ECE3411 – Fall 2015

Lecture 4b.

Review Session

Syed Kamran Haider, Marten van Dijk
Department of Electrical & Computer Engineering
University of Connecticut
Email: {syed.haider, vandijk}@engr.uconn.edu

UCONN

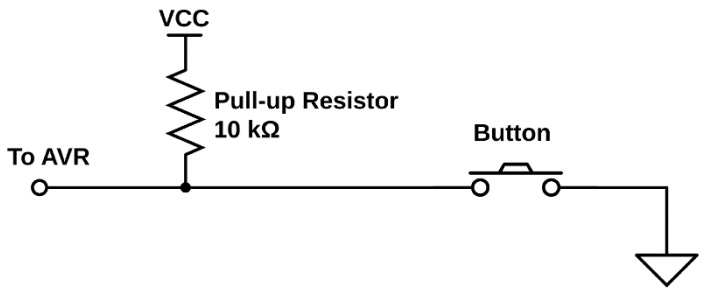
Hardware Registers of a Port

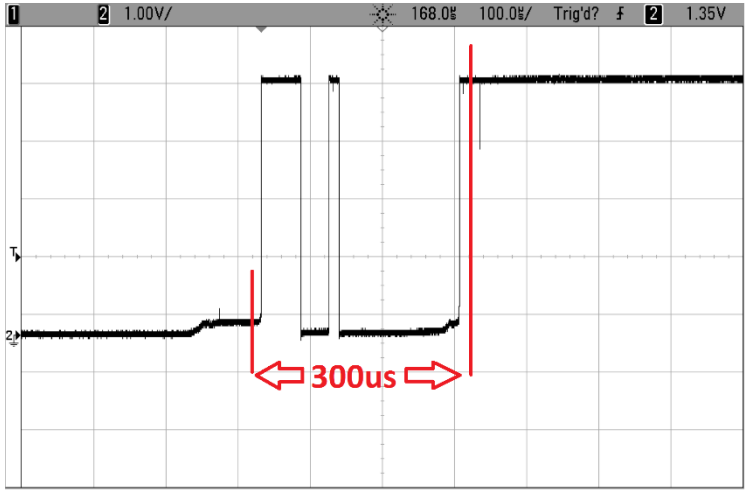
Each Port on the Mega AVRs has three hardware registers associated to it:

- **DDRx** : *Data-Direction Register for Port x*
  - Controls whether each pin is configured for input (0) or output (1).
  - To enable a pin as output, a '1' is written to that bit in DDRx.
  - By default, all pins are initialized as inputs (DDRx = 0x00).
- **PORTx** : *Port x Data Register*
  - Sets an output pin to logic HIGH (1) or LOW (0).
  - E.g. writing a '1' to a bit position in PORT register will produce logic HIGH at that pin & vice versa.
- **PINx** : *Port x Input Pins Address*
  - Used to read the logic values of each pin that's configured as input.
  - E.g. a value '0' of a bit of PIN register indicates a low voltage at that pin & vice versa.

