Adam MILIK
POLITECHNIKA ŚLĄSKA, INSTYTUT ELEKTRONIKI
ul. Akademicka 16, 44-100 Gliwice

Metoda reprezentacji pośredniej programu PLC opisanego za pomocą języków LD i SFC na potrzeby syntezy sprzętowej

Dr inż. Adam MILIK

Ukończył studia na Wydziale Automatyki, Elektroniki i Informatyki Politechniki Śląskiej. Pracę doktorską obronił w 2003 r. Jest adiunktem w Instytucie Elektroniki Politechniki Śląskiej. Jego zainteresowania naukowe obejmują synteza wysokiego poziomu złożonych układów sprzętowo-programowych implementowanych w układy logiki programowalnej, sterowniki programowalne, techniki kosymulacji systemów cyfrowych i środowisk modelowania algorytmicznego.

e-mail: adam.milik@polsl.pl

Streszczenie

W artykule przedstawiono metody reprezentacji pośredniej programu sterowania opisanego językiem LD oraz SFC zgodnie z IEC61131-3, opracowane na potrzeby syntezy sprzętowej układów sterowania PLC implementowanych w strukturach programowalnych FPGA. W opisie wykorzystano oryginalną implementację grafu skierowanego. Przedstawiono opracowane reguły odwzorowania, zapewniające zachowanie zależności sekwencyjnych przy jednoczesnym uzyskaniu maksymalnego zrównoleglenia działania. Przedstawiono również zarys metod syntezy na podstawie opracowanego odwzorowania pośredniego.

Słowa kluczowe: sterownik programowalny, diagram stykowy, LD, sekwencyjny schemat funkcji, SFC, synteza logiczna wysokiego poziomu, graf przepływu danych, DFG, FPGA, układy rekonfigurowane.

A common intermediate representation of LD and SFC programs for hardware synthesis purposes

Abstract

The increased performance of a PLC can be achieved by direct implementation of a control program in an FPGA device [3, 6, 7, 8, 12, 13]. The paper presents a methodology of transforming a standard PLC program given by LD or SFC according to IEC61131-3 to the common intermediate form dedicated for logic synthesis. The intermediate form of the control program is represented by a data flow graph (DFG, Fig. 1). The set of nodes is carefully selected to minimize the number of different types of nodes while assuring implementation of PLC behavior. Attributed edges and multiple argument nodes are used to reduce size of DFG (Fig. 2). The developed method for creating a DAG maintains sequential dependencies between variables and revel operations parallelism. In PLC programs the variables pass values between operations and computation cycles. In order to maintain sequential dependencies, value assignment to a variable is observed. If the accessed variable has not been assigned, its value is used for a driving node (Fig. 3). The SFC is based on step, actions and transitions [2]. The step variable in the DFG is represented by a JK flip-flop equivalent. The activation function of a step is based on analysis of its dependencies with preceding and succeeding steps and transitions (Fig. 5). Actions that are bounded with steps are controlled according to their types (Fig. 6). The presented intermediated representation has been successfully applied to synthesize a PLC implemented in an FPGA device.

Keywords: PLC, FPGA, high level logic synthesis, LD, SFC, DFG, data flow graph, ladder diagram.

1. Wstęp

Powszechnie dostępne sterowniki programowalne PLC są konstruowane za pomocą układów mikroprocesorowych realizujących program sterowania na drodze szeregowo-cyklicznej. W celu przyśpieszenia realizacji programu, prowadzone są badania nad architekturami jednostek centralnych złożonych z odrębnych procesorów boolowskiego i słowowego [1]. Wykorzystuje się w nich różne techniki przyśpieszenia realizacji fragmentów programu poprzez autonomiczną pracę jednostek, czy też wybieranie fragmentów programu do realizacji na podstawie zachodzących zdarzeń zewnętrznych.

Istotnym elementem w projektowaniu układów sterowania jest ich łatwość programowania. Wprowadzono normę IEC61131 w celu ujednolicenia konstrukcji systemów sterowania oraz metod ich programowania.

