Mikrokontrolery STM32G0

Mikrokontrolery to elementy, które są używane w zastosowaniach, o których kiedyś nawet nie pomyślano. Niska cena powoduje, że wykorzystuje się je na szeroką skalę. Przez długi czas najtańsze były proste jednostki 8-bitowe, ponieważ ich struktury półprzewodnikowe zajmowały relatywnie małe powierzchnie krzemu. Wydajne mikrokontrolery 16-bitowe, a następnie 32-bitowe oferowały coraz większe możliwości, ale jednocześnie były dosyć drogie, jednak postęp technologiczny spowodował, że współcześnie 32-bitowe rdzenie są nierzadko tańsze od 8-bitoywch.

Rdzenie 32-bitowe, a szczególnie typu Cortex M, z powodu ich doskonałych parametrów stawały się coraz bardziej popularne i stosowane przez coraz większą rzeszę projektantów i programistów. Zapewne z tego powodu producenci „korteksowych 32-bitowców" postanowili wykorzystać tę popularność oraz przyzwyczajenia projektantów i zastosować tani w produkcji, uproszczony, 32-bitowy rdzeń Cortex-M0, a potem następcę – Cortex-M0+.

Rdzeń Cortex-M0+ jest w stanie konkurować cenowo z mikrokontrolerami 8-bitowymi. Te ostatnie trzymają się tradycyjnie mocno i ich producenci stają na głowie, aby tę pozycję utrzymać, ale w tym starciu argumenty techniczne są po stronie jednostek 32-bitowych. Jak się potoczą losy tego starcia, nie wiadomo, bo jak pokazuje historia, nie zawsze wygrywa to, co jest obiektywnie lepsze. Mogą decydować różne czynniki – na przykład przyzwyczajenia, polityka marketingowa firm, dostępność narzędzi projektowych itp.

Zwolenników mikrokontrolerów 32-bitowych z rdzeniem Cortex-M nie trzeba przekonywać, ale nawet najbardziej zatwardziali fani związani z tą czy inną firmą produkującą 8-bitowce mogą, a nawet powinni poznać alternatywne rozwiązania. Jednym z nich jest rodzina mikrokontrolerów STM32G0 z rdzeniem Cortex-M0+ produkowana przez firmę ST.

Jedną z ważnych cech mikrokontrolerów STM32 jest kompatybilność programowa i skalowalność architektury w ramach różnych rodzin, z różnymi rdzeniami. Innymi słowy, projektant lub programista stosujący mikrokontrolery STM32 może używać tanich procesorów o nieskomplikowanej architekturze, na przykład z rdzeniem Cortex-M0+, lub zależnie od potrzeb, bardziej wydajnych z rdzeniem

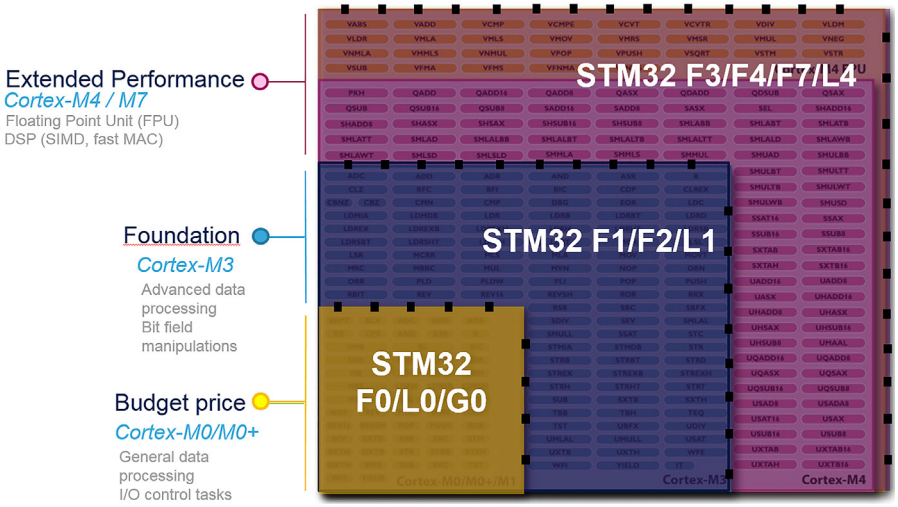
Cortex-M3 lub najbardziej rozbudowanych i wydajnych Cortex-M4 i Cortex-M7. Zmiana rodziny mikrokontrolerów nie wymaga przy tym zmiany przyzwyczajeń, poznawania nowych narzędzi projektowych (IDE, konfiguratora peryferii) itp. Również przenoszenie istniejących aplikacji jest bardzo uproszczone, jeżeli jest możliwe w ramach mikrokontrolerów. Na **rysunku 1** pokazano przypisanie rdzeni Cortex do rodzin mikrokontrolerów produkowanych przez ST.

