

AN-NAJAH NATIONAL UNIVERSITY
FACULTY OF ENGINEERING
TELECOMMUNICATION ENGINEERING DEPARTMENT



Networking Lab

10646540

For

Telecommunication Engineering

Review and prepare

Dr. Saed Tarapiah
Eng. Monir Aghbar
Eng. Walaa Hammoudeh

2018/2019



جامعة النجاح الوطنية
كلية الهندسة

An-Najah National University
Faculty of Engineering

Communication Engineering Department
Networking Lab (10646540)
Lab Report

Instructor Name: Dr. Saed TARAPIAH Student Name(1): _____
Teacher Assistant: Eng. Walaa Hammoudeh Student Name(2): _____
Academic Year: 2018/2019 Student Name(3): _____
Semester: Fall Section: _____
Credit Hours: 1
Total Report Mark: 10

Experiment Number: _____
Experiment Name: _____
Experiment Date : _____

Criteria	Points	ILO's	ILO's %	Item Grade	Required Time
Writing skills, Function in team	2	4	100%		One Week
Data collection, procedure	3	2,3	80%,20%		One Week
Experimental results, Discussion	5	1,2,3	80%,10%,10%		One Week
Student Grade					

Table of Contents

Course Outline	i
Lab Safety Guidelines	ii
General Lab Report Format	iv
Experiments Groups	vii
Exp-1 Network Cables	1
Exp-2 Simple Network	10
Exp-3 TCP/IP Workstation Configuration, Telnet & FTP	14
Exp-4 Wireless Networks configuration 802.11 b/g/n	20
Exp-5 GSM Transmitting/Receiving part	24
Exp-6 Phases of telephony connection	48
Exp-7 Bluetooth discovery/discoverability	54
Exp-8 Connecting Bluetooth devices	71
Exp-9 GPS Module Settings	86
Exp-10 GSM/GPRS/GPS module, SMS-AT command	107
Exp-11 GSM/GPRS/GPS module, Connecting to the internet	117
Exp-12 Wifi Module Programming	127

Department of Telecommunication Engineering

Telecommunication Networking Lab (10646540)

Total Credits 1

major compulsory

Prerequisites P1 : Telecommunication networks (10646444)

Course Contents

This lab-based course examines certain telecommunication networks, such as the fixed telephony networks, central communication networks, computer networks and data communication including network services and applications.

Intended Learning Outcomes (ILO's)	Student Outcomes (SO's)	Contribution
1 An ability to apply knowledge of networking theory in practice	B	30 %
2 An ability to design different kinds on telecommunication networks	C	40 %
3 Ability to simulate networking experiment using different simulation Software.	K	15 %
4 Ability to function in teams.	D	15 %

Textbook and/ or References

Lab manual, Machines manuals and several handouts throughout the semester.

Assessment Criteria	Percent (%)
Reports	40 %
Laboratory Work	30 %
Final Exam	30 %

Course Plan

Week	Topic
1	Network Cables
2	Simple Network
3	TCP/IP Workstation Configuration, Telnet and FTP
4	Wireless Networks configuration 802.11 b/g/n
5	GSM Transmitting/receiving part
6	Phases of telephony connection / PSTN switching system
7	Bluetooth discovery/ discoverability
8	Connecting Bluetooth devices
9	GPS Module Settings
10	GSM/GPRS/GPS module, SMS-AT command
11	GSM/GPRS/GPS module, connecting to the internet
12	WiFi Module Programming

Lab Safety Guidelines

- 1) Be familiar with the electrical hazards associated with your workplace.
- 2) You may enter the laboratory only when authorized to do so and only during authorized hours of operation.
- 3) Be as careful for the safety of others as for yourself. Think before you act, be tidy and systematic.
- 4) Avoid bulky, loose or trailing clothes. Avoid long loose hair.
- 5) Food, beverages and other substances are strictly prohibited in the laboratory at all times. Avoid working with wet hands and clothing.
- 6) Use extension cords only when necessary and only on a temporary basis.
- 7) Request new outlets if your work requires equipment in an area without an outlet.
- 8) Discard damaged cords, cords that become hot, or cords with exposed wiring.
- 9) Before equipment is energized ensure, (1) circuit connections and layout have been checked by a laboratory technician and (2) all colleagues in your group give their assent.
- 10) Know the correct handling, storage and disposal procedures for batteries, cells, capacitors, inductors and other high energy-storage devices.
- 11) Experiments left unattended should be isolated from the power supplies. If for a special reason, it must be left on, a barrier and a warning notice are required.
- 12) Equipment found to be faulty in any way should be reported to the laboratory technician immediately and taken out of service until inspected and declared safe.
- 13) Never make any changes to circuits or mechanical layout without first isolating the circuit by switching off and removing connections to power supplies.
- 14) Know what you must do in an emergency, i.e. Emergency Power Off
- 15) For microwave and antenna trainer:
 - a. You should, whenever possible, remove the power from the gun oscillator before placing yourself in front of transmitting antenna.
 - b. For your safety, do not look directly into the waveguides or horn antennas while power is being supplied by the gun oscillator. Because, although the microwave is invisible, it can be dangerous at high levels or long exposure times.
- 16) For fiber optics trainer:
 - a. Do not look inside the connector of the Optical Sources when these are operating. Although nothing can be seen, as the emitted wavelength should be out of the visible range, it can be dangerous for your sight.

- b. Do not bend the optical cables with too narrow curves, as the fiber inside should cut off or damage. The minimum curving ray is around 2 cm;
- c. Sometimes clean the connectors' head with a cotton wad soaked with alcohol;

Electrical Emergency Response

The following instructions provide guidelines for handling two types of electrical emergencies:

1. Electric Shock:

When someone suffers serious electrical shock, he or she may be knocked unconscious. If the victim is still in contact with the electrical current, immediately turn off the electrical power source. If you cannot disconnect the power source, depress the Emergency Power Off switch.



IMPORTANT:

Do not touch a victim that is still in contact with a live power source; you could be electrocuted.

Have someone call for emergency medical assistance immediately. Administer first-aid, as appropriate.

2. Electrical Fire:

If an electrical fire occurs, try to disconnect the electrical power source, if possible. If the fire is small and you are not in immediate danger; and you have been properly trained in fighting fires, use the correct type of fire extinguisher to extinguish the fire. When in doubt, push in the Emergency Power Off button.

NEVER use water to extinguish an electrical fire.

Lab Report Format

Following the completion of each laboratory exercise, a report must be written and submitted for grading. The purpose of the report is to completely document the activities of the design and demonstration in the laboratory. Reports should be complete in the sense that all information required to reproduce the experiment is contained within. Writing useful reports is a very essential part of becoming an engineer. In both academic and industrial environments, reports are the primary means of communication between engineers.

There is no one best format for all technical reports but there are a few simple rules concerning technical presentations which should be followed. Adapted to this laboratory they may be summarized in the following recommended report format:

- ABET Cover Page
- Title page
- Introduction
- Experimental Procedure
- Experimental Data
- Discussion
- Conclusions

Detailed descriptions of these items are given below.

Title Page:

The title page should contain the following information

- Your name
- ID
- Experiment number and title
- Date submitted
- Instructors Name

Introduction:

It should contain a brief statement in which you state the objectives, or goals of the experiment. It should also help guide the reader through the report by stating, for example, that experiments were done with three different circuits or consisted of two parts etc. Or that additional calculations or data sheets can be found in the appendix, or at the end of the report.

The Procedure

It describes the experimental setup and how the measurements were made. Include here circuit schematics with the values of components. Mention instruments used and describe any special measurement procedure that was used.

Results/Questions:

This section of the report should be used to answer any questions presented in the lab hand-out. Any tables and /or circuit diagrams representing results of the experiment

should be referred to and discussed / explained with detail. All questions should be answered very clearly in paragraph form. Any unanswered questions from the lab hand-out will result in loss of points on the report.

The best form of presentation of some of the data is graphical. In engineering presentations a figure is often worth more than a thousand words. Some simple rules concerning graphs and figures which should always be followed. If there is more than one figure in the report, the figures should be numbered. Each figure must have a caption following the number. For example, "*Figure 1.1:DSB-SC* " In addition, it will greatly help you to learn how to use headers and figures in MS Word.

The Discussion

It is a critical part of the report which testifies to the student's understanding of the experiments and its purpose. In this part of the report you should compare the expected outcome of the experiment, such as derived from theory or computer simulation, with the measured value. Before you can make such comparison you may have to do some data analysis or manipulation.

When comparing experimental data with numbers obtained from theory or simulation, make very clear which is which. It does not necessarily mean that your experiment was a failure. The results will be accepted, provided that you can account for the discrepancy. Your ability to read the scales may be one limitation. The value of some circuit components may not be well known and a nominal value given by the manufacturer does not always correspond to reality. Very often, however, the reason for the difference between the expected and measured values lies in the experimental procedure or in not taking into account all factors that enter into analysis.

Conclusion:

A brief conclusion summarizing the work done, theory applied, and the results of the completed work should be included here. Data and analyses are not appropriate for the conclusion.

Notes

Typed Reports are required. Any drawings done by hand must be done with neatness, using a straightedge and drawing guides wherever possible.

Freehand drawings will not be accepted.

Experiments distribution over groups

	Group 1	Group 2	Group 3	Group 4	Group 5
Week 1	Exp 1	Exp 1	Exp 1	Exp 1	Exp 1
Week 2	Exp 2	Exp 2	Exp 2	Exp 2	Exp 2
Week 3	Exp 3	Exp 3	Exp 3	Exp 3	Exp 3
Week 4	Exp 4	Exp 4	Exp 4	Exp 4	Exp 4
Week 5	Exp 5	Exp 5	Exp 12	Exp 12	Exp 12
Week 6	Exp 12	Exp 12	Exp 5	Exp 5	Exp 5
Week 7	Exp 6	Exp 6	Exp 6	Exp 9	Exp 9
Week 8	Exp 9	Exp 9	Exp 9	Exp 6	Exp 6
Week 9	Exp 7	Exp 7	Exp 7	Exp 10	Exp 10
Week 10	Exp 10	Exp 10	Exp 8	Exp 7	Exp 7
Week 11	Exp 8	Exp 8	Exp 10	Exp 11	Exp 11
Week 12	Exp 11	Exp 11	Exp 11	Exp 8	Exp 8

AN-NAJAH NATIONAL UNIVERSITY
FACULTY OF ENGINEERING
Communication ENGINEERING DEPARTMENT
Dr. Saed Tarapiah & Eng. Monir Aghbar

Experiment # 1
Network Cables

Introduction:

UTP Ethernet Cabling: Cabling is one of the most critical areas of network design and implementation. The cabling is expected to last from 10 to 15 years. The quality of cable and connections is a major factor in reducing network problems and time spent troubleshooting. Unshielded Twisted Pair (UTP) copper cable is the most common cable used in Ethernet networks. There are various Categories (CAT 3, CAT 5, CAT 5e etc.) but all of them contain 8 wires or conductors and use RJ45 connectors. A UTP patch cable in a network is usually wired as a straight-thru or crossover. In order to follow proper specifications, all 8 conductors must be used even though with earlier versions of Ethernet, not all 8 conductors were used. You will create these cables in this lab. Also you will learn how to use Cable Testers to examine cable connectivity.

1. 1 Straight-Thru Cable

Objectives:

Build a straight-through Ethernet patch cable to T568-B (OR T568-A) standards for connection from workstation to hub/switch or patch panel to hub/switch.

Background:

In this lab you will learn how to build a Category 5 (CAT 5) Unshielded Twisted Pair (UTP) Ethernet network patch cable (or patch cord) and test it for good connections (continuity) and correct pin outs (correct color of wire on the right pin). This will be a 4-pair (8-wires) "straight through" cable. It will be wired to (TIA/EIA-568-B or A) standards for 10Base-T Ethernet which determines what color wire is on each pin. T568-B (also called AT&T specification) is more common, but many installations are also wired to T568-A (also called ISDN).

This patch cable will conform to the structured cabling standards and is considered to be part of the "horizontal" cabling which is limited to 99 meters total between workstation and hub or switch. It can be used in a workstation area to connect the workstation NIC to the wall plate data jack or it can be used in the wiring closet to connect the patch panel (horizontal cross connect) to an Ethernet hub or switch. Patch cables are wired straight thru since the cable from the workstation to the hub or switch is normally crossed over automatically at the switch or the hub. Note that the ports on most hubs have an X next to them. This means the send and receive pairs will be crossed when the cabling reaches the switch. The pin outs will be T568-B and all 8 conductors (wires) should be terminated with RJ45 modular connectors (only 4 of the 8 wires are used for 10/100base-T Ethernet, all 8 are used for 1000Base-T Ethernet).

Apparatus required:

The following resources will be required:

- Two to three foot length of Cat 5 cabling (one per person or one per team)
- Four RJ45 connectors (two extra for spares)
- RJ45 crimping tools to attach the RJ45 connectors to the cable ends
- Ethernet cabling continuity tester which can test straight-thru or crossover type cables (T568-A or T568-B).
- Wire cutters

Experimental procedures**Step 1: Cabling Information.**

Explanation: Instructions are provided here for building a T568-A or T568-B cable. Either can be used as long as all connections (pin outs) from the workstation to the wiring closet and terminating electronics (hubs or switches) are consistent. If cables are to be built for an existing network it is important to keep the same standard as already exists (either T568-A or B). A patch cable that is wired "straight through" will have the same color of wire on the same pin (1 – 8) at both ends. A straight through patch cable (T568-A or B) can be used to connect a PC workstation to a wall plate in a work area or it can be used to connect from a patch panel in a wiring closet to a hub or a switch. A PC can also be connected directly to a port on a hub or switch with this cable. If a cable will be used to connect from an "uplink" port on one hub to a "crossover" front port on another hub then a straight through cable should be used.

Step 2: Create a T568-B straight-thru patch panel cable.

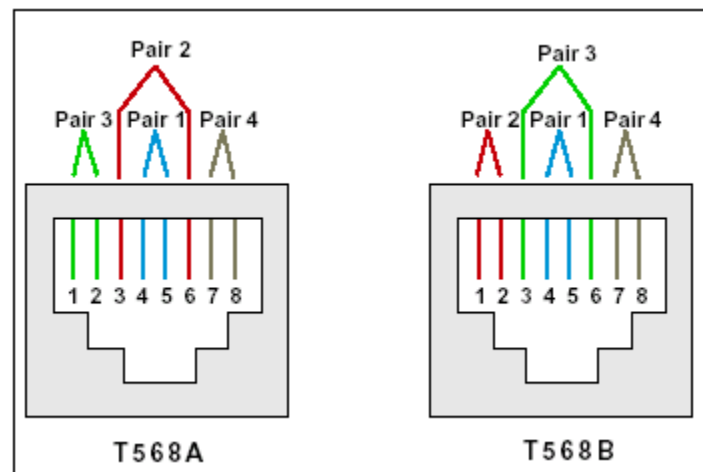
Task: Use the following tables and diagrams and steps to create a T568-B patch panel cable.

Explanation: Both cable ends should be wired the same when looking at the conductors. Only four wires are used with 10Base-T or 100Base-TX Ethernet:

T568-B Cabling

Pin#	Pair#	Function	Wire Color	Used with 10/100 Base-T Ethernet?	Used with 100 Base-T4 and 1000 Base-T Ethernet?
1	2	Transmit	White/Orange	YES	YES
2	2	Transmit	Orange/White	YES	YES
3	3	Receive	White/Green	YES	YES
4	1	Not used	Blue/White	NO	YES
5	1	Not used	White/Blue	NO	YES
6	3	Receive	Green/White	YES	YES
7	4	Not used	White/Brown	NO	YES
8	4	Not used	Brown/White	NO	YES

Diagram showing both T568-A and T568-B cabling wire colors



1. Determine the distance between devices, or device and plug, then add at least 12" to it. The maximum length for this cord is 3 m; standard lengths are 6' and 10'.
2. Cut a piece of stranded Cat 5 unshielded twisted-pair cable to the determined length. You will use stranded cable for patch cables because it is more durable when bent repeatedly. Solid wire is fine for cable runs that are punched down into jacks.
3. Strip 2" of jacket off of one end of the cable.
4. Hold the 4 pairs of twisted cables tightly where jacket was cut away and then reorganize the cable pairs into the order of the 568-B wiring standard. Take care to maintain the twists since this provides noise cancellation (orange pair, green pair, blue pair and brown pair).
5. Hold the jacket and cable in one hand; untwist a short length of the green and blue pairs, and reorder them to reflect the 568-B wiring color scheme. Untwist and order the rest of the wire pairs according to the color scheme.
6. Flatten, straighten, and line up the wires, then trim them in a straight line to within 1/2" - 3/4" from the edge of the jacket. Be sure not to let go of the jacket and the wires, which are now in order! You should minimize the length of untwisted wires because overly-long sections that are near connectors are a primary source of electrical noise.
7. Place an RJ-45 plug on the end of the cable, with the prong on the underside and the orange pair to the left side of the connector.
8. Gently push the plug onto wires until you can see the copper ends of the wires through the end of the plug. Make sure the end of the jacket is inside the plug and all wires are in the correct order. If the jacket is not inside the plug, it will not be properly strain relieved and will eventually cause problems. If everything is correct, crimp the plug hard enough to force the contacts through the insulation on the wires, thus completing the conducting path.
9. Repeat steps 3-8 to terminate the other end of the cable, using the same scheme to finish the straight through cable.
10. Test the finished cable and have the instructor check it. How can you tell if your cable is functioning properly?

1. 2 Cross Over Cable

Objectives:

Build a crossover Ethernet patch cable to T568-B (or T-568-A) standards for connection from workstation to workstation or from switch to switch.

Background:

In this lab you will learn how to build a Category 5 (CAT 5) Unshielded Twisted Pair (UTP) Ethernet crossover network cable and test it for good connections (continuity) and correct pin outs (correct color of wire on the right pin). This will be a 4-pair (8-wires) "crossover" cable which means that pairs 2 and 3 on one end of the cable will be reversed on the other end. It will be wired to TIA/EIA-568-B and A standards for 10Base-T Ethernet which determines what color wire is on each pin. The pin outs will be T568-A on one end and T568-B on the other end. All 8 conductors (wires) should be terminated with RJ45 modular connectors. This patch cable will conform to the structured cabling standards and, if it is used between hubs or switches, is considered to be part of the "vertical" cabling also known as backbone cable. A crossover cable can be used as a backbone cable to connect two or more hubs or switches in a LAN or to connect 2 isolated workstations to create a mini-LAN. This will allow you to connect two workstations together or a server and a workstation without the need for a hub between them. This can be very helpful for training and testing.

Apparatus required:

The following resources will be required:

- Two to three foot length of Cat 5 cabling (one per person or one per team)
- Four RJ45 connectors (two extra for spares)
- RJ45 Crimping Tools to attach the RJ45 connectors to the cable ends
- Ethernet cabling continuity tester which can test crossover type cables (T568-A to T568-B).
- Wire cutters

Experimental Procedures

Step 1: Create a crossover patch panel cable.

Use the following tables and diagrams and steps to create a crossover cable. One end of the cable should be wired to the T568-A standard and the other end to the T568-B standard. This crosses the transmit and receive pairs (2 and 3) to allow communication to take place. Only four wires are used with 10 Base-T or 100 Base-TX Ethernet:

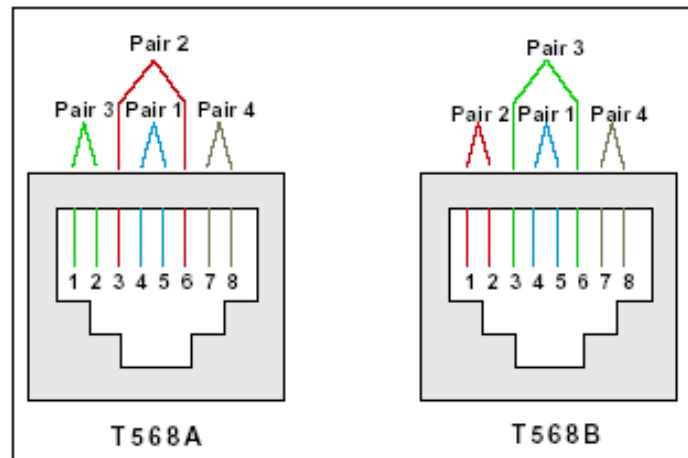
T568-A Cabling

Pin#	Pair#	Function	Wire Color	Used with 10/100 Base-T Ethernet?	Used with 100 Base-T4 and 1000 Base-T Ethernet?
1	3	Transmit	White/Green	YES	YES
2	3	Transmit	Green/White	YES	YES
3	2	Receive	White/Orange	YES	YES
4	1	Not used	Blue/White	NO	YES
5	1	Not used	White/Blue	NO	YES
6	2	Receive	Orange/White	YES	YES
7	4	Not used	White/Brown	NO	YES
8	4	Not used	Brown/White	NO	YES

T568-B Cabling

Pin#	Pair#	Function	Wire Color	Used with 10/100 Base-T Ethernet?	Used with 100 Base-T4 and 1000 Base-T Ethernet?
1	2	Transmit	White/Orange	YES	YES
2	2	Transmit	Orange/White	YES	YES
3	3	Receive	White/Green	YES	YES
4	1	Not used	Blue/White	NO	YES
5	1	Not used	White/Blue	NO	YES
6	3	Receive	Green/White	YES	YES
7	4	Not used	White/Brown	NO	YES
8	4	Not used	Brown/White	NO	YES

Diagram showing both T568-A and T568-B cabling wire colors



1. Determine the distance between devices, or device and plug, then add at least 12" to it. The maximum length for this cord is 3 m; standard lengths are 6' and 10'.
2. Cut a piece of stranded Cat 5 unshielded twisted-pair cable to the determined length. You will use stranded cable for patch cables because it is more durable when bent repeatedly. Solid wire is fine for cable runs that are punched down into jacks.
3. Strip 2" of jacket off of one end of the cable.
4. Hold the 4 pairs of twisted cables tightly where jacket was cut away, and then reorganize the cable pairs into the order of the 568-B wiring standard. Take care to maintain the twists since this provides noise cancellation (orange pair, green pair, blue pair and brown pair).
5. Hold the jacket and cable in one hand; untwist a short length of the green and blue pairs, and reorder them to reflect the 568-B wiring color scheme. Untwist and order the rest of the wire pairs according to the color scheme.
6. Flatten, straighten, and line up the wires, then trim them in a straight line to within 1/2" - 3/4" from the edge of the jacket. Be sure not to let go of the jacket and the wires, which are now in order! You should minimize the length of untwisted wires because overly-long sections that are near connectors are a primary source of electrical noise.
7. Place an RJ-45 plug on the end of the cable, with the prong on the underside and the orange (green on the 586-A end) pair to the left side of the connector.
8. Gently push the plug onto wires until you can see the copper ends of the wires through the end of the plug. Make sure the end of the jacket is inside the plug and all wires are in the correct order. If the jacket is not inside the plug, it will not be properly strain relieved and will eventually cause problems. If everything is correct, crimp the plug hard enough to force the contacts through the insulation on the wires, thus completing the conducting path.
9. Repeat steps 3-8 to terminate the other end of the cable, using the 568-A scheme to finish the crossover cable.
10. Test the finished cable and have the instructor check it. How can you tell if your cable is functioning properly?

1.3 Rollover Cable

Objectives:

Build a rollover cable for connection from a workstation to the console port on a router or switch.

Background:

In this lab you learn how to build a Category 5 (CAT 5) Unshielded Twisted Pair (UTP) console rollover cable and test it for good connections (continuity) and correct pin outs (correct wire on the right pin). This will be a 4-pair (8-wires) "rollover" cable.

This cable should be approximately 10 feet in length but can be as long as 25 feet. It can be used to connect a workstation or dumb terminal to the console port on the back of a router or Ethernet switch in order to be able to configure the router or switch. This cable uses an asynchronous serial interface to the router or switch (8 data bits, No parity and 2 Stop bits). Both ends of the cable you build will have RJ45 connectors on them. One end plugs directly into the RJ45 console management port on the back of the router or switch and the other end plugs into an RJ45-to-DB9 terminal adapter. This adapter converts the RJ 45 to a 9-pin female D connector that plugs into the DB9 serial port male adapter on the back of a PC running terminal emulation software such as HyperTerminal. A DB25 terminal adapter is also available to connect with a dumb terminal which has a 25 pin connector.

A rollover cable uses 8 pins but is different from the straight-through cable or crossover cable that you will build in other labs. With a rollover cable, pin 1 on one end connects to pin 8 on the other end. Pin 2 connects to pin 7, pin 3 connects to pin 6 and so on. This is why it is referred to as a rollover since the pins on one end are all reversed on the other end as though one end of the cable was just rotated or rolled over.

A flat black rollover cable comes with each new router or switch along with the terminal adapters for both DB9 and DB25 connections to terminals or PC serial ports. It is approximately 8 feet long. This lab will enable you to build another cable if the one that comes with the router or switch is damaged or lost. It will also allow you to connect to routers or switches from workstations that are greater than 8 feet away by building your own longer cables.

Apparatus required:

The following resources will be required:

- 10 to 20 foot length of Cat 5 cabling (one per person or one per team)
- Four RJ45 connectors (two extra for spares)
- RJ45 Crimping Tools to attach the RJ45 connectors to the cable end
- An RJ45 to DB9 female terminal adapter (available from Cisco)
- Cabling continuity tester
- Wire cutters

Experimental procedures

Step 1: Review Cable Connections and Pin Locations

Use the table as a reference to answer the questions below and to help you create a rollover console cable.

Questions:

1. Which signal on the Router port (column 1 of the table) will be used to transmit data to the PC when the PC is first connected and HyperTerminal is started (this is what displays the router prompt on the workstation.)?

2. Which pin is this connected to on the router end of the RJ45 cable?

3. Which pin is this connected to on the other end of the RJ45 cable?

4. Which pin is this connected to in the DB9 connector?

5. Which console device signal does this connect to?

6. What would happen if pin 3 on the left cable end were attached to pin 3 as with a straight-thru cable?

Rollover Console Cable Table

To connect a router or switch console port to a PC workstation, running "HyperTerminal" terminal emulation software. Console port signaling and cabling using an RJ45 rollover and DB9 Adapter.

Router or switch Console port (DTE)	RJ45 to RJ45 Rollover Cable (left end)	RJ45 to RJ45 Rollover Cable (right end)	RJ45 to DB9 Adapter	Console Device (PC workstation serial port)
Signal	From RJ45 Pin No.	To RJ45 Pin No.	DB9 Pin No.	Signal
RTS	1	8	8	CTS
DTR	2	7	6	DSR
TxD	3	6	2	RxD
GND	4	5	5	GND
GND	5	4	5	GND
RxD	6	3	3	TxD
DSR	7	2	4	DTR
CTS	8	1	7	RTS

Signal Legend: RTS = Request To Send, DTR = Data Terminal Ready, TxD = Transmit Data, GND = Ground (One for TxD and one for RxD), RxD = Receive Data, DSR = Data Set Ready, CTS = Clear To Send.

Step 2: Use the following steps to build the rollover console cable.

1. Determine the distance between devices, then add at least 12" to it. Make your cable about 10 feet unless you are connecting to router or switch from a greater distance. The maximum length for this cable is about 8m (appx 25 feet).
2. Strip 2" of jacket off of one end of the cable.
3. Hold the 4 pairs of twisted cables tightly where jacket was cut away, then reorganize the cable pairs and wires into the order of the 568-B wiring standard. You can order them in any sequence but use the 568-B sequence to become more familiar with it.
4. Flatten, straighten, and line up the wires, then trim them in a straight line to within 1/2" - 3/4" from the edge of the jacket. Be sure not to let go of the jacket and the wires, which are now in order!
5. Place an RJ-45 plug on the end of the cable, with the prong on the underside and the orange pair to the left side of the connector.
6. Gently push the plug onto wires until you can see the copper ends of the wires through the end of the plug. Make sure the end of the jacket is inside the plug and all wires are in the correct order. If the jacket is not inside the plug, it will not be properly strain relieved and will eventually cause problems. If everything is correct, crimp the plug hard enough to force the contacts through the insulation on the wires, thus completing the conducting path.
7. Repeat steps 2-6 to terminate the other end of the cable, but reversing every pair of wires as indicated in the table above. (pin 1 to pin 8, pin 2 to pin 7, pin 3 to pin 6 and so on. Alternate Method - Arrange the wires into the order of the 568-B wiring standard. Place a RJ-45 plug on the end with the prong on the top side of the connector. This method will achieve the proper reversing of every pair of wires.
8. Test the finished cable and have the instructor check it. How can you tell if your cable is functioning properly?

AN-NAJAH NATIONAL UNIVERSITY
FACULTY OF ENGINEERING
Communication ENGINEERING DEPARTMENT
Dr. Saed Tarapiah & Eng. Monir Aghbar

Experiment # 2
Simple Network

Objectives:

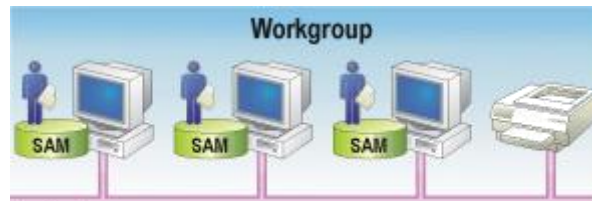
- Create a simple LAN with two PCs using a single crossover cable to connect the workstations
- Create a simple LAN with two PCs using an Ethernet hub and two straight-thru cables to connect the workstations
- Use the Control Panel / Network and Sharing Center to verify and configure the network settings
- Use the ICMP(Internet Control Message Protocol) Ping command to verify the TCP/IP connection between the two workstations
- Use the IPCONFIG.EXE utility to verify all IP configuration settings

Background:

In this lab you learn how to connect two PCs to create a simple Peer-to-Peer LAN or workgroup. You will share a folder on one workstation and connect to that folder from the other workstation.

Workgroup

A workgroup is a small group of networked computers that work together and where centralized administration is not required.



A workgroup has the following characteristics:

- Resources, administration, and authentication of users are performed on each computer in the workgroup.
- Each computer has its own local Security Accounts Manager (SAM) database, which is the local security accounts database. A user must have a user account on each computer to which she or he wants to gain access.
- There are 10 or fewer computers. These can be computers running one of the Windows 2000 server products, but each has its own SAM. Workgroups become more difficult to manage when there are more than 10 computers. In addition, the number of simultaneous connections to a computer running Windows 2000 Professional is 10.

This lab is divided into two exercises as follows:

Exercise A - The two PCs (or workstations) will be connected directly to each other from one Network Interface card (NIC) to the other NIC using a crossover cable. This can be useful to allow you to create a mini-lab for testing purposes without the need for a hub.

Exercise B – The two PCs will be connected with a hub between them. The use of a hub allows for more than just two workstations to be connected depending on the number of ports on the hub. Hubs can have anywhere from 4 to 24 ports.

Note: For both exercises A and B, you will verify that the workstations are functioning and that network hardware is installed properly. You will also need to verify and configure all TCP/IP protocol network settings for the two workstations to communicate such as IP address and subnet mask.

Apparatus required:

The following resources will be required:

- Two Pentium-based workstations with a NIC in each (NIC drivers should be available)
- Exercise A - One CAT5 Crossover cable to connect the workstations without a hub
- Exercise B - An Ethernet hub (4 or 8 port) and two CAT5 straight-wired cables

In this lab you will set up a small peer-to-peer Ethernet LAN workgroup using two workstations. Answer the following questions with each step as you check and/or configure the necessary components.

Note: Steps 1 and 2 (physical LAN connections) will be different between exercises A and B. The steps from 3 on should be the same since they relate only to the workstations and should be performed on both workstations.

Experimental Procedures:

Step 1: Check Local Area Network (LAN) Connections

Task: Verify the cables

Explanation: You should check the cables to verify that you have good layer 1 physical connections

Exercise A - A single CAT 5 crossover cable is used to connect the workstations together. Verify that the pins are wired as a crossover by holding both RJ-45 connectors side by side with the clip down and inspect them. Pairs 2 and 3 should be reversed. Refer to Experiment 1 for correct wire color and pin locations.

Exercise B - Check each of the two CAT 5 cables from each workstation to the hub. Verify that the pins are wired straight thru by holding the two RJ-45 connectors for each cable side by side with the clip down and inspect them. All pins should have the same color wire on the same pin at both ends of the cable.
(pin 1 should match pin 1 and pin 8 should match pin 8 etc.)

1. Are the cable(s) wired correctly?

Step 2: Plug in and connect the equipment

Task: Check the workstations (and hub for exercise B)

Explanation:

Exercises A and B: Check to make sure that the NICs are installed correctly in each workstation. Plug in the workstations and turn them on.

Exercise B - Plug the hub or its AC adapter into a power outlet. Plug the straight through cable from workstation 1 into port 1 of the hub and the cable from workstation 2 into port 2 of the hub. After the workstations have booted, check the green link light on the back of each NIC and the green lights on ports 1 and 2 of the hub to verify that they are communicating. This also verifies a good physical connection between the Hub and the NICs in the workstations (OSI Layers 1 and 2). If the link light is not on it usually indicates a bad cable connection, an incorrectly wired cable or the NIC or hub may not be functioning correctly.

1. Are the NIC and hub link lights on?

Step 3: Network Adapters and Protocols.

Task: Check the Network Adapter (NIC): Use the Control Panel, System, Device Manager Utility to verify that the Network Adapter (NIC) is functioning properly for both workstations. Double click on Network Adapters and then right click the NIC adapter in use. Click Properties to see if the device is working properly.

Step 4: Check the TCP/IP Protocol Settings:

Task: Use the Control Panel, select “Network and Sharing Center” and then select the “Local Area Connection” under “View your active networks” and click on details. Check the IP Address and Subnet mask for both workstations.

Explanation: The IP addresses can be set to anything as long as they are compatible and on the same network. Record the existing settings before making any changes in case they need to be set back. For this lab, use the Class C network address of 200.150.100.0

Step 5: Check the TCP/IP Settings with the IPCONFIG Utility

Task: Use the ipconfig command to see your TCP/IP settings on one screen.

Explanation: Enter the IPCONFIG to see all TCP/IP related settings for your workstation in the cmd program (click on start and then type cmd in the search field).

Step 6: Check the network connection with the Ping Utility

Task 1: Use the Ping Command to check for basic TCP/IP connectivity. Click on Start, All Programs and then Accessories and finally the Command Prompt. Enter the Ping command followed by the IP address of the other workstation.

Explanation: This will verify that you have a good OSI Layers 1 thru 3 connections.

1. What was the result of the Ping command?

Task 2: Type “Ping -a” command followed by the IP address of the other workstation in the Command Prompt.

Explanation: This will resolve the IP address to hostname as well as checking the connection between workstations.

1. What was the result of the Ping -a command?

Task 3: Type “Ping -n” command followed by any number of echo requests followed by IP address of the other workstation in the Command Prompt.

Explanation: This will send echo requests depending on the number that you put after parameter “-n”.

1. What was the result of the Ping -n command?

Task 4: Type “Ping 127.0.0.1” in the Command Prompt.

Explanation: This is another way to check the Network Adapter (NIC) through cmd, since the address 127.0.0.1 is called the loop back address.

1. What was the result of this command?
2. Is your NIC functioning properly?

Task 5: Type “Ping /?” command and see what other parameters that you can use with ping command.

Step 7: Check the network connection in Ubuntu operating system

Task: Check Network Configuration in Ubuntu: Use the terminal by pressing “Ctrl+Alt+t” at the same time and then:

1. Type “ifconfig” in the terminal and see the IP address and subnet mask.
2. Use ping command in the terminal as we used it before.

