

z/Architecture



# Preliminary Decimal-Floating-Point Architecture

November, 2006



**Note:**

Before using this information and the product it supports, be sure to read the general information under “Notices” on page v.

**November, 2006**

The facilities discussed in this publication is available on certain System z Processor Complexes. The information published herein should not be construed as implying any intention by IBM to provide these facilities on models other than those described herein.

This publication is provided for use in conjunction with other relevant IBM publications, and IBM makes no warranty, express or implied, relative to its completeness or accuracy. The information in this publication is current as of its publication date but is subject to change without notice.

© Copyright International Business Machines Corporation 1990-2006. All rights reserved.

US Government Users Restricted Rights — Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.



---

# Notices

IBM Corporation  
New Orchard Rd.  
Armonk, NY 10504  
U.S.A.

Produced in the United States of America  
November, 2006  
All Rights Reserved

The information in this document is intended to provide guidance for those implementing decimal-floating-point architecture. It discusses findings based on a solution that was created and tested under laboratory conditions. These findings may not be realized in all customer environments, and implementation in such environments may require additional steps, configurations, and performance analysis. The information herein is provided "AS IS" with no warranties, express or implied. This information does not constitute a specification or form part of the warranty for any IBM product. Implementation and certification of the solution rests on the implementation team. The users of this document should always check the latest release information for the applicable product and check the product Web pages for the latest updates and findings.

References in this publication to IBM products or services do not imply that IBM intends to make them available in every country in which IBM operates. Consult your local IBM business contact for information on the products, features and services available in your area.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, North Castle Drive, Armonk, NY, 10504-1785 USA.

Photographs shown are of engineering prototypes. Changes may be incorporated in production models.

This equipment is subject to all applicable FCC rules and will comply with them upon delivery.

Information concerning non-IBM products was obtained from the suppliers of those products. Questions concerning those products should be directed to suppliers.

IBM hardware products are manufactured from new parts, or new and used parts. Regardless, our warranty terms apply.

---

## Trademarks

IBM, the IBM logo, and z/Architecture are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

IEEE is a trademark of the Institute of Electrical and Electronics Engineers, Inc. in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java and all Java-based marks are trademarks or registered trademarks of Sun Microsystems Inc. in the United States and other Countries.

Other trademarks and registered trademarks are the properties of their respective companies.



---

# Preface

This is a preliminary description of the decimal-floating-point (DFP) facility and the floating-point-support-enhancement facility. It will be superseded by the next published version of the *IBM z/Architecture Principles of Operation* (SA22-7832).

The definitions, terminology, and document structure in this document is subject to change.

The reader should be familiar with the *IBM z/Architecture Principles of Operation* (SA22-7832).

Terms and concepts referred to in this publication but explained in the aforementioned publication are not explained again in this publication.

This document includes updates to some sections in Chapters 4, 6, 9, and 19 of the *IBM z/Architecture Principles of Operation*. Change bars are provided to indicate specific changes; a vertical ellipsis (⋮) is used to indicate material from these chapters which

has been omitted. This document also includes Chapter 20, which is a newly added chapter for which no change bar is provided.

The DFP and BFP facilities share the same floating-point control register even though the description of the floating-point control register is provided in Chapter 19 only.

To accommodate the inclusion of the DFP facility, the following name changes will be made:

- The instruction SET ROUNDING MODE is renamed SET BFP ROUNDING MODE.
- The rounding mode “biased round to nearest” is changed to “round to nearest with ties away from zero.”
- The rounding mode “round to nearest” is changed to “round to nearest with ties to even.”



# Chapter 4. Control

⋮

---

## Facility Indications

⋮

Bit	Meaning When Bit Is One
-----	-------------------------

⋮

- |    |  |
|----|--|
| 41 | The floating-point-support enhancement facility is installed in the z/Architecture architectural mode. |
| 42 | The DFP (decimal-floating-point) facility is installed in the z/Architecture architectural mode.       |

⋮

*Figure 4-18. Assigned Facility Bits*



# Chapter 6. Interruptions

:

## Program Interruption

:

### Data-Exception Code (DXC)

When a data-exception condition causes a program interruption, a data-exception code (DXC) is stored at location 147, and zeros are stored at locations 144-146. The DXC distinguishes between the various types of data-exception conditions. When the AFP-register (additional floating-point register) control bit, bit 45 of control register 0, is one, the DXC is also placed in the DXC field of the floating-point-control (FPC) register. The DXC field in the FPC register remains unchanged when any other program exception is reported. The DXC is an 8-bit code indicating the specific cause of a data exception. The data exceptions and data-exception codes are shown in Figure 6-1 and Figure 6-2 on page 6-2.

### Priority of Program Interruptions for Data Exceptions

When more than one data-exception condition applies and is enabled, the exception with the smallest DXC value is reported. Thus, for example, DXC 2 (BFP instruction) takes precedence over any IEEE exception condition.

When both a specification exception and an AFP-register data exception apply, it is unpredictable which one is reported.

DXC (Hex)	Data Exception
00	Decimal operand
01	AFP register
02	BFP instruction
03	DFP instruction
08	IEEE inexact and truncated
0B	IEEE inexact (IISE)
0C	IEEE inexact and incremented
10	IEEE underflow, exact
13	IEEE underflow, exact (IISE)
18	IEEE underflow, inexact and truncated
1B	IEEE underflow, inexact (IISE)
1C	IEEE underflow, inexact and incremented
20	IEEE overflow, exact
23	IEEE overflow, exact (IISE)
28	IEEE overflow, inexact and truncated
2B	IEEE overflow, inexact (IISE)
2C	IEEE overflow, inexact and incremented
40	IEEE division by zero
43	IEEE division by zero (IISE)
80	IEEE invalid operation
83	IEEE invalid operation (IISE)
<b>Explanation:</b>	
IISE	IEEE-interruption-simulation event.

Figure 6-1. Data-exception codes (DXC)

Exception	Applicable Instruction Types	Effect of CR0.45	Mask	Flag	DXC (Binary)	Interruption Action	DXC Placed in Real Loc 147	DXC Placed in FPC Byte 2
Decimal operand	Decimal <sup>1</sup>	0	none	none	0000 0000	Suppress or Terminate	Yes	No
		1					Yes	Yes
AFP register	FPS & HFP	0*	none	none	0000 0001	Suppress	Yes	No
BFP instruction	BFP	0*	none	none	0000 0010	Suppress	Yes	No
DFP instruction	DFP	0*	none	none	0000 0011	Suppress	Yes	No
IEEE invalid operation	BFP & non-IIS DFP	1*	0.0	1.0	1000 0000	Suppress	Yes	Yes
IEEE division by zero	BFP & non-IIS DFP	1*	0.1	1.1	0100 0000	Suppress	Yes	Yes
IEEE overflow	BFP & non-IIS DFP	1*	0.2	1.2	0010 xy00	Complete	Yes	Yes
IEEE underflow	BFP & non-IIS DFP	1*	0.3	1.3	0001 xy00	Complete	Yes	Yes
IEEE inexact	BFP & non-IIS DFP	1*	0.4	1.4	0000 1y00	Complete	Yes	Yes
IEEE invalid operation	IIS DFP	1*	0.0	1.0	1000 0011	Complete	Yes	Yes
IEEE division by zero (IISE)	IIS DFP	1*	0.1	1.1	0100 0011	Complete	Yes	Yes
IEEE overflow (IISE)	IIS DFP	1*	0.2	1.2	0010 w011	Complete	Yes	Yes
IEEE underflow (IISE)	IIS DFP	1*	0.3	1.3	0001 w011	Complete	Yes	Yes
IEEE inexact (IISE)	IIS DFP	1*	0.4	1.4	0000 1011	Complete	Yes	Yes

**Explanation:**

- <sup>1</sup> Decimal-operand data exception applies to the decimal instructions (Chapter 8), the general instructions COMPRESSION CALL and CONVERT TO BINARY (Chapter 7), and the DFP instructions CONVERT FROM SIGNED BCD and CONVERY FROM UNSIGNED BCD (Chapter 20).
- 0\* This exception is recognized only when CR0.45 is zero.
- 1\* This exception is recognized only when CR0.45 is one.
- xy Bits 4 and 5 of the DXC are set to 00, 10, or 11 binary, indicating that the result is exact, inexact and truncated, or inexact and incremented, respectively.
- y Bit 5 of the DXC is set to zero or one, indicating that the result is inexact and truncated or inexact and incremented, respectively.
- w Bit 4 of the DXC is set to bit 4 (x) of the signaling flags.
- BFP Binary-floating-point instructions (Chapter 19).
- DFP Decimal-floating-point instructions (Chapter 20).
- FPS Floating-point-support instructions (Chapter 9).
- HFP Hexadecimal-floating-point instructions (Chapter 18).
- IIS DFP LOAD FPC AND SIGNAL and SET FPC AND SIGNAL (Chapter 20).
- IISE IEEE-interruption-simulation event.
- Non-IIS DFP instructions other than LOAD FPC AND SIGNAL and SET FPC AND SIGNAL (Chapter 20).
- DFP

Figure 6-2. Data Exceptions

## Program-Interruption Conditions

⋮

### Data Exception

The data exceptions are shown in Figure 6-2 on page 6-2. A mask bit may or may not control whether an interruption occurs, as noted for each condition.

When a non-maskable data-exception condition is recognized, a program interruption for a data exception always occurs.

Each of the IEEE exceptions is controlled by a mask bit in the floating-point-control (FPC) register. The handling of these exception conditions is described in the section “IEEE Exception Conditions” in Chapter 9, “Floating-Point Overview and Support Instructions”.

Occurrence of an IEEE-interruption-simulation (IIS) event is controlled by the masks in the source operand of LOAD FPC AND SIGNAL or SET FPC AND SIGNAL (Chapter 20). The handling of this event is described in the instruction description of these instructions.

A data exception is recognized for the following cases:

- **Decimal-operand** data exception is recognized when an instruction which operates on decimal operands encounters invalid decimal digit or sign codes or has its operands specified improperly. The operation is suppressed, except that, for EDIT and EDIT AND MARK, the operation is terminated. See the section “Decimal-Operand Data Exception” in Chapter 8 “Decimal Instructions” for details. A decimal-operand data exception is also recognized when COMPRESSION CALL encounters errors in its dictionaries, and, in this case, the operation is terminated. The

decimal-operand data exception is reported with DXC 0.

- **AFP-register** data exception is recognized when bit 45 of control register 0 is zero, and a floating-point-support (FPS) instruction or a hexadecimal-floating-point (HFP) instruction specifies a floating-point register other than 0, 2, 4, or 6. The operation is suppressed and is reported with DXC 1.
- **BFP-instruction** data exception is recognized when bit 45 of control register 0 is zero and a binary-floating-point (BFP) instruction is executed. The operation is suppressed and is reported with DXC 2.
- **DFP-instruction** data exception is recognized when bit 45 of control register 0 is zero and a decimal-floating-point (DFP) instruction is executed. The operation is suppressed and is reported with DXC 3.
- **IEEE-exception** data exceptions are recognized when a BFP or DFP instruction encounters an IEEE exception condition which is enabled by the corresponding mask. The operation is suppressed or completed, depending on the type of condition. See the section “IEEE Exception Conditions” in Chapter 9, “Floating-Point Overview and Support Instructions” for details.
- **IEEE-interruption-simulation-event** data exceptions are recognized when a signaling flag is one and the corresponding source mask is enabled during the execution of LOAD FPC AND SIGNAL or SET FPC AND SIGNAL (Chapter 20). The operation is completed.

The instruction-length code is 1, 2, or 3.

The data exception is indicated by a program interruption code of 0007 hex (or 0087 hex if a concurrent PER event is indicated).

⋮



# Chapter 9. Floating-Point Overview and Support Instructions

⋮

## Registers and Controls

### Floating-Point Registers

All floating-point instructions (FPS, BFP, DFP, and HFP) use the same 16 floating-point registers. The floating-point registers are identified by the numbers 0-15 and are designated by a four-bit R field in floating-point instructions. Each floating-point register is 64 bits long and can contain either a short (32-bit) or a long (64-bit) floating-point operand.

⋮

### AFP-Register-Control Bit

Bit 45 of control register 0 is the AFP-register-control bit. The AFP registers, the BFP instructions, and the DFP instructions can be used successfully only when the AFP-register-control bit is one. Attempting to use one of the 12 additional floating-point registers by an FPS or HFP instruction when the AFP-register-control bit is zero results in an AFP-register data exception (DXC 1). Attempting to execute any BFP instruction when the AFP-register-control bit is zero results in a BFP-instruction data exception (DXC 2). Attempting to execute any DFP instruction when the AFP-register-control bit is zero results in a DFP-instruction data exception (DXC 3).

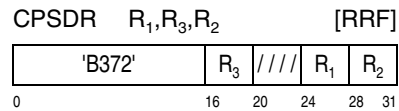
The initial value of the AFP-register-control bit is zero.

⋮

## Instructions

⋮

### COPY SIGN



The second operand is placed at the first-operand location with the sign bit set to the sign of the third operand. The first, second, and third operands are each in a 64-bit floating-point register. The sign bit of the second operand and bits 1-63 of the third operand are ignored.

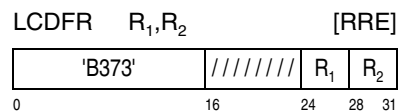
**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 1, AFP register
- Operation (if the floating-point-support enhancement facility is not installed)

⋮

### LOAD COMPLEMENT



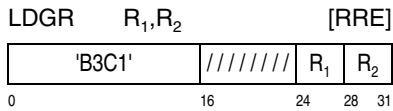
The second operand is placed at the first-operand location with the sign bit inverted. Both the first and second operands are each in a 64-bit floating-point register.

**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 1, AFP register
- Operation (if the floating-point-support enhancement facility is not installed)

## LOAD FPR FROM GR



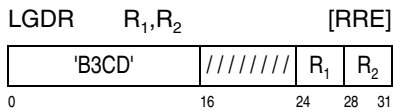
The second operand is placed at the first-operand location. The second operand is in a general register, and the first operand is in a floating-point register.

**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 1, AFP register
- Operation (if the floating-point-support enhancement facility is not installed)

## LOAD GR FROM FPR



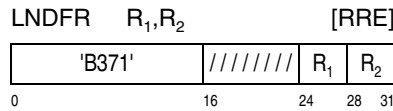
The second operand is placed at the first-operand location. The second operand is in a floating-point register, and the first operand is in a general register.

**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 1, AFP register
- Operation (if the floating-point-support enhancement facility is not installed)

## LOAD NEGATIVE



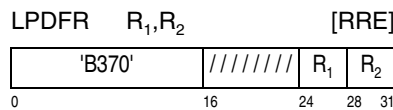
The second operand is placed at the first-operand location with the sign bit set to one. Both the first and second operands are each in a 64-bit floating-point register.

**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 1, AFP register
- Operation (if the floating-point-support enhancement facility is not installed)

## LOAD POSITIVE



The second operand is placed at the first-operand location with the sign bit set to zero. Both the first and second operands are each in a 64-bit floating-point register.

**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 1, AFP register
- Operation (if the floating-point-support enhancement facility is not installed)

⋮

# Chapter 19. Binary-Floating-Point Instructions

## Floating-Point-Control (FPC) Register

The floating-point-control (FPC) register is a 32-bit register that contains mask bits, flag bits, a data exception code, and rounding-mode bits. An overview of the FPC register is shown in Figure 19-1 on page 19-1. Details are shown in Figure 19-2 on page 19-1, Figure 19-3 on page 19-1, and Figure 19-4 on page 19-1. (In Figure 19-2, the abbreviations “IM” and “SF” are based on the terms “interruption mask” and “status flag”, respectively.)

The bits of the FPC register are often referred to as, for example, FPC 1.0, meaning bit 0 of byte 1 of the register.

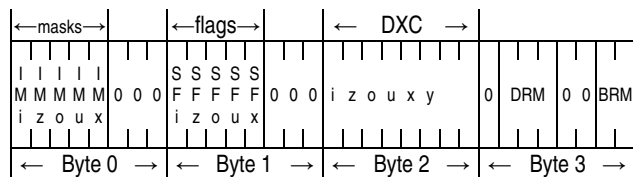


Figure 19-1. FPC Register Overview

Byte	Bit(s)	Name	Abbr.
0	0	IEEE-invalid-operation mask	IMi
0	1	IEEE-division-by-zero mask	IMz
0	2	IEEE-overflow mask	IMo
0	3	IEEE-underflow mask	IMu
0	4	IEEE-inexact mask	IMx
0	5-7	(Unassigned)	0
1	0	IEEE-invalid-operation flag	SFi
1	1	IEEE-division-by-zero flag	SFz
1	2	IEEE-overflow flag	SFo
1	3	IEEE-underflow flag	SFu
1	4	IEEE-inexact flag	SFx
1	5-7	(Unassigned)	0
2	0-7	Data-exception code	DXC
3	0	(Unassigned)	0
3	1-3	DFP rounding mode	DRM
3	4-5	(Unassigned)	0
3	6-7	BFP rounding mode	BRM

Figure 19-2. FPC-Register Bit Assignments

FPC Byte 3 Bits 1-3	DFP Rounding Mode
000	Round to nearest with ties to even
001	Round toward 0
010	Round toward $+\infty$
011	Round toward $-\infty$
100	Round to nearest with ties away from 0
101	Round to nearest with ties toward 0
110	Round away from 0
111	Round to prepare for shorter precision

Figure 19-3. DFP Rounding Mode

FPC Byte 3 Bits 6-7	BFP Rounding Mode
00	Round to nearest with ties to even
01	Round toward 0
10	Round toward $+\infty$
11	Round toward $-\infty$

Figure 19-4. BFP Rounding Mode

## IEEE Masks and Flags

The FPC register contains five IEEE mask bits and five IEEE flag bits that each correspond to one of the five arithmetic exception conditions that may occur when a BFP or DFP instruction is executed. The mask bits, when one, cause an interruption to occur if an exception condition is recognized. If the mask bit for an exception condition is zero, the recognition of the condition causes the corresponding flag bit to be set to one. Thus, a flag bit indicates whether the corresponding exception condition has been recognized at least once since the program last set the flag bit to zero. The mask bits are ignored, and the flag bits remain unchanged, when arithmetic exceptions are recognized for floating-point-support (FPS) and HFP instructions.

The IEEE flag bits in the FPC register are set to zero only by explicit program action, initial CPU reset, clear reset, or power-on reset.

## FPC DXC Byte

Byte 2 of the FPC register contains the data-exception code (DXC), which is an eight-bit code indicating the specific cause of a data exception. When the AFP-register-control bit, bit 45 of control register 0, is one and a program interruption causes the DXC to be placed at real location 147, the DXC is also placed in the DXC field of the FPC register. The DXC

field in the FPC register remains unchanged when the AFP-register-control bit is zero or when any other program exception is reported. The DXC is described in “Data-Exception Code (DXC)” on page 6-1.

