Check Processing Control System

Installation Guide

Release 11

GA34-2178-08

Check Processing Control System

Installation Guide

Release 11

GA34-2178-08

Note!

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

| Ninth Edition (November 2000)

| This edition is a revision of GA34-2178-07. This edition applies to Version 1 Release 11 of the IBM Check Processing Control
| System licensed program (Program No. 5734-F11). This publication is current as of PTF numbers **UQ44297** and **UQ44298**. Each change is indicated by a vertical line (revision bar) in the left margin.

Information in this manual is subject to change from time to time. Before using this publication in connection with the operation of IBM systems, consult your IBM representative to be sure you have the latest edition and any Technical Newsletters.

IBM does not stock publications at the address below; any 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, IBM Payment Solutions, Department 58G, MG96/204, 8501 IBM Drive, Charlotte, NC 28262-8563, U.S.A. IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Contents

Part 2. Sample Problems

Figures

Notices

References in this publication to IBM products, programs, or services do not imply that IBM intends to make these available in all countries in which IBM operates. Any reference to an IBM product, program, or service is not intended to state or imply that only IBM's product, program, or service can be used. Any functionally equivalent product, program, or service that does not infringe any of IBM's intellectual property rights or other legally protectable rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, programs, or services, except those expressly designated by IBM, are the user's responsibility.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact: IBM Corporation, Department MG39/201, 8501 IBM Drive, Charlotte, NC 28262-8563, U.S.A. Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

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 Commercial Licensing, IBM Corporation, North Castle Drive, Armonk, NY 10504-1785 U.S.A.

Year 2000 Compliance

IBM announces that the Check Processing Control System, Version 1 Release 11, at PTF numbers UN99696 and UN99801, supports Year 2000. This IBM product, when used in accordance with its associated documentation, is designed to be capable of correctly processing, providing, and receiving date data within and between the twentieth and twenty-first centuries. This has been done by allowing the user to set the date format as a default throughout the system.

In the complex global computing environment that we have today, this IBM product's support for Year 2000 is, of course, dependent on the capabilities of all the other products that are working together (for example, hardware, software, and firmware) to properly exchange accurate date data.

Trademarks

The following are trademarks of International Business Machines Corporation in the United States and/or other countries:

IBM®, ACF/VTAM®, ES/9000®, MVS/DFP™, MVS/ESA™, MVS/SP™, RACF™, System/370™, 3090™.

Other company, product, and service names may be trademarks or service marks of others.

About This Book

This book tells you how to install the IBM Check Processing Control System (CPCS) with no modifications and how to run the sample problems. The sample problem exercises are included to help you verify correct operation after you install the software. You can also use this book and the *CPCS Customization Guide* when you are modifying and tuning the operation of CPCS.

This book contains information that was previously in the *Install Document* (member name INSTDOC) shipped with the CPCS product tape. Read the *Program Directory* that you receive with the product tape before you begin installation of the software to ensure that you have the most current information.

Who Should Read This Book?

This book is for people who are responsible for installing and testing CPCS. You can also use Part 2, “Sample Problems,” for training new users to your CPCS system.

How Is This Book Organized?

This book contains the following sections:

Part 1. Installation

- Chapter 1, “Preparing to Install CPCS,” describes installation tape contents and the prerequisite hardware and software that you need to install and operate CPCS. This chapter also describes installation options for Enhanced System Manager, logging, and expanded mass-data-set (MDS) records.
- Chapter 2, “Loading CPCS on Your System,” describes how to use the System Modification Program Extended (SMP/E) procedures to allocate your CPCS data sets and load the source code. It also includes special information to consider before using the SMP/E procedures.
- Chapter 3, “Installing CPCS,” describes the tasks you must perform to install CPCS software on your system, including allocating required data sets and generating program modules.

Part 2. Sample Problems

- Chapter 4, “Overview of the Sample Problems,” contains a summary of the sample problems and the general setup instructions.
- Chapter 5, “Sample Problem 1: Base Functions,” provides instructions for the most common types of CPCS processing.
- Chapter 6, “Sample Problem 1A: HSRR,” provides instructions for using High-Speed Reject Re-entry (HSRR) to capture marginal items rejected on prime pass.
- Chapter 7, “Sample Problem 1B: HSRR AND EMRG,” provides instructions for using High-Speed Reject Re-entry (HSRR) to capture marginal items rejected on prime pass, and for using Enhanced Merge (DKNEMRG) to substitute for the Merge task (DKNMRGE).

- Chapter 8, “Sample Problem 2: Enhanced Prime,” provides instructions for capturing multiple entries in a single sorter run, with each entry automatically becoming available for distribution, kill, and reporting, while the sorter is still running.
- Chapter 9, “Sample Problem 3: Enhanced Reject Processing,” provides instructions for using alternate reject pockets and consolidated reject strings to meet different transit deadlines.
- Chapter 10, “Sample Problem 3A: Enhanced Reject Processing with DKNEMRG,” provides instructions for using alternate reject pockets and consolidated reject strings to meet different transit deadlines, and for using Enhanced Merge (DKNEMRG) to substitute for the Merge task (DKNMRGE).
- Chapter 11, “Sample Problem 4: Unqualified Data,” provides instructions for CPCS processing of entries with blank amount fields. The online reject re-entry task (DKNOLRR) is used to simulate a key entry system for inscribing the correct amounts.
- Chapter 12, “Sample Problem 5: Divider Re-synchronization,” provides instructions for using divider documents to resynchronize data during a subsequent pass.
- Chapter 13, “Sample Problem 6: Base Functions with Expanded MDS,” provides instructions for CPCS processing with an expanded MDS record.

Appendix

- Appendix A, “Macro and Member Lists,” lists the CPCS macros and copy members that are intended as general-use programming interfaces.

This book also contains a bibliography and an index.

Related Publications

The following publications contain information that relates to the IBM Check Processing Control System (CPCS). See “Bibliography” on page X-9 for a list of books that contain information on other IBM products used during the installation of CPCS.

- *CPCS General Information*, GH20-1008

  This manual gives a general introduction to the Check Processing Control System (CPCS). It describes various features and advantages of CPCS and the hardware and software requirements for operating CPCS. It also discusses CPCS support of the IBM 3890 Document Processor and the 3890/XP Series document processors, along with some of the features of these processors.

- *CPCS Online Adjustments Guide*, GC31-2723

  This guide provides the program descriptions and terminal operation instructions for online adjustments. It includes information about customization, system and user requirements, the user adjustment-code data set, the adjustment-record formats, and sample reports.

- *CPCS Customization Guide*, SC31-2853

  This guide provides customization information for CPCS programmers, including system-programming information, generation procedures, and installation procedures.

- *CPCS Programming and Diagnostic Guide*, SC31-2854

  This guide contains guidelines for CPCS programmers, including descriptive information about application-program processing, problem analysis and documentation procedures, and CPCS module descriptions.

- *CPCS Terminal Operations Guide*, SH20-1229

  This guide describes the CPCS tasks and task initiation formats, and explains how to perform these tasks for the CPCS operators. It also explains application-task commands and supervisor commands. CPCS messages, which used to appear in this book, are now in *CPCS Messages and Codes*.

- *CPCS Messages and Codes*, SC31-4004

  This manual contains terminal and supervisor messages, their responses, and return code information for CPCS application tasks.

- *CPCS Master Index*, SC31-2857

  This reference combines the index entries for all the publications in the CPCS library.

- *CPCS Enhanced System Manager User's Guide*, SC31-4002

  This manual contains the guidelines for the CPCS personnel who use the Enhanced System Manager subsystem. It explains the Enhanced System Manager's interface functions, online functions, and processing. This publication is shipped with the Enhanced System Manager (ESM) product.

CPCS Web Site

Visit us at our web site:
www.ibm.com/products/cpcs

Summary of Changes for GA34-2178-08

- **PTF Number:** This publication is current as of PTF numbers **UQ44297** and **UQ44298**.
- **Tracer Data Set:** Removed the sorter restart information from the tracer data set and placed it in its own VSAM file to make all RSCB and user restart information easily available.
- **Electronic Transaction Support:**
  - Enabled the ETOT profile to determine the name and size of the transmission file that will be created.
  - Electronic Transaction support user exits now use the User Exit Facility.
  - Enabled the ETIN read and write user exits to insert items.
  - HSRR strings are now supported for ETUT.
- **DKNICRE Support:** Added extended string support to DKNICRE.
- **MICR Support:** MICR support for an SPDEF HH record for HSRR hardware definitions has been added.
- **Auto-startable Tasks:** DKNCLSM is now automatically started through ESM.
- **User Exits:**

  - DKNATASK user exit that allows supervisor terminal messages to be modified.
  - DKNATASK user exit that provides an ENQ/DEQ facility for vendor/user requirements.
- **M-string Support:** Added subsequent pass M-string support to DKNSLST.

Summary of Changes for GA34-2178-07

- **PTF Number:** This publication is current as of PTF numbers UQ39721 and UQ39722.
- **User Exits:** User exits have been added to DKNMICR, DKNATASK, and DKNETCFU.
- **Extended String Support:** Extended string support has been added to the EMRG, SCAT, and MDIS tasks.
- **Auto-startable Tasks:** Tasks ECYC, ICRE, and MCRE have been made auto-startable.
- **OCO Modules:** Various MICR task modules have been made OCO.
- **New Supervisor Command:** Supervisor command SYSLEVEL has been added. This command returns the latest PTF maintenance level applied to CPCS.

Summary of Changes for GA34-2178-06

- **PTF Number:** This publication is current as of PTF numbers UQ34786 and UQ34787.
- **CPCS Password Protection:** CPCS provides password protection to ensure it can be executed only for customers who are registered with IBM. IBM does provide a preset grade period for CPCS customers; during this grace period, CPCS can be started without a password. However, once this grace period expires, CPCS cannot be started without a valid password. Customers must use this grade period to contct IBM and to obtain a valid CPCS password.
- **System Manager:** Only Enhanced System Manager (not System Manager) is supported by CPCS Version 1 Release 11.
- **Subset Processing:** All subset processing has been disabled from CPCS; however, all subset processing features have not been removed from CPCS, for example, in screens, settings, etc. All these features will be removed in the near future.
- **CPCS Transaction Charging:** The CPCS License Agreement requires that any transaction that a customer introduces into CPCS must now use the transaction charging output modes when writing the transactions to the CPCS mass data set. The output modes and the transaction charging guidelines are detailed in the *CPCS Programming and Diagnostic Guide*, Chapter 2, “Accessing the Mass Data Set and Its Index.”
- **Sample Problems:** All the sample problems in the *CPCS Installation Guide* have been rewritten so the sort patterns fit on a six-pocket sorter.
- **Exit Point for MTASK:** An exit point has been added for MTASK (DKNMTASK_MICR_ABEND_EXIT0100).

- **Electronic Cash Letter Support:** A new module (DKNETCSH) creates electronic cash letters. Three new user exits support this function.
- **Electronic Transaction Data Matching:** This feature allows an institution to match the codelines of items from strings within a CPCS environment. The task matches strings created through a CPCS MICR task against strings that were either created as part of the initial MICR capture or strings that were imported to CPCS using the electronic transaction import process.

Summary of Changes for GA34-2178-05

Extended String Support: CPCS supports strings with records having one or more associated user areas.

Electronic Transaction Input/Output: The term ETIO refers to the importing and exporting of electronic transactions. It includes the following new CPCS features: electronic transaction output (ETOT), electronic transaction input (ETIN), and electronic transaction input/output display (ETIO).

Electronic Transaction Output: The ETOT task allows a financial institution to export strings from CPCS to a sequential data set (known as transmission data sets). Strings can be extracted either individually, by endpoint, or can be grouped by using Enhanced System Manager.

Electronic Transaction Input: The ETIN task allows a financial institution to create strings within CPCS from a transmission data set. The electronic records can be in any format; consequently, the import process can load transmission data sets from various sources.

Electronic Transaction Input/Output Display: The electronic transaction input/out (ETIO) task allows a financial institution to display online the status of either electronic control file records or electronic string file records. The institution may then display and update or delete records on either of these files.

Electronic Transaction Utility: The electronic transaction utility (ETUT) task allows the loading and unloading of both standard strings and extended strings for testing and maintenance purposes.

Summary of Changes for GA34-2178-04

The main changes for this revision are:

Merge Before Main Support: Dividers can optionally precede kill bundle items in a capture. With a Sort Pattern Definition option, a merge feed document can be sprayed to each kill pocket before any main hopper item is processed. Also, when this feature is active, CPCS post-capture tasks associate kill bundles to their preceding dividers. Merge-Before-Main is only supported for expanded-sort types using stacker-select routines that include PROLOGX.

User Exit Facility: The CPCS user exit facility (UEF) is designed to isolate CPCS programs from user exit processing. To use this facility, a CPCS program simply calls the module DKNUEM at a defined exit point, and UEM does the rest.

User Area Manager: The CPCS User Area Manager (UAM) isolates user data from CPCS control blocks and other data. User data may be added to applicable control blocks and processed through UAM.

System Manager Module Name Changes: All system manager module names changed from “DKNSM*” to “DKNZM*”.

New MDS Exit Point: The mass data set FREE exit point (MDS_FREE_EXIT) allows multiple exits to be invoked during MDS string FREE processing. You may use these exits to deny a FREE SPACE request.

DKNPEXIT Profile Member: The DKNPEXIT profile member is a system profile data set that contains records that are used to specify user exits to be run at the specific exit point with which they are associated. More than one exit may be specified for a single exit point, and the exits are called in the order specified in this profile member.

Feature Disable/Disengage Option: If selected features have been initialized in the IREC and are subsequently disabled, you can have the sorter issue a disengage.

Removed the TRKP option from the CPCSOPTN macro.

Summary of Changes for GA34-2178-03

The main changes for this revision are:

- Allows for more customization of DKNOLRR.
- Allows you to generate application profile records to the DKNAPPL data set.
- Shows IBM sequence number changes.

Summary of Changes for GA34-2178-02

The main changes for this revision are:

Year 2000 Changes:

- **Date Customization:** CPCS now provides the ability to specify a date format at the system level (as a default). This format is propagated throughout CPCS in reports, screens, and in data sets.
- For more information regarding CPCS code compliance with Year 2000, see “Year 2000 Compliance” on page ix. For a summary of Year 2000 changes, refer to the special appendix on this subject in the *CPCS Programming and Diagnostic Guide*.

Enhanced Prime: CPCS supports the capture of multiple entries on a single prime pass without the use of subsets.

Task Groups: This enhancement allows multiple BLDL tasks to be grouped for performance tuning.

System and Application Profile Data Sets: These data sets are used to pass information to programs now and in the future. These profiles contain configuration and run-time option information for the CPCS system and application programs.

MICR JAM Enhancement: You can configure the MICR task to have a refreshed enhanced jam screen displayed at the end of a runout on the 3890/XP.

Sequence Number Assignment: When the **P** record in the sort pattern specifies an **XF** sort, the item sequence number is returned to CPCS from the 3890/XP for each item processed.

PTF Numbers: This publication is current as of PTF Numbers UN99696 and UN99801.

New Web Site: Visit us at our new web site:

`www.ibm.com/products/cpcs`

Enhanced System Manager Support: CPCS supports the IBM Enhanced System Manager feature, which provides workflow management functions including task starting (based on workflow, time of day, after CPCS End Cycle, after Cold Start, after Warm Start, etc.), task tracking (auto-started and manual, Task Suppression (deadline management), Unit of Work (UOW) functions, and automatic generation of workflows.

Summary of Changes for GA34-2178-01

The main change for this revision is:

Enhanced System Manager Support: CPCS supports the IBM Enhanced System Manager feature, which provides work scheduler functions including task starting (based on workflow, time of day, after CPCS End Cycle, after Cold Start, after Warm Start, etc.), task tracking (auto-started and manual), Unit of Work (UOW) functions, and automatic generation of workflows.

Summary of Changes for CPCS Release 11

The new and enhanced functions in this release are in the following areas:

Automatic Restart: When a host or link failure occurs, CPCS restarts interrupted item-capture operations without the physical intervention of operators. The 3890/XP Control Program maintains a restart buffer where it stores a copy of the records that it sends to the host. When it is recovering from a host or link failure, CPCS retrieves the records contained in the restart buffer and replaces the intermediate buffer records that were lost when the failure occurred.

Blocked BDAM: To use disk storage space more efficiently, you can change the basic direct access method (BDAM) files that are used in CPCS to a blocked format. This capability can improve processing time, especially during startup, when initialization and compression of files occur.

Data-Set Duplexing: CPCS maintains the integrity of all data sets through the data-set duplexing (DUPLEX) function. CPCS duplexes all data sets that are critical to its operation. If a disk failure occurs, you can reinitialize or re-create duplexed data sets through predefined procedures.

Enhanced Logging: CPCS logging has some significantly enhanced features for the automatic recovery of mass-data-set strings. These features include:

- Tracer data-set recovery.
- Automated tracking (history) of mass-data-set strings and logging data sets.
- Elimination of dedicated tape drives.
- Automated full mass-data-set recovery. (Recovery automatically determines which strings should be recovered.)
- Recovery of strings with different mass-data-set definitions. (Recovery performs data conversions to match the existing mass-data-set definition.)
- Recovery of multiple strings with one pass of the logging data set.
- Selective recovery of strings by a generic search capability.

Enhanced Reject Processing: CPCS provides for two types of reject pockets: system reject pockets and user-defined reject pockets.

Expanded Document Processor Read Record: The 3890/XP Series document processors give your sort program access to a larger read record. You can now expand the read record to contain as many as 12 header bytes and 244 data bytes. CPCS supports 15 logical fields, 14 of which are available to you. The document processors write data from the codeline into the first seven fields. CPCS uses field 8 during codeline data-match processing. Your sort program can set the remaining seven fields. For example, you can use the additional fields for:

- The depositor's account number
- An alternate account-numbering system
- A pass-pocket history of the pocket used on each pass
- Optical character recognition (OCR) data capture.

Full Document Image Capture: When you add the IBM 3897 Image Capture System to your 3890/XP Document Processor, CPCS can perform full document image capture. This system, when supported by the IBM ImagePlus High Performance Transaction System Application Library Services licensed program, lets you capture an image (both front and back) of each item in an entry. These images can assist in reject repair. You can also use them for amount capture, eliminating the need to encode amounts before entry. You can store the images for later use in back-end systems, such as exception-item processing, cycle sorting, and statement assembly.

Merged String Distribution: The merged string (***M-string***) distribution module (DKNMDIS) enables CPCS to distribute M-strings.

Online Adjustments: The Online Adjustments program lets you adjust the merge string created by merging the reject and the input strings. The resulting balanced merge string is then available for account posting and other purposes.

Online Adjustments includes the following functions:

- Adjustment entry
- Adjustment listing
- Trial balancing (including forced).

Power Encoding: CPCS supports the IBM 3892/XP Power Encoder feature. It encodes the amount data on your checks and the entire MICR codeline for MICR rejects. To ensure accuracy, the Power Encoder feature compares the codeline data from the prime pass with the data on the check that it is encoding. The 3892/XP verifies the data and stops processing when it reaches a set number of consecutive discrepancies. You determine this number during installation.

Part 1. Installation

Chapter 1. Preparing to Install CPCS

Review the information in this chapter and in Chapter 2, “Loading CPCS on Your System,” before beginning the installation process. If you have not used the IBM System Modification Program Extended (SMP/E) to install software, read “Using SMP/E” on page 2-1. For an overview of Check Processing Control System (CPCS) operation, concepts, and terminology, see the *CPCS General Information* manual.

SMP/E loads the files delivered on the product tape on your system. After SMP/E loads the files in the SMP/E libraries, you can begin the process of allocating data sets and generating program modules for CPCS. When the CPCS installation process is complete, you can run the sample problems described in Part 2 of this book.

Note: Before you install CPCS, contact your IBM representative about preventive service planning (PSP) information for CPCS that might have been added after the printing of this book.

If you are using the IBM ImagePlus High Performance Transaction System, you must identify its applications to CPCS in the DKNBLDL table. You can use the Resource Access Control Facility (RACF) to manage security requirements for High Performance Transaction System transactions. For information about adding application tasks to DKNBLDL, see the *CPCS Programming and Diagnostic Guide*. For more information about installation requirements for the High Performance Transaction System, see both the *High Performance Transaction System Planning Guide* and the *High Performance Transaction System Installation Guide* (shown in the “Bibliography” on page X-9).

Important!

The installation procedures described in this book create data sets with CPCS.V1R11 as the default high-level qualifier. This is the qualifier used throughout this book.

You can choose another qualifier based on the data set naming conventions for your system.

Tape Contents

The machine-readable material is distributed on a standard label, 9-track magnetic tape or a 3480 Magnetic Tape Subsystem cartridge tape. The format you receive depends on the feature number that you used to order the CPCS product.

Figure 1-1. Tape Descriptions

Tape Format	Feature Number	External Tape Label	Tape Volume ID
9/1600	9029	MVS Source	HK903
9/6250	9031	MVS Source	HK903
3480	9081	MVS Source	HK903

File 1 of this tape contains the SMP/E control statements in RELFILE format. The files on the tape are as follows:

Figure 1-2. Installation Tape Contents

File	Data Set Name	Contents
1	SMPMCS	SMP/E control file
2	HCHK110.F1	CPCS source, copybooks, and macros
3	HCHK110.F2	Sample source code
4	HCHK110.F3	JCL
5	HCHK110.F4	SMPJCLN
6	HCHK110.F5	Report Lists

Several sample programs are also provided. These programs are in source form; you can find them in the sample source library (referred to in this document as CPCS.V1R11.SAMPLIB) after you complete the SMP/E procedures. For an overview of the sample programs and general setup instructions, see “General Setup Instructions” on page 4-5.

Hardware and Software Requirements

You can install CPCS on any IBM System/370,* 308x, 4381, 3090,* or ES/9000* processor operating in extended architecture mode. There must be enough real storage to meet the combined storage requirements of the host operating system, access methods, and CPCS. CPCS requires a minimum region size of 4 megabytes (MB) but IBM recommends at least 8 MB. For a more detailed description of the minimum-required CPCS operational environment, see the *CPCS Customization Guide*.

For all the requirements, both hardware and software, for the installation of CPCS, refer to the *CPCS Program Directory*.

Customizing CPCS

CPCS is delivered from IBM with the logging subsystem inactive and with a standard mass data set (MDS) record length. The installation procedures in Chapter 3, “Installing CPCS,” create a basic system that is capable of running the sample problems. Additional customization is required to meet the needs of your particular financial institution.

Using Enhanced System Manager

The Enhanced System Manager (ESM) is not a part of CPCS but is an additional product that may be used to supplement CPCS. ESM manages the flow of work through your CPCS system. It uses functional units of work to represent mass data set strings and to control units of work to represent specific events (time of day, end cycle, etc.) to schedule individual task-starting and notification of manual tasks that need performing. It tracks these tasks to completion, providing easy task restarts and task reconciliation after a CPCS restart. Any authorized operator may use ESM to monitor the progress of work through the system, start new tasks, prevent tasks from starting, restart abnormally ended tasks through a flexible, online menu system. For more information about using the Enhanced System Manager feature, see the *Enhanced System Manager User’s Guide*.

Using Logging

The CPCS Logging function maintains the integrity of the CPCS mass data set (MDS) and its associated index. This subsystem records, or logs, to tape or disk all MDS activity. This process lets you reinitialize or re-create these data sets and restore logged data if a disk failure occurs during CPCS processing.

Note: CPCS is delivered with the Logging function inactive (LOG=NO in the CPCS system profile member DKNPCPCS). You can run the sample problems without the CPCS Logging function.

We recommend using the dynamic allocation function for your tape and disk data sets, document processors, and printers. This lets you avoid stopping CPCS processing, changing JCL statements, and restarting CPCS to release or regain control of those devices.

* Trademark of IBM

For more information about activating Logging functions for CPCS, see the *CPCS Customization Guide*.

Expanding the Mass Data Set

CPCS is delivered with a 50-byte MDS record length. You can tailor CPCS with an MDS record length that is larger than the 50-byte default size to meet the requirements of your financial institution.

Note: Only sample problem 6 requires a change in the MDS record length.

The following steps are required to create a system with an expanded MDS record length:

1. Use the MDX macro to create copybooks with expanded fields.
2. Assemble, or compile, and link-edit program modules affected by MDS changes.
3. Allocate expanded data sets.
4. Generate a new DKNMICR module.
5. Generate a new DKNMTASK module.

For details about tailoring CPCS with an expanded MDS record length, see the *CPCS Customization Guide*.

