УДК: 620.9:658.011.56

А.Н. Рыбалев, Д.А. Теличенко, В.Ю. Косицин, Р.А. Ахметшин, А.В. Белоусова

РАЗРАБОТКА И ИССЛЕДОВАНИЕ АЛГОРИТМОВ И ПРОГРАММ УПРАВЛЕНИЯ ИСПОЛНИТЕЛЬНЫМИ МЕХАНИЗМАМИ СИСТЕМ АВТОМАТИЧЕСКОГО РЕГУЛИРОВАНИЯ

В статье рассматриваются алгоритмы формирования управляющих воздействий на исполнительные механизмы постоянной скорости движения рабочего органа в составе систем автоматического управления. Предложены способы программной реализации широтно-импульсной модуляции. Приведены результаты моделирования в Simulink Matlab и экспериментов на лабораторном стенде.

The article considers algorithms for the formation of the control action on the actuators with a constant rate of movement of the working body composed of automatic control systems. The several ways to program implementation of pulse-width modulation proposed. The simulation results in the Simulink Matlab and experiments on a laboratory stand are given.

Введение и постановка задачи

В различных отраслях промышленности и сельского хозяйства системы автоматического регулирования технологических параметров в качестве исполнительного механизмов включают механизмы постоянной скорости. Они приводят в движение разнообразные регулирующие клапаны, задвижки, заслонки, шиберы, изменяя тем самым подачу среды, теплоносителя, реагентов и т.д. Управляются такие механизмы двумя дискретными сигналами (командами): «больше» и «меньше».

Наиболее часто в системах применяется тот или иной вариант пропорционально-интегрально-дифференциального (ПИД) закона регулирования. Настройка регулятора производится различными методами, однако в большинстве случаев используются непрерывные модели объекта, исполнительного механизма и системы в целом. Рассчитанный ПИД-регулятор практически реализуется «связкой» программного алгоритма с исполнительным механизмом. Поскольку механизм представляет собой, по сути, звено интегрирующего типа, для получения ПИД-закона программный алгоритм должен в общем случае формировать воздействие по ПДД^2-закону. Для преобразования выходного сигнала программного алгоритма в дискретные команды управления механизмом обычно применяется широтно-импульсная модуляция (ШИМ).

Построенная таким образом «квазинепрерывная» система автоматического управления лишь приближенно воспроизводит поведение исходной непрерывной модели: управление тем ближе к идеальному, чем меньше период модуляции. Однако уменьшение периода модуляции приводит к увеличению частоты срабатывания коммутирующей аппаратуры и включений привода, что нежелательно. Кроме того, исполнительный механизм просто не в состоянии отреагировать на импульсы малой длительности. Поэтому в некоторых случаях качество процессов регулирования на практике заметно ухудшается и даже становится неудовлетворительным [1, 2].

В 2007 г. на кафедре АППиЭ была введена в эксплуатацию лабораторная установка [3], позволяющая проводить эксперименты по исследованию систем автоматического регулирования, в

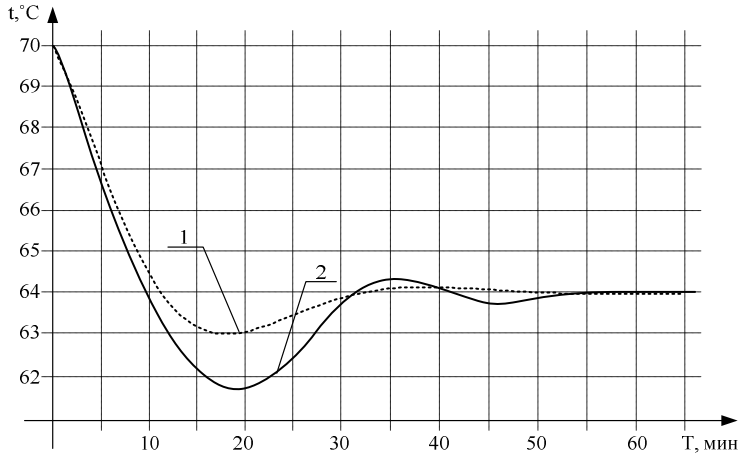


Рис. 1. Переходный процесс системы по заданию:
1 – результаты моделирования; 2 – результаты эксперимента.

