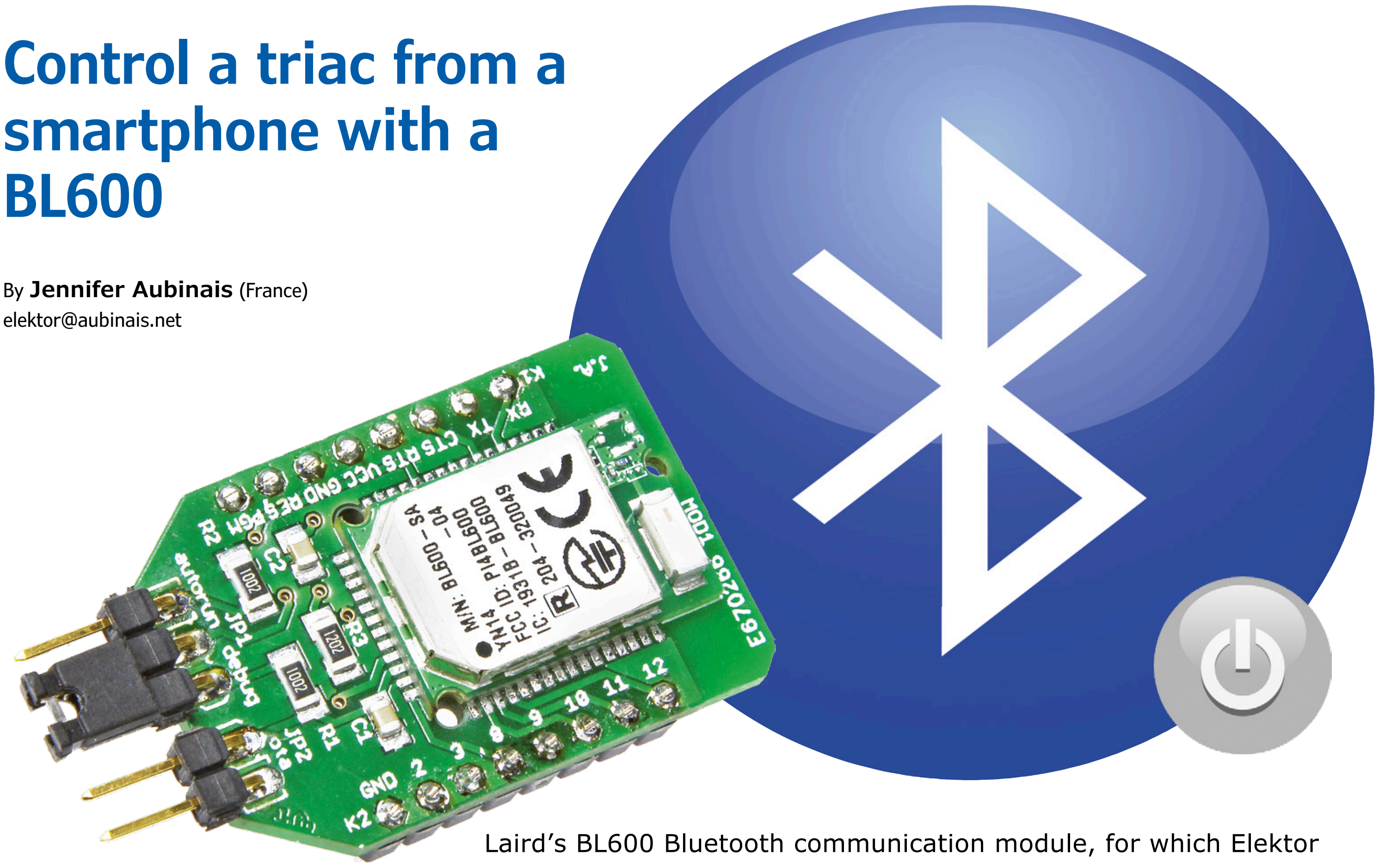
Line AC Switch Controlled by Bluetooth Low Energy

Control a triac from a smartphone with a BL600

By **Jennifer Aubinais** (France)
elektor@aubinais.net



Laird's BL600 Bluetooth communication module, for which Elektor offers the famous e-BoB breakout board, is used here in a complete new and autonomous project to switch mains voltage, for example switch a light on or off from an Android smartphone.

This article describes the electronics, the BL600 program and the Android program; the last being an Android smartphone application with which your mains switch circuit may be controlled wirelessly. You may notice an unusual detail (for these pages) — the transformerless power supply — and that is where we will begin (**Figure 1**).

