We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

15

Remote SMS Instrumentation Supervision and Control Using LabVIEW

Rafael C. Figueiredo[1], Antonio M. O. Ribeiro[1],
Rangel Arthur[2] and Evandro Conforti[1]
[1]*Department of Microwave and Optics - University of Campinas (FEEC/Unicamp),*
[2]*Division of Telecommunication Technology (FT/Unicamp),*
Brazil

1. Introduction

The hot emergence of remote monitoring and control systems in recent years is closely related to the outstanding advance in electronics and instrumentation techniques. As one of the earliest examples, the telemetry experiments using satellite-relay systems for remote hydrologic data collection (Glasgow et al., 2004) in the 70's should be remembered. Since then, a fast and continual appearance of real time monitoring systems occurred, new devices and tools have kept coming up, and more and more powerful resources are available with decreasing prices. Hence, remote monitoring and control systems are gaining market-share in diversified areas such as industry, security, education and even in health-care applications.

In every distinct area it is possible to evince different advantages and applications when using remote systems. To exemplify the industrial field, a remote data acquisition may be used to monitor machine health and energy consumption in environments where it is difficult to access or to use wired data acquisition systems (Khan et al., 2004). As another example, a remote fault diagnostic system to check machines status including data, image, and video trough the Internet and mobile terminals by wireless application protocol (WAP) (Wang et al., 2007) was implemented. In the security area, a robot can inspect home environment giving general information and alarms about dangerous situations using wireless network (Zhang et al., 2007). In addition, real-time remote health monitoring systems can acquire and process data to evaluate long bridge structure reliability (Jianting et al., 2006). Regarding the educational field, there is a web-based remote control laboratory that employs a greenhouse scale model for teaching greenhouse climate control techniques using different hardware and software platforms (Guzmán et al., 2005), and also a virtual laboratory that permits remote access via Transmission Control Protocol/Internet Protocol (TCP/IP), enabling control and supervision through the Internet (Garbus et al., 2006). In addition, the results of an evaluation of remote monitoring in home health care show a reduction in nursing visit and hospitalization rates in the remote monitoring patient group (Rosati, 2009).

The example list can be much longer. The technological advances allow development of systems with more and better resources and higher transmission rates, consequently increasing its complexity and budget. However, in various applications we just need to

control or monitor simple tasks, which can be performed by less complex and cheaper systems. In some cases there is a very small amount of data being collected and the necessary command to perform a specific action can be sent in a simple text message with a few characters. In this context, the short message service (SMS) used in cellular networks is being adapted to several remote applications. Despite some restrictions, such as the limit of 160 characters per message without a user-friendly graphical interface, the SMS is widely available throughout the digital cellular network. It can be used by any 2G cell phone, it is a low cost service, and it does not overload the network since this service does not consume much network resources. These and other advantages led to the development of diversified remote applications using SMS. Among many examples, the SMS can be used to monitor and/or control remotely the functions of an automotive alarm (Figueiredo et al., 2008), the ambient temperature in regions of difficult access (Zarka et al., 2006), a computer network (Sarram et al., 2008), an electronic switching system (Zahariah et al., 2009), as well as the status of power earth lines (Xiao et al., 2009).

Addressing now the main focus of this chapter, the Laboratory Virtual Instrumentation Engineering Workbench (LabVIEW) is a graphical programming environment from National Instruments that deserves attention not only on remote systems, but in a wide range of applications. Originally released in 1986, LabVIEW is commonly used to develop sophisticated measurement, test, and control systems; and its graphical language permits the development of programs even by inexpert programmers. The popularity of this platform has been increasing. Some years ago it was included in the curriculum of electrical engineering students by practical approaches involving experiments in electronics (Alsaialy et al., 2003; Higa et al., 2002). It is also possible to highlight a long list of applications using LabVIEW, among which we cite a system to analyze and monitor the mechanical vibration of industrial machines (Lita et al., 2005); a remote-accessed instrumentation laboratory for digital signal processors training on the Internet (Gallardo et al., 2006); a solution of data acquisition/distribution designated for remote process control and long distance learning (Sgârciu & Stamatescu, 2010); a tele-health system that detects changes in heart rate and blood pressure of the patient in advance and sends an alert SMS to the doctor (Sukanesh et al., 2010); and a low cost system to monitor and control laboratory instruments remotely by SMS (Figueiredo et al., 2009). This chapter will be developed based on this last application, focusing on the necessary steps for the LabVIEW-based software development and describing all the programming details for creating a remote monitoring and control system using LabVIEW and SMS. Additionally, at the end of the chapter, an example of system applicability is illustrated in a measurement procedure, validating the developed tool and demonstrating the implementation in a specific application.

In general terms, a computer running LabVIEW is connected to a Global System for Mobile Communication (GSM) modem via RS-232 serial communication. This GSM modem is controlled by AT commands, which is a standard-command set with instructions used for modem control, detailed in Section 2 of this Chapter. The GSM modem enables the user to monitor and control a specific instrument while away from it, using a cell phone that sends and receives text messages through SMS. As an example to be discussed here, the instrument can be connected to LabVIEW via IEEE-488 General Purpose Interface Bus (GPIB), but it can also be easily adapted for other communication forms.

The chapter is organized as follows:

- Section 2 aims to provide basic knowledge about issues that will be raised during the system development. The text brings basic information about LabVIEW, SMS and AT

commands, always focusing on the resources that will be applied in this particular project;
- Section 3 describes the system architecture. Also, the program created in LabVIEW will be fully detailed;
- Section 4 presents a case study in which the designed tool is applied to an 1.8-GHz narrow band envelope measurement made in an urban area to characterize the coherence bandwidth between two frequency-spaced radio mobile signal (Ribeiro et al., 2007), demonstrating the remote SMS system operation and results.;
- Section 5 gives the concluding remarks followed by acknowledgments in Section 6;
- Section 7 comes with the references.

2. Background

As mentioned earlier, the purpose of this Section is to provide a basic understanding of the issues to be raised throughout the chapter. Initially, the SMS will be briefly addressed, followed by some explanation of AT commands, listing the main commands that will be used to control the GSM modem. Lastly, some basic LabVIEW concepts related to the development of our remote application will be quickly described.

2.1 Short Message Service (SMS)

The SMS technology evolved from the GSM standard in the early 90's. It is currently available in the entire GSM network and can be accessed by practically every cell phone. The service enables the user to send messages with the maximum length of 160 or 140 characters when using 7-bit or 8-bit encoding, respectively. The system can segment messages that exceed the maximum length into shorter messages. In this case, it must use part of the payload for a user-defined header that specifies the segment sequence information. When a text message is sent from a user to another, the phone actually sends it to a short message service center (SMSC), where the message is stored and delivered when the recipient is on the network. Namely, it is a store-and-forward operation. In this paradigm, the system keeps resending the message for some period of time until it is successfully received. The SMSC usually has a configurable time limit for how long it will store the message, and be able to resend it in case of failure in the previous attempts. Messages can be sent using text mode or protocol data unit (PDU). The text mode is a character-based protocol and it is the simplest mode to send messages, suitable for unintelligent terminals. The PDU may contain control information, address information, or data, i.e., this mode adds the convenience of binary data to be transmitted, not just characters (Peersman et al., 2000; Brown et al., 2007). In this project the exchanged messages will be composed only of ASCII characters. Therefore, the SMS text mode will be used, which, despite allowing only transmission of characters, it is simpler to work with.

In this project, the GSM modem is responsible for sending and receiving text messages. It can accesses the GSM network like a mobile phone, differing in the way it is controlled, which is accomplished through AT commands. The command set may vary according to the device manufacturers. Here, the GSM modem used was assembled with the Motorola G24 module, whose operation and main AT commands are described below.

2.2 AT commands

AT or Hayes commands are a set of data stream commands for communication between computers and modems. Originally developed by Hayes Microcomputer Products in the beginning of modem communications, the AT commands became a standard for modem control. AT is the prefix for almost every command-line set; it is derived from the word "attention" and tells the modem that a request is coming. Requests can be made to accomplish a particular action, to change or query a specific parameter and they are sent to the modem through strings (Figueiredo et al., 2008; Jawarkar et al., 2007; Zhicong et al., 2008).

An AT command line may be composed by one or more commands, but it must not exceed 140 characters. Extended commands are preceded by the "+" symbol and are separated on the same line by a semicolon, whereas basic commands can be separated by a single blank space. The syntax of an extended command enables to query or change more than one sub-parameter, but the sub-parameters that will not be modified can be omitted from the command line. In order to clarify the structure of an AT command line, an example is shown in Figure 1, where the suffix <CR> is the ASCII (American Standard Code for Information Interchange) character to "carriage return". After a query, the modem responds with a result code that can include the current values and the range of possible values of the consulted parameter. The result code is also emitted after a request, informing the success or failure of the execution through the strings OK or ERROR, respectively. An example of the result code is shown in Figure 2, where <LF> is the ASCII character to "line feed" (Figueiredo et al., 2008; Motorola, 2008).

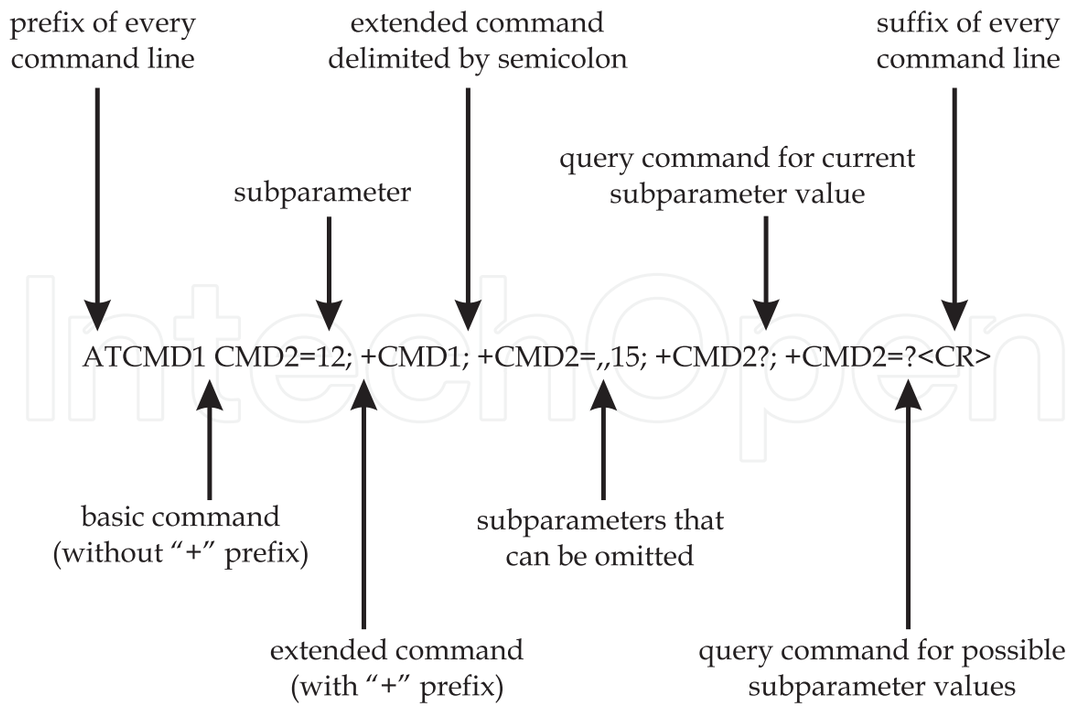


Fig. 1. Basic structure of an AT command line [adapted from (Motorola, 2008)].

