Buletinul Ştiinţific al Universităţii "Politehnica" din Timişoara

Seria ELECTRONICĂ şi TELECOMUNICAŢII
TRANSACTIONS on ELECTRONICS and COMMUNICATIONS

Tom 49(63), Fascicola 2, 2004

Mobile Accident Warning System -The LoRD-

Dan-Marius Dobrea[1], Nicolae Cleju[2], Andrei-Tudor Sechelea[3], Alexandru Banar[4]

Abstract - The Location Retrieval Device (LoRD) is a powerful, versatile and accurate warning system. Designed to be integrated on a car, it instantly reacts in case of an accident, providing the rescue units with extremely useful information about the coordinates, speed of the car, the structural integrity and temperature of the cockpit. The project is based on two-way SMS messaging between the main components: the mobile LoRD modules, that send the information using a mobile phone, and the base station that receives these messages, and also manages and configures the modules.
Keywords: emergency, GSM, GPS, SMS, location retrival

I. INTRODUCTION

One of the major problems with respect to the design and management of any emergency system is the ability to quickly handle any distress call. The faster the better is the main principle that governs the functioning of any emergency service. Only a quick response to fires, car crashes or accidents can diminish the risks of human injuries or fatalities. According to the US Bureau of Transportation Statistics, car crashes are an important issue of public safety around the world, with a number of 42,815 fatalities and 2,9[illegible]5,758 injured people in 2002, in the US [1]. The victim's chances of survival can be greatly improved with an accurate response, implying that the paramedic units should be notified as soon as possible about the precise location where an accident took place. The lack of precision means wasting extremely precious time.

We set to develop a system that reduces and almost completely excludes wasted time in a distress call. It helps rescue teams locate the victims easier and faster. Our system's main goals are:

- ✓ To instantly notify a central base station when an accident took place,
- ✓ To provide the coordinates of the location, so the emergency unit can reach it in minimum time,
- ✓ To gather other crucial information from the site of the accident: the temperature (in order to detect a possible fire) and the structural integrity of the car,
- ✓ To continue to periodically send as much data as possible after the accident took place, providing a real-time evolution of the situation.

II. SYSTEM OVERVIEW

II.1. The way it works

The overall system is composed of mobile LoRD (Location Retrieval Device) units located in cars and a central base station that receives all the emergency notifications. They communicate by means of SMS (Short Message Service) messages. The functioning of the system is presented schematically in Fig.1.

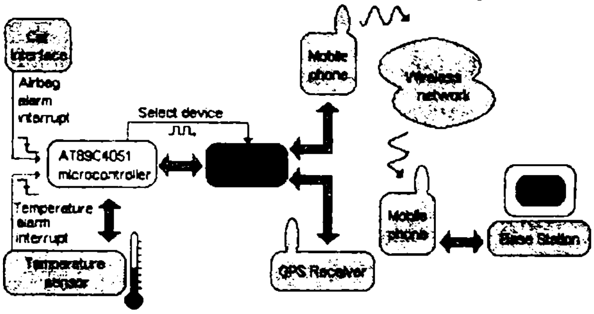

Fig. 1. The block scheme (simplified) of the entire system

[1] Technical University "Gh. Asachi", Faculty of Electronics and Telecommunications, Iasi, Romania, e-mail: mdobrea@etc.tuiasi.ro
[2] Student, Technical University "Gh. Asachi", Faculty of Electronics and Telecommunications, Iasi, Romania, e-mail: nikcleju@gmail.com
[3] Student, Technical University "Gh. Asachi", Faculty of Electronics and Telecommunications, Iasi, Romania, e-mail: astek@personal.ro
[4] Student, Technical University "Gh. Asachi", Faculty of Electronics and Telecommunications, Iasi, Romania, e-mail: alex584@apropo.ro

