

УДК 681.326

- Мірошник Марина** *докт. техн. наук, професор; професор кафедри теоретичної та прикладної системотехніки, Харківський національний університет імені В. Н. Каразіна, майдан Свободи 4, м. Харків, Україна, 61022*
e-mail: m.miroshnyk@karazin.ua
<https://orcid.org/100000002223125291>
- Пшеничний Кирило Юрійович** *аспірант доцент кафедри автоматизації проектування обчислювальної техніки; Харківський національний університет радіоелектроніки, пр. Науки, 14, м. Харків, Україна, 61166*
e-mail: kyrylo.pshenychnyi@nure.ua
<https://orcid.org/0009000707996604>
- Шафранський Андрій Володимирович** *аспірант кафедри теоретичної та прикладної системотехніки, Харківський національний університет імені В. Н. Каразіна, майдан Свободи 4, м. Харків, Україна, 61000*
e-mail: shafranskyi.andrei@student.karazin.ua
<https://orcid.org/0009-0004-7725-3556>
- Шкіль Олександр Сергійович** *к.т.н., доцент; доцент кафедри автоматизації проектування обчислювальної техніки Харківський національний університет радіоелектроніки, пр. Науки, 14, м. Харків, Україна, 61166* e-mail: oleksandr.shkil@nure.ua
<https://orcid.org/0000000310713445>

Підвищення тестопридатності часових автоматів Мура

У роботі запропоновано метод проектування тестопридатних цифрових пристроїв реального часу, представлених у вигляді скінчених автоматів та описаних за допомогою мов опису апаратури (Hardware Description Languages, HDL).

Актуальність. Актуальність роботи полягає у можливості діагностування цифрових пристроїв реального часу під час активної експлуатації.

Методи дослідження. Основним методом дослідження є представлення апаратної надлишковості у вигляді додаткових HDL конструкцій у коді опису пристрою та додаткових дуг на графі переходів.

Результати. Моделювання легкотестованого автомата реального часу підтвердило працездатність запропонованого підходу. Результати синтезу за допомогою САПР Xilinx ISE показали, що додаткові апаратні витрати не перевищують 20% порівняно з канонічною моделлю опису.

Висновки. Вирішено задачу проектування тестопридатних пристроїв реального часу на базі легкотестованих автоматів Мура. Запропонований метод дозволяє встановлювати автомат у довільний стан за фіксований час. Такий підхід дозволяє значно спростити процес проведення діагности пристрою.

Наукова новизна даної роботи полягає в розробці підходів і методів створення тестопридатних HDL моделей часових автоматів та їх комбінування шляхом модифікації HDL опису цільового пристрою. Такі методи можна інтегрувати в системи САПР, що дозволяє скоротити загальний час проектування та верифікації.

Ключові слова: легкотестовані системи, автоматизація проектування, скінчені автомати, верифікація, мови опису апаратури, Verilog.

Як цитувати: Мірошник М. А., Пшеничний К. Ю., Шафранський А. В., Шкіль О. С. Підвищення тестопридатності часових автоматів Мура. *Вісник Харківського національного університету імені В. Н. Каразіна, серія Математичне моделювання. Інформаційні технології. Автоматизовані системи управління*. 2023. вип. 58. С.37-47.

<https://doi.org/10.26565/2304-6201-2023-58-04>

How to quote: Miroshnyk M., Pshenychnyi K., Shafranskyi A., Shkil O., "Models of Testable Timed Moore Finite State Machines" *Bulletin of V. N. Karazin Kharkiv National University, series Mathematical modelling. Information technology. Automated control systems*, vol. 58, pp.37-46, 2023.

<https://doi.org/10.26565/2304-6201-2023-58-04>

1 Вступ

Сучасні цифрові пристрої являють собою складні системи, які складаються з безлічі логічних елементів. Мови опису апаратури, такі як Verilog і VHDL, забезпечують високий рівень абстракції, що дозволяє інженеру працювати з цифровим пристроєм у вигляді коду, який описує його функціональність. Таким чином, розробник використовує синтаксис мови для реалізації цільового

пристрою незалежно від його складності та розміру. Водночас залишається актуальним питання діагностики.

Варто відзначити, що структурні методи діагностики, які використовуються для виявлення константних несправностей, не є ефективними. Крім того, ці методи практично неможливо застосувати на практиці для реальних пристроїв.

Відомо, що модель скінченного автомата (Finite State Machine, FSM) є широко поширеною для опису поведінки цифрових систем. Скінчений автомат є математичною абстракцією, яку можна представити різними способами, наприклад, таблицею переходів, графом переходів або блок-схемою. Така модель використовується і для пристроїв логічного управління реальним часом. Для їх реалізації використовується темпоральна модель автомата (timed FSM), а для їх візуального представлення – темпоральний граф переходів (temporal state diagram).

