ATmega128의 내장 병렬 I/O 포트

Jee-Hwan Ryu

School of Mechanical Engineering
Korea University of Technology and Education

기본 구조

- 6개의 8비트 양방향 병렬 I/O 포트
  - Port A ~ Port F
- 1개의 5비트 양방향 병렬 I/O 포트
  - Port G
- Port A ~ Port E
  - 범용 I/O 포트로 사용될 경우 read-modify-write동작 가능
  - 즉, 입출력 방향 변경 없이 SBI 및 CBI명령에서 포트의 동작방향 달라질 수 있다.

병렬 I/O포트 레지스터 영역

- DDRxn
  - x: Port A ~ Port G
  - n: 각 포트의 비트번호
  - 입출력의 방향을 설정, 1(출력), 0(입력)
  - Read-write 가능
- PORTxn
  - 데이터 출력
  - Read-write 가능
- PINxn
  - 포트 입력
  - Only read
- SFIOR (Special Function I/O Register)의 PUD
  - PUD(Pull-up Disable) 비트를 1로 셋트하면 그 기능이 금지

각 병렬 I/O 포트에 관련된 레지스터

Bit	7	6	5	4	3	2	1	0	
	PORTA7	PORTA6	PORTA5	PORTA4	PORTA3	PORTA2	PORTA1	PORTA0	PORTA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	DDA7	DDA6	DDA5	DDA4	DDA3	DDA2	DDA1	DDA0	DDRA
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	PINA7	PINA6	PINA5	PINA4	PINA3	PINA2	PINA1	PINA0	PINA
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

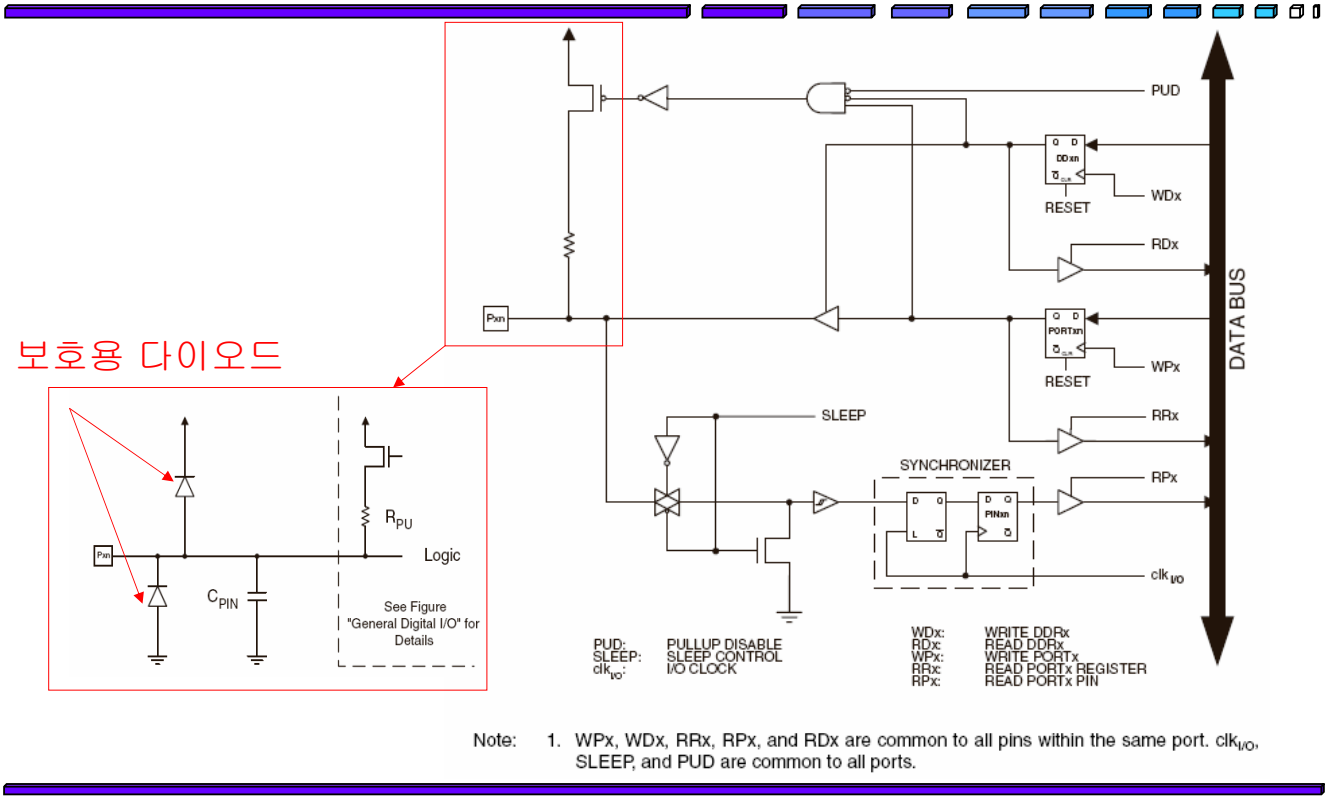
Bit	7	6	5	4	3	2	1	0	
	–	–	–	PORTG4	PORTG3	PORTG2	PORTG1	PORTG0	PORTG
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	TSM	–	–	–	ACME	PUD	PSR0	PSR321	SFIOR
Read/Write	R/W	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	–	–	–	DDG4	DDG3	DDG2	DDG1	DDG0	DDRG
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
	–	–	–	PING4	PING3	PING2	PING1	PING0	PING
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	N/A	N/A	N/A	N/A	N/A	

