, bit CAPx of TMx is cleared to “0” and bits 16BIT of TM1(3) should be cleared to “0” (Table 12-1).

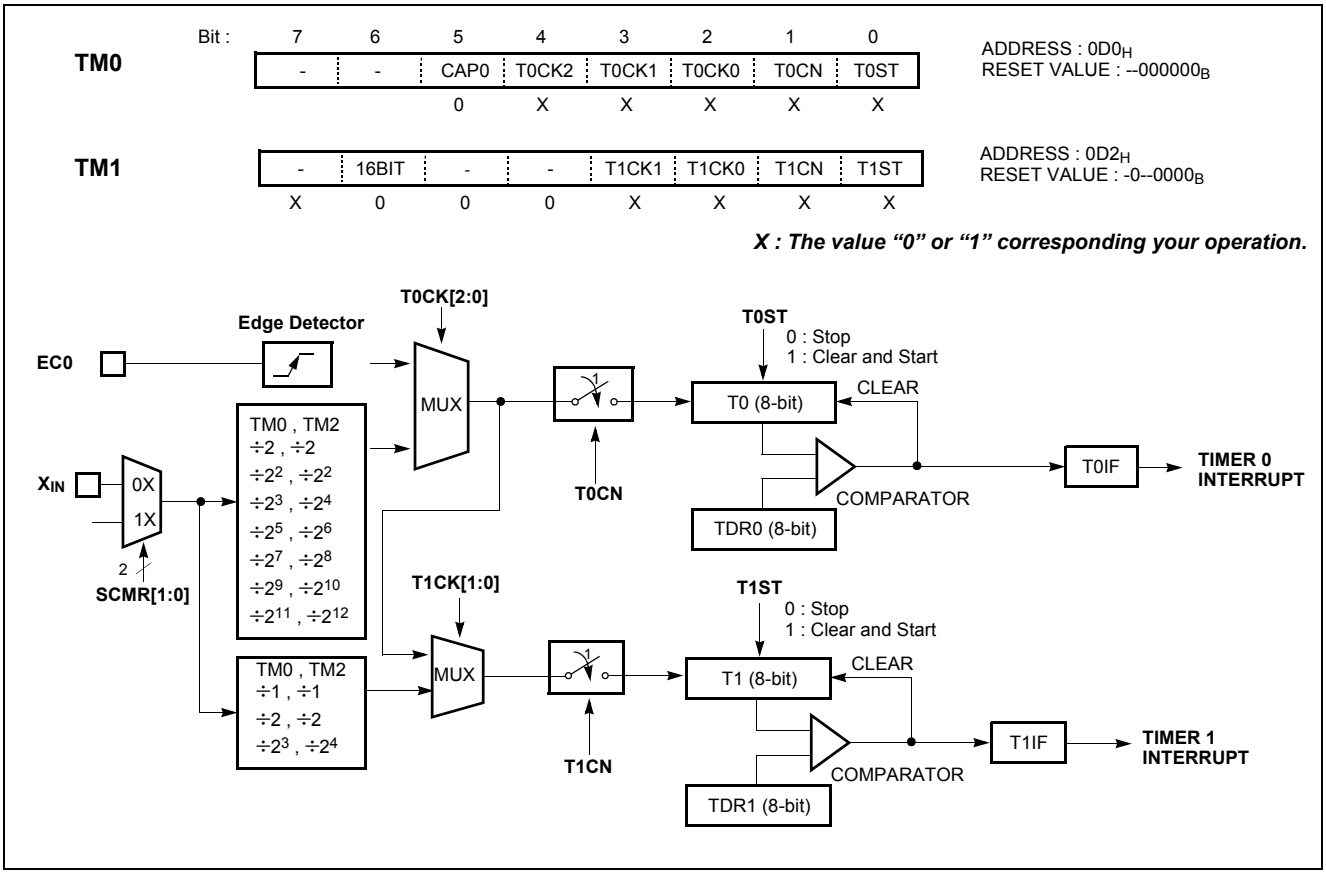


Figure 12-4 Block Diagram of Timer/Event Counter0,1

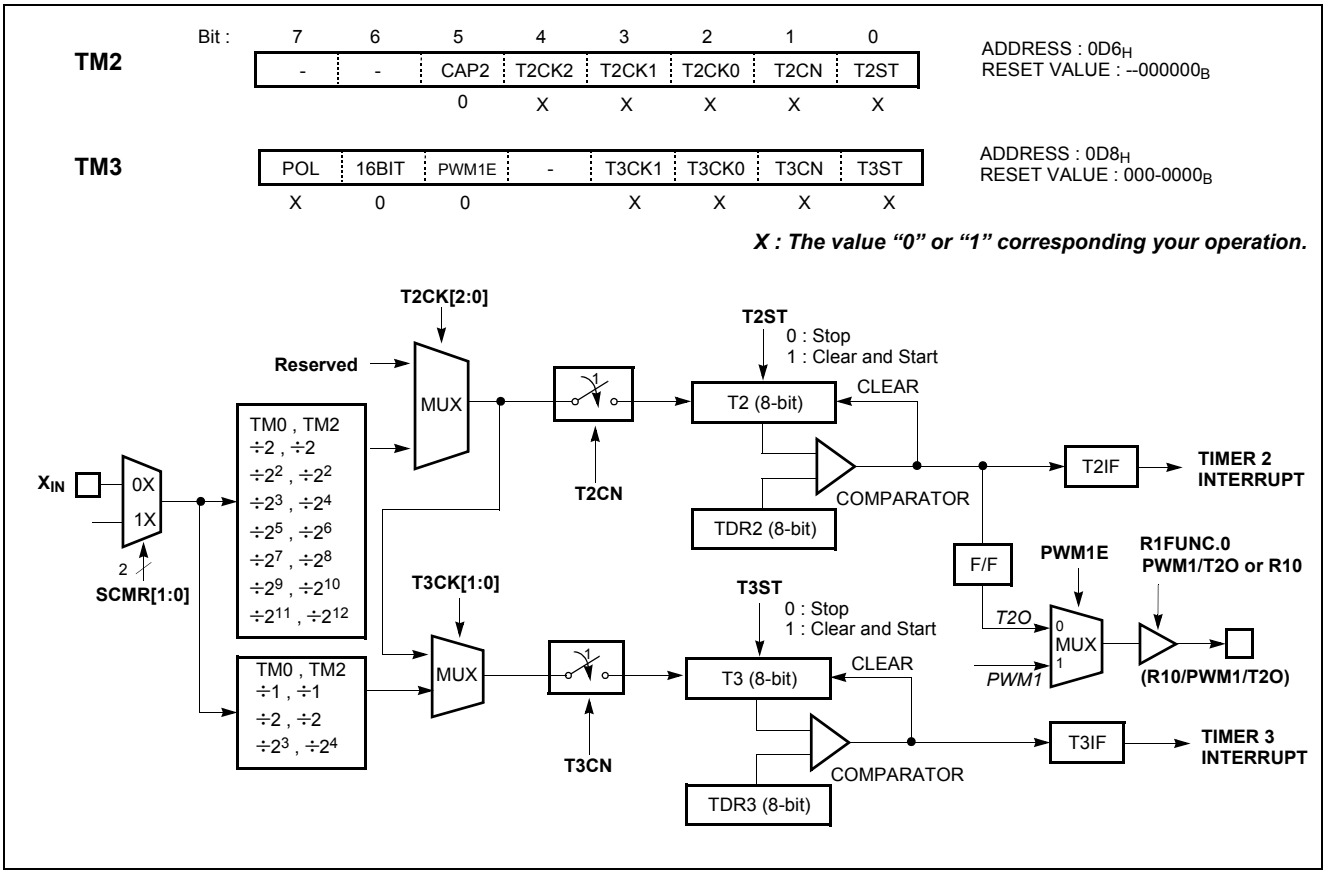


Figure 12-5 Block Diagram of Timer/Event Counter2,3

These timers have each 8-bit count register and data register. The count register is increased by every internal or external clock input. The internal clock has a prescaler divide ratio option of 2, 4, 8, 32,128, 512, 2048 (selected by control bits TxCK2, TxCK1 and TxCK0 of register TM0(2)) and 1, 2, 8 (selected by control bits TxCK1 and TxCK0 of register TM1(3)).

In the Timer, timer register Tx increases from 00_H until it matches TxDR and then reset to 00_H. If the value of Tx is equal with TxDR, Timer x interrupt is occurred (latched in TxIF bit). TxDR and T0 register are in same address, so this register is read from T0 and written to TDR0.

In counter function, the counter is increased every 0-to 1 (rising edge) transition of EC0(1) pin. In order to use counter function, the bit R01(5) of the R0 Direction Register (R0IO) should be set to "0" and the bit EC0(1) of Port Mode Register R0FUNC should set to "1". The Timer 0 and 2 can be used as a counter by pin EC0(1) input, but Timer 1 can not used as a counter.

Note: *The contents of TDR0, TDR1, TDR2 and TDR3 must be initialized (by software) with the value between 1_H and $0FF_H$, not 0_H.*

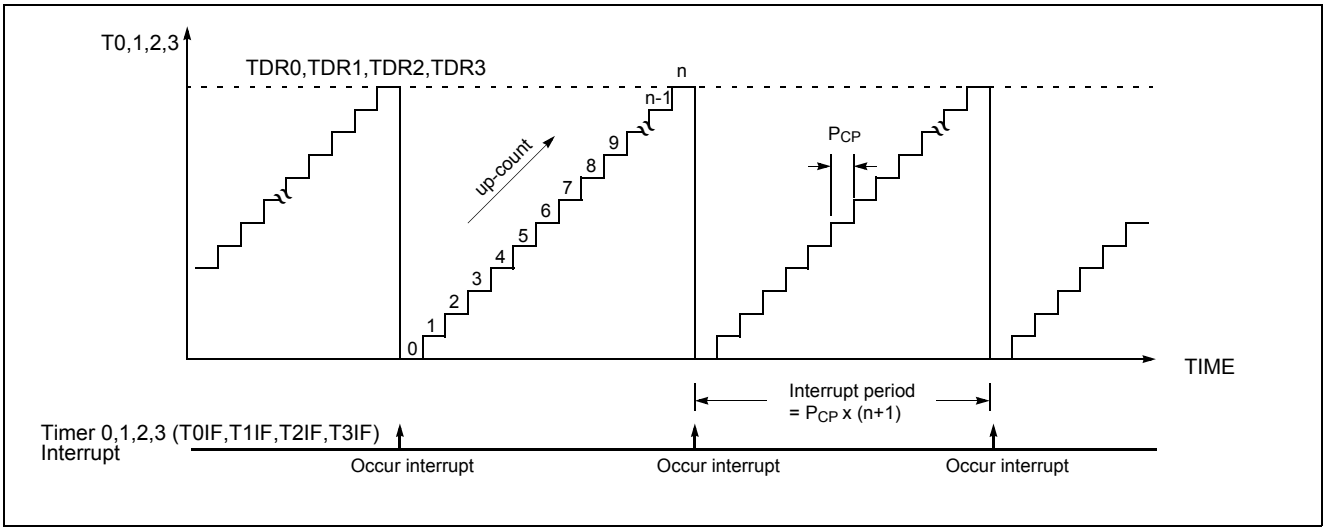


Figure 12-6 Counting Example of Timer Data Registers

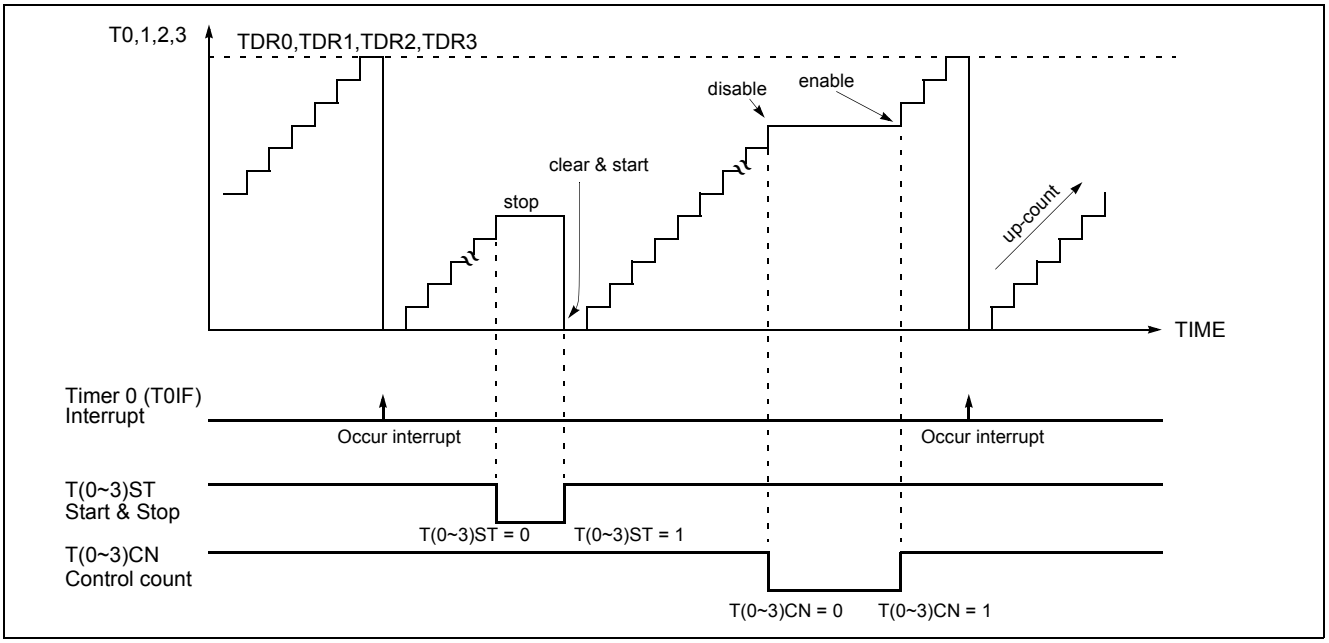


Figure 12-7 Timer Count Operation

12.2 16 Bit Timer/Counter Mode

The Timer register is running with 16 bits. A 16-bit timer/counter register T0, T1 are increased from 0000_H until it matches TDR0, TDR1 and then resets to 0000_H. The match output generates Timer 0 interrupt not Timer 1 interrupt.

The clock source of the Timer 0 is selected either internal or external clock by bit T0CK2, T0CK1 and T0CK0.

In 16-bit mode, the bits T1CK1,T1CK0 and 16BIT of TM1

should be set to “1” respectively.

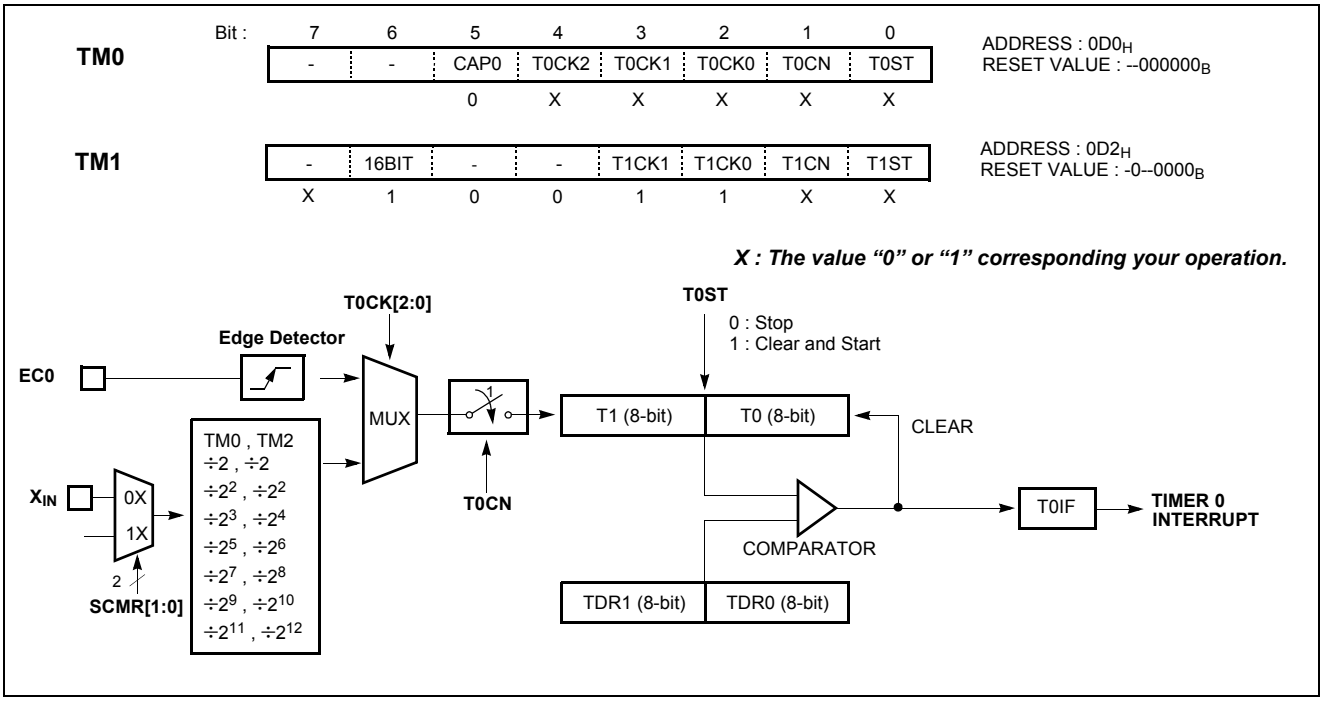


Figure 12-8 16-bit Timer / Counter Mode 0

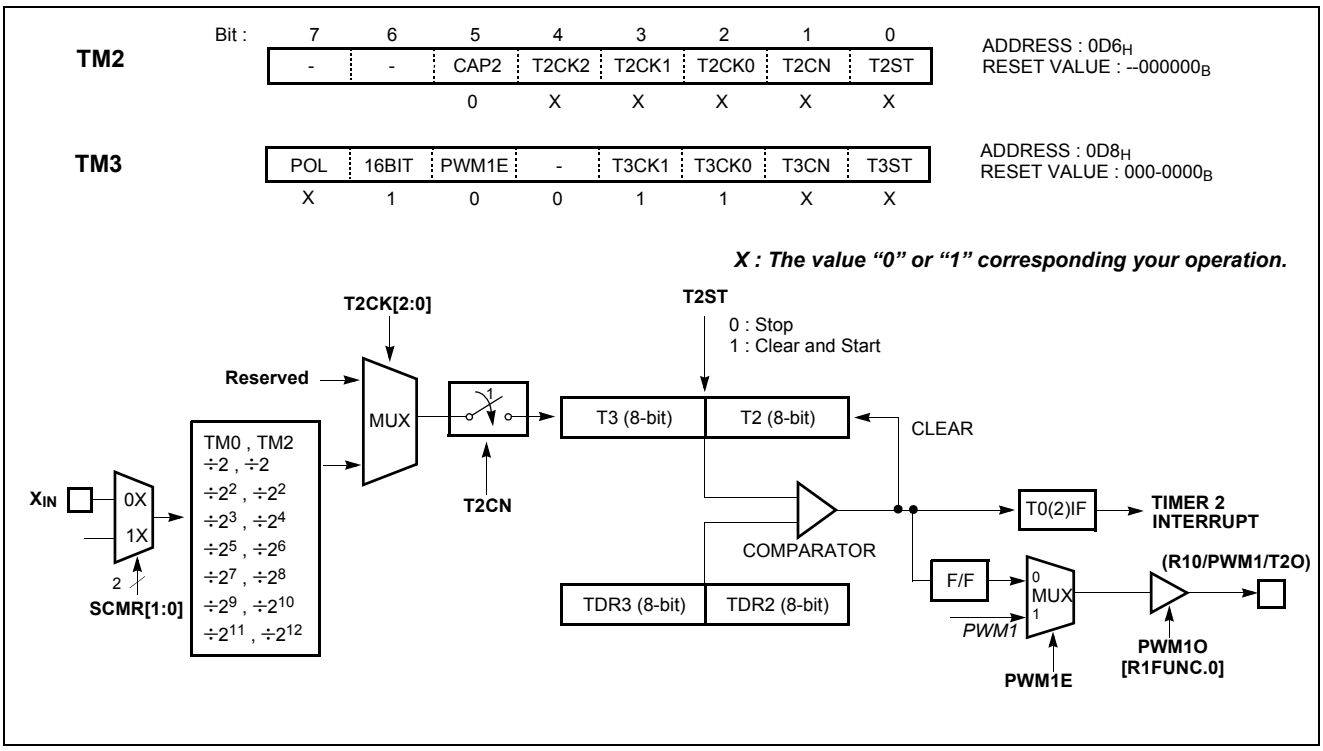


Figure 12-9 16-bit Timer / Counter Mode 2

12.3 8-Bit Capture Mode

The Timer 0 capture mode is set by bit CAP0 of timer mode register TM0 (bit CAPx of timer mode register TMx for Timer 1,2,3) as shown in .

As mentioned above, not only Timer 0 but Timer 1,2,3 can also be used as a capture mode.

The Timer/Counter register is increased in response internal or external input. This counting function is same with normal timer mode, and Timer interrupt is generated when timer register T0 (T1,2,3) increases and matches TDR0 (TDR1,TDR2,TDR3).

This timer interrupt in capture mode is very useful when the pulse width of captured signal is more wider than the maximum period of Timer.

For example, in , the pulse width of captured signal is wider than the timer data value (FF_H) over 2 times. When external interrupt is occurred, the captured value (13_H) is more little than wanted value. It can be obtained correct value by counting the number of timer overflow occurrence.

Timer/Counter still does the above, but with the added feature that a edge transition at external input INTx pin causes the current value in the Timer x register (T0,T1,T2,T3), to be captured into registers CDRx (x=0,1,2,3), respectively. After captured, Timer x register is cleared and restarts by hardware.

It has three transition modes: “falling edge”, “rising edge”, “both edge” which are selected by interrupt edge selection register IEDS (Refer to External interrupt section). In addition, the transition at INTx pin generate an interrupt.

Note: *The CDR0, TDR0 and T0 are in same address. In the capture mode, reading operation is read the CDR0 and in timer mode, reading operation is read the T0. TDR0 is only for writing operation.*
The CDR2, T2, TDR2 are in same address. In the capture mode, reading operation is read the CDR2

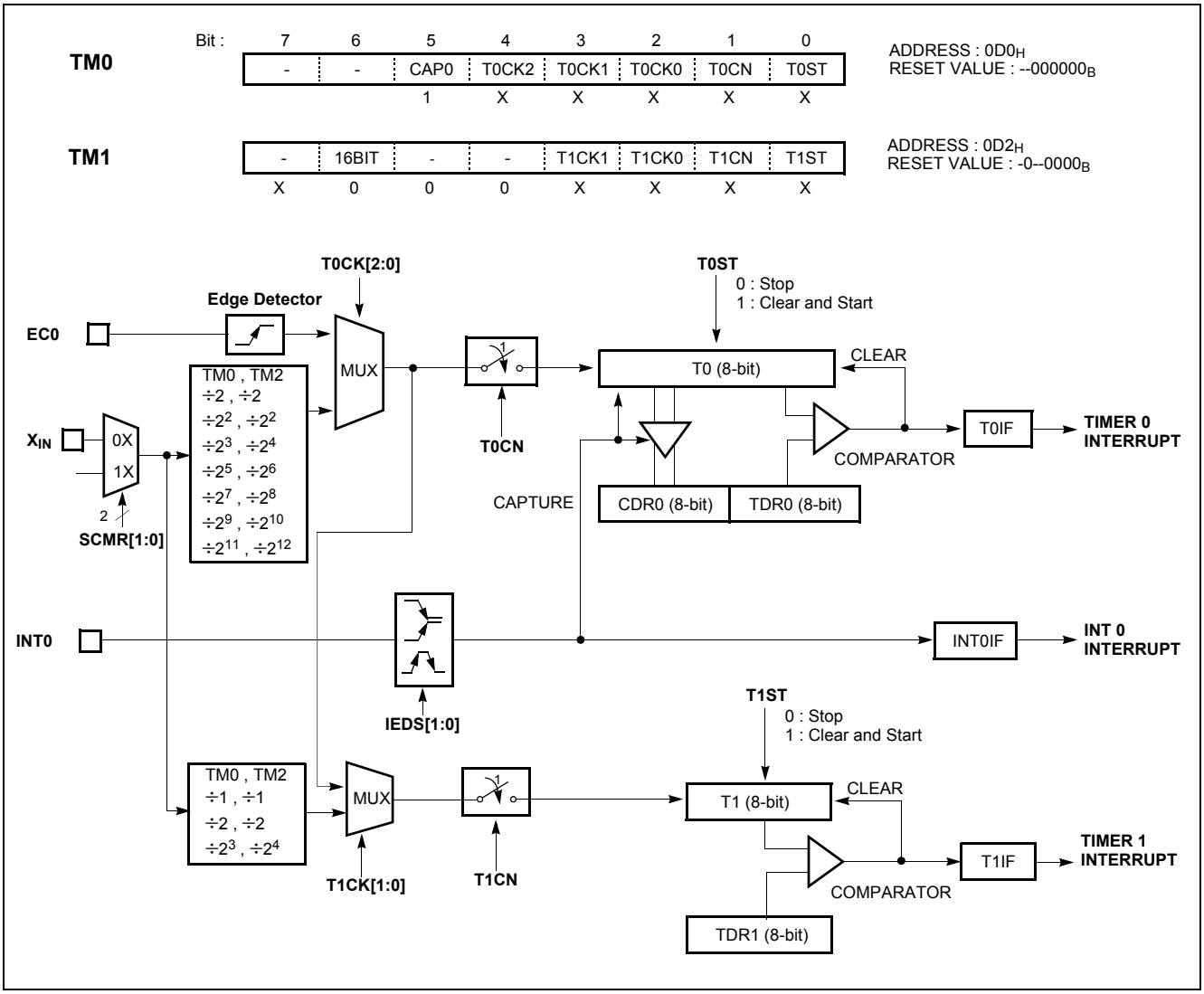


Figure 12-10 8-bit Capture Mode 0

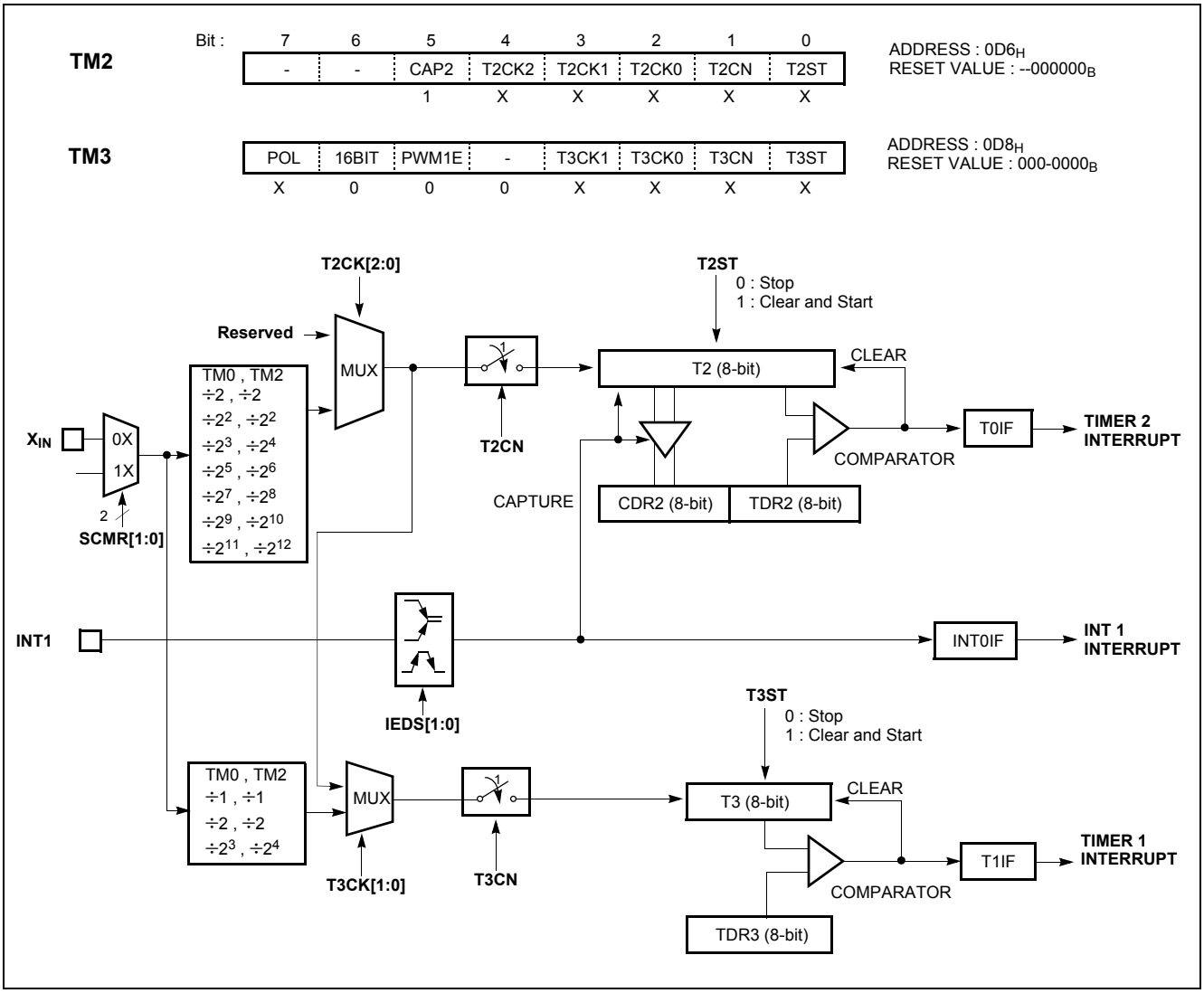


Figure 12-11 8-bit Capture Mode 1

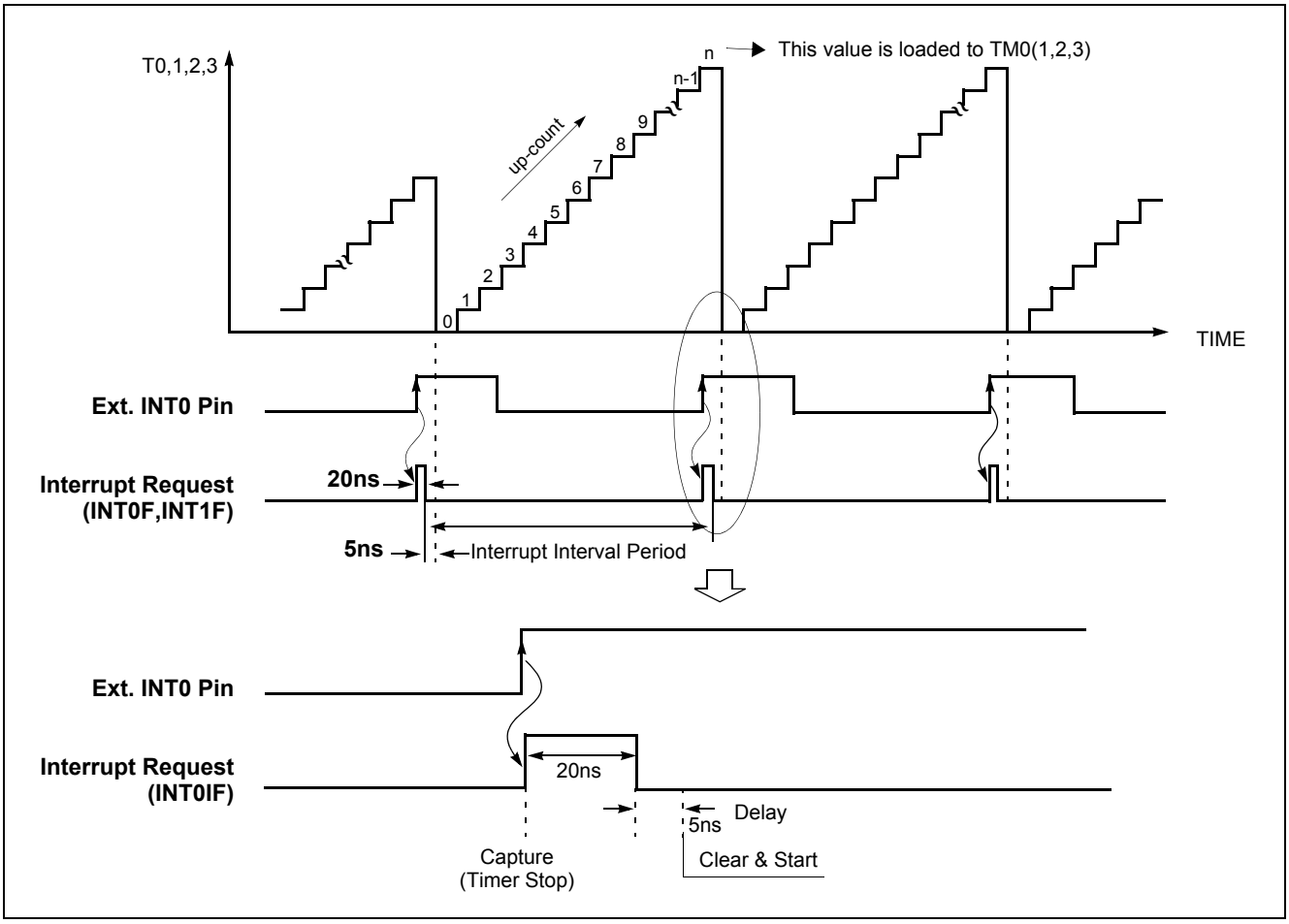


Figure 12-12 Input Capture Operation

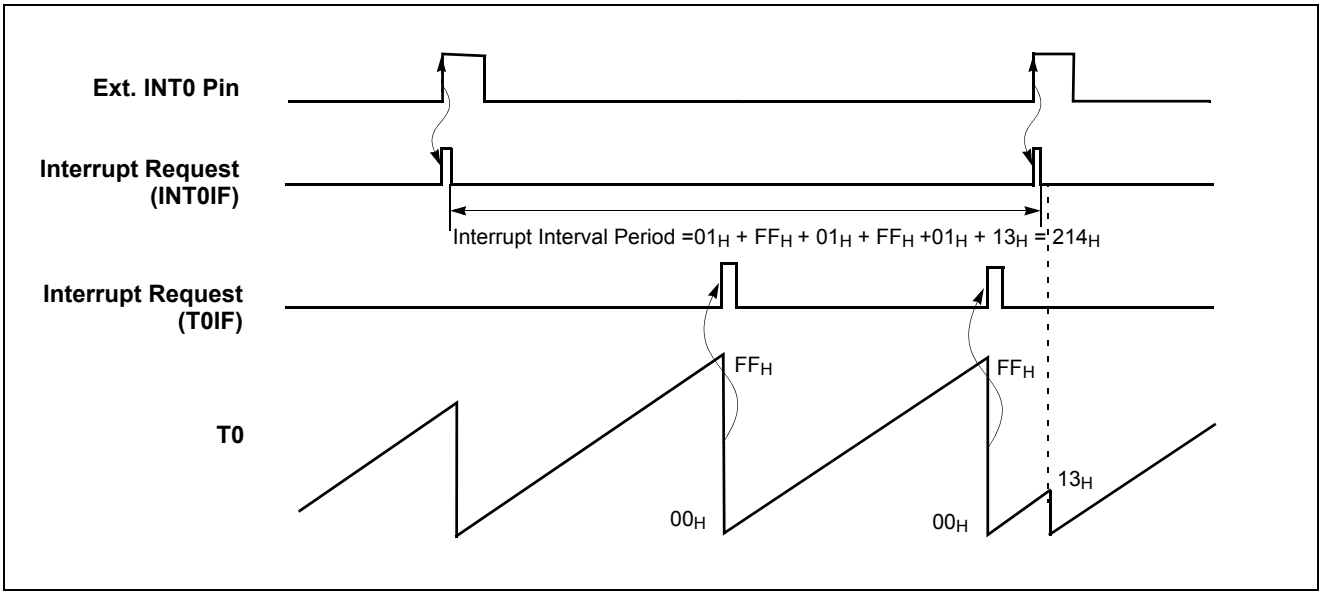


Figure 12-13 Excess Timer Overflow in Capture Mode

12.4 16-bit Capture Mode

16-bit capture mode is the same as 8-bit capture, except that the Timer register is running with 16 bits.

