1-PODSTAWY PIC18 TUTORIAL

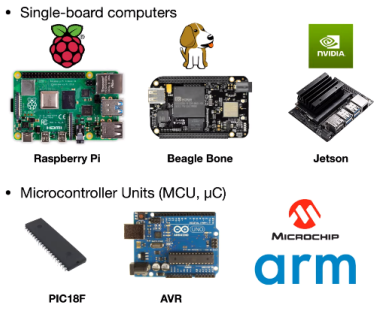
Piotr Chmura

30 września 2021

1 Wstęp

W poniższym artykule znajdują się tłumaczenia filmów oraz zdjęć użytych przez dr.Eddin w celu wytłumaczenia istoty programowania mikrokontrolerów. Link do oficjalnego video w języku angielskim zostanie zamieszczony w bibliografii.

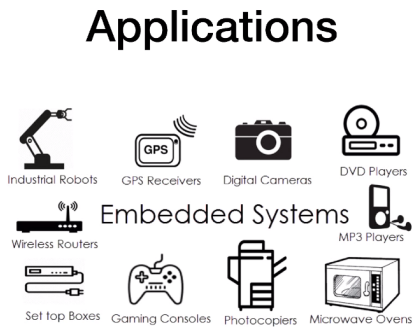
2 Płytki prototypowe.



Rysunek 1: Płytki i mikrokontroler

Podczas prób realizacji projektów zadajemy sobie pytanie czy powinniśmy używać Arduino, Raspberry Pi czy mikrokontrolerów. Postaram się wyjaśnić jakie różnice istnieją pomiędzy nimi. Przed dojściem do detali związanych z pytaniem o istotę używania mikrokontrolerów, należy wyjaśnić ten podział. Techniczne rzecz ujmując wszystkie urządzania znajdujące się na rysunku poniżej to urządzenia obliczeniowe będące w stanie wykonywać wiele innych zadań. Niektóre z nich mają większą moc obliczeniową inne natomiast są szybsze. Raspberry Pi oraz reszta z poniższych nazywana jest często komputerami jednopłytkowymi, której części to między innymi procesor, różne rodzaje pamięci jak i piny wejścia/wyjścia wmontowane w płytkę PCB. Zostały one stworzone ponieważ zaistniał popyt na digitalizację w społecznościach biedniejszych oraz słabiej rozwiniętych, jak i na prostotę w możliwościach nauczania. Jest to cały komputer zamykający się w cenie 30 dolarów. Podłączenie wyjść do wyświetlacza jak i uruchomienie systemu operacyjnego Linux nie sprawia najmniejszych problemów tego typu płytkom. Raspberry Pi zawdzięcza swoją popularność rzeszy młodych elektroników używających tej płytki od wczesnego czasu jej powstania. Do popularnych płytek należy Beagle Bone oraz Nvidia Jetson.

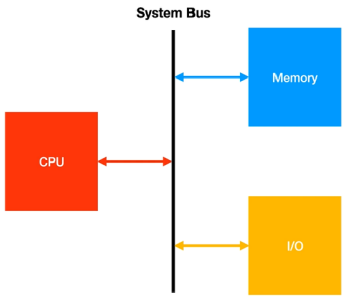
3 Mikrokontrolery



Rysunek 2: Zastosowanie mikrokontrolerów

Istnieje kilka głównych firm zajmujących się produkcją mikrokontrolerów. Dwa najbardziej popularne to PIC oraz AVR (Arduino posiada mikrokontroler AVR), wcześniej AVR należało do Atmel, lecz zostało przejęte przez Microchip. W tym kursie używać będziemy PIC18F4. Mikrokontrolery są dedykowane do pojedynczych zadań w przeciwieństwie do płytek prototypowych. Pobierają one mniej mocy z racji braku konieczności posiadania systemu operacyjnego, odznaczają się szybszą kompilacją oraz uruchomieniem ponieważ nie musimy czekać na uruchomienie całego systemu operacyjnego, przez co programy mogą kompilować się znacznie szybciej.