Rodzina mikrokontrolerów STM32G0 jest ulepszonym następcą znanej i popularnej rodziny STM32F0. Podzielono ją w zależności od wyposażenia w układy peryferyjne na trzy linie: Value Line, Access Line, Access Line & Encryption. Wyposażenie w zasoby (pamięć, peryferia itp.) w zależności od linii zostało pokazano na **rysunku 2**. Spójrzmy, jakich modyfikacji użytkowych możemy się spodziewać.

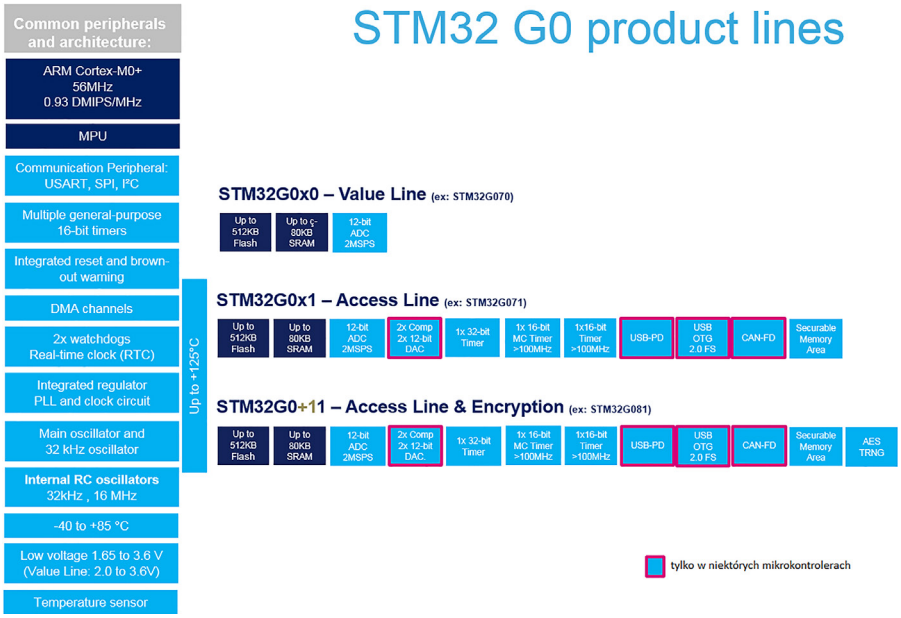
Podstawowym generatorem sygnału zegarowego w wielu mikrokontrolerach jest wbudowany w strukturę, precyzyjny generator RC. To źródło zegara jest powiązane z układem programowanego preskalera, a w jednostkach, które wymagają większej częstotliwości taktowania, dodatkowo jest wbudowany programowany układ PLL do powielania/dzielenia częstotliwości. Generator RC wykonany w strukturze układu scalonego nie jest zbyt dokładny, ponieważ źle radzi sobie ze zmianą temperatury otoczenia. Jednak z czasem dopracowano to rozwiązanie i w STM32G0 stabilność generatora jest na poziomie 1% w zakresie temperatury 0…85°C i +1,5% do –2% w zakresie temperatury –40…105°C. Generator o takich parametrach wystarcza do większości zastosowań. Wybór wewnętrznego taktowania RC pozwala na uproszczenie projektu płytki drukowanej. Na **rysunku 3** pokazano konfigurację taktowania z wykorzystaniem oscylatora RC HSI o częstotliwości 16 MHz. Tę częstotliwość powielono ×8 w układzie PLL i następnie podzielono przez 2. W wyniku tych operacji częstotliwość taktująca mikrokontroler ma wartość 64 MHz.

Obudowy mikrokontrolerów 32-bitowych często mają kilka par wyprowadzeń linii zasilających. Komplikuje to projekt płytki i ogranicza liczbę linii portów dostępnych w obudowie danego typu. W rodzinie STM32G0 w obudowach mających do 64 wyprowadzeń zasilanie jest doprowadzane tylko przez dwa wyprowadzenia zasilania. W obudowie 64-nóżkowej mamy do dyspozycji 60 linii I/O. Na **rysunku 4** pokazano obudowę LQFP64 mikrokontrolera STM32G071 z zaznaczonymi wyprowadzeniami zasilania.

