

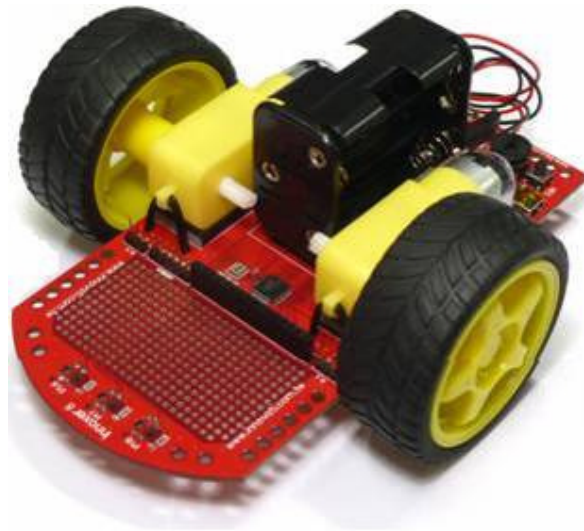
Innoxer™ 8

Line Follower

Instruction Manual


Document Rev 1.4

June 17, 2013



Passion for innovation

Trademark

Innovati®, , and BASIC Commander® are registered trademarks of Innovati, Inc.

InnoBASIC™, cmdBUS™ and Innoxer™ are trademarks of Innovati, Inc.

Copyright © 2012 by Innovati, Inc. All Rights Reserved.

Due to continual product improvements, Innovati reserves the right to make modifications to its products without prior notice. Innovati does not recommend the use of its products for application that may present a risk to human life due to malfunction or otherwise.

No part of this publication may be reproduced or transmitted in any form or by any means without the expressed written permission of Innovati, Inc.

Disclaimer

Full responsibility for any applications using Innovati products rests firmly with the user and as such Innovati will not be held responsible for any damages that may occur when using Innovati products. This includes damage to equipment or property, personal damage to life or health, damage caused by loss of profits, goodwill or otherwise. Innovati products should not be used for any life saving applications as Innovati's products are designed for experimental or prototyping purposes only. Innovati is not responsible for any safety, communication or other related regulations. It is advised that children under the age of 14 should only conduct experiments under parental or adult supervision.

Errata

We hope that our users will find this instruction manual a useful, easy to use and interesting publication, as our efforts to do this have been considerable. Additionally, a substantial amount of effort has been put into this instruction manual to ensure accuracy and complete and error free content, however it is almost inevitable that certain errors may have remained undetected. If you find any errors in the instruction manual, contact us via email service@innovati.com.tw. For the most up-to-date information, please visit our web site at <http://www.innovati.com.tw>.

Table of Contents

Overview	3
Features	3
Part List	4
Board Layout	5
Pin Assignment	6
Assembly Procedures	7
Installing the InnoBASIC™ Workshop	12
Hardware Installation	12
Creating Your First Program	12
How does the Line Follower work?	13
Tutorial Programs	15
● Ex. 1 --- LEDs	16
● Ex. 2 --- Buttons	16
● Ex. 3 --- Buzzers	18
● Ex. 4 --- Infrared Intensity Measuring	19
● Ex. 5 --- Motor Rotation Direction Control	21
● Ex. 6 --- Motor Speed Control	23
● Ex. 7 --- Follow Line	24
● Ex. 8 --- PID Control	28

Overview

Innoxer™ 8 Line Follower is a wheeled robot designed for the beginners to learn how to navigate it by following a line. The Innoxer™ 8 is powered by the BASIC Commander® system, which is based on an 8-bit microcontroller; hence, the name Innoxer™ 8.

This manual is divided in four parts. The first part contains instructions on how to assemble the Innoxer™ 8 Line Follower.

The second part shows how to install the InnoBASIC™ Workshop, integrated development environment in the computer to write your program code. Only the necessary knowledge about how to run the project will be given in this manual. For more detailed information about the BASIC Commander® system and the usage of the innoBASIC™ language, refer to “BASIC Commander & innoBASIC Workshop User’s Manual.”

The third part is a brief introduction to the line follower fundamentals, which will help you understand what a line follower does in order to know how to write Line Follower programs. More complicated control techniques can be deployed on the Innoxer™ 8 Line Follower by modifying the sample programs.

The fourth part of the manual contains several step-by-step exercises, with simple LED, button and buzzer control, which help you build up your logical thinking. Then you will learn how to use infrared sensors to find the line, and how to control motors to follow the line. With sensing, control capabilities and intelligent decision making skills, you will be able to use the Innoxer™ 8 Line Follower for more explorations.

Features

With a few steps of assembly, Innoxer™ 8 Line Follower is ready as an educational platform for robotics. It is equipped with motors, wheels, battery holder and the following additional features.

- Built-in BASIC Commander® system as controller.
- USB interface for program downloading and debugging.
- 8 General Purpose Digital I/Os.
- 14 dedicated I/Os for built-in device access.
- Built-in H-bridge driver devices.

- 6-Pin cmdBUS™ connector for Innovati's smart module interface.

