

Name : Tazmeen Afroz  
Roll No : 22P-9252  
Section : BAI-4A  
Coal Lab Task 9

### Problem: Multiplying Larger Numbers

When multiplying numbers, we face a limitation when dealing with larger numbers that exceed the register size (16 bits or 32 bits). Standard shifting operations (SHL, SHR) can only manipulate bits within the register size, causing significant bits to be dropped or lost when shifting larger numbers. This results in incorrect products. Forexample: When multiplying two 16-bit numbers, the result can be up to 32 bits long. However, we only have 16-bit registers available. This means we need to find a way to store and process the partial products without overflowing the registers.

### Solution: Extended Shifting

To overcome the memory issue, we use extended shifting.

### The Algorithm

The extended shifting algorithm involves two key instructions: SHL (Shift Left) and RCL (Rotate Carry Left). This technique is used to shift a 32-bit number left by 16 bits, ensuring no valuable bit is lost during the operation.

```
num1: dd 40000  
shl word [num1], 1  
rcl word [num1+2], 1  
word [num1+2], 1
```

In this example, num1 is a 32-bit number stored in memory. The SHL instruction shifts the lower 16 bits of num1 to the left, and the most significant bit is dropped into the carry. The RCL instruction then pushes this bit into the least significant bit of the next word, effectively joining the two 16-bit words.

For **shifting right**, the process is reversed. The SHR (Shift Right) and RCR instructions are used to ensure that no valuable bit is lost.

```
num1: dd 40000  
shr word [num1+2], 1  
rcr word [num1], 1  
word [num1], 1
```

- ADC (Add with Carry):
  - Adds two numbers and the carry flag.

## CODE

```
[org 0x0100]
    jmp start
multiplicand: dd 1300 ; 16bit multiplicand 32bit space
multiplier: dw 500 ; 16bit multiplier
result: dd 0 ; 32bit result
start: mov cl, 16 ; initialize bit count to 16
      mov dx, [multiplier] ; load multiplier in dx
checkbit: shr dx, 1 ; move right most bit in carry
      jnc skip ; skip addition if bit is zero
      mov ax, [multiplicand]
      add [result], ax ; add less significant word
      mov ax, [multiplicand+2]
      adc [result+2], ax ; add more significant word
skip: shl word [multiplicand], 1
      rcl word [multiplicand+2], 1 ; shift multiplicand left
      dec cl ; decrement bit count
      jnz checkbit ; repeat if bits left
      mov ax, 0x4c00
```

## Code Logic:

1. Initialize bit count to 16 and load multiplier in DX.
  - Set CL to 16 to keep track of the number of bits in the multiplier.
  - Load the multiplier into DX.
2. Check the rightmost bit of the multiplier (DX). If 0, skip addition.
  - Shift DX right by one bit, moving the rightmost bit to the carry.
  - If the carry is not set (meaning the rightmost bit of the multiplier was 0), proceed to step 4.
3. If the rightmost bit of the multiplier was 1, proceed with addition:
  - Add the less significant word of the multiplicand to the result.
  - Add the more significant word of the multiplicand to the result with carry.
4. Shift the multiplicand left by one bit.

- Shift the lower 16 bits of multiplicand left by one bit, moving the leftmost bit to the carry and filling the rightmost bit with 0.
5. **Rotate the multiplier right by one bit with carry.**
    - Rotate the carry left into the least significant bit of the upper 16 bits of multiplicand.
  6. **Decrement the bit count and repeat steps until all bits are processed.**
    - Decrement CL by one.
    - If CL is not zero, repeat steps 2-5.
  7. **Finalization:** After all iterations are complete, the result contains the 32-bit product of the two 16-bit numbers.

