8 BIT ARITHMETIC OPERATIONS

Name: Prasanna Kumaran D
Registration Number: 185001110 24 August 2020

1 AIM:

To write and execute 8086 programs for basic arithmetic operations like addition, subtraction, multiplication and division.

2 PROCEDURE:

- Write the program in a text editor and save it as a ***.asm*** file under the MASM directory.
- Launch DOSBOX application and mount the MASM folder using the command prompt.
- Use the following syntax for mounting : ' **mount [LOCAL DRIVE] FILEPATH** '. Enter into the local drive('LOCAL DRIVE:').
- The code file can be edited using the command **edit FILENAME.asm**. Save the changes and exit.
- Assemble the code using the command '**masm FILENAME.asm**' to generate the object file. The object file is in the format 'FILENAME.obj'
- Add dynamic libraries using the syntax **'link FILENAME.obj'** to generate the executeable(**.exe**) file.
- Enter the debug mode using debug FILENAME.exe to execute and analuse the memory contents. The various commands used in debug mode are as follows:-
  - U :- Displays unassembled code.
  - D :- Refers to the offset from which contents in the memory are displayed.
  - E :- Change the value in memory.
  - G :- execute the code.
  - Q :- Quit debug mode.

3 Algorithm

INITIALIZATION:

- Declare and initialize the operands and the code and data segments.

3.1 8 BIT ADDITION:

- Move the data segment to the AX register and then move it to the DS register.
- Move the first operand to AH register and the second to BH register.
- Initialize CH register to 00h.(hexadecimal)
- Add the operands using **ADD AH, BH**.

- Check for carry bit using JNC instruction.
  - Increment CH by 1 if there is a carry.
- Move the contents of AH and CH to RESULT and CARRY operands respectively.

3.2 8 BIT SUBTRACTION:

- Move the data segment to AX register and then move it to the DS register.
- Move the first and second operand to AH and BH register respectively.
- Initialize CH register to 00h.
- Subtract the operands **using SUB AH, BH**.
- Check for carry bit using JNC instruction.
  - If AH ¡ BH carry bit is generated and take 2's complement of AH using **NEG AH** and increment CH by 1.
- Move the contents of AH and CH to RESULT and CARRY operands respectively.

3.3 8 BIT MULTIPLICATION:

- Move the data segment to the AX register and move the AX to DS register.
- Move the first and second operands to AL and BL registers respectively.
- Multiply the operands using **MUL BL**
  - Note : AL is default operand register for MUL instruction and specifying the second operand register is sufficient.
- The result is stored in the **AX** register and since product of two 8 bit numbers yield a 16 bit number move the contents of AX register to RESULT.

3.4 8 BIT DIVISION

- Move the data segment to the AX register and then move it to the DS register.
- Now, set AH register to 00h and move first operand to AL register.(Since we cant directly divide a 8bit number by 8 bit number in 8086, we now make our dividend 16 bit by storing 00h in AH register and the 8-bit operand 1 in AL register).
- Move the second operand to the BL register.
- Now divide using DIV BL.(It will perform AX / BL. Because AH is 00h, what actually happens is the division of a 16 bit number by a 8 bit number.)
- The quotient and remainder stored in AL and AH should be moved to QUOTIENT and REMAINDER respectively.

4 Programs

4.1 8 BIT ADDITION:

PROGRAM	COMMENTS
mov ah,opr1	Transfers contents of operand 1to AH register.
mov bh,opr2	Transfers contents of operand 2to BH register.
mov ch,00h	Initialises CH register with 00h.
add ah,bh	AH=AH+BH.
jnc here	Jumps to „Here Label if carry bit is not generated.
inc ch	CH=CH+1(Increment CH by 1).
here: mov result,ah	Transfers contents of AH register to RESULT.
mov carry,ch	Transfers contents of CH register to CARRY.
mov ah, 4ch	Move the hexadecimal value 4c to ah
int 21h	When software interrupt 21 is called with AH=45,process is terminated

4.2 8 BIT SUBTRACTION:

Program	**Contents**
mov ah,opr1	Transfers contents of operand 1to AH register.
mov bh,opr2	Transfer contents of operand 2 to BH register.
mov ch,00h	Intialize CH register with 00h
sub ah,bh	AH = AH - BH
jnc here	Jumps to „Here Label if carry bit is not generated
neg ah	AH = 2's complement of AH
inc ch	CH = CH + 1
here: mov result,ah	Transfer contents from AH to RESULT
mov carry,ch	Transfer contents from CH to CARRY.
mov ah, 4ch	Move the hexadecimal value 4c to ah
int 21h	When software interrupt 21 is called with AH=45,process is terminated

