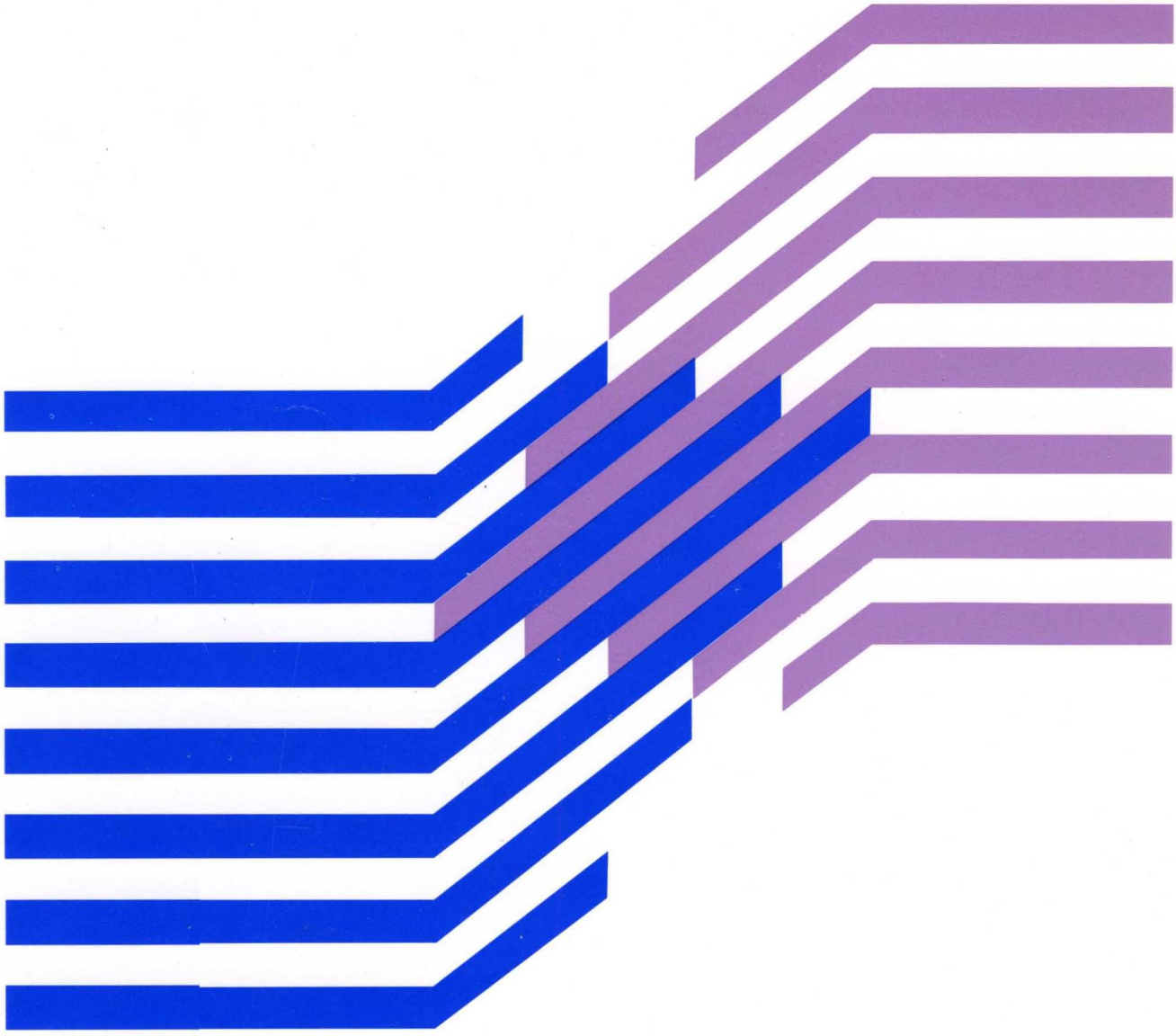
LY28-1432-0

MVS/ESA
Component Diagnosis and Logic: Alternate CPU Recovery

MVS/System Product:
JES2 Version 3
JES3 Version 3



LY28-1432-0

MVS/ESA
Component Diagnosis and Logic: Alternate CPU Recovery

MVS/System Product:
JES2 Version 3
JES3 Version 3

First Edition (July, 1988)

This edition with Technical Newsletter LN28-1334 applies to Version 3 of MVS/System Product 5685-001 or 5685-002 and to all subsequent releases until otherwise indicated in new editions or Technical Newsletters. See the Summary of Changes following the Contents for a summary of the changes to this manual. Changes are made periodically to the information herein; before using this publication in connection with the operation of IBM systems, consult the latest IBM System/370 Bibliography, GC20-0001, for the editions that are applicable and current.

Publications are not stocked at the address given below. Requests for IBM publications should be made to your IBM representative or to the IBM branch office serving your locality.

A form for reader's comments is provided at the back of this publication. If the form has been removed, comments may be addressed to IBM Corporation, Information Development, Department D58, Building 921-2, PO Box 950, Poughkeepsie, NY 12602. IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

PROGRAMMING INTERFACES

This book does NOT contain any programming interface information.