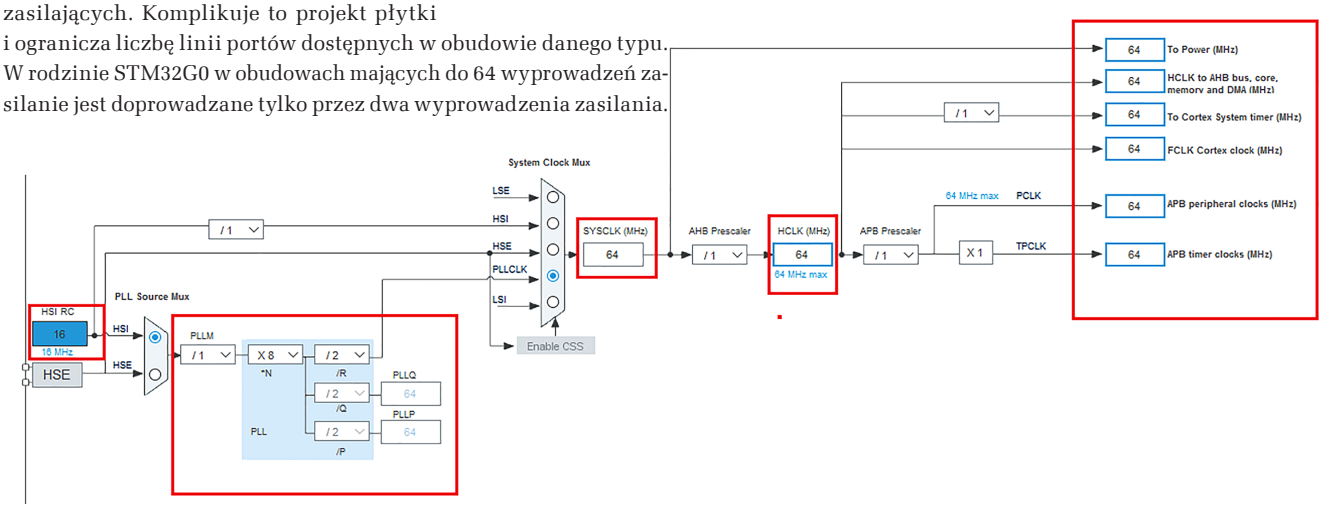
Intencje producenta odnośnie do wchodzenia w obszar tradycyjnie zajmowany przez 8-bitowce można poznać też po wielkości oferowanych obudów. STM32G0 są oferowane w obudowach



Rysunek 1. Rodziny mikrokontrolerów STM32 w powiązaniu z rdzeniami Cortex



Rysunek 2. Linie produktów STM32G0



Rysunek 3. Konfiguracja taktowania mikrokontrolera STM32G071 (STM32CubeMX)

od 8do 100 wyprowadzeń. Mikrokontrolery w obudowach 8-nóżkowych, mają od 32 do 64 kB pamięci Flash i 8 kB pamięci RAM. To w połączeniu z 32-bitowym rdzeniem dość sporo, jak na, w założeniu, proste aplikacje. Na **rysunku 5** pokazano wyposażenie w pamięć programu Flash i pamięć danych w funkcji liczby wyprowadzeń.

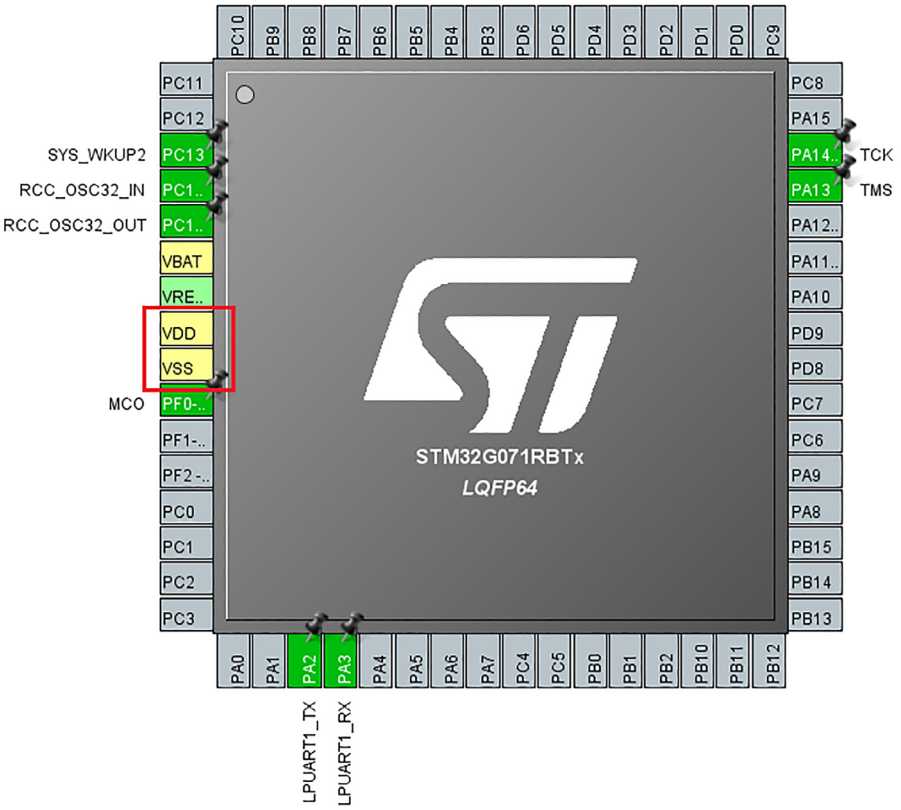
Jednymi z podstawowych cech mikrokontrolera, oczekiwanych szczególnie w kontekście wykorzystywania w aplikacjach IoT, są ogólnie pojęte niezawodność i bezpieczeństwo. Wiąże się to między innymi z zapewnieniem bezpieczeństwa aplikacji zapisanej w pamięci Flash. Znanym od lat problemem jest ochrona własności intelektualnej, która tu przyjmuje nowy wymiar, ponieważ aplikacja IoT musi również być odporna na próby fałszowania przesyłanych danych. Dlatego konieczne jest zabezpieczenie przed nieuprawnionym odczytaniem całej lub części pamięci kodu. Pamięć Flash w STM32G0 jest zabezpieczana następującymi mechanizmami:

- **RDP** (Readout Protection), który zabrania dostępu do rejestrów Flash/SRM/Backup przez debugger (SWD), kiedy program jest wykonywany z pamięci RAM lub jest aktywny bootloader.
- **PCROP** (Propiertary Code Protection), używany do ochrony zapisu i odczytu specyficznych obszarów pamięci programu. Chroniony kod może być tylko wykonywany (**rysunek 6**).
- **WRP** (Write Protection), który jest używany do ochrony określonego obszaru kodu przed zapisaniem i skasowaniem.

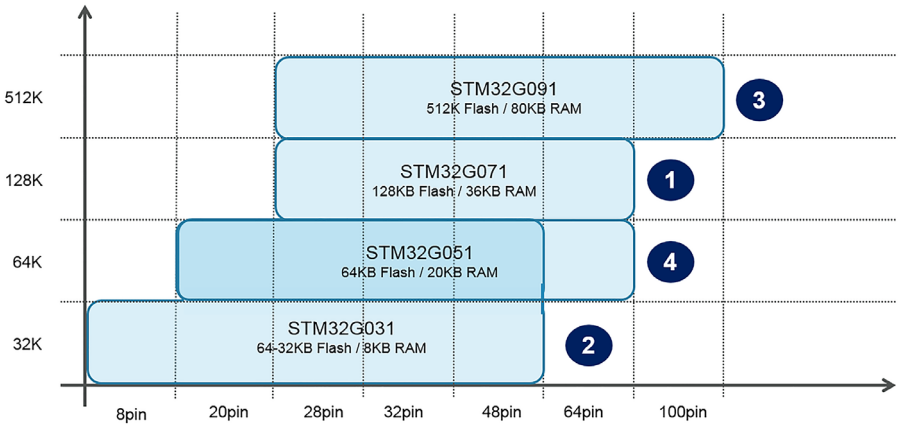
Użytkownik może programowo aktywować specjalny obszar pamięci o definiowanej wielkości, nazywany *Securable Memory Area*. Kiedy ten obszar jest aktywny, wtedy to każda próba dostępu do niej: pobranie kodu, odczyt, programowanie czy kasowanie, jest odrzucana i powoduje błąd magistrali. Tak definiowana pamięć może przechowywać, na przykład, klucze do kodów deszyfrujących (**rysunek 7**). Dezaktywacja jest możliwa za pomocą firmware'u użytkownika.

Oprócz opisanych zabezpieczeń duże znaczenie dla poprawnego działania programu ma bezbłędny zapis pamięci kodu Flash. Aby stwierdzić, czy pamięć jest zapisana poprawnie dla każdych dwóch zapisywanych słów 32-bitowych, jest obliczana 8-bitowa suma kontrolna CRC i ta suma jest umieszczana w specjalnym obszarze pamięci Flash. CRC pozwala na wykrycie i skorygowanie błędu odczytu na jednej pozycji. Przy przekłamaniach na większej liczbie pozycji korekcja nie jest możliwa, ale jest wykrywana i zgłaszana. Taki mechanizm nazywa się ECC (Error Code Correction). W pamięć Flash wbudowano dodatkowy obszar pamięci OTP (One Time Programmable) o wielkości 1 kB. Tę pamięć można zaprogramować tylko raz i nie można jej skasować. Jest również wydzielony specjalnie chroniony obszar na kod ST bootloader.

