Asian Journal of Convergence in Technology **Volume V Issue I**
ISSN NO: 2350-1146 I.F-5.11

Single Line Independent Communication Protocol

Nikhil M Jeby,
Electronics And Communication
Govt. Model Engineering College
Ernakulum, India
nikhil.m.jeby@gmail.com

Abstract— **This paper proposes a communication protocol for inter-microcontroller communication using a single GPIO port.**

Keywords— Single Line, Hardware Independent, Half Duplex, Asynchronous, inter-microcontroller communication.

I. Introduction

Single line hardware independent half duplex asynchronous communication protocol for inter-microcontroller communication. The protocol was developed for microcontrollers with no dedicated hardware for easy communication. The protocol is purely software based, hence making it reliable and scalable. The protocol uses just 1 GPIO port for communication and all the nodes can be connected over a single line making it more portable and affordable in the sense of both cost and space.

II. Hardware Implementation

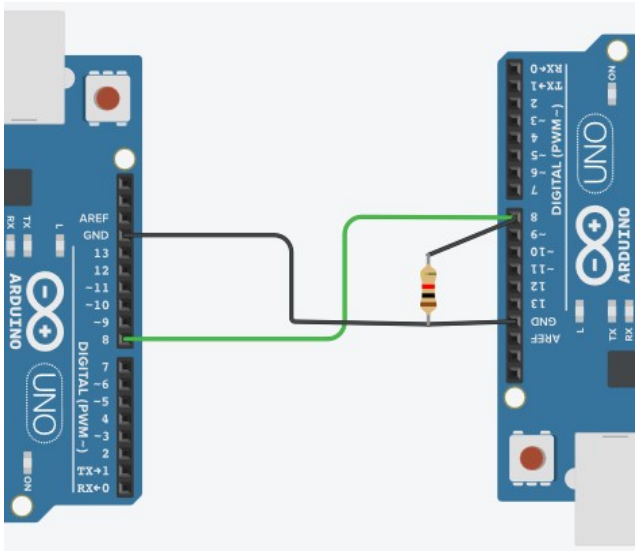

Figure 1: Hardware setup of SLIC

The hardware setup uses a single GPIO pin of 2 or more microcontrollers interconnected. The wire is then pulled down to make the idle signal absolute zero. Since the protocol is hardware independent, any kind of GPIO pin which is capable of inputting and outputting can be used.

III. Software Implementation

Generally all the nodes are in their idle or receiver mode. Whenever a transmission has to be made, that particular node act as the transmitter and sends data as individual bytes representing an ASCII data. The protocol sends a onetime synchronizing byte (10101010) to set the threshold for the bit detection.

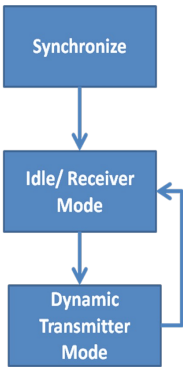

Figure2: Working Block Diagram

The protocol uses pulse position modulation for communication. This helps in maintaining constant average power dissipation as well as increases the SNR value of the channel. The current scheme uses an inbuilt ASCII translator program to convert the bit stream to equivalent ASCII characters and vice versa.

IV. SIGNAL IMPLEMENTATION

The data signals uses pulse position modulation for transmitting. Pulse position modulation has the advantages of constant average power output and it is the least power consuming modulation scheme. From various experimentation with different delays for high pulse and low pulses, for a 16-20Mhz microcontroller, the optimum pulse width were found to be in the range of 5µs -15µs for high pulses and 20µs-60µs for low state. Hence, 10µs high pulses were used as reference and 50µs low state representing '1' and 30µs low state representing '0'. 8 bits were transmitted together in-order to use an ASCII coding scheme for easier decoding.

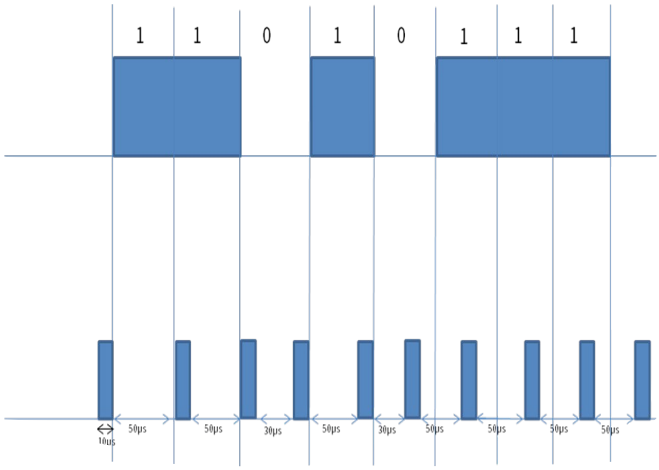

Figure 3: Sample data and equivalent pulse position modulated signal.

Each high pulses uses the voltage of the inbuilt GPIO only, hence avoiding any amplifiers and voltage level changers for communication in short range spaces.

V. Experiments

Currently the protocol uses software delays made using regular looping instructions. The protocol was successfully tested between 2 Arduino Unos which uses the microcontroller atmega328p clocked at 16Mghz. It was also tested between Arduino Uno and Arduino nano, both uses the same microcontroller but works on different boot-loader. It was also tested between Arduino Uno works at 5v voltage levels and ESP8266 12E(clocked at 80Mhz) which works at 3.3v voltage levels. By using an opamp voltage adjuster circuit the voltage corrections were made which solved any problem which existed. From the experimentation a practical speed of about 8Kbps was observed with no code multiplexing over a distance of 20cm (which is not the maximum possible length).

VI. Conclusion

The protocol was developed for some specific systems like in the case of IOT and sensing control systems where data rates are low but the number of sensors are huge and wired connections are preferred over wireless. The C++ library is open sourced for using and the same is about 3.5KB in size. The protocol can also be used by microcontrollers with no communication hardware or those with limited GPIOs. The system is currently in its development phase with futures scopes in adding functionalities such as node addressing and code multiplexing for better performance.

VII. REFERENCES

[1] B. Huang, J. Lei, Y. Bo, "The reading data error analysis of 1- wire bus digital temperature sensor DS18B20", ***Proc. Int. Conf. Modeling Identification. Control***, pp. 433-436, Jun. 2012.

[2] http://www.1wire.org

.