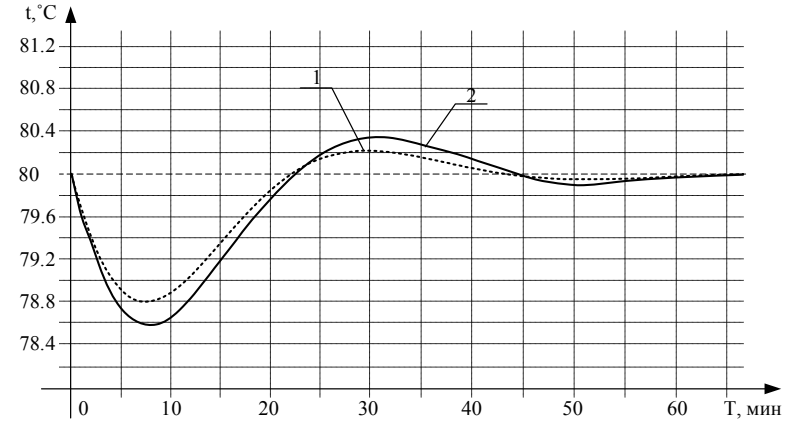


Рис. 2. Переходный процесс системы по возмущению снижением мощности нагревателя:
1 – результаты моделирования; 2 – результаты эксперимента.

том числе описанного выше класса. Выпускником Д.А. Горкун в ходе выполнения дипломного проекта «Исследование характеристик и разработка систем управления лабораторным стендом» среди прочих была реализована система регулирования температуры воды с воздействием на положение воздушной заслонки в контуре охлаждения. Управляющее воздействие на привод заслонки формировал регулирующий микроконтроллер Ремиконт Р130. Программное обеспечение контроллера было построено на основе стандартной конфигурации «Регулирование импульсное» (РЕГИ), включающей алгоритмы импульсного регулирования РИМ и импульсного вывода ИВБ. Результаты экспериментов показали достаточно хорошее качество процесса регулирования. Поведение реальной системы приближается к поведению ее модели (рис.1, 2).

В 2008 г. для того же лабораторного объекта была смонтирована система управления на базе современного программируемого логического контроллера ПЛК 150, модуля ввода аналогового МВА8 и модуля вывода управляющего МВУ8 фирмы «ОВЕН» [4]. Обмен информацией между контроллером и модулями осуществляется по промышленной сети RS485, протокол «ОВЕН».

Первоначально новая реализация системы регулирования температуры с воздействием на положение воздушной заслонки предполагала:

формирование контроллером двух непрерывных управляющих сигналов в диапазоне от нуля до единицы (каналы «больше» и «меньше») по пропорционально-дифференциальному алгоритму из библиотеки среды программирования CoDeSys;

передачу сформированных сигналов модулю МВУ8 по сети;

преобразование модулем МВУ8 полученных сигналов в сигналы управления механизмом «больше» и «меньше» посредством широтно-импульсной модуляции (при этом скважность импульса 0-100% соответствует изменению входного сигнала 0-1, а период модуляции задается при конфигурировании МВУ8).

Однако результаты экспериментов, проведенных при различных настройках регулятора, показали, что система регулирования практически неработоспособна: наблюдались незатухающие колебания температуры и хода исполнительного механизма. На основе анализа трендов экрана визуализации CoDeSys был сделан вывод, что причина заключается в «жестком» алгоритме широтно-импульсной модуляции. При малых сигналах управления длительность импульса настолько мала, что

механизм не успевает его отработать, в результате бóльшая часть импульсов оказывается потерянной. Увеличение периода модуляции не решает проблемы, поскольку свойства системы при этом еще более отличаются от свойств ее непрерывной модели. Попытки перенести процесс формирования импульсов в контроллер не увенчались успехом, так как в библиотеках, предлагаемых системой программирования, найти подходящие алгоритмы не удалось.

Таким образом, актуальной стала задача разработки и программной реализации алгоритмов импульсной модуляции для контролеров, программируемых в среде CoDeSys.

