Aleksander Strzeboński, Piotr Radecki

Końcowa dokumentacja projektu z przedmiotu Programowanie Sieciowe

1. **Temat projektu**

Zrealizowany został temat nr 13: Implementacja protokołu: Synchronizing Internet Clock frequency protocol (SIC).

1. **Zawartość projektu**

Do projektu dołączam folder plikami:

* changing\_time.c
* check\_time.c
* correct\_time.c
* sic.c
* sic\_extended.c
* udpserver.c
* Makefile

1. **Opis protokołu**

Protokół SIC ma za zadanie dosynchronizować zegar wewnętrzny klienta do zegara wewnętrznego serwera. Główne założenia protokołu są następujące. Klient pobiera znacznik czasowy t1 czyli aktualny czas liczony w mikrosekundach od 1 stycznia 1970 roku i wysyła go do serwera. Serwer zaraz po odebraniu wiadomości zaraz przed odesłaniem wiadomości pobiera znaczniki czasowe t2 i t3 i dopisuje je do wiadomości odsyłanej do klienta. Po otrzymaniu wiadomości klient pobiera znacznik czasowy t4. Na podstawie tych czterech znaczników czasowych można obliczyć różnicę phi pomiędzy zegarami wewnętrznymi serwera i klienta. Należy pamiętać, że różnica ta nie jest stała, gdyż częstotliwości zegara wewnętrznego klienta i serwera nie są jednakowe. Klient powtarza powyższy schemat obliczania phi wielokrotnie. Następnie przy pomocy regresji liniowej przybliża funkcję phi(t) funkcją liniową:

phi(t[us]) = K + F\*t[s]

Obliczone współczynniki K i F zapisywane są w pliku. Na podstawie tych współczynników można w każdej chwili wyliczyć w kliencie aktualny czas serwera.

1. **Opis programów**

**Uwaga: przed skompilowaniem programów komendą *make* należy zainstalować bibliotekę GSL. W CentOSie robi się to komendą:**

**yum install gsl-devel**

* **changing\_time.c**

Program changing\_time.c służy do zmiany zegaru wewnętrznego komputera o bezwzględną wartość. Program należy wywołać w następujący sposób:

./changing\_time sec usec

Gdzie sec oznacza liczbę sekund o jaką chcemy zmienić zegar, a usec liczbę mikrosekund o jaką chcemy zmienić zegar (liczba mikrosekund nie może przekroczyć miliona). Uwaga: program należy wykonać z uprawnieniami roota, gdyż tylko root może zmieniać zegar wewnętrzny.

* **udpserver.c**

Program udpserver.c przy pomocy funkcji fork tworzy dwa serwery. Należy wysołać go w następujący sposób:

./udpserver adres\_ipv4

Czyli np.:

./udpserver 192.168.56.25

adres\_ipv4 to adres na którym będzie uruchomiony Serwer nr 2.

Serwer nr 1: serwer multicastowy, który co 5 sekund na adres 224.0.0.1 z portu 5300 wysyła wiadomość o treści np. „192.168.56.25”. Jest to adres na którym działa drugi serwer unicastowy. Serwer nr 2 jest serwerem UDP działającym na podanym adresie i porcie 4444. Jest to serwer implementujący protokół SIC po stronie serwera. Oczekuje on na wiadomość a następnie odsyła otrzymaną wiadomość z dodanymi znacznikami t2 i t3. Program działa w trybie demona. Wywołuje się go bez dodatkowych argumentów.

* **sic.c**

Program sic.c implementuje protokół SIC po stronie klienta. Program należy wywołać w następujący sposób:

./sic 224.0.0.1 5300 scaler(opcjonalnie)

Program najpierw czeka na adresie multicast 224.0.0.1 na porcie 5300 na wiadomość od serwera z adresem unicast na którym uruchomiony jest serwer SIC. Następnie klient UDP co sekundę przez 60 sekund wymienia wiadomości z serwerem SIC i oblicza różnice czasu phi oraz RTT (Round Trip Time). Na podstawie obliczonych phi klient oblicza współczynniki K oraz F i zapisuje je w pliku *sic.txt* w katalogu z którego został uruchomiony w formacie:

K

F

Jeżeli program wywołany jest z argumentem scaler, to klient pobrane znaczniki t1 i t4 przemnaża przez scaler w celu zasymulowania niewłaściwej częstotliwości zegara wewnętrznego. Jeżeli w trakcie wymiany danych zmieni się ścieżka, co skutkuje dużą zmianą RTT, procedura wymiany danych z serwerem SIC rozpocznie się od nowa.

* **correct\_time.c**

Program correct\_time.c należy wywołać w następujący sposób:

./correct\_time scaler(opcjonalnie)

Program ten służy do poprawienia wewnętrznego zegara klienta przy użyciu współczynników K i F zapisanych wcześniej w pliku *sic.txt*. Wywołany bez argumentu zmienia zegar wewnętrzny klienta na czas serwera. Uwaga: w wyniku tej zmiany współczynniki K i F dezaktualizują się i nie powinny być już więcej wykorzystywane. Gdy program ten wywołany zostanie z argumentem scaler, to jedynie obliczy on czas serwera mnożąc wcześniej swój czas przez scaler w celu zasymulowania niepoprawnej częstotliwości zegara. Program należy uruchomić z poziomu użytkownika root.

