W ostatnim odcinku cyklu przedstawimy dalsze informacje o budowie i działaniu mikroprocesora. Oczywiście artykuł ten nie kończy tematu mikroprocesorów na łamach EdW. To jest dopiero szkic, ogólny zarys arcyciekawego zagadnienia. Przygotowujemy praktyczny kurs programowania jednego z popularnych mikrokontrolerów jednoukładowych. Obejmie on nie tyle omówienie właściwości i rozkazów mikroprocesora - dotyczyć będzie przede wszystkim praktycznego wykorzystania tych układów. Zainteresowanych Czytelników zaopatrzymy w płytki, mikroprocesory i niezbędne oprogramowanie. Cykl ten rozpocznie się za kilka miesięcy.

Adresy i adresowanie

Pojęcie adresu kojarzy się z listem, kopertą i listonoszem. Dokładnie adresujemy list, ponieważ nie znamy innego sposobu wyróżnienia odbiorcy, tak aby niezawodnie otrzymał przeznaczone dlań wieści. Nie wystarczy napisać na kopercie: "Dla sympatycznego pana Henia" - trzeba podać nazwę miejscowości, ulicę, numer domu, ewentualny numer mieszkania.

Rodzajem adresu jest też numer telefonu, czyli *liczba*, którą trzeba wybrać za pomocą tarczy aparatu telefonicznego.

Co ważne, adres w postaci liczby jest zrozumiały dla mikroprocesora! Dla niego adres jest liczbą określającą jednoznacznie miejsce w pamięci - poszczególne komórki pamięci są więc ponumerowane. Pamięć ma zwykle organizację bajtową, to znaczy że pod jednym adresem zapisuje się lub odczytuje jednocześnie osiem bitów.

Rejestry

Rejestry są szczególnymi fragmentami (komórkami) wewnętrznej pamięci zapisywalnej mikroprocesora. Oprócz "centralnego rejestru", nazywanego akumulatorem, współpracującego z jednostką arytmetyczno-logiczną, istnieje szereg rejestrów pomocniczych. Rejestry tym różnią się od "zwykłych" komórek pamięci, że można ich zawartość przetwarzać w różny sposób, a nie tylko zapisywać i odczytywać. Na zawartości rejestrów zazwyczaj mogą być wykonywane operacje arytmetyczne i logiczne.

Licznik rozkazów

Każdy mikroprocesor posiada licznik rozkazów PC (*Program Counter*). Licznik rozkazów jest nie tyle licznikiem liczącym "w górę", ale raczej rejestrem, w którym jest pamiętany bieżący adres wykonywanej instrukcji. Jego długość odpowiada długości szyny adresowej pamięci programu. W zasadzie po każdym cyklu rozkazowym licznik zwiększa swą zawartość o 1, czyli adresuje następną komórkę pamięci. W ten sposób program zapisany w pamięci jest realizowany kolejno krok po kroku. Istnieją jednak rozkazy, które zmieniają zawartość licznika rozkazów, tym samym powodują, że wykonywany jest fragment programu (podprogram lub procedura obsługi przerwania) umieszczony w jakimś odległym zakątku pamięci programu.

Stos

Stos jest wydzielonym fragmentem pamięci RAM systemu mikroprocesorowego. Nazwa ta dokładnie odzwierciedla

Kody liczbowe

Kody liczbowe to sposób zapisu informacji liczbowej. Jeśli informacja przetwarzana zawiera symbole literowe i znaki specjalne to taki kod nazwiemy kodem alfanumerycznym. Inne kody, które służą do wykrywania błędów i ewentualnego ich usuwania z informacji przesyłanej na pewną odległość (np. łączem telefonicznym) nazywają się kodami korekcyjnymi.

Możemy więc powiedzieć, że kody, jakie przyjdzie nam spotykać w praktyce mikroprocesorowej są zapisem informacji według pewnego, ściśle określonego przepisu.

