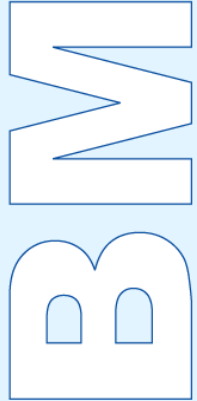




ATOM Family

BM-ATOM1.0-Korean-V1.7



Brief Manual of ATOM1.0 Family

4-bit Microcontrollers with Reduced 8051 Architecture

V1.7

August 2009

- ◆ 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 month 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 products listed in this document are intended for usage in general electronics applications. These CORERIVER 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.

1. 제품 개요

2. 특징

3. 블록 도표

4. 핀 구성

5. 핀 설명

6. 기능 설명

✓ CPU 설명

- 메모리 구조
- 특수기능레지스터의 구성과 설명
- 명령어 세트 설명
- CPU 타이밍

✓ 주변회로 설명

- I/O 단자
- Clock 구조
- Carrier 주파수 생성
- LVD (저전압 검출회로)
- WDT (Watchdog Timer)
- Reset 회로
- 전력 관리
- IAP (In Application Programming)

7. Absolute Maximum Ratings

8. DC 특성

9. AC 특성

10. Package Dimensions

11. 제품 번호 체계

12. 지원 툴

13. 부록

A. 명령어 세트

B. 특수기능레지스터 설명

C. Update History

1. 제품 개요 (1/2)

Preliminary

◆ ATOM1.0 군 -GC49C501 계열 (저비용, 저전력 응용 MCU)

Product	Mask-ROM (byte)	FLASH (byte)	EEPROM (byte)	RAM (Nibble)	Volt (V)	Freq. (MHz)	T/C (16bits)	Serial I/O	WDT	REM Output	IR. LED Drive Tr.	I/O Pins	Package	Others	Available Time
GC49C501G0-SO24I	-	1K	(128)	64	1.8~5.5	10 (5)	-	-	1	1	Yes	18 (20)	24-SOIC	POR/LVD Ring OSC ISP/IAP	NOW
GC49C501G0-SJ20I	-	1K	(128)	64	1.8~5.5	10 (5)	-	-	1	1	Yes	14 (16)	20-SOIC (JEDEC)	POR/LVD Ring OSC ISP/IAP	NOW
GC49C501R0-SO24I	-	1K	(128)	64	1.8~5.5	10 (5)	-	-	1	1	Yes	18 (20)	24-SOIC	POR/LVD Calibrated Ring OSC ISP/IAP	NOW
GC49C501R0-SJ20I	-	1K	(128)	64	1.8~5.5	10 (5)	-	-	1	1	Yes	14 (16)	220-SOIC (JEDEC)	POR/LVD Calibrated Ring OSC ISP/IAP	NOW
GC49C501RP-SO8I	-	1K	(128)	64	1.8~5.5	10 (5)	-	-	1	-	-	6	8-SOIC	POR/LVD Calibrated Ring OSC ISP/IAP	NOW
GC49C501RP-SP8I	-	1K	(128)	64	1.8~5.5	10 (5)	-	-	1	-	-	6	8-SPDIP	POR/LVD Calibrated Ring OSC ISP/IAP	NOW

* 프로그램 영역의 일부(128 bytes)를 EEPROM으로 사용할 수 있는데, 그 영역은 소프트웨어에서 IAP 기능으로 수정할 수 있다.

* ATOM1.0 군의 최대 동작 주파수는 VDD가 2.7V보다 낮을 때 5 MHz이다.

1. 제품 개요 (2/2)

Preliminary

◆ ATOM1.0 군 -GC49C501 계열 (저비용, 저전력 응용 MCU)

Product	Mask-ROM (byte)	FLASH (byte)	EEPROM (byte)	RAM (Nibble)	Volt (V)	Freq. (MHz)	T/C (16bits)	Serial I/O	WDT	REM Output	IR. LED Drive Tr.	I/O Pins	Package	Others	Available Time
GC41C501G0-SO24I	1K	-	-	64	1.8~5.5	10 (5)	-	-	1	1	Yes	18 (20)	24-SOIC	POR/LVD Ring OSC	NOW
GC41C501G0-SJ20I	1K	-	-	64	1.8~5.5	10 (5)	-	-	1	1	Yes	14 (16)	20-SOIC (JEDEC)	POR/LVD Ring OSC	NOW
GC41C501G0-SO8I	1K	-	-	64	1.8~5.5	10 (5)	-	-	1	-	-	6	8-SOIC	POR/LVD Ring OSC	NOW
GC41C501G0-SP8I	1K	-	-	64	1.8~5.5	10 (5)	-	-	1	-	-	6	8-SPDIP	POR/LVD Ring OSC	NOW

* 프로그램 영역의 일부(128 bytes)를 EEPROM으로 사용할 수 있는데, 그 영역은 소프트웨어에서 IAP 기능으로 수정할 수 있다.

* ATOM1.0 군의 최대 동작 주파수는 VDD가 2.7V보다 낮을 때 5 MHz이다.

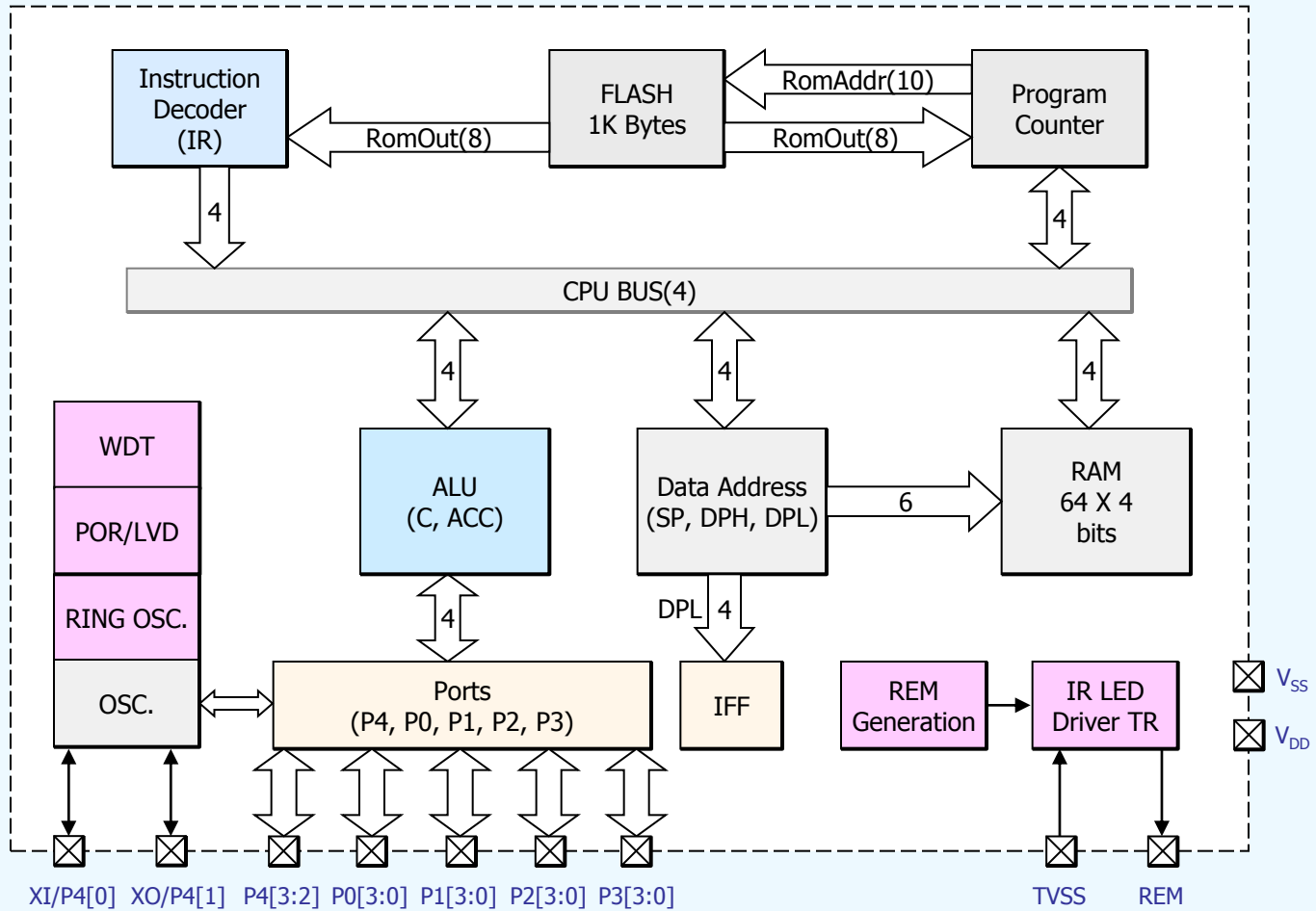
- ◆ CPU
 - ✓ 4-비트 축소 8051 구조
 - ✓ 연속적 프로그램 주소, 페이지 방식은 아님
 - ✓ Push, pop, logic inst.을 포함하는 51 명령어
 - ✓ 명령어 주기 : $F_{sys}/6$
 - ✓ 다중 서브루틴 내포 RAM 기반 스택.
- ◆ On-chip 메모리
 - ✓ FLASH : 1024 bytes (128 바이트 EEPROM 포함)
 - ✓ RAM : 64 nibbles (스택 포함)
- ◆ ISP (In System Programming) of FLASH
- ◆ IAP (In Application Programming) of FLASH
- ◆ 입출력 단자
 - ✓ P0 : 4비트 병렬 I/O (Open drain output)
 - ✓ P1 : 병렬 I/O (Open drain output),
24핀은 4비트, 20핀은 2비트.
 - ✓ P2, P3 : 4비트 병렬/비트 선택 I/O (Open drain output)
 - ✓ P4 : 병렬 I/O (Open drain output).
내부 클럭을 사용할 때 2비트
24핀 패키지는 추가 2비트.
- ◆ REM 출력 (원격 조정 송신기)
 - ✓ 적외선 LED 구동용 Built-in Transistor
 - ✓ $I_{OL} = 300 \text{ mA}$ (최대), $V_{DD} = 3V$ 그리고 $V_O = 0.4V$
- ◆ Carrier 펄스 생성기 : 7 가지
- ◆ Built-in 발진기
 - ✓ Crystal/Ceramic 공진기
 - ✓ 정밀 내부 발진기
 - 2.1 ~ 3.3V에서 $\pm 3\%$ 범위로 조정됨
 - 2.5V에서 $\pm 1\%$ 범위로 조정됨
 - ✓ GC49C501RX 디바이스에만 7.28MHz 발진조정이 적용됨.
- ◆ Built-in Reset
 - ✓ Power-on Reset, Power-fail Reset
 - ✓ WDT (Watch-Dog Timer) Reset
 - ✓ Clock switching reset
- ◆ 전력 관리
 - ✓ Power-down (stop) mode
 - ✓ P0, P1 포트가 0으로 떨어지면 stop mode 탈출
 - ✓ Sleep mode

2. 특징

- ◆ 전력 소모
 - ✓ Stop mode : <math>< 0.1\mu\text{A}</math> (Typ.) at 2.0V
1 μA (Max.) at 5.0V
 - ✓ Normal mode : 400 μA (Typ.) at 2.0V, $F_{\text{SYS}} = 4 \text{ MHz}$
- ◆ 전압 대비 동작 주파수
 - ✓ 최대 $F_{\text{OSC}} = 10 \text{ MHz}$ ($2.7 \text{ V} \leq V_{\text{DD}} \leq 5.5\text{V}$)
 - ✓ 최대 $F_{\text{OSC}} = 5 \text{ MHz}$ ($1.8 \text{ V} \leq V_{\text{DD}} \leq 2.7\text{V}$)
- ◆ 동작 온도 : $-40 \text{ }^\circ\text{C} \sim 85 \text{ }^\circ\text{C}$
- ◆ E.S.D. protection 2,000V
- ◆ Latch-up protection $\pm 200\text{mA}$
- ◆ 패키지
 - ✓ 24-pin SOIC
 - ✓ 20-pin SOIC
 - ✓ 8-pin SOIC/SPDIP

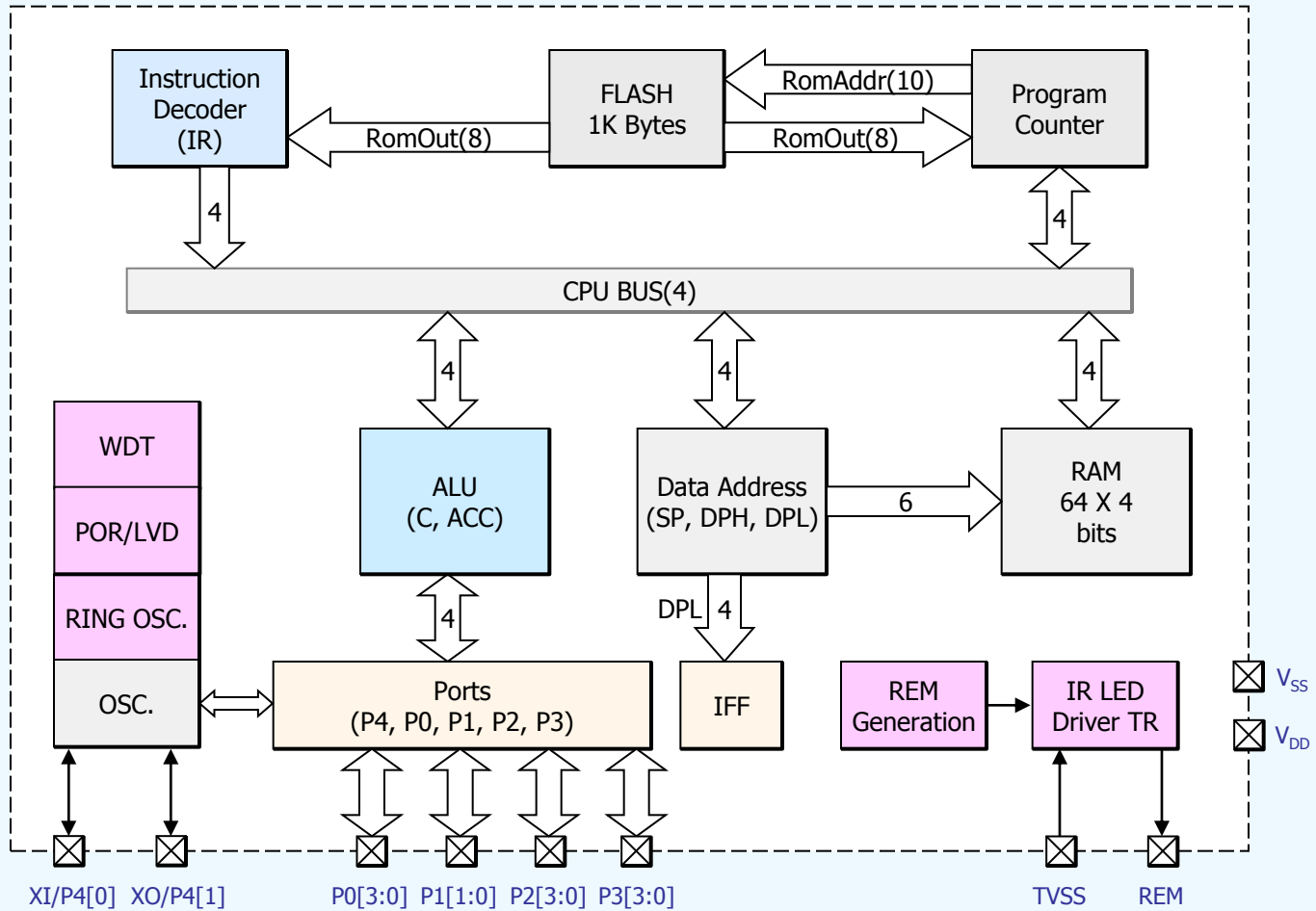
3. 블록 도표 (24-PIN)

