Innovative RoboClock Application

Introduction

This application note presents a unique application of RoboClock, using its complex and precise waveform generation capability to implement PWM (Pulse Width Modulation) to enhance color images and increase the resolution of laser printers. The first section of this application note provides a brief description of RoboClock and presents three methods that the user can employ to configure it. Second, a brief background on image and resolution enhancement is presented. Finally, the required waveform to implement the image enhancement and the configuration of RoboClock is presented.

Overview of RoboClock

The CY7B991 and CY7B992, commonly known as RoboClock, are programmable skew clock buffers capable of generating thousands of various clocking combinations. As shown in *Figure 1*, the eight high drive outputs are arranged in four pairs, which can be configured by three-level inputs (HIGH, LOW, and MID logic level). The internal PLL is fully self-contained and does not require any external components to operate. The PLL buffer stages are differential, greatly enhancing the robustness of the PLL operation in terms of jitter over voltage, and temperature variations.

Basically, the PLL aligns the output clock in both phase and frequency with the reference clock. The simplest mode of operation is the low-skew output mode. In this mode the outputs are virtually skewless. The maximum skew is only a few hundred picoseconds. Please refer to the CY7B991/992 datasheet for various skew specifications. The second mode is the programmable skew mode. The outputs of RoboClock can be skewed (advanced and delayed) by increments of one time unit (t_U), which is 0.7 to 1.5 ns, determined by the operating frequency and range of the PLL.

$$t_U = 1/(F_{nom} * N) \quad \text{Eq. 1}$$

As shown in *Table 1*, the frequency range of the PLL is determined by the three-level FS input. For each frequency range, there is a corresponding integer "N" that can be used in *Equation 1* to calculate t_U. Up to $\pm 12 t_U$ skew can be achieved between the outputs of RoboClock (positive t_U represents delaying

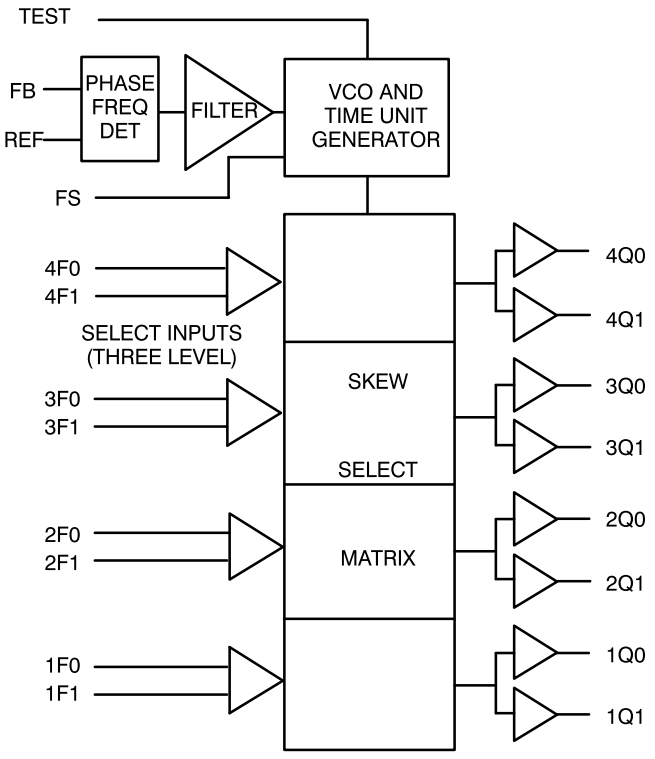

Figure 1. Logic Block Diagram

the output with respect to REF and negative t_U represents advancing the output with respect to REF).

The third mode of operation is the Multi-function mode. In this mode the outputs may be multiplied by 2 or 4, divided by 2 or 4, or inverted. Most importantly, the skew features can be combined with multiply, divide, and invert functions. This results in, literally, over 26,000 timing configurations. For more detailed information on the operation of the RoboClock, please refer to the following application notes "Using the CY7B991 with the 50-MHz 486 Cache Module and the 40-MHz R3000" and "Everything You Need to Know About CY7B991/CY7B992 (RoboClock) But Were Afraid to Ask." This application note is meant to complement the topics discussed in above mentioned application notes.

Table 1. Frequency Range Select and t_U Calculation

FS	f_{NOM} (MHz) Min.	f_{NOM} (MHz) Max.	$t_U = \frac{1}{f_{NOM} \times N}$ where N =	Approximate Frequency (MHz) At Which t_U = 1.0 ns
LOW	15	30	44	22.7
MID	25	50	26	38.5
HIGH	40	80	16	62.5

Usually, one of the outputs of RoboClock is used as the Feedback input. If the desired waveform is not directly generated by RoboClock, an imaginative user may run an output of RoboClock through a logic block, then send it back to the FB input of the RoboClock. Through this scheme, unlimited additional functions may be implemented by RoboClock. Note that in this case, all the other outputs of the RoboClock will be shifted by a period equal to the delay through the external logic block, because the PLL will align the FB input with the REF input, both in phase and frequency.

Cascading two RoboClocks in series will also dramatically increase the output possibilities. In this case, one of the outputs of the first stage will serve as the REF input for the second stage. Multiple feedback configurations are possible, which can result in an innovative set of outputs.

