Ukryte naruszenia bezpieczeństwa w układach automatycznego sterowania procesami technologicznymi

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

Bartłomiej Kozioł
Szkoła Doktorska Nauk Inżynieryjno-Technicznych na Politechnice Rzeszowskiej

Streszczenie: Postępująca automatyzacja i robotyzacja w zakładach pracy oraz coraz większa złożoność systemów sterowania maszyn zespolonych powodują konieczność ciągłej poprawy bezpieczeństwa funkcjonalnego maszyn przez prawidłową walidację systemów bezpieczeństwa. Mimo przeprowadzonego procesu walidacji potencjalne błędy programowe mogą ujawnić się w trakcie użytkowania maszyny jako ukryte naruszenia bezpieczeństwa. W artykule przedstawiono przykłady naruszeń bezpieczeństwa rzeczywistych zespołów maszyn oraz próby wdrażania rozwiązań zautomatyzowanych mechanizmów do wykrywania problemów z bezpieczeństwem. Kolejnym aspektem poruszanym w artykule jest nowe podejście do wykrywania ukrytych naruszeń bezpieczeństwa. Dzięki zastosowaniu modelu „cyfrowego bliźniaka" maszyny, programu generującego sekwencję zdarzeń do testowania systemów sterowania i zastosowanie wirtualnej rzeczywistości (wizualna weryfikacja programów bezpieczeństwa) możliwa jest maksymalizacja funkcji bezpieczeństwa funkcjonalnego zespołów maszyn.

Słowa kluczowe: ukryte naruszenia bezpieczeństwa, walidacja systemu bezpieczeństwa, programowalne sterowniki bezpieczeństwa, bezpieczeństwo funkcjonalne maszyn zintegrowanych, automatyczna walidacja

1. Wprowadzenie

W obecnym stanie prawnym nie trzeba mieć specjalnych uprawnień, aby projektować systemy sterowania odpowiedzialne za bezpieczeństwo. Wraz ze wzrostem stopnia złożoności zespołów maszyn zwiększa się możliwość popełnienia błędów, przeoczenia czasowych związków przyczynowo-skutkowych przy pisaniu oprogramowania sterującego, w tym oprogramowania bezpieczeństwa. Błędy oprogramowania są w swojej naturze zawsze systematyczne. Wynikają one ze sposobu opracowania, napisania lub kompilowania programu. Tym samym z realizacji procesu projektowania, a nie z działania programu [3]. Brak wiedzy i doświadczenia programisty może doprowadzić do zaprogramowania układu sterowania bez, lub z niewystarczającą strukturą kontrolowanych działań obsługi, co przy nieodpowiednich sposobach obsługi maszyny prowadzi do niezamierzonego działania układu sterowania [2].

Pracownicy producenta odpowiedzialni za instalację i uruchomienie maszyny przed oddaniem jej do użytku powinni wielokrotnie testować i walidować elementy systemu sterowania bezpieczeństwem w celu potwierdzenia maksymalnego bezpieczeństwa funkcjonalnego oraz niezawodności systemu bezpieczeństwa. Dla skomplikowanych pod względem sterowania zespołów maszyn powołuje się zespół walidacyjny, który m.in. symuluje usterki elementów sterowniczych, sprawdza, jak zadziałają instalacje zabezpieczające i urządzenia ochronne, sprawdza działanie maszyny po odłączeniu zasilania, aby zbadać, czy powróci ona do bezpiecznego stanu, inicjuje procedury awaryjnego zatrzymania maszyny, testuje oprogramowanie [5]. Często ze względu na napięty harmonogram prac wdrożeniowych czas na testy oprogramowania, w tym oprogramowania bezpieczeństwa, jest ograniczany do minimum. W przypadku dużych systemów – maszyn zintegrowanych przeprowadzane są walidacje częściowe przed połączeniem poszczególnych elementów, a następnie sprawdzane są skutki połączenia w zakresie zachowania bezpieczeństwa sterowania. Walidacja potwierdza, że maszyny lub urządzenia spełniają wymogi bezpieczeństwa zawarte w przepisach prawa PL, UE i normach PN-EN, a pra-

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

Artykuł recenzowany
nadesłany 26.05.2021 r., przyjęty do druku 18.06.2021 r.

cownicy są bezpieczni podczas ich obsługi, przy czym można ją wykonywać w warunkach rzeczywistych lub symulowanych [5].

Najważniejsze normy regulujące proces walidacji to [6]:

- Norma PN-EN ISO 13849-1/-2 Bezpieczeństwo maszyn – Elementy systemów sterowania związane z bezpieczeństwem – Część 2: Walidacja.
- Norma PN-EN 60261 Bezpieczeństwo maszyn – Bezpieczeństwo funkcjonalne elektrycznych, elektronicznych i programowalnych elektronicznych systemów sterowania związanych z bezpieczeństwem.
- Norma PN-EN 61508 Bezpieczeństwo funkcjonalne elektrycznych/elektronicznych/programowalnych systemów związanych z bezpieczeństwem.

Mimo przeprowadzonego procesu walidacji zgodnie z normami, potencjalne błędy programowe mogą ujawnić się w trakcie użytkowania maszyny jako ukryte naruszenia bezpieczeństwa. Pod pojęciem tym kryją się m.in. niedoskonałości oprogramowania, potencjalnie mogące wywołać niezamierzone działanie układu sterowania maszyny (w tym układu bezpieczeństwa). Nieoczekiwane błędy logiczne mogą spowodować poważne problemy, takie jak obrażenia fizyczne, straty materialne czy kosztowne przestoje linii.

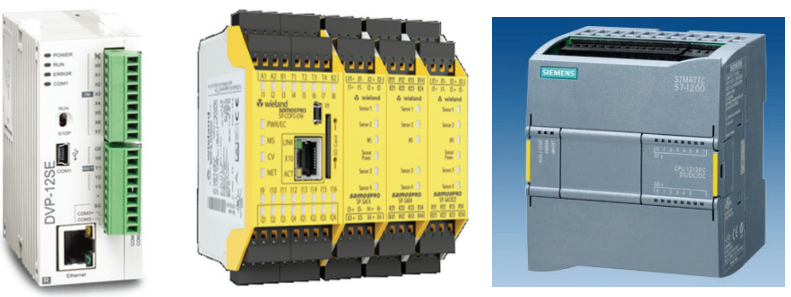
Algorytm sterowania maszyny zintegrowanej, zapisany w wybranym języku programowania, jest realizowany przez mikroprocesorowe urządzenie przetwarzające. Często wybieranymi do tego zadania rozwiązaniami są sterowniki przemysłowe, które w układach automatyki nie pracują jednak samodzielnie (ang. *PLC is NOT Working Alone*), wymieniając informacje oraz sygnały z zewnętrznymi czujnikami, aktuatorami, innymi sterownikami. Testowanie oprogramowania PLC wymaga podania zewnętrznych sygnałów wyzwalających zdarzenia, co więcej sygnały te muszą najczęściej przychodzić w określonym czasie i w określonej kolejności czasowej [1]. Udział człowieka (operatora) w procesie sterowania wprowadza do tego procesu nieprzewidywalność związaną np. z różnym czasem reakcji operatorów na zdarzenia oraz możliwością postępowania niezgodnego z instrukcjami obsługi maszyn. Prawidłowo wykonany układ sterowania powinien zabezpieczyć człowieka, maszyny jak i proces technologiczny przed nieprawidłową reakcją operatora. Na rzeczywistym obiekcie optymalizuje się czasy, parametry technologiczne (ciśnienia, prędkości itp.), parametry bloków funkcyjnych bezpieczeństwa, co wpływa na bezpieczeństwo użytkowania. Zastosowanie PLC w maszynach zespolonych przyniosło elastyczność i możliwość konfiguracji systemu sterowania. Równocześnie wprowadziło złożoność, a tym samym niepewność, zwłaszcza co do krytycznych dla bezpieczeństwa funkcji oprogramowania.