The electronics

We can separate the schematic into three parts:

- The power supply: C1, R1, R2, R7, R8, D1 to D5, C2, IC1;
- The *Bluetooth Low Energy* module: MOD1, R4, LED1;
- The switching: T1, IC2, R5, R6, TRI1, R3, LED.

The power supply is notable in that to save space, it does away with a voltage transformer and thus poses a **risk of electrocution**, because there is no galvanic isolation between this circuit and the AC line (230 V or 115 V). This demands extra vigilance and care, and strict respect for the rules for construction and use of this type of circuit.

The BL600 module (MOD1) is powered at low DC voltage by a 3.3-V regulator (IC1), the input voltage of which is limited by a 6.8-V zener diode (D2). To correctly dimension the power components we must estimate the total consumption of the Bluetooth switch components:

- BL600 module: 6 mA while transmitting;
- 2 LEDs: 3.3 [V] – 1.6 [V]) / 1000 [Ω] = 2 [mA] so 4 mA for the two LEDs (estimate);
- MOC3041 optocoupler (3.3 [V] – 1.3 [V]) / 220 [Ω] = 9 [mA];
- LP2950 regulator current max.: 12 mA;

Giving a total of some 31 mA. This modest consumption allows us to dispense with a power transformer and instead use a capacitor (C1). It is this capacitor which is the main part of the power circuit that reduces the voltage from the AC line (230 VAC or 115 VAC) applied to K1. In effect, the capacitive reactance of a capacitor, designated X_c and expressed in Ω, is, under ideal conditions, a **resistance to alternating current**, inversely proportional to the frequency of the AC voltage. Here this frequency is that of the power line, 50 Hz in Europe, 60 Hz in the USA. From this we can calculate the capacitance required as follows:

$X_c = 1 / (2 \pi f C)$

With
f: frequency in Hz
C: capacitance in farads
$C = 0.031 / (2 \times 3.14 \times 50 \times (230 - 6.8)) = 0.44$ µF
We can use **470 nF** for 230 VAC line voltage, or **1 µF** for 115 VAC line voltage.

Calculation of resistors

On the application of power, C1 is discharged and effectively is briefly seen as a short circuit; to reduce the peak current that may result, we insert two resistors in series (because of the possible maximum of 325 V which most resistors are not rated for). The total resistance of 650 Ω (2 x 330 Ω) limits the current to around 500 mA. As soon as C1 is charged, the current through the resistors falls to 30 mA, and the power to 300 mW, but taking account of the possible much stronger peak power, we have rated these resistors at 1 W.
The purpose of 4.7-MΩ resistors R7 and R8 of is to ensure that the 470-nF capacitor is discharged once AC line power is removed. Be cautious around these components, even when disconnected from the grid.

Rectification and regulation

Once the AC line voltage is reduced by C1, a bridge rectifier of four 1N4007 diodes (to use both half-cycles) which feeds an LP2950 to obtain the low voltage DC required for the BL600 module. The entry voltage for the regulator, limited to 6.8 V by D2, is smoothed by the 100-µF electrolytic capacitor C2 and filtered at input and output by 100-nF capacitors.

Optocoupler

Because of the use of the bridge rectifier, the voltage of the ground of the BL600 Bluetooth module floats with respect to the AC line wires. We therefore use an optocoupler (IC2) which maintains galvanic isolation between the output pin of the low voltage circuit (pin 12 of MOD1) and the triac circuit at 230 V.
But make no mistake, this 'isolation' thus obtained between the 230 V AC line voltage and the Bluetooth module does not

