If you quickly need to flip the bits in a byte in reverse (turning bits from 01234567 to 76543210) you can use this unrolled loop.

```
        ldx #0
.for(var i=0;i<8;i++)
{
        lsr // shift A down, bit 0 to C
        tay // copy to Y doesn't change C
        txa // pull x to a, doesn't change C
        rol // shift left, C to bit 0
        tax // stash a in x
        tya // get start a back from y
}
        txa
```

(kickassembler loop syntax)

This takes a byte in A, reverses the bits and exits with the reversed bits in A again, using X and Y for temporary storage and only short 2-cycle instructions in the loop. Of course a table lookup will be faster if you do this a lot in your code.

– this would be equally fast, at 100cycl +rts (enter with the value in .A, result in .A, .X will be 0)

```
        ldx #8
loop    asl
        ror $2
        dex
        bne loop
        lda $2

        rts
```

or unrolled (56 cycl +rts):

```
        asl
        ror $2
        asl
        ror $2
        asl
        ror $2
        asl
        ror $2
        asl
        ror $2
        asl
        ror $2
        asl
        ror $2
        asl
        ror $2
        asl
        lda $2
        ror
```

```
        rts
```

– Optimized version, at 84cycl +rts (enter with the value in .A, result in .A)

```
    sta    $02    ; 3
    lda    #1    ; 2
lp:
    ror    $02    ; 5
    rol        ; 2
    bcc    lp    ; 3(2)
            ; =84
    rts
```

or slightly unrolled:

```
    sta    $02    ; 3
    lda    #1    ; 2
lp:
    ror    $02    ; 5
    rol        ; 2
    ror    $02    ; 5
    rol        ; 2
    bcc    lp    ; 3(2)
            ; =72
    rts
```