Applications of Fuzzy Program Graph in Symbolic Checking of Fuzzy Flip-Flops

Gholamreza Sotudeh [a*], Ali Movaghar [b]

[a] *Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran*
[b] *Department of Computer Engineering, Sharif University of Technology, Tehran, Iran*

Abstract

All practical digital circuits are usually a mixture of combinational and sequential logic. Flip–flops are essential to sequential logic therefore fuzzy flip–flops are considered to be among the most essential topics of fuzzy digital circuit. The concept of fuzzy digital circuit is among the most interesting applications of fuzzy sets and logic due to the fact that if there has to be an ultimate fuzzy computer then fuzzy circuitry is inevitable. In this research field, hardware realization of fuzzy negation, t–norms and t–conorms have been well studied in details meanwhile no formal model is introduced for more complex fuzzy circuitry such as combinational circuits, sequential circuits or memory modules. The lack of a formal model checker indicates flaws and design deficiencies are usually remain out of sight therefore validating fuzzy logic circuits was impossible to this date. In this paper we are elaborating the application of Fuzzy Program Graph in symbolic checking of fuzzy flip–flops; thus, the content is mainly focused on formal modelling of fuzzy flip–flops and investigating their correctness. To this purpose we investigated design deficiencies of a multivalued D flip–flop and found a dynamic hazard then we proposed a formal model toward fuzzy J–K flip–flops to further elaborate applications of proposed formal model and model checking approach in detecting design phase deficiencies.

Keywords: Fuzzy Flip–Flop, Fuzzy Kripke Model, Fuzzy CTL, Fuzzy Program Graph.

1. Introduction

Model Checking is a technique proposed by Clarke in order to perform formal verification of temporal properties for reactive systems [3]. During the process a mathematical model (usually in the form of a graph) is extracted from a system profile then a set of verification related propositions of the system called "properties" are also conformed to a formal language (denoted as temporal logic). Finally, in a mechanized approach, an algorithm is defined and implemented in order to exhaustively and automatically check properties of the model.

Most of recent studies in digital circuit design showed a vast amount of interest in model checking and generally formal verification techniques to verify synchronous and asynchronous logics. In particular, model checking is used to find bugs such as lack of stability, and hazard occurrence early in the design phase of a digital circuit. Practical digital circuitry is a combination of sequential and combinational circuits.

* Corresponding author. Email: setoodeh@iaushiraz.ac.ir

Burch et al. introduced a symbolic model checking approach to verify sequential circuits by modelling sequential behaviour of a circuit using Binary Decision Diagram (BDD) and then used Computation Tree Logic CTL for model verification [2].

There are more than a few efforts of making fuzzy circuitry practical, none of which could provide a formal verification approach for fuzzy digital circuits. Fuzzy sequential logic is among the first steps toward a fuzzy computer. Mori et al. [8] studied a high level abstracted model of fuzzy sequential logic, yet no formal verification approach is devised since then. They have overlooked the preliminaries of devising a fuzzy sequential circuit by ignoring the role of fuzzy flip–flops in fuzzy circuitry. Flip–flops are building blocks of sequential circuits, thus we cannot ignore them in devising a fuzzy sequential circuit.

Hirota et al. introduced the concept of fuzzy flip–flops in late 1980s [4]. They introduced fuzzy flip–flop as an extended or "fuzzyfied" binary flip–flop. Due to the fact that fuzzy connectives do not conform to all Boolean axioms, Hirota and his students proposed a unified formula for fuzzy J–K flip–flops [9]. Yet again no formal verification approach is devised since then. In this paper we demonstrate a formal model of fuzzy J–K flip–flop and its properties expressed by a fuzzy temporal logic then we address an issue with the behaviour of proposed flip–flop.

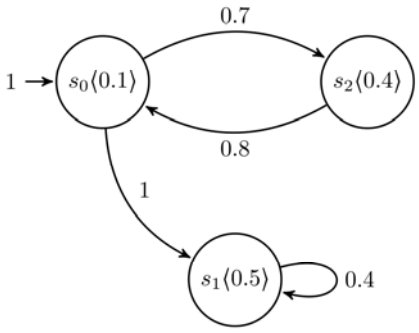


Fig. 1. FzKripke K_2 in which X={x} and function I returns 1 for s_0 and 0 for other states. The edges of K_2 are demonstrating the R relation and the L relation is represented by decimals depicted in each state.

The outline of this paper is as follows. Sections 2 to 4 are dedicated to review the premises and preliminary definitions of formal methods used in this paper, previously introduced in [13]. In section 5, first we investigate the correctness of a multivalued D flip–flop proposed by [1] then we present a formal model of fuzzy J–K flip–flops using two variety of fuzzy logic gates; in this section the main properties of a fuzzy flip–flops are conformed to a formal language and their validity are examined. Finally, configurations of flip–flop's model are also traced for improper behaviour of D flip–flop and proposed J–K flip–flops.

