A Systematic Design Procedure for Asynchronous Counters Using El Naga's Transitions Technique

Nagi M. El Naga,
California State University,
Northridge

Halima M. El Naga
California State Polytechnic University,
Pomona

Abstract

Although asynchronous counters have been discussed in almost every logical circuits design textbook, they all failed to provide the readers with a systematic design procedure. This is due to the fact that the conventional method of designing sequential circuits lacks the means to provide such procedure. In this paper, a systematic procedure for designing asynchronous counters based on the use of El Naga's Transitions technique is presented. This technique is based on the use of the four transitions: α , the transition from 0 to 1, β , the transition from 1 to 0, I , the transition from 1 to 1, and ϕ , the transition from 0 to 0. This technique also provides the designer of logical sequential circuits with various testing algorithms that check the correctness of almost every step in the design procedure. If the provided testing algorithms are followed after each step of the design, the final design will be almost error free.

The design techniques presented in this paper make the process of designing asynchronous counters much simpler and, more importantly, make the process of teaching them much easier.

I. Introduction

In this section, the four transitions used in El Naga's Transitions technique [1] are first presented. Then, the excitation equation of each data input of each type of flip-flop is driven in terms of these four transitions.

During the transition from one state to another, a flip-flop can go through one of four possible transitions, which are defined as:

1. α , the transition from 0 to 1,
2. β , the transition from 1 to 0,
3. I , the transition from 1 to 1, and
4. ϕ , the transition from 0 to 0.

The transition methodology is based on the use of these four transitions. For a particular data input of a specific type of flip-flop, any of the above transitions could be either an

essential transition, *don't care* transition, or a *zero* transition. These are explained in the following:

1. **Essential Transition:** A transition is defined as an *essential* transition for a specific data input of a flip-flop if it is necessary to apply a logical 1 to that input to see the flip-flop go through this transition.
2. **Don't care Transition:** A transition is defined as a *don't care* transition for a specific data input of a flip-flop if it doesn't matter whether to a logical 1 or a logical 0 is applied to that input to see the flip-flop go through this transition.
3. **Zero Transition:** A transition is defined as a *zero* transition for a specific data input of a flip-flop if it is necessary to apply a logical 0 to that input to see the flip-flop go through this transition.

For example, to see an RS flip-flop go through α transition, 0 to 1, it is necessary to apply a logical 1 to the S input and a logical 0 to the R input. Therefore, the α transition is considered an *essential* transition for the S input and a *zero* transition for the R input.

The excitation equation of a data input of a flip-flop consists of two parts separated by a "+" sign. The first part represents a list of the *essential* transitions of this input. The second part starts with "D.C.", an abbreviation of "don't care", followed by a list of the *don't care* transitions of this input included between brackets. The general don't care terms are considered as part of the *don't care* transitions list and are represented by "X." Therefore, the *essential* transitions and the *don't care* transitions are included in the excitation equation while the *zero* transitions are not included at all. According to this definition, the excitation equations of all data inputs of all types of flip-flops are listed below:

1. RS flip-flop:

$$S = \alpha + \text{D.C. } [I, X] \dots\dots\dots(1)$$

$$R = \beta + \text{D.C. } [\phi, X] \dots\dots\dots(2)$$

2. T flip-flop:

$$T = \alpha, \beta + \text{D.C. } [X] \dots\dots\dots(3)$$

3. JK flip-flop:

$$J = \alpha + \text{D.C. } [\beta, I, X] \dots\dots\dots(4)$$

$$K = \beta + \text{D.C. } [\alpha, \phi, X] \dots\dots\dots(5)$$

4. D flip-flop:

$$D = \alpha, I + \text{D.C. } [X] \dots\dots\dots(6)$$

For example, in Equation 4, for the J input, α is an *essential* transition, β and I are *don't care* transitions. ϕ is the only *zero* transition, and, as such is not included in the excitation equation of the input.

II. El Naga's Transitions Systematic Methodology for the Design of Asynchronous Sequential Circuits

In this section, the systematic procedure to design an asynchronous counter using El Naga's Transition technique is presented.

In synchronous counters, all flip-flops of the counter are clocked from the main clock. Accordingly, each flip-flop will be clocked in every state of the counter. On the other hand, in asynchronous counters some flip-flops are clocked from the outputs of the other flip-flops in the counter. When a flip-flop is clocked from the output of another flip-flop in the counter and not from the main clock input, it will not be clocked in every state of the counter. As a result, the inputs to this flip-flop in all the states during which it will not be clocked are considered don't care. This reduces the amount of hardware required for the flip-flop.