The DXC is a code, meaning it should be treated as an integer rather than as individual bits. However, when bits 6 and 7 are zero, bits 0-5 are bit significant; bits 0-4 (i,z,o,u,x) are trap flags and correspond to the same bits in bytes 0 and 1 of the FPC register (IEEE masks and IEEE flags), and bit 5 (y) is used in conjunction with bit 4, inexact (x), to indicate that the result has been incremented in magnitude. The trap flag for an exception, instead of the IEEE flag, is set to one when an interruption for the exception is enabled by the corresponding IEEE mask bit.

## Operations on the FPC Register

The following unprivileged instructions allow problem-state programs to operate on the FPC register:

- EXTRACT FPC
- LOAD FPC
- LOAD FPC AND SIGNAL
- SET BFP ROUNDING MODE
- SET DFP ROUNDING MODE
- SET FPC
- SET FPC AND SIGNAL
- STORE FPC

These instructions are subject to the AFP-register-control bit, bit 45 of control register 0. When the AFP-register-control bit is zero, an attempt to execute the above instructions results in a data exception with DXC 2 or 3, depending on whether they are BFP or DFP instructions.

# Chapter 20. Decimal-Floating-Point Instructions

## DFP Number Representation

A DFP finite number consists of three components: a sign bit (S), a signed exponent (X), and a coefficient (C). The signed exponent is a signed binary integer. The coefficient consists of a number of decimal digits, which are to the left of the implied decimal point. The rightmost digit of the coefficient is called the units digit. The numerical value of a DFP finite number is represented as  $(-1)^{\text{sign}} \times \text{coefficient} \times 10^{\text{exponent}}$ , and the unit value of this number is  $(1 \times 10^{\text{exponent}})$ , which is called the quantum.

DFP finite numbers are not normalized. This allows leading zeros and trailing zeros to exist in the coefficient. This unnormalized DFP number representation allows some values to have redundant forms; each form represents the DFP number with a different combination of the coefficient value and the exponent value. For example,  $1000000 \times 10^5$  and  $10 \times 10^{10}$  are two different forms of the same numerical value. A form of this number representation carries information about both the numerical value and the quantum of a DFP finite number.

The significant digits of a DFP finite number are the digits in the coefficient beginning with the leftmost nonzero digit and ending with the units digit.

## DFP Data Formats

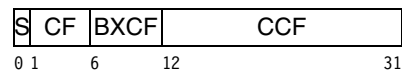
DFP numbers and NaNs may be represented in floating-point registers in any of three data formats: short, long, or extended. The contents of each data format represent encoded information. Special codes are assigned to NaNs and infinities.

The sign is encoded as a one bit binary value. The coefficient is encoded as an unsigned decimal integer in two distinct parts. The leftmost digit (LMD) of the coefficient is encoded as part of the combination field (CF); the remaining digits of the coefficient are encoded in the coefficient-continuation field (CCF). Similarly, the exponent is represented in two parts. However, prior to encoding, the exponent is converted to an unsigned binary value called the biased exponent by adding a bias value which is a constant for each format. The two leftmost bits of the biased exponent are encoded in the combination field and

the remaining 6, 8, or 12 bits (depending on the format) are encoded in the biased exponent continuation field (BXCF).

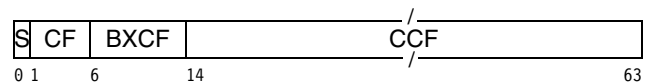
The decimal floating-point data representation comprises four fields, as diagrammed below for each of the three formats:

### Short Format



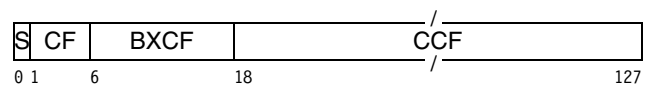
When a number or NaN in the DFP short format is loaded into a floating-point register, it occupies the left half of the register, and the right half remains unchanged.

### Long Format



When a number or NaN in the DFP long format is loaded into a floating-point register, it occupies the entire register.

### Extended Format



When a number or NaN in the DFP extended format is loaded into a floating-point register pair, the leftmost 64 bits occupy the entire left register of the pair and the rightmost 64 bits occupy the entire right register.

The fields are defined as follows:

### Sign (S)

The sign bit is in bit 0 of each format, and is zero for plus and one for minus.

### Combination Field (CF)

This is a 5-bit field which contains the encoding of NaN or infinity, or the two leftmost bits of the biased

exponent and the leftmost digit (LMD) of the coefficient. The following tables show the encoding:

Meaning	Combination Field (binary)
NaN	11111
Infinity	11110
Finite numbers	All others

Figure 20-1. Encoding of the Combination Field for Special Symbols

LMD	Leftmost two-bit value of biased exponent		
	0	1	2
0	00000	01000	10000
1	00001	01001	10001
2	00010	01010	10010
3	00011	01011	10011
4	00100	01100	10100
5	00101	01101	10101
6	00110	01110	10110
7	00111	01111	10111
8	11000	11010	11100
9	11001	11011	11101

Figure 20-2. Encoding of the Combination Field for Finite Numbers

## Biased Exponent-Continuation Field (BXCf)

For DFP finite numbers, this field contains the remaining bits of the biased exponent. For NaNs, the leftmost bit in this field is used to distinguish a Quiet NaN from a Signaling NaN; the remaining bits in a source operand are ignored and they are set to zeros in a target operand by most operations. For infinities, bits in this field of a source operand are ignored and they are set to zeros in a target operand by most operations.

## Coefficient-Continuation Field (CCF)

For DFP finite numbers, this field contains the remaining coefficient digits. For NaNs, this field may be used to contain diagnostic information. For infinities, contents in this field of a source operand are ignored and they are set to zeros in a target operand by most operations. The coefficient-continuation field is a multiple of 10-bit blocks. The multiple depends on the format. Each 10-bit block is called a DPD block and represents three decimal digits, using the Densely Packed Decimal (DPD) encoding defined in the section “Densely Packed Decimal” on page 20-9.

The properties of the three DFP formats are summarized in Figure 20-3 on page 20-2:

Property	Format		
	Short	Long	Extended
Format length (bits)	32	64	128
Combination length (bits)	5	5	5
Biased exponent continuation length (bits)	6	8	12
Maximum biased exponent	191	767	12,287
Coefficient continuation length (bits)	20	50	110
Precision (digits), p	7	16	34
Maximum exponent, $X_{max}$	90	369	6111
Minimum exponent, $X_{min}$	-101	-398	-6176
Exponent bias	101	398	6176
Largest (in magnitude) normal number, $N_{max}$	$(10^7 - 1) \times 10^{90}$	$(10^{16} - 1) \times 10^{369}$	$(10^{34} - 1) \times 10^{6111}$
Smallest (in magnitude) normal number, $N_{min}$	$1 \times 10^{-95}$	$1 \times 10^{-383}$	$1 \times 10^{-6143}$
Smallest (in magnitude) subnormal number, $D_{min}$	$1 \times 10^{-101}$	$1 \times 10^{-398}$	$1 \times 10^{-6176}$

Figure 20-3. Summary of DFP Formats

## Preferred DPD Encoding

Execution of some DFP instructions decodes source operands from DFP data formats to an internal format for processing, and encodes the operation's result before the final result is placed at the target operand location.

As part of the decoding process, DPD blocks in the coefficient-continuation field of the source operands are decoded to their corresponding BCD<sup>1</sup> digit codes using the DPD-to-BCD decoding algorithm. As part of the encoding process, BCD digit codes to be stored into the coefficient-continuation field of the target operand are encoded into DPD blocks using the

BCD-to-DPD encoding algorithm. Both the decoding and encoding algorithms are defined in the section “Densely Packed Decimal” on page 20-9.

As explained in the section “Densely Packed Decimal”, there are eight 3-digit decimal values that have redundant DPD codes and one preferred DPD code. All redundant DPD codes are recognized in source operands for the associated 3-digit decimal number. Only the preferred DPD code is generated by DFP operations for the coefficient-continuation field of the target operand.

## Classes of DFP Data

There are six classes of DFP data, which include numerical and non numerical entities. The numerical entities include zero, subnormal number, normal number, and infinity data classes. The non numerical entities include quiet and signaling NaN data classes. The value of a DFP finite number, including zero, subnormal number, and normal number, is a quantization of the real number based on the data format. The TEST DATA CLASS instruction may be used to determine the class of a DFP operand.

The following tables show the value ranges for finite-number data classes, and the codes for NaNs and infinity.

Data Class	Sign	Magnitude
Zero	±	0*
Subnormal	±	$D_{\min} \leq  X  < N_{\min}$
Normal	±	$N_{\min} \leq  Y  \leq N_{\max}$
<b>Explanation:</b>		
* The coefficient is zero and the exponent is any representable value		

Figure 20-4. Value Ranges for Finite Number Data Classes

### Zeros

Zeros have a zero coefficient and any representable value in the exponent. A +0 is distinct from a -0, and zeros with different exponents are distinct, except that comparison treats them as equal.

### Subnormal Numbers

Subnormal numbers have values that are smaller than  $N_{\min}$  and greater than zero in magnitude.

1. A BCD number consists of digit codes and, if signed, a sign code (see figure 8-1 on page 8-2 of the *z/Architecture Principles of Operation*).

Data Class	Sign	Combination (bits)	Biased Exponent Continuation (bits)	Coefficient Continuation (bits)
Infinity	±	11110	xxx ... xxx	xxx ... xxx
Quiet NaN	±	11111	0xx ... xxx	xxx ... xxx
Signaling NaN	±	11111	1xx ... xxx	xxx ... xxx
<b>Explanation:</b>				
x Don't care				

Figure 20-5. Encoding of NaNs and Infinity Data Classes

### Normal Numbers

A normal number is a nonzero finite number whose magnitude is between  $N_{\min}$  and  $N_{\max}$  inclusively.

### Infinities

An infinity is represented by a combination field of 11110 binary. When an operation is defined to generate an infinity as the result, a default infinity is sometimes supplied. A default infinity has all bits in the biased-exponent-continuation and coefficient-continuation fields set to zeros.

When used as a source operand, the contents of the biased-exponent-continuation and the coefficient-continuation fields of an infinity are usually ignored. In all cases, the biased-exponent-continuation field of a generated infinity contains all zeros.

Infinities can participate in most arithmetic operations and give a consistent result. In comparisons, any  $+\infty$  compares greater than any finite number, and any  $-\infty$  compares less than any finite number. All  $+\infty$  are compared equal and all  $-\infty$  are compared equal.

### Signaling and Quiet NaNs

A NaN (Not-a-Number) is represented by a combination field of all ones. There are two types of NaNs, Signaling and Quiet. A Signaling NaN (SNaN) is distinguished from a Quiet NaN (QNaN) by the leftmost bit in the biased-exponent-continuation field: 0 for the QNaN and 1 for the SNaN. A special QNaN is sometimes supplied as the default QNaN for a disabled IEEE invalid-operation exception; it has a plus sign, and all bits in the biased-exponent-continuation field and the coefficient-continuation field are set to zeros.

Normally, source QNaNs are propagated during operations so that they will remain visible at the end.

When a QNaN is propagated, the sign is preserved, the decimal value of the coefficient-continuation field is preserved but is re-encoded using the preferred DPD codes, and the contents in the biased-exponent-continuation field are set to zero.

A source SNaN generally causes an IEEE invalid-operation exception. If the exception is disabled, the SNaN is converted to a corresponding QNaN which is propagated. The differences between an SNaN and the corresponding QNaN are that the contents of the biased-exponent-continuation field of the QNaN are set to zero and the same decimal value of the coefficient-continuation field is re-encoded using the preferred DPD codes in the QNaN. For some special cases of the format-conversion instructions, a source SNaN does not cause an IEEE invalid-operation exception, and an SNaN is returned as the target operand.

## DFP Rounding

DFP operations are performed as if they first produce an intermediate result correct to infinite precision and with unbounded range. The intermediate result is then rounded to the destination's precision according to one of the eight DFP rounding modes. If the rounded result has only one form, it is delivered as the final result; if the rounded result has redundant forms, then an ideal exponent is used to select the form of the final result. Most operations which produce a DFP result have an ideal exponent defined. The ideal exponent of an operation determines the form, not the value, of the final result. (See "Formation of Final Result" on page 20-5.)

Rounding takes a number regarded as infinitely precise and, if necessary, modifies it to fit the destination's precision. The destination's precision of an operation defines the set of permissible resultant values. For most operations, the destination precision is the target-format precision and the permissible resultant values are those values representable in the target format. For some special operations, the destination precision is constrained by both the target format and some additional restrictions, and the permissible resultant values are a subset of all of the values representable in the target format.

Rounding may cause an overflow exception or underflow exception; it may also cause an inexact exception.

## Rounding Mode

There are eight rounding modes. The current DFP rounding mode is specified by the value of three rounding-mode bits in the FPC register, as follows:

**000 Round to nearest ties to even.** In this mode, the permissible resultant value nearest to the infinitely precise result is delivered. If the two nearest permissible resultant values are equally near, the delivered value is the one whose units digit would have been even in the form with the largest common quantum of the two permissible resultant values. However, an infinitely precise result with magnitude at least  $(N_{\max} + 0.5Q(N_{\max}))$  is rounded to infinity with no change in sign; where  $Q(N_{\max})$  is the quantum of  $N_{\max}$ .

**001 Round toward 0.** In this mode, the permissible resultant value delivered is the one closest to and no greater in magnitude than the infinitely precise result.

**010 Round toward  $+\infty$ .** In this mode, the permissible resultant value delivered is the one (possibly  $+\infty$ ) closest to and no less than the infinitely precise result.

**011 Round toward  $-\infty$ .** In this mode, the permissible resultant value delivered is the one (possibly  $-\infty$ ) closest to and no greater than the infinitely precise result.

**100 Round to nearest ties away from 0.** In this mode, the permissible resultant value nearest to the infinitely precise result is delivered; if the two nearest permissible resultant values are equally near, the one that is greater in magnitude than the infinitely precise result is delivered. However, an infinitely precise result with magnitude at least  $(N_{\max} + 0.5Q(N_{\max}))$  is rounded to infinity with no change in sign; where  $Q(N_{\max})$  is the quantum of  $N_{\max}$ .

**101 Round to nearest ties toward 0.** In this mode, the permissible resultant value nearest to the infinitely precise result is delivered; if the two nearest permissible resultant values are equally near, the one that is less in magnitude than the infinitely precise result is delivered. However, an infinitely precise result with magnitude greater than  $(N_{\max} + 0.5Q(N_{\max}))$  is rounded to infinity with no change in sign; where  $Q(N_{\max})$  is the quantum of  $N_{\max}$ .

**110 Round away from 0.** In this mode, the permissible resultant value delivered is the one closest to and no less in magnitude than the infinitely precise result.

**111 Round to prepare for shorter precision.** In this mode, the permissible resultant value that is closest to and no greater in magnitude than the infinitely precise result is selected. If the selected value is not exact and the units digit of the selected value is either 0 or 5, then the digit is incremented by one and the incremented result is delivered. In all other cases, the selected value is delivered. When a value has redundant forms, the units digit is determined by using the form that has the smallest quantum.

## Formation of Final Result

An ideal exponent is defined for each DFP instruction that returns a DFP data operand.

### Use of Ideal Exponent

For all DFP operations, if the rounded intermediate result has only one form, then that form is delivered as the final result. If the rounded intermediate result has redundant forms and is exact, then the form with the exponent closest to the ideal exponent is delivered. If the rounded intermediate result has redundant forms and is inexact, then the form with the smallest exponent is delivered.

### Summary of Ideal Exponent

Figure 20-6 on page 20-5 summarizes all ideal exponents.

Operations	Ideal Exponent
ADD	Lesser exponent value of the two source operands
CONVERT FROM FIXED	Zero
CONVERT FROM SIGNED BCD	Zero
CONVERT FROM UNSIGNED BCD	Zero
DIVIDE	The exponent of the divisor subtracted from the exponent of the dividend
INSERT BIASED EXPONENT	The specified target exponent
LOAD FP INTEGER	Greater value of zero and the exponent of the second operand
LOAD LENGTHENED	The exponent of the source operand
LOAD ROUNDED	The exponent of the source operand
MULTIPLY	Sum of the exponents of the multiplier and multiplicand
QUANTIZE	The specified target exponent
REROUND	Greater value of the referenced exponent and the third operand's exponent
SUBTRACT	Lesser exponent value of the two source operands

Figure 20-6. Summary of Ideal Exponents

## IEEE Comparison

For finite numbers, comparisons are performed on values, that is, all forms of a value are treated equal.

Comparisons are always exact and cannot cause an IEEE-inexact condition.

Comparison ignores the sign of zero, that is, +0 equals -0.

Infinities with like sign compare equal, that is,  $+\infty$  equals  $+\infty$ , and  $-\infty$  equals  $-\infty$ .

A NaN compares as unordered with any other operand, whether a finite number, an infinity, or another NaN, including itself.

Two sets of instructions are provided: COMPARE and COMPARE AND SIGNAL. In the absence of QNaNs, these instructions work the same. These instructions work differently only when both of the following are true:

- Neither operand of the instruction is an SNaN
- At least one operand of the instruction is a QNaN

In this case, COMPARE simply sets condition code 3, but COMPARE AND SIGNAL recognizes the IEEE-invalid-operation condition. If any operand is an

SNaN, both instructions recognize the IEEE-invalid-operation condition.

The action when the IEEE-invalid-operation condition is recognized depends on the IEEE-invalid-operation mask bit in the FPC register. If the mask bit is zero, then the instruction execution is completed by setting condition code 3, and the IEEE-invalid-operation flag in the FPC register is set to one. If the mask bit is one, then the condition is reported as a program interruption for a data exception with DXC 80 hex (IEEE invalid operation).

## DFP-Format Conversion

The instructions LOAD LENGTHENED and LOAD ROUNDED perform conversions of numbers between the short, long, and extended formats.

When converting a finite number to a wider format, the result is exact. When converting a finite number to a narrower format, the source operand is rounded to the target-format precision, which is specified by the instruction, not by the target register size.

When converting a finite number, the ideal exponent of the result is the source exponent.

Conversion of an infinity or NaN to a different format does not preserve the source biased-exponent-continuation field. When the result is an infinity or QNaN, the target biased-exponent-continuation field is set to all zeros. When the result is an SNaN, the leftmost bit in the target biased-exponent-continuation field is set to one and all other bits are set to zeros.

When converting an infinity and the suppression of invalid-operation indication is one, a propagated infinity is produced. Digits in the source coefficient-continuation field are re-encoded using the preferred DPD codes with sufficient zeros appended to or removed from the left to form the target coefficient-continuation field.

When converting an infinity and the suppression of invalid-operation indication is zero, a default infinity with the same sign is produced.

When converting a QNaN to a wider format, digits in the source coefficient-continuation field are re-encoded using the preferred DPD codes with sufficient zeros appended on the left to form the target coefficient-continuation field. When converting a

QNaN to a narrower format, the appropriate number of leftmost digits of the source coefficient-continuation field are removed and the remaining digits of the field are re-encoded using the preferred DPD codes to form the target coefficient-continuation field.

