دوره مهندسی معکوس نرم افزار

- Site: OnHexGroup.ir
- Youtube: @onhexgroup
- Telegram: onhex_ir
- Twitter: @onhexgroup
- Github: onhexgroup

نسخه ی ۳۲ و ۶۴ بیتی

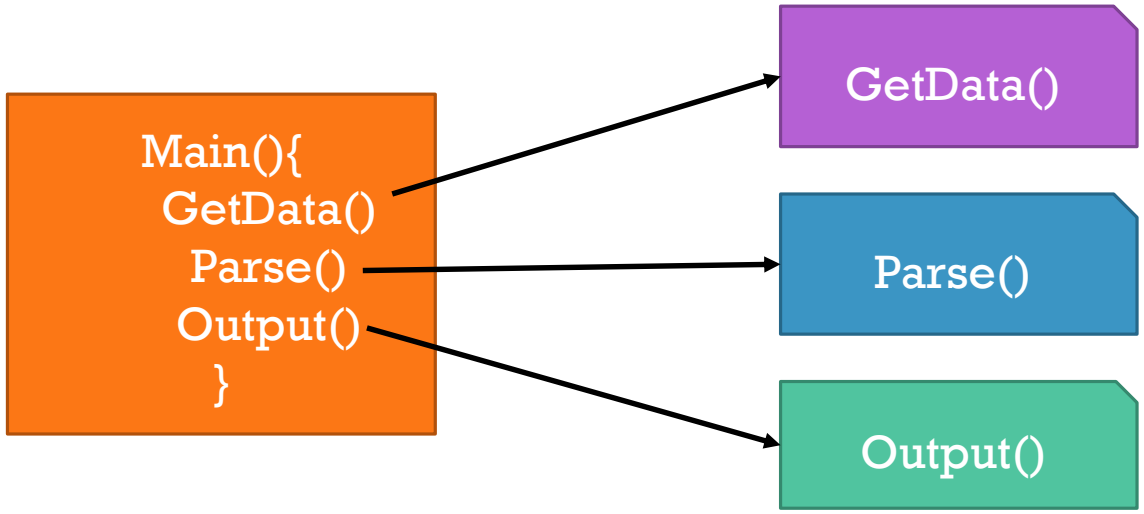
پلتفرم: ویندوز

ارائه دهنده : **OnhexGroup**

Onhexgroup.ir

توابع

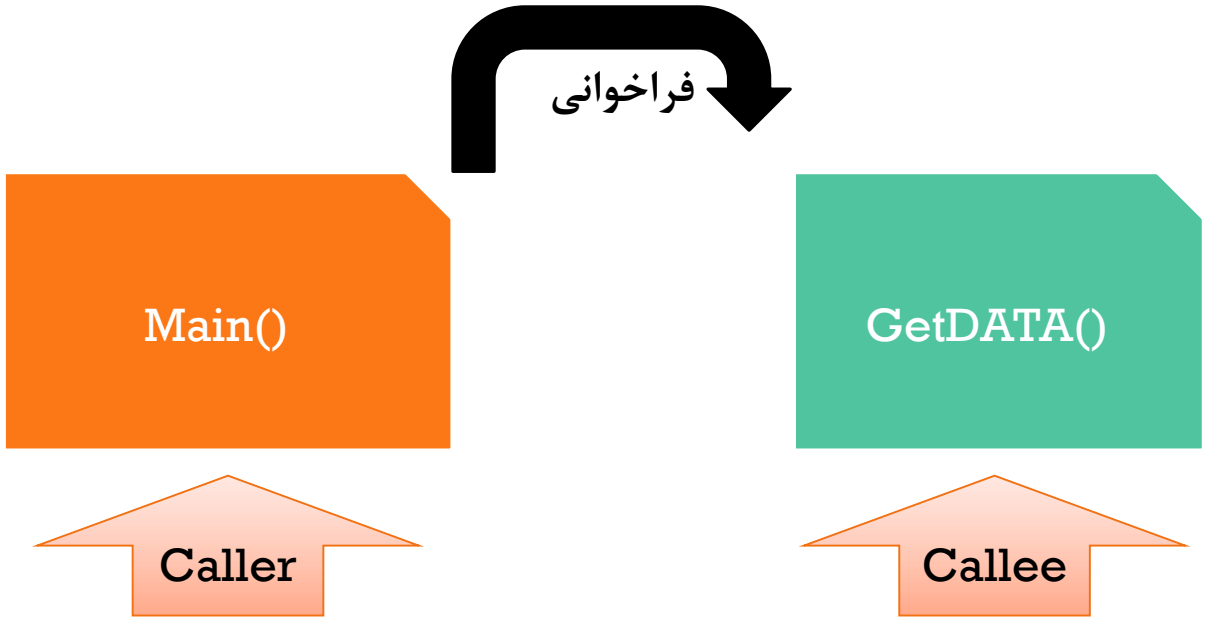
- به کمک توابع میتونیم برنامه های بزرگ رو به قسمت های کوچیک تقسیم کنیم:
  - توسعه برنامه ساده تر میشه
  - نگهداری و عیب یابی برنامه ساده تر میشه.
- تابع، روال ، رویه، متد، زیربرنامه، **Subroutine** ، **Procedure**



