



ATOM1.1 Family: Flash / Mask ROM 4-bit MCU with Reduced 8051 Architecture

**Rev. 1.1
January 2010**

**Copyright CORERIVER Semiconductor Co., Ltd. 2009
All Rights Reserved**

- ◆ *CORERIVER Semiconductor reserves the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time.*
- ◆ *CORERIVER shall give customers at least a three advance notice of intended discontinuation of a product or a service through its homepage.*
- ◆ *Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete.*
- ◆ *The CORERIVER Semiconductor products listed in this document are intended for usage in general electronics applications. These CORERIVER Semiconductor products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury.*

Table of Contents

1	PRODUCT OVERVIEW	6
1.1	ORDERING INFORMATION.....	6
1.1.1	<i>ATOM1.1 Family</i>	6
2	FEATURES	7
3	BLOCK DIAGRAM	9
4	PIN CONFIGURATIONS.....	11
5	PIN DESCRIPTION.....	12
6	FUNCTIONAL DESCRIPTION	13
6.1	CPU DESCRIPTION.....	13
6.1.1	<i>Memory Organization</i>	13
6.1.2	<i>Instruction Set Summary</i>	16
6.1.3	<i>CPU Timing</i>	20
6.2	PERIPHERAL DESCRIPTION	21
6.2.1	<i>I/O Ports</i>	21
6.2.2	<i>Carrier frequency generation</i>	26
6.2.3	<i>POR(Power-On Reset) and LVD(Low Voltage Detector)</i>	28
6.2.4	<i>WDT (Watchdog Timer)</i>	31
6.2.5	<i>Reset</i>	32
6.2.6	<i>Clock Circuit (On-chip oscillators)</i>	34
6.2.7	<i>Power Management</i>	36
6.2.8	<i>IAP (In Application Programming)</i>	36
7	ABSOLUTE MAXIMUM RATINGS.....	40
8	DC CHARACTERISTICS.....	41
9	AC CHARACTERISTICS.....	42
10	PACKAGE DIMENSION	43
11	PRODUCT NUMBERING SYSTEM.....	44
12	APPENDIX A: INSTRUCTION SET	45
13	APPENDIX B: SFR DESCRIPTION.....	81
13.1	P0: PORT 0 REGISTER.....	81

13.2	P4: PORT 4 REGISTER.....	81
13.3	DPL: THE LOW NIBBLE OF DATA POINTER.....	81
13.4	DPH: THE HIGH NIBBLE OF DATA POINTER.....	82
13.5	P1: PORT 1 REGISTER.....	82
13.6	REMC: REM OUTPUT CONTROL REGISTER.....	82
13.7	SPL: LOW NIBBLE OF STACK POINTER.....	82
13.8	SPH: HIGH NIBBLE OF STACK POINTER.....	83
13.9	P2: PORT 2 REGISTER.....	83
13.10	IAPCON: IAP CONTROL REGISTER.....	83
13.11	GDL: LOW NIBBLE OF GENERAL PURPOSE DATA REGISTER.....	84
13.12	GDH: HIGH NIBBLE OF GENERAL PURPOSE DATA REGISTER.....	84
13.13	P3: PORT 3 REGISTER.....	84
13.14	CKCFG: LVD CONFIGURATION REGISTER.....	84
13.15	IOCFG: I/O PORT CONFIGURATION REGISTER.....	85
14	APPENDIX C: HISTORY.....	86

List of Figures

Figure 3-1	Block Diagram of a 24-pin device.....	9
Figure 3-2	Block Diagram of a 20-pin device.....	10
Figure 4-1	Pin Configuration.....	11
Figure 6-1	Memory Organization.....	13
Figure 6-2	Instruction execution of the ATOM1.1 family.....	21
Figure 6-3	P0[3:0], P1[3:0], P3[3:0], P4[3:2].....	22
Figure 6-4	P2 port.....	23
Figure 6-5	XI/XO pins.....	24
Figure 6-6	I/O port mapping of 20-pin package.....	25
Figure 6-7	I/O port mapping of 24-pin package.....	25
Figure 6-8	ESD protection scheme I.....	26
Figure 6-9	ESD protection scheme II.....	26
Figure 6-10	IR LED driver control.....	27
Figure 6-11	REMI* waveform examples.....	28
Figure 6-12	Power-On Reset and Low Voltage Detector circuit.....	29
Figure 6-13	Power-On Reset Pulse Generation.....	29
Figure 6-14	Power-fail pulse generation.....	30

Figure 6-15 The requirements of V_{DD} notch.....	31
Figure 6-16 Reset Resources	32
Figure 6-17 Clock Circuit.....	35
Figure 6-18 Configuration for the Sytem Clock	35
Figure 6-19 Flash regions.....	37
Figure 10-1 SOIC (JEDEC) 20-pin Package Dimension.....	43
Figure 10-2 SOIC 24-pin Package Dimension	43
Figure 11-1 Product Numbering System.....	44

List of Tables

Table 1-1 ATOM1.1 Family – GC49C501G1 Series (Low Cost, Low Power Application MCU).....	6
Table 5-1 Pin Description for the 24/20 pin package	12
Table 6-1 Summary of SFRs.....	14
Table 6-2 Indirect Function Flag (IFF) Description.....	16
Table 6-3 Summary of Instruction Set.....	17
Table 6-4 Carrier frequency selection	27
Table 6-5 Time-out intervals of the Watchdog Timer	32
Table 6-6 System clock scaling.....	34
Table 6-7 IAP region and function.....	38
Table 6-8 Electrical characteristics of IAP.....	39
Table 7-1 Absolute Maximum Ratings	40
Table 7-2 Recommended Operating Conditions	40
Table 8-1 DC Characteristics	41
Table 9-1 AC Characteristics.....	42
Table 12-1 Abbreviations and symbols	45
Table 12-2 Opcode map.....	46
Table 13-1 System clock scaling.....	85

1 Product Overview

CORERIVER's ATOM1.1 family is a group of 4-bit microcontrollers with reduced 8051 architecture. Six clocks per one machine cycle are consumed in the redesigned processor core of the ATOM1.1 Family.

The ATOM1.1 family offers maximum 18 I/O ports, a Watchdog timer, and LVD (Low Voltage Detector) as peripherals.

The ATOM1.1 family provides power saving modes for the power-critical applications.

1.1 Ordering Information

1.1.1 ATOM1.1 Family

Table 1-1 ATOM1.1 Family – GC49C501G1 Series (Low Cost, Low Power Application MCU)

Product	Flash (byte)	EEPROM (byte)	RAM (Nibble)	Volt (V)	Freq (MHz)	WDT	REM Output	IR. LED Drive Tr.	I/O pin	Package	Others
GC49C501G1-SO24I	1K	(128)	64	1.8 ~ 5.5	10 (5)	1	1	Yes	18 (20)	24-SOIC	POR/LVD Ring OSC ISP/IAP
GC49C501G1-SJ20I	1K	(128)	64	1.8 ~ 5.5	10 (5)	1	1	Yes	14 (16)	20-SOIC (JEDEC)	POR/LVD Ring OSC ISP/IAP

Table 1-2 ATOM1.1 Family – GC41C501G1 Series (Low Cost, Low Power Application MCU)

Product	Mask ROM (byte)	EEPROM (byte)	RAM (Nibble)	Volt (V)	Freq (MHz)	WDT	REM Output	IR. LED Drive Tr.	I/O pin	Package	Others
GC41C501G0-SO24I	1K	-	64	1.8 ~ 5.5	10 (5)	1	1	Yes	18 (20)	24-SOIC	POR/LVD Ring OSC
GC41C501G0-SJ20I	1K	-	64	1.8 ~ 5.5	10 (5)	1	1	Yes	14 (16)	20-SOIC (JEDEC)	POR/LVD Ring OSC

- 128 bytes of program memory (Flash) can be used as EEPROM, the contents of which can be modified by using IAP functions during SW operation.
- Max. operating frequency of ATOM1.1 family is 5 MHz when VDD is less than 2.7 V.

2 Features

- ◆ CPU
 - ✓ 4-bit reduced 8051 architecture
 - ✓ Continuous addressing, not paged program memory.
 - ✓ 51 instructions including push, pop and logic instructions.
 - ✓ Instruction cycle : $F_{SYS}/6$
 - ✓ Multi-level subroutine nesting with RAM based stack.
 - ◆ On-chip Memories
 - ✓ FLASH : 1024 bytes (including 128 EEPROM)
 - ✓ RAM : 64 nibbles (including stack)
 - ◆ ISP (In System Programming) of FLASH
 - ◆ IAP (In Application Programming) of FLASH
 - ◆ I/O Ports
 - ✓ P0 : 4-bit parallel I/O (Open drain output)
 - ✓ P1 : Parallel I/O (Open drain output), 4-bit for 24-pin, 2-bit for 20-pin.
 - ✓ P2, P3 : 4-bit parallel/bit-selectable I/O (Open drain output)
 - ✓ P4 : 2-bit Parallel I/O (Open drain output) for 24-pin packages.
 - ◆ REM output (Remote control transmitter)
 - ✓ Built-in Transistor for I.R. LED Drive
 - ✓ $I_{OL} = 300 \text{ mA (Max.)}$ at $V_{DD} = 3V$ and $V_O = 0.4V$
 - ◆ Carrier Pulse Generation : 7 types
 - ◆ Built-in Oscillator
 - ✓ Crystal/Ceramic resonator
 - ✓ Internal oscillator: 8MHz.
 - ◆ Built-in Reset
 - ✓ Power-on Reset, Power-fail Reset
 - ✓ WDT (Watch-Dog Timer) Reset
 - ✓ Clock switching reset
 - ◆ Power Management
 - ✓ Power-down (stop) mode
 - ✓ Release stop by input changes
 - ✓ Sleep mode
 - ◆ Power Consumption
-

- ✓ Stop mode: < 0.1uA (Typ.) at 2.0V, 1 uA (Max.) at 5.0V
- ✓ Normal mode : 400 uA (Typ.) at 2.0V, $F_{SYS} = 4$ MHz
- ◆ Operating frequency vs. voltage
 - ✓ Max. $F_{OSC} = 10$ MHz ($2.7\text{ V} \leq V_{DD} \leq 5.5\text{V}$)
 - ✓ Max. $F_{OSC} = 5$ MHz ($1.8\text{ V} \leq V_{DD} < 2.7\text{V}$)
- ◆ Operating temperature : -40 °C ~ 85 °C
- ◆ ESD protection up to 2,000V
- ◆ Latch-up protection up to $\pm 200\text{mA}$
- ◆ Package
 - ✓ 24-pin SOIC
 - ✓ 20-pin SOIC

3 Block Diagram

Figure 3-1 and Figure 3-2 show the block diagrams of ATOM1.1 family. The CPU fetches instructions from the program memory (Flash) and executes them. Data are read from or written to data memory (RAM) or integrated peripherals via special function registers (SFRs). The arithmetic and logic unit (ALU) inside the CPU processes data.

Special function registers are used for data processing as well as data reading and writing. Except for the program counter (PC), these registers are located at the special function register space from addresses 0H to FH. All special function registers have 4-bit data length. The internal data memory size of ATOM1.1 is 64 nibbles. The program counter resides inside the CPU.