3.1 Podstawowe komponenty mikrokontrolerów



Rysunek 3: Podstawowy podział

Wyróżnia się kilka podstawowych składowych takich jak procesor, piny wyjścia/wejścia oraz pamięć, podłączone do szyny systemowej, będącej fizycznymi ścieżkami wypalonymi w krzemie (Verilog)do której podłączone są wszystkie z wyżej wymienionych części. By zostać lepszym programistą mikrokontrolerów należy zapoznać się z budową wewnętrzną mikrokontrolera a co za tym idzie każdego z jego podstawowych komponentów.

3.2 Urządzenie peryferyjne oraz I/O

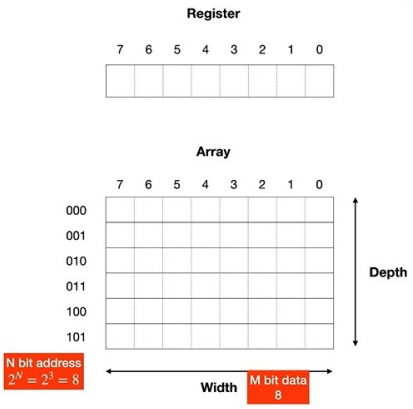
Zazwyczaj mikrokontrolery w swej strukturze posiadają w jednym układzie scalonym wbudowane każde z ww. struktur, mając do czynienia z płytką rozwojową, każdy z bloków może być rozmieszczony w różnym miejscu płytki. PIC18F posiada liczniki, konwertery analogowo cyfrowe, komparatory, piny GPIO, oraz wyjścia **seryjne** IO do komunikacji szynami I2C,SPI oraz UART. Pozwalają one

na podłączenie różnorakich sensorów używając SIO lub GPIO. W późniejszej części kursu zagłębimy się w detale sterowania.

3.3 Pamięć

Osobom zaznajomiony z działaniem przerzutników oraz rejestrów będzie łatwiej zrozumieć zawarte informacje w owej podsekcji. W uproszczeniu rejestr składa się z kilku przerzutników mogących składować logiczne zero lub jeden w celu reprezentacji binarnej liczby. Mogą one wykonywać podstawowe działania na zawartej w sobie liczbie takie jak przesuwanie w prawo, w lewo oraz operacje dodawania tzw. liczniki. Jednak pamięć to coś więcej niż rejestry.To jak składowane/wpisywane są bity do rejestru pozwala na dokonanie dalszych podziałów pamięci, lecz niezależnie gdy mówimy o pamięci powinniśmy mieć przed sobą obraz tablicy dwuwymiarowej. Każdy z wierszy posiada swoją szerokość. Zakładając iż pierwszy wiersz posiada długość 8 bitów, możemy powiedzieć iż mamy do czynienia ze **słowem** wpisanym do komórki pamięci. Ilość wierszy w naszej pamięci 2D możemy nazwać głębokością. To co powinniśmy z tego zapamiętać jest to że pamięć to rejestr 2D mogący składować różne wiersze mogące posiadać różną długość. Różne mikrokontrolery posiadać będą różne rozmiary owej pamięci 2D.