2. Fuzzy Kripke Model

This model is similar to XKripke model, except for its lattice; which is the continuous interval [0,1].

This model is defined as an ordered tuple, in the form of M=(S,X,R,L,I) where $X=(x_1,\ldots,x_m)$ is a set of attributes and $S=\{s_1,\ldots,s_n\}$ is a set of states. Every attribute has a possibility value and different possible values for the whole attributes is expressible by *Val(X)* as shown in (1).

$$Val(X)=\{\langle v_1,\ldots,v_m\rangle \mid v_i\in[0,1]\} \quad (1)$$

For attribute evaluation, access to the value of an attribute is defined by a dot operator as shown in (2).

$$\mu\in Val(X),\mu=\langle v_1,\ldots,v_m\rangle \Rightarrow v_i=\mu.x_i \quad (2)$$

A function called **L**, assigns a label to each state as the state's specific valuation, see (3). Relation (4) defines **R** as a function that specifies the possibility of transition from one state to another. Entrance possibility for each state at start time can be represented by **I** a function that is defined in (5).

$$L:S\rightarrow Val(X) \quad (3)$$
$$R:S\times S\rightarrow[0,1] \quad (4)$$
$$I:S\rightarrow[0,1] \quad (5)$$

Following notation represents the transition between two states with a specific possibility.

$$s_i \xrightarrow{r} s_j \Leftrightarrow \left(\left(s_i, s_j\right), r\right) \in R$$

A finite execution path π starting from s'_0 can be defined as follows:

$$x \in Path_{fin}(s'_0) \Leftrightarrow \pi = s'_0 \xrightarrow{r_0} s'_1 \xrightarrow{r_1} s'_2 \xrightarrow{r_2} \ldots \xrightarrow{r_{u-1}} s'_u$$
$$\forall i \in 0..u_{i-1}.r_i \in [0,1]$$

The infinite execution path can be defined similarly:

$$x \in Path_{fin}(s'_0) \Leftrightarrow \pi = s'_0 \xrightarrow{r_0} s'_1 \xrightarrow{r_1} s'_2 \xrightarrow{r_2} \ldots$$
$$\forall i \in N.r_i \in [0,1]$$

A certain state on a path and a sub-path can be defined by the following notation:

$$\forall i \in N.\pi[i] = s'_i \wedge \pi[i\ldots] = s'_i \xrightarrow{r_i} s'_{i+1} \xrightarrow{r_{i+1}} \ldots$$

3. Fuzzy Computation Tree Language

Properties of Kripke Structures are often expressed by temporal logics; similarly a suitable fuzzy extended temporal logic is required to express properties of certain FzKripke models. CTL is a suitable temporal logic to demystify properties of Kripke structures; hence, extending CTL to include fuzzy logic is the preferable choice to this purpose. In order to define Fuzzy Computation Tree Language (FzCTL) introducing some fuzzy operators is necessary. Fuzzy operators have a high variety of implementations, See [12, 15], in this paper we use their simplest implementations whose properties are similar to the properties of quasi-Boolean algebra related to XCTL. The fuzzy operations are defined as follows:

true = 1
false = 0

$$\neg a = 1 - a$$
$$a \prod b = \min(a, b)$$
$$a \coprod b = \max(a, b)$$
$$a \to b = \max(b, 1-a) = b \coprod \neg a$$

Using the saturation operator [7], shown in (6), we have defined *bounded-add* operator. Although this operator is not seen in XCTL, it is defined in this logic.

$$\prec a \succ = \max(0, \min(1,0)) \tag{6}$$

FzCTL is a two–stage syntax logic in which formulae are separated to state and path formulae. State propositions are formulated by the following grammar:

$$\varphi ::= r \mid x \mid \neg\varphi \mid \varphi \prod \varphi \mid \prec \varphi + \varphi \succ \mid \varphi \geq \varphi \mid A\Phi \mid E\Phi$$

where $r \in Q \cap [0,1]$ and x is a state attribute. The path propositions are formulated by the following grammar:

$$\Phi ::= \varphi \mid \neg\Phi \mid \Phi \prod \Phi \mid \prec \Phi + \Phi \succ \mid \Phi \geq \Phi \mid X\Phi \mid \Phi U \Phi$$

Modal adverbs are also needed; the adverb *"Finally"* is defined as shown in (7) and *"Generally"* is defined in (8) as another modal adverb.

$$F\Phi \overset{def}{=} true \cup \Phi \tag{7}$$

$$G\Phi \overset{def}{=} \neg F \neg \Phi \tag{8}$$

A set of auxiliary operations may be defined in this grammar. The operators can be implemented using previously defined simple operators, each of which can be used for defining state propositions as well as path propositions.

