

Documentation Développeurs Apple Computer France 1987

Document développeur numéro 40

Numéric Magnitude comp on IIGS

type d'upgrade de ce document : 4

- 1 Documentation de première catégorie inchangée
- 2 Documentation de deuxième catégorie mise à jour
- 3 Documentation de deuxième catégorie inchangée
- 4 Mise à jour payante de la documentation de première catégorie
- 5 Mise à jour gratuite de la documentation de première catégorie
- 6 Nouveautés payantes non vitales
- 7 Nouveautés gratuites et vitales

Taille : 4 page(s) environ

Domaine : 816

VERSION : rev 2
DATE : 20.11.85

1. The first part of the document is a list of names and addresses.

2. The second part of the document is a list of names and addresses.

3. The third part of the document is a list of names and addresses.

4. The fourth part of the document is a list of names and addresses.

5. The fifth part of the document is a list of names and addresses.

6. The sixth part of the document is a list of names and addresses.

7. The seventh part of the document is a list of names and addresses.

8. The eighth part of the document is a list of names and addresses.

9. The ninth part of the document is a list of names and addresses.

10. The tenth part of the document is a list of names and addresses.

11. The eleventh part of the document is a list of names and addresses.

12. The twelfth part of the document is a list of names and addresses.

13. The thirteenth part of the document is a list of names and addresses.

14. The fourteenth part of the document is a list of names and addresses.

Numeric Magnitude Comparisons on the 65816

To: Gumbo Team
From: Jim Jatczynski
Date: First draft: 11/13/85; Revision 1: 11/14/85; Revision 2: 11/20/85

SHOULD I READ THIS MEMO?

Consider the following code for testing $p < q$ where p and q are 16-bit two's complement integers:

```
lda    p
cmp    q
bmi    dest
```

If you think this works, you should probably read this memo. It describes correct methods of performing signed and unsigned arithmetic comparisons on the 65816. If you have read previous versions of this memo, throw them away and save this one—it has additional routines.

NOTATION

Let n , z , v , and c be the negative, zero, overflow, and carry bit values and $\sim n$, $\sim z$, $\sim v$, and $\sim c$ be their complements. Adjacency indicates logical and and "+" indicates logical or.

DID YOU KNOW SBC AND CMP ARE NOT THE SAME?

Subtraction affects the n , z , v , and c status bits; comparison affects only the n , z , and c bits. As you will see, knowing the value of v is crucial in certain comparisons. Such comparisons must use SBC rather than CMP to compare magnitudes of the operands.

SIGNED TWO'S COMPLEMENT COMPARISON

We want to compare two 16-bit two's complement integers, p and q , and branch to location `dest` if one of the conditions $<$, \leq , $=$, \neq , \geq , or $>$ is met. We can set the condition codes by executing

```
lda    p
sec
sbc    q
{ conditional branch code }
```

The following table shows the appropriate branch conditions in terms of the n , z , v and c statusbits:

$p < q$	$n\sim v + \sim nv$
$p \leq q$	$z + n\sim v + \sim nv$
$p = q$	z
$p \neq q$	$\sim z$
$p \geq q$	$nv + \sim n\sim v$
$p > q$	$nv\sim z + \sim n\sim v\sim z$

It is immediately clear from this table why the code shown in the first section doesn't work. First, CMP doesn't even affect the v flag. But, even if it did, the branch instruction BMI only looks at the n flag which will only be correct if there is no overflow during the subtraction.

So, correct code for "if $p < q$ then go to `dest` else continue" might look like this:

Administrative records of the Department of Justice, Office of the Inspector General, dated 1998.

1998

Administrative records of the Department of Justice, Office of the Inspector General, dated 1998.

1998

Administrative records of the Department of Justice, Office of the Inspector General, dated 1998.

1998

Administrative records of the Department of Justice, Office of the Inspector General, dated 1998.

1998

Administrative records of the Department of Justice, Office of the Inspector General, dated 1998.