3.4 Uzyskanie dostępu do jednego z bitów słowa

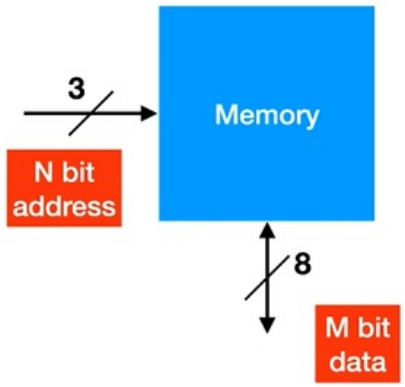


Rysunek 4: Podstawowy podział

Załóżmy iż musimy się dostać do konkretnego bitu naszej pamięci niech będzie to 2 wiersz i 4 kolumna. Należy aktywować wiersz, a następnie odczytać wartość kolumny danego wiersza. **Co mamy na myśli mówiąc aktywować wiersz?** Chcąc wskazać pamięci w które dokładnie miejsce chcemy wpisać bit lub go odczytać należy posłużyć się adresowaniem. Analizując rysunek 4 zauważyć możemy iż nasza hipotetyczna pamięć adresowana jest z 3 bitów, co daje na możliwość obliczenia możliwości adresowych ze wzoru: $2^N = 2^3 = 8$ Co mówi nam iż możemy skorzystać z 8 wierszy pamiętając iż każde słowo zawarte w nich ma 8 bitów [b] - 1Bajt [B]. Tym co chciałbym byście zrozumieli z tego podrozdziału jest to że kiedykolwiek zostanie przytoczone cokolwiek na temat pamięci, w uproszczeniu tyczy się to pamięci 2D posiadających adresowane wiersze wraz z M bitami danych do odczytu/zapisu. Przykładowo załóżmy iż chcemy użyć 4 wiersza, co wymaga

podania odpowiedniej wartości binarnej na wejście adresowe. Poprzez podanie tej wartości będziemy w stanie zapisać lub odczytać wartość binarną przy pomocy komend.

3.5 Kalkulacje

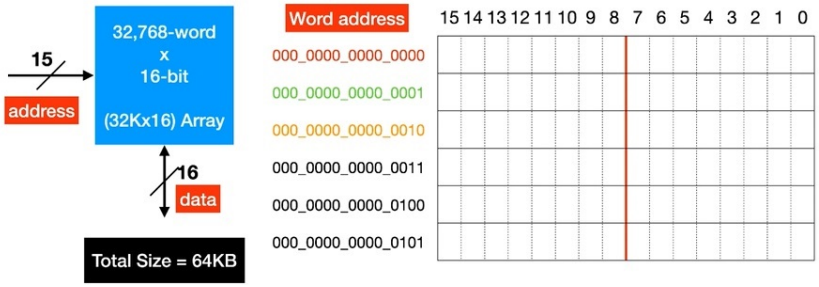


Rysunek 5: Uproszczony wygląd pamięci

Przykładowa kalkulacja przeprowadzona dla 3 bitowej szyny adresowej o 8 bitach rejestru. Mamy do czynienia z 8 słowami po 8 bitów każdy.

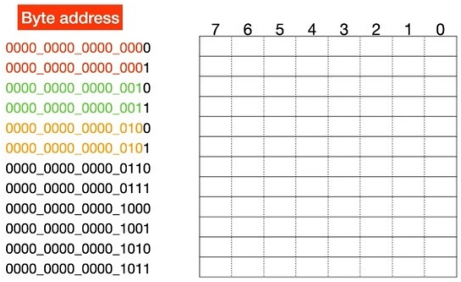
$2^3 \cdot 8 = 64_b$
$\frac{64_b}{8} = 8_B$
Pamiętać należy iż powyższe obliczenia należy mnożyć przez 2 gdy mamy do czynienia z szyną danych 16 bitową, przez 3 gdy 32 bitową etc. Dalsze kalkulacje zostaną przeprowadzone dla pamięci mikrokontrolera PIC18F posiadającego 15 bitów adresowych i 16 bitów danych - słowo.

Dla pamięci słowa każdy kolejny o jeden wiekszy to kolejne 16 bitów, co za tym idzie, w rejestrze możemy składować 2 bajty[B]. Procesor nie postrzega składowanej pamięci w kontekście słowa bitowego. Rozdzielamy 16 bitów na 1 bajt starszy i młodszy, nazwijmy je LSB (Least-Significant-Byte) dla bitów [7:0] Bajt najmniejszej wagi oraz dla bitów [15:8] MSB (Most-Significant-Byte) oznaczający Bajt największej wagi.