병렬 I/O 포트의 기본구조



Note: 1. WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

병렬 I/O 포트 핀의 동작

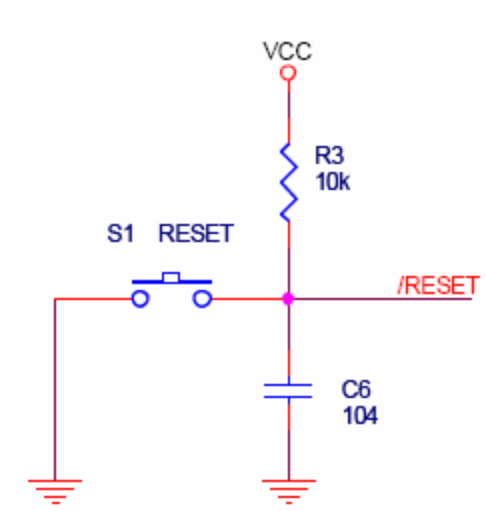
- 사용하지 않는 입력핀은 플로팅 시켜두지 말고 외부에서 풀업 또는 풀다운으로 확실한 논리상태를 입력하는 것이 전력소비를 감소시킨다.
- 핀을 직접 Vcc나 GND에 연결하는 것은 이 핀들이 출력 방향으로 잘못 설정되는 경우에 큰 단락전류가 흐를 수 있다.

DDxn	PORTxn	PUD (In SFIOR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if ext. pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