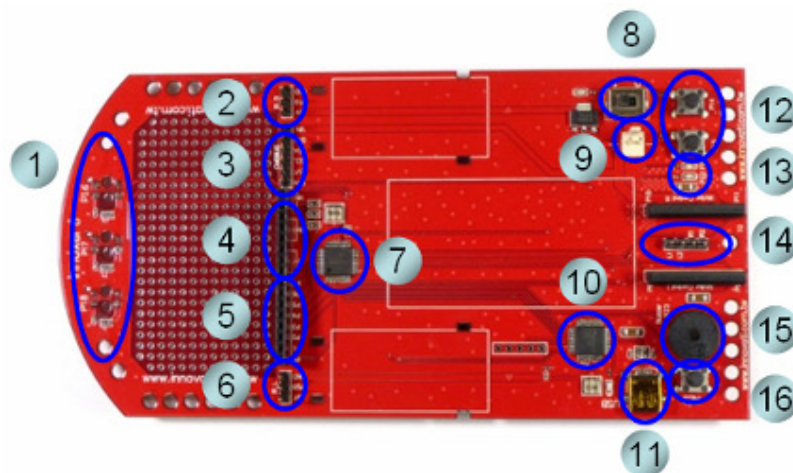
Part List

Item	Illustration	Qty
Control Board (Chassis)		1
DC Motor with Gear Box		2
Wheels & Tires		2
6-AA battery Holder (Batteries not included)		1
Cable Ties		8
Motor Power Wires		4
Support Pad Set		1
Velcro Tapes (Pair)		1

Double-sided Foam Tape		2
Heat Shrink Tube		1
Mini USB Cable		1
Electric Tape		1
CD		1

Board Layout

This section illustrates the different parts of the Innoxer™ 8 Line Follower Control Board and their functions. The Control Board is also used as the chassis of the line follower.



Item	Description	I/O Pins
1	Three pairs of infrared LEDs and detectors.	P16, P17, P18
2	Connector IR_R is reserved for right side infrared detector.	P19
3	A cmdBUS™ connector for Innovati's Smart Module connection. Check the label on board when connecting the cmdBUS™ cable, incorrect insertion may damage the modules.	
4	Eight general-purpose I/O pins.	P0 to P7
5	Eight power pins of power input regulated 5V rating 200mA labeled VIN and GND. for user's application.	
6	Connector IR_L is reserved for left side infrared detector.	P20
7	Main microcontroller, where user's program is downloaded and executed.	
8	Power switch. Set to position 1 to turn on and 0 to turn off the power.	
9	Power input connector for the battery holder.	
10	USB controller IC.	
11	Mini USB connector for program downloading and debugging.	
12	Two built-in buttons for user's application.	P14, P15
13	Two built-in LEDs for user's application.	P21, P22
14	Motor power supply. Pins labeled L1, L2 are for left side motor and R1, R2 for right side motor. They are connected to output pins of the H-bridge driver devices.	P8 to P11
15	Built-in Buzzer for user's application.	P23
16	RES Button for resetting the program execution. Note that it is prohibited to press this button during downloading. Doing so will result in download failure.	

Pin Assignment

Pins	Function
P0~P7	General-purpose I/O pins.
P8~P11	H-Bridge Drivers control.
P12, P13	Unused and not available.
P14, P15	Built-in press buttons.
P16, P17, P18	Infrared LEDs and sensors.

P19, P20	Reserved for left and right side infrared detectors.
P21, P22	Built-in LEDs control.
P23	Built-in Buzzer control.

Assembly Procedures

Step 1: Connecting the motor wires

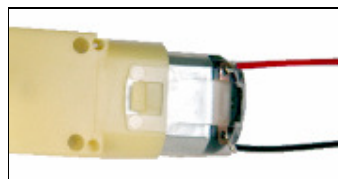
Take the red motor power wire and rip the cut plastic insulation. Gently twist the strands of copper wire with your fingers.



Insert the twisted strands into the DC motor power line connector ring and twist it tight as shown below. If you have a soldering iron, you can use it to secure the joint.

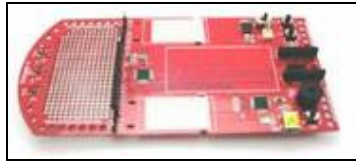


Follow the same steps for the black wire. To be consistent with the control of motors in the tutorial programs later, follow the color order as shown below. If the two wires are exchanged, the wheel will rotate reversely. Apply same steps for the second motor.



Step 2: Installing the DC motors

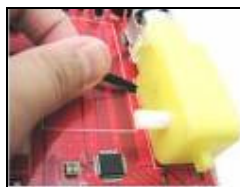
Place the two straps of double-sided foam tape on the board onto the area marked with white line and then remove the upper cover of the tape for use.



Carefully place the DC motor on the double-sided tape. Make sure the DC motor is placed correctly with the screw head on the top side facing outward at the edge of the board as shown in the figure below. Otherwise, the wheels will rotate reversely as expected in the tutorial programs.



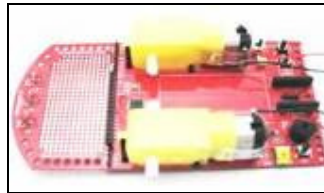
Insert a cable tie through the board and the gear box and then tighten the cable tie to secure the DC motor as shown below.



Similarly, insert another cable tie through the board and the ring of the gear box and tighten the cable tie to secure the DC motor as shown below. Make sure that the motors are fixed on the board firmly and safely. Cut the excess part of the cable ties.

