

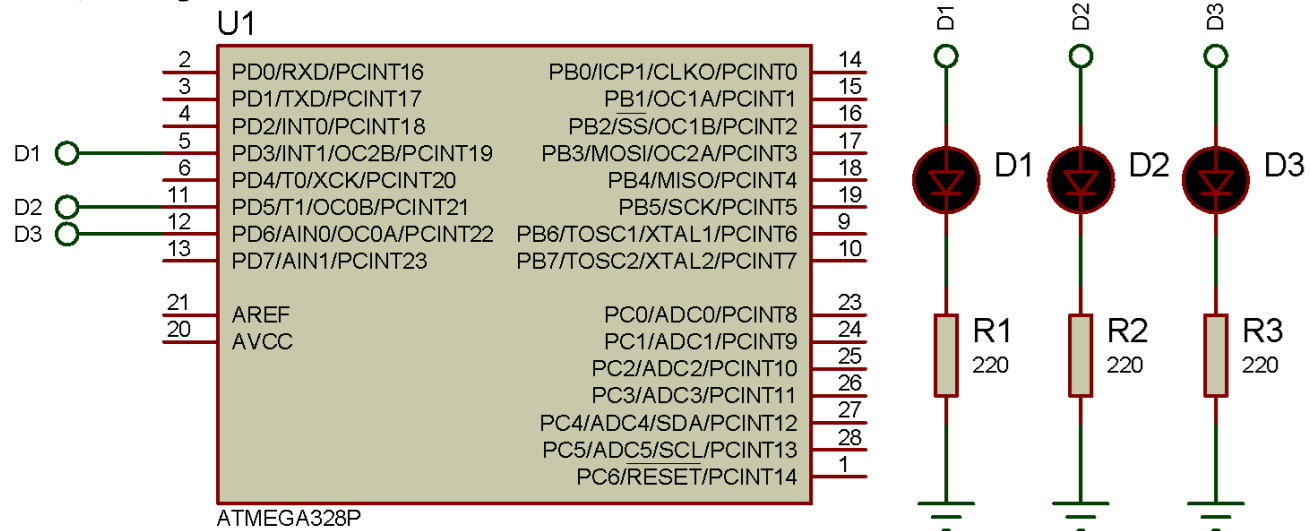
Interfacing with Simple Display Devices

Lecture 4 Embedded Systems

Adjusting the Brightness of an LED

```
#define DT 100
const int firstLed = 3;
const int secondLed = 5;
const int thirdLed = 6;
int brightness = 0;
int increment = 1;
void setup()
{
    // pins driven by analogWrite do not
    // need to be declared as outputs
}
void loop()
{
    if(brightness==255) increment*=-1;
    brightness+=increment;

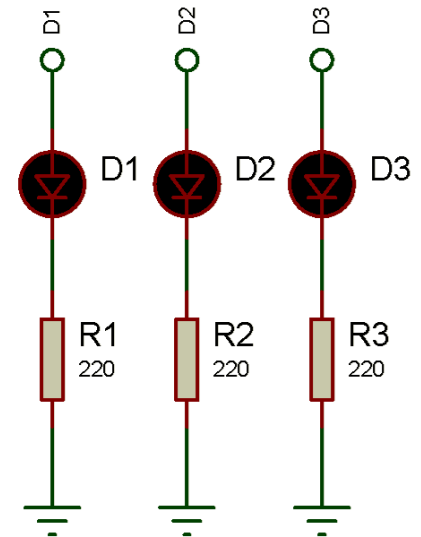
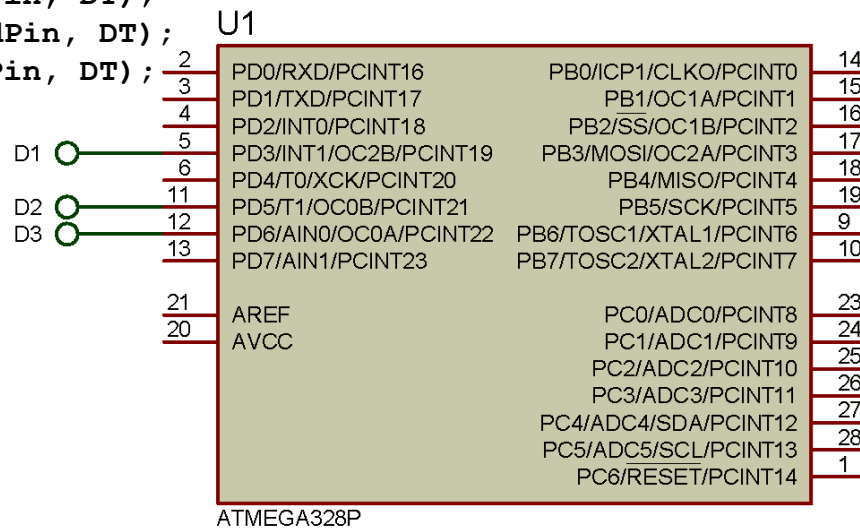
    analogWrite(firstLed, brightness);
    analogWrite(secondLed, brightness);
    analogWrite(thirdLed, brightness );
    delay(DT) ;
}
```



```

#define DT 10
const int firstLedPin = 3;
const int secondLedPin = 5;
const int thirdLedPin = 6;
void blinkLED(int pin, int duration)
{
    digitalWrite(pin, HIGH);
    delay(duration);
    digitalWrite(pin, LOW);
    delay(duration);
}
void setup()
{
    pinMode(firstLedPin, OUTPUT);
    pinMode(secondLedPin, OUTPUT);
    pinMode(thirdLedPin, OUTPUT);
}
void loop()
{
    blinkLED(firstLedPin, DT);
    blinkLED(secondLedPin, DT);
    blinkLED(thirdLedPin, DT);
}

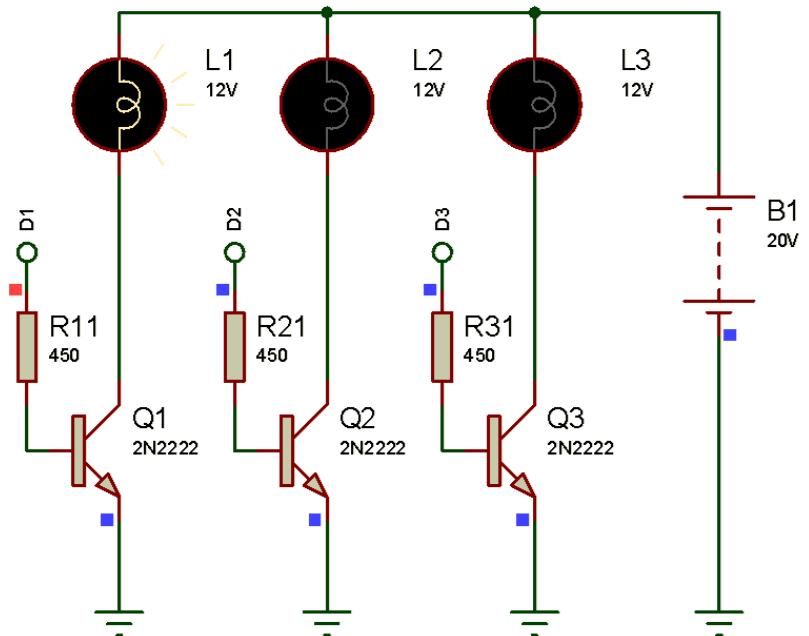
```



