I2C to UART Converter
SLG46855

This app note implements a I2C to UART converter. It describes the implemented logic, GreenPAKs implementation and the obtained results. This application note comes complete with design files which can be found in the References section.

Contents

Terms and Definitions

CNT	Counter
IC	Integrated circuit
I2C	Inter-Integrated Circuit
LSB	Least Significant Bit
MSB	Most Significant Bit
UART	Universal Asynchronous Receiver/Transmitter

1. References

For related documents and software, please visit:

https://www.renesas.com/eu/en/products/programmable-mixed-signal-asic-ip-products/greenpak-programmable-mixed-signal-products

Download our free GreenPAK Designer software [1] to open the .gp files [2] and view the proposed circuit design. Use the GreenPAK development tools [3] to freeze the design into your own customized IC in a matter of minutes. Dialog Semiconductor provides a complete library of application notes [4] featuring design examples as well as explanations of features and blocks within the Dialog IC.

[1] GreenPAK Designer Software, Software Download and User Guide

[2] I2C to UART.gp, GreenPAK Design File

[3] GreenPAK Development Tools, GreenPAK Development Tools

[4] GreenPAK Application Notes, GreenPAK Application Notes

[5] SLG46855V, Datasheet

2. Introduction

In this application note, we will clarify how to create a I2C to UART converter. The design is based upon the GreenPAK SLG46855V, although any other GreenPAK with available resources can be used.

The SLG46855V is small package GreenPAK with huge amount of digital logic. The design can be easily modified to use with difference bus speed.

An Inter-Integrated Circuit bus, also known as I2C, is a very common bidirectional communication bus, using two lines to send serial information between devices. Basic I2C communication is used to transfer 8 bits or bytes of data. In normal state, SCL and SDA lines are high. The communication starts with start bit when the SDA line goes low while the SCL line is high. One bit is transferred during each clock pulse of the SCL. Data is transferred Most Significant Bit (MSB) first. Data on the SDA line must remain stable during the high phase of the clock period, as changes in the SDA line when SCL is high are interpreted as control commands (START or STOP bits).

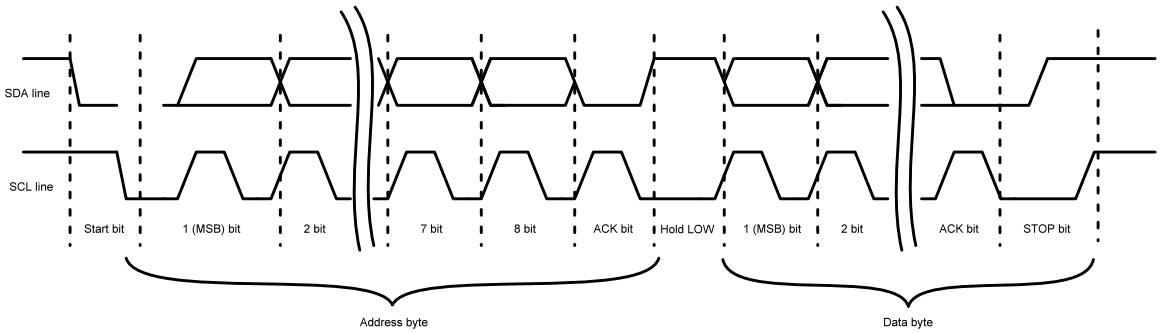

Figure 1: I2C Basic Command

One of the features of the I2C protocol is clock stretching. A slave device can hold the SCL line low after receiving a bit, indication that it is not yet ready to process more data. The master must wait until the SCL line goes high. Clock stretching is the only time where the slave controls SCL line.

A Universal Asynchronous receiver-transmitter (UART) is a simple communication bus that uses only one data line. Basic UART communication is used to transfer 8 bits or bytes with parity bit. In normal state TX line is high. The communication starts with start bit when TX line goes low for one clock cycle. Data is transferred Least Significant Bit (LSB) first. After the data bits, the parity bit is transmitted which describes the evenness or oddness of transmitted byte.