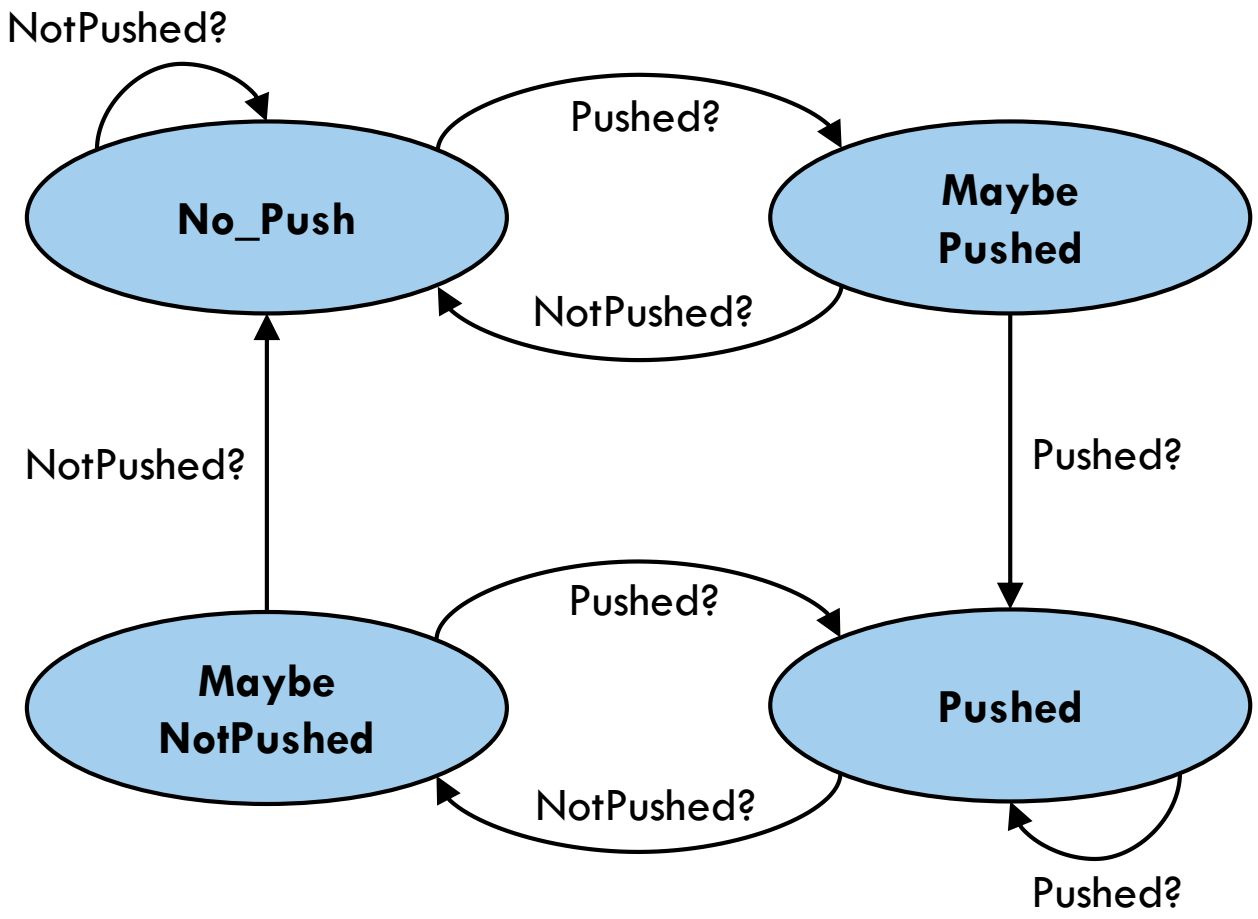
Debouncing of Bouncing Signals

- A button push results in a bouncy transition
  - Due to physical limitations of the contact surfaces
  - Bouncing is often very fast → orders of few us to ms
- Debouncing in software
  - Key idea: Read → Wait → Verify
  - Wait time needs to be carefully controlled
  - E.g. wait time should be at least 300us for this example.





Software Debouncing State Machine

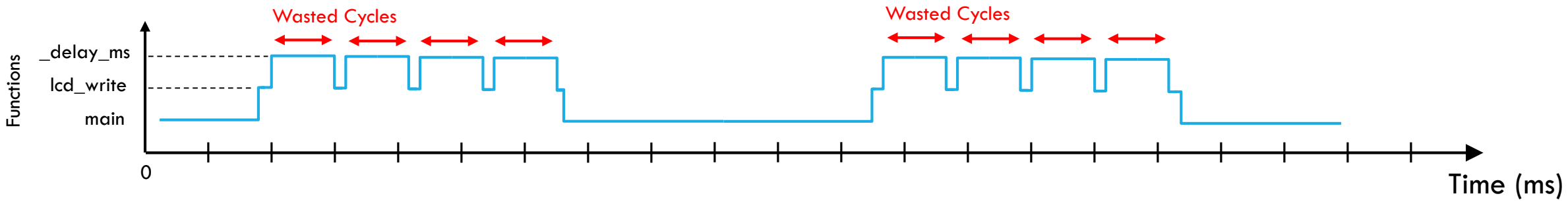


LCD Data Write (4-bit Mode)

