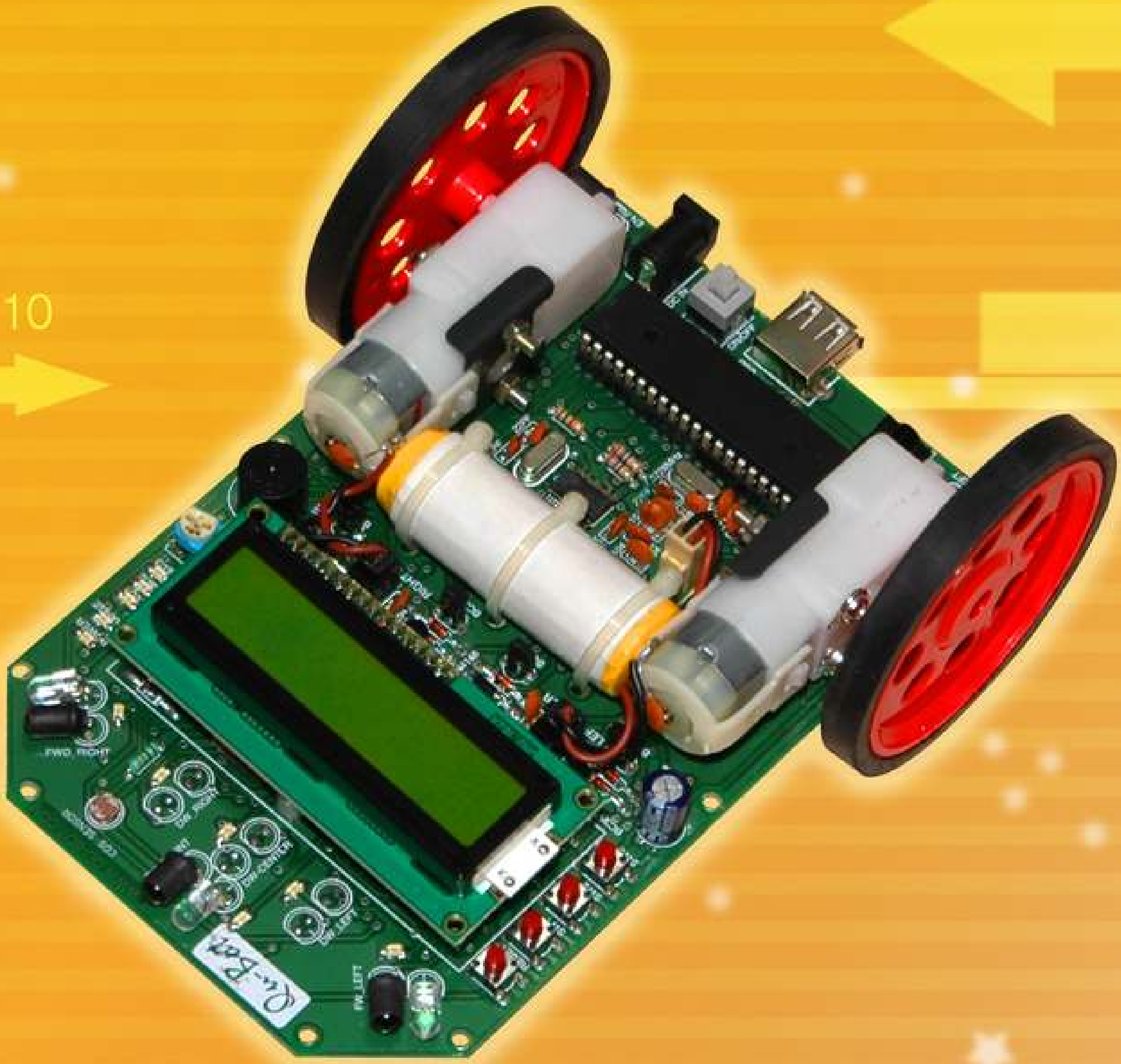
Qu-Bot

Quickly Programmable Robot



User's Manual

http://www.qu-bot.com

Precautions

This manual contains crucial information including the precautions that must be taken seriously to insure trouble free operation of Qu-Bot robot as well as prevent damage for the users and others. Please read the manual carefully before starting assembly and programming of Qu-Bot.

- **Do not disassemble or tamper with any part of the machine.**
- **When power is turned on do not touch any conductor such as metal parts to robot. This may damage the robot or some parts of it permanently.**
- **Remove the battery connector when robot is not in use for a long time.**
- **Do not store the robot for long time when the battery is discharged. This will drastically decrease the life of battery.**
- **Keep the kit away from small children. There are many small parts which can create problem if swallowed.**
- **Battery and motor connectors are polarized. Do not try to force insert them in wrong direction. This will damage the robot permanently.**
- **Do not dispose the robot or battery in fire. There is a danger of explosion and poisonous fumes.**
- **There are many possible uses of this robot except described in this manual. All these experiments and modifications should be carried out at owner's risk.**
- **Specifications and product design are subject to change without prior notice. Please refer the latest manual from website www.qu-bot.com.**

Preface

Qu-Bot is a Quickly Programmable Robot. Qu-Bot is specially designed for beginners in robotics and for all kind of robotics enthusiasts, starting from small kids to adults. The Qu-Bot visual interface software allows the robot to be programmed easily and quickly.

The Qu-Bot Robot Kit is a very easy to assemble, programmable robotic kit that includes a variety of sensors, motors, controller board, remote control, visual icon based software QBVLab, battery, charger, manual etc. Virtually everything you need to build, code, program and execute the program on Qu-Bot Kit. This product is geared towards classroom and group instruction of fundamentals of robotics. Bulk discounts and classroom multi-packs are available. Please contact **info@qu-bot.com** for inquiries.

The Qu-Bot Robot kit connects to the PC via USB cable (included) and is programmable through an easy-to-use QBVLab drag and drop modular icon based software program. After the program sequences have been uploaded into the robot via USB, it can be controlled by remote control, PC or can behave according to the user defined parameters.

The Qu-Bot runs on Li-ion battery and offers hours of endless educational entertainment. Also the battery can be recharged by USB or charger included in the kit. This is an entry level product that requires no programming or robotics experience. A great practical tool for hands-on in the field of robotics.

Qu-Bot can simply turn to following operations.

1. Line Follower Robot
2. Wall Hugger Robot
3. Object Tracer Robot
4. Light Tracer Robot
5. Hand Hugger Robot
6. Obstacle Avoider Robot
7. Intelligent Grid Mover Robot
8. Remote Controlled Robot
9. PC USB Controlled Robot
10. Maze Solver Robot
11. Sumo Robot

Qu-Bot Salient features:

1. Easy and simple to use
2. No soldering or gluing required
3. Made for complete beginners (No knowledge of Robotics or Programming needed)
4. Develop skills of microelectronics, programming, robotics and sensors
5. Assembles in 10minutes – easy and quick start into robotics.
6. Virtually all items included. Nothing needs to be bought separately. E.g. Battery, charger, remote, USB Cable etc.
7. USB connectivity for maximum compatibility with all PC's and Laptops.
8. Easy Icon-based visual programming for beginners – sample programs included
9. Onboard li-ion rechargeable battery which can run Qu-Bot for more than 3 hours.
10. Battery charged through USB or special charger

Input Modules Onboard:

1. 3X Proximity Sensors
2. 3X Line Follower Sensors
3. 2X Motor Counter Sensors
4. 1X Remote Control Receiver Sensor
5. 1X Ambient Light Sensor
6. 3X General Purpose Switches

Output Modules Onboard:

1. 1X 16X2 Character LCD
2. 3X General Purpose LED's
3. 2X Motors & Motor Drivers
4. 1X Buzzer
5. 1X USB TX & RX

Other Board Features:

1. 1X Power Indication LED
2. 1X Charge Indication LED
3. 1X 4 Way Sensor Selection Switch
4. 8X Sensor State Indication LED's
5. 1X Reset Switch
6. 1X Power On/Off Switch
7. 1X DC Input Jack

Qu-Bot Package Includes:

1. 1X Fully Assembled Main Circuit board with all above features
2. 1X Li-ion Rechargeable Battery 3.7V, 1.4Ah with protection circuit
3. 2X Geared DC motors
4. 2X Mounting Clamps for motor
5. 2X Wheels
6. 1X Castor
7. 1X USB Cable
8. 1X Remote Control
9. 1X Battery Charger Adaptor
10. 1X CD with all software and documentation
11. 1X Printed Manual
12. All screws, nuts and cable ties

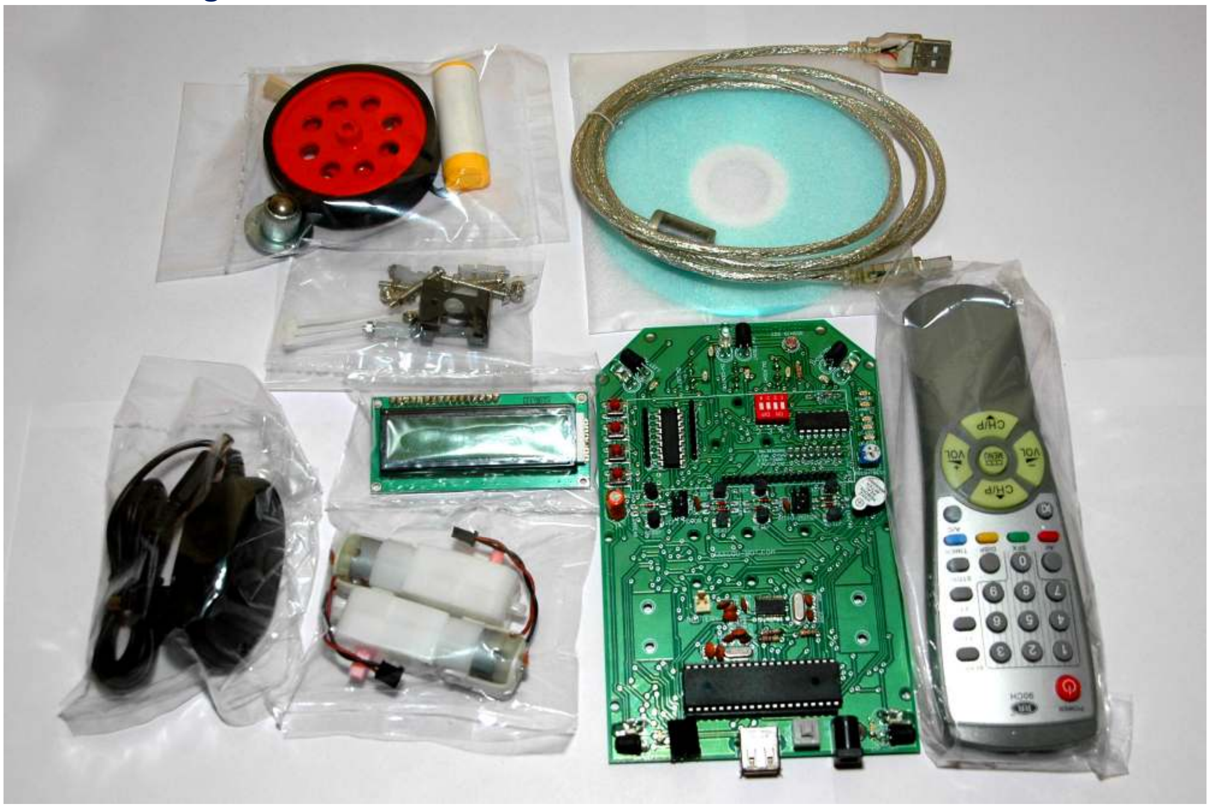
Tools required assembling Qu-Bot:

1. Screw driver
2. Pliers or spanner

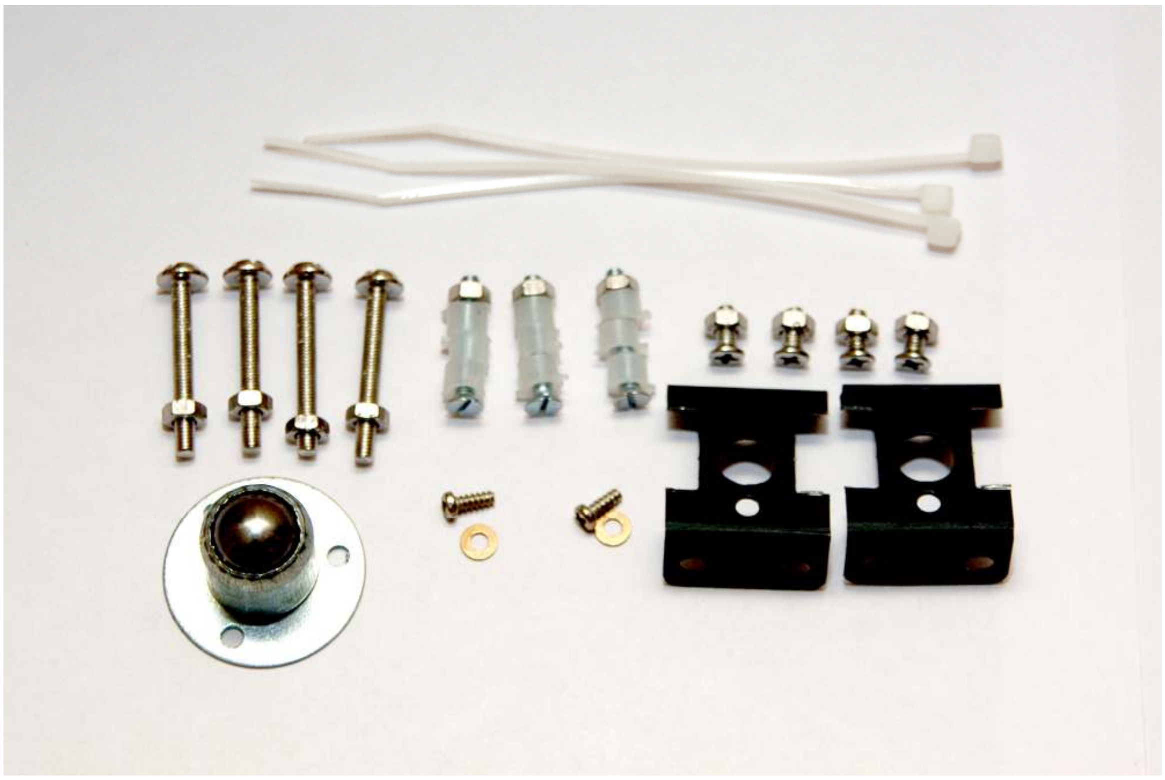
Index

1. Identification of components & Assembling

1.1. Assembling Qu-Bot

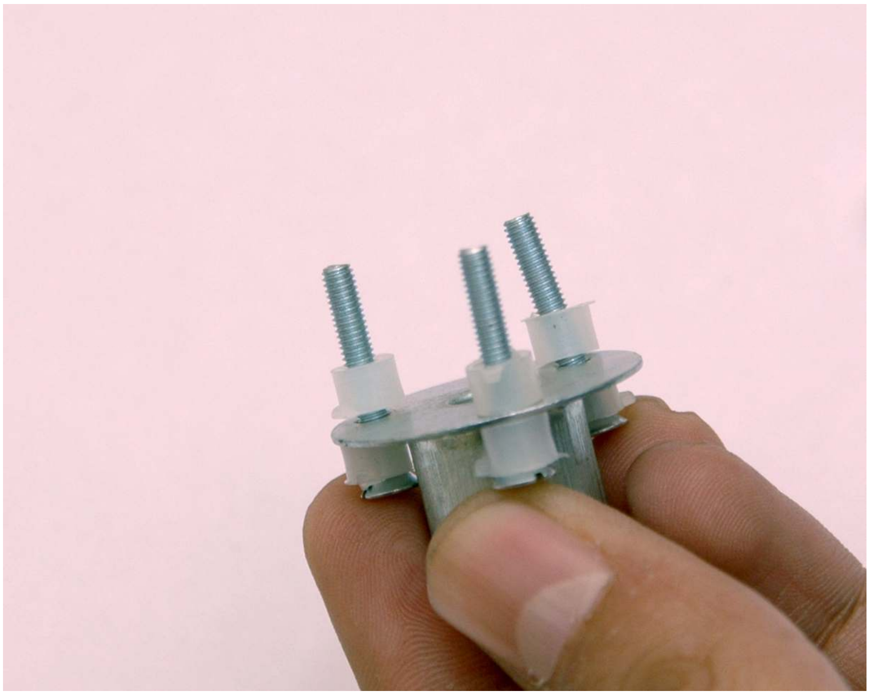


Qu-Bot Kit contents out of the box.



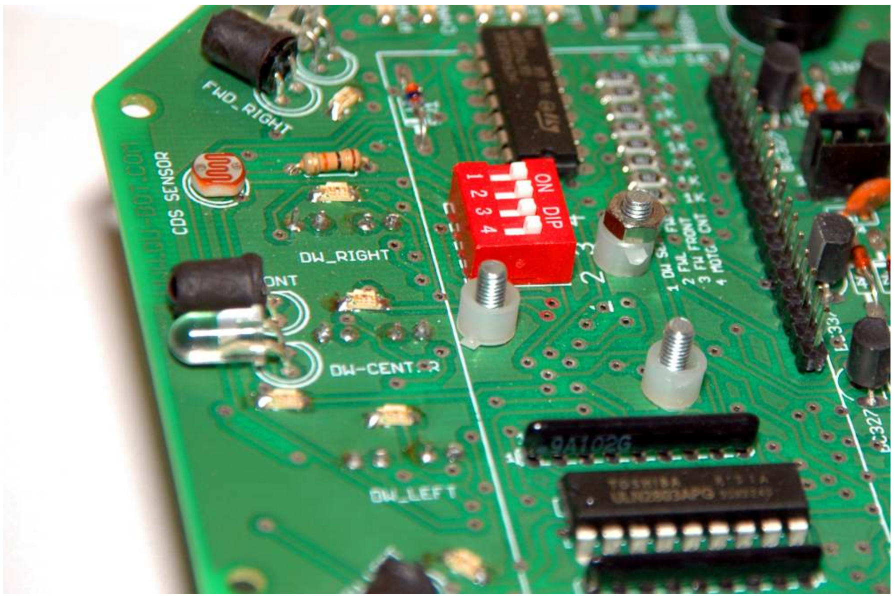
Castor, motor clamps, cable ties and different screws and nuts included in kit.

Step 1



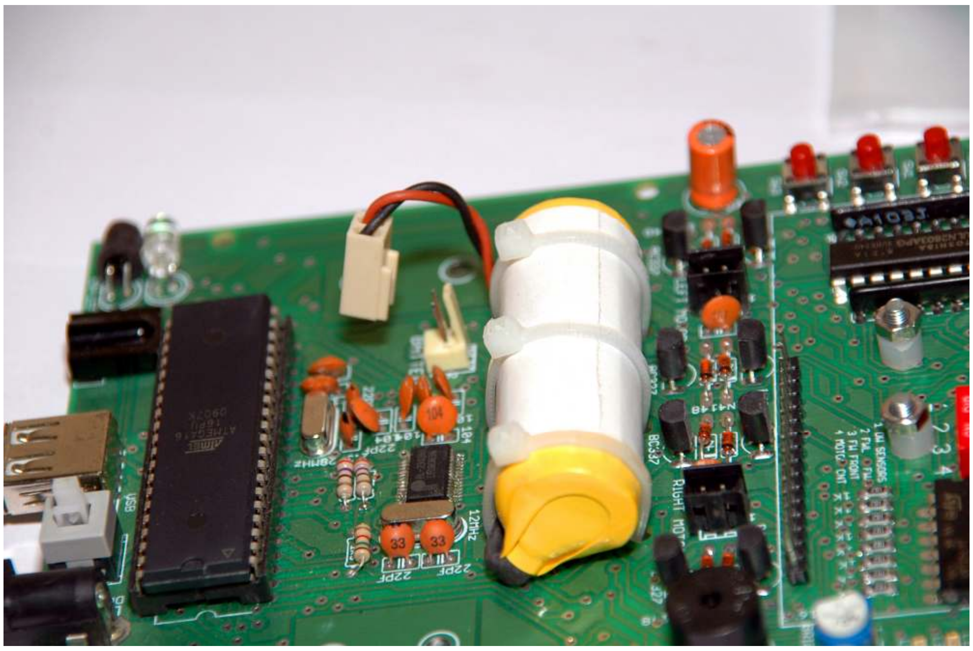
Insert 3 long screws in the castor wheels 3 holes keeping plastic washers in-between as shown in picture.

Step 2



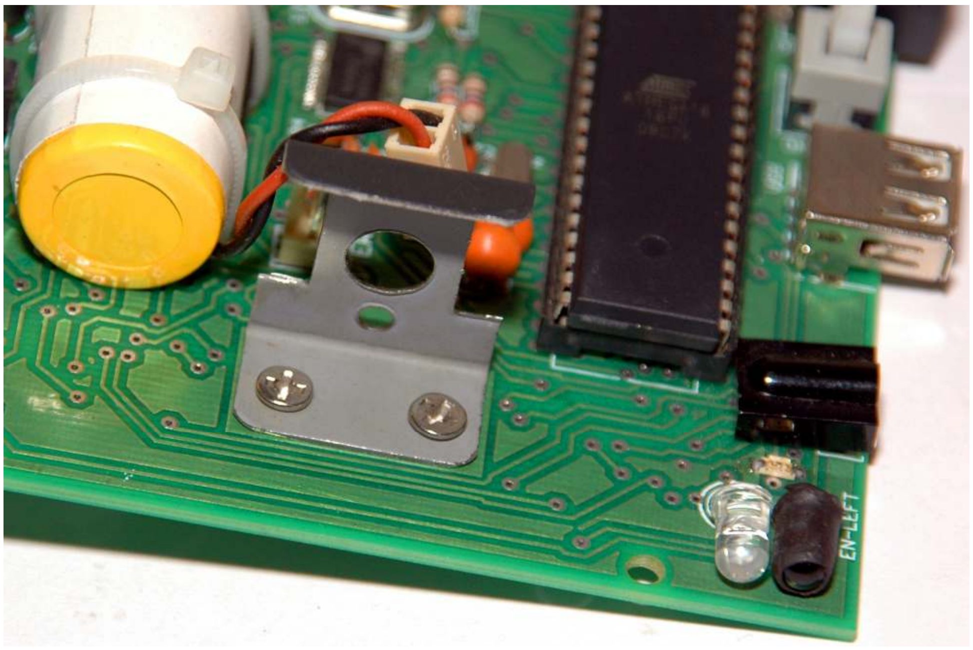
Insert castor wheel with screws to the Qu-Bot PCB allocating the 3 holes. After inserting place 3 more plastic washers on the top and lock the castor wheel to PCB with nuts.

Step 3



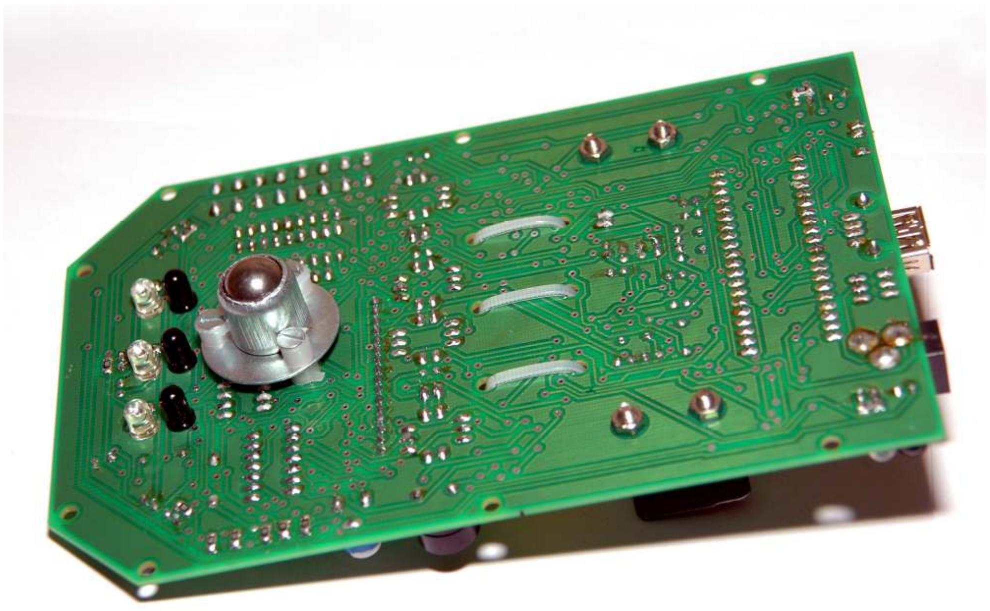
Place the battery in the center of Qu-Bot between 6 holes. Secure the battery to PCB with help of 3 cable ties included in the kit as shown above. Cut the extra portion of cable ties with paper cutter or nipper.