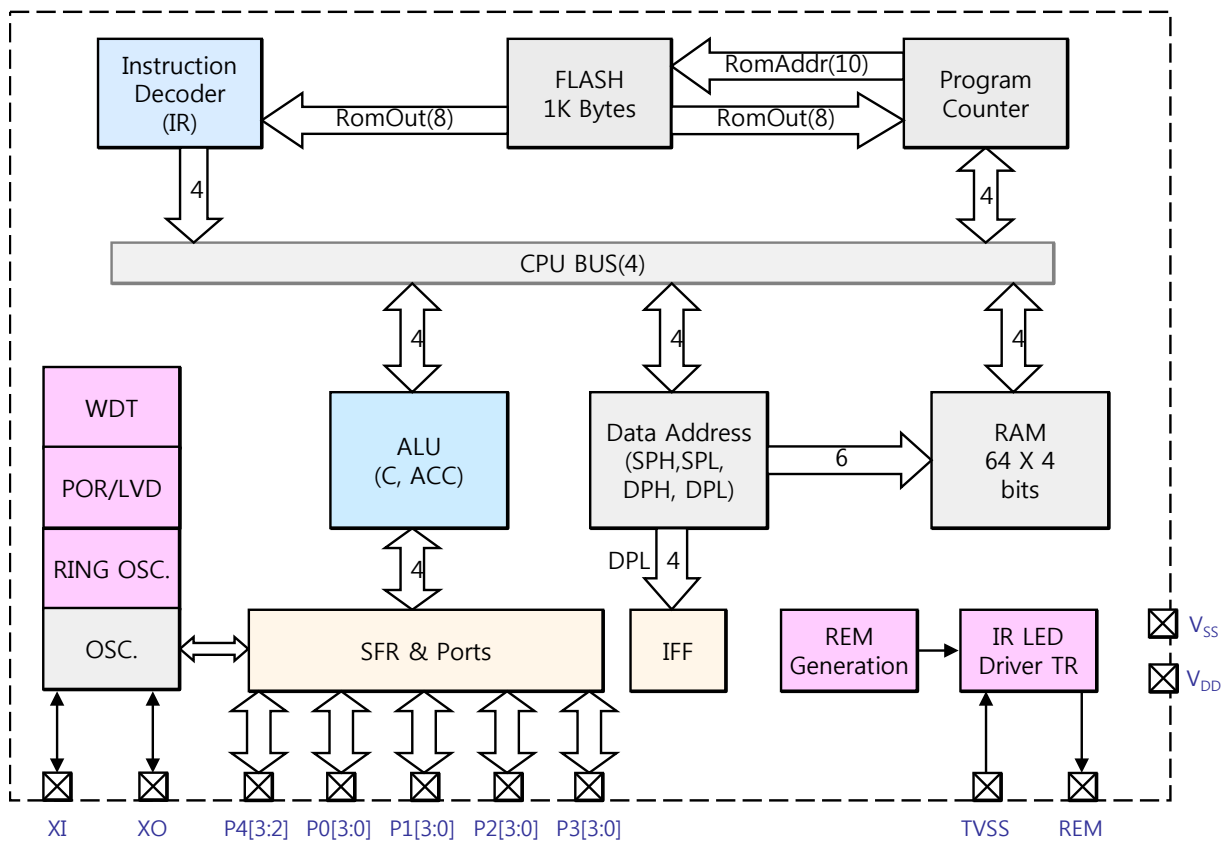


Figure 3-1 Block Diagram of a 24-pin device

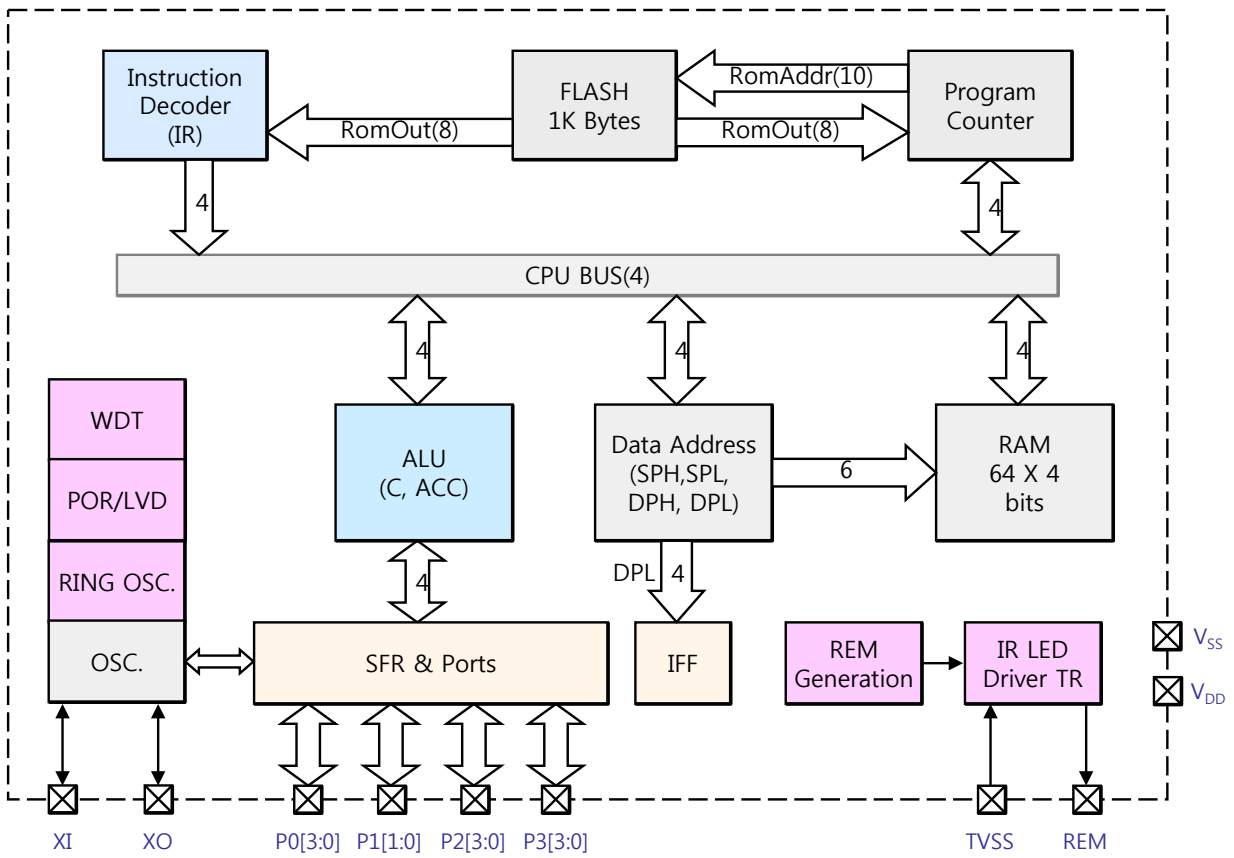
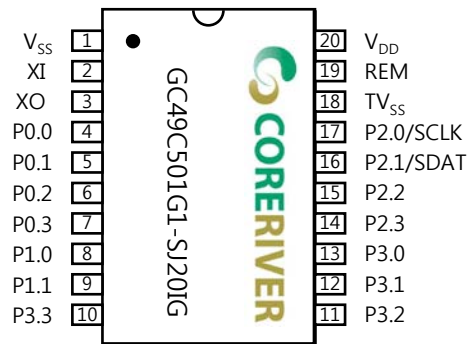


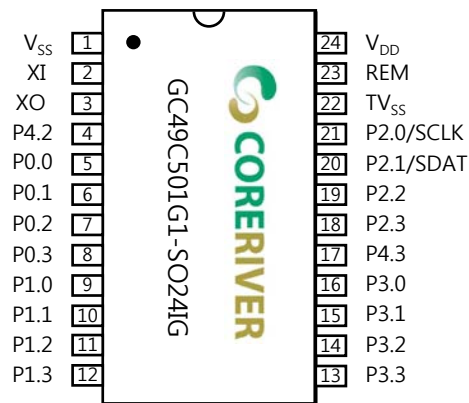
Figure 3-2 Block Diagram of a 20-pin device

4 Pin Configurations

The ATOM1.1 family supports various packages, e.g. 24-SOIC, 20-SOIC. The pin configurations are shown in Figure 4-1.



[20-SOIC]



[24-SOIC]

Figure 4-1 Pin Configuration

5 Pin Description

Table 5-1 Pin Description for the 24/20 pin package

Symbol	Type	Description	Remark
V_{DD}	Power	Power supply	-
V_{SS}	Power	Ground	-
REM	Output	Output for IR LED drive Transistor. The transistor is n-channel device.	-
TV_{SS}	Power	Ground for IR LED drive Transistor	
XI	Input	Input to the inverting oscillator amplifier.	-
XO	Output	Output from the inverting oscillator amplifier.	-
P4[3:2]	In/Out	Parallel Input/Output Port (Only for 24-pin package). Each bit can be individually set or cleared. Schmitt Trigger input and open-drain output with an internal pull-up TR.	-
P0[3:0]	In/Out	Parallel Input/Output Port. Schmitt Trigger input and open-drain output with an internal pull-up TR. The stop mode is terminated by low input applied to any pin of this port.	-
P1[1:0]	In/Out	Parallel Input/Output Port. Schmitt Trigger input and open-drain output with an internal pull-up TR. The stop mode is terminated by low input applied to any pin of this port.	-
P1[3:2]	In/Out	Parallel Input/Output Port (Only for 24-pin package). Schmitt Trigger input and open-drain output with an internal pull-up TR. The stop mode is terminated by low input applied to any pin of this port.	-
P2[3:0]	In/Out	Parallel Input/Output Port. Each bit can be individually set or cleared. Schmitt Trigger input and open-drain output with an internal pull-up TR. This port can be configured as a push-pull output port. P2[0] and P2[1] are also used for ISP of Flash memory	-
P3[3:0]	In/Out	Parallel Input/Output Port. Each bit can be individually set or cleared. Schmitt Trigger input and open-drain output with an internal pull-up TR.	

6 Functional Description

6.1 CPU Description

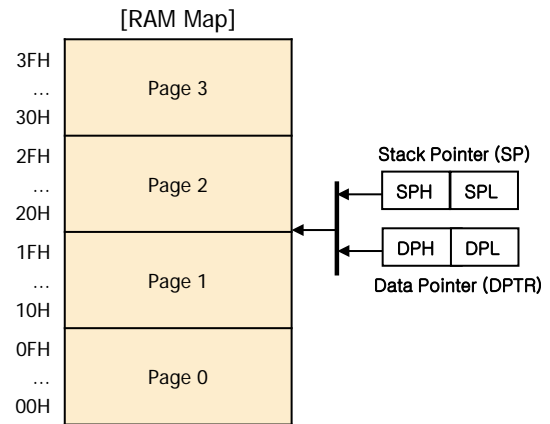
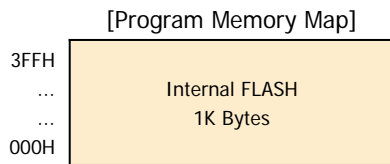
6.1.1 Memory Organization

ATOM1.1 family has separate address spaces for program and data memory. The logical separation of program and data memory allows the data memory to be accessed by 6-bit addresses.

Program Memory can only be read, not written to. There can be up to 1K bytes of Program Memory.

◆ Address Space

- ✓ Program memory : 1K Bytes.
Continuously addressed by Byte.
- ✓ Indirect data memory : 64 Nibbles.
Bit accessible.
- ✓ Special function registers : 16 Registers.
Directly addressed.
- ✓ Indirect function flags : 16 bits.
Bit position is selected by DPL.



[Special Function Register Map]

0CH	P3	CKCFG	IOCFG	-
08H	P2	IAPCON	GDL	GDH
04H	P1	REMC	SPL	SPH
00H	P0	P4	DPL	DPH

[Indirect Function Flag Map]

15	14	13	12	11	10	9	8
STOP	SLEEP	WDTE	WDTR	MAP1	MAP0	P4.2	P4.3
7	6	5	4	3	2	1	0
P3.3	P3.2	P3.1	P3.0	P2.3	P2.2	P2.1	P2.0

Figure 6-1 Memory Organization

6.1.1.1 Program Memory

As shown in the Figure 6-1, the Program Memory of ATOM1.1 consists of a 1 KByte Flash ROM. ROM addressing is supported by the 10-bit program counter. After reset, the CPU executes the instructions from location 000H.

6.1.1.2 Special Function Registers

ATOM1.1 has 15 SFRs (special function registers). They are located from address 00h to 0Eh. They are addressed by only direct addressing.

Table 6-1 Summary of SFRs

Symbol	Address	Description	Power-On Reset Value	Other Reset Value
P0	00h	Port 0 output register	1111	1111
P4	01h	Port 4 output register	1111	1111
DPL	02h	The low nibble of data pointer (DPTR)	0000	0000
DPH	03h	The high nibble of data pointer (DPTR)	--00	--00
P1	04h	Port 1 output register	1111	1111
REMC	05h	REM output control register	0000	0000
SPL	06h	The low nibble of stack pointer (SP)	1111	1111
SPH	07h	The high nibble of stack pointer (SP)	--01	--01
P2	08h	Port 2 output register	1111	1111
IAPCON	09h	IAP control register. This is accessible only if MAP1 is set and MAPO is cleared.	0000	0000
GDL	0Ah	The low nibble of general purpose data register	0000	0000
GDH	0Bh	The high nibble of general purpose data register	0000	0000
P3	0Ch	Port 3 output register	1111	1111
CKCFG	0Dh	The clock configuration register. Initialized only by power-on-reset	0000	uuuu
IOCFG	0Eh	The I/O port configuration register. Initialized only by power-on-reset	0000	uu0u
-	0Fh	Reserved	----	----

-: Unimplemented bit. Read value is 0.

u: Unchanged.

6.1.1.3 Data Memory

The internal data memory of ATOM1.1 consists of a 64-nibble RAM which is organized into 4 pages of 16 nibble each. RAM addressing is implemented by the 6-bit data pointer register (DPH, DPL). Page number is specified by the 2-bit data pointer high (DPH) register. Each page includes a 16-nibble memory space. Each nibble in a page is addressed by the 4-bit data pointer low (DPL) register. The internal data memory can be addressed by only indirect addressing.

Several instructions (CLR bit / JNB bit, rel / SETB bit / JB bit, rel) access each bit of the internal data memory. Similarly byte addressing is implemented by the 6-bit data pointer register. In addition, the opcode of the instructions includes the bit address in a byte.

6.1.1.4 Indirect Function flag (IFF)

ATOM1.1 has 16 indirect function flags. They are accessible by the indirect addressing with the contents of the Data Pointer Low nibble (DPL). In an assembly program, DPL is denoted by 'L'. No instruction can read them.

In order to modify the value of an indirect flag, the following steps must be taken.

1. At first, write the address of the target indirect function flag to DPL by using the assembly code 'MOV L, #n'.
2. If you want to set the target flag, use the assemble code 'SETB @L'. You can clear the flag by using the assembly code 'CLR @L'.

The individual set/clear of pins is available only in case that the corresponding parallel pin is supported for the used package.

Ex 1) For setting the target flag

```
MOV L, #n
SETB @L
```

Ex 2) For clearing the target flag

MOV L, #n

CLR @L

Table 6-2 Indirect Function Flag (IFF) Description

Flag	Address (DPL)	Description	Reset Value
STOP	15	Enter stop mode. Not set until all pins of P0 and P1 are high.	0
SLEEP	14	Enter sleep mode. Released by WDT reset.	0
WDTE	13	Enable flag of WDT. If this flag is cleared, WDT stops running and holds the state. This flag can be modified if and only if MAP1 bit set and MAP0 bit cleared. This flag is also set by H/W when user sets SLEEP flag or writes IAPCON SFR.	1
WDTR	12	Reset WatchDog Timer. Set by S/W. Cleared by H/W after WDT is reset.	0
MAP1	11	Address map extension bit 1 for SFR/IFF	0
MAP0	10	Address map extension bit 0 for SFR/IFF. Don't set this flag for the future compatibility	0
P4.2	9	Individual bit set/clear for P4	1
P4.3	8	Individual bit set/clear for P4	1
P3.3	7	Individual bit set/clear for P3	1
P3.2	6	Individual bit set/clear for P3	1
P3.1	5	Individual bit set/clear for P3	1
P3.0	4	Individual bit set/clear for P3	1
P2.3	3	Individual bit set/clear for P2	1
P2.2	2	Individual bit set/clear for P2	1
P2.1	1	Individual bit set/clear for P2	1
P2.0	0	Individual bit set/clear for P2	1

6.1.2 Instruction Set Summary

The ATOM1.1 family has a reduced 8051 architecture and different timing to those of the standard 8051. This means the relative duration of the individual instructions, i.e. the number of clock cycles per

instructions, is different to that of 8051.

Refer to Appendix A for more details about instruction set.

Table 6-3 Summary of Instruction Set

Type	Instruction	Description
Arithmetic	ADD A #data	Add data to ACC.
	INC A	Increment ACC.
	DEC A	Decrement ACC.
	ADD A, @DP	Add the indirect data memory nibble to ACC
	ADDC A, @DP	Add the indirect data memory nibble to ACC with the Carry represented by C
	SUB A, @DP	Subtract the indirect data memory nibble from ACC
	INC @DP	Increment the indirect data memory nibble.
	DEC @DP	Decrement the indirect data memory nibble.
Logical	CLR A	Clear ACC
	CPL A	Complement ACC
	RRC A	Rotate right ACC with Carry flag
	ANL A @DP	Logical AND for ACC and the indirect data memory nibble.
	ORL A, @DP	Logical OR for ACC and the indirect data memory nibble.
	XRL A, @DP	Logical Exclusive-OR for ACC and the indirect data memory nibble.
Data Transfer	MOV dir, A	Move ACC to the special function register
	MOV A, dir	Move the special function register to ACC
	MOV A, @DP	Move the indirect data memory nibble to ACC
	MOV A, #data	Move data to ACC
	MOV L, @DP	Move the indirect data memory nibble to DPL
	MOV @DP, A	Move ACC to the indirect data memory nibble.
	MOVI @DP, A	Move ACC to the indirect data memory nibble and increment the data pointer
	MOVD @DP, A	Move ACC to the indirect data memory nibble and decrement the data pointer
	XCH A, @DP	Exchange ACC and the indirect data memory nibble.
	MOVI @DP, #data	Move data to the indirect data memory nibble and increment the data pointer

	MOV L, #data	Move data to DPL
	MOV H, #data	Move data to DPH
	PUSH A	Push ACC to stack
	POP A	Pop stack to ACC
Branch	CJNE @DP, #data, rel	Jump if the indirectl data memory nibble is not equal to the data
	CJNE L, #data, rel	Jump if DPL is not equal to the data
	CJNE A, dir, rel	Jump if ACC is not equal to the special function register
	CJNE A, @DP, rel	Jump if ACC is not equal to the indirectl data memory nibble.
	CJLE A, @DP, rel	Jump if ACC is less than or equal to the indirectl data memory nibble.
	CJNE A, #data, rel	Jump if ACC is not equal to the data
	DJNZ A, rel	Decrement ACC. Jump if the result is not zero.
	JB bit, rel	Jump if the indirectl data memory nibble bit is 1.
	JNB bit, rel	Jump if the indirectl data memory nibble bit is not 1.
	JC rel	Jump if C is 1.
	JNC rel	Jump if C is not 1.
	JMP addr	Jump to given address.
	CALL addr	Call subroutine.
	RET	Return from subroutine.
NOP	No Operation.	
Bit & Misc.	SETB @L	Set the indirect function flag.
	CLR @L	Clear the indirect function flag.
	SETB bit	Set the indirect memory data bit.
	CLR bit	Clear the indirect memory data bit.
	SETB C	Set Carry flag
	CLR C	Clear Carry flag
	INC DPTR	Increment the data pointer
	DEC DPTR	Decrement the data pointer

6.1.2.1 Addressing Modes

The operands used in the instructions can be addressed in different modes. The ATOM1.1 family uses four different addressing modes to this end:

- Immediate
- Direct
- Indirect
- Register-specific

6.1.2.1.1 Immediate Addressing Mode

Immediate addressing mode means that constants can be loaded to registers. The constant is part of the instruction in program memory.

