# Emilian ŚWITALSKI, Krzysztof GÓRECKI

Uniwersytet Morski w Gdyni, Wydział Elektryczny, Katedra Elektroniki Morskiej

ORCID: 1. 0000-0002-5437-381X 2. 0000-0002-9857-8235

doi:10.xxxxx/xx.xxxx.

**Ewolucja programowalnych płytek jako odpowiedź na zmiany rynkowe i rozwój paradygmatu technologicznego**

***Streszczenie***: *Artykuł przedstawia ewolucję programowalnych płytek w kontekście zmian rynkowych, technologicznych oraz zmieniającego się podejścia do projektowania systemów wbudowanych. Omówiono historyczny rozwój płytek ewaluacyjnych, ich wpływ na inżynierię oraz przejście od zamkniętych do otwartych standardów. Szczególną uwagę poświęcono roli otwartego oprogramowania i modularnych architektur w kształtowaniu współczesnych rozwiązań. Na przykładzie projektu* ***OpenCPLC*** *pokazano, jak warstwowe podejście do projektowania może zwiększyć elastyczność i zastosowanie sterowników PLC zarówno w przemyśle, jak i edukacji.*

***Abstract***: *This article explores the evolution of programmable boards in the context of market shifts, technological advancements, and the changing approach to embedded system design. It examines the historical development of evaluation boards, their impact on engineering, and the transition from closed to open standards. Focus is given to the role of open-source software and modular architectures in shaping modern solutions. Using the* ***OpenCPLC*** *project as an example, the paper illustrates how a layered design approach can enhance the flexibility and applicability of PLC controllers in both industrial and educational environments.*

**Słowa kluczowe:** płyty ewaluacyjne, mikrokontrolery, sterowniki PLC, otwarte standardy, systemy wbudowane, OpenCPLC, warstwowa architektura

**Keywords:** evaluation boards, microcontrollers, PLC controllers, open standards, embedded systems, OpenCPLC, layered architecture

# Wstęp

**Płyty ewaluacyjne,** znane także jako płytki deweloperskie, odgrywają istotną rolę w rozwoju systemów wbudowanych od początków mikroelektroniki. Są nieodzownym narzędziem w procesie projektowania, zwłaszcza na wczesnych etapach prac, a także pełnią funkcję edukacyjną i demonstracyjną. Ich rozwój nie przebiegał w izolacji, lecz był ściśle związany z ewolucją mikrokontrolerów i narzędzi programistycznych umożliwiających ich programowanie. Płyty te były raczej wynikiem postępu technologicznego mikrokontrolerów i dostępnych narzędzi niż samodzielnie rozwijającym się elementem. Całość wpisuje się w kontekst dynamicznie zmieniającej się technologii, rosnących potrzeb inżynierów i wymagań rynkowych. Przyjrzyjmy się, jak te narzędzia ewoluowały na przestrzeni lat. Od wczesnych lat 70. XX wieku aż po współczesność.

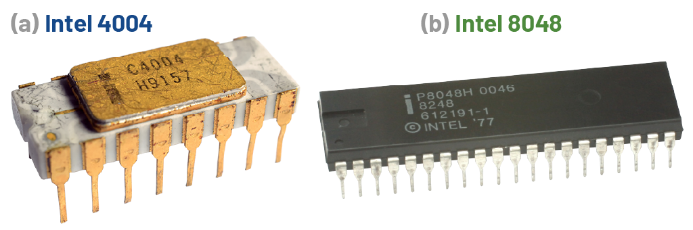
Wczesne komputery *(np.* ***IBM****)* oraz późniejsze mikrokomputery *(np.* ***Raspberry Pi****)*, przeznaczone do pracy z systemami operacyjnymi, także mogą być uznawane za płyty ewaluacyjne i niewątpliwie stanowią część systemów wbudowanych. W niniejszym artykule pominięto jednak ten temat, aby skupić się wyłącznie na mikrokontrolerach.

