Mikroelektronika w Technice i Medycynie
Luty 2017
Instrukcja do ćwiczeń/kolokwium z mbed_cz2 - wersja tymczasowa

1) Wysyłanie/odbiór znaków

Po podłączeniu zestawu uruchomieniowego do PC powinien pojawić się w systemie dodatkowy port komunikacji szeregowej. W poniższych programach odbioru i wysłania znaków należy użyć funkcji bibliotecznych embed.

Napisać program, który będzie:

a) Co 0,5 sekundy wysyłał kolejny znak ASCII zaczynając od „a”,

b) Zapalał LED-a po odebraniu znaku,

c) Negował LED-a po odebraniu znaku,

d)

- Zapalał LED-a po odebraniu znaku „s”,
- Gasił LED-a po odebraniu znaku „c”,
- Negował LED-a po odebraniu znaku „t” (toggle),

e) Odsyłał wysłane do niego znaki (echo).

Razem 5 programów.

2) Wysyłanie łańcuchów znakowych

a) Specyfikacja

Funkcja „puts” powinna:

- wysyłać łańcuch znakowy zakończony znakiem powrotu karetki, przy czym znak powrotu karetki powinien być wysyłany zamiast znaku NULL znajdującego się na końcu łańcucha do wysłania,
- Wykorzystywać funkcje "putch",
- Przyjmować:
  - wskaźnik na tablice z łańcuchem do wysłania,
  - rozmiar tablicy,
- Kończyć działanie po wysłaniu znaku powrotu karetki,
- zwracać:, „1” jeżeli w tablicy z łańcuchem do wysłania nie występuje NULL, "0" w przeciwnym przypadku, przy czym sprawdzenie obecności NULL-a powinno nastąpić przed wysłaniem pierwszego znaku.

b) Testy

i) Napisać program, który będzie wysyłał, co sekundę łańcuch "hardbeat\n",

ii) Zmodyfikować poprzedni program tak aby wysyłał co sekundę "hardbeat number: {tu wstawić kolejny numer}\n"**.** Do konwersji liczby na odpowiadający jej łańcuch znakowy użyć standardowej funkcji "C",

iii) Napisać test, który sprawdzi wykrywanie braku NULL-a w łańcuchu do wysłania. Do sygnalizacji użyć LED-a.

3) Odbiór łańcuchów znakowych

a) Specyfikacja

Funkcja "puts" powinna:

- Odbierać łańcuch znakowy zakończony znakiem powrotu karetki, przy czym znak powrotu karetki powinien zostać zamieniony na NULL,
- Wykorzystywać funkcje "getch",
- Przyjmować:
  - Wskaźnik na tablice, do której maja być wstawiane znaki odbieranego łańcucha,
  - Rozmiar tablicy,
- Kończyć działanie po odebraniu znaku powrotu karetki,
- Zwracać: "1", jeżeli nastąpiło przepełnienie bufora, "0" w przeciwnym przypadku.

b) Testy

i) Napisać program, który będzie po odebraniu dowolnego łańcucha znakowego zakończonego znakiem powrotu karetki zmieniał stan LED0 na przeciwny.

ii) Napisać program, który będzie po odebraniu:

(1) "on" zaświeci LED,

(2) "off" zgasi LED

(3) "toggle" zmieni stan LED na przeciwny**.** Do porównania łańcuchów znakowych użyć standardowej funkcji "C".

(4) Napisać test, który sprawdzi wykrywanie przepełnienia bufora. Do sygnalizacji przepełnienia bufora użyć LED-a.

4) Loopback "łańcuchowy"

a) Napisać program, który będzie odsyłał wysłane do niego łańcuchy znakowe (echo). Użyć funkcji "gets" i "puts",

b) Zmodyfikować poprzedni program tak aby "podwajał" odsyłany łańcuch oraz zamieniał duże litery na małe. Przykład: "InputString" - > "inputstringinputstring". Do "podwajania" łańcucha użyć standardowej funkcji "C".

5) Sterowanie serwomechanizmem.

a) Napisać program do sterowania serwomechanizmem z silnikiem krokowym i detektorem krańcowym z lab. MiTP.

i) Program powinien wykonywać komendy:

- id -> id anyidentifier // identyfikacja urządzenia,
- calib -> ok // kalibracja serwomechanizmu,
- goto 0x{pozycja w krokach heksadecymalnie} -> ok // przesuń do,
- step 0x{pozycja w krokach heksadecymalnie} -> ok // przesuń o,

ii) W przypadku nierozpoznania komendy program powinien odsyłać "unknowncommand",

iii) Do dekodowania użyć funkcji dekodujących oraz funkcji do operacji na stringach z lab. MiTP,

iv) Struktura pętli głównej powinna wyglądać następująco:

Odebranie komendy (gets)
Dekodowanie komendy
Wykonanie komend w tym wysłanie odpowiedzi (puts)

b) Zmodyfikować wcześniejszy program tak, aby funkcje do dekodowania komend używały standardowych funkcji C do operacji na łańcuchach.