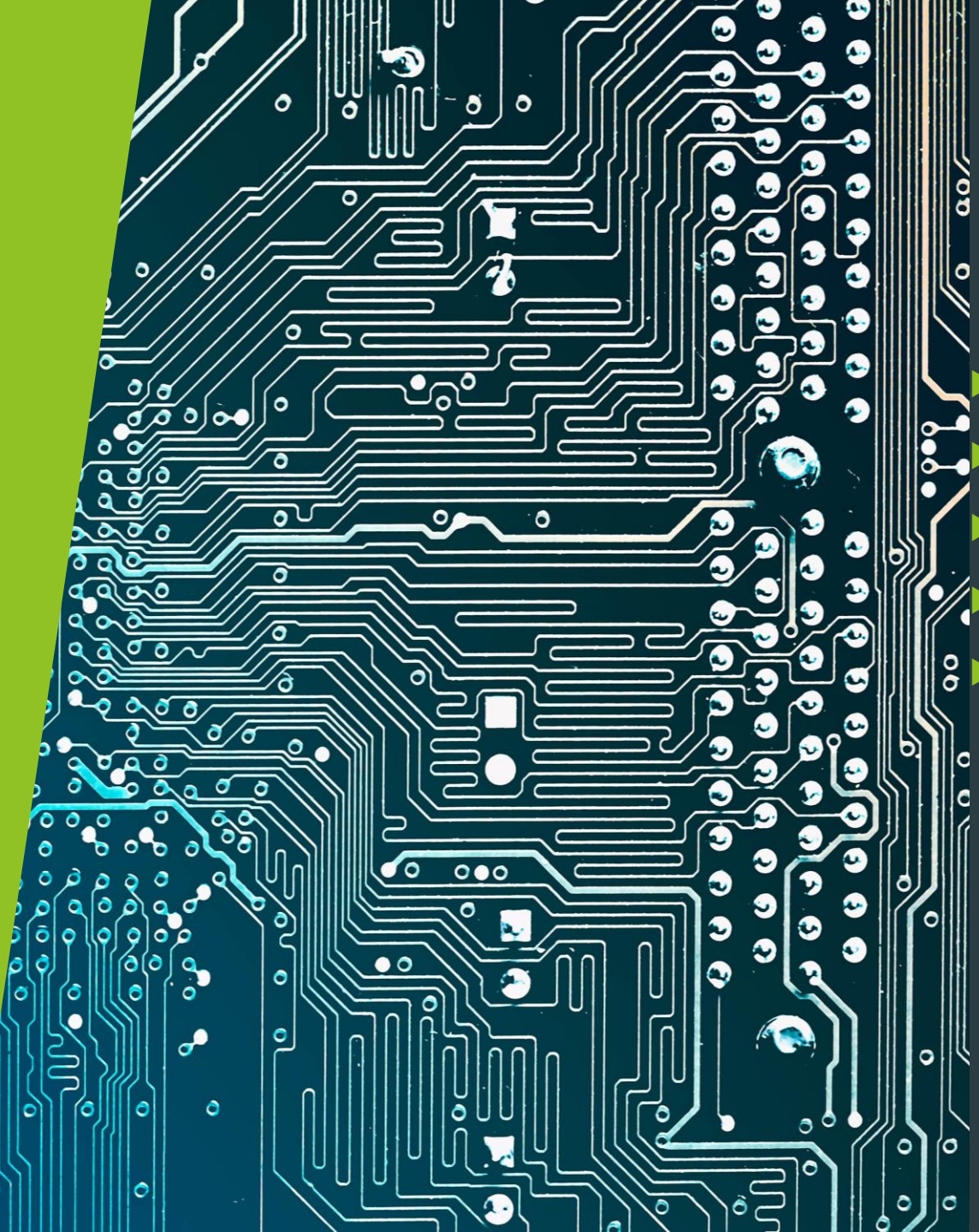
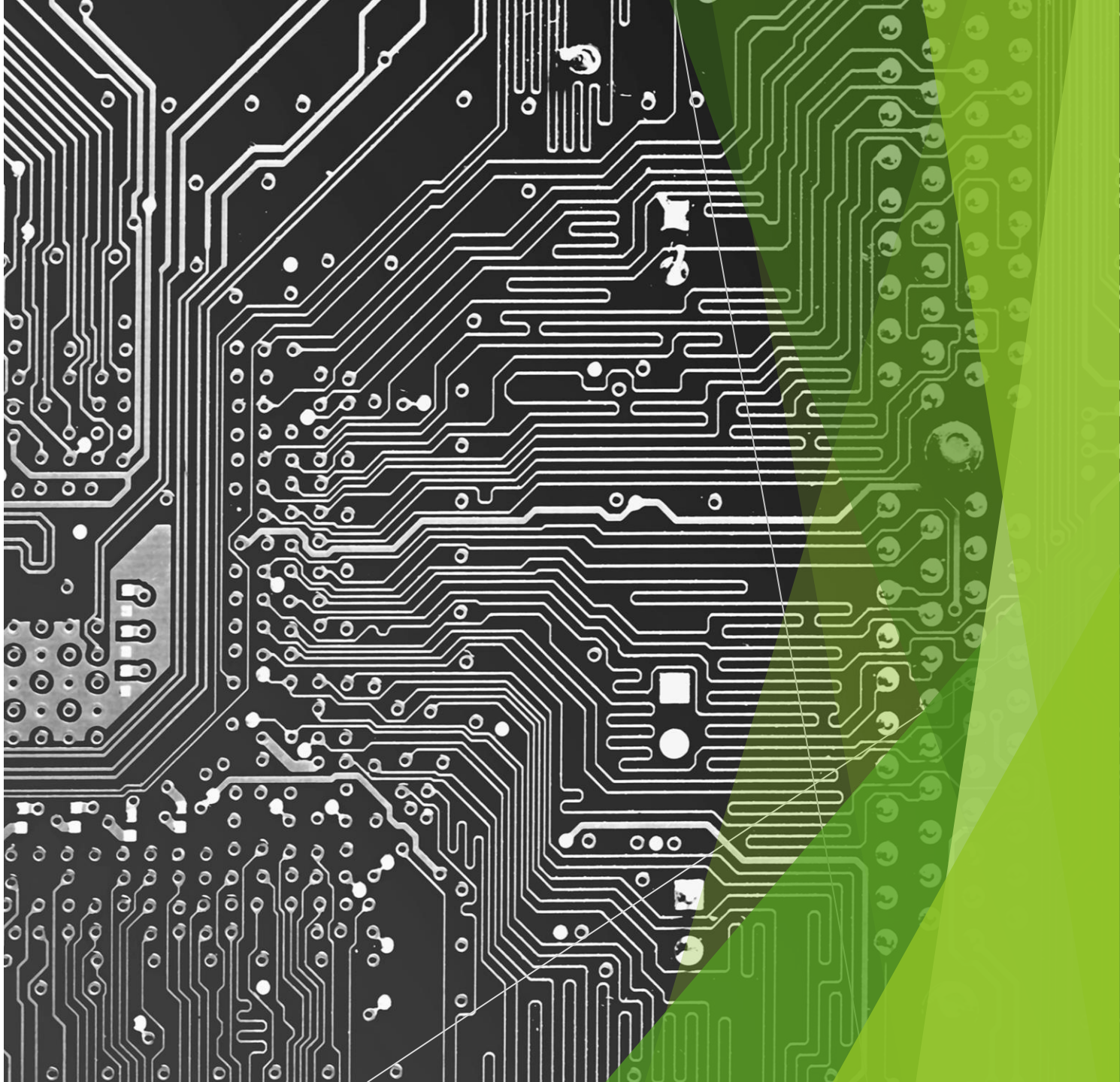
Emulator INTEL8086



Zawarte funkcjonalności:

- Komenda MOV (rejestry + pamięć)
- Komenda XCHG (rejestry + pamięć)
- Komenda POP
- Komenda PUSH
- Walidacja oraz Reset

Interfejs



Skrócony opis interfejsu.

