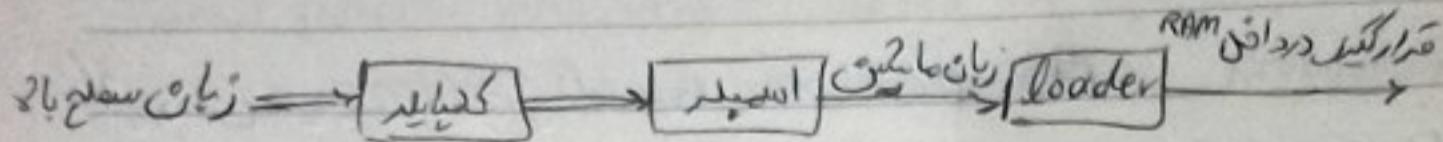


پردازشگر AVR و اسپیلر
Intel پردازشگر AVR
میکروکنترلر دارای

درینی پردازشگر باقی می‌شود microcontroller → microprocessor

صحیح افهام → K.L. Short, Microprocessors and programmed logic Barry B.Bray The Intel 8086/8088, AVR میکروکنترلر



← دستگاه اینسلا نرم افزار و سمت افزار (instruction set) می‌باشد
فازهای احری دستورالعمل همراه

needs PC, IR and other registers

Instruction Fetch

separate opcode and... ← Instruction Decode

execute instruction in Alu ← Execution after this Update Registers

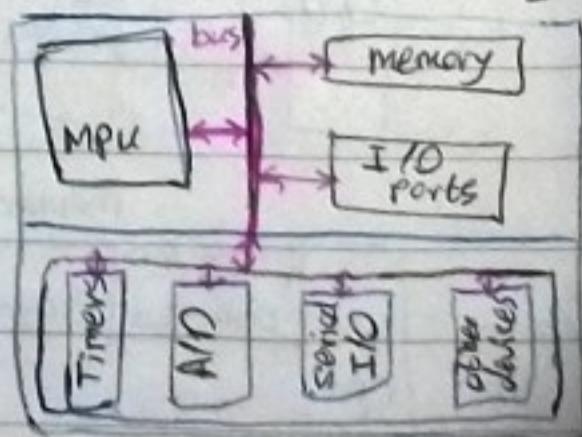
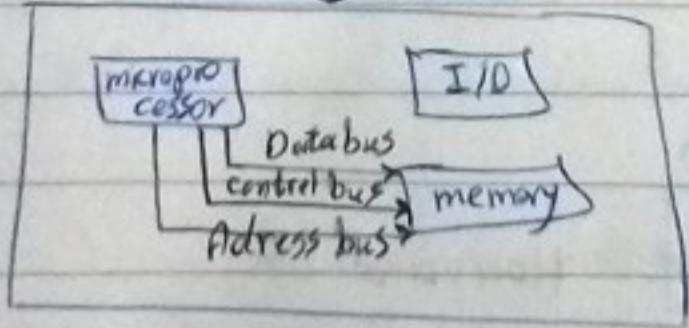
میکروپردازشگر بی جیب فریمی است که (رسانیت register, Control unit و منطقی است) واحدهای واحد حساب و منطقی شکل شده است microprocessor

← حافظه دارد و تاکریز ندارد، I/O ندارد (I/O port)

← از طریق لیست رده معنی و مانیتور با دستگاه خارج در اینجا این میکروکنترلر (microcontroller) دارای

حافظه I/O های می‌باشد

(MCU) microController



و Embedded cmp
دستگاههای که در نگاه اول عجالی هستند پس از میکرو پردازش

Year: Month: Day:

Subject:

Complex Instruction Set Computer

پردازندهای CISC و RISC

Reduced Instruction set computer

Complex
کمترین تردد

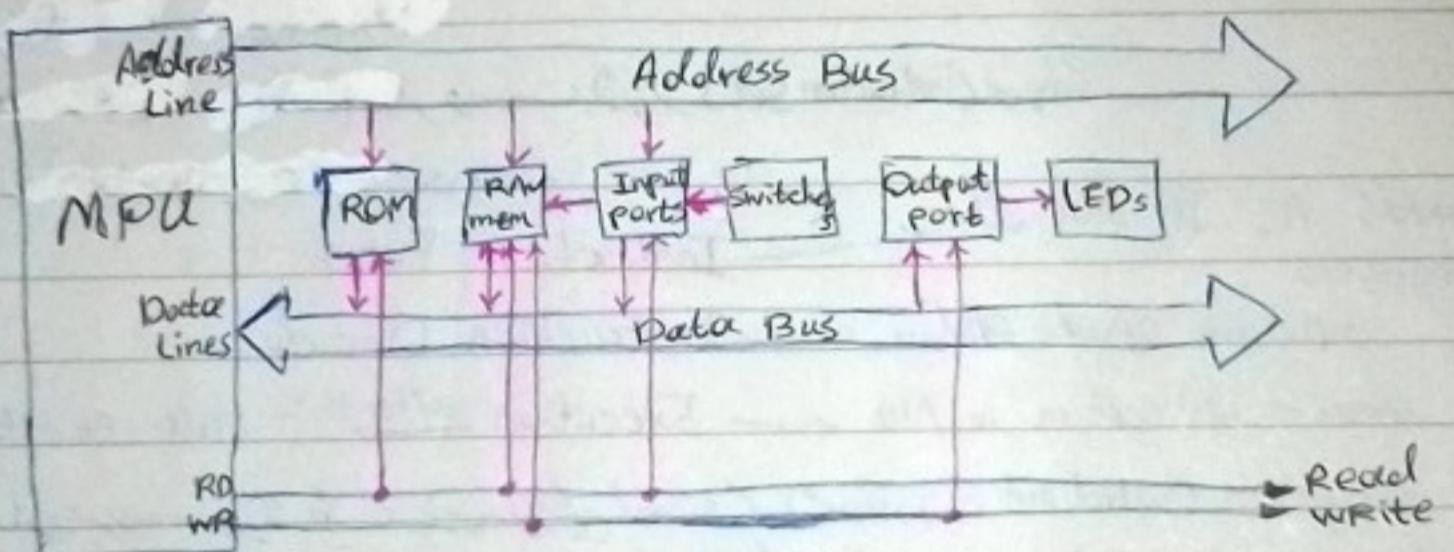
Reduce
کمترین تردد

$add [100], [200], [300]$

load AC, [100]

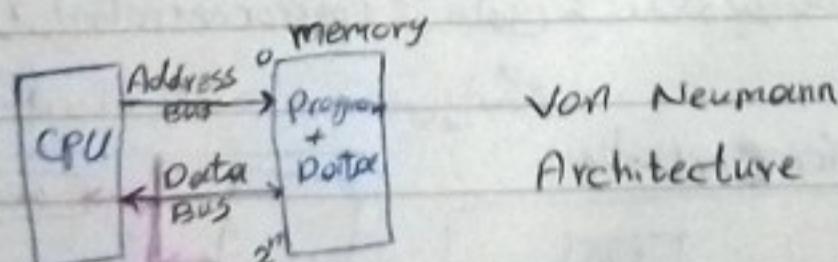
$mem[100] \leftarrow (mem[200] + mem[300])$