Решение

На первом этапе было проведено исследование алгоритма импульсного вывода, применяемого в контроллере Ремиконт Р130. Программные библиотеки данного контроллера прошиты в ПЗУ и закрыты, а программирование осуществляется с помощью специального пульта. На момент проведения экспериментов отсутствовала возможность подключения Ремиконта Р130 к персональному компьютеру, поэтому выходной сигнал алгоритма импульсного вывода регистрировался непосредственно на дискретном выходе контроллера самописцем РП160. На вход алгоритма ИВБ подавался сигнал, сформированный задатчиком ЗУ05.

В результате экспериментов было установлено, что импульсный вывод осуществляет широтно-импульсную модуляцию с переменным периодом. Импульсы подаются таким образом, чтобы при любом значении входного модулируемого сигнала обеспечить равенство либо времени импульса, либо времени паузы минимальному времени импульса/паузы, которое задается на настроечном входе алгоритма. При этом, если требуется обеспечить скважность менее 50%, минимальным берется время импульса, более 50% − время паузы. Таким образом достигается максимально возможная частота модуляции при ограничении на время срабатывания механизма. Данный подход и был принят за основу в дальнейших разработках.

На втором этапе разработан, промоделирован и реализован на контроллере ПЛК150 алгоритм ШИМ с переменным периодом.

В процессе формирования импульсов алгоритм фиксирует время начала текущего состояния (импульса или паузы). Расчет длительности импульса/паузы производится в зависимости от значения модуля сигнала управления.

Если значение модуля управляющего сигнала менее 0,5 (скважность менее 50%), время импульса берется минимальным, а время паузы определяется по формуле:

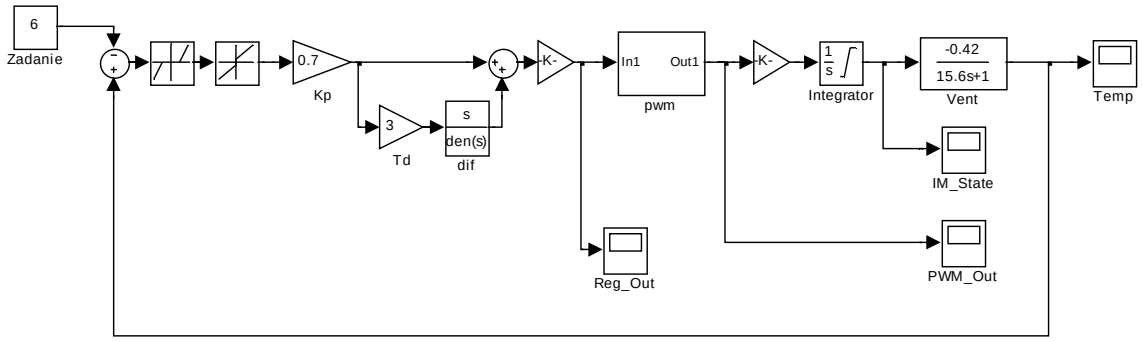
$t_n = \frac{|u|}{1-|u|} \cdot t_{\min}$, где $|u|$ – модуль текущего значения управляющего сигнала; $t_{\min}$ – минимальное время импульса/паузы.

Если значение модуля управляющего сигнала превышает 0,5, время импульса находится по формуле:

$t = \frac{1-|u|}{|u|} \cdot t_{\min}$, а время паузы выбирается минимальным.

В качестве отправной точки для исследований взята упрощенная непрерывная модель системы регулирования температуры с воздействием на изменение положения воздушной заслонки. Передаточная функция объекта (определена путем обработки кривой разгона):

$W_o = \frac{k}{Tp+1}$, где k – коэффициент передачи объекта, равный -0,42°С/% хода механизма; Т – постоянная времени, равная 15,6 мин.



Исходя из требований к быстродействию и с учетом ограничений на колебательность и предельные уровни управляющего сигнала в переходном процессе, выбран ПИ-регулятор с передаточной функцией

$$W_{\text{р}} = k_{\text{р}}\left(1+\frac{1}{T_{из}\,p}\right),$$

где $k_{\text{р}} = 2$ – коэффициент регулятора; $T_{\text{из}} = 1/3$ мин. – постоянная времени изодрома.