When converting an SNaN and the suppression of invalid-operation indication is zero, the invalid-operation exception occurs; if the invalid-operation exception is disabled, the result is converted to the corresponding QNaN and the appropriate number of leftmost digits of the source coefficient-continuation field are removed or padded with zeros to form the target coefficient-continuation field. When converting an SNaN and the suppression of invalid-operation indication is one, it is converted to an SNaN without causing an invalid-operation exception or setting the invalid-op status flag.

## IEEE Exception Conditions

The results of each of the IEEE exception conditions are controlled by a mask bit in the FPC register. When an IEEE exception condition is recognized, one of two actions is taken:

- If the corresponding mask bit in the FPC register is zero, a default action is taken, as specified for each condition, and the corresponding flag bit in the FPC register is set to one. Program execution then continues normally.
- If the corresponding mask bit in the FPC register is one, a program interruption for a data exception occurs, the operation is suppressed or completed, depending on the condition, and the data-exception code (DXC) assigned for that condition is provided.

## IEEE Invalid Operation

An IEEE-invalid-operation condition is recognized when, in the execution of a DFP instruction, any of the following occurs:

1. An SNaN is encountered in any DFP operation except for TEST DATA CLASS, TEST DATA GROUP, COMPARE EXPONENT, EXTRACT SIGNIFICANCE, CONVERT TO SIGNED BCD, CONVERT TO UNSIGNED BCD, EXTRACT BIASED EXPONENT, INSERT BIASED EXPONENT, LOAD ROUNDED (with suppression of invalid-operation indication), LOAD LENGTHENED (with suppression of invalid-operation indi-

cation), SHIFT COEFFICIENT LEFT, and SHIFT COEFFICIENT RIGHT.

2. A QNaN is encountered in a comparison by COMPARE AND SIGNAL.
3. A difference is undefined (addition of infinities of opposite sign, or subtraction of infinities of like sign).
4. A product is undefined (zero times infinity).
5. A quotient is undefined (DIVIDE instruction with both source operands zero or both source operands infinity).
6. The QUANTIZE operation detects that the coefficient associated with the specified target exponent would have more significant digits than the target-format precision.
7. For the QUANTIZE operation, when one source operand specifies an infinity and another source operand specifies a finite number.
8. The REROUND operation detects that the target precision's largest exponent ( $X_{max}$ ) is exceeded by what would have been the exponent of the rounded result were the range unbounded.
9. The CONVERT TO FIXED operation involving a number greater in magnitude to be represented in the target format, or involving a NaN.

If the IEEE-invalid-operation mask bit in the FPC register is zero, the IEEE-invalid-operation flag bit in the FPC register is set to one. The completion of the operation depends on the type of operation and the operands.

If the instruction performs a comparison and no program interruption occurs, the comparison result is unordered.

If the instruction is one that produces a DFP result, if no program interruption occurs, and if none of the operands is a NaN, the result is the default QNaN. If one of the operands is a NaN, then a propagated QNaN is returned.

If the IEEE-invalid-operation mask bit in the FPC register is one, the operation is suppressed, and the condition is reported as a program interruption for a data exception with DXC 80 hex.

### IEEE Division-By-Zero

An IEEE-division-by-zero condition is recognized when in DFP division the divisor is zero and the dividend is a finite nonzero number.

If the IEEE-division-by-zero mask bit in the FPC register is zero, the IEEE-division-by-zero flag bit in the FPC register is set to one. The operation is completed using as the result an infinity with a sign that is the exclusive or of the dividend and divisor signs.

If the IEEE-division-by-zero mask bit in the FPC register is one, the operation is suppressed, and the condition is reported as a program interruption for a data exception with DXC 40 hex.

### IEEE Overflow

Except for REROUND, the following describes the handling of the IEEE overflow exception condition. The REROUND operation does not recognize an overflow exception condition.

An IEEE-overflow condition is recognized whenever the target precision's largest finite number ( $N_{max}$ ) is exceeded in magnitude by what would have been the rounded floating-point result were the range unbounded.

If the IEEE-overflow mask bit in the FPC register is zero, the IEEE-overflow flag bit in the FPC register is set to one. The result of the operation depends on the sign of the intermediate result and on the current DFP rounding mode as described in Figure 20-7:

Rounding Mode	Sign of the Intermediate Result is	
	Plus	Minus
Round to nearest with ties to even	$+\infty$	$-\infty$
Round toward 0	$+N_{max}$	$-N_{max}$
Round toward $+\infty$	$+\infty$	$-N_{max}$
Round toward $-\infty$	$+N_{max}$	$-\infty$
Round to nearest with ties away from 0	$+\infty$	$-\infty$
Round to nearest with ties toward 0	$+\infty$	$-\infty$
Round away from 0	$+\infty$	$-\infty$
Round to prepare for shorter precision	$+N_{max}$	$-N_{max}$

Figure 20-7. Overflow Results When Exception is Disabled

If the IEEE-overflow mask bit in the FPC register is one, the operation is completed by producing a wrapped rounded result, and the condition is

reported as a program interruption for a data exception with DXC 20, 28, or 2C hex, depending on whether the delivered result is exact, inexact and truncated, or inexact and incremented, respectively.

The delivered result is derived as follows:

1. The infinitely precise result is divided by  $10^a$ . That is, the exponent adjustment ( $a$ ) is subtracted from the exponent. This is called the wrapped result. The exponent adjustment for all operations, except LOAD ROUNDED, is 576 for long and 9216 for extended. For LOAD ROUNDED, the exponent adjustment depends on the source format, and is 192 for long and 3072 for extended.
2. The wrapped result is rounded to the target-format precision. This is the wrapped rounded result.
3. If the wrapped rounded result has only one form, it is the delivered result. If the wrapped rounded result has redundant forms and is exact, the result of the form that has the exponent closest to the wrapped ideal exponent is returned. If the wrapped rounded result has redundant forms and is inexact, the result of the form that has the smallest exponent is returned. The wrapped ideal exponent is the result of subtracting the exponent adjustment from the ideal exponent.

### IEEE Underflow

Except for REROUND, the following describes the handling of the IEEE underflow exception condition. The REROUND operation does not recognize an underflow exception condition.

A tininess condition is recognized when the intermediate result with both the precision and range unbounded would be nonzero and less than the target precision's smallest normal number,  $N_{\min}$ , in magnitude.

If the IEEE-underflow mask bit in the FPC register is zero, then the action depends on whether the result can be represented exactly and, if not, also on the setting of the IEEE-inexact mask bit in the FPC register.

If the result can be represented exactly, the exact value is given. If the result cannot be represented exactly and the IEEE-inexact mask bit in the FPC

register is zero, the rounded result is given, and the IEEE-underflow and IEEE-inexact flag bits in the FPC register are set to ones. In this case, the infinitely precise result is rounded to the target-format precision. If the rounded result has only one form, it is the delivered result. If the rounded result has redundant forms, the result of the form that is closest to the ideal exponent is given.

If the result cannot be represented exactly and the IEEE-inexact mask bit in the FPC register is one, the IEEE-underflow flag bit in the FPC register is set to one, and the inexact condition is reported as a program interruption for a data exception with DXC 08 or 0C hex, depending on whether the result is inexact and truncated or inexact and incremented, respectively.

If the IEEE-underflow mask bit in the FPC register is one, then, regardless of whether the result could have been represented exactly, the operation is completed by producing a wrapped rounded result, and the condition is reported as a program interruption for a data exception with DXC 10, 18, or 1C hex, depending on whether the delivered result is exact, inexact and truncated, or inexact and incremented, respectively.

The delivered result is derived as follows:

1. The infinitely precise result is multiplied by  $10^a$ . That is, the exponent adjustment ( $a$ ) is added to the exponent. This is called the wrapped result. The exponent adjustment for all operations, except LOAD ROUNDED, is 576 for long and 9216 for extended. For LOAD ROUNDED, the exponent adjustment depends on the source format, and is 192 for long and 3072 for extended.
2. The wrapped result is rounded to the target-format precision. This is the wrapped rounded result.
3. If the wrapped rounded result has only one form, it is the delivered result. If the wrapped rounded result has redundant forms and is exact, the result of the form that has the exponent closest to the wrapped ideal exponent is returned. If the wrapped rounded result has redundant forms and is inexact, the result of the form that has the smallest exponent is returned. The wrapped ideal exponent is the result of adding the exponent adjustment to the ideal exponent.

## IEEE Inexact

Except for LOAD FP INTEGER (with suppression of inexact indication), the following describes the handling of the IEEE inexact exception condition. The LOAD FP INTEGER (with suppression of inexact indication) operation does not recognize an inexact exception condition.

An IEEE-inexact condition is recognized when the delivered result differs in value from what would have been computed were both the precision and exponent range unbounded. The condition is also recognized if rounding the result causes IEEE overflow and the IEEE-overflow mask bit is zero. The operation is completed using the rounded result or, in case of overflow or underflow, the result specified for IEEE overflow or IEEE underflow.

If the IEEE-inexact mask bit in the FPC register is zero, the IEEE-inexact flag bit in the FPC register is set to one. If the IEEE-inexact mask bit in the FPC register is one, the operation is completed, and the condition is reported as a program interruption for a data exception with DXC 08 or 0C hex, depending on whether the result is inexact and truncated or inexact and incremented, respectively.

## Densely Packed Decimal

The coefficient continuation field of the decimal floating-point data format is encoded using Densely Packed Decimal (DPD). DPD encoding is a compression technique which supports the representation of decimal integers of arbitrary length. Translation operates on three Binary Coded Decimal (BCD) digits at a time compressing the 12 bits into 10 bits with an algorithm that can be applied or reversed using simple Boolean operations. In the following examples, a 3-digit BCD number is represented as (abcd)(efgh)(ijklm), a 10-bit DPD number is represented as (pqr)(stu)(v)(wxy), and the Boolean operations, & (AND), | (OR), and ~ (NOT) are used.

## BCD-to-DPD Translation

The translation from a 3-digit BCD number to a 10-bit DPD can be performed through the following Boolean operations.

$$p = (a \& f \& i) | (a \& j) | b$$

$$q = (a \& g \& i) | (a \& k) | c$$

$$r = d$$

$$s = (\sim a \& e \& j) | (f \& \sim i) | (\sim a \& f) | (e \& i)$$

$$t = (\sim a \& e \& k) | (a \& i) | g$$

$$u = h$$

$$v = a | e | i$$

$$w = (\sim e \& j) | (e \& i) | a$$

$$x = (\sim a \& k) | (a \& i) | e$$

$$y = m$$

Alternatively, the following table can be used to perform the translation. The most significant bit of the three BCD digits (left column) is used to select a specific 10-bit encoding (right column) of the DPD.

aei	pqr stu v wxy
000	bcd fgh 0 jkm
001	bcd fgh 1 00m
010	bcd jkh 1 01m
011	bcd 10h 1 11m
100	jdk fgh 1 10m
101	fgd 01h 1 11m
110	jdk 00h 1 11m
111	00d 11h 1 11m

The full translation of a 3-digit BCD number (000 - 999) to a 10-bit DPD is shown in Figure 20-8 on page 20-11, with the DPD entries shown in hexadecimal format. The BCD number is produced by replacing ‘\_’ in the leftmost column with the corresponding digit along the top row. The table is split into two halves, with the right half being a continuation of the left half.

## DPD-to-BCD Translation

The translation from a 10-bit DPD to a 3-digit BCD number can be performed through the following Boolean operations.

$$a = (\sim s \& v \& w) | (t \& v \& w \& x) | (v \& w \& \sim x)$$

$$b = (p \& s \& x) | (p \& \sim w) | (p \& \sim v)$$

$$c = (q \& s \& x) | (q \& \sim w) | (q \& \sim v)$$

$$d = r$$

$$e = (t \& v \& \sim w \& x) | (s \& v \& w \& x) | (\sim t \& v \& x)$$

$$f = (p \& t \& v \& w \& x) | (s \& \sim x) | (s \& \sim v)$$

$$g = (q \& t \& w) | (t \& \sim x) | (t \& \sim v)$$

$$h = u$$

$$i = (t \& v \& w \& x) | (s \& v \& w \& x) | (v \& \sim w \& \sim x)$$

$$j = (p \& \sim s \& \sim t \& w) | (s \& v \& \sim w \& x) | (p \& w \& \sim x) | (\sim v \& w)$$

$$k = (q \& \sim s \& \sim t \& v \& w) | (q \& v \& w \& \sim x) | (t \& v \& \sim w \& x) | (\sim v \& x)$$

$$m = y$$

Alternatively, the following table can be used to perform the translation. A combination of five bits in the DPD encoding (leftmost column) are used to specify a translation to the 3-digit BCD encoding. Dashes (-) in the table are don't cares, and can be either one or zero.

<b>vwkst</b>	<b>abcd</b>	<b>efgh</b>	<b>ijklm</b>
0---	Opqr	Ostu	Owxy
100--	Opqr	Ostu	100y
101--	Opqr	100u	0sty
110--	100r	Ostu	0pqy
11100	100r	100u	0pqy
11101	100r	0pqu	100y
11110	0pqr	100u	100y
11111	100r	100u	100y

The full translation of the 10-bit DPD to a 3-digit BCD number is shown in Figure 20-9 on page 20-12. The 10-bit DPD index is produced by concatenating the 6-bit value shown in the left column with the 4-bit index along the top row, both represented in hexadecimal. The values in parentheses are non-preferred translations and are explained further in the following section.

### Preferred DPD encoding

Translating from a 3-digit BCD number (1000 numbers) to a 10-bit DPD encoding (1024 combinations) leaves 24 redundant translations. The 24 redundant combinations are evenly assigned to eight BCD numbers and are shown in the following table, with the non-preferred encoding in parentheses. The preferred encoding is produced by translating a 3-digit BCD number with the translation table or Boolean operations shown in "BCD-to-DPD Translation" on page 20-9. The redundant DPD encodings are all valid and will be correctly translated to their respec-

tive BCD value through the mechanisms provided in "DPD-to-BCD Translation" on page 20-9. For decimal floating point operations all DPD encodings are recognized as source operands.

<b>DPD Code</b>	<b>BCD Value</b>	<b>DPD Code</b>	<b>BCD Value</b>
0x06E	888	0x0EE	988
(0x16E)		(0x1EE)	
(0x26E)		(0x2EE)	
(0x36E)		(0x3EE)	
0x06F	889	0x0EF	989
(0x16F)		(0x1EF)	
(0x26F)		(0x2EF)	
(0x36F)		(0x3EF)	
0x07E	898	0x0FE	998
(0x17E)		(0x1FE)	
(0x27E)		(0x2FE)	
(0x37E)		(0x3FE)	
0x07F	899	0x0FF	999
(0x17F)		(0x1FF)	
(0x27F)		(0x2FF)	
(0x37F)		(0x3FF)	

### Summary of Rounding And Range Actions

Figure 20-10 on page 20-13 and page 20-14 summarizes rounding and range actions.

The exceptions to this summary are:

1. The REROUND operation does not recognize an underflow or an overflow exception condition.
2. The LOAD FP INTEGER (with suppression of inexact indication) operation does not recognize an inexact exception condition.

	0	1	2	3	4	5	6	7	8	9
00_	000	001	002	003	004	005	006	007	008	009
01_	010	011	012	013	014	015	016	017	018	019
02_	020	021	022	023	024	025	026	027	028	029
03_	030	031	032	033	034	035	036	037	038	039
04_	040	041	042	043	044	045	046	047	048	049
05_	050	051	052	053	054	055	056	057	058	059
06_	060	061	062	063	064	065	066	067	068	069
07_	070	071	072	073	074	075	076	077	078	079
08_	00A	00B	02A	02B	04A	04B	06A	06B	04E	04F
09_	01A	01B	03A	03B	05A	05B	07A	07B	05E	05F
10_	080	081	082	083	084	085	086	087	088	089
11_	090	091	092	093	094	095	096	097	098	099
12_	0A0	0A1	0A2	0A3	0A4	0A5	0A6	0A7	0A8	0A9
13_	0B0	0B1	0B2	0B3	0B4	0B5	0B6	0B7	0B8	0B9
14_	0C0	0C1	0C2	0C3	0C4	0C5	0C6	0C7	0C8	0C9
15_	0D0	0D1	0D2	0D3	0D4	0D5	0D6	0D7	0D8	0D9
16_	0E0	0E1	0E2	0E3	0E4	0E5	0E6	0E7	0E8	0E9
17_	0F0	0F1	0F2	0F3	0F4	0F5	0F6	0F7	0F8	0F9
18_	08A	08B	0AA	0AB	0CA	0CB	0EA	0EB	0CE	0CF
19_	09A	09B	0BA	0BB	0DA	0DB	0FA	0FB	0DE	0DF
20_	100	101	102	103	104	105	106	107	108	109
21_	110	111	112	113	114	115	116	117	118	119
22_	120	121	122	123	124	125	126	127	128	129
23_	130	131	132	133	134	135	136	137	138	139
24_	140	141	142	143	144	145	146	147	148	149
25_	150	151	152	153	154	155	156	157	158	159
26_	160	161	162	163	164	165	166	167	168	169
27_	170	171	172	173	174	175	176	177	178	179
28_	10A	10B	12A	12B	14A	14B	16A	16B	14E	14F
29_	11A	11B	13A	13B	15A	15B	17A	17B	15E	15F
30_	180	181	182	183	184	185	186	187	188	189
31_	190	191	192	193	194	195	196	197	198	199
32_	1A0	1A1	1A2	1A3	1A4	1A5	1A6	1A7	1A8	1A9
33_	1B0	1B1	1B2	1B3	1B4	1B5	1B6	1B7	1B8	1B9
34_	1C0	1C1	1C2	1C3	1C4	1C5	1C6	1C7	1C8	1C9
35_	1D0	1D1	1D2	1D3	1D4	1D5	1D6	1D7	1D8	1D9
36_	1E0	1E1	1E2	1E3	1E4	1E5	1E6	1E7	1E8	1E9
37_	1F0	1F1	1F2	1F3	1F4	1F5	1F6	1F7	1F8	1F9
38_	18A	18B	1AA	1AB	1CA	1CB	1EA	1EB	1CE	1CF
39_	19A	19B	1BA	1BB	1DA	1DB	1FA	1FB	1DE	1DF
40_	200	201	202	203	204	205	206	207	208	209
41_	210	211	212	213	214	215	216	217	218	219
42_	220	221	222	223	224	225	226	227	228	229
43_	230	231	232	233	234	235	236	237	238	239
44_	240	241	242	243	244	245	246	247	248	249
45_	250	251	252	253	254	255	256	257	258	259
46_	260	261	262	263	264	265	266	267	268	269
47_	270	271	272	273	274	275	276	277	278	279
48_	20A	20B	22A	22B	24A	24B	26A	26B	24E	24F
49_	21A	21B	23A	23B	25A	25B	27A	27B	25E	25F