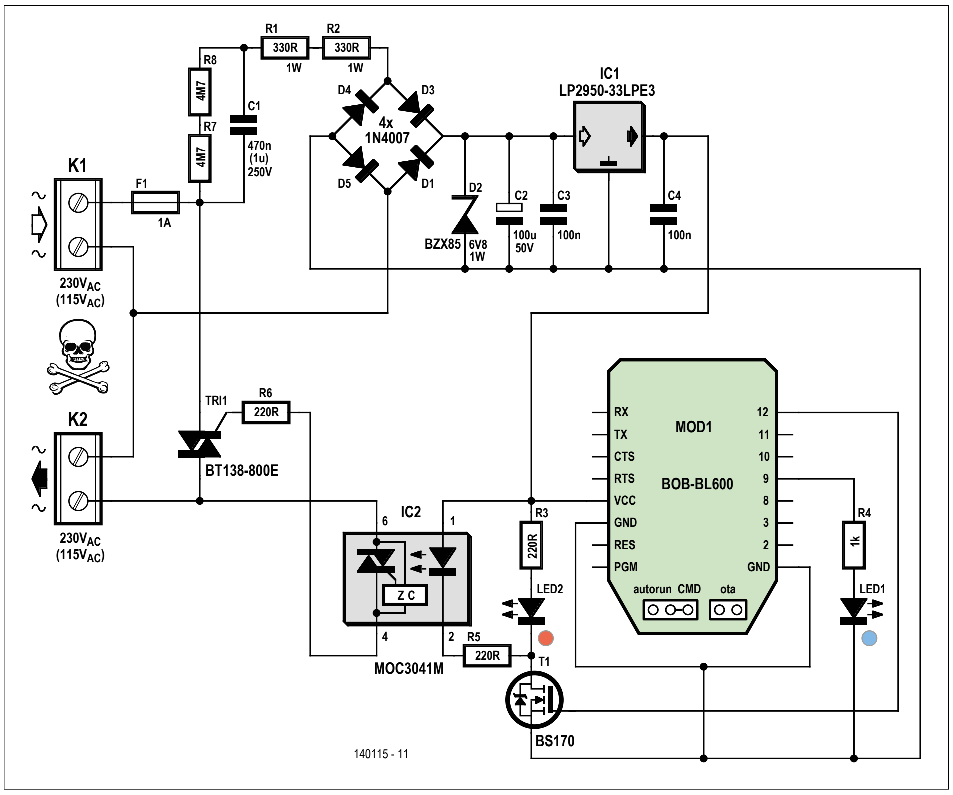


Figure 1. The absence of a transformer is (partially) compensated for in the schematic by a skull and crossbones. Don't let your fingers or other body parts get anywhere near this circuit!

change anything in respect of guarding against the risk of electrocution.

Output GPIO12

During tests at the Elektor Labs, it was found that the current requirements of the LED of the optocoupler and LED2 were too high for the output of the BL600 module to drive directly, so we have added a FET (T1).

LED

The blue LED shows the status of the Bluetooth connection on pin GPIO9: it flashes when the module is not connected and is on when it is. The red LED on pin GPIO12 shows the state of the optocoupler and thus the status of the load connected to the terminals (K2) controlled by the triac.

The triac

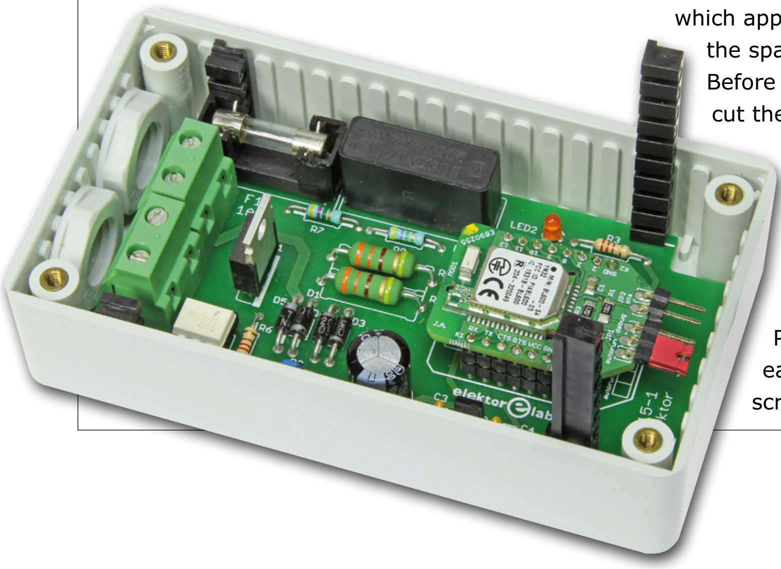
According to the NXP datasheet, the maximum average RMS current in the triac is 12 A. However the heating effects come into play here: a TO-220 package without a heatsink can dissipate around 1.5 watts at ambient temperature, and it is this which limits the current. The maximum allowable current is then estimated at 0.6 A, let's say 0.5 A to give us a safe margin. The maximum power of the load is thus around 115 W — again let's say 100 W to be safe.

