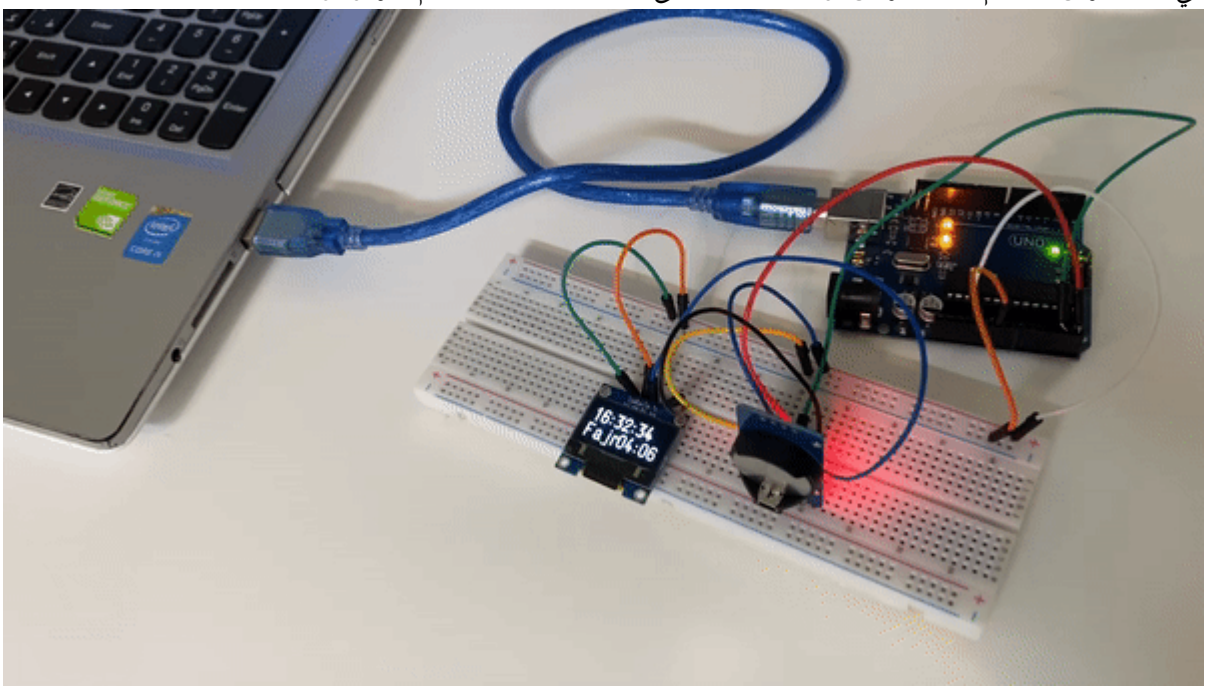
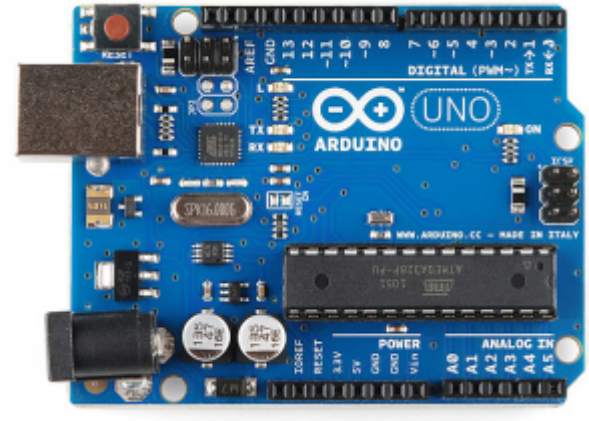
عرض أوقات الصلاة على الشاشة باستخدام الاردوينو

مقدمة

في هذا الدرس ستتعلم كيف تعرض أوقات الصلاة على شاشة Oled باستخدام الاردوينو.