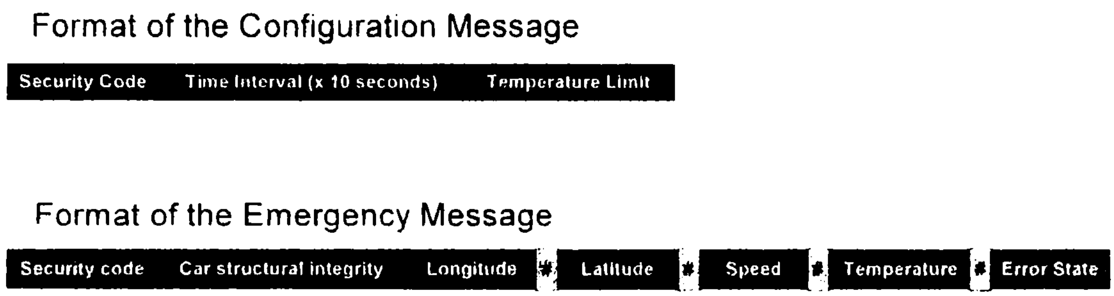

Fig. 2 Format of the most important messages

Before using any LoRD module, it must be remotely configured by the central base station. This is done by sending a message to it that contains the necessary information: the phone number to which the LoRD will reply, the time interval between two consecutive messages and the temperature limit which will trigger the thermal alarm. This process is allowed only if one specific jumper is set. After a successful configuration, unsetting this jumper causes the LoRD to reject any other configuration attempt.

In normal mode, the LoRD module continuously monitors for a possible abnormal situation. The events that trigger the alarm are the car security systems (opening of the airbags, car alarm, or panic button) or the detection of a fire, when the temperature exceeds the previously programmed limit. Therefore the LoRD gathers information about the car status and location and uses the attached mobile phone to send it through a SMS to the base station. The message contains the coordinates of the car, obtained with a GPS Receiver, the temperature, in order to detect a possible fire, and information about the extent of the cockpit damage, Fig.2. If it is not damaged, the LoRD continues to send messages periodically, allowing the base station to monitor the evolution of the situation.

There is also a remote control mode, as an alternative function of the LoRD. The base station may send a request to receive alert messages, although no accident was reported. This can be useful, for example, in tracking down the LoRD in scenarios like mountain rescues, fire-fighting or retrieving stolen vehicles. The base station sends a "START" message, which will enable the LoRD to send back notifying messages. The analogue command "STOP" will disable any message sending.

The central station runs the software that manages all the incoming messages. It is developed in the LabWindows™ /CVI version 5.5 integrated environment and it uses a dedicated database server to securely store the information. Upon receiving an emergency message, the software reads it from the attached mobile phone and presents it to the human operator.

II.2. Security and reliability features

As the system handles critical data, the performance requirements are very high, and we had to consider reliability and security issues from the early stages of the project.

The module performs numerous self-tests to ensure the reliability of the information supplied. Any error detected is reported in the SMS sent to the base. Any phone error is visually signaled to the driver.

To ensure a prompt reaction in case of an accident the alarm signals and the entire serial communication are processed through the microcontroller's system of interrupts. The alarm signals have the highest priority, so as soon as a collision occurs the LoRD begins to send the emergency messages.

The SMS messages by which the LoRD modules and the base station communicate with each other contain a certain security code, computed from the sender's own number. By checking this security code against the sender's phone number, as it is reported by the phone network, the receiving device is able to detect and reject fake messages.

The whole system is designed to be infrastructure-independent in order to have the highest level of generality and to be easy extendable. The communication process is independent of the type of phone network, because all of the GSM, CDMA and TDMA technologies support SMS technology. The LoRD module, the mobile phone, and the GPS Receiver have their own power supplies, which are switched on only in case of a crash; the rest of the time they are powered by the car battery.

Using the dedicated MySQL database server adds to the overall independence and security. The server can reside on a remote host (MS Windows, Unix etc.), and communicate through SSL secure TCP/IP protocols. We decided upon a database server because its security, platform-independence and the capacity to handle and store data with minimal risks of failure make it an excellent choice.

III. HARDWARE IMPLEMENTATION OF THE LORD MODULE

The LoRD was designed around an Atmel™ microcontroller of the 8051 family. We decided upon this family as it is commonly used and has proven very reliable since its development in the 80's.

