FPGA Interface For Optical Mouse

Rotar Dan
"Vasile Alecsandri" Bacău University,
Calea Mărășești, 157, Bacău, 600115, Romania
drotar@ub.ro; drotarubro@yahoo.com

Abstract

Measuring the displacements with optical sensors similar to those used in mouse type devices has been used in various fields for several years. Among these areas we can name: measuring the movement of robots, measuring the movement of their assemblies, positioning paper from printers, vibration measurement, etc…

The main advantages of such a sensor are represented by the fact that this is a bidirectional sensor without contact. The interface usually used for such a sensor is the PS / 2 (Personal System / 2) interface. This bidirectional interface ensures the serial transmitting of the commands to the mouse and the receiving of the position or of the status words from the mouse.

Typically, such systems using optical positioning devices also employ microcontrollers with which the PS2 interface is obtained. The software resources used for programming such interfaces should be considered and the situation gets more complicated if the use of multiple mouse devices is necessary.

The paper shows how to achieve an interface with multiple mouse devices using a Xilinx Spartan E3 type programmable logic array (FPGA - Field-Programmable Gate Array). Up to four mouse type devices can be connected at the achieved interface and the interface can be connected to the microcontroller via a USART (Universal Synchronous/Asynchronous Receiver/Transmitter) serial interface, a I2C (Inter Integrated Circuit bus) interface, a SPI (Serial Peripheral Interface) interface or through a high-speed parallel interface.

The interface is equipped with additional facilities to enable work with mouse type devices. The interface allows the setup of each device separately, the storing of the information received from the mouse type devices, the prescription limits and the signaling errors. The configuration is done by the microcontroller interface to which it is attached.

The FPGA device programming was done in VHDL (**V**HSIC - Very High Speed Integrated Circuit - **H**ardware **D**escription **L**anguage). The paper presents the blocks made, how they interconnect and the operation of the interface.

The main benefits of the solution presented in the paper are: relieving the microcontroller from the communication activities with mouse type devices, allowing the use of a microcontroller with reduced performances, increasing operational safety and adding new features to the system.

The paper also presents the experiments and their results.

Keywords

PS/2 interface, PS/2 mouse protocol, Field-Programmable Gate Array, microcontroller, optical position sensor.

Introduction

Because the optical positioning devices similar to the mouse type devices used in the computer field have become widespread in mechatronics, the necessity to use a specialized interface between the mouse type optical sensors and the computers appeared. When using the mouse type devices, it is very important to respect the conditions regarding time imposed by the protocol for PS / 2 [1]. If the mechatronic system uses more optical sensors, it is not convenient to connect them directly to the computer system because using these sensors in real time requires a large amount of calculation and an increased CPU speed. This interface is designed to simplify the programmer's work, to relieve the computer of certain activities on these sensors and to increase working speed and reliability.

This paper presents the work done on the design and on the use of a specialized interface [2] between the mouse type sensor system and the computer. The interface was achieved by programming in VHDL; and the application was developed in the Xilinx ISE WebPACK programming environment, for the Xilinx Spartan E3 programmable logic array.

Firstly, the PS / 2 mouse protocol [3] will be briefly explained in order to understand the structure of the

achieved interface.

The PS / 2 standards, introduced by IBM Company in 1987, allow connecting the keyboard and the mouse devices to the personal computer type computer systems. The connector used is a mini-DIN connector with 6 pins. The signals present at this connector are: Date, Clock, Vcc and GND (ground). The devices attached to the PS / 2 connector are powered by the Vcc and GND pins. Current devices can operate at a voltage of 3.3V or 5V. For a correct operation of the mouse device, its voltage must be respected. If the mouse device is powered from an incorrect voltage, it is likely the device will provide wrong information.

The mouse (like the keyboard) uses a serial bus with two lines (Data and Clock) to communicate with the host device. The communication is both ways. The host device sends control information to the mouse and the mouse device responds at commands and sends status information and position information to the host device. In order to ensure bidirectional communication, the two bus lines of the PS / 2 are open-collector connection lines being connected to Vcc voltage through two resistors. This way, when the two lines are in rest state, they are in a high logic state (they have a high voltage).

