THE GREENPAK™ COOKBOOK

April 2020
Abstract

This document explores a wide variety of techniques and applications that can be performed by Dialog Semiconductor’s GreenPAK Mixed-Signal ICs.

Introduction to GreenPAK

Dialog Semiconductor’s GreenPAK ICs are a family of Configurable Mixed-signal ICs (CMICs) that provide a small, cost-friendly, and personalized solution to many common problems that system-level circuit designers face. GreenPAK provides a means of considerably reducing PCB size, BOM cost, and design lead time.

<table>
<thead>
<tr>
<th>Original Design</th>
<th>w/GreenPAK</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>Approx. Difference w/GreenPAK</td>
</tr>
<tr>
<td>Layout Size</td>
<td>17.8 mm²</td>
</tr>
<tr>
<td>Cost Savings</td>
<td>$1.33</td>
</tr>
</tbody>
</table>

Due to the configurability and various features of GreenPAKs, narrowing the scope of possible applications can be difficult. With the right motivation and know-how, a designer can add a GreenPAK into almost any application within most industries.

This document is designed to bolster this motivation and know-how: we provide a "cookbook" to designers to help them see where a GreenPAK can be used within their projects. This document outlines different techniques and provides completed applications to help designers use GreenPAK on their own.

Cookbook Structure

The main subsections within this document are organized into two different types: Techniques and Applications. Techniques focus on a task that can be accomplished using only one or a few macrocell blocks. Application sections describe how techniques can be meshed together to create real, valuable applications. Generally, the easiest techniques and applications will be at the beginning of a chapter.

Each application has an associated GreenPAK Designer file that can be viewed and edited.

Making Your Own Design With the Cookbook

The applications outlined in the cookbook are simple realizations of real-world applications. However, GreenPAK ICs have the necessary macrocells and functionality to add far more value than the designs in this cookbook. Dialog Semiconductor has helped designers create thousands of unique designs, where simple applications both similar and different to the cookbook applications were expanded, combined, and personalized.

For example, the Application: Basic Sequencer can be combined with many of the applications within Chapter 4, Safety Features to create a self-regulating, customized sequencing application.

Original Design w/GreenPAK

The resulting integrated solution is more complex but doesn’t come close to using all the available macrocells. With the full GreenPAK family of ICs at your disposal, the number of permutations and modifications available for the designs in this cookbook are endless. Whether you wish to completely reuse a design shown in this cookbook, or you’d rather incorporate some of the techniques in this paper into your own design, feel free. After all, it’s your recipe.
Dialog Semiconductor

Contents

Cookbook Structure...................................................... 2
Making Your Own Design With the Cookbook.......................... 3
Table of Contents.......................................................... 4-5

Chapter 1: Basic Blocks & Functions................................. 6
Learning More about a Macrocell........................................... 7
Digital Macrocells............................................................ 7
Configuring Standard Logic w/ LUT Macrocells..................... 8
Oscillators........................................................................ 8
Analog Comparators.......................................................... 8
I/Os.............................................................................. 9
Interconnections............................................................ 9
Simulation and Emulation................................................... 10
OE Pin.........................................................................11
Parity Bit Generator.......................................................... 12-13
8-bit Multiplexer............................................................. 14-15
Demultiplexer............................................................... 16-17

Chapter 2: Timing Functions............................................. 18
Optimizing CNT/DLY Accuracy.......................................... 19
Sequencing CNT/DLY Blocks............................................ 20
System Reset............................................................... 22-23
Several-button Reset...................................................... 24-25
Basic Sequencer........................................................... 26-27
Cascaded Sequencer...................................................... 28-29
Power Sequencer.......................................................... 30-31
Frequency Divider........................................................ 32-33
10 Year Counter........................................................... 34-35
Global RC Oscillator Controller......................................... 36-37
Global Crystal Oscillator Controller.................................... 38-39

Chapter 3: Signal Conditioning........................................ 40
Using a CNT/DLY Block as a Deglitch Filter....................... 41
Wake-Sleep Controller.................................................... 42
Interrupt Controller........................................................ 44-45
Ship Mode Controller..................................................... 46-47
Creating a Bidirectional Counter....................................... 48
Encoder.......................................................................... 50-51
Multiplexing a Bitstream.................................................... 52
Creating a Synchronous State Machine from an ASM........ 53
N-Length Bitstream.......................................................... 54-55
Distance Sense............................................................. 56-57
Frequency Range Detector............................................... 58-59

Chapter 4: Safety Features.............................................. 60
Reducing ACMP Power Consumption.................................. 61-62
Zero-Voltage Cross Detection............................................ 63
Window Comparator........................................................ 64-65
Over Temperature Protection............................................ 66-67
Battery Charge Indicator................................................... 68-69
Watchdog Timer............................................................ 70-71
Voltage Level Detection.................................................... 72-73
Power Backup Management.............................................. 74-75

Chapter 5: Communication Protocols................................. 76
Changing Your Design with I2C........................................ 77
Creating an I2C Command................................................ 78
Using Serial to Parallel Interface (SPI) block..................... 80-81
Level Shifting............................................................... 82
Sending a Preset Number of Pulses................................. 83
Building a Shift Register................................................. 84
I2C GPIO Expansion........................................................ 86-87
Serial to Parallel (External Clock)...................................... 88-89
Serial to Parallel (Internal Clock)....................................... 90-91
Parallel to Serial.......................................................... 92-93
Bi-directional Communication (Transmit First).................... 94-95
Bi-directional Communication (Receive First)...................... 96-97
7-Segment Display Using ASM and I2C............................ 98-99
Communication MUX Using I2C........................................ 100-101
I2C Level Shifter.......................................................... 102-103

Chapter 6: Pulse-based Control...................................... 104
Setting a Constant Duty Cycle........................................... 105
H-Bridge Control............................................................ 106-107
Constant Current LED Driver.......................................... 108-109
LED Control via I2C....................................................... 110-111
Charge Pump............................................................... 112-113
Two-stage Charge Pump.................................................. 114-115
Using DCMP and PWM Macrocell in PWM Mode............ 116
PWM Selection............................................................ 118-119

PWM Generator Using ACMP and DAC............................ 120-121
PWM Generator Using ADC............................................. 122-123
Creating a Breathing LED Pattern....................................... 124
Creating RGB LED......................................................... 126-127
Duty Cycle Detection....................................................... 128
RGB LED Control via I2C................................................. 130-131
Associated GreenPAK Designer Files............................... 132

Techniques are italicized
Chapter 1

Basic Blocks & Functions

This section will outline the most common building blocks of GreenPAKs. Each macrocell type will be given an overview and may include simple techniques and applications for using the block. Further information on specific cells can be found in the particular GreenPAK’s datasheet, or the Help info within the GreenPAK Designer software.

Technique: Learning More About a Macrocell

This technique will work with any version of GreenPAK Designer.

This section provides a brief overview and several techniques for the most common blocks in GreenPAKs. However, it may happen that you wish to learn more about a specific macrocell. This can be done by selecting the macrocell in the GreenPAK Designer, then clicking the Information button (Figure 2) at the bottom-left of the Properties window.

Overview: Digital Macrocells

Digital Macrocells are the basic functional components of any GreenPAK. They include:

- Look-Up Tables (LUTs)
- D Flip-Flop (DFF) / Latch
- Counter / Delay (CNT/DLY)

Communication:

- I2C (many devices)
- SPI (select devices)

Less Common:

- Pattern Generator (PGEN)
- Pipe Delay
- Programmable delay (PDLY)
- Filter / Edge Detector

Many of the components in GreenPAK Designer can be selected to be one of multiple types of macrocells. This is indicated by the name of the digital macrocell: for example, “2-bit LUT0/DFF/LATCH0” can be, as the name implies, a LUT, DFF, or a Latch. The selection of macrocell type is configured using the Type option in the Properties window.
Technique: Configuring Standard Logic w/ LUT Macrocells

This technique will work with any GreenPAK.

Look-up tables can be used in GreenPAK Designer to configure any digital logic for a two, three or four input, single output logic macrocell. The logic configuration can be edited in the Properties window.

Most logic implemented in GreenPAK designs is standardized logic, such as MUX, AND, OR, etc. To expedite these common configurations, the Properties window has a Standard gates option that can automatically convert the logic table into a standard gate configuration. If the Regular shape option is left unchecked the LUT shape will change to the standardized gate symbol.

Overview: Oscillators

GreenPAK ICs contain at least two oscillators; the more recent models, such as the SLG46826, have three oscillators. The most common, non-divided frequencies of the oscillators within GreenPAK are:

- 2KHz low speed, low power oscillator
- 2MHz medium speed
- 25MHz high speed

Each oscillator has several outputs, each with several pre-dividers to allow flexibility in clocking. To save power Auto-power on allows you to turn off the oscillator when the clock is not needed.

More information about oscillators can be found by using the Information button when the component is selected.

Overview: Analog Comparators

Almost every GreenPAK is equipped with two or more analog comparators (ACMPs), each with two input sources; IN+ and IN-. The input source to each can be configured in the Properties window:

1. More information about analog comparators can be found by using the Information button when the component is selected.

Overview: I/Os

Most I/Os within GreenPAK are very flexible. The I/O capabilities vary significantly from pin to pin and part to part, so a design concept should be mapped to the necessary pin configuration before committing to a specific GreenPAK IC.

Outputs can be configured to be Push-pull or open-drain in either a NMOS or PMOS configuration. A scaling factor, such as 2x, indicates that the output strength is doubled. Additionally, pull-up and pull-down resistor options of 10kΩ, 100kΩ, and 1MΩ are available on output pins.

Multiple input options are available as well, such as: Digital-In, Digital-In with Schmitt trigger, Low Voltage Digital-In and Analog-In. Analog in is used as an input to an ACMP.

Overview: Interconnections

Interconnection with GreenPAK Designer is easy. The system will guide you on which connections you can make. When you click on any connection point, the system:

1. Highlights all available connections in green.
2. Gives you a "rubber band" connection that you can stretch to any of these green connections.
3. This results in a green wire to show you interconnections you have made.
When developing a design it's important to be able to quickly test the functionality. GreenPAK Designer makes debugging effective and easy.

There are two ways to quickly check your design:

1. Simulation
2. Emulation

Simulation simulates the operation of the circuit in conditions depictive of reality without the need of a physical IC. It should be kept in mind that simulation can't provide for all the nuances of a real-world system.

Emulation, with the presence of a demo board and the GreenPAK chip, allows you to check the operation of your design directly in the hardware without permanently programming a part. This allows you to quickly make changes to the project and use your emulation to check your guesswork.

1. If the design is ready for debug select the Debug button (Figure 1) to go to the emulation/simulation selection menu.
2. Next, select the platform with which you want to check your design (Figure 2).
3. After selecting a platform, go to the debug menu, where further actions will be suggested depending upon the platform you choose (Figure 3).
4. If you need to change the platform, you can do this at any time by selecting Change platform (Figure 3).

Technique: OE Pin

This technique can be used within any GreenPAK with OE pins.

Typically GreenPAK I/O can be configured as an input or output. Output enable (OE) pins are select pins within most GreenPAK that allow the pin to dynamically change between a Digital input and Digital output (Figure A). When the OE GPIO is set as a permanent input the OE pin is set to ground and if the GPIO is set as a permanent output the OE pin is set to VDD. Setting the GPIO as a digital input/output allows for this selection to be made in the matrix.

Setting a GPIO as a digital input/output allows for two-way communication. It also allows for the GPIO to be set to Hi-Z in addition to a logical high and low.

If the GPIO is used for two-way communication, it's important to implement a timing circuit for OE selection. In the example circuit below the OE pins of CLK_IO and Data_IO are switched from low to high after CNT2 sees 8 clocks, consequently setting the OE pins as outputs to transmit the internal signals. After another 8 clocks the OE pins are reset low, setting them as inputs again to receive an external signal.
Application:
Parity Bit Generator

Ingredients
Any GreenPAK
No other components are needed

Design Steps
1. Connect input pins using XOR gates using Technique: Configuring Standard Logic w/ LUT Macrocells. XOR gates can be used to calculate the running sum of 1's.
2. Add logic for the ENABLE signal.

Parity Bit Generators are used to check the integrity of a signal; it is the simplest implementation of a Cyclic Redundancy Check (CRC). Parity Bits are used prior to committing data to an MCU or other control unit to ensure the incoming data hasn't been corrupted.
A Multiplexer, or MUX, is used to select an output from multiple input signals. This is used in applications where several communication lines need to be sent across a single line. By using a GreenPAK as a MUX the latency time in transmission can be in nanoseconds, comparable to discrete logic IC’s.

### Application:

**8-bit Multiplexer**

#### Ingredients

Any GreenPAK

No other components are needed

#### Design Steps

1. Connect input pins to 4 LUT’s configured as multiplexers using Technique: Configuring Standard Logic w/ LUT Macrocells. INx should connect to A or B. SEL0 should connect to S on all 4 mux’s.

2. Add second and third stage cascading multiplexer blocks to create the more significant SEL bits.

3. Add an output pin connected to the last-stage multiplexer.
Application:

Demultiplexer

Ingredients

Any GreenPAK

No other components are needed

Design Steps

1. Connect the IO like the GreenPAK diagram above. For a larger demultiplexer use more pins.
2. Configure the LUTs to each have a unique, active low value. For example, 2-L3 will be LO when both SEL0 and SEL1 are LO.

A Demultiplexer, or demux, is used to select which of several channels is sent an input signal. A demux is used in applications where it is useful to send several different types of data across one line and is commonly found in communication systems.

GreenPAK Diagram
Chapter 2

Timing Functions

Technique: Optimizing CNT/DLY Accuracy

This technique works with any GreenPAK. The accuracy of the oscillator and CNT/DLY blocks vary from part-to-part.

GreenPAK ICs, like all chips with internal oscillators, have inherent variation in timing. This is attributed to factors like manufacturing, temperature and, in the case of GreenPAK, user design practices. By using simple design rules, the accuracy of counters and delays within a GreenPAK design can be improved.

The relationship between the oscillators and CNT/DLY blocks should be considered. The oscillators are global oscillators; they can be used for any number of CNT/DLY blocks and aren’t initially synchronized to the start/stop of a delay or counter. Consequently, when a counter or delay is enabled it will only begin to increment on the next clock edge. This is depicted in Figure 9, where an enable signal for a delay is activated mid-clock-cycle and doesn’t begin to decrement until the next rising edge.

![Figure 9 - Behavior of Rising Enable for Delay](image)

This is factored into the typical delay time calculation for the CNT/DLY blocks:

\[ \text{Delay}_{\text{typ}(\text{typ})} = \frac{(\text{Counter\_Data} + 1) + t}{\text{clock}} \]

Thus, as the value of “Counter\_Data” increases, the influence of “t” on the delay time will be proportionately less. Additionally, the absolute value of the delay time can be kept the same, despite using a larger “Counter\_Data” value, if a faster “clock” value is used. In the Properties window of the selected CNT/DLY block both the Counter Data value and the clock source can be modified.

Additionally, the timing characteristics within the datasheet of the respective GreenPAK should be referenced to ensure factors such as Power-ON time, frequency settling time and percent deviation across temperature are taken into account.
Technique: Sequencing CNT/DLY Blocks

This technique will work with any GreenPAK.

DLY blocks can be chained together to sequence signals. By chaining the output of one delay block to the input of another a sequential set of delays can be made.

The DLY blocks in the sequential set should be set to Delay in the Mode setting within the Properties window. Typically, the Edge select setting should be the same for all sequenced components. Figure 11 shows the effect of the two sequential CNT/DLY blocks set to rising-edge, 8ms delays of the Power-On-Reset (POR) signal.

One can also chain together CNT blocks for a longer counted time. When chaining CNT blocks together, the CLK of the CNT should be driven by the output of the previous counter. This can be done in the properties by selecting a CNT block and in the Properties window, choosing the Clock connection to be sourced from the previous CNTx/DLYx.
Application: System Reset

Ingredients

Any GreenPAK

No other components are needed

Design Steps

1. Configure an I/O as an input for each input signal.
2. Add LUT logic to create a HI signal when any of the lines are active. The logic is dependent on whether each signal is active-hi or active-low.
3. Configure a CNT/DLY block to One shot mode, with Edge select configured to Rising. Set the Counter data to create the desired length of pulse. For an active-low pulse change the Output polarity to Inverted (nOUT).
4. Connect the CNT/DLY block’s output to an output pin.

System Reset ICs are used to provide a reset to a microprocessor during faults, manual resets, brown-outs and more.
Application:

Several-button Reset

Ingredients

Any GreenPAK

No other components are needed

Design Steps

1. Configure an I/O as an input for each button.
2. Add LUT logic to create a HI signal when both buttons are active. The logic is dependent on whether each signal is active-hi or active-low.
3. Configure a CNT/DLY block to Delay mode, with Edge select configured to Rising. Set the Counter data to create the desired length of button hold time. For an active-low pulse change the Output polarity to Non-inverted (OUT).
4. Configure a second CNT/DLY block to One shot mode, with Edge select configured to Rising. Set the Counter data to create the desired length of pulse. For an active-low pulse change the Output polarity to Inverted (nOUT).
5. Connect the CNT/DLY block’s output to an output pin.

Pressing and holding several buttons to initiate a hard reset is a common interface in many devices. Implementing this application in a separate IC ensures the reset will be acknowledged and acted upon, even if the rest of the system is experiencing one or more software, firmware, or hardware issues.
Application:

Basic Sequencer

Ingredients

Any GreenPAK

No other components are needed

Sequencers are used when a designer needs to sequentially activate different portions of a system. Specifically, this can be critical for applications that require several power rails.

Design Steps

1. Use LUTs to configure the desired start-up condition.
2. Use a latch or DFF to maintain the start-up signal so it can be read by DLY blocks.
3. Chain the Delays using Technique: Sequencing CNT/DLY Blocks.
4. Connect each delay channel to the desired output pin/s.
Application:
Cascaded Sequencer

Ingredients
Any GreenPAK
No other components are needed

Design Steps
1. Configure input structure. Here you can select between level and latching control with Pin#17.
2. Multiplex inputs to DLYs within a Multi-function block to achieve a cascaded effect.
3. Connect the DLY outputs to push pull output pins.

Sequencers are used when a designer needs to sequentially activate different portions of a system. This can be critical for applications that require several power rails. It is typical to have a cascaded sequence, such that a rail does not turn off until all the rails below it have turned off.
Application:

Power Sequencer

Ingredients

Any GreenPAK
Six resistors

Power sequencers are used to power up and shut down external devices with specified time intervals in between steps. They can be used when multiple semiconductor devices with different power supply voltages are together on one board, or when multiple pieces of equipment are connected.

Design Steps

1. Configure input pins for the EN and voltage monitoring.
2. Configure output pins to sequence system.
3. Power on ACMPs, connecting POR to PWR UP and configure the IN- source of each with the desired voltage threshold levels.
4. Configure DLVs with the desired delay times.
5. Configure LUTs with the proper logic functions.
Application: Frequency Divider

Ingredients
Any GreenPAK
No other components are needed

Design Steps
1. Configure inputs for each input signal (input frequency, coefficient).
2. Use DFF and Pipe Delay to divide the frequency for the first stage.
3. Use another DFF to divide the signal by a factor for the second stage.
4. Configure LUT logic to decide outputs into specified coefficient.

*Frequency dividers are used to divide the frequency into different coefficients. It can used for improving the performance of electronic countermeasure equipment, communication systems and laboratory instruments.*
**Application:**

**10 Year Counter**

**Ingredients**

Any GreenPAK

*No other components are needed*

Ultra long counters can be used to determine the lifetime of a product without requiring a large tax on the power budget.

**Design Steps**

1. Chain the Counters using Technique: Sequencing CNT/DLY Blocks.
2. Connect input pins and output pin.
3. Set timing in CNT properties.
Application:

Global RC Oscillator Controller

Ingredients

Any dual rail GreenPAK
Two resistors
One capacitor

Global RC oscillators are used to synchronize multiple devices. The resistive and capacitive elements can be variable for specific frequencies.

Design Steps

1. Configure pins as inputs and outputs of oscillator. Feedback output must be a 1x 3-State Output.
2. Configure logic to enable the path between the input and output of the oscillator.
3. Connect resistor and capacitor to feedback output node. The other resistor node should be connected to VDD2 and the other capacitor node should be connected to ground.
4. Connect the other resistor in between VDD and VDD2.
5. Connect the feedback output to the oscillator input.
Global Crystal Oscillator Controller

Ingredients
Any dual rail GreenPAK
Four resistors
Two capacitors
One crystal oscillator

Design Steps
1. Configure pins as inputs and outputs of oscillator. Feedback output must be a 1x 3-State Output.
2. Configure logic to enable the path between the input and output of the oscillator.
3. Connect one node of a resistor to the feedback output and the other node to the crystal oscillator.
4. Connect the other node of the crystal oscillator to the GreenPAK’s oscillator input.
5. Connect each capacitor on either node of the crystal oscillator. Connect the other node of both capacitors to ground.
6. Connect resistor in between VDD and VDD2.
7. Connect resistor in between the feedback output and the oscillator input.
Technique: Using a CNT/DLY Block as a Deglitch Filter

This technique can be used in any GreenPAK, since each GreenPAK includes CNT/DLY blocks.

Deglitch / debounce filters are common when it’s necessary to eliminate glitches - spurious signal transitions. Glitches may occur in many cases: for example, when a button is being pressed/released, or when a voltage level is very close to a threshold (of an input PIN or analog comparator) and there is no hysteresis or Schmitt trigger.

There are 3 possible edge-triggered options to configure a deglitch delay: rising, falling or both. In this case delay block will filter pulses shorter than delay value with corresponding polarity: active High for rising edge, active Low for falling edge, both High and Low for both edge delay. See diagram below:
**Technique: Wake-Sleep Controller**

Waking and sleeping analog macrocells is useful when reducing the power consumption. It is possible to accomplish this with the wake-sleep controller for macrocells like ACMPs and ADCs.

Wake-sleep involves switching the analog macrocells of the chip on and off periodically. For some GreenPAKs this function can be implemented using the WS Ctrl block. For those that don’t have this block, it can be implemented using two counters (one counter if there is no need to change the wake time), a D flip-flop, and an inverter. The figures below display two examples using both methods.

Without wake-sleep implemented, the total current consumption consists of:

1. Quiescent current
2. ACMP current

With wake-sleep implemented, the quiescent current is calculated as follows:

\[
I_{\text{ws}} = I_{\text{without WS}}\cdot \text{System Wake} + I_{\text{without WS}}\cdot \text{System Sleep} / \text{WS Period}
\]

The total current with wake-sleep implemented is:

**Total Current** = Quiescent Current + OSC Current + Wake-Sleep Current (\(I_{\text{ws}}\))
Application:

Interrupt Controller

Ingredients

Any GreenPAK

No other components are needed

Design Steps

1. Configure PINs # 16, 17, and 18 as Analog Input/Output.
2. Configure levels of ACMPs to proper threshold.
3. Wire DFFs to OR gate and set PIN#04 as output.
4. Configure DLY1 to desired debounce time.

The GreenPAK can be configured to monitor multiple different interruptible signals and aggregate that information for the host processor to act upon. The µP or SOC can read via I2C the output of each DFF to determine the source of the fault.
Application:

Ship Mode Controller

Ingredients

Any GreenPAK
External PMOS load switch

Design Steps

1. Configure PIN#02 as an input with 1MΩ pullup.
2. Set desired Button delay time in CNT1/DLY1.
3. Modify LUT contents for correct polarity in and out of ship mode.

An ultra-low power button monitor can save battery life while a product is not yet with the end user. This enables a better first experience by the user.
Technique: Creating a Bidirectional Counter

This technique can be used within any GreenPAK that includes a SPI interface. Alternatives to this technique can be accomplished using other GreenPAKs with an FSM block and storing the counter information using an I²C read command, parallel output, or other method.

A Counter is a basic digital circuit used for counting input events (pulses, edges), often constructed using a cascade of digital flip-flops. In GreenPAK some CNT/DLY blocks are more robust, and can be used as a finite state machine (FSM) that is not only capable of incrementing but can decrement or hold the current value, dictated by interconnects in the GreenPAK matrix. This technique exemplifies this behaviour by using two FSM blocks in GreenPAK to monitor a pulse input (Clock) and output the corresponding 16-bit sequence via the SLG46140V’s SPI macrocell.

Figure 1 - “16-bit FSM with SPI Output”

The “16-bit FSM with SPI output” counts input clock pulses in a constructed 16-bit register (FSM0, FSM1). At any time a user can read the value via SPI, reset the 16-bit register, or change the count direction.

The 16-bit counter is implemented using two counters (FSM0 and FSM1 blocks) with additional logic. Bits [15:8] are stored in FSM0, [7:0] in FSM1. Both FSMs are connected to the SPI block, which can output serial data via SPI. The count direction is controlled by an Up/Down pin, directly connected to the FSM blocks’ UP matrix output. If this pin is HIGH, the system counts UP, if this pin is LOW, the system counts DOWN. Gen Reset pin is used to reset both counter values (active HIGH).

The Clock input pin is applied simultaneously at the CLK input of FSM1 and FSM0. FSM1 counts each clock, whereas FSM0 counts only when FSM1 counter value is 255 and Up/Down signal is HIGH or when FSM1 counter value is 0 and Up/Down signal is LOW. This functionality is achieved using the KEEP input of the FSM0. When this signal is HIGH the counter value of the FSM0 is not changing despite the clock signal. KEEP is connected to FSM1 output through an inverter. In turn, FSM1’s output is only HIGH when counter value is 0 and Up/Down signal is LOW, or when counter value is 255 and Up/Down signal is HIGH.
Application:

Encoder

Ingredients

Any GreenPAK

No other components are needed

Design Steps

1. Configure pins as digital inputs.
2. Set CNT3/DLY3 and CNT5/DLY5 to the One shot mode with the desired filter time.
3. Configure DFFs to detect direction (Up or Down).
4. Set CNT1/DLY1 and CNT2/DLY2 to the One shot mode with the desired output pulse width.
**Technique: Multiplexing a Bitstream**

This technique can be used in any GreenPAK.

GreenPAKs are often used to transfer a data pattern. If the data is transferred from the GreenPAK or the data is transmitted along several lines from the SoC, they must be amalgamated for transmission on one line. Below is an example of the GreenPAK multiplexing a bitstream originating from the ASM output RAM.

*Image of a schematic showing the multiplexer and generator.*

The **generator** circuit is shown above. The generator implements the operation of the CLK line for the synchronous data transmission, enabled by the EN signal. The generator also implements the multiplexer operation algorithm for the correct combination of the transmitted data on one line.

*Image of the 8-bit multiplexer.*

The **8-bit Multiplexer** circuit is shown above. All LUTs are configured collectively as an 8-bit multiplexer (see AN-1003). The **MUX Truth Table** is also shown above. The multiplexer outputs the combination of bits from the ASM block outputs according to the algorithm of the generator and the data is transmitted on one line to DATA. The DATA output will always be the same as the MSB of the ASM output RAM when EN is LOW. The ASM output RAM can be changed via I2C. Logic can also be implemented to change the state of the ASM in order to change the data bits. If the ASM isn’t available, the inputs can be pulled high or low for the data values.

**Technique: Creating a Synchronous State Machine from an ASM**

Synchronous state machines (SSM) do transitions on the edge of an incoming clock if the transition condition is met. The generic approach to convert an ASM into a SSM uses a clock signal with a pulse width greater than the ASM transition time.

*Image of a synchronous state machine diagram.*

Consider the SSM in the 3-bit counter example above. CNT2 and 2-bit LUT1 are used to generate the clock. The ASM uses 8 states connected in series. 2-bit LUT0 and 2-bit LUT3 are used to prevent a logic high signal on two near state transitions. The value of the ASM output for every state is shown below.

The ASM changes from the reset state (State 0) to the next state (State 1) when PIN6 goes high. The following transitions through the states occur as CNT2 toggles first high, then low, and so on.

For more a detailed information about the process of creating an SSM with the ASM see “AN-1126 ASM to Synchronous Conversion.”
N-Length Bitstream

Ingredients
Any GreenPAK with an ASM

Design Steps
1. Configure a generator and 8-bit Multiplexer using Technique: Multiplexing a bitstream.
2. Configure CNT1 to determine the length of the bitstream.
3. Configure the ASM using Technique: Creating a synchronous state machine from an ASM.
4. Connect the outputs of the 8-bit multiplexer and generator to the desired output PINs.
5. The length of the bitstream (counter data of CNT1) can be changed using I^2C.
6. The DATA stored in the ASM output RAM can be changed using I^2C.

A bitstream is a sequence of bits transmitted continuously over a communications path. The GreenPAK is capable of creating a repeating string of up to 64 bits.
Application:

Distance Sense

Ingredients

Any GreenPAK
Eight LEDs
Eight resistors

Ultrasonic ranging modules provide a non-contact measurement function. This design is a controller for an ultrasonic rangefinder based on the HC-SR04.

Design Steps

1. Configure GPIO input for Echo and output for Trig.
2. Add LUT logic and CNT/DLY0 to create a generator with ENABLE signal.
3. Add Pipe Delay and CNT/DLY2 to create a generator for detect distance.
4. Configure CNT/DLY blocks to rising-edge delay mode to measure varied distances.
5. Add and configure DFFs to latch distance data.
6. Connect each DFF output to the desired output pins and configure as open-drain.
Many devices have a specific frequency range in which they operate and require the input clock monitored to stay within this frequency. This application can be used to detect if the input clock frequency is within the desired range.

**Application:**

**Frequency Range Detector**

**Ingredients**

*Any GreenPAK*

**Design Steps**

1. Configure GPIO pins as an input for the clock and output for the flag.
2. Configure CNT/DLY blocks to the Frequency detect mode with a rising edge detect.
3. Set each CNT/DLY block respectively to the minimum and maximum frequency values.
4. Configure a LUT to go high when the frequency is outside the desired frequency range.
Chapter 4 - Safety Features

Technique: Reducing ACMP Power Consumption

This technique can be used in GreenPAKs that include ACMPs. The reduction in power consumption will vary.

GreenPAK CMICs are often used in projects to reduce the system’s current consumption. However, several components within GreenPAKs, when active, can cause a noticeable change in current consumption. Amongst the most consumptive macrocells are the analog comparators. Table 1 is taken from the SLG46826 datasheet to highlight the ACMP’s consumption.

Luckily, ACMPs can be shut down when not in use. This can be done in two ways:

1. Through the PWR UP input of the ACMP.
2. Enabling a Wake-Sleep (WS) counter for the ACMP.

Wake-sleep control requires a dedicated counter configured to WS mode. This is available in many (but not all) GreenPAKs. PWR UP control can be used in any GreenPAK with ACMPs. When the signal is HI the ACMP is on; using logic, counters or other macrocells to turn off the ACMP can drastically reduce power consumption. For example, if two different voltage thresholds are needed the higher-threshold ACMP can be kept inactive until the lower threshold is met.

<table>
<thead>
<tr>
<th>Note</th>
<th>V_{DD} = 3.3 V</th>
<th>V_{DD} = 3.3 V</th>
<th>V_{DD} = 3.3 V</th>
<th>Unit</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chip Quiescent</td>
<td>0.39</td>
<td>0.43</td>
<td>0.53</td>
<td>μA</td>
</tr>
<tr>
<td>Vref OUTI (Source none, Source Temp Sensor, Buffer On)</td>
<td>12.79</td>
<td>12.95</td>
<td>13.57</td>
<td>μA</td>
</tr>
<tr>
<td>Vref OUTI (Source none, Source Temp Sensor, Buffer Off)</td>
<td>7.62</td>
<td>7.67</td>
<td>7.87</td>
<td>μA</td>
</tr>
<tr>
<td>Vref OUTI (Source none, Source Buffer On)</td>
<td>6.53</td>
<td>6.61</td>
<td>7.02</td>
<td>μA</td>
</tr>
<tr>
<td>Vref OUTI (Source none, Buffer Off)</td>
<td>1.40</td>
<td>1.44</td>
<td>1.54</td>
<td>μA</td>
</tr>
<tr>
<td>Vref (ACMPHIL, 0.32 mV, Buffer On)</td>
<td>12.24</td>
<td>12.59</td>
<td>12.21</td>
<td>μA</td>
</tr>
<tr>
<td>Vref (ACMPHL, 0.32 mV, Buffer On)</td>
<td>6.93</td>
<td>7.01</td>
<td>7.43</td>
<td>μA</td>
</tr>
</tbody>
</table>

Table 1 - SLG46826 Current Consumption
**Technique: Zero-Voltage Cross Detection**

This technique can be used in GreenPAKs that include ACMP’s.

Zero-voltage cross detection is commonly used as an accurate method of detecting AC characteristics, such as frequency and phase.

GreenPAKs have a pin voltage range of 0V to a VDD value of 5.5V. To interpret an AC signal that crosses at the 0V point using the GreenPAK, a DC offset shift should be implemented between the AC signal and the GreenPAK pin. This can be accomplished by a 1:1 resistor divider between VDD of the GreenPAK and the AC signal, shown in Figure 16.

Zero-voltage cross detection requires, at minimum, one or two comparators and a counter. The comparators check the incoming AC signal against a reference voltage, which can either come from the GreenPAK’s available reference voltages or an external reference point. If a desired ZVCD voltage is greater than the available GreenPAK reference voltages, the AC signal may instead be reduced by using the IN+ gain option within the comparator’s property settings and comparing the reduced AC value to a similarly-scaled reference (Figure 17).
Application:

Window Comparator

Ingredients

Any GreenPAK w/ ACMP’s

No other components are needed

Design Steps

1. Configure the High-side ACMP2L by using the IN- source and IN+ gain options to set the desired high-side threshold.
2. Repeat step 1 for the low side ACMP with the low-side threshold.
3. Change the IN+ source for the second comparator to ACMP2L IN+ source.
4. Add the LUT logic to trigger an interrupt when the LOW-side comp is low or high-side is high.

Window comparators are an essential part of any design that runs off a depletable power source, like a battery or supercapacitor. By monitoring battery voltage, a device can choose to stop using nonessential resources at low battery levels. This can prevent permanent damage to the device.
Application:

Over Temperature Protection

Ingredients

Any GreenPAK w/ ACMP’s
One resistor
One NTC thermistor

Design Steps

1. Set the IN+ source of ACMP2L to Pin#18 and IN- source to the desired threshold.
2. Connect one node of the resistor to VDD and the second node to Pin#18.
3. Connect one node of the NTC thermistor to Pin#18 and the second node to GND.
Application:

Battery Charge Indicator

Ingredients

Any GreenPAK / ACMP’s

No other components are needed

Battery charge indicators are used in battery-powered devices to indicate the state of charge. This design is optimized for a lithium-ion battery.

Design Steps

1. Connect enable to PWR UP pin of ACMP0H, ACMP1H, ACMP2L, and ACMP3L.
2. Set IN+ source of all the ACMPs to VDD/Pin#20 and each IN- source to the desired threshold level.
Application:

Watchdog Timer

Ingredients

Any GreenPAK w/ ACMP’s

No other components are needed

Watchdog timers are used for automatically generating a system reset signal if the microcontroller or microprocessor neglects to periodically send a pulse. Monitoring for low supply voltage is an additional, common feature of watchdog ICs.

Design Steps

1. Set undervoltage threshold with an ACMP.
2. Configure two CNT/DLY blocks as Frequency detects.
3. Design the digital logic to combine active signals from undervoltage and watchdog timeout.
4. Add a One-shot counter to trigger the reset pulse. It can be inverted to be active low.
Application:

Voltage Level Detection

Ingredients

Any GreenPAK with ACMPs, DCMPs, and an ADC
Up to Eight LEDs and resistors

Design Steps

1. Power on the ADC, DCMPs, and ACMPs.
2. Configure the DCMPs using Technique: Using DCMP PWM block in PWM mode.
3. Set the IN- of each ACMP and DCMP with the desired voltage threshold levels.
4. Add LUT and DFF logic to select and write data of the amplitude of the analog voltage from DCMP1.

Some applications require multiple voltage levels to be evaluated, rather than a few distinct levels. This application shows how to use of ACMPs, DCMPs, and an ADC to monitor the voltage amplitude.
Application:

Power Backup Management

Ingredients

Any GreenPAK with three ACMPs

External resistor dividers to attenuate input signal to the operating value range of ACMPs

Design Steps

1. Use three ACMPs to detect power input signals.
2. Use CNT/DLY blocks configured as a Delay to implement a debounce filter.
3. Add logic cells to create switching priority between input sources.
**Technique: Changing Your Design with I2C**

Many of the techniques and applications available in this section rely upon a GreenPAK’s I2C capability. To learn about I2C within a GreenPAK please consult the chip’s Datasheet.

This technique can be used in any I2C compatible device.

If a GreenPAK device is I2C compatible its behavior can be edited even after it has been programmed. However, a device must be MTP-compatible and undergo In-System Programming (ISP) to retain design changes after it has lost power. This technique outlines a fast way to determine which I2C commands need to be performed to change a design.

1. Complete your initial design. This is the design the IC will use whenever it’s booted up.
2. In GreenPAK Designer, select the I2C Tools Button to open the I2C Reconfigurator.

3. In the I2C Reconfigurator select the snapshot button (in red), or press SHIFT+A. This will take an I2C commandlist “snapshot” of your current design.

4. Change your design to the next configuration.

5. Take a snapshot using the method in step 3 to create the second snapshot.

6. Click the Snapshot Diffs button (in green). This will show the I2C commands necessary to create this design. It is not necessary to program these, since they are instantiated on the boot-up of the GreenPAK.

7. Scroll down the Snapshot diffs list, where you will find the second snapshot. This will show only the values that have changed between the first and second snapshot.

8. These values, shown in the red box on the right, correlate to the hexadecimal address and data value that need to be sent in I2C to change the threshold value of the ACMP (or any other change that may occur).
Technique: Creating an I2C command

This technique can be used in any GreenPAK with I2C.

The I2C generator allows a user to make an I2C signal based on logic generators. It consists of two logic generators acting as SDA and SCL lines. The user can combine predefined I2C primitives to generate the needed waveform and choose an SCL frequency: 1k, 2.5k, and 5 kHz for the GreenPAK Advanced Development Platform and 1k, 2.5k, 5k, 10k, 20k, 50k, 100k, 200k, 400k, 1000 kHz for the GreenPAK Pro Development Platform.

To create an I2C signal using the I2C Generator:

1. Select the Debug button.
2. Select I2C generator on the SDA input of the I2C block.
3. Go to Signal Wizard by clicking EDIT.
4. Select PIN#8 as SCL and set the speed of the clock.
5. Choose Read or Write composite commands.
6. Open composite command and set the Slave address and Word address. For a “Read” command set byte count. For a “Write” command set data to write.

Figure A - I2C Write Command

Figure B - I2C Read Command
Technique: Using Serial to Parallel Interface (SPI) block

This technique is for the Serial to Parallel Interface (SPI) block, available in the SLG46140, SLG46620, and SLG46621.

The SPI block is a special macrocell that can be used for communication between a GreenPAK and a SOC. It can either translate serial data to parallel or parallel data to serial. The inputs are standard SPI I/O connections (MOSI, MISO, nCSB, SCLK, and OUTs). nCSB is an active low chip select. SCLK is the serial clock which clocks the SPI macrocell.

---

 SPI macrocell in serial to parallel mode

The SPI can be used to transfer data to such blocks as:
- FSM
- DCMP
- DAC (through DCMP)

The same SPI can be used to transfer data from:
- ADC
- FSM

All this can be used with other macrocells for functionality like:
- Pulse Width Modulation
- Analog to Digital Comparison
- Digital to Analog Comparison
- Comparing two results with DCMP
- SDIO and LCD

The SPI can be selected to work in either 8-bits or 16-bits. Remember that the SPI macrocell cannot send and receive serial data in the same program file. It must be set up either in the S2P or P2S mode.

In the P2S mode, the ‘INTR’ pin pulses high for one clock period each time data after transmission completes.

Otherwise, the SPI implemented in GreenPAK meets the generally accepted standard. It is possible to set the clocking frequency up to 2 MHz. It is also possible to configure the clock polarity with the CPOL bit and clock phase with the CPHA bit. When CPHA = 0, data can only be transmitted from serial to parallel, not from parallel to serial. When CPHA = 1, data can be transmitted both from serial to parallel and from parallel to serial.
**Technique: Level Shifting**

This technique will work with any GreenPAK that has dual voltage rails, such as the SLG46826V.

Often in a system-level design it’s necessary to combine the data from two signals that operate at different voltage levels. For example, the analog rails in a system might operate at 5.0V while the digital rails operate at 3.3V.

Many GreenPAK ICs solve this problem by using dual voltage rails: signals that operate at different rails can be input into the GreenPAK, manipulated, then output from the GreenPAK at either of the voltage rail levels.

When starting a new GreenPAK Designer design using a dual rail part you’ll be asked to input the voltage range of both rails (Figure 21). The available ranges of both rails will vary from part-to-part. The higher voltage rail should be designated as VDD, not VDD2.

**Figure 21 - Dual Rail Project Info**

In dual rail parts the GPIO connections to the first and second rails are indicated by the color of the IO PIN within GreenPAK Designer (Figure 22). VDD will be indicated by blue pins and VDD2 will be indicated by amber pins. Once inside the GreenPAK matrix the signals from the different voltage levels will behave identically.

**Figure 22 - Dual Rail Logic Example**

---

**Technique: Sending a Preset Number of Pulses**

This technique can be done in any GreenPAK. Multi-function blocks within some GreenPAKs help reduce the component count.

In many communication protocols it’s necessary for a set number of bits to be sent or received by another IC. Typically, this means the GreenPAK must track the number of pulses sent or received. For example, in a shift register receiving data, the number of bits must be monitored to ensure that the expected data is in the correct register, rather than skewed incorrectly or relentlessly continuing to shift.

There are many ways to set a predetermined number of pulses in GreenPAK: a scaleable, efficient way is described in this technique. It also has the added benefit of limiting clock skew between the other IC and the GreenPAK by resetting the clock skew after every transaction. Figure 23 shows a pipeline of blocks for creating a preset number of pulses. It consists of: Turnon Condition, Pulse Period, and a Pulse Counter.

**Figure 23 - Preset Pulse Generator Design**

In dual rail parts the GPIO connections to the first and second rails are indicated by the color of the IO PIN within GreenPAK Designer (Figure 22). VDD will be indicated by blue pins and VDD2 will be indicated by amber pins. Once inside the GreenPAK matrix the signals from the different voltage levels will behave identically.

The Turnon Condition stage has two components: a DLY block to ensure a minimum active signal to begin the pulse generation and a LUT to ensure the turnon condition isn’t re-instantiated before the last transaction has completed. Depending upon the application this stage may not be necessary.

The Pulse Period stage has two elements: a DFF and a counter. The DFF will disable the RESET_IN signal of the counter when the turnon condition is met. It will stay low until the pulse count has finished. The counter is used to set the period of the discrete pulses and can be configured to match the application.

The Pulse Count stage has a falling-edge delay and a logic gate. The delay’s CLK is connected to the counter in the pulse period stage, which causes it to track the number of pulses. The DLY_IN of the delay is connected to the DFF in the Pulse Period stage. This causes the delay to fall low after a set number of pulses, resetting the DFF and halting the shift clock.
**Technique: Building a Shift Register**

This technique can be used within any GreenPAK. The size of the shift register is dependent upon the components available within the specific GreenPAK.

Shift registers are a critical component for serializing or deserializing data. A shift register is a chain of flip-flops that can be sequentially linked and whose outputs can be individually accessed. Each has a connection to a shared clock; on the rising edge of the clock the registers will “shift” their data to the next flip-flop in the sequence. Figure 24 shows a basic, 4-bit shift register. The D flip-flops can be globally reset using a shared reset signal.

![Figure 24 - Basic Shift Register](image)

Often, shift registers must be loaded within the GreenPAK. This can be done by adding a MUX standard logic cell before each DFF. When data is ready to be loaded the MUX select input (“S” in GreenPAK Designer) is toggled and the DFF clock input is triggered to commit the value for each register. Figure 25 shows the addition of a MUX on 2 of the bits in Figure 24’s basic shift register. **LOAD_EN** shares the clk input to commit the loaded values to **DFF3** and **DFF4**.

![Figure 25 - Loading a Shift Register](image)
Application:

**I²C IO Expansion**

**Ingredients**

Any GreenPAK with I²C

*No other components are needed*

**Design Steps**

1. Configure GPIO pins as output.
2. Connect to I²C virtual inputs.
3. I²C virtual inputs can be changed individually or simultaneously using the I²C virtual output address, found in the GreenPAK’s datasheet.
Application:

Serial to Parallel (External Clock)

Ingredients

Any GreenPAK
An IC an external clock output

Design Steps

1. Configure the shift register as shown in Technique: Building a Shift Register.
2. Add an input connection for the internal clock and connect it to the CK input of the shift registers.
3. Add an input connection for the reset function and connect it to the nRESET of the shift registers.

Deserialization ICs are used so that data can be sent across one wire and interpreted as multiple bits of data. They often use an external clock tied to the same device as the data line to avoid clocking the data at an incorrect time.
Application:

Serial to Parallel (Internal Clock)

Ingredients

Any GreenPAK

No other components are needed.

Design Steps

1. Configure the shift register as shown in Technique: Building a Shift Register.
2. Add an input connection for the reset function and connect it to the nRESET of the shift registers.
3. Configure a preset number of pulses to match the number of shift registers. This is outlined in Technique: Sending a Preset Number of Pulses.
Application:
Parallel to Serial

Ingredients
Any GreenPAK
No other components are needed

Parallel to Serial convertors are commonly used to send data from one IC to another using one or two wires (Data and Clock). Internal or external clock can be used here and number of parallel bits is limited by number of GPAK I/Os and internal blocks available.

Design Steps
1. Configure the shift register as shown in Technique: Building a Shift Register.
2. Add and configure 3-bit LUTs for each DFF to operate as MUX using Technique: Configuring Standard Logic w/ LUT Macrocells.
3. Configure Load/Shift function and connect internal blocks to inputs, output.
Application:

**Bi-Directional Communication (Transmit First)**

**Ingredients**

Any GreenPAK with OE pins

No other components are needed

**Design Steps**

1. Configure two GPIO pins as Input/Output, one for the data line and one for the clock.
2. Create an internal clock signal using a CNT/DLY block configured as a reset counter.
3. Configure a shift register to store incoming data, shown in Technique: Building a Shift Register. Shift registers can be read with PC.
4. Add PGEN to send outgoing data. The PGEN contents can be changed using PC.
5. Configure DFF to enable transmission. Connect output to transmit clock signal and OE of I/O pins.
6. Add CNT/DLY block to disable transmission, with clk input from DFF. Connect Edge Det block to reset DFF after set number of clock pulses.
7. Connect the external clock to the shift register and connect the internal clock to PGEN and I/O pins.
Application:

**Bi-Directional Communication (Receive First)**

**Ingredients**

Any GreenPAK with OE pins

*No other components are needed*

---

**Design Steps**

1. Configure two GPIO pins as Input/Output, one for the data line and one for the clock.
2. Create an internal clock signal using a CNT/DLY block configured as a reset counter.
3. Configure a shift register to store incoming data, shown in Technique: Building a Shift Register. Shift registers can be read with I2C.
4. Add PGEN to send outgoing data. The PGEN contents can be changed using I2C.
5. Configure DFF to enable transmission. Connect output to transmit clock signal and OE of I/O pins.
6. Add CNT/DLY block as a reset counter to disable transmission, with clk input from DFF. Connect Edge Det block to reset DFF after set number of clock pulses.
7. Add CNT/DLY block as a reset counter with input as external clock and output to enable transmission DFF.
8. Connect the external clock to the shift register and connect the internal clock to PGEN and I/O pin.
Application:

7-Segment Display Using ASM and I²C

Ingredients

Any GreenPAK with I²C and ASM
7-segment display
Eight resistor

Design Steps

1. Configure GPIO pins as output and connect them to the ASM output.
2. Add shift register using the Technique: Building a Shift Register.
3. Create a logic generator using the PGEN block, corresponding the required number of digits.
4. Add and configure Asynchronous State Machine (ASM) to match the initial digital sequence.
5. Update one or more digits to an ASM state via I²C.

A 7-segment indicator is a common numerical display. The GreenPAK asynchronous state machine and I²C can be used to provide directions to the segments as to which number should be displayed. The provided example is compatible with a 4-digit, 4 decimal display.
Application:

**Communication MUX using I²C**

**Ingredients**

Any GreenPAK with I²C
Two resistors
Two capacitors

An I²C Communication MUX is used when a designer needs to combine several I²C input signals and forward them into a single output line.

**GreenPAK Diagram**

**Design Steps**

1. Configure a 1:3 demux with LUTs to share the TX signal from the MCU, sent to external UART ports.
2. Configure a 3:1 mux with LUTs to receive an RX signal from the external UART ports to the MCU.
3. Use the I²C to select the input port.
**Application:**

**I²C Level Shifter**

**Ingredients**

- Any GreenPAK
- Four resistors

**I²C Level Shifters** allow two I²C-enabled devices to communicate to each other across two different voltage levels. The given example level shifts from 3.3V to 1.8V.

**GreenPAK Diagram**

**Design Steps**

1. Configure four GPIO pins as digital input/outputs with the output mode set to open drain NMOS.
2. Configure one pin as a digital input for the enable signal.
3. Add an AND gate to each input/output.
4. Configure four multi-function blocks (or four LUTs and four CNT/DLY blocks if Multi-function blocks aren’t available) as a NOR gate feeding into a falling edge delay.
5. Select OSC2 as the clock source for the delay blocks and set it to “Force Power On”.
Pulse-based Control

Chapter 6

Technique: Setting a Constant Duty Cycle

This technique will work with any GreenPAK.

Setting an immutable duty cycle requires one CNT/DLY block, an oscillator, and a DFF. The macrocells should be configured as shown in Figure 27.

The oscillator determines the period, the DFF is a rising edge detector, and the CNT/DLY block determines the duty cycle. When the rising edge from the oscillator is registered by the DFF it will send a LO pulse to the CNT/DLY block. This will set the CNT/DLY output LO, and the output will only rise after the Delay Counter data has been met.

From the DFF’s initial configuration change the Q output polarity to Inverted (nQ) and connect the output of the DFF. This will allow it to operate as a rising edge detector; it will remain HI until a rising edge is detected on the clock, whereupon it will briefly drop LO. The FILTER/EDGE DET block can also be configured for this purpose.

The oscillator OUT0 or OUT1 is connected to the DFF’s clock input to generate the period. The period should always be greater than the duty cycle. In this example the period, set by ‘OUT1’ second divider by OSC/64, is set to OSC/64.

The CNT/DLY block’s Counter data option sets the duty cycle. The delay time sets the duration of the low signal. The duty cycle is calculated as:

$$D = \frac{T_{period} - T_{delay}}{T_{period}}$$

Figure 26 - CNT/DLY Configuration

Figure 27 - Simple Duty Cycle Configuration

Figure 28 - Simulation of Duty Cycle = 50%
Application: H-Bridge Control

Ingredients
Any GreenPAK
Four transistors
Four diodes

An H-Bridge is an electronic circuit that enables a voltage to be applied across a load in opposite direction. These circuits are often used in robotics and other applications to allow DC motors to run forwards or backwards.

Design Steps
1. Add and configure inputs and outputs.
3. Add and configure LUTs for each output using Technique: Configuring Standard Logic w/ LUT Macrocells.
Application:

**Constant Current LED Driver**

**Ingredients**
- Any GreenPAK
- Four capacitors
- Four diodes (2 LEDs, 2 silicon diodes)
- Two resistors

LED drivers provide and regulate the current to LEDs. They keep the power at safe levels to prevent burnout/thermal runaway.

**Design Steps**
1. Configure two ACMPs, each with their IN-source set to the desired threshold.
2. Configure LUTs to enable the LED outputs.
3. Connect LED1_1 and LED1_2 to the anode of a silicon diode and connect the cathode of the silicon diode to the anode of an LED.
4. Connect a resistor between the cathode of the LED and ground.
5. Connect a capacitor between the anode of the LED and ground.
6. Repeat steps 3-5 for the LED2 outputs.
7. Connect V_SENSE1 and V_SENSE2 to the cathode of LED1 and LED2 respectively.
Application:

LED Control via I²C

Ingredients

Any GreenPAK with I²C
RGB LED
Three resistors

A system may need a single LED that can change colors in a very wide spectrum. To accomplish this an RGB led control is used to change the duty cycle for different colors. I²C is used in this application as an easy way to change duty cycle.

Design Steps

1. Configure GPIO pins as open-drain outputs for RGB cathode connection.
2. Add LUT logic and CNT/DLY2 to create a generator with EN signal.
3. Configure a CNT/DLY block to rising-edge delay.
4. Add and configure LUTs for each output using Technique: Configuring Standard Logic w/ LUT Macrocells.
5. Connect each LUT output to the desired output pins.
6. I²C virtual inputs can be changed individually or simultaneously using the I²C virtual output address.
7. Counter data of CNT/DLY blocks can be changed individually or simultaneously using the I²C.
**Application:**

**Charge Pump**

**Ingredients**
- Any GreenPAK
- Two capacitors
- Two diodes

**Design Steps**
1. Set the divider in OSC1 to obtain the desired output frequency.
2. Configure logic to provide a shutdown function, either through IO or I2C (if available).
3. Connect a diode (D1) and $C_{pump}$ between VDD and CP_DRIVE.
4. Connect the anode of D2 to the cathode of D1.
5. Connect $C_{out}$ and $R_{load}$ in parallel between the cathode of D2 and ground.

A charge pump is a DC-DC converter that uses capacitors for energetic charge storage to create different voltage levels. It can be used to provide additional voltage levels for powering specific interface circuits, sensors etc.
Application:

**Two-stage Charge Pump**

**Ingredients**

- Any GreenPAK
- Three capacitors
- Three diodes

**Design Steps**

1. Create a basic charge pump design using the steps in Application: Charge Pump.
2. Configure logic to add an inverter. Connect the inverter between an output pin and the oscillator.
3. Connect an additional diode and capacitor between the first stage and the output stage of the charge pump. The new capacitor and diode should be the same type and value as the first stage’s components.

A multi-stage charge pump can push the output voltage to beyond double the input voltage. Schottky diodes are recommended for best performance.
**Technique: Using DCMP/PWM Macrocell in PWM Mode**

This technique is for the DCMP block, available in the SLG46140, SLG46620, and SLG46621.

**Overview of the DCMP/PWM Macrocell**

The DCMP/PWM block is a special macrocell used to compare two 8-bit values or generated PWM signals. There are three DCMP/PWM blocks per IC that can operate independently, and each DCMP/PWM has two 8-bit inputs (IN+, IN-) that can be used to generate a PWM signal. Inputs MTRX SEL#0 and MTRX SEL#1 are used during static PWM generation to select one of the four available registers, shared amongst the three blocks. Input SHARED PD is used to POWER ON/OFF. The PWM output duty cycle range can be configured to range from 0% to 99.61% or 0.39% to 100%.

![Diagram of DCMP/PWM Macrocell](image)

**Creating the PWM Signal**

The PWM generator is similar to a digital comparator, with the exception that the data on one of the inputs is linearly cycling through a set PWM period. The other input is static at a set duty cycle or updating at a very slow rate.