The clock source of the Timer 0,2 is selected either internal or external clock by bit TxCK2, TxCK1 and TxCK0.

In 16-bit mode, the bits TxCK1,TxCK0 and 16BIT of TM1,TM3 should be set to "1" respectively.

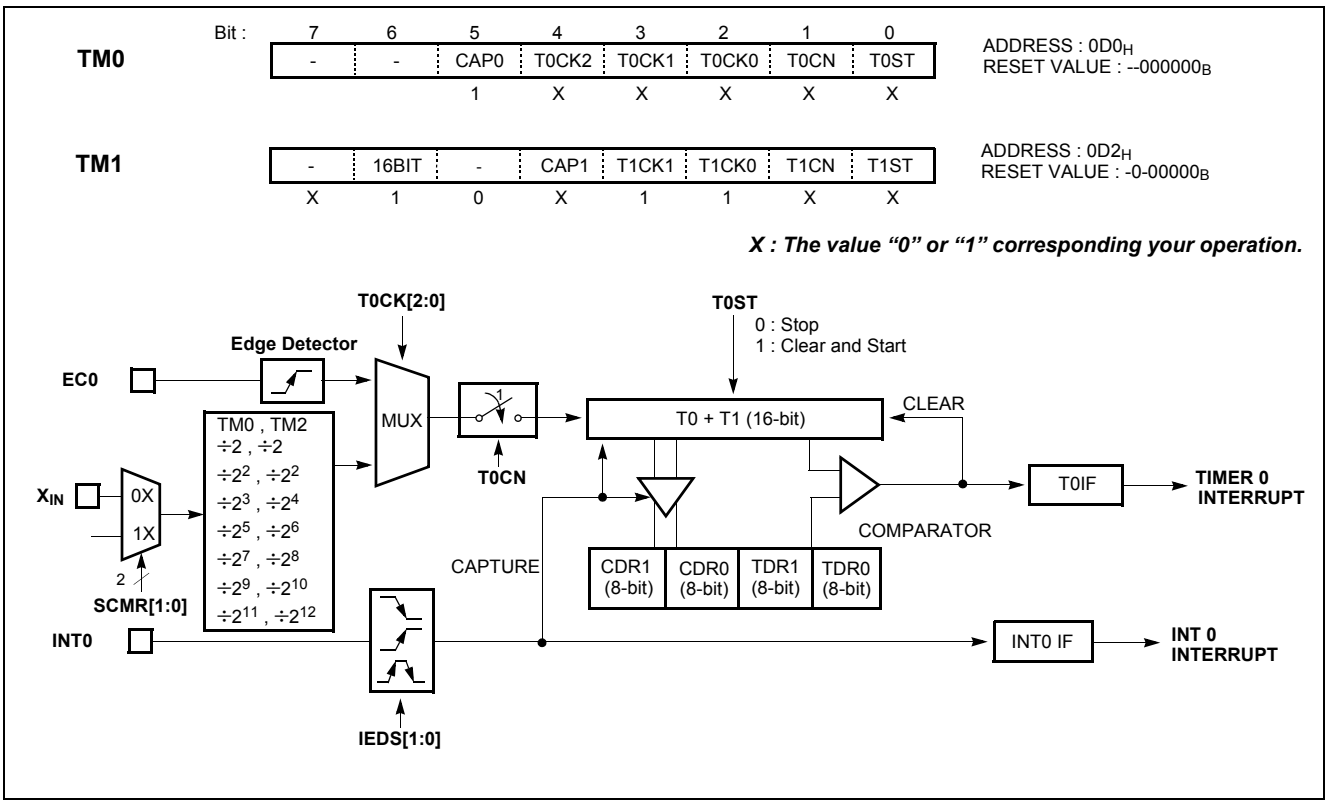


Figure 12-14 16-bit Capture Mode 0

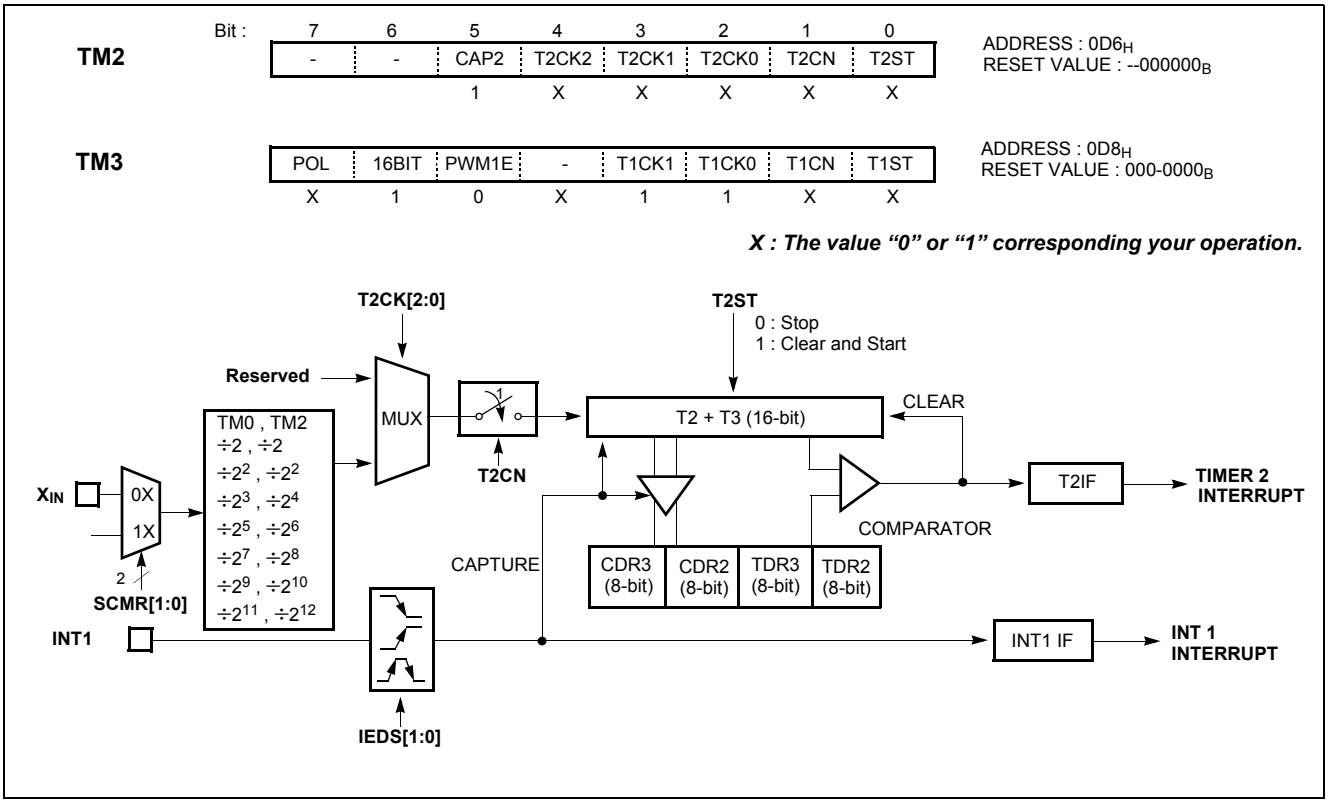


Figure 12-15 16-bit Capture Mode 1

12.5 8-Bit (16-Bit) Compare OutPut Mode

The MC80X7208 has a function of Timer Compare Output. To pulse out, the timer match can goes to port pin (R10) as shown in and . Thus, pulse out is generated by the timer match. These operation is implemented to pin, R10/PWM1/T2O.

In this mode, the bit PWM1O of Port Mode Register R1FUNC should be set to "1", and the bit PWM1E of Timer3 Mode Register (TM3) should be cleared to "0".

In addition, 16-bit Compare output mode is available, also.

This pin output the signal having a 50: 50 duty square wave, and output frequency is same as below equation

$$f_{COMP} = \frac{f_{XIN}}{2 \times PrescalerValue \times (TDR + 1)}$$

12.6 PWM Mode

The MC80X7208 has one high speed PWM (Pulse Width Modulation) function which shared with Timer3. In PWM mode, the R10/PWM1 pins operate as a 10-bit resolution PWM output port. For this mode, the bit PWM1O of Port Mode Register (R1FUNC) and the bit PWM1E of timer3 mode register (TM3) should be set to "1" respectively.

The period of the PWM output is determined by the T3PPR (T3 PWM Period Register) and T3PWHR[3:2] (bit3, 2 of T3 PWM High Register) and the duty of the PWM output is determined by the T3PDR (T3 PWM Duty Register) and T3PWHR[1:0] (bit1, 0 of T3PWM High Register).

The user can use PWM data by writing the lower 8-bit period value to the T3PPR and the higher 2-bit period value to the T3PWHR[3:2]. And the duty value can be used with the T3PDR and the T3PWHR[1:0] in the same way.

The T3PDR is configured as a double buffering for glitchless PWM output. In , the duty data is transferred from the master to the slave when the period data matched to the counted value. (i.e. at the beginning of next duty cycle).

The bit POL1 of TM3 decides the polarity of duty cycle.

The duty value can be changed when the PWM outputs. However the changed duty value is output after the current

period is over. And it can be maintained the duty value at present output when changed only period value shown as . As it were, the absolute duty time is not changed in varying frequency.

Note: *If the user need to change mode from the Timer3 mode to the PWM mode, the Timer3 should be stopped firstly, and then set period and duty register value. If user writes register values and changes mode to PWM mode while Timer3 is in operation, the PWM data would be different from expected data in the beginning.*

The relation of frequency and resolution is in inverse proportion. Table 12-2 shows the relation of PWM frequency vs. resolution.

PWM Period = [T3PWHR[3:2]T3PPR+1] X Source Clock

PWM Duty = [T3PWHR[1:0]T3PDR+1] X Source Clock

If it needed more higher frequency of PWM, it should be reduced resolution.

Note: *If the duty value and the period value are same, the PWM output is determined by the bit POL1 (1: High, 0: Low). And if the duty value is set to "00_H", the PWM output is determined by the bit POL1(1: Low, 0: High). The period value must be same or more than the duty value, and 00_H cannot be used as the period value.*

Resolution	Frequency		
	T3CK[1:0] =00 (250nS)	T3CK[1:0] =01 (500nS)	T3CK[1:0] =10 (2uS)
10-bit	3.9kHz	1.95kHz	0.49kHz
9-bit	7.8kHz	3.9kHz	0.98kHz
8-bit	15.6kHz	7.8kHz	1.95kHz
7-bit	31.2kHz	15.6kHz	3.90kHz

Table 12-2 PWM Frequency vs. Resolution at 4MHz

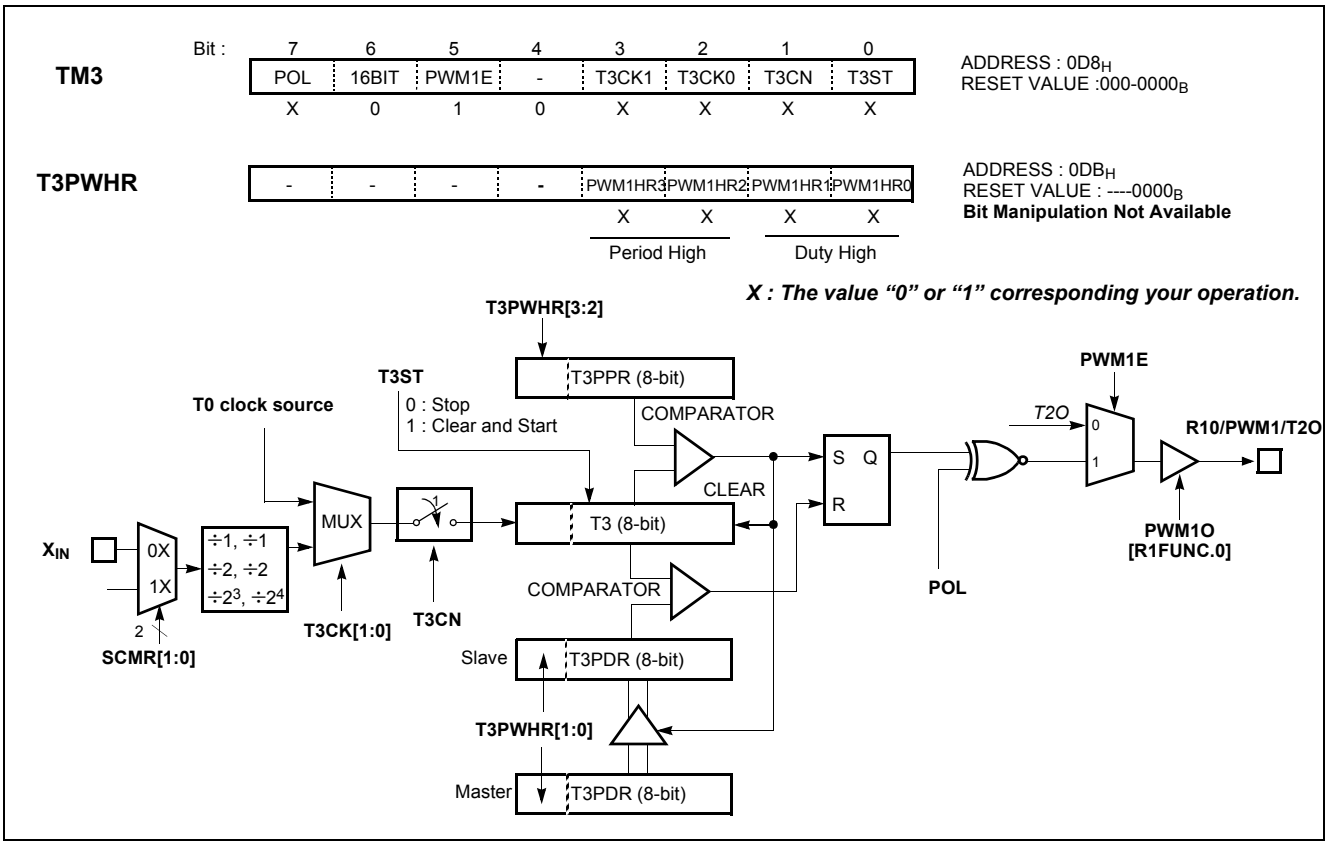


Figure 12-16 PWM Mode

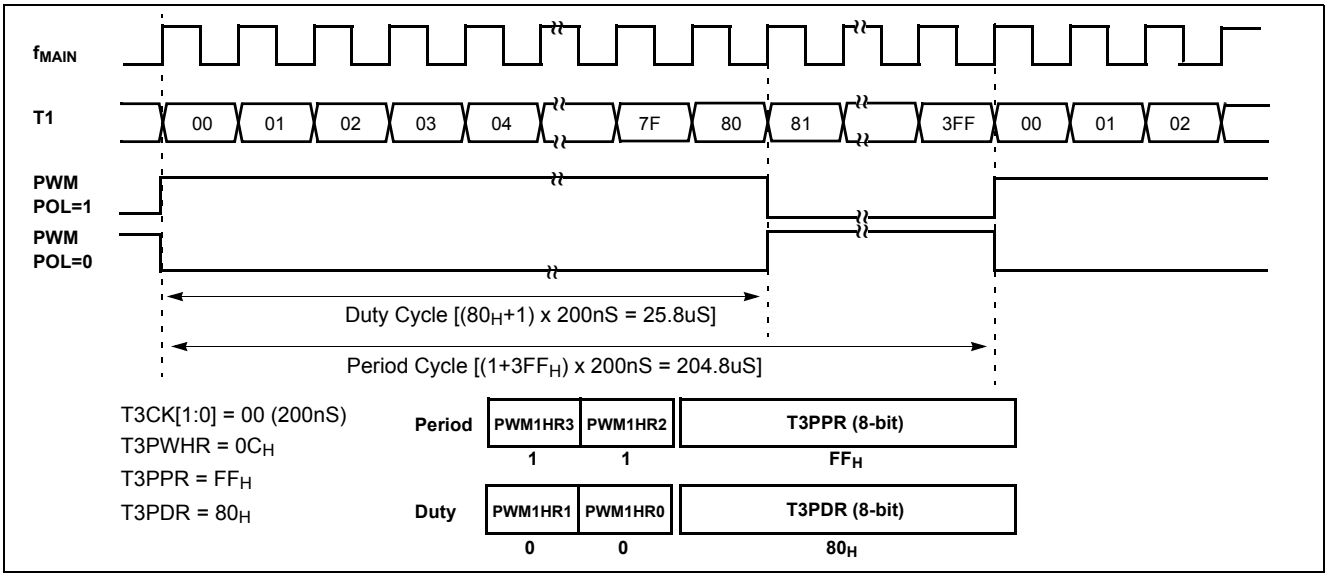


Figure 12-17 Example of PWM at 5MHz

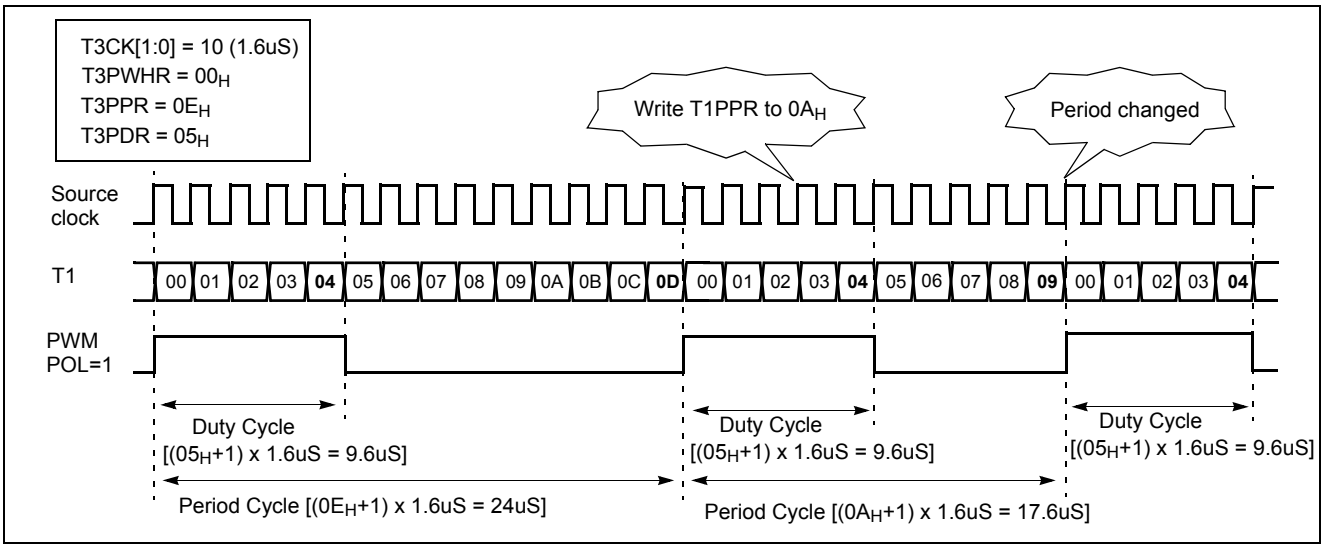


Figure 12-18 Example of Changing the Period in Absolute Duty Cycle (@5MHz)

Example:

```
Timer1 @4Mhz, 4kHz -20% duty PWM mode

   LDM  R1IO,#0000_XXX1B     ;R00 output
   LDM  TM3,#0010_0000B      ;pwm enable
   LDM  T3PWHR,#0000_1100B   ;20% duty
   LDM  T3PPR,#1110_0111B    ;period 250uS
   LDM  T3PDR,#1100_0111B    ;duty 50uS
   LDM  R1FUNC,#XXXX_XXX1B   ;set pwm port.
   LDM  TM3,#0010_0011B      ;timer1 start
```

X means don't care

13. WATCH TIMER

The watch timer generates interrupt for watch operation. The watch timer consists of the clock selector, 21-bit binary counter and watch timer mode register. It is a multi-purpose timer. It is generally used for watch design.

The bit 1, 2 of WTMR select the clock source of watch timer among sub-clock, $f_{MAIN}÷2^7$ of main-clock and f_{MAIN} of main-clock. The f_{MAIN} of main-clock is used usually for watch timer test, so generally it is not used for the clock source of watch timer. The $f_{MAIN}÷2^7$ of main-clock is used when the single clock system is organized. In $f_{MAIN}÷2^7$ clock source, if the CPU enters into stop mode, the main-clock is stopped and then watch timer is also stopped. If the sub-clock is the source clock, the watch timer count cannot be stopped. Therefore, the sub-clock does not stop and continues to oscillate even when the CPU is in the STOP mode. The timer counter consists of 21-bit binary counter and it can count to max 60 seconds at sub-clock.

The bit 2, 3 of WTMR select the interrupt request interval of watch timer among 2Hz, 4Hz, 16Hz and 1/64Hz.

WTR (Watch Timer Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	WTCL	WT6	WT5	WT4	WT3	WT2	WT1	WT0

ADDRESS: $0E8_H$
INITIAL VALUE:0111_1111_B

WTCL (WT Clear)
0: Free Run
1: WT Clear(Auto clear after 1cycle)

WT[6:0] (WT Interrupt Interval Value)
WT Interrupt Interval(IFWT) = (fwck/2^{14}) x (7bit WT Value+1)

WTMR (Watch Timer Mode Register)

Bit :	R/W 7	R/W 6	5	4	R/W 3	R/W 2	R/W 1	R/W 0
	WTEN	LOADEN	-	-	WTIN1	WTIN0	WTCK1	WTCK0

ADDRESS: $0EA_H$
INITIAL VALUE:$00--_0000_B$

WTEN (Watch Timer Enable Bit)
0: Watch Timer Disable
1: Watch Timer Enable

LOADEN (7bit reload Counter Write Enable Bit)
0: Watch Dog Timer Write Enable
1: Watch Timer Write Enable

WTIN[1:0] (Watch Timer Interrupt Interval Selection)
00: $f_{wck} / 2^{11}$ [16Hz]*
01: $f_{wck} / 2^{13}$ [4Hz]*
10: $f_{wck} / 2^{14}$ [2Hz]*
11: $f_{wck} / 2^{14}$ x (7bit WT value+1) [2Hz x (7bit WT value+1)]*

WTCK[1:0] (Watch Timer/LCD Clock Source Selection) : f_{wck}
00: Sub. Clock (f_{SUB})
01: Main Clock ($f_{MAIN}÷2^7$)
10: Main Clock (f_{MAIN})
11: Main Clock ($f_{MAIN}÷2$)

** When f_{SUB} = 32.768 kHz and f_{MAIN} = 4.19 MHz*

Example:

```
; 1 minute watch timer interrupt selection

   LDM  WTMR, #1100_1100B    ; T = 1/fSUB x 2^14 x (count+1)
   LDM  WTR, #1111_0111B     ; 080h + 119(count)
```

Figure 13-1 Watch Timer Mode Register

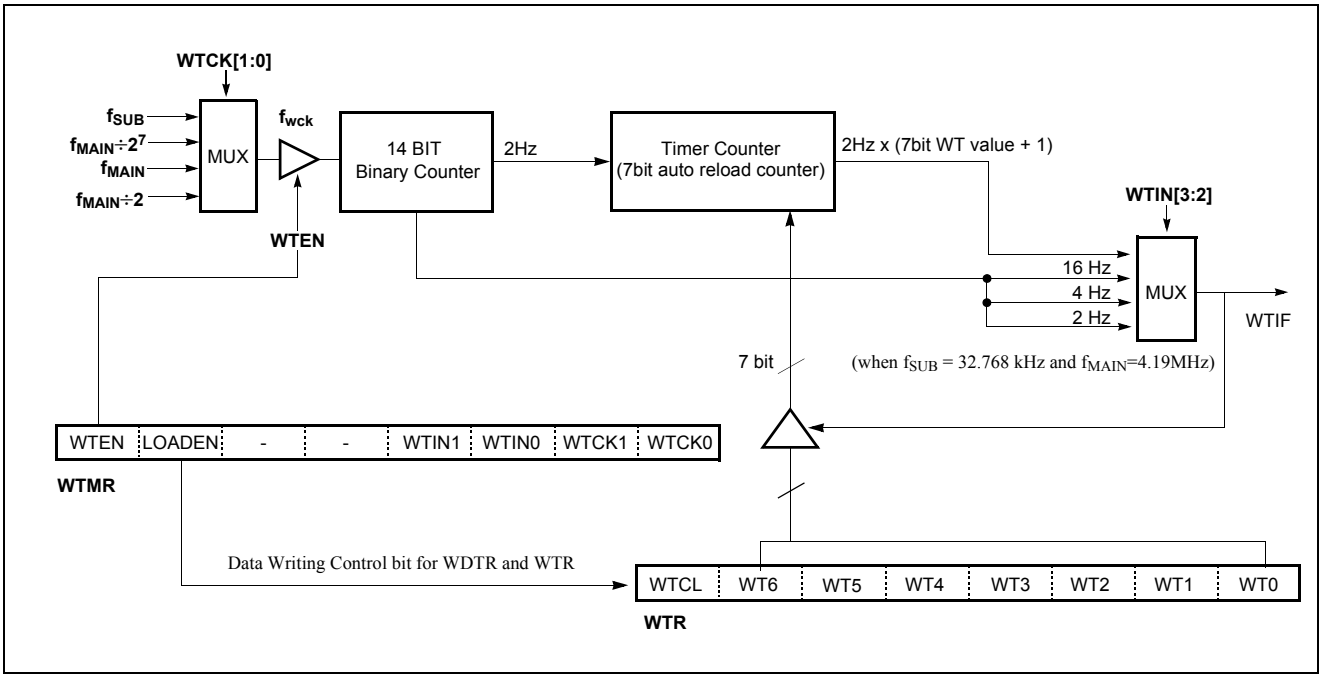


Figure 13-2 Watch Timer Block Diagram

Usage of Watch Timer in STOP Mode

When the system is off and the watch should be kept working, follow the steps below.

1. Determines which mode is to be performed between main mode and sub mode when the MCU is released from Stop mode and set the clock source of watch timer to sub-clock.
2. Enters into STOP mode.
3. After released by watch timer interrupt, counts up timer and refreshes LCD Display. When performing count up and refresh the LCD, the CPU may operate either in main frequency mode or sub frequency mode.
4. Enters into STOP mode again.
5. Repeats 3 and 4.

When using STOP mode, if the watch timer interrupt interval is selected to 2Hz, the power consumption can be reduced considerably.

14. WATCH DOG TIMER

The watch dog timer (WDT) function is used for checking program malfunction due to external noise or other causes and return the operation to the normal contion.

The watchdog timer consists of 7-bit binary counter and the watchdog timer register(WDTR). The source clock of WDT is overflow of Basic Interval Timer. When the value of 7-bit binary counter is equal to the lower 7-bits of WDTR, the interrupt request flag is generated. This can be used as WDT interrupt or CPU reset signal in accordance with the bit WDTON. When WDTCL is set, 7-bit counter of WDT is reset. After one cycle, it is cleared by hardware. When LOADEN of watch timer mode register(WTMR) is set, WDTR can not be wrote.

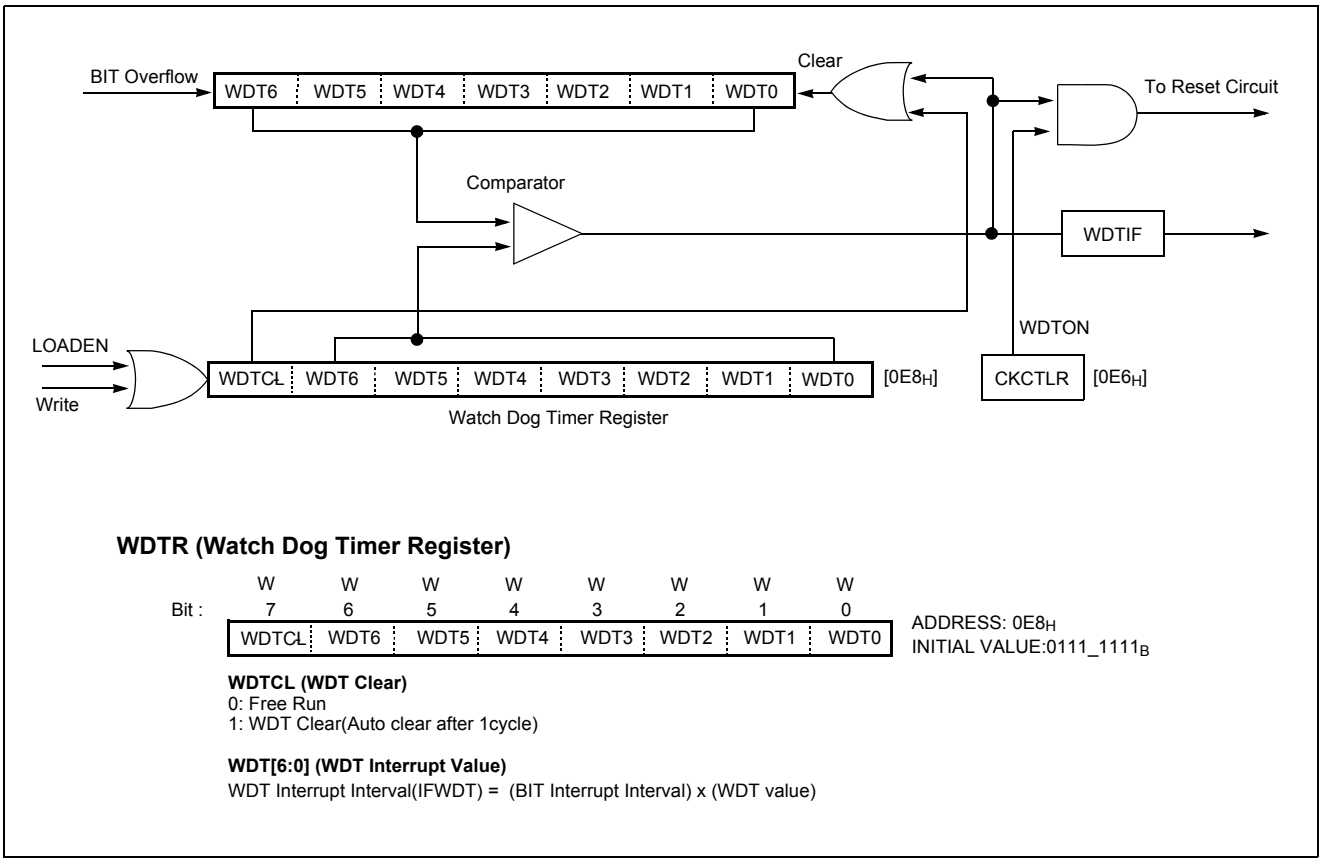


Figure 14-1 Block Diagram of Watch Dog Timer

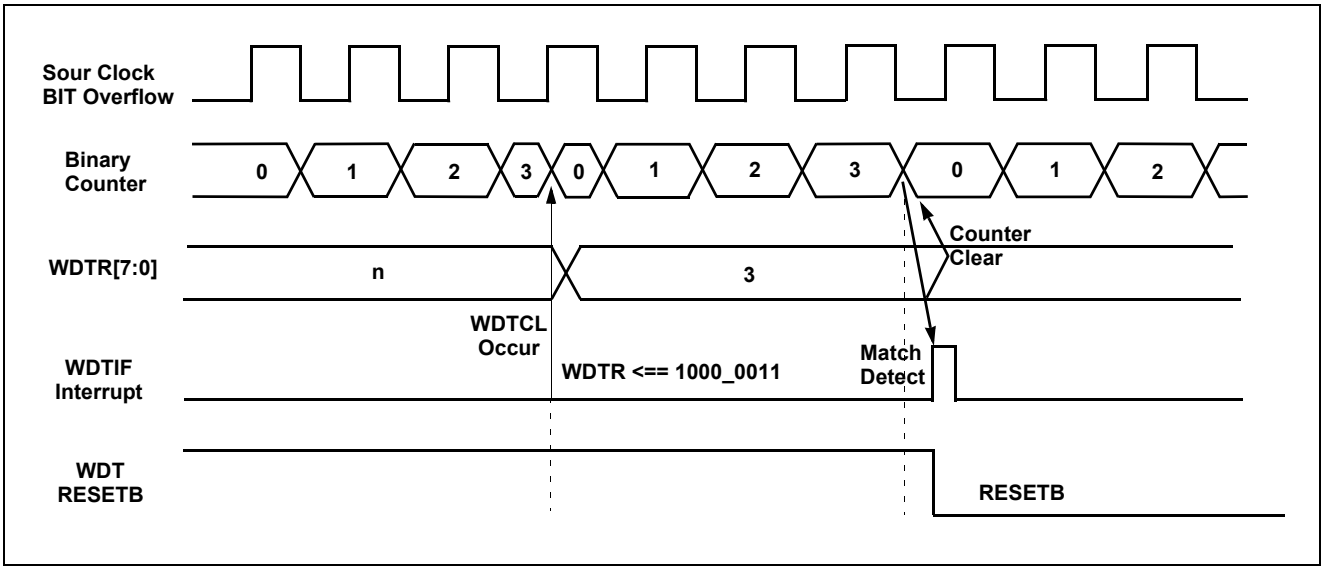


Figure 14-2 Watch Dog Timer Interrupt Time

15. ANALOG TO DIGITAL CONVERTER

The analog-to-digital(A/D) converter allows conversion of an analog input signal to a corresponding 8-bit digital value. The A/D module has six analog inputs, which are multiplexed into one sample and hold. The output of the sample and hold is the input of the converter, which generates the result via successive approximation. The analog supply voltage is connected to AV_{DD} of ladder resistance of A/D module.

