

Application Note

Servo Demultiplexer

AN-CM-280

Abstract

This application note describes how to create a model servo motor signal demultiplexer using a Dialog GreenPAK IC.

This application note comes complete with design files which can be found in the References section.

Servo Demultiplexer

Contents

Abstract	1
Contents	3
Figures	3
Tables	3
1 Terms and Definitions	4
2 References	4
3 Introduction	5
4 Servo Motor 1 RF Channel Multiplex	6
5 GreenPAK Design Schematic	8
5.1 GreenPAK Timing Blocks	9
5.2 GreenPAK ASM	9
6 GreenPAK Design Pinout	11
7 Test Results	12
8 Conclusion and Results Discussion	13
Appendix A Source Code	14
Revision History	18

Figures

Figure 1: SG90 Model Servo Motor	5
Figure 2: Time Multiplex – 8x 90° @ 50kS/s	6
Figure 3: Time Multiplex – 7x 90°, #5 =0° @ 50kS/s.....	7
Figure 4: Top View of the GreenPAK Design Schematic	8
Figure 5: Close-up of Timing Blocks Settings.....	9
Figure 6: Close View of ASM Settings	10
Figure 7: Pin Configuration - STQFN20L.....	11
Figure 8: Input/Output - Measurement.....	12
Figure 9: SLG46537V Resources Used	13
Figure 10: Logic Analyzer – All Signals	13

Tables

Table 1: Design Pinout.....	11
-----------------------------	----

Servo Demultiplexer

1 Terms and Definitions

ASIC	Application specific integrated circuit
ASM	Asynchronous state machine
CPLD	Complex programmable logic device
ICs	Integrated Circuits
MCU	Microcontroller

2 References

For related documents and software, please visit:

<https://www.dialog-semiconductor.com/configurable-mixed-signal>

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, Dialog Semiconductor
- [2] [AN-CM-280 Servo Demultiplexer.gp](#), [GreenPAK Design File](#), Dialog Semiconductor
- [3] [GreenPAK Development Tools](#), [GreenPAK Development Tools Webpage](#), Dialog Semiconductor
- [4] [GreenPAK Application Notes](#), [GreenPAK Application Notes Webpage](#), Dialog Semiconductor

Servo Demultiplexer

3 Introduction

Servo Motors are widely used for commercial and industrial applications as linear or rotary actuators. In this example the SG90 model is used, which is low cost, consumes little power, and is lightweight. This makes these type of servos ideal for consumer device RC toys such as RF-controlled race cars, airplanes, and others.

However, to properly control a servo motor, the electronic driver must generate appropriate voltage patterns to the servo motor DATA pin. The waveforms should be pulses shorter than 2 milliseconds, repeating every 20 milliseconds for one servo motor.

This application note will present the design of one 1 to 8 servo motor signal demultiplexer with the SLG46537V [GreenPAK™](#) IC. The designed signal demultiplexer would drive up to 8 servo motors on 8 output pins with the multiplexed signal coming in on 1 input pin. The SLG46537V can also integrate additional functionality, such as additional logic or voltage monitoring, depending upon the system requirements.

The following sections will show:

- A servo motor 1 RF channel signal multiplex;
- The SLG46537V [GreenPAK](#) servo demultiplexer design in detail;
- How to drive 8 servo motors with 1 [GreenPAK](#) device.



Figure 1: SG90 Model Servo Motor

Servo Demultiplexer

4 Servo Motor 1 RF Channel Multiplex

The standard for small 9g servo motors is that they operate from 0.5 ms ~ 1.5 ms HIGH pulses with a period of 20 milliseconds. 0.5 ms correlates to 0°, 1 ms to 90°, and 1.5 ms to 180°.

In this way, if we reserve a 2 milliseconds window for each motor pulse, 8 ms x 2 ms can be used for 8 motor pulse windows with the remaining 4 ms silence at the end used for synchronization. [Figure 2](#) shows a time multiplex period of 20 ms for 8 motors, all at 90° position (1 ms pulses). [Figure 3](#) shows a time multiplex period of 20 ms for 8 motors, all at 90° position (1 ms pulses) except for motor #5 at a 0° position (0.5 ms pulse).

[Figure 2](#) and [Figure 3](#) are waveforms of signed digital samples generated by a Python script for a sampling rate of 50,000 Hz.