Далее для проверки алгоритма ШИМ в системе имитационного моделирования Simulink построена полная модель исследуемой системы (рис. 3)

Рис. 3. Полная модель системы.

Модель содержит ПД-регулятор с зоной нечувствительности и фильтром при дифференциальной составляющей, блок широтно-импульсной модуляции, исполнительный механизм и объект регулирования.

Коэффициенты ПД-регулятора определены по коэффициентам исходного ПИ-регулятора и времени полного хода исполнительного механизма.

Блок широтно-импульсной модуляции (рис. 4) построен на базе функции Matlab, код которой приведен в приложении 1.

Графики переходных процессов при изменении задающего сигнала на 6°, полученные в результате моделирования, представлены на рис. 5, 6. Из рисунков видно, что построенная модель по своим свойствам достаточно близка исходной непрерывной модели и даже обеспечивает несколько лучший переходный процесс.

Для экспериментальной проверки алгоритма программа Matlab была портирована в среду CoDeSys, язык ST (structured text – структурированный текст). Листинг ST-программы приведен в приложении 2.

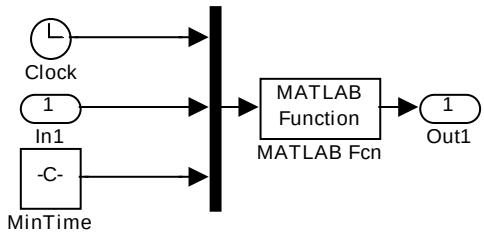


Рис. 4. Блок ШИМ.

Эксперимент проводился при следующих условиях:

ПД-регулятор – типовой из библиотеки CoDeSys, с рассчитанными ранее настройками;

мощность нагревателя – 60 % (1500 Вт);

скорость вращения вентилятора – максимальная;

начальное положение воздушной заслонки – 60%;

начальная температура – 82,5°C;

изменение задания – +6°C;

минимальное время импульса/паузы – 1,4 сек.

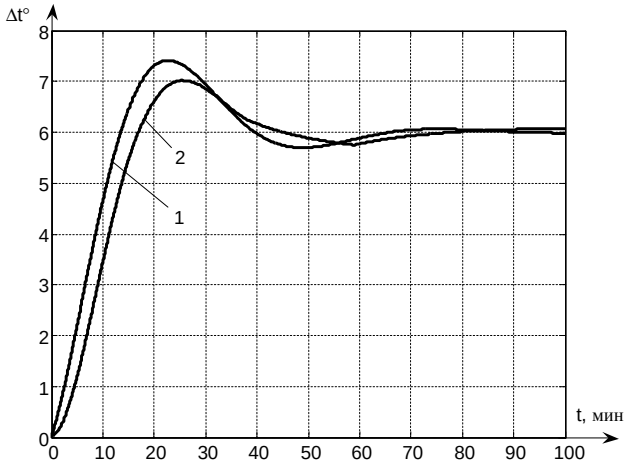


Рис. 5. Результаты моделирования – график изменения температуры:
1 – упрощенная непрерывная модель; 2 – полная модель.

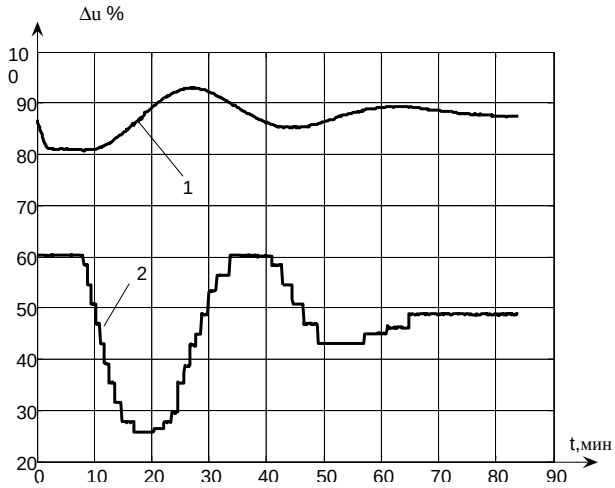


Рис. 6. Результаты моделирования – график изменения управляющего воздействия:
1 – упрощенная непрерывная модель; 2 – полная модель.