RoboClock Configuration Methodologies

Using One Small Table

The entire set of programmable skew configurations is summarized in a single small table shown in *Table 2*. Every possible combination can be driven from this small table. For example, if $+2t_U$ is required from 3Qx (3Q0 or 3Q1) outputs, based on *Table 2*, the corresponding 3Fx inputs should be set as 3F1= MID and 3F0=HIGH. Any one of 1Qx, 2Qx, or 4Qx outputs may be used as FB input, by leaving its corresponding 1Fx, 2Fx, or 4Fx inputs floating (i.e., 1F1= MID , 1F0= MID). Note that *Table 2* represents only the cases where the feedback is an output with no skew, divide, or invert function. Basically, a $0t_U$ output is used for FB input.

Table 2. Output Adjustment Configurations

Function Selects		Output Functions		
1F1, 2F1, 3F1, 4F1	1F0, 2F0, 3F0, 4F0	1Q0, 1Q1, 2Q0, 2Q1	3Q0, 3Q1	4Q0, 4Q1
LOW	LOW	$-4t_U$	Divide by 2	Divide by 2
LOW	MID	$-3t_U$	$-6t_U$	$-6t_U$
LOW	HIGH	$-2t_U$	$-4t_U$	$-4t_U$
MID	LOW	$-1t_U$	$-2t_U$	$-2t_U$
MID	MID	$0t_U$	$0t_U$	$0t_U$
MID	HIGH	$+1t_U$	$+2t_U$	$+2t_U$
HIGH	LOW	$+2t_U$	$+4t_U$	$+4t_U$
HIGH	MID	$+3t_U$	$+6t_U$	$+6t_U$
HIGH	HIGH	$+4t_U$	Divide by 4	Inverted

Now, to generate additional output functions, if the feedback output is programmed to skew, divide or invert, then output functions of other outputs may not be directly read from *Table 2*. In this case, to figure out the final output function observed on the output, simply subtract whatever the feedback term is programmed to, from the output function programmed on the corresponding output. Therefore, by using only *Table 2* and the following simple algorithm, every single combination of RoboClock can be figured out.

Final Output Function = Output Function − FB Function

If there is any ambiguity, the following example should clarify the use of this method. Let's say $+7t_U$

of delay is required. Obviously, $+7t_U$ is not a choice, available in *Table 2*. However, any two functions from two different outputs of *Table 2* may be combined to achieve a desired function. For this example, there are several solutions, and only one of them will be presented. One way to achieve $+7t_U$ is to subtract $-3t_U$ from $+4t_U$.

$+7t_U = +4t_U - (-3t_U)$

Therefore, if 1Qx output is programmed to have $-3t_U$ of skew (1F1=LOW, 1F0=MID), and used as the FB input, and if the 3Qx is programmed to have $+4t_U$ of skew (3F1=HIGH, 3F0=LOW), the final output function observed on 3Qx will be $+7t_U$.

One exception to this simple rule is that if a divided output is used as the FB input, then the other outputs will be multiplied by the same factor (2 or 4). The reason for this is that the PLL will force the FB to align with the REF both in phase and frequency. Therefore, if the FB term is programmed to divide by 2, the PLL will speed up twice to force the FB term to align with the REF frequency. As an example, if advance by $6t_U$ and multiply by 4 function is required ($-6t_U$ and f*4), then

*Final Function = $-6t_U$ − (divide by 4) => ($-6t_U$ and f*4)*

The solution for this example is to program 3Qx to divide by 4 (3F1=HIGH, 3F0=HIGH) and use it as FB, and program 4Qx to have $-6t_U$ of skew (4F1=LOW, 4F0=MID). The final function observed on 4Qx will be REF frequency multiplied by 4 and advanced by $6t_U$ ($-6t_U$ and f*4).

By this method, one can easily determine if a desired function can be implemented by RoboClock or not. RoboClock can generate a waveform composed of any two functions from two different outputs of *Table 2*.

Using Three Tables for Multiple Outputs

If multiple outputs with various functions are required, using the previous method could be a little cumbersome. All the possible combinations of RoboClock outputs are in three tables, illustrated in *Tables 3* through *5*. Each table represents all the possible output combinations with a given output connected to FB input. For example, once 3Qx output is used as FB input, then all the possible output combinations could be found in *Table 4*. These three tables are extremely valuable tools in determining what FB term to use and how to configure the RoboClock, when multiple outputs with various functions are required.

Once the required multiple functions are determined in terms of t_U, an effort should be made to locate one row in one of the three tables that contains the required functions. For example, if one of the desired functions is divide by 2 and delay $4t_U$ ($+4t_U$ and f/2), then by observation, that choice can be located in row 1 of *Table 3*, row 3 of *Table 4*, and row 3 of *Table 5*. Now, the one to be selected as a solution would depend on what the other required functions are, because once an output, which is programmed to perform a certain function, is selected as FB input, all the outputs of a RoboClock are limited to a single row found in *Tables 3* through *5*. If in the previous example, the second required function happens to be invert and skew by $4t_U$ ($+4t_U$ and INV), then the only solution is row 1 of *Table 3*. In this case 1Qx could be used as the FB input with its inputs hardwired to GND (1F1=LOW, 1F0=LOW), and 3F1=HIGH, 3F0=HIGH to generate ($+4t_U$ and f/2) function on 3Qx outputs, and set 4F1=HIGH, 4F0=HIGH to generate (+4tu and INV) function on 4Qx outputs. In this configuration, the 2Qx outputs could be programmed to have any one of $0t_U$ through $+8t_U$ skew. For example, if $+7t_U$ is another required output, then 2F1=High, 2F0=Mid will generate $+7t_U$ skew on 2Qx. Note that even though the 1Qx outputs are programmed to have $-4t_U$ skew, they are forced by PLL to align with the REF frequency, therefore 1Qx output could be used as a $0t_U$ output.