Po lewej górnej stronie okna znajdują się 4 podstawowe rejestry(AX, BX, CX, DX) kliknięcie na pole z danymi pozwala wprowadzić własne dane.

Po prawej znajdują się rejestry pomocnicze(SI, DI, BP, SP oraz DISP).

Są one wykorzystywane przy operacjach na pamięci. Z wyjątkiem SP który oznacza szczyt stosu, wszystkie mogą być modyfikowane podobnie jak podstawowe rejestry.

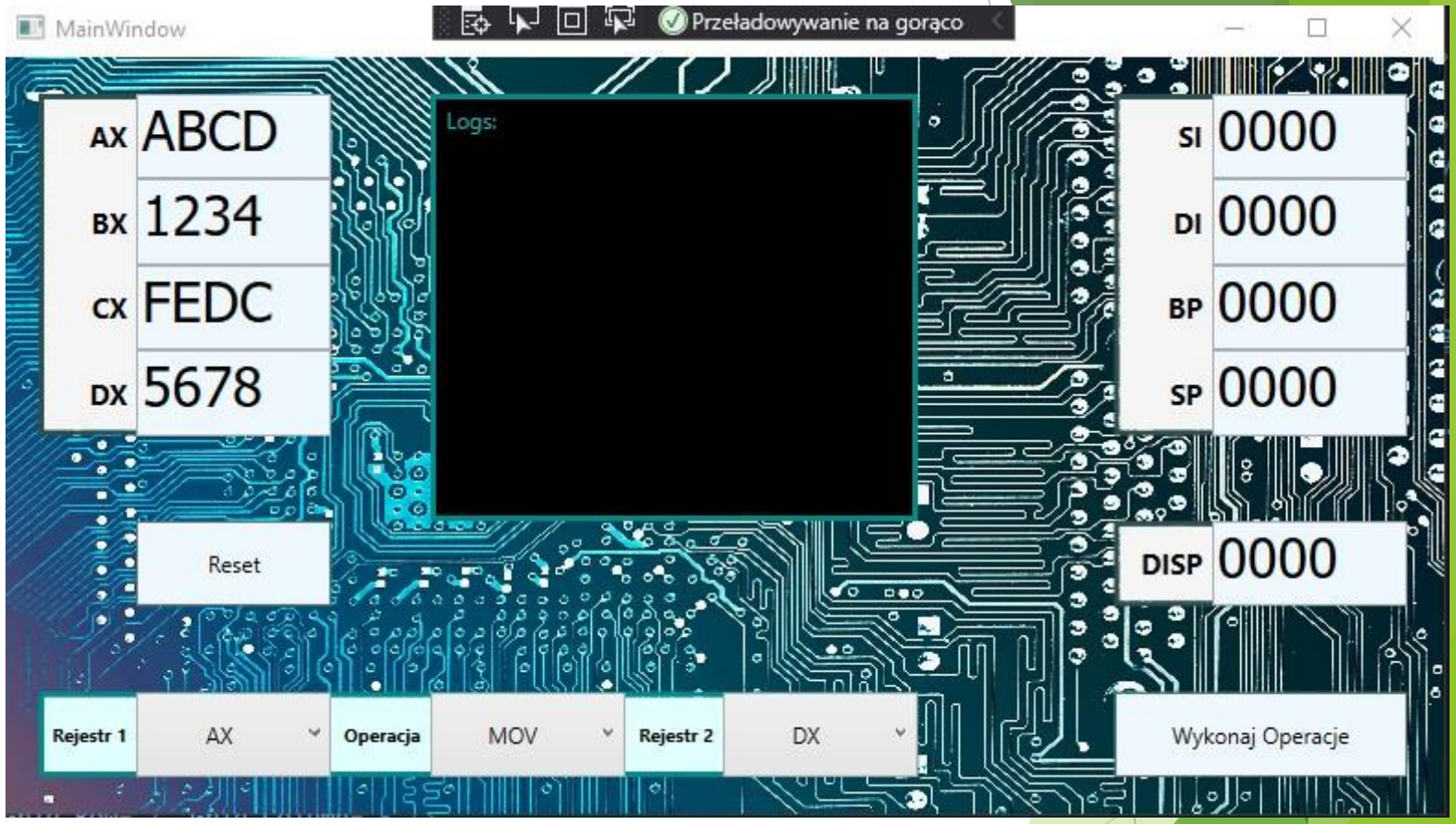
Na sródku znajduje sie okno z logami, będą w nim się pojawiać wszystkie wykonane operacje, można je przewijać za pomocą pokrętłą myszy.

Przycisk Reset pod rejestrami podstawowymi pozwala na reset do stanu początkowego.

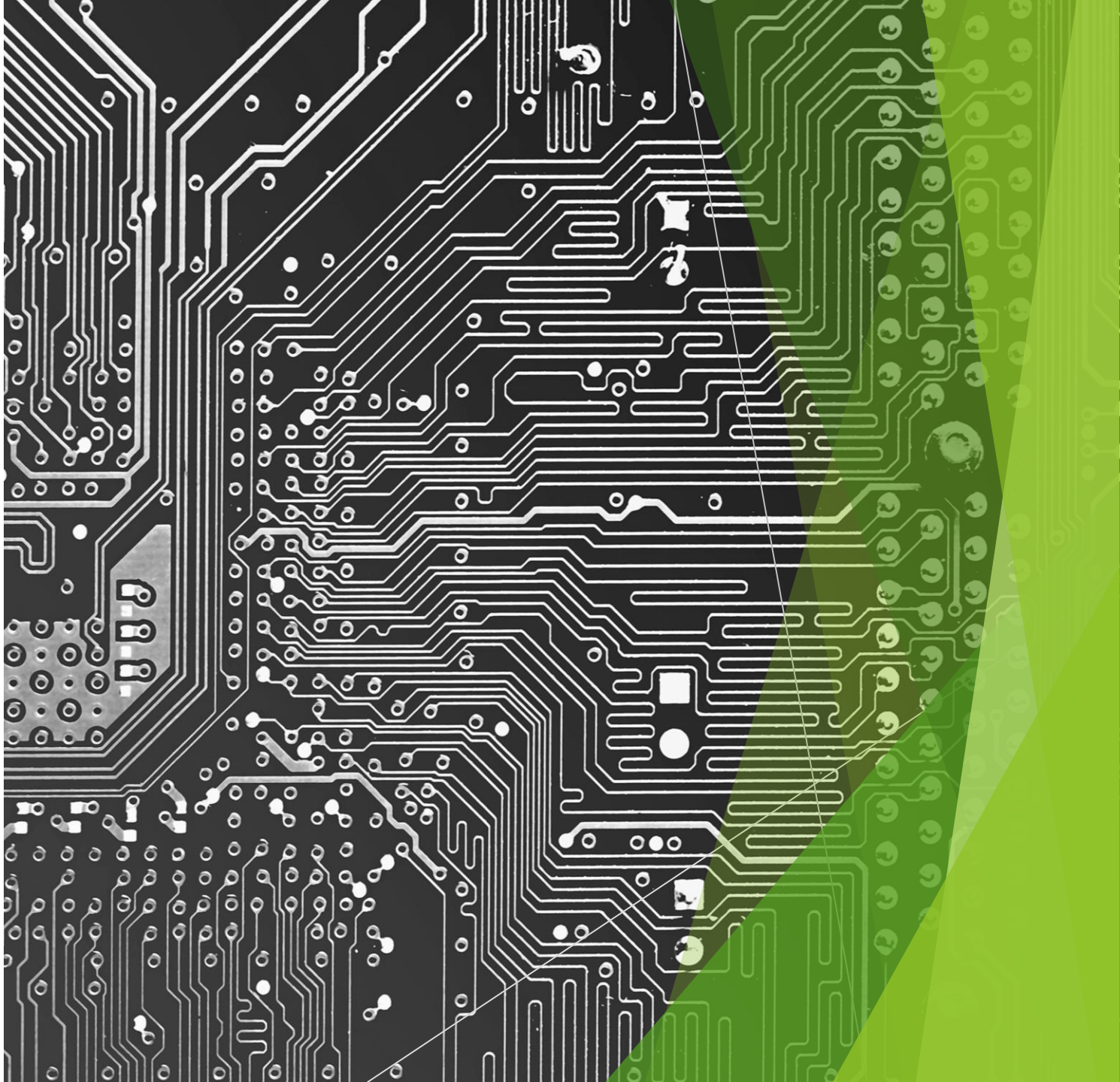
Na samym dole znajduje sie panel kontrolny pozwalajacy wybrac typ operacji ktora chcemy wykonać. Mamy tutaj trzy menu z opcjami. “Rejestr1” oraz “Rejestr2” na których bedzie wykonana operacja oraz “Operacja” ktory pozwala wybrac typ operacji.