Microprocessor-Based System

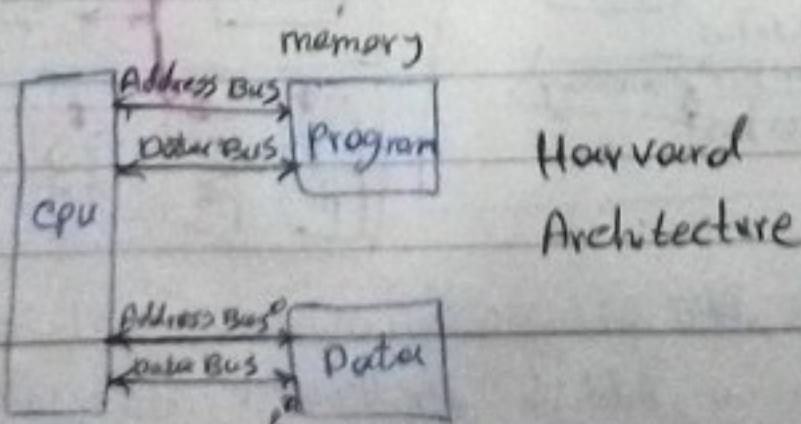


نکس از پردازندهای سیستمی System Bus کو microprocessor-Based سیستم،
Control Bus، Data Bus، Address Bus

و PCI بود که Bus آن که سیستم Bus نام دارد ISI -
و MC68000 Architecture



Von Neumann
Architecture



Harvard
Architecture

پردازنده های کامپیوتر های خانگی General purpose هستند
 پردازنده های Embedded فردایی که انجام صورت داده و آن کار را قابل انجام نمایند
 این های هستند که تا خیر دن آن ها معمولی اس ای Application، Realtime و Application
 مثلاً "دیجیتال ویدیو" اگر لذت برداری داشت رفع دهنده، قابل تعلیم بود هر آنکه پخش ویدیو
 ای ای Realtime

Special purpose processors: Using on Special works
 like Consoles, Netbooks, ...

یک کلاس از Applications ها را من توانند اجرا کنند

System on a chip (SOC): Set of systems like processor and
 memory, I/O and so on that assembled on a package
 Actually SOC is a strong instance of MicroController

: (X86) Intel 8086

با این معنا 2MHz یا 2MHz برابر با 2MHz clock، پردازنده 2MHz Clock rate

است که کلی پردازنده در حدود ۲ میلیون پیغام بینی دارد

پردازنده حافظه را با استفاده از آدرس دهنده Control Bus, Data Bus, Address Bus

است دست دست دهنده این محتواست که رجیستر های پردازنده نیز آن هستند

پردازنده های 8085، 8080 که نسبت به 8086 کمتر قدرت دارند ساخت کنن

پیشرفت حاصل شد

Data type اگر مقدار 8 بیتی باشد مانند Int را دارد و اگر 16 بیتی باشد مانند long

External data bus داشت 16 بیت بعد که باعث شد که جیب کراس های از آن در میان

کاری میشکنند در سایر روشی دستگاه های خوبی نداشتم بود، این مشکل در پردازنده 8088

رفع شد

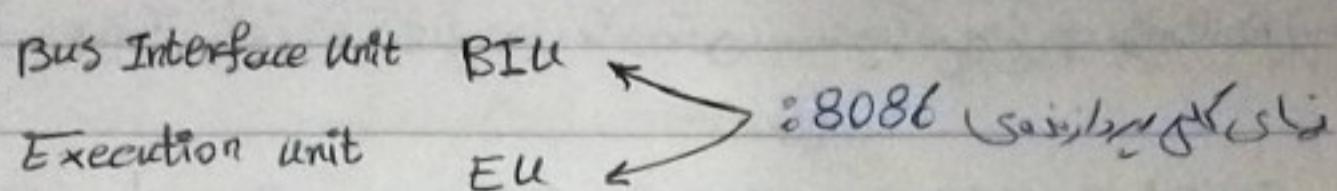
در پردازنده های 80286 این فایل Virtual mem

در پردازنده های 80486 این فایل Cache

PCB (Printed Circuit Board) : like MotherBoards

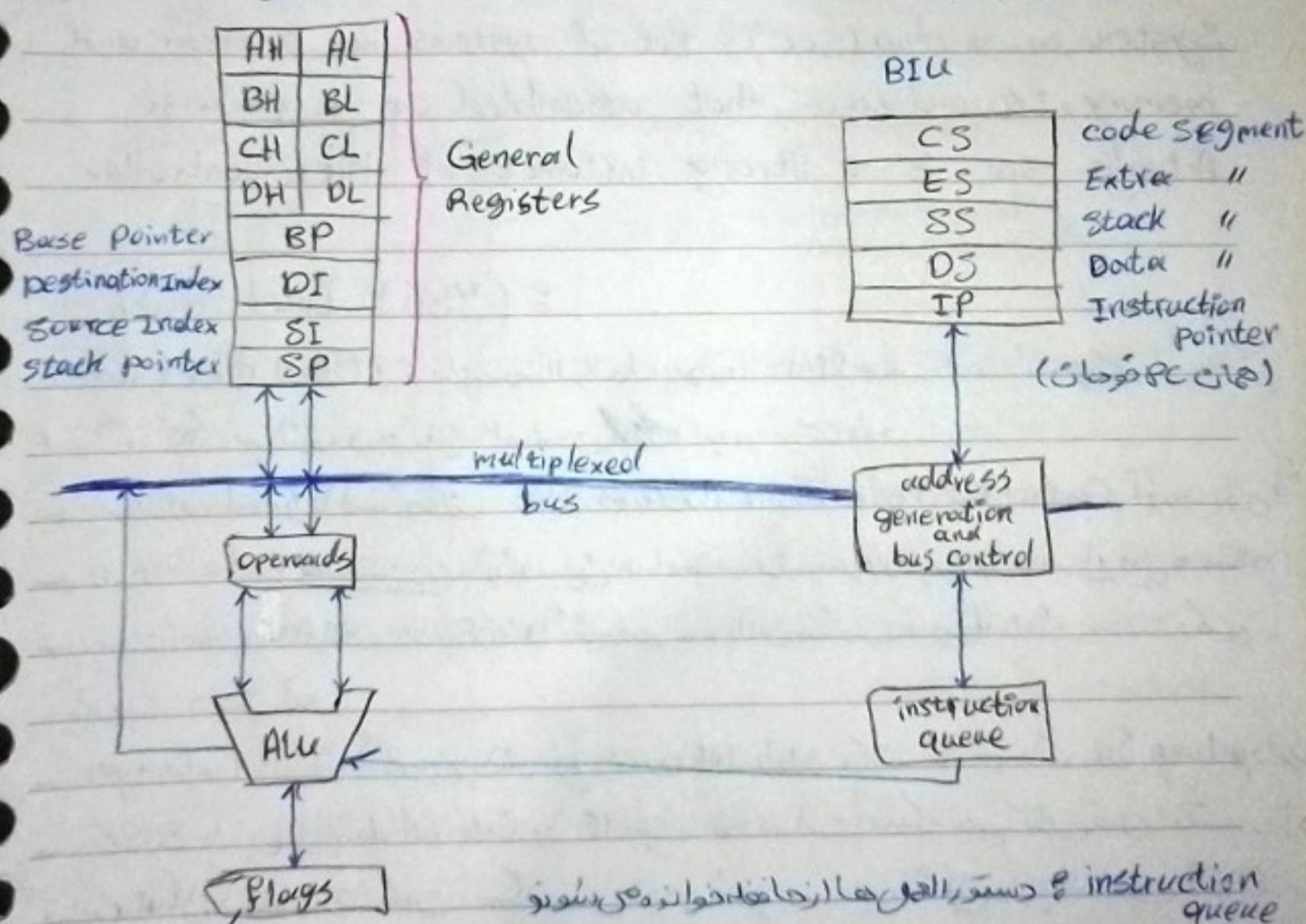
Virtual memory ← مفهومی نهایت سد حافظه

protected سیستم عامل در حالت کاری کند یک حالت real و دو حالت



Fetch دستورات از BIU کردن

EU از EU دستورات



instruction queue دستورات را از BIU دریافت کرده و آنها را در همان وسیله اجرا می کند