Connecting and Using LEDs

```
#define DT 100
const int firstLed = 3;
const int secondLed = 5;
const int thirdLed = 6;
Unsigned char brightness = 0;
int increment = 1;
void setup()
{
    // pins driven by analogWrite do not
    // need to be declared as outputs
}
void loop()
{
    if(brightness==255)increment*=-1;
    brightness+=increment;

    analogWrite(firstLed, brightness);
    analogWrite(secondLed, brightness);
    analogWrite(thirdLed, brightness );
    delay(DT) ;
}
```

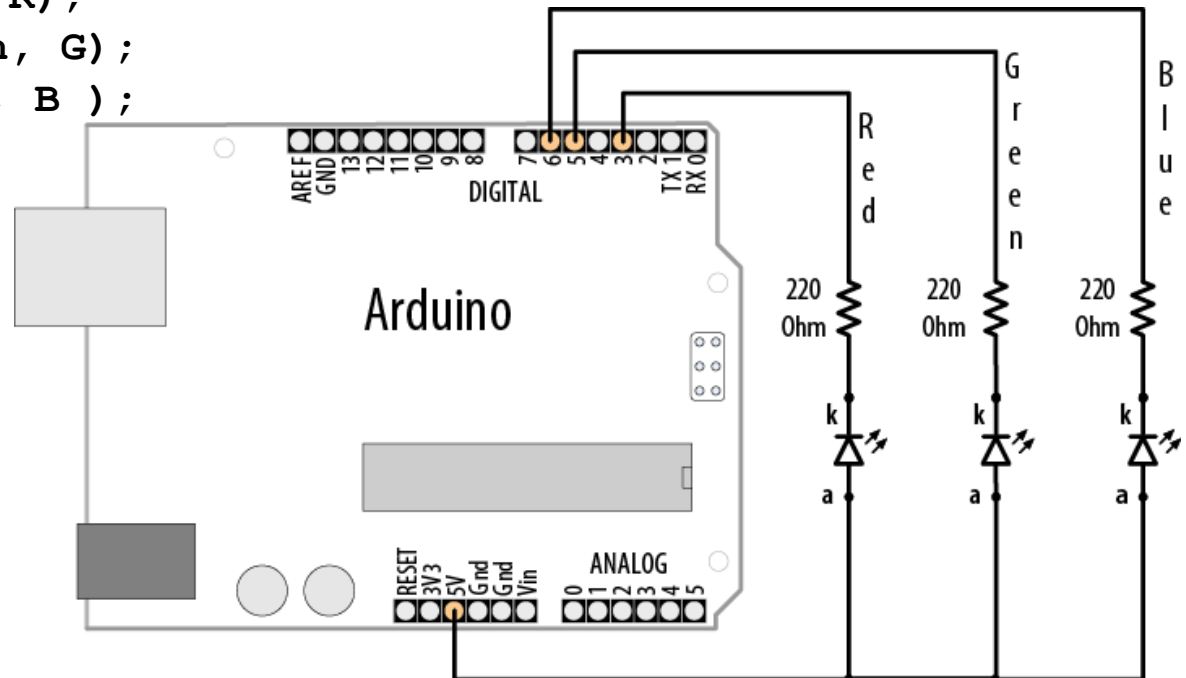


Adjusting the Color of an LED

```
#define DT 10
const int redPin = 3;
const int greenPin = 5;
const int bluePin = 6;
int R = 255, G = 165, B = 0; // Orange
void setup()
{
    // pins driven by analogWrite
    // do not need to be declared as outputs
}
void loop()
{
    analogWrite(redPin, R);
    analogWrite(greenPin, G);
    analogWrite(bluePin, B );
    delay(DT);
}
```



