Real Time Clock (RTC) *smart*BASIC Sample Application

Application Note ***v1.1***

INTRODUCTION

This guide demonstrates how to load and run a *smart*BASIC Real Time Clock service sample application (**rtcs.erver.sb**) onto the BT900 development board and remotely control it (**rtcc.lient.sb**) using a BT900 or BL620. The sample application advertises an RTC service. The BT900 sends advert packets out with the current time every second and, when connected to a BLE Central role device, the application allows configuration changes to be made to the module. There is also an alarm wake-up mode that can be used to activate remote devices.

RTC OVERVIEW

The module can present a Real Time Clock (RTC) service in the local GATT table consisting of three mandatory characteristics and four custom characteristics.

Mandatory characteristics:

- Current time from which the GATT client can get the current time by reading or getting notifications.
- Local time (DST/Timezone settings)
- Reference time (time source).

The UUIDs of the service and characteristics are shown in Table 1.

Table 1: RTC Service and Characteristic UUIDs

RTC Service or Characteristic	UUID
RTC Service	1805
Current Time Characteristic	2a2b
Local Time Characteristic	2a0f
Reference Time Characteristic	2a14
Update Time Characteristic	0000**2b00**-0000-1000-8000-00805f9b34fb
Alarm Mode Characteristic	0000**2b02**-0000-1000-8000-00805f9b34fb
Alarm Time Characteristic	0000**2b03**-0000-1000-8000-00805f9b34fb
Version Characteristic	0000**2b04**-0000-1000-8000-00805f9b34fb

REQUIREMENTS

- PC running Windows XP or later
- UWTerminal 6.96 or later
- DVK-BT900 Development Kit loaded with v9.1.2.0 firmware or later (see Note)
- **rtcs.erver.sb** *and* **rtcc.lient.sb** *smart*BASIC sample applications, available from https://github.com/LairdCP/BT900-Applications
- USB A to mini B cable
- FTDI Drivers http://www.ftdichip.com/Drivers/VCP.htm (for some versions of Windows)

Note: The latest BT900 firmware, XCompiler, and upgrade documentation is available here: http://www.lairdtech.com/Products/Embedded-Wireless-Solutions/Bluetooth-Radio-Modules/BT900-series#productGroupTabs-2147488080

DEVELOPMENT KIT SETUP

To setup the BT900 development kit, follow these steps:

1. Configure the host BT900 development kit to the following settings (Figure 1):
   - DC/USB power source switch (SW4) – USB
   - 1.8V/3.3V switch (CON17) – 3.3V
   - IIC/Buzzer switch (CON15) – RTC_SDA, RTC_SCL, Buzzer on
   - SIO_3-5 switch (JP2) – Jumper between SIO_20 and ALARM (Pins 1 and 3)
   - Autorun switch (J6) – Jumper on develop mode

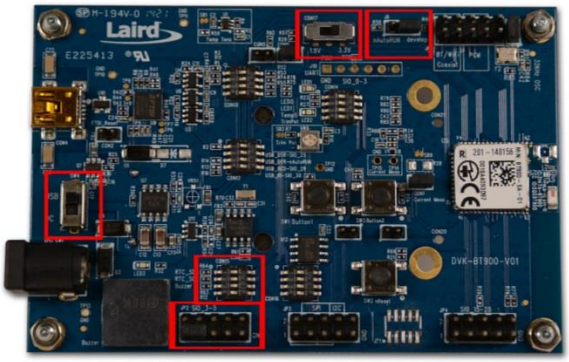

Figure 1: Switch and jumpers position

2. Configure the client BT900 development kit to the following settings (Figure 2) or use a BL620-US packaged USB BL620 adapter (Figure 3):
   - DC/USB power source switch (SW4) – USB
   - 1.8V/3.3V switch (CON17) – 3.3V
   - Autorun switch (J6) – Jumper on develop mode