Step 4



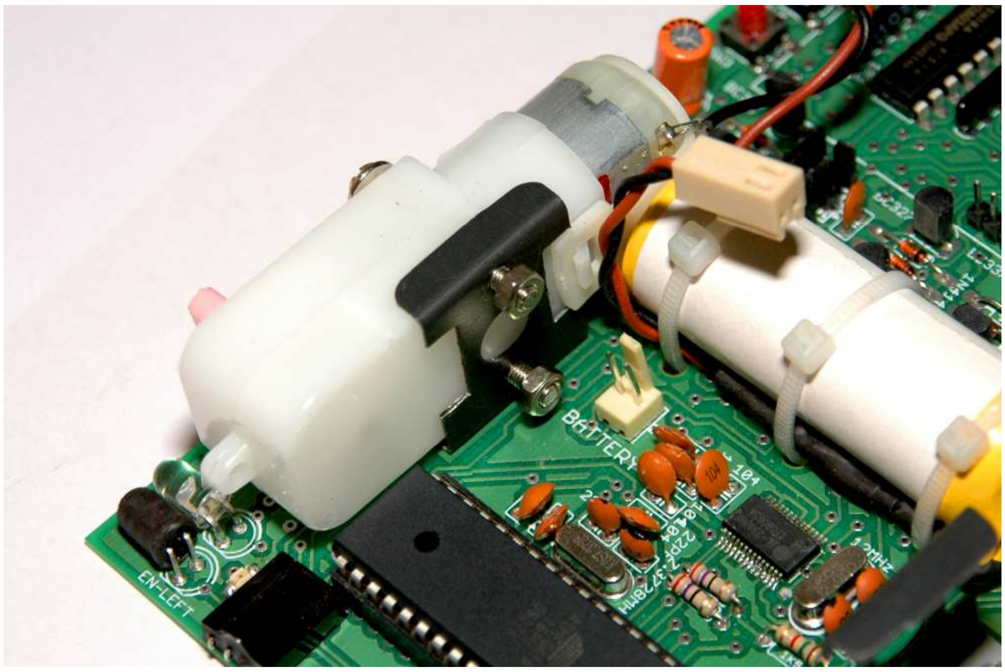
Place the clamp on the 2 holes on the back part of robot. Clamp it with PCB with 2 small screws and nuts. The placement is very critical. If the clamp is not parallel to PCB edge the motor will not be straight and robot will now move in a straight line. A good idea is to align the clamp to PCB track while assembly.

Step 5



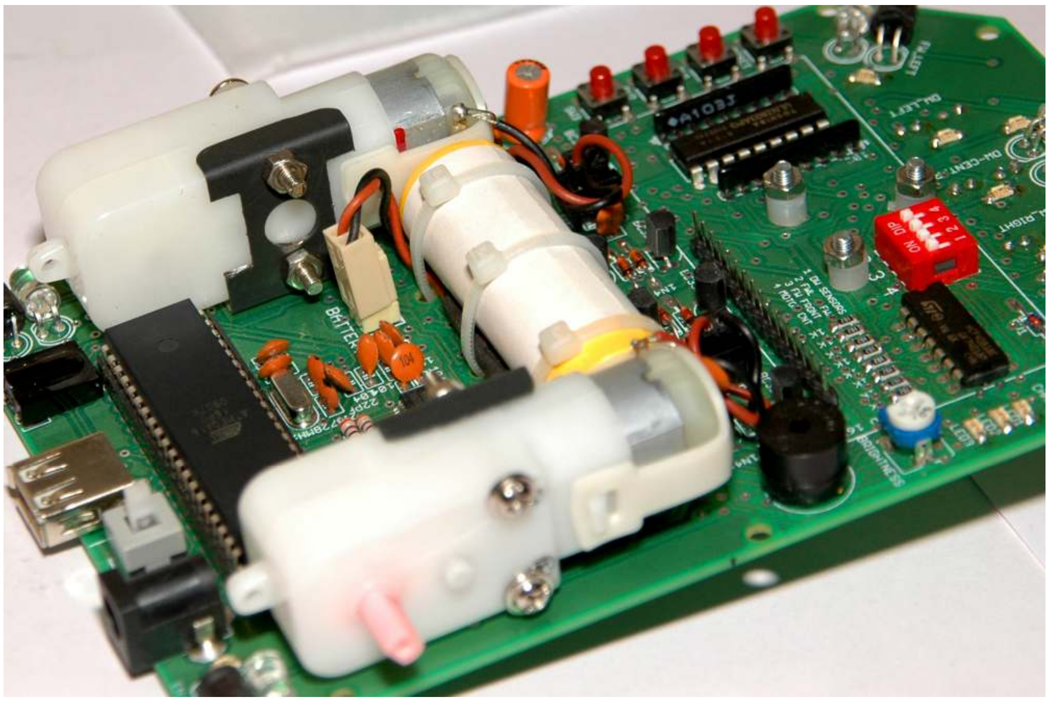
After securing a castor, battery and 2 clamps the robot now looks as above from back side.

Step 6



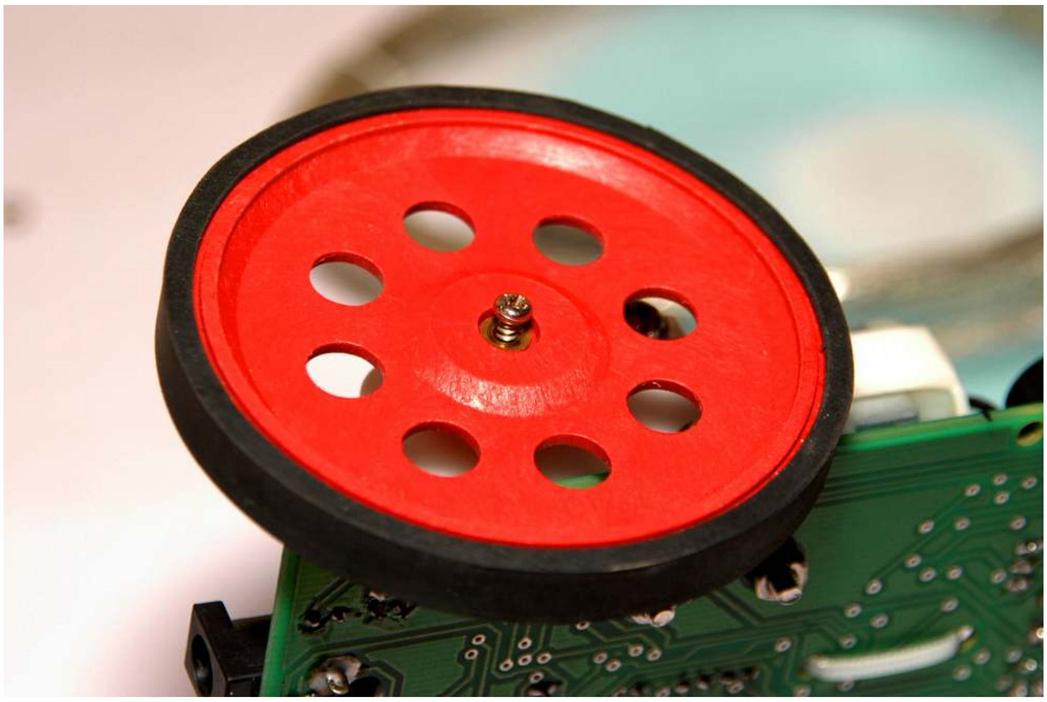
Now attach motor to the clamp as shown above. Put the screws inside the holes in motors through clamps and tighten nuts on them as shown in above picture.

Step 7



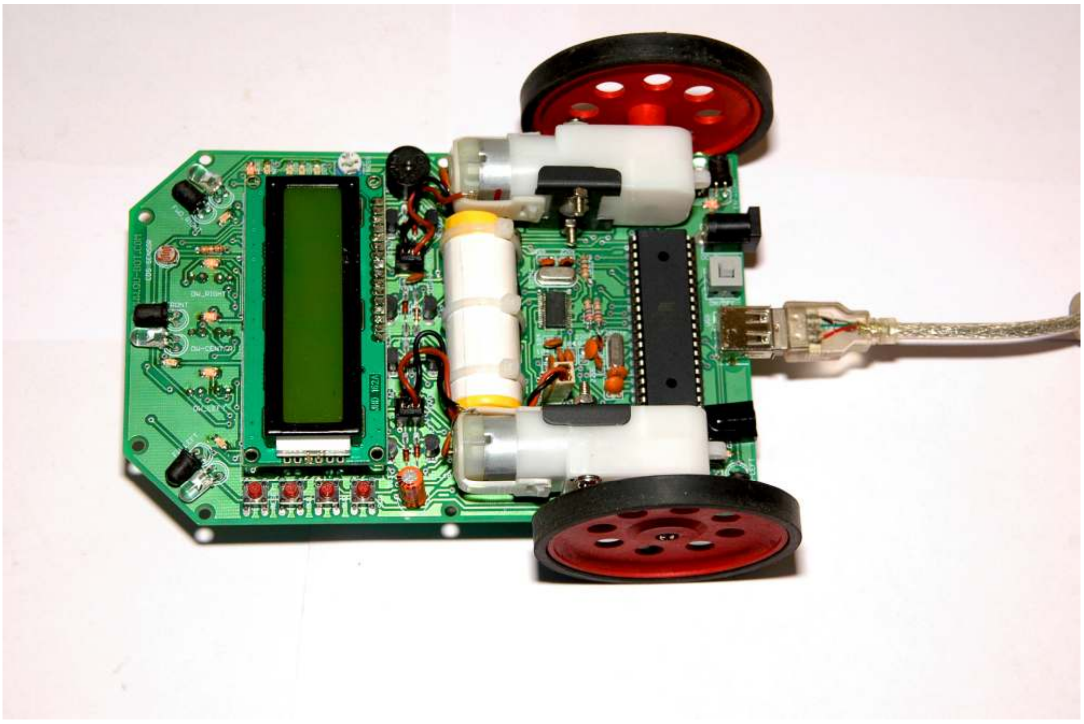
After screwing 2 motors, connect the motors and battery to the board. Take care while plugging in the connectors. All the connectors are polarized so they cannot be inserted wrongly but still do not try to force them in different configuration. This may damage the board permanently.

Step 8



Now place 2 wheels over the motor shaft, press them properly and secure them with a metal washer and screw as shown in picture.

Step 9

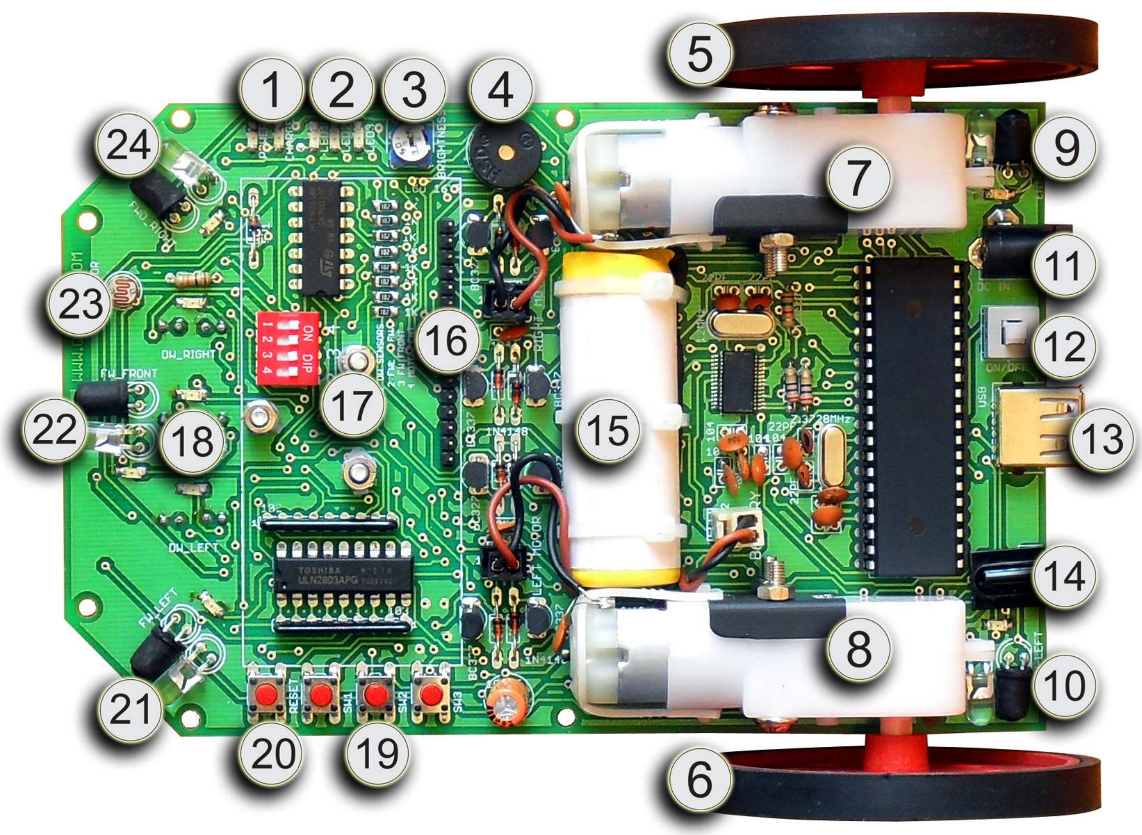


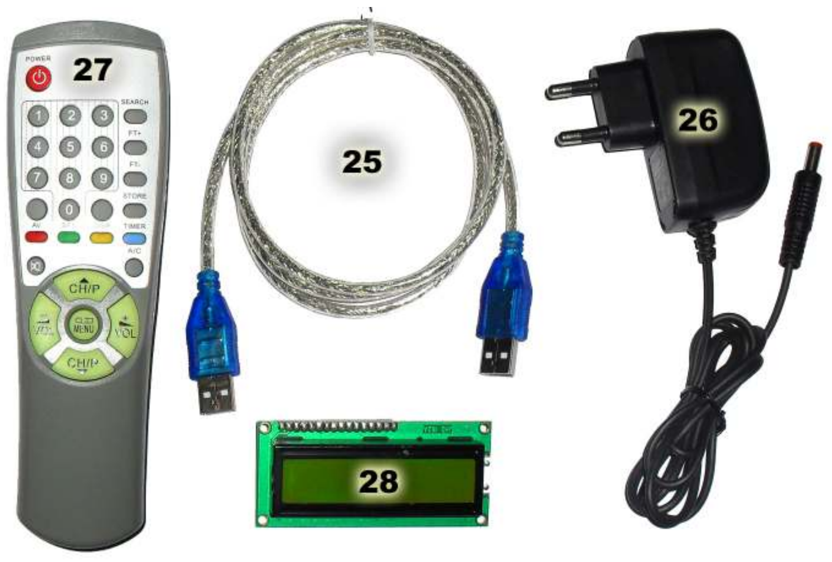
Insert the LCD on the top. Now the Qu-Bot is ready to use. It's advisable to charge the battery before using it. The Qu-Bot can also get charged or run directly on USB power but it's advisable to use AC charger for fast charging. When the charge led will glow the charging is complete.

When Qu-Bot is connected to PC using USB it gets powered up irrespective of the state of power switch. When AC charger is connected the Qu-Bot will be disconnected from battery and charger but if USB cable is connected to PC the robot will be working and battery will get charged through AC power.

Charging battery though USB may take up to 12-15 hours. Disconnect LCD and switch off sensors (See Section 1.2, Component 17) to save power and charge faster through USB.

1.2. Identification of parts on assembled Qu-Bot





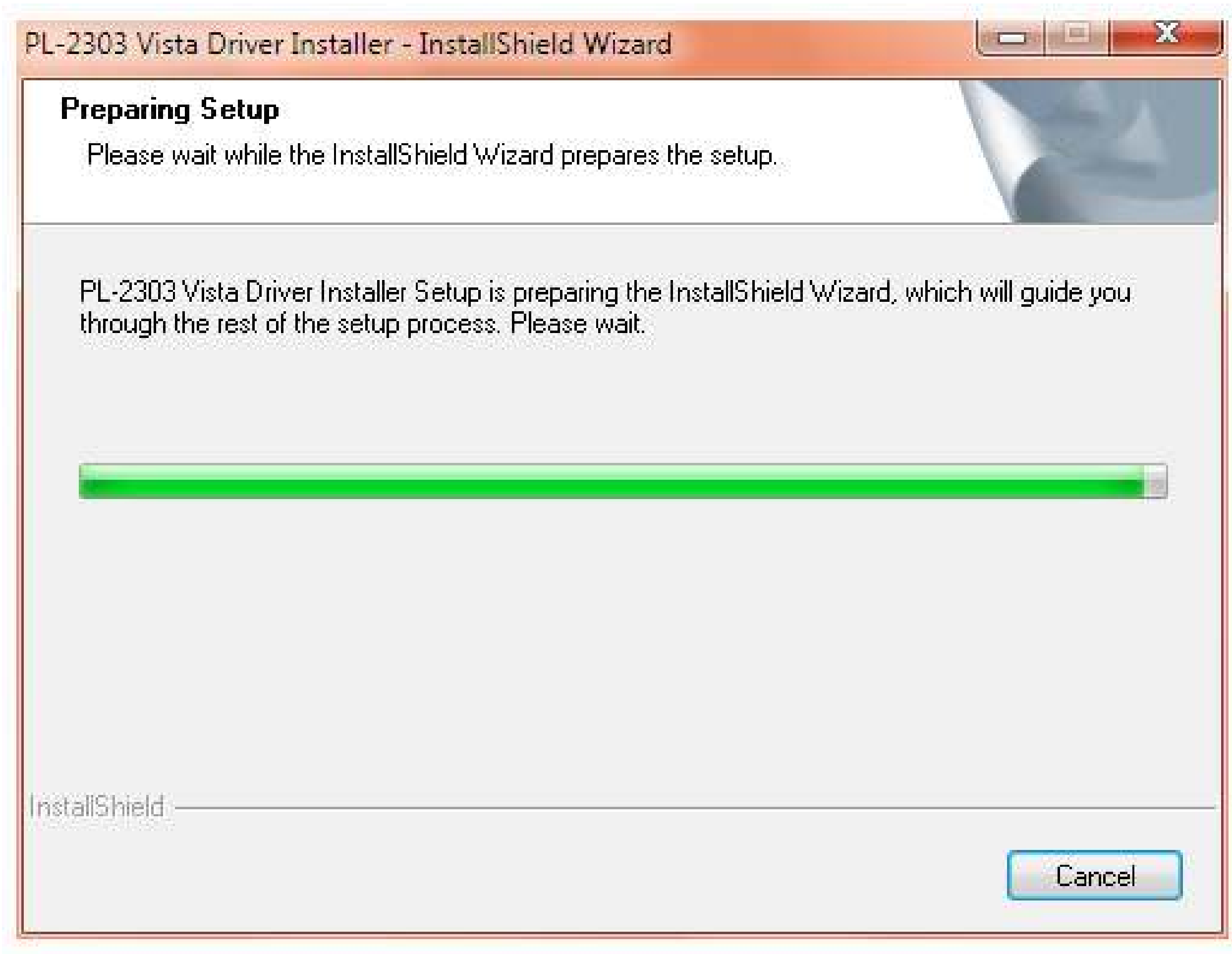
Component	Usage
1. Power & Charge LED's	Indicate whether the Qu-Bot has been powered up. Charge LED will glow once battery is charged through AC charger.
2. LED1, LED2 & LED3	Lights up through software

3. LCD Brightness Control	Controls Brightness of 16X2 LCD
4. Buzzer	Buzzes
5. Right Wheel	Right Driving wheel
6. Left Wheel	Left Driving wheel
7. Right Motor	Right Driving motor
8. Left Motor	Left Driving motor
9. Left encoder sensor	Counts steps the left motor has moved. Counts automatically. It Will not count if wheel moves due to any external force other than motor.
10. Left encoder sensor	Counts steps the right motor has moved.
11. DC Jack	Input for charging through AC Charger.
12. On/Off Switch	Power On/Off Switch. If Pressed power on.
13. USB Connector	Connects to USB of PC through USB cable. Qu-Bot also gets powered and charged.
14. IR Remote Sensor	Receives signals from IR remote
15. Li-ion Battery	Battery which powers up Qu-Bot. Remove Battery connector when not in use.
16. LCD Header	Connects to 16X2 LCD.
17. Sensor Select Switch	Controls power to sensors 9,10,18,21,22&24 Sensors use lot of battery power. Turn them off if they are not used. Switch1 controls power to sensor 18 Switch2 controls power to sensor 21,24 Switch3 controls power to sensor 22 Switch4 controls power to sensor 9,10 To turn off power drag switch to left side.
18. Line follower sensors	Also called as downward Sensors.
19. SW1, SW2, SW3	Input Switches
20. Reset Switch	Press and release to restart program.
21. Left Obstacle sensor	Also called as Forward Left sensor
22. Center Obstacle Sensor	Also called as Forward Center sensor
23. LDR Sensor	Ambient Light Sensor
24. Right Obstacle Sensor	Also called as Right Center sensor
25. USB Cable	Used to connect Qu-Bot to PC
26. AC Charger	For recharging of battery
27. IR Remote	Sends wireless infrared signals to Qu-Bot
28. LCD	16X2 Characters LCD. Connects to 16.

2. USB driver & QBVLab Software Installation

2.1. USB Driver Installation

Before connecting the Qu-Bot to USB port of PC you need to install driver for USB on your PC. Qu-Bot is compatible with most windows versions. In this manual all screenshots are taken from Windows 7 Ultimate operating system, however Qu-Bot is tested on most of the windows versions successfully. There are 2 files USB driver XP.exe and USB driver Vista-7.exe in Qu-Bot CD. If your operating system is Windows XP or earlier then run USB driver XP.exe. If you are running Windows Vista or Windows 7 then run USB driver Vista-7.exe.

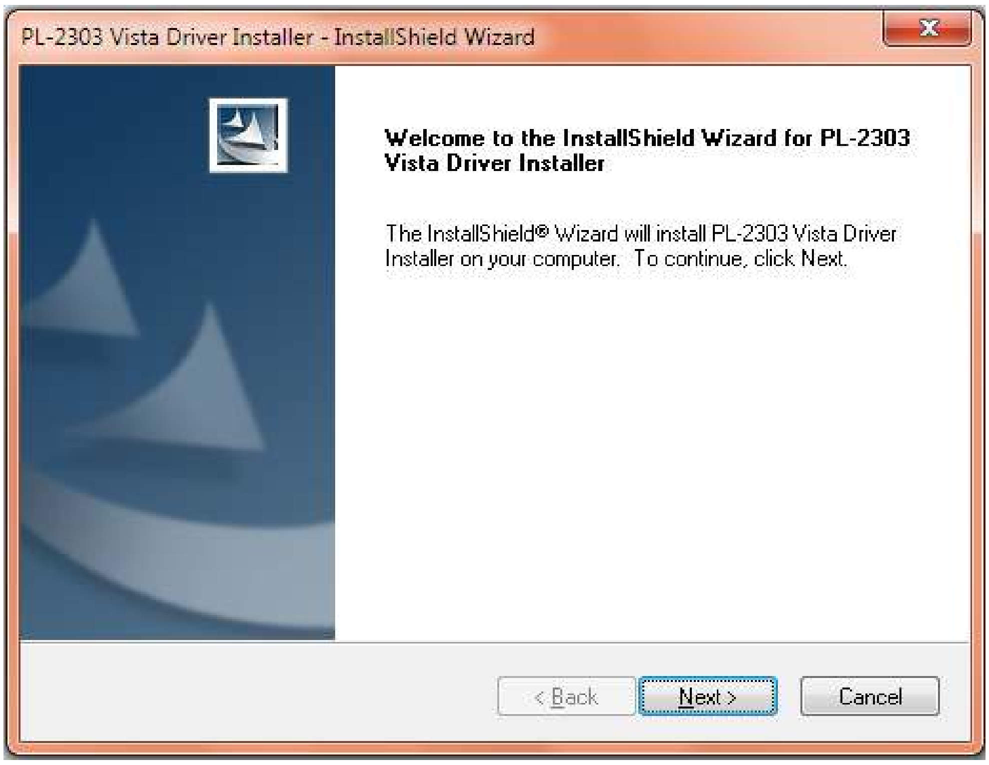

PL-2303 Vista Driver Installer - InstallShield Wizard

Preparing Setup

Please wait while the InstallShield Wizard prepares the setup.