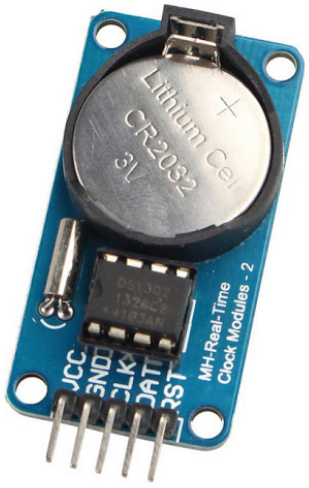
المواد والأدوات



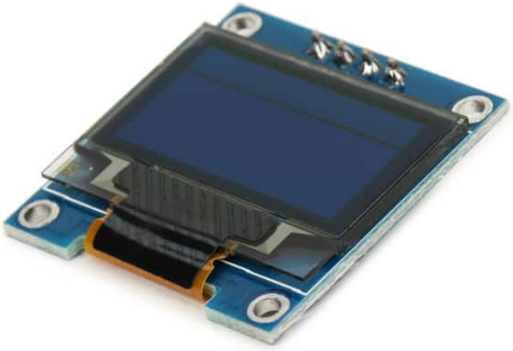
1× اردوينو اونو

1× سلك الاردوينو

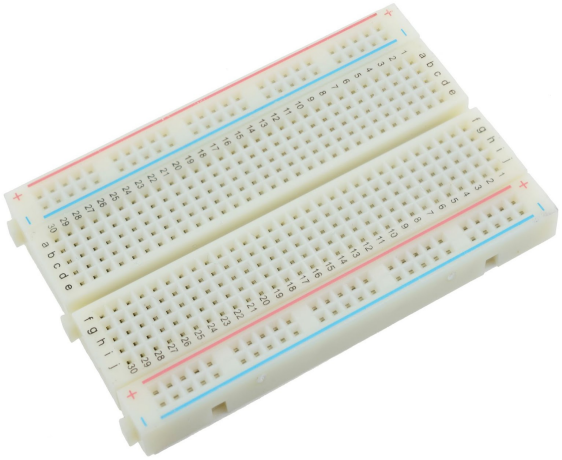
حزمة أسلاك توصيل (ذكر- ذكر)



1× وحدة الوقت الحقيقي (RTC)



1× شاشة (OLED)