Twitter:Onhexgroup

کار با توابع

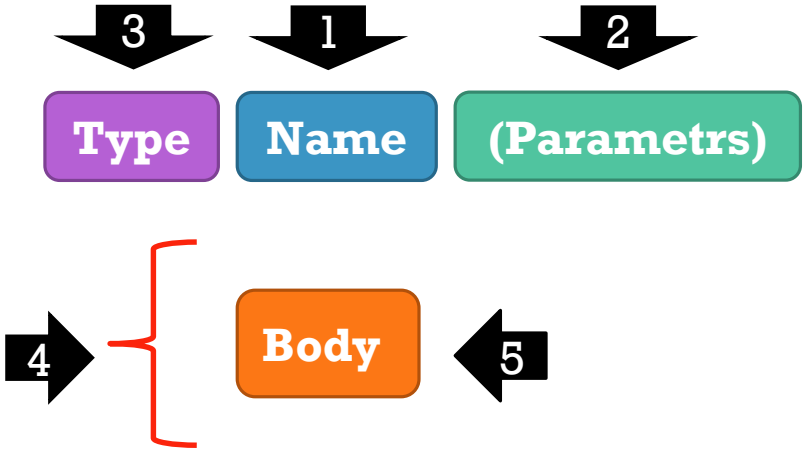
- برای کار با توابع:
  - نحوه ی تعریف یک تابع
  - نحوه ی فراخوانی یک تابع



Twitter:Onhexgroup

موارد تشکیل دهنده ی تابع

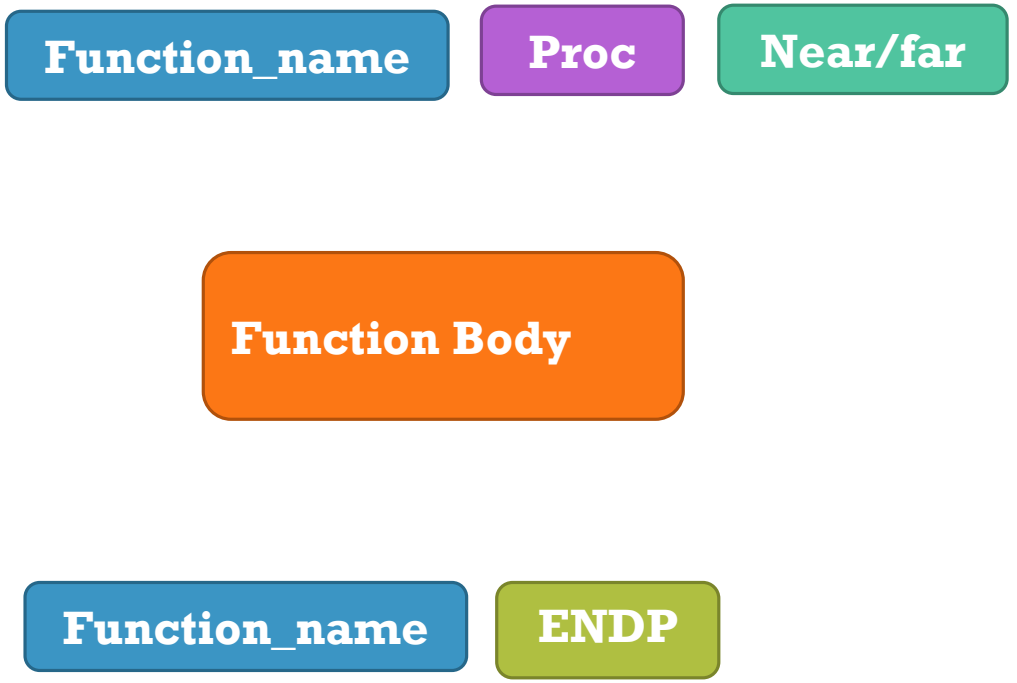
- یک تابع میتونه از موارد زیر تشکیل بشه:
  1. نام
  2. پارامترهای ورودی
  3. نوع
  4. محدوده
  5. بدنه