Examples:

MOV A, #Fh	ACC is loaded with the constant value Fh
ADD A, #4h	Add the constant value 4h to ACC.
MOV L, #3h	Load data pointer low nibble with the constant value 3h

6.1.2.1.2 Direct Addressing Mode

In direct addressing mode, the operand is specified by a 4-bit address field, which is part of the instruction.

Note: Special function registers can be addressed in this mode.

Examples:

MOV A, dir	dir is the 4-bit address of the SFR.
MOV dir, A	dir is the 4-bit address of the SFR.

6.1.2.1.3 Indirect Addressing Mode

In indirect addressing mode, the operand is specified by a 4-bit or 6-bit address that is stored in a register. For 4-bit addresses, the content of DPL (the Data Pointer Low Nibble) is used. For 6-bit addresses, 6-bit wide data pointer (DPH and DPL) can be used.

Note: The internal RAM can be addressed by using 6-bit addresses. No external data memory can be used by ATOM1.1 applications.

Examples:

MOV A, @DP	RAM is addressed by contents of DP (6-bit address)
SETB @L	The indirect function flag is addressed by contents of the data pointer low nibble DPL (4-bit address)

6.1.2.1.4 Register-Specific Addressing Mode

Some instructions are specific to a certain register. For example, some instructions always operate on the accumulator, or data pointer, etc., so no address byte is needed to point to it.

6.1.3 CPU Timing

The CPU timing for the ATOM1.1 family is an important aspect, especially for those who wish to use software instructions to generate timing delays. Also, it provides the user with an insight into the timing differences between the ATOM1.1 family and the standard 80C52 as shown in Figure 6-2. In the ATOM1.1 family, each machine cycle is six clock periods long. Each clock period is designated a state. Thus each machine cycle is made up of six states, S1, S2, S3, S4, S5, and S6 in that order. The state of SFR, I/O ports, and IFF flags is updated at the end of an instruction (S6). To reduce the instruction execution time, both the rising and the falling clock edges are used for internal timing. Hence it is important that the duty cycle of the clock be as close to 50% as possible to avoid timing conflicts. All instructions except branch instructions are completed in one machine cycle. All branch instructions consume 2 machine cycles whether the branch is taken or not. So ATOM1.1 consumes the approximately same number of machine cycles with the instructions to execute.

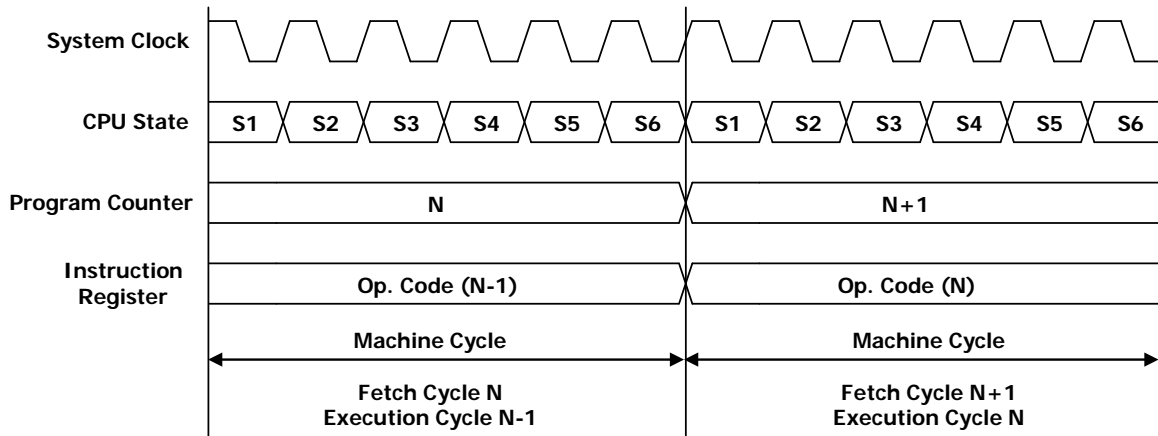


Figure 6-2 Instruction execution of the ATOM1.1 family

6.2 Peripheral Description

6.2.1 I/O Ports

The ATOM1.1 family provides bi-directional I/O ports. Each I/O port consists of a latch (Special Function Register P0 through P4), an output driver, and an input buffer. All the latches are 4-bits I/O ports. P2 and P3 are nibble-addressable and bit-addressable. A bit instruction has a different opcode from a byte instruction. All ports are initialized asynchronously by all kinds of resets. By the initialization, they are configured as input and their pull-ups are turned on. Only P2 can support push-pull output. In the stop mode and the sleep mode, the state of all ports is not changed. The ATOM1.1 family reads a port but writes the port register.

6.2.1.1 PORT 0

Port 0 is a 4-bit, open-drain, bi-directional I/O port with internal pull-ups. The pull-ups are switched on/off by the value of the P0 register. When a bit of the P0 register has 0, the corresponding pull-up is switched off. Writing 1 to the bit switches the pull-up on.

Port 0 pins that have 1s written to them are pulled high weakly by the internal pull-ups and can be used as inputs.

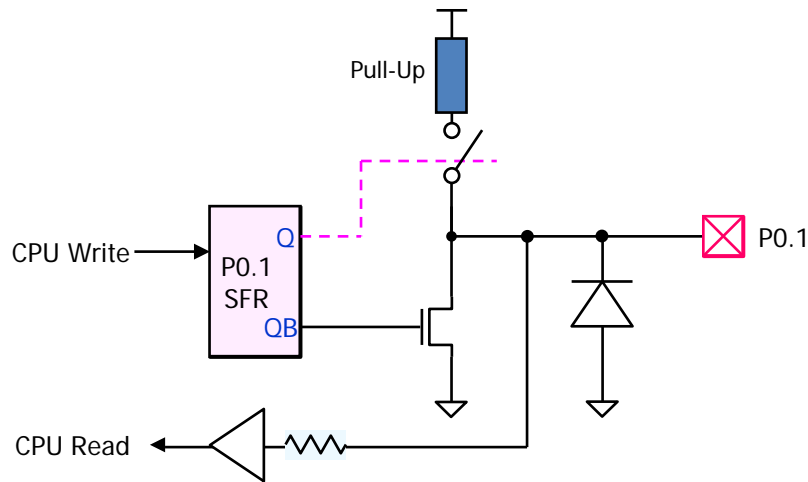


Figure 6-3 P0[3:0], P1[3:0], P3[3:0], P4[3:2]

6.2.1.2 PORT 1

Port 1 is a 4-bit, open-drain, bi-directional I/O port with internal pull-ups. The pull-ups are switched on/off by the value of the P1 register. When a bit of the P1 register has 0, the corresponding pull-up is switched off. Writing 1 to the bit switches the pull-up on. Port 1 pins that have 1s written to them are pulled high weakly by the internal pull-ups and can be used as inputs.

Pins 2 and 3 of PORT 1 are available for only 24-pin package. When reading these pins of 20-pin device, a user will obtain their default value of P1[3:2]. Please maintain their default value of P1[3:2] for 20-pin device.

6.2.1.3 PORT 2

Port 2 is a 4-bit, open-drain, bi-directional I/O port with internal pull-ups. The pull-ups are switched on/off by the value of the P2 register. When a bit of the P2 register has 0, the corresponding pull-up is switched off. Writing 1 to the bit switches the pull-up on.

Port 2 pins that have 1s written to them are pulled high weakly by the internal pull-ups and can be used as inputs. Port 2 is bit-addressible. Port 2 supports push-pull output if the P2OEN bit in the IOCFG register is set.

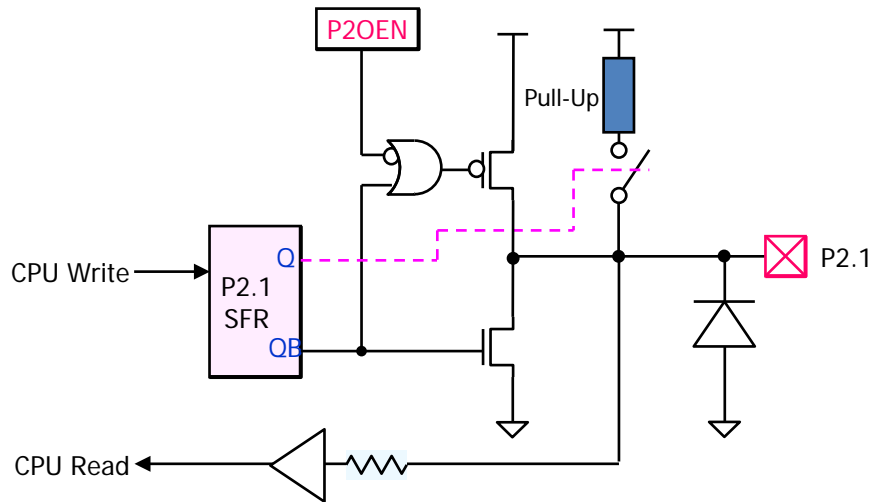


Figure 6-4 P2 port

6.2.1.4 PORT 3

Port 3 is a 4-bit, open-drain, bi-directional I/O port with internal pull-ups. The pull-ups are switched on/off by the value of the P3 register. When a bit of the P3 register has 0, the corresponding pull-up is switched off. Writing 1 to the bit switches the pull-up on.

Port 3 pins that have 1s written to them are pulled high weakly by the internal pull-ups and can be used as inputs. PORT3 is bit-addressible.

6.2.1.5 PORT4

PORT4[3:2] is a bi-directional I/O pins with internal pull-ups. The pull-ups are switched on/off by the value of the P4 register. When a bit of the P4 register has 0, the corresponding pull-up is switched off. Writing 1 to the bit switches the pull-up on.

PORT4[3:2] pins that have 1s written to them are pulled high weakly by the internal pull-ups and can be used as inputs. These pins are bit-addressible. They are available only for 24-pin package. When reading these pins of 20-pin device, a user will obtain their default value of P4[3:2]. Please maintain their default value of P4[3:2] for 20-pin device.

6.2.1.6 I/O port mapping

- **IOCFG:** I/O port Configuration Register

Address = 0EH		Reset Value = 0000B		
Bit	3	2	1	0
	IOMAP1	IOMAP0	P2OEN	-
	R/W(0)	R/W(0)	R/W(0)	-

P2OEN: Configure P2 as a push-pull output port.

IOMAP[1:0]: Selection of port mapping options.

IOMAP1	IOMAP0	Port mapping
0	0	Default
0	1	A port mapping option for 20-pin package
1	0	A port mapping option for 24-pin package
1	1	Reserved

This register is initialized by only power-on-reset. However, the P2OEN bit is cleared by all kinds of resets.

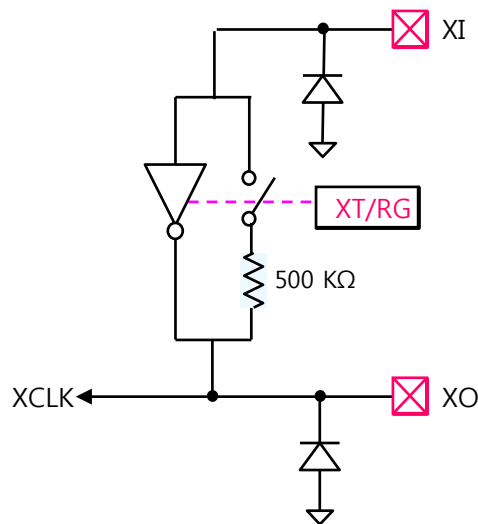


Figure 6-5 XI/XO pins

A user can select I/O port mapping options by changing the value of the IOCFG register. However, even if another I/O port mapping option is selected, the function of each pin is not changed. These mapping options are useful for maintaining pin-to-pin compatibility with existing devices.

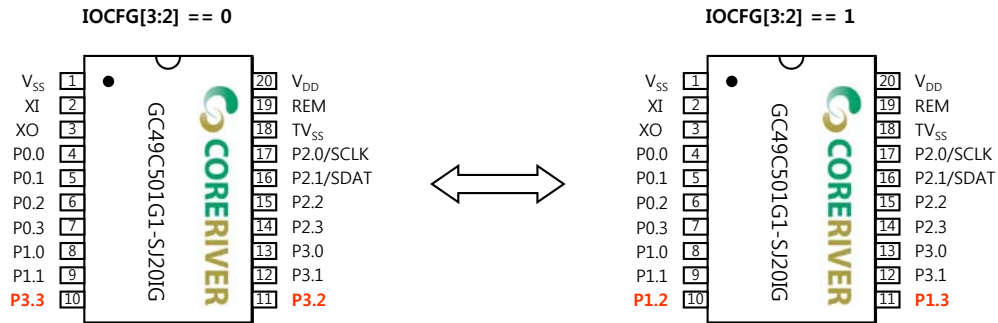


Figure 6-6 I/O port mapping of 20-pin package

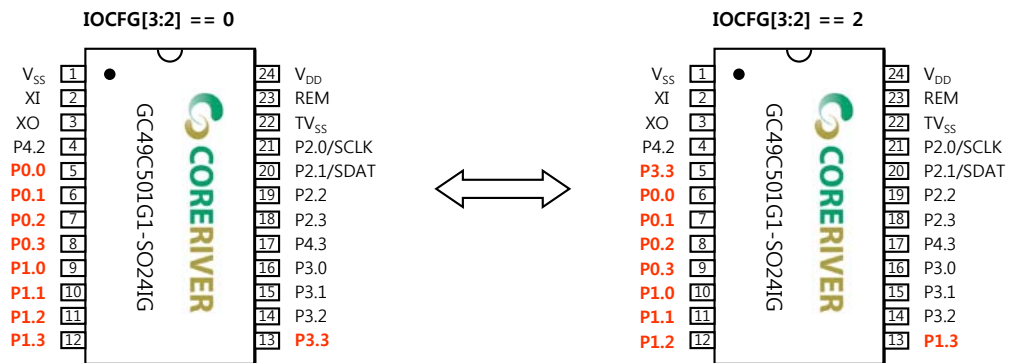


Figure 6-7 I/O port mapping of 24-pin package

6.2.1.7 ESD protection of I/O ports

As electrostatic discharge (ESD) problems become more common in electronic circuits, various devices have been used to protect circuits from ESD. All pins of the ATOM1.1 family excluding the V_{DD}, V_{SS} and TV_{SS} pins, use two diodes and one resistor for ESD protection. The ESD protection scheme is shown in Figure 6-8.

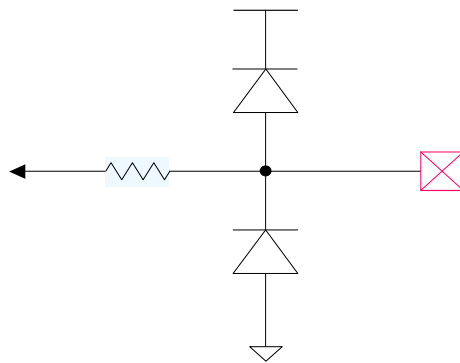


Figure 6-8 ESD protection scheme I

As voltage-clamping devices, the two diodes and a resistor limit the surge voltage to a safe level for the circuit being protected.