W artykule przedstawiono sprzętowo-programową realizację funkcji bezpieczeństwa wraz z przykładami naruszeń bezpieczeństwa rzeczywistych maszyn zintegrowanych. Próby wdrażania rozwiązań zautomatyzowanych mechanizmów wykrywania ukrytych naruszeń bezpieczeństwa przedstawione są w rozdziale dotyczącym automatycznej walidacji systemów sterowania bezpieczeństwem. Nowe podejście do wykrywania ukrytych naruszeń bezpieczeństwa umożliwi maksymalizację funkcji bezpieczeństwa funkcjonalnego maszyny zintegrowanej.

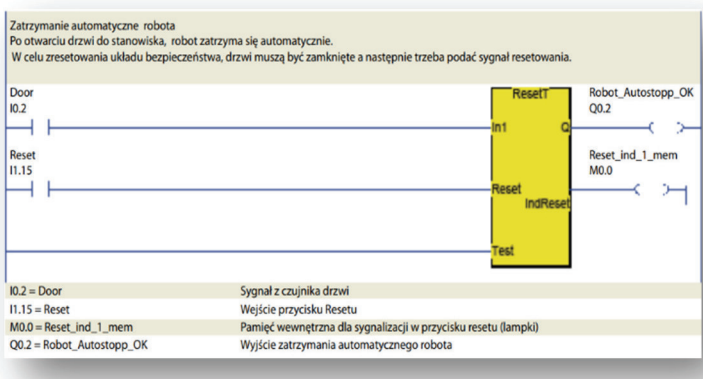
2. Sprzętowo-programowa realizacja funkcji bezpieczeństwa

Współczesne konstrukcje maszyn zawierają złożone systemy sterowania oparte głównie na programowalnych i konfigurowalnych sterownikach logicznych PLC (rys. 1a), których szczególnym rozwinięciem są programowalne sterowniki bezpieczeństwa sPLC (rys. 1b). Te z kolei mogą pracować niezależnie od sterownika PLC lub być z nim zintegrowane siPLC (ang. *safety integrated PLC* – rys. 1c).

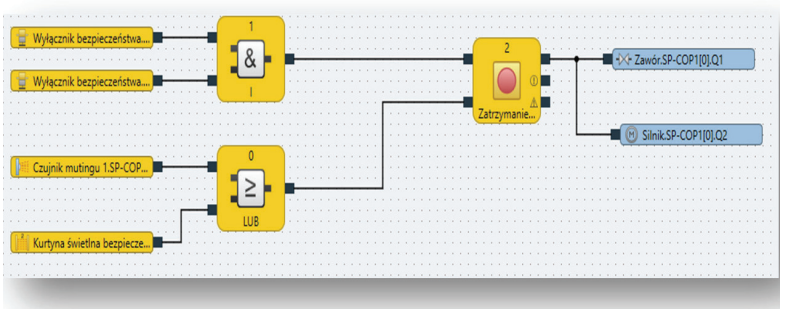
Sterowniki sPLC oraz siPLC umożliwiają połączenie różnych elementów bezpieczeństwa i sterowania elementami bezpieczeństwa. Są to elementy maszyny wymagające od programisty szczególnej uwagi i doświadczenia w programowaniu. Wybór sprzętowy sterownika bezpieczeństwa wiąże się z wyborem języka programowania. Do najbardziej popularnych zalicza się:



Rys. 1 a) Przykład PLC - Delta SE [24] b) sPLC - Wieland SamosPro [25] c) siPLC – Siemens S7-1200F [26]
Fig. 1 a) PLC example - Delta SE [24] b) sPLC - Wieland SamosPro [25] c) siPLC – Siemens S7-1200F [26]



Rys. 2. Przykład użycia drabinkowego języka programowania sterownika PLUTO ABB [8]
Fig. 2. An example of using the ladder programming language of the PLUTO ABB controller [8]



Rys. 3. Przykład zastosowania graficznego języka programowania – SamosPro Wieland
Fig. 3. An example of using a graphical programming language – SamosPro Wieland

Język drabinkowy (ang. *Ladder Diagram*),
Najbardziej popularny język graficzny programowania siPLC. Rysunek 2 przedstawia przykład programowania sterownika Pluto (ABB). Programowanie odbywa się za pomocą języka drabinkowego lub algebry Boole'a – z licznikami, pamięcią zmiennych, rejestrami, programowaniem sekwencyjnym oraz blokami funkcyjnymi zatwierdzonymi przez TUV [8].

Język schematów blokowych FBD (ang. *Function Block Diagram*),
Język graficzny programowania jest szczególnie popularny w sterownikach sPLC. Przykład z rys. 3 przedstawia użycie biblioteki certyfikowanych bloków funkcyjnych.

Język tekstowy wysokiego poziomu – safety C (uproszczony)
Język tekstowy wysokiego poziomu umożliwia programowanie sterowników bezpieczeństwa z uwzględnieniem specjalnych ograniczeń funkcji i możliwości języka C w celu uniknięcia możliwości kompilacji programów o niejednoznacznej semantyce [28]. W porównaniu z programowaniem opartym na blokach funkcyjnych, języki wysokiego poziomu umożliwiają programiście sprawniejsze tworzenie zaawansowanych algorytmów sterowania. Język programowania wysokiego poziomu najczęściej stosowany jest w droższych jednostkach PLC, sPLC oraz siPLC. Na rys. 4 pokazano przykład użycia struktury w sterowniku Beckhoff do zaprogramowania bezpiecznych stanów wejść sterownika.

Język tekstowy niskiego poziomu
Język tekstowy niskiego poziomu ma składnię podobną do języków typu asembler. Program w języku niskiego poziomu składa się z sekwencji rozkazów, z których każdy kolejny zaczyna się w nowej linii (rys. 5). Stosowanie uproszczonych języków programowania w systemach sterowania bezpieczeństwem, jak przedstawiony powyżej, umożliwiło ograniczenie błędów systematycznych oprogramowania [12].