Experiment # 3

TCP/IP Workstation Configuration, Telnet & FTP

Introduction:

The purpose of this lab is design, build and configure our own networks but let's start with a working network (CNLAB) and see what we can learn by investigating current configuration parameters. By the time you are finished, you should know how to display many configuration parameters as well as know their significance. This lab covers a wide range of activities, some of which may be trivial for some students and difficult for others.

Objectives:

- Learn how to configure PC to connect to a Network.
- Learn about integrated firewall in Windows 7.
- Learn how to connect to other computers using telnet and ftp.
- Learn about ports on computer and routing.

Hardware & Software Supplied:

- A PC with Windows 7 installed.
- A network interface card (NIC).
- A twisted pair Ethernet cable.
- An Ethernet port on a hub.

Experimental procedures:

Part 1: Network Configurations.

1. Click start, click Control Panel, and then click on Network and Sharing Center.
2. Double click on "Local Area Connections" What is the speed of the connection?
3. Click on properties. What is the name of the network card?
4. What items will the connection use?
5. Click on client for Microsoft Networks. Click on install (we won't actually be installing a client). What are the three types of network components you can install? What does each of them do?
6. Double click on client. What other network client can be installed?
7. Click on cancel.
8. Double click on "Internet Protocol Version 4 TCP/IPv4". Does your machine obtain IP addresses automatically or does it use a fixed IP address?
9. If you specified an IP address what additional information would you have to provide?
10. Does your machine obtain DNS addresses automatically or does it use a fixed DNS server(s) address?
11. Click cancel (internet protocol properties window).

Part 2: Network Identification.

1. Right Click on "Computer icon" that located on Desktop.
2. Click on Properties. What is the computer's full name?
3. What workgroup is it part of?

4. Click on "Change settings" "To rename this computer or join a domain". Your computer can be a member of a workgroup or a domain.

5. Click on cancel.
6. Close the whole window.

7. Now from Control Panel, click on Network and Sharing Center and then Click on Network symbol that lies at the top of the page. What are the machines which you see listed?

8. All these machines are in what workgroup?

Part 3 - Learning about your PC's network configuration using DOS

1. Determine the following information about your PC by using the "ipconfig/all" command in the cmd.

2. Host name _____
3. Physical Address _____
4. IP Address _____
5. Default gateway IP address _____

Note: "ipconfig/all" command is used to see the MAC address (physical address) of PC.

Part 4: Firewall Configurations.

1. From Control Panel, click on Network and Sharing Center and then click on Windows Firewall (at the left bottom corner of the page). Is your computer protected by Windows Firewall?
2. Learn more about Firewall. Click on “How does a firewall help protect my computer?”, what is the Firewall?
3. How does the firewall restrict inbound traffic?
4. What does happen to the packets that arrive that are restricted from entering?
5. Click on “Allow a program or feature through Windows Firewall” that lies at the left of the page, and see how you can allow PC programs and ports through Firewall.
6. Close the Firewall window.

Part 5 - Low level network access

1. Look up the term "ping" on the internet Encyclopedia <http://www.yahoo.com/>.
2. What is the average approximate round trip time in milliseconds?
3. What is the average approximate round trip time in milliseconds to Yahoo Web Site?
4. What is the IP address of Yahoo.com?

Part 6 - Accessing another computer using TELNET and FTP

1. Using Ubuntu to configure Telnet Server and FTP, you have to be familiar with linux command .
2. Get the linux IP, using **ifconfig** command.
3. Setting up telnet server in Ubuntu
 - a. You have to install telnet server package using the following command
Sudo apt-get install telnetd

- b. To start, stop or restart telnet server, use the following command
Sudo /etc/init.d/openbsd-inetd restart
4. You should now be able to telnet the server from windows PC.
 - a. Run command prompt, Type the following commands :
 - **telnet**
 - **telnet>open ip address**
 - b. Here you will be prompted to enter the user name and password of Linux.
5. Install FTP Server on Ubuntu,
 - a. **Vsftpd** is a very secure FTP daemon available in Ubuntu. It is easy to install, set up and maintain. To install **vsftpd**, you can run the following command:
Sudo apt-get install vsftpd
 - b. You may type the following command to start vsftpd :
Sudo /etc/init.d/vsftpd start
6. You should now be able to transfer files from Windows to Ubuntu and vice versa.
 - a. In Windows, to connect to ftp server, run command prompt and write this command.
ftp "ip address"
 - b. You may use help command ' ? ' to view all available commands.
 - c. You may use **put** and **get** command to upload and download files.
7. You may use **FileZilla** program- a cross-platform graphical FTP- which has a lot of features, supporting Windows, Linux, Mac OS X and more.

Part 7 - Active ports on your computer

Ports identify which application program is running on your computer and intending to communicate using TCP/IP.

1. Download the active ports program from <http://www.protect-me.com/freeware.html>, install it, and run it.
2. List all the active ports less than 200. Find out what programs have been assigned to these known ports by searching the Internet for TCP/IP known ports assignments.

Part 8 - Routing

1. Access the www.visualware.com web site and download and install visual route. Run visual route and determine how many hops a packet makes going from your machine to cs.stanford.edu?
2. What cities does your packet go through?
3. What is the IP address of cs.stanford.edu?
4. Who is the contact for the machine and where is she/he located (click on node name)?
5. What is the average round trip packet time to cs.stanford.edu?

Part 9 – Trace Route

Trace route is a computer network diagnostic tool for displaying the route (path,hops) and measuring transit delays of packets across an Internet Protocol (IP) network.

1. Open the Command Prompt as we did before.
2. Type “tracert” followed by the IP address of the destination.
3. If you need to check the route to Google server for example then type “tracert 173.194.40.48”
4. How many routers (hops) are there to Google server?
5. In Ubuntu, to use trace route type “traceroute 173.194.40.48”

Part 10 – Domain Name System (DNS)

Network resources (Routers, Servers) are identified by IP addresses, but these IP addresses are difficult for network users to remember. The DNS database contains records that map user-friendly names for network resources to the IP address used by those resources for communication. In this way, DNS acts as a mnemonic device, making network resources easier to remember for network users.

1. Open the Command Prompt as we did before.
2. Type “nslookup” and then press enter.
3. If you need to know the IP address of Yahoo server for example then type “www.yahoo.com”, what are the IP addresses of Yahoo server?
4. Now find the IP address of Facebook server?

Experiment # 4

Wireless Networks configuration

802.11 b/g/n

Introduction:

In this lab we will learn how to configure an ADSL Wireless Router modems. Where we will use different adapters of manufacturers i.e. TP-LINK and CISCO Also we will check the network performance when using different 802.11x technologies i.e. 802.11b/g/n.

Objectives:

Learn how to configure ADSL wireless router modems, and creating you WLAN using A.P. or Ad-Hoc mode.

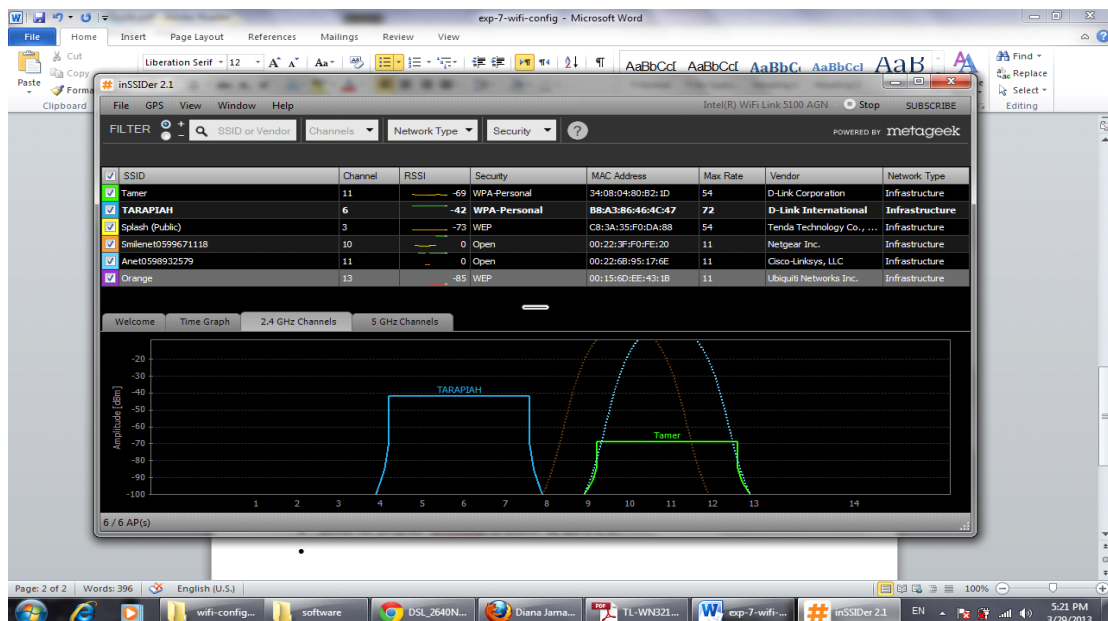
Experiment requirements:

300Mbps Portable Wireless AP/Router
54M Wireless USB Adapter (802.11b/g)
150M Wireless USB Adapter (802.11N)

Experimental procedures:

- First of all install the software driver on your machine for:
 - 1) 300Mbps Portable Wireless AP/Router
 - 2) 54M Wireless USB Adapter (802.11b/g)
 - 3) 150M Wireless USB Adapter (802.11N)
- Connect the ASDL router to the local machine using the Ethernet cable
- Make hard-ware reset on the router
- Configure your local LAN adapter on the PC to have a fixed IPv.4 in the same class similar to the default IP on the router.
- Open the internet explorer, and connect to the router based web-page using the default IP and user name/password shown in the back cover of the router
 - 1) i.e. 192.168.1.1
 - 2) username: admin
 - 3) password: admin
- configure the router to the following configuration, which is given by the ADSL internet service provider
 - 1) Connection Type: PPPoE
 - 2) Physical interface: atm0(8/35)

- 3) VPI: 8
 - 4) VCI: 35
 - 5) Encapsulation Mode: LLC
 - 6) PPP username: <Telephone-number>@mada
 - 7) Password: 12345
 - 8) Authentication algorithm: PAP
- If the earlier configurations are correct and the ADSL line is connected with the router then, you have established the connection between the router and the ISP
 - Now we have to apply some configurations on the WLAN
 - 1) Change the AP ID to: LAB1, LAB2, LAB3, Each group will choose a unique ID different from other groups
 - 2) Choose the location (country) to Israel
 - 3) Choose the channel in use: 1-14 or you can choose auto, here chose a fixed channel i.e. 1
 - 4) Now, we need to change the security settings
 - 5) The default: is open (no security)
 - 6) Use the WPA-PSK network authentication
 - 7) Choose appropriate password i.e. 1234567890
 - Now connect the USP wireless adapter to the PC, and connect to the appropriate A.P.
 - ❖ Each time connect one adapter B/G or N
 - Make the wireless adapter configured to DHCP to get the IP automatically
 - Check the IP at the PC and the gateway.
 - Install the program (inSSIDer) available on the O.C.C.



- Now, we will configure the USP wireless adapter to work on different mode b/g or N and to check the link rate

❖ State any difference you can notice using insider in the time graph

- At the end, we will configure all the AP to work on a specific channel, the most overlapped one, and check the difference of network on the insider and show in a table the following

SSID	Channel	RSSI	MAC	Max Rate

- Now, configure the routers to work on Auto channel selections, and check the same parameters as stated in the table

SSID	Channel	RSSI	MAC	Max Rate

- Now, just let one AP. is active (ON)
- All PCS will be connected through the same AP
- One PC is configured to be an FTP server
- Others will be FTP clients
- The FTP server will host a file of a given size i.e. 4GB
- All other clients will download the file
- Let us estimate the throughput at each client
 - ❖ We know the file size, just we need to know the downloading time
- Show the throughput at each PC and the average downloading time
- Now, disconnect all PC from the A.P. except the FTP server and one FTP client
- Calculate the downloading time and the average throughput
- Check the firmware of the A.P.
- Backup the configurations
- Learn how to upgrade the system firmware

Discussion:

- **State the main aspects for the different type of security encryption relate to the A.P. in use**
- **Fill the following table**

Technology	Max Rate	No. non-overlapping channel	Coverage area	No. of channels	Channel B.W.
802.11b					
802.11g					
802.11n					

- **Show the number of Antenna in MIMO technology embedded in the AP 802.11n**

Experiment # 5
GSM Transmitting/Receiving part

Phase A: Transmitting Part

1. Experiment Purposes

- 1) Understand the concept of modulation and measure FM modulation index.
- 2) Understand the concept of resonance circuit through experiment.
- 3) Understand the role of exciter power amplification part and final power amplification part.
- 4) Understand the role of matching circuit

2. Theory

Block diagram of the transmitter for general radio frequency circuit of wireless system can be simply expressed as follows Figure 10-1.

In transmitter circuit, the carrier wave oscillation part generates the carrier wave to carry the signal using crystal or PLL(Phase Locked Loop) system. Modulation part loads the signal to the radio frequency. There are several kinds of modulation such as amplitude modulation, frequency modulation and pulse modulation. This system adopts frequency modulation. Voice signal or data is used as modulation signal. Since deviation of modulation frequency is proportionate to amplitude of signal in FM, control the modulation degree by adjusting the amplitude and signal inputted in modulation(*MOD.IN*) in actual experiment.

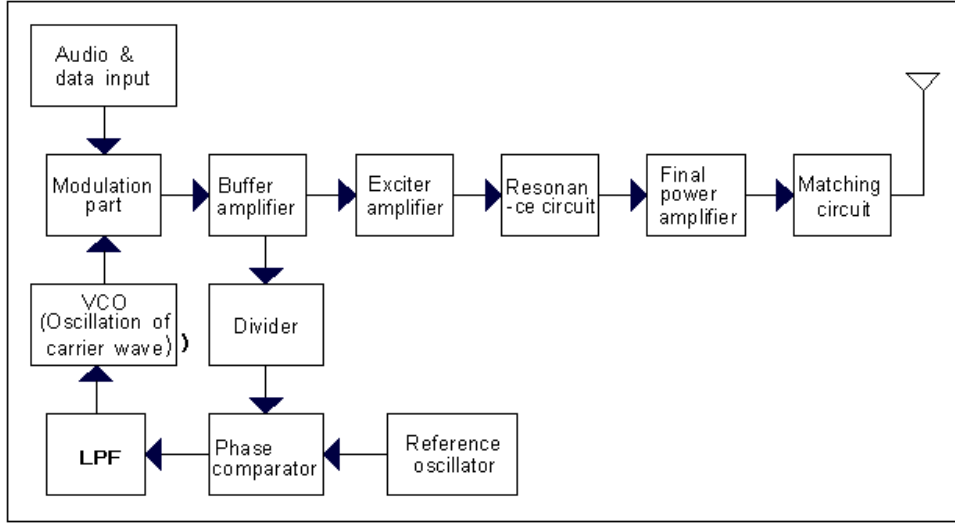


Figure 10-1. Diagram of radio frequency circuit of transmitter

The signal of carrier wave without modulation signal can be expressed as the following formula.

$$v(t) = A \cos \Theta(t) = A \cos \int \omega(t) dt = A \cos(\omega_c t + \Theta_0)$$

Where ω_c is angular frequency of carrier wave and Θ_0 is the initial phase, which is 0 in this case.

The frequency modulation(FM) modulates the frequency of carrier wave by the signal to be sent, $f(t)$. Here, angular frequency $\omega(t)$ which is proportionate to modulation signal($f(t)$), can be defined as follows.

$$\omega(t) = \omega_c + k_f f(t) \quad (\because \omega_c, k_f : \text{Constant})$$

$$\begin{aligned} \Theta(t) &= \int \omega(t) dt = \int (\omega_c + k_f f(t)) dt \\ &= \omega_c t + k_f \int f(t) dt \end{aligned}$$

Where, the phase is expressed as integral of modulation signal $f(t)$. Accordingly, FM wave can be expressed as follows:

$$v_{FM}(t) = A \cos \Theta(t) = A \cos(\omega_c t + k_f \int f(t) dt)$$

Where, if $f(t) = V_m \cos \omega_m t$,

$$\begin{aligned} v_{FM}(t) &= A \cos(\omega_c t + k_f \int V_m \cos \omega_m t dt) \\ &= A \cos(\omega_c t + \frac{k_f V_m}{\omega_m} \sin \omega_m t) \\ &= A \cos(\omega_c t + \frac{\Delta \omega}{\omega_m} \sin \omega_m t) \\ &= A \cos(\omega_c t + m_f \sin \omega_m t) \end{aligned}$$

Where, let's set $\Delta \omega$ as the maximum frequency deviation ($\Delta \omega = k_f V_m$) from the carrier wave.

$$\omega(t) = \omega_c + k_f f(t) = \omega_c + k_f V_m \cos \omega_m t = \omega_c + \Delta \omega \cos \omega_m t$$

$$m_f = \frac{\Delta \omega}{\omega_m} = \frac{k_f V_m}{\omega_m} \text{ is modulation index.}$$

Modulation is performed inside IC. As oscillation part and exciter amplifier at the back are separated by **Buffer amplifier**, the signal goes through external resonance circuit and is entered to exciter amplifier. **Exciter amplifier** amplifies only necessary power and inputs the signal to the final power amplifier after resonance circuit. **Final power amplification part** amplifies the power to the level required to radiate the modulated wave to the far location. Amplified final output is radiated to the air through antenna. **Matching circuit** should be

added for good impedance matching between antenna and output part of final amplifier to effectively radiate the wave to the air.

Figure 10-2 is a radio frequency power amplifier circuit using transistor, and figure 10-3 shows the result of simulating the circuit in the Figure 10-2. It is found out that the output level was amplified about 6 times(16dB) compared to input level.

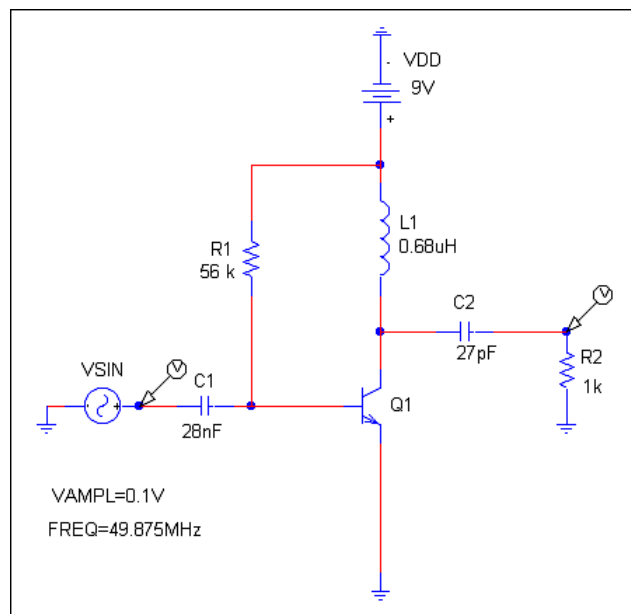


Figure 10-2. Power amplifier circuit

3. Preparations for the Experiment

- 1) DC power supplier or (POWER board +9[V], 300[mA] adapter)
- 2) Digital multi-meter
- 3) Oscilloscope (100MHz)
- 4) Frequency counter
- 5) CONTROL board
- 6) Radio frequency(RF) board

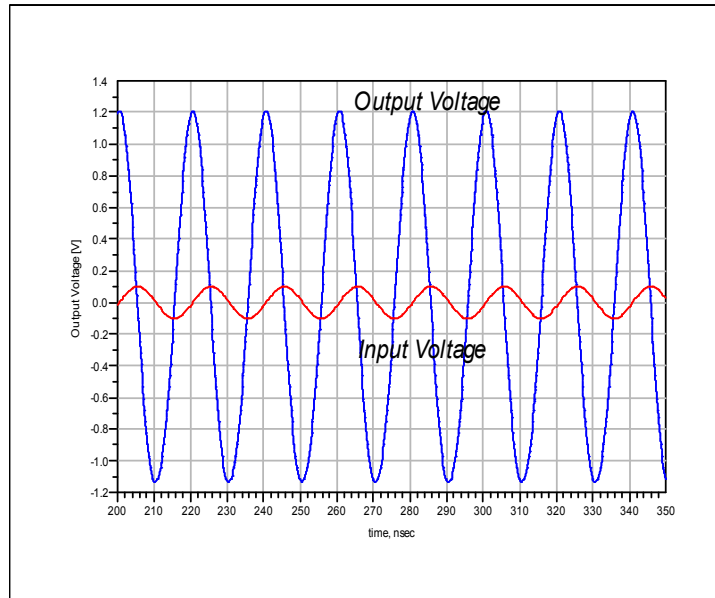


Figure 10-3. Power amplifier simulation result

4. Procedure of Experiment

- 1) Connect the RF board, the CONTROL board and the POWER board as shown in the Figure 10-4.
- 2) Power is provided to each board from POWER board through connector. However, if you experiment after separating the board from the bag, supply the power to GND and VDD using DC power supplier.

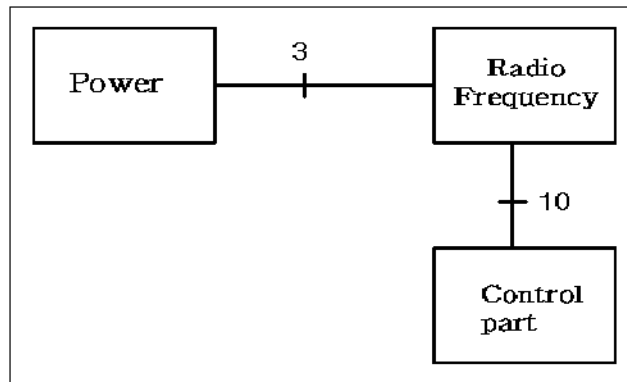


Figure 10-4. Connection Diagram of Transmitting Experiment

- 3) Input the sine wave of 1[kHz], 200[mV_{p-p}] using low frequency signal generator. This signal is modulation signal. To check the frequency and level of the signal, measure the waveform at *TX.AUD* terminal with digital multi-meter and oscilloscope and record the value.

Frequency _____ [kHz] Voltage level _____ [mV_{p-p}]

- 4) Since FM modulation index is proportionate to amplitude of modulation signal, it is possible to control the modulation index of signal by adjusting VR2. Adjust the VR2 so that the amplitude of 1[kHz], 200[mV_{p-p}] measured in *MOD.IN* is 30[mV_{p-p}] in case of mobile station, and is 20[mV_{p-p}] in case of base station. Record this value.

MOD.IN _____ [mV_{p-p}]

- 5) Separate the low frequency signal generator from *TX.AUD* terminal.
 ※ To observe the waveform of transmitting data, set the data transfer mode in the order shown below:

☞ Press the TX-DATA button in control keypad.

- The number in Text LCD of control part increase one by one.
- The data corresponding to the number is made at microcontroller and is entered to modulation circuit.
- *TX.DATA* LED blinks.

- 6) To check the level and frequency of data to be sent, measure the waveform of *TX.DATA* with oscilloscope at DC mode and record the frequency and voltage level. Since two different kinds of frequencies are combined, record the lower frequency.

Frequency _____ [kHz]

Voltage level _____ [V]

- 7) To measure the data level inputted to modulation part, measure and record the voltage level of data at DC mode using oscilloscope at *MOD.IN*.

Data voltage level _____ [mV_{p-p}]

- 8) Press TX-DATA button in control keypad. The data transmission is stopped, channel is displayed in Text LCD.(since it is set as channel 1 at the factory, '01' is displayed)
- 9) To check the reference oscillation frequency, measure the oscillation frequency and output level of crystal oscillation frequency, *REF.OSC* and measure the value. (It is correct if it is 10.240[MHz]. Adjust the CV2 only when the error of oscillation frequency is big. Be careful not to frequently adjust CV2 since it may result in damage.)

※ If oscillation output are not displayed in *REF.OSC*, measure it in No. 3 pin of DEMOD IC.

Frequency _____ [MHz]

Voltage level _____ [mV_{p-p}]

10) Set to channel 5 using the keypad in CONTROL board.

☞ Set the mode selection switch to MENU.

- Press the keypad of control board, *CHAN.* $\Rightarrow 0 \Rightarrow 5 \Rightarrow$ *CHAN.*
[Press the number in the blinking digit]
- 05 will be displayed in Text LCD and, CH D2 and CH D0 will be ON.

11) Input the sine wave of 1[kHz], 200[mV_{p-p}] at *TX.AUD* again using low frequency signal generator.

12) Measure and record the output frequency and voltage level of VCO at *TX.VCO* using frequency counter and oscilloscope. At this time, base station(BS) should have 46.730[MHz] and mobile station(MS) should have 49.875[MHz] for accurate display. If there is a difference from these values, these differences are the errors. Measure with probe at $\times 10$.

Frequency_____ [MHz] Voltage level_____ [mV_{p-p}]

13) Measure and record the output frequency and voltage level of modulated wave amplified at *BUF.AMP* with probe at $\times 10$, using the frequency meter and oscilloscope.

Frequency_____ [MHz] Voltage level_____ [mV_{p-p}]

14) Measure the output frequency and voltage level of modulated wave amplified at *EXC.OUT* with probe at $\times 10$, using the frequency meter and oscilloscope. Check the voltage level and record the result value(can be adjusted at VR4).

Frequency_____ [MHz] Voltage level_____ [mV_{p-p}]

- 15) Measure the final power amplifier voltage level of modulated wave at *RF.OUT* with probe at $\times 10$, using the oscilloscope. Check whether the voltage level is about $2,500[mV_{p-p}]$ for base station, and $750[mV_{p-p}]$ for mobile station, and record the result value(can be adjusted at VR1).

Voltage level_____ $[mV_{p-p}]$

- 16) Measure the radiation power voltage level of modulated wave at *RAD.PWR* using oscilloscope and frequency counter with probe at $\times 10$ and record the result value.

Frequency_____ $[MHz]$

Voltage level_____ $[mV_{p-p}]$

Experiment Report

Experiment title	Experiment 10 : Transmitting Part	Date of Experiment	D/M/Y	
Student's number and name	Student's number : [] Student's name : []	Year and class Experiment team	Year Class Experiment team : []	
Name of co-experimenter			Score	

-
- 3) Frequency _____ [kHz] Voltage level _____ [mV_{p-p}]
- 4) MOD. IN _____ [mV_{p-p}]
- 6) Frequency _____ [MHz] Voltage level _____ [mV_{p-p}]
- 7) Data voltage level _____ [mV_{p-p}]
- 9) Frequency _____ [MHz] Voltage level _____ [mV_{p-p}]
- 12) Frequency _____ [MHz] Voltage level _____ [mV_{p-p}]
- 13) Frequency _____ [MHz] Voltage level _____ [mV_{p-p}]
- 14) Frequency _____ [MHz] Voltage level _____ [mV_{p-p}]
- 15) Voltage level _____ [mV_{p-p}]
- 16) Frequency _____ [MHz] Voltage level _____ [mV_{p-p}]

Phase B: Receiving Part

1. Experiment Purposes

- 1) Understand the concept of FM demodulation
- 2) Understand the concept of super-heterodyne detection
- 3) Understand the role of receiving filter and RF amplifier
- 4) Experiment on the detection of data and voice signal

2. Theory

Configuration of receiver in radio frequency circuit of general wireless system can be simply expressed as shown in the Figure 11-1.

In receiver circuit, antenna should be designed to induce as much signals as possible though very weak signal is received. Impedance matching is required between receiving part so that this induced signal can smoothly pass to the receiving circuit. Matching circuit plays this role of impedance matching. Band pass filter corresponds to duplex filter that separates the transmitting and receiving.

The RF amplifier amplifies the weak signal that is received at the receiving antenna. In addition, noise figure of RF amplifier may have big impact on overall system, RF amplifier uses FET or transistor with small noise figure, even if they are expensive. Figure 11-2 is the RF amplifier circuit that uses FET. Frequency of 5 channel of mobile station, 49.875[MHz], was used as an input signal. DC voltage of 4[V] was supplied.

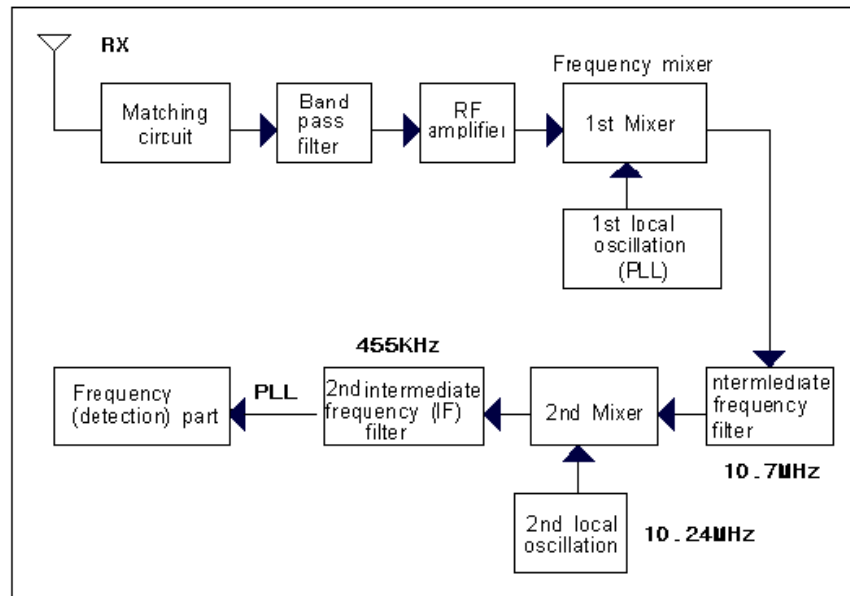


Figure 11-1. Diagram of radio frequency circuit of receiver

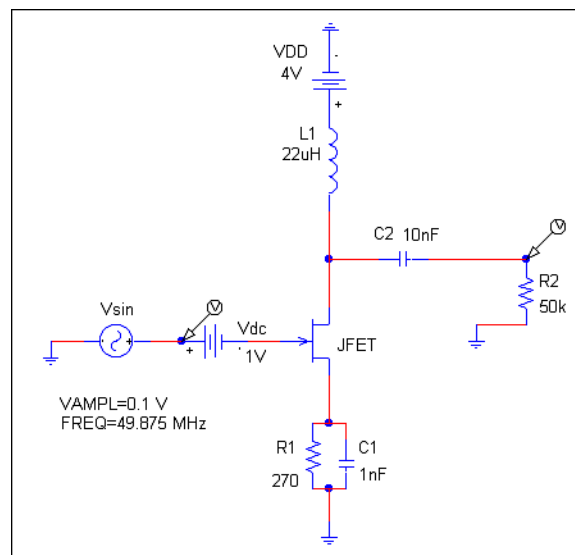


Figure 11-2. Very High Frequency Amplifier Circuit

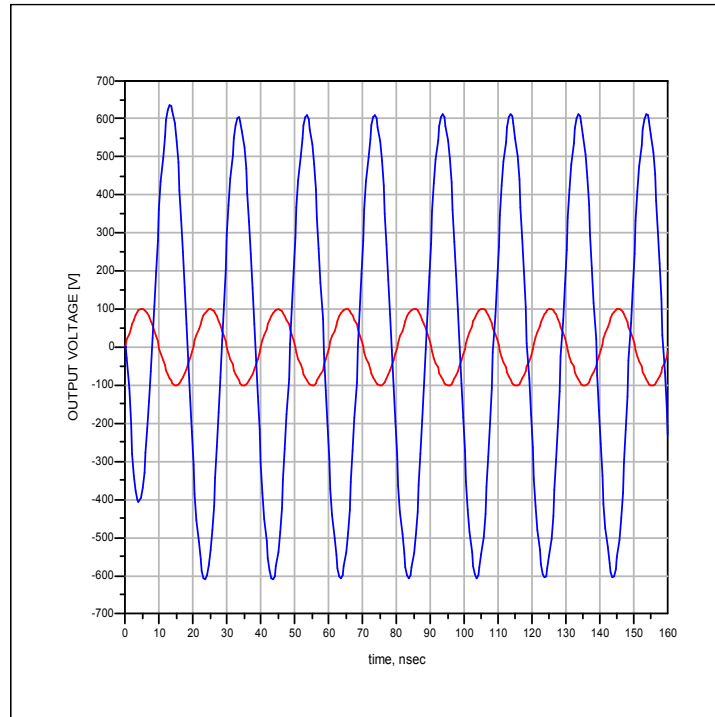


Figure 11-3. Simulation result of very high frequency amplifier

The first mixer(frequency converter) is a circuit that outputs the 1st intermediate frequency by mixing modulated wave with local oscillation frequency. The PLL oscillator was used for the local oscillation. This system used the 1st IF(intermediate frequency) of about 10.7[MHz]. The output is inputted to the 2nd mixer after IF(intermediate frequency) filter. The 2nd local oscillator oscillates 10.24[MHz] and outputs the IF of 455[kHz]. After this frequency passes through the filter, the signal is detected.

The circuit used for detection in this system is a quadrature detector that is widely used in TV or FM radio demodulation circuit.

By multiplying the signal whose phase is changed 90 degree with the input FM signal, we can obtain the signals of 2 frequency elements, which correspond to the sum and difference of two signals. In these two signals,

discard the higher one and select the audio frequency using LPF. This signal contains the information signal we want. Figure 11-4 is the block diagram of quadrature detector.

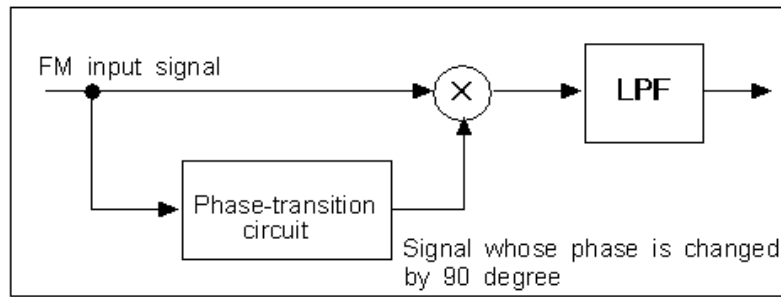


Figure 11-4. Quadrature Detector

Based on the information signal of $f(t) = V_m \cos \omega_m t$, following is the expression for the situation when the modulated FM signal is entered as an input of detector

$$v_{FM}(t) = A \cos \Theta(t) = A \cos \left(\omega_c t + \frac{\Delta \omega}{\omega_m} \sin \omega_m t \right) = A \cos (\omega_c t + \phi(t)) \quad (11-1)$$

$$\text{Where } \phi(t) = \frac{\Delta \omega}{\omega_m} \sin \omega_m t.$$

When passing through 90 degree phase delay FM input signal can be expressed as follows due to group delay.

$$v_{90}(t) = A \cos \left(\omega_c t - \frac{2}{\pi} + \phi(t - t_1) \right) = A \sin (\omega_c t + \phi(t - t_1)) \quad (11-2)$$

By multiplying two signals of $v_{FM}(t)$ and $v_{90}(t)$,

$$\begin{aligned}
v_D(t) &= A^2 \cos(\omega_c t + \phi(t)) \sin(\omega_c t + \phi(t - t_1)) \\
&= -\frac{A^2}{2} [\sin(\phi(t) - \phi(t - t_1)) - \sin(2\omega_c t + \phi(t) + \phi(t - t_1))]
\end{aligned} \tag{11-3}$$

When this signal is passed to LPF, the element of higher frequency is removed and the first element, which is lower frequency, remains. Assuming that the time delay is small, it can be expressed as follows:

$$v_D(t) = -\frac{A^2}{2} \sin(\phi(t) - \phi(t - t_1)) \approx -\frac{A^2}{2} (\phi(t) - \phi(t - t_1)) \tag{11-4}$$

In this differential equation,

$$\frac{dv}{dt} = \frac{v(t) - v(t - t_1)}{t_1} \tag{11-5}$$

Using

$$\frac{d\phi(t)}{dt} = \frac{\phi(t) - \phi(t - t_1)}{t_1} \tag{11-6}$$

then,

$$\phi(t) - \phi(t - t_1) = t_1 \frac{d\phi(t)}{dt} = t_1 \frac{d}{dt} \left[\frac{\Delta\omega}{\omega_m} \sin \omega_m t \right] = t_1 \Delta\omega \cos \omega_m t \tag{11-7}$$

Therefore, the final detection output can be expressed as in the formula 11-8. As you may see, the original information signal $f(t) = V_m \cos \omega_m t$ is detected, from the detection output waveform.

$$v_D(t) = -\frac{A^2}{2} (\phi(t) - \phi(t - t_1)) = -\frac{A^2}{2} t_1 \Delta\omega \cos \omega_m t = K_D \cos \omega_m t \tag{11-8}$$

Where $K_D = -\frac{A^2}{2} t_1 \Delta\omega$ is a detection constant.

