



XWooCaashX /  
Ask7



<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security



Ask7 / README.md



XWooCaashX Update README.md

8468257 · 1 minute ago



256 lines (210 loc) · 9.35 KB

Preview

Code

Blame



Raw



# Odejmowanie w SMS32v50



Autor: [Łukasz Kopański]



## Treść projektu

Pisemne odejmowanie w systemie dwójkowym i dziesiętnym. Wyboru systemu użytkownik dokonuje na początku działania programu. Liczby wprowadzane są z klawiatury, a wynik wyświetla się na wyświetlaczu VDU. Zalecane jest także wyświetlanie wprowadzanych liczb. Dodawane liczby mają 1-10 cyfr. Jeżeli podczas wprowadzania zostanie podane więcej niż 10 cyfr, to fakt ten zostanie zasygnalizowany na sygnalizacji świetlnej. Podobnie należy zasygnalizować przekroczenie 10 cyfr w wynikowej liczbie.



## Spis treści

1. [Treść projektu](#)
2. [Instrukcja uruchomienia](#)
3. [Opis działania programu i założenia](#)
4. [Logika działania \(Schemat blokowy\)](#)
5. [Kluczowe algorytmy](#)
  - o [Inicjalizacja systemu](#)
  - o [Wprowadzanie liczb](#)
  - o [Mechanizm odejmowania](#)
  - o [Obsługa pożyczki](#)

- [Konwersja i wyświetlenie wyniku](#)
- [Obsługa błędów](#)
- 6. [Szczegóły techniczne](#)
- 7. [Testowanie](#)
  - [Przypadek 1: Odejmowanie dziesiętne](#)
  - [Przypadek 2: Odejmowanie dziesiętne \(wynik ujemny\)](#)
  - [Przypadek 3: Odejmowanie binarne](#)
  - [Przypadek 4: Odejmowanie binarne \(wynik ujemny\)](#)
  - [Przypadek 5: Błąd przekroczenia](#)
- 8. [🎮 Symulator w HTML](#)
- 9. [Dokumentacja](#)
- 10. [Autor i wersja](#)

## Instrukcja uruchomienia

---

Aby uruchomić program w emulatorze **SMS32v50**, otwórz odpowiedni plik w programie:

- [kod\\_main.asm](#) – krótka wersja kodu z krótkimi komentarzami.
- [kod\\_dok.asm](#) – wersja z pełną dokumentacją, komentarzami i podziałem na sekcje.