Zasady programowania graficznych języków programowania są szczegółowo opracowywane przez producentów oprogramowania. Dzięki zastosowaniu certyfikowanych bloków funkcjonalnych do obsługi funkcji bezpieczeństwa programista nie pisze oprogramowania od podstaw, tylko stosuje procedury zawarte w bibliotekach (rys. 6).

Głównym zadaniem programisty systemu bezpieczeństwa staje się więc konfiguracja systemu bezpieczeństwa poprzez:

a) wybór odpowiedniego sprzętu bezpieczeństwa (konfiguracja sprzętowa systemu) (rys. 7, etap A),
b) wybór i konfiguracja wejść/wyjść systemu bezpieczeństwa (rys. 7, etap B),
c) zaprogramowanie algorytmu sterującego bezpieczeństwem (logika) (rys. 7, etap C).

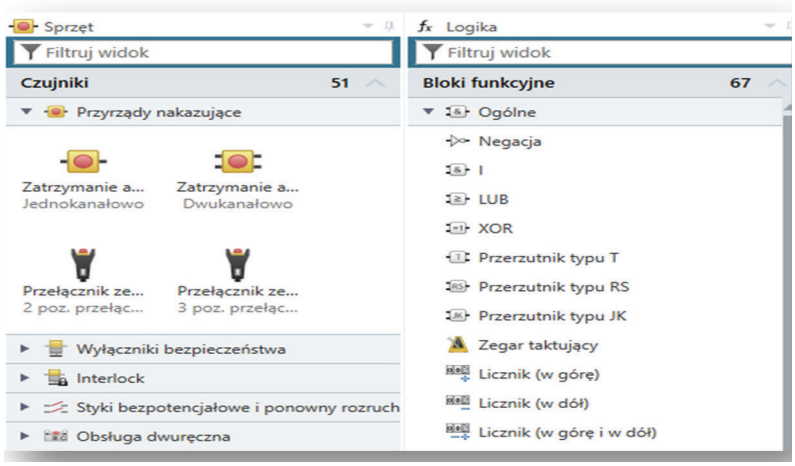
Kolejne kroki programowania funkcji systemu bezpieczeństwa przedstawiono na przykładzie oprogramowania SamosPro Wieland na rys. 7.

```
//! Struct providing input data of the corresponding safety alias devices
struct SafetyInputs
{
    //! ..\Alias Devices\EL1904_FSoE_211.sds
    struct _EL1904_FSoE_211
    {
        safeBOOL InputChannel1;
        safeBOOL InputChannel2;
        safeBOOL InputChannel3;
        safeBOOL InputChannel4;
    } EL1904_FSoE_211;

};
```

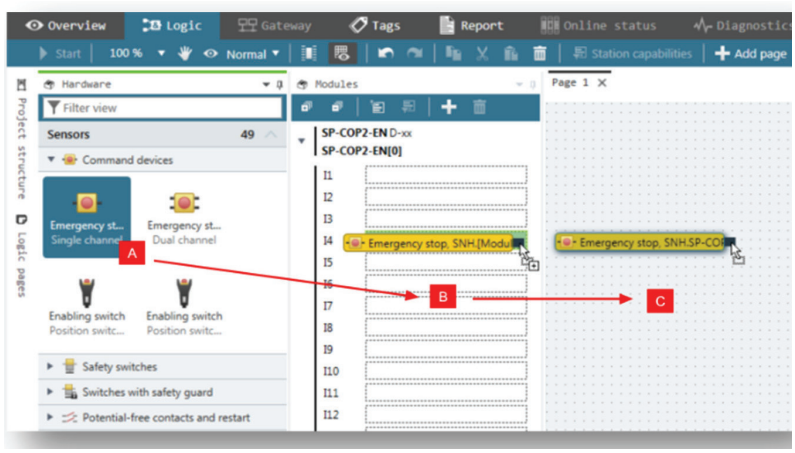
Rys. 4. Przykład zastosowania języka programowania wysokiego poziomu – Beckhoff TwinCat [9]
Fig. 4. An example of the use of a high-level programming language – Beckhoff TwinCat [9]

```
S0.1_00
Q0.1 = I0.2
J(+1) = Q0.10*M0.7

S0.1_01
S(Q0.2) = I0.3
J(10) = M0.10

S0.1_10
R(Q0.2) = I0.4
J(0) = GM0.0
```

Rys. 5. Przykład zastosowania języka programowania niskiego poziomu – Pluto ABB [8]
Fig. 5. An example of the use of a low-level programming language – Pluto ABB [8]



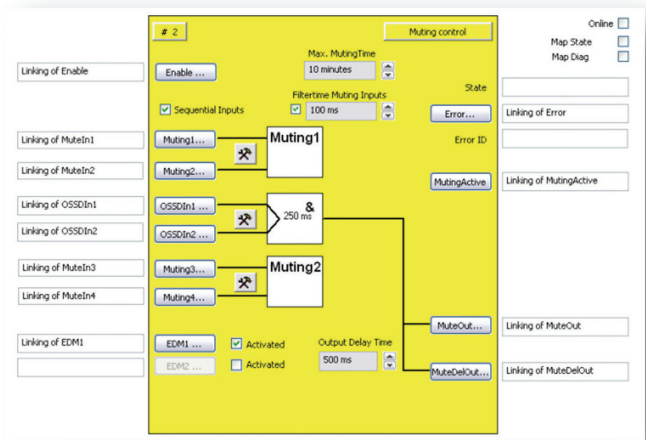
Rys. 6. Przykład sprzętowych i logicznych bibliotek programu Samos Pro Wieland w języku FBD
Fig. 6. Example of hardware and logical libraries of Samos Pro Wieland in FBD language



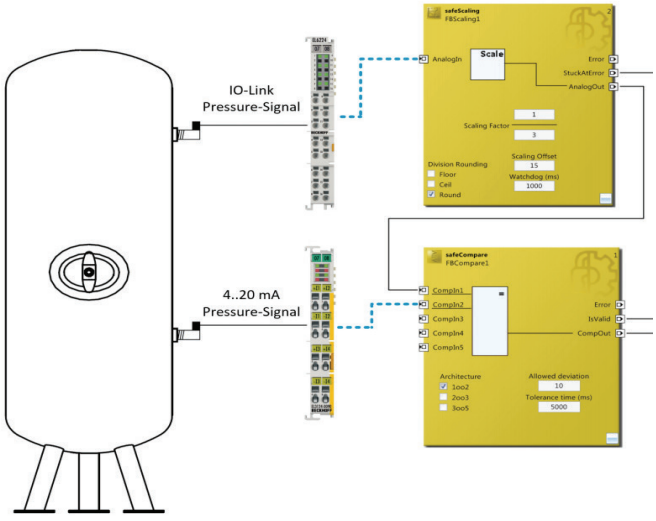
Rys. 7. Przykład konfiguracji systemu bezpieczeństwa – SamosPro Wieland [11]
Fig. 7. Example of a security system configuration – SamosPro Wieland [11]

Programista odpowiada więc za prawidłowe użycie wejść, wyjść oraz sygnałów dynamicznych związanych z bezpieczeństwem. W każdym z języków programowania sterowników bezpieczeństwa występują elementy (np. w postaci bloków funkcyjnych) do sparametryzowania, odpowiadające za bezpieczeństwo. Przykładem parametryzowanego bloku funkcyjnego FB (ang. *Function Block*) może być FB muting (rys. 8).