사용하지 않는 경우

Pull Up 저항

- 스위치를 이용하여 0/1 논리 레벨을 확실히 정해주기 위함
- 저항기 GND 측에 연결되어 있는 경우: Pull Down 저항



스위치 입력과 풀업/풀다운 저항

Open 시
0 도 1도 아님

(a) 스위치 입력 (b) 풀업저항 사용

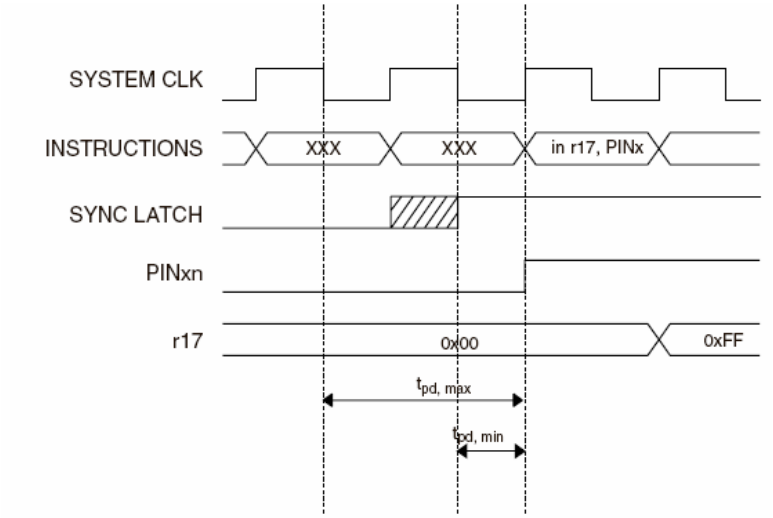
<그림 1> L 스위치 입력과 풀업 저항

(a) 스위치 입력 (b) 풀다운저항 사용

<그림 2> H 스위치 입력과 풀다운 저항

외부 핀을 읽어 들이는 동작

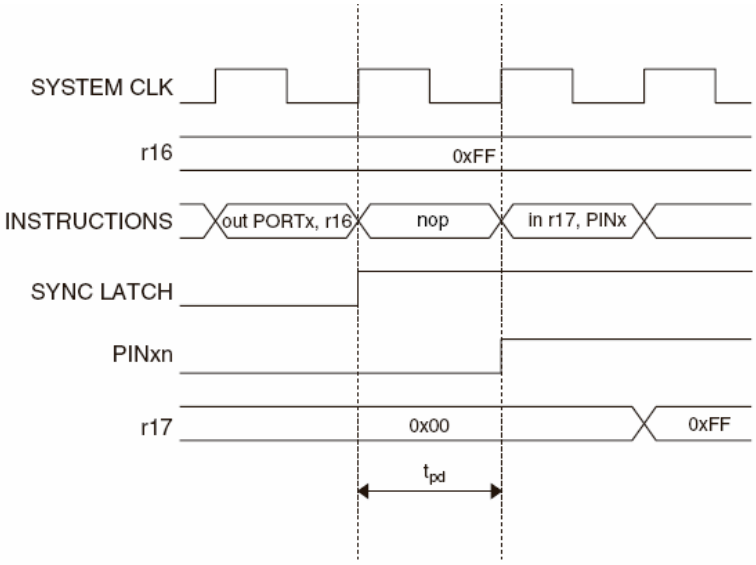
- PINxn 레지스터는 metastability 현상 피하기 위해 클럭 동기회로 사용
- 0.5 ~ 1.5클럭 시간지연 발생



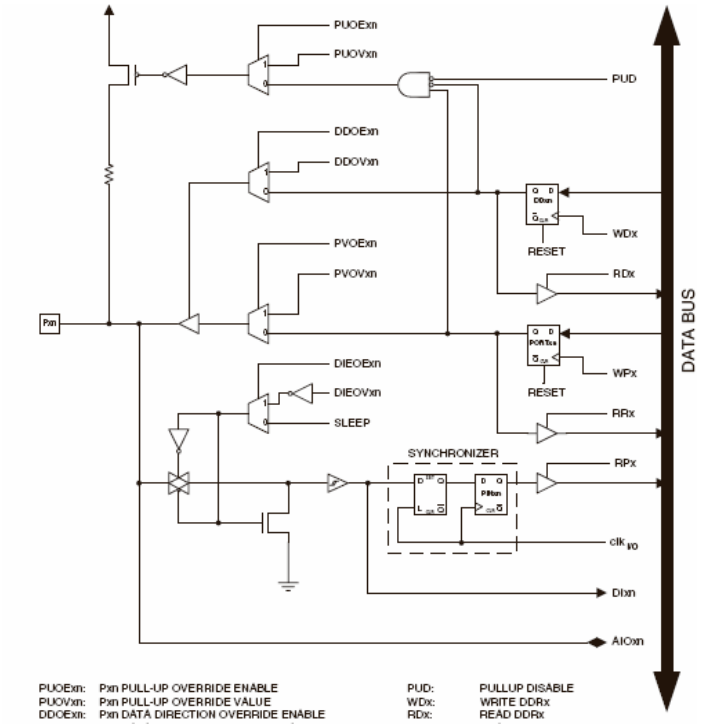
CLK이 Low level이 되어야 first latch 통과

출력 후 다시 읽어 들이는 동작

- 출력 후 다시 읽기 위해서는 곧바로 읽지 말고 중간에 NOP명령 삽입



병렬 I/O 부가기능



Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD Register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn Register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn Register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn Register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn Register bit.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU-state (Normal mode, Sleep modes).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, Sleep modes).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the schmitt trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/output	This is the Analog Input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bi-directionally.

PORT A의 부가기능

- 외부 메모리를 인터페이스 하기 위한 시분할 다중화된 데이터버스 및 어드레스 버스의 하위 바이트
- 시분할 다중화된 버스로부터 어드레스버스 분리하는데 ALE (Address Latch Enable) 신호 사용

Port Pin	Alternate Function
PA7	AD7 (External memory interface address and data bit 7)
PA6	AD6 (External memory interface address and data bit 6)
PA5	AD5 (External memory interface address and data bit 5)
PA4	AD4 (External memory interface address and data bit 4)
PA3	AD3 (External memory interface address and data bit 3)
PA2	AD2 (External memory interface address and data bit 2)
PA1	AD1 (External memory interface address and data bit 1)
PA0	AD0 (External memory interface address and data bit 0)

PORT B의 부가기능

- 타이머/카운터
- SPI기능을 위한 신호

Port Pin	Alternate Functions
PB7	OC2/OC1C(1) (Output Compare and PWM Output for Timer/Counter2 or Output Compare and PWM Output C for Timer/Counter1)
PB6	OC1B (Output Compare and PWM Output B for Timer/Counter1)
PB5	OC1A (Output Compare and PWM Output A for Timer/Counter1)
PB4	OC0 (Output Compare and PWM Output for Timer/Counter0)
PB3	MISO (SPI Bus Master Input/Slave Output)
PB2	MOSI (SPI Bus Master Output/Slave Input)
PB1	SCK (SPI Bus Serial Clock)
PB0	$\overline{SS}$ (SPI Slave Select input)