Construction and test

Having programmed our eBOB-BL600 (see the section on the application for the module), construction and implementation should not pose particular problems, but we must not lose sight of the fact that this circuit, having no transformer, is connected direct to the AC line voltage. For our initial tests, instead of powering the switch from the grid, we'll use a low voltage DC lab power supply connected to the solder pads of D2, before this diode is fitted to the board. Thus may we test the correct functioning of the BL600 without connecting the circuit to any dangerous voltages. In the absence of a lab power supply, you can use any mains adapter giving a DC output of 7 to 9 volts. We can then simply check for the establishment of a Bluetooth connection between the BL600 and the tablet or smartphone. Fortunately, with the circuit supplied from low voltage DC, it is also possible to program the BL600 in OTA mode!
LED2 shows if the switch is on or off. Establish communication with the Android application (see below) or use the appli-

Mounting in the case

Whatever type of case you use, don't forget to use cable glands to protect the wires from strains and pulling. There was not a lot of space in the model we used, but we got there. The nice feature of this particular case is that it can take four grooved and trimmable PCB guides which allow the PCB to be secured in place without any screws! On one side of these strips are grooves in which the PCB can sit, the other side has a U-shaped profile that allows them to slide up and down the grooves in the sides of the case. A very clever system which appreciably reduces the space requirements. Before closing the case, cut the strips to the right length — just up to the cover of the case. The other advantage is that you can fit and remove the PCB quickly and easily without using a screwdriver.



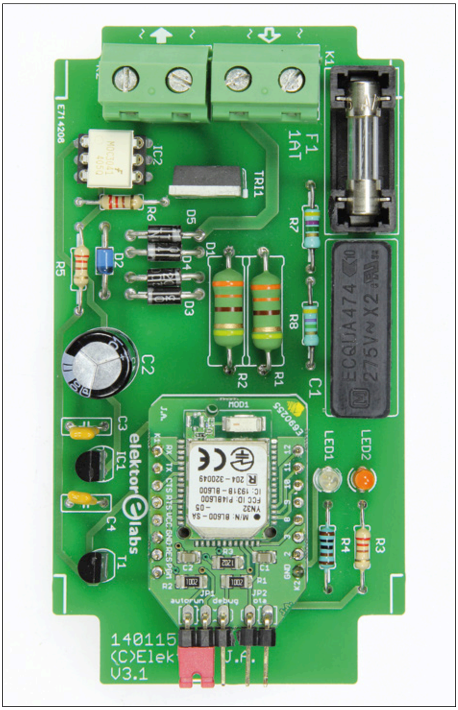


Figure 3. The prototype assembled and ready for installation in the case.

Component List

Resistors
R1,R2 = 330Ω 1W
R3,R5,R6 = 220Ω 0.25W
R4 = 1kΩ 0.25W 250V
R7,R8 = 4.7MΩ carbon film 0.25W 250V

Capacitors
C1 = 470nF 275 VAC, 22.5mm pitch (for 230 VAC)
C1 = 1µF 275 VAC 22.5 mm pitch (for 115 VAC)
C2 = 100µF 50 V radial
C3,C4 = 100nF 50V

Semiconductors:
LED1 = LED, red, 3mm
LED2 = LED, blue, 3mm
D1,D3,D4,D5 = 1N4007
D2 = BZX85C6V8 (6.8V 1W zener diode)
T1 = BS170
Tri1 = BT138-V
IC1 = LP2950-33
IC2 = MOC3041M

Miscellaneous
F1 = fuse, 1A slow-blow 20mm
Fuseholder, 5x20 mm (1162740)
K1,K2 = 2-way PCB screw terminal 0.3'' (7.68mm), 630V rated (1793006)
MOD1 = BL600 e-BoB, ready assembled, ElektorSTORE # 140270-91
Case = Hammond 1591BTCL (transparent) (1877127) or 1591BGY (grey), 112x62x31 mm (4437019)
PCB adapters for case (1876930)
PG7 cable gland (Lapp Cable) (1178859)

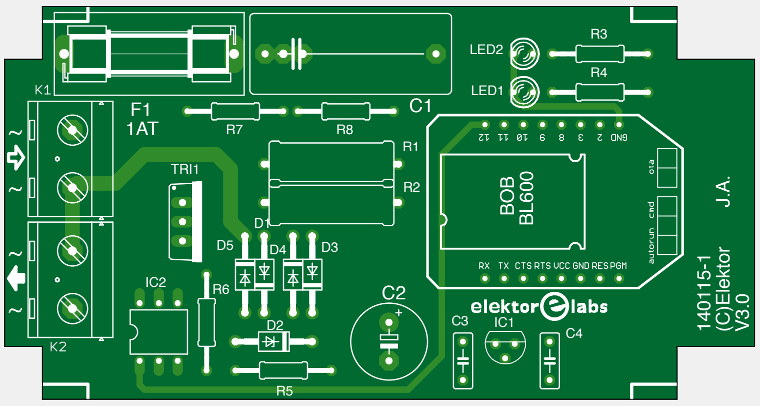


Figure 2. The printed circuit board of the BT-LE controlled AC Line switch.