The clock signal is always generated by the mouse device on the Clock line, the host using this line for synchronization or signaling (when it wants to interrupt the reception). The frequency of the clock signal on the PS / 2 line is 10 - 16.7 kHz.

On the Data line, the mouse device submits the data, which are received by the host, or the host submits the data to the mouse device. When received, the data are read by the host on the falling edge of the mouse clock signal and when emitted, the data are read by the device on the rising edge of the mouse clock signal.

The information is sent in the form of 11 or 12 bits frames. These bits are:

- 1 start bit (always logic zero);
- 8 data bits (the least significant bit is transmitted first);
- 1 parity bit (the 1 bit values of data plus this bit gives an odd number – odd parity);
- 1 stop bit (always has the logical value of one);
- 1 acknowledge bit (used only to emission).

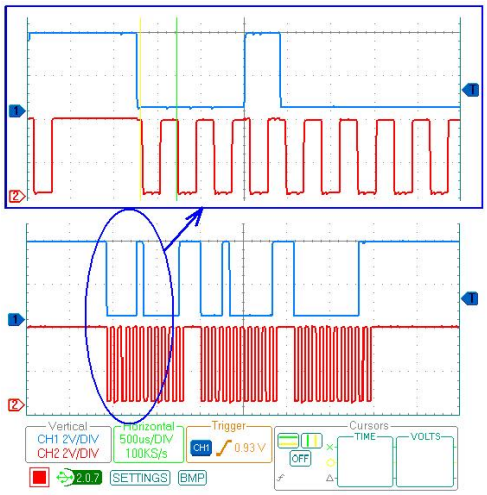

Figure 1: The form of the signal received by the host

As can be seen, the protocol for reception (the sending of the data from the mouse device to the host) differs from the protocol for transmission (the sending of the data from the host to the mouse device).

For the reception, the mouse device should check if it could begin sending data to the host. To do this, it checks the status of the Clock line. The Clock line must remain at rest (at high logic state) at least 50 microseconds for the mouse device to start data transmission.

The host can interrupt the reception if it puts the Clock line to the zero logic state at least 100 microseconds.

The data received from host is read on the falling front of the clock signal. For the design of the receiver, we consider that the Data line changes when the Clock signal is high and it is stable when the Clock signal is low (Figure 1).

As shown in Figure 1, for reception the host receives a three frames data packet from the mouse device type, each frame consisting of 11 bits each, so a 33-bit package. The structure of this package is presented in

Table 1. It should be noted here that there are extensions of this protocol. The best known is the extension of PS / 2 Microsoft Intellimouse for which 4 bytes are sent. The fourth byte contains information from the additional buttons and from the scroll wheel.

The mouse has two 9-bit counters for the movement coordinates (8 bits transmitted in the least significant byte 1 or 2), the sign bit, the most significant bit (values are represented in two's complement) and overflow bits (if the counter capacity is exceeded at a movement of the mouse device). The contents of these registers are the relative position of the device since the last mouse move. In other words, the counters are initialized to zero after the data is successfully sent to the host. Therefore, if we are to determine the position to an initial point, it is necessary for the mouse device interface to collect the values received at each move.

Table 1. The received package structure.

	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
Byte 0	X overflow	Y overflow	Y sign bit	X sign bit	Always 1	Middle Btn.	Right Btn.	Left Btn.
Byte 1	X movement							
Byte 2	Y movement							
Byte 3	Z movement							

To determine the movement of the mouse device its two parameters must be taken into account: resolution and scaling. The default resolution is 4 counts / millimeter and it can be changed with the command "Set Resolution" (0xE8) sent from the host to the mouse device [3]. Scaling does not affect the content of the mouse device counters but it changes the value transmitted to the host. The default scaling is 1:1, but it may be changed by the "Set Scaling 2:1" (0xE7) command to the value of 2:1 when for a n value from the counter, a 2 * n value is sent to the host.

For the emission, the host sends commands to the mouse device. Therefore, the host puts the Data and the Clock lines in "request for transmission" to announce the mouse device that it will initiate a data transmission. The Clock line is set for this in the low state at least 100 microseconds and after that the Data line state is in the low state, releasing the Clock line. The mouse responds generating the clock signal. The low state of the Data line represents the start bit, and the host ads to this the 8 bits of data, a parity bit and a stop bit. The mouse responds with an acknowledge signal on the data bus which they put in the low state during the last clock pulse. If a command requires a response from the mouse device, it must respond to 20 ms.