هر دو بخش EU و BIU همان وسیله را اجرا می کنند

کاری کند

کامپریسر 16 بیتی
کسر کوچک و مضبوط رسانی
می دهد

٥ معادل يا Sign => درجه حراري معاصر Alfa هميت سعود
١ درجه حراري " " " منطقه " "

T معادل با Trap \Leftrightarrow یک نوع interrupt است و کاری می‌کند که دستور العمل هادر (CPU) را کنترل نماید (معنی می‌گیرد که رایانه این دستور را پس از این کار را در این دستور اجرا نماید).
 I معادل با Interrupt : در صورتی که این Flag ۰ باشد بروزگرد و قطعه های اینجا نمی‌توانند دستور اجرا شوند. این عمل mask کردن گفته می‌شود و فقط در صورتی که وقوعه یا سیگنال داده شده باشد این Flag ۱ خواهد بود.
 D معادل با Direction : از چیزی یه راست یعنی راز راست به چیزی پسند می‌باشد که در همان استعداد می‌تواند String

پنجم: مثال A برای حالت کنندگان در حالات BCD بیان می‌شود

• DI، SI، BP، SP (جسیر)

SP : بہاڑی یونیٹ اسکارہ میں کتو (بہاڑی آدرس یا لارائیز گرسنگ و آدرس دھی عنصر مستحق)

BP من گذشت ای ساره بیک

و SI و DI که برای انرژی اکتیویتی در آزادی ها استفاده می شوند

سورات اسماعیل :

MON dess, Src

MDV, 1-1

نحو: این دسته الفعل را فلماً اینها که زمان تذارد

(ذیه و معدار بُلْبَت) را پس تعلق بدلی ریسیرهای سکفت انجام داد immediate load

- انداره‌ی میدان و مقصد را برای این راسته
و قعده برجی رجیسترهاي عمومي هم توان انجام داد 

- اندازه‌ی میدان و مقصد پایر پر این را سئد

نئی سلطان داڑھو کے عدد
نایت دھنیہ مکر

نکته: باید مقدار کذا بر رجیسترهای ساخته از یک رجیستر و ایندراستفاده من کنیم

MOV AX, 2345H.

MOV DS, AX

رجیستر
عملونهای پسند
خانه حافظه
Immediate
ADD des,src
۴- دستور

پسند: MOV AL, 55H

MOV CL, 23H

(درست) مدار
AL = 78H

ADD AL, CL →

AL = AL + CL

(32 bit)

تعلاجی های عمل و نهاد باید برابر باشد

اصراحتاً باید پرتابه

dash

→ R (Report) همه رجیسترها را مشاهده کنیم

interrupt ۳

-A (Assemble) توسعه کرد اسembly

پیغام شماره ۳ را اپل کن

-U unassemble کردن کدهای حافظه

-G (GO) شروع برنامه از حافظه

صلیدستورالعنی G است با این خطوات که لیکن بخوبی اپل کنند

-T (Trace) Trap

(برای debug کردن کد)

-F Fill (برای کردن آدرس صاریح و جا فلتم)

-E Enter (لایود کردن لیست از ۱۶)

-D (Dump) خواندن فانکچر از حافظه

که از آدرس آغاز و تعدادی که دارد من سود

Code Segment (program memory)

رجیسترهاي Segment

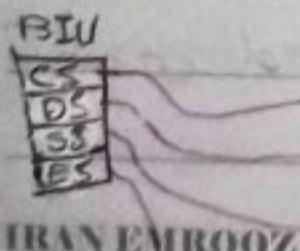
Data Segment (data ..)

دافتاری پردازندهای x86

Stack memory (stack segment)

بها رفته رمی و قصیم می شود

Extra .. (Extra ..)



آدرس آغاز هر کدام از قسمت ها در segment کمتر گیر (ضخیم از ۱۶ است)

نکته در داده Stack داره های آبی ریخته هی سود

انواع آدرس دهنده در پردازنده 8086

$$AL = AL + 45H$$

$$Addl AL, 45H$$

مقداری آدرس دهنده روش های مختلف مسفن کوت عملوند

آدرس دهنده مسفن (immediate)

نکته در صور آدرس دهنده رجیستر های امنیع و مقعد باید مرتب باشند

move AL, BX X error

۱- روش آدرس دهنده رجیستر غیر مستقیم

MOV AL, [BX]

memory

MOV 00H

BX

عملوند داخل طبقه قدر من کرد

سواری رجیستر داخل دستره هم قدر آدرس کرد

آدرس دهنده داخل رجیستر است

داخل دستور
ققداره
راهن تحریک
Register

۲- روش آدرس دهنده Direct: عملوند داخل خانه ای از حافظه

Add AL, [4411] و آدرس خانه داخل دستور العمل

کد

روش آدرس دهنده Register: عملوند داخل رجیستر

Add AX, BX سواری رجیستر داخل دستور العمل

Base Address با طبقه کوت

MOV CX, [BX] + 10 : Based relative روش آدرس دهنده

offset دستور العمل

$\Rightarrow \text{MOV CX, 10[BX]} \equiv \text{MOV [BX+10]}$

MOV DX, [SI] + 5

: Indexed relative

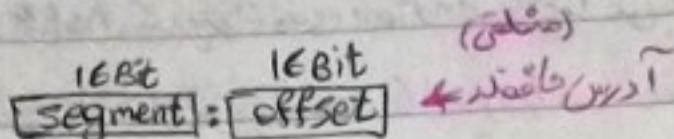
آدرس دهنده

MOV CL, [BX][DI] + 8

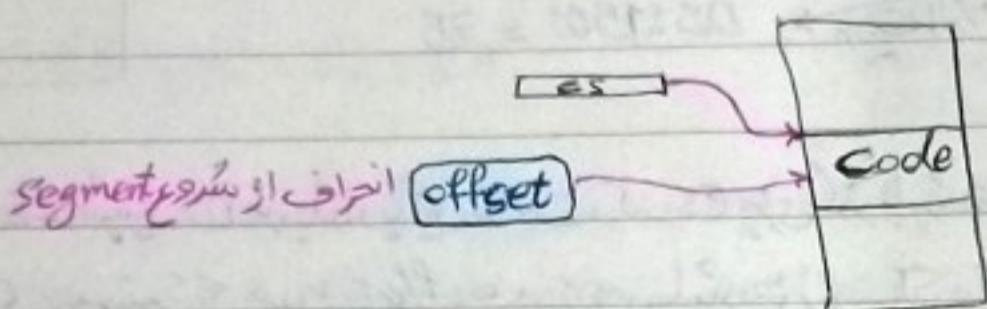
آدرس دهنده Based Indexed

MOV CH, [BX][SI] + 20

در پردازنده 8086 چون حافظه 1MB ص با سر برای آدرس دهن این حافظه به 20 بیت تواند داریع و پنهان متغیر (و عدد رجیستر 16 بیتی را استفاده می کند) (رجیستری پردازنده همان 16 بیت داشتند)



- نحوه این است آورت آدرس فیزیکی حافظه با استفاده از offset و segment
- ۱- سیستم دادن 4 بیت segment به سمت چیز (یا اگر Hex بودن ضرور ۱۶ بیت شود یعنی یک ۳۲ بیتی عدد Hex من نویسیم)
 - ۲- عدد مابین ۰ تا ۶۴K را با offset پیچید کنیم
- نکته: هر آدرس در پردازنده 8086 دو قسم است offset و segment



