z/Architecture

The Load-Program-Parameter and the CPU-Measurement Facilities

SA23-2260-06

Note:

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

Seventh Edition (September, 2019)

This edition obsoletes and replaces *The Load-Program-Parameter and the CPU-Measurement Facilities*, SA23-2260-05.

The facilities discussed in this publication are 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.

Notices

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

Produced in the United States of America
September, 2019

The information in this document is intended to provide guidance for those implementing CPU-measurement 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

The first part of this document defines the LOAD PROGRAM PARAMETER instruction in the z/Architecture architectural mode. The facility is a prerequisite for the CPU-measurement sampling facility.

The second part of this document defines the CPU-measurement facility in the z/Architecture architectural mode. The facility consists of two parts: a CPU-measurement counter facility and a CPU-measurement sampling facility. The former provides a means to measure activities in the CPU and some shared peripheral processors; the latter provides a means to take a snapshot of the CPU at a specified sampling interval. The CPU-measurement sampling facility requires the load-program-parameter facility as a prerequisite. Each part consists of a number of instructions and several events of an external-interruption subclass. Measurement data provided by the CPU-measurement facility is intended for software performance tuning.

Summary of Changes in Seventh Edition

The seventh edition of this publication differs from the previous edition principally as follows:

1. Counter numbers 80-83 are defined for the crypto-activity counter set when the counter second version number (CSVN) is six.
2. The maximum number of counters in the extended counter set is increased to 160 when the counter second version number (CSVN) is six.
3. Format codes 8005 hex and 8006 hex are defined for the diagnostic-sampling data entry (DSDE).
4. A reserved for IBM use (RIBM) field is added to the information block of QUERY SAMPLING INFORMATION.

Summary of Changes in Sixth Edition

The sixth edition of this publication differs from the previous edition principally as follows:

1. A counter first version number (CFVN) equal to three is defined. Models supporting a CFVN equal to three provide a different set of counters in the problem-state counter set than models supporting a CFVN equal to one. Models supporting a CFVN equal to three provide the same set of counters in the basic counter set as models supporting a CFVN equal to one.
2. A configuration level (CL) field and a host indicator (H) field are added to the basic-sampling data entry (BSDE).
3. Format codes 8002 hex, 8003 hex, and 8004 hex are defined for the diagnostic-sampling data entry (DSDE).
4. A basic-sampling data entry size (BSDES) field and a diagnostic-sampling data entry size (DSDES) field are added to the trailer entry of a sample-data block.

Summary of Changes in Fifth Edition

The fifth edition of this publication differs from the previous edition principally as follows:

1. Added many clarifications due to the introduction of multithreading.
2. Added MT-diagnostic counter set.
3. A new loss-of-MT-counter-data alert is added.
4. A new STORE CPU COUNTER MULTIPLE instruction is added to store a selection of counters within a counter set.

Summary of Changes in Fourth Edition

The fourth edition of this publication differs from the previous edition principally as follows:

1. The maximum number of extended counters is increased to 128 for counter second version numbers (CSVN) greater than two.
2. The range of defined extended counter numbers are contiguous beginning with counter number 128 to the last defined extended counter.
3. Condition code 3 is set for EXTRACT CPU-COUNTER when a counter number is specified in the instruction that is greater than the last defined extended counter number. CC3 indicates there are no defined extended counter numbers equal to greater than the counter number specified in the instruction.
4. The following Programming Note is inserted as Programming Note 1 in the Programming Notes for the LOAD PERIPHERAL-COUNTER SET CONTROLS instruction:

   "1. When the coprocessor-group counter sets and the controls for the coprocessor-group counter sets are not installed, condition code 3 is set for the instruction."

Summary of Changes in Third Edition

The third edition of this publication differs from the previous edition principally as follows:

1. A counter second version number equal to two is added to that increases the number of counters provided in the CPU extended counter set from 32 to 48 for counter second version number two.
2. A Programming Note is added to the sections "Cyrypto-Activity Counter Set" and Coprocessor-Group Counter Set" to state that the Galois-field (GF) multiplier function may be provided in by the DEA/AES coprocessor and, if so, the AES counters include counts for functions that use the GF multiplier.
3. A Programming Note is added to the section "Problem-State Counter Set" to state that when the problem-state counters are active for a guest 1 and a guest 2 is running, collection of the guest 1 problem-state counters is performed based on the guest 2 problem state (i.e., the guest 1 problem-state counters are collected when the guest 2 is in the problem state).
4. The definition of the Sampling Interval Register is modified to indicate it is an approximation of the average number of CPU cycles within each sampling interval.
5. The definition of the Trailer Entries is modified to include a timestamp format bit to indicate whether the format of the timestamp in the Trailer Entry is in STCK or STCKE format.
6. The definition of the Coprocessor-Group Counter Set is modified to specify that the counters are not cleared to zeros when the counter set is put into the unauthorized state for a CPU.

Summary of Changes in Second Edition

The second edition of this publication differs from the previous edition principally as follows:

7. The set-program-parameter facility is renamed to the load-program-parameter facility. Correspondingly, the instruction SET PROGRAM PARAMETER (SPP) is renamed LOAD PROGRAM PARAMETER (LPP).
8. The instruction SET CPU-COUNTER-SET CONTROLS (SCCTL) is renamed LOAD CPU-COUNTER-SET CONTROLS (LCCTL).
9. The instruction SET PERIPHERAL-COUNTER-SET CONTROLS (SPCTL) is renamed LOAD PERIPHERAL-COUNTER-SET CONTROLS (LPCTL).
10. The instruction SET SAMPLING CONTROLS (SSCTL) is renamed LOAD SAMPLING CONTROLS (LSCTL).
11. The instruction diagram for LOAD CPU-COUNTER-SET COUNTERS is corrected to represent an S-format instruction.

12. The mnemonic for the EXTRACT COPROCESSOR-GROUP ADDRESS instruction is changed to ECPGA.

13. Activation controls are defined to be cleared to zeros by CPU reset.

14. The definition of the coprocessor-group address is changed, and a 16-bit maximum coprocessor-group address is added; the value may be inspected by executing the QUERY COUNTER INFORMATION instruction.

15. The coprocessor-group-address-change indicator is added to the global counter-set-state control register; the value may be inspected by executing the QUERY COUNTER INFORMATION instruction.

16. The loss-of-counter-data alert is changed to loss-of-CPU-counter-data alert and is no longer applicable to the coprocessor-group counters.

17. The definition of the sampling interval register is changed so that it to specifies the average number of CPU cycles within each sampling interval.

Section 1. The Load-Program-Parameter Facility

Introduction

This architecture defines the load-program-parameter facility which is available in the z/Architecture architecture mode. When the facility is installed on a CPU, the same facility is installed on all CPUs in the configuration.

The facility consists of the LOAD PROGRAM PARAMETER instruction, and one 64-bit program-parameter register. The contents of this register are cleared to zeros by initial CPU reset, clear reset, or power-on reset.

This facility is a prerequisite for the CPU-measurement sampling facility.

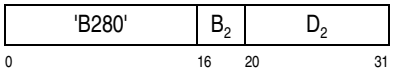
Facility Indication

Bit 40 of the list of facility bits provided by the STORE FACILITY LIST EXTENDED instruction indicates the installation status of the load-program-parameter facility. When the bit is one, the facility is installed in the z/Architecture architectural mode; otherwise, the facility is not installed.

Instruction

LOAD PROGRAM PARAMETER

LPP $D_2(B_2)$ [S]



The eight-byte program parameter in storage locations designated by the second-operand address is placed in the program-parameter register.

The program parameter is placed in the program-parameter register only when at least one sampling function is enabled (that is, the enable control is one) at this configuration level or the level below. When none of the sampling functions is enabled at any of these two configuration levels, the instruction is executed as a no-operation.

Condition Code: The code remains unchanged.

Program Exceptions:

- Access (fetch, operand 2)
- Privileged operation
- Operation (if the load-program-parameter facility is not installed)

Programming Notes:

1. There are three configuration levels: the basic machine level, the logical partition level, and the virtual machine level. The basic machine level is one level below the logical partition level; the logical partition level is one level below the virtual machine level.
2. z/VM may enable a sampling function for collecting measurement data for all guests. In this case, the sampling function is enabled at the logical partition level but not at the virtual machine level. Guests running at the virtual machine level would not be able to detect whether a sampling function is enabled at the level below. To support this case, a guest operating system can always issue this instruction. When no one enables any sampling function, the instruction execution is fast and is a no-operation.

Section 2. The CPU-Measurement Facility

Introduction

This architecture defines the CPU-measurement facility (CPUMF) which is available in the z/Architecture architectural mode. When the facility is installed, the same facility is installed on all CPUs in the configuration.

The CPU-measurement facility consists of two parts: a CPU-measurement counter facility and a CPU-measurement sampling facility. The former provides a means to measure activities in the CPU and some shared peripheral processors; the latter provides a means to take a snapshot of the CPU at a specified sampling interval and requires the LOAD PROGRAM PARAMETER facility as a prerequisite. Each part consists of a number of instructions to set and extract controls or measurement data, and consists of several events of an external-interruption subclass.

The external-interruption subclass, called the measurement-alert subclass, and a subclass mask bit, bit 58 of control register 0, are provided when either the CPU-measurement counter facility or the CPU-measurement sampling facility is installed.

Measurement data provided by the CPU-measurement counter facility or the CPU-measurement sampling facility is intended for statistical estimation of performance characteristics. Measurement data is only substantially accurate, and may not be repeatable. For example, it is unpredictable if the instruction count counts an instruction that caused an exception or counts the target of EXECUTE. Also, certain system internal activities may be included in measurement data.

Measurement data provided on one model may not be provided on future models. Also, future models may provide additional measurement data. Two counter version numbers are defined to indicate supported measurement data for counters; two format codes are defined to indicate supported measurement data for sampling.

Facility Indication

Bits 67, 68 and 142 of the list of facility bits provided by the STORE FACILITY LIST EXTENDED instruction indicate the installation status of the CPU-measurement counter facility, the CPU-measurement sampling facility, and the store-CPU-counter-multiple facility respectively. When one of these bits is one, the corresponding facility is installed in the z/Architecture architectural mode; otherwise, the facility is not installed. The CPU-measurement counter-multiple facility may only be installed when the CPU-measurement counter facility is installed.

Programming Note: The CPU-measurement counter facility and the CPU-measurement sampling facility should be disabled when they are not being used.

CPU-Measurement Counter Facility

The CPU-measurement counter facility on each CPU consists of a local counter-set-state control register, a number of counters, several external-interruption events, a measurement-counter-extraction-authorization control, and a number of instructions. The following summarizes the instructions:

- EXTRACT COPROCESSOR-GROUP ADDRESS
- EXTRACT CPU COUNTER
- EXTRACT PERIPHERAL COUNTER
- LOAD CPU-COUNTER-SET CONTROLS
- LOAD PERIPHERAL-COUNTER-SET CONTROLS
- QUERY COUNTER INFORMATION
- SET CPU COUNTER
- SET PERIPHERAL COUNTER

If the Store CPU counter multiple facility is installed the following instruction is included:

- STORE CPU COUNTER MULTIPLE

The measurement-counter-extraction-authorization control, bit 15 of control register 0, specifies whether selected CPU counter contents can be extracted in the problem state. The bit is initialized to zero.

The external-interruption events include a counter-authorization-change alert, a loss-of-MT-counter-data alert, and a loss-of-CPU-counter-data alert. These events are part of an external-interruption subclass, called the measurement-alert subclass. A subclass-mask bit, bit 58 of control register 0, is also provided. The bit is initialized to one, which enables the interruption.

The facility provides three kinds of counters: one is local to the CPU, one is local to a core, and another is global to all CPUs. The counters local to the CPU are used to count logical CPU activities and are grouped into several CPU counter sets. The counters local to a core count events on the core, which may include one or more CPUs. The global counters are used to count activities of peripheral processors, which are shared among certain CPUs, and are grouped into different peripheral counter sets. All counters are 64 bits. A carry out of bit position 0 of any counter is ignored.

When any peripheral counter set is provided, the facility also includes a global counter-set-state control register.

The number and the type of installed counters are model dependent. Two 16-bit counter version numbers, called counter first version number and counter second version number, are provided by QUERY COUNTER INFORMATION; each version number identifies installed counters in certain counter sets.

When there is a change to the meaning of a counter or the number of installed counters in any counter set specified by a counter version number, a different value is indicated by that counter version number. Both counter version numbers start at one and increase consecutively.

Counter Sets

The facility provides the following CPU counter sets on each CPU:

- Basic counter set
- Problem-state counter set
- Crypto-activity counter set
- Extended counter set
- MT-diagnostic counter set

Most counters in the CPU counter sets count CPU activities on the logical CPU basis; some may count activities on a logical core basis. The facility may also provide one kind of peripheral counter sets, called coprocessor-group counter sets. When coprocessor-group counter sets are installed one coprocessor-group counter set is provided for each coprocessor group. Counters in a coprocessor-group counter set count activities of each coprocessor in the group at the basic machine level. The LOAD PERIPHERAL-COUNTER SET CONTROLS instruction may be used to determine whether the coprocessor-group counter sets are installed (see the “Programming Notes” for LPCTL on page 2-29).

At the basic machine level, a coprocessor group consists of a number of coprocessors and may be attached to one or more CPUs. When authorized, all CPUs in the model have access to all coprocessor-group counter sets and the global counter-set-state control register.

