Preliminaries on Latching Low Level Physical Signals from a parallel port of a PC

Mustafa Dülger

Möhringer Strasse 108/C 70199 Stuttgart - Germany
http://www.plcturk.com mailto:admin@plcturk.com

Abstract — in this study, a practical approach is viewed to capture low level physical signals, which are in form of TTL (Transistor-Transistor-Logic) signals, from a standard parallel port of a personal computer, PC. Both hardware and software consideration is taken into account. An appropriate hardware design to latch TTL signal out of the PC, is proposed. High level programming language commands as well as operating system commands are mentioned in context with programming issues.

Index Terms — TTL, Low Level Signal, Parallel Transmission, Serial Transmission, RS232, TWI, IO.

I. INTRODUCTION

Aim of this work is to provide a compact design in order to get low level signals from a PC as the use of PC widely spreads out in industry. Under the term *low level signal* [1], one should understand TTL signal. TTL is the abbreviation of Transistor-Transistor-Logic and corresponds to the low level voltage signal in electronic. The range of TTL signal varies 0-5 volt where 0-2 volt shows logic false and 3,5-5 volt shows logic true. Hereby 2-3.5 volt interval is undefined and must be avoided to occur.

Today's PCs are capable of processing huge amount of information in the binary form. One can communicate with PC not only through standard input/output (IO) devices, such as mouse and monitor, but also through standard ports such as parallel or serial port under certain protocols. For instance, printer is connected to PC via parallel port and exchanges data in the binary form.

Additional ports, either serial or parallel, can be installed on PC by inserting related port card in its slot as the need increases.

With the convenient software, or even with operating system commands, PC can send and/or receive binary information to/from its ports. This information manifests itself in the form of the voltage level. With the help of appropriate hardware, one can catch and set these voltages from within the PC.

In industrial use, one needs extra digital and/or analogue input - output channel to perform signal transmission. In the majority of applications, PC signals are to be amplified to the level as required by an external process.

In following sections, preliminaries will be briefly highlighted in order to ease the hard- and software design. Not only the creation of low level signals, but also the amplification of these signals is important. Programming aspect of these signals will also be discussed.

II. DATA TRANSMISSION & PROTOCOLS

One must differentiate between two types of data transmissions to/from PC. They are namely serial and parallel data transmissions. The main differentiating characteristic is the number of physical lines involved to transmit data. Only one physical line is engaged in serial data transmission. Where as more physical lines are simultaneously implemented in parallel data transmission. Serial data transmission requires certain transmission protocols such as RS232 [2], TWI [3] and SPI [4]. Parallel data transmission does not necessarily dependent on a certain protocol.

For a serial port, mostly the hardware protocol RS232 is implemented and the signal level is different than the TTL signal level. It is intentionally so because RS232 protocol is designed to convey serial data over relatively large distances, e.g. more than 2 meter and until maximum 12 meter. It must least sensible to all kind of disturbing interferences. Therefore signal level is designed to lie in the range of -12 volt and +12 volt at serial port. One installs therefore extra RS232 driver IC for a serial port in a PC.

Parallel port is designed to exchange data between PC and external devices where data rate is to be relatively dense and transmission distance is very short. The signal level at parallel port is a

TTL signal level. Because of TTL signal level, parallel port can not transmit data over long distances. TTL signal is extremely influenced by external interferences. Maximum allowable distance for the TTL signal should not exceed one and half meter. The nominated parallel port LPT1, for instance, is initially designed to communicate with a printer connected via the parallel port. This does not however necessarily imply that parallel port can not be used for other purposes.

III. DESIGN OF LATCH MODULE

Because we mainly interested in getting TTL signals from a PC, we do work on the parallel port of a PC. Our target is to develop a latch module which gets TTL level signals from the parallel port and amplify them for an external use.

Fig.1 illustrates schematically how a signal from a PC is cached to an external process. Latch-IC allows data to be passed to the process (external-system) bus. Both data bus and control bus are connected directly parallel port(s) of a PC. Data flow is in one direction, namely from PC to the process. In case of bi-directional data transmission, Latch-IC [5] is to be replaced with bus driver-IC [6] in order to permit data flow in both directions.

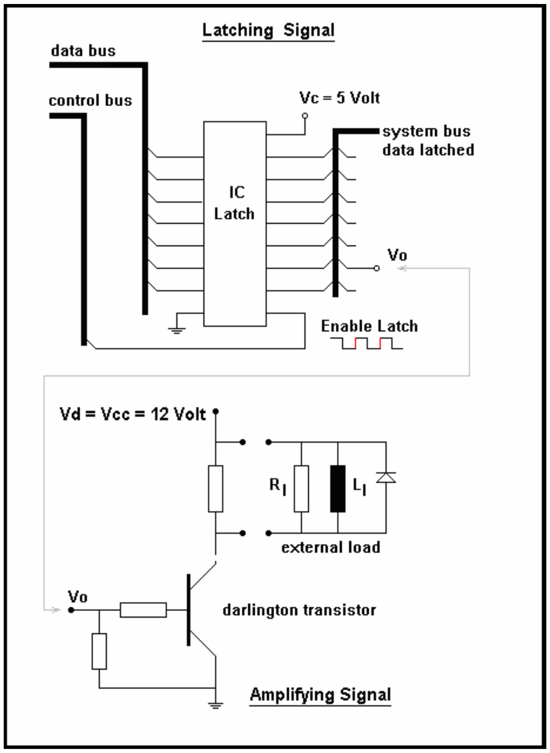

Fig.1 Latching and Amplifying Signal from a PC

When the Latch-IC is activated by the enable latch signal on the control bus, data which are in form of a voltage on pins of the data bus port at the time of activation, are delegated (latched) to the system bus. One should notice that the system bus is an external bus. The latched signal (Vo) is at low level (0 - 5 volt). In most of the processes, latched signals are amplified in order to drive an external load Rl. This is achieved through an external amplifying circuit. The voltage Vd is the driving voltage of the process driven by a corresponding controlling signal at the port of the PC.