Chapter 2. Loading CPCS on Your System

This chapter describes how to allocate the data sets used by the System Modification Program Extended (SMP/E) and how to use SMP/E to load the Check Processing Control System (CPCS) into those data sets. All SMP/E steps assume the existence of a cataloged procedure that contains the data-definition (DD) statements that are needed when you run the SMP/E functions.

The following sections describe special considerations for installing this release with an existing CPCS system and for installing CPCS for the first time.

Using SMP/E

CPCS is packaged for installation with SMP/E. The sample JCL included in the *CPCS R11 Program Directory* should be used to complete the SMP/E installation of CPCS on your system.

Substitute your procedure name when you run the SMP/E jobs. If you do not have an SMP/E procedure, you can obtain one from either the Put tape or the installation productivity option (IPO) system. Instructions for creating a procedure are in the *System Modification Program Extended (SMP/E) Reference*. For examples of JCL for SMP/E procedures, see the *CPCS R11 Program Directory*.

Installing with an Existing CPCS System

If you have never used SMP/E to install CPCS software and you have an existing CPCS system, see the *CPCS R11 Program Directory* for examples of SMP/E procedures that you can use to establish the required libraries, catalogs, and data sets. You will need to use these procedures, with some changes, to prepare your system for the SMP/E installation process.

Warning: SMP/E installation is not designed to delete a previous level of CPCS. You must decide either to maintain multiple levels of CPCS or to replace an existing level.

Coexisting with a Previous Release of CPCS

Allocate and initialize a separate set of SMP/E libraries before you install CPCS to coexist with a previous level of CPCS. Otherwise, existing macro and module names will cause SMP/E installation to fail. You must also allocate a separate set of CPCS libraries before you start the SMP/E installation process. For examples of JCL that you can use to allocate and initialize the SMP/E data sets, see the *CPCS R11 Program Directory*.

Deleting Existing Levels of CPCS

The following JCL is in member SMPDELE. Figure 2-1 is an example of an SMP/E job for deleting an earlier release of CPCS, which was installed using SMP/E.

```
//JOB1 JOB ....
//RECEIVE EXEC SMPEPROC
//SMPPTFIN DD *
++FUNCTION (USERMODFMID).          (SEE NOTE BELOW)
++VER (Z038) DELETE(HMRXXXX).      (SEE NOTE BELOW)
//SMPCNTL DD *
    SET BOUNDARY(GLOBAL).
    RECEIVE.       (NOTE: CONDITION CODE 4 IS NORMAL)
/*

//JOB2 JOB ....
//APPLY EXEC SMPEPROC
//SMPCNTL  DD *
    SET BOUNDARY(TARGET1).
   APPLY S(USERMODFMID) C(ALL) DIS(WRITE).
/*                     OREMOVE 'DIS(WRITE)'
//JOB3 JOB ....
//ACCEPT EXEC SMPEPROC
//SMPCNTL  DD *
    SET BOUNDARY(DLIB1).
   ACCEPT S(USERMODFMID) C(ALL) DIS(WRITE).
/*                      OREMOVE 'DIS(WRITE)'
NOTE:  USERMODFMID = SOME UNIQUE USER DEFINED FMID
       HMRXXXX     = FMID OF EARLIER RELEASE BEING DELETED
```

Figure 2-1. Sample JCL to Delete Previous Release of CPCS

Installing CPCS for the First Time

If you are installing CPCS for the first time, allocate and initialize a set of SMP/E libraries to be used by SMP/E. You can use an existing set of SMP/E libraries, or you can allocate a set of libraries specifically for this product. For examples of JCL that you can use to allocate and initialize these libraries, see the *CPCS R11 Program Directory*.

Installing User Modifications Using SMP/E

For an explanation of how to convert the installation of user modifications, see the *System Modification Program Extended (SMP/E) Program Packaging Guide*.

Chapter 3. Installing CPCS

The SMP/E procedures described in the *CPCS Program Directory* tell you how to copy the CPCS files from the product tape to your system. The installation procedures described in this chapter tell you how to allocate the required CPCS data sets and how to assemble, or compile, and link-edit the program modules.

The steps for installing CPCS are:

1. Run the MDX macro to create CLIB.
2. Modify the Assembler proc.
3. Assemble and link-edit called Assembler source.
4. Assemble and link-edit base Assembler source.
5. Assemble and link-edit DKNMICR Assembler source.
6. Select and modify the COBOL proc.
7. Compile and link-edit COBOL source.
8. Assemble and link-edit the DKNMICR module.
9. Assemble and link-edit the exits and SCI programs.
10. Create the ABA and BCF files.
11. Allocate the CPCS data sets.
12. Allocate the duplex data sets.
13. Define the VSAM data sets.
14. Generate the system profiles (SYSTPROF)
15. Changing the CPCS System Profile Member DKNPCPCS
16. Generate the application profiles (DKNAPPL)
17. Copy the endpoint tables to the DKNEP data set.
18. Copy the sort-pattern definitions to the DKNSPDEF data set.
19. Initialize the item-sequence-number data set.
20. Generate the system Help file.
21. Generate the system Message file.
22. Create Adjustments code file.
23. Add the CPCS load library to the APF list.
24. Add DKNMTASK to the program property table.

Note: The installation procedures described in this chapter create a system that you can use to run the sample problems described in Part 2, “Sample Problems.” Additional customization is required to create a system that meets the needs of your financial institution. For information about modifying CPCS operation, see the *CPCS Customization Guide*.

Step 1: Run the MDX Macro to Create CLIB

This step is required. It allocates CPCS.V1R11.CLIB and updates it with members that have the indicated field sizes. The JCL is in member GENCLIB of CPCS.V1R11.CTRL.

This JCL creates the standard expansion configuration. If the standard mass data set LRECL value is not changing, the MDX macro has no operands. If you are expanding the mass data set from the standard 50-byte size, include MDX operands for the fields you want to change. If you do not want a dash (–) in field 5 for online reject reentry (OLRR) and Online Adjustments, edit the MDX macro by changing the value of &DASHDSP from 1 to 0.

For details about MDX operands, see the *CPCS Customization Guide*.

Step 2: Modify the Assembler Proc

In this step, you must modify HLAPROC (Proc for High Level Assembler) for your use.

Modify the proc using the high-level qualifiers, device names, etc., being used by your site.

Step 3: Assemble and Link-Edit the Called Assembler Source

In this step, you assemble and link-edit the Assembler source modules you will use during the link-edit in "Step 4: Assemble and Link-Edit the Base Assembler Source." The JCL is in the member DKNJASM1 of CPCS.V1R11.CTRL.

This step must run before any other assembly or compile procedures.

Step 4: Assemble and Link-Edit the Base Assembler Source

In this step, you assemble and link-edit Assembler source modules used for this release of CPCS. The JCL is in the member DKNJASM2 of CPCS.V1R11.CTRL. The load modules are put in CPCS.V1R11.LLIB.

Note: If RACF is not installed, DKNSECRR will not successfully assemble. For user-exit interface descriptions and programming requirements, see the *CPCS Customization Guide*. No changes to the user exits are required to run the sample problems.

Step 5: Assemble and Link-Edit the DKNMICR Assembler Source

In this step, you assemble and link-edit Assembler source modules needed to run DKNMICR in this release of CPCS. The JCL is in the member ASSMMICR of CPCS.V1R11.CTRL. The load modules are put in CPCS.V1R11.LLIB.

Note: Return code 0004 is normal on the Link Edit step for programs where the LNKATTR=NCAL is specified.

Step 6: Select and Modify the COBOL Proc

In this step, you must select the COBOL proc COB370 (Proc for COBOL370 sites), and modify it for your use.

Modify the proc using high-level qualifiers, device names, etc., that are used by your site.

Step 7: Compile and Link-Edit the COBOL Source

In this step, you compile and link-edit all COBOL source modules for this release of CPCS. The JCL is in the member DKNJCOB of CPCS.V1R11.CTRL. The load modules are placed in CPCS.V1R11.LLIB.

Modify the JCL DKNJCOB to use the compile proc selected and modified in “Step 6: Select and Modify the COBOL Proc” on page 3-2.

Note: Use the CEEUOPT that was created in “Step 3: Assemble and Link-Edit the Called Assembler Source” on page 3-2. If modifications are needed to the CEEUOPT, the changes should be made and then reassembled with DKNJASM1.

Step 8: Assemble and Link-Edit the DKNMICR Module

In this step, you assemble and link-edit the DKNMICR module for this release of CPCS. The load module is placed in CPCS.V1R11.LLIB. The JCL is in the member DKNGMICR of CPCS.V1R11.CTRL.

Note: Return code 0004 is valid for step LKEDMGEN.

For more information about parameters for the CPCSRDR and CPCSOPTN macros, see the *CPCS Customization Guide*.

Step 9: Assemble and Link-Edit the Exits and SCI Programs

In this step, you assemble and link-edit the supplied sample user exit programs, along with the SCI and OLRR programs used in the sample problems. The JCL is in the member DKNJSAMP in CPCS.V1R11.CTRL. The load modules are put in CPCS.V1R11.LLIB.

Step 10: Create ABA and BCF Files

In this step, you use DKNLOAD to:

1. Allocate the data sets.
2. Format the ABA file in the DKNAB data set.
3. Format the BCF file in the DKNBCF data set.

The JCL is in member DKNGBCAB. Change XXXXXX to the volume identifier used by your installation.

For more information about the ABA and BCF files, see the *CPCS Customization Guide*.

Step 11: Allocate the CPCS Data Sets

In this step, you allocate the data sets needed to run CPCS. The JCL is in the member DKNGALLO of CPCS.V1R11.CTRL. Before you tailor the JCL, see the description in the CPCS system profile member DKNPCPCS parameters BLKSIZE, BLKSEG, SEGDEV, and MAXDA in the *CPCS Customization Guide*.

This job allocates all the required CPCS data sets, using the following default assumptions:

- They do not already exist.
- They are all put on volume XXXXXX.
  (Use a volume label defined for your installation.)
- They are all assigned a high-level qualifier of CPCS.V1R11.

Important!

You can install CPCS with the default qualifier of CPCS.V1R11, or you can change the qualifier to meet the requirements of your installation. If you change the qualifier, you must review all the JCL members referenced to ensure that you are using the correct qualifier before you run the jobs.

You can use a blocked BDAM format for the DKNKB and DKNMF data sets. For more information about creating those data sets using a blocked BDAM format, see the *CPCS Customization Guide*.

Step 12: Allocate the Duplex Data Sets

If you are not using data-set duplexing, continue with "Step 13: Define the VSAM Data Sets."

In this step, you allocate the CPCS duplex data sets. The JCL is in the member DKNGALDX of CPCS.V1R11.CTRL. These data sets are necessary for recovery of critical operating information if a disk failure occurs.

The JCL allocates the CPCS duplex data sets, using the following assumptions:

- They do not already exist.
- They are all on volume YYYYYY.

  Warning: Do not use the same volume for the duplex data sets and for standard CPCS data sets allocated by DKNGALLO. If they are on the same volume and a DASD failure occurs, you will lose both data sets.
- They are all assigned a high-level qualifier of CPCS.V1R11. (You can change CPCS.V1R11 to a qualifier selected for your system.)

If you use a blocked BDAM format for the DKNKB or DKNMF data sets, you must also use that format for their corresponding duplex data sets. For more information about using the blocked BDAM format with those data sets, see the *CPCS Customization Guide*.

Step 13: Define the VSAM Data Sets

In this step, you define the VSAM data sets for CPCS. The JCL is in the members DKNJALVS and DKNJALV2 of CPCS.V1R11.CTRL. This JCL allocates the VSAM data sets, using the following assumptions:

- They do not already exist.
- They are put on volume XXXXXX.
- They are assigned a high-level qualifier of CPCS.V1R11.

Step 14: Generate the System Profiles (SYSTPROF)

In this step, you generate the system profile records to the SYSTPROF data set. The SYSTPROF data set was allocated with the DKNGALLO JCL.

To generate the system profile records, modify the DKNGSYST JCL to meet your system requirements, and submit this JCL to be run.

Step 15: Changing the CPCS System Profile Member DKNPCPCS

In this step, you change the CPCS system profile member DKNPCPCS parameter for your system requirements. Run DKNJEOTP to edit the changes made to your system parameters.

For a description of the CPCS system profile parameters, see the “DKNPCPCS Profile Member” section in the *CPCS Customization Guide*.

Step 16: Generate the Application Profiles (DKNAPPL)

In this step, you generate the application profile records to the DKNAPPL data set. The DKNAPPL data set was allocated with the DKNGALLO JCL.

To generate the application profile records, modify the DKNGAPPL JCL to meet your system requirements, and submit this JCL to be run.

Step 17: Copy the Endpoint Tables to the DKNEP Data Set

In this step, you set up the endpoint tables for CPCS. The JCL is in the member DKNGEPT of CPCS.V1R11.CTRL.

Step 18: Copy the Sort-Pattern Definitions to the DKNSPDEF Data Set

In this step, you set up the sort-pattern definitions used during CPCS operation. The JCL is in the member DKNGSPDF of CPCS.V1R11.CTRL.

Step 19: Initialize the Item-Sequence-Number Data Set

In this step, you submit job DKNGISN to allocate and initialize the item sequence number data set (CPCS.V1R11.ISEQ.FILE). This data set is used to ensure unique sequence numbers during CPCS processing. The JCL is in the member DKNGISN.

DKNIISEQ uses as input an optional parameter that controls segmentation of sorter zero. This parameter, if used, must be 17 characters long; see “Customizing the Item Sequence File” in the *CPCS Customization Guide* for more specific information on how to set the item sequence number format and ranges. Also, for more information about the item-sequence-number data set, see the *CPCS Customization Guide*. For information about CPCS processing when using item sequence numbers, see the *CPCS Programming and Diagnostic Guide*.

Step 20: Generate the System Help File

In this step, you define the VSAM data set that contains the CPCS system help data. The JCL is in the member DKNGHELP of CPCS.V1R11.CTRL. This JCL allocates the VSAM data set CPCS.V1R11.HELP.FILE, using the following assumptions:

- It deletes the existing data set, if present.
- It is put on volume XXXXXX.

  Note: Only help screens for CPCS are to be loaded. If you have purchased the Enhanced System Manager (Product 5799-ESM), be sure to uncomment the DD card referring to CPCS.V1R11.SDKNSAM2 (DKNISMH1) before submitting this job.

The JCL then uses DKNHELPL to load the data set with the CPCS system help data, which is in the member DKNIHENU of CPCS.V1R11.SAMPLIB.

Step 21: Generate the System Message File

In this step, you define the VSAM data set that contains the CPCS system messages. The JCL is in the member GENSMSG of CPCS.V1R11.CTRL. This JCL allocates the VSAM data set CPCS.V1R11.DKNSMSG.FILE, using the following assumptions:

- It deletes the existing data set, if present.
- It is put on volume XXXXXX.

  Note: Only messages for CPCS are to be loaded. If you have purchased the Enhanced System Manager (Product 5799-ESM), be sure to uncomment the DD card referring to CPCS.V1R11.SDKNSAM2 (DKNSMMG1) before submitting this job.

The JCL then uses IDCAMS to load the data set with the CPCS system messages, which are in the member SYSMSG of CPCS.V1R11.SAMPLIB.

Step 22: Create the Adjustment Code Table

In this step, you create the adjustment code table (for use with the online adjustment feature). The JCL is in the member GENADJCD.

The //SYSIN statement in the JCL provides the data for the codes. The IEBGD utility creates the data set CPCS.V1R11.DKNADJCD from the input data supplied in the job stream.

Step 23: Add the CPCS Load Library to the APF List

Note: This step authorizes the DKNMTASK module and is performed by your system programmer.

In this step, you add entries for the load library data sets (CPCS.V1R11.LLIB and CPCS.V1R11.SDKNMOD1) to your system's authorized program facility (APF) list. You can add to the APF list in two separate ways:

1. You can add the entries for the load library data sets (CPCS.V1R11.LLIB and CPCS.V1R11.SDKNMOD1) to SYS1.PARMLIB (member IEAAPFxx). To make the changes effective using this method requires you to IPL the system.
2. You can use dynamic APF that does not require an IPL. In this case, the data sets (CPCS.V1R11.LLIB and CPCS.V1R11.SDKNMOD1) are added to SYS1.PARMLIB (member PROGxx).

For more information about APF list changes, see the *MVS/ESA Initialization and Tuning Reference*.

Step 24: Add DKNMTASK to the Program Property Table

Note: This step is performed by your system programmer.

When CPCS is running, you must ensure that DKNMTASK remains in real storage. To make DKNMTASK a storage-resident routine, and make the CPCS region non-swappable, do the following:

1. Add DKNMTASK to the Program Property Table (PPT), specifying X'20' in the program property attribute.
2. Add an entry in member SCHED00 of SYS1.PARMLIB for DKNMTASK, as shown in Figure 3-1.

Note: This does not prevent the region from being paged.

```
COL1|  COL8|
    |      |
    PPT    PROGNAME(DKNMTASK)
           NOSWAP
```

Figure 3-1. Example of SCHED00 Entry for DKNMTASK

For more information about using non-swappable tasks, see the *MVS/ESA Initialization and Tuning Reference*.

You can now run the sample problems to verify the CPCS installation.

If you are installing the logging subsystem, or expanding the MDS record length, follow the procedures for these options as described in the *CPCS Customization Guide*.

If you are installing the Enhanced System Manager feature, follow the installation options in the *CPCS Enhanced System Manager User's Guide*.

Part 2. Sample Problems

Chapter 4. Overview of the Sample Problems

The Check Processing Control System (CPCS) sample problems run on the system that you create when you follow the installation steps that are described in Chapter 2 and Chapter 3. The sample problems help you verify that you have correctly installed CPCS, show several functions available with CPCS, and provide new users with a way to learn the processing flow of CPCS.

The 3890/XP MVS Support Program supplies a simulated document-processor facility that lets you run both the prime- and subsequent-pass entries without a physical document processor. The simulator supports high-speed reject reentry (HSRR) as well as online reject reentry (OLRR) to let you fix data errors, and it also supports both standard and expanded format (XF) sorts.

The sample OLRR edit routines require that a MUPA[1] Expanded Work Area be available for the storage of temporary variables. This work area can be created by adding an `SSWORK` parameter to the CPCSOPTN macro in DKNMGEN, then submitting DKNGMICR to perform a MICR generation. For more information, see Chapter 3, “Installing CPCS” on page 3-1 and the “MICR Task Generation” section in Chapter 2, “Installation and Generation Procedures” of the *CPCS Customization Guide*.

Each sample problem is a complete entity and can be run separately. The sample problems include descriptions of the following:

- The control card definitions necessary to run DKNMTASK.
- Any required SCI or OLRR modules.
- The sort pattern definitions necessary to show the functions specific to the problem.
- Data for the specific problem.

  To demonstrate the reporting and balancing capabilities of CPCS, the data provided for each problem contains deliberate errors. You can correct these errors using balancing and adjustments programs such as CPCS’s DKNADJ, DKNTBAL, and DKNALST.

  Note: You can use the data supplied with the simulator as is (except for Problem 5) or you can encode it for use with a physical document processor. If you encode the data, do not include the periods (.) shown in the records.
- Operating instructions to guide you through the sample exercises.

  For more help with CPCS commands, see the *CPCS Terminal Operations Guide*.
- Output listings from the steps run in the problem.

  These let you verify the operation of the system after installation and correlate the out-of-balance conditions with the data errors.

[1] MICR user parameter area

Sample Problem 1: Base Functions

This problem shows the basic functions of CPCS. It includes:

1. Prime-pass entry with rehandle pocket
2. Item distribution reports
3. Online reject re-entry
4. Balancing reports
5. Subsequent-pass entry
6. Data generation for back-end application processing

Sample Problem 1A: HSRR

This problem shows how the basic functions of CPCS can be enhanced by using High-Speed Reject Reentry (HSRR) to capture marginal items rejected by Prime-Pass capture.

Sample Problem 1B: HSRR and EMRG

This problem is a follow-on to Sample Problem 1A and shows how you can substitute Enhanced Merge (EMRG) for MRGE. EMRG is designed to work with the Enhanced System Manager (ESM) feature of CPCS. EMRG can be manually started, as in this problem; however, it works best when auto-started by ESM.

Sample Problem 2: Enhanced Prime

This problem shows the capabilities of Enhanced Prime, which is the capturing of multiple prime-pass entries, stacked one behind the other, as a continuous stream of documents. Each such entry must use the same sort pattern.

The problem includes:

1. Prime-pass run consisting of three stacked entries
2. Automatic end of each entry, and automatic start of the next entry, whenever a new tracer group is read
3. Automatic start-up of distribution and kill
4. Online reject reentry
5. Merge
6. Subsequent-pass entry
7. Report generation
8. Data generation for back-end application processing
9. Completion of the cycle

The prime-pass run shows the processing of multiple entries. The sample problem does not pause between these entries. DKNMICR starts DKNDIST automatically at the end of each entry, even as it moves on to capture the next. DKNDIST starts DKNKILL automatically in this sample problem.

The OLRR, MRGE, and remaining steps can be performed as soon as each entry distributes, that is, they too can be performed while DKNMICR is still capturing entries.

Sample Problem 3: Enhanced Reject Processing

This problem shows the advantages of using multiple reject pockets. It shows you how to selectively process various pockets in a timely manner to meet transit deadlines. The problem includes the following functions:

1. Prime-pass entry with multiple alternate reject pockets and kill pockets that contain bad items.
2. Production of kill bundles and cash letters for a first deadline.
3. Subsequent pass of the rehandle data, including distribution, kill, and cash letter production for a second deadline.
4. Repair of the transit rehandle pocket, including distribution, kill, and cash letter production for a third deadline.
5. Repair and distribution of the on-us reject data
6. Creation of a consolidated reject string from the bad data in good pockets and repair of these items
7. Repair of the error items in the system reject pocket
8. Merge of all corrected data to a final M-string
9. Distribution of the final M-string to produce revised kill lists and cash letters
10. Backup of MDS data and completion of the cycle

Sample Problem 3A: Enhanced Reject Processing with EMRG

This problem is a follow-on to Sample Problem 3 and shows the advantages of using multiple reject pocket processing with EMRG. It uses the same simulated environment as Sample Problem 3 and also shows selective pocket processing with the use of EMRG. This problem includes the following functions:

1. Running of a prime-pass entry with multiple alternate reject pockets and kill pockets that contain bad items.
2. Production of kill bundles and cash letters for a first deadline.
3. Subsequent pass of the rehandle data, including distribution, kill, and cash letter production for a second deadline
4. Repair of the transit rehandle pocket, including distribution, kill, and cash letter production for a third deadline
5. Repair and distribution of the on-us reject data
6. Creation of a consolidated reject string from the bad data in good pockets and repair of these items
7. Repair of the error items in the system reject pocket
8. Merge of all corrected data to a final M-string
9. Distribution of the final M-string to produce revised kill lists and cash letters
10. Backup of MDS data and completion of the cycle

Sample Problem 4: Unqualified Data

This problem processes entries that have a blank amount field. It uses OLRR to simulate a key entry system for inscribing the amount. The problem shows the 3892/XP Document Processor power encoding function, under the following conditions:

1. None of the sample items contain an amount field.
2. The SCI program distributes the items to several alternate reject pockets.
3. You use OLRR to repair this data by adding the amounts.
4. You simulate power encoding of the repaired items during the subsequent pass.

Sample Problem 5: Divider Re-synchronization

This problem shows how you can use divider documents to re-synchronize data during a subsequent pass. Divider re-synchronization helps to greatly reduce the effects of *hand swapping*.

Note: This problem cannot be run using the simulator. You must use a physical document processor and encode the data on actual test documents.

Sample Problem 6. Base Functions with Expanded MDS

This problem is similar to Sample Problem 1, with the following exceptions:

- The expanded process control field (field 6) contains data to show its potential use in return item processing.
- The MDS record has two new fields, 9 and 12.
- The amount field (field 1) contains 16 digits.
- The SCI program sorts amounts over one million dollars to a special high-dollar pocket.

The problem includes the following functions:

1. Prime pass and distribution
2. OLRR error correction and merge of corrected data
3. Production of kill bundles and cash letters
4. Subsequent-pass processing
5. End-of-cycle processing

The XF sort POD and XF sort HSRR exercises show the expanded capture feature. The sort pattern provided does not capture extended data but does show the MICR XF screens and the MICR XF messages logged in the scroll data set.