Ciągły rozwój układów programowalnych FPGA a szczególnie znaczący wzrost pojemności logicznej, skłania do ich wykorzystania w sterownikach programowalnych w miejsce implementacji szeregowo-cyklicznej opartej na mikroprogramowaniu. Programowalny układ logiczny jest wykorzystywany do budowy algorytmu w sposób bezpośredni. Bezpośrednia implementacja sprzętowa algorytmów sterowania pozwala osiągnąć istotne zwiększenie wydajności obliczeniowej poprzez równoległą realizację wielu zadań sterowania. Kluczowym elementem do przeprowadzenia efektywnej implementacji układu sterowania jest opracowanie zestawu narzędzi syntezy logicznej wysokiego poziomu, pozwalającej na przekształcenie programu sterowania do postaci podlegającej odwzorowaniu technologicznemu w wybranym układzie programowalnym.

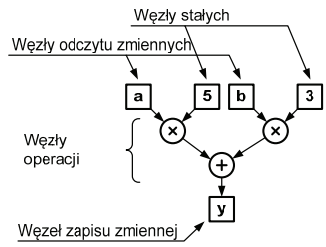
Wiele zespołów badawczych zaobserwowało ogromny potencjał obliczeniowy układów programowalnych FPGA w zakresie implementacji układów sterowania PLC [3, 4, 6, 8, 12, 13]. Istotną trudnością napotykaną w bezpośredniej implementacji programu sterowania jest zbudowanie narzędzi umożliwiających dokonanie syntezy układu sterowania. Ciekawym rozwiązaniem jest zaproponowanie dedykowanej architektury układu programowalnego, przeznaczonej do bezpośredniego odwzorowania operacji logicznych diagramu stykowego [13]. Można zauważyć propozycje dość złożonych metod odwzorowania charakteryzujących się tworzeniem znacznej liczby cykli pośrednich [3, 12]. Innym przykładem podejścia, opisanym w [4] jest zamiana programu sterowania na język C lub SystemC a następnie wykorzystanie narzędzi syntezy wysokiego poziomu (Catapult C) w celu otrzymania syntezowalnego opisu strukturalnego.

Podsumowując, na podstawie dokonanego przeglądu, rysuje się istotna potrzeba opracowania narzędzia syntezy i odwzorowania sprzętowego programu sterowania. Istotnym elementem każdego narzędzia syntezy jest opracowanie uniwersalnej reprezentacji pośredniej, uzyskanej w wyniku rozbioru semantycznego zdań języka. Postać pośrednia powinna charakteryzować się łatwością operowania, umożliwiającą przekształcanie oraz optymalizację włączając w to proces syntezy i odwzorowania technologicznego.

2. Reprezentacja pośrednia programu PLC

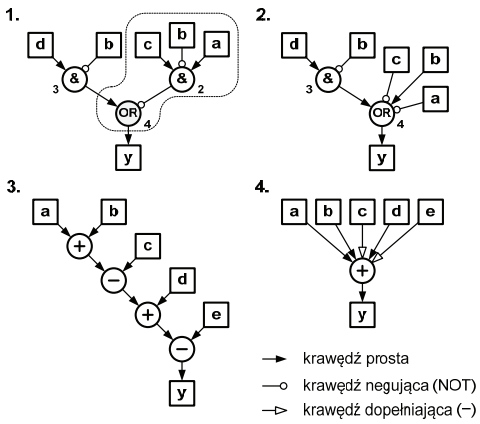
Do reprezentacji pośredniej programu sterowania wykorzystano graf przepływu danych (dalej zwany DFG). Jest on powszechnie wykorzystywany w procesie rozbioru wyrażeń oraz syntezy logicznej wysokiego poziomu [5, 11]. DFG opisany jest dwójką $G = \langle V, E \rangle$, która składa się ze zbioru wierzchołów V oraz zbioru krawędzi skierowanych E. Elementami zbioru E są uporządkowane pary $e = (a, b)$ gdzie $a, b \in V$. Przytoczona definicja grafu skierowanego nie obejmuje jednak pewnych istotnych aspektów jakie powinien on posiadać na potrzeby reprezentacji pośredniej programu sterowania. Podstawowy graf przepływu informacji pokazano na rysunku (rys. 1). Graf składa się z węzłów reprezentujących odczyt i zapis wartości zmiennych, wartości stałych oraz węzłów operacji. W celu efektywniejszego operowania grafem zastosowano dla operacji przemiennych węzły wieloargumentowe.