The mouse can be placed in four modes:

- Reset – represents the behavior of the mouse device when the power is on or to the initialization command. In this way the self-test of the device and its initialization are made. The initial values are: Sample Rate = 100 samples/sec, Resolution = 4 counts/mm, Scaling = 1/1, Data Reporting = disabled.
- Stream – it is current work mode in which the mouse sends data packets when its position changes or when one of the buttons is pressed.
- Remote – in this mode the data are not sent automatically at the execution of move from the mouse device but only at the request of the host.
- Wrap – is a diagnostic mode in which the mouse resends all data packets received from the host.

Communication interface with mouse devices

The communication interface was modularly designed in VHDL for a Xilinx Spartan-E3 circuit. The block diagram of the realized interface is presented in Figure 2. The main components of the mouse devices interface are four interfaces PS / 2 which connect the four mouse devices type that operate independently, the controller device that handles data between the mouse devices and the computer and the interfaces that connect the main computer system. Four types of interfaces are designed to connect the main computer system: USART serial interface, I2C interface, SPI interface and parallel interface. Only one of these interfaces is connected to the main computer system at a time. The used interface is activated via a switch on the mouse interface devices. The enabled interface has the selection signal positioned in the high logic state and the other interfaces have the selection signal in the low logic state. The unselected interfaces put the Data_out bus in the high impedance state. The selection signals for each interface are: sel_usart, sel_i2c, sel_spi and sel_parallel.

The signals shown in Figure 2 are described below.

The PS / 2 bus interfaces signals denoted by PS2_INTERFACE_n, n = 0, 1, 2, 3, are:

- ps2c – bidirectional bus clock signal for PS/2;

- ps2d – bidirectional bus data signal for PS/2;
- reset – initialization signal common to all modules (input);
- clk – control clock signal common to all modules (input);
- busy – response signal indicating that the interface shall carry out an internal activity and cannot read or write the interface (output);
- err – interface signals an error occurs (output);
- rd – interface states that have valid data to send (output);
- wr – writing in the interface command (input);
- data_in – 8-bit input data bus (input);data_out – 8-bit output data bus (output).