Preliminary



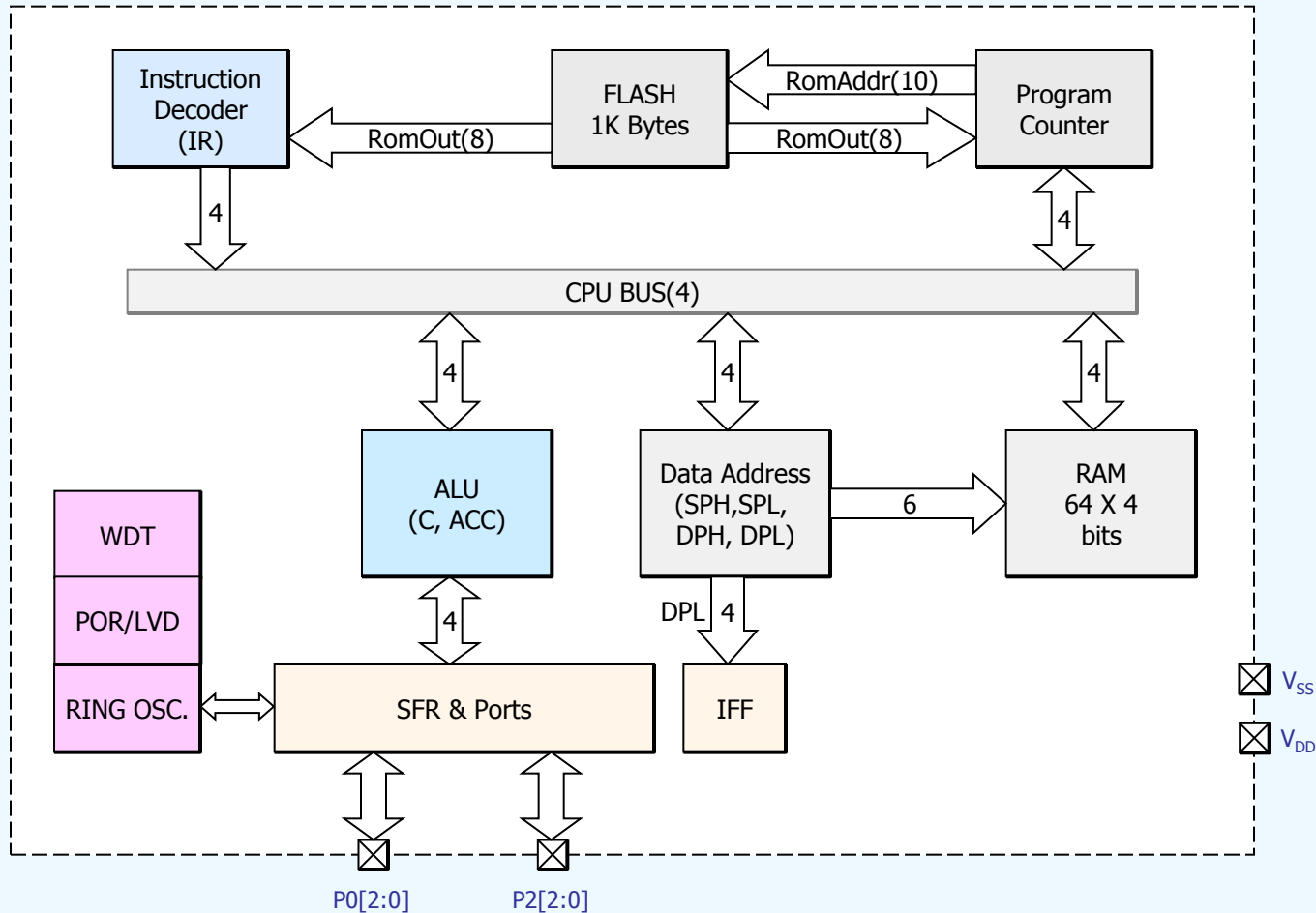
3. 블록 도표 (20-PIN)

Preliminary

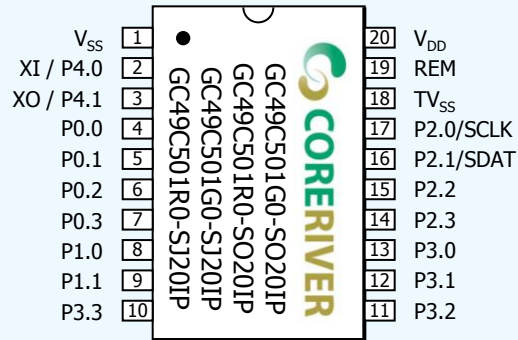


3. 블록 도표(8-PIN)

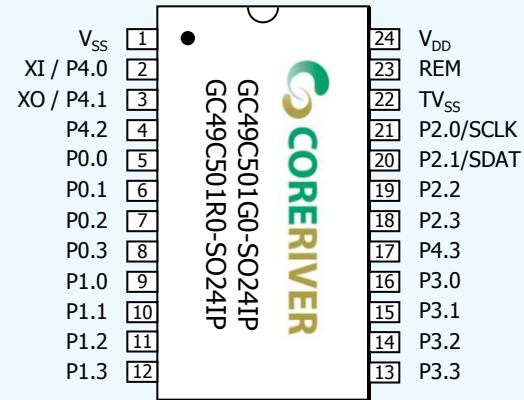
Preliminary



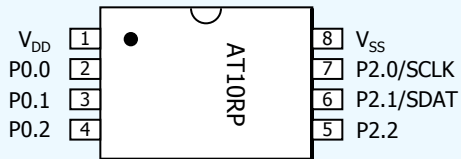
4. 핀 구성



[20-SOIC]
GC49C501G0-SO20I
GC49C501R0-SO20I
GC49C501G0-SJ20I
GC49C501R0-SJ20I



[24-SOIC]
GC49C501G0-SO24I
GC49C501R0-SO24I



[8-SOIC/SPDIP]
GC49C501RP-SO8I
GC49C501RP-SP8I

5. 핀 설명 (20핀/24핀)

Preliminary

Symbol	Direction	Description	Remark
V _{DD}	Power	전력 공급	
V _{SS}	Power	Ground	
REM	Output	적외선 LED 구동 n-채널 Transistor 출력.	
TV _{SS}	Power	적외선 LED 구동 Transistor Ground	
XI / P4[0]	Input/Output	위상반전 발진기의 증폭기 입력. 입출력 단자로 설정되면, P4[0]는 병렬 입출력 단자다. Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력.	
XO / P4[1]	Input/Output	위상반전 발진기의 증폭기 출력. 입출력 단자로 설정되면, P4[0]는 병렬 입출력 단자다. Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력.	
P4[3:2]	Input/Output	병렬 입출력 단자(오직 24핀 패키지에서 사용됨) 각 비트는 개별적으로 설정되거나 소거될 수 있다. Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력.	
P0[3:0]	Input/Output	병렬 입출력 단자 Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력. 각 핀의 입력이 0이면 STOP mode에서 탈출함.	
P1[1:0]	Input/Output	병렬 입출력 단자 Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력. 각 핀의 입력이 0이면 STOP mode에서 탈출함.	
P1[3:2]	Input/Output	병렬 입출력 단자(오직 24핀 패키지에서 사용됨) Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력. 각 핀의 입력이 0이면 STOP mode에서 탈출함.	
P2[3:0]	Input/Output	병렬 입출력 단자. 각 비트는 개별적으로 설정되거나 소거될 수 있다. Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력. P2는 push-pull 출력단자로 설정될 수 있음. P2[0]와 P2[1]는 또한 FLASH memory의 ISP로도 사용됨.	
P3[3:0]	Input/Output	병렬 입출력 단자. 각 비트는 개별적으로 설정되거나 소거될 수 있다. Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력.	

5. 핀 설명 (8 핀)

Symbol	Direction	Description	Remark
V _{DD}	Power	전력 공급	
V _{SS}	Power	Ground	
P0[2:0]	Input/Output	병렬 입출력 단자 Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력. 각 핀의 입력이 0이면 STOP mode에서 탈출함.	
P2[2:0]	Input/Output	병렬 입출력 단자. 각 비트는 개별적으로 설정되거나 소거될 수 있다. Schmitt Trigger 입력 그리고 내부 pull-up TR.을 갖는 open-drain 출력. P2는 push-pull 출력단자로 설정될 수 있음. P2[0]와 P2[1]는 또한 FLASH memory의 ISP로도 사용됨.	

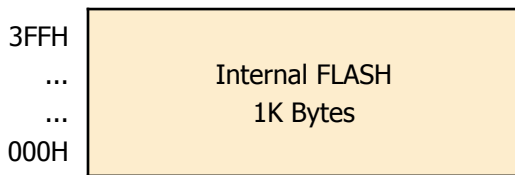
6.1. 메모리 구조

Preliminary

◆ 주소 공간

- ✓ 프로그램 메모리 : 1K Bytes.
바이트 단위로 연속 주소.
- ✓ 간접 주소지정 데이터 메모리 : 64 Nibbles.
비트 단위 접근 가능.
- ✓ 특수기능 레지스터 : 16개.
직접 주소지정.
- ✓ 간접 주소지정 기능 flags : 16 비트.
비트 위치는 DPL에 의해 선정됨.

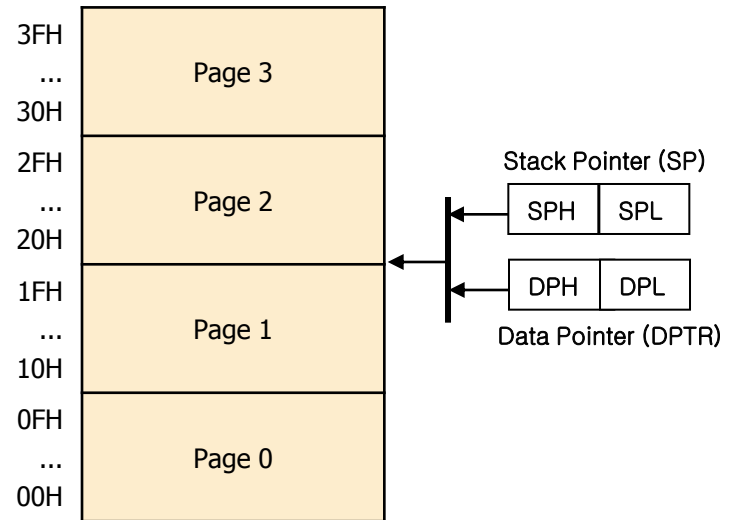
[Program Memory Map]



[Special Function Register Map]

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

[RAM Map]



[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

6.2. SFR에 대한 간단한 설명

Preliminary

레지스터	주소	설명	Power-On Reset Value	Other Reset Value
P0	00H	단자 0 출력 레지스터.	1111	1111
P4	01H	단자 4 출력 레지스터.	1111	1111
DPL	02H	데이터 포인터의 하위 nibble (DPTR).	0000	0000
DPH	03H	데이터 포인터의 상위 nibble (DPTR).	--00	--00
P1	04H	단자 1 출력 레지스터.	1111	1111
REMC	05H	REM 출력 제어 레지스터.	0000	0000
SPL	06H	스택 포인터의 하위 nibble (SP).	1111	1111
SPH	07H	스택 포인터의 상위 nibble (SP).	--01	--01
P2	08H	단자 2 출력 레지스터.	1111	1111
IAPCON	09H	IAP (In Application Programming) 제어 레지스터. 오직 MAP1이 1이고 MAP0이 0일 때 접근이 가능함	0000	0000
GDL	0AH	범용 데이터 레지스터의 하위 nibble	0000	0000
GDH	0BH	범용 데이터 레지스터의 상위 nibble	0000	0000
P3	0CH	단자 3 출력 레지스터.	1111	1111
CKCFG	0DH	클럭 상태 설정 레지스터. 파워 온 리셋에 의하여 초기화됨.	0000	uuuu
IOCFG	0EH	I/O 단자 설정 레지스터. 파워 온 리셋에 의하여 초기화됨.	0000	uu0u
LVCFG	0FH	LVD 상태 설정 레지스터. 파워 온 리셋에 의하여 초기화됨.	1x00	uxuu

- : Unimplemented bit. Read as 0.

u: Remains unchanged.

x: The value of the bit is not determined

Note for 8-pin devices.

- Not supported SFRs : P1, P3, P4, REMC.

- Writing to the not-supported SFRs may cause unexpected behavior.

6.2. 간접주소지정 기능 Flag (IFF) 설명

◆ 간접주소지정 기능 Flag (IFF)

- ✓ 쓰기만 가능, 다음의 명령으로 접근이 가능함: MOV L, #n, SETB @L, CLR @L
- ✓ 단자를 개별적으로 설정/소거하는 것은 사용하는 패키지가 상응하는 병렬 단자를 지원할 때만 가능하다.