* **check\_time.c**

Program check\_time.c uruchamia klienta UDP który łączy się z serwerem SIC, wymienia jedną wiadomość i oblicza różnicę w czasach phi w celu sprawdzenia czy klient jest zsynchronizowany z serwerem. Program należy wywołać w następujący sposób:

./check\_time adres\_serwera scaler(opcjonalne)

Program oblicza czas serwera na podstawie współczynników K i F zapisanych w pliku sic.txt i tak obliczone czasy t1 i t4 używa do komunikacji z serwerem i obliczenia phi. Jeżeli podany jest argument scaler to program przemnaża wewnętrzny czas klienta przez scaler. Program nie zmienia czasu wewnętrznego na czas serwera, dzięki czemu współczynniki K i F nie dezaktualizują się.

* **sic\_extended.c**

Program sic\_extended jest programem demona, który spina powyższe programy klienta w całość. Wywołanie jest następujące:

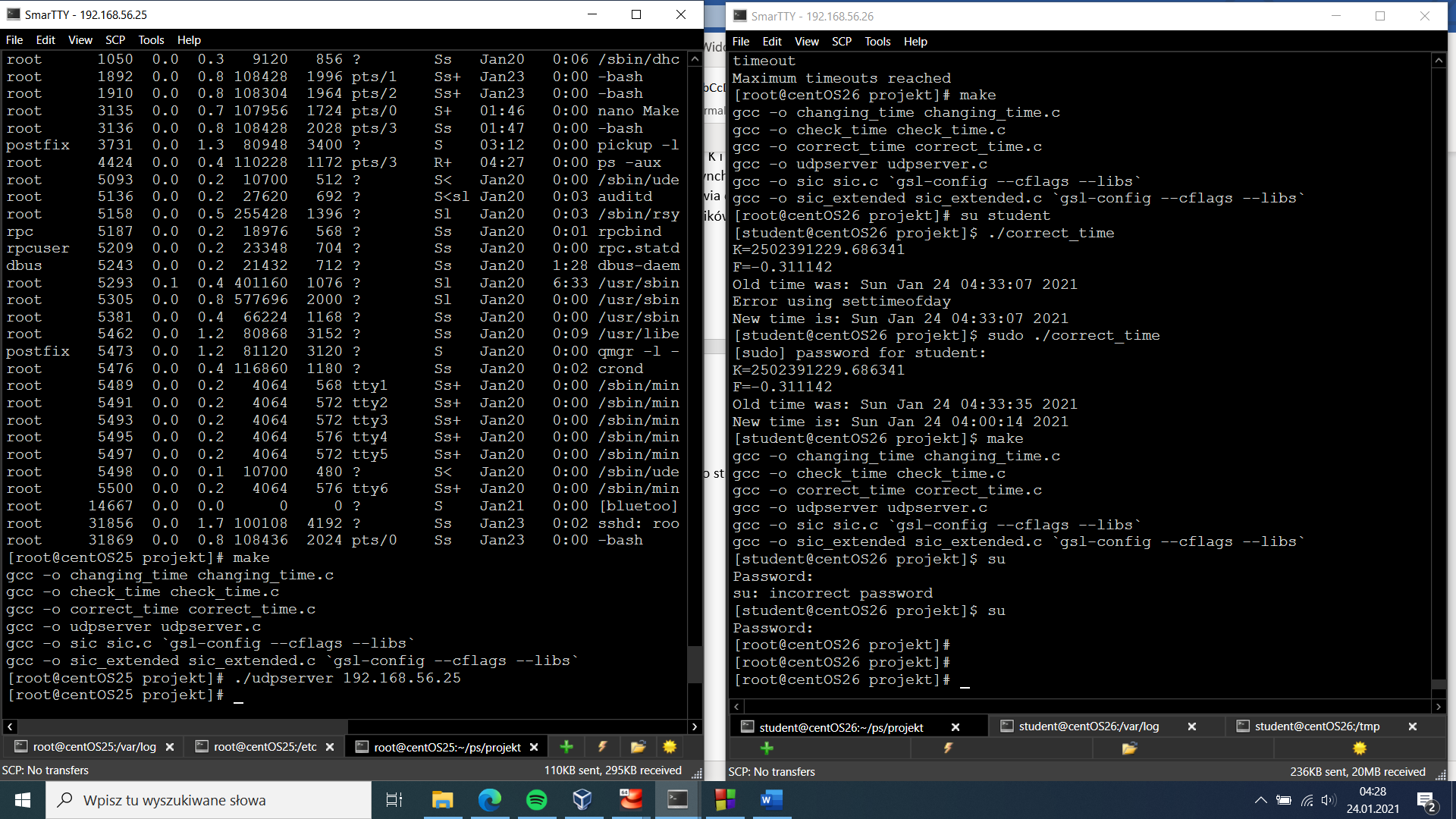
./sic\_extended 224.0.0.1 5300 scaler(opcjonalne)

scaler działa analogicznie jak w powyższych programach.