Blok muting umożliwia realizację czasowego wyłączenia funkcji ochronnej, np. w czasie transportu materiału do strefy ochronnej. Wyjście bloku funkcyjnego pozostaje aktywne mimo przerwania podłączonego czujnika bezpieczeństwa (np. kurtyny świetlnej) [7], co w standardowych warunkach pracy prowadziłoby do awaryjnego zatrzymania maszyny zespolonej. Tylko w jednym przykładowym FB typu muting do sparametryzowania są co najmniej trzy parametry, które odpowiadają za bezpieczeństwo: maksymalny czas wstrzymania reakcji podczas pracy elektroczułych urządzeń zabezpieczających, czas filtrowania wejść bloku oraz czas opóźnienia wyjścia. W rzeczywistych dużych maszynach zintegrowanych (liniach zespolonych) mogą być zastosowane setki bloków funkcyjnych bezpieczeństwa, co może wymagać doboru tysięcy parametrów odpowiedzialnych za bezpieczeństwo, które trzeba poprawnie skonfigurować i walidować. Często zmiana jednego parametru ma wpływ na działanie innych bloków funkcyjnych (rys. 9).



Rys. 8. Przykład konfiguracji bloku muting w oprogramowaniu Beckhoff TwinSafe [7]
Fig. 8. Example of a muting block configuration in the Beckhoff TwinSafe software [7]



Rys. 9. Zmiana parametru „Scale" ma wpływ na pozostałe bloki bezpieczeństwa [13]
Fig. 9. Changing the "Scale" parameter affects the remaining safety blocks [13]

Wyjście sygnałowe z pierwszej funkcji bezpieczeństwa (SafeScaling) może być wejściem sygnałowym do kolejnej funkcji (SafeCompare). Na przykładzie z rys. 9 – błędnie podane wartości skalowania powodują rozbieżność przy porównywaniu wartości sygnału ciśnienia z dwóch różnych czujników.

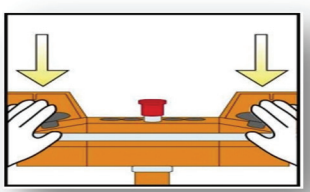
3. Przykłady naruszeń bezpieczeństwa

Autorzy w pracy zawodowej spotkali się z sytuacją, w której do sterowania linią zespoloną zastosowano protokół MODBUS TCP tak, że urządzenie sterujące stanowił komputer (nie oparty na systemie czasu rzeczywistego) z oprogramowaniem sterującym linią. Sterował on bezpośrednio wyjściami PLC za pomocą przypisanych rejestrów MODBUS TCP. Możliwość bezpośredniego sterowania wyjściami PLC przez system, który może wygenerować odpowiedź po diametralnie różnym czasie (przykładowo raz może to być 5 ms, a innym razem 5 s) jest naruszeniem bezpieczeństwa (Dyrektywa Maszynowa [30] pkt. 1.2.1. – defekty sprzętu komputerowego i oprogramowania układu sterowania nie mogą prowadzić do sytuacji zagrożenia).

W nowoczesnych sterownikach bezpieczeństwa możliwe jest przypisanie ról, co daje możliwość:
- operatorom – podglądu programu bezpieczeństwa,
- pracownikom sekcji utrzymania ruchu – resetowania systemu bezpieczeństwa po awarii maszyny,
- integratorowi – zmiany krytycznych parametrów związanych z bezpieczeństwem.

Brak odpowiedniego przypisania ról lub zabezpieczenia przed zmianą krytycznych parametrów systemu przez osoby nieuprawnione stanowi naruszenie bezpieczeństwa (Dyrektywa Maszynowa pkt. 1.2.1. – układy sterowania muszą być zaprojektowane i wykonane w sposób, który zapobiegnie powstawaniu sytuacji zagrożenia). Autorzy spotkali się z sytuacją, gdzie automatyczny magazyn wysokiego składowania miał nadany certyfikat zgodności CE, natomiast możliwa była zmiana krytycznych parametrów ruchu z poziomu HMI przez operatora, co skutkuje niezgodnością z dyrektywą 2006/42/WE w zakresie pkt. 1.2.1.

Potencjalne naruszenie bezpieczeństwa omówiono na przykładzie niepoprawnego zastosowania elementu sterowania dwuręcznego. Pulpit sterowania dwuręcznego składa się z dwóch przycisków załączanych dłońmi w standardowym trybie działania oraz przycisku zatrzymania awaryjnego STOP (rys. 10).



Rys. 10. Pulpit sterowania dwuręcznego [10]
Fig. 10. Two-hand control desk [10]

Autorzy kilkukrotnie spotkali się z niepoprawnym użyciem tego elementu w zaawansowanych aplikacjach renomowanych producentów maszyn. Walidacja wykazała, że przyciski sterowania dwuręcznego działają prawidłowo, tzn. po naciśnięciu przycisków rozpoczyna się realizacja procesu, po puszczeniu przycisków realizacja procesu jest zatrzymywana. Jednak szczegółowa analiza programu sterującego wykazała występowanie błędów zmniejszających bezpieczeństwo, takich jak:

1. **Użycie tylko jednego kanału wejściowego jako sygnał dla sterownika PLC (rys. 11).**

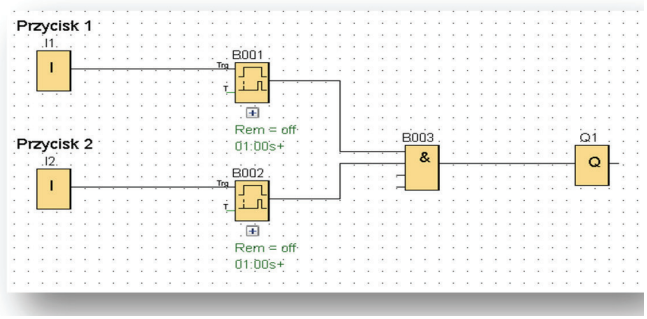
Połączenie szeregowe elementów Przycisk1 oraz Przycisk2 jest niepoprawne z powodu:

– braku odporności układu sterowana na uszkodzenia mechaniczne przycisków, co więcej do układu sterowania nie jest dostarczana informacja o tym, który przycisk uległ awarii,
– możliwości ominięcia zabezpieczenia przez operatora, np. poprzez sklejenie taśmą jednego z przycisków i uruchomienie procesu naciśnięciem jedynie drugiego przycisku.