Flag	주소 (DPL)	설명	Reset Value
STOP	15	Stop mode에 들어감. P0와 P1의 상태가 high일 때만 1로 설정됨.	0
SLEEP	14	Sleep mode에 들어감. WDT(WatchDog Timer) 리셋에 의하여 탈출.	0
WDTE	13	WDT의 인에이블 flag. 이 비트가 소거되면 WDT는 동작을 멈추고 이전 상태를 유지한다. 오직 MAP1 비트가 설정되고 MAP0 비트가 소거되어있는 상태에서만 이 비트를 변경할수 있다. 이 비트는 사용자가 SLEEP 비트를 설정하거나 IAPCON SFR에 값을 쓰면 하드웨어에 의해서 자동으로 1로 설정된다.	1
WDTR	12	WDT 초기화. 소프트웨어에서 1로 설정, WDT 초기화 후에 하드웨어가 0으로 소거.	0
MAP1	11	SFR/IFF 위한 주소 map 확장 비트 1.	0
MAP0	10	SFR/IFF 위한 주소 map 확장 비트 0. 향후 호환성을 위하여 이 flag를 1로 설정하지 말 것.	0
P4.2	9	P4의 개별 비트 설정/소거	1
P4.3	8	P4의 개별 비트 설정/소거	1
P3.3	7	P3의 개별 비트 설정/소거	1
P3.2	6	P3의 개별 비트 설정/소거	1
P3.1	5	P3의 개별 비트 설정/소거	1
P3.0	4	P3의 개별 비트 설정/소거	1
P2.3	3	P2의 개별 비트 설정/소거	1
P2.2	2	P2의 개별 비트 설정/소거	1
P2.1	1	P2의 개별 비트 설정/소거	1
P2.0	0	P2의 개별 비트 설정/소거	1

6.3. 명령어 세트 요약 (1/2)

Preliminary

- ◆ 자세한 것은 부록 A (명령어 세트)를 참고하라.

종류	명령	설명
Arithmetic	ADD A, #data INC A DEC A ADD A, @DP ADDC A, @DP SUB A, @DP INC @DP DEC @DP	Add data to ACC. Increment ACC. Decrement ACC. Add the indirect memory nibble to ACC. Add the indirect memory nibble to ACC with the Carry in C. Subtract the indirect memory nibble from ACC. Increment the indirect memory nibble. Decrement the indirect memory nibble.
Logical	CLR A CPL A RRC A ANL A, @DP ORL A, @DP XRL A, @DP	Clear ACC. Complement ACC. Rotate right ACC with Carry flag. Logical AND for ACC and the indirect memory nibble. Logical OR for ACC and the indirect memory nibble. Logical Exclusive-OR for ACC and the indirect memory nibble.
Data Transfer	MOV dir, A MOV A, dir MOV A, @DP MOV A, #data MOV L, @DP MOV @DP, A MOVI @DP, A MOVD @DP, A XCH A, @DP MOVI @DP, #data MOV L, #data MOV H, #data PUSH A POP A	Move ACC to the special function register. Move the special function register to ACC. Move the indirect memory nibble to ACC. Move data to ACC. Move the indirect memory nibble to DPL. Move ACC to the indirect memory nibble. Move ACC to the indirect memory nibble and increment the data pointer (DPH,DPL). Move ACC to the indirect memory nibble and decrement the data pointer (DPH,DPL). Exchange ACC and the indirect memory nibble. Move data to the indirect memory nibble and increment the data pointer (DPH,DPL). Move data to DPL. Move data to DPH. Push ACC to stack. Pop stack to ACC.

6.3. 명령어 세트 요약 (2/2)

Preliminary

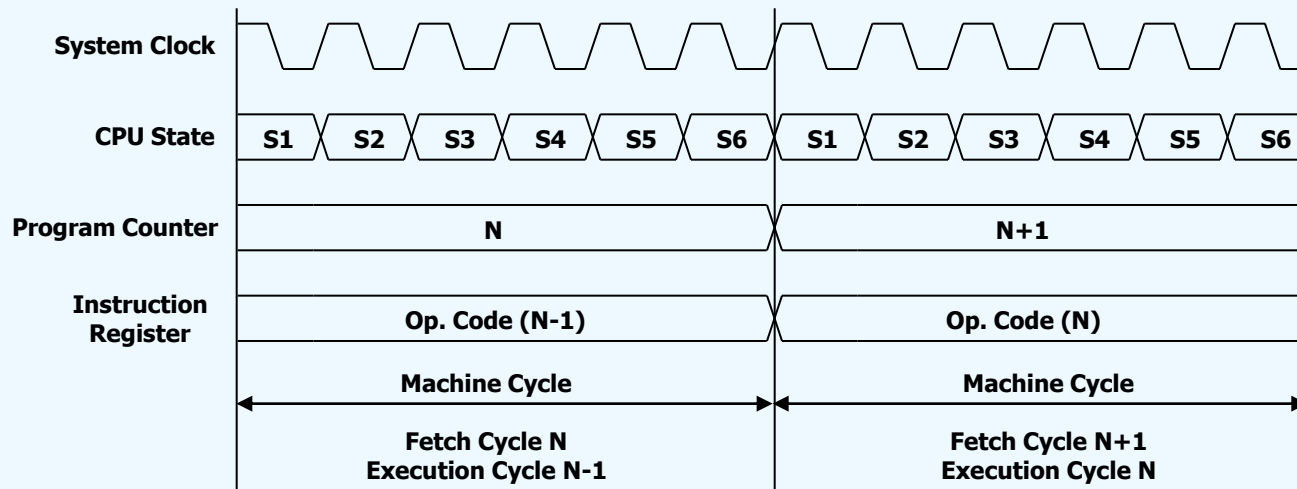
- ◆ 자세한 것은 부록 A (명령어 세트)를 참고하라.

Type	Instruction	Description
Branch	CJNE @DP, #data, rel CJNE L, #data, rel CJNE A, dir, rel CJNE A, @DP, rel CJLE A, @DP, rel CJNE A, #data, rel DJNZ A, rel JB bit, rel JNB bit, rel JC rel JNC rel JMP addr CALL addr RET NOP	Jump if the indirect memory nibble is not equal to the data. Jump if DPL is not equal to the data. Jump if ACC is not equal to the special function register. Jump if ACC is not equal to the indirect memory nibble. Jump if ACC is less than or equal to the indirect memory nibble. Jump if ACC is not equal to the data. Decrement ACC. Jump if the result is not zero. Jump if the indirect memory bit is 1. Jump if the indirect memory bit is 0. Jump if C is 1. Jump if C is 0. Jump to given address. Call subroutine. Return from subroutine. No operation.
Bit & Misc.	SETB @L CLR @L SETB bit CLR bit SETB C CLR C INC DPTR DEC DPTR	Set the indirect function flag. Clear the indirect function flag. Set the indirect memory bit. Clear the indirect memory bit. Set Carry flag. Clear Carry flag. Increment the data pointer. Decrement the data pointer.

6.4. CPU 타이밍

Preliminary

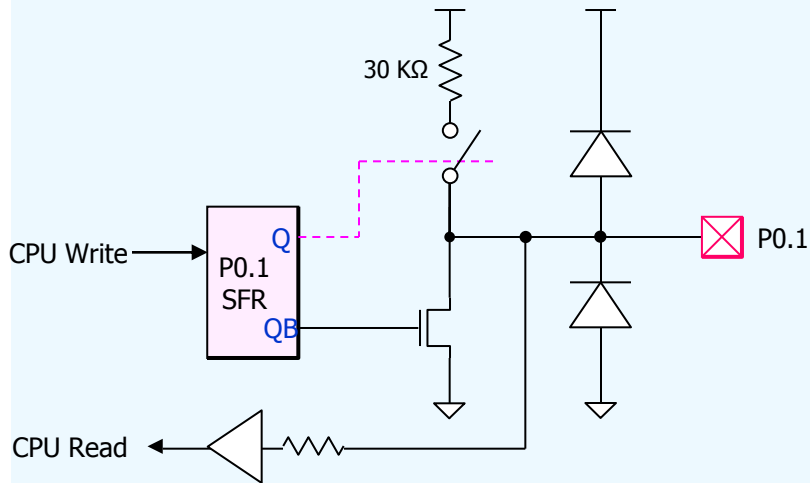
- ◆ CPU가 기계어 1주기를 실행하는데 6 주기의 클럭이 소요된다.
- ◆ 분기 명령어를 제외하면 모두 1 기계어 주기에 명령어 실행이 완성된다.
- ◆ 모든 분기 명령어는 분기를 하든지 안하든지 2 기계어 주기를 소모한다.
- ◆ SFR, 입출력 단자, IFF flag는 명령어 끝(S6)에서 변한다.



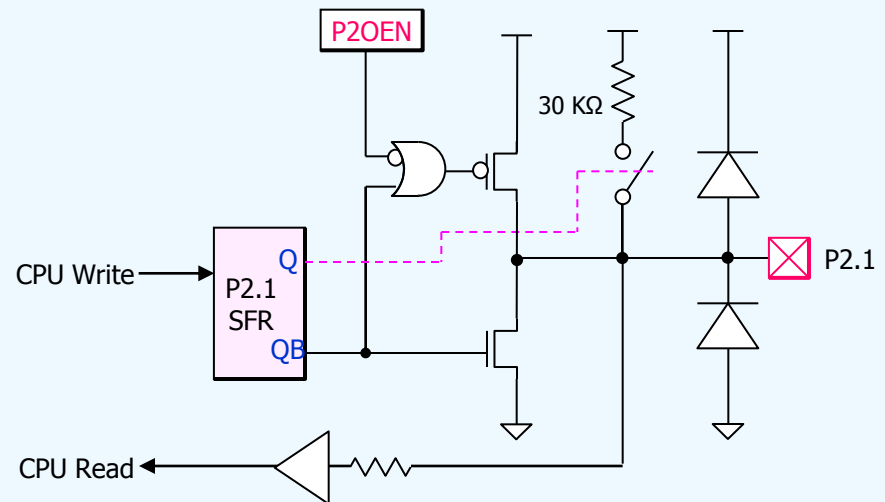
6.5. 입출력 단자 : PORT0 ~ PORT4

Preliminary

- ◆ 모든 단자는 전원이 켜질 때 비동기적으로 초기화된다.
- ◆ 기본지정(리셋)으로 pull-up 연결되고 입력으로 설정된다.
- ◆ Open drain 방식으로 active low 출력.
- ◆ P2[3:0]은 push-pull 출력 단자로 설정될 수 있다.
- ◆ CPU는 항상 SFR 레지스터에 쓰지만, 단자 핀을 읽는다.
- ◆ Stop mode 또는 sleep mode에서 전 상태를 유지한다.



P0[3:0], P1[3:0], P3[3:0], P4[3:2] 회로



P2[3:0] 회로

6.5. 입출력 단자 : PORT4[1:0] (XI/XO)

Preliminary

- ◆ 클럭 신호 입출력을 위한 XI/XO
 - ✓ CKCFG SFR의 XT/RG 비트가 1이 되면 인에이블.
 - ✓ STOP mode에서 디세이블 (XI와 XO는 논리 0 상태).
- ◆ 입출력 단자로서 XI/XO
 - ✓ IOCFG SFR의 IOXEN 비트가 1이 되면, XI와 XO는 can 입출력 단자로 설정될 수 있음.
 - ✓ 사용자는 XT/RG와 IOXEN를 동시에 1로 설정하지 말아야 함.
 - ✓ 기본지정(리셋)으로 pull-up 인에이블과 입력.
 - ✓ Open drain 방식으로 active low 출력.
 - ✓ CPU는 항상 SFR 레지스터에 쓰지만, 단자 핀을 읽는다.
 - ✓ Stop mode에서 전 상태를 유지함.

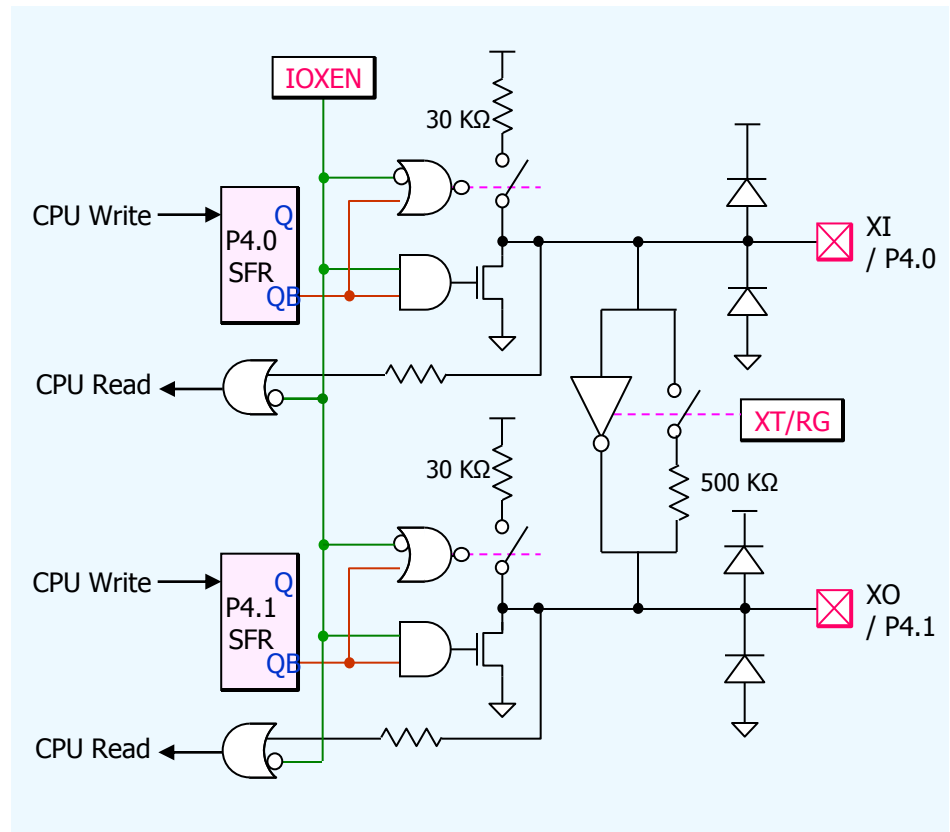
✓ **IOCFG (0Eh) : 입출력 단자 구조 레지스터**

IOMAP1	IOMAP0	P2OEN	IOXEN
R/W(0)	R/W(0)	R/W(0)	R/W(0)

- IOXEN : XI와 XO를 입출력 단자로 인에이블.
0 = XI와 XO는 클럭 입력으로 사용됨 (기본지정).
1 = XI와 XO는 PORT4[1:0]를 위하여 사용됨.
- P2OEN : P2를 push-pull 출력 단자로 설정함.
- IOMAP[1:0] : 입출력 단자 대응을 설정.