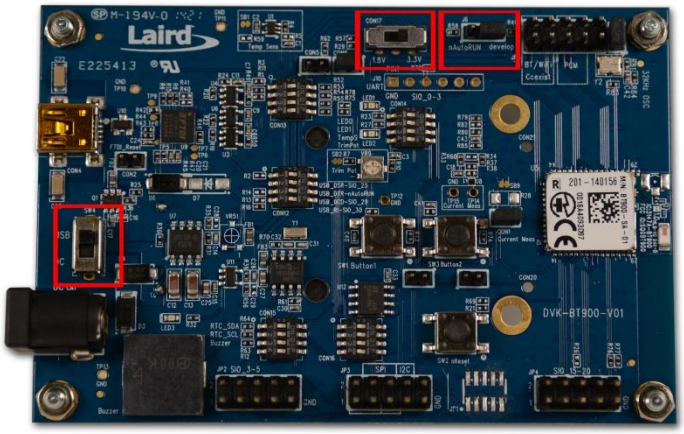

Figure 2: Switch and jumpers position

Figure 3: BL620 USB Module

Note: Refer to the BL620 Quick Start Guide for information on its use. You can access all BL620 documents from the Laird support site: https://laird-ews-support.desk.com/?b_id=5402

3. Connect one end of the mini USB cable to CON4 on the development board and the other end of the cable to your PC, repeat for the second BT900.
4. Follow the on-screen prompts. Depending on your version of Windows, you may need to install the FTDI drivers. When complete, the development board appears in the Windows device manager as a *USB Serial Port*. Note the COM port number.
5. Extract UWTerminal to a selected folder and run **UwTerminal.exe**.
6. Configure the COM port with the port number seen in the device manager with the following settings for a BT900 (Figure 4):
   - Baudrate –115200
   - Stop Bits – 1
   - Data Bits – 8
   - Handshaking – CTS/RTS

   Or the following settings for a BL620 (Figure 5):
   - Baudrate – 9600
   - Stop Bits – 1
   - Data Bits – 8
   - Handshaking – CTS/RTS

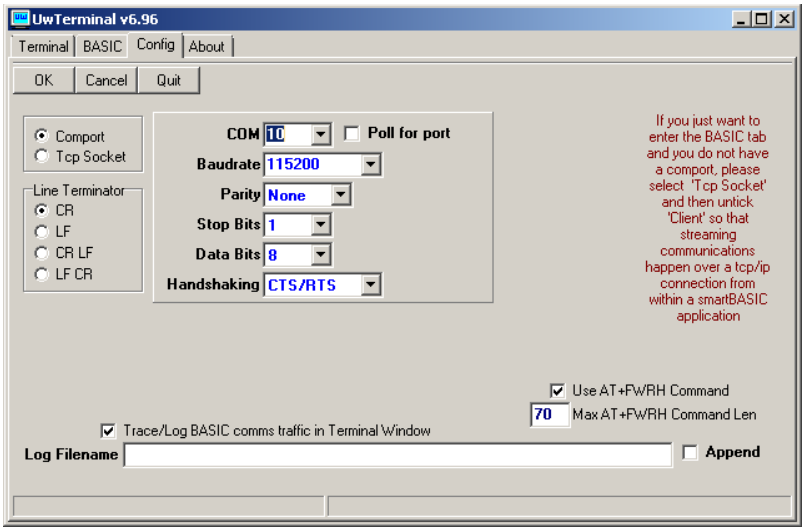

Figure 4: Comms Settings for BT900

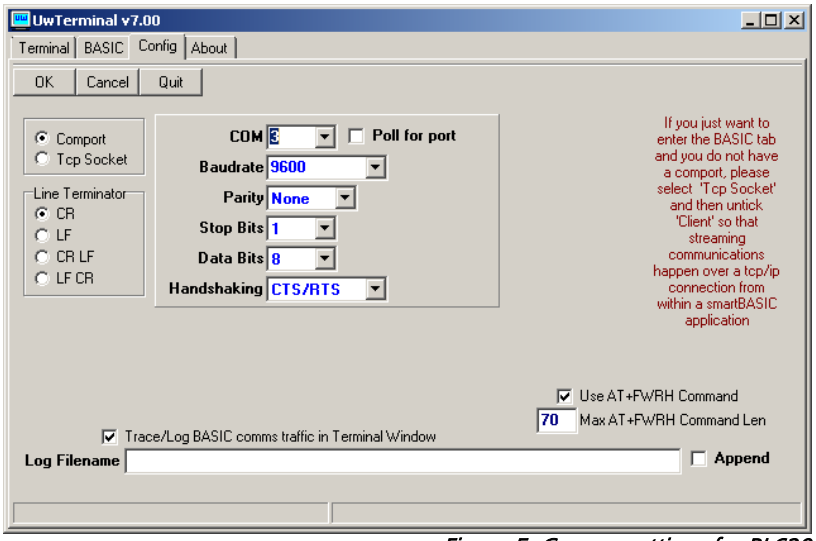

Figure 5: Comms settings for BL620

7. Confirm that you can communicate with the development board by typing *at* and press enter. The module responds with 00 (Figure 6).