Zależnie od wybranej operacji różne opcje bedą dostępne w Rejestrach, podobnie zależnie od wyboru w Rejestrze1 rozne opcje bedą dostępne w Rejestrze2.

Np.: jeżeli w Rejestrze 1 wybierzemy “AH”, w Rejestrze 2 bedą dostępne tylko połówki rejestrów (AL, BH, BL etc..).



Komenda MOV

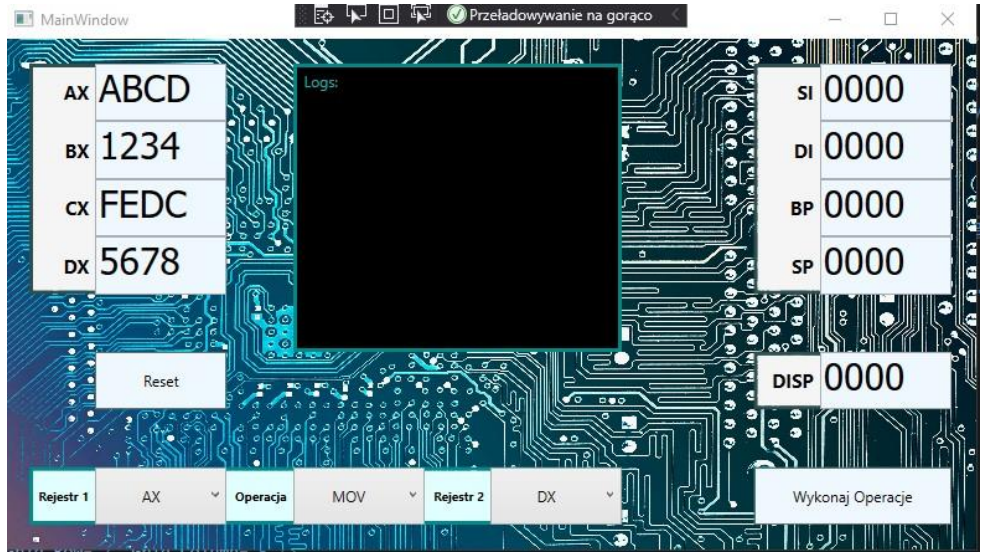


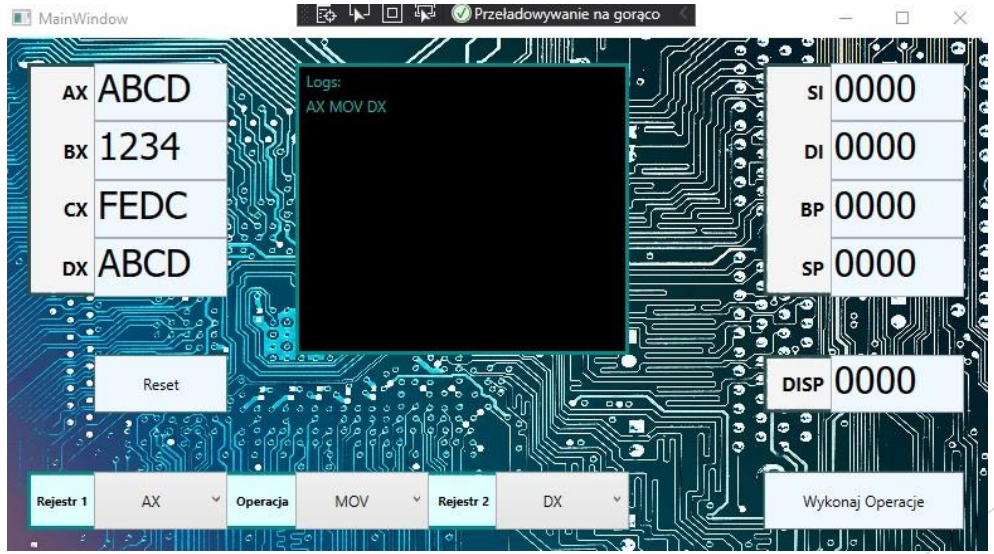
Komenda MOV pozwala na zapisanie danych z jednego rejestru/pamieci w innym rejestrze pamieci.

Na tym przykładzie dane z AX zapiszemy do DX

Pierwszy obrazek przedstawia dane początkowe, na drugim widzimy dane po wykonaniu operacji.