مثال: آدرس رویه را به آدرس فیزیکی تبدیل کنید
24F6:634A → آدرس مطلق

$$\begin{array}{r}
 24F6 \\
 + 634A \\
 \hline
 3B3AA
 \end{array}$$

→ داده ای آدرس

→ داده ای آدرس

→ 24F60

→ 24F60

→ FFFF

→ 34F5F

هر Segment در حافظه 64K ۶۴ کانه من پاسد

داتا و داده (Data Segment)

ADD AL, [0200]

offset → آدرس مطلق

segment = DS → DS:0200

در این قسم رجیسترهاي DI, SI, BX از اسناد offset اضافه شود

DS:0200

IRAN EMROOZ

ADD AL, [BX]

آدرس مطلق معادل DS:BA04

کانه حافظه ۱ بیت است

دستورالعمل INC : مولند آن (یک محتوی) را باشد که یک واحد آن اضافه من کند

$INC BX // BX++$

بردازندگی ۸۰۸۶ بینهایت little endian من پاسد

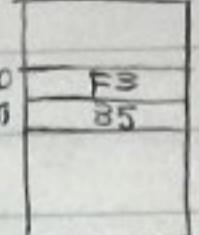
درین دارند های little endian ۱۶ بیت را در حافظه نصواعیم بینهایت چون حافظه آسی است پایه این عدد ۱۶ بیت را در دو قله از حافظه بینهایت ولذا چون بردازندگی من پاسد است که این را در خانه اعلی و اینست با ارزشی را در خانه پایین رسانی کنند

MOV AX, 35F3H $\rightarrow AX = 35F3H$

MOV [1500], AX

پیش کم ارزشی
خانه ای از حافظه
پیش با ارزش

DS = 1500
DS = 1501



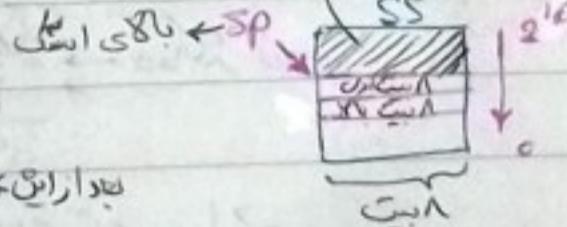
ES سیم کل من پاسد پاین مقادیر که ES پرای کار با رسته ها من پاسد

Offset پیشنهادی همان DS من پاسد یعنی BX, DI, SI

→ سلکت SS قسمتی offset درین من تعداد دو، چیزی SP و BP باشد

که او Stack پرای تکه درین افعالات function هادر هنگام تابع های تودر رها استفاده می شود
که های داخل استک

PUSH AX



بعد از این عملیات SP من ۲ من سود
 $SP = SP - 2$

که بردازندگی ۸۰۸۶ مقدار ۱۶ و ۳۲ بیت را من تراهم داخل push, stack کنیم و معکوس آسی
را من توانی push کنیم (زیرا عمل push دو خانه پیازدارد و SP که اینها را ۲ من کند)

POP AX $\rightarrow SP = SP + 2$

Overflow درین Stack Overflow است اگر بینهایت عملیات PUSH انجام دهیم
Underflow درین Stack Underflow وقتی که stack خالی است اگر بینهایت عملیات POP انجام دهیم
این من دهن

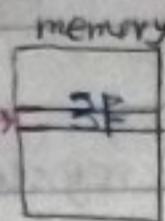
MOV AL, BL

آدرس دهنده باشد

= Register

MOV AL, [441]

آدرس (ادس)



آدرس دهنده

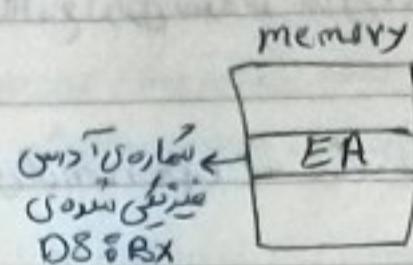
AL = 3F

آدرس دهنده رجیستر غیر مستقیم:

MOV AL, [BX]

آدرس دهنده

AL = EA



= DS: BX + 10

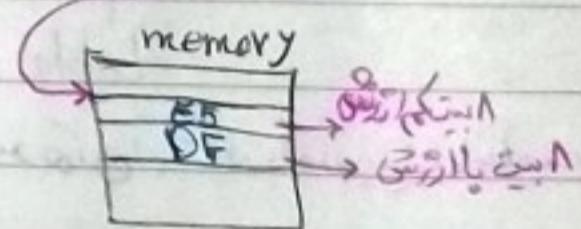
MOV CX, [BX] + 10

offset

آدرس منطبق

CX = PFE5

آدرس داخل حافظه است

آدرس دهنده از رجیسترها DI و SI استفاده می کند
آدرس دهنده از تابع BP و BX و Base Indexed استفاده می کندMOV CL, [BX][DI]+8 = DS: BX + DI + 8

offset

segment offset

MOV CL, [BP][SI]+5 → SS: BP + SI + 5

segment offset

Segment رجیستر های	CS	DS	ES	SS
Offset رجیستر های	IP	SI, DI, BX	SI, DI, BX	SP, BP

Overriding default offset register :

MOV DX, SS:[SI]

تغییر حالت بیس فرض سکمت ها:

MOV CX, CS:[BX]+12

پایه های 8086: معنای دارد که 20 پایه آن که یا AD نام دارند سده اند برای آدرس دهنده حافظه ها پاسخ دهنده هستند.

instruction دستور العمل، مجموعه ای که شامل یک opcode و یک عدد از operand هاست، عبارت های زبان اصلیه

۱- دستور العمل های اسماً پنهان ساده هستند

۲- سمعت زبان اسماً پنهان پلاس ترین پاسخ

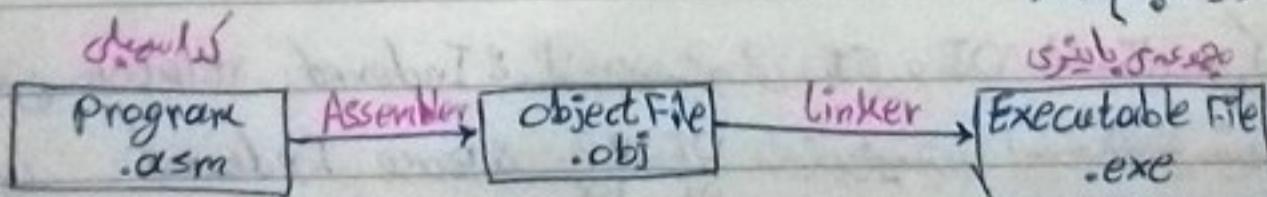
۳- صفت حافظه در دستور العمل های اسماً پنهان پنهان باشد، و نیز مستعینی با حافظه کارهای کمی