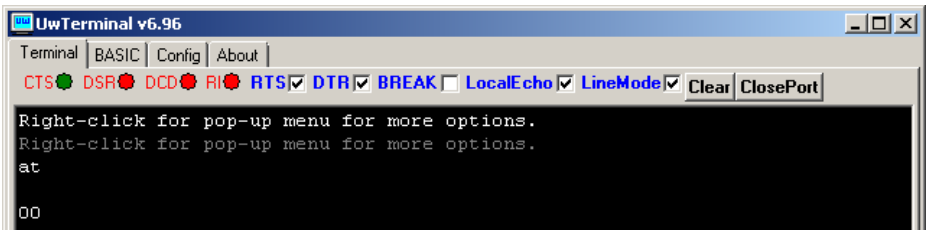

Figure 6: Comms OK

LOADING A *SMART* BASIC APPLICATION

Note: When swapping between profiles on the same device, it may be necessary to clear any existing pairings on the module and other devices. On the BT900, this can be done with the command **at+btd***.

To load a *smart*BASIC application, follow these steps:

1. Ensure the cross compiler is located in the same folder as UWTerminal. Its name is similar to XComp_BT900_CA0D_1DA6, where *CA0D_1DA6* indicates a hash key. Each firmware version requires its corresponding cross compiler with a matching hash key.
2. To compile and load a *smart*BASIC application, right-click in the main UWTerminal window and select **XCompile + Load** (Figure 7).

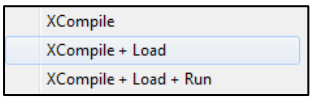

Figure 7: Right-click menu

3. Locate and open the **rtcs.erver.sb** application located on the Laird BT900 GitHub repository (available at https://github.com/LairdCP/BT900-Applications) When the application is successfully compiled and loaded, the console displays **+++ DONE +++** (Figure 8).

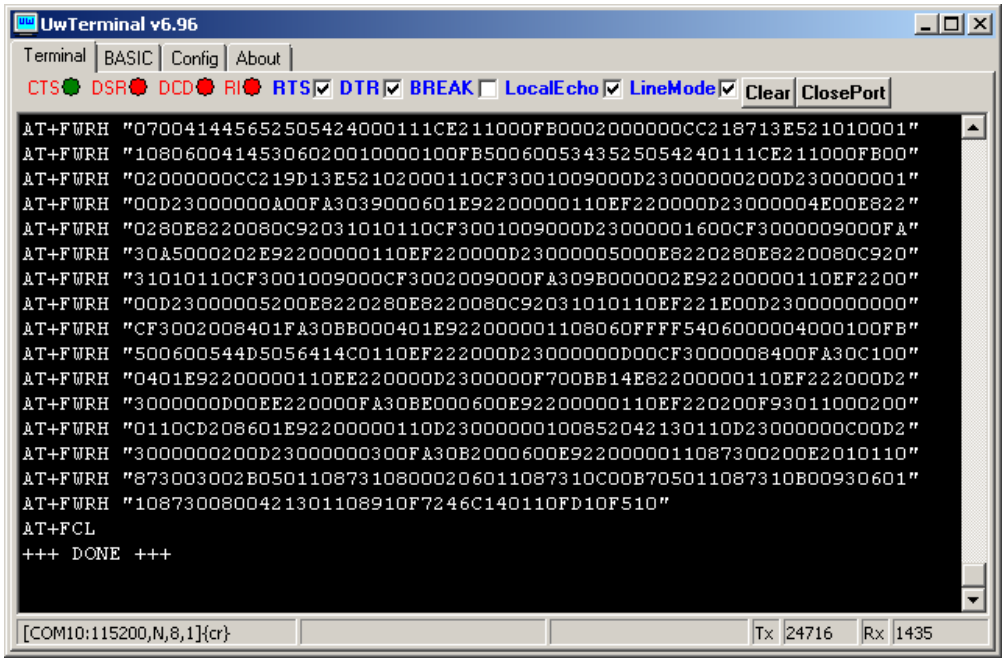

Figure 8: Compiled and loaded

If the correct version of cross compiler is not present, an error displays as shown in Figure 9.

