Final Project Report

Georgia Shay

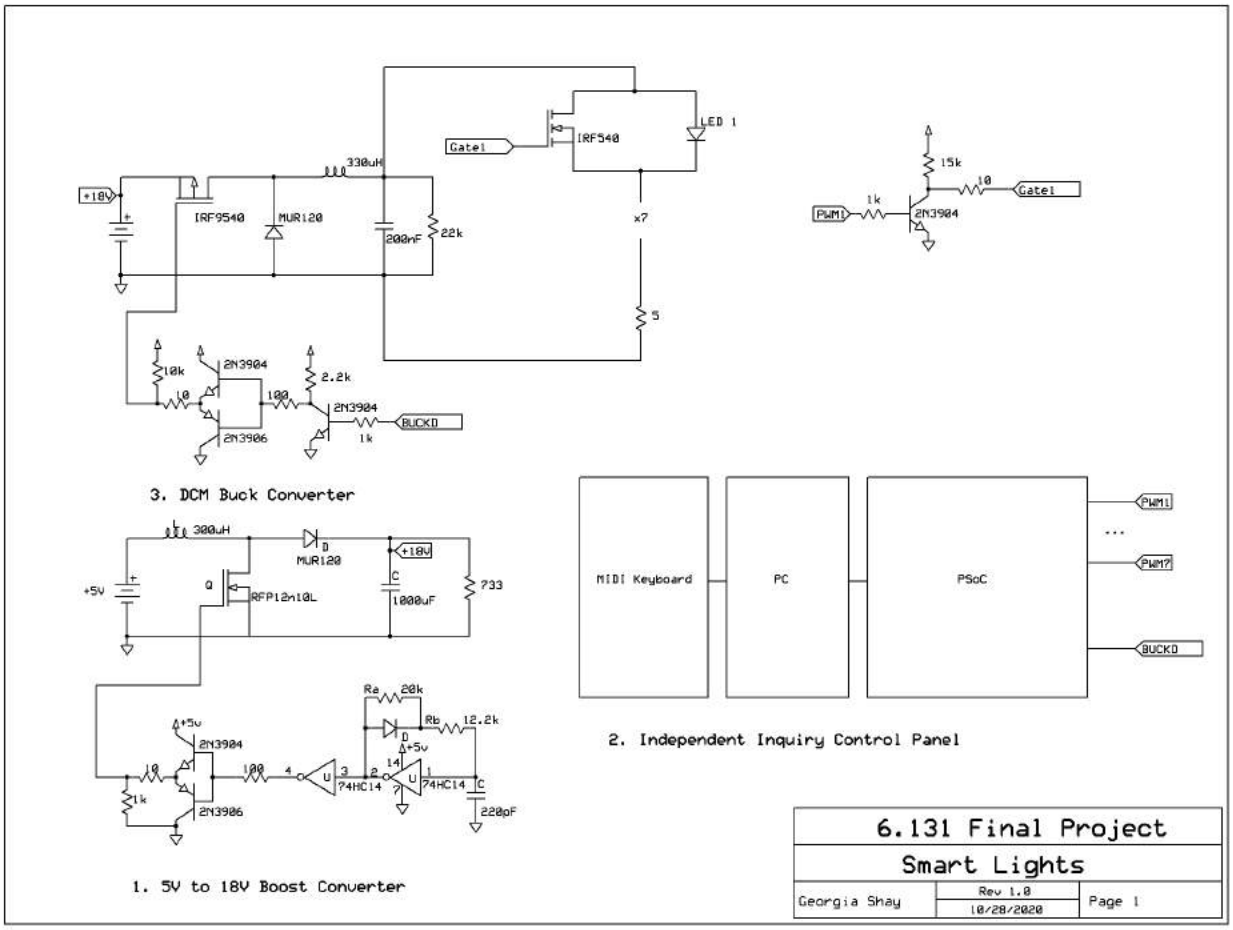
December 9 2020

1 Introduction

This final project is a smart light system designed to facilitate ease of use through automation as well as to conserve energy. The 6.131 portion of the project focuses on energy conservation by exploring DCM buck converters, while the independent inquiry concentrates on programming of light systems. The system is designed to independently control 7 LEDs without using wasteful current-limiting resistors for each one. The independent inquiry portion is a PSoC-based control panel for the 7 lights which is programmed by a MIDI keyboard. This idea was motivated by an interest in MIDI protocol, which is decoded in order to control the lights.

The main reference on lighting control systems for this project is "Lighting Control: Technology and Applications" by Robert Simpson. This reference has greatly informed the interface design for the control panel. For example, many industry control panels contain sliders to control different light levels, as it is very important for the user to be able to visually see the relative brightnesses that are currently programmed (p. 354). In this design, the LEDs on the PSoC provide visual feedback on the different light levels. Industry control panels, from theatrical to residential building applications, use "scene presets", or sets of brightness levels for each light that the user can smoothly fade between (p. 355-356). This interface design uses the same standard, with the ability to store up to 8 preset scenes. A switch bank on the PSoC will allow the user to choose which scenes to make active. In addition to its information on the control systems for lighting, this reference material also helped inform the control protocol for the panel, which will be motivated by simplicity. The simplest lighting systems use analog voltages or PWM signals to directly communicate brightness levels to individual lightbulbs (p. 280, 283). The LEDS in this design use PWM outputs from the control panel for communication. While larger systems often need to use common buses with more advanced communication protocols (p. 303), PWM will be sufficient for this small scale system. In addition, MIDI protocol will be used to communicate with the control panel. While industry usages often have to contend with controlling different "channels" or "groups" of lights (p. 356, 288-303), in this simplified system one key on the keyboard can be directly associated with one light. Finally, this textbook contained a wealth of information on improving power efficiency in lighting systems. This in turn influenced the standard 6.131 portion of the project. One way to improve efficiency is the scheduling of the lights to turn off when no one is there, something possible with an extension of the panel's protocol. Another way is to ensure the lights themselves operate efficiently, which motivated the decision to operate the LEDs with a Discontinuous Conduction Mode (DCM) converter (p. 402).

2 Hardware Schematic



The circuit design includes a boost converter, a DCM buck converter, and a control panel (a PSoC) that can be interacted with through a MIDI keyboard.

The boost converter is controlled by a Schmitt trigger clock that creates a fixed duty cycle to boost a 5 V input to an 18 V output.

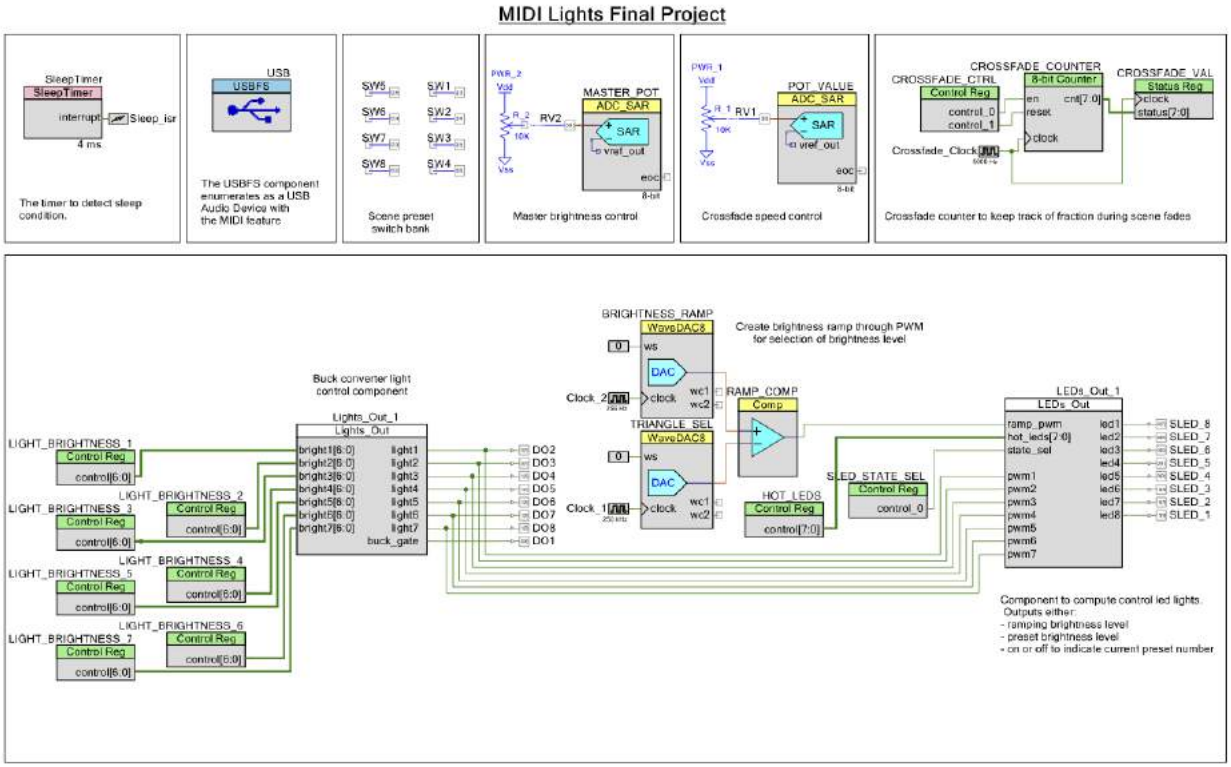
This 18 V output is used to power 7 LEDs as well as control circuitry. The 7 LEDs compose the series load of a buck converter operating in discontinuous conduction mode. The LEDs can be taken in and out of series with the load through a MOSFET connected drain to source across the LED. If the MOSFET is turned on, then there will be no voltage drop across the LED, and thus no current flowing through it. These MOSFETs can be operated quickly, at a PWM rate of around 100 Hz to avoid too much flicker, to acheive different brightnesses of the LEDs. The buck converter should be operated at different duty cycles depending on the number of LEDs currently in the load (not shorted by MOSFETs) in order to maintain a constant current of 20 mA through the LEDs. A current of 20 mA is the maximum average current through the LEDs and would allow them to be as bright as possible.