### Binary Representation:

Multiplicand (1300): 0000010110100100

Multiplier (500): 0000000110011000

### Step-Wise Code Explanation:

#### Initialization:

[org 0x0100] ; Origin at memory address 100h

jmp start ; Jump to the start of the program

#### Data Definition:

**multiplicand:** dd 1300 ; Define a 32-bit data element (double word)

The dd 1300 directive reserves 4 bytes of memory. The value 1300 in hexadecimal is **0x0514**. Since it's a 16-bit value, it's stored in the lower 16 bits of the 32-bit space, with the upper 16 bits being zero.

1	0	1	2	3	4	5	6	7
DS:0100	E9	0A	00	14	05	00	00	F4
DS:0108	01	00	00	00	00	B1	10	8B

**multiplier:** dw 500 ; Define a 16-bit data element (word)

Similar to line 3, dw 500 reserves 2 bytes. The value 500 in hexadecimal is 0x01F4.

**result:** dd 0 ; Define a 32-bit data element (double word) to store the result

1	0	1	2	3	4	5	6	7
DS:0100	F9	0A	00	14	05	00	00	F4
DS:0108	01	00	00	00	00	B1	10	8B

## Program Start

start:

mov cl, 16 ; Initialize the bit count to 16 (0010 in hex)

```
AX 0000 SI 0000 CS 19F5 IP 010F Stack
BX 0000 DI 0000 DS 19F5
CX 0010 BP 0000 ES 19F5 HS 19F5
DX 0000 SP FFFE SS 19F5 FS 19F5

CMD >

010D B110 MOV CL,10
```

mov dx, [multiplier] ; Load the multiplier into the dx register

DX 01F4

```
AX 0000 SI 0000 CS 19F5 IP 0113 Stack
BX 0000 DI 0000 DS 19F5
CX 0010 BP 0000 ES 19F5 HS 19F5
DX 01F4 SP FFFE SS 19F5 FS 19F5

CMD >

010F 8B160701 MOV DX,[0107]
```

## Iteration Loop

First Iteration:

checkbit:

shr dx, 1 ; Shift the multiplier right by one bit and move the rightmost bit to the carry  
(111110100 → 011111010) (hex will be 0xFA)(Decimal → 250)

```
AX 0000 SI 0000 CS 19F5 IP 0115 Stack
BX 0000 DI 0000 DS 19F5
CX 0010 BP 0000 ES 19F5 HS 19F5
DX 00FA SP FFFE SS 19F5 FS 19F5

CMD >

0113 D1EA SHR DX,1
```

jnc skip ; skip the addition

As carry flag is not set we are skipping the addition.



dec cl ; decrement bit count  
jnz checkbit ; repeat if bits left

Iteration 2:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD										-		X		
AX 0000	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7214									
BX 0000	DI 0000	DS 19F5		+2 20CD										
CX 000F	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF									
DX 0071	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 0 1 1 0									
CMD >														
						1	0	1	2	3	4	5	6	7
0129 D1160501 RCL W/[0105],1						DS:0100	E9	0A	00	50	14	00	00	F4
012D FEC9 DEC CL						DS:0108	01	00	00	00	00	B1	10	8B
012F 75E2 JNZ 0113						DS:0110	16	07	01	D1	EA	73	0E	A1
0131 B8004C MOV AX,4C00						DS:0118	03	01	01	06	09	01	A1	05
0134 85D2 TEST DX,DX						DS:0120	01	11	06	0B	01	D1	26	03
0136 7504 JNZ 013C						DS:0128	01	D1	16	05	01	FE	C9	75
0138 85C0 TEST AX,AX						DS:0130	E2	B8	00	4C	85	D2	75	04
013A 741C JZ 0158						DS:0138	85	C0	74	1C	C7	46	DC	00
013C C746DC0000 MOV [BP-24],0000						DS:0140	00	8E	5E	FC	83	7D	0E	00
						DS:0148	74	09	8B	46	F2	48	3B	46

Iteration 3:

multiplicand : 0000000001111101

After shift right carry flag is set.