There are three main interconnecting subunits: the AT89C4051 microcontroller, the BU4066B Quad Bilateral Switch and the MAX232N dual RS-232 Transmitter/Receiver.

The AT89C4051 microcontroller is a low voltage, high performance, 8-bit CMOS microcomputer with 4K bytes of Flash programmable and erasable read-only memory – PEROM. Because of its useful built-in features, high flexibility and extremely high reliability, it was the best cost-effective solution we could find for our LoRD module, fully meeting our requirements.

The MAX232N features two pairs of drivers and receivers for communication through the RS-232 serial interface. It transmits and receives data from two different devices that communicate through the serial interface: the mobile phone and the GPS receiver.

The BU4066B is a multiplexing circuit, which handles data from the MAX232N, deciding on which of the two devices (phone or GPS) sends and receives data from the microcontroller.

The LoRD also features a LM7805 voltage converter circuit. The LM7805 is powered either by a 12V-car battery or by a 9V-accumulator, providing that the car battery is damaged in the accident. The voltage output is regulated at the +5V level required by the digital circuits on the LoRD module.

A 2-wire I^2C bus interface is integrated in the LoRD, ensuring the communication between the microcontroller and the DS1629 digital thermometer [2] . The DS1629 provides 9-bit temperature readings, indicating the temperature of the device. It works as a "temperature-to-digital" converter in the range of -55°C to +125°C (-67°F to 257°F). The circuit also features a thermal alarm, which sets a particular pin to low state, consequently activating an interrupt request in the microcontroller, whenever the temperature of the device exceeds the programmed over-temperature limit stored in a special function register (TH).

An external connector is interfaced to the car security systems (airbag opening command, car alarm and panic button) so that the system is activated whenever an unwanted event takes place. For this particular connection a RC filter is necessary to eliminate the noises that appear due to the car motor. A four-stroke gasoline engine generates parasitic spikes of maximum 10 µs with the frequency given by the formula:

$$f = a \cdot z \cdot \frac{n}{60} \qquad (1)$$

where a is the number of cylinders, z is a constant with the value of 0,5 and n is the rpm (revolutions per minute) number, ranging, for an average car, from 900 to 7500. These spikes would generate system activation if the RC filter wouldn't clear the line. Solving for the worst scenario (n = 7500) we get a minimum 4 ms period between the parasite spikes. This value is used to calculate the required values for the resistor and the capacitor, ensuring good filtering without distorting the waveforms in normal functioning.

Finally, six switches directly connected to some of the microcontroller inputs are to be placed on the car cockpit key components. This makes it easy to assess the amount of damage caused by an accident because the message sent by the LoRD also contains the status of the switches, damaged ones appearing as closed contacts. So by sending their state of these switches, the rescue teams immediately know the full extent of the accident and are able to swiftly act accordingly.