$$\prec a - b \succ = \neg \prec \neg a + b \succ$$

$$if(a,b,c) = (a \prod b) \coprod (\neg a \prod c)$$

Symbols P and $\mapsto$ are used to present the truth possibility of a proposition, when being in a state or

on a path. In the following notations, the FzKripke model is represented by M and s is representing a certain state of the correspondent model. π is an infinite path defined on the M.

$$P(M, s \mapsto \varphi) \overset{def}{=} P(\varphi \mid M, s)$$

$$P(M, \pi \mapsto \Phi) \overset{def}{=} P(\Phi \mid M, \pi)$$

The semantic of propositions on states is as follows, where "op" can be a binary logical operator, a comparison operator, or the bounded-add/subtract.

$$P(M, s \mapsto r) = r$$
$$P(M, s \mapsto x) = L(s).x$$
$$P(M, s \mapsto \neg\varphi) = \neg P(M, s \mapsto \varphi)$$
$$P(M, s \mapsto \varphi \, op \, \psi) = P(M, s \mapsto \varphi) \, op \, P(M, s \mapsto \psi)$$

Konikowska and Penczek articulated some problematic issues on the symbol $\mapsto$ in [6]. This symbol is inefficient and ambiguous in expressing semantic of XCTL formulas, therefore instead of using $\mapsto$ we use $\mapsto_A$ and $\mapsto_E$ in following expressions.

$$P(M, s \mapsto A\Phi) = \prod_{\pi \in Path_{\inf}(s)} P(M, \pi \mapsto_A \Phi)$$
$$P(M, s \mapsto E\Phi) = \coprod_{\pi \in Path_{\inf}(s)} P(M, \pi \mapsto_E \Phi)$$

Semantic of propositions on paths is as follows. In which, "*op*" can be a binary logical operator, a comparison operator, or the bounded-add/subtract.

$$P(M, \pi \mapsto_A \varphi) = P(M, \pi \mapsto_E \varphi) = P(M, \pi[0] \mapsto \varphi)$$
$$P(M, \pi \mapsto_A \Phi \, op \, \Psi) = P(M, \pi \mapsto_A \Phi) \, op \, P(M, \pi \mapsto_A \Psi)$$
$$P(M, \pi \mapsto_E \Phi \, op \, \Psi) = P(M, \pi \mapsto_E \Phi) \, op \, P(M, \pi \mapsto_E \Psi)$$
$$P(M, \pi \mapsto_A \neg\Phi) = \neg P(M, \pi \mapsto_E \Phi)$$
$$P(M, \pi \mapsto_E X\Phi) = R(\pi[0], \pi[1]) \prod P(M, \pi[1...] \mapsto_E \Phi)$$
$$P(M, \pi \mapsto_A X\Phi) = R(\pi[0], \pi[1]) \to P(M, \pi[1...] \mapsto_A \Phi)$$
$$P(M, \pi \mapsto_A \Phi U \Psi) = P(M, \pi \mapsto_A \Psi) \coprod (P(M, \pi \mapsto_A \Phi) \prod P(M, \pi \mapsto_A X(\Phi U \Psi)))$$
$$P(M, \pi \mapsto_E \Phi U \Psi) = P(M, \pi \mapsto_E \Psi) \coprod (P(M, \pi \mapsto_E \Phi) \prod P(M, \pi \mapsto_E X(\Phi U \Psi)))$$

Consequently the truth possibility of a proposition on the whole model should be defined as follows:

$$P(M \mapsto \varphi) \overset{def}{=} \prod_{s \in S} \left(I(s) \to P(M, s \mapsto \varphi) \right)$$

Using the above-defined propositions along with the semantic of FzCTL correctness of the following equations can be investigated, where μ and ν stands for the greatest and smallest fixed-points [14], respectively.

$$\neg AG\varphi = EF\neg\varphi$$
$$\neg AF\varphi = EG\neg\varphi$$
$$\neg A(\varphi U \psi) = EG(\neg\psi) \coprod E(\neg\psi \, U \, \neg\varphi \prod \neg\psi)$$
$$E(\varphi U \psi) = \mu Z.(\psi \coprod (\varphi \prod EX(Z)))$$
$$\neg A(\varphi U \psi) = \nu Z.(\neg\psi \prod (\neg\varphi \coprod EX(Z)))$$