Машинний час, в якому працюють автомати, визначається тактами синхросигнала. Однак цифрові пристрої реального часу працюють в метричному часі. Іншими словами, стан таких пристроїв визначається як вхідними сигналами, так і часом обробки цих сигналів [1]. Таким чином, оскільки переходи між станами безпосередньо залежать від часу, необхідно виразити метричний час у термінах тактових циклів. Також необхідно визначити часові обмеження на графі переходів, який є відправною точкою моделювання.

Канонічна модель кінцевого автомата з часом представлена як $Y(t) = g(X(t), Z(t), T)$, $Z(t+1) = f(X(t), Z(t), T)$. Тут X – множина вхідних сигналів, Z – множина внутрішніх станів автоматів, Y – множина вихідних сигналів, t – машинний час, визначений тактами, d – вихідна функція, t – функція переходу. $T = \{t_c, t_o, t_d\}$ – набір часових параметрів: t_c – часові обмеження (timing constraints), t_o – тайм-аут вихідних сигналів (output timeouts), t_d – вихідні затримки (output delays). Темпоральний граф як канонічне візуальне представлення часових автоматів був введений в [2]. На такому графі присутній таймер, який використовується для затримок у станах. Таймер використовується для зациклювання автомата в певному стані протягом фіксованої кількості тактів. Приклад часового автомата Мура показано на малюнку 1.1.

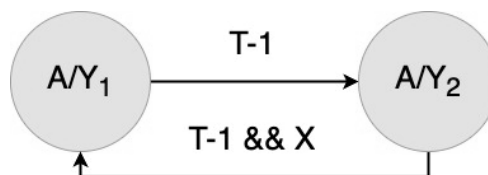


Рис.1.1 Приклад темпорального графу часового автомата Мура

Тут перехід АВ є звичайним часовим переходом, під час якого автомата залишається у стані А протягом T тактів синхросигнала, видаючи вихідний сигнал $Y1$. Умова записується у формі $T - 1$, оскільки внутрішній таймер працюватиме від $t = 0$ до $t = T - 1$. Перехід ВА залежить від стану допоміжного таймера та вхідного сигналу X (умовно-часовий перехід). Таким чином, автомат залишатиметься в стані В, видаючи сигнал $Y2$ принаймні протягом $T - 1$ тактових циклів. З точки зору HDL, часові переходи реалізуються через перебування в певному стані – зациклюванням у стані.

2 Постановка задачі

На початковому етапі проектування цифровий пристрій реального часу представляється у вигляді кінцевого автомата – математичної абстракції, яка базується на специфікації цифрового пристрою. Автоматний шаблон— це форма опису логіки автомата за допомогою мов опису апаратури, таких як SystemVerilog, Verilog, VHDL, SystemC тощо. Зазвичай автоматний шаблон складається з окремих процесів, які представляють функції переходів, виходів та спеціального синхронізованого процесу, який реалізує перехід у новий стан.

Було розроблено різні методики верифікації та діагностики з урахуванням складності та специфіки скінчених автоматів. Однією з них є модифікація моделей автомата для впровадження апаратної надлишковості, як структурно (додаткові входи та виходи для спрощення проведення діагностики), так і функціонально (доповнення та зміни функціонального опису автомата, зокрема графу). Такі стратегії мають на меті спростити подальшу функціональну верифікацію цифрового пристрою.

Метою цієї статті є розробка процедур побудови легкотестованих скінчених часових автоматів з використанням мов опису апаратури шляхом введення апаратної надлишковості у автоматний шаблон.

3 Огляд літератури

У [1] розглядаються апаратні методи реалізації подієвих автоматів. В залежності від обробки вхідних сигналів автомати класифікуються на активні та пасивні, в залежності від способу генерації вихідних сигналів – на моделі Мура та Мілі. Лічильник автоматних тактів використовується для реалізації часових параметрів. Моделі часових автоматів реалізуються за допомогою автоматного шаблону на мові опису апаратури VHDL. Поведінкове моделювання запропонованих моделей, їхній синтез та реалізація в ПЛІС, а також постсинтез-моделювання були проведені за допомогою САПР Xilinx ISE 14.7.

У [5] усі скінчені автомати розподілено на три категорії: регулярні, часові та рекурсивні. Часовий автомат визначається як автомат, який має принаймні один залежний від часу перехід. Для кожної з розглянутих категорій пропонується загальна реалізація на мовах Verilog та VHDL.

У [2] запропоновано спосіб верифікації систем логічного керування реального часу за допомогою асерцій. Вхідною точкою запропонованої методики є темпоральний граф переходів і відповідна реалізація на мові опису апаратури. Запропонований метод ґрунтується на апараті тверджен, який використовується для опису темпоральних властивостей часових автоматів.

Питання знаходження встановлюючих (діагностичних) послідовностей для дискретних автоматів та визначення їхньої довжини розглядається в [4].

