Poprawa bezpieczeństwa funkcjonalnego oprogramowania PLC za pomocą analizy sygnatur

Marcin Szuster
Politechnika Rzeszowska im. Ignacego Łukasiewicza, Wydział Budowy Maszyn i Lotnictwa, Katedra Mechaniki Technicznej, al. Powstańców Warszawy 12, 35-959 Rzeszów

Bartłomiej Kozioł
Szkoła Doktorska Nauk Inżynieryjno-Technicznych na Politechnice Rzeszowskiej, al. Powstańców Warszawy 12, 35-959 Rzeszów

Streszczenie: W artykule zaprezentowano ideę techniki kompresji odpowiedzi zwaną analizą sygnatur zastosowaną do walidacji oprogramowania PLC pod kątem bezpieczeństwa funkcjonalnego sterowania. Na przykładach symulacyjnych przedstawiono sposób implementacji opracowanej idei walidacji, mającej praktyczne zastosowanie. Przedstawiono wyniki badań na wybranym przykładzie.

Słowa kluczowe: analiza sygnatur, ukryte naruszenia bezpieczeństwa, walidacja programu PLC, bezpieczeństwo funkcjonalne sterowania maszyn

1. Wprowadzenie

Bezpieczeństwo jest właściwością systemu, która odzwierciedla zdolność systemu do standardowego lub niestandardowego działania, bez niebezpieczeństwa spowodowania obrażeń lub śmierci człowieka, czy zagrożenia środowiska. Ważne jest, aby wziąć pod uwagę bezpieczeństwo oprogramowania. W większości przemysłowe systemy sterowania oparte są na oprogramowaniu tworzonym w jednym z języków normy IEC 61131-3 [3], opisującej graficzne i tekstowe języki programowania sterowników PLC. Sposób walidacji oprogramowania jest bezpośrednio związany z ogólnym bezpieczeństwem systemu. Jednym ze sposobów diagnozowania układów cyfrowych jest porównywanie oczekiwanej odpowiedzi układu z wzorcem. W artykule przedstawiono technikę, historycznie opracowaną do sprzętowego diagnozowania układów cyfrowych, zwaną analizą sygnatur. Na przykładach symulujących pracę sterownika PLC, opracowanych w oprogramowaniu Delta ISPSoft (DVP Simulator), dla sterownika DVP-SS2 przedstawiono ideę zastosowania analizy sygnatur do walidacji oprogramowania sterowania maszyny.

Metody analizy sygnatur są stosowane w działach utrzymania ruchu do analizy sygnatur uszkodzeń maszyn – konserwacja predykcyjna maszyn (ang. *predictive maintenance*) [1, 4]. Przykładami usterek, które można zidentyfikować na podstawie analizy sygnatur np. poprzez analizę drgań są: usterka łożyska tocznego, usterka łożyska poprzecznego, usterka sprzęgła elastycznego itp. Stosując analizy sygnaturowe prądu silnika można przewidzieć jego stopień zużycia [7].

Autor korespondujący:
Marcin Szuster, mszuster@prz.edu.pl

Artykuł recenzowany
nadesłany 20.06.2022 r., przyjęty do druku 26.09.2022 r.

Układ sterowania musi być tak zbudowany, żeby sterowany proces był zawsze bezpieczny. Jest to możliwe tylko wówczas, gdy proces ten będzie po uruchomieniu nieustannie kontrolowany przez sterownik PLC wykonujący odpowiedni program użytkownika [2]. Kod programu PLC sterowany zdarzeniami jest wrażliwy na czas i kolejność ich wystąpienia. Istnieją rozwiązania, w których walidacja oprogramowania przeprowadzana jest w sposób automatyczny przez zewnętrzny program testujący [9], który automatycznie generuje czasowe sekwencje zdarzeń. Zawarty w zewnętrznym urządzeniu testującym analizator zdarzeń wskazuje obszary, w których mogą znajdować się ukryte naruszenia bezpieczeństwa związane z błędami programistycznymi. Takie podejście wymaga dużych nakładów czasu na napisanie i rozwój programu wykrywającego potencjalne naruszenia bezpieczeństwa. Metoda poprawy bezpieczeństwa funkcjonalnego oprogramowania PLC, przedstawiona w niniejszym artykule, zastosowana w procesie walidacji umożliwia wykrycie naruszeń bezpieczeństwa spowodowanych niewłaściwie napisanym programem sterującym. Oznacza to najczęściej źle zaprojektowany algorytm sterowania, w którym np. nie uwzględniono błędnych sposobów postępowania obsługi [2] – struktura programu z tzw. kontrolowanymi działaniami obsługi, które nie były wystarczająco przetestowane lub programista przeoczył je pisząc program sterujący. W dostępnej literaturze autorzy nie spotkali się z zastosowaniem metody analizy sygnatur do walidacji systemów bezpieczeństwa w danych chwilach czasowych lub przy zmianach stanu systemu. Zaproponowana metoda jest elastyczna i prosta w realizacji, dzięki czemu nie wymaga znacznych nakładów czasowych na jej implementację.

Oprogramowanie napisane w sposób przedstawiony w niniejszym artykule można stosować do wspomagania procesu walidacji systemu sterowania przed oddaniem maszyny do użytku, czy okresowego sprawdzania poprawnej pracy programu sterującego. W pracy na wybranym przykładzie określono sygnaturę prawidłowo działającego programu, a następnie wprowadzono losowe uszkodzenia i porównano otrzymane sygnatury z wzorcowymi. Podano podstawowe właściwości układu diagnostycznego tzw. analizatora sygnatur [6]. Stosowanie analizy sygnatur wymaga uwzględnienia specyficznych wymagań w procesie projektowania układu sterowania.

