COMPUTER BASED REMOTE CONTROL FOR LAYOUT OF SCALED MODEL TRAINS

Ivan Ivanov Monov, Ivan Petrov Gorchev

Technical University - Sofia, 8 Kliment Ohridski Blvd., Sofia 1000, Bulgaria,
phone: +359 887 204488, +359 886 796456
e-mail: ivanmonov@yahoo.com, ivan.gorchev@gmail.com

For the realization of the project a special device was made together with a software application for remote control of a set up electric layout along which scaled model trains move. This program sends commands to the device which controls the layout.

The device is designed to work with 8051 based microcontroller and controls the direction of movement, the speed as well as separate element of the layout (switches, signals, etc.).

The program is developed in C++ and used for visualizing the layout as well as for visualizing and changing of the variable elements along the layout at a certain moment.

Serial port for communication was used for exchanging data between the program and the device.

The project was realized and presented at the technical exhibition "Expo Intellect" at the Technical University of Sofia in November 2004 and was awarded special prizes from the Bulgarian Industrial Association and Association "Bulgaria".

Keywords: computer, remote, control, trains, layout

1. SOFTWARE APPLICATION

1.1 Basic Functions

The software application works in two modes – a mode of editing and a mode of control of the layout. In the editing mode elements of the layout may be added, changed or deleted. In the control mode by means of clicking on a certain element of the layout it may change its state, and by sending a command to the hardware device, it may change the state of the responding element of the model layout. On the outlined scheme of the layout, as in a real controller's station, every element which can be controlled is indexed. The user may change the size of the working grid, the speed and the direction of the train's movement along the layout.

1.2 Realization

For the implementation of the program a multitude of separate modules was created. The main program links to them with the aim to facilitate the development of the necessary algorithms and the reduction of the amount of written code. By using the so called "multiuse instancing" memory and disk space are saved.

A document-view architecture supplied by MFC (Microsoft Foundation Classes) is used. In this way the data in the document is separated from the view which the user gets about it.

The developed program product is an MFC SDI (Single Document Interface) application which uses as a main class of the view CScrollView. This enables the

user to move across the screen by means of scroll bars. The main part of the application is concentrated in four classes – class of the application, class on the main frame, class of the view and class of the document.

The code of the base document class interacts with the menu-elements for work with files (FileNew, FileOpen, FileSave) and the derivative document class does the reading itself and the writing of the document object's data. The base view class is a window contained in the window of the main frame. The derivative view class interacts with the associated to it document class and performs the input-output operations of the application for print on the screen and to the printer. The derivative view class and its base classes process the Windows messages. The MFC libraries dictate all interactions between the documents, views, windows of frames and object-application, mainly by virtual functions.

