МИНОБРНАУКИ РОССИИ

САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ

ЭЛЕКТРОТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

«ЛЭТИ» ИМ. В.И. УЛЬЯНОВА(ЛЕНИНА)

КАФЕДРА МО ЭВМ

ОТЧЕТ

по лабораторной работе №2

по дисциплине «Организация ЭВМ и систем»

Тема: Изучение режимов адресации и формирования исполнительного адреса

Студентка гр. 1381 Деркачева Д.Я.

Преподаватель Ефремов М. А.

Санкт-Петербург

2022

Цель работы

Изучить основные принципы трансляции, отладки и выполнения программ на языке Ассемблера. Разобраться в используемых режимах адресации и получаемых результатах.

Текст задания

Лабораторная работа 2 предназначена для изучения режимов адресации, использует готовую программу lr2_comp.asm на Ассемблере, которая в автоматическом режиме выполняться не должна, так как не имеет самостоятельного функционального назначения, а только тестирует режимы адресации. Поэтому ее выполнение должно производиться под управлением отладчика в пошаговом режиме. В программу введен ряд ошибок, которые необходимо объяснить в отчете по работе, а соответствующие команды закомментировать для прохождения трансляции. Необходимо составить протокол выполнения программы в пошаговом режиме отладчика по типу таблицы 1 предыдущей лабораторной работы.

Ход выполнения работы

1. Изменение значения наборов vec1, vec2 и matr исходного файла на значения из файла lr2.dat для 7 варианта
2. Трансляция программы с последующим разбором, комментированием и закомменированием ошибок и предупреждений в тексте программы
    a. Ошибка lr2.asm(55): error A2052: Improper operand type (Неверный тип операнда)

        Строка 55: mov mem3, [bx]

        Тип операнда, нельзя читать из памяти и писать в память одной командой. В данном случае необходимо перевести

информацию из памяти в регистр, а затем уже перевести информацию из регистра в необходимый сегмент.

b. Предупреждение lr2.asm(62): warning A4031: Operand types must match (Несоответствие типов операндов)

Строка 62: mov cx, vec2[di]

Несоответствие типов операндов, cx – 1 слово, элемент vec2 – 1 байт.

c. Предупреждение lr2.asm(66): warning A4031: Operand types must match (Несоответствие типов операндов)

Строка 66: mov cx, matr[bx][di]

Несоответствие типов операндов, cx – 1 слово, элемент matr – 1 байт.

d. Ошибка lr2.asm(67): error A2055: Illegal register value (Незаконное использование регистра)

Строка 67: mov ax,matr[bx*4][di]

Здесь используется базово-индексная адресация. Такая форма адресации используется в тех случаях, когда в регистре находится адрес начала структуры данных, а доступ надо осуществить к какому-нибудь элементу этой структуры. При данном типе адресации надо сначала изменить значение регистра, затем уже переводить информацию.

e. Ошибка lr2.asm(87): error A2046: Multiple base registers (несколько базовых регистров)

Строка 87: mov ax,matr[bp+bx]

Нельзя складывать регистры bp и bx. Так как здесь оба регистра базовые, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра. Необходимо сначала в регистр bp занести общую сумму, затем уже производить смещение.

f. Ошибка lr2.asm(88): error A2047: Multiple index registers (несколько индексных регистров)

Строка 88: mov ax,matr[bp+di+si]

Нельзя складывать регистры di и si. Так как здесь два индексных регистра, надо сначала сложить значения регистров, и затем уже передавать информацию указателю из одного регистра. Необходимо сначала в регистр di занести общую сумму, затем уже производить смещение.

g. Ошибка lr2.asm(95): error A2006: Phase error between passes

Строка 95: Main ENDP

Данная ошибка свидетельствует о том, что в функции main содержатся ошибки.

3. Повторная трансляция программы и компоновка загрузочного модуля.

4. Выполнение программы в пошаговом режиме под управлением отладчика с фиксацией содержимого используемых регистров и ячеек памяти до и после выполнения команды.

Текст исходного файла программы lr2.asm

Текст исходной программы lr2.asm см. в приложении А.

Текст файла диагностического lr2.lst

Текст диагностического файла lr2.lst см. в приложении В.