The counter first version number (CFVN) identifies installed counters in the basic counter set and problem-state counter set; the counter second version number (CSVN) identifies installed counters in the crypto-activity counter set, the extended counter set, the MT-diagnostic counter set, and the coprocessor-group counter sets.

Programming Notes:

1. Enabling each counter set causes some system overhead. The program should enable only the counter sets for collecting needed measurement data.
2. The coprocessor-group counter set can be used to estimate the utilization of each coprocessor in the group and the blocking effect on each coprocessor in the group.
3. The crypto-activity counter set can be used to identify the crypto activities contributed by the logical CPU and the blocking effects on the logical CPU.

Local Counter-Set-State Control Register

On each CPU, the CPU-measurement counter facility provides a local counter-set-state control register which contains a one-bit authorization control, a one-

bit enable control, and a one-bit activation control for each CPU counter set, and also contains a one-bit authorization control for all coprocessor-group counter sets. When multi-threading is not active, except for authorization controls, the contents of the local counter-set-state control register are cleared to zeros by initial CPU reset, clear reset, or power-on reset. Activation controls are also cleared to zeros by CPU reset. When multi-threading is active, the above happens except that the enable and activation controls for the MT-diagnostic counter set are only cleared to zeros by an action which disables multi-threading. Authorization controls are set or reset by an external means.

Global Counter-Set-State Control Register

On each model, when the coprocessor-group counter sets are installed, the CPU-measurement counter facility provides a three-bit global counter-set-state control register, which contains a one-bit enable control, a one-bit activation control, and a one-bit coprocessor-group-address-change indicator. The enable or activation control provides control of all coprocessor-group counter sets in the model. The coprocessor-group-address-change indicator, when set, indicates that a potential change of any coprocessor-group address has occurred.

When the enable control in the global counter-set-state control register is one and a potential change of any coprocessor-group address occurs, the coprocessor-group-address-change indicator is set to one.

When the enable control in the global counter-set-state control register is changed from one to zero, the coprocessor-group-address-change indicator is set to zero. The indicator remains zero as long as the enable control is zero.

State of Counter Set

When the CPU is in the operating state, each CPU counter set can be in one of the following four states: unauthorized, disabled, inactive, and active.

At any time, all coprocessor-group counter sets are in the same state with respect to a CPU and can be in one of those four states. For CPUs that are authorized, all coprocessor-group counter sets are in the disabled, inactive, or active state, depending on the setting of the global enable control and the global activation control; and, for CPUs that are unauthorized, all coprocessor-group counter sets are in the unauthorized state.

Unauthorized: When a counter set is in the unauthorized state, counters in the set cannot be used and do not increment. An external means is provided to authorize or unauthorize the use of each counter set.

Disabled: When a counter set is in the disabled state, the counter set is authorized for use but the program has not enabled the counter set yet. In this state, counters in the counter set do not increment and the counter contents cannot be extracted or set.

Inactive: When a counter set is in the inactive state, the counter set is authorized, enabled, and deactivated. In this state, counters in the counter set do not increment; the counter contents remain unchanged and can be extracted or set.

Active: When a counter set is in the active state, the counter set is authorized, enabled, and activated. In this state, each counter in the set increments for the defined activity; the counter contents can be extracted or set.

The unauthorized state and the disabled state are collectively called the not-enabled state.

When the CPU enters the stopped state from the operating state, active CPU counter sets are stopped. When the CPU enters the operating state from the stopped state, CPU counter sets resume the states they were in when they were last stopped.

The following table summarizes the definition of valid states in terms of state-control bits for each counter set.

State Control			
A	**E**	**C**	**Counter-Set State**
0	0	0	Unauthorized
1	0	0	Disabled
1	1	0	Inactive
1	1	1	Active

Explanation:

A Authorization control
E Enable control
C Activation control
All other combinations of A, E, and C are invalid.

Figure 2-1. Counter-Set State and State Control

State Transition

The following table summarizes actions that cause state transition of a counter set.

From	To			
	Unauthorized	**Disabled**	**Inactive**	**Active**
Unauthorized	—*	External control	N/A for CPU counter sets; external control for coprocessor-group counter sets.	N/A for CPU counter sets; external control for coprocessor-group counter sets.
Disabled	External control	—*	Enabled & deactivated by LCCTL or LPCTL.	Enabled & activated by LCCTL or LPCTL[3].
Inactive	External control	Disabled by LCCTL or LPCTL, or by reset[1].	—*	Activated by LCCTL or LPCTL
Active	External control	Disabled by LCCTL or LPCTL, or reset[1,4].	Deactivated by LCCTL or LPCTL, CPU reset, or by errors[2].	—*

Explanation:

1	All CPU counter sets are disabled and contents in these sets are cleared to zeros by initial CPU reset, clear reset, or power-on reset.
2	Each active non-MT counter set is deactivated by a loss-of-CPU-counter-data alert.
3	When a Set Multi-threading SIGNAL PROCESSOR order is issued with an accepted non-zero parameter the MT-diagnostic counter set is set to the active state.
4	The MT-diagnostic counter set is set to the disabled state only when and only when multi-threading is disabled.
*	When a counter set is in the unauthorized, disabled, inactive, or active state, if execution of LCCTL or LPCTL sets the state controls to the same state as the original state, the state controls are considered successfully set.
—	No action required.
LCCTL	The LOAD CPU-COUNTER-SET CONTROLS instruction.
LPCTL	The LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction.
N/A	Not applicable – this state transition cannot occur.

Figure 2-2. Counter Set State Transition

Programming Notes:

1. The external control that turns on or off the authorization control of coprocessor-group counter sets for a CPU may cause the counter sets to change from the unauthorized state to the disabled, inactive, or active state, depending on the setting of the global enable and activation controls.
2. Since all CPUs have access to the global enable and activation controls and have access to the counters of every coprocessor-group counter set, the task that sets up controls and collects measurement data for coprocessor-group counter sets can run on any CPU.

CPU Counter Sets

All counters in a CPU counter set are incremented for the specific activities only when the CPU is in the operating state and the counter set is active. An external means may be provided to authorize or unauthorize the use of each counter set.

All cycle-count counters in the CPU counter sets are incremented for the specific activities by one for each CPU cycle.

The contents of all counters in a set are cleared to zeros when the set is disabled or unauthorized. Each CPU counter set can be enabled or disabled, and activated or deactivated by executing LOAD CPU-COUNTER-SET CONTROLS with the exception of the MT-diagnostic counter set.

In the following section, cache hierarchy is described in terms of cache level. The smaller the cache level number, the faster the cache access time. The term "*directory write*" means the cache directory is updated because either a cache line is brought in for cache miss or the state of a cache line is changed. Note that some cache misses may not result in a directory write. For example, cache misses caused by incorrect branch prediction may not result in a directory write if the branch is resolved in time.

For cache level 1, separate counters are provided for measuring instruction-cache activities and data-cache activities. When a model has a unified instruction and data cache at level 1, cache level-1 activities are measured in the instruction-cache counters and the contents of the data-cache counters are set to zeros.

The value of a cache-penalty cycle count divided by the value of the corresponding cache-directory-write count gives the average cache penalty per cache directory write.

Basic Counter Set

All counters in this counter set count the specified activities, regardless of whether the CPU is in the problem state or the supervisor state. The counters are defined for the following counter first version numbers.

Counter First Version Number (CFVN) = 1:

When the counter first version number (CFVN) is one, the following defines the installed counters in the basic counter set:

Cycle Count: This counter counts the total number of CPU cycles, excluding the number of cycles while the CPU is in the wait state.

Instruction Count: This counter counts the total number of instructions executed by the CPU.

Level-1 I-Cache Directory Write Count: This counter counts the total number of level-1 instruction-cache or unified-cache directory writes.

Level-1 I-Cache Penalty Cycle Count: This counter counts the total number of cache penalty cycles for level-1 instruction cache or unified cache.

Level-1 D-Cache Directory Write Count: This counter counts the total number of level-1 data-cache directory writes.

Level-1 D-Cache Penalty Cycle Count: This counter counts the total number of cache penalty cycles for level-1 data cache.

Figure 2-3 summarizes the counters in the basic counter set for CFVN = 1.

Counter number	Counter
0	Cycle count
1	Instruction count
2	Level-1 I-cache directory-write count
3	Level-1 I-cache penalty cycle count
4	Level-1 D-cache directory-write count
5	Level-1 D-cache penalty cycle count
6-31	Reserved

Figure 2-3. Counters in the Basic Counter Set for CFVN=1

Counter First Version Number (CFVN) = 3:

When the counter first version number (CFVN) is three, the following defines the installed counters in the basic counter set:

Cycle Count: This counter counts the total number of CPU cycles, excluding the number of cycles while the CPU is in the wait state.

Instruction Count: This counter counts the total number of instructions executed by the CPU.

Level-1 I-Cache Directory Write Count: This counter counts the total number of level-1 instruction-cache or unified-cache directory writes.

Level-1 I-Cache Penalty Cycle Count: This counter counts the total number of cache penalty cycles for level-1 instruction cache or unified cache.

Level-1 D-Cache Directory Write Count: This counter counts the total number of level-1 data-cache directory writes.

Level-1 D-Cache Penalty Cycle Count: This counter counts the total number of cache penalty cycles for level-1 data cache.

Figure 2-4 summarizes the counters in the basic counter set for CFVN = 3.

Counter number	Counter
0	Cycle count
1	Instruction count
2	Level-1 I-cache directory-write count
3	Level-1 I-cache penalty cycle count
4	Level-1 D-cache directory-write count
5	Level-1 D-cache penalty cycle count
6-31	Reserved

Figure 2-4. Counters in the Basic Counter Set for CFVN=3

Problem-State Counter Set

All counters in this counter set count the specified activities only when the CPU is in the problem state. The counters are defined for the following counter first version numbers.

Counter First Version Number (CFVN) = 1:

When the counter first version number (CFVN) is one, the following defines the installed counters in the problem-state counter set:

Problem-State Cycle Count: This counter counts the total number of CPU cycles when the CPU is in the problem state, excluding the number of cycles while the CPU is in the wait state.

Problem-State Instruction Count: This counter counts the total number of instructions executed by the CPU while in the problem state.

Problem-State Level-1 I-Cache Directory Write Count: This counter counts the total number of level-1 instruction-cache or unified-cache directory writes while the CPU is in the problem state.

Problem-State Level-1 I-Cache Penalty Cycle Count: This counter counts the total number of penalty cycles for level-1 instruction cache or unified cache while the CPU is in the problem state.

Problem-State Level-1 D-Cache Directory Write Count: This counter counts the total number of level-1 data-cache directory writes while the CPU is in the problem state.

Problem-State Level-1 D-Cache Penalty Cycle Count: This counter counts the total number of penalty cycles for level-1 data cache while the CPU is in the problem state.

Figure 2-5 summarizes the counters in the problem-state counter set for CFVN = 1.

Counter number	Counter
32	Problem-state cycle count
33	Problem-state instruction count
34	Problem-state level-1 I-cache directory-write count
35	Problem-state level-1 I-cache penalty cycle count
36	Problem-state level-1 D-cache directory-write count
37	Problem-state level-1 D-cache penalty cycle count
38-63	Reserved

Figure 2-5. Counters in the Problem-State Counter Set for CFVN = 1

Counter First Version Number (CFVN) = 3:

When the counter first version number (CFVN) is three, the following defines the installed counters in the problem-state counter set:

Problem-State Cycle Count: This counter counts the total number of CPU cycles when the CPU is in the problem state, excluding the number of cycles while the CPU is in the wait state.

Problem-State Instruction Count: This counter counts the total number of instructions executed by the CPU while in the problem state.

Figure 2-6 summarizes the counters in the problem-state counter set for CFVN = 3.

Counter number	Counter
32	Problem-state cycle count
33	Problem-state instruction count
34-63	Reserved

Figure 2-6. Counters in the Problem-State Counter Set for CFVN = 3

Programming Notes:

1. On some models, activation of the problem-state counter set may slow down execution of instructions that update the problem-state bit (P) of PSW.

2. When the problem-state counters are active for a guest 1 and a guest 2 is running, collection of the problem-state counters is performed based on the guest 2 problem state (i.e., the guest 1 problem-state counters are collected when the guest 2 is in the problem state).

Crypto-Activity Counter Set

All counters in this counter set count the specified activities of a coprocessor group contributed by this logical CPU. The counters are defined for the following counter second version numbers.

Counter Second Version Number (CSVN) = 1, 2, 3, 4, or 5:

When the counter second version number (CSVN) is one, two, three, four, or five, the following defines the installed counters in the crypto-activity counter set:

PRNG Function Count: This counter counts the total number of the pseudorandom-number-generation functions issued by the CPU.

PRNG Cycle Count: This counter counts the total number of CPU cycles when the DEA/AES/SHA coprocessor is busy performing the pseudorandom-number-generation functions issued by the CPU.

PRNG Blocked Function Count: This counter counts the total number of the pseudorandom-number-generation functions that are issued by the CPU and are blocked because the DEA/AES/SHA coprocessor is busy performing a function issued by another CPU.

PRNG Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the pseudorandom-number-generation functions issued by the CPU because the DEA/AES/SHA coprocessor is busy performing a function issued by another CPU.

SHA Function Count: This counter counts the total number of the SHA functions issued by the CPU.

SHA Cycle Count: This counter counts the total number of CPU cycles when the SHA coprocessor is busy performing the SHA functions issued by the CPU.

