УДК 681.326

АВТОМАТИЗИРОВАННОЕ ПРОЕКТИРОВАНИЕ СИСТЕМ ЛОГИЧЕСКОГО УПРАВЛЕНИЯ С ИСПОЛЬЗОВАНИЕМ ШАБЛОНОВ АВТОМАТНОГО ПРОГРАММИРОВАНИЯ

ШКИЛЬ А.С., КУЛАК Э.Н., ФИЛИППЕНКО И.В., КУЧЕРЕНКО Д.Е., ГОГА М.В.

Предлагается использовать шаблоны автоматного программирования для проектирования устройств логического управления на основе конечных автоматов. При проектировании конечного автомата на технологической платформе ПЛИС FPGA алгоритм функционирования представляется на языке описания аппаратуры VHDL, а синтез устройства осуществляется в САПР XILINX ISE. При проектировании конечного автомата на базе микроконтроллера семейства MCS 51 алгоритм функционирования описывается на подмножестве языка С в среде разработки Keil.

Ключевые слова: логическое управление, конечный автомат, граф переходов, микроконтроллер, ПЛИС, САПР, язык VHDL, язык С.

Key words: logical control, finite state machine, state diagram, microcontroller, PLD, CAD, VHDL language, C language.

1. Введение

На основе общей концепции построения систем автоматизированного управления в них всегда можно выделить управляющие устройства и управляемые объекты. Следуя этой концепции, системы управления на основе конечных автоматов (finite state machines) также можно разделить на две части:

– управляющую часть, ответственную за логику поведения – выбор выполняемых действий, зависящий от текущего состояния и входного воздействия, а также за переход в новое состояние;

– управляемую часть, ответственную за выполнение действий, выбранных для выполнения управляющей частью, и, возможно, за формирование некоторых компонентов входных воздействий для управляющей части – обратных связей.

Среди всего множества управляющих устройств можно выделить устройства логического управления, у которых управляющие воздействия (control value) представляются в двоичном алфавите. Поскольку для реализации управляющей части в таких устройствах, как правило, используются конечные автоматы, то они называются управляющими автоматами (УА). Подобные устройства широко применяются в системах Internet of Things.

Любое локальное цифровое устройство, реализующее алгоритм обработки информации или управления, может быть реализовано двумя способами: аппаратным или программно-аппаратным.

При аппаратном способе реализации заданный алгоритм представляется на языке описания аппаратуры (Hardware Description Language, HDL) и синтезируется инструментальными средствами систем автоматизированного проектирования (САПР) в ПЛИС (программируемые логические интегральные схемы) или ASIC (application-specific integrated circuit или интегральная схема специального назначения). Достоинством такого подхода является аппаратная гибкость (возможность реализовать любой алгоритм) и достаточно большое быстродействие. К недостаткам данного подхода можно отнести необходимость разработки интерфейса ввода-вывода для связи разрабатываемого устройства с внешними устройствами и сложности в реализации временных параметров.

При программно-аппаратном способе реализации алгоритм описывается на аппаратно-ориентированном языке программирования (например, на языке С со специальными библиотеками) с учетом аппаратной архитектуры, на которой будет реализовываться заданная программа. Это, как правило, различные семейства микроконтроллеров (МК). Достоинством данного подхода является наличие специальных аппаратно-ориентированных функций (таймеров контроллеров прерываний), а также аппаратно реализованных интерфейсов обмена с внешними устройствами. К недостаткам можно отнести меньшее быстродействие и ограничения, накладываемые аппаратной архитектурой на реализуемый алгоритм.

При описании алгоритма функционирования цифровых устройств логического управления в САПР цифровых устройств одним из стилей написания кода является стиль автоматного программирования. Суть автоматного программирования состоит в отделении описания логики поведения (при каких условиях необходимо выполнить те или иные действия) от описания его семантики (собственно смысла каждого из действий). В автоматном программировании в качестве базового используется понятие «состояние», а не «класс», «объект», а в качестве визуального представления алгоритма функционирования используется граф переходов.

