

zad 1

wynik działania i podgląd debugera

C:\MIKRO\ZAD1>zad1

Piotr Heinzelman

Wydz. Elektryczny, kierunek "Informatyka Stosowana"

C:\MIKRO\ZAD1>debug zad1.exe

-u

12D8:0000 B8D312	MOV	AX, 12D3
12D8:0003 50	PUSH	AX
12D8:0004 1F	POP	DS
12D8:0005 8D160000	LEA	DX, [0000]
12D8:0009 B409	MOV	AH, 09
12D8:000B CD21	INT	21
12D8:000D 8D161700	LEA	DX, [0017]
12D8:0011 CD21	INT	21
12D8:0013 B44C	MOV	AH, 4C
12D8:0015 B000	MOV	AL, 00
12D8:0017 CD21	INT	21
12D8:0019 D380FB02	ROL	WORD PTR [BX+SI+02FB], CL
12D8:001D 7316	JNB	0035
12D8:001F 2E	CS:	
12D8:0020 8ABF7101	MOU	BH, [BX+0171]

-

12A3:0270 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!
-d

12A3:0280 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:0290 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:02A0 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:02B0 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:02C0 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:02D0 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:02E0 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 21 21 STACK!!!STACK!!!

12A3:02F0 53 54 41 43 4B 21 21 21-53 54 41 43 4B 21 00 00 STACK!!!STACK!..

-d

12A3:0300 0A 0D 0A 0D 50 69 6F 74-72 20 48 65 69 6E 7A 65Piotr Heinze

12A3:0310 6C 6D 61 6E 0A 0D 24 57-79 64 7A 2E 20 45 6C 65 lman..\$Wydz. Ele

12A3:0320 6B 74 72 79 63 7A 6E 79-2C 20 6B 69 65 72 75 6E ktryczny, kierun

12A3:0330 65 6B 20 22 49 6E 66 6F-72 6D 61 74 79 6B 61 20 ek "Informatyka

12A3:0340 53 74 6F 73 6F 77 61 6E-61 22 0A 0D 24 0D 0A 24 Stosowana"...\$...\$

12A3:0350 B8 D3 12 50 1F 8D 16 00-00 B4 09 CD 21 8D 16 17 ...P.....!....

12A3:0360 00 CD 21 B4 4C B0 00 CD-21 D3 80 FB 02 73 16 2E ..!..L...!....s..

12A3:0370 8A BF 71 01 AC 8A D8 56-8B F7 2B F3 1E 06 1F F3 ..q....V...+....

-r

AX=0000 BX=0000 CX=0269 DX=0000 SP=0200 BP=0000 SI=0000 DI=0000

DS=12A3 ES=12A3 SS=12B3 CS=12D8 IP=0000 NV UP EI PL NZ NA PO NC

12D8:0000 B8D312 MOV AX, 12D3

-

zad 2

wynik działania i podgląd debugera

C:\MIKRO\ZAD1>zad2

Piotr Heinzelman

Wydz. Elektryczny, kierunek "Informatyka Stosowana"

23:49:20 23-12-2023

C:\MIKRO\ZAD1>zad2

Piotr Heinzelman

Wydz. Elektryczny, kierunek "Informatyka Stosowana"

23:49:23 23-12-2023

C:\MIKRO\ZAD1>

C:\MIKRO\ZAD1>debug zad2.exe

-r

AX=0000 BX=0000 CX=02FA DX=0000 SP=0200 BP=0000 SI=0000 DI=0000
DS=12A3 ES=12A3 SS=12B3 CS=12DA IP=0000 NV UP EI PL NZ NA PO NC
12DA:0000 B8D312 MOV AX,12D3

-u

12DA:0000 B8D312	MOV	AX,12D3
12DA:0003 50	PUSH	AX
12DA:0004 1F	POP	DS
12DA:0005 8D160000	LEA	DX,[0000]
12DA:0009 B409	MOV	AH,09
12DA:000B CD21	INT	21
12DA:000D 8D161700	LEA	DX,[0017]
12DA:0011 CD21	INT	21
12DA:0013 8D364D00	LEA	SI,[004D]
12DA:0017 B402	MOV	AH,02
12DA:0019 CD1A	INT	1A
12DA:001B 52	PUSH	DX
12DA:001C 51	PUSH	CX
12DA:001D 51	PUSH	CX
12DA:001E 5A	POP	DX
12DA:001F 86D6	XCHG	DL,DH

