J.NAWROCKI, M. ANTCZAK, H. ĆWIEK, W. FROHMBERG, A. HOFFA, M. KIERZYNKA, S. WĄSIK

ASSEMBLER

WORKING ENVIRONMENT

1. Download the ***masm.zip* file** (*Macro Assembler 6.15* & *Segmented Executable Linker 5.60).*
2. Extract the above archive to the working directory (during classes it is probably C:\Temp).
3. Ważne pliki wykorzystywane podczas kompilacji:
   - *ml.exe, LINK.EXE* – executable files of *Macro Assembler 6.15* and *Segmented Executable Linker 5.60*; they are used during compilation of assembler programs,
   - *clean.bat* – used to remove following files: **.bak*, *prog.exe, prog.obj, prog.map, prog.lst, prog.com* from the current directory,
   - *prog.asm* – source code of the assembler program (*blank.asm* – the template of the program written in assembler that can be filled with instructions solvig some exercice to receive fully working program – it contains the code that is needed to successfuly compile any of the problems provided below),
   - *make.bat* – contains commands that have to be executed to successfully compile assembler program (for example use `make.bat prog` to compile the file *`prog.asm`*).

     ```
     ml /Fe'kod_wynikowy' /Fl'listing_kompilacji' /Fm'mapa_kompilacji'
     /Fo'kod_przejściowy' 'kod_zrodlowy'
     link 'kod_przejściowy',,,,,
     ```

     for example:

     ```
     ml /Feprog.exe /Flprog.lst /Fmprog.map /Foprog.obj prog.asm
     link prog.obj,,,,,
     ```

EXERCICES

EX. 1(*). Compile and run programs presented during lecture.

EX. 2. Select correct processor's registers: a) AX, b) FX, c) BX, d) GX, e) CX.

EX. 3. Select correct answers. The instruction add `p, z`:

a) adds two numbers that are stored in registers p and z,
b) after the execution of the instruction the result is stored in registry p,
c) after the execution of the instruction the result is stored in registry z,
d) multiplies two numbers that are stored in registers p and z,

EX. 4. What instruction terminates the program written in assembler?

EX. 5. Select correct answers:

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

a) during the compilation process we use only compiler,
b) during the compilation process we use only linker,
c) during the compilation process we use compiler and linker,
d) during the compilation process the program is first linked and after it compiled,
e) during the compilation process the program is first compiled and after it linked,
f) compiler generates the executable code,
g) linker generates the executable code,
h) compiler generates the object code,
i) linker generates the object code,

EX. 6. Select correct answers:

a) to test the assembler program we use masm,
b) to test the assembler program we use debug,
c) to test the assembler program we use link,
d) liczby w rejestrach procesora są przechowywane w postaci binarnej,
e) liczby w rejestrach procesora są przechowywane w postaci dziesiętnej,
f) liczby w rejestrach procesora są przechowywane w postaci heksadecymalnej.

EX. 7. Fill gaps in statements about compilation process of assembler program provided below:

a) during compilation following files can be generated:
- object code stored in a file with extension ……………………………………
- executable code stored in a file with extension ………………………………………
- compilation listing stored in a file with extension ……………………………………
- compilation map stored in a file with extension ……………………………………

b) to get the help about options of *ml* program that can be used during compilation process the following command should be executed: `ml` ………………………………………………………………………

c) to get the help about options of *debug* program that can be used during testing process the following command should be executed: debug ………………………………………………………………

d) ………………………………………………….. – sample command that executed *debug* program to test the *prog.exe* program

e) What will be printed to the screen after executing following commands:

-
  ```
  -rax
  AX 0000
  :1
  ```
  ………

-
  ```
  -rax
  AX 0003
  :
  ```
  ………

-
  ```
  -g
  ```
  ………

-
  ```
  -q
  ```
  ………

EX. 8. Use the **direct addition** method to calculate following operations (numbers are provided in **hexadecimal system**, result should be provided also in hexadecimal system), (*4-6):

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

```
  524        735        345        2A8        E45        CCE
+ 214      + 879      + CBA      + 34C      + 28B      + ABA
```

EX. 9. Use the **direct subtraction** method to calculate following operations (numbers are provided in **hexadecimal system**, result should be provided also in hexadecimal system), (*4-6):

```
  548        EAB        189        437        1F2        E49
- 211      - 341       - 67       - 78       - BC      - AC8
```

EX. 10. What do we use two's complement code for and how is it constructed?

