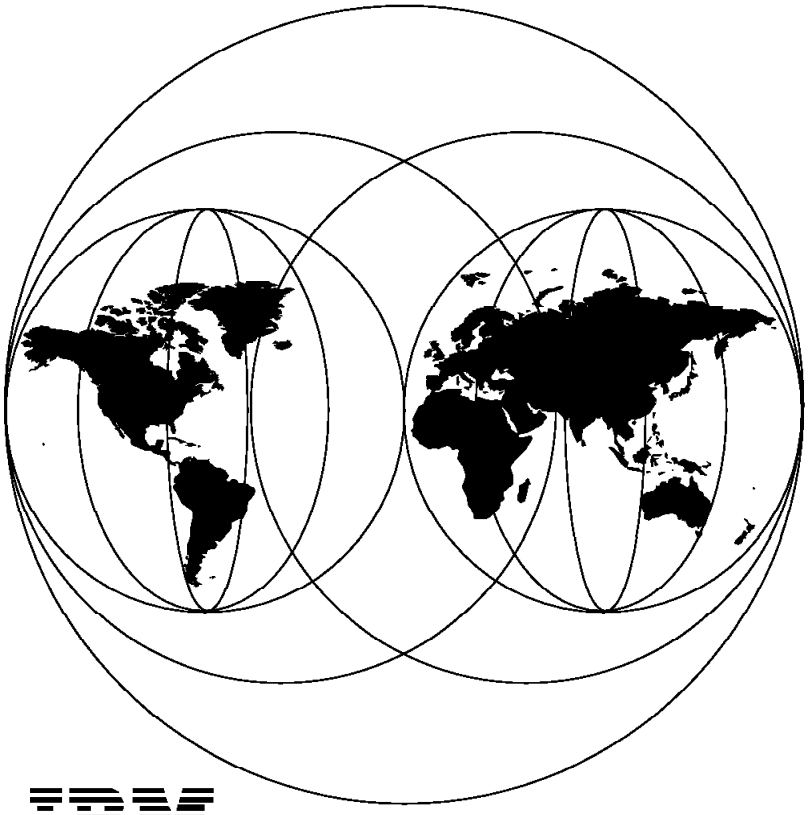
SG24-4932-00

Preparing Your VSE System for the Year 2000

December 1996



IBM

International Technical Support Organization
Boeblingen Center

International Technical Support Organization

SG24-4932-00

Preparing Your VSE System for the Year 2000

December 1996

Take Note!

Before using this information and the product it supports, be sure to read the general information in Appendix C, “Special Notices” on page 163.

First Edition (December 1996)

This edition applies to Version 1 Release 4 of the IBM Language Environment for VSE/ESA (LE/VSE), program number 5686-094, for use with the VSE/ESA Operating System Version 2, program number 5690-VSE.

Comments may be addressed to:
IBM Corporation, International Technical Support Organization
Dept. 3222 Building 71032-02
Postfach 1380
71032 Böblingen, Germany

When you send information to IBM, you grant IBM a non-exclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

Contents

Figures

Tables

Preface

The turn of the century presents the user with the problem of converting 2-digit date formats to 4 digits. This redbook outlines how to overcome this obstacle with short-term and long-term solutions (together with LE/VSE) pertaining to programs written in high level languages. Numerous examples are given as well as a comprehensive overview of the COBOL and CICS Command Level Conversion Aid for VSE.

This redbook is intended for customer and IBM technical personnel with the responsibility for planning and executing the transition of the VSE system environment into Year 2000.

How This Redbook Is Organized

The redbook is organized as follows:

- Chapter 1, “Understanding the Year2000 Problem”

  This chapter explains the Year2000 problem, lists planning aspects, and gives a short description of possible solutions.

- Chapter 2, “The Long-Term Solution”

  This chapter describes the long-term solution using 4-digit years.

- Chapter 3, “The Short-Term Solution”

  This chapter describes the short-term solution using the windowing technique or special encoding.

- Chapter 4, “VSE/ESA Components and Functions Year2000 Support”

  This chapter shows what VSE/ESA and selected optional products are providing to be Year2000-enabled.

- Chapter 5, “Languages and the Year 2000”

  This chapter gives you examples of how to change your applications to make use of 4-digit years. COBOL, PL/I, C, and RPG II languages are covered.

- Chapter 6, “Migration Considerations”

  This chapter provides information how to perform the migration to a Year2000-ready system environment and describes also test possibilities.

- Chapter 7, “COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)”

  This chapter gives you an example of using CCCA/VSE, a tool to convert your ′old′ COBOL programs to COBOL/VSE.

- Appendix A, “Product Information for VSE/ESA 2.1.x and 1.4.x”

  This appendix provides the status of VSE/ESA base and optional products regarding their Year2000 support.

- Appendix B, “Partition Communication Region (COMREG)”

  This appendix describes the layout of the VSE/ESA Communication Region.

The Team that Wrote this Redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization Böblingen Center.

Annegret Ackel from the International Technical Support Organization Böblingen Center, was the project leader.

Keith Miller from ISSC Australia.

Jan Vanvaeck from IBM Belgium.

Comments Welcome

We want our redbooks to be as helpful as possible. Should you have any comments about this or other redbooks, please send us a note at the following address:

redbook@vnet.ibm.com

Your comments are important to us!

Chapter 1. Understanding the Year2000 Problem

1.1 The Year2000 Problem

The scope of the Year2000 problem spans the entire Information Technology industry.

It affects all computer platforms, is not unique to IBM processors, and it does affect customers with the VSE operating system.

Historically many computer programs made use of a two-digit date format (95 rather than 1995), and this causes programs (both system and application) that perform arithmetic operations, comparisons or sorting of date fields to give incorrect results when working with dates outside the range of 1900-1999.

Although referred to as the Year2000 problem, it is really a 2-digit year problem.

1.2 Planning Aspects

You need to plan for and address the date problem well before January 2000, as your computing environment will not function as expected because the systems and/or application programs will not process dates later than 31 December 1999 properly.

Planning for, and implementing a strategy to overcome the cause of the Year2000 problem is not a trivial project.

It is clear that to provide solutions to the Year2000 problem changes must be made both to the operating system environment, and also to applications and application programs. The problem cannot be ignored.

Plans must be put into place to ensure both the operating system itself, and applications (including application programs) address the Year2000 problem.

To address the date problem within the VSE operating environment, you should plan to migrate to VSE/ESA 2.2, which will incorporate all requirements necessary to handle dates beyond 1999, or if this is not possible at this time because of non ESA hardware, or other constraints, you should plan to migrate to VSE/ESA 1.4.2 and apply the necessary PTFs to enable your system for the Year2000.

To address the Year2000 problem within application systems and application programs requires decisions on the adoption of either a long-term or short-term solution, and the detailed planning for resolution. When selecting a proposed Year2000 solution, your should evaluate the following factors:

1. What is the external impact due to incompatible date format changes?

   What other programs or what output will be affected and to what extent will those programs require change if this solution is implemented for this particular application?

2. How current are the program modules that reference the date format externalized by the exposures?

Are there any plans to either eliminate or replace this particular program or routine, the programs that provide input to the program, or those receiving output from the program?

3. What functions will be impaired due to Year2000 exposures?

   Will any mission critical function within your business be compromised due to not reworking or replacing a particular program?

You should also have a plan in place to quantify the objectives before making the decision for either a short-term or long-term solution.

You may consider the following quantifiers:

1 **Identify and communicate the organization goal**

The goal is to have the function and operation of an organization Year2000 ready before any disruption caused by 2-digit year data occurs.

Year2000 ready means that products, programs, files, databases, and processes have or produce no logical or arithmetic inconsistencies when dealing with dates beyond 1999.

2 **Identify the deliverables and associated schedules for the following:**

- *Hardware*
  - Does the current processor allow for additional processing overhead and the test workload?
  - If I order a new processor, when will it arrive? When will it be installed?
  - Are there enough disk volumes for a new release of VSE/ESA or for an expansion of files and databases to cater for more bytes in each date field?
  - Are there enough disk volumes for test data?
  - Does my current configuration cater for additional DASD? Do I need additional controllers as well? If I order now, when will they be installed?
  - Do I have enough terminals to cater for additional workloads for my people? Do they require PS/2s?
- *Software*
  - I know I have to migrate to VSE/ESA 1.4 or VSE/ESA V2.1 or VSE/ESA 2.2. When do I schedule this migration? What impact will it have on my production system?
- *Systems Software Vendors*
  - I know that I have to upgrade or migrate to compatible versions of non-IBM vendor software. When do I install these products? Should I install now if possible? Will they run on my current version of VSE, so that I may take advantage of new functions now? What PTFs do I need to apply?
  - Are these products Year2000 enabled? Contact vendors for a commitment for Year2000-ready products.
  - Some vendor products check for the date for entitlement logic. How will your vendor handle that for doing the test?

- *Data Interchange*
  - How is your environment organized? Are you communicating with other systems/locations within your company? Are you interchanging data with suppliers or customers? In this case you need a thorough coordination.
- *Documentation*
  - Just how good is my documentation? Do I need to update my system diagrams to reflect new hardware and software? When will this be done?
  - When are the new DASD layouts going to be available? Has my system programmer caught up with the documentation from the last upgrade?
- *Training*
  - I know that the new VSE/ESA console is different from my current console. What training is available on the new console support?
  - There are new operator messages. What are they? Where are they documented? When is education scheduled for the new releases of my optional products? Do I need to send all my applications people to COBOL for VSE courses? When?
- *Maintenance*
  - Are all my devices at the correct hardware EC level? Does my hardware engineer have to order new microcode? When will the upgrade be performed?
- *Operations/Administration*

  Because of changes to cater for Year2000 enabled programs, are there extra programs to run in the batch window? Will my batch window now be too small?
  - Will my on-line system take longer to initialize?
  - Are there changes to the network?
  - Are there changes to the procedures?
- *Acceptance criteria for all deliverables*
  - Has all hardware been tested and in production? Has the microcode been placed in operation or in test mode? Is it reversible?
  - Has the new system software been tested thoroughly? Is there any impact on my production system? Can I revert back to my old system? How long will it take?

3 **Analyze job assignments**

- Identify the responsible person or organization. For example:

  ⇒ Customers

  Identify the person responsible for changes of data format on reports or documents shipped with goods. Determine people in customer sites who will be dealing with data interchange. Coordinate changes made internally and externally. Make the customers aware of changes being made.

⇒ Management

- Chief Executive Officer

  Determine how the Chief Executive will be informed of progress and who will prepare documentation for the Board of Directors for approval.

- Chief Financial Officer

  Determine who is responsible for additional expenditure for hardware and software, if required.

- Software Development Managers

  Determine who is going to manage the Year2000 project and under which criteria or priority. Determine which projects may be deferred by giving priority to the Year2000 project. Determine Team Leadership and project ownership.

- Operations/Administration Managers

  Determine who will manage and own the Year2000 project. Determine the best testing techniques and who will manage the resources.

- Budget/Finance Managers

  Determine with whom to liaise concerning budget over-runs, additional staff where required, who pays for what and when.

⇒ Computer Vendors

⇒ Solution Developers

⇒ Consulting/Integration service providers

⇒ System analysts

⇒ System designers

⇒ System/application programmers

⇒ Operations personnel

⇒ End Users

⇒ Auditors/quality assurance people

- Measure the estimated completion time
- Identify precedences/dependencies
- Identify resources and skills
- Identify critical path schedule
- Measure efforts
- Measure costs
- Identify technical factors

- Analyze potential benefits
  - Return on investment
  - Achievement of business goals
  - Potential quality and acceptance of the approach
  - Your business keeps running
- Analyze risk factors
  - Complexity of the task
  - Resource/time constraints
  - Length of project
  - Critical development skills

4 **Measure the criticality of each task and set priorities**

Evaluate and determine how critical the functions of each entity are to the business success of the organization, and set a priority for providing Year2000-ready solutions. The factors contributing to how critical a task may be might include pressure of demand from end users, legal issues, financial issues, or political issues.

