AMD64

DANIEL LINS MACIEL

Arquitetura Computacional Avançada

IMED - 2020

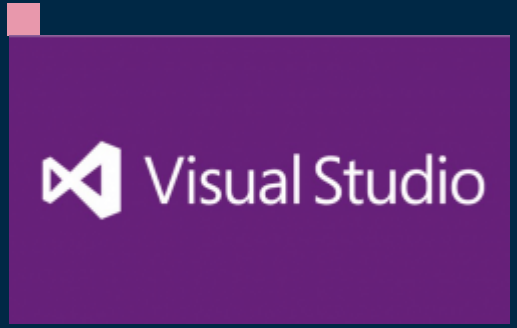
A arquitetura

x86-64

IA-64

Pré Requisitos

IDE: Visual Studio Community



Download: https://visualstudio.microsoft.com/pt-br/vs/community/

Dependências necessárias:

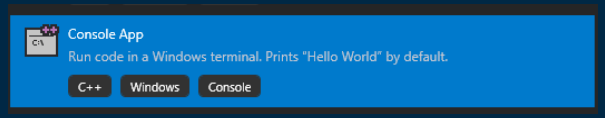
SDK do Windows / C++ Compiler

Extensão ASM Dude:

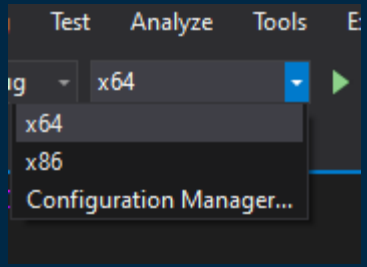
https://marketplace.visualstudio.com/items?itemName=Henk-JanLebbink.AsmDude&ssr=false

Configuração do Ambiente

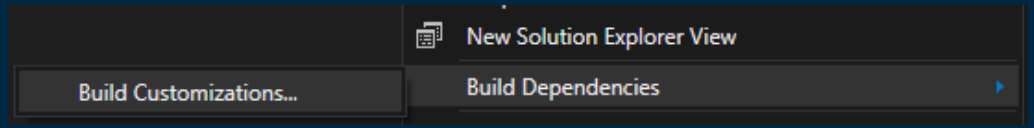
Crie um projeto novo em C++ de Console App

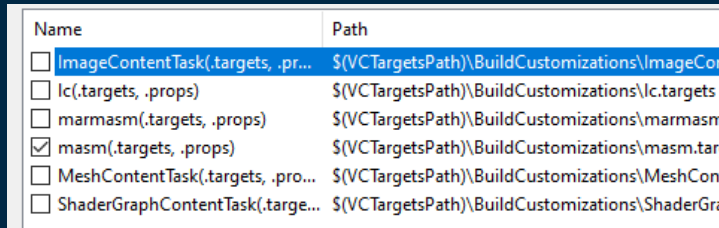


Mude o Debug para x64



Clique com o botão direito do mouse e edite as opções de Build e ative a opção MASM:





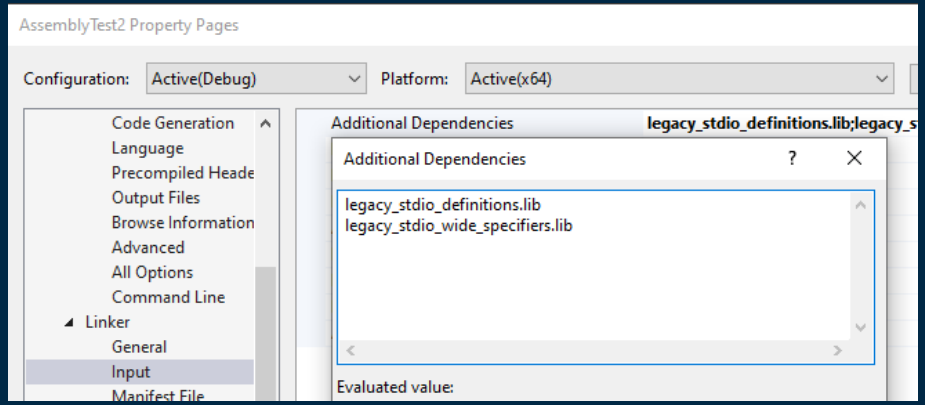
Configuração do Ambiente

Adicione um arquivo no projeto (nos arquivos de origem (source files), com o nome de asm_code.asm
Adicione as dependências no arquivo inicial .cpp do projeto e a chamada para o assembly:

```
#include <iostream>

extern "C"  int funcao();

int main()
{
		//nome da função asm a ser chamada
		funcao();
		return 0;
}
```

Configuração do Ambiente

Clique com o botão direito no Projeto, vá em propriedades e adicione as dependências no linker:



Pronto: Só Programar!

```
hello_world_asm PROC

    push rbp ; save frame pointer
    mov rbp, rsp ; fix stack pointer
    ;sub rsp, 8 * (4 + 2) ; allocate shadow register area + 2 QWORDs for stack alignment
    sub rsp, 32
    ; Get a window handle
    call GetForegroundWindow
    add rsp, 32
    mov rcx, rax ; rax contains window handle

    ; WINUSERAPI int WINAPI MessageBoxA(
    ;  RCX =>  _In_opt_ HWND hWnd,
    ;  RDX =>  _In_opt_ LPCSTR lpText,
    ;  R8  =>  _In_opt_ LPCSTR lpCaption,
    ;  R9  =>  _In_ UINT uType);

    mov rdx, offset hello_msg
    mov r8, offset info_msg
    mov r9, 0 ; MB_OK

    sub rsp, 32
    ;and rsp, not 8 ; align stack to 16 bytes prior to API call
    ;sub rsp, 8 ; mess up the alignment
    call MessageBoxA
    add rsp, 32
    ; epilog. restore stack pointer
    mov rsp, rbp
    pop rbp

    ret
hello_world_asm ENDP
```