1 Red
2 Ground
3 Green
4 Blue

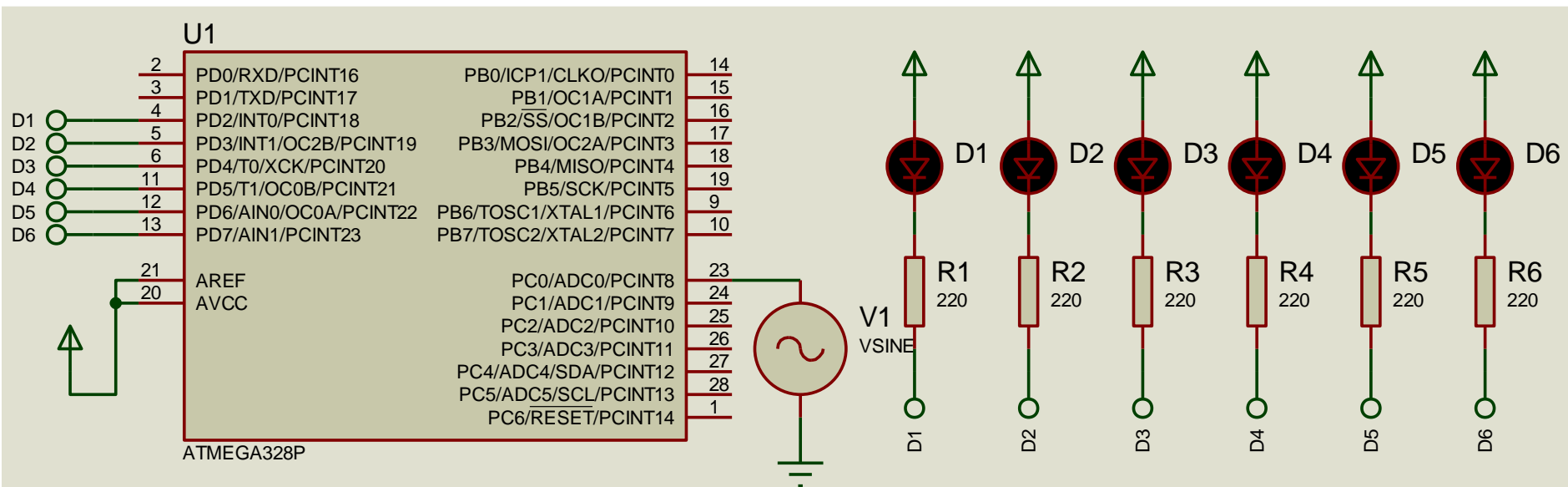


```

const int leds[] = { 2, 3, 4, 5, 6, 7};
const int nLeds = sizeof(leds)/sizeof(int);
const int analogInPin = 0;
void setup() {
    for (int i = 0; i < nLeds; i++)
        pinMode(leds[i], OUTPUT);
}
void loop() {
    int sensorValue = analogRead(analogInPin);
    sensorValue = map(sensorValue, 0, 1023, 0, nLeds);
    for (int i = 0; i < nLeds; i++)
    {
        if(i < sensorValue)
            digitalWrite(leds[i], HIGH);
        else
            digitalWrite(leds[i], LOW);
    }
}

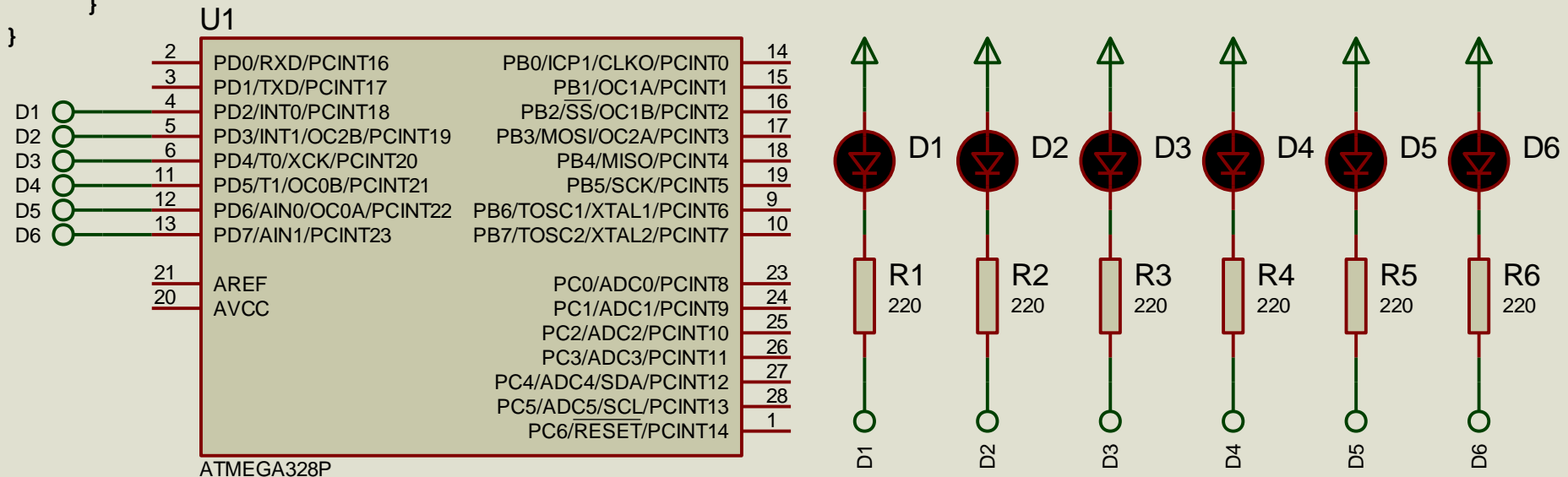
```