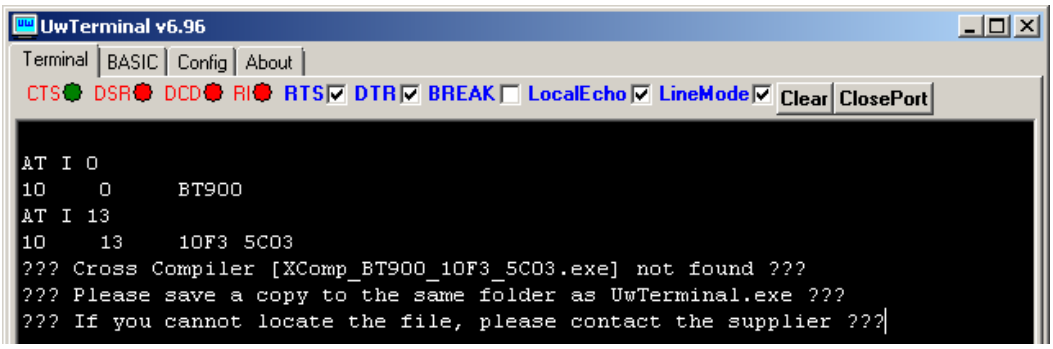

Figure 9: Cross compiler error

4. Locate the correct version and place it in the same folder as UWTerminal.
5. Confirm that **rtcs.erver.sb** is loaded by using the command **at+dir** (Figure 10).

Note: The file extension is truncated from files copied onto the BT900 module. Therefore, when **rtcs.erver.sb** is copied to the device, its name becomes **rtcs**.

LOADING THE CLIENT

To load the RTC client *smart*BASIC application on an additional BT900 DVK or BL620, follow these steps:

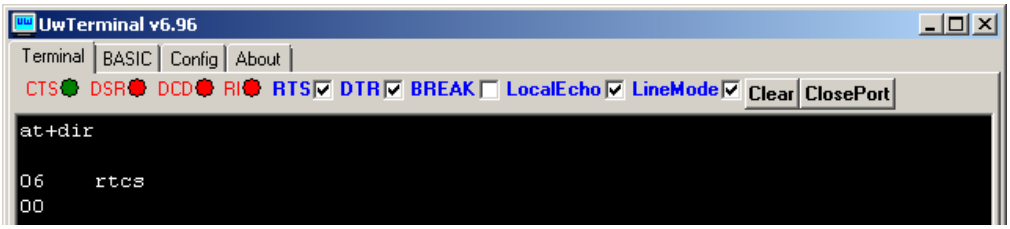

Figure 10: UwTerminal

1. To compile and load a *smart*BASIC application, right-click in the main UWTerminal window and select **XCompile + Load** (Figure 7).
2. Locate and open the **rtcc.lient.sb** application located on the Laird BT900 GitHub repository (available at https://github.com/LairdCP/BT900-Applications). When the application is successfully compiled and loaded, the console displays **+++ DONE +++** (Figure 8).
3. Confirm that **rtcc.lient.sb** is loaded by using the command **at+dir** (Figure 10).

Note: The file extension is truncated from files copied onto the BT900 module. Therefore, when **rtcc.lient.sb** is copied to the device, its name becomes **rtcc**.