General Setup Instructions

Part 1 describes how to create a CPCS system that is ready to run the sample problems. To match the sample output to validate your installation, run the sample problems before customizing your system.

To run the sample problems on any other CPCS system, use the source and JCL listed below to change your system. The *CPCS Customization Guide* contains additional information to assist you in this case.

Figure 4-1. Sample Problem Matrix

Description	**Source in CPCS.V1R11.SAMPLIB**	**Install JCL in CPCS.V1R11.CTRL**
CPCS system parameters	DKNPCPCS	DKNJEOTP - Batch Editor
DKNMICR generation (see note 1)	DKNMGEN	DKNGMICR and related jobs
Endpoint Table file (DKNEP) (see note 2)	EPT001, EPT002, EPT003, EPT031, EPT033, EPT061, EPT063	DKNGEPT
Endpoint name and address (DKNAB)	DKNIAB	DKNGBCAB
Bank Control File data set (DKNBCF)	DKNIBCF	DKNGBCAB
Sort Pattern Definitions file DKNSPDEF	SPTYP001, SPTYP002, SPTYP003, SPTYP004, SPTYP005, SPTYP006	DKNGSPDF
Stacker-select and User-edit routines (see note 3)	DKN0011R, DKN0011Z, DKN0012Z, DKN0013Z, DKN0031R, DKN0031Z, DKN0032Z, DKN0033Z, DKN0034Z, DKN0041R, DKN0041Z, DKN0042Z, DKN0043Z, DKN0051R, DKN0051Z, DKN0052Z, DKN0053Z, DKN0061R, DKN0061Z, DKN0062Z, DKN0063Z	DKNJSAMP
Execution JCL	DKNJRUN	
MVS Support Package simulator input	DKNT01MU, DKNT01SU, DKNT02SU, DKNT03SU, DKNT04SU, DKNT05SU, DKNT06SU, DKNT08SU, DKNT09SU	

Sample Problem Matrix Notes:

1. DKNMGEN:

   This DKNMICR generation task includes definitions for the following document processors:

   - Host-simulated
     - Three 3890 Model A document processors with the microfilm feature
     - One 3890 Model B document processor with the microfilm feature
     - One 3890 Model XP document processor with the microfilm feature
     - One 3890 Model XP document processor with the microfilm and image capture features
     - One 3892 Model XP document processor with the power encode feature
     - One (dummy) 3890 Model XP document processor with the microfilm and image capture features, for use with CSBU.
   - One channel-attached 3890 Model XP Document Processor with the microfilm feature.

   The exact definitions used are shown in Figure 4-2:

```
         TITLE 'CPCS - MICR TASK GENERATION'
*********************************************************************
*        READER SORTER 1
*
RDR1     CPCSRDR TYPE=3890-1,ATTACH=SIM,DDRDRIN=SORTER01,
               MODEL=A,MFILM=YES,ENDORSE=YES,INF=YES
*
*********************************************************************
*        READER SORTER 2
*
RDR2     CPCSRDR TYPE=3890-1,ATTACH=SIM,DDRDRIN=SORTER02,
               MODEL=B,MFILM=YES
*
*********************************************************************
*        READER SORTER 3
*
RDR3     CPCSRDR TYPE=3890-2,ATTACH=SIM,DDRDRIN=SORTER03,
               MODEL=XP,MFILM=YES,PEND=YES,IMAGE=YES
*
```

Figure 4-2 (Part 1 of 3). Sample Code for CPCS -- MICR Task Generation

```
*****************************************************************
*        READER SORTER 4
*
RDR4     CPCSRDR TYPE=3892-2,ATTACH=SIM,DDRDRIN=SORTER04,
               MODEL=XP,MFILM=YES,PEND=YES,POWER=YES
*
*****************************************************************
*        READER SORTER 5
*
RDR5     CPCSRDR TYPE=3890-2,ATTACH=CHANNEL,DDRDRIN=SORTER05,
               MODEL=XP,MFILM=YES,PEND=YES
*
*****************************************************************
*        READER SORTER 6
*
RDR6     CPCSRDR TYPE=3890-2,ATTACH=SIM,DDRDRIN=SORTER06,
               MODEL=XP,MFILM=YES,PEND=YES
*
*****************************************************************
*        READER SORTER 8
*
RDR8     CPCSRDR TYPE=3890-1,ATTACH=SIM,DDRDRIN=SORTER08,
               LDPN=08,MODEL=A,MFILM=YES,ENDORSE=YES,INF=YES
*
*****************************************************************
*        READER SORTER 9
*
RDR9     CPCSRDR TYPE=3890-1,ATTACH=SIM,DDRDRIN=SORTER09,
               MODEL=A,MFILM=YES,ENDORSE=YES,INF=YES
*
*****************************************************************
*        READER SORTER 10
*
RDR10    CPCSRDR TYPE=3890-6,ATTACH=EMICR,DDRDRIN=SORTER10,
               LDPN=10,MODEL=A,MFILM=NO
*
```

Figure 4-2 (Part 2 of 3). Sample Code for CPCS -- MICR Task Generation

```
*********************************************************************
*
*-------------------------------------------------------------------*
*    ADD ALL NEW READER SORTERS HERE                                 *
*-------------------------------------------------------------------*
*
*********************************************************************
*        READER SORTER 99
*
RDR99    CPCSRDR TYPE=3890-6,ATTACH=SIM,DDRDRIN=SORTER99,
               MODEL=XP,MFILM=YES,IMAGE=YES,LDPN=99
*
*********************************************************************
       EJECT
*********************************************************************
*        CPCS OPTIONS
*
OPTIONS  CPCSOPTN TRACER=(5555-5555,5),              -
               BLOCK=(5533-3333,5),                  -
               BATCH=(5566-6666,5),                  -
               SBATCH=(5599-9999,5),                 -
               DIVIDER=(4666-6666,5),                -
               CTLSEQ=PP,                            -
               BEXIT=BEGNEXIT,                       -
               MAXTG=50,                             -
               MAXRP=15,                             -
               MAXSCI=256K,                          -
               SSWORK=2040
         END
```

Figure 4-2 (Part 3 of 3). Sample Code for CPCS -- MICR Task Generation

These definitions are used, unchanged, for all sample problems.

2. EPTxxx:

   EPT001 endpoints are killed on prime pass and EPT002 endpoints are killed on subsequent pass. EPT003 combines EPT001 and EPT002 for convenience in producing cash letter summaries. EPT031 and EPT033 are variants of EPT001 and EPT003, designed for Sample Problem 3. EPT061 and EPT063 are variants of EPT001 and EPT003, designed for Sample Problem 6.

3. DKN####Z and DKN####R:

   These are source programs for the sample problem stacker-select and user-edit routines. All stacker-select routines have a Z as the eighth character of the module's name. The user-edit (OLRR) routines require an R as the eighth character of their names. Assemble and link-edit the following stacker-select and user-edit routines so that the load module name is the same as the source module name.

   Note: Sample Problem 2 uses the same SCI and OLRR modules as Sample Problem 1.

   DKN0011Z Prime-pass, stacker-select routine (non-XF sort) for Sample Problems 1 and 2

DKN0012Z	Subsequent-pass, stacker-select routine (non-XF sort) for Sample Problems 1 and 2
DKN0013Z	Subsequent-pass, stacker-select routine (non-XF sort) for Sample Problems 1 and 2
DKN0011R	OLRR validation routine (non-XF sort) for Sample Problems 1 and 2
DKN0031Z	Prime-pass stacker-select routine for Sample Problem 3
DKN0032Z	Subsequent-pass stacker-select routine for Sample Problem 3
DKN0033Z	Subsequent-pass stacker-select routine for Sample Problem 3
DKN0034Z	Subsequent-pass stacker-select routine for Sample Problem 3
DKN0031R	OLRR validation routine for Sample Problem 3
DKN0041Z	Prime-pass stacker-select routine for Sample Problem 4
DKN0042Z	Subsequent-pass stacker-select routine for Sample Problem 4
DKN0043Z	Subsequent-pass stacker-select routine for Sample Problem 4
DKN0041R	OLRR validation routine for Sample Problem 4
DKN0051Z	Prime-pass stacker-select routine for Sample Problem 5
DKN0052Z	Subsequent-pass stacker-select routine for Sample Problem 5
DKN0053Z	Subsequent-pass stacker-select routine for Sample Problem 5
DKN0051R	OLRR validation routine for Sample Problem 5
DKN0061Z	Prime-pass stacker-select routine for Sample Problem 6
DKN0062Z	Subsequent-pass stacker-select routine for Sample Problem 6
DKN0063Z	Subsequent-pass stacker-select routine for Sample Problem 6
DKN0061R	OLRR validation routine for Sample Problem 6

To run an exercise with a physical document processor:

- Inscribe the MICR documents as shown in the individual problems. Each document has a sequence number inscribed in the serial-number field of the MICR line. This facilitates offline sort back into the original sequence for repeated tests.
- Create the specific digit errors required for OLRR. One way to achieve this is to deface the MICR characters.
- Copy DKNMGEN from CPCS.V1R11.SAMPLIB and add CPCSRDR macros to define physical document processors. Each document processor can have as few as six pockets (one stacker module). For channel-attached sorters, specify ATTACH=CHANNEL and supply a DDRDRIN name; for LU 6.2-attached sorters, specify ATTACH=LU62 and supply an LUNAME. Regenerate MICR to incorporate the new CPCSRDR definitions into your system.
- For channel-attached sorters, add definitions either to your run JCL or to your DKNDSAT table. For LU 6.2-attached sorters, add definitions to your LNODE table and recompile it using your GENLNODE JCL.

Chapter 5. Sample Problem 1: Base Functions

This sample problem shows the basic functions of the Check Processing Control System (CPCS). It guides you through a sample run that shows the following:

- Data capture
- Distribution
- Online reject reentry (OLRR)
- Merging of the repaired data with the input data
- Producing kill lists and cash letters
- Subsequent-pass processing
- End-of-cycle processing

Preparing to Run the Sample Problem

The test material contains:

- DKNPCPCS parameters for DKNMTASK execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, batch, and subbatch slips
- SCI and OLRR definitions for sorting.

CPCS Parameter Generation Options

CPCS requires no special options in the DKNPCPCS System Profile control card for this sample problem.

DKNSPDEF Options

This sample problem uses member SPTYP001 in CPCS.V1R11.SAMPLIB as the sort-pattern definition.

```
*          TEST DECK:        DKNT01SU
*          MERGE FEED DECK:  DKNT01MU
*          ENTRY:            0010
*
*      PPH    | SCI/N.L. EDIT |   OLRR EDIT
*  -----------|---------------|---------------
*  1-00-00-00 | DKN0011Z      | DKN0011R
*  2-03-00-00 | DKN0012Z      |
*  2-05-00-00 | DKN0013Z      |
*
************************************************************
*
P1000000DKN0011Z        021           20030            2
B   001
R    0303
K018888888802888888880588888888
*
************************************************************
*
P2030000DKN0012Z          2            0030
B   001
K0109000000021000000003110000000412000000
*
************************************************************
*
P2040000DKN0013Z          2            0030
B   001
K010210000202021302350302100030040210000805020000000
*
```

Figure 5-1. Sample Problem 1

Pocket Selection for Sample Problem 1

Figure 5-2, Figure 5-3 on page 5-3, and Figure 5-4 on page 5-3 describe the contents of each of the pockets for this sample problem. In the columns labeled Routing/Transit, the letter *n* can be replaced by any digit.

Figure 5-2 (Page 1 of 2). Sample Problem 1: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System reject pocket
01	Kill	11nn-nnnn	PC=AAAA33	On-us Credits

Figure 5-2 (Page 2 of 2). Sample Problem 1: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
02	Kill	11nn-nnnn	PC=AAAAAA	On-us Debits
03	R/H	12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn	N/A	First Deadline Transits
04	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transits
05	Kill	11nn-nnnn	EXT PC not = AA and PC=AAAAAA	On-us Debit Returns

Figure 5-3. Sample Problem 1: Pocket Selection Criteria - First Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System reject pocket
01	Kill	12nn-nnnn	N/A	First Deadline Transit EP=09000000
02	Kill	13nn-nnnn	N/A	First Deadline Transit EP=10000000
03	Kill	14nn-nnnn	N/A	First Deadline Transit EP=11000000
04	Kill	15nn-nnnn	N/A	First Deadline Transit EP=12000000
05	---	N/A	N/A	Not used

Figure 5-4 (Page 1 of 2). Sample Problem 1: Pocket Selection Criteria - Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System reject pocket

Figure 5-4 (Page 2 of 2). Sample Problem 1: Pocket Selection Criteria - Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02010002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Data for Sample Problem 1

The data for this sample problem consists of one set of tracers with associated data, including block, batch, and subbatch slips. This data also contains deliberate errors to show the balancing features of the system.

The member DKNT01SU in CPCS.V1R11.SAMPLIB contains this input. If you use the simulator for this sample problem, you must include the fields represented by periods as shown in the records. If you use a physical document processor, do not encode the fields represented by the periods.

Reports for Sample Problem 1

The reports that CPCS produces when you run this sample problem are in CPCS.V1R11.RPTLIB. Print out member SAMPLE1 from this library and compare it with your results for accuracy.

Operating Instructions for Sample Problem 1

1. Start CPCS by submitting the job in member DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminal by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

   ```
   MICR
   ```

   Open sorter 1 by entering:

   ```
   O 1
   ```

   on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 1.
6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 001, and entry number 0010. Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual compare.
7. End the prime-pass run when you see the `Intervention Required` message on the MICR status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the end command.
8. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the CLOSE command. The distribution task (DKNDIST) displays a message on the supervisor terminal indicating the end of the distribution task.
9. Use DKNOLRR to correct the rejects from the prime pass. The command to use is:

   ```
   OLRR 0010,01
   ```

   Three documents in the example contain digit errors. Correct the errors as follows:

   - R/T error: `12*122222` should be `122122222`
   - Amt error: `00000*0002` should be `0000010002`
   - R/T error: `11100006*7` should be `111000627`
10. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.

11. Start the string merge task (DKNMRGE) for the corrected and entry (on which SCAT was run) with the command:

    ```
    MRGE 1,0010
    ```

    DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the concatenated R-string produced by SCAT.

12. DKNMRGE then automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.

13. The Exception Entry Master List report, produced by DKNPLST, identifies one subbatch as being `OUT OF BALANCE`. The subbatch that is out-of-balance is the subbatch slip with the serial number of 51. Because this subbatch is out-of-balance, the batch and block are also out-of-balance.

    This out-of-balance condition is caused by the intentional entry of an incorrect amount ($100.02) during OLRR for this subbatch. ($.02 is the correct amount.)

14. Start the MICR subsequent pass for the documents sorted to the first rehandle pocket (pocket 03) on prime pass for this entry.

    a. Open sorter 1 and proceed to the Begin screen.

    b. On the Begin screen, enter:

       ```
       ,,0010-010
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

15. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

16. Start the MICR subsequent pass for the documents sorted to the second rehandle pocket (pocket 04) on prime pass for this entry.

    a. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

    b. On the Begin screen, enter:

       ```
       ,,0010-013
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

17. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

18. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the CLOSE command.

19. The distribution task started automatically when you ended each entry. Because the entries were on a subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.

20. Start the Kill List task (DKNKILL) for the first deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT001** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT001 contains the endpoints for each first deadline kill pocket.

21. Start the Kill List task (DKNKILL) for the second deadline items with the command:

    ```
    KILL 8
    ```

    Where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT002 contains the endpoints for each second deadline kill pocket.

22. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    Where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT003** when prompted for the table name. This table contains all the endpoints from all deadlines, both first and second. When DKNCLSM prompts you to respond to whether this is a duplicate letter, press **ENTER** to select NO.

23. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent-pass reject D-strings, and R-strings.

24. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

25. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

26. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system. After you run this task, the only string left on the mass data set is the M-string for the prime pass.

27. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures there are no active strings present for this cycle, other than the prime-pass M-strings.

    It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

28. Go to the supervisor terminal and end CPCS with the command

    ```
    STOP
    ```

Chapter 6. Sample Problem 1A: HSRR

This sample problem is identical to Sample Problem 1 (Base Functions), but shows how High-Speed Reject Re-Entry can be used to capture marginal items rejected during prime-pass capture. You can use either a physical sorter or a simulated one.

If you use a physical sorter:

- Follow the instructions at Chapter 4, “Overview of the Sample Problems” on page 4-1 to set up a physical sorter on which to capture.
- For the prime pass, turn several documents upside-down so they are rejected.
- For the HSRR pass, turn the documents that you want HSRR to read back to their correct orientation.

If you use a simulated sorter, use sorter 8 for the prime pass and sorter 9 for the HSRR pass.

Preparing to Run the Sample Problem

The test material contains:

- DKNPCPCS parameters for CPCS execution
- DKNPOLRR parameters for OLRR execution
- DKNPMRGE parameters for MRGE execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, batch, and subbatch slips
- SCI and OLRR definitions for sorting

CPCS Parameter Generation Options

CPCS requires no special options in the DKNPCPCS system profile control card for this sample problem.

DKNPOLRR Parameter Options

Edit into the DKNPOLRR member of your DKNAPPL application profile data set. Find the AUTOSTART_SCAT_WHEN_OLRRING_SYSTEM_REJECT_D-STRING_FOR_HSRR control card and set it to YES. This causes DKNSCAT to automatically concatenate any HSRR System Reject R-strings that DKNOLRR creates.

DKNPMRGE Parameter Options

Edit into the DKNPMRGE member of your DKNAPPL application profile data set. Find the USE_SCATTED_HSRR_SYSTEM_REJECT_R-STRING card and set it to YES. This causes MRGE to pass the correct start parameters when automatically starting DKNRLST.

DKNSPDEF Options

This sample problem uses member SPTYP001 in CPCS.V1R11.SAMPLIB as the sort-pattern definition.

```
*
*          TEST DECK:        DKNT01SU
*          MERGE FEED DECK:  DKNT01MU
*          ENTRY:            0010
*
*      PPH     | SCI/N.L. EDIT |   OLRR EDIT
*  ------------|---------------|---------------
*  1-00-00-00 | DKN0011Z      | DKN0011R
*  2-03-00-00 | DKN0012Z      |
*  2-05-00-00 | DKN0013Z      |
*
************************************************************
*
P1000000DKN0011Z        021           20030            2
B   001
R    0303
K0188888888028888888880588888888
*
************************************************************
*
P2030000DKN0012Z          2            0030
B   001
K0109000000021000000003110000000412000000
*
************************************************************
*
P2040000DKN0013Z          2            0030
B   001
K0102100002020213023503021000300402100008050200000 0
*
```

Figure 6-1. Sample Problem 1A

Pocket Selection for Sample Problem 1A

Figure 6-2 on page 6-2, Figure 6-3 on page 6-3, and Figure 6-4 on page 6-4 describe the contents of each of the pockets for this sample problem. In the columns labeled Routing/Transit, the letter *n* can be replaced by any digit.

Figure 6-2. Sample Problem 1A: Pocket Selection Criteria - Prime and HSRR Passes

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	11nn-nnnn	PC = AAAA33	On-us Credits
02	Kill	11nn-nnnn	PC=AAAAAA	On-us Debits
03	R/H	12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn	N/A	First Deadline Transits
04	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transits
05	Kill	11nn-nnnn	EXT PC not = AA and PC = AAAAAA	On-us Debit Returns

Figure 6-3. Sample Problem 1A: Pocket Selection Criteria - First Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	12nn-nnnn	N/A	First Deadline Transit EP=09000000
02	Kill	13nn-nnnn	N/A	First Deadline Transit EP=10000000
03	Kill	14nn-nnnn	N/A	First Deadline Transit EP=11000000
04	Kill	15nn-nnnn	N/A	First Deadline Transit EP=12000000
05	---	N/A	N/A	Not used

Figure 6-4. Sample Problem 1A: Pocket Selection Criteria - Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02010002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Data for Sample Problem 1A

The data for this sample problem consists of one set of tracers with associated data, including block, batch, and subbatch slips. This data also contains deliberate errors to show the balancing features of the system.

Two different members of CPCS.V1R11.SAMPLIB contain this data. DKNT08SU is used for the prime-pass capture, and DKNT09SU for the HSRR capture. If you use the simulator for this sample problem, you must include the fields represented by periods as shown in the records. If you use a physical document processor, do not encode the fields represented by the periods.

Reports for Sample Problem 1A

The reports that CPCS produces when you run this sample are similar to those in member SAMPLE1 of CPCS.V1R11.RPTLIB. Print out this member and compare it with your results for accuracy.

Operating Instructions for Sample Problem 1A

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminals by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal, indicating the change of a cycle's status.
5. Run the prime-pass entry.
   a. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

      ```
      MICR
      ```

      Open sorter number 8 by entering:

      ```
      O 8
      ```

      on the MICR Options screen.

   b. Enter BEGIN to display the Begin screen. Specify cycle 8, sort pattern 001, and entry number 0010.

      Note: Remember this entry number. It is referred to later in these instructions as the prime-pass entry number.

      Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.

   c. End the prime pass when you see the `Intervention Required` message on the MICR Status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.

   d. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. DKNDIST displays a message on the supervisor terminal indicating the end of the distribution task.
6. High Speed Reject Reentry (HSRR)
   a. If you use a physical sorter, manually recondition the rejects from the prime-pass system reject pocket to ensure that all documents that are to be corrected on the HSRR pass are positioned correctly, and remain in the same sequence as the prime pass.

      Place a new set of tracer slips in front of the documents from the reject pocket (including the prime-pass tracer slips).

Note: Remember the new tracer slip number. It is referred to later in these instructions as the *HSRR entry number*.

b. Start MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

```
MICR
```

Open sorter 9 by entering:

```
O 9
```

on the MICR options screen.

c. Enter BEGIN to display the Begin screen. On the Begin screen, enter:

```
8,1,0100,0010
```

This tells MICR to use cycle 8, sort pattern 001, and a HSRR entry number of 0100, with 0010 as the associated prime pass entry number.

Note: Remember the HSRR entry number. It is referred to later in these instructions.

Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.

d. End the HSRR pass when you see the `Intervention Required` message on the MICR Status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.

e. After ending the HSRR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the CLOSE command. DKNDIST displays a message on the supervisor terminal indicating the end of the distribution task.

7. The three items encoded with digit errors should remain in the reject pocket, while all other non-control documents are sorted to their correct pockets. You must use OLRR to correct the remaining rejects.

8. Use DKNOLRR to correct the rejects from the HSRR pass. Use the command:

```
OLRR 0100,01
```

Note that you use the HSRR entry number when you start OLRR.

Three documents in the example contain digit errors. Correct the errors as follows:

- R/T error: `12*122222` should be `122122222`
- Amt error: `00000*0002` should be `0000010002`
- R/T error: `1110006*7` should be `111000627`.

9. After you correct the rejects, the OLRR task ends automatically. DKNSCAT automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.

10. Start the string merge task (DKNMRGE) for the corrected entry (that has gone through SCAT) with the command:

```
MRGE 1,0100
```

Note that you use the HSRR entry number when you start MRGE.

DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the HSRR I-string (produced by the HSRR MICR run)

and the concatenated R-string (produced by SCAT). The M-string uses the prime-pass entry number.

11. DKNMRGE then automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.
12. The Exception Entry Master List report, produced by DKNPLST, identifies one subbatch as being `OUT OF BALANCE`. The subbatch that is out-of-balance is the subbatch slip with the serial number of 51. Because this subbatch is out-of-balance, the batch and block are also out-of-balance.

    This out-of-balance condition is caused by the intentional entry of an incorrect amount ($100.02) during OLRR for this subbatch. ($.02 is the correct amount.)
13. You can run the subsequent passes in one of two ways:

    a. Place the HSRR pass rehandle pocket 03 items behind the prime pass rehandle pocket 03 items, and run them together as a single subsequent pass. Then, place the HSRR pass rehandle pocket 04 items behind the prime pass rehandle pocket 04 items, and run them together as a single subsequent pass.

    b. Run each of the four rehandle pockets (two from the prime pass, two from the HSRR pass) as their own, separate subsequent pass entries.

