|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **SPRAWOZDANIE** | | | | | | **PROSZĘ PODAĆ NR GRUPY:** | | | | | | | |
| **ZIISS1** | | | **3** | **5** | **1** | **2** | **IO** |
| **IMIĘ** | **NAZWISKO** | **Temat ćwiczenia zgodny z wykazem tematów:** | **PONIŻEJ PROSZĘ PODAĆ TERMIN ZAJĘĆ:** | | | | | **ROK:** | | | | | |
| **Podstawy tworzenia projektu i obsługa portów obsługa przerwań** | **2023 r.** | | | | | |
| **MICHAŁ** | **WARSZAWSKI** | **PN** | **WT** | **SR** | | **CZ** | | **PT** | | **SB** | | **ND** |
| **GODZINA ROZPOCZĘCIA ZAJĘĆ:** | | | | | | | | **11 : 30** | | |
| UWAGA !!! Wypełniamy tylko białe pola. W **punkcie 1**, proszę zakreślić odpowiednie pola i podać godzinę w której odbywają się zajęcia, zgodnie z planem zajęć. | | | | | | | | | | | | | |

Wprowadzenie teoretyczne:

**Opisz rodzaje płyt głównych Arduino i procesorów zaimplementowanych na tych płytach   
(Opisz symbole na płytach)**

1. **Rodzaje płyt głównych Arduino:**

**Arduino Uno:**

Procesor: ATmega328P

Wejścia/Wyjścia cyfrowe: 14

Wejścia analogowe: 6

Zegar: 16 MHz

Pamięć Flash: 32 KB

Pamięć RAM: 2 KB

**Arduino Mega:**

Procesor: ATmega2560

Wejścia/Wyjścia cyfrowe: 54

Wejścia analogowe: 16

Zegar: 16 MHz

Pamięć Flash: 256 KB

Pamięć RAM: 8 KB

**Arduino Nano:**

Procesor: ATmega328P

Wejścia/Wyjścia cyfrowe: 14

Wejścia analogowe: 8

Zegar: 16 MHz

Pamięć Flash: 32 KB

Pamięć RAM: 2 KB

**Arduino Due:**

Procesor: Atmel SAM3X8E (ARM Cortex-M3)

Wejścia/Wyjścia cyfrowe: 54

Wejścia analogowe: 12

Zegar: 84 MHz

Pamięć Flash: 512 KB

Pamięć RAM: 96 KB

**Arduino Leonardo:**

Procesor: ATmega32u4

Wejścia/Wyjścia cyfrowe: 20

Wejścia analogowe: 12

Zegar: 16 MHz

Pamięć Flash: 32 KB

Pamięć RAM: 2.5 KB

1. **Rodzaje procesorów zaimplementowanych na płytach Arduino:**

* **ATmega328:**

Rodzaj: Mikrokontroler AVR 8-bitowy.

Płyty Arduino: Arduino Uno, Arduino Nano.

Charakterystyka: Posiada 32 KB pamięci flash, 2 KB RAM, 14 cyfrowych pinów wejścia/wyjścia (w tym 6 z funkcją PWM) oraz 6 analogowych wejść.

* **ATmega2560:**

Rodzaj: Mikrokontroler AVR 8-bitowy.

Płyty Arduino: Arduino Mega.

Charakterystyka: Dużo większa pamięć niż ATmega328 - 256 KB flash, 8 KB RAM. Posiada 54 cyfrowe piny wejścia/wyjścia (w tym 15 z funkcją PWM) oraz 16 analogowych wejść.

* **ATtiny85:**

Rodzaj: Mikrokontroler AVR 8-bitowy.

Płyty Arduino: Arduino Gemma, Arduino Trinket.

Charakterystyka: Mniejszy od ATmega, ale nadal potężny mikrokontroler z 8 KB flash, 512 B RAM. Zazwyczaj wykorzystywany w mniejszych projektach z ograniczonymi zasobami.

* **SAMD21:**

Rodzaj: Mikrokontroler ARM Cortex-M0+ 32-bitowy.

Płyty Arduino: Arduino Zero, Arduino MKR series.

Charakterystyka: Współczesny mikrokontroler 32-bitowy z 256 KB flash, 32 KB RAM. Posiada zaawansowane funkcje, takie jak interfejs USB, multipleksowane wejścia analogowe, i więcej.

* **ESP8266:**

Rodzaj: Mikrokontroler WiFi.

Płyty Arduino: NodeMCU, WeMos D1.

Charakterystyka: Oprócz standardowych funkcji mikrokontrolera, posiada wbudowany moduł WiFi, co umożliwia łatwe tworzenie projektów IoT.

* **ESP32:**

Rodzaj: Mikrokontroler WiFi i Bluetooth.

Płyty Arduino: Arduino ESP32.

Charakterystyka: Podobnie jak ESP8266, ale bardziej zaawansowany. Posiada dwurdzeniowy procesor, więcej pamięci, obsługę Bluetooth.