The Table method is recommended for multiple outputs with various function requirements. If the exact required outputs cannot all be found in one row, then the designer can use the three tables to understand the design choices that are available within the three tables. Based on the design requirements, the user can make a judgement on what outputs must exactly meet the required specs, and what outputs may be slightly compromised. If the required outputs are not found in one row of the three tables,

a. Standard unmodulated

b. Modulated laser beam with shorter "on" periods and variable-size dots.

Figure 2. Laser Images

and no compromise can be made on the requirements, then two or more RoboClocks may be used to meet the specific required outputs.

Using RoboClock in Resolution Enhancement of a Laser Printer

Background

Laser printers are no different from any other electrical systems, in that the higher the resolution or the accuracy of the system, the higher the complexity and cost of the system. It has been and will always be the goal of system and design engineers to achieve the highest performance and resolution at the lowest cost.

In the case of a laser printer, to achieve higher resolution than the nominal low-end 300 DPI (dot per inch), the throughput of the processor, size of the memory, and the glue logic should increase accordingly. In many cases, the additional hardware cost does not justify the enhancement in the resolution. A few years ago, a new technique called Resolution Enhancement Technology (RET) was developed by Hewlett Packard. The main advantage of this technique versus conventional resolution enhancement techniques is that the resolution enhancement is gained with hardly any increase in the throughput of the processor or memory size. Therefore, this approach is a very economical way of gaining enhanced resolution.

For obvious reasons, the entire laser printer industry is using some form of this technique. Various flavors of the same technique are being applied to different image-enhancement machines. The halftones or gray scales are common in most laser printers. The underlying technique is fairly simple. This is accomplished by modulating the laser beam, as opposed to the conventional "on" or "off" state of the laser beam, for the entire cycle time. The laser beam could be turned "on" or "off" 25%, 50%, 75% or 100% of the period. By this, large and small dots can be produced on a given image, therefore gaining much higher "perceived" resolution compared with images constructed by only one size dot. The varying size dots produce much smoother text files and generate much sharper images through shades of gray. Refer to *Figure 2a* and *2b*. The same idea is used in color image enhancement where much smoother and more pleasant images may be produced in a given color image by dilating black dots and shrinking the red. This filtering or color enhancement feature can be used to produce various special effects, or simply be employed to create more appealing color images. Please refer to *Figure 3a* and *3b*. To further clarify this technique, the true resolution, technically, still remains the same, but the images are "perceived" to be higher resolution. It does not matter how the "better image" was

a. Black dot dilated **b. Red dot shrunk**

Figure 3. Color Enhancements

created, as long as it looks good and the cost of the hardware is affordable.

Design Implementation

RoboClock is used to generate precise complex waveforms needed for laser beam modulation. The particular laser system discussed in this application note requires eight levels of modulation, which consists of 100% on, 75% on, 50% on, 25% on, 100% off, 75% off, 50% off, and 25% off. The eight waveforms are shown in *Figure 4*.

Note that all waveforms are synchronized to the rising edge of the system clock.

Analyzing the entire circuitry of the laser printer is beyond the scope of this application note, and only the waveform modulation section is discussed. The system clock runs at 66.67 MHz, which translates into 15 ns cycle time. The simplified diagram of the modulation section is shown in *Figure 5*. The modulation section consists of a RoboClock, a 256K*4 SRAM that contains the pixel information, four NOR gates with complemented outputs, and an 8:1 MUX. In this application, since the laser head interface uses ECL levels, 500 ps ECL NOR gate with complemented outputs (MC10E101) and ECL 8:1 MUX (MC10E163) is used. Unused inputs of the quad four input NOR gates are tied LOW. The TTL outputs of CY7B991 are translated into ECL levels by Cypress Semiconductor high-speed, low-skew TTL to ECL translator (CY10E384L). To keep the modulation logic diagram simple, the translate block is not shown in *Figure 5*. Note that, generally, 74AS logic parts are used to implement external logic functions, which requires no translate logic to interface with RoboClock.

The 66.67-MHz system clock is fed to the REF input of the RoboClock. RoboClock generates very precise waveforms and, with one level of gating, all the six modulated waveforms are produced and fed to the 8:1 MUX. For this design, RoboClock generates precise 90-degree phase-shifted, true and complemented versions of the 66.67-MHz REF input frequency. Note that only six waveforms are generated. The “100% on” and “100% off” modes are hardwired HIGH and LOW to the 8:1 MUX. Three bits of the SRAM are used to select one out of eight possible modulated signals. The output of very fast MUX is directly sent to the laser head. Therefore, all eight levels of modulated waveforms are present at the input of the MUX at all times. Only one is routed to the laser head, depending on the required modulation level stored in the SRAM.