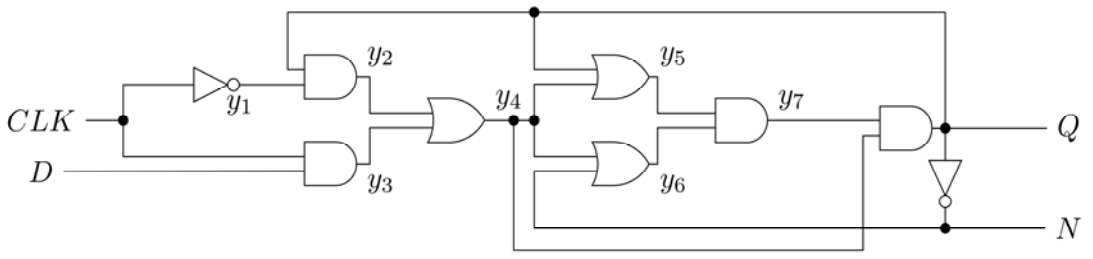


Fig. 2. Block diagram of multivalued D flip–flop which was proposed in [1]. AND gates represent min function; similarly OR gates represent max function. We have labeled the output of each and every gate with y 1 to y 7 , N, and Q for further readability.

For example the following proposition holds for k_1 as shown in figure 1:

$$P(K_1 \mapsto EF(x)) = 0.5$$
$$P(K_1 \mapsto EF(\neg x)) = 0.9$$
$$P(K_1 \mapsto EG(x < 0.5)) = 0.7$$

4. Fuzzy Program Graph

Although FzKripke provides a comprehensive modeling approach for discrete time fuzzy systems, its state space is often too large to handle in memory. In this section we will review a more compressed modeling structure for fuzzy systems based on Program Graph called *Fuzzy Program Graph (FzPG)*. FzPG is an extended Program Graph in the form of a quintuple *G=(S,s₀,X,Init,Act)*. In this model, $s \in S_0$ is the initial state, and *Init* is a function that defines entrance possibility to initial state. *Act* is a relation defining state transitions each of which has two parts; (1) a function that defines transition possibility, and (2) a function that maps possibility values from each and every attribute of the source state to those of destination state.

$$Init \in f_x$$

$$Act \in S \times S \to F_X \times G_X$$

Where $F_X \in PowerSet(val(X) \to [0,1])$ and $G_X = F_X^{|X|}$ and all functions belonging to F_X have this constraint that they must be compatible with the syntax of following grammar's syntax.

$$\varphi ::= x \left| \kappa \right| \neg\phi \left| \varphi \sqcap \varphi \right| \varphi \sqcup \varphi \left| \varphi \to \varphi \right| \prec \varphi \pm \kappa \succ$$

$$\kappa ::= r \left| \varphi > \kappa \right| \varphi < \kappa \left| \varphi \geq \kappa \right| \varphi \leq \kappa \left| \varphi = \kappa \right| \varphi \neq \kappa \left|\right.$$

$$\prec \theta(\kappa,...,\kappa) \succ_\varepsilon$$

Where $r, \varepsilon \in Q \cap [0,1], x \in X$, $\prec a \succ_\varepsilon = \varepsilon \left\lfloor \frac{a}{\varepsilon} \right\rfloor$ and $\frac{1}{\varepsilon}$ is a natural number, κ is an expression with discrete values, and θ stands for an arbitrary analytical function with one or more arguments as follows:

$$\theta \in \bigcup_{k \in N^+} ([0,1]^k \to R)$$

meanwhile each logical operator as well as comparison, or bounded-add/subtract, can be a specific case of θ.

It is shown an arbitrary FzPG is convertible to a fuzzy extended Kripke structure [13]. When the equivalent FzKripke model accepts a proposition with a certain possibility, we say that the FzPG also accepts the proposition with the same possibility. Equivalent to each FzKripke, an FzPG with the same number of states can be defined; however, this conversion is not valuable. The main goal of defining of FzPG is to express FzKripke in a highly compressed format.

5. Fuzzy Flip–Flops

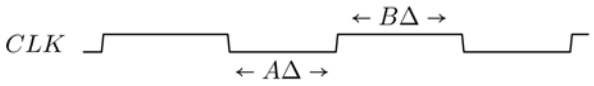
The concept of fuzzy flip–flops is already proposed by Hirota et al. [4, 5, 10, 11, 1]. They presented the idea of designing fuzzy hardware systems using fuzzy flip–flops. Fuzzy flip–flops are made of fuzzy logic gates through having some extensions to binary flip–flops, yet few of them are suitable for realizing neurons in a multilayer perceptron for instability is the issue with almost all of them; this means the output of the circuit may fluctuate under certain conditions. In this section we investigate the correctness of a previously proposed multivalued D flip–flop using our proposed formal method then we introduce a formal model of fuzzy J–K flip–flop and investigate the correctness of its behaviour.

