Basic dla mikrokontrolerów MCS-51, część 1

Jest to pierwsza część artykułu, w którym pokrótce omówimy możliwości nowoczesnego interpretera Basica dla mikrokontrolerów rodziny MCS-51. Pomimo opinii, że Basic jest najgorszym z możliwych języków programowania, nam się wydaje, że początkujący mikroprocesorowcy mogą śmiało z niego korzystać - jego prostota i stosunkowo duże możliwości niwelują wszelkie niedoskonałości. No to RUN...

Możliwość wykorzystania we własnych projektach mikrokomputerów jednoukładowych jest dla większości konstruktorów łakomym kąskiem. Nawet bardzo złożone funkcjonalnie układy realizuje się przy ich pomocy w bardzo prosty sposób. Niewielkie płytki drukowane, niższy pobór prądu, łatwość wprowadzania zmian i modyfikacji, które zawsze przecież towarzyszą prototypom, to argumenty, wobec których nikt nie może być obojętny.

Ci z Czytelników, którzy śledzą projekty zamieszczane np. w Elektronice Praktycznej z pewnością są pod wrażeniem finezji układów z komputerami jednoukładowymi, a pozostali powinni koniecznie zapoznać się na przykład z projektem miernika częstotliwości zamieszczonym w numerze 12/96.

Początkujący konstruktorzy rzadko sięgają po mikroprocesory. Boją się ich złożoności, nie mają narzędzi wspomagających pracę, nie potrafią programować. Nie bez znaczenia jest również bariera psychologiczna - niechętnie sięgamy po podzespoły nieznane lub takie, które mogą sprawić kłopoty.

W pismach wydawnictwa AVT opublikowano szereg projektów, które w założeniach miały ułatwić początkującym start w tej dziedzinie. Przykładowo, kit AVT-222 zawierający procesor 8031 z pamięcią EPROM i układami towarzyszącymi uwalnia nas od kłopotów sprzętowych, gdyż w zasadzie jest to gotowy, przetestowany sterownik. Do tej niewielkiej płytki od razu można podłączyć urządzenia wykonawcze (np. przekaźniki) i uzyskać gotowe urządzenie. Część sprzętowa nie powinna zatem sprawić kłopotu temu, kto umie posługiwać się lutownicą i miernikiem uniwersalnym

Pozostaje oczywiście problem oprogramowania. Niestety tutaj jest o wiele gorzej - konstruktor amator, nie dysponujący dużą gotówką, skazany jest właściwie na programowanie w asemblerze, przy wykorzystaniu sharewa-re'owych wersji kompilatora (np. dostępnego na płycie CD-EP1). Innymi słowy: nie dość, że od razu trafia na najtrudniejszy język programowania, to jeszcze musi mieć komputer PC - platformę do pracy wspomnianego asemblera skrośnego. Samo napisanie i skompilowanie bez błędów programu nie oznacza końca kłopotów.

Prawa Murphy'ego są nieubłagane i zazwyczaj nawet programy kilkudziesięcioliniowe zawierają w swej początkowej formie po kilka błędów. Dużą część z nich można wprawdzie wyłapać na programowych symulatorach procesora (znowu problem z pecetem!), jednak zawsze będą takie błędy, które ujawnią się dopiero po zaprogramowaniu EPROM-u. Symulator programowy jest bowiem zbyt hermetyczny i w zasadzie pozwala przetestować jedynie poprawność algorytmu. Komfortowa praca nad oprogramowaniem wymaga zatem sprzętowego symulatora procesora lub chociażby symulatora pamięci EPROM.

Czy zatem nie ma żadnej prostszej metody? Takiej dla początkującego amatora? Taniej a niezłej? Czy nikt do tej pory nic nie wymyślił? Owszem TAK!

Rozwiązanie to nazywa się Tiny Basic. Jest to prosty interpreter języka Basic przeznaczony dla rodziny jednoukładowych komputerów serii MCS-51 i został napisany przez firmę Intel. Jego zaletą jest niewielki obszar zajmowanej pamięci stałej, np. EPROM (3kB), możliwość pracy bez dodatkowej, zewnętrznej pamięci RAM oraz właśnie to, że jest to interpreter pozwalający wykonywać program instrukcja po instrukcji, z pełnym wglądem programisty do pamięci portów, rejestrów i zmiennych. Jak

Tabela 1. Zestawienie skróconych postaci poleceń i funkcji Tiny Basic.