ABOUT THIS BOOK

This book provides a general understanding of this component's processing. It contains information that can be useful when diagnosing a problem suspected to have been caused by the component.

TRADEMARKS

The following is a trademark of International Business Machines Corporation:

- MVS/ESA(TM)

WHO THIS BOOK IS FOR

This book is for anyone who wants to gain a general understanding of the component's processing, or who needs to diagnose a problem that appears to be caused by the component.

The level of detail at which this book is written assumes that the reader:

- Understands the commonly used system diagnostic tasks and aids, such as those presented in the Basics of Problem Determination book
- Codes in assembler language, and reads assembler, and linkage editor output
- Understands basic system concepts and the use of system services
- Understands the externals of the component

HOW THIS BOOK IS ORGANIZED

This book contains the following chapter headings. However, it might not include information for a chapter because the information is not applicable. If the information for a chapter is not applicable, the chapter heading will contain the words "Not Applicable".

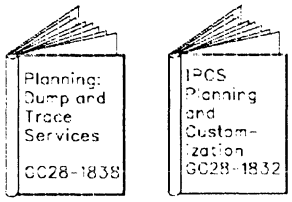
- Part 1. Diagnostic Procedures for the component
  - **Chapter 1. Diagnosis for** this component suggests how to diagnose problems in the component.
- Part 2. Reference for the component
  - **Chapter 2. Introduction to** this component gives the functions performed by the component and the expected inputs and outputs.
  - **Chapter 3. Control Block Overview** shows the significant fields and the chaining structure of the component's control blocks.
  - **Chapter 4. Process Flow** shows the control flow among the component's modules.
  - **Chapter 5. Method of Operation** describes the functional organization of the component.
  - **"Index".**

RELATED INFORMATION

While using this book, you will need the information in the following books:

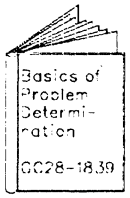
Short Title Used in This Book	Title	Order Number
Basics of Problem Determination	MVS/ESA(TM) Basics of Problem Determination	GC28-1839
IPCS Command Reference	MVS/ESA Interactive Problem Control System (IPCS) Command Reference	GC28-1834
IPCS User's Guide	MVS/ESA Interactive Problem Control System (IPCS) User's Guide	GC28-1833
Service Aids	MVS/ESA Service Aids	GC28-1844
System Messages	MVS/ESA Message Library: System Messages, Volumes 1 - 2	GC28-1812 and GC28-1813

PLANNING:

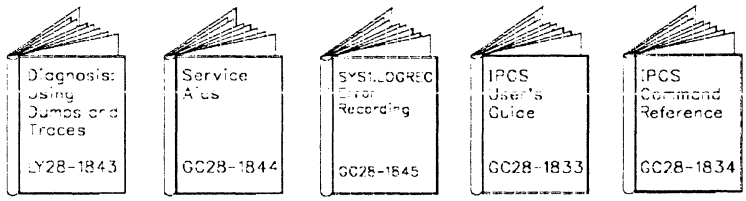


DIAGNOSIS:

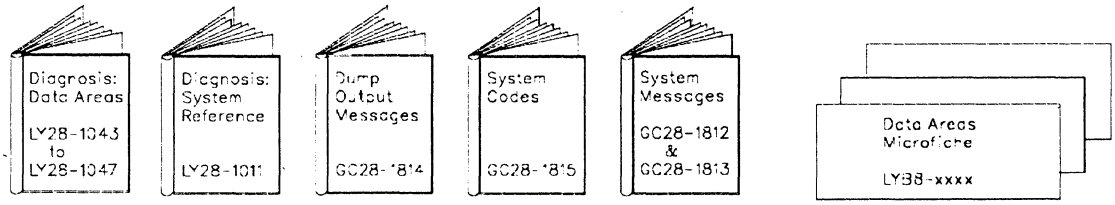
For Diagnostic Procedures



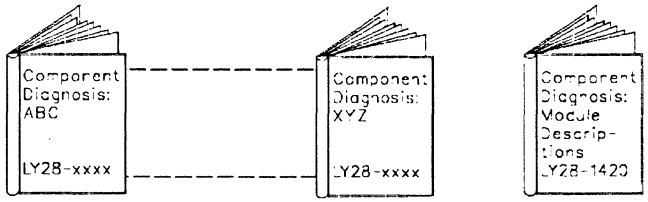
For Data Collection



For Data Interpretation



For Component-Specific Information
(To identify the component, see Basics of Problem Determination)



CONTENTS

FIGURES

SUMMARY OF CHANGES

Summary of Changes
for LY28-1432-0
as updated December 22, 1989
by Technical Newsletter LN28-1334

CHANGED INFORMATION: The Diagnosis Library figure has been updated.

Minor editorial changes have been made in the front matter.

Summary of Changes
for LY28-1432-0
MVS/System Product Version 3 Release 1.0

This book contains information previously presented in MVS/XA System Logic Library: Alternate CPU Recovery, LY28-1617. The following summarizes the changes to that information.