Impact to the Business

To determine the impact to your business, consider:

- Is it critical to the operation of the business (legal compliance)?
- Is it critical to the uninterrupted operation of the business (such as payroll)?
- Is it required to support the business (such as management/financial reports)?
- Is it desirable, but not absolutely required to support the business?

Impact to Operations

Once classified by task, then determine impact severity. For example, you could use categories such as:

- **Fatal**

  Operations will ABEND or terminate

- **Critical**

  Operations will produce an incorrect result, for example, expiration dates for food items are calculated as over 100 years old, not one or two days old.

- **Marginal**

  Minor inconvenience, annoyance, or irritation, for example, inventory reports collate dates 00 prior to 99.

5 **Establish a 'critical event horizon'**

Business environments are unique, and so is the initial date that Year2000 problems will occur. If, for example, you prepare your business forecasts on a three year cycle, the fourth quarter of 1996 might be your critical event horizon. It is unlikely that in any business cycle the Year2000 problem will not occur prior to 1999 or 2000.

6 **Provide data administration**

- Identify the responsibility and scope of migrating the affected data
  - Exclusive. The data object is created and processed independent of any other business area. There is no data or file integration with other applications.
  - Primary responsibility. This means that this application defines and creates the application data, and these definitions and data are passed to other business areas which should use them as defined.
  - Secondary responsibility. This means that this application receives and accepts the data and data definitions from a primary application.
  - External Exposures.
    - The definitions and data are defined within the business and exported outside the enterprise and beyond the scope of the enterprise.
    - Data is created outside your enterprise and then imported into it for use.
- Determine formats of the data dictionary
- Determine procedures for changing and entering data definitions
- Determine procedures for data sharing and use

7 **Decide technical and management approaches**

- Programming standards
- Hardware/software
- Development and test procedures
- Prototyping and parallel development
- Process/data modelling
- Data dictionary
- Documentation structure, layout and standards
- Reviews
- Quality assurance procedures
- Testing methods
- Automated tools
- Migration of and bridging to existing Year2000-ready systems
- Estimated cost of future maintenance

8 **Identify project constraints, interfaces and dependencies**

- End users/customers
  - Availability of test and other data
  - Availability of facilities and services
  - Responsibility for end user tests
  - Other actions
    - Interfaces and dependencies with other projects

- Supporting services and facilities required
- Solution Developer automated tools
- Risks and alternatives
- Other assumptions

1.3 Solutions

1.3.1 Overview of Solutions

Apart from getting your operating system Year2000-enabled you must correct improper date notation used in your applications and databases. The following two ways to solve the problem are described in detail in this redbook:

1. The long-term solution

   This means rewriting your applications using Year2000-ready languages for 4-digit years and rebuilding your databases with 4-digit years. This solution is described in Chapter 2, “The Long-Term Solution” on page 9.

2. The short-term solution

   This is a 2-digit solution and uses the windowing technique or special encoding. This approach requires changes to your programs only, no data changes are required. This solution is described in Chapter 3, “The Short-Term Solution” on page 13.

1.3.2 IBM Assistance

IBM has made available to everyone a comprehensive Year 2000 resource guide, at no charge. The guide includes a compilation of IBM′s Year 2000 findings, recommended approaches and product listings. Also included is a bibliography of other Year 2000 publications available throughout the industries. This guide also contains a list of tools that are currently available (IBM and non-IBM software vendors) to assist in identifying potential exposures.

The document, entitled *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation* is available in softcopy form on the World Wide Web through the IBM Software Home Page, at

http://www.software.ibm.com/year2000/index.html or through IBM.

Chapter 2. The Long-Term Solution

2.1 Overview

The long-term solution to the Year2000 problem is to plan for, and implement changes to your applications, application programs, files and databases to incorporate full 4-digit year notation.

You should plan to migrate to VSE/ESA 2.2 or VSE/ESA 1.4.2 (plus PTFs) for full operating system enablement of the 4-digit year, and provide the base to support the correct levels of language compilers, and optional products.

You should also plan to migrate to a programming language that supports 4-digit dates either internally, or by using the date/time services of Language Environment/VSE (LE/VSE).

The programming languages that support full 4-digit year notation are:

- COBOL/VSE
- PL/I VSE
- C/VSE

Note: VS COBOL II may use the date/time services of LE/VSE dynamically, but does not itself support 4-digit years.

2.2 Conversion to Full 4-Digit Year Format

This approach requires changes to data and programs by **converting all references or uses** of 2-digit year format (YY) to 4-digit year format (YYYY). It also requires that you convert all software programs that reference or use the updated data simultaneously, or use a ′bridging′ mechanism to perform the conversion between old and new data and programs, otherwise you will immediately encounter data integrity problems caused by the inconsistency of date/time data formats.

Getting requirements and design changes into the development cycle takes time. The review and the modifications of the application take time and resources.

Bridging programs are often used to convert data from one record format to another. If you use such a program, it should define the:

- Input date format and encoding method
- Output date format and encoding method
- Logic that converts the date from input format to output format based on their encoding methods.

You can apply bridge programs during program execution or file and/or database conversion. When used during program execution, the conversion occurs each time data is passed between programs or between program and source data using different record formats. When used during file and/or database conversion, the bridge program reads one record at a time from the source data, transparently converts the record format, and writes out the data in

the new format to the destination. This process is incremental and can continue until all the records in the source are converted.

Bridge programs performing data format conversion provide the following benefits:

- Granularity when changing the data/code

  With the scope of the Year2000 project, it is not practical to change all the code and data at once. Bridge programs allow the gradual conversion of the programs and/or data and still maintain the compatibility between data formats. For example, you could change some of your programs to adopt new data format and still be able to communicate with programs using the old format (after conversion by the bridge programs).

  Therefore, changes to the remainder of your programs can be performed in an incremental manner as convenient.

- Flexibility when choosing appropriate solutions

  Bridge programs allow you to select the appropriate mix of different solutions to best meet your specific circumstances, while maintaining the compatibility between different data formats. For example, you can design your programs so that they can process data in different formats. You can then have active data in a 4-digit year format and archive the same type of data in 2-digit year format. The bridge program distinguishes the data in these various formats by reading the records and, when necessary, converting the data to the appropriate format.

To ease your migration, you might consider ignoring any non-impact (cosmetic) data fields in the YY format. A cosmetic date is one, that if externalized, is only interpreted by humans. Such occurrences might include the date on a printed separator page, or a display only date on a screen from a CICS/VSE application.

Note: Be careful when selecting those situations that you can decide to ignore and call cosmetic only. Be certain that they will not cause any data integrity exposures or are not accessed by any other program.

If you allow an end user to continue to input 2-digit dates for compatibility and ease of data entry, then the responsibility to translate that data into full 4-digit data falls to you. One possible solution would be to apply a context sensitive prompt to allow the user to select a century indicator. For example, allow all dates to be entered in 2-digit format, and automatically prefix these dates with the current century unless the date is a future date or historical date. What constitutes ′future′ or ′historical′ is your decision. Using this scheme, a future date in context of a loan maturity date could be set to 20YY, or a historical date forces the user to select a century from a ′choose a century′ prompt list.

2.2.1 VSE/VSAM Based Systems

You should identify all fields representing date fields, and relocate these fields as full 4-digit year fields. This may cause relocation of adjacent fields.

You should pay particular attention to concatenated keys which include dates as part of the key.

Migration from one format to another may be achieved by the use of program logic, either under special program control, or by using the bridging techniques

as described earlier. These programs could also update century information into the newly loaded records.

Applications and programs will need to reflect the new date format prior to accessing the new VSE/VSAM files to prevent data checks when manipulating data contained in these files.

2.2.2 DL/I Based Systems

You should identify all fields representing date fields, and relocate these fields as full 4-digit year fields. This may cause relocation of adjacent fields.

The existing data bases will need to be unloaded using the DL/I utilities ready for migration.

The new format will require a Database Generation to generate the new DL/I database formats, prior to reloading with the utilities using the unloaded data as input to the utility.

If there is on-line access to the DL/I databases from the CICS/VSE system, the DLZACT will need to be assembled and link edited to provide a new DLZNUC nucleus.

An application program may be required to provide the logic to update the century information into the date data fields in the new DL/I segments.

All programs accessing the DL/I database segments will have to reflect the date formats to prevent data errors when manipulating the data stored in the DL/I databases.

2.2.3 SQL/DS Based Systems

You should identify all fields representing date fields, and ensure that these fields have been defined as being in date format. If this is the case, correct 4-digit year information is fully supported.

If the date fields have been defined as integer or character format, then the 4-digit year notation may or may not be in correct format, and you should consider changing all of these date fields to SQL/DS date format.

To do this, you will have to:

- Unload the SQL/DS table.

  You may choose to use the SELECT statement to format the field correctly for the data unload.
- Drop the SQL/DS table.
- Redefine the SQL/DS table with the correct date format.
- Reload the SQL/DS table using the data unloaded previously as input to the load utility.

2.2.4 Advantages of Converting to 4-Digit-Years

1. Can provide a 4-digit-year format. It is considered to be the **only** complete, permanent, and obvious solution.
2. Provides increased security against potential inappropriate decisions today if you do not selectively ignore ′cosmetic only′ situations.
3. Can ease your migration if you selectively ignore ′cosmetic only′ situations.

2.2.5 Disadvantages of Converting to 4-Digit-Years

1. Need to convert the year data from 2-digit format to 4-digit format in all cases.
2. Requires that you relocate adjacent fields in the data field layout and usually requires that you increase record lengths.
3. Inherent future risk in initial assessment that decided that a particular situation could be ignored as ′cosmetic only′.
4. Minor aspects

   Increased DASD space usage required due to date field expansion of data. Consider including not only active, but also archived data, and duplication during conversion.

   Might experience a performance impact due to increased time in processing and data access.

Chapter 3. The Short-Term Solution

The short-term solution to resolve the Year2000 problem is to plan for, and implement changes to your applications, applications programs, operating system base and optional products to support full 4-digit years without making alterations to files and databases.

It is still necessary to migrate to VSE/ESA 2.2 or VSE/ESA 1.4.2 (with PTFs) to provide full operating system support for 4-digit years.

Migration to programming languages that support 4-digit years either internally or by the callable services of LE/VSE is also required.

The languages that support the 4-digit year notation are:

- COBOL/VSE
- PL/I VSE
- C/VSE

Note: VS COBOL II may use the date/time callable services of LE/VSE dynamically, but does not support 4-digit dates internally.

The following methods may be used as interim measures within applications, application programs, and routines to provide a solution to the Year2000 problem.

3.1 Windowing Techniques Introduction

This is a 2-digit solution that externalizes either 2-digit or 4-digit year formats. This approach requires changes to your programs only; no data changes are required.

Caution

These approaches can be applied only to the dates within a maximum 100-year period at any one time. This solution is considered temporary because there is no guarantee that in future your application will not expand to process dates that are more than 100 years apart. Therefore, this approach always carries with it a potential future exposure. (For example, humans are living longer. Therefore databases that contain birthdays (medical, civil, insurance, and so on) and the applications that access that data are already at risk with many dates spanning 100+ years.)