2. Analiza sygnatur

Sygnatura to znak lub napis rozpoznawczy mający znaczenie podpisu [5]. W niniejszym artykule sygnaturą nazwany jest cyfrowy obraz stanu wejść i wyjść PLC w określonym punkcie testowym (Rys. 1), przekształcony do krótszej, bardziej czytelnej postaci za pomocą obliczenia zastępczej sumy kontrolnej o stałej długości, niezależnej od ilości wejść, dla której jest obliczana. Podstawową zasadą analizy sygnatur, zdefiniowanej jako metoda określania błędów w systemie poprzez pewnego rodzaju dopasowywanie wzorców, jest to, że efekt jest wynikiem przyczyny. Wynika z tego, że każda aplikacja do analizy sygnatur powinna opierać się na procesie mapowania, lepiej znanym jako rozpoznawanie wzorców. Sygnatury mogą być obrazami, przebiegami sygnału, lub dowolnym rodzajem pomiaru, który należy sklasyfikować.

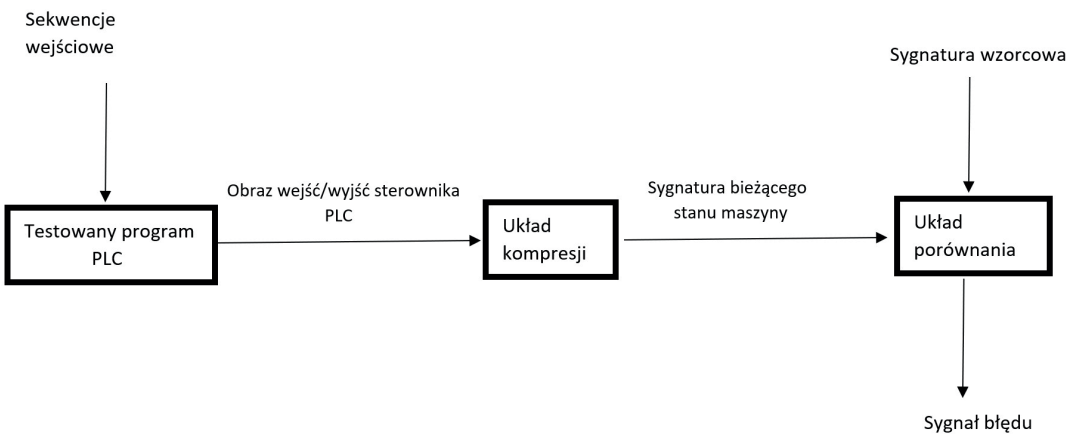
Technika zwana analizą sygnatur polega na kompresji odpowiedzi układu do krótszej postaci i wywodzi się z techniki cyklicznych kodów nadmiarowych CRC (ang. *Cyclic Redundancy Checks*), gdzie służyła pierwotnie do diagnostyki systemów cyfrowych. Metoda ta była od 1977 r. propagowana przez firmę Hawlett Packard, która wprowadziła pierwsze urządzenie analizujące: Signature analyzer 5004A [6]. Przy jej pomocy można testować zarówno warstwę sprzętową jak i programową [8], co pokazano w niniejszym artykule. Znając odpowiedź układu sterowania pracującego poprawnie można diagnozować program układu sterowania poprzez porównanie poprawnej odpowiedzi z odpowiedzią testowanego programu układu sterowania. W przypadku długich danych binarnych ich analiza i porównywanie z wzorcem jest uciążliwe i czasochłonne, ponieważ obraz każdego wejścia i wyjścia cyfrowego sterownika PLC musiałby zostać porównany bezpośrednio z wzorcem. Opracowano więc metodę umożliwiającą walidację programu PLC pod kątem bezpieczeństwa funkcjonalnego, w której ciąg bitów ulega kompresji do krótkiej sekwencji bitowej (technika kompresji odpowiedzi) – sygnatury zawierającej opis wejść i wyjść cyfrowych testowanego programu PLC. Sygnatury wzorcowe są otrzymywane empirycznie w wyniku badania spodziewanego ciągu wejść i wyjść sterownika PLC. Zmiana jednego bitu powoduje dużą odległość sygnatury bieżącego stanu maszyny od sygnatury wzorcowej. Przez porównanie sygnatury bieżącego stanu maszyny otrzymanej w trakcie pracy lub symulacji PLC z sygnaturą wzorcową, układ porównania może stwierdzić prawidłową pracę oprogramowania bądź potencjalne defekty programowalnego układu sterowania.