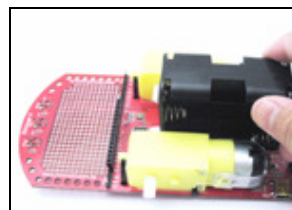


Repeat the same steps to install the second DC motor on the board. When finished, it would look like as shown below.



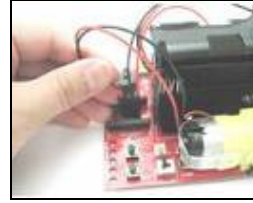
Step 3: Installing the battery holder

To secure the battery holder on board, place the Velcro sticky back tape on the central area on the board. Then place the Velcro tape on the edge side of the battery holder. Place the battery holder on the central area as shown below. Do not insert the batteries until the Innoxer™ 8 Line Follower is assembled completely.



Step 4: Connecting the wires to the board

To be consistent with the tutorial programs, connect the left motor power wires to the central connector pins, with the red wire on pin L1 and black wire on pin L2. Connect the right motor power wires with red wire on pin R2 and black wire on pin R1. Otherwise the motors will rotate reversely to the settings in the tutorial programs.



Connect the battery holder wires to the power input connector as shown below. Use a cable tie to tie up the wires neatly.



Step 5: Installing the Infrared insulation tube

A line following course can have black surface with white lines or vice versa. By measuring the infrared intensity reflected from a surface, we can distinguish the white and black color on the course. To ensure the infrared light detected by the infrared detector is reflected from the course but not leaking from the infrared LED next to it, it is necessary to add an insulation tube to the infrared LEDs. Heat shrink tubes are used as the insulator.

Cut three pieces of heat shrink tubes of 7mm length and cover the infrared LEDs all the way to the bottom as shown below.



Step 6: Installing the support wheel (pad)

To keep the Innoxer™ 8 Line Follower steady while moving, add the support pad as a third wheel on the board. The pad is designed to run on smooth surfaces. First, fasten

the Hex Post to the bottom of the board and then screw the support pad to the Hex Post as shown below. Then, close the flip cap of the pad.

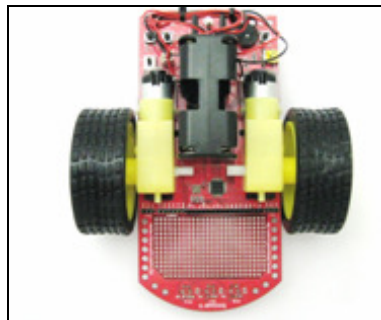


Step 7: Installing the wheels

Place the wheels in the rims and make sure the tire tread directions are the same on each tire. After the wheels are assembled, insert them to the motor shafts directly. Push the wheel as far as it can go while inserting, so the wheel will not fall out when running.



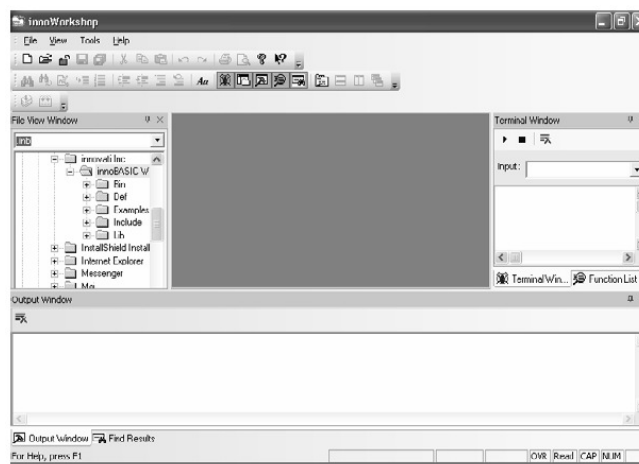
Once the Innoxer™ 8 Line Follower is completely assembled, it should look like as shown below. Place six AA batteries into the battery holder and the Line Follower is ready to be programmed.



Installing the InnoBASIC™ Workshop

This section will demonstrate how to install the integrated development system InnoBASIC™ Workshop, and how to use the basic functions such as writing program and editing, compiling, downloading and debug in the InnoBASIC™ Workshop environment.

Insert the supplied CD into the CD-ROM drive and follow the on-screen instructions to install the innoBASIC™ Workshop, or visit Innovati® website to download the latest version of the innoBASIC™ Workshop. After the installation is complete, run the innoBASIC™ Workshop.



Hardware Installation

Connect the USB cable to the PC and the mini USB connector on the Innoxer™ 8 Line Follower. As the USB port power supply is not designed to be used by the Innoxer™ 8 Line Follower, the power on the Innoxer™ 8 Line Follower needs to be switched on for USB hardware installation and operation.

If the USB drivers are not installed on the PC, it will prompt for installation of new hardware found. Follow the instructions on the screen to complete the installation of the USB drivers.

Creating Your First Program

Once the hardware and software are successfully installed, following the below steps to write the first program.

Step 1:

Select File/New menu or click the "New" icon to open a new file. A new editing window will appear on the screen with the title "Untitled1".

Step 2:

Write the following code in the editing window.

```
Sub Main ()  
Debug "Hello World!"  
End Sub
```

Step 3:

Select File/Save menu command or click Save File icon to save the file. You can select the location where the file is to be saved. Note that the file is saved with the extension ".inb", which stands for innoBASIC™ language program file.