Najbardziej rozpowszechnionymi kodami liczbowymi są kody naturalne. Znanym wszystkim od przedszkola jest kod dziesiętny. Cały świat powszechnie go używa. Nie ma w nim nic ciekawego, ot, dziesięć cyfr i już. Jednak ten kod ma pewne cechy, które są wspólne dla wszystkich naturalnych kodów liczbowych. Po pierwsze, liczba zapisana w kodzie naturalnym jest ciągiem cyfr, czyli pozycji, z którego każda cyfra reprezentuje wielokrotność tzw. wagi danej pozycji. W kodzie dziesiętnym waga jest zawsze potęgą liczby 10, czyli patrząc od prawej strony liczby jest $10^0=1$, potem $10^1=10$, następnie $10^2=100$ itd. Nikt z nas jednak nie zastanawia się nad tym.

Każdy kod naturalny ma swoją podstawę. Podstawą jest podstawa potęgi wagi. W kodzie dziesiętnym podstawa wynosi 10.

sposób dostępu do informacji. Przypomina on stos talerzy. Ażeby zdjąć talerz, np. czwarty od góry, bez rozrzucenia całego stosu, trzeba zdjąć wszystkie na nim leżące - w naszym przykładzie trzy. Każde dołożenie do stosu jeszcze jednego talerza powoduje, że dostęp do pierwszych z nich jest coraz bardziej utrudniony. Stos jest realizacją kolejki danych typu LIFO (*Last In First Out* - ostatni wszedł, pierwszy wyjdzie). Zapewne zmywanie naczyń nie jest twoją ulubioną czynnością, wiedz jednak, że stos w mikroprocesorze jest naprawdę bardzo użyteczny.

Na stos są zapisywane adresy powrotu z podprogramów oraz mogą być przechowywane parametry dla tychże podprogramów. Mikroprocesor pamięta tylko adres ostatnio zapisanej komórki pamięci, czyli adres *wierzchołka stosu.* Adres wierzchołka stosu jest przechowywany w specjalnym rejestrze, zwanym *wskaźnikiem stosu*, w skrócie oznaczanym SP (*stack pointer*).

Wskaźniki procesora

Wskaźniki procesora stanowią pojedyncze bity pamięci, przechowywane w wydzielonym rejestrze zwanym słowem stanu PSW (*Program Status Word).* Najczęściej spotykane wskaźniki:

- wskaźnik przeniesienia/pożyczki CY (*Carry)*, pamiętający stan przeniesienia/pożyczki po wykonaniu operacji arytmetycznych,
- wskaźnik przepełnienia OV (*Overload*), pamiętający skutki ewentualnego przekroczenia zakresu dla wybranych operacji, np. dzielenia przez zero,
- wskaźnik zerowania, w którym jest pamiętana informacja o zerowym wyniku ostatniej operacji arytmetycznej bądź logicznej,
- wskaźnik parzystości, w którym jest informacja, że w ostatnim wyniku operacji arytmetycznej czy logicznej przeprowadzanej przez jednostkę centralną zawarta była parzysta liczba jedynek.

Oprócz powyższych wskaźników, występujących niemal we wszystkich mikroprocesorach, pojawiają się inne, charakterystyczne dla konkretnego typu.

Przerwania

Jednym z wielu elementów w czasie podróży po morzu, jakie musi brać pod uwagę kapitan statku, są komunikaty meteo. Gdy kurs przecina strefę burz, trzeba podjąć decyzję: wpłynąć w nią czy ją ominąć. Co ważne, na początku rejsu nie można przewidzieć, czy i ewentualnie kiedy, statek taką burzę napotka.

Mikroprocesor jest maszyną synchroniczną, która pracuje w pewnym otoczeniu, jakim są urządzenia zewnętrzne, np. klawiatura, wyświetlacz, układ grafiki, urządzenia pamięci masowej (dyski, streamery), interfejsy itp. Wiele z nich co pewien czas wymaga interwencji mikroprocesora, czy to w celu odebrania danych, czy też ich przekazania, czasem wysterowania wybranych linii itp. Ponieważ czas pomiędzy zgłoszeniami takich potrzeb jest nieokreślony, możemy uznać, że przychodzą one asynchronicznie, czyli niezależnie od aktualnie wykonywanych rozkazów, a tym bardziej od impulsów zegarowych. Asynchroniczne zdarzenie wymagające reakcji mikroprocesora nazywamy *przerwaniem.*

Mikroprocesor w każdym cyklu maszynowym sprawdza stan wskaźników przerwań lub specjalnych linii przerwań. Jeśli wykryje żądanie obsługi przerwania, następuje zawieszenie wykonywania programu głównego i przejście do procedury obsługi przerwania.