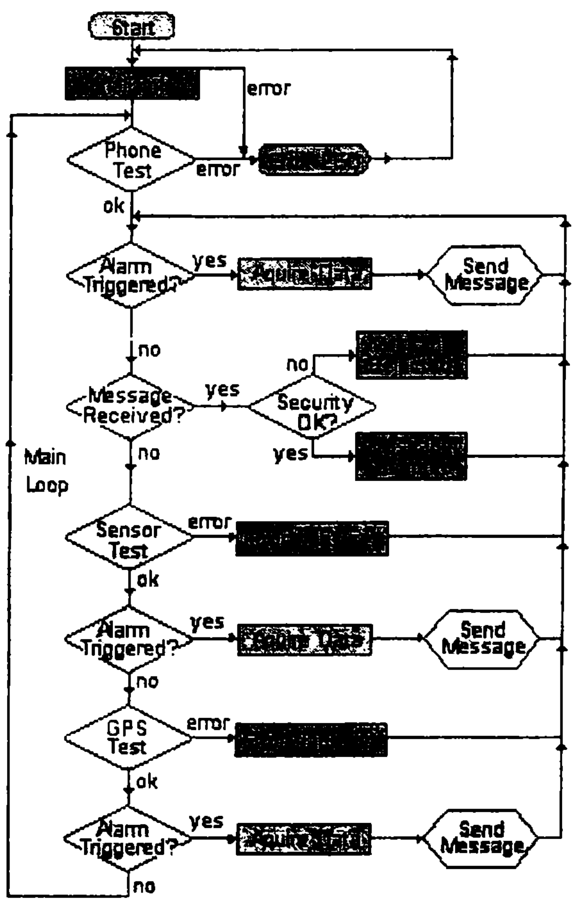

Fig. 3. Main program organization

IV. THE SOFTWARE OF THE LORD MODULE

IV.1. Overview

The microcontroller software was developed in assembly language, as it is faster than a similar program developed in a C environment. The program flow is represented in Fig.3.

Upon startup, the LoRD runs an initialization routine, and then it normally remains in a loop where it performs various self-tests and component-tests. Whenever an external interrupt is activated, the hardware interrupt routines are executed and they set an acknowledgement semaphore. The software verifies the status of these acknowledge flags and if an alarm is indicated it runs the message sending routine.

IV.2. The Initialization Routine

The initialization routine starts with a self-test – it erases all the locations in the internal memory and verifies if they are valid. Then it enables internal timers, interrupts, serial communication and initializes external devices.

The AT89C4051 microcontroller has two timers. One of them, TIMER 0, is used to set the time interval between two consecutive messages, and the other one, TIMER 1, must set the default 9600 bps baud rate used by the microcontroller to communicate through the serial port.

To ensure the priority of alarm operations, we used all of the microcontrollers interrupts in the following way:

- Two external interrupts: they have the highest priority [3], and the activation of any of them will determine the microcontroller to send alarm messages.
- Two timer interrupts [3]: they increment some time counters each time the interrupts are activated;
- The serial interrupt: it has the lowest priority and it is activated each time a character must be sent or received [3].

IV.3. Testing Routines

While it remains in a continuous loop, the microcontroller performs periodical tests to check the system integrity:

- It checks for the presence of a mobile phone by sending the Hayes "AT" command and verifying that the answer received was the expected "OK" (see section "*IV.5. Serial communication routines*"). This test is the most important of all, because without the phone the LoRD cannot send any emergency message. That is why any phone error is visually signaled using a LED.
- It verifies the temperature sensor by writing a certain bit pattern to a location in its memory. Receiving acknowledge bits assures of the proper functioning of the I^2C bus, and the correct retrieval of the pattern validates the sensor circuit itself.
- It verifies the presence of the GPS Receiver by polling commands [4] that ensure the device is connected and working properly, as it must have acquired at least three satellites in order to be able to function. In case the Receiver acquires no signal, it returns null instead of the coordinates, in which case an error flag is set.

This continuous testing allows the fastest response in case of an emergency. Whenever an external interrupt is activated, signaling an emergency, the LoRD does not waste any more time on tests. The current state of the device is already determined, so it can start sending the message as soon as possible.

IV.4. Processing Hardware Interrupts

There are two causes that will determine the microcontroller to enter the alarm state: either a signal from the car interface (the opening of the airbags or other alarm systems) activates External Interrupt 0, or the temperature exceeds a critical level, activating the External Interrupt 1.

We used the entire interrupt system in a way that enables the LoRD to react as promptly as possible to these situations.

We had to analyze how to obtain the fastest response to an emergency, still doing all the necessary tests in order not to compromise the reliability degree of the provided information. (i.e. the LoRD should detect and report any temperature sensor error, because the base station should know that the temperature readings may be incorrect). The following issue, which cannot be avoided, is very important: if the interrupt is requested during any serial communication, the communication process must be allowed to finish. Sending the emergency SMS requires serial communication with the phone, so simply cutting off the previous communication may issue scrambled commands to the phone or the GPS receiver. Our solution is presented in Fig.4.