We protect the V_{DD} pin from ESD with one diode and one resistor as shown in Figure 6-9

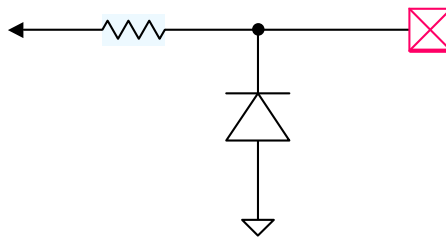


Figure 6-9 ESD protection scheme II

Similarly, the diode and the resistor limit the surge voltage to a safe level for the circuit being protected.

The V_{SS} and TV_{SS} pins use no diode for ESD protection.

6.2.2 Carrier frequency generation

The ATOM1.1 family can drive an IR LED for remote controller application. It supports 7 carrier frequencies for data transmission. The REMC register is used for control of the IR LED driver and the carrier frequency.

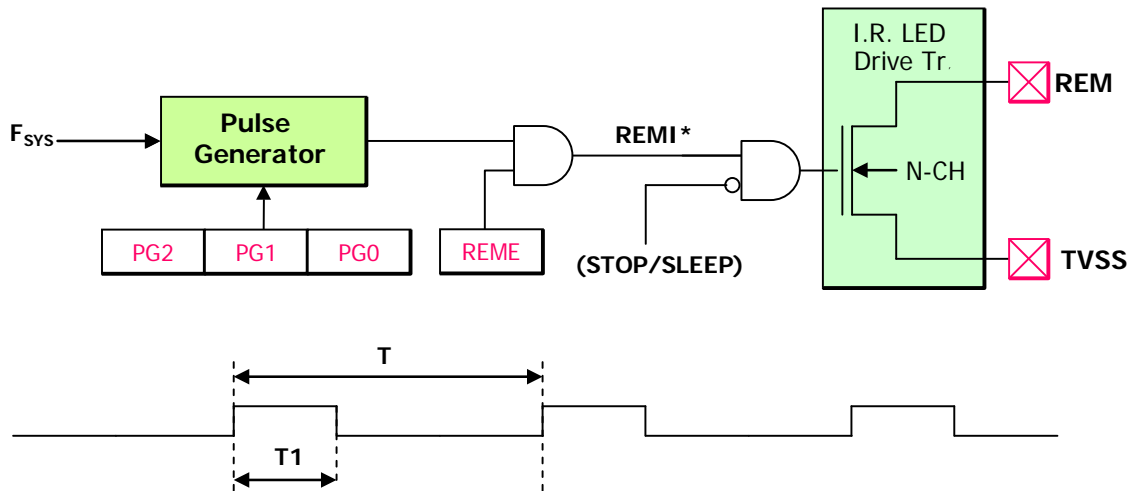


Figure 6-10 IR LED driver control

The IR LED driver is a n-channel MOS transistor. The REM output is the inverse of the REMI* signal. The IR LED is turned on when REMI* is high.

Table 6-4 Carrier frequency selection

REME	PG2	PG1	PG0	Transmission Control (REMI)
0	X	X	X	0 (Disable)
1	0	0	0	$1/T = F_{SYS}/12, T1/T = 1/3$
1	0	0	1	$1/T = F_{SYS}/8, T1/T = 1/2$
1	0	1	0	$1/T = F_{SYS}/12, T1/T = 1/4$
1	0	1	1	1 (No carrier)
1	1	0	0	$1/T = F_{SYS}/12, T1/T = 1/2$
1	1	0	1	$1/T = F_{SYS}/8, T1/T = 1/4$
1	1	1	0	$1/T = F_{SYS}/11, T1/T = 4/11$
1	1	1	1	1 (No carrier)

- **REMC(05h):** The REM Output Control Register

Address = 05H	Reset Value = 0000B				
Bit	3 2 1 0				
	<table border="1" style="margin-left: auto; margin-right: auto;"> <tr> <td style="padding: 2px 10px;">REME</td> <td style="padding: 2px 10px;">PG2</td> <td style="padding: 2px 10px;">PG1</td> <td style="padding: 2px 10px;">PG0</td> </tr> </table>	REME	PG2	PG1	PG0
REME	PG2	PG1	PG0		
	<table style="margin-left: auto; margin-right: auto;"> <tr> <td style="padding: 0 10px;">R/W</td> <td style="padding: 0 10px;">R/W</td> <td style="padding: 0 10px;">R/W</td> <td style="padding: 0 10px;">R/W</td> </tr> </table>	R/W	R/W	R/W	R/W
R/W	R/W	R/W	R/W		

PG[2:0]: Carrier frequency selection

REME: REM output enable

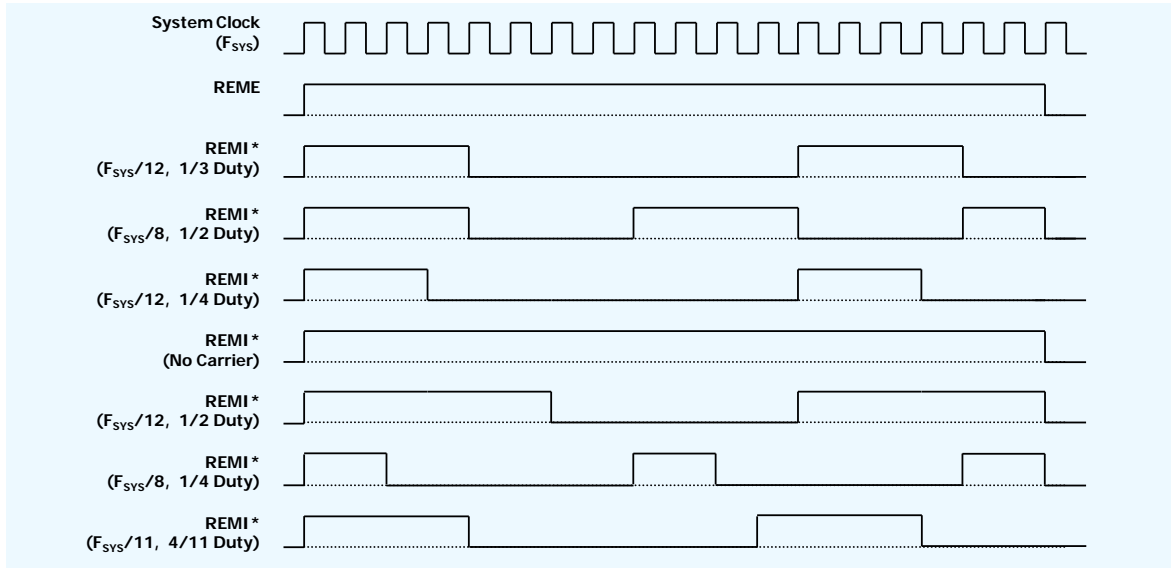


Figure 6-11 REMI* waveform examples

6.2.3 POR(Power-On Reset) and LVD(Low Voltage Detector)

6.2.3.1 Power-On Reset

The ATOM1.1 family contains the internal RC POR (Power-On-Reset) and the internal LVD POR. The logical OR between RC POR and LVD POR generates a power-on-reset pulse which restarts the system. Before the restart, the device reset timer makes about 4.5ms delay until the system clock is stabilized. After power-on reset, the POR flag is set.

When the power voltage rises rapidly, the RC POR pulse is generated. Otherwise, the LVD POR pulse is generated.

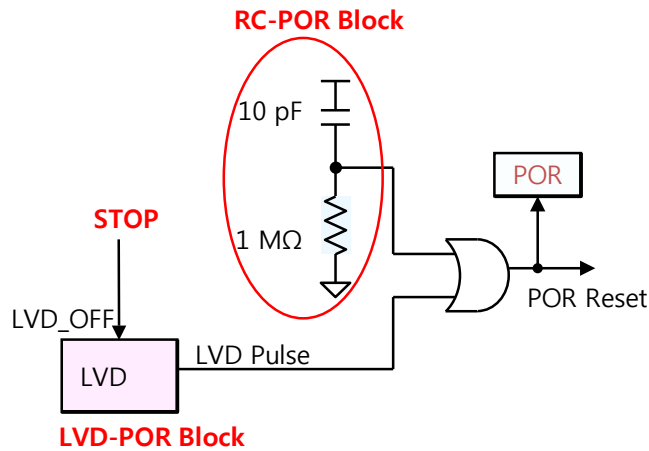


Figure 6-12 Power-On Reset and Low Voltage Detector circuit

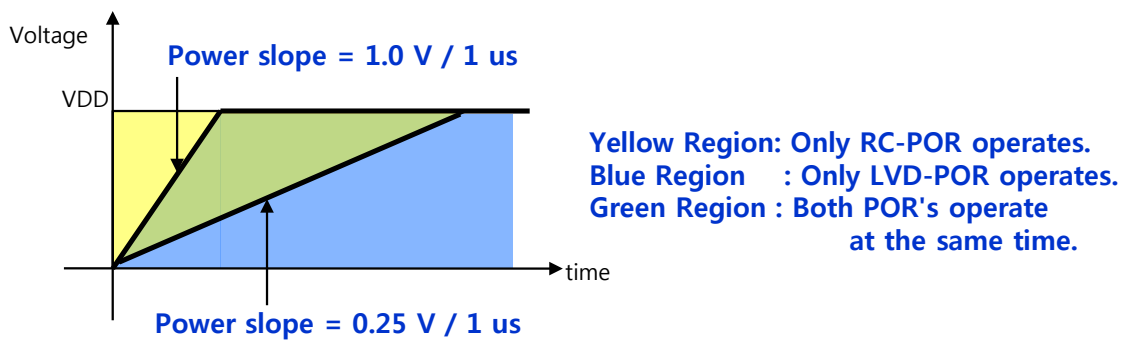


Figure 6-13 Power-On Reset Pulse Generation

6.2.3.2 Power-Failure Reset

If the power voltage falls below power-failure reset voltage, the ATOM1.1 family will enter the reset state again. This reset state will be maintained until the power voltage exceeds the power-up voltage. After exiting the reset state, the system restarts by the power-on reset and the POR flag is set.

- **LVCFG**: LVD Configuration Register

Address = 0FH		Reset Value = 1000B		
Bit	3	2	1	0
	POR	Reserved	Reserved	Reserved
	R/W(1)	R(X)	R/W(0)	R/W(0)

Reserved: Don't set this bit for the future compatibility

POR : Power-On-Reset flag to distinguish Power-On-Reset from other resets.

The reserved bits need to be blocked by AND operation while accessing this bit.

The ATOM1.1 family has the 1.6V power-fail reset voltage. Similarly, the ATOM1.1 family the 1.7V power-up voltages. When the power voltage rises above 1.7V, the Power-On-Reset pulse is generated. The power voltage falling below 1.6V, the Power-Failure-Reset pulse is generated.

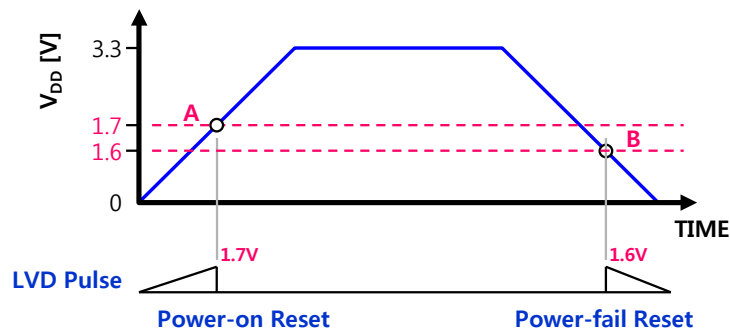


Figure 6-14 Power-fail pulse generation

6.2.3.3 POR considerations for remote controller application

The microcontroller of a remote controller spends almost of all time in the stop mode. In the stop mode, the LVD circuit is disabled by setting the STOP flag and cannot generate the LVD pulse. So only the RC-POR circuit can restart the system by the power-on reset.

For successful operation of the RC-POR circuit, several requirements of power voltage should be satisfied.

The capacitor of the RC-POR circuit must be discharged before the rising edge. The low level of the V_{DD} notch needs to be maintained for a proper period. The length of the period varies with the voltage of the low level. As the voltage of low level rises higher, the discharging period should be longer. Figure 6-15 shows discharging periods when the low level voltage is 0V, 0.5V, and 1V, respectively. For activation of RC-POR, the low level voltage should fall below 1V.

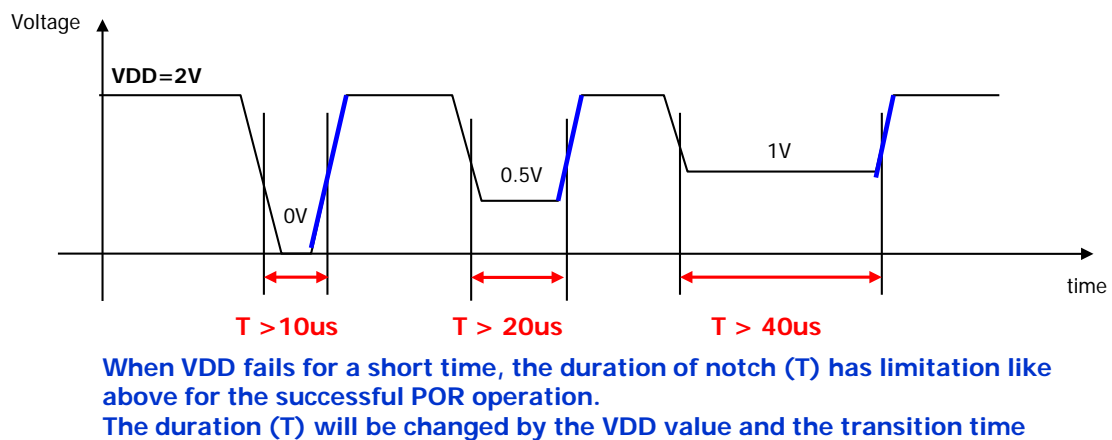


Figure 6-15 The requirements of V_{DD} notch

6.2.4 WDT (Watchdog Timer)

A Watchdog Timer automatically invokes a reset unless it is initialized regularly by the intended signals. It is used in applications that is subject to electrical noise, power glitches, electrostatic discharges, etc, or where high reliability is required.

The Watchdog timer contains a free-running counter. Its time-out interval is adjustable by changing the system clock frequency. It counts the system clock pulses. As it overflows, an internal reset, called WDT reset, is generated. The user can initialize the Watchdog Timer by setting the WDTR bit namely IFF[12]. WDTR is cleared after initializing the Watchdog Timer. The user can disable the timer by clearing the WDTE bit, namely IFF[11]. To modify this bit, first MAP1 (IFF[11]) must be set and MAP0 (IFF[10]) cleared. When the user sets flag SLEEP (IFF[14]) or writes to the SFR IAPCON, WDTE is set by hardware.

To hold off WDT reset, the user has two options:

- 1) Periodically initialize the Watchdog Timer by setting the WDTR so it will never overflow.
- 2) Disable the Watchdog Timer by clearing the WDTE bit.

The Watchdog Timer is initialized when the following events occur.

- 1) All kinds of internal resets
- 2) Entering the SLEEP mode.

- 3) Start of IAP Flash programming (erase/write)
- 4) Setting WDTR

The Watchdog time-out interval varies with the system clock frequency. Table 6-5 shows the time-out intervals of the Watchdog Timer.

