

This ULU

The *ULU.07 Universal 4-bit display* can be used to show a four-bit, binary value in several ways: as a bit pattern, a hexadecimal number and a signed or unsigned number. It is implemented using an Arduino Nano.

Used parts

The following standard parts are used:

1x black aluminum casing 50*80*20mm;
4x black M3 bolt 5mm, button head;
2x 7mm female/female brass M3 standoff;
1x 50*25mm 2.54mm prototype PCB;
2x 1-pole ON-ON switch;
1x 1-pole ON-OFF-ON switch;
1x power connector;

2x 4-bit data connector;
2x single digit 0.56" 7-segment display;
1x Arduino nano;
12x 1K2 resistor for the 7-segment display;
4x 10K pull down resistor;
1x 20 x 8-hole experiment PCB.

Specifications

Two 7-segment displays and an Arduino Nano are used to implement the necessary logic. Six display modes are available: 1. binary, 2. hexadecimal, 3. decimal, 4. signed, 5. 1-complement and 6. 2-complement. These modes are shown in Figure 1.

| | byte | 1 | 2 | 3 | 4 | 5 | 6 |
|----|------|----|---|-----|-----|-----|-----|
| 0 | 0000 | | 0 | 0 | 0 | 0 | 0 |
| 1 | 0001 | . | 1 | 1 | 1 | 1 | 1 |
| 2 | 0010 | . | 2 | 2 | 2 | 2 | 2 |
| 3 | 0011 | .. | 3 | 3 | 3 | 3 | 3 |
| 4 | 0100 | . | 4 | 4 | 4 | 4 | 4 |
| 5 | 0101 | . | 5 | 5 | 5 | 5 | 5 |
| 6 | 0110 | . | 6 | 6 | 6 | 6 | 6 |
| 7 | 0111 | . | 7 | 7 | 7 | 7 | 7 |
| 8 | 1000 | . | 8 | 8 | - 0 | - 1 | - 8 |
| 9 | 1001 | . | 9 | 9 | - 1 | - 6 | - 7 |
| 10 | 1010 | . | A | 1 0 | - 2 | - 5 | - 6 |
| 11 | 1011 | . | b | 1 1 | - 3 | - 4 | - 5 |
| 12 | 1100 | .. | C | 1 2 | - 4 | - 3 | - 4 |
| 13 | 1101 | .. | d | 1 3 | - 5 | - 2 | - 3 |
| 14 | 1110 | .. | E | 1 4 | - 6 | - 1 | - 2 |
| 15 | 1111 | .. | F | 1 5 | - 7 | 0 | - 1 |
| •• | | | | | | • | • • |

Figure 1 – Display mode

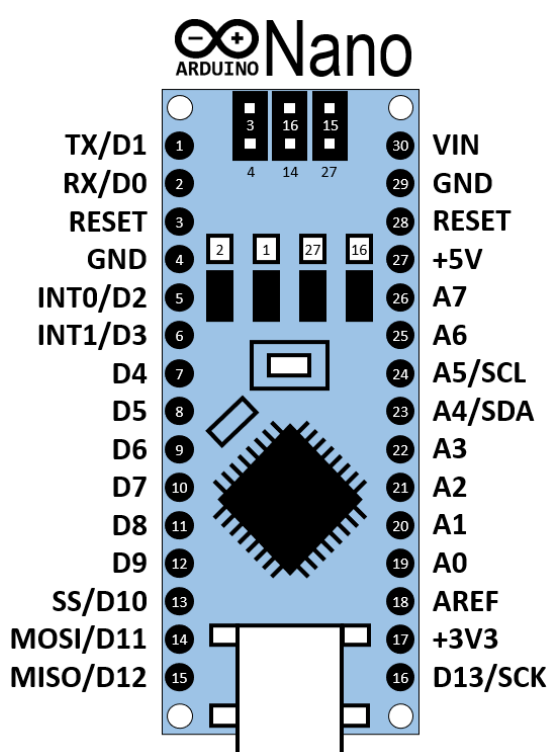


Figure 2 – Arduino Nano pinout



Only one wire is used to connect the 4-bit connectors to each other and to the Arduino. In order to do this, a small segment in the middle of the wire is stripped. The input marked with pull up (pu) is directly connected to the Arduino, because the internal pull up resistor is used. As result of that, the witches must be fed with the ground (-). When both A5 and A6 of the Arduino Nano are connected to the ground, a test program will run. Then the display will first show r.l. 8. and then start counting from 0 to F repeatedly.

