Using Timers

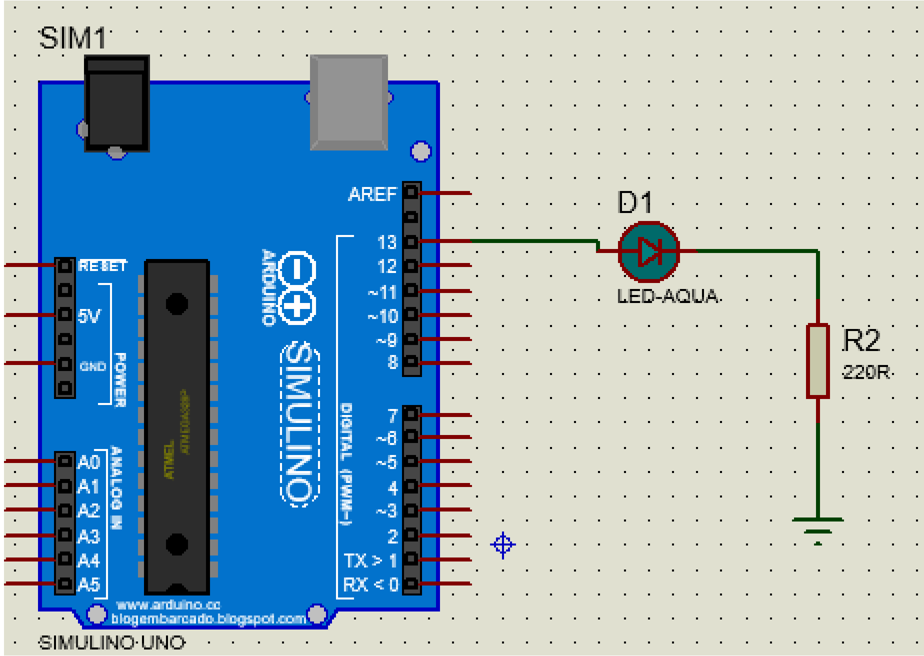
Making Manual Timer

```
#define LED 13

void flash() {
	static boolean output = HIGH;
	digitalWrite(LED, output);
	output = !output;
}
int oldTime = 0;
void setup() {
	pinMode(LED, OUTPUT);
}
void loop() {
	int time = millis();
	if((time-oldTime)>250)
	{
		flash();
		oldTime = time;
	}
}
```

Notes:

1. This is a manual non real-time timer.
2. Timer interval may exceed 250 ms.

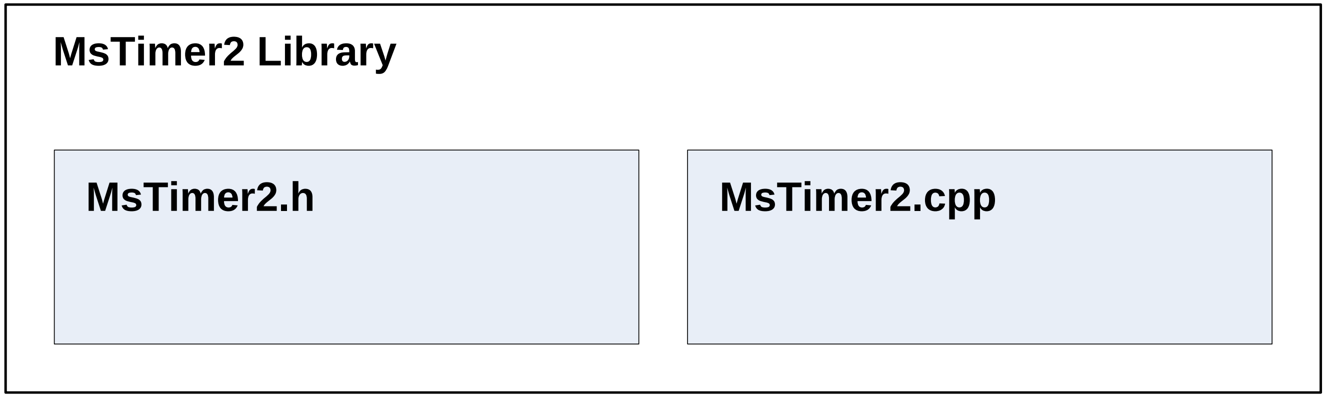


Other Time Functions

- mills(): measure the time in ms since the board is started
- micros(): measure the time in us since the board is started
- delay(): stops the program for the specified period in ms
- delayMicroseconds(): stops the program for the specified period in us

Installing MsTimer2 Library

- Install MsTimer2 Library
  1. Download from http://www.arduino.cc/playground/Main/MsTimer2
  2. Unzip
  3. Place the folder Inside {Arduino Path}/ libraries
  4. Restart Arduino Software
- Note: To make your own library you have to study
  - AVR Architecture
  - AVR programming g using C/C++ or Assembly



Using MsTimer2 Library

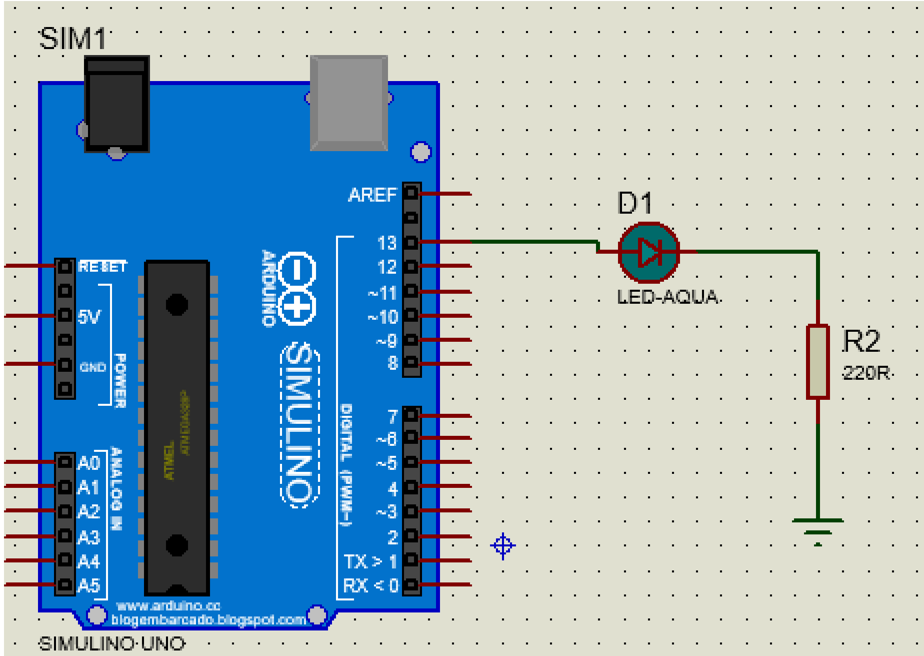
```
#include <MsTimer2.h>

#define LED 13

void flash() {
      static boolean output = HIGH;
      digitalWrite(LED, output);
      output = !output;
}

void setup() {
      pinMode(LED, OUTPUT);
      MsTimer2::set(500, flash);
      MsTimer2::start();
}

void loop() {
}
```



Notes:

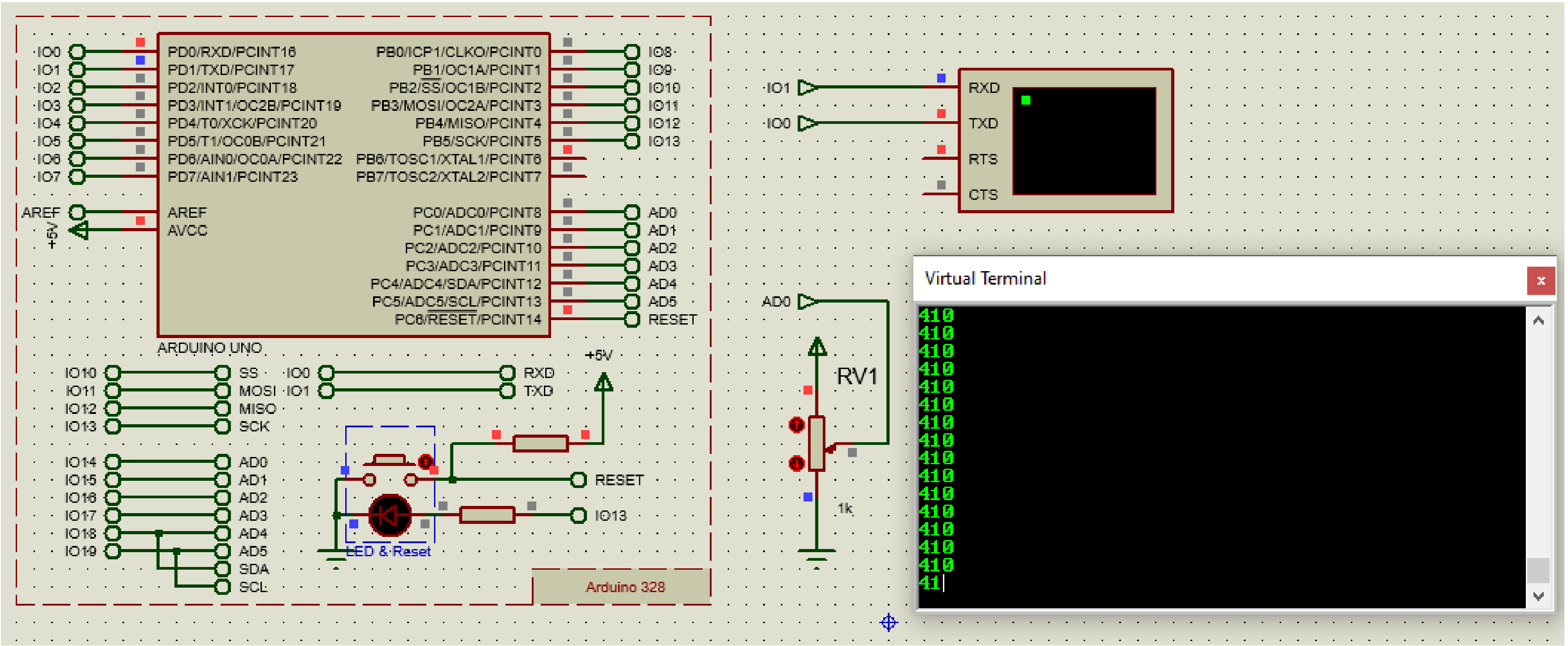
1. This is a real-time timer.
2. Timer interval exactly equals 250 ms.

Other MsTimer2 Library Functions

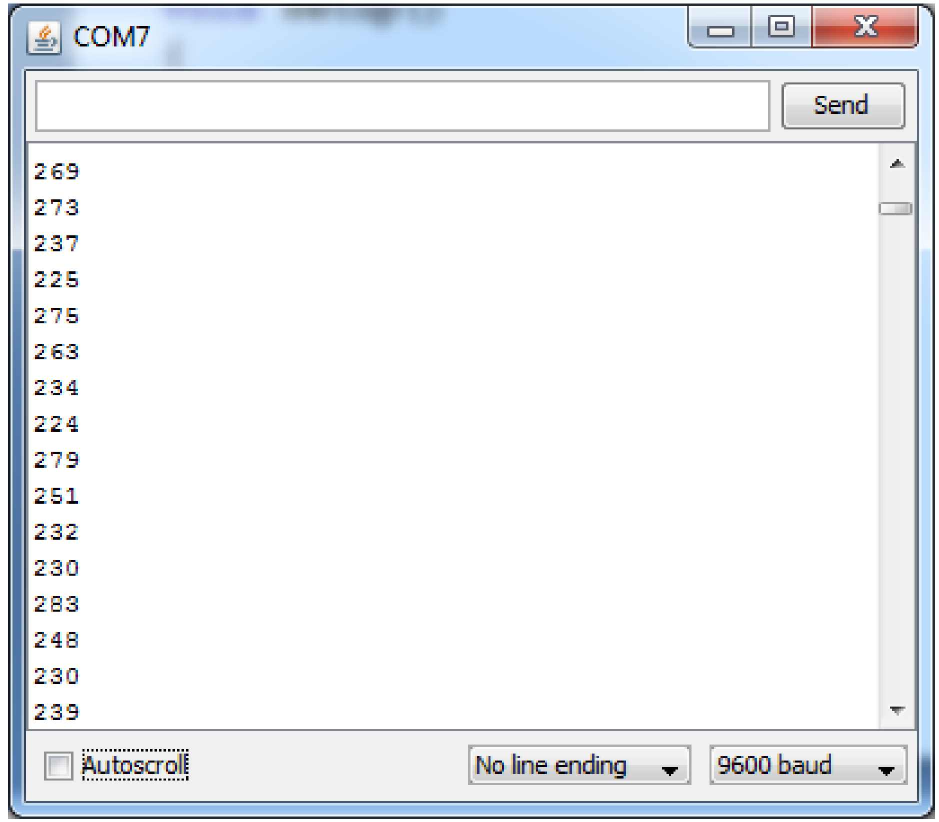
- set(interval, callbackfn): Set the real-time timer interval in ms, and sets the callback function name
- start(): Starts the timer
- stop(): Stops the timer

Reading Analog Signal

Reading Analog Input to Computer



Reading Analog Input to Computer




```
#define AINPUT 0

void setup()
{
  Serial.begin(9600);
}

void loop()
{
  int val = analogRead(AINPUT);
  Serial.println(val);
}
```

ATMega328 A/D Converter

- Read 6 analog inputs
- Analog range : 0→5V / 0→3.3V depending on the power signal (VCC)
- Resolution: 10 bit
- Digital range : 0→1023
- 1 bit change = 5V/1024 = 0.0049V
- Use **analogReference**(type) function to change the range bellow the maximum

Changing A/D Input Voltage Range

- Using **analogReference(type)** function
- Type can take:
  - DEFAULT: 5V or 3.3V based on Board Type
  - INTERNAL: 1.1V for UNO Boards, 2.56 for Mega Boards
  - INTERNAL1V1: 1.1V for Mega Boards
  - INTERNAL2V56: 2.56V for Mega Boards
  - EXTERNAL: External volt supplied to AREF Pin (Pin21 internal). Limited by 5V or 3.3V depending on board type

