CEFET-MG — Centro Federal de Educação Tecnológica de Minas Gerais

DEPARTAMENTO DE COMPUTAÇÃO DE DIVINÓPOLIS — DECOM-DV

Microprocessadores e Microcontroladores

Primeira Atividade Avaliativa

Aluno: JOÃO PEDRO MARTINS ESPÍNDOLA

Valor: 30 pontos (cada questão vale 5 pontos)

Turma: 2024/1

Prof. M. Sc. Diego Ascânio Santos

Respostas:

1	2	3	4	5	6

Questão 1

Acerca do ATmega328P, microcontrolador presente no arduino UNO e NANO, avalie as assertivas:

I. A CPU do ATmega328P é baseada na arquitetura Harvard com um conjunto complexo de instruções (CISC);

II. O ATmega328P possui 32KB de memória flash;

III. A CPU do ATmega328P é baseada na arquitetura Von Neummann com um conjunto reduzido de instruções (RISC);

IV. É possível usar as 6 entradas analógicas do ATmega328P simultâneamente;

V. Para qualquer pino presente no microcontrolador, o ATmega328P não permite que um mesmo pino possa executar mais de uma função;

VI. Exceder os limites operacionais de tensão e corrente do ATmega328P não danifica o microcontrolador.

Quais estão corretas?

a) Nenhuma das assertivas.

b) II.

c) II, IV, V.

d) II, IV, V, VI.

e) Todas as assertivas.

Questão 2

Considere o código abaixo:

```
const int ledPin = 13;
const int interruptPin = 2; // only pin 2 and 3 can be used for interrupts

volatile int state = LOW;

void blink() { // ISR function
    state = !state; // toggle the state
}
```

```
// missing setup function

void loop() {
    digitalWrite(ledPin, state);
}
```

É desejado que o LED conectado ao pino 13 comute de estado a cada vez que o botão conectado ao pino 2 for pressionado. O *pushbutton* conectado ao pino 2 do arduino também está conectado ao pino `GND` do microcontrolador.

Qual alternativa contém a implementação da função `setup()` que atende a esse requisito no Arduino UNO?

a)

```
void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(ledPin, OUTPUT);
    attachInterrupt(
        digitalPinToInterrupt(interruptPin),
        blink,
        KEEPING
    );
}
```

b)

```
void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(ledPin, OUTPUT);
    attachInterrupt(
        digitalPinToInterrupt(interruptPin),
        blink,
        CHANGE
    );
}
```

c)

```
void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(ledPin, OUTPUT);
    attachInterrupt(
        digitalPinToInterrupt(interruptPin),
        blink,
        HIGH
    );
}
```

d)

```
void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(ledPin, OUTPUT);
    attachInterrupt(
        digitalPinToInterrupt(interruptPin),
        blink,
        FALLING
    );
}
```

e)

```
void setup() {
    pinMode(2, INPUT_PULLUP);
    pinMode(ledPin, OUTPUT);
    attachInterrupt(
        digitalPinToInterrupt(interruptPin),
        blink,
        LOW
    );
}
```

Questão 3

A respeito de entradas e saídas digitais do Arduino, resistores pull-up e pull-down, contatos normalmente abertos e normalmente fechados, avalie as assertivas:

I. Por padrão as entradas digitais do Arduino estão preparadas para receber sinais digitais em nível

lógico TTL (0V a 5V).

II. Não é necessário realizar quaisquer tipos de adaptações para conectar circuitos digitais não-TTL (por exemplo, CMOS) ao Arduino.

III. Um contato normalmente aberto é um contato que, em repouso, não permite a passagem de corrente elétrica.

IV. O nível lógico de uma entrada digital do Arduino conectada a um contato normalmente aberto em seu estado de repouso é sempre 0V.

V. Resistores pull-up são utilizados para garantir que uma entrada digital do Arduino esteja sempre em nível lógico alto em seu estado padrão.