Методи підвищення тестоздатності шляхом додавання апаратної надлишковості в реалізацію цифрових пристроїв широко використовуються під час проектування. Концепція «контрольованості» (controllability) та «спостережуваності» (observability) були запропоновані як фундаментальні ознаки тестопридатності у класичній роботі з розробки та класифікації структурних методів тестопридатного проектування [7]. Викладено принципи організації зсувного регістра в ділянці пам'яті цифрового пристрою та створення на їх основі шляху сканування (scanned path). Крім того, запропоновано вдосконалення відомих структурних методів тестування з використанням технологій сканування. У [8] наведено практичні рекомендації щодо використання структурних підходів для підвищення тестоздатності цифрових пристроїв та розробки на їх основі діагностичних алгоритмів.

Функціональні підходи до розробки тестопридатних цифрових пристроїв детально обговорюються в [9]. Для цифрових автоматів, представлених у вигляді таблиці переходів, введено поняття про діагностовані та безумовно діагностовані класи автоматів та запропоновано методи приведення таблиць станів автоматів до потрібних класів. Розглянуто процедури проведення діагностичних експериментів над автоматами із застосуванням встановлюючих та синхронізуючих послідовностей. Крім того, надано обґрунтування підвищенню тестопридатності автомата шляхом включення апаратної надлишковості через розширення вхідного алфавіту, вихідного алфавіту та алфавіту внутрішніх станів. У [10] розглянуто проблеми організації діагностичних експериментів шляхом відвідування всіх вузлів та дуг на графі переходів. Запропоновано метрику визначення довжини та повноти діагностичних експериментів.

Питання автоматизації проведення діагностичних експериментів у автоматних HDL моделях в [13, 14]. Розглянуто процедури організації діагностичних експериментів у системі верифікації САПР для ПЛІС та запропоновано процедури локалізації проектних помилок у моделях HDL.

Аспекти підвищення надійності на початкових етапах проектування цифрових пристроїв, у тому числі способи захисту цифрового контенту вбудованих систем, розглядаються в [15].

4 Виклад основного матеріалу

Визначимо тестопридатний цифровий пристрій як такий що задовольняє наступним вимогам:

1. Можливість генерації тестових наборів;
2. Наявність метрики оцінки ефективності тестів.
3. Можливість проведення тестової діагностики.

Типовий діагностичний експеримент над автоматом складається з двох етапів: подання вхідних послідовностей та аналіз відповідних вихідних послідовностей. Такі експерименти поділяються на три категорії: ідентифікація внутрішніх станів автомата, ідентифікація вхідних послідовностей автомата та ідентифікація автомата з n станами, який відрізняється від усіх інших автоматів з

такою ж кількістю станів. Діагностичний експеримент, який обходить усі вершини графу переходів, використовується для виявлення проблем проектування в моделях автоматів, представлених таблицею переходів або графом переходів.

Легкотестований автомат визначається як автомат, який можна встановити в будь-який стан за менш ніж n циклів без синхронізуючих послідовностей, де n – загальна кількість станів [14].

Наведене вище твердження має бути адаптовано для часових автоматів. Очевидно, що в часовий автомат може перебувати в певному стані протягом більш ніж n тактів. Крім того, правильність часових переходів є важливим аспектом такого роду автоматів, який необхідно перевірити на ранніх стадіях проектування. Таким чином, визначимо легкотестований часовий автомат як автомат, який задовольняє наступним вимогам:

1. Можливість перемикавання автомата в режим тестування і навпаки на будь-якому циклі роботи.

2. Автоматизація запропонованого методу засобами САПР.

Для класичних автоматів підвищення тестоздатності можливе лише шляхом розширення множини вхідних сигналів X , множини внутрішніх станів Z або множини вихідних сигналів Y [9]. З одного боку, це збільшує кількість елементів, які використовуються під час синтезу схеми. З іншого боку це підвищує тестоздатність дизайну на основі автомата. У [6] автори статті вводять апаратну надлишковість у автоматний шаблон на мові VHDL шляхом додавання фрагментів коду, які дозволяють встановити автомат у довільний стан без синхронізуючих послідовностей. Для ілюстрації запропонованого методу була використана модель регістра зсуву. Вхідний алфавіт розширено новим символом – Sh, який дає змогу увімкнути режим діагностики, який реалізує циклічний обхід всіх станів.

Для ілюстрації запропонованих методів розглянемо автомат, який реалізує модуль управління світлофором на пішохідному переході, запропоновану в [1].