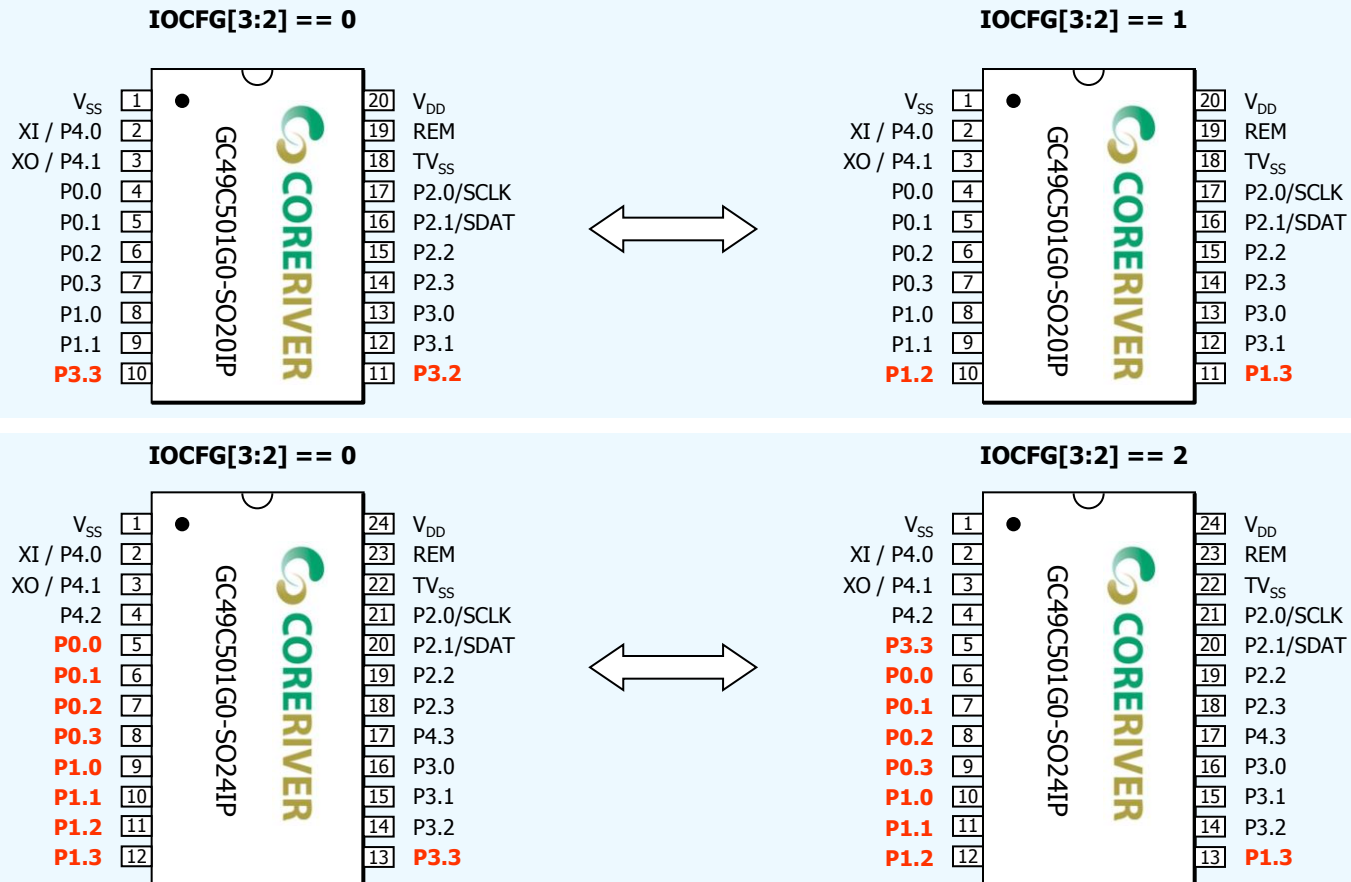
IOMAP1	IOMAP0	입출력 대응
0	0	기본 지정.
0	1	20핀 입출력 단자 대응 옵션
1	0	24핀 입출력 단자 대응 옵션
1	1	Reserved

- ◆ IOCFG
 - ✓ 이 SFR은 power-on-reset에 의하여 기본지정으로 초기화된다. 단자 P2OEN 비트는 다른 리셋으로 0으로 소거된다.
 - ✓ 8 핀 패키지에서는 오직 P2OEN만 사용가능하다. 다른 비트를 설정하지 말아야 한다.



6.5. 입출력 단자 : 입출력 대응

- ◆ 사용자는 IOCFG SFR의 값을 설정함에 따라 입출력단자의 대응을 선택할 수 있다.
- ◆ 각 I/O 핀들의 기능은 어떤 대응에 대해서도 같다.
- ◆ 이 대응방식 옵션은 기존의 디바이스와 핀들 간의 호환성이 필수적일 때 유용하다.



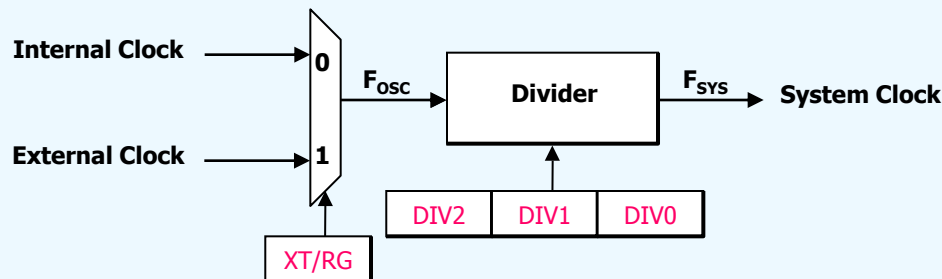
6.6. 클럭 설정

- ◆ 두개의 시스템 클럭 발생원 : 내부 링 발진기 또는 외부 Resonator/Crystal
- ◆ 기본 시스템 클럭은 링 발진기이다.
- ◆ 사용자가 클럭 발생원을 교체(XT/RG 비트)를 교체하면, 내부 리셋이 발생한다.
- ◆ 내부 리셋은 CKCFG 레지스터에 영향을 미치지 않는다.
- ◆ CKCFG SFR의 상태는 power-on reset에 의하여 초기화된다.
- ◆ 사용자는 분주 조건을 변경함으로써 동작중에 클럭 주파수를 바꿀 수 있다.

✓ **CKCFG (0Dh) : 클럭 상태 레지스터.**

XT/RG	DIV2	DIV1	DIV0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

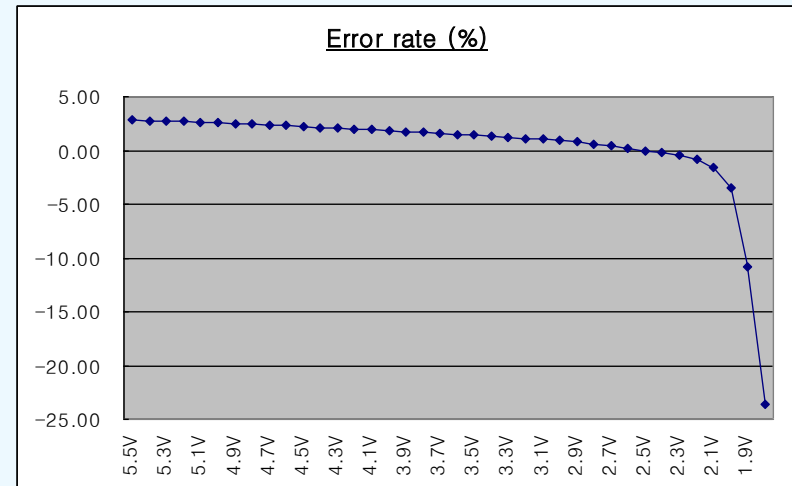
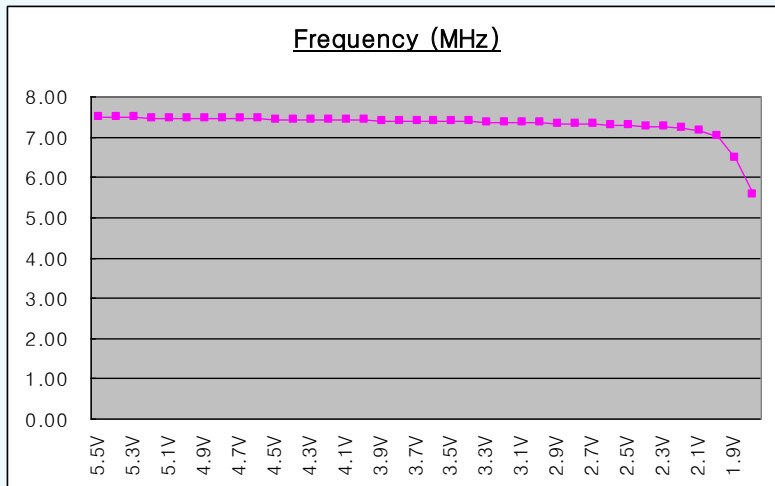
- **XT/RG** : 시스템 클럭 발생원 선택.
 0 = 시스템 클럭으로 내부 링 발진기를 선택.
 외부 클럭 발진기는 디세이블됨.
 1 = 시스템 클럭으로 외부 클럭 선택.
 내부 링 발진기는 디세이블됨.
 8 핀 패키지에서 이 비트를 사용하지 말라.
- **DIV[2:0]** : 시스템 클럭 분주비 선택.



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	-

6.6. 클럭 구조 : 내부 링 발진기

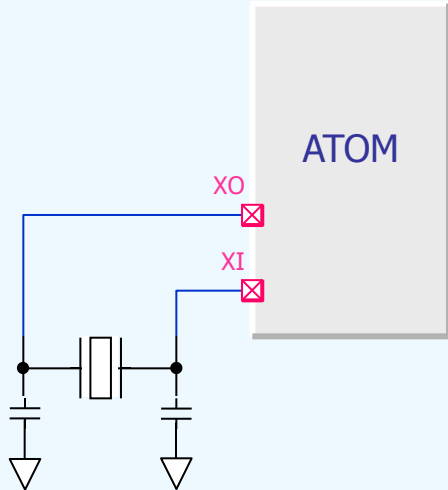
- ◆ 내부 링 발진기는 고정된 시스템 클럭을 공급한다.
 - ✓ 2.1V ~ 3.3V에서 $\pm 3\%$ 로 조정됨.
 - ✓ 2.5V에서 $\pm 1\%$ 로 조정됨.
- ◆ 7.28MHz로 조정된 결과는 단지GC49C501RX 디바이스들에 적용됨.



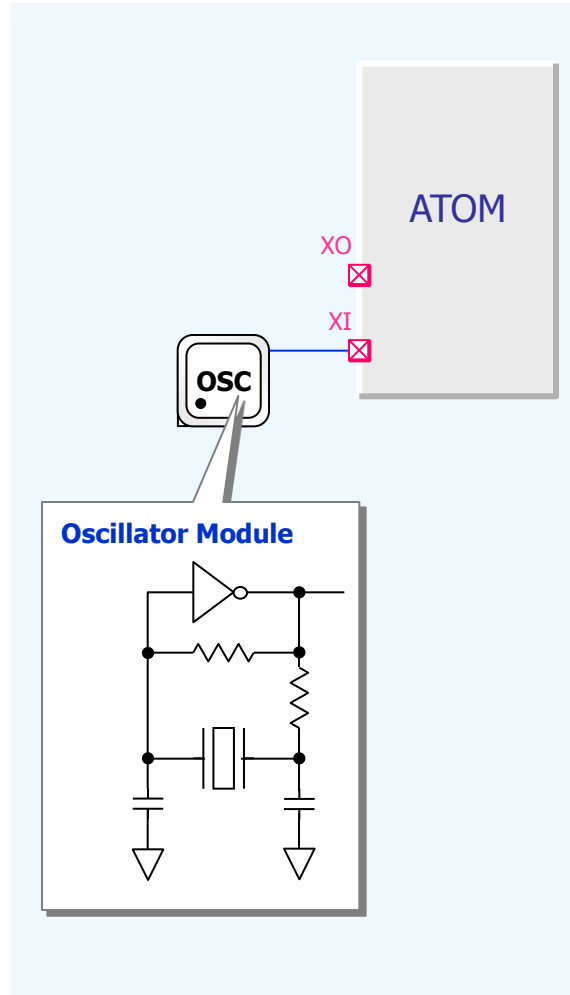
VOLTAGE-FREQUENCY GRAPH

6.6. 클럭 구조: Guideline

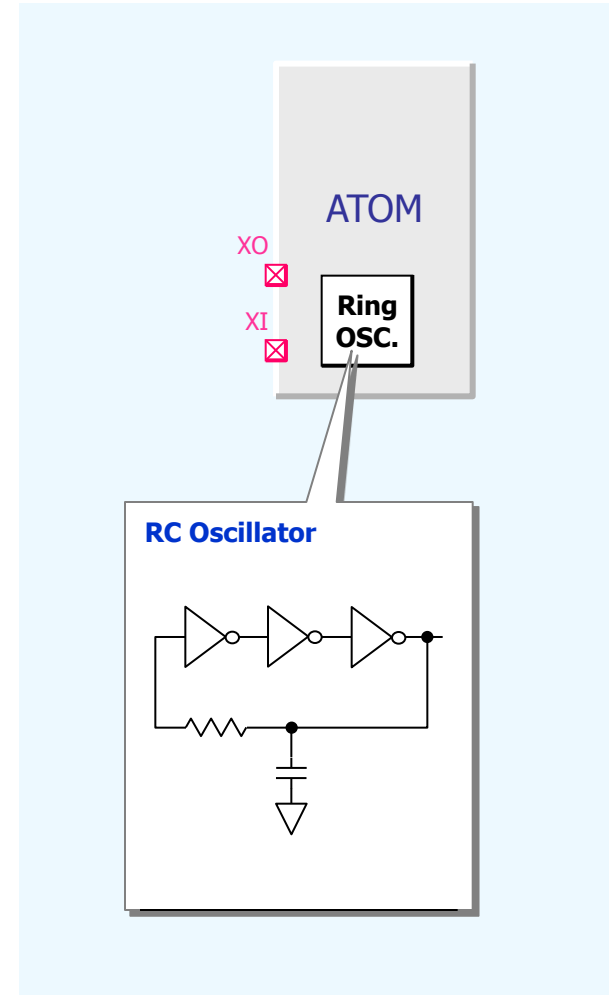
◆ Resonator / Crystal Oscillator



◆ Oscillator Module



◆ Internal Ring Oscillator



6.7. Carrier 주파수 생성

Preliminary

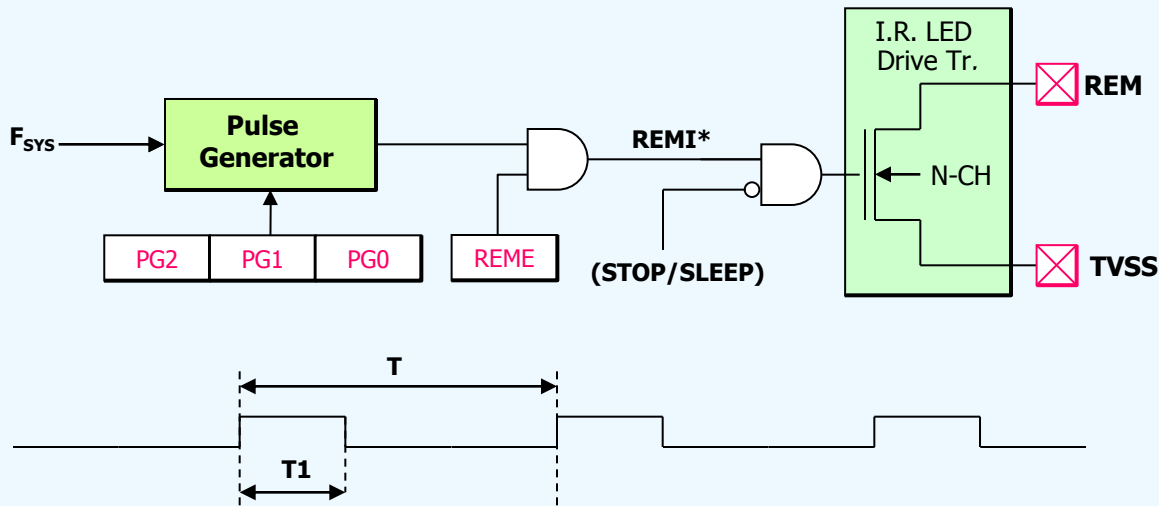
◆ 7 가지의 carrier 주파수 지원.