## Opis działania programu i założenia

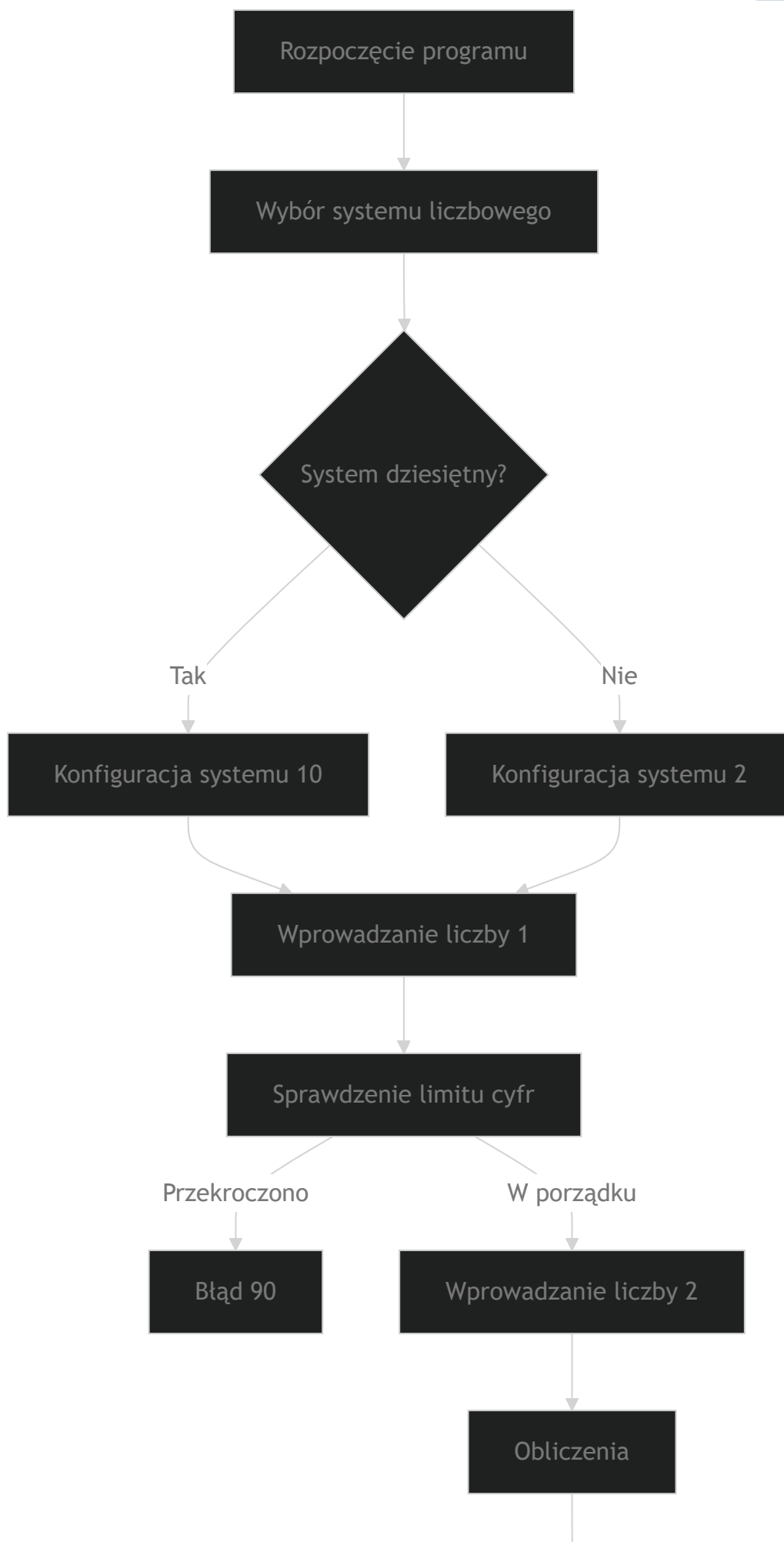
---

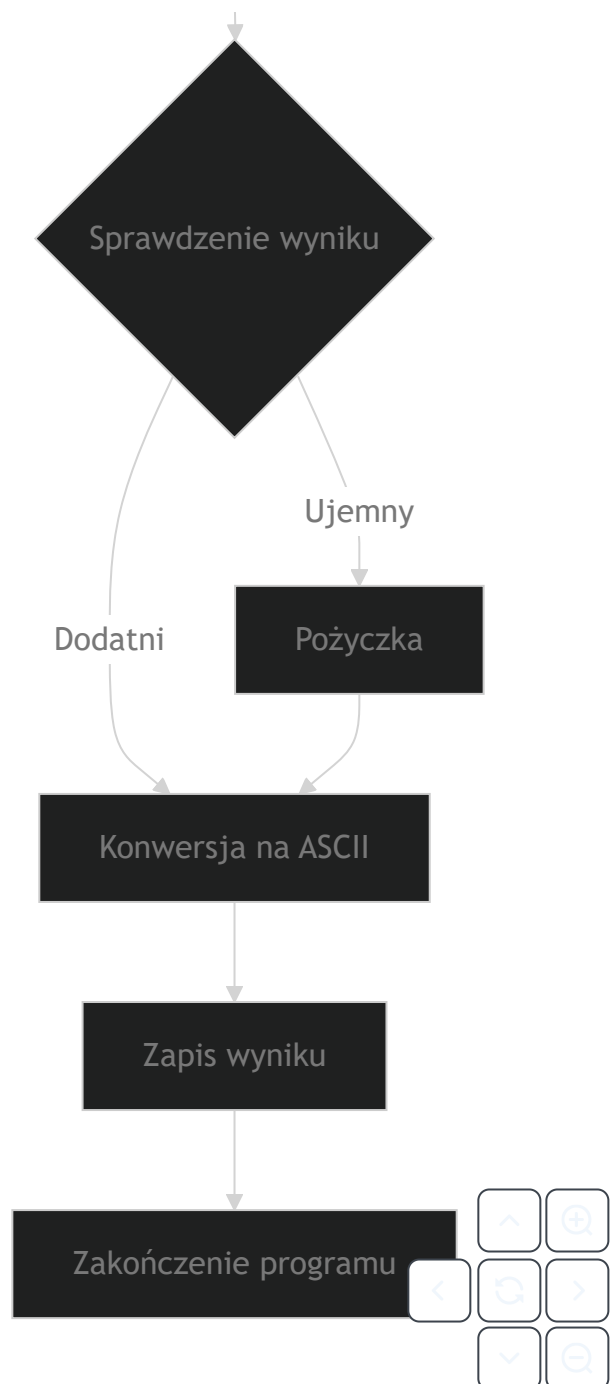
Program realizuje odejmowanie liczb w systemie dziesiętnym lub binarnym. Główne założenia i funkcjonalność:

1. **Wybór systemu liczbowego** - Użytkownik na początku działania programu wybiera system dziesiętny lub binarny.
2. **Wprowadzanie liczb** - Program odczytuje liczby z klawiatury i przechowuje je w pamięci.
3. **Obsługa limitu cyfr** - Maksymalna długość liczby to 10 cyfr, przekroczenie limitu sygnalizowane jest diodą LED.
4. **Mechanizm odejmowania** - Realizowany cyfrowo, z obsługą pożyczki w przypadku wartości ujemnych.
5. **Wyświetlanie wyników** - Wynik operacji jest konwertowany na ASCII i prezentowany na wyświetlaczu VDU.
6. **Obsługa błędów** - Program sygnalizuje przekroczenie limitów cyfr oraz błędy obliczeniowe diodą LED.









## Kluczowe algorytmy

### **1** Inicjalizacja systemu

Resetuje flagi i rejestry, a następnie pobiera wybór systemu liczbowego.

```
CLO          ; Czyścimy pamięć programu
IN 00        ; Oczekiwanie na wybór użytkownika: '1' - system dziesiętny
CMP AL, 31    ; Sprawdzamy, czy wciśnięto '1' (ASCII 31)
JZ system_dziesiętny ; Jeśli tak, przechodzimy do systemu dziesiętnego
MOV AL, 2     ; W przeciwnym razie ustawiamy system binarny
```



```
MOV [BC], AL    ; Zapisujemy wybór użytkownika w pamięci
JMP wprowadz_liczby ; Przechodzimy do etapu wprowadzania liczb
```

## 2 Wprowadzanie liczb

Program pobiera cyfry od użytkownika, sprawdza limit długości i zapisuje dane w pamięci.

- Odczyt znaków z klawiatury
- Sprawdzenie, czy użytkownik nacisnął ENTER (oznacza koniec liczby)
- Kontrola, czy liczba nie przekroczyła 10 cyfr
- Zapisanie cyfry w odpowiedniej komórce pamięci
- Obsługa drugiej liczby po zakończeniu wpisywania pierwszej

```
petla_wprowadzania:
IN 00          ; Pobieramy cyfrę z klawiatury
CMP AL, 0D     ; Sprawdzamy, czy wciśnięto ENTER (ASCII 0D)
JZ krok       ; Jeśli tak, przechodzimy do obliczeń
CMP CL, DL     ; Sprawdzamy, czy liczba nie przekroczyła 10 cyfr
JZ blad_przekroczenia ; Jeśli tak, zgłaszamy błąd
MOV [CL], AL   ; Zapisujemy cyfrę w pamięci
INC CL        ; Przechodzimy do następnej cyfry
JMP petla_wprowadzania ; Powtarzamy pętlę dla kolejnej cyfry
```



## 3 Mechanizm odejmowania

Główna pętla wykonuje odejmowanie cyfrowe, obsługuje sytuacje wymagające pożyczki i sprawdza poprawność obliczeń.