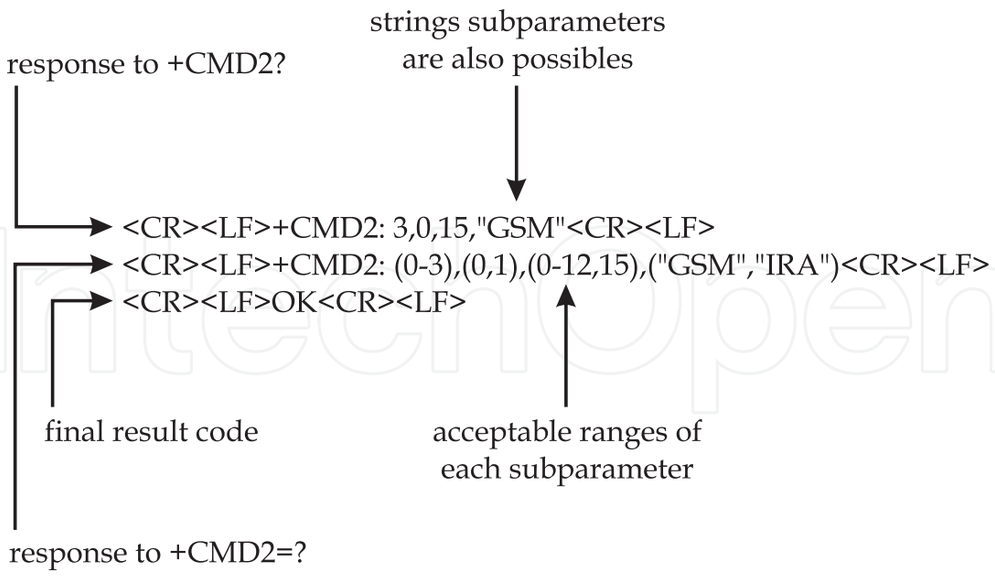


Fig. 2. Basic structure of result codes [adapted from (Motorola, 2008)].

2.2.1 Main AT commands

The list of all AT commands available for a particular device can be obtained from its manufacturer. Here we listed only the main commands that were used to develop this project and for the specific employed modem (Motorola, 2008).

- **AT&K0**: this command disables the flow control and must be the first command sent when the serial communication cable does not have the pins to transmit the RTS (request to send) and CTS (clear to send) signals;
- **ATE0**: disables the echo from the modem, i.e., the ASCII characters sent are not repeated back;
- **AT+CSQ?**: queries the modem signal intensity and ranges from 0 (no signal) to 31 (maximum signal);
- **AT+CMGF**: configures the message format; "AT+CMGF=1" sets to text mode, where the body of the message and its headers are given as separate parameters;
- **AT+CNMI**: configures how incoming messages will be handled by the modem; "AT+CNMI=,2" routes incoming messages directly to the terminal in a result code with the prefix "+CMT". Analogously in a cell phone configured with this mode, instead of an incoming message warning, the content of the message is immediately displayed on screen;
- **AT+CNMA**: acknowledges the receipt of a message and when the "CNMI =,2" mode is configured this acknowledgment must be sent whenever a new message is received;
- **AT+CMGS**: sends a short message from the modem to a given recipient and its syntax is "AT+CMGS=<"*recipient number*">,<*call type code*>" followed by the message content, the call type code for local calls is "129".

2.3 LabVIEW

LabVIEW is a powerful platform for algorithm development, design, prototyping, and interfacing of embedded system with real-world hardware. LabVIEW uses graphical

language (G) that enables the creation of programs in block-diagram form, which are called virtual instruments (VI). The main executable program is called the top-level VI and the programs used as modular subroutines are called subVIs. Every VI consists of two main components, the graphical user interface that is named front panel; and the graphical code that implements the functionality of the VI, called block diagram. Once a VI is created and saved, it can be used as a subVI in the block diagram of another VI, acting as a subroutine (Gan & Kuo, 2007; Oussalah & Djezzar, 2010; Sumathi & Surekha, 2007).

The front panel contains objects known as controls and indicators. Controls are the inputs that provide data to the block diagram, and indicators shows output data from the block diagram. These objects, such as buttons, switches, LEDs and graphs are selected from the controls palette. In the Figure 3, the controls palette and the front panel with some examples of controls and indicators are shown.

Front panel controls and indicators have a corresponding terminal block on the block diagram, which holds the graphical source code that defines how the VI will operate. A block diagram is built with terminals and nodes available in the functions palette and that are connected by wires. An example using the same controls and indicators used in the front panel is illustrated in the Figure 4; the colors indicate the data type, e.g., numerical indicators are blue and floating-point ones are orange.

The block diagram code executes based on the principle of dataflow. The rule of dataflow states that functions execute only when all of their required inputs are populated with data. This programming architecture allows both sequential and parallel execution to be easily defined graphically. When a node completes execution, it supplies data to its output terminals on the right, and follows the dataflow imposed by the wires connecting the block diagram objects (Gan & Kuo, 2007; Sumathi & Surekha, 2007).

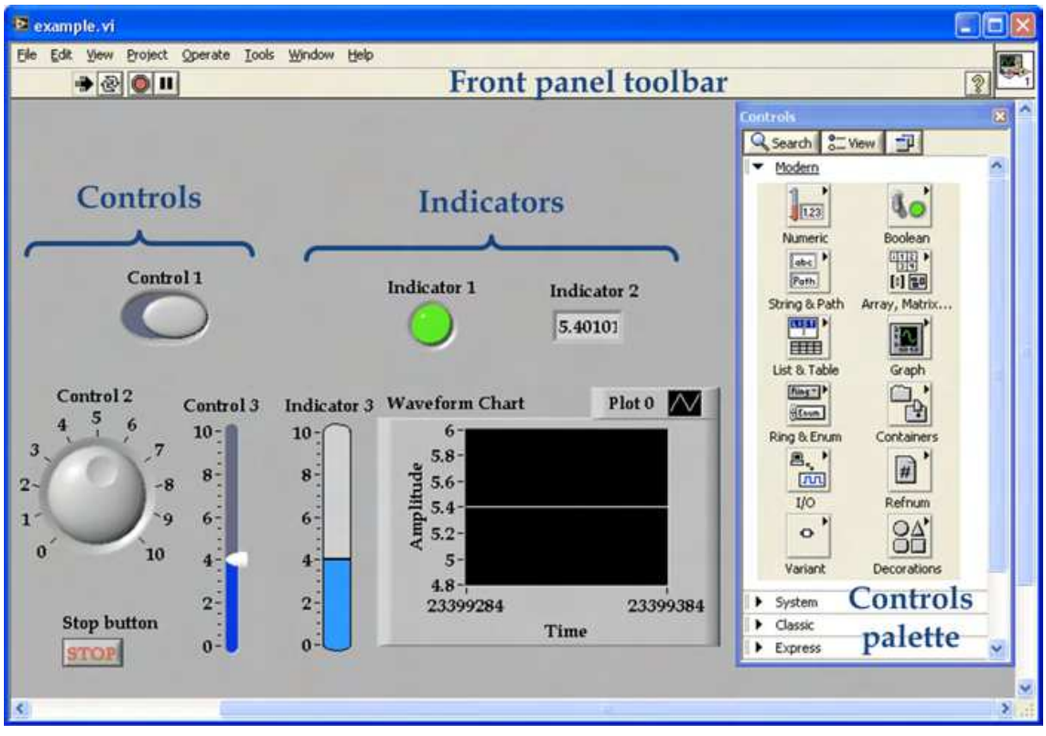


Fig. 3. LabVIEW front panel example and the controls palette.

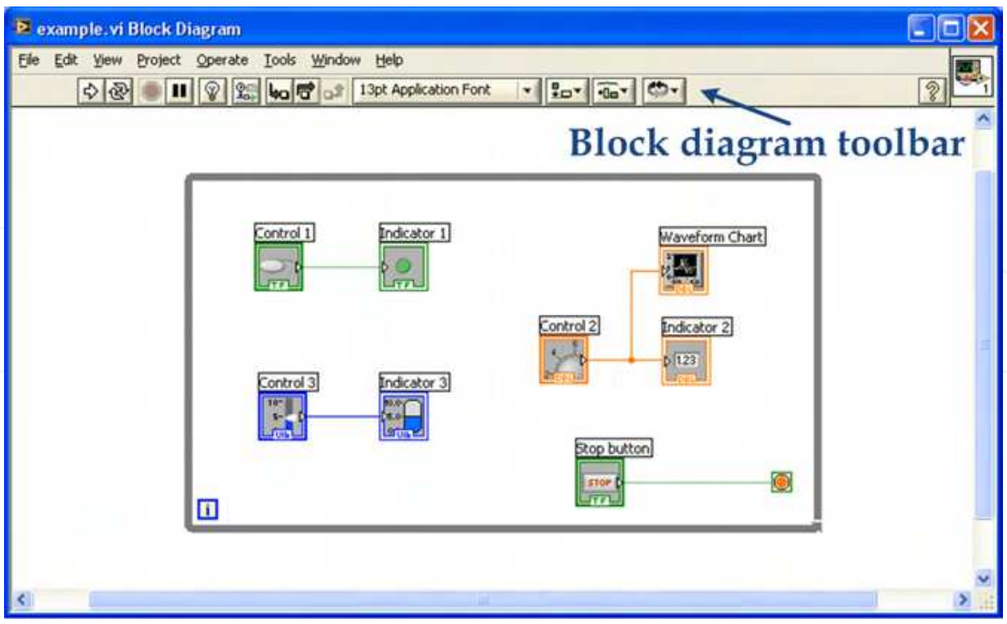


Fig. 4. LabVIEW block diagram example with the controls and indicator showed in the above front panel.

2.3.1 Some programming structures

The LabVIEW structures are graphical representations of the loops and case statements of text-based programming languages. Structures are found in the functions palette and used on the block diagram to repeat blocks of code and to execute code conditionally or in a specific order. Among various structures available in LabVIEW, the While loop will be briefly described in this subsection, including the case structure, and sequence structures. Illustrations of those graphical LabVIEW structures are shown in Figure 5.