cation *Serial* from Laird: using the Scan function, start by establishing a connection with the BL600, and select "JA_SWITCH". Choose the option *Connect*. LED2 should react when you touch the button in the application, or when *Serial* sends a 1 (on) or a 0 (off).

Everything working OK? Install D2 on the PCB and then mount it, **well isolated**, in the case before you connect the mains voltage.

The application for the module

Both while testing the board and when programming the BL600, it should never

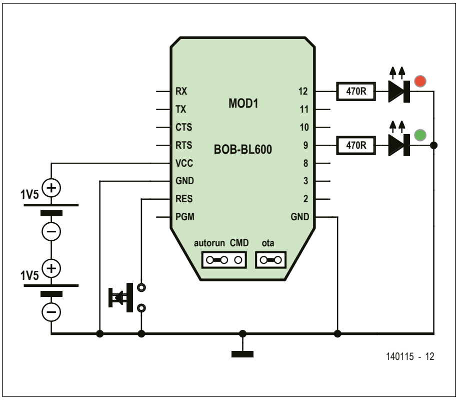


Figure 4. Test Circuit for programming of the BL600 — mount it on a breadboard away from AC line electricity.

be handled while it is connected to mains voltage. For programming, we recommend to mount the e-BoB BL600 on a breadboard (**Figure 4**) powered by two batteries, for loading the program.
You'll need two 470-Ω resistors, two LEDs (red and green), a pushbutton, and a battery holder and two AAA batteries (if you don't have those in your junk box, use the ones from your TV remote.)

Download from the Elektor website [5] to your Android phone the file $autorun$.pgmSWITCH.uwc, as well as the application Laird Toolkit [6] from their website. On the e-BoB BL600, put JP2 in the OTA position and JP1 in the autoRUN position, then power up your test setup. Start the Toolkit application and choose the OTA option (**Figure 5**), then tap on *Select File* et find the file $autorun$.pgmSWITCH.uwc on your telephone (**Figure 6**). Start *Scan*, then choose your module LAIRD BL600. If it does not work, reset the module (RESET). Tap on *Upload*... During the transfer from your telephone to the e-BoB, the scroll bar will progress… at the end, **don't forget to tap *DISCONNECT*.** Your BL600 is programmed. The green LED will flash. While you have your switch mounted on a breadboard, take the opportunity, before connecting the AC line voltage, to test the BL600 with the LEDs using the application described below. Start the application, switch the red LED on and off as if you were switching the triac. If it works, it's ready to go.

The Android program

For the Android application (version 4.3 or up), our model was the description of the SERIAL program in the July & August 2015 edition of Elektor Magazine [7]. The source code is available on the Elektor Magazine website [8]. The program is already available for download from *Google Play* [9]. The principle is simple. After connecting to the Bluetooth switch (JA_SWITCH), the application sends the character 1 or 0 to switch

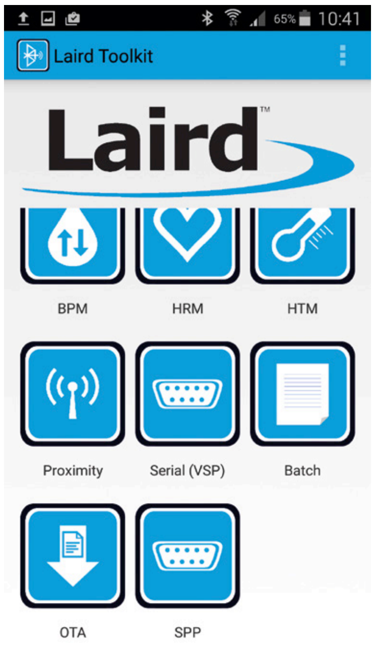


Figure 5. Copy of the Android application screen.