1998

```

        lda    p            ;load p
        sec                    ;preset carry for proper subtraction
        sbc    q            ;compare and set condition codes
        bvs    lab1         ;check v
        bmi    dest         ;here ~v, check n
        bra    continue     ;here ~v~n
lab1    bpl    dest         ;here v, check n
continue ;here the branch condition is not met
. . . . .
dest    ;here the branch condition is met, i.e.
        ;n~v + ~nv is true

```

Rich Williams has pointed out a neat alternative to this code that saves 1 byte in native mode and 2 bytes in 8-bit mode:

```

        lda    p            ;branch if p<q
        sec                    ;preset carry for proper subtraction
        sbc    q            ;compare and set condition codes
        bvs    lab1         ;check v, branch if set
        eor    #$8000       ;here ~v, flip sign of result
lab1    bpl    dest         ;check n and v have different values
. . . . .
dest

```

Code similar to the first example can be written for all the other branch conditions. For example, the code for \geq is as follows:

```

        lda    p            ;branch if p≥q
        sec                    ;preset carry for proper subtraction
        sbc    q            ;compare and set condition codes
        bvs    lab1         ;check v, branch if set
        bpl    dest         ;here ~v, flip sign of result
lab1    bra    continue     ;here ~v~n
continue ;here the branch condition is not met
. . . . .
dest    ;here the branch condition is met, i.e.
        ;nv + ~n~v

```

Rich's method can also be applied to this branch condition as follows:

```

        lda    p            ;branch if p≥q
        sec                    ;preset carry for proper subtraction
        sbc    q            ;compare and set condition codes
        bvs    lab1         ;check v, branch if set
        eor    #$8000       ;here ~v, flip sign of result
lab1    bmi    dest         ;check n and v have different values
. . . . .
dest

```

The code for $=$ and \neq is simple because each involves only the z bit.

```

        lda    p            ;branch if p=q
        cmp    q            ;can use CMP because only z is needed
        beq    dest

        lda    p            ;branch if p≠q
        cmp    q
        bne    dest

```

THE UNIVERSITY OF CHICAGO
DEPARTMENT OF CHEMISTRY

RESEARCH REPORT
NO. 1000
BY
J. H. GOLDSTEIN
AND
R. F. FIESHER

DEPARTMENT OF CHEMISTRY
5780 SOUTH CAMPUS DRIVE
CHICAGO, ILLINOIS 60637

RECEIVED
MAY 15 1968

THIS REPORT IS THE PROPERTY OF THE UNIVERSITY OF CHICAGO
AND IS LOANED TO YOUR INSTITUTION. IT AND ITS CONTENTS
ARE NOT TO BE DISTRIBUTED OUTSIDE YOUR INSTITUTION.

SYNOPSIS

The infrared spectra of the polyacetylene-iodine complex have been studied in the region 1000-1500 cm⁻¹. The results show that the complex is formed by the addition of iodine to the double bonds of the polyacetylene chain. The iodine atoms are located in the interchain spaces and are coordinated to the carbon atoms of the chain. The complex is stable in the solid state and does not decompose at room temperature.

Wavenumber (cm ⁻¹)	Assignment
1450	ν(C-H)
1380	ν(C-H)
1300	ν(C-H)
1250	ν(C-H)
1150	ν(C-H)
1050	ν(C-H)

INTRODUCTION
The polyacetylene-iodine complex has been studied extensively in the literature. It is a well-known example of a charge-transfer complex. The complex is formed by the addition of iodine to the double bonds of the polyacetylene chain. The iodine atoms are located in the interchain spaces and are coordinated to the carbon atoms of the chain. The complex is stable in the solid state and does not decompose at room temperature.

The code for $>$ and \leq is more complex because it involves the n, z, and v bits. Rich's method can be written as follows:

```

        lda    p            ;branch if p>q
        sec
        sbc    q
        beq    continue    ;if equal, don't branch to dest
        bvs    lab1
        eor    #$8000
lab1    bmi    dest
continue
. . . . .
dest

```

```

        lda    p            ;branch if p≤q
        sec
        sbc    q
        beq    dest        ;if equal, branch to dest now
        bvs    lab1
        eor    #$8000      ;flip sign of result
lab1    bpl    dest
. . . . .
dest

```

The code above is actually most applicable if operand p is already in the accumulator. When this isn't the case, it may be more efficient to reverse the operands and use the opposite branch condition to get more faster code. For example, the code for $<$ is generally faster than the code for $>$, and the code for \geq is generally faster than the code for \leq .