PORT C의 부가기능

- 외부 메모리 인터페이스 위한 어드레스버스의 상위 바이트

Port Pin	Alternate Function
PC7	A15
PC6	A14
PC5	A13
PC4	A12
PC3	A11
PC2	A10
PC1	A9
PC0	A8

PORT D의 부가기능

- 타이머/카운터
- 외부 인터럽트
- USART1, TWI 직렬통신 포트 기능을 위한 신호들

Port Pin	Alternate Function
PD7	T2 (Timer/Counter2 Clock Input)
PD6	T1 (Timer/Counter1 Clock Input)
PD5	XCK1(1) (USART1 External Clock Input/Output)
PD4	ICP1 (Timer/Counter1 Input Capture Pin)
PD3	INT3/TXD1(1) (External Interrupt3 Input or UART1 Transmit Pin)
PD2	INT2/RXD1(1) (External Interrupt2 Input or UART1 Receive Pin)
PD1	INT1/SDA(1) (External Interrupt1 Input or TWI Serial DAta)
PD0	INT0/SCL(1) (External Interrupt0 Input or TWI Serial CLock)

PORT E의 부가기능

- 타이머/카운터
- 외부 인터럽트
- USART0, 직렬통신 포트, 아날로그비교기, ISP 기능을 위한 신호들

Port Pin	Alternate Function
PE7	INT7/ICP3(1) (External Interrupt 7 Input or Timer/Counter3 Input Capture Pin)
PE6	INT6/ T3(1) (External Interrupt 6 Input or Timer/Counter3 Clock Input)
PE5	INT5/OC3C(1) (External Interrupt 5 Input or Output Compare and PWM Output C for Timer/Counter3)
PE4	INT4/OC3B(1) (External Interrupt4 Input or Output Compare and PWM Output B for Timer/Counter3)
PE3	AIN1/OC3A (1) (Analog Comparator Negative Input or Output Compare and PWM Output A for Timer/Counter3)
PE2	AIN0/XCK0(1) (Analog Comparator Positive Input or USART0 external clock input/output)
PE1	PDO/TXD0 (Programming Data Output or UART0 Transmit Pin)
PE0	PDI/RXD0 (Programming Data Input or UART0 Receive Pin)

PORT F의 부가기능

- A/D 컨버터
- JTAG 인터페이스 기능을 위한 신호

Port Pin	Alternate Function
PF7	ADC7/TDI (ADC input channel 7 or JTAG Test Data Input)
PF6	ADC6/TDO (ADC input channel 6 or JTAG Test Data Output)
PF5	ADC5/TMS (ADC input channel 5 or JTAG Test Mode Select)
PF4	ADC4/TCK (ADC input channel 4 or JTAG Test ClocK)
PF3	ADC3 (ADC input channel 3)
PF2	ADC2 (ADC input channel 2)
PF1	ADC1 (ADC input channel 1)
PF0	ADC0 (ADC input channel 0)

PORT G의 부가기능

- PG4~PG0 5비트만 유효
- 외부메모리 인터페이스나 타이머/카운터

Port Pin	Alternate Function
PG4	TOSC1 (RTC Oscillator Timer/Counter0)
PG3	TOSC2 (RTC Oscillator Timer/Counter0)
PG2	ALE (Address Latch Enable to external memory)
PG1	$\overline{RD}$ (Read strobe to external memory)
PG0	$\overline{WR}$ (Write strobe to external memory)

Delay Functions

```
void Delay_us(unsigned char time_us)            /* time delay for us */
{ register unsigned char i;

  for(i = 0; i < time_us; i++)                  // 4 cycle +
    { asm volatile(" PUSH  R0 ");                 // 2 cycle +  R0: temporary register
      asm volatile(" POP   R0 ");             // 2 cycle +
      asm volatile(" PUSH  R0 ");             // 2 cycle +
      asm volatile(" POP   R0 ");             // 2 cycle +
      asm volatile(" PUSH  R0 ");             // 2 cycle +
      asm volatile(" POP   R0 ");             // 2 cycle = 16 cycle = 1 us for 16MHz
    }
}
```

```
void Delay_ms(unsigned int time_ms)           /* time delay for ms */
{ register unsigned int i;

  for(i = 0; i < time_ms; i++)
    { Delay_us(250);
      Delay_us(250);
      Delay_us(250);
      Delay_us(250);
    }
}
```