	0	1	2	3	4	5	6	7	8	9
50_	280	281	282	283	284	285	286	287	288	289
51_	290	291	292	293	294	295	296	297	298	299
52_	2A0	2A1	2A2	2A3	2A4	2A5	2A6	2A7	2A8	2A9
53_	2B0	2B1	2B2	2B3	2B4	2B5	2B6	2B7	2B8	2B9
54_	2C0	2C1	2C2	2C3	2C4	2C5	2C6	2C7	2C8	2C9
55_	2D0	2D1	2D2	2D3	2D4	2D5	2D6	2D7	2D8	2D9
56_	2E0	2E1	2E2	2E3	2E4	2E5	2E6	2E7	2E8	2E9
57_	2F0	2F1	2F2	2F3	2F4	2F5	2F6	2F7	2F8	2F9
58_	28A	28B	2AA	2AB	2CA	2CB	2EA	2EB	2CE	2CF
59_	29A	29B	2BA	2BB	2DA	2DB	2FA	2FB	2DE	2DF
60_	300	301	302	303	304	305	306	307	308	309
61_	310	311	312	313	314	315	316	317	318	319
62_	320	321	322	323	324	325	326	327	328	329
63_	330	331	332	333	334	335	336	337	338	339
64_	340	341	342	343	344	345	346	347	348	349
65_	350	351	352	353	354	355	356	357	358	359
66_	360	361	362	363	364	365	366	367	368	369
67_	370	371	372	373	374	375	376	377	378	379
68_	30A	30B	32A	32B	34A	34B	36A	36B	34E	34F
69_	31A	31B	33A	33B	35A	35B	37A	37B	35E	35F
70_	380	381	382	383	384	385	386	387	388	389
71_	390	391	392	393	394	395	396	397	398	399
72_	3A0	3A1	3A2	3A3	3A4	3A5	3A6	3A7	3A8	3A9
73_	3B0	3B1	3B2	3B3	3B4	3B5	3B6	3B7	3B8	3B9
74_	3C0	3C1	3C2	3C3	3C4	3C5	3C6	3C7	3C8	3C9
75_	3D0	3D1	3D2	3D3	3D4	3D5	3D6	3D7	3D8	3D9
76_	3E0	3E1	3E2	3E3	3E4	3E5	3E6	3E7	3E8	3E9
77_	3F0	3F1	3F2	3F3	3F4	3F5	3F6	3F7	3F8	3F9
78_	38A	38B	3AA	3AB	3CA	3CB	3EA	3EB	3CE	3CF
79_	39A	39B	3BA	3BB	3DA	3DB	3FA	3FB	3DE	3DF
80_	00C	00D	10C	10D	20C	20D	30C	30D	02E	02F
81_	01C	01D	11C	11D	21C	21D	31C	31D	03E	03F
82_	02C	02D	12C	12D	22C	22D	32C	32D	12E	12F
83_	03C	03D	13C	13D	23C	23D	33C	33D	13E	13F
84_	04C	04D	14C	14D	24C	24D	34C	34D	22E	22F
85_	05C	05D	15C	15D	25C	25D	35C	35D	23E	23F
86_	06C	06D	16C	16D	26C	26D	36C	36D	32E	32F
87_	07C	07D	17C	17D	27C	27D	37C	37D	33E	33F
88_	00E	00F	10E	10F	20E	20F	30E	30F	06E	06F
89_	01E	01F	11E	11F	21E	21F	31E	31F	07E	07F
90_	08C	08D	18C	18D	28C	28D	38C	38D	0AE	0AF
91_	09C	09D	19C	19D	29C	29D	39C	39D	0BE	0BF
92_	0AC	0AD	1AC	1AD	2AC	2AD	3AC	3AD	1AE	1AF
93_	0BC	0BD	1BC	1BD	2BC	2BD	3BC	3BD	1BE	1BF
94_	0CC	0CD	1CC	1CD	2CC	2CD	3CC	3CD	2AE	2AF
95_	0DC	0DD	1DC	1DD	2DC	2DD	3DC	3DD	2BE	2BF
96_	0EC	0ED	1EC	1ED	2EC	2ED	3EC	3ED	3AE	3AF
97_	0FC	0FD	1FC	1FD	2FC	2FD	3FC	3FD	3BE	3BF
98_	08E	08F	18E	18F	28E	28F	38E	38F	0EE	0EF
99_	09E	09F	19E	19F	29E	29F	39E	39F	0FE	0FF

Figure 20-8. BCD to DPD Translation

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00_	000	001	002	003	004	005	006	007	008	009	080	081	800	801	880	881
01_	010	011	012	013	014	015	016	017	018	019	090	091	810	811	890	891
02_	020	021	022	023	024	025	026	027	028	029	082	083	820	821	808	809
03_	030	031	032	033	034	035	036	037	038	039	092	093	830	831	818	819
04_	040	041	042	043	044	045	046	047	048	049	084	085	840	841	088	089
05_	050	051	052	053	054	055	056	057	058	059	094	095	850	851	098	099
06_	060	061	062	063	064	065	066	067	068	069	086	087	860	861	888	889
07_	070	071	072	073	074	075	076	077	078	079	096	097	870	871	898	899
08_	100	101	102	103	104	105	106	107	108	109	180	181	900	901	980	981
09_	110	111	112	113	114	115	116	117	118	119	190	191	910	911	990	991
0A_	120	121	122	123	124	125	126	127	128	129	182	183	920	921	908	909
0B_	130	131	132	133	134	135	136	137	138	139	192	193	930	931	918	919
0C_	140	141	142	143	144	145	146	147	148	149	184	185	940	941	188	189
0D_	150	151	152	153	154	155	156	157	158	159	194	195	950	951	198	199
0E_	160	161	162	163	164	165	166	167	168	169	186	187	960	961	988	989
0F_	170	171	172	173	174	175	176	177	178	179	196	197	970	971	998	999
10_	200	201	202	203	204	205	206	207	208	209	280	281	802	803	882	883
11_	210	211	212	213	214	215	216	217	218	219	290	291	812	813	892	893
12_	220	221	222	223	224	225	226	227	228	229	282	283	822	823	828	829
13_	230	231	232	233	234	235	236	237	238	239	292	293	832	833	838	839
14_	240	241	242	243	244	245	246	247	248	249	284	285	842	843	288	289
15_	250	251	252	253	254	255	256	257	258	259	294	295	852	853	298	299
16_	260	261	262	263	264	265	266	267	268	269	286	287	862	863	(888)	(889)
17_	270	271	272	273	274	275	276	277	278	279	296	297	872	873	(898)	(899)
18_	300	301	302	303	304	305	306	307	308	309	380	381	902	903	982	983
19_	310	311	312	313	314	315	316	317	318	319	390	391	912	913	992	993
1A_	320	321	322	323	324	325	326	327	328	329	382	383	922	923	928	929
1B_	330	331	332	333	334	335	336	337	338	339	392	393	932	933	938	939
1C_	340	341	342	343	344	345	346	347	348	349	384	385	942	943	388	389
1D_	350	351	352	353	354	355	356	357	358	359	394	395	952	953	398	399
1E_	360	361	362	363	364	365	366	367	368	369	386	387	962	963	(988)	(989)
1F_	370	371	372	373	374	375	376	377	378	379	396	397	972	973	(998)	(999)
20_	400	401	402	403	404	405	406	407	408	409	480	481	804	805	884	885
21_	410	411	412	413	414	415	416	417	418	419	490	491	814	815	894	895
22_	420	421	422	423	424	425	426	427	428	429	482	483	824	825	848	849
23_	430	431	432	433	434	435	436	437	438	439	492	493	834	835	858	859
24_	440	441	442	443	444	445	446	447	448	449	484	485	844	845	488	489
25_	450	451	452	453	454	455	456	457	458	459	494	495	854	855	498	499
26_	460	461	462	463	464	465	466	467	468	469	486	487	864	865	(888)	(889)
27_	470	471	472	473	474	475	476	477	478	479	496	497	874	875	(898)	(899)
28_	500	501	502	503	504	505	506	507	508	509	580	581	904	905	984	985
29_	510	511	512	513	514	515	516	517	518	519	590	591	914	915	994	995
2A_	520	521	522	523	524	525	526	527	528	529	582	583	924	925	948	949
2B_	530	531	532	533	534	535	536	537	538	539	592	593	934	935	958	959
2C_	540	541	542	543	544	545	546	547	548	549	584	585	944	945	588	589
2D_	550	551	552	553	554	555	556	557	558	559	594	595	954	955	598	599
2E_	560	561	562	563	564	565	566	567	568	569	586	587	964	965	(988)	(989)
2F_	570	571	572	573	574	575	576	577	578	579	596	597	974	975	(998)	(999)
30_	600	601	602	603	604	605	606	607	608	609	680	681	806	807	886	887
31_	610	611	612	613	614	615	616	617	618	619	690	691	816	817	896	897
32_	620	621	622	623	624	625	626	627	628	629	682	683	826	827	868	869
33_	630	631	632	633	634	635	636	637	638	639	692	693	836	837	878	879
34_	640	641	642	643	644	645	646	647	648	649	684	685	846	847	688	689
35_	650	651	652	653	654	655	656	657	658	659	694	695	856	857	698	699
36_	660	661	662	663	664	665	666	667	668	669	686	687	866	867	(888)	(889)
37_	670	671	672	673	674	675	676	677	678	679	696	697	876	877	(898)	(899)
38_	700	701	702	703	704	705	706	707	708	709	780	781	906	907	986	987
39_	710	711	712	713	714	715	716	717	718	719	790	791	916	917	996	997
3A_	720	721	722	723	724	725	726	727	728	729	782	783	926	927	968	969
3B_	730	731	732	733	734	735	736	737	738	739	792	793	936	937	978	979
3C_	740	741	742	743	744	745	746	747	748	749	784	785	946	947	788	789
3D_	750	751	752	753	754	755	756	757	758	759	794	795	956	957	798	799
3E_	760	761	762	763	764	765	766	767	768	769	786	787	966	967	(988)	(989)
3F_	770	771	772	773	774	775	776	777	778	779	796	797	976	977	(998)	(999)

Figure 20-9. DPD to BCD Translation

Range of v	Case	Normal Result (r) when Rounding Mode Is							
		RNE	RNTZ	RNAZ	RAFZ	RTMI	RFSP	RTPI	RTZ
$v < -N_{\max}, q < -N_{\max}$	Overflow	$-\infty^1$	$-\infty^1$	$-\infty^1$	$-\infty^1$	$-\infty^1$	$-N_{\max}$	$-N_{\max}$	$-N_{\max}$
$v < -N_{\max}, q = -N_{\max}$	Normal	$-N_{\max}$	$-N_{\max}$	$-N_{\max}$	—	—	$-N_{\max}$	$-N_{\max}$	$-N_{\max}$
$-N_{\max} \leq v \leq -N_{\min}$	Normal	b	b	b	b	b	b	b	b
$-N_{\min} < v \leq -D_{\min}$	Tiny	b*	b*	b*	b*	b*	b*	b	b
$-D_{\min} < v < -D_{\min}/2$	Tiny	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	-0	-0
$v = -D_{\min}/2$	Tiny	-0	-0	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	-0	-0
$-D_{\min}/2 < v < 0$	Tiny	-0	-0	-0	$-D_{\min}$	$-D_{\min}$	$-D_{\min}$	-0	-0
$v = 0$	EZD	+0	+0	+0	+0	-0	+0	+0	+0
$0 < v < +D_{\min}/2$	Tiny	+0	+0	+0	$+D_{\min}$	+0	$+D_{\min}$	$+D_{\min}$	+0
$v = +D_{\min}/2$	Tiny	+0	+0	$+D_{\min}$	$+D_{\min}$	+0	$+D_{\min}$	$+D_{\min}$	+0
$+D_{\min}/2 < v < +D_{\min}$	Tiny	$+D_{\min}$	$+D_{\min}$	$+D_{\min}$	$+D_{\min}$	+0	$+D_{\min}$	$+D_{\min}$	+0
$+D_{\min} \leq v < +N_{\min}$	Tiny	b*	b*	b*	b*	b	b*	b*	b
$+N_{\min} \leq v \leq +N_{\max}$	Normal	b	b	b	b	b	b	b	b
$+N_{\max} < v, q = +N_{\max}$	Normal	$+N_{\max}$	$+N_{\max}$	$+N_{\max}$	—	$+N_{\max}$	$+N_{\max}$	—	$+N_{\max}$
$+N_{\max} < v, q > +N_{\max}$	Overflow	$+\infty^1$	$+\infty^1$	$+\infty^1$	$+\infty^1$	$+N_{\max}$	$+N_{\max}$	$+\infty^1$	$+N_{\max}$

**Explanation:**

— This situation cannot occur.

<sup>1</sup> The normal result r is considered to have been incremented.

\* The rounded value, in the extreme case, may be  $N_{\min}$ . In this case, the exception conditions are underflow, inexact, and incremented.

b The value derived when the precise result v is rounded to the destination's precision, including both bounded precision and bounded exponent range.

q The value derived when the precise result v is rounded to the destination's precision, but assuming an unbounded exponent range.

r This is the returned value when neither overflow nor underflow is enabled.

v Precise result before rounding, assuming unbounded precision and an unbounded exponent range. For data-format conversion operations, v is the source value.

$D_{\min}$  Smallest (in magnitude) representable subnormal number in the target format.

EZD The result r of the exact-zero-difference case applies only to ADD and SUBTRACT with both source operands having opposite signs. (For ADD and SUBTRACT, when both source operands have the same sign, the sign of the zero result is the same sign as the sign of the source operands.)

$N_{\max}$  Largest (in magnitude) representable finite number in the target format.

$N_{\min}$  Smallest (in magnitude) representable normalized number in the target format.

RAFZ Round away from 0.

RFSP Round to prepare for shorter precision.

RNAZ Round to nearest ties away from 0.

RNE Round to nearest ties to even.

RNTZ Round to nearest ties toward 0.

RTMI Round toward  $-\infty$ .

RTPI Round toward  $+\infty$ .

RTZ Round toward 0.

Figure 20-10. (Part 1 of 2) Rounding and Range Actions

Case	Is r inexact (r≠v)	Overflow Mask (FPC 0.2)	Underflow Mask (FPC 0.3)	Inexact Mask (FPC 0.4)	Is r Incremented ( r > v )	Is q inexact (q≠v)	Is q Incremented ( q > v )	Results
Overflow	Yes <sup>1</sup>	0	—	0	—	—	—	T(r), SFo ← 1, SFx ← 1
Overflow	Yes <sup>1</sup>	0	—	1	No	—	—	T(r), SFo ← 1, PIDx(08)
Overflow	Yes <sup>1</sup>	0	—	1	Yes	—	—	T(r), SFo ← 1, PIDy(0C)
Overflow	Yes <sup>1</sup>	1	—	—	—	No	No <sup>1</sup>	Tw(q ÷ β), PIDo(20)
Overflow	Yes <sup>1</sup>	1	—	—	—	Yes	No	Tw(q ÷ β), PIDox(28)
Overflow	Yes <sup>1</sup>	1	—	—	—	Yes	Yes	Tw(q ÷ β), PIDoy(2C)
Normal	No	—	—	—	—	—	—	T(r)
Normal	Yes	—	—	0	—	—	—	T(r), SFx ← 1
Normal	Yes	—	—	1	No	—	—	T(r), PIDx(08)
Normal	Yes	—	—	1	Yes	—	—	T(r), PIDy(0C)
Tiny	No	—	0	—	—	—	—	T(r)
Tiny	No	—	1	—	—	No <sup>1</sup>	No <sup>1</sup>	Tw(q • β), PIDu(10)
Tiny	Yes	—	0	0	—	—	—	T(r), SFu ← 1, SFx ← 1
Tiny	Yes	—	0	1	No	—	—	T(r), SFu ← 1, PIDx(08)
Tiny	Yes	—	0	1	Yes	—	—	T(r), SFu ← 1, PIDy(0C)
Tiny	Yes	—	1	—	—	No	No <sup>1</sup>	Tw(q • β), PIDu(10)
Tiny	Yes	—	1	—	—	Yes	No	Tw(q • β), PIDux(18)
Tiny	Yes	—	1	—	—	Yes	Yes	Tw(q • β), PIDuy(1C)

**Explanation:**

- The results do not depend on this condition.
- <sup>1</sup> This condition is true by virtue of the state of some condition to the left of this column.
- β 10<sup>a</sup>, where a is the exponent adjustment. (See the sections, “IEEE Overflow” on page 20-7 and “IEEE Underflow” on page 20-8.)
- PIDc(h) Program interruption for data exception, condition c, with DXC of h in hex. See Figure 19-13 on page 19-14 of the *z/Architecture Principles of Operation*.
- q The value derived when the precise result v is rounded to the destination's precision, but assuming an unbounded exponent range.
- r The result as defined in Part 1 of this figure.
- SFo IEEE overflow flag, FPC 1.2.
- SFu IEEE underflow flag, FPC 1.2.
- SFx IEEE inexact flag, FPC 1.2.
- T(x) The value x is placed at the target operand location.
- Tw(x) The wrapped rounded result x is placed at the target operand location. For all operations except LOAD ROUNDED, the wrapped rounded result is in the same format and length as normal results at the target location. For LOAD ROUNDED, the wrapped rounded result is in the same format and length as the source, but rounded to the target-format precision.
- v Precise result before rounding, assuming unbounded precision and unbounded exponent range.

Figure 20-10. (Part 2 of 2) Rounding and Range Actions

## Result Figures

Concise descriptions of the results produced by many of the DFP instructions are made by means of figures which contain columns and rows representing all possible combinations of DFP data class for the source operands of an instruction. The information shown at the intersection of a row and a column is one or more symbols representing the result or results produced for that particular combination of source-operand data classes. Explanations of the symbols used are contained in each figure. In many cases, the explanation of a particular result is in the form of a cross-reference to another figure. In many cases, the information shown at the intersection consists of several symbols separated by commas. All such results are produced unless one of the results is a program interruption. In the case of a program interruption, the operation is suppressed or completed as shown in Figure 20-12.

## Data-Exception Codes (DXC) and Abbreviations

Figure 20-11 shows IEEE exception-condition and flag abbreviations that are used in the result figures, and it explains the symbols “X<sub>i</sub>” and “X<sub>z</sub>” that are used in the figures. Bits 0-4 (i,z,o,u,x) of the eight-bit data-exception code (DXC) in byte 2 of the FPC register are trap flags and correspond to the same bits in bytes 0 and 1 of the register (IEEE masks and IEEE flags). The trap flag for an exception, instead of the IEEE flag, is set to one when an interruption for the exception is enabled by the corresponding IEEE mask bit. Bit 5 (y) of byte 2 is used in conjunction with bit 4, inexact (x), to indicate that the result has been incremented in magnitude.

Figure 20-12 shows the various DXCs that can be indicated, the associated instruction endings, and abbreviations that are used for the DXCs in the result figures. (The abbreviation “PID” stands for “program interruption for a data exception.”)