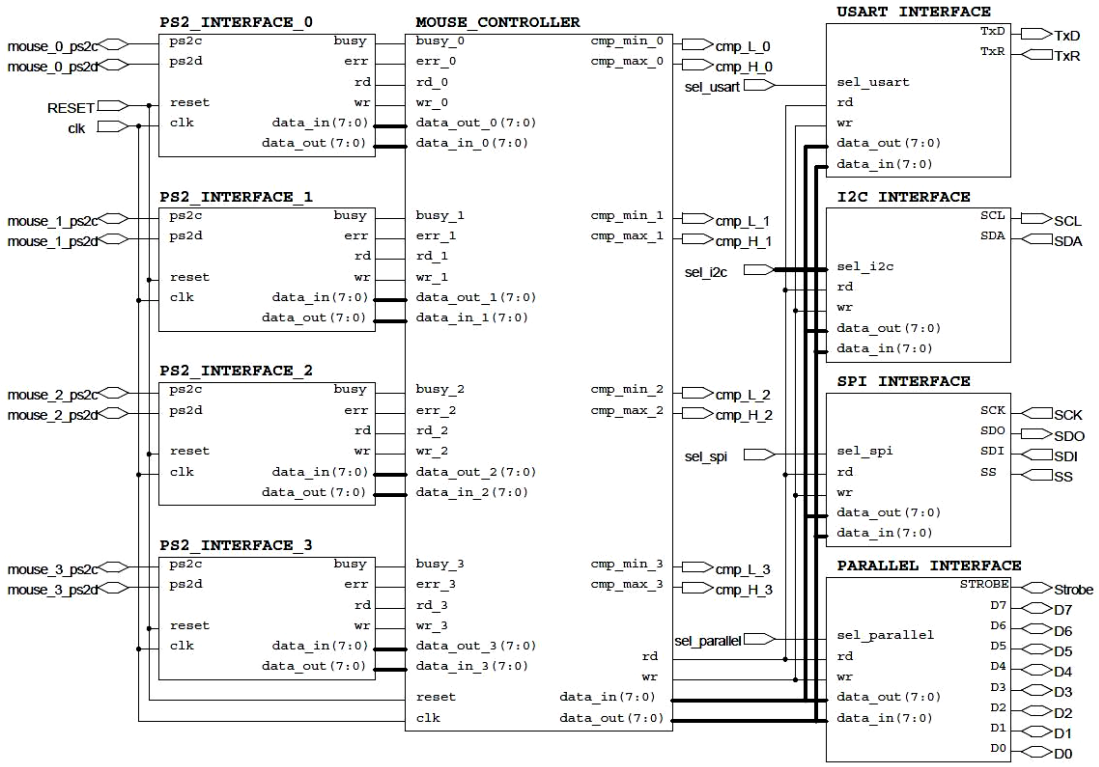

Figure 2: Block diagram of the designed interface.

The significance of the interfaces signals for connecting to the main computing system is similar to those shown. The communication interferences also present standardized signals that are described in many specialized works.

The interfaces that connect the mouse devices, denoted by PS2_INTERFACE_n, n = 0, 1, 2, 3 are interfaces for connecting the mouse devices to the PS / 2 bus. This mainly means that between the interface and device there will be a two-way flow of information. The structure of this interface is the usual one that is described in numerous specialized works [4] and therefore no longer presented here.

The interfaces for connecting with the computer: USART, I2C, SPI or the parallel interface structures are known also and will not be discussed here because there is a rich literature in this field.

The novelty presented in this paper is the mouse controller. This controller consists of several finite state machines (FSM) operating in parallel. A finite state machine is used for each PS / 2 interfaces and another finite state machine is used for communication with the main computer system. The finite state machines connected to the PS / 2 interface take information from them and depose the information in a common memory area, they send commands from the main computer system to the mouse device, they report the exceeding of the limits and they monitor the mouse device status. At this level, the operation of the finite state machines is identical for the four interfaces. For this reason, the general mode of operation of the 'n' machine with n = 0, 1, 2 or 3 will be presented. The states of the finite states machine are: initialization (INIT), reading data from the mouse device (DATA_M) mouse device write command (CMD), reading data from the main computer to the maximum and

minimum displacement (DATA_C), comparing the current position with the prescribed values (COMP) and determining the status of the mouse device (STATE). The ASMD simplified diagram (Algorithmic State Machine with Data path chart) is shown in Figure 3. In this figure, INIT is the initial state. In normal operation (no errors) the finite state machine reads data from the mouse device (DATA_M), sends commands to the mouse device (CMD), reads the prescribed sizes (DATA_C), compares the current prescribed sizes (COMP) and tests the mouse device (STATE). If an error occurs, reset (INIT) of the mouse device is attempted and if this fails, the error is then reported.

