



|  |  |  |  |
| --- | --- | --- | --- |
| 3 | 2 | 1 | 0 |
| 7 | 6 | 5 | 4 |
| 11 | 10 | 9 | 8 |
| 15 | 14 | 13 | 12 |

Sunt in total 16 coloane/4 straturi.

|  |  |
| --- | --- |
| **void cub\_inchidere()**  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }      for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 0);  }  } | Sunt in toatal 16 coloane  -sting toate LED-urile  As fi putut sa fac astfel:  digitalWrite(column[0], 0);  digitalWrite(column[1], 0);  digitalWrite(column[2], 0);  digitalWrite(column[3], 0);  digitalWrite(column[4], 0);  digitalWrite(column[5], 0);  digitalWrite(column[6], 0);  digitalWrite(column[7], 0);  digitalWrite(column[8], 0);  digitalWrite(column[9], 0);  digitalWrite(column[10], 0);  digitalWrite(column[11], 0);  digitalWrite(column[12], 0);  digitalWrite(column[13], 0);  digitalWrite(column[14], 0);  digitalWrite(column[15], 0);  **Sting LED-urile de pe cele 4 straturi**  As fi putut face astfel:  digitalWrite(layer[0], 0);  digitalWrite(layer[1], 0);  digitalWrite(layer[2], 0);  digitalWrite(layer[3], 0); |

|  |  |
| --- | --- |
| void cub\_deschidere()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  } | Sunt in toatal 16 coloane  -sting toate LED-urile  As fi putut sa fac astfel:  digitalWrite(column[0], 0);  digitalWrite(column[1], 0);  digitalWrite(column[2], 0);  digitalWrite(column[3], 0);  digitalWrite(column[4], 0);  digitalWrite(column[5], 0);  digitalWrite(column[6], 0);  digitalWrite(column[7], 0);  digitalWrite(column[8], 0);  digitalWrite(column[9], 0);  digitalWrite(column[10], 0);  digitalWrite(column[11], 0);  digitalWrite(column[12], 0);  digitalWrite(column[13], 0);  digitalWrite(column[14], 0);  digitalWrite(column[15], 0);  **Aprind LED-urile de pe cele 4 straturi**  As fi putut face astfel:  digitalWrite(layer[0], 1);  digitalWrite(layer[1], 1);  digitalWrite(layer[2], 1);  digitalWrite(layer[3], 1); |

|  |  |  |
| --- | --- | --- |
|  | void parcurgere\_straturi\_sus\_jos()  {  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  delay(100);  digitalWrite(layer[i], 0);  delay(100);  }  } | Se merge de sus de la stratul A0, în jos la stratul A3  (i scade cu 1 ⇔i++)  i=0 A0  i=1 A1  i=2 A2  i=3 A3  -se aprinde stratul, apoi se stinge stratul |
|  | void parcurgere\_straturi\_jos\_sus()  {  for(int i = 3; i>=0; i--)  {  digitalWrite(layer[i], 1);  delay(100);  digitalWrite(layer[i], 0);  delay(100);  }  } | Se merge de jos, de la stratul A3, în sus la stratul A0  (i scade cu 1 ⇔i--)  i=3 A3  i=2 A2  i=1 A1  i=0 A0  -se aprinde stratul, apoi se stinge stratul |