STARTING THE APPLICATIONS

1. On the BT900 or BL620 client, type **at+run "rtcc"** to start the client (Figure 11).

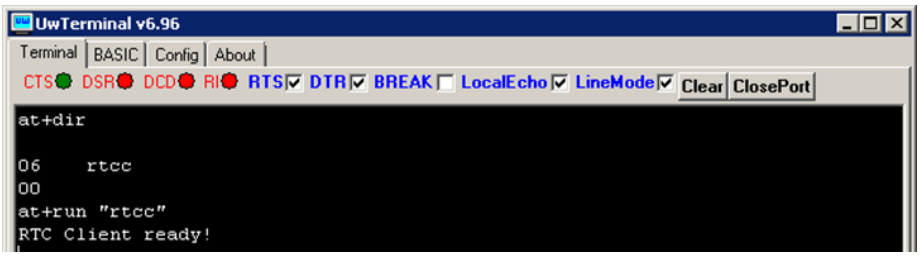

Figure 11: Running the RTC client

2. On the BT900 server, type **at+run "rtcs"** to start the RTC server application (Figure 12).

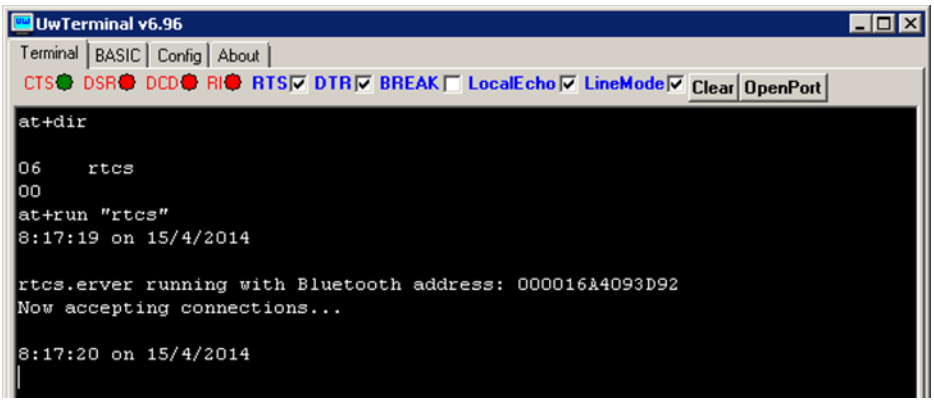

Figure 12: Running the RTC server

RTC CONTROL

A variety of items (such as time updates, alarm enabling/disabling, and a version tag) can be changed on the RTC server which are not part of the core Bluetooth RTC specification.

The following is a step-by-step guide to the major function configurations in the RTC. By default, the RTC client does not connect to an RTC server on startup.

The first function enables scanning for RTC advert packets and displays their data along with the Bluetooth address of the device that is advertising. This function is activated by typing **scan** on the client (Figure 13). To stop scanning for packets, press **Enter/Return**.

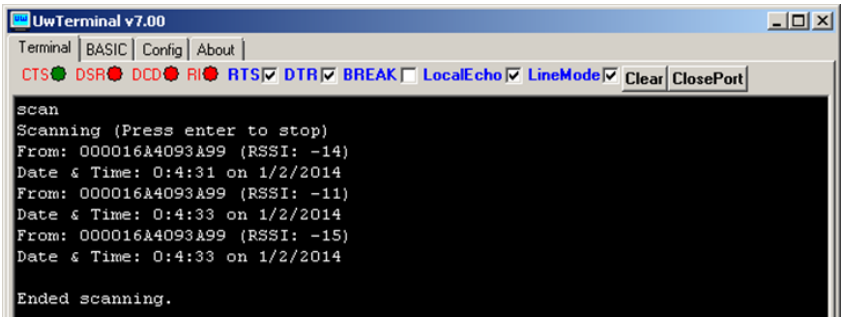

Figure 13: Scanning for RTC advertisement packets

A connection to the RTC server is required to perform additional steps. To establish a connection, configure the Bluetooth address of the server on the client:

- Find the Bluetooth address of the server in one of the following ways:
  - By using **ATI 4** in interactive mode on the device

  - From advert packets (Figure 13)
  - From the startup of the server application (Figure 12)
- The client must set the Bluetooth address of the server to connect by using **set addr <btaddr>** (Figure 14).

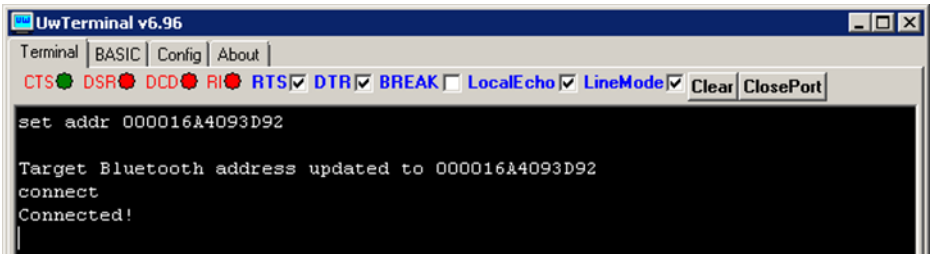

Figure 14: Changing the Bluetooth address of the server

- Establish a connection to the RTC server by typing *connect*. If successful, a connection message displays (Figure 14).