PL-2303 Vista Driver Installer Setup is preparing the InstallShield Wizard, which will guide you through the rest of the setup process. Please wait.

After starting the setup the above window appears. Qu-Bot uses PL2303 chip for USB communication. This is the driver for the same.

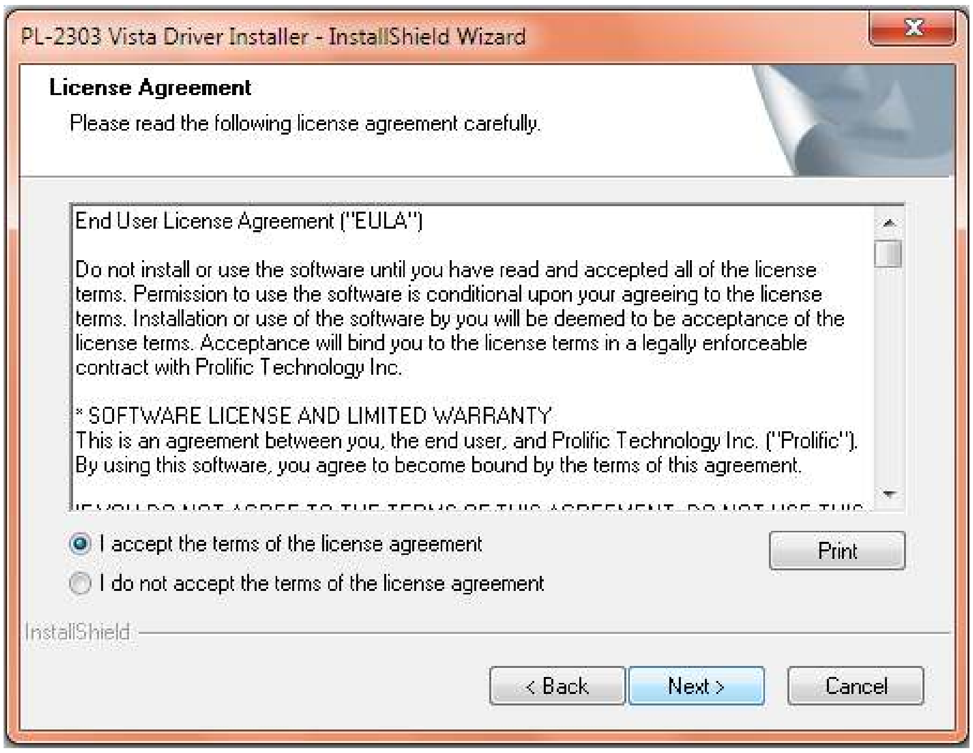


PL-2303 Vista Driver Installer - InstallShield Wizard

Welcome to the InstallShield Wizard for PL-2303 Vista Driver Installer

The InstallShield® Wizard will install PL-2303 Vista Driver Installer on your computer. To continue, click Next.

Press next to continue.



PL-2303 Vista Driver Installer - InstallShield Wizard

License Agreement

Please read the following license agreement carefully.

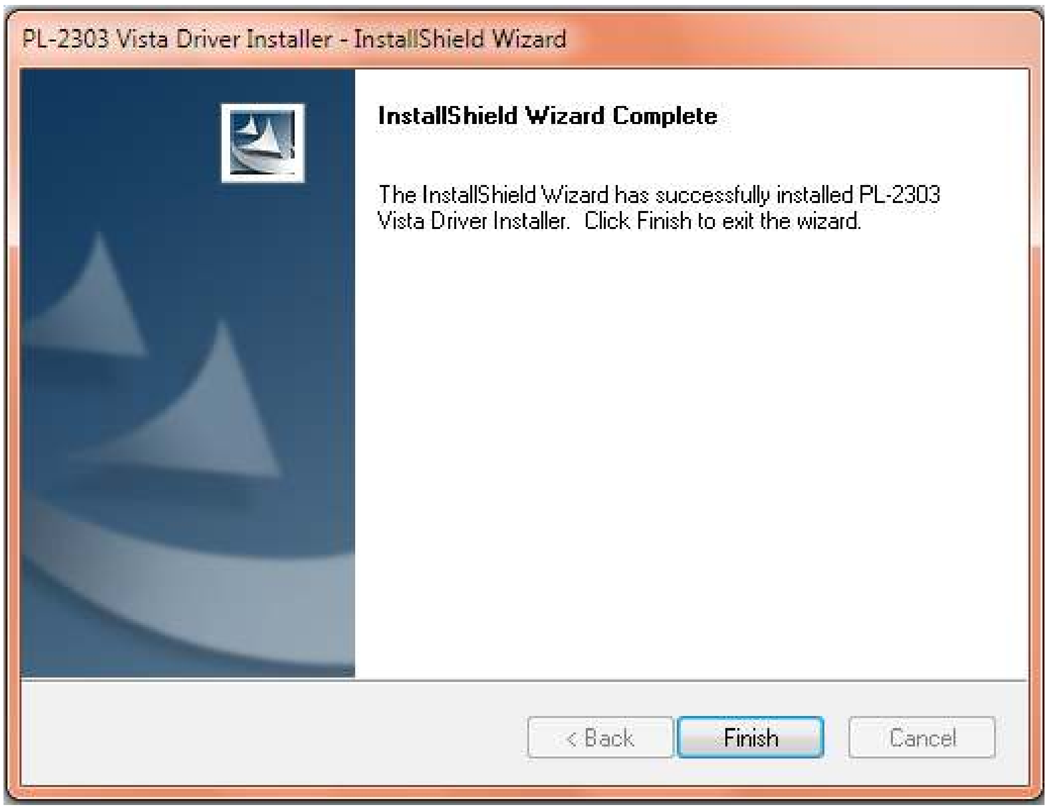
End User License Agreement ("EULA")

Do not install or use the software until you have read and accepted all of the license terms. Permission to use the software is conditional upon your agreeing to the license terms. Installation or use of the software by you will be deemed to be acceptance of the license terms. Acceptance will bind you to the license terms in a legally enforceable contract with Prolific Technology Inc.

* SOFTWARE LICENSE AND LIMITED WARRANTY
This is an agreement between you, the end user, and Prolific Technology Inc. ("Prolific"). By using this software, you agree to become bound by the terms of this agreement.

[✓ I accept the terms of the license agreement]
I do not accept the terms of the license agreement

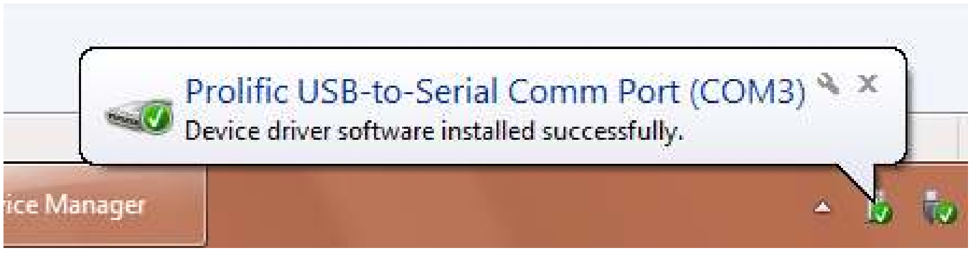
On License agreement page click "I accept…" and click next.


PL-2303 Vista Driver Installer - InstallShield Wizard

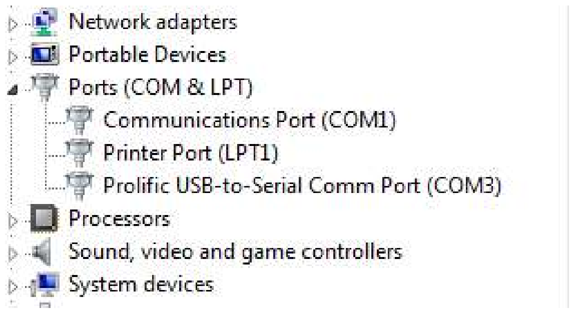
InstallShield Wizard Complete

The InstallShield Wizard has successfully installed PL-2303 Vista Driver Installer. Click Finish to exit the wizard.

This finishes the installation. Press finish to close the window. Now the driver is installed.


Prolific USB-to-Serial Comm Port (COM3)
Device driver software installed successfully.

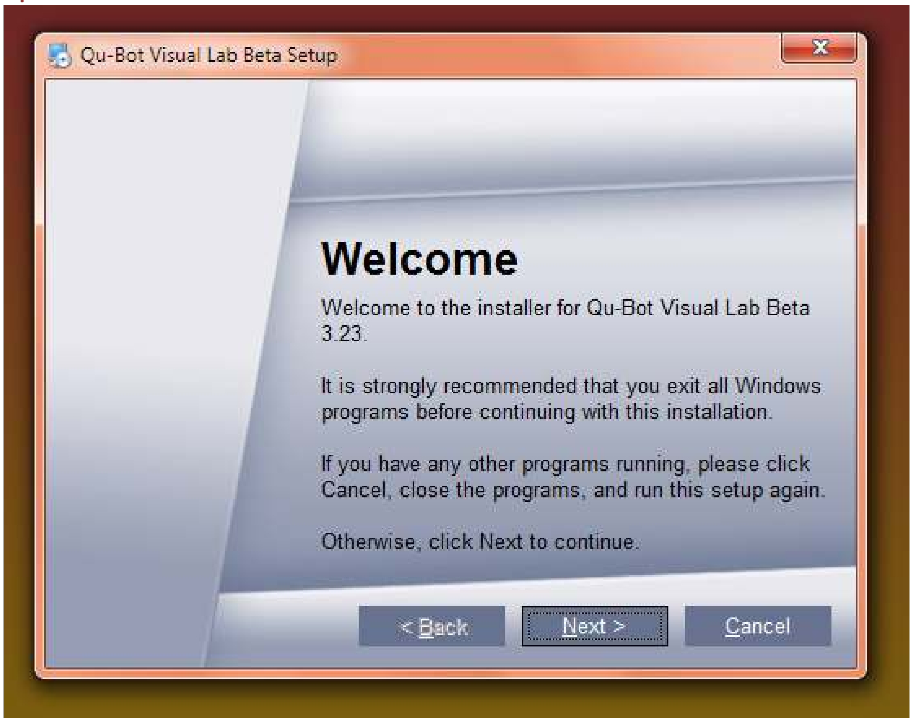
Inserting the USB cable from Qu-Bot should now automatically detect the device and give connect device sound. It will also show balloon from the task bar as shown in above picture.


Network adapters
Portable Devices
Ports (COM & LPT)
Communications Port (COM1)
Printer Port (LPT1)
Prolific USB-to-Serial Comm Port (COM3)
Processors
Sound, video and game controllers
System devices

In case of any problem to verify that driver is installed correctly you can check Control Panel-System-Device Manager – Ports. There should be a device called Prolific USB-to-Serial Comm Port (Port number).

2.2. Qu-Bot Visual Lab Installation

In Qu-Bot CD there is a setup file for Qu-Bot Visual Lab software. The file name is QBVLabSetup.exe. Run this file to install QBVLab software.



Qu-Bot Visual Lab Beta Setup

Welcome

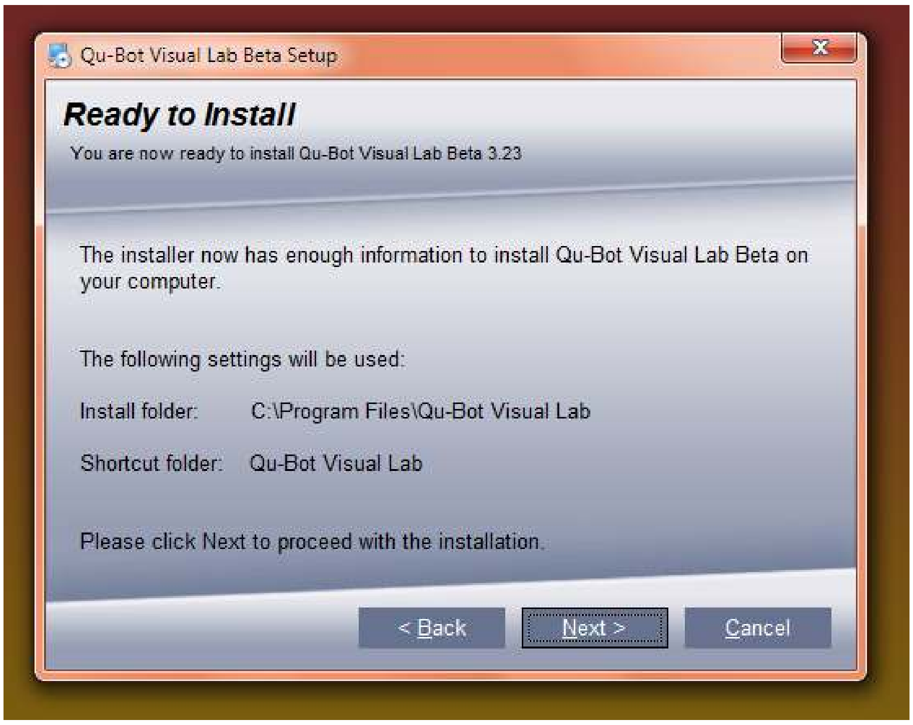
Welcome to the installer for Qu-Bot Visual Lab Beta 3.23.

It is strongly recommended that you exit all Windows programs before continuing with this installation.

If you have any other programs running, please click Cancel, close the programs, and run this setup again.

Otherwise, click Next to continue.

On the welcome screen click next to install software.



Qu-Bot Visual Lab Beta Setup

Ready to Install

You are now ready to install Qu-Bot Visual Lab Beta 3.23

The installer now has enough information to install Qu-Bot Visual Lab Beta on your computer.

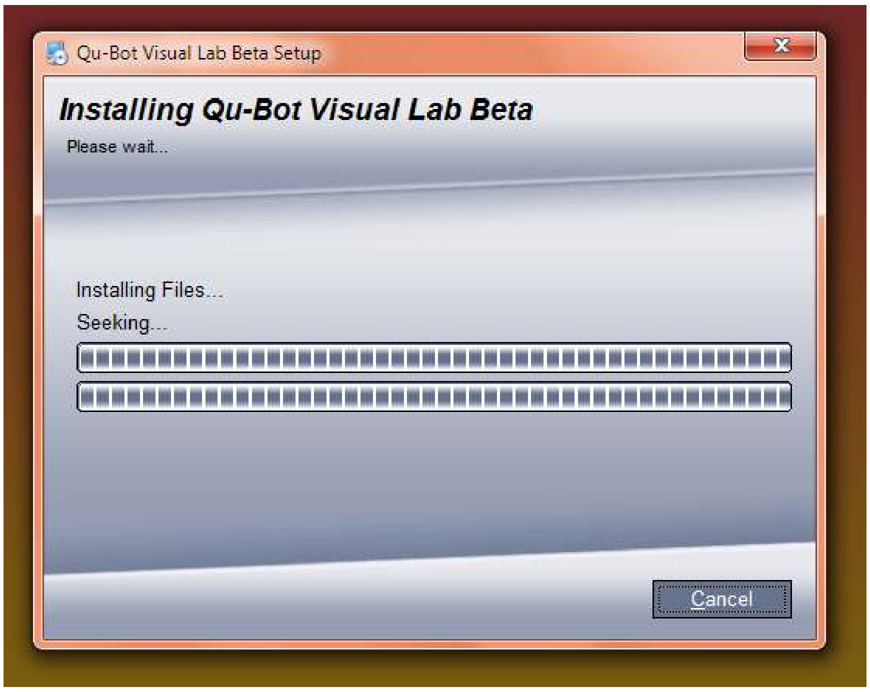
The following settings will be used:

Install folder: C:\Program Files\Qu-Bot Visual Lab

Shortcut folder: Qu-Bot Visual Lab

Please click Next to proceed with the installation.

On ready to install page it shows location where the files will be installed. Click next to start setup.


Qu-Bot Visual Lab Beta Setup

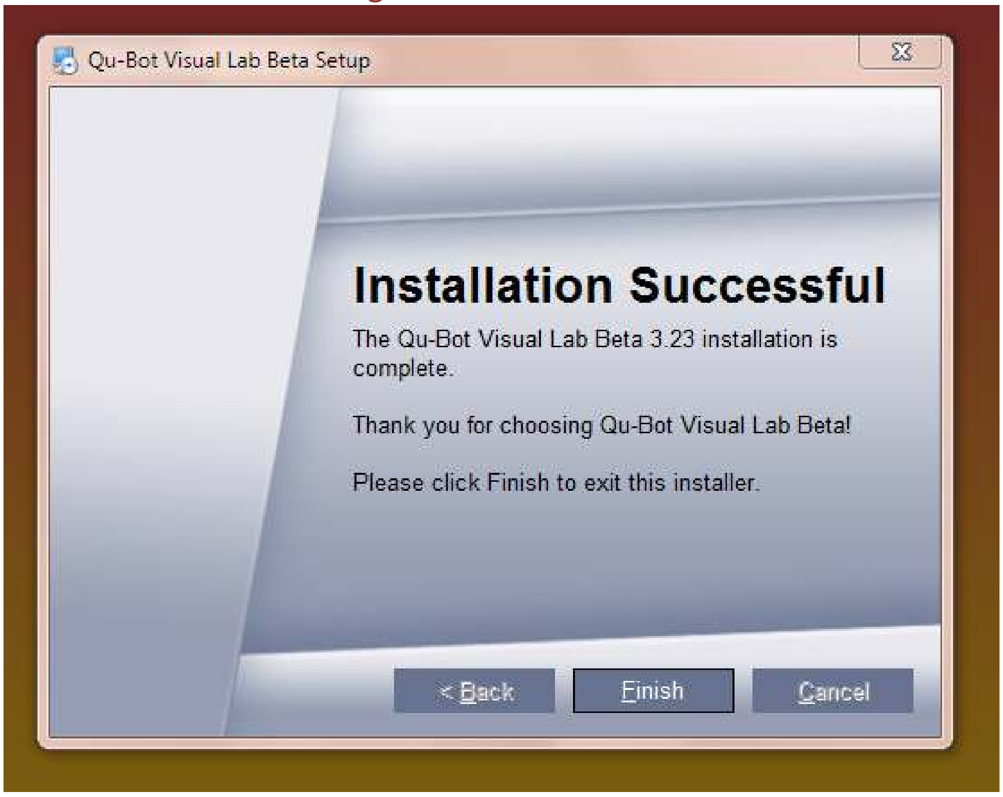
Installing Qu-Bot Visual Lab Beta

Please wait...

Installing Files...

Seeking...

Within few seconds the software gets installed.


Qu-Bot Visual Lab Beta Setup

Installation Successful

The Qu-Bot Visual Lab Beta 3.23 installation is complete.

Thank you for choosing Qu-Bot Visual Lab Beta!

Please click Finish to exit this installer.

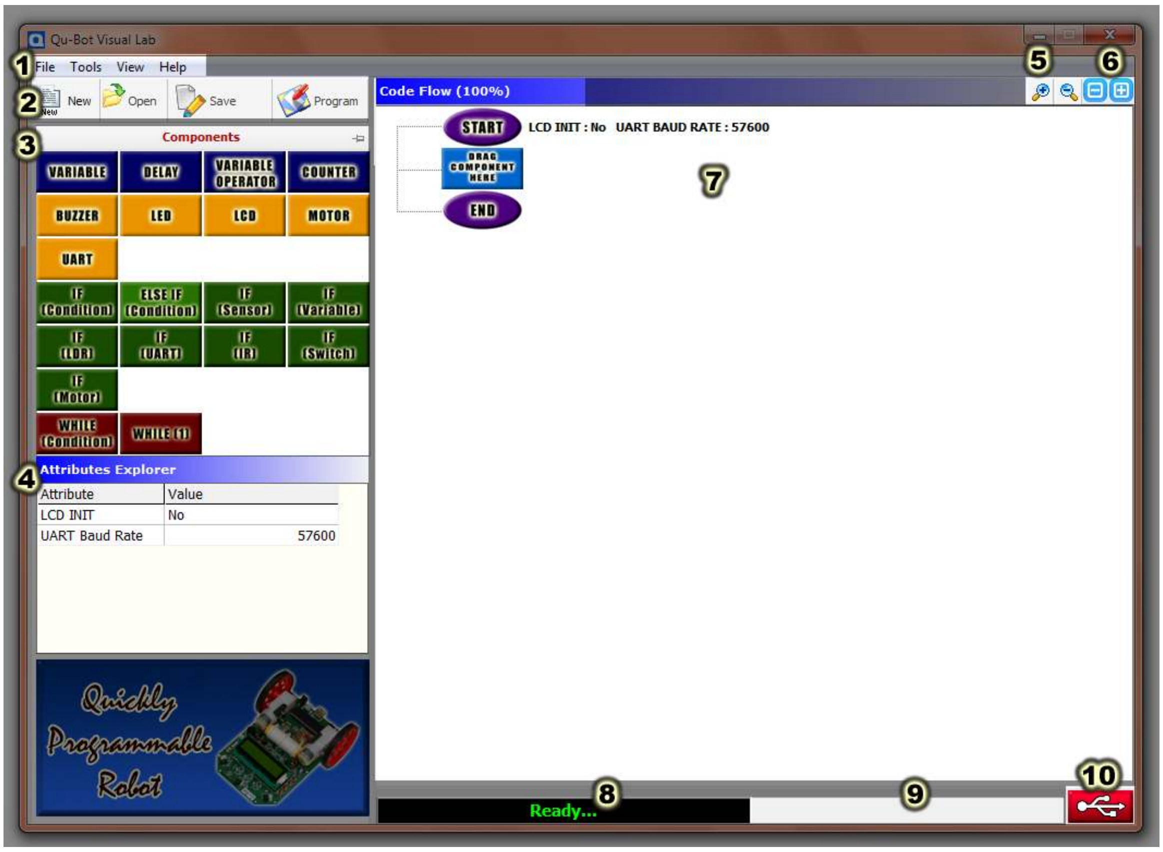
Now the software is installed. Press finish to exit installer. The QBVLab icon now appears on desktop as well as start menu. Run it to start the software.

3. Basic usage of Qu-Bot Visual Lab V3.23 (QBVLAB)

3.1. Qu-Bot Visual Lab Structure

3.1.1. Screen display

After successful installation open the QBVLab software. The software will look as below.


Qu-Bot Visual Lab
File Tools View Help
New Open Save Program
Components
VARIABLE DELAY VARIABLE OPERATOR COUNTER
BUZZER LED LCD MOTOR
UART
IF (Condition) ELSE IF (Condition) IF (Sensor) IF (Variable)
IF (LDR) IF (UART) IF (IR) IF (Switch)
IF (Motor)
WHILE (Condition) WHILE (1)
Attributes Explorer
Code Flow (100%)
START
DRAG COMPONENT HERE
END
LCD INIT : No UART BAUD RATE : 57600
Ready...
Quickly Programmable Robot
1
2
3
4
5
6
7
8
9
10

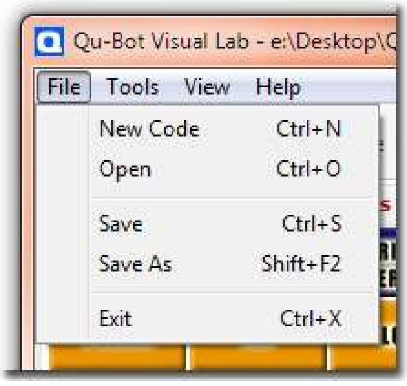
Attribute	Value
LCD INIT	No
UART Baud Rate	57600

Below is the list of basic components marked by numbers in the screenshot.

1. Menu bar: Contains File, Tools, View and Help.
2. Quick menu bar: Contains New, Load, Save and Program options.
3. Components window: Contains operational, output and conditional blocks.
4. Attribute explorer: Indicates component block properties.
5. Zoom out & Zoom in: Shown only if further zoom in or zoom out is possible.
6. Collapse all and expand all buttons: Expands and collapse all branches.
7. Code flow window: Space for programming Qu-Bot
8. Status bar: Shows status of programming
9. Progress bar: Shows programming progress
10. USB status: Switches between green and red while connected to Qu-Bot.

