

Edited by Bill Travis

Latching power switch uses momentary-action pushbutton

Anthony Smith, Scitech, Biddenham, Bedfordshire, UK

MOST INEXPENSIVE pushbutton switches, particularly pc-board-mounting and membrane types, have momentary action. Latching types are often larger and relatively expensive, and they frequently are unavailable in the style you'd like to use. You can thus have a problem if you need a small, inexpensive on/off switch for latching power to a load. The circuit in **Figure 1** shows how you can use a simple, momentary-action, SPNO (single-pole, normally open) pushbutton switch to latch power to a load. Requiring just a handful of common, garden-variety components, the circuit works over a wide voltage range and is ideal for single-cell applications, because it can operate at voltages as low as 1V or less. Transistors Q_2 and Q_3 form an SCR-like structure that functions as a simple latch, Q_4 switches power to the load, and S_1 is the momentary pushbutton switch.

When you first apply the supply voltage, V_s , all four transistors are off, and capacitor C_1 charges via R_1 and R_2 until its voltage, V_{C1} , is equal to V_s . The circuit is now in its off, or unlatched, state, and the load voltage, V_L , is 0V. A momentary closure of the pushbutton switch, however, causes C_1 to dump its charge into the base of Q_3 , which conducts and furnishes bias

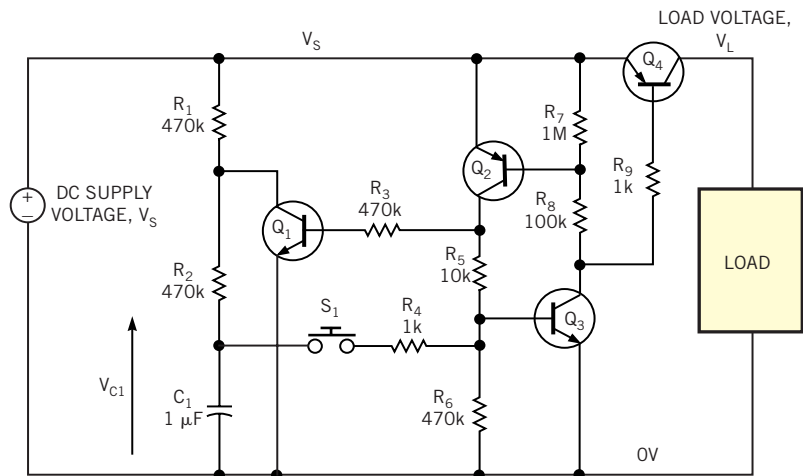


Figure 1 The momentary-action pushbutton switch, S_1 , provides positive latching action in this circuit.

for Q_2 and Q_4 , which both turn on. Q_2 now provides base bias for Q_3 via R_3 , and also for Q_1 via R_3 . The circuit is now in its on, or latched, state and remains that way even though S_1 is open. The load is now energized, and V_L is roughly equal to V_s . Transistor Q_1 is now saturated, causing C_1 to discharge via R_2 such that V_{C1} falls to a few tens of millivolts (Q_1 's collector-emitter saturation voltage). Another momentary closure of the pushbutton switch couples this low voltage to Q_3 's base, turning it off. As a result, all four transistors turn off, and the circuit reverts to its off, or unlatched, state. The load is now de-energized, and V_L falls to 0V. Because Q_1 is now off, C_1 begins to charge again via R_1 and R_2 , such that another momentary closure of S_1 latches the circuit on again.

Timing capacitor C_1 , acting with R_1 and R_2 , provides debouncing for the pushbutton switch, such that contact bounce has no effect on the desired latching function. Without the RC time delay,

the circuit would “chatter” on and off each time you pressed the pushbutton switch and would end up in an indeterminate state. Although **Figure 1** shows a value of 1 μ F, other values may be more suitable for a particular application, so prepare to experiment. None of the resistor values is particularly critical, and the values shown in **Figure 1** are fairly optimal for a supply voltage of approximately 1 to 1.5V—in other words, a single cell. At higher voltages, the resistor values should increase proportionally, although you should hold R_2 and R_4 constant at approximately 470 and 1 k Ω , respectively. Keeping the R_2 - C_1 time constant fixed at a few hundred milliseconds ensures that the time taken to discharge the capacitor is not excessive; otherwise, once the circuit has been latched, there may follow an unacceptable delay before it can be unlatched. Resistor R_4 limits the current flowing from C_1 into Q_3 's base to a safe level; its value should be fairly small to ensure that R_5 and R_6 do