Sequencing Multiple LEDs 1

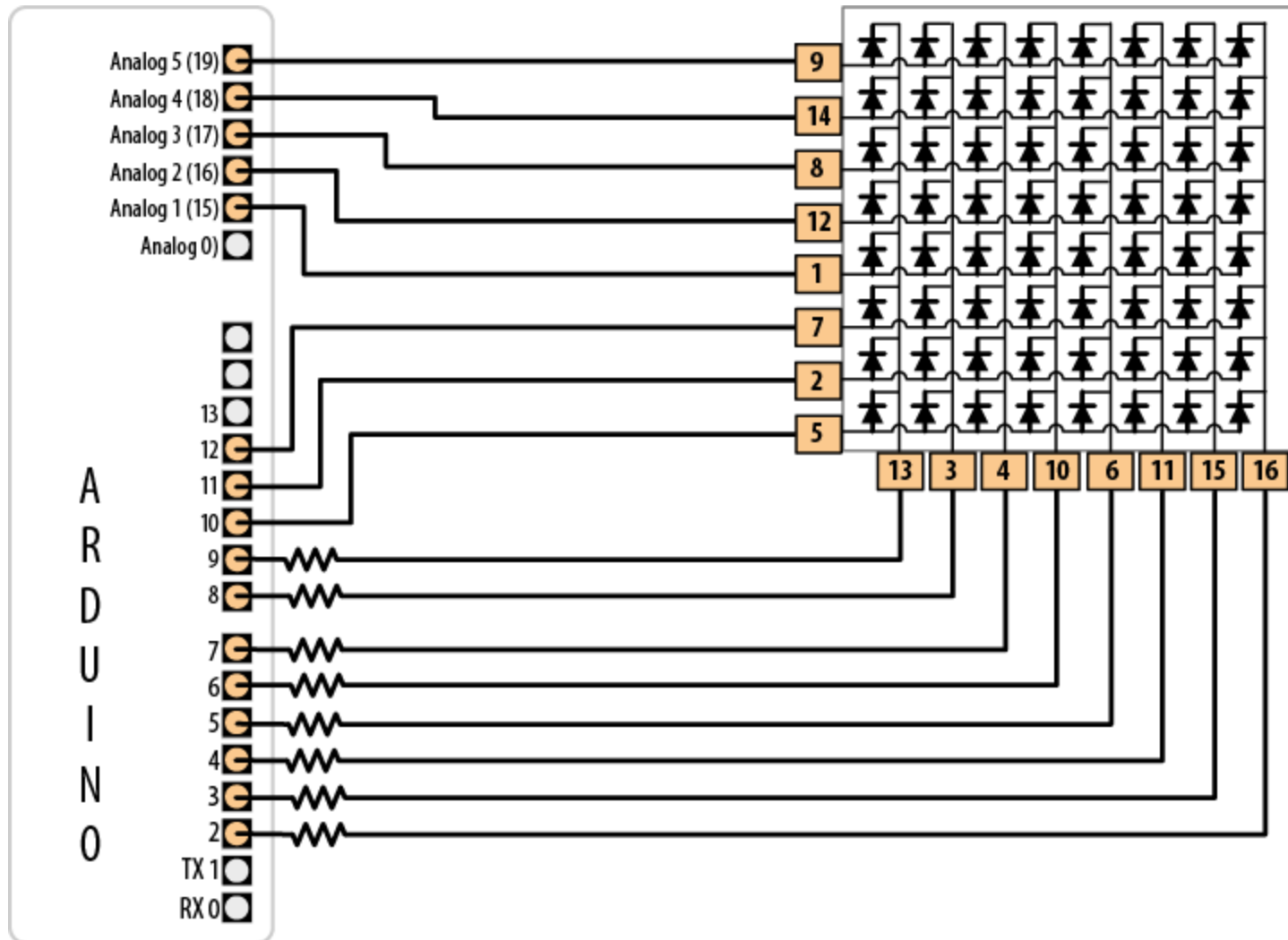


Sequencing Multiple LEDs 2

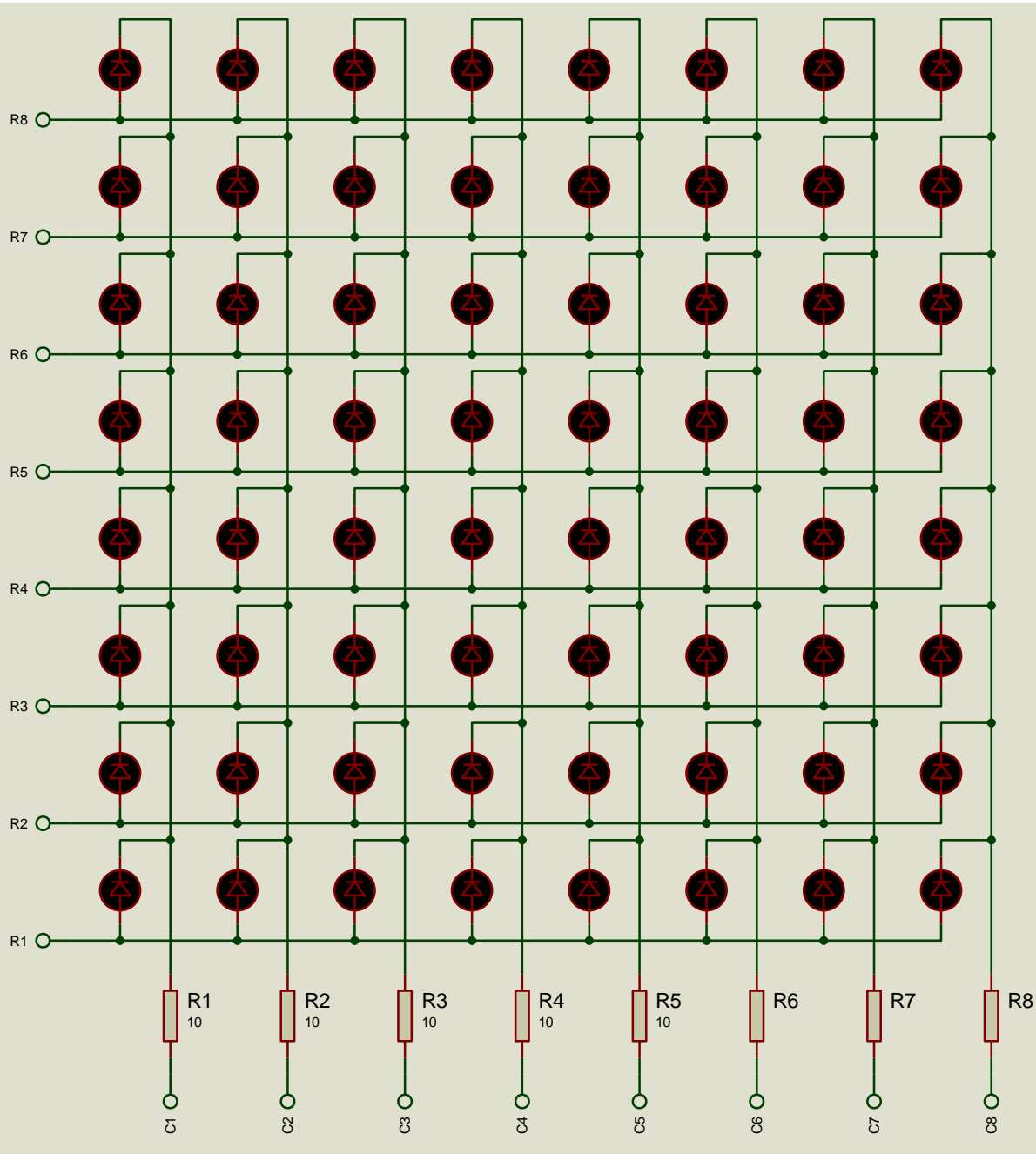
```
#define DT 5
const int leds[] = { 2, 3, 4, 5, 6, 7};
const int nLeds = sizeof(leds)/sizeof(int);
const int analogInPin = 0;
void setup() {
    for (int i = 0; i < nLeds; i++)
        pinMode(leds[i], OUTPUT);
}
void loop() {
    for (int i = 0; i < nLeds-1; i++)
    {
        digitalWrite(leds[i], HIGH); delay(DT);
        digitalWrite(leds[i+1], HIGH); delay(DT);
        digitalWrite(leds[i], LOW); delay(DT*2);
    }
    for (int i = nLeds; i > 0; i--) {
        digitalWrite(leds[i], HIGH); delay(DT);
        digitalWrite(leds[i-1], HIGH); delay(DT);
        digitalWrite(leds[i], LOW); delay(DT*2);
    }
}
```