Umożliwia to redukcję liczby węzłów reprezentacji grafowej oraz ułatwia redukcję argumentów poprzez wykorzystanie podstawowych przekształceń logicznych i arytmetycznych w obrębie pojedynczego węzła. Wykorzystanie węzłów wieloargumentowych jest również wygodne z punktu widzenia syntezy logicznej. Węzły takie poddaje się ekspansji do postaci nadającej się do bezpośredniego odwzorowania technologicznego. Podczas procesu ekspansji kontroluje się dystrybucję operacji tak aby dążyć do równomiernego rozkładu czasu obliczeń oraz ich zrównoleglenia (redukcja wysokości drzewa).



Rys. 1. Graf przepływu informacji – postać ogólna
Fig. 1. The Data Flow Graph (DFG) – general representation

Do opisu układów sterowania wykorzystano węzły reprezentujące podstawowe operacje logiczne tj. AND, OR oraz węzły operacji arytmetycznych: sumy, iloczynu i ilorazu. Dodatkowo wykorzystano węzeł wyboru warunkowego, umożliwiający powiązanie warunków logicznych z warunkowym wyborem wyniku wypracowanym przez wybrane poddrzewo grafu. Argumentami węzła wyboru mogą być zarówno operacje arytmetyczne jak i logiczne. W grafie zrezygnowano z węzłów operacji jednoargumentowych inwersji logicznej oraz dopełnienia arytmetycznego. Operacje te realizowane są poprzez nadanie krawędziom grafu atrybutów odpowiednio dopełnienia algebraicznego lub inwersji logicznej. Na rysunku 2 pokazano zastosowanie krawędzi z atrybutem. Rysunki 2.1 i 2.2 przedstawiają przekształcenie operacji logicznej z wykorzystaniem praw de Morgana. Rysunek 2.3 pokazuje ogólną implementację wyrażenia $y = a+b-c+d-e$. Stosując opisaną metodę krawędzi z atrybutem dopełnienia arytmetycznego oraz koncepcję węzła wieloargumentowego, operację można sprowadzić do postaci pokazanej na rysunku 2.4.



Rys. 2. Reprezentacja operacji logicznych i arytmetycznych z wykorzystaniem krawędzi z atrybutem
Fig. 2. The DAG representation of logic and arithmetic functions with use of attributed edges

3. Metody konstruowania DAG z LD

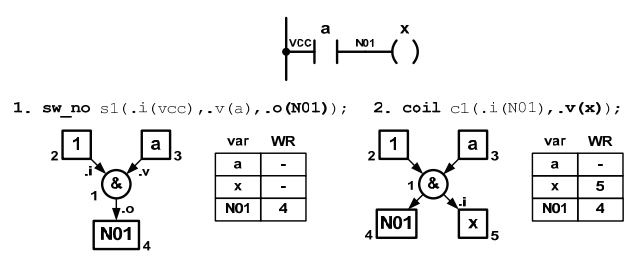
Schemat stykowy przedstawia układ sterowania w postaci sieci połączonych elementów symbolizujących przepływ energii pomiędzy liniami zasilania. Analiza poszczególnych sieci odpowiada pojedynczemu obiegowi pętli programu sterowania. Ze względu na szeregowo-cykliczny sposób analizy przebiegający w sposób wierszowy, kolejność umieszczenia elementów na schemacie wpływa na sposób jego interpretacji przez sterownik PLC.

