

РОЗРОБКА ТА ВИКОРИСТАННЯ ІМІТАЦІЙНИХ МОДЕЛЕЙ ДЛЯ ВІДЛАГОДЖЕННЯ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ ПРОГРАМОВАНИХ ЛОГІЧНИХ КОНТРОЛЕРІВ

Пропонується методика розробки та використання імітаційних моделей об'єктів управління, розроблених на базі стандартних (МЕК 61131) мов програмування програмованих логічних контролерів (ПЛК), для відлагодження програмного забезпечення ПЛК та SCADA/HMI. Запропоновані розрахункові формули та наведені приклади програм типових імітаційних блоків для ПЛК, які програмуються програмним пакетом UNITY PRO.

***Ключові слова:** об'єкт управління, імітаційна модель, програмований логічний контролер*

При розробці автоматизованих систем управління технологічними процесами виникає необхідність початкового налагодження програмного забезпечення мікропроцесорних контролерів та SCADA-програм до моменту впровадження їх на об'єкті. Практично доведено, що ґрунтовний підхід до попередньої перевірки роботи програмно-технічних засобів в комплексі, значно зменшує часові та фінансові затрати на впровадження розробленої системи. Крім того, для функціонально-небезпечних процесів, до моменту впровадження, система управління повинна пройти обов'язкове тестування. Одним із найбільш економічних та доступних способів перевірки функціонування програмних засобів таких систем є використання програмної імітації технологічних процесів.

У зв'язку з різноманітністю існуючих програмно-технічних засобів для систем АСУТП, напрацювання бази імітаторів для типових об'єктів управління з використанням мов програмування високого рівня стає проблематичним. Альтернативним рішенням цієї задачі є використання для цього ресурсів мікропроцесорних контролерів, які можуть бути як в складі системи управління, яка розробляється так і поза її межами. У якості базової платформи для програмування пропонується використати стандартні мови програмування MEK 61131, зокрема Function Block Diagram (FBD), Ladder Diagram (LD), та Structured Text (ST) які підтримуються більшістю сучасних ПЛК.

Для зменшення витрат часу на побудову імітаційної моделі пропонуються програмні функціональні блоки для імітації типових ланок. Модульність імітаційних моделей дає можливість порівняно легко конструювати і адаптувати моделі під конкретну систему.

Для тестування роботи системи управління, програмний імітатор пропонується включати одним із наступних способів у складі (рис.1):

- а) програмного забезпечення ПЛК управління;
- б) програмного забезпечення окремого ПЛК, що інтегрується з ПЛК управління мережними засобами;
- в) програмного забезпечення окремого ПЛК, що інтегрується з ПЛК управління пристроями зв'язку з об'єктом (ПЗО);
- г) програмного забезпечення імітатора ПЛК на комп'ютері зі встановленою програмою SCADA/HMI.

Досвід роботи з імітаційними моделями показує, що найбільш економічним і простим способом інтеграції є варіант (г), так як він не потребує необхідності використання самого ПЛК, а імітатори для ПЛК є в наявності у всіх потужних платформах ПЛК відомих брендів.