Автоматные программы строго структурированы и в них выделены три вида функций: функции переходов, функции выходов, функции реализации задержек и перехода в новое состояние. Автоматные программы строго шаблонизированы с использованием операторов многопозиционного выбора (switch, case), условных операторов (if, select) и функций реализации таймера или фронта (синхросигнала Clk). Автоматные программы инвариантны к способу кодирования (языку про-

граммной реализации). Есть примеры автоматных программ на разных языках описания аппаратуры: на C, на JScript.

Таким образом, актуальной становится задача разработки единого шаблона языкового описания автоматных устройств логического управления в стиле автоматного программирования с учетом особенностей функционирования систем реального времени. Целью данного исследования является разработка шаблонов описания конечных управляющих автоматов в системах реального времени на языках описания аппаратуры и программирования с последующей схемной реализацией разработанных программных кодов.

2. Модель структурного автомата в системах реального времени

В качестве абстрактной модели дискретного устройства логического управления будем использовать конечный автомат, определяемый шестеркой $F =< X, A, Y, \delta, \lambda, a_1 >$, где $X = \{x_1, x_2, ..., x_m\}$ – множество букв входного алфавита; $A = \{a_1, a_2, ..., a_n\}$ – множество состояний автомата; $Y = \{y_1, y_2, ..., y_r\}$ – множество букв выходного алфавита; $\delta(a_i, x_k) = a_j$ – функция переходов автомата, $\lambda(z_i, x_k) = y_\alpha$ – функция выходов автомата, a_1 – начальное состояние автомата. Одной из форм представления модели абстрактного автомата является граф переходов (state diagram).

При переходе от модели абстрактного автомата к схемной реализации устройства управления используется модель структурного автомата. В структурных моделях входной алфавит преобразуется в множество входных воздействий (set input values), алфавит состояний преобразуется во множество внутренних переменных (internal variables), а выходной алфавит – во множество выходных значений (set output values). При этом все три множества являются конечными. Общепринятая модель структурного автомата – модель Хаффмена (рис. 1), состоит из комбинационного и последовательностного компонентов.

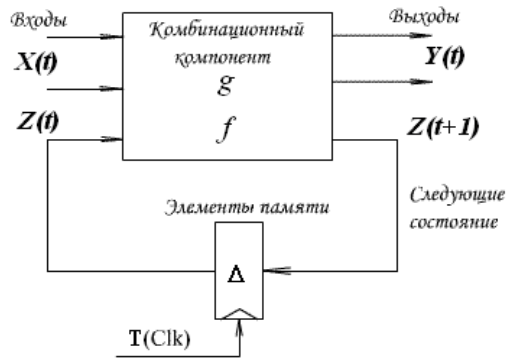


Рис. 1. Модель структурного автомата

Последовательностный компонент содержит элементы памяти, такие как синхронные триггеры, которые запоминают состояние и позволяют изменять его синхронно. Комбинационный компонент состоит из логических элементов, реализующих две логические функции: функцию выходов g, которая вычисляет значения выходных сигналов, и функцию переходов f, которая вычисляет новые значения элементов памяти (т.е. значения следующего состояния) [2].

Структурный автомат функционирует в автоматном времени, которое измеряется в тактах {t, t + 1, t + 2 ...}, т.е. автомат переходит из одного состояния в другое за один автоматный такт. Каждое состояние абстрактного автомата a_i кодируется вектором внутренних переменных z_i, и в аналитическом виде модель структурного автомата имеет вид $Y(t) = g(X(t), Z(t))$, $Z(t+1) = f(X(t), Z(t))$, где g – функция выходов структурного автомата, f – функция переходов структурного автомата. При этом $Z(t+1) \equiv Z(t)$, но в следующем автоматном такте.