UNSIGNED COMPARISON

Unsigned comparisons can be done using the sequence "LDA p, CMP q" because the v bit is not involved in the comparison conditions which are as follows:

```

p < q    ~c
p ≤ q    [see below]
p = q    z
p ≠ q    ~z
p ≥ q    c
p > q    [see below]

```

Thus, the code for $<$ and \geq is very simple:

```

lda    p
cmp    q
bcc    dest    ;branch to dest if p<q

lda    p
cmp    q
bcs    dest    ;branch to dest if p≥q

```

Many assemblers have the aliases blt and bge for bcc and bcs, respectively, to make code reading easier.

The easiest way to test the conditions \leq and $>$ is to reverse the operands and use the code above. For example, to test $p \leq q$, use the following:

1. The first part of the document is a list of names and addresses of the members of the committee.

MEMBERS OF THE COMMITTEE

The following is a list of the names and addresses of the members of the committee, as of the date of the meeting.

1. Mr. J. H. Smith, 123 Main Street, New York, N. Y.

2. Mr. R. L. Jones, 456 Broadway, New York, N. Y.

3. Mr. W. D. Brown, 789 Park Avenue, New York, N. Y.

4. Mr. T. E. White, 1010 Fifth Avenue, New York, N. Y.

5. Mr. S. K. Green, 1212 Madison Avenue, New York, N. Y.

6. Mr. M. P. Black, 1414 Lexington Avenue, New York, N. Y.

7. Mr. N. O. Gray, 1616 Central Park West, New York, N. Y.

8. Mr. P. Q. Blue, 1818 Riverside Drive, New York, N. Y.

9. Mr. R. S. Red, 2020 West End Avenue, New York, N. Y.

10. Mr. T. U. Purple, 2222 East 86th Street, New York, N. Y.

11. Mr. V. W. Yellow, 2424 East 72nd Street, New York, N. Y.

12. Mr. X. Y. Orange, 2626 East 58th Street, New York, N. Y.

13. Mr. Z. A. Pink, 2828 East 44th Street, New York, N. Y.

14. Mr. B. C. Light, 3030 East 30th Street, New York, N. Y.

15. Mr. D. E. Green, 3232 East 16th Street, New York, N. Y.

16. Mr. F. G. Blue, 3434 East 2nd Street, New York, N. Y.

17. Mr. H. I. Red, 3636 East 12th Street, New York, N. Y.

18. Mr. J. K. Purple, 3838 East 22nd Street, New York, N. Y.

19. Mr. L. M. Yellow, 4040 East 32nd Street, New York, N. Y.

20. Mr. N. O. Orange, 4242 East 42nd Street, New York, N. Y.

21. Mr. P. Q. Pink, 4444 East 52nd Street, New York, N. Y.

22. Mr. R. S. Light, 4646 East 62nd Street, New York, N. Y.

23. Mr. T. U. Green, 4848 East 72nd Street, New York, N. Y.

24. Mr. V. W. Blue, 5050 East 82nd Street, New York, N. Y.

25. Mr. X. Y. Red, 5252 East 92nd Street, New York, N. Y.

26. Mr. Z. A. Purple, 5454 East 102nd Street, New York, N. Y.

27. Mr. B. C. Yellow, 5656 East 112nd Street, New York, N. Y.

28. Mr. D. E. Orange, 5858 East 122nd Street, New York, N. Y.

29. Mr. F. G. Pink, 6060 East 132nd Street, New York, N. Y.

30. Mr. H. I. Light, 6262 East 142nd Street, New York, N. Y.


```
lda    q
cmp    p
bcs    dest    ;branch to dest if p≤q
```

MULTIPLE PRECISION COMPARISONS

The code for multiple precision comparisons is more complicated, so we don't show any of it here. A good reference for multiple precision comparison routines is Leventhal's and Saville's *6502 Assembly Language Subroutines*, OSBORNE/McGraw-Hill.