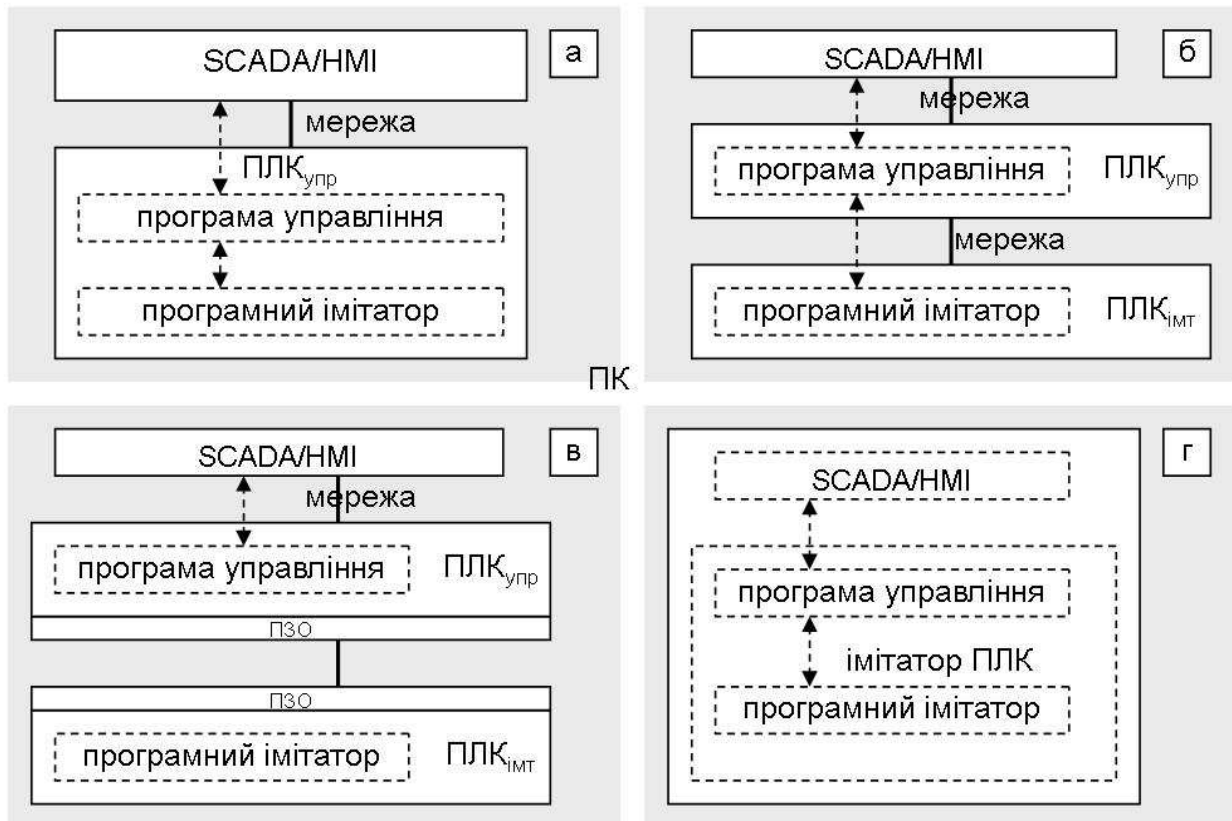


Рис.1. Включення програмних імітаторів в систему, що тестується

Враховуючи, що задачею імітаційного моделювання, в контексті поставленої задачі, є перевірка роботи програми управління, визначимо деякі особливості цих імітаційних моделей:

- імітаційні моделі повинні відображати сутність процесу, однак не обов'язково бути параметрично адекватними реальному об'єкту;
- при використанні ресурсів програмою імітації того ж ПЛК в якому виконується програма управління, модель повинна бути максимально простою і викликатися з періодом достатнім для даного типу об'єкта;
- імітаційні моделі повинні мати можливість виклику в прискореному або сповільненому масштабі часу;

Наведемо деякі приклади окремих імітаційних модулів, які розроблені у середовищі UNITY PRO для програмування контролерів Schneider Electric.

Генератор випадкових чисел. Генератор випадкових чисел може бути використаний при імітації випадкових збурень. Для цього рекомендується

використовувати лінійний конгруентний метод генерації псевдовипадкових чисел, описаний в [1]. Для генерації цілих чисел в діапазоні 0- M , випадкова величина r розраховується за формулою:

$$r_{i+1} = \text{mod}(k \cdot r_i + b, M) \quad (1)$$

де r_{i+1} - значення випадкової величини на новому циклі перерахунку, M , k , b - коефіцієнти, r_0 - початкове значення, MOD – залишок від ділення.