Exception Condition		FPC IEEE Mask bit	IEEE Flag	
Name	Abbr.		FPC Bit	Abbr.
IEEE invalid operation	X <sub>i</sub> <sup>1</sup>	0.0	1.0	SFi
IEEE division by zero	X <sub>z</sub> <sup>1</sup>	0.1	1.1	SFz
IEEE overflow	X <sub>o</sub>	0.2	1.2	SFo
IEEE underflow	X <sub>u</sub>	0.3	1.3	SFu
IEEE inexact	X <sub>x</sub>	0.4	1.4	SFx

**Explanation:**

<sup>1</sup> In a figure, the symbol “X<sub>i</sub>” (or “X<sub>z</sub>”) followed by a list of results indicates that when FPC 0.0 (or FPC 0.1) is zero, then instruction execution is completed by setting SFi [FPC 1.0] (or SFz [FPC 1.1]) to one and producing the indicated results; and when FPC 0.0 (or FPC 0.1) is one, then instruction execution is suppressed, the data exception code (DXC) is set to 80 (or 40) hex, and a program interruption for a data exception occurs.

Figure 20-11. IEEE Exception-Condition and Flag Abbreviations

Abbr.	DXC (hex)	Data-Exception-Code Name	Instruction Ending
PIDx	08	IEEE inexact and truncated	Complete
PIDy	0C	IEEE inexact and incremented	Complete
PIDu	10	IEEE underflow, exact	Complete, wrap exponent
PIDux	18	IEEE underflow, inexact and truncated	Complete, wrap exponent
PIDuy	1C	IEEE underflow, inexact and incremented	Complete, wrap exponent
PIDo	20	IEEE overflow, exact	Complete, wrap exponent
PIDox	28	IEEE overflow, inexact and truncated	Complete, wrap exponent
PIDoy	2C	IEEE overflow, inexact and incremented	Complete, wrap exponent
PIDz	40	IEEE division by zero	Suppress
PIDi	80	IEEE invalid operation	Suppress

Figure 20-12. IEEE Data-Exception Codes (DXC) and Abbreviations

## Instructions

The Decimal-Floating-Point (DFP) instructions and their mnemonics and operation codes are listed in Figure 18. The figure indicates, in the column labeled “Characteristics,” the instruction format, when the condition code is set, and the exceptional conditions in operand designations, data, or results that cause a program interruption.

All DFP instructions are subject to the AFP-register-control bit, bit 45 of control register 0. For the DFP instructions to be executed successfully, the AFP-register-control bit must be one; otherwise, a DFP-instruction data exception, DXC 3, is recognized.

Mnemonics for the DFP instructions are distinguished from corresponding HFP or BFP instructions

by a T in the mnemonic. Mnemonics for the DFP instructions have an R as the last letter when the instruction is in the RRE or RRF format. Certain letters are used for DFP instructions to represent operand-format length, as follows:

- G Sixty-four-bit fixed point
- E Short
- D Long
- X Extended

**Note:** In the detailed descriptions of the individual instructions, the mnemonic and the symbolic operand designation for the assembler language are shown with each instruction. For a register-to-register operation using COMPARE (long), for example, CDTR is the mnemonic and R<sub>1</sub>,R<sub>2</sub> the operand designation.

Name	Mnemonic	Characteristics						Op Code
ADD (extended DFP)	AXTR	RRR	C	DF	SP	Dt Xi Xo Xu Xx	I	B3DA
ADD (long DFP)	ADTR	RRR	C	DF		Dt Xi Xo Xu Xx	I	B3D2
COMPARE (extended DFP)	CXTR	RRE	C	DF	SP	Dt Xi		B3EC
COMPARE (long DFP)	CDTR	RRE	C	DF		Dt Xi		B3E4
COMPARE AND SIGNAL (extended DFP)	KXTR	RRE	C	DF	SP	Dt Xi		B3E8
COMPARE AND SIGNAL (long DFP)	KDTR	RRE	C	DF		Dt Xi		B3E0
COMPARE EXPONENT (extended DFP)	CEXTR	RRE	C	DF	SP	Dt		B3FC
COMPARE EXPONENT (long DFP)	CEDTR	RRE	C	DF		Dt		B3F4
CONVERT FROM FIXED (64 to extended DFP)	CXGTR	RRE		DF	SP	Dt	I	B3F9
CONVERT FROM FIXED (64 to long DFP)	CDGTR	RRE		DF		Dt Xx	I	B3F1
CONVERT FROM SIGNED BCD (128 to extended DFP)	CXSTR	RRE		DF	SP	Dt Dd	I	B3FB
CONVERT FROM SIGNED BCD (64 to long DFP)	CDSTR	RRE		DF		Dt Dd	I	B3F3
CONVERT FROM UNSIGNED BCD (128 to extended DFP)	CXUTR	RRE		DF	SP	Dt Dd	I	B3FA
CONVERT FROM UNSIGNED BCD (64 to long DFP)	CDUTR	RRE		DF		Dt Dd	I	B3F2
CONVERT TO FIXED (extended DFP to 64)	CGXTR	RRF	C	DF	SP	Dt Xi Xx		B3E9
CONVERT TO FIXED (long DFP to 64)	CGDTR	RRF	C	DF		Dt Xi Xx		B3E1
CONVERT TO SIGNED BCD (extended DFP to 128)	CSXTR	RRF		DF	SP	Dt		B3EB
CONVERT TO SIGNED BCD (long DFP to 64)	CSDTR	RRF		DF		Dt		B3E3
CONVERT TO UNSIGNED BCD (extended DFP to 128)	CUXTR	RRE		DF	SP	Dt		B3EA
CONVERT TO UNSIGNED BCD (long DFP to 64)	CUDTR	RRE		DF		Dt		B3E2
DIVIDE (extended DFP)	DXTR	RRR		DF	SP	Dt Xi Xz Xo Xu Xx	I	B3D9
DIVIDE (long DFP)	DDTR	RRR		DF		Dt Xi Xz Xo Xu Xx	I	B3D1
EXTRACT BIASED EXPONENT (extended DFP to 64)	EEXTR	RRE		DF	SP	Dt		B3ED
EXTRACT BIASED EXPONENT (long DFP to 64)	EEDTR	RRE		DF		Dt		B3E5
EXTRACT SIGNIFICANCE (extended DFP)	ESXTR	RRE		DF	SP	Dt		B3EF
EXTRACT SIGNIFICANCE (long DFP)	ESDTR	RRE		DF		Dt		B3E7
INSERT BIASED EXPONENT (64 to extended DFP)	IEXTR	RRF		DF	SP	Dt	I	B3FE
INSERT BIASED EXPONENT (64 to long DFP)	IEDTR	RRF		DF		Dt	I	B3F6

Figure 20-13. Summary of DFP Instructions (Part 1 of 3)

Name	Mnemonic	Characteristics						Op Code
LOAD AND TEST (extended DFP)	LTXTR	RRE	C	DF	SP	Dt Xi		B3DE
LOAD AND TEST (long DFP)	LTDTR	RRE	C	DF		Dt Xi		B3D6
LOAD FP INTEGER (extended DFP)	FIXTR	RRF		DF	SP	Dt Xi Xx <sup>1</sup>	I	B3DF
LOAD FP INTEGER (long DFP)	FIDTR	RRF		DF		Dt Xi Xx <sup>1</sup>	I	B3D7
LOAD FPC AND SIGNAL	LFAS	S		DF	A SP	Dt	B <sub>2</sub>	B2BD
LOAD LENGTHENED (long to extended DFP)	LXDTR	RRF		DF	SP	Dt Xi <sup>2</sup>	I	B3DC
LOAD LENGTHENED (short to long DFP)	LDETR	RRF		DF		Dt Xi <sup>2</sup>	I	B3D4
LOAD ROUNDED (extended to long DFP)	LDXTR	RRF		DF	SP	Dt Xi <sup>2</sup> Xo Xu Xx	I	B3DD
LOAD ROUNDED (long to short DFP)	LEDTR	RRF		DF		Dt Xi <sup>2</sup> Xo Xu Xx	I	B3D5
MULTIPLY (extended DFP)	MXTR	RRR		DF	SP	Dt Xi Xo Xu Xx	I	B3D8
MULTIPLY (long DFP)	MDTR	RRR		DF		Dt Xi Xo Xu Xx	I	B3D0
QUANTIZE (extended DFP)	QAXTR	RRF		DF	SP	Dt Xi Xx	I	B3FD
QUANTIZE (long DFP)	QADTR	RRF		DF		Dt Xi Xx	I	B3F5
REROUND (extended DFP)	RRXTR	RRF		DF	SP	Dt Xi Xx	I	B3FF
REROUND (long DFP)	RRDTR	RRF		DF		Dt Xi Xx	I	B3F7
SET DFP ROUNDING MODE	SRNMT	S		DF		Dt		B2B9
SET FPC AND SIGNAL	SFASR	RRE		DF	SP	Dt		
SHIFT COEFFICIENT LEFT (extended DFP)	SLXT	RXF		DF	SP	Dt		ED48
SHIFT COEFFICIENT LEFT (long DFP)	SLDT	RXF		DF		Dt		ED40
SHIFT COEFFICIENT RIGHT (extended DFP)	SRXT	RXF		DF	SP	Dt		ED49
SHIFT COEFFICIENT RIGHT (long DFP)	SRDT	RXF		DF		Dt		ED41
SUBTRACT (extended DFP)	SXTR	RRR	C	DF	SP	Dt Xi Xo Xu Xx	I	B3DB
SUBTRACT (long DFP)	SDTR	RRR	C	DF		Dt Xi Xo Xu Xx	I	B3D3
TEST DATA CLASS (extended DFP)	TDCXT	RXE	C	DF	SP	Dt		ED58
TEST DATA CLASS (long DFP)	TDCDT	RXE	C	DF		Dt		ED54
TEST DATA CLASS (short DFP)	TDCET	RXE	C	DF		Dt		ED50
TEST DATA GROUP (extended DFP)	TDGXT	RXE	C	DF	SP	Dt		ED59
TEST DATA GROUP (long DFP)	TDGD	RXE	C	DF		Dt		ED55
TEST DATA GROUP (short DFP)	TDGET	RXE	C	DF		Dt		ED51

Figure 20-13. Summary of DFP Instructions (Part 2 of 3)

Name	Mne- monic	Characteristics	Op Code
<b>Explanation:</b>			
<sup>1</sup>		When the suppression of inexact indication bit is zero.	
<sup>2</sup>		When the suppression of invalid operation bit is zero.	
A		Access exceptions for logical addresses.	
B <sub>2</sub>		B <sub>2</sub> field designates an access register in the access-register mode.	
C		Condition code is set.	
Dd		Decimal-operand data exception (DXC = 00 hex).	
Dt		DFP-instruction data exception (DXC = 03 hex).	
DF		Decimal-Floating-Point facility.	
I		An ideal exponent is defined for this instruction.	
RRE		RRE instruction format.	
RRF		RRF instruction format.	
RRR		RRR instruction format.	
RXE		RXE instruction format.	
RXF		RXF instruction format.	
S		S instruction format.	
SP		Specification exception.	
Xi		IEEE invalid-operation condition.	
Xo		IEEE overflow condition.	
Xu		IEEE underflow condition.	
Xx		IEEE inexact condition.	
Xz		IEEE division-by-zero condition.	

Figure 20-13. Summary of DFP Instructions (Part 3 of 3)

## ADD

Mnemonic R<sub>1</sub>,R<sub>2</sub>,R<sub>3</sub> [RRR]

'Op Code'	R <sub>3</sub>	///	R <sub>1</sub>	R <sub>2</sub>
0	16	20	24	28 31

Mnemonic	Op Code	Operands
ADTR	'B3D2'	Long DFP
AXTR	'B3DA'	Extended DFP

The third operand is added to the second operand, and the sum is placed at the first-operand location.

If both operands are numeric and finite, they are added algebraically, forming an intermediate sum. The intermediate sum, if nonzero, is rounded to the target-format precision according to the current DFP rounding mode. An appropriate form of the rounded intermediate sum is selected based on the ideal exponent and is placed at the result location. The ideal exponent is the lesser exponent (in value) of the two source operands.

The sign of the sum is determined by the rules of algebra. This also applies to a result of zero:

- If the result of rounding a nonzero intermediate sum is zero, the sign of the zero result is the sign of the intermediate sum.
- If the sum of two operands with opposite signs is exactly zero, the sign of the result is plus in all rounding modes except round toward  $-\infty$ , in which mode the sign is minus.
- The sign of the sum  $x$  plus  $x$  is the sign of  $x$ , even when  $x$  is zero.

If one operand is an infinity and the other is finite and numeric, the result is that infinity. If both operands are infinities of the same sign, the result is the same infinity. If the two operands are infinities of opposite signs, an IEEE-invalid-operation condition is recognized.

See Figure 20-14 on page 20-19 for a detailed description of the results of this instruction.

For AXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

Second Operand (b) is	Results for ADD (b+c) when Third Operand (c) is				
	$-\infty$	F	$+\infty$	QNaN	SNaN
$-\infty$	T(-dINF), cc1	T(-dINF), cc1	X <sub>i</sub> : T(dNaN), cc3	P(c), cc3	X <sub>i</sub> : U(c), cc3
F	T(-dINF), cc1	S(b+c), ccrs	T(+dINF), cc2	P(c), cc3	X <sub>i</sub> : U(c), cc3
$+\infty$	X <sub>i</sub> : T(dNaN), cc3	T(+dINF), cc2	T(+dINF), cc2	P(c), cc3	X <sub>i</sub> : U(c), cc3
QNaN	P(b), cc3	P(b), cc3	P(b), cc3	P(b), cc3	X <sub>i</sub> : U(c), cc3
SNaN	X <sub>i</sub> : U(b), cc3	X <sub>i</sub> : U(b), cc3	X <sub>i</sub> : U(b), cc3	X <sub>i</sub> : U(b), cc3	X <sub>i</sub> : U(b), cc3

**Explanation:**

b+c The value c added to b, rounded to the target-format precision, and returned in the appropriate form. (See Figure 20-10 on page 20-13.)

ccn Condition code is set to n.

ccrs Condition code is set according to the resultant sum. (See Figure 20-15 on page 20-19)

dINF Default infinity.

dNaN Default quiet NaN.

F All finite numbers, including zeros.

P(x) The QNaN of operand x is propagated and placed at the first operand location.

S(x) The value x is placed at the first operand location with the sign set by the rules of algebra.

T(x) The value x is placed at the first operand location.

U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.

X<sub>i</sub>: y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-14. Results: ADD

Value of Result (r)	Condition code
r = 0	cc0
r < 0	cc1
r > 0	cc2

**Explanation:**

ccn Condition code is set to n.

Figure 20-15. Condition Code for Resultant Sum

### Resulting Condition Code:

- 0 Result is zero
- 1 Result is less than zero
- 2 Result is greater than zero
- 3 Result is a NaN

### IEEE Exception Conditions:

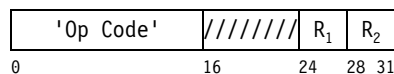
- Invalid operation
- Overflow
- Underflow
- Inexact

### Program Exceptions:

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (AXTR only)

## COMPARE

Mnemonic R<sub>1</sub>,R<sub>2</sub> [RRE]



Mnemonic	Op Code	Operands
CDTR	'B3E4'	Long DFP
CXTR	'B3EC'	Extended DFP

The first operand is compared with the second operand, and the condition code is set to indicate the result.

If both operands are numeric and finite, the comparison is algebraic and follows the procedure for DFP subtraction, except that the difference is discarded after setting the condition code, and both operands remain unchanged. If the difference is exactly zero with either sign, the operands are equal; this includes zero operands (so +0 equals -0). If a nonzero difference is positive or negative, the first operand is high or low, respectively.

$+\infty$  compares greater than any finite number, and all finite numbers compare greater than  $-\infty$ . Two infinity operands of like sign compare equal and all zeros compare equal.

Numeric comparison is exact, and the condition code is determined for finite operands as if range and pre-

cision were unlimited. No overflow or underflow condition can occur.

If either or both operands are QNaNs and neither operand is an SNaN, the comparison result is unordered, and condition code 3 is set.

If either or both operands are SNaNs, an IEEE-invalid-operation condition is recognized. If the IEEE invalid-operation mask bit is one, a program interruption for a data exception with DXC 80 hex (IEEE

invalid operation) occurs. If the IEEE-invalid-operation mask bit is zero, the IEEE-invalid-operation flag bit is set to one, and instruction execution is completed by setting condition code 3.

See Figure 20-16 on page 20-20 for a detailed description of the results of this instruction.

For CXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

First Operand (a) is	Results for COMPARE (a:b) when Second Operand (b) is				
	$-\infty$	F	$+\infty$	QNaN	SNaN
$-\infty$	cc0	cc1	cc1	cc3	X <sub>i</sub> : cc3
F	cc2	C(a:b)	cc1	cc3	X <sub>i</sub> : cc3
$+\infty$	cc2	cc2	cc0	cc3	X <sub>i</sub> : cc3
QNaN	cc3	cc3	cc3	cc3	X <sub>i</sub> : cc3
SNaN	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3

**Explanation:**

ccn Condition code is set to n.  
C(a:b) Algebraic comparison. See Figure 20-17 on page 20-20.  
F All finite numbers, including zeros.  
X<sub>i</sub>: y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-16. Results: COMPARE

Relation of value (a) to value (b)	Condition code for C(a:b)
a = b	cc0
a < b	cc1
a > b	cc2

**Explanation:**

ccn Condition code is set to n.

Figure 20-17. Basic Compare Results

**Resulting Condition Code:**

- 0 Operands equal
- 1 First operand low
- 2 First operand high
- 3 Operands unordered

**IEEE Exception Conditions:**

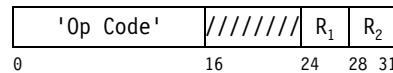
- Invalid operation

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (CXTR only)

## COMPARE AND SIGNAL

Mnemonic R<sub>1</sub>,R<sub>2</sub> [RRE]



Mnemonic	Op Code	Operands
KDTR	'B3E0'	Long DFP
KXTR	'B3E8'	Extended DFP

The first operand is compared with the second operand, and the condition code is set to indicate the result. The operation is the same as for COMPARE except that QNaN operands cause an IEEE-invalid-operation condition to be recognized. Thus QNaN operands are treated as if they were SNaNs.

See Figure 20-18 on page 20-21 for a detailed description of the results of this instruction.

For KXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

First Operand (a) is	Results for COMPARE AND SIGNAL (a:b) when Second Operand (b) is			
	$-\infty$	F	$+\infty$	NaN
$-\infty$	cc0	cc1	cc1	X <sub>i</sub> : cc3
F	cc2	C(a:b)	cc1	X <sub>i</sub> : cc3
$+\infty$	cc2	cc2	cc0	X <sub>i</sub> : cc3
NaN	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3	X <sub>i</sub> : cc3

**Explanation:**

ccn Condition code is set to n.  
C(a:b) Algebraic comparison. See Figure 20-17 on page 20-20.  
F All finite numbers, including zeros.  
X<sub>i</sub>: y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-18. Results: COMPARE AND SIGNAL

**Resulting Condition Code:**

- 0 Operands equal
- 1 First operand low
- 2 First operand high
- 3 Operands unordered

**IEEE Exception Conditions:**

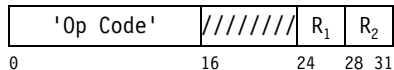
- Invalid operation

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (KXTR only)

## COMPARE EXPONENT

Mnemonic R<sub>1</sub>,R<sub>2</sub> [RRE]



Mnemonic	Op Code	Operands
CEDTR	'B3F4'	Long DFP
CEXTR	'B3FC'	Extended DFP

The exponent of the DFP first operand (X<sub>a</sub>) is compared to the exponent of the DFP second operand (X<sub>b</sub>), and the condition code is set to indicate the result.