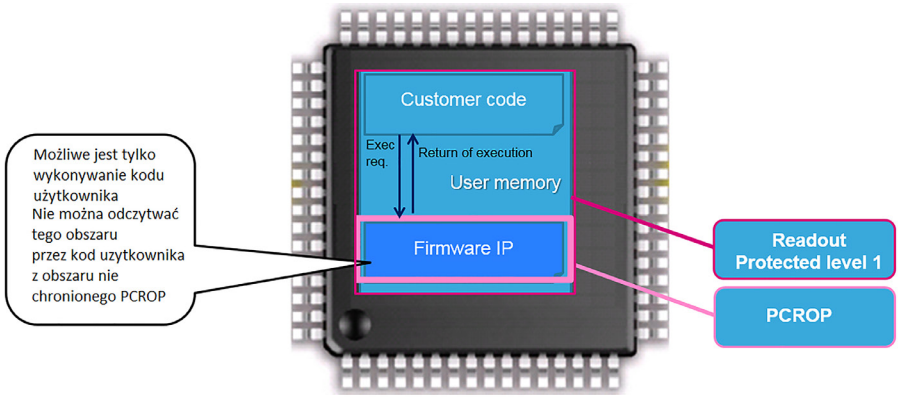
W urządzeniach IoT zwykle wymaga się możliwości komunikowania się z innymi urządzeniami przez łącze bezprzewodowe. Najczęściej są to standaryzowane łącza, na przykład BLE (Bluetooth V5), ale mogą to być jakieś własne rozwiązania. W wielu wypadkach jest niedopuszczalne, aby osoby postronne miały możliwość „podsłuchania"



Rysunek 4. Wyprowadzenia mikrokontrolera STM32G071



Rysunek 5. Wielkości pamięci programu i danych w funkcji liczby wyprowadzeń



Rysunek 6. Ochrona kodu PCROP

przesyłanych danych i ewentualnego włamania się do systemu komunikacyjnego. Dla zabezpieczenia stosuje się między innymi szyfrowanie transmisji. W najbardziej rozwiniętej linii rodziny SRTM32G0 do tego celu można wykorzystać wbudowany moduł szyfrujący AES256. Szyfruje on dane po stronie nadawcy za pomocą klucza użytkownika i praktycznie nie jest możliwe ich odczytanie bez znajomości tego klucza. Odbiorca, który ma klucz deszyfrujący, deszyfruje dane do pierwotnej postaci.

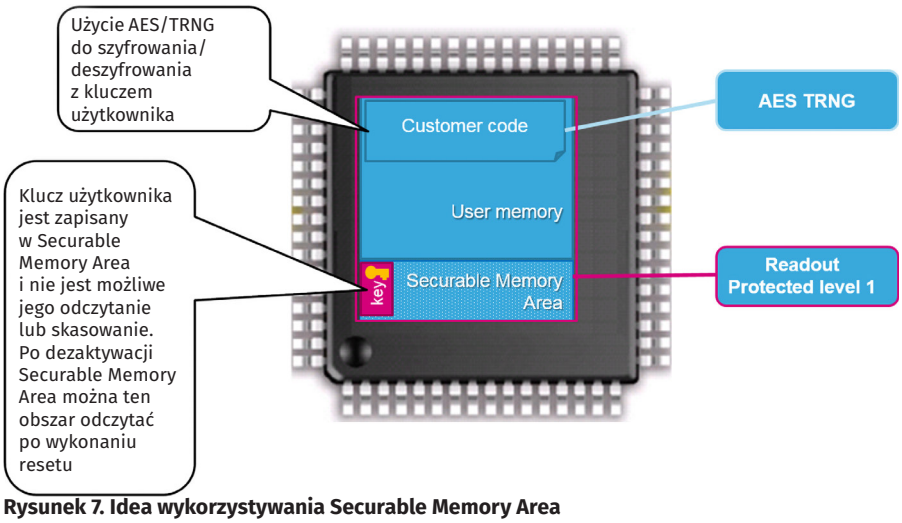
Rozwinięciem zabezpieczenia z szyfrowaniem jest szyfrowanie z autentykacją. Szyfrowanie zapewnia, że przesyłana wiadomość jest nieczytelna, a autentykacja, że nie została zmodyfikowana na przykład przez hakera, który jest w posiadaniu klucza. Na **rysunku 8** pokazano ideę szyfrowania z autentykacją stosowaną w STM32G0. Algorytmy szyfrujące wspiera dodatkowy moduł generowania liczb losowych RNG (Random Number Generation).

