
# Jak dobierać mikrokontrolery?

*Doboru mikrokontrolera do tworzonego projektu dokonuje się tuż po rozpoczęciu prac. Biorąc pod uwagę ofertę rynkową, dostępne są trzy grupy mikrokontrolerów: 8-bitowe, 16-bitowe i 32-bitowe. W wielu wypadkach wybór mikrokontrolera jest bezpośrednio związany z obraną strategią rozwoju projektu. Najczęściej twórcy decydują się na jedno z dwóch podejść: budowa nieskomplikowanej wersji urządzenia w oparciu o tani mikrokontroler, a następnie jego rozwijanie lub wykonanie pierwszego projektu z użyciem zaawansowanego układu, a następnie tworzenie prostszych odmian produktu.*

Wybór mikrokontrolera nie jest ograniczony żadnymi ścisłymi zasadami, ale istnieje wiele czynników, które powinny mieć wpływ na podejmowaną decyzję, takich jak zapotrzebowanie na moc obliczeniową czy też zużycie energii. Nierzadko trudno jest ocenić, które z cech mikrokontrolera będą najważniejsze w danej aplikacji. Przykładowo, odporność na podwyższone temperatury lub dostępność odpowiednich interfejsów i mały pobór mocy mogą być wymaganiami sprzecznymi, które sprawiają, że inżynier musi iść na kompromis. Dobrą metodą rozwiązania tego problemu jest stworzenie opisu wymagań, który ułatwi porównywanie cech poszczególnych układów.

## Opis wymagań projektowych

Pierwszy dział opisu powinien dotyczyć podstawowej funkcjonalności, czyli cech kluczowych dla danego projektu. Powinien zawierać informacje o tym:

- jakie zadania będzie wykonywać projektowany system?
- jakie są sygnały wejściowe i wyjściowe?
- ile miejsca na dane będzie potrzebne?
- jak szybko system musi reagować na zdarzenia?

Przykład pierwszej części opisu dla prostego sterownika termostatu zamieszczono w **tabeli 1**. W kolejnej części warto opisać ograniczenia projektowe. Składają się na nie odpowiedzi na pytania:

- jaki ma być docelowy koszt podzespołów i montażu?
- jaką maksymalną moc może pobierać urządzenie?
- jakie są ograniczenia odnośnie do wymiarów?
- w jakim środowisku będzie pracowało gotowe urządzenie?

Przykład opisu odpowiadającego na powyższe pytania dla prostego sterownika termostatu został zademonstrowany w ramce.

## Specyfikacja wstępna

W oparciu o spisane wymagania można wyspecyfikować listę zasobów, którymi powinien dysponować system. Zasoby te obejmują wielkość pamięci operacyjnej, danych i Flash, peryferia, obwody zasilające i wy-

**Tabela 1. Wstępne wymagania projektowe dla przykładowego, prostego, inteligentnego termostatu domowego**

| Lista funkcji | Pamięć danych | Wejścia/wyjścia | Szybkość |
|---|---|---|---|
| Sterowanie pracą grzałki.<br>Sterowanie pracą układu chłodzącego.<br>Przesyłanie danych do PC.<br>Odbieranie danych z PC.<br>Obsługa wyświetlacza LCD.<br>Obsługa pojemnościowego interfejsu dotykowego.<br>Monitorowanie temperatury.<br>Dekodowanie poleceń.<br>Zegar czasu rzeczywistego . | 32 bajty: bufor nadawania transmisji szeregowej.<br>32 bajty: bufor odbierania transmisji szeregowej/<br>20 bajtów: pamięć do parsowania poleceń.<br>10 bajtów: zegar czasu rzeczywistego.<br>20 bajtów: bufor LCD.<br>6 bajtów: generator dźwięków.<br>45 bajtów: pamięć na tymczasowe zmienne<br><br>RAZEM 165 bajtów | Wejście szeregowe RS232.<br>Wyjście szeregowe RS232.<br>Wejście dla interfejsu pojemnościowego.<br>Przełącznik ogrzewanie/chłodzenie.<br>Wejście czujnika temperatury/<br>Wyjście sterujące ogrzewaniem/chłodzeniem.<br>Wyjście wyświetlacza LCD.<br>Wyjście głośniczka piezoelektrycznego.<br>Wejście napięcia zasilania | 2 kHz – częstość odczytywania interfejsu dotykowego.<br>60 Hz – częstość aktualizacji wyświetlacza LCD.<br>4 kHz – generator dźwięków.<br>5 Hz- sterowanie ogrzewaniem/chłodzeniem.<br>1 Hz – sprawdzanie stanu baterii.<br>1 kHz – obsługa portu szeregowego (6900 baud) |

dajność. Kontynuując przykład termostatu, jesteśmy w stanie stwierdzić, że potrzebne jest 165 bajtów w pamięci danych oraz 2300 słów w pamięci Flash. Listę peryferiów można podzielić na te konieczne i przydatne. Do koniecznych zaliczymy obwody LCD, port USART i przetwornik A/C. Do przydatnych – interfejs do ekranu dotykowego i zegar czasu rzeczywistego. Wśród potrzebnych, dodatkowych obwodów elektrycznych znajdą się: czujnik temperatury, watchdog, wyjścia typu otwarty kolektor oraz regulator napięcia. Potrzebną moc obliczeniową można oszacować na około 0,5…1 MIPS.