Podstawowymi elementami tworzącymi relacje logiczne są styki, cewki i węzły. Styki reprezentują odczyt wartości zmiennej natomiast cewki reprezentują zapis wartości zmiennych. W zbiorze zmiennych układu sterowania rozróżnia się trzy podzbiory, na podstawie skojarzenia zmiennych z sygnałami wejściowymi, wyjściowymi oraz wewnętrznymi układu sterowania. Zmienne skojarzone z sygnałami wejściowym mogą podlegać jedynie odczytowi. Zmienne skojarzone z sygnałami wyjściowymi i wewnętrznymi podlegają zarówno odczytowi jak i zapisowi. Odpowiadają one za przekazywanie wartości pomiędzy kolejnymi cyklami procesu sterowania. Kolejność umieszczenia elementów na schemacie stykowym decyduje o pobraniu wartości należącej do cyklu poprzedniego lub wypracowanego w bieżącym obiegu pętli. W przypadku układów sprzętowych, bardziej adekwatnym pojęciem będzie cykl obliczeniowy.

Zaproponowany model przetwarzania składa się z pojedynczego cyklu, w którym na podstawie bieżących wartości zmiennych zostaną wyznaczone ich nowe wartości. Istotą jest uchwycenie zależności zachodzących pomiędzy wartościami zmiennych. Prostota konstrukcji oraz minimalna liczba cykli stanowią istotną zaletę w stosunku do modelu translacji zaproponowanego przez [3, 6, 8, 12].

Kluczowym elementem wykorzystywanym przy budowie DFG na podstawie LD jest wzajemna relacja odczytu i zapisu zmiennych. Kolejność odczytu i zapisu determinuje budowę niejawnego automatu sekwencyjnego wykorzystującego blok pamięci, zbudowany ze zmiennych skojarzonych z sygnałami wewnętrznymi lub wyjściowymi. Sformułowano następujące reguły konstrukcji DFG: 1. Jeżeli zmienna nie posiada nadanej wartości w bieżącym cyklu, należy pobrać wartość zmiennej z poprzedniego cyklu poprzez utworzenie węzła odczytu wartości zmiennej. 2. W przypadku gdy zmienna posiada przypisaną wartość w miejsce odczytu wartości zmiennej wprowadza się referencje do węzła nadającego wartość zmiennej. 3. Nadanie wartości zmiennej tworzy węzeł zapisu oraz łączy go z węzłem sterującym. 4. Jeżeli do zmiennej posiadającej niepustą referencję zapisu została przypisana nowa wartość, zostaje zaktualizowana referencja węzła źródłowego w węźle zapisu wartości.

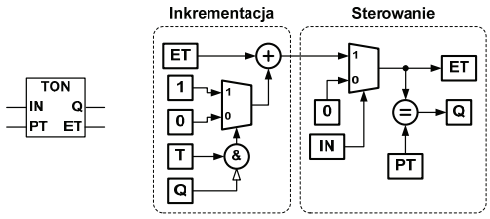
Sposób analizy diagramu stykowego zilustrowano przykładem pokazanym na rysunku (rys. 3). W pierwszej kolejności dokonuje się analizy styku, który wypracowuje wartość zmiennej pomocniczej N01. Następnie analizowana jest cewka x sterowana z N01. Zmienna N01 posiada przypisaną wartość pochodzącą z wyjścia styku a. Do zmiennej *x* zostaje przypisany węzeł sterujący wartością zmiennej N01.



Rys. 3. Sposób konstruowania DFG na podstawie analizy diagramu stykowego
Fig. 3. Method of building DFG based on analysis of LD

Obok operacji elementarnych, w schemacie stykowym występują złożone bloki funkcjonalne. Zostają one zastąpione odpowiadającymi im drzewami DFG włączanymi do struktury projektu. Na rysunku (rys. 4) pokazano implementację DFG bloku czasomierza TON. Jest on przykładem integracji operacji logicznych

i arytmetycznych. Na podobnych zasadach tworzone są inne bloki złożone czasomierzy , liczników i regulatorów.