The buck converter's duty cycle will be controlled through a square wave output of the PSoC. As this waveform will have a maximum of 5 V, rather than the 18 V required to control the PFET of the buck converter, an open drain output is created using an NPN transistor and its output is pulled up to the appropriate 18 V. This (inverted) 18 V square wave is fed into a push-pull BJT driver that ultimate controls the PFET gate. Similarly, the PWM waveforms for each LED originate from the PSoC. They also need to be at 18 V, so an NPN transistor and pullup resistor

is used for each.

The control panel handles the programming of the lights. A MIDI keyboard can be plugged into a computer, and the computer plugged into the PSoC control panel. The MIDI output from the keyboard is passed through to the PSoC. Depending on key presses from the MIDI keyboard, the user can enter the programming flow to set "scenes" of brightness levels for each LED.

3 PSoC Software

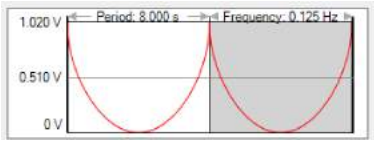


Above is the top-level design for the PSoC control panel. The PSoC is responsible for interpreting incoming MIDI messages to advance scenes of lights or to program their brightnesses, and output signals that control the status LEDs on the PSoC as well as the 7 lights in the buck converter.

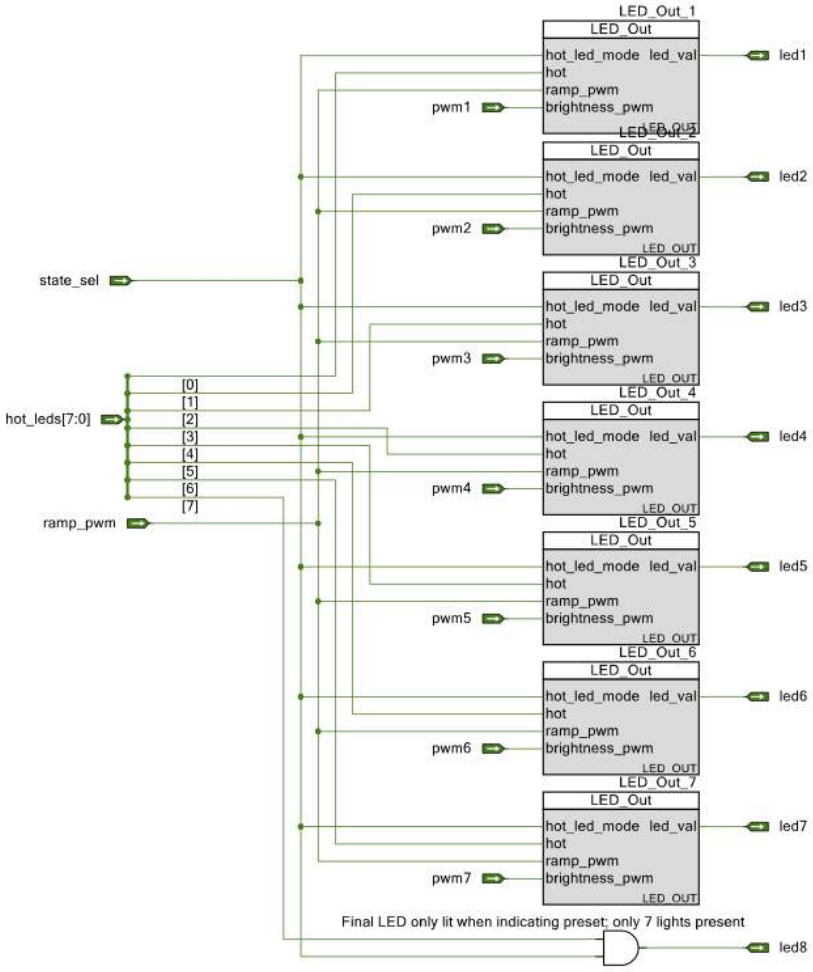
The SleepTimer and USBFS components serve to deal with the MIDI protocol. A switch bank provides a way to disable scenes. This disabling of scenes is handled in software. One of the potentiometers on the PSoC shield acts as the master brightness control, scaling every brightness in the scene at once. When the current scene is advanced, the next scene fades in slowly. The speed of this fade is controlled by another potentiometer on the PSoC shield. The potentiometer value is fed into a ADC, whose output is used to adjust the clock speed of the crossfade counter (top right of schematic). When the scene is advanced, the crossfade counter is started and controls the amount of each preset scene to mix into the brightness levels of each LED, until it reaches a final value of 255 and stops.

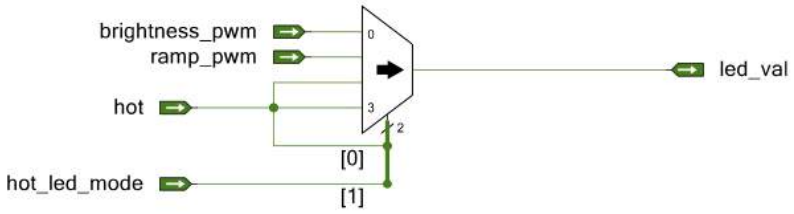
The major operation of the PSoC is shown in the large central area of the schematic. First (lower left), software-controlled brightnesses are fed into a user-defined component that controls both the duty cycle of the buck converter and the 7 gate waveforms for the LED-shorting MOSFETs. Next, a separate component determines the digital output to send to the status LEDs on the PSoC itself. As inputs, it takes the digital outputs of the previous stage (i.e. the PWM waveforms for the buck

lights' brightnesses), state values from software, as well as a PWM waveform corresponding to an LED brightness which ramps up and down. This "brightness ramp" is created by using an analog comparator on a high-frequency triangle wave and a user-defined waveform shown below.