- در اسمبلی برای تعریف توابع از ساختار زیر استفاده میکنیم:

Github: Onhexgroup

تعریف توابع



- در C/C++ برای تعریف توابع از ساختار زیر استفاده میکنیم:
(Inline assembly 32 bit)

Github: Onhexgroup

تعریف توابع

Onhexgroup.ir

فراخوانی توابع

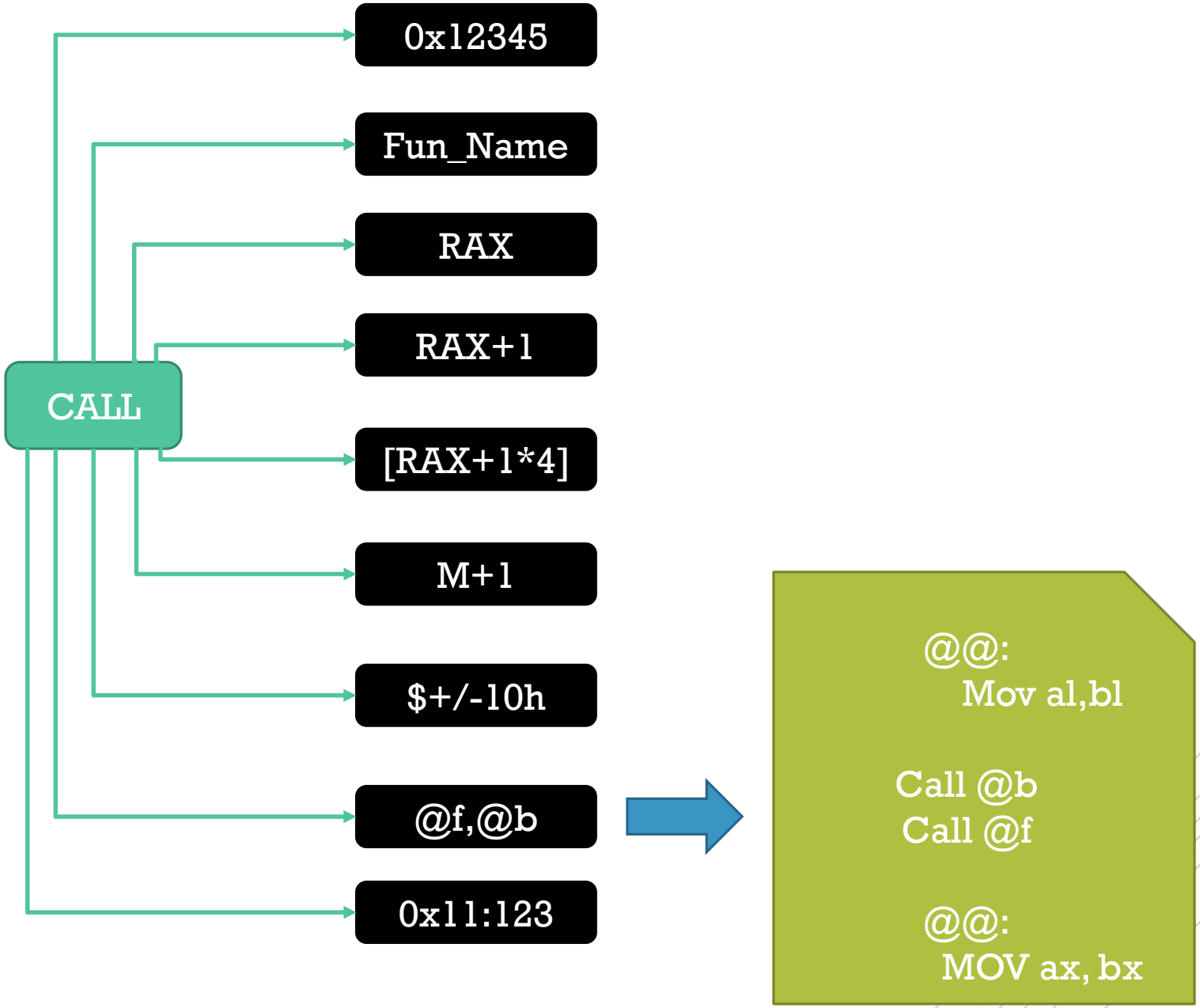
- برای فراخوانی توابع از دستور **Call** استفاده میکنیم.
- عملکرد این دستور:
  - آدرس بازگشت را در پشته قرار میدهد
  - اجرای برنامه رو به تابع فراخوانی شده، میبره. (سر تابع)