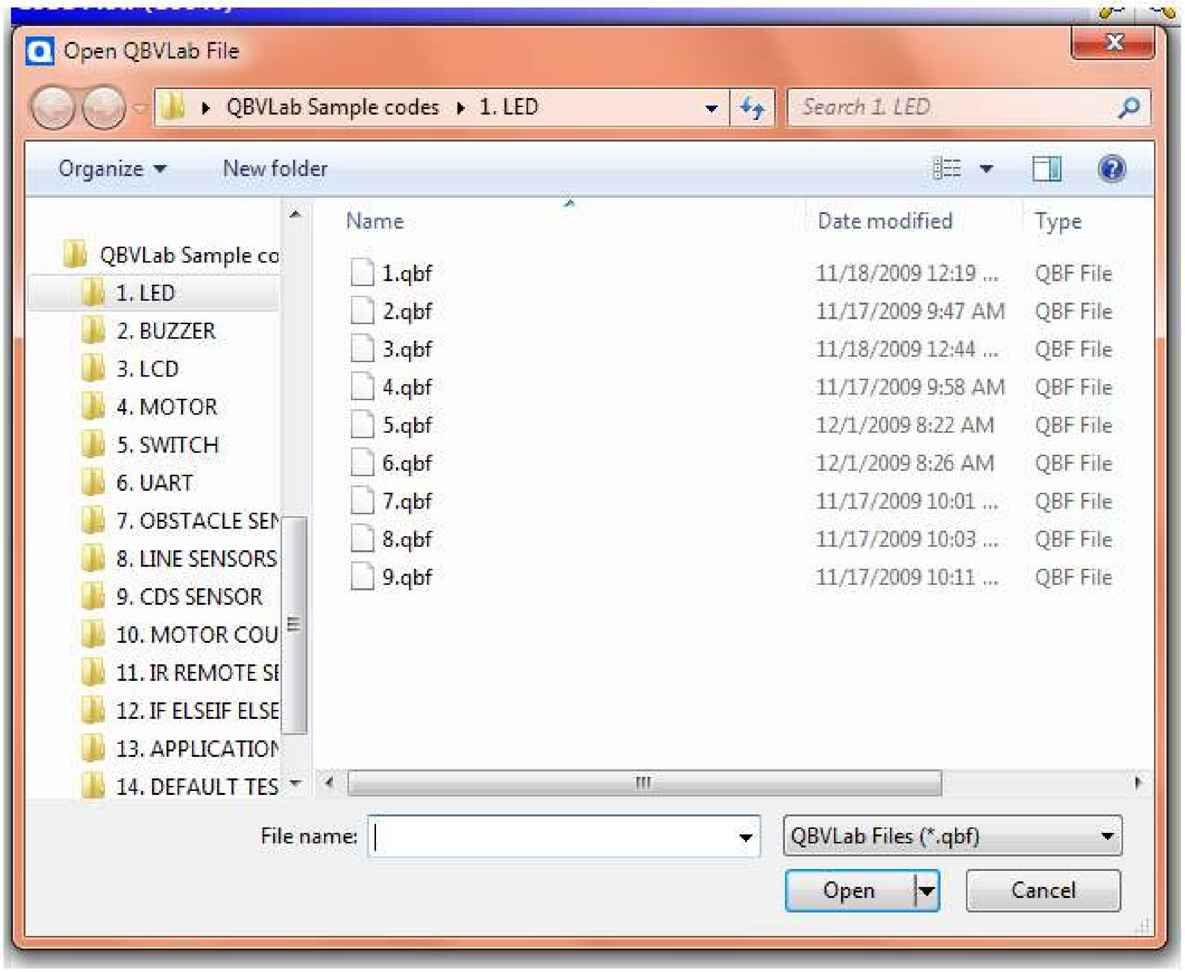
3.1.2. Menu Bar

3.1.2.1. File



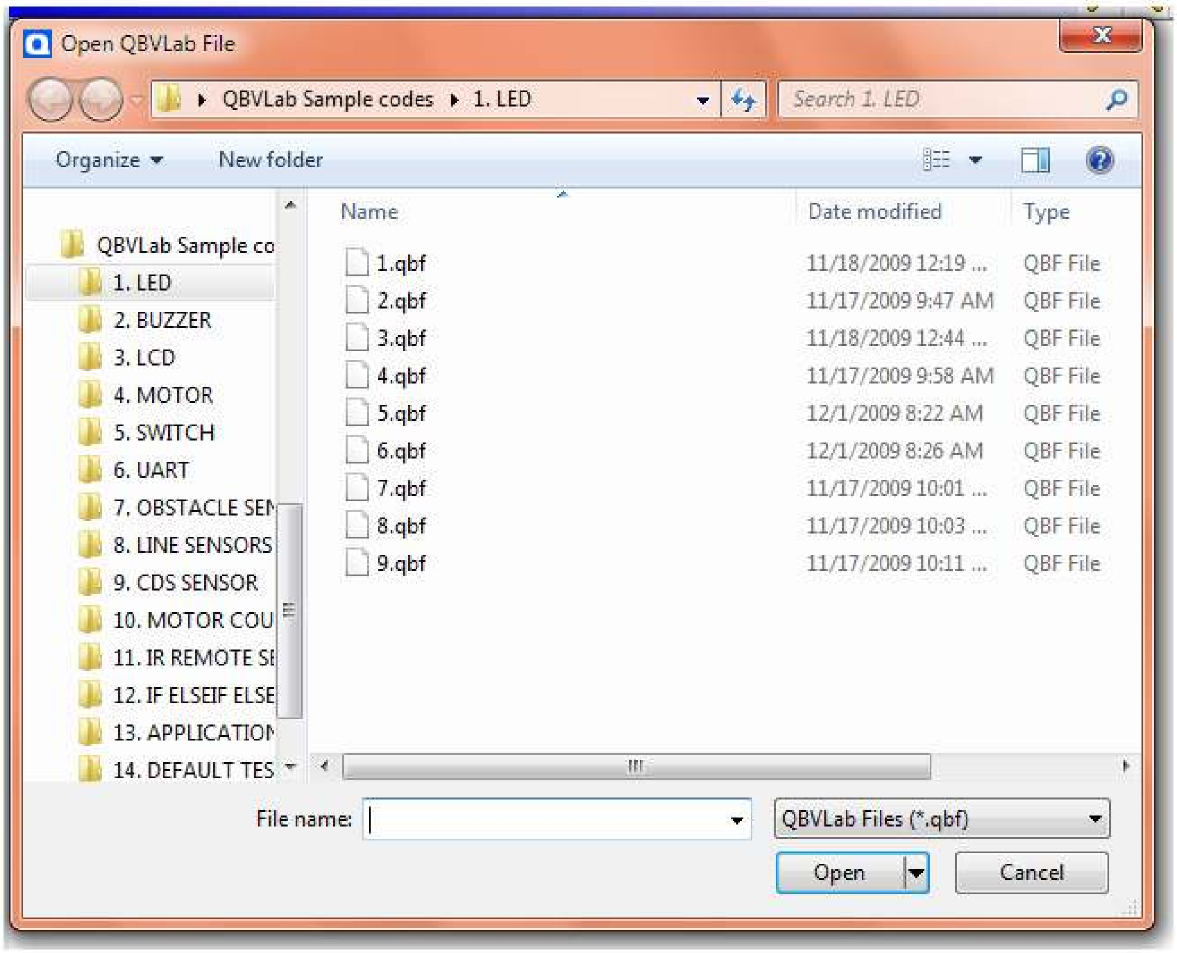
3.1.2.1.1. New Code (Ctrl+N): Creates a new blank file. Click the "New" button on quick menu bar or "File-> New Code".

3.1.2.1.2. Open (Ctrl+O): Opens a previously saved file. Click "Open" button on quick menu bar or "File-> Open". Then select ".QBF" file and click open.



Name	Date modified	Type
1.qbf	11/18/2009 12:19 ...	QBF File
2.qbf	11/17/2009 9:47 AM	QBF File
3.qbf	11/18/2009 12:44 ...	QBF File
4.qbf	11/17/2009 9:58 AM	QBF File
5.qbf	12/1/2009 8:22 AM	QBF File
6.qbf	12/1/2009 8:26 AM	QBF File
7.qbf	11/17/2009 10:01 ...	QBF File
8.qbf	11/17/2009 10:03 ...	QBF File
9.qbf	11/17/2009 10:11 ...	QBF File

3.1.2.1.3. Save (Ctrl+S): Saves a file. Click the "Save" button on quick menu bar or "File-> Save". Once you see following window add proper name to the file and click save. There is no need of adding ".QBF" extension.

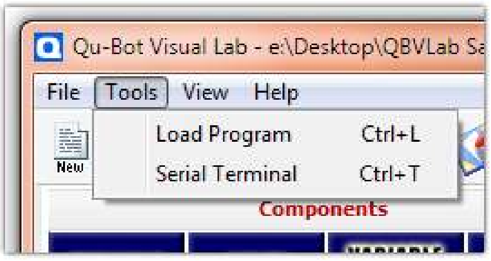


Name	Date modified	Type
1.qbf	11/18/2009 12:19 ...	QBF File
2.qbf	11/17/2009 9:47 AM	QBF File
3.qbf	11/18/2009 12:44 ...	QBF File
4.qbf	11/17/2009 9:58 AM	QBF File
5.qbf	12/1/2009 8:22 AM	QBF File
6.qbf	12/1/2009 8:26 AM	QBF File
7.qbf	11/17/2009 10:01 ...	QBF File
8.qbf	11/17/2009 10:03 ...	QBF File
9.qbf	11/17/2009 10:11 ...	QBF File

3.1.2.1.4. Save As (Shift+F2): Similar to "Save". Saves file in different name Click "File-> Save As".

3.1.2.1.5. Exit (Ctrl+X): Exit from QBVLab.

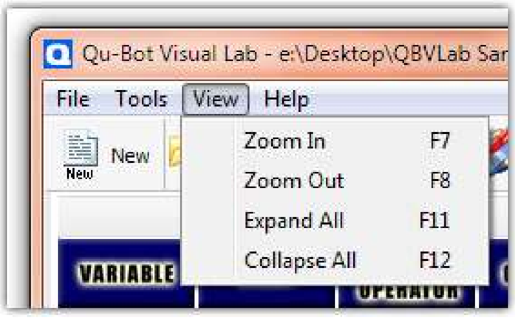
3.1.2.2. Tools



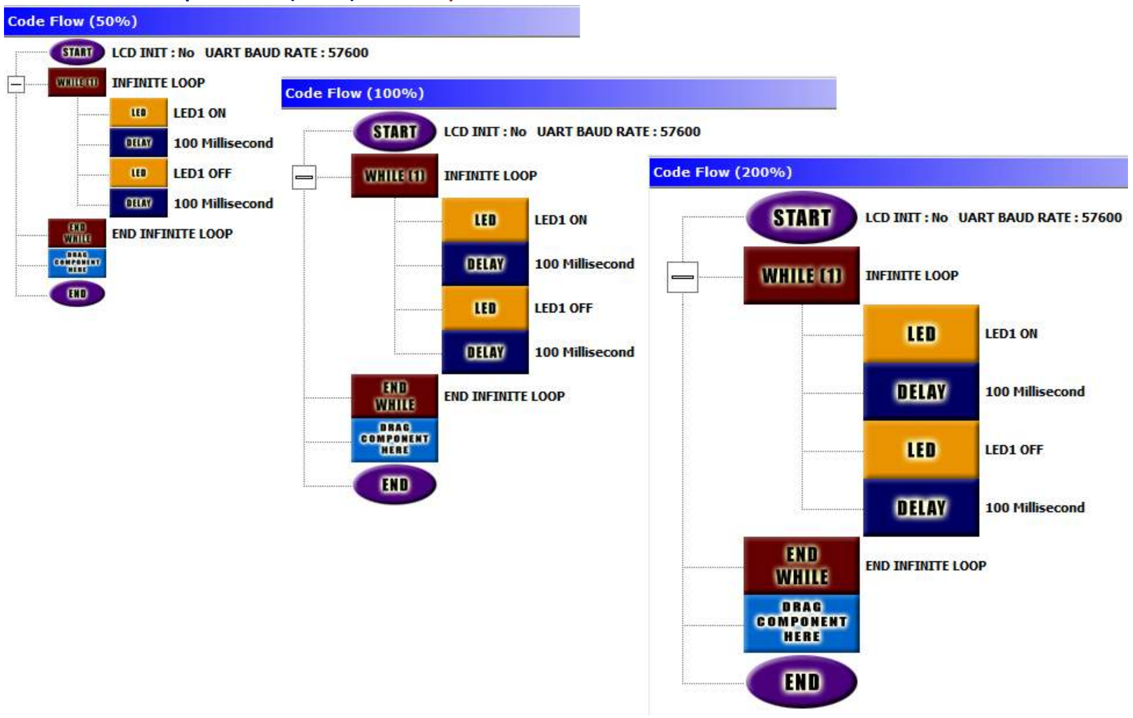
3.1.2.2.1. Load Program (Ctrl+L): Program written in code flow is transferred to Qu-Bot via USB Cable. Please connect USB Cable before programming.

3.1.2.2.2. Serial Terminal (Ctrl+T): Opens Serial Terminal window from which you can send and receive UART commands.

3.1.2.3. View

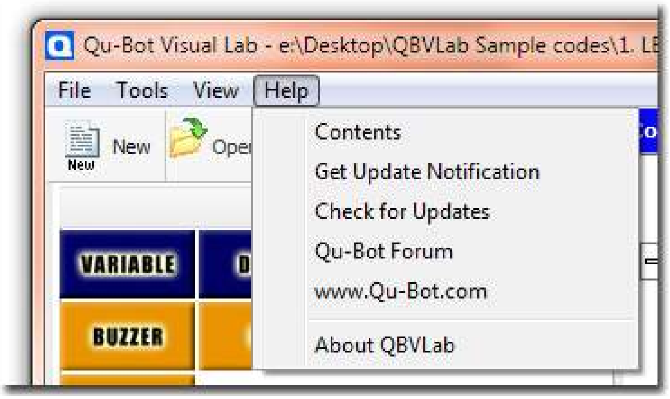


3.1.2.3.1. Zoom In (F7): Makes All icons smaller in size.
3.1.2.3.2. Zoom Out (F8): Makes All icons smaller in size.
3.1.2.3.3. Expand All (F11): Expands All Branches.
3.1.2.3.4. Collapse All (F12): Collapse all Branches.



Zoom In & Zoom Out

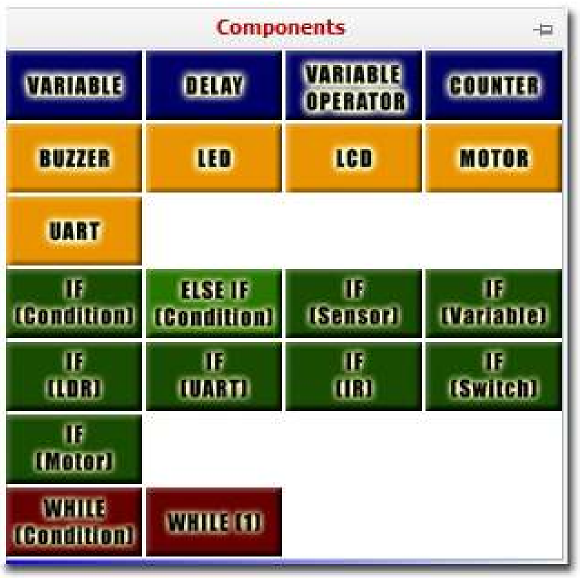
3.1.2.4. Help



3.1.2.4.1. Contents: Browse Help.

3.1.2.4.2. Get Update Notification: Register your email id for notification of new QBVLab software update.
3.1.2.4.3. Check for Updates: Check for current latest QBVLab Version.
3.1.2.4.4. Qu-Bot Forum: Are you facing any problems?? Visit forums for solutions.
3.1.2.4.5. www.Qu-Bot.com: Visit website www.Qu-Bot.com.
3.1.2.4.6. About QBVLab: Shows credits, copyright and version info.

3.1.3. Component Blocks



Depending upon the use of the component blocks they are divided in five different categories. Start & End logic blocks are already added to the code flow which indicates start and end of Qu-Bot Program. These blocks can't be dragged from component window.

1. Output blocks (Yellow)
2. Operational blocks (Blue)
3. Conditional blocks (Green)
4. Loop blocks (Red)
5. Start and End blocks (Purple)

3.1.3.1. Output blocks (Yellow)

BUZZER	Controls Buzzer on, off and toggle.
LED	Controls LED1, 2 and 3 on, off and toggle.
LCD	Displays Text & Variables on 16X2 LCD.
MOTOR	Motors operate according to speed and direction input value.
UART	Displays text and variables on serial terminal window. Used for giving data to PC through USB.

3.1.3.2. Operational blocks (Blue)

VARIABLE	Used for Initialization of variables. E.g. Variable=Constant Value Or Variable=Variable. **Note: In case of UART_Input & IR_Input refer section 7.3.1.**

DELAY	Generates Delay in Microseconds, Milliseconds and Seconds.
VARIABLE OPERATOR	Operates on more than One Operands. One of the operands must be Variable and second operand can be either variable or constant. Operators available are (+,-,*, /). **E.g.** **1. Variable1 = Variable2 – Variable3** **2. Variable1 =Variable2 * Constant** Notes: 1) Various data type variables can be used in one operation. 2) Value will be limited as per the data type of Variable1 in above example operations. 3) Constant value in e.g. limited to 0-255.
COUNTER	Increment or decrement counter variable.

3.1.3.3. Conditional blocks (Green)

IF (Condition)
ELSE IF (Condition)
ELSE
END IF
	1. Split code flow depending upon the condition. If condition is true execute IF part and then go to step5.
	2. Check for next ELSEIF or ELSE block.
	3. If ELSEIF is encountered check condition. If condition is true execute ELSEIF part and then go to step5. Else go to step2.
	4. If ELSE is encountered execute else part.
	5. Execute code flow below ENDIF.
	
	In this conditional blocks you can write multiple conditions having OR & AND logic operations through Condition builder window.
	***All following functions can be done using this IF(Condition) master block.**

IF (Sensor)	Branch depending upon the Front obstacle sensors or line follower sensors.
IF (Variable)	Branch depending upon variable value.
IF (LDR)	Branch depending upon LDR_Vaue Variable.

IF (UART)	Branch depending upon UART input Variable.
IF (IR)	Branch depending upon input IR Code.
IF (Switch)	Branch depending upon the Switch1, 2 & 3 status.
IF (Motor)	Branch depending upon the Motor counter (Encoder) Variables.

3.1.3.4. Loop blocks (Red)

WHILE (1)	Infinite loop.
WHILE (Condition)	Conditional Loop. Loop until condition is true.

3.1.3.5. START & END (Purple)

START	Indicates start of Program. LCD & UART properties can be set from this block.
END	Indicates End of program. Three short beeps will indicate program execution is ended on Qu-Bot.

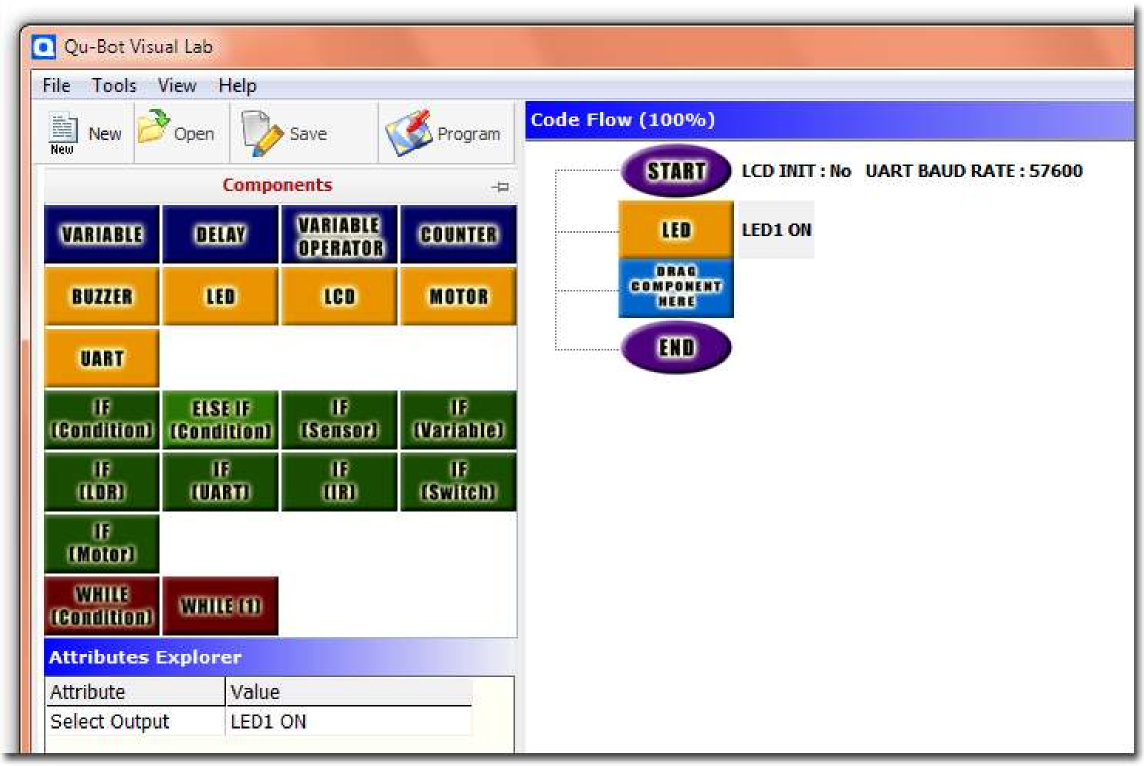
3.2. Programming Qu-Bot in QBVLab

3.2.1. How to Drag & Drop Component Blocks??

The component block is dragged until the click is pressed. Simply click on any of the component block you want to drag from component window. Drag it to the position you want in the code flow & release click on mouse to drop the component block. If you have dragged the component block to the correct place in code flow window then the component block will be added in the code flow at that position. (Try to move LED block in the code flow.)

3.2.2. How to set attributes of any component block

All component blocks can be set to do different functions by choosing attributes of the block from attribute explorer window. To set or change attributes of the component block please click on the component block that is added in code flow. Once you click on the component block you will see all the attributes of that component block in attribute explorer window. Once attributes are set or changed the attributes will appear on the right side of that block. (Set attributes of LED block to LED1 ON. Now your program will look as shown in the image below.)



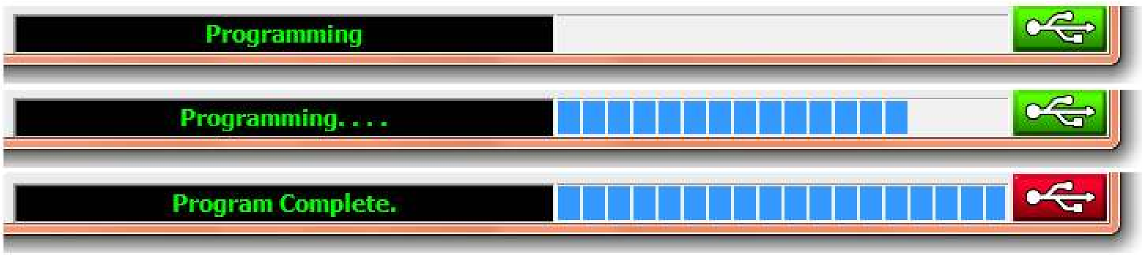
LCD INIT : No UART BAUD RATE : 57600

LED1 ON

Attribute	Value
Select Output	LED1 ON

3.2.3. Programming Qu-Bot

Check that you have connected USB cable to your Qu-Bot. Once you have built program as above click on program. Check Status bar for programming status. Once you get programming complete on status bar and USB Status symbol Red you can check your Qu-Bot. Do not disconnect USB cable while USB status is green. LED1 will be ON. Also you will hear three short beeps from buzzer which indicates END of program.



Programming status indication

3.2.4. Deleting block from code flow window.

To delete any block and right click on the block and select delete from the menu.

***All sub Items that are shown with indent from that block will be deleted automatically.**

4. Programming in Qu-Bot Visual LAB (Output Blocks)

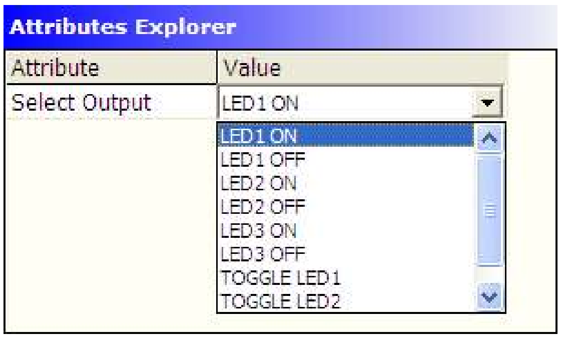
In this chapter we will learn how to use blocks listed under Section 3.1.3.1.