- Pobranie cyfry z drugiej liczby
- Przejście do pierwszej liczby i wykonanie odejmowania
- Obsługa sytuacji, w której wynik jest ujemny (pożyczka)
- Przejście do kolejnej cyfry i kontynuowanie obliczeń

```
petla_obliczen:
CMP CL, CF     ; Sprawdzamy, czy zakończono obliczenia
JZ koniec_programu ; Jeśli tak, kończymy program
MOV AL, [CL]   ; Pobieramy cyfrę z drugiej liczby
CMP AL, 20     ; Sprawdzamy, czy to spacja (ASCII 20)
JZ koniec_programu ; Jeśli tak, kończymy program
SUB CL, 10     ; Przechodzimy do pierwszej cyfry
MOV BL, [CL]   ; Pobieramy cyfrę z pierwszej liczby
ADD CL, 20     ; Przechodzimy do miejsca wyniku
SUB BL, AL     ; Odejmujemy drugą cyfrę od pierwszej
```



```
PUSH BL      ; Zapisujemy wynik na stosie
POP AL       ; Pobieramy wynik ze stosu
CMP AL, 0    ; Sprawdzamy, czy wynik jest ujemny
JS pożyczka  ; Jeśli tak, przechodzimy do obsługi pożyczki
JMP wypisz_wynik ; Jeśli nie, przechodzimy do wyświetlenia wyniku
```

## 4 Obsługa pożyczki

Jeśli wynik odejmowania jest ujemny, program wykonuje pożyczkę i koryguje wynik.

```
pożyczka:
ADD AL, DL    ; Dodajemy wartość systemu (2 lub 10), żeby poprawić wynik
DEC CL       ; Cofamy się do poprzedniej cyfry
MOV BL, [CL]  ; Pobieramy wartość
DEC BL       ; Odejmujemy 1 (pożyczka)
MOV [CL], BL  ; Zapisujemy nową wartość
ADD CL, 10    ; Wracamy do właściwej cyfry
```

## 5 Konwersja i wyświetlenie wyniku

Po zakończeniu obliczeń wynik jest konwertowany na ASCII i wyświetlany na ekranie.

```
ADD AL, 30    ; Konwersja wyniku na ASCII
MOV [CL], AL  ; Zapisujemy wynik w pamięci
```

## 6 Obsługa błędów

Jeżeli użytkownik przekroczy limit cyfr, program wyświetli błąd i zapali diodę LED.

```
bład_przekroczenia:
MOV AL, 90    ; Przypisanie do AL wartości 90 (czerwone światło)
OUT 01        ; Wyświetlenie czerwonego światła na wyjściu 01
```

## Szczegóły techniczne

- Rejestry:
  - AL/BL – operacje na cyfrach
  - CL/DL – wskaźniki i liczniki
- Pamięć:
  - C1-CB – pierwsza liczba
  - D1-DB – druga liczba

