Imię, nazwisko

Przedmiot.

Sprawozdanie z laboratorium 1.

1. Celem pierwszej części zajęć było dokonanie pomiaru czasu wykonywania operacji wejścia/wyjścia oraz operacji arytmetycznych za pomocą dostarczonych funkcji, korzystających z procedur systemowych.

Realizując zadania skopiowałem potrzebne pliki ze strony przedmiotu oraz zbadałem dostarczone funkcje służące do pomiaru czasu korzystające z procedur systemowych: *getrusage(RUSAGE_SELF, &rp) i gettimeofday(&tp,&tzp).* Pierwsza z procedur pobiera informacje o wykorzystaniu zasobów systemowych, między innymi czasu CPU, przez proces (*getresusage – get resource usage* – zwraca strukturę, której pola *ru_utime.tv_usec i ru_utime.tv_usec* pozwalają obliczyć zużycie czasu CPU przez proces w sekundach). Funkcja *gettimeofday* zwraca czas odmierzany od pewnego punktu, także w polach struktury będącej jej argumentem: *tv_sec i tv_usec*. Obie funkcje pozwalają uzyskać pomiar czasu z dokładnością ok. mikrosekundy, znacznie wyższą od dokładności funkcji *clock* języka C.

Dostarczone funkcje pomiaru czasu pozwalają na pracę w dwóch trybach: funkcje *inicjuj_czas() i drukuj_czas()* stanowią parę – pierwsza inicjuje zerami lokalną strukturę danych do odmierzania czasu, drugą drukuje różnicę aktualnych wartości pomiaru czasu i wartości zapisanych w strukturze. Funkcje *czas_zegara() i czas_CPU()* zwracają aktualne wartości odmierzanych czasów i wymagają co najmniej dwóch pomiarów w różnych punktach programu z zapisaniem wartości w zmiennych lokalnych do ustalenia czasu użytego pomiędzy punktami programu

Dokonałam modyfikacji dostarczonego pliku źródłowego *moj_program.c* umieszczając dyrektywę dołączenia pliku nagłówkowego kodu pomiaru czasu oraz wywołania procedur pomiaru i wydruku czasu (kod w załączniku 1).

2. W celu uruchomienia programu dokonałam odpowiednich modyfikacji pliku Makefile.

Modyfikacje polegały na uzupełnieniu dostarczonego wzorca tak aby poprawnie tworzyć plik binarny *moj_program* poprzez:

- dodanie plików *moj_program.c pomiar_czasu.h* w linii zależności pliku *moj_program.o*
- wpisanie polecenia kompilacji z odpowiednimi opcjami pliku źródłowego *moj_program.c*

Wydruk pliku Makefile zawiera załącznik 2.

3. Ostatnim etapem było uruchomienie kodu i dokonanie pomiaru czasu wykonywania operacji

Otrzymane wyniki pozwalają obliczyć czas wykonania pojedynczej operacji:

- arytmetycznej (mnożenia):

  - czas CPU – $8{,}64*10^{-9}$
  - czas zegara – $8{,}64*10^{-9}$

- we/wy (wydruk pojedynczej liczby w terminalu z przesłaniem danych z serwera przez sieć):

  - czas CPU – $186{,}63*10^{-9}$
  - czas zegara – $265{,}84*10^{-9}$

(pomiar realizowany jest na serwerze, a wydruk jest lokalny – czas mierzony obejmuje obsługę wysłania przez sieć)

4. Celem zadań dodatkowych było przetestowanie umiejętności tworzenia bibliotek z wykorzystaniem narzędzia *make.* W ramach zadań dodatkowych zrealizowałam:

- przeniosłam pliki związane z pomiarem czasu do odrębnego katalogu pomiar_czasu
- w katalogu pomiar_czasu utworzyłam bibliotekę *libpomiar_czasu.a* z pliku *pomiar_czasu.o* poleceniem *ar -rs libpomiar_czasu.a pomiar_czasu.o*
- zmodyfikowałam plik *Makefile* tak aby zamiast z plików źródłowych i pośrednich pomiaru czasu korzystał z biblioteki:
  - usunąłem odniesienia do plików *pomiar_czasu.c i pomiar_czasu.o*
  - nadałem właściwe wartości symbolom LIB i INC – pierwszy wskazuje na położenie bibliotek (opcja -L) oraz dołączenie bibliotek przy konsolidacji (opcja -l), drugi (INC) wskazuje położenie plików nagłówkowych do włączania do plików z kodem źródłowym w trakcie kompilacji kodu
  - dodałem właściwe wykorzystanie symboli LIB i INC w trakcie kompilacji

Zawartość zmodyfikowanego pliku Makefile znajduje się w załączniku 4.