Rys. 11. Niepoprawny układ szeregowy przycisków (Logo Soft Comfort)
Fig. 11. Incorrect layout of the buttons in series (Logo Soft Comfort)

2. Użycie dwóch kanałów PLC, lecz bez wymuszenia puszczenia obydwu przycisków (rys. 12)

W układzie niepoprawnym (rys. 12) zastosowano dwa kanały wejściowe PLC (dobór liczby kanałów musi wynikać z normy [29] oraz analizy ryzyka). Program z rys. 12 został napisany dla sterownika PLC, którego wyjścia nie zapewniają bezpieczeństwa w układach wymagających sterowania dwuręcznego.



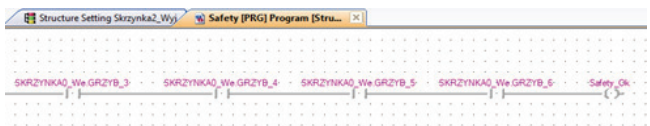
Rys. 12. Niepoprawny układ dwukanałowy sterowania dwuręcznego (Logo Soft Comfort)
Fig. 12. Incorrect two-channel two-hand control system (Logo Soft Comfort)

Do układu sterowania jest dostarczana informacja o tym, który przycisk uległ awarii, lecz nie jest ona obsłużona programowo. Nadal istnieje możliwość ominięcia zabezpieczenia przez operatora. Prawidłowo napisany program dla układu sterowania dwuręcznego powinien wykorzystać w układzie sterowania – klasyczny zespół sterownik PLC z przekaźnikiem bezpieczeństwa lub sterownik ze zintegrowanym (programowalnym) bezpieczeństwem umożliwiającym obsługę sterowania dwuręcznego przez odtworzenie na drodze programowej wymaganego przez przepisy sposobu zachowywania się takiego sterownika. Prawidłowa programowa realizacja układu sterowania powinna wymusić najpierw puszczenie obydwu przycisków, a następnie ich wciśnięcie, aby kolejny cykl realizacji procesu mógł wystartować (ważny jest czas wystąpienia zdarzeń – aktywacja obydwu przycisków z odstępem czasu mniejszym niż 500 ms (synchroniczność) oraz kolejność ich wystąpienia).

Wystarczy mieć wiedzę odnośnie działania przycisków sterowania dwuręcznego (PN-EN 574+A1:2010) i już można zacząć budować system ekspertowy w oparciu o reguły działania prawidłowego układu sterowania (tab.1). System ekspertowy może zostać zastosowany do zautomatyzowania walidacji systemu bezpieczeństwa i tym samym do zmniejszenia liczby programowych, ukrytych naruszeń bezpieczeństwa.

Podany przykład systemu reguł dla przycisków sterowania dwuręcznego ma na celu przedstawienie idei automatyzacji walidacji. Dla reguły R1 w zależności od oceny ryzyka dla układów o najwyższym poziomie wymaganej niezawodności mogą być stosowane przyciski z dwoma stykami, co może wymagać zastosowania nawet czterech wejść sterownika. Na dużych liniach maszyn zespolonych system reguł pracy maszyny najlepiej znany jest operatorom, osobom z utrzymania ruchu oraz serwisantom. Wiedza i doświadczenie tych osób zdobyte w trakcie pracy linii zespolonej (w postaci systemu reguł) mogą również posłużyć do poszukiwania ukrytych naruszeń bezpieczeństwa.

Drugim przykładem ukrytego naruszenia bezpieczeństwa może być użycie wejść/wyjść PLC do sterowania funkcjami bezpieczeństwa (Dyrektywa Maszynowa 1.2.4.3 – zatrzymanie awaryjne musi możliwie najszybciej zatrzymywać niebezpieczny proces, bez stwarzania dodatkowego ryzyka). PLC może zatrzymać proces jedynie technologicznie. Przykład z rys. 13 przedstawia cztery sygnały wyjściowe przycisków bezpieczeństwa (przyciski bezpieczeństwa normalnie zamknięte), będące wejściami PLC, połączone z zastosowaniem operatora logicznego „i" oraz przypisane do wyjścia „Safety_Ok". Wyjście realizuje jeden z warunków załączania ruchu taśmy produkcyjnej w przypadku braku naciśnięcia przycisku awaryjnego. Naciśnięcie jednego z przycisków awaryjnych nie spowoduje bezpiecznego zatrzymania ruchu taśmy produkcyjnej, a jedynie zatrzymanie technologiczne. To tak jak z wyłączeniem telewizora za pomocą pilota (zatrzymanie technologiczne) lub wyłączeniem telewizora przez wypięcie wtyczki kabla zasilającego z gniazda (bezpieczne zatrzymanie urządzenia).



Rys. 13. Przykład zastosowania sygnałów informacyjnych z wyłączników bezpieczeństwa do sterowania procesem (przykład wykonany w GX Works 2)
Fig. 13. Example of the use of information signals from safety switches for process control (example taken in GX Works 2)

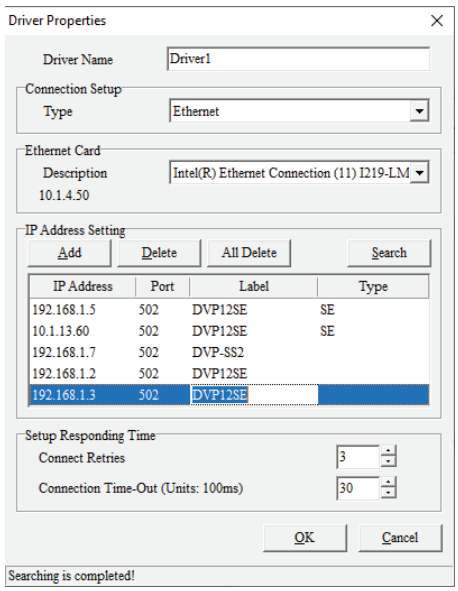
Wejścia, które są wymagane w programie bezpieczeństwa należy odczytać bezpośrednio w programie bezpieczeństwa. Należy unikać przejść sygnałów bezpieczeństwa przez program użytkownika oraz tworzenia skomplikowanych ścieżek sygnałów.

Kolejnym przykładem ukrytego naruszenia bezpieczeństwa (Dyrektywa Maszynowa 1.2.1 – parametry maszyny nie mogą zmieniać się w sposób niekontrolowany, jeżeli taka zmiana może prowadzić do sytuacji zagrożenia) jest brak ochrony bloków bezpieczeństwa przed dostępem osób nieuprawnionych oraz brak zabezpieczenia programu PLC przed nadpisaniem innym kodem (np. w przypadku użycia na linii kilku sterowników tego samego typu, które są widoczne w sieci) (rys. 14). Programista może przez pomyłkę przeprogramować niewłaściwy PLC.