|  |  |
| --- | --- |
| void parcurgere\_orizontala()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  //ordine parcurgere: 0, 1, 2, 3, 7, 6, 5, 4, 8, 9, 10, 11, 15, 14, 13, 12    digitalWrite(column[0], 1);  delay(100);  digitalWrite(column[1], 1);  delay(100);  digitalWrite(column[2], 1);  delay(100);  digitalWrite(column[3], 1);  delay(100);  digitalWrite(column[7], 1);  delay(100);  digitalWrite(column[6], 1);  delay(100);  digitalWrite(column[5], 1);  delay(100);  digitalWrite(column[4], 1);  delay(100);  digitalWrite(column[8], 1);  delay(100);  digitalWrite(column[9], 1);  delay(100);  digitalWrite(column[10], 1);  delay(100);  digitalWrite(column[11], 1);  delay(100);  digitalWrite(column[15], 1);  delay(100);  digitalWrite(column[14], 1);  delay(100);  digitalWrite(column[13], 1);  delay(100);  digitalWrite(column[12], 1);  delay(100);  } |  |
| void parcurgere\_verticala()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  //ordine parcurgere: 0, 4, 8, 12, 13, 9, 5, 1, 2, 6, 10, 14, 15, 11, 7, 3    digitalWrite(column[0], 1);  delay(100);  digitalWrite(column[4], 1);  delay(100);  digitalWrite(column[8], 1);  delay(100);  digitalWrite(column[12], 1);  delay(100);  digitalWrite(column[13], 1);  delay(100);  digitalWrite(column[9], 1);  delay(100);  digitalWrite(column[5], 1);  delay(100);  digitalWrite(column[1], 1);  delay(100);  digitalWrite(column[2], 1);  delay(100);  digitalWrite(column[6], 1);  delay(100);  digitalWrite(column[10], 1);  delay(100);  digitalWrite(column[14], 1);  delay(100);  digitalWrite(column[15], 1);  delay(100);  digitalWrite(column[11], 1);  delay(100);  digitalWrite(column[7], 1);  delay(100);  digitalWrite(column[3], 1);  delay(100);  } |  |
| void parcurgere\_spirala\_1()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }    for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  //ordine parcurgere: 0, 1, 2, 3 7, 11, 15, 14, 13, 12, 8, 4, 5, 6, 10, 9    digitalWrite(column[0], 1);  delay(100);  digitalWrite(column[1], 1);  delay(100);  digitalWrite(column[2], 1);  delay(100);  digitalWrite(column[3], 1);  delay(100);  digitalWrite(column[7], 1);  delay(100);  digitalWrite(column[11], 1);  delay(100);  digitalWrite(column[15], 1);  delay(100);  digitalWrite(column[14], 1);  delay(100);  digitalWrite(column[13], 1);  delay(100);  digitalWrite(column[12], 1);  delay(100);  digitalWrite(column[8], 1);  delay(100);  digitalWrite(column[4], 1);  delay(100);  digitalWrite(column[5], 1);  delay(100);  digitalWrite(column[6], 1);  delay(100);  digitalWrite(column[10], 1);  delay(100);  digitalWrite(column[9], 1);  delay(100);  } |  |
| void parcurgere\_spirala\_2()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  //ordine parcurgere> 0, 4, 8, 12, 13, 14, 15, 11, 7, 3, 2, 1, 5, 9, 10, 6    digitalWrite(column[0], 1);  delay(100);  digitalWrite(column[4], 1);  delay(100);  digitalWrite(column[8], 1);  delay(100);  digitalWrite(column[12], 1);  delay(100);  digitalWrite(column[13], 1);  delay(100);  digitalWrite(column[14], 1);  delay(100);  digitalWrite(column[15], 1);  delay(100);  digitalWrite(column[11], 1);  delay(100);  digitalWrite(column[7], 1);  delay(100);  digitalWrite(column[3], 1);  delay(100);  digitalWrite(column[2], 1);  delay(100);  digitalWrite(column[1], 1);  delay(100);  digitalWrite(column[5], 1);  delay(100);  digitalWrite(column[9], 1);  delay(100);  digitalWrite(column[10], 1);  delay(100);  digitalWrite(column[6], 1);  delay(100);  } |  |
| void parcurgere\_diagonala\_1()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  //ordine parcurgere: 3, 7, 2, 11, 6, 1, 15, 10, 5, 0, 14, 9, 4, 13, 8, 12    digitalWrite(column[3], 1);  delay(100);  digitalWrite(column[7], 1);  delay(100);  digitalWrite(column[2], 1);  delay(100);  digitalWrite(column[11], 1);  delay(100);  digitalWrite(column[6], 1);  delay(100);  digitalWrite(column[1], 1);  delay(100);  digitalWrite(column[15], 1);  delay(100);  digitalWrite(column[10], 1);  delay(100);  digitalWrite(column[5], 1);  delay(100);  digitalWrite(column[0], 1);  delay(100);  digitalWrite(column[14], 1);  delay(100);  digitalWrite(column[9], 1);  delay(100);  digitalWrite(column[4], 1);  delay(100);  digitalWrite(column[13], 1);  delay(100);  digitalWrite(column[8], 1);  delay(100);  digitalWrite(column[12], 1);  delay(100);    } |  |
| void parcurgere\_diagonala\_2()  {  for(int i = 0; i<16; i++)  {  digitalWrite(column[i], 0);  }  for(int i = 0; i<4; i++)  {  digitalWrite(layer[i], 1);  }  //ordine parcurgere> 0, 4, 1, 8, 5, 2, 12, 9, 6, 2, 13, 10, 7, 14, 11, 15    digitalWrite(column[0], 1);  delay(100);  digitalWrite(column[4], 1);  delay(100);  digitalWrite(column[1], 1);  delay(100);  digitalWrite(column[8], 1);  delay(100);  digitalWrite(column[5], 1);  delay(100);  digitalWrite(column[2], 1);  delay(100);  digitalWrite(column[12], 1);  delay(100);  digitalWrite(column[9], 1);  delay(100);  digitalWrite(column[6], 1);  delay(100);  digitalWrite(column[3], 1);  delay(100);  digitalWrite(column[13], 1);  delay(100);  digitalWrite(column[10], 1);  delay(100);  digitalWrite(column[7], 1);  delay(100);  digitalWrite(column[14], 1);  delay(100);  digitalWrite(column[11], 1);  delay(100);  digitalWrite(column[15], 1);  delay(100);    } |  |