۴- در 8086 در گذشت ۶ operand ۲ عدد صریح پاسخ

حالی دستور العمل های اسماً پنهان

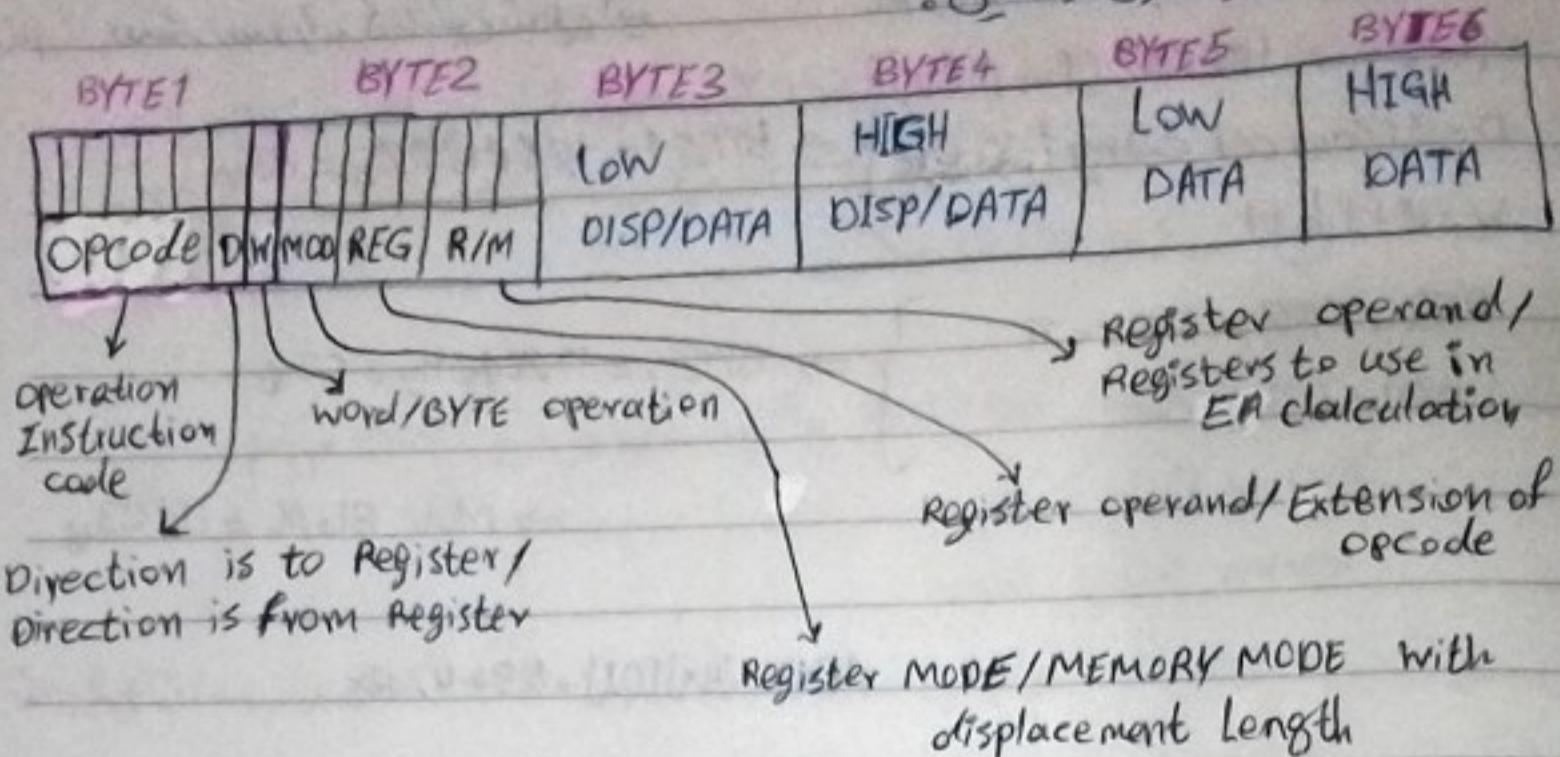
label : opcode op1, op2

نام: زمانی که کامپایلر کد زبان سطح پایه ها را به اسماً پنهان تبدیل می کند همان است که با مرتبه ناپیوسته حافظه کار را انجام دهد



نکته: برنامه قابل اجرا من توانسته فایل های پافروخت، com، exe و bin تبدیل شوند

طالب عومن دستور حاسوبی که می‌خواهیم:



000 دستور AX دستور خود درین طرزی را نیز می‌دانیم، آن را $\text{REG} \oplus \text{D}$ می‌نامیم

ADD AX, BX دستور العمل ها لکه را داشت با نیز $\text{D} \oplus \text{REG}$

$D \begin{cases} 0 & \text{source} \\ 1 & \text{destination} \end{cases}$ دستور العمل ها لکه را داشت با نیز $\text{REG} \oplus \text{D}$ داشت

آخر ۱ دستور العمل ها لکه را داشت با نیز $\text{REG} \oplus \text{D} = 1$

آخر ۰ دستور العمل ها لکه را داشت با نیز $\text{REG} \oplus \text{D} = 0$

طول دستور العمل را مستخرج می‌کنند $W = 1$ معادل ۱۶ بیت

معادل ۸ بیت $W = 0$

REG	$W=0$	$W=1$
000	AL	AX
001	CL	CX
010	BL	BX

MOD

00

MOV AL, [SI]

01

MOV AL, [SI]+8

10

MOV AL, [SI]+2352

11

MOV AL, BL

باتوجه به مقدار D از مقدار REG ویک دستور می‌گیرد

که در ۳ بایت که آنرا دارد سه قطعه دستور دارد

MOV BL, AL

مثال دستور روما كماسی تبدیل کنید

$$\left. \begin{array}{l} \text{opcode} = 100010 \text{ (for MOV)} \\ D = \emptyset \text{ (source) } \text{کلمه منبع} \\ W = 0 \text{ (8-bit)} \end{array} \right\} \Rightarrow \text{BYTE 1} = 10001000_2 = 88_{16}$$

$$\left. \begin{array}{l} \text{REG} = 000 \quad \text{MOD} = 11 \\ \text{R/M} = 011 \end{array} \right\} \Rightarrow \text{BYTE 2} = 11000011_2 = C3_{16}$$

$$\Rightarrow \text{MOV BL, AL} = 88C3_{16}$$

ADD [BX][DI] + 1234H, AX

8.2 جلسه

$$\left. \begin{array}{l} \text{opcode} = 000000 \text{ (for ADD)} \\ D = \emptyset \text{ (source) } \text{کلمه منبع} \\ W = 1 \text{ (16 Bit)} \end{array} \right\} \Rightarrow \text{BYTE 1} = 00000001_2 = 01_{16}$$

$$\left. \begin{array}{l} \text{REG} = 000 \\ \text{MOD} = 10 \\ \text{R/M} = 001 \end{array} \right\} \Rightarrow \text{BYTE 2} = 10000001_2 = 81_{16}$$

$$\text{BYTE 3} = 34_{16} \quad \text{BYTE 4} = 12_{16}$$

$$\text{دسترسی} = 01813412_{16}$$

MOV WORD PTR [BP][DI] + 1234H, 0ADCDH

8.2 جلسه

این دستور عالی عومن را در پیل می کند

MOV 1100011W and W=1 for word-size data

$$\text{BYTE 1} = 1100011_2 = C7_{16}$$

$$\text{BYTE 2} = (\text{MOD}) = 000 \quad (\text{R/M}) = 000 \Rightarrow 10000011_2 = 83_{16}$$

$$\text{BYTE 3} = 34_{16} \quad \text{BYTE 4} = 12_{16} \quad \text{BYTE 5} = CD_{16} \quad \text{BYTE 6} = AB_{16}$$

$$\text{دسترسی} = C7833412CDAB_{16}$$

دستور العمل 8086

دستور العمل 1- انتقال داده لامثلجات ذي صفاتي كـ CS لـ DS

دستور العمل يعنى داخل ثبات IP و CS قرار من غير عصمت آن داخل IP عصمت Segment آن داخل CS قرار من غير