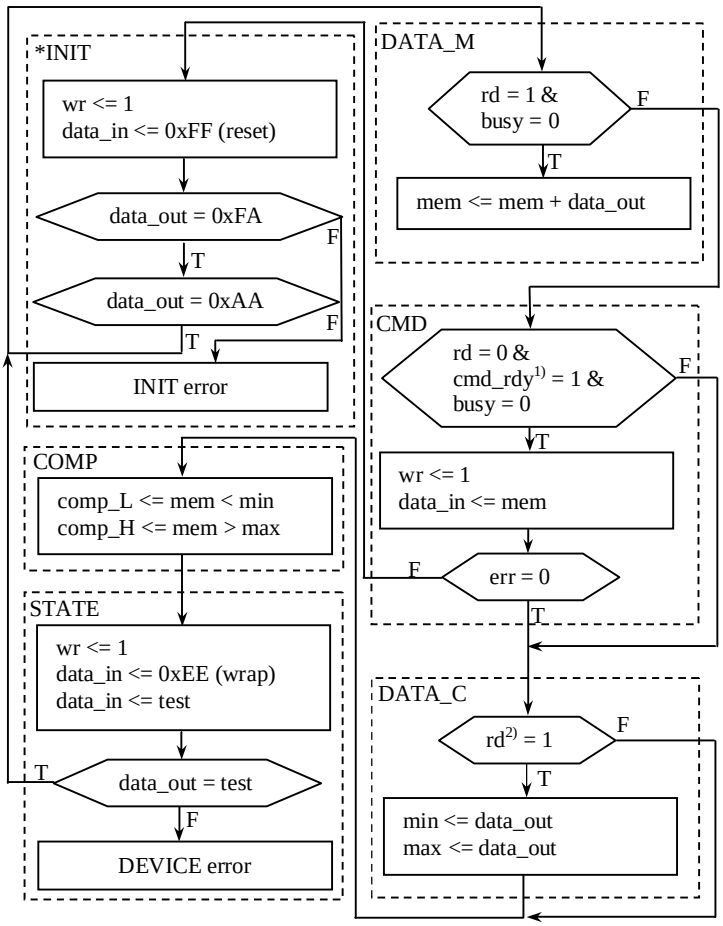

Figure 3: Simplified ASMD chart.
1 – internal signal; 2 – rd signal from PC interface.

The mouse device type initialization is done by sending the "Reset" command to the mouse device and its answer is expected that if the mouse device is initialized properly, it must be the sequence: "Acknowledge" and "Self-test passed". After the mouse device initialization the following commands can be sent: "Set Sample Rate", "Set Resolution" and "Set Scaling" in the CMD state. If these commands are not sent, the mouse device is set to the default values. Once the value of the parameters are established, the command "Enable" is sent, in which case the interface considers the mouse device functional when acknowledgement is received to this command from the mouse device. The interface can send the command "Read Device Type" to determine the type of mouse device attached.

In the DATA_M state, the coordinates read from the mouse device are added to the value previously read and stored in the coordinates memory described below. The PS / 2 interfaces execute operations of writing or reading. During the writing / reading, the "busy" signal is enabled (has a high logic value). The reading operation is a priority in relation to the writing. When data are read, the interface enables the "rd" signal.

In the CMD state, the interface sends a command retrieved from the commands and limits memory to the mouse device. Detecting the existence of a new command is done using the "cmd_rdy" internal signal and the

mouse device is notified of the presence of the command with the "wr" signal.

Because the mouse device type sends its position only when it is moved, there is the possibility that damage to the device is not detected. For this reason the mouse controller periodically enters the STATE state for the test purpose. In the STATE state the finite state machine put the mouse device in the "wrap" state. In this state the mouse device returns the received character. This will verify the existence and the functioning of the device.

In the DATA_C state, data about the limits for the coordinates set by the operator are taken. These values have only a limited importance because they are necessary for the activation / deactivation of the signals "cmp_L_n" and "cmp_H_n" with n = 0, 1, 2, 3, in the COMP state. The maximum and the minimum limit values are taken from the memory for commands and the limits are described below. The signals "cmp_L_n" and "cmp_H_n" n = 0, 1, 2, 3, modified as a result of the comparison can be used as protection signals to stop drives when the prescribed limits are reached.

The mouse Controller comes with two memory areas for storing the data. A data memory area containing the measured coordinates from the four mouse devices. The structure of this memory area is shown in Figure 4.

header	x coord$_0$	y coord$_0$	x coord$_1$	y coord$_1$	x coord$_2$	y coord$_2$	x corod$_3$	y coord$_3$

Figure 4: The coordinate memory structure.

In Figure 4, each element is represented by a word (two bytes), which means that the memory has 18 bytes. The most significant word is a header containing the information about the correctness of the coordinate information words. Each coordinate is written by two bytes containing two's complement representation of it. Whole series of memory locations, which represent the coordinates, are continuously sent to the selected interface to connect to the main computer system. Depending on the established speed of communication we can determine the refresh rate of the coordinate information on the main computer system. Because information is transmitted continuously, it is possible that some coordinates are erroneous for various reasons, as discussed above. This thing is signaled to the main computer system through the header word.