SHA Blocked Function Count: This counter counts the total number of the SHA functions that are issued by the CPU and are blocked because the SHA coprocessor is busy performing a function issued by another CPU.

SHA Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the SHA functions issued by the CPU because the SHA coprocessor is busy performing a function issued by another CPU.

DEA Function Count: This counter counts the total number of the DEA functions issued by the CPU.

DEA Cycle Count: This counter counts the total number of CPU cycles when the DEA/AES coprocessor is busy performing the DEA functions issued by the CPU.

DEA Blocked Function Count: This counter counts the total number of the DEA functions that are issued by the CPU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another CPU.

DEA Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the DEA functions issued by the CPU because the DEA/AES coprocessor is busy performing a function issued by another CPU.

AES Function Count: This counter counts the total number of the AES functions issued by the CPU.

AES Cycle Count: This counter counts the total number of CPU cycles when the DEA/AES coprocessor is busy performing the AES functions issued by the CPU.

AES Blocked Function Count: This counter counts the total number of the AES functions that are issued by the CPU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another CPU.

AES Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the AES functions issued by the CPU because the DEA/AES coprocessor is busy performing a function issued by another CPU.

Figure 2-7 summarizes the counters in the crypto-activity counter set.

Counter number	Counter
64	PRNG function count
65	PRNG cycle count
66	PRNG blocked function count
67	PRNG blocked cycle count
68	SHA function count
69	SHA cycle count
70	SHA blocked function count
71	SHA blocked cycle count
72	DEA function count
73	DEA cycle count
74	DEA blocked function count
75	DEA blocked cycle count
76	AES function count
77	AES cycle count
78	AES blocked function count
79	AES blocked cycle count
80-127	Reserved

Figure 2-7. Counters in the crypto-activity counter set for CSVN = 1, 2, 3, 4, or 5

Counter Second Version Number (CSVN) = 6:

When the counter second version number (CSVN) is six, the following defines the installed counters in the crypto-activity counter set:

PRNG Function Count: This counter counts the total number of the pseudorandom-number-generation functions issued by the CPU.

PRNG Cycle Count: This counter counts the total number of CPU cycles when the DEA/AES/SHA coprocessor is busy performing the pseudorandom-number-generation functions issued by the CPU.

PRNG Blocked Function Count: This counter counts the total number of the pseudorandom-number-generation functions that are issued by the CPU and are blocked because the DEA/AES/SHA coprocessor is busy performing a function issued by another CPU.

PRNG Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the pseudorandom-number-generation functions issued by the CPU because the DEA/AES/SHA coprocessor is busy performing a function issued by another CPU.

SHA Function Count: This counter counts the total number of the SHA functions issued by the CPU.

SHA Cycle Count: This counter counts the total number of CPU cycles when the SHA coprocessor is busy performing the SHA functions issued by the CPU.

SHA Blocked Function Count: This counter counts the total number of the SHA functions that are issued by the CPU and are blocked because the SHA coprocessor is busy performing a function issued by another CPU.

SHA Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the SHA functions issued by the CPU because the SHA coprocessor is busy performing a function issued by another CPU.

DEA Function Count: This counter counts the total number of the DEA functions issued by the CPU.

DEA Cycle Count: This counter counts the total number of CPU cycles when the DEA/AES coprocessor is busy performing the DEA functions issued by the CPU.

DEA Blocked Function Count: This counter counts the total number of the DEA functions that are issued by the CPU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another CPU.

DEA Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the DEA functions issued by the CPU because the DEA/AES coprocessor is busy performing a function issued by another CPU.

AES Function Count: This counter counts the total number of the AES functions issued by the CPU.

AES Cycle Count: This counter counts the total number of CPU cycles when the DEA/AES coproces-

sor is busy performing the AES functions issued by the CPU.

AES Blocked Function Count: This counter counts the total number of the AES functions that are issued by the CPU and are blocked because the DEA/AES coprocessor is busy performing a function issued by another CPU.

AES Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the AES functions issued by the CPU because the DEA/AES coprocessor is busy performing a function issued by another CPU.

ECC Function Count: This counter counts the total number of the elliptic-curve cryptography (ECC) functions issued by the CPU.

ECC Cycle Count: This counter counts the total number of CPU cycles when the ECC coprocessor is busy performing the elliptic-curve cryptography (ECC) functions issued by the CPU.

ECC Blocked Function Count: This counter counts the total number of the elliptic-curve cryptography (ECC) functions that are issued by the CPU and are blocked because the ECC coprocessor is busy performing a function issued by another CPU.

ECC Blocked Cycle Count: This counter counts the total number of CPU cycles blocked for the elliptic-curve cryptography (ECC) functions issued by the CPU because the ECC coprocessor is busy performing a function issued by another CPU.

Figure 2-8 summarizes the counters in the crypto-activity counter set when the CSVN is 6.

Counter number	Counter
64	PRNG function count
65	PRNG cycle count
66	PRNG blocked function count
67	PRNG blocked cycle count
68	SHA function count
69	SHA cycle count
70	SHA blocked function count
71	SHA blocked cycle count
72	DEA function count
73	DEA cycle count
74	DEA blocked function count
75	DEA blocked cycle count
76	AES function count
77	AES cycle count
78	AES blocked function count
79	AES blocked cycle count
80	ECC function count
81	ECC cycle count
82	ECC blocked function count
83	ECC blocked cycle count
84-127	Reserved

Figure 2-8. Counters in the crypto-activity counter set for CSVN = 6

Programming Notes:

1. When the Galois-field (GF) multiplier function is provided by the DEA/AES coprocessor, the counts of functions involving the GF multiplier are accumulated in the AES counters in the crypto-activity counter set. The following GF Multiplier counts are included in the corresponding AES counters:

   - GF Multiplier function count (in AES Function Count)
   - GF Multiplier cycle count (in AES Cycle Count)
   - GF Multiplier blocked function (in AES Blocked Function Count)
   - GF Multiplier blocked cycle (in AES Blocked Cycle Count)

2. The activity of both the CIPHER MESSAGE WITH CHAINING instruction's PRNG function and the PERFORM PSEUDORANDOM NUMBER OPERATION instruction are counted in the PRNG counters, regardless of the hashing algorithm used by the function.

Extended Counter Set

The number and the meaning of counters in this counter set are model-dependent and are not defined by this architecture. The system library publication for each model, titled *The CPU-Measurement Facility Extended Counter Definition*, provides information on these counters for the supported counter version number. The maximum number of counters that can be provided for each counter second version number is defined below:

- When the counter second version number is one, a maximum of 32 counters can be provided in this counter set and the first counter number is 128 and the maximum counter number is 159.
- When the counter second version number is two, a maximum of 48 counters can be provided in this counter set and the first counter number is 128 and the maximum counter number is 175.
- When the counter second version number is three, four, or five, a maximum of 128 counters can be provided in this counter set and the first counter number is 128 and the maximum counter number is 255.
- When the counter second version number is greater than five, a maximum of 160 counters can be provided in this counter set and the first counter number is 128 and the maximum counter number is 287.

The counter numbers must be assigned sequentially beginning with counter number 128 up to the maximum installed counter.

MT-diagnostic Counter Set

When the counter second version number is one, two, or three, the counters in this counter set are not installed.

When the counter second version number is greater than three, a maximum of 48 counters can be provided and the first counter number is 448. The counter numbers are assigned sequentially beginning with counter number 448 up to the maximum installed counter.

All counters in this counter set count the specified activities, regardless of whether the CPU is in the problem state or the supervisor state. All counters in this counter set count the specified activities of a core and the same values are reported to all CPUs of the core.

The counter set is enabled and activated on all CPUs in the configuration upon completion of an accepted SIGNAL PROCESSOR set multi-threading signal processor order with a non-zero parameter. When multi-threading is enabled, any action which causes multi-threading to become disabled will disable the counter set. The counters are monotonically increasing and will wrap to zero when they reach their maximum value. All of the counters in this counter set are read-only and no means is provided to set or clear these counters while the counters are activated.

Counter number	Counter
448	Cycle count with one thread active
449	Cycle count with two threads active
450-456	Reserved for IBM Use
457-495	Reserved

Figure 2-9. Counters in the MT-diagnostic counter set

Peripheral Counter Sets

Counters for one type of peripheral processor, the coprocessor, are provided and a counter set defined as the coprocessor-group counter set is provided for the coprocessor.

Coprocessor-Group Address

Each coprocessor group at the basic machine level has a 16-bit coprocessor-group address assigned. The address of the coprocessor group that is accessible by a CPU is determined by executing the EXTRACT COPROCESSOR-GROUP ADDRESS instruction. This CPU is identified by using the CPU address at the basic machine level.

For a model, the maximum address of all coprocessor groups at the basic machine level and the maximum address of all CPUs at the basic machine level are provided by executing the QUERY COUNTER INFORMATION instruction.

Coprocessor-Group Counter Set

All counters in a coprocessor-group counter set are incremented for the specific activities only when the counter set is active. An external means is provided to authorize or unauthorize the use of the counter set. At any time, all coprocessor-group counter sets are in the same state with respect to a CPU.

All cycle-count counters in the coprocessor-group counter set are incremented for the specific activities by one for each CPU cycle.

When the coprocessor-group counter sets are put into the disabled state, the contents of counters in all of the counter sets are cleared to zeros. All coprocessor-group counter sets are disabled by basic machine clear reset or power-on reset, or by executing LOAD PERIPHERAL-COUNTER-SET CONTROLS. The contents of the coprocessor-group counters are not cleared to zero or otherwise modified as the result of putting the coprocessor-group counter sets into the unauthorized state for a CPU.

The following defines the installed counters in a coprocessor-group counter set for all counter second version numbers:

SHA function count: This counter counts the total number of the SHA functions performed by the SHA coprocessor.

SHA cycle count: This counter counts the total number of CPU cycles when the SHA coprocessor is busy performing SHA functions.

SHA blocked function count: This counter counts the total number of SHA functions that are blocked because the coprocessor is busy.

SHA blocked cycle count: This counter counts the total number of CPU cycles blocked from all attached CPUs because the SHA coprocessor is busy.

DEA/AES/MAC function count: This counter counts the total number of the DEA/AES/MAC functions performed by the DEA/AES coprocessor.

DEA/AES/MAC cycle count: This counter counts the total number of CPU cycles when the DEA/AES coprocessor is busy performing DEA/AES/MAC functions.

DEA/AES/MAC blocked function count: This counter counts the total number of DEA/AES/MAC functions that are blocked because the DEA/AES coprocessor is busy.

DEA/AES/MAC blocked cycle count: This counter counts the total number of CPU cycles blocked from all attached CPUs because the DEA/AES coprocessor is busy.

Figure 2-10 summarizes the counters in the coprocessor-group counter set.

Counter number	Counter
0	SHA function count
1	SHA cycle count
2	SHA blocked function count
3	SHA blocked cycle count
4	DEA/AES/MAC function count
5	DEA/AES/MAC cycle count
6	DEA/AES/MAC blocked function count
7	DEA/AES/MAC blocked cycle count
8-63	Reserved

Figure 2-10. Counters in the coprocessor-group counter set

Programming Note: When the Galois-field (GF) multiplier function is provided by the DEA/AES coprocessor, the counts of functions involving the GF multiplier are accumulated in the DEA/AES/MAC counters in the coprocessor-group counter set. The following GF Multiplier counts are included in the corresponding DEA/AES/MAC counters:

- GF Multiplier function count (in DEA/AES/MAC Function Count)
- GF Multiplier cycle count (in DEA/AES/MAC Count)
- GF Multiplier blocked function (in DEA/AES/MAC Blocked Function Count)
- GF Multiplier blocked cycle (in DEA/AES/MAC Blocked Cycle Count)

CPU-Measurement Sampling Facility

The CPU-measurement sampling facility on each CPU consists of two sampling functions, several sampling control registers, several external-interruption events, and two instructions. The following summarizes the instructions:

- LOAD SAMPLING CONTROLS
- QUERY SAMPLING INFORMATION

The facility provides a means to take a snapshot of the logical CPU at a specified sampling interval, which is a processing time interval as seen by the CPU. Each snapshot produces a set of sample data, which includes the instruction address of an instruction being executed and some state information about the CPU. The CPU-measurement sampling facility requires the LOAD PROGRAM PARAMETER facility as a prerequisite.

The external-interruption events include an invalid-entry-address alert, an incorrect-sample-data-block-table-entry alert, a program-request alert, a sampling-authorization-change alert, and a loss-of-sample-data alert. These events are part of an external-interruption subclass, called the measurement-alert subclass. A subclass-mask bit, bit 58 of control register 0, is also provided. This bit is initialized to one, which enables the interruption.

The facility includes two sampling functions: basic sampling and diagnostic sampling. They both use the same sampling control registers, the same sampling buffer structure, the same external interruption events, and the same instructions. The main difference between these two functions is the sample data.

The sample-data size and format for each sampling function are model dependent, and are determined by a 16-bit data-entry-format code. which is stored in each sample data. For a data-entry-format code, the sample-data size and format of the basic-sampling function are defined in this document; the sample-data size of the diagnostic-sampling function is defined in this document, but the sample-data format is not defined here. The data-entry-format code starts with 0001 (hex) for the basic-sampling data entry; the code starts with 8001 (hex) for the diagnostic-sampling data entry. Both data-entry-format codes continue consecutively. When either the size, the meaning, or the format of a sample data is different from the previous model, the appropriate data-entry-format code is incremented.