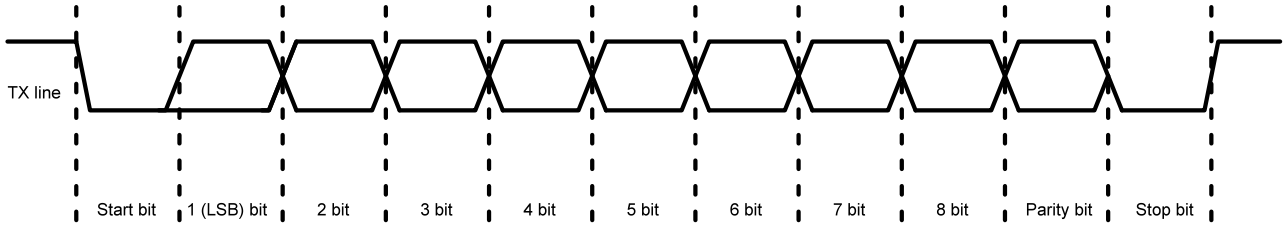

Figure 2: UART Basic Command

3. GreenPAK Design

The idea is to use SLG46855 to store 8 bits of I2C command and pull down SCL line. The I2C master will stop transmit if SCL line is pulled down. Next SLG46855 will push these 8 bits to UART and then release SCL line to receive next 8 bits. Figure 2 shows the block diagram of the connections between I2C master, UART and the SLG46855 for this application.

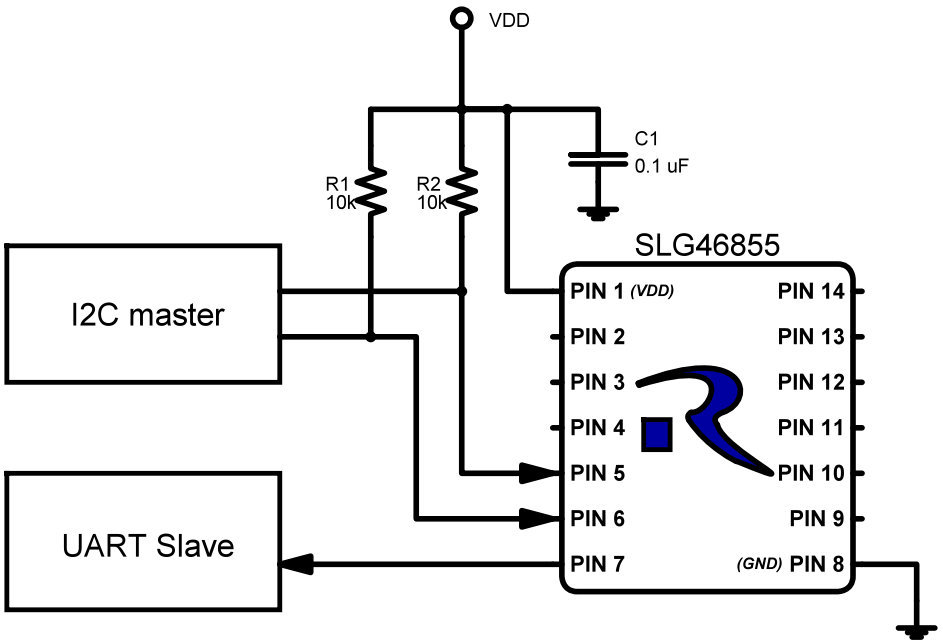

Figure 3. Block Diagram

The main problem is that I2C protocol first transmits MSB and UART's LSB. That's why we need to save 8 bits in shift register and push these bits in reverse order like stack memory.

