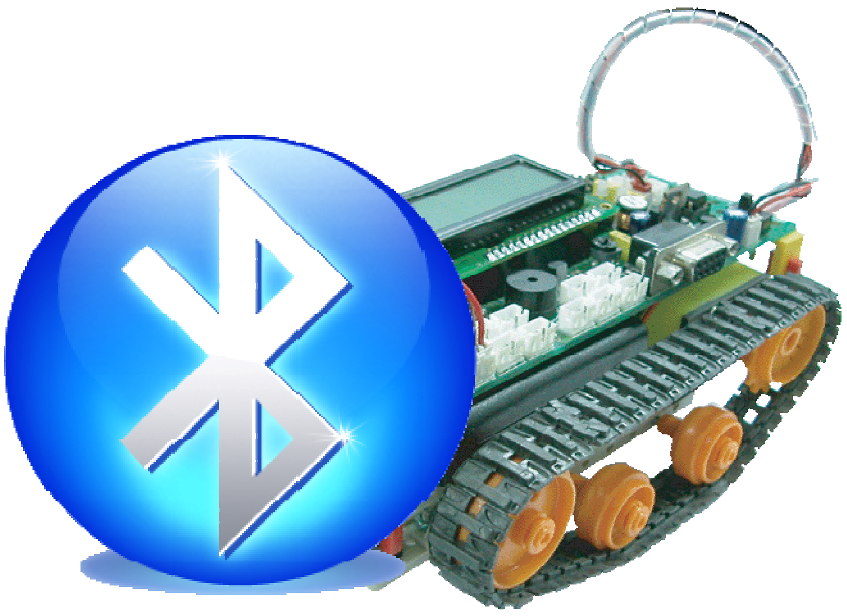
ALIATRON

Supervisor: Tayeb Habib, B.Sc. (Hons.)

Bluemote V.1.0



By:
Pedro Emanuel Antunes Vicente

Report Submitted: 3rd of Septembre of 2010

Acknowledgments:

I would like to thank engineer Tayeb Habib, for all his supervision, patience, help, time and everything he taught me, among other things that made this project successful.

I would also like to thank Aliatron and its employees, for their receptivity and possibility of achieving this traineeship, their companionship and for believing in me, giving me support and this opportunity.

I’m grateful to Professor Carlos Amaro, who first gave the incentive to follow this area and second for believing in my abilities; but, mostly for making possible the contact with Aliatron.

Finally I thank to my family for the education, values, support and comprehension.

To my Girlfriend, thank you for the patience and also for the unconditional support.

Index:

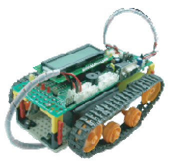
This project was realized during my traneeship at Aliatron, in July, August and September.

Objectives:

- Explore the serial communications
- Transmission/reception via Bluetooth
- Control system with 8051 and use of Bluetooth communication
- Control via web the created system

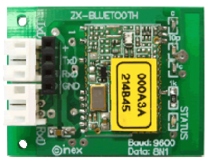
Apparatus and Instrumentation:

- Kit Robot 51, Inex



- Bluetooth micro Adapter Class I, Sweex

- Module Zx-bluetooth, Inex



- BlueSoleil 5.4 (software)
- Termite 2.4 (software)
- SerialTerminal (software)
- XAMPP (software)
- RIDE 51 V6.1, Raissonance
- FLIP V2.2, Atmel

Theoretical Introduction:

The Serial communication was created with the purpose of connecting a teletype to a modem, however it was standardized later on by Electronics Industries Association (EIA) to multiple devices. Apart its most common designation (RS-232 protocol), there is still the EIA RS-232C designation, due to this association.

Later, with the inclusion of RS-232 Protocol in the computers, produced by IBM, it became ubiquitous in all devices and it turned out to be one of the most used communications in the world. In this case, the exchange of information is done between a DTE (Data Terminal Equipment), a computer, and a DCE (Data Communication Equipment), for example, a modem. The two are usually connected through a D shaped cable with 9 pins.

The serial communication allows several velocities/speed of transmission and, currently, there are devices who can even communicate at 115.200bps (initially they could only support till 20.000bps). The bits sent are codified in a system "start-stop-asynchronous", usually with a bit of initiation,7/8 data bits, one stop bit and one bit for parity. The RS-232 came to be replaced by USB in local communications, due to being slower and more complex, yet it continues to be mainly used in industry and in sales posts, principally in tills.

Also note that there is also a synchronous serial communication, other than as described so far. This type of communication implies the sharing of the clock between the two devices to ensure they are in tune. The serial communication, present in many devices with different faces and functions remains one of the most usual ways of communication nowadays.

Bluetooth is one of the most known means of communication. It is present in all types of equipment, from Desktops, Laptops, phones, PDAs, etc. The aforementioned communication is done via a short-range radio frequency and it is necessary to consider the existence of three distinct Bluetooth classes. That distinction is made according to the maximum range and power that the equipments have. The class I is the best, because it can reach a range up to, approximately, 100 meters.

A Bluetooth master can connect with up to 7 devices simultaneously, forming, thus, an ad-hoc network of eight elements, which we call *piconet*. Because it has multiple protocols, the Bluetooth, allows to communicate in many different ways and can have different forms (in this project, the Bluetooth, was used as a virtual serial port).

Using the same device, the Bluetooth can send and receive information, which compared to its “rival” infra-red is a plus. It also allows other type of connections, particularly, TCP/TP, WAP, OBEX, etc protocols (however, the protocols mentioned, were not used in this project).

In this project we used the microcontroller T89C51AC2 from Atmel, based on the architecture of the famous microcontrollers 8051. The 8051 is designated as the mother of all micro-controllers; still, it is necessary to highlight the following creations of the 8051: in 1981, by Intel Corporations, allowing the usage of C language and Assembly.

For the robot used in this experiment, the T89C51AC2 scheme comes in the annex, as well as its respective connections. The 8051 have a set of serial ports, TxD and RxD, which allows serial communication, both to program and to send information during the actions made by the robot. The 8051 still hold, namely, two timers, 32 pins of input/output and a RAM memory of 128 bytes, among other features. One of the gains of these microcontrollers are the Interrupt ports, in our case the P3.2 and P3.3. The 8051 better manage interrupts than any other microcontroller, making them among the most commonly used at industrial level.