Also a solder check program and test mode is available to check the soldering and find possible solder or connection errors. This program contains two parts. First it will turn the left display on, followed by the right display. Segment by segment, in a clockwise manner. Then it will show the inputs on the right display, as shown in Figure 8, left part.

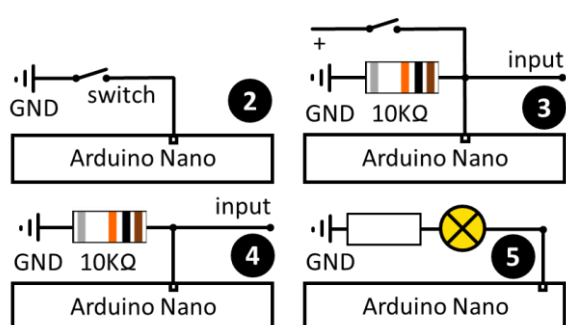


Figure 7 – Arduino interfaces

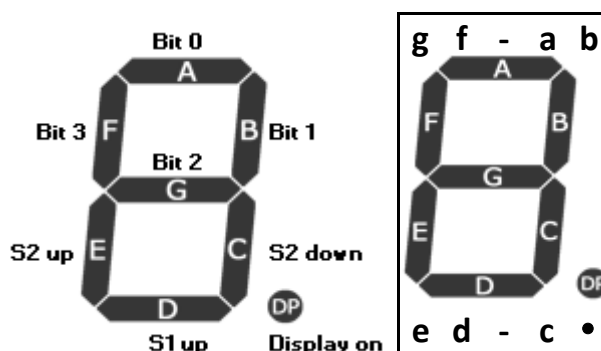


Figure 8 – Solder check display & 7-segment pinout

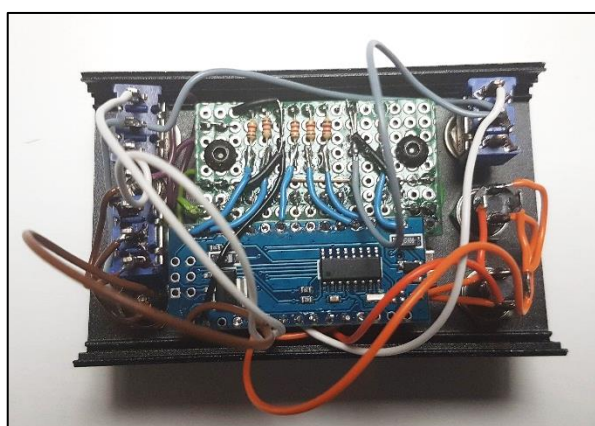


Figure 9 – ULU inside

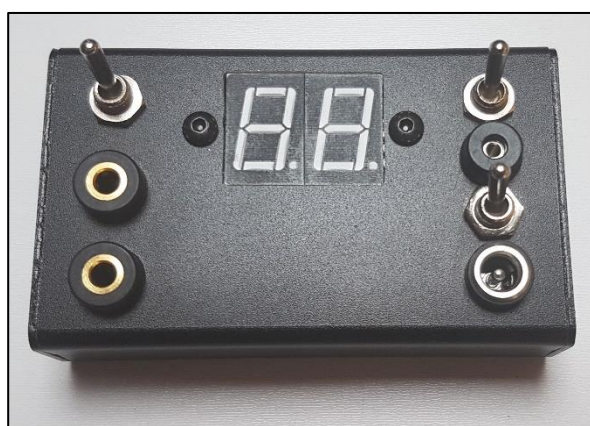


Figure 10 – Finished ULU

Usage

The two top switches are used to select one of the six different display modes:

Left switch is set to top

1. Binary
2. Hexadecimal
3. Decimal

Left switch is set to bottom

4. Signed
5. 1-complement
6. 2-complement

Right top switch is set to top position

Right top switch is set to neutral position

Right top switch is set to bottom position

When the 1- complement code is used the right decimal point is switched on, when the 2-complement code is used both decimal points are switched on. The display can be turned on by both a switch and a signal connector. The right bottom switch is used to switch the display on (bottom position) or off (top position). The black input connector has the same function: if a logical 1 is offered, the display is switched on. There is an OR relation between switch and connector: either of them can switch the display on. That makes it possible to use the display as a controlled output device, connected to a data

bus. It can also be used to for zero suppression: the zero is only shown when the keyboard is pressed or the data is entered.

Both the data bus connectors are interconnected. That offers the possibility to 'tap' a data-line and put the display between both ULUs connected to each other. When the enable switch is set up, the corresponding socket can be used to obtain a logical 1 for use in the circuit.

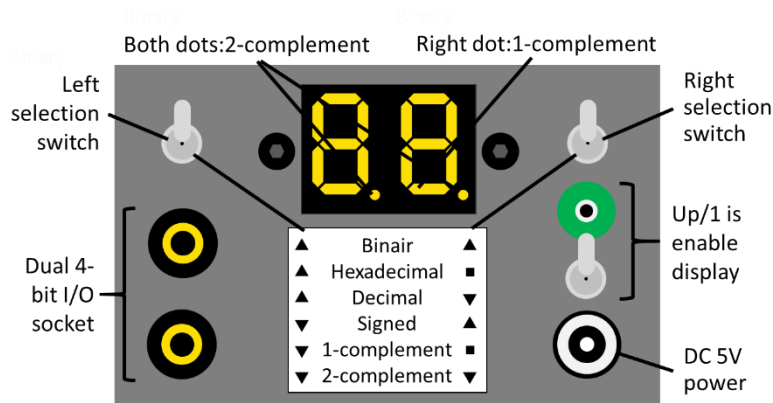


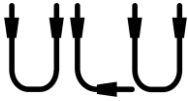
Figure 11 – Connectors & controls

Solder check program

```
/* ULU.05 Universal 4-bit 7-segment display - solder check */
/* CC BY-NC-SA Jeroen Brinkman */
int v0, v1, v2, v3, s1, s2u, s2d, s3u;

#define LB 10
#define LC 8
#define LE 11
#define LG 12
#define LP 7
#define RA 0
#define RB 1
#define RC 3
#define RD 5
#define RE 6
#define RF 4
#define RG 9
#define RP 2
#define RP 2
#define B0 A3
#define B1 A2
#define B2 A1
#define B3 A0
#define SW1D A4
#define SW2U A5
#define SW2D A6
#define SW3U A7

void setup() {
  /* Define I/O pins */
  pinMode(LB, OUTPUT); // left display, segment b
  pinMode(LC, OUTPUT); // left display, segment c
  pinMode(LE, OUTPUT); // left display, segment e
  pinMode(LG, OUTPUT); // left display, segment g
  pinMode(LP, OUTPUT); // left display, segment dp
  pinMode(RA, OUTPUT); // right display, segment a
  pinMode(RB, OUTPUT); // right display, segment b
  pinMode(RC, OUTPUT); // right display, segment c
  pinMode(RD, OUTPUT); // right display, segment d
  pinMode(RE, OUTPUT); // right display, segment e
  pinMode(RF, OUTPUT); // right display, segment f
  pinMode(RG, OUTPUT); // right display, segment g
  pinMode(RP, OUTPUT); // right display, segment dp
  pinMode(13, OUTPUT); // blinking led showing the programma's heartbeat