Устройства логического управления, построенные на основе конечных автоматов, функционируют в автоматном времени, которое измеряется в автоматных тактах, т.е. дискретных отрезках времени, за которое автомат переходит из одного состояния в другое. Длительность автоматного такта в структурных автоматах, как правило, определяется частотой синхросигнала Clk. Но, с другой стороны, устройства логического управления являются устройствами реального времени и переход из одного управляющего состояния в другое определяется временными параметрами алгоритма функционирования устройства.

В программировании при анализе программ, описывающих автоматные системы реального времени, широко применяются временные автоматы [3]. В реализации временного автомата каждому состоянию ставятся в соответствие часы, которые определяют интервалы метрического (непрерывного) времени. Переходы из состояния в состояние во временном автомате происходят мгновенно, но автомат не переходит в новое состояние, пока часы не покажут возможность перехода (истечет заданный интервал метрического времени). Данная модель соответствует конечному автомату Мура.

Для внесения реального времени в описание структурного автомата предлагается использовать расширенную функцию переходов $Z(t+1) = f(X(t), Z(t), T)$, в которой аргументом выступает реальное время. При таком подходе аргумент T соответствует показаниям часов временного автомата. Граф переходов автомата, где учитывается параметр времени, будем называть

темпоральный граф переходов (temporal state diagram). В этом графе каждому состоянию ставится в соответствие задержка T_i, определяющаяся числом автоматных тактов, в течение которых автомат находится в данном состоянии. Выходные сигналы, соответствующие данному состоянию автомата, в течение этого времени не меняются (для автомата Мура). Проблема пересчета метрического времени в автоматные такты является достаточно важной задачей, но в данной работе подробно не рассматривается.

Как уже упоминалось выше, граф переходов конечного автомата является визуальным представлением модели абстрактного автомата [2]. При переходе к модели структурного автомата, которая используется в САПР, необходимо указать следующие параметры: способ кодирования состояний, разрядность входных и выходных переменных, вид синхронизации и, для временного автомата, задержку в каждом состоянии. Такой граф переходов, по аналогии с содержательной граф-схемой алгоритма (ГСА), будем называть содержательным графом переходов конечного автомата [4]. Таким образом, содержательный темпоральный граф переходов является не только визуальным представлением алгоритма работы структурного автомата, но и его полной математической моделью. Содержательный граф переходов в САПР описывается шаблонами специального вида (patterns) на языках программирования или описана аппаратуры, что будет показано ниже.

Задержка в каждой вершине темпорального графа переходов реализуется через петлю, условиями для которой являются подсчет числа тактов Clk, что схемно реализуется счетчиком в ПЛИС или таймером с прерыванием в МК. На рис. 2 приведено соответствие условного обозначения задержек в вершинах темпорального графа переходов автомата Мура без петель и с петлями.

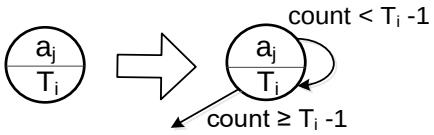
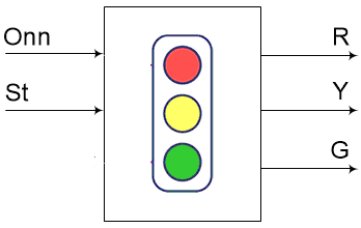


Рис. 2. Реализация задержек в темпоральном графе переходов автомата Мура

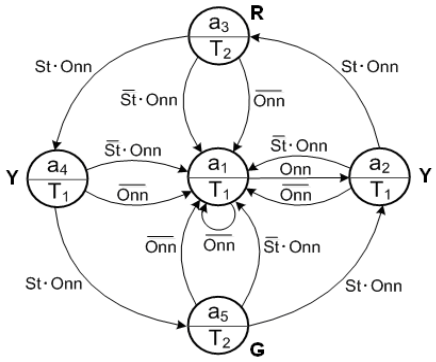
Задержка T_i осуществляется многократным переходом из состояния в это же состояние, при этом число переходов определяется числом тактов задержки. Значение счетчика count сравнивается с T_i-1, поскольку при переходе в состояние a_j автомат один такт находится в нем до проверки count, и чтобы задержка была точно T_i тактов, необходимо еще T_i-1 тактов повторения.