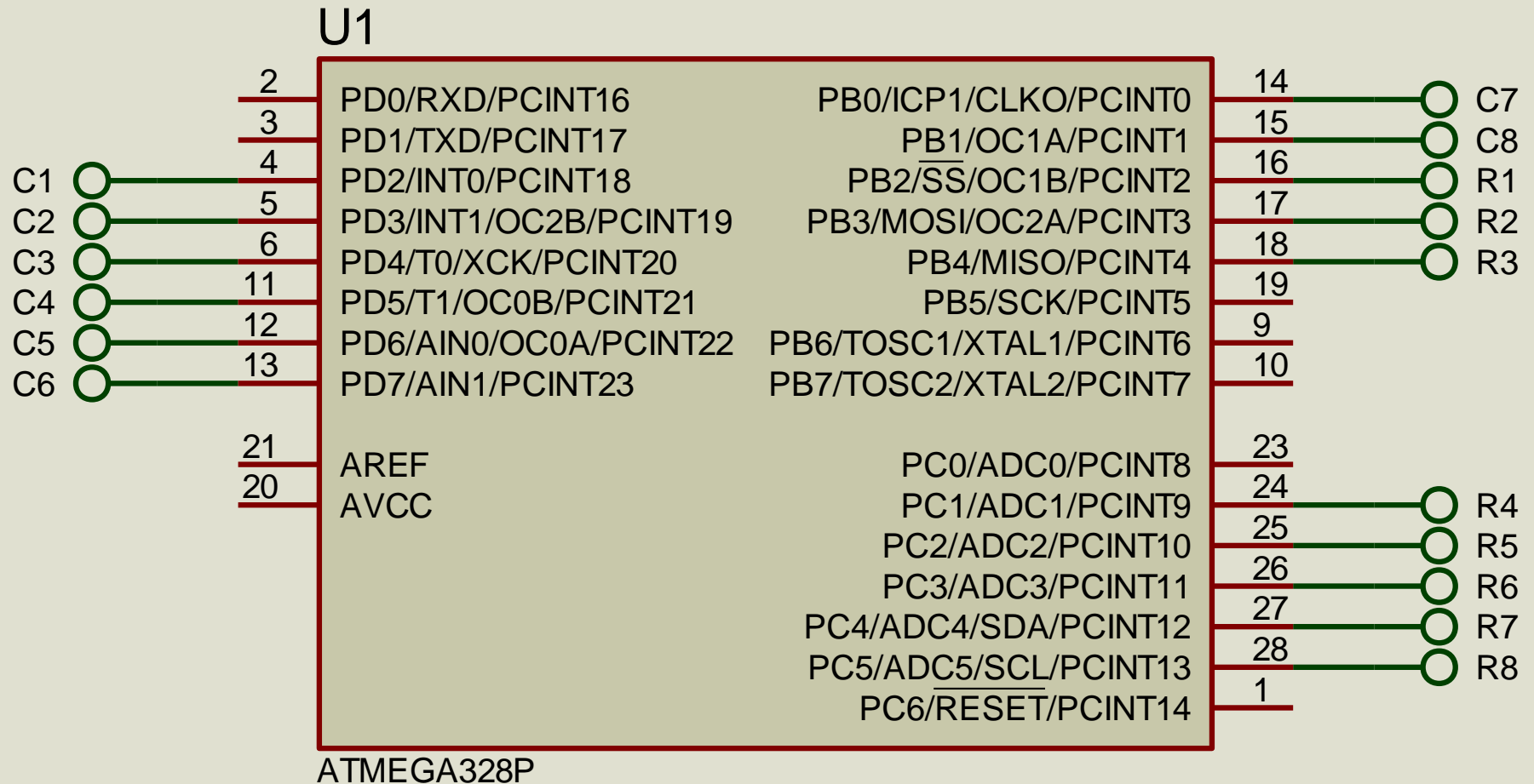
Controlling a LED Matrix Using Multiplexing



Controlling a LED Matrix Using Multiplexing



Controlling an LED Matrix Using Multiplexing



Controlling an LED Matrix Using Multiplexing

```
#define DT 10
const int columnPins[] = { 2, 3, 4, 5, 6, 7, 8, 9};
const int rowPins[] = { 10,11,12,15,16,17,18,19};
int iRow = 0;
int iColumn = 0;

void setup() {
    for (int i = 0; i < 8; i++){
        pinMode(columnPins[i], OUTPUT);
        pinMode(rowPins[i], OUTPUT);
    }
}

void loop() {
    for(int c = 0; c < 8 ; c++)
        digitalWrite(columnPins[c], (c==iColumn)?LOW:HIGH);

    for(int r = 0; r < 8 ; r++)
        digitalWrite(rowPins[r], (r==iRow)?HIGH:LOW);

    delay(DT);
    iColumn++;
    if(iColumn==8){iColumn = 0; iRow++;}
    if(iRow==8){iColumn = 0; iRow = 0;}
}
```

```

#define DT 50
byte bigHeart[] = { B01100110, B11111111, B11111111, B11111111,
                    B01111110, B00111100, B00011000, B00000000};
byte smallHeart[]={ B00000000, B00000000, B00010100, B00111110,
                    B00111110, B00011100, B00001000, B00000000};

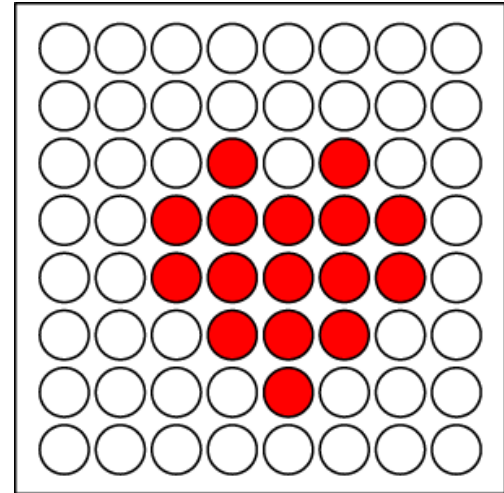
const int columnPins[] = { 2, 3, 4, 5, 6, 7, 8, 9};
const int rowPins[] = { 10,11,12,15,16,17,18,19};
void show( byte * image, unsigned long duration){
    unsigned long start = millis();
    while (start + duration > millis()){
        for(int row = 0; row < 8; row++){
            digitalWrite(rowPins[row], HIGH);
            for(int column = 0; column < 8; column++){
                boolean pixel = bitRead(image[row],column);
                if(pixel)digitalWrite(columnPins[column], LOW);
                delayMicroseconds(300);
                digitalWrite(columnPins[column], HIGH);
            }
            digitalWrite(rowPins[row], LOW);
        }
    }
}

void setup() {
    for (int i = 0; i < 8; i++){
        pinMode(rowPins[i], OUTPUT)
        pinMode(columnPins[i], OUTPUT);
        digitalWrite(columnPins[i], HIGH);
    }
}

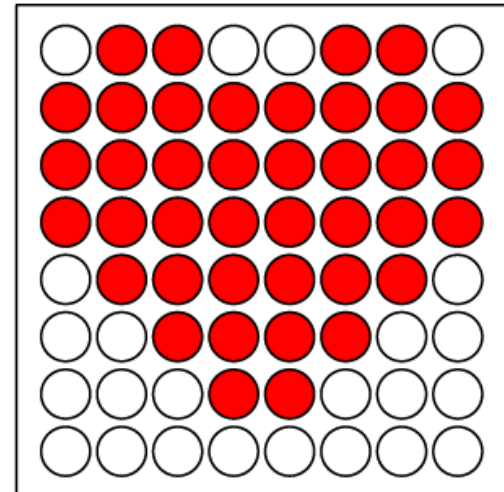
void loop() {
    show(smallHeart, DT);
    show(bigHeart, 2*DT);
    delay(5*DT);
}