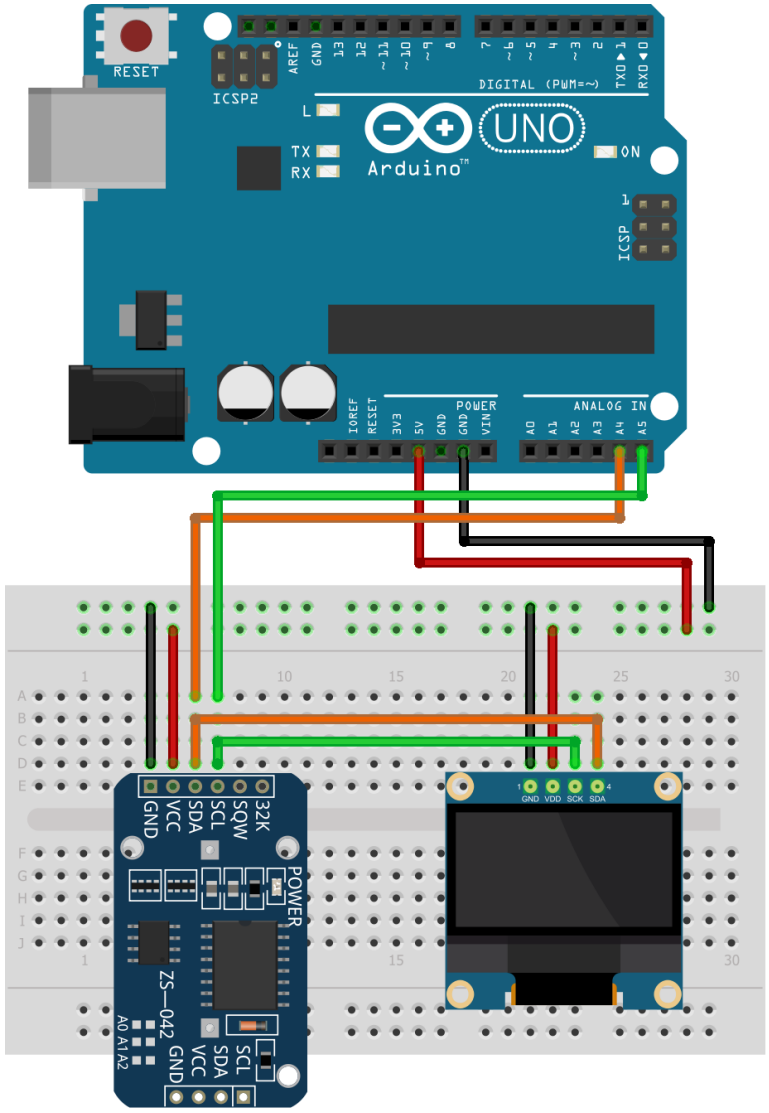


1× لوحة تجارب حجم وسط

توصيل الدائرة

للمزيد حول الشاشة (OLED) يمكنك الرجوع للدرس شاشة عرض (OLED) Display.

وللمزيد حول وحدة الوقت الحقيقي يمكنك الرجوع للدرس استخدام DS3231 RTC Module مع الاردوينو.



الكود البرمجي

بالبداية عليك تحميل المكتبات التالية.