VI. O Arduino não dispõe de resistores pull-up internos, sendo necessário adicionar resistores externos para este fim.

São verdadeiras as assertivas:

a) I, II, III, IV, V e VI.

b) I, III, IV, V e VI.

c) I, III, IV e V.

d) II e V.

e) II e VI.

Questão 4

A respeito das fontes de interrupção no Arduino avalie as seguintes afirmativas:

I. O temporizador do Arduino pode ser configurado para gerar interrupções em intervalos regulares de tempo.

II. O ADC pode ser configurado para gerar uma interrupção quando a conversão analógico-digital é concluída.

III. Quaisquer pinos do Arduino UNO podem ser configurados como pinos de interrupção externos.

IV. Não é possível associar interrupções para o recebimento de dados pela porta serial.

Quais são verdadeiras?

a) I, II, III e IV.

b) I, II e III.

c) I, II e IV.

d) I, III e IV.

e) Apenas III.

Questão 5

Leia o seguinte código em Arduino:

```
int greenLED = 3, redLED = 4;
int leftButton = 6, rightButton = 7;

void setup() {
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  pinMode(leftButton, INPUT_PULLUP);
  pinMode(rightButton, INPUT_PULLUP);
}

void loop() {
  while (digitalRead(leftButton) == LOW) {
    digitalWrite(redLED, HIGH);
  }
```

```
  while (digitalRead(rightButton) == LOW) {
    digitalWrite(greenLED, HIGH);
    delay(1000);
    digitalWrite(greenLED, LOW);
    delay(1000);
  }
  while (digitalRead(leftButton) == HIGH && digitalRead(rightButton) == HIGH) {
    digitalWrite(greenLED, LOW);
    digitalWrite(redLED, LOW);
  }
}
```

O que este programa faz?

a) Acende o LED verde quando `rightButton` é pressionado e apaga o LED vermelho quando `leftButton` é pressionado.

b) Apaga os LEDs se nenhuma das teclas for pressionada, acende o LED vermelho enquanto `leftButton` é pressionado e pisca o LED verde de 1 em 1 segundo enquanto `rightButton` é pressionado.

c) Acende o LED verde quando `rightButton` é pressionado e apaga o LED vermelho quando `leftButton` é pressionado, mas não apaga os LEDs se nenhuma das teclas for pressionada.

d) Apaga os LEDs se as teclas forem processionadas, acende o LED vermelho enquanto `leftButton` não é pressionado e pisca o LED verde de 1 em 1 segundo enquanto `rightButton` não é pressionado.

e) Nenhuma das alternativas anteriores, pois, não existe configuração de entrada do tipo `INPUT_PULLUP`.

Questão 6

Raul deseja fazer um LED comutar de estado a cada 5 segundos. Sem muito critério do entendimento viu nos códigos dos slides do prof. Ascânio o seguinte ISR que fazia o LED comutar de estado a cada 5 segundos:

Rotina de interrupção de Overflow associada ao `Timer2`

```
ISR(TIMER2_OVF_vect) {
    overflows ++;
    if (overflows == 306) {
        overflows = 0;
        // Toggle the LED state
        digitalWrite(13, !digitalRead(13));
    }
}
```

Entretanto, ao copiar os códigos, não se atentou ao modo correto de funcionamento e configurou — sem querer — o prescaler do `Timer2` para o fator de 256, através do registrador de controle `TCCR2B` pela seguinte instrução na função `setup()`:

```
    TCCR2B = 0b00000110;
```

Com isso, em vez do LED comutar de estado a cada 5 segundos, ele comutava a cada 1.25 segundos (aproximadamente). Qual dos seguintes modos de *prescaling* do `Timer2` faz com que o LED comute de estado a cada 5 segundos, como deseja Raul?

a) `TCCR2B = 0b00000001;`

b) `TCCR2B = 0b00000111;`

c) `TCCR2B = 0b00000011;`

d) `TCCR2B = 0b00000000;`

e) `TCCR2B = 0b00000010;`