Step 4:

Select the Build commander under the Build menu or click the Build icon to compile and download the program to the Innoxer™ 8 Line Follower. If the program has been written correctly, the program will be compiled into machine code and downloaded via the USB cable to the Innoxer™ 8 Line Follower. A green LED on the board will flash indicating the download process is in progress.

Step 5:

After the program is downloaded completely on the Innoxer™ 8 Line Follower, it will be executed automatically. The "Hello World!" message will appear in the Terminal Window.

How does the Line Follower work?

The Line Follower is always the first project for people who want to explore the robotics world. Here are the basic concepts of a Line Follower.

Motor Speed Control

The wheels of a Line Follower are controlled separately by two motors. By controlling the direction of current that flows into the motor with an H-bridge driving IC, we can control

the rotation direction of the motor. For more details about the use of the H-bridge driving IC, refer to the Motor Rotation Direction Control section.

For more precise motion control, it is necessary to control also the rotation speed. The easiest way to control the motor speed is to use the PWM (Pulse Width Modulation) technique, which controls the circuit on/off duty ratio, so the current flows through the motor is controlled. By setting the two motors with different speed, the line follower can change its heading direction.

Infrared Sensors for Line Tracking

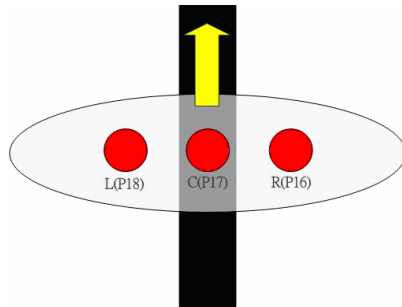
Usually, the line on a Line Follower track is a black line on the white surface. The infrared light is absorbed by the black surfaces. So we use an infrared LED as light source to shed on the surface and an infrared detector to measure the intensity of the infrared reflection. If the black strip is detected, then the reflection intensity will be low; for the white area, the reflection intensity will be high.

To track a line, usually we need three pairs of infrared LEDs and sensors. By checking the reflection intensity of the three detectors, we will know the location of the line under the Line Follower. By changing the speed of the two motors, we can keep the black line under the center of three sensors. There are more advanced analog methods to calculate more accurately the black line location and more advanced PID control theories to improve the tracking stability.

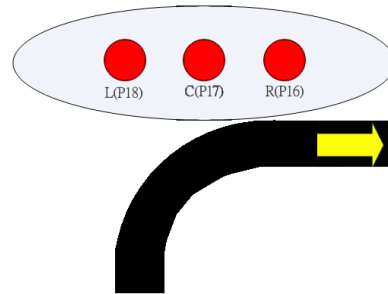
Line Tracking

To track a line, we need to place at least two infrared sensors on the left and right side of the line to keep the line in the center of the line follower. For more complicated course conditions, we need three infrared sensors, which can detect a broken line or an intersection. For faster line following applications, more IR sensors can be used, for example, five sensors, to keep track of the line.

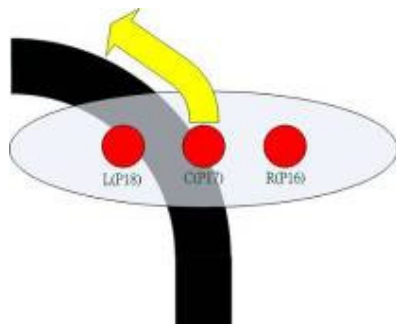
The following drawings show the typical course conditions. By reading the status of infrared sensors, we know the line curving direction ahead and change motor speeds to keep the line follower on the track. For more details about the use of the H-bridge driving IC and line tracking, refer to the Motor Rotation Direction Control section.



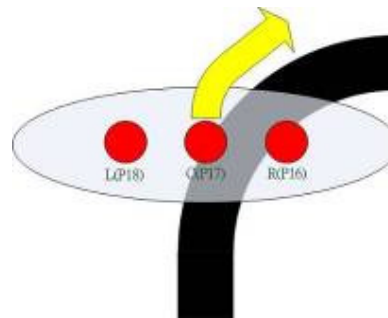
Straight Line: detected by P17



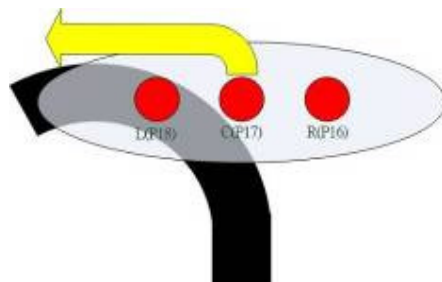
Derail: not detected



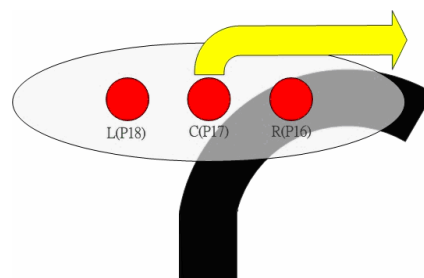
Left Turn: detected by P17 and P18



Right Turn: detected by P17 and P16



Sharp Left Turn: detected by P18



Sharp Right Turn: detected by P16

Tutorial Programs

The Innoxer™ 8 Line Follower is an excellent robotics kit for beginners. To help beginners be familiar with the built-in electronic components and programming skills for sensor and control, a series of step-by-step tutorial programs is provided in this section.