Generally, one should be very cautious about using the output of a MUX, since during the period when MUX select bits are changing, the MUX output will usually be glitching, until the MUX select bits are stabilized. This behavior is due to the fact that all the select bits do not arrive at exactly the same time. Even if they did arrive at the same time, delay path variations and logic switching internal to MUX may create a glitch on the output. As a word of caution, the above mentioned scheme should not be used for a clocking scheme. When a new MUX input is se-

lected, there will probably be a glitch on the output. If the first cycle glitch can be tolerated or masked, then this scheme can be used for clock distribution. A delayed clocked version of the MUX output could be safely used for clock distribution. Obviously, the delay should be larger than the maximum propagation delay of MUX. For this particular application, the glitch is not as important, because the total duration of ON and OFF times of the laser beam is the concern, not the rising or falling edges of the waveform. Also, the laser head is turned off during the MUX address selection, totally masking any possible glitches.

Configuring RoboClock and Design Analysis

A close observation of waveforms shown in *Figure 5* reveals the fundamental idea behind generating all six modulated waveforms. Simply by gating the 90-degree phase-shifted REF with the true and complemented version of the REF clock, all six waveforms are generated. For simplicity's sake, lets call the 90-degree phase-shifted waveform $+4t_U$, 66.67-MHz clock F, and the inverted clock $\overline{F}$. Let's look at how each one is generated.

75% HIGH: $(+4t_U)$ OR (F)

50% HIGH: F

25% HIGH: $(+4t_U)$ NOR $(\overline{F})$

75% LOW: $(+4t_U)$ NOR (F)

50% LOW: $\overline{F}$

25% LOW: $(+4t_U)$ OR $(\overline{F})$

Note that very fast NOR gates with true and complemented outputs were selected to achieve uniform delay for all outputs. Also, the 50% HIGH and 50% LOW signals are routed OR gates configured as buffers to ensure matched delay signals. Please note that all three RoboClock outputs, $+4t_U$, F, and $\overline{F}$, have the same number of loads. It is very impor-

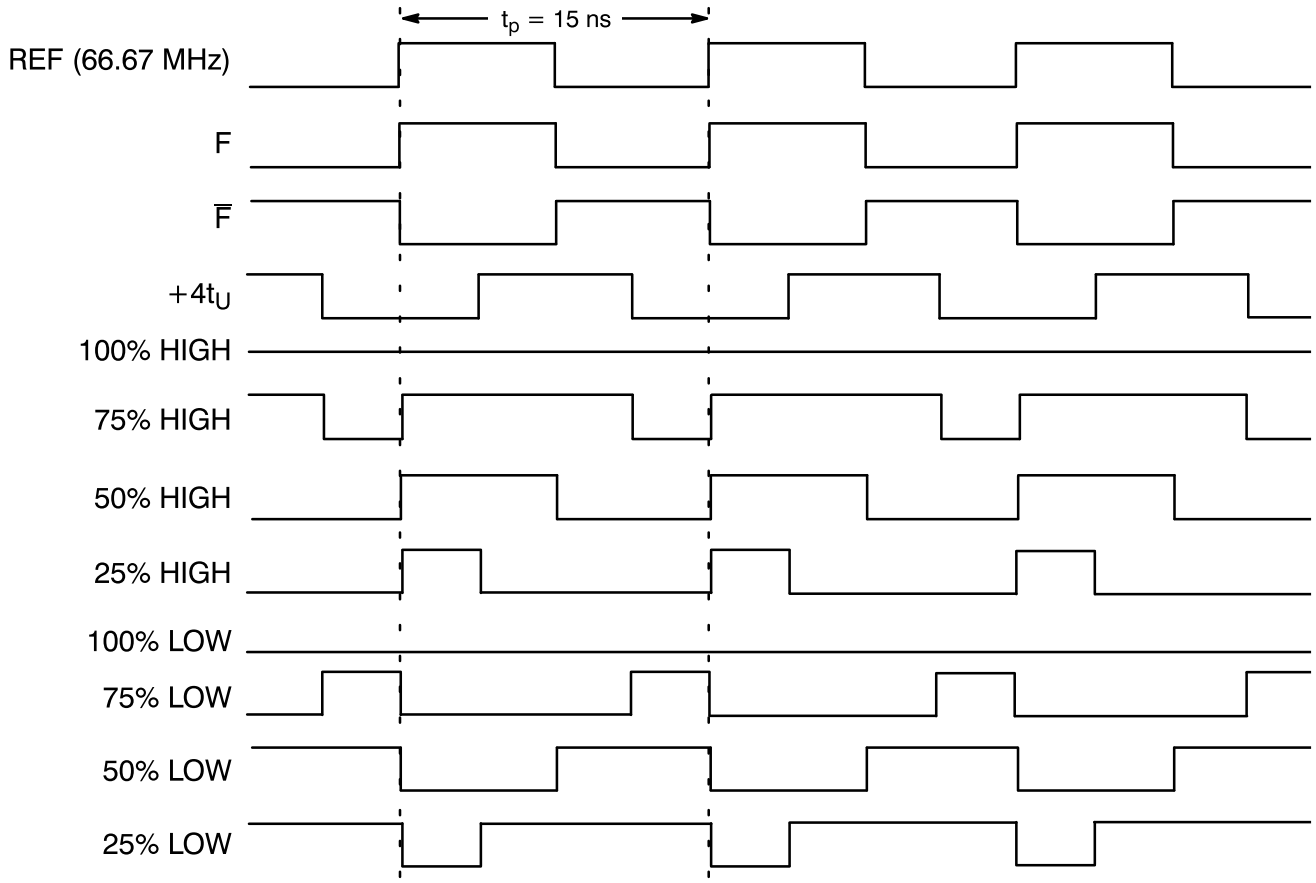

Figure 4. Generated Waveforms

tant during layout to match all the trace lengths from RoboClock to NOR gates and from the NOR gates to the MUX to prevent undesirable skew, which will translate into phase shift and pulsewidth variation on the laser beam.