New Information

"Part 1. Diagnostic Procedures for ACR" is new and contains the following:

- "Chapter 1. Diagnosis for ACR"

"Part 2. Reference for ACR" is new and contains the following:

- "Chapter 2. Introduction"
- "Chapter 3. Control Block Overview"
- "Chapter 4. Process Flow"
- "Chapter 5. Method of Operation"

Changed Information

ACR processing has been updated in support of this release. In addition, the HIPO "Alternate CPU Recovery Overview" has been updated to include control register identification.

PART 1. DIAGNOSTIC PROCEDURES FOR ACR

CHAPTER 1. DIAGNOSIS FOR ACR

Prerequisites

If you are diagnosing a suspected problem in this component, first collect the following problem data. If you do not have this data, refer to the Basics of Problem Determination book and perform its procedures.

- The problem type, such as an ABEND or wait state.
- The product name.
- The component name: ACR

  Use this book to diagnose problems only in component ACR. If the component name is not ACR, return to the Basics of Problem Determination book to identify the component. Use the component diagnosis book for the identified component.
- The module name
- The symptoms used as the initial search argument.
- A dump, if needed for the problem. Format the dump for component ACR, as described in Basics of Problem Determination.
- The system execution status, if needed for the problem.

The alternate CPU recovery component does not provide component-specific symptom data. If you report the problem to IBM, see the Basics of Problem Determination book for a list of problem data to collect. The remainder of this book might help you diagnose the problem.

PROCESSING DIAGNOSIS FOR ACR

Alternate CPU recovery (ACR), running on a properly functioning processor, initiates the release of global resources that are held by programs that were running on the failed processor. ACR does this by changing control block pointers so that the operating system can pretend that it is running on the failed processor, and thereby run the functional recovery routines (FRRs) for that processor. This results in the release of the locks and other global resources held on the failed processor.

ACR consists of three processing phases: pre-processing, intermediate, and post-processing.

ACR pre-processing sets the LCCARCPU and LCCADCPU fields in the LCCAs of both processors with the addresses of the LCCAs of the recovery and failed processors respectively.

The FRR processing for the failed processor could require a resource (such as the SALLOC lock to free storage) held by the recovery processor. In this case, the FRR also invokes ACR's intermediate phase. This entry results in the suspension of the FRR processing for the failed processor, and another processor-switch back to the recovery processor to resume its processing (to allow the resource needed by the failed processor's FRR to be freed).

ACR's post-processing consists of issuing a SYSEVENT to inform SRM about the failed processor, calling the service processor to take the failed processor physically offline, issuing the ACR-complete message, and resetting the flags to indicate that ACR is complete.

CHAPTER 2. INTRODUCTION TO ALTERNATE CPU RECOVERY

Alternate CPU recovery (ACR) is the process by which the operating system dynamically adjusts to the unexpected failure of a processor in a multiprocessing configuration. ACR saves as much work from the failed processor as possible, and terminates work in progress as an abnormal termination condition. This allows ACR to attempt software recovery through the use of recovery and retry routines defined in the system at the time of the malfunction in the failed processor. Any tasks with processor affinity to the failed processor are terminated when they are redispatched.

OPERATING ENVIRONMENT

When the system is running in an MP environment, it must prevent programs running on different processors from simultaneously assigning resources, such as storage, or updating critical data fields, such as ASM control blocks. The operating system does this by assigning a lock to each resource or data field, and by establishing a protocol whereby a program must obtain the appropriate lock before it acquires the related resource or updates the related data area. Since a lock can be held by only one program at a time, it prevents the simultaneous use of its resource by programs on different processors.

When a program on one processor needs to use a resource but cannot obtain the related lock, it generally waits and keeps trying to obtain the lock. The locking protocol requires that a program hold a lock for as short a time as possible and that the program holding a lock be disabled (except for machine checks). This generally results in little or no waiting for a lock when the system is running normally.

However, various hardware and software malfunctions can cause a processor to stop processing while the program running on that processor holds one or more locks. Since it is inevitable that some program running on the other processor will need one of these locks, the system will "hang" unless a way can be found to identify and free (on the running processor) the locks held on the failed processor.

ACR PROCESSING

ACR consists of three phases: pre-processing, intermediate, and post-processing. See Figure 1 on page ACR-13 for an overview of ACR processing.

PRE-PROCESSING PHASE

When one of the processors in the system has a malfunction and is about to stop processing, it issues either a malfunction alert (MFA) for a hardware failure or an emergency signal (EMS) for a software failure. The MFA or EMS interruption handler then

- invokes ACR's pre-processing phase on the recovery (healthy) processor to mark the failed processor offline
- saves the failing CPU's system mask
- saves the failing CPU's work unit status in its ACR work/save area
- sets up control block pointers for later recovery processing
- sets the ACR flags.

When this processing is complete, control returns to the dispatcher that applies its normal algorithms to decide what processing should now be resumed on the recovery processor. The failed processor is logically offline and assumed to be not operating. The recovery processor and the failed processor are referred to as the "ACR-pair."