Mikroprocesor posiada zwykle cały system przerwań, który może przyjmować przerwania z kilku źródeł. Istnieje więc potrzeba arbitrażu obsługi przerwań, czyli zadecydowania, które z przerwań jakie nadeszły jednocześnie, powinno być obsłużone w pierwszej kolejności.

W celu ich rozróżnienia narzucane są priorytety jednych przerwań nad drugimi, polegające na tym, że przerwanie o priorytecie wyższym może przerwać obsługę przerwania o priorytecie niższym, ale nigdy odwrotnie, a w przypadku wykrycia kilku przerwań jednoczesnych, będzie obsługiwane to z nich, które ma najwyższy priorytet.

Wprowadzane są też znaczniki programowego zezwolenia na obsługę przerwania, czyli po zablokowaniu obsługi przez taki znacznik, mikroprocesor nie będzie reagował na to przerwanie i takie przerwania nazwiemy *maskowalnymi.* Jeśli są przerwania maskowalne, zatem muszą istnieć *przerwania niemaskowalne*, które posiadają najwyższy priorytet i służą przede wszystkim do reakcji na zdarzenia katastroficzne, np. wyłączenie zasilania, awaryjne zatrzymanie działania sterowanego urządzenia itp. Bywa również tak, że system priorytetów jest budowany programowo, a więc jeden podprogram obsługi jest wspólny dla wszystkich źródeł i w samej procedurze zawarto kolejność reakcji. O tym, jak pracuje układ przerwań decyduje konstruktor mikroprocesora i my, użytkownicy, nie mamy na to wpływu.

Interfejs we/wy

Interfejs we/wy jest przeznaczony do komunikacji mikroprocesora ze światem zewnętrznym. Są to "usta i uszy" mikroprocesora, nazywane wrotami albo portami. Do iterfejsu są podłączone "oczy i ręce" - czyli wszelkiej maści urządzenia zewnętrzne, takie bez których mikroprocesor w zasadzie może się obejść, ale z którymi musi współpracować, bo po to został włożony do urządzenia.

Fizycznie interfejs zawiera pewną ilość linii sygnałowych, podzielonych na kilka grup, zwanych *szynami.* Mamy więc szyny danych, szyny adresowe, szyny linii współpracy i inne. O ich strukturze i szerokości (ilości linii składowych) decyduje obszar zastosowań samego mikroprocesora.

Szyna danych służy do przesyłania danych do/z mikroprocesora.

Na szynę adresową mikroprocesora wystawiany jest adres potrzebny dekoderowi adresów do uaktywnienia właściwego urządzenia zewnętrznego (np. pamięci zewnętrznej).

Szyna linii współpracy zawiera w sobie pojedyncze linie, po których przesyłane są sygnały informujące o stanie urządzeń zewnętrznych i o stanie mikroprocesora. Może być to linia informująca o wystawieniu danych na szynę danych i adresu na szynę adresową w celu przesłania ich do pamięci zewnętrznej, linia potwierdzenia odebrania danych, linia żądania przerwania oraz komplementarna do niej, linia potwierdzenia odebrania przerwania itp. Do szyny linii współpracy zaliczono też linie przerwań, chyba słusznie, bowiem one również wnoszą jakąś jednobitową informację.

Mikroprocesor jednoukładowy ma często kilka portów, których poszczególne linie mogą pełnić różne funkcje określone przez program.

Rozkazy

Rozkazy to polecenia. Mikroprocesor nie rozumie poleceń wydawanych ludzkim głosem. Rozumie za to liczby. *Rozkaz jest więc liczbą.* Rozkazy są kolejno odczytywane z pamięci i przesyłane do dekodera rozkazów, który jak wiemy powoduje wykonanie w efekcie szeregu elementarnych mikrooperacji.

Pamięć programu większości procesorów ma organizację bajtową, co oznacza, że wystawienie adresu przez licznik rozkazów powoduje wczytanie z pamięci programu 8-bitowego kodu rozkazu. Daje to $2^8 = 256$ różnych rozkazów - wydawałoby się, że to wystarczy.