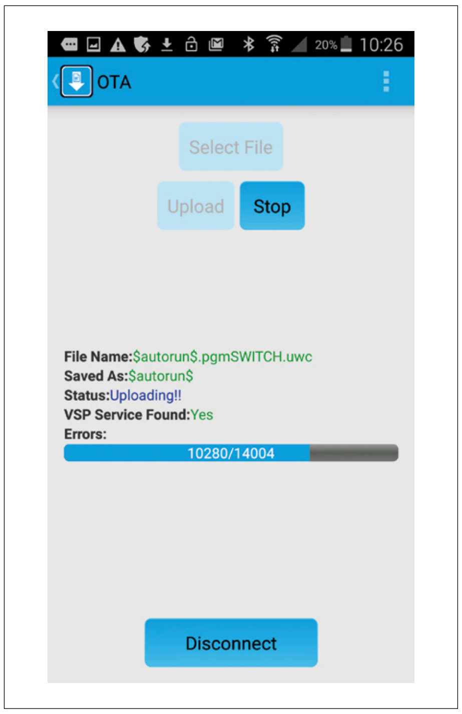
The roles of the Bluetooth Low Energy module

Our e-BoB BL600 carries out the following functions:

- Flashing LED1 (as an indicator of power as well as the absence of a connection to a smartphone)
- Connection via Bluetooth 4.0
- Reception via Bluetooth of commands sent from the telephone
  - 1: switch on the triac, to activate the connector K2
  - 0: Switch off the triac, to remove power from connector K2.
- Send to the telephone the status of connector K2:
  - *Your switch is OFF/ON* (useful to know the state of K2 when you establish a connection)
  - Light LED1 to show that the module is connected
  - Light LED2 to show the status of K2; it is the same pin which controls the triac

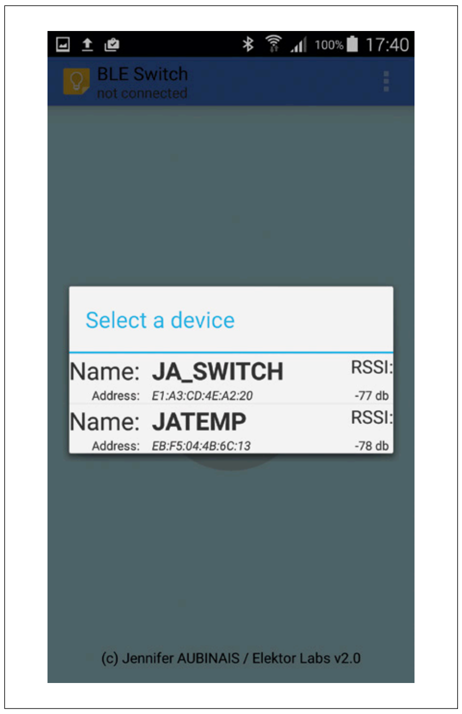
With the colors, it is easy to locate the code corresponding to these tasks in **listing 1**.

- Flashing LED1 (bleu): two timers start alternately: TIMER0 lights LED1, TIMER1 turns it off [3].
- Connection: we use the Handler HandlerBleMsg, renamed MyHandlerBleMsg, [4]. We will concern ourselves with two messages:
  - BLE_EVBLEMSG_CONNECT : (green)
    - After Bluetooth connection, stop the two timers (no matter which is active) by the function TIMERCANCEL
    - Switch on LED1 which was flashing
    - Read the state of the output which controls the triac, to send to the telephone text advising of its state
  - BLE_EVBLEMSG_DISCONNECT : (orange)
    - message sent on disconnection of the module
    - resume flashing LED1 by starting TIMER1
  - reception of a command from the téléphone: (red)
    - in the handler MyHandlerLoop
      - If the code 0x0D (carriage return) is seen in the characters received via Bluetooth (variable text$), we enter into the IF condition.
      - If the received character is “0“
        - GPIO12 output changes to 0
        - Send to the telephone the text “OK, Your switch is OFF“
      - If the received character is “1“
        - GPIO12 output changes to 1
        - Send to the telephone the text “OK, Your switch is ON“



File Name:$autorun$.pgmSWITCH.uwc
Saved As:$autorun$
Status:Uploading!!
VSP Service Found:Yes
Errors:
10280/14004

Figure 6. Copy of the “Upload” screen.