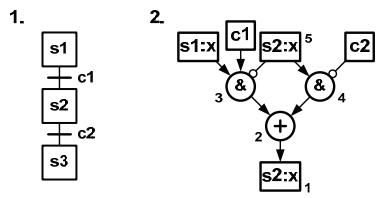


Rys. 4. Przykład DFG modułu złożonego DFG wykorzystywane do podstawienia czasomierza TON
Fig. 4. An example of complex unit DFG used for substituting TON timer

4. Metody konstruowania DAG z SFC

Sekwencyjny schemat funkcjonalny (dalej zwany w skrócie SFC) jest przeznaczony do reprezentacji graficznej sekwencyjnych procesów sterowania. Umożliwia on opisywanie zależności o charakterze równoległym poprzez graficzne przedstawienie zasad przekazywania sterowana [2]. W literaturze można znaleźć metodę przekształcenia schematu do postaci automatu sekwencyjnego [9]. Występowanie zadań równoległych podnosi złożoność uzyskanego opisu automatu. Na potrzeby syntezy sprzętowej układów sterowania, zaproponowano wykorzystanie niejawnego automatu sekwencyjnego zbudowanego na podstawie analizy relacji przejść i kroków.

Podobnie jak w normie IEC61131, związano z każdym krokiem zmienną Boolowską. W ten sposób każdy krok posiada indywidualny znacznik, który może być ustawiany i kasowany niezależnie od pozostałych znaczników. Przejścia, które łączą kroki można opisać za pomocą uporządkowanej trójki: $t = \langle S_P, S_S, c \rangle$ gdzie S_P to zbiór kroków poprzedzających przejście, S_S zbiór kroków następujących po przejściu, c wyrażenie logiczne określające warunek dokonania przejścia.



Rys. 5. Fragment SFC (1) oraz implementacja DFG aktywności kroku s2 (2)
Fig. 5. An exemplary SFC (1) and respective DFG of step s2 activity (2)

Uaktywnienie kroku odbywa się poprzez wyznaczenie sumy logicznej warunków wykonania przejść prowadzących do danego kroku, przy założeniu, że każde przejście łączy dokładnie jeden krok poprzedzający z krokiem następującym.

$$s.x_{SET} = \sum_{t \in T} sp.x \cdot c \quad \text{dla } t : s \in S_S \tag{1}$$

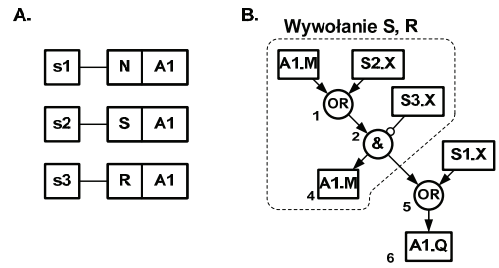
Jeżeli moc zbioru kroków poprzedzających S_P danego przejścia jest większa od 1, to wszystkie kroki należące do S_P muszą być aktywne aby przejście mogło być wykonane. W ogólnym przypadku wyznacza się iloczyn logiczny zmiennych aktywności należących do zbioru kroków poprzedzających analizowanego przejścia:

$$s.x_{SET} = \sum_{t \in T} \left(\prod_{sp \in Sp} sp.x \right) \cdot c \quad \text{dla } t : s \in S_S \tag{2}$$

Znacznik kroku zostaje skasowany gdy możliwe jest wykonanie przejścia, w którym dany krok znajduje się w zbiorze kroków poprzedzających. Należy zwrócić uwagę, że w przypadku zbioru kroków poprzedzających przejście o mocy większej od 1, wszystkie z nich muszą być aktywne. Prowadzi to do ogólnego wzoru wyznaczającego utratę aktywności kroku

$$s.x_{CLR} = \sum_{t} \left(\prod_{sp \in Sp} sp.x \right) \cdot c \quad \text{dla } t : s \in S_P \tag{3}$$

Przedstawione wymagania nasuwają skojarzenie zmiennej aktywności kroku z przerzutnikiem *sr* lub *rs*. W konstrukcji DFG zmiennej aktywności kroku na podstawie SFC wykorzystano funkcję wzbudzeń przerzutnika JK. Na rysunku (rys. 5) pokazano przykładowy SFC oraz uzyskany zgodnie z opisanymi regułami DFG. Z każdym krokiem SFC można związać akcje. Aktywność akcji jest zależna od aktywności kroku, z którym jest związana oraz kwalifikatora jej przetwarzania. Na rysunku pokazano implementację DFG akcji wyzwalanej z kwalifikatorami N, S oraz R (kasowanie akcji).