Experimental Procedure:

Based on the experiment previously performed with the robot 51, dated August 2010, the one which referred to the remote control (infra-red) available in the kit, we started our project.

Having as base the source code used in that activity, the remoteb.h library was slightly changed to the values described below, due to the fact that the previous equipment operated in a different speed transmission (1200bps). The module Zx-Bluetooth works at a baud rate of 9600bps in an 8N1 format (eight data bits, without parity and with one stop bit), and because of that, the loaded program in the microcontroller should have a lower delay_band in order to wait the required time for each bit sent by the Bluetooth, in other words, the “delay” function should be eight times faster.

In order to read correctly the received information by the microcontroller’s P3.2 port, the remoteb.h library has an interrupt, this interrupt allows to receive, at any time of the program, the information that the Bluetooth sends to that same port. When the microcontroller detects an input signal at P3.2 port it will stop the main program and execute the interrupt, which in this case is the function "void service_ex0 (void)". After finishing this function, the program will run normally where it was stopped. It is in the interrupt that all information is processed, and is therefore the base that makes serial communication possible, because of that the speed in which the microcontroller is working should be born in mind, predominantly the crystal velocity and the operating timer. In our case 8051 was in X2 mode, whose timer is two times faster. The usage of the serial ports available in the microcontroller, in this project, could be a practicable hypothesis if those same ports weren’t being used for programming. It could be an advantage to use these ports, since they are prepared specifically for this type of communication, having all the necessary protocols and is easily set the baudrate used.

For the communications between devices proves to be effective, it is necessary to install correctly the Bluetooth adapter and the Bluesoleil software. This software will create virtual serial ports for each new device found, including a port to receive data and another to send it. Therefore, the software will detect the module, already installed in the robot, giving it the name of SPP (Serial Port Profile), since the computer sees it as a serial device. And, after choosing the serial connection between devices, we are ready to send information to the microcontroller. It is essential to highlight that the robot should have been previously loaded with the act19_bl.c program with the respective libraries. Of these, only the library remoteb.h is the subject of study in this project, the remainder may be found in more detail in the online manual (of robot-51) available in Webgraphy.

In the current project, we used the software Ride of Raissonance to compile the program, creating a HEX, which we, later, loaded to the microcontroller through the Atmel's software FLIP. If the connection between the virtual serial port, created by BlueSoleil and the Zx-Bluetooth module, is available we can finally send the information we want, which may be sent in various ways. In this project specifically it was done through the software Termite and SerialTerminal, each one of these programs has its advantages. If on one hand, for the SerialTerminal it's enough to enter one character in the keyboard, that this is automatically sent to the microcontroller; on the other, the Termite allows to send more than one character at a time, having, also, a more pleasant graphic environment.

I must stress that in all communications, it is crucial that the virtual port is in a 9600bps baud rate, only like this they will be possible.

The next step would be to create an automatism, so that through just a simple click or ENTER, the information required could be sent, to the microcontroller. This automatism can be created by a script in vbs (Visual Basic Script), using the already mentioned SerialTerminal to initiate the communication with the serial port, where the Bluetooth is connected, as well as the commands. O script will then be used to send the characters, as if it was a virtual keyboard. This process was also described, in the forum: *A generation of new technology*, by the user Bill Blaton (link available in the Webgraphy).

The commands that we used to drive the robot are as follow: the letters to send would be d, to move forward; a, to move backwards; b, to turn left; c, to turn right; and s, to stop. The use of this script, proved an intelligent and agreeable way of "attack" the problem, since we didn't needed to choose the use of any additional software, or some specific algorithm. This one uses the command line to start the SerialTerminal program, afterwards, it sends the sought letters, and finally shuts the program down automatically. It would be also viable to reformulate the SerialTerminal's source code (available in C language) so that it would send the requested information, however due to lack of time this resource was not explored. Although we tried, as in the picaxe control whose report is available at http://redacacia.wordpress.com/2010/08/05/internet-control-of-a-picaxe , to use the Kermit, its usage was not possible, since it could not communicate with the microcontroller or it couldn't interpret the received information. In our metaphorical understanding, this problem resembles a relay race, in which one of the runners takes the baton in his hand and has to give it to the next athlete, but if they are not sufficiently synchronized, the baton will eventually be lost at the time of its transition, getting the baton (information) lost somewhere on the track. The same will happen in the communication via Kermit. We believe that this one sends the "baton" to the microcontroller, as it is prepared to do, but the microcontroller doesn't receive it, and when it tries to go get it the information is no longer there, and is no longer available.

To control through the Internet a server was created, we preferred XAMPP, only then we would have a free web server on Windows. After the installation of XAMPP, .php scripts were created (present in the annex) to control the .vbs scripts, that had the commands to control the robot obtaining, therefore, a breakthrough regarding the bluemote function in the internet. In this case, the bluemote.php page interprets the information clicked by the user, opening and sending a piece of information to processing.php, this one will call the script responsible for that particular action. But this can be proven in detail on the attached files, for each one of the five achievable actions. After the process had been locally tested was necessary to set the IP on our computer so that we could place it on the internet. Because it already existed, in the enterprise, a working server, we set our server as a principal's proxy. The web server is based on apache and the httpd.conf file should be changed to point to proxy, however the strategy used falls outside the scope of this Project. The proxy's creation and the XAMPP's tutorial can be found in the Webgraphy.

Conclusion and Recommendations:

With this project we can conclude that is possible not only to control a robot/microcontroller via Bluetooth but also through the internet. We were able to successfully establish communication between the computer and the robot and consistently solve the issue of the baud rate used by both. The automatism, necessary for the posterior implementation of the system control on the internet, was equality achieved, still its accomplishment was only possible after we found the SerialTerminal, which proved to be an asset for the web application. The discovery of this software saved us time because we didn't need to research the software Kermit, since it overcomes the issue announced earlier in the experimental procedure. The use of Kermit could have been a workable possibility but it needed to be explored with time.

Having as basis this project, there are countless applications that can be created: from projects in the area of home automation to projects for the industrial area. Applications that use the phone to send information to the microcontroller, (using specific software), it would be similarly interesting, however the applications weren't applied due to lack of time, this would resolve the issue of mobility, since it would be possible to control the robot in any environment, even without the presence of a computer.