Протокол пошагового исполнения программ

Табл.1. Протокол пошагового исполнения программы lr2

Адрес Команды	Символический код команды	16-ричный команды	Содержимое регистров и ячеек памяти	
			до выполнения	после выполнения
0000	Push DS	1E	(DS) = 19F5 (SP) = 0018 (IP) = 0000 (Stack) +0 0000	(DS) = 19F5 (SP) = 0016 (IP) = 0001 (Stack) +0 19F5
0001	Sub AX, AX	2BC0	(AX) = 0000 (IP) = 0001	(AX) = 0000 (IP) = 0003
0003	Push AX	50	(AX) = 0000 (SP) = 0016 (IP) = 0003 (Stack) +0 19F5 (Stack) +2 0000	(AX) = 0000 (SP) = 0014 (IP) = 0004 (Stack) +0 0000 (Stack) +2 19F5
0004	Mov AX, 1A07	B8071A	(AX) = 0000 (IP) = 0004	(AX) = 1A07 (IP) = 0007
0007	Mov DS, AX	8ED8	(DS) = 19F5	(DS) = 1A07

			(AX) = 1A07 (IP) = 0007	(AX) = 1A07 (IP) = 0009
0009	Mov AX, 01F4	B8F401	(AX) = 1A07 (IP) = 0009	(AX) = 01F4 (IP) = 000C
000C	Mov CX, AX	8BC8	(AX) = 01F4 (CX) = 00B0 (IP) = 000C	(AX) = 01F4 (CX) = 01F4 (IP) = 000E
000E	Mov BL, 24	B324	(BX) = 0000 (IP) = 000E	(BX) = 0024 (IP) = 0010
0010	Mov BH, CE	B7CE	(BX) = 0024 (IP) = 0010	(BX) = CE24 (IP) = 0012
0012	Mov [0002], FFCE	C7060200CE FF	(IP) = 0012	(IP) = 0018
0018	Mov BX, 0006	BB0600	(BX) = CE24 (IP) = 0018	(BX) = 0006 (IP) = 001B
001B	Mov [0000], AX	A30000	(AX) = 01F4 (IP) = 001B	(AX) = 01F4 (IP) = 001E
001E	Mov AL, [BX]	8A07	(IP) = 001E (AX) = 01F4	(IP) = 0020 (AX) = 0126
0020	Mov AL, [BX+03]	8A4703	(IP) = 0020 (AX) = 011F	(IP) = 0023 (AX) = 0123
0023	Mov CX, [BX+03]	8B4F03	(CX) = 01F4 (IP) = 0023	(CX) = 1F23 (IP) = 0026

0026	Mov DI, 0002	BF0200	(DI) = 0000 (IP) = 0026	(DI) = 0002 (IP) = 0029
0029	Mov AL, [000E+DI]	8A850E00	(IP) = 0029 (AX) = 0123	(IP) = 002D (AX) = 01BA
002D	Mov BX, 0003	BB0300	(BX) = 0006 (IP) = 002D	(BX) = 0003 (IP) = 0030
0030	Mov AL, [0016+BX +DI]	8A811600	(AX) = 01BA (IP) = 0030	(AX) = 01F9 (IP) = 0034
0034	Mov AX, 1A07	B8071A	(AX) = 01F9 (IP) = 0034	(AX) = 1A07 (IP) = 0037
0037	Mov ES, AX	8EC0	(AX) = 1A07 (ES) = 19F5 (IP) = 0037	(AX) = 1A07 (ES) = 1A07 (IP) = 0039
0039	Mov AX, ES:[BX]	268B07	(AX) = 1A07 (IP) = 0039	(AX) = 00FF (IP) = 003C
003C	Mov AX, 0000	B80000	(AX) = 00FF (IP) = 003C	(AX) = 0000 (IP) = 003F
003F	Mov ES, AX	8EC0	(AX) = 0000 (ES) = 1A07 (IP) = 003F	(AX) = 0000 (ES) = 0000 (IP) = 0041
0041	Push DS	1E	(DS) = 1A07 (IP) = 0041 (SP) = 0014	(DS) = 1A07 (IP) = 0042 (SP) = 0012