```

Showing Images



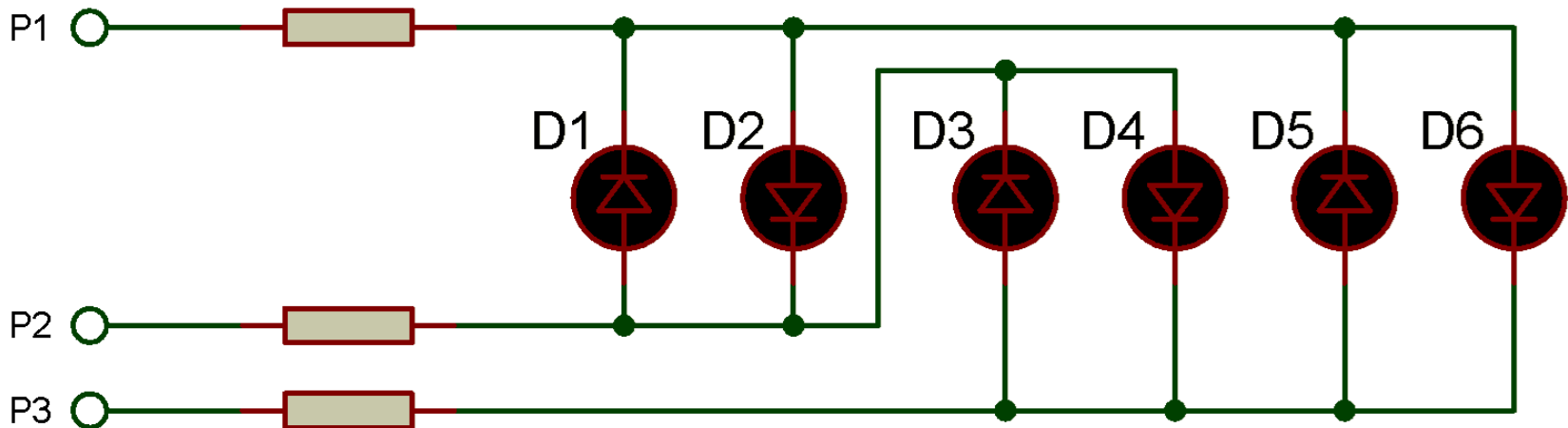
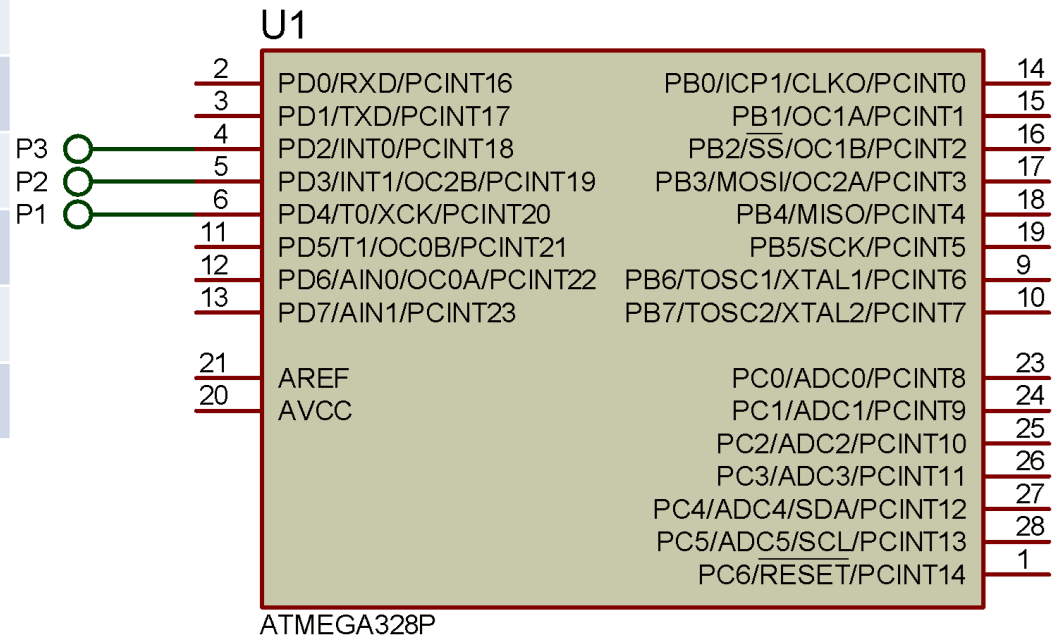
Small Heart