Streaming.h و "mainroutines.h" و <U8g2lib.h>

اقرأ شرح الكود البرمجي قبل رفع الكود البرمجي إلى لوحة الاردوينو.

```
#include <Wire.h>
#include <RTClib.h>
#include <Streaming.h>
#include <SoftwareSerial.h>
#include <U8g2lib.h>
#include "mainroutines.h"

const unsigned int text1_y0=30, text2_y0=60, text2_x1=0, text2_x2=58;
char text1[8], text2[6], todaydate[9];
int displaypage=0;
const uint8_t BTTX = 3;
const uint8_t BTRX = 2;
int quartsec, OldDay, OldMinute, OldSecond, dayMinutes, NextSalat;
```

```
int i;
DateTime now;
char time[9];

U8G2_SSD1306_128X64_NONAME_2_HW_I2C u8g2(U8G2_R0, /* clock=*/ SCL, /* data=*/ SDA);
SoftwareSerial BTSerial(BTRX, BTTX);

struct FLAGS {
unsigned Recalcule : 1;
unsigned CheckTime : 1;
unsigned seconde : 1;
unsigned SeqAthan : 1;
unsigned Toggles : 1;
unsigned demiseconde: 1;
unsigned heures : 1;
unsigned displayPageTog : 1;
};
struct FLAGS Flags;

void setup()
{
Serial.begin(115200);
BTSerial.begin(9600);
BTSerial.listen();
u8g2.begin();
u8g2.setFont(u8g2_font_logisoso24_tr);
u8g2.setFontMode(0);


u8g2.firstPage();
do {
u8g2.drawUTF8(0,text1_y0,"SalatTime");
u8g2.drawUTF8(0,text2_y0,"L. Baghli");
}
while ( u8g2.nextPage() );

Wire.begin();
rtc.begin();
if (! rtc.isrunning()) {
Serial.println(F("RTC NOT running!"));
rtc.adjust(DateTime(2021,8, 24, 16,8, 0));
}

Serial.flush();
STinit();
Flags.Recalcule=1;
Flags.heures=0;
Flags.CheckTime=0;
Flags.displayPageTog=0;
cli();
TCCR1A = 0;
TCCR1B = (0<<WGM13) | (1<<WGM12) | 4;
OCR1A = 15625;
```

```
TIMSK1 = 1<<OCIE1A;
TIFR1 = 0;
TCNT1 = 0;
sei();
}
void CheckTime()
{
now = rtc.now();
Flags.Toggles = !Flags.Toggles;
if (OldSecond != now.second()) {
if (++displaypage>5) displaypage=0;
OldSecond = now.second();
Serial << now.day() << F("/")<< now.month() << F("/")<<now.year()
<< F(" ")<< now.hour() << F(":")<<now.minute() << F(":")<<now.second() << endl;
}
if (OldDay != now.day()) Flags.Recalcule = 1;
if ((OldMinute != now.minute()) && (Flags.Recalcule == 0))
{
if ((SalatT.m[NextSalat]==now.minute()) && (SalatT.h[NextSalat]==now.hour()))
{// Athan
Flags.SeqAthan = 1;
}
else Flags.SeqAthan = 0;
OldMinute = now.minute();
dayMinutes = now.hour()*60+now.minute();
for (i=4; i>=0; i--)
if (SalatT.h[i]*60 + SalatT.m[i] >= dayMinutes)
{
NextSalat = i;
Serial << F("SalatT.m[")<< i <<F("] =") << SalatT.m[i] << endl;
}
}
}
void MaJRTC(int * dt)
{
if ((dt[0]<1) || (dt[0]>31)) return;
if ((dt[1]<1) || (dt[1]>12)) return;
if ((dt[2]<2000) || (dt[2]>2099)) return;
if ((dt[3]<0) || (dt[3]>23)) return;
if ((dt[4]<0) || (dt[4]>59)) return;
if ((dt[5]<0) || (dt[5]>59)) return;
rtc.adjust(DateTime(dt[2], dt[1], dt[0], dt[3], dt[4], dt[5]));
}
ISR(TIMER1_COMPA_vect) {
if (Flags.SeqAthan)
{
Flags.Toggles = !Flags.Toggles;
}
if (++quartsec == 4) quartsec = 0;
Flags.CheckTime = 1; // check time chaque 1/4 seconde
Flags.displayPageTog = 1; // page affichée chaque 1/4 seconde
if (quartsec == 0) {
Flags.seconde = 1;
```

```
}
}
void DisplayPages()
{
switch(displaypage)
{
case 0: sprintf( text1, "Fajr");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[0],SalatT.m[0] );
break;
case 1: sprintf( text1, "Duhr");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[1],SalatT.m[1] );
break;
case 2: sprintf( text1, "Asr");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[2],SalatT.m[2] );
break;
case 3: sprintf( text1, "Mgrb");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[3],SalatT.m[3] );
break;
case 4: sprintf( text1, "Isha");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[4],SalatT.m[4] );
break;
for (int i=0; i<8; i++) text1[i] = todaydate[i];
sprintf( text2, "");
break;
}
sprintf( time, "%02hhu:%02hhu:%02hhu", now.hour(), now.minute(), now.second() );
u8g2.firstPage();
do {
u8g2.drawUTF8(0,text1_y0,time);
u8g2.drawUTF8(text2_x1,text2_y0, text1);
u8g2.drawUTF8(text2_x2,text2_y0, text2);
} while ( u8g2.nextPage() );
}
void loop()
{
if (Flags.CheckTime) {
CheckTime();
Flags.CheckTime = 0;
}
if (Flags.Recalcule) {
ComputeSalatTime();
OldDay = now.day();
sprintf( todaydate, "%02hhu/%02hhu/%02hhu", now.day(),now.month(),now.year()%100 );
todaydate[8]=0;
OldMinute = 61;
OldSecond = 61;
Flags.Recalcule = 0;
}
if (Flags.displayPageTog) {
DisplayPages();
Flags.displayPageTog = 0;
}
```

```
}
```

شرح الكود البرمجي

هنا تم استدعاء المكتبات التي سنستخدمها لعرض أوقات الصلاة على الشاشة.

```
#include <Wire.h>
#include <RTClib.h>
#include <Streaming.h>
#include <SoftwareSerial.h>
#include <U8g2lib.h>
#include "mainroutines.h"
```

هنا يتم تعريف المتغيرات التي سنستخدمها لتخزين النص الذي سينطبع على الشاشة.

```
char text1[8], text2[6], todaydate[9];
```

ربط المنافذ في الشاشة OLED بحيث يكون المنفذ SCL مع المنفذ SCL الموجود في وحدة الوقت والمنفذ SDA مع المنفذ SDA في وحدة الوقت.

```
U8G2_SSD1306_128X64_NONAME_2_HW_I2C u8g2(U8G2_R0, /* clock=*/ SCL, /* data=*/ SDA);
```

يتم تقسيم الجهد 5V BTTX يحمل القيمة 3 و BTRX يحمل القيمة 2.

```
const uint8_t BTTX = 3;
const uint8_t BTRX = 2;
```