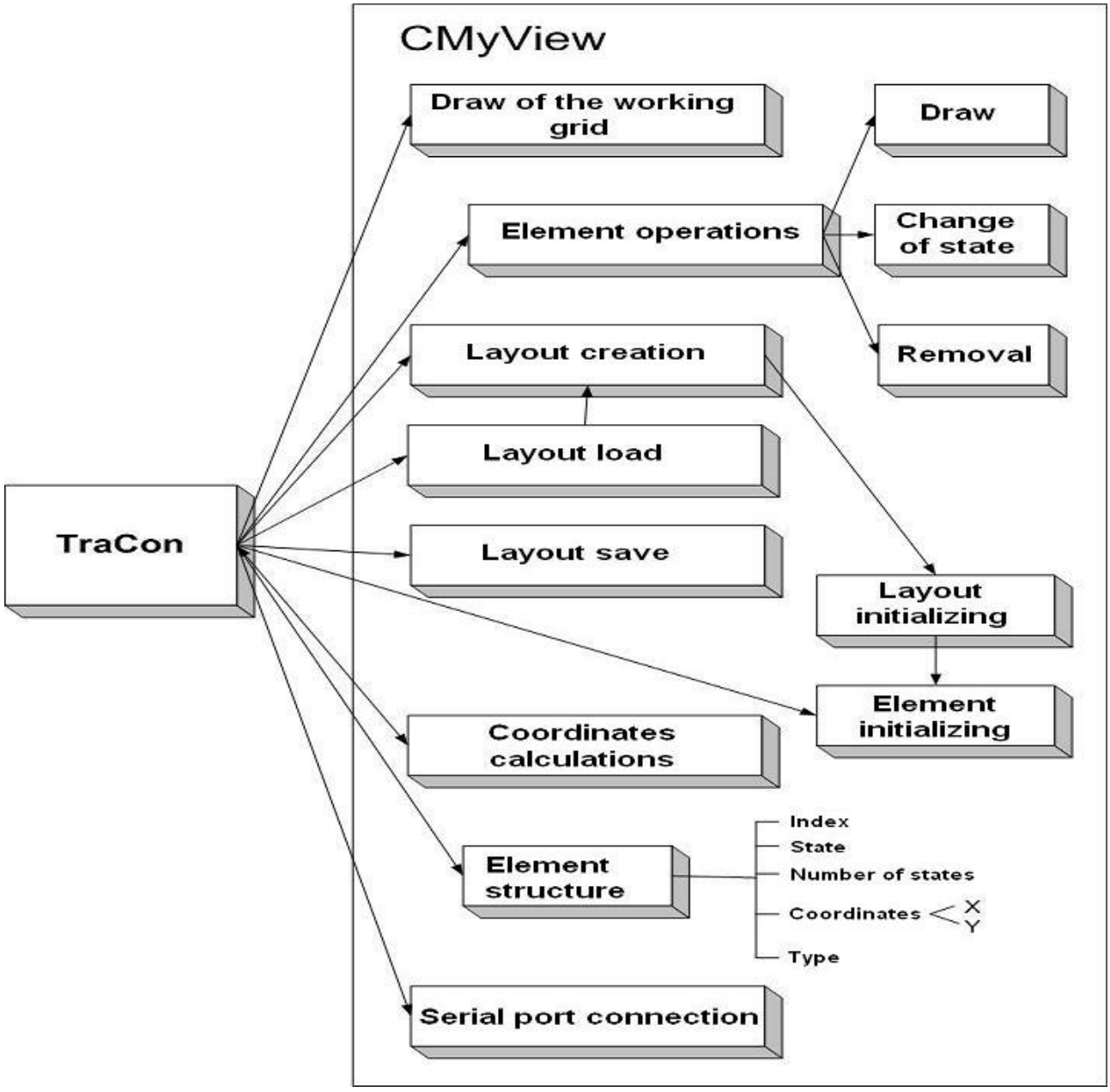


Fig. 1 Structure of the view class.

Each dialog in the application has its own class in which all the information about it is contained – the controls which it has, the variables to them, the functions for processing of certain events as well as all the necessary functionalism.

In different cases the dialogs may be loaded modally (e.g. the Splash screen dialog) where the user cannot return and use the main view until he hasn't closed the respective modal dialog. The other way of showing dialogs is non-modal where the user may work simultaneously in the window of the dialog and in the main view (e.g. the dialog of the speed control and the direction of movement).

The application stores it's settings in the system registry and at each run they are loaded from it and applied.

2. HARDWARE DEVICE

2.1. Design Description and Basic Functions

The purpose of the electronic device is to allow the user to control a scaled train model in real time via PC. The connection between the PC and the device is via serial RS-232 port. The speed is 9600 bits per second, 8 data bits, one stop bit without parity control. The electronic device can manage 128 elements of the layout like switches and signals separated in 16 zones with 8 elements in every zone, the direction of movement – forward of backward, 8 steps of the speed of movement of the composition – from zero to full speed with smooth acceleration. The device is designed as one Main Unit and Extension Units. The main unit controls the speed and the direction of motion. There are 16 Extension Units, each of them controls 8 elements of the layout.