4.1. Lighting LED's

4.1.1. Characteristics of LED Block

LED Block controls on, off and toggle status of LED's. There are in total 3 LED's on Qu-Bot which can be separately operated by this module. LED's are indicated by number in figure2.1.

4.1.2. Changing Attributes



Attributes Explorer

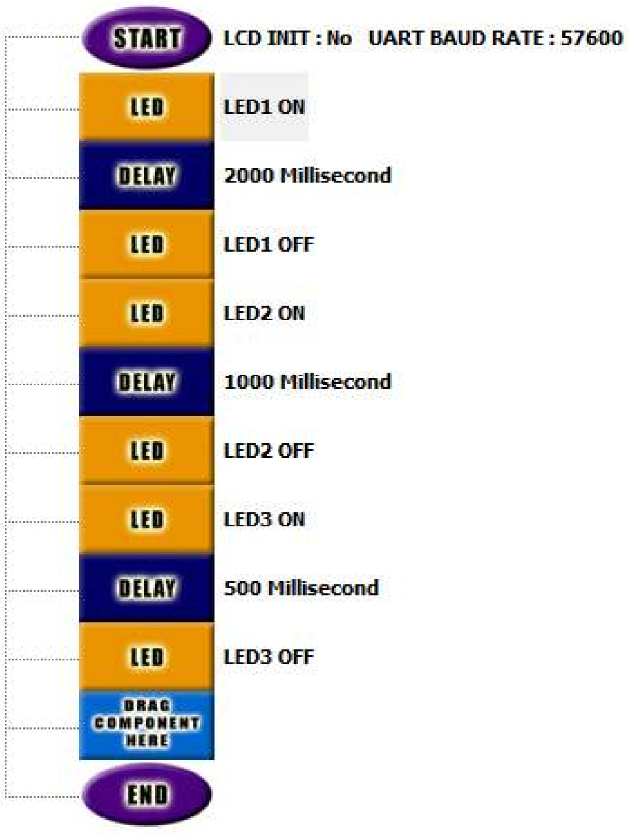
Attribute	Value
Select Output	LED1 ON

Left Click on the LED block in code flow to see attributes.

"Select Output": Left click on value block to see drop down menu.

For Each LED there are three options. Total options available are LED1 ON, LED1 OFF, LED2 ON, LED2 OFF, LED3 ON, LED3 OFF, TOGGLE LED1, TOGGLE LED2, and TOGGLE LED3.

4.1.3. LED On/Off Sequence



Block	Attributes
START	LCD INIT : No UART BAUD RATE : 57600
LED	LED1 ON
DELAY	2000 Millisecond
LED	LED1 OFF
LED	LED2 ON
DELAY	1000 Millisecond
LED	LED2 OFF
LED	LED3 ON
DELAY	500 Millisecond
LED	LED3 OFF
DRAG COMPONENT HERE	
END	

Arrange the block and set attributes as shown in the figure.

Check section 7.5 for setting delay block attributes.

The program will execute as below when downloaded to Qu-Bot.

LED1 will be on for 2 Seconds.

LED1 will be off and LED2 will be on for 1 second.

LED2 will be off and LED3 will be on for 500mS.

LED3 will be off.

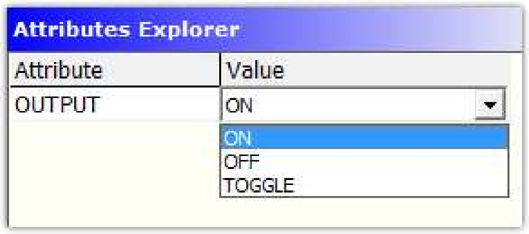
Buzzer will beep thrice indicating end of program.

4.2. Ringing buzzer

4.2.1. Characteristics of Buzzer Block

Buzzer Block controls on, off and toggle status of Buzzer.

4.2.2. Changing Attributes



Attributes Explorer

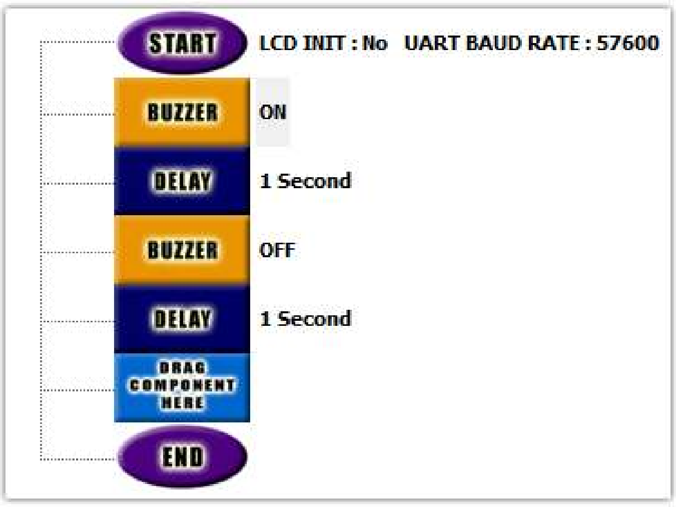
Attribute	Value
OUTPUT	ON

Left Click on the Buzzer block in code flow to see attributes.

"Output": Selects buzzer output.

For Buzzer there are three options ON, OFF and TOGGLE.

4.2.3. Ringing Buzzer for 1 second



START	LCD INIT : No UART BAUD RATE : 57600
BUZZER	ON
DELAY	1 Second
BUZZER	OFF
DELAY	1 Second
DRAG COMPONENT HERE	
END	

Arrange the blocks and set attributes as shown in the figure.
The program will execute as below when downloaded to Qu-Bot.
Buzzer will be on for 1 second.
Buzzer will be off for 1 second.
Buzzer will beep thrice indicating end of program.

4.3. Writing to LCD

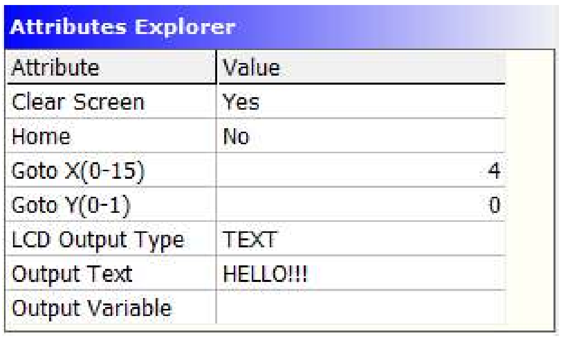
4.3.1. Characteristics of LCD Block

LCD Block controls 16X2 Liquid Crystal Display on Qu-Bot. LCD is detachable from Qu-Bot hence you can select whether to use LCD or not from Start Block attribute. Set "LCD Init" attribute to "Yes" to use LCD.

***If you have not initialized LCD through start block attributes explorer then LCD block cannot be dragged from Component window to code flow window.**

***If you have initialized LCD through start block attributes explorer and you have not connected LCD to Qu-Bot, your program may work erratically.**

4.3.2. Changing Attributes

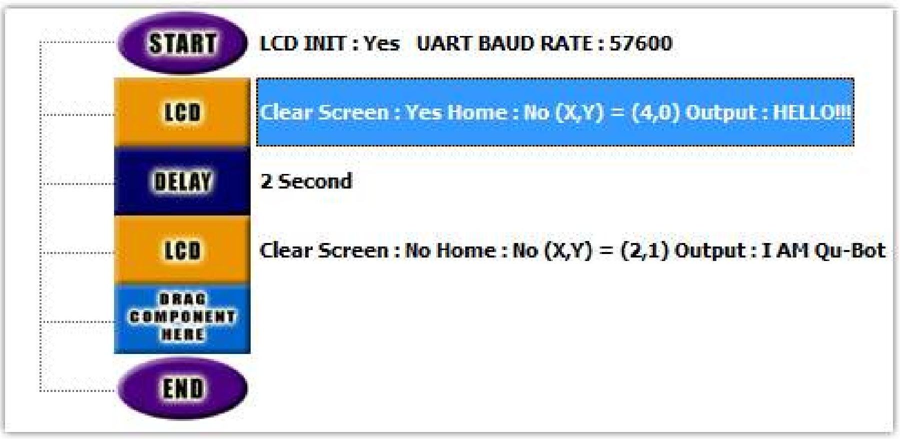


Attributes Explorer

Attribute	Value
Clear Screen	Yes
Home	No
Goto X(0-15)	4
Goto Y(0-1)	0
LCD Output Type	TEXT
Output Text	HELLO!!!
Output Variable	

Left Click on the LCD block in code flow to see attributes.
"Clear Screen": Yes, No (Clear LCD and move cursor to First line and first column)
"Home": Yes, No (Move cursor to First line and first column)
"Goto X (0-15)": Go to column X. X=0 1st column.
"Goto Y (0-1)": Go to Row Y. Y=0 1st row.
"LCD Output Type": Text, Variable
"Output Text": Active only when "LCD Output Type" value is "Text". The text value is written to LCD at current cursor position.
"Output Variable": Active only when "LCD Output Type" value is "Variable". The variable value is written to LCD at current cursor position.

4.3.3. Display Text on LCD



LCD INIT : Yes UART BAUD RATE : 57600
Clear Screen : Yes Home : No (X,Y) = (4,0) Output : HELLO!!!
2 Second
Clear Screen : No Home : No (X,Y) = (2,1) Output : I AM Qu-Bot

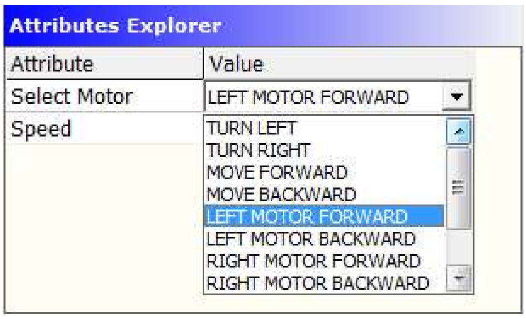
Arrange the blocks and set attributes as shown in the figure.
The program will execute as below when downloaded to Qu-Bot.
Clear LCD.
Move cursor to Column 5 Row 1.
Print text "Hello!!!" to LCD.
Wait for 2 seconds.
Go to Column 3 Row 2.
Print "I AM Qu-Bot"
Buzzer will beep thrice indicating end of program.

4.4. Moving Qu-Bot

4.4.1. Characteristics of Motor Block

Motor Block Controls direction and speed of motors individually or combined.

4.4.2. Changing Attributes


Attributes Explorer

Attribute	Value
Select Motor	LEFT MOTOR FORWARD
Speed	

TURN LEFT
TURN RIGHT
MOVE FORWARD
MOVE BACKWARD
LEFT MOTOR FORWARD
LEFT MOTOR BACKWARD
RIGHT MOTOR FORWARD
RIGHT MOTOR BACKWARD

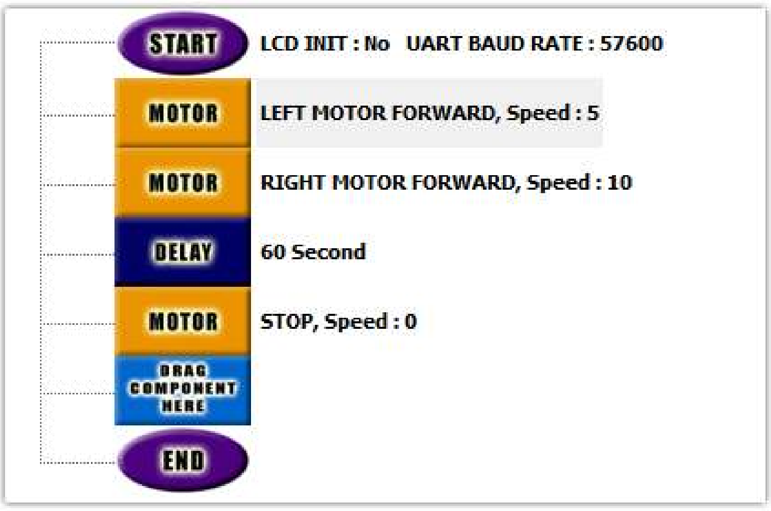
Left Click on the Motor block in code flow to see attributes.

"Select Motor": Selects motor and motor Direction.

Options in drop down menu are TURN LEFT, TURN RIGHT, MOVE FORWARD, MOVE BACKWARD, LEFT MOTOR FORWARD, LEFT MOTOR BACKWARD, RIGHT MOTOR FORWARD, RIGHT MOTOR BACKWARD, STOP, LEFT MOTOR STOP and RIGHT MOTOR STOP

"Speed": Speed of motor can be selected from drop down menu ranging from 0 to 10. 10 is maximum speed.

4.4.3. Moving Qu-Bot in a circle


START LCD INIT : No UART BAUD RATE : 57600
MOTOR LEFT MOTOR FORWARD, Speed : 5
MOTOR RIGHT MOTOR FORWARD, Speed : 10
DELAY 60 Second
MOTOR STOP, Speed : 0
DRAG COMPONENT HERE
END

Arrange the blocks and set attributes as shown in the figure.

The program will execute as below when downloaded to Qu-Bot.

Left motor will move at half speed of right motor for 60 seconds. Hence Qu-Bot moves in anti-clockwise circle for 60 seconds.

Motors will stop and Qu-Bot will stand still after 60 seconds.

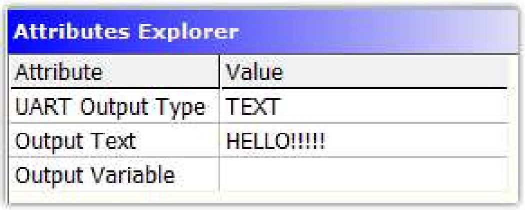
Buzzer will beep thrice indicating end of program.

4.5. Communicate with PC through UART

4.5.1. Characteristics of UART Block

UART Block is used for sending data to the PC. Data can be seen from Tools -> Serial Terminal (Ctrl+T).

4.5.2. Changing Attributes



Attributes Explorer

Attribute	Value
UART Output Type	TEXT
Output Text	HELLO!!!!!
Output Variable	

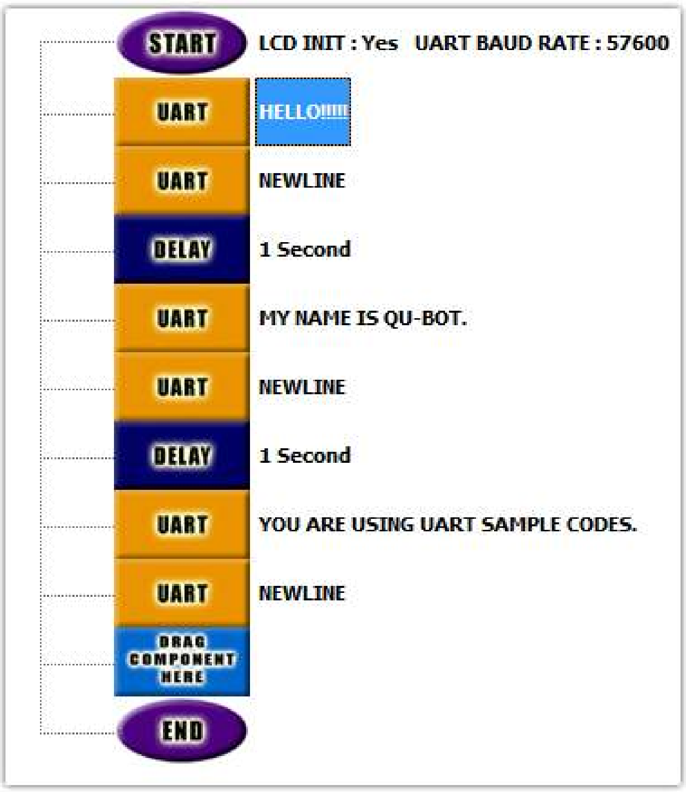
Left Click on the UART block in code flow to see attributes.

"UART Output Type": Selects output type between TEXT, VARIABLE and NEWLINE.

"Output Text": Send Text output to PC. Active only if "UART Output Type" is "Text".

"Output Variable": Send Variable Value as output to PC. Active only if "UART Output Type" is "Variable".

4.5.3. Display Text on Serial Terminal

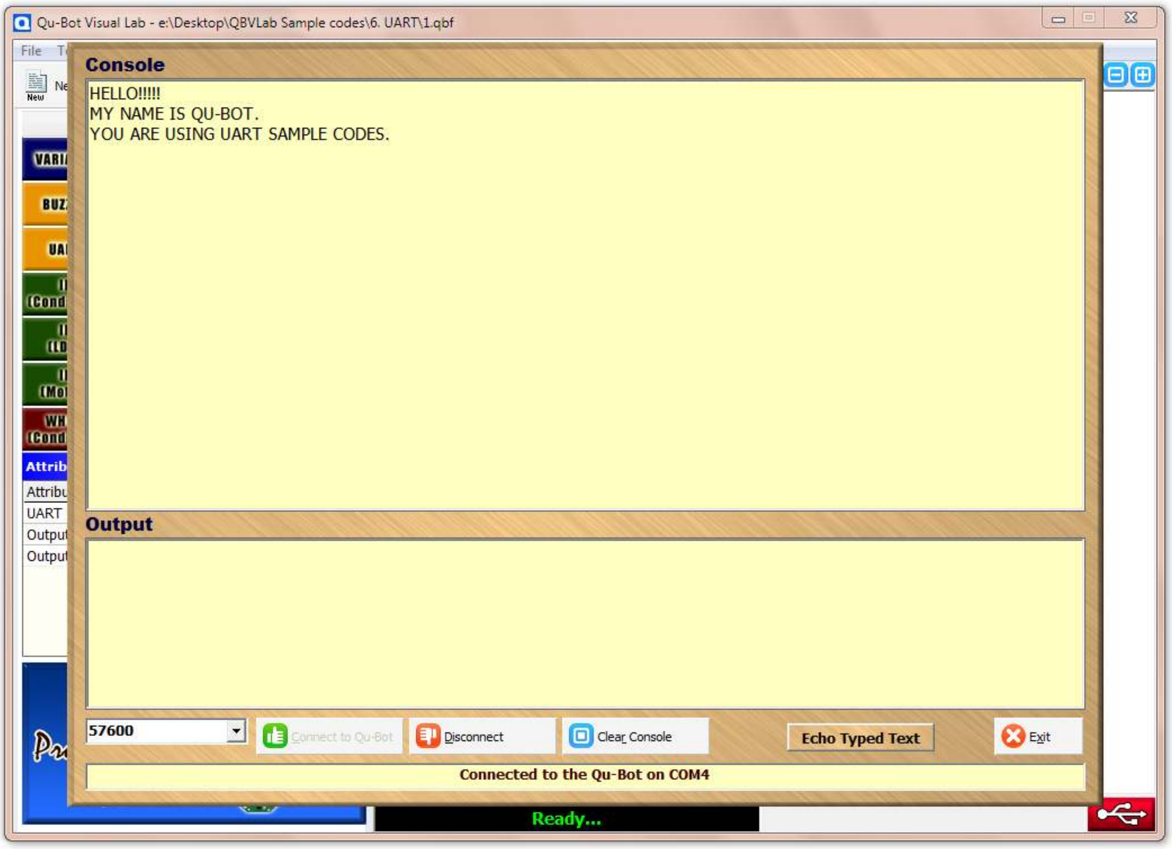


```
START   LCD INIT : Yes   UART BAUD RATE : 57600
UART    HELLO!!!!!
UART    NEWLINE
DELAY   1 Second
UART    MY NAME IS QU-BOT.
UART    NEWLINE
DELAY   1 Second
UART    YOU ARE USING UART SAMPLE CODES.
UART    NEWLINE
DRAG COMPONENT HERE
END
```

Arrange the blocks and set attributes as shown in the figure.
The program will execute as below when downloaded to Qu-Bot.
The execution of this program can be checked on serial terminal window.
Qu-Bot will print "HELLO!!!" and NEWLINE.
Wait for 1 Second.

Qu-Bot will print "MY NAME IS QU-BOT." and NEWLINE.
Wait for 1 second.
Qu-Bot will print "YOU ARE USING UART SAMPLE CODES" and NEWLINE.
Buzzer will beep thrice indicating end of program.

4.5.4. Using Serial Terminal to check output of above program



```
HELLO!!!!!
MY NAME IS QU-BOT.
YOU ARE USING UART SAMPLE CODES.
```

Connected to the Qu-Bot on COM4

Select Tools -> Serial terminal to start serial terminal.
Select proper Baud Rate, Click "Clear Console" and then click on "Connect to Qu-Bot".
You will see above output on console.

5. Programming in Qu-Bot Visual LAB (Loop Blocks)

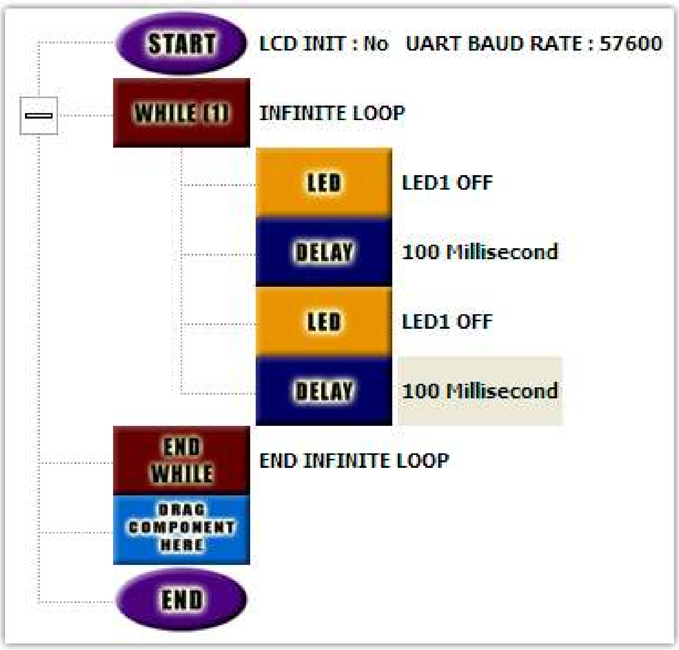
We frequently need to perform an action over and over, often with variations in the details each time. The mechanism, which meets this need, is the 'loop'. In this section we will learn how to use blocks listed under section 3.1.3.2.

5.1. WHILE(1) Infinite loop

5.1.1. Characteristics of WHILE(1) Block

This block generates an Infinite loop. Program will never reach beyond END WHILE block of WHILE(1). Program sequence between WHILE(1) and END WHILE will be repeated infinite times.

5.1.2. Blink LED1 forever



START LCD INIT : No UART BAUD RATE : 57600
WHILE (1) INFINITE LOOP
LED LED1 OFF
DELAY 100 Millisecond
LED LED1 OFF
DELAY 100 Millisecond
END WHILE END INFINITE LOOP
DRAG COMPONENT HERE
END

Above program will be executed as following.

First of all when code flow will encounter WHILE(1) block. The Qu-Bot will save the position of the next block.

Then it will execute the program normally and LED1 will be on and off for 100mS.

When code flow encounters ENDWHILE block it will return to the position saved by WHILE(1) block.

Hence the same sequence will be repeated endlessly.

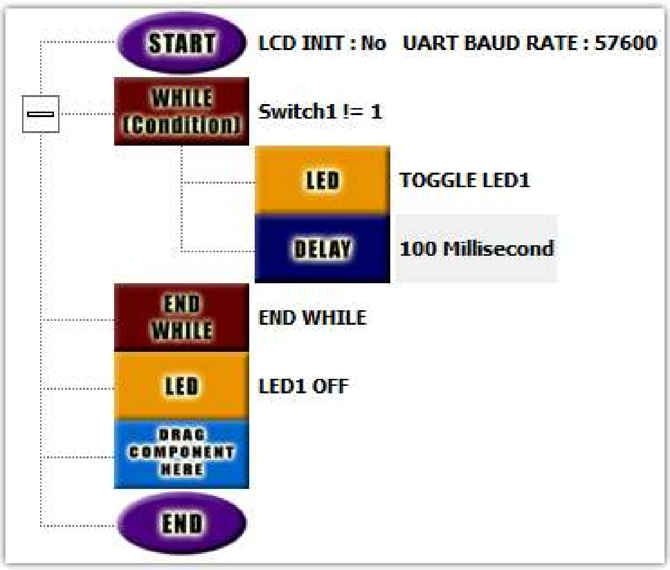
5.2. WHILE(Condition) Conditional loop

This section is just to introduce this conditional looping block. Please check section 6.1.2 for more details on condition building.

