Order this document
by EB253/D

Motorola Semiconductor Engineering Bulletin

EB253

How to Use the Table Lookup and Interpolate Instruction on the CPU32

By Sharon Darley
Austin, Texas

Introduction

The table lookup and interpolate instruction approximates a number that lies between two consecutive entries in a lookup table. It can be used to approximate a number that is a function of one variable or a number that is a function of several variables. This engineering bulletin focuses on the case where only one variable is involved.

An explanation of how to handle cases with multiple variables is given on page 4-203 in the *CPU32 Reference Manual*, Motorola document order number CPU32RM/AD, Rev. 1. In general, section 4.6 in the reference manual is devoted to discussion of the table lookup and interpolation instruction and gives several practical examples, as does section 7.7 in the textbook *The Motorola MC68332 Microcontroller*, Motorola document order number TB325/D. Reading at least one of these references is strongly recommended.

General Information

The table lookup and interpolate function assumes that the result (point on the function) falls linearly between the two consecutive entry points in the table. Thus, for a linear function, few points in the table are needed.

Freescale Semiconductor, Inc.

© Motorola, Inc., 1999

MOTOROLA

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

In fact, for a function that is simply a line, all the instruction really needs is a start point and an end point. However, more complex functions require more points in the table, particularly in the most non-linear regions. An example in this snapshot illustrates how to handle very non-linear regions.

Formats

The basic syntax of the table lookup and interpolate instruction is:

TBL <S%><R>.<l> <EA>,Dx

Where:

<S> = S (signed) or U (unsigned)

<R> = N (unrounded) or nothing (rounded)

<l> = B (byte), W (word), or L (long word)

<EA> = the starting address of the lookup table in memory

Dx = a data register that holds the independent variable before execution and the interpolated result after execution.

The table lookup and interpolate instruction has four formats. All four formats support byte, word, and long-word numbers.

TBLS returns a signed, rounded result with this format:

Table 1. Format of TBLS/TBLU Result

Length	31:24	23:16	15:8	7:0
Byte	Unaffected	Unaffected	Unaffected	Result
Word	Unaffected	Unaffected	Result	Result
Long	Result	Result	Result	Result

For More Information On This Product,
Go to: www.freescale.com

TBLSN returns a signed, unrounded result with this format:

Table 2. Format of TBLSN Result

Length	31:24	23:16	15:8	7:0
Byte	Sign extended	Sign extended	Result	Fraction
Word	Sign extended	Result	Result	Fraction
Long	Result	Result	Result	Fraction

TBLU returns an unsigned, rounded result. The result has the same format as for the TBLS instruction shown in **Table 1**.

TBLUN returns an unsigned, unrounded result with this format:

Table 3. Format of TBLUN Result

Length	31:24	23:16	15:8	15:8
Byte	Zero extended	Zero extended	Result	Fraction
Word	Zero extended	Result	Result	Fraction
Long	Result	Result	Result	Fraction

How the Table is Stored in Memory

The lookup table consists of 257 entries, entry number 0 to entry number 256. Each entry occurs at a multiple of 256 of the independent variable X.

The table can consist of fewer (or more) entry numbers, but the user program must scale the independent variable before executing the table lookup and interpolate instruction to obtain correct results. For simplicity, the 257 entry table will be discussed first.

Each entry in the table consists of two components: X and Y, where Y = F(X). Only the values for Y are actually stored in memory; the values of X are assumed. As an example, take the function where Y is the square root of X.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

The lookup table for this function is shown in Table 4. Note in the table that the values for Y are approximated to the nearest whole number. The lookup and interpolate instruction can return a fractional result, but it cannot read a fractional input.

Table 4. F(x) = $x^{(1/2)}$

Entry Number	X Value (Decimal)	Y Value (Decimal)	X Value (Hex)	Y Value (Hex)
0	0	0	0	0
1	256	16	100	10
2	512	23	200	17
3	768	28	300	1C
4	1024	32	400	20
5	1280	36	500	24
6	1536	39	600	27
7	1792	42	700	2A
8	2048	45	800	2D
:	:	:	:	:
253	64768	254	FD00	FE
254	65024	255	FE00	FF
255	65280	255	FF00	FF
256	65536	256	10000	100

How the Result Is Calculated

The number that the user code must write to data register Dx before instruction execution has this format:

Table 5. Format for Data Register Dx

Not used	Table entry offset	Interpolation fraction

The upper 16 bits of the data register are not used. Bits 15:8 hold the table entry offset, which is the entry number that corresponds to the X value that is closest to and less than the number to be interpolated. The interpolation fraction is the difference between the number to be interpolated and the value of X that lies at the previous entry number. As an example, using Table 4, determine the format for data register Dx for the independent variable X = 361 (hex $169).