Тренды архивировались контроллером ПЛК 150 в файле специального формата, который впоследствии был прочитан и обработан программой на языке Matlab.

Результаты эксперимента представлены на рис. 7.

В целом алгоритм широтно-импульсной модуляции показал свою работоспособность, хотя качество переходного процесса и оставляет желать лучшего. В ходе анализа сделано предположение, что причиной повышенной колебательности является неточная реализация ПД закона регулирования в типовом регуляторе CoDeSys, поэтому в настоящее время ведется работа над собственной программной реализацией регулятора.

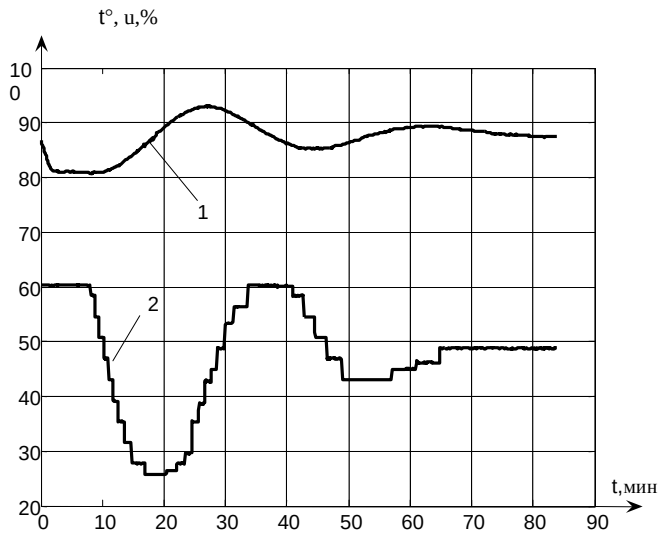


Рис. 7. Результаты эксперимента:
1 – температура; 2 – ход исполнительного механизма.

Заключение

В статье предложены модель и программная реализация алгоритма широтно-импульсной модуляции для управления исполнительными механизмами постоянной скорости, входящими в состав систем автоматического регулирования технологических параметров. Результаты апробированы в системе имитационного моделирования Simulink Matlab и на экспериментальной установке под управлением контроллера ОВЕН ПЛК 150.

1. Куць, И.И., Валько, В.Ф. Изменение типовых схем регулирования (регуляторов питания котла водой и температуры перегретого пара) по опыту эксплуатации // Сб. трудов IV Всероссийской науч.-техн. конф. «Энергетика: качество и эффективность использования энергоресурсов».– Благовещенск: Амурский гос. ун-т, 2005.– С. 496-499.

2. Рыбалев, А.Н. Исследование алгоритмов регулирования температуры перегретого пара, применяемых на Благовещенской ТЭЦ // Сб. трудов IV Всероссийской науч.-техн. конф. «Энергетика: качество и эффективность использования энергоресурсов».– Благовещенск: Амурский гос. ун-т, 2005– С.500-505.

3. Рыбалев, А.Н., Редозубов, Р.Д., Колесников, П.С. Разработка лабораторного стенда по дисциплине «Автоматизация технологических процессов и производств» // Вестник АмГУ. – Благовещенск: Амурский гос. ун-т. – 2007. – Вып. 39. – С. 79-83.

4. Рыбалев, А.Н., Редозубов, Р.Д., Колесников, П.С. Модернизация лабораторного стенда по дисциплине «Автоматизация технологических процессов и производств» // Вестник АмГУ. – Благовещенск: Амурский гос. ун-т. – 2008. – Вып. 43. – С. 50-53.