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD										-		x		G	
AX 0000	SI 0000	CS 19F5	IP 0115	Stack +0 0000	Flags 7211										
BX 0000	DI 0000	DS 19F5		+2 20CD											
CX 000E	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF										
DX 003E	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 0 1 0 1										
CMD >															
0113 D1EA SHR DX,1						1	0	1	2	3	4	5	6	7	
0115 730E JNC 0125						DS:0100	E9	0A	00	50	14	00	00	F4	
0117 A10301 MOV AX,[0103]						DS:0108	01	00	00	00	00	B1	10	8B	
011A 01060901 ADD [0109],AX						DS:0110	16	07	01	D1	EA	73	0E	A1	
011E A10501 MOV AX,[0105]						DS:0118	03	01	01	06	09	01	A1	05	
0121 11060B01 ADC [010B],AX						DS:0120	01	11	06	0B	01	D1	26	03	
0125 D1260301 SHL W/[0103],1						DS:0128	01	D1	16	05	01	FE	C9	75	
0129 D1160501 RCL W/[0105],1						DS:0130	E2	B8	00	4C	85	D2	75	04	
012D FEC9 DEC CL						DS:0138	85	C0	74	1C	C7	46	DC	00	
						DS:0140	00	8E	5E	FC	83	7D	0E	00	
						DS:0148	74	09	8B	46	F2	48	3B	46	

Addition:

mov ax, [multiplicand]  
add [result], ax ; add less significant word  
mov ax, [multiplicand+2]  
adc [result+2], ax ; add more significant word

mov the multiplicand to ax

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX	1450	SI	0000	CS	19F5	IP	011A	Stack	+0 0000	Flags	7211						
BX	0000	DI	0000	DS	19F5				+2 20CD								
CX	000E	BP	0000	ES	19F5	HS	19F5		+4 9FFF	OF	DF	IF	SF	ZF	AF	PF	CF
DX	003E	SP	FFFE	SS	19F5	FS	19F5		+6 EA00	0	0	1	0	0	1	0	1

multiplicand 0001010001010000

add [result], ax ;

1	0	1	2	3	4	5	6	7
DS:0100	E9	0A	00	50	14	00	00	F4
DS:0108	01	50	14	00	00	B1	10	8B
DS:0110	16	07	01	D1	EA	73	0E	A1
DS:0118	03	01	01	06	09	01	A1	05
DS:0120	01	11	06	0B	01	D1	26	03
DS:0128	01	D1	16	05	01	FE	C9	75
DS:0130	E2	B8	00	4C	85	D2	75	04
DS:0138	85	C0	74	1C	C7	46	DC	00
DS:0140	00	8E	5E	FC	83	7D	0E	00
DS:0148	74	09	8B	46	F2	48	3B	46

mov ax, [multiplicand+2]

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD

AX 0000

BX 0000

CX 000E

DX 003E

SI 0000

DI 0000

BP 0000

SP FFFE

CS 19F5

DS 19F5

ES 19F5

SS 19F5

IP 0121

HS 19F5

FS 19F5

Stack +0 0000

+2 20CD

+4 9FFF

+6 EA00

Flags 7204

OF DF IF SF ZF AF PF CF

0 0 1 0 0 0 1 0

CMD >

0000

1

0 1 2 3 4 5 6 7

DS:0100 E9 0A 00 50 14 00 00 F4

DS:0108 01 50 14 00 00 B1 10 8B

DS:0110 16 07 01 D1 EA 73 0E A1

DS:0118 03 01 01 06 09 01 A1 05

DS:0120 01 11 06 0B 01 D1 26 03

DS:0128 01 D1 16 05 01 FE C9 75

DS:0130 E2 B8 00 4C 85 D2 75 04

DS:0138 85 C0 74 1C C7 46 DC 00

DS:0140 00 8E 5E FC 83 7D 0E 00

DS:0148 74 09 8B 46 F2 48 3B 46

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 0125 Stack +0 0000 Flags 7244
BX 0000 DI 0000 DS 19F5 +2 20CD
CX 000E BP 0000 ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF
DX 003E SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 1 0 1 0