  pinMode(B3, INPUT); // Bit 3
  pinMode(B2, INPUT); // Bit 2
  pinMode(B1, INPUT); // Bit 1
  pinMode(B0, INPUT); // Bit 0
  pinMode(SW1D, INPUT_PULLUP);
  pinMode(SW2U, INPUT_PULLUP);
  pinMode(SW2D, INPUT_PULLUP);
  pinMode(SW3U, INPUT);
  /* test display */
  digitalWrite(LB, LOW);
  digitalWrite(LC, LOW);
  digitalWrite(LE, LOW);
  digitalWrite(LG, LOW);
  digitalWrite(LP, LOW);
  digitalWrite(RA, LOW);
}
```



```
digitalWrite(RB, LOW);
digitalWrite(RC, LOW);
digitalWrite(RD, LOW);
digitalWrite(RE, LOW);
digitalWrite(RF, LOW);
digitalWrite(RG, LOW);
digitalWrite(RP, LOW); delay(1000);
digitalWrite(LB, HIGH); delay(500);
digitalWrite(LC, HIGH); delay(500);
digitalWrite(LE, HIGH); delay(500);
digitalWrite(LG, HIGH); delay(500);
digitalWrite(LP, HIGH); delay(500);
digitalWrite(RA, HIGH); delay(500);
digitalWrite(RB, HIGH); delay(500);
digitalWrite(RC, HIGH); delay(500);
digitalWrite(RD, HIGH); delay(500);
digitalWrite(RE, HIGH); delay(500);
digitalWrite(RF, HIGH); delay(500);
digitalWrite(RG, HIGH); delay(500);
digitalWrite(RP, HIGH); delay(500);
digitalWrite(LB, LOW);
digitalWrite(LC, LOW);
digitalWrite(LE, LOW);
digitalWrite(LG, LOW);
digitalWrite(LP, LOW);
digitalWrite(RA, LOW);
digitalWrite(RB, LOW);
digitalWrite(RC, LOW);
digitalWrite(RD, LOW);
digitalWrite(RE, LOW);
digitalWrite(RF, LOW);
digitalWrite(RG, LOW);
digitalWrite(RP, LOW);
};

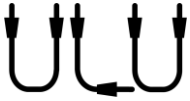
void loop(){
  digitalWrite(13, (millis() / 1000) % 2); //1s heartbeat for the onboard led
  /* Read switches & inputs */
  v0 = analogRead(B0) > 500 ? 1 : 0;
  v1 = analogRead(B1) > 500 ? 1 : 0;
  v2 = analogRead(B2) > 500 ? 1 : 0;
  v3 = analogRead(B3) > 500 ? 1 : 0;
  s1 = analogRead(SW1D) < 300 ? 1 : 0; // Switch 1 is on, switch uses pull down resistor so equals 0 when on
  s2u = analogRead(SW2U) < 300 ? 1 : 0; // Switch 2 is up, switch uses pull down resistor so equals 0 when on
  s2d = analogRead(SW2D) < 300 ? 1 : 0; // switch 2 is down, switch uses pull down resistor so equals 0 when on
  s3u = analogRead(SW3U) < 300 ? 0 : 1; // display is on, switch puts +5V to connector

  /* Display all inputs on right display */
  digitalWrite(RA, (v0 == 1) ? HIGH : LOW); // LSB
  digitalWrite(RB, (v1 == 1) ? HIGH : LOW); // bit 1
  digitalWrite(RG, (v2 == 1) ? HIGH : LOW); // bit 2
  digitalWrite(RF, (v3 == 1) ? HIGH : LOW); // MSB
  digitalWrite(RC, (s2d == 1) ? HIGH : LOW); // Switch 2 down
  digitalWrite(RE, (s2u == 1) ? HIGH : LOW); // Switch 2 up
  digitalWrite(RD, (s1 == 1) ? HIGH : LOW); // Switch 1 down
  digitalWrite(RP, (s3u == 1) ? HIGH : LOW); // DP display on
}
```