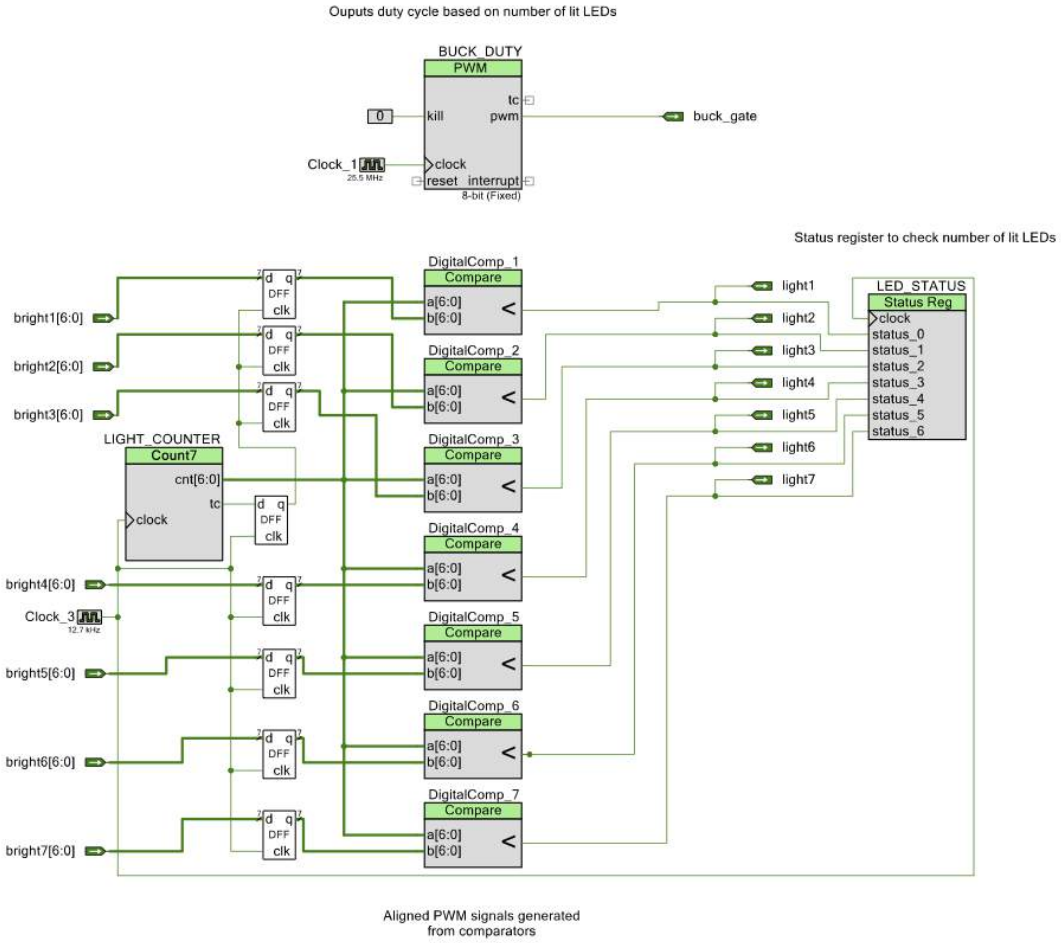
This waveform is created to give more emphasis to lower brightnesses, as light is perceived logarithmically. Below is the custom status LED component, which determines which of the three potential outputs to send to the status LEDs. This work is mostly delegated to the LED_Out component.





The LED_Out component uses a multiplexer to select one of the three potential values for the status LED. The "hot_led_mode" refers a state set by software which is 1 if a single LED should be lit to indicate which scene is being controlled. It should be 0 if the LEDs should be displaying PWM brightnesses. The "hot" wire is 1 if this given LED is "active" and 0 if it is "inactive" (only 1 LED can be "active" at a time). Since these are combined for the select signal, the led will follow its "hot" value if it is in a hot led mode - i.e. turn on only if it is the active scene LED. If it is not in hot led mode, it will either select the brightness ramp (if the LED is active/"hot"), or the scene brightness level for that LED (if the LED is not active/"hot").

Finally, the buck lights are controlled by a different component, Lights_Out.



The lights are controlled with a counter and digital comparator. The counter will complete its periods at a rate of 100 Hz. As the counter counts down, when it reaches a value below that of the

digital input brightness value (bright1, bright2, ...), that light's output will go high. The higher the brightness value, the longer the output is high. All of the lights will end their cycle at the same time. This is crucial, because it means that the number of light outputs that are high will always increase until the end of the cycle. This helps keep current spikes to a minimum as the output voltage readjusts.

The flip-flops in this design ensure that the brightness value does not change until the end of a period. The led status register allows the software to keep track of the number of lights that are lit at any given time. This value can be used to control the duty cycle of the buck converter. The buck converter's duty cycle is controlled by a PWM component and updated in software as shown below.

```
// Duty cycles for each number of LEDs 0 - 7
uint8 `$INSTANCE_NAME`_dutyCycles[8] = { 0u, 8u, 20u, 30u, 46u, 77u, 91u, 101u };

uint8 `$INSTANCE_NAME`_lastNum = 0u;

void `$INSTANCE_NAME`_Start() {
    `$INSTANCE_NAME`_BUCK_DUTY_Start();
    `$INSTANCE_NAME`_LIGHT_COUNTER_Start();
}

void `$INSTANCE_NAME`_Poll() {
    // Count the number of LEDs currently on
    int numLeds = 0u;
    uint8_t hotLeds = `$INSTANCE_NAME`_LED_STATUS_Read();
    while (hotLeds) {
        numLeds += hotLeds & 1u;
        hotLeds = hotLeds >> 1;
    }
    // Set PWM to appropriate duty cycle
    if (`$INSTANCE_NAME`_lastNum != numLeds) {
        `$INSTANCE_NAME`_BUCK_DUTY_WriteCompare(`$INSTANCE_NAME`_dutyCycles[numLeds]);
        `$INSTANCE_NAME`_lastNum = numLeds;
    }
}
```

The duty cycles (out of 255) were calibrated by hand, adjusted from the pre-calculated values. The code simply selects the appropriate duty cycle based on the number of LEDs currently lit, so that an average 20 mA current is maintained.

A crucial part of the software is the interpretation of MIDI messages. The USBFS component provides the MIDI messages directly to software. For the control panel, the only needed information is when specific keys are pressed and when they are unpressed.

The programming keys (those on the pads of the MIDI keyboard, rather than the regular note keys), should change the mode or update the scene number when they are pressed (and do nothing when unpressed). Seven of the regular note keys are used to start a brightness ramp for a certain LED and then stop that brightness ramp when the desired brightness level is reached. For these

note keys, it is critical to know both when the key is pressed and when it is unpressed, to start and then release the brightness ramp. MIDI messages consist normally of three bytes - the first byte containing the type of message, the second byte containing the key number, and the third byte containing the key velocity - or, how hard the key is pressed. To determine when a key is pressed or unpressed, at each MIDI event the message is checked for whether its first byte is a "note on" or "note off" message and what key number is in the second byte. However, most MIDI controllers send a "note on" message with 0 velocity (as this can be sent more efficiently). So, the code checks for both a note off message or a note on message with a velocity 0 to determine whether a key has been unpressed.

```
if (mode == PRESET_MODE && isLightKey) {
    if (midiMsg[MIDI_MSG_TYPE] == USB_MIDI_NOTE_OFF ||
        (midiMsg[MIDI_MSG_TYPE] == USB_MIDI_NOTE_ON && midiMsg[MIDI_NOTE_VELOCITY] == 0u)) {
        // Set control LED to active
        hotLeds &= ~(oneHotKey);

        // Update stored brightnesses based on stopping point in brightness ramp
        storedBrightnesses[preset][keyNumber] = BRIGHTNESS_RAMP_VDAC8_Data >> 1;
    } else if (midiMsg[MIDI_MSG_TYPE] == USB_MIDI_NOTE_ON) {
        // Set control LED to off
        currentKeyNumber = keyNumber;
        hotLeds |= oneHotKey;
    }
}
```

Finally, the code must keep track of the current state as well as the brightnesses of each LED for each scene. Normal operation is playback mode, where pressing the "play/pause" key advances the lights to the next scene preset. The user can press program, and then the next and previous buttons, to cycle through the presets and select one to program. The user can then press the preset button to begin programming a specific scene, at which point the note keys can select the brightness levels. Pressing the buttons in reverse - preset, then program - returns the user to playback mode.

The 56 brightnesses (7 brightnesses per scene, for 8 scenes) are stored in an array, and both the current mode (playback, program, or preset) and current preset number (0 through 7) are stored as well. Then, in each iteration of the main loop, the code assigns the correct brightnesses to the hardware registers, as well as other state values. In playback mode, the status LEDs corresponding to the current preset scene number should be on and the buck lights should have the brightnesses from that preset. So, the code in that case will set the correct LED to "hot" and set the state to "hot led mode" for the status LEDs. It will also index into the brightness array and set the correct brightnesses for the buck light registers. In program mode, the buck lights do not change but the status LED displays the selected preset, so the code again sets the register to hot led mode with the correct status LED selected. Finally, in preset mode, both the status LEDs and buck lights display brightnesses for the selected preset, so hot led mode is turned off and the correct brightnesses are sent to the registers. Since the preset mode is where a user selects the brightnesses, a hot led should also be set if a MIDI key is pressed, so that that LED can receive the brightness ramp. These values are set on every iteration of the main loop, so that any changes in the state machine from key presses or crossfading take effect as immediately as possible.

4 Boost Converter

4.1 Specifications

The input voltage for the boost converter is 5 V. It is designed to produce an output of 18 V for usage in the next stage to power the 7 LEDs. This should ensure sufficient voltage is available despite losses in the circuit.

The switch frequency should be above audio range. The boost converter will use a reasonable switch frequency of 250 kHz.

The boost converter will operate with close to no load when all the LEDs are off, thus requiring a robust safety resistor as a minimum load. However, a large power dissipation in this safety resistor load is not desired. The boost converter should stay in CCM down to power dissipations comparable to that which is dissipated the LEDs, so that the safety resistor must not necessarily represent a large fraction of power delivered by the boost output. A single LED at full brightness dissipates $20\,\text{mA} \cdot 2\,\text{V} = 0.04\,\text{W}$, for a 0.28 W dissipation in all 7 LEDs. Here the assumption is made that only the safety resistor would be dissipating power under the no load condition for the LEDs; this is not true but is a reasonable starting point.