Big Heart

Using Charlieplexing

	P1	P2	P3
-	LOW	LOW	LOW
D1	LOW	HIGH	INPUT
D2	HIGH	LOW	INPUT
D3	INPUT	LOW	HIGH
D4	INPUT	HIGH	LOW
D5	LOW	INPUT	HIGH
D6	HIGH	INPUT	LOW



```

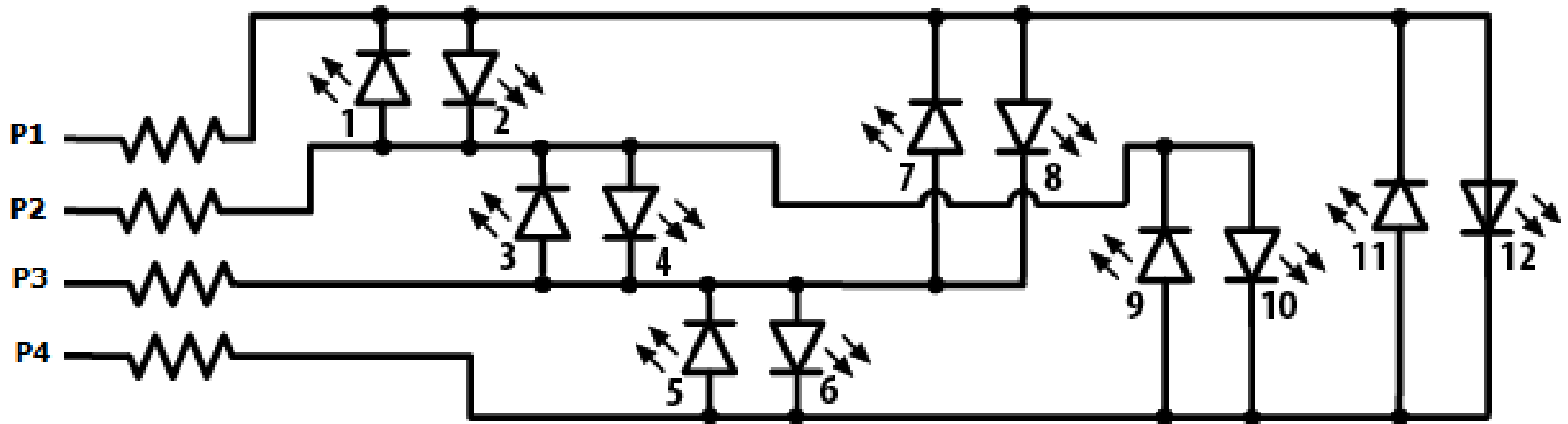
#define DT 100
byte pins[] = {2,3,4};
const int NUMBER_OF_PINS = sizeof(pins)/ sizeof(pins[0]);
const int NUMBER_OF_LEDS = NUMBER_OF_PINS * (NUMBER_OF_PINS-1);
byte pairs[NUMBER_OF_LEDS/2][2] = { {0,1}, {1,2}, {0,2} };
void setup(){}
void loop(){
    for(int i=0; i < NUMBER_OF_LEDS; i++){
        lightLed(i);
        delay(DT);
    }
}
void lightLed(int led){
    int indexA = pairs[led/2][0];
    int indexB = pairs[led/2][1];
    int pinA = pins[indexA];
    int pinB = pins[indexB];
    for(int i=0; i < NUMBER_OF_PINS; i++){
        if(i!=indexA && i!=indexB){
            pinMode(pins[i], INPUT);
            digitalWrite(pins[i],LOW);
        }
    }

    pinMode(pinA, OUTPUT);
    pinMode(pinB, OUTPUT);
    if( led % 2 == 0){
        digitalWrite(pinA,LOW);
        digitalWrite(pinB,HIGH);
    }
    else{
        digitalWrite(pinB,LOW);
        digitalWrite(pinA,HIGH);
    }
}

```

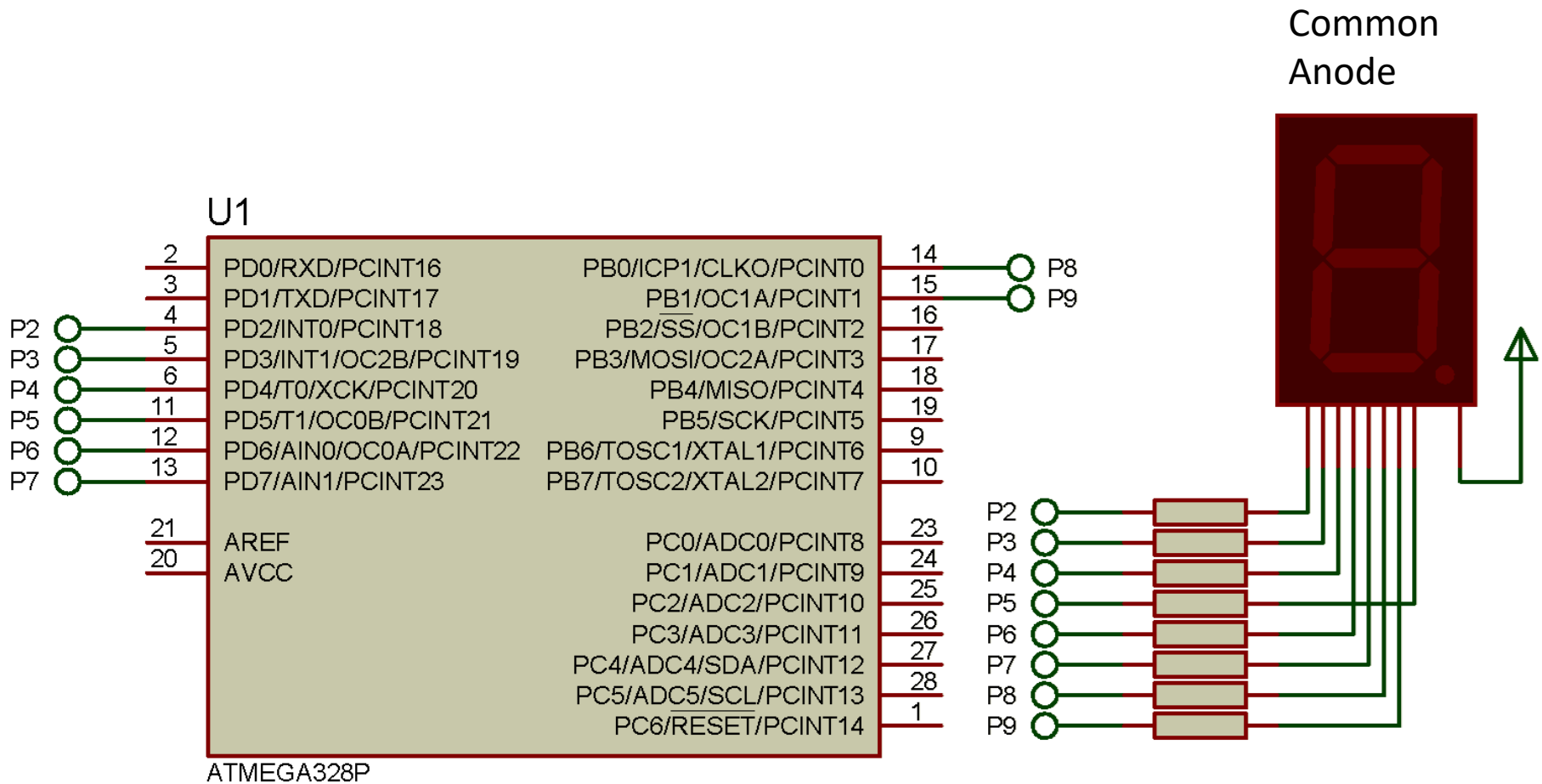
Using Charlieplexing

Using Charlieplexing



	P1	P2	P3	P4
-	LOW	LOW	LOW	LOW
D1	LOW	HIGH	INPUT	INPUT
D2	HIGH	LOW	INPUT	INPUT
D3	INPUT	LOW	HIGH	INPUT
D4	?	?	?	?