Tymczasem wiele rozkazów nawet prostego ośmiobitowego mikroprocesora powinno być zbudowanych z kilku części: części operacyjnej, która określa czynność (przesłanie, dodawanie itp.) oraz części argumentowej. Argumentem może być adres, stała (liczba), kod rejestru itp.

W rzeczywistości 8 bitów to zbyt mało, aby w nich zmieścić część operacyjną i jeszcze mieć miejsce na argumenty. Wymyślono więc, że *długość rozkazu będzie zmienna*, równa wielokrotności bajtu.

Tak więc w pamięci programu oprócz liczb przedstawiających kody rozkazów

zawarte są także liczby reprezentujące dane. Ta sama liczba dwójkowa 00000100 zawarta w pamięci programu może oznaczać "zwiększ zawartość akumulatora o 1" lub też po prostu liczbę 4. O interpretacji, czyli liczbie bajtów do odczytania decyduje część operacyjna, czyli pierwszy bajt rozkazu.

Czas wykonania rozkazów jest mierzony w cyklach procesora. Są takie mikroprocesory, które wszystkie swoje rozkazy realizują w jednym cyklu maszynowym i jest to cechą procesorów o zredukowanej liście rozkazów RISC (*Reduced Instruction Set Computer*). Mikroprocesory wykonujące rozkazy w czasie kilku cykli to przedstawiciele grupy procesorów CISC (*Complete Instruction Set Computer*).

Trudno nam będzie opisać szczegółowe właściwości rozkazów mikroprocesora, jeśli nie mamy na myśli konkretnego typu. Można jednak znaleźć cechy wspólne i określić je dla pewnych grup rozkazów.

Rozkazy przesłań to wszelkiego rodzaju rozkazy służące do powielania danych w różnych miejscach pamięci systemu mikroprocesorowego. Lista tych rozkazów może być bardzo bogata i umożliwia ona przesłanie informacji w dowolne miejsce obszaru adresowego mikroprocesora z wykorzystaniem różnych sposobów adresowania. Rozkazy przesłań są zazwyczaj dwuargumentowe, z których jednym argumentem jest miejsce (adres) źródła danych, a drugim miejsce (adres) jego przeznaczenia, niekoniecznie w takiej kolejności. Mikroprocesor może przesyłać dane jednobitowe, jednobajtowe i wielobajtowe.

Rozkazy operacji arytmetycznych i logicznych - ich liczba jest zależna od możliwości (można powiedzieć - "inteligencji") samej jednostki arytmetyczno-logicznej. Spotykamy następujące rozkazy:

- dodawania, z przeniesieniem lub bez niego;
- odejmowania, z pożyczką lub bez niej;
- inkrementacji (zwiększenia o 1);
- dekrementacji (zmniejszenia o 1);
- porównania;
- mnożenia;
- dzielenia;
- korekcji dziesiętnej w celu uzyskania wyniku w kodzie BCD;
- iloczynu logicznego;
- sumy logicznej;
- wyłącznej sumy logicznej;
- przesunięcia bitowego, w prawo i w lewo.

Rozkazy sterujące wykonaniem programu (skoki) przenoszą wykonanie programu w jego inne miejsce. Ich działanie polega na odpowiedniej modyfikacji licznika rozkazów w taki sposób, ażeby po wykonaniu skoku, następnym rozkazem był rozkaz odczytany z innego miejsca pamięci programu. Skoki dzielimy na bezwarunkowe, warunkowe, skoki do podprogramów i powroty z podprogramów. Ze względu na zasięg skoku mogą to być skoki krótkie, długie i względne.

Skoki mogą być wykonane w ramach pewnego, wydzielonego obszaru całej przestrzeni adresowej i takie skoki nazwiemy skokami w ramach strony albo *skokami krótkimi*. Praktyka programowania dowodzi, że wiele skoków jest wykonywanych do pobliskich adresów, nie ma więc istotnej potrzeby wprowadzania do kodu rozkazu pełnego adresu, który może mieć np. 32 bity, jeśli mamy pewność, że zmieni się tylko mniej niż np. 10 najmłodszych bitów. Oszczędzamy w ten sposób na długości programu. Oczywiście muszą istnieć *skoki długie*, adresujące w swym kodzie całą dostępną przestrzeń programu.