A voltage swing which drops below the 14 V required across a full load of LEDs would be unacceptable. This would be a voltage ripple of 8 V.

An additional concern would be the current ripple in the LEDs which is caused by a voltage ripple in the boost converter. As a first approximation, this current is proportional to the boost converter voltage squared, due to the equation for current from the DCM buck converter:

$$\langle i_O \rangle = \frac{\frac{V_{in}}{V_O}(V_{in} - V_O)D^2T}{2L}$$

While estimates for the peak current the LED can withstand vary greatly, an assumption of 30 mA is consistent with many datasheets. A conservative approach would be to limit the maximum current to 25 mA, or a factor of 1.25. As a first order approximation, an increase of ΔV_{in} will increase V_{in}^2 by $2V_{in}\Delta V_{in}$. Considered at the operating point $V_{in} = 17\,\text{V}$:

$$\frac{2V_{in}\Delta V_{in}}{V_{in}^2} < 0.25$$

$$\Delta V_{in} < 2.125\,\text{V}$$

$$V_{ripple} = 2\Delta V_{in} < 4.25\,\text{V}$$

Conservatively, the voltage ripple should be well below the maximum for both of these voltage ripple considerations. Taking a factor of 10 to be sufficient, the voltage ripple should be no more than 400 mV. For the LEDs, this ripple can come from the boost or buck converter. Since the buck converter has more conflicting needs, most of the 400 mV will be budgeted for it, and only 25 mV will be budgeted for the boost converter.

4.2 Choosing Components

The boost converter can be run with a logic level NFET, since its source is referenced to ground. The RFP12n10L, provided in the lab kit, is a logic level NFET which can handle the voltage and current requirements of this circuit. It can handle both the 17 V output as well as the potential 20 mA or so which may be flowing through it.

The boost converter diode could be a MUR120, which can also handle the voltage and current requirements of this circuit.

The duty cycle of this circuit must be chosen such that $V_O = \frac{V_{in}}{1-D}$. Since $V_O = 18\,\text{V}$ and $V_{in} = 5\,\text{V}$, the requirement is $D = 0.72$.

The boost converter is required to be in CCM down to output powers of 0.28 W. This corresponds to an input current of $i_{in} = P_{out}/V_{in} = 56\,\text{mA}$.

$$L > \frac{V_{in}DT}{2\langle i_{in} \rangle}$$

$$L > \frac{5\,\mathrm{V} \cdot 0.72 \cdot \frac{1}{250\,\mathrm{kHz}}}{2 \cdot 56\,\mathrm{mA}}$$
$$L > 129\,\mu\mathrm{H}$$

The voltage ripple should be 25 mV or less. A conservative estimate for maximum output current, above the supposed maximum for the LEDs of 25 mA, would be 50 mA. The control circuitry may draw additional current through pullup resistors. Assuming low pullup resistor values, a conservative upper bound on the total output current is 2 A. This will turn out to be an overestimate due to high resistor values being used.

$$C > \frac{\langle i_O \rangle DT}{\Delta V_{spec}}$$
$$C > \frac{2\,\mathrm{A} \cdot 0.72 \cdot \frac{1}{250\,\mathrm{kHz}}}{25\,\mathrm{mV}}$$
$$C > 230\,\mu\mathrm{F}$$

The LC resonant frequency of the circuit should be far below the switching frequency of the circuit, 250 kHz. A factor of 10 should be sufficient. This gives the requirement:

$$\frac{1}{2\pi\sqrt{LC}} \leq 25\,\mathrm{kHz}$$
$$LC > 4.05 \cdot 10^{-11}$$

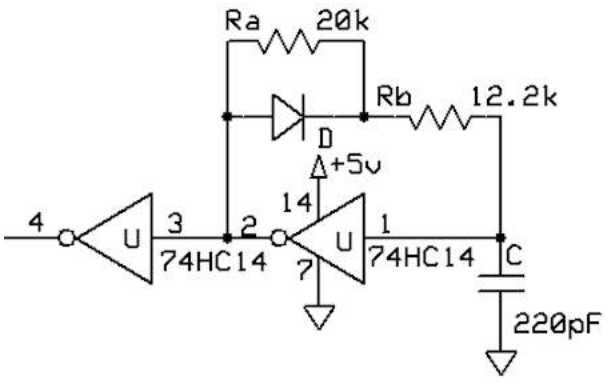
Any capacitor and inductor meeting the previous constraints will also meet this constraint.

To meet the CCM constraint, a 300 µH inductor can be constructed on the T-106 toroid core with 57 turns. A prewound inductor cannot be used, since the power dissipated in the LEDs, safety resistor, and BJT pullup resistors will exceed its 0.5 W rating.

An appropriate capacitor is 3 330 µF electrolytic capacitors, in parallel with a 10 µF ceramic capacitor. This exceeds the constraint as well as providing a good high frequency response, which is necessary given the switching frequency.

As provided by the constraint, the safety resistor must dissipate at least 0.28 W. At an 18 V output, this resistor must be less than $\frac{18\,\mathrm{V}^2}{0.28\,\mathrm{W}} = 1157\,\Omega$. A reasonable choice of safety resistor is 3 2.2 kΩ resistors in parallel. Each resistor is rated for 0.25 W, so multiple must be used to implement it.

A 74HC14 chip can be used to create the PWM signal for the boost converter. The duty cycle required for a 5 V to 18 V boost converter is 0.72. A 1N4148 diode is used to short across one of two resistors while the capacitor is charging, creating a duty cycle of less than 50% in one inverter. This is fed to another inverter on the chip as a buffer, creating the greater than 50% duty cycle that is desired.



Keeping in mind that the diode may have a voltage drop around 0.6 V:

$$t_{OFF} = RC \ln \frac{V_{T+}}{V_{T-}} = RC \ln \frac{2.85}{1.75} = 0.488RC = 0.488(R_A + R_B)C$$

$$t_{ON} = RC \ln \frac{V_{CC} - V_{T-}}{V_{CC} - V_{T+}} = RC \ln \frac{5 - 0.6 - 1.75}{5 - 0.6 - 2.85} = 0.536RC = 0.536R_BC$$

$$\frac{t_{OFF}}{t_{ON} + t_{OFF}} = 0.72$$

$$0.28t_{OFF} = 0.72t_{ON}$$

$$0.28(0.488(R_A + R_B)C) = 0.72(0.536(R_BC))$$

$$0.137R_A + 0.137R_B = 0.386R_B$$

$$0.137R_A = 0.249R_B$$

$$R_A = 1.824R_B$$

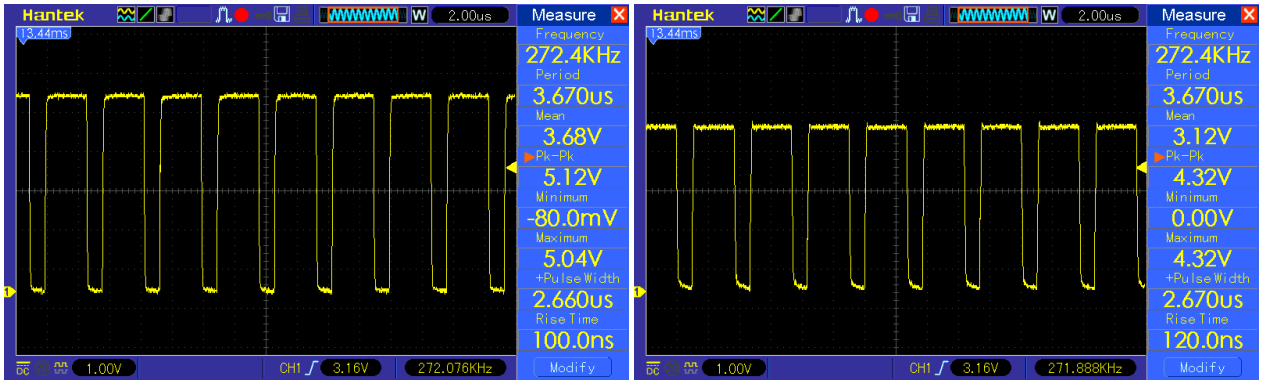
$$T = t_{ON} + t_{OFF} = (0.488R_A + 1.024R_B)C = \frac{1}{250\,\text{kHz}}$$

These requirements can be roughly satisfied with $C = 680\,\text{pF}$, $R_A = 3.2\,\text{k}\Omega = 1\,\text{k}\Omega + 2.2\,\text{k}\Omega$, $R_B = 5.7\,\text{k}\Omega = 4.7\,\text{k}\Omega + 1\,\text{k}\Omega$.