- przetestowałem kompilacje kodu z nową wersja pliku Makefile

Wnioski:

- **dzięki wykorzystaniu narzędzia make oraz odpowiednio modyfikowanego pliku Makefile udało się zrealizować sprawne i elastyczne tworzenie kodu binarnego z plików źródłowych**
- **procedury systemowe Linuxa pozwalają na dokładny pomiar czasu wykonania fragmentów kodu - i czasu CPU, i czasu zewnętrznego zegarowego, z dokładnością ok. 1 mikrosekundy**
- **pomiar czasu realizacji procedur wykazał, że:**
  - **czas realizacji procedur wejścia/wyjścia jest znacznie dłuższy niż czas realizacji operacji arytmetycznych**
  - **przy realizacji procedur wejścia/wyjścia czas wykorzystania procesora (czas CPU) jest niższy od czasu zegarowego (proporcja może zależeć od sposobu funkcjonowania środowiska graficznego systemu oraz od faktu czy pomiar realizowany jest dla operacji lokalnych czy wydruk odbywa się zdalnie)**
  - **optymalizacje kodu wpływają na czas realizacji instrukcji kodu źródłowego, natomiast nie zmieniają czasu wykonania funkcji systemowych**
- *itd. itp.*

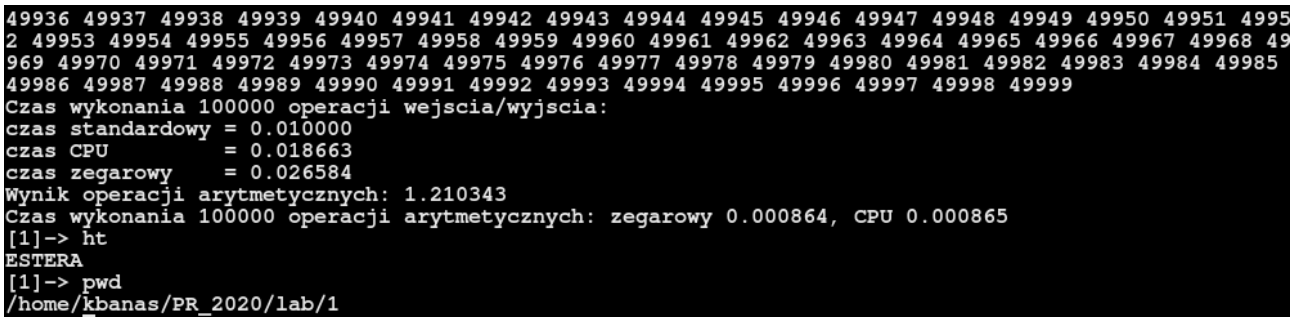
Załącznik 1. kod programu *moj_program.c* z umieszczonymi funkcjami pomiaru czasu

```
#include"pomiar_czasu.h"
....
inicjuj_czas();
for(i=0;i<liczba;i++){
   printf("%d ",k+i);
}
printf("\n");
drukuj_czas();
...
t1=czas_zegara(); t2=czas_CPU();
for(i=0;i<liczba;i++){
   a = 1.000001*a;
}
t1=czas_zegara()-t1; t2=czas_CPU()-t2;
printf("Czas wykonania %d operacji arytmetycznych: zegarowy %lf, CPU %lf\n",liczba,t1,t2);
```

Załącznik 2 Uzupełniony plik *Makefile* służacy do uzyskania pliku binarnego moj_program

```
# jak uzyskac plik moj_program.o ?
moj_program.o: moj_program.c pomiar_czasu.h
        $(CCOMP) -c $(OPT) moj_program.c $(INC)
```

Załącznik 3 Zrzut ekranu z widokiem terminala po wykonaniu programu *moj_program* i wynikami pomiaru czasu wykonania operacji drukowania w terminalu (z przesyłaniem danych przez sieć) oraz operacji arytmetycznych



Załącznik 4 Plik *Makefile* służący do uzyskania biblioteki z funkcjami pomiaru czasu.

```
# pliki naglowkowe
INC = -I./pomiar_czasu
# biblioteki
LIB = -L./pomiar_czasu -lpomiar_czasu -lm
# zaleznosci i komendy
moj_program: moj_program.o
	$(LINKER) $(OPT) moj_program.o -o moj_program $(LIB)
# jak uzyskac plik moj_program.o ?
moj_program.o: moj_program.c pomiar_czasu/pomiar_czasu.h
	$(CCOMP) -c $(OPT) moj_program.c $(INC)
```

itd. itp.