Two types of **windowing** techniques have been defined:

- the fixed window technique
- the sliding window technique.

3.2 Fixed Window Technique

The **fixed window** technique uses a static 100-year interval that generally crosses a century boundary. This technique determines the century of a 2-digit year by comparing the 2-digit year against a window of 100 years. The user specifies the number of years in the past and future relative to a specific year within the 100-year interval.

For example, the window (1950, 2049) provides the following range of valid dates from 19XX to 20XX :

- 19XX where XX = 50 to 99
- 20XX where XX = 00 to 49

3.2.1.1 Advantages of Fixed Windows

1. No need to expand the 2-digit year date to a 4-digit format.
2. Can provide a 4-digit year format for data reference.
3. Can distinguish years from different centuries using only 2-digit year format (provided the years being processed are in the range of 100 years at any one time).
4. Can be useful if the particular program is being phased out, and a temporary solution is required.

3.2.1.2 Disadvantages of Fixed Windows

1. Potential exposures exist if the function of the software application needs to process years beyond the range of 100 years.
2. Expect a performance impact in direct proportion to the quantity of data processing the particular application handles due to the overhead of 2- to 4-digit year conversion.
3. All programs that use the fixed window technique may need to be changed on an annual basis depending on how your date routine is packaged.
4. Retaining a 2-digit year representation does not provide collating sequence support, nor does the use of a fixed window technique provide indexing sequence support when 2-digit years are used as index keys in indexed files. You will need to provide additional processing to obtain correct collating and index sequence output.

3.3 Sliding Window Technique

The **sliding window** technique uses a self-advancing 100-year interval that generally crosses a century boundary. This technique determines the century of a 2-digit year by comparing a 2-digit year against a window of 100 years. The user specifies the number of years in the past and in the future relative to the system year (generally the current year) that the system sets and maintains. Your applications can access the date that the system sets and automatically advances. This is the main advantage of using a sliding window over the fixed window (where the window is immovable without manually revising the program every year).

As appropriate to your application environment, you can maintain more than one window. For example, you could set one window to process historical dates, one for mortgage dates, one for birth dates, and so on. The program adjusts the

system date and the past and future windows to meet the specific application's needs.

A sliding window approach requires programming logic to interpret the meaning of all 2-digit year data. Such additional programming logic could be packaged into a common date/time service routine, callable from a 2-digit year data exploiter. This would reduce the programming overhead and impact to the calling programs.

With this technique, the boundaries of the window are defined relative to the current year. For example, the window (-70, +20) relative to the current year (1996) provides the following dates from 19XX to 20XX :

- 19XX where XX = 17 to 99
- 20XX where XX = 00 to 16

This means that the resulting 100 year window ranges from 1917 to 2016.

3.3.1.1 Advantages of Sliding Windows

1. No need to expand the 2-digit year format to a 4-digit format.
2. Can provide 4-digit-year format for data reference.
3. Can distinguish years from different centuries using only 2-digit year format provided that the years being processed are in the range of 100 years at any one time.
4. No need to convert the date data to a new representation scheme.

3.3.1.2 Disadvantages of Sliding Windows

1. Potential exposure exists if the function of the software application needs to process years beyond the range of 100 years.
2. Potential performance impact in direct proportion to the quantity of date processing the application handles.
3. All programs that accept output from the sliding window technique must use the same assumptions (current date, past and future windows).
4. Retaining a 2-digit year representation does not provide collating sequence support, nor does the use of a sliding window technique provide indexing sequence support when 2-digit years are used as index keys in indexed files. You will need to provide additional processing to obtain correct collating and index sequence output.

3.4 A 2-Digit Encoding/Compression Scheme

This is a 2-digit solution that externalizes only a 2-digit-year format. The encoding/compression technique allows you to represent a 4-digit date in your existing 2-byte field. You can convert the date to hexadecimal or to unsigned packed decimal. This method requires changes to both your data and your programs. It also requires that you convert, simultaneously, all applications that reference or use the updated data.

Example techniques that are useful when using this solution include **encoding** or **compressing** 4-digit year data into 2-digit existing space. This section presents several specific examples, many others exist, and might prove more applicable to your specific needs.

Caution

Apply this approach with caution. It is considered to be the least desirable approach and should only be used if absolutely necessary. Be certain that the new encoding or numbering scheme does not affect the proper functioning of your programs after all the data changes are implemented. If you exchange data with other systems, you should not use exotic encoding methods.

This solution is considered temporary because there is no guarantee that in the future, your applications will not expand to process dates outside the encoding limited.

Following are some examples:

- **Example 1:** Convert the numbering scheme from decimal to hexadecimal. However, two hexadecimal digits can only provide a maximum of 255 years, for example:

  D¢1900¢ + X¢FF¢ = 1900 + 255 = 2155

  Specific date conversions might be:

  - Convert the year 1900 represented by D'00' to X'00'
  - Convert the year 1999 represented by D'99' to X'63'
  - Convert the year 2000 represented by D'00' to X'64'

- **Example 2:** Convert the data type from the 2-byte character representation of the 2-digit year to a 1-byte **unsigned packed decimal** (two digits) representation and use the freed byte to append two **unsigned packed decimal** digits to represent a 4-digit year. For example:

  - Convert the year 1900 represented by character string '00' (X'F0F0') to unsigned packed decimal X'00' and prefix unsigned packed decimal X'19' in front of X'00' to yield X'1900' in unsigned packed decimal.
  - Convert the year 1999 represented by character string '99' (X'F9F9') to unsigned packed decimal X'99' and prefix unsigned packed decimal X'19' in front of X'99' to yield X'1999' in unsigned packed decimal.
  - Convert the year 2000 represented by character string '00' (X'F0F0') to unsigned packed decimal X'00' and prefix unsigned packed decimal X'20' in front of X'00' to yield X'2000' in unsigned packed decimal.

- **Example 3:** Convert the numbering scheme from decimal to a user defined numbering scheme. The mapping between the new and old schemes can be defined by a table or mapping function. The conversion between the two numbering systems can be done by table lookup or functional mapping. Table 1 on page 17 presents one such possible user-defined table that provides for values up to 1295 within a 2-digit field.This scheme uses the characters 0-9 and A-Z to represent the decimal values 0-35 respectively. For example :

  D¢1900¢ + $base_{36}$¢ZZ¢ = 1900 + 1295 = 3195

2-character year (encoded) value	Converted Date/Year Value	Year (When using 1900 as the Base Year)
00-0Z	00-35	1900-1935
10-1Z	36-71	1936-1971
20-2Z	72-107	1972-2007
30-3Z	108-143	2008-2043
40-4Z	144-179	2044-2079
..	..	..
R0-RZ	972-1007	2872-2907
..	..	..
Z0-ZZ	1260-1295	3160-3195

Table 1. Example User-Defined Date/Year Conversion Table

- **Example 4:** Pack a 3-digit date field with a 4-digit year date by the use of the CYY format. Using a conversion table or offset, you can indicate, for example that C=0, 1, or 2 represents 19, 20, or 21 respectively. When your application appends the C to the YY field, your system produces full 4-digit year dates, the range of which depends on the conversion mechanism. Using decimals 0-9 to represent 19-28, this scheme represents a solution from 1900-2899, but it is likely to require end-user procedural changes, education, and typical learning curve time and errors.

  One advantage of setting C=0 to represent 19 and so on is that it might provide a compatible, non-disruptive change to some existing application routines if such a field is currently prefixed to your YY data field and set to 0.

  A variation on this same scheme would include the use of CCYY format where CC can be used to represent the actual century indicator, 18, 19, 20, and so on, or an encoded value for example,″00″, ″01″, and ″02″ to represent 19, 20, and 21 respectively.

  When adding either the C or CC prefix to the YY field for CYY or CCYY representation, C or CC can be extracted from a separate field, one that is not necessarily adjacent to or preceding the YY field. This can then relieve any restrictions you might have currently because of your date field length. It does, however, require further programming logic and data manipulation.

3.4.1.1 Advantages of a 2-digit Encoding/Compression Scheme

1. No need to expand the 2-digit year data format to 4-digit data format. (For example, there is no need to increase the fields in databases and files to accommodate dates above 99 which would increase DASD usage.) Further, this saves the effort that would be required to rebuild your databases or files.
2. Can distinguish years from different centuries using the 2-character year format.
3. If you use a COBOL COMP-3 format, you can pack a CCYYMMDD date into an existing six byte field. This technique allows you to retain the original field size and eliminates your need to relocate adjacent fields.
4. If you use a flagged Julian format (CYYDDD) where C is used as the century indicator, the format does not require expansion of the data field.

3.4.1.2 Disadvantages of a 2-digit Encoding/Compression Scheme

1. Depending on the choice of data representation you implement, this scheme can be applied only to a limited date range. For example, you are limited to 255 years when using hexadecimal representation.
2. All programs that use this scheme and need to access the output of the 2-character conversion must change simultaneously.
3. Due to data conversion (calls and processing) you might experience a performance impact in direct proportion to the quantity of data processing the particular application handles.
4. Depending on the choice of data representation you implement, you might experience incorrect data sequencing if you do not add further programming logic.
5. Encoded dates require conversion whenever you work with that data. Therefore, the presence of encoded dates will add another layer of complexity to tasks such as problem determination.
6. You must convert the data before it can be displayed in Gregorian format, and some encoded data can only be viewed in hexadecimal format. This is both impractical for human reading and also impractical or impossible to print.

3.5 Sliding Windows and Expiration Dates in a VSE/ESA System

Following is a list of rules which should be followed across VSE/ESA and its components when handling 2-digit years and expiration dates. The rules are based on the use of the sliding window technique.

1. Most dates in VSE/ESA relate to past events. For example, file creation date, date of last update, date when a fix was applied, or date of a log entry. If only a 2-digit year is available for such dates (supplied as input or stored), a sliding window (ranging from -79 to +20) is used to obtain a date with a 4-digit year. This is also the default of the YEAR224 macro support.
2. File expiration dates are future orientated and need special handling:
   a. Whenever a file expiration date is defined with a 2-digit year (JCL or VSE/VSAM), a sliding window (0,+99) is used to complement the date with a century.
   b. Whenever a file expiration date is stored with a 2-digit year (VSE/VSAM and Diskette), a window with creation date as the lower boundary is used to determine century. This method ensures a consistent interpretation of file creation and expiration dates at all times.
3. Files with expiration dates of 1999/nnn, with nnn > 365 are always considered unexpired. Such date specifications are supported both by JCL and VSE/VSAM (JCL with nnn=366 only) and are used for files that should never expire.
4. Files with an expiration date of 1999/365 will **not** expire if the date was specified explicitly. They will, however, expire, if this date was calculated from a retention period.

3.6 YEAR224 Macro

This new assembler macro is provided by VSE/ESA 2.2 (and as PTF for VSE/ESA 1.4.2) and complements a 2-digit year field, supplied as input, with a 2-digit century field, depending on a specified 100 year window relative to the current year. The format is as follows:

[name] **YEAR224 YEAR=**{address|(rx)}

[,**WINDOW=**{number|**20**|(rx)}]

[,**LINK={YES|NO}**]

[,**NAME=**{name|**$IJBY224|IJBY224S}**]

YEAR=address|(rx) Specifies the address of a 4-byte area containing the 2-digit year input in the form ′..yy′. The contents of substring ′..′ is ignored on input and is replaced by the century on output, resulting in a 4-digit year ′yyyy′.

WINDOW=number of years|20|(rx) Specifies a number between 0 and 99 that is interpreted as the forward width of a 100 year window relative to the current year.