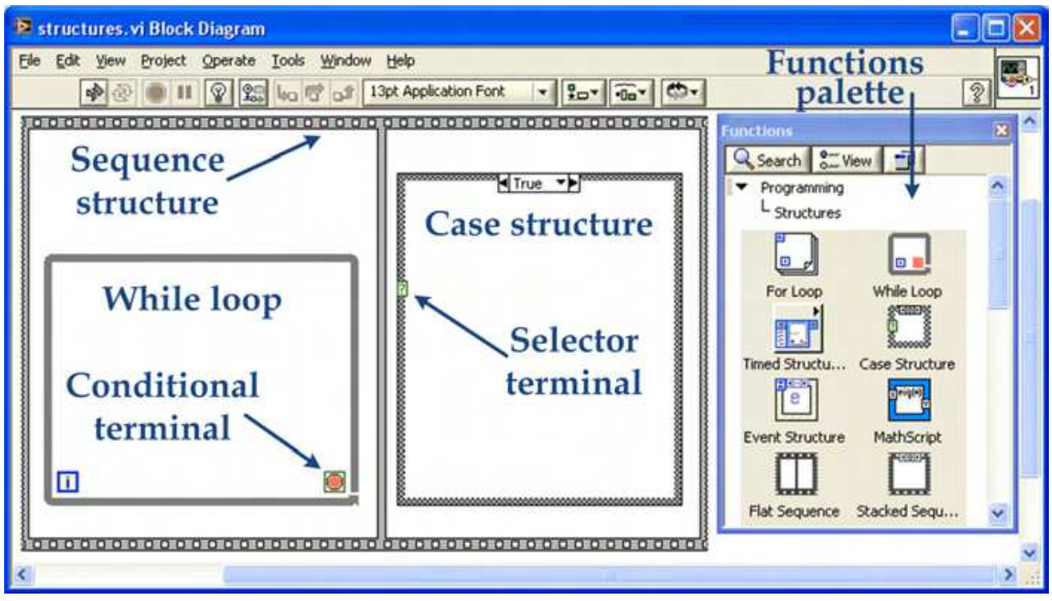


Fig. 5. Some LabVIEW programming structures and the functions palette.

- **While loop**: structure that executes the subdiagram inside its borders, until the Boolean value wired to its conditional terminal be "false". The conditional terminal value is checked at the end of iteration, and if the value is "true" the iteration repeats. Since the default value of the conditional terminal is "false", the loop iterates only once if left unwired, but it is possible to change this default status. Hence, the looping "while true" can be modified to loop unless it is true (Sumathi & Surekha, 2007).
- **Case structures**: they have two or more subdiagrams and only one of which will be executed, depending on the value of the Boolean, numeric, or string value wired to the selector terminal. If a Boolean value is wired to the selector terminal, the structure has two cases, "false" and "true". If a numeric or string data type is wired to the selector, the structure can have almost unlimited cases (Sumathi & Surekha, 2007).
- **State Machine**: programming concept to provide an intelligent control structure and that revolves around three concepts: the state, the event, and the action. In simple terms, a state machine in LabVIEW is a case structure inside a While loop. The While loop provides the ability to continuously execute until the conditional operator is set "false" and the case statement allows for variations in the code to be run (Bitter et al., 2006).
- **Sequence structures**: these types of structure are used to control the execution order of nodes that are not data dependent on each other by arranging its elements in a certain sequence that is called control flow. The sequence structure looks like a frame of film, and executes first the subdiagram inside the first frame, followed by the subdiagram in the second, until the last frame (Sumathi & Surekha, 2007).

3. System development

This Section presents the basic system architecture, an overview of the program, and all the necessary steps for the development of the main program and its subVIs, which constitute the programming framework for the SMS system.

3.1 System architecture

The system architecture is relatively simple, consisting of a GSM modem and a computer with LabVIEW. This main block is then connected to the instruments that will be monitored and/or be controlled. The interface between the user and the system can be accomplished by any 2G phone, which will be responsible for the communication with the modem, as illustrated in Figure 6. In this project, in particular, the Version 8 of the LabVIEW platform was used, and a GSM modem was assembled with the Motorola G24 module, as displayed in Figure 7.

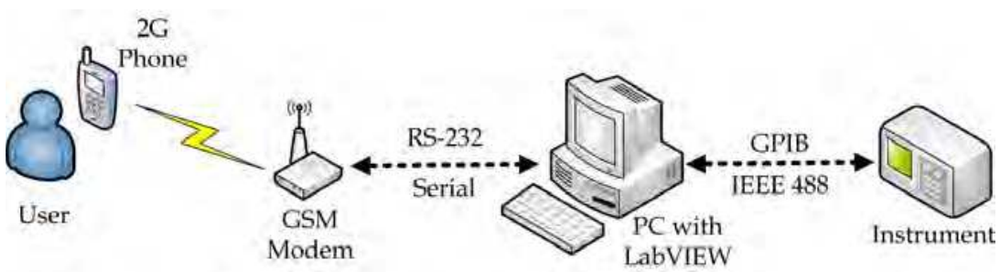


Fig. 6. SMS system architecture diagram.

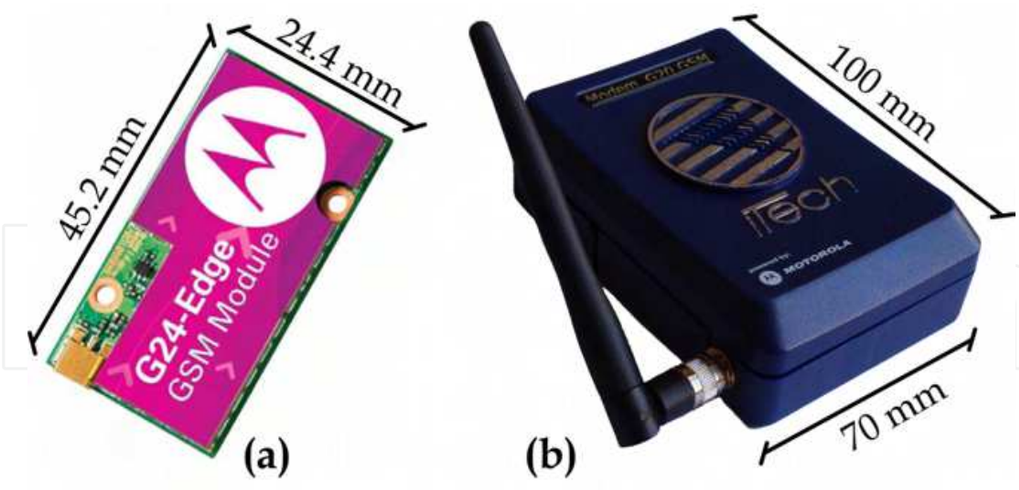


Fig. 7. (a) The G24 module [adapted from (Motorola, 2008)] and (b) the GSM modem used in the project, with their respective dimensions.

3.2 Program overview

In general terms, the program will perform the initial settings of the GSM modem; prepare it for sending/receiving messages; wait for commands sent via SMS; execute the requested action; and re-await a new text message with another command. The basic flowchart of these actions is illustrated in Figure 8. An overview of the block diagram program created in LabVIEW is shown in Figure 9.

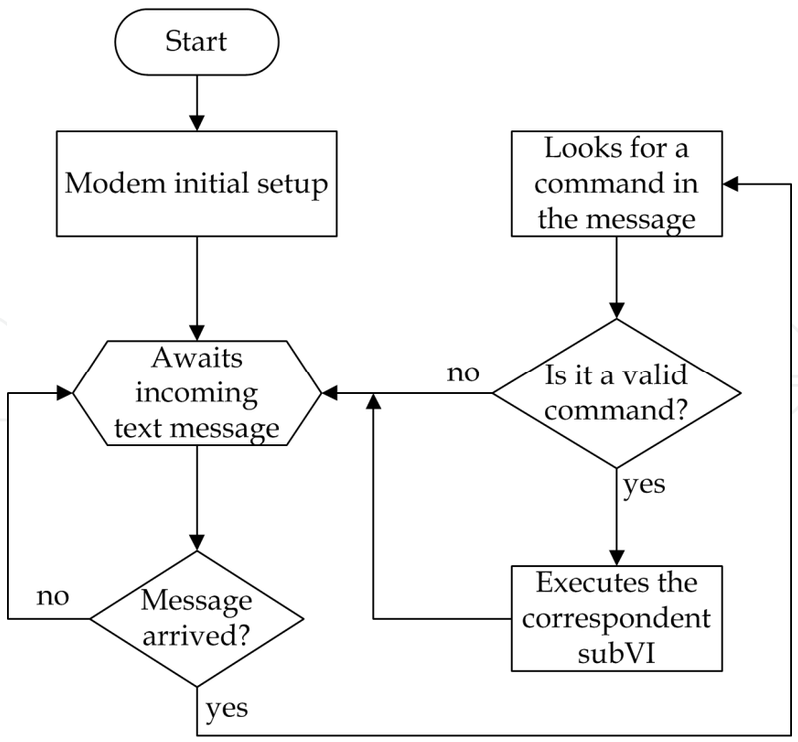


Fig. 8. Flowchart of the main functions to be programmed using LabVIEW.

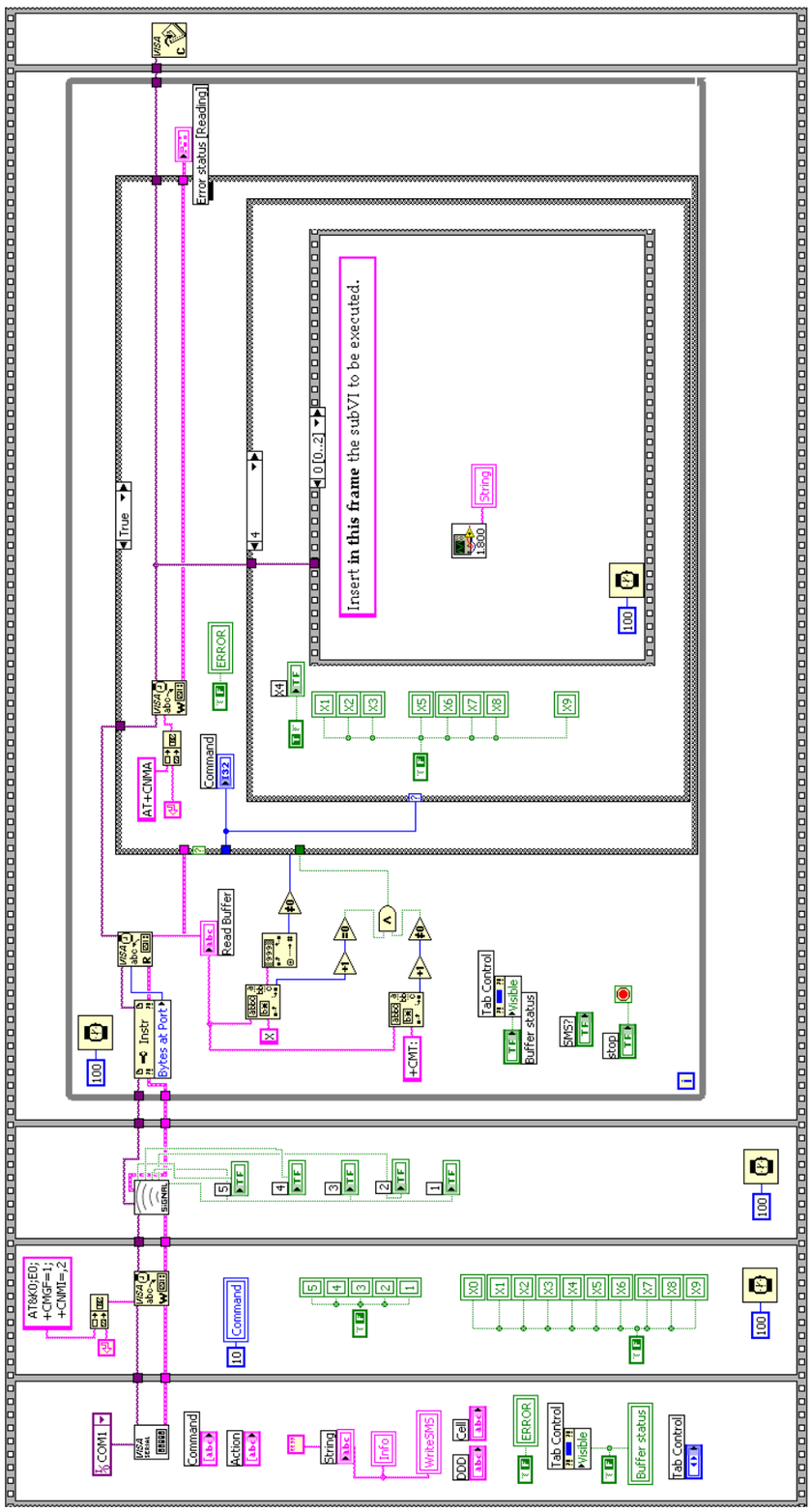


Fig. 9. Main program overview.