DFF4-DFF8, DFF18 and Pipe Delay creates shift register and are used to store 8 bits. 2-bit LUT1-3, 3-bit LUT6, 3-bit LUT7, 3-bit LUT11-13, 3-bit LUT8 are used to control direction of the shift register. See Figure 4 for the design. The design loads 8 bits from SDA pin if DIR signal is '1' and pushes these 8 bits when DIR signal is '0'. In default state, DIR signal is '1' and design is ready to store data from SDA line and the SCL pin is configured as digital input. The design pushes the bits from SDA pin to shift register on every rising edge on SCL pin. CNT0 uses to count clock on SCL pin and after 8 clock cycles, the DIR signal goes '0'.

When DIR signal is '0' the design pulls down SCL line and I2C master waits until SCL line is released. CNT2 creates CLK signal to push 8 bits from shift register to UART_TX. CLK signal defines baud rate of UART. After the last bit is pushed then DLY7 creates an ACK on SDA line (pull down SDA line).

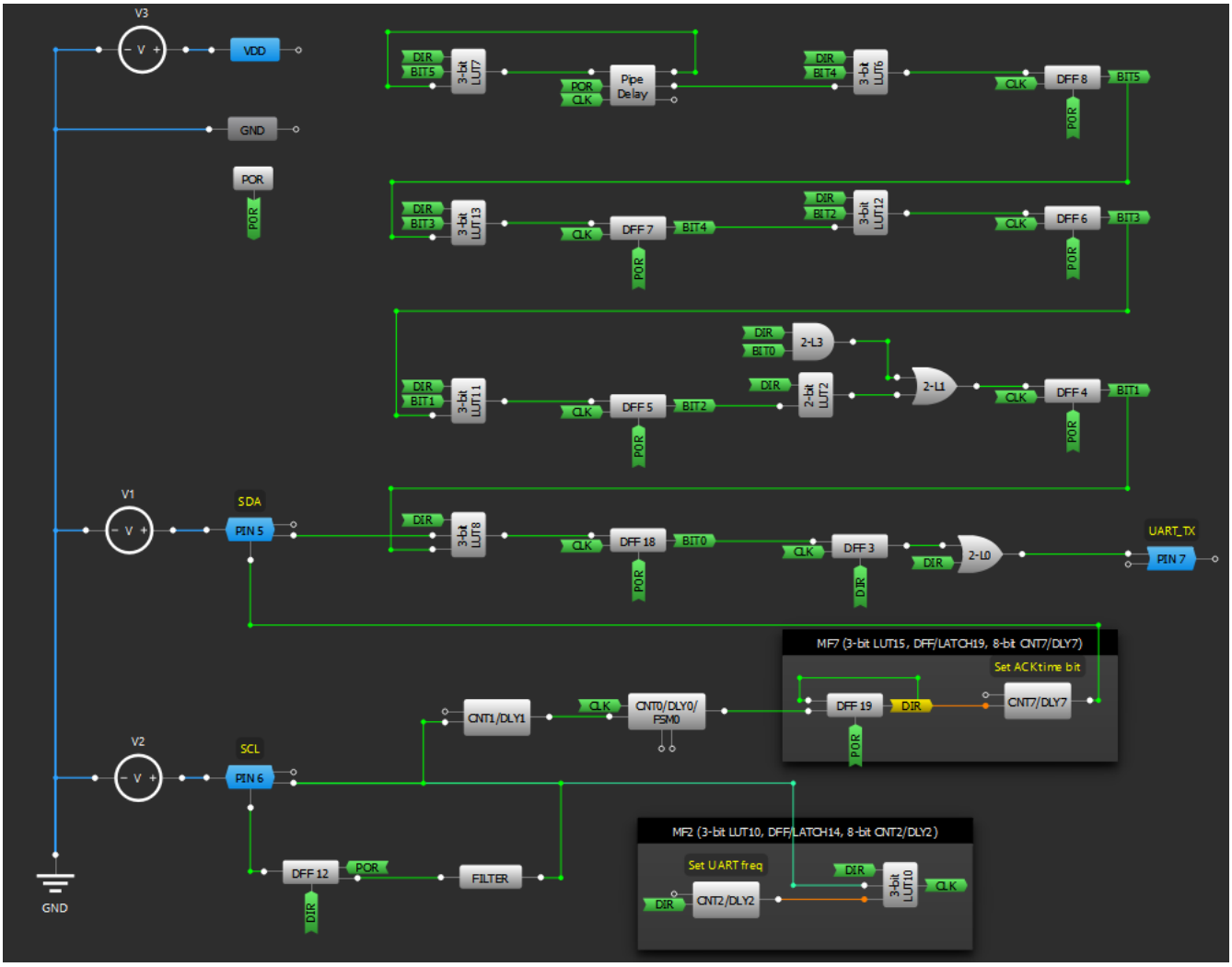

Figure 4. Smart Blind Controller Block Diagram

To test the design, the I2C master writes data **0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x007** to address **0x01**.

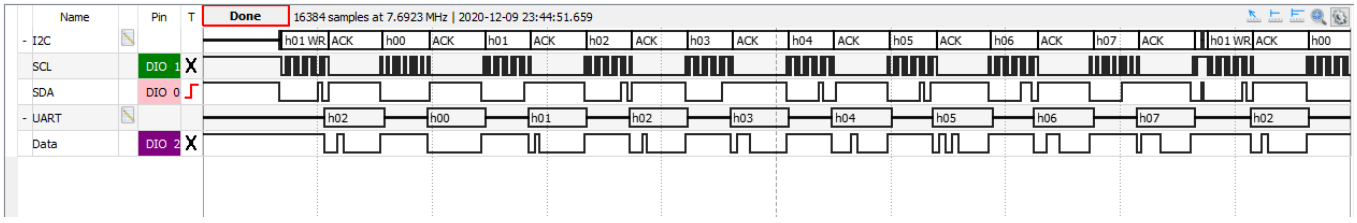

Figure 5. Experimental Transmission

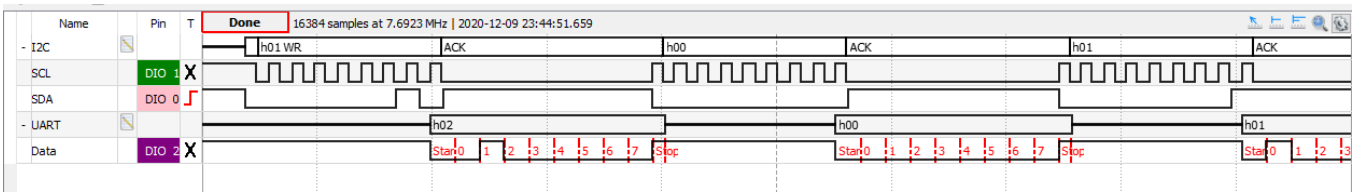

Figure 6. Zoomed First Two Bytes

Simulation results are divided into a few parts.

For Part 1, I2C master generates start bit and transmits write command for slave with address 0x01. The design receives the data and pulls SCL line to GND. On PIN 7, user can observe the UART start bit and converted slave address from I2C master. GPAK release SCL line and pulls SDA line to GND for ACK signal. I2C master transmits the data 0x00 for part 2, data 0xAA for part 3 and 0xFF for part 4. See simulation result below in Figure 7.