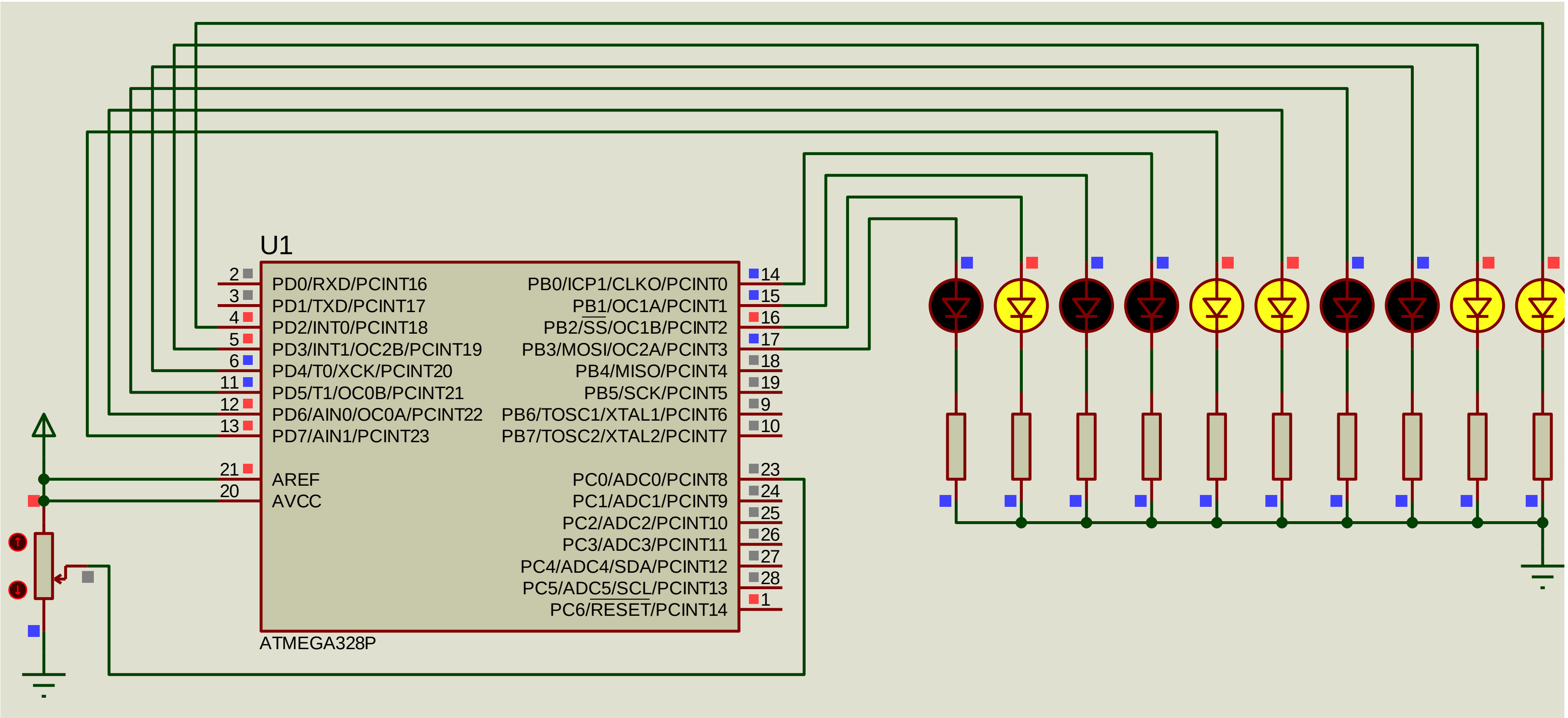
Read Analog Input and Display its Binary Value on 10 LEDs

```
#define AINPUT 0
#define OP 2

void setup()
{
	for(int i=0;i<10;i++)
		pinMode(OP+i, OUTPUT);
	analogReference(EXTERNAL);
}

int value = 0;
void loop()
{
	value = analogRead(AINPUT);
	for(int i=0;i<10;i++)
	{
		digitalWrite(OP+i, value&0x1);
		value>>=1;
	}
}
```