Detected signals are audio and video which are received via different routes. Audio is sent to and processed at speaker circuit and the data is sent to and processed at microcomputer. Since *RX.DATA* is to check the modulation status of the received data, it shapes and outputs the waveform as square wave.

3. Preparations for the Experiment

- 1) DC power supplier or (POWER board +9[V], 300[mA] adapter)
- 2) Digital multi-meter
- 3) Oscilloscope (100MHz)
- 4) Frequency counter
- 5) Radio frequency generator
- 6) CONTROL board
- 7) Radio frequency(RF) board

4. Procedure of Experiment

- 1) Connect the RF board, the CONTROL board and the POWER board as shown in the figure.

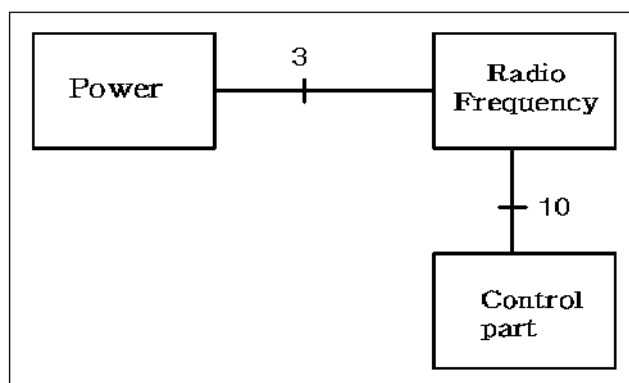


Figure 11-5. Board Connection Diagram

- 2) Power is provided to each board from POWER board through connector. However, if you experiment after separating the board from the bag, supply the power to GND and VDD using DC power supplier.
- 3) Set the frequency of RF(radio frequency) signal generator to the channel 5. It is 49.875[MHz] for base station and 46.730[MHz] for mobile station. Set the 1[kHz] sine wave to FM frequency deviation 3[kHz] and record the value.
 - ※ If you don't have RF signal generator, set the mobile station and base station to the same channel and communicate wirelessly. That is, the mobile station acts a role of signal generator to the base station.

Signal wave frequency_____ [kHz] Frequency deviation_____ [kHz]

- 4) Connect the RF signal generator to the test point connector of RF board using the connector.
- 5) Set the output of RF signal generator to -10[dBm] and enter this to *RF.IN* terminal. Measure and record the receiving input voltage of FM modulated wave at *LNA.IN* with oscilloscope in alternating mode and probe in $\times 10$. At this time, turn off the transmitting power of equipment by pressing TX.PWR

button for clear measurement.

- ※ If the maximum output level of RF signal generator does not reach $-10[\text{dB}]$, set it to $-20[\text{dB}]$.

Voltage_____ $[\text{mV}_{\text{p-p}}]$

- 6) Measure the output waveform voltage of RF amplifier of FM demodulation wave at *LNA.OUT* using oscilloscope. Record the values.

Voltage_____ $[\text{mV}_{\text{p-p}}]$

- 7) Measure the frequency and amplitude of *RX.VCO* using the frequency counter and oscilloscope in order to check the output of the 1st local oscillator. Record the values.

Frequency_____ $[\text{MHz}]$

Voltage _____ $[\text{mV}_{\text{p-p}}]$

- 8) Measure the frequency and amplitude of *1ST.IF* using the frequency counter and oscilloscope in order to check the output of $10.695[\text{MHz}]$, the 1st intermediate frequency. Record the values. At this time, set the ratio of probe as $\times 1$.

Frequency_____ $[\text{MHz}]$

Voltage _____ $[\text{mV}_{\text{p-p}}]$

- 9) Measure the frequency and amplitude of the *REF.OSC* using the frequency counter and oscilloscope in order to check the output of the 2nd local oscillator of the 2nd frequency converter. Record the values. At this time, measure the frequency using the probe of oscilloscope and set the ratio of probe as $\times 10$.

Frequency_____ $[\text{MHz}]$

Voltage _____ $[\text{mV}_{\text{p-p}}]$

- 10) Measure the frequency and amplitude of *2ND.IF* using the frequency counter

and oscilloscope in order to check the output of 455[kHz], the 2nd intermediate frequency. Record the values.

Frequency_____ [kHz] Voltage _____ [mV_{p-p}]

- 11) Measure the frequency and amplitude of *RX.AUD* using the frequency counter and oscilloscope in order to check the output of demodulated audio frequency signal. Record the values. Adjust the IFT coil to obtain the optimal waveform. If it is normally demodulated, the demodulated output will be been as clear signal of 1[kHz], which was used as signal wave for modulation at the RF signal generator.

Frequency_____ [kHz] Voltage _____ [mV_{p-p}]

- 12) Measure the frequency and amplitude of *RX.DATA* using the frequency counter and oscilloscope. Record the values. If it is normally demodulated, the demodulated output will be displayed as a square wave that shaped from the sine wave of 1[kHz], which was used as the signal for modulation at the RF signal generator.

Frequency_____ [kHz] Voltage _____ [mV_{p-p}]

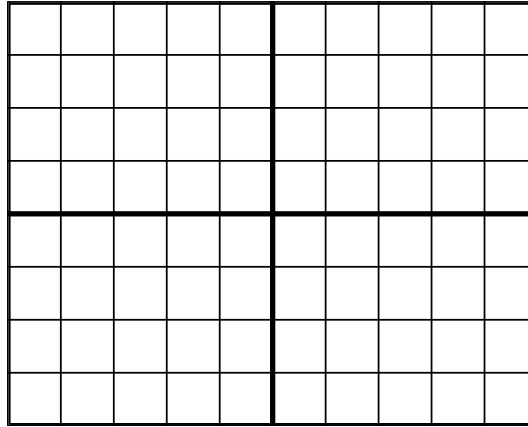


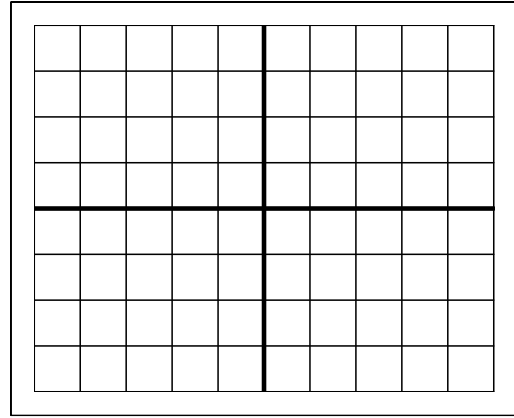
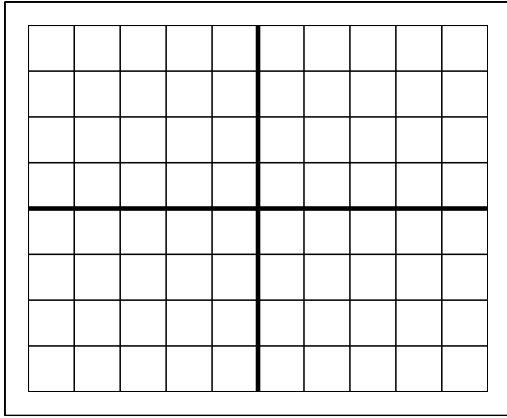
Figure 11-6. Waveform of *RX.DATA*

- 13) Measure the receiving signal at *RX.AUD* and record the frequency and level while decreasing the output of RF signal generator from -50[dBm] to -90[dBm] by 10[dBm] step for each. At this time, change the channel or turn off either of mobile station or base station, if the receiving signal is not properly displayed due to the frequency interference of same channels nearby.

Frequency _____ [kHz]

Voltage _____ [mV_{p-p}]

- 14) First of all, record the waveform of *RSSI* and amplitude value after setting the output of RF signal generator to -50[dBm]. At this time, set the oscilloscope to DC mode, Volt/Div to 0.2[V], and probe to 10. Next, turn off the output of RF signal generator, and record the waveform of *RSSI* and amplitude value.
- ※ If there is an experiment team who performs the same channel test nearby, the wave can be received through antenna eve if the receiving power is lowered. Therefore, do not use the same channel with other experiment team.



a) Output of radio frequency
signal generator -50[dBm]

b) Output of radio frequency
signal generator OFF

Figure 11-7. Output of received signal strength indicator(RSSI)

Experiment Report

Experiment title	Experiment 11 : Receiving Part	Date of Experiment	D/M/Y
Student's number and name	Student's number : [] Student's name : []	Year and class Experiment team	Year Class Experiment team : []
Name of co-experimenter		Score	

3) Signal wave frequency _____ [kHz] Frequency deviation _____ [kHz]

5) Voltage _____ [mV_{p-p}]

6) Voltage _____ [mV_{p-p}]

7) Frequency _____ [MHz] Voltage _____ [mV_{p-p}]

8) Frequency _____ [MHz] Voltage _____ [mV_{p-p}]

9) Frequency _____ [MHz] Voltage _____ [mV_{p-p}]

10) Frequency _____ [kHz] Voltage _____ [mV_{p-p}]

11) Frequency _____ [kHz] Voltage _____ [mV_{p-p}]

12) Frequency _____ [kHz] Voltage _____ [mV_{p-p}]

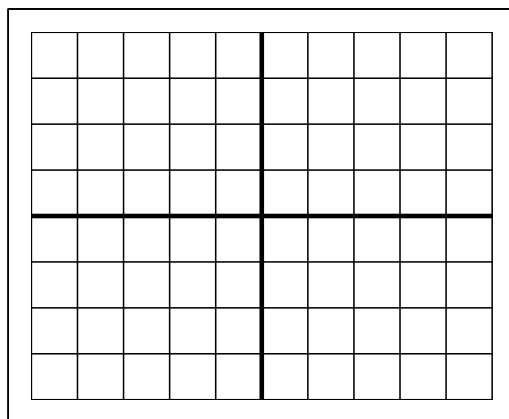
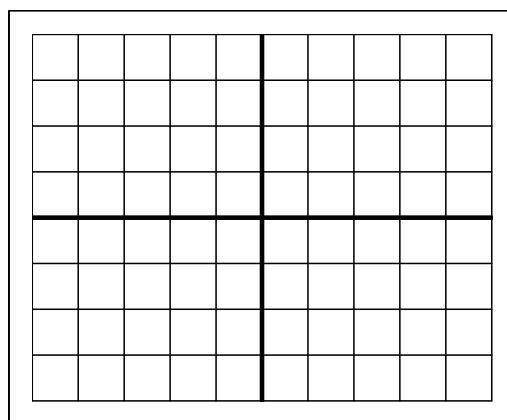
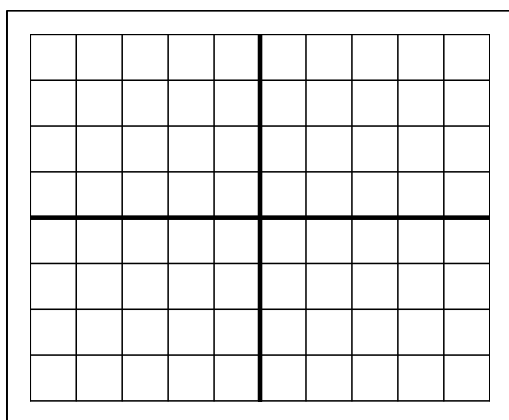


Figure 11-6. Waveform of *RX.DATA*

13) Frequency_____ [kHz]

Voltage _____ [mV_{p-p}]

14)



a) Output of radio frequency
signal generator -50[dBm]

b) Output of radio frequency
signal generator OFF

Figure 11-7. Output of received signal strength indicator(RSSI)

Experiment # 6
PHASE A: Phases of telephony connection

Objectives

The aim of this exercise consists in following the phases of the connection between two telephone sets and of its release. The table 10.1 indicates these phases and their respective signalling.

PHASE	TYPE of SIGNAL	DIRECTION of the SIGNAL	
		CALLING USER	CALLED USER
Engaging the user (off-hook)	Line closed onto user terminal	User->Exchange	
Inviting to dialling	"DIAL" tone	User->Exchange	
Dialling	Pulse or "DTMF"	User->Exchange	
Busy line	Busy tone	User->Exchange	
Disengaged line	Free tone	User->Exchange	
Ringing	Ringing current		User->Exchange
Engaging the answer (off-hook)	Line closed onto user terminal		User->Exchange
ESTABLISHING the CONNECTION			
Disengagement (hanging up)	Opening the user loop	User->Exchange	
RELEASING the CONNECTION			

Table 6.1

In detail, this exercise will:

- analyze the activity carried out by the exchange when a line is engaged:
 - closing the user loop determines the transition of the engagement signal (SWITCH HOOK DETECTOR) of user interface to the active state
 - the exchange will answer with the DIAL tone
- detect the pulse sequences of line user opening due to the pulse dialling:
 - a transition of the engagement signal output by the user interface, to the rest condition, will correspond to each opening pulse
 - the control unit will count these transitions to trace the sent digits, assess the duration of interdigit pauses (in closing phase) besides acquiring the dialling number used to establish the required connection
- observe the behaviour of multifrequency signalling where the dialling digits are coded by pairs of acoustic tones decoded by the user interface:

- the decoding circuit outputs the value of the digits sent as binary codes of 4 bits plus a line of "valid data item"; the binary codes are displayed by LEDs (ON = State "1") "Q4-Q3-Q2-Q1" in DTMF CODE
- verify the establishment and release of a connection.

Necessary equipment

- Unit PCM/EV with telephone sets
- Oscilloscope • Multimeter

Operations

- Switch the system on keeping the handsets cradled • Connect the multimeter (50 Vfs) between TP13 (Test Point 13) and TP12: the line voltage #1 is equal to approximately 40 Vdc
- Take off the telephone #1 and observe that the line voltage is reduced to 10 V because the user loop is closed on the dc impedance of the telephone: this state is signalled by the "DIAL" LED in LINE SIGNALING
- Connect the oscilloscope (in alternating voltage) between TP13 and earth: adjusting the scales properly will emphasize the DIAL tone; this tone is also signalled by a LED
- The same tone can be observed between TP15 and earth, as voice input of the user interface (RECEIVE INPUT)
- Beings the input of the oscilloscope in dc voltage (1V/Div) connect the probe with TP17, corresponding to the engagement signal (SW HOOK DETECTOR) of the user interface: the active logic state is "0", signalled by the LED of "USER1" in SWITCH HOOK DETECTOR
- Dial a digit on the telephone#1 and realize that the DIAL tone disappears. The state transitions due to the pulse dialling (line opening pulses) will be displayed on the oscilloscope. The dialling pulses are also signalled by the LED of "USER1" in SWITCH HOOK DETECTOR
- dial other digits adjusting the time scale of the oscilloscope to assess the durations of pulses and of pauses
- the telephone can receive the busy tone or the free tone, followed by the ringing signal in one of the remaining telephones, according to the dialled digits
- the return of the engagement line (SW HOOK DETECTOR, TP17) to the rest condition (high level)
- connect the oscilloscope (in alternating voltage, 0.5 V/Div, probe 10:1) with TP16 and take off the telephone #1 (multifrequency) observing that the LED connected with the engagement line (SW HOOK DETECTOR) of the user interface, will go on

- after hearing the DIAL tone dial a digit: note that this tone disappears and that a signal is displayed on the oscilloscope (the duration of this signal depends on the time the key is kept pressed)
- dial other digits and adjust the time scale and the trigger of the oscilloscope to optimize the observation of the DTMF tones arriving at the decoding circuit. These tones can also be heard on the receiver. The signal available in TP16 is the result of the sum of two (high and low) tones that code the dialled digit
- observing the LEDs “Q4-Q3-Q2-Q1” in DTMF CODE connected with the decoder lines checks whether the dialled digit corresponds to the codes output by the circuit:

Q1	Q2	Q3	Q4	DIGIT:
ON	OFF	OFF	OFF	1
OFF	ON	OFF	OFF	2
ON	ON	OFF	OFF	3
OFF	OFF	ON	OFF	4
ON	OFF	ON	OFF	5
OFF	ON	ON	OFF	6
ON	ON	ON	OFF	7
OFF	OFF	OFF	ON	8
ON	OFF	OFF	ON	9
OFF	ON	OFF	ON	0

The telephone can receive the busy tone or the free tone, followed by the ringing signal in one of the remaining telephones, according to the dialled digits.

- hang up the receiver observing that the engagement LEDs go out
- connect a probe of the oscilloscope (50V/Div) with TP12 (RING) and the other probe with TP14 (RELAY DRIVER)
- after hearing the DIAL tone dial the number of the line #1 (01) on a telephone set different from #1. As the ringing signal arrives at the telephone #1, this state is signalled by the LED of “USER1” in INCOMING CALLING, whereas the ringing signal of approximately 150 Vpp (50 Hz) is displayed in TP12 and the level switches to “0” in TP14
- note the correspondence between the state of the calling line, the enabling of the relay followed by the going on of the ringing LED, the current crossing the line and the free tone sent to the calling telephone: all these signalling have the same timing (a second of activity every five seconds)
- when the telephone #1 is taken off the engagement LED goes on and the ringing current is immediately interrupted by the user interface. Then the control unit removes the ringing command and the free tone from the calling telephone (these events can be seen only if the telephone is taken off during the active phase).

At this point the connection between the two telephones is established

- verify the voice connection with the calling telephone. Hang up and take off the telephone #1 observing that the speech is kept. Hang up the calling telephone noting that the connection is released. Hang up the telephones
- connect the oscilloscope (in ac voltage; 1ms/Div; 0.5V/Div; probe 10:1) with TP15 (RECEIVE INPUT). Take off the telephone #1 and observe that the LED signalling the DIAL tone goes on. Examine the characteristics of this tone adjusting the scales of the oscilloscope properly
- take off one of the remaining telephone set (the telephone #2, for instance) and dial its number (02). The “DIAL” LED will go on in LINE SIGNALING: this LED indicates that the busy tone can be heard on the telephone #1. Examine the characteristics of this tone adjusting the scales of the oscilloscope properly
- hang up both telephones and then repeat the dialling from the telephone #1. Wait for the ringing signal on the selected telephone and for the going on of the LED signalling the fre tone on the telephone #1. Examine the characteristics of this tone adjusting the scales of the oscilloscope properly
- take off the called telephone. Verify the presence of the voice connection. Hang up the telephones.

PHASE B: CONNECTING TWO SYSTEMS PCM/EV

Objectives

The aim of this exercise consists in showing how two systems PCM/EV connected with each other can simulate two telephone exchanges connected via Trunking.

Necessary equipment

- Two Units PCM/EV with telephone sets (PCM1 and PCM2)

Operations

- Connect its own telephone sets with each unit PCM/EV • Connect the two units with each other using the coaxial cables of the equipment and the IN/OUT jacks available on the fore panel.

Connect:



Set the control devices for both the units as indicated here below:

- Potentiometer **NOISE** at minimum (0)
- Potentiometer **ATTENUATION** at minimum (0)
- Switch of **CONTROL SELECTION** to the **REMOTE** position
- Power the units PCM/EV
- Press the button **MODE SELECT** to enable the **REMOTE MODE**: the LED of the selected mode will go on to confirm the selection carried out
- Switch of **REMOTE MODE** to the position



Verify the possibility of carrying out some remote connections between users. For instance:

- Pick up the telephone **USER1** connected with the system **PCM1** and call the remote telephone **USER1** connected with the system **PCM2** by dialling the number 51
- Make sure that a correct communication is established between these two telephones
- Observe that no noise nor attenuation can be inserted because the **LINE SIMULATOR** is not used in this configuration,

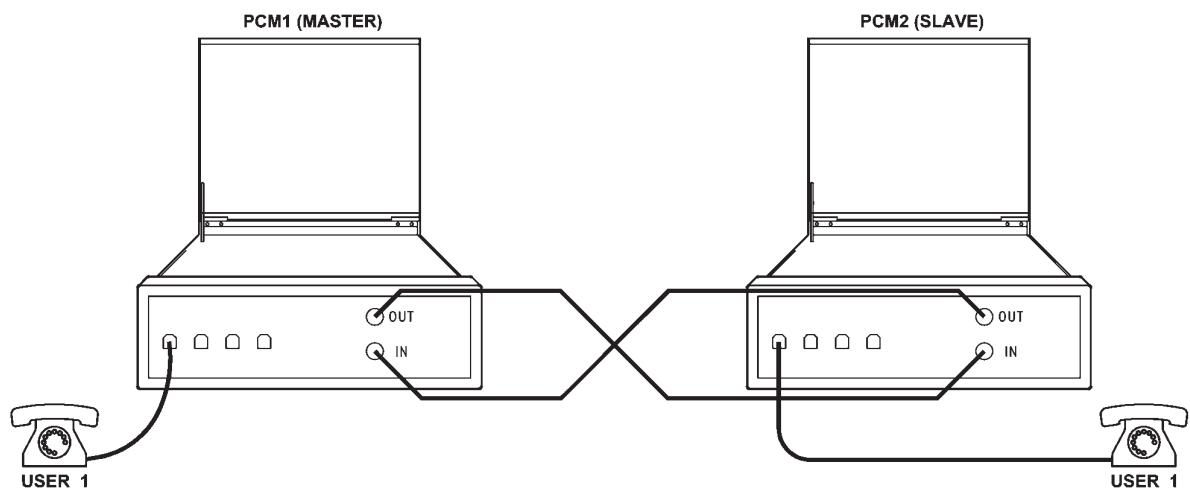


Fig. 6.1

Experiment # 7

Bluetooth Discovery/Discoverability

7.1 Theory: Finding other Bluetooth devices

To be able a Bluetooth device to communicate using Bluetooth there needs to be another device to communicate to. With Bluetooth this can be any device within communications range. The first step to establishing communications is to find that other device. The BLU2i module has an inquiry command, AT+BTI, that can be used to inquire about other devices in range. The AT+BTI command sends a signal that is picked up by any Bluetooth device within range. These Bluetooth devices can then elect to respond to this inquiry to let the inquiring device know that they are out there. Devices respond to an inquiry by sending their device address that can then be used later on to contact the receiving Bluetooth system.

7.1.1 The Inquiry command

The Inquiry command is `AT+BTI<devclass> {Inquire}`.

Full details of the command can be found in section 2.2.34 of the BLU2i AT Command set document. This is included on the EB617 CD ROM. Whenever a new AT Command is introduced refer to the AT Commands set document to gain an understanding of the new command. Building up knowledge of the BLU2i AT Commands set is an important body of knowledge. The AT+BTI command causes the Bluetooth module to transmit an Inquiry signal to which other Bluetooth devices can respond. The optional <devclass> parameter is used to filter which devices to check for and will be dealt with later. For now the base AT+BTI command can be used. The AT+BTI command uses two S Registers to set the delay and maximum number of responses. These can be left at their default values for the exercises and examples in this course.

The AT+BTI command can receive the following responses:

- 123456789012 – The 12 digit Hexadecimal address of the Bluetooth device answering the inquiry. Note that not all devices within range may answer. Bluetooth devices can be set to ignore the Inquiry command. This will be covered next chapter.
- OK – Received when the Inquiry is complete.
- ERROR 14 – An error has occurred. The Bluetooth module is not configured correctly for sending the Inquiry command. (See the AT Command Set document for details).

A number of responses can be received: one address response from each device answering the inquiry, and an OK message once all have been received. Checking can be done and if it is not an OK response then the address can be displayed.

7.1.2 Additional Inquiry commands and the <devclass> parameter

In addition to AT+BTI there are a number of other related commands such as AT+BTIV and AT+BTIN.

AT+BTIV <devclass> functions the same as AT+BTI except that the 6 digit <devclass> of the responding device is added to the response message.

For example: 123456789012,123456

AT+BTIN <devclass> functions the same as AT+BTI except that the 6 digit <devclass> of the responding device, and the name of the device is added to the response message.

For example: 123456789012,123456,"EZURIO blu2i RS232"

The Inquiry commands take an option <devclass> parameter that can be used to filter responses. The <devclass> parameter is a 2 or 6 digit hexadecimal character value. To check for device of a specific type use the full 6 character <devclass> for that device type. For

instance to check for all headsets within range you would use the <devclass> value 200404 i.e. AT+BTI200404. The 2 character major class code can be used to check for types of devices such as Audio devices or PC peripherals. For example 20 will search for Audio devices such as headsets phones and music players.

7.1.3 Discovery example

There are 4 Bluetooth devices within range, named Blue1-Blue4. The devices are set up as follows:

Device	Address	Configuration
Blue1	00809872F3D4	Accepts Inquiry commands
Blue2	00809864DD44	Accepts Inquiry commands
Blue3	00809894E620	Does not accept Inquiry commands
Blue4	00809894E5D5	Accepts Inquiry commands

Device Blue1 transmits the AT+BTI inquiry command.

The following responses are received:

- 00809864DD44
- 00809894E5D5
- OK

7.2 Exercise 1: Discovering Bluetooth devices

7.2.1 Introduction

To be able to communicate to a Bluetooth device it is necessary to know that the device is there. The first step in Bluetooth communications is therefore to inquire as to what devices are present, and to be able to identify them so that communications can be established with them.

7.2.2 Objectives

- Develop a program that performs a Bluetooth device Inquiry and displays the addresses of any devices found on the LCD display.
- Display a 'Finished' message once all the responses have been received.

7.2.3 Pre-requisites

- An understanding of standard Flowcode Components and icons, such as Decision icons and the LCD component and macros.
- An understanding of sending Bluetooth commands (See the Bluetooth Component Help file in Flowcode for details on the macros involved)
- An understanding of receiving and retrieving message data (See the Bluetooth Component Help file for details on the macros involved, and an outline of the relevant strategies)

7.2.4 Hardware/Software requirements

The following items of hardware are required:

- Bluetooth solution for the Exercise program.
- 1 or more additional Bluetooth devices for demonstrating the program.
Note: The second Bluetooth board from the Bluetooth solution can be used for this Exercise.
The program BT_EX1_NODE_A.FCF FCF can be used to set up the board.
- Set hardware jumpers as specified in the Getting starting section.
- Set up PPP to configure the microcontroller as specified in the Getting starting section.

7.2.5 Exercise information

The Inquiry command is AT_BTI. See the EZURiO AT Command set document (supplied on CD ROM) for details on the AT+BTI command.

The expected responses are:

- A 12 digit device hexadecimal address from devices that respond.
- An ERROR message.
- OK once the Inquiry is complete.

The objectives can be broken down into the following:

Tasks

- Send the Inquiry command.
- Check for responses.
- Display the device address of any device that responds.
- Check for OK response.
- Display 'Finished' once the inquiry is complete.

Control structures:

- If the Inquiry is complete display a 'Finished' message.
Otherwise continue checking for further responses.

7.2.6 Learning outcome

Primary learning outcomes for this exercise are:

- Understanding basic AT commands
- Understanding the discovery process
- Creating and sending commands
- Checking for and retrieving responses
- Checking for specific responses (e.g. OK)

7.2.7 Additional tasks

The following are additional tasks that can be implemented to add extra features or improved functionality to the basic program:

- Add a COUNT for how many addresses were found.
- Perform basic error checking on the CreateCommand and SendCommand macro.
- Check for an ERROR 14 response.
- Retrieve and display the device name(s) using AT+BTIM.

7.3 Practical implementation: Discovering Bluetooth devices

7.3.1 Discovering and discoverable

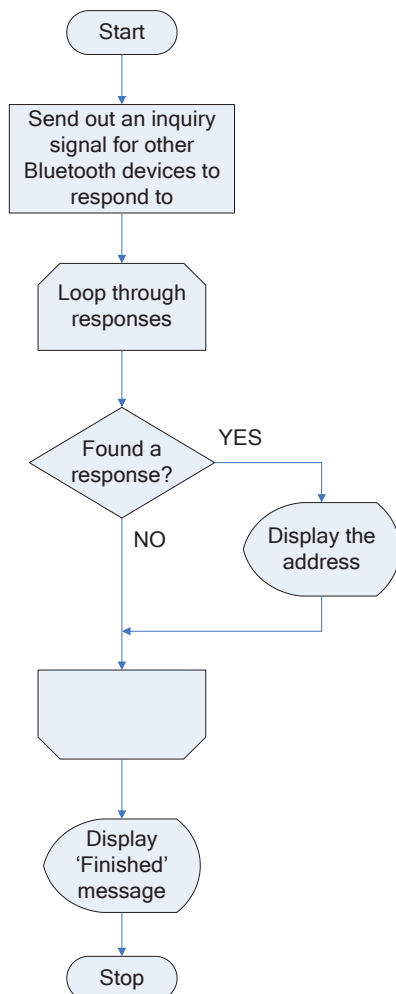
Note that the programs created for Exercise 1 needs to be used in conjunction with the program created in Exercise 2 to form a discoverable and discovering pair of boards. Using the two exercises as a linked pair allows students to explore discovering and discoverability using the same program from Exercise 1 in both Exercises, giving them continuity and the ability to see the earlier Exercise in action.

The two programs from Exercise 1 and Exercise 2 will form the core part of most of the subsequent exercises, additional commands being added in as the course progresses. As such a continuity approach can be taken where the final programs from previous Exercises can often be used as the starting point for the next. Using this approach allows students to see how the process flows and grows as they progress through the exercises.

The program students need to write in Exercise 1 is to discover any other Bluetooth devices in range. This necessitates a program in the corresponding Bluetooth device being active and that Bluetooth device being discoverable. Before starting students should download the program `BLUETOOTH_TEST.fcf` into both nodes. This will ensure that both nodes are discoverable.

7.3.2 Planning the program

Before writing a program there needs to be an idea of what the program objectives are, and an outline of how this will be implemented. Below is a rudimentary plane for an Inquiry program.



The plan is to send out the inquiry signal, then to loop through checking for any responses, displaying the address of any device that responds, and a 'Finished' message once the inquiry is complete.

Hardware requirements to achieve the plan include a Bluetooth module (obviously), and a display device that is capable of displaying the entire 12 digit address of the responding devices. Here we will be using the EB005 LCD display.

Software requirements include:

- Sending an Inquiry signal
- Checking for responses
- Obtaining the response data
- Displaying the data.

7.3.3 Required macros

Assuming the student has a basic familiarity with Flowcode and basic components such as the LCD display then only Bluetooth specific macros need to be covered.

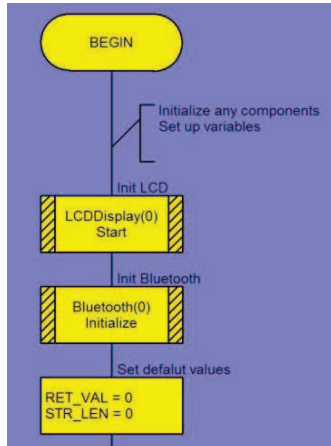
The following Bluetooth component macros will be needed for the program:

- Initialize
- CreateCommand
- SendCommand
- StringReceive
- StringRead

The macros break down into a Send command set – CreateCommand and SendCommand, and a Receive response set – StringReceive and StringRead.

7.3.4 Initializing

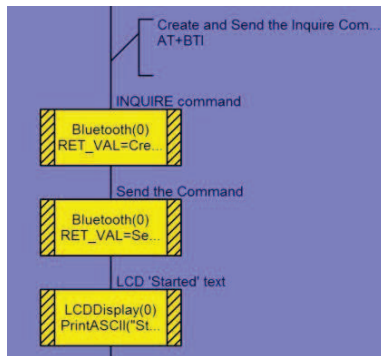
The basic program flow is to start off by initializing any components, and setting up any default variable values. Add in an LCD Start macro, and a Bluetooth Initialize macro.



The Bluetooth Initialize macro is required for all programs that use the Bluetooth component, and needs to be added to all Bluetooth programs. The best place to add initialization macros is right at the start of the program.

Add a calculation icon and enter default variable values in here. This can be updated whilst constructing the program to add in any new variables that are needed. Setting default values is good practice and can help avoid difficult to trap errors due to erroneous values.

Once all the components and variables are set up the first task can be considered, that of sending the Inquiry command.

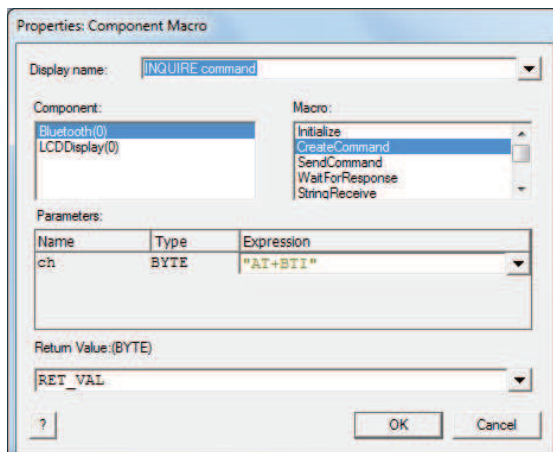


7.3.5 Sending a command

Before a command can be sent it needs to be created. The CreateCommand macro takes a *ch* parameter. Characters entered into the CreateCommand macro are added to the Command buffer. Commands can be built using a single CreateCommand macro, or by using several to build the Command up.

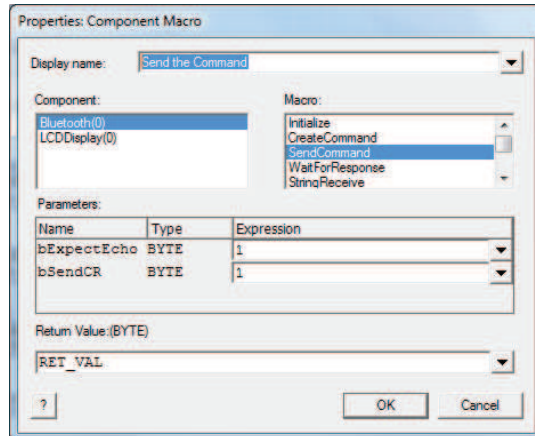
The Return value is 1 for success, or 0 for failure to create the command. The Command buffer is 32 characters in length, sufficient for the needs of the BLU2i command set. If the Command buffer is full CreateCommand will return 0 for a failure to create the command.

In this case the inquiry command AT+BTI needs to be sent. This can be built using a single CreateCommand macro. Add a CreateCommand macro to the flowchart.



The *ch* parameter is set to "AT+BTI", and a RET_VAL variable is used for the return value. The return value will not be checked in this example to avoid making this first program over complex. However in general it is good practice to use the return value for basic error checking to ensure commands are created correctly.

Once the Inquiry command is created it can then be sent using the SendCommand macro. Add a SendCommand macro to the flowchart.