After much adjustment, the desired duty cycle was achieved with $C = 330\,\text{pF}$, $R_A = 10\,\text{k}\Omega + 4.7\,\text{k}\Omega$, $R_B = 10\,\text{k}\Omega$.

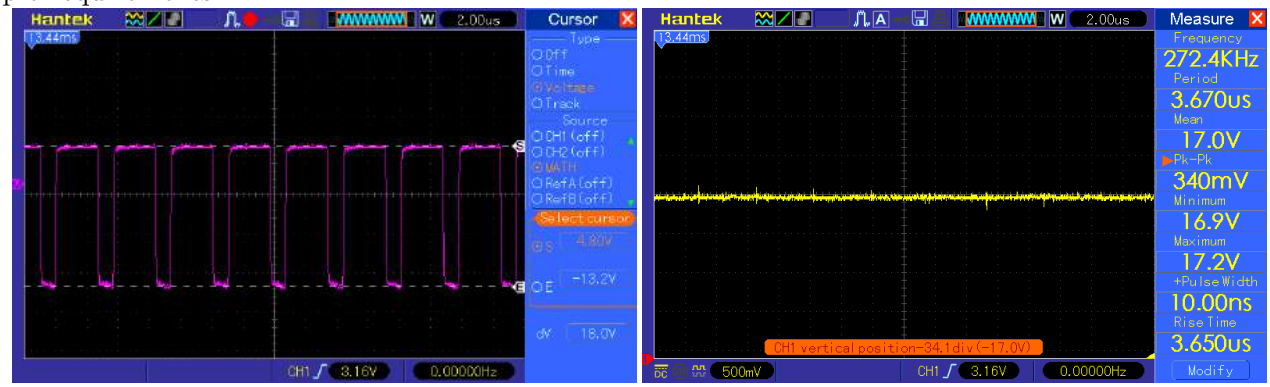
4.3 Results

The built circuit conformed to expectations. Shown below is the output of the clock stage and subsequent gate voltage on the logic-level NFET in the boost converter.



As determined by the oscilloscope, the duty cycle achieved is roughly $\frac{2.670\,\mu s}{3.670\,\mu s} = 0.728$.

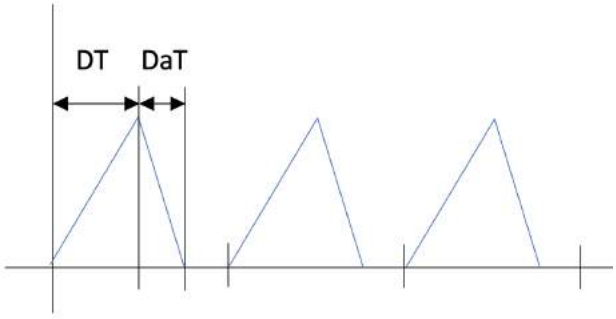
The inductor voltage (shown below in purple) swings between about 5 V (from the input voltage) and -13 V. The inductor is operating in CCM as designed with no off period or LC oscillations evident. The output voltage is a steady approximately 17 V when loaded, and is well within voltage ripple requirements.



5 DCM Buck Converter

5.1 Equations

In DCM, a buck converter's inductor current will ramp down after a time D_aT which is less than $(1-D)T$.



When the switch is open, the inductor has a voltage of V_o across it, and when the switch is closed it has a voltage of $V_{in} - V_o$. Thus,

$$\Delta i_L = \frac{(V_{in} - V_o)DT}{L} = \frac{V_o D_a T}{L}$$

$$(V_{in} - Vo)D = VoD_a$$
$$D_a = \frac{(V_{in} - V_o)D}{V_o} = (\frac{V_{in}}{V_o} - 1)D$$

DCM requires that $D_a < (1 - D)$.

$$D_a < 1 - D$$
$$(V_{in} - V_o)D < V_o(1 - D)$$
$$V_{in}D - V_oD < V_o - V_oD$$
$$V_{in}D < V_o$$
$$D < \frac{V_o}{V_{in}}$$
$$D_{max} = \frac{V_o}{V_{in}}$$

The average inductor current can be found by integrating the inductor current curve, i.e. finding the charge and dividing by the period.

$$\langle i_L \rangle = \frac{Q}{T}$$
$$Q = \frac{1}{2}bh = \frac{1}{2}(D + D_a)T \cdot \frac{(V_{in} - V_o)DT}{L}$$
$$\langle i_L \rangle = \frac{1}{2}(D + (\frac{V_{in}}{V_o} - 1)D) \cdot \frac{(V_{in} - V_o)DT}{L}$$
$$\langle i_L \rangle = \frac{1}{2}(\frac{V_{in}}{V_o}D) \cdot \frac{(V_{in} - V_o)DT}{L}$$
$$\langle i_L \rangle = \frac{\frac{V_{in}}{V_o}(V_{in} - V_o)D^2T}{2L}$$

In steady state, the average capacitor current is 0, so the average output current is the same as this average inductor current.

Based on this output current, a value for the maximum inductance can be calculated.

$$L_{max} = \frac{\frac{V_{in}}{V_o}(V_{in} - V_o)D_{max}^2T}{2\langle i_o \rangle_{max}}$$

The output voltage ripple can be found by computing the charge accumulating on the capacitor when it is charging, since $Q = C\Delta V$. The capacitor will only be charging when $i_L > \langle i_o \rangle$. This charge lies in a triangle between the points where the inductor current rises above and drops below $\langle i_o \rangle$.

The height of the triangle is then $i_{pk} - \langle i_L \rangle$. The value of i_{pk} is the same as $\Delta i_L = \frac{(V_{in} - V_o)DT}{L}$. The slope of the rising edge of the triangle is $\frac{i_pk}{DT}$. Thus, the base of this portion of the triangle is $\frac{(i_{pk} - \langle i_o \rangle)DT}{i_{pk}}$. The slope of the falling edge of the triangle is $-\frac{i_{pk}}{D_aT}$. Thus, the other portion the base is $\frac{(i_{pk} - \langle i_o \rangle)D_aT}{i_{pk}}$.

$$Q = \frac{1}{2}(i_{pk} - \langle i_o \rangle) \cdot (\frac{(i_{pk} - \langle i_o \rangle)DT}{i_{pk}} + \frac{(i_{pk} - \langle i_o \rangle)D_aT}{i_{pk}})$$

$$Q = \frac{1}{2i_{pk}}(i_{pk} - \langle i_o \rangle)^2 (D + D_a)T$$

The condition on a capacitor value is then

$$C > \frac{Q}{\Delta V_{spec}}$$

5.2 Specifications and Component Values

The buck converter should operate at a higher frequency than the PWM of the LEDs themselves, in order to reach steady state quickly on the timescale of the PWM. A reasonable choice of frequency is 100 kHz. The PWM frequency will be at minimum 100 Hz to avoid the perception of flicker. To support a range of 128 different brightness levels, 0-127, the smallest pulse widths will be at 1/127 of the period, or 12.7 kHz.

The output voltage on the capacitor will rise or decay to steady state when the number of LEDs on the output changes. For removing LEDs, based on LTSpice simulations, with low resistance values this is dominated by a worst-case constant 100 mA as the capacitor voltage linearly decays to its new value. Following this, RC time constants bring the output current to its final value. Ramp-up, on the other hand, is fairly instant.

Two constraints govern how fast this transition must occur. First, the LEDs cannot sustain a current of 100 mA. Most datasheets for similar LEDs state that the LEDs can handle such a current for a 10% duty cycle and maximum pulse width of 0.1 ms. Since the software will ensure there is a maximum of one transition to a lower number of LEDs, and with a 10 ms period, this can be met by ensuring a pulse width of less than 0.1 ms.

$$T_{transition} < 100\,\mu s$$

The other constraint is that the transition should occur fast enough on the timescale of the PWM for the LEDs. That is, within the lowest possible duty cycle, $\frac{1}{127}$, the transition should occur at least one-half of the way into the pulse.

$$T_{transition} < \frac{1}{12.7\,\text{kHz}} \cdot 0.5 = 39.4\,\mu s$$

Since $T_{transition}$ depends approximately on a pulse plus RC decay for low values of RC (found by simulation):

$$T_{transition} \approx \frac{\Delta V}{\frac{i_C}{C}} + 5RC < 39.4\,\mu s$$

given that the output current will settle in 5 time constants.

