Сохранение регистров и переменных при вызове процедур

Последнее обновление: 02.07.2023

Процедура может активно задействовать регистры для различных задач. Например:

```
.code
sum proc
    mov rbx, 5
    mov rax, 10
    add rax, rbx
    ret
sum endp

main proc
    mov rbx, 125
    call sum
    add rax, rbx    ; RAX = 20
    ret
main endp
end
```

Для примера здесь определена процедура sum, которая устанавливает регистры rbx и rax и складывает их значения, помещая результат в регистр rax.

В функции main помещаем в регистр RBX чило 125, затем вызываем процедуру sum. Далее результат процедуры sum - содержимое регистра RAX складываем с числом из регистра RAX. Однако, поскольку вызов процедуры sum изменил значение регистра RBX, то в после ее вызова в этом регистре будет не число 125, а число 5. Соответственно итоговый результат может отличаться от ожидаемого. В данном случае в регистре RAX будет число 20, а не 140.

Кончено, мы могли бы выбрать разные регистры, но данный пример - упрощение, в реальной процедуре одновременно может быть задействовано для сложных вычислений множество регистров. Кроме того, разрабатываемые процедуры могут использоваться во внешней программе, и при разработке процедуры мы можем не знать, какие именно регистры и как именно будет использовать внешняя программа. Поэтому хорошей, а иногда и необходимой практикой при вызове процедуры является сохранение значений используемых регистров в стек, а при завершении процедуры - их восстановление. Например:

```
.code
sum proc
    push rbx        ; сохраняем в стек регистр rbx
    mov rbx, 5
    mov rax, 10
    add rax, rbx
    pop rbx         ; восстанавливаем из стека регистр rbx
    ret
sum endp

main proc
    mov rbx, 125
    call sum
    add rax, rbx    ; RAX = 140
    ret
main endp
end
```

В данном случае в начале процедуры sum сохраняем значение rbx, а при завершении восстанавливаем его. Подобным образом стоит сохранять и восстанавливать все используемые регистры в стек. Регистр RAX в данном случае не сохраняется, так как мы ожидаем, что через него функция main получить результат процедуры sum.

При этом в принципе необязательно сохранять регистры в вызываемой процедуре, мы могли бы это сделать и в вызывающей процедуре:

```
.code
sum proc
    mov rbx, 5
    mov rax, 10
    add rax, rbx
    ret
sum endp

main proc
    mov rbx, 125
    push rbx        ; сохраняем в стек регистр rbx
    call sum
    pop rbx         ; восстанавливаем из стека регистр rbx
    add rax, rbx
    ret
main endp
end
```

Однако сохранение регистров в вызываемой процедуре может быть более предпочтительным, так как функция main может вызывать кучу других процедур и постоянно сохранять/восстанавливать регистры при каждом вызове значительно утяжелит код функции main.

Стоит отметить, что при взаимодействии с внешними API может потребоваться в обязательном порядке сохранять данные некоторых регистров. Так, если мы взаимодействуем с Windows API, то есть общие соглашения, что при вызове функций WinAPI необходимо сохранять в стек, а после вызова восстанавливать регистры RBX, RBP, RDI, RSI, RSP, R12, R13, R14 и R15, а также XMM6 - XMM15.

Сохранение и восстановление переменных

Все, что касается регистров, также можно отнести и к глобальным переменным, которые используются в процедурах:

```
.data
    num word 45
.code
sum proc
    mov num, 15
    mov rax, 10
    add ax, num     ; AX = AX + num = 10 + 15 = 25
    ret
sum endp

main proc
    call sum
    add ax, num     ; AX = AX + num = 25 + 15 = 40
    ret
main endp
end
```

В данном случае в функции main мы ожидаем, что переменная num будет равна 45, но вызов процедуры sum меняет ее значение на 45. И подобным образом в вызываемой процедуре переменную можно сохранить в стек:

```
.data
    num word 45
.code
sum proc
    push num        ; сохраняем значение num в стек
    mov num, 15
    mov rax, 10
    add ax, num     ; AX = AX + num = 10 + 15 = 25
    pop num         ; восстанавливаем num из стека
    ret
sum endp

main proc
    call sum
    add ax, num     ; AX = AX + num = 25 + 45 = 70
    ret
main endp
end
```

Назад Содержание Вперед