Over voltage and temperature variation, all the outputs of the RoboClock are very stable. The PLL inside the RoboClock is constructed from differential stages, which makes it self-compensating against voltage and temperature variations. Consequently, RoboClock generates robust output waveforms in terms of phase and frequency. The external OR gates may distort the waveform due to the effects of voltage and temperature variation.

Earlier, the 90-degree phase-shifted waveform was equated with $+4t_U$. Let's see how that was derived. Based on *Table 1*, each time unit is calculated by the following equation:

$$t_U = 1/(F_{nom} * N) \quad \text{Eq. 1}$$

Where, as indicated in the *Table 1*, N can be any one of 44, 26, or 16 integer numbers, depending on the maximum output frequency of the RoboClock. Since the output frequency is 66.67 MHz, then FS is selected to be HIGH (frequency range of 40 to 80 MHz), N = 16, and F_{nom} = 66.67 MHz. By simply plugging the numbers in the *Equation 1*, the time unit or the t_U can be calculated as:

$t_U = 1/(66.67\ MHZ * 16)$

$t_U = 0.9375\ ns$

In terms of phase shift, if 66.67 MHz or 15-ns cycle time is 360 degrees, then the 90-degree phase-shift is essentially 15 ns divided by 4.

90 Degree Phase Shift = 15 ns / 4 = 3.75 ns

Therefore, the number of time units to shift to obtain 90-degree phase-shift, is simply derived by dividing 3.75 ns by t_U.

Number of Time Units = 3.75 ns / 0.9375 = 4

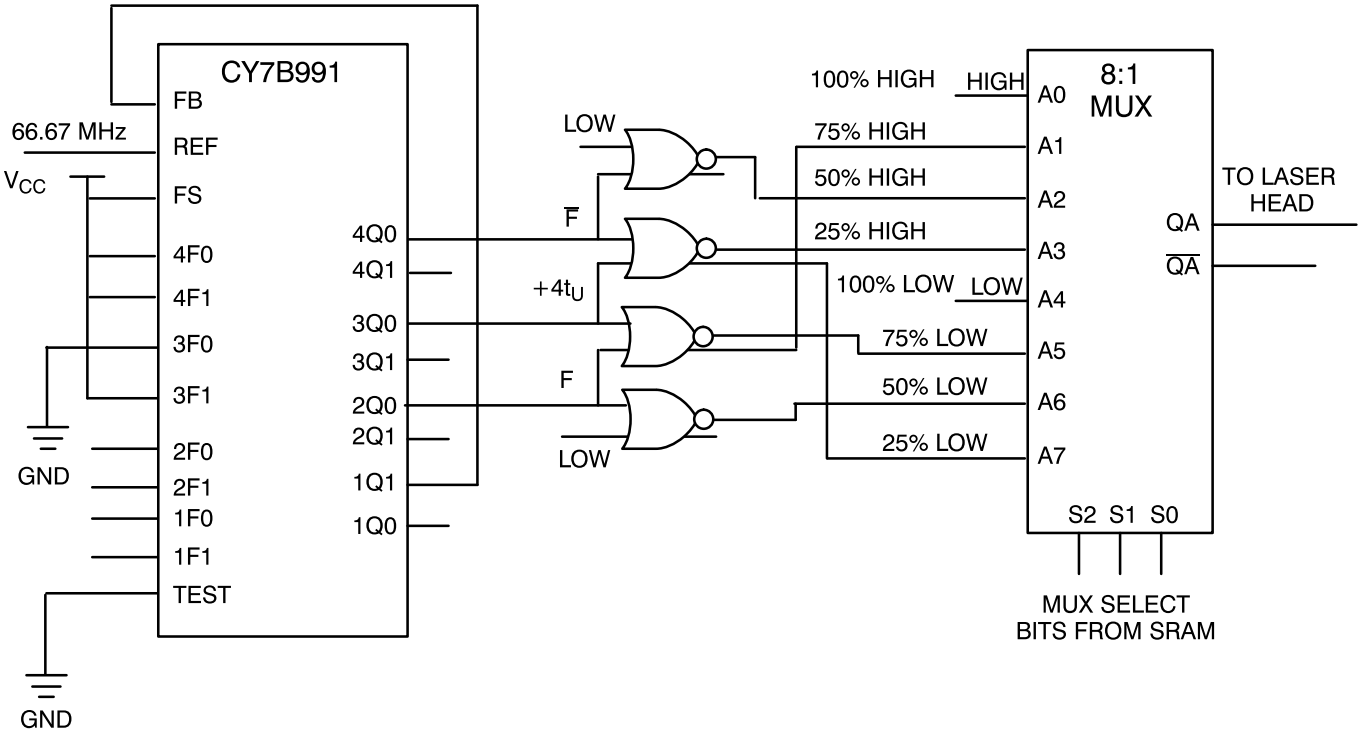

Figure 5. Simplified Laser Modulation Diagram

Therefore, in FS = HIGH mode, 4 t_U translates into 90-degree phase-shift. An observant reader might have already noticed the fact that in FS = HIGH mode, N is equal to 16, based on *Table 1*. This means that, in FS = HIGH mode, an entire cycle or 360 degrees, is equivalent to the delay through 16 stages of ring oscillator, and each stage represents one t_U (In FS = LOW the number of delay stages or N is 44 and in FS = MID it is 26. As shown in *Figure 6*, note that the actual number of ring oscillator buffer stages is half the N, because each cycle contains a LOW and HIGH period, which means to complete a full cycle the signal propagates through the ring oscillator twice.) In order to derive a 90-degree phase-shift, all one needs to do is to multiply N by 1/4 (where 90/360 = 1/4 cycle). Therefore, 16/4 = 4 time units, in FS = HIGH mode, represents a 90 degree phase shift.