دستور العمل سى mov بلائحة دستور العمل mov من كـ CS

انتقال MOVSB
word mov

XCHG ؛ جابجي دو علود رعيت آن (Swap کوت خلاي هاي آن)
IN IN دستور العمل

IN AL, im.byte
AX, im.byte
AL, DX
AX > DX

Input from port into AX

Example IN AL, OC8H

IN AX, 34H

MOV DX, OFF78H

IN AL, DX

IN AX, DX

OUT

OUT im.byte, AL
im.byte, AX
DX, AL
DX, AX

first operand is a port number

Example

OUT 3BH, AL
OUT 2CH, AX

PUSH REG | SREG | memory | immediate
store 16 bit value in the stack

• PUSH دستور

Algorithm:

$$SP = SP - 2$$

SS : [SP] (top of the stack) = operand

دستور که از SP کم کند و BX را در آن ذخیره می کند

PUSH BX

" " DS "

PUSH DS

دستور که از SP کم کند و بیت ورد را در آن ذخیره می کند

PUSH TABLE[BX]

EA = TABLE + [BX]

• دستوری که EU را در یک رजیستر واحد پسندیده باشد : PUSH A

• PUSHF دستور

→ no operands

store flags register in the stack

Algorithm:

$$SP = SP - 2$$

SS : [SP] (top of the stack)

POP REG | SREG | memory

• POP دستور

جیسرا که می بینیم

Get 16 bit value from the stack

Algorithm:

operand = SS : [SP] (top of the stack)

$$SP = SP + 2$$

Example pop DX

→ no operand

pop all general purpose registers

⇒ POPA جملة

→ no operands

Get flags registers from the stack

Algorithm:

flags = \$5 ; [SP]

SP = SP + 2

⇒ POPF جملة

تمسّك علويّ راسٍ / بمحليّ مسْتَوى مُورِّيّ

⇒ LEA دستور المُفْرِج
Data Segment

LDS BX,[4326]

Data Segment Job

Acts like LDS

Load ⇒ LES دستور المُفْرِج
Extra Segment

ADD اضافة

INC اضافة

ADC اضافه مع التكميل

AAA تخلص الملفوفات

DDA

دستور المُفْرِج بحسب المعايير

ASCII Adiast After Addition

16 bit بحسب معنی WORD

AAA & Corrects result in AH and AL after addition

when working with BCD values

Example: MOV AX, 15 ; AH=00, AL=0Fh

AAA

, AH=01, AL=05

RET

DAA : Decimal Adjust After Addition

Corrects the result of addition of two packed BCD values.

Example:

MOV AL, OFh ; AL = OFh (15)

DAA ; AL = 15h

return ← RET

بی پایه دو را پس از جمع محتوای درست کنید.

(DX,CX) ← (DX,CX) + (BX,AX)

وام حمله

(DX,CX) = FEDCBA98₁₆

(BX,AX) = 01234567₁₆

MOV DX, OFE0CH

MOV CX, 0B98H

MOV BX, 01234H

MOV AX, 04567H

ADD CX, AX

ADC DX, BX;

دستور العمل های تصریف

SUB, SBB

NEG, DEC, AAS, DAS

sub کم کردن

Sub ds

sbb پاکی تغیری کردن

Sbb ds

dec کم کردن

neg منفی کردن

das تغیل اسکرین

aas تغیل دهی ای پری تغییری

نتیجہ اجرائی دستور العمل
SBB BX, CX
در صورتی کہ محتوای رجیستری CX, BX پر ترتیب 0123₁₆, 1234₁₆ ہے تو
- مقدار صفر اسے CARRY FLAG رکھ دیں۔

$$(BX) - (CX) - (CF) \rightarrow (BX)$$

$$(BX) = 1234_{16} - 0123_{16} - 0_{16} = 1111_{16}$$

صفر با من ملئو۔ Carry Flag

مثال: دن لفڑا بگیرید کہ رجیستر BX دلائی معتدل 003A₁₆ میں باشد۔ نتیجہ اجرائی دستور العمل
NEG BX

زیر حفیضت P

$$(BX) = 0000_{16} - (BX) = 0000_{16} + 2^{\text{complement of }} 003A_{16}$$

$$= 0000_{16} + FFC6_{16} = FFC6_{16}$$

یوں ہیج دستور العمل کو ایجاد نہیں سو دیں مقدار متمم آن
یعنی 1 من مندرجہ

دستور العمل DAS محتوا کیت رجیستر را کہ تفریق انجام دادا ہے دسحال تبدیل میں کند
درسترد-سازنے (DAS)

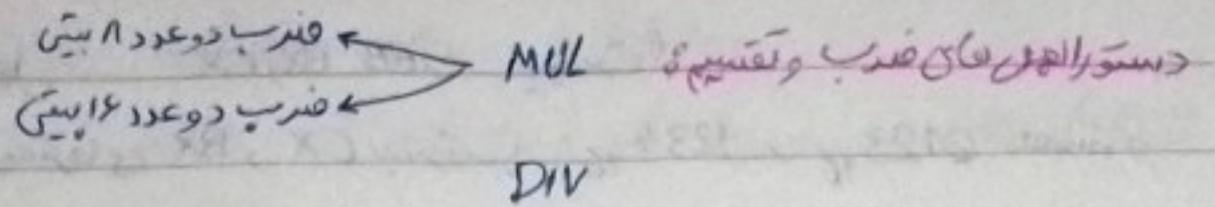
MOV AL OFFh ; // $AL = OFFh \uparrow (-1)$

DAS ; // $AL = 99$ (سیسمونی) CF = 1

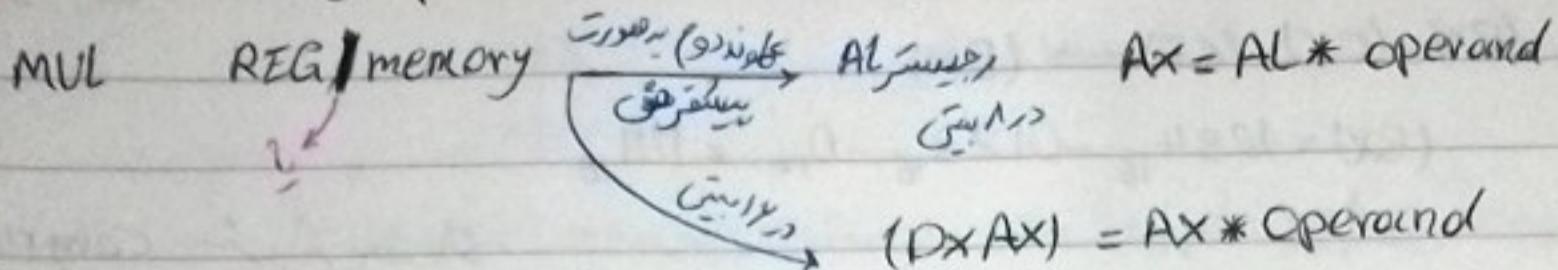
RET ;

$$\begin{array}{r} 1111 \\ - 0001 \\ \hline 1110 + 1 = 1111 \end{array} \leftarrow \begin{array}{l} \text{تینی عدد} \\ \text{کو 1 کیلئے} \\ \text{کو 2 کیلئے} \end{array} \leftarrow r=2$$

$$\begin{array}{r} 99 \\ - 01 \\ \hline 98 + 1 = 99 \end{array} \leftarrow \begin{array}{l} \text{کو 1 کیلئے} \\ \text{کو 2 کیلئے} \end{array} \leftarrow r=10$$



هان قيمات ملائم دار باستكمال



جواب MOV AL, 200

MOV BL, 4

MUL BL

RET

خارج

مسوّر الظلّي تقسيم

for byte $\Rightarrow AL = AX / \text{operand}$

AH = remain

JBC: DIV BL

for word $\Rightarrow AX = (DX \times AX) / \text{operand}$

DX = remain

مسوّر الظلّي \leftarrow تبديل عدد سهله باینری (تنظیم AX برای ضرب)

$$AL = (AH \times 10) + AL \quad \text{و} \quad AH = 0$$

MOV AX, 0105h ; AH=01, AL=05

AAD ; AH=00, AL=0Fh(15)