CALL R/M/IMM

- انواع دستور **CALL**:
  - **NEAR** یا نزدیک : در همان سگمنت کد
  - **FAR** یا دور: در یک سگمنت دیگه
  - **Inter-privilege-level far** : همان نوع **FAR** فقط سطح دسترسی فرق داره.
  - **Task switch**: در یک تسک دیگه (در ۶۴ بیتی پشتیبانی نمیشه)

Onhexgroup.ir

فراخوانی توابع



Onhexgroup.ir

بازگشت از توابع

- برای بازگشت از توابع از دستور **RET** استفاده میکنیم.
- عملکرد این دستور:
  - آدرس بازگشت رو از پشته در **EIP/RIP** قرار میده
  - اجرای برنامه رو به آدرس مورد نظر، میبره. (آدرس بازگشت)

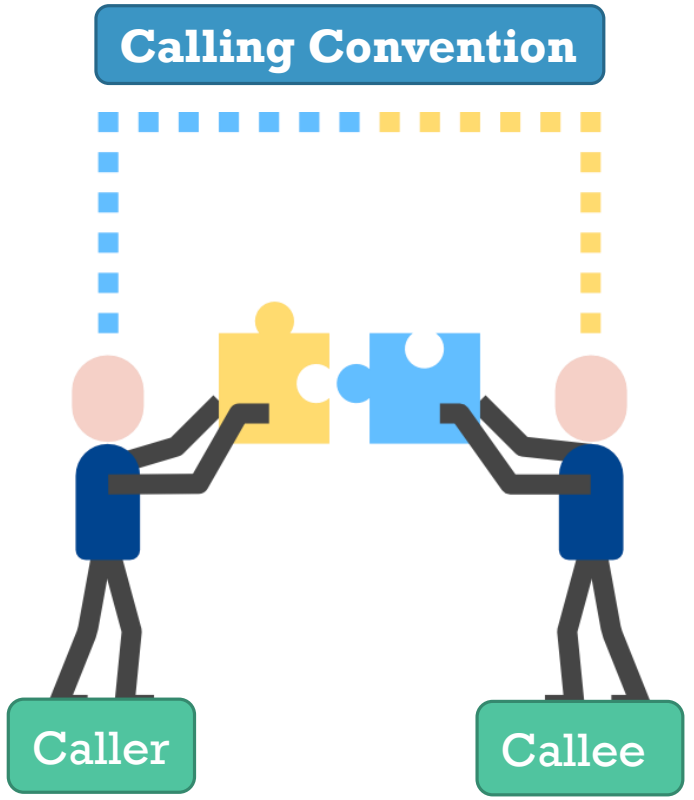
RET IMM16

- انواع دستور **RET**:
  - **NEAR** یا نزدیک : در همان سگمنت کد
  - **FAR** یا دور: در یک سگمنت دیگه
  - **Inter-privilege-level far** : همان نوع **FAR** فقط سطح دسترسی فرق داره.