Read Analog Input and Display its Binary Value on 10 LEDs

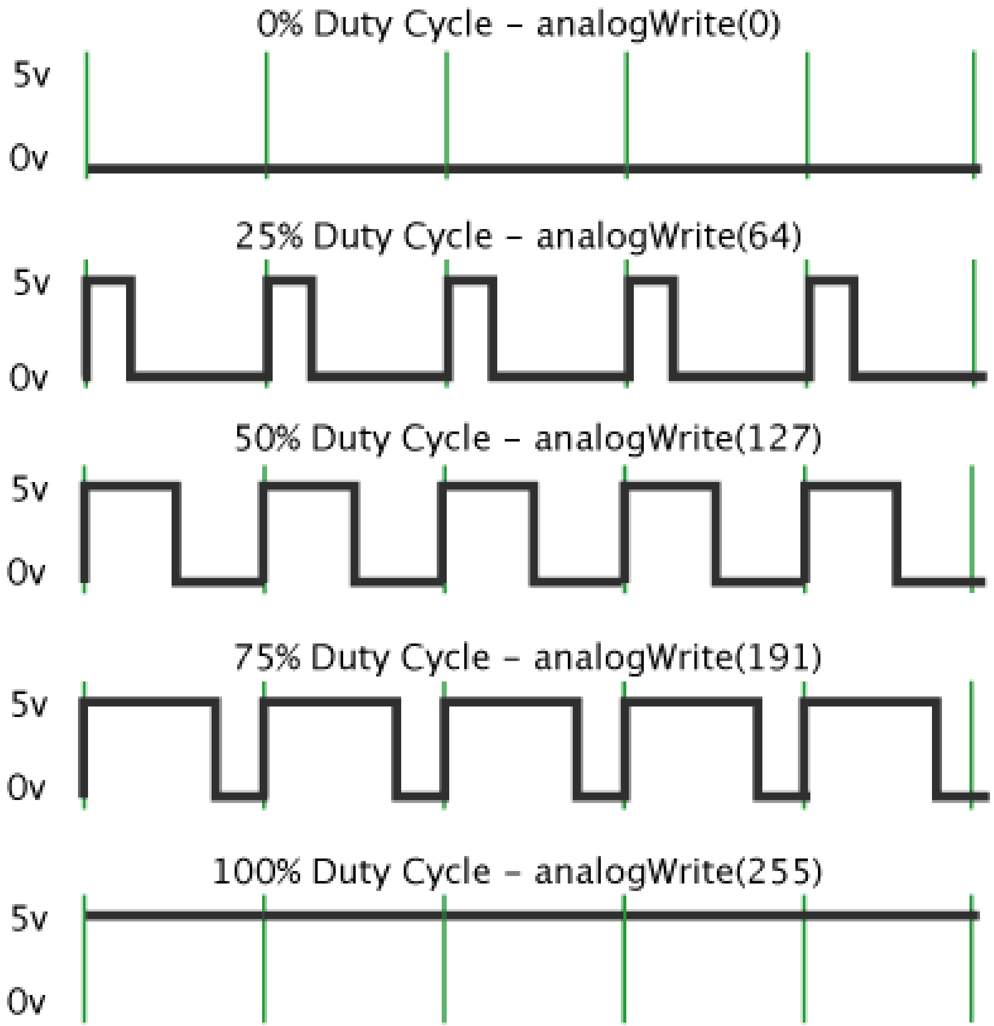


Producing Analog Signal

Producing Analog Signal

- PWM Analog-Like Signal
- Normal Analog Signal

PWM



Zero Energy

$$E = \int_0^t I(t) \times V(t)\, dt$$

Maximum Energy

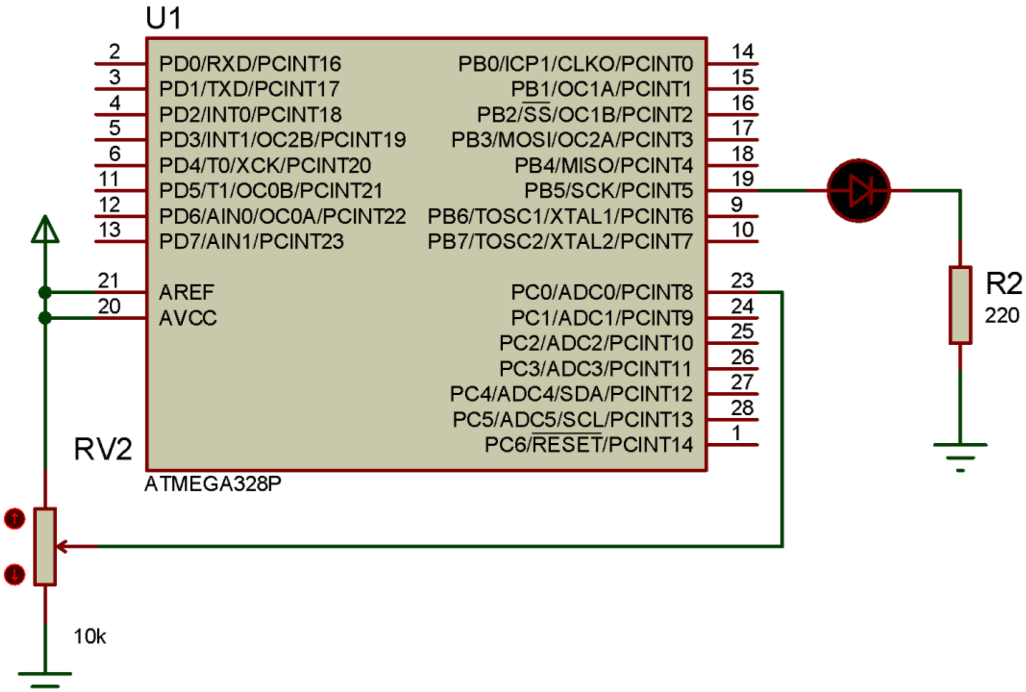
Generating Manual PWM

```
#define AINPUT 0
#define LED 13

void setup()
{
      pinMode(LED, OUTPUT);
      analogReference(EXTERNAL);
}

void loop()
{
      int value = analogRead(AINPUT);
      int onTime = map(value, 0, 1023, 0, 100);

      digitalWrite(LED, HIGH);
      delay(onTime);
      digitalWrite(LED, LOW);
      delay(100-onTime);
}
```