Dane z AX zostały poprawnie zapisane pod DX



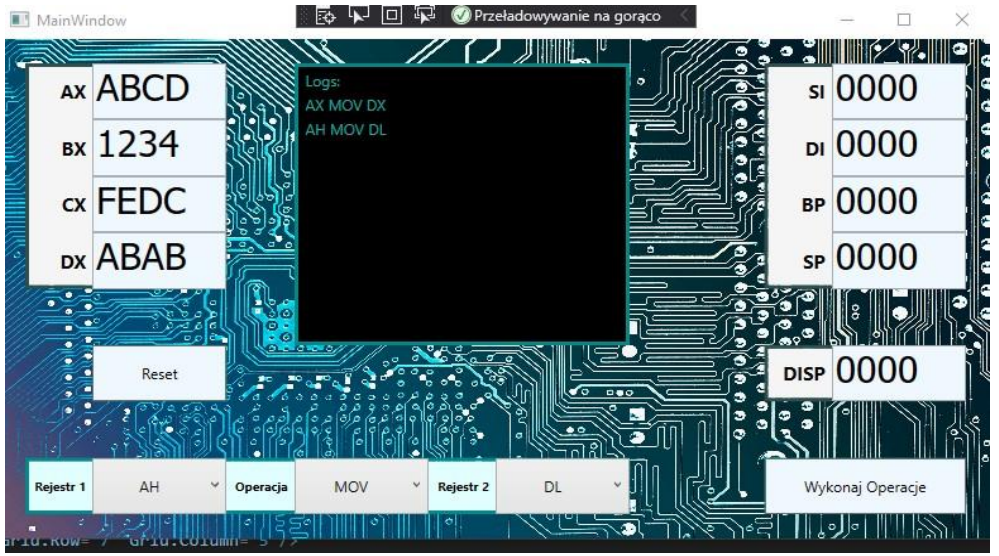


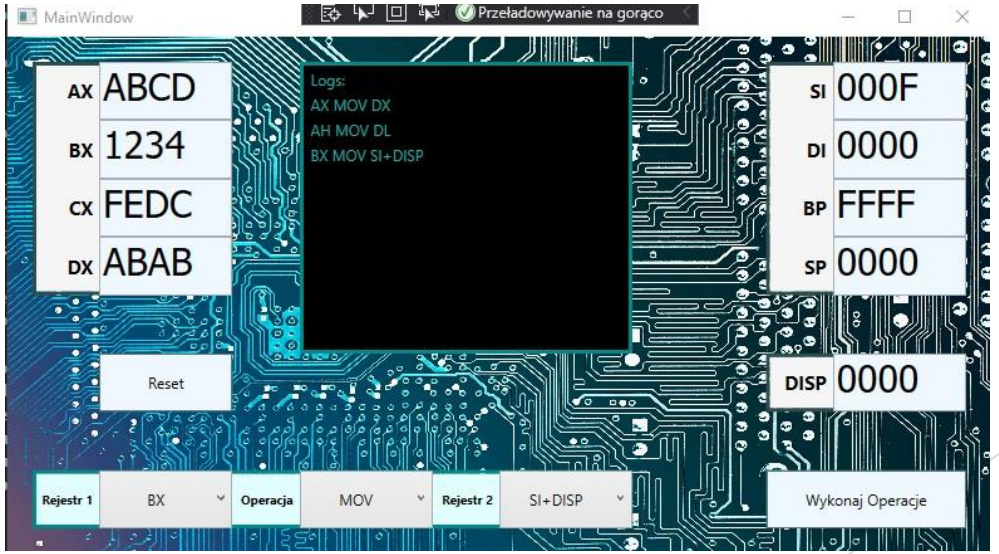
Podobna operacja może rownież zostac wykonana na “połówkach” rejestrow.

Na pierwszym obrazku widzimy przykład gdzie dane z AH zostały zapisane w DH.

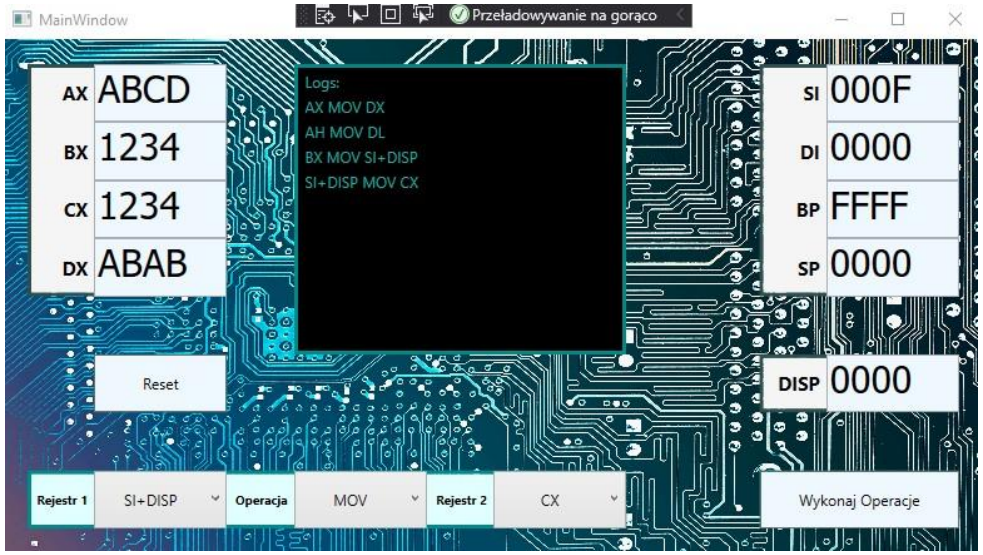
Na drugim obrazku widzimy przykład zapisywania danych do pamięci.

W tym wypadku dane z rejestru BX zostaly zapisane w pamieci, docelowy segment bazy danych zostal ustalony indeksowo. Mozliwe jest rowniez ustalenie go bazowo oraz indeksowo-bazowo.





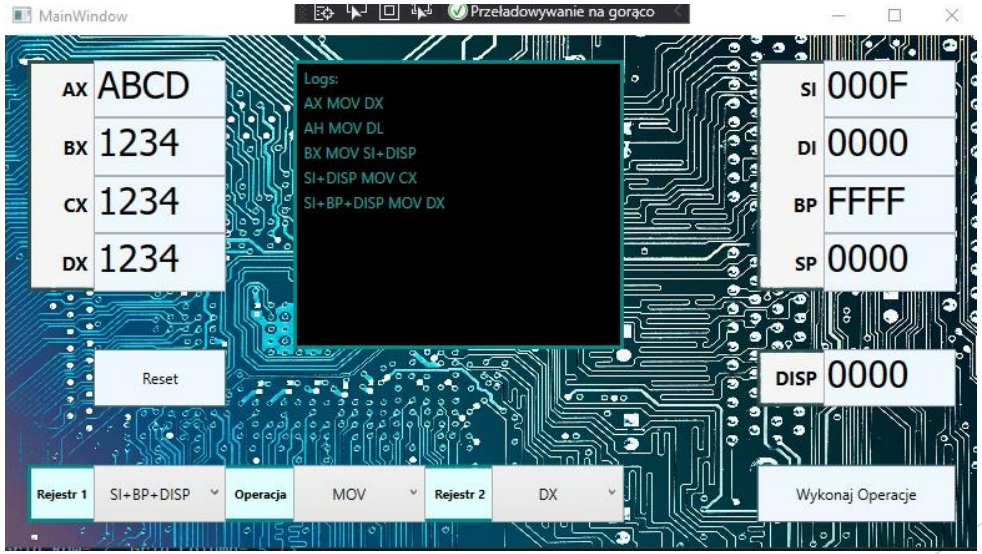
Aby potwierdzić ze operacja MOV z poprzedniej strony zadziałała teraz ja odwrocimy I dane z pamięci pod tym samym adresem segmentu danych jak porzednio zapiszemy w rejestrze CX. Jak widzimy dane w CX zostały poprawnie nadpisane.



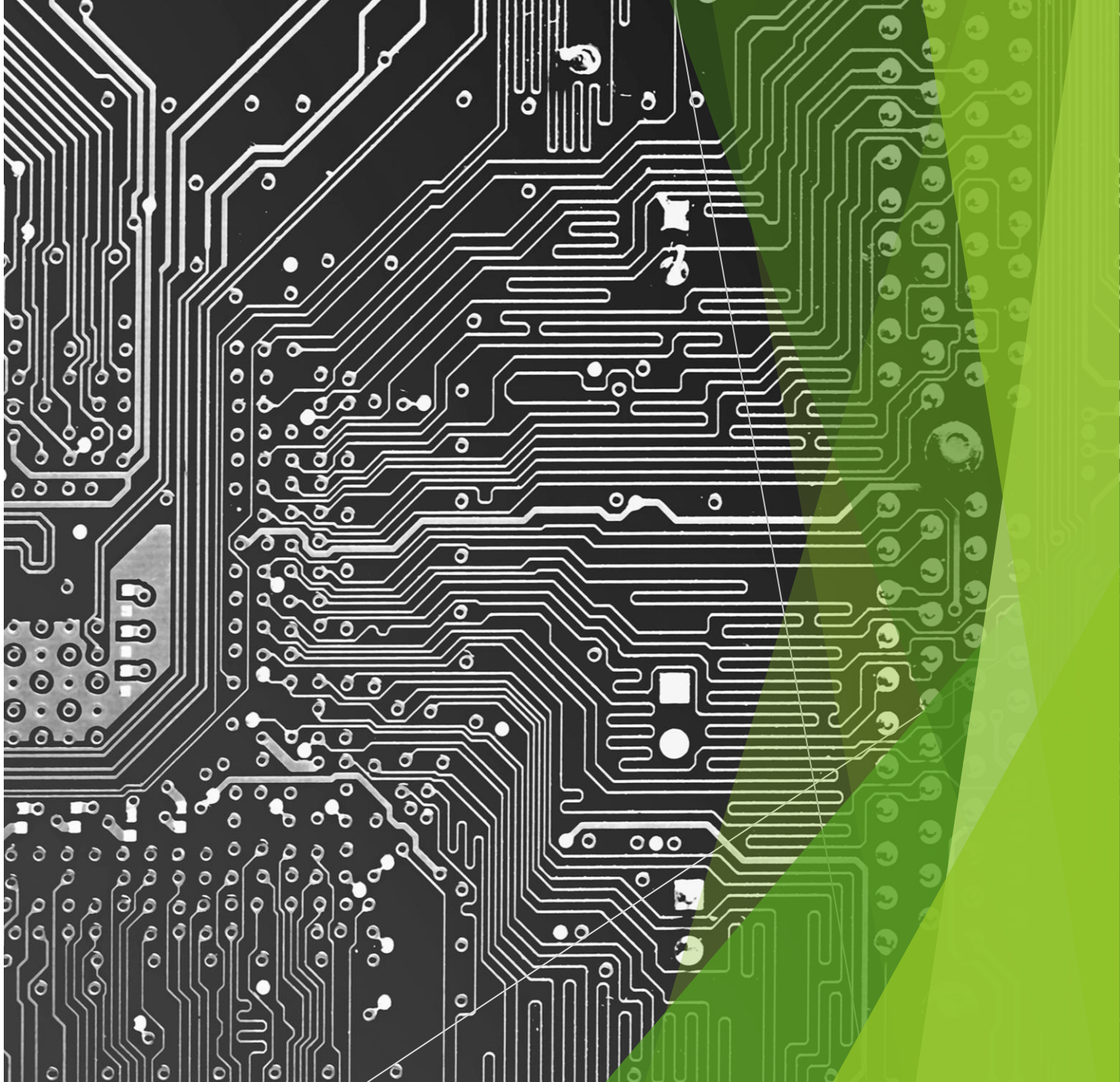
Na ostatnim obrazku widzimy przykład tego co sie stanie gdy adres segmentu danych jest wyższy niż ich ilość w pamięci(65536).