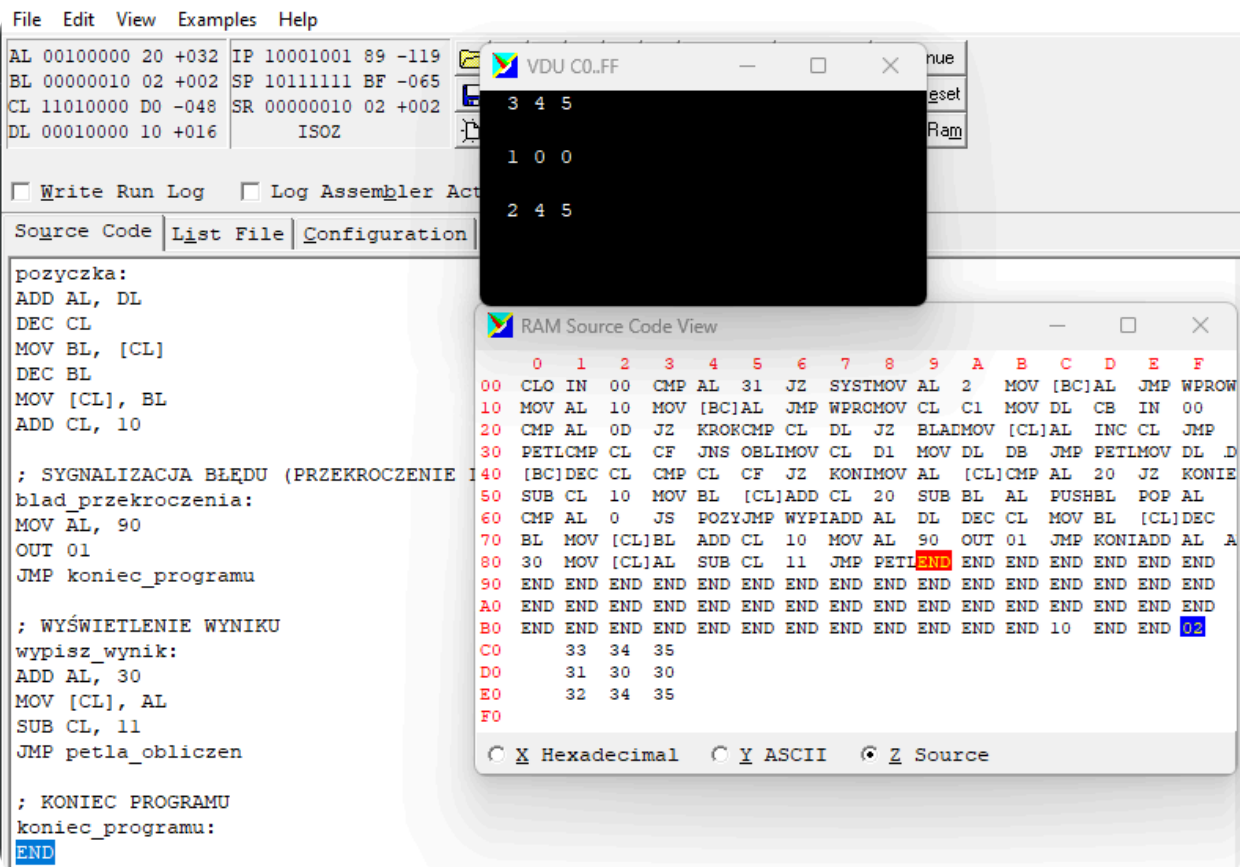


- E0-EF – wynik
- Flagi:
  - ZF – wykrywanie ENTER
  - CF – kontrola pożyczek
  - SF – wykrywanie wyników ujemnych

## Testowanie

### ✓ Przepadek 1: Odejmowanie dziesiętne

Wybierz system: 1  
 Liczba 1: 345  
 Liczba 2: 100  
 Wynik: 245

The screenshot shows an assembly program in a text editor and its execution results in two windows.

**Assembly Program:**

```

File Edit View Examples Help
AL 00100000 20 +032 IP 10001001 89 -119
BL 00000010 02 +002 SP 10111111 BF -065
CL 11010000 D0 -048 SR 00000010 02 +002
DL 00010000 10 +016 ISOZ

Write Run Log Log Assembler Act
Source Code List File Configuration

pozyczka:
ADD AL, DL
DEC CL
MOV BL, [CL]
DEC BL
MOV [CL], BL
ADD CL, 10

; SYGNALIZACJA BŁĘDU (PRZEKROCZENIE 1
blad_przekroczenia:
MOV AL, 90
OUT 01
JMP koniec_programu

; WYŚWIETLENIE WYNIKU
wypisz_wynik:
ADD AL, 30
MOV [CL], AL
SUB CL, 11
JMP petla_obliczen

; KONIEC PROGRAMU
koniec_programu:
END
  
```

**VDU C0..FF Window:**

```

3 4 5
1 0 0
2 4 5
  
```

**RAM Source Code View Window:**

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	CLO	IN	00	CMP	AL	31	JZ	SYSTM	MOV	AL	2	MOV	[BC]	AL	JMP	WPROW
10	MOV	AL	10	MOV	[BC]	AL	JMP	WPRCM	MOV	CL	C1	MOV	DL	CB	IN	00
20	CMP	AL	0D	JZ	KROK	CMP	CL	DL	JZ	BLAD	MOV	[CL]	AL	INC	CL	JMP
30	PETL	CMP	CL	CF	JNS	OBLI	MOV	CL	D1	MOV	DL	DB	JMP	PETL	MOV	DL
40	[BC]	DEC	CL	CMP	CL	CF	JZ	KONI	MOV	AL	[CL]	CMP	AL	20	JZ	KONIE
50	SUB	CL	10	MOV	BL	[CL]	ADD	CL	20	SUB	BL	AL	PUSH	BL	POP	AL
60	CMP	AL	0	JS	POZY	JMP	WYPI	ADD	AL	DL	DEC	CL	MOV	BL	[CL]	DEC
70	BL	MOV	[CL]	BL	ADD	CL	10	MOV	AL	90	OUT	01	JMP	KONI	ADD	AL
80	30	MOV	[CL]	AL	SUB	CL	11	JMP	PETL	END	END	END	END	END	END	END
90	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
A0	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
B0	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END	END
C0		33	34	35												
D0		31	30	30												
E0		32	34	35												
F0																