The SendCommand macro sends the command that is currently in the CreateCommand buffer. The return value is 1 for a successful send, or 0 for failure to send the command. Once again this particular example will not check the return value in order to simplify the program.

The parameter bExpectEcho is set to 1 as the Bluetooth module generally echoes sent data back to the transmitting module. This will generally be set to one when sending commands in this course. Exceptions will be noted where they occur.

The parameter bSendCR is set to 1 to send a carriage return with the command. This will generally be set to one when sending commands in this course. Exceptions will be noted where they occur.

Finally add an LCD PrintASCII macro to display a message, such as "Starting inquiry", to say that the Inquiry command has been sent.

7.3.6 Checking for responses

Once the inquiry has been sent the next step is to listen for responses. There can be a number of responses so a loop structure is needed to go through them all. If an OK response is received then the inquiry is finished and the loop can be ended.

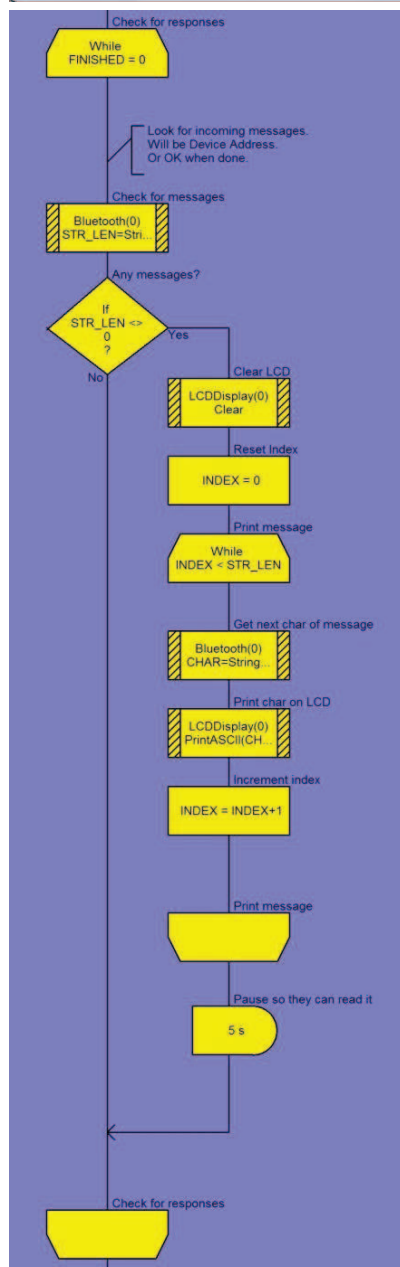
The basic flow is a Loop icon set to loop until a variable, FINISHED, is > 0.

To check for a Response a StringReceive macro is used. The StringReceive macro checks to see if a message has been received and returns the length of the message string or 0 if no message string has been received. The return value variable, such as STR_LEN, can be tested to see if it is greater than 0. If it is a message has been received and can be displayed.

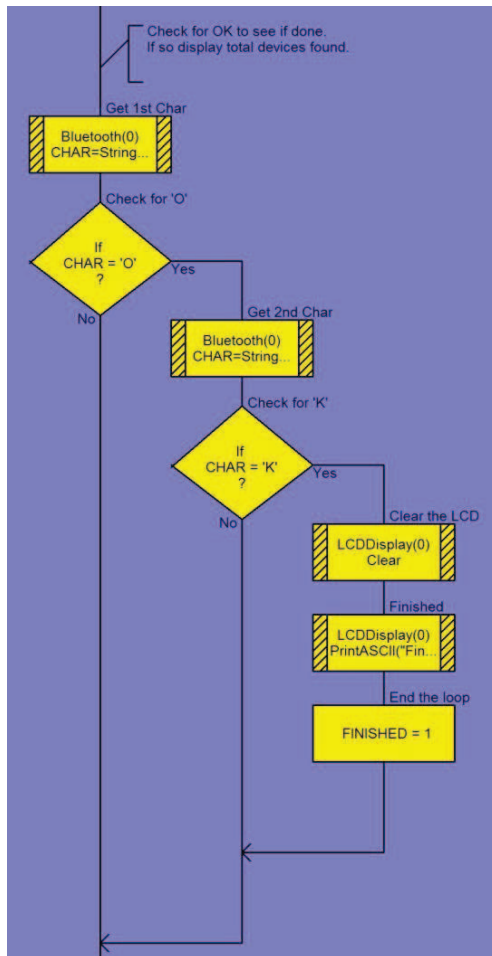
Clear the LCD ready for the message. The STR_LEN variable has the string length so we can display the message stepping through for STR_LEN number of times and printing the character. Create an INDEX variable set to 0 and loop through incrementing INDEX while INDEX is less than STR_LEN – the size of the message.

The StringRead macro returns the ASCII value of the character idx of the message. Using INDEX for the idx parameter will return the message characters which can then be displayed using the PrintASCII LCD macro.

A delay is required so that the response can be read.



Later programs will require us to use or interact with the addresses gathered here, but for this example all we need to do is display them. A delay of 5 seconds should be sufficient to demonstrate the addresses being received.



Responses need to be checked to see if they are an “OK” response, at which point the Inquiry is done and we can finish the loop by setting FINISHED to 1.

An OK command will have an ‘O’ and a ‘K’ as the first and second characters of the response. Retrieve the first character – idx 0 using the StringRead command. If it is an ‘O’ then check the second character – idx 1 of the message to see if it is the ‘K’. If so set FINISHED to 1 to end the loop.

To round off the program a ‘Finished’ message of some sort is required to inform us that the Inquiry has completed.

7.3.7 Running the complete program

The complete program can be found in example BT_EX1_NODE_B.FCF.

For testing a Bluetooth solution board is required. See the Getting started section for details on setting up the board and configuring the microcontroller.

In addition at least one other Bluetooth device needs to be active and discoverable. This can be the other Bluetooth solution board, or a third party Bluetooth device. Be warned though that not all Bluetooth devices will respond. When demonstrating the program it may be prudent to test the Bluetooth devices prior to using them for the demonstration to check if they can be discovered or not.

For example, if you have a Bluetooth-enabled mobile phone you will need to turn its Bluetooth on and set the phone's visibility so that it is not hidden.

The Bluetooth test file as detailed in the Getting started section is configured to respond to Inquiries and can be used to set up the second Bluetooth board.

Compile and download the program to the microcontroller on one of the Bluetooth boards. Briefly unplug the power, plug the power back in and press the Reset switch on the Programmer board. This ensures that the Bluetooth module resets and restarts correctly.

The LCD will display "Starting Inquiry". Next the LCD will display the 12 digit address of any devices found, with a 5 second pause between each one. Finally a "Finished" once an OK response is received indicating that the Inquiry is complete.

7.4 Theory: Discoverability

The inquiry command covered in the previous chapter is sent to all Bluetooth devices. However, not all devices will respond to the Inquiry signal. This is because Bluetooth devices can be set to hide themselves from other devices. Devices can also set themselves to be discoverable, in which case they will respond to the Inquiry command.

7.4.1 Configuring for start up

Bluetooth devices have a range of configuration options that cover everything from Passkeys and encryption to baud rate and the number of rings before answering. Defaults are generally used so that devices will normally work 'out of the box'. However it is good practice to configure devices with the correct settings upon start up to ensure the device is operating as desired.

In the BLU2i module these options can be configured using S registers, or via other AT commands.

7.4.2 Using S registers

The EZURIO BLU2i module uses a set of onboard registers to store operational parameters such as number of rings before answering, whether to echo a message, whether the device is connectable and discoverable etc. The registers used to store the parameters are called S Registers.

The S Registers are accessed using the AT command `ATS<register>=<value>`. For example `ATS512=3` will set the S Register 512 to the value 3. Looking up S Register 512 in the AT Command Set document, along with a parameter value of 3, informs us that the device will be configured to be connectable but not discoverable. To make the device discoverable only we can check the parameter values for S Register 512 and see that a value of 2 should be suitable.

There are a lot of S registers, details of which can be found in the AT Command Set document.

The major S registers are listed in the Appendix under `ATS<register>` in the AT commands section. There are a number of invaluable S Registers that will be used in most programs. Taking the time to learn these S Registers and their values will aid in not only correctly configuring devices, but in understanding how Bluetooth operates.

7.4.3 Using AT commands

There are a set of AT commands that configure the system in a specific mode. For instance `AT+BTO` opens the device and makes it discoverable. These AT commands can be used to quickly configure a device.

Examples of AT command configurations are:

Command	Description
AT+BTG	Makes the device Connectable but not Discoverable
AT+BTO	Makes the device Discoverable
AT+BTP	Device is Connectable and Discoverable
AT+BTQ	Responds to enquiries only.

7.4.4 S Registers or AT Commands?

Whilst both approaches are viable this course will primarily use S Registers for configuring the BLU2i module. Using S registers allows the user flexibility in that they can be set individually unlike the general mode configurations that single AT command configurations use. Users also have the ability to examine and look up the individual S Register settings to understand

what exactly is being configured - making them more transparent than the single AT commands.

7.4.5 Making a device discoverable

For this exercise we are interested in making the device answer automatically and to be discoverable and connectable, and for us to be able to send data to it. We also need to set this configuration on the device so that we can turn it on and off without needing to reconfigure every time.

Breaking the required configuration details down the following list is obtained:

7.4.6 Baud rate and settings

There are a number of S Registers that deal with the Baud rate and other hardware connection details. However, these should be left at their default values for the exercises in this course.

7.4.7 Discoverable and Connectable

- Discoverable sets whether the device will respond to an Inquiry signal
- Connectable sets whether the device can be connected to.

Both are set by the same S Register so need to be considered together. The S Register 512 takes a value 0-7 that specifies the power up state of the device. There are a number of complex states, however for now we can simplify it down to 3 states that could be useful for us.

- 2: set the device to be discoverable only.
Useful when a device needs to be the one that initiates contact, not receive it. Such as a paging system.
- 3: set the device to be connectable but not discoverable.
This state allows a device to hide from random Inquiries, but to be connectable by devices that already know its connection details (i.e. its Bluetooth address). Useful for secure communications.
- 4: set the device to be both discoverable and connectable.
As the plan is to connect to the Bluetooth device in subsequent exercises an S Register value of 4 will be used for most exercises in this course.

7.4.8 Number of rings before answering

Just like an answering machine waits for a number of rings before cutting in automatically, so does Bluetooth. S Register 0 specifies the number of rings before a call is answered. Setting S Register 0 to a value of 1 configures the device so that the call is answered on the first ring – i.e. immediately.

7.4.9 Whether to allow remote command for data sending

One important configuration item is whether the device will allow remote commands or not. Remote commands are when another device issues commands direct to this device. This allows the other device to send data, instructions, and to configure this device. S Register 536 set to 1 enables Remote commands. 0 disables it.

7.4.10 Saving the configuration

The AT command AT+W causes the current S Register values to be written to Non Volatile Memory so that they are retained when the power is turned off. This allows for a device to be configured for a particular task, and to retain those settings.

7.4.11 Implementing the configuration

Once the configuration details are set, the device can be rebooted using the ATZ command to allow the new settings to take affect and configure the device accordingly.

7.4.12 A basic configuration set up

A simple set of commands to make the device discoverable are:

AT&F*
ATS0=1
ATS512=4
ATS536=1
AT&W
ATZ

AT Command	S Register	Description
AT&F*		Return to factory settings
ATS0=1	0	Number of rings before automatically answering. 1 = 1 ring i.e. straight away.
ATS512=4	512	Specifies power up state. When set to 4 the device will be connectable and discoverable.
ATS536=1	536	Allows the device to enter Remote Command Mode. I.e. to be connected to so that AT commands can be sent to it.
AT&W		This command causes the current S registers to be written to Non Volatile memory so that they are retained when the power is turned off.
ATZ		Reboot to allow the earlier settings to activate and configure the device.

7.5 Exercise 2: Discoverability

7.5.1 Introduction

For a device to be found it needs to be discoverable. A Bluetooth device that is discoverable is able to respond to an Inquiry command from another Bluetooth device.

7.5.2 Objectives

- Develop a program that configures the Bluetooth device to be discoverable. This is the complementary exercise to Exercise 1.

7.5.3 Pre-requisites

- An understanding of the Discovery process as detailed in Exercise 1. In particular the ability to be able to recognize an Inquiry command.
- An understanding of receiving and retrieving message data (See the Bluetooth Component Help file for details on the macros involved, and an outline of the relevant strategies)

7.5.4 Hardware/Software requirements

The following items of hardware are required:

- Bluetooth solution for the Exercise program.
- 1 or more additional Bluetooth devices for demonstrating the program.
Note: The second Bluetooth board from the Bluetooth solution can be used for this Exercise.
Either the program from Exercise 1 can be used to send the Inquiry command, or the test program BT_EX1_INQUIRY.fcf can be used to set up the board. See the Getting started section for details.
- Set hardware jumpers as specified in the Getting starting section.
- Configure the microcontroller as specified in the Getting starting section.

7.5.5 Exercise information

The configuration data to be sent:

```
AT&F*  
ATS0=1  
ATS512=4  
ATS536=1  
AT&W  
ATZ
```

The objectives can be broken down into the following:

Tasks

- Send the configuration commands.
- Check for messages.
- Display any messages.

Control structures:

- Checking for and displaying responses.

7.5.6 Learning outcome

Primary learning outcomes for this exercise are:

- Understanding the discovery process.
- An understanding of how to configure the BLU2i device.
- An understanding of S Registers.
- Checking for and retrieving responses.

7.5.7 Additional tasks

- Referring in the Bluetooth module manual S Register settings and make the Bluetooth module non-discoverable.
- What implications does this have for the person using the equipment after you?
- Using the programs in Exercise 1 and 2 make a note of the Bluetooth addresses of each of the Bluetooth modules – you will need these in the next exercise.

7.6 Practical implementation: Discoverability

Exercise 2 accompanies Exercise 1 in that having shown how Discovering devices work in Exercise 1 the next step is to show how to make devices discoverable. In addition, configuration methods and the use of S Registers are discussed. It is important to stress that Just as the AT commands used here are particular to the specific module used (The EZURiO BLU2i device) so are the S Registers are also a feature of the BLU2i chip and not an inherent part of Bluetooth. Different chips may have different registers, or even use none at all. However the general principles are the same i.e. setting how many rings to wait, or electing to make a device discoverable.

Before starting connect the USB lead to the second Bluetooth system – you can leave the program from Exercise 1 in the first Bluetooth system and we will use it to test if the program created here works.

7.6.1 Continuing development

The Students programs created for Exercise 1 can be used in conjunction with the program created in this exercise to form a discoverable and discovering pair of boards.

The two programs from Exercise 1 and Exercise 2 will form the core part of most of the subsequent exercises, additional commands being added in as the course progresses. As such a continuity approach can be taken where the final program from previous exercises can often be used as the starting point for the next.

7.6.2 Configuring the Bluetooth device

For this Exercise the Bluetooth device needs to be configured to be discoverable and connectable. We will not be connecting to it in this particular Exercise; however we may as well set up this basic configuration now.

For this exercise use the following configuration commands:

AT&F* - optional

ATS0=1

ATS512=4

ATS536=1

AT&W

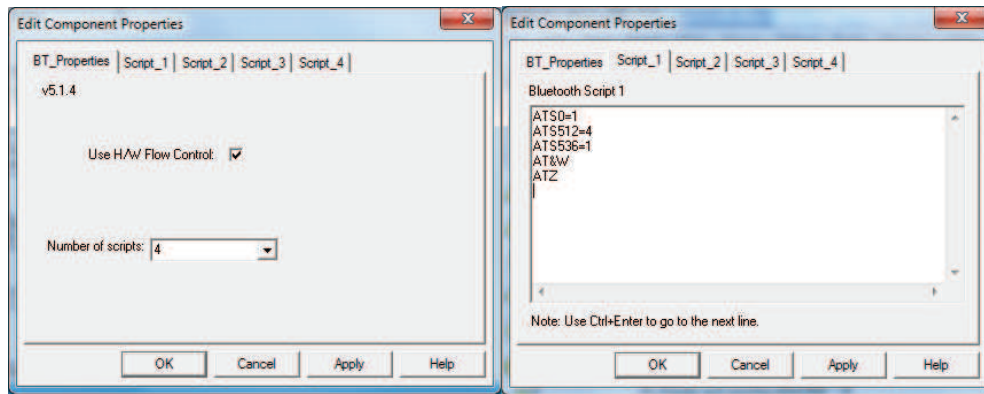
ATZ

These commands will set the device up to answer immediately, and to be both discoverable and connectable.

7.6.3 Single commands and scripts

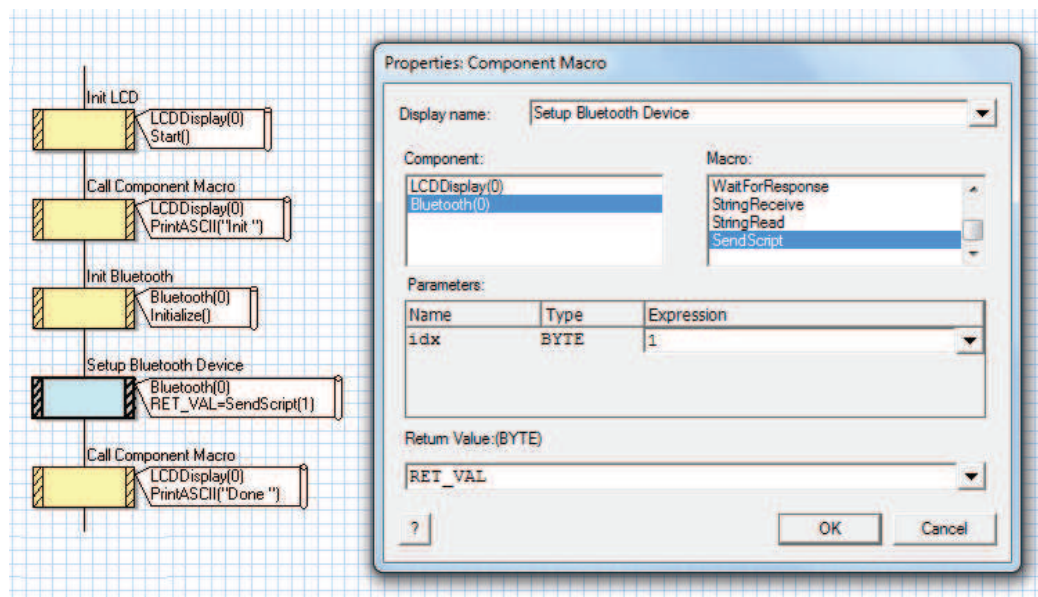
The Commands can be implemented one of two ways: Single commands or via scripts. Single commands can be created and sent using the *CreateCommand* and *SendCommand* macros detailed in Exercise 1. In addition scripts can be used.

Scripts are a set of command entered into a text box. Scripts can be accessed from the Bluetooth components Properties page. There are 5 tabs: a General properties tab and 4 script tabs. The Number of Scripts option on the general properties page can be set from 1-4 allowing use of 1-4 scripts in the program.

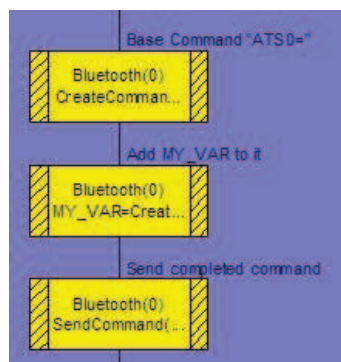


Text can be entered into the text boxes on the script tabs. Any text can be entered, AT commands or data for instance. One item to note is that *Ctrl + Enter* is needed to enter a carriage return in the script boxes. When a script is sent the script is sent one line at a time as if each line were a separate command that had been created and sent using *CreateCommand* and *SendCommand*.

Scripts are sent using the *SendScript* macro. *SendScript* takes a single parameter *idx*, the number of the script to send. The *idx* value will be 1-4 depending on which script is to be sent. The optional return value is 0 for success or 255 for any problems.



The advantage of using scripts is that you can send a whole group of commands with one single macro. The disadvantage is that the commands in the scripts are static and cannot be built up in the program.

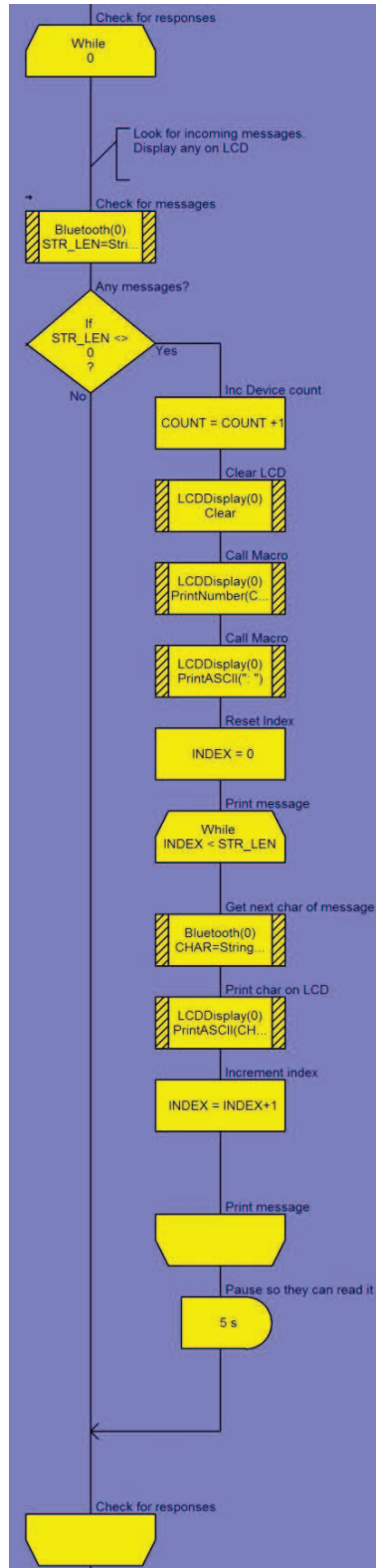


For instance we can set the device to answer immediately using *ATS0=1* in a script, but could build a command such as *ATS0=<MY_VAR>* using a variable *MY_VAR* to set S Register 0 to a program variable that depends on user input. Scripts are extremely useful for blocks of commands that will not vary with program start up. This makes scripts ideal for start up configuration settings.

Adding the text:

ATS0=1
ATS512=4
ATS536=1
AT&W
ATZ

to a script allows us to send the whole set of configuration setting commands in one go.



Once the device has been configured to be discoverable nothing else is actually needed for this Exercise. However to allow users to see that the device has received an inquiry command a loop can be added to check for messages and to display them on the LCD. This is a simplified version of the loop created in Exercise 1. There is no need to check for the OK message, or to do anything other than simply display any incoming messages.

7.6.4 Running and demonstrating the program

For best effect the program built here can be used with the Inquiry program built in Exercise 1. The program from Exercise 1 can be used in the first Bluetooth board, and the Exercise 2 program can be used with the second Bluetooth board.

- Create the program, noting as you do the 12 digit device address of the Bluetooth board that will be used with the Exercise 2 program.
- Download and run Exercise 2 on the Bluetooth board whose address you have noted.
- Download and run an Inquiry program to the other board. This can be either:

A) The program from Exercise 1

B) The test program BTC_DISCOVER.FCF

- Monitor the LCD Display on the board to be discovered. When the *AT+BT/* Inquiry command is sent it will be displayed on the LCD.
- Monitor the LCD on the Inquiry program board. When the discoverable board is sent the Inquiry command it will respond by returning its 12 bit address (the number you noted down earlier). Check the LCD display to see that the address is displayed.

Experiment # 8

Connecting Bluetooth devices

8.1 Theory: Connecting – Addresses

As no physical connection exists between two devices all communications are capable of being picked up by any other Bluetooth device. An immediate problem is how to communicate with one particular device and not inadvertently communicate with other devices.

All Bluetooth devices have an address, a unique 12 digit hexadecimal number. This address is the same number that is returned in response to an inquiry command. This address can be either retrieved via an inquiry command, or can be stored in the program if a specific device address is to be used and is known in advance.

If a Bluetooth device is connectable and the address is known then other Bluetooth devices can initiate connection to that device. Generally devices will be discoverable and connectable so that the address can be determined from an Inquiry command. However it is possible for devices to be connectable, but not discoverable. For instance a set of phones that are designed to communicate only with each other may be set to be connectable but not discoverable. The pair of phones have the address of each phone in memory so that a discovery command is not needed, and the phone is ready to start communicating with its partner device.

The basic connection command is:
ATD<bt_addr>

Where <bt_addr> is the 12 digit hexadecimal address of the device to initiate connection with.

The Bluetooth addresses of the BLU2i modules are set in the factory and cannot be changed. The address is written on both the underside of the Bluetooth E-Block and the underside of the BLU2i module itself. Note that if the BLU2i modules are replaced or swapped for any reason, the address displayed on the underside of the Bluetooth E-Block will not be the same as the address of the BLU2i module itself.

8.2 Exercise 1: Connecting to a device

8.2.1 Introduction

If the address of a Bluetooth device is known, and the device has been configured to be connectable, then that device can be connected to.

8.2.2 Objectives

- Develop a program in one Bluetooth system (node B) that connects to another Bluetooth system (node A), and displays the reply to show that the connection has succeeded.
- Develop a program in one Bluetooth system (node A) that allows another Bluetooth system (node B) to connect to it. This program is supplied for you in BT_EX3_Node_A.fcf.

8.2.3 Pre-requisites

- An understanding of the Discovery process as detailed in Exercise 1 and Exercise 2.

8.2.4 Hardware/Software requirements

The following items of hardware are required:

- Bluetooth solution for the Exercise program.
- 1 or more additional Bluetooth devices for demonstrating the program.
Note: The second Bluetooth board from the Bluetooth solution can be used for this Exercise.
Download the program BT_EX3_Node_A.fcf into this second system.
- Set hardware jumpers as specified in the Getting Starting section.
- Configure the microcontroller as specified in the Getting Starting section.

8.2.5 Exercise information

The Initiate Connection command is:

ATD<bt_addr>

Where <bt_addr> is the address of the second Bluetooth board.

Set up two programs: one in Node B to Dial a second Bluetooth device in Node A. When you issue the ATD command the receiving node will send the acknowledgement CONNECT 123456789012. Display this on the LCD display of node A.

8.2.6 Learning outcome

Primary learning outcomes for this exercise are:

- Connecting to another Bluetooth device.

8.2.7 Further work

The LCD display only has 16 characters. Devise an extension to the code in BT_EX3_Node_A.FCF so that all any text overflow is displayed on line 2 of the display.

8.3 Practical implementation: Connecting

Like previous exercises students actually need to develop two program – one in the Bluetooth device initializing the communication, and the second in the device receiving the initial communication.

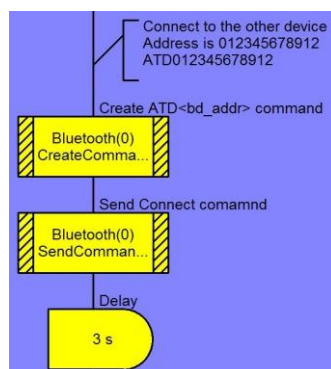
This exercise is a quick and simple one from the workload point of view, but has a number of technical pitfalls to be aware of.

1) Device address. The example program BT_EX3_Node_B.FCF uses the address of a Bluetooth device used for testing here at Matrix Multimedia. This will need to be edited to match the address of the board that is to be used with the example program. For this you should use the address you found in Exercise 2.

Be clear about which is the sending node and which is the receiving node and what their addresses are.

2) The Bluetooth device to be connected to needs to have been configured to be connectable. Once again the program BT_EX3_Node_A.FCF has been set up with this issue in mind.

3) Once a connection has been made there needs to some way of verifying to ourselves that the connection has been made. The program BT_EX3_Node_A.FCF is set up to display any messages sent, so we can simply send a message and check that it is received by the other device. The program you write should display the returning message from node A.



A delay is needed after the connection command has been sent to allow time for the connection messages to be handled. No error checking is done at present. The program assumes that the connection will succeed, which may not always be the case. A better method would be to use responses to error check the process. Responses and error checking will be dealt with in a later chapter.

8.3.1 Resetting the systems

The Bluetooth board EB024 is not equipped with a reset button. The only mechanism of issuing a reset is to remove power. If you have programmed the Bluetooth module to carry out some activity then pressing the reset button on the Multiprogrammer board will not necessarily reset the Bluetooth module.

For this reason when developing pairs of programs it **may** be necessary to remove power from the system and reboot the Bluetooth modules.

For this exercise we recommend that you remove power from both systems each time you download a program. Then power up the receiving system, press reset and give it a few seconds to set up. Then power up the transmitting system, press reset and give it a few seconds to set up.

8.4 Passkeys and Pairing

8.4 Theory: Passkeys and Pairing

Bluetooth communicates via radio which is an inherently unsecure transmission medium. Any device within transmission range can receive the signals sent to any other device. Using a unique address solves the problem of specifying which device you intend to communicate to, but other methods are needed to prevent unauthorized access to a device.

The basic security system used with Bluetooth is called Pairing. Pairing is when two devices connect to each other using both a device address and the device's secret "link key" known as the Passkey. The Passkey can not be retrieved from another device in the same way the address can. To be able to connect to a device you need to know the Passkey number for that device.

8.4.1 Sending the Passkey command

To set up a device so that it can be paired with a Passkey needs to be set up for the device. The command to set up a Passkey is:

```
AT+BTK=<passkey>
```

Where <passkey> is the Passkey PIN number. The Passkey is a 0-8 digit number. Setting the Passkey to a blank string clears the current passkey. Generally Passkey PIN numbers are supplied with documentation that accompanies a particular device. Device PIN numbers should be kept safe, just like you would with bank card PIN numbers.

For the programs in this course a simple Passkey of "1234" will be used for all exercises

8.4.2 Initiating pairing

To pair to a device the device initiating contact needs to send two commands

A passkey command

```
AT+BTK=<passkey>
```

Where <passkey> is the passkey value of the device to be paired with.

An Initiate Pairing command is:

```
AT+BTW<bt_addr>
```

Where <bt_addr> is the address of the Bluetooth device to be paired with.

8.4.3 When to send the Passkey command

The Passkey command can be sent before or after the Initiate Pairing command. If the Passkey command is sent before then Pairing will be initiated upon sending the Passkey command. If the Passkey Command has not been sent when an Initiate Pairing command is received the device to be paired to will send an unsolicited PIN? response which will need to be checked for by the program. Pairing will not proceed until the Passkey is sent. Responses will be covered in a later chapter, so for now send the Passkey command before the Pairing command. An Initiate Pairing command on its own is not enough. A passkey command needs to be sent as well.

In many instances, such as when pairing a mobile phone with a headset, the Passkey will be in the product documentation and the program will prompt you to enter the Passkey.

8.5 Exercise 2: Passkeys and Pairing

8.5.1 Introduction

Devices can be paired with each other for communications. Pairing requires the address of the device to pair with, and a Passkey value to establish contact. Here you will need to develop two programs – one that establishes what the pass key is in a system, and the other that connects to that system and sends data to it.

8.5.2 Objectives

- Develop programs for two Bluetooth systems that allow pairing to take place. Develop a program for Node A that assigns a Passkey of “1234”, and make it discoverable. Develop routines that show any data sent to the node on the LCD. Develop a program for node B that Pairs with node A using Node A's address and Passkey.
- Once Paired establish a simple communication between the systems that shows communication is taking place – i.e. a counter on the sending count data as a single byte which is also displayed on the receiving node.

8.5.3 Pre-requisites

- An understanding of the connection process as detailed in Exercise 3.
- An understanding of creating, sending and receiving commands as detailed in Exercises 1 and 2.

8.5.4 Hardware/Software requirements

The following items of hardware are required:

- Both Bluetooth solutions are required for this Exercise.
- Solution 1 will initiate pairing.
- Solution 2 will be paired to.
- Set hardware jumpers as specified in the Getting starting section.
- Configure the microcontroller as specified in the Getting starting section.

8.5.5 Exercise information

The Initiate pairing commands are:

- `AT+BTK=<passkey>`
Where <passkey> is the passkey value of the device to be paired with.
- `AT+BTW<bt_addr>`
Where <bt_addr> is the address of the second Bluetooth board.

Once a connection has been established loop through and send the numbers 0-9 to be displayed on the other device.

8.5.6 Learning outcome

Primary learning outcomes for this exercise are:

- Understanding of the pairing process.
- Understanding of the role of the Passkey.
- Understanding of the security implications of the Passkey.

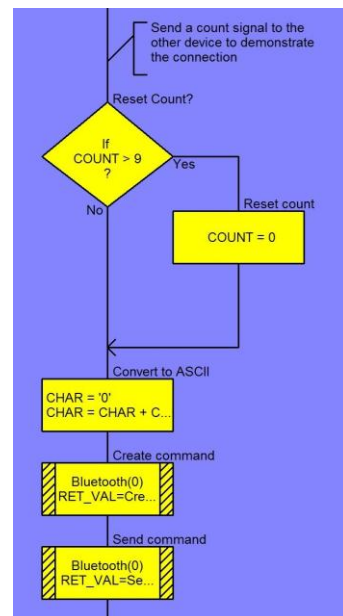
8.5.7 Further work

- Using a keypad and an array variable, develop two programs that allow the passkey of both the receiving system to be set, and that allows the passkey used by the transmitting system to be set.
- Develop a program that displays the ‘friendly name’ of all other Bluetooth devices in the locality.

8.6 Practical implementation: Passkeys and pairing

8.6.1 The basic program

Pairing is a key process in Bluetooth operations. The pairing process is relatively straight forward on one level, but complex on another. Pairing involves the sending of both the Passkey and an initiate pairing command.



AT+BTK=<passkey>
AT+BTW<bd_addr>

For Example:
AT+BTK="1234"
AT+BTW012345678912

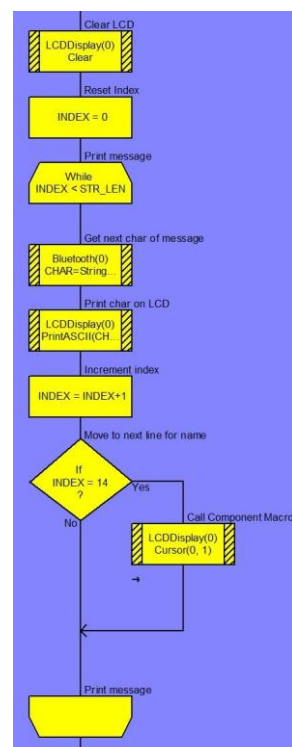
If the details are known in advance then they can be simply entered into the program and the commands built and sent. If the passkey and address are not known in advance then they need to be established.

The boards in the Bluetooth solution have the device addresses on labels on the bottom of both the BLU2i module board, and on the bottom of the E-block. You may wish to note down the addresses for the various boards.

To demonstrate the connection is working a simple 0-9 repeating signal can be sent to be displayed on the receiving device.

Note: The delay icons used in this chapter allow for the devices to finish communicating with each other. A better method would be to know when a process is complete, or more importantly if it has failed. This can be done with responses, which will be covered next chapter.

8.6.2 Advanced features: Choosing what device to connect to



Device addresses can be retrieved using the Inquiry command. However knowing if we need to connect to connect to device 007284972989 or 00234869030 is another matter. The AT+BTI Inquiry command can be extended to retrieve extra information above and beyond the 12 digit address. AT+BTIN retrieves not only the device address, but the friendly name of the device as well. The friendly name is a device description string that is easier for users to understand than the device address.