			(Stack) +0 0000 (Stack) +2 19F5 (Stack) +4 0000	(Stack) +0 1A07 (Stack) +2 0000 (Stack) +4 19F5
0042	Pop ES	07	(ES) = 0000 (IP) = 0042 (SP) = 0012 (Stack) +0 1A07 (Stack) +2 0000 (Stack) +4 19F5	(ES) = 1A07 (IP) = 0043 (SP) = 0014 (Stack) +0 0000 (Stack) +2 19F5 (Stack) +4 0000
0043	Mov CX, ES:[BX-01]	268B4FFF	(CX) = 1F23 (IP) = 0043	(CX) = FFCE (IP) = 0047
0047	Xchg AX, CX	91	(AX) = 0000 (CX) = FFCE (IP) = 0047	(AX) = FFCE (CX) = 0000 (IP) = 0048
0048	Mov DI, 0002	BF0200	(DI) = 0002 (IP) = 0048	(DI) = 0002 (IP) = 004B
004B	Mov ES:[BX+DI], AX	268901	(IP) = 004B	(IP) = 004E
004E	Mov BP, SP	8BEC	(SP) = 0014 (BP) = 0010 (IP) = 004E	(SP) = 0014 (BP) = 0014 (IP) = 0050
0050	Push 01F4	FF360000	(SP) = 0014 (IP) = 0050 (Stack) +0 0000 (Stack) +2 19F5	(SP) = 0012 (IP) = 0054 (Stack) +0 01F4 (Stack) +2 0000

			(Stack) +4 0000	(Stack) +4 19F5
0054	Push FFCE	FF360200	(SP) = 0012 (IP) = 0054 (Stack) +0 01F4 (Stack) +2 0000 (Stack) +4 19F5 (Stack) +6 0000	(SP) = 0010 (IP) = 0058 (Stack) +0 FFCE (Stack) +2 01F4 (Stack) +4 0000 (Stack) +6 19F5
0058	Mov BP, SP	8BEC	(BP) = 0014 (IP) = 0058	(BP) = 0010 (IP) = 005A
005A	Mov DX, [BP+02]	8B5602	(DX) = 0000 (IP) = 005A	(DX) = 01F4 (IP) = 005D
005D	Ret Far 0002	CA0200	(CS) = 1A0A (IP) = 005D (SP) = 0010 (Stack) +0 FFCE (Stack) +2 01F4 (Stack) +4 0000 (Stack) +6 19F5	(CS) = 01F4 (IP) = FFCE (SP) = 0016 (Stack) +0 19F5 (Stack) +2 0000 (Stack) +4 0000 (Stack) +6 0000

Выводы по работе

В ходе выполнения лабораторной работы были получены основные навыки программирования на ассемблере, изучены основные режимы адресации памяти.

Приложение А

Текст исходного файла lr2

```
; Программа изучения режимов адресации процессора IntelX86
EOL EQU '$'
ind EQU 2
n1 EQU 500
n2 EQU -50
; Стек программы
AStack SEGMENT STACK
 DW 12 DUP(?)
AStack ENDS
; Данные программы
DATA SEGMENT
; Директивы описания данных
mem1 DW 0
mem2 DW 0
mem3 DW 0
vec1 DB 21,22,23,24,28,27,26,25
vec2 DB 40,50,-40,-50,20,30,-20,-30
matr DB 5,6,-8,-7,7,8,-6,-5,1,2,-4,-3,3,4,-2,-1
DATA ENDS
; Код программы
CODE SEGMENT
 ASSUME CS:CODE, DS:DATA, SS:AStack
; Головная процедура
Main PROC FAR
```

```
 push DS
 sub AX,AX
 push AX
 mov AX,DATA
 mov DS,AX
; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ НА УРОВНЕ СМЕЩЕНИЙ
; Регистровая адресация
 mov ax,n1
 mov cx,ax
 mov bl,EOL
 mov bh,n2
; Прямая адресация
 mov mem2,n2
 mov bx,OFFSET vec1
 mov mem1,ax
; Косвенная адресация
 mov al,[bx]
 mov mem3,[bx]
; Базированная адресация
 7
 mov al,[bx]+3
 mov cx,3[bx]
; Индексная адресация
 mov di,ind
 mov al,vec2[di]
 mov cx,vec2[di]
; Адресация с базированием и индексированием
```

```
 mov bx,3
 mov al,matr[bx][di]
 mov cx,matr[bx][di]
 mov ax,matr[bx*4][di]
; ПРОВЕРКА РЕЖИМОВ АДРЕСАЦИИ С УЧЕТОМ СЕГМЕНТОВ
; Переопределение сегмента
; ------ вариант 1
 mov ax, SEG vec2
 mov es, ax
 mov ax, es:[bx]
 mov ax, 0
; ------ вариант 2
 mov es, ax
 push ds
 pop es
 mov cx, es:[bx-1]
 xchg cx,ax
; ------ вариант 3
 mov di,ind
 mov es:[bx+di],ax
; ------ вариант 4
 mov bp,sp
 mov ax,matr[bp+bx]
 mov ax,matr[bp+di+si]
; Использование сегмента стека
 push mem1
 push mem2
```

```
 mov bp,sp
 mov dx,[bp]+2
 ret 2
Main ENDP
CODE ENDS
 END Main
```