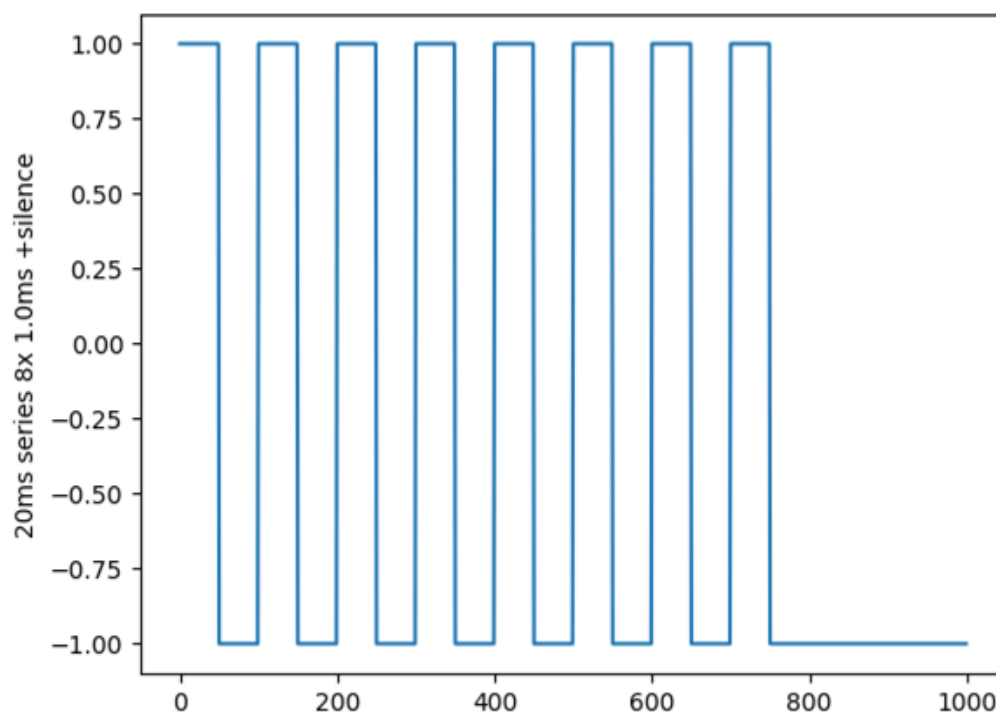


Figure 2: Time Multiplex – 8x 90° @ 50kS/s

Servo Demultiplexer

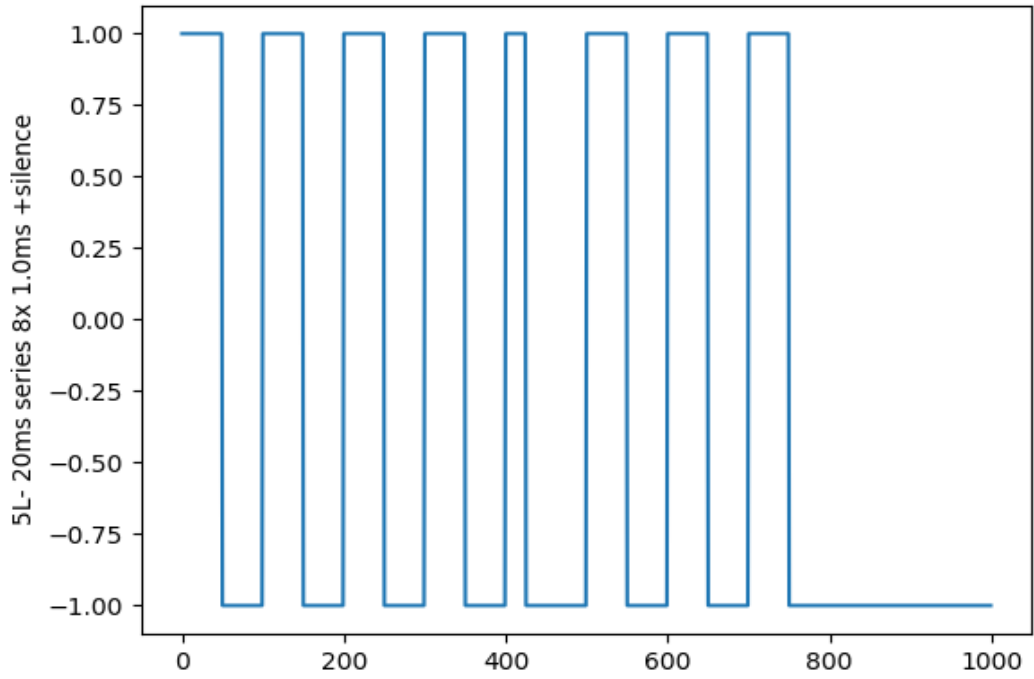


Figure 3: Time Multiplex – 7x 90°, #5 = 0° @ 50kS/s

Servo Demultiplexer

5 GreenPAK Design Schematic

The schematic of the GreenPAK design is shown in Figure 4. The fundamental blocks of the design are the internal Oscillator, 2x Counter, 2x Filter, 8x Flip-Flop, ASM with reset, 8x Pins with OE and Pull-downs, input pin and supply pins.