EX. 11. What is the **range** of numbers that can be coded using **5 bits** in two's complement code?

EX. 12. Calculate the hexadecimal, absolute value of following numbers coded in two's complement using 16 bits (*4-6):

```
FFFE       FFF7       FFD3       FEF9       F30E       88FF
```

EX. 13. Select correct answers. Change of sign during construction of negative number from its positive value is done by:

a) negation of bits and adding 1,
b) adding 1 and then negation of bits.

EX. 14. Present following numbers using two's complement and 16 bits (*4-6):

```
-4         -B         -E         -23        -C4        -A8
```

EX. 15. Select correct answers. The instruction sub p, z:

a) adds two numbers that are stored in registers p and z,
b) after the execution of the instruction the result is stored in registry p,
c) after the execution of the instruction the result is stored in registry z,
d) subtracts two numbers that are stored in registers p and z,

EX. 16. Select correct answers. After execution of instruction `neg c`:

a) the value in registry c is not changed,
b) in registry c there is a binary negation of the value that was stored in registry c.

EX. 17. Processor executes instructions coded in executable code using following scheme:

………………………………………………………………………………………………………,

Instruction pointer is ……………………………., which points to ……………………………….

EX. 18. Conditional jumps are executed using comparison instruction ………………….., and jump operation. What assembler instruction codes the following jumps:

a) **j**ump if **e**qual - ……………………………………………………………………………….
b) **j**ump if **n**ot **l**ess - …………………………………………………………………………….
c) **j**ump if **g**reater - ……………………………………………………………………………….

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

d) **j**ump if **n**ot **g**reater - ……………………………………………………………………………

e) **j**ump if **l**ess - ………………………………………………………………………………

EX. 19. Write in assembler the **while** loop and conditional instruction **if-else**.

EX. 20. Write the program that stores in registry AX **the smallest** number from numbers stored in registries BX, CX and DX. Below there are some example test cases.

Input data (hex)			Result
BX	**CX**	**DX**	**AX**
1	2	3	1
5	FF	FFFF	FFFF

EX. 21. Write the program that saves in registry AX **reminder from the division** of natural number stored in registry BX by the positive number stored in registry CX. Use **the method of multiple subtraction**. Below there are some test cases presented in decimal system:

Input data		Result
BX	**CX**	**AX**
6	5	1
16	5	2
4	5	4
5	5	0
F	4	3

EX. 22. Write a program that calculates the value of factorial n!. Assume that *n* is stored in registry BX and the result should be stored in registry AX. To multiply use the following instruction:

```
imul s
```

which stores in registries pair DX:AX the result of multiplication of registry AX by the numer stored in registry *s*. If the result can be stored using only 16 bits and is **not negative**, then the value of registry DX equals 0 (if the result can be stored using only 16 bits and is **negative** then the value of registry equals FFFF). For example instruction

```
imul bx
```

can be explained using following C instruction:

```
ax = ax * bx;
```

Below there are some example test cases.

Input: n	**Result: n!**
BX	**AX**
0	1
1	1
2	2
3	6
4	18

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.

What is the largest value of n for which the value of n! can be saved using 16 bits?

How the program should be modified to store the value of n in registry AX and the result in registry BX?

`EX. 23.` Write a program that calculates **number of decimal digits** of some number *n*. Use the instruction:

```
idiv s
```

that divides the value stored in registries pair DX:AX by the value *s*, stores the result in the registry AX and the reminder from the division in the registry DX. When a dividend is 16-bit you should remember about setting registry DX to 0.

For example following C instruction:

```
ax = ax / bx;
```

can be executed using following instructions:

```
mov dx, 0
idiv bx
```

Below there are some example test cases.

Input: n	Result:
DX	**CX**
D	2

`EX. 24(*).` Write a program that calculates **the value of a^n**, where *a* and *n* are natural numbers (a>0, *n* >= *0*). Use the **imul** instruction described in exercise 22.

Below there are some example test cases.

Input: n	Result: a^n	Input: a
BX	**AX**	**CX**
3	8	2

`EX. 25(*).` Write a program that calculates **the sum of decimal digits** of some number *n*. Use the **idiv** instruction described in exercise 23.

Below there are some example test cases.

Input: n	Result:
DX	**CX**
D	4

(*) gwiazdką oznaczone są zadania, które nie są realizowane na ćwiczeniach i są przeznaczone do wykonania jako zadania domowe.