3. Przykład implementacji analizy sygnatur w języku LAD

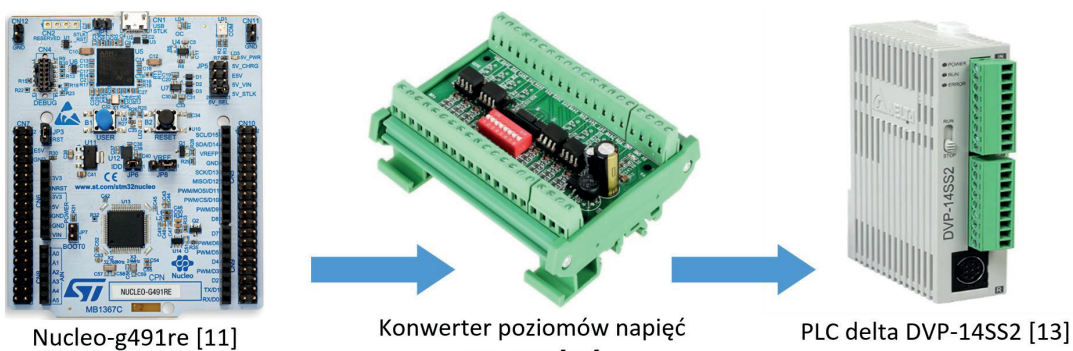
Poniżej przedstawiono fragmenty programu analizatora sygnatur, opracowanego w języku LAD i oprogramowaniu ISP Soft. Użyty sterownik DVP-SS2 umożliwia tworzenie 16-bitowych sygnatur wejść/wyjść cyfrowych [10]. Podczas walidacji układ testowy musi być pobudzany do pracy, aby wystąpiły niezbędne zmiany stanów wejść cyfrowych. Konieczny jest do tego zespół cyfrowych ciągów pobudzających, które można wygenerować w następujący sposób:

- stosując zewnętrzne urządzenie do zadawania stanów wejść w odpowiednich chwilach czasowych (Rys. 2),
- dostarczając sygnały z sensorów maszyny,
- stosując programowe źródło ciągów testowych – wymuszanie stanów.

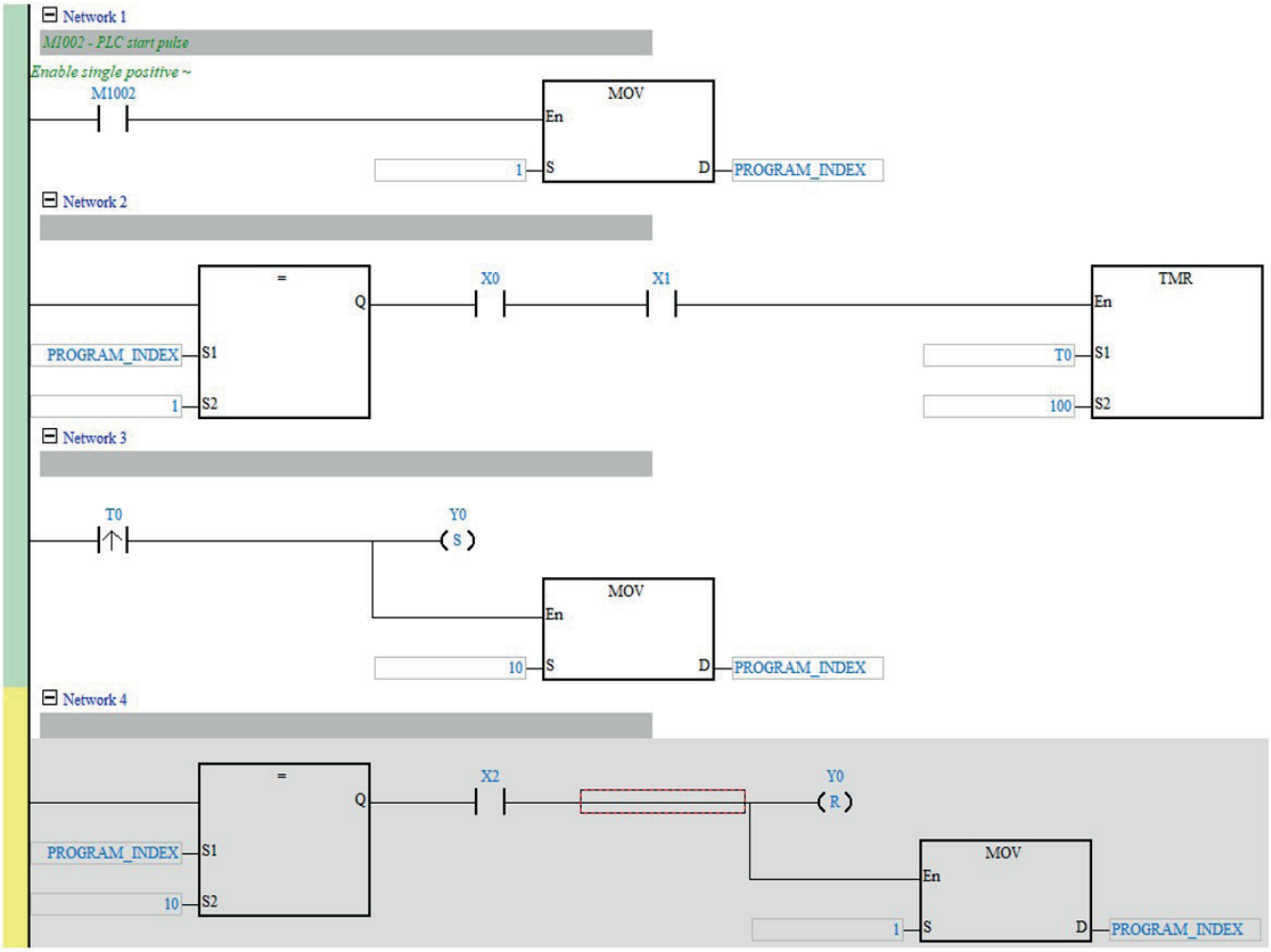
Zewnętrzne urządzenie do zadawania sygnałów można zbudować w oparciu o zestaw startowy z mikrokontrolerem np. Nucleo-g491re [11], którego konfigurację i programowanie można zrealizować w darmowym oprogramowaniu STM32 CubeMX, STM32 CubeProgrammer, lub STM32 CubeIDE 1.4.0. Dołącze-