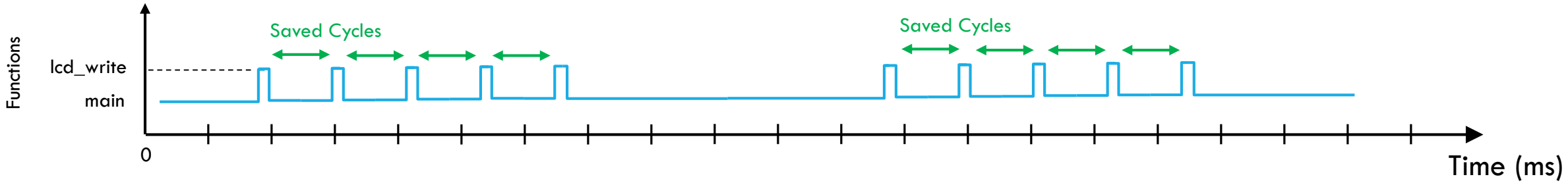
```
void LcdDataWrite(uint8_t da)
{
        // First send higher 4-bits
①       DATA_PORT = (DATA_PORT & 0xf0) | (da >> 4);      //give the higher half of cm to DATA_PORT
        CTRL_PORT |= (1<<RS);                           //setting RS=1 to choose the data register
        CTRL_PORT |= (1<<ENABLE);                       //setting ENABLE=1

        _delay_ms(1);                                   // allow the LCD controller to successfully read command in

②       CTRL_PORT &= ~(1<<ENABLE);                      // Setting ENABLE=0

        _delay_ms(1);                                   // allow long enough delay

        // Send lower 4-bits
③       DATA_PORT = (DATA_PORT & 0xf0) | (da & 0x0f);   //give the lower half of cm to DATA_PORT
        CTRL_PORT |= (1<<RS);                           //setting RS=1 to choose the data register
        CTRL_PORT |= (1<<ENABLE);                       //setting ENABLE=1

        _delay_ms(1);                                   // allow the LCD controller to successfully read command in

④       CTRL_PORT &= ~(1<<ENABLE);                      // Setting ENABLE=0

        _delay_ms(1);                                   // allow long enough delay
}
```

Blocking vs. Non Blocking LCD Write Timing

Blocking Writes:



Non-Blocking Writes:



Interrupts & ISRs

A few questions:

- Who calls the ISR?
- Can you “pass” a variable to an ISR?
- What is the return value of an ISR?
- How does the AVR know where to find the code for the corresponding ISR?

Interrupts & ISRs

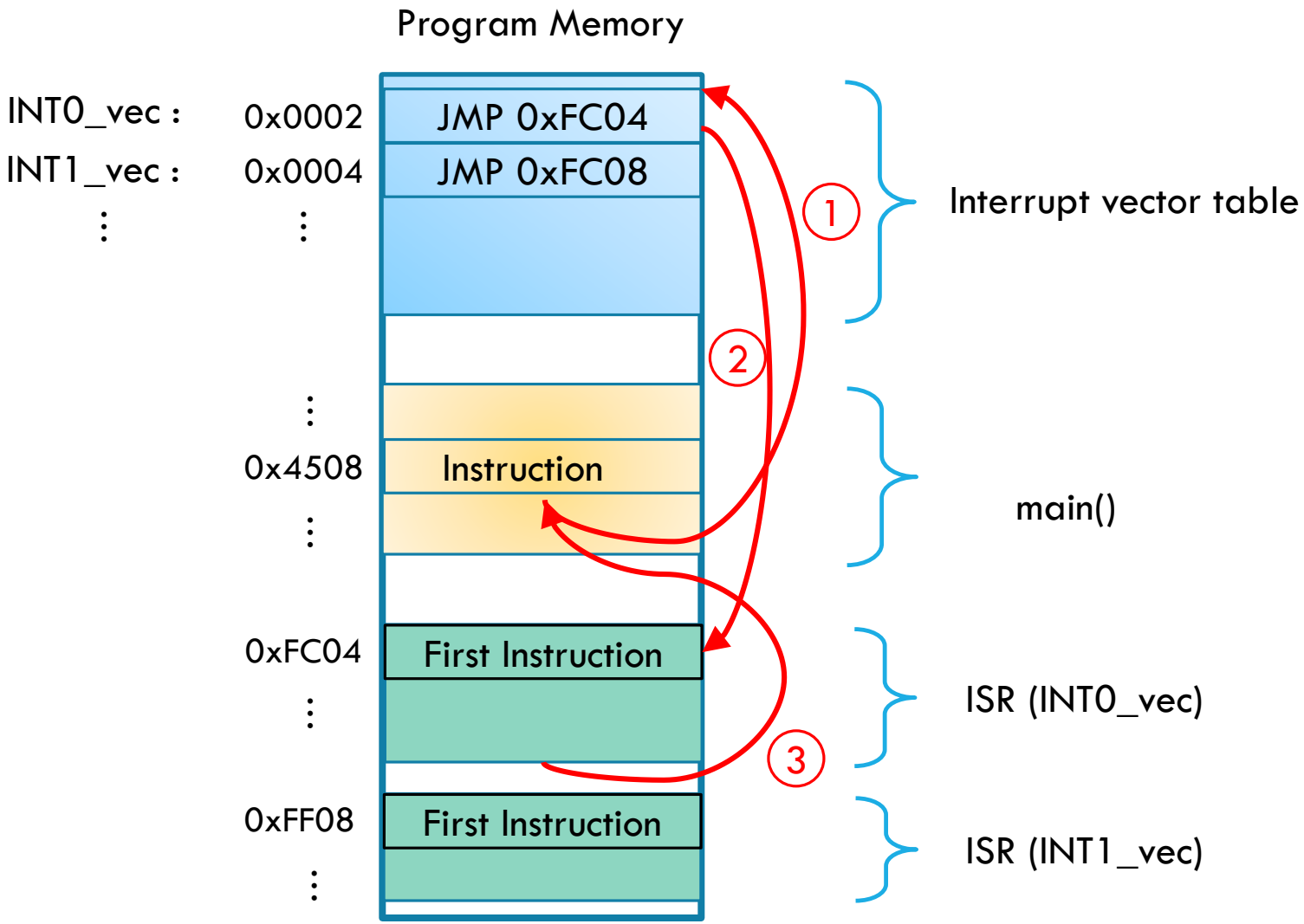
- Who calls the ISR?
  - The hardware!
- Can you “pass” a variable to an ISR?
  - No! The variable must be globally defined.
- What is the return value of an ISR?
  - Nothing! However, it can store some value in a global variable.
- How does the AVR know where to find the code for the corresponding ISR?
  - Through the Interrupt Vector Table.