The second area of memory contains the mouse device commands or the data for the mouse device controller data (minimum and maximum coordinates prescribed). The contents of this memory are shown in Figure 5.

header	cmd$_0$	x,y min$_0$	x,y max$_0$	cmd$_1$	x,y min$_1$	x,y max$_1$	cmd$_2$	x,y min$_2$	x,y max$_2$	cmd$_3$	x,y min$_3$	x,y max$_3$

Figure 5: The limit and command memory structure.

In Figure 5, the size of the elements is different: the header is two bytes; cmdn is a byte; x, y, Minn., and x, y maxn where n = 0, 1, 2, 3 have two bytes per coordinate. The data memory area for controls and the coordinates' limits are 38 bytes. This information is sent from the main computer system to the interface with mouse devices only when necessary (when new information). The word header provides the information on the fields containing updated information.

The two areas of memory: the memory of the coordinates data to be transmitted from the interface to the main computer and the memory of the limit coordinates and the mouse device commands that are received by the interface from the main computer system are also managed by a finite state machine. This machine continuously sends data to the selected interface; receives data from the main data system and produces control signals for other elements of the interface.

The main computer runs an application written using Visual Studio 2008 development environment made for testing the interface. The application can be a model for the use of the interface because it exploits the main features. The application contains multiple windows that open depending on the context but its main window may make explaining the operation of the application. The main application window is shown in Figure 6.

The Figure 6 window is displayed when starting the application "Mouse Interface Test". As shown in this figure, only two mouse devices: device 0 and 1 are connected at the interface. The other two devices 2 and 3 are not connected to see if the interface sends the correct information. This consisted of information displayed on the "Device status" (devices 0 and 1 is displayed "OK" and the devices 2 and 3 is displayed "Error").

The main menu that appears at the top of the window contains the options: File, Select Interface Select Password and Help. The File option allows data from the mouse to be saved in a text file. The "Select Interface" allows the choice of one of these possible communication interfaces: USART, I2C, SPI or parallel interface. Depending on

the interface chosen, if necessary, the configuration data for that interface are made. The Set Password option allows the user to use a password to work with the program. This is necessary to avoid accidentally sending orders to the mouse device or to prevent unauthorized modification of the minimum and maximum limits set for the movement on the two axes. The Help option provides information on how to work with the application "Mouse Interface Test", information on configuring interfaces and information about the mouse commands.

In the main window of the program more information on the four mouse devices that can be connected to the interface is presented. This information can be interactive in the sense that the user can change it at will.

The first line, called the "Position" is a read-only information line. The user cannot change this information and they are the coordinates provided by the four mouse devices. The interface for the mouse devices sends to the main computer the mouse position information expressed in counts / mm as they were recorded. Depending on the settings established by the operator in terms of resolution and scale factor used and sent to the mouse device through the commands, the application "Mouse Interface Test" determines the real coordinates expressed in millimeters.

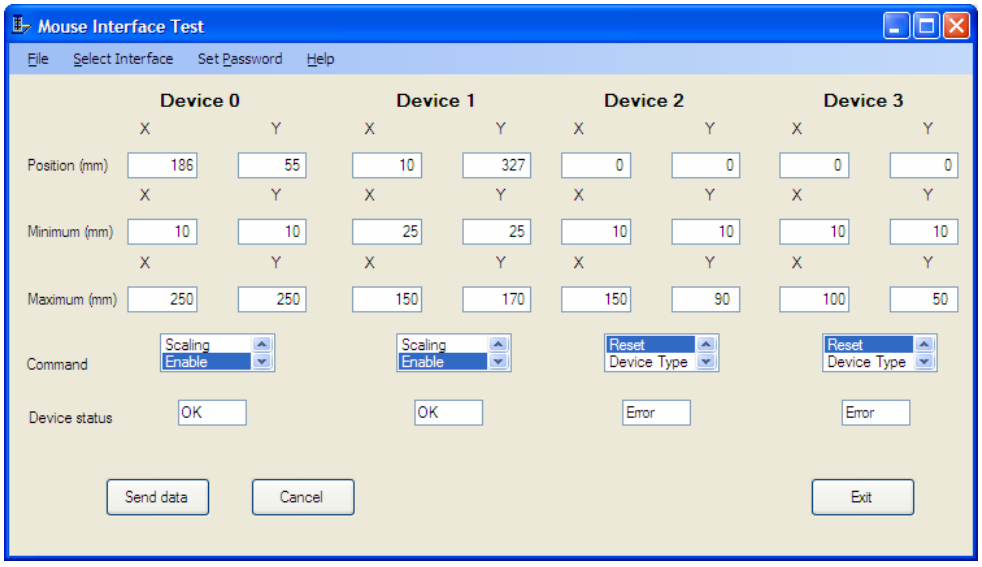

Figure 6: The main window of the mouse interface test application.