# Narodziny Mikroprocesorów

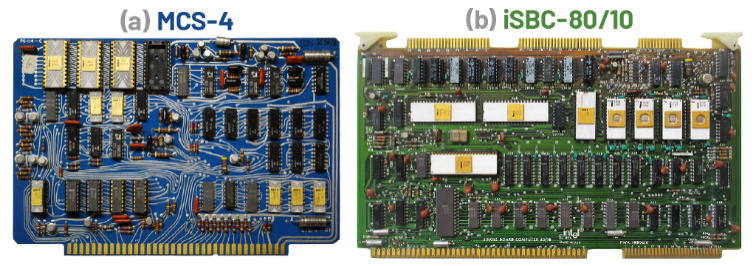
Początek lat 70. to okres, w którym zaczęto rozwijać mikroprocesory. Jako pierwszy pojawił się 4-bitowy **Intel 4004** *(rys. 1a, rys. 2a)*, który był jednostką arytmetyczno-logiczną *(ALU)*, wymagającym podłączenia dodatkowych układów zewnętrznych, takich jak kontrolery I/O, pamięci RAM i ROM. Programowanie odbywało się przez zapisywanie programu w pamięci **ROM**, która była trwale programowana w procesie produkcyjnym, a schematy logiki często były projektowane przy użyciu kart perforowanych.

W kolejnych latach na rynku zaczęły pojawiać się bardziej kompletne układy, które można już było określić mianem mikrokontrolerów, na przykład **TMS 1000** od Texas Instruments oraz 8-bitowy **Intel 8048** *(rys. 1b, rys. 2b)*. Układy programowane były w **asemblerze**, z wykorzystaniem prostych terminali. Zastosowanie pamięci **EEPROM** pozwoliło na wielokrotne programowanie bez wymiany układu, ułatwiając szybkie modyfikacje kodu.

W tamtych czasach płyty rozwojowe *(deweloperskie)* nie były postrzegane jako narzędzia do nauki czy eksperymentowania, ale jako kluczowe narzędzia mające na celu usprawnienie realizacji konkretnych projektów inżynieryjnych, a ich funkcjonalność była zbliżona do samego mikrokontrolera.



Rys. 1. (a) Mikroprocesor Intel 4004; (b) Mikrokontroler 8048



Rys. 2. Płyty rozwojowe: (a) Rodzina MCS-4, w tym Intel 4004; (b) iSBC-80/10 z mikrokontrolerem 8048

**Wczesny rozwój techniki mikroprocesorowej**

Szybko dostrzeżono, że nadchodzi technologiczna rewolucja, a technologie oparte na mikrokontrolerach, rozwijane od lat 70., mają znaczący potencjał rozwoju i ulepszeń. Pojawiła się zatem potrzeba zaprezentowania ich funkcjonalności szerszemu gronu odbiorców. W odpowiedzi na to płyty deweloperskie zaczęły być projektowane jako bardziej przemyślane i kompleksowe produkty, przeznaczone dla inżynierów elektroniki z różnych sektorów.

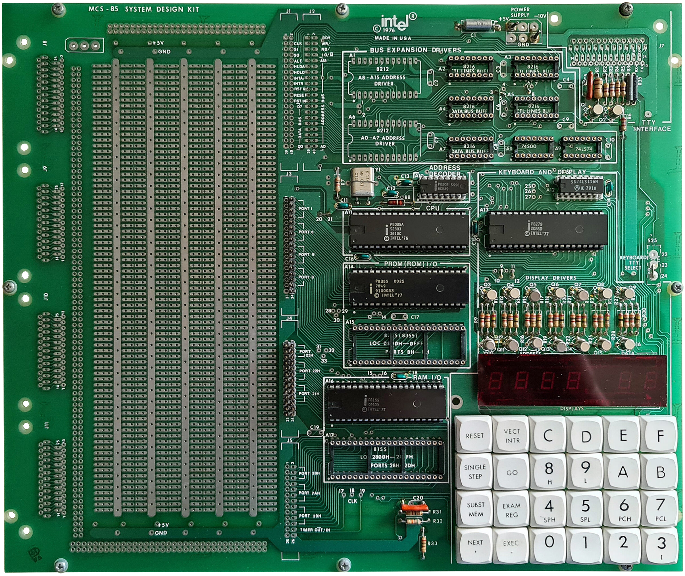
Przełomowa konstrukcja prototypowa, **SDK-85** z mikroprocesorem Intel **8085** *(rys. 2)*, zapewniała kompleksowe środowisko testowe. Umożliwiała programowanie w asemblerze zarówno przez zewnętrzny terminal podłączany przez interfejs pętli prądowej **20mA**, jak i za pomocą wbudowanej **klawiatury** oraz **wyświetlacza**. Było to szczególnie istotne w czasach, kiedy poziom komputeryzacji był jeszcze niski, a taka opcja okazywała się bardziej ergonomiczna dla większości inżynierów. Ponadto, klawiaturę i wyświetlacz można było wykorzystać bezpośrednio z poziomu aplikacji. Płyta rozwojowa SDK-85 jest wyposażona również w:

* Zegar systemowy **3MHz**
* Pamięć RAM **256B**/512B
* Pamięć EPROM **2kB**/4kB
* Timer **14-bit**owy programowalny
* **22** linie **I/O** *(+16 I/O z rozszrzeniem)*

W ślad za Intelem inni producenci również wprowadzili własne płyty deweloperskie, m.in.:

* Zilog ZDS Z80
* Motorola M68HC11 Evaluation Board
* National Semiconductor SC/MP II
* Texas Instruments TMS 9900
* MOS Technology KIM-1

Jednak żadna z nich nie osiągnęła tak dużego sukcesu jak SDK-85, głównie z powodu późniejszego wejścia na rynek. Intel zaprezentował SDK-85 już w 1976 roku, a w 1980 roku udoskonaloną wersję **SDK-86**, wspierającą systemy operacyjne. Konkurencyjne rozwiązania pojawiły się później, co znacząco ograniczyło ich rynkową przewagę.



Rys. 3. Płyty rozwojowe SDK-85 z mikroprocesorem 8085

# Początki język C oraz IDE

W latach 80. pojawiły się pierwsze mikrokontrolery obsługujące **język C**, co znacząco ułatwiło rozwój oprogramowania. Jednym z pierwszych był **Intel 8051** z płytą rozwojową **EVK-51**, który zyskał popularność dzięki ekosystemowi Keil, integrującemu kluczowe narzędzia. W jego skład wchodził kompilator **Keil C51**, zintegrowane środowisko programistyczne (1982) z wbudowanym debugerem oraz programatory, takie jak Willem EPROM Programmer, umożliwiające zapis wsadu do pamięci mikrokontrolera za pomocą **ISP** *(In-System Programming)*, a w późniejszych wersjach również przez **UART** *(RS-232)*. Jednak za sprawą szerszego wsparcia w koło lat 90 największą popularnością cziszył się IAR Em,beded Workbanch, który jednak operał się na tych samych założeniach.

Inżynierowie oczekiwali narzędzi, które pozwalały na szybkie i wygodne tworzenie oprogramowania bez konieczności ręcznej konfiguracji wielu oddzielnych komponentów. **Keil** wyróżniał się kompletną integracją, stabilnością oraz wsparciem technicznym, co zapewniło mu dominację na rynku.

Wprowadzenie **języka** **C** do systemów wbudowanych ułatwiło rozwój oprogramowania, zapewniając kontrolę nad sprzętem i przenośność kodu między różnymi architekturami. Dzięki efektywności, elastyczności oraz ugruntowanej pozycji, jaką zdobył jako pierwszy szeroko stosowany język w systemach wbudowanych, C do dziś pozostaje dominującym standardem w programowaniu mikrokontrolerów.

# Dalszy rozwój mikroprocesorów

W połowie lat 80. wyraźnie ukształtował się podział między mikroprocesorami **CPU** a mikrokontrolerami **MPU**. Mikroprocesory rozwijano z myślą o komputerach osobistych i systemach operacyjnych, natomiast mikrokontrolery integrowały w jednym układzie scalonym coraz więcej peryferiów takich jak przetwoniki **ADC** i DAC, komparatory, interfejsy **SPI**, **UART** i **I²C**, skomplikowane układy taktowania z pętlą **PLL** *(tab. 1)*. Obecnie wiele z tych funkcji stał się standardem i są integrowane w więkosżci konstrukcji MPU ogónego przeznaczniea, eliminując potrzebę stosowania wielu zewnętrznych komponentów, także na płytach demonstracyjnych.