The sample data provided by the basic-sampling function is not included in the sample data provided by the diagnostic-sampling function. To get meaningful diagnostic-sampling data, both sampling functions shall be activated.

The state of each sampling function can be individually set by executing LOAD SAMPLING CONTROLS. Both sampling functions are disabled by initial CPU reset, clear reset, or power-on reset.

Note: The diagnostic-sampling function is not intended for ordinary use; it can be authorized by IBM service Personnel.

Sampling Control Registers

The CPU-measurement sampling facility provides a number of sampling control registers. Except for authorization controls, the contents of these control registers are cleared to zeros by initial CPU reset, clear reset, or power-on reset; and are also cleared to zeros by executing LOAD SAMPLING CONTROLS that disables all sampling functions. Activation controls are also cleared to zeros by CPU reset. Authorization controls are set or reset by an external means.

Table-Entry-Address Register (TEAR)

The table-entry-address register is 64 bits, and contains the address of the current sample-data-block-table entry. It is unpredictable whether the address is real or absolute.

Data-Entry-Address Register (DEAR)

The data-entry-address register is 64 bits, and contains the address of the next sample-data-block data entry. It is unpredictable whether the address is real or absolute.

Maximum-Buffer-Size Indicator

The maximum-buffer-size indicator is one bit. When the indicator is zero, the maximum size of all sample-data-block tables and the exact size of all sample-data blocks are 4K bytes. When the indicator is one, the maximum size of all sample-data-block tables

and the exact size of all sample-data blocks are 1M bytes.

Sampling-Function-State Control Register

The sampling-function-state control register is six bits. Three bits are assigned to the basic-sampling function and another three to the diagnostic-sampling function. For each sampling function, the three state-control bits are: authorization control (A), enable control (E), and activation control (C).

Sampling Interval Register

The sampling interval register is 64 bits. The contents of the register approximate the average number of CPU cycles within each sampling interval.

Host indicator

The host indicator is one bit and is available to CPUs at the logical partition level. When the CPU is running at the logical partition level and if a sampling function is active, the host indicator, when zero, specifies that the contents of the program-parameter register are stored into the guest-program-parameter field of the sample-data blocks by the sample-data-block update process; the host indicator, when one, specifies that the contents of the program-parameter register are stored into the host-program-parameter field.

State of Sampling Function

When the CPU is in the operating state, a sampling function can be in any of the following four states: unauthorized, disabled, inactive, and active.

Unauthorized: When a sampling function is in the unauthorized state, the function cannot be used and no sample data is stored. An external means is provided to authorize or unauthorize the use of the function.

Disabled: When a sampling function is in the disabled state, the function is authorized for use but the program has not enabled the function yet. In this state, no new sample data is stored, and the contents of the sample-data blocks remain unchanged, and no sampling control, except for authorization controls, is preserved.

Inactive: When a sampling function is in the inactive state, the function is authorized, enabled, and deactivated. In this state, no new sample data is stored, the contents of sample data blocks remain unchanged, and sampling controls are preserved and can be extracted.

Active: When a sampling function is in the active state, the function is authorized, enabled, and activated. In this state, new sampling data is stored during each sampling interval and sampling controls can be extracted.

The unauthorized state and the disabled state are collectively called the not-enabled state.

When the CPU enters the stopped state from the operating state, active sampling functions are stopped. When the CPU enters the operating state from the stopped state, sampling functions resume the states they were in when they were last stopped.

Figure 2-11 summarizes the definition of valid states in terms of control bits for each sampling function.

State Control			Sampling-Function State
A	E	C	
0	0	0	Unauthorized
1	0	0	Disabled
1	1	0	Inactive
1	1	1	Active

Explanation:

A Authorization control
E Enable control
C Activation control
All other combinations of A, E, and C are invalid.

Figure 2-11. Sampling-Function State and State Control

State Transition

Figure 2-12 summarizes actions that cause state transition of a sampling function.

From	To			
	Unauthorized	Disabled	Inactive	Active
Unauthorized	—*	External control	N/A	N/A
Disabled	External control	—*	Enabled & deactivated by LSCTL.	Enabled & activated by LSCTL.
Inactive	External control	Disabled by LSCTL, or by reset[1].	—*	Activated by LSCTL.
Active	External control	Disabled by LSCTL, or reset[1].	Deactivated by LSCTL, CPU reset, or by errors[2].	—*

Explanation:

[1] Each enabled sampling function is disabled by initial CPU reset, clear reset, or power-on reset.
2 Each active sampling function is deactivated by an invalid-entry-address alert, an incorrect-sample-data-block-table-entry alert, or a loss-of-sample-data alert.
* When a sampling function is in the unauthorized, disabled, inactive, or active state, if execution of LSCTL sets the state controls to the same state as the original state, the state controls are considered successfully set.
— No action required.
LSCTL The LOAD SAMPLING CONTROLS instruction.
N/A This state transition cannot occur.

Figure 2-12. Sampling Function State Transition

Sampling Buffer Structure

A number of sample-data blocks are allocated by the control program for the machine to store sample data during each sampling interval. Each sample-data block is designated by a block-link entry in a sample-data-block table. The current entry of the sample-data-block table is designated by the contents of the table-entry-address register; the next data entry of the sample-data block is designated by the contents of the data-entry-address register. Figure 2-13 shows the structure of sample-data-block tables and sample-data blocks.

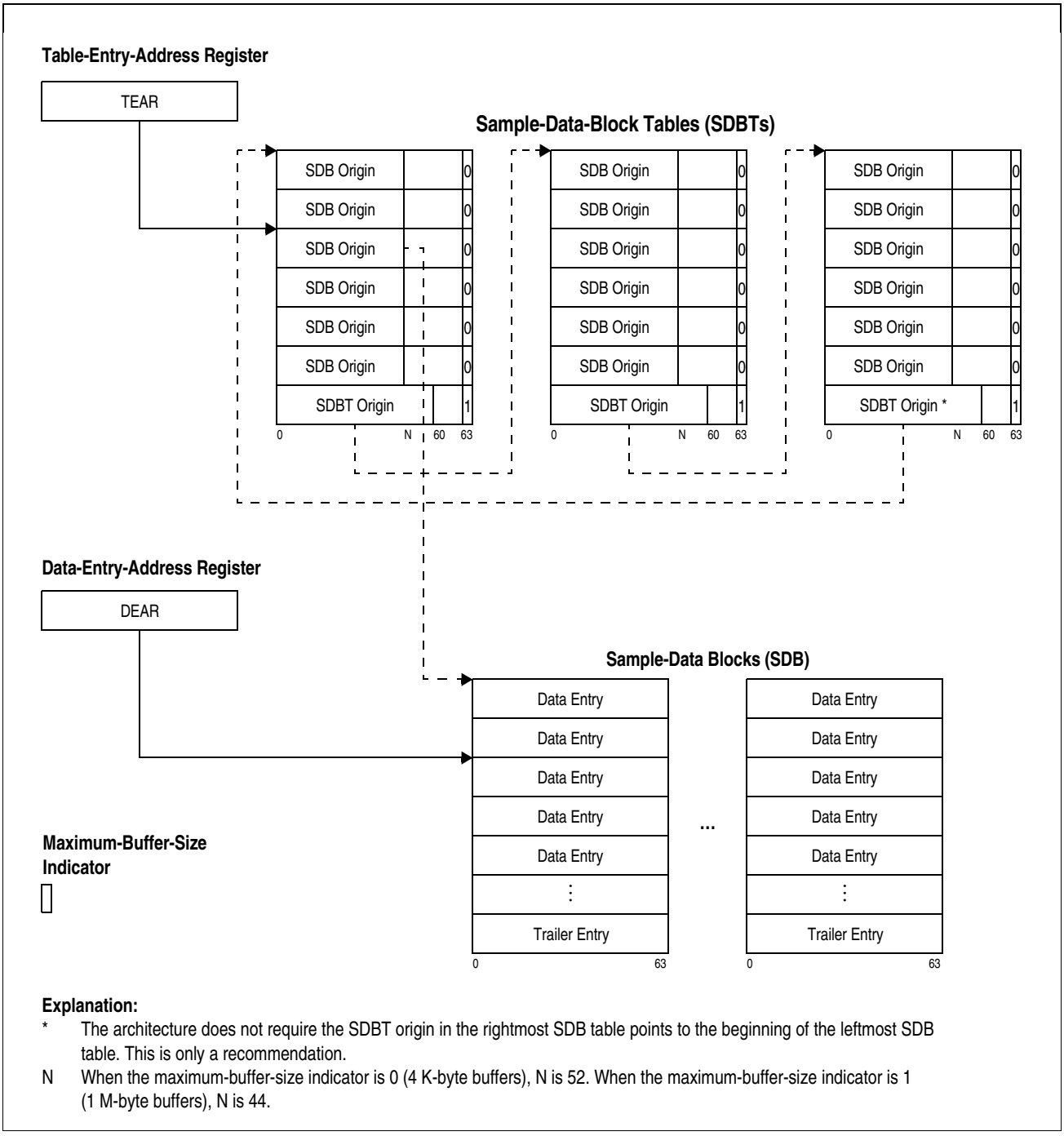


Figure 2-13. Sampling Buffer Structure

When the maximum-buffer-size indicator is zero, the maximum size for all sample-data-block tables and the exact size for all sample-data blocks are 4K bytes; the origin of all sample-data-block tables is at an integral boundary of 16 bytes, and the origin of all sample-data blocks is at an integral boundary of 4K bytes; none of sample-data-block tables or sample-data blocks crosses an integral boundary of 4K bytes.

When the maximum-buffer-size indicator is one, the maximum size for all sample-data-block tables and the exact size for all sample-data blocks are 1M (mega) bytes; the origin of all sample-data-block tables is at an integral boundary of 16 bytes, and the origin of all sample-data blocks is at an integral boundary of 1M bytes; none of sample-data-block tables or sample-data blocks crosses an integral boundary of 1M bytes.

Entries in sample-data-block (SDB) tables must remain unchanged during the period a sampling function is enabled, that is inactive or active. A change to any entry in a sampling-data-block table while a sampling function is enabled may cause sample data to be stored at unpredictable locations and may cause an invalid-entry-address external exception to be recognized for each modified entry.

Programming Note: A very small sampling buffer or sampling interval would severely impact system performance and significantly distort sample data. A reasonably large sampling buffer and sampling interval should be used.

Sample-Data-Block Table

There are two kinds of entry in each sample-data-block table: block-link entry and table-link entry. Each block-link entry contains a sample-data-block origin and the table-link entry contains a sample-data-block-table origin. Each entry is 8 bytes. Each sample-data-block table contains a number of block-link entries and one table-link entry. Bit 63 in each entry distinguishes a block-link entry from a table-link entry. When bit 63 is zero, the entry is a block-link entry; when bit 63 is one, the entry is a table-link entry.

Each sample-data-block table starts at an integral boundary of 16 bytes. A table-link entry is the last entry in a sample-data-block table. The actual size of a sample-data-block table is determined by the location of the table-link entry, and does not exceed the maximum buffer size. The origin and the table-link entry of a sample-data-block table cannot be separated by an integral boundary of the maximum-buffer size.

Block-Link Entry

Each block-link entry is 8 bytes, and has one of the following two formats:

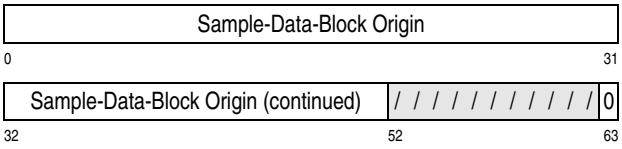


Figure 2-14. Block-Link Entry when the Maximum-Buffer-Size Indicator is 0

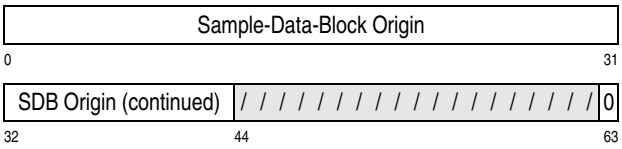


Figure 2-15. Block-Link Entry when the Maximum-Buffer-Size Indicator is 1

Sample-Data-Block (SDB) Origin : When the maximum-buffer-size indicator is zero, bits 0-51 of a block-link entry contain the origin of a sample-data block in real or absolute storage. Bits 52-62 of the entry are ignored during the sample-data-block update process. When the sample-data-block origin is to be placed in the data-entry-address register, it is appended with 12 zeros on the right to form a 64-bit address and the address is then placed in the register.

When the maximum-buffer-size indicator is one, bits 0-43 of a block-link entry contain the origin of a sample-data block in real or absolute storage. Bits 44-62 of the entry are ignored during the sample-data-block update process. When the sample-data-block origin is to be placed in the data-entry-address register, it is appended with 20 zeros on the right to form a 64-bit address and the address is then placed in the register.

Table-Link Entry

Each table-link entry is 8 bytes, and has the following format:

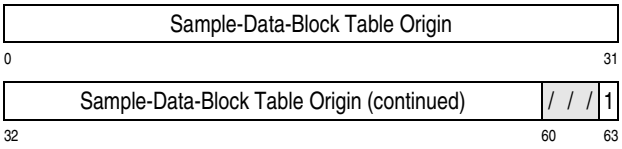


Figure 2-16. Table-Link Entry

Sample-Data-Block-Table (SDBT) Origin : Bits 0-59 of a table-link entry contain the origin of a sample-data-block table in real or absolute storage. Bits 60-62 of the entry are ignored during the sample-data-block update process. When the sample-data-block-table origin is to be placed in the table-entry-address register, it is appended with four zeros on the right to form a 64-bit address and the address is then placed in the register.