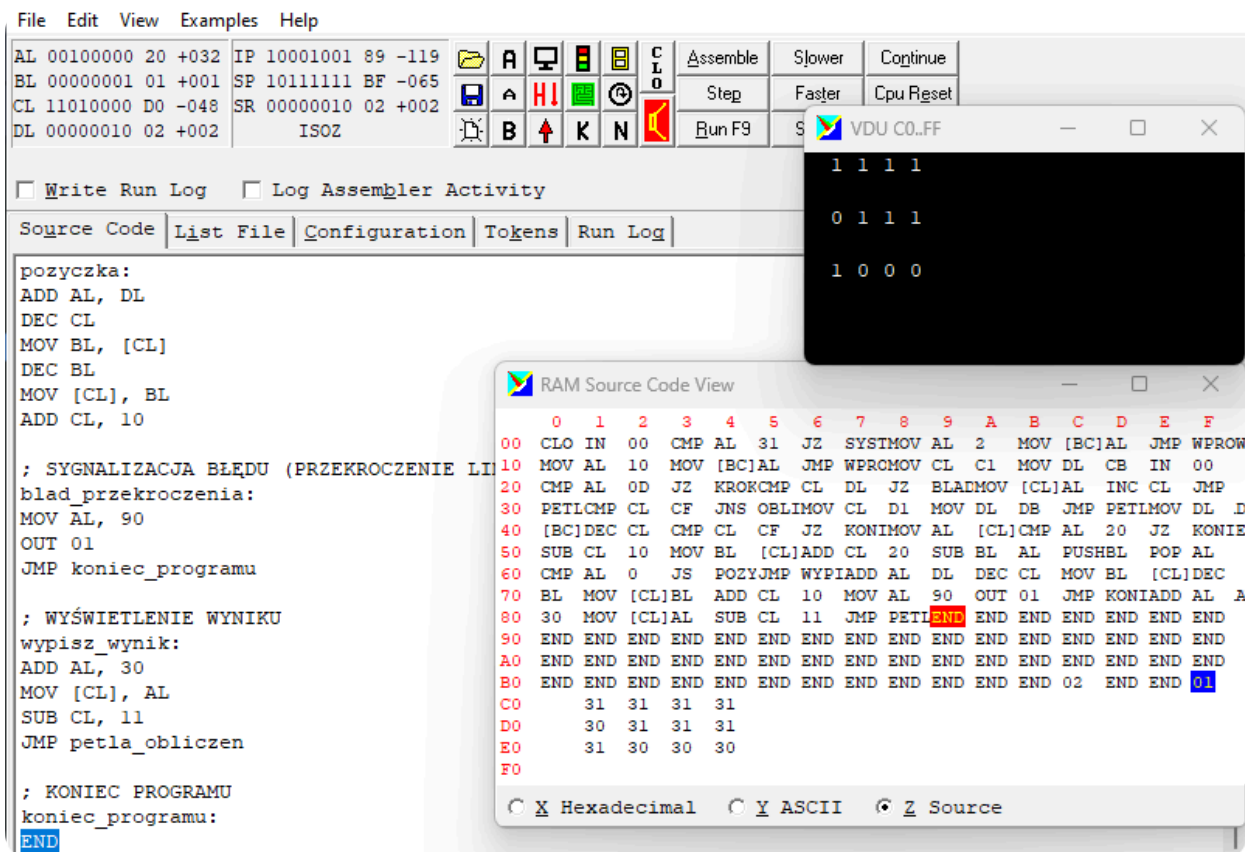
Hexadecimal ASCII Source

### ✓ Przepadek 2: Odejmowanie dziesiętne (przy wyniku na minuse)

Wybierz system: 1  
 Liczba 1: 8  
 Liczba 2: 9  
 Wynik: Błąd







## ✓ Przypadek 4: Odejmowanie binarne (przy wyniku na minusie)

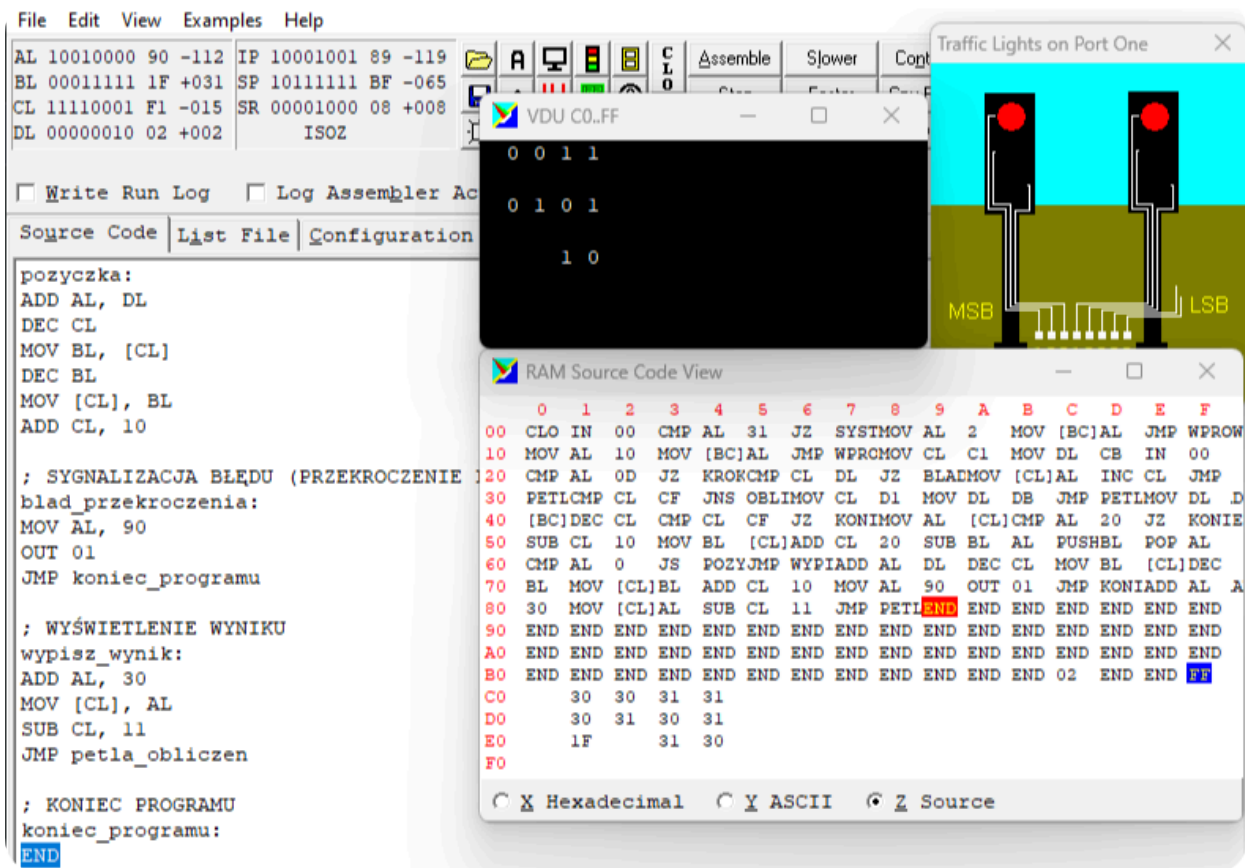
Wybierz system: 0

Liczba 1: 0011 ---> 10-tny: 3