5.1. Fuzzy D Flip–Flop

Ben Choi and Kankana Shulka proposed a multivalued D flip–flop in [1]. Although they investigated the validity of their proposed circuit via computer simulations, under certain conditions gate delays lead to dynamic hazards that can be

demonstrated by the proposed model checking approach; this means there is a condition where unpredictable sequence of outputs are generated constantly which leads to an unstable fuzzy circuit. For your information, we used a quite similar approach to verify another multivalued D flip–flop in [13] which led to discovery of static hazards in that circuitry.

Figure 2 illustrates the block diagram of Choi's D flip–flop. For the sake of simplicity we assume propagation delay for all gates are the same and the value is equal to $\Delta = 2^{-h}$ where h is a positive integer. We also assume input D is stable and the *CLK* is a pulse in the following form:



where A and B are integers. Great values of A and B do not cause any problems except slowing down the circuit. On the other hand, if A and B would be smaller than a limit there are no chance for the circuit to stabilize its output therefore it acts incorrectly. It can be proved by model checking that this design requires the values of A and B to be at least 7. Let us assume $N = 1/\Delta$ be the maximum possible value for A and B (obviously this is an assumption to simplify the modeling problem, greater values would not affect the circuit but slowing it down).

We have also labeled all the outputs for each logical gate in this block diagram, please see figure 2. We also use the prime symbol ($'$) to label the outputs of logical gates after Δ time units; now it is easy to compute the output of each and every gates considering their inputs. From this point onward we denote *CLD* as *C* for further readability.

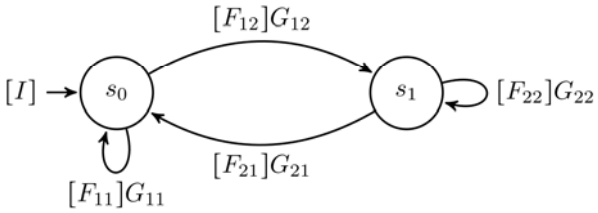


Fig. 3. FzPG G which represents the multivalued D flip–flop depicted in figure 2.

$$y'_1 = \neg C$$
$$y'_2 = y_1 \sqcap Q$$
$$y'_3 = D \sqcap C$$
$$y'_4 = y_2 \sqcup y_3$$
$$y'_5 = y_4 \sqcup Q$$
$$y'_6 = y_4 \sqcup N$$
$$y'_7 = y_5 \sqcap y_6$$
$$Q' = y_4 \sqcap y_7$$
$$N' = \neg Q$$

Now we consider an FzPG with two states s_0, and s_1. Being in state s_0 means *CLK* is 0 and being in s_1 means the opposite, please see figure 3. The attribute set X is as follows:

$$X = \langle T, u, D, C, y_1, y_2, y_3, y_4, y_5, y_6, y_7, Q, N \rangle$$

Attribute T represents the passage of time in states of FzPG with steps of Δ, therefore on the verge of entering a new state the value for this attribute is 0. Whilst on a state, as soon as T changes the output of all gates will change accordingly. Attribute u represents the raising edge of the clock pulse; it is set to 1 while the clock is pulse raised. Once u is set it will preserve its value.

Participating functions in graph G are defined as:

$$I = (T = 0) \sqcap (C = 0) \sqcap (u = 0)$$
$$F_{11} = (T < A\Delta)$$
$$F_{12} = (T = A\Delta)$$
$$F_{22} = (T < B\Delta)$$
$$F_{21} = (T = B\Delta)$$
$$G_{11} = G_{22} = \langle \prec T + \Delta \succ, u, D, C, \neg C, y_1 \sqcap Q, D \sqcap C,$$

$y_2 \amalg y_3, y_4 \amalg Q, y_4 \amalg N, y_5 \prod y_6, y_4 \prod y_7, \neg Q \rangle$

$G_{12} = \langle 0,1,D,1,y_1,y_2,y_3,y_4,y_5,y_6,y_7,Q,N \rangle$

$G_{21} = \langle 0,u,D,0,y_1,y_2,y_3,y_4,y_5,y_6,y_7,Q,N \rangle$

5.1.1. Properties of Multivalued D Flip–Flop

In order to verify Choi's design of multivalued D flip–flop, we defined its properties using FzCTL and then investigated their correctness using model checking.

Property 1. 6Δ after the very first raising edge of clock pulse, we will have $Q=d$ for the rest of time. This property is expressible in terms of FzCTL propositions as follows:

$$AG(u=1 \rightarrow AX^6(AG(Q=d))),$$

Where AX^n means applying AX operator n times consecutively. This proposition evaluates to 0.

Property 2. 6Δ after the very first raising edge of clock pulse, the possible values for Q are d, $\bar{d}$, and 0. This property is expressible in terms of FzCTL propositions as follows:

$$AG(u=1 \rightarrow AX^6(AG(Q=d \amalg Q=0 \amalg Q=\bar{d})))$$

Finally, this proposition evaluates to 1.