Rysunek 6: Pamięć PIC18

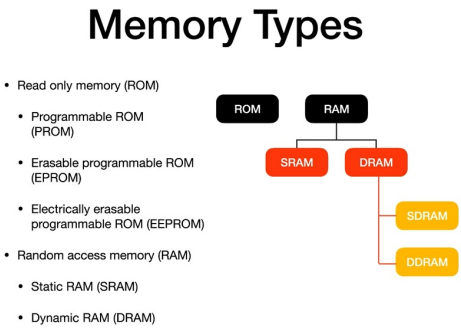
3.6 Bajtowa organizacja pamięci



Rysunek 7: Pamięć Bajtu

W celu rozróżnienia pomiędzy pamięcią słowa oraz pamięcią bajtu, procesor dokonuje podziału słowa z 16 bitów na dwa słowa 8 bitowe, przy czym wartościom MSB przypisuje binarne 1 na końcu adresu bajtowego, natomiast dla LSB przypisuje 0. Konsekwencją tego podziału jest dodanie jednego nadmiarowego bitu. Dodatkowy bit dokonuje zmiany w szynie adresowej z 15 do 16 bitów. Gdyby nasze słowo było długości 32 bitów dodatkowo w celu uzyskania pamięci bajtu dodalibyśmy 2 bity. Należy pamiętać o owym rozróżnieniu. Jest to tzw. wyrównana pamięć, korzystająca z kolejności bajtów.

3.7 Typy pamięci

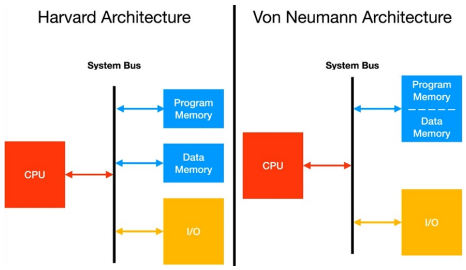


Rysunek 8: Typy pamięci

Zapewne większość z was słyszała o istnieniu wielu różnych rodzajach pamięci. Najbardziej ogólnym podziałem jest rozróżnienie mające na uwadze składowanie bitu. Z perspektywy użytkownika posiadamy kilka kategorii pamięci. **Pamięć ROM** (Read Only Memory), jest to pamięć z której można jedynie odczytywać dane bez możliwości modyfikacji. Był to układ scalony którego wysyłało się do fabryki w celu zapisu danych w układzie. Inżynierzy fabryk podjeli decyzję o wprowadzeniu odmiennego podejścia do programowania pamięci, co poskutkowało wysłaniem chipu z **pamięcią PROM** do kupca, który posiadając odpowiednie narzędzie mógł programować ową pamięć. Z racji występowania błędów, oraz chęci usuwania zawartości, stworzono nowy typ pamięci - **pamięć EPROM**. (Erasable Programmable ROM). Sposobem usuwania danych było naświetlanie przez szkiełko w układzie. By usunąć zawartość poddawało się chip promieniowaniu UV przez około 30 minut, co nie było dość efektywne. Dzisiaj używa się **pamięci EEPROM** (Electrically Erasable Programmable ROM), co pozwala na programowanie oraz usuwanie zawartości, co zmienia użycie z samego zapisu w szerszy nurt możliwości. Dojdziemy do tego dlaczego używamy ciągle nazewnictwa ROM. W pamięci ROM zapisujemy program który chcemy by wykonywał nasz mikrokontroler. Jest to **pamięć nieulotna**, która po odłą-