Sample-Data Block

Each sample-data block starts at an integral boundary of the maximum-buffer size. The size of a sample-data block is the maximum-buffer size.

There are two kinds of entry in each sample-data block: data entry and trailer entry. The last 64 bytes of a sample-data block form the trailer entry; all other space in the block are used to form data entries.

Data Entry

When at least one sampling function is active, a data entry is stored during each sampling interval. If only the basic-sampling function is active, the data entry stored is a basic-sampling data entry; if only the diagnostic-sampling function is active, the data entry stored is a diagnostic-sampling data entry. If both sampling functions are active, the data entry stored is a combined-sampling data entry.

Basic-Sampling Data Entry

Figure 2-17 shows the format and size of the basic-sampling data entry for a format code of 0001 (hex):

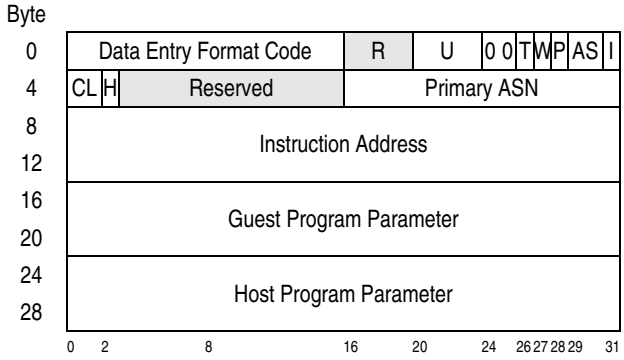


Figure 2-17. Format of the Basic-Sampling Data Entry

Data Entry Format Code: Bits 0-15 of the data entry contains the format code of the data entry.

Reserved (R): Bits 16-19 of the data entry are reserved for programming use. Zeros are stored in this field by the sample-data-block update process.

Number of Unique Instructions (U): Bits 20-23 of the data entry specify the number of unique, completed instructions that were executed simultaneously during the sampling cycle when the unique cycle indicator is on.

A pipelined CPU can execute multiple instructions concurrently in an overlapped fashion: each of these instructions being executed concurrently is in a different pipeline stage. Furthermore, on some models, each stage of a pipelined CPU may execute multiple instructions simultaneously.

During an instruction execution, a unique-cycle indicator is turned on for one cycle at the sampling point, that is the place in the CPU the sample data is taken from. The sampling point depends on the model but is the same for all instructions executed on the same model. For a pipelined CPU, the sampling point is usually a particular pipeline stage. Depending on the model, it is unpredictable when the unique-cycle indicator is turned on during an instruction execution. This field contains the number of instructions executed simultaneously at the sampling point when the unique-cycle indicator is on.

When a sampling occurs and if the sampling point is not busy because either the CPU is in the wait state or because of delay in some other pipeline stage, the contents of this field are set to zero.

The contents of this field can be used to estimate cycles per instruction when a sufficiently small sampling interval and an adequately larger number of samples are used.

The cycles per instruction for a particular measurement can be estimated by dividing the number of busy samples, that is samples with the wait-state bit being set to zero, by the total number of unique instructions in all busy samples.

DAT Mode (T): Bit 26 of the data entry contains the DAT mode bit in the PSW of the CPU.

Wait State (W): Bit 27 of the data entry contains the wait state bit in the PSW of the CPU.

Problem State (P): Bit 28 of the data entry contains the problem state bit in the PSW of the CPU.

Address-Space Control (AS): Bits 29-30 of the data entry contain the address-space control in the PSW of the CPU.

Invalid Indication (I): Bit 31 of the data entry indicates whether the entry is valid or invalid. When the bit is zero, the entry is valid; when the bit is one, the entry is invalid. An entry is set to invalid when sample data in the entry are not consistent.

Configuration Level (CL): Bits 32-33 of the data entry contain an unsigned binary integer used as a code to indicate the configuration level of the CPU during the sample cycle, as follows:

code	meaning
0	no configuration level specified
1	logical partition level
2	virtual machine level
3	reserved

Host Indicator (H): When the basic-sampling data entry (BSDE) is stored to sample-data blocks (SDB) established by a control program run at the logical partition level, bit 34 of the data entry contains the host indicator; otherwise bit 34 contains zero.

Reserved: Bits 35-47 of the data entry are reserved. Zeros are stored in this field by the sample-data-block update process.

Primary ASN: Bytes 6-7 of the data entry contain the primary ASN in bits 48-63 of control register 4 of the CPU.

Instruction Address: Bytes 8-15 of the data entry contain the instruction address of an instruction that the CPU was executing during the sampling cycle.

Instruction addresses are treated as real addresses in the real mode, as primary virtual addresses in the primary-address mode, secondary-space mode, or access-register mode, and as home virtual addresses in the home-space mode.

When the sampling point is executing multiple instructions simultaneously during the sampling cycle, only the address of one instruction among these simultaneously executed instructions is reported. The selection of which instruction address to be reported is model dependent.

On some models, the address of the target instruction of an execute type instruction is never reported in the sample data. When the wait state bit is one, the contents of this field are unpredictable. When a sampling occurs and if the sampling point is not executing any instruction because of delay in some other pipeline stage, it is unpredictable which address of the instructions being executed concurrently in the CPU is reported.

Guest Program Parameter: When sampling occurs, if the CPU is running at the virtual machine level, and if the host indicator is one at the logical partition level, bytes 16-23 of the data entry contain the program parameter most recently set by the CPU at the virtual machine level; if the host indicator is zero, the contents of bytes 16-23 are unpredictable.

When sampling occurs, if the CPU is running at the logical partition level with the host indicator being set to zero, bytes 16-23 of the data entry contain the program parameter most recently set by the CPU at the logical partition level; if the host indicator is one, the contents of bytes 16-23 are set to zeros.

Host Program Parameter: When sampling occurs, if the host indicator is one at the logical partition level, bytes 24-31 of the data entry contain the program parameter most recently set by the CPU at the logical partition level; if the host indicator is zero, the contents of bytes 24-31 are set to zeros.

Programming Notes:

1. Bit 19 of the basic-sampling data entry is intended for the operating system to indicate the maximum buffer size: 0 means 4K bytes, and 1 means 1M bytes.
2. The program parameter issued by a non-VM control program is always placed at the guest-program-parameter field in the basic-sampling data entry.
3. The guest program parameter can be used to identify the individual task that contributed to the sample data.
4. When VM is collecting sample data for guests, the host program parameter can be set by the VM control program to identify the individual guest that contributed to the sample data.
5. One may observe the wait-state (W) bit in sample data being set to one for dedicated CPUs, or for nondedicated CPUs in some unusual situations or when multi-threading is enabled. For nondedicated CPUs when all CPUs of a core are in wait state, time spent in the wait state is not considered part of the process time and, thus, the actual sampling interval may appear to be longer than the specified one.
6. The reserved code point of zero for the configuration level (CL) field definition in the basic-sampling data entry provides compatibility between various program levels and various machine models available before and after the introduction of the configuration level (CL). The compatibility permits maintaining the basic-sampling data entry format code of 0001 (hex) with the introduction of the configuration level (CL). The host indicator (H) field may be treated as an extension to the CL field for the purpose of maintaining the compatibility described within this programming note.

Diagnostic-Sampling Data Entry

Figure 2-18 shows the header and size of the diagnostic-sampling data entry with a format code of 8001 (hex). Figure 2-19 shows the header and size of the diagnostic-sampling data entry with a format code of 8002 (hex). Figure 2-20 shows the header and size of the diagnostic-sampling data entry with a format code of 8003 (hex). Figure 2-21 shows the header and size of the diagnostic-sampling data entry with a format code of 8004 (hex). Figure 2-22 shows the header and size of the diagnostic-sampling data entry with a format code of 8005 (hex). Figure 2-23 shows the header and size of the diagnostic-sampling data entry with a format code of 8006 (hex).

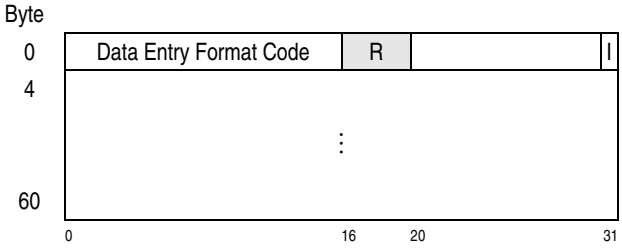


Figure 2-18. Format of the Diagnostic-Sampling Data Entry with format code 8001 (hex)

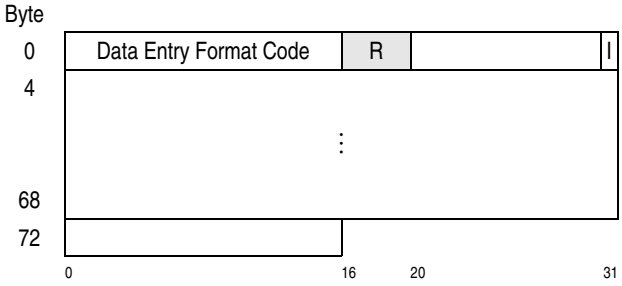


Figure 2-19. Format of the Diagnostic-Sampling Data Entry with format code 8002 (hex)

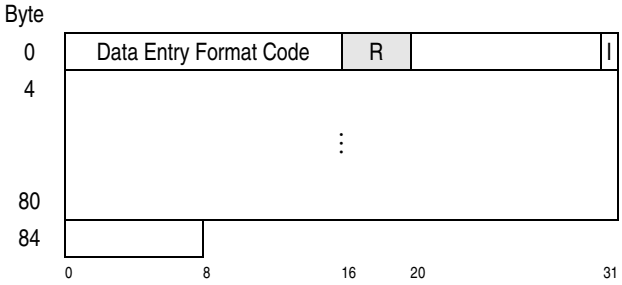


Figure 2-20. Format of the Diagnostic-Sampling Data Entry with format code 8003 (hex)

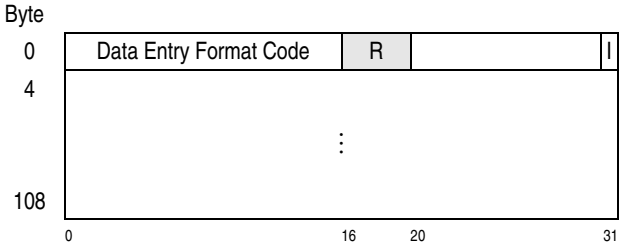


Figure 2-21. Format of the Diagnostic-Sampling Data Entry with format code 8004 (hex)

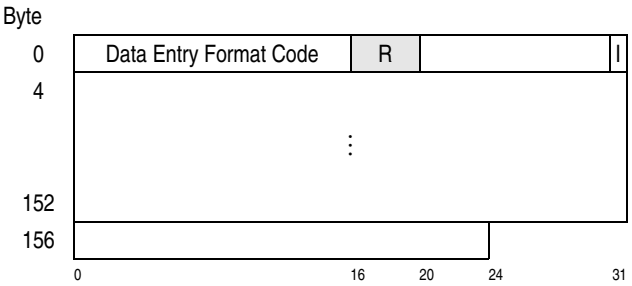


Figure 2-22. Format of the Diagnostic-Sampling Data Entry with format code 8005 (hex)

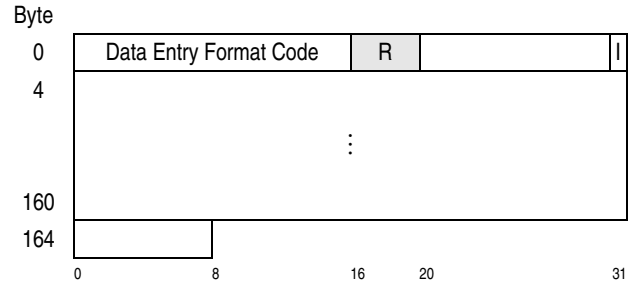


Figure 2-23. Format of the Diagnostic-Sampling Data Entry with format code 8006 (hex)

Bytes 0-3 of the diagnostic-sampling data entry are referred to as the header and consist of the following fields:

Data Entry Format Code: Bits 0-15 of the data entry contains the format code of the data entry.

Reserved (R): Bits 16-19 of the data entry are reserved for programming use. Zeros are stored in this field by the sample-data-block update process.

Invalid Indication (I): Bit 31 of the data entry indicates whether the entry is valid or invalid. When the bit is zero, the entry is valid; when the bit is one, the entry is invalid. An entry is set to invalid when sample data in the entry are not consistent.

The remaining bytes of the diagnostic-sampling data entry contain model dependent values and are not defined in this document.

Programming Note: Bit 19 of the diagnostic-sampling data entry is intended to indicate the maximum buffer size: 0 means 4K bytes, and 1 means 1 mega bytes.

Combined-Sampling Data Entry

When both the basic-sampling function and the diagnostic-sampling function are active, the sample data stored during each sampling interval is a combined-sampling data entry, which consists of a basic-sampling data entry followed by a diagnostic-sampling data entry.

When a combined-sampling data entry is to be stored, it is stored in the current sample-data block only if there exists enough space for the entire entry. When there is not enough space in the current sample-data block, if there is enough space in the next sample-data block, then the entire combined sample data is stored in the next sample-data block. The basic-sampling data entry and the diagnostic-sampling data entry of a combined-sampling data entry are not stored in different sample-data blocks. When the combined sample data is discarded because of no space available, the sample overflow count of the current sample-data block is incremented by one.