4.3 8 BIT MULTIPLICATION:

Program	**Contents**
mov al, opr1	Transfers contents of operand 1to AL register
mov bl, opr2	Transfer contents of operand 2 to BL register
mul bl	BL = BL* AL
mov result,ax	Transfer contents of AX to RESULT
mov ah, 4ch	Move the hexadecimal value 4c to ah
int 21h	When software interrupt 21 is called with AH=45,process is terminated

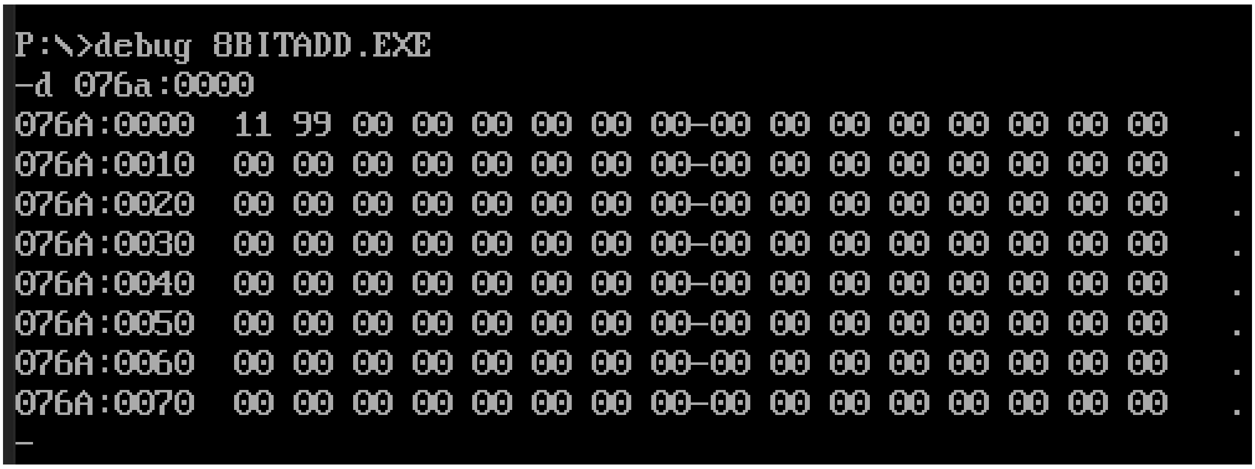
4.4 8 BIT DIVISION:

Program	**Contents**
mov ah, 00h	Move the value 00h to AH register.
mov al, opr1	Transfers contents of operand 1to AL register.
mov bl, opr2	Transfer contents of operand 2 to BL register.
div bl	Performs AX/BL.
mov quotient,al	Transfer contents from AL to QUOTIENT
mov remainder, ah	Transfer contents from AH to REMAINDER
mov ah, 4ch	Move the hexadecimal value 4c to ah
int 21h	When software interrupt 21 is called with AH=45,process is terminated

5 OUTPUTS:

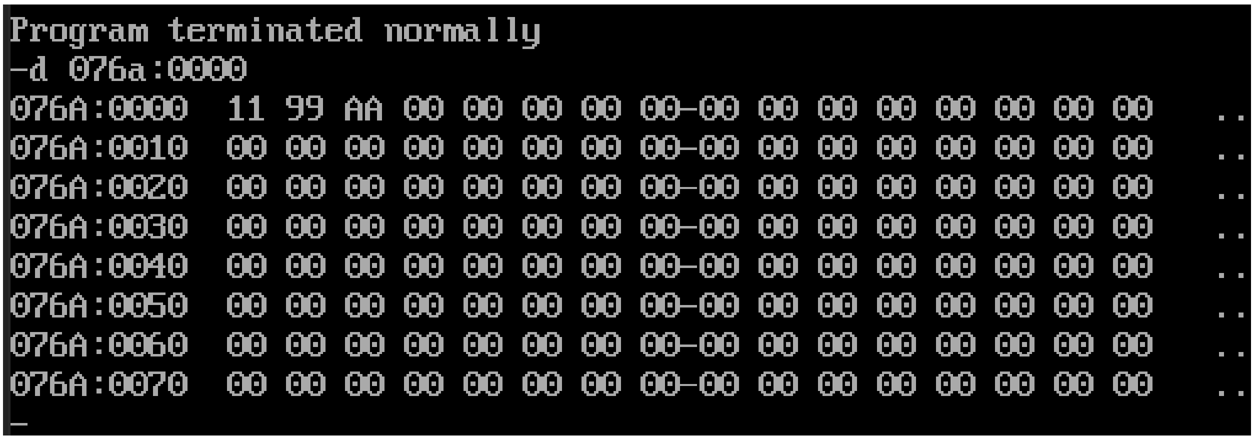
6 RESULT:

Thus,8086 programs for arithmetic operations like addition, subtraction, multiplication and division have been executed successfully using MS - DOSBox



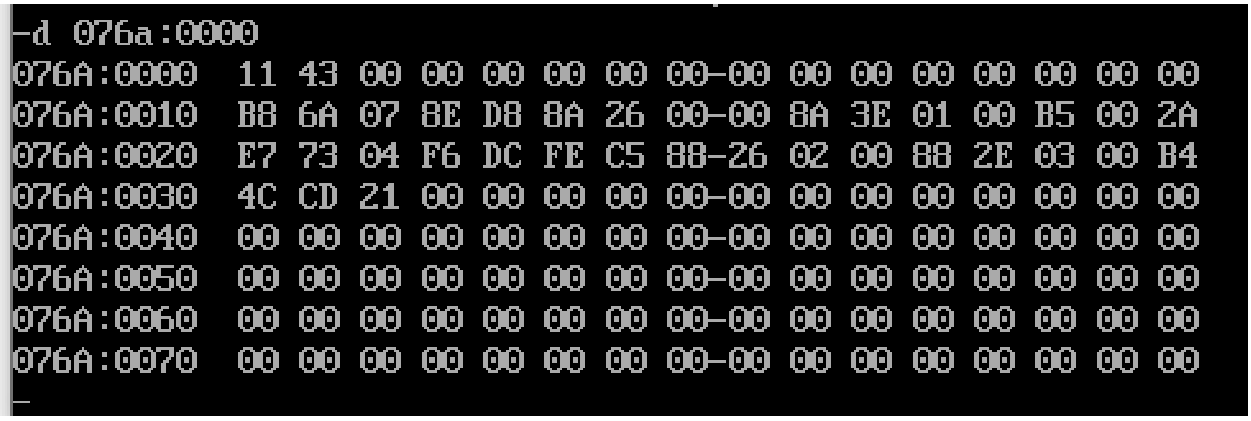
```
P:\>debug 8BITADD.EXE
-d 076a:0000
076A:0000  11 99 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   .
-
```

Figuur 1: 8 bit ADD Input



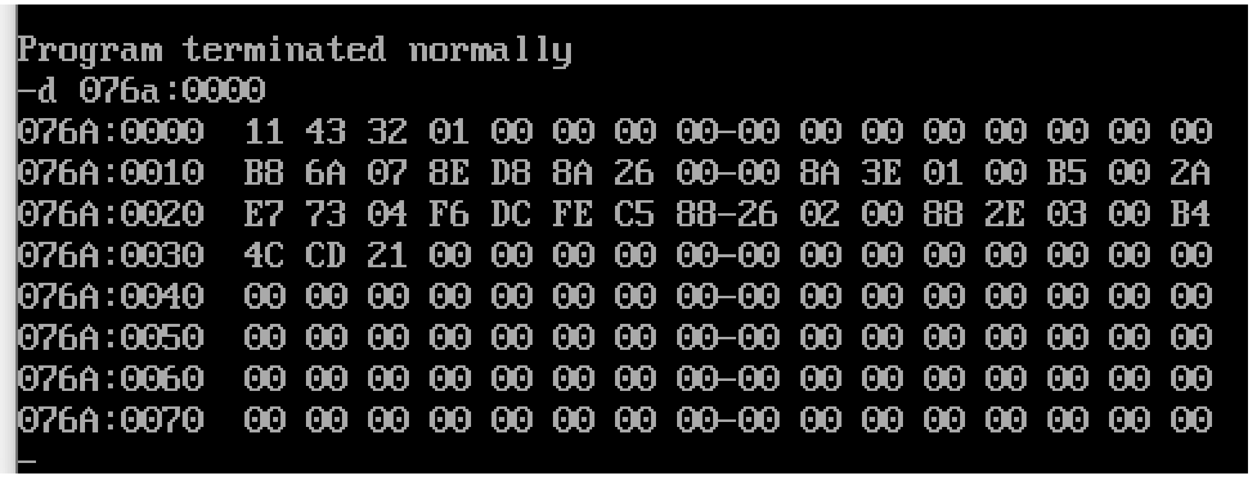
```
Program terminated normally
-d 076a:0000
076A:0000  11 99 AA 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ..
-
```