Asynchronous counters are slower than synchronous counters. However, asynchronous counters are used simply because they require less hardware and are cheaper to implement.

The hardest step in designing asynchronous counters is identifying which flip-flops can be clocked from the outputs of other flip-flops. In some counters, none of the flip-flops can be clocked from the output of any of the other flip-flops in the counter. In this case, these counters have to be designed synchronously. Therefore, not every counter can be designed asynchronously.

Figure 1. Edge-triggered Flip-Flops

Edge triggered flip-flops, which are used in asynchronous counters, require just an edge for clocking. As shown in Figure 1, there are two types of edge triggered flip-flops. Leading-edge (positive edge) triggered flip-flops which require a zero to one transition, the α transition, for clocking and trailing-edge (negative edge) triggered flip-flops which require a one to zero transition, the β transition, for clocking. Either type can be used.

To demonstrate the asynchronous counter design methodology, the design of 4-bit asynchronous binary up counter is considered. Starting the design from the state table shown in Figure 2, the design steps are as follows:

- Step 1:** Identify each of the present states (incount states) using a minterm.
Step 2: Determine the general don't care terms (out of count states). In this example, there are no don't care terms since all states are incount states.
Step 3: Develop a transition column for each flip-flop (A, B, C, D) used in the design. Only one transition column is needed for each flip-flop regardless of the type of flip-flop to be used in the design. These columns are filled with the transitions (α, β, I, ϕ), which each flip-flop goes through in moving from the present states to the next states.

The results of Step 1 and Step 3 are shown in Figure 2.

	Present State				Next State				A	B	C	D
	A	B	C	D	A	B	C	D				
P ₀	0	0	0	0	0	0	0	1	ϕ	ϕ	ϕ	α
P ₁	0	0	0	1	0	0	1	0	ϕ	ϕ	α	β
P ₂	0	0	1	0	0	0	1	1	ϕ	ϕ	I	α
P ₃	0	0	1	1	0	1	0	0	ϕ	α	β	β
P ₄	0	1	0	0	0	1	0	1	ϕ	I	ϕ	α
P ₅	0	1	0	1	0	1	1	0	ϕ	I	α	β
P ₆	0	1	1	0	0	1	1	1	ϕ	I	I	α
P ₇	0	1	1	1	1	0	0	0	α	β	β	β
P ₈	1	0	0	0	1	0	0	1	I	ϕ	ϕ	α
P ₉	1	0	0	1	1	0	1	0	I	ϕ	α	β
P ₁₀	1	0	1	0	1	0	1	1	I	ϕ	I	α
P ₁₁	1	0	1	1	1	1	0	0	I	α	β	β
P ₁₂	1	1	0	0	1	1	0	1	I	I	ϕ	α
P ₁₃	1	1	0	1	1	1	1	0	I	I	α	β
P ₁₄	1	1	1	0	1	1	1	1	I	I	I	α
P ₁₅	1	1	1	1	0	0	0	0	β	β	β	β

Figure 2. Sequencer State and Transition Tables

- Step 4:** Construct a Karnaugh map for each flip-flop. Each map is filled with the transitions (α, β, I, ϕ) from the corresponding flip-flop transition column. The results of this step are shown in Figure 3.

Figure 3. Maps for the Sequencer

It should be noticed that all the work done in the previous four steps depends only on the sequencer, which is defined by the state table, and has nothing to do with the type of flip-flop that will be selected for the design. Therefore, the change of the type of flip-flop at any time later on will not require the repetition of any of the above steps, which is one of the great advantages of this method.

Since no output change occurs during the ϕ or I transitions, a flip-flop needs to be clocked only during the states that have either α or β transitions. A flip-flop, Y, can be clocked from the output of a flip-flop, Z, if the α transitions alone or β transitions alone in the map of flip-flop Z cover all the states that have α or β transitions in the map of flip-flop Y. Further if the covering transition of flip-flop Z is the same as the transition required to clock flip-flop Y, flip-flop Y is clocked from the *one (true)* output of flip-flop Z. If the covering transition and the transition required for clocking are different, flip-flop Y is clocked from the *zero (inverted)* output of flip-flop Z.