- Latching power switch uses momentary-action pushbutton 101
 - Method provides overpower protection for quasiresonant supplies 102
 - Circuit delivers dimming control for white-LED driver 106
 - System implements digital-clock modulation 108
- Publish your Design Idea in EDN. See the What's Up section at www.edn.com.**

a result, I_p needs to decrease in response to the feedback-loop requirements. For a widely varying line-voltage application, the peak current almost doubles between high and low input voltages for a constant output power. But quasiresonant controllers feature only overcurrent protection. This limitation is part of a structural problem. The controller monitors the peak current, and, when they reach the maximum allowed value, the controller circuitry detects an overload. Unfortunately, if the power supply delivers its nominal power at the lowest worst-case input voltage, it delivers more power for a higher input voltage. For a widely varying line-voltage application, this power could be more than three times higher. This fact is the consequence of the flyback equation.

A classic way to compensate this variable-power effect is to create an offset on the current-sense pin that compensates for the peak-current variations as a function of the input voltage, V_{IN} . You obtain this effect using an overpower-protection scheme—wiring a compensation resistor from the high-voltage rail to the current-sense information (Figure 1a). Unfortunately, you cannot always implement this scheme. Whether you use the CS (current-sense) pin for another function

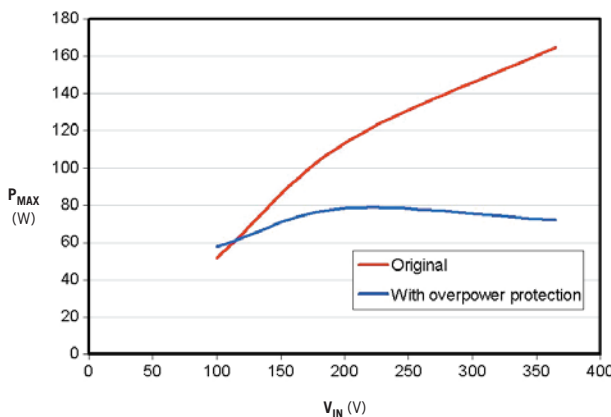


Figure 2 The output power rises to 165W without compensation but remains within 70W using the compensation scheme.

or you need to keep the pin impedance low for noise purposes, it forces you to adopt a low value for resistor R_{CS} in series with the current-sense information. It then requires a low-value compensation resistor, R_{COMP} , wasting a lot of power. When you need low standby power, this approach is unacceptable. To overcome the problem, it might be useful to use a fraction of the input voltage to lower the voltage drop on R_{COMP} . The power the resistor wastes would then become negligible.

You achieve this method by using the forward voltage of an auxiliary winding. On a forward winding, a voltage proportional to V_{IN} occurs during the on-time, which is the scenario you are looking for. Usually, you use a flyback auxiliary wind-

ing to supply the controller and to detect the core-reset event. By modifying the arrangement of the winding, you can generate the flyback information for the demagnetization detection during off-time and combine, on the same winding, the forward information for the overpower compensation during on-time. By adding a diode in series with the auxiliary winding, you can access the forward voltage (Figure 1b). This forward voltage is proportional to $N \times V_{IN}$, where N is the turns ratio between the primary and the auxiliary windings. You

add R_{FWD} to supply the reverse current during the forward activity.