To maintain the tutorial programs free of error and up-to-date, they are subject to change without notice. Visit Innovati website to download the up-to-date tutorial programs. To learn more about the BASIC Commander® system, refer to the “BASIC Commander® and innoBASIC™ Workshop User's Manual” for more information.

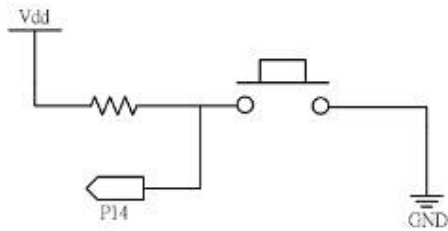
Ex. 1 --- LEDs

LEDs are widely used in our daily life and they are also easy electronic devices for beginners to control. On the Innoxer™ 8 Line Follower, there are two built-in LEDs for learning and application purposes. They are connected to the I/O pin 21 and pin 22.

```
'-----  
' Innoxer 8  
' Ex.1: LEDs  
' Function: LEDs blink alternatively  
'-----  
Sub main()  
Do  
    High 21      'turn on LED at pin 21  
    Pause 200   'wait 0.2 second  
    Low 21       'turn off LED at pin 21  
  
    High 22      'turn on LED at pin 22  
    Pause 200   'wait 0.2 second  
    Low 22       'turn off LED at pin 22  
Loop  
End Sub
```

Ex. 2 --- Buttons

Buttons are the most widely used input devices. On the Innoxer™ 8 Line Follower, two push buttons are built-in for learning and application purposes. They are connected to the I/O pin 14 and pin 15. Below diagram shows configuration of pin 14.



The IN command can be used to read the I/O status. The syntax of IN command is shown as below:

Status = IN(Pin)

The **Pin** is I/O pin number, where the button is connected. On the Line Follower, they are connected to pin 14 and 15. The **Status** is the result returned from the I/O pin. If the button is pressed, a logic 0 will be returned and stored in **Status**, otherwise a logic 1 will be returned. In this program, we add the code to the Ex.1.

```

'-----
' InnoXer 8
' Ex.2: Buttons
' Function: Press buttons to light LEDs
'-----

Sub main()
Dim status As Byte
Do
    status = In(14)           'read button status (pin 14)
    If status = 0 Then       'if button pressed
        High 21              'turn on LED (pin 21)
    Else                      'otherwise
        Low 21               'turn off LED (pin 21)
    End If

    status = In(15)         'read button status (pin 15)
    If status = 0 Then       'if button pressed
        High 22              'turn on LED (pin 22)
    Else                      'otherwise
        Low 22               'turn off LED (pin 22)
    End If

```

```
Loop
End Sub
```

Ex. 3 --- Buzzer

Buzzer is made of piezoelectric ceramic, which can produce sound based on the inverse piezoelectric effect. The built-in buzzer is connected to the I/O pin 23. The `FREQOUT` command is used to generate tones. The syntax is shown as below:

`FREQOUT Pin, Duration, Frequency`

The *Pin* is I/O pin number, where the buzzer is connected. On this Line Follower, it is connected to pin 23. The *Duration* is a constant or variable ranging from 0 to 65535 of unit ms to indicate how long to play a note. *Frequency* is a constant or variable ranging from 0 to 65535 to specify what square wave frequency is to be generated. In this program, we combine Ex.1 and Ex. 2 to create a music instrument program.

```
'-----
' InnoXer 8
' Ex.3: Buzzer
' Functions: Control Buzzer to play notes.
'-----

Sub main()
Dim status As Byte
Do
    status = In(14)           'read button status (pin 14)

    If status = 0 Then       'if button pressed
        High 21              'turn on LED (pin 21)
        Freqout(23, 250, 523) 'Do   1 C
        Freqout(23, 250, 587) 'Re   2 D
        Freqout(23, 250, 659) 'Mi   3 E
        Freqout(23, 250, 698) 'Fa   4 F
        Freqout(23, 250, 785) 'Sol  5 G
        Freqout(23, 250, 880) 'La   6 A
        Freqout(23, 250, 988) 'Ti   7 B
```

```

    Freqout(23, 250, 1047) 'Do 8 C (high)
    Low 21                  'turn off LED (pin 21)
End If

status = In(15)           'read button status (pin 15)
If status = 0 Then       'if button pressed
    High 22              'turn on LED (pin 22)
    Freqout(23, 200, 785) 'Sol 5 G
    Pause 50
    Freqout(23, 200, 659) 'Mi 3 E
    Pause 50
    Freqout(23, 450, 659) 'Mi 3 E
    Pause 50
    Freqout(23, 200, 698) 'Fa 4 F
    Pause 50
    Freqout(23, 200, 587) 'Re 2 D
    Pause 50
    Freqout(23, 450, 587) 'Re 2 D
    Pause 50
'-----
' Add more lines here to complete this music.
'-----

    Low 22                'turn off LED (pin 22)
End If
Loop
End Sub

```

Ex. 4 --- Infrared Intensity Measuring

The most important mission of Innoxer™ 8 Line Follower is to find the line and follow it. The most common technique is to use infrared light. When an infrared LED is used to generate infrared light, it will be reflected mostly on a white surface or absorbed mostly on a black one. By measuring the intensity of infrared reflection among the several infrared sensors, we can tell where the line is and take actions to follow the line.