Rys. 6. Przykładowe sposoby wywołania akcji (A) oraz ich reprezentacja DFG (B)
Fig. 6. An exemplary action activations (A) and respective DFG implementation (B)

5. Podsumowanie

W artykule przedstawiono fragment opracowanego narzędzia syntezy, przeznaczonego do tworzenia reprezentacji pośredniej programów sterowania opisanych diagramem stykowym oraz sekwencyjnym schematem funkcjonalnym. Przedstawione metody, odtwarzają relacje współbieżne oraz zachowują relacje sekwencyjne operacji na podstawie analizowanego programu LD i SFC. Znaczącą zaletą metody jest jednocyklowy sposób realizacji obliczeń. Na podstawie uzyskanego DFG jest prowadzona synteza układu sterowania. Ze względu na implementację w ograniczonych zasobach logicznych tworzony jest harmonogram operacji a następnie dokonuje się ich rozmieszczenia. Prace nad pakietem syntezy układów sterowania są prowadzone w dalszym ciągu. Planuje się podjęcie próby wykorzystanie opisanych metod również do generacji optymalizowanego strumienia instrukcji dla wieloprocesorowych jednostek centralnych układów sterowania.

6. Literatura

[1] Chmiel M., Hrynkiewicz E.: Concurrent operation of processors in the bit-byte CPU of a PLC. Control Cybernetics. 2010 vol. 39 nr 2, pp. 559-579.
[2] David R.: Grafcet: A Powerful Tool for Sepcification of Logic Controllers. IEEE Transactions on Control Systems Technology, vol. 3 no. 3, 1995, pp 253-268.
[3] D. Du, X. Xu, and K. Yamazaki: A study on the generation of silicon based hardware PLC by means of the direct conversion of the ladder diagram to circuit design language. Springer London, 2010, vol. 49.
[4] Economakos C.; Economakos G.: C-based PLC to FPGA translation and implementation: The effects of coding styles, 16th International

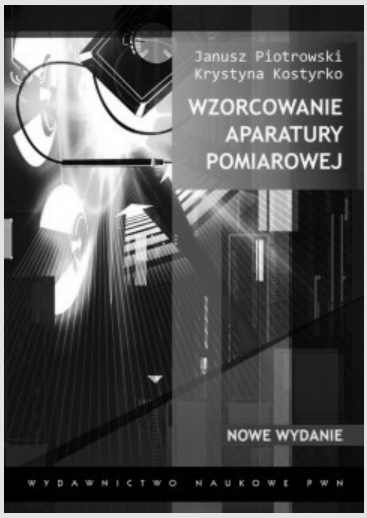
Conference on System Theory, Control and Computing (ICSTCC), 2012 , pp.1-6, 12-14 Oct. 2012.
[5] D. Gajski, N. Dutt, A. We, S. Lin: High-Level Synthesis Introduction to Chip and System Design, Kluwer Academic Publishers, 1994.
[6] Ichikawa S., Akinaka M., Kieda R., Yamamoto H.: Converting PLC instruction sequence into logic circuit: A preliminary study, IEEE International Symposium on Industrial Electronics, July 2006, vol. 4, pp.2930-2935.
[7] Milik A., Hrynkiewicz E.: Synthesis and implementation of reconfigurable PLC. International Journal of Electronics and Telecommunications, vol. 58, nr 1, March 2012, pp. 85-94.
[8] Mocha J., Kania D.: Metoda sprzętowej realizacji programu LD z wykorzystaniem układów FPGA, Pomiary Autom. Kontrola 2012 vol. 58 nr 1, s. 88-92.
[9] Philippot A., Tajer A.: From GRAFCET to Equivalent Graph for synthesis control of discrete events systems, 18th Mediterranean Conference on Control & Automation (MED), 2010 pp.683-688.
[10] Subbaraman S., Patil, M. M., Nilkund, P. S.: Novel integrated development environment for implementing PLC on FPGA by converting ladder diagram to synthesizable VHDL code, 11th International Conference on Control Automation Robotics & Vision (ICARCV), pp.1791-1795, 7-10 Dec. 2010.
[11] Wirth N.: Algorytmy + Struktury Danych = Programy" WNT, Warszawa 1989.
[12] Yadong L., Kazuo Y., Makoto F., Masahiko M.: Model-driven programmable logic controller design and FPGA-based hardware implementation, ASME International Design Engineering Technical Conferences and Computers and Information in Engineering Conference-DETC2005, 2005, pp. 81-88.
[13] Welch, J.T.; Carletta, J.: A direct mapping FPGA architecture for industrial process control applications, International Conference on Computer Design, 2000. pp.595-598, 2000.

