

# IntToStr/StrToInt

---

**Source:**

<http://coding.derkeiler.com/Archive/Delphi/borland.public.delphi.language.basm/2005-08/msg00195.html>

---

- *From:* gordy <[gordy@xxxxxxxxxxxxxxxxxxxxxx](mailto:gordy@xxxxxxxxxxxxxxxxxxxxxx)>
  - *Date:* Thu, 11 Aug 2005 16:55:15 -0700
- 

Hi, I just recently came across the fastcode project online. I have kept my own unit over the years with asm optimized alternatives to system functions.

a couple that I use often are inttostr and strtoint. I didn't find these included in the fastcode project. I'm also not sure where to submit code for this project so I figured I would just post it here:

these are signed versions. I never had a need for unsigned versions so I never made them...

```
function IntToStr(Value: Integer): ShortString;
// Value = eax
// Result = edx
asm
    push ebx
    push esi
    push edi

    mov edi,edx
    xor ecx,ecx
    mov ebx,10
    xor edx,edx

    cmp eax,0 // check for negative
    setl dl
    mov esi,edx
    jnl @reads
    neg eax

@reads:
    mov edx,0 // edx = eax mod 10
    div ebx // eax = eax div 10
    add edx,48 // '0' = #48
    push edx
    inc ecx
```

## IntToStr/StrToInt

```
    cmp  eax,0
    jne  @reads

    dec  esi
    jnz  @positive
    push 45 // '-' = #45
    inc  ecx

@positive:
    mov  [edi],cl // set length byte
    inc  edi

@writes:
    pop  eax
    mov  [edi],al
    inc  edi
    dec  ecx
    jnz  @writes

    pop  edi
    pop  esi
    pop  ebx
end;

function StrToInt(Value: ShortString): Integer;
// Value   = eax
// Result  = eax
asm
    push ebx
    push esi

    mov  esi,eax
    xor  eax,eax
    movzx ecx,Byte([esi]) // read length byte
    cmp  ecx,0
    je  @exit

    movzx ebx,Byte([esi+1])
    xor  edx,edx // edx = 0
    cmp  ebx,45 // check for negative '-' = #45
    jne @loop
```

## IntToStr/StrToInt

```
dec edx // edx = -1
inc esi // skip '-'
dec ecx
```

```
@loop:
    inc esi
    movzx ebx,Byte([esi])
    imul eax,10
    sub ebx,48 // '0' = #48
    add eax,ebx
    dec ecx
    jnz @loop
```

```
mov ecx,eax
and ecx,edx
shl ecx,1
sub eax,ecx
```

```
@exit:
    pop esi
    pop ebx
end;
```

.