Trailer Entries

Each trailer entry is 64 bytes and resides in the last 64 bytes of a sample data block. The entry has the following format:

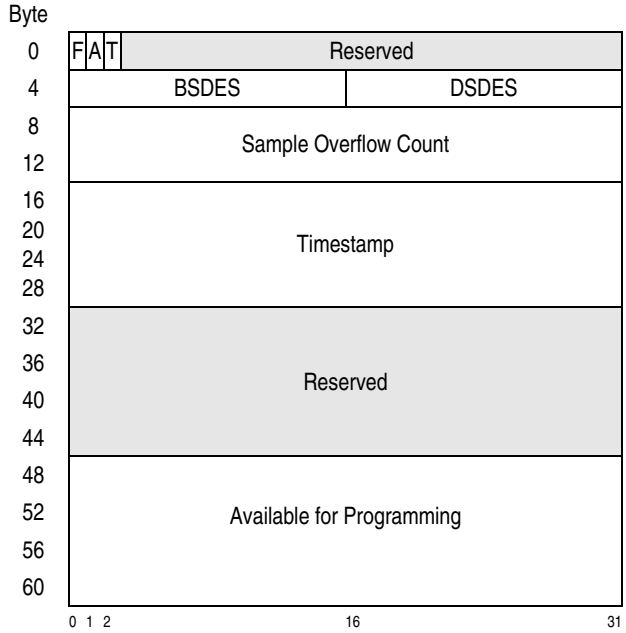


Figure 2-24. Format of the Trailer Entry

Block Full Indicator (F): Bit 0 of byte 0 of the trailer entry is the block full indicator. When the indi-

cator is one, the sample-data block is full. This bit is set, as appropriate, by the sample-data-block update process.

Alert Request Control (A): Bit 1 of byte 0 of the trailer entry is the alert request indicator. When the indicator is one and the sample-data block becomes full, a program-requested-alert external-interruption event is recognized at the completion of the sample-data-block update process. This bit is set by the program.

Timestamp Format (T): Bit 2 of byte 0 of the trailer entry is the timestamp format bit. When the bit is zero, the format of the timestamp stored in the timestamp field is in STORE CLOCK format; when the bit is one, the format of the timestamp stored in the timestamp field is in STORE CLOCK EXTENDED format.

Reserved: Bits 3-7 of byte 0 and bytes 1-3 of the trailer entry are reserved. The contents of this field remain unchanged.

Basic-Sampling Data Entry Size (BSDES): Bytes 4-5 of the trailer entry contain an unsigned binary integer, specifying the size, in bytes of the basic-sampling data entry.

Diagnostic-Sampling Data Entry Size (DSDES):

Bytes 6-7 of the trailer entry contain an unsigned binary integer, specifying the size, in bytes of the diagnostic-sampling data entry.

Sample Overflow Count: Bytes offsets 8-15 of the trailer entry contain the number of sample data that has been lost because the sample-data block is full.

Timestamp: The timestamp field contains the TOD clock value at the time when the sample-data block becomes full.

When the timestamp format (T) bit is zero, bytes 16-23 of the trailer entry contain a TOD clock value in STORE CLOCK format and bytes 24-31 are reserved.

When the T bit is one, bytes 16-31 of the trailer entry contain a TOD clock value in STORE CLOCK EXTENDED format.

Reserved: Bytes 32-39 of the trailer entry are reserved and remain unchanged by the CPU-measurement sampling facility.

Bytes 40-47 of the trailer entry are reserved for IBM use and may or may not be changed by the CPU-measurement sampling facility.

Bytes 48-63 of the trailer entry are available for use by programming and remain unchanged by the CPU-measurement sampling facility.

Programming Note: When a program-requested alert occurs, it is expected that the program reads out sample data from the sample-data blocks that are full. To free up these blocks, the program shall reset the block-full indicator (F) and the sample overflow count, and shall also reestablish an alert-request control (A). Updating these fields in a sample-data block shall be performed in an interlocked fashion using COMPARE DOUBLE AND SWAP (CDSG). All of these reads and updates are performed while sampling functions remain active.

Sample-Data-Block Update Process

When the CPU is in the operating state and at least one sampling function is active, the sample-data-block update process is performed during each sampling interval. The process locates space for the new data entry, forms the entry, and updates the contents of the data-entry-address register so that the register contents designate the location of the next data entry.

During the sample-data-block update process, if any address is formed through the addition of a value to another address, a carry out of bit position zero of the address, if any, is ignored. Similarly, when the contents of the sample overflow count field is incremented, a carry out of bit position zero of the count, if any, is ignored.

Accesses to a sample-data block are not subject to key-controlled protection; nor is it subject to low-address protection.

When storage access to a sample-data-block entry or sample-data-block-table entry is performed, if the address is invalid, a measurement-alert external-interruption event (invalid entry address) is recog-

nized, and all active sampling functions for that CPU are placed in the inactive state. An entry address is invalid if the address is in the range 0-8191, if the designated sample-data-block entry is inside the trailer entry, or if the designated storage location is not available in the configuration.

When storage access to a sample-data-block-table entry is performed, if an incorrect SDB table entry is detected, a measurement-alert external-interruption event (incorrect sample-data-block-table entry) is recognized, and all active sampling functions for that CPU are placed in the inactive state. A sample-data-block table entry is incorrect if the entry is a table-link entry and it designates another table-link entry, or if the last table entry is a block-link entry.

Process Steps

The contents of the data-entry-address register are used to locate the next data entry in the current sample data block. If the next data entry resides inside the last 64 bytes of the sample-data block, then a measurement-alert external-interruption event (invalid-entry-address alert) is recognized and all sampling functions for that CPU are placed in the inactive state.

Inner Loop

If the block full indicator in the trailer entry of the current sample data block is one, then go to step 1.

If the block full indicator in the trailer entry of the current sample-data block is zero and there exists enough space for the next data entry, then go to step 2.

If the block full indicator in the trailer entry of the current sample-data block is zero and if there is not enough space for the next data entry, then go to step 3.

1. The contents of the sample overflow count field in the trailer entry is incremented by one, the sample data to be stored is discarded, and the update process is completed.

2. The sample data is stored in the next data entry, the contents of the data-entry-address register are incremented by the data-entry size, and the update process is completed.

3. The block full indicator in the trailer entry of the current sample-data block is set to one and the TOD clock value is placed in the timestamp field of the trailer entry. If the alert-request bit in the trailer entry is one, a measurement-alert external-interruption event (program-requested alert) is recognized at the end of the update process. The contents of the table-entry-address register are incremented by the SDB table entry size so that the next entry in the SDB table becomes the current SDB table entry. The current SDB table entry is fetched and bit 63 of the entry is examined.

   a. If the fetched entry is the last entry in the SDB table before the table crosses an integral boundary of the maximum buffer size, and if bit 63 of the entry is zero, then go to step 3.a.1; if bit 63 of the entry is one, then go to step 3.a.2.

      1) A measurement-alert external-interruption event (incorrect-SDB-table-entry alert) is recognized, all active sampling functions are placed in the inactive state, and the update process is completed.

      2) The address of the origin of the SDB table specified in the entry is placed in the table-entry-address register so that the table becomes the current SDB table. The current SDB table entry is fetched and bit 63 of the entry is examined. If bit 63 is one, then go to step 3.a.2.x; if bit 63 is zero, then go to step 3.a.2.y.

         x) A measurement-alert external-interruption event (incorrect-SDB-table-entry alert) is recognized, all active sampling functions are placed in the inactive state, and the update process is completed.

         y) The address of the origin of the sample-data block specified in the entry is placed in the data-entry-address register so that the block becomes the current sample-data block. Go to the beginning of Inner Loop.

   b. If the fetched entry is not the last entry in the SDB table before the table crosses an integral boundary of the maximum buffer size, and

1) if bit 63 of the fetched table entry is zero, then go to step 3.a.2.y.

2) If bit 63 of the fetched entry is one, then go to step 3.a.2.

External Interruption

When the CPU-measurement counter or sampling facility is installed, an external-interruption subclass called the measurement-alert subclass is included.

For measurement-alert interruptions, a 32-bit parameter is associated with the interruption and is stored at real locations 128-131. The measurement-alert external-interruption condition is held pending.

The priorities for external-interruption requests in descending order are as follows:

- Interrupt key
- Malfunction alert
- Emergency signal
- External call
- Clock comparator
- CPU timer
- Timing alert
- Service signal or Measurement alert

The priority between service signal and measurement alert is undefined.

Measurement Alert

An interruption request for a measurement alert is generated when any alert event is recognized.

The measurement-alert subclass-mask bit is in bit position 58 of control register 0. This bit is initialized to one, which enables the interruption.

The measurement-alert condition is indicated by an external interruption code of 1407 hex. The interruption parameter stored at real locations 128-131 indicate the cause or causes for the interruption.

The measurement-alert condition exists when any of the following alert events is recognized:

Invalid Entry Address: An invalid-entry-address-alert event is recognized when an invalid address is detected during access to a sample-data-block entry or sample-data-block-table entry. An entry address is invalid if the address is in the range 0-8191, if the designated sample-data-block entry is inside of the trailer entry, or if the designated storage location is not available in the configuration.

Incorrect Sample-Data-Block-Table Entry: An incorrect-sample-data-block-table-entry-alert event is recognized when an incorrect table entry is detected during access to a sample-data-block-table entry. A sample-data-block-table entry is incorrect if the table entry is a table-link entry which designates another table-link entry, or if it is the last entry in the sample-data-block table and it is a block-link entry.

Program Request Alert: A program-request-alert event is recognized when the alert-request bit in the trailer entry of the current sample-data block is one and the current sample-data block becomes full.

Counter Authorization Change Alert: A counter-authorization-change-alert event is recognized when the authorization control of an enabled counter set (that is, the enable control is one) changes from authorized to unauthorized.

Sampling Authorization Change Alert: A sampling-authorization-change-alert event is recognized when the authorization control of an enabled sampling function (that is, the enable control is one) changes from authorized to unauthorized.

Loss Of Sample Data Alert: A loss-of-sample-data alert event is recognized when sample data is lost because of certain machine internal high-priority activities.

Loss Of CPU Counter Data Alert: A loss-of-CPU-counter-data alert event is recognized when counter data of a CPU counter is lost because of certain machine internal high-priority activities.

Loss of MT Counter Data Alert: A loss-of-MT-counter-data alert event is recognized when counter data of a CPU counter in the MT-diagnosticcounter set is lost because of certain machine internal high-priority activities. At the time the loss-of-MT-counter-data alert is made pending, the content of the counters is unpredictable, but will continue to increment from that point forward.

If the same measurement-alert event is recognized more than once before the interruption occurs, the request is generated only once. The request is generated for the CPU and remains pending at the CPU until it is cleared. The pending request is cleared when it causes an interruption at the CPU and by CPU reset. In addition, the pending invalid-entry-address, incorrect-sampling-data-block-table-entry, program-request-alert, and loss-of-sample-data-alert

interruption requests are cleared when no sampling functions are enabled; the pending loss-of-CPU-counter-data-alert interruption request is cleared when no non-MT CPU counter sets are enabled. However, a pending counter-authorization-change or sampling-authorization-change interruption request is not cleared when no counter sets or sampling functions, respectively, are enabled. The pending loss-of-MT-counter-data-alert event may also be cleared when a STORE CPU COUNTER MULTIPLE instruction is executed with an M_3 value of 9.

Recognition by the machine of an invalid-entry-address alert, an incorrect-sample-data-block-table-entry alert, or a loss-of-sample-data alert places all active sampling functions in the inactive state; recognition of a loss-of-CPU-counter-data alert for a counter in a CPU counter set places that counter set in the inactive state. Recognition by the machine of a loss-of-MT-counter-data alert leaves the counter set in the active state. However, any instruction that attempts to access a counter in the MT-diagnostic counter set while a loss-of-MT-counter-data alert is pending on the CPU in which it is executing completes with a condition code of 3.

A 32-bit external-interruption parameter is associated with the measurement-alert interruption and is stored at real locations 128-131. The field is defined as shown below:

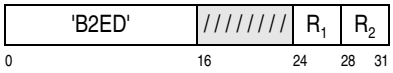
Bit	Meaning
0	Invalid entry address
1	Incorrect sample-data-block-table entry
2	Program request alert
3-7	Reserved
8	Sampling authorization change alert
9	Loss of sample data alert
10-15	Reserved
16	Loss of MT counter data alert
17-23	Reserved
24	Counter authorization change alert
25	Loss of CPU counter data alert
26-31	Reserved

Multiple alert events may be indicated concurrently in the external-interruption parameter field.

Programming Note: An authorization control is normally set to the same value for all logical CPUs in the configuration. When an authorization-change alert occurs, it normally occurs on all logical CPUs in the configuration.

EXTRACT COPROCESSOR-GROUP ADDRESS

ECPGA R_1,R_2 [RRE]



The address of the coprocessor group attached to the specified CPU is placed at the first-operand location.

Bits 48-63 of general register R_2 specify the CPU address at the basic machine level. Bits 0-47 of the general register must be zeros; otherwise, a specification exception is recognized.

Bits 40-55 of the first-operand location contains the coprocessor-group address at the basic machine level. Bits 32-39 and 56-63 of the first-operand location are set to zeros; bits 0-31 of the first-operand location remain unchanged.