otrzymano / received: 20.05.2013
przyjęto do druku / accepted: 03.07.2013 *artykuł recenzowany / revised paper*

RECENZJE

Wzorcowanie aparatury pomiarowej

Janusz Piotrowski, Krystyna Kostyrko
Wydawnictwo naukowe PWN, Warszawa, 2012, str. 582, ISBN 978-83-01-17051-6



Pozycja jest adresowana przede wszystkim do inżynierów i specjalistów zajmujących się konstruowaniem i stosowaniem przyrządów i systemów pomiarowych we wszystkich dziedzinach pomiarów – od mechaniki, elektrotechniki po chemię fizyczną i analityczną. Słusznie, dość obszernie potraktowano badania metodami analitycznymi, gdyż właśnie w pomiarach fizykochemicznych występuje największa różnorodność metod wzorcowania.

Książka powinna być zalecana pracownikom instytucji metrologicznych, w tym pracownikom akredytowanych i dopiero wdrażających system jakości laboratoriów badawczych i wzorcujących. Podano w niej m.in. cele i zasady akredytacji, wymagania obowiązujące w akredytowanej jednostce, sposoby kontroli spełnienia tych wymagań oraz niezbędną dokumentację. Warto podkreślić, że „Wzorcownie aparatury pomiarowej" to jedyna pozycja, w której udało mi się znaleźć – wartościowy dla personelu laboratorium starającego się o akredytację – przykład Księgi Jakości spełniającej wymagania normy PN-EN ISO/IEC 17025. Ponadto w pozycji omówiono sposób zarządzania komputerową obsługą laboratorium podporządkowany wymaganiom ww. normy.

Książka Janusza Piotrowskiego i Krystyny Kostyrko stanowi wartościowy materiał również dla studentów i pracowników naukowych związanych z szeroko pojętą metrologią. W książce wskazano bowiem najnowsze dokumenty istotne dla rozwoju metrologii opracowane z inicjatywy Międzynarodowego Biura Miar BIPM. Zamieszczono również informacje o państwowych wzorcach podstawowych jednostek miar i o postępie prac prowadzonych obecnie przez wiodące Krajowe Instytuty Metrologiczne (ang. NMI) zmierzających do ustanowienia kwantowego układu jednostek miar. Omówiono tutaj zatwierdzone na Generalnej Konwencji Miar w roku 2011 teoretyczne założenia kwantowego układu jednostek miar SI, opartego na siedmiu podstawowych stałych fizycznych.

Podsumowując: książka Janusza Piotrowskiego i Krystyny Kostyrko „Wzorcowanie aparatury pomiarowej" to pozycja, w której obszernie omówiono wymagania kalibracyjne będące dzisiaj standardem nowoczesnej metrologii. Stanowi wartościowy materiał dla szerokiego grona metrologów praktyków opracowujących metody i wykonujących pomiary na różnym poziomie dokładności.

Krzysztof MUSIOŁ