В качестве примера устройства логического управления рассмотрим устройство управления дорожным светофором, которое является частью автоматизированной системы управления дорожным движением. В качестве управляющего автомата используется модель Мура.

На рис. 3,а показан интерфейс данного управляющего автомата. Множество входных сигналов X = {Onn, St}, где Onn = {0, 1} – сигнал включения светофора, St = {0, 1} – сигнал запуска стандартного цикла работы светофора. Множество выходных сигналов Y = {R, Y, G}, где R = {Red} – сигнал включения красного света светофора, Y={Yellow} – сигнал включения желтого света светофора, G = {Green} – сигнал включения зеленого света светофора.



а



б

Рис. 3. Интерфейс и граф переходов автомата Мура для устройства управления светофором

Данное устройство реализует два цикла работы светофора:

– ночной (аварийный) цикл – мигание желтого света светофора с интервалом (задержкой) T_1;

– стандартный цикл работы светофора {Y - R - Y - G -Y}, при этом задержка R, G будет T_2 (5 тактов), а задержка Y будет T_1 (один такт).

На рис. 3,б представлен темпоральный граф переходов данного устройства управления. На рассматриваемом графе (рис. 2,б) все сигналы являются одноразрядными, поэтому их разрядность не указывается, способ кодирования состояний определяется в настройках САПР, синхронизация осуществляется по переднему фронту.

3. Проектирование конечного автомата на ПЛИС FPGA

При автоматизированном проектировании цифровых устройств на ПЛИС для описания алгоритма функционирования, как правило, используются языки описания аппаратуры (HDL). При описании алгоритма функционирования на HDL важно, чтобы разработанный код не выходил за пределы синтезируемого подмножества конкретного HDL. Одно- и двухпроцессные шаблоны описания HDL-моделей конечных автоматов широко применяются в САПР ПЛИС [5].

Но попытка применить функцию реализации задержки в конструкции process при назначении нового состояния State <= NextState; в двухпроцессном шаблоне для описания темпорального графа переходов (рис. 4) закончилась неудачей – разработанный VHDL-код не синтезировался, хотя система моделирования Active-HDL показала правильный результат работы УА.

```
function TICK (T : integer; T_comp:integer)
          return integer is
          variable compare:integer:=T_comp;
  begin
          if(T_comp = T) then compare:=1;
          else       compare := T_comp+1;
          end if;
          return compare;
end TICK;
```

Рис. 4. VHDL-функция реализации задержек в двухпроцессом шаблоне

Поэтому для описания VHDL-модели рассматриваемого УА был выбран однопроцессный шаблон, что соответствует рекомендациям в [6].

Вне процесса p1 объявляются константные значения задержек T1 и T2, задаваемых в периодах Clk. Также объявляется сигнал count для реализации счетчика периодов Clk. При активном сигнале сброса Reset=1 автомат устанавливается в начальное состояние a1, а счетчик периодов Clk обнуляется. При описании перехода из состояния a_i с задержкой T_i с каждым новым тактом автомат переводится в исходное a_i состояние, и на этом же переходе счетчик увеличивается на единицу. Это должно происходить при работе автомата до тех пор, пока значение счетчика count не станет равным T_i-1. При переходе из состояния a_i в каждое следующее состояние $a_j \neq a_i$ счетчик должен обнуляться, чтобы при попадании в следующее состояние, в котором необходим новый цикл счета, он был в исходном состоянии.

Если автомат попадает в несанкционированное состояние, он должен вернуться в начальное состояние a1. Управляющие (выходные) сигналы G, Y и R формируются вне процесса с помощью параллельного оператора условного назначения, что позволяет получить при синтезе комбинационную функцию выходов и значительно сокращает аппаратурные затраты.

