

# The Pre-Loaded Loop

By Brian Cypress

I know that a lot of programmers and managers are going to say I'm nit-picking, but I've seen non-pre-loaded loop so many times in both packaged and custom code that I think its important to show what a pre-loaded loop is.

Most loops are groups of instructions to be performed based on some event or condition. The problem arises when the event is the **first** instruction in the loop. The following code is a non-pre-loaded loop.

```
1234567890123456789012345678901234567890123456789012345678901234567890
C          Factor 1 OpCo Factor 1 Result      > < =
C          SETOF                                     50
C          SOMKEY  SETLLSOMFIL
C          *IN50   DOWEQ*OFF
C          SOMKEY  READESOMFIL                      50
C          *IN50   IFEQ *OFF
C          some more instructions....
C          END
C          END
```

As you can see in the above code, indicator 50 is tested **twice** for every iteration of the loop (Once for loop control, once after the read). Below is an example of the same code, but as a pre-loaded loop.

```
1234567890123456789012345678901234567890123456789012345678901234567890
C          Factor 1 OpCo Factor 1 Result      > < =
C          SOMKEY  CHAIN SOMFIL                    50
C          *IN50   DOWEQ*OFF
C          some more instructions....
C          SOMKEY  READESOMFIL                      50
C          END
```

As you can now see, indicator 50 is only tested once. The secret is performing the event **before** the loop, and then once again in the loop *as the last instruction* in the loop.

I can hear some of you saying "*So what? Its only one extra test!*". The question is how many of these loops exist in each program. And as for the question of readability, IMHO (in my humble opinion, for those who don't surf), I think the pre-loaded loop is easier.