See Figure 20-19 on page 20-21 for a detailed description of the results of this instruction.

For CEXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Resulting Condition Code:**

- 0 Exponents equal
- 1 First-operand exponent low
- 2 First-operand exponent high
- 3 Exponents unordered

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (CEXTR only)

First operand (a) is	Result for COMPARE EXPONENT (X <sub>a</sub> :X <sub>b</sub> ) when second operand (b) is			
	F	$\infty$	QNaN	SNaN
F	C(X <sub>a</sub> :X <sub>b</sub> )	cc3	cc3	cc3
$\infty$	cc3	cc0	cc3	cc3
QNaN	cc3	cc3	cc0	cc0
SNaN	cc3	cc3	cc0	cc0

**Explanation:**

C(X<sub>a</sub>:X<sub>b</sub>) Algebraic comparison. See Figure 20-20.  
ccn Condition code is set to n.  
F All finite numbers, including zeros.

Figure 20-19. Results: COMPARE EXPONENT

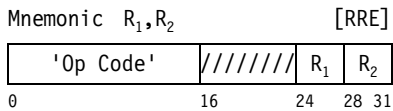
Relation of value X <sub>a</sub> to value X <sub>b</sub>	Condition Code for C(X <sub>a</sub> :X <sub>b</sub> )
X <sub>a</sub> = X <sub>b</sub>	cc0
X <sub>a</sub> < X <sub>b</sub>	cc1
X <sub>a</sub> > X <sub>b</sub>	cc2

**Explanation:**

ccn Condition code is set to n.

Figure 20-20. Results: COMPARE EXPONENT (part 2)

## CONVERT FROM FIXED



Mnemonic	Op Code	Operands
CDGTR	'B3F1'	64-bit binary-integer operand, long DFP result
CXGTR	'B3F9'	64-bit binary-integer operand, extended DFP result

The fixed-point second operand is converted to the DFP format, and the result is placed at the first-operand location. The ideal exponent is zero.

The second operand is a 64-bit signed binary integer that is located in the general register designated by  $R_2$ .

When the second operand is a nonzero number, it is converted to the exact DFP number, rounded to the target-format precision according to the current DFP rounding mode, and then placed at the first-operand location. See Figure 20-10 on page 20-13 for a detailed description of the rounding and range action.

When the second operand is zero, a positive zero with a zero exponent is returned.

For CXGTR, the  $R_1$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

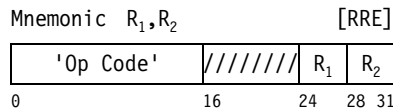
### IEEE Exception Conditions:

- Inexact (CDGTR only)

### Program Exceptions:

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (CXGTR only)

## CONVERT FROM SIGNED BCD



Mnemonic	Op Code	Operands
CDSTR	'B3F3'	64-bit BCD operand, long DFP result
CXSTR	'B3FB'	128-bit BCD operand, extended DFP result

The signed BCD number in the second operand is converted to a DFP number with the same value, and the result is placed at the first-operand location. The ideal exponent is zero.

The second operand is a signed BCD number that is located in the general register designated by  $R_2$ .

When an invalid BCD digit or sign code is detected in the source operand, a decimal-operand data exception condition is recognized.

For CXSTR, the  $R_1$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized. Also, the  $R_2$  field designates an even-odd pair of general registers and must designate an even-numbered register; otherwise, a specification exception is recognized.

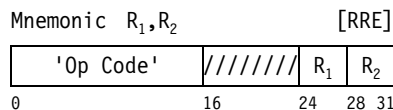
**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:** None.

### Program Exceptions:

- Data
- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (CXSTR only)

## CONVERT FROM UNSIGNED BCD



Mnemonic	Op Code	Operands
CDUTR	'B3F2'	64-bit BCD operand, long DFP result
CXUTR	'B3FA'	128-bit BCD operand, extended DFP result

The unsigned BCD number in the second operand is converted to a positive DFP number with the same magnitude, and the result is placed at the first-operand location. The ideal exponent is zero.

The second operand is an unsigned BCD number that is located in the general register or general-register pair designated by  $R_2$ .

When an invalid BCD digit code is detected in the source operand, a decimal-operand data exception condition is recognized.

For CXUTR, the  $R_1$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized. Also, the  $R_2$  field designates an even-odd pair of general registers and must designate an even-numbered register; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data
- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (CXUTR only)

## CONVERT TO FIXED

Mnemonic  $R_1, M_3, R_2$  [RRF]

'Op Code'	$M_3$	///	$R_1$	$R_2$
0	16	20	24	28 31

Mnemonic	Op Code	Operands
CGDTR	'B3E1'	Long DFP operand, 64-bit binary-integer result
CGXTR	'B3E9'	Extended DFP operand, 64-bit binary-integer result

The DFP second operand is rounded to an integer value and then converted to the fixed-point format. The result is placed at the first-operand location.

The result is a signed binary integer that is placed in the general register designated by  $R_1$ .

If the second operand is numeric and finite, it is rounded to an integer value by rounding as specified by the modifier in the  $M_3$  field:

### $M_3$ Rounding Method

- 0 According to the current DFP rounding mode
- 1-7 Reserved
- 8 Round to nearest with ties to even
- 9 Round toward 0
- 10 Round toward  $+\infty$
- 11 Round toward  $-\infty$
- 12 Round to nearest with ties away from 0
- 13 Round to nearest with ties toward 0
- 14 Round away from 0
- 15 Round to prepare for shorter precision

When the  $M_3$  field contains zero, rounding is controlled by the current DFP rounding mode specified in the FPC register. When bit 0 of the  $M_3$  field is one, rounding is performed as specified by the modifier, regardless of the current DFP rounding mode. Rounding for modifiers 8-15 is the same as for rounding modes 0-7 (binary 000-111), respectively.

The sign of the result is the sign of the second operand, except that a zero result has a plus sign.

See Figure 20-21 on page 20-24 for a detailed description of the results of this instruction.

For CGXTR, the  $R_2$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Resulting Condition Code:**

- 0 Source was zero
- 1 Source was less than zero
- 2 Source was greater than zero
- 3 Special case

**IEEE Exception Conditions:**

- Invalid operation
- Inexact

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (CGXTR only)

Second Operand (b)	p is	Is n Inexact (n ≠ b)	Inv.-Op. Mask (FPC 0.0)	Inexact Mask (FPC 0.4)	Is n Incremented ( nl >  bl)	Results
$-\infty \leq b < MN$	< MN	—	0	—	—	T(MN), SF <sub>i←1</sub> , cc3
$-\infty \leq b < MN$	< MN	—	1	—	—	PIDi(80)
$-\infty < b < MN$	= MN	—	—	0	—	T(MN), SF <sub>x←1</sub> , cc1
$-\infty < b < MN$	= MN	—	—	1	—	T(MN), cc1, PIDx(08)
$MN \leq b < 0$	—	No	—	—	—	T(n), cc1
$MN \leq b < 0$	—	Yes	—	0	—	T(n), SF <sub>x←1</sub> , cc1
$MN \leq b < 0$	—	Yes	—	1	No	T(n), cc1, PIDx(08)
$MN \leq b < 0$	—	Yes	—	1	Yes	T(n), cc1, PIDy(0C)
-0	—	No <sup>1</sup>	—	—	—	T(0), cc0
+0	—	No <sup>1</sup>	—	—	—	T(0), cc0
$0 < b \leq MP$	—	No	—	—	—	T(n), cc2
$0 < b \leq MP$	—	Yes	—	0	—	T(n), SF <sub>x←1</sub> , cc2
$0 < b \leq MP$	—	Yes	—	1	No	T(n), cc2, PIDx(08)
$0 < b \leq MP$	—	Yes	—	1	Yes	T(n), cc2, PIDy(0C)
$MP < b < +\infty$	= MP	—	—	0	—	T(MP), SF <sub>x←1</sub> , cc2
$MP < b < +\infty$	= MP	—	—	1	—	T(MP), cc2, PIDx(08)
$MP < b \leq +\infty$	> MP	—	0	—	—	T(MP), SF <sub>i←1</sub> , cc3
$MP < b \leq +\infty$	> MP	—	1	—	—	PIDi(80)
NaN	—	—	0	—	—	T(MN), SF <sub>i←1</sub> , cc3
NaN	—	—	1	—	—	PIDi(80)

**Explanation:**

- The results do not depend on this condition or mask bit.
- <sup>1</sup> This condition is true by virtue of the state of some condition to the left of this column.
- ccn Condition code is set to n.
- n The value p converted to a fixed-point result.
- p The value derived when the source value a is rounded to an integer using the specified rounding mode.
- MN Maximum negative number representable in the target fixed-point format.
- MP Maximum positive number representable in the target fixed-point format.
- PIDc(h) Program interruption for data exception, condition c, with DXC of h in hex.
- SF<sub>i</sub> IEEE invalid-operation flag, FPC 1.0.
- SF<sub>x</sub> IEEE inexact flag, FPC 1.4.
- T(x) The value x is placed at the first operand location

Figure 20-21. Results: CONVERT TO FIXED

## CONVERT TO SIGNED BCD

Mnemonic R<sub>1</sub>,R<sub>2</sub>,M<sub>4</sub> [RRF]

'Op Code'	////	M <sub>4</sub>	R <sub>1</sub>	R <sub>2</sub>
0	16	20	24	28 31

Mnemonic	Op Code	Operands
CSDTR	'B3E3'	Long DFP operand, 64-bit BCD result
CSXTR	'B3EB'	Extended DFP operand, 128-bit BCD result

For CSDTR, the rightmost 15 coefficient digits and the sign bit of the DFP second operand are converted into BCD digits and placed at the first-operand location with the sign digit padded on the right.

For CSXTR, the rightmost 31 coefficient digits and the sign bit of the DFP second operand are converted into BCD digits and placed at the first-operand location with the sign digit padded on the right.

Bit 3 of the M<sub>4</sub> field (M<sub>4</sub>.3) is the Plus Sign-Code Selection bit. When M<sub>4</sub>.3 is zero, the plus sign is encoded as 1100; when the bit is one, the plus sign is encoded as 1111. Bits 0-2 are reserved for future extensions and are ignored by the machine.

The result is a signed BCD number that is placed in the general register or general-register pair designated by R<sub>1</sub>.

This operation is performed for any second operand, including an infinity, QNaN, or SNaN, without causing an IEEE exception.

For CSXTR, the  $R_2$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized. Also, the  $R_1$  field designates an even-odd pair of general registers and must designate an even-numbered register; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (CSXTR only)

## CONVERT TO UNSIGNED BCD

Mnemonic	$R_1, R_2$	[RRE]								
<table border="1"> <tr> <td>'Op Code'</td> <td>////////</td> <td><math>R_1</math></td> <td><math>R_2</math></td> </tr> <tr> <td>0</td> <td>16</td> <td>24</td> <td>28 31</td> </tr> </table>			'Op Code'	////////	$R_1$	$R_2$	0	16	24	28 31
'Op Code'	////////	$R_1$	$R_2$							
0	16	24	28 31							

Mnemonic	Op Code	Operands
CUDTR	'B3E2'	Long DFP operand, 64-bit BCD result
CUXTR	'B3EA'	Extended DFP operand, 128-bit BCD result

For CUDTR, if the DFP second operand is a finite number, the rightmost 16 coefficient digits are converted into 16 BCD digits to form the result. If it is an infinity or NaN, the 5 10-bit DPD blocks in the coefficient-continuation field are converted into 15 BCD digits codes and a zero is padded to the left to form the result. The result is placed at the first-operand location.

For CUXTR, the rightmost 32 coefficient digits of the DFP second operand are converted into BCD digits and placed at the first-operand location.

The result is an unsigned BCD number that is placed in the general register or general-register pair designated by  $R_1$ .

This operation is performed for any second operand, including an infinity, QNaN, or SNaN, without causing an IEEE exception.

For CUXTR, the  $R_2$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized. Also, the  $R_1$  field designates an even-odd pair of general registers and must designate an even-numbered register; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (CUXTR only)

## DIVIDE

Mnemonic	$R_1, R_2, R_3$	[RRR]										
<table border="1"> <tr> <td>'Op Code'</td> <td><math>R_3</math></td> <td>////</td> <td><math>R_1</math></td> <td><math>R_2</math></td> </tr> <tr> <td>0</td> <td>16</td> <td>20</td> <td>24</td> <td>28 31</td> </tr> </table>			'Op Code'	$R_3$	////	$R_1$	$R_2$	0	16	20	24	28 31
'Op Code'	$R_3$	////	$R_1$	$R_2$								
0	16	20	24	28 31								

Mnemonic	Op Code	Operands
DDTR	'B3D1'	Long DFP
DXTR	'B3D9'	Extended DFP

The second operand (the dividend) is divided by the third operand (the divisor), and the quotient is placed at the first-operand location. No remainder is preserved. The ideal exponent is the difference of subtracting the exponent of the divisor from the exponent of the dividend.

If the dividend is a finite number and the divisor is a finite nonzero number, the dividend is divided by the divisor to form an intermediate quotient. The intermediate quotient is rounded to the target-format precision according to the current DFP rounding mode.

When the quotient is numeric (including infinity and zero) the sign of the quotient is the exclusive or of the operand signs.

When the dividend is a finite number and the divisor is infinity, then a true zero (zero coefficient and most negative exponent) is returned.

If the divisor is zero but the dividend is a finite nonzero number, an IEEE-division-by-zero condition is recognized. If the dividend and divisor are both zero, or if both are infinite, regardless of sign, an IEEE-invalid-operation condition is recognized.

Dividend (b) is	Results for DIVIDE (b ÷ c) when divisor (c) is				
	0	Fn	∞	QNaN	SNaN
0	X <sub>i</sub> : T(dNaN)	S(b÷c)	S(zt)	P(c)	X <sub>i</sub> : U(c)
Fn	X <sub>z</sub> : S(dINF)	S(b÷c)	S(zt)	P(c)	X <sub>i</sub> : U(c)
∞	S(dINF)	S(dINF)	X <sub>i</sub> : T(dNaN)	P(c)	X <sub>i</sub> : U(c)
QNaN	P(b)	P(b)	P(b)	P(b)	X <sub>i</sub> : U(c)
SNaN	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)

**Explanation:**

b÷c The value b divided by c, rounded to the target-format precision and returned in the appropriate form. (See Figure 20-10 on page 20-13.)

dINF Default infinity.

dNaN Default quiet NaN.

Fn Finite nonzero number (includes both normal and subnormal numbers).

P(x) The QNaN of operand x is propagated and placed at the first operand location.

S(x) The value x is placed at the first operand location with the sign set to the exclusive or of the operand signs.

T(x) The value x is placed at the first operand location.

U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.

X<sub>i</sub>: y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

X<sub>z</sub>: y IEEE division-by-zero exception. The results (y) are produced only when FPC 0.1 is zero.

zt True zero (zero coefficient and most negative exponent).

Figure 20-22. Results: DIVIDE

See Figure 20-22 on page 20-26 for a detailed description of the results of this instruction.

For DXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

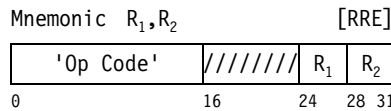
**IEEE Exception Conditions:**

- Invalid operation
- Division by zero
- Overflow
- Underflow
- Inexact

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (DXTR only)

## EXTRACT BIASED EXPONENT



Mnemonic	Op Code	Operands
EEDTR	'B3E5'	Long DFP operand, 64-bit binary-integer result
EEXTR	'B3ED'	Extended DFP operand, 64-bit binary-integer result

When the second operand is a finite number, the biased exponent of the second operand is placed into the first-operand location. When the second operand is an infinity, QNaN, or SNaN, a special code is placed into the first-operand location.

The result is a 64-bit signed binary integer that is placed in the general register designated by R<sub>1</sub>. See Figure 20-23 on page 20-27 for a detailed description of the results of this instruction.

The operation is performed for any operand without causing an IEEE exception.

For EEXTR, the R<sub>2</sub> field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

Second-operand data class	First-operand value
Finite number	$B_e$
Infinity	-1
QNaN	-2
SNaN	-3
<b>Explanation:</b>	
$B_e$	Biased exponent

Figure 20-23. Results: EXTRACT BIASED EXPONENT

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (EEXTR only)

## EXTRACT SIGNIFICANCE

Mnemonic  $R_1, R_2$  [RRE]

'Op Code'	////////	$R_1$	$R_2$
0	16	24	28 31

Mnemonic	Op Code	Operands
ESDTR	'B3E7'	Long DFP operand, 64-bit binary-integer result
ESXTR	'B3EF'	Extended DFP operand, 64-bit binary-integer result

The number of significant digits of the DFP second operand is placed at the first-operand location.

The result is a 64-bit signed binary integer that is placed in the general register designated by  $R_1$ . See Figure 20-24 for a detailed description of the results of this instruction.

Second-operand data class	First-operand value
Finite nonzero number	$NSD_b$
Zero	+0
Infinity	-1
QNaN	-2
SNaN	-3
<b>Explanation:</b>	
$NSD_b$	The number of significant digits of the second operand.

Figure 20-24. Results: EXTRACT SIGNIFICANCE

The operation is performed for any operand without causing an IEEE exception.

For ESXTR, the  $R_2$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (ESXTR only)

## INSERT BIASED EXPONENT

Mnemonic  $R_1, R_3, R_2$  [RRF]

'Op Code'	$R_3$	////	$R_1$	$R_2$
0	16	20	24	28 31

Mnemonic	Op Code	Operands
IEDTR	'B3F6'	64-bit binary-integer second operand, long DFP third operand, long DFP result
IEEXTR	'B3FE'	64-bit binary-integer second operand, extended DFP third operand, extended DFP result

The biased target exponent in the second operand is combined with the sign bit and the coefficient value of the DFP third operand, and the result is placed in the first-operand location. The ideal exponent is the specified target exponent.

The second operand is a 64-bit signed binary integer that is located in the general register designated by  $R_2$ .

When the second operand is a special code, an infinity, QNaN, or SNaN is formed in the result with the coefficient-continuation field containing the value from the coefficient-continuation field of the third operand. When the result is an infinity or QNaN, the biased-exponent-continuation field contains all zeros; when it is an SNaN, bit 0 of the field is one and all other bits of the field are zeros. The sign of the result is the same as the sign of the third operand.

See Figure 27 for a detailed description of the results of this instruction.

The operation is performed for any operand without causing an IEEE exception.