ATmega328P Interrupt Vector Table

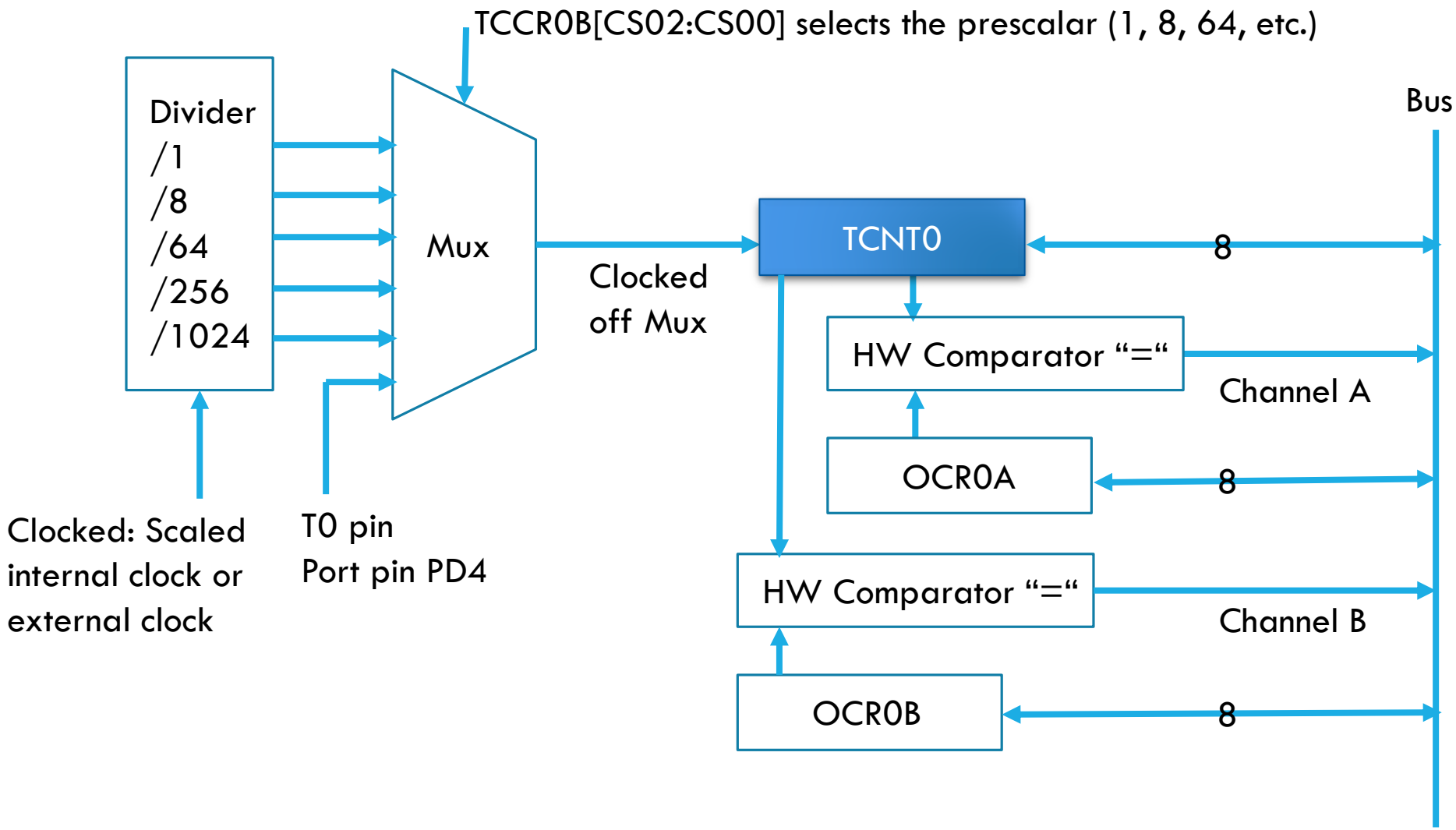
- The AVR knows what type of interrupt has occurred.
- It jumps to the program address specified in Interrupt Vector Table.
  - E.g. Address 0x0002 for INT0
- There it sees another Jump instruction which takes it to the ISR code.