Another appealing project would be to receive information from the robot. This required the use of Bluetooth's transfer port (TxD) that in this project was inactive (as it can be seen in the annex), therefore, we could use all the potentiality of the Zx-Bluetooth module. Probably, in this scenario it would be necessary another library in order to send information from the robot, at a 9600bps baud rate. This information from the robot, referred above, could be, for example, a feedback from the instructions sent to the robot, or the images captured by a video camera. .

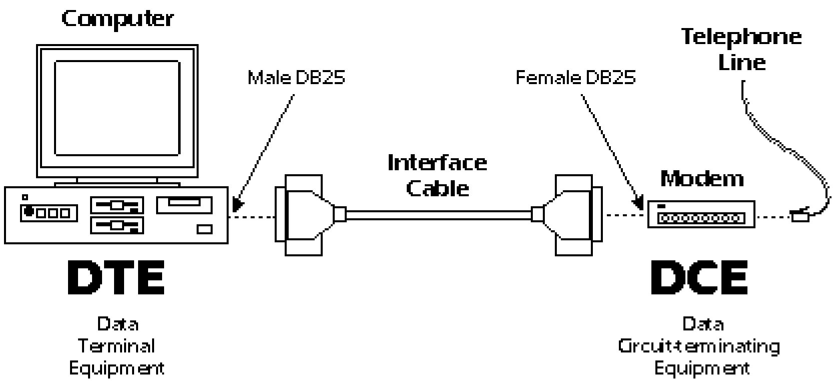
Our choice, in this project, was Bluetooth, but it would also be doable to use WiFi. This choice is justified, because Bluetooth ensure greater safety to the system, keeping the connection always available. On the other hand, the WiFi would connect to another device, if it has a bigger power in the function zone of our microcontroller.

That said, I believe that this project was quite interesting, because it is not static, there is, therefore, the opportunity to continually expand much more.

Everything that was described above can be viewed at:

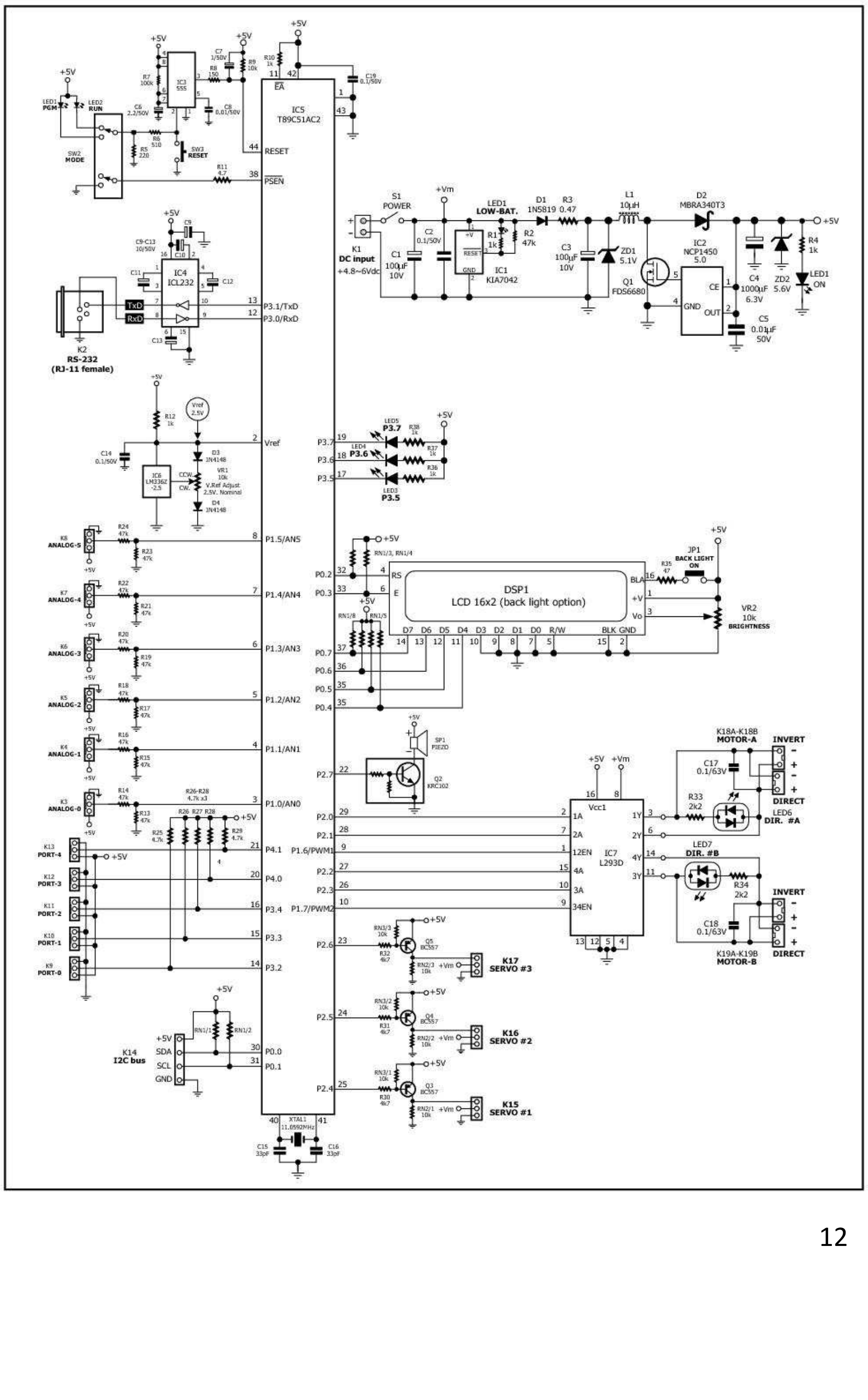
http://www.youtube.com/watch?v=ZGFYZh66vdA