Table 1 depicts a trace that nullifies the first property. There is an issue caused by the propagation delay of NOT gate while falling edge occurs. According to table 1, a dynamic hazard with a period of 3Δ emerged at time $t+5\Delta$. As can be observed if $d<\bar{d}$ then on the verge of falling edge of the clock pulse the next value for Q is among d, 0, or $\bar{d}$ otherwise if $d>\bar{d}$ then Q is either d or 0. At the rising edge of the clock pulse in a short time (i.e. less than 6Δ) the sate of circuit will be reverted to column t and as long as clock would be high the state is preserved.

During the model checking process different parameter values for Δ ranging from 2^{-3} to 2^{-6} and a variety of parameter values for A and B ranging from 7 to $1/\Delta$ were used; in all permutations the first property was evaluated to 0 while the second property was evaluated to 1. In case of parameter values less than 7 for either A or B both properties were evaluated to 0. Modifying Δ did not affect the results yet it affected memory consumption and execution time of the proposed model checker.

Table 1

Unstable condition of Fuzzy D flip–flop

Values	Time											
	t	$t+\Delta$	$t+2\Delta$	$t+3\Delta$	$t+4\Delta$	$t+5\Delta$	$t+6\Delta$	$t+7\Delta$	$t+8\Delta$	$t+9\Delta$	$t+10\Delta$	…
c	1	0	0	0	0	0	0	0	0	0	0	…
y_1	0	1	1	1	1	1	1	1	1	1	1	…
y_2	0	0	d	d	0	d	$\bar{d} \cap d$	0	d	$\bar{d} \cap d$	0	…
y_3	d	0	0	0	0	0	0	0	0	0	0	…
y_4	d	d	0	d	d	0	d	$\bar{d} \cap d$	0	d	$\bar{d} \cap d$	…
y_5	d	d	d	d	d	d	$\bar{d} \cap d$	d	d	$\bar{d} \cap d$	d	…
y_6	$\bar{d} \cup d$	$\bar{d} \cup d$	$\bar{d} \cup d$	$\bar{d}$	$\bar{d} \cup d$	1	$\bar{d}$	$\bar{d} \cup d$	1	d	$\bar{d} \cup d$	…
y_7	d	d	d	d	$\bar{d} \cap d$	d	d	$\bar{d} \cap d$	d	d	$\bar{d} \cap d$	…
Q	d	d	d	0	d	$\bar{d} \cap d$	0	0	$\bar{d} \cap d$	0	d	…
N	$\bar{d}$	$\bar{d}$	$\bar{d}$	$\bar{d}$	1	$\bar{d}$	$\bar{d} \cup d$	1	$\bar{d}$	$\bar{d} \cup d$	1	…

5.2. Fuzzy J–K Flip–Flop

In this subsection we are destined neither to provide an on–chip fuzzy system nor their applications but a formal model realizing a fuzzy flip–flops.

5.2.1. Fuzzy NAND Gate

Fuzzy operations are simple to realize physically, except for more complex gates like eXclusive–OR. Variations of fuzzy NAND gates are implemented in [11], however they are ignored in this study because of their complexity. The NAND gate depicted in fig:JKflipflop is a fuzzy logic gate simply defined by *min-max* norms as follows:

$$NAND_1(x,y) = 1 - \min(x,y) = \neg(x \prod y)$$

This gate can also be defined using the Lukasiewicz t–norm as follows:

$$NAND_2(x,y) = 1 - \max(0, \min(x+y,1)) = \prec \neg x + \neg y \succ$$

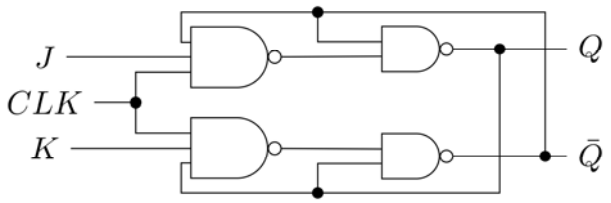


Fig. 5. J–K flip–flop block diagram.

5.2.2. Properties of Fuzzy J–K Flip–Flop

Input and output values of fuzzy *NAND* gate are decimals between [0, 1]. Suppose numbers smaller or equal to 0.25 as low and numbers larger or equal to 0.75 as high and the numbers in–between these two bounds as invalid values. According to figure 4 the following properties holds for each fuzzy J–K flip–flop.

Property 3. If J is high and K is low at the clock edge then Q output is forced high and stays high while $\overline{Q}$ is forced low and stays low for sure. Note that at the beginning initial values of Q and $\overline{Q}$ are random and even may be invalid values like 0.5.