--

zad 3

wynik działania i podgląd debugera

C:\MIKRO\ZAD3>zad3

GREETING PROFESSOR FALKEN

Enter first number A < 9999:9
second number:6
NWD: 0003
C:\MIKRO\ZAD3>zad3

GREETING PROFESSOR FALKEN

Enter first number A < 9999:120
second number:3
NWD: 0003
C:\MIKRO\ZAD3>zad3

GREETING PROFESSOR FALKEN

Enter first number A < 9999:120

C:\MIKRO\ZAD3>debug zad3.exe

-u

```
11E8:0000 B8E211      MOV     AX,11E2
11E8:0003 50          PUSH    AX
11E8:0004 1F          POP     DS
11E8:0005 8D160000    LEA     DX,[0000]
11E8:0009 E88C00      CALL    0098
11E8:000C E8D900      CALL    00E8
11E8:000F 93          XCHG    BX,AX
11E8:0010 8D163E00    LEA     DX,[003E]
11E8:0014 E88100      CALL    0098
11E8:0017 E8CE00      CALL    00E8
11E8:001A E8EF00      CALL    010C
11E8:001D 3BD3        CMP     DX,BX
11E8:001F 7504        JNZ    0025
```

-r

AX=0000 BX=0000 CX=03C8 DX=0000 SP=0200 BP=0000 SI=0000 DI=0000
DS=11B2 ES=11B2 SS=11C2 CS=11E8 IP=0000 NV UP EI PL NZ NA PO NC
11E8:0000 B8E211 MOV AX,11E2

-g

GREETING PROFESSOR FALKEN

Enter first number A < 9999:9999

second number:2

NWD: 0001

Program terminated normally

-d

```
11E8:0000  B8 E2 11 50 1F 8D 16 00-00 E8 8C 00 E8 D9 00 93  ...P.....
11E8:0010  8D 16 3E 00 E8 81 00 E8-CE 00 E8 EF 00 3B D3 75  ..>.....;u
11E8:0020  04 3B C1 74 63 3B D3 7F-09 7C 04 3B C1 7D 03 87  .;.tc;...!.;}..
11E8:0030  D3 91 80 F9 00 74 1E 3A-C1 7D 18 80 FC 00 75 0F  ....t.:};...u.
11E8:0040  80 C4 0A 80 FA 00 75 05-80 C2 0A FE CE FE CA FE  .....u.....
11E8:0050  CC 04 0A 2A C1 80 FD 00-74 15 3A E5 7D 0F 80 C4  ...*....t.:}...
11E8:0060  0A 80 FA 00 75 05 FE CE-80 C2 0A FE CA 2A E5 80  ....u.....*..
11E8:0070  FB 00 74 0B 3A D3 7D 05-80 C2 0A FE CE 2A D3 80  ..t.:}.....*..
```

Zad 3

Odejmowanie BCD (16 bit)
DX:AX - BX:CX

①

BH BL CH CL
DH | DL | AH | AL

CL* = 0?

AL < CL

AL > CL?

Borrow:

AH + 1

AL + 10 = AL + 0AH

No Sub

No Borrow

...
SUB AL, CL

CH = 0? — No Sub

AH > CH — No Borrow

Borrow:

DEC DL

ADD AH, OAH

Sub:

SUB AH, CH

BL = 0? — No Sub

DL > BL — No Borrow

DEC DH

ADD DL, OAH

...

SUB DL, BL

BH = 0?

SUB

DH, BL

No Sub

(2)

odejmowanie BCD ma też przykłady
 miscalculation min. że wynikowe AAS
 obejmuje tylko AL a nie inne rejestrów.

Można zapisywać i odczytywać wielobitowe
 liczby do parci ale kiedy zapis / odczyt
 to zarówno zapisywyć **kilkanaście cyfr**
zejrz!

Aby nie powtarzać się tego samego
 -一样的 od razu na rejestrach.

Sprawdzać czy $A > B$ czyli $DX:AX > BX:CX$.
 jeśli tak to wtedy zmiany wejściowe

$\times CHG DX,BX$

$\times CHG AX,CX$ zainicjuj dwa cykle zegara.

a odejmowanie zmiany dwóch prostych.
 wicze ze $A > B$.