2.2. Block Structure of the Electronic Device

Figure 2 shows the Block structure of the Main electronic device.

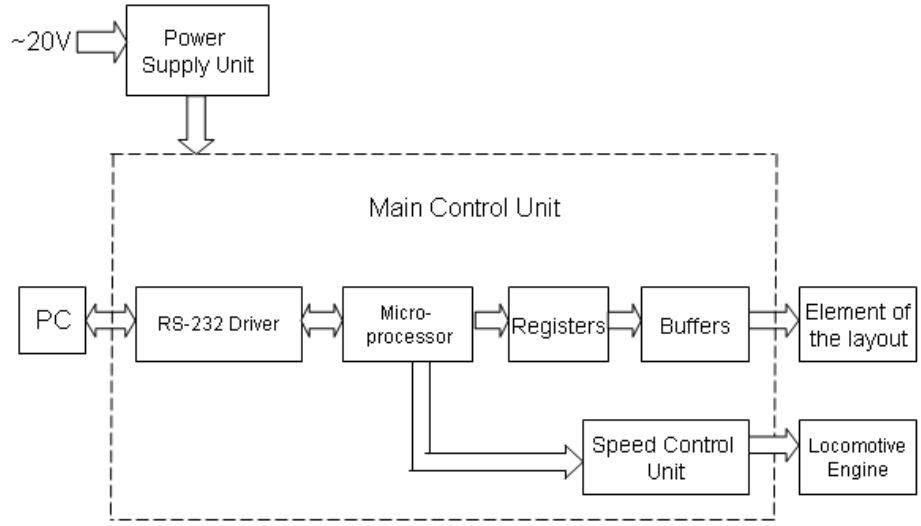


Fig. 2 Block Structure of the Electronic Device.

2.3. Block Description

The ***Power Supply Unit*** comprises a transformer, a rectifier and two switching power regulators. The transformer is the original one from the PIKO Train model

sets. The Power Supply Unit provides +5V for the integrated circuits and +18V for the switches and the signals.

The ***RS-232 Driver*** is based on MAX232 IC. Its purpose is to convert the specific levels of RS-232C signal to TTL compatible levels and vice versa.

The electronic device is based on AT89C52 microcontroller, manufactured by ATMEL with Intel 8051 core. Depending on the commands received from the PC the microcontroller generates signals which control the elements of the layout and the speed. Received commands are separated in two classes – speed control commands or layout element control commands. The microcontroller keeps information for the condition of each element of the layout in an array in its internal memory.

Block ***Registers*** contain registers for the layout condition. The information written is a copy of the information stored in the microcontroller memory. When the condition of element is changed the microcontroller rewrites the contents of the corresponding register.

Block ***Buffers*** is based on MOS transistors. Its purpose is to control the elements of the layout with high current consumption – switches and signals. The peak current is almost 1.5A in transient moments. The active output level is 0. There are a couple of transistors for each buffer.

The ***Speed Control Unit*** is based on a switching power regulator. The speed is controlled by the voltage of the locomotive engine. When the feedback factor is changed the output voltage changes too. There are 8 fixed feedback steps. There are 8 corresponding speeds of motion. The motion direction changes with changing of the locomotive engine voltage polarity. The polarity is changed by a relay.

2.4. Firmware

The Firmware is written in Assembler for Intel 8051. The purpose is maximum optimization and optimal usage of the microcontroller resources – ROM, Flash, RAM. Figure 3 shows the block structure of the Firmware and the format of the PC commands. The purpose of the ***Initializing*** block is to initialize the device after Reset. The main loop contains PC command waiting, command recognition, execution and reply. The commands are two types – commands for speed control and commands for element control. The type of command depends on the MSB (Most Significant Bit) - $S/\overline{E}$. If the MSB is set ($S/\overline{E}$=1), the command is for speed control and the ***Speed Control*** procedure runs. The 3 LSBs (Least Significant Bit) indicate the number of the speed – from 0 to 7. The MSB of the less significant nibble (D) shows the direction of motion - forward if D=1 and backward if D=0. If the MSB is reset ($S/\overline{E}$=0), the command is for element control. The ***Element Control*** procedure runs. The 4 bits of the less significant nibble indicate the zone where the element is located – from 0 to 15, the 3 LSBs of the most significant nibble indicate the index of the element in the zone – from 0 to 7. After the procedure is executed a ***Reply*** is sent to the PC. This ***Reply*** is the code of the executed command.