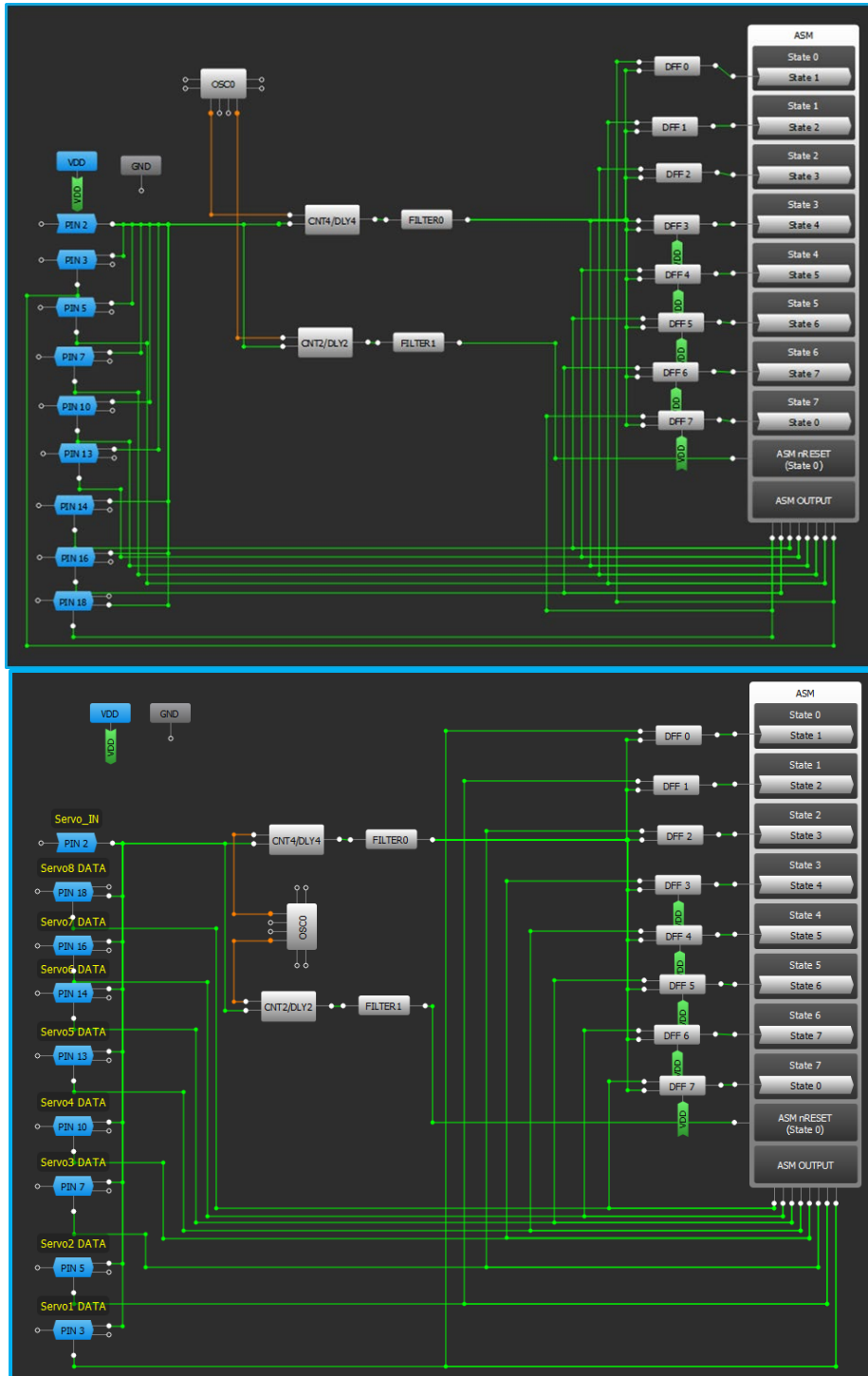


Figure 4: Top View of the GreenPAK Design Schematic

Servo Demultiplexer

5.1 GreenPAK Timing Blocks

The OSC0 internal oscillator is used. It drives the CNT2 counter with $2\text{ MHz}/8/64 \approx 3.9\text{ kHz}$ and the CNT4 counter with $2\text{ MHz}/8/4 \approx 62.5\text{ kHz}$. Settings are shown in Figure 5. CNT2 is set to trigger repeatedly on all falling edges (Delay Mode) and give a Non-inverted OUT after 4.096 ms. This is the 4 ms silence detection at the end of our 20 ms time multiplex period, which resets the ASM to State 0. CNT4 is set to trigger once every falling edge (One Shot Mode) after 96 μs to give a short pulse to all Flip-Flop CLOCK inputs. Only the Flip-Flop of the current state has a HIGH input and will trigger the the ASM to transition to the following state.