For example:
012345678912, "TDK BLU2i RS232"
012345678912, "TDK Headset S102"

Displaying the friendly address can help users select the correct device out of a list. The practicalities of displaying the friendly name are a simple modification to the display response code from Example 1. Adding in the command space the first 14 characters can be displayed on the top line of the LCD, and the rest on the bottom line by moving the cursor when the first 14 characters have been displayed.

In addition we can add the 12 address characters into an array, storing the current address for further use. This can be done at the

same time as we are printing out the characters. Add the first 12 characters to the current address.

Obtaining the passkey is more difficult as this it can not be retrieved via an AT command. For the examples used in this course the Passkey "1234" is used throughout. In practice a passkey may need to be obtained from the device documentation and either entered in the program code, or entered into the program manually.

8.6.3 Resetting the systems

The Bluetooth board EB024 is not equipped with a reset button. The only mechanism of issuing a reset is to remove power. If you have programmed the Bluetooth module to carry out some activity then pressing the reset button on the Multiprogrammer board will not necessarily reset the Bluetooth module.

For this reason when developing pairs of programs it may be necessary to remove power from the system and reboot the Bluetooth modules.

For this exercise we recommend that you remove power from both systems each time you download a program. Then power up the receiving system, press reset and give it a few seconds to set up. Then power up the transmitting system, press reset and give it a few seconds to set up.

8.7 Checking responses

8.7 Theory: Checking responses

In the previous exercise the pairing worked but only with a large delay put in to allow the devices to work through the pairing messages and responses. This is all well and good if we know that the devices will pair correctly, and can afford to wait until it is definitely done. However what happens if there is a problem, or the connection can not be established in the delay we set? A more useful method would be to monitor the process checking for the right response at the right time.

When a command is sent a response is generally sent to that command. For many commands and occasions the response will be an 'OK' message. However many commands have specific responses. The Discover command for instance sends addresses as responses, and once finished then it sends an OK response.

8.7.1 Solicited and unsolicited responses

When a response is the direct result of sending a command it is a solicited response i.e. a response that is expected. For instance sending an AT+BTK="1234" command will result in a solicited response of "OK".

Sometimes a response is sent that is not in response to a specific command. This is an 'unsolicited' response. For instance when an initiate pairing command is sent an expected response (solicited) of "OK". will be sent. However if the Passkey has not been provided an unsolicited response of "PIN?" will also be sent. This is an unsolicited response as it is not in response to the command sent but is instead a prompt to the device to provide extra information, in this case the Passkey value. Unsolicited response can be sent at various times to indicate either a need for further interaction, or to provide additional feedback. Examples include the CONNECT and RING responses sent during the establishment of communications to inform the other device of the current status of the operation.

Unsolicited responses are sent when required. Using the pairing example from above, if the Passkey command had been sent before the Initiate pairing command then an unsolicited "PIN?" Response would not be needed and would not therefore be sent.

Unsolicited responses are detailed in the AT Commands Set document.

8.7.2 Response handling macros

There are two macros available to help with handling responses: StringReceive and WaitForResponse.

The StringReceive macro checks to see if a response has arrived and returns 0 or the length of the response message. *StringReceive* was used in the section on Discovery and details of using *StringReceive* can be found in the Help file in Flowcode. *StringReceive* is useful for general message monitoring when messages may be present. Exercise 1 demonstrates *StringReceive* in action monitoring incoming messages for Addresses.

WaitForResponse pauses program flow until a response message is received.

WaitForResponse takes the parameters response_code and timeout.

The response_code parameter indicates the type of response to wait for. This takes the form of a 1-7 value that refers to the following response types:

Response type	response_code value	Notes
OK	1	Can means no more than confirming the command has been successfully received. For Example An OK with AT+BTW<bt_addr> does not confirm pairing occurred; only that

		the command to initiate pairing was received.
ERROR <xx>	2	See AT Command Set document for details of ERROR code. Examples include: 09 = Invalid Bluetooth address 16 = Pairing in progress 24 = Remote address same as local address 33 = S Register value is invalid
CONNECT <bt_addr>	3	
NO CARRIER	4	
AUDIO <string>	5	<string> indicates the status of the Audio channel. Can be: ON, OFF or FAIL
PAIR <n> <bt_addr>	6	Values for <n>: 0 = success 1 = timeout occurred 2 = Other unsuccessful outcome.
RING <bt_addr>	7	

The timeout parameter is the number of milliseconds to wait before timing out and assuming that no response is coming. Normally this will be used to help indicate if a command failed or not.

The return value for *WaitForResponse* is as follows:

- Returns zero to indicate that the specified response has been received.
- Returns 0xFF for timeout or an illegal response code
- Returns the number of characters received if an unexpected response is received

WaitForResponse can be used for both error checking and sequence control. Sequence control can be accomplished by checking for the expected response type. For example an Initiate pairing command should return a PAIR <bt_addr> response (response_code value 6). Error checking can be done by checking for timeouts or unexpected responses. If a timeout occurs the expected response has not arrived and appropriate action needs to be taken. Generally a timeout will indicate a communications failure at that stage of the sequence. If an unexpected response occurs it can be further checked to see if it requires action (e.g. it is an unsolicited response) or if it is an error message that needs processing or reporting.

The *WaitForResponse* command does not look for a specific error number, address or audio state. *WaitForResponse* indicates if a response of the specified type was received or not. To find check the specific details of a response the *StringReceive* command can be used to retrieve the response for further checking.

For example, to find out a specific Bluetooth address after receiving zero (success) from a *WaitForResponse* (3) command, send the *StringReceive* command. The received string will then be the Bluetooth address.

8.7.3 Command and response sequences

When a command is sent one or more responses are returned. For example when a set of Pairing commands are sent. The sequence will be something like:

AT+BTW0123456789012	- Command sent (request to initiate pairing)
OK	- Response (Pair command received and is ok).
PIN?	- Unsolicited response (PIN number request)
AT+BTK="1234"	- Command sent (Passkey)
OK	- Response (OK)
PAIR 0 012345678912	- Unsolicited response (Pair succeeded)

Note the two unsolicited responses. The AT protocol is a Command/Response protocol. A command is responded to. In the case of the AT+BTW<bd_addr> command the response is OK to indicate that the command has been received. However the command itself requires further action when it is processed. These actions cause the device to issue an unsolicited response. In this case firstly a request for the passkey to be sent. Later ,once the Passkey has been sent and responded to with an OK to say that the command has been received; a further unsolicited response is sent to inform the device that Pairing was successful.

Side note: For some commands, such as AT+BTW<bt_addr> the OK response confirms that the command was received correctly. It does not mean that the request was successful. Subsequent responses may need to be examined to indicate that the command was successful.

Details of the expected responses can be found in the AT Command Set documentation. By checking that the expected response has arrived it is possible to determine whether an error has occurred, or when something unexpected has occurred. For instance a return value of 255 (0xFF) indicates that a timeout has occurred, possibly due to a connection issue. A response to an Inquiry with AUDIO or CONNECT 123456789012 would indicate that something has gone wrong with the sequence as a different response was expected.

8.8 Exercise 3: Checking responses

8.8.1 Introduction

When a device receives a command it returns a response. By checking these responses it is possible to both monitor for any errors that occur and to wait for expected responses before continuing a sequence of commands.

8.8.2 Objectives

- Develop programs for two Bluetooth systems that allow pairing to take place. Develop a program for Node A that assigns a Passkey of “1234”, and make it discoverable. Develop routines that show any data sent to the node on the LCD. You can use the program from Exercise 4 for this.
- Develop a program for node B that Pairs with node A using Node A’s address. Don’t use the Passkey with the Pair command – alter the program you wrote in Exercise 4 to include routines that provide the Passkey when prompted. Once Paired wait for and check the responses to ensure that the pairing completes successfully.
- Report any errors that occur on node B

8.8.3 Pre-requisites

- An understanding of the Pairing process as detailed in Exercise 4.
- An understanding of the WaitForResponse macro as detailed in the Bluetooth component help file.
- An understanding of the Response types as detailed in the Bluetooth component help file.

8.8.4 Hardware/Software requirements

The following items of hardware are required:

- Both Bluetooth solutions are required for this Exercise.
- Solution 1 will initiate pairing.
- Solution 2 will be paired to.
- Set hardware jumpers and configuration as specified in the Getting starting section.

8.8.5 Exercise information

The Initiate pairing commands are:

- `AT+BTK=<passkey>`
Where <passkey> is the passkey value of the device to be paired with.
Expected response is OK.
- `AT+BTW<bt_addr>`
Where <bt_addr> is the address of the second Bluetooth board.
Expected response is OK.
- Once Pairing has occurred an unsolicited response PAIR <bd_addr> will be sent.
- If the Initiate connection command is successful a CONNECT <bd_addr> response will be returned.
- Load the program BT_BASIC_CONNECT.FCF on Bluetooth Solution 2.
- A list of response types can be found below.

8.8.6 Learning outcome

Primary learning outcomes for this exercise are:

- Command and response sequences.
- Response types.
- Error checking and sequence monitoring.
- Unsolicited responses.

Response type	response_code value
OK	1
ERROR <xx>	2

CONNECT <bd_addr>	3
NO CARRIER	4
AUDIO <n> <audio state>	5
PAIR <bd_addr>	6
RING <bd_addr>	7

8.9 Practical implementation: Checking responses

Two important bits of information to note down are the Bluetooth Device addresses, and the Passkeys. For this exercise it is assumed that the general Passkey "1234" will be used. However the Bluetooth devices will need to be modified accordingly.

The task requires two programs, one to initiate pairing, and one to display data sent to the paired device. The Pairing programs developed in Exercise 4 can be used as the base of Exercise 5 allowing the student to see how their code progresses.

The display program only needs to display data sent, so the display program from Exercise 4 can be used as is.

The main focus will be with the Pairing program, and expanding that program to include response checking.

8.9.1 Establishing the expected sequence

Before a sequence can be error checked the pattern of commands and expected responses must be established.

AT+BTK="1234"	- Passkey command
OK	- Response. Type 1.
AT+BTW0123456789012	- Initiate pairing command
OK	- Response. Type 1.
PAIR 0 012345678912	- Response. Type 6.
ATD0123456789012	- Initiate connection command
CONNECT 012345678912	- Response. Type 3.

The sequence, with types and suggested timeout values is as follows:

```
SendScript(1): ATZ – Reset
                WaitForResponse(1,200)
                OK response from ATZ

SendScript(3): AT+BTK="1234"
                WaitForResponse(1,10)
                OK response from Passkey command

SendScript(2): AT+BTW00809894E5DF - Pairing command
                WaitForResponse(1,10)
                OK response from Initiate pairing command
....
                WaitForResponse(6,200) – Unsolicited PAIR command sent when pairing complete.

SendScript(4): ATD00809894E5DF
                WaitForResponse (3,200)
```

8.9.2 Additional sequence considerations

The above sequence is a simple one as all the Response type are covered and can be waited for. However if the Passkey command has not been sent when the Initiate pairing command is sent then an unsolicited PIN? Response will be received after the solicited OK response. The PIN? Response will be registered as an unexpected response and will need to be trapped and the appropriate action taken (namely sending the Passkey command). It is recommended that students send the Passkey first to simplify the program initially. Advanced students can then tackle the unsolicited PIN? response as both an exercise in handling unsolicited responses and in advanced error checking.

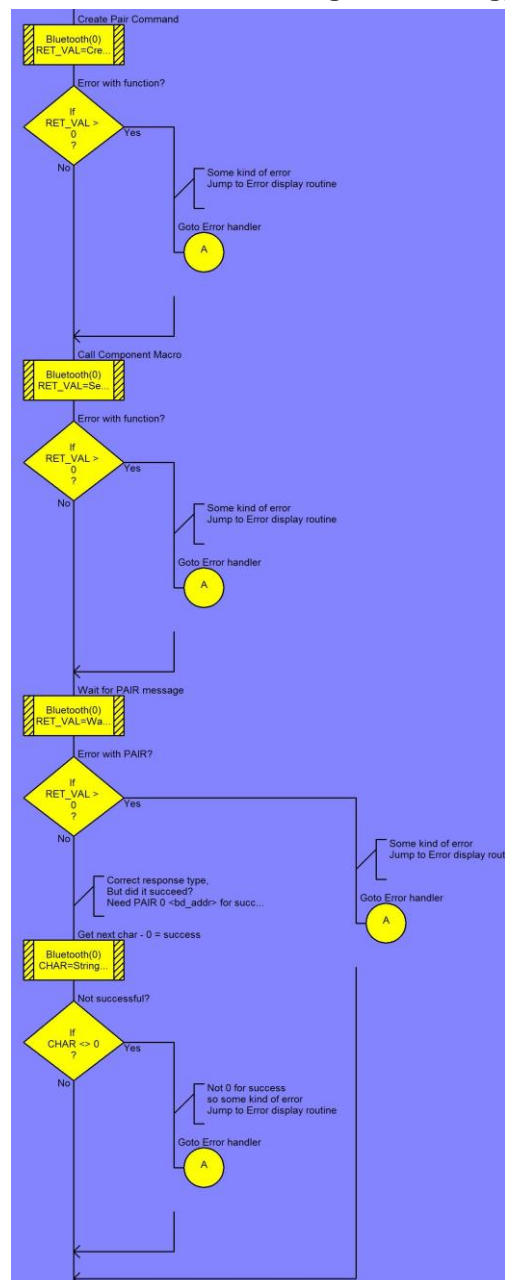
8.9.3 Using the WaitForResponse macro

Once a sequence is known adding WaitForResponse macros is relatively straightforward. Add the macro and edit the properties so that the expected response code parameter uses the correct response type, and an appropriate timeout value is used. Set a return value (RET_VAL is the generic return value variable used throughout this course) for the error checking. The timeout value can vary depending on factors such as baud rate and speed of operation. The example values set out below are the ones used successfully when developing this course and can be used as guides to setting the values.

Example timeouts values used for the example programs is:

Response	Value in milliseconds
Awaiting general OK response	10 - 100
ATZ reset OK response	200
PAIR command	200
CONNECT response	200
RING	200
AUDIO	200

8.9.4 Error checking methodology



A basic error checking methodology has been adopted here. As the commands are created and sent they are first checked using the return value from the functions. A failure here indicates that there is a problem with either creating or trying to send the command.

Once a command has been sent the WaitForResponse return value can be checked to see if it received a response of the specified type (0 for success). If not an error message is displayed on the LCD detailing the stage at which the sequence failed. A connection icon is used to jump to the end of the program to exit without running the rest of the program.

If a successful response has been received this too may need further checking to determine that the action was both successful and that the result was what was expected.

The WaitForResponse (6,200) command will only wait for the "PAIR" response (plus a following space character), which on its own does not indicate a successful pairing. Immediately following the reception of "PAIR " is a single digit representing the outcome of the pairing attempt. 0 means success, 1 means timeout and 2 means another unsuccessful outcome. Following this single digit is a space, then the 12-digit hexadecimal string representing the address of the other Bluetooth device. In this example both the outcome and the address need to be checked to ensure that we have successfully paired to the correct device.

Advanced error checking could involve separate user macros testing the responses and giving

more detailed information – e.g. displaying ERROR <nn> codes, or checking for a NO CARRIER and giving users the option to turn on the other device and try again etc.

Experiment # 9
GPS Module Settings

Exp 9-1 GPS Module Baud Rate Setting

OBJECTIVE

After completing this experiment, you should be able to set the baud rate of the GPS Module

DISCUSSION

Setting the same transmission speed (baud rate) and communication protocol provides a communication channel to different communication devices for transferring data between them.

In this experiment, you will set the baud rate of GPS Module to agree with that of COM port for transmitting and displaying longitude and latitude message on graphical user interface.

PROCEDURE

1. Turn on dip-switch(MODULE) 1 and 2, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the RS-232 connector J2 on DGS-100 to the RS-232 port on PC using the RS-232 cable. Turn all power on. Open the DGS-100 main screen as shown in Figure E1-1-1.

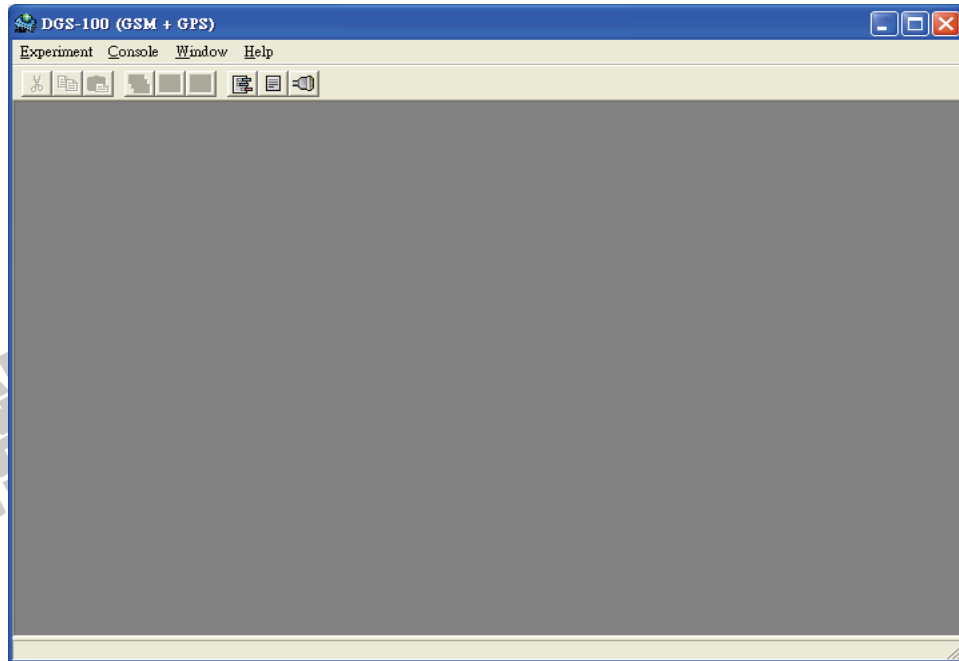


Figure E1-1-1 DGS-100 main screen

3. Select **Experiment** -> **OpenPort** command to open the Connection Setting dialog box as shown in Figure E1-1-2. Specify the BaudRate of RS-232 serial port to 4800, and then click **Set** to setup a communication link between DGS-100 GSM/GPS Experimental Set and PC.

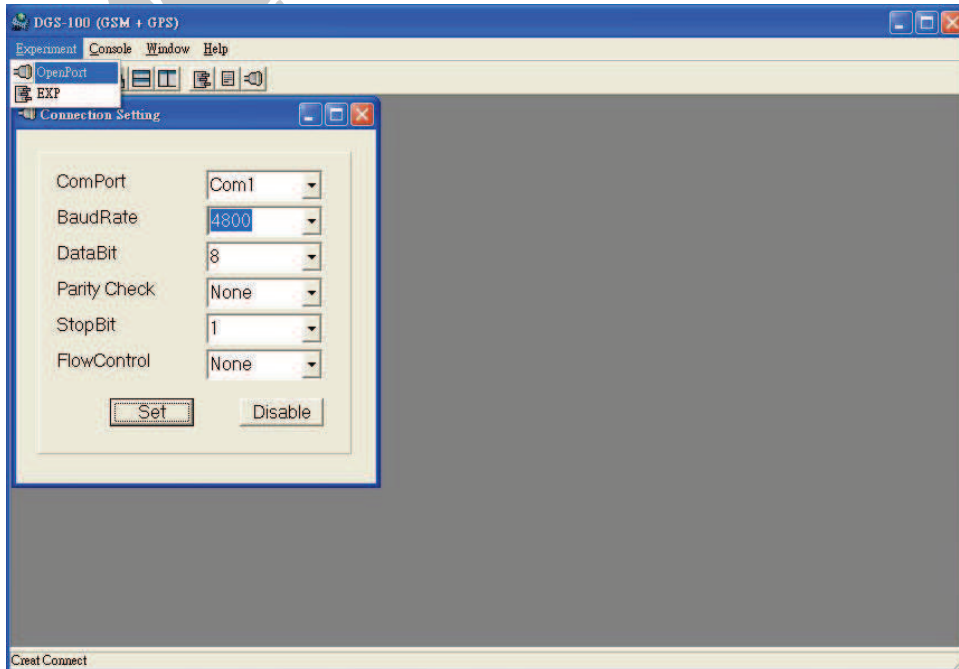


Figure E1-1-2 Connection Setting dialog box

- When connected, select **Experiment -> EXP** command to open the Experiment window. Select **EXP1_1 GPS Module Baud Rate Setting** (default) from the drop-down menu as shown in Figure E1-1-3.

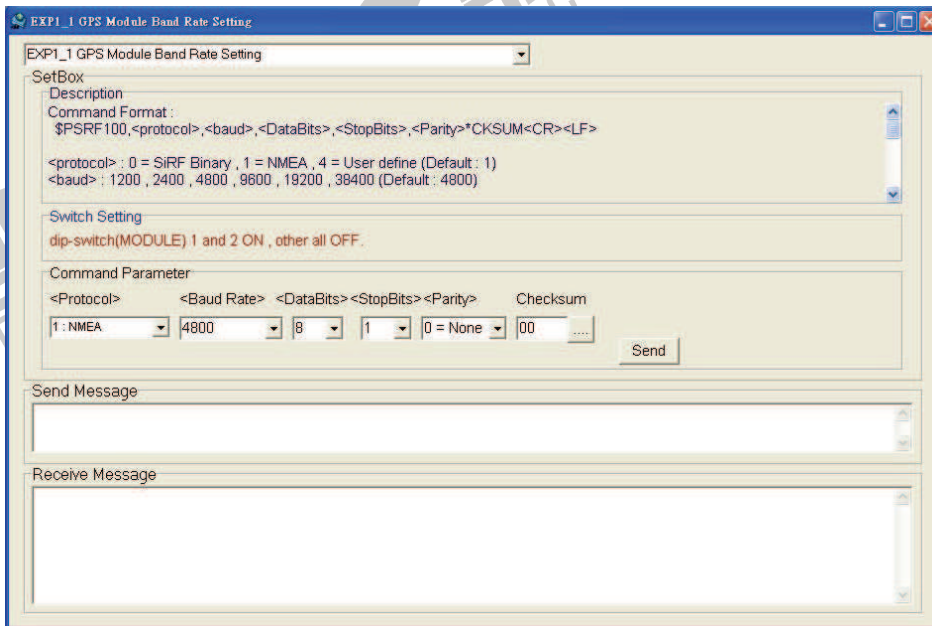


Figure E1-1-3 Experiment window

- In the Receive Message window, you can see the position message received by GPS Module from satellites. See Figure E1-1-4.

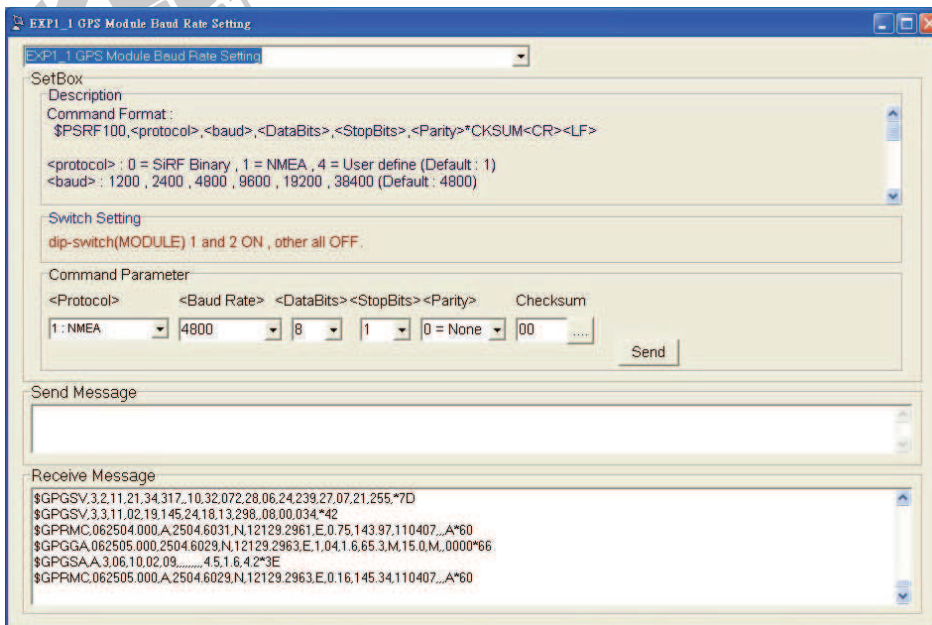



Figure E1-1-4 Position message received by GPS Module

6. If you wish to change the baud rate of GPS Module, modify the numeric parameter <Baud Rate> in the Command Parameter frame; for example, change baud rate from 4800 to 9600. If you have any problems about baud rate setting, refer to the detailed description and examples for command parameters from Help menu.
7. When the parameters are set, click the  button to calculate the checksum and then click **Send**. The output command will be sent to GPS Module and displayed in Send Message window.
8. When the command is sent, you will see that random codes appear in the Receive Message window as shown in Figure E1-1-5. This is because the baud rate of GPS Module was changed and disagreed with the baud rate setting of RS-232 serial port.

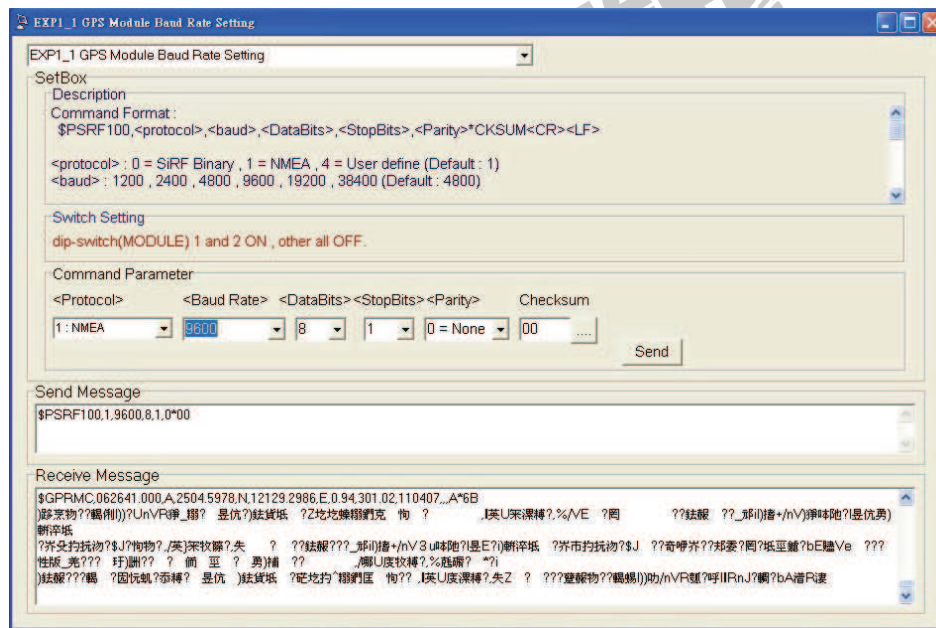


Figure E1-1-5

- To change the baud rate of RS-232 port, select **Experiment -> OpenPort** command to open the Connection Setting dialog box and modify the BaudRate from 4800 to 9600 as shown in Figure E1-1-6.

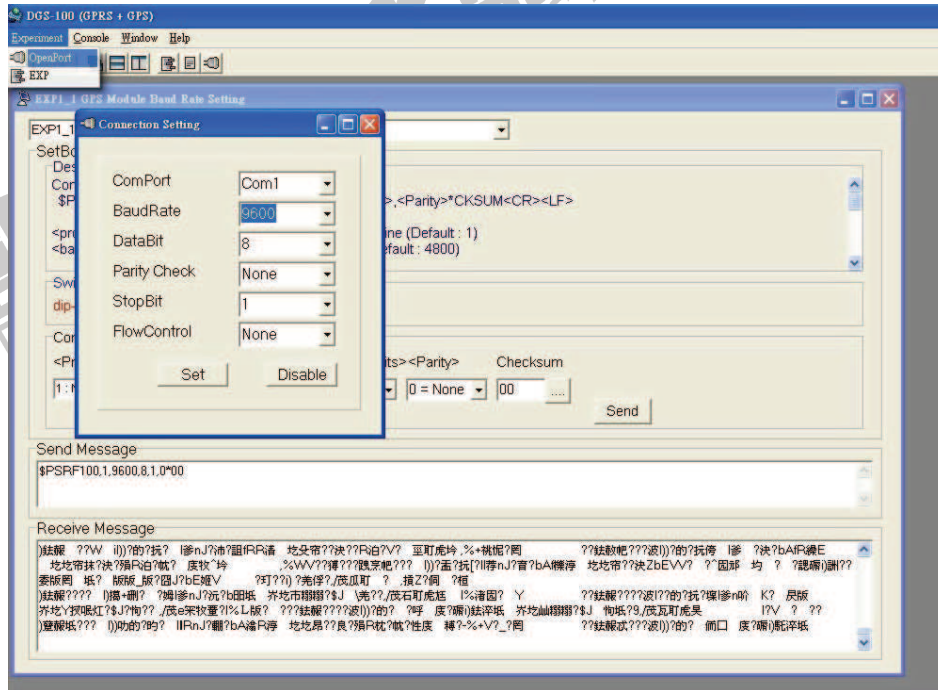


Figure E1-1-6 Changing baud rate

- Return to EXP1_1 GPS Module Baud Rate Setting. If the correct position message is displayed in the Receive Message window, the baud rate has been changed. If it is still random code, check the settings of the RS-232 port. See Figure E1-1-7.

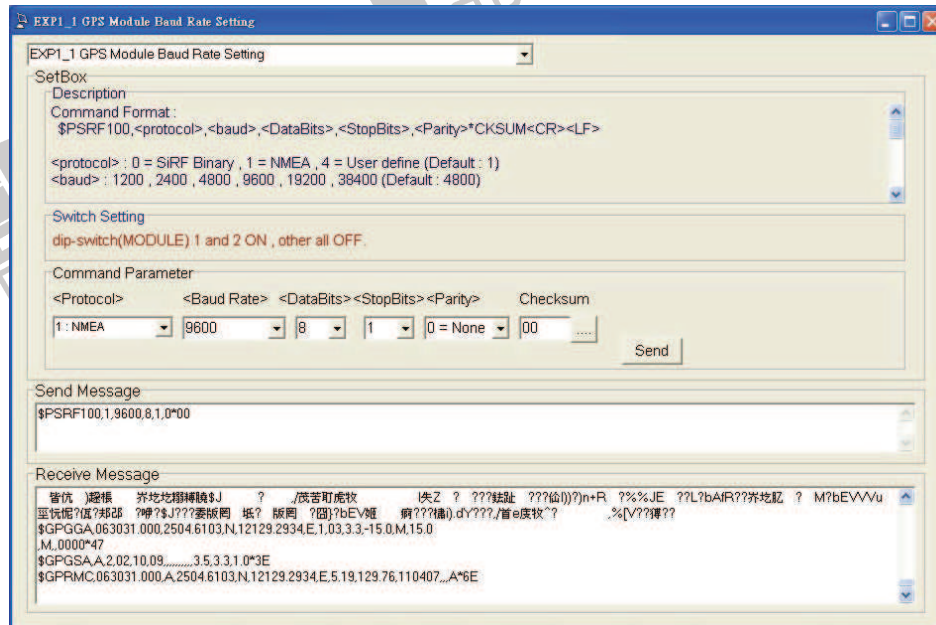


Figure E1-1-7 Correct position message displayed

Notes:

- The baud rate setting of Command Parameter (DGS-100 program) must be same as that in Connection Setting dialog box (GPS Module); otherwise, the application program cannot correctly display the received position message in Receive Message window.
- In this experiment, only the baud rate of COM port needs to be changed, keep other parameters as default.
- The last settings of GPS Module will be saved and applied to the GPS Module when DGS-100 is powered up again.
- Remember that the checksum must be calculated before pressing the Send button.

Exp 9-2 GPS Module Update Rate Control

OBJECTIVE

After completing this experiment, you should be able to enable or disable the display and update rate of position message received by the GPS Module.

DISCUSSION

By factory default, the GPS Module has six output messages: GGA, GLL, GSA, GSV, RMC, and VTG. These messages stand for different formats of longitude and latitude data. You can enable or disable the display and update rate of each message from the graphical user interface.

PROCEDURE

1. Turn on dip-switch(MODULE) 1 and 2, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the RS-232 connector J2 on DGS-100 to the RS-232 port on PC using the RS-232 cable. Turn all power on. Run DGS-100 program and open the DGS-100 main screen as shown in Figure E1-2-1.

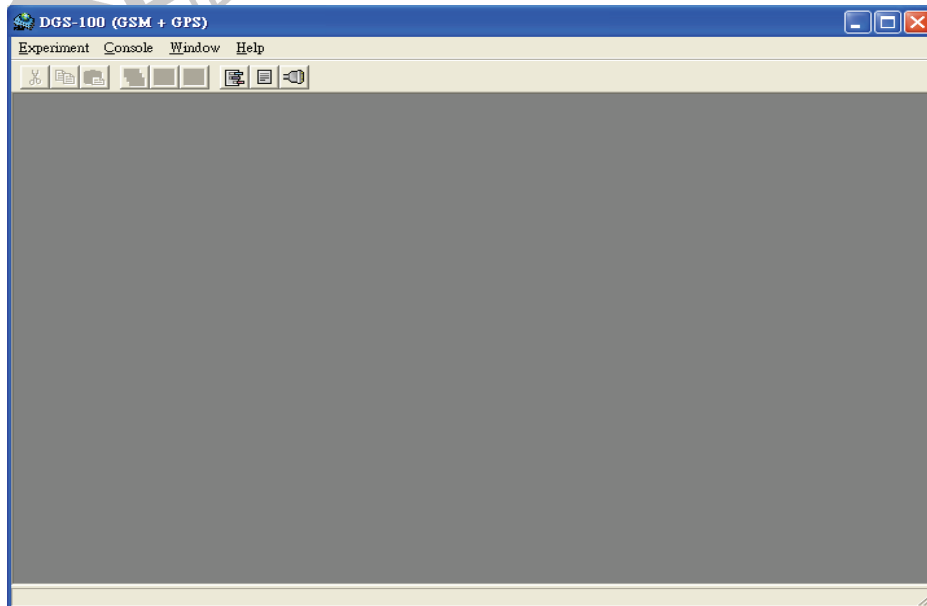


Figure E1-2-1 DGS-100 main screen

3. Select **Experiment -> OpenPort** command to specify the RS-232 serial port. The Connection Setting dialog box opens as shown in Figure E1-2-2. Specify the BaudRate of RS-232 serial port to 4800, and then click **Set** to setup a communication link between DGS-100 GSM/GPS Experimental Set and PC.