Tylko ostatnie 4 bity będą brane pod uwagę dlatego w tym wypadku SI(000F) + (BP(FFFF) + DISP(0000) da nam segment danych o adresie 000F czyli ten sam ktorego używalismy w poprzednim przykładzie.

Jak widzimy dane zapisane do rejestru DX są takie same jak w CX co to potwierdza.



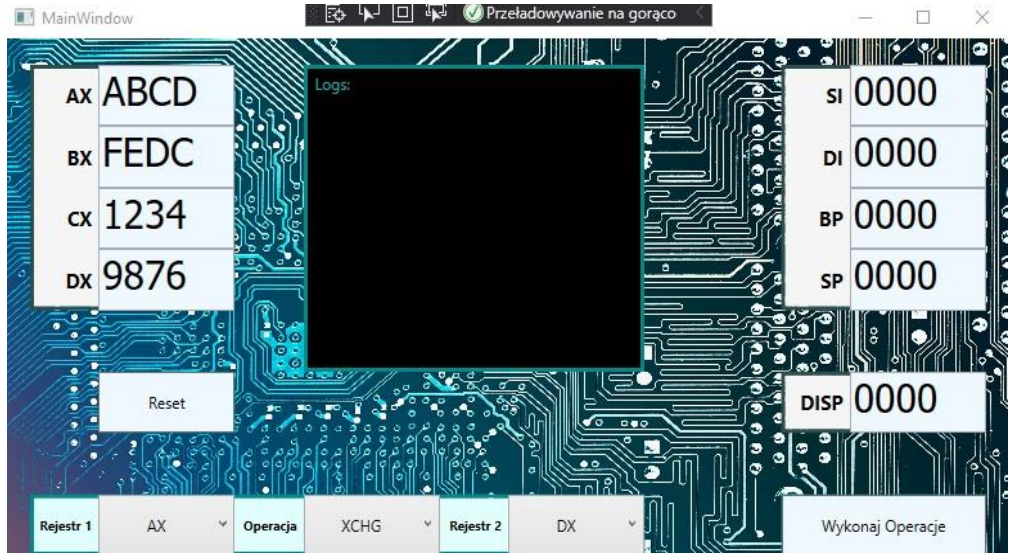
Komenda XCHG



Komenda XCHG pozwala nam na zamienienie danych pomiędzy dwoma rejestrami albo rejestrem I pamięcią.

Na pierwszym obrazku pokazane są dane początkowe.

Na drugim widzimy rezultat po wykonaniu komendy XCHG na rejestrach AX oraz DX. Jak widzimy danych w nich zawarte zamieniły się miejscami.



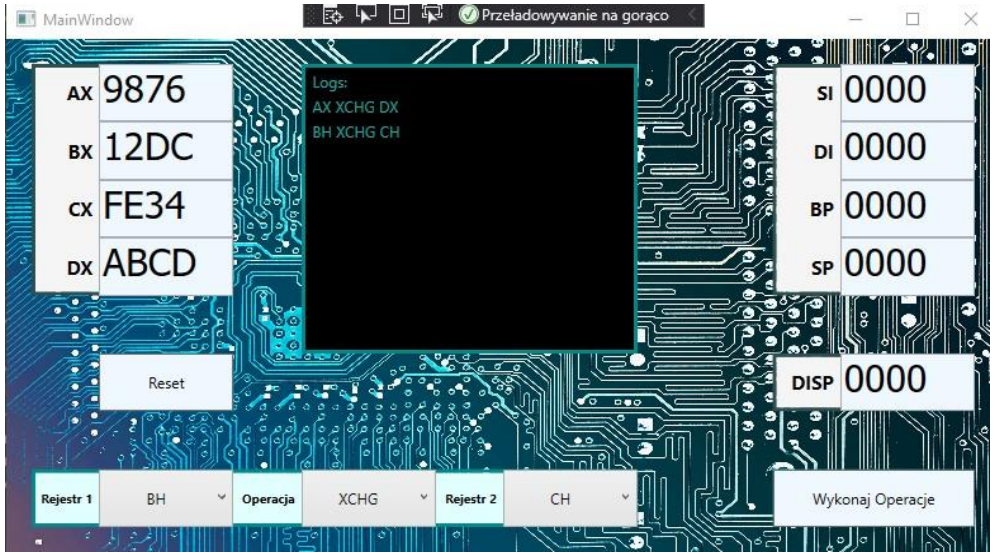
Podobnie jak MOV, XCHG również można użyć na połówkach rejestrów. Na pierwszym obrazku mamy przykład gdzie dane w BH zostały zamienione z danymi w CH

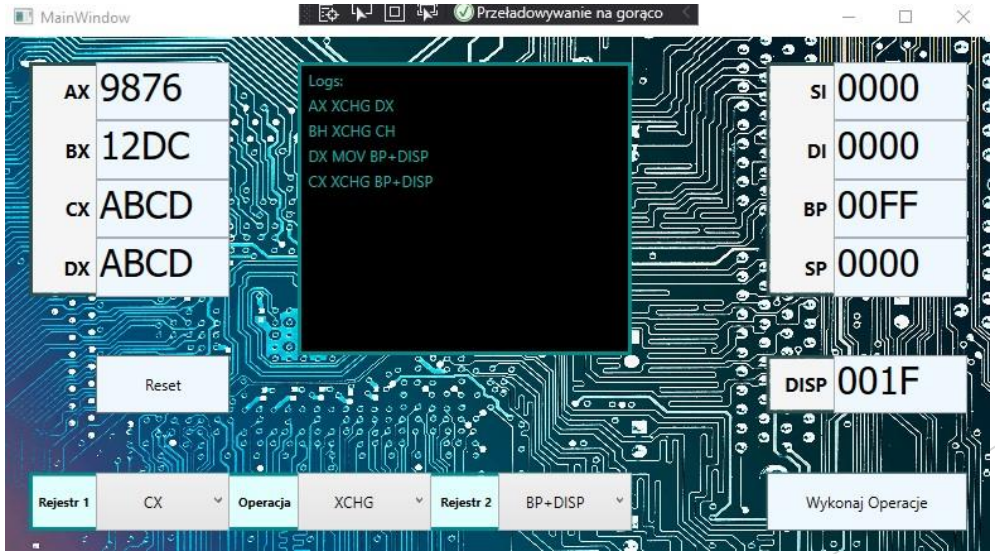
No drugim obrazku mamy przyklad wykorzystania XCHG na pamięci.