CMD >
1 0 1 2 3 4 5 6 7
DS:0100 E9 0A 00 50 14 00 00 F4
DS:0108 01 50 14 00 00 B1 10 8B
DS:0110 16 07 01 D1 EA 73 0E A1
DS:0118 03 01 01 06 09 01 A1 05
DS:0120 01 11 06 0B 01 D1 26 03
DS:0128 01 D1 16 05 01 FE C9 75
DS:0130 E2 B8 00 4C 85 D2 75 04
DS:0138 85 C0 74 1C C7 46 DC 00
DS:0140 00 8E 5E FC 83 7D 0E 00
DS:0148 74 09 8B 46 F2 48 3B 46

```

```

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD
AX 0000 SI 0000 CS 19F5 IP 012D Stack +0 0000 Flags 7214
BX 0000 DI 0000 DS 19F5 +2 20CD
CX 000E BP 0000 ES 19F5 HS 19F5 +4 9FFF OF DF IF SF ZF AF PF CF
DX 003E SP FFFE SS 19F5 FS 19F5 +6 EA00 0 0 1 0 0 1 1 0

CMD >

0129 D1160501 RCL W/[0105],1
012D FEC9 DEC CL
012F 75E2 JNZ 0113
0131 B8004C MOV AX,4C00
0134 85D2 TEST DX,DX
0136 7504 JNZ 013C
0138 85C0 TEST AX,AX
013A 741C JZ 0158
013C C746DC0000 MOV [BP-24],0000