LINK = **YES|NO** Determines how the service routine is invoked. With the default of LINK=NO, SVA phase $IJBY224 is invoked (no link-editing is required). When LINK=YES is specified, the macro invokes IJBY224S as the default CSECT for link-editing. LINK=YES means that the call sequence generates an external reference to the entry point of the service routine (CSECT) to be linked with the calling program.

NAME=name|$IJBY224|IJBY224S Specifies a user-defined phase or CSECT as the service routine to be invoked. If no name is specifed, the following default names are used, as mentioned in the LINK parameter description:
For LINK=NO, phase name $IJBY224
For LINK=YES, CSECT name IJBY224S

Register usage:

(rx) Register notation is only supported with registers 2-12.

0 Is used for input to pass the window parameter, and returns one of the following reason codes:

- **0** No errors (register 15 contains also 0)l
- **1** YEAR input is not numeric
- **2** WINDOW is invalid (negative or >99)

1 Is used for input to pass the YEAR parameter

13 Is assumed to contain the address of a 72-byte save area

14 Is used as link register

15 Is used on input for the address of the service routine, and returns one of the following return codes:

- **0** No errors
- **8** Input is invalid as indicated by the reason code in register 0.

Macro expansion and service routine are both reentrant. The service routine uses the GETIME macro and references the COMREG field SYSDATE to obtain the current date, and is therefore VSE-dependent. It may also be used in a CMS environment under VM/ESA.

Example of usage:

```
         YEAR224 YEAR=yyyy
         ....

yyyy     DC    ¢..02¢
```

complements field yyyy to '2002' since 2002 is the year ending with 02 in the default window (1996-79,1996+20) = (1917,2016).

Chapter 4. VSE/ESA Components and Functions Year2000 Support

In this chapter you will find a description of the changes made to the VSE/ESA base and optional products to enable them for the Year 2000. These changes are incorporated in VSE/ESA 2.2.0. They are also available for VSE/ESA 2.1.x or VSE/ESA 1.4.2 via PTFs. Refer to Appendix A, “Product Information for VSE/ESA 2.1.x and 1.4.x” on page 149 if you plan to upgrade your VSE/ESA via PTF installation. If you are currently not on one of these 3 levels, you should start migrating to VSE/ESA 2.2 or VSE/ESA 1.4.2 as soon as possible.

All VSE/ESA extensions for Year 2000 maintain compatibility for dates stored in file labels, libraries, system files, and symptom records. They also preserve the capability of sharing data with other platforms, or with VSE/ESA systems that do not have the extensions yet applied.

Compatibility is also maintained for all control statements that continue to accept a 2-digit year input, although such inputs are now interpreted according to a documented 100 year window. Date specification with a 4-digit year is added as an option.

Because the date information in many messages and formatted printout is extended after the system is enabled for the Year 2000, programs that are scanning the hardcopy file or SYSLST output, may require changes to cope with the new date format.

One of the most important changes is the extension to the IPL SET command. This change makes it possible to set the correct century information in the partition COMREG. The date from the COMREG is used by many other functions or program products (for example, CICS/VSE). If the date is not correct in the COMREG, those products that retrieve the date will have incorrect century information.

All described changes are applicable to VSE/ESA V2 and VSE/ESA 1.4 unless explicitly specified.

After reading this chapter it should be clear that you should migrate now to a level of VSE/ESA that has the enhanced Year2000 support. You will also find some guidelines to assist you when preparing your system for the Year 2000.

Note: Whenever we speak about a date format in the form of mm/dd/yy, it could also be dd/mm/yy, depending on the DATE option of the STDOPT statement.

In the following description, for most changes you will find two headings ***″Current Systems″*** and ***″Year2000 Enabled Systems″***. Under the first heading you will find a description of the behavior of a VSE/SP or a VSE/ESA system that has no Year2000 support. Under the second heading you will find a description of the system after installing VSE/ESA 2.2 or the corresponding PTFs.

4.1 Initial Program Load (IPL)

The IPL SET command is extended to allow the specification of a 4-digit year in the DATE operand.

Current Systems

During IPL message **0I30I** is displayed and the system date in the supervisor is initialized. The message displays the date in the format mm/dd/yy, where ′yy′ represents the last two digits of the year. In case the TOD clock has not yet been set, for example, because of POWER ON, this is done using the command
IPL SET DATE= ,CLOCK=
This command also causes the system date to be initialized, and the DATE operand has the format mm/dd/yy. For ′yy′ you have to enter the last two digits of the year and 19yy is assumed.

When the clock is in SET state, and the IPL SET command is used for ZONE correction only (SET ZONE=), no problem occurs when crossing the Year2000 boundary. The date is correctly displayed and the system date is correctly set by the VSE/ESA system. A date of 01/02/00 is to be interpreted as January 2, 2000.

If the TOD clock has to be set, for whatever reason, after January 1, 2000, it will not be possible to do this correctly on a current system. The ′yy′ entered in the SET command always means 19yy and there is no way to set a date beyond December 31, 1999.

Example:

The command : **SET DATE=01/02/00,CLOCK=12/30/12**

would initialize the clock and system date to January 2, 1900 on a system that does not have the required support.

Message **0I30I** has the following format:

0I30I DATE=mm/dd/yy,CLOCK=hh/mm/ss,ZONE=zone/hh/mm

It is left to the user to find out if the yy means 19yy or 20yy.

Test Results:

We did a test with a VSE/ESA 1.3.6 system. We used **DATE=01/01/00** in the IPL SET command. A display of the COMREG field JOBDATE showed: **01/02/00/19**

When we used **DATE=02/29/00**, we received message:

0I87D INVALID SPECIFICATION FOR KEYWORD DATE

When we used **DATE=02/28/00**, the COMREG field JOBDATE contained **02/29/00/19**

This clearly shows, that it will be impossible to set a correct date on the 1.3.x systems after December 31, 1999. Not only is the century incorrect, it will be impossible to IPL with the DATE set to January 1.

Year2000 Enabled Systems

The IPL SET command is extended to allow the specification of the year including the century when entering the date. For compatibility reasons, the specification of only the last two digits of the year is still accepted. If the DATE format mm/dd/yy is used, the yy will have the meaning 19yy if it is above 50. For values below or equal to 50 the meaning will be 20yy.

Example:

A SET command is entered:

SET DATE=01/15/02,CLOCK=12/50/00

will set the TOD clock to 12.50 PM on January 15, 2002.

The new format of the IPL SET command is:

```
SET DATE=mm/dd/{yyyy|yy},CLOCK=hh/mm/ss,ZONE={EAST|WEST}/hh/mm
```

Figure 1. New Format of the IPL SET Command

Message 0I30I will have the following format:

```
0I30I DATE=mm/dd/yyyy,CLOCK=hh/mm/ss,ZONE=zone/hh/mm
```

Figure 2. New Format of Message 0I030I

The highest DATE and CLOCK value that can be specified is

SET DATE=09/17/2042,CLOCK=23/53/47.

Any higher value will cause a TOD clock overflow and will be rejected.

4.2 Attention Routine Commands

Current Systems

On systems that do not have the support installed, the AR commands AUTOIPL, SIR and TIME will display the date with a two digit year. The output of these commands will show the year 2000 as ″00″ in the date field. The TIME command supports only dates up to December 31, 1999.

Year2000 Enabled Systems

The date output fields of the AR commands AUTOIPL, SIR and TIME are displayed with a 4-digit year in the date field. The TIME command is changed to accept both a date with a 2-digit year or a date with a 4-digit year. When a two digit year is specified, a number above 50 is interpreted as 19yy and a number below or equal to 50 as 20yy.

4.3 Job Control

The JCL statements JOB, /&, DATE, DLBL, SET, and TLBL are enhanced for the Year 2000.

- JOB and /&

  The output of the JOB and /& (EOJ) statements on SYSLOG and SYSLST has been changed to show a 4-digit date field. If, however, a DATE statement with an operand length of eight has been specified in a job, the EOJ date is shown in the old format.

- DATE

  Current Systems

  Currently, job control does not check the operands of the DATE statement, except for a length of eight characters. Any character string with a length of eight that does not contain any of the characters blank, comma, or equal is considered to be a valid date.

  Year2000 Enabled Systems

  The DATE statement processing has been extended to support 4-digit years. The new statement accepts operands with a length of 10 characters and, for compatibility reasons, operands with a length of eight characters. Syntax and validity checking is done for the new format only.

  The JOBDATE field in the COMREG is used, as it is currently, to store the DATE specification if the old format is used. The new format uses the COMREG field JOBDATWC which appends the century to JOBDATE. The existing flag DATEBIT is set on when a DATE statement is active for a job in a partition. An additional flag, IJBDATEO in JCSW9, is set on to indicate that the old DATE format was used.

- SET DATE

  SET DATE accepts only the formats mm/dd/yy and dd/mm/yy. It does not support the Year 2000. In addition, the current description of the SET DATE statement is technically not correct. The statement ″sets the system date permanently to the specified value″ is no longer valid. It is recommended not to use this statement in future.

- DLBL and TLBL

  Current Systems

  Currently the DATE operand can be supplied in different formats:

  – The retention period format which can be specified as:

    1. Decimal number dddd (from 0-9999) with an operand length of 1 to 4.
    2. Decimal number in the form 00/dddd (0-9999) with an operand length of 4 to 7.

  – The date format which can be specified as:

    1. yy/ddd (with yy not equal to 00); this is the same as specifying 19yy/ddd
    2. 19yy/ddd
    3. 20yy/ddd

Year2000 Enabled Systems

There are no changes to the DLBL and TLBL processing if:

1. The DATE operand has been omitted, or has been specified in the retention period format.
2. The DATE operand has been specified in the 19yy/ddd or 20yy/ddd format.

The DLBL and TLBL processing has been changed if the DATE operand is specified in the yy/ddd format. In this case the yy/ddd information is complemented by the YEAR224 macro to ccyy/ddd. The century information supplied by the YEAR224 macro depends on:

- The date yy/ddd to be complemented.
- The current date.
- The WINDOW operand used for the YEAR224 macro.

In the context of expiration dates, WINDOW=99 is used to complement yy/ddd. This is because expiration dates are future-oriented. In the context of creation dates, WINDOW=20 is used. This is because creation dates are past-oriented.

Note: For more information about this windowing techniques and the YEAR224 macro see 3.6, “YEAR224 Macro” on page 19.

Examples:

1. In the year 1996 the expiration date 98/180 is interpreted as 1998/180, since 1998 is the year ending with 98 in the window (1996-0,1996+99) = (1996,2095). The expiration date 95/180 is interpreted as 2095/180, whereas the creation date is interpreted as 1995/180, since 1995 is the year 95 in the window (1995-79,1995+20) = (1916,2015).
2. In the year 1999 the expiration date 98/180 is interpreted as 2098/180 because 2098 is the year ending with 98 in the window (1999-0,1999+99) = (1999,2028). The creation date 98/180 is interpreted as 1998/180 because 1998 is the year ending with 98 in the window (1999-79,1999+20)= (1920,2019).

4.3.1 Job Accounting

Whenever a job step is completed, job control invokes the user accounting routine $JOBACCT. Accounting data is passed to the user in the accounting table ACCTABLE addressed by the COMREG pointer JAPART. There are no changes to the current support. It is up to the user to retrieve the century information from the COMREG field SYSCENT.

4.4 Linkage Editor

Current Systems

The linkage editor prints a date on two header lines of its printout on SYSLST. This date has the format mm/dd/yy or dd/mm/yy.