The second line represents the limit of movement on one axis. This line is interactive in that the user can change the displayed value. When a value changes, the buttons "Send data" and "Cancel", initially inactive, are activated. Pressing the button "Send data" causes the sending of the data to the selected interface by the mouse devices controller. Pressing the "Cancel" button invalidates the change and the previous data is displayed again (the changes are deleted). The data corresponding to the minimum limit must also to be sent in the counts per millimeter and the application must convert the values before they are sent to the interface.

The next line, similar to the previous one, refers to the maximum limit set for movement on the two axes counted by a mouse device.

The "Command" line is also interactive, allowing the user to send commands to the mouse device. The commands that can be sent are: Reset, Device Type, Sample Rate, Set Resolution, Scaling and Enable. Changes and the cancellation of these changes are the same as described in "Minimum" line.

The "Device Status" line provides information on the mouse device state, information sent via the selected interface, from the mouse devices controller to the main computer system.

Conclusions

The mouse is an inexpensive two directions movement sensor [5] that makes this device adequate to be used in certain applications of mechatronics. If an application wants to use more such sensors then they must be connected to a computer via an interface. This paper presents such an interface that can connect up to four mouse devices. The interface can work independently, in which case it can send control signals when limits are reached or it can work when connected to a computer.

Using the interface presented in this paper leads to the simplifying of the design work, to the increase of operational safety and to the decrease of the time for making an application.

The interface has achieved low power consumption. However the interface design was aimed at increasing reliability. If energy consumption is an important criterion it is necessary that the interface be designed so as to achieve certain optimizations.

For the connection between the interface and the computer, the USART interface is recommended. This interface provides full duplex communication at high speed that ensures a good control system.

The experiments carried out using an experimental system that interfaces with four mouse devices indicates that the operation is achieved with very good results in terms of safety, the possibility of rapid detection of a defect, simple maintenance and it can also control the system in a complex way.

Bibliography

[1] Rotar Dan, "Optical Tracking Method for Mechatronic Systems", Romanian Review Precision Mechanics, Optics & Mechatronics, ISSN 1584 - 5982, pp. 47-52, no. 38/2010

[2] J. O. Hamblen, T. S. Hall, and M. D. Furman, "Rapid Prototyping of Digital Systems", SOPC Edition, New York, Springer, 2008.

[3] Adam Chapweske, "PS/2 Mouse/Keyboard Protocol", http://www.computer-engineering.org, 2011

[4] Pong P. Chu, "FPGA prototyping by VHDL examples. Xilinx SpartanTM-3 Version", John Wiley & Sons, USA, 2008

[5] T. W. Ng, "The optical mouse as a two-dimensional displacement sensor, Sensors and Actuators A: Physical", Volume 107, Issue 1, Pages 21-25, 1 October 2003

[6] T. W. Ng, K. T. Ang, "The optical mouse for vibratory motion sensing, Sensors and Actuators A: Physical", Volume 116, Issue 2, Pages 205-208, 15 October 2004

[7] W.P.H. Kamphuis, "Using optical mouse sensors for sheet position measurement", Technische Universiteit Eindhoven, February, 2007

[8] Marcel Tresanchez, Tomàs Pallejà, Mercè Teixidó and Jordi Palací, "Using the Optical Mouse Sensor as a Two-Euro Counterfeit Coin Detector", Sensors, no. 9, pp. 7083-7096, 2009

[9] WANG Xin, Tactile "Sensing with Optical Mouse Sensor", Department of Advanced Systems Control Engineering Graduate School of Science and Engineering, Ph. D. Thesis, Saga University, Japan, September 2008