Value (b) in second operand	Result in first operand	Results for INSERT BIASED EXPONENT when third operand (c) is			
		F	$\infty$	QNaN	SNaN
$b > \text{MBE}$	QNaN	Q, Rc	Q, Rc	Q, Rc	Q, Rc
$\text{MBE} \geq b \geq 0$	Finite number with biased exponent b	N, Rc	Z, Rc	Z, Rc	Z, Rc
$b = -1$	$\infty$	I, Rc	I, Rc	I, Rc	I, Rc
$b = -2$	QNaN	Q, Rc	Q, Rc	Q, Rc	Q, Rc
$b = -3$	SNaN	S, Rc	S, Rc	S, Rc	S, Rc
$B \leq -4$	QNaN	Q, Rc	Q, Rc	Q, Rc	Q, Rc

**Explanation:**

- F All finite numbers, including zeros.
- I The combination field of the result is set to indicate an infinity and the biased-exponent continuation field (BXCF) is set to zero.
- MBE Maximum biased exponent for the target format.
- N The combination field and the BXCF of the result are set to the biased exponent in the second operand and the leftmost coefficient digit of the third operand.
- Q The combination field of the result is set to indicate a NaN and the BXCF is set to zero.
- Rc The contents of the third operand's coefficient-continuation field (CCF) are re-encoded using preferred DPD codes and the re-encoded result is placed in the first operand's CCF. The third operand's sign bit is copied into the first operand's sign bit.
- S The combination field of the result is set to indicate a NaN, and the leftmost bit in the BXCF is set to one and the remaining bits in the BXCF are set to zero.
- Z The combination and BXCF of the result are set to the biased exponent in the second operand and a leftmost coefficient digit of zero.

Figure 20-25. Results: INSERT BIASED EXPONENT

For IEXTR, the  $R_1$  and  $R_3$  fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

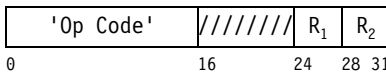
**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (IEXTR only)

**LOAD AND TEST**

Mnemonic  $R_1, R_2$  [RRE]



Mnemonic	Op Code	Operands
LTDR	'B3D6'	Long DFP
LTXR	'B3DE'	Extended DFP

The second operand is placed at the first-operand location, and its sign and magnitude are tested to determine the setting of the condition code. The con-

dition code is set the same as for a comparison of the second operand with zero.

If the second operand is a finite number, the same value and form is placed at the first-operand location with a re-encoded coefficient-continuation field. If the second operand is an infinity, a default infinity is placed at the first-operand location. If the second operand is a QNaN, the propagated QNaN is placed at the first-operand location. If the second operand is an SNaN, an IEEE-invalid-operation condition is recognized; if there is no interruption, the result is the corresponding QNaN.

See Figure 20-26 on page 20-29 for a detailed description of the results of this instruction.

For LTXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Resulting Condition Code:**

- 0 Result is zero
- 1 Result is less than zero
- 2 Result is greater than zero
- 3 Result is a NaN

Instruction	Suppression of Invalid-Operation bit (M <sub>4</sub> bit 0)	Results for selected load instructions when second operand (b) is			
		F	∞	QNaN	SNaN
LOAD AND TEST	—	T(b)	T(dINF)	P(b) <sup>1</sup>	X <sub>i</sub> ; U(b) <sup>1</sup>
LOAD LENGTHENED	0	T(b)	T(dINF)	P(b) <sup>1,3</sup>	X <sub>i</sub> ; U(b) <sup>1,3</sup>
LOAD LENGTHENED	1	T(b)	P(b) <sup>1,3</sup>	P(b) <sup>1,3</sup>	P(b) <sup>2,3,5</sup>
LOAD ROUNDED	0	R(b)	T(dINF)	P(b) <sup>1,4</sup>	X <sub>i</sub> ; U(b) <sup>1,4</sup>
LOAD ROUNDED	1	R(b)	P(b) <sup>1,4</sup>	P(b) <sup>1,4</sup>	P(b) <sup>2,4,5</sup>

**Explanation:**

— The results do not depend on this condition.

<sup>1</sup> The biased-exponent continuation field is set to zero.

<sup>2</sup> Bits in the biased-exponent continuation field are set to zeros, except for bit 0 which is set to 1.

<sup>3</sup> The coefficient-continuation field is padded on the left with zeros.

<sup>4</sup> The leftmost digits in the coefficient-continuation field are removed.

<sup>5</sup> The invalid-op status flag is not set.

dINF Default infinity

F All finite numbers, including zeros.

P(x) The special symbol of operand x is propagated and placed at the first operand location.

R(x) The value x is rounded to the target-format precision. See Figure 20-10 on page 20-13.

T(x) The value x is placed at the first operand location.

U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.

X<sub>i</sub>; y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-26. Results: Selected Load Instructions

### IEEE Exception Conditions:

- Invalid operation

### Program Exceptions:

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (LTXTR only)

## LOAD FP INTEGER

Mnemonic R<sub>1</sub>, M<sub>3</sub>, R<sub>2</sub>, M<sub>4</sub> [RRF]

'Op Code'	M <sub>3</sub>	M <sub>4</sub>	R <sub>1</sub>	R <sub>2</sub>
0	16	20	24	28 31

Mnemonic	Op Code	Operands
FIDTR	'B3D7'	Long DFP
FIXTR	'B3DF'	Extended DFP

If the second operand is a finite number, and if the exponent is less than zero, the second operand is converted and rounded to an integer value in the same floating-point format, and the result is placed at the first-operand location. If the exponent of the second operand is greater than or equal to zero, the

result is set to the value of the second operand. The ideal exponent is the greater value of zero and the exponent of the second operand.

The rounding mode is specified by the modifier in the M<sub>3</sub> field:

### M<sub>3</sub> Rounding Method

- 0 According to the current DFP rounding mode
- 1-7 Reserved
- 8 Round to nearest with ties to even
- 9 Round toward 0
- 10 Round toward +∞
- 11 Round toward -∞
- 12 Round to nearest with ties away from 0
- 13 Round to nearest with ties toward 0
- 14 Round away from 0
- 15 Round to prepare for shorter precision

When the M<sub>3</sub> field contains zero, rounding is controlled by the current DFP rounding mode specified in the FPC register. When bit 0 of the M<sub>3</sub> field is one, rounding is performed as specified by the modifier, regardless of the current DFP rounding mode. Rounding for modifiers 8-15 is the same as for rounding modes 0-7 (binary 000-111), respectively.

Bit 1 of the  $M_4$  field ( $M_4.1$ ) is the Suppression of Inexact Indication bit. Bits 0, 2, and 3 are reserved for future extensions and are ignored by the machine.

When the delivered result differs in value from the second operand and  $M_4.1$  is zero, an inexact exception is recognized. Otherwise, if  $M_4.1$  is one, the IEEE-inexact condition is not recognized nor is the inexact status flag set.

No underflow exception is recognized by this operation, regardless of the value of the second operand.

If the second operand is an infinity or a QNaN, the result is the default infinity or the propagated QNaN, respectively; if the second operand is an SNaN, the result is the corresponding QNaN.

The sign of the result is the sign of the second operand, even when the result is zero.

See Figure 20-27 on page 20-30 for a detailed description of the results of FIDTR and FIXTR.

For FIXTR, the R fields must designate valid floating-point-register fields; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:**

- Invalid operation
- Inexact ( $M_4.1=0$  only)

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (FIXTR only)

Second Operand (b)	Is n Inexact (n ≠ b)	Suppression of Inexact Indication bit ( $M_4$ bit 1)	Inexact Mask (FPC 0.4)	Is n Incremented ( nl >  bl)	Inv.-Op. Mask (FPC 0.0)	Results
$-\infty$	No <sup>1</sup>	—	—	—	—	T(-dINF)
F	No	—	—	—	—	W(n)
F	Yes	0	0	—	—	W(n), SF <sub>x←1</sub>
F	Yes	0	1	No	—	W(n), PID <sub>x</sub> (08)
F	Yes	0	1	Yes	—	W(n), PID <sub>y</sub> (0C)
F	Yes	1	—	—	—	W(n)
$+\infty$	No <sup>1</sup>	—	—	—	—	T(+dINF)
QNaN	No <sup>1</sup>	—	—	—	—	P(b)
SNaN	No <sup>1</sup>	—	—	—	0	U(b), SF <sub>i←1</sub>
SNaN	No <sup>1</sup>	—	—	—	1	PID <sub>i</sub> (80)

**Explanation:**

— The results do not depend on this condition or mask bit.

<sup>1</sup> This condition is true by virtue of the state of some condition to the left of this column.

dINF Default infinity.

F All finite numbers, includes zeros.

n The value derived when the source operand, b, is rounded to an integer.

P(x) The QNaN of operand x is propagated and placed at the first operand location.

PIDc(h) Program interruption for data exception, condition c, with DXC of h in hex.

SF<sub>i</sub> IEEE invalid-operation flag, FPC 1.0.

SF<sub>x</sub> IEEE inexact flag, FPC 1.4.

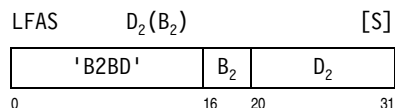
T(x) The value x is placed at the first operand location.

U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.

W(x) The value x in the form of zero exponent or the source exponent is placed at the first operand location.

Figure 20-27. Results: LOAD FP INTEGER

## LOAD FPC AND SIGNAL



The format of the four-byte source operand is treated the same as that of the floating-point control (FPC) register. When any signaling flag (the flag in the original FPC register) is one and the corresponding source mask is also one, an IEEE-interruption-simulation (IIS) event is recognized, which causes a specific content to be placed in the FPC register and a data-exception program interruption at completion of the instruction execution. Otherwise, a different content is placed in the FPC register, and instruction execution completes without recognizing an IIS event.

The source operand is the second operand in storage.

When an IIS event is recognized, the source masks, the source DFP rounding mode, and the source BFP rounding mode are placed in the corresponding fields of the FPC register. The flags in the FPC register are set to the logical OR of the signaling flags and the source flags. The DXC for the interruption is shown in Figure 20-28.

Enabled Signaling Flags					DXC (Binary)
Bit#	0	1	2	3 4	
	1	-	-	-	1000 0011
	0	1	-	-	0100 0011
	0	0	1	-	0010 w011
	0	0	0	1	0001 w011
	0	0	0	0 1	0000 1011

**Explanation:**

# Each bit is the logical AND of the corresponding bit in the source masks and the signaling flags.  
 - Don't care.  
 w Bit 4 of the signaling flag.

Figure 20-28. DXC for an IIS event

When no IIS event is recognized, the contents of the source operand are placed in the FPC register, except that the flags in the FPC register are set to the logical OR of the signaling flags and the source flags.

See Figure 2 for a detailed description of the result of this instruction.

	Resulting FPC Register Contents				
	Masks	Flags	DXC	DRM	BRM
IIS Event	S	OR	See Figure 20-28	S	S
No IIS Event	S	OR	S	S	S

**Explanation:**

BRM BFP rounding mode.  
 DRM DFP rounding mode.  
 DXC Data-exception code.  
 IIS Event IEEE-interruption-simulation event. When the logical AND of signaling flags and source masks is nonzero.  
 No IIS Event When the logical AND of signaling flags and source masks is zero.  
 OR Set to the logical OR of signaling flags and source flags.  
 S Set to the contents of the corresponding field in the source operand.

Figure 20-29. Result: LFAS and SFASR

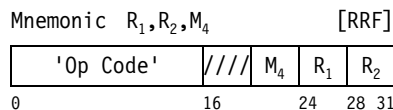
Bits in the source operand that correspond to unsupported bit positions in the FPC register must be zero; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Access (fetch, operand 2)
- Data with DXC 3, DFP instruction
- Data with DXC for IIS event
- Operation (if the DFP facility is not installed)
- Specification

## LOAD LENGTHENED



Mnemonic	Op Code	Operands
LDETR	'B3D4'	Short DFP source, long DFP target
LXDTR	'B3DC'	Long DFP source, extended DFP target

The second operand is extended to a longer format, and the result is placed at the first-operand location. The sign of the result is the same as the sign of the second operand. The ideal exponent is the exponent of the second operand.

Bit 0 of the  $M_4$  field ( $M_{4,0}$ ) is the Suppression of Invalid Operation bit. Bits 1-3 are reserved for future extensions and are ignored by the machine.

If the second operand is a finite number, the biased exponent of the second operand is converted to the corresponding biased exponent in the result format, and the coefficient is re-encoded and extended by appending zeros on the left.

If the second operand is an infinity, and  $M_{4,0}$  is zero, the result is a default infinity for the target format. Otherwise, if  $M_{4,0}$  is one, the result is the propagated infinity. Digits in the source coefficient-continuation field are re-encoded with sufficient zeros appended on the left and the target biased-exponent-continuation field (BXCF) is set to all zeros.

If the second operand is a QNaN, the result is the propagated QNaN. Digits in the source coefficient-continuation field are re-encoded with sufficient zeros appended on the left and the target BXCF is set to all zeros.

If the second operand is an SNaN and  $M_{4,0}$  is zero, an invalid-operation exception condition is recognized; if the invalid-operation-exception mask bit is zero, the result is the corresponding QNaN for the target format. Otherwise, if  $M_{4,0}$  is one, the IEEE-invalid-operation condition is not recognized, the invalid-op status flag is not set, and the result is the propagated SNaN. Digits in the source coefficient-continuation field are re-encoded with sufficient zeros appended on the left. The leftmost bit in the target BXCF is set to one, and all other bits in the BXCF are set to zero.

See Figure 20-26 on page 20-29 for a detailed description of the results of this instruction.

For LXDTTR, the  $R_1$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

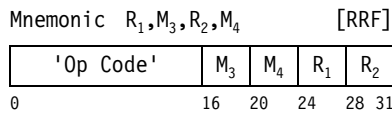
**IEEE Exception Conditions:**

- Invalid operation ( $M_{4,0}=0$  only)

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (LXDTR only)

**LOAD ROUNDED**



Mnemonic	Op Code	Operands
LEDTR	'B3D5'	Long DFP source, short DFP target
LDXTR	'B3DD'	Extended DFP source, long DFP target

The second operand, in the format of the source, is rounded to the precision of the target, and the result is placed at the first-operand location. The sign of the result is the same as the sign of the second operand. The ideal exponent is the exponent of the second operand.

The rounding mode is specified by the modifier in the  $M_3$  field:

**$M_3$  Rounding Method**

- 0 According to the current DFP rounding mode
- 1-7 Reserved
- 8 Round to nearest with ties to even
- 9 Round toward 0
- 10 Round toward  $+\infty$
- 11 Round toward  $-\infty$
- 12 Round to nearest with ties away from 0
- 13 Round to nearest with ties toward 0
- 14 Round away from 0
- 15 Round to prepare for shorter precision

When the  $M_3$  field contains zero, rounding is controlled by the current DFP rounding mode specified in the FPC register. When bit 0 of the  $M_3$  field is one, rounding is performed as specified by the modifier, regardless of the current DFP rounding mode. Rounding for modifiers 8-15 is the same as for rounding modes 0-7 (binary 000-111), respectively.

Bit 0 of the  $M_4$  field ( $M_{4,0}$ ) is the Suppression of Invalid Operation bit. Bits 1-3 are reserved for future extensions and are ignored by the machine.

If the second operand is a finite number, the biased exponent of the second operand is converted to the corresponding biased exponent in the result format. The source coefficient is rounded to the precision of the target according to the current DFP rounding mode. Normally, the result is in the format and length of the target. However, when an IEEE overflow or underflow occurs and the corresponding mask bit is one, the operation is completed by producing a wrapped rounded result in the same format and length as the source but rounded to the precision of the target.

If the second operand is an infinity, and  $M_{4,0}$  is zero, the result is a default infinity for the target format. Otherwise, if  $M_{4,0}$  is one, the result is the propagated infinity. Leftmost digits in the source coefficient-continuation field are removed, and the remaining digits of the field are re-encoded. The contents of the target biased-exponent-continuation field (BXCF) are set to zeros.

If the second operand is a QNaN, the result is the propagated QNaN. The appropriate number of leftmost digits in the source coefficient-continuation field are removed, and the remaining digits of the field are re-encoded. The target BXCF is set to zeros.

If the second operand is an SNaN, and  $M_{4,0}$  is zero, an invalid-operation exception condition is recognized; if the invalid-operation-exception mask bit is zero, the result is the corresponding QNaN for the target format. The appropriate number of leftmost digits in the source coefficient-continuation field are removed, and the remaining digits of the field are re-encoded. The target BXCF is set to zeros.

If the second operand as an SNaN, and  $M_{4,0}$  is one, the IEEE-invalid-operation condition is not recognized, the invalid-op status flag is not set, and the result is the propagated SNaN. The appropriate number of leftmost digits in the source coefficient-continuation field are removed, and the remaining digits of the field are re-encoded. The leftmost bit in the target BXCF is set to one, and all other bits in the BXCF are set to zero.

See Figure 20-26 on page 20-29 for a detailed description of the results of this instruction.

For LDXTTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:**

- Invalid operation ( $M_{4,0}=0$  only)
- Overflow
- Underflow
- Inexact

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (LDXTTR only)

## MULTIPLY

Mnemonic  $R_1, R_2, R_3$  [RRR]

'Op Code'	$R_3$	///	$R_1$	$R_2$
0	16	20	24	28 31

Mnemonic	Op Code	Operands
MDTR	'B3D0'	Long DFP
MXTR	'B3D8'	Extended DFP

The product of the second operand (the multiplicand) and the third operand (the multiplier) is placed at the first-operand location. The ideal exponent is the sum of the exponent of the two source operands.

If both source operands are numeric and finite, they are multiplied to form an intermediate product. The intermediate product is rounded to the target-format precision according to the current DFP rounding mode.

The sign of the product, if the product is numeric, is the exclusive or of the operand signs. This includes the sign of a zero or infinite product.

If one source operand is a zero and the other is an infinity, an IEEE-invalid-operation condition is recognized.

See Figure 20-30 on page 20-34 for a detailed description of the results of this instruction.

For MXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

Multiplicand (b) is	Results for MULTIPLY (b • c) when multiplier (c) is				
	0	Fn	$\infty$	QNaN	SNaN
0	S(b • c)	S(b • c)	X <sub>i</sub> : T(dNaN)	P(c)	X <sub>i</sub> : U(c)
Fn	S(b • c)	S(b • c)	S(dINF)	P(c)	X <sub>i</sub> : U(c)
$\infty$	X <sub>i</sub> : T(dNaN)	S(dINF)	S(dINF)	P(c)	X <sub>i</sub> : U(c)
QNaN	P(b)	P(b)	P(b)	P(b)	X <sub>i</sub> : U(c)
SNaN	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)	X <sub>i</sub> : U(b)

**Explanation:**

b • c The value b multiplied by c, rounded to the target-format precision and returned in the appropriate form. (See Figure 20-10 on page 20-13.)

dINF Default infinity.

dNaN Default quiet NaN.

Fn Finite nonzero number (includes both normal and subnormal numbers).

P(x) The QNaN of operand x is propagated and placed at the first operand location.

S(x) The value x is placed at the first operand location with the sign set to the exclusive or of the operand signs.

T(x) The value x is placed at the first operand location.

U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.