Для генерації випадкової величини можна створити функціональний блок (рис.2). Вхід *init* призначений для ініціалізації блоку. Коефіцієнти M , k , b , та початкове значення r_0 задаються як глобальні змінні блоку. Внутрішня змінна ri призначена для запам'ятовування попереднього значення. При ініціалізації програми користувача, що характерно для початкового кроку – $ri = r_0$, в іншому випадку значення ri береться з попереднього розрахунку. Коефіцієнти вибрані типу UDINT для того, щоб при операції MOD не втрачалась точність, а на вихід вже подається перетворене значення в INT за допомогою універсальної функції UDINT_TO.

```
if init then ri:=r0; end_if;
ri:=MOD (k*ri+b,M);
OUT:=UDINT_TO (ri);
```

Рис.2. Реалізація функціонального блоку для генерації випадкових величин

Генератор шумів. Генератор шумів – це функціональний блок, який вносить в корисний сигнал типу INT шуми відповідної амплітуди від minNoise до maxNoise, тобто формує вихідний сигнал за формулою:

$$OUT = IN + \text{шум} \quad (2)$$

Програма для цього блоку може мати вигляд наведений на рис.3. За основу взятий математичний апарат розрахунку випадкових чисел в діапазоні 0-32767,

$$\text{де } M = 2^{31} - 1 = 16\#7FFF, k = 69069, b = 7, r^0 = 7.$$

```

if %s0 then ri:=7; end_if;
ri:=MOD (69069*ri+7,16#7FFF); (*Randomize 0-32767*)
OUT:=DINT_TO_INT(INT_TO_DINT(maxNoise-minNoise)
                *UDINT_TO_DINT(ri)
                /32767)
+ minNoise
+IN;

```

Рис.3. Реалізація функціонального блоку для генерації шумів

При холодному старті ПЛК ($\%S0=TRUE$ – перший цикл після холодного старту), проводиться ініціалізація випадкової величини $r_i=7$. Далі проводиться масштабування сгенерованого числа за лінійним законом, згідно вказаних меж для шуму (min_Noise , $maxNoise$). Для коректності операцій множення та ділення всі операції проводяться з типом DINT, а потім перетворюються в INT. Шум додається до вихідного сигналу IN , а результат записується в OUT .

Імітаційне моделювання на основі рішення диференціальних рівнянь явним методом Ейлера. Моделювання режиму динаміки так чи інакше пов'язане з диференціальними рівняннями за часом. Числові методи вирішення таких рівнянь добре описані в літературі, зокрема в [2], [3], [4]. Серед них найпростішим, хоч і найменш точним, є явний метод Ейлера.

Диференціальне рівняння 1-го порядку описується формулою:

$$\frac{dy}{dt} = f(t, y(t)) \quad (3)$$

при початкових умовах $y(t_0)=y_0$, де y – вихідна величина, а t – час.

Диференціали в рівнянні (3) замінюються на кінцеві різницеві вирази:

$$\frac{dy}{dt} = \frac{\Delta y}{\Delta t} = \frac{(y_{n+1} - y_n)}{\Delta t} \quad (4)$$

де $n+1$ – крок, на якому відбувається розрахунок, n -попередній крок, $\Delta t=t_{n+1}-t_n$ - величина кроку що дорівнює проміжку часу між розрахунками.

Проблема розрахунку (3) заключається у тому, що значення $y(t)$ на кроці $n+1$ використовується у розрахунку того самого $y(t)$, тобто для вирішення такого рівняння треба користуватися пошуковими методами, що потребує

затрати часу. У явному методі Ейлера в праву частину рівняння (3) замість $y(t)$ підставляють значення на попередньому кроці:

$$y_{n+1} = y_n + \Delta t \cdot f(t_n, y_n) \quad (5)$$

Хоч рішення методом Ейлера може давати великі розбіжності, він годиться для імітаційного моделювання з метою перевірки працездатності програм. Якщо імітаційні моделі необхідно використати в алгоритмах управління, можна скористуватися більш точними методами.