Knowing the value of the forward voltage and the series resistor, R_{CS} , you can then easily calculate the value of compensation resistor R_{COMP} to create the desired offset on the current-sense signal at high input voltage. On a demonstration board built on the NCP1207 from On Semiconductor (www.onsemi.com), D_1 is a 1N4448 diode, $R_{CS} = 680\Omega$, $R_{COMP} = 18\text{ k}\Omega$, and $R_{FWD} = 4.7\text{ k}\Omega$, the protection toggles at 60W at 100V dc and 70W at 365V dc, instead of 55W at 100V dc and 165W at 365V dc without compensation (Figure 2). Figure 3 shows circuit waveforms of the line compensation at $V_{IN} = 365\text{V}$, and Figure 4 shows the waveforms at $V_{IN} = 100\text{V}$. □

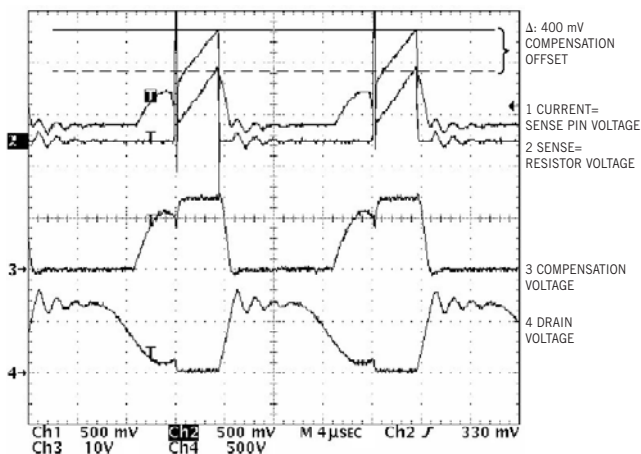


Figure 3 These waveforms represent line compensation at input voltage of 365V.

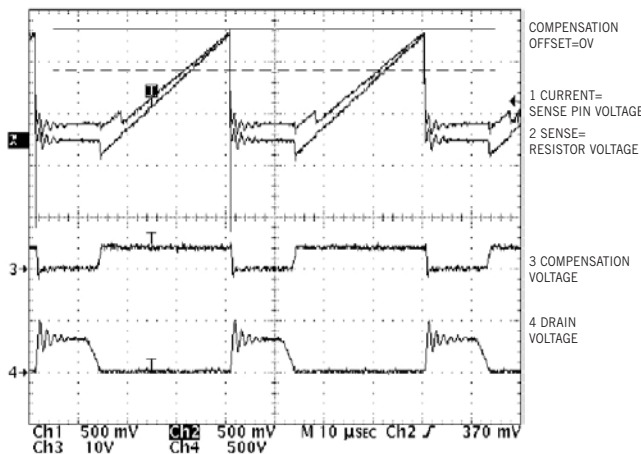


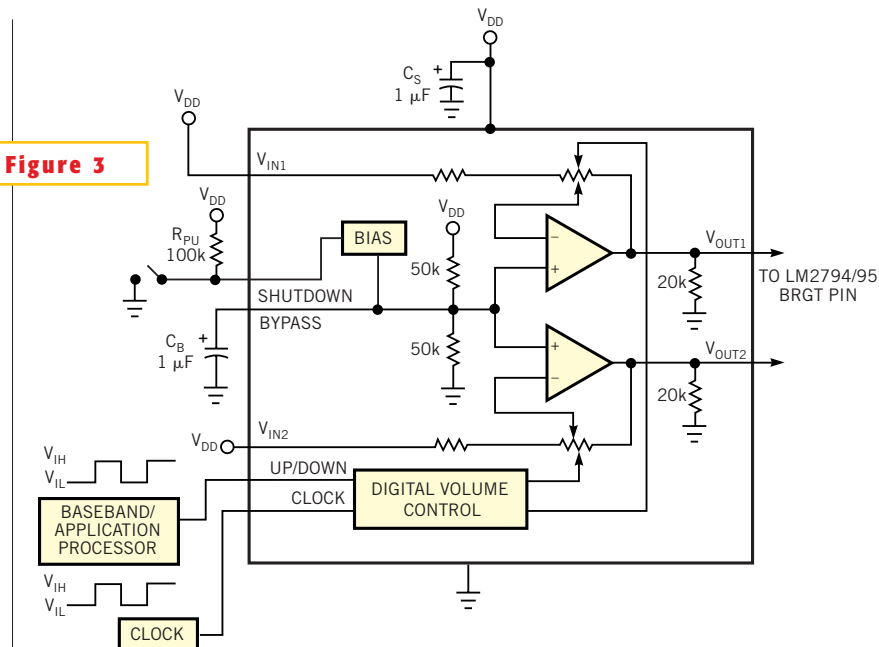
Figure 4 A set of waveforms accompanies line compensation at input voltage of 100V.