Визначимо множину станів:

1. a_1 – включення автомата, вихідні сигнали відсутні, затримка до одиниці;
2. a_2 – жовтий при зміні (G–R), виходи {YGR, R1, R2}, затримка to_2 ;
3. a_3 – червоний на дорозі, зелений на переході, виходи {R1, G2}, затримка to_3 ;
4. a_4 – жовтий при зміні (R–G), виходи {YRG, R1, R2}, затримка to_2 ;
5. a_5 – зелений на дорозі, червоний на переході, виходи {G1, R2}, затримка to_3 ;
6. a_6 – зелений на дорозі, червоний на переході, виходи {G1, R2}, затримка to_3 ;
7. a_7 – горить тільки жовтий, затримка to_1 .

На рисунку 4.1 представлено темпоральний граф переходів автомата Мура даного пристрою.

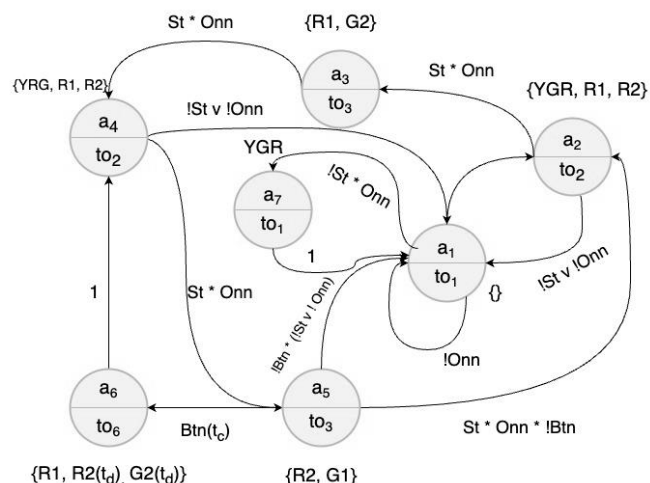


Рис.4.1 Темпоральний граф переходів часового автомата Мура для управління світлофором

Алгоритм функціонування світлофора описується наступним чином: при увімкненні пристрою керування ($Onn=1$), спочатку виконується нічний цикл роботи світлофора, під час якого відбувається миготіння жовтого світла на головній дорозі ($a_1 - a_7$), тоді як світлофор на пішохідному переході не активний. Після початку денного циклу ($St=1$) запускається система переходів ($a_2 - a_3 - a_4 - a_5 - a_2$). У стані a_5 , коли на головній дорозі горить зелений,

встановлюється вікно прийому t_c для зовнішньої події Btn (натискання кнопки переходу). При обробці цієї події керуючий автомат переходить в стан a_6 . В цей момент на головній дорозі вмикається червоний світлодіод, а на пішохідному переході червоний світлодіод затримується, а зелений світлодіод вмикається зі затримкою t_d (час на підготовку до переходу). За період t_c може бути прийнятий тільки один сигнал (зовнішня подія) Btn .

Розширимо вхідний набір сигналом Bps (Bypass). Коли цей сигнал дорівнює 1, автомат працює у режимі діагностики, який полягає у послідовному обході всіх станів автомата. При $Bps = 0$ автомат реалізує заданий алгоритм. Оновлений граф переходів показаний на рисунку 4.2.

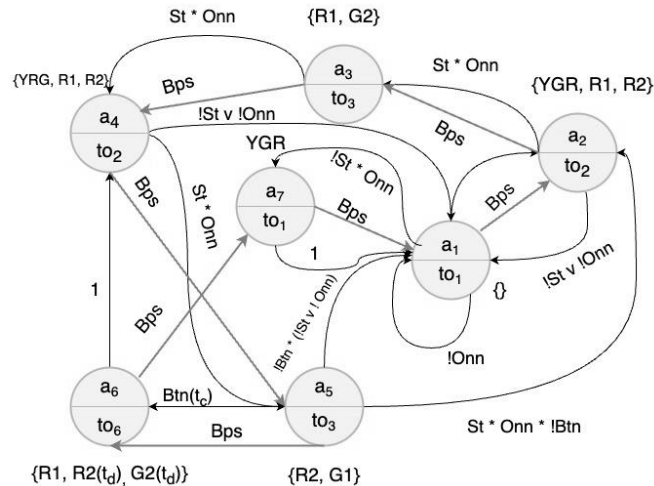


Рис.4.2 Темпоральний граф переходів часового автомата Мура для управління світлофором з діагностичним входом Bps

Необхідно внести відповідні зміни в автоматний шаблон, а саме в процес, що відповідає за визначення наступного стану. Перевірка сигналу bps повинна мати найвищий пріоритет у операторі if-else. Відповідний фрагмент коду Verilog представлено на малюнку 4.3. Тут сигнал bps перевіряється перед іншими сигналами та внутрішнім лічильником. Важливо відзначити, що сигнал $count1$ повинен бути призначений в операторі if, щоб уникнути синтезу латч-тригера для цього сигналу.

```

a1: begin
  if (bps)
    nextState = a2;
    count1 = 3'b000;
  else
    if(count < T1 - 1) begin
      nextState = a1;
      count1 = count + 1'b1;
    end
    else if(st && onn) begin
      nextState = a2;
      count1 = 3'b000;
    end
    else if (!st && onn) begin
      nextState = a7;
      count1 = 3'b000;
    end
    else
      begin
        nextState = a1;
        count1 = 3'b000;
      end
  end
end