RET

Convert Byte to Word CBW

Convert word to double CWD

کسٹریس عالمت (پیپر)

پاس کرنے والے بائینری پاس کرنے والے

AND D, S

بیکھای 0 دی کر رہا ہم
پس کرنے والے AND

X	Y	$X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

دستور العمل کا منطقی
 AND ←
 OR ←
 XOR ←
 NOT ←

$$\begin{array}{r} 01010101 \\ \text{AND } 00011111 \\ \hline 00010101 \end{array}$$

AND AX, 000F₁₆

حصیر کردن ہایک کردن سیستم مادر ریسٹریو

منطقی ←

SAL, SHR, SAL, SAR

دستور العمل کا منطقی

Shift Logical Right

1100	input
0110	result

Shift " " Left

1000

Shift Arith Right

1110

Shift " " Left

1000

SAR => memory, immediate | REG, immediate | memory, cl | REG, cl

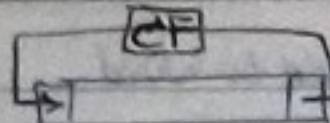
JMP SAR [SI], 4

SAR AX, 4

SAR [DI], cl

SAR AL, cl

منطقی
CF پر تاثیر نہیں ملے جائیں، بلکہ CF پر تاثیر ملے جائیں



دستورالعمل های پر خسی

ROL سه سهت بیت \Rightarrow چرخش پهلوی
نه راست ترین بیت رو داد

ROR

ROL ~~B~~ D, COUNT

RCL چرخش پاوا سعلی کری

\downarrow مقصد تعدادی های پارامتری چرخش

RCR

نکته: در دستورالعمل های پر خسی اگر بیت چرخش را فتح بدهی CF نشود از دو کردن
نه: در چرخش های پوشاکی کری محتوایات قبلی کری استعاری صیغه
دستورالعمل های لفڑی Flag

AHF \leftarrow هیک پایین قلک را داخل AH بودی کن

SAHF \leftarrow AH پایین دستهی \downarrow خود را خارج کن

STI وقفه راییک کن

CMC کری را تمم کن

CLC نه راییک کن

CLI وقفه راییک کن

STC کری راییک کن

carry Flag نه %CF

برای تعیین وصفروید کردن
از 5 Flag ها استفاده می کنند

CMP مقایسه

برای حلک های رو به رو ایجاد کناره OF

یک عملیات تغیریق انجام گیرد

اگر 1 مسدود
لطفی خود را
مسادی اند
 \uparrow

Condition code After CMP

unsigned operands
equality / inequality

signed operands
equality / inequality

C $OP1 < OP2 (C=1)$ $OP1 \geq OP2 (C=0)$

no meaning

S no meaning

no meaning

O no "

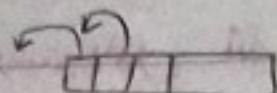
"

⑨

Subject:

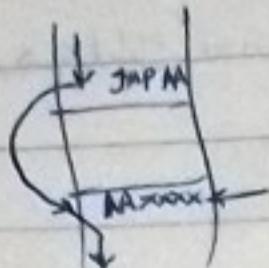
Year _____ Month _____ Day _____

مكتبة مراجعة سخيفه overflow کوچک دویت آفرا XOR کتنی

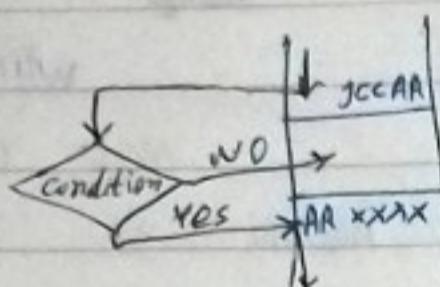


$$OF = C_n \oplus C_{n-1}$$

Call Label $\hookrightarrow AA$



مسند العمل جسته رفع و حذف کنیه JMP



مسند العمل برس سطحی

JMP operand

hiFlag رفع و حذف

LABEL
نردنی
بر
حافض
ریسٹر
حافظه

LABEL
بر
حافض
ریسٹر
حافظه

IF THEN ساختار برعایتی

CMP AX, BX

JE EQUAL

} if $AX \neq BX$

EQUAL :-

} if $AX = BX$

CMP AX, BX

$C=1 \quad AX < BX$

$C=0 \quad AX \geq BX$

$AX > BX \quad C=0, Z=0$

$AX = BX \quad Z=1$

$AX \neq BX$

$S \oplus 0$

overflow

$1 \quad AX \neq BX$

IRAN EMRUGZ

I, BX, AX کے اعداد پر عمل کا سندھر کی جو Jump Address JA ہے جس سے دستور اچھا ہے۔ $AX > BX \leftarrow$

Set میں سورج All instructions کو درج کرنے والے status Flag / جیسے کہ

MOV CL, COUNT

حلہ کا

AGAIN :

```

---+
---+ |
---+ |
DEC CL
JNZ AGAIN
---+

```

~~loop~~

WHIL-DO

MOV CL, COUNT

AGAIN : JZ NEXT

Loop is complete if

$CL = 00H (ZF=1)$

? ?

DEC CL

JMP AGAIN

NEXT : --- --

~~loop~~ JNZ : Jump Not zero

Loop :

MOV CX, COUNT

: Loop کا جائزہ

jNZ NEXT :

if $CX = 0 \Rightarrow$ End Loop

Loop[↑] zero

LOOPNEXT

LoopE/LoopZ :

checks $CX \neq 0$ and $ZF = 1 \} \Rightarrow$ End when this condition

LoopNE/LoopNZ :

checks $CX \neq 0$ and $ZF \neq 0 \}$

wrong

CALL → فرایند است

RET → بازگشت

CALL operand

Near-proc

Far-proc

Memptr16

Regptr16

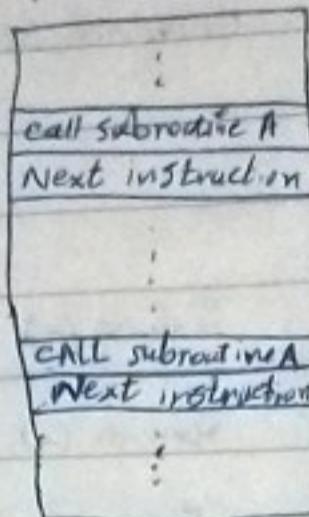
Memptr32

دال سکیت

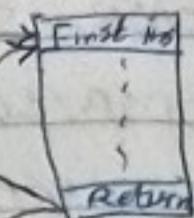
بررسی

Subroutine دستور الفرعی

main program



Subroutine



→ در قسم اجرای CALL آدرس دستور الفرع پس از CALL در ذخیره کرد

RET operand → no operand

immediate Return

RET Far → بازی اسک دو مقدار SP و CS را قبل از دستور زدن ذخیره کرده بعده را من عواید آنجامی روید