INTERMEDIATE PHASE

The dispatcher or a system routine that cannot obtain a global resource initiates ACR's intermediate phase. The system routine determines that the required resource is held by a processor other than the one on which the routine is running, and that ACR is in progress.

ACR's intermediate phase (running on the recovery processor) switches some control block pointers and data fields to make the system think that it is running on the failed processor. ACR saves the current processor's work unit status in its ACR work/save area and then copies the other processor's work unit status from its ACR work/save area to the current processor. ACR then simulates a machine check, to get MCH to perform hardware recording. ACR exits from this phase, still simulating the failed processor, to RTM which initiates execution of the failed processor's FRRs. This frees all system resources held on the failed processor and allows the routine that initiated the intermediate phase to continue.

This "ping-ponging" between the two processors continues until both no longer hold any global resources and are ready to run enabled. At this point, the dispatcher initiates the call to ACR's post-processing phase.

POST-PROCESSING PHASE

The dispatcher initiates the ACR post-processing phase to complete ACR processing. ACR is not complete until the dispatcher calls ACR twice, once for the recovery processor, and once for the failed processor. Each entry indicates that the processing on the related processor is enabled. When the call is from the dispatcher, IEAVTACR determines if the other processor (the one not presently being simulated), is ready to run enabled. If it is, ACR starts its post-processing phase. If the other processor is not ready to run enabled, ACR executes another processor-switch to simulate the other processor and continue with its processing. When this processor is ready to be enabled, the dispatcher is entered and calls ACR for the second time. This call results in the execution of the post-processing phase.

If the failed processor did not hold any resources when it failed, the recovery processor never would have needed to initiate ACR's intermediate phase. In this case, the termination of ACR is initiated when the recovery processor next enters the dispatcher. The dispatcher still calls ACR and this call results in a processor-switch to the processor, the call to RTM for hardware processing, and subsequent exit from ACR to RTM (still simulating the failed processor). Control eventually gets back to the dispatcher which will make the second call to the ACR post processing phase. This second call results in a switch back to the recovery processor and execution of the post-processing phase.

ADDRESSING AND RESIDENCY MODES

Addressing mode determines whether a full word address is treated as either a 24-bit address or a 31-bit address. Addressing mode is determined by bit 32 of the PSW. For AMODE 24, a module must reside below the 16 megabyte line and can only reference data within this area of virtual storage. For AMODE 31, a module can reside anywhere and reference data anywhere in virtual storage.

Residency mode is the location in virtual storage where a module resides. A module's residency mode can be either 24 (its address is less than 16 megabytes) or ANY (it can reside anywhere in virtual storage).

All alternate CPU recovery (ACR) modules have an addressing mode (AMODE) of 31 and an residency mode (RMODE) of ANY.

Addressing and residency mode is explained in detail in the 31-Bit Addressing publication.

CHAPTER 3. CONTROL BLOCK OVERVIEW (NOT APPLICABLE)

There is no control block information for this component.