Rys. 1. Idea walidacji oprogramowania przy pomocy metody analiz sygnatur
Fig. 1. The idea of software validation using method of signature analysis



Rys. 2. Idea zewnętrznego układu testującego do zmiany stanów wejść PLC w zadanych chwilach czasowych
Fig. 2. The idea of an external test system to change the state inputs PLC in the specified time



Rys. 3. Program sterujący – program sekwencyjny napisany przy użyciu PROGRAM_INDEX
Fig. 3. Control program – sequence program written using PROGRAM_INDEX

nie konwertera sygnałów 5–24 V umożliwia sterowanie wejściami PLC z poziomu mikrokontrolera. Duża liczba dostępnych timerów (11) i ich precyzja umożliwiają zadawanie sygnałów sterujących w dowolnych chwilach czasowych. Jednak ze względu na potrzeby niniejszego artykułu i prostotę rozwiązania – cyfrowe ciągi pobudzające są dostarczane poprzez programowe źródło ciągów testowych. Oprogramowanie ISPSoft sterownika PLC umożliwia zarówno generowanie stanów logicznych, jak i analizę wyjść badanego układu. Do testów analizatora sygnatur zastosowano programy napisane w sposób sekwencyjny implementując identyfikator tzw. „PROGRAM_INDEX", informujący o aktualnym stanie realizacji algorytmu sterującego (Rys. 3).

Poniżej omówiono przykład implementacji identyfikatora PROGRAM_INDEX do sekwencyjnej kontroli pracy układu sterowania. Ponieważ rzeczywiste programy sterujące pracą maszyn są skomplikowane i ich opis zająłby wiele miejsca, więc na potrzeby artykułu uproszczono do minimum program sterujący, który następnie rozbudowano o program monitorujący.

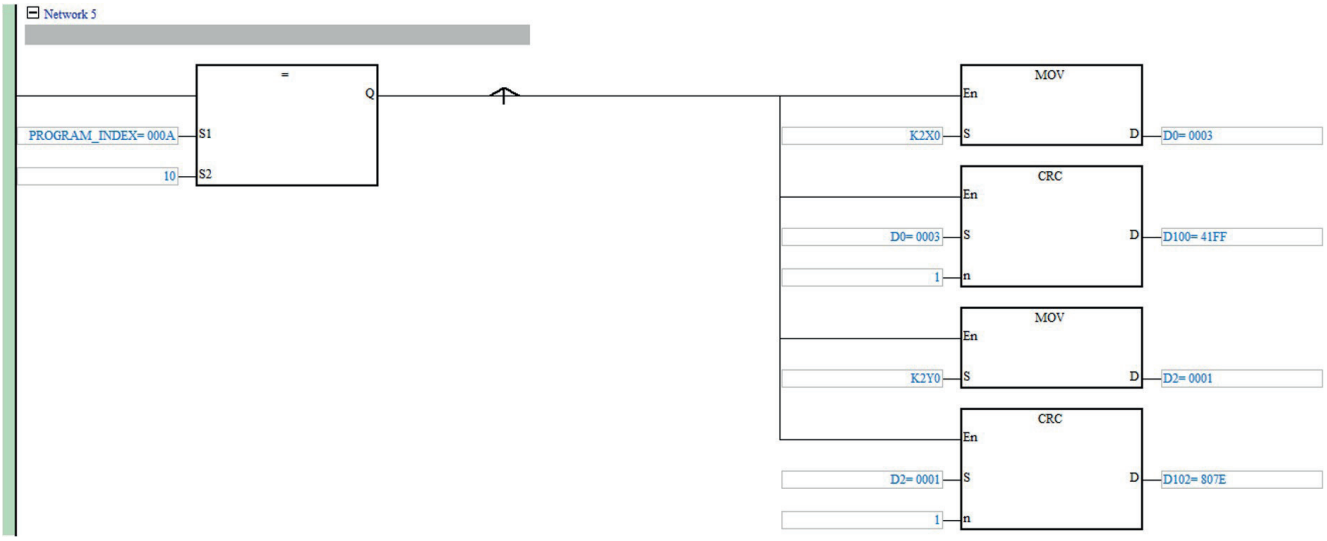
Program sterujący za pomocą markera M1002 (Rys. 3) reaguje na rozpoczęcie pracy sterownika PLC. Do PROGRAM_INDEX wpisywana jest wartość 1. Jeśli w tym stanie zostaną załączone czujniki X0 i X1 na co najmniej 10 sekund, wówczas zostanie załączony zegar T0 ustawiając wyjście Y0 w stan wysoki i nastąpi wpisanie do PROGRAM_INDEX wartość 10. Jeśli PROGRAM_INDEX = 10 i sygnał z czujnika X2 = 1 wyjście Y0 zostanie ustawione w stan Y0 = 0, a realizacja programu rozpocznie się od początku.

Zmiana stanu PLC jest równoznaczna zmianie identyfikatora PROGRAM_INDEX programu sterującego, co zastosowano do obliczenia sygnatury sterownika w zadanym stanie.