Приложение 1

Matlab-функция для моделирования ШИМ

```
function output = winfun(input)
persistent state time_of_begin % сохраняемые переменные: состояние и время его начала
time = input(1); % текущее время
u = input(2); % управление -1...1
module_u = abs(u); % модуль управления
minimal_time = input(3); % минимальное время импульса/паузы
% Начальная инициализация: пауза
if (time == 0)
        state = 0;
        time_of_begin = 0;
end
% время текущего состояния
time_of_state = time - time_of_begin;
if (state == 1) % импульс на выходе есть
  if ((module_u < 1)&&(time_of_state > minimal_time))... % импульс может быть снят
  && ((module_u < 0.5)||(time_of_state > module_u/(1 - module_u)*minimal_time)) % и импульс  должен
                                                                                % быть снят
        time_of_begin = time;
        state = 0;
        output = 0;
   end
else % импульса на выходе нет
  if ((module_u > 0) && (time_of_state > minimal_time))... % импульс может быть подан
    && ((module_u > 0.5) || (time_of_state > (1 - module_u)/module_u*minimal_time)) % и импульс
                                                                %должен быть подан
        time_of_begin = time;
        state = 1;
        output = 1;
  end
end
output = state*sign(u);
```

Приложение 2

ST-программа регулирования

```
PROGRAM PLC_PRG
VAR
        INP_BUST: REAL;
        INP_PCH: REAL;
RESET_PD: BOOL;
        RES_PD: BOOL;
        KR:REAL;
        TD:REAL;
        CUR_TEMP: REAL;
TEMP_UST:REAL;
        VAR2: REAL;
REGULATOR_PD: PD;
        UPR_SIGN_PD: REAL;
        SysTime: SysTime64;
        GetTime: CurTime;
        CurrentTime: DWORD;
        TimeOfBegin: DWORD := 0;
        TimeOfState: DWORD := 0;
        MinimalTime: DWORD;
module_u: REAL;
        STATE:BOOL;
        STATER: REAL;
        SIGN_UPR_SIGN: REAL;
        OUT: REAL;
END_VAR
MVU1:=1;
MVU2:=0;
INP_PCH := 100;
```

```
INP_BUST := 60;
MVU7:=INP_PCH/100;
MVU8:=INP_BUST/100;
IF  RESET_PD=TRUE    THEN
          RES_PD:=FALSE;
ELSE
          RES_PD:=TRUE;
END_IF
KR:=0.7;
TD:=180;
CUR_TEMP :=MVA8;
TEMP_UST := 88;
VAR2:=MVA2;
REGULATOR_PD(ACTUAL:=MVA8, SET_POINT:=TEMP_UST, KP:=KR, TV:=TD,
Y_MIN:=-100, Y_MAX:=100);
UPR_SIGN_PD:=REGULATOR_PD.Y/100;
SysTime.ulHigh := 0;
SysTime.ulLow := 0;
GetTime (SystemTime := SysTime);

CurrentTime := SysTime.ulLow;
TimeOfState := CurrentTime - TimeOfBegin;
MinimalTime := 500000;
module_u := ABS(UPR_SIGN_PD);
IF STATE = TRUE AND (module_u < 1 AND TimeOfState > MinimalTime) AND (module_u < 0.5 OR
   TimeOfState > module_u/(1 - module_u)*MinimalTime)  THEN
          TimeOfBegin := CurrentTime;
          STATE := 0;
ELSIF STATE = FALSE AND (module_u > 0 AND TimeOfState >  MinimalTime) AND (module_u > 0.5
         OR TimeOfState > (1 -  module_u)/module_u*MinimalTime) THEN
            TimeOfBegin := CurrentTime;
            STATE := 1;
ELSE
          MVU3 := 0;
          MVU4 := 0;
END_IF
STATER:=BOOL_TO_REAL(STATE);
IF  module_u=0 THEN
          SIGN_UPR_SIGN:=1;
ELSE
          SIGN_UPR_SIGN:=UPR_SIGN_PD/module_u;
END_IF
OUT:=SIGN_UPR_SIGN*STATER;
IF OUT>0 THEN
          MVU3:=0;
          MVU4:=OUT;
ELSIF OUT<0    THEN
          MVU4:=0;
           MVU3:=ABS(OUT);
ELSE
          MVU4:=0;
          MVU3:=0;
END_IF
```