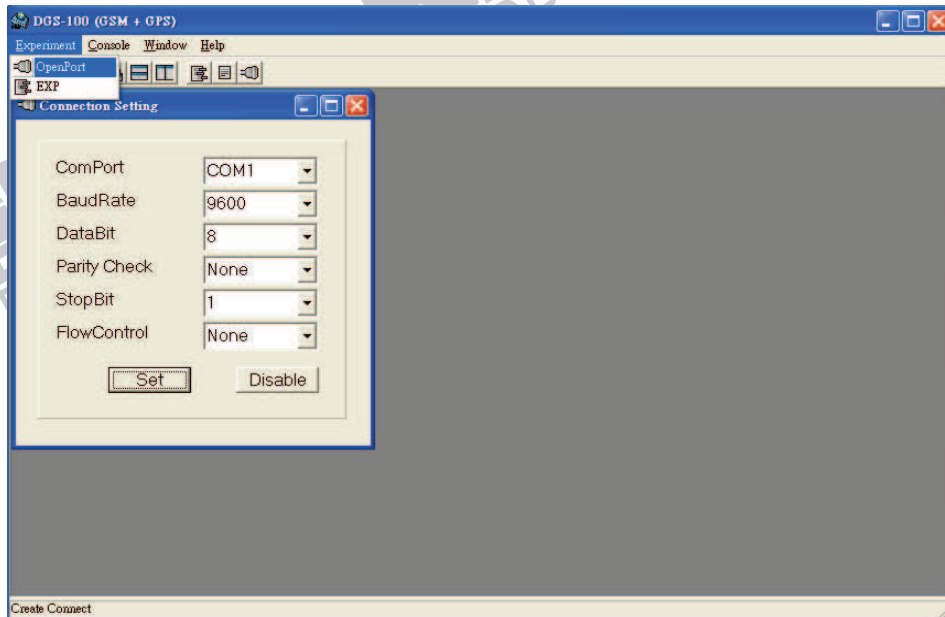


Figure E1-2-2 Connection Setting dialog box

4. When connected, select **Experiment -> EXP** command to open the Experiment window (EXP1_1 GPS Module Baud Rate Setting by default).
5. Select **EXP1_2 GPS Module Update Rate Control** from the drop-down menu as shown in Figure E1-2-3.

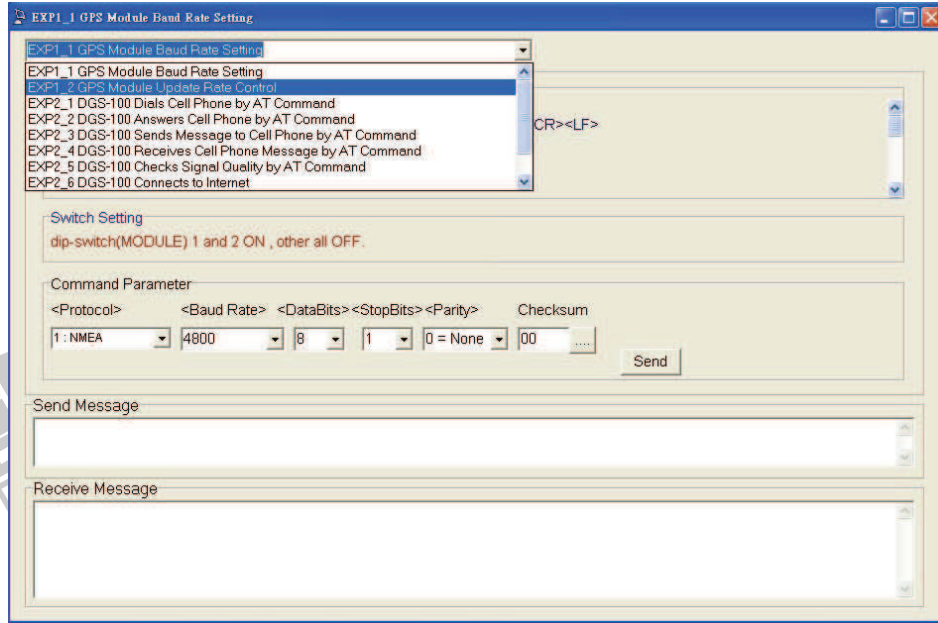


Figure E1-2-3 Experiment window

6. In Receive Message window, you will see the satellite position message received by GPS Module as shown in Figure E1-2-4. If random codes appear in Receive Message window, select **Experiment -> OpenPort** command to open Connection Setting dialog box and set Baud Rate to 4800 or other values until the random codes disappear.

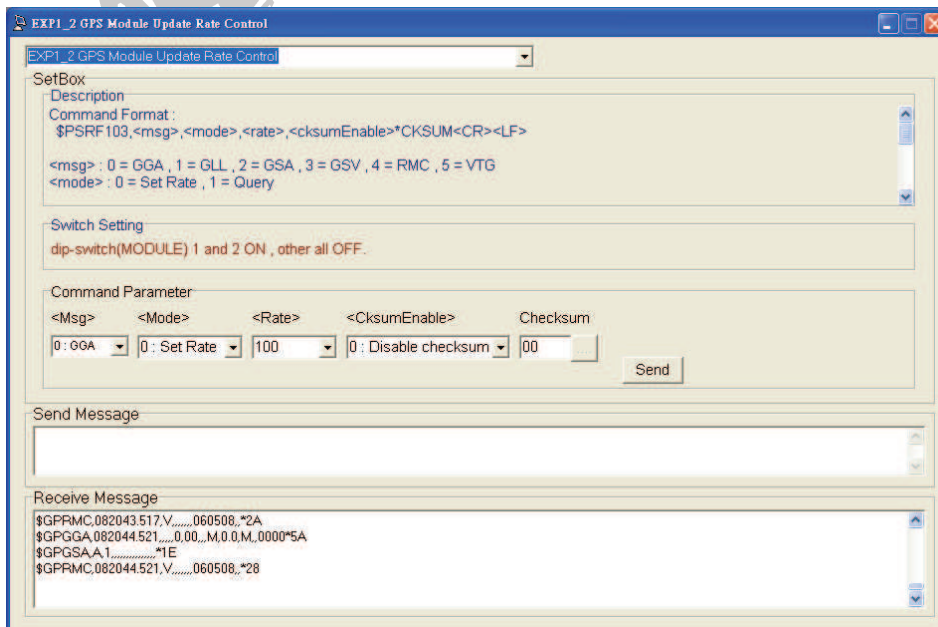



Figure E1-2-4 EXP1_2 window

7. In Command Parameter frame, you can set the update rates of satellite positioning information. To do so, select **0:Set Rate** from <Mode> drop-down menu, and then select a desired update rate in seconds from <Rate> drop-down menu. (For example, select **00** to disable the display; select **100** to update the displayed message at the rate of every 100 seconds). Select **1:Query** from <Mode> drop-down menu to update message whenever the Send button is pressed despite the settings of <Rate>. Select **1:Enable checksum** from the <CksumEnable> drop-down menu and then click on the **Calculate** button , the result will display in the Checksum field. If you have any problems about the command parameters, see the description and examples from Help menu.
8. Once the command parameters set, click **Send** button. This command will be sent to GPS Module and displayed in the Send Message window.
9. You will see the result shown in Receive Message window. See Figure E1-2-5.

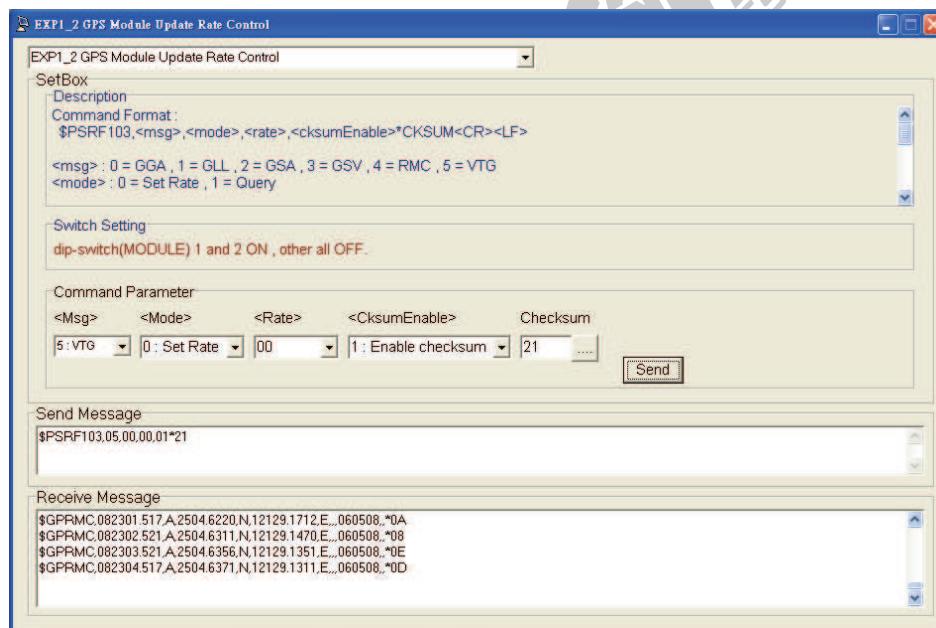


Figure E1-2-5

Notes:

1. Remember that the checksum must be calculated before pressing the Send button.
2. In this experiment only the baud rate of COM port needs to be changed, keep other parameters as default.

Exp 9-3 DGS-100 Dials Cell Phone by AT Command

OBJECTIVE

After completing this experiment, you should be able to dial a cell phone and make conversation using the DGS-100 GSM/GPS Experimental Set.

DISCUSSION

To dial a cell phone by AT commands using the GSM/GPRS Module, the connection mode of GSM/GPRS Module must be in voice mode. Consequently, the current connection mode should be queried first, and then set the connection to voice mode. Once completed, you can begin to make a call. In this case, the GSM/GPRS Module acts as a simplest cell phone.

The DGS-100 GSM/GPS Experimental Set is equipped with two jacks: EAR_HF and MIC_HF. The EAR_HF jack is used to connect an earphone or speaker, and MIC_HF is used to connect a microphone.

PROCEDURE

1. Turn on dip-switch(MODULE) 5 and 6, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the GSM/GPRS antenna to the GSM/GPRS Module as shown in Figure E2-1-1, and do not remove frequently. Stick the antenna on the GSM/GPRS ANTENNA PLACE.



Figure E2-1-1 Connecting GSM/GPRS Module to quad-band antenna

3. Refer to the connections shown in Figure E2-1-2. Connect the earphone (black wire) of the supplied earphone-microphone set to the EAR_HF jack on DGS-100 Experimental Set, and then connect the microphone (red wire) to the Microphone jack on PC. Connect the MIC_HF jack on DGS-100 Experimental Set to the Earphone jack on PC. Now your DGS-100 Experimental Set can operate as a cell phone.

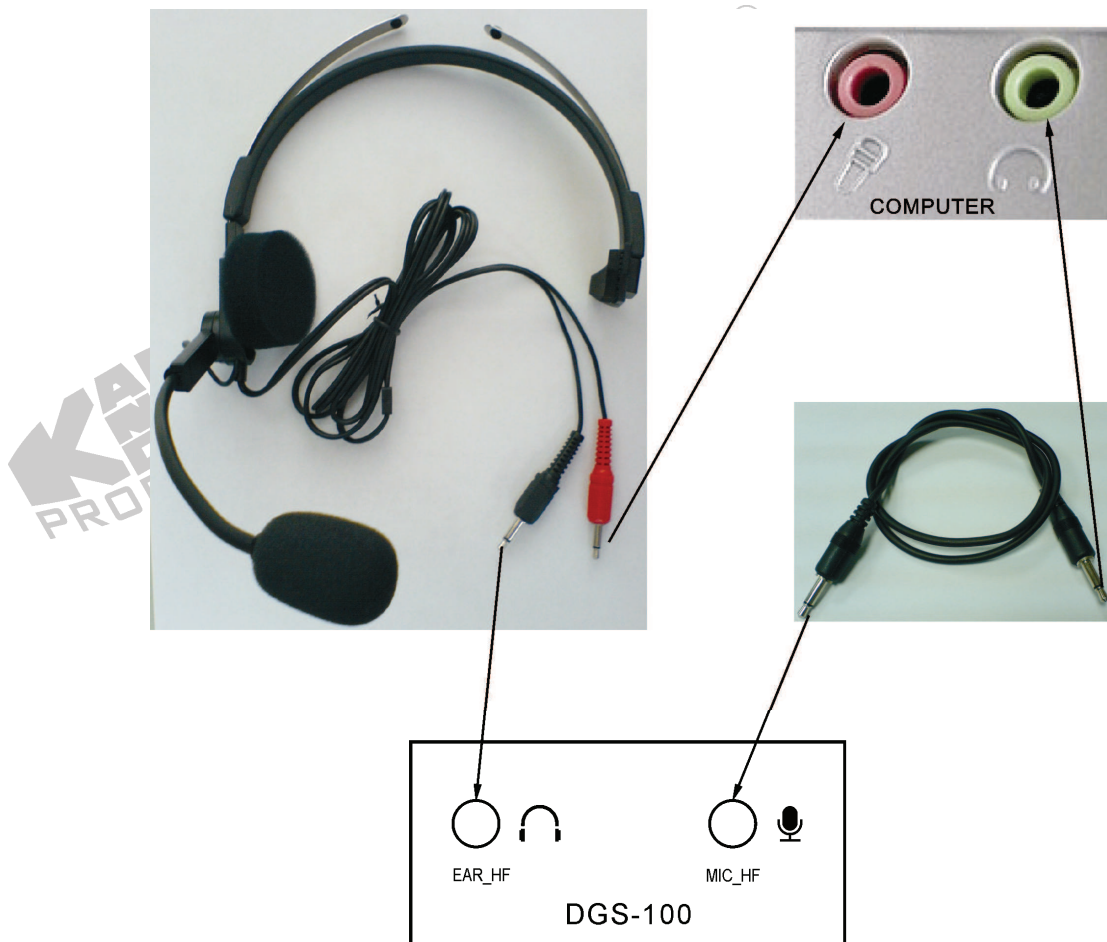


Figure E2-1-2 Microphone and earphone connections

4. Connect the RS-232 connector J2 on DGS-100 to the RS-232 port on PC using the RS-232 cable. Turn all power on. Run DGS-100 program and open the DGS-100 main screen as shown in Figure E2-1-3.

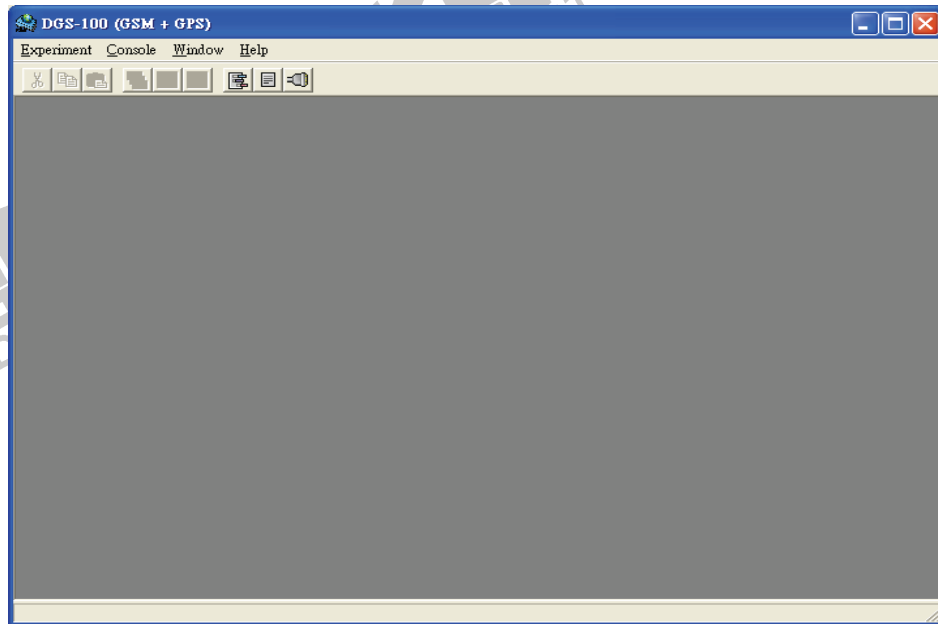


Figure E2-1-3 DGS-100 main screen

5. Select **Experiment** → **OpenPort** command to open the Connection Setting dialog box as shown in Figure E2-1-4. Complete the settings as shown and click **Set**.

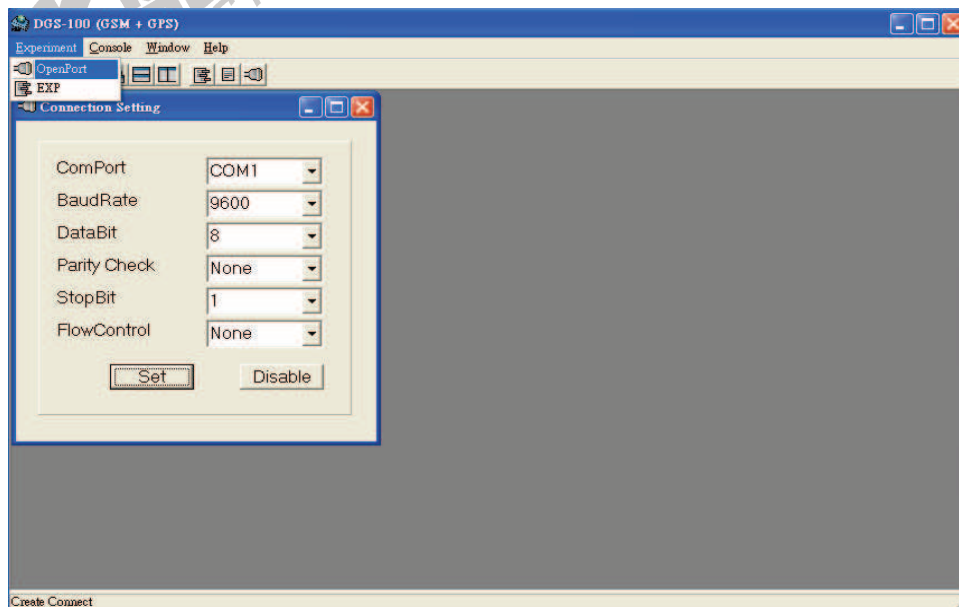


Figure E2-1-4 Connection Setting dialog box

6. When connected, select **Experiment -> EXP** command to open the Experiment window.
7. Select the **EXP2_1 DGS-100 Dials Cell Phone by AT command** from the drop-down menu. See Figure E2-1-5.

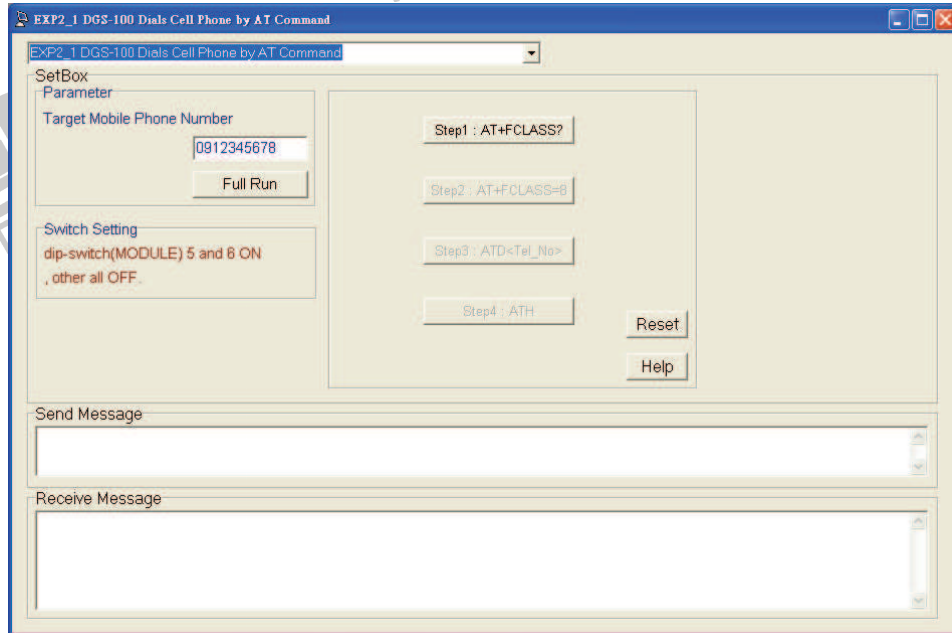


Figure E2-1-5 EXP2_1 window

8. You can now dial a cell phone step by step. If you have any problems about the related settings, click on **Help** button to open the Help window as shown in Figure E2-1-6. Enter the telephone number in Target Mobile Phone Number field (0912345678 by default).

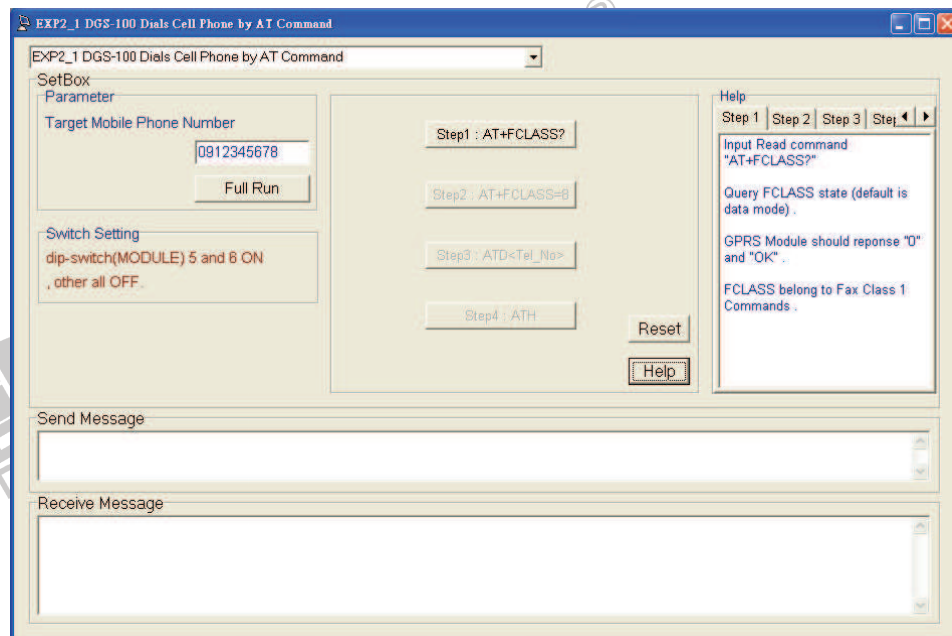


Figure E2-1-6 Help window

9. To query the current connection mode of GSM/GPRS Module, click **Step1:AT+FCLASS?**. In this case, GSM/GPRS Module will reply with "0" to indicate that it is in data mode. See Figure E2-1-7.

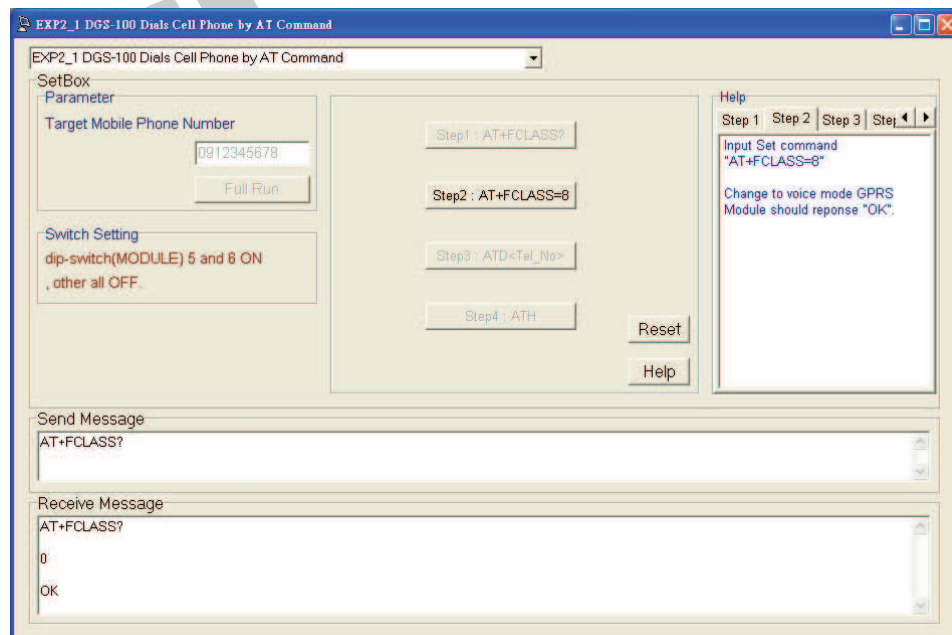


Figure E2-1-7

10. Click **Step2:AT+FCLASS=8** to set the GSM/GPRS Module to voice mode. See Figure E2-1-8.

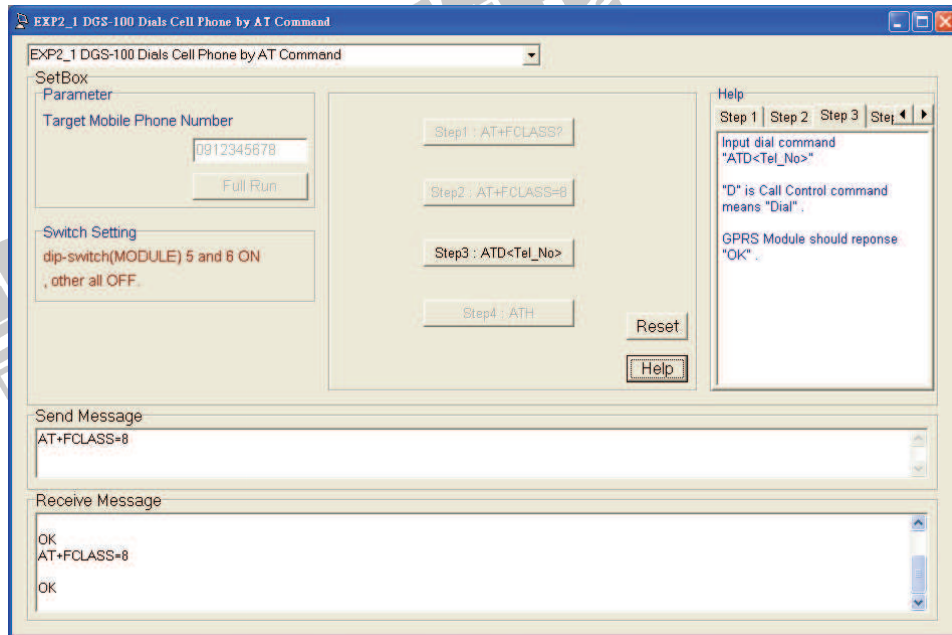


Figure E2-1-8

11. Click **Step3:ATD<Tel_No>** to make a phone call to the phone number given by Target Mobile Phone Number parameter, where <Tel_No> stands for the Target Mobile Phone Number to be dialed.
12. When your partner picks up the phone, you can use earphone and microphone to carry on a conversation with your partner.
13. Click **Step4:ATH** to close the current voice conversation.
14. If you click **Full Run** after a phone number entered, the system will automatically run from Step1 to Step4.

Notes:

1. Make sure the GSM/GPRS Module has a SIM card inserted.
2. Refer to Help window or Appendix A for AT commands.
3. Make sure that the function of microphone and earphone of your computer is activated (Start -> Control Panel -> Sounds and Audio Devices -> Sounds and Audio Devices Properties).

Exp 9-4 DGS-100 Answers Cell Phone by AT Command

OBJECTIVE

After completing this experiment, you should be able to pick up or hang up the phone using AT commands.

DISCUSSION

When the GSM/GPRS Module receives an incoming call, you can control it to pick up or hang up the phone using AT commands. Through the practice in this experiment you will understand how a cell phone responds to an incoming phone call.

PROCEDURE

1. Turn on dip-switch(MODULE) 5 and 6, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the RS-232 connector J2 on DGS-100 to the RS-232 port on PC using the RS-232 cable. Turn all power on. Run DGS-100 program and open the DGS-100 main screen as shown in Figure E2-2-1.

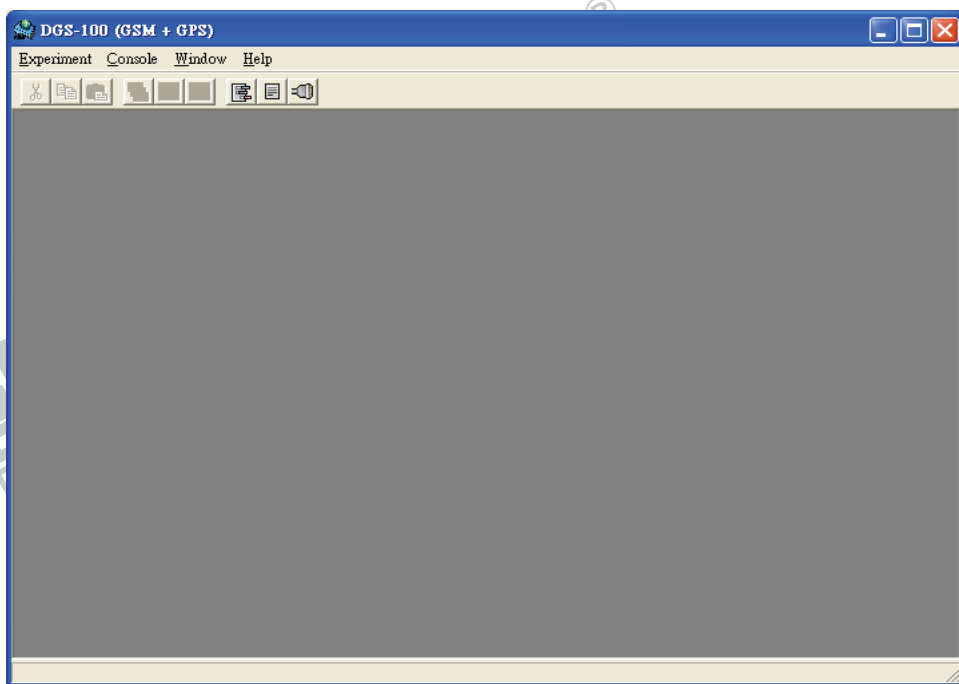


Figure E2-2-1 DGS-100 main screen

3. Select **Experiment -> OpenPort** command to open the Connection Setting dialog box as shown in Figure E2-2-2. Complete the settings as shown and click **Set**.

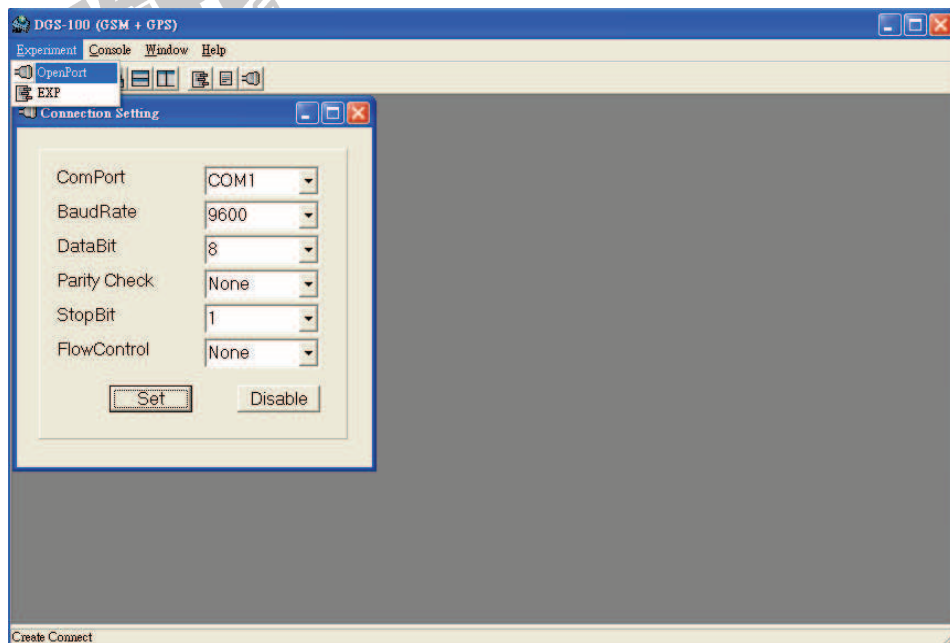


Figure E2-2-2 Connection Setting dialog box

4. When connected, select **Experiment -> EXP** command to open the Experiment window.
5. Select **EXP2_2 DGS-100 Answers Cell Phone by AT Command** from the drop-down menu as shown in Figure E2-2-3.

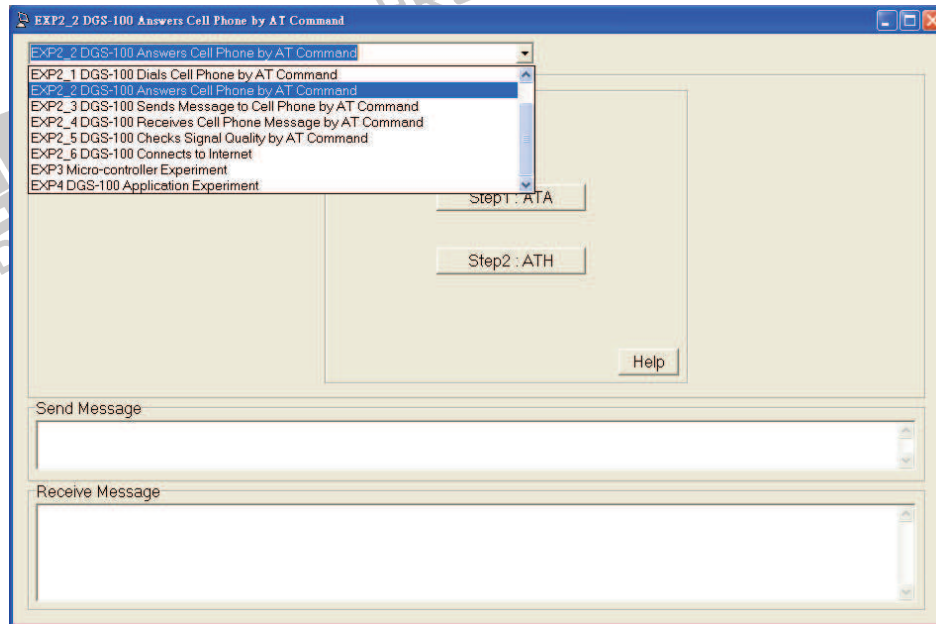


Figure E2-2-3 EXP2_2 window

6. You can now dial DGS-100 (Refer to SIM card for telephone number) by your cell phone. If successful, a "RING" message will be shown in Receive Message window to indicate you have an incoming call. Click **Step1:ATA** button to pick up the phone. A reply of "OK" means that "you are connected." See Figure E2-2-4. If the microphone and earphone connections of Exp 2-1 are complete, a conversation can be made.

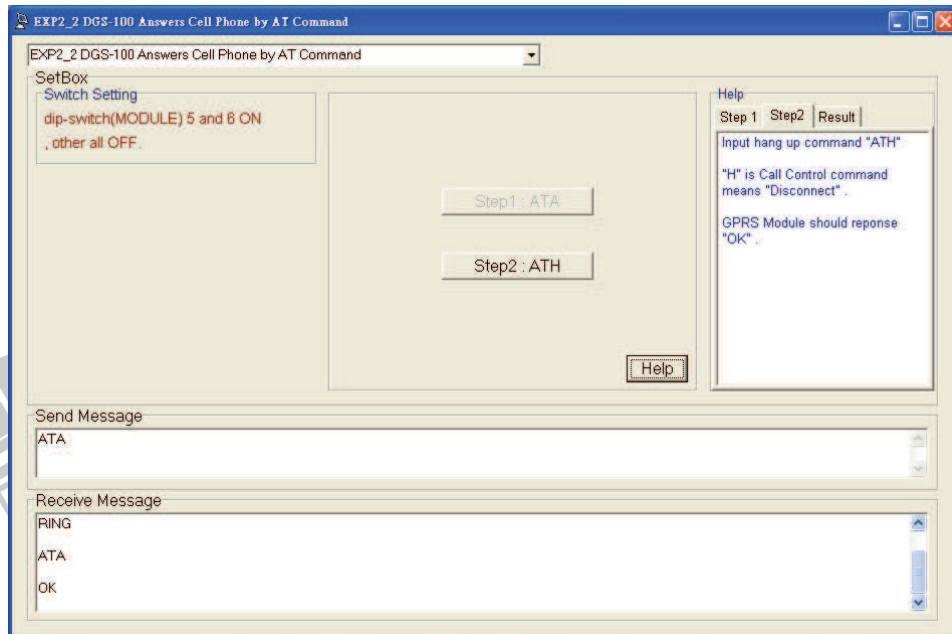


Figure E2-2-4

7. If you wish to hang up the phone or the conversation is over, click **Step2:ATH**. A reply of "OK" in Receive Message window indicates that the conversation is closed. See Figure E2-2-5.