In addition to these actions, the user can choose to receive text messages confirming the execution of the requested actions. Also, there are some additional settings that need to be performed in a specific sequence, such as serial communication port settings and GSM modem signal intensity checking. Hence, the LabVIEW main program was implemented in a flat sequence structure, as shown in the above picture. In the figure it is not possible to identify clearly all the details, but for now the purpose is to provide only an overview of the programming framework, since each step of the program will be detailed in the following subsections.

3.3 Block diagram

The graphical source code will be explained following the sequence structure in which it is inserted, that is, from left to right. Likewise, the subVIs will be described at the moment they appear in the code sequence.

3.3.1 Program initialization

The code piece responsible for the program initialization is shown in Figure 10.

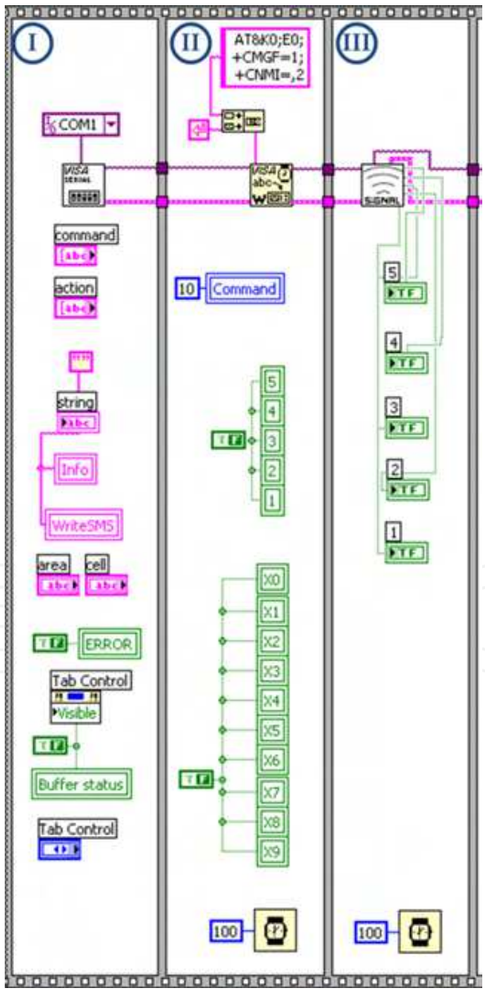


Fig. 10. Frames responsible for the communication port and modem initialization.

In frame I, explaining in a top-down approaching, there is a VISA block to configure the serial communication port. VISA is a standard application programming interface that can control GPIB, USB, serial or other communication protocols, making the appropriate driver calls. So the user does not have to learn the instrument communication protocol. In this block it is necessary to inform only the port used for communication, "COM1" in this case. For the other settings the block default options are used, which are 8 data bits, 1 stop bit, no flow control, no parity, and baud rate of 9600 bps. The arrays "command" and "action" are placed under the VISA block, forming a table in the front panel where the user is able to write the actions performed by each code. This is followed by the array "info", which is initially filled with an empty string. But later, it will be used to display an execution confirmation message, which is the text to be sent for the user, if this confirmation option (to be explained later) is activated. Continuing, there are the arrays "area" and "cell", through which the user should inform the area code and the phone number for which the confirmation messages aforementioned will be sent. At the bottom of the frame there is a tab control setting, which allows the visualization of what is being read and written in the serial buffer. It is useful to be used during the program development, although when the application is working correctly this tab is not very helpful. Therefore, a button was created that allows users to either view it or not.
The key function of Frame II is to configure the GSM modem by sending it the AT commands to disable the flow control and the echo; configure the SMS to text mode; and adjust the incoming messages settings, as described early in the subsection 2.2.1. The commands are sent by serial communication using the "VISA write" block in LabVIEW. This frame also adjusts the value of the global variable "command" to 10, but this functionality will be explained later, in subsection 3.3.3. In addition, the Boolean value "false" is attributed to the global variables representing the GSM modem signal intensity and the commands codes available in the program. These global variables correspond to LED indicators, which will be seen on the front panel.
The frame III checks the GSM modem signal intensity through a subVI, described below.

3.3.2 SubVI to check signal intensity

The routine to check the modem signal was bundled into a subVI, shown in Figure 11. In general terms, this subVI queries the signal intensity from the GSM modem via serial communication in the first frame and reads the answer in the second frame. Then, the Boolean value "true" is attributed to indicators 1 to 5, representing LEDs on the front panel in a manner analogous to the indicators of mobile phone screens.
To check the signal intensity, the corresponding AT command is sent through a "VISA write" block. Next, a property node is used to read the number of bytes of the modem response and use it as an input parameter of a "VISA read" block, and then the decimal number corresponding to the signal intensity is extracted from the string received by the modem. When this number is zero, there is no signal. If the number is between 1 and 9, there is a weak signal corresponding to one signal bar. A number greater than 9 indicates two signal bars, greater than 19 indicates three bars, greater than 29 indicates four bars, and a number greater than 30 indicates the maximum signal level, with five bars.
Next, the parameters obtained in this subVI are forwarded to the indicators of the main program.

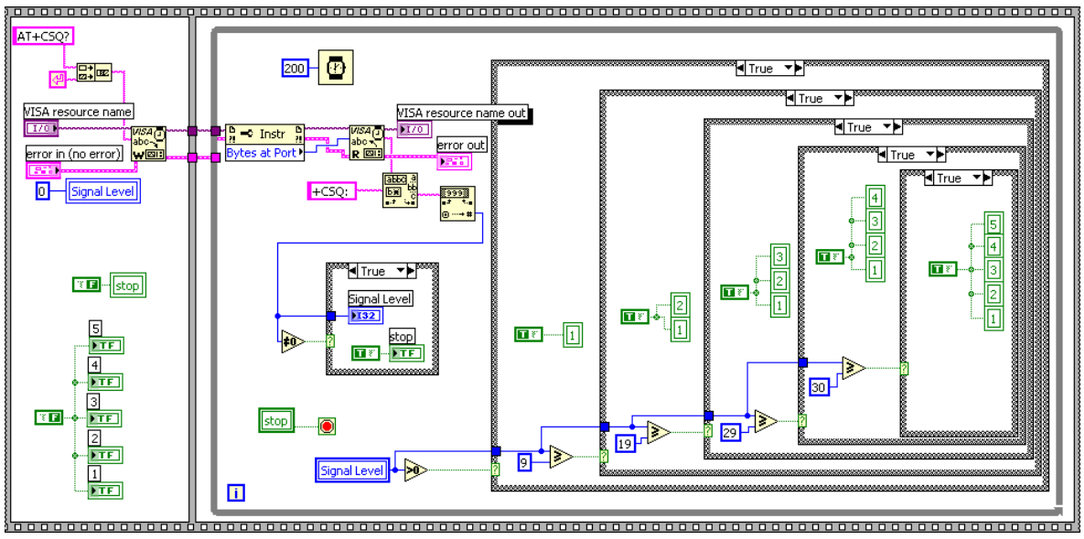


Fig. 11. Block diagram of the subVI to check the GSM modem signal intensity.

3.3.3 Recognizing and executing SMS commands

This part of the graphical source code was designed as a state machine. It can be considered the heart of the system and subdivided into two main stages. The first recognizes the code in a message received by the modem and the second selects the action to be performed, according to the command received.

In the first stage, shown in Figure 12, the program is within a While loop waiting for incoming text messages. When a message is received, the property node informs its number of bytes to the "VISA read" block, from where the message goes to the "read buffer" array, to be shown in the front panel tab. The message content also goes to "match pattern" blocks, where it is performed the search for a valid command code in the message text. At this point Boolean results are generated, which will be used in the second stage.

The message received by the modem is composed by several other characters besides the text informed by the user, such as date and number of the sender. Therefore, to recognize a command in the received message it is necessary to create a code table. In this project the codes are represented by the character X followed by a decimal digit from 0 to 9, and they were chosen to avoid confusion with the header message characters and to be easily typed on the keypad mobile phone (Figueiredo et al., 2009).

Furthermore, in this first stage a control variable is created that will be displayed on the front panel as a switch, allowing the user to choose whether it is desired or not to receive confirmation messages after the execution of a requested command. In the While loop, is also checked whether the user has enabled the view of the "tab control" mentioned earlier.

At the beginning of the second stage, shown in Figure 13, the acknowledgment AT command "+CNMA" (required after each received message) is sent to the modem. Next, the program enters into a case structure, according to the Boolean results obtained from the "match pattern" blocks of the first stage. If the block does not find a valid code, the program goes to a "false" case structure shown in Figure 14. This will return an invalid code message to the front panel, through the array "info". It will also send the user mobile phone a warning via SMS, if the confirmation option is active. When a valid command is received, the program enters into a second case structure, according to the decimal number after the X

character. In this step, one of the X0 to X9 indicators will receive Boolean value “true”, in accordance with the received code, turning on a LED indicator on the front panel, representing the code that is running. After that, still inside this case structure, the program runs a stacked sequence structure, where it must be placed the subVI to be executed for each code. Thus, in the case #0, it is placed the subVI to be executed when the X0 command is sent to the GSM modem via text message. In the case #1 the subVI is placed in such a way that will run for the received code X1 and so on, until the last code, which is X9 in this project. Besides the case structures being used, it is necessary to create an additional case, which is left blank and enabled as default. In this case, the empty structure is the eleventh one. For this reason the number 10 was assigned to the global variable “command” mentioned earlier in subsection 3.3.1.

Increasing the number of codes available for execution is not a difficult task. However, it is necessary to increase the number of cases in the structure and modify the Boolean routine that enters in the case structure, in order to make it able to interpret the new codes.