The A/D module has two registers which are the A/D converter mode register (ADCM) and A/D converter result register (ADCR). The ADCM register, shown in , controls the operation of the A/D converter module. The port pins can be configured as analog inputs or digital I/O. To use analog inputs, each port should be assigned analog input port by setting R2IO direction register as input mode. And select the corresponding channel to be converted by setting ADS[2:0].

The processing of conversion is start when the start bit ADST is set to "1". After one cycle, it is cleared by hardware. The register ADCR contains the result of the A/D conversion. When the conversion is completed, the result is loaded into the ADCR, the A/D conversion status bit ADF is set to "1", and the A/D interrupt flag ADIF is set. The block diagram of the A/D module is shown in . The A/D status bit ADF is automatically set when A/D conversion is completed, cleared when A/D conversion is in process. The conversion is completed on 10us@8MHz.

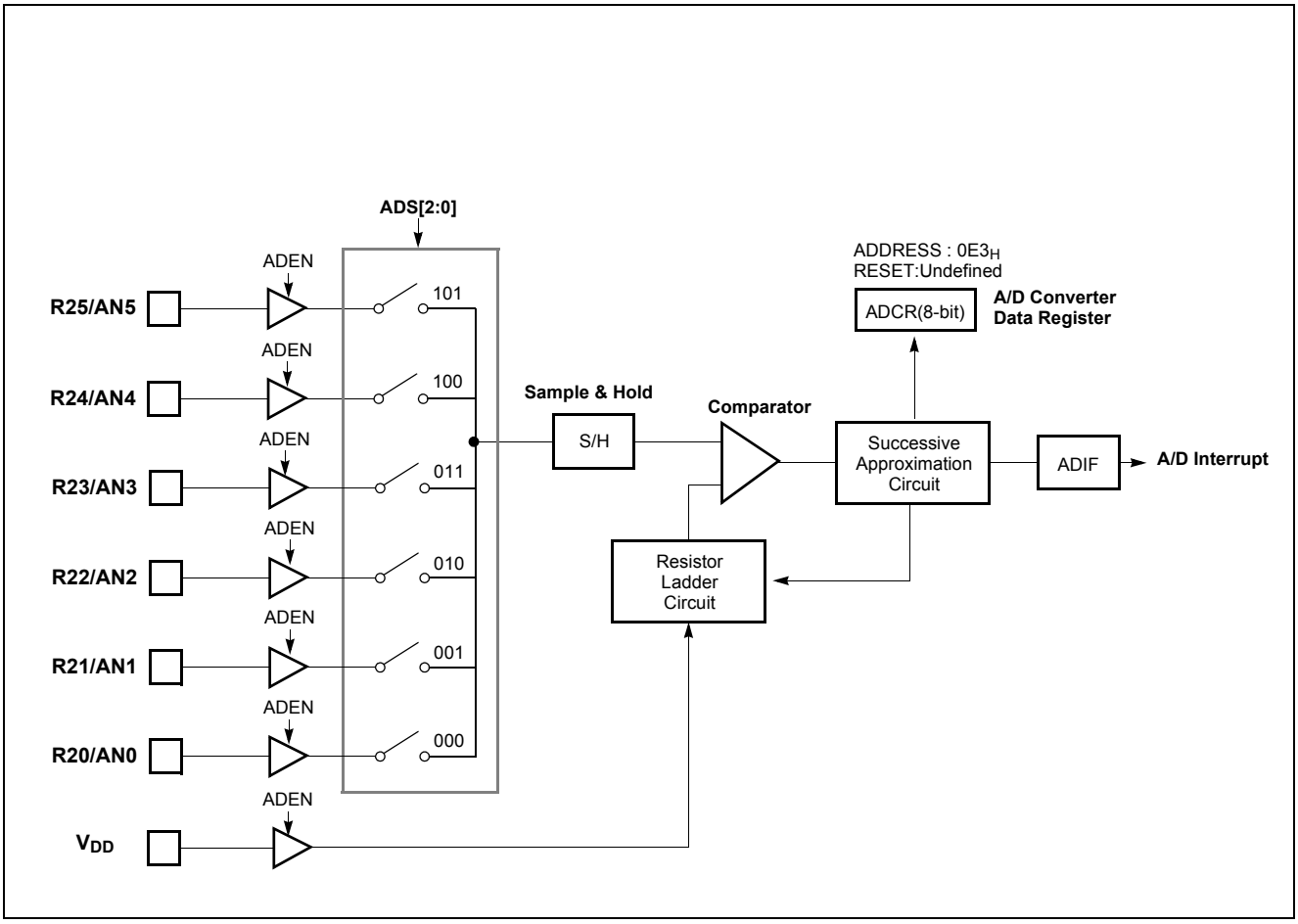


Figure 15-1 A/D Converter Block Diagram & Registers

ADCM (A/D Converter Mode Register)

Bit :	7	6	5	4	3	2	1	0
	R/W	R/W		R/W	R/W	R/W	R/W	R
	-	ADEN	-	ADS2	ADS1	ADS0	ADST	ADF

ADDRESS : $0E2_H$
RESET VALUE : $00\text{-}00001_B$

ADEN (A/D Converter Enable bit)
1 : Enable
0 : Disable

ADST (A/D Start bit)
1 : A/D Conversion is started
After 1 cycle, cleared to "0"
0 : Bit force to zero

ADF (A/D Status bit)
0 : A/D Conversion is in process
1 : A/D Conversion is completed

ADS[2:0] (A/D Converter Input Selection)
000 : Channel 0 (R20/AN0)
001 : Channel 1 (R21/AN1)
010 : Channel 2 (R22/AN2)
011 : Channel 3 (R23/AN3)
100 : Channel 4 (R24/AN4)
101 : Channel 5 (R25/AN5)
110 : Reserved
110 : Reserved

ADCR (A/D Converter Result Register)

Bit :	7	6	5	4	3	2	1	0
	R	R	R	R	R	R	R	R
	ADR7	ADR6	ADR5	ADR4	ADR3	ADR2	ADR1	ADR0

ADDRESS : $0E3_H$
RESET VALUE : Undefined

Figure 15-2 A/D Converter Mode & Result Registers

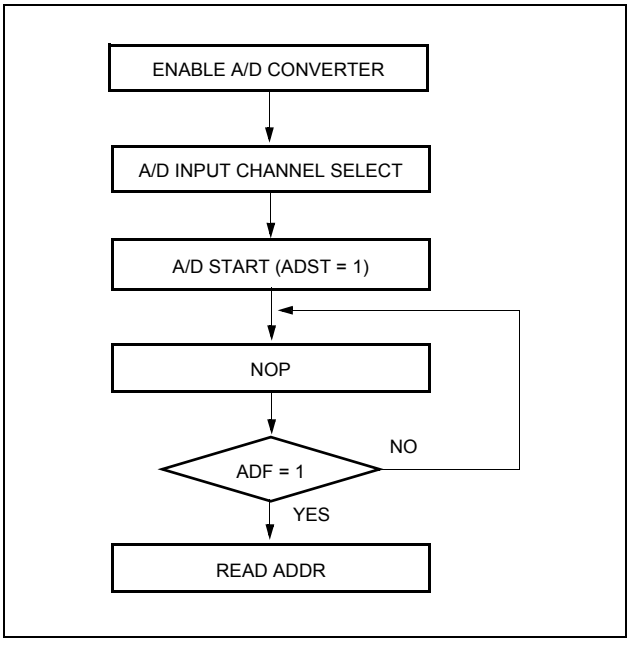


Figure 15-3 A/D Converter Operation Flow

A/D Converter Cautions

(1) Input range of AN0 to AN5

The input voltages of AN0 to AN7 should be within the specification range. In particular, if a voltage above AV_{DD} or below V_{SS} is input (even if within the absolute maximum rating range), the conversion value for that channel can not be determinated. The conversion values of the other channels may also be affected.

(2) Noise counter measures

In order to maintain 8-bit resolution, any attention must be paid to noise on pins AV_{DD} and AN0 to AN7. Since the effect increases in proportion to the output impedance of the analog input source, it is recommended that a capacitor is connected externally as shown below in order to reduce noise.

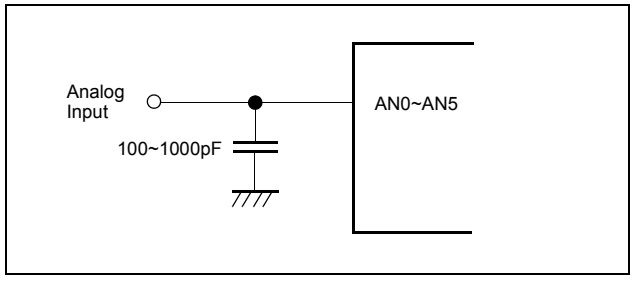


Figure 15-4 Analog Input Pin Connecting Capacitor

(3) Pins AN0/R20 to AN5/R25

The analog input pins AN0 to AN5 also function as input/output port (PORT R2) pins. When A/D conversion is performed with any of pins AN0 to AN57 selected, be sure not

to execute a PORT input instruction while conversion is in progress, as this may reduce the conversion resolution.

Also, if digital pulses are applied to a pin adjacent to the pin in the process of A/D conversion, the expected A/D conversion value may not be obtainable due to coupling noise. Therefore, avoid applying pulses to pins adjacent to the pin undergoing A/D conversion.

(4) AV_{DD} pin input impedance

A series resistor string of approximately 10KΩ is connected between the AV_{DD} pin and the V_{SS} pin.

Therefore, if the output impedance of the reference voltage source is high, this will result in parallel connection to the series resistor string between the AV_{DD} pin and the V_{SS} pin, and there will be a large reference voltage error.

16. BUZZER OUTPUT FUNCTION

The buzzer driver consists of 6-bit binary counter, the buzzer driver register BUZR and the clock selector. It generates square-wave which is very wide range frequency (500 Hz~125 kHz at f_{MAIN} = 4MHz) by user programmable counter.

Pin R04/BUZO is assigned for output port of Buzzer driver by setting the bit BUZO of R0 Function Register(R0FUNC) to "1".

The 6-bit buzzer counter is cleared and start the counting by writing signal to the register BUZR. It is increased from 00_H until it matches with BUR[5:0].

Also, it is cleared by counter overflow and count up to output the square wave pulse of duty 50%.

The bit 0 to 5 of BUZR determines output frequency for buzzer driving. BUZR[5:0] is initialized to $3F_H$ after reset. Note that BUZR is a write-only register. Frequency calculation is following as shown below.

$$f_{BUZ} = \frac{f_{XIN}}{2 \times DivideRatio \times (BUZR[5:0] + 1)}$$

The bits BUCK1, BUCK0 of BUZR select the source clock from prescaler output.

f_{BUZ}: Buzzer frequency
f_{XIN}: Oscillator frequency
Divide Ratio: Prescaler divide ratio by BUCK[1:0]
BUZR[5:0]: Lower 6-bit value of BUZR. Buzzer control data
.

BUZR (Buzzer Driver Register)

Bit :	W 7	W 6	W 5	W 4	W 3	W 2	W 1	W 0
	BUCK1	BCUK0	BUR5	BUR4	BUR3	BUR2	BUR1	BUR0

ADDRESS : $0CE_H$
RESET VALUE : FF_H
Bit manipulation is not available.

BUCK<1:0> (Buzzer Clock Source)
00: $f_{MAIN} \div 2^3$
01: $f_{MAIN} \div 2^4$
10: $f_{MAIN} \div 2^5$
11: $f_{MAIN} \div 2^6$

BUR[5:0] (Buzzer Control Data)

R0 Function Selection Register

R0FUNC	INT1E	INT0E	-	BUZO	RX0I	TX0O	EC0E	-

ADDRESS : $0AA_H$
RESET VALUE : 00-0000-$_B$

BUZO (Buzzer Output)
0 : R04 Port (turn off buzzer)
1 : BUZO port (turn on buzzer)

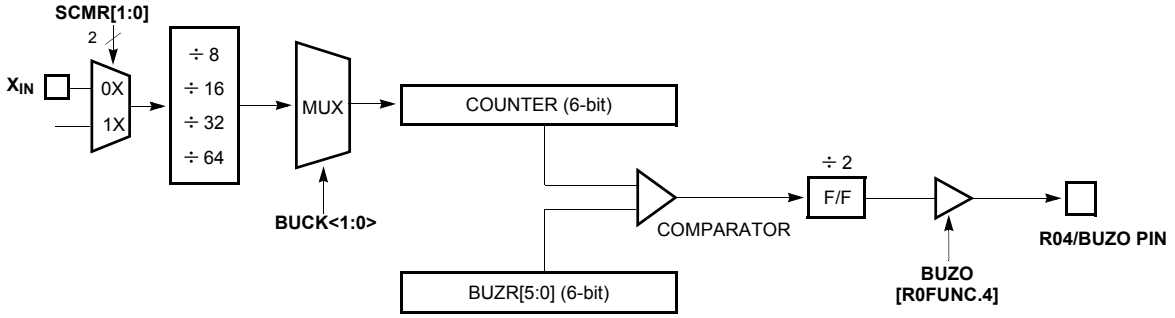


Figure 16-1 Buzzer Driver

Example: 2.5kHz output at 4MHz.

```
LDM     R0FUNC,#XXX1_XXXXB
LDM     BUZR,#1001_1000B
```

X means don't care

Buzzer Output Frequency

When main-frequency is 4MHz, buzzer frequency is shown as below and if sub-frequency is selected as clock source, buzzer frequency is used after dividing by 128.

BUZR [5:0]	Frequency Output (kHz) BUZR[7:6]			
	00	01	10	11
00	250.000	125.000	62.500	31.250
01	125.000	62.500	31.250	15.625
02	83.333	41.667	20.833	10.417
03	62.500	31.250	15.625	7.813
04	50.000	25.000	12.500	6.250
05	41.667	20.833	10.417	5.208
06	35.714	17.857	8.929	4.464
07	31.250	15.625	7.813	3.906
08	27.778	13.889	6.944	3.472
09	25.000	12.500	6.250	3.125
0A	22.727	11.364	5.682	2.841
0B	20.833	10.417	5.208	2.604
0C	19.231	9.615	4.808	2.404
0D	17.857	8.929	4.464	2.232
0E	16.667	8.333	4.167	2.083
0F	15.625	7.813	3.906	1.953
10	14.706	7.353	3.676	1.838
11	13.889	6.944	3.472	1.736
12	13.158	6.579	3.289	1.645
13	12.500	6.250	3.125	1.563
14	11.905	5.952	2.976	1.488
15	11.364	5.682	2.841	1.420
16	10.870	5.435	2.717	1.359
17	10.417	5.208	2.604	1.302
18	10.000	5.000	2.500	1.250
19	9.615	4.808	2.404	1.202
1A	9.259	4.630	2.315	1.157
1B	8.929	4.464	2.232	1.116
1C	8.621	4.310	2.155	1.078
1D	8.333	4.167	2.083	1.042
1E	8.065	4.032	2.016	1.008
1F	7.813	3.906	1.953	0.977

BUZR [5:0]	Frequency Output (kHz) BUZR[7:6]			
	00	01	10	11
20	7.576	3.788	1.894	0.947
21	7.353	3.676	1.838	0.919
22	7.143	3.571	1.786	0.893
23	6.944	3.472	1.736	0.868
24	6.757	3.378	1.689	0.845
25	6.579	3.289	1.645	0.822
26	6.410	3.205	1.603	0.801
27	6.250	3.125	1.563	0.781
28	6.098	3.049	1.524	0.762
29	5.952	2.976	1.488	0.744
2A	5.814	2.907	1.453	0.727
2B	5.682	2.841	1.420	0.710
2C	5.556	2.778	1.389	0.694
2D	5.435	2.717	1.359	0.679
2E	5.319	2.660	1.330	0.665
2F	5.208	2.604	1.302	0.651
30	5.102	2.551	1.276	0.638
31	5.000	2.500	1.250	0.625
32	4.902	2.451	1.225	0.613
33	4.808	2.404	1.202	0.601
34	4.717	2.358	1.179	0.590
35	4.630	2.315	1.157	0.579
36	4.545	2.273	1.136	0.568
37	4.464	2.232	1.116	0.558
38	4.386	2.193	1.096	0.548
39	4.310	2.155	1.078	0.539
3A	4.237	2.119	1.059	0.530
3B	4.167	2.083	1.042	0.521
3C	4.098	2.049	1.025	0.512
3D	4.032	2.016	1.008	0.504
3E	3.968	1.984	0.992	0.496
3F	3.906	1.953	0.977	0.488

Table 16-1 Buzzer Output Frequency

17. INTERRUPTS

The MC80X7208 interrupt circuits consist of Interrupt enable register (IENH, IENM, IENL), Interrupt request flag register(IRQH, IRQM, IRQL), Interrupt flag register(INTFH, INTFL), Interrupt Edge Selection Register (IEDS), priority circuit and Master enable flag ("I" flag of PSW). The interrupts are controlled by the interrupt master enable flag I-flag (bit 2 of PSW), the interrupt enable register and the interrupt request flag register except Power-on reset and software BRK interrupt. The configuration of interrupt circuit is shown in and interrupt priority is shown in Table .

Reset/Interrupt	Symbol	Priority	Vector Addr.
Hardware Reset	RESET	-	$FFFE_H$
Key Scan Int.	KS	1	$FFFC_H$
External Int. 0	INTR0	2	$FFFA_H$
External Int. 1	INTR1	3	$FFF8_H$
-	INTR2	-	
UART_RX0[1]	**RX0**	4	**$FFF4_H$**
UART_TX0[1]	**TX0**	5	**$FFF2_H$**
-	RX1	-	
-	TX1	-	
Timer 0 Int.	T0	6	$FFEE_H$
Timer 1 Int.	T1	7	$FFEC_H$
Timer 2 Int.	T2	8	$FFEA_H$
Timer 3 Int.	T3	9	$FFE8_H$
Carrier Generator Int.	CG	10	$FFE6_H$
A/D Int.	ADC	11	$FFE4_H$
BIT Int.	BIT	12	$FFE2_H$
Watch Dog timer int.	WDT	13	$FFE0_H$
Watch timer int.	WT	14	

Table 17-1 Vector Table

1. **ONLY FLASH MCU IS AVAILABLE.**

Each bit of interrupt request flag registers(IRQH, IRQM, IRQL) in is set when corresponding interrupt condition is met. The interrupt request flags that actually generate external interrupts are bit INT0F, INT1F and INT2F in Register IRQH. The External Interrupts INT0, INT1 and INT2 can each be transition-activated (1-to-0, 0-to-1 and both transition). The Key Scan Interrupt is generated by KSIF if any one or more of key scan input is low signal. The RX0, RX1, TX0 and TX1 of UART0,1 Interrupts are generated by RX0IF,RX1IF,TX0IF and TX1IF which are set by finishing the reception and transmission of data.

The Timer 0,1,2 and Timer 3 Interrupts are generated by T0IF,T1IF,T2IF and T3IF, which are set by a match in their respective timer/counter register. The Carrier Generator Interrupt is occurred by CGIF which will be explained in "20. REMOCON CARRIER GENERATOR" on page 83. The AD converter Interrupt is generated by ADCIF which is set by finishing the analog to digital conversion.

The Basic Interval Timer Interrupt is generated by BITIF which is set by overflow of the Basic Interval Timer Register (BITR). The Watch dog Interrupt is generated by WDTIF which set by a match in Watch dog timer register (when the bit WDTON is set to "0"). The Watch Timer Interrupt is generated by WTIF which is set periodically according to the established time interval.

When an interrupt is generated, the bit of interrupt request flag register(IRQH, IRQM, IRQL) that generated it is cleared by the hardware when the service routine is vectored to only if the interrupt was transition-activated.

Each bit of Interrupt flag register(INTFH, INTFL) is set when corresponding interrupt flag bit as well as interrupt enable bit are set. The bits of interrupt flag register are never cleared by the hardware although the service routine is vectored to. Therefore, the interrupt flag register can be used to distinguish a right interrupt source from two available ones in a vector address. For example, RX1 and TX1 which have the same vector address($FFF0_H$) may be distinguished by INTFH register.

Interrupt enable registers are shown in . These registers are composed of interrupt enable bits of each interrupt source, these bits determine whether an interrupt will be accepted or not. When enable bit is "0", a corresponding interrupt source is prohibited. Note that PSW contains also a master enable bit, I-flag, which disables all interrupts at once. When an interrupt is occurred, the I-flag is cleared and disable any further interrupt, the return address and PSW are pushed into the stack and the PC is vectored to.

In an interrupt service routine, any other interrupt may be serviced. The source(s) of these interrupts can be determined by polling the interrupt request flag bits. Then, the interrupt request flag bit(s) must be cleared by software before re-enabling interrupts to avoid recursive interrupts. The Interrupt Request flags are able to be read and written.

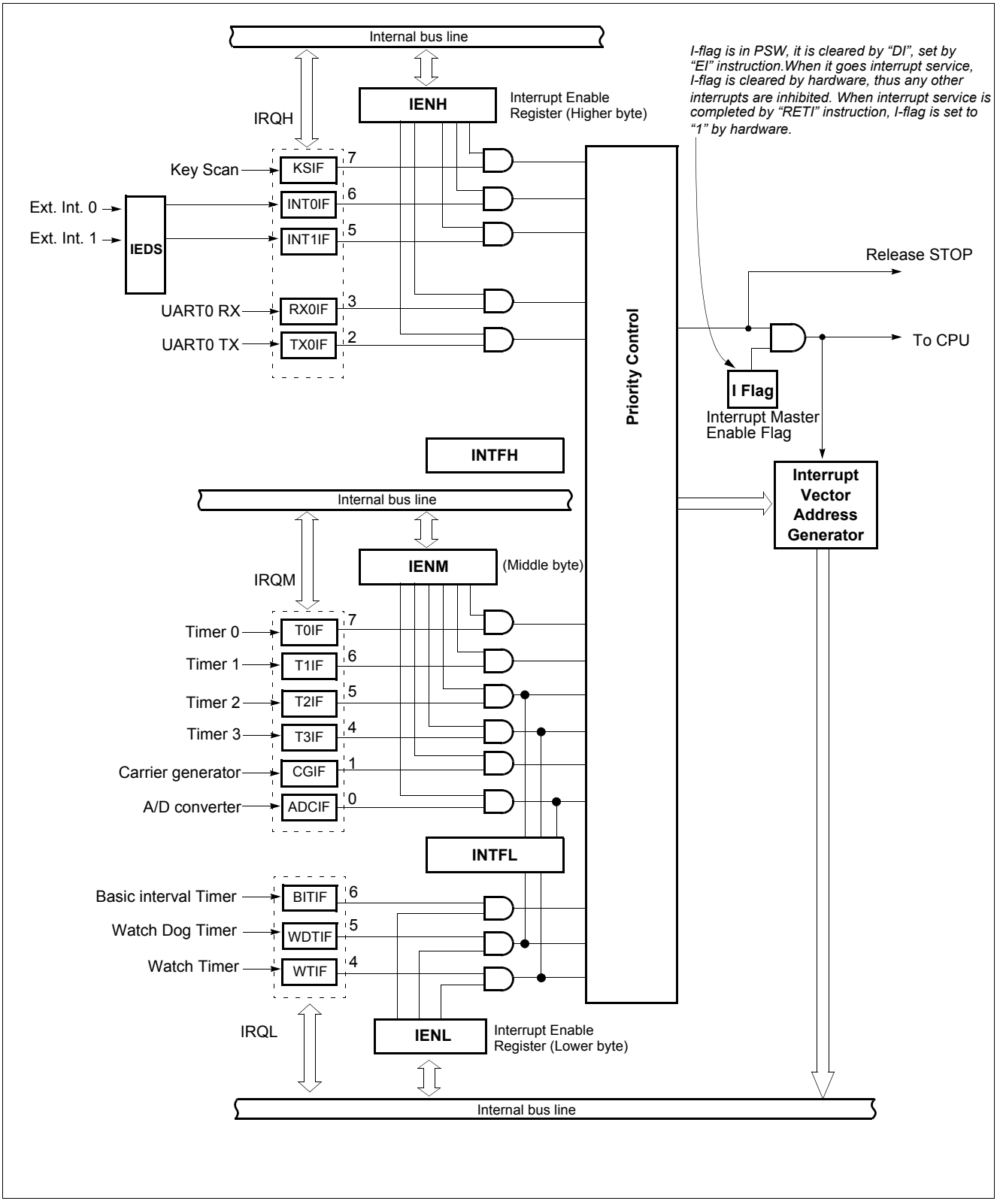


Figure 17-1 Block Diagram of Interrupt

IENH (Interrupt Enable High Register)

Bit :	R/W 7	R/W 6	R/W 5	4	R/W 3	R/W 2	1	0
	KSE	INT0E	INT1E	-	RX0E	TX0E	-	-

ADDRESS : $0F6_H$
RESET VALUE : 000-00--$_B$

ONLY FLASH MCU IS AVAILABLE

IENM (Interrupt Enable Middle Register)

Bit :	R/W 7	R/W 6	R/W 5	R/W 4	3	2	R/W 1	R/W 0
	T0E	T1E	T2E	T3E	-	-	CGE	ADC

ADDRESS : $0F7_H$
RESET VALUE : 0000--00_B

IENL (Interrupt Enable Low Register)

Bit :	7	R/W 6	R/W 5	R/W 4	3	2	1	0
	-	BIT	WDT	WT	-	-	-	-

ADDRESS : $0F8_H$
RESET VALUE : -000----$_B$

Enables or disables the interrupt individually

0 : Disable
1 : Enable

IRQH (Interrupt Request Flag High Register)

Bit :	R/W 7	R/W 6	R/W 5	4	R/W 3	R/W 2	1	0
	KSIF	INT0IF	INT1IF	-	RX0IF	TX0IF	-	-

ADDRESS : $0F9_H$
RESET VALUE : 000-00--$_B$

ONLY FLASH MCU IS AVAILABLE

IRQM (Interrupt Request Flag Middle Register)

Bit :	R/W 7	R/W 6	R/W 5	R/W 4	3	2	R/W 1	R/W 0
	T0IF	T1IF	T2IF	T3IF	-	-	CGIF	ADCIF

ADDRESS : $0FA_H$
RESET VALUE : 0000--00_B

IRQL (Interrupt Request Flag Low Register)

Bit :	7	R/W 6	R/W 5	R/W 4	3	2	1	0
	-	BITIF	WDTIF	WTIF	-	-	-	-

ADDRESS : $0FB_H$
RESET VALUE : -000----$_B$

0 : Interrupt not occurred
1 : Interrupt Request

INTFL (Interrupt Flag Register Low)

Bit :	R/W 7	R/W 6	5	4	3	R/W 2	R/W 1	R/W 0
	T2F	T3F	-	-	-	ADCF	WTF	WDTF

ADDRESS : $0F5_H$
RESET VALUE : 00---000_B

T2F(T2 Interrupt Flag)
0 : No Generation
1 : Generation

T3F(T3 Interrupt Flag)
0 : No Generation
1 : Generation

ADCF(ADC Interrupt Flag)
0 : No Generation
1 : Generation

WTF(WT Interrupt Flag)
0 : No Generation
1 : Generation

WDTF(WDT Interrupt Flag)
0 : No Generation
1 : Generation

Figure 17-2 Interrupt Enable Registers and Interrupt Request Registers

17.1 Interrupt Sequence

An interrupt request is held until the interrupt is accepted or the interrupt latch is cleared to "0" by a reset or an instruction. Interrupt acceptance sequence requires 8 f_{OSC} (2 μs at f_{MAIN}=4MHz) after the completion of the current instruction execution. The interrupt service task is terminated upon execution of an interrupt return instruction [RETI].

Interrupt acceptance

1. The interrupt master enable flag (I-flag) is cleared to "0" to temporarily disable the acceptance of any following maskable interrupts. When a non-maskable interrupt is accepted, the acceptance of any following interrupts is temporarily disabled.
2. Interrupt request flag for the interrupt source accepted is cleared to "0".
3. The contents of the program counter (return address) and the program status word are saved (pushed) onto the stack area. The stack pointer decreases 3 times.
4. The entry address of the interrupt service program is read from the vector table address and the entry address is loaded to the program counter.
5. The instruction stored at the entry address of the interrupt service program is executed.

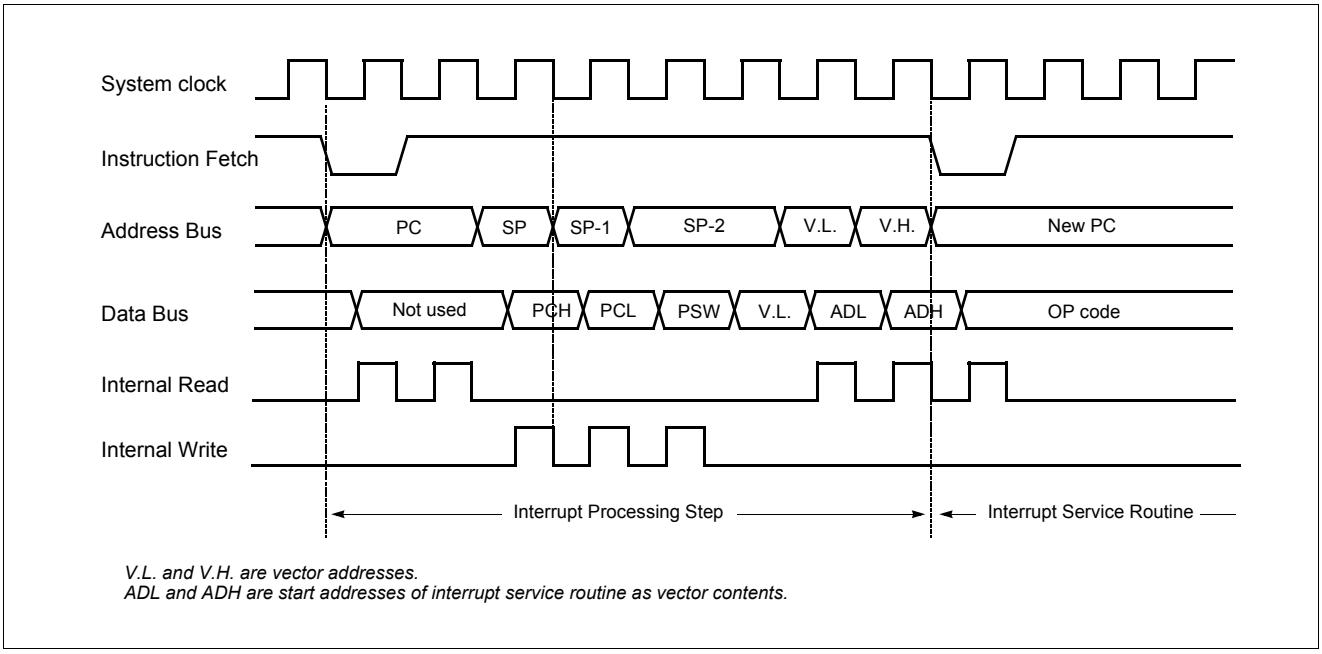
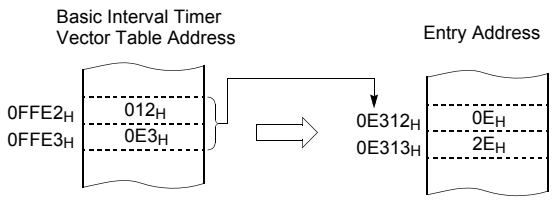


Figure 17-3 Timing chart of Interrupt Acceptance and Interrupt Return Instruction



Correspondence between vector table address for BIT interrupt and the entry address of the interrupt service program.

An interrupt request is not accepted until the I-flag is set to "1" even if a requested interrupt has higher priority than that of the current interrupt being serviced.

When nested interrupt service is required, the I-flag should be set to "1" by "EI" instruction in the interrupt service program. In this case, acceptable interrupt sources are selectively enabled by the individual interrupt enable flags.

Saving/Restoring General-purpose Register

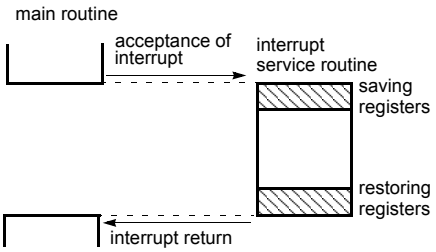
During interrupt acceptance processing, the program counter and the program status word are automatically saved on the stack, but accumulator and other registers are not saved itself. If necessary, these registers should be saved by the software. Also, when multiple interrupt services are nested, it is necessary to avoid using the same data memory area for saving registers.

The following method is used to save/restore the general-purpose registers.

Example: Register saving

```
INTxx:  PUSH    A       ;SAVE ACC.
        PUSH    X       ;SAVE X REG.
        PUSH    Y       ;SAVE Y REG.
        interrupt processing
        POP     Y       ;RESTORE Y REG.
        POP     X       ;RESTORE X REG.
        POP     A       ;RESTORE ACC.
        RETI            ;RETURN
```

General-purpose registers are saved or restored by using push and pop instructions.



17.2 BRK Interrupt

Software interrupt can be invoked by BRK instruction, which has the lowest priority order.

Interrupt vector address of BRK is shared with the vector of TCALL 0 (Refer to Program Memory Section). When BRK interrupt is generated, B-flag of PSW is set to distinguish BRK from TCALL 0.

Each processing step is determined by B-flag as shown in .