Program

```
/* ULU.05 Universal 4-bit 7-segment display - program code */
/* CC BY-NC-SA Jeroen Brinkman */

int value, setting, pointer, look, todo;
int show1[16][4] =
{{0,0,0,0},{0,0,0,1},{0,0,1,0},{0,0,1,1},{0,1,0,0},{0,1,0,1},{0,1,1,0},{0,1,1,1},{1,0,0,0},{1,0,0,1},{1,0,1,0},{1,0,1,1},{1,1,0,0},{1,1,0,1},{1,1,1,0},{1,1,1,1}};
int show2[8][7] =
{{1,1,1,1,1,1,0},{0,1,1,0,0,0,0},{1,1,0,1,1,0,1},{1,1,1,1,0,0,1},{0,1,1,0,0,1,1},{1,0,1,1,0,1,1},{1,0,1,1,1,1,1},{1,1,1,0,0,0,0}};
int show3[8][5][11] = { // values for Lb, Lc, Le, Lg, Ra, Rb, Rc, Rd, Re, Rf, Rg
{{0,0,0,0,1,1,1,1,1,1,1}, {0,0,0,0,1,1,1,1,1,1,1}, {0,0,0,1,1,1,1,1,1,1,0}, {0,0,0,1,1,1,1,1,0,0,0,0}, {0,0,0,1,1,1,1,1,1,1,1}},
{{0,0,0,0,1,1,1,1,0,1,1}, {0,0,0,0,1,1,1,1,0,1,1}, {0,0,0,1,0,1,1,1,0,0,0,0}, {0,0,0,1,0,1,1,1,1,1,1}, {0,0,0,1,1,1,1,0,0,0,0}},
{{0,0,0,0,1,1,1,0,1,1,1}, {1,1,0,0,1,1,1,1,1,1,0}, {0,0,0,1,1,1,0,1,1,0,1}, {0,0,0,1,1,0,1,1,0,1,1}, {0,0,0,1,1,0,1,1,1,1,1}},
{{0,0,0,0,0,0,1,1,1,1,1}, {1,1,0,0,0,1,1,0,0,0,0}, {0,0,0,1,1,1,1,1,0,0,1}, {0,0,0,1,0,1,1,0,0,1,1}, {0,0,0,1,0,1,1,0,1,1,1}},
{{0,0,0,0,1,0,0,1,1,1,0}, {1,1,0,0,1,0,1,1,0,1,1}, {0,0,0,1,0,1,1,0,0,1,1}, {0,0,0,1,1,1,1,1,0,0,1}, {0,0,0,1,0,1,1,0,0,1,1}},
{{0,0,0,0,0,0,1,1,1,0,1}, {1,1,0,0,1,1,1,0,0,0,1}, {0,0,0,1,1,0,1,1,0,0,1}, {0,0,0,1,1,0,1,1,0,1,1}, {0,0,0,1,1,1,1,0,0,0,1}},
{{0,0,0,0,1,0,0,1,1,1,1}, {1,1,0,0,0,1,1,0,0,1,1}, {0,0,0,1,1,0,1,1,1,1,1}, {0,0,0,1,0,1,1,0,0,0,0}, {0,0,0,1,1,1,0,1,1,0,1}},
{{0,0,0,0,0,1,0,0,0,1,1}, {1,1,0,0,1,0,1,1,0,0,1,1}, {0,0,0,1,1,1,1,0,0,0,0}, {0,0,0,0,1,1,1,1,1,1,0}, {0,0,0,0,1,0,1,1,0,0,0,0}}
};
#define LB 10
#define LC 8
#define LE 11
#define LG 12
#define LP 7
#define RA 0
#define RB 1
#define RC 3
#define RD 5
#define RE 6
#define RF 4
#define RG 9
#define RP 2
#define RP 2
#define B0 A3
#define B1 A2
#define B2 A1
```