Płyty przeznaczone do pracy z "pełnoprawnymi" mikrokontrolerami to m.in. seria **PICDEM** od Microchip oraz **STK500** dla układów **AVR** firmy Atmel. Były one kierowane głównie do profesjonalistów oraz studentów uczelni technicznych zajmujących się systemami wbudowanymi. Wyposażono je w liczne złącza oraz elementy testowe, takie jak diody LED, przyciski i wyświetlacze, co umożliwiało szybkie wdrażanie i testowanie nowych rozwiązań. Dzięki zastosowaniu podstawek pozwalały na pracę z wieloma modelami mikrokontrolerów. Płyty te stanowią reprezentatywny przykład rozwiązań swojej epoki.

Wykorzystywały dedykowane interfejsy programowania **ICSP** *(Microchip)*, **ISP** *(Atmel)* oraz **JTAG**, który jako pierwszy stał się otwartym standardem do programowania i debugowania. Szybko zyskał popularność, a większość współczesnych procesorów i mikrokontrolerów jest w niego wyposażona.

Wraz z pojawieniem się pierwszych porządnych/dopracoanych (zmień to słowo) edytorów o architrktó®zze modułowj (pozwaljących na integracje własnych rozwiązań), ktakich jak NetBeens oraz Visual Studio Potencaci tametgo okresu postanowimu je zykożystać. Microchp przerobił NetBeens na MPLAB a Atmel Visal Studio na AVR Studio (puźniej atmel studuiio)

Pod koniec XX wieku mikrokontrolery zyskały na popularności w przemyśle, a wraz z nimi pojawiły się pierwsze układy oparte na architekturze **ARM7**, łączące wysoką wydajność z energooszczędnością. Do pionierów należały **LPC2000** od Philipsa *(obecnie NXP)* z płytą **LPC-P2148** od Olimex, oraz **STR7** od STMicroelectronics, które zapoczątkowały erę **32-bit**owych mikrokontrolerów. Choć mikrokontrolery idealnie odpowiadały na potrzeby rynku, dostrzeżono, że efektywniejszym rozwiązaniem jest oddzielenie rozwoju rdzenia, licencjonowanego przez **ARM**, od projektowania układów peryferyjnych przez producentów mikrokontrolerów, co potwierdza popularność procesorów ARM w kolejnych latach.

# Minimalizm i otwarte standardy

Na początku XXI wieku, wraz z pojawieniem się platformy **Arduino**, zmieniło się podejście do projektowania płyt ewaluacyjnych. Nowa koncepcja zakładała minimalizację kosztów poprzez ograniczenie liczby komponentów na płytach rozwojowych, co zwiększyło ich dostępność dla szerokiego grona użytkowników, dzięki redukcji bariery wejścia poprzez dedykowane IDE, wsparcie jedynie wybranych mikrokontrolerów, a także wysokiej jakości dokumentacji i materiałów dodatkowych. Efektem tego był rozwój aktywnej społeczności, której zaangażowanie w tworzenie nowych rozwiązań dodatkowo napędzało dalszy rozwój platformy.

Standardowa konstrukcja tych płytek obejmowała jedynie niezbędne elementy, takie jak mikroprocesor z bootloaderem *(umożliwiający programowanie bez zewnętrznego programatora)*, stabilizator napięcia, rezonator kwarcowy oraz listwy kołkowe *(goldpiny)* do podłączania układów peryferyjnych.

Z czasem producenci modułów rozwojowych zaczęli dostosowywać wyprowadzenia płytek testowych w taki sposób, aby były kompatybilne z tym standardowym rozstawem złączy, co ułatwiło ich integrację i stało się powszechną praktyką dzięki zastosowaniu licencji **open hardware** *(CC BY-SA)*. Takie rozwiązanie zredukowało koszty, trafiając do jeszcze szerszego grona odbiorców. Śladem Arduino podążył Microchip, wprowadzając otwarty standard **microBUS**, który mimo bardziej dopracowanego projektu nie osiągnął tak dużej popularności.

Od tego momentu większość modułów i płyt testowych zaczęła być udostępniana jako otwarte projekty. Choć oferowały jedynie podstawową funkcjonalność, towarzyszyła im pełna dokumentacja techniczna, co znacznie ułatwiało ich integrację w bardziej złożonych systemach. Dzięki temu inżynierowie systemów wbudowanych mogą dobrać odpowiednie moduły i rozpocząć prace deweloperskie jeszcze przed zaprojektowaniem płytki PCB. Dodatkowym ułatwieniem są udostępniane na licencji **open source** biblioteki demonstracyjne, które przyspieszają proces wdrożenia.