Імітаційна модель рівня. Одним з найбільш популярних об'єктів в харчовій та хімічній промисловості є ємність з рідиною, в якій необхідно підтримувати заданий рівень. Припустимо ємність має 3 підводи рідини з витратами F_{in1} , F_{in2} , F_{in3} і 3 відводи рідини з витратами F_{out1} , F_{out2} , F_{out3} (рис.4). При однаковій густині ρ всіх вхідних рідин, матеріальний баланс для даного об'єкта буде мати вигляд:

$$\frac{dV \cdot \rho}{dt} = F_{in1} \cdot \rho + F_{in2} \cdot \rho + F_{in3} \cdot \rho - F_{out1} \cdot \rho - F_{out2} \cdot \rho - F_{out3} \cdot \rho \quad (6)$$

Скоротивши вираз (6), та застосувавши метод Ейлера отримаємо:

$$V_{n+1} = V_n + \Delta t \cdot (F_{in1} + F_{in2} + F_{in3} - F_{out1} - F_{out2} - F_{out3}) \quad (7)$$

Якщо ємність циліндрична або паралелепіпед, тобто по висоті не змінюється площа перерізу, то:

$$L = \frac{V}{S} \quad (8)$$

Якщо ємність неправильної форми, можна застосувати алгоритм перетворення об'єму в рівень з використанням кусочно-лінійної функції. У будь якому випадку, об'єм рідини в ємності не може бути більше об'єму ємності і менше нуля. Тому в програмі імітації треба передбачити вихід за ці значення.

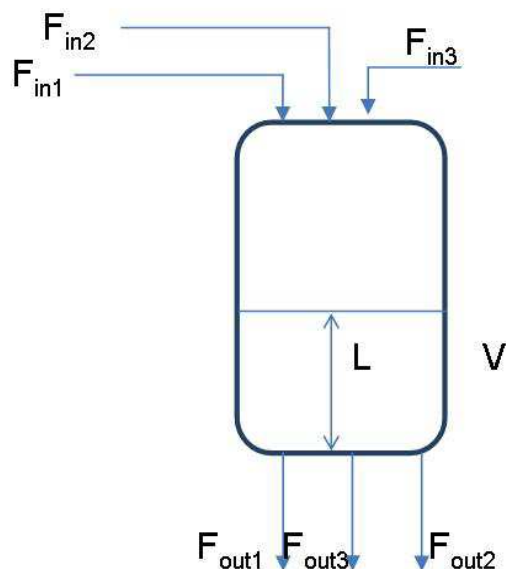


Рис.4. Ємність з декількома підводами і відводами

Для імітації рівня з 3-ма підводами і 3-ма відводами можна створити блок, який показаний на рис.5. Екземпляри блоку smLevelCyl повинні викликатися періодично, наприклад шляхом застосування параметру EN, який буде прив'язаний до імпульсу заданої періодичності на 1 цикл. Для роботи в реальному часі, періодичність виклику повинна співпадати з заданою d_t . Змінна d_t має тип REAL і задається в секундах. Так, якщо необхідно задати періодичність виклику 500 мс, треба вказати $d_t=0.5$. Тип REAL був вибраний для зменшення кількості перетворень.