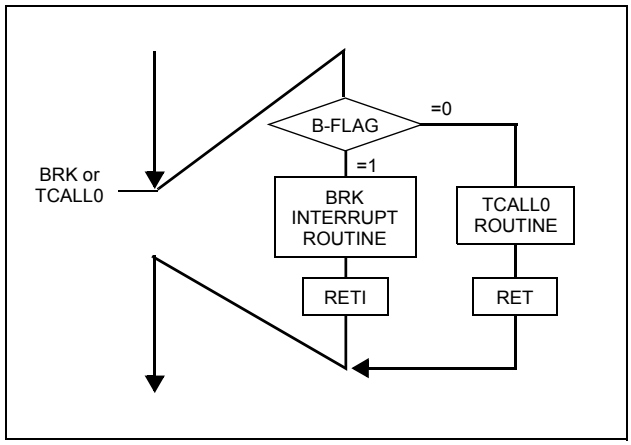


Figure 17-4 Execution of BRK/TCALL0

17.3 Multi Interrupts

If two requests of different priority levels are received simultaneously, the request of higher priority level is serviced. If requests of the interrupt are received at the same time simultaneously, an internal polling sequence determines by hardware which request is serviced.

However, multiple processing through software for special features is possible. Generally when an interrupt is accepted, the I-flag is cleared to disable any further interrupt. But as user sets I-flag in interrupt routine, some further interrupt can be serviced even if certain interrupt is in progress.

Example: Even though Timer1 interrupt is in progress, INT0 interrupt serviced without any suspend.

```
TIMER1: PUSH   A
        PUSH   X
        PUSH   Y
```

```
LDM     IENH,#40H     ;Enable INT0 only
LDM     IENM,#0       ;Disable other
LDM     IENL,#0       ;Disable other
EI                    ;Enable Interrupt
:
:
:

:
:
:
LDM     IENH,#0FFH    ;Enable all interrupts
LDM     IENM,#0FFH
LDM     IENL,#0F0H
POP     Y
POP     X
POP     A
RETI
```

.

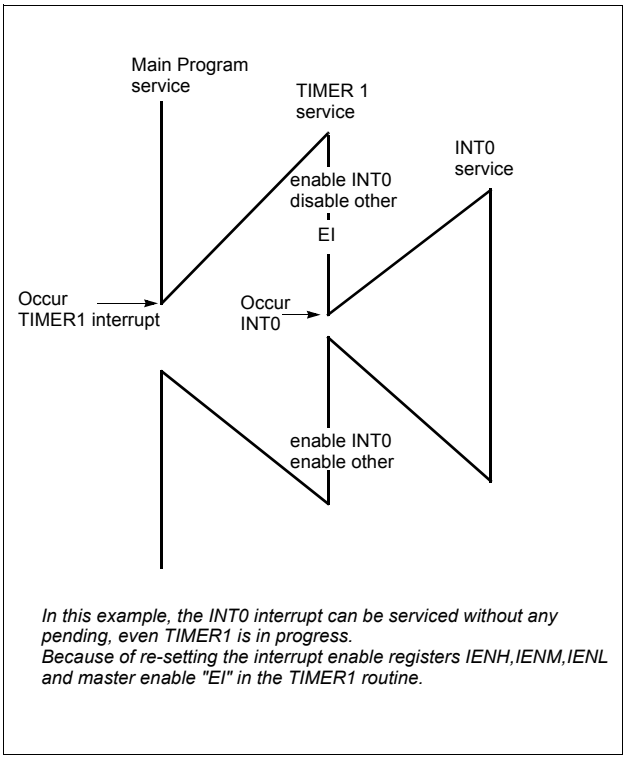


Figure 17-5 Execution of Multi Interrupt

17.4 External Interrupt

The external interrupt on INT0, INT1 and INT2 pins are edge triggered depending on the edge selection register IEDS (address $0FC_H$) as shown in .

The edge detection of external interrupt has three transition activated mode: rising edge, falling edge, and both edge.

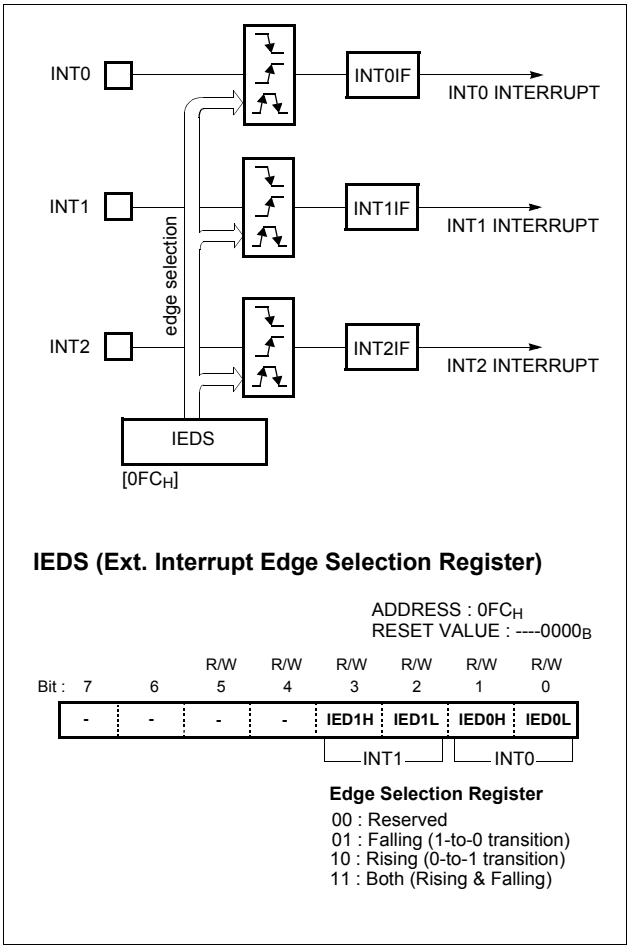


Figure 17-6 External Interrupt Block Diagram

Example: To use as an INT0

```
        :
        :
;**** Set port as an input port R0
        LDM     R0IO,#1011_1111B
        ;
;**** Set port as an interrupt port
        LDM     R0FUNC,#0100_0000B
        ;
;**** Set Falling-edge Detection
        LDM     IEDS,#0000_0001B
        :
        :
        :
```

Response Time

The INT0, INT1 and INT2 edge are latched into INT0F, INT1F and INT2F at every machine cycle. The values are not actually polled by the circuitry until the next machine cycle. If a request is active and conditions are right for it to be acknowledged, a hardware subroutine call to the requested service routine will be the next instruction to be executed. The DIV itself takes twelve cycles. Thus, a maximum of twelve complete machine cycles elapse between activation of an external interrupt request and the beginning of execution of the first instruction of the service routine.

Interrupt response timings are shown in .

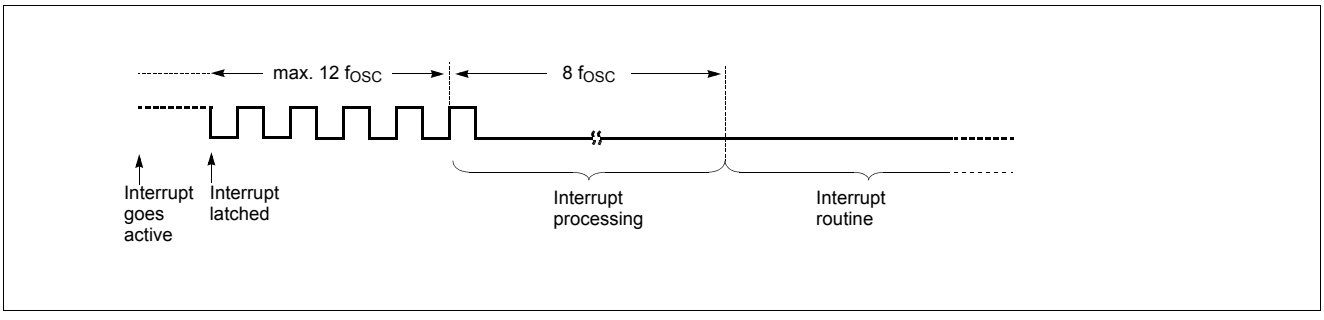


Figure 17-7 Interrupt Response Timing Diagram

18. KEY SCAN

The key-scan block consists of key scan mode register (KSMR) and R3 pull-up register (R3PU). When the key scan interrupt is used, key scan mode register KSMR (address EB_H) should be set properly as shown in . The pins which is to be used as key scan input should be set by KSMR and the strobe output pins should be set as open drain. The strobe output pins could be selected from among R0[7:0], R1[3:0], R2[7:0] and R3[2:7].

If the "L" signal is input to any one or more of key scan input pins, the KSIF request flag is set to "1". This generates an interrupt request. It also can be used in the way of release from STOP mode.

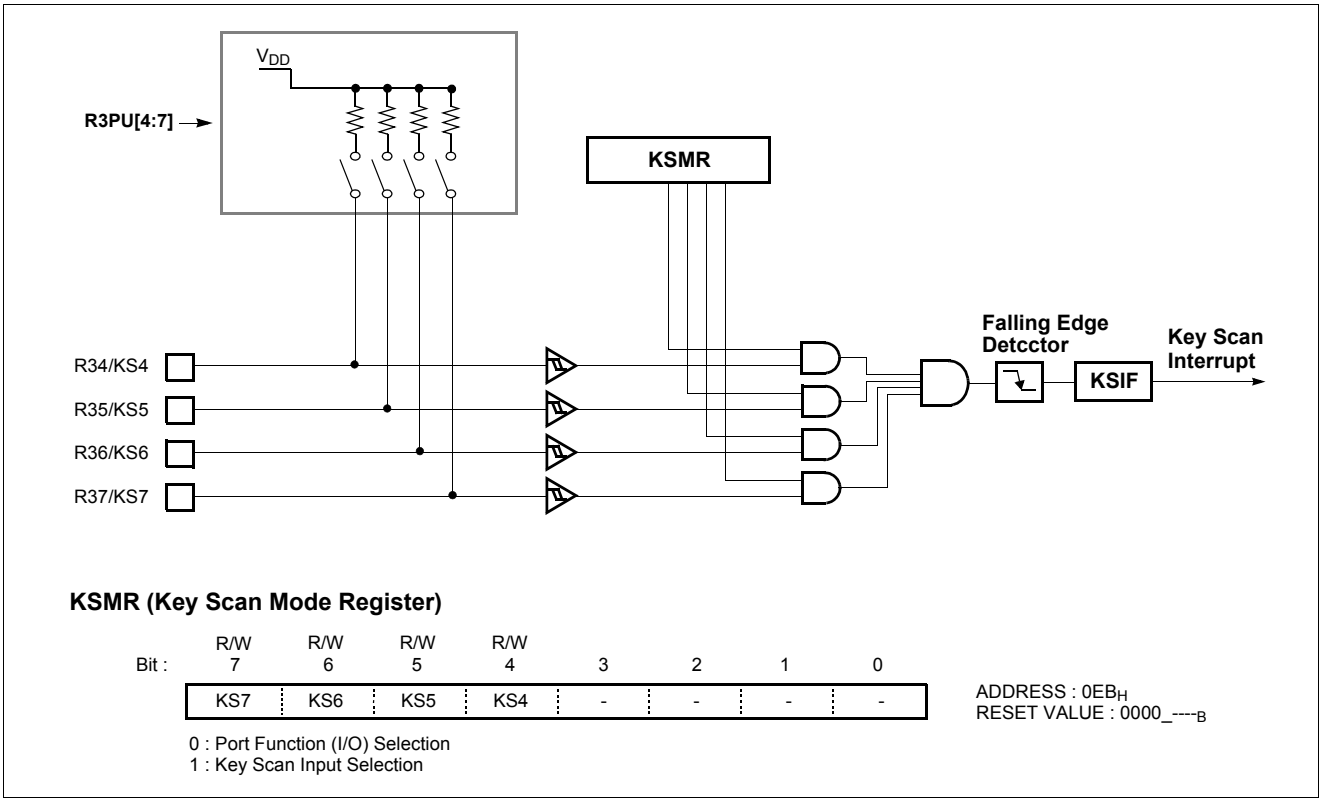


Figure 18-1 Key Scan Interrupt Block Diagram

Usage of Key Scan

When key board scanning, it is recommended that set the output strobe to "L" first and then read R3 port after 60us delay time. Because the rising time of the output strobe port from "L" to "H" is so long. The explain this reason.

```
;Program Example,

LDM     R1,#0000_1110b ;R1<0> Port set to low(Strobe output)
CALL    Delay_60us     ;60us time delay routine
LDA     R3             ;read R3 port
```

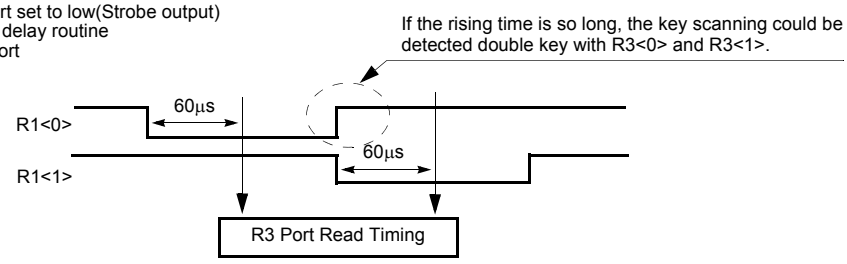


Figure 18-2 Key Scan Timing

19. LCD DRIVER

The MC80X7208 has the circuit that directly drives the liquid crystal display (LCD) and its control circuit. The Segment/Common Driver directly drives the LCD panel, and the LCD Controller generates the segment/common signals according to the RAM which stores display data. In addition, VCL2 ~ VCL0 pin are provided as the drive power pins.

The MC80X7208 has the following pins connected with LCD.

1. Segment output port 27 pins (SEG0-23, SEG32~34)

2.Common output port 4 pins (COM0-COM3)

19.1 Configuration of LCD driver

shows the configuration of the LCD driver.

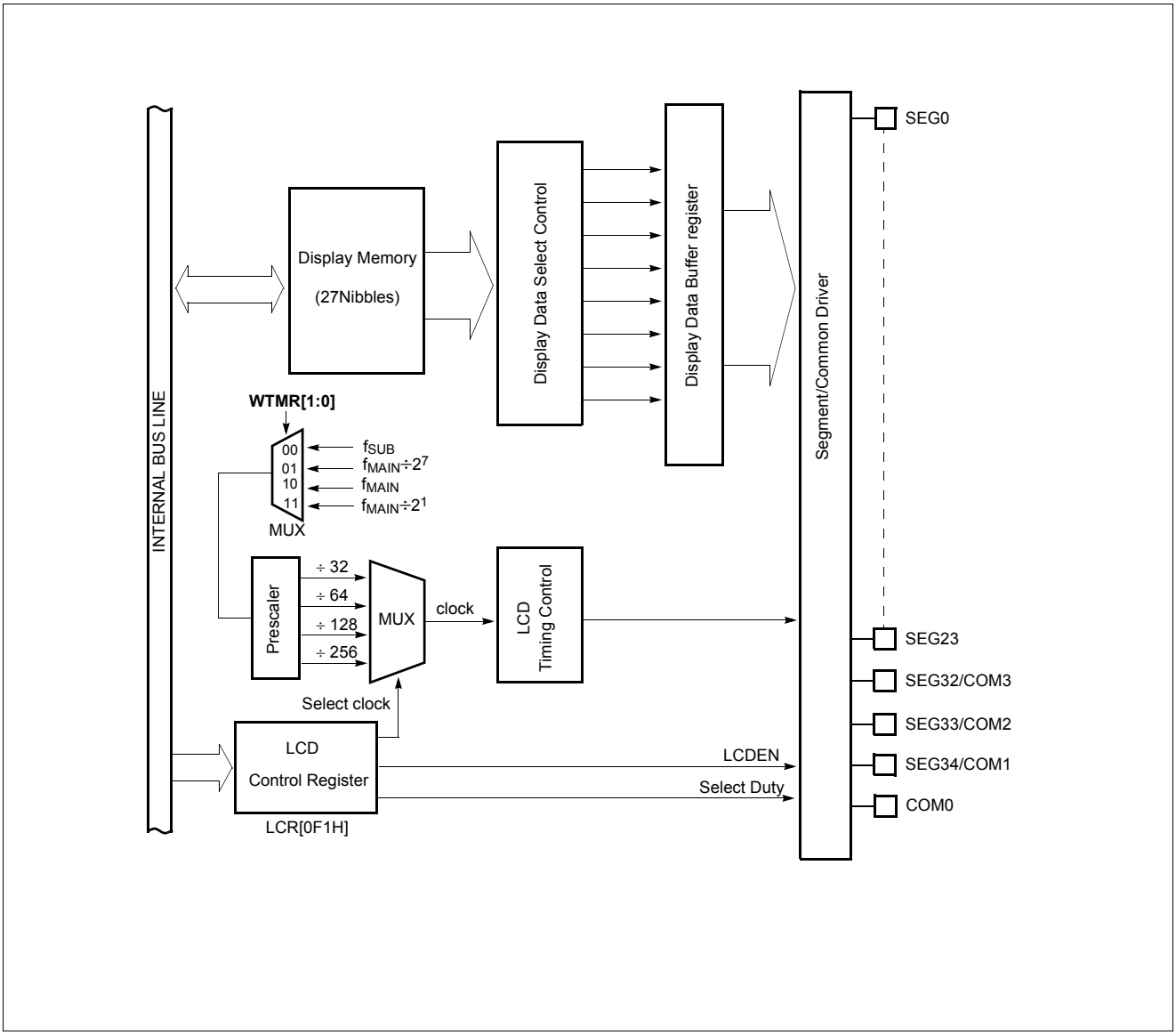


Figure 19-1 LCD Driver Block Diagram

19.2 Control of LCD Driver Circuit

The LCD driver is controlled by the LCD Control Register (LCR). The LCR[1:0] determines the frequency of COM signal scanning of each segment output. RESET clears the LCD control register LCR values to logic zero. The LCD display can continue to operate during SLEEP and STOP modes if a sub-frequency clock is used as system clock source. The constant voltage booster circuit for using LCD driver is built in, so the definite voltage could supplied regardless of power source voltage fluctuations.

Note: *The Sub clock is used as voltage booster source clock, so the stabilization time is need to use voltage booster. Normally, the stabilization time is need more than 500ms. The external bias registers cannot be used for LCD display supply voltage.*

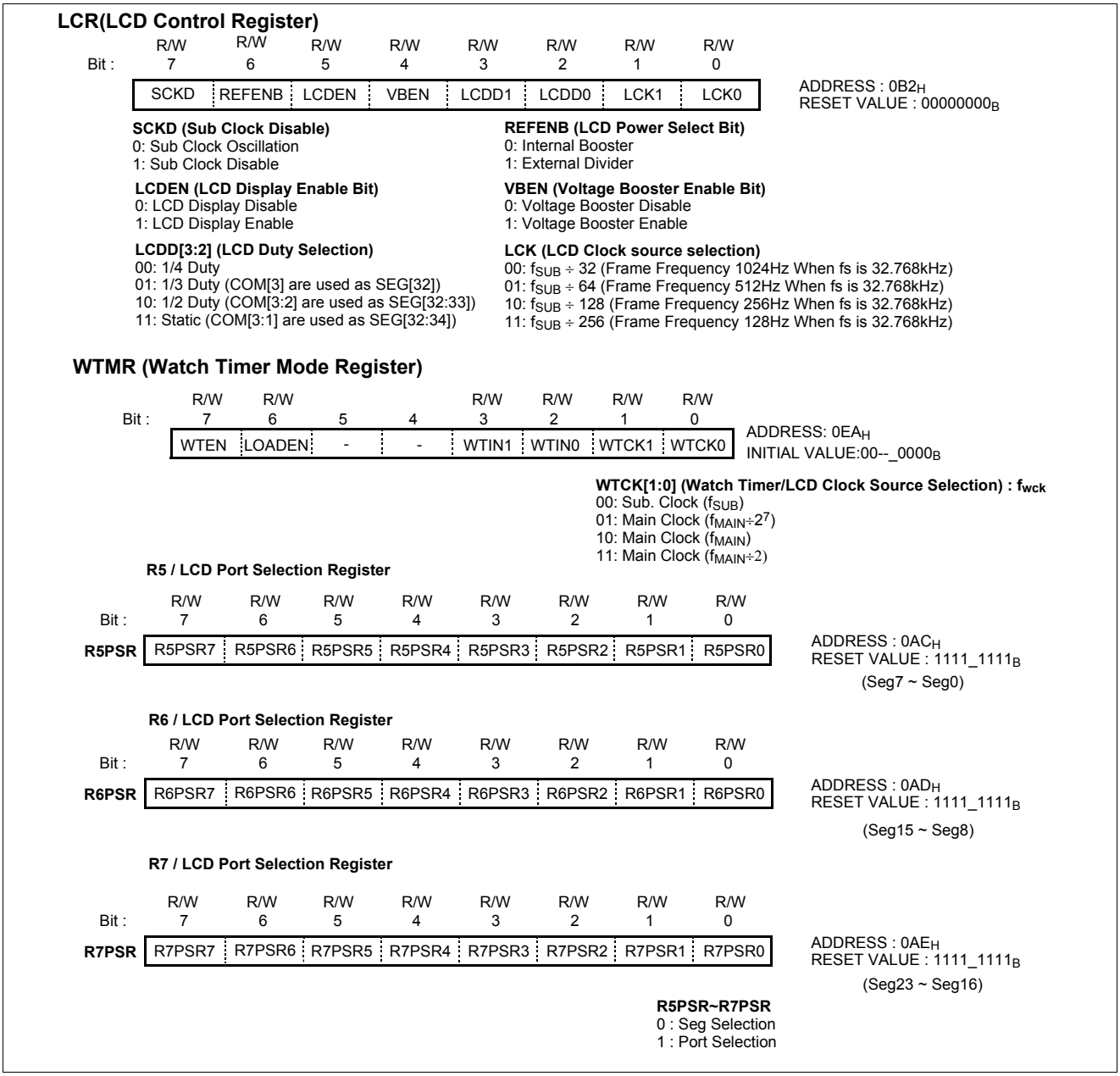


Figure 19-2 LCD Control Register

Selecting Frame Frequency

Frame frequency is set to the base frequency as shown in the following Table 19-1. The f_S is selected to f_{SUB} (sub clock) which is 32.768kHz.

LCR[1:0]	LCD clock	Frame Frequency (Hz)			
		Duty = Static	Duty = 1/2	Duty = 1/3	Duty = 1/4
00	$f_{SUB} \div 32$	1024	512	341.3	256
01	$f_{SUB} \div 64$	512	256	170.7	128
10	$f_{SUB} \div 128$	256	128	85.3	64
11	$f_{SUB} \div 256$	128	64	42.7	32

Table 19-1 Setting of LCD Frame Frequency

The matters to be attended to use LCD driver

In reset state, LCD source clock is sub clock. So, when the power is supplied, the LCD display would be flickered before the oscillation of sub clock is stabilized. It is recommended to use LCD display on after the stabilization time of sub clock is considered enough. If the LCD is reset during display, the display would be blotted by the capacity of LCD power circuit. The external circuit of constant voltage booster for using LCD driver is shown at right.

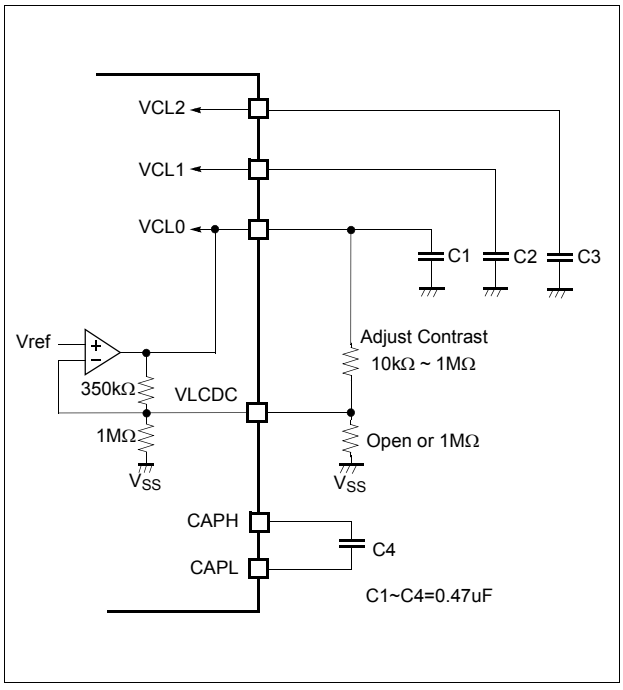


Figure 19-3 Internal Booster Mode

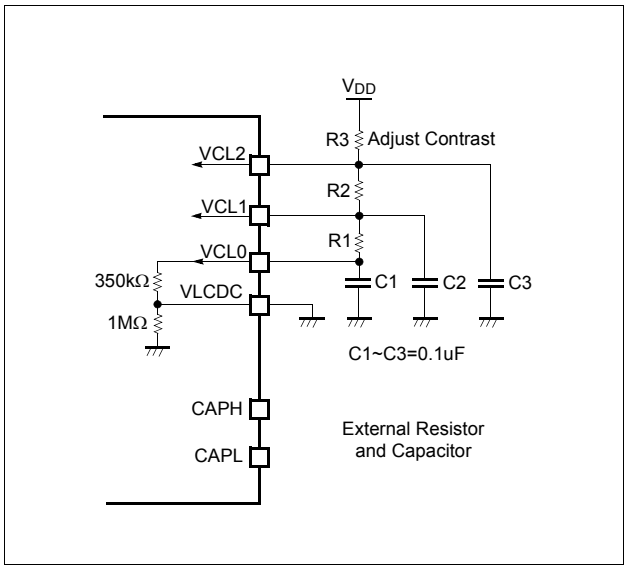


Figure 19-4 External Divide Mode

19.3 LCD Display Memory

Display data are stored to the display data area (page 4) in the data memory.

The display datas which stored to the display data area (address 0460_H-0482_H) are read automatically and sent to the LCD driver by the hardware. The LCD driver generates the segment signals and common signals in accordance with the display data and drive method. Therefore, display patterns can be changed by only overwriting the contents of the display data area with a program. The table look up instruction is mainly used for this overwriting.

 shows the correspondence between the display data area and the SEG/COM pins. The LCD lights when the display data is “1” and turn off when “0”.

The SEG data for display is controlled by RPR (RAM Paging Register).

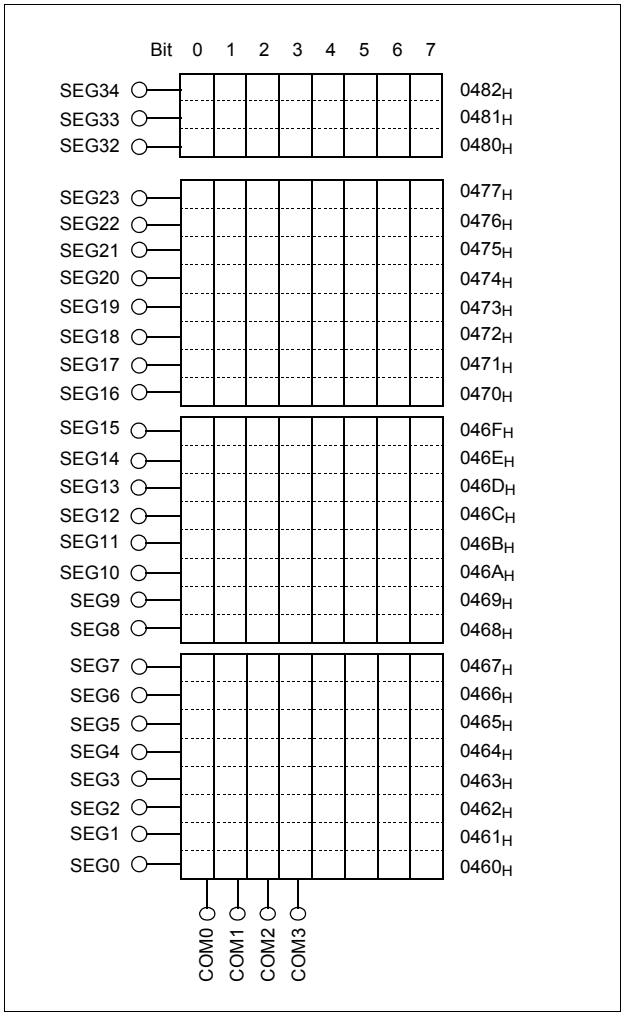


Figure 19-5 LCD Display Memory

19.4 Control Method of LCD Driver

Initial Setting

Flow chart of initial setting is shown in .

Example: Driving of LCD

```
Select Frame Frequency   LDM   LCR,#12H    ;fF=64Hz, 1/4 duty(fSUB= 32.768kHz)
                         :
                         LDM   RPR,#4;Select LCD Memory(1 page)
                         SETG
Clear                    LDX   #60H
LCD Display      C_LCD1: LDA   #0 ;RAM Clear
Memory                           ;(0460H->0482H)
                         STA   {X}+
                         CMPX  #083H
                         BNE   C_LCD1
                         CLRG
                         :
Turn on LCD              SET1  LCR.5;Enable display
                         :
```

.

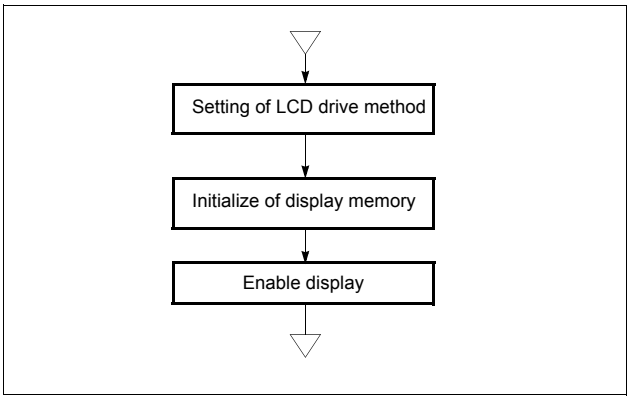


Figure 19-6 Initial Setting of LCD Driver

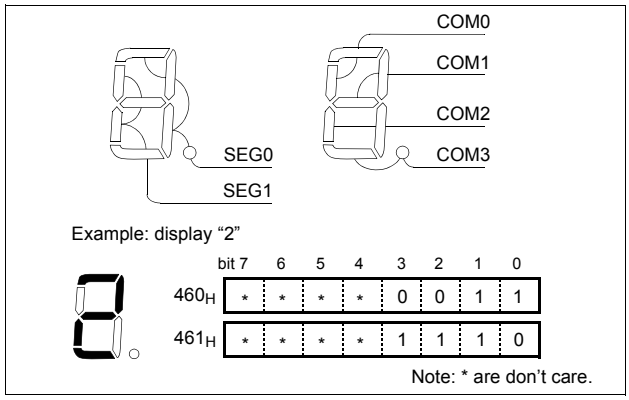


Figure 19-7 Example of Connection COM & SEG

Display Data

Normally, display data are kept permanently in the program memory and then stored at the display data area by the table look-up instruction. This can be explained using character display with 1/4 duty LCD as an example as well as any LCD panel. The COM and SEG connections to the LCD and display data are the same as those shown is . Following is showing the Programming example for displaying character.

Note: *When power on RESET, sub oscillation start up time is required. Enable LCD display after sub oscillation is stabilized, or LCD may occur flicker at power on time shortly.*

Write into the LCD Memory

```
        :
        CLRG
        LDX#<DISPRAM;Address included the data
              ;to be displayed.
GOLCD:  LDA{X}
        TAY
        LDA!FONT+Y ;LOAD FONT DATA
        LDMRPR,#4;Set RPR = 4 to access LCD
        SETG  ;Set Page 4
        LDX#60H
        STA{X}+;LOWER 4 BITS OF ACC. seg0
        XCN
        STA{X};UPPER 4 BITS OF ACC. seg1
        CLRG  ;Set Page = 0
        :
```

Font data

```
FONT    DB 1101_0111B;   "0"
        DB 0000_0110B;   "1"
        DB 1110_0011B;   "2"
        DB 1010_0111B;   "3"
        DB 0011_0110B;   "4"
        DB 1011_0101B;   "5"
        DB 1111_0101B;   "6"
        DB 0000_0111B;   "7"
        DB 1111_0111B;   "8"
        DB 0011_0111B;   "9"
```

LCD Waveform

The LCD duty can be selected by LCR register. The kinds of LCD waveforms are four totally. Among them, 1/4 duty waveforms are shown .

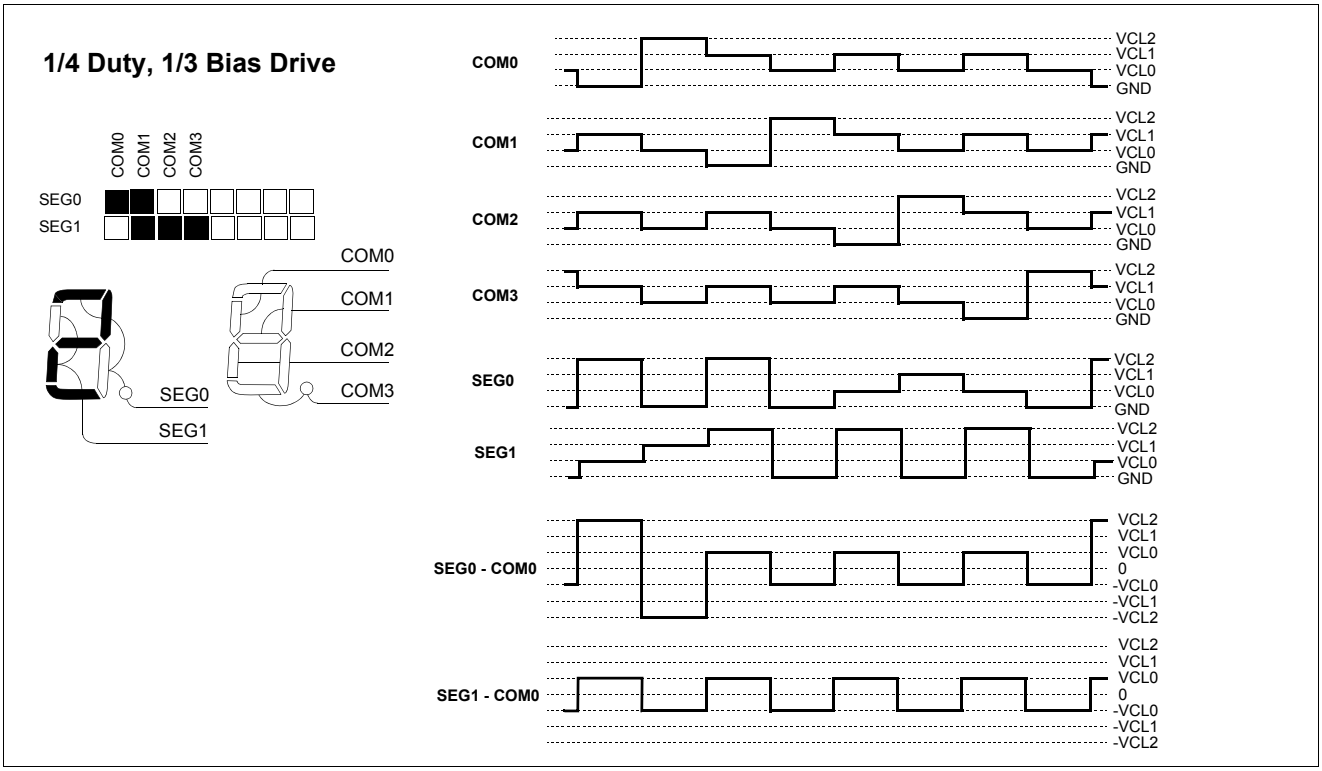


Figure 19-8 Example of LCD drive output

19.5 Duty and Bias Selection of LCD Driver

4 kinds of driving methods can be selected by LCDD[1:0] (bits 3 and 2 of LCD Control Register and connection of VCL pin externally. shows typical driving waveforms for LCD.).