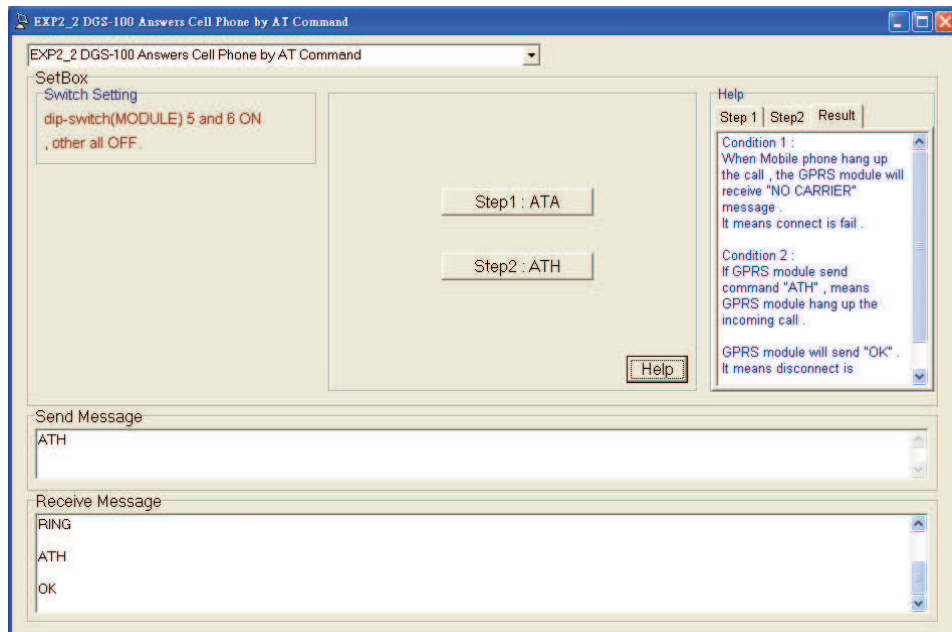


Figure E2-2-5

Notes:

1. Make sure the GSM/GPRS Module has a SIM card inserted.
2. Refer to Help window or Appendix A for AT commands.

Experiment # 10[®]
GSM/GPRS/GPS module, SMS-AT Command

**Exp 10-1 DGS-100 Sends Message to Cell Phone by
AT Command**

OBJECTIVE

After completing this experiment, you should be able to send an SMS message from DGS-100 GSM/GPS Experimental Set to cell phone using AT commands.

DISCUSSION

Short Message Service (SMS) is a standard service of mobile phones. It transfers information through the transmission of text in real time. Since the digital text has the properties of massive, accurate description and easy to store, SMS promotes the efficiency of communication, increases the reliability of message and satisfies the need of mass communication.

To send an SMS message, a SIM card must be inserted in the GSM/GPRS Module.

PROCEDURE

1. Turn on dip-switch(MODULE) 5 and 6, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the RS-232 connector J2 on DGS-100 to the RS-232 port on PC using the RS-232 cable. Turn all power on. Run DGS-100 program and open the DGS-100 main screen as shown in Figure E2-3-1.

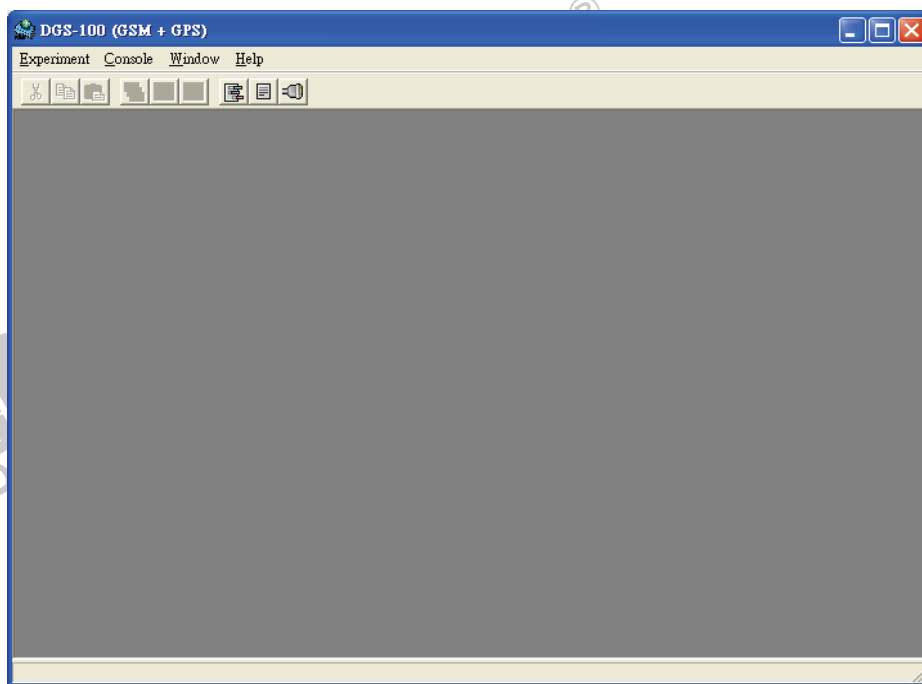


Figure E2-3-1 DGS-100 main screen

3. Select **Experiment -> OpenPort** command to open the Connection Setting dialog box as shown in Figure E2-3-2. Complete the settings as shown and click **Set**.

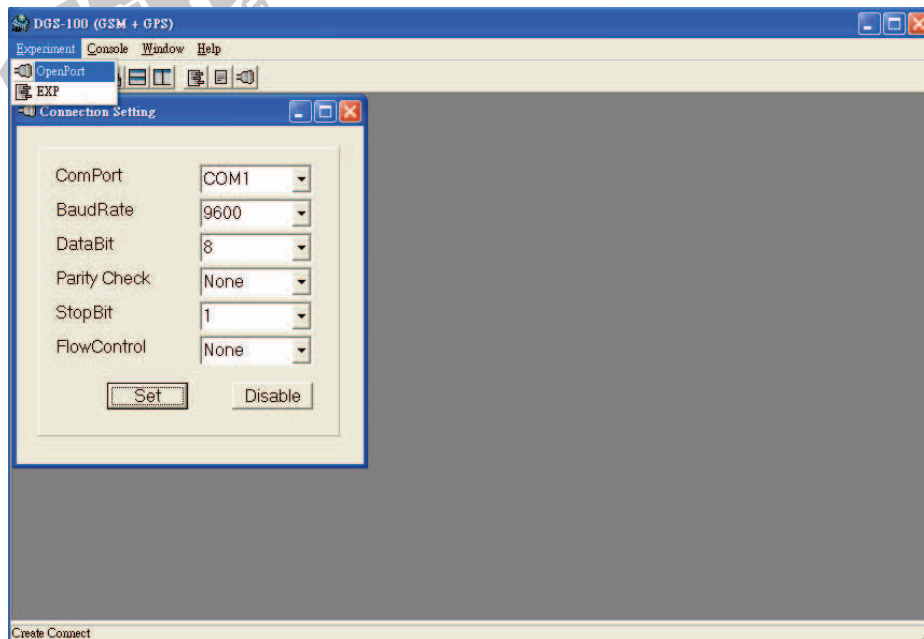


Figure E2-3-2 Connection Setting dialog box

4. When connected, select **Experiment -> EXP** command to open the Experiment window.
5. Select **EXP2_3 DGS-100 Sends Message to Cell Phone by AT Command** from the drop-down menu. See Figure E2-3-3.

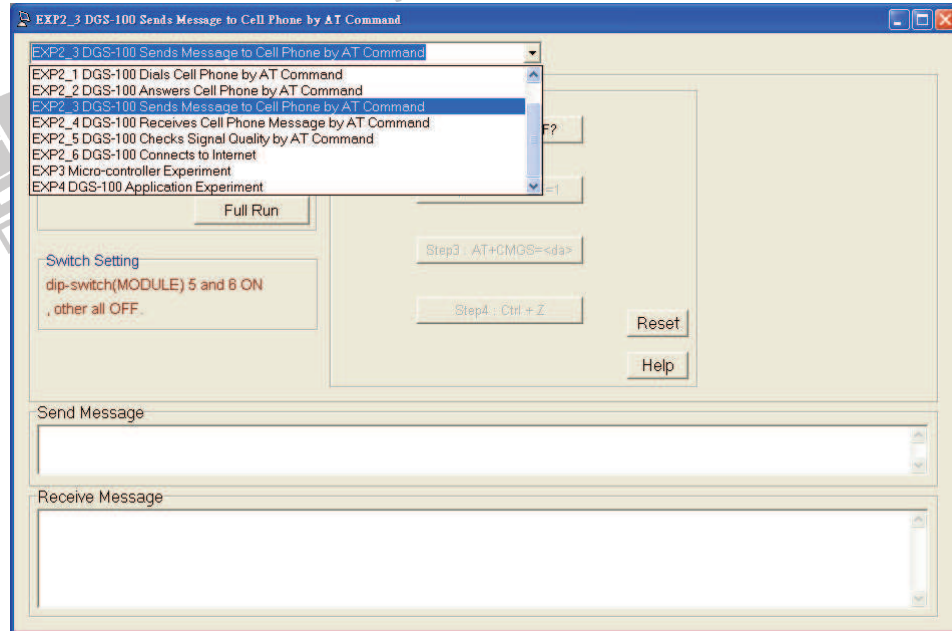


Figure E2-3-3 EXP2_3 window

6. To send SMS step by step, at first enter Target Number and Message Content in Parameter frame, and then click **Step1:AT+CMGF?** to query whether the GSM/GPRS Module is in text mode or not. A reply of "0" indicates the current connection mode is PDU. In this mode, a short message in text cannot be sent. See Figure E2-3-4.

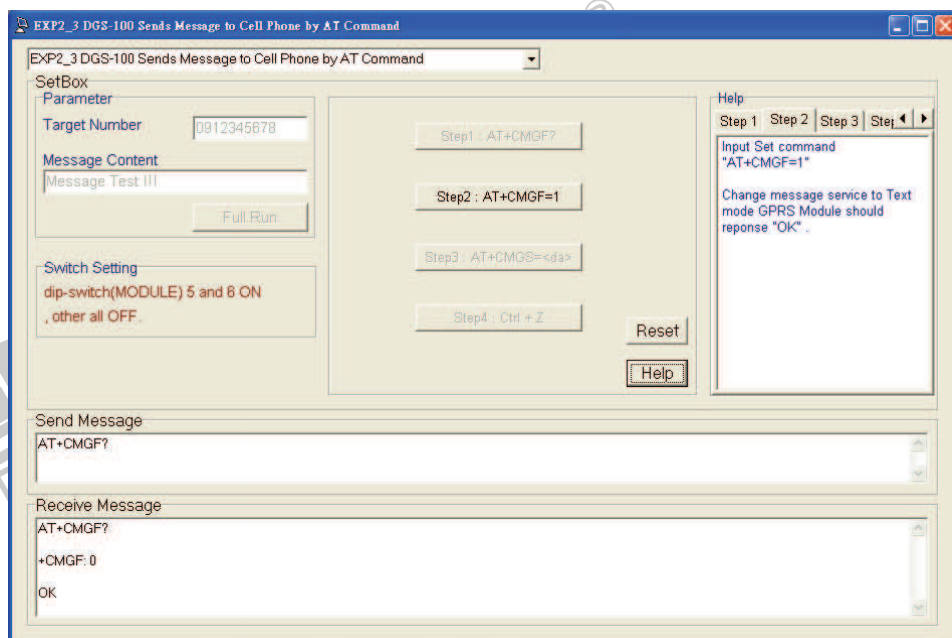


Figure E2-3-4

7. Click **Step2:AT+CMGF=1** to set the connection mode to "1", text mode. See Figure E2-3-5.

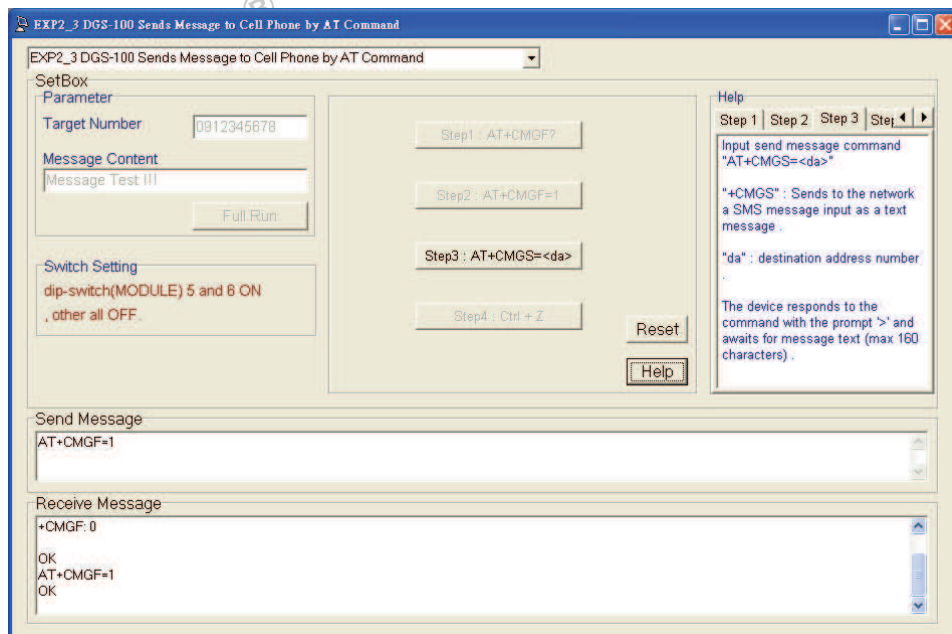


Figure E2-3-5

8. Click **Step3:AT+CMGS<da>** to send an SMS message to the target phone number <da>. The GSM/GPRS Module responds to the command with the prompt ">" and waits for a message text (see Figure E2-3-6). You can enter the text of SMS message up to 160 ASCII characters. Since the target number and message content were entered in Step 6, omit these entries and proceed to the next step.

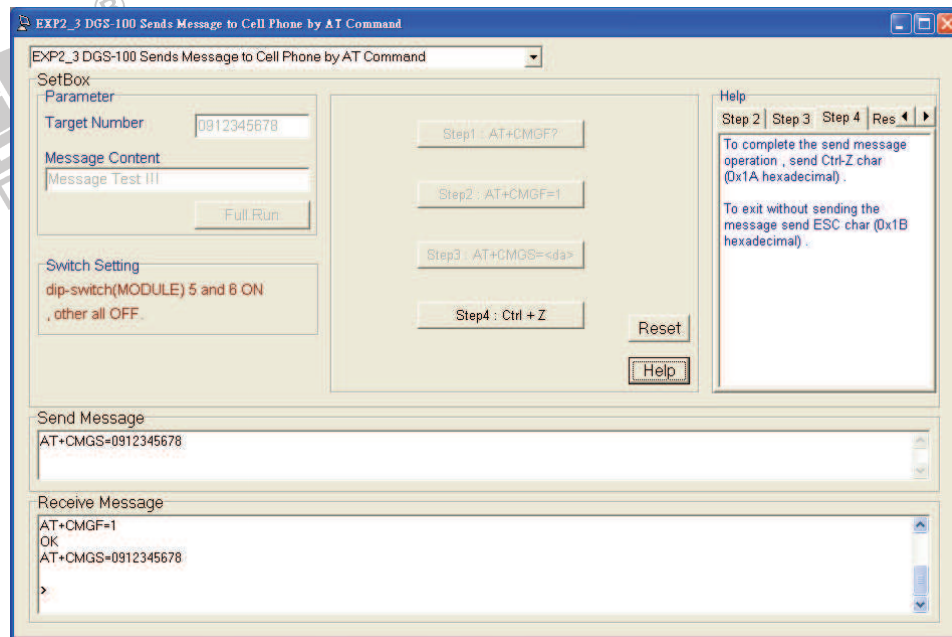


Figure E2-3-6

9. Click **Step4:Ctrl+Z** to complete the operation.

10. GSM/GPRS Module immediately sends the SMS message out. If message is successfully sent to network, then the result is sent in the format: “+CMGS 32”, where 32 is the message reference number in GSM/GPRS Module register and it is not a fixed value. Then GSM/GPRS Module replies with “OK”. See Figure E2-3-7.

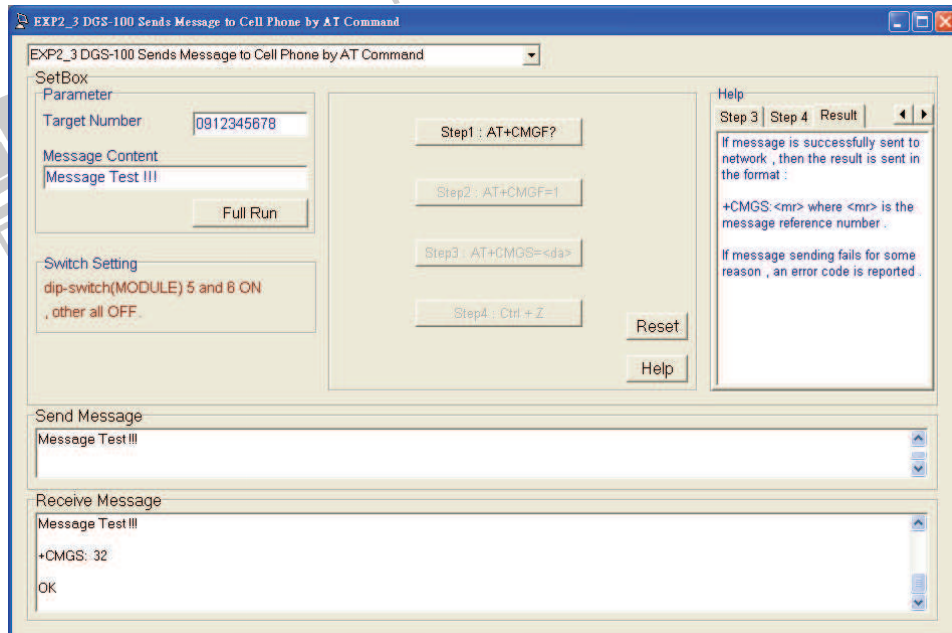


Figure E2-3-7

11. To send an SMS message automatically, enter telephone number in Target Number field and a message text in Message Content field and click **Full Run**.

Notes:

1. Make sure the GSM/GPRS Module has a SIM card inserted.
2. Remember that the **Step4:Ctrl+Z** button (the character string terminator) must be pressed after an SMS message entered.
3. Refer to Help window or Appendix A for AT commands.

Exp 10-2 DGS-100 Receives Cell Phone Message by AT Command

OBJECTIVE

After completing this experiment, you should be able to send an SMS message from cell phone to GSM/GPRS module and read the received SMS using AT commands.

DISCUSSION

When the GSM/GPRS Module receives a new SMS message from cell phone, it will store the received SMS in memory and wait for reading. You can read the message using AT command.

PROCEDURE

1. Turn on dip-switch(MODULE) 5 and 6, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the RS-232 connector J2 on DGS-100 to the RS-232 port on PC using the RS-232 cable. Turn all power on. Run DGS-100 program and open the DGS-100 main screen as shown in Figure E2-4-1

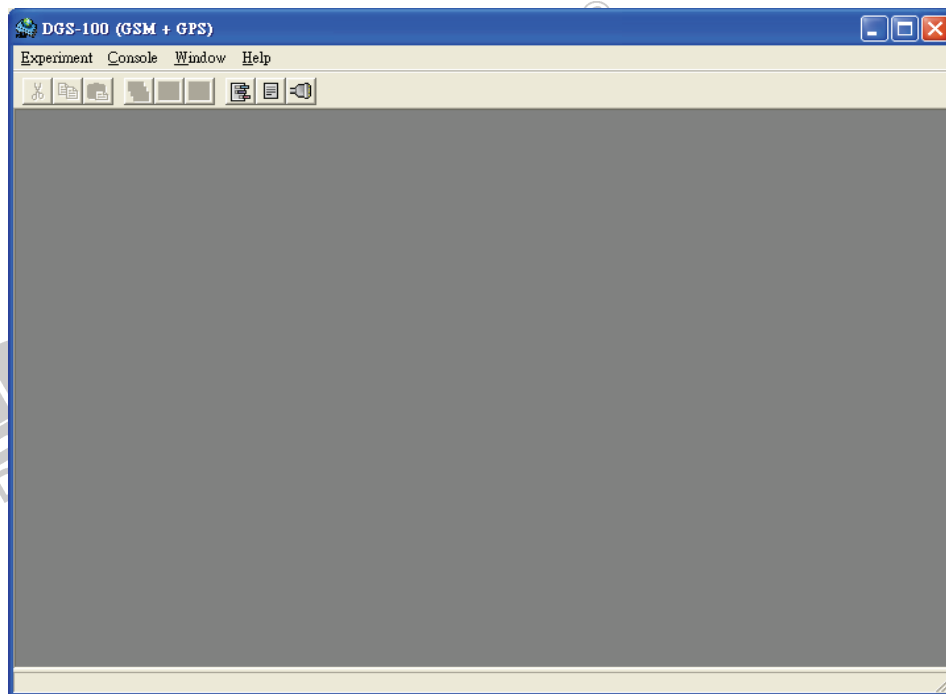


Figure E2-4-1 DGS-100 main screen

3. Select **Experiment -> OpenPort** command to open the Connection Setting dialog box as shown in Figure E2-4-2. Complete the settings as shown and click **Set**.

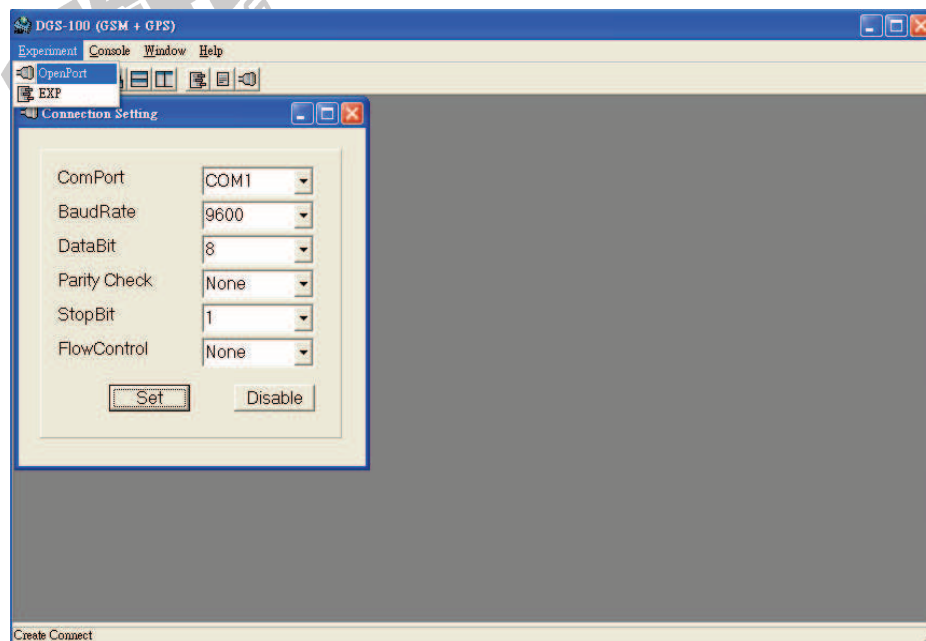


Figure E2-4-2 Connection Setting dialog box

4. When connected, select **Experiment -> EXP** command to open the Experiment window.
5. Select **EXP2_4 DGS-100 Receives Cell Phone Message by AT Command** from the drop-down menu.
6. First send an SMS (e.g., DGS-100) to GSM/GPRS Module using cell phone, and then click **Step1:AT+CMGL="REC UNREAD"** to read new SMS message. The GSM/GPRS Module will respond with: +CMGL: 17, "REC UNREAD", "+886912345678" DGS-100, where "17" represents the received message number in GSM/GPRS Module's register, "REC UNREAD" stands for the new message unread, "+886912345678" is the telephone number that transmits this message, and "DGS-100" is message text. Finally reply with "OK" to terminate the read operation. See Figure E2-4-3.

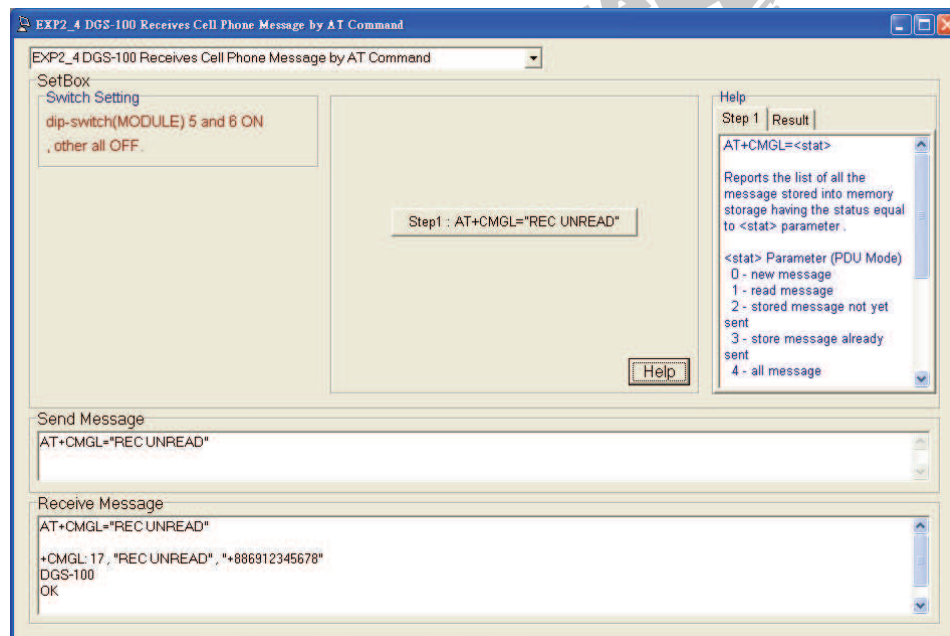


Figure E2-4-3 EXP2_4 window

Notes:

1. If there is no new short message in GSM/GPRS Module, the AT+CMGL="REC UNREAD" command will have no response.
2. Since the GSM/GPRS Module stores the received SMS in SIM card which can store 10 to 20 SMS messages depending on the network provider. If your SIM card is full, the execution of CMGL command will have no response. To delete SMS message using CMGD command, refer to C:\Program Files\DGS-100\GM862_AT_Commands_Guide.pdf file (CMGD command) and execute Console program (Appendix C).
3. Make sure the GSM/GPRS Module has a SIM card inserted.
4. Refer to Help window or Appendix A for AT commands.

Experiment # 11

GSM/GPRS/GPS module, Connecting to the internet

OBJECTIVE

After completing this experiment, you should be able to activate the GPRS web access service and connect to remote host over Internet.

DISCUSSION

The most important function of the GSM/GPRS Module is to connect to Internet using the TCP/IP transmission of GPRS service. This provides higher utilization of the limited bandwidth for data transmission and promotes the transmission speed of the GSM/GPRS Module.

In this experiment, you will learn how to activate the web access service of GPRS using the GSM/GPRS Module and how to connect to the specified remote host on Internet.

PROCEDURE

1. Turn on dip-switch(MODULE) 5 and 6, and turn off other dip-switch from the DGS-100 GSM/GPS Experimental Set.
2. Connect the RS-232 connector on DGS-100 to the RS-232 port on PC using RS-232 cable. Turn all power on. Run DGS-100 program and open the DGS-100 main screen as shown in Figure E2-6-1.

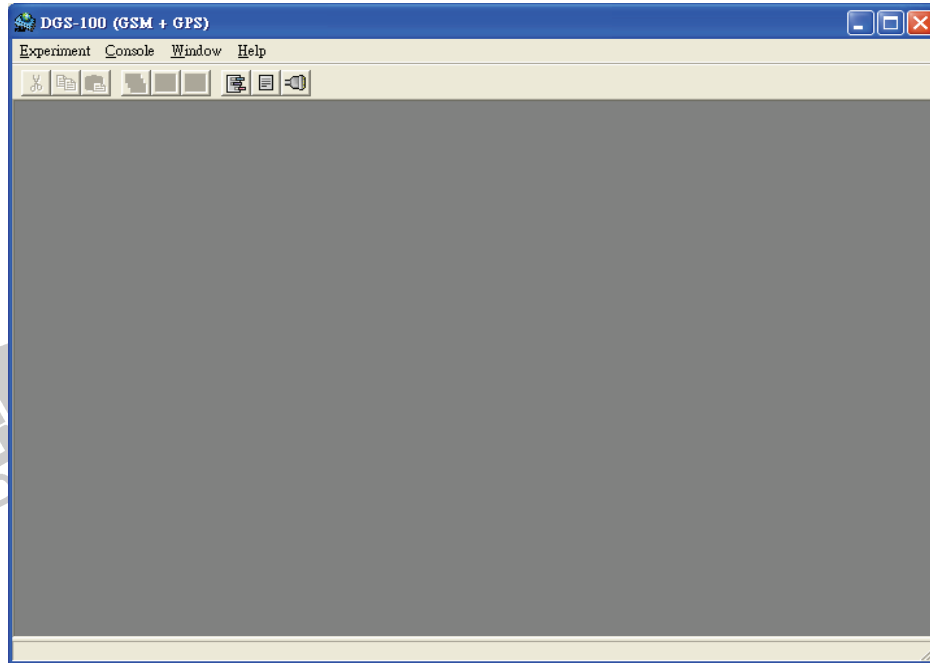


Figure E2-6-1 DGS-100 main screen

3. Select **Experiment -> OpenPort** command to open the Connection Setting dialog box as shown in Figure E2-6-2. Complete the settings as shown and click **Set**.

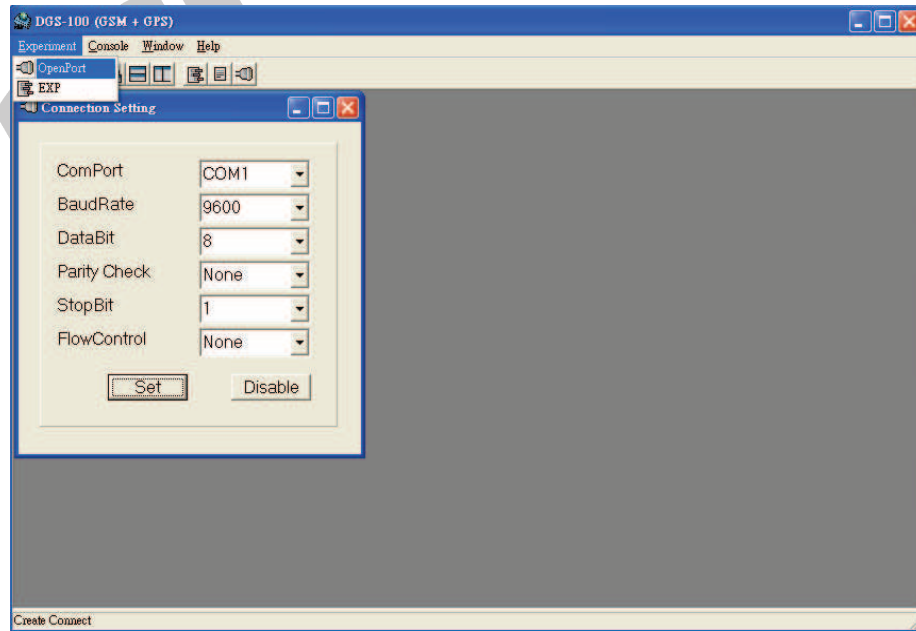


Figure E2-6-2 Connection Setting dialog box

4. When connected, select **Experiment -> EXP** command to open the Experiment window.
5. Select **EXP2_6 DGS-100 Connects to Internet** from the drop-down menu. See Figure E2-6-3.

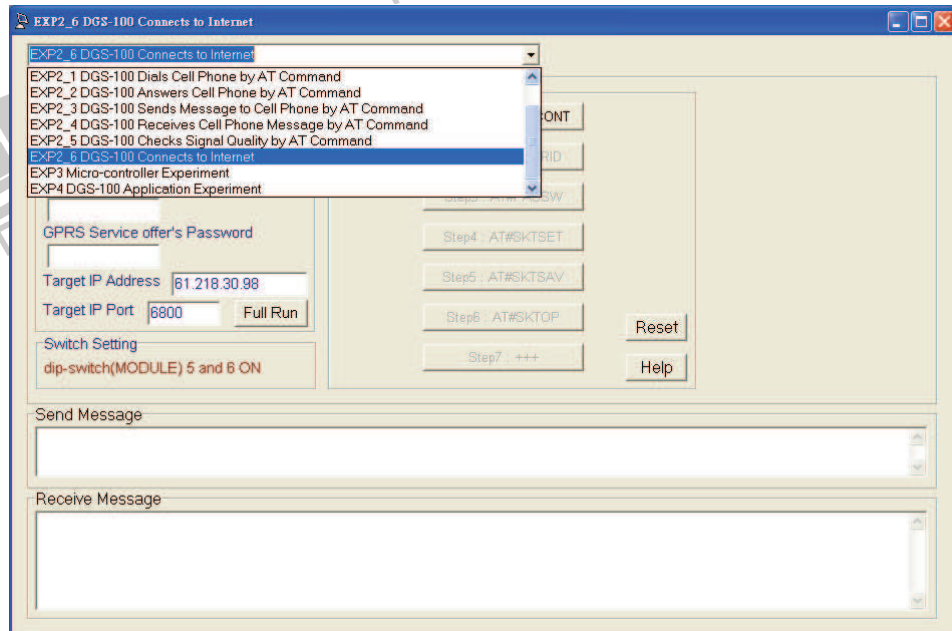


Figure E2-6-3

6. In Configuration Data frame, select the country from the Country drop-down menu. See Figure E2-6-4.

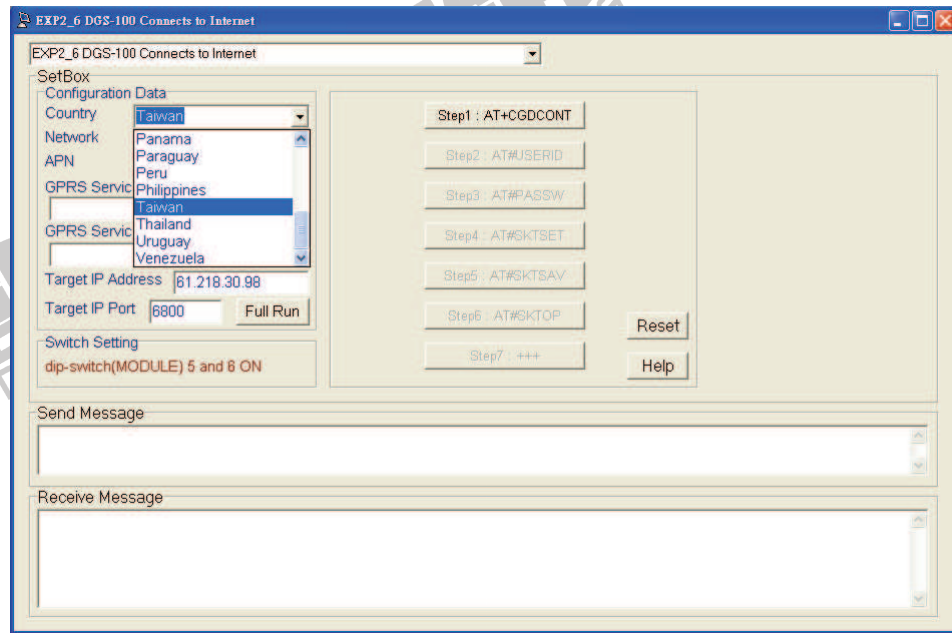


Figure E2-6-4 EXP2_6 window

7. In Configuration Data frame, select the network provider from the Network drop-down menu. See Figure E2-6-5.

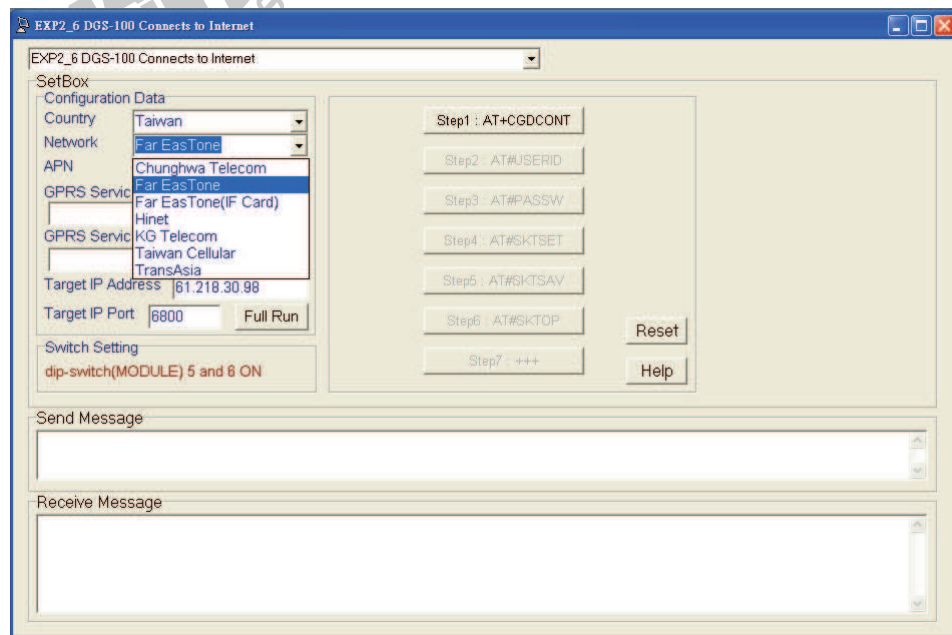


Figure E2-6-5

8. Once Country and Network are selected, the program will automatically generate APN (Access Point Name), User ID (blank if unnecessary), and Password (blank if unnecessary). Enter Target IP Address and Target IP Port of the remote host to be connected. See Figure E2-6-6.