Onhexgroup.ir

قرارداد فراخوانی

- قرارداد فراخوانی یا **Calling Convention** یک قرار داد بین **Caller** و **Calee** هستش که موارد زیر رو مشخص میکنه:
  - نحوه ی ارسال آرگومانها
  - نحوه ی بازگردانی نتایج
  - مشخص کردن مسئول پاکسازی و مدیریت پشته



Onhexgroup.ir

قراردادهای فراخوانی در ویندوز

- قراردادهای فراخوانی در نسخه ی ۳۲ بیتی:
  - **Cdecl**
  - **Stdcall**
  - **Fastcall**
  - **Thiscall**
- قراردادهای فراخوانی در نسخه ی ۶۴ بیتی:
  - **Windows x64 Calling Convention**

Onhexgroup.ir

قرارداد فراخوانی در ۳۲ بیتی

- نوع **Cdecl (C Declaration)**:
  - ترتیب ارسال پارامترها: از راست به چپ
  - ارسال پارامترها : از طریق پشته.
    - سایز: براساس دستور **PUSH** – معمولا **Dword**
  - بازگرداندن نتیجه: از طریق **EAX**
    - سایز بسته به نوع دارد. بایت =**AL** و ...
  - پاکسازی پشته: بر عهده **Caller**
  - فراخوانی پیش‌فرض در زبان‌های **C**و **C++**
  - در توابع **Varargs (Variadic)** استفاده میشه:

```
Add(int a , int b);


Printf(“onhexgroup”);
Printf(“Result of %d+%d= %d”,a,b,c)
```

Onhexgroup.ir

قرارداد فراخوانی در ۳۲ بیتی

- نوع **Stdcall (Standard Call)**:
  - ترتیب ارسال پارامترها : از راست به چپ
  - ارسال پارامترها : از طریق پشته.
    - سایز: براساس دستور **PUSH** – معمولا **Dword**
  - بازگرداندن نتیجه: : از طریق **EAX**
    - سایز بسته به نوع دارد. بایت =**AL** و ...
  - پاکسازی پشته: بر عهده **Callee**
  - این توافق به طور گسترده در **Windows API** استفاده میشه و به همین دلیل معمولاً به عنوان **WinAPI calling convention** هم شناخته میشه.
  - چون تابع خودش مسئول پاک کردن پشته هستش، بنابراین فراخوانی توابع ساده هستش.

Onhexgroup.ir

قرارداد فراخوانی در ۳۲ بیتی

- نوع **Fastcall**:
  - ترتیب ارسال آرگومانها : از راست به چپ
  - ارسال پارامترها: دو پارامتر اول در **ECX** و **EDX**، و بقیه از طریق پشته.
    - سایز: بسته به نوع داره + دستور **PUSH**
  - بازگرداندن نتیجه: : از طریق **EAX**
    - سایز بسته به نوع دارد. بایت =**AL** و **...**
  - پاکسازی پشته: بر عهده **Callee**
  - برای بهبود عملکرد و زمانیکه نیاز به سرعت بالاست استفاده میشه.

Onhexgroup.ir

قرارداد فراخوانی در ۳۲ بیتی

- نوع **thiscall**:
  - برای متدهای کلاسها استفاده میشه.
  - اشاره گر **this** به طور ضمنی در**ecx** قرار میگیره. (سایز **Dword**)

Onhexgroup.ir

قرارداد فراخوانی در ۶۴ بیتی

- نوع **Windows x64 Calling Convention**:
  - **X64 fastcall**
  - ترتیب ارسال پارامترها: از راست به چپ
  - ارسال پارامترها: (سایز بسته به مقدار ارسالی دارد)
    - پارامتر اول: **RCX** پارامتر دوم:**RDX**
    - پارامتر سوم:**R8** پارامتر چهارم:**R9**
    - بقیه: پشته
  - بازگرداندن نتیجه: از طریق **RAX**
    - **Byte=ax – word,dword=eax,qword=rax**
  - پاکسازی پشته: **Caller**
  - با توجه به اینکه از رجیسترها استفاده میکنه، باعث بهبود عملکرد میشه.

Onhexgroup.ir

متغیرهای محلی

- متغیرهای محلی، متغیرهایی هستند که در داخل بدنه ی یک تابع تعریف میشن و تا زمان اجرای تابع، عمر دارن.
- در کدهای زیر متغیر **r**، یک متغیر محلی است.

```
int ADD(int a, int b){
        int r;
        r= a+b;
        return r;
}
```

- متغیرهای محلی داخل پشته ذخیره میشن.