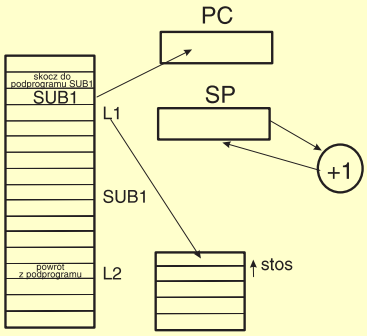
Innym sposobem skrócenia długości kodu skoku jest *skok względny*. Metoda zmiany zawartości licznika rozkazów polega na dodaniu do licznika programu pewnego przesunięcia zawartego w części argumentowej. Przesunięcie to może być jednobajtowe, rozumiane jako liczba zapisana w kodzie uzupełnieniowym do dwóch. Oznacza to, że skok odbywa się w granicach od +127 (w przód) do -128 bajtów (wstecz).

Skoki bezwarunkowe, jak sama ich nazwa wskazuje, są wykonywane niezależnie od stanu mikroprocesora.

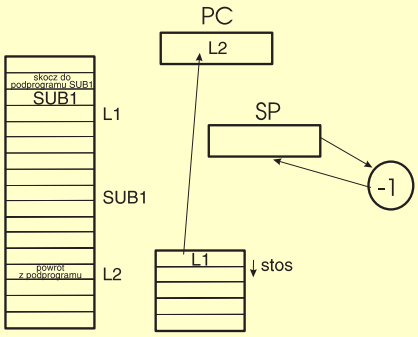
Skoki warunkowe są wykonywane, jeśli zostanie spełniony warunek stanu określonego znacznika. Może to być stan bitu albo bajtu. Niespełnienie warunku powoduje przejście do rozkazu znajdującego się bezpośrednio po rozkazie skoku warunkowego. W kodzie rozkazu może być umieszczony adres jako jeden argument operacji, drugim argumentem operacji jest adres bitu czy bajtu warunkującego. Innym rozwiązaniem skoku warunkowego jest *skok z ominięciem*. W kodzie instrukcji nie ma adresu skoku, natomiast spełnienie warunku powoduje, że następna instrukcja pozostanie zignorowana i procesor wykona kolejną instrukcję programu. Omijana instrukcja jest z reguły skokiem bezwarunkowym.

Skok do podprogramu różni się nieco od wyżej opisanego skoku. Oprócz przeniesienia wykonania programu w inne miejsce musi on zapamiętać adres powrotu do instrukcji następującej po nim. Adres ten jest pamiętany na stosie.

Z chwilą zdekodowania instrukcji skoku do podprogramu, wyznaczany jest adres następnej instrukcji i wysyłany jest na stos. Wskaźnik stosu jest modyfikowany w taki sposób, ażeby wskazywał adres wierzchołka stosu, czyli komórki pamięci, w których zapisano *adres powrotu z podprogramu* (**rysunek 10**).



Skok do podprogramu



Powrót z podprogramu

Rys. 10.

Instrukcja powrotu z podprogramu działa odwrotnie niż skoku do podprogramu. Zdejmuje ona z wierzchołka stosu adres powrotu z podprogramu i zapisuje go do licznika rozkazu, jednocześnie modyfikując wskaźnik stosu, który będzie wskazywał na ewentualny adres powrotu z innego, wcześniej wywołanego podprogramu. Nieco inaczej działa instrukcja powrotu z podprogramu obsługi przerwania. W odróżnieniu od instrukcji powrotu ze zwykłego podprogramu, informuje ona system przerwań o zakończeniu obsługi przerwania. Ma to znaczenie dla innych przerwań, które mogą "starać" się o "dostęp" do procesora.

Instrukcje skoku do podprogramu i powrotu z nich mogą być wykonywane warunkowo na zasadach przedstawionych wyżej.

Podane informacje mogą, choć nie powinny nikogo stresować - programista naprawdę nie musi znać szczegółów budowy mikroprocesora. Powinien orientować się w jego architekturze, znać listę rozkazów i przede wszystkim rozumieć sposób działania.

Niech przedstawiony artykuł ośmieli Czytelników do bliższego zapoznania się z mikroprocesorami i do stosowania ich w swoich konstrukcjach!

Mirosław Lach