Tym razem dla odmiany uzyjemy indeksu bazowego(BP)

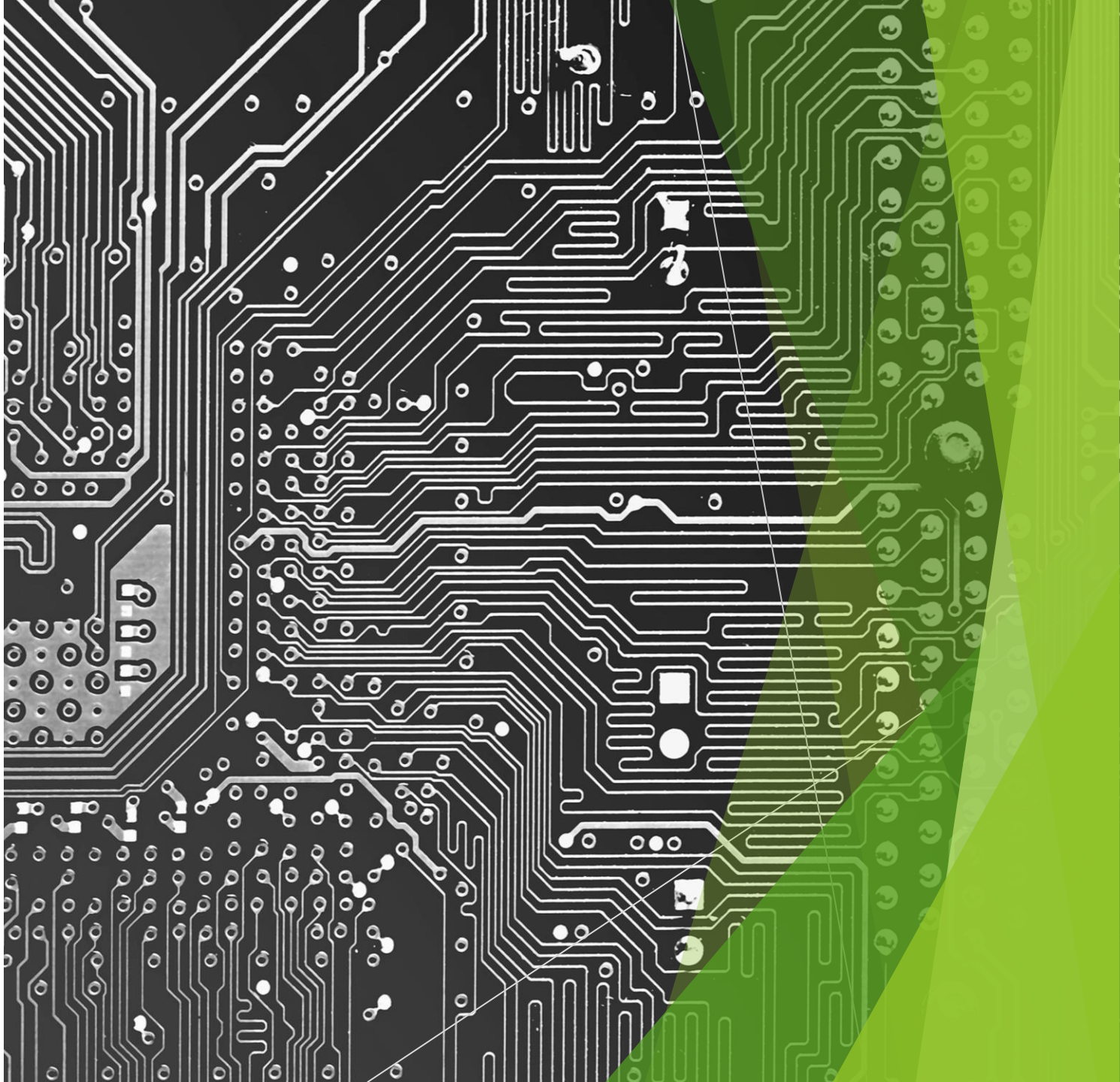
Najpierw za pomoca MOV zapisalismy pod tym adresem dane z rejestru DX.

Następnie za pomocą XCHG zamienilismy dane z rejestru CX z danymi w pamięci. Dane w CX po operacji są takie same jak w DX co potwierdza poprawne wykonanie operacji.





Komenda PUSH



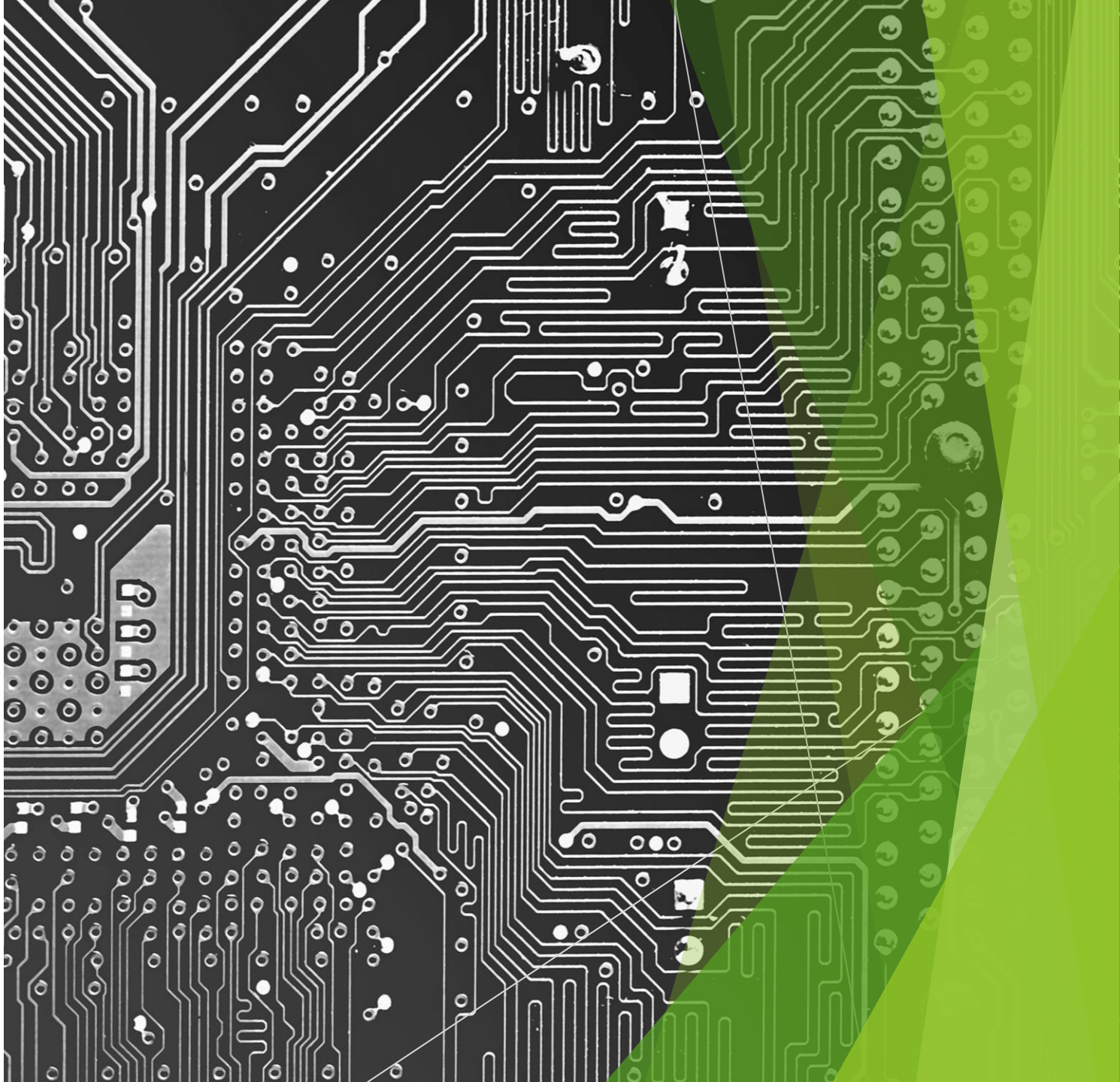
Komenda PUSH pozwala nam zapisywać dane na stosie.