The same simple methodology can be used to figure out the number of time units of delay or advance to implement a n arbitrary degree of phase shift. The number of units of skew (T_U needed for an arbitrary phase shift is calculated as follows:

$$\#\,T_U = \frac{N - phaseshift}{360} \qquad \text{Eq. 2}$$

Rounding this number to the nearest integer will introduce a small phase error from the desired phase shift. For example, if 60 degree phase shift is required when FS = LOW, then:

Required Phase Shift = 60/360 = 1/6 cycle

*Number of Time Units = N * 1/6 = 44/6 = 7.33* t_U

Since the number of PLL stages for each FS mode is an integer number, then the nearest time unit shift, in this case, will be seven. Obviously, this will create a phase error of 0.33 t_U.

Let's go back and discuss how the +4t_U, F and $\overline{F}$ waveforms are generated by RoboClock. Since multiple outputs from a single RoboClock is expected, as an exercise, let's use the three-table method. There are several solutions for the current requirement; only one of the simplest is presented. By observing *Table 3*, titled as "1Qx/2Qx Output Connected to FB Input," one may select the 1Q0 to be used as FB, and leave the corresponding inputs floating (1F0=1F1=MID). Thus, essentially, we have ac-

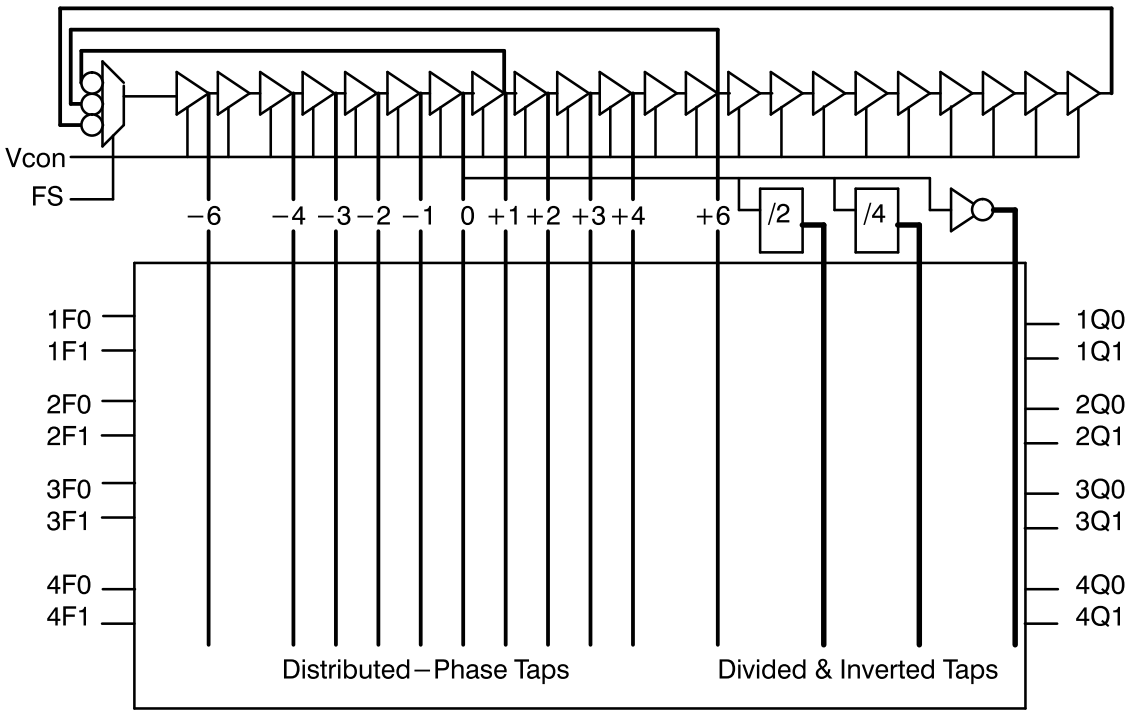

Figure 6. Distributed-Phase Clock Oscillator and Output Adjust Matrix

cess to all the terms available in row two of the given table. Now, by selecting the inputs, the RoboClock may be configured to generate various waveforms. By setting 2F0=2F1=MID or floating the 2Fx inputs, the 2Q0 and 2Q1 will generate the required F signal (also, 1Qx could have been used for F signal). Setting 3F0=LOW and 3F1=HIGH will generate $+4t_U$ signal at 3Q0 and 3Q1. Finally, setting 4F0=4F1=HIGH will generate the $\overline{F}$ signal. As mentioned earlier, by fixing the feedback term, in this case, all the elements of the 2nd row of *Table 3* are available for the user. RoboClock is a flexible clock distribution buffer that may be reconfigured easily during the prototyping phase of a design. For example, if instead of generating a 0 T_U output on 2Qx, it is required to have the 2Qx signals advanced by $2t_U$, then this can be accomplished simply by setting 2F0=HIGH and 2F1=LOW. This is one of the commonly used features of RoboClock that offers thousands of variations for prototyping purposes. Often, during prototyping phase, some modification in the clock or the waveform is required.