Annex



Annex.1 Example of serial comunication

Annex.2 D form cable



Annex.3 Wiring diagram of T89C51AC2 on robot 51

```
/*----------------------------------------------------------------------------*/
// Program: Remote control robot
// Description : Robot controlled by Bluetooth control
//Robot receive command  from bluetooth by serial communication at baudrate 9600 bps
// Filename: Bluemote.c
// C compiler: RIDE 51 V6.1
/*----------------------------------------------------------------------------*/
#include <C51ac2.h>             // Header include register of T89C51AC2
#include <delay_robo51.h>       // Module function delay process
#include <dc_motor.h>           // Module function drive DC motor
#include <remoteb.h>            // Module function for remote
#include <sound_robo51.h>       // Module function drive sound
#include <lcd_robo51.h>         // Module function LCD
#define pow 200                 // Define constant for power drive DC motor


void run_fd(int time_spin) {
        motor_fd(2,pow);                // Motor channel 2 forward
        motor_fd(1,pow);                // Motor channel 1 forward
        delay_ms(time_spin);            // Delay time for robot drive forward
}
void run_bk(int time_spin) {
        motor_bk(2,pow);                // Motor channel 2 backward
        motor_bk(1,pow);                // Motor channel 1 backward
        delay_ms(time_spin);            // Delay time for robot drive backward
}
void turn_left(int time_spin) {
        motor_fd(2,pow);                // Motor channel 2 forward
        motor_bk(1,pow);                // Motor channel 1 backward
        delay_ms(time_spin);            // Delay time for robot spin turn left
}
```

```
void turn_right(int time_spin) {
            motor_bk(2,pow);                // Motor channel 2 backward
            motor_fd(1,pow);                // Motor channel 1 forward
            delay_ms(time_spin);            // Delay time for robot spin turn right
}
void main() {
            beep();                         // Sound beep 1 time
            remote_init();                  // Initial remote
            lcd_init();                     // Initial Lcd
            while(1)  {                     // Infinite loop
                        switch(get_remote()) {      // Check command for receive
                        case 'a'  :run_bk(100);     // Drive robot backward when receive command "a"
                                    clear_remote(); // Clear command
                                    break;          // Out from case
                        case 'b' : turn_right(100);            // Turn right when receive command "b"
                                    clear_remote();            // Clear command
                                    break;                     // Out from case
                        case 'c' : turn_left(100);             // Turn right when receive command "c"
                                    clear_remote();            // Clear command
                                    break;                     // Out from case
                        case 'd' : run_fd(100);                // Turn right when receive command "d"
                                    clear_remote();            // Clear command
                                    break;                     // Out from case
                        case 's':  motor_stop(all);            // robot stop
                                    clear_remote();            // Clear command
                                    break;                     // Out from case
                        default:  break;                       // Out from case
                        }
            }
}
```

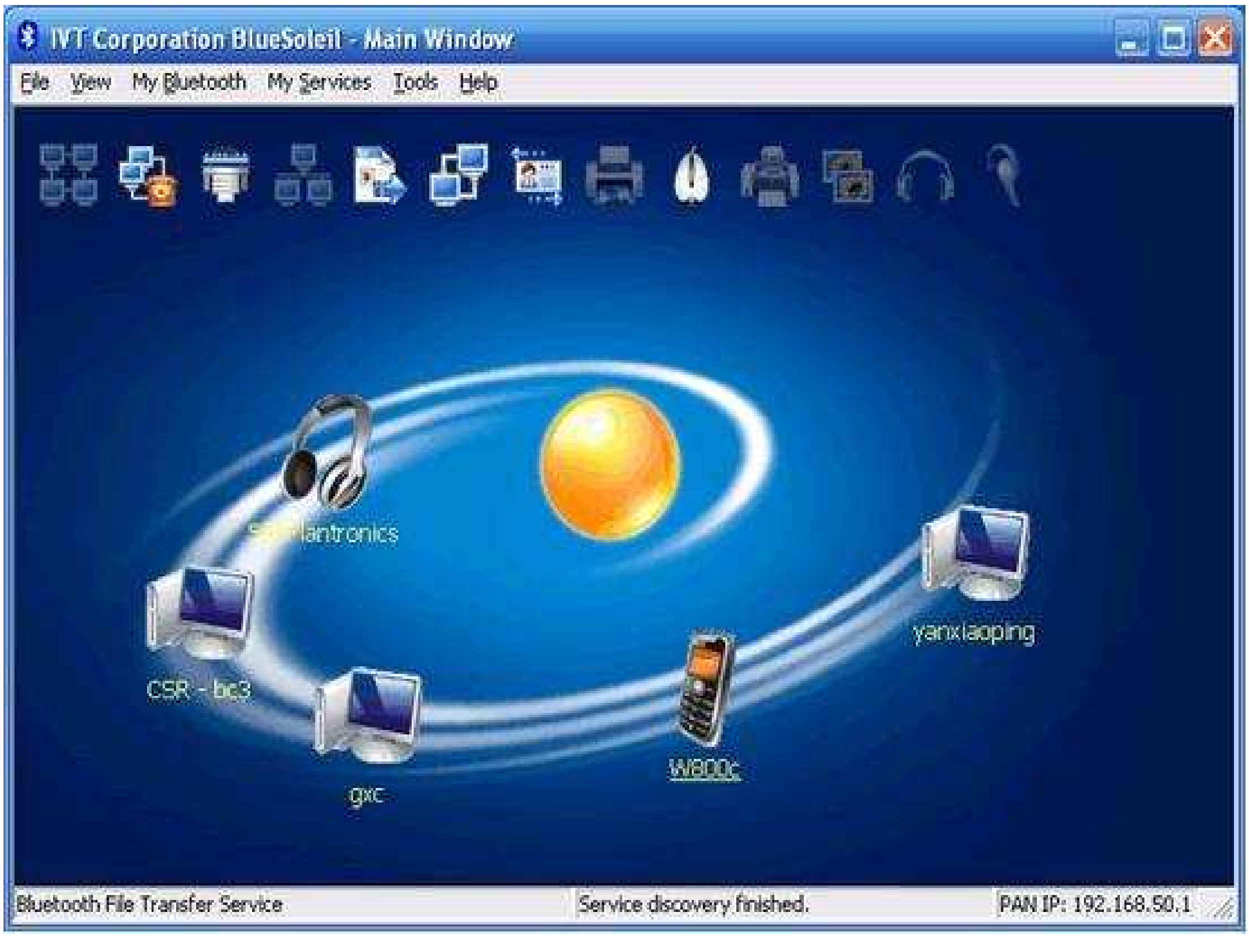
Annex.4 “Bluemote.c” file

```
/*----------------------------------------------------------------------------*/
// Program          : Decode command from Bluetooth
// Description      : receive command from Bluetooth to control robot
// Filename         : remoteb.h
// C compiler       : RIDE 51 V6.1
// Adapt to bluetooth with baudrate 9600 bps by Pedro Vicente
/*----------------------------------------------------------------------------*/
sbit irm_data = P3^2;       // Define port input remote sensor
unsigned char ir_command=0;       // Define for buffer command from Easy remote
/****************************************************************************/
/****************** Function Delay for baudrate 9600 ************************/
/****************************************************************************/
void delay_baud9600()     {                      // Delay baudrate 9600 bps
            unsigned int x,a;                    // Keep for counter loop
            for(x=0;x<1;x++) {
                        for(a=0;a<91;a++);       // Loop for delay 104 us  // inicially at 833 us
            }
}
/****************************************************************************/
/****************** Function Delay for baudrate 9600(for start bit) **********/
/****************************************************************************/
void start_9600()  {                             // Delay baudrate 9600 while start bit
            unsigned int x,a;                    // Keep for counter loop
            for(x=0;x<1;x++) {
                        for(a=0;a<46;a++);       // Loop for delay 53us //inicially at 416 us
            }
}
```