✓ **REMC** (05h) : REM 출력 제어 레지스터.

REME	PG2	PG1	PG0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

- PG[2:0] : Carrier 주파수 선택.
- REME : REM 출력 인에이블.

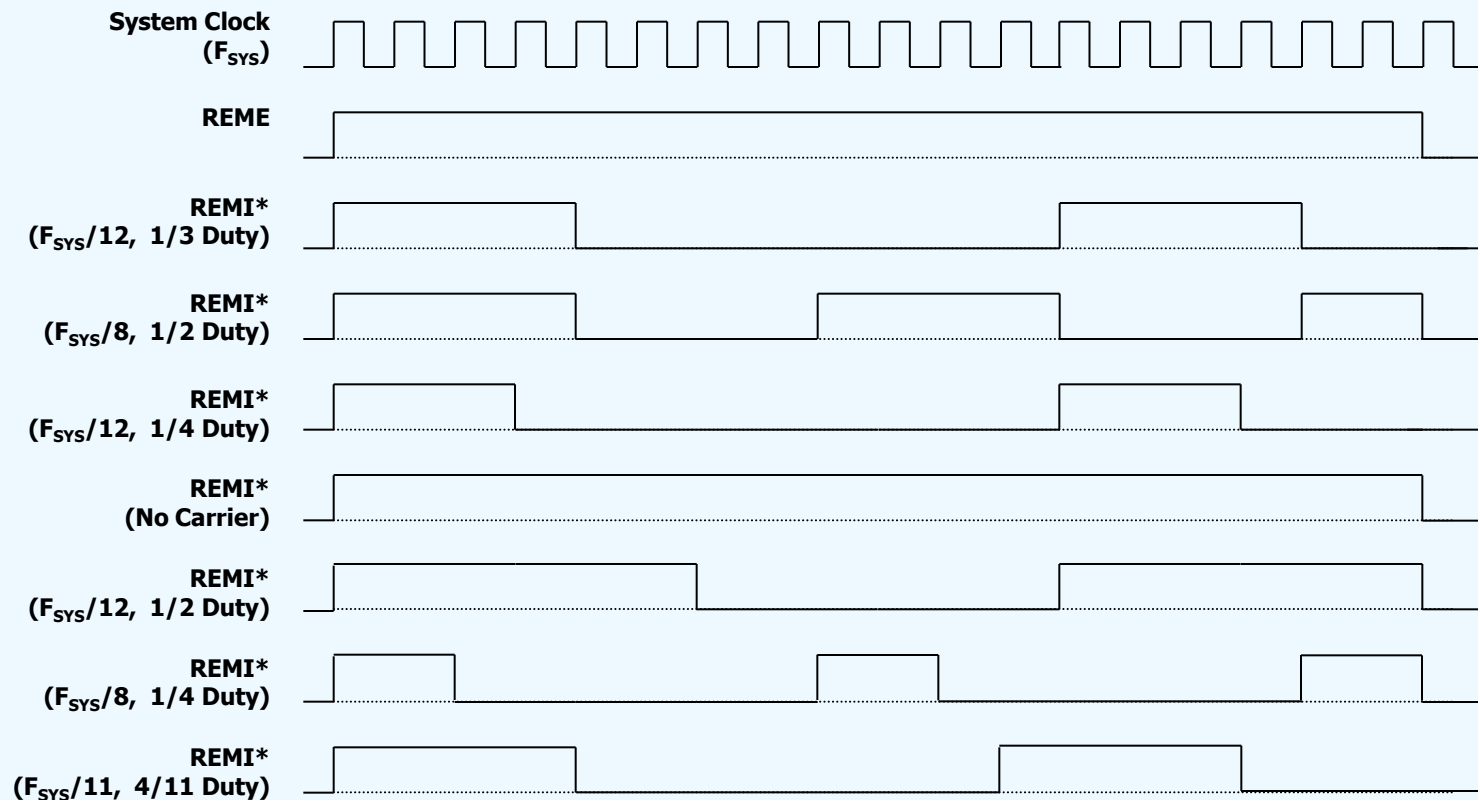
REME	PG2	PG1	PG0	Transmission Control (REMI)
0	X	X	X	0 (Disable)
1	0	0	0	$1/T = F_{SYS}/12, T1/T = 1/3$
1	0	0	1	$1/T = F_{SYS}/8, T1/T = 1/2$
1	0	1	0	$1/T = F_{SYS}/12, T1/T = 1/4$
1	0	1	1	1 (No Carrier)
1	1	0	0	$1/T = F_{SYS}/12, T1/T = 1/2$
1	1	0	1	$1/T = F_{SYS}/8, T1/T = 1/4$
1	1	1	0	$1/T = F_{SYS}/11,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



6.7. Carrier 주파수 생성

◆ 파형 예제

- ✓ REM 출력은 REMI*가 반전되어 나타난다.
- ✓ ATOM의 적외선 LED 구동 tr.은 N-형이므로, 적외선 LED는 REMI*가 high일 때 켜진다.



6.8. POR & LVD : 저전압 검출회로

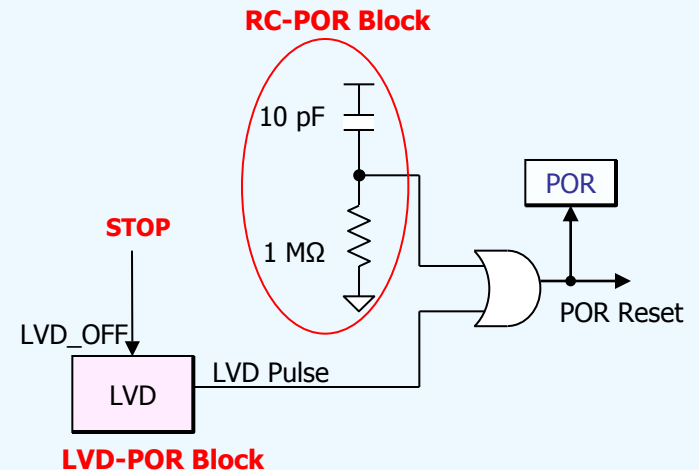
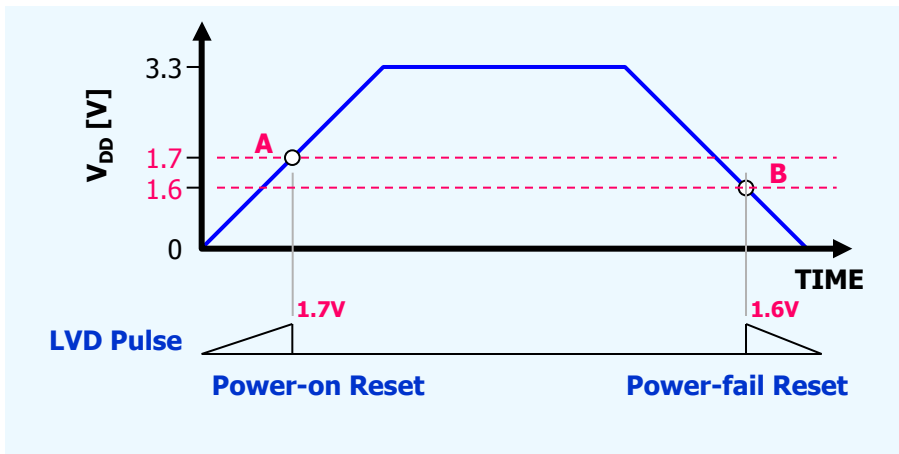
Preliminary

- ◆ 내부 Power-On-Reset은 RC-POR과 LVD-POR의 논리합으로 구성된다.
- ◆ RC-POR은 전원이 짧은 시간안에 상승할 때 동작한다.
- ◆ LVD
 - ✓ 전원의 상승시간이 상대적으로 긴 경우 POR를 발생한다.
 - ✓ POR 전압 : 1.7V
 - ✓ 전원이 1.6V일때 Power-fail-reset을 발생한다.
- ◆ POR 펄스가 사라진 후에, 내부 클럭의 안정을 위해 카운터가 동작하여, power-on 리셋을 약 4.5 ms 동안 연장한다.

✓ LVCFG (0Fh) : LVD Configuration Register

POR	Reserved	Reserved	Reserved
R/W(1)	R(X)	R/W(0)	R/W(0)

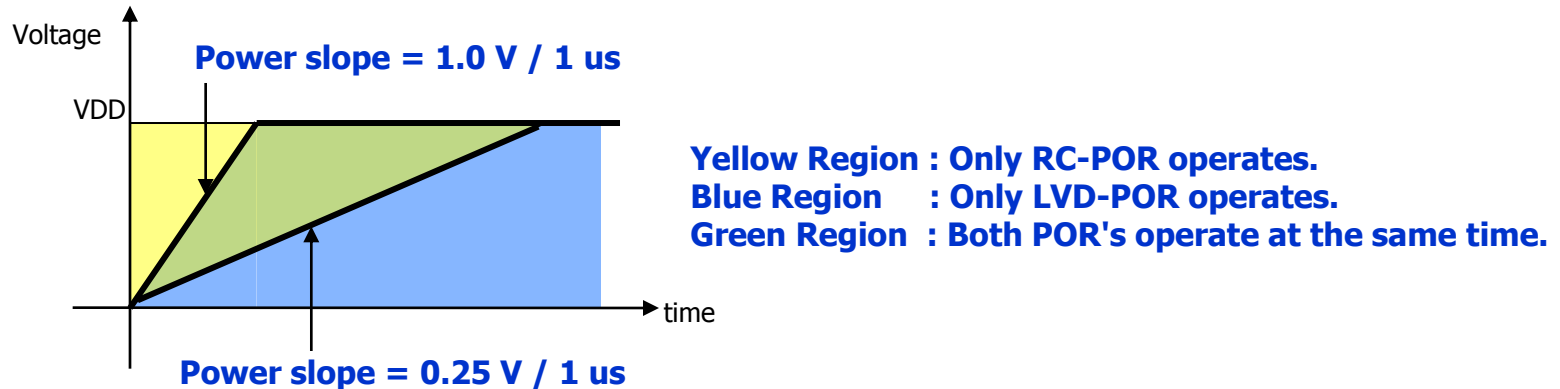
- Reserved: 향후 호환성을 위해 이 비트를 설정하지 말라.
- POR : Power-on-reset Flag.
이 비트를 사용하여 Power-On-Reset을 다른 reset과 구분할 수 있다. 이 비트를 접근할 때 reserved bits들은 AND operation으로 수정되지 않도록 막아야 한다.



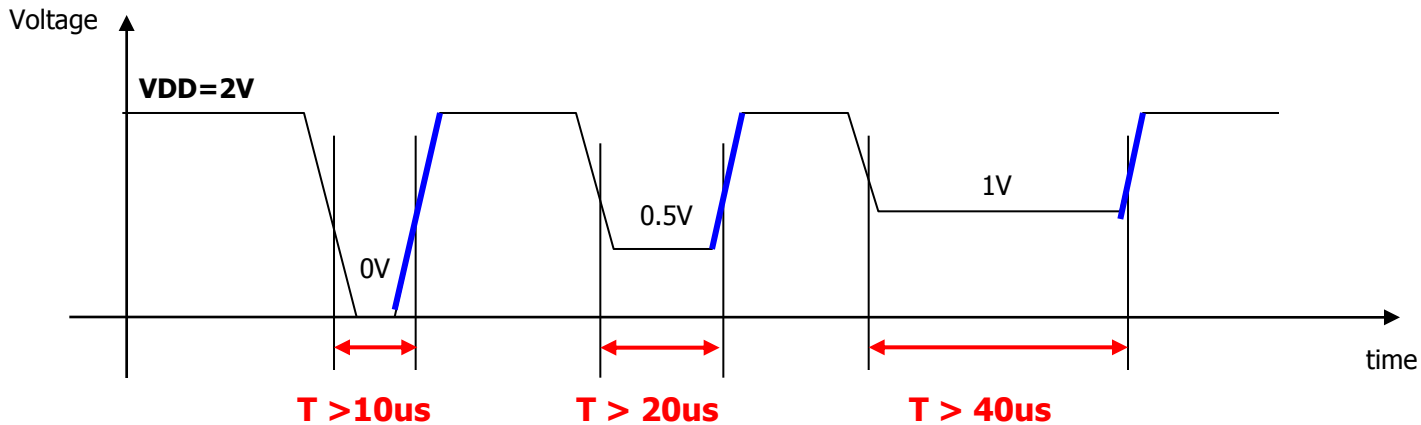
6.8. POR & LVD : 전력 notch 조건

Preliminary

- ◆ Power-on-reset is independent of power-rising slope.



- ◆ The cases of reset generation by VDD notch



When VDD fails for a short time, the duration of notch (T) has limitation like above for the successful POR operation.

The duration (T) will be changed by the VDD value and the transition time

6.9. WDT (Watchdog Timer)

Preliminary

◆ WDT

- ✓ 2¹⁷ 시스템 클럭 주기마다 CPU를 리셋하는 자동 진행 카운터.
- ✓ 카운터의 길이는 고정되었지만, WDT 오버플로의 주기는 현재 시스템 클럭의 주파수에 따라 변한다.
- ✓ WDT는 STOP mode에서 정지하거나 사용자에게 의하여 디세이بل된다.

◆ WDT 초기화 방법

- ✓ 소프트웨어에서 IFF[12]안에 WDTR 비트를 설정. WDTR 비트는 WDT를 초기화하고 하드웨어에 의하여 자동적으로 소거됨.
- ✓ 어떤 발생원에 의하여 내부 리셋이 발생하면 초기화됨.
- ✓ SLEEP 모드에 들어 감.
- ✓ IAP에 의해 FLASH programming (erase/write)이 시작됨.