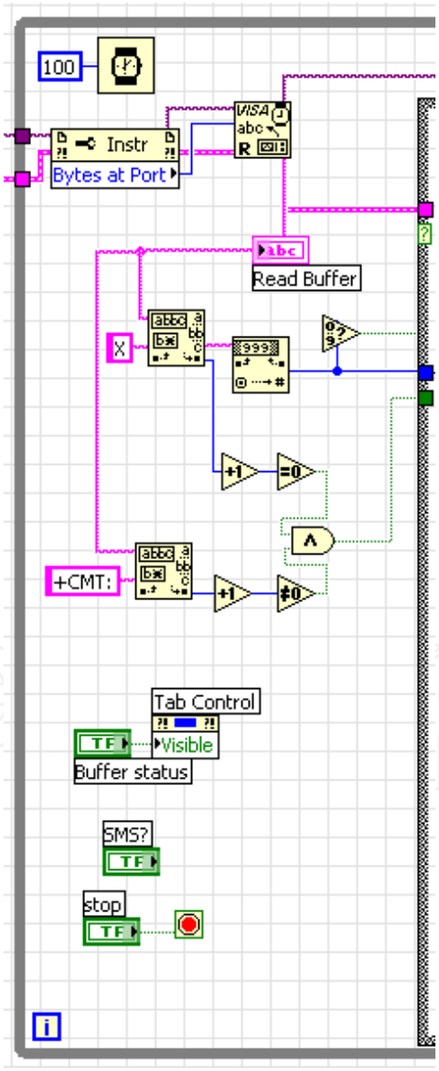


Fig. 12. Code that searches for a valid code in the message received by the modem.

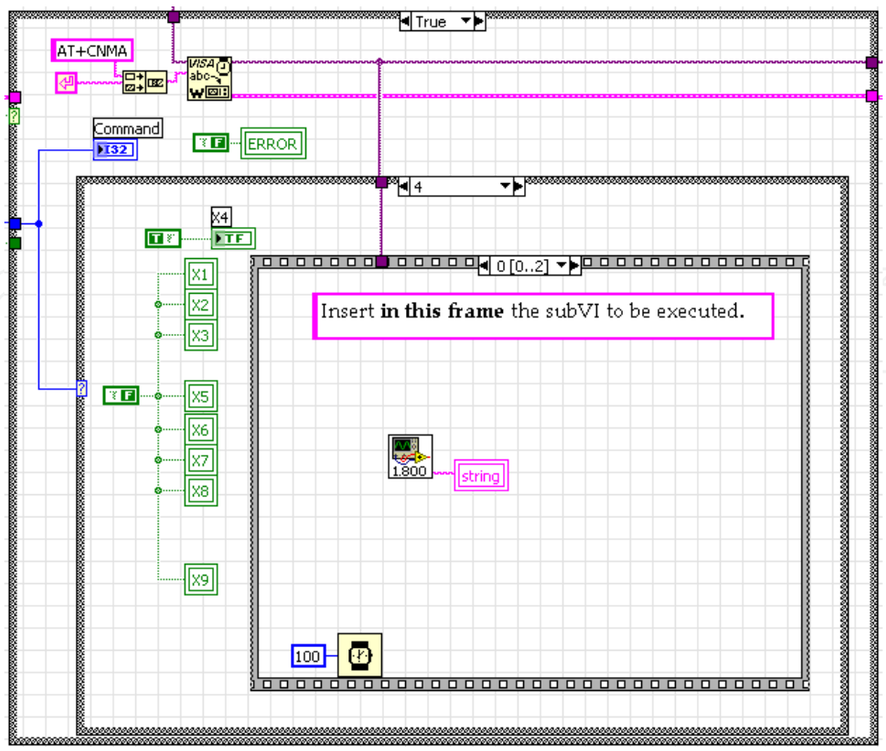


Fig. 13. Structure that executes the subVI corresponding to the received command.

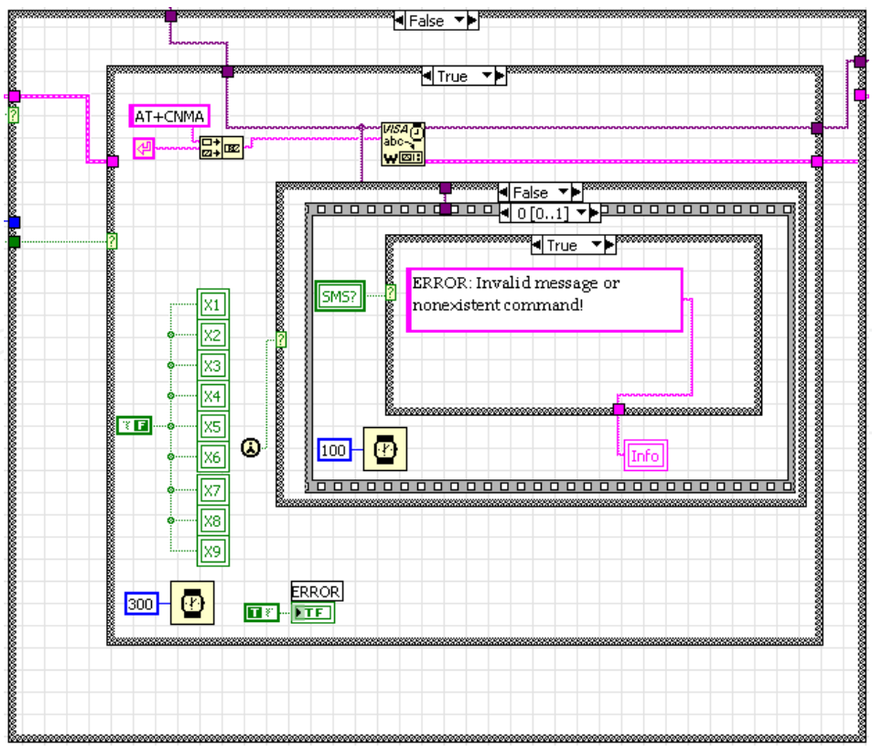


Fig. 14. Routine that deals with cases where an invalid command is received by the modem.

The subVIs to be executed for each command must be placed within the first frame (frame zero) in the sequence structure that is inside the case structure, as illustrated above in Figure 13. After running the subVI, the next frame assembles a text that confirms the execution of the requested action and sends it to the front panel, through the "info" indicator, as shown in Figure 15. This same text is used by the modem to send the user mobile phone a message via SMS. This text message is sent through a subVI that runs only if the confirmation option is active, and it is detailed in the next subsection.

3.3.4 SubVI to write message

This subVI is built in a sequence structure, numbered from I to VII, as shown in Figure 16. The frame I specifies the resource to be opened informing the VISA name to a "VISA Flush I/O Buffer" block, in this case the resource to be opened is the serial communication port COM1. The frame II gets the phone number informed by the user and passes it to the frame III, where it is assembled in the AT command to send a text message from the GSM modem. Frame IV only transmits the ASCII code for "carriage return", enabling the entry of the characters that form the message payload, which is transmitted in the frame V. In frame VI the "\1A" ASCII characters, which correspond to the "CTRL+Z" command, are sent to the modem, indicating that the message was written, and enabling the modem to send the SMS. In this frame is used a large time delay for allow the modem to send the text message.

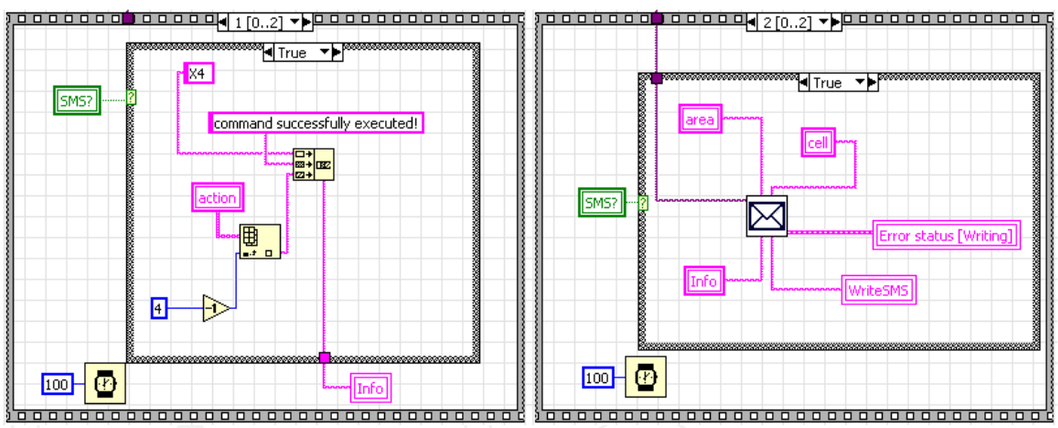


Fig. 15. Sequence structure that assembles and sends the text message.

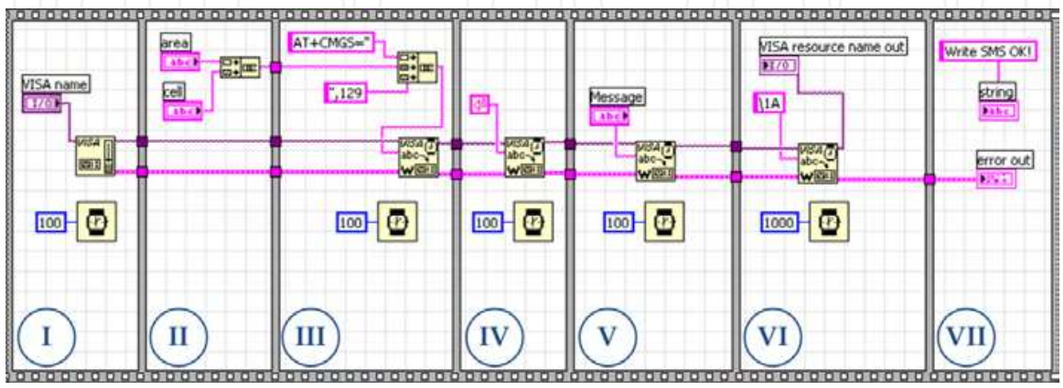


Fig. 16. SubVI to send messages from the GSM modem to the user mobile phone.

The frame VII just informs that the subVI for writing and sending the message was executed correctly, showing this information in the "tab control" placed on the front panel.
All procedures described as the heart of the system are inside a While loop. Hence, after performing all these actions described, the program returns to the point where it awaits for the arrival of new text messages by the modem. When a message arrives, this whole process is repeated until the user press the stop button located on the front panel.
When the stop button is pressed, before the program stops running completely, it goes to the last frame of the sequence structure, which can be seen in Figure 9 shown earlier. This last frame function is just to close the communication with the COM1 port through the "VISA close" block, leaving it available for other possible activities that need to use this communication port.
At this point it is possible to redesign the flowchart showed earlier in Figure 8, adding now all program features and also the routine hierarchy, including the subVIs, as shown in Figure 17. Thus, the block diagram explanation is closed with a summarized review of its overall operation.
As seen in Figure 17, the computer communication port and GSM modem settings are the first tasks accomplished. Then, the signal intensity of the modem is checked and the program enters in a routine waiting for incoming text messages. When a message arrives, the program searches for a valid command in the text, and once it has been found, the corresponding subVI is executed. Next, if the confirmation option is active, a text message is sent the user mobile phone by the modem, confirming the action execution. Lastly, when the program is stopped, the serial communication port is closed and liberated for other applications.