RoboClock, with its thousands of configurations, resolves some of the unexpected timing problems. In fact, during prototyping, if multiple timing variations are expected, it is advised to use a three-state register to drive the RoboClock inputs. Then, each output of the register must have a 10K pull-up and 10K pull-down resistor, to ensure that MID level is held at half the supply voltage when the register is three-stated. In this case, the user may write a word in the input register, and by doing so, reconfigure the entire operation of the RoboClock, without using any jumpers. Note that not all the inputs need to be reconfigurable for a given design. Often, a couple of reconfigurable signals are all that is needed. In that case, most inputs may be hardwired and the inputs needed to reconfigure various outputs may be registered with the 10K pull-up and pull-down resistors.

Summary

RoboClock was used to generate very precise complex waveforms to enhance color images and increase the resolution of laser printers. Even though RoboClock is widely used for clock distribution, this application note presented an alternative use of RoboClock for complex precise waveform generation.

Table 3. 1Qx or 2Qx Output Connected to FB Input (Part 1)

Feedback Section		2Qx Output Section									
1Qx(2Qx)→FB		Configuration Block	2Qx(1Qx) Outputs with respect to REF								
			2F1 (1F1)	L	L	L	M	M	M	H	H	H
1F1 (2F1)	1F0 (2F0)		2F0 (1F0)	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		0t	+1t	+2t	+3t	+4t	+5t	+6t	+7t	+8t
L	M			−1t	0t	+1t	+2t	+3t	+4t	+5t	+6t	+7t
L	H			−2t	−1t	0t	+1t	+2t	+3t	+4t	+5t	+6t
M	L			−3t	−2t	−1t	0t	+1t	+2t	+3t	+4t	+5t
M	M			−4t	−3t	−2t	−1t	0t	+1t	+2t	+3t	+4t
M	H			−5t	−4t	−3t	−2t	−1t	0t	+1t	+2t	+3t
H	L			−6t	−5t	−4t	−3t	−2t	−1t	0t	+1t	+2t
H	M			−7t	−6t	−5t	−4t	−3t	−2t	−1t	0t	+1t
H	H			−8t	−7t	−6t	−5t	−4t	−3t	−2t	−1t	0t

Table 3. 1Qx or 2Qx Output Connected to FB Input (Part 2)

Feedback Section		3Qx Output Section									
1Qx(2Qx)→FB		Configuration Block	3Qx Output with respect to REF								
			3F1	L	L	L	M	M	M	H	H	H
1F1 (2F1)	1F0 (2F0)		3F0	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		+4t, f/2	−2t	0t	+2t	+4t	+6t	+8t	+10t	+4t, f/4
L	M			+3t, f/2	−3t	−1t	+1t	+3t	+5t	+7t	+9t	+3t, f/4
L	H			+2t, f/2	−4t	−2t	0t	+2t	+4t	+6t	+8t	+2t, f/4
M	L			+1t, f/2	−5t	−3t	−1t	+1t	+3t	+5t	+7t	+1t, f/4
M	M			0t, f/2	−6t	−4t	−2t	0t	+2t	+4t	+6t	0t, f/4
M	H			−1t, f/2	−7t	−5t	−3t	−1t	+1t	+3t	+5t	−1t, f/4
H	L			−2t, f/2	−8t	−6t	−4t	−2t	0t	+2t	+4t	−2t, f/4
H	M			−3t, f/2	−9t	−7t	−5t	−3t	−1t	+1t	+3t	−3t, f/4
H	H			−4t, f/2	−10t	−8t	−6t	−4t	−2t	0t	+2t	−4t, f/4

Table 3. 1Qx or 2Qx Output Connected to FB Input (Part 3)

Feedback Section		4Qx Output Section									
1Qx(2Qx)→FB		Configuration Block	4Qx Output with respect to REF								
			4F1	L	L	L	M	M	M	H	H	H
1F1 (2F1)	1F0 (2F0)		4F0	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		+4t, f/2	−2t	0t	+2t	+4t	+6t	+8t	+10t	+4t, INV
L	M			+3t, f/2	−3t	−1t	+1t	+3t	+5t	+7t	+9t	+3t, INV
L	H			+2t, f/2	−4t	−2t	0t	+2t	+4t	+6t	+8t	+2t, INV
M	L			+1t, f/2	−5t	−3t	−1t	+1t	+3t	+5t	+7t	+1t, INV
M	M			0t, f/2	−6t	−4t	−2t	0t	+2t	+4t	+6t	0t, INV
M	H			−1t, f/2	−7t	−5t	−3t	−1t	+1t	+3t	+5t	−1t, INV
H	L			−2t, f/2	−8t	−6t	−4t	−2t	0t	+2t	+4t	−2t, INV
H	M			−3t, f/2	−9t	−7t	−5t	−3t	−1t	+1t	+3t	−3t, INV
H	H			−4t, f/2	−10t	−8t	−6t	−4t	−2t	0t	+2t	−4t, INV

Table 4. 3Qx Output Connected to FB Input (Part 1)