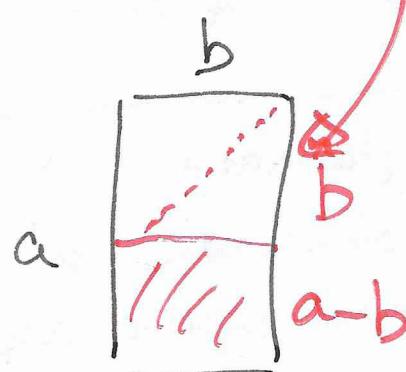
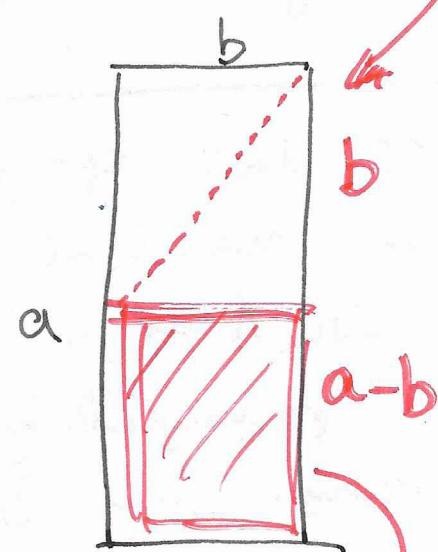
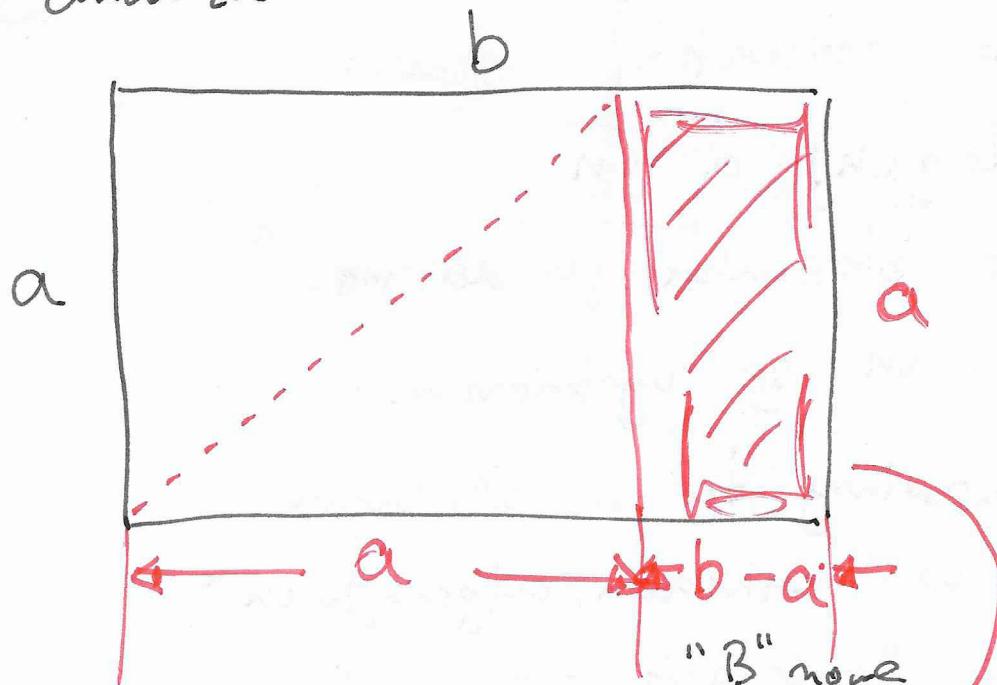
Wyże AAS daje dwa rezultaty na starych
 gątach odejmowania, obojętnie ten problem
 zatrafiające się zauważ ten problem, —
 oznajm go "potyczając" jawnie przed
 odejmowaniem a nie po odejmowaniu.

Generuje maca z BCD to działa przez np.
 Taki przykład:

INPUT (10) \rightarrow Value (HEX) \rightarrow obliczenia ... \rightarrow wyników \rightarrow de-

analiza

③



many 2 dropi alba $\frac{a-b}{b-a}$ i $\frac{b-a}{a-b}$ algo
jáki $a > b \rightarrow a = b \frac{b-a}{a-b}$ i odefinwy a-b



(4)

Złożoność obliczeniowa - jest bardzo niewielka, wynosi $\log(n)$ a \sqrt{n} i zależy od stosunku n do m . Wartość n i m są wymienne:

Ogólnie mówiąc tu o złożoności algorytmu w "cyklach algorytmu" a nie w "cyklach separacji".

Zabawa w BCD daje sporo marnotrawstwa cyklach separacji na cykl algorytmu, gdybyśmy kopiując ali z litą NSB - wówczas ALU - pomagały nam w sposób naturalny. przy BCD mamy możliwości zapisu 2 ALU.

Zadane 6 wykonać w DEC \rightarrow HEX \rightarrow obliczenia \rightarrow HEX \rightarrow DEC \rightarrow PRINT. Wtedy obliczenia cyklu algorytmu zapisz po kilka cykli separacji.

Konieczne 2 RAM dla kilku modyfikacji algorytmu da kilka cykli separacji na jeden cykl algorytmu.