```
#include <DS3231.h>
#include <Wire.h>
DS3231  clock;
bool century = false;
bool h12Flag;
bool pmFlag;
int ID[10];
int NUM, tim = 10;
int digitPins[10] = {8, 9, 10, 11, 12, 13, A0, A1, A2, A3}; //Digits: 1,2,3,4 <--put
one resistor (ex: 220 Ohms, or 330 Ohms, etc, on each digit pin)
int segmentPins[7] = {1, 2, 3, 4, 5, 6, 7}; //Segments: A,B,C,D,E,F,G,Period
unsigned long bth = 0, btm = 0, bty = 0, btmo = 0, btd = 0;
int hrs_plus , min_plus , day_plus , mon_plus , yrs_plus ;
String hr, mi, dy, mo, yr;

byte Year;
byte Month;
byte Date;
byte Hour;
byte Minute;
String SDIGIT, D_STRING;

int segA = 0, segB = 0, segC = 0, segD = 0, segE = 0, segF = 0, segG = 0;

void GetDateStuff(byte& Year, byte& Month, byte& Day,
                  byte& Hour, byte& Minute) {
  boolean GotString = false;
  char InChar;
  byte Temp1, Temp2;
  char InString[20];

  for (byte j = 0; j < 10; j++) {
    InString[j] = SDIGIT[j];
  }
  // Read Year first
  Temp1 = (byte)InString[0] - 48;
  Temp2 = (byte)InString[1] - 48;
  Year = Temp1 * 10 + Temp2;
  // now month
  Temp1 = (byte)InString[2] - 48;
  Temp2 = (byte)InString[3] - 48;
  Month = Temp1 * 10 + Temp2;
  // now date
  Temp1 = (byte)InString[4] - 48;
  Temp2 = (byte)InString[5] - 48;
  Day = Temp1 * 10 + Temp2;
```

```
  // now Hour
  Temp1 = (byte)InString[6] - 48;
  Temp2 = (byte)InString[7] - 48;
  Hour = Temp1 * 10 + Temp2;
  // now Minute
  Temp1 = (byte)InString[8] - 48;
  Temp2 = (byte)InString[9] - 48;
  Minute = Temp1 * 10 + Temp2;
}

void setup() {
  Wire.begin();

  for (int i = 0; i < 10; i++) {
    pinMode(digitPins[i], OUTPUT);
  }
  for (int i = 0; i < 10; i++) {
    digitalWrite(digitPins[i], LOW);
  }
  for (int i = 0; i < 7; i++) {
    pinMode(segmentPins[i], OUTPUT);
  }
  for (int i = 0; i < 7; i++) {
    digitalWrite(segmentPins[i], LOW);
  }
 pinMode(hrs_plus, INPUT);
 pinMode(min_plus, INPUT);
 pinMode(day_plus, INPUT);
 pinMode(mon_plus, INPUT);
 pinMode(yrs_plus, INPUT);

 clock.setClockMode(false);

}

void loop() {
 getAndDisplay();

  if (digitalRead(hrs_plus) == HIGH && millis() < bth) {
    bth = millis() + 500;
    GetDateStuff(Year, Month, Date, Hour, Minute);
    clock.setHour(Hour);
  }
  if (digitalRead(min_plus) == HIGH && millis() < btm) {
    btm = millis() + 500;
    GetDateStuff(Year, Month, Date, Hour, Minute);
```

```
    clock.setMinute(Minute);
  }
  if (digitalRead(day_plus) == HIGH && millis() < btd) {
    btd = millis() + 500;
    GetDateStuff(Year, Month, Date, Hour, Minute);
    clock.setDate(Date);
  }
  if (digitalRead(mon_plus) == HIGH && millis() < btmo) {
    btmo = millis() + 500;
    GetDateStuff(Year, Month, Date, Hour, Minute);
    clock.setMonth(Month);
  }
  if (digitalRead(yrs_plus) == HIGH && millis() < bty) {
    bty = millis() + 500;
    GetDateStuff(Year, Month, Date, Hour, Minute);
    clock.setYear(Year);
  }
}


void getAndDisplay() {
  hr = String (clock.getHour(h12Flag, pmFlag), DEC);
  mi = String (clock.getMinute(), DEC);
  dy = String (clock.getDate(), DEC);
  mo = String (clock.getMonth(century), DEC);
  yr = String (clock.getYear(), DEC);
  if (hr == "0") {
    hr = "00";
  }
  else if (hr.toInt() < 10) {
    hr = "0" + hr;
  }
  if (mi == "0") {
    mi = "00";
  }
  else if (mi.toInt() < 10) {
    mi = "0" + mi;
  }
  if (dy == "0") {
    dy = "00";
  }
  else if (dy.toInt() < 10) {
    dy = "0" + dy;
  }
  if (mo == "0") {
    mo = "00";
```

```
  }
  else if (mo.toInt() < 10) {