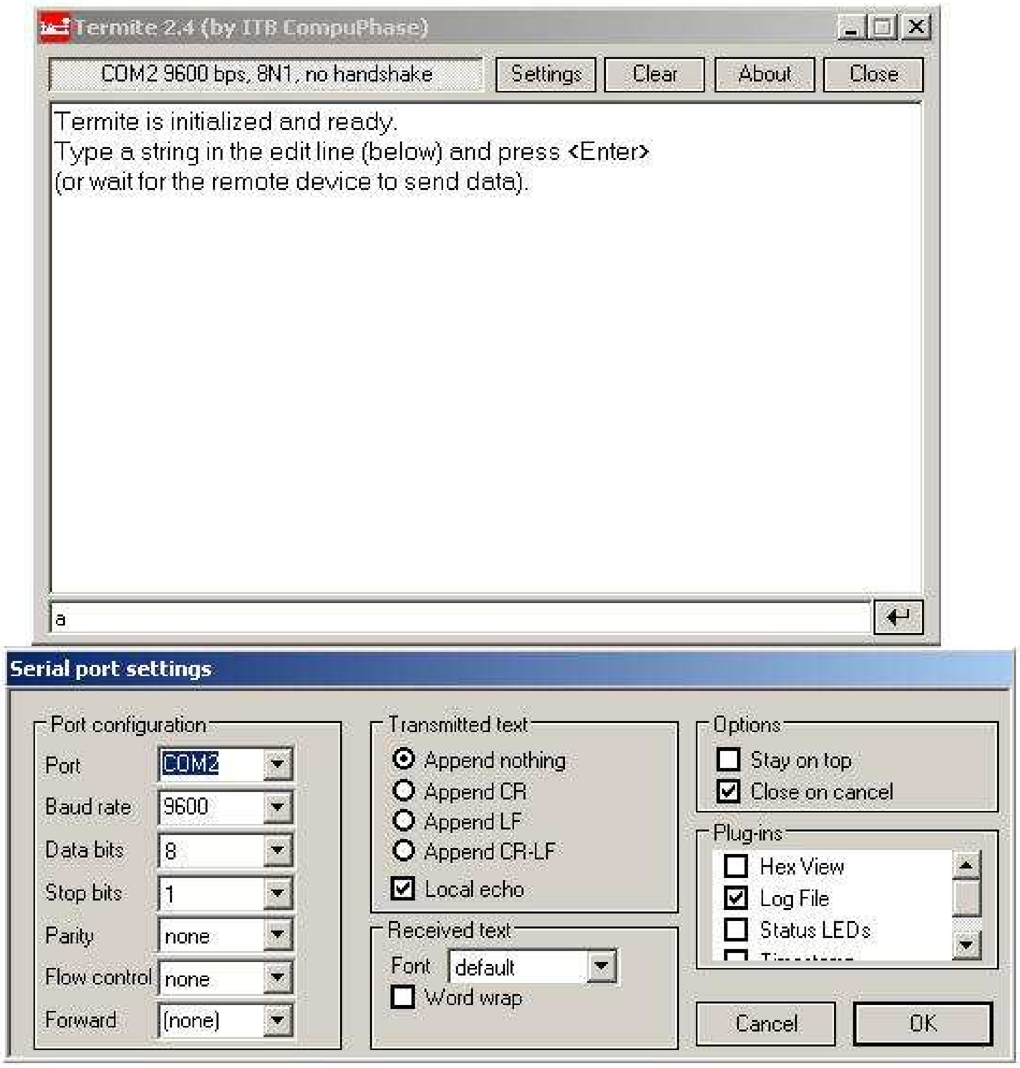
```
/*****************************************************************************/
/************* Service interrupt routine for receive command ******************/
/*****************************************************************************/
void service_ex0(void) interrupt 0 {
        unsigned char i;   // Define for counter loop
        if(irm_data==0) {   // Check start bit true?
                start_9600();                   // Delay for start bit
                for(i=0;i<8;i++)  {                     // For loop count 8 time(for receive data 8 bit)
                        delay_baud9600();                       // Delay for data 1 bit
                        ir_command = ir_command>>1;                 // Shift data bit to right 1 time
                        if(irm_data)                                // If data bit = high
                        ir_command = ir_command | 0x80;             // Config data bit = "1"
                }
                delay_baud9600();                                   // Delay for stop bit
        }
}
/*****************************************************************************/
/************** Function read Command from Bluetooth ***********************/
/*****************************************************************************/
unsigned char get_remote(void) {
        return(ir_command);                     // Return command
}
/*****************************************************************************/
/*********************** Function clear command ******************************/
/*****************************************************************************/
void clear_remote() {
        ir_command = 0;         // Clear command
}
```

```
/*******************************************************************************/
/********************** Function initial for Bluetooth *******************/
/*******************************************************************************/
void remote_init(void) {
        irm_data = 1;    // Configuration input P3.2
        EX0 = 1;         // Enable External interrupt0
        IT0 = 1;          // Detect falling edge
        EA = 1;          // Enable interrupt all
}
```