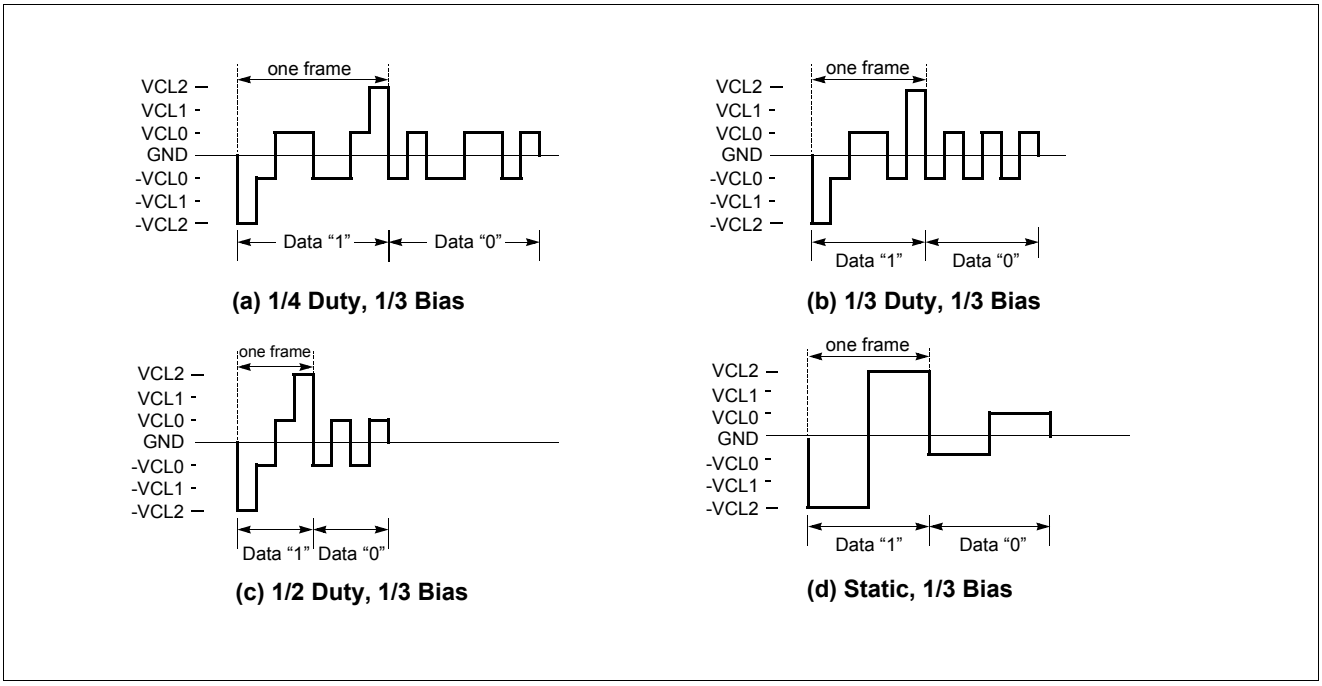


Figure 19-9 LCD Drive Waveform (Voltage COM-SEG Pins)

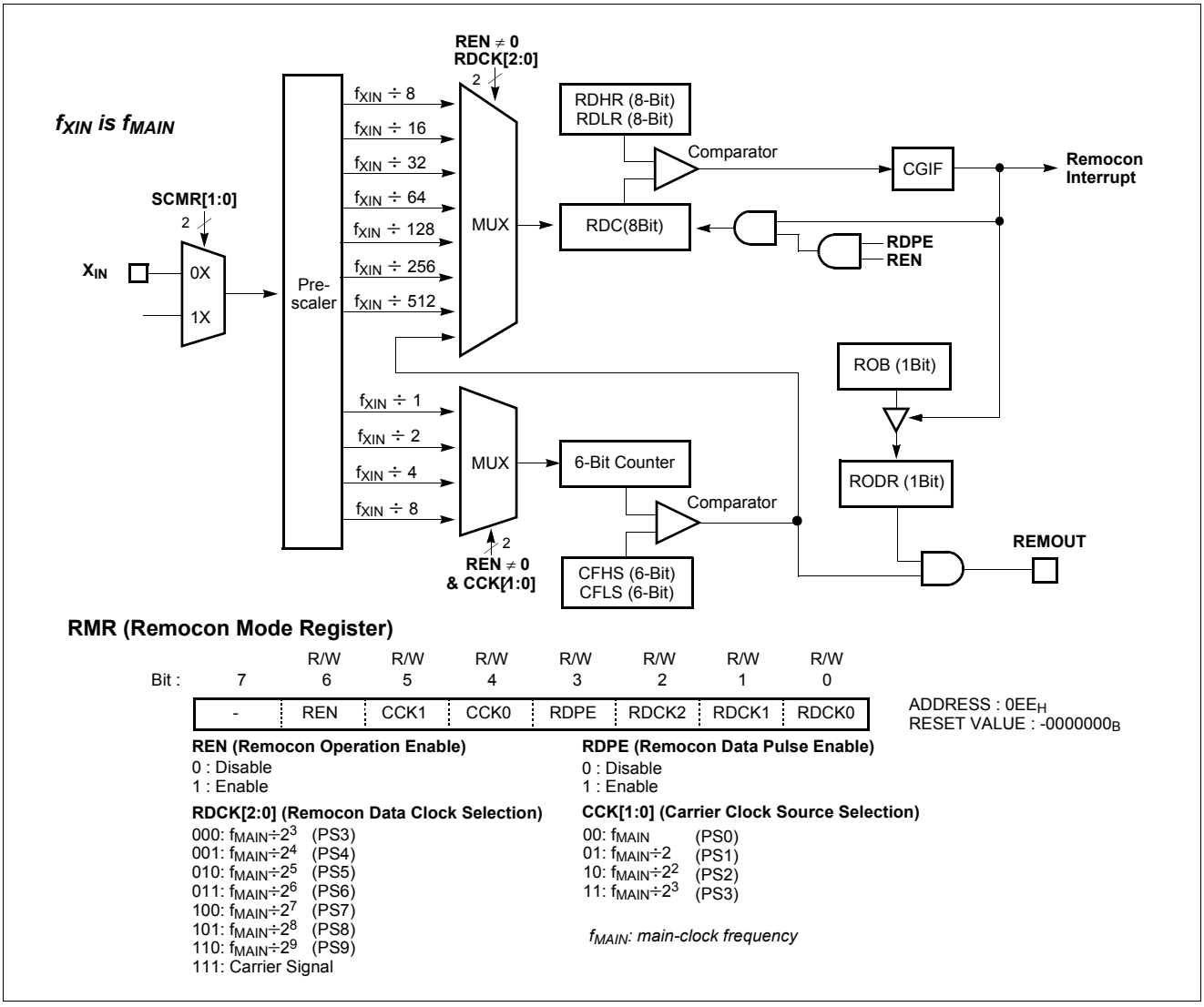
20. REMOCON CARRIER GENERATOR

The MC80X7208 has a circuit to generate carriers for the remote controller. This circuit consists of Remocon Mode Register (RMR), Carrier Frequency High Selection (CFHS), Carrier Frequency Low Selection (CFLS), Remocon Data High Register (RDHR), Remocon Data Low Register (RDLR), Remocon Data Counter (RDC), Remocon Output Data Register (RODR) and Remocon Output Buffer (ROB) as shown in . A carrier duty and frequency are determined by the contents of these registers. A source clock input to the 6-bit counter is selected by diving the frequency of the system clock by two (main or sub clock).

20.1 Remocon Signal Output Control

The output of the REMOUT pin which outputs carriers is controlled by RODR and ROB register. While the Bit-0 of RODR is “1”, the REMOUT pin outputs a carrier signal generated by the remote controller carrier generator. While this Bit is “0”, the output of the REMOUT pin is low.

The content of the ROB is automatically transferred to the RODR by an interrupt signal generated by the 8-Bit timer. The content of the RODR.0 is output to the REMOUT pin. Namely, the REMOUT pin outputs a high-level signal when the bit0 of RODR is “1” and a low-level signal when the bit0 of RODR is “0”.



RMR (Remocon Mode Register)

Bit :	7	6	5	4	3	2	1	0
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	-	REN	CCK1	CCK0	RDPE	RDCK2	RDCK1	RDCK0

ADDRESS : $0EE_H$
RESET VALUE : -0000000_B

REN (Remocon Operation Enable)
0 : Disable
1 : Enable

RDPE (Remocon Data Pulse Enable)
0 : Disable
1 : Enable

RDCK[2:0] (Remocon Data Clock Selection)
000: $f_{MAIN}÷2^3$ (PS3)
001: $f_{MAIN}÷2^4$ (PS4)
010: $f_{MAIN}÷2^5$ (PS5)
011: $f_{MAIN}÷2^6$ (PS6)
100: $f_{MAIN}÷2^7$ (PS7)
101: $f_{MAIN}÷2^8$ (PS8)
110: $f_{MAIN}÷2^9$ (PS9)
111: Carrier Signal

CCK[1:0] (Carrier Clock Source Selection)
00: f_{MAIN} (PS0)
01: $f_{MAIN}÷2$ (PS1)
10: $f_{MAIN}÷2^2$ (PS2)
11: $f_{MAIN}÷2^3$ (PS3)

f_{MAIN}: *main-clock frequency*

Figure 20-1 Remocon Carrier Generator Block Diagram

CFHS (Carrier Frequency High Selection)

Bit :	7	6	5	4	3	2	1	0
	W	W	W	W	W	W	W	W
	-	-	CFH5	CFH4	CFH3	CFH2	CFH1	CFH0

ADDRESS : $0EC_H$
RESET VALUE : $--111111_B$

Carrier High Interval = The Value of CFHS x Clock Source Period

CFLS (Carrier Frequency Low Selection)

Bit :	7	6	5	4	3	2	1	0
	W	W	W	W	W	W	W	W
	-	-	CFL5	CFL4	CFL3	CFL2	CFL1	CFL0

ADDRESS : $0ED_H$
RESET VALUE : $--111111_B$

Carrier Low Interval = The Value of CFLS x Clock Source Period

RDHR (Remocon Data High Register)

Bit :	7	6	5	4	3	2	1	0
	W	W	W	W	W	W	W	W
	RDH7	RDH6	RDH5	RDH4	RDH3	RDH2	RDH1	RDH0

ADDRESS : $0EF_H$
RESET VALUE : 11111111_B

Remocon Data High Interval = The Value of RDHR x Clock Source Period

RDLR (Remocon Data Low Register)

Bit :	7	6	5	4	3	2	1	0
	W	W	W	W	W	W	W	W
	RDL7	RDL6	RDL5	RDL4	RDL3	RDL2	RDL1	RDL0

ADDRESS : $00F0_H$
RESET VALUE : 11111111_B

Remocon Data Low Interval = The Value of RDLR x Clock Source Period

RDC (Remocon Data Counter)

Bit :	7	6	5	4	3	2	1	0
	R	R	R	R	R	R	R	R
	RDC7	RDC6	RDC5	RDC4	RDC3	RDC2	RDC1	RDC0

ADDRESS : $00F1_H$
RESET VALUE : 00000000_B

Remocon Data Counter Value

RODR (Remocon Output Data Register)

Bit :	7	6	5	4	3	2	1	0
								R/W
	-	-	-	-	-	-	-	ROD0

ADDRESS : $0F2_H$
RESET VALUE : $-------0_B$

Remocon Data Output Value

ROB (Remocon Output Buffer)

Bit :	7	6	5	4	3	2	1	0
								R/W
	-	-	-	-	-	-	-	ROB0

ADDRESS : $0F3_H$
RESET VALUE : $-------0_B$

Remocon Data Output Buffer

Figure 20-2 Remocon Registers

20.2 Carrier Frequency

The carrier frequency and the pulse of data are calculated by below formula. The lengths of carrier frequency and pulse of data are shown in .

t_H = source clock(RMR[5:4]+1) × CFHS
t_L = source clock(RMR[5:4]+1) × CFLS
f_C (Carrier Frequency) = $1/(t_H+t_L)$
t_{DH} = source clock(RMR[2:0]+1) × RDHR
t_{DL} = source clock(RMR[2:0]+1) × RDLR

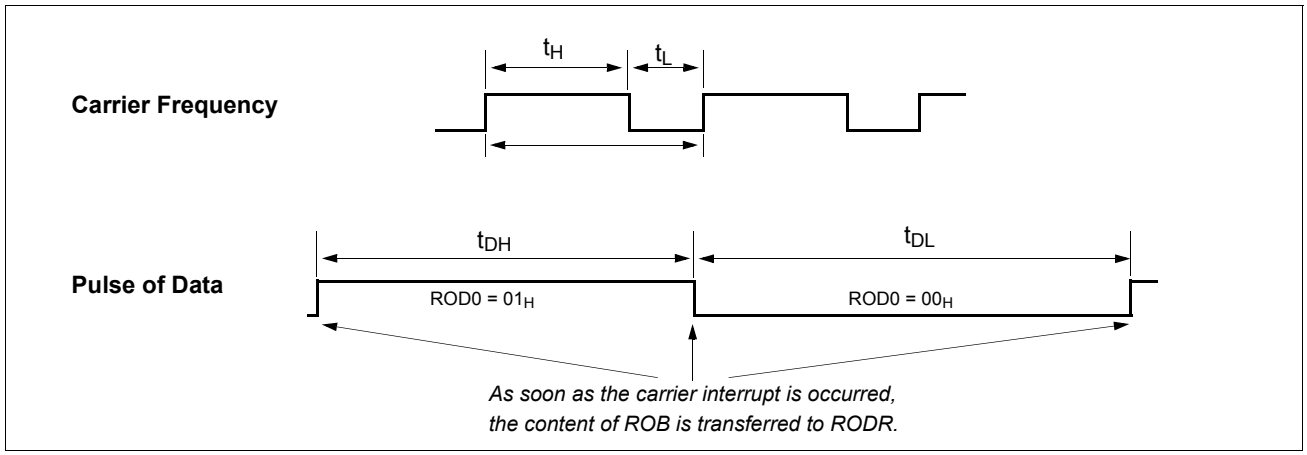


Figure 20-3 Carrier Frequency & Pulse of Data

The Table shows high and low length of carrier frequency according to CFLS and CFHS. This only shows when the source clock is selected f_{MAIN} and $f_{MAIN}÷2^2$ at 4MHz.

Set Value		Selection of PS0		Selection of PS2	
CFHS	CFLS	t_H(us)	t_L(us)	t_H(us)	t_L(us)
00_H	00_H	-	-	-	-
01_H	01_H	0.25	0.25	1.00	1.00
02_H	02_H	0.50	0.50	2.00	2.00
03_H	03H	0.75	0.75	3.00	3.00
04_H	04_H	1.00	1.00	4.00	4.00
05_H	05_H	1.25	1.25	5.00	5.00
06_H	06_H	1.50	1.50	6.00	6.00
07_H	07_H	1.75	1.75	7.00	7.00
08_H	08_H	2.00	2.00	8.00	8.00
09_H	09_H	2.25	2.25	9.00	9.00
$0A_H$	$0A_H$	2.50	2.50	10.00	10.00
$0B_H$	$0B_H$	2.75	2.75	11.00	11.00
$0C_H$	$0C_H$	3.00	3.00	12.00	12.00
$0D_H$	$0D_H$	3.25	3.25	13.00	13.00
$0E_H$	$0E_H$	3.50	3.50	14.00	14.00
$0F_H$	$0F_H$	3.75	3.75	15.00	15.00
10_H	10_H	4.00	4.00	16.00	16.00
11_H	11_H	4.25	4.25	17.00	17.00
12_H	12_H	4.50	4.50	18.00	18.00
13_H	13_H	4.75	4.75	19.00	19.00
14_H	14_H	5.00	5.00	20.00	20.00
15_H	15_H	5.25	5.25	21.00	21.00
16_H	16_H	5.50	5.50	22.00	22.00
17_H	17_H	5.75	5.75	23.00	23.00
18_H	18_H	6.00	6.00	24.00	24.00
19_H	19_H	6.25	6.25	25.00	25.00
$1A_H$	$1A_H$	6.50	6.50	26.00	26.00
$1B_H$	$1B_H$	6.75	6.75	27.00	27.00
$1C_H$	$1C_H$	7.00	7.00	28.00	28.00
$1D_H$	$1D_H$	7.25	7.25	29.00	29.00
$1E_H$	$1E_H$	7.50	7.50	30.00	30.00
$1F_H$	$1F_H$	7.75	7.75	31.00	31.00

Set Value		Selection of PS0		Selection of PS2	
CFHS	CFLS	t_H(us)	t_L(us)	t_H(us)	t_L(us)
20_H	20_H	8.00	8.00	32.00	32.00
21_H	21_H	8.25	8.25	33.00	33.00
22_H	22_H	8.50	8.50	34.00	34.00
23_H	23_H	8.75	8.75	35.00	35.00
24_H	24_H	9.00	9.00	36.00	36.00
25_H	25_H	9.25	9.25	37.00	37.00
26_H	26_H	9.50	9.50	38.00	38.00
27_H	27_H	9.75	9.75	39.00	39.00
28_H	28_H	10.00	10.00	40.00	40.00
29_H	29_H	10.25	10.25	41.00	41.00
$2A_H$	$2A_H$	10.50	10.50	42.00	42.00
$2B_H$	$2B_H$	10.75	10.75	43.00	43.00
$2C_H$	$2C_H$	11.00	11.00	44.00	44.00
$2D_H$	$2D_H$	11.25	11.25	45.00	45.00
$2E_H$	$2E_H$	11.50	11.50	46.00	46.00
$2F_H$	$2F_H$	11.75	11.75	47.00	47.00
30_H	30_H	12.00	12.00	48.00	48.00
31_H	31_H	12.25	12.25	49.00	49.00
32_H	32_H	12.50	12.50	50.00	50.00
33_H	33_H	12.75	12.75	51.00	51.00
34_H	34_H	13.00	13.00	52.00	52.00
35_H	35_H	13.25	13.25	53.00	53.00
36_H	36_H	13.50	13.50	54.00	54.00
37_H	37_H	13.75	13.75	55.00	55.00
38_H	38_H	14.00	14.00	56.00	56.00
39_H	39_H	14.25	14.25	57.00	57.00
$3A_H$	$3A_H$	14.50	14.50	58.00	58.00
$3B_H$	$3B_H$	14.75	14.75	59.00	59.00
$3C_H$	$3C_H$	15.00	15.00	60.00	60.00
$3D_H$	$3D_H$	15.25	15.25	61.00	61.00
$3E_H$	$3E_H$	15.50	15.50	62.00	62.00
$3F_H$	$3F_H$	15.75	15.75	63.00	63.00

Table 20-1 Length of Carrier Frequency (at 4MHz)

Example:

```
        Carrier Frequency = 37.8kHz, high = 8.52ms, low = 4.24ms, @4MHz

Rem_sig:LDM    RMR,#0001_0010B    ;carrier clock(PS1), remocon data clock(PS5)
        LDM    CFHS,#18           ;carrier low(IR LED)=18*PS1(0.5us)=9us
        LDM    CFLS,#35           ;carrier high(IR LED)=35*PS1(0.5us)=17.5us
        CLR1   ROD0
        LDM    R_bit,#1111_1000B
        LDM    RDHR,#213          ;213*5*PS5(8us)=8.52ms
        LDM    RDLR,#177          ;177*3*PS5(8us)=4.248ms
        LDX    #9

        CALL   DATA
        SET1   RMR.6              ;Remocon operation enable
        SET1   RMR.3              ;Remocon data pulse enable
        SET1   IENL.6             ;Remocon int.

Loop1:  NOP
        CMPX   #0
        BNE    Loop1

Finish: CLR1   ROD0
        CLR1   ROB0
        RET

;********
Data:   ROL    R_bit
        BCS    Set_rob0
        CLR1   ROB0
        RET

Set_rob0:SET1ROB0
        RET

;*********************************************;
;         Remocon int service routine         ;
;*********************************************;
;
Remocon_INT:

        CALL   Data
        DEC    X
        RETI
```

21. UNIVERSAL ASYNCHRONOUS SERIAL INTERFACE (UART:ONLY FLASH MCU IS AVAILABLE)

Note: *This Function is in the MC80F7208(FLASH MCU).*

The Asynchronous serial interface(UART) enables full-duplex operation wherein one byte of data after the start bit is transmitted and received. The on-chip baud rate generator dedicated to UART enables communications using a wide range of selectable baud rates.

The UART driver consists of TXSR0, RXBR0, ASIMR0 and BRGCR0 register. Clock asynchronous serial I/O mode (UART) can be selected by ASIMR register. shows a block diagram of the serial interface (UART).

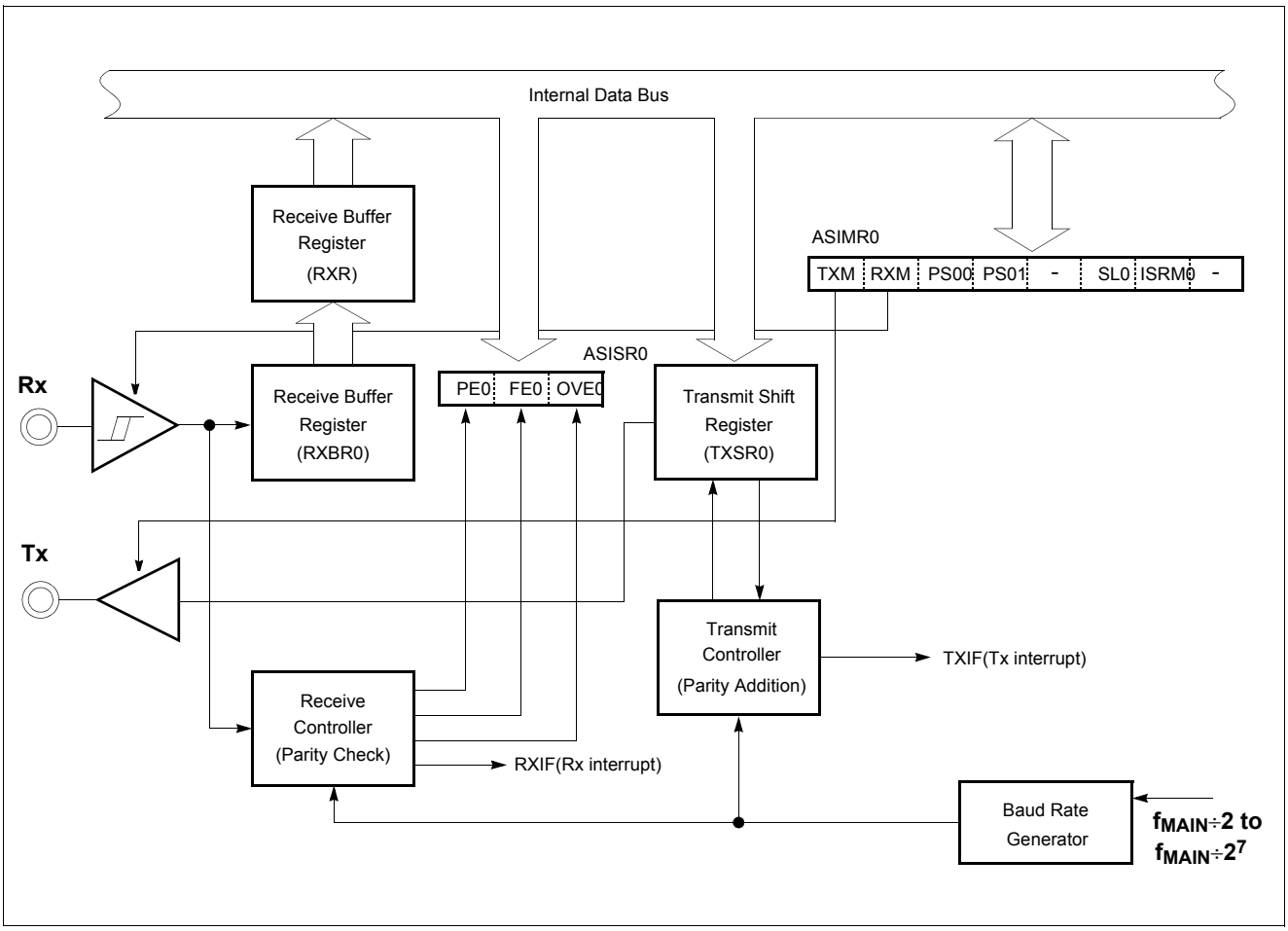


Figure 21-1 UART Block Diagram

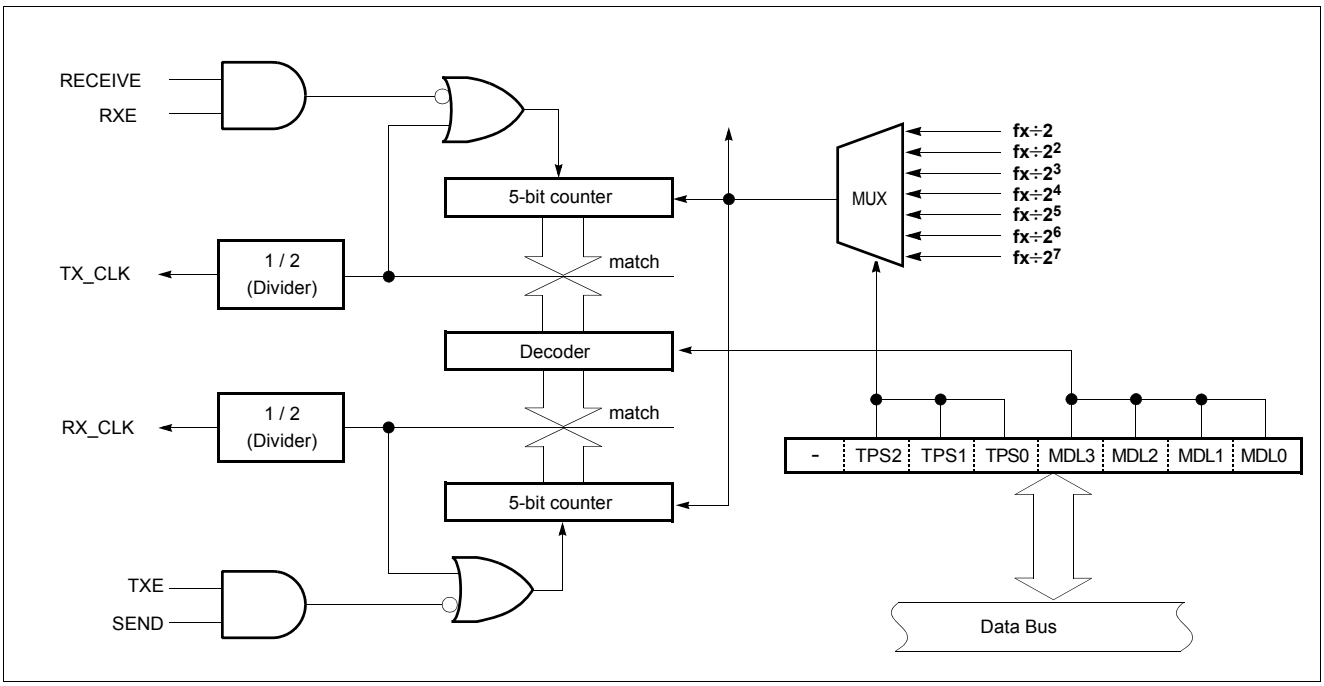


Figure 21-2 Baud Rate Generator Block Diagram

21.1 Serial Interface Configuration

The serial interface (UART) consists of the following hardware.

Item	Configuration
Register	Transmit shift register (TXSR0) Receive buffer register (RXBR0)
Control register	Serial interface mode register (ASIMR0) Baudrate generator control register (BRGCR0)

Table 21-1 Serial Interface Configuration

Transmit Shift Register (TXSR0)

This is the register for setting transmit data. Data written to TXSR0 is transmitted as serial data. When the data length is set as 7 bit, bit 0 to 6 of the data written to TXSR0 are transferred as transmit data. Writing data to TXSR0 starts the transmit operation. TXSR0 can be written by an 8 bit memory manipulation instruction. It cannot be read.

Note: *Do not write to TXSR0 during a transmit operation. The same address is assigned to TXSR0 and the receive buffer register (RXBR0). A read operation reads values from RXBR0.*

Receive Buffer Register (RXBR0)

This register is used to hold received data. When one byte of data is received, one byte of new received data is transferred from the receive shift register. When the data length is set as 7 bits, received data is sent to bits 0 to 6 of RXBR0. In this case, the MSB of RXBR0 always becomes 0. RXBR0 can be read by an 8 bit memory manipulation instruction. It cannot be written.

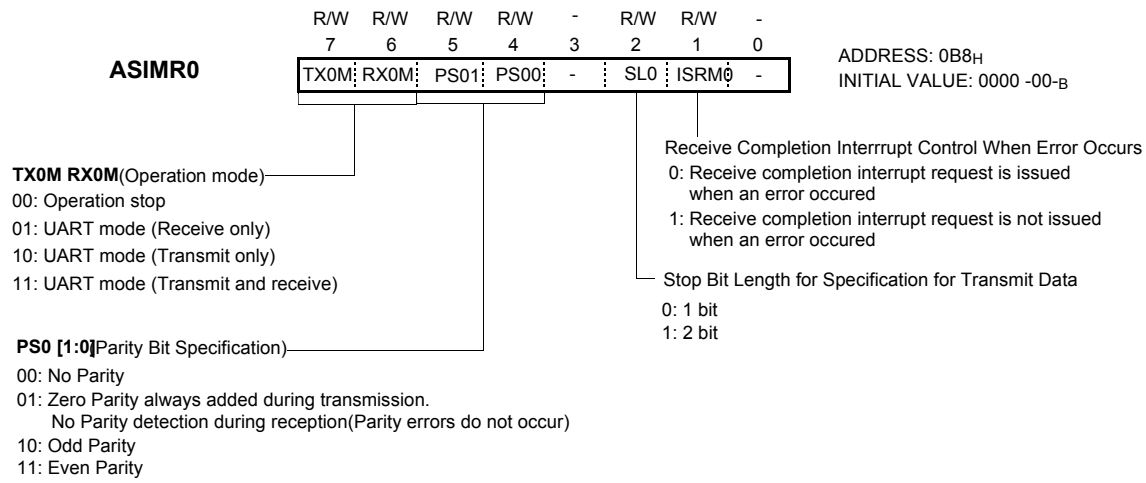
Note: *The same address is assigned to RXBR0 and the tansmit shift register (TXSR0). During a write operation, values are written to TXSR0.*

Asynchronous serial interface mode control register (ASIMR0)

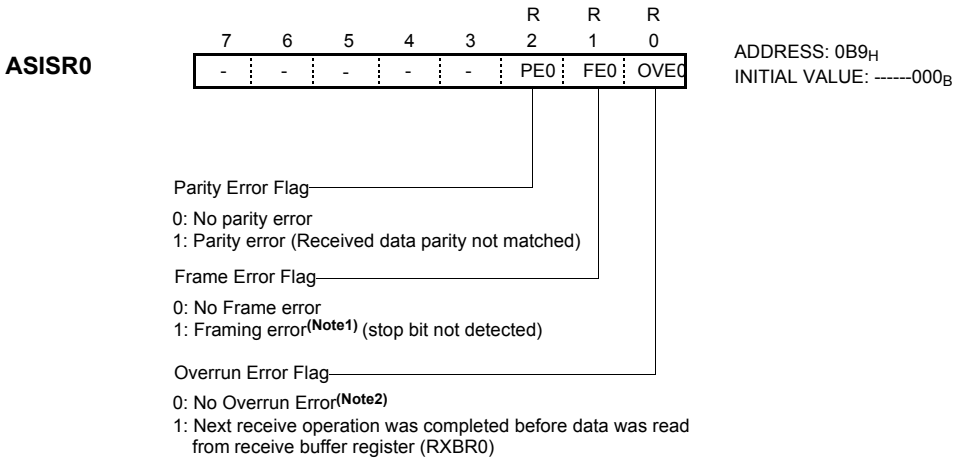
This is an 8 bit register that controls serial interface (UART)'s serial transfer operation. ASIMR0 is set by a 1 bit or 8 bit memory manipulation instruction.

Baud rate generator control register (BRGCR0)

This register sets the serial clock for serial interface. BRGCR0 is set by an 8 bit memory manipulation instruction.



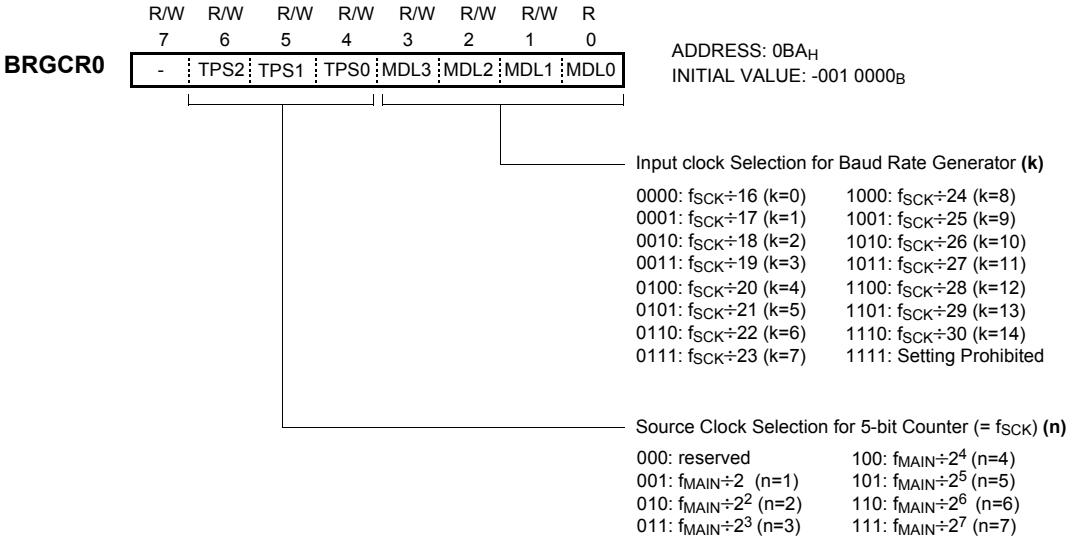
Caution : Do not switch the operation mode until the current serial transmit/receive operation has stopped.