1 0 1 2 3 4 5 6 7
DS:0100 E9 0A 00 A0 28 00 00 F4
DS:0108 01 50 14 00 00 B1 10 8B
DS:0110 16 07 01 D1 EA 73 0E A1
DS:0118 03 01 01 06 09 01 A1 05
DS:0120 01 11 06 0B 01 D1 26 03
DS:0128 01 D1 16 05 01 FE C9 75
DS:0130 E2 B8 00 4C 85 D2 75 04
DS:0138 85 C0 74 1C C7 46 DC 00
DS:0140 00 8E 5E FC 83 7D 0E 00
DS:0148 74 09 8B 46 F2 48 3B 46

```



#### Iteration 4:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD													
AX 0000	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7290								
BX 0000	DI 0000	DS 19F5		+2 20CD									
CX 000C	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF								
DX 000F	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 1 0 1 0 0								
CMD >													
0129 D1160501	RCL	W/[0105],1		1	0	1	2	3	4	5	6	7	
012D FEC9	DEC	CL		DS:0100	E9	0A	00	80	A2	00	00	F4	
012F 75E2	JNZ	0113		DS:0108	01	90	65	00	00	B1	10	8B	
0131 B8004C	MOV	AX,4C00		DS:0110	16	07	01	D1	EA	73	0E	A1	
0134 85D2	TEST	DX,DX		DS:0118	03	01	01	06	09	01	A1	05	
0136 7504	JNZ	013C		DS:0120	01	11	06	0B	01	D1	26	03	
0138 85C0	TEST	AX,AX		DS:0128	01	D1	16	05	01	FE	C9	75	
013A 741C	JZ	0158		DS:0130	E2	B8	00	4C	85	D2	75	04	
013C C746DC0000	MOV	[BP-24],0000		DS:0138	85	C0	74	1C	C7	46	DC	00	
				DS:0140	00	8E	5E	FC	83	7D	0E	00	
				DS:0148	74	09	8B	46	F2	48	3B	46	

#### Iteration 5:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD													
AX 0000	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7214								
BX 0000	DI 0000	DS 19F5		+2 20CD									
CX 000B	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF								
DX 0007	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 0 1 1 0								
CMD >													
0129 D1160501	RCL	W/[0105],1		DS:0100	E9 0A 00 00 45 01 00 F4								
012D FEC9	DEC	CL		DS:0108	01 10 08 01 00 B1 10 8B								
012F 75E2	JNZ	0113		DS:0110	16 07 01 D1 EA 73 0E A1								
0131 B8004C	MOV	AX,4C00		DS:0118	03 01 01 06 09 01 A1 05								
0134 85D2	TEST	DX,DX		DS:0120	01 11 06 0B 01 D1 26 03								
0136 7504	JNZ	013C		DS:0128	01 D1 16 05 01 FE C9 75								
0138 85C0	TEST	AX,AX		DS:0130	E2 B8 00 4C 85 D2 75 04								
013A 741C	JZ	0158		DS:0138	85 C0 74 1C C7 46 DC 00								
013C C746DC0000	MOV	[BP-24],0000		DS:0140	00 8E 5E FC 83 7D 0E 00								
				DS:0148	74 09 8B 46 F2 48 3B 46								

Iteration 6:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD											
AX	0001	SI	0000	CS	19F5	IP	012D	Stack	+0	0000	Flags 7294
BX	0000	DI	0000	DS	19F5				+2	20CD	
CX	000A	BP	0000	ES	19F5	HS	19F5		+4	9FFF	OF DF IF SF ZF AF PF CF
DX	0003	SP	FFFE	SS	19F5	FS	19F5		+6	EA00	0 0 1 1 0 1 1 0
CMD >						1 0 1 2 3 4 5 6 7					
0129	D1160501	RCL	W/[0105],1			DS:0100	E9 0A 00 00	8A 02 00 F4			
012D	FEC9	DEC	CL			DS:0108	01 10 4D 02	00 B1 10 8B			
012F	75E2	JNZ	0113			DS:0110	16 07 01 D1	EA 73 0E A1			
0131	B8004C	MOV	AX,4C00			DS:0118	03 01 01 06	09 01 A1 05			
0134	85D2	TEST	DX,DX			DS:0120	01 11 06 0B	01 D1 26 03			