Sygnatura jest w procesie testowania przyporządkowywana do danej wartości identyfikatora PROGRAM_INDEX, przy każdej zmianie PROGRAM_INDEX tworzony jest obraz wejść i wyjść cyfrowych PLC. Program monitorujący można „wmontować" w program sterujący lub dopisać na końcu, czyli po programie sterującym. Przykład programu monitorującego dopisanego do programu sterującego znajduje się na Rys. 4 i 5. W programie monitorującym przejście wartości z PROGRAM_INDEX = 1 na PROGRAM_INDEX = 10 (zbocze narastające) powoduje zapis obraz stanu wejść cyfrowych X0–X7 oraz wyjść cyfrowych Y0–Y7. Obraz wejść zapisywany jest do rejestru D100, obraz wyjść do rejestru D102. Do zapisu stanu cyfrowych wejść/wyjść zastosowano instrukcję MOV, która przenosi stan wejść/wyjść PLC – wejścia od X0–X7 do pamięci D0, wyjścia Y0–Y7 do pamięci D2. Zapis K2X0 oznacza pobranie ośmiu kolejnych bitów stanu obrazu wejść PLC, a więc X0–X7.

W przykładzie z Rys. 4 ustawione są bity X0 i X1, co heksadecymalnie równe jest wartości 3. Na wyjściu Y0 pojawia się stan wysoki Y0 = 1, co heksadecymalnie równe jest 1. Następnie obraz stanu wejść cyfrowych X0–X7 oraz wyjść Y0–Y7 jest poddawany kompresji odpowiedzi (Rys. 4) za pomocą algorytmu CRC przedstawionego w Tabeli 1. Sposób generowania sygnatury wywodzi się z techniki stosowanej w telekomunikacji. Za pomocą generacji CRC dokonuje się kompresji do sygnatury o wielkości dwóch bajtów.

Wynik kompresji odpowiedzi, czyli sygnatura bieżącego stanu programu sterującego, może być przesyłany za pomocą jednego z przemysłowych protokołów komunikacji (np. Modbus) do zewnętrznego urządzenia analizującego wyznaczone sygnatury, bądź poddany analizie przy pomocy wewnętrznego, programo-



Rys. 4. Przykład obliczania CRC dla 8 wejść oraz 8 wyjść cyfrowych PLC (symulacja dla X0 = 1, X1 = 1)
Fig. 4. CRC calculation example for 8 digital inputs and 8 digital outputs PLC (simulation for X0 = 1, X1 = 1)

Tabela 1 [10]

Algorytm sprawdzenia sumy kontrolnej (CRC)

Krok 1: Ustawienie 16 bitowego rejestru (rejestr CRC) = 0xFFFF.

Krok 2: Wykonanie operacji XOR na pierwszej 8-bitowej danej wskazywanej przez rejestr CRC i niższych 8 bitach rejestru CRC. Wynik przechowany w rejestrze CRC.

Krok 3: Przesuń rejestr CRC w prawo o bit i wypełnij zerem najwyższy bit.

Krok 4: Sprawdź najniższy bit (bit 0) przesuniętej wartości. Jeśli bit 0 ma wartość 0, wpisz nową wartość uzyskaną w kroku 3 do rejestru CRC. Jeśli bit 0 jest niezerowy, wykonaj operację XOR na A001H i przesuniętej wartości i zapisz wynik w rejestrze CRC.

Krok 5: Powtórz kroki 3–4, aby zakończyć całą operację na wszystkich 8 bitach.

Krok 6: Powtarzaj kroki 2–5, aż wszystkie operacje zostaną zakończone.

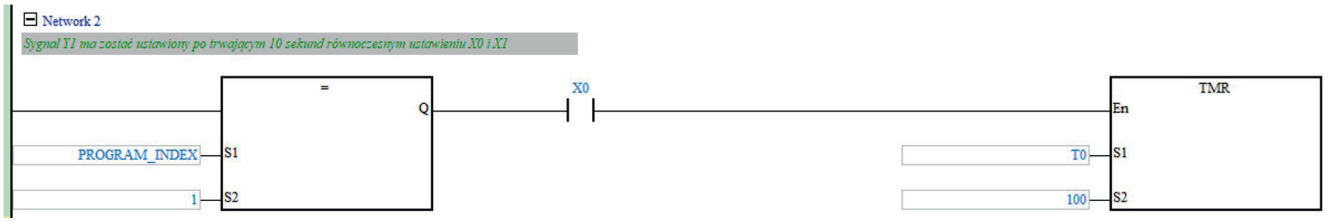
Przykład obliczeń CRC w języku C:

```
unsigned char* data // wskaźnik do rejestru zawierającego dane do obliczenia CRC
unsigned char length // rozmiar danych
unsigned int crc_chk( unsigned char* data, unsigned char length )
  {
        int j;
        unsigned int reg_crc = 0Xffff; // inicjalizacja rejestru reg_crc wartościami „1”
        while(length--)
              {
                    reg_crc ^= *data++;
                    for (  j=0; j<8; j++ )
                    {
                   if (reg_crc & 0x01)
                {
                   reg_crc = ( reg_crc>>1 ) ^ 0xA001;  /* LSB(b0)=1 */
                }
                   else
                {
                   reg_crc = reg_crc >>1;
                }
              }
    return reg_crc;
}
```