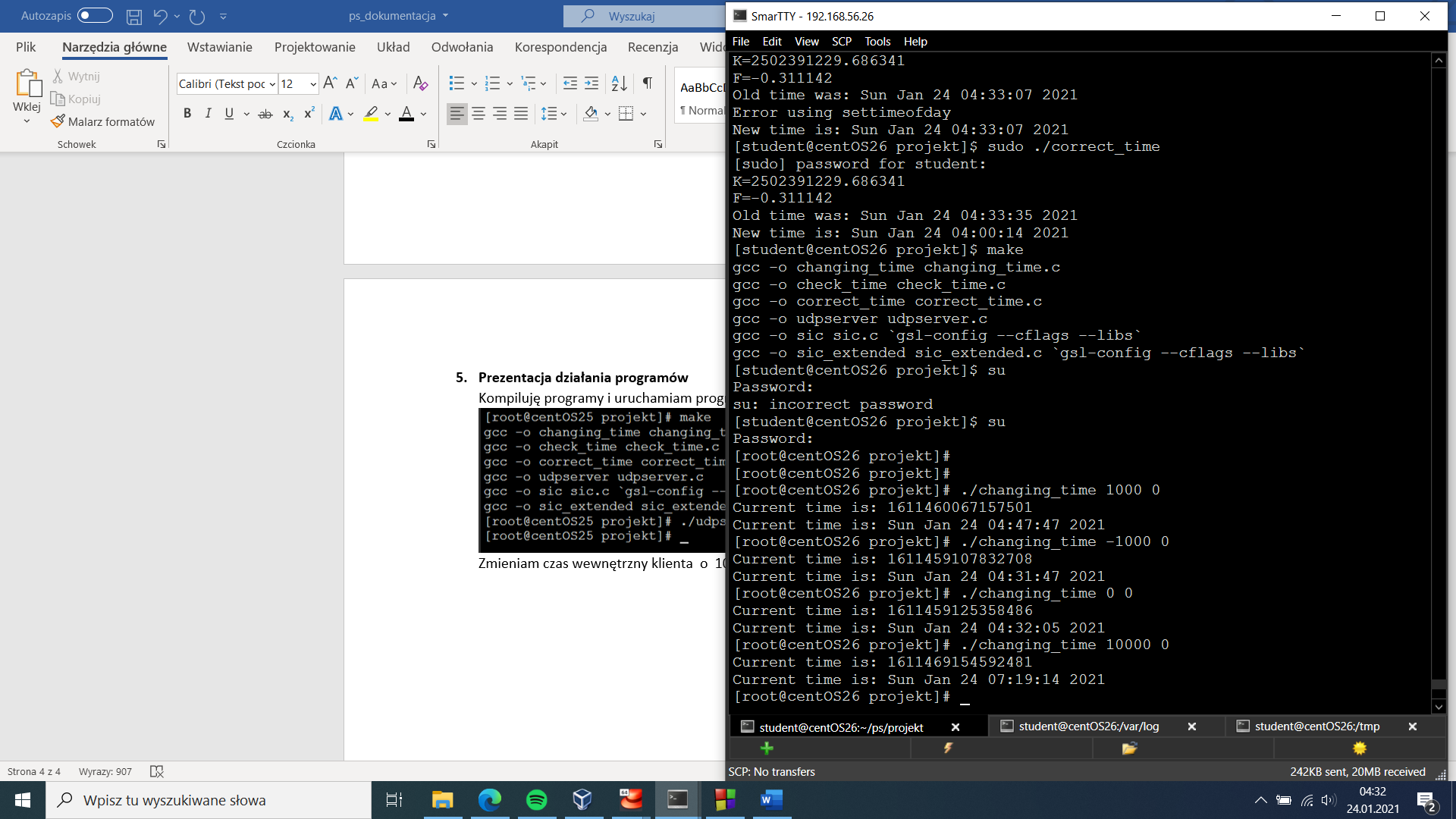
Program najpierw robi to samo co program sic.c: Odbiera wiadomość multicast, prowadzi komunikację z serwerem SIC i zapisuje obliczone współczynniki K i F w pliku *sic\_ext.txt* w swoim katalogu roboczym tzn. */tmp*. Następnie program poprawia czas wewnętrzny klienta tak jak robi to program correct\_time.c. Potem program po raz kolejny prowadzi w komunikację z serwerem SIC w celu zaktualizowania zdezaktualizowanych współczynników K i F. Potem program co 10 sekund przeprowadza pojedynczą komunikację z serwerem SIC w celu obliczenia różnicy czasów tak jak robi to program check\_time.c. Jeżeli różnica czasów będzie większa niż jedna sekunda, to program ponownie oblicza współczynniki K i F, poprawia czas wewnętrzny klienta, oblicza współczynniki K i F i przechodzi z powrotem do regularnego sprawdzania czy zegary są zsynchronizowane. Jeżeli podany jest argument scaler, to program nie poprawia czasu wewnętrznego klienta i co za tym idzie nie musi obliczać współczynników K i F po dwa razy.

1. **Prezentacja działania programów**

Kompiluję programy i uruchamiam program udpserver.c po stronie serwera:

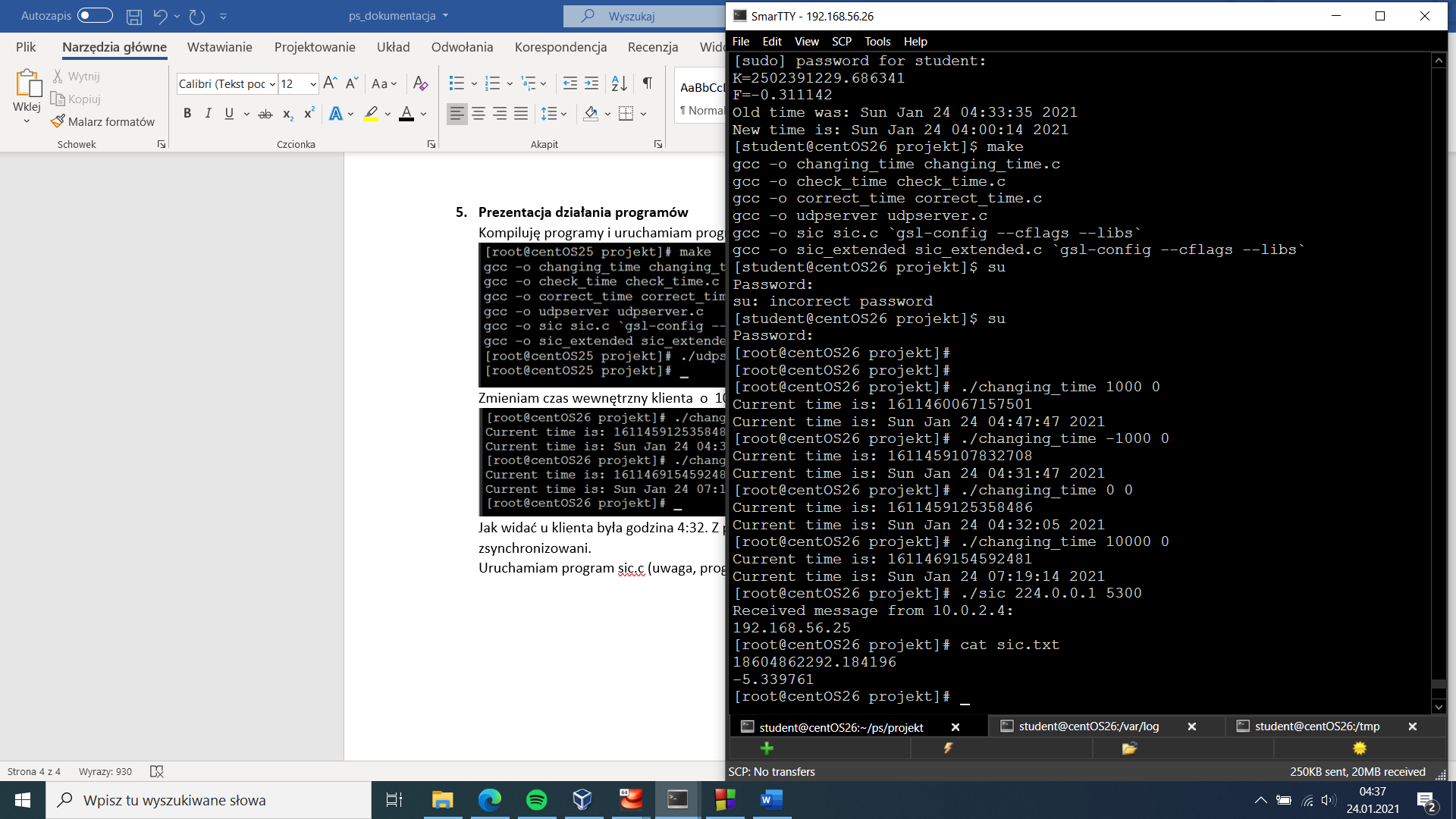


Zmieniam czas wewnętrzny klienta o 10000 sekund:

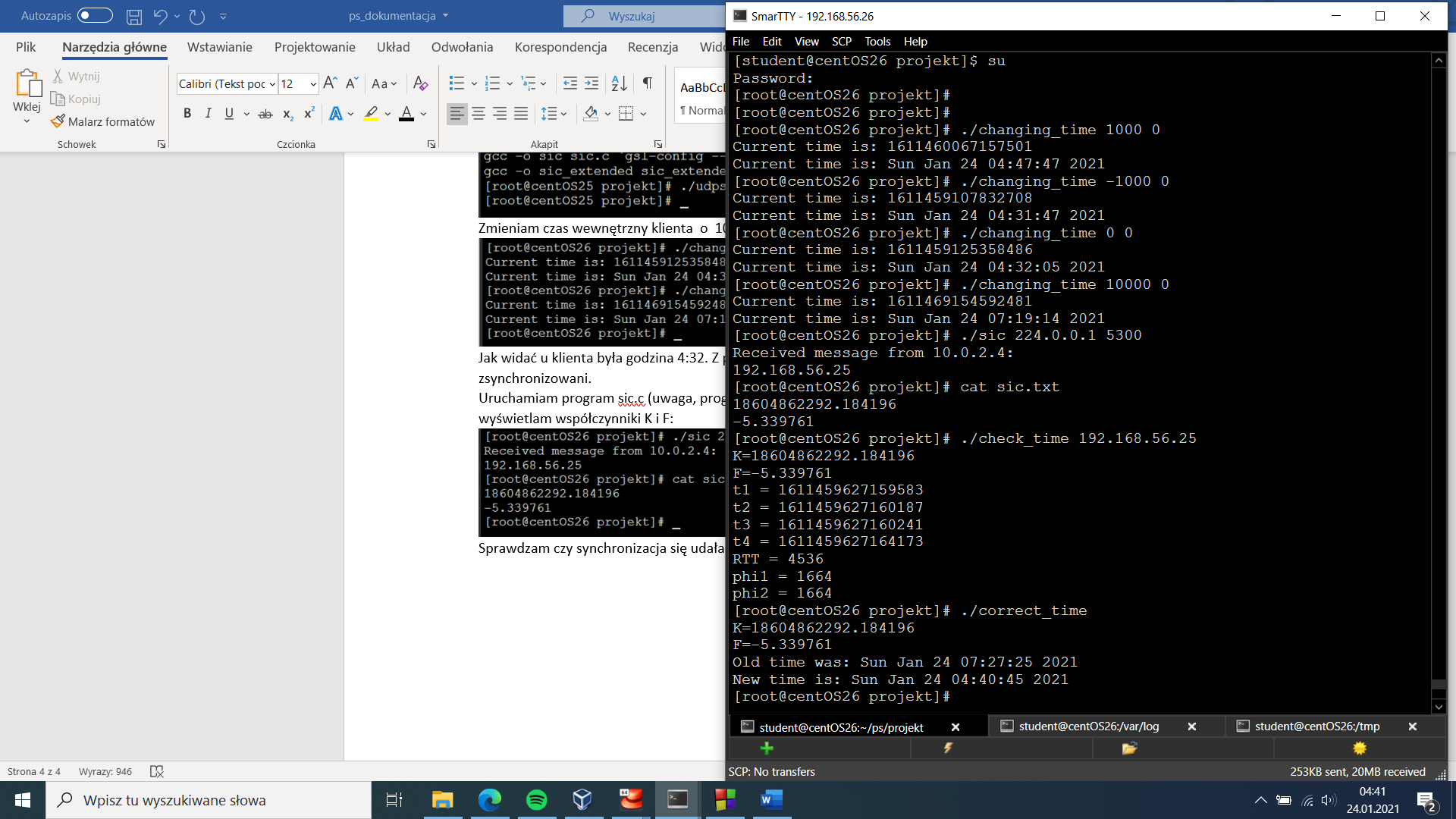


Jak widać u klienta była godzina 4:32. Z początku serwer i klient byli zsynchronizowani.

Uruchamiam program sic.c (uwaga, program wykonuje się około minutę) i wyświetlam współczynniki K i F:

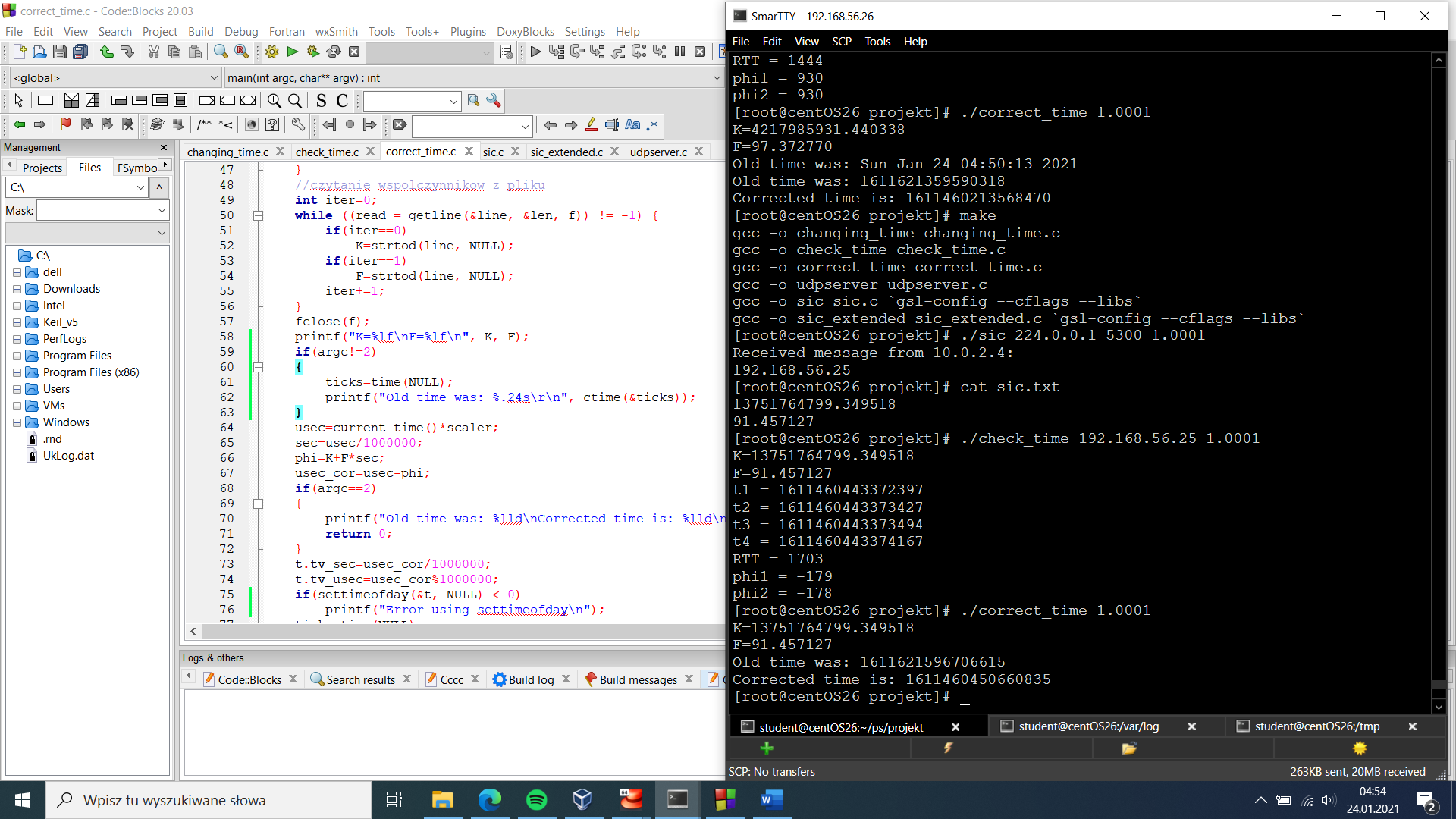


Sprawdzam czy synchronizacja się udała a następnie poprawiam czas klienta:

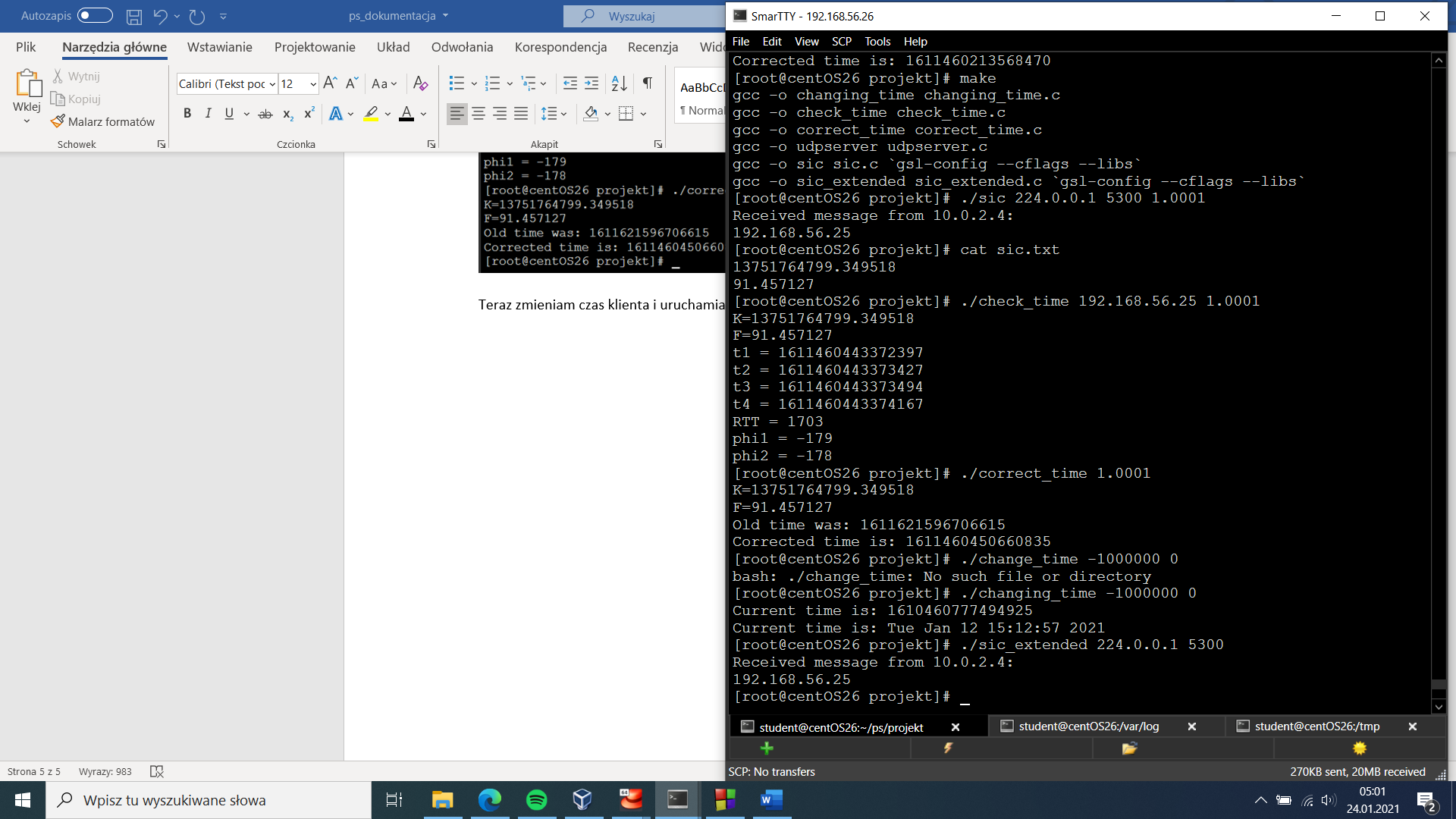
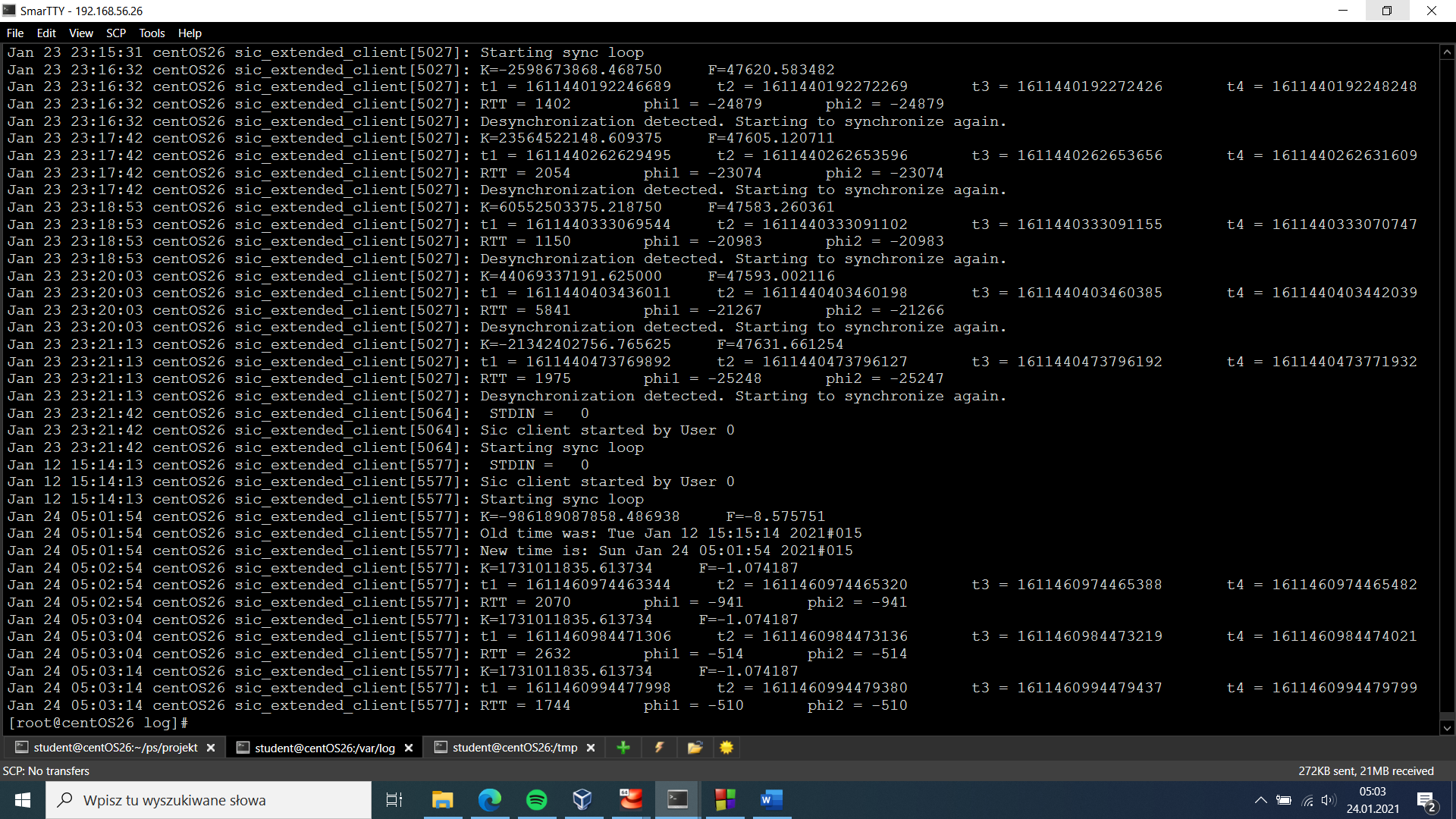