[WDT 주기 예]

XT/RG	DIV2	DIV1	DIV0	F _{osc} (MHz)	F _{sys}	WDT Period (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

◆ WDT의 동작 제어

- ✓ IFF[13]의 WDTE flag가 0으로 소거되면, WDT가 디세이بل된다.
- ✓ WDT가 디세이بل되면 WDT의 전 상태가 유지된다.
- ✓ 오직 IFF[11]의 MAP1 flag를 설정하고 IFF[10]의 MAP0 flag를 소거하여 WDTE를 수정할 수 있다.
- ✓ WDTE는 내부 리셋에 의해서 또는 IFF[14]의 SLEEP flag를 설정하거나 IAPCON SFR을 쓸 때 하드웨어에 의하여 설정된다.

◆ WDT를 디세이블시키기 위한 프로그램 순서

```

MOV L, #11
SETB @L      ; Enable MAP1
MOV L, #13
CLR @L       ; Disable WDT
MOV L, #11
CLR @L       ; Disable MAP1
    
```

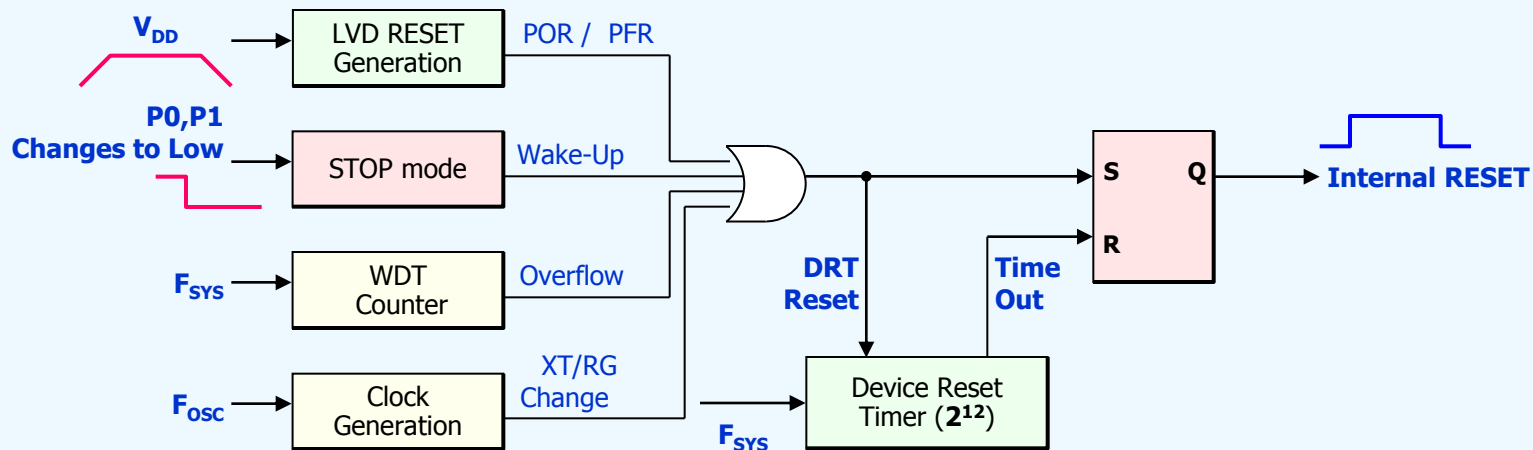
6.10. 리셋 회로

◆ 리셋 발생원

- ✓ 전원이 켜질 때 Power-on Reset (POR).
- ✓ 전력-오류 Reset
- ✓ 입력단자 P0와 P1의 변화가 발생하여 STOP mode 탈출.
- ✓ 비정상 상태로 인한 WDT 오버플로 SLEEP mode.
- ✓ 클럭 발생원 변화 (CKCFG[3]의 상태 변화).

◆ 디바이스 리셋 타이머

- ✓ 내부 리셋이 발생하면 DRT(디바이스 리셋 타이머)가 만료될 때까지 high 상태를 유지한다.
- ✓ 리셋 시간은 CKCFG SFR의 시스템 클럭 설정에 달려 있다.
- ✓ 예로서, F_{sys} 가 455 KHz이면 2^{12} 주기는 9 ms이다.
- ✓ CKCFG는 내부 리셋에 의하여 영향을 받지 않는다.
- ✓ Power-on reset의 경우에 리셋 시간은 약 4.5 ms이다.



6.11. 전력 관리 : 3 모드

◆ Active Mode

- ✓ CPU와 주변 회로는 동작한다.

◆ Sleep Mode

- ✓ 오직 WDT만 동작한다.
- ✓ 입출력 단자는 **sleep mode**의 전 상태를 유지한다.
- ✓ WDT 오버플로에 의하여 깨어남.
- ✓ WDT 오버플로 시간은 내부 링 클럭이 사용될 때 최대 1.1초가 된다.
- ✓ 디바이스는 리셋된다.

◆ Stop Mode

- ✓ 외부 클럭 발진기를 포함한 디바이스의 모든 기능이 정지한다.
- ✓ 입출력 단자들은 **stop mode** 전 상태를 유지한다.
- ✓ P0, P1 입력핀에 변화가 있으면 깨난다.
- ✓ 디바이스는 리셋된다.

6.12. In Application Programming (IAP)

◆ In Application Programming

- ✓ 소프트웨어는 사용중에 IAP 기능으로 FLASH의 특정한 영역을 읽고 수정할 수 있다.
- ✓ EEPO/1 영역은 프로그램 또는 데이터 메모리로 사용될 수 있다.
- ✓ CPU는 IAP 동안 정지되고 IAP 이후에 IAPCON을 설정한 다음의 명령어를 계속 수행한다.
- ✓ IAP로 한 바이트를 읽는데 6 클럭 주기가 걸린다.
- ✓ IAP로 한 바이트를 쓰거나 지우는데 약 2ms 걸린다.
- ✓ IAPCON에 쓸 때, IFF[13]인 WDTE 비트도 또한 1로 설정된다.
- ✓ 쓰거나 소거의 IAP 동작이 실행되면, 프로그래밍이 시작되기 전에 WDT가 초기화된다.

◆ IAP 관련 SFR

- ✓ DPH / DPL : IAP의 LSB 6-비트 주소.
- ✓ GDH / GDL : IAP에 의하여 읽거나 쓰는 8비트 데이터 버퍼.
- ✓ IAPCON : IAP 제어 SFR. IAP 실행 후에 자동적으로 0으로 소거됨.

◆ IAP 인에이블 조건

- ✓ IAP로 INFO 영역을 지우거나 쓸 수 없다.
- ✓ IAPCON는 단지 다음의 경우에 쓸 수 있다.
 - IFF[10]인 MAP0 비트가 소거되고,
 - IFF[11]인 MAP1 비트가 설정되고,
 - CFGWD[2:1]의 상응 비트가 설정되었을 때.
- ✓ 위의 조건에 의해 IAP가 억제되면, "MOV IAPCON, A" 명령어는 "NOP" 명령어처럼 작용한다.

✓ IAPCON (09h) : IAP 제어 레지스터

RGS1	RGS0	OPS1	OPS0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

- RGS[1:0] : IAP 영역 선택
- OPS[1:0] : IAP 기능 선택

RGS1	RGS0	IAP 영역
0	0	EEP0 (0x1C0 ~ 0x1FF)
0	1	EEP1 (0x3C0 ~ 0x3FF)
1	0	INFO (0x0 ~ 0x7)
1	1	Reserved

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

6.12. In Application Programming (IAP)

Preliminary

◆ IAP의 전기적 특성

- ✓ 프로그램 시간은 시스템 클럭의 주파수에 달려 있다.
- ✓ 시스템 클럭 주파수가 IAP 범위를 벗어나 있으면, IAP 전후에 CKCFG SFR을 설정하여 F_{SYS}를 변경할 필요가 있다.

Parameter	Symbol	MIN	TYP	MAX	Unit
Power Supply Voltage	V _{DD}	2.7	-	5.5	V
시스템 클럭 주파수	F _{SYS}	5	8	11	MHz
기록 / 소거 시간	T _p	1.5	2.0	3.3	ms

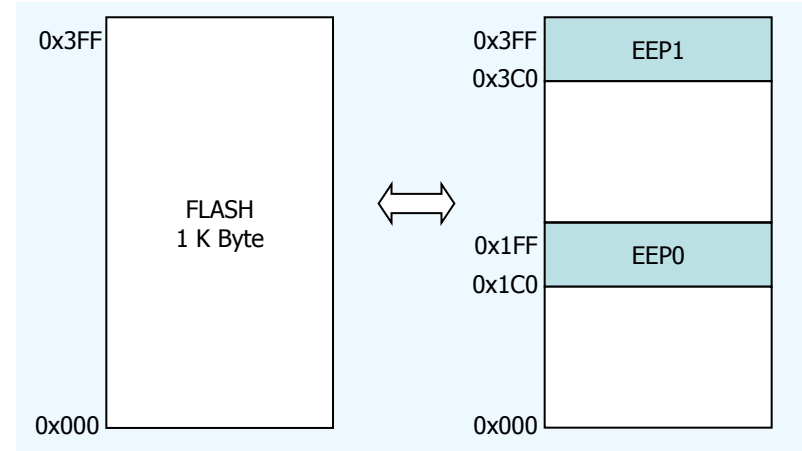
◆ 정보 영역

ADDRESS	0	1	2	3	4	5	6	7
Mnemonic	CFGWD							

- ✓ 첫 바이트는 CFGWD를 포함한다.
- ✓ 사용자 ID, 또는 checksum 등을 저장하기 위하여 사용할 수 있다.
- ✓ 단지 ISP의 전체 칩 소거 기능만 이 영역을 소거할 수 있다.

◆ FLASH 영역

- ✓ EEPROM 영역은 프로그램 메모리의 일부분이다.



◆ CFGWD : Configuration Word

- ✓ CFGWD[0] (ISP_LOCK) : 전체 칩 소거를 제외한 ISP에 의한 읽기, 쓰기, 또는 소거 금지
- ✓ CFGWD[1] (IAP_RE) : IAP에 의한 읽기 인에이블.
- ✓ CFGWD[2] (IAP_PE) : IAP에 의한 쓰기 또는 소거 인에이블.

7. Absolute Maximum Ratings

◆ Absolute Maximum Ratings

Symbol	Parameter	Rating	Unit
V_{DD}	DC supply voltage	-0.5 to 6.5	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 pin active : -100	mA
I_{OL}	DC output low current	One I/O pin active : 30	mA
		All I/O pin active : 150	mA
T_{STG}	Storage temperature	-55 to 125	°C

◆ 추천 동작 조건

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 특성

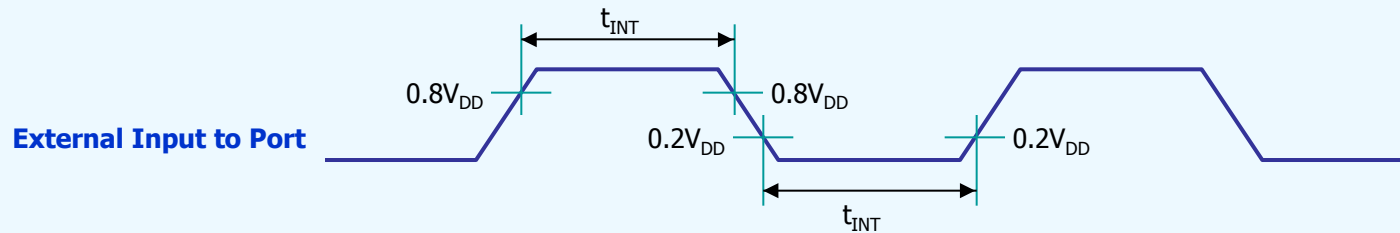
* 별다른 언급이 없으면 $TA = -40\text{ }^{\circ}\text{C} \sim +85\text{ }^{\circ}\text{C}$, $V_{DD} = 1.8\text{V} \sim 5.5\text{V}$.

Parameter	Symbol	Pin	Conditions	Value			Unit
				Min.	Typ.	Max.	
Input Low Voltage	V_{IL1}	P0, P1, P2, P3, P4.3, P4.2	$V_{DD} = 1.8\text{V} \sim 5.5\text{V}$	-0.5	-	$0.2V_{DD} - 0.1$	V
	V_{IL2}	XI / P4.0, XO / P4.1		-0.5	-	$0.3V_{DD}$	
Input high Voltage	V_{IH1}	P0, P1, P2, P3, P4.3, P4.2	$V_{DD} = 1.8\text{V} \sim 5.5\text{V}$	$0.2V_{DD} + 1.0$	-	$V_{DD} + 0.5$	V
	V_{IH2}	XI / P4.0, XO / P4.1		$0.7V_{DD}$	-	$V_{DD} + 0.5$	
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, P4	$I_{OL} = 20\text{mA} @ V_{DD} = 5\text{V}$ ($I_{OL} = 3\text{mA} @ V_{DD} = 2.2\text{V}$)	-	-	$0.3V_{DD}$	V
Output Low Voltage	V_{OL2}	REM	$I_{OL2} = 280\text{mA} @ V_{DD} = 3\text{V}$	-	-	0.4	V
Output High Voltage	V_{OH}	P2 (Configured as push-pull output)	$I_{OH} = -15\text{mA} @ V_{DD} = 5\text{V}$ ($I_{OH} = -2\text{mA} @ V_{DD} = 2.2\text{V}$)	$0.7V_{DD}$	-	-	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_{I0}	All	$V_{DD} = 5\text{V}$	-	10	-	pF

9. AC 특성

* 별다른 언급이 없다면, $TA = -40\text{ }^{\circ}\text{C} \sim +85\text{ }^{\circ}\text{C}$. TBD = To Be Determined.