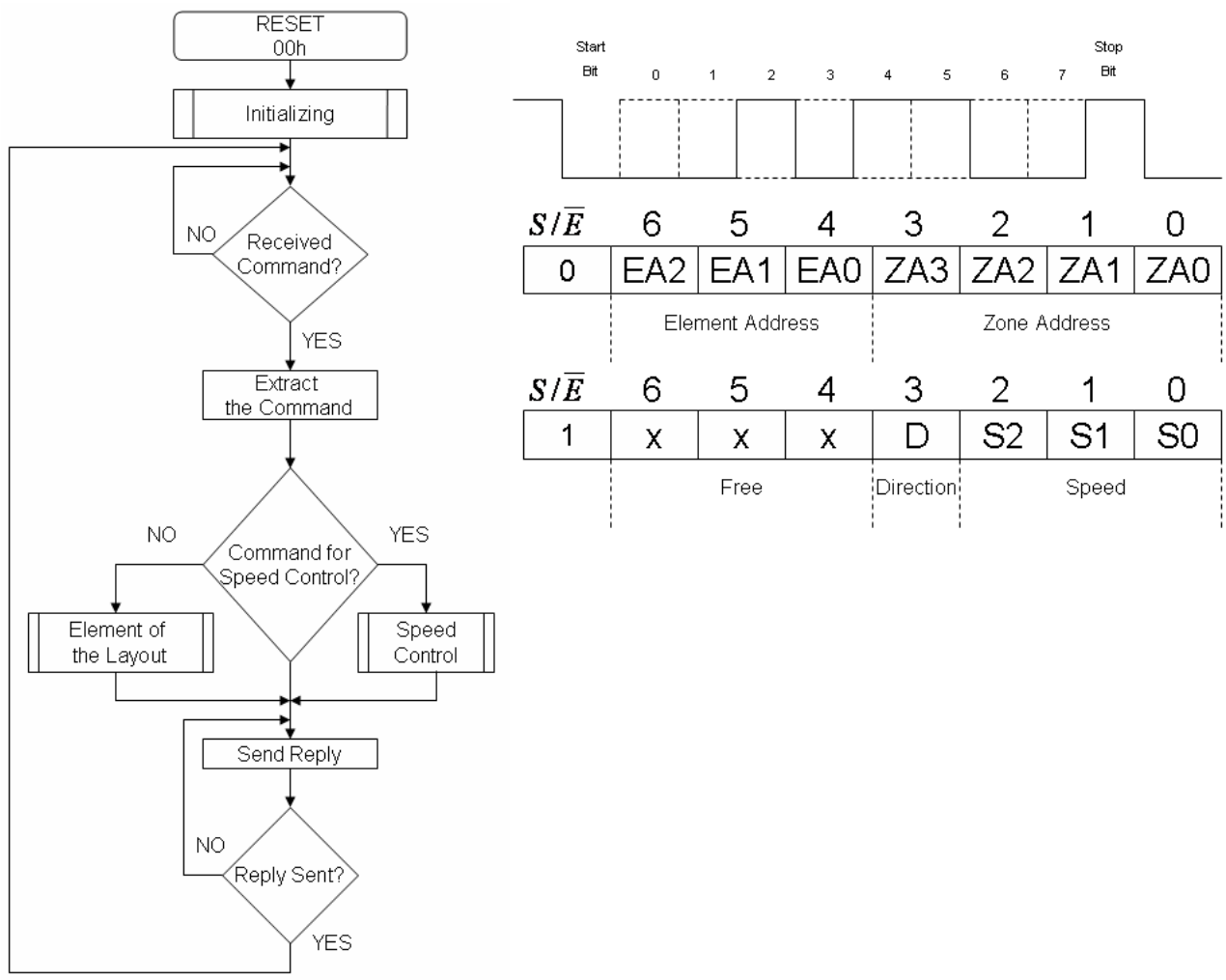
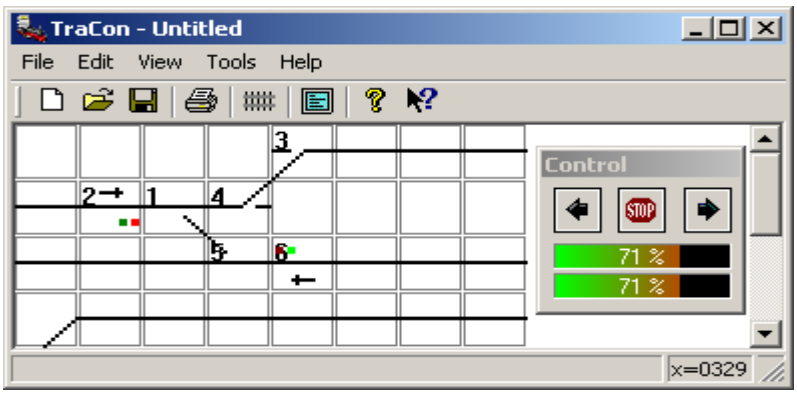