Obecnie najpopularniejszą platformą deweloperską jest płytka **Nucleo**, co jest bezpośrednio powiązane z sukcesem mikrokontrolerów **STM32**. Istotną rolę odgrywa niska cena płytki, oscylująca w okolicach 45-85zł, która na dodatek zawiera programator/debugger **ST-Link**. Wśród innych popularnych platform znajdują się **FRDM** od NXP oraz **LaunchPad** od*(Texas Instruments)*. Jednak kluczowym elementem decydującym o popularności danego rozwiązania jest jego *"ekosystem"*. Oprócz wysokiej jakości dokumentacji, użytkownicy oczekują dostępności bibliotek, przykładów demonstracyjnych zarówno dla firmware, jak i software, oraz zapewnienia długoterminowego wsparcia. Rozwój tego ekosystemu nie musi zależeć wyłącznie od producentów mikrokontrolerów. Firmy produkujące moduły, takie jak Waveshare, Adafruit, oraz społeczności publikujące na forach czy **GitHub**, mogą aktywnie przyczyniać się do jego rozwoju. Z uwagi na zdywersyfikowaną naturę tego *"ekosystemu"*, otwartość publikowanych produktów staje się kluczowym czynnikiem jego sukcesu. Płyty deweloperskie stanowią ważną część tego ekosystemu, a ich sukces jest ściśle skorelowany z jego kondycją na rynku.

Na początku XXI wieku podejście do środowisk IDE zaczęło się zmieniać. Coraz więcej inżynierów dostrzegało zależności między kompilatorem, debugerem, programatorem i językiem programowania, co uwidoczniło ograniczenia silnie zintegrowanych środowisk. Ich zamknięta architektura często utrudniała elastyczne dostosowanie do specyficznych wymagań projektowych. Odpowiedzią stał się modułowy i otwarty **Eclipse**, wspierany przez IBM, który dzięki systemowi wtyczek umożliwiał swobodne rozszerzanie funkcjonalności. W kolejnych latach producenci mikrokontrolerów zaczęli tworzyć na jego bazie dedykowane wersje, takie jak **STM32CubeIDE**.

Coraz większą popularność zdobywa **VSCode** *(Visual Studio Code)*, wyróżniający się szybkością działania i otwartą architekturą. Jego lekka struktura zużywa mniej zasobów niż tradycyjne IDE, a rozbudowany ekosystem rozszerzeń umożliwia łatwą integrację z narzędziami do programowania systemów wbudowanych. Obecnie Arduino zrezygnowało z dalszego rozwoju swojego IDE, adaptując VSCode do swoich potrzeb.

# 

Rys. X. Ewolucja IDE uwzględniając ich popularność

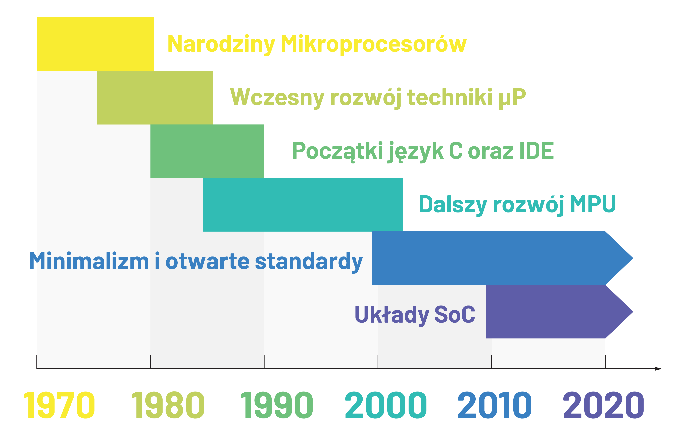
# Układy SoC

Gdy dana funkcjonalność zaczęła pojawiać się coraz częściej, zwłaszcza w przypadku interfejsów i protokołów komunikacyjnych takich jak USB, CAN czy Ethernet była ona integrowana jako kolejny układ peryferyny rozbudowują mikrokontorlr (tab 1). Jednak jak układy były już zbyt rozbudowane i skopikowane jak w przyapdku bluetootch, Wi-Fi czy LoRa, naturalną konsekwencją było zamknęcie w jednym układe scalonych kilku modłów z słąbszą integracją. Nie za pomoną rjestrów jak w przypadku peryferiów, magistral (np. Wewnętrze sPI) lub osobnych rdzeni z firmware. W efekcie wcześniej zewnętrzne układy stały się cześcią ukłądu programowalnego, a takie rozwiązania określamy mianem **SoC** *(****System on Chip****)*.