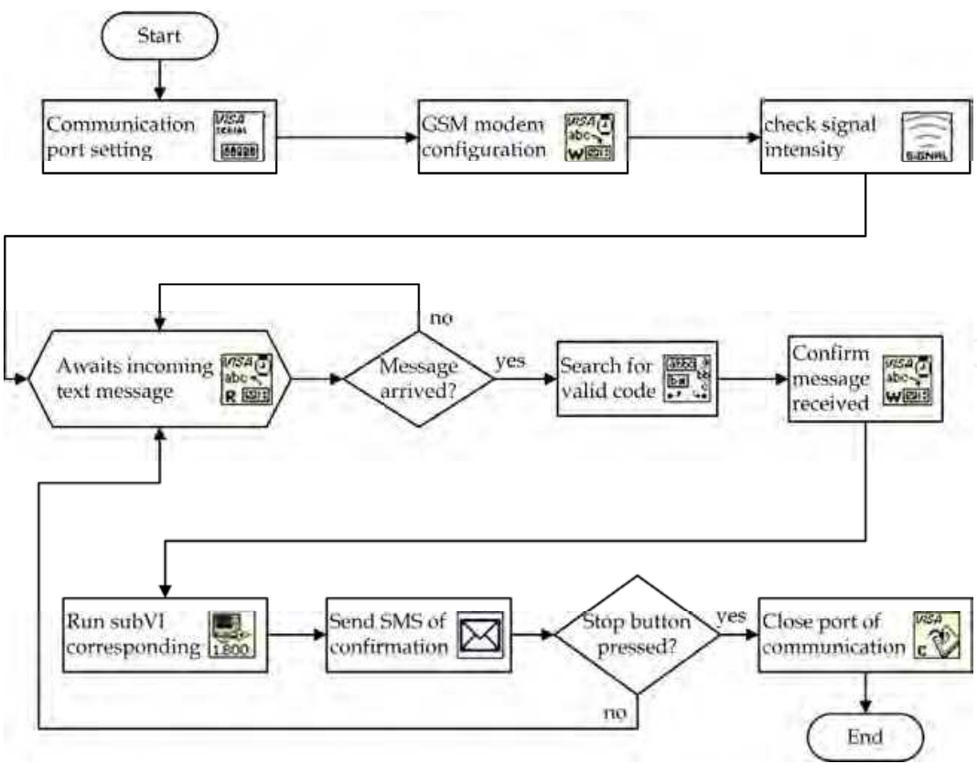


Fig. 17. Complete flowchart of actions performed by the block diagram in LabVIEW.

3.4 Front panel

The front panel of the program developed in LabVIEW is shown in Figure 18, and its resources are explained below.

The graphical programming allows the creation of detailed interfaces and with high iteration levels. However, in this application it is not necessary to provide many resources in the graphical user interface (GUI), because the user will use it only to start the program and after will work in the distance using SMS. The functions available on the front panel designed for this project will be explained according to the numbered blocks showed in Figure 18.

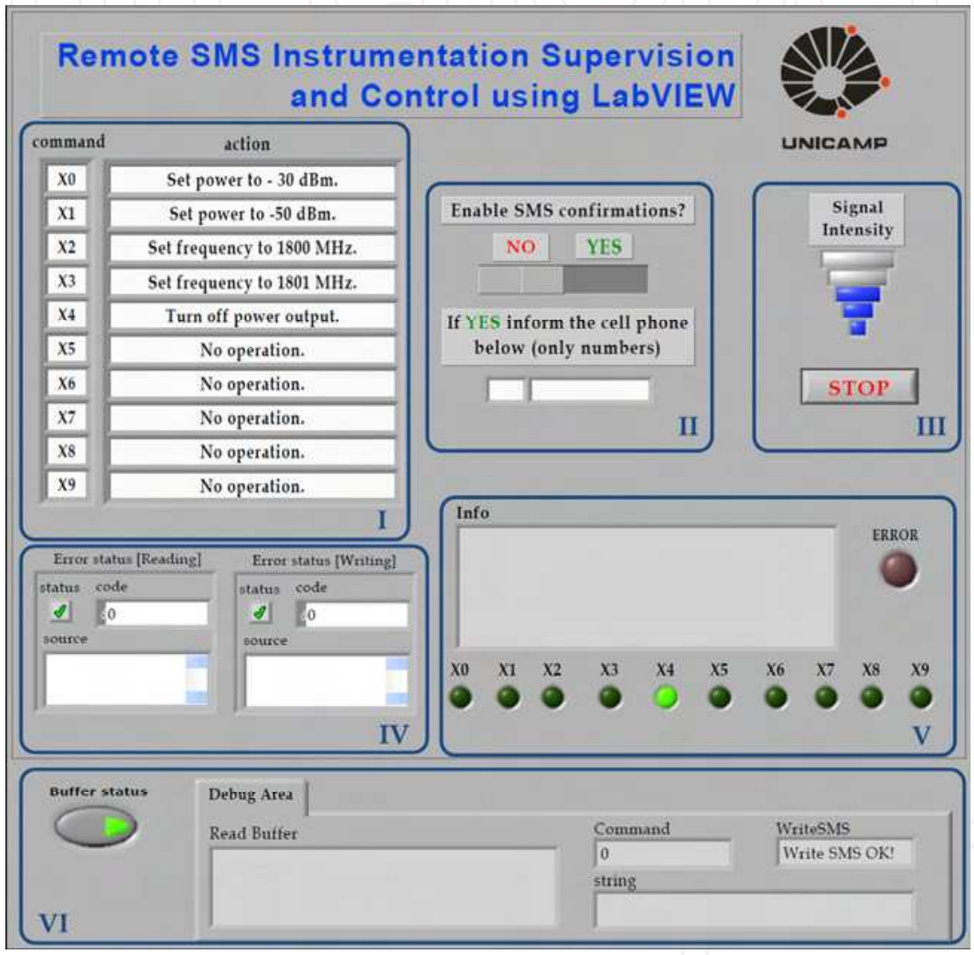


Fig. 18. The application GUI created in the LabVIEW front panel.

Block I is composed of a table created by the arrays "command" and "action". The user should fill the array "action" with a brief description about each code function. Subsequently, the text of these fields will be used to assemble the confirmations messages sent to the user.

In Block II there is the switch that lets the user choose whether to receive or not the confirmation messages after running a requested command. This feature will send the user mobile phone a text message reporting success or failure of the requested action. Therefore, it is necessary to inform the phone number to which the confirmations will be sent. Since the

sending of messages implies in additional charges, the switch was designed to enable or disable this feature.
In Block III there are the signal intensity indicator and a button that closes the program. As explained before, the indicator is processed through a subVI and its operation is similar to that seen in cell phones displays: the stronger the signal, the greater the number of bars. The inactive bars are gray and the active bars are represented in the blue color.
Block IV is formed by the errors indicators. Although there were no failures during the tests performed in this project, errors may occur during the program execution while reading or writing in the serial communication buffer, and the error description is displayed in the corresponding box. Besides, when an error occurs the red LED indicator located in block V is turned on.
Block V is also composed by information about the program execution. In the "info" box appears the text of the confirmation messages, which will be sent the user mobile phone by the GSM modem if the switch of this resource is activated. The indicators X0 to X9 indicate the last command executed, by turning the corresponding LED to green.
The Block VI is a tab with information about the serial communication channel. The buffer content is displayed in the "read buffer" box, and the remaining fields show if a valid code was found in a received message, besides the current string sent to the buffer, and if the subVI for writing and sending messages was performed correctly. This tab information is very useful to the programmer, because allows to analyze the status of the serial communication and locate possible errors with the modem communication. However, it has no great value to the user, so it is possible to hide the tab by using the button on the left.
Thus, the creation of a user-friendly panel with some basic features allows the user to put the program to run and accesses its key information without the need to access the block diagram. It permits to evaluate if everything is fully operational before start to operate the program remotely via SMS. Moreover, such panel can be useful if there is the need to publish a web page to monitor the program via Internet.
Once explained the whole operation and programming of the remote SMS system using LabVIEW, the application will be evaluated in a measurement process described in the next section.

4. Case study

The objective of this case study is to test and evaluate the proposed remote SMS control and monitoring. Hence, the developed system was applied to an 1.8-GHz radio mobile envelope measurement carried out in an urban area. This section will describe the measurement process and how the SMS tool was used in these measurements, followed by the achieved results.

4.1 System and measurement description

In the mobile radio channel, signal propagation frequently takes place in a typical urban environment. In such a situation, there is normally no line-of-sight condition between the transmitter and mobile receiver and multipath propagation is the predominant phenomenon. This fact leads to frequency selectivity, since the frequency response of the channel is no longer flat over all frequencies. One measure of the channel frequency selectivity is the coherence bandwidth. The coherence bandwidth refers to the frequency separation between two signals, which results in a given level of correlation between their

envelope amplitudes. Here, the aim of the measurement process is to characterize the coherence bandwidth between two frequency-spaced radio mobile signals, based on narrow-band 1.8-GHz field trials carried out in an urban open area. The equipment list of the transmitter setup is composed by two signal generator, an amplifier, and a Yagi antenna. On the other side, the receiver setup is composed by a monopole antenna, a preamplifier, a splitter, two spectrum analyzers, a trigger signal, and a laptop with the LabVIEW software installed. The schematic diagram of the measurement systems is shown in Figure 19 (Ribeiro et al., 2007).

In the transmitter, two frequency-separated continuous wave signals (CW) around 1.8 GHz are combined and then amplified to about 30 dBm in power. This signal is transmitted from a 12-dBi vertically polarized Yagi antenna with 33º and 36º E- and H-plane 3-dB beam widths, respectively. The transmitting antenna is located 10 m above the street level on the rooftop of a laboratory building (Ribeiro et al., 2007).

The receiver setup is onboard in a vehicle driven along the local area streets. The signal is received with a 3-dBi monopole antenna, located on the vehicle roof, and then amplified before being split into two branches. Each signal branch is fed to a spectrum analyzer to separately detect and record the envelopes of each frequency. In order to simultaneously record the two frequency-spaced envelopes, the sweep of each analyzer is time-synchronized by the same external trigger signal (Ribeiro et al., 2007).