На рис. 5 представлен фрагмент VHDL-модели, соответствующей темпоральному графу переходов, представленному на рис. 2, б.

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.STD_LOGIC_UNSIGNED.all;
entity TFSM is
          port ( Clk, Reset, St, Onn: in STD_LOGIC;
                    R, G, Y: out STD_LOGIC);
end;
architecture TFSM1 of TFSM is
  type State_type is (a1, a2, a3, a4, a5);
  signal State: State_type;
  signal count: STD_LOGIC_vector (3 downto 0 );
-- periods CLK counter
  constant T1: STD_LOGIC_vector (3 downto 0):=
"0001";
  constant T2: STD_LOGIC_vector (3 downto 0):=
"0101";   -- delay T1 = 1and T2=5  CLK periods
begin
p1: process (Clk, reset)      -- transition function
begin
if Reset='1' then  State <= a1; count <= (others => '0') ;
elsif Clk'event and Clk = '1' then
case State is
when a1 =>
     if count < T1 - 1 then  State <= a1; count <= count +
1;
     elsif  Onn='1'  then State <= a2; count<= (others
=>'0');
     else  State <= a1; count <= (others => '0');
     end  if;
when a2 =>
      if count < T1 - 1 then State <= a2; count <= count +
1;
      elsif Onn='0'  then State <= a1; coun <= (oth-
ers=>'0');
      elsif  St='0'   then State <= a1; count <= (others=>'0');
      else State <= a3; count <= (others => '0');
      end if;
 when a3 =>
      if count < T2 - 1 then State <= a3; count <= count +
1;
      elsif  Onn='0'  then State <= a1; count<= (oth-
ers=>'0');
      elsif  St='0'  then State <= a1; count <= (others =>'0')
;
      else State <= a4; count <= (others => '0') ;
      end if;
      . . .
when others => State <= a1;
end case;
end if;
end process;
-- output signals (output function)
R<='1'   when   State=a3  else '0';
Y<='1'   when   State=a2 or State=a4  else  '0';
G<='1'   when   State=a5  else '0';
end;
```

Рис. 5. Фрагмент VHDL-модели автомата Мура устройства управления светофором

На рис. 6 приведена временная диаграмма (waveform) результатов моделирования работы рассматриваемого УА в системе Active-HDL ALDEC.

Установка автомата в начальное состояние осуществляется в течение первого автоматного такта, дальше приведены различные варианты развития событий. Видно, что выходной управляющий сигнал формируется сразу, как только автомат переходит в новое состояние, отличное от a1. Длительность сигнала Y составляет 1 такт, сигналов G и R – 5 тактов.

Для обеспечения необходимой длительности автоматного такта следует использовать стандартный делитель частоты (счетчик).

В данном примере длительность одного автоматного такта t равна одной минуте (t = 60 с). Период синхросигнала с выхода генератора тактовой частоты определяется как $Tclk = 1/Hclk$ (с), где Hclk – частота синхросигнала (Гц). Для того чтобы определить, во сколько раз необходимо понизить частоту (Hclk) или увеличить период (Tclk), необходимо найти отношение $t/Tclk$, которое определит значение коэффициента пересчета счетчика, и рассчитать n из выражения $2^{n-1} < \frac{t}{Tclk} \le 2^n$ для определения разрядности счетчика.