    To run each rehandle pocket as a separate subsequent pass, do the following:

    a. Open sorter 8 and proceed to the Begin screen.

    b. On the Begin screen, enter:

       ,,`0010-010`

       That is, the tracer and slip number of the first tracer to sort to prime pass rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge the data is correct by pressing **ENTER**.

    d. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

    e. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

    f. On the Begin screen, enter:

       ,,`0010-013`

       That is, the tracer and slip number of the first tracer to sort to prime pass rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

    g. Subsequent-pass data appears on the MICR Begin screen. Acknowledge the data is correct by pressing **ENTER**.

    h. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

i. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

j. On the Begin screen, enter:

```
,,0100-010
```

That is, the tracer and slip number of the first tracer to sort to HSRR pass rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

k. Subsequent-pass data appears on the MICR Begin screen. Acknowledge the data is correct by pressing **ENTER**.

l. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

m. Press ENTER to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.

n. On the Begin screen, enter:

```
,,0100-013
```

That is, the tracer and slip number of the first tracer to sort to HSRR pass rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

o. Subsequent-pass data appears on the MICR Begin screen. Acknowledge the data is correct by pressing **ENTER**.

p. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

q. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

r. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.

14. Start the Kill List task (DKNKILL) for the first deadline subsequent-pass items with the command:

```
KILL 8
```

Where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT001** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT001 contains the endpoints for each first deadline subsequent-pass kill pocket.

15. Start the Kill List task (DKNKILL) for the second deadline subsequent-pass items with the command:

```
KILL 8
```

Where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT002 contains the endpoints for each second deadline subsequent-pass kill pocket.

16. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    Where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT003** when prompted for the table name. This table contains all the endpoints from all deadlines, both first and second. When DKNCLSM prompts you to respond to whether this is a duplicate letter, press **ENTER** to select NO.

17. Run the Master Create (DKNMCRE) task for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

18. Run the Input Create (DKNICRE) task for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the specified cycle to the input-create file.

19. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

20. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system. After you run this task, the only string left on the mass data set is the M-string.

21. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures that there are no active strings present for this cycle other than the M-string.

    It deletes this M-string and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

22. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 7. Sample Problem 1B: HSRR AND EMRG

These instructions demonstrate how to verify the installation of DKNEMRG was done correctly. DKNEMRG is designed to work with the ESM feature of CPCS. It can be run manually, but it does not delete any strings or automatically start any tasks, such as DKNRLST or DKNPLST.

Important!

Do not run DKNEMRG in a production non-ESM environment; use DKNMRGE instead. DKNMRGE contains editing and auditing controls that are used by follow-on tasks.

This sample problem is a follow-on to "Sample Problem 1A: HSRR". However, it is run with DKNEMRG rather than DKNMRGE. To run this sample problem, use the same preparation methods and select the same options that apply to Sample Problem 1A.

After running this sample problem manually, generate an ESM workflow for this problem and run it again using the Enhanced System Manager. A workflow that automatically starts DIST, SCAT, EMRG, PLST, RLST, and STGD at the appropriate times allows an easier running of the sample problem. Please refer to the *CPCS Enhanced System Manager User's Guide* for more information.

Operating Instructions for Sample Problem 1B

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminals by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the READY prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Run the prime-pass entry.
   a. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

      ```
      MICR
      ```

      Open sorter 8 by entering:

      ```
      O 8
      ```

      on the MICR Options screen.

b. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 001, and entry number 0010.

   Note: Remember this entry number. It is referred to later in these instructions as the prime-pass entry number.

   Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.

c. End the prime pass when you see the `Intervention Required` message on the MICR Status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.

d. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. DKNDIST displays a message on the supervisor terminal indicating the end of the distribution task.

6. High Speed Reject Reentry (HSRR)

   a. If you use a physical sorter, manually recondition the rejects from the prime-pass system reject pocket to ensure that all documents that are to be corrected on the HSRR pass are positioned correctly, and remain in the same sequence as the prime pass.

      Place a new set of tracer slips in front of the documents from the reject pocket (including the prime-pass tracer slips).

      Note: Remember the new tracer group number. It is referred to later in these instructions as the HSRR entry number.

   b. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

      ```
      MICR
      ```

      Open sorter 9 by entering:

      ```
      O 9
      ```

      on the MICR options screen.

   c. Enter **BEGIN** to display the Begin screen. On the Begin screen, enter:

      ```
      8,001,0100,0010
      ```

      This tells MICR to use cycle 8, sort pattern 001, and a HSRR entry number of 0100, with 0010 as the associated prime-pass entry number.

      Note: Remember the HSRR entry number. It is referred to later in these instructions.

      Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.

   d. End the HSRR pass when you see the `Intervention Required` message on the MICR Status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.

   e. After ending the HSRR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. DKNDIST displays a message on the supervisor terminal indicating the end of the distribution task.

7. The three items encoded with digit errors should remain in the reject pocket, while all other non-control documents are sorted to their correct pockets. You must use OLRR to correct the remaining rejects.
8. Use DKNOLRR to correct the rejects from the HSRR pass. The command to use is:

   ```
   OLRR 0100,01
   ```

   Note that you use the HSRR entry number when you start OLRR.

   Three documents in the example contain digit errors. Correct the errors as follows:

   - R/T error: "12*122222" should be "122122222"
   - Amt error: "00000*0002" should be "0000010002"
   - R/T error: "1110006*7" should be "111000627"
9. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.
10. Start the string merge task (DKNEMRG) for the corrected entry (that has gone through SCAT) with the command:

    ```
    EMRG 0100-1-R -00-00-00-R-000
    ```

    Note that you use the HSRR entry number when you start EMRG.

    DKNEMRG first merges the HSRR R-string and the HSRR I-string to create the HSRR 50-M string (0100-1-00-00-00-50-M-000). Next, DKNEMRG does a codeline data-match with the Prime-Pass System Reject D-string and the HSRR I-string to produce the HSRR 52-M string (0100-1-00-00-00-52-M-000). Finally, using the HSRR 52-M string as a guide, DKNEMRG merges the HSRR 50-M string and the Prime-Pass I-string to create the Prime-Pass 00-M string (0010-1-00-00-00-00-M-000). All these steps take place automatically. Messages written to the supervisor terminal track DKNEMRG's progress.
11. DKNEMRG does not automatically start DKNPLST or DKNRLST, nor does it delete any strings. These functions are normally taken care of by the ESM workflows. Because we have been manually running DKNEMRG, we should at this point:

    a. Enter:

       ```
       RLST
       ```

       to start DKNRLST. When RLST starts, enter:

       ```
       1,0100
       ```

       to print the System Reject Pocket reject list.

    b. Enter:

       ```
       PLST 0010
       ```

       to start DKNPLST. When PLST starts, select option 3 to get the Exception Entry Master List.

    c. Enter:

```
DELE 0010-1-00-00-00-00-I-000
DELE 0010-1-R -00-00-00-D-000
DELE 0100-1-00-00-00-00-I-000
DELE 0100-1-R -00-00-00-D-000
DELE 0100-1-00-00-00-50-M-000
DELE 0100-1-00-00-00-52-M-000
```

to delete the I-string, System Reject D-string, and work strings.

12. The Exception Entry Master List report, produced by DKNPLST, identifies one subbatch as being `OUT OF BALANCE`. The subbatch that is out-of-balance is the subbatch slip with the serial number of 51. Because this subbatch is out of balance, the batch and block are also out of balance.

    This out-of-balance condition is caused by the intentional entry of an incorrect amount ($100.02) during OLRR for this subbatch. ($.02 is the correct amount.)

13. You can run the subsequent passes in one of two ways:

    a. Place the HSRR pass rehandle pocket 03 items behind the prime-pass rehandle pocket 03 items, and run them together as a single subsequent pass. Then, place the HSRR pass rehandle pocket 04 items behind the prime pass rehandle pocket 04 items, and run them together as a single subsequent pass.

    b. Run each of the four rehandle pockets (two from the prime pass, two from the HSRR pass) as their own, separate, subsequent pass entries.

    To run each rehandle pocket as a separate subsequent pass, do the following:

    a. Open sorter 8 and proceed to the Begin screen.

    b. On the Begin screen, enter:

       ,,`0010-010`

       That is, the tracer and slip number of the first tracer to sort to prime-pass rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

    d. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

    e. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

    f. On the Begin screen, enter:

       ,,`0010-013`

       That is, the tracer and slip number of the first tracer to sort to prime-pass rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

    g. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

    h. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

i. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

j. On the Begin screen, enter:

```
,,0100-010
```

That is, the tracer and slip number of the first tracer to sort to HSRR pass rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

k. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

l. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

m. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

n. On the Begin screen, enter:

```
,,0100-013
```

That is, the tracer and slip number of the first tracer to sort to HSRR pass rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

o. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

p. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

q. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

r. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.

14. Start the Kill List task (DKNKILL) for the first deadline subsequent-pass items with the command:

```
KILL 8
```

where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**, enter **EPT001** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT001 contains the endpoints for each first deadline subsequent-pass kill pocket.

15. Start the Kill List task (DKNKILL) for the second deadline subsequent-pass items with the command:

```
KILL 8
```

where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**, enter **EPT002** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT002 contains the endpoints for each second deadline subsequent-pass kill pocket.

16. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT003** when prompted for the table name. This table contains all the endpoints from all deadlines, both first and second. When DKNCLSM prompts you to respond to whether this is a duplicate letter, press **ENTER** to select NO.

17. Run the Master Create (DKNMCRE) task for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

18. Run the Input Create (DKNICRE) task for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the specified cycle to the input-create file.

19. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

20. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system. After you run this task, the only string left on the mass data set is the M-string.

21. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures that there are no active strings present for this cycle other than the M-string.

    It deletes this M-string and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

22. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 8. Sample Problem 2: Enhanced Prime

This sample problem shows the capabilities of Enhanced Prime capture. In Enhanced Prime, whenever MICR encounters a new set of tracers, instead of starting a new tracer group within the current entry, it starts a new entry. The old entry ends and becomes immediately available for Distribution, Kill, OLRR, and all other normal post-capture processing. Meanwhile, the sorter continues to process further entries. As many entries as you like can be captured in a single sorter run.

This sample problem also demonstrates how the Kill List task (DKNKILL) can be made to auto-start after each run of DKNDIST.

Preparing to Run the Sample Problem

The test material contains:

- DKNPCPCS parameters for CPCS execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, and batch slips. There are actually three sets of tracers in the sample data. Each set represents the start of a new entry.
- SCI and OLRR definitions for sorting

CPCS Parameter Generation Options

The member DKNPCPCS in CPCS.V1R11.SAMPLIB contains the system parameters. Change the KILLAUT parameter within this member to the following value:

```
KILLAUT=1
```

This causes DKNDIST to automatically start DKNKILL for each I-string it distributes.

CPCS requires no other special options in the DKNPCPCS System Profile control care for this sample problem.

DKNSPDEF Options

This sample problem uses member SPTYP002 in CPCS.V1R11.SAMPLIB as the sort-pattern definition. The O-record contains all the options for Enhanced Prime capture.

Position	Option
6	Prime Pass Type <blank> = Conventional Capture E = Enhanced Prime Capture
7	Enhanced Prime Sorter Initialization Option <blank> = Sorter not re-initialized at the start of each new entry Y = Sorter is re-initialized at the start of each new entry The principal value of re-initializing the sorter is to reset the pocket counters. Suppose the kill bundle count (P record positions 40 - 43) was set at 0020. If fifteen items had sorted to a kill pocket at the time an entry ended, then normally during the next entry the pocket would be killed (a divider slip selected to it) after only five items. By re-initializing

the sorter at the start of each entry, you guarantee the first bundle of each kill pocket is a full twenty items long.

Note: Position 7 must be Y if you are using both Enhanced Prime and Merge-Before-Main (O-record position 30 is B).

13 Tracers in kill pockets

Y = Tracers placed in each kill pocket at the start of each entry
N = No tracers in kill pockets

This option can be used to provide a visual separation between entries in each kill pocket. Selecting this option requires you to re-generate MICR with a larger MAXRP parameter to account for the additional tracers; also, it requires you to reallocate a larger tracer group set (DKNTG). See the *CPCS Customization Guide*, Chapter 1 “System Programming and Operations,” section “External Storage Requirements,” heading “Direct Access Storage Requirements,” subheading “DKNTG (Pass-to-Pass Control Data Set).” In that same book, see also in Chapter 2 “Installation and Generation Procedures,” section “MICR Task Generation,” heading “CPCSOPTN Macro Parameters,” subheading “MAXRP,” for more information.

15 Pause Option

Y = MICR pauses between entries
N = MICR does not pause between entries

This option allows you to make the sorter stop at the end of each entry so the operator can take some type of action. The operator uses the CONT command to signal when he or she is ready to have the sorter resume capture again.

If the Pause option is not selected, the sorter automatically begins capturing the next entry, without stopping or requiring operator intervention.

30 Divider Spray Type

<blank> = Place dividers after kill bundles
A = Place dividers after kill bundles
B = Place dividers before kill bundles (Merge-Before-Main)

If you select the Merge-Before-Main option, then all kill bundles, including the first bundle in each kill pocket for each entry, is preceded by a divider slip. Normally dividers are placed only between adjacent kill bundles.

Note: Sort pattern SPTYP002:

- Selects for Enhanced Prime
- Re-initializes the sorter at the start of each entry
- Does not put tracers in kill pockets
- Does not pause between entries
- Does not use Merge-Before-Main

If you want to test alternative options, you must change the sort pattern definition.

```
*
*          TEST DECK:        DKNT02SU
*          MERGE FEED DECK:  DKNT01MU
*          ENTRY:            0020
*
*      PPH     | SCI/N.L. EDIT |   OLRR EDIT
*  ------------|---------------|--------------
*  1-00-00-00 | DKN0011Z      | DKN0011R
*  2-03-00-00 | DKN0012Z      |
*  2-05-00-00 | DKN0013Z      |
*
*********************************************************
*
P1000000DKN0011Z        021           20008            2
B   001
O    EY
R    0303
K0188888888028888888880588888888
*
*********************************************************
*
P2030000DKN0012Z          2            0030
B   001
K0109000000021000000003110000000412000000
*
*********************************************************
*
P2040000DKN0013Z          2            0030
B   001
K0102100002020213023503021000300402100008050200000 0
*
```

Figure 8-1. Sample Problem 2

Pocket Selection for Sample Problem 2

Figure 8-2, Figure 8-3 on page 8-4, and Figure 8-4 on page 8-4 describe the contents of each of the pockets for this sample problem. In the columns labeled *Routing/Transit*, the letter *n* can be replaced by any digit.

Figure 8-2 (Page 1 of 2). Sample Problem 2: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Field	Description
RJ	Reject	N/A	N/A	System reject pocket
01	Kill	11nn-nnnn	PC=AAAA33	On-us Credits
02	Kill	11nn-nnnn	PC=AAAAAA	On-us Debits
03	R/H	12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn	N/A	First Deadline Transits

Figure 8-2 (Page 2 of 2). Sample Problem 2: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Field	Description
04	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transits
05	Kill	11nn-nnnn	EXT PC not = AA and PC=AAAAAA	On-us Debit Returns

Figure 8-3. Sample Problem 2: Pocket Selection Criteria - First Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	12nn-nnnn	N/A	First Deadline Transit EP=09000000
02	Kill	13nn-nnnn	N/A	First Deadline Transit EP=10000000
03	Kill	14nn-nnnn	N/A	First Deadline Transit EP=11000000
04	Kill	15nn-nnnn	N/A	First Deadline Transit EP=12000000
05	---	N/A	N/A	Not Used

Figure 8-4 (Page 1 of 2). Sample Problem 2: Pocket Selection Criteria - Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02010002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235

Figure 8-4 (Page 2 of 2). Sample Problem 2: Pocket Selection Criteria - Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
03	Kill	20nn-nnnn	N/A	Second deadline transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Data for Sample Problem 2

The data for this sample problem consists of three sets of tracers, each with associated data, including block and batch slips. This data also contains deliberate errors to show the balancing features of the system.

The member DKNT02SU in CPCS.V1R11.SAMPLIB contains this input. If you use the simulator for this sample problem, you must include the fields represented by periods as shown in the records. If you use a physical document processor, do not encode the fields represented by the periods.

Reports for Sample Problem 2

The reports that CPCS produces when you run this sample problem are in CPCS.V1R11.RPTLIB. Print out member SAMPLE2 from this library and compare it with your results for accuracy.

Operating Instructions for Sample Problem 2

1. Start CPCS by submitting the job in member DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminals by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the display instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering:

   ```
   MICR
   ```

   Open sorter 2 by entering:

   ```
   O 2
   ```

   on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 2.
6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 002, and entry number 0020. Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.
7. If you have selected the Pause Option in the DKNSPDEF O-record, DKNMICR pauses and prompts you at the end of each entry. Type **CONT** and press **ENTER** to continue capture of the next entry. (Other alternatives include entering **E** to end the entry and **CA** to cancel it. Both these alternatives abort capture of further entries.) If you did not select the pause option, DKNMICR automatically ends each entry and begins the next, continuing on until it has read all items.
8. When the `Intervention Required` message appears on the MICR screen, three entries will have been captured: 0020, 0021, and 0022. End the prime-pass run by entering **E** at the prompt. Enter another **E** to verify the End command.
9. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. The distribution task (DKNDIST) displays messages on the supervisor terminal, indicating that it has been run for each of entries 0020, 0021, and 0022.
10. Use DKNOLRR to correct the rejects from prime-pass entry 0020. The command to use is:

    ```
    OLRR 0020,01
    ```

    Three documents in the example contain digit errors. Correct the errors as follows:

    a. R/T error: `12*122222` should be `122122222`
    b. Amt error: `00000*0002` should be `0000010002`

c. R/T error: `1110006*7` should be `111000627`.

11. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.
12. Use DKNOLRR to correct the rejects from prime-pass entry 0021. The command to use is:

    ```
    OLRR 0021,01
    ```

    Two documents in the example contain digit errors. Correct the errors as follows:

    a. R/T error: `111*00627` should be `111000627`
    b. Amt error: `00000*0002` should be `0000000002`

13. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.
14. Use DKNOLRR to correct the rejects from prime-pass entry 0022. The command to use is:

    ```
    OLRR 0022,01
    ```

    Four documents in the example contain digit errors. Correct the errors as follows:

    a. R/T error: `12*122222` should be `122122222`
    b. Amt error: `00111*0000` should be `0011110000`
    c. R/T error: `1110006*7` should be `111000627`
    d. Amt error: `00002222*0` should be `0000222220`

15. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.
16. Start the string merge task (DKNMRGE) for entry 0020 with the command:

    ```
    MRGE 1,0020
    ```

    DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the concatenated R-string produced by SCAT.

17. DKNMRGE automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.
18. The Exception Entry Master List report, produced by DKNPLST, identifies one block as being “OUT OF BALANCE”. The block that is out of balance is the block slip with the serial number of 49. This out-of-balance condition is caused by the intentional entry of an incorrect amount during OLRR for this block.
19. Start the string merge task (DKNMRGE) for entry 0021 with the command:

    ```
    MRGE 1,0021
    ```

    DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the concatenated R-string produced by SCAT.

20. DKNMRGE automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE

displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.

21. The Exception Entry Master List report, produced by DKNPLST, shows that entry 0021 is in balance.
22. Start the string merge task (DKNMRGE) for entry 0022 with the command:

    ```
    MRGE 1,0022
    ```

    DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the concatenated R-string produced by SCAT.
23. DKNMRGE automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.
24. The Exception Entry Master List report, produced by DKNPLST, identifies one block as being "OUT OF BALANCE." The block that is out of balance is the block slip with the serial number of 249. This out-of-balance condition is caused by an erroneous amount encoded on the item with serial number 260.
25. Start the MICR subsequent pass for the documents sorted to the first rehandle pocket (pocket 03) on prime pass for entry 0020.
    a. Open sorter 2 and proceed to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0020-010
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
26. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
27. Start the MICR subsequent pass for the documents sorted to the first rehandle pocket (pocket 03) on prime pass for entry 0021.
    a. Press **ENTER** to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0021-010
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
28. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

29. Start the MICR subsequent pass for the documents sorted to the first rehandle pocket (pocket 03) on prime pass for entry 0022.
    a. Press **ENTER** to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.
    b. On the Begin screen, enter:

       ,,0022-010

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
30. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
31. Start the MICR subsequent pass for the documents sorted to the second rehandle pocket (pocket 04) on prime pass for entry 0020.
    a. Press **ENTER** to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.
    b. On the Begin screen, enter:

       ,,0020-013

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
32. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
33. Start the MICR subsequent pass for the documents sorted to the second rehandle pocket (pocket 04) on prime pass for entry 0021.
    a. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.
    b. On the Begin screen, enter:

       ,,0021-013

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
34. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
35. Start the MICR subsequent pass for the documents sorted to the second rehandle pocket (pocket 04) on prime pass for entry 0022.

a. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

b. On the Begin screen, enter:

```
,,0022-013
```

That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

36. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

37. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

38. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.

39. Because you specified KILLAUT=1 in the system profile DKNPCPCS, each DKNDIST also automatically started DKNKILL. DKNKILL produces all kill lists for whatever endpoints, first deadline or second, had been distributed by DKNDIST.

40. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    where 8 is the current cycle number.

    Select the Endpoint Table option and use table EPT003 when prompted for the table name. This table contains all the endpoints from all deadlines, both first and second. When DKNCLSM prompts you to respond to whether this is a duplicate letter, press **ENTER** to select NO.

41. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

42. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

43. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

44. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system. After you run this task, the only strings left on the mass data set are the M-strings for the three prime pass entries.

45. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures that there are no active strings present for this cycle, other than the prime-pass M-strings.

    It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

46. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 9. Sample Problem 3: Enhanced Reject Processing

This sample problem shows the capabilities of alternate reject pockets and enhanced reject processing. The instructions provided tell you how to:

- Run a prime pass “breakout run” that distributes to multiple rehandle pockets.
- Prepare kill bundles and cash letters for the first deadline.
- Process rehandle pockets. DKNMICR, DKNKILL, and DKNCLSM process the data to produce the kill bundles and cash letters for the second deadline.
- Repair and process transit rejects. This step also creates kill bundles and cash letters to satisfy the first and second deadlines.
- Repair the bad data from the on-us reject pocket.
- Repair the consolidated rejects. This step corrects errors in bad records sent to pockets that were originally good kill pockets.
- Repair the errors from the system reject pocket.
- Merge all the corrected data using DKNMRGE.
- Distribute the merged string using DKNMDIS. The new D-strings replace the old and incorporate all the repairs you have made. Revised kill lists and cash letters can be prepared from the corrected D-strings.
- Perform final processing. This includes backing up the data, generating reports, and deleting data that is no longer useful.

Preparing to Run the Sample Problem

The test material contains:

- DKNPCPCS parameters for CPCS execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, batch, and subbatch slips
- SCI and OLRR definitions for sorting

CPCS Parameter Generation Options

CPCS requires no special options in the DKNPCPCS System Profile control card for this sample problem.

DKNSPDEF Options

This sample problem uses member SPTYP003 in CPCS.V1R11.SAMPLIB as the sort-pattern definition. The options necessary to specify Enhanced Reject Processing are contained on the O- and J-records.

O-Record

Position	Option
29	Enhanced reject processing <blank> = Not active Y = Active. The J-record defines what type of ERP pockets are in use.

J-Record

Position	Option
5 - 39	Each position corresponds to a pocket 1 - 35. <blank> = Standard kill or rehandle pocket R = Alternate reject pocket. Items containing rejects can be sorted to here. The pocket may be either a kill or a rehandle pocket. E = Eligible-to-receive pocket. Items containing rejects can be sorted to here. A special DKNDIST option combines all eligible pockets into a single Consolidated Rejects D-string.

In SPTYP003, prime pass pockets 2 and 3 are defined as Alternate Reject pockets (and are also rehandle pockets); pockets 4 and 5 are Eligible-to-Receive.

SPTYP003's prime pass is also defined as running on an image-capture sorter. This is for the benefit of those institutions with a Key Entry application which wish to use it to correct the Consolidated Rejects D-string. In this sample problem, however, we use OLRR to substitute for Key Entry.

Two of the passes in SPTYP003 are defined as running on a power-encode sorter. The O-record, in columns 16 - 23, names a special DKNMIPI user exit, DKNMXS32, that loads corrections from OLRR into an otherwise unused field, field 9 (the O-record also names that field. See columns 24 - 25). Since field 9 is normally unused, MICR knows nothing about it; consequently, a FLD09 card exists to tell MICR how large the field is. A special Run Profile (SAMPXF01) copies data from the field into the process buffers, from which an H-record then instructs the sorter to power encode all seven fields.