CHAPTER 4. PROCESS FLOW

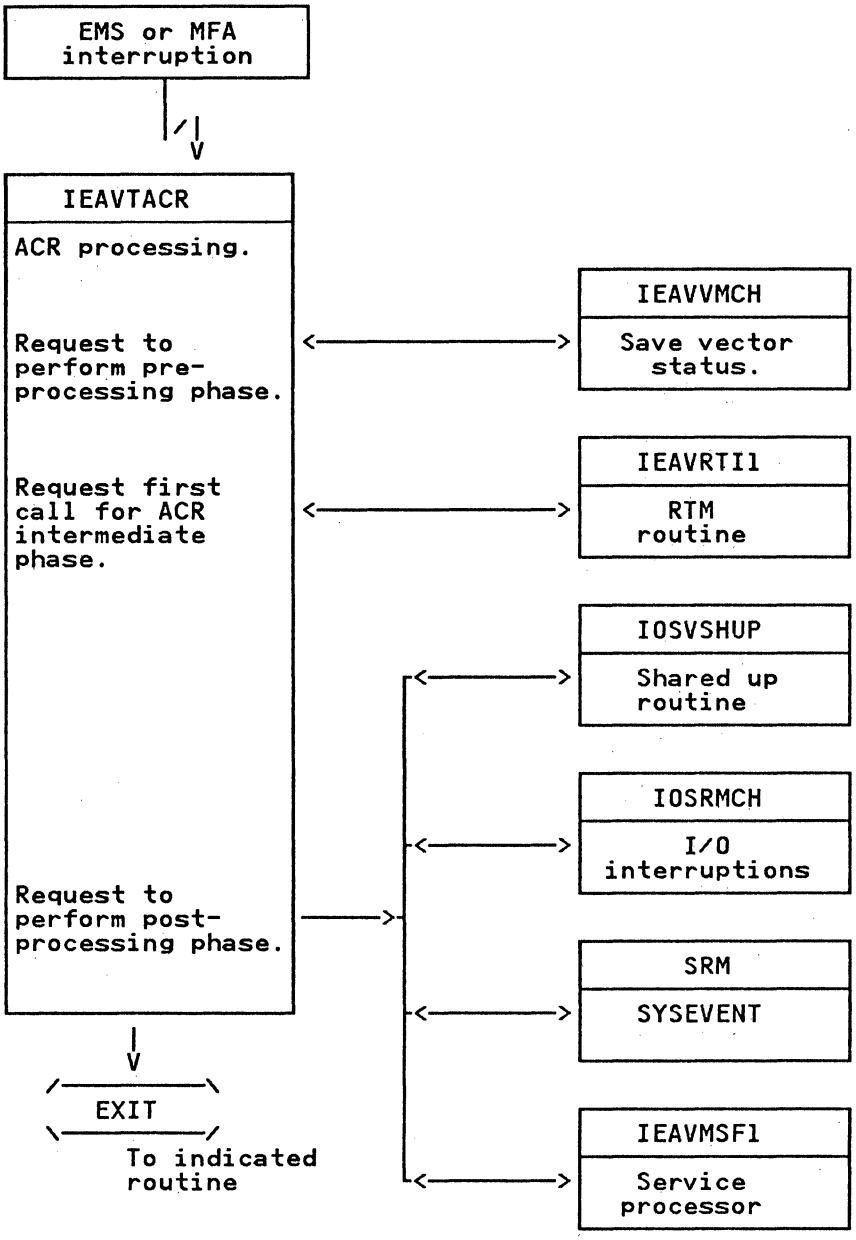


Figure 1. ACR Processing Overview

CHAPTER 5. METHOD OF OPERATION

This section contains method of operation diagrams in the HIPO format. HIPO diagrams are arranged in an input-processing-output format. The left side of the diagram contains data that serves as input to the processing steps in the center of the diagram, and the right side contains the data that is output from the processing steps. Each processing step is numbered; the number corresponds to an amplified explanation of the step in the "Extended Description" box. The object module name and labels in the extended description point to the code that performs the function.

The following figure shows the symbols used in HIPO diagrams. The relative size and the order of fields in control block illustrations do not always represent the actual size and format of the control block.