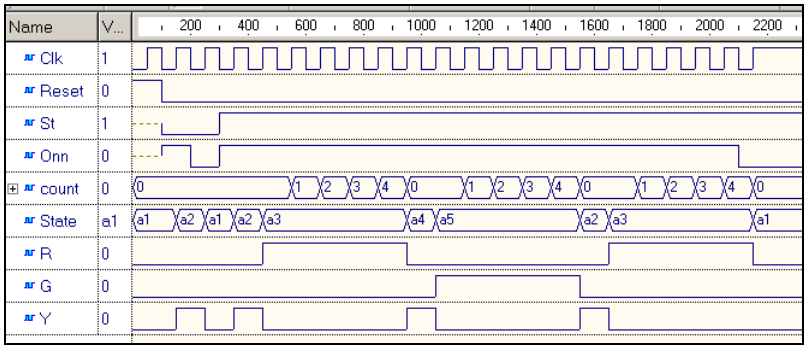


Рис. 6. Результаты моделирования устройства управления светофором

В таблице приведены результаты синтеза рассматриваемого УА в ПЛИС в САПР XILINX ISE для устройства: плата Spartan 3E, микросхема FPGA XC3S500E, Package FG 320 (xc3s500e-4fg320).

Результаты синтеза

Device Utilization Summary (estimated values)		
Logic Utilization	**Used**	**Available**
Number of Slices	7	4656
Number of Slice Flip Flops	6	9312
Number of 4 input LUTs	13	9312
Number of bonded IOBs	7	232
Number of GCLKs	1	24

Ниже представлен протокол синтеза данного устройства:

Synthesizing Unit <TFSM>;
Analyzing FSM <FSM_0> for best encoding;
Optimizing FSM <State/FSM> on signal <State[1:3]> with sequential encoding.
State | Encoding a1 |000 a2 |001 a3 |010 a4|011 a5|100.

Синтезируется структура, состоящая из двух блоков: управляющий автомат (3 триггера Dff, комбинационные схемы, реализующие функции переходов и выходов) и счетчик на 3 триггера Dff. Приведенные результаты подтверждают корректность разработанной HDL-модели.

4. Проектирование конечного автомата на микроконтроллере MCS 51

Для задач логического управления наиболее оптимальными являются микроконтроллеры, которые имеют в своем составе необходимую периферию при невысокой стоимости. Одним из самых распространенных МК для указанных задач при небольшой вычислительной мощности является микроконтроллер Intel MCS 51. Для программной реализации алгоритма функционирования используем процедурное подмножество языка С в среде разработки Keil.

Такое подмножество позволяет реализовать классический автоматный шаблон с функциями переходов, выходов и назначением нового состояния, каждая из которых реализуется отдельным оператором switch [7].

Программа состоит из бесконечного цикла, в котором реализуются заданные функции переходов и выходов while(1) { StateTransition(); }, и функции реализации прерывания через таймер InitTimer0(); .

Для инициализации таймера задается маска режима работы TMOD |= 0x01; .

В обработчике прерываний Timer0_isr(void) interrupt 1 using 1 задаются значения таймера для отсчета 1 миллисекунды:

TH0=(65536-500)/256; TL0=(65536-500)*256.

Отсчет времени нахождения в определенном состоянии DelayMs=T_i задается в миллисекундах (T1=1000, T2=5000), что аналогично одному и пяти тактам работы автомата. Выдача сигналов на включение соответствующих световых индикаторов выполняется через порты микроконтроллера, которым поставлены в соответствие переменные:

sbit LED_R=P1^0; sbit LED_Y=P1^1;
sbit LED_G=P1^2; .

На рис. 7 приведен фрагмент микроконтроллерной программы на языке С в форме автоматного шаблона.