W przypadku płytek deweloperskich z tego typu układami konieczne jest wyposażenie ich w odpowiednie komponenty umożliwiające pełne wykorzystanie dostępnych funkcji. Obejmuje to między innymi układy toru radiowego w systemach bezprzewodowych oraz transceivery dostosowujące poziomy sygnałów i sposób transmisji do wymagań przewodowych standardów komunikacyjnych.

Najpopularniejszym SoC jest **ESP32**, które zdobyło uznanie dzięki niskiej cenie i łatwości implementacji. Zawiera zintegrowane moduły Wi-Fi i Bluetooth. W bardziej zaawansowanych zastosowaniach powszechnie stosowane są układy firmy **Nordic** Semiconductor *(****nRF52****)* oraz **STMicroelectronics** (STM32WB, STM32WL). Rozwój tych technologii jest ściśle związany z Internetem Rzeczy *(****IoT****)* oraz informatyzacją przemysłu *(Przemysł 4.0).*

# Podsumowanie



Rys. X. Chronologiczny układ rozdziałów w pracy

Tabela. 1. Lista przełomowych (w których nowa funkcjonalość stała się standardem lub zystkała sporą popularość) lub popularnych mikroprocesorów wraz z płytami ewaluacyjnymi i datami premiery

|  |  |  |  |
| --- | --- | --- | --- |
| Rok | MPU lub SoC | Peryferia i cechy szczególne | Devboard |
| 1976 | Intel 8048 | **RAM** + **ROM** + Timer | Intel MCS-48 Dev Kit |
| 1976 | Intel 8051 | **UART** | 8051 Development Board *(Keil MCB51)* |
| 1978 | Texas Instruments TMS770C | **ADC** *(10-bit)* | TMS770C Development Kit |
| 1980 | Motorola 68HC05 | **SPI** | Motorola 68HC05 EVB |
| 1982 | Intel 8096 | **PWM** | Intel 8096 Evaluation Board |
| 1985 | Philips 8051XA | **I²C** | Keil MCB51 *(dla 8051)* |
| 1986 | Atmel AT89C51 | Kompatybilny z Intel 8051, popularny w edukacji | Atmel AT89C51 Development Kit |
| 1987 | Motorola MC68HC11 | Pętlą **PLL** | Motorola MC68HC11 EVB |
| 1992 | Freescale MC68HC05X4 | **CAN** | Freescale MC68HC05 EVB |
| 1996 | Motorola MC68HC16 | Sterowanie pojazdami ADAS | Motorola EVB68HC16 |
| 2000 | Philips LPC2000 | 32-bitowych  Popularny | LPC2148 Olimex |
| 2004 | Microchip PIC18F4550 | **USB** | PICDEM FS USB *(Microchip)* |
| 2007 | Microchip PIC18F97J60 | **Ethernet** | Microchip Ethernet Starter Kit |
| 2008 | Atmel ATmega328P | Popularny | Arduino Uno |
| 2012 | Nordic nRF51 | **Bluetooth** | Nordic nRF51 DK |
| 2013 | Infineon AURIX TC2xx | Sterowanie pojazdami ADAS | AURIX TC2xx Triboard |
| 2014 | ESP8266 *(Espressif)* | **Wi-Fi** | ESP8266 NodeMCU |
| 2016 | ESP32 *(Espressif)* | 2×Wi-Fi + Bluetooth | ESP32 DevKit V1 *(DOIT)* |
| 2019 | STM32H7 | Dostosowany  pod **AI** | STM32H7 Nucleo |
| 2020 | STM32WLE5 | **LoRa** | STM32WLE5 Nucleo |

Dynamiczny rozwój mikrokontrolerów, narzędzi programistycznych oraz płyt ewaluacyjnych ujawnia wyraźny trend. Początkowo dominowały komercyjne, zamknięte rozwiązania, które zdobywały rynek dzięki szybkiemu dopasowaniu do jego potrzeb. Jednak w dłuższej perspektywie okazywały się barierą innowacji, ustępując miejsca technologiom otwartoźródłowym. To właśnie otwarte standardy przetrwały próbę czasu i nadal są rozwijane, podczas gdy wiele zamkniętych ekosystemów zostało porzuconych.