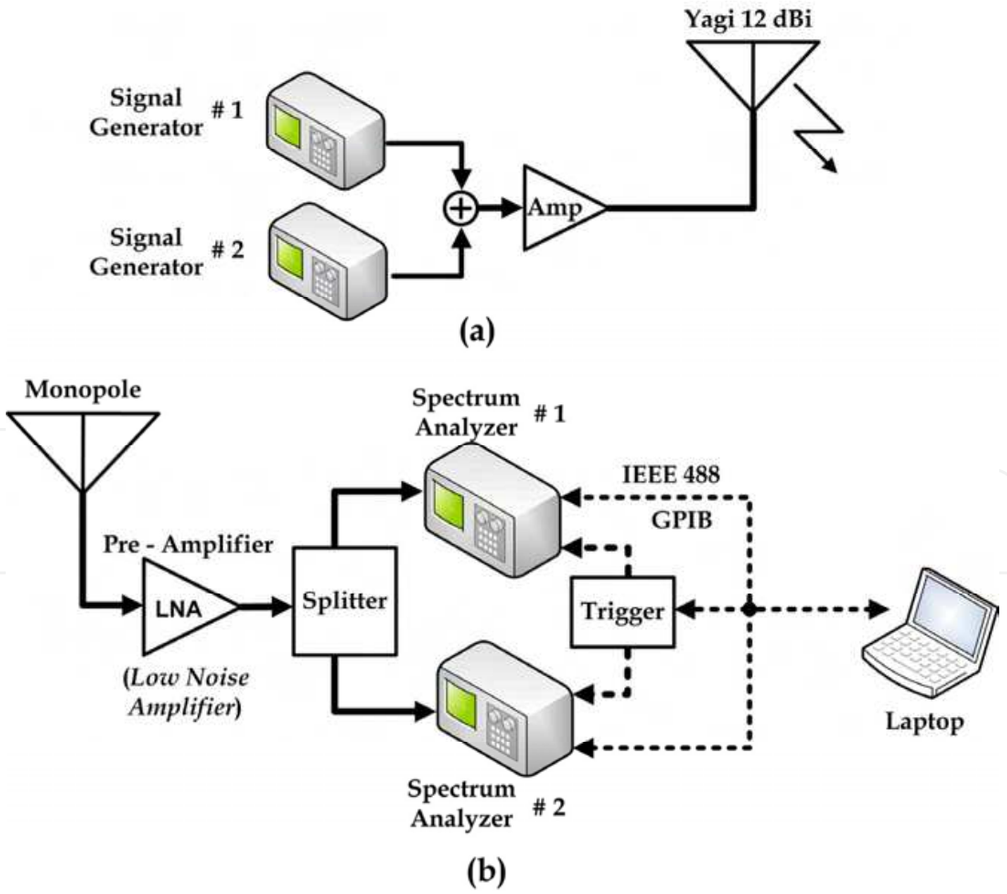


Fig. 19. Measurement system schematic of the (a) transmitter and (b) receiver setup.

The recorded envelopes are stored in a personal computer for latter analysis. In the vehicle, the configuration and control of the equipments, as well the data acquisitions are carried out using programs developed in LabVIEW. The communication between the instruments and the laptop is performed via GPIB using an USB/GPIB interface (Ribeiro et al., 2007).

4.2 Applying the SMS tool in the measurement process

The measurements are acquired with the vehicle in movement at a distance of about 1 km from the transmitter, and frequently it is necessary to change the parameters of one of the generators located at the laboratory. As a consequence, a person need to stay in the laboratory to perform this function or the one who is in the vehicle has to return to there whenever a parameter has to be changed. However, this role can be attributed to the remote SMS instrumentation supervision and control. Hence, the person who is performing the measurements in the vehicle can send a text message by a cellular phone to the GSM modem that transmits the message to the computer with LabVIEW by serial communication, where the message is interpreted and translated in a command, which is finally sent by GPIB to the signal generator. Thus this procedure saves time, since there is no need to return to the laboratory for changing parameters, and there is no need of an additional person in the laboratory (Ribeiro et al., 2007; Figueiredo et al., 2009).

For this purpose, the configuration shown previously in Figure 6 is added to that shown in Figure 19 (a). At this point it is necessary to create a subVI for each signal generator parameter that will be changed and insert it in the appropriate place within the SMS main program, indicated previously in Figure 13. One can argue about the need to make a program for each action, but, making the first one, the next subVIs can be made using this first as model, and no more than one GPIB command needs to be changed, which is a relatively simple task. In one measurement set it is necessary to vary the signal frequency in a specific range, in this case from 1800 MHz to 1801 MHz, in steps of 0.1 MHz. To do this, ten subVIs were designed and placed in the appropriate frame of the case structure, where each subVI sets the frequency in a determined value according to Table 1, which was also written in the program front panel.

The block diagram of a subVI that adjusts the frequency at 1801 MHz is shown as an example in Figure 20.

Code	Action
X0	Set frequency to 1800.1 MHz
X1	Set frequency to 1800.2 MHz
X2	Set frequency to 1800.3 MHz
X3	Set frequency to 1800.4 MHz
X4	Set frequency to 1800.5 MHz
X5	Set frequency to 1800.6 MHz
X6	Set frequency to 1800.7 MHz
X7	Set frequency to 1800.8 MHz
X8	Set frequency to 1800.9 MHz
X9	Set frequency to 1801 MHz

Table 1. Command list to be executed via SMS in the signal generator.

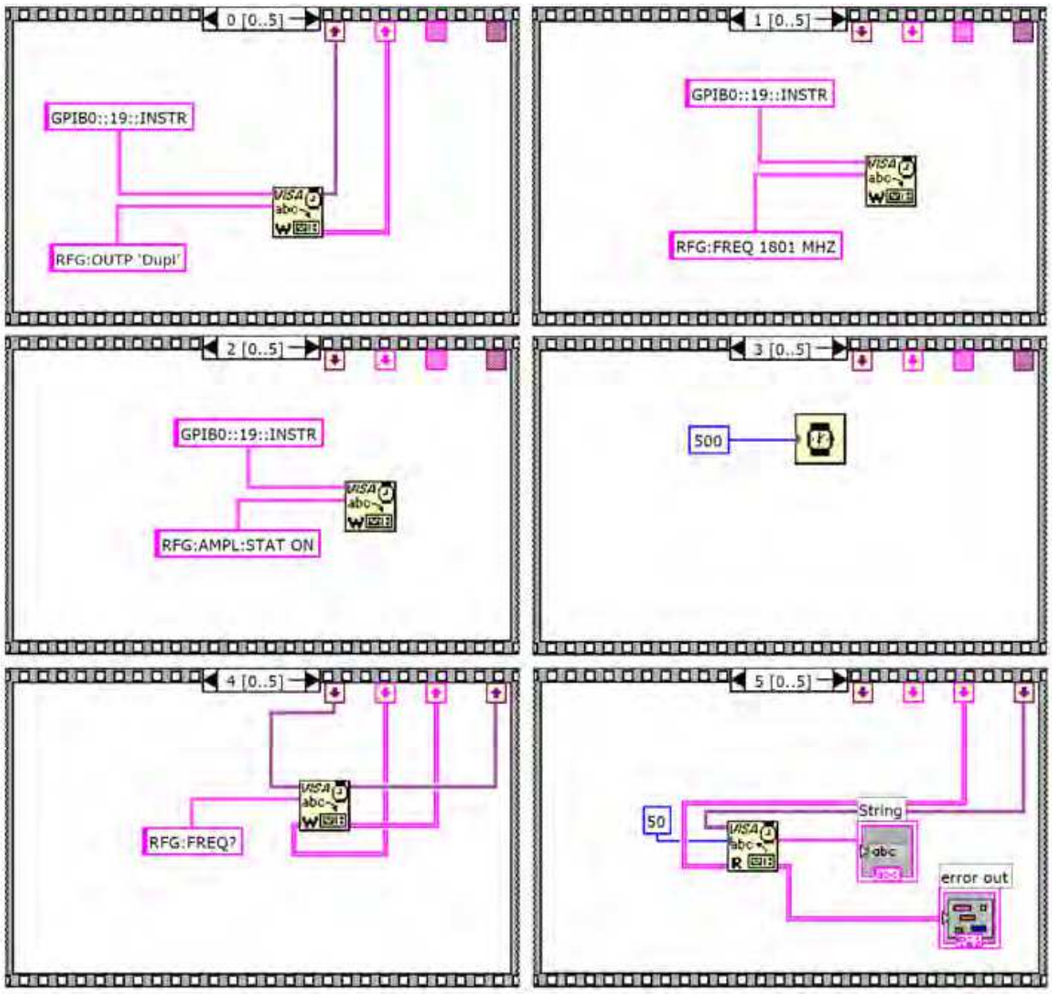


Fig. 20. All frames of the subVI that sets the signal generator frequency at 1801 MHz.

The program was developed in a sequence structure with six frames, saved as a subVI, and inserted in the main program. In the frame 0 the initial instrument configuration is done by setting its RF output port to duplex, the frame 1 sets the frequency; the frame 2 turns on the amplifier output; the frame 3 is just a time delay in the subVI running; the frame 4 queries the current signal generator frequency; and the last frame reads the answer from the instrument. After this first subVI, the other nine were obtained changing the frequency value in the frame 1. In this case, ten subVIs were used, but the number of available options can be increased, as explained previously.

4.3 Results

The first tests were conducted with a simplified setup, entirely mounted within the laboratory. The tests were basically done by sending the commands and examining whether the requests were properly executed by observing the parameter changes in the instruments. To measure the full response time of an execution, the confirmation messages were chosen to be received. Each command showed in Table 1 was sent several times throughout a week, and the execution time was measured from the moment the message with a command was

sent until the moment when the confirmation returns. Thus it was possible to analyze how long the system took to receive, interpret and execute the command, and then assemble and send the confirmation message to the user. Accounting for the delays intentionally inserted into some LabVIEW routines, the execution times were around thirty seconds. Additional tests were performed during times of network congestion and the full response time of an execution did not show large variations (Figueiredo et al., 2009).

Further tests were performed in the vehicle, during the measurement process, in different days and times. In these tests the confirmation messages were disabled, because it is possible to note if the action was executed through the spectrum analyzer screen in the vehicle. As in the previous tests, all requests were correctly executed with times below thirty seconds, since in this case there are no times of confirmation messages included.

In addition to the tests conducted for this case study, preceding tests were executed for controlling and monitoring of car alarm functionalities via SMS, where the results were satisfactory, and with runtimes similar to those obtained here (Figueiredo et al., 2008). Furthermore, measurements conducted by other authors investigated the transmission time and message loss probability, in order to analyze the SMS reliability for emergency warning systems. The measurement results have shown that it is possible to use SMS in such systems, since there was no message loss and almost every message was received within a short time, increasing the trustworthiness of applications based on SMS (Pries et al., 2006).

5. Conclusion

Considering the growth of remote monitoring and control systems and knowing that a large number of such applications can be achieved using simple text messages, this Chapter has presented the feasibility of a flexible and low cost monitoring and control solution using SMS, which can be easily applied and adapted to various applications.

The system is LabVIEW-based and uses standard interfaces for communication. So, it does not require expert programmers to perform adjustments in the program. It requires basically a computer with LabVIEW, and a GSM modem, besides the instruments to be controlled and/or monitored. The access to the system can be carried out using any 2G cell phone without the need to use high cost devices.

This Chapter showed all the details of how to develop the programming framework with detailed instructions of each routine within the main program, which makes easier the task of adapting the system for applications different from that illustrated here.

The developed system was applied to a RF signal procedure measurement saving time and staff in this process. The tool development and its use in a specific application showed the LabVIEW versatility. Furthermore, the results showed that SMS is suitable and reliable for several kinds of applications.

At last, the remote SMS instrumentation supervision and control using LabVIEW allows a wide range of future work, because it is just needed to adapt the program in accordance with the requirements of any application where it is desirable to control or monitor instrumentation remotely, and this Chapter gives the necessary foundation for such work.