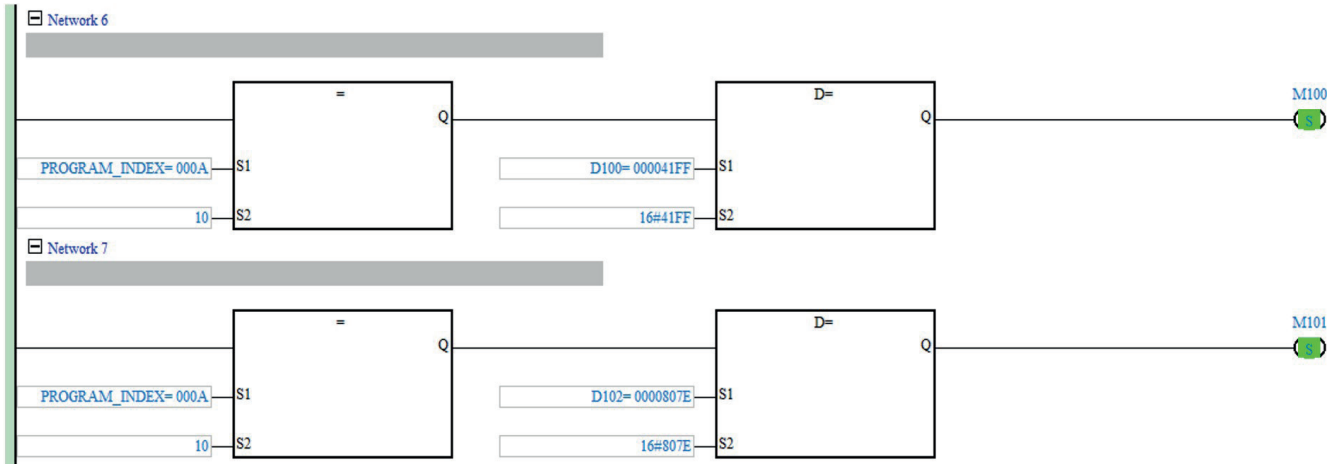
wego analizatora sygnatur. Sygnaturę wejść X0-X7 PLC dla programu sterującego z Rys. 3 (X0 = 1 i X1 = 1) stanowi wartość rejestru D100 = 41FF natomiast dla wyjść Y0–Y7 (Y0 = 1) D102 = 807E. Jeśli przyjmiemy, że dla prawidłowo działającego układu sterowania dla PROGRAM_INDEX = 10 powinny być załączone wejścia X0 = 1, X1 = 1 oraz wyjście Y0 = 1, wówczas każda modyfikacja wejść/wyjść PLC (np. przez dopisanie kolejnych programów cyklicznych modyfikujących stany wejść/wyjść PLC) spowoduje zmianę sygnatur. Przykładowo dla programu sterującego z Rys. 3 Network 2 zastąpiono kodem z Rys. 5 otrzymując program napisany niewłaściwie, niezgodnie z opisem słownym – programista uwzględnił jedynie sygnał X0 zamiast X0 i X1 (komentarz dla Network 2).

Sygnatura wejść bieżącego stanu programu sterującego uległa zmianie w porównaniu do poprawnie napisanego układu sterowania i wynosi D100 = 807E. Program monitorujący oprócz tworzenia sygnatur może zawierać również analizator sygnatur, czyli algorytm porównujący sygnatury otrzymane na każdym z etapów działania PROGRAM_INDEX z wzorcami, będącymi spodziewanymi stanami wejść/wyjść PLC. Analizator sygnatur może również stanowić np. zewnętrzny program analizujący pracę sterownika PLC, do którego dane przesyłane są poprzez jeden z protokołów komunikacji. Przykład wewnętrznego analizatora sygnatur przedstawiono na Rys. 6.

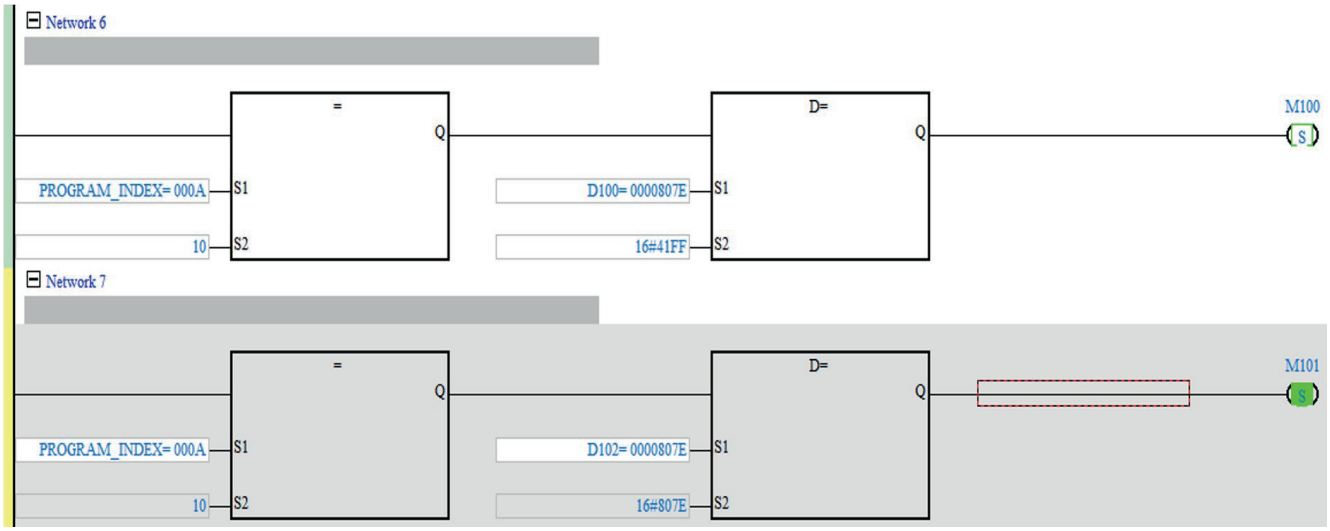
Markery M100 i M101 służą do stwierdzenia poprawności porównania z wzorcem. Dla poprawnie działającego układu sterowania z Rys. 3 określono sygnatury prawidłowo działającego programu (wzorce), a następnie wprowadzono losowe uszkodzenie w układzie wejść (niepoprawny układ sterowania z Rys. 5). Porównując otrzymane sygnatury z wzorcowymi (analizator