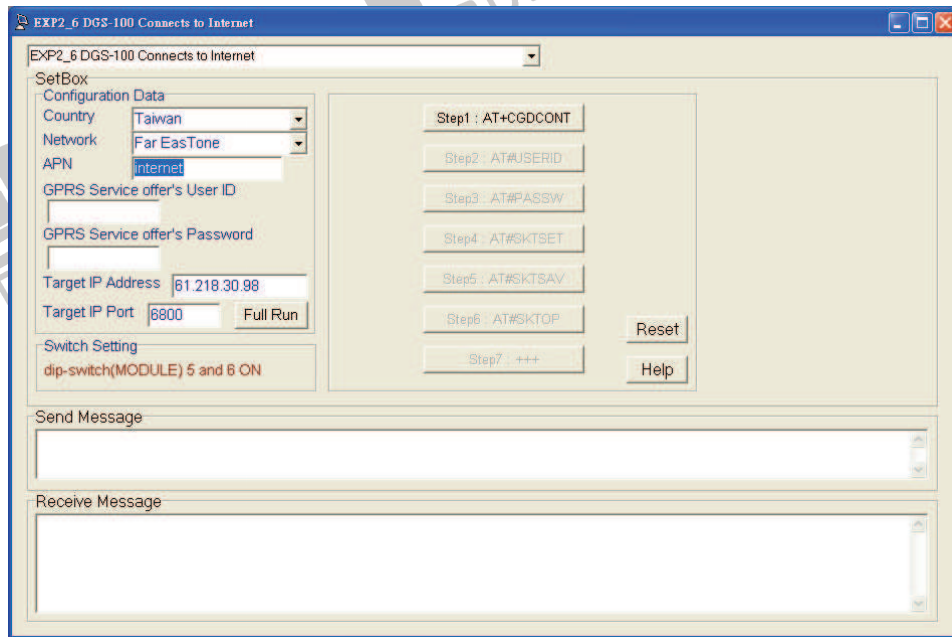


Figure E2-6-6

9. To understand how the GSM/GPRS Module internetworks with the Internet and connects to remote host step by step. Click **Step1:AT+CGDCONT** to send APN to GSM/GPRS Module. See Figure E2-6-7.

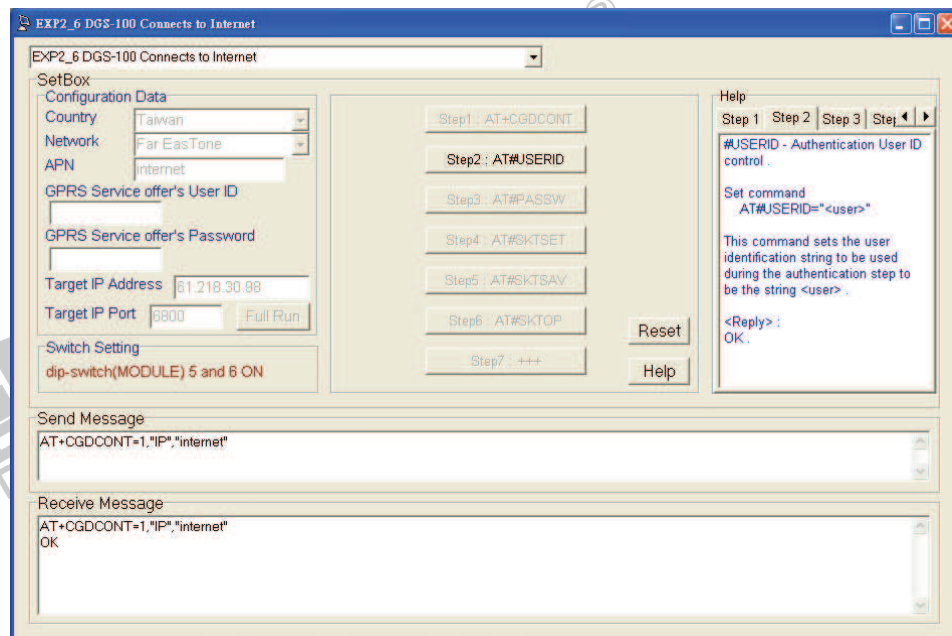


Figure E2-6-7

10. Click **Step2:AT#USERID** to enter User ID (offered by GPRS service provider or blank). See Figure E2-6-8.

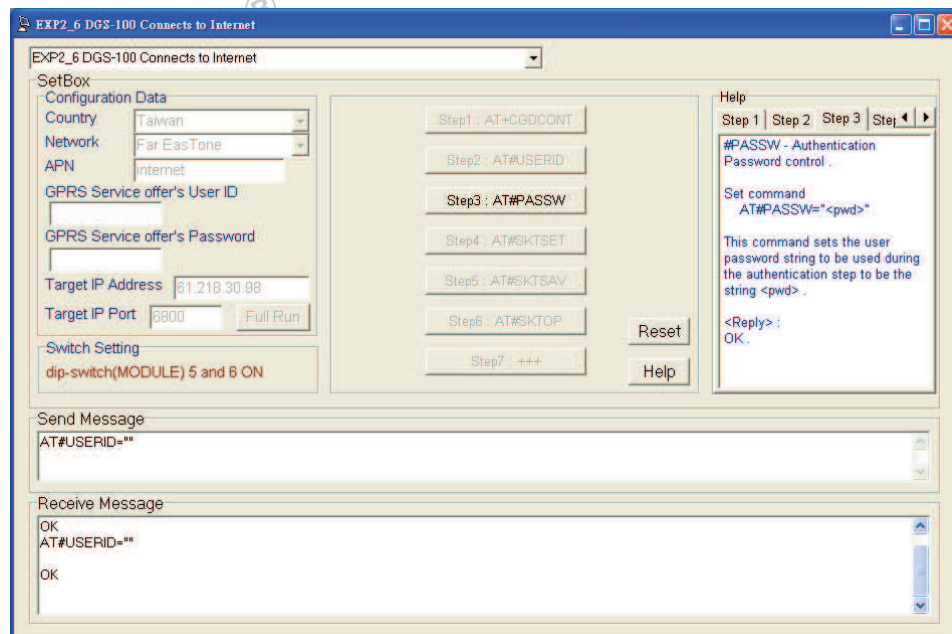


Figure E2-6-8

11. Click **Step3:AT#PASSW** to enter Password (offered by network provider or blank). See Figure E2-6-9.

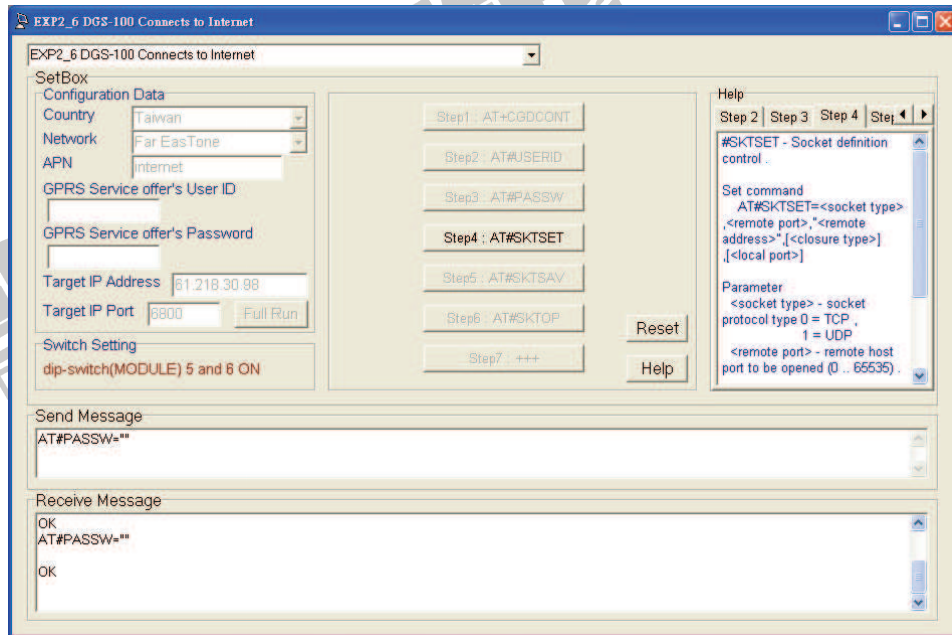


Figure E2-6-9

12. Click **Step4:AT#SKTSET** to enter Target IP Address and Target IP Port of remote host. See Figure E2-6-10.

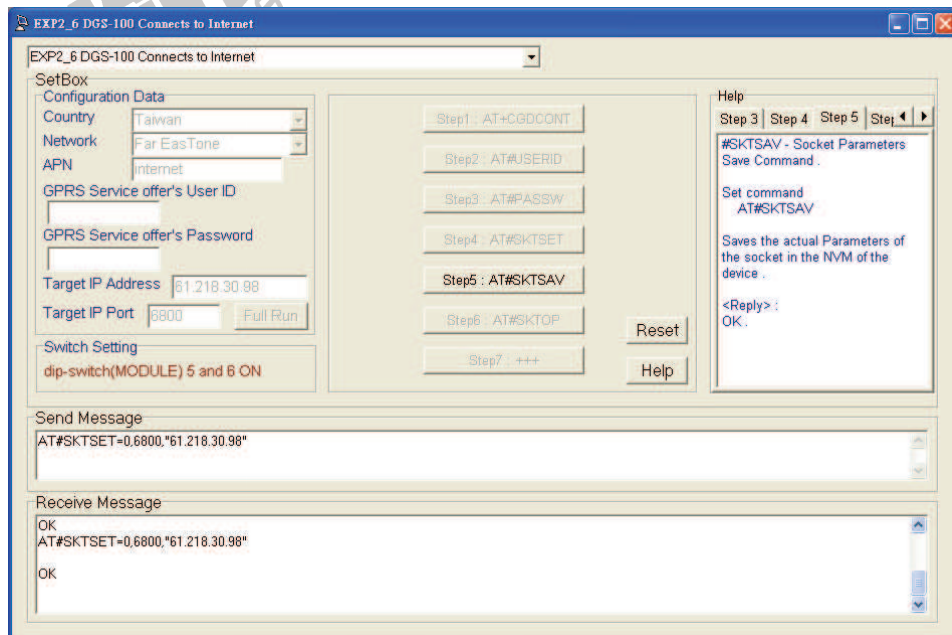


Figure E2-6-10

13. Click **Step5:AT#SKTSAV** to save the settings. See Figure E2-6-11.

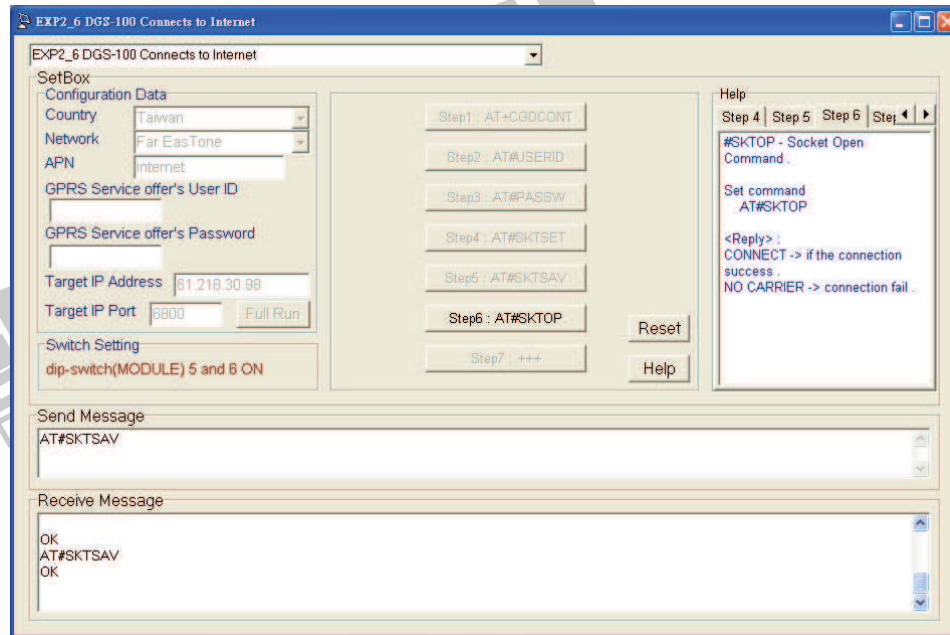


Figure E2-6-11

14. Click **Step6:AT#SKTOP** to start the connection. Reply with “CONNECT” to indicate the connection successful (see Figure E2-6-12). If a “NO CARRIER” is replied (the connection failure), check your SIM card for a deactivated GPRS service, incorrect Target IP Address and Target IP Port.

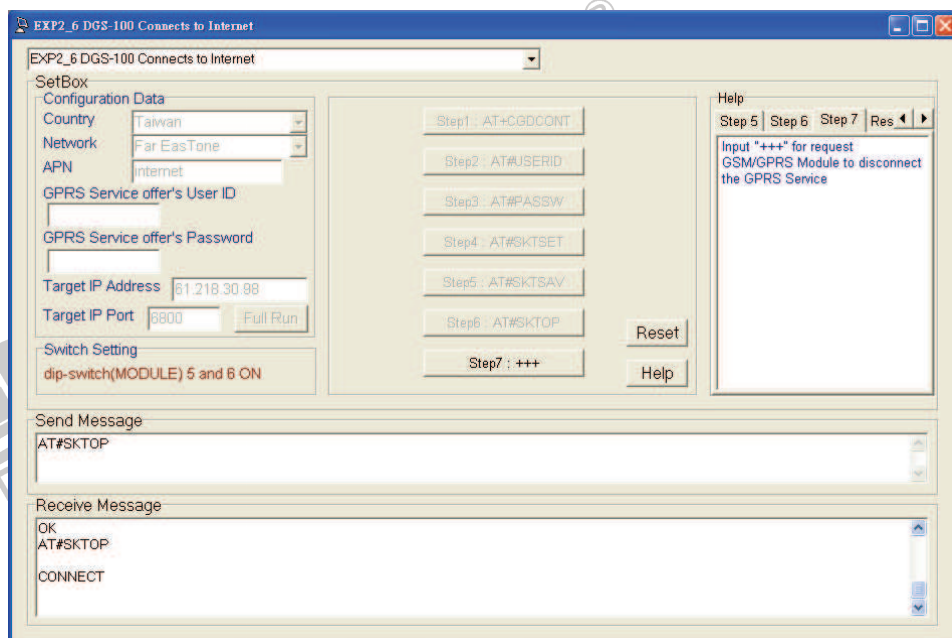


Figure E2-6-12

15. For the convenience of the following experiments, when the connection succeeds, click **Step7:+++** to terminate GPRS web access service. Reply with "NO CARRIER" to indicate that GPRS network service is terminated (see Figure E2-6-13).

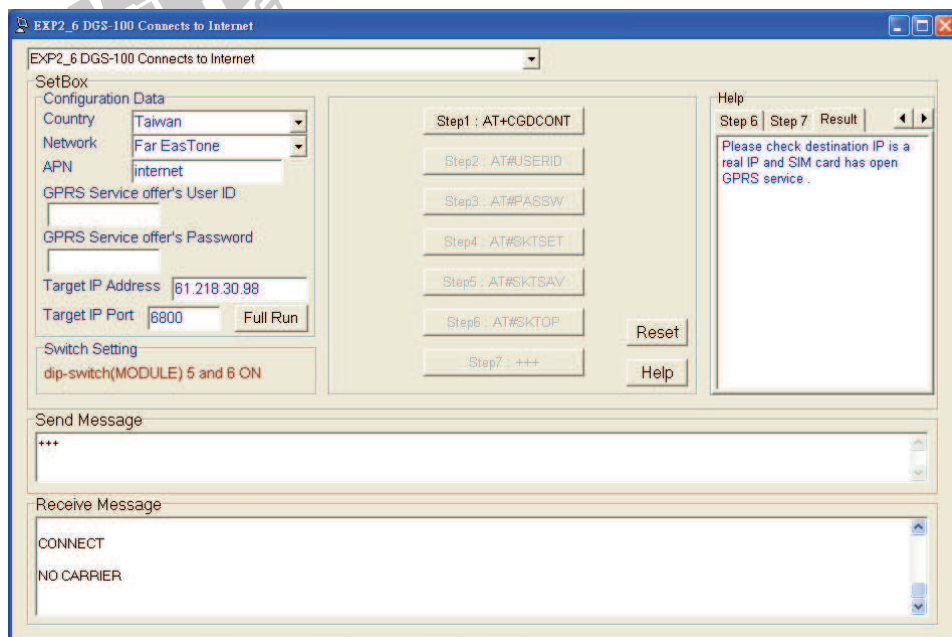


Figure E2-6-13

16. Once the parameters Country, Network, GPRS Service offer's USER ID, GPRS Service offer's Password, Target IP Address and Target IP Port are set, directly click **Full Run**. The GSM/GPRS module will automatically activate the GPRS web access service and connect to target IP.

Notes:

1. Make sure the GSM/GPRS Module has a SIM card inserted.
2. Make sure the GPRS service has been activated.
3. User ID and Password are provided by the SIM card providers.
4. Target IP must be a physical IP or a virtual IP which is mapping to a physical IP.
5. Refer to Help window or Appendix A for AT commands.
6. Please contact with your telecommunication company and ask for APN, GPRS Service offer's User ID and GPRS Service offer's Password if your Country or Network name is not listed in drop-down menu, and enter these information to the relative box manually.

Experiment # 12

Wifi Module Programming

12.1 Introduction

The RCM5400W RabbitCore modules use the Wi-Fi/802.11b/g functionality of the Rabbit[®] 5000 microprocessor to allow you to create a low-cost, low-power, embedded wireless control and communications solution for your embedded control system. The Rabbit[®] 5000 microprocessor features include hardware DMA, clock speeds of up to 100 MHz, I/O lines shared with up to six serial ports and four levels of alternate pin functions that include variable-phase PWM, auxiliary I/O, quadrature decoder, and input capture. Coupled with the existing opcode instructions that help to reduce code size and improve processing speed, this equates to a core module that is fast, efficient, and the ideal solution for a wide range of wireless embedded applications.

Wi-Fi, a popular name for 802.11b/g, refers to the underlying technology for wireless local area networks (WLAN) based on the IEEE 802.11 suite of specifications conforming to standards defined by IEEE. IEEE 802.11b describes the media access and link layer control for a 2.4 GHz implementation, which can communicate at a top bit-rate of 11 Mbits/s. Other standards describe a faster implementation (54 Mbits/s) in the 2.4 GHz band (802.11g). The adoption of 802.11 has been fast because it's easy to use and the performance is comparable to wire-based LANs. Things look pretty much like a wireless LAN.

The Wi-Fi channels have a certain amount of overlap with each other. The further apart two channel numbers are, the less the likelihood of interference. If you encounter interference with a neighbouring WLAN, change to a different channel. For example, use channels 1, 6, and 11 to minimize any overlap.

12.2 RCM5400W module

The Development Kit has the essentials that you need to design your own wireless microprocessor-based system, and includes a complete Dynamic C software development system. This Development Kit also contains a Prototyping Board that will allow you to evaluate the RCM5400W RabbitCore modules and to prototype circuits that interface to the RCM5400W modules. You will also be able to write and test software for these modules.

Program code is stored in parallel flash and is loaded into a fast SRAM for execution when power is applied to the RCM5400W modules. Serial flash and low-power SRAM memories are available for data storage. The data interface between the processor MAC and the AL2236 transceiver consists of a D/A converter and an A/D converter. Both converters convert “I” and “Q” data samples at a rate of 40 MHz.

The RCM5400W is automatically in Program Mode when the PROG connector on the programming cable is attached, and is automatically in Run Mode when no programming cable is attached. When the Rabbit 5000 is reset, the operating mode is determined by the status of the SMODE pins. When the programming cable's PROG connector is attached, the SMODE pins are pulled high, placing the Rabbit 5000 in the Program Mode. When the programming cable's PROG connector is not attached, the SMODE pins are pulled low, causing the Rabbit 5000 to operate in the Run Mode.

There are two LEDs close to the RP-SMA antenna connector at P2, a green LED at DS2 (LINK) to indicate association with the Wi-Fi access point, and a yellow LED at DS1 (ACT) to indicate activity.

Once the RCM5400W has been programmed successfully, remove the programming cable from the programming connector and reset the RCM5400W. The RCM5400W may be reset by cycling the power off/on or by pressing the RESET button on the Prototyping Board.

Note: we will use Dynamic C program to program the Wifi module.

12.3 Experiment Procedure

12.3.1 Connecting the Prototyping Board to Dynamic C

There are three steps to connecting the Prototyping Board for use with Dynamic C and the sample programs:

1. Prepare the Prototyping Board for Development.
2. Attach the antenna to the RCM5400W module.
3. Attach the RCM5400W module to the Prototyping Board.
4. Connect the programming cable between the RCM5400W and the PC.
5. Connect the power supply to the Prototyping Board

The programming cable connects the module to the PC running Dynamic C to download programs and to monitor the module during debugging. Connect the 10-pin connector of the programming cable labelled PROG to header J2 on the RCM5400W. Be sure to orient the marked (usually red) edge of the cable towards pin1 of the connector. (Do not use the DIAG connector, which is used for a normal serial connection).

Start Dynamic C by double-clicking on the Dynamic C icon on your desktop or in your Start menu. Select Store Program in Flash on the "Compiler" tab in the Dynamic C Options > Project Options menu. Then click on the "Communications" tab and verify that Use USB to Serial Converter is selected to support the USB programming cable. Click OK. You may have to select the COM port assigned to the USB programming cable on your PC. In Dynamic C, select Options > Project Options, then select this COM port on the "Communications" tab, and then click OK. You may type the COM port number followed by Enter on your computer keyboard if the COM port number is outside the range on the dropdown menu.

12.3.2 WifiScan program

Objective:

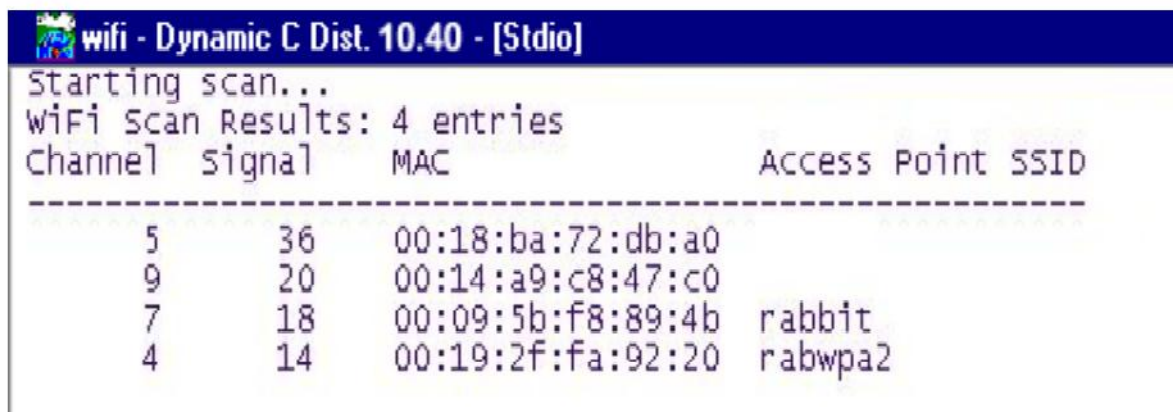
1. Scans and displays existing Wi-Fi networks.

Apparatus and components:

1. Wi-Fi module
2. Existing Wi-Fi network.

Wi-Fi Scan program initializes the RCM5400W and scans for other Wi-Fi devices that are operating in either the ad-hoc mode or through access points in the infrastructure mode. No network parameter settings are needed since the RCM5400W does not actually join an 802.11 network. This program outputs the results of the scan to the Dynamic C STDIO window.

Now find the *WIFISCAN.C* sample program in the Dynamic C Samples\WiFi folder, open it with the File menu, then compile and run the sample program by pressing F9. The Dynamic C STDIO window will display Starting scan...., and will display a list of access points/ad-hoc hosts as shown here.



The screenshot shows a window titled "wifi - Dynamic C Dist. 10.40 - [Stdio]". The text inside the window reads: "Starting scan...", "WiFi Scan Results: 4 entries", followed by a table with four columns: "Channel", "signal", "MAC", and "Access Point SSID". The table contains four rows of data.

Channel	signal	MAC	Access Point SSID
5	36	00:18:ba:72:db:a0	
9	20	00:14:a9:c8:47:c0	
7	18	00:09:5b:f8:89:4b	rabbit
4	14	00:19:2f:fa:92:20	rabwpa2

The Wi-Fi configurations are contained within TCPCONFIG 1 (no DHCP) and TCPCONFIG 5 (with DHCP, used primarily with infrastructure mode). You will need to #define TCPCONFIG 1 or #define TCPCONFIG 5 at the beginning of your program.

This what TCPCONFIG 1 macro definition include:

1. `#define _PRIMARY_STATIC_IP "10.10.6.100"`
2. `#define _PRIMARY_NETMASK "255.255.255.0"`
3. `#define MY_NAMESERVER "10.10.6.1"`
4. `#define MY_GATEWAY "10.10.6.1"`

12.3.2 WifiPingYou program

Objective:

1. Connect to known network and ping an IP.

Apparatus and components:

1. Wifi module.
2. Existing wifi network.

Experimental procedure:

1. IP address for PC is 10.10.8.2
2. SSID: rab-hoc

Find the *WIFIPINGYOU.C* sample program in the Dynamic C Samples\WiFi folder, open it with the File menu, then compile and run the sample program by pressing F9.

WIFIPINGYOU.C—sends out a series of pings to a RabbitCore module on an ad-hoc Wi-Fi network. This sample program uses some predefined macros. The first macro specifies the default TCP/IP configuration from the Dynamic C LIB\Rabbit4000\TCPIP\TCP_CONFIG.LIB library.

```
#define TCPCONFIG 1
```

Use the next macro unchanged as long as you have only one RCM5400W RabbitCore module. Otherwise use this macro unchanged for the first RabbitCore module.

```
#define NODE 1
```

Then changes the macro to #define NODE 2 before you compile and run this sample program on the second RCM5400W RabbitCore module. The next macros assign an SSID name and a channel number to the Wi-Fi network.

```
#define IFC_WIFI_SSID "rab-hoc"
```

```
#define IFC_WIFI_OWNCHANNEL "5"
```

Finally, IP addresses are assigned to the RabbitCore modules.

```
#define IPADDR_1 "10.10.8.1"
```

```
#define IPADDR_2 "10.10.8.2"
```

As long as you have only one RCM5400W RabbitCore module, the Dynamic C STDIO window will display the pings sent out by the module. You may set up a Wi-Fi enabled laptop with the IP address in IPADDR_2 to get the pings.

12.3.3 WifiScanAssociate program

Find the *WIFISCANASSOCIATE.C* sample program in the Dynamic C Samples\WiFi folder, open it with the File menu, then compile and run the sample program by pressing F9.

WIFISCANASSOCIATE.C— demonstrates how to scan Wi-Fi channels for SSIDs use *ifconfig IFS_WIFI_SCAN*. This takes a while to complete, so *ifconfig()* call callback function when it is done. The callback function is specified using *ifconfig IFS_WIFI_SCAN*.

Before you run this sample program, configure the Dynamic C TCP_CONFIG.LIB library and your TCPCONFIG macro.

1. Use macro definitions in the “Defines” tab in the Dynamic C Options > Project Options menu to modify any parameter settings.

If you are not using DHCP, set the IP parameters to values appropriate to your network

```
_PRIMARY_STATIC_IP = "10.10.6.100"  
_PRIMARY_NETMASK = "255.255.255.0"  
MY_NAMESERVER = "10.10.6.1"  
MY_GATEWAY = "10.10.6.1"  
IFC_WIFI_SSID = "My Access Point"
```

Set the macro *IFC_WIFI_SSID="My Access Point"* to define a C-style string to set the SSID of your access point as, for example, or use an empty string, "", to associate with the strongest BSS available.

2. If you are using DHCP, change the definition of the TCPCONFIG macro to 5. The default value of 1 indicates Wi-Fi with a static IP address.

Now compile and run the sample program. Follow the menu options displayed in the Dynamic C STDIO window.

While waiting for user input, it is important to keep the network alive by calling *tcp_tick(NULL)* regularly.

12.3.4 WifiMultipleaps program

Objective:

1. test networks with passwords keys and connects to them.

Apparatus and components:

1. wifi module+existing wifi network

Experimental procedure:

1. IP address for PC 10.10.6.250
2. Connect to first AP using *key-0*: 12345... WEP128bit
3. Connect to second AP using *key-1*: WEP

Find the *WIFIMULTIPLEAPS.C* sample program in the Dynamic C Samples\WiFi folder, open it with the File menu, then compile and run the sample program by pressing F9.

WIFIMULTIPLEAPS.C—demonstrates changing access points using WEP keys. You will need two access points to run this sample program. The access points should be isolated or on separate networks.

The sample program associates the RabbitCore module with the first access point (AP_0 defined below) with WEP key KEY0 (defined below). After associating, the sample program waits for a predefined time period, and then pings the Ethernet address of the access point (AP_ADDRESS_0). The sample program then associates with the second access point and pings its Ethernet address (AP_1, KEY1, AP_ADDRESS_1), and then switches back and forth between the two access points. When changing access points, first bring the IF_WIFIO interface down by calling *ifdown(IF_WIFIO)*. Next, change the SSID and key(s) using *ifconfig()* calls.

Finally, bring the IF_WIFIO interface back up by calling *ifup(IF_WIFIO)*. Note that the sample program checks for status while waiting for the interface to come up or down. Before you compile and run this sample program, check the TCP/IP configuration parameters, the IP address, and the SSID in the macros, which are reproduced below.

```
#define TCPCONFIG 1
```

```
#define IFC_WIFI_ENCRYPTION IFPARAM_WIFI_ENCR_WEP
```

You do not need to configure the SSID of your network since that is done from the access point names.

Now configure the access to the two access points.

```
// First Access Point
```

```

#define AP_0 "test1"

#define AP_0_LEN strlen(AP_0)

#define MY_ADDRESS_0 "10.10.6.250" // use this static IP when connected to AP 0

#define PING_ADDRESS_0 "10.10.6.1" // address on AP 0 to ping

#define KEY_0 "0123456789abcdef0123456789"

// Second Access Point

#define AP_1 "test2"

#define AP_1_LEN strlen(AP_1)

#define MY_ADDRESS_1 "10.10.0.99" // use this static IP when connected to AP 1

#define PING_ADDRESS_1 "10.10.0.50" // address on AP 1 to ping

#define KEY_1 "0123456789abcdef0123456789"

#define IFC_WIFI_SSID AP_0

#define _PRIMARY_STATIC_IP MY_ADDRESS_0

```

Modify the access point names and keys to match your access points and network.

12.3.5 WifiDHCPorTStatic program

Find the *DHCPpromodified.c* sample program in the Dynamic C Samples\WiFi folder, open it with the File menu, then compile and run the sample program by pressing F9.

WIFIDHCPORSTATIC.C—demonstrates the runtime selection of a static IP configuration or DHCP. Before you compile and run this sample program, check the TCP/IP configuration parameters, the IP address, and the SSID in the macros, which are reproduced below.

```

#define USE_DHCP

#define TCPCONFIG 1

#define _PRIMARY_STATIC_IP "10.10.6.100"

#define IFC_WIFI_SSID "rabbitTest"

```

Modify the values to match your network. You may also need to modify the values for MY_GATEWAY if you are not pinging from the local subnet.

Now press F9 to compile and run the sample program. When prompted in the Dynamic C STDIO window, type 's' for static configuration or 'd' for DHCP.

12.3.6 WifiPassPhrase program

Objective:

1. Generates a random key

Apparatus and components:

1. Wifi module network

Find the *WIFIDHCPORSTATIC.C* sample program in the Dynamic C Samples\WiFi folder, open it with the File menu, then compile and run the sample program by pressing F9.

This program demonstrates how to perform the CPU-intensive process of converting an ASCII passphrase into a WPA PSK hex key. This is motivated by the need to convert passphrases to hex keys in order for WPA PSK to work. For security reasons, however, the mapping function is deliberately designed to be very CPU intensive, in order to make a dictionary-based attack more difficult. For example, the conversion can take of the order of 15 seconds on the RCM5400W. Since this may be an unacceptable amount of time to "block" the application program, a method of splitting up the computation is provided. The complete process takes 4096 iterations of a secure hashing function. You can perform the process a few iterations at a time. For example, performing one iteration at a time will block the application for only about 1/100sec. Performing 10 at a time will block for 1/10 sec, and so on. There is only a few percent overhead in breaking up the entire process into single iterations.

The procedure depends on the following function:

```
wpa_passphrase_to_psk_init(  
    &pps,                // <- state structure, managed by library  
    "passphrase", // <- your passphrase  
    "ssid",            // <- target SSID  
    ssid_length,    // <- length of the SSID string (since allowed binary)  
    key );          // <- returned hex key - must be 32 byte char array.
```

After running the program, you can see how a passphrase can be created.