W przeciwieństwie do zapisywania danych w pamięci, tutaj nie musimy podawac adresu. Dane zawsze bedą lądowac na szczycie stosu.

Na pierwszym obrazku mamy przykład wykonania operacji PUSH z rejestrem CX. Jak widzimy Rejest SP zmienił swoją wartość, reprezentuje on adres szczytu naszego stosu.

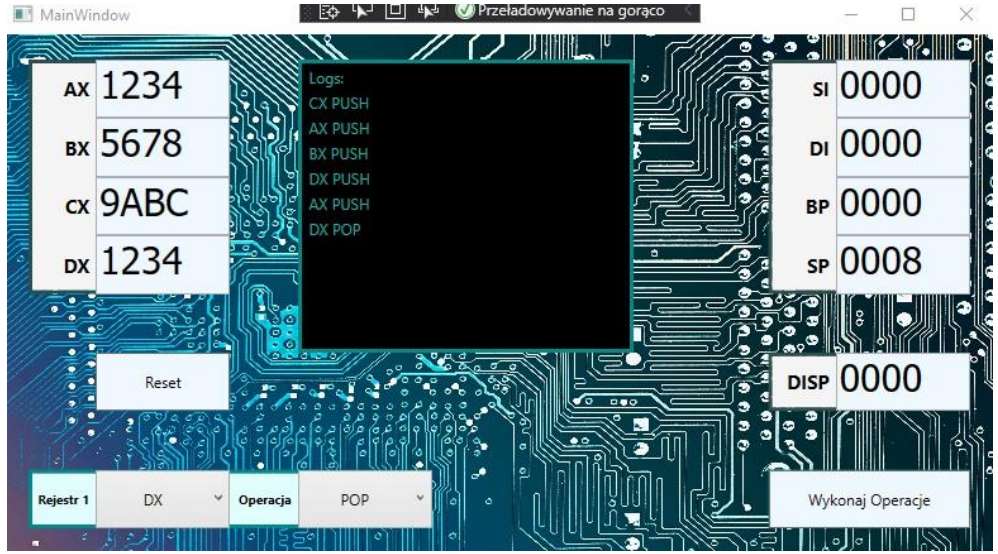
Na drugim obrazku przyklad jak jego wartość rosnie wraz z dodawaniem kolejnych danych na stos.

Komenda POP



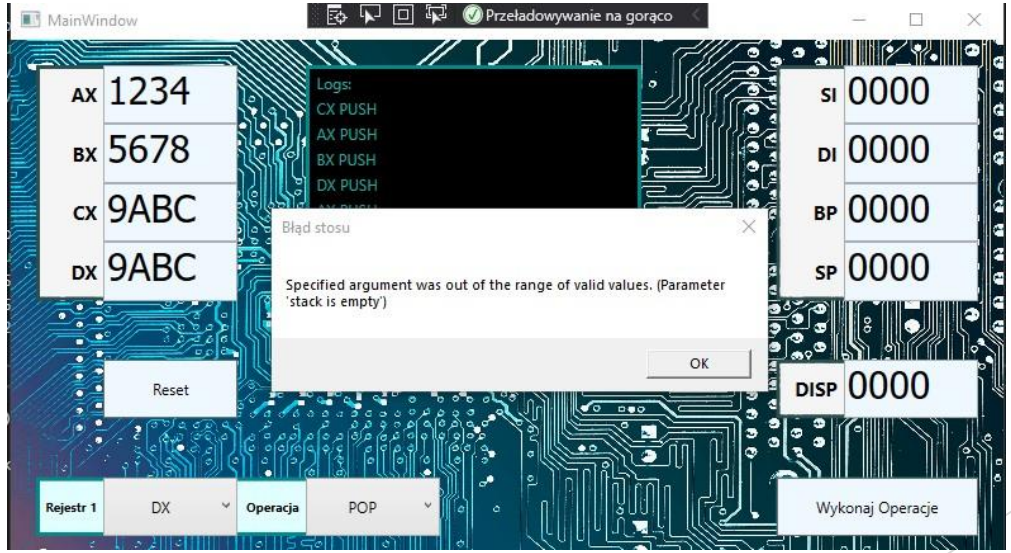
Komenda POP pozwala nam wydobyć dane ze stosu. Wywołanie jej zapisze dane ze szczytu stosu pod wskazanym rejestrem oraz usunie je ze stosu.

Na pierwszym screenie mamy przykład jej użycia. Dane ze stosu zostały zapisane w rejestrze DX.