**Zadanie 1**

**Tutaj wklej zrzut ekranu Twojego projektu:**

|  |
| --- |
|  |

**Tutaj wklej listing Twojego programu**

|  |
| --- |
| void setup() {  pinMode(A15, OUTPUT);  pinMode(A14, OUTPUT);  pinMode(A13, OUTPUT);  pinMode(21, INPUT);  pinMode(20, INPUT);  pinMode(19, INPUT);  pinMode(2, INPUT\_PULLUP);  pinMode(3, INPUT\_PULLUP);  pinMode(A2, INPUT\_PULLUP);  pinMode(4, INPUT\_PULLUP);  pinMode(9, INPUT\_PULLUP);  pinMode(A1, INPUT\_PULLUP);  digitalWrite(A1, HIGH);  digitalWrite(A1, HIGH);  digitalWrite(A1, HIGH);  }  void loop() {  if(digitalRead(4) == LOW || digitalRead(9) == LOW || digitalRead(A1) == LOW){  pinMode(20, OUTPUT);  pinMode(19, OUTPUT);  pinMode(21, OUTPUT);  digitalWrite(A15, HIGH);  digitalWrite(A14, HIGH);  digitalWrite(A13, HIGH);  }  else{  pinMode(21, INPUT);  pinMode(20, INPUT);  pinMode(19, INPUT);  digitalWrite(A15, LOW);  digitalWrite(A14, LOW);  digitalWrite(A13, LOW);  }  } |

**Zadanie 2**

**Tutaj wklej zrzut ekranu projektu:**

|  |
| --- |
|  |

**Tutaj wklej listing programu:**

|  |
| --- |
| void setup() {  pinMode(A15, OUTPUT);  pinMode(A14, OUTPUT);  pinMode(A13, OUTPUT);  pinMode(21, INPUT);  pinMode(20, INPUT);  pinMode(19, INPUT);  pinMode(2, INPUT\_PULLUP);  pinMode(3, INPUT\_PULLUP);  pinMode(A2, INPUT\_PULLUP);  }  void loop() {  if(digitalRead(4) == LOW || digitalRead(9) == LOW || digitalRead(A1) == LOW){  pinMode(20, OUTPUT);  pinMode(19, OUTPUT);  pinMode(21, OUTPUT);  digitalWrite(A15, HIGH);  digitalWrite(A14, HIGH);  digitalWrite(A13, HIGH);  }  else{  pinMode(21, INPUT);  pinMode(20, INPUT);  pinMode(19, INPUT);  digitalWrite(A15, LOW);  digitalWrite(A14, LOW);  digitalWrite(A13, LOW);  }  } |

**Zadanie 3**

**Tutaj wklej listing programu**

|  |
| --- |
| bool toogle = false;  void setup() {  pinMode(A15, OUTPUT);  pinMode(A14, OUTPUT);  pinMode(A13, OUTPUT);  pinMode(21, INPUT);  pinMode(20, INPUT);  pinMode(19, INPUT);  pinMode(2, INPUT\_PULLUP);  pinMode(3, INPUT\_PULLUP);  pinMode(A2, INPUT\_PULLUP);  pinMode(13, OUTPUT);  attachInterrupt(4, funkcjaPrzerwania, RISING);  }  void loop() {  if(digitalRead(4) == LOW || digitalRead(9) == LOW || digitalRead(A1) == LOW){  pinMode(20, OUTPUT);  pinMode(19, OUTPUT);  pinMode(21, OUTPUT);  digitalWrite(A15, HIGH);  digitalWrite(A14, HIGH);  digitalWrite(A13, HIGH);  }  else{  pinMode(21, INPUT);  pinMode(20, INPUT);  pinMode(19, INPUT);  digitalWrite(A15, LOW);  digitalWrite(A14, LOW);  digitalWrite(A13, LOW);  }  }  void funkcjaPrzerwania() {  toogle = !toogle;  digitalWrite(13, toogle);  } |

**Wnioski:**

**Zadanie 1:**

W programie są zdefiniowane piny jako wejścia i wyjścia.

Ustawione są rezystory podciągające dla niektórych pinów.

W pętli loop program sprawdza stan trzech pinów (4, 9, A1) i w zależności od tego ustawia stan trzech innych pinów (A15, A14, A13).

Jeśli którykolwiek z pinów 4, 9, A1 jest niski, ustawiane są wyjścia A15, A14, A13 na wysoki stan, w przeciwnym razie są ustawiane na niski stan.

**Zadanie 2:**

Program jest bardzo podobny do Zadania 1, ale brakuje definicji rezystorów podciągających dla pinów 2, 3, A2. Wynika to z innego podłączenia układów.

**Zadanie 3:**

Dodany jest pin 13 jako dioda LED, która zmienia stan w funkcji przerwania.

Funkcja przerwania (funkcjaPrzerwania) zmienia stan diody LED na pinie 13 przy każdym zboczu narastającym na pinie 4.