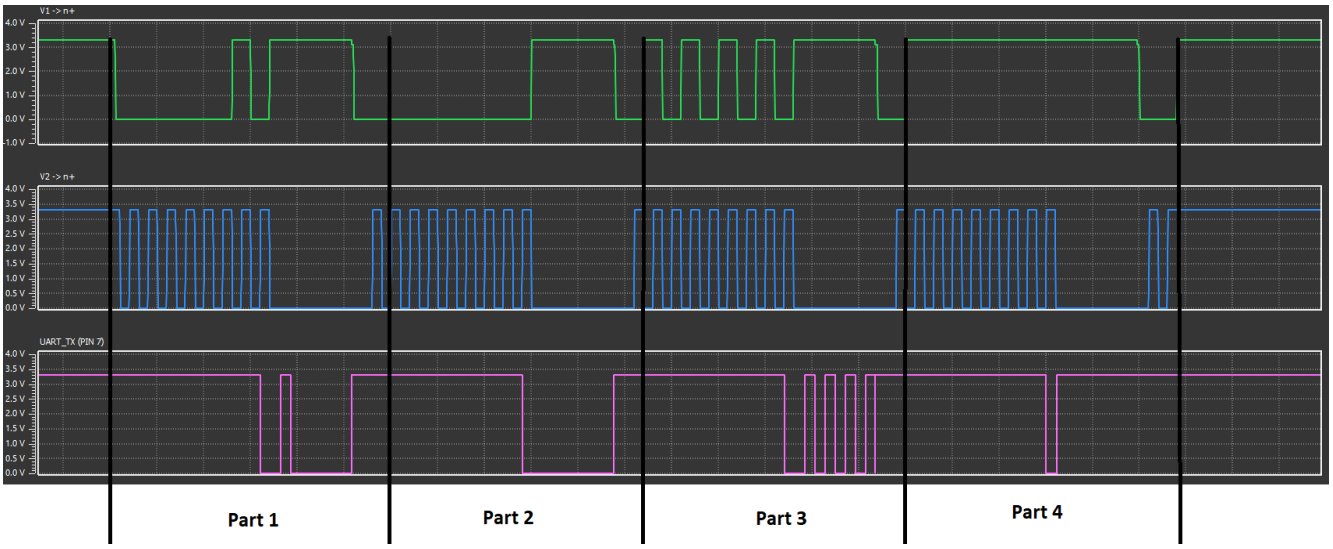

Figure 7: Simulation Result

4. Conclusions

This application note demonstrates how to create an I2C to UART converter using a GreenPAK IC. The design is limited to a message size of 8 bits. SLG46855 has a lot of digital resources in small package which can be used to create unique digital designs.

5. Revision History

Revision	Date	Description
1.00	Sep 28, 2022	Initial release.

IMPORTANT NOTICE AND DISCLAIMER

RENESAS ELECTRONICS CORPORATION AND ITS SUBSIDIARIES (“RENESAS”) PROVIDES TECHNICAL SPECIFICATIONS AND RELIABILITY DATA (INCLUDING DATASHEETS), DESIGN RESOURCES (INCLUDING REFERENCE DESIGNS), APPLICATION OR OTHER DESIGN ADVICE, WEB TOOLS, SAFETY INFORMATION, AND OTHER RESOURCES “AS IS” AND WITH ALL FAULTS, AND DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT OF THIRD-PARTY INTELLECTUAL PROPERTY RIGHTS.

These resources are intended for developers who are designing with Renesas products. You are solely responsible for (1) selecting the appropriate products for your application, (2) designing, validating, and testing your application, and (3) ensuring your application meets applicable standards, and any other safety, security, or other requirements. These resources are subject to change without notice. Renesas grants you permission to use these resources only to develop an application that uses Renesas products. Other reproduction or use of these resources is strictly prohibited. No license is granted to any other Renesas intellectual property or to any third-party intellectual property. Renesas disclaims responsibility for, and you will fully indemnify Renesas and its representatives against, any claims, damages, costs, losses, or liabilities arising from your use of these resources. Renesas' products are provided only subject to Renesas' Terms and Conditions of Sale or other applicable terms agreed to in writing. No use of any Renesas resources expands or otherwise alters any applicable warranties or warranty disclaimers for these products.

(Disclaimer Rev.1.01)

Corporate Headquarters

TOYOSU FORESIA, 3-2-24 Toyosu,
Koto-ku, Tokyo 135-0061, Japan
www.renesas.com

Contact Information

For further information on a product, technology, the most up-to-date version of a document, or your nearest sales office, please visit www.renesas.com/contact-us/.

Trademarks

Renesas and the Renesas logo are trademarks of Renesas Electronics Corporation. All trademarks and registered trademarks are the property of their respective owners.