The effect of all this is that whatever OLRR corrections are made to the Alternate Reject rehandle items get power-encoded back onto the items when they run through their subsequent passes.

```
*
*          TEST DECK:        DKNT03SU
*          MERGE FEED DECK:  DKNT01MU
*          ENTRY:            0030
*
*      PPH     | SCI/N.L. EDIT |   OLRR EDIT
*  ------------|---------------|--------------
*  1-00-00-00 | DKN0031Z      | DKN0031R
*  2-01-00-00 | DKN0032Z      |
*  2-02-00-00 | DKN0032Z      |
*  2-03-00-00 | DKN0033Z      |
*  3-01-04-00 | DKN0034Z      |
*  3-03-03-00 | DKN0034Z      |
*
**********************************************************
*
P1000000DKN0031Z        031           10030XF         M11
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 3 - ERP -         PRIME PASS
I   P  BW       BW
J    RREE
O                           Y
R060306
K04090000000511000000
*
**********************************************************
*
P2010000DKN0032Z        011            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 3 - ERP -         ONUS
R      03
K018888888802888888880388888880588888888
*
**********************************************************
*
P2020000DKN0032Z          1            0030XF
B   001
RP      SAMPXF01                        PE
BMSG  SAMPLE PROBLEM 3 - ERP -         ONUS REPAIRS
H        YY  P Y       1234567
O              DKNMXS3209
FLD09     045
K018888888802888888880388888880588888888
*
```

Figure 9-1 (Part 1 of 2). Sample Problem 3

```
*********************************************************
*
P2030000DKN0033Z        012            0030XF
B   001
RP      SAMPXF01                        PE
BMSG  SAMPLE PROBLEM 3 (US) - ERP -    TRANSIT REPAIRS
H         YY  P Y       1234567
O              DKNMXS3209
FLD09     045
R    03
K040900000000511000000
*
*********************************************************
*
P3010400DKN0034Z          2            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 3 - ERP -         2ND TRANSIT DEADLINE
K0102100002020213023503021000300402100008050200000 0
*
*********************************************************
*
P3030300DKN0034Z          2            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 3 - ERP -         2ND TRANSIT DEADLINE
K0102100002020213023503021000300402100008050200000 0
*
```

Figure 9-1 (Part 2 of 2). Sample Problem 3

Pocket Selection for Sample Problem 3

Tables 9-2 through 9-7 describe the contents of each of the pockets for this sample problem. In the column labeled Routing/Transit, the letter *n* can be replaced by any digit.

Figure 9-2. Sample Problem 3: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 11nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	All Fields Good	On-us and Second Deadline Transits
02	R/H	11nn-nnnn	Any Bad Field	On-us Rejects
03	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	Bad Amounts	Transit Rejects
04	Kill	12nn-nnnn 13nn-nnnn	All Fields Good; or, Non-Amount Field Error	First Deadline Transit EP=09000000
05	Kill	14nn-nnnn 15nn-nnnn	All Fields Good; or, Non-amount Field Error	First Deadline Transit EP=11000000

Figure 9-3. Sample Problem 2: Pocket Selection Criteria - On-us Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	11nn-nnnn	Amount > $1M	On-us High Dollar Debits
02	Kill	11nn-nnnn	PC=AAAA33	On-us Credits
03	Kill	11nn-nnnn	PC=AAAAAA	On-us Debits
04	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transits
05	Kill	11nn-nnnn	EXT PC not = AA and PC=AAAAAA	On-us Debit Returns

Figure 9-4. Sample Problem 3: Pocket Selection Criteria - On-us Repairs Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	11nn-nnnn	Amount > $1M	On-us High Dollar Debits
02	Kill	11nn-nnnn	PC=AAAA33	On-us Credits
03	Kill	11nn-nnnn	PC=AAAAAA	On-us Debit Returns
04	---	N/A	N/A	Not Used
05	Kill	11nn-nnnn	EXT PC not = AA and PC=AAAAAA	On-us Debit Returns

Figure 9-5. Sample Problem 3: Pocket Selection Criteria - Transit Repairs Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	---	N/A	N/A	Not Used
02	---	N/A	N/A	Not Used
03	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit
04	Kill	12nn-nnnn 13nn-nnnn	N/A	First Deadline Transit EP=09000000
05	Kill	14nn-nnnn 15nn-nnnn	N/A	First Deadline Transit EP=11000000

Figure 9-6. Sample Problem 3: Pocket Selection Criteria - Second Transit Deadline Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02100002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Figure 9-7. Sample Problem 3: Pocket Selection Criteria - Repaired Second Transit Deadline

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=021000002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Data for Sample Problem 3

The data for this sample problem consists of one set of tracers with associated data,including block and batch slips. This data also contains deliberate errors. The errors are used to demonstrate how alternate reject pockets and eligible-to-receive kill pockets can enhance the repairs process.

The member DKNT03SU in CPCS.V1R11.SAMPLIB contains this input. If you use the simulator for this sample problem, you must include the fields represented by periods as shown in the records. If you use a physical document processor, do not encode the fields represented by the periods.

Reports for Sample Problem 3

The reports that CPCS produces when you run this sample problem are in CPCS.V1R11.RPTLIB. Print out member SAMPLE3 from this library and compare it with your results for accuracy.

Operating Instructions for Sample Problem 3

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminal by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

   ```
   MICR
   ```

   Open sorter 3 by entering:

   ```
   O 3
   ```

   on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 3.
6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 003, and entry number 0030. Press **ENTER**.
7. If this is the first time sorter 3 has run since the most recent cold start, you will be prompted for a microfilm sequence number. Enter any six-digit value. Press **ENTER** again to confirm the Begin screen data and start the actual capture.
8. End the prime-pass run when you see the `Intervention Required` message on the MICR Status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.
9. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. The distribution task (DKNDIST) displays a message on the supervisor terminal indicating the end of the distribution task.
10. Start the MICR subsequent pass for the On-us and Second Deadline Transit items sorted to the first rehandle pocket (pocket 1) on prime pass for this entry.
    a. Open sorter 3 and proceed to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0030-010
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 01. The two commas ensure the data winds up in the ".ENTRY" field.

c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

11. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
12. Start the MICR subsequent pass for the Second Deadline Transit items that sorted to rehandle pocket 04 during step 10 on page 9-9.

    a. Press **ENTER** to return to the MICR Options screen, then type **B** and press **ENTER** to get back to the Begin screen.

    b. On the Begin screen, enter:

    `,,0030-011`

    That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04 in step 10 on page 9-9. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

13. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
14. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.
15. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.

16. Start the Kill List task (DKNKILL) for the second deadline items with the command:

    `KILL 8`

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and select option C on the Cash Letter Override screen. This causes DKNKILL to print Cash Letters as well as Kill Lists.

17. Use DKNOLRR to correct the rejects in the First and Second Transit Deadline items sorted to the third rehandle pocket (pocket 03) on prime pass for this entry. The command to use is:

    `OLRR`

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    `0030-1-Z3-00-00-00-D-000`

    The "Z" in the pocket number identifies this string as containing rejects. Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.

18. In this sample problem, incorrect items appear with asterisks (*) in place of unreadable digits. Replace all the asterisks with 1's.
19. When OLRR completes, it creates string 0030-1-03-00-00-01-R-000. Note that pocket “Z3” has been renamed “03”, to indicate that it no longer contains rejects. The “01” in the fourth pocket position names the OLRR operator ID.
20. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as, among other things, it updates the tracer group data set.) The command to use is:

    ```
    SCAT 0030-1-03-00-00-01-R-000
    ```

    SCAT creates a new string 0030-1-03-00-00-00-R-000 for use by DKNMRGE, and by the sorter’s subsequent pass codeline data match routines.
21. Start the MICR subsequent pass for the items you have just corrected.
    a. Open sorter 4. Sorter 4 is defined as a power-encode sorter. If you are running this sample problem on a live power-encode sorter, it prints a fresh MICR line on each item it reads. These replacement MICR lines include all the corrections you keyed in step 18.
    b. Go to the Begin screen and enter:

       ,,0030-019

       That is, the tracer and slip number of the first tracer to sort to prime-pass rehandle pocket 03. The two commas ensure the data winds up in the “.ENTRY” field.
    c. At this point, the following warning message appears:

       ```
       WARNING=MICMSPFL 1002 FLD09 BYTE LEN 045 IS
       GREATER THAN MDX LEN 000
       ```

       What has happened is that the sort pattern uses a special MIPI exit--DKNMXS32--to download to the sorter, power-encode information in an otherwise unused field (field 9). MICR is simply noting that field 9 appears not to be empty. You can safely ignore the message.
    d. Press **ENTER** to accept the data on the MICR Begin screen and start the actual capture.
22. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
23. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.
24. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to product a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.
25. Start the Kill List task (DKNKILL) for the first deadline items with the command:

    ```
    KILL 8
    ```

where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT031** when prompted for the table name, and select option C on the Cash Letter Override screen. This causes DKNKILL to print Cash Letters as well as Kill lists.

26. Start the MICR subsequent pass for the Second Deadline Transit items that sorted to rehandle pocket 03 during step 21 on page 9-11.

    a. Open sorter 3 and proceed to the Begin screen.

    b. On the Begin screen, enter:

       ,,0030-020

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03 in step 21 on page 9-11. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

27. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

28. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

29. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

30. Start the Kill List task (DKNKILL) for the second deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and select option C on the Cash Letter Override screen. This causes DKNKILL to print Cash Letters as well as Kill lists.

31. Use DKNOLRR to correct the rejects in the On-us items sorted to the second rehandle pocket (pocket 02) on prime pass for this entry. The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0030-1-Z2-00-00-00-D-000
    ```

    The "Z" in the pocket number identifies this string as containing rejects. Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.

32. In this sample problem, incorrect items appear with asterisks (*) in place of unreadable digits. Replace all the asterisks with 1's.

33. When OLRR completes, it creates string 0030-1-02-00-00-01-R-000. Note that pocket “Z2” has been renamed “02”, to indicate that it no longer contains rejects. The “01” in the fourth pocket position names the OLRR operator ID.

34. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run; however, as among other things, it updates the tracer group data set.) The command to use is:

    ```
    SCAT 0030-1-02-00-00-01-R-000
    ```

    SCAT creates a new string 0030-1-02-00-00-00-R-000 for use by DKNMRGE, and by the sorter’s subsequent pass codeline data match routines.

35. Start the MICR subsequent pass for the items you have just corrected.

    a. Open sorter 4. Sorter 4 is defined as a power-encode sorter. If you are running this sample problem on a live power-encode sorter, then it prints a fresh MICR line on each item it reads. These replacement MICR lines include all the corrections you keyed in step 32 on page 9-12.

    b. Go to the Begin screen and enter:

       ```
       ,,0030-016
       ```

       That is, the tracer and slip number of the first tracer to sort to prime-pass rehandle pocket 02. The two commas ensure the data winds up in the “.ENTRY” field.

    c. At this point, the following warning message appears:

       ```
       WARNING=MICMSPFL 1002 FLD09 BYTE LEN 045 IS
       GREATER THAN MDX LEN 000
       ```

       What has happened is that the sort pattern uses a special MIPI exit--DKNMXS32--to download to the sorter power-encode information in an otherwise unused field (field 9). MICR is simply noting that field 9 appears not to be empty. You can safely ignore the message.

    d. Press **ENTER** to accept the data on the MICR Begin screen and start the actual capture.

36. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

37. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

38. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

39. Run DKNDIST to distribute the Consolidated Reject pocket. To do this, enter the command:

    ```
    DIST
    ```

    and press **ENTER** to display the initial screen for DNDIST. Next, enter the I-string name:

```
0030-1-00-00-00-00-I-000
```

The next menu from DIST asks for the type of distribution; enter:

```
RC
```

DKNDIST ends and creates a Consolidated Reject D-string, 0030-1-RC-00-00-00-D-000, that contains items from all Eligible-to-Receive pockets (in this case, prime pass pockets 04 and 05).

40. Because a Consolidated Reject D-string contains items from several different pockets, it is normally infeasible to run OLRR against it. The preferred method would be to use some sort of Key Entry system (this is why sorter 3 is defined as having an Image camera).

    However, in this sample problem we know exactly which items sorted to the Eligible-to-Receive pockets, so we can use OLRR to substitute for Key Entry.

    The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0030-1-RC-00-00-00-D-000
    ```

    Leave the operator ID as “01”, and the BYPASS and RESTART options as “N”. Press **ENTER**.

41. In this sample problem, incorrect items appear with asterisks (*) in place of unreadable digits. Replace all the asterisks with 1’s.

42. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as DKNMRGE expects concatenated R-strings.) The command to use is:

    ```
    SCAT 0030-1-RC-00-00-01-R-000
    ```

    SCAT creates a new string 0030-1-RC-00-00-00-R-000 for use by DKNMRGE.

43. Use DKNOLRR to correct the rejects from the prime pass. The command to use is:

    ```
    OLRR 0030,01
    ```

    One document in this example contains a digit error. Correct the error as follows:

    - R/T error: `12*122222` should be `122122222`

44. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.

45. Start the string merge task (DKNMRGE) for the corrected and concatenated prime-pass System Reject pocket. The command to use is:

    ```
    MRGE 2,0030,000
    ```

    DKNMRGE combines the prime pass I-string with the Concatenated System Reject R-string created in step 44. Because there are still rejects left to merge, the result is an interim M-string (0030-1-00-00-00-01-M-000).

46. DKNMRGE then automatically starts the formatted R-string list task (DKNRLST) for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNRLST displays messages on the supervisor terminal.

47. Start the string merge task (DKNMRGE) for the corrected and concatenated Consolidated Reject pocket. The command to use is:

    ```
    MRGE 4,0030,000
    ```

    DKNMRGE combines the interim M-string with the Concatenated Consolidated Reject R-string created in step 42 on page 9-14. Because there are still rejects left to merge, the result is still an interim M-string (0030-1-00-00-00-01-M-000).

48. Start the string merge task (DKNMRGE) for the corrected and concatenated on-us rejects rehandle pocket. The command to use is:

    ```
    MRGE 6,0030,000,02
    ```

    DKNMRGE combines the interim M-string with the Concatenated Alternate Reject R-string created in step 34 on page 9-13. Because there are still rejects left to merge, the result is still an interim M-string (0030-1-00-00-00-01-M-000).

49. Start the string merge task (DKNMRGE) for the corrected and concatenated transit rejects rehandle pocket. The command to use is:

    ```
    MRGE 6,0030,000,03
    ```

    DKNMRGE combines the interim M-string with the Concatenated Alternate Reject R-string created in step 20 on page 9-11. Because these are the last rejects left to merge, the result is an original M-string (0030-1-00-00-00-00-M-000).

50. DKNMRGE then automatically starts the entry master list (DKNPLST). DKNMRGE displays messages on the terminal that started the task. DKNPLST displays messages on the supervisor terminal.

51. The Exception Entry Master List report, produced by DKNPLST, identifies one block as being “OUT OF BALANCE”. The block that is out-of-balance is the block slip with the serial number of 49. This out-of-balance condition is caused by the intentional entry of an incorrect amount during OLRR for this block.

52. We now need to create revised Kill Lists and Cash Letters that include the Consolidated Reject string corrections you made in step 41 on page 9-14. Begin by distributing replacement D-strings from the M-string; enter:

    ```
    MDIS
    ```

    and press **ENTER** to display the menu for DKNMDIS. For the M-string name type:

    ```
    0030-1-00-00-00-00-M-000
    ```

    Then select option 3 and press **ENTER** to verify distribution. The resulting distribution yields D-strings for pockets 04 and 05.

53. To print the revised Kill Lists and Cash Letters, enter the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT031** when prompted for the table name, and select option C on the Cash Letter Override screen.

54. In a production environment, you may at this point find yourself making changes to the M-string (for example, if you use a Balancing and Adjustments

package). These changes may require you to revise Kill Lists and Cash Letters yet again. To do so, enter:

```
MDIS
```

and press **ENTER** to display the menu for DKNMDIS. For the M-string name enter:

```
0030-1-00-00-00-00-M-000
```

Then select option 4 and press **ENTER** to verify distribution. The resulting distribution yields D-strings for pockets 02, 03, 04, and 05. The existing D-strings are always replaced, even if they themselves have been created by MDIS.

55. To produce the revised Kill Lists and Cash Letters, enter the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT033** when prompted for the table name, and select option C on the Cash Letter Override screen. We use endpoint table EPT033 here because it contains every endpoint sorted to by sort pattern 003, and so catches all D-strings that MDIS replaces, no matter what they are.

56. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

57. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

58. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

59. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system.

60. Back in step 54 on page 9-15, MDIS redistributed not only Kill D-strings, but also the rehandle D-strings 0030-1-02-00-00-00-D-000 and 0030-1-03-00-00-00-D-000. However, the corresponding subsequent passes have already been run, back in steps 35 on page 9-13 and 21 on page 9-11, where we used R-strings for the codeline data matching. We therefore will never use these rehandle D-strings. However, unless we do something about them, they will hang around and block the End Cycle process. We must delete them:

    ```
    DELE 0030-1-02-00-00-00-D-000
    DELE 0030-1-03-00-00-00-D-000
    ```

61. Start the Microfilm Report task (DKNFILM). This task produces a report of the records in the Microfilm Data Set and marks them for deletion by DKNCOMP. The command to use is:

    ```
    FILM
    ```

    Select option 1 (print unlisted records) and press **ENTER**.

62. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures there are no active strings present for this cycle, other than the prime-pass M-strings.

    It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

63. Start the Compress task (DKNCOMP). This task deletes fully-processed records from the Kill Bundle and Microfilm Data Sets, and compresses them to free space for the next cycle. The command to use is:

    ```
    COMP
    ```

64. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 10. Sample Problem 3A: Enhanced Reject Processing with DKNEMRG

These instructions demonstrate how to verify that the installation of DKNEMRG was done correctly. DKNEMRG is designed to work with the Enhanced System Manager (ESM) feature of CPCS. It can be run manually, but it does not delete any strings or automatically start any tasks, such as DKNRLST or DKNPLST.

Important!

Do not run DKNEMRG in a production non-ESM environment; use DKNMRGE instead. DKNMRGE contains editing and auditing controls that are used by follow-on tasks.

This sample problem is a follow-on to Chapter 9, "Sample Problem 3: Enhanced Reject Processing" on page 9-1. However, it is run with DKNEMRG rather than DKNMRGE. To run this sample problem, use the same preparation methods and select the same options that apply to Sample Problem 3.

After running this sample problem manually, generate an ESM workflow for this problem and run it again using the Enhanced System Manager. A workflow that automatically starts DIST, SCAT, EMRG, PLST, RLST, and DELE at the appropriate times allows an easier running of the sample problem. Please refer to the *Enhanced System Manager User's Guide* for more information.

Operating Instructions for Sample Problem 3A

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminal by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

   ```
   MICR
   ```

   Open sorter 3 by entering:

   ```
   O 3
   ```

on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 3.

6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 003, and entry number 0030. Press **ENTER**.
7. If this is the first time sorter 3 has run since the most recent cold start, you are prompted for a microfilm sequence number. Enter any six-digit value. Press **ENTER** again to confirm the Begin screen data and start the actual capture.
8. End the prime-pass run when you see the `Intervention Required` message on the MICR status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.
9. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. The distribution task (DKNDIST) displays a message on the supervisor terminal indicating the end of the distribution task.
10. Start the MICR subsequent pass for the On-us and Second Deadline Transit items sorted to the first rehandle pocket (pocket 01) on prime pass for this entry.
    a. Open sorter 3 and proceed to the Begin screen.
    b. On the Begin screen, enter:

       ,,`0030-010`

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 01. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
11. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
12. Start the MICR subsequent pass for the Second Deadline Transit items that sorted to rehandle pocket 04 during step 10.
    a. Press **ENTER** to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.
    b. On the Begin screen, enter:

       ,,`0030-011`

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04 in step 10. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
13. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
14. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

15. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.

16. Start the Kill List task (DKNKILL) for the Second Deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and select option C on the Cash Letter Override screen. This causes DKNKILL to print Cash Letters as well as Kill lists.

17. Use DKNOLRR to correct the rejects in the First and Second Transit Deadline items sorted to the third rehandle pocket (pocket 03) on prime pass for this entry. The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0030-1-Z3-00-00-00-D-000
    ```

    The 'Z' in the pocket number identifies this string as containing rejects. Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.

18. In this sample problem, incorrect items appear with asterisks (*) in place of unreadable digits. Replace all the asterisks with 1's.

19. When OLRR completes, it creates string 0030-1-03-00-00-01-R-000. Note that pocket "Z3" has been renamed "03", to indicate that it no longer contains rejects. The "01" in the fourth pocket position names the OLRR operator ID.

20. Concatenate all of OLRR's partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as, among other things, it updates the tracer group data set.) The command to use is:

    ```
    SCAT 0030-1-03-00-00-01-R-000
    ```

    SCAT creates a new string 0030-1-03-00-00-00-R-000 for use by DKNEMRG, and by the sorter's subsequent pass codeline data match routines.

21. Start the Enhanced Merge task (DKNEMRG) for the items you have just corrected. The command to use is:

    ```
    EMRG 0030-1-03-00-00-00-R-000
    ```

    DKNEMRG combines the prime pass I-string with the Concatenated Alternate Reject R-string created in step 20. Because there are still rejects left to merge, the result is an interim M-string (0030-1-00-00-00-01-M-000).

22. Start the MICR subsequent pass for the items you have just corrected.

    a. Open sorter 4. Sorter 4 is defined as a power-encode sorter. If you are running this sample problem on a live power-encode sorter, then it prints a fresh MICR line on each item it reads. These replacement MICR lines include all the corrections you keyed in step 18 on page 10-3.

b. Go to the Begin screen and enter:

```
,,0030-019
```

That is, the tracer and slip number of the first tracer to sort to prime pass rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

c. At this point, the following warning message appears:

```
WARNING=MICMSPFL 1002 FLD09 BYTE LEN 045 IS
GREATER THAN MDX LEN 000
```

What has happened is that the sort pattern uses a special MIPI exit--DKNMXS32--to download to the sorter power-encode information in an otherwise unused field (field 9). MICR is simply noting that field 9 appears not to be empty. You can safely ignore the message.

d. Press **ENTER** to accept the data on the MICR Begin screen and start the actual capture.

23. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

24. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

25. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

26. Start the Kill List task (DKNKILL) for the First Deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT031** when prompted for the table name, and select option C on the Cash Letter Override screen. This causes DKNKILL to print Cash Letters as well as Kill lists.

27. Start the MICR subsequent pass for the Second Deadline Transit items that sorted to rehandle pocket 03 during step 22 on page 10-3.

    a. Open sorter 3 and proceed to the Begin screen.

    b. On the Begin screen, enter:

    ```
    ,,0030-020
    ```

    That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03 in step 22 on page 10-3. The two commas ensure the data winds up in the ".ENTRY" field.

    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

28. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

29. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

30. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

31. Start the Kill List task (DKNKILL) for the Second Deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), and enter **EPT002** when prompted for the table name, and select option C on the Cash Letter Override screen. This causes DKNKILL to print Cash Letters as well as Kill lists.

32. Use DKNOLRR to correct the rejects in the On-us items sorted to the second rehandle pocket (pocket 02) on prime pass for this entry. The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0030-1-Z2-00-00-00-D-000
    ```

    The 'Z' in the pocket number identifies this string as containing rejects. Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.

33. In this sample problem, incorrect items appear with asterisks (*) in place of unreadable digits. Replace all the asterisks with 1's.

34. When OLRR completes, it creates string 0030-1-02-00-00-01-R-000. Note that pocket "Z2" has been renamed "02", to indicate that it no longer contains rejects. The "01" in the fourth pocket position names the OLRR operator ID.

35. Concatenate all of OLRR's partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as, among other things, it updates the tracer group data set.) The command to use is:

    ```
    SCAT 0030-1-02-00-00-01-R-000
    ```

    SCAT creates a new string 0030-1-02-00-00-00-R-000 for use by DKNEMRG, and by the sorter's subsequent pass codeline data match routines.

36. Start the Enhanced Merge task (DKNEMRG) for the items you have just corrected. The command to use is:

    ```
    EMRG 0030-1-02-00-00-00-R-000
    ```

    DKNEMRG combines the interim M-string with the Concatenated Alternate Reject R-string created in step 35. Because there are still rejects left to merge, the result will still be an interim M-string 0030-1-00-00-00-01-M-000).