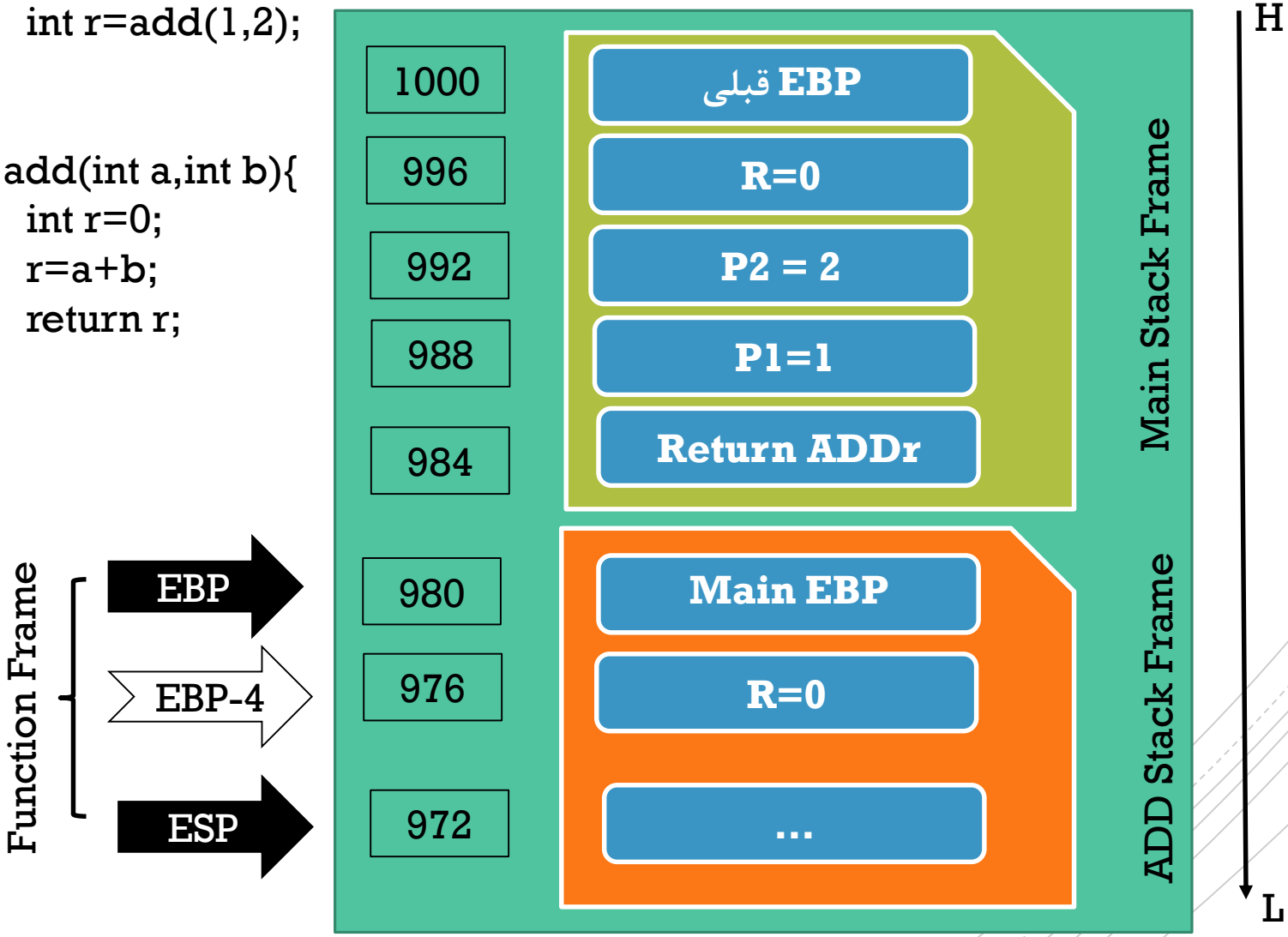
- در ۳۲ بیتی برای دسترسی به متغیرهای محلی از **EBP** استفاده میکنیم.