0136	7504	JNZ	013C			DS:0128	01 D1 16 05	01 FE C9 75			
0138	85C0	TEST	AX,AX			DS:0130	E2 B8 00 4C	85 D2 75 04			
013A	741C	JZ	0158			DS:0138	85 C0 74 1C	C7 46 DC 00			
013C	C746DC0000	MOV	[BP-24],0000			DS:0140	00 8E 5E FC	83 7D 0E 00			
						DS:0148	74 09 8B 46	F2 48 3B 46			

Iteration 7:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD											
AX	0002	SI	0000	CS	19F5	IP	012D	Stack	+0	0000	Flags 7214
BX	0000	DI	0000	DS	19F5				+2	20CD	
CX	0009	BP	0000	ES	19F5	HS	19F5		+4	9FFF	OF DF IF SF ZF AF PF CF
DX	0001	SP	FFFE	SS	19F5	FS	19F5		+6	EA00	0 0 1 0 0 1 1 0
CMD >						1 0 1 2 3 4 5 6 7					
0129	D1160501	RCL	W/[0105],1			DS:0100	E9 0A 00 00	14 05 00 F4			
012D	FEC9	DEC	CL			DS:0108	01 10 D7 04	00 B1 10 8B			
012F	75E2	JNZ	0113			DS:0110	16 07 01 D1	EA 73 0E A1			
0131	B8004C	MOV	AX,4C00			DS:0118	03 01 01 06	09 01 A1 05			
0134	85D2	TEST	DX,DX			DS:0120	01 11 06 0B	01 D1 26 03			
0136	7504	JNZ	013C			DS:0128	01 D1 16 05	01 FE C9 75			
0138	85C0	TEST	AX,AX			DS:0130	E2 B8 00 4C	85 D2 75 04			
013A	741C	JZ	0158			DS:0138	85 C0 74 1C	C7 46 DC 00			
013C	C746DC0000	MOV	[BP-24],0000			DS:0140	00 8E 5E FC	83 7D 0E 00			
						DS:0148	74 09 8B 46	F2 48 3B 46			

# Iteration 8:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD									
AX 0005	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7214				
BX 0000	DI 0000	DS 19F5		+2 20CD					
CX 0008	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF	DF	IF	SF	ZF AF PF CF
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0	0	1	0	0 1 1 0
CMD >				1 0 1 2 3 4 5 6 7					
0129 D1160501	RCL	W/[0105],1		DS:0100	E9	0A	00	00	28 0A 00 F4
012D FEC9	DEC	CL		DS:0108	01	10	EB	09	00 B1 10 8B
012F 75E2	JNZ	0113		DS:0110	16	07	01	D1	EA 73 0E A1
0131 B8004C	MOV	AX,4C00		DS:0118	03	01	01	06	09 01 A1 05
0134 85D2	TEST	DX,DX		DS:0120	01	11	06	0B	01 D1 26 03
0136 7504	JNZ	013C		DS:0128	01	D1	16	05	01 FE C9 75
0138 85C0	TEST	AX,AX		DS:0130	E2	B8	00	4C	85 D2 75 04
013A 741C	JZ	0158		DS:0138	85	C0	74	1C	C7 46 DC 00
013C C746DC0000	MOV	[BP-24],0000		DS:0140	00	8E	5E	FC	83 7D 0E 00
				DS:0148	74	09	8B	46	F2 48 3B 46

# Iteration 9:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD									
AX 0005	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7214				
BX 0000	DI 0000	DS 19F5		+2 20CD					
CX 0007	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF	DF	IF	SF	ZF AF PF CF
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0	0	1	0	0 1 1 0
CMD >				1 0 1 2 3 4 5 6 7					
0129 D1160501	RCL	W/[0105],1		DS:0100	E9	0A	00	00	50 14 00 F4
012D FEC9	DEC	CL		DS:0108	01	10	EB	09	00 B1 10 8B
012F 75E2	JNZ	0113		DS:0110	16	07	01	D1	EA 73 0E A1
0131 B8004C	MOV	AX,4C00		DS:0118	03	01	01	06	09 01 A1 05
0134 85D2	TEST	DX,DX		DS:0120	01	11	06	0B	01 D1 26 03
0136 7504	JNZ	013C		DS:0128	01	D1	16	05	01 FE C9 75
0138 85C0	TEST	AX,AX		DS:0130	E2	B8	00	4C	85 D2 75 04
013A 741C	JZ	0158		DS:0138	85	C0	74	1C	C7 46 DC 00