Circuit delivers dimming control for white-LED driver

Wallace Ly, National Semiconductor, Santa Clara, CA

THE DEMAND FOR POWER in color cell-phone handsets is constantly increasing. New applications steadily emerge, making it imperative to reduce power consumption in the design. Examination of usage data shows that more than 50% of the power consumed is in providing backlighting for the color screen. People use the phones for playing games and MP-3s and a multitude of other heavy-duty, multimedia functions. **Figure 1** shows the classic way to use a backlight-driver IC to provide dimming. By modulating an external PWM signal, the circuit controls the white-LED current. By adjusting the on/off-time ratio, or duty cycle, the circuit can provide drive ranging anywhere from full-on to full-off. This circuit relies on the fact that the baseband or application processor has a PWM timer available. A second method of obtaining dimming is to use an adjustable analog-input interface (**Figure 2**). The drawback of this approach is that it requires a DAC block in the digital-baseband or application processor.

This Design Idea presents a more widely adaptable approach to the dimming challenge. Although the LM4811 headphone amplifier is not designed to operate as a DAC, you can tweak it to do so. **Figure 3** shows the dimming application. The implementation is straightforward. The output of the LM4811 attaches to the BRGT pin of the LM2794. The output current from the LM4811 is directly proportional to the digital value stored in the digital-volume-control block. The rising edge of the clock, along with the polarity of the Up/Down pin, sets the appropriate output current of the LM4811 and, thus, the output current of



A headphone amplifier finds dual use in a white-LED dimming circuit.

the white-LED driver. The resultant approach requires only two general-purpose input/output lines, which are available in all modern baseband and application processors. Moreover, this method requires zero processor cycles once the LED current is set. The approach is therefore optimal for both software and hardware. □

the white-LED driver. The resultant approach requires only two general-purpose input/output lines, which are available in all modern baseband and application processors. Moreover, this method requires zero processor cycles once the LED current is set. The approach is therefore optimal for both software and hardware. □

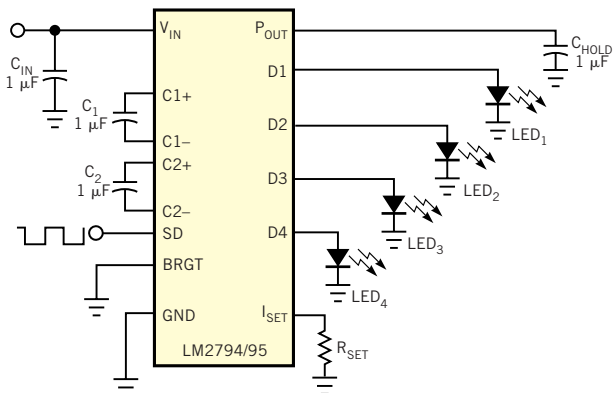


Figure 1 In this classic LED-dimming configuration, an external PWM signal varies the duty cycle of the driver.

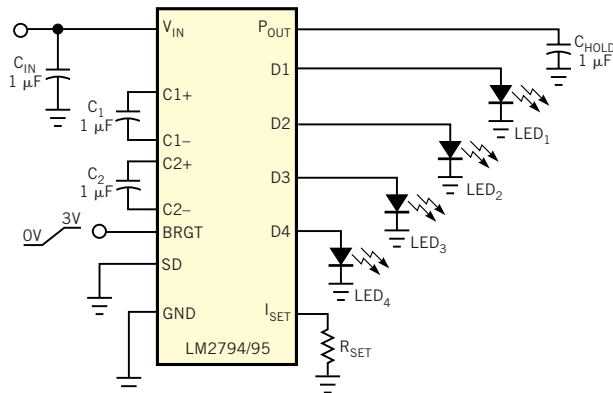


Figure 2 This dimming circuit requires the existence of a DAC in the application processor.