5.2.1. Characteristics of WHILE(Condition) Block

This block generates a loop where condition is checked first. If the condition is true the program sequence between WHILE(Condition) and END WHILE Blocks will be executed and the condition will be checked again. Until the condition is true the sequence between WHILE(Condition) and END WHILE Blocks will be repeated. If at any point of time condition is false the code flow will shift to next block of ENDWHILE block.

5.2.2. Blink LED1 until Switch1 is pressed



LCD INIT : No UART BAUD RATE : 57600
Switch1 != 1
TOGGLE LED1
100 Millisecond
END WHILE
LED1 OFF

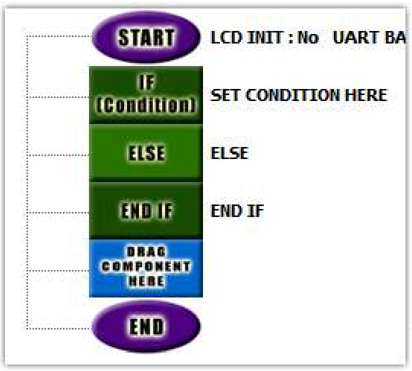
Above program will be executed as following.
First of all when code flow will encounter WHILE(Condition) block it will check for the condition. The condition is "Switch!=1" or "switch is not pressed".
If you have not pressed the switch the loop is just like a infinite WHILE(1) loop. Hence it will blink LED1.
Once you press Switch1 the code flow will move to WHILE(Condition) block within max 100ms and check condition.
***If you have put 5 seconds of delay instead of 100mS and you press and release switch1 within delay time the condition won't be checked and the loop won't break. Hence the LED1 will continue to blink. Try and check result yourself.**
Since the switch is pressed the condition will be false and code flow will move to block next to ENDWHILE which is a LED block with attributes "LED1 OFF".
Hence LED1 will be off and Buzzer will beep thrice indicating end of program.

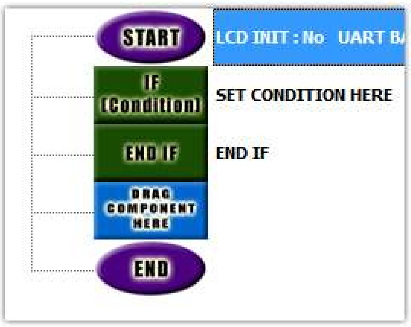
6. Programming in Qu-Bot Visual LAB (Conditional Blocks)

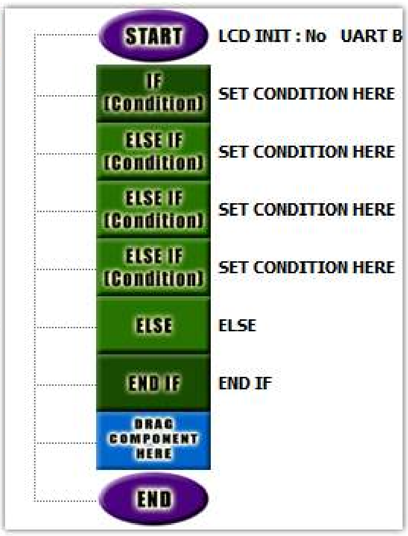
Sometimes we need to alter our actions in the face of changing circumstances. If there is an obstacle we need to turn our Qu-Bot right. You can notice that these decisions depend on some condition being met. We can check such conditions from these conditional blocks. In this chapter we will learn how to use blocks listed under Section 3.1.3.3.

6.1. IF(Condition), ELSE IF(Condition), ELSE & ENDIF blocks

6.1.1. Structure







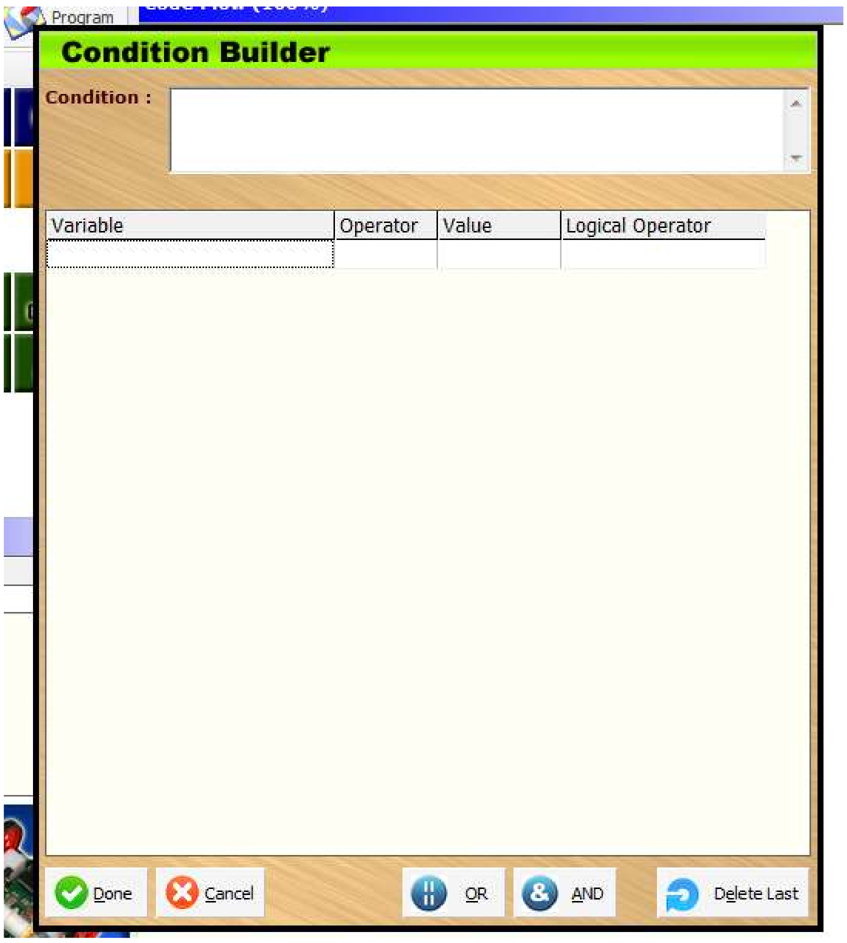
Once you drag IF(Condition) block the code flow window will look like left image. If you do not want ELSE block you can right click on the block and select delete (See 3.2.4). Then your code flow will look as shown in center image.

***If you delete ELSE from any IF structure you will not be able to restore it.**

ELSE IF(Condition) can be put between IF and ELSE Block. If there is no ELSE block it can be put anywhere in between IF(Condition) and ENDIF blocks. There can be multiple ELSE IF(Condition) blocks as shown in right figure. To add ELSE IF(Condition) block drag the block on IF(Condition) block.

6.1.2. Condition Building, Operands and Operators

Left click on IF(Condition) block to see attributes of this block. Click on Condition Value block in attribute explorer. You will see a condition builder window as shown below.



6.1.1.1. Operand1

There are four columns in the condition builder block. First of all click on Variable blank box to see drop down menu. Here you will see all sensor input variables. Also there are general purpose variables available which can be compared through this block. Check section 7.2.

Eight variables represent 3X line sensors, 3X Obstacle sensors and 2X Motor counter (Encoder) Values.

LDR Value variable represents ambient light value.

Switch1, Switch2 & Switch3 variables represent switch press status.

All the above variables fetch latest input from the system when they are used.

IR Input & UART Input variables also can be compared but they will need different structure. Please check section 7.3.1.

6.1.1.2. Operator

Second column in condition builder is used to select operator.

==	Checks equality
!=	Checks inequality
<	Less Than
>	Greater Than

<=	Less than or Equal to
>=	Greater than or Equal to

6.1.1.3. Operand2

Value is to be typed or selected by dropdown menu.

6.1.1.4. Logical operator

&&	Logical AND (Both condition must be true)
||	Logical OR (Any one condition should be true)

These operators are used between two conditions.

6.1.1.5. Condition structure

Condition structure will be **operand1, operator, and operand2.**

Operand1 will be compared to operand2 as per operator and condition result will be returned.

If you have used a Logical operator the structure will be **condition1, Logical Operator and condition2**.

There can be multiple conditions separated by logical operators.

Some examples of conditions

LDR Value < 100	Ambient light intensity Less than 100 (0-99)
BOTTOM CENTER == 1	Line follower center censor is on
Switch1 != 1	Switch1 not pressed
LEFT MOTOR COUNT > 1000	Left motor has moved more than 1000 counts forward
FRONT LEFT == 1	Obstacle left sensor senses obstacle
Switch1 == 1 AND Switch2 == 1 AND Switch3 == 1	All three switches are pressed
Switch1 == 1 OR Switch2 == 1 OR Switch3 == 1	Any switch is pressed
LDR Value < 50 OR LDR Value > 100	Ambient light intensity is between (0-49 & 101-255). Where 255 is max value.

6.1.3. Function of IF, ELSE IF, ELSE, ENDIF

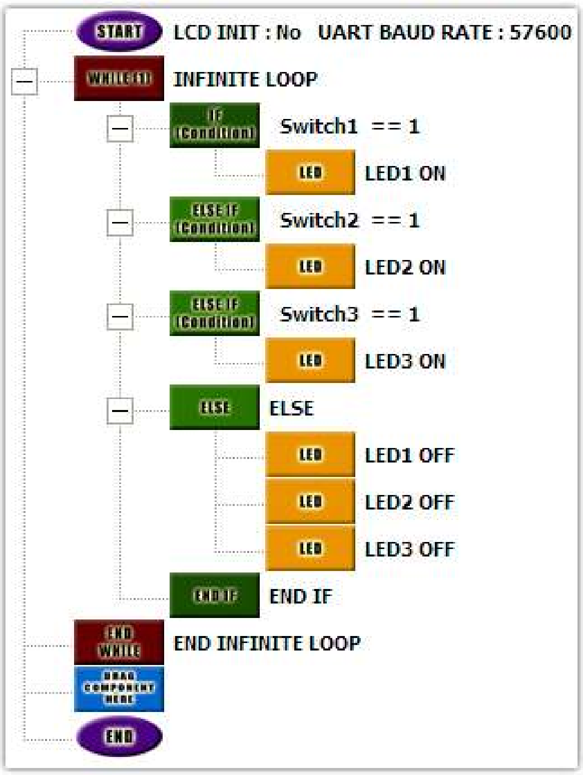
When code flow encounters IF(Condition) or ELSEIF(Condition) block it checks for the condition.

If condition is true code flow will execute items in that block and move directly to ENDIF.

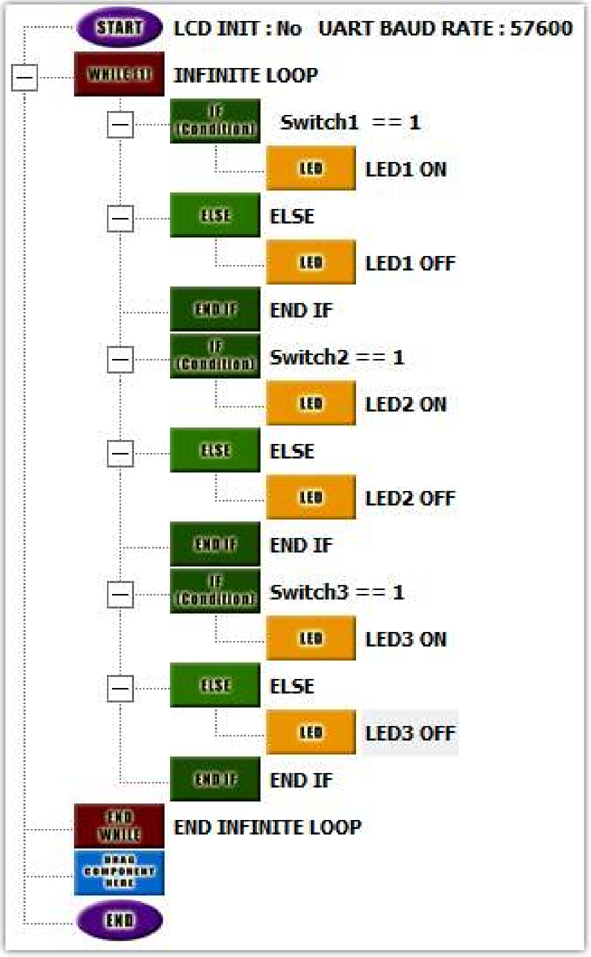
If condition is false it will search for next ELSEIF or ELSE block.

If none of the IF or ELSEIF condition is true and you have encountered ELSE then execute items in ELSE block.

6.1.4. Glow LED's as per switch press status



```
START      LCD INIT : No   UART BAUD RATE : 57600
WHILE(1)   INFINITE LOOP
    IF (Condition)        Switch1 == 1
        LED               LED1 ON
    ELSE IF (Condition)   Switch2 == 1
        LED               LED2 ON
    ELSE IF (Condition)   Switch3 == 1
        LED               LED3 ON
    ELSE                  ELSE
        LED               LED1 OFF
        LED               LED2 OFF
        LED               LED3 OFF
    END IF                END IF
END WHILE  END INFINITE LOOP
DRAG COMPONENT HERE
END
```



```
START      LCD INIT : No   UART BAUD RATE : 57600
WHILE(1)   INFINITE LOOP
    IF (Condition)   Switch1 == 1
        LED          LED1 ON
    ELSE             ELSE
        LED          LED1 OFF
    END IF           END IF
    IF (Condition)   Switch2 == 1
        LED          LED2 ON
    ELSE             ELSE
        LED          LED2 OFF
    END IF           END IF
    IF (Condition)   Switch3 == 1
        LED          LED3 ON
    ELSE             ELSE
        LED          LED3 OFF
    END IF           END IF
END WHILE  END INFINITE LOOP
DRAG COMPONENT HERE
END
```

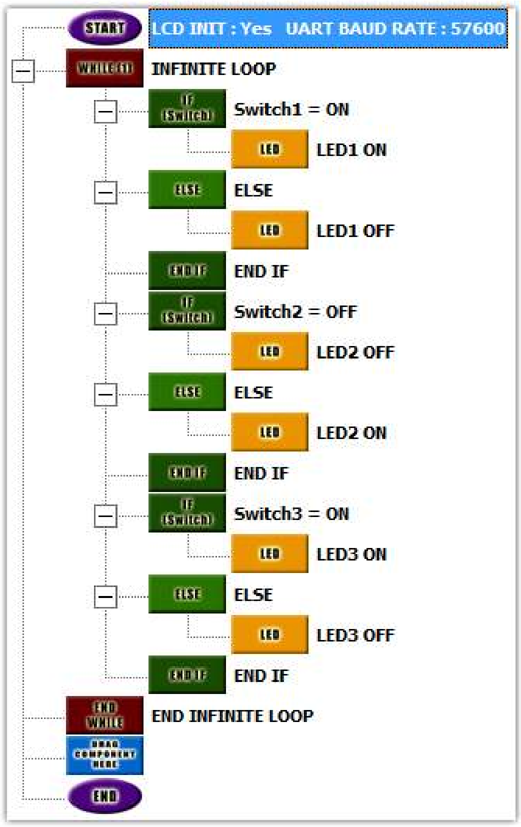
There are two programs written for glowing LED's on Qu-Bot depending upon the switch press.
IF blocks are used within an infinite loop.

From first sight it looks that both the programs are same. But there is functional difference. Find out what is the difference?
Hint: Press more than one switch and release only switch.

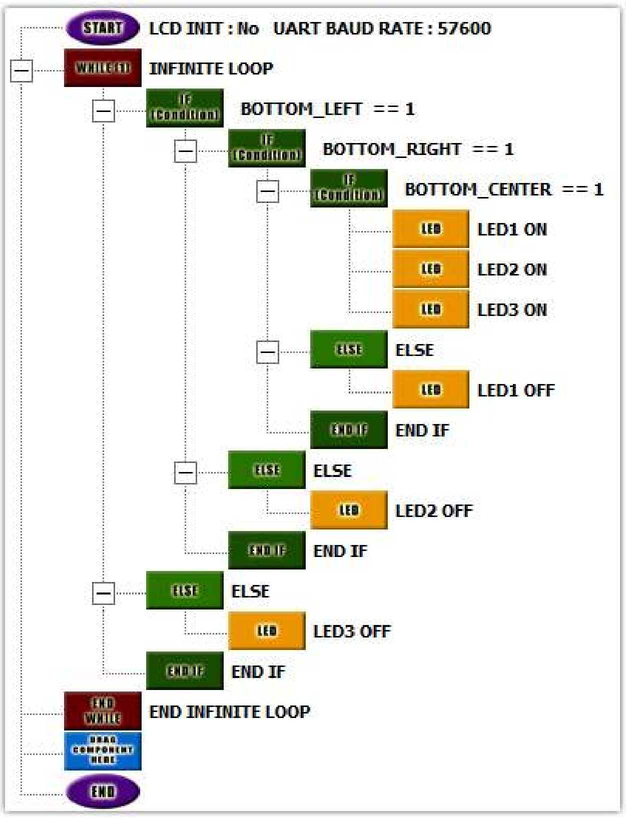
6.2. Other IF blocks

Right side program in section 6.1.4 can also be written through IF(Switch) block as shown in above figure. Also it will be much faster.

***In all IF() blocks you can add ELSE IF(Condition) block.**



```
LCD INIT : Yes   UART BAUD RATE : 57600
INFINITE LOOP
    Switch1 = ON
        LED1 ON
    ELSE
        LED1 OFF
    END IF
    Switch2 = OFF
        LED2 OFF
    ELSE
        LED2 ON
    END IF
    Switch3 = ON
        LED3 ON
    ELSE
        LED3 OFF
    END IF
END INFINITE LOOP
```



```
LCD INIT : No   UART BAUD RATE : 57600
INFINITE LOOP
    BOTTOM_LEFT  == 1
        BOTTOM_RIGHT  == 1
            BOTTOM_CENTER  == 1
                LED1 ON
                LED2 ON
                LED3 ON
            ELSE
                LED1 OFF
            END IF
        ELSE
            LED2 OFF
        END IF
    ELSE
        LED3 OFF
    END IF
END INFINITE LOOP
```

6.3. Nested IF ELSE sample

IF block inside IF or ELSE Block generates nested IFELSE. Above right side image shows three levels of nesting and the same program should clarify your all doubt of IF ELSE structure.

Press all three switches, release them in different sequences and check LED output.

7. Programming in Qu-Bot Visual LAB (Operational Blocks)

Many a times we need variables which change values depending upon the as the code progresses.

7.1. Data Types

There are three types of variables in Qu-Bot.

1. Character Variable : 0-255
2. Integer Variable: -327868 to 32767
3. Long Variables: -2147483648 to 2147483647

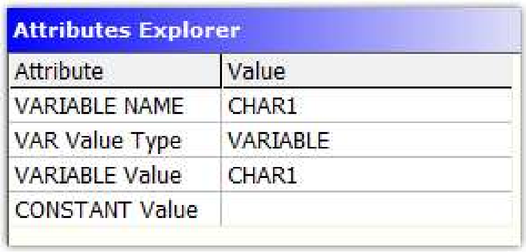
7.2. Variables in Qu-Bot

Following is the list of general purpose variables in Qu-Bot

1. 32 Character Variables: CHAR1 to CHAR32
2. 32 Integer Variables: INT1 to INT32
3. 10 Long Variables: LONG1 to LONG10
4. 10 Counter variables: CNT1 to CNT10 (Character Data Type)

7.3. Variable Block

This block is used to initialize variables. This block can initialize any variable in section 7.2 and UART Input and IR Input. Left click on the variable block in code flow and check attribute explorer.



Attributes Explorer

Attribute	Value
VARIABLE NAME	CHAR1
VAR Value Type	VARIABLE
VARIABLE Value	CHAR1
CONSTANT Value	

Variables can be initialized like blow.

1. Variable1 = Constant
2. Variable1 = Variable2

Select Variable1 from variable name dropdown box.
Select Value type from "Variable" or "Constant".
If you have selected Variable choose variable from the list or else type constant value.
Once the variables are initialized then only they will be populated in condition builder block variable list.

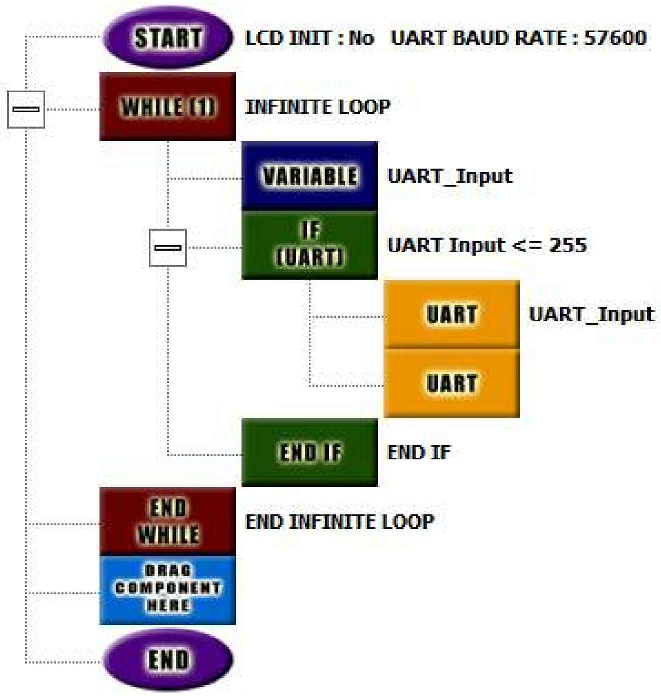
7.3.1. Special Cases

This block is used to initialize UART Input and IR Input. Select Variable Name as any one of them in attribute explorer window.
In this case rest of the blocks needs not to be initialized.
These variables are kept separate because of some reason.

7.3.1.1. UART Input

Refer section 4.5.4 where we used Serial Terminal. Here we have not used output window. This window can be used to transfer characters to Qu-Bot. Whenever you write something in the output box it will be transferred to Qu-Bot and it will be stored in a FIFO (First in First Out) buffer. This is done so that you don't loose any character while you do other stuff.

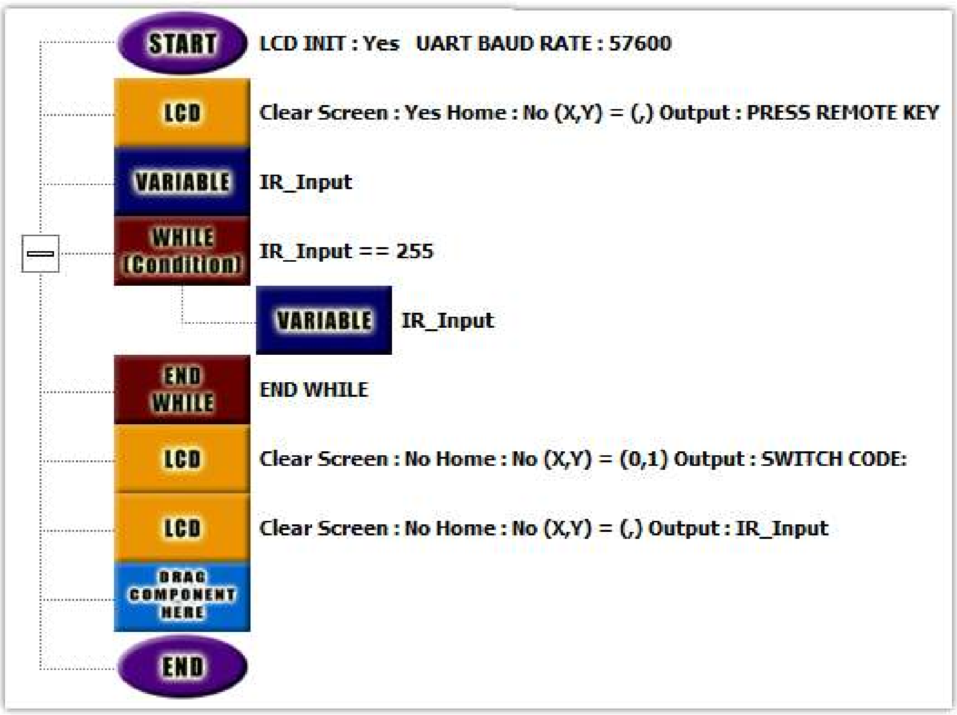
E.g. You typed "We are using UART" on output window when your program is processing delay. It will save all of above in FIFO. Once you use the above variable block once you will get UART Input='W' and FIFO="e are using UART". If used again UART Input='e' & FIFO = " are using UART".