VectorNo.	Program Address(2)	Source	Interrupt Definition
1	0x0000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2
7	0x000C	WDT	Watchdog Time-out Interrupt
8	0x000E	TIMER2 COMPA	Timer/Counter2 Compare Match A
9	0x0010	TIMER2 COMPB	Timer/Counter2 Compare Match B
10	0x0012	TIMER2 OVF	Timer/Counter2 Overflow
11	0x0014	TIMER1 CAPT	Timer/Counter1 Capture Event
12	0x0016	TIMER1 COMPA	Timer/Counter1 Compare Match A
13	0x0018	TIMER1 COMPB	Timer/Coutner1 Compare Match B
14	0x001A	TIMER1 OVF	Timer/Counter1 Overflow
15	0x001C	TIMER0 COMPA	Timer/Counter0 Compare Match A
16	0x001E	TIMER0 COMPB	Timer/Counter0 Compare Match B
17	0x0020	TIMER0 OVF	Timer/Counter0 Overflow
18	0x0022	SPI, STC	SPI Serial Transfer Complete
19	0x0024	USART, RX	USART Rx Complete
20	0x0026	USART, UDRE	USART, Data Register Empty
21	0x0028	USART, TX	USART, Tx Complete
22	0x002A	ADC	ADC Conversion Complete

Execution of an ISR



Timer 0


TCCR0B[CS02:CS00] selects the prescalar (1, 8, 64, etc.)
Divider
/1
/8
/64
/256
/1024
Mux
Clocked
off Mux
TCNT0
8
Bus
HW Comparator "="
Channel A
OCR0A
8
HW Comparator "="
Channel B
OCR0B
8
Clocked: Scaled
internal clock or
external clock
T0 pin
Port pin PD4

Timer 0 Modes of Operation

- Normal Mode
  - Timer counts up from 0
  - Timer overflows at 0xFF (i.e. 255)
  - Interrupt can be generated upon Overflow
- CTC Mode
  - OCR0A is loaded with some value between 0 to 255
  - Timer counts up from 0
  - A compare match (kind of an overflow) occurs when TCNT0 = OCR0A
  - Interrupt can be generated upon Compare Match

Timer 0 Mode Selection

Table 14-8. Waveform Generation Mode Bit Description

Mode	WGM02	WGM01	WGM00	Timer/Counter Mode of Operation	TOP	Update of OCRx at	TOV Flag Set on[(1)(2)]
0	0	0	0	Normal	0xFF	Immediate	MAX
1	0	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	0	1	0	CTC	OCRA	Immediate	MAX
3	0	1	1	Fast PWM	0xFF	BOTTOM	MAX
4	1	0	0	Reserved	–	–	–
5	1	0	1	PWM, Phase Correct	OCRA	TOP	BOTTOM
6	1	1	0	Reserved	–	–	–
7	1	1	1	Fast PWM	OCRA	BOTTOM	TOP

Bit	7	6	5	4	3	2	1	0	
0x24 (0x44)	COM0A1	COM0A0	COM0B1	COM0B0	–	–	WGM01	WGM00	TCCR0A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