Year2000 Enabled Systems

The linkage editor extends the year representation to four digits. It retrieves this from the COMREG field JOBDATWC. Programs that are scanning the printout of the linkage editor may have to be adapted because of the new date format.

There is one exception to this rule. The linkage editor will not use four digits to display the year if the //DATE job control statement has been used, specifying only two digits for the year. In this case, the header lines also show a 2-digit year.

4.5 Basic Access Methods (SAM, DAM, ISAM)

Current Systems

- For tape files using DTFMT or DTFPH, the following applies:
  - Data files or standard labeled work files will be overwritten on December 31, 1999, if an expiration date of 99/365 was specified in the TLBL statement or if the retention period from the TLBL plus the current date yields 99/365.
  - Standard labeled files with an expiration date > 99/365 will never expire.
  - Standard labeled work files or standard labeled data files for extension will not be overwritten or extended after December 31, 1999, if an expiration date of 99/365 is specified in the TLBL statement, or if the retention period from the TLBL statement plus the current date yields 99/365. BAM always uses ″19″ as the century for the current date and also when calculating the expiration date if a retention period is specified in the TLBL statement.
- For disk files using DTFSD, DTFDI, DTFCP, DTFDA, DTFIS, or DTFPH, the following applies:
  - Standard labeled disk files will be overwritten on December 31, 1999, if 99/365 was specified in the DLBL statement as expiration date, or if the retention period from the DLBL statement plus the current date yields 99/365.
  - Standard labeled disk files with an expiration date > 99/365 will never expire.
- For unit record files using DTFCP or DTFDI where the logical unit is assigned to tape, the following applies:
  - Standard labeled files will be overwritten on December 31, 1999, if 99/365 was specified in the TLBL statement as expiration date, or if the retention period from the TLBL statement plus the current date yields 99/365.
  - Standard labeled files with an expiration date > 99/365 will never expire. BAM always uses ″19″ as the century for the current date and also when calculating the expiration date if a retention period is specified in the TLBL statement.
- For unit record files, using DTFDU for 3540 diskette, the following applies:
  - If using DTFDU for a 3540 device, files will be overwritten if an expiration date of > 99/365 was specified. Also, all files will be overwritten when the Year 2000 is reached.

Year2000 Enabled Systems

For all files mentioned under ″Current support″, BAM processing has been modified to account for century information in file labels. For this purpose, BAM uses the contents of the fields SYSDATE and SYSCENT in the partition COMREG. Expiration dates are treated according to the following rules:

- Files with an expiration date of 1999/nnn, where nnn > 365 will never expire.
- Files are treated as unexpired when 99/365 or 1999/365 are explicitly specified.
- Files will expire normally on December 31, 1999 if 1999/365 is calculated from a retention period.

4.6 VSE/VSAM

Current Systems

Currently the following date fields are maintained by VSE/VSAM as 2-digit numbers:

1. The creation date, the yyddd of the date that the file was explicitly or implicitly defined. This date is generated by VSE/VSAM.
2. The expiration date, the yyddd of the date that the file will expire. This expiration date can be specified via:
   - The DEFINE CLUSTER parameter TO(yyddd). There is currently no check to verify the ddd value, so values of ddd<=999 may exist.
   - The DEFINE CLUSTER parameter FOR(nnnn), with nnnn the number of days the cluster should be retained.
   - For an implicitly defined VSE/VSAM managed SAM file via the DLBL statement. However VSE/VSAM does currently not process the century information even if specified.

The consequences of files expiring are:

1. The file can be deleted via an IDCAMS DELETE CLUSTER command without specifying the PURGE parameter.
2. A reusable file would be reset or deleted if processed with a close disposition of date. The close disposition can be specified on the DLBL statement, and in the PARM=(CLOSDSP=...) parameter of the ACB macro.

Without additional VSE/VSAM support retention periods crossing the Year 2000 would not be handled properly. Usually an expired file would show up as unexpired and, consequently, would not be deleted. This can cause operational difficulties, but no data integrity exposure.

Year2000 Enabled Systems

The central VSAM repository is the VSAM catalog. It contains date values either in the format of TOD clock (called time stamps), or in a yyddd format. Time stamps are not affected by the Year 2000 transition, but the yyddd format cannot handle different centuries. To keep the catalog format (including yyddd dates) unchanged for compatibility reasons, these dates will be interpreted correctly by using the sliding window technique.

VSE/VSAM will use different windows for dates that are typically in the past (such as the creation date) and dates that are usually in the future.

VSAM object definition

Creation date: The creation date is kept in the catalog to provide a history of the VSAM object and has little influence on the actual processing. The creation date will now be determined on the basis of the current system date (SYSDATE) rather than the job date. This means that // DATE can no longer be used to manipulate the creation date in the VSAM catalog.

The IDCAMS LISTCAT output now shows the formatted creation date with a 4-digit year.

Expiration date: The expiration date has definite consequences for file handling. It specifies the date up to which the VSAM object is not eligible for deletion and requires the PURGE keyword on the IDCAMS DELETE command to force deletion. Expiration dates are generated when a VSAM object is defined, either explicitly or implicitly:

- At explicit definition, for example using the IDCAMS DEFINE command, you can specify the expiration date on the TO() clause, or the retention period on the FOR() clause. The expiration date in the TO() can now be specified as yyddd or yyyyddd. If you specify a value between 1831 and 9999 (the maximum) in the FOR() clause, the file will never expire.
- At implicit definition, the expiration date can be specified on the DLBL statement. The DLBL syntax allows to specify either an expiration date or a retention period.

When a cluster is defined, the expiration dates you specify are to be checked for validity. Valid dates are dates which are not more than 99 years ahead of the current date. Both creation and expiration dates are stored into the catalog in the format *yyddd*.

VSAM object deletion

When an attempt is made to delete a VSAM object, either explicitly or implicitly, VSAM checks if the object has expired or not according to the following rules:

1. A date of 99/366 is considered as never expired.
2. A date of 99/365 is considered as expired if the expiration date was calculated from a retention period (FOR). An explicitly specified expiration date (TO) of 99/365 is considered as unexpired.
3. The creation date is known to be in the past. Therefore, it is converted from yyddd format to yyyyddd format, using a sliding window that ensures that it is earlier than the current date. After that, the expiration date is calculated in such a way that it is later than the creation date. If the current system date is later than, or equal to, the expiration date in the yyyyddd format, then the object is considered as expired.

VSAM listings

All dates that appear in the VSAM list output are now printed with 4-digit years.

4.7 System Utilities

4.7.1 LVTOC Utility

- Currently LVTOC displays the job date in printout in the format mm/dd/yy, using the JOBDATE from the partition COMREG, without the century information. Changes have been made to extract the century information from the COMREG.
- The FORMAT 1 creation and expiration dates for files are currently calculated using the dates stored in the FORMAT 1 label and adding them to 1900. LVTOC displays a 4-digit year in the date fields for creation and expiration date. Therefore, no changes have to be made to the current support. The creation and expiration dates of a file that is considered ″never expiring″ are prefixed by asterisks (***).
- LVTOC displays the VSE/VSAM time stamp available in the FORMAT 4 label. This time stamp conforms to the TOD clock. No changes are required.

4.7.2 VSE/Fast Copy Program

Current Systems

1. When creating a backup with VSE/Fast Copy using the DUMP FASTCOPY control statement, the date of the backup is stored on the tape. The JOBDATE from the partition COMREG is used without century information.

   For RESTORE of a FASTCOPY backup, the backup date is displayed in message 8F56I in the form of mm/dd/yy.

   When crossing the Year 2000, the date stored in the record on tape, and displayed by message 8F56I, could be misleading in terms of the exact year, when the dump was taken.

2. The system date from the partition COMREG is used to compare the file expiration date. The system date is used without any century considerations, whereas the actual year of the expiration date in the FORMAT 1 label can be calculated by adding yy to 1900.

   The expiration date is used to indicate an overlap on an unexpired file, and also for the NOEXPIRED parameter in the FASTCOPY control statement. NOEXPIRED specifies that all expired files are to be excluded from being dumped or copied.

   By comparing only the yy portion of the system date with information in the FORMAT 1 label, most of the files will never expire, nor a prompt for overlap on unexpired file would be issued in most cases and the NOEXPIRED parameter would not exclude any files.

Year2000 Enabled Systems

1. During FASTCOPY backup, when extracting the JOBDATE from the COMREG, the century value will be considered and stored in the information record on tape. When issuing message 8F56I, during FASTCOPY RESTORE, the date will show the year including the century information that was stored in the information record on the tape. If there is no century information available on the tape, 1900 is used as the base for the date to be displayed.
2. For the evaluation of expired/unexpired files, the century value of the system date is used to calculate the corresponding year. The same rules apply as described in 4.5, “ Basic Access Methods (SAM, DAM, ISAM)” on page 26.

4.7.3 Standalone JCL (for VSE/ESA 2.1 and following)

Current Systems

In message SA08D standalone JCL displays the job date in the format mm/dd/yy. This information is extracted from the JOBDATE field in the COMREG. The message also prompts for a possible new job date in the format of mm/dd/yy. When crossing the Year 2000, the date displayed could be misleading.

The date conversion routines do not consider the century information and that the Year 2000 is a leap year. This can lead to differences in tape label creation and expiration dates.

The century information in the tape label creation and expiration dates are not considered. Labeled tapes could expire at the end of 1999, because the century information in the tape label is not stored accordingly.

Year2000 Enabled Systems

Changes have been made to extract also the century information from the COMREG and display it in message SA08D. The new job date format, mm/dd/yyyy, will be accepted as input. The old format is also still accepted, but century information in the partition COMREG will not be updated in this case.

The date conversion routines used to calculate the file creation and expiration date are now changed recognizing the Year 2000 as a leap year, and use the century information to calculate the file creation and expiration date, and store this information in the corresponding label fields.

The century information is used for tape label creation and expiration dates. The same rules apply as described in 4.5, “ Basic Access Methods (SAM, DAM, ISAM)” on page 26.

4.7.4 Standalone Utilities (only applicable to VSE/ESA 1.4)

1. **Display and change of job date.**

   Standalone JCL prompts the user to specify a date. The entered date is stored in the standalone communication area. When crossing the Year 2000, because of the missing century information, the job and system date could be misleading. Also the calculation for expired or unexpired files for the stand alone utilities could be wrong. Therefore, when prompting for the job date, the century information will also be considered. A date entered in the old format will default to CC=19.

2. **Tape label handling**

   For tape labels, standalone utilities do not use the century information in the IBM Standard File Labels on tape. Changes will be made to set the century information correctly in the tape labels, (blank=19, 0=20, 1=21).

3. **Standalone Fastcopy**

   - Stored and displayed information about time and date when the backup has been taken.

     When creating a backup with Standalone Fastcopy, information about the time and date the backup has been taken, is stored on the tape. The standalone communication area is used as input for this. If the century information has been specified as an answer to the JCL prompt, it will be

stored on the backup tape. If no century information is available, 19 is assumed. The date in message 8F56I will also show the century information if it is available in the communication area, otherwise century 19 will be assumed.

- Checking for file expiration.

  Currently the system date from the communication area is used to compare the expiration date for a file. The date is used without any century considerations. The actual date can be calculated from the FORMAT 1 label by adding yy to 1900. For calculating the expiration date, the century value of the date in the communication area will be used to calculate the actual year.