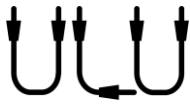
```
#define B3 A0
#define SW1D A4
#define SW2U A5
#define SW2D A6
#define SW3U A7

void setup() {
  // declare the ledPins as an OUTPUT:
  pinMode(LB, OUTPUT); // left display, segment b
  pinMode(LC, OUTPUT); // left display, segment c
  pinMode(LE, OUTPUT); // left display, segment e
  pinMode(LG, OUTPUT); // left display, segment g
  pinMode(LP, OUTPUT); // left display, segment dp
  pinMode(RA, OUTPUT); // right display, segment a
  pinMode(RB, OUTPUT); // right display, segment b
  pinMode(RC, OUTPUT); // right display, segment c
  pinMode(RD, OUTPUT); // right display, segment d
  pinMode(RE, OUTPUT); // right display, segment e
  pinMode(RF, OUTPUT); // right display, segment f
  pinMode(RG, OUTPUT); // right display, segment g
  pinMode(RP, OUTPUT); // right display, segment dp
  pinMode(13, OUTPUT); // blinking led showing the program heartbeat
  pinMode(B3, INPUT); // Bit 3
  pinMode(B2, INPUT); // Bit 2
  pinMode(B1, INPUT); // Bit 1
  pinMode(B0, INPUT); // Bit 0
  pinMode(SW1D, INPUT_PULLUP);
  pinMode(SW2U, INPUT_PULLUP);
  pinMode(SW2D, INPUT_PULLUP);
  pinMode(SW3U, INPUT);
};

int readvalue() {
  int v0, v1, v2, v3;
  v0 = analogRead(B0) > 500 ? 1 : 0;
  v1 = analogRead(B1) > 500 ? 1 : 0;
  v2 = analogRead(B2) > 500 ? 1 : 0;
  v3 = analogRead(B3) > 500 ? 1 : 0;
  return v0 + (v1 * 2) + (v2 * 4) + (v3 * 8);
}

int readsetting() {
  int r, s1d, s2u, s2d, s3u;
  s1d = analogRead(SW1D) < 500 ? 1 : 0; // Switch 1 is on, switch uses pull down resistor so equals 0 when on
  s2u = analogRead(SW2U) < 500 ? 1 : 0; // Switch 2 is up, switch uses pull down resistor so equals 0 when on
  s2d = analogRead(SW2D) < 500 ? 1 : 0; // switch 2 is down, switch uses pull down resistor so equals 0 when on
  s3u = analogRead(SW3U) < 500 ? 0 : 1; // display is on, switch puts +5V to connector
  r = 2;
  if (s2u == 1) {
    r = 1;
  } else {
    if (s2d == 1) r = 3;
  }
  if (s1d == 1) r = r + 3;
  if (s3u == 0) r = 0; // if display needs to be off, result = 0
  if ((s2u==1)&&(s2d==1)) r = 7; // if testmodus result = 7
  return r;
  // 0. Display off
  // 1. Binary
  // 2. Hexdecimal
  // 3. Decimal
  // 4. Signed
  // 5. 1-complement
  // 6. 2-complement
  // 7. System test
}

void loop(){
  digitalWrite(13, (millis() / 1000) % 2); //1s heartbeat for the onboard led
  setting = readsetting();
  if (setting == 7){
    delay(500);
    value = value + 1; setting = 2;
    if (value == 17) value = 0; // reset the counter
  } else {
    value = readvalue();
  }
  todo = 4;
  if (value < 8) todo = 2;
  if (value == 16) todo = 3; // all segments on
  if (setting == 0) todo = 0; // dimming the display
  if (setting == 1) todo = 1; // show binary
  // todo = 0 -> display off
  // todo = 1 -> binary display