Jak widać różnica czasów wyniosła 1664us więc synchronizacja powiodła się (8 minut między 4:32 a 4:40 uciekło na robienie zrzutów ekranu i opisywanie).

Podobne kroki wykonuję używając opcji scaler:

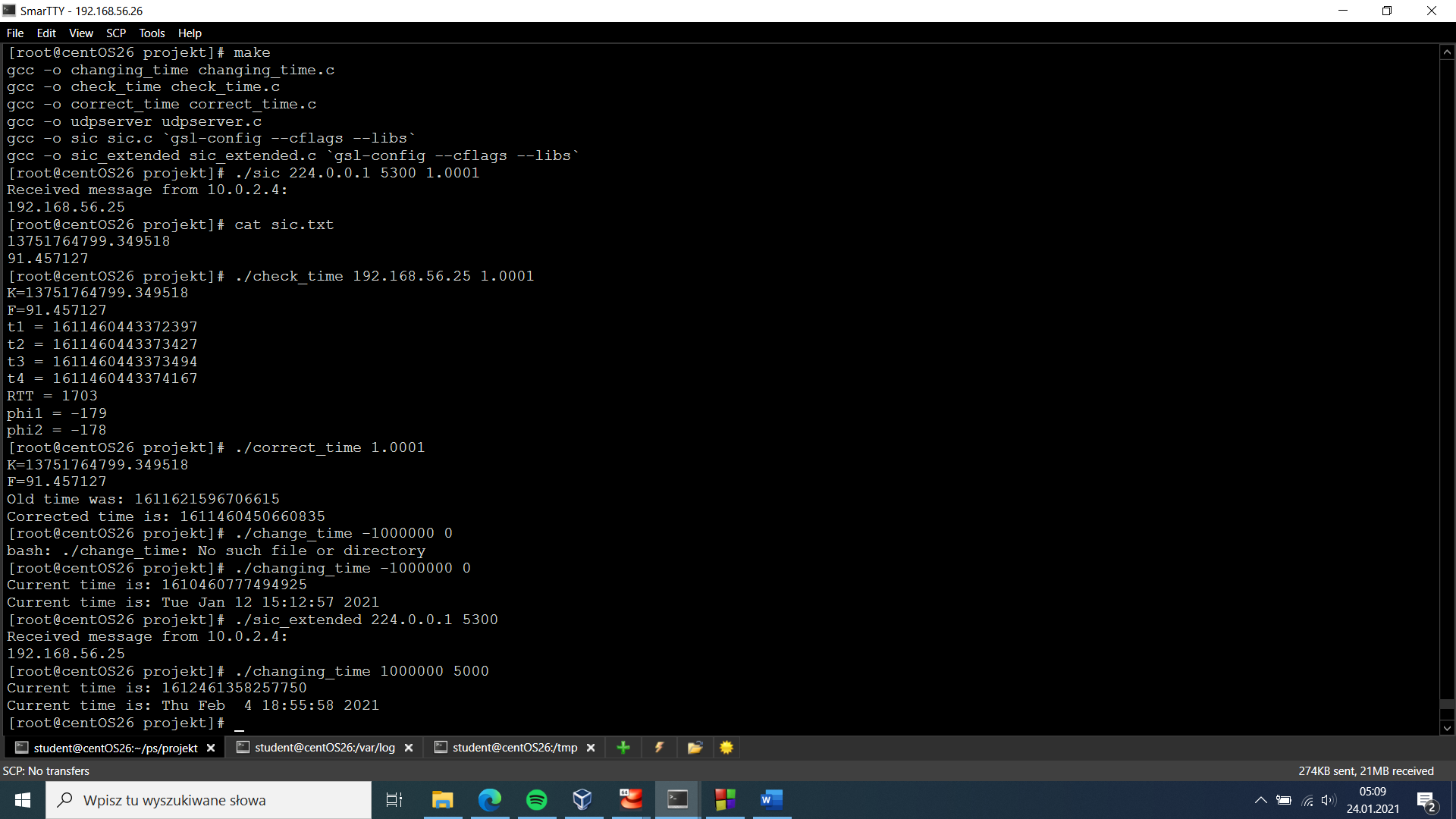


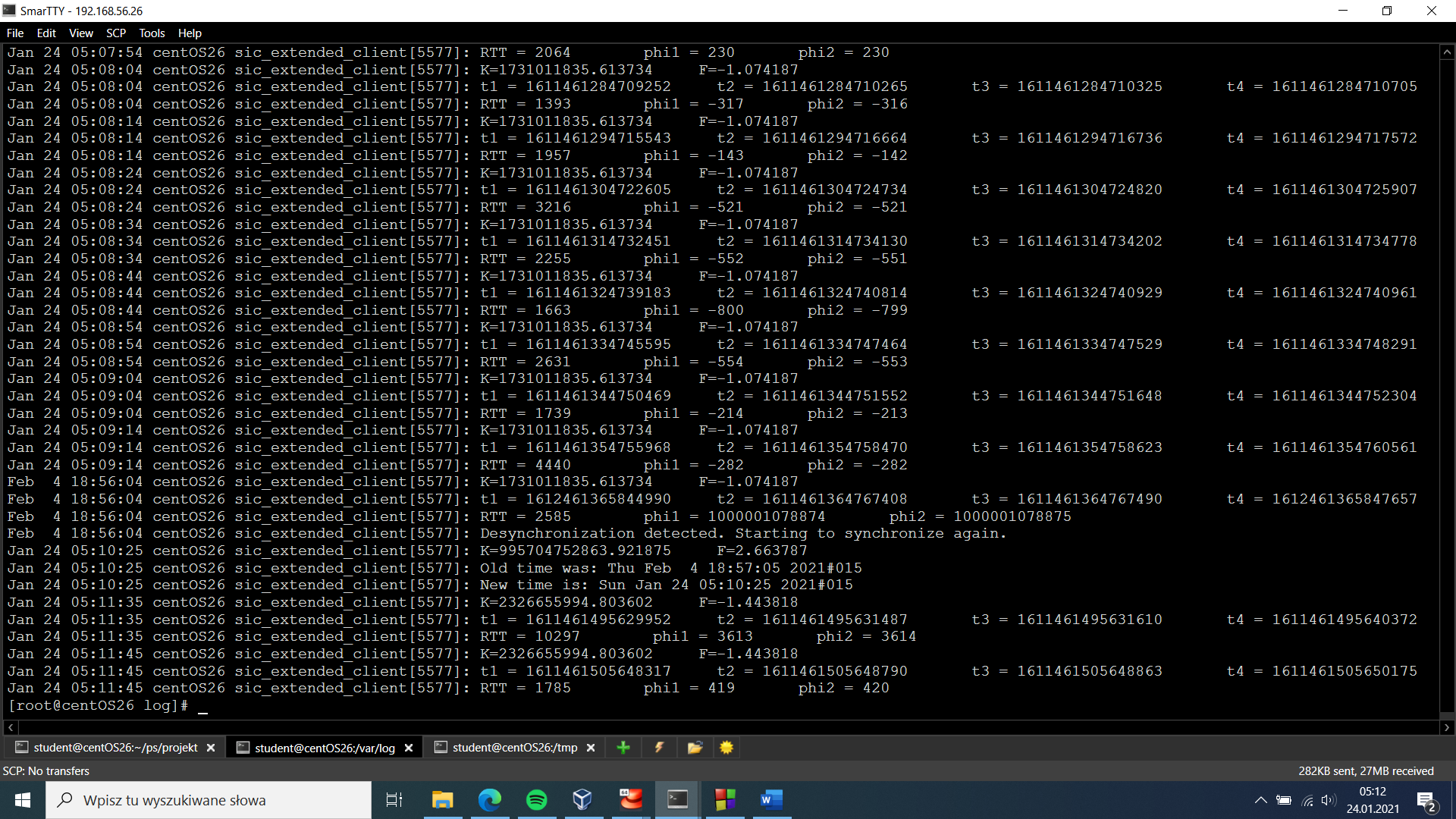
Teraz zmieniam czas klienta i uruchamiam program sic\_extended.c



Jak widać po godzinach logów demona program pomyślnie zsynchronizował się z serwerem i zaczął obliczanie czasów phi.

Następnie jeszcze raz zmieniam czas wewnętrzny klienta i patrzę jak zareaguje.





Desynchronizacja została wykryta i naprawiona.

1. **Źródła**

* <https://tools.ietf.org/id/draft-alavarez-hamelin-tictoc-sic-02.html>
* Materiały zamieszczone na stronie UPEL przedmiotu Programowanie Sieciowe