Feedback Section		1Qx, 2Qx Output Section									
3Qx ⧫ FB		Configuration Block	1Qx (2Qx) Output Delay to REF								
			1F1, (2F1)	L	L	L	M	M	M	H	H	H
3F1	3F0		1F0, (2F0)	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		−4t, f*2	−3t, f*2	−2t, f*2	−1t, f*2	0t, f*2	+1t, f*2	+2t, f*2	+3t, f*2	+4t, f*2
L	M			+2t	+3t	+4t	+5t	+6t	+7t	+8t	+9t	+10t
L	H			0t	+1t	+2t	+3t	+4t	+5t	+6t	+7t	+8t
M	L			−2t	−1t	0t	+1t	+2t	+3t	+4t	+5t	+6t
M	M			−4t	−3t	−2t	−1t	0t	+1t	+2t	+3t	+4t
M	H			−6t	−5t	−4t	−3t	−2t	−1t	0t	+1t	+2t
H	L			−8t	−7t	−6t	−5t	−4t	−3t	−2t	−1t	0t
H	M			−10t	−9t	−8t	−7t	−6t	−5t	−4t	−3t	−2t
H	H			−4t, f*4	−3t, f*4	−2t, f*4	−1t, f*4	0t, f*4	+1t, f*4	+2t, f*4	+3t, f*4	+4t, f*4

Table 4. 3Qx Output Connected to FB Input (Part 2)

Feedback Section		3Qx Output Section									
3Qx ⧫ FB		Configuration Block	4Qx Output with Delay to REF								
			4F1	L	L	L	M	M	M	H	H	H
3F1	3F0		4F0	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		0t	−6t, f*2	−4t, f*2	−2t, f*2	0t, f*2	+2t, f*2	+4t, f*2	+6t, f*2	INV, f*2
L	M			+6t, f/2	0t	+2t	+4t	+6t	+8t	+10t	+12t	+6t, INV
L	H			+4t, f/2	−2t	0t	+2t	+4t	+6t	+8t	+10t	+4t, INV
M	L			+2t, f/2	−4t	−2t	0t	+2t	+4t	+6t	+8t	+2t, INV
M	M			0t, f/2	−6t	−4t	−2t	0t	+2t	+4t	+6t	0t, INV
M	H			−2t, f/2	−8t	−6t	−4t	−2t	0t	+2t	+4t	−2t, INV
H	L			−4t, f/2	−10t	−8t	−6t	−4t	−2t	0t	+2t	−4t, INV
H	M			−6t, f/2	−12t	−10t	−8t	−6t	−4t	−2t	0t	−6t, INV
H	H			0t, f*2	−6t, f*4	−4t, f*4	−2t, f*4	0t, f*4	+2t, f*4	+4t, f*4	+6t, f*4	INV, f*4

Table 5. 4Qx Output Connected to FB Input (Part 1)

Feedback Section		1Qx, 2Qx Output Section									
4Qx ⧫ FB		Configuration Block	1Qx, 2Qx Output with respect to REF								
			1F1, 2F1	L	L	L	M	M	M	H	H	H
4F1	4F0		1F0, 2F0	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		−4t, f*2	−3t, f*2	−2t, f*2	−1t, f*2	0t, f*2	+1t, f*2	+2t, f*2	+3t, f*2	+4t, f*2
L	M			+2t	+3t	+4t	+5t	+6t	+7t	+8t	+9t	+10t
L	H			0t	+1t	+2t	+3t	+4t	+5t	+6t	+7t	+8t
M	L			−2t	−1t	0t	+1t	+2t	+3t	+4t	+5t	+6t
M	M			−4t	−3t	−2t	−1t	0t	+1t	+2t	+3t	+4t
M	H			−6t	−5t	−4t	−3t	−2t	−1t	0t	+1t	+2t
H	L			−8t	−7t	−6t	−5t	−4t	−3t	−2t	−1t	0t
H	M			−10t	−9t	−8t	−7t	−6t	−5t	−4t	−3t	−2t
H	H			−4t, INV	−3t, INV	−2t, INV	−1t, INV	0t, INV	+1t, INV	+2t, INV	+3t, INV	+4t, INV

Table 5. 4Qx Output Connected to FB Input (Part 2)

Feedback Section		1Qx, 2Qx Output Section									
4Qx ⧫ FB		Configuration Block	4Qx Output with respect to REF								
			3F1	L	L	L	M	M	M	H	H	H
4F1	4F0		3F0	L	M	H	L	M	H	L	M	H
L	L	Output Selection Block		0t	−6t, f*2	−4t, f*2	−2t, f*2	0t, f*2	+2t, f*2	+4t, f*2	+6t, f*2	0t, f/2
L	M			+6t, f/2	0t	+2t	+4t	+6t	+8t	+10t	+12t	+6t, f/4
L	H			+4t, f/2	−2t	0t	+2t	+4t	+6t	+8t	+10t	+4t, f/4
M	L			+2t, f/2	−4t	−2t	0t	+2t	+4t	+6t	+8t	+2t, f/4
M	M			0t, f/2	−6t	−4t	−2t	0t	+2t	+4t	+6t	0t, f/4
M	H			−2t, f/2	−8t	−6t	−4t	−2t	0t	+2t	+4t	−2t, f/4
H	L			−4t, f/2	−10t	−8t	−6t	−4t	−2t	0t	+2t	−4t, f/4
H	M			−6t, f/2	−12t	−10t	−8t	−6t	−4t	−2t	0t	−6t, f/4
H	H			INV, f/2	−6t, INV	−4t, INV	−2t, INV	0t, INV	+2t, INV	+4t, INV	+6t, INV	INV, f/4

Document #: LOG0002-0494