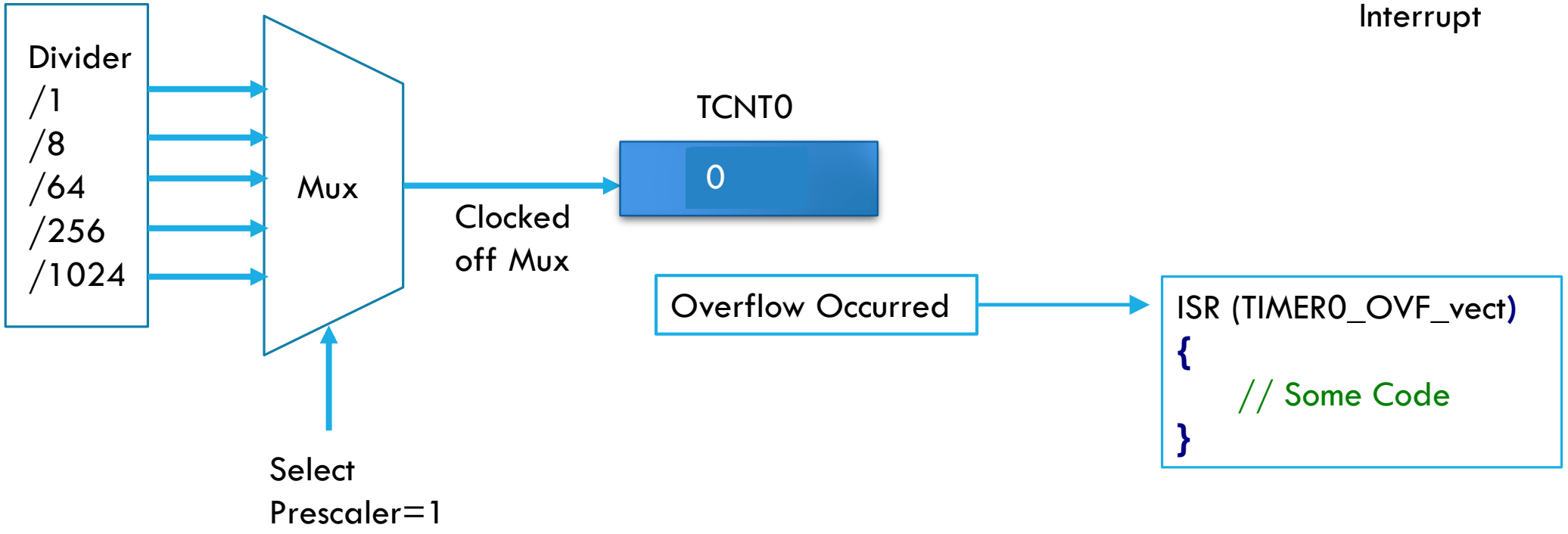
Bit	7	6	5	4	3	2	1	0	
0x25 (0x45)	FOC0A	FOC0B	–	–	WGM02	CS02	CS01	CS00	TCCR0B
Read/Write	W	W	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Timer 0 Overflow Interrupt

TIMSK0 – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Enables Overflow Interrupt