In this program, we use the RCTIME command to measure RC charging time, which is in proportion with the intensity of the infrared reflection. The RCTIME command syntax is shown as below:

RCTIME Pin, State, Variable

The **Pin** is I/O pin number, where the Infrared RC circuit is connected. On this Line Follower, they are connected to pin 16, 17 and 18. The **State** represents the discharge time with value 0 or charging time with value 1. On this Line Follower, a charging circuit is deployed, therefore, the value will be always 1. The **Variable** is a variable of WORD data type to store the measured charging time or discharging time. The number stored in the variable is of 5us unit. In this program the values measured are displayed in the Terminal Window.

```
'-----
' InnoXer 8
' Ex.4: Infrared Intensity Measuring
' Function: Read and display Infrared reflection intensity.
'-----
Sub main()
Dim IR_L, IR_C, IR_R As Word

Do
    High 16          'IR light on
    Rctime 16,1, IR_R  'measure right side IR intensity
    Low 16           'IR light off

    High 17          'IR light on
    Rctime 17,1, IR_C  'measure central IR intensity
    Low 17           'IR light off

    High 18          'IR light on
    Rctime 18,1, IR_L  'measure left side IR intensity
    Low 18           'IR light off

    Debug CSRXY(1,1), "IR_R: ", IR_R, CLREOL  'right IR value
    Debug CSRXY(1,2), "IR_C: ", IR_C, CLREOL  'central IR value
    Debug CSRXY(1,3), "IR_L: ", IR_L, CLREOL  'left IR value
```

Loop
End Sub

Ex. 5 --- Motor Rotation Direction Control

The brushed DC motor generates torque directly from the DC power supply. By changing the direction of the current that flows through the motor coil, we can control the motor's rotating direction. The following truth tables show how to control the current direction through the built-in dual channel H-Bridge Driver.

Input		Output		LEFT
P8	P9	L1	L2	WHEEL
0	0	∞	∞	STOP
0	1	L	H	BACKWARD
1	0	H	L	FORWARD
1	1	L	L	BRAKE

Input		Output		RIGHT
P10	P11	R1	R2	WHEEL
0	0	∞	∞	STOP
0	1	L	H	BACKWARD
1	0	H	L	FORWARD
1	1	L	L	BRAKE

For instance, if we set pin 8 to logic 1 and pin 9 to logic 0, then the output pin of H-Bridge Driver L1 will be high and pin L2 will be low. When the L1 and L2 pins are connected to a motor, the current will flow into the motor from L1 and flow out from L2. If you have connected the pins and installed the motors as described in the assembly section, the left wheel will move forward.

Similarly, we can set the pins to either STOP or BRAKE the DC motor. For the STOP condition, the motor will stop gradually due to the residue inductive current. For the BRAKE condition, the motor stops immediately as the residue inductive current will be grounded. However, in our Line Follower application, you can hardly see the difference between the two cases. You may use either setting to stop the motor.

This program shows how to control the rotation directions of the motors by inputting the command via the Terminal Window. If the motors do not rotate as expected, check the assembly procedure again.

```
'-----
' Innoxer 8
```

```
' Ex.5: Motor Rotation Direction Control
' Function: Use H-Bridge to control motors
' -----
```

```
Sub main()
Dim i As Byte
Debug "Input number to control motors.",CR
Debug "1: forward",CR
Debug "2: backward",CR
Debug "3: turn right",CR
Debug "4: turn left",CR
Debug "5: brake",CR
```

```
Do
  Debugin i
  Select Case i
    Case 1      'forward
      High 8
      Low 9
      High 10
      Low 11
    Case 2      'backward
      Low 8
      High 9
      Low 10
      High 11
    Case 3      'turn right
      High 8
      Low 9
      Low 10
      High 11
    Case 4      'turn left
      Low 8
      High 9
      High 10
      Low 11
    Case 5      'brake
      High 8
      High 9
```

```
        High 10
        High 11
    End Select
Loop
End Sub
```

Ex. 6 --- Motor Speed Control

In order to have a more precise control of the direction, we also need to control the rotation speed of the DC motor. The rotation speed is in proportion with the current flows into the motor coil. The PWM (Pulse Width Modulation) is a commonly used technique for controlling the power delivered to the motor.

The following program demonstrates how to write PWM as a subroutine, which performs a specific task when it is invoked. It is a useful programming technique to reduce the program size by reusing the repeated program code and also to make the program code easy to maintain. You will see more subroutines in the following programs.

```
'-----
' InnoXer 8
' Ex.6: Motor Speed Control
' Function: Use PWM method to control forward speed
'-----
Sub main()
Dim Speed As Byte

Debug "Please enter speed ranging (1~10)"
Do
    KeyIn Speed
    Debug CSRXY(1,2), "Speed : ", Speed, CLREOL
    Forward(Speed)
Loop
End Sub

Sub Forward(Hightime As Byte)
    Dim Lowtime As Byte
```



```

Lowtime = 10 - Hightime
High 8
Low 9
High 10
Low 11
Pause Hightime

High 8
High 9
High 10
High 11
Pause Lowtime
End Sub

```

Ex. 7 --- Follow Line

In this program, we use the technique that we learned in the exercise of measuring the infrared intensity on how to set the central value of the black line on the white surface, in order to find out the line position under the infrared sensors. Then we learned how to use the PWM technique to set the motors speeds to keep the Line Follower on the track based on different track conditions.