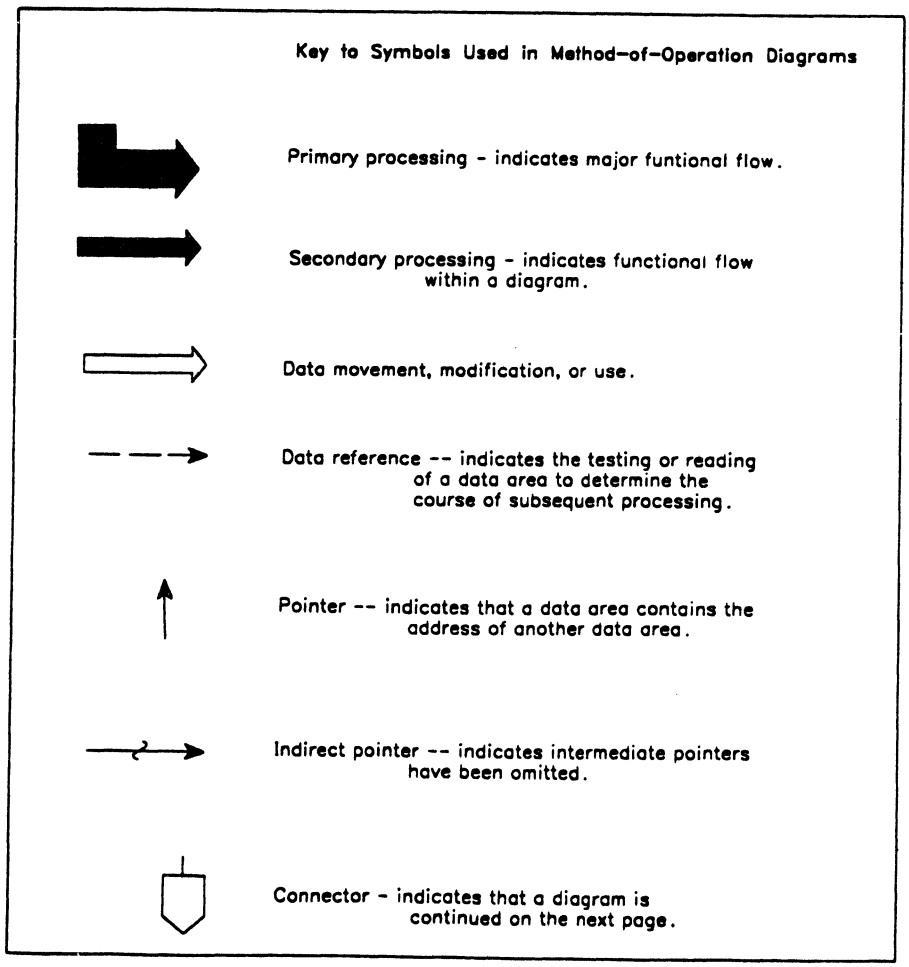
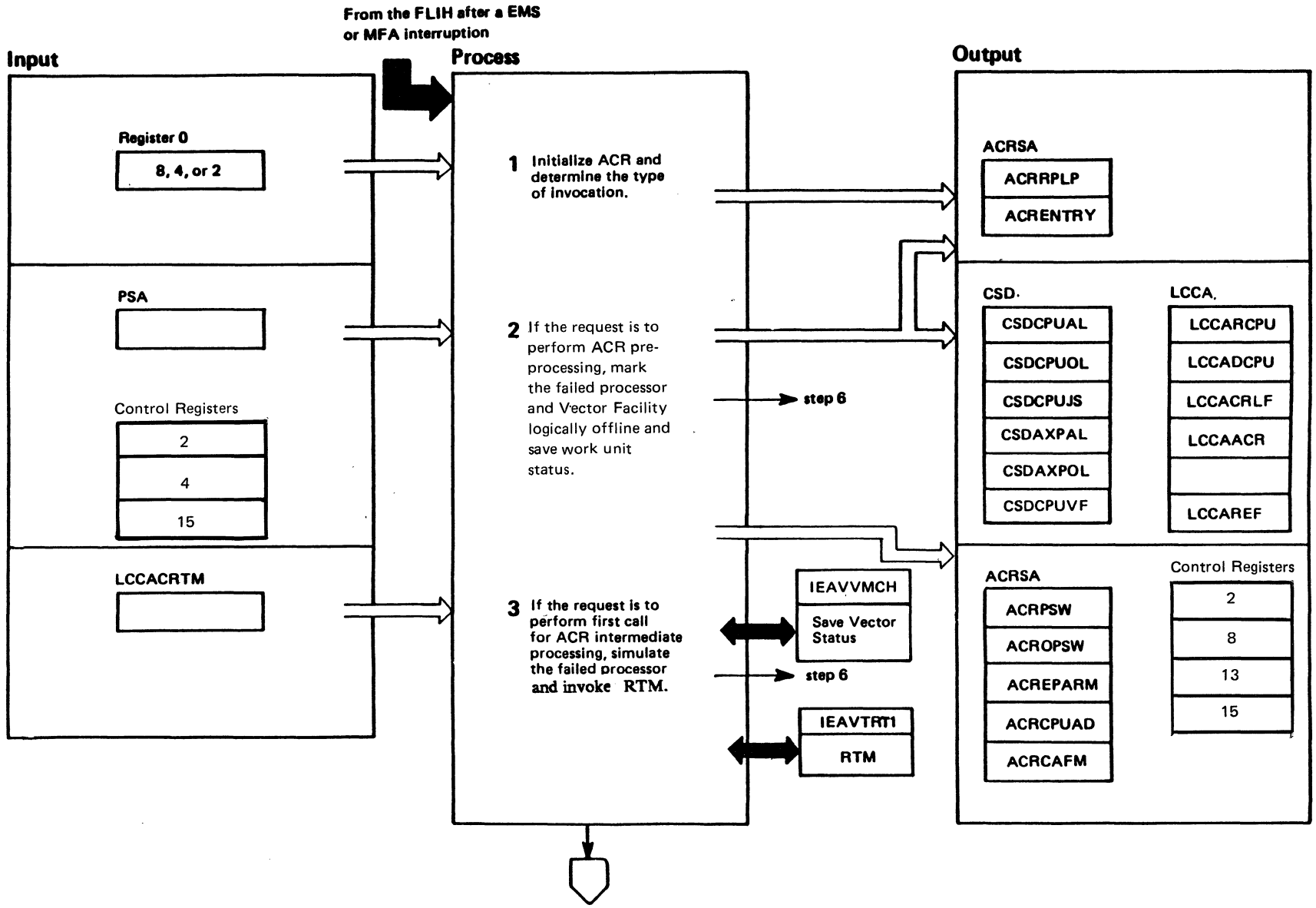


Figure 2. Key to HIPO Diagrams

Note: Brief alternate CPU recovery module descriptions appear in Component Diagnosis: Module Descriptions, which contains module descriptions for all the system components described in the Component Diagnosis and Logic books.