Códigos

```
;;;A=B+C
funcao proc
    mov rax, valorA
    mov rbx, valorB
    mov rcx, valorC
    mov rdx, valorD

    ;A = A+B
    add rax, rbx
    ;A = A+C
    add rax, rcx

    ret
funcao endp
```

```
;;;A=B-C+D
funcao2 proc
    mov rax, valorA
    mov rbx, valorB
    mov rcx, valorC
    mov rdx, valorD

    ;A = A + B
    add rax, rbx
    ;A = A-C
    sub rax, rcx
    ;A = A +D
    add rax, rdx

    ret
funcao2 endp
```

```
;;;A = B + C - D + 10
funcao3 proc
    mov rax, valorA
    mov rbx, valorB
    mov rcx, valorC
    mov rdx, valorD
    ; A= A+B
    add rax, rbx
    ;A = A+C
    add rax, rcx
    ;A = A - D
    sub rax, rdx
    ;A = A +10
    add rax, 10

    ret
funcao3 endp
```

```
;;;o A = B + C + (D - E)
funcao4 proc
    mov rax, valorA
    mov rbx, valorB
    mov rcx, valorC
    mov rdx, valorD
    mov rsi, valorE

    ; B+C
    add rax, rbx
    add rax, rcx

    ;D-E
    sub rdx, rsi

    ;SOMA DOS RESULTADOS
    add rax, rdx

    ret
funcao4 endp
```

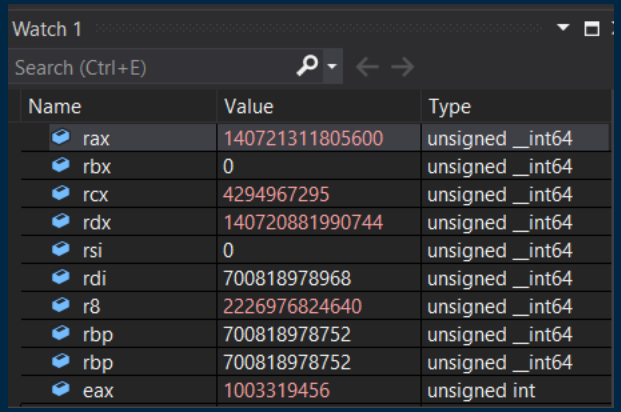
Códigos

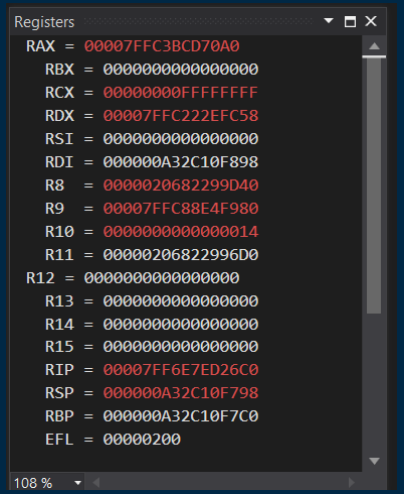
```
;;loop até zero
funcao5 proc
;  = 5;
mov  rax, 5
volta_loop:
; while (i > 0)
 cmp     rax, 0
 jle     SHORT fim

; i--;
  dec    rax
  jmp    SHORT volta_loop

fim:
        ret
funcao5 endp
```

```
funcao6 proc
;A = 5;
mov  rbx, 5
;B = 7;
mov  rcx, 7
; M = 0;
mov  rax, 0

;if (A > B)
cmp  rbx, rcx
jle  SHORT e_menor

; M = A;
mov  rax, rbx
jmp  SHORT fim
e_menor:
; else
; M = B;
mov  rax, rcx
fim:
ret
funcao6 endp
```

```
;;multiplicação
funcao7 proc
;A = 0;
mov  rax, 0
;B = 7;
mov  rbx, 7
;C = 5;
mov  rcx, 5

; coloca o B no A
mov  rax, rbx
; A = B * C;
imul rax, rcx

funcao7 endp

end
```

Registradores





64-bit register	Lower 32 bits	Lower 16 bits	Lower 8 bits
rax	eax	ax	al
rbx	ebx	bx	bl
rcx	ecx	cx	cl
rdx	edx	dx	dl
rsi	esi	si	sil
rdi	edi	di	dil
rbp	ebp	bp	bpl
rsp	esp	sp	spl
r8	r8d	r8w	r8b
r9	r9d	r9w	r9b
r10	r10d	r10w	r10b
r11	r11d	r11w	r11b
r12	r12d	r12w	r12b
r13	r13d	r13w	r13b
r14	r14d	r14w	r14b
r15	r15d	r15w	r15b

Há duas maneiras de visualizar os dados durante o debug
Através do Watch, onde converte automaticamente pra float
e através da janela dos Registradores.