So, to create a PWM signal, we need a PWM ramp source. This is an internal or external counter that counts from 255 down to 0 or vice versa. Another source of data for the DCMP/PWM block should be stable for at least 1 PWM signal period (PWM ramp counter period). It could be data from the SPI, ADC, FSM blocks, or from an internal register of DCMP/PWM. Figure 1 shows the operation of a DCMP/PWM when IN- is connected to a CNT/DLY block that counts from 255 down to 0 (PWM ramp counter) and the IN+ source is an internal register set to 250.

The IN+ input configurability for the DCMP/PWM block is the key to the large configurability in the macrocell. Set PWM values can be made using the internal registers, PWM dynamic feedback can be made using the ADC, and an MCU-controlled PWM can be set by using the SPI interface.

The output OUT- and OUT+ has dead band time that can be from 10 to 80 ns and set in the properties pane of DCMP/PWM.

![Diagram of PWM Signal](image)
Application:

PWM Selection

Ingredients

GreenPAK with DCMP

No other components are needed

Design Steps

1. Enable the DCMP by removing VDD from SHARED PD input. Set DCMP/PWM power register to Power on. Set IN+ selector to Register selected through from matrix. Set IN- selector to FSM1. Set the registers of DCMP0: register 0 - 51; register 1 - 102; register 2 - 154; register 3 - 0.

2. Add CNT/DLY block configured as Counter/FSM. Set counter data to 255.

3. Configure RC OSC power mode to Force power on.

4. Configure PIN10 and PIN9 to open drain NMOS.

5. Add LUT as an inverter.

6. Connect input pins to the MTRX SEL options of a DCMP/PWM block.
Application:

PWM Generator Using DAC and ACMP

Ingredients

GreenPAK with DAC

No other components are needed

PWM generators can be used to control devices such as dc motors and LEDs. This implementation uses an analog signal that an ACMP compares with the signal of DAC0. CNT3 is used to generate the value for the DAC.

Design Steps

1. Connect ACMPO PWR UP input to POR. Set IN- source Ext. Vref (DAC0 out).
2. Set DAC0 power on signal to Power on and input selection From DCMP1’s input.
3. Configure FSM-compatible CNT/ DLY to Counter/FSM mode with counter data = 255.
4. Set RC OSC power mode to Force power on.
Application:

**PWM Generator Using ADC**

**Ingredients**

GreenPAK with ADC

No other components are needed

PWM generators can be used to control devices such as dc motors and LEDs. This implementation uses an analog signal connected to an ADC to compare with the value of CNT2 in PWM0. If the CNT2 value is less than the digitized analog signal the output of PWM0 is high. After CNT2 value is 0 the output of PWM0 is low.

**Design Steps**

1. Remove VDD from the ADC’s PWR DOWN input and set PGA power on signal to Power on.
2. Set PIN 6 to Analog input/output.
3. Configure DCMP0/PWM0 by deleting VDD from SHARED PD input. DCMP/PWM power register set Power on. Check that IN+ selector connect to ADC and IN- selector connect to FSM0.
4. Configure 4-bit LUT1/14-bit CNT2/DLY2/FSM0 as Counter/FSM with counter data equal to 255.
5. Connect DCMP0/PWM0 OUT+ to output pin.
Technique: Creating a Breathing LED Pattern

This technique can be used within any GreenPAK. The quantity of independent Soft ON/OFF channels depends on the number of counters available within the particular part.

An LED breathing pattern can be generated through a basic PWM implementation that utilizes a constant change between two counters. Each counter outputs a high pulse for one clock cycle of their programmed period. Two CNT/DLY blocks are programmed with different counter data settings to provide a small offset between their outputs. These output signals are used to set and reset a flip-flop within the device. Figure 1 depicts a basic implementation, wherein CNT2/DLY2 sets the ON period and CNT3/DLY3 sets the duty cycle.

In the implementation within Figure 1 the frequency of the PWM is set by CNT2 and can be calculated with the equation:

\[ f_{PWM} = \frac{f_{OSC}}{(\text{Data}_{\text{CNT2}} + 1)} \]

The effect of the small offset is shown through the waveforms of Figure 2. The PWM cycle ends when the counters’ outputs coincide. This causes a short high impulse on AND gate and DFF flops. The NXOR gate makes the inversion of the PWM, which provides a soft OFF. PIN2 is the enable signal and while it is HIGH the counters are in high level reset.
Application:

Breathing RGB LED

Ingredients

Any GreenPAK
One RGB LED
Three resistors

Design Steps

1. Configure GPIO pins as open drain NMOS outputs.
2. Create soft ON/OFF circuit as shown in Technique: LED breathing.
3. Configure Ripple Counter - set Functionality mode to Range: SV-EV cycles (SV=1, EV=3).
4. Configure LUTs collectively as a demultiplexer.
5. Add enable (EN) signal to start/stop RGB breathing.

RGB LEDs can be used to add more complexity to LED indication systems and be controlled with a GreenPAK. They can also be paired with a soft ON/OFF circuit for a breathing pattern aesthetic, and/or with an I2C-compatible part for additional functionality.
**Technique: Duty cycle detection**

This technique can be used within any GreenPAK. Since the input frequency range is limited by the maximum FSM counter data, it is better to use a 16-bit FSM. The PWM detection input frequency should be much slower than the duty cycle reference frequency to improve the accuracy.

Duty cycle detection is important for applications such as overload protection, DC/DC conversion, servo motor control, and protocol detection. This design can be easily implemented using a GreenPAK with an FSM block.

In the implementation shown above, when PIN#4 goes HIGH, FSM0 starts counting DOWN, clocked by the internal oscillator. FSM0 is set to 65535 by a rising edge on PIN#4. When PIN#4 goes LOW, FSM0 starts counting UP with the frequency from internal oscillator divided by CNT1’s data value (Figure B). If FSM0 reaches 65535, DFF3 will be set LOW by the next edge rising edge from PIN#04, which flags that the duty cycle is below the set threshold.

The duty cycle reference frequency can be adjusted by changing the CNT1 counter data value via I2C. To calculate the duty cycle threshold, the following formula is used:

$$Duty\ Cycle\ Threshold = \frac{100}{CNT1\ Counter\ data^2} \ (%)$$
Application:

RGB LED control via I²C

Ingredients

Any GreenPAK with I²C
One RGB LED
Three resistors

Design Steps

1. Create a soft ON circuit as shown in Technique: Creating an LED Breathing Pattern but instead set the counter data for CNT1 and CNT2 to the same value.
2. Add MF0 which adds a small offset between CNT1 and CNT2.
3. Configure LUT3-LUT5 as a multiplexer switched by the EN signal controlled via I²C.
4. Configure LUT0-LUT2 as a demultiplexer which passes the breathing signal according to the timing diagram in Figure A.
Associated GreenPAK Designer Files

All GreenPAK Designer files shown within the GreenPAK Cookbook Applications can be downloaded from the Dialog Semiconductor website:

https://www.dialog-semiconductor.com/greenpak-cookbook

Disclaimer

Information in this document is believed to be accurate and reliable. However, the author of this document and Dialog Semiconductor do not give any representations or warranties, expressed or implied, as to the accuracy or completeness of such information. The author of this document and Dialog Semiconductor furthermore take 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.

Applications, software, and semiconductor products described in this document are for illustrative purposes only. The author of this document and Dialog Semiconductor make 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.

Customer notes that 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 are subject to Dialog Semiconductor’s Standard Terms and Conditions of Sale, available on the company website (www.dialog-semiconductor.com) unless otherwise stated.

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

© 2020 Dialog Semiconductor. All rights reserved.
This publication is issued to provide outline information only, which unless agreed by Dialog Semiconductor may not be used, applied, or reproduced for any purpose or be regarded as a representation relating to products. All use of Dialog Semiconductor products, software and applications referred to in this document are subject to Dialog Semiconductor’s Standard Terms and Conditions of Sale, available on the company website (www.dialog-semiconductor.com) unless otherwise stated.

Dialog and the Dialog logo are trademarks of Dialog Semiconductor plc or its subsidiaries. All other product or service names are the property of their respective owners. © 2020 Dialog Semiconductor. All rights reserved.