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

As shown in **Table 4**, the table entry offset that is closest to and less than 361 is 1. The interpolation fraction is \$169 – \$100 = \$69 (decimal 105).

Thus, the value that the user code must write to data register Dx is \$0169. Note that this is the same as the original number to be interpolated. Thus, for a table that has 256 values of X between each entry point, the user code does not have to do any calculations to convert the number to be interpolated to the number that must be written in data register Dx.

The CPU calculates the interpolated result as:

Y = {(F(n+1) – Fn) x (Dx)[7:0] / 256} + Fn

Continuing with the example above, this would become:

Y = {(23–16) x 105 / 256} + 16 = 18.8

This is very close to the correct answer of 19, and in fact, the CPU will round the answer of 19 if the TBLU or TBLS instruction is executed.

However, now take the example of X = 64 (\$40). The CPU calculates the interpolated result as:

Y = {16 – 0) x 64 / 256} + 0 = 4

This answer is clearly wrong (the correct answer should be 8). The square root function is non-linear in the region (X,Y) = (64,8). The function does not become relatively linear until the region around the point (X,Y) = (256,16). Thus, the table lookup and interpolate instruction will only return an accurate result (particularly if the result is rounded) after that point. The area of the curve below that point can be handled by making a separate lookup table for that area, where the entry points are closer together. An example of such a table is shown in example 2.

Freescale Semiconductor, Inc.

Example 1

This example uses **Table 4** to estimate values of Y, where Y is equal to the square root of X. Note that there are 256 interpolation values between every two entry points in **Table 4**. The results of this example will be very accurate for values of X greater than 256 (\$100) and very inaccurate for values of X less than 256.

For More Information On This Product,
Go to: www.freescale.com

```
        ORG $400
        MOVE.W #$1000,A0  ;address of beginning of table
        MOVE.W #$02A4,D0  ;This is the number to be interpolated.
                          ;Experiment with this value to obtain
                          ;different results.
                          ;In this case, X = 676 decimal.
INTERP  TBLU.W (A0),D0    ;Do word-long, unsigned, rounded
                          ;interpolation.
                          ;The result (26 = $1A) is returned in D0.
DONE    BRA DONE

        ORG $1000         ;Beginning of the Lookup Table. The
                          ;values of Y only are stored
        DW $0000          ;Y = $0, for X = $0 for X = $100
        DW $0017          ;Y = $17 for X = $200
        DW $001C          ;Y = $1C for X = $300
        DW $0020          ;Y = $20 for X = $400
        DW $0024          ;Y = $24 for X = $500
        DW $0027          ;Y = $27 for X = $600
        DW $002A          ;Y = $2A for X = $700
        DW $002D          ;Y = $2D for X = $800
        :
        :
```

*** Continue the lookup table until X = $10000 ***

Example 2

This example is the same as the previous one, except that it uses two lookup tables: one for the more linear region above X = 256 and one for the very non-linear region below 256. The spacing for the table that includes values where X is greater than or equal to 256 is the same as for the first example (**Table 4**). However, the spacing for the table that includes values where X is less than 256 is such that there are only 16 ($10) interpolation values between every two entry points. This table is shown in **Table 6**. To use **Table 6**, the user code must scale the X value before executing the interpolation instruction by shifting it left by four.

As a side note, this example will not return accurate results for values of X < 16. To increase the accuracy within the number range 0 < X < 16, a third table could be added to handle these values.

For More Information On This Product,
Go to: www.freescale.com

Freescale Semiconductor, Inc.