Szczególnymi miejscami, w których można poszukiwać ukrytych naruszeń bezpieczeństwa w kodzie programu są wywoła-

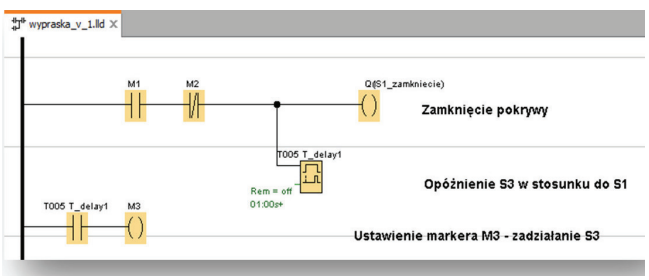
Tab. 1. Przykład systemu reguł dla przycisków sterowania oburęcznego
Tab. 1. An example of a rule system for two-hand control buttons

System reguł:
$\begin{cases} R1: Sterowanie\ dwuręczne - weryfikacja\ liczby\ kanałow\ wejściowych\ PLC \\ R2: Wyzwolenie\ Q1\ za\ pomoca\ obydwu\ przycisków\ z\ zastosowaniem\ synchroniczności \\ R3: Ponowne\ wyzwolenie\ sygnalu\ Q1\ możliwe\ po\ uprzednim\ resecie\ I1\ i\ I2 \end{cases}$



Rys. 14. Przykład wyszukiwania PLC w sieci sterowników Delta-DVP
Fig. 14. Example of searching for a PLC in the network of Delta-DVP controllers

nia zegarów (ang. *timers*). Przy nieodpowiednio napisanym programie zmiana nastaw czasowych, np. czasu opóźnienia załączania będzie miała wpływ na działanie, funkcjonalność i bezpieczeństwo sterowania (Dyrektywa Maszynowa 1.2.1 – błędy w układach logicznych sterowania nie mogą prowadzić do sytuacji zagrożenia). W trakcie walidacji należy ograniczyć możliwość zmiany parametrów nastaw czasowych do wartości bezpiecznych i przetestowanych. Pokazany na rys. 15 przykład przedstawia użycie bloku typu timer opóźniający – T_delay1, przy czym ustawiony za krótki czas spowoduje, że siłownik pneumatyczny (S3) będzie miał kolizję mechaniczną

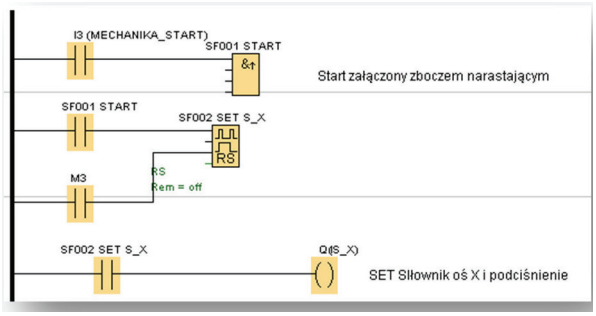


Rys. 15. Przykład użycia zegara do sterowania procesem (przykład Logo Soft Comfort)
Fig. 15. Example of using the clock to control the process (example Logo Soft Comfort)

z siłownikiem pneumatycznym – (S1). Rozwiązaniem bezpieczniejszym jest zastosowanie czujników wykrywający położenie tłoczyska.

W maszynach, w których użyto „impulsów" do sterowania procesem (rys. 16) ważne jest odpowiednio długie trwanie sygnału, aby urządzenia współpracujące z sygnałem mogły wykryć zmianę stanu. Jeśli w przykładzie marker M3 załączy się zbyt szybko, wówczas siłownik osi X oraz eżektor na siłowniku X mogą nie zadziałać prawidłowo.

Panel operatorski realizujący interfejs HMI to niezbędny i wygodny element w pracy operatora maszyny. Sygnały z panelu HMI nie są związane z bezpieczeństwem i nie są monitorowane [15]. Panel HMI często jednak używany jest do wizualizacji stanu bezpieczeństwa maszyny, żądania potwierdzania błędów. W takim przypadku zachodzi potrzeba wymiany danych między standardowym programem użytkownika (zawierającym obsługę HMI), a programem bezpieczeństwa. Komunikacja między HMI i sterownikiem jest acykliczna. Podczas przetwarzania programu bezpieczeństwa HMI może mieć możliwość zapisu danych a program bezpieczeństwa będzie te dane odczytywał. Może to spowodować uszkodzenie danych w programie bezpieczeństwa, co spowoduje zatrzymanie CPU. Przed taką sytuacją chronią specjalne bloki systemowe, np. w oprogramowaniu firmy Siemens jest to blok ACK_OP realizujący potwierdzanie błędów z zastosowaniem HMI. Ten blok systemowy jest wyjątkiem od zalecanej realizacji wymiany danych [15].



Rys. 16. Przykład użycia bloku RS do sterowania procesem (przykład Logo Soft Comfort)
Fig. 16. Example of using the RS block for process control (example Logo Soft Comfort)

Przykład z rys. 17 przedstawia możliwość załączenia silnika (SKRZYNKA_Wy_ZAL_WYL_SILNIK) za pomocą przycisku start (przycisk normalnie otwarty) umieszczonego na szafie sterowniczej (SKRZYNKA_0We_START_SZAFA) lub za pomocą przycisku na panelu HMI. Sterowanie elementami bezpieczeństwa za pomocą panelu HMI bez zastosowania specjalnych bloków systemowych jest naruszeniem bezpieczeństwa. Dla dużych (pod względem zajmowanej powierzchni) zespołów maszyn do sterowania procesem technologicznym często stosuje się kilka pulpitów sterowniczych. Dyrektywa maszynowa w pkt. 1.2.2 załącznika 1 wymaga, by „w przypadku, gdy maszyna posiada dwa lub więcej stanowiska operatora, każde stanowisko musi być tak wyposażone we wszystkie wymagane elementy sterownicze, aby operatorzy nie przeszkadzali sobie lub nie stwarzali wzajemnie sytuacji zagrożenia". Pod kątem programowania ważne jest odpowiednie zabezpieczenie programowe, tak by nie można było sterować procesem technologicznym z kilku pulpitów sterowniczych jednocześnie w sposób mogący powodować zagrożenie. Norma PN-EN ISO 13850:2016-03 wprowadziła możliwość podziału na strefy bezpieczeństwa maszyn zatrzymanych awaryjnie, co oznacza, że dla każdego z pulpitów sterowniczych różne elementy układu sterowania (strefy bezpieczeństwa) mogą być wyłączane przyciskiem bezpieczeństwa. Oznaczenie, która strefa bezpieczeństwa jest przypisana do danego przycisku musi być jednoznaczne. Za każdym razem dany przycisk bezpieczeństwa musi działać tak samo, co oznacza, że nie można jego działania uzależnić od np. zadziałania jakiegokolwiek czujnika.