في الدالة ()setup يتم تهيئة الشاشة استعدادًا لطباعة الوقت وحساب أوقات الصلاة.

```
void setup()
{
Serial.begin(115200);
BTSerial.begin(9600);
BTSerial.listen();
u8g2.begin();
u8g2.setFont(u8g2_font_logisoso24_tr);
u8g2.setFontMode(0);
u8g2.firstPage();
do {
u8g2.drawUTF8(0,text1_y0,"SalatTime");
u8g2.drawUTF8(0,text2_y0,"L. Baghli");
}
while ( u8g2.nextPage() );
Wire.begin();
rtc.begin();
if (! rtc.isrunning()) {
Serial.println(F("RTC NOT running!"));
rtc.adjust(DateTime(2021,8, 24, 16,8, 0));
```

```
}
```

هذا السطر لابد من تعديله حسب التاريخ والوقت الحالي لديك و كتابة الوقت بالثواني و الدقائق و الساعة ، و التاريخ اليوم و الشهر والسنة

```
rtc.adjust(DateTime(2021,8, 24, 16,8, 0));
```

هنا يتم تهيئة المتغيرات التي تستخدم لحساب أوقات الصلاة.

```
STinit();
Flags.Recalcule=1;
Flags.heures=0;
Flags.CheckTime=0;
Flags.displayPageTog=0;
```

في الدالة ()CheckTime يتم حساب الوقت وطباعته على الشاشة كل ثانية.

```
void CheckTime()
{
now = rtc.now();
Flags.Toggles = !Flags.Toggles;
if (OldSecond != now.second()) {
if (++displaypage>5) displaypage=0;
OldSecond = now.second();
Serial << now.day() << F("/")<< now.month() << F("/")<<now.year()
<< F(" ")<< now.hour() << F(":")<<now.minute() << F(":")<<now.second() << endl;
}
if (OldDay != now.day()) Flags.Recalcule = 1;
if ((OldMinute != now.minute()) && (Flags.Recalcule == 0))
{
if ((SalatT.m[NextSalat]==now.minute()) && (SalatT.h[NextSalat]==now.hour()))
{
Flags.SeqAthan = 1;
}
else Flags.SeqAthan = 0;
OldMinute = now.minute();
dayMinutes = now.hour()*60+now.minute();
for (i=4; i>=0; i--)
if (SalatT.h[i]*60 + SalatT.m[i] >= dayMinutes)
{
NextSalat = i;
Serial << F("SalatT.m[")<< i <<F("] =") << SalatT.m[i] << endl;
}
}
}
```

هنا يتم برمجة وحدة الوقت بحيث الساعات باليوم تكون 24 والدقائق 60 دقيقة للساعة و60 ثانية بالدقيقة.

```
void MaJRTC(int * dt)
{
if ((dt[0]<1) || (dt[0]>31)) return;
```

```
if ((dt[1]<1) || (dt[1]>12)) return;
if ((dt[2]<2000) || (dt[2]>2099)) return;
if ((dt[3]<0) || (dt[3]>23)) return;
if ((dt[4]<0) || (dt[4]>59)) return;
if ((dt[5]<0) || (dt[5]>59)) return;
rtc.adjust(DateTime(dt[2], dt[1], dt[0], dt[3], dt[4], dt[5]));
}
ISR(TIMER1_COMPA_vect)
{
if (Flags.SeqAthan)
{
Flags.Toggles = !Flags.Toggles;
}
if (++quartsec == 4) quartsec = 0;
Flags.CheckTime = 1;
Flags.displayPageTog = 1;
if (quartsec == 0) {
Flags.seconde = 1;
}
}
```

هنا سيتم برمجة الشاشة لتحتوي على ست شاشات مختلفة في المحتوى.

الأولى: عرض موعد صلاة الفجر.

الثانية: عرض موعد صلاة الظهر.

الثالثة: عرض موعد صلاة العصر.

الرابعة: عرض صلاة المغرب.

الخامسة: عرض صلاة العشاء.