Rys. 5. Przykład programu napisanego niewłaściwie
Fig. 5. Example of a program written incorrectly



Rys. 6. Przykład analizatora sygnałów (poprawny program sterujący)
Fig. 6. An example of signals analyzer (with correct control program)



Rys. 7. Przykład analizatora sygnałów (niepoprawny program sterujący)
Fig. 7. An example of signals analyzer (with invalid control program)

sygnatur) można wykryć nieprawidłowe działanie układu sterowania (Rys. 7). Sygnatura wejść i wyjść jest porównywana z wzorcem, co tworzy najprostszy analizator sygnatur. W przykładzie z Rys.7 nie zgadza się sygnatura wejść cyfrowych sterownika PLC, spodziewana wartość to 41FF dla wejść X0 i X1 w stanie wysokim, pozostałe wejścia w stanie niskim. Oznacza to, że co najmniej jeden bit jest przekłamany i występuje duża różnica w CRC między obrazem stanu bieżącego wejść/wyjść, a obrazem wzorcowym (Rys. 7).

Analizator sygnatur można zastosować do przygotowania biblioteki sygnatur, w której opisane zostaną uszkodzenia odpowiadające danym sekwencjom. Po wyodrębnieniu fałszywej sygnatury można stwierdzić, która część oprogramowania nie pracuje prawidłowo. Za pomocą sygnatur możliwe jest również zgłaszanie ostrzeżeń dotyczących pracy programu. W przyszłości taka biblioteka znacznie uprości sprawdzanie poprawności działania układu cyfrowego oraz umożliwi szybką diagnozę wystąpienia awarii. Baza sygnatur może być tworzona wraz z rozwojem oprogramowania. Utworzenie i rozbudowa biblioteki sygnatur umożliwia szybką ocenę działania układu sterowania względem stanu wzorcowego.

W przypadku programów sterujących, w których mamy uwarunkowania liczbowe (liczniki) i złożone długie sekwencje sygnałów sterujących, może być potrzebne tzw. nałożenie maski na wybrane wejścia/wyjścia przed policzeniem CRC wejść / wyjść PLC zależnych od liczników i branie ich pod uwagę zawsze ze stałą wartością np. 1 przy liczeniu CRC. Stanowi to wadę tej metody, która dla szybkich, licznikowo zależnych sygnałów jest trudna do zastosowania, natomiast może sprawdzać się dla programów sekwencyjnych i pełnić również rolę dodatkowej autodiagnostyki układu sterowania. Druga wadą opisanej idei jest fakt, że nie istnieje algorytm umożliwiający na podstawie wyznaczonej sygnatury obliczyć, jakim danym wejściowym ona odpowiada. Jest więc konieczne opracowanie bazy sygnatur i na jej podstawie dopasowanie odpowiedzi układu sterowania do wzorca.

4. Podsumowanie

Wydaje się, że analiza sygnatur może być bardzo użyteczna w odniesieniu do walidacji programów sterujących. Podstawową zaletą metody jest jej uniwersalność. Zgodność otrzymanych sygnatur z wzorcowymi oznacza, z dużym prawdopodobieństwem, że oprogramowanie oraz sprzęt mogą swoje funkcje spełniać prawidłowo. W systemach sterowania mogą występować uśpione usterki, które pozostają niewykryte przez wiele lat i rzadko się pojawiają. Pisanie oprogramowania z uwzględnieniem analizy sygnatur może spowodować wyeliminowanie znacznej liczby uśpionych usterek na etapie walidacji oprogramowania.

Przedstawiona metoda może być zastosowana do dowolnych zmiennych sterownika PLC, które zostaną „ułożone" w spójny ciąg bitów, dla którego można obliczyć w opisany sposób sygnaturę. Na przedstawionym przykładzie działania analizatora sygnatur może być tworzone oprogramowanie walidujące poprawność pracy programowanego układu sterowania bazujące na oprogramowaniu i sterownikach innych producentów oraz na innym jak drabinkowy języku programowania. Zastosowanie analizy sygnatur może dostarczyć dowody weryfikacji i walidacji oprogramowania systemu sterowania oraz informacji, że proces integracji programowej maszyny został przeprowadzony z zastosowaniem procesów inżynierii bezpieczeństwa.