Amplification is done by a Darlington transistor. The characteristic of a Darlington transistor is that, it is formed from a couple of transistors. It has an internal protection diode, which protects system from over current that may occur while switching. Because of high current amplification, Darlington transistor is generally used to drive external device by a Low Level Control Signal coming from a PC or a TTL control circuit.

IV. LATCH IC

Latch-IC is an integrated circuit which has an external one byte memory and keeps the state of input values at the time of enabling the IC. Enabling is performed with a positive clock pulse. State is only changed on positive side of the pulse. That means, it updates its internal states with input data only if the clock pulse propagates from low states to high states (positive triggered). Otherwise it holds the internal state unchanged and delegates its internal state to the output.

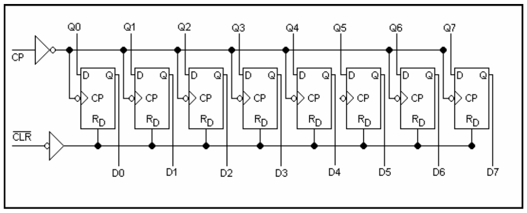

Fig.2 Logic Diagram of a Latch-IC

Fig.2 is a logical representation of a Latch-IC consisting of eight positive-edge-triggered flip-flops with individual D-type inputs. The buffered clock and clear are common to all flip-flops. The eight individual flip-flops will store the state of their individual D-type inputs. When the clock is either high (H) or low (L), the D-input signal has no effect at the output. When the clear is low, all outputs will be forced low. The truth table is given in Table.1.

Table.1 Truth Table of a Latch-IC

INPUTS			OUTPUTS
D	CP	CLR	Q
H	L to H	H	H
L	L to H	H	L
X	H or L	H	Q
X	X	L	L

V. SOFTWARE CONSIDERATION

Communicating with ports is performed either through operating system or through direct programming by high level programming languages. Under windows operating system, the system command,

Copy 19 lpt1 / b

yields the binary equivalent of 19 to be copied to the printer port LPT1. This command is however only valid for those ports defined in the operating system. Access on the additional ports through the system command is not possible.

Programming with high level languages gives more flexibility. C++ provides powerful libraries to program hardware of a PC. By including these libraries in an end application (Application Program), all functionality will be passed to the end application. Communicating with a port sitting in an address space of a PC is hence nothing else than including couple of statements in the quell code of an end application. The function,

int _outp(unsigned short *portnumber*, int *databyte*)

for instance, causes value contained in *databyte* argument to be written to the port identified by *portnumber*. Here type indicator int shows that function _outp returns an integer value after successfully calling the operation. The returned value is equal to output data if the call does not fail. Otherwise the returned value is negative.

Both function arguments *portnumber* and *databyte* are in-parameters and passed to the function. Portnumber identifies the port with which the function should operate. It is a place holder for the physical address of the port. *Databyte* is the data which will be sent to the port identified by *Portnumber*. The port argument *portnumber* can be any unsigned integer in the specified range 0 - 65,535. The data argument *databyte* is limited to any byte in the range 0 – 255.

VI. DISCUSSION & CONCLUSION

A practical and reliable design procedure is presented to get low level signals from a PC. The classification of these Signals is also done. The characteristics of these signals are mentioned in comparison with each other. Hardware protocols are briefly highlighted.

A new and compact hardware module is developed in order to latch signals from the parallel port of a PC. Programming aspect of these signals is also highlighted. How the hardware module communicates with PC is then presented. Amplification of these signals is shortly explained as they are required by an external process.

We can conclude from the forgoing discussion that a PC is not a pure software agent but a hardware programmable device as well. Hardware programming, including setting/getting signals from/to a nominated port, is not a complicated issue, in contrast to the common believing. With the hardware programming of PC, it becomes possible that the PC is involved in uncountable number of industrial applications.

VII. SYMBOLS & ABBREVIATIONS

L_l	:	Load Inductance.
R_l	:	External Load.
Vo	:	TTL Signal Voltage.
Vc	:	TTL Signal Voltage.
Vd	:	Load Voltage.
Vcc	:	Load Voltage.
PC	:	Personal Computer.
RS232	:	Serial Data Transmission Protocol.
IC	:	Integrated Circuits.
TWI	:	Two Wire Interface.
IO	:	Input / Output.
D	:	Input Data.
Q	:	Output Data.
H	:	High Level.
L	:	Low Level.
X	:	Don’t Care.
CP	:	Clock Pulse.
CLR	:	Clear.

VIII. REFERENCES

1. Dülger, M. (2003). *Developing a Software Package for Electronic Jacquard System of Classical Textile Machines.* Ph.D. Thesis, University of Gaziantep.

2. RS232 protocol, http://www.camiresearch.com/Data_Com_Basics/RS232_standard.html

3. TWI protocol, http://www.plcturk.com/webserver/kumanda/TWI.htm

4. SPI protocol, http://www.plcturk.com/webserver/kumanda/Spi_01.htm

5. 74HC273, Octal D-Type Flip-Flop with Reset.

6. 74HC244, Octal Buffer Line Driver.