Polecenia

C.	CALL	D.	DECIMAL	E.	END
F.	FOR	G.	GOTO	GOS.	GOSUB
H.	HEX	I.	IF	IN.	INPUT
L.	LET	LI.	LIST	N.	NEXT
NEW	NEW	P.	PRINT	PRO.	PROM
R.	RETURN	RA.	RAM	RES.	RESET
RO.	ROM	RU.	RUN	REM	REMARK
T.	TO	T.	THEN		

Funkcje i zmienne specjalne

A	ABS	A.	AND	C.	CBYTE
D.	DBYTE	M.	MOD	N.	NOT
O.	OR	R.	RBIT	RN.	RND
X.	XBYTE	X.	XOR		

bardzo ułatwia to uruchamianie programów nie trzeba chyba nikogo przekonywać.

Interpreter pracuje w kilku trybach zależnych od sprzętowej konfiguracji sterownika. Najprostsza konfiguracja wymaga jedynie procesora 8751 (lub 8031 z zewnętrznym EPROM-em) zawierającego kod interpretera, układu MAX 232 konwertującego poziomy napięć interfejsu RS232 procesora i terminala współpracującego z procesorem poprzez ten interfejs. Zasada pracy programu jest bowiem następująca: komputer (terminal) zawierający monitor i klawiaturę jest dla sterownika urządzeniem do wprowadzania i wyświetlania danych. Znaki pisane na klawiaturze są przesyłane do sterownika, a wyniki pracy wyświetlane na jego ekranie.

Terminalem może być dowolny komputer PC, na którym uruchomiono program obsługi portu szeregowego (np. terminal windowsowy, Telix, Procomm, Telemate) lub inne komputery, nawet 8-bitowe C64, Atari XL/XE, Amstrad czy ZX Spectrum, zawierające programowe lub sprzętowe łącze RS232. Tak niskie wymagania spowodowane są tym, że zadaniem terminala jest jedynie wyświetlanie na ekranie znaków wysyłanych przez sterownik i wysyłanie do sterownika znaków z klawiatury. Nie należy myśleć, że podłączony terminal jest zawsze potrzebny (byłoby to bez sensu), wykorzystuje się go jedynie na etapie pisania i uruchamiania programu. Później, po zakończeniu prac, gotowe dzieło umieszcza się w pamięci stałej i terminal można odłączyć.

Dodanie do wersji minimalnej pamięci RAM pozwala na bardziej komfortową pracę z Basiciem (giną ograniczenia, o których będzie mowa później), natomiast dodanie stałej pamięci zewnętrznej lub jej rozbudowanie ponad 4 KB dla wersji z 8031 pozwala zapamiętać nawet duże programy lub dane (tablice stałych). Basic potrafi automatycznie rozpoznać, ile pamięci RAM jest w systemie i skorzystać z rozszerzenia poza 128 bajty zawarte w chipie. Dużym ułatwieniem jest również wbudowana procedura detekcji szybkości pracy łącza RS232. Pozwala to stosować praktycznie dowolne kwarce w sterowniku, uwalniając się od zakupu kryształu o magicznej częstotliwości 11,0592MHz.

Testowanie szybkości RS-a odbywa się podczas startu systemu. Po włączeniu zasilania sterownika, na klawiaturze terminala naciskamy kilkakrotnie spację lub małe „c“, aż do pojawienia się winiety Basica i znaku zachęty interpretera „>“. O tym, czy Basic będzie pracował w trybie interakcyjnym z terminalem, czy też ma automatycznie przejść do wykonywania gotowego programu Basicowego zapisanego w pamięci stałej, decyduje stan logiczny wejścia RxD procesora. Zwarcie tej nóżki do masy, zostanie zrozumiane podczas startu sterownika jako żądanie wykonania gotowego programu. Interpreter pominie również procedurę detekcji szybkości RS-232.

Szybkość pracy interpretera nie jest porywająca, jednak można uznać ją za wystarczającą. Nie ma żadnej przeszkody, aby krytyczne z punktu widzenia szybkości procedury programu zrealizować za pomocą wstawek asemblerowych. Ponieważ są to jednak wstawki, a nie całe procedury, ich pisanie i testowanie jest o wiele przyjemniejsze.

Tiny Basic jest interpreterem wyłącznie stałoprzecinkowym. Wszystkie używane liczby muszą być całkowite i zawierać się w przedziale -32767 do +32768.

Opis języka Tiny Basic

Ponieważ na temat języka Basic napisano wiele książek i artykułów prasowych, bez sensu byłoby na łamach EP opisywać szczegółowo metody i sposoby programowania w Basicu. Dlatego w niniejszym artykule ograniczymy się do skrótowego opisu komend języka odsyłając jednocześnie do literatury tych wszystkich, dla których okaże się on niewystarczający.