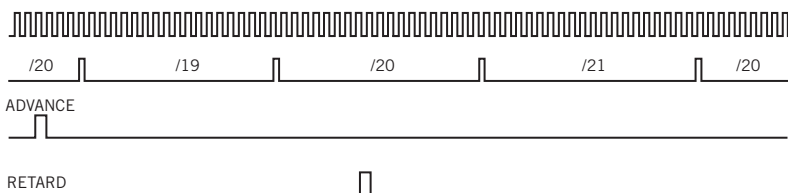
System implements digital-clock modulation

Dan Doberstein, DKD Instruments, Nipomo, CA

IN SPREAD-SPECTRUM and direct-sequence receivers, it's often necessary to change the frequency of the clock oscillator, and thus the spreading-code clock, to lock the receiver's reference pseudorandom-noise code to the incoming code. The original design used a VCXO (voltage-controlled crystal oscillator) for this function, but this revised design implements an all-digital method that provided additional functions, using a midlevel CPLD from Altera (www.altera.com). A commercially available part, the 74HC297, provides the function. This part is an all-digital PLL chip that most engineers have never heard of. The 74HC297 can digitally modulate an applied clock. It performs this task by adding or subtracting a pulse to or from a divided-down clock.

Upon examining the functions of this part, two issues emerge. First, it divides the input clock by two before modulating it. Second, the modulation method inserts and subtracts pulses. Although the pulse-addition/subtraction issue is acceptable, the divide-by-two function reduces the phase/frequency-control resolution. With the inspiration of the 74HC297 in mind, we sought a method that had finer resolution in phase and frequency for the same applied clock frequency. At first, we tried discrete counters using a digital switch to select different divide ratios. This approach worked at low clock frequencies but failed as clock speed increased because of propagation-delay effects at the switch. Eventually, we devised a method that used a long shift register and a tap-selection switch to implement a variable clock divider/phase modulator. **Figure 1** shows the block diagram of the system.

You load the 21-bit shift register with a single one and the rest zeros. A switch-selectable tap point feeds back to the shift register's input. In effect, a single one now goes around and around at a frequency, F_{OUT} , determined by the tap point and the applied clock, F_{CLK} . The switch function allows selecting one of three divide ratios:



NOTE: ADVANCE/RETARD PULSES TAKE EFFECT ON THE NEXT CYCLE, NOT THE CURRENT CYCLE.

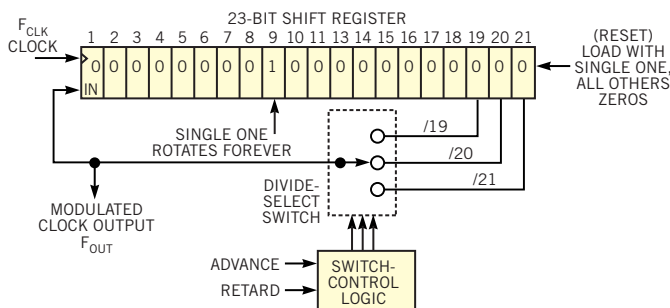


Figure 1 This scheme allows you to change the rate of the spreading-code clock in spread-spectrum applications.

19, 20, or 21. The choice of the tap points is a function of the target application's needs, although you can choose other tap points. In operation, if the system detects an Advance pulse, the switch selects the divide-by-19 for one F_{OUT} cycle. If the system detects a Retard pulse, the switch selects divide-by-21 for one F_{OUT} cycle. After every Retard or Advance pulse, the system returns the selection switch to the divide-by-20 position. The single one, as it shifts through the register, effects the setting and resetting of the switch. At first, it appears that this system will modulate the F_{CLK} input in an FM-like fashion. But, on second thought, it also acts as a phase modulator. On a single Advance pulse, approximately one-twentieth of an output cycle is subtracted, and a single Retard pulse adds approximately one-twentieth of an output cycle. If the Advance or Retard pulses are continuous streams at frequency F_{MOD} , the output frequency is $F_{OUT} = (F_{CLK} \pm F_{MOD})/N$, where $N=20$ in this case, and the sign is positive for Advance pulses and negative for Retard pulses.