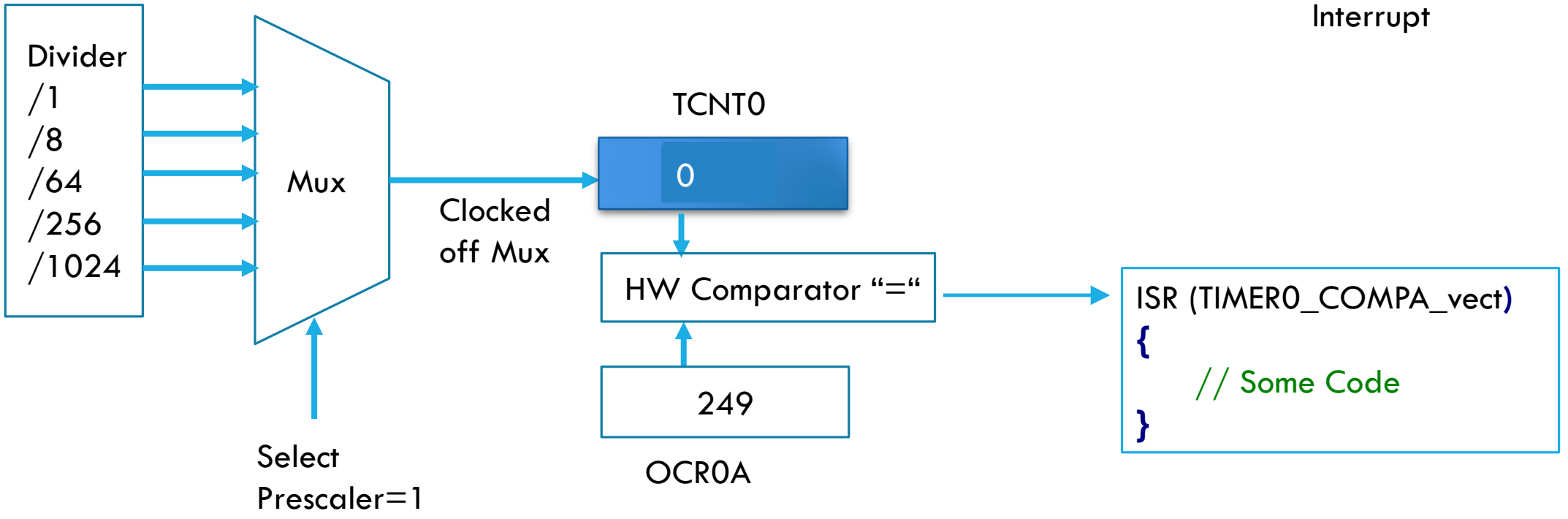


Timer 0 Compare Match Interrupt

TIMSK0 – Timer/Counter Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
(0x6E)	–	–	–	–	–	OCIE0B	OCIE0A	TOIE0	TIMSK0
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Enables Compare_Match_A Interrupt



Timer 1 Modes of Operation

- Normal Mode
  - Timer counts up from 0
  - Timer overflows at 0xFFFF (i.e. 65535)
  - Interrupt can be generated upon Overflow
- CTC Mode
  - OCR1A is loaded with some value between 0 to 65535
  - Timer counts up from 0
  - A compare match (kind of an overflow) occurs when TCNT1 = OCR1A
  - Interrupt can be generated upon Compare Match

Timer 1 Mode Selection

Table 15-4. Waveform Generation Mode Bit Description[1]

Mode	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation	TOP	Update of OCR1x at	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	BOTTOM	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	BOTTOM	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	BOTTOM	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	BOTTOM	TOP
15	1	1	1	1	Fast PWM	OCR1A	BOTTOM	TOP

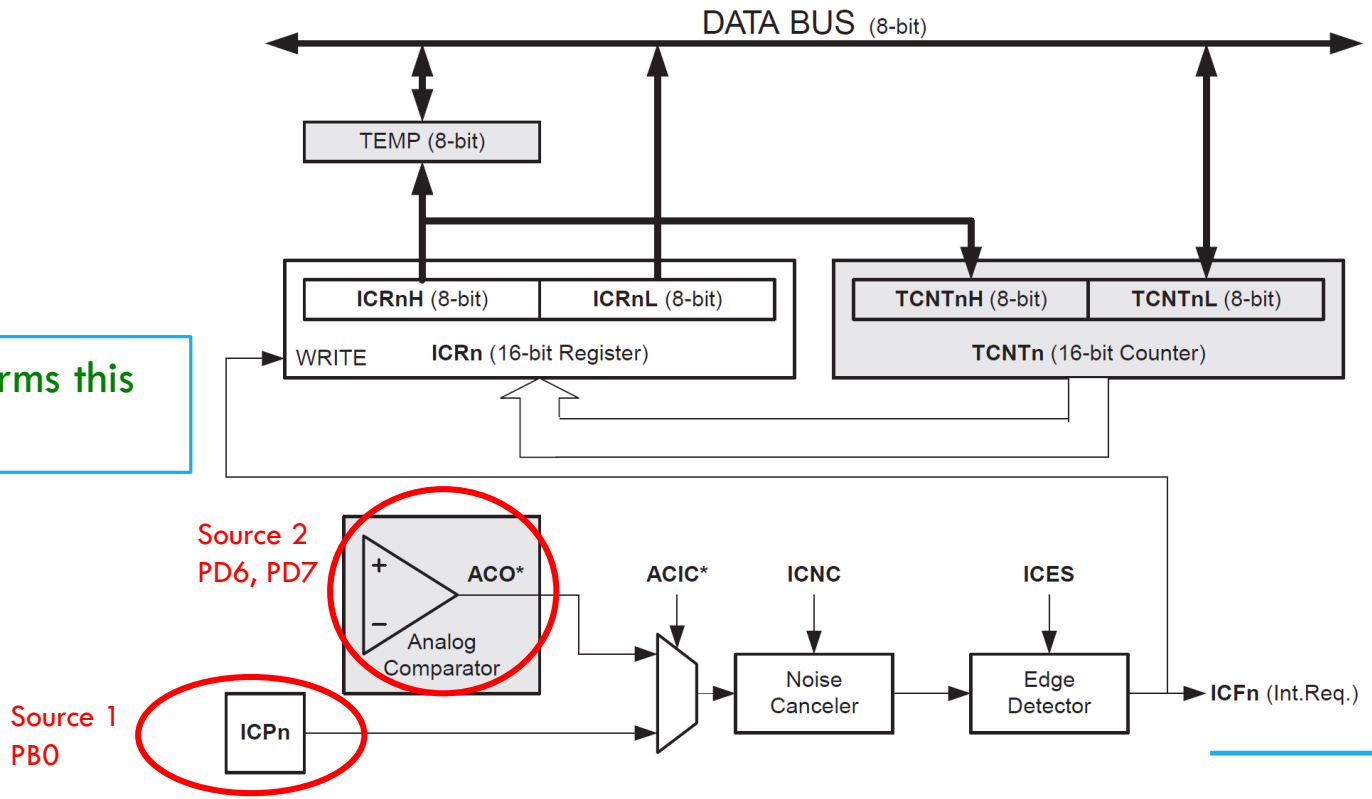
Note: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