Annex.5 “remoteb.h” file



Annex. 6 BlueSoleil’s Interface


Termite 2.4 (by ITB CompuPhase)

COM2 9600 bps, 8N1, no handshake

Termite is initialized and ready.
Type a string in the edit line (below) and press <Enter>
(or wait for the remote device to send data).

a

Serial port settings

Port configuration
Port: COM2
Baud rate: 9600
Data bits: 8
Stop bits: 1
Parity: none
Flow control: none
Forward: (none)

Transmitted text
[✓ Append nothing]
Append CR
Append LF
Append CR-LF
[✓ Local echo]

Received text
Font: default
Word wrap

Options
Stay on top
[✓ Close on cancel]

Plug-ins
Hex View
[✓ Log File]
Status LEDs

Annex.7 Termite 2.4's Interface

(COM2 – port connect with the robot)

```
/*--------------------------------------------------------------------------*/

// Program: Send letter to serial port

// Description: Send “d” to serial port to move foward

// Filename: fw.vbs

/*--------------------------------------------------------------------------*/

Set oShell = WScript.CreateObject("WScript.Shell")

oShell.Run("C:\r51\serialterm com2 9600")

WScript.Sleep(500) 'wait (0.5 seconds (# milliseconds))

oShell.Sendkeys "d"

oShell.Sendkeys"{ESC}"
```

Annex.8 script fw.vbs

```
/*------------------------------------------------------------------------*/

// Program: Send letter to serial port

// Description: Send “a” to serial port to move backward

// Filename: bk.vbs

/*------------------------------------------------------------------------*/

Set oShell = WScript.CreateObject("WScript.Shell")

oShell.Run ("C:\r51\serialterm com2 9600")

WScript.Sleep(500) 'wait (0.5 seconds (# milliseconds))

oShell.Sendkeys "a"

oShell.Sendkeys"{ESC}"
```

Annex. 9 Script bk.vbs

```
/*------------------------------------------------------------------------*/

// Program: Send letter to serial port

// Description: Send “b” to serial port to turn rigth

// Filename: rg.vbs

/*------------------------------------------------------------------------*/

Set oShell = WScript.CreateObject("WScript.Shell")

oShell.Run ("C:\r51\serialterm com2 9600")

WScript.Sleep(500) 'wait (0.5 seconds (# milliseconds))

oShell.Sendkeys "b"

oShell.Sendkeys"{ESC}"
```

Annex. 10 Script rg.vbs

```
/*------------------------------------------------------------------------*/

// Program: Send letter to serial port

// Description: Send “c” to serial port to turn left

// Filename: lf.vbs

/*------------------------------------------------------------------------*/

Set oShell = WScript.CreateObject("WScript.Shell")

oShell.Run ("C:\r51\serialterm com2 9600")

WScript.Sleep(500) 'wait (0.5 seconds (# milliseconds))

oShell.Sendkeys "c"

oShell.Sendkeys"{ESC}"
```

Annex. 11 Script lf.vbs

```
/*------------------------------------------------------------------------*/

// Program: Send letter to serial port

// Description: Send “s” to serial port to stop

// Filename: st.vbs

/*------------------------------------------------------------------------*/

Set oShell = WScript.CreateObject("WScript.Shell")

oShell.Run ("C:\r51\serialterm com2 9600")

WScript.Sleep(500) 'wait (0.5 seconds (# milliseconds))

oShell.Sendkeys "s"

oShell.Sendkeys"{ESC}"
```