Table 6-5 Time-out intervals of the Watchdog Timer

XT/RG	DIV2	DIV1	DIV0	F _{OSC} (MHz)	F _{SYS}	Interval (ms)
1	0	1	1	3.64	F _{OSC} /8	288
0	0	0	0	7.28	F _{OSC}	18
0	1	1	0	7.28	F _{OSC} /64	1152

6.2.5 Reset

The ATOM1.1 family has the five reset options: power-on reset, power-failure reset, reset for exiting from the stop mode by receiving high input from one of Port 0 and Port1 pins, watchdog timer reset, clock source changing reset. All kinds of resets restart the ATOM1.1 system. Before the restart, the device reset timer makes about 4.5ms delay until the system clock is stabilized. After reset, 00h is loaded into the program counter. So the execution starts at the 00h address.

By reset, all SFRs excluding CKCFG and IOCFG are initialized. The two SFRs are initialized only by the power-on reset. The power-on reset set the POR flag (LVCFG[3]). So a user should clear this flag after reading it. Then the flag can be used to know whether the reset source is the power-on reset or not. Their contents can be used for determination of the next action.

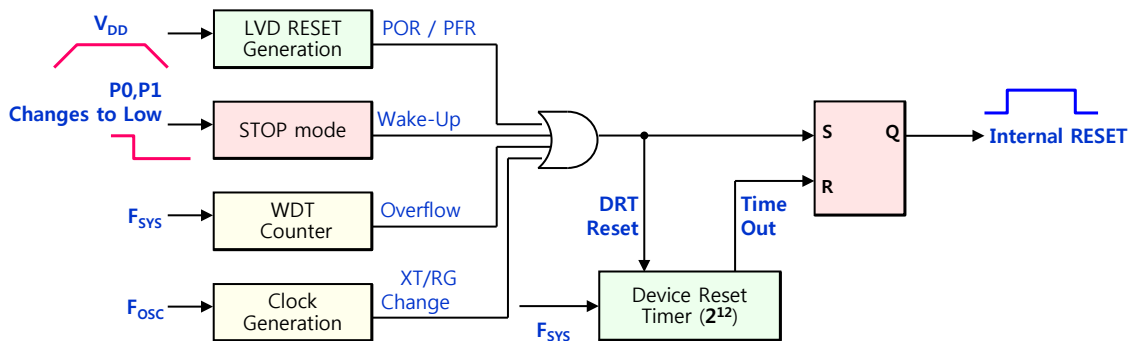


Figure 6-16 Reset Resources

6.2.5.1 Power On/Failure Reset

Refer to the 'POR and LVD' section.

6.2.5.2 Exit from the stop mode

The LVD and the watchdog timer of the ATOM1.1 family are disabled in the stop mode. So the ATOM1.1 system can exit the stop mode only by detecting a low input signal through the Port 0 and 1. The ATOM1.1 family maintains the stop mode when the input signals of Port 0 and 1. If any signal of them changes to low state, the ATOM1.1 family resets and exits from the stop mode.

6.2.5.3 Watchdog timer reset

If the watchdog timer is not initialized periodically, it overflows and invokes reset. For more information, refer to the 'watchdog timer' section.

6.2.5.4 Clock source change reset

The ATOM1.1 family has two clock sources: the internal ring oscillator and the external crystal oscillator. After power-on reset, it operates with the internal ring oscillator. To use the external crystal oscillator as the system clock, set the XT/RG bit in the CKCFG register. Then the ATOM1.1 system resets to change the clock source. After this reset, the system uses the external crystal oscillator and the internal ring oscillator stops. The contents of CKCFG register are not changed by this reset.

6.2.5.5 Device reset timer

When the system restarts by any reset, the system clock should be stabilized first. So the ATOM1.1 family contains the timer, called 'device reset timer'. It is a free-running 12 bit counter. Before the system restarts, the timer operates first. The overflow of the timer generates the internal reset signal. By the signal, the system restarts.

6.2.6 Clock Circuit (On-chip oscillators)

The ATOM1.1 family has two clock options: 1) internal ring oscillator, and 2) external resonator/crystal oscillator. After the ATOM1.1 application system starts by the power-on reset, it uses the internal ring oscillator as its system clock. To use the external crystal oscillator, set the XT/RG bit in the CKCFG register and an internal reset for changing the system clock will be caused. The internal reset doesn't change the register. Only the power-on reset initializes it.

When an ATOM1.1 application system operates, a user can change the system clock frequency by modifying the DIV[2:0] bits in the CKCFG register. Table 6-6 shows the relation between the scaling ratio and the value of DIV[2:0].

- **CKCFG:** Clock Configuration Register

	Address = 0DH		Reset Value = 0000B	
Bit	3	2	1	0
	XT/RG	DIV2	DIV1	DIV0
	R/W	R/W	R/W	R/W

XT/RG: System clock source selection

0 = The internal ring oscillator is selected. The external oscillator is disabled.

1 = The external oscillator is selected. The internal ring oscillator is disabled.

DIV[2:0]: Clock scaling bits

Table 6-6 System clock scaling

DIV2	DIV1	DIV0	F _{SYS}
0	0	0	F _{OSC}
0	0	1	F _{OSC} /2
0	1	0	F _{OSC} /4
0	1	1	F _{OSC} /8
1	0	0	F _{OSC} /16
1	0	1	F _{OSC} /32
1	1	0	F _{OSC} /64
1	1	1	-

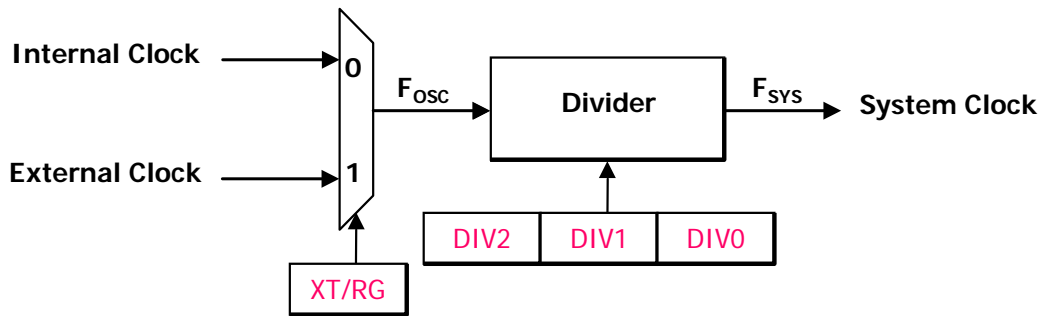
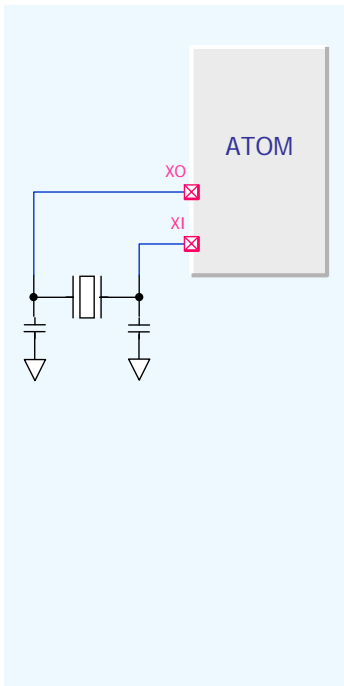


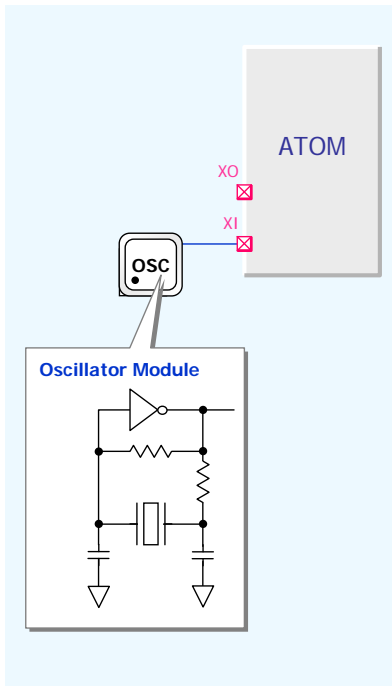
Figure 6-17 Clock Circuit

There are three options of the system clock, as shown in Figure 6-18.

◆ Resonator / Crystal Oscillator



◆ Oscillator Module



◆ Internal Ring Oscillator

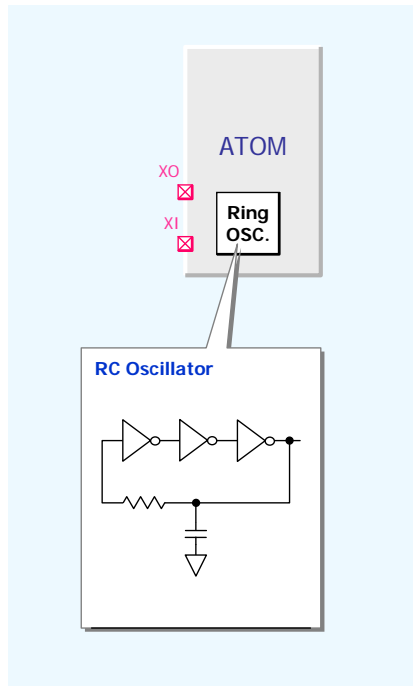


Figure 6-18 Configuration for the System Clock

6.2.7 Power Management

The ATOM1.1 family has three operation modes related with power consumption: the active mode, the sleep mode, and the stop mode. The active mode is the normal operation mode. In the mode, the CPU and all peripherals operate normally and consume much power. The sleep and the stop modes are used for power saving.

6.2.7.1 Sleep Mode

Setting the SLEEP flag (IFF[14]) put the ATOM1.1 family into the sleep mode. Only the Watchdog timer operates. The other blocks including the CPU stop. During the sleep mode, the state of I/O ports is maintained. The ATOM1.1 reset when the watchdog timer overflows. As a result, it exits from the sleep mode. The contents of CKCFG are not changed by this reset.

6.2.7.2 Stop Mode

Setting the STOP flag (IFF[15]) put the ATOM1.1 family into the sleep mode. The CPU and all peripherals stop. The state of all I/O ports is maintained. An external low input through PORT 0 or PORT 1 causes the ATOM1.1 family to reset. As a result, it exits from the stop mode. The contents of CKCFG are not changed by this reset.

6.2.8 IAP (In Application Programming)

The ATOM1.1 family can read and modify specific regions (EEP0 and EEP1) of the Flash with IAP functions during operation. Program code or data can be stored in the EEP0/1 regions.

To start IAP operation, determine the IAP region and the IAP function to use by setting the IAPCON register as shown in Table 6-7. Then the CPU suspends and the IAP function is executed. After the IAP operation, the IAPCON register is cleared automatically and the CPU resumes operating. It takes 6 system clock cycles to an IAP function read a byte. However, the IAP function for writing (erasing) a byte consumes about 2 ms. When data is written to IAPCON, the WDTE bit (IFF[13]) is also set. The watchdog timer counts the time necessary for writing (erasing) a byte by the IAP function. To do so, the timer initialized before the IAP operation. It is initialized again after the operation.

6.2.8.1 Flash regions

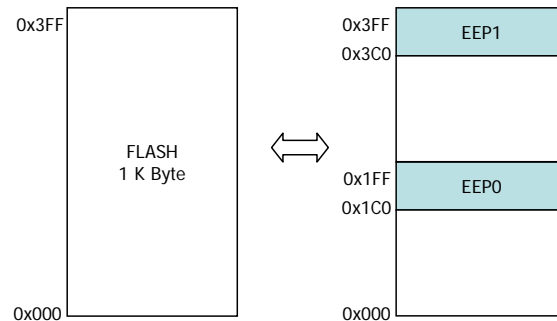


Figure 6-19 Flash regions

The flash memory (Program memory) space is divided into two program subspaces (000h–1BF0h, 200h–3BFh) and two data subspaces (EEP0: 1C0h–1FFh, EEP1: 3C0h–3FFh). The program code is stored in the two program subspaces by ISP. IAP functions access the two data spaces. If not using IAP operation, a user can use the whole flash memory for only program space.

6.2.8.2 Information region

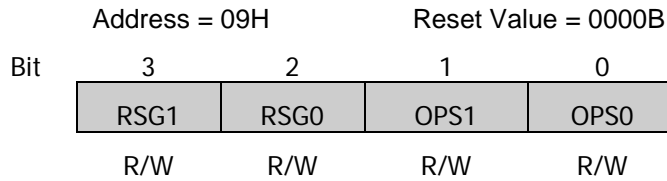
The ATOM1.1 family contains another 8-byte separate memory space, called information region. Data is written in this region by only ISP. However, only the full chip erase function of ISP can erase this region. User ID, checksum, etc. are stored in this region. Its first byte, called CFGWD, controls IAP functions.

- **CFGWD**: Configuration Word
 - CFGWD[0] (ISP_LOCK) : Disable all ISP functions excluding the full chip erase.
 - CFGWD[1] (IAP_RE) : Enable the IAP read function.
 - CFGWD[2] (IAP_PE) : Enable the IAP write/erase functions

6.2.8.3 The related SFRs

The data pointer register (DPH/DPL) is used as the least significant address register of the target byte for the IAP operation. The general purpose register (GDH/GDL) is used as the 8-bit data buffer. The IAP control register contains the IAP control information.

- **IAPCON:** IAP Control Register



RGS[1:0]: Select IAP region.

OPS[1:0]: Select IAP function.

Table 6-7 IAP region and function

RGS1	RSG0	IAP region
0	0	EEP0 (0x1C0 ~ 0x1FF)
0	1	EEP1 (0x3C0 ~ 0x3FF)
1	0	INFO (0x0 ~ 0x7)
1	1	Reserved

OPS1	OPS0	IAP region
0	0	No operation
0	1	Byte Read
1	0	Byte Erase
1	1	Byte Write

6.2.8.4 IAP enable

An IAP function can only read the INFO region. The region is not erased or written. It consists of 8 bytes.

Some information can be written to it by ISP. Its LSB byte is named CFGWD (Configuration Word).

The following conditions have to be satisfied for writing/reading data to the IAP register.

- The MAP0 bit (IFF[10]) is cleared.
- The MAP1 bit (IFF[11]) is set.
- CFGWD[1] is set for reading.
- CFGWD[2] is set for writing/erasing.

When the above conditions are not met, a user cannot access the IAPCON register. “MOV IAPCON, A” or “MOV A, IAPCON” doesn’t work.

6.2.8.5 Electrical characteristics of IAP

The IAP function execution time depends on the system clock frequency. If the frequency is out of the IAP frequency range, a user needs to adjust it for IAP operation and restore it.