Rozwój płytek ewaluacyjnych pokazał, że minimalizacja kosztów i maksymalne uproszczenie wdrożeń dzięki modułowemu podejściu, zastosowaniu otwartych standardów, solidnej dokumentacji oraz wysokiej jakości bibliotekom i przykładom to kluczowe czynniki zapewniające sukces na rynku.

Projektując nowe rozwiązania, warto opierać się na otwartych standardach oraz tworzyć oprogramowanie i rozwiązania sprzętowe w sposób otwarty i modułowy, z naciskiem na potrzeby odbiorcy, który ma je wdrażać i wykorzystywać. Kluczowe jest, aby proces ten jak najlepiej wspierał użytkownika końcowego. Jest to szczególnie istotne w środowisku akademickim oraz wszędzie tam, gdzie celem jest rozwój technologii, a nie wyłącznie krótkoterminowy zysk.

Nie można jednak całkowicie zaniedbać potrzeb rynku, o czym przypomina sukces SoC, gdzie silna integracja kontrastuje z podejściem modułowym. Mimo to, integracja ta przyczynia się do miniaturyzacji i obniżenia ostatecznego kosztu produktu końcowego.

# Sterowniki PLC

Sterowniki **PLC** *(Programmable Logic Controllers)* są programowalnymi urządzeniami, które można porównać do mikrokontrolerów, ponieważ posiadają układy peryferyjne I/O oraz interfejsy. Różnią się jednak pod względem standardów i specyfiki zastosowań. Mikrokontrolery są uniwersalnymi układami, które mogą być stosowane w różnych dziedzinach, podczas gdy sterowniki PLC są dedykowane automatyce przemysłowej i opierają się na mikrokontrolerze jako centralnym elemencie. W tym sensie sterownik PLC stanowi warstwę abstrakcji nad mikrokontrolerem, oferując dodatkowe funkcje wspierające procesy przemysłowe, ale jednocześnie mogąc ograniczać lub uniemożliwiać dostęp do niektórych funkcji mikrokontrolera.

Najpopularniejsze komercyjne sterowniki PLC, takie jak **Siemens** **S7-1200**, Wago 750-881 i Allen-Bradley MicroLogix 1400, współpracują z dedykowanym oprogramowaniem, które umożliwia tworzenie i wgrywanie programów. W centrum każdego sterownika znajduje się mikrokontroler odpowiedzialny za obliczenia, sterowanie peryferiami i komunikację. Dedykowane środowiska programistyczne narzucają jednak pewne ograniczenia, umożliwiając korzystanie jedynie z funkcji i bloków dostarczonych przez producenta. Choć ułatwia to tworzenie przejrzystych programów, ogranicza dostęp do niższych warstw, co bywa problematyczne przy zaawansowanych zastosowaniach. Dodatkowo sterowniki danego producenta mogą być programowane wyłącznie w jego środowisku, co prowadzi do większej zależności od dostawcy, ogranicza elastyczność programistów i może negatywnie wpływać na konkurencyjność oraz innowacyjność rynku.

Widać, że paradygmaty techonlogiczne w przypadku sterowników PLC są zuoełnie inne niż w przypadu mikrokontrerów, SoC i płyt programowalnych. Widać, że dużo ważnieksze niż otwartość powodująca innowacyjny i zrównoważony razwuj ustępujnie niezawodności oraz dopasowaniu do potrzeb branży i skóteczności producentów PLC na rynku (TODO. Uzupełnić zalety sterowników PLC). Nie oznacza to, że ten trend się nie odwróci. Początkowo wiele rozwiązań oprartyczh na kimkorntolrerach oraz wiele płyt rozwojowych prezontowało podobny charakter jak współczesne sterowniki. Świadczy o tym rosnąca populartność otwartych i warstwowych sterowników PLC, większość dostępnych rozwiązań opiera się na mikrokomputerach z systemem **Linux** i modułami rozszerzeń dostosowanymi do standardów automatyki.