Liczba 2: 0101 ---> 10-tny: 5

Wynik: 1000 ---> 10-tny: 5

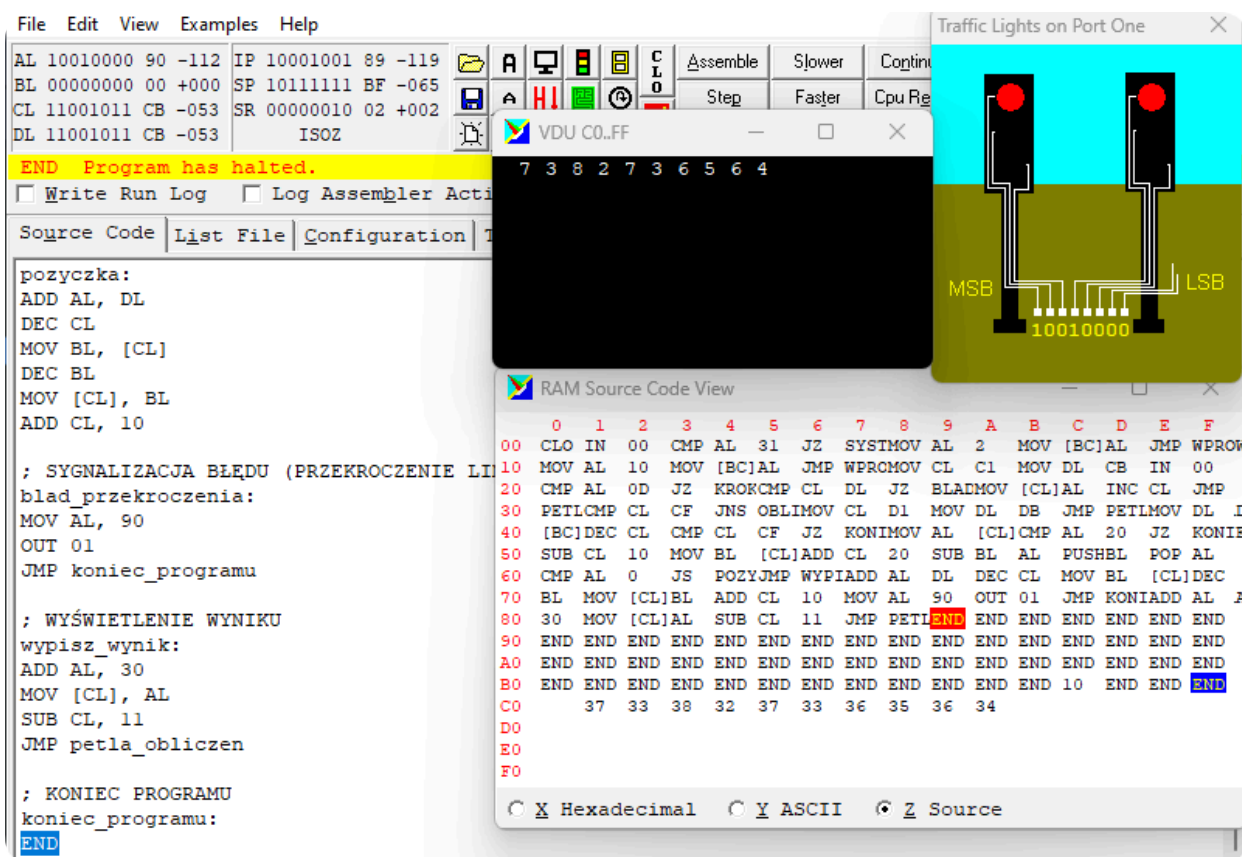




## 🚨 Przypadek 5: Błąd przekroczenia

Wprowadzasz: 7382736564 ----> Brak ostatniej liczby ze względu na przekroczenie limitu  
SYSTEM LED: Czerwony (kod 90)





## Symulator w HTML tego zadania (z podobną zasadą działania)

Zrobiłem to w ramach zajawki, aby lepiej zrozumieć działanie systemu.


[Dostępne tutaj na stronie internetowej.](#)

## Dokumentacja


Pełna dokumentacja kodu dostępna w pliku [dokumentacja](#).

## Zaległe prace z labolatorii

[Dostępne tutaj na stronie internetowej.](#)

 Autor: [Łukasz Kopański]

 Nr albumu: 91667  Wersja: 1.0

 Ostatnia aktualizacja: 2025-01-29

