CHIPKIT UNO 32: PLAYING TONES ON MULTIPLE OUTPUTS USING

The tone() function

Author: Umashankar Shetty C

This example shows how to use the tone() command to play different notes on multiple outputs.

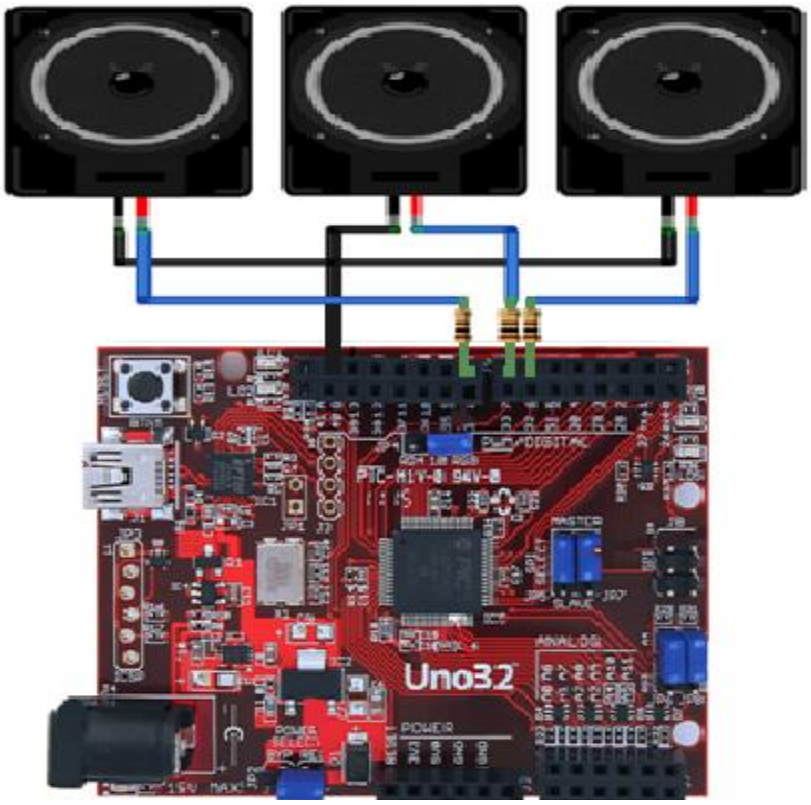
The tone() command works by taking over one of the Atmega's internal timers, setting it to the frequency you want, and using the timer to pulse an output pin. Since it's only using one timer, you can only play one note at a time.

You can, however, play notes on multiple pins sequentially. To do this, you need to turn the timer off for one pin before moving on to the next.

Hardware Required

- (3) 8-ohm speakers
- (3) 100 ohm resistor
- breadboard
- hook up wire

Hardware Connection:



Code

The sketch below plays a tone on each of the speakers in sequence, turning off the previous speaker first. Note that the duration of each tone is the same as the delay that follows it.
Here's the main sketch:

```
/*
  Multiple tone player

 Plays multiple tones on multiple pins in sequence

 circuit:
 * 3  8-ohm speaker on digital pins 6, 7, and 11
  */

void setup() {

}

void loop() {
   // turn off tone function for pin 11:
```

```
  noTone(11);
 // play a note on pin 6 for 200 ms:
 tone(6, 440, 200);
 delay(200);

 // turn off tone function for pin 6:
 noTone(6);
 // play a note on pin 7 for 500 ms:
 tone(7, 494, 500);
 delay(500);

 // turn off tone function for pin 7:
 noTone(7);
 // play a note on pin 11 for 500 ms:
 tone(11, 523, 300);
 delay(300);

}
```