Należy zaznaczyć, że na tym etapie tworzenia specyfikacji projektu, dokładność nie jest najważniejsza. Celem jest bowiem zgrubne oszacowanie wymaganych parametrów układu, aby móc porównywać ze sobą poszczególne cechy i możliwości wybieranego mikrokontrolera. Przykładowo, funkcje cechy i funkcje jak: sprzętowa implementacja procedur, reakcje w czasie rzeczywistym, praca w trybie uśpienia, odporność na trudne warunki elektryczne i łańcuchowe przetwarzanie sygnałów analogowych są typowe dla układów 8-bitowych. Układ z 32-bitowym rdzeniem warto wybrać, jeśli chce się implementować procedury w postaci dodatkowego oprogramowania, uzyskać wielozadaniowość (również z rygorami odpowiedzi w określonym czasie) lub dynamiczne zarządzanie energią zasilającą i dużą moc obliczeniową. Układy 16-bitowe są natomiast optymalne do rozwiązań pośrednich. Poniżej prezentujemy zalety i wady różnych rodzajów podejścia do tworzenia aplikacji z mikrokontrolerem.

## Sprzętowo czy programowo?

Wybór sprzętowej lub programowej realizacji funkcji ma poważne konsekwencje dla całego projektu. Jest to zarazem wybór pomiędzy małym zapotrzebowaniem na moc obliczeniową, a prostotą projektu. Przykładowo, programowy sterownik silnika krokowego zwiększa wymagania odnośnie szybkości CPU i wielkości pamięci, podczas gdy

zastosowanie sprzętowego kontrolera zwiększa wymiary płytki, a często także zużycie prądu.

Co więcej, podejmując decyzję trzeba się orientować w ukrytych konsekwencjach dokonanego wyboru. Przykładowo możliwe jest zrealizowanie funkcji wysyłania i odbierania danych poprzez emulowany port USART, ale w tym celu procedura nasłuchująca musi ciągle sprawdzać, czy nie pojawiło się opadające zbocze bitu startowego, by dokładnie zsynchronizować się z nadajnikiem. Pętla tego typu może zużywać dużą część mocy obliczeniowej procesora. Natomiast wyzwalanie funkcji odbierania danych przez USART za pomocą przerwania wywoływanego zmianą stanu na monitorowanym wejściu może powodować problemy z synchronizacją ze względu na zbyt duże opóźnienia pomiędzy rozpoczęciem transmisji a chwilą, gdy zacznie być ona odbierana.

## Własny czy RTOS?

Należy również podjąć decyzję na temat wielozadaniowości i tego, jak ma być ona realizowana. Pełny system operacyjny czasu rzeczywistego samodzielnie przełącza się pomiędzy poszczególnymi stanami i istotnie upraszcza programowanie. Jednakże własnoręcznie zaprojektowane maszyny stanów zaimplementowane w mikrokontrolerze zajmują mniej pamięci i wymagają mniej mocy obliczeniowej.

## Dwa czy więcej stanów uśpienia?

Generalnie rzecz ujmując, duża liczba różnych stanów uśpienia, w które można wprowadzić mikrokontroler, jest zaletą. Jednakże jest to cecha typowa dla 32-bitowych układów, podczas gdy prostsze MCU mają zazwyczaj jeden lub dwa tryby pracy o obniżonym poborze mocy. Często bywa tak, że układ może pracować w normalnym trybie lub zostać wyłączony. Dla wielu aplikacji, szczególnie tych prostszych, będzie to zupełnie wystarczający wachlarz możliwości, ale w bardziej zaawansowanych projektach korzystne może być obsłużenie większej liczby trybów usypiania, co pozwala zminimalizować średni pobór mocy.

## Niezawodność elektryczna

O ile w idealnym świecie, wszystkie urządzenia powinny być odporne na wszelkie wahania na linii zasilającej, w praktyce konieczny jest kompromis pomiędzy możliwościami regulacji napięcia zasilania , a jego niezawodnym dostarczaniem. Przykładowo, przetwornica napięcia może być zupełnie niezależnym, sprzętowym elementem, co najwyżej sterowanym za pomocą sygnału PWM, albo opierać się o wykorzystanie cyfrowej funkcji regulacyjnej, zależnej od mikrokontrolera który nią steruje. Oba podejścia sprawdzają się i są stosowane w praktycznych rozwiązaniach. Różnica pomiędzy nimi polega m.in. na tym, że w przypadku programowej realizacji sterowania, mikrokontroler musi nieprzerwanie, sprawnie pracować. Jeśli w jego programie pojawi się usterka, układ zasilania przestanie działać, podczas gdy w przypadku w pełni sprzętowego regulatora napięcia, system nie byłby narażony na tego typu błędy.