The screenshot shows the configuration for four timing blocks in the GreenPAK design:

- OSC0:** Control pin mode: Force on; OSC power mode: Force Power On; Clock selector: OSC; EXT CLK Pin selector: PIN 20 (IO17); Fast start-up: Enable; RC OSC frequency: 2 MHz; 'CLK' predivider by: 8; 'OUT0' second divider by: 64; 'OUT1' second divider by: 64.
- 3-bit LUT7/8-bit CNT4/DLY4:** Type: CNT/DLY; Mode: One shot; Counter data: 5; Pulse width (typical): 96 μs ; Edge select: Falling; Output polarity: Inverted (nOUT); Q mode: None; Stop and restart: None; Connections: Clock: OSC0 CLK /4; Clock source: RC OSC Freq. /8 /4; Clock frequency: 62.5 kHz.
- 3-bit LUT5/8-bit CNT2/DLY2:** Type: CNT/DLY; Mode: Delay; Counter data: 15; Delay time (typical): 4.096 ms; Edge select: Falling; Output polarity: Non-inverted (OUT); Q mode: None; Stop and restart: None; Connections: Clock: OSC0 CLK /64; Clock source: RC OSC Freq. /8 /64; Clock frequency: 3.90625 kHz.
- FILTER0/EDGE DET0:** Type: FILTER; Output polarity: Normal; Information table:

VDD (V)	Delay (ns)	Pulse Width (ns)
1.8	210	<114
3.3	83	<47
5.0	55	<30
- FILTER1/EDGE DET1:** Type: FILTER; Output polarity: Normal; Information table:

VDD (V)	Delay (ns)	Pulse Width (ns)
1.8	210	<75
3.3	83	<30
5.0	55	<19

A block diagram at the bottom shows the connections: OSC0 is connected to CNT2/DLY2 and FILTER0. CNT2/DLY2 is connected to FILTER0.

Figure 5: Close-up of Timing Blocks Settings

5.2 GreenPAK ASM

The Flip-Flops in the GreenPAK design are used to change the asynchronous state machine into a synchronous machine. As described in the previous section, CNT/DLY4 delivers a short pulse to all Flip-Flop CLOCK inputs, but only the Flip-Flop of the current state has a HIGH input and will trigger the transition to State +1. ASM runaway thru more than one state is prevented since all the other Flip-Flops were just loaded with LOW inputs, so the ASM has to wait until the next pulse for the next transition. This is necessary since all state transition conditions are the same 1 condition. Settings are shown in Figure 6.

Servo Demultiplexer

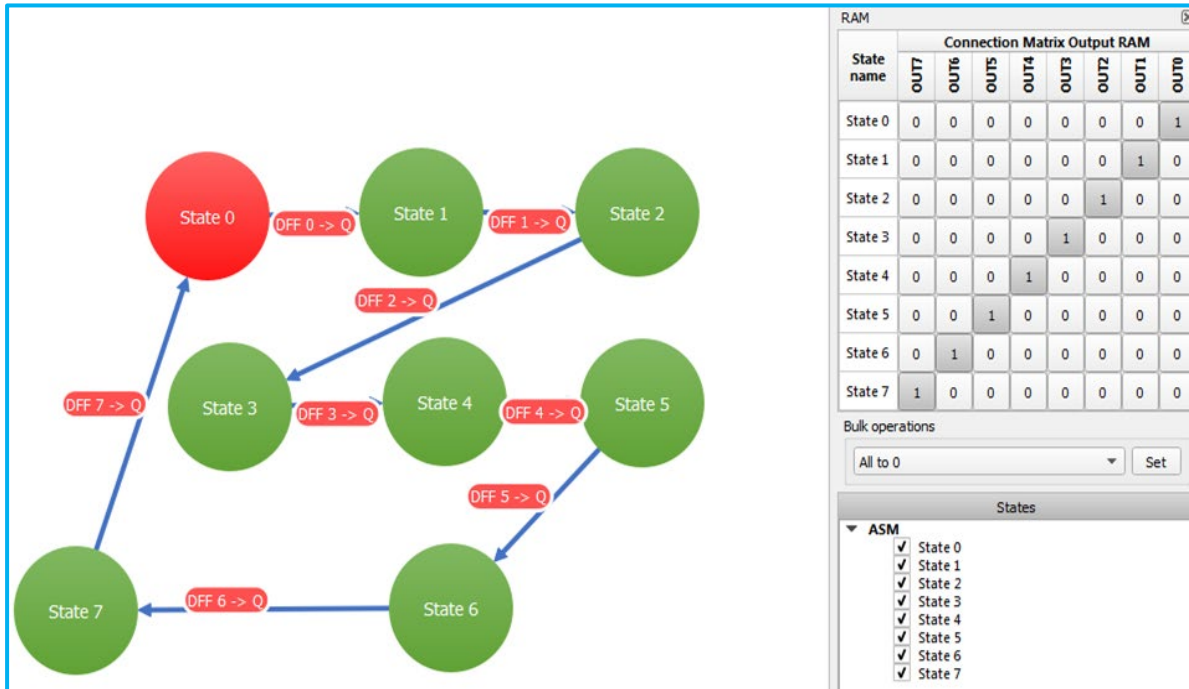


Figure 6: Close View of ASM Settings

Servo Demultiplexer

6 GreenPAK Design Pinout

The Signal input IO0 is configured as a Digital in without Schmitt trigger. The ServoX DATA outputs are configured as 1x Push-Pull at OE = 1, at OE = 0 they are inputs with a 10 k Pull-down resistor. Pinout is shown in Table 1 and Figure 7.