السادسة: عرض تاريخ اليوم.

```
void DisplayPages()
{
switch(displaypage)
{
case 0: sprintf( text1, "Fajr");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[0],SalatT.m[0] );
break;
case 1: sprintf( text1, "Duhr");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[1],SalatT.m[1] );
break;
case 2: sprintf( text1, "Asr");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[2],SalatT.m[2] );
break;
case 3: sprintf( text1, "Mgrb");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[3],SalatT.m[3] );
break;
case 4: sprintf( text1, "Isha");
sprintf( text2, "%02hhu:%02hhu", SalatT.h[4],SalatT.m[4] );
```

```
break;
for (int i=0; i<8; i++) text1[i] = todaydate[i];
sprintf( text2, "");
break;
}
sprintf( time, "%02hhu:%02hhu:%02hhu", now.hour(), now.minute(), now.second() );
u8g2.firstPage();
do {
u8g2.drawUTF8(0,text1_y0,time);
u8g2.drawUTF8(text2_x1,text2_y0, text1);
u8g2.drawUTF8(text2_x2,text2_y0, text2);
} while ( u8g2.nextPage() );
}
```

في الدالة ()loop سيتم تحديث الوقت وموعد الصلاة والتاريخ بشكل مستمر.

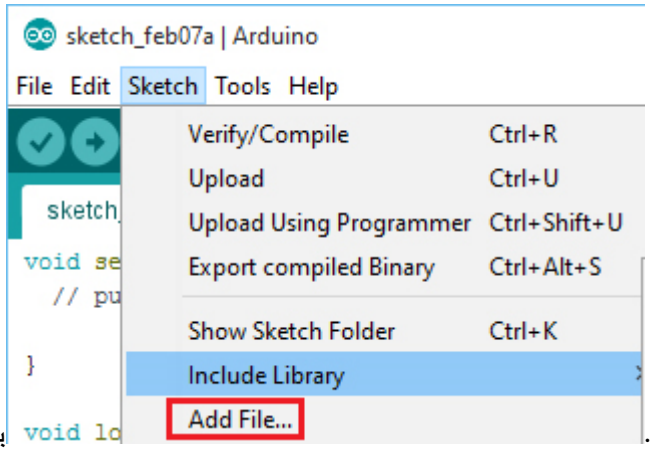
```
void loop()
{
if (Flags.CheckTime) {
CheckTime();
Flags.CheckTime = 0;
}
if (Flags.Recalcule) {
ComputeSalatTime();
OldDay = now.day();
sprintf( todaydate, "%02hhu/%02hhu/%02hhu", now.day(),now.month(),now.year()%100 );
todaydate[8]=0;
OldMinute = 61;
OldSecond = 61;
Flags.Recalcule = 0;
}
if (Flags.displayPageTog) {
DisplayPages();
Flags.displayPageTog = 0;
}
}
```

حمّل الملفات التالية التي تحتوي على الدوال المهمة؛ لحساب أوقات الصلاة بناء على موقعك الجغرافي.

mainroutines.cpp

و

mainroutines.h



بعد انقر على Add File أضف الملفات اللذان قمت بتحميلهما.

إضافتهما يمكنك التعديل عليهما اذهب إلى الملف mainroutines.cpp.

حرر الأسطر التالية لكي تناسب منطقة السكن لديك.

ادخل على هذا الرابط وضع منطقة السكن لديك (استخدم نسخة Expert).

عدّل []CountryName و []TownName و TimeZoneTown و Convention و DST بناء على البيانات التي عرضها.

استخدم هذا الرابط لعرض خريطة منطقة السكن لديك.

انسخ وعدل قيم latitude و longitude الموجودة في الكود البرمجي.

في هذا الدرس استخدمنا البيانات الخاصة بمدينة الرياض.

```
/** change your town */
        const char CountryName[] PROGMEM =  "Arabie Saoudite";
        const char TownName[] PROGMEM = ""Ar Riyad";
        const double latitude = 24.6498255*deg2rd;
        const double longitude = 46.7687988*deg2rd;
        const int TimeZoneTown = 3;
        const int Convention =  5;
        const int DST = 0;
```

بعد تعديل الكود البرمجي يمكنك رفع الكود إلى لوحة الاردوينو.

يمكنك اختبار صحة خطواتك.

لا تنسَ فصل مصدر الطاقة بعد الانتهاء من استخدام النظام.