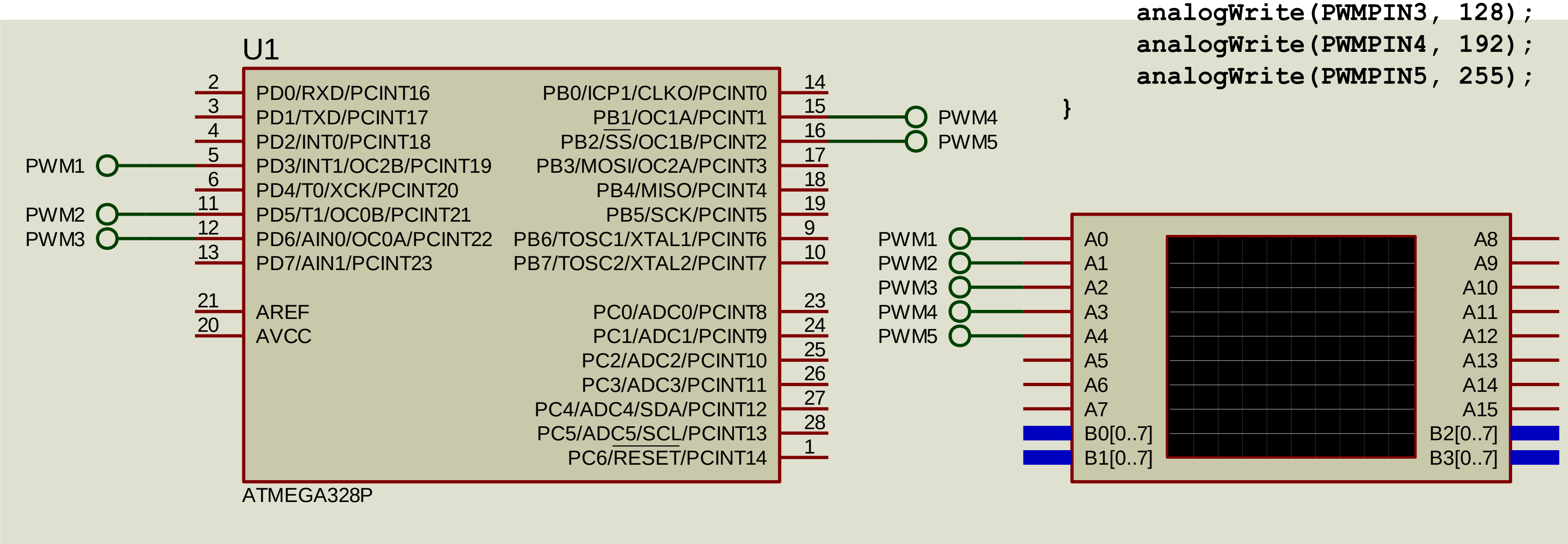


Using Built-in PWM

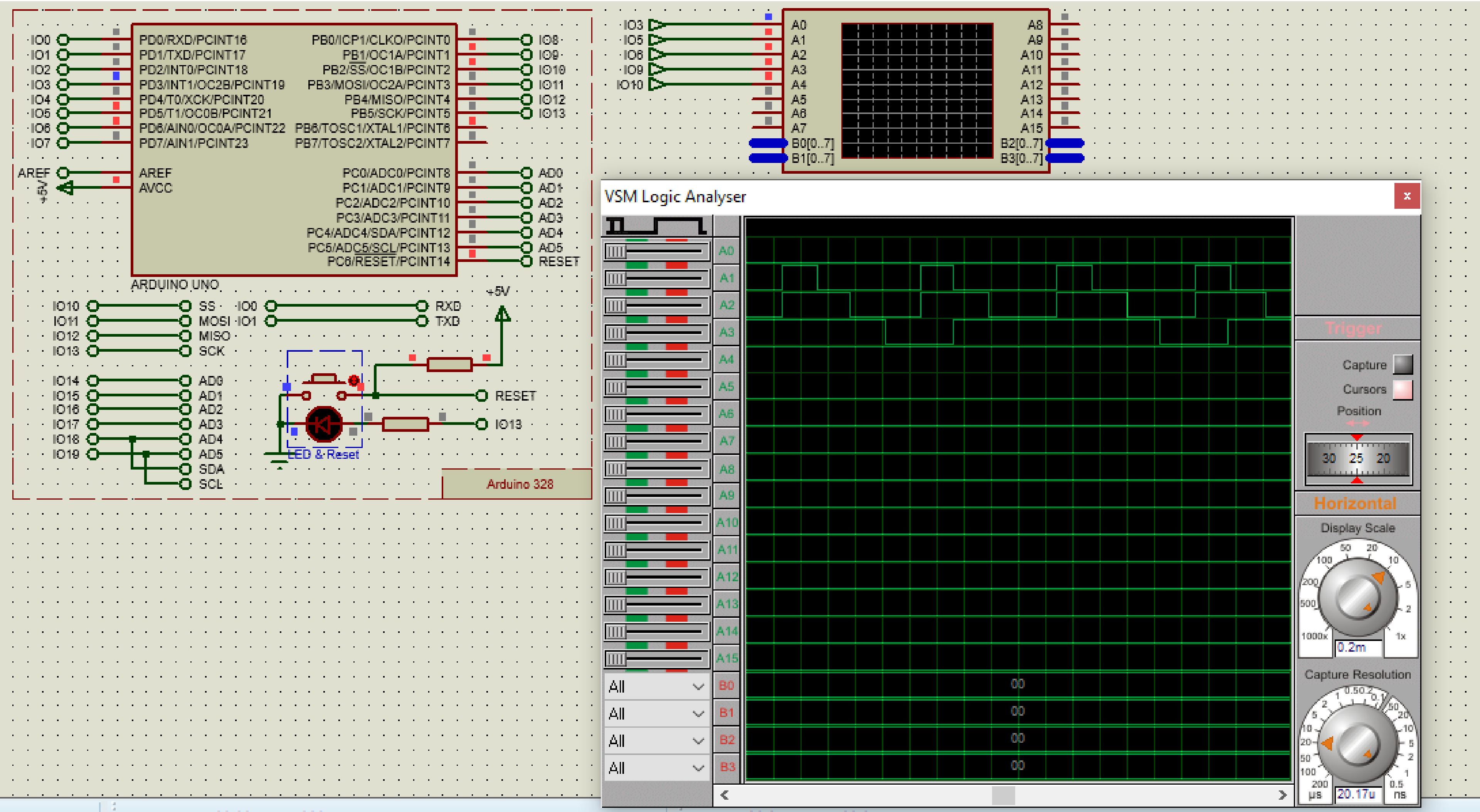
```
#define PWMPIN1 3
#define PWMPIN2 5
#define PWMPIN3 6
#define PWMPIN4 9
#define PWMPIN5 10

void setup()
{
	pinMode(PWMPIN1, OUTPUT);
	pinMode(PWMPIN2, OUTPUT);
	pinMode(PWMPIN3, OUTPUT);
	pinMode(PWMPIN4, OUTPUT);
	pinMode(PWMPIN5, OUTPUT);
}

void loop()
{
	analogWrite(PWMPIN1, 0);
	analogWrite(PWMPIN2, 64);
	analogWrite(PWMPIN3, 128);
	analogWrite(PWMPIN4, 192);
	analogWrite(PWMPIN5, 255);
}
```



Using Built-in PWM

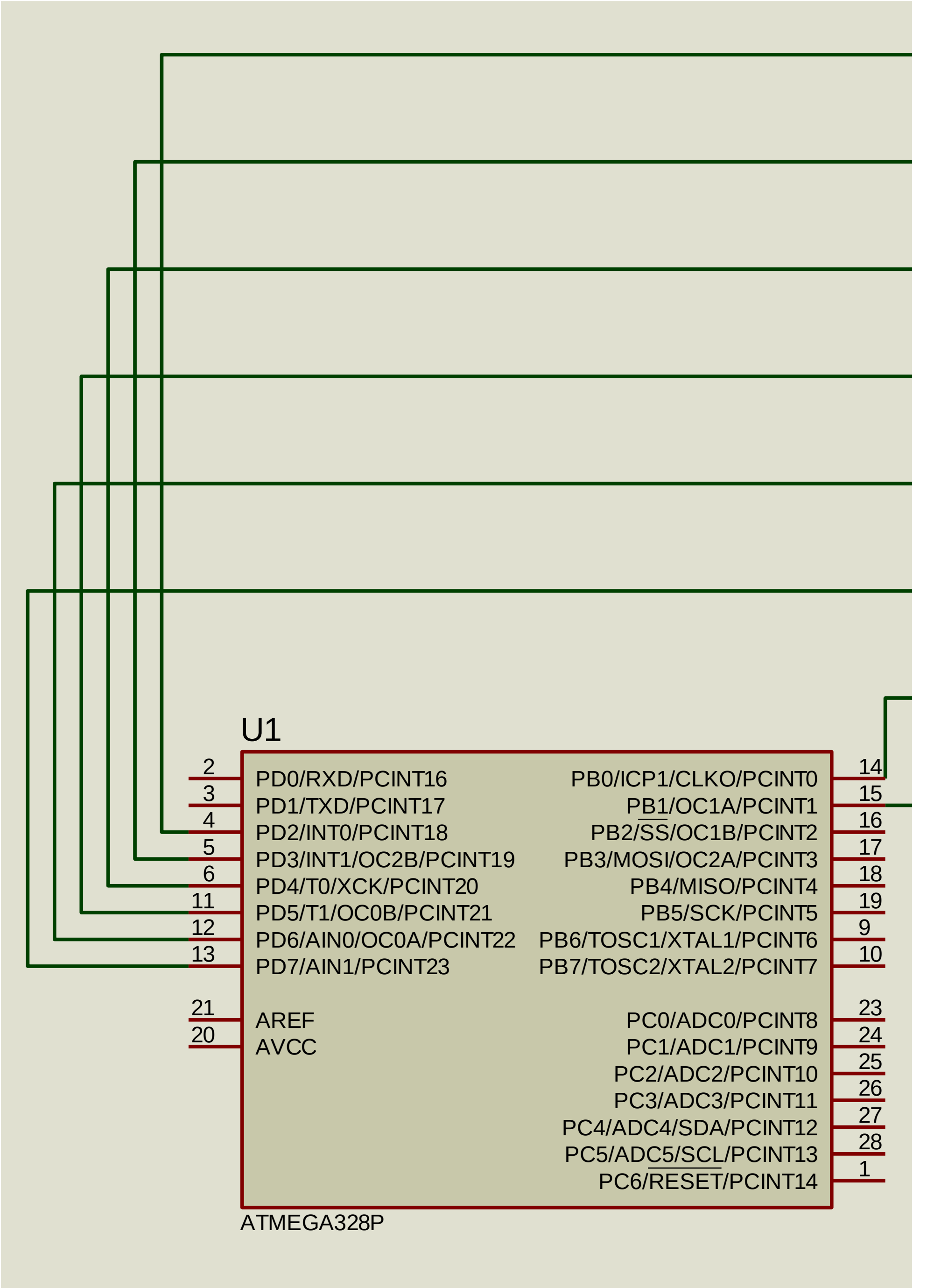


Producing Real Analog Signal

```
#define OP 2

void setup()
{
	for(int i=0;i<8;i++)
		pinMode(OP+i, OUTPUT);
}

float time = 0;
void loop()
{
	int value = 128 + 127 * sin(time);
	for(int i=0;i<8;i++)
	{
		digitalWrite(OP+i, value&0x1);
		value>>=1;
	}
	time += 0.01;
}
```