37. Start the MICR subsequent pass for the items you have just corrected.

a. Open sorter 4. Sorter 4 is defined as a power-encode sorter. If you are running this sample problem on a live power-encode sorter, then it will print a fresh MICR line on each item it reads. These replacement MICR lines include all the corrections you keyed in step 33.

b. Go to the Begin screen and enter:

```
,,0030-016
```

That is, the tracer and slip number of the first tracer to sort to prime-pass rehandle pocket 02. The two commas ensure the data winds up in the ".ENTRY" field.

c. At this point, the following warning message appears:

```
WARNING=MICMSPFL 1002 FLD09 BYTE LEN 045 IS
GREATER THAN MDX LEN 000
```

What has happened is that the sort pattern uses a special MIPI exit--DKNMXS32--to download to the sorter power-encode information in an otherwise unused field (field 9). MICR is simply noting that field 9 appears not to be empty. You can safely ignore the message.

d. Press **ENTER** to accept the data on the MICR Begin screen and start the actual capture.

38. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

39. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

40. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

41. Run DKNDIST to distribute the Consolidated Reject pocket. To do this, enter the command:

    ```
    DIST
    ```

    and press **ENTER** to display the initial screen for DKNDIST. Next, enter the I-string name:

    ```
    0030-1-00-00-00-00-I-000
    ```

    The next menu from DIST asks for the type of distribution. Enter:

    ```
    RC
    ```

    DKNDIST ends and creates a Consolidated Reject D-string, 0030-1-RC-00-00-00-D-000, that contains items from all Eligible-to-Receive pockets (in this case, prime-pass pockets 04 and 05).

42. Because a Consolidated Reject D-string contains items from several different pockets, it is normally infeasible to OLRR it. The preferred method would be to use some sort of Key Entry system (this is why sorter 3 is defined as having an Image camera).

However, in this sample problem we know exactly which items sorted to the Eligible-to-Receive pockets, so we can use OLRR to substitute for Key Entry.

The command to use is:

```
OLRR
```

DKNOLRR displays a screen that includes a default string name. Replace it with:

```
0030-1-RC-00-00-00-D-000
```

Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.

43. In this sample problem, incorrect items appear with asterisks (*) in place of unreadable digits. Replace all the asterisks with 1's.

44. Concatenate all of OLRR's partial R-strings by running DKNSCAT. (Note that, in this sample problem we have created only one partial R-string. DKNSCAT must still be run, however, as DKNEMRG expects concatenated R-strings.) The command to use is:

    ```
    SCAT 0030-1-RC-00-00-01-R-000
    ```

    SCAT creates a new string 0030-1-RC-00-00-00-R-000 for use by DKNEMRG.

45. Start the Enhanced Merge task (DKNEMRG) for the items you have just corrected. The command to use is:

    ```
    EMRG 0030-1-RC-00-00-00-R-000
    ```

    DKNEMRG combines the interim M-string with the Concatenated Consolidated Reject R-string created in step 44. Because there are still rejects left to merge, the result will still be an interim M-string (0030-1-00-00-00-01-M-000).

46. Use DKNOLRR to partially correct the rejects from the prime pass System Reject pocket. The command to use is:

    ```
    OLRR 0030,17
    ```

    where the 17 indicates you are OLRR operator number 17.

    You will be prompted to specify a starting document. Simply press **ENTER** to start at the beginning of the System Reject D-string.

    Your ending document is block 0061111111 (it appears on the first screen). As soon as you see it, tab to the colon (:) field at the bottom of the screen and enter **+END**.

    This partial R-string has no rejects to correct.

47. When OLRR ends, it automatically starts DKNSCAT. Since not all of the System Reject pocket has been corrected yet, SCAT creates an Interim System Reject Concatenated R-string (0030-1-R -00-00-99-R-000). Watch for supervisor terminal messages indicating that DKNSCAT has ended.

48. Use DKNOLRR to partially correct the rejects from the prime-pass System Reject pocket. The command to use is:

    ```
    OLRR 0030,23
    ```

    where the 23 indicates you are OLRR operator number 23.

    You will be prompted to specify a starting document. Tab to the BLOCK field and enter **0200006666**.

Your ending document is block 0000030000. As soon as you see it, tab to the colon (:) field at the bottom of the screen and enter **+END**.

Along the way, you will encounter an item that contains a digit error. Correct the error as follows:

- R/T error: `12*122222` should be `122122222`

49. When OLRR ends, it automatically starts DKNSCAT. Since not all of the System Reject pocket has been corrected yet, SCAT creates an Interim System Reject Concatenated R-string (0030-1-R -00-00-99-R-000). Watch for supervisor terminal messages indicating that DKNSCAT has ended.
50. Use DKNOLRR to partially correct the rejects from the prime-pass System Reject pocket. The command to use is:

    ```
    OLRR 0030,19
    ```

    where the 19 indicates you are OLRR operator number 19.

    You will be prompted to specify a starting document. Tab to the BLOCK field and enter **0000022222**.

    Continue processing the System Reject D-string until you reach its end. At that point OLRR automatically ends.

    This partial R-string has no rejects to correct.
51. When OLRR ends, it automatically starts DKNSCAT. Since now the entire System Reject pocket has been corrected, SCAT creates a true Concatenated System Reject R-string (0030-1-R -00-00-00-R-000). Watch for supervisor terminal messages indicating that DKNSCAT has ended.
52. Start the Enhanced Merge task (DKNEMRG) for the items you have just corrected. The command to use is:

    ```
    EMRG 0030-1-R -00-00-00-R-000
    ```

    DKNEMRG combines the interim M-string with the Concatenated System Reject R-string created in step 51. Because these are the last rejects left to merge, the result is an original M-string (0030-1-00-00-00-00-M-000).
53. DKNEMRG does not automatically start DKNPLST or DKNRLST, nor does it delete any strings. These functions are normally taken care of by the ESM workflows. Because we have been manually running DKNEMRG, we should at this point:

    a. Enter:

       ```
       RLST
       ```

       to start DKNRLST. When RLST starts, enter:

       ```
       1,0030
       ```

       to print the System Reject Pocket reject list.

    b. Enter:

       ```
       PLST 0030
       ```

       to start DKNPLST. When PLST starts, select option 3 to get the Exception Entry Master List.

    c. Enter:

       ```
       DELE 0030-1-00-00-00-00-I-000
       DELE 0030-1-R -00-00-00-D-000
       ```

to delete the I and System Reject D-strings.

54. The Exception Entry Master List report, produced by DKNPLST, identifies one block as being “OUT OF BALANCE”. The block that is out-of-balance is the block slip with the serial number of 49. This out-of-balance condition is caused by the intentional entry of an incorrect amount during OLRR for this block.

55. We now need to create revised Kill Lists and Cash Letters that include the Consolidated Reject string corrections you made in step 43 on page 10-7. Begin by distributing replacement D-strings from the M-string. Enter:

    ```
    MDIS 0030-1-00-00-00-00-M-000,3
    ```

    and press **ENTER**. The resulting distribution yields D-strings for pockets 04 and 05.

56. To print the revised Kill Lists and Cash Letters, enter the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT031** when prompted for the table name, and select option C on the Cash Letter Override screen.

57. In a production environment, you may at this point find yourself making changes to the M-string (for example, if you use a Balancing and Adjustments package). These changes may require you to revise Kill Lists and Cash Letters yet again. To do so, enter:

    ```
    MDIS 0030-1-00-00-00-00-M-000,4
    ```

    and press **ENTER**. The resulting distribution yields D-strings for pockets 02, 03, 04, and 05. The existing D-strings will always be replaced, even if they themselves had been created by MDIS.

58. To produce the revised Kill Lists and Cash Letters, enter the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT033** when prompted for the table name, and select option C on the Cash Letter Override screen. We use endpoint table EPT033 here because it contains every endpoint sorted to by sort pattern 003, and so will catch all D-strings that MDIS replaces, no matter what they are.

59. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

60. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

61. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

62. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system.

63. Back in step 57 on page 10-9, MDIS redistributed not only Kill D-strings, but also the rehandle D-strings 0030-1-02-00-00-00-D-000 and 0030-1-03-00-00-00-D-000. However, the corresponding subsequent passes have already been run, back in steps 37 on page 10-5 and step 22 on page 10-3, where we used R-strings for the codeline data matching. We therefore never use these rehandle D-strings. However, unless we do something about them, they will hang around and block the End Cycle process. We need to delete them:

    ```
    DELE 0030-1-02-00-00-00-D-000
    DELE 0030-1-03-00-00-00-D-000
    ```

64. Start the Microfilm Report task (DKNFILM). This task produces a report of the records in the Microfilm data set and marks them for deletion by DKNCOMP. The command to use is:

    ```
    FILM
    ```

    Select option 1 (print unlisted records) and press **ENTER**.

65. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures that there are no active strings present for this cycle, other than the prime-pass M-strings.

    It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

66. Start the Compress task (DKNCOMP). This task deletes fully-processed records from the Kill Bundle and Microfilm data sets, and compresses them to free space for the next cycle. The command to use is:

    ```
    COMP
    ```

67. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 11. Sample Problem 4: Unqualified Data

This sample problem simulates the use of a power encoder to inscribe unqualified fields. The prime pass sorts to three on-us kill pockets (a credit kill pocket, a debit kill pocket, and a returns pocket) and two transit rehandle pockets.

One of the transit rehandles (the second deadline) is set up to require power-encoding. The rehandle pocket is defined as unencoded. You can process the items that sort to this pocket through a key entry application, where the amounts are entered. You can then run the subsequent pass on a sorter equipped with the power encode feature. There the amounts are inscribed on their paper documents, which are then sorted to their final kill pockets.

With the exception of one returns item, none of the other documents have amounts, either. You must use key entry on these items as well. However, in this sample problem, it has not been deemed necessary to power-encode the amounts back on to the paper.

Preparing to Run the Sample Problem

The test material contains:

- DKNPCPCS parameters for CPCS execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, and batch slips
- SCI and OLRR definitions for sorting

CPCS Parameter Generation Options

CPCS requires no special options in the DKNPCPCS System Profile control card for this sample problem.

DKNSPDEF Options

This sample problem uses member SPTYP004 in CPCS.V1R11.SAMPLIB as the sort-pattern definition.

```
*          TEST DECK:        DKNT04SU
*          MERGE FEED DECK:  DKNT01MU
*          ENTRY:            0040
*
*      PPH    | SCI/N.L. EDIT |   OLRR EDIT
*  -----------|---------------|---------------
*  1-00-00-00 | DKN0041Z      | DKN0041R
*  2-03-00-00 | DKN0042Z      |
*  2-04-00-00 | DKN0043Z      |
*
************************************************************
*
P1000000DKN0041Z        021           10030XF          11 2
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 4 - UNQUAL DATA - PRIME PASS
J   RRUR
O                           Y
R    0303
K018888888802888888880588888888
*
************************************************************
*
P2030000DKN0042Z          1            0030XF
B   001
RP      SAMPXF01                        PE
BMSG  SAMPLE PROBLEM 4 - UNQUAL DATA - QUALIFIED TRANSITS
H        Y   P YY      1
FLD01     005  10             N
K0102100002020213023503021000300402100008050200000 0
*
************************************************************
*
P2040000DKN0043Z          1            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 4 - UNQUAL DATA - UNQUALIFIED TRANSITS
K0109000000021000000003110000000412000000
*
```

Figure 11-1. Sample Problem 4

Pocket Selection for Sample Problem 4

Tables 11-2 through 11-4 describe the contents of each of the pockets for this sample problem. In the column labeled Routing/Transit, the letter *n* can be replaced by any digit.

Figure 11-2. Sample Problem 4: Pocket Selection Criteria–Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	11nn-nnnn	PC=AAAA33	On-us Credits; items are unqualified.
02	Kill	11nn-nnnn	PC=AAAAAA	On-us Debits; items are unqualified.
03	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transits; items are unqualified.
04	R/H	12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn	N/A	First Deadline Transits; items are unqualified.
05	Kill	11nn-nnnn	EXT PC not = AA and PC=AAAAAA	On-us Debit Returns

Figure 11-3. Sample Problem 4: Pocket Selection Criteria–First Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02100002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Figure 11-4. Sample Problem 4: Pocket Selection Criteria–Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	12nn-nnnn	N/A	First Deadline Transit EP=09000000; items are unqualified.
02	Kill	13nn-nnnn	N/A	First Deadline Transit EP=10000000; items are unqualified.
03	Kill	14nn-nnnn	N/A	First Deadline Transit EP=11000000; items are unqualified.
04	Kill	15nn-nnnn	N/A	First Deadline Transit EP=12000000; items are unqualified.
05	---	N/A	N/A	Not used

Data for Sample Problem 4

The data for this sample problem consists of one set of tracers with associated data, including block and batch slips. Only control documents and one returns item contain amounts; all other documents are unencoded.

The member DKNT04SU in CPCS.V1R11.SAMPLIB contains this input. If you use the simulator for this sample problem, you must include the fields represented by periods as shown in the records. If you use a physical document processor, do not encode the fields represented by the periods.

Reports for Sample Problem 4

The reports that CPCS produces when you run this sample problem are in CPCS.V1R11.RPTLIB. Print out member SAMPLE4 from this library and compare it with your results for accuracy.

Operating Instructions for Sample Problem 4

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminal by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

   ```
   MICR
   ```

   Open sorter 4 by entering:

   ```
   O 4
   ```

   on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 4.
6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 004, and entry number 0040. Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.
7. End the prime-pass run when you see the `Intervention Required` message on the MICR status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.
8. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. The distribution task (DKNDIST) displays a message on the supervisor terminal indicating the end of the distribution task.
9. Use DKNOLRR to simulate Key Entry on the First Deadline Transit Rehandle pocket (pocket 04). The command to use is:

   ```
   OLRR
   ```

   DKNOLRR displays a screen that includes a default string name. Replace it with:

   ```
   0040-1-Z4-00-00-00-D-000
   ```

   The 'Z' in the pocket number identifies this string as containing rejects. Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.
10. In this sample problem, the account field of each item contains the value that should be encoded in the amount field. Copy the accounts to the amounts.

11. When OLRR completes, it creates string 0040-1-04-00-00-01-R-000. Note that pocket “Z4” has been renamed “04”, to indicate that it no longer contains rejects. The “01” in the fourth pocket position names the OLRR operator ID.
12. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as, among other things, it updates the tracer group data set.) The command to use is:

    ```
    SCAT 0040-1-04-00-00-01-R-000
    ```

    SCAT creates a new string 0040-1-04-00-00-00-R-000 for use by DKNMRGE, and by the sorter’s subsequent pass codeline data match routines.
13. Use DKNOLRR to correct the prime-pass System Reject Pocket. The command to use is:

    ```
    OLRR 0040,01
    ```

    One document in this example contains a digit error. Correct the error as follows:

    - R/T error: `13*122225` should be `132122225`
    - Copy the account number to the amount field.
14. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.
15. Start the string merge task (DKNMRGE) for the corrected and concatenated prime-pass System Reject pocket. The command to use is:

    ```
    MRGE 1,0040
    ```

    DKNMRGE combines the prime pass I-string with the Concatenated System Reject R-string created in step 14. Because there are still rejects left to merge, the result is an interim M-string (0040-1-00-00-00-01-M-000).
16. DKNMRGE then automatically starts the formatted R-string list task (DKNRLST) for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNRLST displays messages on the supervisor terminal.
17. Start the string merge task (DKNMRGE) for the corrected and concatenated First Deadline Transit Rehandle pocket. The command to use is:

    ```
    MRGE 5,0040,04
    ```

    DKNMRGE combines the interim M-string with the Concatenated Alternate Reject R-string created in step 12. Because there are still rejects left to merge, the result is still an interim M-string (0040-1-00-00-00-01-M-000).
18. Start the MICR subsequent pass for the First Deadline Transit items.
    a. Open sorter 4 and proceed to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0040-013
       ```

       That is, the tracer and slip number of the first tracer to sort to prime pass rehandle pocket 04. The two commas ensure the data winds up in the “.ENTRY” field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

19. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

20. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

21. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to product a Subsequent Pass Master List.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

22. Start the Kill List task (DKNKILL) for the first deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT001** when prompted for the table name, and select option D on the Cash Letter Override screen.

23. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT001** when prompted for the table name. When DKNCLSM prompts you to respond to whether this is a duplicate letter, select NO.

24. Use DKNOLRR to simulate Key Entry on the Second Deadline Transit Rehandle pocket (pocket 03). The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0040-1-Z3-00-00-00-D-000
    ```

    The ‘Z’ in the pocket number identifies this string as containing rejects. Leave the operator ID as “01”, and the BYPASS and RESTART options as “N”. Press **ENTER**.

25. In this sample problem, the account field of each item contains the value that should be encoded in the amount field. Copy the accounts to the amounts.

26. When OLRR completes, it creates string 0040-1-03-00-00-01-R-000. Note that pocket “Z3” has been renamed “03”, to indicate that it no longer contains rejects. The “01” in the fourth pocket position names the OLRR operator ID.

27. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as, among other things, it updates the tracer group data set.) The command to use is:

    ```
    SCAT 0040-1-03-00-00-01-R-000
    ```

    SCAT creates a new string 0040-1-03-00-00-00-R-000 for use by DKNMRGE, and by the sorter’s subsequent pass codeline data match routines.

28. Start the MICR subsequent pass to power-encode the Second Deadline Transit items you just corrected.

a. Open sorter 4 and proceed to the Begin screen.

b. On the Begin screen, enter:

   ,,0040-010

   That is, the tracer and slip number of the first tracer to sort to prime pass rehandle pocket 03. The two commas ensure the data winds up in the ".ENTRY" field.

c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

29. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

30. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the CLOSE command.

31. The distribution task started automatically when you ended the entry. Because the entry was subsequent pass, DKNDIST also automatically started DKNSLST to produce a Subsequent Pass Master list.

    The Subsequent Pass Master List identifies, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For this entry, there are no exception conditions.

32. Start the Kill List task (DKNKILL) for the second deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and select option D on the Cash Letter Override screen.

33. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT002** when prompted for the table name. When DKNCLSM prompts you to respond to whether this is a duplicate letter, select NO.

34. Use DKNOLRR to simulate Key Entry on the On-us Credits pocket (pocket 01). The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0040-1-Z1-00-00-00-D-000
    ```

    The 'Z' in the pocket number identifies this string as containing rejects. Leave the operator ID as "01", and the BYPASS and RESTART options as "N". Press **ENTER**.

35. In this sample problem, the account field of each item contains the value that should be encoded in the amount field. Copy the accounts to the amounts.

36. When OLRR completes, it creates string 0040-1-01-00-00-01-R-000. Note that pocket “Z1” has been renamed “01”, to indicate that it no longer contains rejects. The “01” in the fourth pocket position names the OLRR operator ID.

37. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT must still be run, however, as DKNMRGE expects concatenated R-strings.) The command to use is:

    ```
    SCAT 0040-1-01-00-00-01-R-000
    ```

    SCAT creates a new string 0040-1-01-00-00-00-R-000 for use by DKNMRGE.

38. Use DKNOLRR to simulate Key Entry on the On-us Debits pocket (pocket 02). The command to use is:

    ```
    OLRR
    ```

    DKNOLRR displays a screen that includes a default string name. Replace it with:

    ```
    0040-1-Z2-00-00-00-D-000
    ```

    The ‘Z’ in the pocket number identifies this string as containing rejects. Leave the operator ID as “01”, and the BYPASS and RESTART options as “N”. Press **ENTER**.

39. In this sample problem, the account field of each item contains the value that should be encoded in the amount field. Copy the accounts to the amounts.

40. When OLRR completes, it creates string 0040-1-02-00-00-01-R-000. Note that pocket “Z2” has been renamed “02”, to indicate that it no longer contains rejects. The “01” in the fourth pocket position names the OLRR operator ID.

41. Concatenate all of OLRR’s partial R-strings by running DKNSCAT. (Note that, in this sample problem, we have created only one partial R-string. DKNSCAT have created only one partial R-string. DKNSCAT must still be run, however, as DKNMRGE expects concatenated R-strings). The command to use is:

    ```
    SCAT 0040-1-02-00-00-01-R-000
    ```

    SCAT creates a new string 0040-1-02-00-00-00-R-000 for use by DKNMRGE.

42. Start the string merge task (DKNMRGE) for the corrected and concatenated On-us Credits pocket. The command to use is:

    ```
    MRGE 5,0040,01
    ```

    DKNMRGE combines the interim M-string with the Concatenated Alternate Reject R-string created in step 37. Because there are still rejects left to merge, the result is still an interim M-string (0040-1-00-00-00-01-M-000).

43. Start the string merge task (DKNMRGE) for the corrected and concatenated On-us Debits pocket. The command to use is:

    ```
    MRGE 5,0040,02
    ```

    DKNMRGE combines the interim M-string with the Concatenated Alternate Reject R-string created in step 41. Because there are still rejects left to merge, the result is still an interim M-string (0040-1-00-00-00-01-M-000).

44. Start the string merge task (DKNMRGE) for the corrected and concatenated First Deadline Transit pocket. The command to use is:

    ```
    MRGE 5,0040,03
    ```

DKNMRGE combines the interim M-string with the Concatenated Alternate Reject R-string created in step 27 on page 11-8. Because these are the last rejects left to merge, the result is an original M-string (0040-1-00-00-00-00-M-000).

45. DKNMRGE then automatically starts the entry master list (DKNPLST). DKNMRGE displays messages on the terminal that started the task. DKNPLST displays messages on the supervisor terminal.

46. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

47. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

48. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

49. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system.

50. There are four Alternate Reject D-strings left over at the end of processing. These must be deleted before we can run end cycle. Enter the commands:

    ```
    DELE 0040-1-Z1-00-00-00-D-000
    DELE 0040-1-Z2-00-00-00-D-000
    DELE 0040-1-Z3-00-00-00-D-000
    DELE 0040-1-Z4-00-00-00-D-000
    ```

51. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures that there are no active strings present for this cycle, other than the prime-pass M-strings.

    It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

52. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 12. Sample Problem 5: Divider Re-synchronization

This sample problem shows the use of divider documents to re-synchronize data during a subsequent pass. It cannot be run on a simulator (this is because the simulator, on a subsequent pass, draws its input data from the codeline data-match buffers. The whole point of the test is to present input data that differs from that in the codeline data-match buffers).

The sorter definitions in CPCS.V1R11.SAMPLIB(DKNMGEN) and in the run JCL CPCS.V1R11.CTRL(DKNJRUN) include a six-pocket channel-attached 3890/XP sorter, number 005. Please use this sorter when running this sample problem.

Preparing to Run the Sample Problem

The test material contains:

- DKNPCPCS parameters for DKNMTASK execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, and batch slips. This sample data is encoded on live items for the test.
- SCI and OLRR definitions for sorting

CPCS Parameter Generation Options

CPCS requires no special options in the DKNPCPCS System Profile control card for this sample problem.

DKNSPDEF Options

This sample problem uses member SPTYP005 in CPCS.V1R11.SAMPLIB as the sort-pattern definition. The O-record contains all the options for divider re-synchronization:

Position	Option
5	Divider Re-Synchronization Indicator <blank> No re-synchronization D Select a divider into a rehandle pocket every time its item count exceeds the number in O record positions 9 - 12. Record information on the divider into the Divider Data Set and use this data to re-synchronize on subsequent passes.
9 - 12	Rehandle pocket divider merge count. Every time the number of items in a rehandle pocket exceeds this value, select a divider from the merge feed hopper into the rehandle pocket.

```
*
*          TEST DECK:        DKNT05SU (LIVE ITEMS)
*          MERGE FEED DECK:  DKNT01MU (LIVE ITEMS)
*          ENTRY:            0050
*
*      PPH    | SCI/N.L. EDIT |   OLRR EDIT
*  -----------|---------------|--------------
*  1-00-00-00 | DKN0051Z      | DKN0051R
*  2-03-00-00 | DKN0052Z      |
*  2-05-00-00 | DKN0053Z      |
*
*********************************************************
*
P1000000DKN0051Z        021           20030XF          11
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 5 - DIVIDER       RESYNCH - PRIME PASS
O   D   0007
R    0303
K018888888802888888880588888888
*
*********************************************************
*
P2030000DKN0052Z          2            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 5 - DIVIDER       RESYNCH - 1ST TRANSIT DEADLINE
O   D   0007
K0109000000021000000003110000000412000000
*
*********************************************************
*
P2040000DKN0053Z          2            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 5 - DIVIDER       RESYNCH - 2ND TRANSIT DEADLINE
O   D   0007
K010210000202021302350302100030040210000805020000000
*
```

Figure 12-1. Sample Problem 5

Pocket Selection for Sample Problem 5

Tables 12-2 through 12-4 describe the contents of each of the pockets for this sample problem. In the column labeled Routing/Transit, the letter *n* can be replaced by any digit.