Jedną z istotnych modyfikacji rdzenia Cortex-M0+ w porównaniu do Cortex-M0 jest dodanie 16-bajtowej pamięci cache, do której są zapisywane pobierane kody rozkazów. Takie rozwiązanie, znane z wielu rozbudowanych i wydajnych jednostek, może wyraźnie zwiększyć szybkość pracy mikrokontrolera. W trakcie wykonywania kodu rozkazy są pobierane z szybkiej cache do momentu napotkania rozkazu skoku. Podstawowe różnice pomiędzy rodzinami STM32F0 i STM32G0 pokazano na **rysunku 9**.

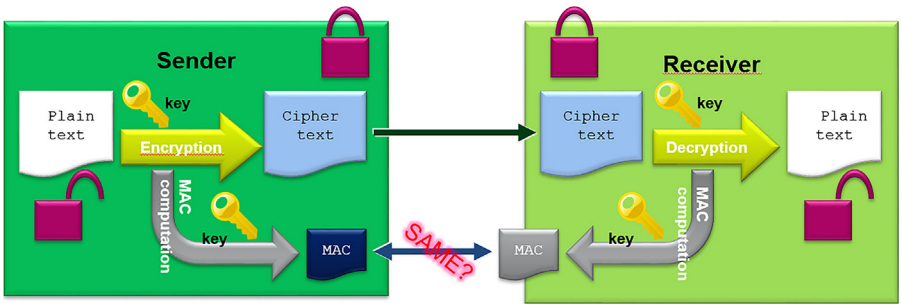
O atrakcyjności mikrokontrolera czy rodziny mikrokontrolerów, oprócz typowych właściwości rdzenia, układów bezpieczeństwa, czy zasobów typu pamięć programu i danych, decyduje też wyposażenie w moduły peryferyjne. Na **rysunku 10** pokazano schemat blokowy mikrokontrolera STM32G081 z najbardziej rozbudowanej linii Acces Line. Oprócz interfejsów komunikacyjnych: SPI, USART, I²C, rozbudowanych układów PWM i liczników, znajdziemy tu wspomniane bloki szyfrujące AES256, moduł komunikacyjny USB PDI i zmodyfikowany moduł przetwornika A/C. Peryferyjne układy analogowe są uzupełnione o 12-bitowy przetwornik C/A, komparator i czujnik temperatury.

Zmodyfikowany przetwornik analogowo-cyfrowy ma rozdzielczość 12-bitową i może wykonywać próbkowanie w czasie 0,4 µs. Dla przypomnienia, przetwornik w rodzinie STM32F0 mógł wykonywać konwersję w czasie 1 µs. Został też rozszerzony zakres napięcia pracy – wynosi on 1,62…3,6 V. W moduł przetwornika wbudowano układ nadpróbkowania pozwalający na rozszerzenie w pewnych warunkach rozdzielczości przetwornika do 16 bitów. Dokładne informacje można znaleźć w nocie aplikacyjnej AN2668. Do monitorowania progów napięć wejściowych przeznaczonych do konwersji można użyć trzech modułów Analog Watchog. Więcej informacji na ten temat można znaleźć w nocie aplikacyjnej AN2558.

Moduł USB PDI (Power Delivery Interface) 3.0 wspiera wszystkie tryby standardu zasilania urządzenia PD (Power Delivery). Mechanizm PD w założeniu może dostarczać do odbiornika do 100 W mocy przesyłanej. Możliwe jest przesyłanie energii w obu kierunkach, tzn. od hosta do urządzenia peryferyjnego i od urządzenia peryferyjnego do hosta. Ważną cechą sterowników urządzenia PD jest możliwość elastycznego dostarczania mocy zależnie od potrzeb urządzenia zasilanego, a także negocjowania wartości napięcia zasilającego w zakresie 5….20 V. Warstwa fizyczna modułu (PHY) wspiera obsługę złącza USB C. USB C ma więcej styków w porównaniu ze starszymi standardami:



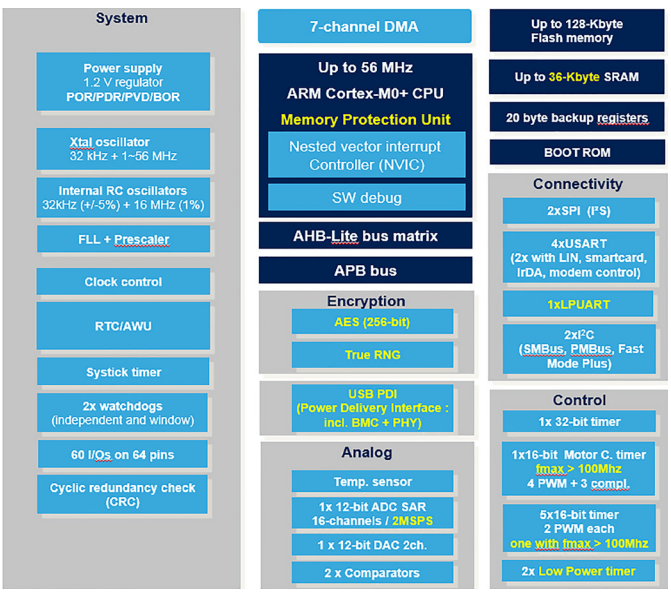
Rysunek 7. Idea wykorzystywania Securable Memory Area