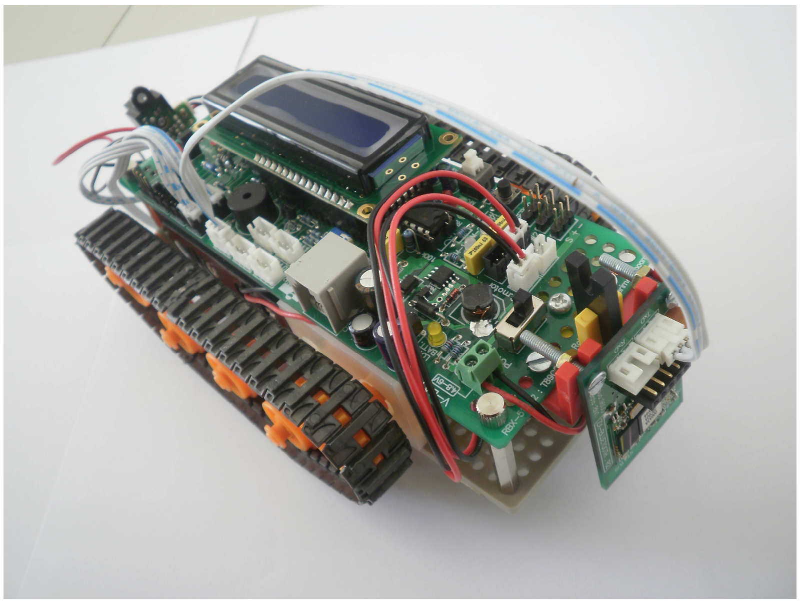
Annex.12 Script st.vbs

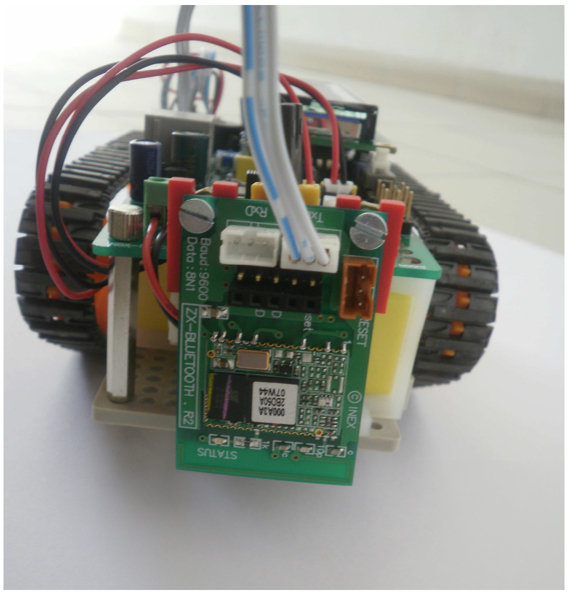
```
<? php
echo "<head><title>Internet Control of a 8051 Robot</title></head>
<body>
<center>
<font face=\"arial, sans-serif\" color=>
<h2 align='center'><font color=\"#4992BB\">Internet Control of a 8051 Robot</font></h2>


<hr align='center' size=\"4\" color=\"#4992BB\"><center>
<TABLE WIDTH=150 BORDER=0 CELLPADDING=0>
<TR> <TH COLSPAN=3><font face=\"arial, verdana\" color=\"#4992BB\">Bluemote</font></TH>
</TR>
<TR> <TD></TD>      <TD><a href='processing.php?action=fw'><img src='forw.gif'
border='0'></a></TD> <TD></TD>   </TR>
<TR> <TD><a href='processing.php?action=rg'> <img src='right.gif' border='0'></a></TD>      <TD> <a
href='processing.php?action=st'><img src='stop.gif' border='0'></TD> <TD><a
href='processing.php?action=lf'> <img src='left.gif' border='0'></a></TD>   </TR>
<TR> <TD></TD>  <TD><a href='processing.php?action=bk'> <img src='back.gif'
border='0'></a></TD> <TD></TD>   </TR>
</TABLE></center>
<hr align='center' size=\"4\" color=\"#4992BB\">

</center>
</body>
```

Annex.13 File Bluemote.php

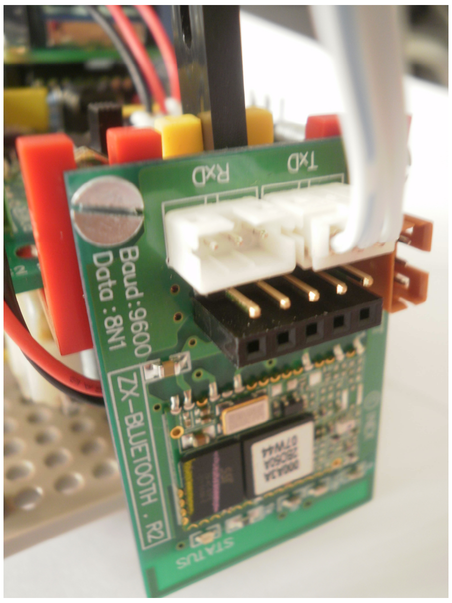
```
<?php

   //check the GET actions variable to see if something needs to be done

if (isset($_GET['action'])) {
//Action has been requested

  if ($_GET['action'] == "fw") {

    $page = "bluemote.php";
     header("Refresh: 2; URL=\"" . $page . "\"");
     exec ('c:\r51\fw.vbs');

      echo "
          <hr align='center' size=\"4\" color=\"#4992BB\"><center>
      <font face=\"arial, sans-serif\"><b>Moving Forward</b></font>";

      }


     else if ($_GET['action'] == "bk") {

     $page = "bluemote.php";

     header("Refresh: 2; URL=\"" . $page . "\"");

     exec ('c:\r51\bk.vbs');

echo "
    <hr align='center' size=\"4\" color=\"#4992BB\"><center>
    <font face=\"arial, sans-serif\"><b>Moving Backward</b></font>";
    }
     else if ($_GET['action'] == "st") {

     $page = "bluemote.php";

     header("Refresh: 2; URL=\"" . $page . "\"");

     exec ('c:\r51\st.vbs');

    echo "
    <hr align='center' size=\"4\" color=\"#4992BB\"><center>
    <font face=\"arial, sans-serif\"><b>Stopping...</b></font>";
    }
     else if ($_GET['action'] == "rg") {

     $page = "bluemote.php";

     header("Refresh: 2; URL=\"" . $page . "\"");

     exec ('c:\r51\rg.vbs');

    echo "
    <hr align='center' size=\"4\" color=\"#4992BB\"><center>
    <font face=\"arial, sans-serif\"><b>Turning Right</b></font>";
    }
     else if ($_GET['action'] == "lf") {

     $page = "bluemote.php";

     header("Refresh: 2; URL=\"" . $page . "\"");

     exec ('c:\r51\lf.vbs');

    echo "
    <hr align='center' size=\"4\" color=\"#4992BB\"><center>
    <font face=\"arial, sans-serif\"><b>Turning Left</b></font>";
    }
    }
?>
```