Note : **1.** Even if a stop bit length is set to 2 bits by setting bit2(SL0) in ASIMR0, stop bit detection during a recive operation only applies to a stop bit length of 1bit.

2. Be sure to read the contents of the receive buffer register(RXBR0) when an overrun error has occurred.
Until the contents of RXBR0 are read, futher overrun errors will occur when receiving data.

Figure 21-3 Asynchronous Serial Interface Mode & Status Register



Cautions : 1. Writing to BRGCR during a communication operation may cause abnormal output from the baud rate generatior and disable further communication operations. Therefore, do not write to BRGCR during a communication operation.

Remarks : **1.** f_{SCK}: Source clock for 5-bit counter

2. f_{MAIN}: Main Oscillation Frequency

3. n: Value set via TPS0 to TPS2

4. k: Value set via MDL0 to MDL3

5. The baud rate generated from the main system clock is determined according to the following formula.

$$\text{Baud Rate} = \frac{f_{MAIN}}{2^{n+1}\,(k + 16)}$$

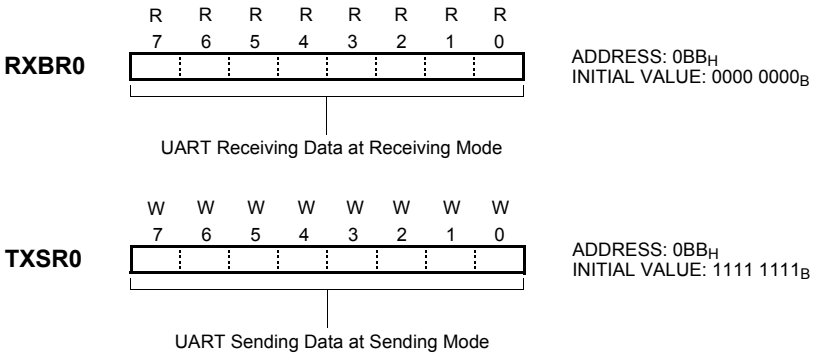


Figure 21-4 Baud Rate Generator Control Register, Receive Buffer Register, Transmit shift Register

21.2 Relationship between main clock and baud rate

The transmit/receive clock that is used to generate the baud rate is obtained by dividing the main system clock. Transmit/Receive clock generation for baud rate is made by using main system clock which is divided.

The baud rate generated from the main system clock is determined according to the following formula

$$BaudRate = \frac{fx}{2^{n+1}(K+16)}$$

- fx : main system clock oscillation frequency
- n : value set via TPS0 to TPS1($1 \le n \le 7$)
- k : value set via MDL0 to MDL3 ($0 \le n \le 14$)

Baud Rate (bps)	f_X = 11.0592M		f_X = 8.00M		f_X = 7.3728M		f_X = 6.00M		f_X = 5.00M		f_X = 4.1943	
	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)	BRGCR	Err (%)
600	-	-	-	-	-	-	-	-	-	-	7BH	1.14
1,200	-	-	7AH	0.16	78H	0.00	73H	2.79	70H	1.73	6BH	1.14
2,400	72H	0.00	6AH	0.16	68H	0.00	63H	2.79	60H	1.73	5BH	1.14
4,800	62H	0.00	5AH	0.16	58H	0.00	53H	2.79	50H	1.73	4BH	1.14
9,600	52H	0.00	4AH	0.16	48H	0.00	43H	2.79	40H	1.73	3BH	1.14
19,200	42H	0.00	3AH	0.16	38H	0.00	33H	2.79	30H	1.73	2BH	1.14
31,250	36H	0.52	30H	0.00	2DH	1.70	28H	0.00	24H	0.00	21H	-1.30
38,400	32H	0.00	2AH	0.16	28H	0.00	23H	2.79	20H	1.73	1BH	1.14
76,800	22H	0.00	1AH	0.16	18H	0.00	13H	2.79	10H	1.73	-	-
115,200	18H	0.00	11H	2.12	10H	0.00	-	-	-	-	-	-

Table 21-2 Relationship Between Main Clock and Baud Rate

22. OPERATION MODE

The system clock controller starts or stops the main frequency clock oscillator, which is controlled by system clock mode register (SCMR). shows the operating mode transition diagram.

System clock control is performed by the system clock mode register (SCMR). During reset, this register is initialized to “0” so that the main-clock operating mode is selected.

Main Active mode

This mode is fast-frequency operating mode. The CPU and the peripheral hardwares are operated on the high-frequency clock. At reset release, this mode is invoked.

SLEEP mode

In this mode, the CPU clock stops while peripherals and the oscillation source continue to operate normally.

STOP mode

In this mode, the system operations are all stopped, holding the internal states valid immediately before the stop at the low power consumption level.

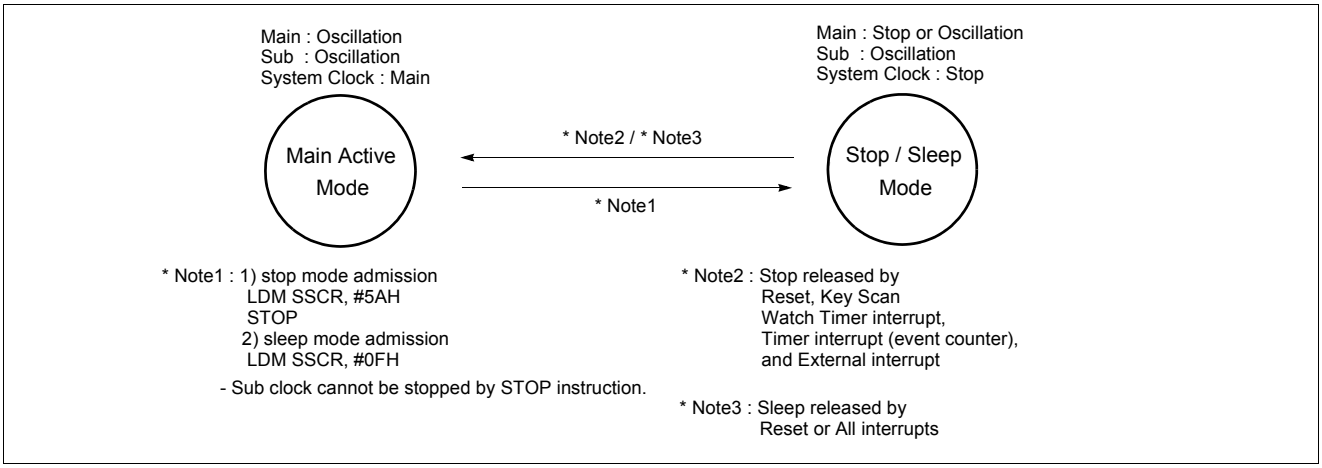


Figure 22-1 Operating Mode

22.1 Operation Mode Switching

Shifting from the Normal operation to the SLEEP mode

By writing “$0F_H$” into SSCR which will be explained in "23.1 SLEEP Mode" on page 93, the CPU clock stops and the SLEEP mode is invoked. The CPU stops while other peripherals are operate normally.

The way of release from this mode is RESET and all available interrupts.

For more detail, See "23.1 SLEEP Mode" on page 93

Shifting from the Normal operation to the STOP mode

By writing “$5A_H$” into SSCR and then executing STOP instruction, the main-frequency clock oscillation stops and the STOP mode is invoked. But sub-frequency clock oscillation is operated continuously.
After the STOP operation is released by reset, the operation mode is changed to Main active mode.

The methods of release are RESET, Key scan interrupt, Watch Timer interrupt, Timer/Event counter1 (EC0 pin) and External Interrupt.

For more details, see "23.2 STOP Mode" on page 94.

Note: *In the STOP and SLEEP operating modes, the power consumption by the oscillator and the internal hardware is reduced. However, the power for the pin interface (depending on external circuitry and program) is not directly associated with the low-power consumption operation. This must be considered in system design as well as interface circuit design.*

23. POWER DOWN OPERATION

MC80X7208 has 2 power down mode. In power down mode, power consumption is reduced considerably in battery operation that battery life can be extended a lot.

23.1 SLEEP Mode

In this mode, the internal oscillation circuits remain active.

Oscillation continues and peripherals are operate normally but CPU stops. The status of all Peripherals in this mode is shown in Table 23-1. Sleep mode is entered by writing "$0F_H$" into SSCR (address $0E9_H$).

It is released by RESET or all interrupt. To be released by interrupt, interrupt should be enabled before Sleep mode.

Sleep mode is entered by writing "$0F_H$" into Stop and Sleep Control Register(SSCR), and STOP mode is entered by writing "$5A_H$" into SSCR and then executing STOP instruction.

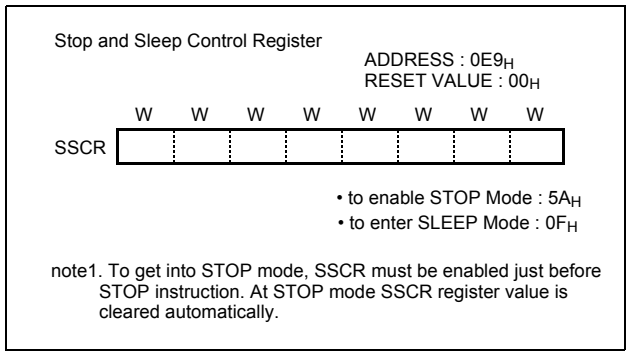


Figure 23-1 SLEEP Mode Register

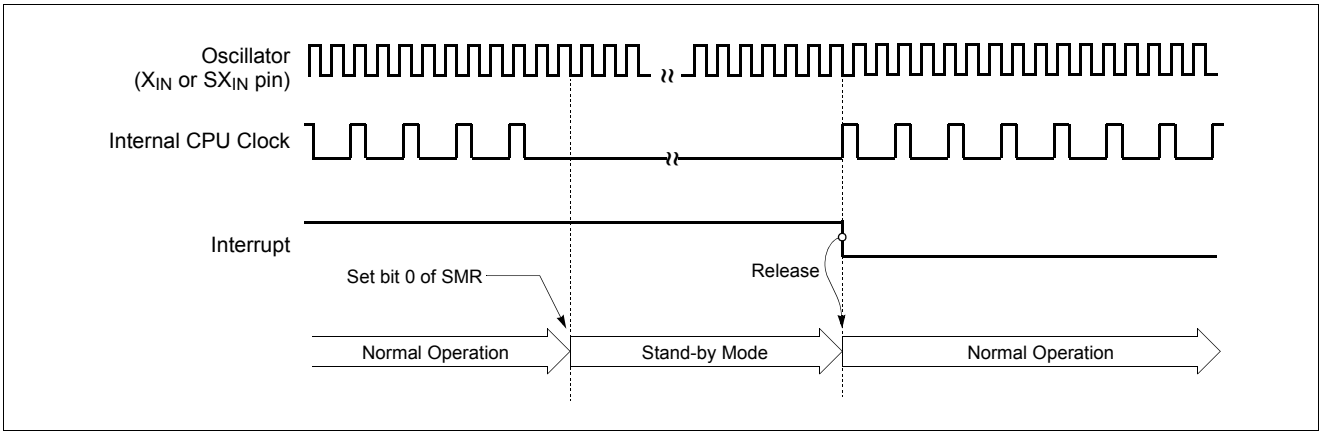


Figure 23-2 Sleep Mode Release Timing by External Interrupt

.

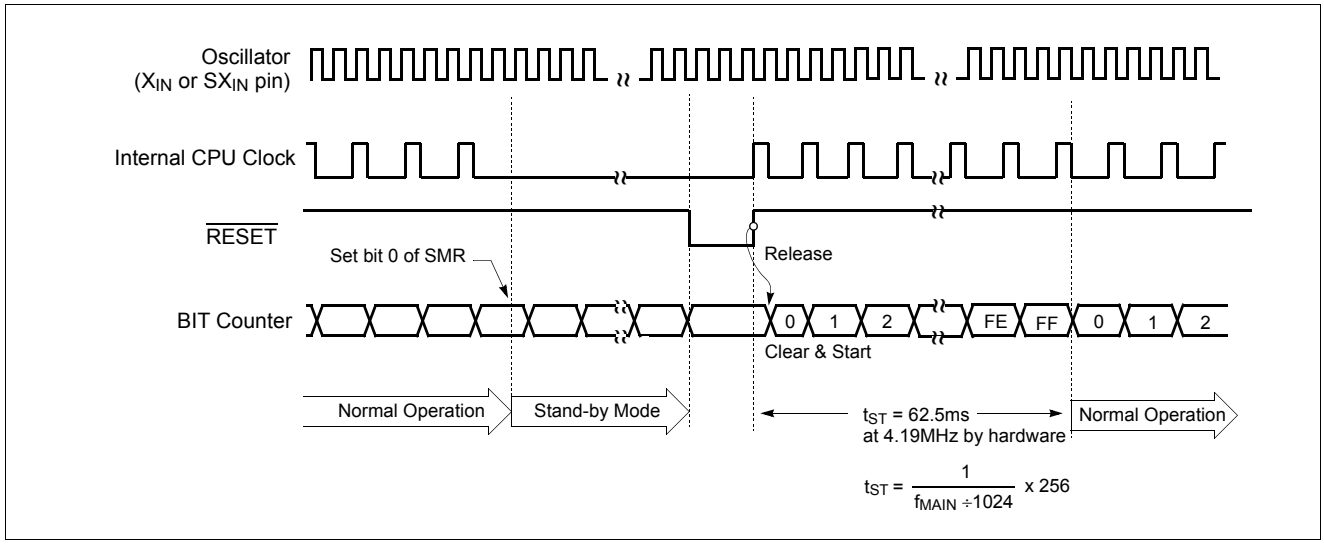


Figure 23-3 SLEEP Mode Release Timing by RESET pin

23.2 STOP Mode

For applications where power consumption is a critical factor, this device provides STOP mode for reducing power consumption.

Start The Stop Operation

The STOP mode can be entered by STOP instruction during program execution. In Stop mode, the on-chip main-frequency oscillator, system clock, and peripheral clock are stopped (Watch timer clock is oscillating continuously:. With the clock frozen, all functions are stopped, but the on-chip RAM and Control registers are held. The port pins output the values held by their respective port data register and the port direction registers. The status of peripherals during Stop mode is shown below.

Peripheral	STOP Mode	Sleep Mode
CPU	All CPU operations are disabled	All CPU operations are disabled
RAM	Retain	Retain
LCD driver	Operates continuously	Operates continuously
Basic Interval Timer	Halted	Operates continuously
Timer/Event counter 0,1	Halted (Only when the Event counter mode is enabled, Timer 0,1 operates normally)	Timer/Event counter 0,1 operates continuously
Watch Timer	Operates continuously	Operates continuously
Key Scan	Active	Active
Main-oscillation	Stop (X_{IN}=L, X_{OUT}=L)	Oscillation[1]
Sub-oscillation	Oscillation	Oscillation
I/O ports	Retain	Retain
Control Registers	Retain	Retain
Release method	by RESET, Key Scan interrupt, Watch Timer interrupt, Timer interrupt (EC0), and External interrupt	by RESET, All interrupts

Table 23-1 Peripheral Operation during Power Down Mode

1. Refer to the Table 23-2.

Operating Clock source	Main Operating Mode	Sleep Mode	Stop Mode
Main Clock	Oscillation	Oscillation	Stop
Sub Clock	Oscillation	Oscillation	Oscillation
System Clock	Active	Stop	Stop
Peri. Clock	Active	Active	Stop[1]

Table 23-2 Clock Operation of STOP and SLEEP mode

1. Except watch timer(sub clock) and voltage booster

Note: *Since the X_{IN} pin is connected internally to GND to avoid current leakage due to the crystal oscillator in STOP mode, do not use STOP instruction when an external clock is used as the main system clock.*

In the Stop mode of operation, V_{DD} can be reduced to minimize power consumption. Be careful, however, that V_{DD} is not reduced before the Stop mode is invoked, and that V_{DD} is restored to its normal operating level before the Stop mode is terminated.

The reset should not be activated before V_{DD} is restored to its normal operating level, and must be held active long enough to allow the oscillator to restart and stabilize.
And after STOP instruction, at least two or more NOP instruction should be written as shown in example below.

Example)

```
:
LDM    CKCTLR,#0000_1111B
STOP
NOP
NOP
:
```

The Interval Timer Register CKCTLR should be initialized by software in order that oscillation stabilization time should be longer than 20ms before STOP mode.

Release the STOP mode

The exit from STOP mode is using hardware reset or external interrupt, watch timer, key scan or timer interrupt (EC0).

To release STOP mode, corresponding interrupt should be enabled before STOP mode.
Specially as a clock source of Timer/Event counter, EC0 pin can release it by Timer/Event counter Interrupt request.

Reset redefines all the control registers but does not change the on-chip RAM. External interrupts allow both on-chip RAM and Control registers to retain their values.

Start-up is performed to acquire the time for stabilizing oscillation. During the start-up, the internal operations are all stopped.

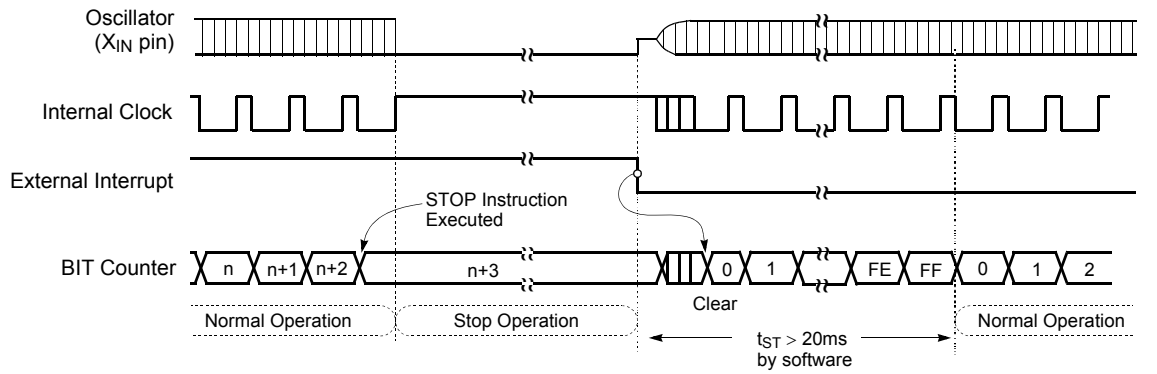


Figure 23-4 STOP Mode Release Timing by External Interrupt

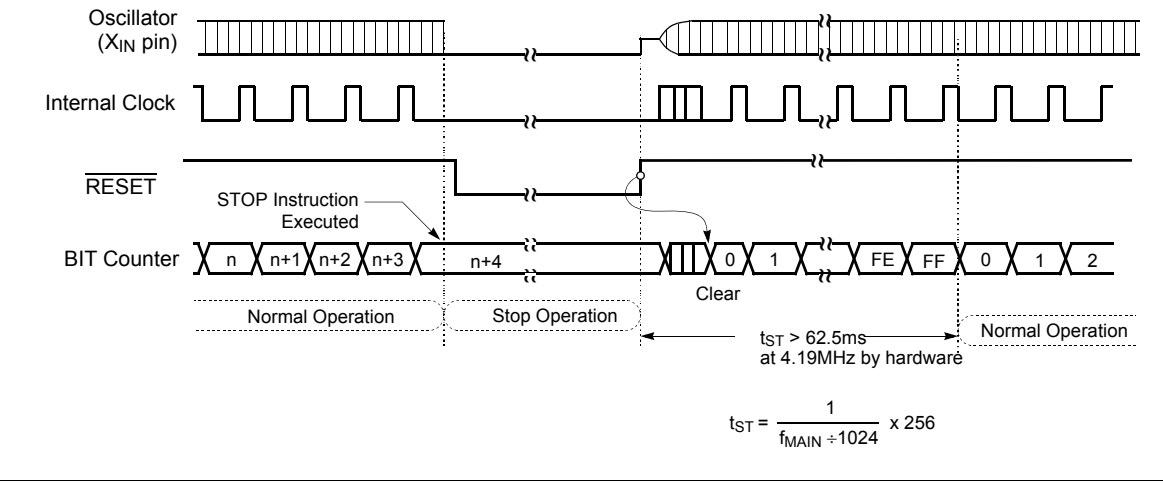


Figure 23-5 STOP Mode Release Timing by RESET

Minimizing Current Consumption

The Stop mode is designed to reduce power consumption. To minimize current drawn during Stop mode, the user should turn-off output drivers that are sourcing or sinking current, if it is practical.

Note: *In the STOP operation, the power dissipation associated with the oscillator and the internal hardware is lowered; however, the power dissipation associated with the pin interface (depending on the external circuitry and program) is not directly determined by the hardware operation of the STOP feature. This point should be little current flows when the input level is stable at the power voltage level (V_{DD}/V_{SS}); however, when the input level becomes higher than the power voltage level (by approximately 0.3V), a current begins to flow. Therefore, if cutting off the output transistor at an I/O port puts the pin signal into the high-impedance state, a current flow across the ports input transistor, requiring it to fix the level by pull-up or other means.*

It should be set properly that current flow through port doesn't exist.

First consider the setting to input mode. Be sure that there is no current flow after considering its relationship with external circuit. In input mode, the pin impedance viewing from external MCU is very high that the current doesn't flow.

But input voltage level should be V_{SS} or V_{DD}. Be careful that if unspecified voltage, i.e. if uniformed voltage level (not V_{SS}or V_{DD}) is applied to input pin, there can be little current (max. 1mA at around 2V) flow.

If it is not appropriate to set as an input mode, then set to output mode considering there is no current flow. Setting to High or Low is decided considering its relationship with external circuit. For example, if there is external pull-up resistor then it is set to output mode, i.e. to High, and if there is external pull-down resistor, it is set to low.

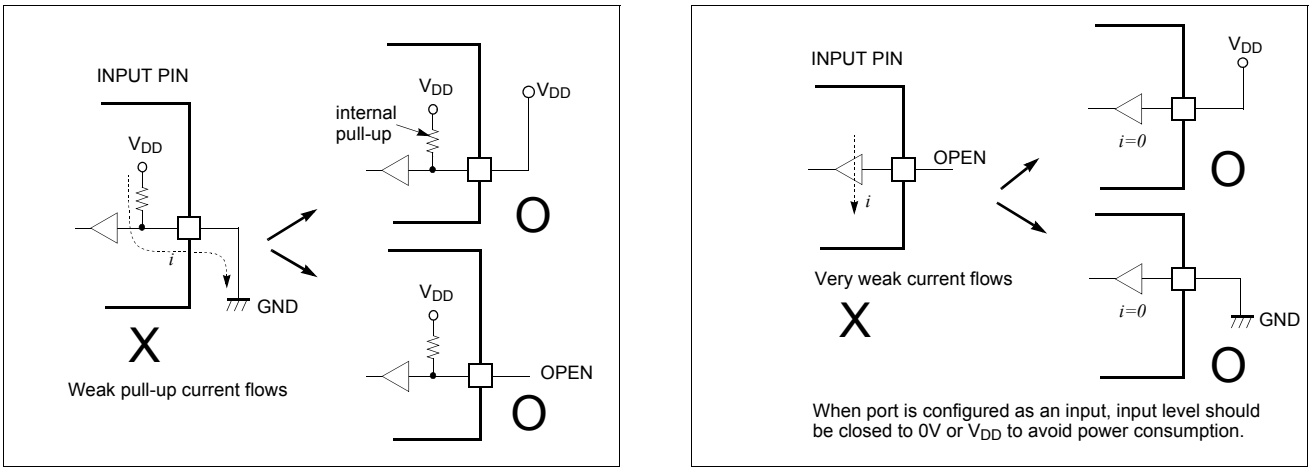


Figure 23-6 Application Example of Unused Input Port

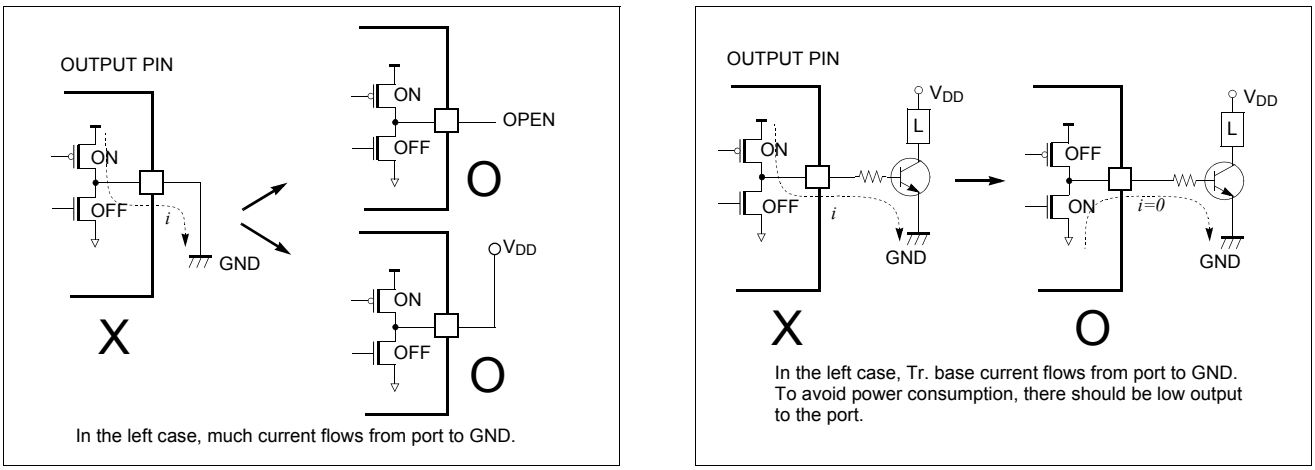


Figure 23-7 Application Example of Unused Output Port

24. OSCILLATOR CIRCUIT

The MC80X7208 has two oscillation circuits internally. X_{IN} and X_{OUT} are input and output for main frequency and SX_{IN} and SX_{OUT} are input and output for sub frequency, respectively, inverting amplifier which can be configured for being used as an on-chip oscillator, as shown in.

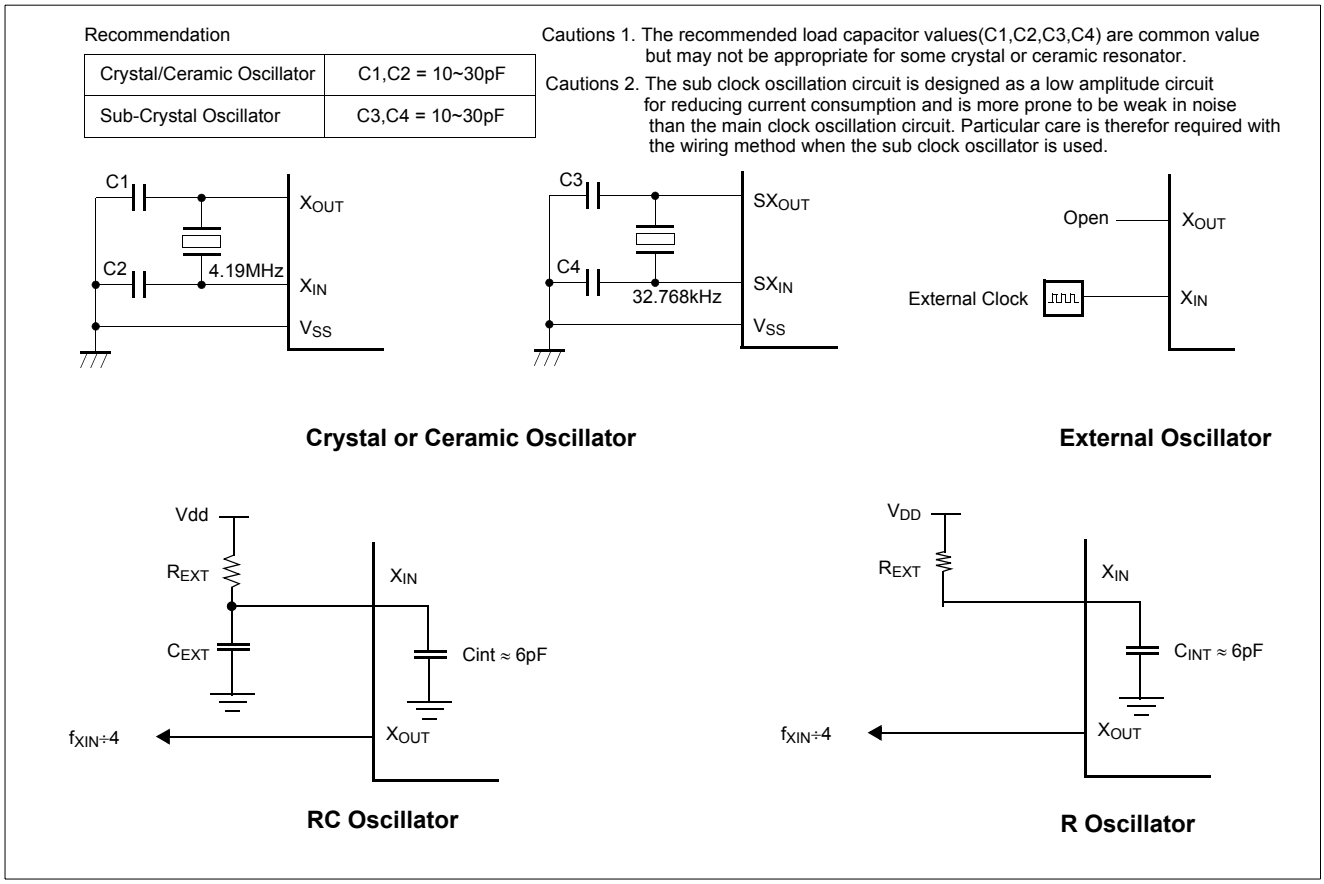


Figure 24-1 Oscillation Circuit

Oscillation circuit is designed to be used either with a ceramic resonator or crystal oscillator. Since each crystal and ceramic resonator have their own characteristics, the user should consult the crystal manufacturer for appropriate values of external components.

In addition, see for the layout of the crystal.

Note: *Minimize the wiring length. Do not allow the wiring to intersect with other signal conductors. Do not allow the wiring to come near changing high current. Set the potential of the grounding position of the oscillator capacitor to that of Vss. Do not ground it to any ground pattern where high current is present. Do not fetch signals from the oscillator.*

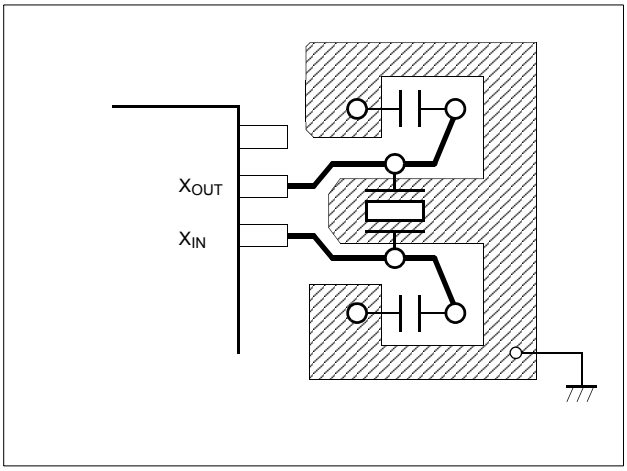


Figure 24-2 Layout of Oscillator PCB circuit

25. RESET

The MC80X7208 has two types of reset generation procedures; one is an external reset input, the other is a watchdog timer reset. Table 25-1 shows on-chip hardware initialization by reset action.

On-chip Hardware	Initial Value	On-chip Hardware	Initial Value
Program counter (PC)	($FFFF_H$) - ($FFFE_H$)	Peripheral clock	On
RAM page register (RPR)	0	PFD	Disable
G-flag (G)	0	Control registers	Refer to Table 8-1 on page 28
Operation mode	Main-frequency clock	Voltage Booster	Disable

Table 25-1 Initializing Internal Status by Reset Action

25.1 External Reset Input

The reset input is the $\overline{RESET}$ pin, which is the input to a Schmitt Trigger. A reset accomplished by holding the $\overline{RESET}$ pin to low for at least 8 oscillator periods, within the operating voltage range and oscillation stable, it is applied, and the internal state is initialized. After reset, 65.5ms (at 4MHz) and 7 oscillator periods are required to start execution as shown in .

Internal RAM is not affected by reset. When V_{DD} is turned on, the RAM content is indeterminate. Therefore, this RAM should be initialized before read or tested it.

When the $\overline{RESET}$ pin input goes to high, the reset operation is released and the program execution starts at the vector address stored at $FFFE_H$ - $FFFF_H$.

A connection for normal power-on-reset is shown in .