```

Рис.4.3 Пріоритизація сигналу bps в описі процесу переходів

На малюнку 4.4 показано результати моделювання модифікованого автомата. Якщо сигнал *bps* дорівнює 0, то автомат працює в стандартному режимі (від 0 до 330 пс). Якщо *bps* дорівнює 1 – усі стани відвідуються циклічно (цикл Гамільтона). Перша сторінка статті повинна завершуватися копірайтом автора в тому ж форматі, що наведено в даному шаблоні.

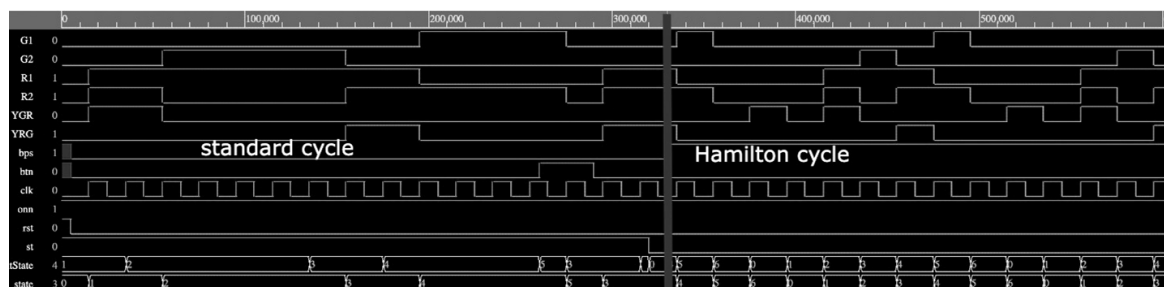


Рис.4.4 Результати моделювання режимів автомата

Результатом синтезу пристрою на ПЛІС FPGA XC3S500E-5fg320 та CPLD XC9572XL-10-TQ100 є автомат з 7 станами. Результати початкового синтезу наведено в таблиці 1. У таблиці 2 наведено результати після модифікації HDL опису та введенням додаткового діагностичного входу *bps*.

Ці результати показують, що додаткова діагностична логіка всередині автомата не збільшує кількість згенерованих тригерів (flips-flops), які використовуються для кодування станів. Відсутні латч-тригери (latches). Натомість запропоновані модифікації HDL опису збільшили використання основних елементів логіки (Basic Elements of Logic, BELs) для обох ПЛІС. Для FPGA зросло використання зрізів (Slices) і таблиць пошуку (Lookup Tables, LUTs) – з 16/30 до 18/35 відповідно. Для CPLD кількість використовуваних макрокомірок (Macrocells) зросла з 10 до 12. Середні додаткові апаратні витрати становлять 20%.

Аналіз швидкодії наведено в таблиці 3. Для CPLD жодних змін у мінімальному тактовому періоді та максимальній частоті виявлено не було. Для FPGA тактовий період збільшився з 3.900 нс до 4.112 нс, а максимальна частота зменшилася з 256.430 МГц до 243.188 МГц. Таким чином, зниження швидкодії становить менше 8%.

Таблиця 1. Результати початкового синтезу автомата для ПЛІС FPGA та CPLD

ПЛІС	Тригери	Латч- тригери	BELs	Slices/ LUTs	Макрокомірки
FPGA XC3S500E- 5fg320	10	0	30	16/30	
CPLD XC9572XL-10- TQ100	6	0	120		10

Таблиця 2. Результати синтезу автомата для ПЛІС FPGA та CPLD з введенням додаткового діагностичного входу *bps*

ПЛІС	Тригери	Латч- тригери	BELs	Slices/ LUTs	Макрокомірки
FPGA XC3S500E- 5fg320	10	0	36	18/35	
CPLD XC9572XL-10- TQ100	6	0	143		12

Таблиця 3. Порівняльні результати швидкодії автомата для FPGA та CPLD з додатковим діагностичним входом

ПЛІС	Мінімальний період тактового сигналу, нс	Максимальна частота, МГц	Мінімальний період тактового сигналу з додатковим входом, нс	Максимальна частота з додатковим входом, МГц
FPGA XC3S500E-5fg320	3.900	256.430	4.112	243.188
CPLD XC9572XL-10-TQ100	6.300	158.730	6.300	158.730

5. Висновки

У цій статті було вирішено проблему створення автоматного шаблону легкотестованого часового автомата Мура. Впровадження апаратної надлишковості є основою для проектування такого роду автоматів. Запропоновано спосіб модифікації опису автомата на мові Verilog. Запропонований підхід надає можливість встановити часовий автомат у довільний стан без синхронізуючих послідовностей та внутрішньої модифікації таймера за фіксовану кількість тактів синхросигнала. Це підвищує тестоздатність і спостережуваність цифрового пристрою, що дозволяє автоматизувати процес створення діагностичних експериментів.