Table 6-8 Electrical characteristics of IAP

Parameter	Symbol	Min.	Typ.	Max.	Unit
Power supply voltage	V _{DD}	2.7	-	5.5	V
System clock frequency	F _{SYS}	5	8	11	MHz
Write/Erase Time	T _P	1.5	2.0	3.3	ms

7 Absolute Maximum Ratings

Table 7-1 Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
V_{DD}	DC supply voltage	-0.5 to 6.5V	V
V_{IN}	DC input voltage	-0.5 to $V_{DD} + 0.5$	V
V_{OUT}	DC output Voltage	-0.5 to $V_{DD} + 0.5$	V
I_{OH}	DC output high current	One I/O pin active: -25	mA
		All I/O pins active: -100	mA
I_{OL}	DC output low current	One I/O pin active: 30	mA
		All I/O pins active: 150	mA
T_{STG}	Storage temperature	-55 to 125	°C

Table 7-2 Recommended Operating Conditions

Symbol	Parameter	Rating	Unit
V_{DD}	DC supply voltage	1.8 to 5.5	V
T_A	Industrial temperature range	-40 to 85	°C

8 DC Characteristics

Table 8-1 DC Characteristics

($T_A = -40^{\circ}\text{C} \sim +85^{\circ}\text{C}$, $V_{DD} = 1.8\text{V} \sim 5.5\text{V}$ unless otherwise specified)

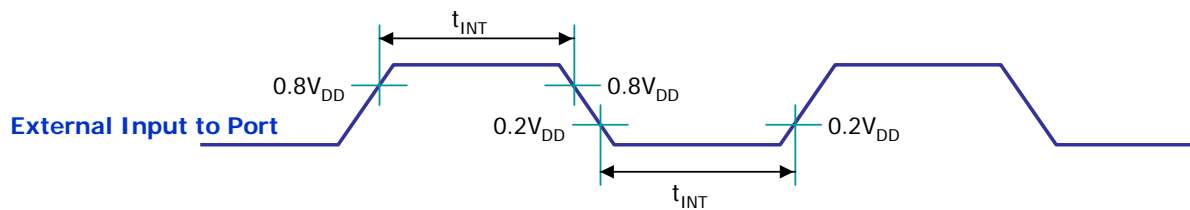
Parameter	Symbol	Pin	Conditions	Value			Unit
				Min.	Typ.	Max.	
Input low voltage	V_{IL1}	P0, P1, P2, P3	$V_{DD} = 1.8\text{V} \sim 5.5\text{V}$	-0.5	-	$0.2V_{DD} - 0.1$	V
Input high voltage	V_{IH1}	P0, P1, P2, P3	$V_{DD} = 1.8\text{V} \sim 5.5\text{V}$	$0.2V_{DD} + 1.0$	-	$V_{DD} + 0.5$	V
Input high leakage current	I_{IH}	All pins except XI, XO	$V_{IN} = V_{DD}$	-1	-	+1	μA
Output low voltage	V_{OL}	P0,P1,P2,P3	$I_{OL} = 20\text{mA} @V_{DD} = 5\text{V}$ ($I_{OL} = 3\text{mA} @V_{DD} = 2.2\text{V}$)	-	-	$0.3V_{DD}$	V
Output low voltage	V_{OL2}	REM	$I_{OL} = 280\text{mA} @V_{DD} = 3\text{V}$	-	-	0.4	V
Output high voltage	V_{OH}	P2 (push-pull output)	$I_{OH} = -15\text{mA} @V_{DD} = 5\text{V}$	$0.7V_{DD}$	-	-	V
Output high voltage	V_{OHP}	Pull-up current	$I_{OHP} = -40\mu\text{A} @V_{DD} = 5\text{V}$ ($I_{OHP} = -15\mu\text{A} @V_{DD} = 2.2\text{V}$)	$0.7V_{DD}$	-	-	V
Pin capacitance	C_{IO}	All	$V_{DD} = 5\text{V}$	-	10	-	pF

9 AC Characteristics

Table 9-1 AC Characteristics

($T_A = -40^\circ\text{C} \sim +85^\circ\text{C}$ unless otherwise specified)

Parameter	Symbol	Pin	Conditions	Value			Unit
				Min.	Typ.	Max.	
Operating Frequency (Internal Clock)	F_{OSC}		$2.7\text{V} \leq V_{\text{DD}} \leq 5.5\text{V}$	-	-	10	MHz
			$1.8\text{V} \leq V_{\text{DD}} \leq 2.7\text{V}$	-	-	5	
Operating Frequency (External Clock)	F_{OSC}	XI, XO	$2.7\text{V} \leq V_{\text{DD}} \leq 5.5\text{V}$	-	-	10	MHz
			$1.8\text{V} \leq V_{\text{DD}} \leq 2.7\text{V}$	-	-	5	
System Frequency	F_{SYS}		$1.8\text{V} \leq V_{\text{DD}} \leq 5.5\text{V}$	1/64	-	1	F_{OSC}
External Input Width	t_{INT}	P0, P1, P2, P3, P4	$1.8\text{V} \leq V_{\text{DD}} \leq 5.5\text{V}$	12	-	-	F_{SYS}



10 Package Dimension

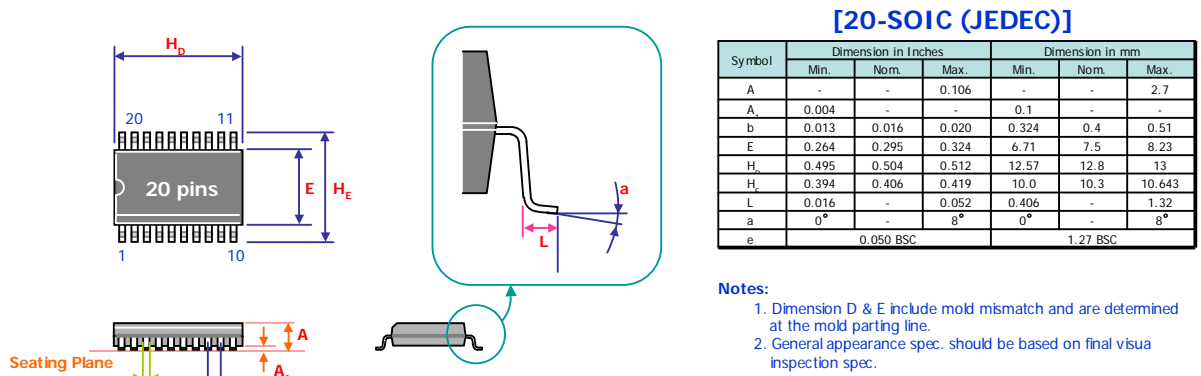


Figure 10-1 SOIC (JEDEC) 20-pin Package Dimension

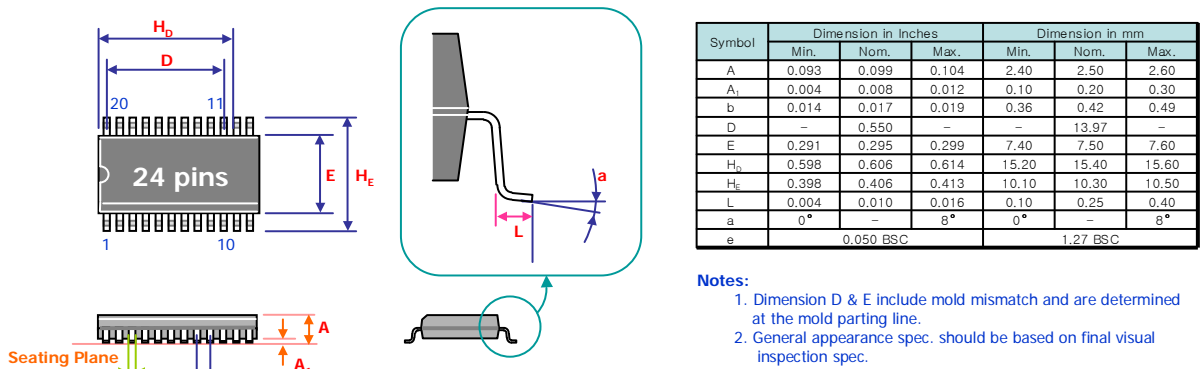


Figure 10-2 SOIC (JEDEC) 24-pin Package Dimension

11 Product Numbering System

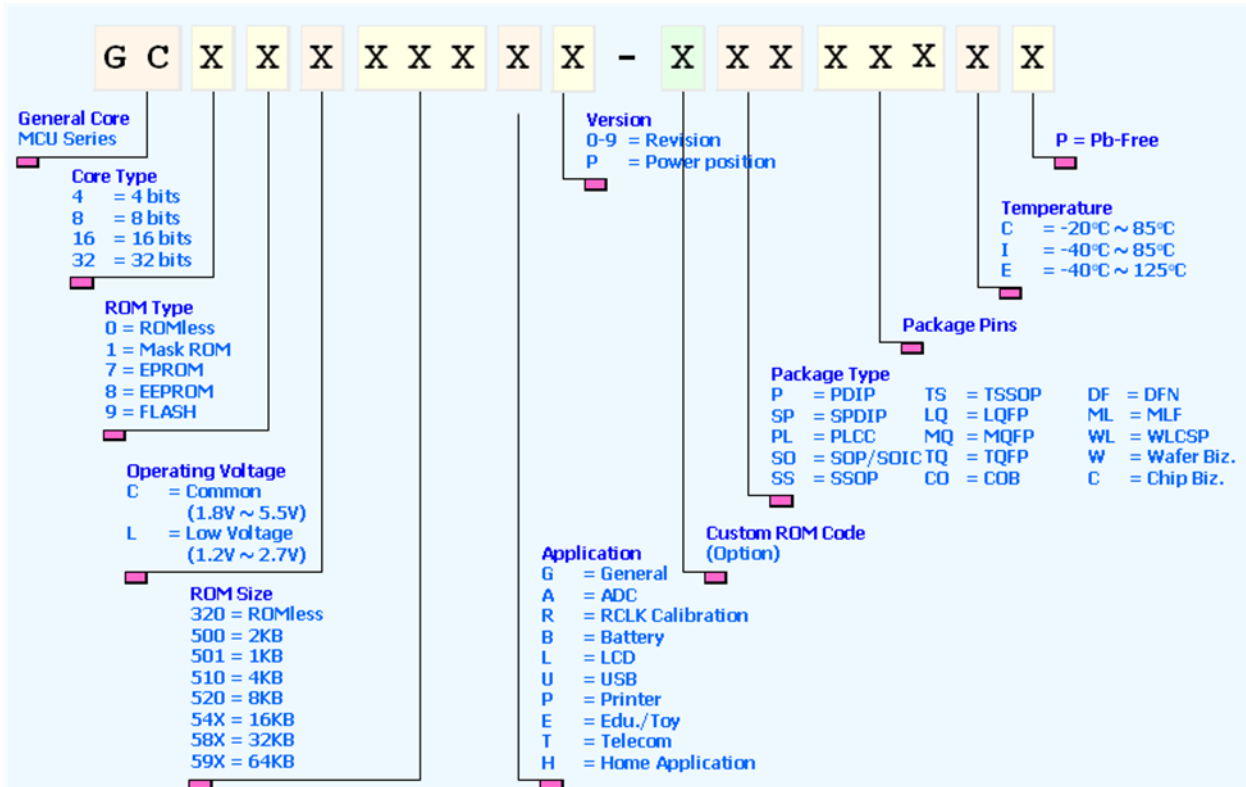


Figure 11-1 Product Numbering System

12 Appendix A: Instruction Set

Table 12-1 Abbreviations and symbols

Symbol	Description	Symbol	Description
PC	The program counter.	(PC)	The contents of PC.
A	The accumulator register (ACC).	(A)	The contents of ACC.
C	The carry flag.	(C)	The contents of C.
SP	The stack pointer register. Concatenation of SPH and SPL.	M[SP]	The contents of RAM addressed by SP.
(DP)	The contents of DPTR.	(SP)	The contents of SP.
DP	The data pointer register (DPTR). Concatenation of DPH and DPL.	M[DP]	The contents of RAM addressed by DPTR.
H	The high nibble of the data pointer (DPH).	(H)	The contents of DPH.
L	The low nibble of the data pointer (DPL).	(L)	The contents of DPL.
F[L]	The contents of indirect function flag (IFF) addressed by DPL.	rel	8-bit signed displacement value for relative branch ($-128 \leq \text{rel} \leq 127$).
#data	4-bit data operand	addr	12-bit absolute branch address.
dir	4-bit direct address of SFRs ($0 \leq \text{dir} \leq 15$)	R[dir]	The contents of SFR or read value of ports.
bit	2-bit pointer of the bit in data memory addressed by DPTR ($0 \leq \text{bit} \leq 3$).	M[DP].bit	The value of memory bit which is addressed by DPTR and bit.
@	Prefix for indirect address	Pm.n	Value of bit n of I/O port m.
\leq	Less than or equal to	.	Value of PC for current instruction.
\leftarrow	Transfer	\leftrightarrow	Exchange
=	Equal to	\neq	Not equal to
>	Greater than	<	Less than
+	Addition	-	Subtraction
&	Bitwise logical AND		Bitwise logical OR
^	Bitwise logical Exclusive-OR	~	Bitwise logical complement
{b,b}	Concatenation of bits		

Table 12-2 Opcode map

H	L	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0		NOP	SETB C	PUSH A	POP A	INC DPTR	DEC DPTR	INC @DP	DEC @DP	ADD A, @DP	ADDC A, @DP	CPL A	SUB A, @DP	ANL A, @DP	ORL A, @DP	XRL A, @DP	RRC A
1		CLR C	INC A	ADD A, #data													DEC A
2		MOV L, #data															
3		MOV H, #data															
4		MOVI @DP, #data															
5		CLR A	MOV A, #data														
6		MOV dir, A															
7		MOV A, dir															
8		MOV A, @DP	XCH A, @DP	MOV L, @DP	MOV @DP, A	MOVI @DP, A	MOVD @DP, A	CLR @L	SETB @L	CLR bit				SETB bit			
9		RET	DJNZ A, rel	CJNE A, @DP, rel	CJLE A, @DP, rel			JNC rel	JC rel	JNB bit, rel				JB bit, rel			
A		CJNE L, #data, rel															
B		CJNE @DP, #data, rel															
C		CJNE A, #data, rel															
D		CJNE A, dir, rel															
E		JMP addr															
F		CALL addr															

ADD A, #data

Binary Code	0001	dddd
Description	Adds the 4-bit data to the Accumulator. The result is stored in Accumulator. When adding unsigned integers, the carry flag indicates an overflow.	
Operation	$(A) \leftarrow (A) + \#data$	
Carry Flag	Set if a carry occurred, cleared otherwise.	
Bytes	1	
Cycles	1	
Example	CLR A ; Clear ACC ADD A, #2 ; Add 2 to ACC. ACC contains 2.	

ADD A, @DP

Binary Code	0000	1000
Description	Adds the contents of indirect data memory to the Accumulator. The result is stored in Accumulator. When adding unsigned integers, the carry flag indicates an overflow.	
Operation	$(A) \leftarrow (A) + M[DP]$	
Carry Flag	Set if a carry occurred, cleared otherwise.	
Bytes	1	
Cycles	1	
Example	; Assumes M[DP] contains 2 MOV A, #8 ; Set ACC as 8. ADD A, @DP ; The result, 10 is stored in ACC.	

ADDC A, @DP	
Binary Code	0000 1001
Description	Simultaneously adds the contents of indirect data memory, the carry flag and the Accumulator. The result is stored in Accumulator. When adding unsigned integers, the carry flag indicates an overflow.
Operation	$(A) \leftarrow (A) + M[DP] + (C)$
Carry Flag	Set if a carry occurred, cleared otherwise.
Bytes	1
Cycles	1
Example	; Assumes M[DP] contains 2 and C is 1. MOV A, #8 ; Set ACC as 8. ADDC A, @DP ; The result, 11 is stored in ACC.

ANL A, @DP

Binary Code	0000	1100
Description	ANL performs the bitwise logical-AND operation between the indirect data memory and ACC. The result is stored in Accumulator.	
Operation	$(A) \leftarrow (A) \& M[DP]$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	; Assumes M[DP] contains 2 MOV A, #0xA ; Set ACC as 10. ANL A, @DP ; The result, 2 is stored in ACC.	