Якщо процес імітації треба провести в прискореному режимі, параметри d_t можна збільшувати, однак це може привести до погіршення точності в розрахунках.

```

if init then V:=V0;
else
  V:=V+d_t*(F_in1+F_in2+F_in3-F_out1-F_out2-F_out3);
  if V<0.0 then V:=0.0; end_if;
  if V>Vmax then V:=Vmax; end_if;
  L:=V/S;
end_if;

```

Рис.5. Програма для імітаційної моделі рівня в ємності циліндричної форми

Для ємностей, що мають неправильну форму функціональний блок буде мати схожий вигляд, за винятком розрахунку рівня (рис.6). Для розрахунку

рівня можна використати функціональний блок кусково-лінійного перетворення. У даному випадку він називається PWL. У якості входних параметрів блока PWL задаються два масиви – координат точок об’єму та рівня рідини в ємності. Для того щоб задати ці координати, вони винесені в інтерфейс блоку smLevelFree.

```

if init then V:=V0;
else
  V:=V+d_t*(Fin1+Fin2+Fin3-Fout1-Fout2-Fout3);
  if V<0.0 then V:=0.0; end_if;
  if V>Vmax then V:=Vmax; end_if;
PWL (X_IN := V,
      X := Vpoints , Y := Lpoints,
      Y_OUT => L);
end_if;

```

Рис.6. Програма для імітаційної моделі рівня ємності неправильної форми

Імітаційна модель аперіодичної ланки (першого порядку). Якщо система управління розробляється для об’єкта, який має розроблені динамічні математичні моделі в операторному вигляді, її доволі просто можна перевести у програму ПЛК. Рівняння, яке описує об’єкт першого порядку має наступний вигляд:

$$\tau \cdot \frac{dy(t)}{dt} + y(t) = K \cdot x(t) \quad (8)$$

або в операторному вигляді:

$$Y(p) = \frac{K}{T \cdot p + 1} \cdot X(p) \quad (9)$$

де X -вхід об’єкту, Y -вихід об’єкту, K – коефіцієнт пропорційності об’єкту по каналу X , τ – стала часу.

Для реалізації аперіодичної ланки необхідно створити функціональний блок, який проводить розрахунки за формулою:

$$y_{n+1} = y_n + \frac{\Delta t}{\tau} \cdot (K \cdot x_{n+1} - y_n) \quad (10)$$

Перепишемо формулу (10) у вигляді:

$$OUT = OUT_OLD + d_t / LAG * (GAIN * IN - OUT_OLD) \quad (11)$$

де OUT – вихід об'єкта на кроці $n+1$, OUT_OLD – вихід об'єкта на кроці n , d_t – періодичність перерахунку Δt , LAG – стала часу τ , $GAIN$ – коефіцієнт об'єкта K .

Позначення які використовуються в (11) вживаються часто в описі функціональних блоків різних ПЛК, тому надалі будемо користуватися ними.

На рис.7 показана реалізація функціонального блоку *smLAG*. Слід звернути увагу, що замість OUT_OLD використовується OUT . Враховуючи, що значення виходів функціонального блока зберігаються між викликами, немає сенсу використовувати додаткову змінну.

```
if init then OUT:=0.0; end_if;
OUT:=OUT+d_t/LAG*(GAIN*IN-OUT);
```

Рис.7. Програма для імітаційної моделі аперіодичної ланки 1-го порядку

Імітаційна модель для транспортного запізнення. Для об'єкта з чистим (транспортним) запізненням, формула для виходу записується:

$$y(t) = x(t - \tau) \quad (12)$$

Функціональний блок транспортного запізнення з назвою *smDELAY* показаний на рис.8. Алгоритм роботи блока базується на буфері типу FIFO, який реалізовується через масив *BUF*. Кожне нове значення записується на вершину стеку, тобто в 0-й елемент масиву ($BUF[0]:=IN$). Для того, щоб всі інші елементи зсунулися вниз, використовується спеціальна процедура *ROL_ARREAL*, яка останній елемент масиву (в нашому випадку 99-й) записує в перший (в нашому випадку 0-й). У інших платформах зсув можна виконувати через внутрішній цикл.

Враховуючи, що буфер потрібен через кожні d_t секунд тільки на часовому діапазоні *DELAY*, елемент з номером $last:=DELAY/d_t$ буде тим значенням, яке повинно подаватися на вихід. При ініціалізації блоку, у всі елементи масиву записується 0 (*MOVE_REAL_ARREAL*).