Onhexgroup.ir

متغیرهای محلی در نسخه ی ۳۲ بیتی

```
Main(){
        int r=add(1,2);
}

Int add(int a,int b){
        int r=0;
        r=a+b;
        return r;
}
```




```
Mov dword ptr [ebp-4],0
```

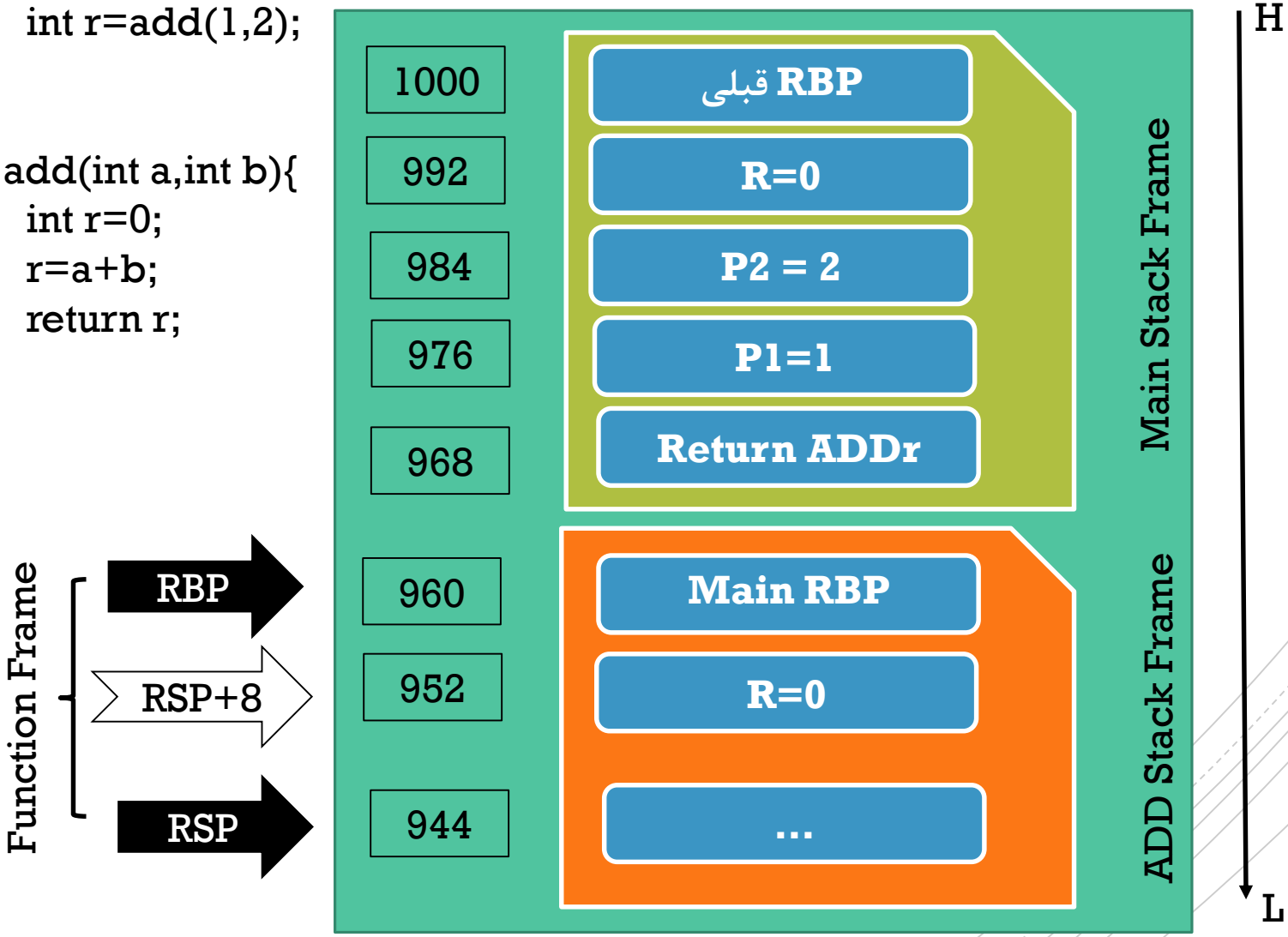
- در 64 بیتی برای دسترسی به متغیرهای محلی از RSP استفاده میکنیم.

Onhexgroup.ir

متغیرهای محلی در نسخه ی 64 بیتی

```
Main(){
        int r=add(1,2);
}

Int add(int a,int b){
        int r=0;
        r=a+b;
        return r;
}
```




```
Mov dword ptr [RSP+8],0
```