Rysunek 8. Szyfrowanie z autentykacją

	STM32F0	STM32G0
Instruction Cache	No	16 bytes
OTP Area	No	1Kbyte
Fast Programming	No	Yes
PCROP + Securable Memory	No	Yes
ECC correction	No	Yes

Rysunek 9. Podstawowe różnice pomiędzy rodzinami STM32F0 i STM32G0



Rysunek 10. Schemat blokowy mikrokontrolera STM32G081

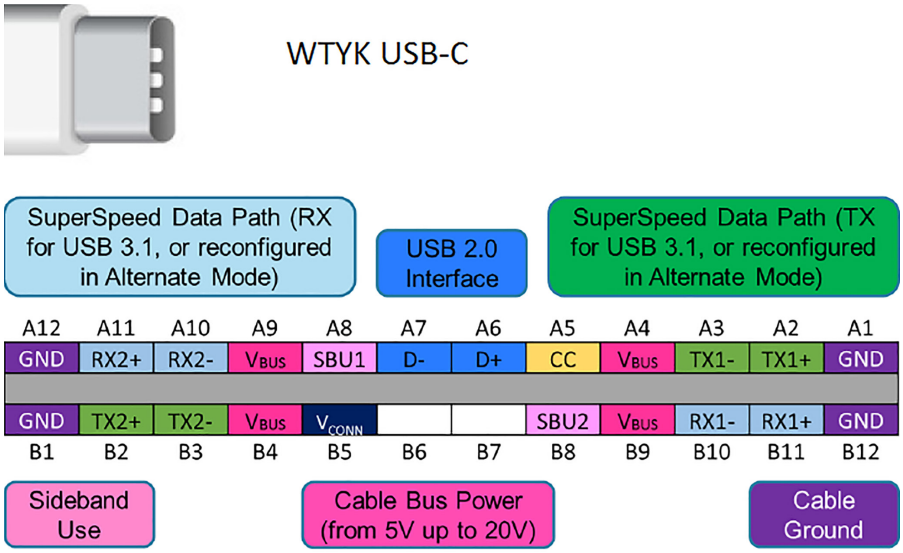
- **GND** (4 styki) i **VBUS** (4 styki) – do przesyłania napięcia zasilania: dla USB2.0 maksymalnie +5 V/500 mA, dla USB-C maksymalnie +20 V/5 A.
- **D+** i **D–**, będące liniami danych standardu USB2.0.

- **RX1–**, **RX1+** oraz **TX1–**, **TX1+**, będące magistralą danych SuperSpeed standardu USB 3.1.
- **SBU1**, **SBU2** – dodatkowa magistrala o mniejszej prędkości.
- **CC1**, **CC2** – magistrala danych przeznaczona do wykrywania wykonania połączenia, identyfikacji położenia wtyku względem gniazda i do negocjacji trybu pracy złącza USB oraz parametrów zasilania

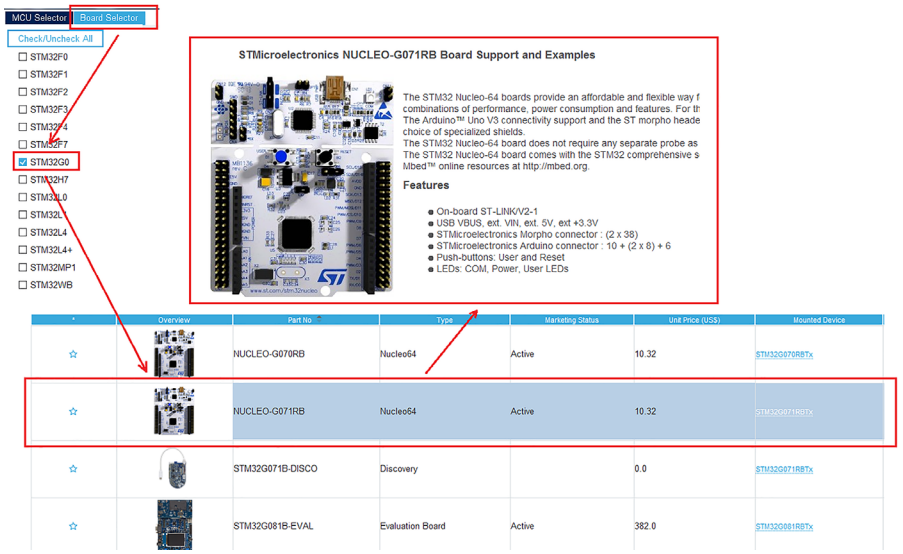
Styki są rozmieszczone symetrycznie, co umożliwia dowolne wkładanie wtyku do gniazda.