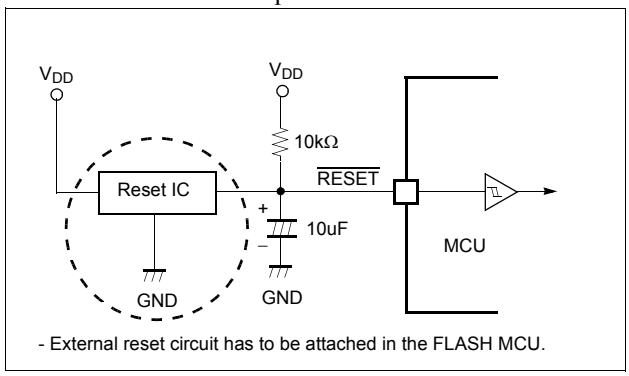


Figure 25-1 Normal Power-on-Reset Circuit

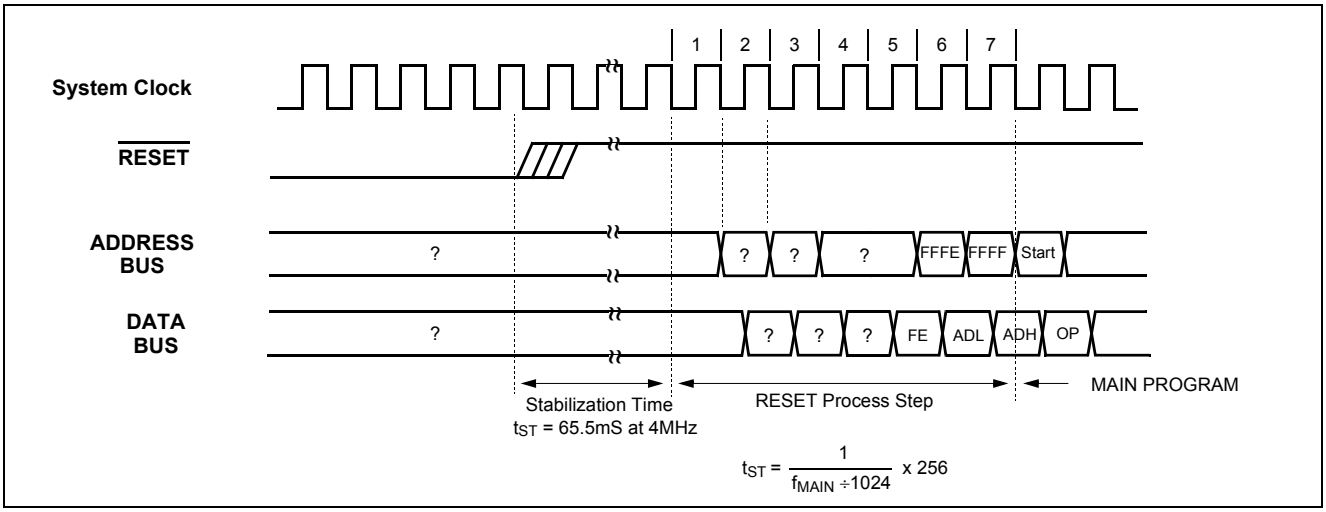


Figure 25-2 Timing Diagram of RESET

25.2 Watchdog Timer Reset

Refer to “14. WATCH DOG TIMER” on page 61.

26. POWER FAIL DETECTOR (Scope: V_{DD}=3.4V to 5.5V, V_{SS}=0V)

The MC80X7208 has an on-chip power fail detection circuitry to immunize against power noise. A configuration register, PFDR, can enable or disable the power fail detect circuitry. Whenever V_{DD} falls close to or below power fail voltage level for 100ns, the power fail situation may reset or freeze MCU according to PFDMS bits of PFDR. Refer to "Figure 26-1 power fail Detector Register" on page 100.

In the in-circuit emulator, power fail function is not implemented and user can not experience with it. Therefore, after final development of user program, this function may be experienced or evaluated.

Also this function is only operated more operating voltage than 3.4V.

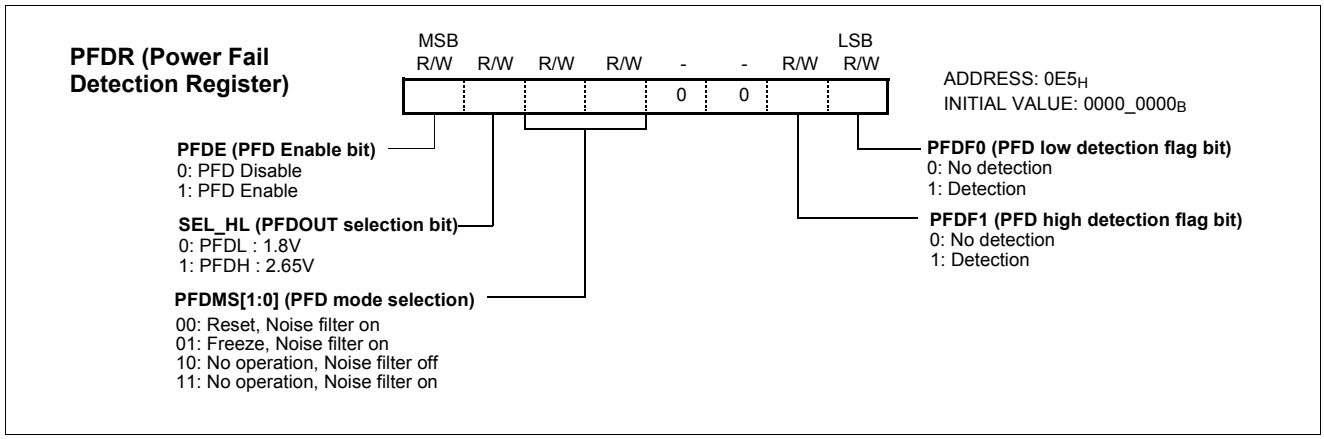


Figure 26-1 power fail Detector Register

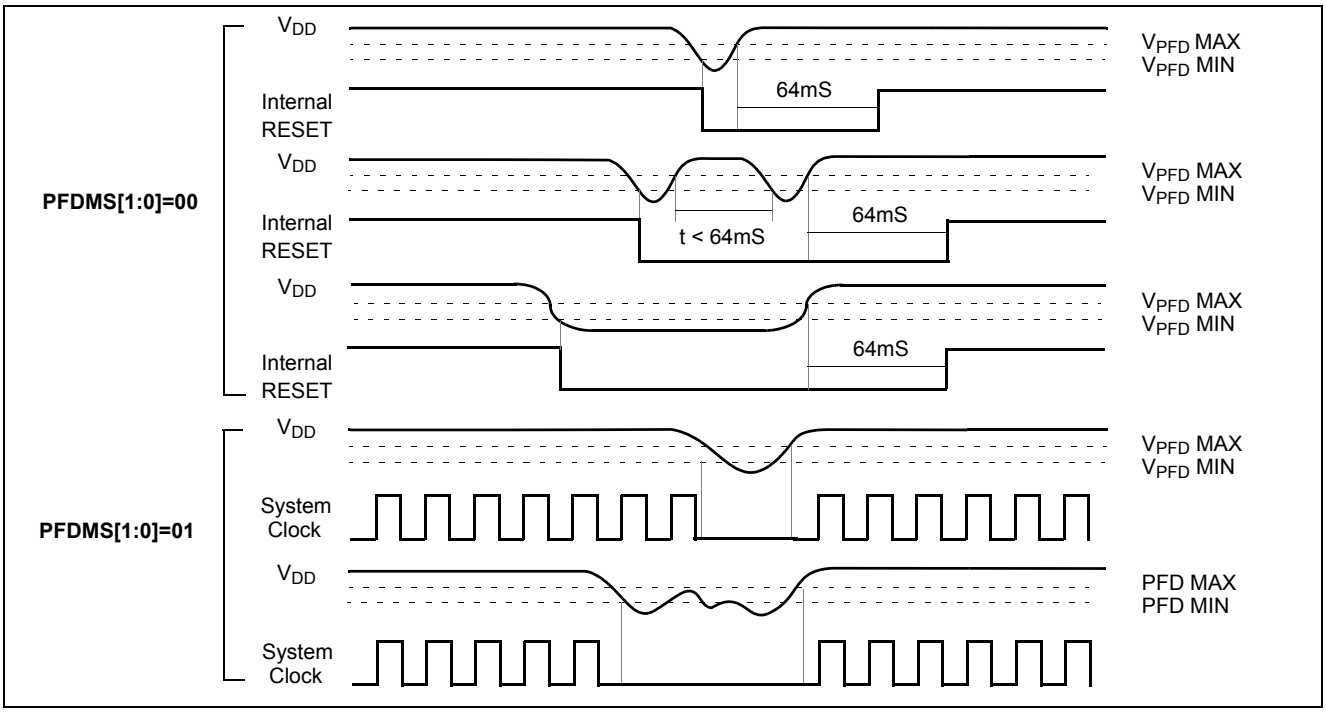


Figure 26-2 Power Fail Detector Situations

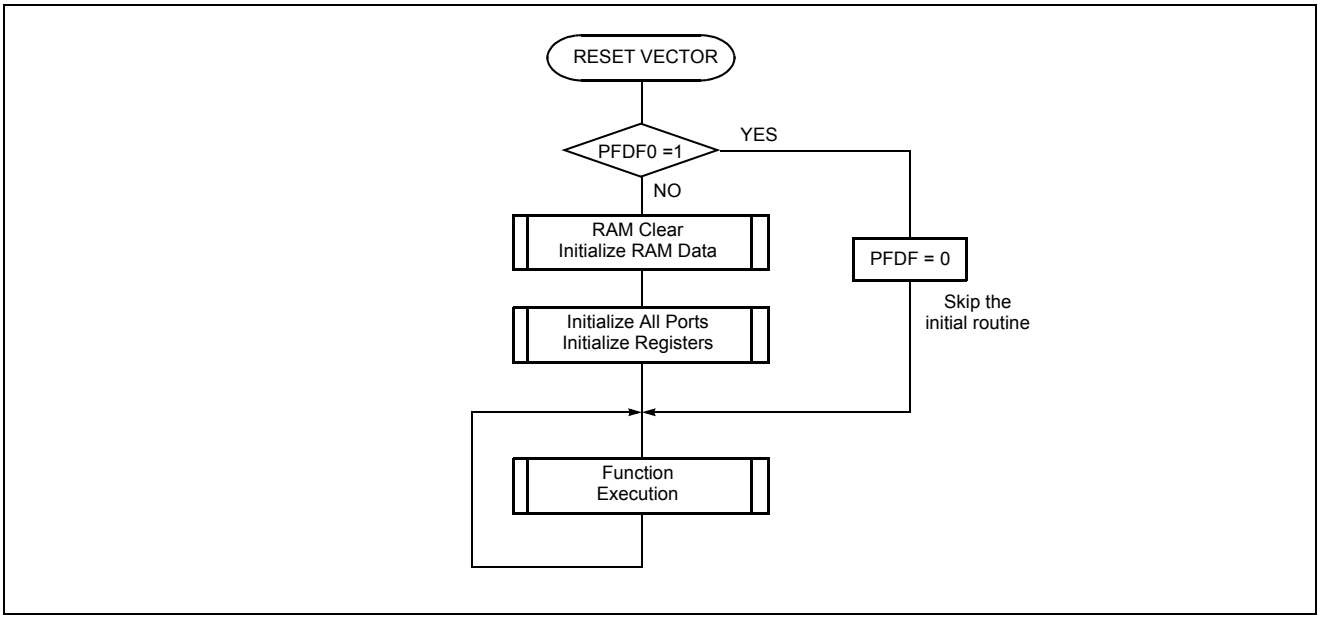


Figure 26-3 Example Flow of Reset flow by Power fail

27. LOW VOLTAGE DETECTOR (LVD)

An on board voltage comparator checks that V_{DD} is at the required level to ensure correct operation of the device. If V_{DD} is below a certain level, Low voltage detector forces the device into low voltage detection mode.

In the low voltage detection mode, there is no power consumption except stop current, stop mode release function is disabled. All I/O port is configured as input mode and data memory is retained until voltage through external capacitor is worn out.

Reset signal result from new battery(normally 3V) wakes the low voltage detection mode and come into normal reset state. it depends on user whether to execute RAM clear routine or not.

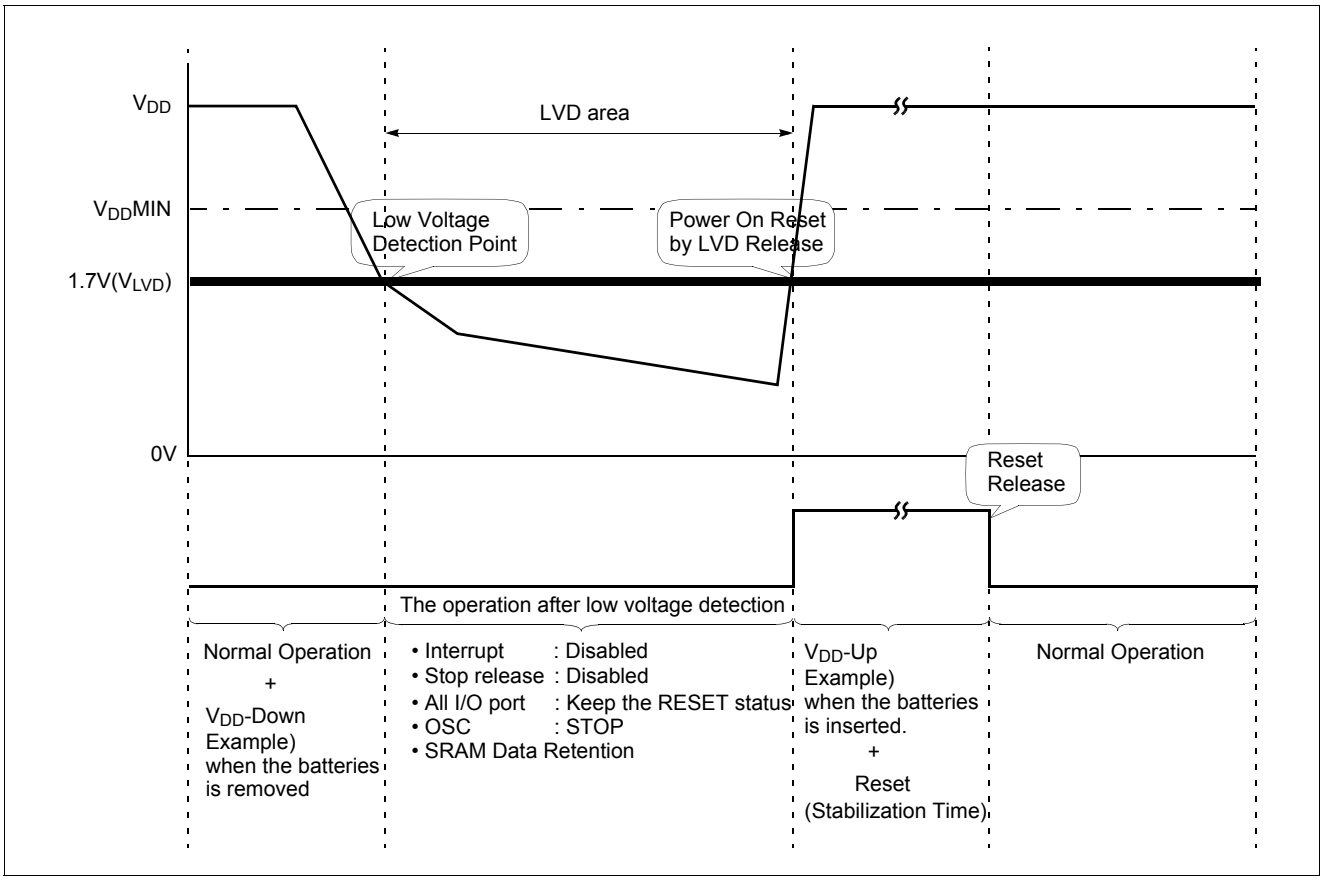
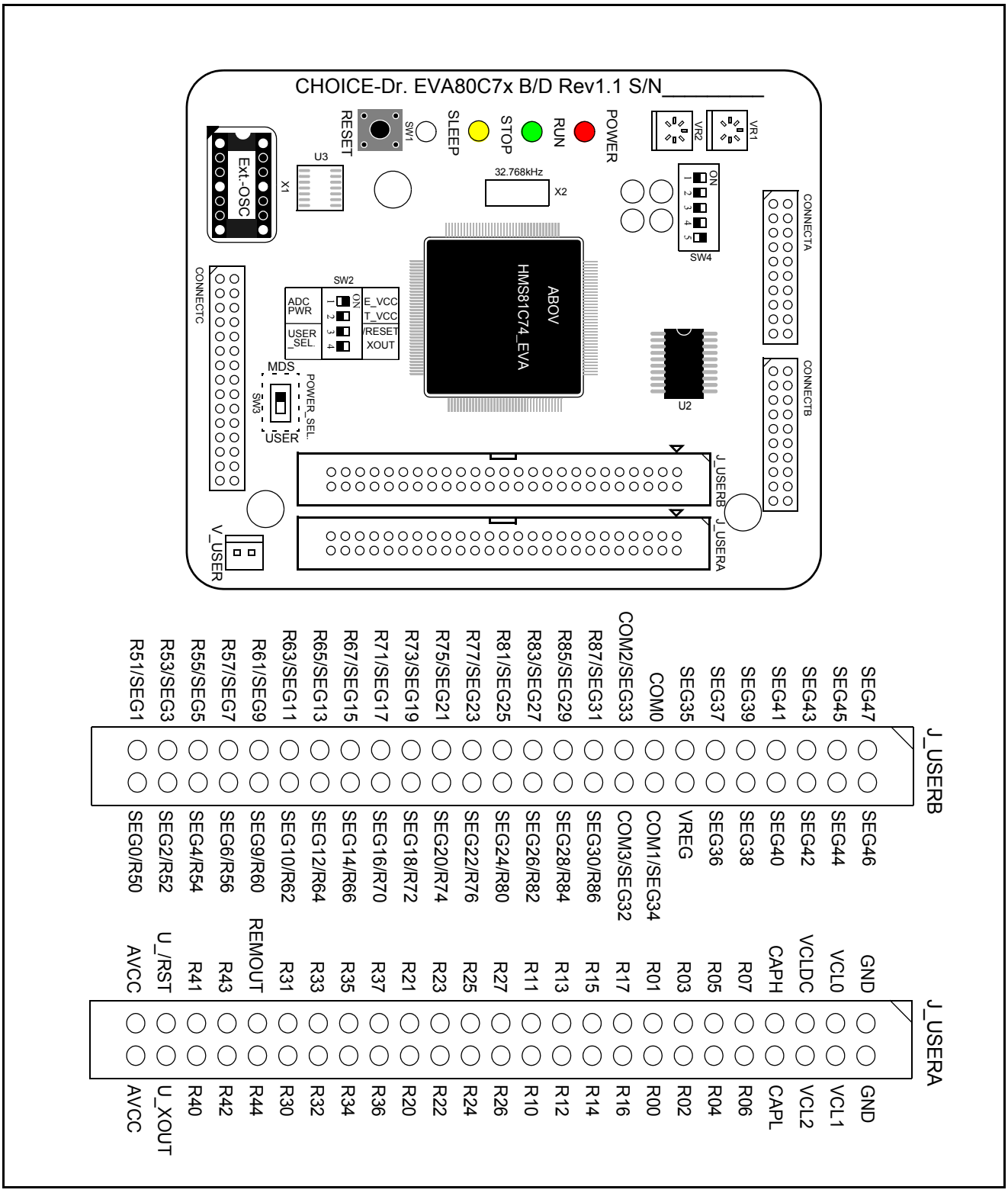


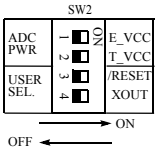
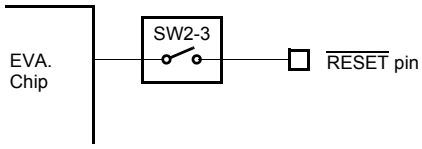
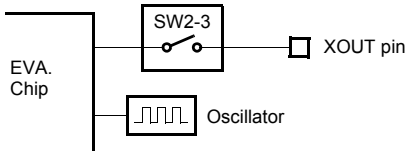
Figure 27-1 LVD Detection and Release Timing Diagram

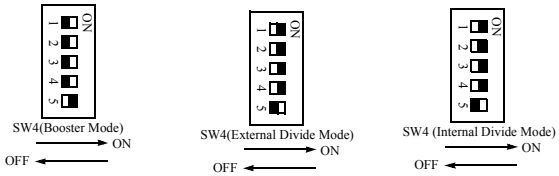
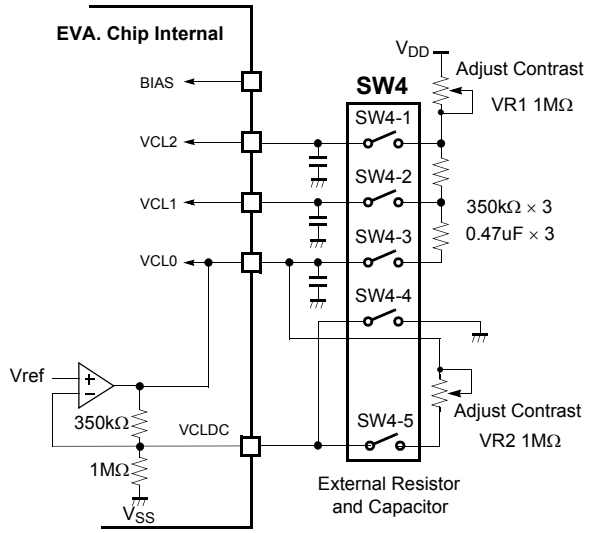
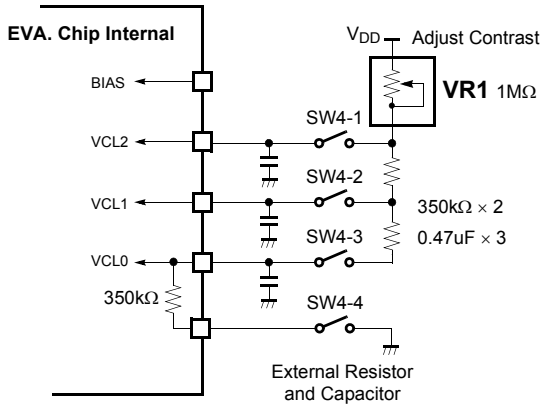
28. EMULATOR EVA. BOARD SETTING

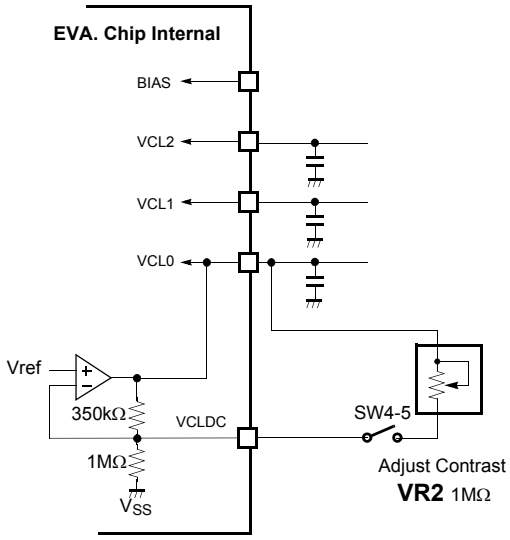


DIP Switch and VR Setting

Before execute the user program, keep in your mind the below configuration

DIP S/W, VR		Description	ON/OFF Setting
SW1	-	Emulator Reset Switch. Reset the Emulator.	Reset the Emulator.
SW2	1 2	ADC voltage is supplied from EVA B/D voltage ADC voltage is supplied from Target(User POD pin) voltage 	SW2-1 must be **OFF** position. SW2-2 must be **OFF** position.
	3	POD RESET pin configuration 	Normally **OFF**. EVA. chip can be reset by external user target board. **ON** : Reset is available by either user target system board or Emulator RESET switch. **OFF** : Reset the MCU by Emulator RESET switch. Does not work from user target board.
	4	POD XOUT pin configuration 	Normally **OFF**. MCU XOUT pin is disconnected internally in the Emulator. Some circumstance user may connect this circuit. **ON** : Output XOUT signal **OFF** : Disconnect circuit
SW3	1	This switch selects Eva. B/D Power supply source. 	Normally **MDS**. This switch select Eva. B/D Power supply source.

DIP S/W, VR		Description	ON/OFF Setting
SW4	1 2 3 4 5	External Bias Resistors Connection  	Must be **ON** position. It serves the external bias resistors. If this switches are turned off, LCD bias voltage does not supplied, floated because there are no internal bias resistors and bias Tr. inside the Emulator.
VR1	-	External Driver Mode : Adjust the LCD contrast. It supply bias voltage and adjust the VCL1 voltage. 	Adjust the proper position as well as LCD display good.

DIP S/W, VR		Description	ON/OFF Setting
VR2	-	Internal boost mode : Adjust the LCD contrast. It supply bias voltage and adjust the VCL1 voltage. 	Adjust the proper position as well as LCD display good.

29. FLASH PROGRAMMING

29.1 FLASH Configuration Byte

Except the user program memory, there is configuration byte(address $20FF_H$) for the selection of program lock and RC oscillation. The configuration byte of FLASH is shown as . It could be served when user use the FLASH programmer.

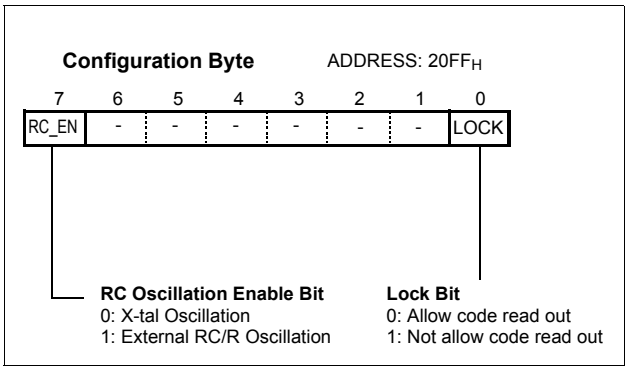


Figure 29-1 The FLASH Configuration Byte

29.2 FLASH Programming

The MC80F7208 is an FLASH type microcontroller. Its internal user memory is constructed with FLASH.

The FLASH type microcontroller is generally used for chip evaluation, first production, small amount production, fast mass production, etc.

Blank of internal FLASH is filled by 00_H, not FF_H.

Note: *In any case, you have to use the *.OTP file for programming, not the *.HEX file. After assemble, both OTP and HEX file are generated by automatically. The HEX file is used during program emulation on the emulator.*

How to Program

To program the FLASH devices, user can use ABOV own programmer.

ABOV own programmer list

Manufacturer: ABOV Semiconductor Programmer:

Choice-Sigma II
StandAlone-Gang4 I/II
PGM-plus II(+Socket)/III

The Choice-Sigma II is a ABOV Universal Single Programmer for all of ABOV FLASH/OTP devices, also the StandAlone-Gang4 I/II can program four FLASH/OTPs at once for ABOV FLASH/OTP.

Ask to ABOV sales part for purchasing or more detail.

Programming Procedure

1. Select device MC80F7208 you want.
2. Load the *.OTP file from the PC. The file is composed of Motorola-S1 format.
3. Set the programming address range as below table.

Address	Set Value
Buffer start address	$E000_H$
Buffer end address	$FFFF_H$
Device start address	$E000_H$

4. Mount the socket adapter on the programmer.
5. Start program/verify.

30. IN-SYSTEM PROGRAMMING (ISP:ONLY FLASH MCU IS AVAILABLE)

30.1 Getting Started / Installation

The In-System Programming (ISP) is performed without removing the microcontroller from the system. The In-System Programming(ISP) facility consists of a series of internal hardware resources coupled with internal firmware through the serial port. The In-System Programming (ISP) facility has made in-circuit programming in an embedded application possible with a minimum of additional expense in components and circuit board area.

The following section details the procedure for accomplishing the installation procedure.

1. **Power off a target system.**
2. **Configure a target system as ISP mode.**

   Refer to "30.3 Hardware Conditions to Enter the ISP Mode" on page 110.
3. **Attach a ISP B/D into a target system.**
4. **Connect the serial (RS-232C) cable between a ISP board and available serial port of your PC.**
5. **Run the ABOV ISP software.**
   Download the ISP S/W from *www.abov.co.kr.*
   Unzip the download file and run ISP_800.exe.
6. **Select a COM port and a device in the ISP software.**
7. **Power on a target system.**
8. **Excute ISP command such as read, program, auto, ... by pressing buttons on the ISP software..**

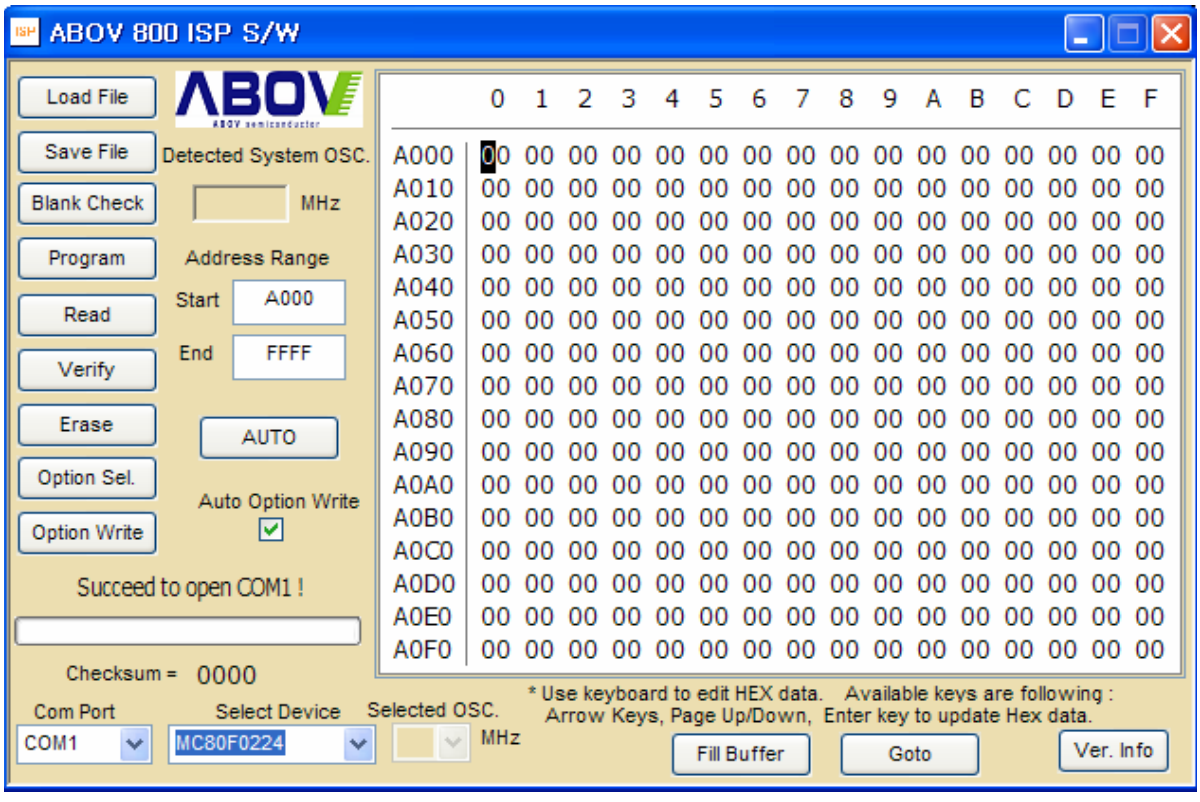


Figure 30-1 ISP software

30.2 Basic ISP S/W Information

The is the ISP software based on Windows™. This software is only supporting devices with UART. Main feature is automatically to search baudrates in range Table 30-2. In case of not detecting its baudrates an user manually have to select specific baudrates.

Function	Description
Load File	Load the data from the selected file storage into the memory buffer.
Save File	Save the current data in your memory buffer to a disk storage by using the Motorola HEX format.
Blank Check	Verify whether or not a device is in an erased or un-programmed state.
Program	This button enables you to place new data from the memory buffer into the target device.
Read	Read the data in the target MCU into the buffer for examination. The checksum will be displayed on the checksum box.
Verify	Assures that data in the device matches data in the memory buffer. If your device is secured, a verification error is detected.
Erase[1]	Erase the data in your target MCU before programming it.
Option Selection	Set the configuration data of target MCU. The security locking is set with this button.
Option Write	Progam the configuration data of target MCU. The security locking is performed with this button.
AUTO	Following sequence is performed ; 1.Erase 2.Program 3.Verify 4.Option Write
Edit Buffer	Modify the data in the selected address in your buffer memory
Fill Buffer	Fill the selected area with a data.
Goto	Display the selected page.
Detected System OSC.	Display user system clock which is detected in Auto baud rate mode.
Start _____	Starting address
End _____	End address
Checksum	Display the checksum(Hexdecimal) after reading the target device.
COM Port	Select a serial port.
Selected OSC[2].	Specify your target oscillator value with discarding below point in ACK mode only.
Select Device	Select a target mcu. User can select mode between auto baud rate and ACK. For example, MC80F0324 supports both mode. Select MC80F0324 for auto baud and MC80F0324_ACK for ACK mode.

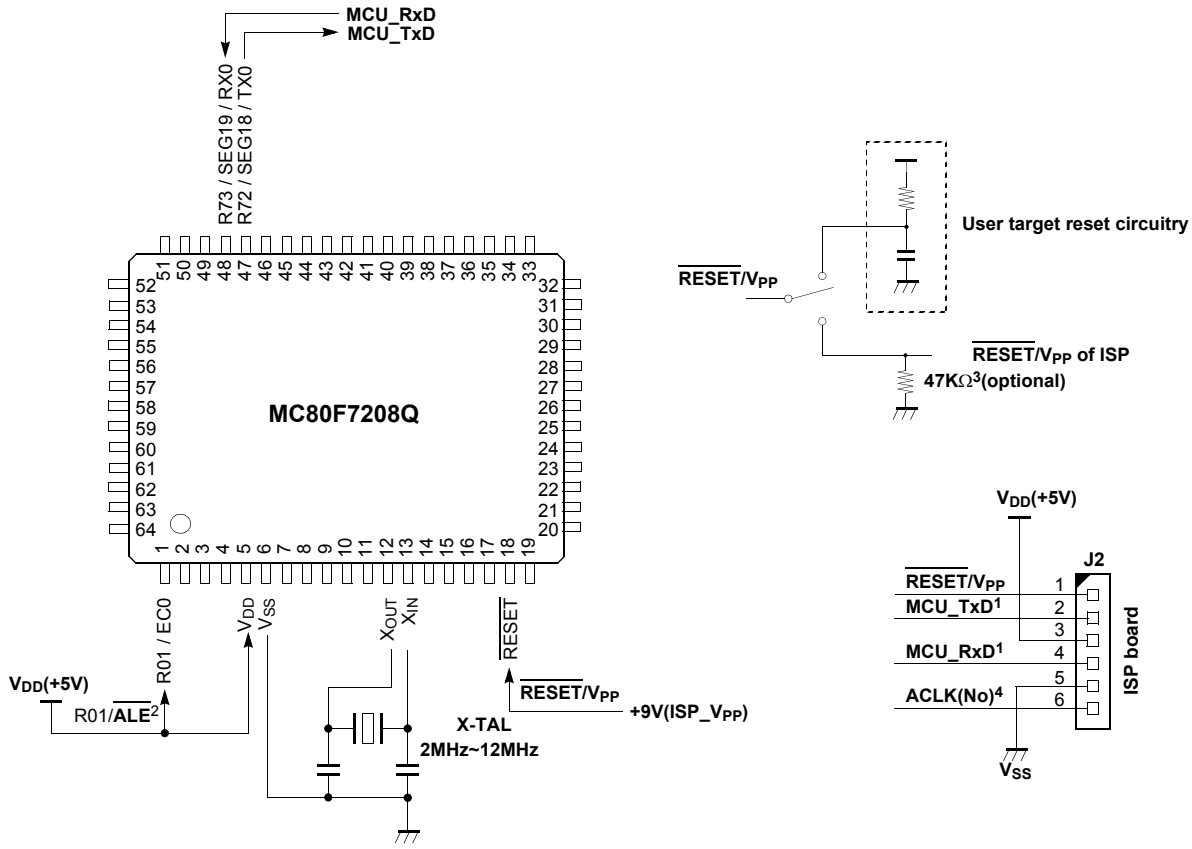
Table 30-1 ISP Function Description

1. MCU configuration value is erased after erase operation. It must be configured to match with user target board. Otherwise, it is failed to enter ISP mode, or its operation is not desirable.
2. For MC80F0204/ MC80F0316 devices, if the user selected OSC. configuration is not matched with user target board, the ISP operation is also failed. Careful attention is needed during configuring OSC. type.