Приложение В

Текст диагностического файла lr2

```
Microsoft (R) Macro Assembler Version 5.10                  10/2/22
06:30:52
                                                              Page
1-1


                    ; ÐŸÑ ÐŸÐ³Ñ Ð°ÐŒÐŒÐ° ÐžÐ·Ñƒ‡ÐµÐœÐžÑ  Ñ ÐµÐ¶ÐžÐ
                    ŒÐŸÐ² Ð°ÐŽÑ ÐµÑ Ð°Ñ†ÐžÐž Ð¿Ñ ÐŸÑ†ÐµÑ Ñ ÐŸÑ Ð° I
                    ntelX86

 = 0024                  EOL EQU '$'
 = 0002                  ind EQU 2
 = 01F4                  n1 EQU 500
 =-0032                  n2 EQU -50

                    ; Ð¡Ñ,ÐµÐ° Ð¿Ñ ÐŸÐ³Ñ Ð°ÐŒÐŒÑ‹
 0000                    AStack SEGMENT STACK
 0000  000C[                 DW 12 DUP(?)
        ????
               ]

 0018                    AStack ENDS

                    ; Ð"Ð°ÐœÐœÑ‹Ðµ Ð¿Ñ ÐŸÐ³Ñ Ð°ÐŒÐŒÑ‹
 0000                    DATA SEGMENT

                    ; Ð"ÐžÑ ÐµÐ°Ñ,ÐžÐ²Ñ‹ ÐŸÐ¿ÐžÑ Ð°ÐœÐžÑ  ÐŽÐ°ÐœÐœÑ
                    ‹Ñ…
```

```
 0000  0000                mem1 DW 0
 0002  0000                mem2 DW 0
 0004  0000                mem3 DW 0
 0006  15 16 17 18 1C 1B vec1 DB 21,22,23,24,28,27,26,25

       1A 19
 000E  28 32 D8 CE 14 1E vec2 DB 40,50,-40,-50,20,30,-20,-30
       EC E2
 0016  05 06 F8 F9 07 08 matr                                    DB
5,6,-8,-7,7,8,-6,-5,1,2,-4,-3,3,4,-2,-1
       FA FB 01 02 FC FD
       03 04 FE FF

 0026                      DATA ENDS

                   ; ÐšÐŸÐŽ Ð¿Ñ ÐŸÐ³Ñ Ð°ÐŒÐŒÑ‹
 0000                      CODE SEGMENT
                           ASSUME CS:CODE, DS:DATA, SS:AStack

                   ; Ð“ÐŸÐ»ÐŸÐ²ÐœÐ°Ñ  Ð¿Ñ ÐŸÑ†ÐµÐŽÑƒÑ Ð°
 0000                      Main PROC FAR
 0000  1E                 push DS
 0001  2B C0                    sub AX,AX
 0003  50                 push AX
 0004  B8 ---- R                mov AX,DATA
 0007  8E D8                    mov DS,AX

                   ; ÐŸÐ Ð'Ð Ð ÐšÐ  Ð Ð Ð-Ð˜ÐœÐ Ð' Ð Ð"Ð Ð Ð¡Ð Ð
                   ŠÐ˜Ð˜ Ð Ð  Ð£Ð Ð Ð'Ð Ð  Ð¡ÐœÐ Ð©Ð Ð Ð˜Ð™
                   ; Ð ÐµÐ³ÐžÑ Ñ‚Ñ ÐŸÐ²Ð°Ñ  Ð°ÐŽÑ ÐµÑ Ð°Ñ†ÐžÑ
 0009  B8 01F4                  mov ax,n1
 000C  8B C8                    mov cx,ax
 000E  B3 24                    mov bl,EOL
Microsoft (R) Macro Assembler Version 5.10                  10/2/22
06:30:52
                                                             Page
