

Given the code in Listing 01

- Which pin is used as INT0 on the ATmega328P?
 - PD3
 - PD2
 - PB0
 - PC0

Correct Answer: B

- What causes the external interrupt INT0 to trigger this code?
 - Falling edge on PD2
 - Pressing a normally-high push button connected to PD2
 - Level change on PD2
 - Timer overflow

Correct Answers: A, B

- What is the purpose of `sei();` in the code?
 - Enable external interrupt INT0
 - Enable global interrupts
 - Initialize the interrupt vector table
 - Allow all enabled interrupts to be recognized by the CPU

Correct Answers: B, D

- What is the function of `PORTB ^= (1 << PORTB0);` in the ISR?
 - It turns the LED off
 - It toggles the LED
 - It keeps the LED always on
 - It changes the current state of PB0

Correct Answers: B, D

- Which register(s) are involved in enabling and configuring INT0?
 - EIMSK
 - EICRA
 - TIMSK
 - GICR

Correct Answers: A, B

- After executing the `setup_external_interrupt()` function, what are the correct values of the following registers?
 - DDRD = 0x00
 - PORTD = 0x04
 - EICRA = 0x03

- EICRA = 0x02
- EIMSK = 0x01
- PORTD = 0x00
- DDRD = 0xFF
- EIMSK = 0x02

Correct Answers: A, B, D, E

- After executing the `setup_external_interrupt()` function, which of the following are correct?
 - Bit 2 of **DDRD** is cleared to configure PD2 as input
 - Bit 2 of **PORTD** is set to enable pull-up on PD2
 - Bits **ISC01=1** and **ISC00=0** set INT0 to trigger on falling edge
 - Bit 0 of **EIMSK** is set to enable INT0

Correct Answers: A, B, C, D

- What is the role of the ISR of the external interrupt?
 - It is automatically called when INT0 interrupt is triggered
 - It toggles the state of pin PB0 (on/off)
 - It continuously turns the LED on
 - It requires manual calling from the main loop
 - It changes the value of PORTB0 every time INT0 is triggered
 - It disables the global interrupt flag
 - It can be used to blink an LED when a button is pressed (connected to PD2)
 - It toggles all bits of PORTB

Correct Answers: A, B, E, G

- What is the purpose of the `setup_led_output()` function?
 - It sets pin PB0 as an output pin
 - It enables the internal pull-up resistor on PB0
 - It prepares PB0 to control an LED or output device
 - It clears bit 0 of DDRB
 - It modifies the data direction of PB0
 - It should be called before toggling PB0 with PORTB
 - It sets the value of PB0 to HIGH

Correct Answers: A, C, E, F

Given the UART data frame in Table 01

10. What is the minimum number of bits in a UART data frame?
- A. 5
 - B. 9
 - C. 6
 - D. 8
 - E. 4

Correct Answer: B

11. Which of the following are configurable in a UART transmission?
- A. Number of stop bits
 - B. Number of data bits
 - C. Baud rate
 - D. Type of clock (internal or external)
 - E. Frame start voltage

Correct Answers: A, B, C

12. How is the start of a UART frame identified?
- A. By a rising edge (LOW to HIGH)
 - B. By a falling edge (HIGH to LOW)
 - C. When the data line goes LOW while idle
 - D. When the parity bit is detected

Correct Answers: B, C

Given the listing 03

13. What is the main purpose of the function `USART_Init()` in this code?
- A. To configure GPIO pins
 - B. To initialize UART with a specific baud rate and frame format
 - C. To enable Timer interrupts
 - D. To configure UART receive and transmit
 - E. To send data through UART
 - F. To set global interrupts

Correct Answers: B, D, F

14. Which registers are configured to set baud rate in this code?
- A. UBRR0H
 - B. UBRR0L
 - C. UCSR0B
 - D. UCSR0C
 - E. UDR0
 - F. TCCR0A

Correct Answers: A, B

15. What does the ISR `USART_RX_vect` do in this code?
- A. It sends a character over UART
 - B. It turns ON the LED when character 'a' is received
 - C. It toggles the LED

- D. It stores received data in a buffer
- E. It turns OFF the LED if character is not 'a'

Correct Answers: B, D, E

What is the frame format of the USART based on this configuration?