Name: JA_SWITCH RSSI:
Address: E1:A3:CD:4E:A2:20 -77 db
Name: JATEMP RSSI:
Address: EB:F5:04:4B:6C:13 -78 db

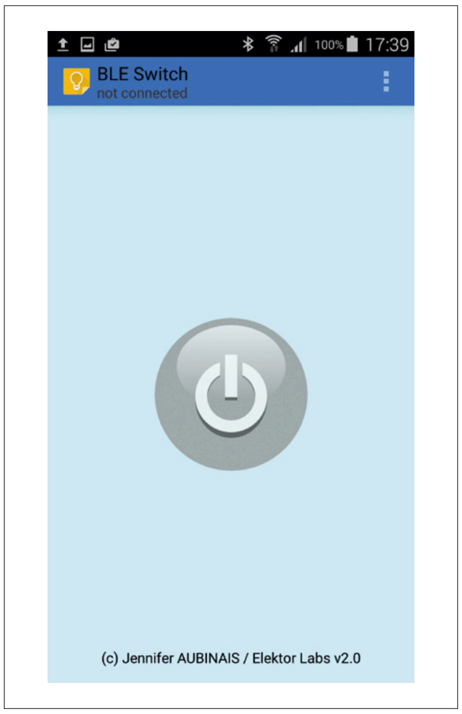
Figure 7. Copy of the “JA_SWITCH” screen.

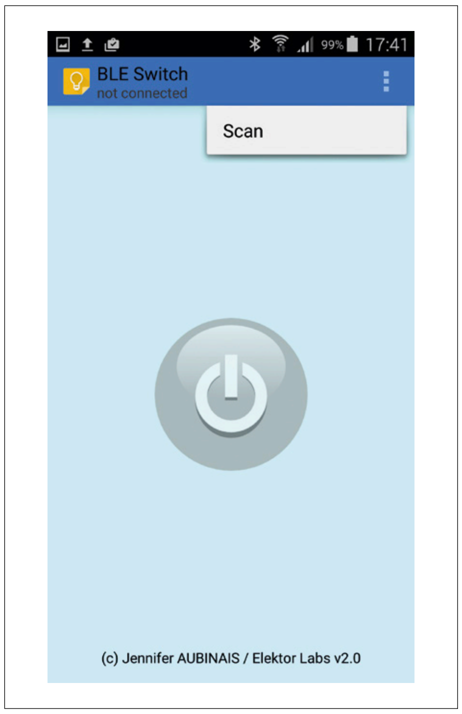
on or off. You can get the same result with the SERIAL application (TOOLKIT from *Laird Technologies* [6]): you connect and send the number 0 or 1 (don’t forget to put in the CR for End of Line).

User Guide

This application, though simple to use, is nonetheless quite complex.
Before using it for the first time, the MAC address of your Bluetooth module is not in your phone’s memory. We must first look for Bluetooth Low Energy devices with the UART service to find our device.

Start the application, tap on the menu, and tap on *Scan*. When the list of devices appears, select JA_SWITCH (Figure 7). The application then shows the status by the color of the button.





The module is connected, the switch is OFF. Tap above to switch ON. The switch is ON (Figure 11). Tap above to switch OFF.
To disconnect from your module, just slide your finger across the screen of your smartphone. You will then find one of the following states:

- The Bluetooth module is not connected and there is no MAC address for your switch in the phone’s memory. You have to use the menu at the top right of the screen.
- The button at the center of the screen is flashing: the Bluetooth module can be connected directly by tapping on the button. This will work every time the application is started.
- Your Bluetooth switch is now ready to be used on your Android smartphone using the large colored button. ⏮

(140115)

Web Links

[1] https://laird-ews-support.desk.com/?b_id=1945

[2] BL600-eBoB Bluetooth Low Energy communication module (1), Elektor March & April 2015, www.elektormagazine.com/140270

[3] Elektor January & February 2016, www.elektormagazine.com/150329

[4] BL600 e-BoB (3): Elektor July & August 2015, www.elektormagazine.com/150129

[4] Elektor July & August 2015 www.elektormagazine.com/magazine/elektor-201507

[5] $autorun$.pgmSWITCH.uwc, www.elektormagazine.comr/140115

[6] https://play.google.com/store/apps/details?id=com.lairdtech.lairdtoolkit&hl=en

[7] BL600 e-BoB (5): Elektor November & December 2015, www.elektormagazine.com/150272

[8] Android Application — source code and APK installation file www.elektormagazine.com/140115

[9] Google PlayStore: https://play.google.com/store/apps/details?id=com.JA.bleswitch