X<sub>i</sub>: y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-30. Results: MULTIPLY

**IEEE Exception Conditions:**

- Invalid operation
- Overflow
- Underflow
- Inexact

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (MXTR only)

**QUANTIZE**

Mnemonic R<sub>1</sub>, R<sub>3</sub>, R<sub>2</sub>, M<sub>4</sub> [RRF]

'Op Code'	R <sub>3</sub>	M <sub>4</sub>	R <sub>1</sub>	R <sub>2</sub>
0	16	20	24	28 31

Mnemonic	Op Code	Operands
QADTR	'B3F5'	Long DFP
QAXTR	'B3FD'	Extended DFP

The DFP third operand is converted and rounded to the form with the same exponent as that of the DFP second operand using the designated DFP rounding mode, and the result is placed at the first-operand location. The sign of the result is the same as the sign of the third operand. The ideal exponent is the specified target exponent.

The designated DFP rounding mode is specified by the modifier in the M<sub>4</sub> field:

**M<sub>4</sub> Rounding Method**

- 0 According to the current DFP rounding mode
- 1-7 Reserved
- 8 Round to nearest with ties to even
- 9 Round toward 0
- 10 Round toward + $\infty$
- 11 Round toward - $\infty$
- 12 Round to nearest with ties away from 0
- 13 Round to nearest with ties toward 0
- 14 Round away from 0
- 15 Round to prepare for shorter precision

When the M<sub>4</sub> field contains zero, rounding is controlled by the current DFP rounding mode specified in the FPC register. When bit 0 of the M<sub>4</sub> field is one, rounding is performed as specified by the modifier, regardless of the current DFP rounding mode. Rounding for modifiers 8-15 is the same as for rounding modes 0-7 (binary 000-111), respectively.

When the value of the third operand is greater than  $(10^p - 1) \times 10^{X_b}$ , where p is the format precision and X<sub>b</sub> is the exponent of the second operand, an invalid-operation exception condition is recognized.

When the delivered result differs in value from the third operand, an inexact exception is recognized. No underflow exception is recognized by this operation, regardless of the value of the third operand.

See Figure 20-31 on page 20-35 for a detailed description of the results of this instruction.

When second operand (b) is	Result for QUANTIZE when third operand (c) is				
	0	Fn	$\infty$	QNaN	SNaN
0	*	*	$X_i$ : T(dNaN)	P(c)	$X_i$ : U(c)
Fn	*	*	$X_i$ : T(dNaN)	P(c)	$X_i$ : U(c)
$\infty$	$X_i$ : T(dNaN)	$X_i$ : T(dNaN)	T(dINF)	P(c)	$X_i$ : U(c)
QNaN	P(b)	P(b)	P(b)	P(b)	$X_i$ : U(c)
SNaN	$X_i$ : U(b)	$X_i$ : U(b)	$X_i$ : U(b)	$X_i$ : U(b)	$X_i$ : U(b)

**Explanation:**

\* See Figure 20-32 on page 20-35.

dINF Default infinity.

dNaN Default quiet NaN.

Fn Finite nonzero number (includes both normal and subnormal numbers).

P(x) The QNaN of operand x is propagated and placed at the first operand location.

T(x) The value x is placed at the first operand location.

U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.

$X_i$ : y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-31. Results: QUANTIZE

Results for QUANTIZE when third operand (c) is			
		0	Fn
$X_b < X_c$	$V_c > (10^P - 1) \times 10^{X_b}$	X(0)	$X_i$ : T(dNaN)
	$V_c \leq (10^P - 1) \times 10^{X_b}$	X(0)	L(c)
$X_b = X_c$		X(0)	W(c)
$X_b > X_c$		X(0)	QR(c)

**Explanation:**

dNaN Default quiet NaN.

X(0) The value of zero with the exponent value  $X_b$  placed at the first operand location.

$X_b$  The exponent of b.

$X_c$  The exponent of c.

Fn Finite nonzero number (includes both normal and subnormal numbers).

L(x) The operand x is converted to the form with the exponent value  $X_b$ .

P The precision of the format.

QR(x) The operand x is rounded to the result of the form with the exponent value  $X_b$  based on the specified rounding mode. The result of that form is placed at the first operand location.

$V_c$  The value of the third operand (c).

W(x) The value and the form of operand x is placed at the first operand location.

$X_i$ : y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-32. Results: QUANTIZE (part 2)

For QAXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:**

- Invalid operation
- Inexact

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (QAXTR only)

**REROUND**

Mnemonic  $R_1, R_3, R_2, M_4$  [RRF]

'Op Code'	$R_3$	$M_4$	$R_1$	$R_2$
0	16	20	24	28 31

Mnemonic	Op Code	Operands
RRDTR	'B3F7'	Long DFP
RRXTR	'B3FF'	Extended DFP

The DFP third operand is rounded to the result in the form that has the same or fewer number of significant digits specified by the requested significance in the second operand based on the rounding mode specified by the modifier in the  $M_4$  field, and the result is placed at the first-operand location.

When the DFP third operand is a finite number, and if the requested significance is zero, or if the requested significance is nonzero and the number of significant digits of the third operand is less than or equal to the requested significance, then the value in the form of the third operand is placed at the first-operand location. If the requested significance is nonzero and the number of significant digits of the third operand is greater than the requested significance, then the third operand is converted and rounded to the requested significance based on the designated rounding mode. The result in the form with the requested significance is placed at the first-operand location. The sign of the result is the same as the sign of the third operand.

The ideal exponent is the greater value of the exponent of the third operand and the referenced exponent. The referenced exponent is the resultant exponent if the third operand would have been converted and rounded to the number of significant digits specified in the referenced significance based on the specified rounding mode.

For this instruction, the number of significant digits of the value 0 is considered to be zero.

The second operand is located in the general register designated by  $R_2$ . The requested-significance field, located in bits 58-63 of the second operand, is an unsigned binary integer that specifies the number of significant digits of the target operand. Bits 0-57 of the second operand are ignored.

The designated DFP rounding mode is specified by the modifier in the  $M_4$  field:

#### **$M_4$ Rounding Method**

- 0 According to the current DFP rounding mode
- 1-7 Reserved
- 8 Round to nearest with ties to even
- 9 Round toward 0
- 10 Round toward  $+\infty$
- 11 Round toward  $-\infty$
- 12 Round to nearest with ties away from 0
- 13 Round to nearest with ties toward 0
- 14 Round away from 0
- 15 Round to prepare for shorter precision

When the  $M_4$  field contains zero, rounding is controlled by the current DFP rounding mode specified in the FPC register. When bit 0 of the  $M_4$  field is one, rounding is performed as specified by the modifier, regardless of the current DFP rounding mode. Rounding for modifiers 8-15 is the same as for rounding modes 0-7 (binary 000-111), respectively.

If the rounded result with unbounded range is greater than  $(10^k-1) \times 10^{x_{\max}}$ , where  $k$  is the requested significance, an invalid operation exception condition occurs. When the invalid-operation exception occurs, and if the exception is disabled, a default QNaN is returned. When an invalid-operation exception condition occurs, no inexact exception condition is recognized.

In the absence of an invalid-operation exception, if the result differs in value from the third operand, an inexact exception condition is recognized.

This operation does not cause either an overflow or an underflow exception.

See Figure 20-33 on page 20-37 for a detailed description of the results of this instruction.

For RRXTR, the  $R_1$  and  $R_3$  fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

#### **IEEE Exception Conditions:**

- Invalid operation
- Inexact

#### **Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (RRXTR only)

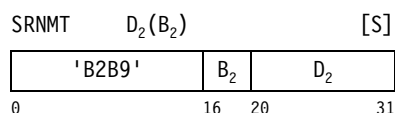
	Results for REROUND when the third operand (c) is				
	0	Fn	$\infty$	QNaN	SNaN
<b>k <math>\neq</math> 0 AND k &lt; m</b>	-	r or X <sub>i</sub> ; T(dNaN)	T(dINF)	P(c)	X <sub>i</sub> ; U(c)
<b>k <math>\neq</math> 0 AND k = m</b>	-	W(c)	T(dINF)	P(c)	X <sub>i</sub> ; U(c)
<b>(k <math>\neq</math> 0 AND k &gt; m) OR k = 0</b>	W(c)	W(c)	T(dINF)	P(c)	X <sub>i</sub> ; U(c)

**Explanation:**

- Not applicable.  
dINF Default infinity.  
dNaN Default quiet NaN.  
Fn Finite nonzero numbers (includes both normal and subnormal numbers).  
k Requested significance, specifies the number of significant digits in the first operand.  
m Number of significant digits in the third operand.  
P(x) The QNaN of operand x is propagated and placed at the first operand location.  
r The rounded result. If  $r \leq (10^k - 1) \times 10^{x_{\max}}$ , then r is returned; otherwise, an invalid-operation exception is recognized.  
T(x) The value x is placed at the first operand location.  
U(x) The SNaN of operand x is converted to the corresponding QNaN and placed at the first operand location.  
W(x) The value and the form of x is placed at the first operand location.  
X<sub>i</sub>; y IEEE invalid-operation exception. The results (y) are produced only when FPC 0.0 is zero.

Figure 20-33. Results: REROUND

## SET DFP ROUNDING MODE



The DFP rounding-mode bits are set from the second-operand address.

The second-operand address is not used to address data; instead, the DFP rounding-mode bits in the FPC register are set with bits 61-63 of the address.

Bits other than 61-63 of the second-operand address are ignored.

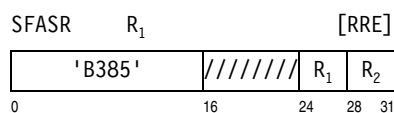
**Condition Code:** The code remains unchanged.

**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)

## SET FPC AND SIGNAL



The format of the four-byte source operand is treated the same as that of the floating-point control (FPC) register. When any signaling flag (the flag in the original FPC register) is one and the corresponding source mask is also one, an IEEE-interruption-simulation (IIS) event is recognized, which causes a specific content to be placed in the FPC register and a data-exception program interruption at completion of the instruction execution. Otherwise, a different content is placed in the FPC register, and instruction execution completes without recognizing an IIS event.

The source operand is in bits 32-63 of the general register designated by R<sub>1</sub>; bits 0-31 of the general register are ignored.

When an IIS event is recognized, the source masks, the source DFP rounding mode, and the source BFP rounding mode are placed in the corresponding fields of the FPC register. The flags in the FPC register are set to the logical OR of the signaling flags and the source flags. The DXC for the interruption is shown in Figure 20-28 on page 20-31.

When no IIS event is recognized, the contents of the source operand are placed in the FPC register, except that the flags in the FPC register are set to the logical OR of the signaling flags and the source flags.

See Figure 20-29 on page 20-31 for a detailed description of the result of this instruction.

Bits in the source operand that correspond to unsupported bit positions in the FPC register must be zero; otherwise, a specification exception is recognized.

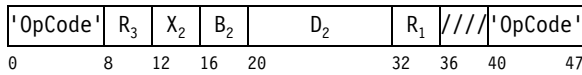
**Condition Code:** The code remains unchanged.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IIS event
- Operation (if the DFP facility is not installed)
- Specification

## SHIFT COEFFICIENT LEFT

Mnemonic  $R_1, R_3, D_2(X_2, B_2)$  [RXF]



Mnemonic	Op Code	Operands
SLDT	'ED40'	Long DFP
SLXT	'ED48'	Extended DFP

The coefficient of the DFP third operand is shifted left the number of digits specified by the second-operand address, and the result is placed at the first-operand location. For a NaN or infinity, all coefficient digits are in the coefficient-continuation field. Digits shifted out of the leftmost digit are lost. Zeros are supplied to the vacated positions on the right. The sign of the result is the same as the sign of the third operand.

When the third operand is a finite number, the exponent of the result is the same as the third operand. When the third operand is an infinity or QNaN, zeros are placed into the biased-exponent-continuation field of the result. When the third operand is an SNaN, bit 0 of the biased-exponent-continuation field of the result is set to 1 and zeros are placed into the rest of the bits.

The second-operand address is not used to address data; its rightmost six bits indicate the number of digits to be shifted. The remainder of the address is ignored.

For SLXT, the  $R_1$  and  $R_3$  fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

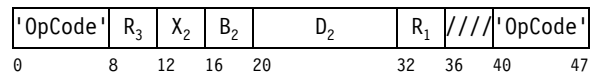
**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (SLXT only)

## SHIFT COEFFICIENT RIGHT

Mnemonic  $R_1, R_3, D_2(X_2, B_2)$  [RXF]



Mnemonic	Op Code	Operands
SRDT	'ED41'	Long DFP
SRXT	'ED49'	Extended DFP

The coefficient of the DFP third operand is shifted right the number of digits specified by the second-operand address, and the result is placed at the first-operand location. For a NaN or infinity, all coefficient digits are in the coefficient-continuation field. Digits shifted out of the units digit are lost. Zeros are supplied to the vacated positions on the left. The sign of the result is the same as the sign of the third operand.

When the third operand is a finite number, the exponent of the result is the same as the third operand. When the third operand is an infinity or QNaN, zeros are placed into the biased-exponent-continuation field of the result. When the third operand is an SNaN, bit 0 of the biased-exponent-continuation field of the result is set to 1 and zeros are placed into the rest of the bits.

The second-operand address is not used to address data; its rightmost six bits indicate the number of digits to be shifted. The remainder of the address is ignored.

For SRXT, the  $R_1$  and  $R_3$  fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Condition Code:** The code remains unchanged.

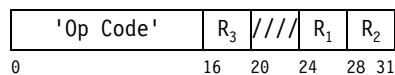
**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (SRXT only)

## SUBTRACT

Mnemonic  $R_1, R_2, R_3$  [RRR]



Mnemonic	Op Code	Operands
SDTR	'B3D3'	Long DFP
SXTR	'B3DB'	Extended DFP

The third operand is subtracted from the second operand, and the difference is placed at the first-operand location. The ideal exponent is the lesser exponent value of the two source operands.

The execution of SUBTRACT is identical to that of ADD, except that the third operand participates in the operation with its sign bit inverted. See Figure 20-14 on page 20-19 for the detailed results of ADD.

For SXTR, the R fields must designate valid floating-point-register pairs; otherwise, a specification exception is recognized.

**Resulting Condition Code:**

- 0 Result is zero
- 1 Result is less than zero
- 2 Result is greater than zero
- 3 Result is a NaN

**IEEE Exception Conditions:**

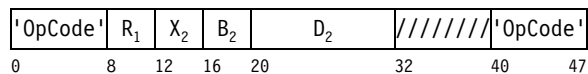
- Invalid operation
- Overflow
- Underflow
- Inexact

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Data with DXC for IEEE exception
- Operation (if the DFP facility is not installed)
- Specification (SXTR only)

## TEST DATA CLASS

Mnemonic  $R_1, D_2 (X_2, B_2)$  [RXE]



Mnemonic	Op Code	Operands
TDCET	'ED50'	Short DFP
TDCDT	'ED54'	Long DFP
TDCXT	'ED58'	Extended DFP

The class and sign of the first operand are examined to select one bit from the second-operand address. Condition code 0 or 1 is set according to whether the selected bit is zero or one, respectively.

The second-operand address is not used to address data; instead, the rightmost 12 bits of the address, bits 52-63, are used to specify 12 combinations of operand class and sign. Bits 0-51 of the second-operand address are ignored.

As shown in Figure 20-34 on page 20-39, DFP operands are divided into six classes: zero, subnormal, normal, infinity, quiet NaN, and signaling NaN:

DFP operand class	Bit used when sign is	
	+	-
Zero	52	53
Subnormal	54	55
Normal	56	57
Infinity	58	59
Quiet NaN	60	61
Signaling NaN	62	63

Figure 20-34. Second-Operand-Address Bits for TEST DATA CLASS

One or more of the second-operand-address bits may be set to one. If the second-operand-address bit corresponding to the class and sign of the first operand is one, condition code 1 is set; otherwise, condition code 0 is set.

Operands, including SNaNs and QNaNs, are examined without causing an IEEE exception.

For TDCXT, the  $R_1$  field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Resulting Condition Code:**

- 0 Selected bit is 0 (no match)
- 1 Selected bit is 1 (match)
- 2 --
- 3 --

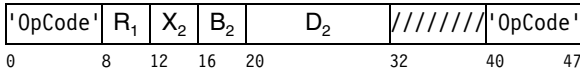
**IEEE Exception Conditions:** None.

**Program Exceptions:**

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (TDCXT only)

**TEST DATA GROUP**

Mnemonic  $R_1, D_2(X_2, B_2)$  [RXE]



Mnemonic	Op Code	Operands
TDGET	'ED51'	Short DFP
TDGDT	'ED55'	Long DFP
TDGXT	'ED59'	Extended DFP

The group and sign of the first operand are examined to select one bit from the second-operand address. Condition code 0 or 1 is set according to whether the selected bit is zero or one, respectively.

The second-operand address is not used to address data; instead, the rightmost 12 bits of the address, bits 52-63, are used to specify 12 combinations of operand group and sign. Bits 0-51 of the second-operand address are ignored.

TEST DATA GROUP is used to determine whether a finite number is safe. A finite number is safe if the exponent is neither maximum nor minimum, and the leftmost coefficient digit is zero.

Figure 20-35 shows the data groups and the bit assignment. There are six data groups: safe zero, zero with extreme exponent, nonzero with extreme exponent, safe nonzero, nonzero leftmost coefficient digit with nonextreme exponent, and special. The special group is defined for infinity and NaN. Depending on the model, subnormal with nonextreme exponent may be placed in the nonzero-with-extreme-exponent group or the safe-nonzero group.

DFP Operand	Exponent	LMD	Data Group	Bit used when sign is	
				+	-
Zero	Nonextreme	z <sup>1</sup>	Safe zero	52	53
	Extreme	z <sup>1</sup>	Zero with extreme exponent	54	55
Finite nonzero	Extreme	-	Nonzero with extreme exponent	56	57
	Nonextreme	z	Safe nonzero	58	59
	Nonextreme	nz	Nonzero leftmost coefficient digit with nonextreme exponent	60	61
Infinity or NaN	na	na	Special	62	63

**Explanation:**

-	The result does not depend on this condition.
1	This condition is true by virtue of the condition to the left of this column.
Extreme	Maximum exponent, X <sub>max</sub> , or minimum exponent, X <sub>min</sub> .
Nonextreme	X <sub>max</sub> < exponent < X <sub>min</sub> .
LMD	Leftmost coefficient digit.
na	Not applicable.
nz	Nonzero.
z	Zero.

Figure 20-35. Second-Operand-Address Bits for TEST DATA GROUP

One or more of the second-operand-address bits may be set to one. If the second-operand-address bit corresponding to the group and sign of the first operand is one, condition code 1 is set; otherwise, condition code 0 is set.

Operands, including SNaNs and QNaNs, are examined without causing an IEEE exception.

For TDGXT, the R<sub>1</sub> field must designate a valid floating-point-register pair; otherwise, a specification exception is recognized.

**Resulting Condition Code:**

- 0 Selected bit is 0 (no match)
- 1 Selected bit is 1 (match)
- 2 --
- 3 --

**IEEE Exception Conditions:** None.

***Program Exceptions:***

- Data with DXC 3, DFP instruction
- Operation (if the DFP facility is not installed)
- Specification (TDGXT only)

---

**End of Document**