4.8 Operator Communications

Within VSE/ESA DOC support, the date presentation affects the following utilities:

- The LISTLOG utility
- The PRINTLOG utility

Both utilities read the hardcopy file and print them on SYSLST. In VSE/ESA 2.1 both utilities do support 4-digit year presentation, both as input and output. This is not the case in VSE/ESA 1.4. Although it is very unlikely that one VSE hardcopy file would be spread over more than 100 years, the PRINTLOG and LISTLOG utilities will be changed in VSE/ESA 1.4 to be consistent with VSE/ESA 2.1.

1. LISTLOG

   The LISTLOG utility will show both in its page header lines and in its output lines a date in the format dd/mm/yyyy instead of dd/mm/yy.

2. PRINTLOG

   The PRINTLOG utility will show both in its page header lines and in its output lines a date in the format dd/mm/yyyy instead of dd/mm/yy.

   The PRINTLOG input option is extended to accept mm/dd/yyyy for the date parameter. In addition, the old format mm/dd/yy will still be supported, and ′yy′ is interpreted using the sliding window technique.

4.9 Problem Determination

4.9.1 Dump Services

Current Systems

1. The DUMP command provides a 2-digit year which is printed directly if the output is directed to a printer, or saved in the symptom record.
2. The standalone dump program saves the 2-digit year in the symptom record.
3. DOSVSDMP retrieves the 2-digit year from the symptom record.

Year2000 Enabled Systems

1. The DUMP command prints the complete year, if the output is directed to a printer. A 2-digit year is still saved in the symptom record.

2. The standalone dump program saves the 2-digit year in the symptom record as before. No changes are made.
3. DOSVSDMP retrieves the 2-digit year from the symptom record and uses the YEAR224 macro to print the 4-digit year.

4.9.2 Info/Analysis Program

Current Systems

Info/Analysis uses a 2-digit date field in the dump management file and in the symptom record. This causes dump names to be sorted incorrectly when crossing the Year 2000. Also if dates are in the stored clock format, Info/Analysis does not recognize the Year 2000 as a leap year.

Year2000 Enabled Systems

The 2-digit year representation in printouts will not be changed. When sorting the dumps in the dump management file, a 4-digit year field is used to make sure the dumps are sorted correctly. Processing has been changed to recognize the Year 2000 as a leap year.

4.9.3 SDAID

SDAID has been changed to extend the date field in the trace TOD output line to a 4-digit year field.

4.10 VSE/OLTEP

Current Systems

There is no checking that the expiration date is 99/365, indicating that the file is permanently unexpired. Furthermore, the expiration checking is incorrect after December 31, 1999 because no century information is used.

Year2000 Enabled Systems

OLTEP is enhanced to use the SYSCENT value from COMREG to check the current date against the date in the HDR1 tape label record. Expiration dates in the HDR1 label records of a tape are handled as in other VSE/ESA components. A tape will never expire if:

- The expiration date is > 1999/365.
- The expiration date is 1999/365 and it was explicitly specified and not calculated from a retention period.

4.11 VSE/ESA Macros

Currently there are two macros used to obtain date information.

1. **The COMREG macro**

   The COMREG extensions to provide century information are already available since VSE/ESA 1.3. Appendix B, “Partition Communication Region (COMREG)” on page 161 shows the date related fields of the COMREG. The field JOBDATWC contains the field JOBDATE and adds century information to

it. The field SYSCENT has to be concatenated with SYSDATE to provide a 4-digit year.

2. **The GETIME macro**

   The GETIME macro already returns mmddyycc or ddmmyycc values if CLOCK=YES is specified.

The YEAR224 macro

The YEAR224 macro is new. It is used to convert 2-digit years to 4-digit years using a sliding window. For more information about this new macro refer to 3.6, "YEAR224 Macro" on page 19.

4.12 Librarian

The DATE= parameter in the LISTDIR command indicates that member directory information is to be displayed only for those members that are created or updated during the specified time interval.

Currently LISTDIR interprets the 2-digit year values correctly according to a fixed window 1950-2049. Changes are planned that will allow for 4-digit year specifications for the DATE= parameter and to add century fields in the directory. These changes will be shipped after the end of 1996.

4.13 Maintain System History Program (MSHP)

The dates listed in the header lines of the job output are displayed with a 2-digit year field. This is not considered as a problem and they have not been changed.

Installation dates in the history file records are used in comparisons to determine the latest service applied. This could pose a problem, because the latest service of a component could show an incorrect APAR or PTF due to incorrect year comparison. No changes have been made to the history file records, because this would create incompatible records. MSHP has been changed by internally extending the years to 4-digit fields according to the sliding window approach, before dates are used in comparisons for ″latest service″.

4.14 VSE/ICCF and Year2000

Current Systems

VSE/ICCF interprets dates, which is considered as input, or assigns dates, which is considered as output by VSE/ICCF.

1. Output

   Responses to VSE/ICCF commands are displayed in the format mm/dd/yy or dd/mm/yy depending on the COMREG flag LTACT. This 2-digit representation for the year is used throughout VSE/ICCF for terminal, SYSLOG and SYSLST output. In the Year 2000, the yy attribute will contain 00. Programs which scan the output, must consider this (for example, the Interactive Interface).

   &&CURDAT is a VSE/ICCF procedure statement which provides the date in the format yymmdd.

2. Input

   Dates can be specified in two ways:

   - The expiration date for the job entry statement /FILE
   - The DTSUTIL PURGE ON|BEFORE|AFTER command.

   The /FILE statement already accepts a 2-digit or 4-digit specification, the PURGE command does not.

YEAR2000 ENABLED SYSTEM

The output representation has been changed from two to four digits without any exception. This means that programs which analyze VSE/ICCF output eventually have to be adapted to the changed output format.

In addition to &&CURDAT which returns the current date in the format yymmdd, a new VSE/ICCF procedure statement, &&CURDTX, is provided. It returns the current date in the format yyyymmdd.

The DTSUTIL PURGE ON|BEFORE|AFTER command now accepts both two and four digit input. VSE/ICCF uses the YEAR224 macro to convert a 2-digit entry to a 4-digit value. The default sliding window (-79,+20) for the YEAR224 macro is applied.

Dates in VSE/ICCF library records

The VSE/ICCF library (DTSFILE) contains four types of directory records. All of them contain a date entry. The system record (A$), the user profile records (B$), and the library header records (C$) will additionally hold the century information in decimal format. The directory entries for members (D$) have a history of changing date formats which VSE/ICCF must still be able to interpret. Now the year is stored as a one-byte binary number relative to 1900.

Field RTDDATE (D$ record)

The length of this field is five bytes and it contains the date in the format yymmdd.

Hexadecimal values for YY

```
       YY = 00   invalid
01 GE YY LE EF   1901 to 2139
F0 GE YY LE F4   1980 to 1984
F5 GE YY LE F9   1975 to 1979
FA GE YY LE FF   invalid
```

X'F5' (1975) represents the introduction of VSE/ICCF.

4.15 Interactive Interface and the Year2000

Leap year calculation

Several program have been changed to do proper leap year calculation. The affected programs are:

- IESXUPM: User profile maintenance
- IESBLDUP: Migration program

- IESIES01: Signon program
- IESXPWD: Change password program
- IESRETP: Route messages to VSE users

Expired password checking

To check for an expired password or a password that expires in n days, the sign-on program can now compare dates of two different centuries.

4.16 VSE/POWER

Date-related processing is based on the contents of the VSE/POWER partition COMREG field JOBDATWC that also contains the century information. The field JOBDATWC reflects either:

- the total system date set by the ′SET DATE′ IPL command, or
- the VSE/POWER partition singular date set by the ′// DATE mm/dd/yyyy′ job control statement.

In simulating a century switch, for example, either method may be used to change the current date. However, this should only be done on test systems, because all VSE/POWER queue entries remember their creation date as well as their TOD Store Clock time stamp, which determines the queueing sequence of a queue entry within its class chain.

4-Digit Year at Queue Entry Creation

Whenever job or output queue entries are created - even when received by PNET or loaded from tape - they are always branded with the current date, including the cc-century and yy-year. The only way to preserve an existing old creation date is to specify the NOJNO operand during POFFLOAD LOAD or SELECT.

You can display the creation date of queue entries with the FULL=YES option of the PDISPLAY command.

4-Digit Creation Year in Queue Record

The user-written exit routines JOBEXIT, OUTEXIT, NETEXIT, and XMTEXIT can access the internal VSE/POWER control block IPW$DQR. This control block is extended to contain a two byte century field on a currently reserved field. The field name is **QRDYC**:

```
offset x¢44¢  current   XL2¢00¢          reserved
offset x¢44¢  QRDYC     CL2¢CC¢          century of JOBDATWC
```

The dates in the NJE control records, which can also be used by user written POWER exits, are not changed. They contain the TOD Store Clock value which can be transformed to an exact time, date and century value.

4-Digit Year in Messages and List Output

All time and status messages as well as separator pages of list output show the date with a 4-digit year.

4-Digit Year in Interface Control Records

For programmed interface communication by Spool-Access CTL or GCM Service requests, VSE/POWER also presents a 4-digit year.

1. **Spool-Access Support Fixed Format Display Record (PWRSPL)**

   A two byte century field is introduced. The name of the field is **PFXMDATC**

```
offset x¢92¢  current   XL2¢00¢          reserved
offset x¢92¢  PXFMDATC  CL2¢CC¢          century of JOBDATWC
```

2. **Job Completion Message Record**

   A two byte century field is introduced. The name of the field is **JCMFECDC**

```
offset x¢44¢  current   CL12             reserved
offset x¢44¢  JCMFECDC  CL2¢CC¢          century of JOBDATWC
offset x¢46¢            CL10             reserved
```

Unique Century Identification in Account Records

For a precise evaluation of accounting data, VSE/POWER identifies the century by a flag in all types of account records. Thus, all account records keep their existing length and do not influence the running of user accounting programs.

These changes are object code transparent. Programs do not need recompilation unless the century information contained in these records will be used. The following records are changed:

1. **Account Record Header**

   The APF, EXEC, LIST, PUNCH, READER, and TRANS/RECV account records identify the century at X′2C′, where the flag byte **ACFLG1** is added. If the first bit in this byte is on, ACDATE is in 20yy format. If the first bit is off, ACDATE is in 19yy format.

```
offset X¢2C¢  current   CL1              reserved
offset X¢2C¢  ACFLG1    BL1¢0¢           ACCOUNT FLAG BYTE
              ACF1CE20  X¢80¢            ON  year ACDATE is 20yy
                                         OFF year ACDATE is 19yy
```

2. **RJE/BSC Account Record**

   At offset X′31′ a century flag byte, **BSCFLG1**, is added. If the first bit in this byte is on, BSCDTE is in the format 20yy. If this bit is off then BSCDTE is in the format 19yy.

```
offset X¢31¢  current   CL1              reserved
offset X¢31¢  BSCFLG1   BL1¢0¢           ACCOUNT FLAG BYTE
              BSC1CE20  X¢80¢            ON  year BSCDTE is 20yy
                                         OFF year BSCDTE is 19yy
```

3. **RJE/SNA Account Record**

   At offset X′28′ a century flag byte, **SNAFLG1**, is added. If the first bit in this byte is on, SNADTE is in the format 20yy. If this bit is off then SNADTE is in the format 19yy.

```
offset X¢28¢  current   CL2          reserved
offset X¢28¢  SNAFLG1   BL1¢0¢       ACCOUNT FLAG BYTE
              SNA1CE20  X¢80¢        ON  year SNADTE is 20yy
                                     OFF year SNADTE is 19yy
offset X¢29¢            CL1          reserved
```