czeniu zasilania dalej istnieje.
Drugim typem pamięci jest **pamięć RAM**. Jest to **pamięć ulotna**.Zazwyczaj RAM jest stosowany do składowania danych pośrednich. Załóżmy iż mamy kilka liczb które chcę pomnożyć przez jakąś wartość - więc zostaną one przechowane w pamięci RAM. Istnieje dużo typów pamięci RAM, w zależności od tego jak w fizyczny sposób składujemy dane. Wyróżniamy **pamięć SRAM** (Static RAM) oraz **pamięć DRAM** (Dynamic RAM). W przypadku DRAM bit składowany jest jako ładunek kondensatora, co w celu odczytania bitu narzuca konieczność rozładowania kondensatora oraz ponowne jego naładowanie w przypadku chęci zatrzymania informacji. Częścią pamięci DRAM jest **pamięć SDRAM** oraz **DDRAM**. Obecnie najczęściej używa się DDRAM, operująca w częstotliwości 1066MHz. Najefektywniejsze w DDRAM jest składowanie ładunku w kondensatorze pozwalające na rozmieszczenie części stykowych, umożliwiając stworzenie lepszej struktury płytki PCB pobierając bardzo małą moc. Znaczenie zwrotu Random Access - dostęp swobodny - oznacza możliwość niesekwencyjnego dostępu do danych przez szynę adresową.

3.8 Architektura Hardware'u



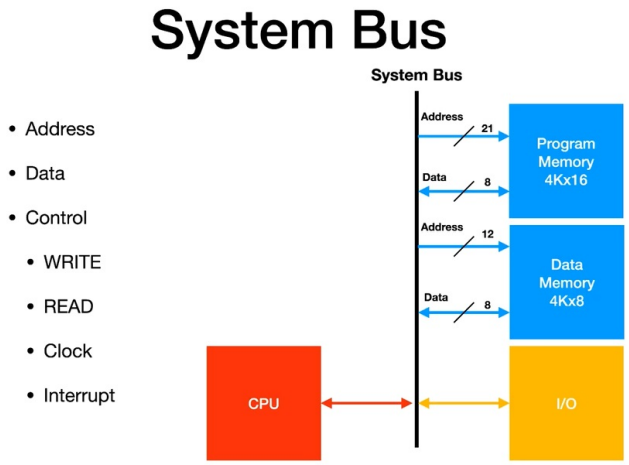
Rysunek 9: Architektury

Z perspektywy mikrokontrolera interesuje nas podział na pamięć ulotną (pamięć danych) oraz nieulotną (pamięć programu). Architektura von Neumann'a posiada pamięć programu oraz danych w jednej jednostce pamięci, co za tym idzie, łączonej możliwości adresacji. Większość mikrokontrolerów używa architektury Harwardzkiej. Rozpatrując architekturę Harwardzką, która wprowadziła rozdział między pamięcią programu i danych, co za tym idzie różne szyny adresowe oraz szyny danych. Pozwala to na CPU na czytanie programu z pamięci oraz wpisywanie lub odczytywanie pamięci danych wpływając na większą wydajność obliczeniową. PIC18F posiadają architekturę Harwardzką.

3.9 Architektura PIC18F

Na rysunku 10 zauważyć możemy że długość słowa to 16 bitów. W przypadku pamięci danych długość słowa to 8 bitów. Pamięć danych w owej wersji mikrokontrolera posiada 512B a pamięć programu 4 tysiące wierszy. Zauważyć można że mając 4 tysiące wierszy pamięci programu szyna danych posiada 21 bitów do **adresacji bajtowej**, służącej do ewentualnego zwiększenia pamięci programu. Należy również zwrócić uwagę, iż dane adresowe są jednokierunkowe, a linie danych są dwukierunkowe. Dlaczego używamy nazewnictwa szyna systemowa?

O szynie systemowej myśleć możemy jako o dużej ilości ścieżek wypalonych w krzemie łączących wszystkie komponenty mikrokontrolera. Pamięć musi się kontaktować z CPU. CPU musi kontaktować się z wyjściami I/O. A wszystko to za pomocą szyny systemowej. Ścieżki szyny systemowej mogą zostać kategoryzowane trójnasób:
Ścieżki Adresacji
Ścieżki Danych
Ścieżki Kontrolne
W przypadku pamięci programu używa się tzw. bitu akcesji który będąc w stanie wysokim wskazuje na wpis, przeciwnie na odczyt. Wspomnieć należy również o wewnętrznym zegarze jak i o możliwości generowania przerwań.



Rysunek 10: Szyna systemowa