- A. 8 data bits, no parity, 1 stop bit
- B. 7 data bits, even parity, 2 stop bits
- C. 8 data bits, even parity, 1 stop bit
- D. 5 data bits, no parity, 1 stop bit
- E. 6 data bits, no parity, 2 stop bits
- F. 8 data bits, no parity, 2 stop bits

Correct Answer: A

17. What does the line do?

```
UCSR0B |= RX_COMPLETE_INTERRUPT;
```

- A. Enables RX interrupt
- B. Enables TX interrupt
- C. Enables interrupt when UART receives data
- D. Disables data register empty interrupt
- E. Causes the ISR `USART_RX_vect` to execute upon reception
- F. Immediately reads from UART buffer

Correct Answers: A, C, E

18. What would happen if you send character 'a' over UART to this AVR?

- A. LED on pin PB5 will turn on
- B. LED on pin PB5 will turn off
- C. The ISR will never be triggered
- D. The character will be lost because no buffer is implemented
- E. `USART_ReceiveBuffer` will contain 'a'
- F. Nothing will happen because TX is disabled

Correct Answers: A, E

19. What does this macro definition do?

```
#define BAUD_PRESCALER (((F_CPU / (USART_BAUDRATE * 16UL))) - 1)
```

- A. Sets the number of stop bits in the UART frame
- B. Calculates the value to load into UBRR0H and UBRR0L for baud rate configuration
- C. Sets the UART baud rate directly
- D. Computes the prescaler required for the USART baud rate generator
- E. Disables UART transmission temporarily
- F. Calculates the timer prescaler for a delay loop

Correct Answers: B, D

20. What is the purpose of the `>> 8` shift in:
`UBRR0H = BAUD_PRESCALER >> 8;`
`UBRR0L = BAUD_PRESCALER;`
- A. It isolates the lower 8 bits of the `BAUD_PRESCALER`
 - B. It isolates the upper 8 bits of the `BAUD_PRESCALER`
 - C. It divides the prescaler value by 8
 - D. It prepares the value for the high byte register (`UBRR0H`)
 - E. It shifts bits to configure the stop bits
 - F. It is needed because the `UBRR` register is 16 bits, split into two 8-bit registers

Correct Answers: B, D, F

21. Why is `sei()` called in `USART_Init()`?
- A. To enable global interrupts
 - B. To enable USART transmitter
 - C. To enable only the USART RX interrupt
 - D. To allow the ISR `USART_RX_vect` to execute when data is received
 - E. To enable UART RX pin
 - F. To disable sleep mode

Correct Answers: A, D

22. What is the function of `UDR0` in the USART system?
- A. Stores the baud rate value
 - B. Used to send and receive UART data
 - C. Holds received data temporarily
 - D. Is the USART status register
 - E. Contains frame format settings
 - F. Stores interrupt vectors

Correct Answers: B, C

23. In the line `PORTB |= 1 << 5;`, what happens?
- A. PB5 is set to logic HIGH
 - B. PB5 is toggled
 - C. PB5 is cleared
 - D. Only bit 5 of `PORTB` is affected
 - E. All bits of `PORTB` are overwritten
 - F. A pull-up resistor is enabled on PB5

Correct Answers: A, D

24. Which of the following are true about using interrupt-based UART reception instead of polling?
- A. Reduces CPU usage during idle time
 - B. Can miss incoming data if interrupts are not enabled
 - C. Automatically stores received bytes into a circular buffer
 - D. Allows execution of other code while waiting for data
 - E. Requires polling of `RXC0` flag

- F. Can trigger `USART_RX_vect` even during sleep modes (if allowed by hardware)

Correct Answers: A, B, D, F

Which of the following conditions must be met for `USART_RX_vect` to trigger?

- A. Global interrupts must be enabled (`sei()`)
- B. `RXEN0` must be set to enable receiver
- C. The baud rate must match on both sender and receiver
- D. Data must be received and fully shifted into `UDR0`

Correct Answers: A, B, C, D,

Given Listing 02 on timer interrupt

Why `loop()` is empty in this code?

- A. All work is handled by interrupts
- B. It's an error
- C. It's waiting for a button press
- D. It's a timer-based, interrupt-driven program
- E. The loop runs but does nothing

Correct Answers: A, D, E

27. What is the purpose of `OCR1A += 16384;` in `TIMER1_COMPA_vect` ISR?
- A. It resets the timer
 - B. It schedules the next compare match event
 - C. It creates a fixed delay between interrupts
 - D. It disables the overflow interrupt
 - E. It changes the timer mode

Correct Answers: B, C

28. What triggers the `ISR(TIMER1_COMPA_vect)` handler?
- A. Timer1 reaching `OCR1A` value
 - B. Timer1 overflowing
 - C. Output Compare A Match
 - D. Serial data reception
 - E. Enables Global interrupts

Correct Answers: A, C

29. Under what conditions will `TIMER1_OVF_vect` trigger?
- A. When Timer1 value exceeds 65535
 - B. When Timer1 reaches `OCR1A`
 - C. When prescaler is disabled
 - D. Continuously with regular intervals based on prescaler
 - E. Only when Timer1 is in overflow mode

Correct Answers: A, D

30. What happens if you remove the line `OCR1A += 16384;` from `TIMER1_COMPA_vect`?

- A. Compare match ISR will never trigger again
- B. Compare match will occur repeatedly at fixed intervals
- C. Compare match will keep firing continuously
- D. Timer will behave like in normal mode
- E. ISR may block Serial output

Correct Answers: B