Step 5: In this step, which flip-flops must be clocked from the main clock and which flip-flops can be clocked from the outputs of other flip-flop are identified. Start by counting the number of α and β transitions in the Karnaugh map of each flip-flop. The flip-flop whose map has the largest number of α and β transitions, flip-flop D in this example, has to be clocked from the main clock input and its map stays unchanged. Next, we check to see if the output of flip-flop D could be used to clock the flip-flop whose map has the next largest number of α and β transitions, flip-flop C in this example. Since the β transitions alone in the map of flip-flop D cover all the states that have α or β transitions in the map of flip-flop C, flip-flop D is used to clock flip-flop C. Now, since flip-flop C will be clocked only in the states where the map of flip-flop D has β transitions. Then the ϕ and I transitions in the other states in the map of flip-flop C are replaced by don't cares. This is repeated for flip-flops A and B. Also, since the β transitions alone in the map of flip-flop C cover all the states that have α or β transitions in the map of flip-flop B, then flip-flop C is used to clock flip-flop B. Also, the β transitions alone in the map of flip-flop B cover all the states that have α or β transitions in the map of flip-flop A, then flip-flop B is used to clock

flip-flop A. Then all the transitions in the states in which the flip-flops are not clocked are replaced by don't cares. The modified Karnaugh maps are shown in Figure 4.

Figure 4. Modified Maps for the Sequencer

Step 6: Decide on the type of flip-flops to be used in the design. In this example, leading edge triggered JK flip-flops are selected.

Step 7: From the maps developed in Step 5 and shown in Figure 4, derive the optimum input equations for each flip-flop.

The process of optimizing a function using a map filled with transitions is very similar to that of a map filled with binary (traditional) values (0, 1, x). In the case of a map filled with transitions, the *essential* transitions of the flip-flop input under consideration are treated the same way as 1's are treated in a map filled with binary (traditional) values. At the same time, the *don't care* transitions of the flip-flop input under consideration, together with the general don't care terms, are treated the same way as x's are treated in a map filled with binary values. *i.e.* the adjacent *essential* transition entries should be combined together, with the help of the *don't care* transitions and the general don't care terms, in the minimum number of groups. Each group should be as large as possible the same way as 1's are grouped in the traditional map. The way these groups are read is also the same.

The following set of flip-flop input functions are driven by applying this procedure to the four maps in Figure 4.

$$\begin{aligned}
 j_a &= 1 & \dots\dots\dots(7) & & k_a &= 1 & \dots\dots\dots(8) \\
 j_b &= 1 & \dots\dots\dots(9) & & k_b &= 1 & \dots\dots\dots(10) \\
 j_c &= 1 & \dots\dots\dots(11) & & k_c &= 1 & \dots\dots\dots(12) \\
 j_d &= 1 & \dots\dots\dots(13) & & k_d &= 1 & \dots\dots\dots(14)
 \end{aligned}$$

Step 8: Using the above set of equations, draw the schematic diagram. In this schematic, flip-flops A, B and C are clocked from the **zero (inverted)** outputs of flip-flops B, C and D respectively since the leading-edge triggered flip-flops used in the

design need an α transition for clocking while the β transition was used for covering.

Figure 5. 4-bit Asynchronous Up-Binary Counter

III. El Naga’s Transitions Systematic Methodology for the Design of Asynchronous Sequential Circuits Applied to a General Example

In this section, the design methodology introduced in the previous section is used to design a general asynchronous counter. The counter is given by the state table shown in Figure 6. The results of Step 1 and Step 3 are also shown in Figure 6.

	Present State				Next State				A	B	C	D
	A	B	C	D	A	B	C	D				
P ₀	0	0	0	0	0	0	0	1	φ	φ	φ	α
P ₁	0	0	0	1	0	0	1	1	φ	φ	α	I
P ₃	0	0	1	1	0	1	0	1	φ	α	β	I
P ₅	0	1	0	1	0	1	1	1	φ	I	α	I
P ₇	0	1	1	1	1	0	0	0	α	β	β	β
P ₈	1	0	0	0	1	0	0	1	I	φ	φ	α
P ₉	1	0	0	1	1	0	1	1	I	φ	α	I
P ₁₁	1	0	1	1	1	1	0	1	I	α	β	I
P ₁₃	1	1	0	1	1	1	1	1	I	I	α	I
P ₁₅	1	1	1	1	0	0	0	0	β	β	β	β

Figure 6. Sequencer State and Transition Tables

The general don’t care terms of Step 2 are given by the following equation:

$$\text{General Don't Care Terms} = P_2 + P_4 + P_6 + P_{10} + P_{12} + P_{14} \quad \dots\dots\dots (15)$$

The Karnaugh maps constructed in step 4 are shown in Figure 7 below.

Figure 7. Maps for the Sequencer

The results of applying Step 5 to this counter are summarized in the following:

1. Both flip-flops C and D are clocked from the main clock input.
2. Flip-flop B is clocked from the output of flip-flop C.
3. Flip-flop A is clocked from the output of flip-flop D.
4. The modified Karnaugh maps are shown in Figure 8.