Please check simple program and output of the same on UART console.

7.3.1.2. IR Input

IR Remote Sends signals every 114ms. Hence we cannot have data instantaneously like other sensors which provide data in 1-2uS. Hence the program flow waits for IR signal for around 200ms. Input is taken by Variable block just like in above case. If no IR input is received within this time IR Input variable will be 255 or you will receive different IR code for each key on remote in IR Input variable.

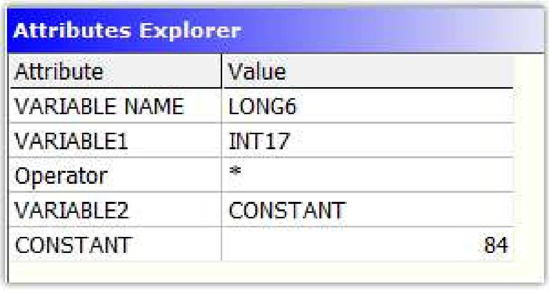


```
LCD INIT : Yes   UART BAUD RATE : 57600
Clear Screen : Yes Home : No (X,Y) = (,) Output : PRESS REMOTE KEY
IR_Input
IR_Input == 255
    IR_Input
END WHILE
Clear Screen : No Home : No (X,Y) = (0,1) Output : SWITCH CODE:
Clear Screen : No Home : No (X,Y) = (,) Output : IR_Input
```

Above program waits until it receives IR remote code and displays the code on LCD.

7.4. Variable Operator Block

This block is used to do mathematical operations like +, -, * and /. Left click on the Variable Operator block in code flow and check attribute explorer


Attributes Explorer

Attribute	Value
VARIABLE NAME	LONG6
VARIABLE1	INT17
Operator	*
VARIABLE2	CONSTANT
CONSTANT	84

This block can perform following actions

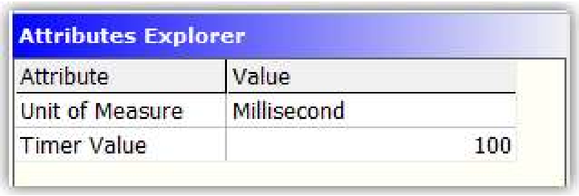
1. Variable = Variable1 (+, -, * or /) Constant
2. Variable = Variable1 (+, -, * or /) Variable2

The above image will multiply INT17 & 84 and store result in LONG6 variable (LONG6 = INT17 * 84). Also you can right the same statement by initializing any character variable to 84 using variable block and then operate as LONG6 = INT17 * CHAR30.

***Constant value must be between 0-255.**

7.5. Delay Block

This block is used to generate delay in sequence. We have already used this blocks for generating delays in previous programs. Left click on the Delay block in code flow and check attribute explorer.



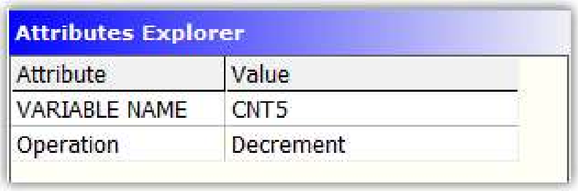
Attributes Explorer

Attribute	Value
Unit of Measure	Millisecond
Timer Value	100

The delay can be specified in Microseconds, Milliseconds and Seconds. Choose Unit from dropdown box. Write "Timer Value" in the box. The above attribute explorer shows the delay value is set to 100 Milliseconds.

7.6. Counter Block

This block increments or decrements CNT1 to CNT10 variables. This block is normally used in loops for counting purpose.



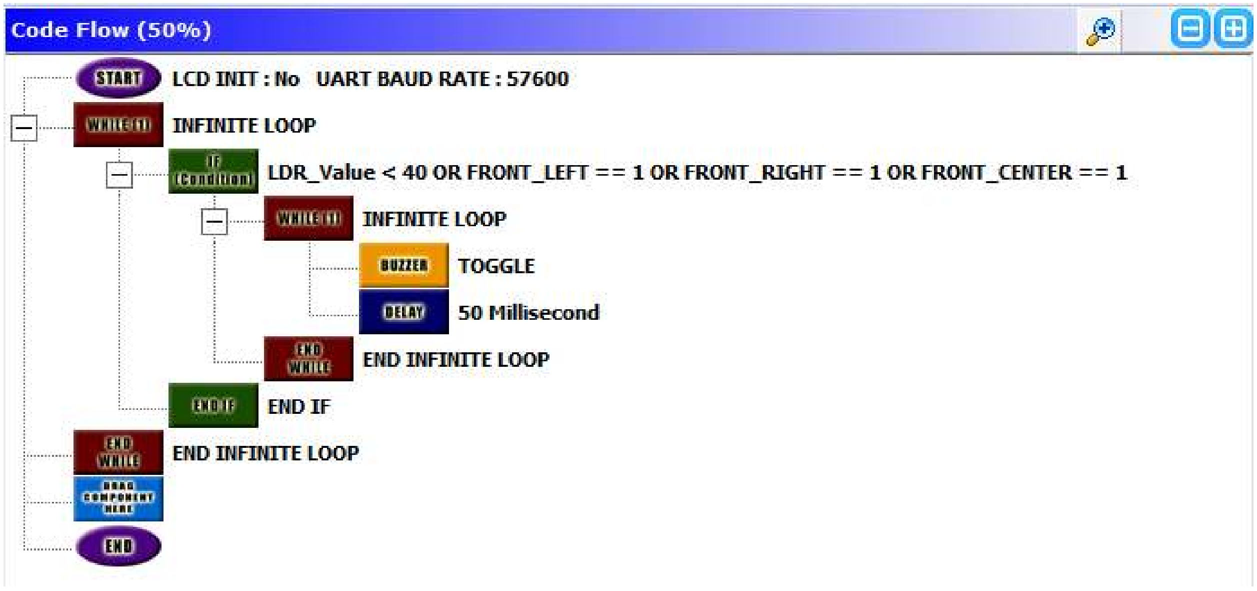
Attributes Explorer

Attribute	Value
VARIABLE NAME	CNT5
Operation	Decrement

Choose Counter number from the drop down box and select operation to be performed. As per above image CNT5 will be decremented.

8. Application Programs

8.1. Bugler alarm

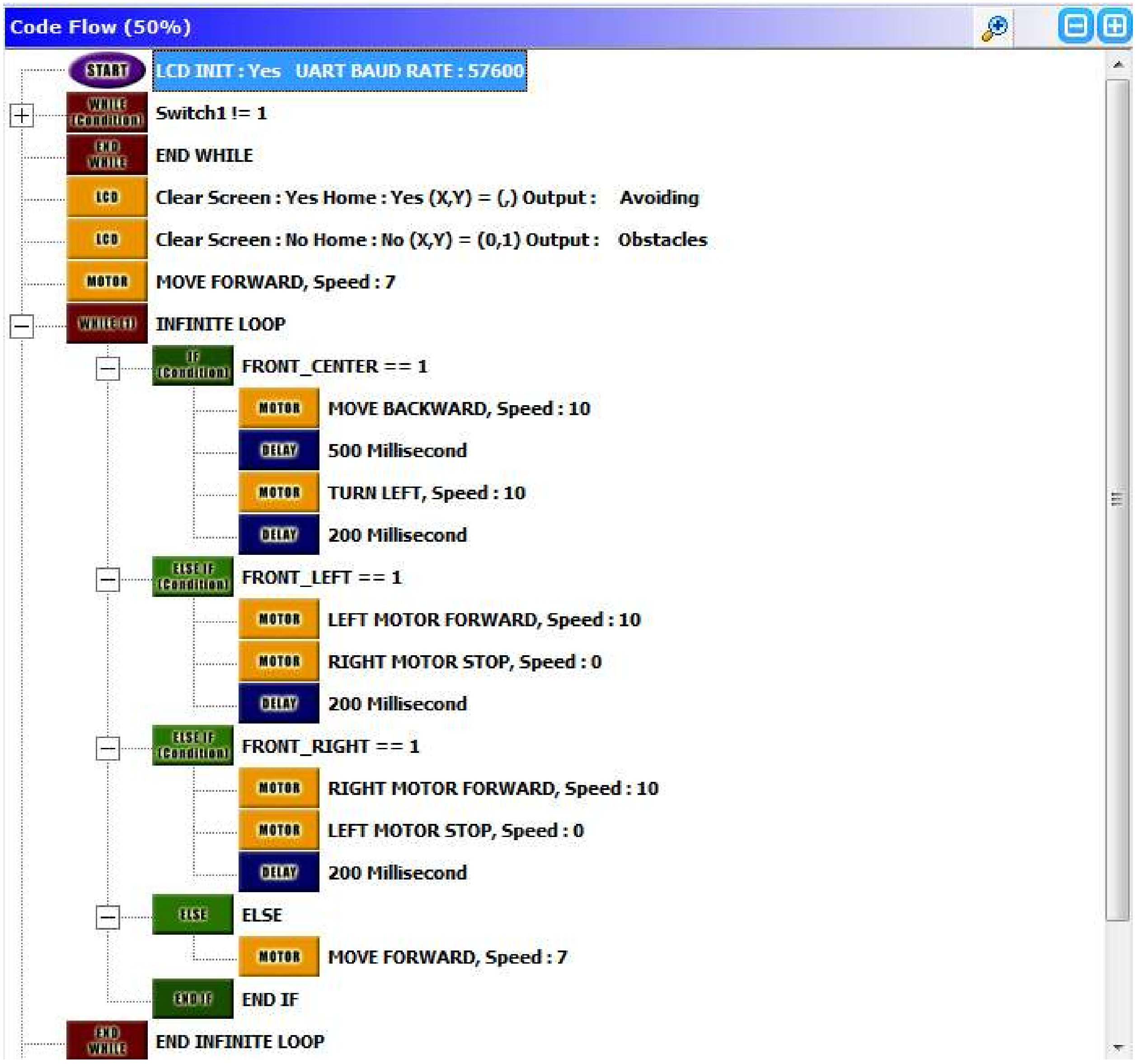


```
START            LCD INIT : No   UART BAUD RATE : 57600
WHILE (1)        INFINITE LOOP
  IF (Condition)   LDR_Value < 40 OR FRONT_LEFT == 1 OR FRONT_RIGHT == 1 OR FRONT_CENTER == 1
    WHILE (1)        INFINITE LOOP
      BUZZER           TOGGLE
      DELAY            50 Millisecond
    END WHILE        END INFINITE LOOP
  END IF           END IF
END WHILE        END INFINITE LOOP
DRAG COMPONENT HERE
END
```

This sample code shows the basic application of sensors and buzzer.

- When any of the front sensors senses an object the buzzer goes on for infinite time.

8.2. Obstacle avoider

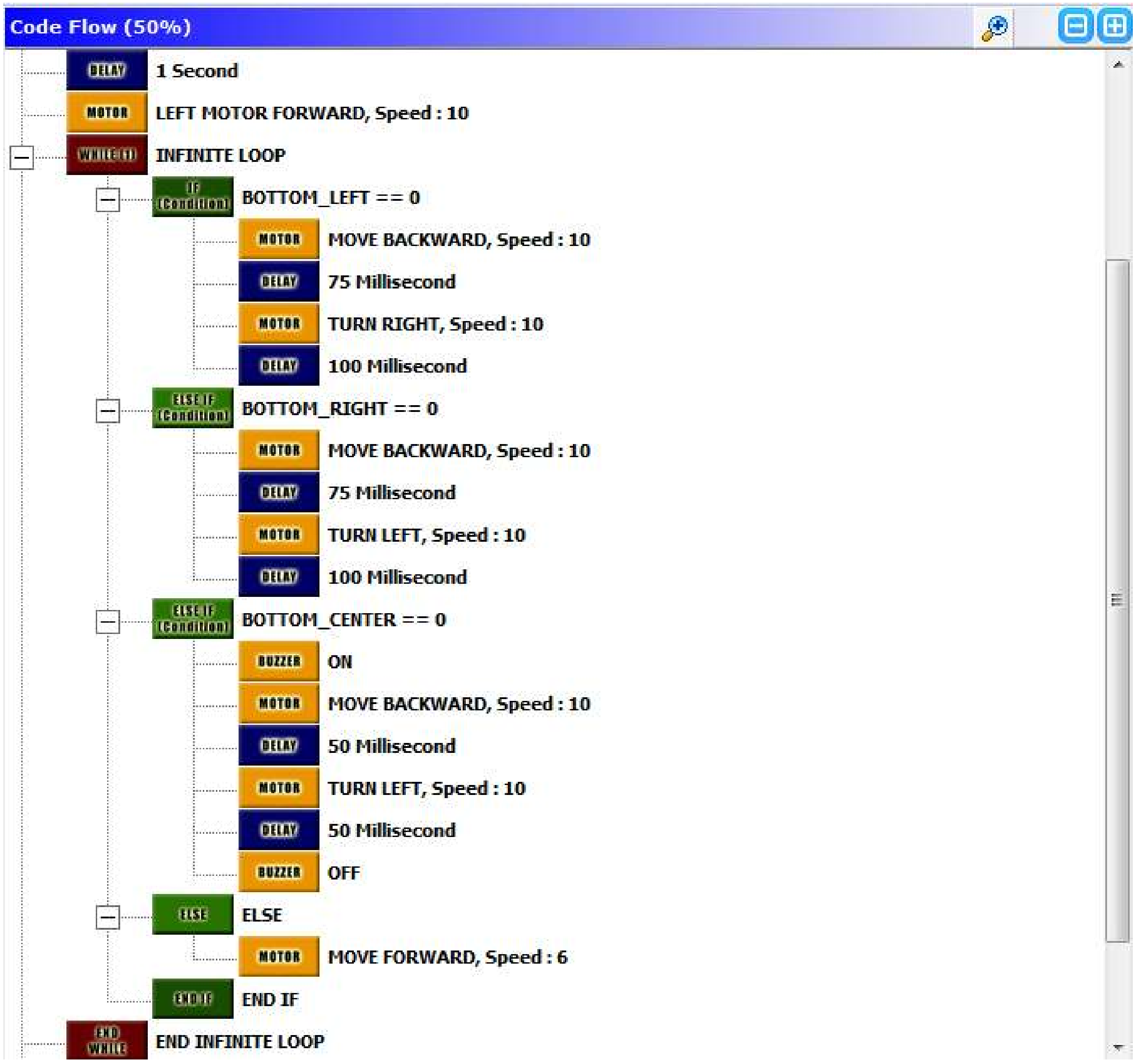


```
START  LCD INIT : Yes   UART BAUD RATE : 57600
WHILE  Switch1 != 1
END WHILE
LCD    Clear Screen : Yes Home : Yes (X,Y) = (,) Output :   Avoiding
LCD    Clear Screen : No Home : No (X,Y) = (0,1) Output :   Obstacles
MOTOR  MOVE FORWARD, Speed : 7
WHILE(1)  INFINITE LOOP
    IF  FRONT_CENTER == 1
        MOTOR  MOVE BACKWARD, Speed : 10
        DELAY  500 Millisecond
        MOTOR  TURN LEFT, Speed : 10
        DELAY  200 Millisecond
    ELSE IF  FRONT_LEFT == 1
        MOTOR  LEFT MOTOR FORWARD, Speed : 10
        MOTOR  RIGHT MOTOR STOP, Speed : 0
        DELAY  200 Millisecond
    ELSE IF  FRONT_RIGHT == 1
        MOTOR  RIGHT MOTOR FORWARD, Speed : 10
        MOTOR  LEFT MOTOR STOP, Speed : 0
        DELAY  200 Millisecond
    ELSE  ELSE
        MOTOR  MOVE FORWARD, Speed : 7
    END IF
END WHILE  END INFINITE LOOP
```

This sample code shows how Qu-Bot can be programmed to make an obstacle avoider robot.

- LCD is initialized to show status of robot.
- The robot starts moving when switch 1 is pressed because there is a while conditional loop on top.
- By default robot keeps moving forward at speed 7
- When front sensor senses obstacle the robot moves backwards for 500 ms and takes a left turn for 200 ms at speed 10
- When left or right sensor senses an obstacle the robot stops opposite side motor to take turn in reverse direction.

8.3. Edge avoider

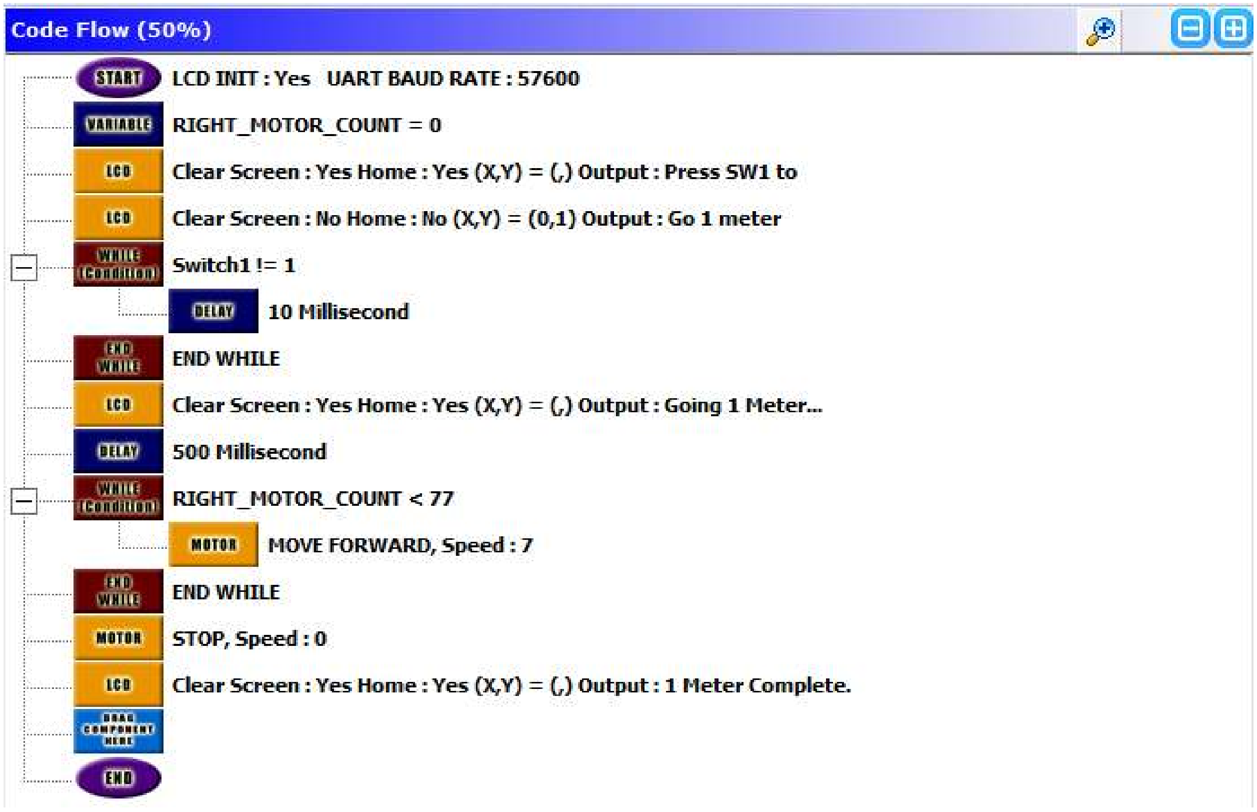


```
DELAY        1 Second
MOTOR        LEFT MOTOR FORWARD, Speed : 10
WHILE (1)    INFINITE LOOP
    IF (Condition)       BOTTOM_LEFT == 0
        MOTOR    MOVE BACKWARD, Speed : 10
        DELAY    75 Millisecond
        MOTOR    TURN RIGHT, Speed : 10
        DELAY    100 Millisecond
    ELSE IF (Condition)  BOTTOM_RIGHT == 0
        MOTOR    MOVE BACKWARD, Speed : 10
        DELAY    75 Millisecond
        MOTOR    TURN LEFT, Speed : 10
        DELAY    100 Millisecond
    ELSE IF (Condition)  BOTTOM_CENTER == 0
        BUZZER   ON
        MOTOR    MOVE BACKWARD, Speed : 10
        DELAY    50 Millisecond
        MOTOR    TURN LEFT, Speed : 10
        DELAY    50 Millisecond
        BUZZER   OFF
    ELSE                 ELSE
        MOTOR    MOVE FORWARD, Speed : 6
    END IF               END IF
END WHILE    END INFINITE LOOP
```

If the robot is kept on a table surface with this code programmed, the robot moves forward by default whenever it senses edge (Bottom sensors doesn't sense table surface) it takes actions accordingly.

- By default the robot moves forward.
- If the bottom center sensor doesn't detect table surface (moves out of edge) robot moves backward and takes a turn
- If bottom left or right sensor moves out of edge the robot takes turn in opposite direction.
- Be careful when using robot with this code programmed. Sometimes there is a chance of robot falling down. Keep alertness to grab the robot if it falls down.

8.4. Distance measurement

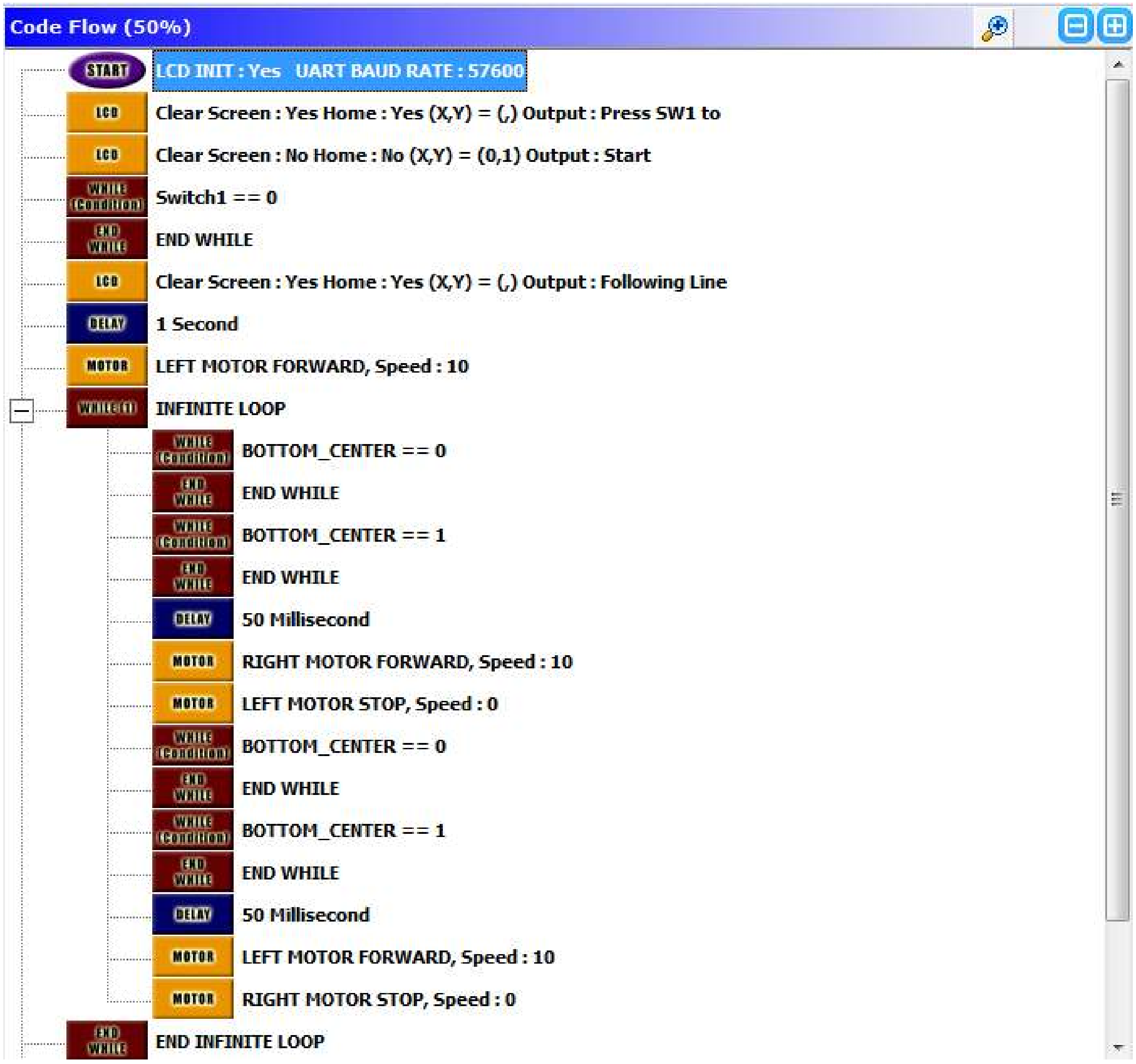

Code Flow (50%)

```
START        LCD INIT : Yes   UART BAUD RATE : 57600
VARIABLE     RIGHT_MOTOR_COUNT = 0
LCD          Clear Screen : Yes Home : Yes (X,Y) = (,) Output : Press SW1 to
LCD          Clear Screen : No Home : No (X,Y) = (0,1) Output : Go 1 meter
WHILE        Switch1 != 1
  DELAY        10 Millisecond
END WHILE    END WHILE
LCD          Clear Screen : Yes Home : Yes (X,Y) = (,) Output : Going 1 Meter...
DELAY        500 Millisecond
WHILE        RIGHT_MOTOR_COUNT < 77
  MOTOR        MOVE FORWARD, Speed : 7
END WHILE    END WHILE
MOTOR        STOP, Speed : 0
LCD          Clear Screen : Yes Home : Yes (X,Y) = (,) Output : 1 Meter Complete.
DRAG COMPONENT HERE
END
```

This sample code shows how to move robot for specific distance using encoder sensors mounted near wheels. This code is made to move robot forward for approximately 1 meter.