013C C746DC0000	MOV	[BP-24],0000		DS:0140	00	8E	5E	FC	83 7D 0E 00
				DS:0148	74	09	8B	46	F2 48 3B 46

Iteration 10:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD										
AX 0005	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7294					
BX 0000	DI 0000	DS 19F5		+2 20CD						
CX 0006	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF					
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 1 0 1 1 0					
CMD >				1 0 1 2 3 4 5 6 7						
0129 D1160501	RCL	W/[0105],1	DS:0100 E9 0A 00 00 A0 28 00 F4							
012D FEC9	DEC	CL	DS:0108 01 10 EB 09 00 B1 10 8B							
012F 75E2	JNZ	0113	DS:0110 16 07 01 D1 EA 73 0E A1							
0131 B8004C	MOV	AX,4C00	DS:0118 03 01 01 06 09 01 A1 05							
0134 85D2	TEST	DX,DX	DS:0120 01 11 06 0B 01 D1 26 03							
0136 7504	JNZ	013C	DS:0128 01 D1 16 05 01 FE C9 75							
0138 85C0	TEST	AX,AX	DS:0130 E2 B8 00 4C 85 D2 75 04							
013A 741C	JZ	0158	DS:0138 85 C0 74 1C C7 46 DC 00							
013C C746DC0000	MOV	[BP-24],0000	DS:0140 00 8E 5E FC 83 7D 0E 00							
			DS:0148 74 09 8B 46 F2 48 3B 46							

Iteration 11:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD										
AX 0005	SI 0000	CS 19F5	IP 012D	Stack +0	0000	Flags 7214				
BX 0000	DI 0000	DS 19F5		+2	20CD					
CX 0005	BP 0000	ES 19F5	HS 19F5	+4	9FFF	OF DF IF SF ZF AF PF CF				
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6	EA00	0 0 1 0 0 1 1 0				
CMD >				1 0 1 2 3 4 5 6 7						
0129 D1160501	RCL	W/[0105],1	DS:0100 E9 0A 00 00 40 51 00 F4							
012D FEC9	DEC	CL	DS:0108 01 10 EB 09 00 B1 10 8B							
012F 75E2	JNZ	0113	DS:0110 16 07 01 D1 EA 73 0E A1							
0131 B8004C	MOV	AX,4C00	DS:0118 03 01 01 06 09 01 A1 05							
0134 85D2	TEST	DX,DX	DS:0120 01 11 06 0B 01 D1 26 03							
0136 7504	JNZ	013C	DS:0128 01 D1 16 05 01 FE C9 75							
0138 85C0	TEST	AX,AX	DS:0130 E2 B8 00 4C 85 D2 75 04							
013A 741C	JZ	0158	DS:0138 85 C0 74 1C C7 46 DC 00							
013C C746DC0000	MOV	[BP-24],0000	DS:0140 00 8E 5E FC 83 7D 0E 00							
			DS:0148 74 09 8B 46 F2 48 3B 46							

### Iteration 12:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD			
AX 0005	SI 0000	CS 19F5	IP 012D
BX 0000	DI 0000	DS 19F5	
CX 0004	BP 0000	ES 19F5	HS 19F5
DX 0000	SP FFFE	SS 19F5	FS 19F5
Stack +0 0000		Flags 7294	
+2 20CD			
+4 9FFF		OF	DF IF SF ZF AF PF CF
+6 EA00		0	0 1 1 0 1 1 0
CMD >			
0129 D1160501	RCL	W/[0105],1	
012D FEC9	DEC	CL	
012F 75E2	JNZ	0113	
0131 B8004C	MOV	AX,4C00	
0134 85D2	TEST	DX,DX	
0136 7504	JNZ	013C	
0138 85C0	TEST	AX,AX	
013A 741C	JZ	0158	
013C C746DC0000	MOV	[BP-24],0000	
DS:0100		E9	0A 00 00 80 A2 00 F4
DS:0108		01	10 EB 09 00 B1 10 8B
DS:0110		16	07 01 D1 EA 73 0E A1
DS:0118		03	01 01 06 09 01 A1 05
DS:0120		01	11 06 0B 01 D1 26 03
DS:0128		01	D1 16 05 01 FE C9 75
DS:0130		E2	B8 00 4C 85 D2 75 04
DS:0138		85	C0 74 1C C7 46 DC 00
DS:0140		00	8E 5E FC 83 7D 0E 00
DS:0148		74	09 8B 46 F2 48 3B 46