Figure 8. Modified Maps for the Sequencer

Selecting leading-edge triggered JK flip-flops for this counter, the following set of flip-flop input functions will be driven:

$$\begin{array}{ll} j_a = 1 & \dots\dots\dots(16) \\ j_b = 1 & \dots\dots\dots(18) \\ j_c = D & \dots\dots\dots(20) \\ j_d = 1 & \dots\dots\dots(22) \end{array} \quad \begin{array}{ll} k_a = 1 & \dots\dots\dots(17) \\ k_b = 1 & \dots\dots\dots(19) \\ k_c = 1 & \dots\dots\dots(21) \\ k_d = BC & \dots\dots\dots(23) \end{array}$$

The result of applying Step 8 is given by the schematic diagram shown in Figure 9.

Figure 9. Asynchronous Counter Schematic

IV. Transition Method's Design Verification Rules

The Transition method provides a number of rules that can be used to verify the correctness of the work done in the major steps of the design. After constructing the transition table, Step 3, the following rules can be applied to check the correctness of the information tables in case of a **closed loop** counter or sequencer:

1. For every α transition there must be a β transition. Therefore, in every transition table, the number of α transitions and number of β transitions should be the same. If they are different, this means there is an error that needs to be corrected.
2. Since a flip-flop cannot go through an α transition twice without going through a β transition in between, then in each transition table, α and β transitions should alternate.
3. After going through an α transition, the flip-flop will contain " I ". Therefore, in each transition table, α transitions can only be followed by β or I transitions. Similarly, β transitions can only be followed by α or ϕ transitions.

After constructing and filling a Karnaugh map with the transitions, it is possible to check if any of these transitions is misplaced. For example, for the Karnaugh map of flip-flop B, input B divides the map into two halves. One half represents B and the other half represents the complement of B ($\bar{B}$). In all the states in the half that represents B, B has a value 1. During the transition from any of these states, this value of 1 can stay 1, I transition, or change to 0, β transition. Therefore, in this half of the map, only β and I transitions and X should be found. If an α or ϕ transition is found in this half, this would be an indication that it is misplaced. Similarly, in the second half of the map, only α and ϕ transitions and X should be found. If β or I transition is found in this half, this would be an indication that it is misplaced.

If any of the above rules fails, it means there is an error that needs to be corrected.

V. Conclusion

In this paper, a systematic procedure to design asynchronous counters based on the use of El Naga's Transitions technique is presented. This technique is based on the use of the four transitions: α , the transition from 0 to 1, β , the transition from 1 to 0, I , the transition from 1 to 1, and ϕ , the transition from 0 to 0. This technique also provides the designer of logical sequential circuits with various testing algorithms that check the correctness of almost every step in the design procedure. If the provided testing algorithms are followed after each step of the design, the final design will almost be error free. The technique presented in this paper is considered the only documented technique that provides a systematic procedure for the design of asynchronous sequencers.

Bibliography

1. El Naga, Nagi M. and Halima M. El Naga. "El Naga's Transitions Technique and its Advantages Compared to the Conventional Method of Designing Sequential Circuits," Proc. 2000 ASEE Annual Conference & Exposition, **June 18-21, 2000**, St. Louis, MO.
2. El Naga, Nagi M. Lecture Notes, Electrical and Computer Engineering Department, California State University, Northridge, California, 1999.
3. John F. Wakerly, Digital Design Principles & Practices, Third Edition, Prentice Hall, 2000.
4. Charles H. Roth, Jr., Fundamentals of Logic Design, West Publishing Company, 1992.

NAGI ELNAGA

Nagi El Naga is a Professor of Electrical and Computer Engineering at California State University, Northridge. His research interests include computer architecture, computer arithmetic, multiprocessor performance evaluation and Error detecting and correcting Systems. Worked as a consultant in the area of digital system design. He received his B.S. and M.Sc. degrees in Electrical Engineering from Ain Shams University in 1970 and 1974 and his Ph.D. in Computer Engineering from the University of Waterloo, Canada in 1978.

HALIMA EL NAGA

Halima El Naga is an Associate Professor of Electrical and Computer Engineering at California State Polytechnic University, Pomona. Her research interests include parallel processing, computer architecture and memory systems for shared memory multiprocessors. She received her B.S. degree in Electrical Engineering from Ain Shams University in 1977, M.S. in Electrical and Computer He received his B.S. and M.Sc. degrees in Electrical Engineering from Ain Shams University in 1970 Engineering from California State University, Northridge in 1987 and her Ph.D. in Computer Engineering from the University of Southern California in 1999.