CALL addr					
Binary Code	<table border="1"> <tr> <td>1111</td> <td>aaaa</td> <td>aaaa</td> <td>aaaa</td> </tr> </table>	1111	aaaa	aaaa	aaaa
1111	aaaa	aaaa	aaaa		
Description	<p>Unconditionally calls a subroutine located at the indicated 12-bit address. The instruction increments the PC twice to obtain the address of the following instruction, then push the result onto the stack (low-order nibble first). The stack pointer is incremented three times. The destination address is obtained by concatenating four low-order bits of the opcode byte and the second byte of the instruction.</p>				
Operation	$(PC) \leftarrow (PC) + 2$ $(SP) \leftarrow (SP) + 1$ $M[SP] \leftarrow (PC_{3-0})$ $(SP) \leftarrow (SP) + 1$ $M[SP] \leftarrow (PC_{7-4})$ $(SP) \leftarrow (SP) + 1$ $M[SP] \leftarrow (PC_{11-8})$ $(PC) \leftarrow \text{addr}$				
Carry Flag	Not affected.				
Bytes	2				
Cycles	2				
Example	CALL SUBR ; Call subroutine located ; at the label SUBR.				

CJLE A, @DP, rel

Binary Code	1001	0011	rrrr	rrrr
Description	Compares the contents of ACC and the indirect memory, and branches if the value in ACC is less than or equal to that in memory. The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of both operands are not affected by comparison. The carry flag is set if the contents are equal.			
Operation	$(PC) \leftarrow (PC) + 2$ $IF (A) \leq M[DP] THEN (PC) \leftarrow (PC) + rel$			
Carry Flag	$IF (A) = M[DP] THEN (C) \leftarrow 1$ $ELSE (C) \leftarrow 0.$			
Bytes	2			
Cycles	2			
Example	; Assumes M[DP] contains 11, ACC 5. CJLE A, @DP, CMP_LE; Branches to CMP_LE			
CMP_LE:			; IF (A) > M[DP]
	JC	CMP_EQ		;
CMP_EQ:			; IF (A) < M[DP]
			; IF (A) = M[DP]

CJNE @DP, #data, rel

Binary Code	1011	dddd	rrrr	rrrr
Description	<p>Compares the contents of the indirect memory and data in four low-order bits of opcode, and branches if their values are not equal.</p> <p>The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of indirect memory is not affected.</p> <p>The carry flag is set if the unsigned integer value of M[DP] is less than the unsigned integer value of the data: otherwise, the carry is cleared.</p>			
Operation	$(PC) \leftarrow (PC) + 2$ IF M[DP] \neq #data THEN $(PC) \leftarrow (PC) + rel$			
Carry Flag	IF M[DP] < #data THEN (C) \leftarrow 1 ELSE (C) \leftarrow 0.			
Bytes	2			
Cycles	2			
Example	; Assumes M[DP] contains 2. CJNE @DP, #8, CMP_NE; Branches to CMP_NE			
CMP_NE:			; IF M[DP] = 8
	JC	CMP_LT		; Branches to
CMP_LT:	CMP_LT			
			; IF M[DP] > 8
			; IF M[DP] < 8

CJNE A, #data, rel

Binary Code	1100	dddd	rrrr	rrrr
Description	Compares the contents of Accumulator and data in four low-order bits of opcode, and branches if their values are not equal. The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of ACC is not affected. The carry flag is set if the unsigned integer value of ACC is less than the unsigned integer value of the data; otherwise, the carry is cleared.			
Operation	$(PC) \leftarrow (PC) + 2$ IF $(A) \neq \#data$ THEN $(PC) \leftarrow (PC) + rel$			
Carry Flag	IF $(A) < \#data$ THEN $(C) \leftarrow 1$ ELSE $(C) \leftarrow 0$.			
Bytes	2			
Cycles	2			
Example	; Assumes ACC contains 11. CJNE A, #8, CMP_NE ; Branches to CMP_NE			
CMP_NE:			; IF $(A) = 8$
	JC CMP_LT			; Branch is not taken.
CMP_LT:			; IF $(A) > 8$
			; IF $(A) < 8$

CJNE A, @DP, rel					
Binary Code	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">1001</td> <td style="padding: 2px 10px;">0010</td> <td style="padding: 2px 10px;">rrrr</td> <td style="padding: 2px 10px;">rrrr</td> </tr> </table>	1001	0010	rrrr	rrrr
1001	0010	rrrr	rrrr		
Description	<p>Compares the contents of ACC and the indirect memory, and branches if their values are not equal.</p> <p>The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of both operands are not affected by comparison. The carry flag is set if the unsigned integer value of ACC is less than the unsigned integer value of M[DP]; otherwise, the carry is cleared.</p>				
Operation	$(PC) \leftarrow (PC) + 2$ IF $(A) \neq M[DP]$ THEN $(PC) \leftarrow (PC) + rel$				
Carry Flag	IF $(A) < M[DP]$ THEN $(C) \leftarrow 1$ ELSE $(C) \leftarrow 0.$				
Bytes	2				
Cycles	2				
Example	; Assumes M[DP] and ACC contain 15. CJNE A, @DP, CMP_NE ; Branch is not taken.				
CMP_NE:	<table style="width: 100%; border: none;"> <tr> <td style="width: 150px;">.....</td> <td style="width: 100px;">; IF (A) = M[DP]</td> </tr> <tr> <td>JNC CMP_GT</td> <td>; IF (A) \neq M[DP]</td> </tr> </table>	; IF (A) = M[DP]	JNC CMP_GT	; IF (A) \neq M[DP]
.....	; IF (A) = M[DP]				
JNC CMP_GT	; IF (A) \neq M[DP]				
CMP_GT:	<table style="width: 100%; border: none;"> <tr> <td style="width: 150px;">.....</td> <td style="width: 100px;">; IF (A) < M[DP]</td> </tr> <tr> <td>.....</td> <td>; IF (A) > M[DP]</td> </tr> </table>	; IF (A) < M[DP]	; IF (A) > M[DP]
.....	; IF (A) < M[DP]				
.....	; IF (A) > M[DP]				

CJNE A, dir, rel

Binary Code	1101	dddd	rrrr	rrrr
-------------	------	------	------	------

Description Compares the contents of ACC and that of SFR addressed by four low-order bits of opcode, and branches if their values are not equal.

The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of both operands are not affected by comparison. The carry flag is set if the unsigned integer value of ACC is less than the unsigned integer value of the SFR; otherwise, the carry is cleared.

Operation $(PC) \leftarrow (PC) + 2$
IF $(A) \neq R[dir]$ THEN $(PC) \leftarrow (PC) + rel$

Carry Flag IF $(A) < R[dir]$ THEN $(C) \leftarrow 1$
ELSE $(C) \leftarrow 0$.

Bytes 2

Cycles 2

Example ; Wait until P0 (Port 0) is 0xE.
MOV A, #0xE
CJNE A, P0, . ; Self looping with "."

CJNE L, #data, rel					
Binary Code	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">1010</td> <td style="padding: 2px 10px;">dddd</td> <td style="padding: 2px 10px;">rrrr</td> <td style="padding: 2px 10px;">rrrr</td> </tr> </table>	1010	dddd	rrrr	rrrr
1010	dddd	rrrr	rrrr		
Description	<p>Compares the contents of DPL and data in four low-order bits of opcode, and branches if their values are not equal.</p> <p>The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of DPL is not affected.</p> <p>The carry flag is set if the unsigned integer value of DPL is less than the unsigned integer value of the data; otherwise, the carry is cleared.</p>				
Operation	$(PC) \leftarrow (PC) + 2$ IF (L) \neq #data THEN $(PC) \leftarrow (PC) + rel$				
Carry Flag	IF (L) < #data THEN (C) \leftarrow 1 ELSE (C) \leftarrow 0.				
Bytes	2				
Cycles	2				
Example	; Looping with DPL MOV L, #9 ; (L) \leftarrow 9 LOOP_L: ; Operations in loop ; Operations in loop DEC DPTR ; (DP) \leftarrow (DP) - 1 CJNE L, #0, LOOP_L ; Repeat until (L) is 0.				

CLR @L

Binary Code	1000	0110
Description	Clears the indirect function flag addressed by DPL.	
Operation	$F[L] \leftarrow 0$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	; Assumes P2 contains 0xF. MOV L, #1 ; (L) \leftarrow 1 CLR @L ; P2.1 \leftarrow 0 MOV A, #0xD ; (A) \leftarrow 13 CJNE A, P2, ERROR ; Check if P2.1 is 0.	

CLR A

Binary Code	0101	0000
Description	Clears the accumulator. This is an abbreviation of MOV A, #0.	
Operation	$(A) \leftarrow 0$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	CLR A	

CLR C

Binary Code	<table border="1" style="display: inline-table;"><tr><td>0001</td><td>0000</td></tr></table>	0001	0000
0001	0000		
Description	Clears the carry flag. This is the same as "ADD A, #0".		
Operation	$(A) \leftarrow (A) + 0$		
Carry Flag	$(C) \leftarrow 0$		
Bytes	1		
Cycles	1		
Example	CLR C		

CLR bit

Binary Code	<table border="1" style="display: inline-table;"><tr><td>1000</td><td>10bb</td></tr></table>	1000	10bb
1000	10bb		
Description	Clears a bit in data memory addressed by DPTR. The bit position of the nibble is obtained by the least significant two bits of opcode.		
Operation	$M[DP].bit \leftarrow 0$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	; Assumes M[DP] contains 7. CLR 2 ; M[DP].2 \leftarrow 0 CJNE @DP, #3, ERROR ; Check result		

CPL A

Binary Code	0000	1010
Description	Complements the contents of ACC.	
Operation	$(A) \leftarrow \sim(A)$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOV A, P0	; (A) ← P0
	CPL A	; ACC contains 1's
		; complement of P0

DEC @DP

Binary Code	0000	0111
Description	Decrements the value of data memory addressed indirectly by DPTR.	
Operation	$M[DP] \leftarrow M[DP] - 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	DEC @DP	

DEC A

Binary Code	<table border="1" style="display: inline-table;"><tr><td>0001</td><td>1111</td></tr></table>	0001	1111
0001	1111		
Description	Decrements the contents of ACC. This is the same as "ADD A, #15". Carry is cleared when the borrow occurs; otherwise, carry is set.		
Operation	$(A) \leftarrow (A) + 15$		
Carry Flag	IF $(A) = 0$ THEN $C \leftarrow 0$ ELSE $C \leftarrow 1$.		
Bytes	1		
Cycles	1		
Example	DEC A		

DEC DPTR

Binary Code	<table border="1" style="display: inline-table;"><tr><td>0000</td><td>0101</td></tr></table>	0000	0101
0000	0101		
Description	Decrements the data pointer.		
Operation	$(DP) \leftarrow (DP) - 1$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	; Assumes DPTR contains 0. DEC DPTR ; By underflow, all bits ; of DPH and DPL are set. DEC DP ; This is also valid.		

DJNZ A, rel

Binary Code	1001	0001	rrrr	rrrr
Description	Decrements the contents of ACC, and branches if the result is not zero. The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. Carry is cleared when the borrow occurs; otherwise, carry is set.			
Operation	$(PC) \leftarrow (PC) + 2$ $(A) \leftarrow (A) - 1$ IF $(A) \neq 0$ THEN $(PC) \leftarrow (PC) + rel$			
Carry Flag	IF $(A) = 0$ THEN $(C) \leftarrow 0$ ELSE $(C) \leftarrow 1$.			
Bytes	2			
Cycles	2			
Example	MOV A, @DP DJNZ A, ACC_NZ			
ACC_NZ:	JNC ACC_ZERO			

INC @DP

Binary Code	0000 0110
Description	Increments the value of data memory addressed indirectly by DPTR.
Operation	$M[DP] \leftarrow M[DP] + 1$
Carry Flag	Not affected.
Bytes	1
Cycles	1
Example	INC @DP

INC A

Binary Code	0001 0001
Description	Increments the contents of ACC. This is the same as "ADD A, #1". Carry is set when the overflow occurs; otherwise, carry is cleared.
Operation	$(A) \leftarrow (A) + 1$
Carry Flag	IF $(A) = 15$ THEN $C \leftarrow 1$ ELSE $C \leftarrow 0$.
Bytes	1
Cycles	1
Example	INC A

INC DPTR

Binary Code

Description Increments the data pointer.

Operation

0000	0100
------	------

Carry Flag Not affected.

Bytes 1

Cycles 1

Example ; Assumes all bits of DPTR is 1.
INC DPTR ; By roll over, all bits
; of DPH and DPL are cleared.
INC DP ; This is also valid.

JB bit, rel					
Binary Code	<table border="1" style="display: inline-table; border-collapse: collapse;"> <tr> <td style="padding: 2px 10px;">1001</td> <td style="padding: 2px 10px;">11bb</td> <td style="padding: 2px 10px;">rrrr</td> <td style="padding: 2px 10px;">rrrr</td> </tr> </table>	1001	11bb	rrrr	rrrr
1001	11bb	rrrr	rrrr		
Description	Branches if the bit in data memory is 1. The address is given by DPTR and bit position is given by two least significant bits of opcode. The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of memory is not affected.				
Operation	$(PC) \leftarrow (PC) + 2$ IF M[DP].bit = 1 THEN $(PC) \leftarrow (PC) + rel$				
Carry Flag	Not affected.				
Bytes	2				
Cycles	2				
Example	JB 0, L_BIT_SET ; IF M[DP].0 = 0 L_BIT_SET: ; IF M[DP].0 = 1				

JC rel

Binary Code	1001	0111	rrrr	rrrr
Description	Branches if the carry flag is 1. The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction.			
Operation	$(PC) \leftarrow (PC) + 2$ IF (C) = 1 THEN $(PC) \leftarrow (PC) + rel$			
Carry Flag	Not affected.			
Bytes	2			
Cycles	2			
Example	JC L_C_SET ; IF (C) = 0 L_C_SET: ; IF (C) = 1			

JMP addr

Binary Code	1110	aaaa	aaaa	aaaa
Description	Transfers program execution to the indicated 12-bit address. The destination address is obtained by concatenating the four low-order bits of the opcode byte and the second byte of the instruction.			
Operation	(PC) ← addr			
Carry Flag	Not affected.			
Bytes	2			
Cycles	2			
Example	JMP LABEL ; Jumps to LABEL. JMP . ; Infinite loop			

JNB bit, rel

Binary Code	1001	10bb	rrrr	rrrr
Description	Branches if the bit in data memory is 0. The address of memory is given by DPTR and bit position is given by two least significant bits of opcode . The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction. The contents of memory is not affected.			
Operation	$(PC) \leftarrow (PC) + 2$ IF M[DP].bit = 0 THEN $(PC) \leftarrow (PC) + rel$			
Carry Flag	Not affected.			
Bytes	2			
Cycles	2			
Example	JNB 3, L_BIT_ZERO			
			; IF M[DP].3 = 1
L_BIT_ZERO:			; IF M[DP].3 = 0

JNC rel					
Binary Code	<table border="1" style="display: inline-table; vertical-align: middle;"> <tr> <td style="padding: 2px 10px;">1001</td> <td style="padding: 2px 10px;">0110</td> <td style="padding: 2px 10px;">rrrr</td> <td style="padding: 2px 10px;">rrrr</td> </tr> </table>	1001	0110	rrrr	rrrr
1001	0110	rrrr	rrrr		
Description	<p>Branches if the carry flag is 0.</p> <p>The branch destination is computed by adding the signed relative-displacement in the second byte of the instruction to the PC, after incrementing the PC to the start of the next instruction.</p>				
Operation	$(PC) \leftarrow (PC) + 2$ IF (C) = 0 THEN $(PC) \leftarrow (PC) + rel$				
Carry Flag	Not affected.				
Bytes	2				
Cycles	2				
Example	<pre>JNC L_C_ZERO ; IF (C) = 1 L_C_ZERO: ; IF (C) = 0</pre>				