To read the time on the device, send the **show time** command. This also displays the date (Figure 15).

Figure 15: RTC server current time

Timestamps must be entered in the following format in order to be valid: **hh:mm:ss DD/MM/YYYY**. They can be displayed from the application using the **timestamp** command (Figure 16).

Figure 16: Timestamp format on the RTC server

Change the time on the RTC server by using the **set time <timestamp>** command. This takes effect immediately on the host. Figure 17 displays an example using **set time 00:01:02 01/02/2014**.

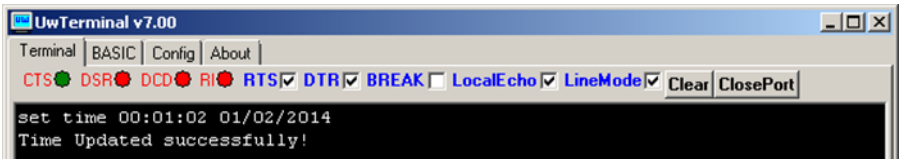

Figure 17: Updating the time of the client-side RTC server

The change can be seen in Figure 18.

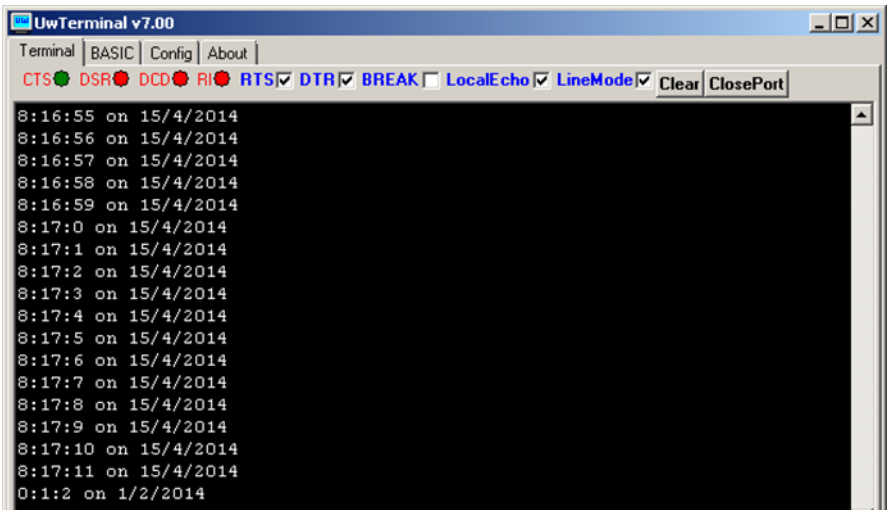

Figure 18: Updating the time of the server-side RTC server

The sample application also includes an alarm service. An alarm can be set which makes the BT900 sound its buzzer for a short duration before returning to normal operation.

The first step to configuring the alarm is to set up the alarm time. To do this, use **set alarm time <timestamp>** with the same timestamp format as was discussed earlier (Figure 16). Enable the alarm using **set alarm mode <on/off> <match>** (match is one of seconds, minutes, hours, date). The match field sets what registers in the RTC set off the alarm. To disable the alarm, use **set alarm mode off**. Figure 19 displays a sample alarm setup.

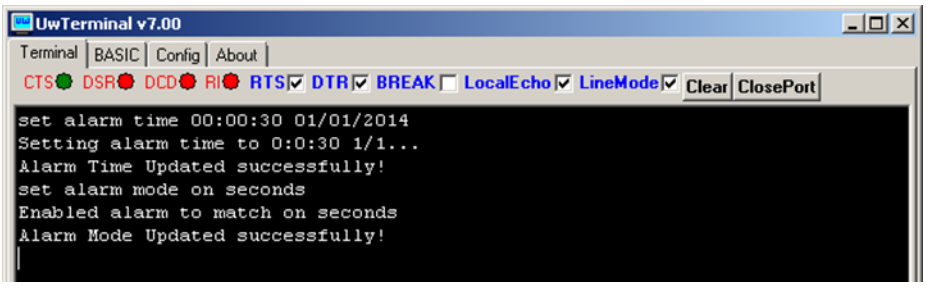

Figure 19: Changing and enabling the alarm of the RTC server

Similarly to the **show time** command, use **show alarm mode** and **show alarm time** to display the various settings of the remote time server (Figure 20).