1-2


 0010  B7 CE                    mov bh,n2
                   ; ÐŸÑ Ñ ÐŒÐ°Ñ  Ð°ÐŽÑ ÐµÑ Ð°Ñ†ÐžÑ
 0012  C7 06 0002 R FFCE        mov mem2,n2
 0018  BB 0006 R                mov bx,OFFSET vec1
 001B  A3 0000 R                mov mem1,ax
```

```
                    ; ĐšĐŸÑ Đ²ĐµĐœĐœĐ°Ñ  Đ°ĐŽÑ ĐµÑ Đ°Ñ†ĐžÑ
001E  8A 07                 mov al,[bx]
                         ;mov mem3,[bx]       ;ĐŒÑ‹ ĐœĐµ ĐŒĐŸ
                    Đ¶ĐµĐŒ Đ² Ñ ĐµĐ³ĐŒĐµĐœÑ, ĐŽĐ°ĐœĐœÑ‹Ñ… ĐŸÑ,Đ¿Ñ Đ
                    °Đ²Đ»Ñ Ñ,ÑŒ ĐŽĐ°ĐœĐœÑ‹Đµ ĐœĐ°Đ¿Ñ Ñ ĐŒÑƒÑ , Ñ,ĐŸ
                    Đ»ÑŒĐ°ĐŸ Ñ‡ĐµÑ ĐµĐ· Ñ ĐµĐ³ĐžÑ Ñ,Ñ Ñ‹
                    ; Đ ‘Đ°Đ·ĐžÑ ĐŸĐ²Đ°ĐœĐœĐ°Ñ  Đ°ĐŽÑ ĐµÑ Đ°Ñ†ĐžÑ
0020  8A 47 03                  mov al,[bx]+3
0023  8B 4F 03                  mov cx,3[bx]
                    ; Đ˜ĐœĐŽĐµĐ°Ñ ĐœĐ°Ñ  Đ°ĐŽÑ ĐµÑ Đ°Ñ†ĐžÑ
0026  BF 0002               mov di,ind
0029  8A 85 000E R          mov al,vec2[di]
                         ;mov cx,vec2[di]    ;cx- DW, vec2[d
                    i]- DB
                    ; Đ ĐŽÑ ĐµÑ Đ°Ñ†ĐžÑ  Ñ  Đ±Đ°Đ·ĐžÑ ĐŸĐ²Đ°ĐœĐŽĐµĐ
                    Œ Đž ĐŽĐœĐŽĐµĐ°Ñ ĐžÑ ĐŸĐ²Đ°ĐœĐŽĐµĐŒ
002D  BB 0003               mov bx,3
0030  8A 81 0016 R          mov al,matr[bx][di]
                         ;mov cx,matr[bx][di]     ;cx- DW, matr[b
                    x][di]- DB
                         ;mov ax,matr[bx*4][di]   ;Ñƒ ĐŒĐœĐŸĐ¶Đ°Ñ,
                    ÑŒ ĐœĐ° Đ°ĐŸĐœÑ Ñ,Đ°ĐœÑ,Ñƒ ĐŒÑ‹ ĐŒĐŸĐ¶ĐµĐŒ Ñ,ĐŸ
                    Đ»ÑŒĐ°ĐŸ ĐžĐœĐŽĐµĐ°Ñ ĐœÑ‹Đµ Ñ ĐµĐ³ĐžÑ Ñ,Ñ Ñ‹, Ñ
                    ,.Đµ. si, di ?????

                    ; ĐŸĐ Đ Đ’Đ Đ ĐšĐ  Đ Đ Đ-Đ˜ĐœĐ Đ’ Đ Đ”Đ Đ Đ¡Đ Đ
                    ŠĐ˜Đ˜ Đ¡ Đ£Đ§Đ Đ¢Đ Đœ Đ¡Đ Đ“ĐœĐ Đ Đ¢Đ Đ’
                    ; ĐŸĐµÑ ĐµĐŸĐ¿Ñ ĐµĐŽĐµĐ»ĐµĐœĐžĐµ Ñ ĐµĐ³ĐŒĐµĐœÑ,
                    Đ°
                    ; ------ Đ²Đ°Ñ ĐžĐ°ĐœÑ, 1
0034  B8 ---- R              mov ax, SEG vec2
0037  8E C0                 mov es, ax
0039  26: 8B 07              mov ax, es:[bx]
003C  B8 0000                mov ax, 0
                    ; ------ Đ²Đ°Ñ ĐžĐ°ĐœÑ, 2
003F  8E C0                 mov es, ax
0041  1E                 push ds
0042  07                 pop es
0043  26: 8B 4F FF           mov cx, es:[bx-1]
0047  91                 xchg cx,ax