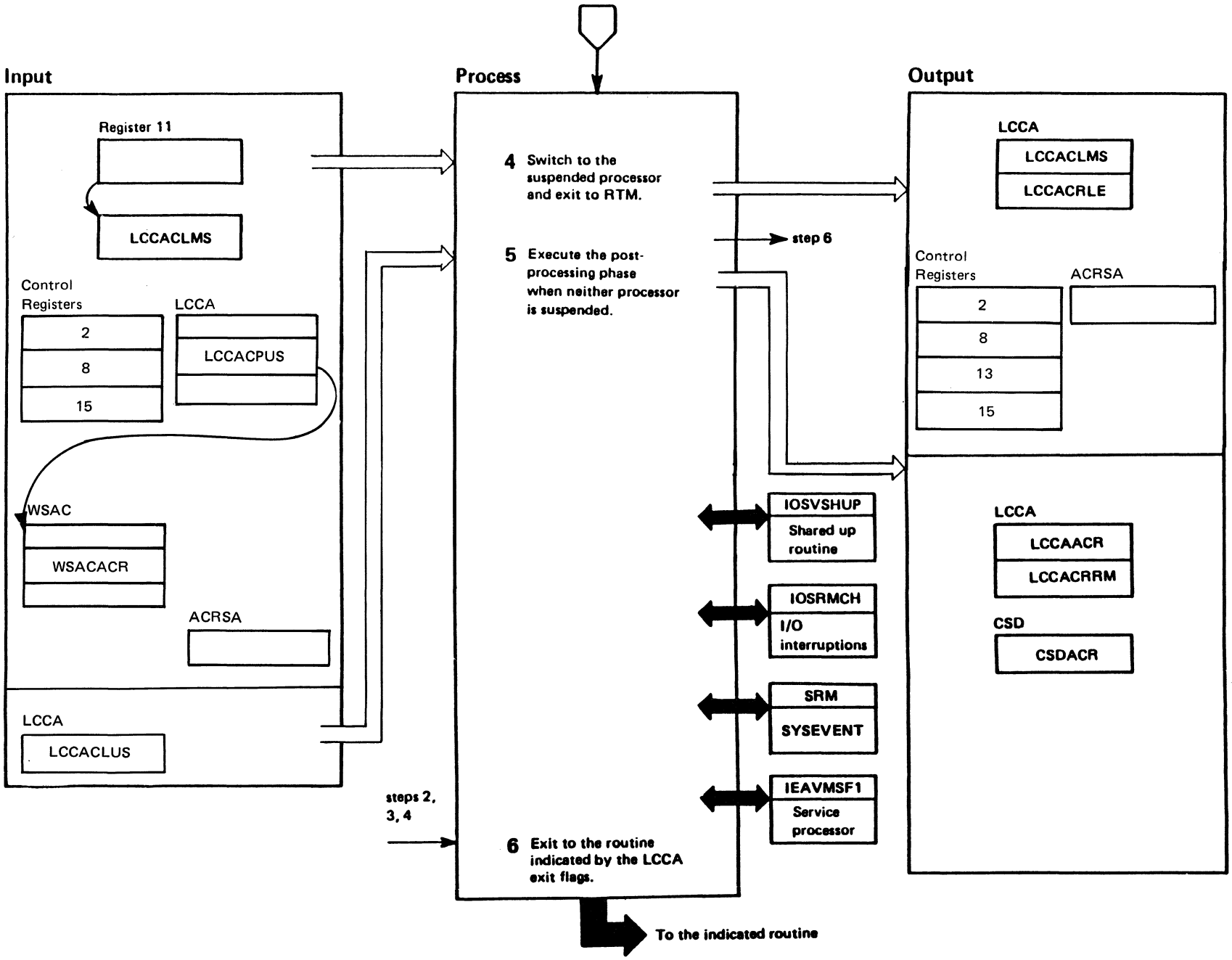
Alternate CPU Recovery (ACR) Overview (Part 1 of 4)



Alternate CPU Recovery (ACR) Overview (Part 2 of 4)

Extended Description	Module	Label
Alternate CPU recovery (ACR) recovers the system on another processor when one processor in a multiprocessing environment fails.		
1 IEAVTACR sets up addressability to the ACR work areas, saves the caller's registers and return address, and determines whether this is a call for the ACR pre-processing phase.	IEAVTACR	ACRSTART
2 When the call is for ACR's preprocessing phase, IEAVTACR marks the failed processor and associated Vector Facility offline to MVS, creates a trace entry to identify the failed processor and that ACR has begun, initializes the ACR fields in the LCCAs of both the recovery and failed processor, and saves data from the failed processor's PSA in the failed processor's ACR work area. ACR then returns to the EMS or MFA interruption handler (step 6).		ACRPREP
3 When the call is for ACR's intermediate processing, IEAVTACR sets the LCCA suspend bit for the processor that ACR is presently simulating. When the call is the first call for the intermediate phase, ACR switches control block data so that the recovery processor is simulating the failed processor. This "processor-switch" consists of saving vector status and saving some of the recovery processor's PSA fields and control registers in the recovery processor's ACR save area, and replacing them with the corresponding data for the failing processor. This data includes hardware data and the base pointers to processor related control blocks such as the LCCA, PCCA, and FRR stacks. This means that although processing is always on the recovery processor (because the other one has failed), processor-switching can make the system recovery routines think they are running on either the recovery or the failed processor.		ACRPROSW
	IEAVVMCH	
After ACR has switched to simulate the failing processor, ACR simultes a machine-check to get control to RTM to perform hardware-recording and recovery processing for the failed processor. On return from RTM, ACR goes to step 6 to exit to RTM. The process-switch will cause RTM to think that it is running on the failed processor, and to initiate execution of the FRRs of that processor.		ACCRTM

Alternate CPU Recovery (ACR) Overview(Part 3 of 4)