```
#include<reg52.h>
sbit LED_R=P1^0;                    /* Description ports */
sbit LED_Y=P1^1;
sbit LED_G=P1^2;
          . . .
#define T1 1000              /* Description delays */
#define T2 5000
          . . .
unsigned char state = A1;
unsigned char nextState;
unsigned int DelayMs=1000;
          . . .
void InitTimer0(void)                    /* Timer init */
{ TMOD |= 0x01;              EA=1; ET0=1;  TR0=1; }
void Timer0_isr(void) interrupt 1 using 1   /* Interrupt
func*/
{
 TH0=(65536-500)/256;
 TL0=(65536-500)*256;              /*1ms Square Wave */
 DelayMs --;
 if (DelayMs ==0)
  {
   state = nextState;
   if ((state == A3) || (state ==  A5)) { DelayMs = T2; }
   else { DelayMs = T1; }
  }
}
void main (void)                          /*begin program  /
{
          InitTimer0();
          while(1) { StateTransition(); }
}
void StateTransition() /*Transition and outputs function  */
{
 switch(state)                            /*Description state*/
  {
   case A1: if(Onn == 1) { nextState = A2; }
              else { nextState = A1; } break;
   case A2: if(St == 0 || Onn == 0) { nextState = A1;}
              else if (ST == 1) { nextState = A3; } break;
   case A3: if(ST == 0 || Onn == 0) { nextState = A1; }
              else if (ST == 1) { nextState = A4; } break;
   . . .
  }
 switch(state)     /* Description output signals*/
  {
   case A2: LED_R = 1;  LED_Y = 0; LED_G = 0;
break;
   case A3: LED_R = 0; LED_Y = 1; LED_G = 0;
break;
   case A4: LED_R = 0; LED_Y = 0; LED_G = 1;
break;
          . . .
   default: LED_R = 0; LED_Y = 0; LED_G = 0;
break;
  }
}
```

Рис. 7. Программа на языке С для программирования МК MCS 51

Данный программный код был промоделирован в САПР Proteus, но ограниченный объем статьи не позволяет привести электрическую принципиальную схему устройства и результаты ее моделирования.

5. Заключение

В результате проведенных исследований было показано, что использование стиля автоматного программирования и шаблонов описания конечных автоматов (finite state machines patterns) эффективно при написании программного кода как на языке описания аппаратуры VHDL, так и на языке программирования С. Схемная реализация разработанного кода инструментальными средствами соответствующих САПР подтвердила работоспособность предложенного метода.

Определенным положительным аспектом предложенного подхода является возможность верификации разработанного кода и диагностирования полученных схем на основе методик экспериментов над конечными автоматами путем разных вариантов обхода графа переходов УА, что значительно сокращает объем диагностического эксперимента [8].

Практическая ценность полученных результатов состоит в том, что авторами предложены шаблоны описания алгоритмов функционирования конечных автоматов в системах логического управления реального времени на языках VHDL и С, которые могут быть использованы начинающими проектировщиками цифровых систем логического управления, а также студентами специальности «Компьютерная инженерия» ХНУРЕ.

Литература: 1. *Шалыто А.А.* Автоматное программирование / Н.И. Поликарпова, А.А. Шалыто. Спб.: Питер, 2011. 167 с. **2.** *Baranov S.* Logic and System Design of Digital Systems / S. Baranov. Tallinn: TUT Press, 2008. 267 p. **3.** *Alur R.* A theory of timed automata / R. Alur, D. L. Dill // Theoretical Computer Science. 1994. V.126, № 2. P. 183-235. **4.** *Шкиль А.С.* Модель процесса перехода от содержательного графа микропрограммы к графу автомата / А.С. Шкиль, В.И. Хаханов, Е.В. Ковалев // АСУ и приборы автоматики. 2000. Вып. 112. С. 112-120. **5.** *Shkil A.S.* Design automation of easy-tested digital finite state machines / M.A. Miroschnyk, Y.V. Pakhomov, A.S. Shkil, E.N. Kulak, D.Y. Kucherenko // Radio Electronics, Computer Science, Control. 2018. №2. P. 117-124. **6.** *Haskell R.* Digital De-

