int ANIMDELAY = 100; // animation delay, default value is 100

int INTENSITYMIN = 0; // minimum brightness, valid range [0,15]

int INTENSITYMAX = 8; // maximum brightness, valid range [0,15]

int DIN\_PIN = 2; // data in pin

int CS\_PIN = 3; // load (CS) pin

int CLK\_PIN = 4; // clock pin

// MAX7219 registers

byte MAXREG\_DECODEMODE = 0x09;

byte MAXREG\_INTENSITY = 0x0a;

byte MAXREG\_SCANLIMIT = 0x0b;

byte MAXREG\_SHUTDOWN = 0x0c;

byte MAXREG\_DISPTEST = 0x0f;

const unsigned char snowflake[] =

{

 B00010000,

 B01010100,

 B00111000,

 B11111111,

 B00111000,

 B01010100,

 B00010000,

 B00010000

};

void setup ()

{

 pinMode(DIN\_PIN, OUTPUT);

 pinMode(CLK\_PIN, OUTPUT);

 pinMode(CS\_PIN, OUTPUT);

 // initialization of the MAX7219

 setRegistry(MAXREG\_SCANLIMIT, 0x07);

 setRegistry(MAXREG\_DECODEMODE, 0x00); // using an led matrix (not digits)

 setRegistry(MAXREG\_SHUTDOWN, 0x01); // not in shutdown mode

 setRegistry(MAXREG\_DISPTEST, 0x00); // no display test

 setRegistry(MAXREG\_INTENSITY, 0x0f & INTENSITYMIN);

 // draw snowflakeh

 setRegistry(1, snowflake[0]);

 setRegistry(2, snowflake[1]);

 setRegistry(3, snowflake[2]);

 setRegistry(4, snowflake[3]);

 setRegistry(5, snowflake[4]);

 setRegistry(6, snowflake[5]);

 setRegistry(7, snowflake[6]);

 setRegistry(8, snowflake[7]);

}

void loop ()

{

 // second beat

 setRegistry(MAXREG\_INTENSITY, 0x0f & INTENSITYMAX);

 delay(ANIMDELAY);

 // switch off

 setRegistry(MAXREG\_INTENSITY, 0x0f & INTENSITYMIN);

 delay(ANIMDELAY);

 // second beat

 setRegistry(MAXREG\_INTENSITY, 0x0f & INTENSITYMAX);

 delay(ANIMDELAY);

 // switch off

 setRegistry(MAXREG\_INTENSITY, 0x0f & INTENSITYMIN);

 delay(ANIMDELAY\*6);

}

void setRegistry(byte reg, byte value)

{

 digitalWrite(CS\_PIN, LOW);

 putByte(reg); // specify register

 putByte(value); // send data

 digitalWrite(CS\_PIN, LOW);

 digitalWrite(CS\_PIN, HIGH);

}

void putByte(byte data)

{

 byte i = 8;

 byte mask;

 while (i > 0)

 {

 mask = 0x01 << (i - 1); // get bitmask

 digitalWrite( CLK\_PIN, LOW); // tick

 if (data & mask) // choose bit

 digitalWrite(DIN\_PIN, HIGH); // send 1

 else

 digitalWrite(DIN\_PIN, LOW); // send 0

 digitalWrite(CLK\_PIN, HIGH); // tock

 --i; // move to lesser bit

 }

}