Для ілюстрації запропонованих методів використано модель контролера світлофора. Початкова модель була розширена додатковим входом, який дозволяє переводити автомат в довільний стан. Результати моделювання підтвердили ефективність та працездатність методики. Результати синтезу в CAD XILINX ISE показали, що додаткові апаратні витрати становлять менше 20%, коли модель розширено додатковим діагностичним входом як для FPGA, так і для CPLD.

Наукова новизна даної роботи полягає у подальшій розробці підходів і методів створення легкотестованих HDL моделей часових автоматів та їх комбінування шляхом модифікації HDL опису. Такі методи можна інтегрувати в системи САПР, що дозволяє скоротити загальний час проектування та верифікації, а також подальшу діагностику.

Практична значущість роботи полягає у введенні HDL шаблону легкотестованого часового автомата Мура шляхом введення додаткових операторів if-else. Запропоновану методологію можна інтегрувати з іншими методами перевірки та тестування, такими як перевірка на основі асерцій, формальні методи та Universal Verification Methodology.

ЛІТЕРАТУРА

1. M. Miroshnyk, A. Shkil, E. Kulak, D. Rakhlis, A. Miroshnyk, N. Malahov Design Timed FSM With VHDL Moore Pattern. *Radio Electronics, Computer Science, Control*. 2020. Issue 14. P. 137-148. doi:10.15588/1607-3274-2020-2-14. <http://ric.zntu.edu.ua/article/view/208496>
2. A. Shkil, A. Miroshnyk, G. Kulak and K. Pshenychnyi. Assertion Based Design of Timed Finite State Machine. *IEEE East-West Design & Test Symposium (EWDTS), Batumi, Georgia*. 2021. P. 1-4. doi: 10.1109/EWDTS52692.2021.9581046. <https://ieeexplore.ieee.org/document/9581046>
3. M. Miroshnyk, Y. Pakhomov, E. German, A. Shkil, E. Kulak and D. Kucherenko. Design automation of testable finite state machines. *IEEE East-West Design & Test Symposium (EWDTS), Novi Sad, Serbia*. 2017. P. 1-6. doi: 10.1109/EWDTS.2017.8110034. <https://ieeexplore.ieee.org/document/8110034>

4. Tvardovskii, A.S., Yevtushenko, N.V. Deriving Homing Sequences for Finite State Machines with Timed Guards. *Aut. Control Comp. Sci.* Issue 55. 2021. P. 738–750 doi: 10.3103/S0146411621070154.
https://www.researchgate.net/publication/347911837_Deriving_Homing_Sequences_for_Finite_State_Machines_with_Timed_Guards
5. V. A. Pedroni. Finite state machines in hardware: theory and design (with VHDL and SystemVerilog). MIT Press. 2013.
<https://direct.mit.edu/books/book/4016/Finite-State-Machines-in-HardwareTheory-and-Design>
6. R. Alur, D. L. Dill. A theory of timed automata. *Theoretical Computer Science*. vol. 126, no. 2. 1994. PP. 183-235.
<https://www.sciencedirect.com/science/article/pii/0304397594900108>
7. М. А. Мирошник. Проектирование диагностической инфраструктуры вычислительных систем и устройств на ПЛИС: монография, Харків: ХУПС, 2012.
<http://lib.kart.edu.ua/bitstream/123456789/5759/1/%D0%9D%D0%B0%D0%B2%D1%87%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9%20%D0%BF%D0%BE%D1%81%D1%96%D0%B1%D0%BD%D0%B8%D0%BA.pdf>
8. G. Wagner, An abstract state machine semantics for discrete event simulation 2017 *Winter Simulation Conference (WSC)*. Las Vegas, NV, USA. 2017. P. 762-773, doi: 10.1109/WSC.2017.8247830.
<https://ieeexplore.ieee.org/document/8247830>
9. Z. Navabi. Digital System Test and Testable Design. Springer New York, NY. 2010. doi: 10.1007/978-1-4419-7548-5.
<https://link.springer.com/book/10.1007/978-1-4419-7548-5>
10. J.E. Hopfort, R. Motwani, J.D. Ullman. Introduction to Automata Theory, Languages, and Computation (3rd. ed.), Addison Wesley Longman Publishing Co., Inc., 2006.
<https://www-2.dc.uba.ar/staff/becher/Hopcroft-Motwani-Ullman-2001.pdf>
11. D. Bresolin, A. Tvardovskii, N. Yevtushenko, T. Villa, M. Gromov. Minimizing Deterministic Timed Finite State Machines. IFAC-PapersOnLine, 2018. P. 486-492, doi:10.1016/j.ifacol.2018.06.344.
<https://www.sciencedirect.com/science/article/pii/S2405896318306748>
12. M. Zhigulin, N. Yevtushenko, S. Maag and A. Cavalli. FSM-Based Test Derivation Strategies for Systems with Time-Outs. 2011 11th International Conference on Quality Software, Madrid, Spain. 2011. P. 141-149, doi: 10.1109/QSIC.2011.30.
<https://ieeexplore.ieee.org/document/6004321>
13. А. С. Шкиль, Г. П. Фастовец, А. С. Серокурова, "Автоматизация поиска ошибок проектирования в HDL-моделях конечных автоматов", АСУ и приборы автоматики. 2014. 168, с. 43-52.
<https://openarchive.nure.ua/items/5b0b4a07-c457-41a4-ae8-79256ac9b730>
14. M. Miroshnyk, Y. Pakhomov, E. German, A. Shkil, E. Kulak and D. Kucherenko. Design automation of testable finite state machines 2017 *IEEE East-West Design & Test Symposium (EWDTS)*, Novi Sad, Serbia. 2017. P. 1-6, doi: 10.1109/EWDTS.2017.8110034.
<https://ieeexplore.ieee.org/document/8110034>
15. М. А. Мірошник, М. С. Курцев, Автоматизація проектування вбудованих систем і програмних засобів на ПЛІС мовою опису апаратури: Навч. посібник, УкрДУЗТ, 2021
<http://lib.kart.edu.ua/handle/123456789/7162>