6. Acknowledgment

This work was supported in part by the Brazilian agencies FAPESP (*Fundação de Amparo a Pesquisa do Estado de São Paulo*), CAPES (*Coordenação de Aperfeiçoamento de Pessoal de Nível*

Superior), and CNPq (*Conselho Nacional de Desenvolvimento Científico e Tecnológico*), under CEPOF-FAPESP, FOTONICOM and TIDIA-Kyatera programs.

7. References

Alsaialy, S.D.; Tawy, D.M & Lord, S.M. (2003). Introduction to LabVIEW two-part exercise, *Proceedings of the 33rd Annual Frontiers in Education (FIE)*, Vol. 1, pp. T4E-1—6, ISBN 0-7803-7961-6, Nov. 5-8, 2003

Bitter, R.; Mohiuddin, T. & Nawrocki, M.R. (2006). *LabVIEW Advanced Programming Techniques* (2nd edition), CRC Press, ISBN 978-0-8493-3325-5, Boca Raton, FL, USA

Brown, J.; Shipman, B. & Vetter, R. (2007). SMS: The Short Message Service, *Computer*, Vol. 40, No. 12, pp. 106-110, Dec. 2007, ISSN 0018-9162

Figueiredo, R.C.; Arthur, R.; Bonani, L.H. & Arnold, F.J. (2008). Wireless Control System Based on SMS, *Proceedings of IEEE Andean Region IV International Conference (ANDESCON)*, Cusco, Peru, Oct. 15-17, 2008

Figueiredo, R.C.; Ribeiro, A.M.O.; Arthur, R. & Conforti, E. (2009). Remote instrumentation control and monitoring based on LabVIEW and SMS, *Proceedings of the 35th Annual Conference of the IEEE Industrial Electronics (IECON)*, pp. 2477-248, ISBN 978-1-4244-4648-3, Porto, Portugal, Nov. 3-5, 2009

Gallardo, S.; Barrero, F.; Toral, S.L. & Duran, M.J. (2006). eDSPlab: A remote-accessed instrumentation laboratory for digital signal processors training based on the Internet, *Proceedings of the 32nd Annual Conference of the IEEE Industrial Electronics (IECON)*, pp. 4656-4661, ISBN 1-4244-0390-1, Paris, Nov. 6-10, 2006

Gan, W.S. & Kuo, S.M. (2007). *Embedded Signal Processing with the Micro Signal Architecture* (1st edition), Wiley-IEEE Press, ISBN 978-0-471-73841-1, Hoboken, NJ, USA

Garbus, R.U.; Aguirre, I.J.O.; Sanchez, R.C. & Pureco, O.R. (2006). Virtual Remote Lab for Control Practices, *Proceedings of Electronics, Robotics and Automotive Mechanics Conference*, Vol. 2, pp. 361-366, Sept. 26-29, 2006

Glasgow, H.B.; Burkholder, J.M.; Reed, R.E.; Lewitus, A.J. & Kleinman, J.E. (2004). Real-time remote monitoring of water quality: a review of current applications, and advancements in sensor, telemetry, and computing technologies, *Journal of Experimental Marine Biology and Ecology*, Vol. 300, No. 1-2, March 2004, pp. 409-448, ISSN 0022-0981

Guzmán, J.L.; Berenguel, M.; Rodríguez, F. & Dormido, S. (2005). Web-Based Remote Control Laboratory Using a Greenhouse Scale Model, *Computer Applications in Engineering Education*, Vol. 13, No. 2, Jul. 2005, pp. 111-124, ISSN 1099-0542

Higa, M.L.; Tawy, D.M. & Lord, S.M. (2002). An introduction to LabVIEW exercise for an electronics class, *Proceedings of the 32nd Annual Frontiers in Education (FIE)*, Vol. 1, pp. T1D-13—T1D-16, ISBN 0-7803-7444-4, Nov. 6-9, 2002

Jawarkar, N.P.; Ahmed, V. & Thakare, R.D. (2007). Remote Control using Mobile through Spoken Commands, *Proceedings of International Conference on Signal Processing, Communications and Networking (ICSCN)*, pp. 622-625, Feb. 22-24, 2007

Jianting, Z.; Hanmin, H.; Shanglian, H.; Weiming, C.; Zhen, J.; Zhixiang, Z. & Simeng, L. (2006). Remote Real-time Health Monitoring and Evaluation System for Long Bridge Structure, *Proceedings of the Multiconference on Computational Engineering in Systems Applications (IMACS)*, pp. 1751-1755, Oct. 4-6, 2006

Khan, S.; Islam, M.R. & Khalifah, O.O. (2004). Wireless and wired remote measurement unit design and applications, *Proceedings of 39th International Universities Power Engineering Conference (UPEC)*, pp. 1282- 1285, Sept. 6-8, 2004

Lita, I.; Visan, D.A.; Mujea, G. & Ghita, D. (2005). LabVIEW Application for Analysis of Mechanical Vibrations from Industrial Environment, *Proceedings of the 28th International Spring Seminar on the Electronics Technology: Meeting the Challenges of Electronics Technology Progress*, pp. 463-467, ISBN 0-7803-9325-2, May 19-20, 2005

Motorola, Inc. (June 2008). Motorola G24 Developer's Guide - AT Commands Reference Manual, In: *M2M Wireless Modules*, 11.Mar.2011, Available from http://motorola.com/web/Business/Products/M2M%20Wireless%20Modules/G24%20Lite/_Documents/static%20files/AT_Commands_Reference_Manual2.pdf

Oussalah, S. & Djezzar B. (2010). Automated MOS Transistor gamma Degradation Measurements Based on LabVIEW Environment, In: *LabVIEW - Modeling Programming and Simulations*, Riccardo de Asmundis, pp. 163-176, InTech, ISBN 978-953-307-521-1, Rijeka, Croatia

Peersman, G.; Griffiths, P.; Spear, H.; Cvetkovic, S. & Smythe, C. (2000). A tutorial overview of the short message service within GSM, *Computing & Control Engineering Journal*, Vol. 11, No. 2, pp. 79-89, Apr. 2000, ISSN 0956-3385

Pries, R.; Hobfeld, T. & Tran-Gia, P. (2006). On the Suitability of the Short Message Service for Emergency Warning Systems, *Proceedings of the IEEE 63rd Vehicular Technology Conference (VTC)*, pp. 991-995, May 7-10, 2006

Ribeiro, A.M.O.; Castelli, C.S.; Barrientos, E.M.M. & Conforti, E. (2007). Coherence bandwidth in a 1.8-GHz urban mobile radio channel, *Proceedings of Microwave and Optoelectronics Conference (IMOC)*, pp. 599-602, ISBN 978-1-4244-0661-6, Oct. 29 - Nov. 1, 2007

Rosati, R.J. (2009). Evaluation of Remote Monitoring in Home Health Care, *Proceedings of the International Conference on eHealth, Telemedicine, and Social Medicine (eTELEMED)*, pp. 151-153, Feb. 1-7, 2009

Sarram, M.; Ghasemzadeh, M. & Aghaei, V. (2008). Remote Control and Overall Administration of Computer Networks, Using Short Message Service, *Proceedings of the International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA)*, pp. 1-5, April 7-11, 2008

Sgârciu, V. & Stamatescu G. (2010). Distance Process Monitoring using LabVIEW Environment, In: *LabVIEW - Modeling Programming and Simulations*, Riccardo de Asmundis, pp. 67-88, InTech, ISBN 978-953-307-521-1, Rijeka, Croatia

Sukanesh, R.; Gautham, P.; Arunmozhivarman, P.T.; Rajan, S.P. & Vijayprasath, S. (2010). Cellular phone based biomedical system for health care, *Proceedings of the IEEE International Conference on Communication Control and Computing Technologies (ICCCCT),* pp. 550-553, ISBN 978-1-4244-7769-2, Oct. 7-9, 2010

Sumathi, S. & Surekha, P. (2007). *LabVIEW based Advanced Instrumentation Systems* (1st edition), Springer, ISBN 978-3-540-48500-1, New York, NY, USA

Wang, W.; Tse, P.W. & Lee, J. (2007). Remote machine maintenance system through Internet and mobile communication, *The International Journal of Advanced Manufacturing Technology*, Vol. 31, No. 7, Jan. 2007, pp. 783-789, ISSN 0268-3768

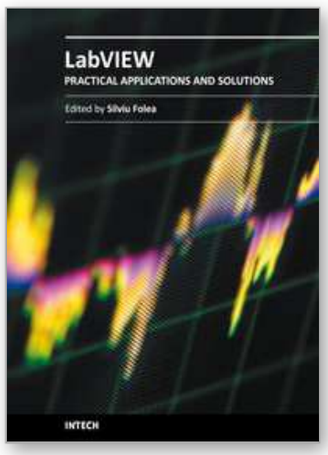
Xiao, J.; Xu, S. & Wu, G. (2009). Monitor System of the Intelligent Power Earth Lines Based on GSM SMS Protocol, *Proceedings of the International Conference on Electronic Measurement & Instruments (ICEMI)*, pp. 3-178-3-181, Aug. 16-19, 2009

Zahariah, M.; Mardiana, B.; Hazura, H.; Fauziyah, S. & Hanim, A.R. (2009). Designing a Low Cost Electronic Devices Switching System Controlled by Short Message Service, *Proceedings of the International Conference on Computer Technology and Development (ICCTD)*, Vol. 2, pp. 292-295, Nov. 13-15, 2009

Zarka, N.; Al-Houshi, I. & Akhkobek, M. (2006). Temperature Control Via SMS, *Proceedings of the International Conference on Information and Communication Technologies: From Theory to Applications (ICTTA)*, Vol. 2, pp. 2678-2683, April 24-28, 2006

Zhang, R.; He, F.; Du, Z. & Sun, L. (2007). An Intelligent Home Environment Inspecting Robot, *Proceedings of the International Conference on Mechatronics and Automation (ICMA)*, pp. 1683-1688, Aug. 5-8, 2007

Zhicong, Q.; Delin, L. & Shunxiang, W. (2008). Analysis and Design of a Mobile Forensic Software System Based on AT Commands, *Proceedings of the IEEE International Symposium on Knowledge Acquisition and Modeling (KAM) Workshop*, pp. 597-600, Dec. 21-22, 2008



Practical Applications and Solutions Using LabVIEW™ Software

Edited by Dr. Silviu Folea

ISBN 978-953-307-650-8
Hard cover, 472 pages
Publisher InTech
Published online 01, August, 2011
Published in print edition August, 2011

The book consists of 21 chapters which present interesting applications implemented using the LabVIEW environment, belonging to several distinct fields such as engineering, fault diagnosis, medicine, remote access laboratory, internet communications, chemistry, physics, etc. The virtual instruments designed and implemented in LabVIEW provide the advantages of being more intuitive, of reducing the implementation time and of being portable. The audience for this book includes PhD students, researchers, engineers and professionals who are interested in finding out new tools developed using LabVIEW. Some chapters present interesting ideas and very detailed solutions which offer the immediate possibility of making fast innovations and of generating better products for the market. The effort made by all the scientists who contributed to editing this book was significant and as a result new and viable applications were presented.

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen