Homework 2

Objectives:
This exercise aims to demonstrate how Assembly code translates to the circuits, which is what you already learned reading about the IA32 architecture.

Description:
Below is the code that calculates the user's age in dog years. Considering that the **.data** segment starts at the address **0x1000 in main memory**, the **.code** segment start at the address **0x2000 in cache memory**, and the value for the age the user gives is 20, write down the changes in variables and registers for each instruction. Assume that when the "OFFSET" operator in used, only the address of the variable is fetched, **NOT** the data. The first 4 instructions are given. You can stop after the instruction "exit"

EXTRA CREDIT (+10 points):
Modify the code in Visual Studio so when the user enters a number for their age below 5, the program prints out a message "Wow! That young, yet you know how to use a computer!".

What to submit:
Both, the document file with the program execution and the .asm file for the extra credit.

```
TITLE Dog years     (demo1.asm)
; Description:  This program gets the age of the user and calculates their age in dog years (age x 7).
INCLUDE Irvine32.inc

.data
       age           DWORD  ?                                              ; User's age
       hi_there      BYTE   "Hi there, this is John",0                      ; Greeting the user
       prompt1       BYTE   "Can I have your age please?",0                 ; Gets age
       output        BYTE   "So, your age in dog years is: ",0              ; Reposts dog age
       byebye        BYTE   "Thanks for passing by, have a great day!",0    ; Bye bye

.code
main PROC
; Greet the user
       mov    EDX, OFFSET hi_there        ; Set up for call to WriteString and greet the user
       call   WriteString
       call   Crlf
; Gets the user's age
       mov    EDX, OFFSET prompt1         ; Asks the user's age
       call   WriteString
       call   Crlf
       call   ReadInt                     ; Reads the users age. Age in EAX
       call   Crlf

; Calculate the dog years and stores the dog age
       mov    EBX, 7
       mul    EBX
       mov    age, EAX                    ; Stores the users dog age. Dog age also in EAX

; Reports the dog years and says bye
       mov    EDX, OFFSET output
       call   WriteString
       mov    EAX, age
       call   WriteDec
       call   Crlf
       mov    EDX, OFFSET byebye
       call   WriteString
       call   Crlf
       exit                               ;exit to operating system

main ENDP
END main
```

Address	Instruction	EIP	EIR	EID	MDR	MAR	EAX	EBX	ECX	EDX	age
0x2000	main PROC	0x2004	main PROC	main PROC	?	?	?	?	?	?	?
0x2004	mov EDX, OFFSET hi_there	0x2008	mov EDX, OFFSET hi_there	mov EDX, OFFSET hi_there	?	0x1004	?	?	?	0x1004	?
0x2008	call WriteString	0x200C	call WriteString	call WriteString	0	0x101A	?	?	?	0x1004	?
0x200C	call Crlf	0x2010	call Crlf	call Crlf	0	0x101A	?	?	?	0x1004	?
0x2010	Mov EDX, OFFSET prompt1	0x2014	Mov EDX, OFFSET prompt1	Mov EDX, OFFSET prompt1	0	0x101A	?	?	?	0x1005	?
0x2014	Call Writestring	0x2018	Call Writestring	Call Writestring	0	0x101A	?	?	?	0x1005	?
0x2018	Call Crlf	0x201C	Call Crlf	Call Crlf	0	0x101A	?	?	?	0x1005	?
0x201C	Call ReadInt	0x2020	Call ReadInt	Call ReadInt	0	0x101A	20	?	?	0x1005	?
0x2020	Call Crlf	0x2024	Call Crlf	Call Crlf	0	0x101A	20	?	?	0x1005	?
0x2024	Mov EBX, 7	0x2028	Mov EBX, 7	Mov EBX, 7	7	0x101A	20	7	?	0x1005	?
0x2028	Mul EBX	0x202C	Mul EBX	Mul EBX	140	0x101A	140	7	?	0x1005	?
0x202C	Mov age, EAX	0x2030	Mov age, EAX	Mov age, EAX	140	0x101A	140	7	?	0x1005	140
0x2030	Mov EDX, OFFSET output	0x2034	Mov EDX, OFFSET output	Mov EDX, OFFSET output	0	0x101A	140	7	?	0x1006	140
0x2034	Call Writestring	0x2038	Call Writestring	Call Writestring	0	0x101A	140	7	?	0x1006	140
0x2038	Mov EAX, age	0x203C	Mov EAX, age	Mov EAX, age	140	0x101A	140	7	?	0x1006	140
0x203C	Call WriteDec	0x2040	Call WriteDec	Call WriteDec	140	0x101A	140	7	?	0x1006	140
0x2040	Call Crlf	0x2044	Call Crlf	Call Crlf	140	0x101A	140	7	?	0x1006	140
0x2044	Mov EDX, OFFSET byebye	0x2048	Mov EDX, OFFSET byeybe	Mov EDX, OFFSET byeybe	0	0x101A	140	7	?	0x1007	140

0x2048	Call Writestring	0x204C	Call Writestring	Call Writestring	0	0x101A	140	7	?	0x1007	140
0x204C	Call Crlf	0x2050	Call Crlf	Call Crlf	0	0x101A	140	7	?	0x1007	140
0x2050	exit	0x2054	exit	exit	0	0x101A	140	7	?	0x1007	140