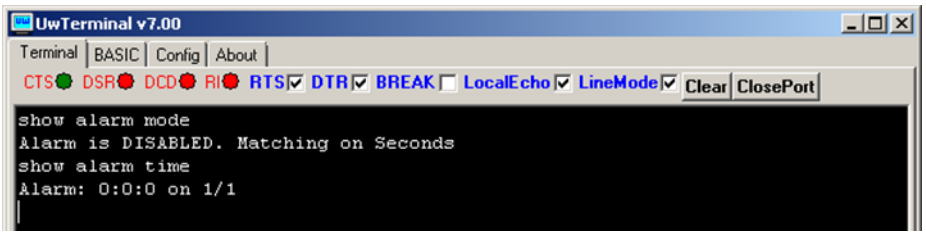

Figure 20: Changing and enabling the alarm of the RTC server

To disconnect from the host device, type **disconnect** (Figure 21). When finished with the application, type **exit** to return to interactive mode.

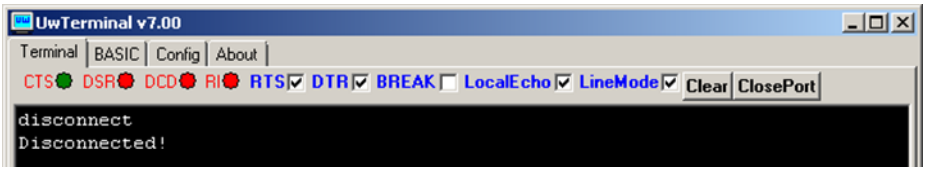

Figure 21: Showing the remote alarm time and mode of the RTC server

DEVELOPMENT KIT SETUP TO AUTORUN RTCS

To have the RTC server start on the target BT900 device automatically at power-up, it must be renamed as an autorun script. To do this, us the **at+ren "rtcs" "$autorun$"** command. Using the **at+dir** command shows that the file is now called $autorun$ (Figure 22); it automatically executes when the BT900 starts up or resets and the J6 jumper is connected to nAutorun.

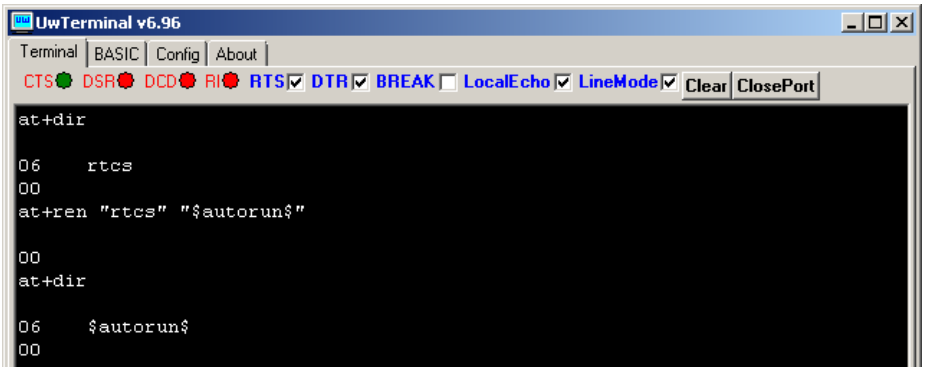

Figure 22: Renaming rtcs as an autorun application to allow automatic running

For more information on autorun, refer to the BT900 Extensions user guide available from the BT900 product pages on the Laird website.

REFERENCES

For more information on the Real Time Clock service as well as any *smart*BASIC commands used in this application note, refer to the following documents:

- Real Time Clock Server sample application rtcs.erver.sb and Real Time Clock Client sample application rtcc.lient.sb available from https://github.com/LairdCP/BT900-Applications
- BT900 smartBASIC extension manual (for FW v9.1.2.0)

All BT900-related documents can be accessed from the BT900 product pages on the Laird website.