When an interrupt takes place during a test that involves serial communication with a device, the interrupt routine is executed but it only sets an acknowledge semaphore. The software is therefore allowed do terminate the current communication process. After every test, the program checks these semaphores and, if it determines that an interrupt took place, runs the routine which sends the emergency message. This way, the maximum possible delay between signaling an emergency and the actual start

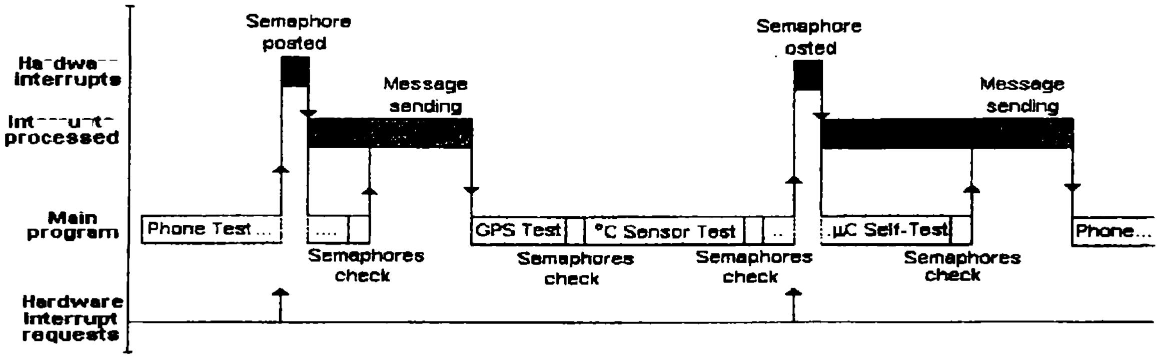

Fig. 4. Processing the interrupt requests

of the message sending routine is the maximum time spent in a testing subroutine.

The other alternative means doing all the tests after the interrupt request, to ensure the reliability of the information. This implies that the delay until the sending of the message would be the sum of all tests' durations. This clearly increases the response time and hence the risks of failure during a crash.

IV.5. Serial Communication Routines

The communication with the mobile phone is based on the assumption that the phone supports the 'Hayes' commands. These are ASCII commands that are transmitted over a RS232 connection and allow using the phone's capabilities. We were interested mostly in the ability to send and receive SMS messages, but also to read phone numbers stored in the SIM card. Sending and reading a SMS message also implies converting the text to and from a serial format, the PDU (Protocol Description Unit) format, which the phone uses.

The communication process with the GPS Receiver is carried out much in the same way as with the phone. In fact, it is even simpler to implement. The Receiver responds to various control NMEA (National Marine Electronics Association) sentences by returning strings of characters containing the required data. These are filtered and the necessary information is extracted.

IV.6. Temperature Sensor Routines

The message transmitted to the base station also contains a temperature reading provided by the DS1629 circuit. The command to read the temperature and the temperature returned by the DS1629 are transmitted through a two-wire serial bus – the I^2C bus.

V. THE BASE STATION

V.1. Overview

The base station runs the software that monitors messages received from the LoRD units. It also allows easy configuration of the new units and reconfiguration of the existing ones. The software controls one mobile phone attached to the serial port of the PC, which is used to send and receive messages. Created in the LabWindows™ /CVI version 5.5 integrated environment, the software displays information about received messages, while in the background it handles receiving and checking the arrived messages, extracting critical information from them and inserting it into a database. The human operator is asked to assign the best rescue unit available to answer the distress call.

The database server that stores and manages the information is the MySQL 4.0.18 server. The software communicates with the server using the MySQL C API, sending SQL commands for retrieving or inserting data.