The following table shows all the conditions that may be encountered on a track. In the table, “1” means the black line is detected by the I/O pin. If “1” appears in two cells, it means both I/Os have detected the black line, in other words, the black line is between the two sensors. In the Action column, it shows the actions to be taken to follow the line. Note that case “000”, “111” and “101” are out of control exceptions. In those cases, we need to stop the motors to prevent the Innoxer™ 8 Line Follower from running away.

Case	Left(P18)	Center(P17)	Right(P16)	Action
010	0	1	0	Forward
110	1	1	0	Left Turn
100	1	0	0	Sharp Left Turn

001	0	0	1	Sharp Right Turn
011	0	1	1	Right Turn
000	0	0	0	Stop
111	1	1	1	Stop
101	1	0	1	Stop

After downloading the program to the Innoxer™ 8 Line Follower, remove the USB cable and press the button on pin P14 to start the Innoxer™ 8 Line Follower.

```

'-----
' Innoxer 8
' Ex.7: Follow Line
' Function: Use the three IR sensors to follow line
'-----

Sub main()
Dim IR_All As Byte
Dim status As Byte

Do                                'wait until button pressed
    status = In(14)                '
Loop Until status = 0            '

Do
    High 16                        'get right IR reflection intensity
    Rctime 16,1,IR_R              '
    Low 16                         '

    High 17                        'get central IR reflection intensity
    Rctime 17,1,IR_C              '
    Low 17                         '

    High 18                        'get left IR reflection intensity
    Rctime 18,1,IR_L              '
    Low 18                         '

```

```
IRALL = 0

If IR_R>12 Then IR_ALL=1           'left IR result
If IR_C>12 Then IR_ALL=IR_ALL+10   'add central IR result
If IR_L>12 Then IR_ALL=IR_ALL+100  'add left IR result

Select Case IR_All
    Case 001           'line under right IR sensor
        TurnR2()      'sharp right turn
    Case 011           'line between right and center IR sensor
        TurnR()       'right turn
    Case 010           'line under center IR sensor
        Forward()     'go straight
    Case 110           'line between left and center IR sensor
        TurnL()       'left turn
    Case 100           'line under left IR sensor
        TurnL2()      'sharp left turn
End Select

Loop
End Sub

'-----
Sub Forward() 'go forward subroutine
    High 8     'both motors off period
    High 9     '
    High 10    '
    High 11    '
    Pause 6    'motors off about 6ms

    High 8     'both motors on period
    Low 9      '
    High 10    '
    Low 11     '
    Pause 3    'motors on about 6ms
End Sub

'-----
Sub Brake()    'brake subroutine
    High 8
    High 9
```

```
High 10
High 11
End Sub
'-----
Sub Stop()      'stop subroutine
  Low 8
  Low 9
  Low 10
  Low 11
End Sub
'-----
Sub TurnR()     'left turn subroutine
  High 8        'both motors off period
  High 9        '
  High 10       '
  High 11       '
  Pause 6       'motors off about 6ms

  High 8        'left motor on
  Low 9         '
  High 10       'right motor off
  High 11       '
  Pause 2       'about 2ms
End Sub
'-----
Sub TurnL()     'right turn subroutine
  High 8        'both motors off period
  High 9        '
  High 10       '
  High 11       '
  Pause 6       'motors off about 6ms

  High 8        'left motor off
  High 9        '
  High 10       'right motor on
  Low 11        '
  Pause 2       'about 2ms
End Sub
```

```
'-----  
Sub TurnR2() 'sharp right turn subroutine  
  High 8 'both motors off period  
  High 9 '  
  High 10 '  
  High 11 '  
  Pause 6 'motors off about 6ms  
  
  High 8 'left motor forward  
  Low 9 '  
  Low 10 'right motor backward  
  High 11 '  
  Pause 2 'about 2 ms  
End Sub  
  
'-----  
Sub TurnL2() 'sharp left turn subroutine  
  High 8 'both motors off period  
  High 9 '  
  High 10 '  
  High 11 '  
  Pause 6 'motors off about 6ms  
  
  Low 8 'left motor backward  
  High 9 '  
  High 10 'right motor forward  
  Low 11 '  
  Pause 2 'about 2 ms  
End Sub  
  
'-----
```

Ex. 8 --- PID Control

The PID (proportional–integral–derivative) control is the most commonly used technique in industrial control systems. The PID control can be interpreted as a weighted sum of present error (P), accumulation of past errors (I), and prediction of future errors (D), based on current changing rate. The PID control can be used to make the line following more smoothly. In the line following program, no final stable state needs to be achieved,

therefore the accumulation of past errors (I) is not used.