Bit	7	6	5	4	3	2	1	0	
(0x80)	COM1A1	COM1A0	COM1B1	COM1B0	–	–	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Bit	7	6	5	4	3	2	1	0	
(0x81)	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Timer 1 Input Capture Interrupt

```
// Hardware performs this
ICR1 = TCNT1;
```




```
ISR (TIMER1_CAPT_vect)
{
    // Some Code
}
```

ACSR – Analog Comparator Control and Status Register

Bit	7	6	5	4	3	2	1	0	
0x30 (0x50)	ACD	ACBG	ACO	ACI	ACIE	ACIC	ACIS1	ACIS0	ACSR
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	N/A	0	0	0	0	0	

External Interrupts

- External Interrupts INT0 & INT1
  - Can detect any logic change in input pins PD2 and PD3 respectively
  - Can also be configured to trigger by a falling or rising edge
  - INT0 has the highest priority among all interrupts, then INT1and so on…
- Pin Change Interrupts PCINT23..0
  - The pin change interrupt PCI0 will trigger if any enabled PCINT7..0 pin toggles
  - The pin change interrupt PCI1 will trigger if any enabled PCINT14..8 pin toggles
  - The pin change interrupt PCI2 will trigger if any enabled PCINT23..16 pin toggles

VectorNo.	Program Address(2)	Source	Interrupt Definition
1	0x0000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog System Reset
2	0x0002	INT0	External Interrupt Request 0
3	0x0004	INT1	External Interrupt Request 1
4	0x0006	PCINT0	Pin Change Interrupt Request 0
5	0x0008	PCINT1	Pin Change Interrupt Request 1
6	0x000A	PCINT2	Pin Change Interrupt Request 2

Configuring INT1

EICRA – External Interrupt Control Register A

The External Interrupt Control Register A contains control bits for interrupt sense control.

Bit	7	6	5	4	3	2	1	0	
(0x69)	–	–	–	–	ISC11	ISC10	ISC01	ISC00	EICRA
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Table 12-1. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

EIMSK – External Interrupt Mask Register

Bit	7	6	5	4	3	2	1	0	
0x1D (0x3D)	–	–	–	–	–	–	INT1	INT0	EIMSK
Read/Write	R	R	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Configuring Pin Change Interrupts

PCICR – Pin Change Interrupt Control Register

Bit	7	6	5	4	3	2	1	0	
(0x68)	–	–	–	–	–	PCIE2	PCIE1	PCIE0	PCICR
Read/Write	R	R	R	R	R	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

PCMSK0 – Pin Change Mask Register 0

Bit	7	6	5	4	3	2	1	0	
(0x6B)	PCINT7	PCINT6	PCINT5	PCINT4	PCINT3	PCINT2	PCINT1	PCINT0	PCMSK0
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	

Corresponding Pins:
PB0, PB1, PB2

PCMSK1 – Pin Change Mask Register 1

Bit	7	6	5	4	3	2	1	0	
(0x6C)	–	PCINT14	PCINT13	PCINT12	PCINT11	PCINT10	PCINT9	PCINT8	PCMSK1
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

PCMSK2 – Pin Change Mask Register 2

Bit	7	6	5	4	3	2	1	0	
(0x6D)	PCINT23	PCINT22	PCINT21	PCINT20	PCINT19	PCINT18	PCINT17	PCINT16	PCMSK2
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	