Dotychczas jednak żadne w pełni otwarte rozwiązanie dedykowane prostszym i bardziej niezawodnym procesom, które bazowałoby na bezpośrednim programowaniu mikrokontrolera, nie zdobyło większj popularności. Takie podejście mogłoby zapewnić wyższą stabilność, mniejsze zużycie energii i krótszy czas uruchamiania, w porównaniu z sterownikami opartymi na mikrokomputerach, co czyniłoby je atrakcyjnym wyborem zarówno dla niektórych zastosowań przemysłowych.

Z edukacyjnego punktu widzenia takie rozwiązanie pozwala nie tylko na naukę programowania PLC, ale także na zrozumienie działania sprzętu i warstw abstrakcji. Dzięki temu inżynierowie zdobywają praktyczne umiejętności pracy z mikrokontrolerami, optymalizacji kodu i integracji z automatyką. Otwartość platformy sprzyja eksperymentom, rozwija myślenie systemowe i eliminuje ograniczenia typowych zamkniętych środowisk.

# OpenCPLC

Lukę w runku może wypełnić projekt **OpenCPLC**, który, mimo wczesnej fazy rozwoju, jest tworzony zgodnie z duchem otwartego oprogramowania. Wykorzystuje sprawdzone technologie, takie jak język C, IDE VSCode, kompilator ARM GCC, OpenOCD do programowania i debugowania oraz system **VRTS** do obsługi wielowątkowiści. Zamiast zamkniętych środowisk dodaje warstwy abstrakcji zgodne ze standardami automatyki, zarówno w postaci bibliotek HAL, jak i dedykowanej konstrukcji sprzętowej. Zapewnia apliajce wizard, która przygotowyuje niszbędne narzędzia do pracy oraz umżliwia pracę z workspace, poiprzez włynne przłczanie się między projetami

Zaleca pracę z VSCode z instalacją odpowiednich wtycznek.

Konstrukcje OpenCPLC to pełnoprawne sterownik PLC oparty na mikrokontrolerze STM32. Obsługują napięcia 12V, 24V i wejścia 230VAC, co pozwala na zastosowanie z maszynami przemysłowymi, w automatyce i systemach domowych. Wyposażony w przekaźniki, tranzystory, triaki i RS485, wspierający protokoły Modbus RTU i BACnet. Otwarta architektura i zgodność z popularnymi narzędziami programistycznymi eliminują ograniczenia zamkniętych systemów. Dzięki temu sterownik sprawdza się także jako platforma edukacyjna, pełniąc rolę płytki testowej i demonstracyjnej w laboratoriach.

# Wnioski końcowe

Ewolucja płyt ewaluacyjnych odzwierciedla dynamiczny rozwój technologii i zmieniające się potrzeby rynkowe. Od lat 70., kiedy płyty deweloperskie były projektowane głównie na potrzeby przemysłu, po lata 80. i 90., kiedy kluczowym było prezentowanie możliwości układów na szerszą skalę, czego efektem było bogactwo i różnorodność układów peryferyjnych i demonstracyjnych. Od przełomu wieku XIX, mikrokontrolery stały się nieodłącznym, a w przemyśle nawet kluczowym elementem mikroelektroniki. W każdej epoce układy programowalne i ich ekosystemy dostosowywały się do bieżących potrzeb rynkowych.

Brak większych zmian koncepcyjnych w ostatnich latach może świadczyć o dojrzałości rynku i braku potrzeby zmian. Projekt OpenCPLC, odbiega od głównego nurtu, zarówno w porównaniu z płytami rozwojowymi, jak i sterownikami PLC. Pod kontem przeznaczenia nie rózni się od sterowników PLC jednak z punktu widzenia paradygmatu technologicznegi jest znacznie bardziej podobnt do płytu deweloperskij.

To połączenie, oferując zarówno teoretyczną naukę, jak i praktyczne zastosowania w rzeczywistych warunkach automatyki, jest kluczowe dla przygotowania przyszłych inżynierów do zawodowych wyzwań w dynamicznie rozwijającym się przemyśle. Niezależnie od kierunku ewolucji technologicznej, płyty ewaluacyjne pozostają niezbędnym narzędziem w inżynierii elektronicznej, odgrywając fundamentalną rolę w procesie tworzenia nowoczesnych urządzeń elektronicznych.