Table 1: Design Pinout

Pin #	Signal Name	Pin Function
1	V _{DD}	+5 V Supply
2	IO0	Signal Input
3	IO1	Servo1 DATA
4	IO2	
5	IO3	Servo2 DATA
6	IO4	
7	IO5	Servo3 DATA
8	IO6	
9	IO7	
10	IO8	Servo4 DATA
11	GND	Ground
12	IO9	
13	IO10	Servo5 DATA
14	IO11	Servo6 DATA
15	IO12	
16	IO13	Servo7 DATA
17	IO14	
18	IO15	Servo8 DATA
19	IO16	
20	IO17	

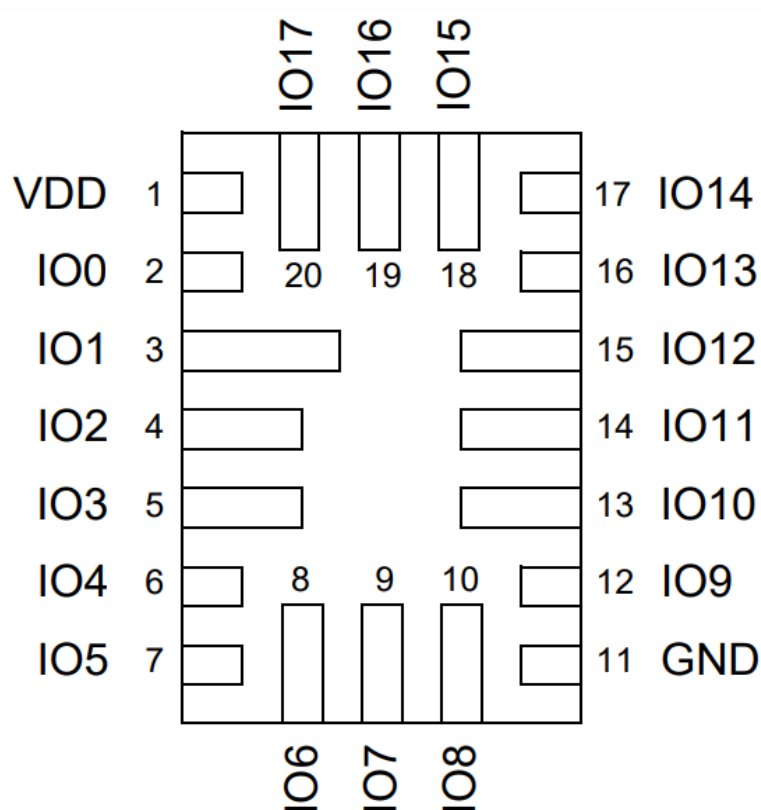


Figure 7: Pin Configuration - STQFN20L

Servo Demultiplexer

7 Test Results

The Signal input is driven by an amplified Audio Out signal (0 ~ 5 V) generated by a Python script (Appendix A). Figure 8 shows the Signal Input in yellow and the Servo1 DATA output in blue. The generated .wav file has 20 seconds of choreographed servo movements and the GreenPAK design decodes all 8 ServoX DATA signals accordingly which is seen by movements of the servo motors.