Figuur 2: 8 bit ADD OUTPUT



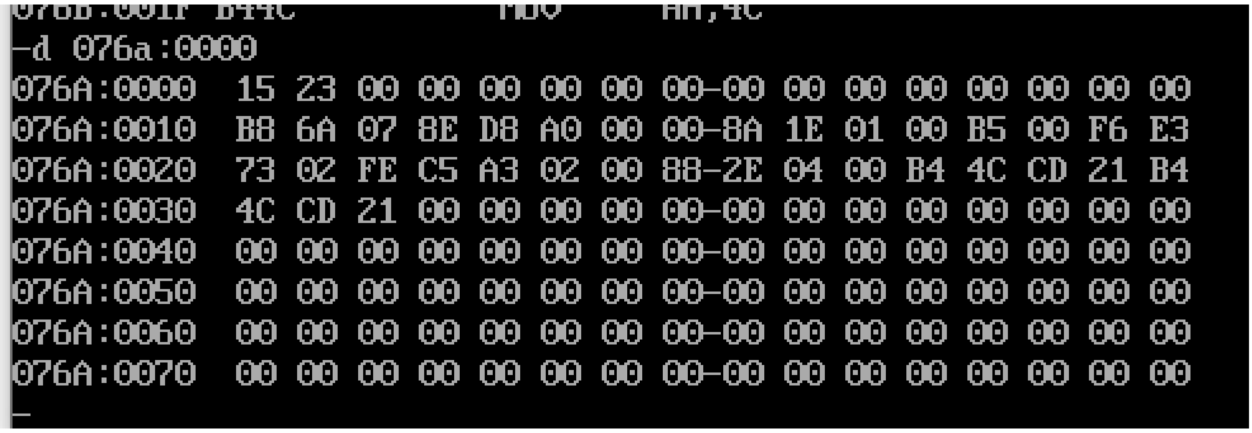
```
-d 076a:0000
076A:0000  11 43 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0010  B8 6A 07 8E D8 8A 26 00-00 8A 3E 01 00 B5 00 2A
076A:0020  E7 73 04 F6 DC FE C5 88-26 02 00 88 2E 03 00 B4
076A:0030  4C CD 21 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
-
```

Figuur 3: 8 bit SUB INPUT



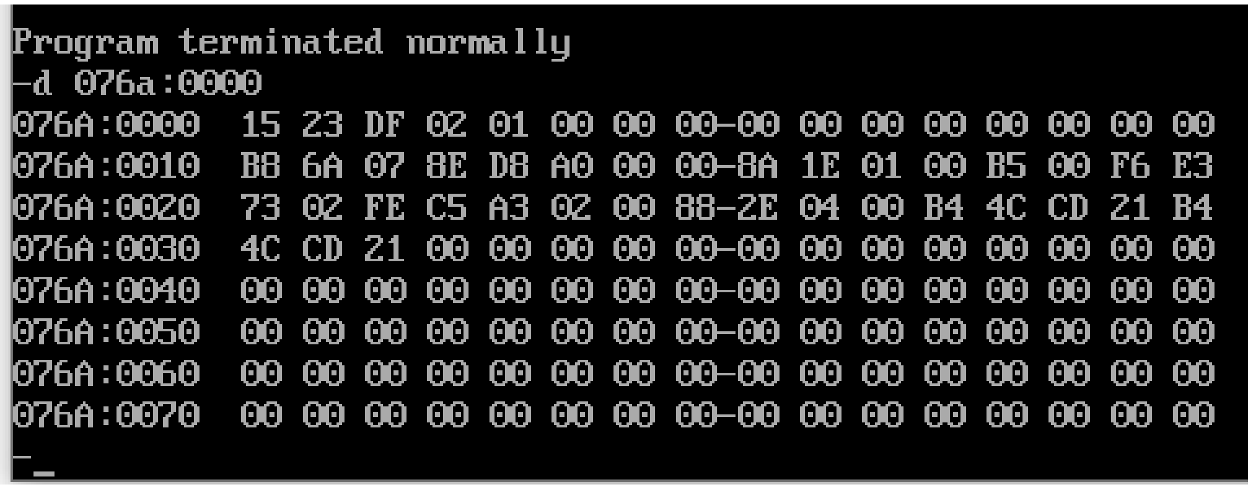
```
Program terminated normally
-d 076a:0000
076A:0000  11 43 32 01 00 00 00 00-00 00 00 00 00 00 00 00
076A:0010  B8 6A 07 8E D8 8A 26 00-00 8A 3E 01 00 B5 00 2A
076A:0020  E7 73 04 F6 DC FE C5 88-26 02 00 88 2E 03 00 B4
076A:0030  4C CD 21 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
-
```