When the specified CPU is not installed, condition code 3 is set and the contents of the first-operand location remain unchanged.

Resulting Condition Code:

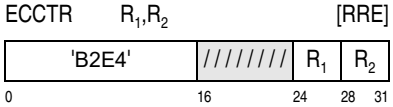
0 Normal completion
1 --
2 --
3 CPU not installed

Program Exceptions:

- Operation (if the CPU-measurement counter facility is not installed)
- Privileged operation
- Specification

EXTRACT CPU COUNTER

ECCTR R_1,R_2 [RRE]



The contents of the specified counter in the CPU counter set are placed at the first-operand location.

Bits 48-63 of general register R_2 specify the counter number of a counter in the CPU counter set. Bits 0-47 of the general register must be zeros; otherwise, a specification exception is recognized.

When the specified counter is enabled (that is, the enable control is one), the contents of the counter are stored at the first-operand location and condition code 0 is set. When the specified counter is not available, condition code 3 is set and the contents of the first-operand location remain unchanged. A counter is not available when the specified counter number is unassigned, not installed, disabled, or unauthorized, or, if the counter is in an MT counter set, with a pending loss-of-MT-counter-data alert in the issuing CPU.

In the problem state, the measurement-counter-extraction-authorization control, bit 15 of control register 0, must be one; otherwise, a privileged-operation exception is recognized. In the supervisor state, the measurement-counter-extraction-authorization control is not examined.

Resulting Condition Code:

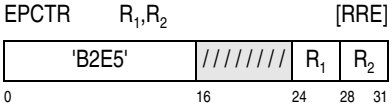
0 Normal completion
1 --
2 --
3 Counter not available

Program Exceptions:

- Operation (if the CPU-measurement counter facility is not installed)
- Privileged operation (measurement-counter-extraction-authorization control is zero in the problem state)
- Specification

EXTRACT PERIPHERAL COUNTER

EPCTR R_1,R_2 [RRE]



The contents of the specified counter in the specified coprocessor-group counter set are placed at the first-operand location.

Bits 40-55 of general register R_2 specify the basic machine coprocessor-group address and bits 56-63 specify the counter number of a counter in the coprocessor-group counter set. Bits 0-39 of the general register must be zeros; otherwise, a specification exception is recognized.

When the specified counter is enabled (that is, the enable control is one), the contents of the counter are stored at the first-operand location and condition code 0 is set. When the specified counter is not available, condition code 3 is set and the contents of the first-operand location remain unchanged. A counter is not available when the specified coprocessor-group address is unassigned or not installed, or the specified counter number is unassigned, not installed, disabled, or unauthorized.

Resulting Condition Code:

0 Normal completion
1 --
2 --
3 Counter not available

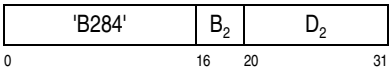
Program Exceptions:

- Operation (if the CPU-measurement counter facility is not installed)
- Privileged operation
- Specification

Programming Note: Any CPU that is authorized for the peripheral counter sets can execute EXTRACT PERIPHERAL COUNTER or SET PERIPHERAL COUNTER. It is not required toextract or set contents of a peripheral counter by the same CPU that established the controls for the counter sets.

LOAD CPU-COUNTER-SET CONTROLS

LCCTL $D_2(B_2)$ [S]

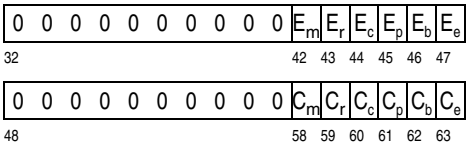


The CPU-counter-set state controls in the storage locations designated by the second-operand address are placed in the local counter-set-state control register, except where specified otherwise.

The second operand is 64 bits. Bits 42-47 and bits 58-63 of the second operand specify state controls for CPU counter sets. All other bits of the second operand, along with bits for any counter sets not supported by the model must be zeros; otherwise a specification exception is recognized.

When the model supports the MT-diagnostic counter set, bits 42 and 58 of the second operand are ignored and will not cause a specification exception. Other actions are required to update the enable and activation controls, of the local counter-set-state control register, for the MT-diagnostic counter set.

The following shows the format of bits 32-63 of the second operand.



Bit 42, E_m, of the second operand contains the enable control for the MT-diagnostic counter set.

Bit 43, E_r, of the second operand is reserved for IBM use.

Bit 44, E_c, of the second operand contains the enable control for the crypto-activity counter set.

Bit 45, E_p, of the second operand contains the enable control for the problem-state counter set.

Bit 46, E_b, of the second operand contains the enable control for the basic counter set.

Bit 47, E_e, of the second operand contains the enable control for the extended counter set.

Bit 58, C_m, of the second operand contains the activation control for the MT-diagnostic counter set.

Bit 59, C_r, of the second operand is reserved for IBM use.

Bit 60, C_c,of the second operand contains the activation control for the crypto-activity-counter set.

Bit 61, C_p, of the second operand contains the activation control for the problem-state counter set.

Bit 62, C_b, of the second operand contains the activation control for the basic counter set.

Bit 63, C_e, of the second operand contains the activation control for the extended counter set.

The specified state controls of non-ignored CPU counter sets are set only if all CPU counter sets can make a valid state transition. (See Figure 2-2 on page 2-5 for the valid state transitions.) If a counter set cannot make a valid state transition, the specified state controls are not set for any counter set.

When the specified state controls of all non-ignored CPU counter sets are set, condition code 0 is set. When the specified state controls of all non-ignored CPU counter sets are not set, condition code 3 is set.

Resulting Condition Code:

0 Normal completion
1 --
2 --
3 Controls not set

Program Exceptions:

- Access (fetch, operand 2)
- Privileged operation
- Operation (if the CPU-measurement counter facility is not installed)
- Specification

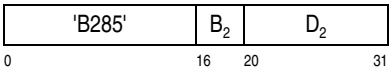
Programming Notes:

1. When a CPU counter set is in the unauthorized, disabled, inactive, or active state before the operation, if the operation sets the state controls to the same state as the original state, the state controls are considered set successfully in the absence of any exception. Thus, if a CPU counter set is unauthorized and if the operation sets the state controls to zeros, condition code 0 is set and no state change occurs.

2. When the counter second version number is less than or equal to three, bits 42-43 and 58-59 of the second operand must be zero; otherwise, a specification exception is recognized. When the counter second version number is greater than three, bits 42 and 58 of the second operand are ignored and will not cause a specification exception. When the counter second version number is four, bits 43 and 59 of the second operand are ignored and will not cause a specification exception. When the counter second version number is greater than or equal to five, bits 43 and 59 of the second operand must be zero; otherwise, a specification exception is recognized.

LOAD PERIPHERAL-COUNTER-SET CONTROLS

LPCTL $D_2(B_2)$ [S]



The peripheral-counter-set state controls in the storage locations designated by the second-operand address are placed in the global counter-set-state control register.

The second operand is 64 bits. Bits 62-63 of the second operand specify the enable control and activation control, respectively, for all coprocessor-group counter sets in the model. All other bits of the second operand must be zeros; otherwise a specification exception is recognized.

The specified state controls of all coprocessor-group sets are set only if all sets can make a valid state transition. (See Figure 2-2 on page 2-5 for the valid state transitions.) If the sets cannot make a valid state transition, the specified state controls are not set.

When the specified state controls of all coprocessor-group counter sets are set, condition code 0 is set. When the specified state controls of all coprocessor-group counter sets are not set, condition code 3 is set.

Resulting Condition Code:

0 Normal completion
1 --
2 --
3 Controls not set

Program Exceptions:

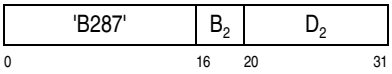
- Access (fetch, operand 2)
- Privileged operation
- Operation (if the CPU-measurement counter facility is not installed)
- Specification

Programming Notes:

1. When the coprocessor-group counter sets and the controls for the coprocessor-group counter sets are not installed, condition code 3 is set for the instruction.
2. When all coprocessor-group counter sets are in the unauthorized, disabled, inactive, or active state before the operation, if the operation sets the state controls to the same state as the original state, the state controls are considered set successfully in the absence of any exception. Thus, if coprocessor-group counter sets are unauthorized and if the operation sets the enable and activation controls to zeros, condition code 0 is set and no state change occurs.
3. The enable and activation controls for coprocessor-group counter sets are global controls, which are observable by all CPUs that are authorized for coprocessor-group counter sets.

LOAD SAMPLING CONTROLS

LSCTL $D_2(B_2)$ [S]



The sampling controls in the request block in the storage locations designated by the second-operand address are placed in the corresponding sampling control registers.

The request block is 64 bytes and has the following format:

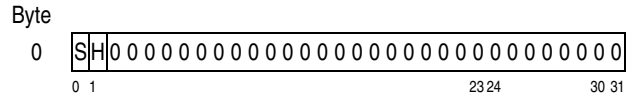


Figure 2-25. Format of the Information Block

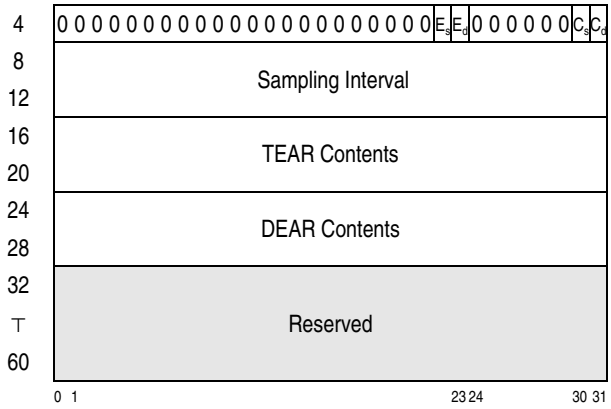


Figure 2-25. Format of the Information Block

Bit 0, S, of the request block specifies the maximum-buffer-size indicator.

When the CPU is running at the logical partition level, bit 1, H, of the request block is the host indicator. When the CPU is running at the virtual machine level, bit 1 is ignored.

Bit 54, E_s, of the request block specifies the basic-sampling enable control; bit 55, E_d, of the request block specifies the diagnostic-sampling enable control. Bit 62, C_s, of the request block specifies the basic-sampling activation control; bit 63, C_d, of the requested block specifies the diagnostic-sampling activation control. Bits 2-53 and 56-61 of the request block must be zeros; otherwise, a specification exception is recognized.

The specified enable and activation controls of sampling functions are set only if both sampling functions can make a valid state transition. (See Figure 2-12 on page 2-15 for the valid state transitions.) If a sampling function cannot make a valid state transition, the specified state controls are not set for any sampling function.

When a sampling function is set to the active state from the disabled, inactive, or active state, the additional controls, including the maximum-buffer-size indicator, the host indicator, the sampling interval, the table-entry-address register contents, and the data-entry-address register contents are placed in the sampling control registers. When no sampling function is placed to the active state by the operation, the additional controls are ignored and are not placed in the sampling control registers.

When a sampling function is to be set to the active state from the disabled, inactive, or active state, and if the requested sampling interval is outside the supported range, no sampling control, including any state control, in the sampling control registers is changed and a specification exception is recognized. The supported sampling interval range is between the maximum and minimum sampling intervals, inclusively, provided by executing QUERY SAMPLING INFORMATION.

Bytes 32-63 of the request block are ignored.

When the enable and activation controls for both sampling functions are set, condition code 0 is set. When the enable and activation controls are not set, condition code 3 is set.

Resulting Condition Code:

0 Normal completion
1 --
2 --
3 State controls not set

Program Exceptions:

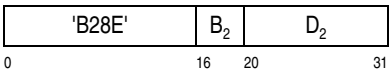
- Access (fetch, operand 2)
- Privileged operation
- Operation (if the CPU-measurement sampling facility is not installed)
- Specification

Programming Notes:

1. When a sampling function is in the unauthorized, disabled, inactive, or active state before the operation, if the operation sets the state controls to the same state as the original state, the state controls are considered set successfully in the absence of any exception. Thus, if a sampling function is unauthorized and if the operation sets the state controls to zeros, condition code 0 is set and no state change occurs.
2. Bytes 32-63 of the requested block should be set to zeros for future extensions.

QUERY COUNTER INFORMATION

QCTRI $D_2(B_2)$ [S]



Information about the CPU-measurement counter facility is placed in the information block designated by the second-operand address.

The information block is 64 bytes and has the following format:

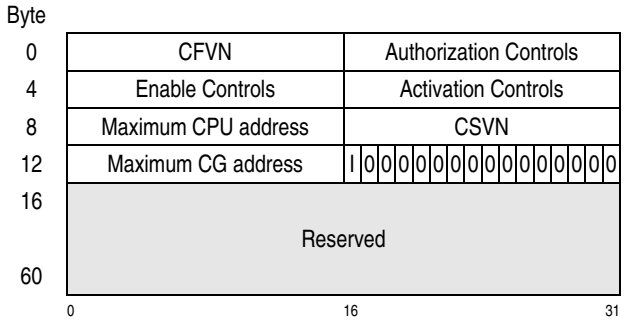
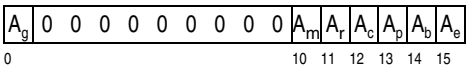


Figure 2-26. Format of the Information Block

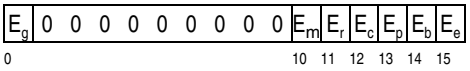
Bytes 0-1 of the information block contain the counter first version number (CFVN).