Listing 1

```
'//******************************************************************************
'// Laird Technologies (c) 2013
'// Jennifer AUBINAIS (c) 2015 version 1.1
'//******************************************************************************
'// Definitions
'//******************************************************************************
#define AUTO_STARTUP                             1
'//Set this to 0 to disable all debugging messages
#define ENABLE_DEBUG_PRINTS                      0
#define DEVICENAME                               "JA_SWITCH"
****** code here *****
//==============================================================================
// Led Connect flashes
//==============================================================================
FUNCTION FuncTimer0()
GpioWrite(9,1)
  TIMERSTART(1,100,0)
ENDFUNC 1
FUNCTION FuncTimer1()
  GpioWrite(9,0)
  TIMERSTART(0,1000,0)
ENDFUNC 1
//==============================================================================
// This handler is called when there is a BLE message
//==============================================================================
function MyHandlerLoop()
  DIM n, rc, tempo$, tx$
  DIM value, pos, return$
  tx$ = "0D"
  return$ = StrDehexize$(tx$)
  tempo$ = ""
  n = BleVSpRead(tempo$,20)
  IF (n > 0) THEN
    PRINT "$"
  ENDIF
  text$ = text$ + tempo$
  pos = STRPOS(text$,return$,0)
  IF ( pos >= 0 ) THEN
    DbgMsg("*")
    tx$ = "0"
    pos = STRPOS(text$,tx$,0)
    DbgMsg("OFF")
    Dim Txt$
    IF ( pos >= 0 ) THEN
      GpioWrite(12,0)
      MemSwitch = 0
      Txt$ = "OK, Your switch is OFF\n"
      rc = BleVspWrite(Txt$)
    ENDIF
    tx$ = "1"
    pos = STRPOS(text$,tx$,0)
    DbgMsg("ON")
    IF ( pos >= 0 ) THEN
      GpioWrite(12,1)
      MemSwitch = 1
      Txt$ = "OK, Your switch is ON\n"
      rc = BleVspWrite(Txt$)
    ENDIF
    text$ = ""
  ENDIF
endfunc 1
//==========================================================================
// This handler is called when there is a BLE message
//==========================================================================
function MyHandlerBleMsg(BYVAL nMsgId AS INTEGER, BYVAL nCtx AS INTEGER) as integer
  // Inform libraries
  //ConnMngrOnBleMsg(nMsgId,nCtx)
  AdvMngrOnBleMsg(nMsgId,nCtx)
  select nMsgId
    case BLE_EVBLEMSGID_CONNECT
      DbgMsgVal(" --- Connect : ",nCtx)
      TIMERCANCEL(0)
      TIMERCANCEL(1)
      GpioWrite(9,1)
      Dim Txt$
      if (MemSwitch == 0) then
        Txt$ = "Your switch is OFF\n"
      else
        Txt$ = "Your switch is ON\n"
      endif
      rc = BleVspWrite(Txt$)
      hConnLast = nCtx
      ShowConnParms(nCtx)
    case BLE_EVBLEMSGID_DISCONNECT
      DbgMsgVal(" --- Disconnect : ",nCtx)
      GpioWrite(9,0)
      TIMERSTART(1,10,0)
    ***** code here *****
'//*****************************************************************************
'// Handler definitions
'//*****************************************************************************
//all events have the same handler
OnEvent  EVVSPRX            call MyHandlerLoop //EVVSPRX is thrown when VSP is open and data has arrived
OnEvent  EVUARTRX           call MyHandlerLoop //EVUARTRX  = data has arrived at the UART interface
OnEvent  EVVSPTXEMPTY       call MyHandlerLoop
OnEvent  EVUARTTXEMPTY      call MyHandlerLoop
OnEvent  EVTMR0      call FuncTimer0
OnEvent  EVTMR1      call FuncTimer1
OnEvent  EVBLEMSG           call MyHandlerBleMsg // EVBLEMSG is called when there is a BLE message
OnEvent  EVBLE_ADV_TIMEOUT  call MyBlrAdvTimOut // TimeOut
'//*****************************************************************************
'// Equivalent to main() in C
'//*****************************************************************************
rc = GpioSetFunc(12,2,0)    // pin 12 : OFF / ON
rc = GpioSetFunc(9,2,0)    // pin 9 : Connected / NO connected
dim Adr$
Adr$=""
rc = bleadvertstart(0,Adr$,25,0,0)
TIMERSTART(0,10,0)
'//-----------------------------------------------------------------------------
'// Wait for a synchronous event.
'//-----------------------------------------------------------------------------
WaitEvent
```