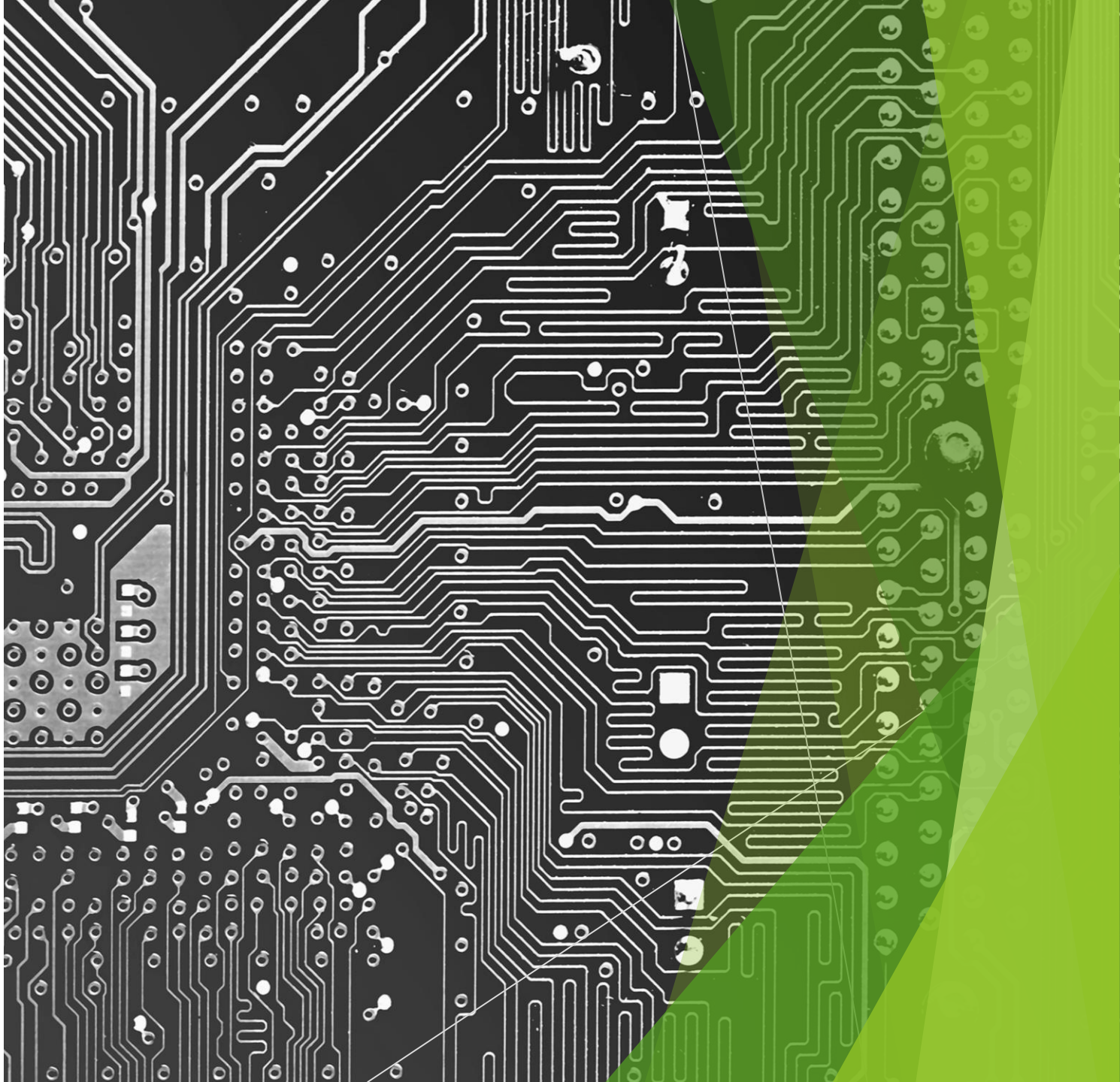


Na drugim obrazku mamy przykład tego co się wydarzy gdy sprobójemy wykorzystać komende POP na pustym stosie. Jako że taka operacja jest niemożliwa dostaniemy komunikat z błędem.

Podobną wiadomość dostaniemy przy operacji PUSH gdy przekroczymy maksymalny rozmiar stosu(65536)



Walidacja oraz Reset

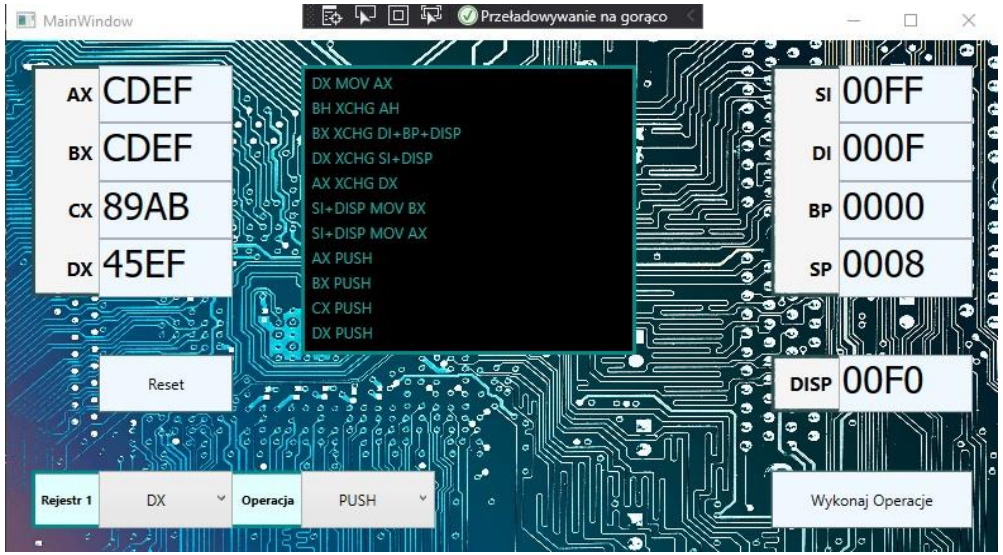


Przycisk “Reset” pozwala zresetowac wszystkie dane.

Warto brać pod uwagę że zresetowane zostaną rownież pamięć oraz stos. Wyczyszczeniu ulegnie także log.

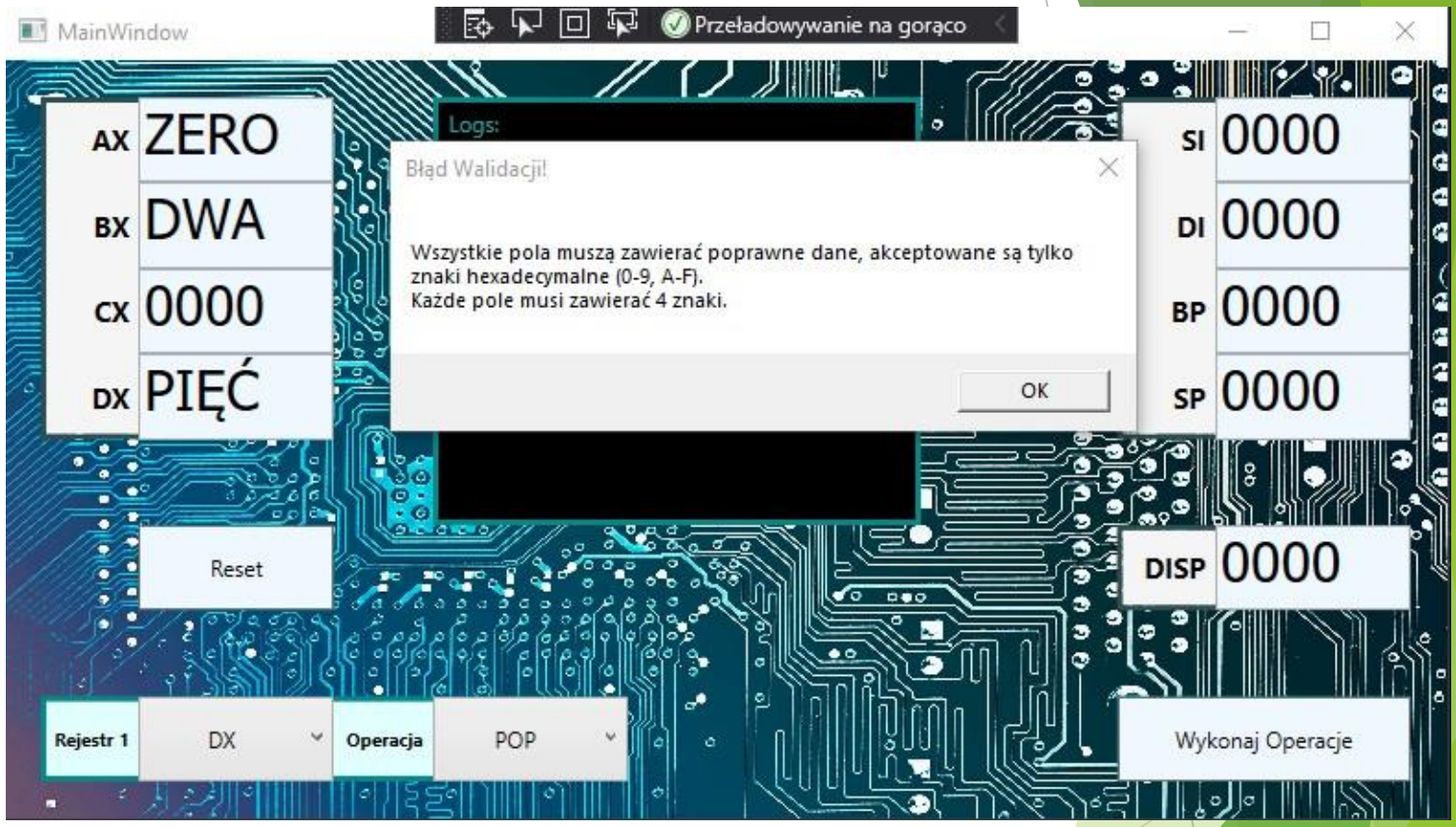
Na pierwszym obrazku przykład przed nacisnięciem przycisku “Reset”

Na drugim po wykonaniu tej operacji



Po wcisnięciu przycisku “Wykonaj Operacje” zanim jakakolwiek operacja zostanie wykonana przeprowadzana jest walidacja danych w rejestrach.

Jeśli ktoryś z nich nie zawiera danych w poprawnym formacie (hexadecymalne znaki, 4 na rejestr) zostanie wyswietlony komunikat z błędem jak na załączonym obrazku.



Link do repozytorium

https://github.com/Kejom/Intel8086