30.3 Hardware Conditions to Enter the ISP Mode

The boot loader can be executed by holding $\overline{ALE}$ high, $\overline{RESET}$/V_{PP} as +9V, and ACLK0(optional) with OSC. 1.8432MHz. The ISP function uses following pins: TxD0, RxD0, ALEB, ACLK0(optional) and $\overline{RESET}$/V_{PP}.

But ACLK0(ACK mode) is not supported at MC80F7208Q/L.



1. If other signals affect UART communiction in ISP mode, disconnect these pins by using a jumper or a switch.
2. The $\overline{ALE}$ can be shared with other function. Toggle between ISP and user mode.

Caution: The $\overline{ALE}$ is only used for the ISP entry, connecting the $\overline{ALE}$ to V_{SS} into user mode is more effective than connecting it to V_{DD} to prevent malfunction(entering ISP mode) by noise.
If the V_{PP} is changed from 0 to 9V(@Vdd=5V) by a noise, the ISP mode could be enabled.
Please make the ALE low to prevent unexpected entering the ISP mode from the user mode.

3. The pull down resistor is optional. If user set to ISP mode without connection ISP board, the Vpp/$\overline{Reset}$ port is in floating state.
   The pull down resistor is for blocking this undefined state.
   Do not power on a application B/D in ISP mode without connecting a ISP board.
4. Refer to the section 30.5 explaining the auto baudrate.
   ACK mode is not supported at MC80F7208Q/L.

Figure 30-2 ISP Configuration

30.4 Sequence to enter ISP mode/User mode

Sequence to enter ISP mode from user mode

1. **Power off a target system.**
2. **Configure a target system as ISP mode.**
3. **Attach a ISP B/D into a target system.**
4. **Power on a target system.**

Sequence to enter user mode from ISP mode.

1. **Close the ISP S/W.**
2. **Power off a target system.**
3. **Configure a target system as user mode**
4. **Detach a ISP B/D from a target system.**

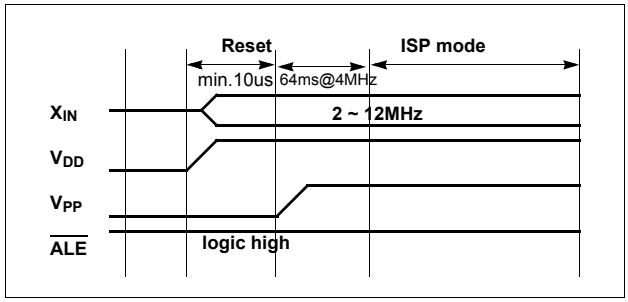


Figure 30-3 Timing diagram to enter the ISP mode

30.5 Difference between auto baud rate and ACK mode

Auto Baud Rate Mode

The ISP S/W detects user system clock and MCU configure a baud rate automatically. ***Do not need to connect the ACK pin of target MCU to ISP B/D.***

ACK mode (No Use)

If the ISP S/W can not detect user system clock, users have to enter a user system clock. This mode is only used when failed to detect user system clock automatically. ***Need to connect the ACK pin to ISP B/D.***

The mode supported with devices is shown in Table 30-2 .

	Auto Baud Rate mode	ACK mode
MC80F0204	-	O
MC80F0224	O	O
MC80F0448	O	O
MC80F7108 MC80F7208 MC80F7308 MC80F7408	O	-
MC80F7532 MC80F7632	O	O

Table 30-2 Supported modes according to devices

Reference Frequency (MHz)	Available Frequency(MHz)	57600 BPS	19200 BPS
1.8432	1.79 ~ 1.89		O
1.9968	1.94 ~ 2.05		O
3.0720	2.98 ~ 3.16		O
3.5328	3.43 ~ 3.63		O
3.6864	3.58 ~ 3.79	O	
3.9168	3.80 ~ 4.03	O	
4.1472	4.03 ~ 4.27	O	
4.6080	4.47 ~ 4.74	O	
5.0688	4.92 ~ 5.22	O	
5.5296	5.37 ~ 5.69	O	
5.9904	5.81 ~ 6.17	O	
6.4512	6.26 ~ 6.64	O	
6.9120	6.71 ~ 7.11	O	
7.8336	7.60 ~ 8.06	O	
8.7552	8.50 ~ 9.01	O	
9.2160	8.94 ~ 9.49	O	
10.1376	9.84 ~ 10.44	O	
11.0592	10.73 ~ 11.39	O	
11.9808	11.63 ~ 12.34	O	
15.6672	15.20 ~ 16.13	O	

Table 30-3 Detected frequency in Auto Baud Rate mode

30.6 Reference ISP Circuit Diagram

The ISP software and hardware circuit diagram are provided at www.abov.co.kr . To get a ISP B/D, contact to sales department. The following circuit diagram is for reference use.

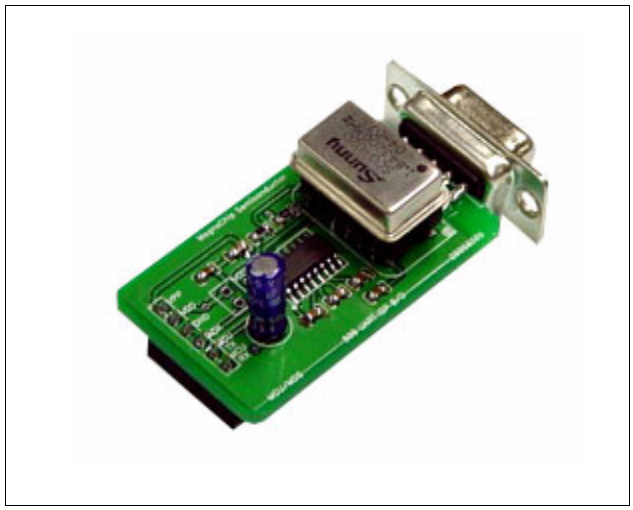
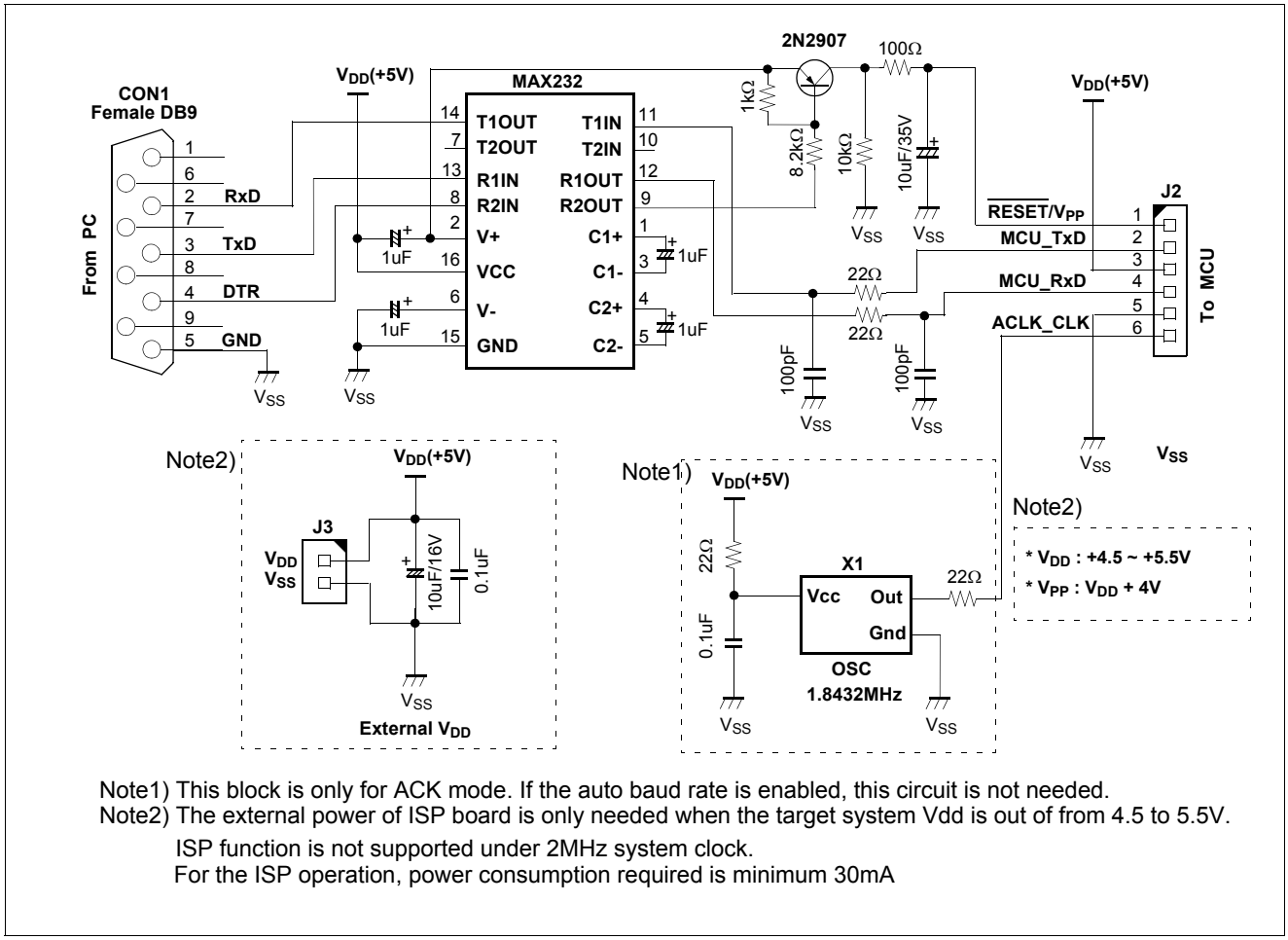


Figure 30-4 ISP Board



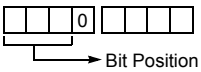
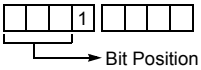
Note1) This block is only for ACK mode. If the auto baud rate is enabled, this circuit is not needed.
Note2) The external power of ISP board is only needed when the target system Vdd is out of from 4.5 to 5.5V.
ISP function is not supported under 2MHz system clock.
For the ISP operation, power consumption required is minimum 30mA

Figure 30-5 Reference ISP Circuit Diagram

APPENDIX

A. INSTRUCTION

A.1 Terminology List

Terminology	Description
A	Accumulator
X	X - register
Y	Y - register
PSW	Program Status Word
#imm	8-bit Immediate data
dp	Direct Page Offset Address
!abs	Absolute Address
[ ]	Indirect expression
{ }	Register Indirect expression
{ }+	Register Indirect expression, after that, Register auto-increment
.bit	Bit Position
A.bit	Bit Position of Accumulator
dp.bit	Bit Position of Direct Page Memory
M.bit	Bit Position of Memory Data (000_H~$0FFF_H$)
rel	Relative Addressing Data
upage	U-page ($0FF00_H$~$0FFFF_H$) Offset Address
n	Table CALL Number (0~15)
+	Addition
x	 Upper Nibble Expression in Opcode
y	 Upper Nibble Expression in Opcode
–	Subtraction
×	Multiplication
/	Division
( )	Contents Expression
∧	AND
∨	OR
⊕	Exclusive OR
~	NOT
←	Assignment / Transfer / Shift Left
→	Shift Right
↔	Exchange
=	Equal
≠	Not Equal

A.2 Instruction Map

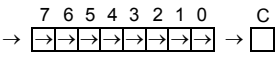
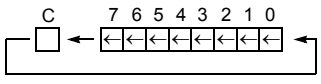
LOW HIGH	00000 00	00001 01	00010 02	00011 03	00100 04	00101 05	00110 06	00111 07	01000 08	01001 09	01010 0A	01011 0B	01100 0C	01101 0D	01110 0E	01111 0F
000	-	SET1 dp.bit	BBS A.bit,rel	BBS dp.bit,rel	ADC #imm	ADC dp	ADC dp+X	ADC !abs	ASL A	ASL dp	TCALL 0	SETA1 .bit	BIT dp	POP A	PUSH A	BRK
001	CLRC				SBC #imm	SBC dp	SBC dp+X	SBC !abs	ROL A	ROL dp	TCALL 2	CLRA1 .bit	COM dp	POP X	PUSH X	BRA rel
010	CLRG				CMP #imm	CMP dp	CMP dp+X	CMP !abs	LSR A	LSR dp	TCALL 4	NOT1 M.bit	TST dp	POP Y	PUSH Y	PCALL Upage
011	DI				OR #imm	OR dp	OR dp+X	OR !abs	ROR A	ROR dp	TCALL 6	OR1 OR1B	CMPX dp	POP PSW	PUSH PSW	RET
100	CLRV				AND #imm	AND dp	AND dp+X	AND !abs	INC A	INC dp	TCALL 8	AND1 AND1B	CMPY dp	CBNE dp+X	TXSP	INC X
101	SETC				EOR #imm	EOR dp	EOR dp+X	EOR !abs	DEC A	DEC dp	TCALL 10	EOR1 EOR1B	DBNE dp	XMA dp+X	TSPX	DEC X
110	SETG				LDA #imm	LDA dp	LDA dp+X	LDA !abs	TXA	LDY dp	TCALL 12	LDC LDCB	LDX dp	LDX dp+Y	XCN	DAS
111	EI				LDM dp,#imm	STA dp	STA dp+X	STA !abs	TAX	STY dp	TCALL 14	STC M.bit	STX dp	STX dp+Y	XAX	STOP

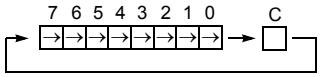
LOW HIGH	10000 10	10001 11	10010 12	10011 13	10100 14	10101 15	10110 16	10111 17	11000 18	11001 19	11010 1A	11011 1B	11100 1C	11101 1D	11110 1E	11111 1F
000	BPL rel	CLR1 dp.bit	BBC A.bit,rel	BBC dp.bit,rel	ADC {X}	ADC !abs+Y	ADC [dp+X]	ADC [dp]+Y	ASL !abs	ASL dp+X	TCALL 1	JMP !abs	BIT !abs	ADDW dp	LDX #imm	JMP [!abs]
001	BVC rel				SBC {X}	SBC !abs+Y	SBC [dp+X]	SBC [dp]+Y	ROL !abs	ROL dp+X	TCALL 3	CALL !abs	TEST !abs	SUBW dp	LDY #imm	JMP [dp]
010	BCC rel				CMP {X}	CMP !abs+Y	CMP [dp+X]	CMP [dp]+Y	LSR !abs	LSR dp+X	TCALL 5	MUL	TCLR1 !abs	CMPW dp	CMPX #imm	CALL [dp]
011	BNE rel				OR {X}	OR !abs+Y	OR [dp+X]	OR [dp]+Y	ROR !abs	ROR dp+X	TCALL 7	DBNE Y	CMPX !abs	LDYA dp	CMPY #imm	RETI
100	BMI rel				AND {X}	AND !abs+Y	AND [dp+X]	AND [dp]+Y	INC !abs	INC dp+X	TCALL 9	DIV	CMPY !abs	INCW dp	INC Y	TAY
101	BVS rel				EOR {X}	EOR !abs+Y	EOR [dp+X]	EOR [dp]+Y	DEC !abs	DEC dp+X	TCALL 11	XMA {X}	XMA dp	DECW dp	DEC Y	TYA
110	BCS rel				LDA {X}	LDA !abs+Y	LDA [dp+X]	LDA [dp]+Y	LDY !abs	LDY dp+X	TCALL 13	LDA {X}+	LDX !abs	STYA dp	XAY	DAA
111	BEQ rel				STA {X}	STA !abs+Y	STA [dp+X]	STA [dp]+Y	STY !abs	STY dp+X	TCALL 15	STA {X}+	STX !abs	CBNE dp	XYX	NOP

A.3 Instruction Set

Arithmetic / Logic Operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	ADC #imm	04	2	2	Add with carry.	
2	ADC dp	05	2	3	A ← (A) + (M) + C	
3	ADC dp + X	06	2	4		
4	ADC !abs	07	3	4		NV--H-ZC
5	ADC !abs + Y	15	3	5		
6	ADC [dp + X]	16	2	6		
7	ADC [dp] + Y	17	2	6		
8	ADC { X }	14	1	3		
9	AND #imm	84	2	2	Logical AND	
10	AND dp	85	2	3	A ← (A) ∧ (M)	
11	AND dp + X	86	2	4		
12	AND !abs	87	3	4		N-----Z-
13	AND !abs + Y	95	3	5		
14	AND [dp + X]	96	2	6		
15	AND [dp] + Y	97	2	6		
16	AND { X }	94	1	3		
17	ASL A	08	1	2	Arithmetic shift left	
18	ASL dp	09	2	4	C 7 6 5 4 3 2 1 0 ← “0”	N-----ZC
19	ASL dp + X	19	2	5		
20	ASL !abs	18	3	5		
21	CMP #imm	44	2	2		
22	CMP dp	45	2	3		
23	CMP dp + X	46	2	4		
24	CMP !abs	47	3	4	Compare accumulator contents with memory contents	N-----ZC
25	CMP !abs + Y	55	3	5	(A) - (M)	
26	CMP [dp + X]	56	2	6		
27	CMP [dp] + Y	57	2	6		
28	CMP { X }	54	1	3		
29	CMPX #imm	5E	2	2	Compare X contents with memory contents	
30	CMPX dp	6C	2	3	(X) - (M)	N-----ZC
31	CMPX !abs	7C	3	4		
32	CMPY #imm	7E	2	2	Compare Y contents with memory contents	
33	CMPY dp	8C	2	3	(Y) - (M)	N-----ZC
34	CMPY !abs	9C	3	4		
35	COM dp	2C	2	4	1’S Complement : (dp) ← ~(dp)	N-----Z-
36	DAA	DF	1	3	Decimal adjust for addition	N-----ZC
37	DAS	CF	1	3	Decimal adjust for subtraction	N-----ZC

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
38	DEC A	A8	1	2	Decrement	N-----Z-
39	DEC dp	A9	2	4	M ← (M) - 1	N-----Z-
40	DEC dp + X	B9	2	5		N-----Z-
41	DEC !abs	B8	3	5		N-----Z-
42	DEC X	AF	1	2		N-----Z-
43	DEC Y	BE	1	2		N-----Z-
44	DIV	9B	1	12	Divide : YA / X Q: A, R: Y	NV--H-Z-
45	EOR #imm	A4	2	2	Exclusive OR	
46	EOR dp	A5	2	3	A ← (A) ⊕ (M)	
47	EOR dp + X	A6	2	4		
48	EOR !abs	A7	3	4		N-----Z-
49	EOR !abs + Y	B5	3	5		
50	EOR [dp + X]	B6	2	6		
51	EOR [dp] + Y	B7	2	6		
52	EOR { X }	B4	1	3		
53	INC A	88	1	2	Increment	N-----ZC
54	INC dp	89	2	4	M ← (M) + 1	N-----Z-
55	INC dp + X	99	2	5		N-----Z-
56	INC !abs	98	3	5		N-----Z-
57	INC X	8F	1	2		N-----Z-
58	INC Y	9E	1	2		N-----Z-
59	LSR A	48	1	2	Logical shift right 	N-----ZC
60	LSR dp	49	2	4		
61	LSR dp + X	59	2	5		
62	LSR !abs	58	3	5		
63	MUL	5B	1	9	Multiply : YA ← Y × A	N-----Z-
64	OR #imm	64	2	2	Logical OR	
65	OR dp	65	2	3	A ← (A) ∨ (M)	
66	OR dp + X	66	2	4		
67	OR !abs	67	3	4		N-----Z-
68	OR !abs + Y	75	3	5		
69	OR [dp + X]	76	2	6		
70	OR [dp] + Y	77	2	6		
71	OR { X }	74	1	3		
72	ROL A	28	1	2	Rotate left through Carry 	N-----ZC
73	ROL dp	29	2	4		
74	ROL dp + X	39	2	5		
75	ROL !abs	38	3	5		

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
76	ROR A	68	1	2	Rotate right through Carry 	N-----ZC
77	ROR dp	69	2	4		
78	ROR dp + X	79	2	5		
79	ROR !abs	78	3	5		
80	SBC #imm	24	2	2	Subtract with Carry A ← (A) - (M) - ~(C)	NV--HZC
81	SBC dp	25	2	3		
82	SBC dp + X	26	2	4		
83	SBC !abs	27	3	4		
84	SBC !abs + Y	35	3	5		
85	SBC [dp + X]	36	2	6		
86	SBC [dp] + Y	37	2	6		
87	SBC { X }	34	1	3		
88	TST dp	4C	2	3	Test memory contents for negative or zero, (dp) - 00_H	N-----Z-
89	XCN	CE	1	5	Exchange nibbles within the accumulator A_7~A_4 ↔ A_3~A_0	N-----Z-

Register / Memory Operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	LDA #imm	C4	2	2	Load accumulator	N-----Z-
2	LDA dp	C5	2	3	A ← (M)	
3	LDA dp + X	C6	2	4		
4	LDA !abs	C7	3	4		
5	LDA !abs + Y	D5	3	5		
6	LDA [dp + X]	D6	2	6		
7	LDA [dp] + Y	D7	2	6		
8	LDA { X }	D4	1	3		
9	LDA { X }+	DB	1	4	X- register auto-increment : A ← (M) , X ← X + 1	
10	LDM dp,#imm	E4	3	5	Load memory with immediate data : (M) ← imm	--------
11	LDX #imm	1E	2	2	Load X-register	N-----Z-
12	LDX dp	CC	2	3	X ← (M)	
13	LDX dp + Y	CD	2	4		
14	LDX !abs	DC	3	4		
15	LDY #imm	3E	2	2	Load Y-register	N-----Z-
16	LDY dp	C9	2	3	Y ← (M)	
17	LDY dp + X	D9	2	4		
18	LDY !abs	D8	3	4		
19	STA dp	E5	2	4	Store accumulator contents in memory	--------
20	STA dp + X	E6	2	5	(M) ← A	
21	STA !abs	E7	3	5		
22	STA !abs + Y	F5	3	6		
23	STA [dp + X]	F6	2	7		
24	STA [dp] + Y	F7	2	7		
25	STA { X }	F4	1	4		
26	STA { X }+	FB	1	4	X- register auto-increment : (M) ← A, X ← X + 1	
27	STX dp	EC	2	4	Store X-register contents in memory	--------
28	STX dp + Y	ED	2	5	(M) ← X	
29	STX !abs	FC	3	5		
30	STY dp	E9	2	4	Store Y-register contents in memory	--------
31	STY dp + X	F9	2	5	(M) ← Y	
32	STY !abs	F8	3	5		
33	TAX	E8	1	2	Transfer accumulator contents to X-register : X ← A	N-----Z-
34	TAY	9F	1	2	Transfer accumulator contents to Y-register : Y ← A	N-----Z-
35	TSPX	AE	1	2	Transfer stack-pointer contents to X-register : X ← sp	N-----Z-
36	TXA	C8	1	2	Transfer X-register contents to accumulator: A ← X	N-----Z-
37	TXSP	8E	1	2	Transfer X-register contents to stack-pointer: sp ← X	N-----Z-
38	TYA	BF	1	2	Transfer Y-register contents to accumulator: A ← Y	N-----Z-

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
39	XAX	EE	1	4	Exchange X-register contents with accumulator :X ↔ A	--------
40	XAY	DE	1	4	Exchange Y-register contents with accumulator :Y ↔ A	--------
41	XMA dp	BC	2	5	Exchange memory contents with accumulator (M) ↔ A	N-----Z-
42	XMA dp+X	AD	2	6		
43	XMA {X}	BB	1	5		
44	XYX	FE	1	4	Exchange X-register contents with Y-register : X ↔ Y	--------

16-BIT operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	ADDW dp	1D	2	5	16-Bits add without Carry YA ← (YA) + (dp +1) (dp)	NV--H-ZC
2	CMPW dp	5D	2	4	Compare YA contents with memory pair contents : (YA) – (dp+1)(dp)	N-----ZC
3	DECW dp	BD	2	6	Decrement memory pair (dp+1)(dp) ← (dp+1) (dp) - 1	N-----Z-
4	INCW dp	9D	2	6	Increment memory pair (dp+1) (dp) ← (dp+1) (dp) + 1	N-----Z-
5	LDYA dp	7D	2	5	Load YA YA ← (dp +1) (dp)	N-----Z-
6	STYA dp	DD	2	5	Store YA (dp +1) (dp) ← YA	--------
7	SUBW dp	3D	2	5	16-Bits subtract without carry YA ← (YA) - (dp +1) (dp)	NV--H-ZC

Bit Manipulation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	AND1 M.bit	8B	3	4	Bit AND C-flag : C ← (C) ∧ (M .bit)	-------C
2	AND1B M.bit	8B	3	4	Bit AND C-flag and NOT : C ← (C) ∧ ~(M .bit)	-------C
3	BIT dp	0C	2	4	Bit test A with memory : Z ← (A) ∧ (M) , N ← (M_7) , V ← (M_6)	MM----Z-
4	BIT !abs	1C	3	5		
5	CLR1 dp.bit	y1	2	4	Clear bit : (M.bit) ← “0”	--------
6	CLRA1 A.bit	2B	2	2	Clear A bit : (A.bit) ← “0”	--------
7	CLRC	20	1	2	Clear C-flag : C ← “0”	-------0
8	CLRG	40	1	2	Clear G-flag : G ← “0”	--0-----
9	CLRV	80	1	2	Clear V-flag : V ← “0”	-0--0---
10	EOR1 M.bit	AB	3	5	Bit exclusive-OR C-flag : C ← (C) ⊕ (M .bit)	-------C
11	EOR1B M.bit	AB	3	5	Bit exclusive-OR C-flag and NOT : C ← (C) ⊕ ~(M .bit)	-------C
12	LDC M.bit	CB	3	4	Load C-flag : C ← (M .bit)	-------C

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
13	LDCB M.bit	CB	3	4	Load C-flag with NOT : C ← ~(M .bit)	-------C
14	NOT1 M.bit	4B	3	5	Bit complement : (M .bit) ← ~(M .bit)	--------
15	OR1 M.bit	6B	3	5	Bit OR C-flag : C ← (C) ∨ (M .bit)	-------C
16	OR1B M.bit	6B	3	5	Bit OR C-flag and NOT : C ← (C) ∨ ~(M .bit)	-------C
17	SET1 dp.bit	x1	2	4	Set bit : (M.bit) ← “1”	--------
18	SETA1 A.bit	0B	2	2	Set A bit : (A.bit) ← “1”	--------
19	SETC	A0	1	2	Set C-flag : C ← “1”	-------1
20	SETG	C0	1	2	Set G-flag : G ← “1”	--1-----
21	STC M.bit	EB	3	6	Store C-flag : (M .bit) ← C	--------
22	TCLR1 !abs	5C	3	6	Test and clear bits with A : A - (M) , (M) ← (M) ∧ ~(A)	N-----Z-
23	TSET1 !abs	3C	3	6	Test and set bits with A : A - (M) , (M) ← (M) ∨ (A)	N-----Z-

Branch / Jump Operation

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	BBC A.bit,rel	y2	2	4/6	Branch if bit clear :	--------
2	BBC dp.bit,rel	y3	3	5/7	if (bit) = 0 , then pc ← (pc) + rel	
3	BBS A.bit,rel	x2	2	4/6	Branch if bit set :	--------
4	BBS dp.bit,rel	x3	3	5/7	if (bit) = 1 , then pc ← (pc) + rel	
5	BCC rel	50	2	2/4	Branch if carry bit clear if (C) = 0 , then pc ← (pc) + rel	--------
6	BCS rel	D0	2	2/4	Branch if carry bit set if (C) = 1 , then pc ← (pc) + rel	--------
7	BEQ rel	F0	2	2/4	Branch if equal if (Z) = 1 , then pc ← (pc) + rel	--------
8	BMI rel	90	2	2/4	Branch if minus if (N) = 1 , then pc ← (pc) + rel	--------
9	BNE rel	70	2	2/4	Branch if not equal if (Z) = 0 , then pc ← (pc) + rel	--------
10	BPL rel	10	2	2/4	Branch if minus if (N) = 0 , then pc ← (pc) + rel	--------
11	BRA rel	2F	2	4	Branch always pc ← (pc) + rel	--------
12	BVC rel	30	2	2/4	Branch if overflow bit clear if (V) = 0 , then pc ← (pc) + rel	--------
13	BVS rel	B0	2	2/4	Branch if overflow bit set if (V) = 1 , then pc ← (pc) + rel	--------
14	CALL !abs	3B	3	8	Subroutine call	
15	CALL [dp]	5F	2	8	M(sp)←(pc_H), sp←sp - 1, M(sp)← (pc_L), sp ←sp - 1, if !abs, pc← abs ; if [dp], pc_L← (dp), pc_H← (dp+1) .	--------
16	CBNE dp,rel	FD	3	5/7	Compare and branch if not equal :	--------
17	CBNE dp+X,rel	8D	3	6/8	if (A) ≠ (M) , then pc ← (pc) + rel.	
18	DBNE dp,rel	AC	3	5/7	Decrement and branch if not equal :	--------
19	DBNE Y,rel	7B	2	4/6	if (M) ≠ 0 , then pc ← (pc) + rel.	
20	JMP !abs	1B	3	3	Unconditional jump	
21	JMP [!abs]	1F	3	5	pc ← jump address	--------
22	JMP [dp]	3F	2	4		
23	PCALL upage	4F	2	6	U-page call M(sp) ←(pc_H), sp ←sp - 1, M(sp) ← (pc_L), sp ← sp - 1, pc_L ← (upage), pc_H ← "$0FF_H$" .	--------
24	TCALL n	nA	1	8	Table call : (sp) ←(pc_H), sp ← sp - 1, M(sp) ← (pc_L),sp ← sp - 1, pc_L ← (Table vector L), pc_H ← (Table vector H)	--------

Control Operation & Etc.

No.	Mnemonic	Op Code	Byte No	Cycle No	Operation	Flag NVGBHIZC
1	BRK	0F	1	8	Software interrupt : B ← ”1”, M(sp) ← (pc_H), sp ←sp-1, M(s) ← (pc_L), sp ← sp - 1, M(sp) ← (PSW), sp ← sp - 1, pc_L ← ($0FFDE_H$) , pc_H ← ($0FFDF_H$) .	---1-0--
2	DI	60	1	3	Disable all interrupts : I ← “0”	-----0--
3	EI	E0	1	3	Enable all interrupt : I ← “1”	-----1--
4	NOP	FF	1	2	No operation	--------
5	POP A	0D	1	4	sp ← sp + 1, A ← M(sp)	--------
6	POP X	2D	1	4	sp ← sp + 1, X ← M(sp)	
7	POP Y	4D	1	4	sp ← sp + 1, Y ← M(sp)	
8	POP PSW	6D	1	4	sp ← sp + 1, PSW ← M(sp)	restored
9	PUSH A	0E	1	4	M(sp) ← A , sp ← sp - 1	--------
10	PUSH X	2E	1	4	M(sp) ← X , sp ← sp - 1	
11	PUSH Y	4E	1	4	M(sp) ← Y , sp ← sp - 1	
12	PUSH PSW	6E	1	4	M(sp) ← PSW , sp ← sp - 1	
13	RET	6F	1	5	Return from subroutine sp ← sp +1, pc_L ← M(sp), sp ← sp +1, pc_H ← M(sp)	--------
14	RETI	7F	1	6	Return from interrupt sp ← sp +1, PSW ← M(sp), sp ← sp + 1, pc_L ← M(sp), sp ← sp + 1, pc_H ← M(sp)	restored
15	STOP	EF	1	3	Stop mode (halt CPU, stop oscillator)	--------

B. MASK ORDER SHEET(MC80C7208)

MASK ORDER & VERIFICATION SHEET

MC80C7208-LC ☐☐☐ ☐

Q or L

Customer should write inside thick line box.

1. Customer Information

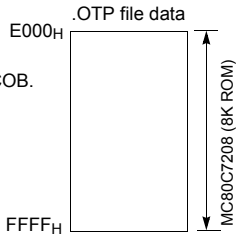
Company Name	
Application	
Order Date	YYYY MM DD • •
Tel: Fax:	
E-mail:	
Name & Signature:	

2. Device Information

Package	☐ 64MQFP	☐ 64LQFP
ROM Size	☐ 8K	☐ COB
OSC	MHz	KHz
RC_OSC	☐ X-tal	☐ Ext. RC/R
Unused ROM	☐ 00_H	☐ FF_H
Mask Data	File Name: (.OTP) Check Sum: ( )	

☐ Internet

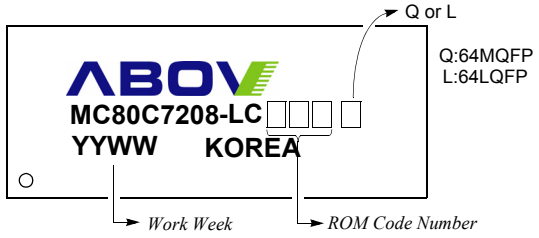
Note :
Contact sales part In case COB.



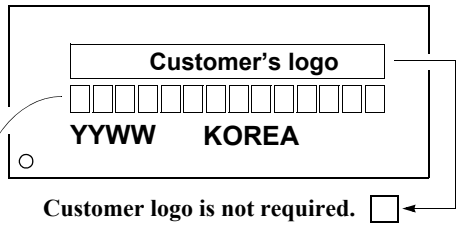
3. Marking Specification

(Please check mark into ☐)

☐



☐



Customer logo is not required. ☐

If the customer logo must be used in the special mark, please submit a clean original of the logo.

Customer's part number ☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐☐

4. Delivery Schedule

	Date	Quantity	ABOV Confirmation
Customer Sample	YYYY MM DD • •	pcs	
Risk Order	YYYY MM DD • •	pcs	

5. ROM Code Verification

Verification Date:	YYYY MM DD • •	Tel: Fax:
Check Sum:		
E-mail:	YYYY MM DD • •	Name & Signature:

ABOV semiconductor