Although the PID control technique is for experienced users, you may download the program to check the difference after deploying the PID control technique.

```
'-----  
' InnoXer 8  
' Ex.8: PID  
' Function: Use the PID to follow line  
'-----  
  
#DEFINE KP      1          'set PID parameters, KI not used  
#DEFINE KD      10  
#DEFINE NORMAL_SPEED_R  250  'set right wheel speed  
#DEFINE NORMAL_SPEED_L  250  'set left wheel speed  
  
Dim PerRCRight,PerRCCenter,PerRCLeft As Persistentword  
Dim RCRight,RCCenter,RCLeft As Word  
Dim RCRightMax,RCCenterMax,RCLeftMax As Word  
Dim RCRightMin,RCCenterMin,RCLeftMin As Word  
  
Dim R, L As Integer          'right and left wheel speed  
Dim Err As Integer           'deviation  
Dim IR0,IR1,IR2,IRAll As Byte 'IR reading status  
Dim But0,But1 As Byte        'Key buttons  
Dim TempR,TempC,TempL As Word  
Dim FollowLine,Other As Byte  
  
Sub main()  
  
START:  
  
Pause 1000  
    Setinitial()  
    Low 21  
    Low 22  
    Do  
        But0 = In(14)          'press key 14 for calibration  
        But1 = In(15)          'press key 15 to follow line
```

```
    If But0 = 0 Then
        Goto CalibrationIR
    ElseIf But1 = 0 Then
        Goto Run
    End IF
Loop
CalibrationIR:
    High 21
    Low 22
    Pause 1000
    Do
        IRinitial()
        But0 = In(14)
    Loop Until But0 = 0    'press key 14 to end calibration
    PerRCRight = (RCRightMax + RCRightMin)\3
    PerRCCenter = (RCCenterMax + RCCenterMin)\3
    PerRCLeft = (RCLeftMax + RCLeftMin)\3
    Goto START
Run:
    High 22
    Low 21
    Do
        GetRCTime()      'IR intensity with RCTIME commands
    Select Case IRAll
        Case 001
            Err = -50
        Case 011
            Err = -30
        Case 010
            Err = 0
        Case 110
            Err = 30
        Case 100
            Err = 50
    End Select
    PID()
Loop
End Sub
```

```
Sub Setinitial()  
    Setdirport1 &B11110000  
    But0 = 0  
    But1 = 0  
    RCRightMax = 0  
    RCCenterMax = 0  
    RCLeftMax = 0  
    RCRightMin = 32767  
    RCCenterMin = 32767  
    RCLeftMin = 32767  
    RCRight = PerRCRight  
    RCCenter = PerRCCenter  
    RCLeft = PerRCLeft  
End Sub  
  
Sub IRinitial()  
    Rctime 16,1,TempR  
    Rctime 17,1,TempC  
    Rctime 18,1,TempL  
    High 16  
    High 17  
    High 18  
  
    If TempR > RCRightMax Then  
        RCRightMax = TempR  
    End IF  
    If TempR < RCRightMin Then  
        RCRightMin = TempR  
    End IF  
    If TempC > RCCenterMax Then  
        RCCenterMax = TempC  
    End IF  
    If TempC < RCCenterMin Then  
        RCCenterMin = TempC  
    End IF  
    If TempL > RCLeftMax Then  
        RCLeftMax = TempL
```



```
End IF
If TempL < RCLeftMin Then
    RCLeftMin = TempL
End IF
End Sub

Sub GetRctime()
    Rctime 16,1,TempR
    Rctime 17,1,TempC
    Rctime 18,1,TempL
    High 16
    High 17
    High 18

    If TempR > RCRight Then
        IR0 = 1
    ELSE
        IR0 = 0
    End IF

    If TempC > RCCenter Then
        IR1 = 1
    ELSE
        IR1 = 0
    End IF

    If TempL > RCLeft Then
        IR2 = 1
    ELSE
        IR2 = 0
    End IF

    IRAll = (IR2 * 100) + (IR1 * 10) + IR0
End Sub

Sub GetButton()
    But0 = In(0)
    But1 = In(1)
```

```
End Sub

Sub MotorControl(SpeedL As Long,SpeedR As Long)
    Dim Speed As Word
    Dim Motor As Byte
    If SpeedR >= 0 Then
        Motor = &B11110111
    ELSE
        Motor = &B11111011
        SpeedR = - SpeedR
    End IF

    If SpeedL >= 0 Then
        Motor = Motor And &B11111101
    ELSE
        Motor = Motor And &B11111110
        SpeedL = - SpeedL
    End IF

    For Speed = 1 To 512
        If Speed > SpeedR Then
            Motor = Motor And &B11110011
        End IF
        If Speed > SpeedL Then
            Motor = Motor And &B11111100
        End IF
        Writeport1(Motor)
    Next
End Sub

Sub PID()
    Dim Derivative As Integer 'derivative value
    Dim PreErr As Integer 'previous error
    Dim Control As Integer 'PID control result

    Derivative = Err - PreErr
    Control = (KP * Err) + (KD * Derivative)
    PreErr = Err
    R = NORMAL_SPEED_R + Control 'adjust left and right speed
```

```
L = NORMAL_SPEED_L - Control
If R>512 Then           'check and set left and right max speed
    R = 512
Elseif R<-512 Then
    R = -512
End If
If L>512 Then
    L = 512
Elseif L<-512 Then
    L = -512
End IF
MotorControl(L,R)
End Sub
```