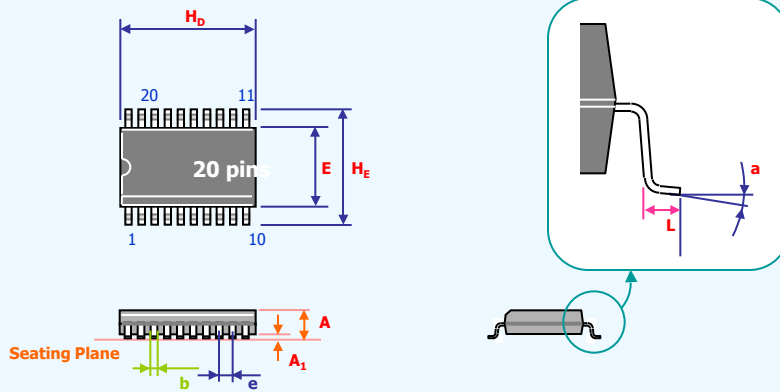
Parameter	Symbol	Pin	Conditions	Value			Unit
				Min.	Typ.	Max.	
Oscillator Frequency (Internal Clock)	F_{OSC}		$2.7\text{ V} \leq V_{DD} \leq 5.5\text{ V}$	-	-	10	MHz
			$1.8\text{ V} \leq V_{DD} \leq 2.7\text{ V}$	-	-	5	
Oscillator Frequency (External Clock)	F_{OSC}	XI, XO	$2.7\text{ V} \leq V_{DD} \leq 5.5\text{ V}$	-	-	10	MHz
			$1.8\text{ V} \leq V_{DD} \leq 2.7\text{ V}$	-	-	5	
System Frequency	F_{SYS}		$1.8\text{ V} \leq V_{DD} \leq 5.5\text{ V}$	1/64	-	1	F_{OSC}
External Input Width	t_{INT}	P0, P1, P2, P3, P4	$1.8\text{ V} \leq V_{DD} \leq 5.5\text{ V}$	6	-	-	F_{SYS}



10. Package Dimensions : 20-SOIC(JEDEC)

Preliminary

[20-SOIC (JEDEC)]



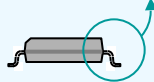
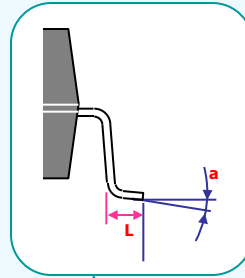
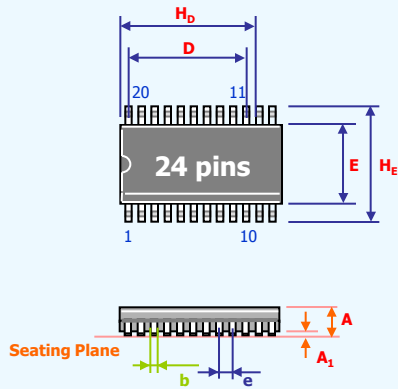
Symbol	Dimension in Inches			Dimension in mm		
	Min.	Nom.	Max.	Min.	Nom.	Max.
A	-	-	0.106	-	-	2.7
A ₁	0.004	-	-	0.1	-	-
b	0.013	0.016	0.020	0.324	0.4	0.51
E	0.264	0.295	0.324	6.71	7.5	8.23
H _D	0.495	0.504	0.512	12.57	12.8	13
H _E	0.394	0.406	0.419	10.0	10.3	10.643
L	0.016	-	0.052	0.406	-	1.32
a	0 ^ø	-	8 ^ø	0 ^ø	-	8 ^ø
e	0.050 BSC			1.27 BSC		

Notes:

1. Dimension D & E include mold mismatch and are determined at the mold parting line.
2. General appearance spec. should be based on final visual inspection spec.

10. Package Dimensions : 24-SOIC

Preliminary



[24-SOIC]

Symbol	Dimension in Inches			Dimension in mm		
	Min.	Nom.	Max.	Min.	Nom.	Max.
A	0.093	0.099	0.104	2.40	2.50	2.60
A_1	0.004	0.008	0.012	0.10	0.20	0.30
b	0.014	0.017	0.019	0.36	0.42	0.49
D	-	0.550	-	-	13.97	-
E	0.291	0.295	0.299	7.40	7.50	7.60
H_b	0.598	0.606	0.614	15.20	15.40	15.60
H_e	0.398	0.406	0.413	10.10	10.30	10.50
L	0.004	0.010	0.016	0.10	0.25	0.40
a	0°	-	8°	0°	-	8°
e	0.050 BSC			1.27 BSC		

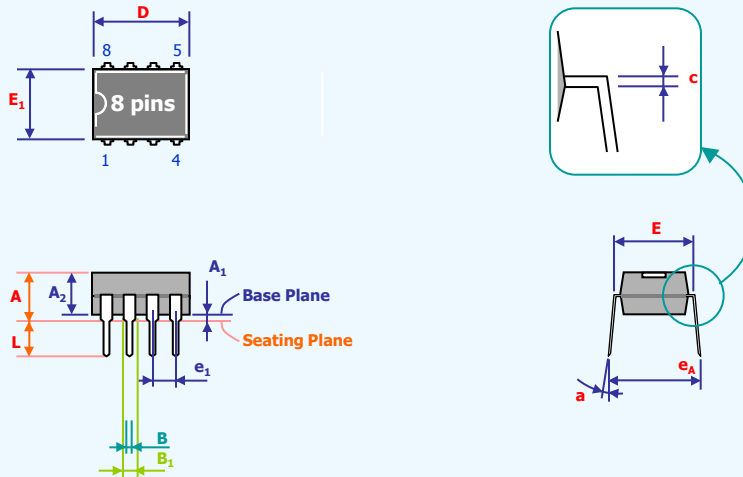
Notes:

1. Dimension D & E include mold mismatch and are determined at the mold parting line.
2. General appearance spec. should be based on final visual inspection spec.

10. Package Dimensions : 8-SPDIP/SOIC

Preliminary

[8-SPDIP]

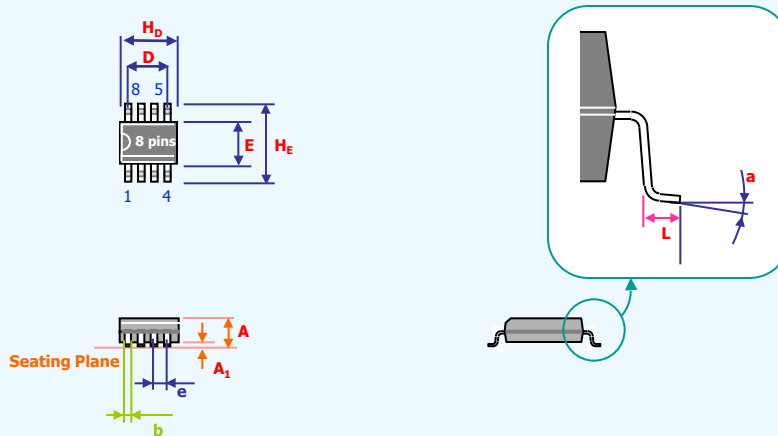


Symbol	Dimension in Inches			Dimension in mm		
	Min.	Nom.	Max.	Min.	Nom.	Max.
A	-	0.155	-	-	3.93	-
A ₁	0.015	-	-	0.380	-	-
A ₂	-	0.140	-	-	3.55	-
B	0.015	0.019	0.022	0.38	0.47	0.56
B ₁	0.050	0.057	0.065	1.27	1.46	1.65
c	0.008	0.011	0.014	0.20	0.28	0.36
D	0.367	0.377	0.387	9.33	9.58	9.83
E	-	0.300	-	-	7.60	-
E ₁	0.240	0.250	0.260	6.10	6.35	6.60
e ₁	-	0.100	-	-	2.54	-
L	0.120	0.130	0.140	3.05	3.30	3.55
a	0 ⁸	-	15 ⁸	0 ⁸	-	15 ⁸
e _A	0.330	0.350	0.370	8.382	8.89	9.398

Notes:

1. Dimension D Max. & S include mold flash or tie bar Burns.
2. Dimension E₁ dose not include interlead flash.
3. Dimension D & E₁ include mold mismatch and are determined at the mold parting line.
4. Dimension B₁ does not include dambar protrusion/intrusion.
5. General appearance spec. should be based on final visual inspection spec.

[8-SOIC]



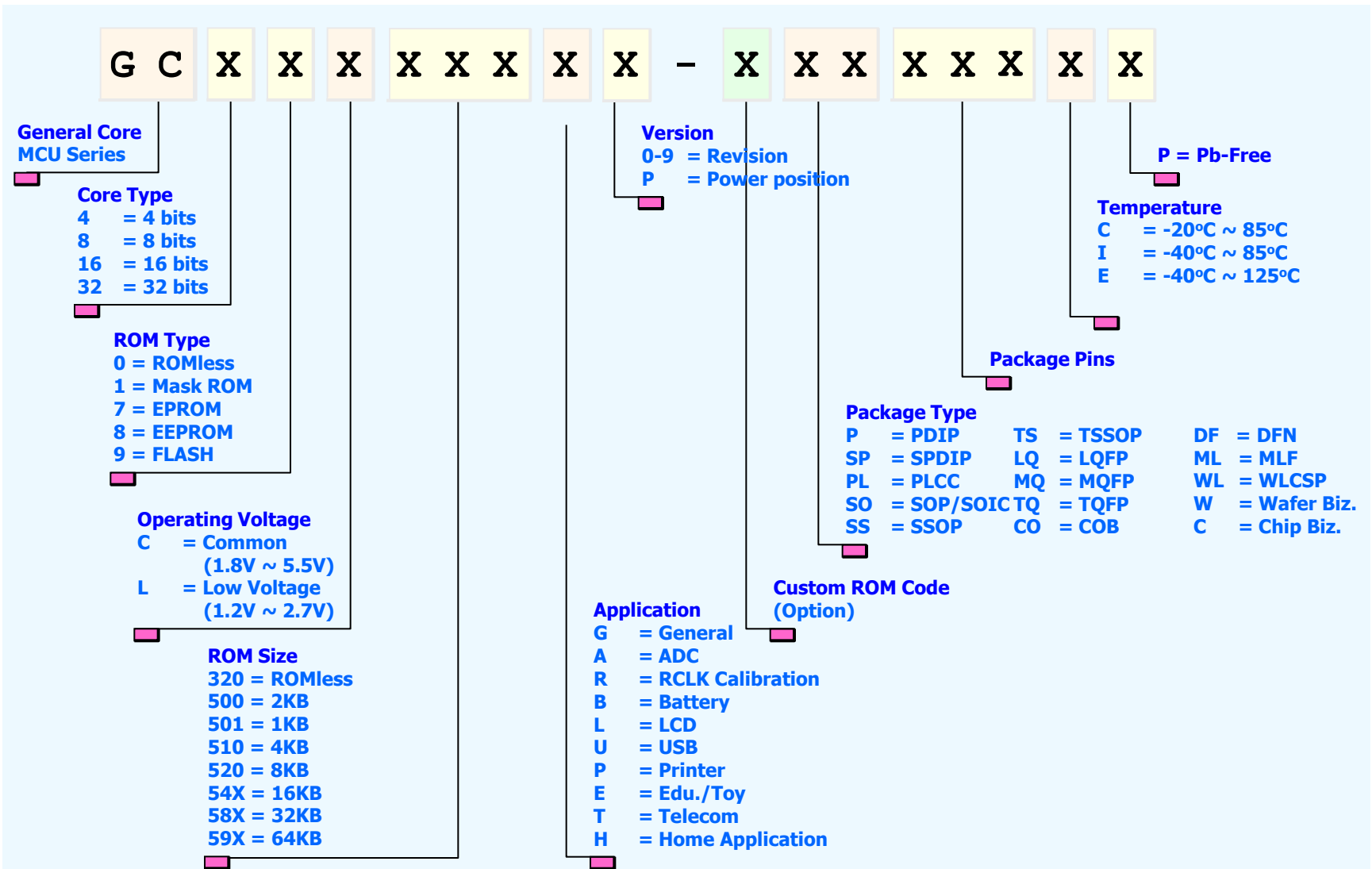
Symbol	Dimension in Inches			Dimension in mm		
	Min.	Nom.	Max.	Min.	Nom.	Max.
A	0.068	0.072	0.075	1.73	1.82	1.90
A ₁	0.004	0.007	0.010	0.10	0.18	0.26
b	0.012	0.016	0.020	0.31	0.41	0.51
D	-	0.150	-	-	3.81	-
E	0.146	0.154	0.161	3.70	3.90	4.10
H _b	0.185	0.193	0.201	4.70	4.90	5.10
H _E	0.224	0.236	0.248	5.70	6.00	6.30
L	0.017	0.026	0.035	0.42	0.65	0.88
a	0 ⁸	-	8 ⁸	0 ⁸	-	8 ⁸
e	0.050 BSC			1.27 BSC		

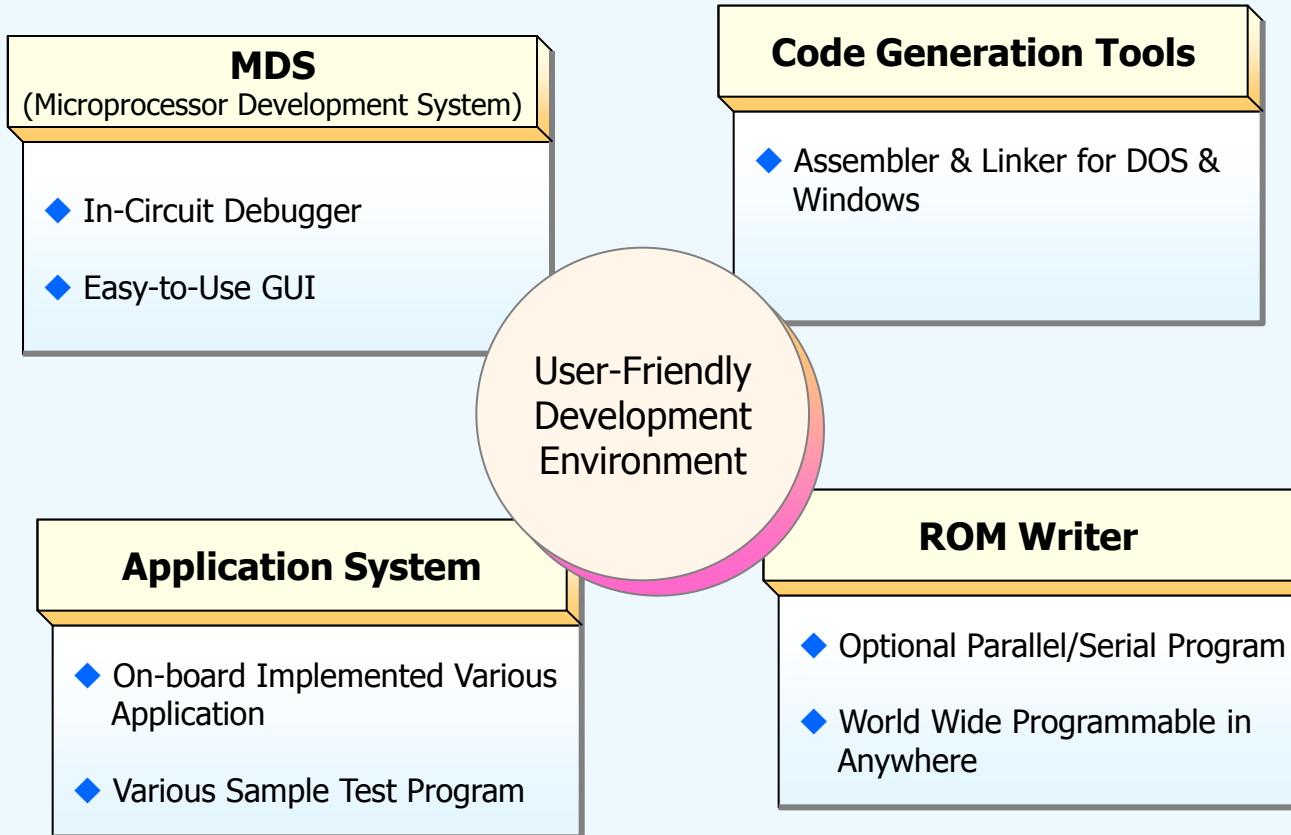
Notes:

1. Dimension D & E include mold mismatch and are determined at the mold parting line.
2. General appearance spec. should be based on final visual inspection spec.