Rys. 17. Przykład zastosowania panelu HMI do sterowania procesem [GX Works 2]
Fig. 17. Example of using the HMI panel for process control [GX Works 2]

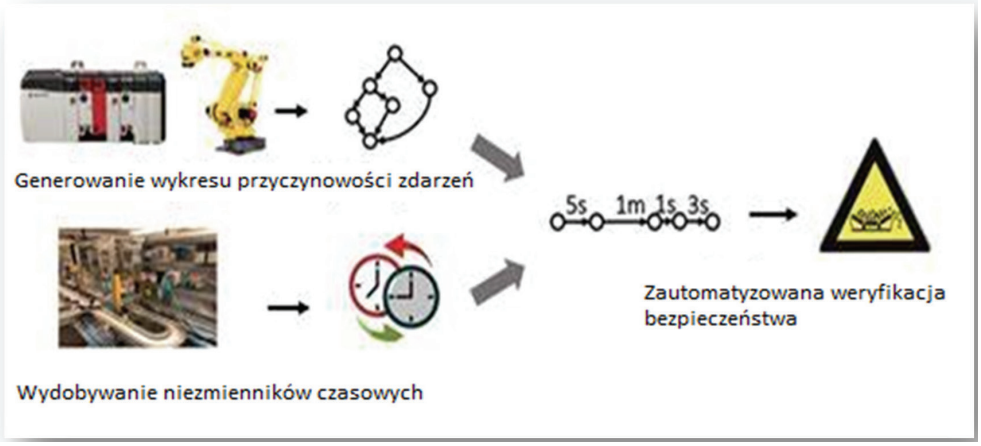
4. Automatyczna walidacja systemu sterowania bezpieczeństwem

Ze względu na złożoność współczesnych systemów sterowania maszyn zespolonych istnieje coraz większa potrzeba rozwijania zautomatyzowanych mechanizmów do wykrywania problemów z bezpieczeństwem. Podczas projektowania zespołów maszyn oraz walidacji systemu sterowania należy uwzględnić tzw. możliwe do przewidzenia niewłaściwe użycie maszyn (Dyrektywa Maszynowa – pkt 1.1.2 c) podczas projektowania i wykonywania maszyny oraz podczas opracowywania instrukcji producent lub jego upoważniony przedstawiciel musi wziąć pod uwagę nie tylko zamierzone zastosowanie maszyny, ale także możliwe do przewidzenia niewłaściwe użycie). Na świecie pojawiają się prace naukowe [1], w których opisywane są próby zastosowania algorytmów automatyzujących proces walidacji systemu sterowania (rys. 19), umożliwiających automatycznie znajdowanie miejsc programu, w których znajdują się potencjalne problemy z bezpieczeństwem. Automatyzacja procesu walidacji umożliwia udokumentowanie przebiegu procesu walidacji, w tym rozwiązanie przynajmniej części problemów wynikających z obowiązku udokumentowania i opisania przypadków możliwego do przewidzenia niewłaściwego użycia maszyny.

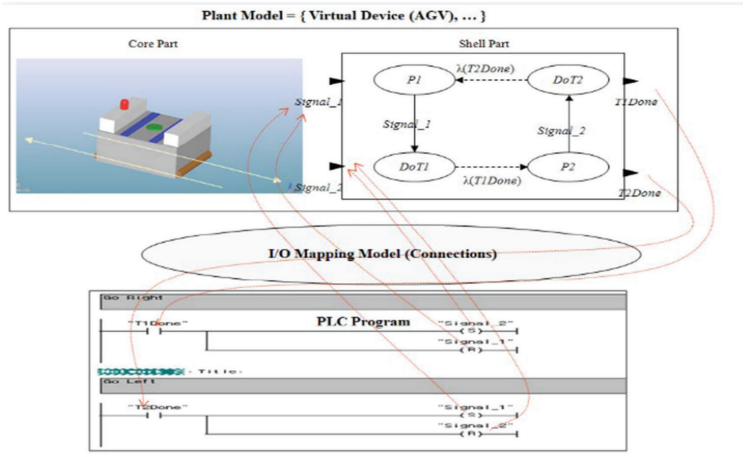
Aplikacja VetPLC [1] może skutecznie generować sekwencje zdarzeń czasowych, które umożliwiają wykrycie błędów bezpieczeństwa zawartych w kodzie PLC ze względu na błędy programistów. Rzeczywisty kod programu PLC sterowany zdarzeniami jest wrażliwy na kolejność i czas ich wystąpienia. VetPLC automatycznie generuje czasowe sekwencje zdarzeń, oraz przez analizę przyczyn zdarzeń w kodzie PLC i wydobywanie danych o czasie zdarzenia z fizycznego stanowiska wskazuje obszary, w których mogą znajdować się ukryte naruszenia bezpieczeństwa (rys. 18). Na tej podstawie możliwe jest udokumentowanie i zautomatyzowanie procesu walidacji dużych maszyn zespolonych.

W pracach [17, 18] zaproponowano architekturę środowiska programistycznego PLC umożliwiającą wizualną weryfikację programów PLC. Proponowana architektura integruje program PLC z odpowiednim modelem zakładu, dzięki czemu użytkownicy mogą intuicyjnie weryfikować program PLC w środowisku graficznym 3D. Model zakładu obejmuje wszystkie urządzenia produkcyjne, a program PLC zawiera logikę sterowania modelem zakładu. Artykuł opisuje metodę konstrukcji modelu zakładu opartą na formalizmie DEVS (ang. *Discrete Event Systems Specifications*), który obsługuje dyskretne specyfikacje modeli zdarzeń w sposób hierarchiczny oraz modułowy (rys. 19).

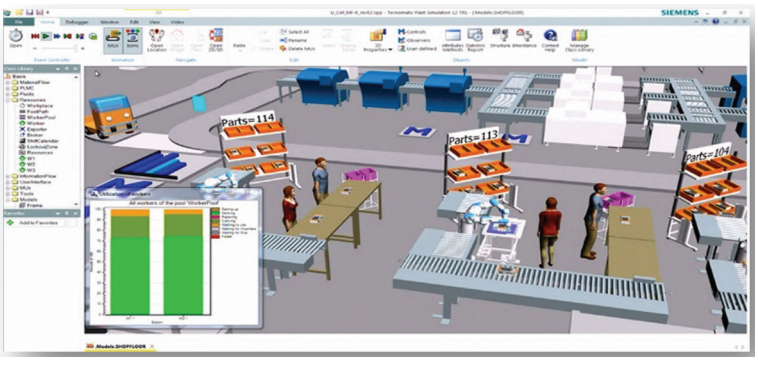
Wizualna weryfikacja programów PLC możliwa jest również przez pracę na modelach „cyfrowych bliźniaków" (ang. *digital twins*) (rys. 20) [19]. Są to cyfrowe repliki fizycznych obiektów, procesów i systemów. Model cyfrowego bliźniaka to połączenie fizycznego obiektu oraz jego cyfrowego odwzorowania w przestrzeni wirtualnej [19].