Liczby

Tiny Basic jest programem stałoprzecinkowym, wszystkie używane liczby oraz wyniki operacji matematycznych muszą zawierać się w zakresie -32767 do 32767.

Zmienne

Użytkownik ma do dyspozycji 26 zadeklarowanych wstępnie zmiennych, o jednoliterowych nazwach od A do Z. Przy pracy bez zewnętrznej pamięci RAM liczba dostępnych zmiennych jest mniejsza i użytkownik dysponuje tylko dwunastoma w zakresie od A do L. Nie jest ważne czy posługujemy się małymi, czy też dużymi literami, gdyż Basic automatycznie dokonuje konwersji na duże litery.

Funkcje

Dostępne są jedynie dwie funkcje:

ABS (X) - daje moduł wyrażenia X

RND (X) - daje pseudolosową liczbę z zakresu od 1 do X

Operacje matematyczne

Oprócz typowych działań jak:

„+“ - czyli dodawania

„-“ - odejmowania

„*“ - mnożenia

„/“ - dzielenia stałoprzecinkowego (na przykład 16/3=5, 14/5=2, 8/5=1),

Dostępny jest jeszcze operator MOD, za pomocą którego można otrzymać resztę z dzielenia całkowitego (na przykład 16 MOD 3 = 1, 14 MOD 5 = 4, 8 MOD 5 = 3).

Operacje logiczne

Lista dostępnych operatorów logicznych jest typowa. Operują na wszystkich bitach danych:

NOT - negacja

AND - iloczyn (3 AND 6 = 2, 24 AND 8 = 8)

OR - suma (3 OR 6 = 7, 24 OR 8 = 24)

XOR - suma modulo 2 (3 XOR 6 = 5, 4 XOR 7 =3)

Operatory porównania

> - więcej niż
< - mniej niż
= - równość
<> - nierówność
>= - więcej lub tyle samo
<= - mniej lub tyle samo

Wynik operatora porównania jest równy 1 dla prawdy i 0 dla fałszu. Taka sama reprezentacja dotyczy operacji logicznych. Pozwala to uprościć w programie zapis badania warunków.

Wyrażenia

Do budowy wyrażeń można użyć liczb, zmiennych i funkcji, łącząc je za pomocą operatorów. Kolejność obliczania przez program wyrażeń jest następująca: na początku są wykonywane operacje negacji, później mnożenie, dzielenie, MOD i AND, a następnie operacje dodawania, odejmowania, OR, XOR i na końcu operacje porównania. Wartość wyrażenia Basic wylicza od lewej do prawej strony. Powyższe reguły kolejności można zmienić za pomocą nawiasów.

Przykład:

```
10 LET A=(X+2)*Y+321+(X=Y)*3+(X<=Y)
```

da: 328 przy X=1 i Y=1,
338 przy X=2 i Y=4

Komendy

Jeśli wpisywane rozkazy nie zostaną poprzedzone numerem linii, zostaną zinterpretowane jako komendy, czyli polecenia do natychmiastowego wykonania. Wszystkie opisane dalej polecenia języka mogą być użyte jako komendy bezpośrednie. Następujących trzech poleceń wolno jednak używać tylko jako komendy bezpośrednie:

RUN - uruchamia wpisany program;
LIST - wyświetla listing programu od pierwszej linii;
LIST 40 - wyświetla listing programu od linii wskazanej;
NEW - kasuje program, ustawia wartość wszystkich zmiennych na zero.

Skróty

Aby maksymalnie efektywnie korzystać z pamięci RAM sterownika, słowa kluczowe języka mogą być skracane. Tak więc słowo PRINT może zostać zapisane jako „P.“, „PR.“, „PRIN.“. Skrótu dokonuje się za pomocą kropki. Dodatkowo, można pominąć słowa LET i THEN w poleceniu IF. Wpisanie bezpośrednio nazwy zmiennej spowoduje wypisanie jej wartości na ekranie terminala. Tak prosta inspekcja jest bardzo wygodna na etapie uruchamiania programu. Zestawienie najbardziej skróconych komend przedstawiono w **tab.1**. Dzięki skrótom w jednej linii Basica można umieścić kilka instrukcji (rozdzielamy je dwukropkami).

Robert Magdziak, AVT

Interpreter Tiny Basic jest dostępny na płycie CD-EP1, dostępnej w sprzedaży wysyłkowej (kupon zamówienia).