11. 제품 번호 체계

Preliminary





◆ 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		

◆ OP CODE 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	MOV D @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	<table border="1"><tr><td>0001</td><td>dddd</td></tr></table>	0001	dddd
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	<table border="1"><tr><td>0000</td><td>1000</td></tr></table>	0000	1000
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	<table border="1"><tr><td>0000</td><td>1001</td></tr></table>	0000	1001
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	<table border="1"><tr><td>0000</td><td>1100</td></tr></table>	0000	1100
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	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.				
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	<p>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.</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.</p> <p>The carry flag is set if the contents are equal.</p>			
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 ; IF $(A) > M[DP]$ CMP_LE: JC CMP_EQ ; ; IF $(A) < M[DP]$ CMP_EQ: ; 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 ; IF $M[DP] = 8$ CMP_NE: JC CMP_LT ; Branches to CMP_LT ; IF $M[DP] > 8$ CMP_LT: ; IF $M[DP] < 8$			

CJNE A, #data, rel

Binary Code	1100	dddd	rrrr	rrrr
Description	<p>Compares the contents of Accumulator 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 ACC is not affected.</p> <p>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.</p>			
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 ; IF (A) = 8 CMP_NE: JC CMP_LT ; Branch is not taken. ; IF (A) > 8 CMP_LT: ; IF (A) < 8			

CJNE A, @DP, rel

Binary Code	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.</p> <p>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. ; IF (A) = M[DP] CMP_NE: JNC CMP_GT ; IF (A) \neq M[DP] ; IF (A) < M[DP] CMP_GT: ; IF (A) > M[DP]			

CJNE A, dir, rel

Binary Code	1101	dddd	rrrr	rrrr
Description	<p>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.</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.</p> <p>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.</p>			
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	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	<table border="1"><tr><td>1000</td><td>0110</td></tr></table>	1000	0110
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	<pre> ; Assumes P2 contains 0xF. MOV L, #1 ; (L) ← 1 CLR @L ; P2.1 ← 0 MOV A, #0xD ; (A) ← 13 CJNE A, P2, ERROR ; Check if P2.1 is 0. </pre>		

CLR A

Binary Code	<table border="1"><tr><td>0101</td><td>0000</td></tr></table>	0101	0000
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"><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"><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	<pre> ; Assumes M[DP] contains 7. CLR 2 ; M[DP].2 ← 0 CJNE @DP, #3, ERROR ; Check result </pre>		

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	<pre>MOV A, P0 ; (A) ← P0 CPL A ; ACC contains 1's ; complement of P0</pre>	

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	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	0000	0101
Description	Decrements the data pointer.	
Operation	$(DP) \leftarrow (DP) - 1$	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	<pre>; Assumes DPTR contains 0. DEC DPTR ; By underflow, all bits ; of DPH and DPL are set. DEC DP ; This is also valid.</pre>	

DJNZ A, rel

Binary Code	<table border="1"><tr><td>1001</td><td>0001</td><td>rrrr</td><td>rrrr</td></tr></table>	1001	0001	rrrr	rrrr
1001	0001	rrrr	rrrr		
Description	<p>Decrements the contents of ACC, and branches if the result is not zero.</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> <p>Carry is cleared when the borrow occurs; otherwise, carry is set.</p>				
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	<table border="1"><tr><td>0000</td><td>0110</td></tr></table>	0000	0110
0000	0110		
Description	Increases 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	<table border="1"><tr><td>0001</td><td>0001</td></tr></table>	0001	0001
0001	0001		
Description	<p>Increases the contents of ACC.</p> <p>This is the same as "ADD A, #1".</p> <p>Carry is set when the overflow occurs; otherwise, carry is cleared.</p>		
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	<table border="1"><tr><td>0000</td><td>0100</td></tr></table>	0000	0100
0000	0100		
Description	Increments the data pointer.		
Operation	$(DP) \leftarrow (DP) + 1$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	<pre> ; 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. </pre>		

JB bit, rel

Binary Code	<table border="1"><tr><td>1001</td><td>11bb</td><td>rrrr</td><td>rrrr</td></tr></table>	1001	11bb	rrrr	rrrr
1001	11bb	rrrr	rrrr		
Description	<p>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 .</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 memory is not affected.</p>				
Operation	$(PC) \leftarrow (PC) + 2$ IF M[DP].bit = 1 THEN $(PC) \leftarrow (PC) + rel$				
Carry Flag	Not affected.				
Bytes	2				
Cycles	2				
Example	<pre> JB 0, L_BIT_SET ; IF M[DP].0 = 0 L_BIT_SET: ; IF M[DP].0 = 1 </pre>				

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	$(PC) \leftarrow addr$			
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	<p>Branches if the bit in data memory is 0.</p> <p>The address of memory is given by DPTR and bit position is given by two least significant bits of opcode .</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 memory is not affected.</p>			
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	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	JNC L_C_ZERO ; IF (C) = 1 L_C_ZERO: ; IF (C) = 0			

MOV @DP, A

Binary Code	<table border="1"><tr><td>1000</td><td>0011</td></tr></table>	1000	0011
1000	0011		
Description	The contents of ACC is copied to data memory whose address is given by DPTR.		
Operation	$M[DP] \leftarrow (A)$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	<pre>MOV H, #2 ; (H) ← 2 MOV L, #14 ; (L) ← 14 MOV @DP, A</pre>		

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, #data

Binary Code	<table border="1"><tr><td>0101</td><td>dddd</td></tr></table>	0101	dddd
0101	dddd		
Description	Sets ACC with the data given in four low-order bits of opcode.		
Operation	$(A) \leftarrow \#data$		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	<pre>MOV A, #-1 ; (A) ← 15 MOV A, #0xC ; (A) ← 12</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) ← #data	
Carry Flag	Not affected.	
Bytes	1	
Cycles	1	
Example	MOV H, #1 ; (H) ← 1	

MOV L, #data

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

MOV L, @DP

Binary Code	1000	0010
Description	Copies the contents of data memory to DPL. The address of memory is given by DPTR.	
Operation	(L) ← 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	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] ← (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	<table border="1"><tr><td>1000</td><td>0101</td></tr></table>	1000	0101
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] ← (A) (DP) ← (DP) - 1		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	MOVD @DP, A		

MOVI @DP, A

Binary Code	<table border="1"><tr><td>1000</td><td>0100</td></tr></table>	1000	0100
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] ← (A) (DP) ← (DP) + 1		
Carry Flag	Not affected.		
Bytes	1		
Cycles	1		
Example	MOVI @DP, A		

MOVI @DP, #data

Binary Code	<table border="1"><tr><td>0100</td><td>dddd</td></tr></table>	0100	dddd
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] ← #data (DP) ← (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	0000	1101
Description	ORL performs the bitwise logical-OR operation between the indirect data memory and ACC. The result is stored in Accumulator.	
Operation	$(A) \leftarrow (A) M[DP]$	
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	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	<table border="1"><tr><td>1001</td><td>0000</td></tr></table>	1001	0000
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	<table border="1"><tr><td>0000</td><td>0001</td></tr></table>	0000	0001
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"><tr><td>0000</td><td>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"><tr><td>1000</td><td>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) ← 1 SETB @L ; P2.1 ← 1 MOV A, #2 ; (A) ← 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	

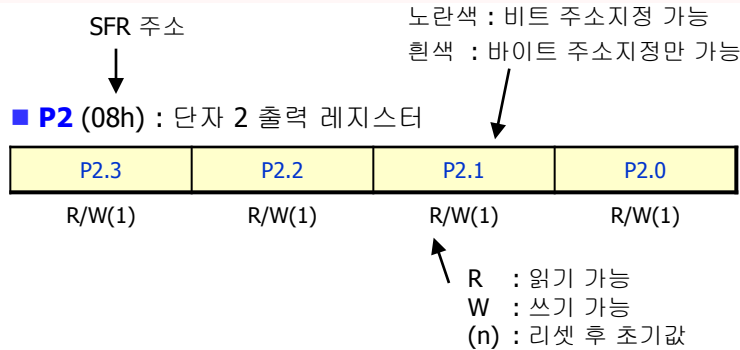
XCH A, @DP

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.	

[특수기능 레지스터 설명 이해하기]



■ **P0 (00h)** : 단자 0 출력 레지스터

P0.3	P0.2	P0.1	P0.0
R/W(1)	R/W(1)	R/W(1)	R/W(1)

■ **P4 (01h)** : 단자 4 출력 레지스터

P4.3	P4.2	P4.1	P4.0
R/W(1)	R/W(1)	R/W(1)	R/W(1)

■ **DPL (02h)** : 데이터 포인터(DPTR)의 하위 Nibble

DPL.3	DPL.2	DPL.1	DPL.0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

■ **DPH (03h)** : 데이터 포인터(DPTR)의 상위 Nibble

-	-	DPH.1	DPH.0
		R/W(0)	R/W(0)

■ **P1 (04h)** : 단자 1 출력 레지스터

P1.3	P1.2	P1.1	P1.0
R/W(1)	R/W(1)	R/W(1)	R/W(1)

■ **REMC (05h)** : REM 출력 제어 레지스터

REME	PG2	PG1	PG0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

- ◆ PG[2:0] : Carrier 주파수 선택.
- ◆ REME : REM 출력 인에이블.

■ **SPL (06h)** : 스택 포인터(SP)의 하위 Nibble

SP.3	SP.2	SP.1	SP.0
R/W(1)	R/W(1)	R/W(1)	R/W(1)

- ◆ 스택이 시작할 주소를 지시함.
- ◆ PUSH에 의하여 증가하고 POP에 의하여 감소.

■ **SPH (07h)** : 스택 포인터(SP)의 상위 Nibble

-	-	SPh.1	SPh.0
		R/W(0)	R/W(1)

■ P2 (08h) : 단자 2 출력 레지스터

P2.3	P2.2	P2.1	P2.0
R/W(1)	R/W(1)	R/W(1)	R/W(1)

■ IAPCON (09h) : IAP 제어 레지스터

RGS1	RGS0	OPS1	OPS0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

◆ RGS[1:0] : IAP 영역 선택.

[0,0] : EEP0 (0x1C0 ~ 0x1FF)

[0,1] : EEP1 (0x3C0 ~ 0x3FF)

[1,0] : INFO (0x0 ~ 0x7)

[1,1] : Reserved

◆ OPS[1:0] : IAP 기능 선택.

[0,0] : NO operation

[0,1] : 바이트 읽기

[1,0] : 바이트 소거

[1,1] : 바이트 쓰기

■ GDL (0Ah) : 범용 데이터 레지스터의 하위 Nibble

GDL.3	GDL.2	GDL.1	GDL.0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

■ GDH (0Bh) : 범용 데이터 레지스터의 상위 Nibble

GDH.3	GDH.2	GDH.1	GDH.0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

■ P3 (0Ch) : 단자 3 출력 레지스터

P3.3	P3.2	P3.1	P3.0
R/W(1)	R/W(1)	R/W(1)	R/W(1)

■ CKCFG (0Dh) : 클럭 설정 레지스터

XT/RG	DIV2	DIV1	DIV0
R/W(0)	R/W(0)	R/W(0)	R/W(0)

◆ XT/RG : 클럭 발생원 선택.

0 : 시스템 클럭으로 내부 링 발진기 선택.

외부 클럭 발진기는 디세이블됨.

1 : 시스템 클럭으로 외부 클럭 발진기를 선택.

내부 링 발진기는 디세이블됨.

8 핀 패키지에서 이 비트를 사용하지 말라.

◆ DIV[2:0] : 시스템 클럭 분주비 선택.

[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] : -

■ IOCFG (0Eh) : 입출력 단자 구조설정 레지스터

IOMAP1	IOMAP0	P2OEN	IOXEN
R/W(0)	R/W(0)	R/W(0)	R/W(0)

- ◆ IOXEN : XI와 XO를 입출력 단자로 인에이블.
0 : XI와 XO는 클럭 입력으로 사용됨 (기본지정).
1 : XI와 XO는 PORT4[1:0]로 사용됨
- ◆ P2OEN : P2를 push-pull 출력 단자로 설정.
- ◆ IOMAP [1:0] : 입출력 단자 대응 설정.
[0,0] : 기본지정.
[0,1] : 20핀 입출력 단자 대응 옵션
[1,0] : 24핀 입출력 단자 대응 옵션
[1,1] : Reserved

■ LVCFG (0Fh) : LVD 설정 레지스터

POR	Reserved	Reserved	Reserved
R/W(1)	R(X)	R/W(0)	R/W(0)

- ◆ Reserved : 향후 호환성을 위하여 1로 설정하지 말라.
- ◆ POR : Power-on-reset flag.
소프트웨어는 전원이 켜질 때 리셋과 켜진 상태에서 리셋이 발생하는 것을 이 flag를 사용하여 구분한다.

- ◆ V1.0
 - ✓ First Official Release
- ◆ V1.1
 - ✓ Modify Internal Ring Spec.
 - ✓ Add Internal Ring OSC. Slide
- ◆ V1.2
 - ✓ Modify Operating frequency
- ◆ V1.3
 - ✓ Added GC49C501RX devices.
 - ✓ Description for POR condition.
 - ✓ LVOFF flag is not supported any more.
- ◆ V1.4
 - ✓ Modify Internal Ring Spec
 - ✓ Add E.S.D. Spec.
- ◆ V1.5
 - ✓ Enhanced description for 8-pin devices.
 - ✓ Optional power-fail reset is not supported any more.
- ◆ V1.6
 - ✓ Add 20-SOIC (JEDEC) Package
 - ✓ Modify Package Dimensions
 - ✓ Now POR block has no limitation for the power rising slope.
- ◆ V1.7
 - ✓ Delete 20-SOIC (Narrow) Package