При використанні моделі треба слідкувати за співвідношенням $DELAY/d_t$, яке не повинно перевищувати значення 99, яким обмежується розмір масиву BUF. Якщо запізнення дуже велике необхідно або збільшити кількість елементів в масиві BUF, що приведе до збільшення ресурсоемності, або збільшити інтервал d_t , що приведе до зменшення точності.

```
if init then
  MOVE_REAL_ARREAL(IN :=0.0, OUT =>BUF);
end_if;
last:=real_to_int(DELAY/d_t);
ROL_ARREAL (N := 1, INOUT:=BUF);
BUF[0]:=IN;
if last<100 then OUT:=BUF[last];end_if;
```

Рис.8. Програма для імітаційної моделі транспортного запізнення

Висновки. Запропонований підхід до відлагодження програмного забезпечення на базі імітаційного моделювання легко інтегрується в різні системи завдяки використанню стандартної мови програмування МЕК і надає можливість: перевірити роботу системи в режимі реального часу; будувати моделі об'єктів різної складності, які описані стандартною системою диференціальних рівнянь; перевірити роботу різних за архітектурою систем. Розроблені функціональні блоки можуть бути використані в процесі імітаційного моделювання для відлагодження систем автоматизованого управління харчовими та хіміко-технологічними процесами.

ЛІТЕРАТУРА

1. О.И.Мухин. Моделирование систем. [Електронний ресурс, режим доступу <http://stratum.ac.ru/textbooks/modelir/contents.html>]
2. Гартман Т.Н. Основы компьютерного моделирования химико-технологических процессов. Учеб. пособие для вузов. / Гартман Т.Н., Клушин Д.В. — М. : ИКЦ «Академкнига», 2006. — 416 с.
3. Carlos A. Principles and Practice of Automatic Process Control, 2nd Edition, Wiley, / Carlos A. Smith, Armando B. Corripio. 1997, - 784 pp

4. Wayne B. Process Dynamics: Modeling, Analysis and Simulation / B. Wayne Bequette, Rensselaer Polytechnic Institute, NY: Prentice Hall, 1998, - 640 pp

Пупена О.М., Эльперин И.В., Кушков В.М.

РАЗРАБОТКА И ИСПОЛЬЗОВАНИЕ ИМИТАЦИОННЫХ МОДЕЛЕЙ ДЛЯ ОТЛАДКИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ ПРОГРАММИРУЕМЫХ ЛОГИЧЕСКИХ КОНТРОЛЛЕРОВ

Предлагается методика разработки и использования имитационных моделей объектов управления, разработанных на базе стандартных (МЭК 61131) языков программирования программируемых логических контроллеров (ПЛК), для отладки программного обеспечения ПЛК и SCADA/HMI. Предложены: варианты включения программы с имитационными моделями в структуру управления, расчетные формулы и примеры программ типовых имитационных блоков для ПЛК (генератор случайных чисел, генератор шума, моделирование уровня, апериодическое звено, звено чистого запаздывания), программируемых с использованием программного пакета UNITY PRO.

***Ключевые слова:** имитационная модель, программируемый логический контроллер.*

A. Pupena, I. Elperin, V. Kushkov

DESIGN AND USE SIMULATION MODELS FOR SOFTWARE DEBUGGING OF PROGRAMMABLE LOGIC CONTROLLERS

The technique of design and use of simulation models of control objects, developed on the basis of standard (IEC 61131) languages programmable logic controllers (PLCs), for debugging software PLC and SCADA / HMI. Offered: the options to include the program with simulation models in the control structure, formulas and examples of typical simulation program blocks for the PLC (random number generator, the generator noise, level modeling, first-order lag, dead time), programmable using a software package UNITY PRO.

***Keywords:** simulation model, a programmable logic controller.*

Надійшла до редколегії вересень 2012 р.