REFERENCES

1. A. Shkil, A. Miroshnyk, G. Kulak and K. Pshenychnyi, "Assertion Based Design of Timed Finite State Machine," 2021 IEEE East-West Design & Test Symposium (EWDTS), Batumi, Georgia, 2021, pp. 1-4, doi: 10.1109/EWDTS52692.2021.9581046.
<https://ieeexplore.ieee.org/document/9581046>
2. M. Miroshnyk, Y. Pakhomov, E. German, A. Shkil, E. Kulak and D. Kucherenko, "Design automation of testable finite state machines," 2017 IEEE East-West Design & Test Symposium (EWDTS), Novi Sad, Serbia, 2017, pp. 1-6, doi: 10.1109/EWDTS.2017.8110034.
<https://ieeexplore.ieee.org/document/8110034>
3. Tvardovskii, A.S., Yevtushenko, N.V. Deriving Homing Sequences for Finite State Machines with Timed Guards. *Aut. Control Comp. Sci.* 55, 2021, 738–750 doi: 10.3103/S0146411621070154.

https://www.researchgate.net/publication/347911837_Deriving_Homing_Sequences_for_Finite_State_Machines_with_Timed_Guards

4. V. A. Pedroni, Finite state machines in hardware: theory and design (with VHDL and SystemVerilog), MIT Press, 2013.
<https://direct.mit.edu/books/book/4016/Finite-State-Machines-in-HardwareTheory-and-Design>
5. R. Alur, D. L. Dill, "A theory of timed automata," Theoretical Computer Science, vol. 126, no. 2, 1994, pp. 183-235.
<https://www.sciencedirect.com/science/article/pii/0304397594900108>
6. M. A. Miroshnyk, Diagnostic infrastructure of computing systems and devices design on FPGA: monograph, KhUPS, 2012. [in Ukrainian]
<http://lib.kart.edu.ua/bitstream/123456789/5759/1/%D0%9D%D0%B0%D0%B2%D1%87%D0%B0%D0%BB%D1%8C%D0%BD%D0%B8%D0%B9%20%D0%BF%D0%BE%D1%81%D1%96%D0%B1%D0%BD%D0%B8%D0%BA.pdf>
7. G. Wagner, "An abstract state machine semantics for discrete event simulation," 2017 Winter Simulation Conference (WSC), Las Vegas, NV, USA, 2017, pp. 762-773, doi: 10.1109/WSC.2017.8247830.
<https://ieeexplore.ieee.org/document/8247830>
8. Z. Navabi, Digital System Test and Testable Design, Springer New York, NY, 2010. doi: 10.1007/978-1-4419-7548-5.
<https://link.springer.com/book/10.1007/978-1-4419-7548-5>
9. J.E. Hopfort, R. Motwani, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation (3rd. ed.), Addison Wesley Longman Publishing Co., Inc., 2006.
<https://www-2.dc.uba.ar/staff/becher/Hopcroft-Motwani-Ullman-2001.pdf>
10. D. Bresolin, A. Tvardovskii, N. Yevtushenko, T. Villa, M. Gromov, "Minimizing Deterministic Timed Finite State Machines," IFAC-PapersOnLine, 2018, pp. 486-492, doi:10.1016/j.ifacol.2018.06.344.
<https://www.sciencedirect.com/science/article/pii/S2405896318306748>
11. M. Zhigulin, N. Yevtushenko, S. Maag and A. Cavalli, "FSM-Based Test Derivation Strategies for Systems with Time-Outs," 2011 11th International Conference on Quality Software, Madrid, Spain, 2011, pp. 141-149, doi: 10.1109/QSIC.2011.30.
<https://ieeexplore.ieee.org/document/6004321>
12. A. S. Shkil, G. P. Fastovets, A. S. Serokurova, "Automation of search for design errors in HDL-models of finite state machines ", MANAGEMENT INFORMATION SYSTEM AND DEVICES, vol. 168, pp. 43-52, 2014. [in Russian]
<https://openarchive.nure.ua/items/5b0b4a07-c457-41a4-aef8-79256ac9b730>
13. M. Miroshnyk, Y. Pakhomov, E. German, A. Shkil, E. Kulak and D. Kucherenko, "Design automation of testable finite state machines," 2017 IEEE East-West Design & Test Symposium (EWDTS), Novi Sad, Serbia, 2017, pp. 1-6, doi: 10.1109/EWDTS.2017.8110034.
<https://ieeexplore.ieee.org/document/8110034>
14. M. A. Miroshnyk, M. S. Kurtcev, Automation of the design of embedded systems and software on FPGAs in the hardware description languages: textbook, UkrSURT, 2021. [in Ukrainian]
<http://lib.kart.edu.ua/handle/123456789/7162>
15. M. A. Miroshnyk, M. S. Kurtcev, Automation of the design of embedded systems and software on FPGAs in the hardware description languages: textbook, UkrSURT, 2021. [in Ukrainian]
<http://lib.kart.edu.ua/handle/123456789/7162>