### Iteration 13:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD			
AX 0005	SI 0000	CS 19F5	IP 012D
BX 0000	DI 0000	DS 19F5	
CX 0003	BP 0000	ES 19F5	HS 19F5
DX 0000	SP FFFE	SS 19F5	FS 19F5
Stack +0 0000		Flags 7254	
+2 20CD			
+4 9FFF		OF	DF
+6 EA00		IF	SF
		ZF	AF
		PF	CF
		0	0
		1	0
		1	1
		1	0
CMD >			
0129 D1160501	RCL	W/[0105],1	
012D FEC9	DEC	CL	
012F 75E2	JNZ	0113	
0131 B8004C	MOV	AX,4C00	
0134 85D2	TEST	DX,DX	
0136 7504	JNZ	013C	
0138 85C0	TEST	AX,AX	
013A 741C	JZ	0158	
013C C746DC0000	MOV	[BP-24],0000	
		1	0
		0	1
		2	3
		4	5
		6	7
		DS:0100	E9 0A 00 00 45 01 F4
		DS:0108	01 10 EB 09 00 B1 10 8B
		DS:0110	16 07 01 D1 EA 73 0E A1
		DS:0118	03 01 01 06 09 01 A1 05
		DS:0120	01 11 06 0B 01 D1 26 03
		DS:0128	01 D1 16 05 01 FE C9 75
		DS:0130	E2 B8 00 4C 85 D2 75 04
		DS:0138	85 C0 74 1C C7 46 DC 00
		DS:0140	00 8E 5E FC 83 7D 0E 00
		DS:0148	74 09 8B 46 F2 48 3B 46

Iteration 14:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD									
AX 0005	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7254				
BX 0000	DI 0000	DS 19F5		+2 20CD					
CX 0002	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF				
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 1 1 1 0				
CMD >					0 1 2 3 4 5 6 7				
0129 D1160501	RCL	W/[0105],1			DS:0100	E9 0A 00 00 00 8A 02 F4			
012D FEC9	DEC	CL			DS:0108	01 10 EB 09 00 B1 10 8B			
012F 75E2	JNZ	0113			DS:0110	16 07 01 D1 EA 73 0E A1			
0131 B8004C	MOV	AX,4C00			DS:0118	03 01 01 06 09 01 A1 05			
0134 85D2	TEST	DX,DX			DS:0120	01 11 06 0B 01 D1 26 03			
0136 7504	JNZ	013C			DS:0128	01 D1 16 05 01 FE C9 75			
0138 85C0	TEST	AX,AX			DS:0130	E2 B8 00 4C 85 D2 75 04			
013A 741C	JZ	0158			DS:0138	85 C0 74 1C C7 46 DC 00			
013C C746DC0000	MOV	[BP-24],0000			DS:0140	00 8E 5E FC 83 7D 0E 00			
					DS:0148	74 09 8B 46 F2 48 3B 46			

Iteration 15:

DOSBox 0.74-3, Cpu speed: 3000 cycles, Frameskip 0, Program: AFD									
AX 0005	SI 0000	CS 19F5	IP 012D	Stack +0 0000	Flags 7254				
BX 0000	DI 0000	DS 19F5		+2 20CD					
CX 0001	BP 0000	ES 19F5	HS 19F5	+4 9FFF	OF DF IF SF ZF AF PF CF				
DX 0000	SP FFFE	SS 19F5	FS 19F5	+6 EA00	0 0 1 0 1 1 1 0				
CMD >					0 1 2 3 4 5 6 7				
0129 D1160501	RCL	W/[0105],1			DS:0100	E9 0A 00 00 00 14 05 F4			
012D FEC9	DEC	CL			DS:0108	01 10 EB 09 00 B1 10 8B			
012F 75E2	JNZ	0113			DS:0110	16 07 01 D1 EA 73 0E A1			
0131 B8004C	MOV	AX,4C00			DS:0118	03 01 01 06 09 01 A1 05			
0134 85D2	TEST	DX,DX			DS:0120	01 11 06 0B 01 D1 26 03			
0136 7504	JNZ	013C			DS:0128	01 D1 16 05 01 FE C9 75			
0138 85C0	TEST	AX,AX			DS:0130	E2 B8 00 4C 85 D2 75 04			
013A 741C	JZ	0158			DS:0138	85 C0 74 1C C7 46 DC 00			
013C C746DC0000	MOV	[BP-24],0000			DS:0140	00 8E 5E FC 83 7D 0E 00			
					DS:0148	74 09 8B 46 F2 48 3B 46			

1300x500

=

650000

Enter hex number

0009EB10

16

= Convert

× Reset

↕ Swap

Binary number (32 digits)

00000000000010011110101100010000

2

Decimal number (6 digits)

650000

10