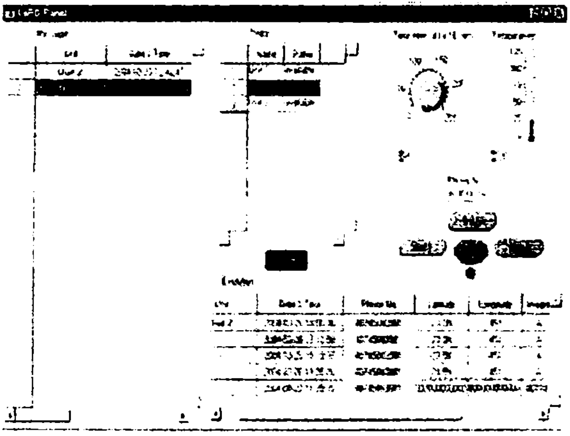

Fig. 5. Main application interface

The main window of the application, shown in Fig.5, displays everything the human operator needs to see in a simple and clear manner. It shows the received messages and their evolution, the rescue teams and their status, and easily allows assigning one or more rescue teams to one of the distress messages. It lets the operator configure and reconfigure LoRD units and also use the remote control mode by sending "START" or "STOP" command messages.

V.2. Communication with the phone

This is accomplished using the 'Hayes' commands that the phone supports. On startup, the software opens the serial port of the PC and checks if a phone is present. If so, it attempts to do a necessary configuration of the phone. This essentially tells it to send a short notifying line of text over the serial connection whenever a new SMS message is received. Then the software enters an idle state, when it permanently checks the phone's status while monitoring both the serial port for notification alerts and the interface for the human operator's actions.

When the phone receives a SMS from the network, it sends a notifying text over the serial connection. Then the software obtains the message after a series of stages, which include decoding it from the PDU format and checking the security bytes.

To send a SMS, the software uses the same process mentioned above, only reversing the stages. When the human operator wants to configure / reconfigure a LoRD unit or to send a start / stop command to it, the software sends the necessary message to the unit, containing the configuration options or the command.

V.3. Communication with the database server

To ensure proper storing and handling of the critical emergency information, as well as fast access and sorting, we decided to use a dedicated database server. Using an existing database server ensures a very low rate of possible failures, as its strong and weak spots are already known and the server is used by many other applications and web services.

Another issue we were interested in was the security of handling and storing the data. The server uses encrypted passwords and has a powerful and versatile security scheme, as it can restrict access depending on the user, the host the user logs from and the user privileges. It also has support for SSL secure TCP/IP connections, although in our project we do not use them. In conclusion, the security functions available are sufficient even for an industrial implementation with very strict security requirements.

We used the available C API to interact with the server. The API provides C functions that allow access to everything the server is capable of.

VI. SUMMARY

In this paper we present a general overview of the system we have developed, as well as provide detailed information about its functioning parameters.
The LoRD is designed to meet all requirements an emergency system has. Providing important, real-time data about the site of any kind of accident, the warning system we built instantly notifies the rescue units of the extent of the danger someone could be in. It also has many built-in security and user-friendly capabilities.

Our system was built to prove the fundamental concept of our project: gathering remote information and sending it to a base station the instant an accident occurs. Further developments, like integrating more different types of sensors – car integrity indicators, smoke detectors etc. – may be carried out. In the end, it is even possible to fully embed the LoRD module into the car computer system using phone and GPS OEM modules.

Amateur hikers, mountain rescue units, firefighters on a mission, all of them can benefit from using our LoRD module without many modifications. Traffic jams and chain collisions can also be detected early and partially avoided. Furthermore, the remote activation mode supports tracing applications like retrieving stolen vehicles, detecting cars exceeding speed limits using the facilities of the GPS, or tracking people who may be lost. The social usefulness of the LoRD system is based on the large number of different applications that call for such an emergency management system, being implied by the potential to minimize injury extent and to save human lives.

REFERENCES

[1] Web Site of The US National Bureau of Transportation http://www.bts.gov/publications/national_transportation_statistics/2003/html/table_02_01.html (road safety statistics)

[2] DS1629 Digital Thermometer Data Sheet, Maxim Semiconductors, http://www.maxim-ic.com

[3] 80C51- Based 8-Bit Microcontrollers, Phillips Semiconductors

[4] B. Hofmann-Wellenhof, H. Lichtenegger, J. Collins, "*GPS Theory and Practice*", Springer WienNewYork, 1992