The buck converter has 8 different potential output voltages, depending on how many, 0-7 of the LEDs are connected to the load. With $D_{max} = \frac{V_o}{V_{in}}$, the value obtained for L_{max} will be greatest either with all 7 of the LEDs connected to the load, or only 1. Each LED contributes 2 V to the output voltage, and will operate with a maximum current of 20 mA. Using a model for the LED of a 1.94 V source and series 4.5 Ω resistance found in a previous lab, and a current limiting resistor of 10 Ω, the voltage across a full load would be $7 \cdot (1.94 + 0.02 \cdot 4.5) + 0.02 \cdot 10 = 14.41$ V. The voltage across a one LED load would be $(1.94 + 0.02 \cdot 4.5) + 0.02 \cdot 10 = 2.23$. The input to this stage from the buck converter will be approximately 17 V, down from the ideal 18 V due to power losses.

$$L_{max} = \frac{\frac{V_{in}}{V_o}(V_{in} - V_o)D_{max}^2 T}{2\langle i_o \rangle_{max}}$$

$$L_{max} = \frac{\frac{17\,\mathrm{V}}{14.41\,\mathrm{V}}(17\,\mathrm{V} - 14.41\,\mathrm{V})(\frac{14.41\,\mathrm{V}}{17\,\mathrm{V}})^2 \frac{1}{100\,\mathrm{kHz}}}{2 \cdot 20\,\mathrm{mA}}$$

$$L_{max} = 549\,\mu\mathrm{H}$$

$$L_{max} = \frac{\frac{17\,\mathrm{V}}{2.23\,\mathrm{V}}(17\,\mathrm{V} - 2.23\,\mathrm{V})(\frac{2.23\,\mathrm{V}}{17\,\mathrm{V}})^2 \frac{1}{100\,\mathrm{kHz}}}{2 \cdot 20\,\mathrm{mA}}$$

$$L_{max} = 484\,\mu\mathrm{H}$$

An inductor safely meeting these conditions is a 330 µH pre-wound inductor from the lab kit. With a relatively low output power (0.28 W + 0.004 W for the resistor) required even at full load for the LEDs, the 0.5 W prewound inductors are sufficient.

The voltage ripple should be no more than 375 mV. Operating under the worst case scenario, only 1 LED would be connected to the load.

$$D = \sqrt{\frac{2\langle i_L \rangle L}{\frac{V_{in}}{V_o}(V_{in} - V_o)T}} = \sqrt{\frac{2 \cdot 20\,\mathrm{mA} \cdot 330\,\mu\mathrm{H}}{\frac{17}{2.23}(17 - 2.23)\frac{1}{100\,\mathrm{kHz}}}} = 0.1083$$

$$i_{pk} = \frac{(V_{in} - V_o)DT}{L} = \frac{(17 - 2.23) \cdot 0.1083 \cdot \frac{1}{100\,\mathrm{kHz}}}{330\,\mu\mathrm{H}} = 48.5\,\mathrm{mA}$$

$$Q = \frac{1}{2 \cdot 48.5\,\mathrm{mA}} \cdot (48.5\,\mathrm{mA} - 20\,\mathrm{mA})^2 \cdot (0.1083 + 1 - \frac{2.23}{17}) \cdot \frac{1}{100\,\mathrm{kHz}} = 81.7\,\mathrm{nC}$$

$$C > \frac{81.7\,\mathrm{nC}}{375\,\mathrm{mV}}$$

$$C > 218\,\mathrm{nF}$$

From simulation this is an overestimate for required capacitance, since the output current ripple changes the charge-time of the capacitor. Acceptable voltage and current ripple can be achieved with $C = 200\,\mathrm{nF}$ ceramic capacitor, implemented with two 100 nF capacitors.

Given this value of capacitor and the constraint on the transition time, an appropriate series resistor value can be calculated. The largest transition is from all 7 LEDS loaded to 0 LEDs loaded (current burst not seen going to 0 LEDs loaded, since the duty cycle will be reduced to 0 before then). This is 14.41 V to 0 V (again using 10 Ω as an approximation for R_S)

$$\frac{\Delta V}{\frac{i_C}{C}} + 5RC < 39.4\,\mu\mathrm{s}$$

$$\frac{14.41\,\mathrm{V}}{\frac{100\,\mathrm{mA}}{200\,\mathrm{nF}}} + 5(4.5\,\Omega + R_S)(200\,\mathrm{nF}) < 39.4\,\mu\mathrm{s}$$

$$R < 6.08\,\Omega$$

A reasonable choice is $R = 5\,\Omega$. It can handle the 100 mA burst with its 0.25 W power rating. Since this value is less than the 10 Ω estimate, the other values calculated are conservative and do not need adjustment. Note that this resistor is not integral to the circuit and is only there to

provide some extra safety were something to go wrong; these calculations are simply to find what values of resistors would not interfere with circuit operation.

Using these circuit values, the required duty cycles and constraints for each of the 8 different load conditions can be calculated. The goal current is 20 mA in each case.

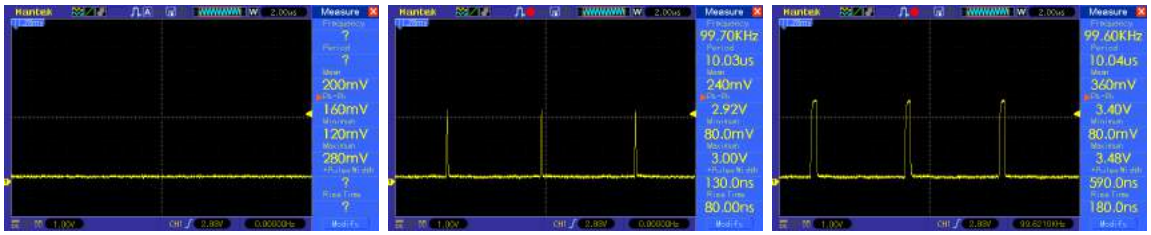
# LEDs	V_o	D_{max}	D	$1 - D$	Current (-1 LED)	Current (+1 LED)
0	0.1 V	0.0059	0.0	1.0	N/A	0.0 mA
1	2.13 V	0.1253	0.1055	0.8945	484.16 mA	8.84 mA
2	4.16 V	0.2447	0.1586	0.8414	45.24 mA	11.32 mA
3	6.19 V	0.3641	0.2109	0.7891	35.35 mA	12.23 mA
4	8.22 V	0.4835	0.2696	0.7304	32.7 mA	12.33 mA
5	10.25 V	0.6029	0.3434	0.6566	32.44 mA	11.67 mA
6	12.28 V	0.7224	0.4495	0.5505	34.27 mA	9.78 mA
7	14.31 V	0.8418	0.6427	0.3573	40.89 mA	N/A

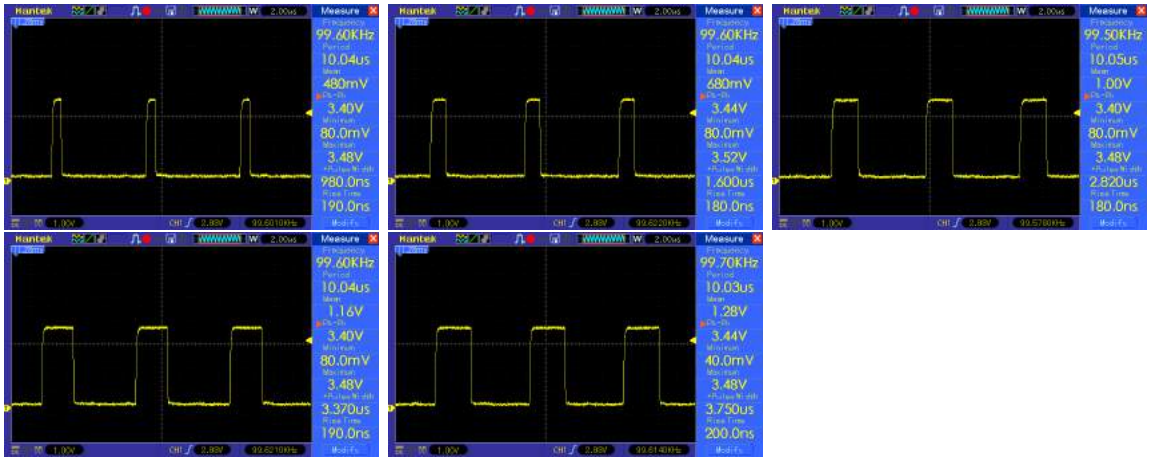
As evident from the table, changing the duty cycle in tandem with the LEDs is important. As mentioned before, to reduce current spikes it is advisable to align the periods of the 7 LEDs and have the LEDs be "first off" and "then on", so that the number of LEDs is always increasing until the end of the period, when they all turn off together.