Figuur 4: 8 bit SUB OUTPUT



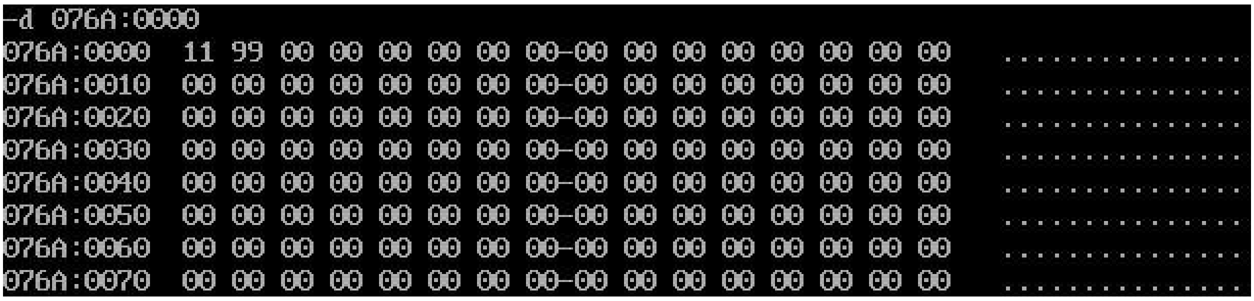
```
076B:001F B44C          MOV     AH,4C
-d 076a:0000
076A:0000  15 23 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0010  B8 6A 07 8E D8 A0 00 00-8A 1E 01 00 B5 00 F6 E3
076A:0020  73 02 FE C5 A3 02 00 88-2E 04 00 B4 4C CD 21 B4
076A:0030  4C CD 21 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
-
```

Figuur 5: 8 bit MUL INPUT



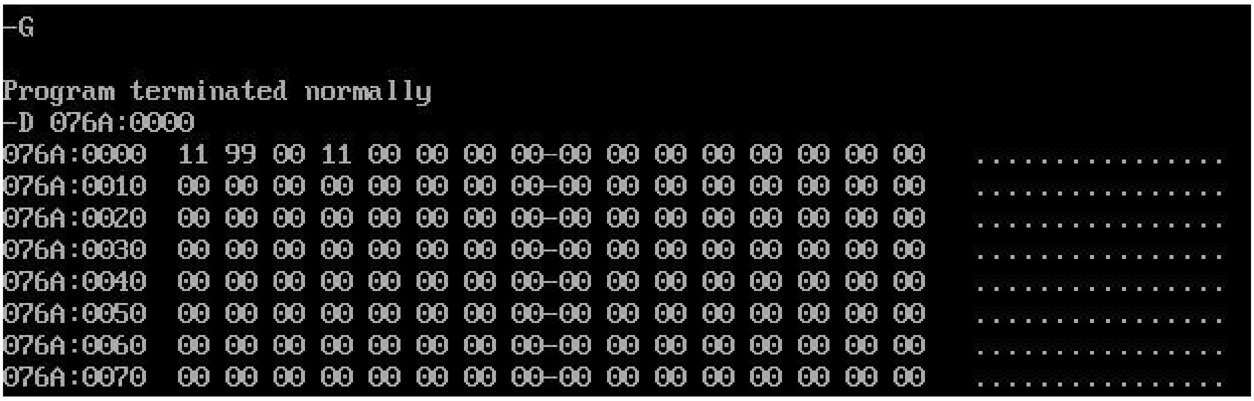
```
Program terminated normally
-d 076a:0000
076A:0000  15 23 DF 02 01 00 00 00-00 00 00 00 00 00 00 00
076A:0010  B8 6A 07 8E D8 A0 00 00-8A 1E 01 00 B5 00 F6 E3
076A:0020  73 02 FE C5 A3 02 00 88-2E 04 00 B4 4C CD 21 B4
076A:0030  4C CD 21 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
-_
```

Figuur 6: 8 bit MUL OUTPUT



```
-d 076A:0000
076A:0000  11 99 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

Figuur 7: 8 bit DIV INPUT



```
-G

Program terminated normally
-D 076A:0000
076A:0000  11 99 00 11 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0010  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0020  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0030  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0040  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0050  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0060  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
076A:0070  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00   ................
```

Figuur 8: 8 bit DIV OUTPUT