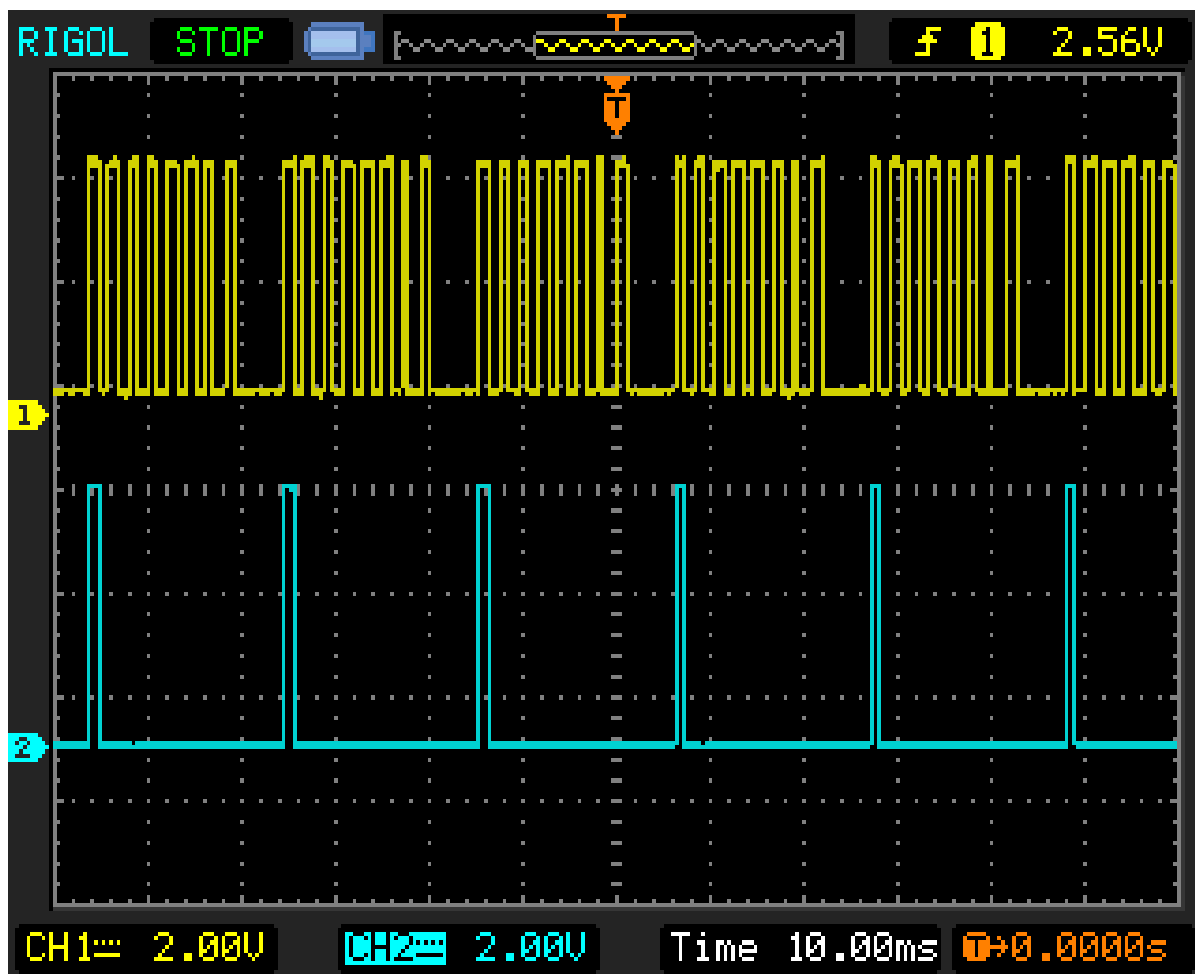


Figure 8: Input/Output - Measurement

Servo Demultiplexer

8 Conclusion and Results Discussion

The Design of a Model Servo Motor Demultiplexer was presented. Through a GreenPAK design with the SLG46537V we have successfully implemented a lightweight, low-power, cost-effective solution. Figure 9 shows the resource usage of the SLG46537V. The design successfully decodes all 8 ServoX DATA signals from one Time Multiplexed signal input - Figure 10.

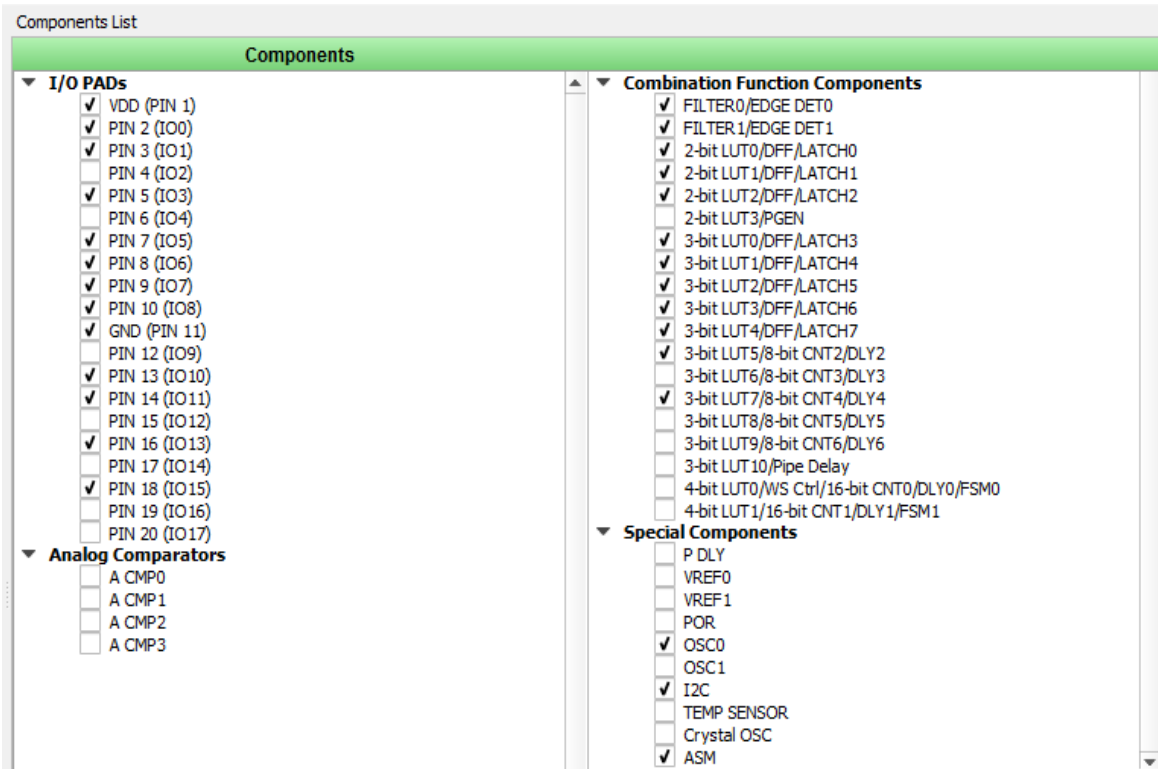


Figure 9: SLG46537V Resources Used

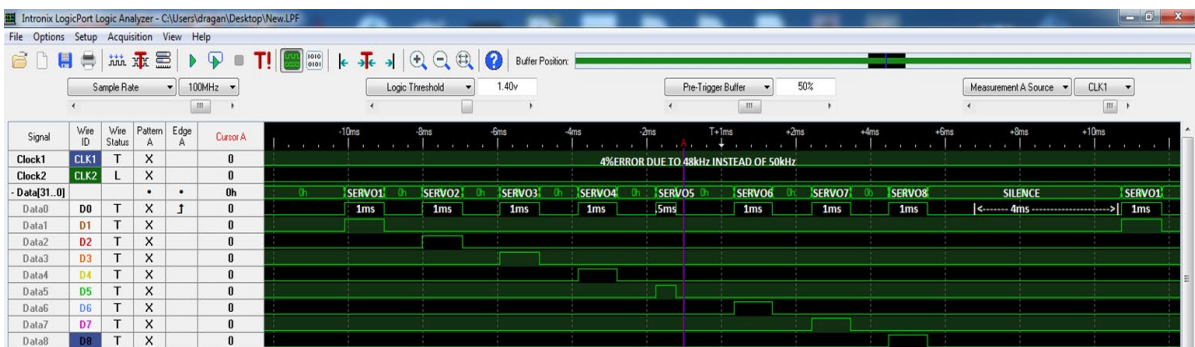


Figure 10: Logic Analyzer – All Signals

Servo Demultiplexer

Appendix A Source Code

```

#servoMUX.py#####BEGIN#
import wave
import numpy as np
import matplotlib.pyplot as plt

def repeat48x(series):
    return np.concatenate(( series, series, series, series, series, series, series, series, series, series, se-
ries, series, series, series, series, series, series, series, series, series, series, series, series, se-
ries, series, series, series, series, series, series, series, series, series, series, series, series, series, se-
ries, series, series, series, series, series, series, series, series, series, series), axis=None)

silence=-np.ones(200)

pulse10=np.concatenate((np.ones(50), -np.ones(50)), axis=None)
plt.plot(pulse10)
plt.ylabel('pulse 1.0ms')
plt.show()
pulse05=np.concatenate((np.ones(25), -np.ones(75)), axis=None)
plt.plot(pulse05)
plt.ylabel('pulse 0.5ms')
plt.show()
pulse15=np.concatenate((np.ones(75), -np.ones(25)), axis=None)
plt.plot(pulse15)
plt.ylabel('pulse 1.5ms')
plt.show()

#second10
series=np.concatenate((pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('20ms series 8x 1.0ms +silence')
plt.show()
second10=repeat48x(series)
plt.plot(second10)
plt.ylabel('1s- 20ms series 8x 1.0ms +silence')
plt.show()

```

Servo Demultiplexer

```
#second1L
series=np.concatenate((pulse05, pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('1L- 20ms series 8x 1.0ms +silence')
plt.show()
second1L=repeat48x(series)
plt.plot(second1L)
plt.ylabel('1L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second2L
series=np.concatenate((pulse10, pulse05, pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('2L- 20ms series 8x 1.0ms +silence')
plt.show()
second2L=repeat48x(series)
plt.plot(second2L)
plt.ylabel('2L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second3L
series=np.concatenate((pulse10, pulse10, pulse05, pulse10, pulse10, pulse10, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('3L- 20ms series 8x 1.0ms +silence')
plt.show()
second3L=repeat48x(series)
plt.plot(second3L)
plt.ylabel('3L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second4L
series=np.concatenate((pulse10, pulse10, pulse10, pulse05, pulse10, pulse10, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('4L- 20ms series 8x 1.0ms +silence')
plt.show()
second4L=repeat48x(series)
```

Servo Demultiplexer

```
plt.plot(second4L)
plt.ylabel('4L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second5L
series=np.concatenate((pulse10, pulse10, pulse10, pulse10, pulse05, pulse10, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('5L- 20ms series 8x 1.0ms +silence')
plt.show()
second5L=repeat48x(series)
plt.plot(second5L)
plt.ylabel('5L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second6L
series=np.concatenate((pulse10, pulse10, pulse10, pulse10, pulse10, pulse05, pulse10, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('6L- 20ms series 8x 1.0ms +silence')
plt.show()
second6L=repeat48x(series)
plt.plot(second6L)
plt.ylabel('6L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second7L
series=np.concatenate((pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, pulse05, pulse10, si-
lence), axis=None)
plt.plot(series)
plt.ylabel('7L- 20ms series 8x 1.0ms +silence')
plt.show()
second7L=repeat48x(series)
plt.plot(second7L)
plt.ylabel('7L-1s- 20ms series 8x 1.0ms +silence')
plt.show()
#second8L
series=np.concatenate((pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, pulse10, pulse05, si-
lence), axis=None)
plt.plot(series)
```