    mo = "0" + mo;
  }
  if (yr == "0") {
    yr = "00";
  }

  SDIGIT = hr + mi + dy + mo + yr;

  for (int i = 0; i < 10; i++) {
    D_STRING = SDIGIT.substring (i, (i + 1));
    ID[i] = D_STRING.toInt();
  }

  for (int i = 0; i < 10; i++) {
    if (i == 0) {
     digitalWrite(digitPins[9], LOW);
    }
    else {
     digitalWrite(digitPins[(i - 1)], LOW);
    }
   digitalWrite(digitPins[i], HIGH);
    NUM = ID[i];
    NUM7();
    switch_7seg();
    delay(tim);
  }
}


void NUM7() {
  switch (NUM) {
    case 0:
      if (segA == 0)segA = 1;
      if (segB == 0)segB = 1;
      if (segC == 0)segC = 1;
      if (segD == 0)segD = 1;
      if (segE == 0)segE = 1;
      if (segF == 0)segF = 1;
      if (segG == 1)segG = 0;
      break;
    case 1:
      if (segA == 1)segA = 0;
      if (segB == 0)segB = 1;
      if (segC == 0)segC = 1;
```

```
  if (segD == 1)segD = 0;
  if (segE == 1)segE = 0;
  if (segF == 1)segF = 0;
  if (segG == 1)segG = 0;
  break;
case 2:
  if (segA == 0)segA = 1;
  if (segB == 0)segB = 1;
  if (segC == 1)segC = 0;
  if (segD == 0)segD = 1;
  if (segE == 0)segE = 1;
  if (segF == 1)segF = 0;
  if (segG == 0)segG = 1;
  break;
case 3:
  if (segA == 0)segA = 1;
  if (segB == 0)segB = 1;
  if (segC == 0)segC = 1;
  if (segD == 0)segD = 1;
  if (segE == 1)segE = 0;
  if (segF == 1)segF = 0;
  if (segG == 0)segG = 1;
  break;
case 4:
  if (segA == 1)segA = 0;
  if (segB == 0)segB = 1;
  if (segC == 0)segC = 1;
  if (segD == 1)segD = 0;
  if (segE == 1)segE = 0;
  if (segF == 0)segF = 1;
  if (segG == 0)segG = 1;
  break;
case 5:
  if (segA == 0)segA = 1;
  if (segB == 1)segB = 0;
  if (segC == 0)segC = 1;
  if (segD == 0)segD = 1;
  if (segE == 1)segE = 0;
  if (segF == 0)segF = 1;
  if (segG == 0)segG = 1;
  break;
case 6:
  if (segA == 0)segA = 1;
  if (segB == 1)segB = 0;
  if (segC == 0)segC = 1;
  if (segD == 0)segD = 1;
```

```
    if (segE == 0)segE = 1;
    if (segF == 0)segF = 1;
    if (segG == 0)segG = 1;
    break;
  case 7:
    if (segA == 0)segA = 1;
    if (segB == 0)segB = 1;
    if (segC == 0)segC = 1;
    if (segD == 1)segD = 0;
    if (segE == 1)segE = 0;
    if (segF == 1)segF = 0;
    if (segG == 1)segG = 0;
    break;
  case 8:
    if (segA == 0)segA = 1;
    if (segB == 0)segB = 1;
    if (segC == 0)segC = 1;
    if (segD == 0)segD = 1;
    if (segE == 0)segE = 1;
    if (segF == 0)segF = 1;
    if (segG == 0)segG = 1;
    break;
  case 9:
    if (segA == 0)segA = 1;
    if (segB == 0)segB = 1;
    if (segC == 0)segC = 1;
    if (segD == 1)segD = 0;
    if (segE == 1)segE = 0;
    if (segF == 0)segF = 1;
    if (segG == 0)segG = 1;
    break;
 }
}

void switch_7seg() {
 if (segA == 0) digitalWrite(segmentPins[0], LOW); //A
 if (segA == 1) digitalWrite(segmentPins[0], HIGH); //A
 if (segB == 0) digitalWrite(segmentPins[1], LOW); //B
 if (segB == 1) digitalWrite(segmentPins[1], HIGH); //B
 if (segC == 0) digitalWrite(segmentPins[2], LOW); //C
 if (segC == 1) digitalWrite(segmentPins[2], HIGH); //C
 if (segD == 0) digitalWrite(segmentPins[3], LOW); //D
 if (segD == 1) digitalWrite(segmentPins[3], HIGH); //D
 if (segE == 0) digitalWrite(segmentPins[4], LOW); //E
 if (segE == 1) digitalWrite(segmentPins[4], HIGH); //E
 if (segF == 0) digitalWrite(segmentPins[5], LOW); //F
```

```
  if (segF == 1) digitalWrite(segmentPins[5], HIGH); //F
  if (segG == 0) digitalWrite(segmentPins[6], LOW); //G
  if (segG == 1) digitalWrite(segmentPins[6], HIGH); //G
}
```