MOV @DP, A

Binary Code	1000	0011
Description	The contents of ACC is copied to data memory whose address is given by DPTR.	
Operation	M[DP] ← (A)	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOV H, #2	; (H) ← 2
	MOV L, #14	; (L) ← 14
	MOV @DP, A	

MOV A, #data

Binary Code	0101	dddd
Description	Sets ACC with the data given in four low-order bits of opcode.	
Operation	(A) ← #data	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOV A, #-1	; (A) ← 15
	MOV A, #0xC	; (A) ← 12

MOV A, @DP

Binary Code	<table border="1"><tr><td>1000</td><td>0000</td></tr></table>	1000	0000
1000	0000		
Description	Copies the contents of data memory to ACC. The address of memory is given by DPTR.		
Operation	$(A) \leftarrow M[DP]$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	<pre>MOV H, #1 ; (H) ← 1 MOV L, #0 ; (L) ← 0 MOV A, @DP</pre>		

MOV A, dir

Binary Code	<table border="1"><tr><td>0111</td><td>dddd</td></tr></table>	0111	dddd
0111	dddd		
Description	The contents of SFR is copied to ACC. The address of SFR is given by four low-order bits of opcode.		
Operation	$(A) \leftarrow R[dir]$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	<pre>MOV A, P0 ; Read Port-0 into ACC. MOV A, L ; Move DPL to ACC. MOV A, SPH ; Move SPH to ACC.</pre>		

MOV H, #data

Binary Code	0011	dddd
Description	Sets DPH with the data given in four low-order bits of opcode.	
Operation	$(H) \leftarrow \#data$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOV H, #1 ; $(H) \leftarrow 1$	

MOV L, #data

Binary Code	0010	dddd
Description	Sets DPL with the data given in four low-order bits of opcode.	
Operation	$(L) \leftarrow \#data$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOV L, #5 ; $(L) \leftarrow 5$	

MOV L, @DP

Binary Code	<table border="1" style="display: inline-table;"><tr><td>1000</td><td>0010</td></tr></table>	1000	0010
1000	0010		
Description	Copies the contents of data memory to DPL. The address of memory is given by DPTR.		
Operation	$(L) \leftarrow M[DP]$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	MOV H, #0 MOV L, #3 MOV L, @DP ; L is changed to M[DP]		

MOV dir, A

Binary Code	<table border="1" style="display: inline-table;"><tr><td>0110</td><td>dddd</td></tr></table>	0110	dddd
0110	dddd		
Description	The contents of ACC is copied to SFR. The address of SFR is given by four low-order bits of opcode.		
Operation	$R[dir] \leftarrow (A)$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	MOV P0, A ; Output ACC to Port-0. MOV H, A ; Move ACC to DPH. MOV DPH, A ; Move ACC to DPH. MOV SPL, A ; Move ACC to SPL.		

MOVD @DP, A

Binary Code	1000	0101
Description	The contents of ACC is copied to data memory whose address is given by DPTR. After that the data pointer is decremented.	
Operation	$M[DP] \leftarrow (A)$ $(DP) \leftarrow (DP) - 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOVD @DP, A	

MOVI @DP, A

Binary Code	1000	0100
Description	The contents of ACC is copied to data memory whose address is given by DPTR. After that the data pointer is incremented.	
Operation	$M[DP] \leftarrow (A)$ $(DP) \leftarrow (DP) + 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOVI @DP, A	

MOVI @DP, #data

Binary Code

0100	dddd
------	------

Description

Set data memory whose address is given by DPTR with the data given in four low-order bits of opcode. After that the data pointer is incremented.

Operation

$$M[DP] \leftarrow \#data$$

$$(DP) \leftarrow (DP) + 1$$

Carry Flag

Not affected.

Bytes

1

Cycles

1

Example

```

; Simple look-up of constant values
MOV L, #0           ; Pointer to store
MOV H, #1           ; look-up values
CALL TABLE
    
```

TABLE:

```

MOVI @DP, #0xC
MOVI @DP, #0x0
MOVI @DP, #0x0
MOVI @DP, #0x1
RET
    
```

NOP

Binary Code	0000	0000
Description	No operation. Just fetches the next instruction.	
Operation	$(PC) \leftarrow (PC) + 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	NOP	

POP A

Binary Code	0000	0011
Description	The contents of stack top is moved to ACC. After that the stack pointer is decremented by 1.	
Operation	$(A) \leftarrow M[SP]$ $(SP) \leftarrow (SP) - 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	; Looping with variable stored in stack MOV A, #7 ; Set loop count LOOP_BGN: PUSH A ; Store loop index in stack. ; Operations in loop POP A ; Restore loop index DJNZ A, LOOP_BGN ; Iteration	

ORL A, @DP

Binary Code	<table border="1" style="display: inline-table;"><tr><td>0000</td><td>1101</td></tr></table>	0000	1101
0000	1101		
Description	ORL performs the bitwise logical-OR operation between the indirect data memory and ACC. The result is stored in Accumulator.		
Operation	$(A) \leftarrow (A) \mid M[DP]$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	; Assumes M[DP] contains 1 MOV A, #0xA ; Set ACC as 10. ORL A, @DP ; The result, 11 is stored in ACC.		

PUSH A

Binary Code	<table border="1" style="display: inline-table;"><tr><td>0000</td><td>0010</td></tr></table>	0000	0010
0000	0010		
Description	The stack pointer is incremented by 1. Then the contents of ACC is copied to the stack.		
Operation	$(SP) \leftarrow (SP) + 1$ $M[SP] \leftarrow (A)$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	PUSH A ; Store ACC in stack MOV A, #0xE ; Assign ACC for port output MOV P2, A ; Drive Port 2 POP A ; Restore ACC from stack		

RET

Binary Code	1001	0000
Description	Returns from subroutine. The stack pointer is decremented three times.	
Operation	$(PC_{11-8}) \leftarrow M[SP]$ $(SP) \leftarrow (SP) - 1$ $(PC_{7-4}) \leftarrow M[SP]$ $(SP) \leftarrow (SP) - 1$ $(PC_{3-0}) \leftarrow M[SP]$ $(SP) \leftarrow (SP) - 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	2	
Example	RET	

SETB C

Binary Code	0000	0001
Description	Sets the carry flag.	
Operation		
Carry Flag	$(C) \leftarrow 1$	
Bytes	1	
Cycles	1	
Example	SETB C	

RRC A

Binary Code	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="padding: 2px 10px;">0000</td><td style="padding: 2px 10px;">1111</td></tr></table>	0000	1111
0000	1111		
Description	Rotates right the contents of ACC with the carry flag.		
Operation	$(A) \leftarrow \{(C), (A_{3-1})\}$		
Carry Flag	$(C) \leftarrow (A_0)$		
Bytes	1		
Cycles	1		
Example	RRC A JC A0_HIGH ; IF $A_0 = 1$ Branches		

SETB @L

Binary Code	<table border="1" style="display: inline-table; vertical-align: middle;"><tr><td style="padding: 2px 10px;">1000</td><td style="padding: 2px 10px;">0111</td></tr></table>	1000	0111
1000	0111		
Description	Sets the indirect function flag addressed by DPL.		
Operation	$F[L] \leftarrow 1$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	; Assumes P2 contains 0. MOV L, #1 ; $(L) \leftarrow 1$ SETB @L ; $P2.1 \leftarrow 1$ MOV A, #2 ; $(A) \leftarrow 2$ CJNE A, P2, . ; Wait until P2.1 is 1.		

SETB bit

Binary Code	1000	11bb
Description	Sets a bit in data memory indirectly addressed by DPTR. The bit position is obtained at the least significant two bits of opcode.	
Operation	$M[DP].bit \leftarrow 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	; Assumes M[DP] contains 5. SETB 2 ; M[DP].2 \leftarrow 1 CJNE @DP, #7, ERROR ; Check result	

SUB A, @DP

Binary Code	0000	1011
Description	Subtracts the contents of indirect data memory from the Accumulator. The result is stored in Accumulator. The carry flag is cleared if the unsigned value of ACC is less than unsigned value of M[DP]; otherwise, C is set.	
Operation	$(A) \leftarrow (A) - M[DP]$	
Carry Flag	If $(A) < M[DP]$ THEN $(C) \leftarrow 0$ ELSE $(C) \leftarrow 1$.	
Bytes	1	
Cycles	1	
Example	SUB A, @DP	

XCH A, @DP

Binary Code	1000	0001
Description	Exchanges the contents of ACC and that of data memory addressed by DPTR.	
Operation	$(A) \leftrightarrow M[DP]$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	XCH A, @DP	

XRL A, @DP

Binary Code	0000	1110
Description	XRL performs the bitwise logical Exclusive-OR operation between the indirect data memory and ACC. The result is stored in Accumulator.	
Operation	$(A) \leftarrow (A) \wedge M[DP]$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	; Assumes M[DP] contains 2 MOV A, #0xA ; Set ACC as 10. XRL A, @DP ; The result, 8 is stored in ACC.	

13 Appendix B: SFR description

Here is an example of SFR description. The reset value described here is the power-on reset value.

P4: Port4 Output Register

	Address = 01H		Reset Value = 1111B	
Bit	3	2	1	0
	P4.3	P4.2	P4.1	P4.0
	R/W	R/W	R/W	R/W

Yellow: Bit/Nibble addressable

White: Nibble addressable

R: Readable bit

W: Writable bit

13.1 P0: Port 0 Register

	Address = 00H		Reset Value = 1111B	
Bit	3	2	1	0
	P0.3	P0.2	P0.1	P0.0
	R/W	R/W	R/W	R/W

13.2 P4: Port 4 Register

	Address = 01H		Reset Value = 1111B	
Bit	3	2	1	0
	P4.3	P4.2	P4.1	P4.0
	R/W	R/W	R/W	R/W

13.3 DPL: The Low Nibble of Data Pointer

	Address = 02H		Reset Value = 0000B	
Bit	3	2	1	0
	DPL.3	DPL.2	DPL.1	DPL.0
	R/W	R/W	R/W	R/W

13.4 DPH: The High Nibble of Data Pointer

	Address = 03H		Reset Value = --00B	
Bit	3	2	1	0
	-	-	DPH.1	DPH.0
			R/W	R/W

13.5 P1: Port 1 Register

	Address = 04H		Reset Value = 1111B	
Bit	3	2	1	0
	P1.3	P1.2	P1.1	P1.0
	R/W	R/W	R/W	R/W

13.6 REMC: REM Output Control Register

	Address = 05H		Reset Value = 0000B	
Bit	3	2	1	0
	REME	PG2	PG1	PG0
	R/W	R/W	R/W	R/W

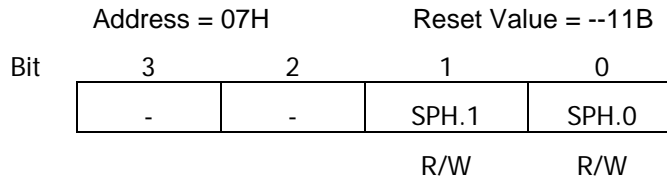
PG[2:0]: Carrier frequency selection.

IOMAP[1:0]: REM output enable.

13.7 SPL: Low Nibble of Stack Pointer

	Address = 06H		Reset Value = 1111B	
Bit	3	2	1	0
	SPL.3	SPL.2	SPL.1	SPL.0
	R/W	R/W	R/W	R/W

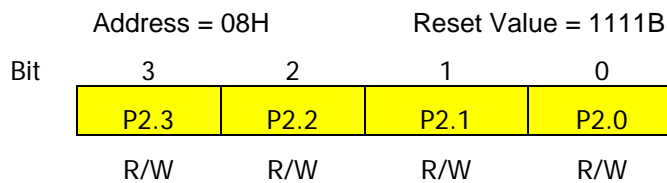
13.8 SPH: High Nibble of Stack Pointer



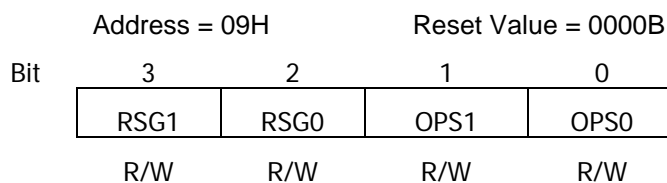
SPH and SPL contain the address of the stack.

The address is incremented by the PUSH instruction and decremented by the POP instruction.

13.9 P2: Port 2 Register



13.10 IAPCON: IAP Control Register



RGS[1:0]: Select IAP region.

OPS[1:0]: Select IAP function.

RGS1	RSG0	IAP region
0	0	EEP0 (0x1C0 ~ 0x1FF)
0	1	EEP1 (0x3C0 ~ 0x3FF)
1	0	INFO (0x0 ~ 0x7)
1	1	Reserved

OPS1	OPS0	IAP region
0	0	No operation
0	1	Byte read
1	0	Byte erase
1	1	Byte write

13.11 GDL: Low Nibble of General Purpose Data Register

Address = 0AH Reset Value = 0000B

Bit	3	2	1	0
	GDL.3	GDL.2	GDL.1	GDL.0
	R/W	R/W	R/W	R/W

13.12 GDH: High Nibble of General Purpose Data Register

Address = 0BH Reset Value = 0000B

Bit	3	2	1	0
	GDH.3	GDH.2	GDH.1	GDH.0
	R/W	R/W	R/W	R/W

13.13 P3: Port 3 Register

Address = 0CH Reset Value = 1111B

Bit	3	2	1	0
	P3.3	P3.2	P3.1	P3.0
	R/W	R/W	R/W	R/W

13.14 CKCFG: LVD Configuration Register

Address = 0DH Reset Value = 0000B

Bit	3	2	1	0
	XT/RG	DIV2	DIV1	DIV0
	R/W	R/W	R/W	R/W

XT/RG: System clock source selection

0 = The internal ring oscillator is selected. The external oscillator is disabled.

1 = The external oscillator is selected. The internal ring oscillator is disabled.

Don't set this bit for the 8-pin package.

DIV[2:0]: Clock scaling bits

Table 13-1 System clock scaling

DIV2	DIV1	DIV0	F _{sys}
0	0	0	F _{OSC}
0	0	1	F _{OSC} /2
0	1	0	F _{OSC} /4
0	1	1	F _{OSC} /8
1	0	0	F _{OSC} /16
1	0	1	F _{OSC} /32
1	1	0	F _{OSC} /64
1	1	1	-

13.15 IOCFG: I/O port Configuration Register

Address = 0EH

Reset Value = 0000B

Bit	3	2	1	0
	IOMAP1	IOMAP0	P2OEN	IOXEN
	R/W	R/W	R/W	R/W

IOXEN: The port configuration of PORT4[1:0]

0: PORT4[1:0] are used as the external clock input out output.

1: PORT4[1:0] are used as general I/O pins.

P2OEN: The port configuration of PORT2.

IOMAP[1:0]: Selection of port mapping options.

[0,0] : Default.

[0,1] : Optional 20-pin I/O port mapping

[1,0] : Optional 24-pin I/O port mapping

[1,1] : Reserved

14 Appendix C: History

- ◆ V1.0
 - ✓ First Official Release

- ◆ V1.1
 - ✓ Change Industrial temperature range