  // todo = 2 -> value less than 8, so only write data to right display
  // todo = 3 -> all segments on
  // todo = 3 -> value > 7 display two segments
  digitalWrite(LP, (setting == 6) ? HIGH : LOW); // write the left dot if 2-complement
  digitalWrite(RP, setting > 4 ? HIGH : LOW); // write the right dot if 1-complement or 2-complement
  switch (todo) {
    case 0: // dim display
      digitalWrite(LB, LOW);
      digitalWrite(LC, LOW);
      digitalWrite(LE, LOW);
      digitalWrite(LG, LOW);
      digitalWrite(LP, LOW);
      digitalWrite(RA, LOW);
      digitalWrite(RB, LOW);
      digitalWrite(RC, LOW);
      digitalWrite(RD, LOW);
  }
```



```
digitalWrite(RE, LOW);
digitalWrite(RF, LOW);
digitalWrite(RG, LOW);
digitalWrite(RP, LOW);
break;
case 1: // binary
digitalWrite(LB, LOW);
digitalWrite(LG, LOW);
digitalWrite(LE, (show1[value][0] == 1) ? HIGH : LOW);
digitalWrite(LC, (show1[value][1] == 1) ? HIGH : LOW);
digitalWrite(RA, LOW);
digitalWrite(RB, LOW);
digitalWrite(RD, LOW);
digitalWrite(RF, LOW);
digitalWrite(RG, LOW);
digitalWrite(RE, (show1[value][2] == 1) ? HIGH : LOW);
digitalWrite(RC, (show1[value][3] == 1) ? HIGH : LOW);
break;
case 2: // value < 8 // only write data to right display
digitalWrite(LB, LOW);
digitalWrite(LG, LOW);
digitalWrite(LE, LOW);
digitalWrite(LC, LOW);
digitalWrite(RA, (show2[value][0] == 1) ? HIGH : LOW);
digitalWrite(RB, (show2[value][1] == 1) ? HIGH : LOW);
digitalWrite(RC, (show2[value][2] == 1) ? HIGH : LOW);
digitalWrite(RD, (show2[value][3] == 1) ? HIGH : LOW);
digitalWrite(RE, (show2[value][4] == 1) ? HIGH : LOW);
digitalWrite(RF, (show2[value][5] == 1) ? HIGH : LOW);
digitalWrite(RG, (show2[value][6] == 1) ? HIGH : LOW);
break;
case 3: // all segments on
digitalWrite(LB, HIGH);
digitalWrite(LC, HIGH);
digitalWrite(LE, HIGH);
digitalWrite(LG, HIGH);
digitalWrite(LP, HIGH);
digitalWrite(RA, HIGH);
digitalWrite(RB, HIGH);
digitalWrite(RC, HIGH);
digitalWrite(RD, HIGH);
digitalWrite(RE, HIGH);
digitalWrite(RF, HIGH);
digitalWrite(RG, HIGH);
digitalWrite(RP, HIGH);
break;
case 4: // write data on both displays
look = value - 8; pointer = setting - 2;
digitalWrite(LB, (show3[look][pointer][0] == 1) ? HIGH : LOW);
digitalWrite(LC, (show3[look][pointer][1] == 1) ? HIGH : LOW);
digitalWrite(LE, (show3[look][pointer][2] == 1) ? HIGH : LOW);
digitalWrite(LG, (show3[look][pointer][3] == 1) ? HIGH : LOW);
digitalWrite(RA, (show3[look][pointer][4] == 1) ? HIGH : LOW);
digitalWrite(RB, (show3[look][pointer][5] == 1) ? HIGH : LOW);
digitalWrite(RC, (show3[look][pointer][6] == 1) ? HIGH : LOW);
digitalWrite(RD, (show3[look][pointer][7] == 1) ? HIGH : LOW);
digitalWrite(RE, (show3[look][pointer][8] == 1) ? HIGH : LOW);
digitalWrite(RF, (show3[look][pointer][9] == 1) ? HIGH : LOW);
digitalWrite(RG, (show3[look][pointer][10] == 1) ? HIGH : LOW);
}
}
```