Rys. 18. Przykład idei automatyzacji procesu walidacji z zastosowaniem programu VetPLC [27]
Fig. 18. An example of the idea of automating the validation process using the VetPLC program [27]



Rys. 19. Mapowanie połączeń między cyfrowym modelem zakładu a programem PLC [18]
Fig. 19. Mapping connections between the digital model of the plant and the PLC program [18]



Rys. 20. Przykład „cyfrowego bliźniaka" w oprogramowaniu Tecnomatix Plant Simulation firmy Siemens [20]
Fig. 20. An example of a "digital twin" in the Tecnomatix Plant Simulation software by Siemens [20]

Połączenie modeli „cyfrowych bliźniaków" [19] maszyn, wraz z zastosowaniem programu generującego sekwencje zdarzeń do testowania systemów bezpieczeństwa [1], przy równoczesnym zastosowaniu wirtualnej rzeczywistości, umożliwi maksymalizację bezpieczeństwa funkcjonalnego przez poszukiwanie ukrytych naruszeń bezpieczeństwa.

5. Podsumowanie

Podstawy bezpieczeństwa przy tworzeniu i obsłudze maszyn zintegrowanych powinny być dobrze zdefiniowane i rozpoznane, aby uniknąć awarii i wypadków, w tym ich konsekwencji dla człowieka i środowiska. Przez prawidłową walidację systemów bezpieczeństwa wspomaganą oprogramowaniem do poszukiwania ukrytych naruszeń bezpieczeństwa możliwa jest poprawa bezpieczeństwa funkcjonalnego związanego z oprogramowaniem sterującym. Automatyzację mechanizmów do wykrywania problemów z bezpieczeństwem utrudniają liczne języki programowania sterowników bezpieczeństwa oraz niekompatybilność oprogramowania bezpieczeństwa różnych producentów. Podane przykłady niepoprawnych praktyk aplikacji wybranych aspektów funkcji bezpieczeństwa umożliwiają lepsze zrozumienie tematyki ukrytych naruszeń bezpieczeństwa. Przedstawione rozwiązania zautomatyzowanych mechanizmów do wykrywania problemów z bezpieczeństwem wskazują kierunek rozwoju związanego z walidacją zespołów maszyn. Przez zastosowanie programów generujących sekwencje zdarzeń do testowania systemów sterowania i zastosowanie wirtualnej rzeczywistości do wizualnej weryfikacji programu PLC można zapewnić bezpieczniejsze miejsce pracy operatora maszyny zintegrowanej.

Bibliografia

1. Zhang M., Chen C.Y., Kao B.C., 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*, 2019 IEEE Symposium on Security and Privacy (SP), DOI: 10.1109/SP.2019.00034.
2. Broel-Plater B., *Układy wykorzystujące sterowniki PLC, Projektowanie algorytmów sterowania*, Wydawnictwo Naukowe PWN, Warszawa 2021.
3. Rockwell Automation, Systemy sterowania związane z bezpieczeństwem maszyn. Zasady, normy i wdrożenie (Podręcznik bezpieczeństwa – wersja 5)
4. https://iautomatyka.pl/kurtyny-bezpieczenstwa-rodzaje-dzialanie-sposoby-stosowania-sick/
5. https://www.magazynprzemyslowy.pl/artykuly/walidacja-bezpieczenstwa-maszyn
6. https://bezpieczenstwo-maszyn.com/uslugi/walidacja/
7. http://ftp.beckhoff.pl/download/document/automation/twinsafe/TwinSAFE-Logic-FBen.pdf
8. http://www.contra-polska.pl/pliki/1000/636_pl_pluto.pdf
9. http://ftp.beckhoff.pl/download/document/automation/twinsafe/TcSafetyPLC_en.pdf
10. https://fabrykaelektryka.pl/pulpit-sterowania-dwureczznego-2-przyciski-czerwony-stop-2r-okapturzony-xy2sb72-schneider-electric-p-80944.html
11. Wieland, Samos PLAN6 Software Manual nr. BA000968
12. https://bezpieczenstwowsystemachsterowania.pl/2018/04/warunki-funkcjonowania-oprogramowania-w-aplikacjach-bezpieczenstwa/
13. http://ftp.beckhoff.pl/download/document/automation/twinsafe/applicationguidetwinsafeen.pdf
14. Siemens, *Przekaźniki bezpieczeństwa – podręcznik aplikacyjny*, edycja 2014
15. Siemens, *Simatic safety – konfiguracja i programowanie* – Podręcznik programowania i obsługi, 10/2019
16. Siemens, Safety Programming Guideline for SIMATIC S7-1200/1500, 10/2017
17. Ko M., Park S., Wang G.N., *Visual Validation of PLC Programs*, 2008, DOI: 10.7148/2008-0410.
18. Park S.C., Park C.M., Wang G.N., Kwak J., Yeo S., *PLC-Studio: Simulation based PLC code verification*, 2008, DOI: 10.1109/WSC.2008.4736071.
19. https://przemyslprzyszlosci.gov.pl/tag/digital-twin/
20. https://przemysl-40.pl/index.php/2017/12/04/cyfrowy-blizniak/
21. http://www.nirtec.com/index.php/plc-virtual-simulator/
22. TwinCAT Safety PLC PC based Safety Controller
23. Delta, "DVP-ES2/EX2/SS2/SA2/SX2/SE&TP Operation Manual – Programming", 2014
24. https://induprogress.pl/produkty/sterowanie-i-wizualizacja-procesow/sterowniki-plc-serii-dvp-slim/
25. https://www.wieland-electric.com/en/products/safety-technology/safety-controller/
26. https://mall.industry.siemens.com/mall/pl/pl/Catalog/Product/6ES7212-1AF40-0XB0
27. https://pdfs.semanticscholar.org/dc67/3d-fe396e171f70e37ef5ec4d989dbe4b15de.pdf?_ga=2.216117650.543777785.1619125049-890432265.1613689756
28. https://download.beckhoff.com/download/document/automation/twinsafe/tcsafetyplc_en.pdf
29. PN-EN ISO 13851:2019-05 - Oburęczne urządzenia sterujące, aspekty funkcjonalne, zasady projektowania
30. Dyrektywa 2006/42/WE Parlamentu Europejskiego i Rady: https://eur-lex.europa.eu/legal-content/PL/TXT/PDF/?uri=CELEX:32006L0042&from=PL

Hidden Security Breaches in Automatic Control of Technological Processes

Abstract: The progressing automation and robotization in the industrial plants as well as the increasing complexity of the control systems of integrated machines make it necessary to constantly improve the functional safety of machines through the correct validation of safety systems. Despite the validation process carried out, the potential software errors may reveal during the usage of the machine as hidden security breaches. The article presents examples of security breaches of real machine tools and attempts to implement solutions of automated mechanisms for detecting security problems. Another aspect of the article is the new approach for detecting hidden security breaches. Using the „digital twin" model of the machine, a program that generates a sequence of events for testing control systems, and the use of a virtual reality (visual verification of the safety programs), it is possible to maximize the functional safety functions of the machine.

Keywords: hidden security breaches, safety system validation, programmable safety controllers, functional safety of integrated machines, automatic validation

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.