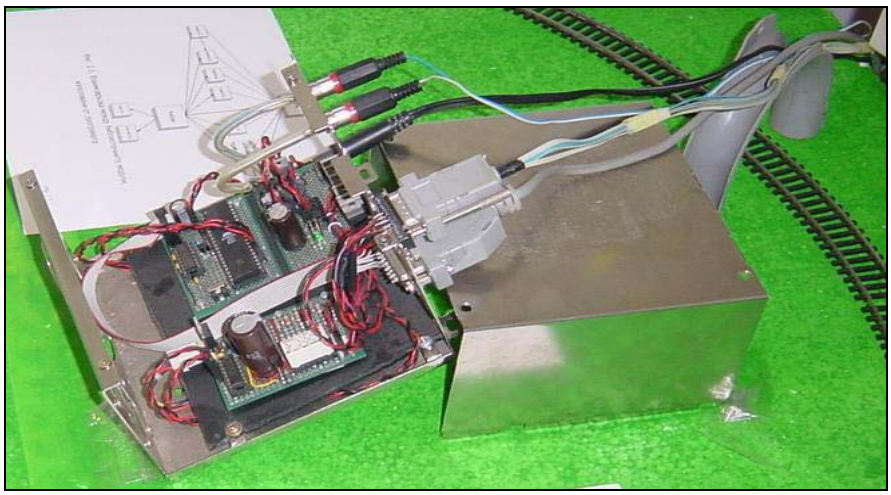
Fig. 3 Block Structure of the Firmware and the Communication Protocol.

3. REFERENCES

[1] Monov I., *Computer Based Remote Control for Layout of Scaled Model Trains - software*, TU-ES, Sofia, 2004.

[2] Gorchev I., *Computer Based Remote Control for Layout of Scaled Model Trains - hardware*, TU-ES, Sofia, 2004.


TraCon - Untitled
File Edit View Tools Help
Control
STOP
71 %
71 %
x=0329




БЪЛГАРСКА СТОПАНСКА КАМАРА
ПОЧЕТНА ГРАМОТА
ИВАН МОНОВ И ИВАН ГОРЧЕВ


награждава с
ДИПЛОМ
ЕКСПО ИНТЕЛЕКТ '04
GOLD 2004