Table 6. F(x) = $x^{(1/2)}$, x <$100

Entry Number	X Value (Decimal)	Y Value (Decimal)	X Value (Hex)	Y Value (Hex)
0	0	0	0	0
1	16	4	10	4
2	32	6	20	6
3	48	7	30	7
4	64	8	40	8
5	80	9	50	9
6	96	10	60	A
7	112	11	70	B
8	128	11	80	B
9	144	12	90	C
:	:	:	:	:
15	240	15	F0	F
16	256	16	100	10

```
        ORG $400
        MOVE.W #$1000,A0 ;address of beginning of table for
                         ;X>=256 ($100).
        MOVE.W #$2000,A1 ;address of beginning of table for
                         ;X<256 ($100).
        MOVE.W #$0019,D0 ;This is the number to be
                         ;interpolated.Experiment
                         ;with this value to obtain different
                         ;results.
        CMPI.W #$100,D0  ;test X to see which table to use
        BLT   LOW_NUM    ;if X < $100, then use scaled table
        TBLU.W (A0),D0   ;Interpolate if X >=256 (in this case,
                         it is not).
        BRA DONE
LOW_NUM LSL.W #$4,D0     ;To use scaled table, must scale value
                         ;of X first.
        TBLU.W (A1),D0   ;Y is returned in D0.In this case,Y=5.
DONE    BRA DONE

        ORG $1000        ;start of table that will be used when
                         ;X>=256.
        DW $0000         ;Y = $0, for X = $0
        DW $0010         ;Y = $10 for X = $100
        DW $0017         ;Y = $17 for X = $200
        DW $001C         ;Y = $1C for X = $300
        DW $0020         ;Y = $20 for X = $400
```


Freescale Semiconductor, Inc.

Engineering Bulletin

```
DW $0024        ;Y = $24 for X = $500
DW $0027        ;Y = $27 for X = $600
DW $002A        ;Y = $2A for X = $700
DW $002D        ;Y = $2D for X = $800
        :
        :
        :
```

*** Continue the lookup table until X = $10000 ***

```
ORG $2000       ;start of table that will be used when
                ;X<256.
DW $0000        ;Y = 0 for X = 0.
DW $0004        ;Y = $4 for X = $10
DW $0006        ;Y = $6 for X = $20
DW $0007        ;Y = $7 for X = $30
DW $0008        ;Y = $8 for X = $40
DW $0009        ;Y = $9 for X = $50
DW $000A        ;Y = $A for X = $60
$000B           ;Y = $B for X = $70
DW $000B        ;Y = $B for X = $80
DW $000C        ;Y = $C for X = $90
        :
        :
        :
```

*** Continue the lookup table until X = $100 ***

Motorola reserves the right to make changes without further notice to any products herein. Motorola makes no warranty, representation or guarantee regarding the suitability of its products for any particular purpose, nor does Motorola assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation consequential or incidental damages. "Typical" parameters which may be provided in Motorola data sheets and/or specifications can and do vary in different applications and actual performance may vary over time. All operating parameters, including "Typicals" must be validated for each customer application by customer's technical experts. Motorola does not convey any license under its patent rights nor the rights of others. Motorola products are not designed, intended, or authorized for use as components in systems intended for surgical implant into the body, or other applications intended to support or sustain life, or for any other application in which the failure of the Motorola product could create a situation where personal injury or death may occur. Should Buyer purchase or use Motorola products for any such unintended or unauthorized application, Buyer shall indemnify and hold Motorola and its officers, employees, subsidiaries, affiliates, and distributors harmless against all claims, costs, damages, and expenses, and reasonable attorney fees arising out of, directly or indirectly, any claim of personal injury or death associated with such unintended or unauthorized use, even if such claim alleges that Motorola was negligent regarding the design or manufacture of the part. Motorola and Ⓜ are registered trademarks of Motorola, Inc. Motorola, Inc. is an Equal Opportunity/Affirmative Action Employer.

How to reach us:
USA/EUROPE/Locations Not Listed: Motorola Literature Distribution, P.O. Box 5405, Denver, Colorado 80217, 1-800-441-2447 or 1-303-675-2140. Customer Focus Center, 1-800-521-6274
JAPAN: Motorola Japan Ltd.: SPD, Strategic Planning Office, 141, 4-32-1 Nishi-Gotanda, Shinagawa-ku, Tokyo, Japan, 03-5487-8488
ASIA/PACIFIC: Motorola Semiconductors H.K. Ltd., Silicon Harbour Centre, 2 Dai King Street, Tai Po Industrial Estate, Tai Po, New Territories, Hong Kong, 852-26629298
Mfax™, Motorola Fax Back System: RMFAX0@email.sps.mot.com; http://sps.motorola.com/mfax/; TOUCHTONE, 1-602-244-6609; US and Canada ONLY, 1-800-774-1848
HOME PAGE: http://motorola.com/sps/

Mfax is a trademark of Motorola, Inc.

© Motorola, Inc., 1999

MOTOROLA

**For More Information On This Product,
Go to: www.freescale.com**

Freescale Semiconductor, Inc.