Bytes 2-3 of the information block contain the authorization controls and have the following format:



A_g Coprocessor-group-counter-set authorization control
A_m MT-diagnostic-counter-set authorization control
A_r Reserved for IBM use
A_c Crypto-activity-counter-set authorization control
A_p Problem-state-counter-set authorization control
A_b Basic-counter-set authorization control
A_e Extended-counter-set authorization control

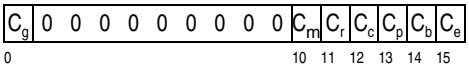
Bytes 4-5 of the information block contain the enable controls and have the following format:



E_g Coprocessor-group-counter-set enable control
E_m MT-diagnostic-counter-set enable control

E_r Reserved for IBM use
E_c Crypto-activity-counter-set enable control
E_p Problem-state-counter-set enable control
E_b Basic-counter-set enable control
E_e Extended-counter-set enable control

Bytes 6-7 of the information block contain the activation controls and have the following format:



C_g Coprocessor-group-counter-set activation control
C_m MT-diagnostic-counter-set activation control
C_r Reserved for IBM use
C_c Crypto-activity-counter-set activation control
C_p Problem-state-counter-set activation control
C_b Basic-counter-set activation control
C_e Extended-counter-set activation control

Bytes 8-9 of the information block contain the maximum CPU address at the basic machine level.

Bytes 10-11 of the information block contain the counter second version number (CSVN).

Bytes 12-13 of the information block contain the maximum coprocessor group address at the basic machine level.

Bit 0 of byte 14 of the information block is the coprocessor-group-address-change indicator. Bits 1-7 of byte 14 and the contents of byte 15 are set to zeros.

The contents of the reserved fields are set to zeros.

Condition Code: The code remains unchanged.

Program Exceptions:

- Access (store, operand 2)
- Privileged operation
- Operation (if the CPU-measurement counter facility is not installed)

Programming Notes:

1. The enable and activation controls for the MT-diagnostic counter set are set by the successful execution of a Set Multithreading SIGNAL PROCESSOR order with a non-zero parameter and reset by any action which disables multi-threading.

2. Each CPU can be individually authorized for the coprocessor-group-counter sets. However, there is only one enable control and activation control for all coprocessor-groupcounter sets. For CPUs that are unauthorized for the coprocessor-group counter sets, they observe all coprocessor-group counter sets are in the unauthorized state. For CPUs that are authorized for the coprocessor-group counter sets, they observe that all coprocessor-group counter sets are in the disabled, inactive, or active state.

3. For a counter measurement run involving coprocessor-group counters, the program should clear the coprocessor-group-address-change indicator at the beginning of the measurement run and check the indicator at the end. If the indicator is set at the end of the measurement run, then the program should discard measured coprocessor-group-counter data, set the global enable control to zero to reset the indicator, and repeat the measurement run. It is suggested the following steps be used to protect against the change of a coprocessor-group address:

   a. A QUERY COUNTER INFORMATION instruction is issued to obtain the maximum CPU address.

   b. A LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction is issued to set both the global enable control and the global activation control to zero. (This ensures that the indicator is cleared if a previous program didn't set the global enable bit to zero.)

   c. A LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction is issued to set the global enable control to one and the global activation control to zero.

   d. The EXTRACT COPROCESSOR-GROUP ADDRESS instruction is repeatedly issued to get the connectivity between all CPUs and all coprocessor groups.

   e. A LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction is issued to set both the global enable control and the global activation control to one.

f. The intended workload begins to run.

g. When the workload is completed, a LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction is issued to set the global enable control to one and the global activation control to zero.

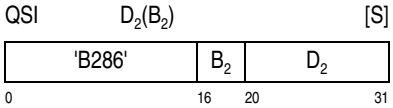
h. A QUERY COUNTER INFORMATION instruction is issued to obtain the indicator. If the indicator is one, then the measurement run is repeated by going to step (b). If the indicator is zero, then go to step (i).

i. The contents of each coprocessor-group counter are obtained by executing the EXTRACT PERIPHERAL COUNTER instruction.

j. A LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction is issued to set both the global enable control and the global activation control to zero.

Note that the LOAD PERIPHERAL-COUNTER-SET CONTROLS instruction can be successfully executed only on a CPU that has the coprocessor-group-counter-set authorization control set to one.

QUERY SAMPLING INFORMATION



Information about the CPU-measurement sampling facility is placed in the information block designated by the second-operand address.

The information block is 64 bytes and has the following format:

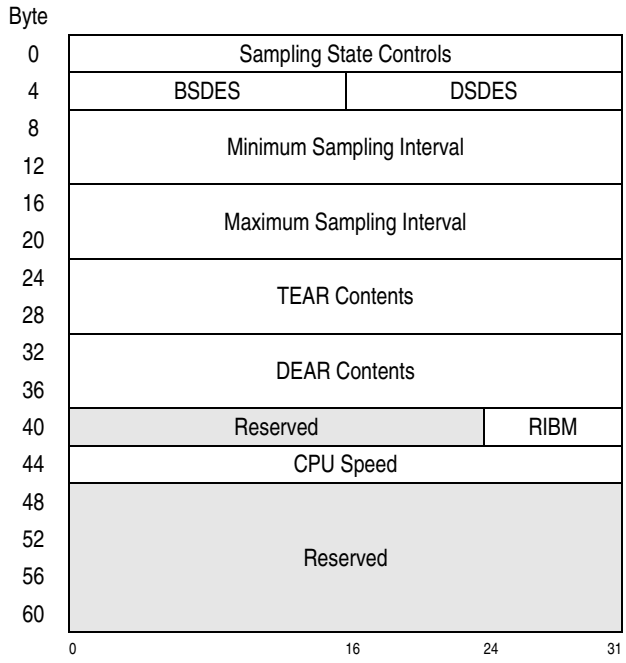
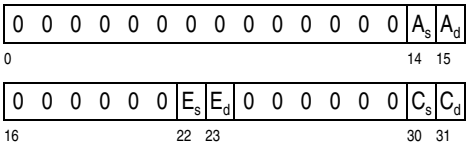


Figure 2-27. Format of the Information Block

Sampling State Controls: Bytes 0-3 of the information block contain the state controls for the basic-sampling and diagnostic-sampling functions and have the following format:



A_s Basic-sampling authorization control
A_d Diagnostic-sampling authorization control
E_s Basic-sampling enable control
E_d Diagnostic-sampling enable control
C_s Basic-sampling activation control
C_d Diagnostic-sampling activation control

Basic-Sampling-Data-Entry Size (BSDES): Bytes 4-5 of the information block contain an unsigned binary integer, specifying the size in bytes of the basic-sampling data entry.

Diagnostic-Sampling-Data-Entry Size (DSDES): Bytes 6-7 contain an unsigned binary integer, specifying the size in bytes of the diagnostic-sampling data entry.

Minimum Sampling Interval: Bytes 8-15 of the information block contain the minimum sampling interval in number of CPU cycles.

Maximum Sampling Interval: Bytes 16-23 contain the maximum sampling interval in number of CPU cycles.

TEAR Contents: When the basic-sampling or diagnostic-sampling function, or both, are enabled (that is, the enable control is one), bytes 24-31 of the information block contain the contents of the table-entry-address register. When neither the basic-sampling nor diagnostic-sampling function is enabled (that is, the enable control is one), zeros are stored in bytes 24-31 of the information block.

DEAR Contents: When the basic-sampling or diagnostic-sampling function, or both, are enabled (that is, the enable control is one), bytes 32-39 of the information block contain the contents of the data-entry-address register. When neither the basic-sampling nor the diagnostic-sampling function is enabled, zeros are stored in bytes 32-39 of the information block.

Reserved for IBM use (RIBM): Byte 43 of the information block is reserved for IBM use.

CPU Speed: Bytes 44-47 contain a unsigned binary integer, which specifies the CPU speed in number of CPU cycles per microsecond.

The contents of the reserved fields are set to zeros.

Condition Code: The code remains unchanged.

Program Exceptions:

- Access (store, operand 2)
- Privileged operation
- Operation (if the CPU-measurement sampling facility is not installed)

Programming Notes:

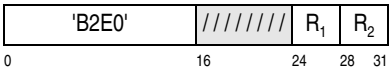
1. All CPUs in a configuration do not necessarily have the same CPU speed.

2. The value of CPU speed can be used to compute the sampling interval and the total size of sample-data blocks.

   When CPUs in the configuration have various speed, it is suggested that sample data is taken on each CPU when the CPU makes about the same progress as other CPUs. Therefore, the same sampling interval should be set to all CPUs, regardless of which CPU speed is used to compute the sampling interval. However, the total size of the sample-data blocks for a CPU depends on the CPU speed of that CPU. The faster CPU needs a larger total sample-data-block size.

SET CPU COUNTER

SCCTR R_1,R_2 [RRE]



The contents of the general register designated by R_1 are placed in the specified CPU counter.

Bits 48-63 of general register R_2 specify the counter number of a counter in the CPU counter set. Bits 0-47 of the general register must be zeros; otherwise, a specification exception is recognized.

When the specified counter is enabled (that is, the enable control is one), the contents of the general register designated by R_1 are placed in the counter and condition code 0 is set. When the specified counter is not available, condition code 3 is set. A counter is not available when the specified counter number is unassigned, not installed, disabled, read-only, or unauthorized.

Resulting Condition Code:

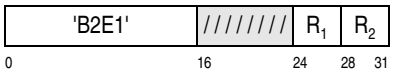
0 Normal completion
1 --
2 --
3 Counter not available

Program Exceptions:

- Operation (if the CPU-measurement counter facility is not installed)
- Privileged operation
- Specification

SET PERIPHERAL COUNTER

SPCTR R_1,R_2 [RRE]



The contents of the general register designated by R_1 are placed in the specified peripheral counter.

Bits 40-55 of general register R_2 specify the basic machine coprocessor-group address and bits 56-63 specify the counter number of a counter in the coprocessor-group counter set. Bits 0-39 of the general register must be zeros; otherwise, a specification exception is recognized.

When the specified counter is enabled (that is, the enable control is one), the contents of the general register designated by R_1 are placed in the counter and condition code 0 is set. When the specified counter is not available, condition code 3 is set. A counter is not available when the specified coprocessor-group address is unassigned or not installed, or the specified counter number is unassigned, not installed, disabled, or unauthorized.

Resulting Condition Code:

0 Normal completion
1 --
2 --
3 Counter not available

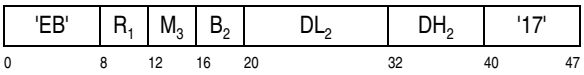
Program Exceptions:

- Operation (if the CPU-measurement counter facility is not installed)
- Privileged operation
- Specification

Programming Note: Any CPU that is authorized for the peripheral counter sets can execute EXTRACT PERIPHERAL COUNTER or SET PERIPHERAL COUNTER. It is not required to extract or set contents of a peripheral counter by the same CPU that established the controls for the counter sets.

STORE CPU COUNTER MULTIPLE

STCCTM $R_1,M_3,D_2(B_2)$ [RSY-b]



The contents of a range of counters specified by the first and third operands are stored to the second operand location.

Bits 48-63 of the first operand specify the maximum number of counters to store. Bits 0-47 must be zero; otherwise, a specification exception is recognized.

The M_3 field specifies a code used to indicate the counter set to be stored. If the M_3 field specifies a reserved value a specification exception is recognized.

The M_3 field is defined as follows:

0 Extended counter set

1 Basic counter set

2 Problem-state counter set

3 Crypto-activity counter set

4 Reserved for IBM use

5 MT-diagnostic counter set

6-7 Reserved

8 Reserved for IBM use

9 MT-diagnostic counter set with clearing of loss-of-MT-counter-data alert

10-15 Reserved

If the number of counters specified by the first operand is greater than the number of available counters in the counter set then the number of doublewords stored is equal to the number of counters in the counter set. The remaining doublewords in the second operand are unchanged. Access exceptions may or may not be indicated on the remaining doublewords.

If the first operand specifies zero counters to store, the specified counterset is not installed, disabled, or unauthorized, or, if the code specified in the M_3 field is 5 and a loss-of-MT-counter-data alert is pending,

condition code three is set and no counters are stored.

The second operand must be aligned on a double-word boundary; otherwise, a specification exception is recognized.

In the problem state, the measurement-counter-extraction-authorization control, bit 15 of control register 0, must be one; otherwise, a privileged-operation exception is recognized. In the supervisor state, the measurement-counter-extraction-authorization control is not examined.

In the problem state, if the measurement-counter-extraction-authorization control is one, and the M_3 field specifies a code of 9, a special-operation exception is recognized.

If the M_3 field specifies a code of 9, and a pending loss-of-MT-counter-data alert exists, the alert is made no longer pending. The remainder of the function behaves the same as code 5.

Resulting Condition Code:

0 Number of counters requested equal to number of available counters in the requested counter set; all counters in counter set stored.

1 Number of counters requested less than number of available counters in the requested counter set; partial set of counters in counter set stored.

2 Number of counters requested greater than number of available counters in the requested counter set; all counters in counter set stored.

3 No counters stored

Program Exceptions:

- Access (store, second operand)
- Operation (if the store-CPU-counter-multiple facility is not installed)
- Privileged operation (measurement-counter-extraction-authorization control is zero in the problem state)
- Special operation
- Specification
- Transaction constraint

Programming Note: Use of this instruction to extract only a few counters from a counter set may provide reduced performance over multiple calls to EXTRACT CPU COUNTER.

End of Document

SA23-2260-06