Annex.14 File processing.php

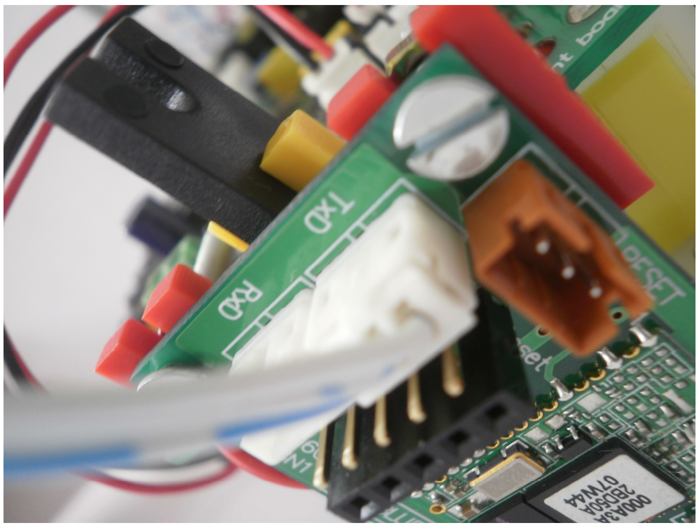

Annex.15 Robot-51 with the Zx-Bluetooth implementation



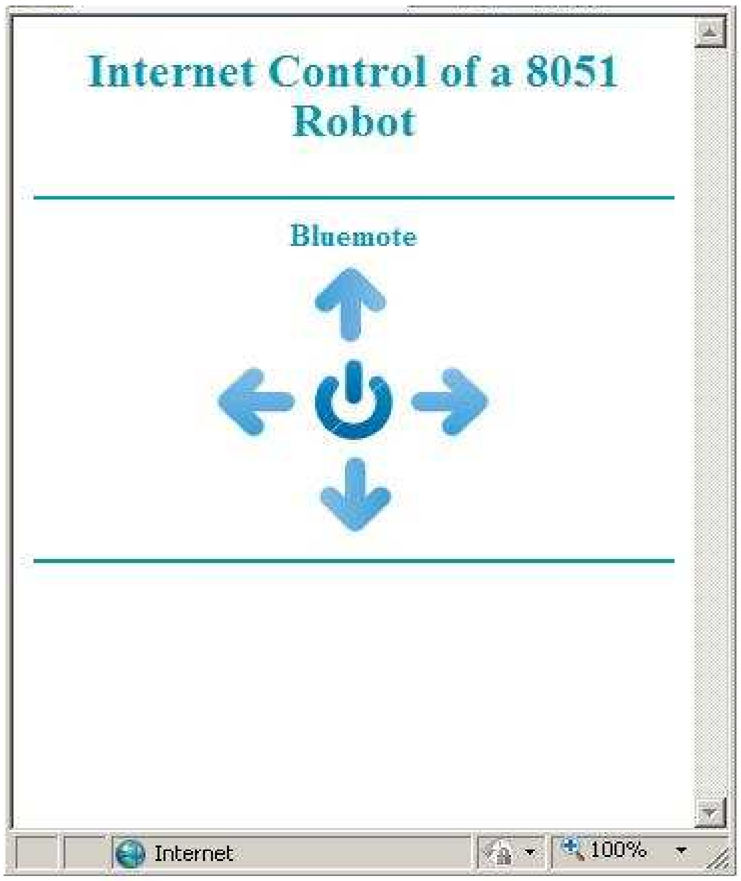
Annex.16 Front View of Robot-51



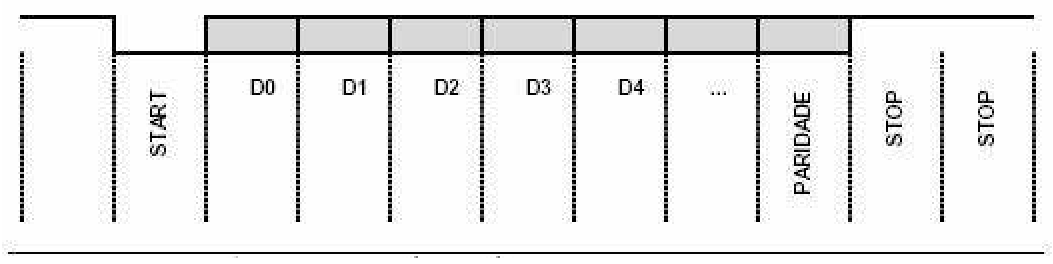
Annex.17 Zx-Bluetooth board



Annex.18 Detail of TxD's connection



Annex.19 Control available in the internet



Annex.20 Example of a Data Waveform
RS232- Serial Communication

Videos in Youtube:

Bluemote - http://www.youtube.com/watch?v=ZGFYZh66vdA

Bluemote Part 2 - http://www.youtube.com/watch?v=QP3fQjwfou4

Bibliography

MAZIDI, Muhammad Ali; MAZIDI, Janice Gillispie; MCKINLAY, Rolin D.; The 8051 Microcontroller and Embedded Systems – Using Assembly and C, Edit. Prentice hall of India; 2ª Edição.

Webgraphy

- Inex Global; Zx-Bluetooth; [consult.2010-09-01]; Available at www: <URL http://www.inexglobal.com/products.php?type=wireless&cat=wireless&model=zxbluetooth >
- Aliatron; Robo-51; [consult. 2010- 08- 30]; Available at www: <URL http://aliatron.com/loja/catalog/product_info.php?products_id=475&language=pt&osCsid=gtb3 >
- Inex Global; Robo-51; Robo-51; [consult. 2010-08-26]; Available at www: < URL http://www.inexglobal.com/products.php?type=ROBOTKIT&cat=EDUROBOTKIT&model=robo51 >
- Inex Global; rb51_e20s; Manual do Robo-51; [consult. 2010-09-01]; Available at www: < URL http://www.inexglobal.com/downloads/rb51_e20s.pdf>
- Sweex; Bluetooth Micro Adapter Class I USB; BT214; [consult.2010-08-31]; Available at www <URL http://www.sweex.com/en/assortiment/connectivity/bluetooth-adapters/BT214
- AAVV; Wikipédia; Bluetooth; [consult. 2010-08-31]; Available at www < URL http://pt.wikipedia.org/wiki/Bluetooth >
- AAVV; Wikipédia; Bluetooth profile; [consult. 2010-09-10]; Available at www < URL http://en.wikipedia.org/wiki/Bluetooth_profile >
- AAVV; Wikipédia; RS-232; [consult. 2010-09-02]; Available at www < URL http://pt.wikipedia.org/wiki/RS-232 >
- Christopher E. Strangio; The RS-232 Standard ; RS-232 [consult. 2010-09-02]; Available at www <URL http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html >

- A generation of new Tecnology; Answer: Keyboard Redirection in a DOS batch file?; Bill Blaton; [consult. 2010-08-24]; Available at www <URL http://us.generation-nt.com/answer/keyboard-redirection-dos-batch-file-help-197975321.html >
- BluSoleil; BlueSoleil-Bluetooth Software; BlueSoleil 8.08; [Consult. 2010-08-22]; Available at www: <URL http://www.bluesoleil.com/products/SoftwareDetail.aspx?ProductID=S0001201005190001 >
- Softpedia; Download Termite 2.4; Termite 2.4; [consult. 2010-08-21]; Available at www: <URL http://www.softpedia.com/get/Network-Tools/Telnet-SSH-Clients/Termite-CompuPhase.shtml >
- Albrecht Schmidt; SerialTerm – command line terminal program; Download SerialTerm.exe; [consult. 2010-08-22]; Available at www: <URL http://www.comp.lancs.ac.uk/~albrecht/sw/terminal/index.html>
- Apache Friends; Apache Friends- XAMPP for windows; XAMPP; [consult. 2010-08-30]; Available at www: <URL http://www.apachefriends.org/en/xampp-windows.html>
- EAthena support board; Tuturial Básico do XAMPP lite; Talker; [consult. 2010-09-01]; Available at www: <URL http://www.eathena.ws/board/index.php?showtopic=162648>