So, this modulator can do both PM

and FM on the applied clock, F_{CLK} . The tap numbers this design uses reflect those that fit the target application. The target application needs approximately 1-MHz final output frequency with a phase resolution of one-twentieth of a cycle, or the equivalent of 50 μ sec in the time domain. If you need more phase resolution, increase the shift register's length. For example, a 101-stage shift register with tap points at 99, 100, and 101 has a phase resolution of approximately 3.6°. The advantages of the shift-register approach are speed and scalability. You can clock shift registers at extremely high speed. Additionally, the shift-register approach automatically provides you with clock signals that are exact advanced or delayed versions of the primary tap used for F_{OUT} . This feature is useful in creating precise phase-related signals. The feature also provides an advantage in the control of the switch function, so that you can manage propagation-delay issues.

Figure 2 shows the schematic in Altera's Max+Plus II design software. This software allows the use of standard logic-gate symbols in schematic format to cre-

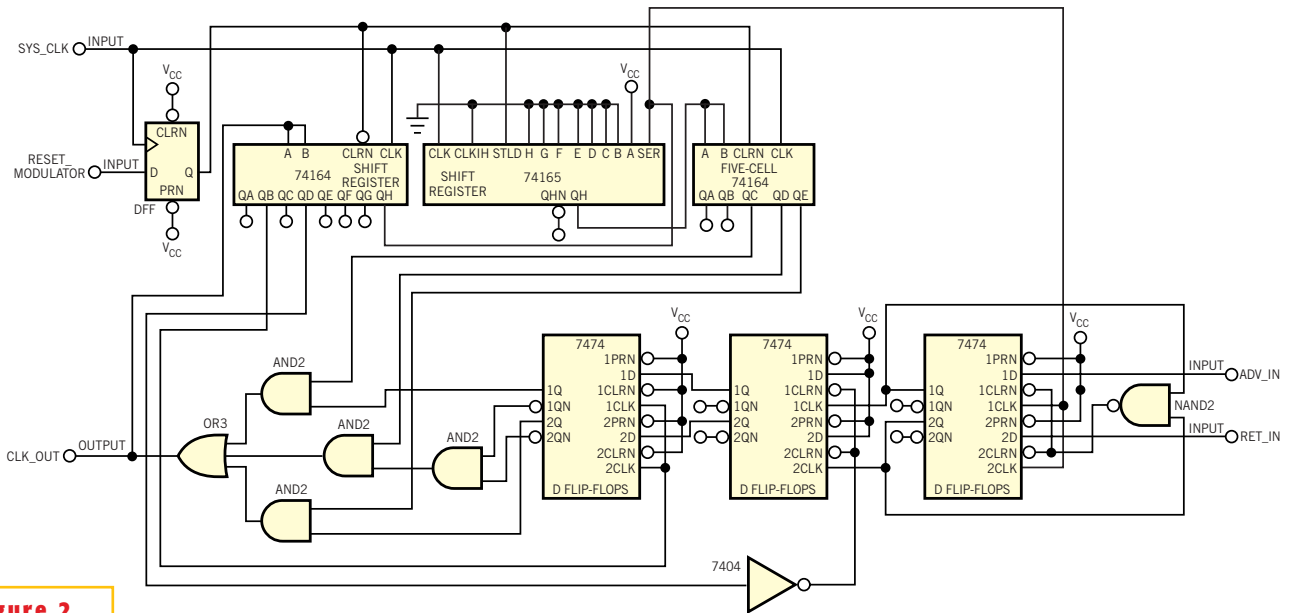


Figure 2

Altera's software allows you to enter standard logic-gate symbols to create a programmed CPLD.

(continued on pg 112)

ate a programming file for the device. Although this design uses a CPLD, a discrete version using digital-logic ICs, as the schematic shows, should also work. The F_{CLK} input connects to all the shift register's clock inputs. A Reset function initializes the registers with a single one and the rest zeros. Using a single flip-flop, F_{CLK} samples the asynchronous reset signal to ensure synchronous operation. Three sets of flip-flops prepare the Advance/Retard signal for use by the selection switch. From the right-hand side of

Figure 2, the first set of flip-flops on the Advance and Retard inputs ensures that these signals are synchronous with the final output frequency, F_{OUT} , and stores

them for the next cycle of F_{OUT} . We assumed that these signals would be asynchronous with the input clock, F_{CLK} . The NAND gate ensures that both of

these flip-flops clear if Advance/Retard pulses arrive simultaneously, an illegal input condition.