Figure 12-2. Sample Problem 1: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	11nn-nnnn	PC=AAAA33	On-us Credits
02	Kill	11nn-nnnn	PC=AAAAAA	On-us Debits
03	R/H	12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn	N/A	First Deadline Transits
04	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transits
05	Kill	11nn-nnnn	EXT PC not = AA and PC=AAAAAA	On-us Debit Returns

Figure 12-3. Sample Problem 1: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	12nn-nnnn	N/A	First Deadline Transit EP=09000000
02	Kill	13nn-nnnn	N/A	First Deadline Transit EP=10000000
03	Kill	14nn-nnnn	N/A	First Deadline Transit EP=110000000
04	Kill	15nn-nnnn	N/A	First Deadline Transit EP=120000000
05	---	N/A	N/A	Not used

Figure 12-4. Sample Problem 1: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02010002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=0200030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Data for Sample Problem 5

The data for this sample problem consists of one set of tracers with associated data, including block, batch, and subbatch slips. This data also contains deliberate errors to show the balancing features of the system.

The member DKNT05SU in CPCS.V1R11.SAMPLIB contains this input. Each line of the member specifies, in order, the contents of fields 7, 6, 5, 3, 2, and 1 (field 4 is not listed as it is not normally present in the standard MICR codeline). You must encode each line of DKNT05SU on a separate physical paper item.

You'll notice that several of the fields are represented by periods. Do not encode the periods; they are simply there to show you where blank, empty fields are.

There are three documents in DKNT05SU where a field contains an asterisk (*). These asterisks indicate where you need to make the sorter report a digit error. Encode the field as normal; then, deliberately deface the digit in question to make it unreadable.

Reports for Sample Problem 5

The reports that CPCS produces when you run this sample problem are in CPCS.V1R11.RPTLIB. Print out member SAMPLE5 from this library and compare it with your results for accuracy.

Operating Instructions for Sample Problem 5

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminal by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing **PF1** and then **ENTER**.
4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.
5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

   ```
   MICR
   ```

   Open sorter 5 by entering:

   ```
   O 5
   ```

   on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 5.
6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 005, and entry number 0050. Press **ENTER**, then press **ENTER** again to confirm your Begin screen data.
7. Place your encoded input deck into sorter 5's hopper, then press the START button to begin the actual capture.
8. End the prime-pass run when you see the `Intervention Required` message on the MICR status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.
9. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. The distribution task (DKNDIST) displays a message on the supervisor terminal indicating the end of the distribution task.
10. Collect the items from rehandle pocket 3 and set them aside for use in step 17 on page 12-6.
11. Collect the items from rehandle pocket 4 and set them aside for use in step 20 on page 12-6.
12. Use DKNOLRR to correct the rejects from the prime pass. The command to use is:

    ```
    OLRR 0050,01
    ```

    Three documents in the example contain digit errors. Correct the errors as follows:

    - R/T error: `12*122222` should be `122122222`

- Amt error: `00000*0002` should be `0000010002`
- R/T error: `1110006*7` should be `111000627`

13. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.
14. Start the string merge task (DKNMRGE) for the corrected entry (against which SCAT was run) with the command:

    ```
    MRGE 1,0050
    ```

    DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the concatenated R-string produced by SCAT.
15. DKNMRGE then automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.
16. The Exception Entry Master List report, produced by DKNPLST, identifies one subbatch as being “OUT OF BALANCE”. The subbatch that is out-of-balance is the subbatch slip with the serial number of 51. Because this subbatch is out-of-balance, the batch and block are also out of balance.

    This out-of-balance condition is caused by the intentional entry of an incorrect amount ($100.02) during OLRR for this subbatch. ($.02 is the correct amount.)
17. Fetch the rehandle pocket 3 items you set aside in step 10 on page 12-5. Remove all the items beginning with the first divider slip and continuing up to, but not including, the second divider slip. Place the items at the end of the rehandle deck.
18. Start the MICR subsequent pass for the documents sorted to the first rehandle pocket (pocket 03) on prime pass for this entry.
    a. Open sorter 5 and proceed to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0050-010
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 03. The two commas ensure the data winds up in the “.ENTRY” field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
    d. Place the deck you modified in step 17 into sorter 5’s hopper, then press the START button to begin the actual capture.
19. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
20. Fetch the rehandle pocket 4 items you set aside in step 11 on page 12-5. Remove all the items beginning with the first divider slip and continuing up to, but not including, the second divider slip. Place the items at the end of the rehandle deck.
21. Start the MICR subsequent pass for the documents sorted to the second rehandle pocket (pocket 04) on prime pass for this entry.

a. Press **ENTER** to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.

b. On the Begin screen, enter:

```
,,0050-013
```

That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.

c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.

d. Place the deck you modified in step 20 on page 12-6 into sorter 5's hopper, then press the START button to begin the actual capture.

22. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.

23. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.

24. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs.

    In this sample problem, there are no exception conditions despite the fact that in steps 17 on page 12-6 and 20 on page 12-6, we deliberately introduced hand-swaps. This is because every time MICR encountered misplaced dividers, it looked them up in the Divider Data Set and from the data therein accurately adjusted its location in the Codeline Data Match buffers to compensate.

25. Start the Kill List task (DKNKILL) for the first deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT001** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT001 contains the endpoints for each first deadline kill pocket.

26. Start the Kill List task (DKNKILL) for the second deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT002 contains the endpoints for each second deadline kill pocket.

27. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT003** when prompted for the table name. This table contains all the endpoints from all deadlines, both first

and second. When DKNCLSM prompts you to respond to whether this is a duplicate letter, press **ENTER** to select NO.

28. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

29. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

30. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

31. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system. After you run this task, the only string left on the mass data set is the M-string for the prime pass.

32. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

    This task ensures that there are no active strings present for this cycle, other than the prime-pass M-strings.

    It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

33. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Chapter 13. Sample Problem 6: Base Functions with Expanded MDS

This sample problem shows the basic functions of CPCS with an expanded mass data set (MDS). It guides you through a sample run that shows the following:

- Data capture
- Distribution
- Online reject reentry (OLRR)
- Merging of the repaired data with the input data
- Producing kill lists and cash letters
- Subsequent-pass processing
- End-of-cycle processing

Preparing to Run the Sample Problem

The test material contains:

- MDX parameters for specifying an expanded Mass Data Set
- DKNPCPCS parameters for CPCS execution
- DKNSPDEF sort pattern definition
- Sample data containing tracer, block, and batch slips
- SCI and OLRR definitions for sorting

MDX Parameters

This sample problem shows you how the system operates in an expanded environment.

To expand the Mass Data Set, follow the instructions in the *CPCS Customization Guide*, Chapter 2 “Installation and Generation Procedures”, section “Expanding the Mass Data Set”. You need to modify the MDX macro imbedded in the GENCLIB JCL (referred to by step 1 in the *CPCS Customization Guide’s* instructions). Change it to read:

```
MDX MDXF01,8,' AMOUNT     ',16,R,L,00,16,             X
    MDXF09,50,' FIELD 09   ',50,L,R,40,50,           X
    MDXF12,50,' FIELD 12   ',50,L,R,40,50
```

This creates a Mass Data Set with an eight-byte (sixteen-digit) amount field, a 50-byte field 9, and a 50-byte field 12.

Follow the remainder of the *CPCS Customization Guide’s* instructions to expand the copybooks, recompile source, reallocate data sets, update DKNDSAT, perform a MICR generation, and update DKNBLDL. In step 3 on page 13-5, use a BLKSIZE of 3072 for the Mass Data Set. Do not cold-start CPCS (step 5 on page 13-6) until after you have updated the DKNPCPCS BLKSIZE parameter, as described below.

CPCS Parameter Generation Options

Change the BLKSIZE parameter in the CPCS system profile member DKNPCPCS to BLKSIZE=3060. This block size is a multiple of the expanded MDS record size of 153.

Once you have changed BLKSIZE, you are ready to perform the cold-start necessary to complete expansion of the Mass Data Set.

DKNSPDEF Options

This sample problem uses member SPTYP006 in CPCS.V1R11.SAMPLIB as the sort-pattern definition. Note that that there are no FLD*nn* parameters. MICR defaults to using the expanded MDS sizes for fields 1, 9, and 12.

```
*
*          TEST DECK:        DKNT06SU
*          MERGE FEED DECK:  DKNT01MU
*          ENTRY:            0060
*
*      PPH    | SCI/N.L. EDIT |   OLRR EDIT
*  -----------|---------------|---------------
*  1-00-00-00 | DKN0061Z      | DKN0061R
*  2-04-00-00 | DKN0062Z      |
*  3-04-05-00 | DKN0063Z      |
*
*********************************************************
*
P1000000DKN0061Z        011           20030XF          11
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 6 - EXPANDED MDS -PRIME PASS
R       06
K01777700010288888888038888888880588888888
*
*********************************************************
*
P2040000DKN0062Z        012            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 6 - EXPANDED MDS -1ST TRANSIT DEADLINE
R        03
K0109000000021000000003110000000412000000
*
*********************************************************
*
P3040500DKN0063Z          2            0030XF
B   001
RP      SAMPXF01                        XP
BMSG  SAMPLE PROBLEM 6 - EXPANDED MDS -2ND TRANSIT DEADLINE
K0102100002020213023503021000300402100008050200000
*
```

Figure 13-1. Sample Problem 6

Pocket Selection for Sample Problem 6

Figure 13-2, Figure 13-3 on page 13-4, and Figure 13-4 on page 13-5 describe the contents of each of the pockets for this sample problem. In the columns labeled Routing/Transit, the letter *n* can be replaced by any digit.

Figure 13-2. Sample Problem 6: Pocket Selection Criteria - Prime Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	11nn-nnnn	PC = AAAAAA AMT > $1M	On-us High Dollar Debits
02	Kill	11nn-nnnn	PC = AAAA33	On-us Credits
03	Kill	11nn-nnnn	PC = AAAAAA	On-us Debits
04	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 12nn-nnnn 13nn-nnnn 14nn-nnnn 15nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Transit Rehandle
05	Kill	11nn-nnnn	EXT PC not = AA and PC = AAAAAA	On-us Debit Returns

Figure 13-3. Sample Problem 6: Pocket Selection Criteria - First Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	12nn-nnnn	N/A	First Deadline Transit EP=09000000
02	Kill	13nn-nnnn	N/A	First Deadline Transit EP=10000000
03	Kill	14nn-nnnn	N/A	First Deadline Transit EP=11000000
04	Kill	15nn-nnnn	N/A	First Deadline Transit EP=12000000
05	R/H	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 18nn-nnnn 19nn-nnnn 20nn-nnnn 21nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit

Figure 13-4. Sample Problem 6: Pocket Selection Criteria - Second Subsequent Pass

Pocket	Type	Routing/ Transit	Other Requirements for Selection	Description
RJ	Reject	N/A	N/A	System Reject Pocket
01	Kill	18nn-nnnn	N/A	Second Deadline Transit EP=02010002
02	Kill	19nn-nnnn	N/A	Second Deadline Transit EP=02130235
03	Kill	20nn-nnnn	N/A	Second Deadline Transit EP=02100030
04	Kill	21nn-nnnn	N/A	Second Deadline Transit EP=02100008
05	Kill	07nn-nnnn 08nn-nnnn 09nn-nnnn 16nn-nnnn 17nn-nnnn 22nn-nnnn	N/A	Second Deadline Transit EP=02000000

Data for Sample Problem 6

The data for this sample problem consists of one set of tracers with associated data, including block and batch slips. This data also contains deliberate errors to show the balancing features of the system.

The member DKNT06SU in CPCS.V1R11.SAMPLIB contains this input. If you use the simulator for this sample problem, you must include the fields represented by periods as as shown in the records. If you use a physical document processor, do not encode the fields represented by the periods.

Reports for Sample Problem 6

The reports that CPCS produces when you run this sample problem are in CPCS.V1R11.RPTLIB. Print out member SAMPLE6 from this library and compare it with your results for accuracy.

Operating Instructions for Sample Problem 6

1. Start CPCS by submitting the job DKNJRUN from the CPCS.V1R11.CTRL data set.
2. From the CPCS logo screen, log on to the CPCS terminal by entering:

   ```
   SGON xxx
   ```

   where *xxx* is the CPCS operator ID.
3. Designate one terminal as the system supervisor terminal with the command:

   ```
   SUPV ON,SYST
   ```

   or by pressing PF1 and then ENTER.

4. Activate cycle 8 by entering:

   ```
   CYCL 8,A
   ```

   Let the cycle and endorse dates default to the current date. Follow the instructions by pressing **ENTER** until the `READY` prompt appears. CPCS sends a message to the supervisor terminal indicating the change of a cycle's status.

5. Start the MICR task (DKNMICR) on any non-supervisor terminal by entering the command:

   ```
   MICR
   ```

   Open sorter 6 by entering:

   ```
   O 6
   ```

   on the MICR Options screen. Press **ENTER** again to confirm that you wish to open sorter 6.

6. Enter **BEGIN** to display the Begin screen. Specify cycle 8, sort pattern 006, and entry number 0060. Press **ENTER**, then press **ENTER** again to confirm your Begin screen data and start the actual capture.

7. End the prime-pass run when you see the `Intervention Required` message on the MICR status screen. Do this by entering **E** at the prompt. Enter another **E** to verify the End command.

8. After ending the MICR run, press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command. The distribution task (DKNDIST) displays a message on the supervisor terminal indicating the end of the distribution task.

9. Use DKNOLRR to correct the rejects from the prime pass. The command to use is:

   ```
   OLRR 0060,01
   ```

   Three documents in the example contain digit errors. Correct the errors as follows:

   - R/T error: `12*122222` should be `122122222`
   - Amt error: `00000*0002` should be `0000010002`
   - R/T error: `1110006*7` should be `111000627`

10. After you correct the rejects, the OLRR task ends automatically. DKNSCAT then automatically starts. Watch for supervisor terminal messages indicating that DKNSCAT has ended.

11. Start the string merge task (DKNMRGE) for the corrected and concatenated entry with the command:

    ```
    MRGE 1,0060
    ```

    DKNMRGE option 1 merges all data into an M-string by combining the I-string from the prime pass with the concatenated R-string produced by SCAT.

12. DKNMRGE then automatically starts the entry master list (DKNPLST) and the formatted R-string list (DKNRLST) tasks for the corrected entry. DKNMRGE displays messages on the terminal that started the task. DKNPLST and DKNRLST display messages on the supervisor terminal.

13. The Exception Entry Master List report, produced by DKNPLST, identifies one block as being "OUT OF BALANCE". The block that is out-of-balance is the

block slip with the serial number of 49. This out-of-balance condition is caused by the intentional entry of an incorrect amount during OLRR for this block.

14. Start the MICR subsequent pass for the documents sorted to the rehandle pocket (pocket 04) on prime pass for this entry.
    a. Open sorter 6 and proceed to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0060-010
       ```

       That is, the tracer and slip number of the first tracer to sort to rehandle pocket 04. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
15. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
16. Start the MICR subsequent pass for the documents sorted to the first rehandle pocket (pocket 05) on the first subsequent pass for this entry.
    a. Press **ENTER** to return to the MICR Options Screen, then type **B** and press **ENTER** to get back to the Begin screen.
    b. On the Begin screen, enter:

       ```
       ,,0010-011
       ```

       That is, the tracer and slip number of the first tracer to sort to subsequent pass rehandle pocket 05. The two commas ensure the data winds up in the ".ENTRY" field.
    c. Subsequent-pass data appears on the MICR Begin screen. Acknowledge that the data is correct by pressing **ENTER**.
17. End the subsequent-pass run by entering **E** when you see the `Intervention Required` message on the MICR Status screen. Enter another **E** to verify the End command.
18. Press **ENTER** to return to the MICR Options screen, and end the MICR session with the Close command.
19. The distribution task started automatically when you ended each entry. Because the entries were subsequent pass, each DKNDIST also automatically started a DKNSLST to produce a Subsequent Pass Master List.

    The Subsequent Pass Master Lists identify, on an item-count and dollar-amount basis, any out-of-balance condition between the prime-pass and subsequent-pass runs. For these entries, there are no exception conditions.
20. To demonstrate how the Mass Data Set expansion affects reports, list the M-string. The command to use is:

    ```
    LIST 0060-1-00-00-00-00-M-000
    ```

    When prompted for a DKNLIST option, select option 1 to list the entire string.
21. To demonstrate how the Mass Data Set expansion affects reports, run the HEXL command on the M-string. The command to use is:

    ```
    HEXL 0060-1-00-00-00-00-M-000
    ```

When prompted for a DKNHEXL option, select option 3 to list the entire string.

22. Start the Kill List task (DKNKILL) for the first deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT061** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT061 contains the endpoints for each first deadline kill pocket.

23. Start the Kill List task (DKNKILL) for the second deadline items with the command:

    ```
    KILL 8
    ```

    where 8 is the current cycle number. Select the Endpoint Table option, accept the default value for reprint (press **ENTER**), enter **EPT002** when prompted for the table name, and press **ENTER** on the Cash Letter Override screen. EPT002 contains the endpoints for each second deadline kill pocket.

24. Start the Cash Letter Summary (DKNCLSM) task with the command:

    ```
    CLSM ALL,8
    ```

    where 8 is the current cycle number.

    Select the Endpoint Table option and use table **EPT063** when prompted for the table name. This table contains all the endpoints from all deadlines, both first and second. When DKNCLSM prompts you to respond to whether this is a duplicate letter, press **ENTER** to select No.

25. Run the Master Create task (DKNMCRE) for the current cycle (cycle 8) with the command:

    ```
    MCRE ALL,8
    ```

    This task transfers outgoing items, which do not require further processing, to a master file and deletes these items from the system. Strings transferred and deleted include killed and on-us D-strings, subsequent pass reject D-strings, and R-strings.

26. Run the Input Create task (DKNICRE) for the current cycle (cycle 8) with the command:

    ```
    ICRE ALL,8
    ```

    This task transfers all M-strings for the particular cycle to the input create file.

27. Run the Cycle task to deactivate the current cycle (cycle 8) with the command:

    ```
    CYCL 8,D
    ```

28. Run the List Directory task (DKNLDIR) for the current cycle (cycle 8) with the command:

    ```
    LDIR 8,D
    ```

    This task lists all string entries for the cycle and deletes all strings that do not need further processing by the system. After you run this task, the only string left on the mass data set is the M-string for the prime pass.

29. Run the End Cycle task (DKNECYC) for the current cycle (cycle 8) with the command:

    ```
    ECYC 8
    ```

This task ensures that there are no active strings present for this cycle, other than the prime-pass M-strings.

It deletes these M-strings and removes all pass-to-pass control records for the cycle. It also automatically starts the End Cycle Two task (DKNECY2), which transfers all kill-bundle records from the kill-bundle data set to a kill-bundle summary file.

30. Go to the supervisor terminal and end CPCS with the command:

    ```
    STOP
    ```

Appendix A. Macro and Member Lists

The macros identified in this appendix are provided as programming interfaces for customers by CPCS.

Warning: Do not use as programming interfaces any CPCS macros other than those identified in this appendix.

The following are provided as general-use programming interfaces for CPCS.

ALLOC	OPACK	VNODE
APCB	OUNPK	VSCANREQ
CALLSUB	PROEND	XRETURN
CALL389X	PROLOG	
CCSDEF	PROLOGX	
CPCSOPTN	PROLOG2	
CPCSRDR	RGENDEF	
CSCB	RGENDSCT	
DIAGTBL	SECRPARM	
DIAGWRKD	SECRWKAR	
DKNAMODE	SETDEVX	
DKNBSCY	STATCHRO	
DKNDSCT	STATCON	
DKNEQU	STATDECV	
DKNGREGS	STATHEXC	
DKNIGPRM	STATINCH	
DKNMSCT	STATMENT	
DKNROPTS	STATMSG	
DKNRSVCS	STATTBL	
DPCB	STATVAR	
DPCBD	STATWRKD	
DSAT	SUBRTN	
FFLDL	SUBSAVE	
FPACK	TBLEND	
FUNPK	TBLSTART	
MDCDSCT	TBLSTRT2	
MDX	TBLSTRT4	
MSGSUBSP	VDSECT	
OFSRRWA	VMSG	

Figure A-1. CPCS Assembler Macros

DKNCATCB	DKNCRURB
DKNCLDS	DKNCRZA
DKNCPARM	DKNCRZB
DKNCRAB	DKNCRZX
DKNCRBC	DKNICRE1
DKNCRBCF	DKNLIST1
DKNCRBIF	DKNLIST2
DKNCRCHK	DKNLIST3
DKNCRDX	DKNLIST4
DKNCRED	DKNLIST5
DKNCRKB	DKNLIST6
DKNCRLOC	DKNPLST1
DKNCRMAL	DKNRPP1C
DKNCRMF	DKNRPP1X
DKNCRNA	DKNRPP2C
DKNCRPKT	DKNSLST1
DKNCRSK	DKNSLST2
DKNCRSP	DKNSLST3
DKNCRST	EXINFOC
DKNCRTG	MRGE001
DKNCRTGU	MRGE002
DKNCRTL	MRGE003
DKNCRTRK	

Figure A-2. CPCS COBOL Copybooks

ATHDLDCT	DXDSCT	OFSRMAIN
ATP1BDCT	EXINFO	ORSC
ATRLLDCT	IGENDSCT	PIMAGE
BCFDSCT	IMSTDSCT	RCVUNTBL
BEGNDSCT	IMWKRDX	REC
BUFRDSCX	INDXDSCT	SDEDSCT
CDWA	ISEQDSCT	SGCBADCT
CPCSCAN	LBCBDSCT	SGCBDSCT
CPCSCTL	LCCBDSCT	SQEDSCT
CPCSNAME	LMCBDSCT	SRCOMM00
CPCSNUM	LNTBDSCT	SRCOMM01
CYCADSCT	LOGRSCB	SRMSG000
CYCLDSCT	LPCBDSCT	SSBDSCT
CYCLTABL	MDSPDSCT	STATARAY
DEBUFF	MDXCOPY	STATGBLS
DIAGGBLS	MDXRFIX	TGARSTXT
DICOMM00	MDXROPEN	TGDSCT
DICOMM01	MDXRRPTB	TGPARMS
DIEXT	MFDSCT	TGUTIL
DIFLGSCT	MICREQU	TLDSCT
DIMSG000	MICRTIOX	ZADSCT
DISEQSCT	MLBXDSCT	ZBDSCT
DISTDSCT	MLPDSCT	ZEDSCT
DISTMF01	MNHLPD01	ZXDSCT
DISTMF02	MNHLPM01	
DISTMF03	MNOPTD01	
DISTMF04	MNOPTO01	
DISTMF05	MNOPTO02	
DIVDSCT	MNPNLD04	
DIVWDSCT	MNPNLD05	
DKNAPCB	MNPNLD06	
DKNAPCB1	MNPNLD07	
DKNAPTCB	MNPNLD08	
DKNDENT	MNPNLD09	
DKNMBUFF	MNPNLD10	
DKNMISDS	MNPNLN04	
DKNPARM	MNPNLN05	
DKNPRINT	MNPNLN06	
DKNRPP1A	MNPNLN07	
DKNRPP2A	MNPNLN08	

Figure A-3 (Part 1 of 2). CPCS Assembler Copy Members

DKNSPOOL	MNPNLN09
DKNTENT	MNPNLN10
DKNTLBUF	MNPNLN11
DKNTLT	MNPNLN11
DKNTYPE	MNPNLN12
DKNTYPE2	MNPNLN13
DKNTYPE4	MNPNLN14
DKNVCOMS	MNPNLN15
DKNVEXTS	MNPNLN16
DKNVINIT	MNPNLN17
DKNVIOER	MNPNLN18
DKNVMAIN	MNPNLN19
DKNVMSGS	MNPNLN20
DKNVOPTS	MNPNLN21
DKNVPFKS	MRDBDSCX
DKNVSCRL	MSGSUB
DKNVUTIL	ODWK

Figure A-3 (Part 2 of 2). CPCS Assembler Copy Members

Glossary

This glossary defines important terms and abbreviations used in this manual. If you do not find the term you are looking for, refer to the Index or to the *IBM Dictionary of Computing*, SC20-1699.

A

ABA. See *American Bankers Association*.