## Analogowe lub cyfrowe przetwarzanie sygnałów

Wybór sposoby przetwarzania nadchodzących sygnałów analogowych niesie za sobą bardzo duże konsekwencje odnośnie do wymagań stawianych mikrokontrolerowi. Gdy zdecydujemy się na przetwarzanie cyfrowe, ważne jest, aby MCU zapewniał wystarczającą przepustowość i miał odpowiednią liczbę przetworników A/C. Bez szybkich funkcji MAC (*multiply-and-accumulate*) wykonywanych w jednym cyklu mogłoby się okazać, że jednostka będzie zużywała całą swoją moc obliczeniową na operacje związane z DSP. Tymczasem budowa układu przetwarzającego sygnały analogowo, o ile pozwala zmniejszyć wymagania odnośnie do mikrokontrolera, w praktyce może okazać się zbyt skomplikowana dla bardziej złożonych operacji. Często optymalnym rozwiązaniem jest implementacja cyfrowego przetwarzania za pomocą oprogramowania w mikrokontrolerze, połączona z niezależnymi obwodami zabezpieczającymi.

## Wolumin produkcyjny

Ostatnia kwestia dotyczy tego czy projektowane urządzenie jest tylko modułem dużej, złożonej maszyny przemysłowej, o specyficznym interfejsie użytkownika obsługiwanym przez wykwalifikowanych operatorów, czy też ma być oferowane jako produkt powszechnie dostępny, wytwarzany w tysiącach sztuk. O ile w przypadku prostego, niedrogiego termostatu, raczej nie ma potrzeby stosowania kolorowego wyświetlacza graficznego z panelem dotykowym, to komponent ten mógłby się okazać przydatny w projekcie systemu sterowania klimatyzacją budynku biurowego. Wybór ten sprowadza się do określenia, jak dużo mocy obliczeniowej i jakie peryferia będą potrzebne do zapewnienia obsługi interfejsu użytkownika.

## Dobór mikrokontrolera

Gdy przeanalizujemy i podejmiemy decyzję odnośnie wszystkich opisanych kompromisów, można zabrać się do poszukiwania mikrokontrolera, który spełni te wymagania. Pomimo ogromnej liczby układów dostępnych na rynku nierzadko okaże się, że trudno jest dobrać dokładnie taki, jakiego potrzebujemy. Na tym etapie przyda się wcześniejszy podział na peryferia konieczne i opcjonalne. Konieczne może się bowiem okazać zrezygnowanie z niektórych opcji. Warto też pamiętać, że z każdą z opcji, na które się decydujemy rośnie koszt gotowego urządzenia. I nie chodzi tu tylko o cenę układu, który im bardziej rozbudowany, tym droższy, ale też o pozostałe komponenty potrzebne do realizacji dodatkowych funkcji, a nawet o nakłady konieczne na ich przetestowanie w gotowym produkcie. Nierzadko bywa tak, że wprowadzenie zaawansowanej funkcjonalności powoduje ogromne zwiększenie złożoności testów i może znacząco wydłużyć czas potrzebny na wypuszczenie produktu na rynek.

**Wstępne ograniczenia projektowe**

Koszt
- materiałów < 13 zł
- elementów mechanicznych < 3 zł
- montażu < 32 zł

Prąd zasilania
- podtrzymanie bateryjne < 20 µA
- normalne zasilanie < 5 mA

Wymiary
- 8 cm×10 cm×3 cm

Środowisko pracy
- temperatura: od 0 do 30 °C w trakcie pracy i od -20 do + 75 w trakcie przechowywania
- szum elektryczny: 1 KV ESD i odporność na zbliżanie telefonu komórkowego.

## Uwagi końcowe

Wybierając architekturę mikrokontrolera warto też zwrócić uwagę na wbudowane stosy komunikacyjne i dostępne biblioteki funkcji. Niektórzy producenci samodzielnie opracowują takie bloki programowe, odpowiadające np. za obsługę wyświetlaczy, szeregowe interfejsy komunikacyjne, czy obsługę ekranu dotykowego. Standardowe, przetestowane fragmenty kodu pozwalają znacząco skrócić czas projektowania, ale nie zawsze będą kompatybilne z danym projektem. Przykładowo – niektóre z nich mogą blokować dostęp do określonych peryferiów i uniemożliwiać wykorzystanie ich do realizacji innych funkcji. Istotne mogą być też ograniczenia czasowe narzucane przez gotowe biblioteki.

Warto też ponownie używać wcześniej napisany kod, bo skraca to czas programowania. Aby to było możliwe, rozsądnie jest wybrać producenta układów, który dostarcza różnorodne mikrokontrolery ale programowane w podobny, kompatybilny ze sobą sposób.

**Keith Curtis**
**Microchip Technology**