Proof. Model checking is the formal method to verify this property. Corresponding FzPG to the J–K

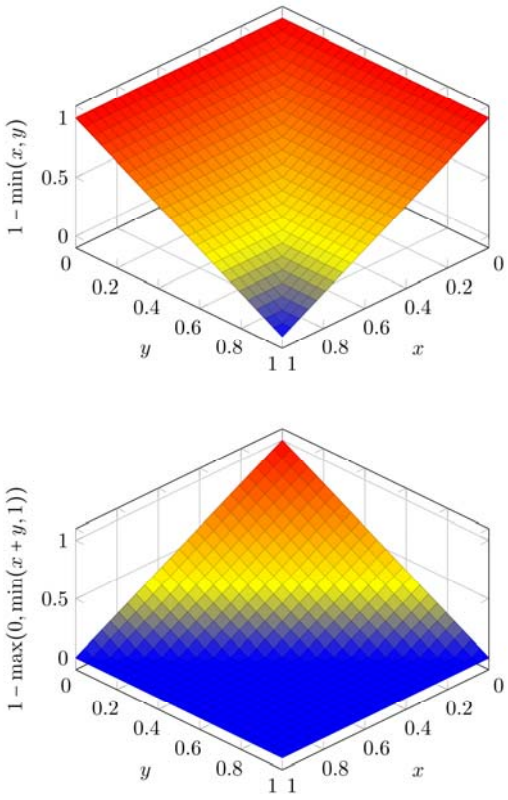


Fig. 4. Surfaces of $NAND_1$ and $NAND_2$.

flip–flop shown in figure 4 is defined as follows:

$$G = \left(\{s_0\}, s_0, \{J, K, Q, \overline{Q}\}, Init, Act\right)$$

$$Init(s_0) = [1]$$

$$Act(s_0, s_0) = ([1], \langle J, K, Q_{next}, \overline{Q}_{new} \rangle),$$
$$Q_{next} = NAND(\overline{Q}, NAND(\overline{Q}, J)),$$
$$\overline{Q}_{next} = NAND(Q, NAND(Q, K))$$

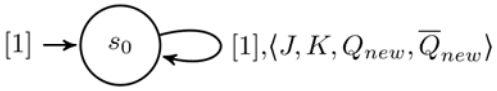


Fig. 6. FzPG of the J–K flip–flop shown in figure 4.

Following is FzCTL proposition of above property:

$$P_1 = J \geq 0.75 \prod K \leq 0.25 \rightarrow AF(AG(Q \geq 0.75 \prod \overline{Q} \leq 0.25))$$

If we rewrite *NAND* functions (as shown bellow) they can be used to construct corresponding FzPG. Discrete saturation operator is used to quantize input values in order to have a finite number of states equivalent FzKripke.

$$NAND_1(x, y) = \neg((\prec x \succ_\varepsilon) \prod (\prec y \succ_\varepsilon))$$

$$NAND_2(x, y) = \prec \neg(\prec x \succ_\varepsilon) + \neg(\prec y \succ_\varepsilon) \succ$$

where $\varepsilon = 2^{-d}, d \geq 2$.

Using vectors of *Ordered Binary Decision Diagram* (OBDD) of BuDDy library [16] an equivalent FzKripke for FzPG *G* is implemented. A verification method for proposition P_1 is implemented whose details require a lot of preparation that does not fit in this case study.

If we use gate $NAND_1$ while constructing FzPG *G* there is a condition in which flip–flop works improperly (and that is when the initial state values for the Q and $\overline{Q}$ are invalid) thus the property is incorrect and proposition P_1 evaluates to 0; see table 2 for traces and configuration of model. By substitution of $NAND_1$ with $NAND_2$, proposition P_1 (for all $\varepsilon \geq 0.25$) evaluates to 1, and the property always holds.

Table 2

Unstable condition of Fuzzy J–K flip–flop using $NAND_1$.

Values	Steps							
	0	1	2	3	4	5	6	…
J	0.75	0.75	0.75	0.75	0.75	0.75	0.75	…
K	0.25	0.25	0.25	0.25	0.25	0.25	0.25	…
Q	0.5	0.625	0.5	0.625	0.5	0.625	0.5	…
$\overline{Q}$	0.625	0.5	0.375	0.375	0.375	0.375	0.375	…

If we want the property holds for both gates $NAND_1$ and $NAND_2$ another criteria should be imposed to proposition P_1. The following proposition corrects improper behaviour of the flip–flop.

$$P_1' = J \geq 0.75 \prod K \leq 0.25 \prod (Q \geq 0.75 \coprod Q \leq 0.25) \prod (\overline{Q} \geq 0.75 \coprod \overline{Q} \leq 0.25) \rightarrow AF(AG(Q \geq 0.75 \prod \overline{Q} \leq 0.25))$$

6. Conclusion & Future Work

In this paper we showed the application of a non–classical logic and its corresponding model in verifying fuzzy systems such as fuzzy logic circuits. We have formally modeled a multivalued D flip–flop and also a Fuzzy J–K flip–flop and verified their properties using provided formal methods.