Bibliografia

1. Bonaldi E.L., de Lacerda de Oliveira L.E., Borges da Silva J.G., Lambert-Torresm G., Borges da Silva L.E., *Predictive Maintenance by Electrical Signature Analysis to Induction Motors*, [In:] Induction Motors – Modelling and Control; DOI: 10.5772/48045.
2. Broel-Plater B., *Układy wykorzystujące sterowniki PLC, Projektowanie algorytmów sterowania*, Warszawa 2021.
3. Huber Ch., *A generic approach for static code analysis of programs*, Master Thesis, Johannes Kepler University Linz, Austria, 2016.
4. Ierace S., Gaiardelli P., Fumagalli L., Dovere E., Macchi M., *Industrial applicability of Electric Signature Analysis as a diagnostic tool for Condition Based Maintenance: a case study*, 2011.
5. Jadczak K., Białek R., *Stanowisko laboratoryjne do badania niezawodności układów cyfrowych z wykorzystaniem analizy sygnatur*, „Journal of Konbin", Vol. 45, Nr 1, 2018, DOI: 10.2478/jok-2018-0009.
6. Laube J., *Analizator sygnatur – uniwersalny przyrząd diagnostyczny do urządzeń cyfrowych*, Warszawa 1982.
7. Pratesh J., Wadhwani A.K., Mulchandani K.B., *Machine Fault Signature Analysis*, "International Journal of Rotating Machinery", 2008, DOI: 10.1155/2008/583982.
8. Schweitzer A., *Improving the Safety Level of Programmable Electronic Systems (Pes) by Applying the Concept of Signature Analysis*, 1986, [In: Safety and Reliability of Programmable Electronic Systems], 199-209, DOI: 10.1007/978-94-009-4317-9_21.
9. Zhang M., Chen Ch.-Y., Kao B.-Ch., Qamsane Y., Shao Y., Lin Y., Shi E., Mohan S., Barton K., Moyne J., Mao Z.M., *Towards Automated Safety Vetting of PLC Code in Real-World Plants*, Proceedings of 2019 IEEE Symposium on Security and Privacy (SP), DOI: 10.1109/SP.2019.00034.

Inne źródła

10. DVP-ES2/EX2/SS2/SA2/SX2/SE&TP Operation Manual – Programming, 2014
11. *STM32 Nucleo-64 development board with STM32G491RE MCU, supports Arduino and ST morpho connectivity*, https://www.st.com/en/evaluation-tools/nucleo-g491re.html
12. https://m.media-amazon.com/images/I/713XTXtWrML._SL1001_.jpg
13. https://induprogress.pl/produkty/sterowanie-i-wizualizacja-procesow/sterowniki-plc-serii-dvp-slim/

Improving the Functional PLC Software Safety Through the Signature Analysis

Abstract: The article presents the idea of the response compression technique called signature analysis used to validate PLC software in terms of functional control safety. The method of implementation of the developed validation idea, having practical application, was presented on the simulation examples. The research results based on a selected example are presented.

Keywords: signature analysis, hidden security breaches, safety software PLC validation, functional control safety of machines

dr hab. inż. Marcin Szuster, prof. PRz
mszuster@prz.edu.pl
ORCID: 0000-0002-9713-4456

Absolwent Wydziału Budowy Maszyn i Lotnictwa Politechniki Rzeszowskiej. W 2007 r. uzyskał stopień magistra inżyniera w specjalności mechatronika, w 2012 r. stopień doktora nauk technicznych w dyscyplinie mechanika z wyróżnieniem, a w 2020 r. stopień doktora habilitowanego nauk inżynieryjno-technicznych w dyscyplinie inżynieria mechaniczna. W 2011 r. podjął pracę w Katedrze Mechaniki Stosowanej i Robotyki Wydziału Budowy Maszyn i Lotnictwa Politechniki Rzeszowskiej, gdzie pracuje do dziś. Jego zainteresowania naukowe obejmują tematykę modelowania i sterowania robotów, implementacji metod sztucznej inteligencji, takich jak układy z logiką rozmytą czy sztuczne sieci neuronowe, oraz metod optymalnych, w układach sterowania ruchem mobilnych robotów kołowych. Jest członkiem Polskiego Towarzystwa Mechaniki Teoretycznej i Stosowanej oraz autorem/współautorem ponad 45 publikacji naukowych opublikowanych w wydawnictwach krajowych i zagranicznych, w tym jednej monografii w języku angielskim.

mgr inż. Bartłomiej Kozioł
bartlomiej.koziol@gmail.com
ORCID: 0000-0002-5998-5890

Absolwent Wydziałów Inżynierii Mechanicznej i Robotyki oraz Elektrotechniki, Automatyki, Informatyki i Inżynierii Biomedycznej Akademii Górniczo-Hutniczej w Krakowie. W 2011 r. uzyskał stopień inżyniera, a w 2012 r. stopień magistra inżyniera na kierunku elektrotechnika. Po studiach podjął zatrudnienie w charakterze nauczyciela przedmiotów zawodowych w Centrum Transferu Nowoczesnych Technologii Wytwarzania w Mielcu. W 2012 r. rozpoczął pracę w przemyśle samochodowym jako konstruktor automatyki, gdzie zdobył cenne doświadczenie zawodowe w trakcie prowadzenia licznych projektów wdrożeniowych zautomatyzowanych linii produkcyjnych. Od 2019 r. jest uczestnikiem studiów III stopnia w Szkole Doktorskiej Nauk Inżynieryjno-Technicznych na Politechnice Rzeszowskiej. Jego zainteresowania naukowe obejmują tematykę implementacji rozwiązań umożliwiających zwiększanie bezpieczeństwa personelu obsługującego zintegrowane linie produkcyjne przez wypracowanie zautomatyzowanych metod wspomagających walidację algorytmów sterowania, również z zastosowaniem nowoczesnych metod sztucznej inteligencji.