For the MOSFETs controlling the LEDs, a good choice of component is the IRF540. These MOSFETs can handle the full 18 V supply voltage on its gate or from drain to source. Since the MOSFETs are connected across the LEDs, the source pin is "floating". It is important then to ensure that the gate-source voltage will always meet the threshold voltage. The threshold voltage of the IRF540 is 2 V-4 V. The gate will always be driven by 18 V. The source voltage will be highest on the first LED in the series. There can be up to 6 LEDs below it in series, which may be a total of around 12 V. This means the minimum gate-source voltage in the circuit is $18\,\text{V} - 12\,\text{V} = 6\,\text{V}$. This exceeds the threshold voltage, so all MOSFETs should operate properly.

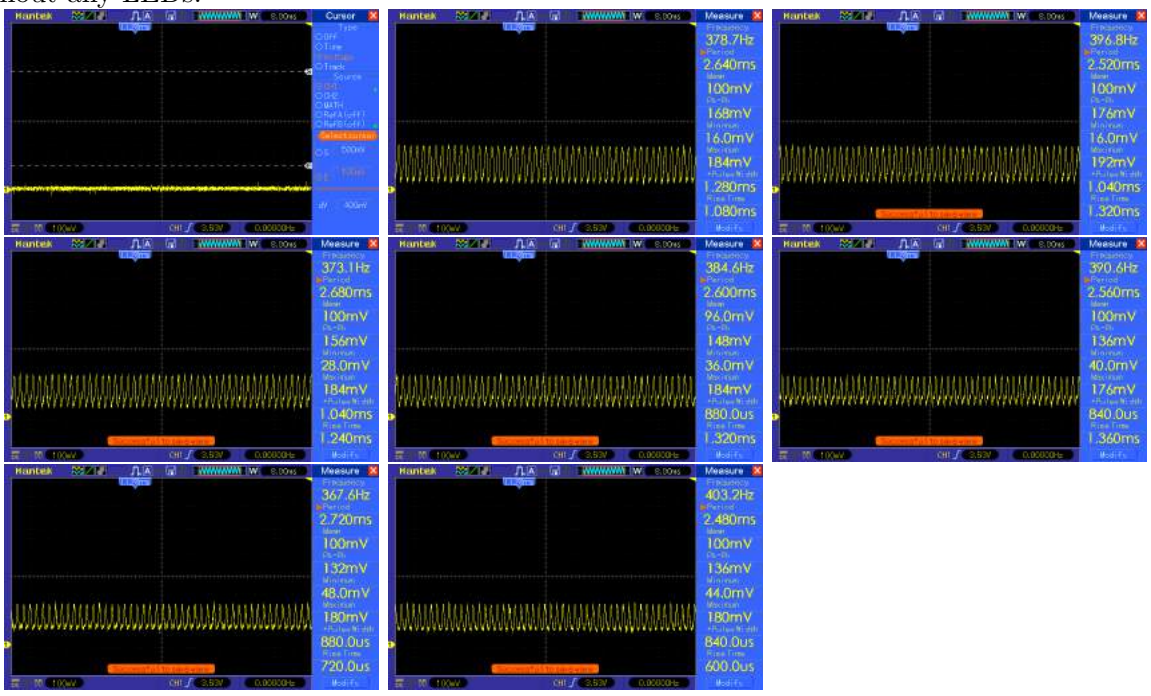
5.3 Build

The duty cycle of the buck converter is controlled through a PSoC digital output pin, so is discretized to values between 0 to 255. Due to the unaccounted-for losses and RC time constants throughout the buck converter, the actual duty cycles used to acheive a constant 20 mA current do not match the above table exactly and had to be adjusted. The values (out of 255) eventually used to achieve the proper duty cycles for 0-7 LEDs being "on" were 0, 8, 20, 30, 46, 77, 91, and 101. Shown below are waveforms from the PSoC pin for these different number of LEDs. Note that the duty cycle is effectively "negated" twice - once due to using an NPN transistor as an open drain for the output of the PSoC, and once due to using a PFET for the buck converter.

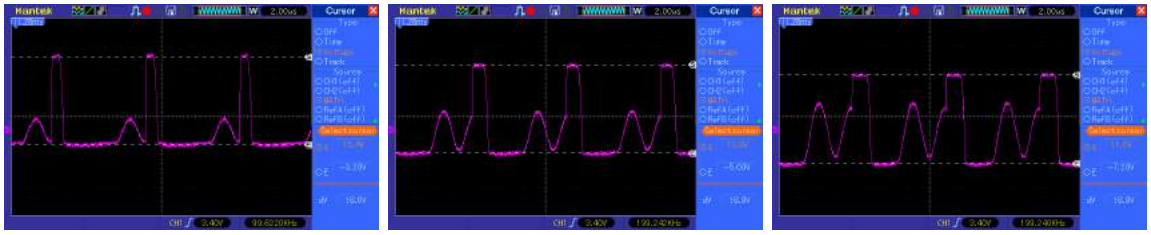


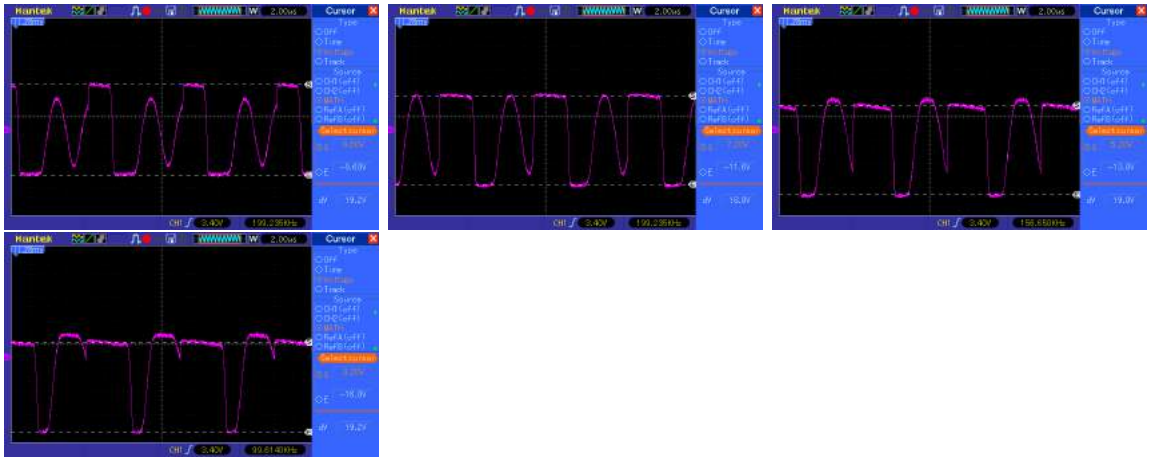


As shown below, for each number of LEDs the buck converter's duty cycle is chosen to maintain an average 20 mA current through the LEDs. The current is measured through a 5 Ω resistor, meaning that a 100 mV voltage is expected. The current ripple decreases with increasing numbers of LEDs. For 0 LEDs on, the current is 0mA as it is unnecessary to maintain a 20 mA current without any LEDs.



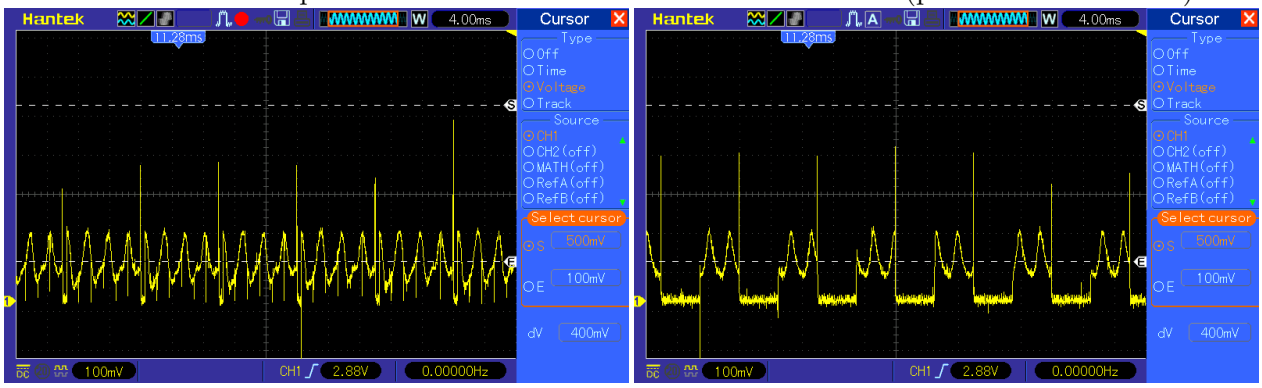
The buck converter is designed to be in DCM for any number of LEDs. Shown below is the voltage across the inductor for 1 to 7 LEDs in the on state. In all cases, there is evidence of an "off period" with LC oscillations.