sign Using Digilent FPGA Boards - VHDL / Active-HDL Edition / Richard E. Haskell, Darrin M. Hanna. LBE Books Rochester Hills, MI, 2009. 381 p. **7.** *Матюшин А.О.* Программирование микроконтроллеров: Стратегия и тактика / А.О. Матюшин. М.: ДМК Пресс, 2017. 356 с. **8.** *Тоценко В.Г.* Алгоритмы технического диагностирования цифровых устройств / В.Г. Тоценко. М.: Радио и связь, 1985. 240 с.

Transliterates bibliography:

1. *Shalyto A.A.* Avtomatnoe programmirovanie / N.I. Polikarpova, A.A. Shalyto. Spb.: Piter, 2011. 167 s.

2. *Baranov S.* Logic and System Design of Digital Systems / S. Baranov. Tallinn: TUT Press, 2008. 267 p.

3. *Alur R.* A theory of timed automata / R. Alur, D. L. Dill // Theoretical Computer Science. 1994. V.126. № 2. P. 183-235.

4. *Shkil' A.S.* Model' processa perehoda ot soderzhatel'nogo grafa mikroprogrammy k grafu avtomata / A.S. Shkil', V.I. Hahanov, E.V. Kovalev // ASU i pribory avtomatiki. 2000. Vyp. 112. S. 112-120.

5. *Shkil A.S.* Design automation of easy-tested digital finite state machines / M.A. Mirosshnyk, Y.V. Pakhomov, A.S. Shkil, E.N. Kulak, D.Y. Kucherenko // Radio Electronics, Computer Science, Control. 2018. №2. P. 117-124.

6. *Haskell R.* Digital Design Using Digilent FPGA Boards - VHDL / Active-HDL Edition / Richard E. Haskell, Darrin M. Hanna. LBE Books Rochester Hills, MI, 2009. 381 p.

7. *Matjushin A.O.* Programmirovanie mikrokontrollerov: Strategija i taktika / A.O. Matjushin. M.: DMK Press, 2017. 356 s.

8. *Tocenko V.G.* Algoritmy tehnicheskogo diagnostirovanija cifrovyh ustrojstv / V.G. Tocenko. M.: Radio i svjaz', 1985. 240 s.

Поступила в редколлегию 09.06.2018

Рецензент: д-р техн. наук, проф. Кривуля Г.Ф.

Шкиль Александр Сергеевич, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: диагностика цифровых систем, дистанционное образование. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. 702-13-26.

Кулак Эльвира Николаевна, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: автоматизированное проектирование цифровых автоматов, языки описания аппаратуры. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. 702-13-26.

Филиппенко Инна Викторовна, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: проектирование цифровых устройств на базе микроконтроллеров, цифровые фильтры. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. 702-13-26.

Кучеренко Дария Ефимовна, канд. техн. наук, доцент кафедры АПВТ ХНУРЭ. Научные интересы: языки описания аппаратуры, экспертные системы, техническая диагностика, нечеткая логика. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. 702-13-26.

Гога Максим, студент ХНУРЭ. Научные интересы: проектирование микроконтроллерных систем. Адрес: Украина, 61166, Харьков, пр. Науки, 14, тел. 702-13-26.

Shkil Alexander Sergeevich, PhD, Associate Professor, Design Automation Department, KNURE. Scientific interests: diagnostics of digital systems, distance education. Address: Ukraine, 61166, Kharkiv, Nauka Avenue, 14, tel. 702-13-26.

Kulak Elvira Nikolaevna, PhD, Associate Professor, Design Automation Department, KNURE. Scientific interests: automated design of digital machines, HDL. Address: Ukraine, 61166, Kharkiv, Nauka Avenue, 14, tel. 702-13-26.

Filippenko Inna Victorovna, PhD, Associate Professor, Design Automation Department, KNURE. Scientific interests: design based on microcontrollers, digital filters. Address: Ukraine, 61166, Kharkiv, Nauka Avenue, 14, tel. 702-13-26.

Kucherenko Dariia Yefimovna, PhD, Associate Professor, Design Automation Department, KNURE. Scientific interests: hardware description languages, expert systems, technical diagnosis, fuzzy logic. Address: Ukraine, 61166, Kharkiv, Nauki Avenue, 14, tel. 702-13-26.

Goga Maksim, student, KNURE. Scientific interests: design microcontroller systems. Address: Ukraine, 61166, Kharkiv, Nauka Avenue, 14, tel. 702-13-26.