4. **PNET Session Account Record**

   One remaining flag bit in **NETTERM** flag byte at offset X′2B′ is used to present the 21st century. If the last bit in this byte is on, NETSOD is in the format 20yy. If this bit is off, NETSOD is in the format 19yy.

```
offset X¢2B¢  NETTERM   BL1¢0¢       Signoff Code
offset X¢2B¢  NET1CE20  X¢01¢        ON  year NETSOD is 20yy
                                     OFF year NETSOD is 19yy
```

5. **VSE/POWER System-Up Account Record**

   At offset X′0C′ a century flag byte, **PWRFLG1**, is added. If the first bit in this byte is on, PWRDTE is in the format 20yy. If this bit is off then PWRDTE is in the format 19yy.

```
offset X¢0C¢  current   PL4¢0¢       reserved
offset X¢0C¢  PWRFLG1   CL1          STARTUP FLAG BYTE 1
              PWR1CE20  X¢80¢        ON  year PWRDTE is 20yy
                                     OFF year PWRDTE is 19yy
offset X¢0D¢            XL3¢0¢       reserved
```

6. **XCONN Spool-Access Connect Account Record**

   At offset X′21′ a century flag byte, **XCOFLG1**, is added. If the first bit in this byte is on, XCODATE is in the format 20yy. If this bit is off then XCODATE is in the format 19yy.

```
offset X¢21¢  current   CL1          unused
offset X¢21¢  XCOFLG1   CL1          SAS CONNECTION FLAG 1
              XCO1CE20  X¢80¢        ON  year XCODATE is 20yy
                                     OFF year XCODATE is 19yy
```

7. **XSPOOL Spool-Access Operation Account Record**

   At offset X′2C′ a century flag byte, **XSPFLG1**, is added. If the first bit in this byte is on, XSPDATE is in the format 20yy. If this bit is off then XSPDATE is in the format 19yy.

```
offset X¢2C¢  current   C¢ ¢         unused
offset X¢2C¢  XSPFLG1   CL1          SAS OPERATION FLAG 1
              XSP1CE20  X¢80¢        ON  year XSPDATE is 20yy
                                     OFF year XSPDATE is 19yy
```

Transparency for Old 2-Digit Creation Years

The Year2000-enabled VSE/POWER 6.1.2 may still encounter queue entries that have been built with a 2-digit creation year, namely when:

- A VSE/POWER 6.1.0 or 6.1.1 queue file is warm started with existing old queue entries.
- The VSE/POWER 6.1.2 system shares the queue and data files via Shared Spooling with a VSE/POWER 6.1.0 or 6.1.1 system.
- Any VSE/POWER pre-6.1.2 queue entry is loaded from tape with the NOJNO operand of the POFFLOAD LOAD/SELECT command.
- A PDISPLAY Tape command is started for any VSE/POWER pre-6.1.2 offload or DISP=T tape.

For these queue entries, any display or other date-related functions will temporarily expand the yy-creation year to a 4-digit year by applying the following **fix-88-window** rule:

```
if yy >  88,  then assume 19yy
if yy <= 88,  then assume 20yy.
```

Note: This rule has always been used for VSE/POWER's Time Event Scheduling support, when DUEDATE=mm/dd/yy is specified in the *$$ JOB statement.

Creation Date Selection on Queue Manipulation commands

The CRDATE operand of the PDISPLAY, PALTER, PDELETE, PHOLD, and PRELEASE commands allows to select queue entries by comparing their creation date against a limit date. Starting with VSE/POWER 6.1.2 the limit date may be specified as:

- mm/dd/yy with a 2-digit year. The year will be interpreted during comparison according to the fix-88-window rule.
- mm/dd/yyyy with a 4-digit year.

Additional Considerations

Whenever a VSE/ESA system is upgraded to the 2.2 level including the Year2000 support for VSE/POWER, it will cope with queue entries that were created prior to this release. VSE/POWER will apply the existing fixed window rule to extend the 2-digit year to a 4-digit year.

Sharing a spool file between a system that is not Year2000 enabled and a system that is Year2000 enabled will be possible. The un-enabled system will not recognize the CC fields in the records created by the upgraded system and will not process this information. On the other hand, the upgraded POWER may find queue records created by the old system and will apply the fix-88-window for these records.

It will also be possible to read tapes created by a POFFLOAD command on a older version of VSE/POWER. With the POFFLOAD BACKUP52/51/41 or SAVE52/51/41, VSE/POWER can be asked to produce spool tapes that can be ported to low level target systems. This will still be possible after the Year2000 upgrade.

4.17 REXX/VSE

The REXX/VSE function DATE returns valid results for years greater than, or equal to, the Year 2000. The DATE function also offers several output formats with 2-digit years. They will switch to '00' when crossing the Year 2000.

Besides retrieving the local date, REXX/VSE function DATE will allow to specify an input date in any REXX/VSE known format, which is converted into a given output format.

4.18 CICS/VSE Support for Year2000

CICS/VSE 2.3, included in VSE/ESA 1.4, VSE/ESA 2.1 and VSE/ESA 2.2 is the first CICS/VSE release to provide Year2000 support. The support consists of Application Programming Interface enhancements, plus internal support to allow for the change of the century.

4.18.1 Application Programming Interface (API)

This section outlines the enhancements to the CICS/VSE 2.3 API for system dates. To make use of these enhancements, changes to the application programs are required.

Command Level Programming

The CICS/VSE API allows access to the system time and date via two commands, EXEC CICS ASKTIME and EXEC CICS FORMATTIME.

- **EXEC CICS ASKTIME**

  If EXEC CICS ASKTIME is used without an option the EIBTIME and EIBDATE fields in the EIB are updated. EIBDATE is a four bytes packed decimal field and contains the date in the form 0CYYDDD+. C shows the century with values 0 for 19xx and 1 for 20xx. EIBDATE has the same format as CSAJYDP (the CSAJYDP format is shown in the macro level programming section below).

  The ABSTIME option is used to provide a data area where CICS/VSE places the time, in packed decimal, since 00:00:00 on January 1, 1900 (in milliseconds). In both cases the century information is already present in the date fields updated by CICS/VSE, so no changes are made to this command.

- **EXEC CICS FORMATTIME**

  EXEC CICS FORMATTIME transforms the absolute date and time into any of a variety of formats. Normally, it uses the value returned by the EXEC CICS ASKTIME ABSTIME command. FORMATTIME has been extended to provide four character year values in the following options:

  DDMMYYYY
  MMDDYYYY
  YEAR
  YYYYDDD
  YYYYDDMM
  YYYYMMDD

  You will have to use one of these options if you want to make use of the century information. The other options of EXEC CICS FORMATTIME still return a 2-digit year values.

Macro Level Programming

The programming interface provided for macro level programs to access the system date remains the field CSAJYDP in the control block DFHCSA. This field is in packed decimal format, with a length of four characters, and now takes the format:

0CYYDDD+

where:

C represents the century (0=1900,1=2000)
YY represents the year
DDD represents the day number
\+ represents the sign (always positive either X′F′ or X′C′)

For example, on the June 30, 1999 CSAJYDP will hold the value 0099181+ and on June 30, 2000 it will hold the value 0100182+.

Macro level applications that do not consider the century information in CSAJYDP must be changed if a 4-digit year is needed. You may also consider migrating these applications to the Command Level Interface.

4.18.2 CICS/VSE Internals

This section describes those areas of CICS/VSE where a date field is used internally by CICS/VSE and which are not always part of an external interface for the user. CICS/VSE uses the operating system services such as the COMREG macro to obtain the current date. If the century information is not correct in the partition COMREG, the century information in CSAJYDP will also be wrong.

- **Journaling date formats**

  Journal control records contain two date fields - JCLRVCD and JCLRDATE. These fields conform to the format described for CSAJYDP above. The fields appear in journal block label area records written to tape or disk. For more details refer to the manual *CICS for VSE/ESA Data Areas*.

  If you use the SMF format option on the journal definition, the date in the SMF header section of the journal records, field SMFDTE, will be in the CSAJYDP format.

- **Restart Data Set date formats**

  Two records in the Restart Data Set contain date fields. The first is stored in the CTL record that is updated on every warm keypoint. It is stored in CSAJYDP format. The second is used in the CSA record. It is also updated on every warm keypoint and contains the time and date in store-clock STCK format. This record is used during CICS/VSE startup to determine whether CICS/VSE is being initialized with a date/time prior to the date/time of the last quiesce. In this case message ′DFH1561 STARTUP TIME EARLIER THAN SHUTDOWN TIME′ is issued.

- **Report date formats**

  DFHSTUP produces reports based on automatic statistics collection and shows the date, for the interval being reported, with a 4-digit year.

- **CICS/VSE Message Transaction CMSG**

  The CICS-supplied transaction CMSG has been altered to allow for the delivery of a message that is to be issued in the next century. However, the

normal CMSG restrictions regarding the time delay for delivery still apply, that is, a message can only be scheduled for 100 hours in the future.

4.18.3 CICS/VSE 2-Digit Date Displays

This section outlines those areas in CICS/VSE 2.3 where a 2-digit date value is displayed to the user. This means that there is no century information displayed in the date field. In most cases, it is assumed that the actual date can be inferred from the context in which the date is displayed.

- **Master Terminal**

  All master terminal screens show the year as a 2-digit value on the screen.

- **Resource Definition Online**

  All Resource Definition Online screens and reports show a two character value for the year.

- **Transaction Dump Output**

  DFHDUP produces transaction dump printouts from dumps stored on the CICS/VSE dump data set. The printouts show the date of a transaction dump with a 2-digit date value.

- **Analyzed Dump Output**

  DFHDAP produces system dump printouts from dumps stored in the VSE/ESA dump library. The printouts show the date of a system dump with a 2-digit date value.

- **Report Controller Panels**

  Currently, CEMS/CEOS shows report selection and JCL report selection fields based on 2-digit date values. The Report Controller Feature has been updated to allow reports to be selected using FROM and TO dates that cross the century boundary. The panels will still allow only 2-digit year specifications, but the fixed 88 window, which is also used in VSE/POWER, will be used by the Report Controller Feature to interpret these 2-digit year specifications as a 4-digit year. This will ensure a correct report selection when crossing the Year 2000.

4.18.4 Conclusion

CICS/VSE 2.3, is Year2000 enabled except for the Report Controller Feature (RCF). The RCF requires PTF UN96558 to allow correct report selection after the century switch.

Application programs will require changes if they use the CICS/VSE API services to obtain the date, and they need a 4-digit year field.

Because CICS/VSE uses the date, retrieved from the operating system, internally for it′s logs and in the Restart Data Set, it is very important that this date is correct in the operating system. If the correct century information is not available in the partition COMREG, CICS/VSE may refuse to start with message DFH1561, and the journals may be corrupted when crossing to the next century.

Warning

Be aware that testing for Year2000 resolutions may cause CICS/VSE to not initialize after a SET DATE in the future is used. It can cause journal corruption also. This testing should never be done on a production system.

4.19 DL/I Support for the Year2000

DL/I's database functions handle all dates provided by an application program as simple binary data. There are no user-defined data formats, and there is no support for special contents such as date or time in predefined areas. It is the user who defines how a date is stored in a DL/I database and the user has to cope with the Year 2000 in his applications and database design.

Data within a DL/I database is routed to, or taken from, the user's input area without any conversion. The SENFLD coding in the PSB is the only exception to this. This conversion changes only a few values such as from hexadecimal to packed decimal. **It is not used to convert dates.**

At various places, DL/I keeps a date field that it uses internally. These date fields are extremely important for recovery.