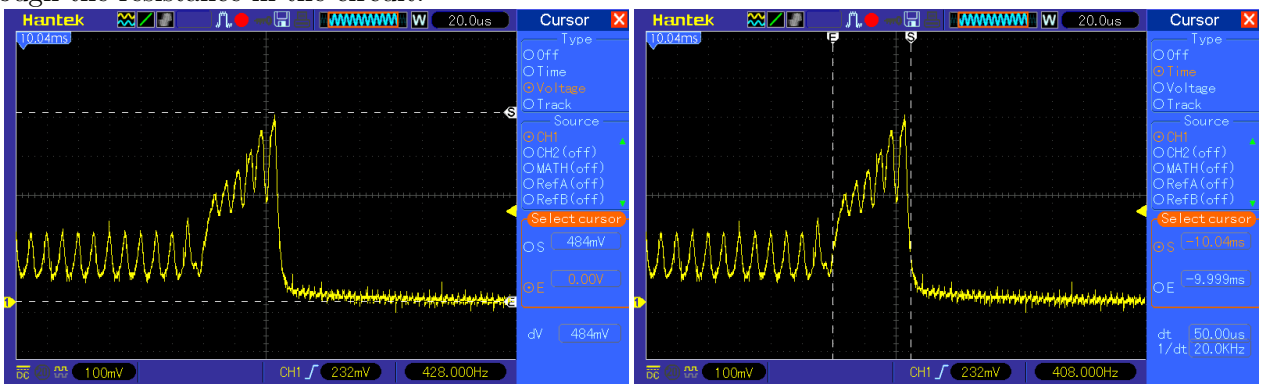


The buck converter additionally performs well under the target conditions, with each LED turning on and off at a PWM rate of 100 Hz. The resolution for this PWM is 0 to 127. To evaluate performance, a test pattern A of 0, 20, 40, 60, 80, 100, and 120 brightness values on the LEDs and a test pattern B with 63 on each LED was used.

Below is shown the output current under each of these test conditions (pattern A on the left).



Each pattern maintains a constant 20 mA current (100 mV measured through the 5 Ω resistor). For the pattern of 50% brightness - a value of 63 for each LED - there is a visible "off period" during which all 7 LEDs are off and 0 mA. Also visible on both waveforms are current spikes at the conclusion of every period. These current spikes are expected: when the duty cycle is brought down to 0 and all LEDs are turned off, the capacitor voltage will be maintained briefly and discharged through the resistance in the circuit.

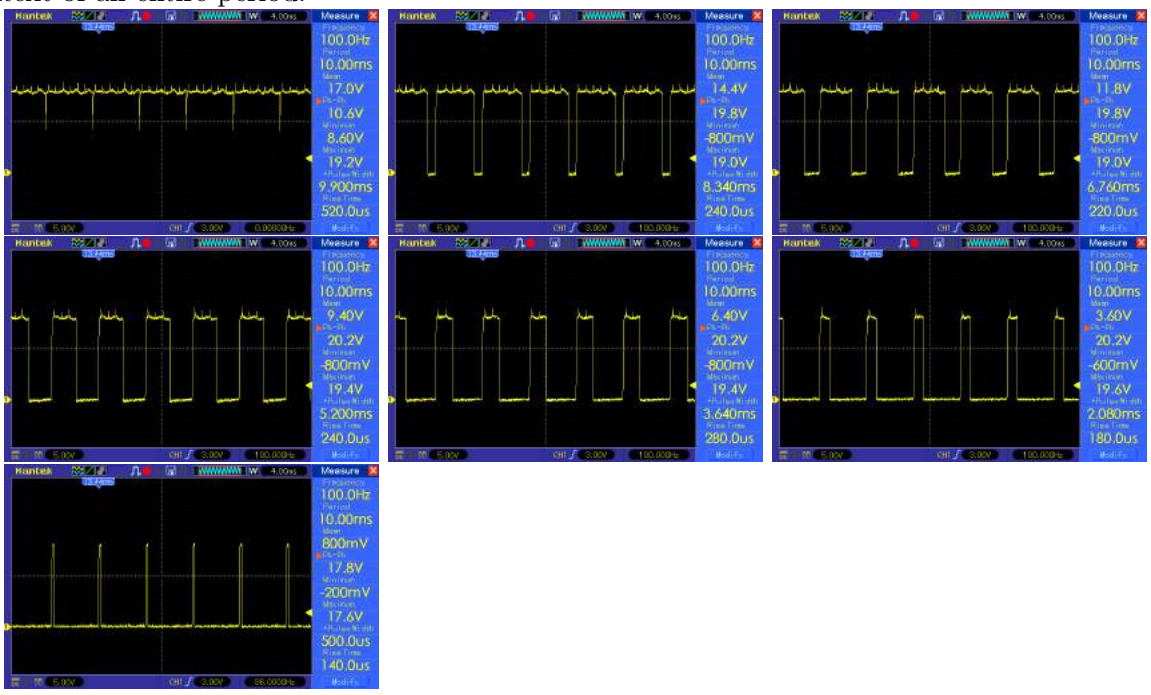


These current spikes (actually made of a series of shorter spikes due to differing turn-on speeds of the MOSFETs) are within specifications for the LEDs. The spikes are less than 100 mA - the maximum current for the LEDs, they are less than 0.1 ms long, and they occur for far less than

1/10 of a duty cycle. It is also reasonably short compared to the period of the PWM.

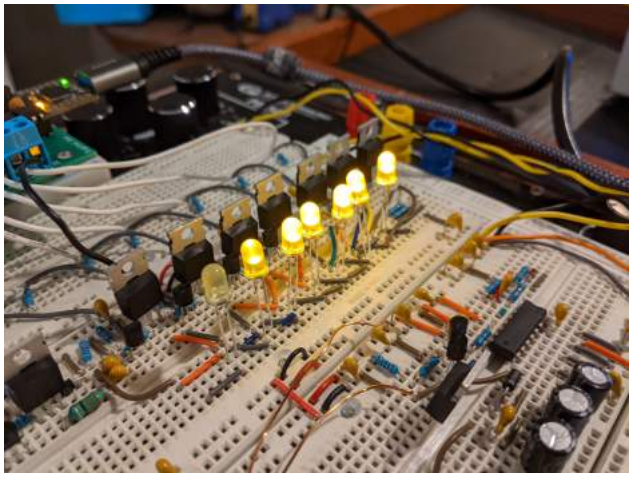
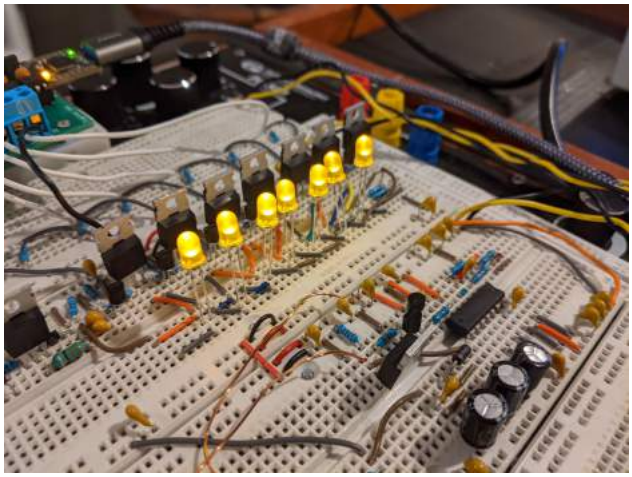
However, with the original circuit design, perhaps due to an inaccurate LED model, the current spikes were far bigger (from 200 mA - 400 mA). To mitigate this, a much higher resistance was used for the pullup resistors on the MOSFET gates. This ensured a low gate current, and thus a gradual turn-on of the MOSFETs. This allowed the output voltage to diminish slowly, reducing the current spikes to a level within specification. This does have the negative effect of reducing accuracy for lower brightness levels, which is especially seen in the difference in behavior between LEDs - due to lower gate-source voltages on LEDs which are first in the series, their turn-on times are different, resulting in different brightnesses even with the same apparent duty cycle from the PSoC. Notably, the increased pullup resistance only affects the turn-on time of the MOSFET. Since the NPN transistor will pull the gate to ground when it is on, the turn-off time of the MOSFET will be very quick comparatively. This means the desired effect of gradually turning off the LEDs (by turning on the MOSFETs) can be achieved without sacrificing the LED turn-on speed.

Shown below are the gate voltages for different MOSFETs under a pattern (pattern A) with different LED brightnesses. The slow rise time for the gate is not very severe when viewing in the context of an entire period.



6 Full System

Below are photographs of the entire system working. First shown are the lights displaying the two built-in patterns (test patterns A and B, with different and equal brightnesses respectively).

Shown below is the PSoC control panel when in scene program mode; the status LEDs are displaying a new pattern being programmed.