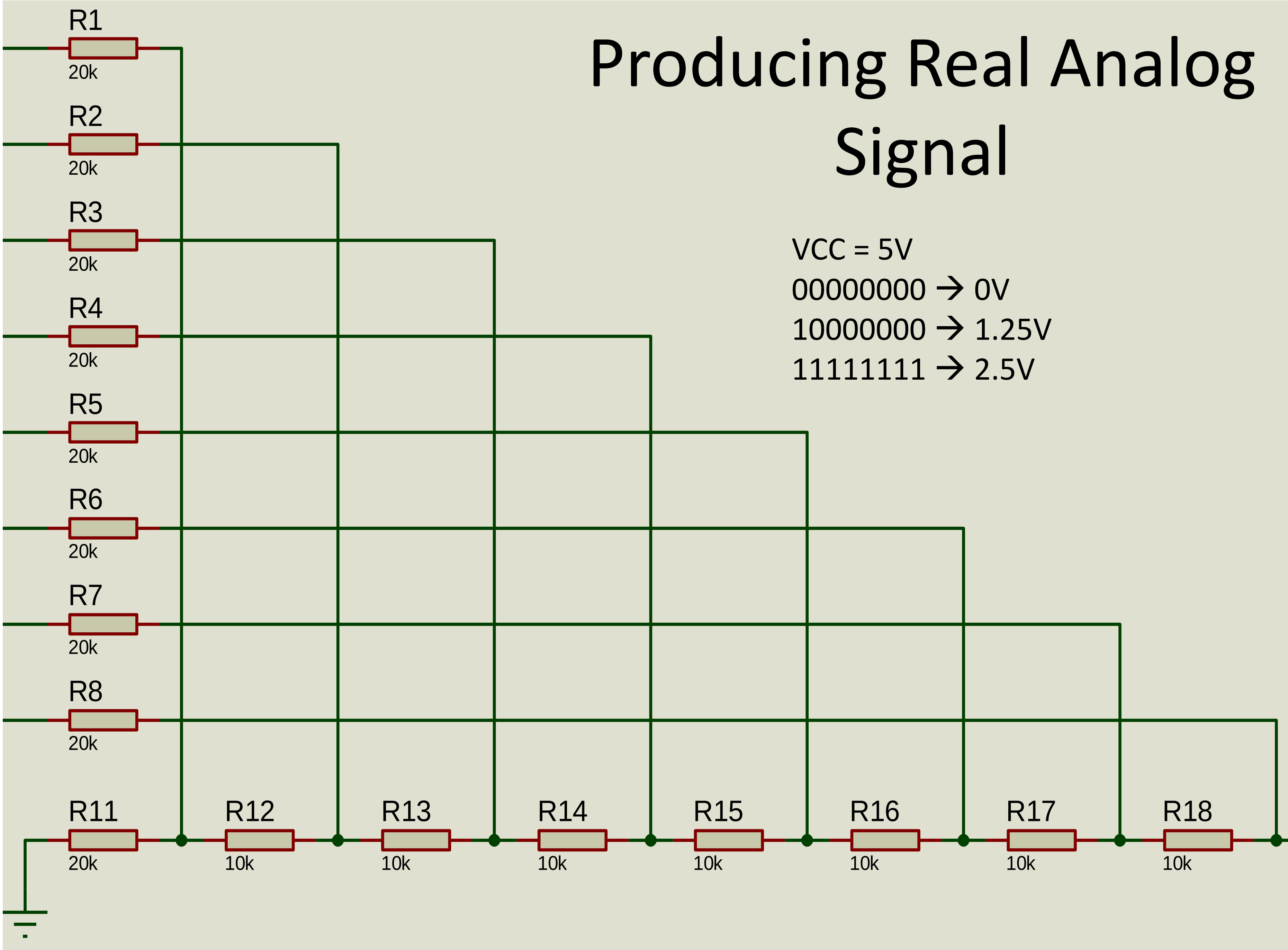
Producing Real Analog Signal

VCC = 5V

00000000 → 0V

10000000 → 1.25V

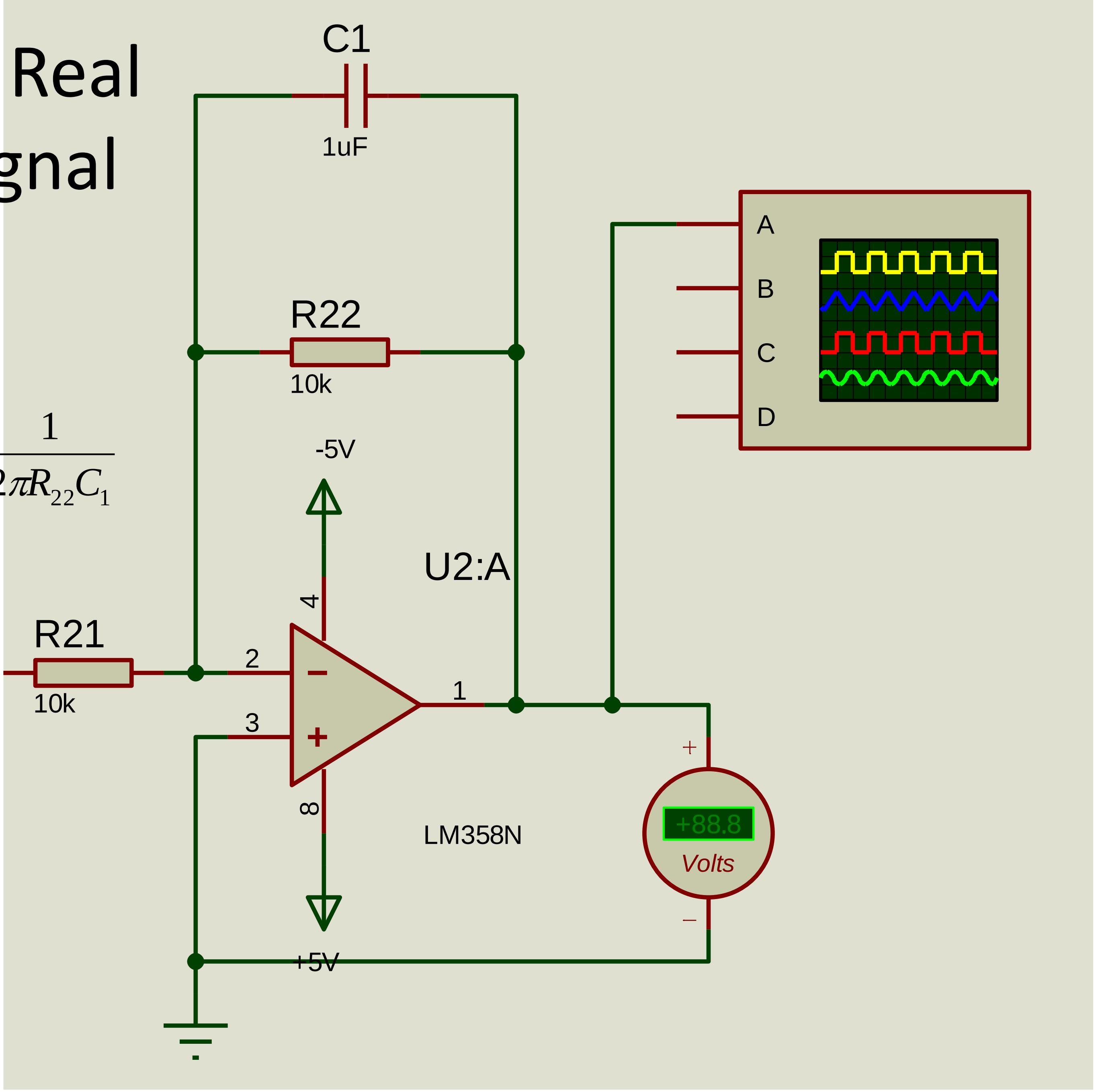
11111111 → 2.5V



Producing Real Analog Signal

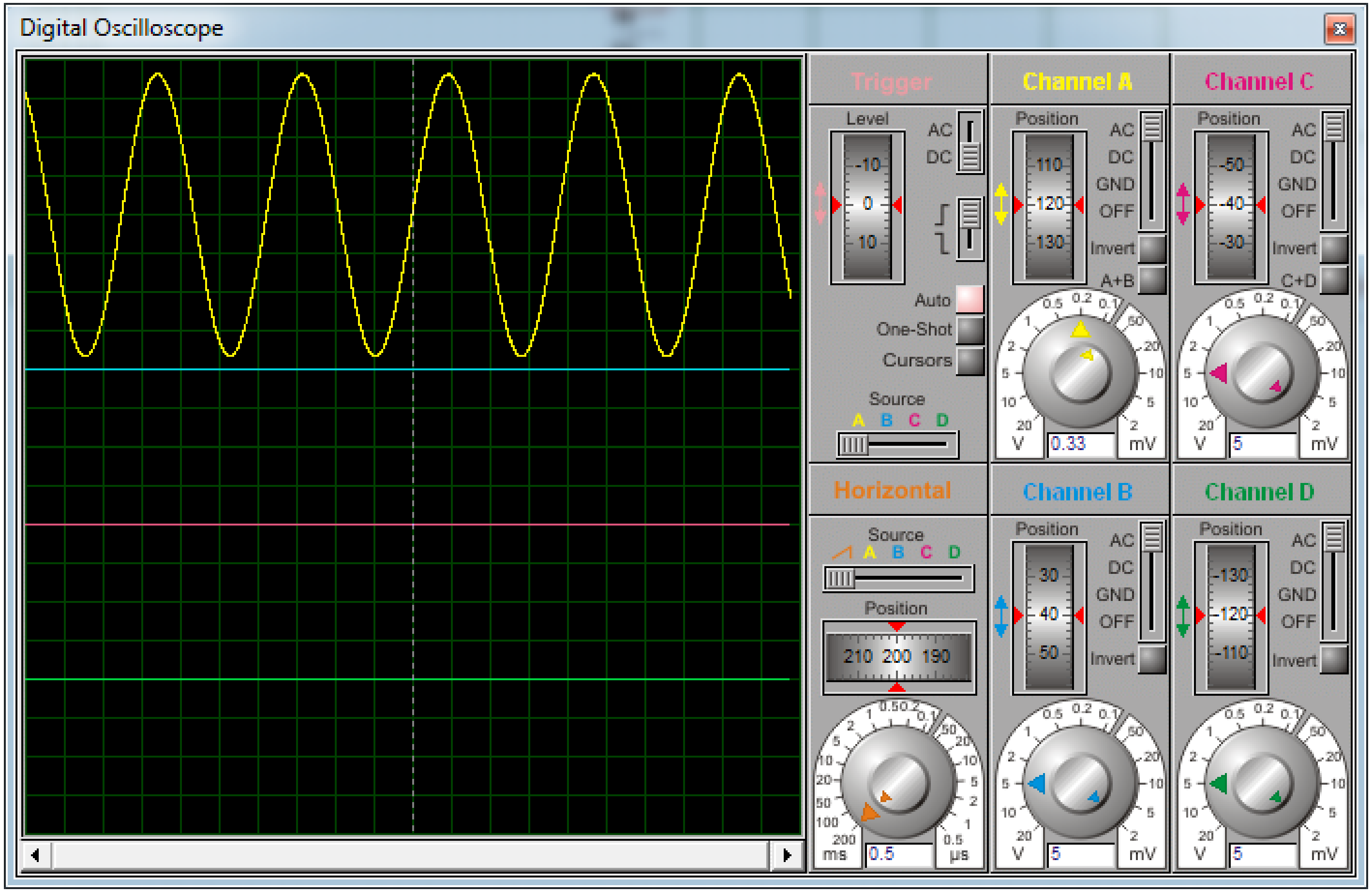
$$Gain = -\frac{R_{22}}{R_{21}}$$

$$Frequencey_{\max} = \frac{1}{2\pi R_{22} C_1}$$



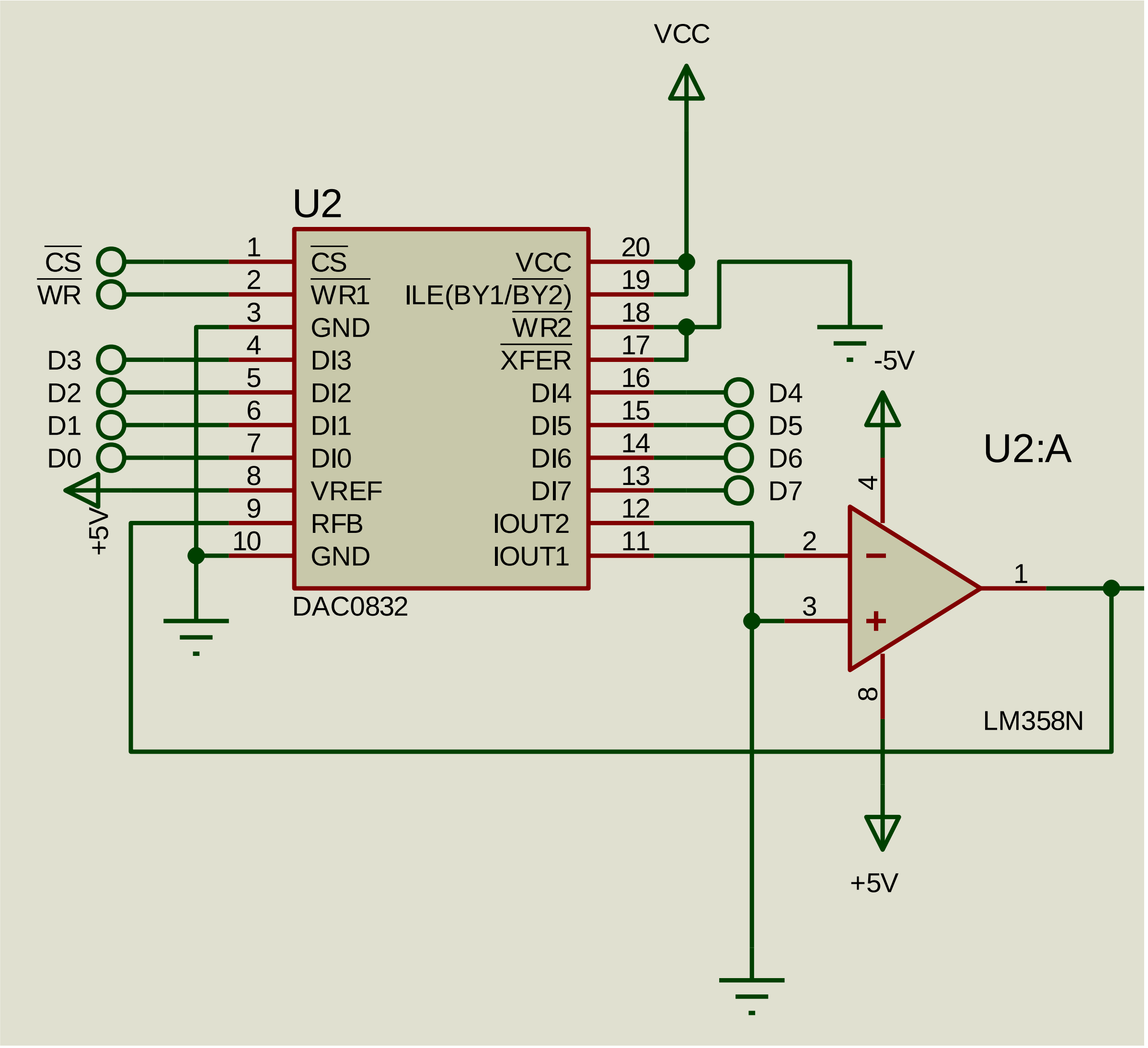