Although we used Zadeh and Lukasiewicz t–norms to formally model fuzzy J–K flip–flop, more sophisticated t–norms (e.g. algebraic product) can be used to modify flip–flop's behaviour. For future work we intend to extend our research to devise formal verification approach for such models. Verifying more sophisticated fuzzy logic circuits which are built on the concept of fuzzy J–K flip–flop such as fuzzy neural networks is also interesting to us.

Acknowledgement

We thank Masoud Ebrahimi for being of high assistance with this research, and his comments that greatly improved the manuscript.

References

[1] B. Choi, K. Shukla, Multi-valued logic circuit design and implementation, International Journal of Electronics and Electrical Engineering 3 (4) (2015) 256–262. doi:10.12720/ijeee.3.4.256-262 .

[2] J. Burch, E. Clarke, D. Long, K. McMillan, D. Dill, Symbolic model checking for sequential circuit verification, Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on 13 (4) (1994) 401–424. doi:10.1109/43.275352 .

[3] E. M. Clarke Jr., O. Grumberg, D. A. Peled, Model Checking , MIT Press Cambridge, MA, USA, 1999. URL http://dl.acm.org/citation.cfm?id=332656

[4] K. Hirota, K. Ozawa, The concept of fuzzy flip–flop , IEEE Transactions on Systems, Man, and Cybernetics 19 (5) (1989) 980–997. doi:10.1109/21.44013. URL http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=44013

[5] K. Hirota, W. Pedrycz, Design of fuzzy systems with fuzzy flip–flops (1995). doi:10.1109/21.362956 .

[6] B. Konikowka, W. Penczek, On designated values in multi-valued ctl* model checking , Fundam. Inf. 60 (1-4) (2003) 211–224. URL http://dl.acm.org/citation.cfm?id=1226781.1226796

[7] W. Liang, W. Bing-wen, G. Yi-Ping, Cell mapping description for digital control system with quantization effect , Tech. rep. (Dec. 2007). arXiv:0712.2501. URL http://arxiv.org/abs/0712.2501

[8] Y. Mori, K. Otsuka, M. Mukaidono, Properties of fuzzy sequential circuit using fuzzy transition matrix and their design method, in: Fuzzy Systems, 1995. International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., Proceedings of 1995 IEEE Int, Vol. 4, 1995, pp. 2133–2138 vol.4. doi:10.1109/FUZZY.1995.409975 .

[9] K. Ozawa, K. Hirota, L. Koczy, K. mori, Algebraic fuzzy flip-flop circuits , Fuzzy Sets and Systems 39 (2) (1991) 215 – 226, applications of fuzzy systems theory. doi:http://dx.doi.org/10.1016/0165-0114(91)90214-B. URL http://www.sciencedirect.com/science/article/pii/016501149190214B

[10] K. Ozawa, K. Hirota, L. T. Koczy, W. Pedrycz, N. Ikoma, Summary of fuzzy flip-flop, in: Proceedings of 1995 IEEE International Conference on Fuzzy Systems, Vol. 3, 1995. doi:10.1109/FUZZY.1995.409897 .

[11] K. Ozawa, K. Hirota, L. Koczy, Fuzzy flip-flop, in: M. J. Patyra, D. M. Mlynek, eds., Fuzzy Logic. Implementation and Applications, Wiley, Chichester, 1996, pp. 197–236.

[12] N. Sladoje, On analysis of discrete spatial fuzzy sets in 2 and 3 dimensions , Ph.D. thesis, Swedish University of Agricultural Sciences Uppsala (2005).

URL http://pub.epsilon.slu.se/963/1/ThesisNSladoje.pdf

[13] G. Sotudeh, A. Movaghar, Abstraction and approximation in fuzzy temporal logics and models , Formal Aspects of Computing (2014) 1–26 doi:10.1007/s00165-014-0318-7 .

URL http://dx.doi.org/10.1007/s00165-014-0318-7

[14] A. Tarski, A lattice-theoretical fixpoint theorem and its applications, Pacific journal of Mathematics 5 (1955) 285–309.

[15] M. Wierman, An Introduction to the Mathematics of Uncertainty , 2010. URL http://aliana.dc.fi.udc.es/files2/MOU.pdf

[16] J. Lind-Nielsen, D. T. U. I. for Informationsteknologi, BuDDy - a Binary Decision Diagram Package, IT-TR, Department of Information Technology, Technical University of Denmark, 1996.