Ciekawym układem peryferyjnym jest moduł LPUART (Low Power UART), mogący pracować również w trybie interfejsu SPI Slave. LPUART w rodzinie STM32G0 został wyposażony w preskaler zegara taktującego transmisję i bufory FIFO oddzielnie dla toru odbioru danych RXFIFO i tory wysyłania danych TXFIFO. Jedną z unikalnych właściwości jest możliwość pracy z bardzo małym poziomem poboru mocy dla prędkości transmisji 9600 b/s przy taktowaniu oscylatorem LSE (oscylator pracujący z zewnętrznym oscylatorem kwarcowym „zegarkowym" o częstotliwości 32,768 kHz).

Producent mikrokontrolerów STM32G0 przygotował wsparcie dla projektantów w postaci modułów Nucleo (zgodnych z Arduino), zaawansowanego modułu ewaluacyjnego z mikrokontrolerem STM32G081 i niewielkiego modułu Discovery. W konfiguratorze STM32Cube można je wybrać w selektorze modułów (Board selector) i stworzyć szkielet programu z konfiguracjami układów peryferyjnych i ewentualnie firmware. Na **rysunku 11** pokazano wybór modułu NucleoG071RB za pomocą selektora Board Selector i rodziny STM32G0. Na **rysunku 12** przedstawiono przykładową konfigurację wyprowadzeń portu USB PDI w trybie Dual Role (jako źródło i jako odbiornik zasilania) w STM32CubeMX, natomiast na **rysunku 13** przykładową konfigurację portu USB PDI w STM32CubeMX.



Rysunek 11. Styki złącza USB C wspierane przez PHY USB PDI STM32G0



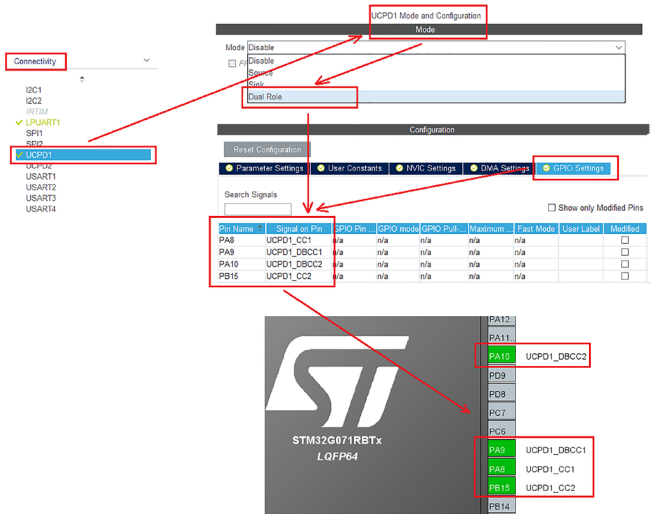
Rysunek 12. Wybór Nucleo G-017RB w STM32CubeMC

Podsumowanie

STM32G0 – ulepszona wersja rodziny STM32F0 przejmuje jej wszystkie zalety i dodaje sporo istotnych modyfikacji w postaci uproszczonego, ale bardziej wydajnego rdzenia z pamięcią cache, rozbudowanym sterownikiem pamięci Flash i precyzyjnego zegara taktującego RC HSI. Jeżeli założymy, że jednym z możliwych obszarów zastosowań są aplikacje tradycyjnie zajmowane przez mikrokontrolery 8-bitowe, na przykład z powodu niskiej ceny, to STM32G0 może na tym polu swobodnie konkurować.

Porównywalna cena to tylko jeden z atutów. Inne to 32-bitowa jednostka, bardzo dobrze wyposażona w tradycyjne moduły peryferyjne, mechanizmy wspierające stosowanie w aplikacjach IoT (np. AES256), porty USB PDI, ale też w obszerne pamięci programu i danych. Nie bez znaczenia jest również możliwość pracy w obecnie już dość zaawansowanym „ekosystemie" przeznaczonym dla mikrokontrolerów STM32 i obejmującym, obok tanich modułów ewaluacyjnych Nucleo z wbudowanym programatorem/debuggerem rozbudowane programy narzędziowe, takie jak STM32CubeMx i bezpłatne lub komercyjne środowiska projektowe, zintegrowane z kompilatorem języka C i z możliwością korzystania z firmowych bibliotek HAL.

Tomasz Jabłoński, EP



Rysunek 13. Przykładowa konfiguracja portu USB PDI w STM32CubeMX