- Miroshnyk Maryna** *Doctor of Technical Sciences, Professor
Professor of theoretical and applied systems
engineering department, V. N. Karazin Kharkiv National University, Svobody Sq., 4,
Kharkiv, Ukraine, 61022
e-mail: m.miroshnyk@karazin.ua
<https://orcid.org/0000000222312529>*
- Pshenychnyi Kyrylo** *Post graduate student of design automation department, Kharkiv National University of
Radioelectronics, Nauky Ave., 14, Kharkiv, Ukraine, 61166
e-mail: kyrylo.pshenychnyi@nure.ua
<https://orcid.org/0009000707996604>*
- Shafranskyi Andrei** *graduate student of the Department of theoretical and applied systems engineering,
Kharkiv National University named after V. N. Karazin Kharkiv National University,
Svobody Sq., 4, Kharkiv, Ukraine, 61022
e-mail: shafranskyi.andrei@student.karazin.ua
<https://orcid.org/0009-0004-7725-3556>*
- Shkil Oleksandr** *Doctor of Philosophy, Associate professor
Associate professor of design automation department, Kharkiv National University of
Radioelectronics, Nauky Ave., 14, Kharkiv, Ukraine, 61166
e-mail: oleksandr.shkil@nure.ua
<https://orcid.org/0000000310713445>*

Models of Testable Timed Moore Finite State Machines

The work proposes a method of designing testable digital devices in real time, presented in the form of finite state machines and described using hardware description languages (HDL).

Relevance. The relevance of the work lies in the possibility of diagnosing digital devices in real time during active operation.

Research methods. The main research method is the introduction of hardware redundancy in the form of an additional HDL code blocks in the device description code and additional fragments on the temporal state diagram. The proposed approach provides a way to set timed FSM into an arbitrary state without synchronizing sequences and internal timer modification within a fixed number of clock cycles. This increases the testability and observability of the digital device allowing to automate the process of diagnostic experiments creation.

Conclusions. The problem of testable real-time devices based on easy-to-test Moore FSM design has been solved. The proposed method allows to set the automata into an arbitrary state within a fixed time. This approach makes it possible to significantly simplify the process of device diagnostics.

Traffic light controller model was used to illustrate the proposed methods. The initial model was extended with an additional input that allows setting the automata into an arbitrary state. Simulation results confirmed the efficiency of the approach. The synthesis results in CAD XILINX ISE showed that hardware costs are less than 20% when the model is extended with an additional debug input for both FPGA and CPLD boards.

The scientific novelty of this paper lies in developing approaches and methods of creating testable HDL models of timed FSM and their combination by modifying the design HDL description. Such methods can be integrated into CAD systems which allows to decrease the overall time of design and verification.

The practical significance of the work is to introduces the HDL pattern of easy-tested timed Moore FSM by introducing additional if-else statements. The propose methodology can be integrated with other verification and testing technics such as assertion based verification, formal methods, and Universal Verification Methodology increasing the overall design reliability.

Keywords: *easy-to-test systems, design automation, finite state machines, verification, hardware description languages, Verilog.*