ABA number. (1) A numbering system devised by the ABA to provide exact identification of financial institutions. The code structure also identifies the Federal Reserve Bank and branch. (2) The MICR-inscribed field on a document, containing the financial institution identification number.

account number field. A MICR-encoded field, on a check or a deposit slip, that represents the account number of the item. The account number field is MICR field 3.

active task status. This indicates that this task is currently processing the string associated with this UOW. See also *task status*.

active UOW status. This indicates that a task in the task list is currently processing the string associated with this UOW. See also *UOW status*.

adjustment. A change or a description of a change that has been made to reflect a detected error in work that has been processed.

advice. A letter that is sent to a financial institution or customer from whom checks have been received, advising that errors have been detected in the checks or in the listing that accompanied the checks.

American Bankers Association (ABA). Among the functions of this group is the specification of banking industry standards for check-handling documents and procedures.

amount field. A MICR-encoded field on an item that represents the amount of that item. The amount field is MICR field 1.

Application Library Services. See *ImagePlus HPTS Application Library Services*.

application program task control block (APTCB). A CPCS area created by the applications task (DKNATASK) for every active subtask in the system. This area contains operating system control blocks that are related to the subtask; it also contains addresses and constants used by the CPCS executive programs.

APTCB. See *application program task control block*.

AST. Assist document.

automatic restart. The process of restarting (continuing) an interrupted entry without having to find and rebatch any item.

auxiliary on-us field. See *serial number field*.

B

balancing. The act of bringing two sets of related figures into agreement (for example, reconciling accumulated-detail totals and input-control totals).

batch. The lowest required level that has dollar control established by a control document.

batch number. The number that uniquely identifies a specific batch of documents.

batch slip. A level of control for balancing items. See also *batch*.

block. (1) A prime-pass control level consisting of one or more batches. In CPCS, this control level is used to total multiple batches. A block can also represent work from a specific source. (2) A data-processing term used to refer to a series of logical records stored contiguously on external storage devices. (3) To insert control documents in preparation for a prime-pass sorter run. See *data preparation*.

block slip. A level of control for balancing batches. See also *block*.

buffer. A main storage area used as a data-transfer area for physical records being read or written.

bypass task status. This indicates that this task should not process for the string associated with this UOW. See also *task status*.

C

cash letter. The group of items to be delivered to an endpoint. Grouping of the items is usually by kill bundle.

cash-letter detail. A listing of all kill bundles. See also *kill list*.

cash-letter summary. A listing that summarizes all of the kill bundles in a cash letter by giving monetary and item controls for each kill list.

check. A draft drawn on a financial institution and payable on demand any time on or after the date indicated.

Check Image Management System (CIMS). A program in ImagePlus HPTS Application Library Services that stores, retrieves, and manages document images.

CIMS. See *Check Image Management System*.

clearing house. An organization, established by financial institutions in the same locality, through which checks and other instruments are exchanged and net balances settled.

code-line data matching. A method by which a computer system controls items on a detail level by comparing the internal data records from a previous pass with data that it reads on the current pass. Code-line data matching occurs on subsequent operations.

code-line data record. See *data record*.

cold start. An initiation of the CPCS region that causes the deletion of the previous contents of the mass data set and the control data sets.

complete task status. This indicates that this task processed successfully for this UOW. See also *task status*.

complete UOW status. This indicates that all tasks in the task list processed successfully or had a bypass status. See also *UOW status*.

control slip. A MICR-encoded document that contains control information, including the amount of the items that the document controls, the source of the items, and a code that describes the level of the control.

control total. The total dollar value or item count for a group of documents.

copy library. A library that contains statements to be modified by the user, accessed by the assembler instruction copy, and inserted into some of the CPCS programs.

correspondent financial institution. A financial institution that carries a deposit balance for, or engages in an exchange of services with, another financial institution.

credit. The opposite of a debit. In normal check collection terminology, deposit slips are credits.

cursor. A small horizontal line on a computer screen that indicates the position to which the next character is transmitted from either the keyboard or the CPU.

cutoff. The financial institution's designated point for balancing or releasing work before processing continues. Also, the designated time after which the financial institution cannot accept work for processing.

cut slip. A control document used to separate and identify one kill bundle from another.

cycle. (1) A group of work or an identification of a group of work processed completely as a single entity. (2) A convenient grouping of work. A cycle normally contains a variable number of entries.

D

data preparation. Preparation of documents for processing by a high-speed check-processing system.

data record. The electronic representation of the MICR code line captured from a check, deposit, debit, credit, or control document. The electronic representation can include additional data to help identify the record.

data set. A single collection of data that can be stored on cards, a tape, or one or more disks (for example, a kill-bundle data set).

debit. A transaction that increases an asset or decreases a liability. In normal check-collection terminology, a check is considered a debit.

deferred printing. The method by which data is processed, transferred to a storage device, and later printed (as opposed to printing during the processing of data).

deleted UOW status. This indicates that the string associated with this UOW is deleted. No more processing can be done for this UOW. See also *UOW status*.

deposit slip. A document that details a deposit. The total of the deposit is MICR encoded on the deposit slip. A deposit is considered a credit.

distributed string (D-string). The distribution task reads I-strings that the MICR task created and produces D-strings. Each D-string contains the records that correspond to all of the documents in a single pocket of the document processor.

divider slip. A control document that is used to separate kill bundles during machine sorting of checks. It can also be used to support the resynchronization of code-line data matching during subsequent-pass processing.

document processor. A device that can read MICR-encoded digits and control characters from documents and sort the documents into multiple pockets.

document processor station. A workstation consisting of a document processor and a terminal for operator communication.

D-string. See *distributed string*.

E

eligible task status. This indicates that this task is waiting for processing. Other tasks must process before this task. See also *task status*.

enclosed and not listed. A condition that exists when an item is in a batch of checks but is not listed on the incoming kill list or inscriber tape.

encode. To imprint a MICR field on a check.

encoder. A machine that encodes.

endorsement. The signature of the endorser; the stamp of a financial institution or company.

endorser. (1) A person or financial institution, other than the maker, who presents a check for payment. (2) A device that stamps an endorsement.

endpoint. The destination of a check.

entry. A variable whole number of blocks that are processed as a single group of work.

entry number. The number of the first tracer group within an entry.

EPC. See *extended process control field*.

error description. The detailed description of an error created, detected, and corrected by the processing financial institution.

exception printing. Printing of only the data that requires action external to a computer.

exchange charge. A charge made by the drawee financial institution for its services in paying checks and other instruments presented to it. The Federal Reserve Act forbids drawee financial institutions to make such charges against Federal Reserve Banks on checks in process of collection. The purpose is to assure that checkbook money will be payable throughout the country at its face value.

extended process control (EPC) field. An optional MICR-encoded item field that indicates special handling (such as return or truncation).

F

fine-sort. (1) The sorting of items into account number order for filing. (2) The sorting of items for a single account into serial-number order as a customer service.

flip-flop. An event that occurs when the volume to which you are writing a file becomes full. The writing continues on a new volume and the full volume is backed up.

float. The portion of a financial institution's total deposits, or of a depositor's account, that represents items (for example, checks and coupons) in the process of collection.

flow code. A 3-digit number (mnemonic) that represents an ordered list of tasks.

flow control. The pairing of a CPCS string with a task list through the specification of sort type, pass pocket history, string type, and flow code.

full-page printing. A method of page formatting in which items are listed in as many columns as can be contained on the page (for example, the first 50 items in column 1, the second 50 in column 2, and so on).

functional unit of work. This unit of work corresponds to a CPCS string.

funds availability. The portion of the financial institution's total deposits or of a depositor's account that represents items (for example, checks and coupons) that have been collected and are now available. This includes cash deposited and checks drawn on the depositor's financial institution.

G

generated total. The total dollar value or item count of checks that are processed by the computer.

H

held task status. This indicates that this task should be the next task to process, but a condition external to CPCS must complete first. See also *task status*.

held UOW status. This indicates that the task to process next for this UOW has a held status. See also *UOW status*.

High Performance Transaction System (HPTS). See *ImagePlus High Performance Transaction System*.

high-speed reject re-entry. The re-entering into the document processor of reconditioned documents that have previously been sorted to the system reject pocket (1-1).

holdover. Items that were not processed in time to meet their deadline.

HPTS. High Performance Transaction System.

I

ImagePlus High Performance Transaction System (HPTS). An IBM system that adds image processing capabilities to document processing.

ImagePlus HPTS Application Library Services. An IBM licensed program that supplies the HPTS system with services such as communication, data-storage management, recognition facilities, data compression, data reconstruction, and device support. The program consists of Image Host Application services, Image Processor Recognition Services, and Image Workstation Application Services.

inclearings. Checks drawn on your financial institution that are sent in for collection by another financial institution, the Federal Reserve Bank, or a clearing house.

incoming sequence number. A number that defines the incoming sequence of an item within the input stream. This unique number is associated with the item throughout the whole cycle of computer processing.

informational unit of work. This usually represents a physical group of checks that CPCS has not yet processed. It represents work to be done. One or more informational units of work are ultimately associated with a functional unit of work when CPCS captures the physical check documents.

input string (I-string). This is a string of documents created by the MICR task. On each document processor run, an I-string is created. The string includes every document read by the document processor, including control documents and rejected documents. Related information, such as the pocket selected, is also stored in each record. The string also includes internally generated control records.

inscriber. A machine that encodes.

I-string. See *input string*.

item. A check, deposit slip, or other machine-readable document.

item number. A number that is associated uniquely with a document throughout the processing cycle.

J

jam. A condition that exists when items form a blockage anywhere in the transport mechanism of a document processor.

joggler. A device that straightens and aligns items before high-speed sorting, principally to line up the lower edge and right side of a group of documents. This device is an integral component of some document processors.

K

kill. To process items to a point where no further distribution is required.

kill bundle. A group of killed items, indicated by divider slips. With concurrent kill, this group can span strings.

kill list. A document that accompanies a kill bundle, listing detail and controls for the items.

kill pass. A pass on which items are distributed to their endpoint pockets.

kill pocket. A document-processor pocket assigned to killed items.

L

legal tender. Any money that must, by law, be accepted in payment of debts. Checkbook money is not legal tender.

listed and not enclosed. A condition that exists when an item is listed on an incoming kill list or inscriber tape but is not enclosed in the kill bundle.

low-speed transit. The manual sorting and processing of checks.

M

magnetic ink character recognition (MICR). The reading of magnetically encoded data on the 5/8" clear band that runs along the bottom of a check. The MICR system uses 10 specially coded digits and four special symbols.

maker. The person on whose account a check is being drawn.

Management Information System (MIS). A DB2 system that maintains data on overall check processing. This is a subcomponent of ImagePlus HPTS Application Library Services (IALS).

manual restart. The process of physically finding and rebatching, before resuming an interrupted entry, the items to be recaptured.

mass data set (MDS). A file that contains records of all active document strings. This file consists of two direct access data sets: a directory index and a data record set.

master list. A list of all items that are read during a computer pass.

MDS. See *mass data set*.

merged string (M-string). The M-string, produced by DKNMRGE, represents the merging of images from the prime-pass I-string with corrected reject data. Reports that result from the M-string let you reconcile and balance input to ensure that all items were captured.

MICR. See *magnetic ink character recognition*.

MICR field 1. See *amount field*.

MICR field 2. See *process control field*.

MICR field 3. See *account number field*.

MICR field 4. See *optional field 1*.

MICR field 5. See *ABA number*.

MICR field 6. See *extended process control field*.

MICR field 7. See *serial number field*.

microfilm number. The assigned item number that is also captured on microfilm.

MIS. See *Management Information System*.

misread. A condition that occurs when a document processor interprets a MICR character as a good character other than that which was actually encoded on the document.

missort. An item that is found in a pocket other than the pocket to which it was sorted.

M-string. See *merged string*.

O

online fine sort. A computer-controlled sorting of on-us checks by either or both the account number and the serial number sequence for filing. This process can use image-processing techniques.

online reject re-entry. Manual entry or correction of MICR data through a display terminal.

on-us checks. Checks that are drawn on the financial institution that is processing them.

optional field 1. An optional, MICR-encoded field used by some financial institutions for check truncation. It can also be used for other internal purposes.

optional field 2. See *extended process control field*.

outgoing sequence number. A sequence number or unique identification assigned to each item, identifying the kill bundle in which the item left the financial institution.

P

pass. A single reading and sorting of a group of checks and control documents on a document processor.

pass-to-pass control. A process that maintains dollar and item control of a group of MICR documents on subsequent passes, when control has been established on the previous pass.

path. The path of a functional unit of work is the ordered list of tasks processed for the associated CPCS string. See also *flow code* and *flow control*.

pending request queue. A first-in-first-out System Manager queue through which CPCS applications interface to the System Manager, in sequence, to perform UOW creations, deletions, inquiries, and updates.

piggyback item. An item that was missing from its assigned pocket in a sorter and sorted “free” to an unidentified pocket, as when one document attaches itself to or overlaps another during processing.

pocket 1-1. See *system reject pocket*.

prime pass. The first pass of an entry on a document processor.

printing after the fact. See *deferred printing*.

process control field. (1) A MICR-encoded field on a document, usually representing the type of document. The process control field is MICR field 2.
(2) Transaction code.

proof. Receives checks that come from tellers, mail and night depository, and internal departments of the financial institution. Proof proves and inscribes the dollar amount in MICR.

proof of deposit. The act of totaling items at the deposit level and ensuring that the total of the credits equals the total of the debits.

Q

queued task status. This indicates that this task is ready and waiting to process. See also *task status*.

queued UOW status. This indicates that no task is currently active for this UOW; however, one task has a queued status. See also *UOW status*.

R

RACF. See *Resource Access Control Facility*.

RBA. See *relative block address*.

reconcile. To find and correct the cause of a difference between two sets of totals.

reconciliation. See *balancing*.

rehandle pocket. A document processor pocket that receives items for multiple endpoints. Items directed to rehandle pockets are processed again on a later pass.

reject. A MICR-encoded document that cannot be read in its entirety by a document processor or that fails certain editing checks. This document is directed to a special pocket called a reject pocket.

reject string (R-string). Strings are created by the online reject re-entry task. Each R-string represents checks that have been re-entered online. R-strings are input to the DKNMRGE task.

relationship. Shows the parent/child hierarchy of units of work.

relative block address (RBA). In CPCS, the calculated location of a specific record.

repass. See *rehandle pocket*.

rerun. A group of items that are sorted into a pocket on one pass and later brought into a document processor for further sorting.

Resource Access Control Facility (RACF). An MVS security subsystem that determines the validity of each operator’s ID password and that controls operator access to application tasks and transactions.

restart. An initiation of the CPCS system after a system failure. A restart is generally used to start the system (after an abnormal end of a task) to cause the executive routines to re-establish the system to the status that existed before the failures.

restart buffer. An XP reader/sorter area where records are stored during online operations until they are sent to the host. The buffer is accessed during automatic restart.

return item. A check that is not honored by the maker’s financial institution and that is returned to the depositor’s financial institution.

routing number. A MICR-encoded check field that represents the financial institution on which the check is drawn. The routing transit field is field 5.

routing transit field. See *ABA number*.

R-string. See *reject string*.

S

scroll. The ability to use the DKNSCRL application to page through or look at the scroll data set. This data set includes supervisor terminal messages and DKNATASK log messages.

separator. See *divider slip*.

sequence number. A number, assigned to a document, that uniquely identifies its position in a group of incoming or outgoing work.

serial number field. A MICR-encoded check field that represents the serial number of that check. Synonymous with *auxiliary on-us field*. The serial number is MICR field 7.

settlement. The act of bringing sets of related figures from two financial institutions into agreement. Adjustments are made to offset the differences.

SMOF. System Manager Online Functions.

sort pattern. A table used by the sort routine to determine the pocket to which a check is to be directed.

sort program. A routine that performs all processing required to select a document to a pocket.

spool data set. A data set used to store printed output lines. Each spool (Simultaneous Peripheral Operations On-Line) data set is written by a CPCS application task and is read by the CPCS output writer as it is being printed.

SSB. See *string status block*.

statistics. The processing of Unit of Work data through a statistical program such as the ImagePlus Application Library Services MIS system. This term can also refer to the processing of Unit of Work data through a user-written statistical program.

SSM. See *string segment map*.

string. The data records representing a group of items entered through a physical or simulated document processor or through OLRR. Can be an I-string, a D-string, or an M-string. See related definitions for details.

string segment map (SSM). One of three types of segment maps in CPCS. Each string in the system is associated with a string segment map. Each bit in a map represents a segment of direct access storage. The bit settings for the string segment map are (1) 0=segment available or (2) 1=segment allocated.

string status block (SSB). This CPCS control block is maintained by the MDS programs for every open string.

subsequent pass. A pass on which previously sorted items are resorted for further distribution.

supervisor. (1) An MVS term used to refer to the system nucleus in internal storage. (2) A person responsible for operation of a financial institution area.

supervisory terminal. A special terminal or operating mode used in CPCS.

suspended task status. This indicates that this task processed, but it did not complete successfully. See also *task status*.

suspended UOW status. This indicates that the last task that processed for this UOW did not complete successfully. See also *UOW status*.

System Manager. A subsystem of CPCS that directs and controls the operations of the IBM 3890/XP Series document processors.

System Manager Online Functions (SMOF). A set of application-level tasks that monitor and modify the queues and databases of System Manager.

system reject pocket. The first physical pocket on the document processor. It is used by CPCS to hold machine and user-selected rejects.

T

tab key. A keyboard function key. The tab key causes the cursor to position to the next colon on the screen or to the top of the screen.

task. A CPCS application or function. A task name must be DKNMICR or it must be in the CPCS BLDL list.

task list. The ordered list of tasks to be performed for a unit of work. It is determined by selecting the flow code for a given flow control record.

task status. A representation of what will happen, what is happening, or what happened during processing of this unit of work. Can be (1) active, (2) bypass, (3) complete, (4) eligible, (5) held, (6) queued, or (7) suspended. See related definitions for details.

total system. A system in which the computer is used for all phases of an operation.

tracer. A check-processing document used to provide pass-to-pass control.

tracer group. An arbitrary grouping of items for control purposes.

transit. The sorting of checks to external destinations.

U

unit of work (UOW). A logical entity that the System Manager uses to track a piece of work through CPCS. It can be informational or functional. See also *informational unit of work* and *functional unit of work*.

UOW. See *unit of work*.

UOW status. This status represents the state of a unit of work and its associated string. Can be (1) active, (2) queued, (3) suspended, (4) complete, (5) deleted, or (6) held. See definitions for details.

W

warm start. An initiation of the CPCS system, causing the contents of the MDS and the control data sets to be retained. A warm start is generally used for restarting CPCS after a normal ending.

work. Any document or group of documents that CPCS processes.

work flow. An ordered list of tasks for a specific CPCS string. Each CPCS string must have a work flow.

Z

zero-balancing. The procedure that ensures that generated totals for a group of items plus any documented errors minus the control total equals zero.

Bibliography

The publications in this bibliography contain information related to CPCS.

ACF/VTAM Publications

The following publications are related to the ACF/VTAM product:

IBM ACF/VTAM Version 3 Programming, SC23-0115

IBM ACF/VTAM Planning and Installation Reference, SC27-0584

IBM ACF/VTAM, 5735-RC2, 5746-RC3, Program Operator's Guide, SC38-0257.

Document Processor Support Publications

The following publications are related to document processor support:

IBM 3890/XP Series Document Processor General Information, GA34-2012

IBM 3890/XP Series Programming Guide, GC31-2662

IBM 3890/XP Series SPXServ Reference, GC31-2704

IBM 3890/XP MVS Support and IBM 3890/XP VSE Support Program Reference, SC31-2654
(TNL SN31-8160)

IBM 3890/XP Series Stacker Control Instructions Reference, SC31-2703.

OS/390 Publications (Version 2, Release 8)

The following publications are related to OS/390:

IBM OS/390 MVS Initialization and Tuning Reference, GC28-1752

IBM OS/390 MVS JCL User's Guide, GC28-1758

IBM OS/390 MVS System Codes, GC28-1780

IBM OS/390 MVS Installation Exits, SC28-1753

IBM OS/390 MVS System Messages Volume 1, GC28-1784

IBM OS/390 MVS System Messages Volume 2, GC28-1785

IBM OS/390 MVS System Messages Volume 3, GC28-1786

IBM OS/390 MVS System Messages Volume 4, GC28-1787

IBM OS/390 MVS System Messages Volume 5, GC28-1788

IBM OS/390 MVS Programming: Assembler Services Guide, GC28-1762

IBM OS/390 MVS Programming: Assembler Services Reference, GC28-1910

MVS Publications

The following publications are related to MVS:

IBM OS/VS2 MVS System Programming Library Job Management, GC28-1303

IBM MVS/ESA Initialization and Tuning Reference, GC28-1635

MVS/ESA JCL User's Guide, GC28-1653

IBM MVS/ESA System Messages Volume 1, GC28-1812

IBM MVS/ESA System Messages Volume 2, GC28-1813

IBM MVS/ESA System Codes, GC28-1815

IBM MVS/ESA Data Administration: Utilities, Version 3.1, SC26-4516.

RACF Publications

The following publications are related to RACF:

IBM Resource Access Control Facility Command Language Reference, SC28-0733

IBM Resource Access Control Facility Master Index, GC28-1035

IBM Resource Access Control Facility Security Administrator's Guide, SC28-1340

IBM System Programming Library: Resource Access Control Facility, SC28-1343

IBM Resource Access Control Facility Macros and Interfaces, SN28-1539.

VTAM Publications

The following publications are related to Version 3 Release 2 of the VTAM product:

IBM SNA Transaction Programmer's Reference Manual for Logical Unit Type 6.2, GC30-3084

IBM VTAM Programming for LU 6.2, SC30-3400. Trademark of IBM

Note: Several of these manuals also contain information about VTAM Version 3 Release 1.2 for VM and VSE, Version 3 Release 1.1 for MVS and VM, and Version 3 Release 1 for VSE.

Other IBM Publications

The following publications provide information about topics related to CPCS:

IBM 3270 Information Display System 3274 Control Unit Description and Programmer's Guide, GA23-0061

IBM 3270 Information Display System Customizing Guide Supplement for 3274 Control Unit, GC11-6045

IBM System/370, 30xx, 4300, and 9370 Processors: Bibliography of Industry Systems and Application Programs, GC20-0370

IBM Data Facilities/Data Set Services User's Guide and Reference, SC26-4125.

Index

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

R

S

T

U

V

W

X

Communicating Your Comments to IBM

Check Processing Control System
Installation Guide
Release 11

Publication No. GA34-2178-08

If you especially like or dislike anything about this book, please use one of the methods listed below to send your comments to IBM. Whichever method you choose, make sure you send your name, address, and telephone number if you would like a reply.

Feel free to comment on specific errors or omissions, accuracy, organization, subject matter, or completeness of this book. However, the comments you send should pertain to only the information in this manual and the way in which the information is presented. To request additional publications, or to ask questions or make comments about the functions of IBM products or systems, you should talk to your IBM representative or to your IBM authorized remarketer.

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

If you are mailing a readers' comment form (RCF) from a country other than the United States, you can give the RCF to the local IBM branch office or IBM representative for postage-paid mailing.

- If you prefer to send comments by mail, use the RCF at the back of this book.
- If you prefer to send comments electronically, use this network ID:

  Internet: www.ibm.com/Products/CPCS

Make sure to include the following in your note:

- Title and publication number of this book
- Page number or topic to which your comment applies.

Readers' Comments — We'd Like to Hear from You

Check Processing Control System
Installation Guide
Release 11

Publication No. GA34-2178-08

Overall, how satisfied are you with the information in this book?

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Overall satisfaction	□	□	□	□	□

How satisfied are you that the information in this book is:

	Very Satisfied	Satisfied	Neutral	Dissatisfied	Very Dissatisfied
Accurate	□	□	□	□	□
Complete	□	□	□	□	□
Easy to find	□	□	□	□	□
Easy to understand	□	□	□	□	□
Well organized	□	□	□	□	□
Applicable to your tasks	□	□	□	□	□

Please tell us how we can improve this book:

Thank you for your responses. May we contact you? □ Yes □ No

When you send comments to IBM, you grant IBM a nonexclusive right to use or distribute your comments in any way it believes appropriate without incurring any obligation to you.

Name _____ Address _____

Company or Organization _____

Phone No. _____

Readers' Comments — We'd Like to Hear from You
GA34-2178-08

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 MAIL PERMIT NO. 40 ARMONK, NEW YORK

POSTAGE WILL BE PAID BY ADDRESSEE

International Business Machines Corporation
Payment Solutions
Department 58G, MG96/204
8501 IBM Drive
Charlotte NC 28262-8563

Fold and Tape | **Please do not staple** | Fold and Tape

Cut or Fold
Along Line

Program Number: 5734-F11

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

GA34-2178-08