Servo Demultiplexer

```

plt.ylabel('8L- 20ms series 8x 1.0ms +silence')
plt.show()
second8L=repeat48x(series)
plt.plot(second8L)
plt.ylabel('8L-1s- 20ms series 8x 1.0ms +silence')
plt.show()

w=wave.open("servo.wav", 'wb')
w.setnchannels(1)
w.setsampwidth(1)
w.setframerate(48000)
#w.setnframes(24000)
#w.setcomptype('NONE', 'wav')
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second1L.astype(np.int8))
w.writeframesraw(100*second2L.astype(np.int8))
w.writeframesraw(100*second3L.astype(np.int8))
w.writeframesraw(100*second4L.astype(np.int8))
w.writeframesraw(100*second5L.astype(np.int8))
w.writeframesraw(100*second6L.astype(np.int8))
w.writeframesraw(100*second7L.astype(np.int8))
w.writeframesraw(100*second8L.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.writeframesraw(100*second10.astype(np.int8))
w.close()
#servoMUX.py#####END#

```


Revision History

Revision	Date	Description
1.0	17-May-2019	Initial Version

Servo Demultiplexer

Status Definitions

Status	Definition
DRAFT	The content of this document is under review and subject to formal approval, which may result in modifications or additions.
APPROVED or unmarked	The content of this document has been approved for publication.

Disclaimer

Unless otherwise agreed in writing, the Dialog Semiconductor products (and any associated software) referred to in this document are not designed, authorized or warranted to be suitable for use in life support, life-critical or safety-critical systems or equipment, nor in applications where failure or malfunction of a Dialog Semiconductor product (or associated software) can reasonably be expected to result in personal injury, death or severe property or environmental damage. Dialog Semiconductor and its suppliers accept no liability for inclusion and/or use of Dialog Semiconductor products (and any associated software) in such equipment or applications and therefore such inclusion and/or use is at the customer's own risk.

Information in this document is believed to be accurate and reliable. However, Dialog Semiconductor does not give any representations or warranties, express or implied, as to the accuracy or completeness of such information. Dialog Semiconductor furthermore takes no responsibility whatsoever for the content in this document if provided by any information source outside of Dialog Semiconductor.

Dialog Semiconductor reserves the right to change without notice the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications. Notwithstanding the foregoing, for any automotive grade version of the device, Dialog Semiconductor reserves the right to change the information published in this document, including, without limitation, the specification and the design of the related semiconductor products, software and applications, in accordance with its standard automotive change notification process.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. Dialog Semiconductor makes no representation or warranty that such applications, software and semiconductor products will be suitable for the specified use without further testing or modification. Unless otherwise agreed in writing, such testing or modification is the sole responsibility of the customer and Dialog Semiconductor excludes all liability in this respect.

Nothing in this document may be construed as a license for customer to use the Dialog Semiconductor products, software and applications referred to in this document. Such license must be separately sought by customer with Dialog Semiconductor.

All use of Dialog Semiconductor products, software and applications referred to in this document is subject to Dialog Semiconductor's Standard Terms and Conditions of Sale, available on the company website (www.dialog-semiconductor.com) unless otherwise stated.

Dialog, Dialog Semiconductor and the Dialog logo are trademarks of Dialog Semiconductor Plc or its subsidiaries. All other product or service names and marks are the property of their respective owners.

© 2019 Dialog Semiconductor. All rights reserved.

Contacting Dialog Semiconductor

United Kingdom (Headquarters)

Dialog Semiconductor (UK) LTD
Phone: +44 1793 757700

Germany

Dialog Semiconductor GmbH
Phone: +49 7021 805-0

The Netherlands

Dialog Semiconductor B.V.
Phone: +31 73 640 8822

Email:

enquiry@diasemi.com

North America

Dialog Semiconductor Inc.
Phone: +1 408 845 8500

Japan

Dialog Semiconductor K. K.
Phone: +81 3 5769 5100

Taiwan

Dialog Semiconductor Taiwan
Phone: +886 281 786 222

Web site:

www.dialog-semiconductor.com

Hong Kong

Dialog Semiconductor Hong Kong
Phone: +852 2607 4271

Korea

Dialog Semiconductor Korea
Phone: +82 2 3469 8200

China (Shenzhen)

Dialog Semiconductor China
Phone: +86 755 2981 3669

China (Shanghai)

Dialog Semiconductor China
Phone: +86 21 5424 9058