Enhancing the OP using Low Pass Filter

Producing Real Analog Signal

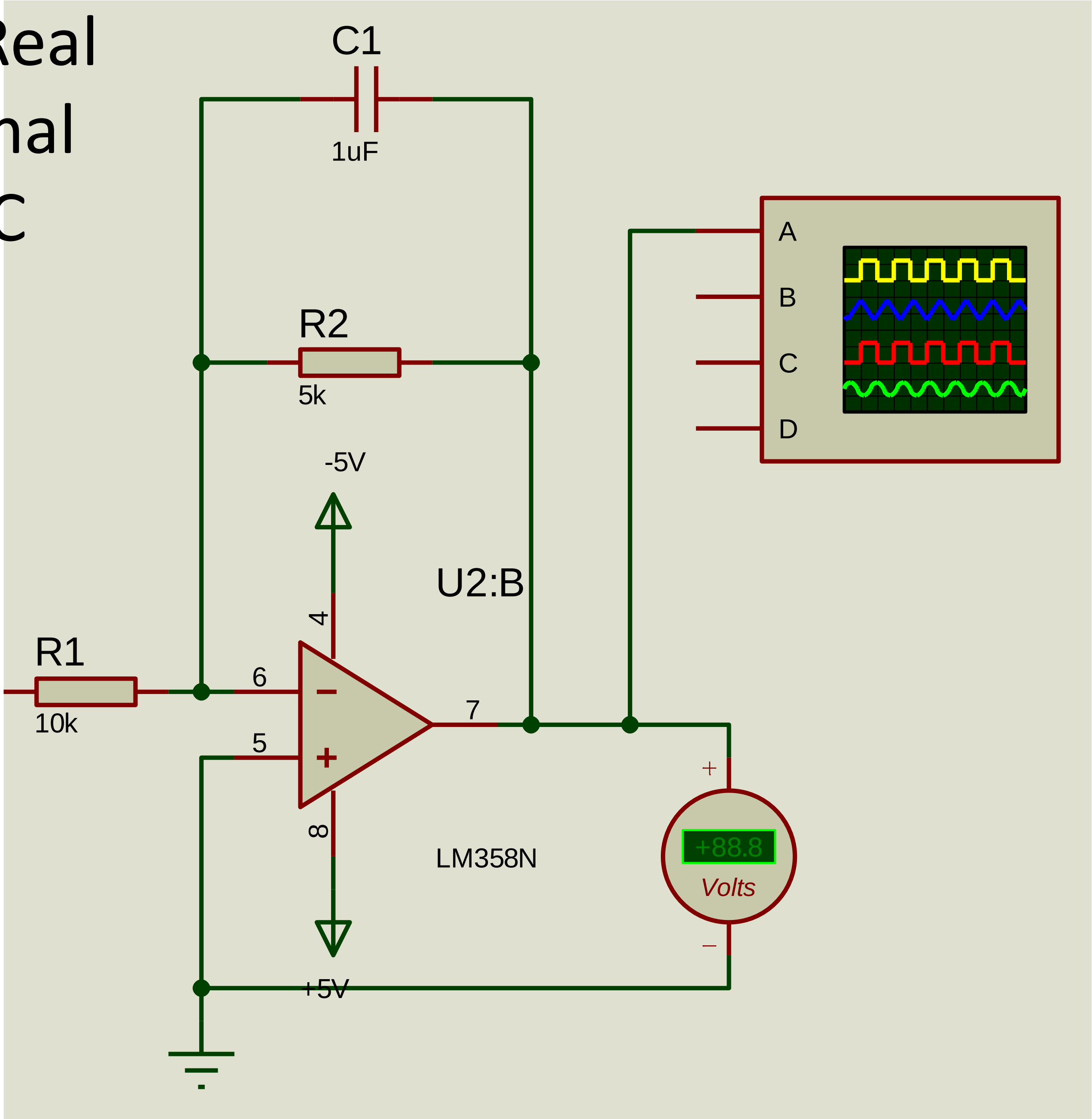


Producing Real Analog Signal using DAC

CS: Chip Select must be 1
WR: Write State must be 1 while writing
VREF: Equals the maximum output



Producing Real Analog Signal using DAC



Enhancing the OP using Low Pass Filter

Producing Real Analog Signal using DAC

```
#define OP 2
#define CS 10
#define WR 11

void setup()
{
	for(int i=0;i<8;i++)
		pinMode(OP+i, OUTPUT);
	pinMode(CS, OUTPUT);
	pinMode(WR, OUTPUT);
	digitalWrite(CS, LOW);
}

float time = 0;
void loop()
{
	digitalWrite(WR, LOW);
	int value = 128 + 127 * sin(time);
	for(int i=0;i<8;i++)
	{
		digitalWrite(OP+i, value&0x1);
		value>>=1;
	}
	digitalWrite(WR, HIGH);
	time += 0.01;
}
```