The second set of flip-flops captures the rising edges of the Advance/Retard inputs. Subsequent rising edges are ignored for a complete cycle of F_{OUT} . This set of flip-flops holds the state of the selection switch for the next cycle of F_{OUT} . After every cycle of F_{OUT} , these flip-flops reset to the divide-by-20 state. The output of the third set of flip-flops controls the AND/OR switch logic, which selects the divide ratio for the current cycle. As the single log-

ic one shifts to the right, it first latches the selected switch position into the last set of flip-flops. This selection could be di-

(continued on pg 116)

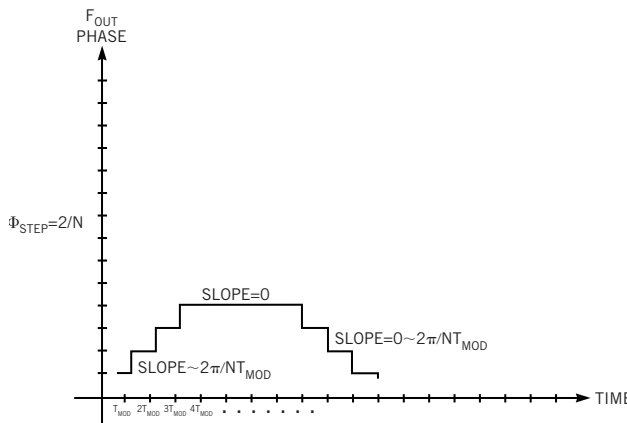


Figure 3 The stair-step portions of this graphic show where phase is added to or subtracted from F_{OUT} .

vide-by-19, -20, or -21, depending on the Advance/Retard inputs. The AND/OR gates now can select which tap point connects to the shift register's input for that cycle of F_{OUT} . After one more shift, the single logic one clears the second set of

flip-flops, thus returning to the divide-by-20 state.

A propagation delay is inherent in the Advance/Retard control inputs. If you apply an Advance/Retard pulse, it will not be applied to the selection of the tap

point until the next cycle of F_{OUT} . We programmed the modulator into an Altera EPM7128-10 (10-nsec) device, and used an input-clock frequency, F_{CLK} , of 20 MHz. When the divide-select switch is in the divide-by-20 state, it is easy to compute the frequency output; it's just the input clock divided by 20. For an N-bit system, it would be the input clock divided by N. But how do you derive a general formula for F_{OUT} if Advance/Retard pulses arrive at a rate of F_{MOD} ? Whenever an Advance or Return pulse is processed, a fixed amount of time is added or subtracted to the time between F_{OUT} pulses. You need an expression for the equivalent amount of added or subtracted phase—in other words, the phase step.

One complete cycle of F_{OUT} with no Advance/Retard modulation takes N/F_{CLK} seconds. This interval is just the period of the output with no modulation. If you add or subtract one clock period, how much phase does this represent with respect to $F_{OUT} = N/F_{CLK}$? In terms of the fraction of time that adding or subtracting one clock period from the nominal output cycle, you can write: Fraction of one output cycle per Advance/Retard pulse = (clock period)/(nominal-output period) = $(1/F_{CLK})(F_{CLK}/N)$ cycles = $1/N$ cycles.

Converting to radians and defining this step as the phase step or Φ_{STEP} , $\Phi_{STEP} = 2\pi/N$ radians. Therefore, every time an Advance or Retard pulse is processed, the phase of the output changes by $\pm 2\pi/N$ radians. You can now use the fact that frequency is the time derivative of phase to derive the formula for F_{OUT} . **Figure 3** shows a time plot of phase of the output F_{OUT} for the three possible commands: Advance, Static (divide-by-N), and Retard. In **Figure 3**, at first phase is added, no change occurs in phase, and then phase is subtracted. The frequency is not changed where the slope is zero, and is equal to F_{CLK}/N . At the stair-step portions of the plot, you can approximate the slope as $\pm 2\pi/NT_{MOD}$. You can interpret this figure as the change in the output frequency (in radians). To convert to cycles, divide by 2π , which gives the change in output frequency as $\pm F_{MOD}/N$ for Advance or Retard pulses arriving at a rate of F_{MOD} . □