Extended Description	Module	Label
4 When the call is not the first entry to the intermediate phase, IEAVTACR could be simulating either the recovery or the failed processor. If the LCCA suspend bit of the processor that ACR is *not* simulating is set, IEAVTACR does a processor-switch to simulate that processor, clear its suspend bit, and go to step 6 to exit. Since ACR has just done a processor switch, this exit is not to the caller that initiated this pass through ACR. It is to the caller of ACR on the processor that ACR just switched to.	IEAVTACR	ACRLRRPT
5 When ACR is called and neither processor is suspended, IEAVTACR executes its post-processing phase. ACR first ensures that is is simulating the recovery processor. ACR calls the IOS shared up routine (IOSVSHUP) to set the UCB for possible reserve/release processing, and calls the IOS machine-check exit routine (IOSRMCH) to process any I/O machine checks. ACR then issues a SYSEVENT to inform SRM that a processor has gone offline, calls the service processor (IEAVMSF1) to take the failed processor physically offline, issues the ACR-complete message to the operator, and resets the ACR flags.		ACRPOSTP
6 IEAVTACR exit processing consists of testing the ACR exit flags in the LCCA and exiting to the indicated system routine. The routines to which ACR may exit are the Lock Manager or a Spin Loop routine (when LCCACRLE='1'B), the Dispatcher (LCCACRRM='1'B), the External (MFA or EMS) FLIH (LCCACREF='1'B), and RTM (LCCACRRT='1'B).		ACREXTR

INDEX

P

R

S

Special Characters

IBM / Technical Newsletter

This Newsletter No.	LN28-1334
Date	December 22 , 1989
Base Publication No.	LY28-1432-0
File No.	S370-37
Prerequisite Newsletters/ Supplements	None

MVS/ESA Component Diagnosis and Logic: Alternate CPU Recovery

MVS/System Product - JES3, Program No. 5685-002
MVS/System Product - JES2, Program No. 5685-001

This newsletter contains replacement pages for *MVS/ESA Component Diagnosis and Logic: Alternate CPU Recovery*. Before inserting any of the attached pages into *MVS/ESA Component Diagnosis and Logic: Alternate CPU Recovery*, read ***carefully*** the instructions on this cover. They indicate when and how you should insert pages.

Pages to be Removed	**Attached Pages to be Inserted***
Title Page - Edition Notice	Title Page - Edition Notice
iii - xii	iii - xiv

*If you are inserting pages from different Newsletters/Supplements and ***identical*** page numbers are involved, always use the page with the latest date (shown in the slug at the top of the page). The page with the latest date contains the most complete information.

A change to the text or to an illustration is indicated by a vertical line to the left of the change.

Summary of Changes

See the Summary of Changes for a list of the changes included in this newsletter.

Note: *Please file this cover letter at the back of the publication to provide a record of changes.*

IBM Corporation, Information Development, Dept. D58, Building 921-2
P.O. Box 950, Poughkeepsie, New York 12602

Printed in U.S.A.

MVS/ESA
Component Diagnosis
and Logic:
Alternate CPU Recovery

READER'S
COMMENT
FORM

LY28-1432-0

This manual is part of a library that serves as a reference source for systems analysts, programmers, and operators of IBM systems. You may use this form to communicate your comments about this publication, its organization, or subject matter, with the understanding that IBM may use or distribute whatever information you supply in any way it believes appropriate without incurring any obligation to you.

Note: ***Copies of IBM publications are not stocked at the location to which this form is addressed. Please direct any requests for copies of publications, or for assistance in using your IBM system, to your IBM representative or to the IBM branch office serving your locality.***

Possible topics for comment are:

Clarity Accuracy Completeness Organization Coding Retrieval Legibility

If you wish a reply, give your name, company, mailing address, and date:

_____

_____

_____

_____

What is your occupation? _____

How do you use this publication? _____

Number of latest Newsletter associated with this publication: _____

Thank you for your cooperation. No postage stamp necessary if mailed in the U.S.A. (Elsewhere, an IBM office or representative will be happy to forward your comments or you may mail directly to the address in the Edition Notice on the back of the title page.)

MVS/ESA Component Diagnosis and Logic: Alternate CPU Recovery

LY28-1432-0

S370-37

Reader's Comment Form

Cut or Fold Along Line

Fold and tape | Please Do Not Staple | Fold and tape

NO POSTAGE
NECESSARY
IF MAILED
IN THE
UNITED STATES

BUSINESS REPLY MAIL
FIRST CLASS PERMIT NO. 40 ARMONK, N.Y.

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Department D58, Building 921-2
PO Box 390
Poughkeepsie, New York 12602

Fold and tape | Please Do Not Staple | Fold and tape

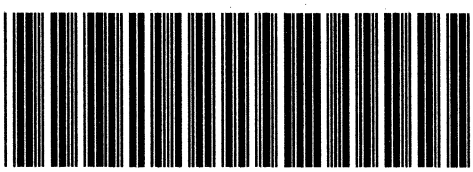
Printed in U.S.A.

IBM®

Program Number
5685-001

File Number
S370-37

LY28-1432-00



Printed in U.S.A.