Driving a 7-Segment LED Display

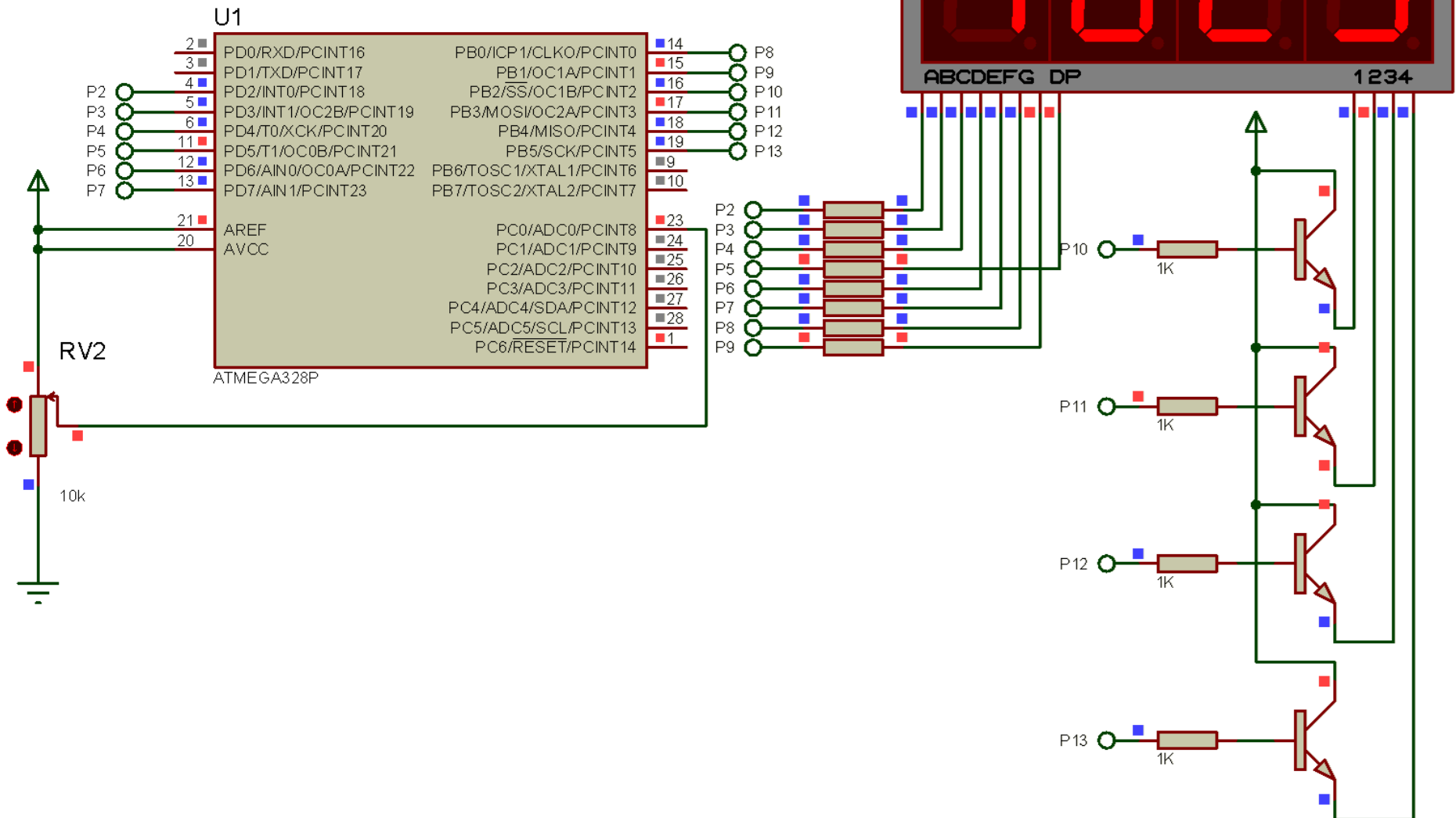


Driving a 7-Segment LED Display

```
#define DT 50
const byte digits[10] = {
    //ABCDEFGP
    B11111100, // 0
    B01100000, // 1
    B11011010, // 2
    B11110010, // 3
    B01100110, // 4
    B10110110, // 5
    B00111110, // 6
    B11100000, // 7
    B11111110, // 8
    B11100110, // 9
};

const int segmentPins[8] = { 5,9,8,7,6,4,3,2};
int number= 0;
void setup(){
    for(int i=0; i < 8; i++)
        pinMode(segmentPins[i], OUTPUT);
}
void loop(){
    showDigit(number++);
    delay(DT);
    if(number==10)number= 0;
}
void showDigit(int number){
    for(int segment = 0; segment < 8; segment++)
    {
        boolean isBitSet = bitRead(digits[number], segment);
        digitalWrite( segmentPins[segment], !isBitSet);
    }
}
```

Driving Multidigit 7-Segment LED Displays



```

#define DT 1
const byte digits[10]={  B11111100, B01100000, B11011010,
                        B11110010, B01100110, B10110110,
                        B00111110, B11100000, B11111110,
                        B11100110};

const int segmentPins[8] = { 5,9,8,7,6,4,3,2};
const int digitPins[4] = {10,11,12,13};
int nDigits = sizeof(digitPins)/sizeof(int);
void setup() {
    for(int i=0; i < 8; i++)
        pinMode(segmentPins[i], OUTPUT);
    for(int i=0; i < nDigits; i++)
        pinMode(digitPins[i], OUTPUT);
}
void loop() {
    int value = analogRead(0);
    showNumber(value);
}
void showNumber(int number) {
    for(int digit = nDigits-1; digit >= 0; digit--){
        showDigit(number % 10, digit) ;
        number = number / 10;
    }
}
void showDigit(int number, int digit){
    digitalWrite(digitPins[digit], HIGH);
    for(int segment = 0; segment < 8; segment++)
    {
        boolean isBitSet = bitRead(digits[number], segment);
        digitalWrite( segmentPins[segment], !isBitSet);
    }
    delay(DT);
    digitalWrite(digitPins[digit], LOW);
}

```

Driving Multidigit 7-Segment LED Displays