                    ; ------ Đ²Đ°Ñ ĐžĐ°ĐœÑ, 3
0048  BF 0002               mov di,ind
004B  26: 89 01              mov es:[bx+di],ax
                    ; ------ Đ²Đ°Ñ ĐžĐ°ĐœÑ, 4
```

```
 004E  8B EC                   mov bp,sp
                             ;mov ax,matr[bp+bx]      ;ÐŒÑ‹ ÐœÐµ ÐŒÐŸ
                        Ð¶ÐµÐŒ Ð² Ñ Ð°Ñ Ñ^ÐžÑ ÐµÐœÐœÐŸÐ¹ ÐžÐœÐŽÐµÐ°Ñ Ð°
                        Ñ†ÐžÐž Ñ Ð°Ð»Ð°ÐŽÑ‹Ð²Ð°Ñ,Ñ‹ ÐœÐµÑ Ð°ÐŸÐ»Ñ‹Ð°ÐŸ
                        Ñ ÐµÐ³ÐžÑ Ñ,Ñ ÐŸÐ²
Microsoft (R) Macro Assembler Version 5.10                  10/2/22
06:30:52
                                                                Page
1-3


                             ;mov ax,matr[bp+di+si]
                        ; Ð˜Ñ Ð¿ÐŸÐ»Ñ‹Ð·ÐŸÐ²Ð°ÐœÐžÐµ Ñ ÐµÐ³ÐŒÐµÐœÑ,Ð° Ñ
                         Ñ,ÐµÐ°Ð°
 0050  FF 36 0000 R            push mem1
 0054  FF 36 0002 R            push mem2
 0058  8B EC                   mov bp,sp
 005A  8B 56 02                     mov dx,[bp]+2
 005D  CA 0002                 ret 2
 0060                       Main ENDP
 0060                       CODE ENDS
                            END Main
Microsoft (R) Macro Assembler Version 5.10                  10/2/22
06:30:52

Symbols-1


Segments and Groups:

                N a m e             Length     Align    Combine
Class

ASTACK . . . . . . . . . . . . . .   0018 PARA STACK
CODE . . . . . . . . . . . . . . .   0060 PARA NONE
DATA . . . . . . . . . . . . . . .   0026 PARA NONE

Symbols:

                N a m e             Type  Value     Attr

EOL  . . . . . . . . . . . . . . .   NUMBER    0024

IND  . . . . . . . . . . . . . . .   NUMBER    0002
```

```
MAIN . . . . . .   F PROC    0000 CODE Length     =
0060
MATR . . . . . .   L BYTE    0016 DATA
MEM1 . . . . . .   L WORD    0000 DATA
MEM2 . . . . . .   L WORD    0002 DATA
MEM3 . . . . . .   L WORD    0004 DATA

N1 . . . . . .     NUMBER    01F4
N2 . . . . . .     NUMBER    -0032

VEC1 . . . . . .   L BYTE    0006 DATA
VEC2 . . . . . .   L BYTE    000E DATA

@CPU . . . . . .   TEXT  0101h
@FILENAME  . . . . . .   TEXT  LR2
@VERSION . . . . . .   TEXT  510


     92 Source  Lines
     92 Total   Lines
     19 Symbols

  47808 + 459452 Bytes symbol space free

      0 Warning Errors
      0 Severe  Errors
```