SQUARE PROC NEAR

PUSH AX

MOV AL,BL

→ POP و LPUSH تعداد

IMUL BL

پایین برآورده شود

MOV BX,AX

→ هادر است که POP، PROC های داشته باشد

POP AX

بجا می دعیم PROC

RET

SQUARE ENDP

TRAN EMROOZ

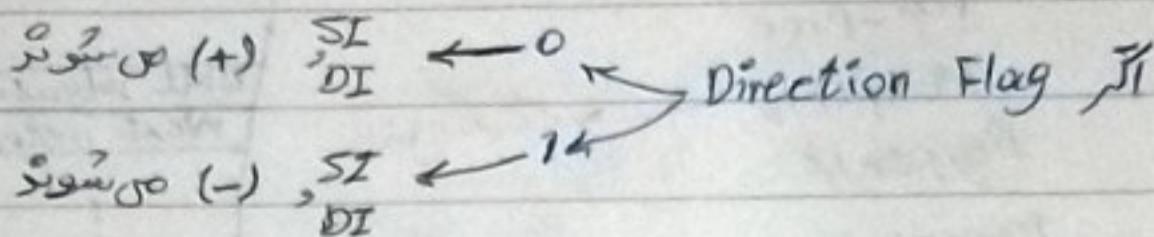
MOVSB / MOVSW Move String Byte : String (متن) کو منتقل کرنا

" " Word Compare String Byte

SCASB / SCASW

LODSB / LODSW

STOSB / STOSW



CLD clear Direction Flag

Prefix used with

REP MOVS repeat while not end of string CX ≠ 0

STOS

REPE/REPZ CMPS repeat while not end of string
SCAS and strings are equal
CX ≠ 0 and ZF = 1

REPNE/REPNZ

CX ≠ 0 and ZF = 0

JG²⁰ MOV Ax, 0

MOV DS, AX

MOV ES, AX

MOV AL, 05

MOV DI, —

MOV CX, OFH

CLD

REP STOSB

keybord:

CLD

clear DF

STD

Set DF

%include "io.inc"

section.data

سلکت دیتا

msg db 'Hello, world!', 0

داده های پرینتر قابل String

section.text

سکت کد

global CMAIN

کدامی پر نام

CMAIN:

;; test for comment

ret

برچیدن در msg

jump location مسخن کردن

: = to LABEL

برچیدن در CMAIN:

variable مسخن کردن آدرس

Local ← = to LABEL انواع
global ←

equ : symbol equ value

تبلیغ ها در برابر const

: = Asm directive ادوات

%define : %define SIZE 100

Data Directives

RESX

داده های که مقدار اولیه ندارند

داده های که مقدار اولیه دارند

DX

	Unit	Letter (X)
Byte	B	بایت
Word	W	دیجیت
doubleword	D	دو بایت
quad word	Q	
ten bytes	T	

آدرس این سطیح است

db

L1 db 110101b

L2 resw 100 // صد عدد Word تعریف شوند
که 10 آدرس اولین Word است

IRAN EMROOZ

L3 db "W", "O", "R", "D", 0

null که اسکرپت null

آخر رشته با لغزیدن پایانگار

استفاده از LABIE :
 mov al, [L1] متغیر L1 را بخواهد
 mov bl, L2 L2 را در خواهد

mov dw word [L6], 1

ذنوب عدد 1 در دوبلیک

کات در پردازنده 80386 پس از رجیسترها 32 بیتی شدند و رجیسترهای EBX، EAX ... تغییر ناواردند
و RBX، RAX در پس از pentium 64 بیتی شدند و با هم آن رجیسترها از پردازنده از پردازنده

string1 db 'assembly program'

Length dw \$-String1
 ↓
 آدرس دستور العمل
 حایی ↓
 آدرس ↓
 دستور العمل
 String1

در این جا دستور length بعده دستور
تمدید آن آدرس آن بعده
است و باقیتای آن ها آدرس
نهان رفته بسته من آندر

imul dest, source

imul dest, source1, source2

اس اسم لحیل myinput1 نوع متغیر مقدار 1000H, 0200H
 ↓ ↓ ↓
 دبل = dd سهتی باش سهتی باش

? imul دستور

dd = define double

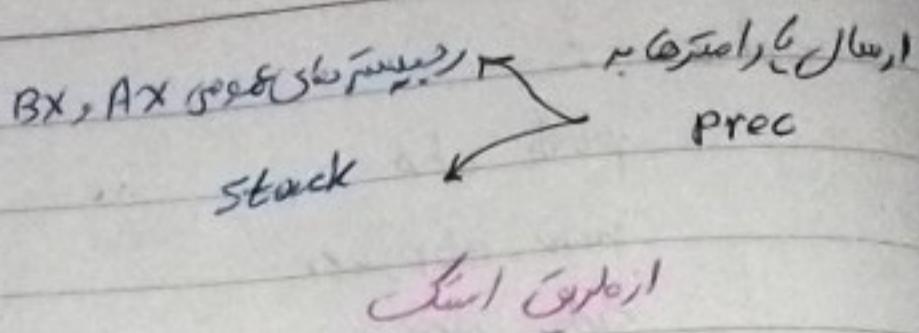
Sub Routine :

call proc-name

proc-name :

ret

فرض اسلامیه استاگر خوشه
push a
از همین ریست دای که می خواهد



mov eax,[a]

mov ebx,[b]

call add

; add:

mov ecx,eax

add eax,ebx

ret

عدد دیگر دارد سپس تعداد ریست
Counter

push a

push b

call add

add esp,8

addl:

mov eax,[esp+4];

add eax,[esp+8];

ret B

دیگر این دستور

در این صورت عدد روی بلوں جمع میں مسود

اسالید ۲۷ سوچ دیوند با آخر دستور Sub اسیہ نوٹھے سے درست اس

Local Variables in sub programs

برای اسناد از روئین میں تقدیر باشد از استاگر برای تحقیق دانست متفہ اسناد کیج

Subprogram-table :

push ebp

mov ebp,esp

sub esp, LOCAL_BYTES

تعداد بیت می

متغیر کوں کوں

& subprogram code

mov esp,ebp

pop ebp

IRAN EMROOZ ret

cal_sum :

عمل

push ebp

mov ebp, esp

sub esp, 4

ماکروها (Macro) همان توابع هستند که کدها را بصیرا، فارزدن کنند.

%macro name premcnt
تعداد پارامترها (بروچ)

! %1 → پارامتر اول

%2 %2 → پارامتر دوم

%endmacro

اسلایر int 80h ←
خدمت از سیسیم عامل برای جابجایی interrupt ←

ترکیب کدها و Assembly

۱- بوسیت C++, ASM ←

Visual Studio SASM

۲- پیروکوت بوسیت NASM داخل فدر

۳- غایبASM، را کثرا فاعل می باشند NASM قادری دارند

۴- کد

ترمیم فایل‌های پارامترها و عرضه آنها در زبان‌های مثل C و از راست پڑید است

pascal // اوچیپ پر راست

گذشت نازهانی کدی تابع متغیر باشد ... وارد جای public تعریف نکرده باشیم نمی توانیم در طایی ریگر Extern تعریف کنیم

گذشت متغیری که در داخل یک پروسس extern تعریف شده است داخل اسمبلی باشد پاکیزه -array مجموع شود (underlined (-))

دستور تبریز خوبی داریں ملکیت داخل اسکے پوسٹ میں گرد رہے

`printf("sum=%d", int-var);`

از راستہ چہہ
پوسٹ میں کند
اویس پارسند
داخل اسکے
پوسٹ میں سود
دو صین را پھر راستہ
داخل اسکے پوسٹ میں سود

۲۳،۴۵۷ → ۲۳،۴۶

تعلیم کردن (Trim)

۲۳،۴۵۷ → ۳۳،۴۶

گرد کردن

۰۰ میٹر سٹاوارہ

تعریف یک عدد میٹر سٹاوارہ

لے لیں ۱۰ سال

تو ان عمدہ بخیر نہیں۔

مرکی جمیع دو عدایاں شان دعا رہیں گئیں۔

در عدد دوم ~~۲~~ اضافت سودہ لذائیں بیت سیفت میں یاد
و ~~۱~~ چون یک ۱ میرون میں افتخار لذائعد کردن میں گردد