- The diameter of the Qu-Bot wheel measures 66.3 mm. So the periphery of the wheel becomes 208.182 mm ($\pi D = 66.3 \times 3.14$).
- The wheel has 8 holes in it so it gives 16 counts (8 on + 8 off) when it rotates 1 revolution. In one revolution it moves forward to 208.182 mm distance (periphery of wheel). So at each count we get distance of 208.182 / 16 = 13 mm. This is also the accuracy which we get when the robot moves on motor count.
- If we want robot to move forward to 1 meter robot has to go forward until it gets 77(1000/13) motor counts on motor counter.
- So this is how it works. When switch1 is pressed robot starts moving forward. The right motor counter is already initialized at code start.
- The while condition checks until the motor count reaches 76. We need 77 counts but there is some inertia when the motors are stopped, so the robot moves a little forward when motors stop, and we get approximately 1 meter move.

8.5. Line Follower using one sensor

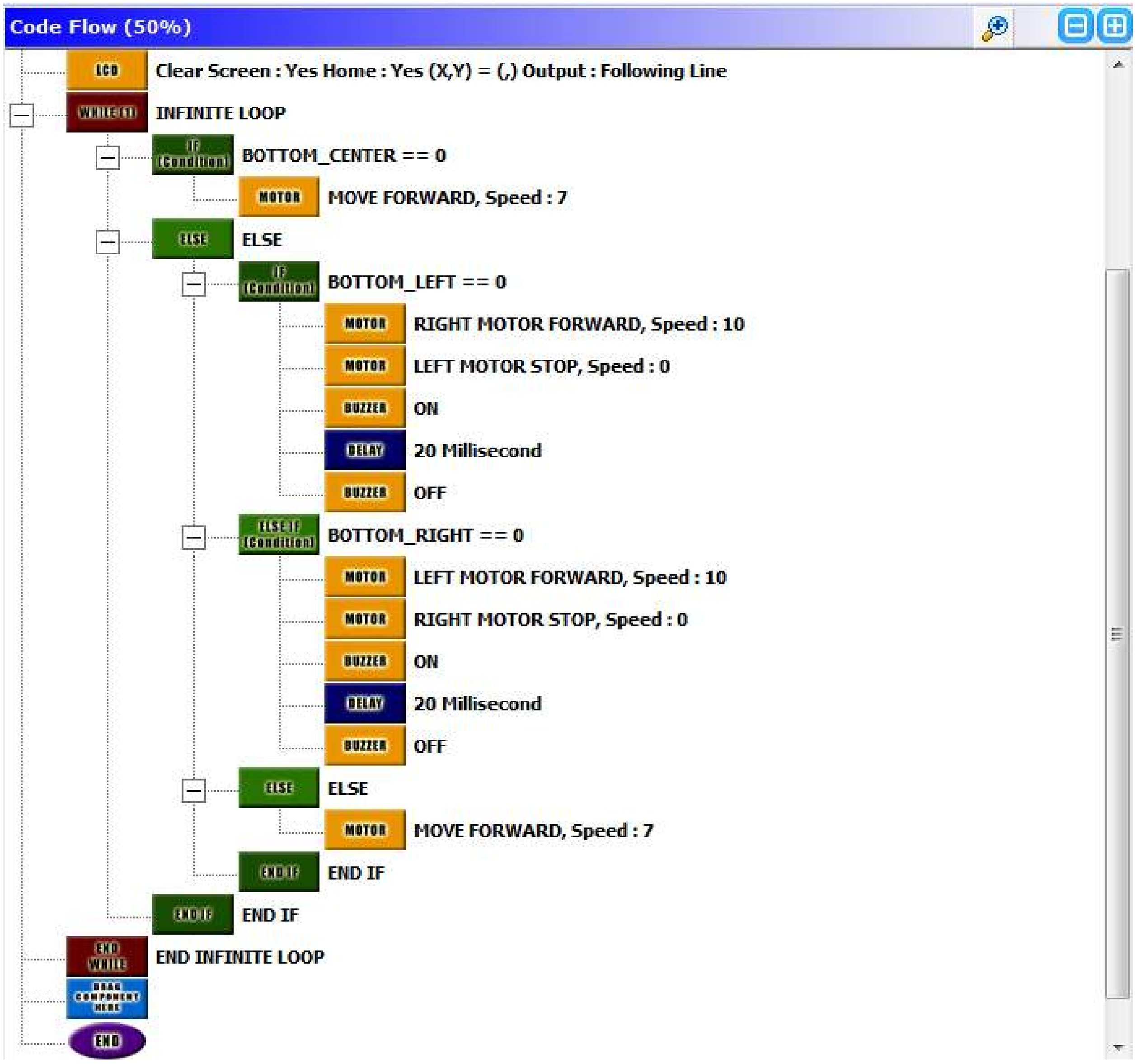


```
START        LCD INIT : Yes   UART BAUD RATE : 57600
LCD          Clear Screen : Yes Home : Yes (X,Y) = (,) Output : Press SW1 to
LCD          Clear Screen : No Home : No (X,Y) = (0,1) Output : Start
WHILE (Condition)  Switch1 == 0
END WHILE    END WHILE
LCD          Clear Screen : Yes Home : Yes (X,Y) = (,) Output : Following Line
DELAY        1 Second
MOTOR        LEFT MOTOR FORWARD, Speed : 10
WHILE(1)     INFINITE LOOP
    WHILE (Condition)  BOTTOM_CENTER == 0
    END WHILE    END WHILE
    WHILE (Condition)  BOTTOM_CENTER == 1
    END WHILE    END WHILE
    DELAY        50 Millisecond
    MOTOR        RIGHT MOTOR FORWARD, Speed : 10
    MOTOR        LEFT MOTOR STOP, Speed : 0
    WHILE (Condition)  BOTTOM_CENTER == 0
    END WHILE    END WHILE
    WHILE (Condition)  BOTTOM_CENTER == 1
    END WHILE    END WHILE
    DELAY        50 Millisecond
    MOTOR        LEFT MOTOR FORWARD, Speed : 10
    MOTOR        RIGHT MOTOR STOP, Speed : 0
END WHILE    END INFINITE LOOP
```

This code uses only one sensor to follow a dark color line on light colored floor.

- In the start robot starts running left motor. Right motor is stopped. So the robot takes a long right turn.
- When it gets the line under the bottom sensor it continues to take right turn until it crosses it completely.
- After a small delay of 50ms the left motor stops and right motor starts moving so robot takes a left turn.
- Again it continues taking the left turn until it crosses the line. After a 50 ms delay the procedure repeats because of infinite loop.

8.6. Line Follower using three sensors

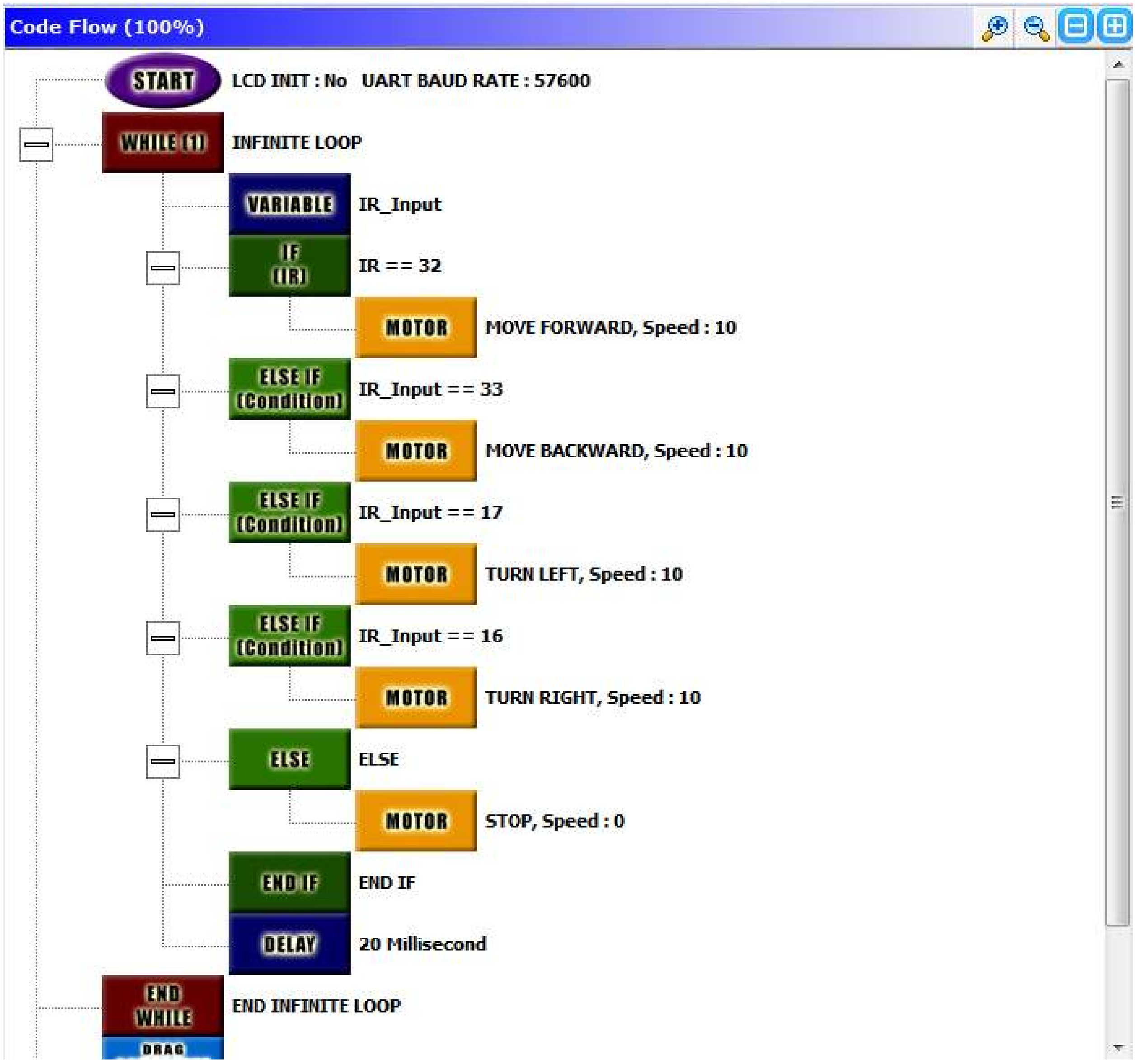


```
LCD           Clear Screen : Yes Home : Yes (X,Y) = (,) Output : Following Line
WHILE (1)     INFINITE LOOP
  IF (Condition)  BOTTOM_CENTER == 0
    MOTOR         MOVE FORWARD, Speed : 7
  ELSE            ELSE
    IF (Condition)  BOTTOM_LEFT == 0
      MOTOR           RIGHT MOTOR FORWARD, Speed : 10
      MOTOR           LEFT MOTOR STOP, Speed : 0
      BUZZER          ON
      DELAY           20 Millisecond
      BUZZER          OFF
    ELSE IF (Condition)  BOTTOM_RIGHT == 0
      MOTOR           LEFT MOTOR FORWARD, Speed : 10
      MOTOR           RIGHT MOTOR STOP, Speed : 0
      BUZZER          ON
      DELAY           20 Millisecond
      BUZZER          OFF
    ELSE            ELSE
      MOTOR           MOVE FORWARD, Speed : 7
    END IF          END IF
  END IF          END IF
END WHILE     END INFINITE LOOP
DRAG COMPONENT HERE
END
```

This code uses 3 bottom sensors to make Qu-Bot follow a dark color line on light color surface.

- When the bottom center sensor is on the dark line the value is 0. This means the robot is on the line. It will move forward in this condition.
- If the bottom center sensor goes off, either left or right sensor will encounter the line. If left sensor senses the line, left motor stops and right motor runs at full speed so robot takes a left turn and comes back on line.
- If Right sensor encounters line right motor stops and left goes on full speed so robot takes right turn and comes back on line.

8.7. Remote Controlled Qu-Bot

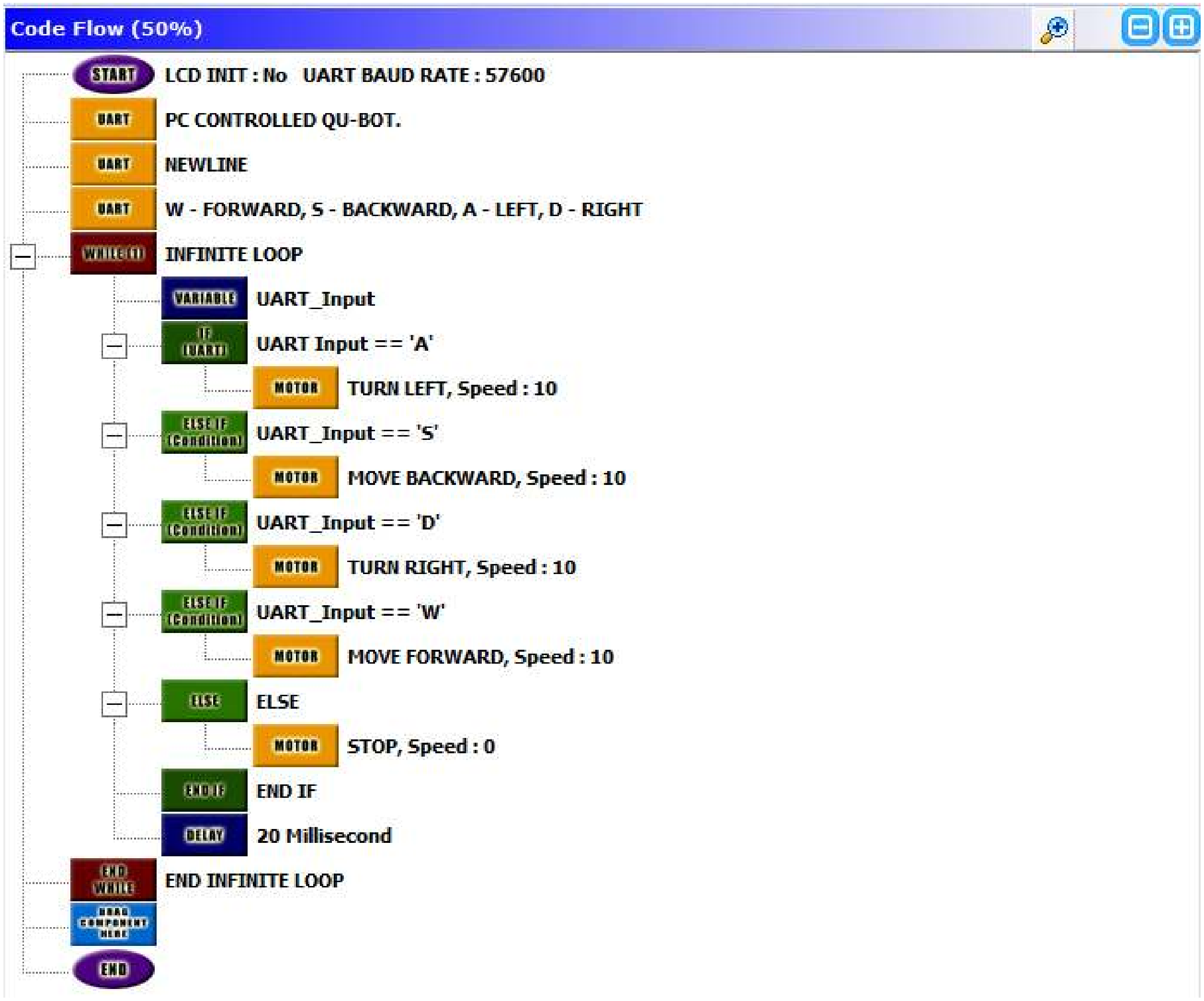

Code Flow (100%)

```
START        LCD INIT : No   UART BAUD RATE : 57600
WHILE (1)    INFINITE LOOP
    VARIABLE     IR_Input
    IF (IR)      IR == 32
        MOTOR        MOVE FORWARD, Speed : 10
    ELSE IF (Condition)   IR_Input == 33
        MOTOR        MOVE BACKWARD, Speed : 10
    ELSE IF (Condition)   IR_Input == 17
        MOTOR        TURN LEFT, Speed : 10
    ELSE IF (Condition)   IR_Input == 16
        MOTOR        TURN RIGHT, Speed : 10
    ELSE         ELSE
        MOTOR        STOP, Speed : 0
    END IF       END IF
    DELAY        20 Millisecond
END WHILE    END INFINITE LOOP
DRAG
```

Programming this code into Qu-Bot makes it a manual remote controlled robot.

- When any switch is pressed on IR remote, it transmits a RC5 code. In the remote included with this kit the code is from 1 to 127.
- There are 4 big arrow keys on the center of remote. This code uses them for moving robot.
- Any key of the remote can be used to create any action on robot. The 4 arrow keys have IR codes of 32(Up), 33(Down), 17(Left) & 16(Right).
- The key codes of other keys can be recognized by printing the IR Input variable to LCD or serial Terminal on PC.
- For manual control robot there is a simple loop. For any input of IR out of 4 keys the robot gives specific action.

8.8. PC controlled Qu-Bot

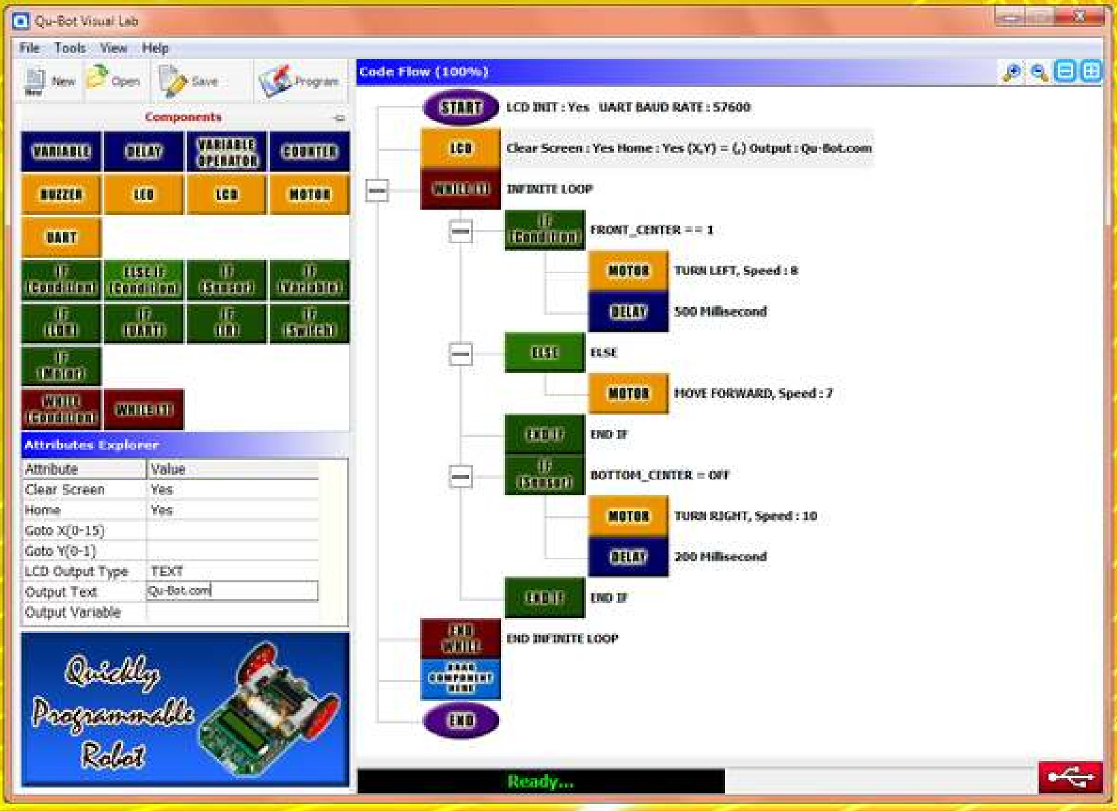


```
START         LCD INIT : No   UART BAUD RATE : 57600
UART          PC CONTROLLED QU-BOT.
UART          NEWLINE
UART          W - FORWARD, S - BACKWARD, A - LEFT, D - RIGHT
WHILE (1)     INFINITE LOOP
    VARIABLE      UART_Input
    IF (UART)     UART Input == 'A'
        MOTOR         TURN LEFT, Speed : 10
    ELSE IF (Condition)  UART_Input == 'S'
        MOTOR         MOVE BACKWARD, Speed : 10
    ELSE IF (Condition)  UART_Input == 'D'
        MOTOR         TURN RIGHT, Speed : 10
    ELSE IF (Condition)  UART_Input == 'W'
        MOTOR         MOVE FORWARD, Speed : 10
    ELSE          ELSE
        MOTOR         STOP, Speed : 0
    END IF        END IF
    DELAY         20 Millisecond
END WHILE     END INFINITE LOOP
DRAG COMPONENT HERE
END
```

Qu-Bot has a facility to communicate with PC through USB. Connecting the USB cable creates a Virtual Serial Port on PC. Through this port serial communication can be done with PC. This communication is called UART (Universal Asynchronous Receiver Transmitter). There are different terminal software's available for serial communication like HyperTerminal and bray's terminal. The QBVLab software itself includes a terminal. In advanced applications it's possible to control Qu-Bot from the inputs of mouse, keyboard, webcam, internet or any other source available on PC. However this needs a high level of programming and hardware knowledge on PC but this sample code gives a demo on how to move Qu-Bot on commands taken from keyboard.

- In an infinite loop the robot seeks for UART input. When a key from A, S, D and W is pressed terminal sends the characters to robot and it takes appropriate action by moving itself to left, back, right or forward.

Qu-Bot Visual Lab Software



Qu-Bot Visual Lab
File Tools View Help
New Open Save Program
Components
VARIABLE DELAY VARIABLE OPERATOR COUNTER
BUZZER LCD LCD MOTOR
UART
IF (Condition) ELSE IF (Condition) IF (Sensor) IF (Variable)
IF (Lcd) IF (UART) IF (IR) IF (Switch)
IF (Motor)
WHILE (Condition) WHILE (1)
Quickly Programmable Robot
Code Flow (100%)
Ready...

```
START          LCD INIT : Yes  UART BAUD RATE : 57600
LCD            Clear Screen : Yes Home : Yes (X,Y) = (,) Output : Qu-Bot.com
WHILE (1)      INFINITE LOOP
  IF (Condition)   FRONT_CENTER == 1
    MOTOR          TURN LEFT, Speed : 8
    DELAY          500 Millisecond
  ELSE             ELSE
    MOTOR          MOVE FORWARD, Speed : 7
  END IF           END IF
  IF (Sensor)      BOTTOM_CENTER = OFF
    MOTOR          TURN RIGHT, Speed : 10
    DELAY          200 Millisecond
  END IF           END IF
END WHILE      END INFINITE LOOP
DRAG COMPONENT HERE
END
```

Attributes Explorer

Attribute	Value
Clear Screen	Yes
Home	Yes
Goto X(0-15)	
Goto Y(0-1)	
LCD Output Type	TEXT
Output Text	Qu-Bot.com
Output Variable	

Qu-Bot Online Resources

***Website** : http://www.qu-bot.com

***Forums :** http://www.qu-bot.com/forum

Made
In
India

For bulk orders contact info@qu-bot.com