Changes to the DL/I date format

DL/I keeps timestamps in the following places:

- Database creation date in control interval 0. This is written at database create or reload time.
- In the header record and detail records of the image copy file.
- In the header record and detail records of the database unload file.
- In each single DL/I log record (CICS/VSE journal record).
- In the change accumulation file as a creation timestamp and as a purge date entry.

In most cases the date is written as a three byte packed decimal field. The layout is X'YYDDDF', where YY represents the year, DDD the date and F is the positive sign.

This format has not been changed for compatibility reasons. So there will be no general date format change in DL/I.

The only exception to this rule of no changes to the date format is the header file of the image file created with the DLZUDMP0 utility. It previously had a four byte date field where the century value was 0. It will now contain the real century information.

The DSECT looks like this:

```
DUMPHDR      DSECT
DDATEOUT     DS    CL4 DATE of DUMP  old X¢00yydddF¢
                                       new X¢yyyydddF¢
```

DL/I′s internal handling of the century information

Almost all dates on tape or disk are in 2-digit values. Internally DL/I expands this value to a 4-digit year field by using a fixed window with a size of 50. This means:

yy = 50-99 = 19yy
yy = 00-49 = 20yy

Changes to Utility Control Statements

Several DL/I utilities have an optional date entry. For example, the Change Accumulation Utility DLZUCUM0 allows for the specification of a purge date. The layout of the control cards for these utilities has not been changed for Year2000. This means that the date is still coded in the old format YYDDDHHMM. The date will be internally converted to a 4-digit year using the same 50 year fixed window. This will not require any changes to existing JCL within job streams.

4.20 DB2 for VSE (SQL/DS 3.5) and Year2000

4.20.1 DB2 for VSE Application Programs

DB2 for VSE users who are using the DATE and TIMESTAMP data types defined in DB2 for VSE will not have any problem. DB2 for VSE has used 4-digit dates since 1988. If another data type to store the date is being used, there may be problems. Anyone using either a numeric or a character data type to store date values will run into problems if using 2-digit years, since the data will sort 00 at the wrong end of the collating sequence. The solution is to convert those columns and the applications that use them to a data type which supports the Year2000.

Example

```
 EMPLOYEE TABLE

EMPNO     CHAR(6)
FIRSTNME  VARCHAR(12)
LASTNAME  VARCHAR(15)
WORKDEPT  CHAR(3)
HIREDATE  DATE
SALARY    DECIMAL(9,2)
BIRTHDATE DATE
```

When dates are defined as shown in the table example above, they are stored as a string of four bytes. Each byte is two packed decimal digits. The first two bytes are the year, the next byte is the month and the last byte is the day. The year is already a 4-digit value. There will be no problems with tables or applications using this data type.

Let us look at the case when the same table is defined as shown on the following page:

```
EMPLOYEE TABLE

EMPNO     CHAR(6)
FIRSTNME  VARCHAR(12)
LASTNAME  VARCHAR(15)
WORKDEPT  CHAR(3)
HIREDATE  CHAR(8)
SALARY    DECIMAL(9,2)
BIRTHDATE CHAR(8)
```

If HIREDATE and BIRTHDATE are stored in the format mm/dd/yy, there is a problem. The columns HIREDATE and BIRTHDATE should be changed to allow for a 4-digit year and the application program using this table should be adapted to cope with these changes. However, the best solution in this case is to change the columns HIREDATE and BIRTHDATE to a data type DATE. In this case the applications would need to be changed to handle the new data type.

4.20.2 General Use of Dates in DB2 for VSE

DB2 for VSE uses a fixed window with a size of 42. The year is assumed to be in 20xx if it is less than 43. Otherwise, it will be assumed to be 19xx. This means that dates before 1943 cannot be given to filtered log recovery or to the Trace Formatter, as these dates will be assumed to be in the 21st century. This is not a problem because there are no traces or logs containing these years, and they will never be produced.

In DB2 for VSE the window rule applies to dates displayed by:

- The filtered log recovery process
- The Trace Formatter
- DBSU output messages
- ISQL printed output
- SQL EXEC terminal output
- DB2 for VSE messages
- CICS/VSE transactions CIRx command output
- Operator command responses, including those returned to an ISQL session or to a program via the RDIIN interface, such as RXSQL.

Filtered log recovery

Filtered log recovery accepts control cards specifying dates and times to delimit the range of its actions. These dates and times are converted to TOD values for comparison to the TOD values in the log. Filtered log recovery parses the input date and time control records, but does not syntax check them. This is done by another module that accepts a 2-digit year and uses the fixed-42 window to convert them to 4-digit years, so no changes are required.

Trace Formatter

The Trace Formatter accepts control cards specifying dates and times to determine the trace records that are selected for formatting and printing. These date values are compared to the TOD values from the trace records after both are converted to a format yymmdd. The Trace Formatter also uses the fixed window of 42 to do the conversion.

Due to the way the Trace Formatter compares dates with the trace records, it is not possible to select a date range that crosses the century boundary. Trace files that span the century boundary must be formatted with two executions of the Trace Formatter. The same restriction exists for time ranges spanning midnight.

Accounting

Accounting is the only area in which a change will be made in the next release. DB2 for VSE will place the century information in the accounting records it generates. While not strictly necessary, it is felt that placing the century information in the accounting records now will give users plenty of time before the turn of the century to adjust their accounting procedures to exploit the century number.

Even though the accounting record format is changing, this will be transparent for user-written applications. This is because the century is being placed into a previously reserved field in the accounting record.

4.21 DOS/VS RPG II

Current Systems

- Language: Operations TIME, UDATE and UYEAR can be used to get the current date. They deliver a 2-digit year.
- Compiler: Listings contain an execution date with a 2-digit year.
- RPG II program execution: No support is available to convert a 2-digit year to a 4-digit year.

Year2000 Enabled Systems

Because of compatibility reasons, the names and length of the operations TIME, UDATE and UYEAR cannot be changed. Compiler listings will show the current year in 2-digit format. This is not considered to be a problem.

For program execution, a conversion routine, ILNY224, is provided, which supports translation of a 2-digit field to a 4-digit field. It can extend current dates, and also supports translation of 2-digit fields representing any kind of year within data records.

This subroutine makes use of the YEAR224 macro and subroutine IJBY224, that are provided by VSE/ESA 1.4 and higher.

IJBY224 will be shipped with RPG II code under the name of ILNYCONV. Modules ILNY224 and ILNYCONV are linked to RPGOBJ during link editing. The RPG programmer specifies a window between 0 and 99, which is interpreted as the forward width of a 100 year window relative to the current year.

In 5.13, “DOS/VS RPG II Considerations” on page 92 you will find an example of an RPG II program using this new support.

4.22 CSP/AE 3.3.0 and CSP/AD 3.3.0

CSP/AE 3.3.0 and CSP/AD 3.3.0 are not Year2000 enabled and there are no plans to enhance the products. They are Year2000 tolerant. This means that they will ensure that the correct values are obtained for a 2-digit year. For example, EZEDAY and EZEDTE will be 00 for the year 2000 and 05 for the year 2005. The user must add coding to his programs to interpret 2-digit years according to the logic of the programs.

VisualGen Host Services provides a migration path for current CSP/AE and CSP/AD users. With VisualGen Host Services 1.1 you can prepare, install and run generated COBOL applications. This product is Year2000 enabled. The redbook *VisualGen: The Future of Your VSE Applications And How to Get There from CSP* will help you to prepare the move to a VisualGen based application development environment.

4.23 SDF/CICS for VSE and Year2000

Screen Definition Facility/CICS Version 1 Release 5 uses system dates and creates dates to log ongoing SDF/CICS user activities. These log records are compared during startup of SDF/CICS user sessions to determine whether to perform a warm or a cold start of a previous SDF user session.

The move to the next century, therefore, only requires to drop SDF/CICS session log records by the end of 1999 and start them again in the year 2000. Thus the restriction applies that no SDF/CICS session should be active over midnight between December 31, 1999 and January 1, 2000.

In its output, on panels and in print output, SDF/CICS will still use 2-digit date presentation. There are no plans to change this.

4.24 DFSORT/VSE and Year2000

Overview

The PTF for Apar PN81797 will enhance DFSORT/VSE′s Year2000 capabilities by providing the ability to sort, merge and transform dates with 2-digit years according to a specified fixed or sliding century window. New Y2C, Y2Z, Y2P and Y2D formats, in conjunction with a new Y2PAST installation and run-time option, will allow for handling 2-digit year data in the following ways:

- Set the appropriate century window for your applications. For example, set a century window of 1915-2014 or 1950-2049.
- Order 2-digit character, zoned decimal, packed decimal or decimal year data, according to the century window, using DFSORT/VSE′s SORT and MERGE control statements. For example, order 96 (representing 1996) before 00 (representing 2000) in ascending sequence, or order 00 before 96 in descending sequence.
- Transform two-digit character, zoned decimal, packed decimal or decimal year data to four-digit character year data, according to the century window, using DFSORT/VSE′s OUTREC control statement. For example, transform 99 to 1999 and 04 to 2004.

DFSORT/VSE will also provide new formats you can use to order and transform parts of packed decimal fields (for example, month and day in a date field) with the SORT, MERGE and OUTREC control statements.

The Y2PAST option

A new installation and run-time option will allow you to specify a fixed or sliding century window to be used with 2-digit years. A century window spans a 100 year period and is used to control how the 2-digit years are interpreted.

Y2PAST=f specifies a fixed century window starting at f. For example, Y2PAST=1985 starts the century window at 1985 (1985-2084).

Y2PAST=s specifies a sliding window starting s years before the current year. For example, if the current year is 1996, Y2PAST=80 starts the century window at 1996-80=1916, providing a century window of 1916-2015. In 1997, this window automatically slides to 1917-2016.

As an example, both Y2PAST=1995 and Y2PAST=81 used in 1996 give a century window of 1915-2014, which results in the following interpretation of 2-digit years by DFSORT/VSE:

YY	Interpreted as
00	2000
14	2014
15	1915
61	1961
62	1962
99	1999

2-digit year formats

DFSORT/VSE will allow you to use new Y2x formats to identify your 2-digit character, zoned decimal, packed decimal, and decimal year data. DFSORT/VSE will correctly sort, merge and transform 2-digit years identified with these formats as if they where 4-digit years, using the fixed or sliding window specified in the Y2PAST option. The Y2x formats are:

- Y2C or Y2Z which identify 2-digit, two-byte character or zoned decimal year data in many types of dates such as C′yy′, C′mm/dd/yy′, C′yy.mm.dd′, Z′ddmmyy′, Z′yyddd′, C′dddyy′ or C′yy-mm′.
- Y2P which identifies 2-digit, two-byte packed decimal year data in many types of dates such as P′yy′, P′mmddyy′, P′yymmdd′, P′ddmmyy′, P′dddyy′ or P′mmyy′.
- Y2D which identifies 2-digit, one-byte decimal year data in many types of dates such as X′yy′ or P′yyddd′.

Sorting and merging 2-digit years

You can use the new 2-digit year formats in DFSORT/VSE′s SORT and MERGE statements to identify 2-digit year data to be ordered according to the century window. We can show this with a simple example for sorting date fields C′mm/dd/yy′ in ascending order:

Assume that the job with the following SORT statements is run in 1996.

```
OPTION Y2PAST=34          *Set 1962-2061 as the window
SORT FIELDS=(7,2,Y2C,A,   *Sort C¢yy¢ as C¢yyyy¢ using the window
             1,2,CH,A     *Sort C¢mm¢
             4,2,CH,A)    *Sort C¢dd¢
```

It will give the following result:

Input	Output
06/22/15	03/18/62
10/03/00	09/01/99
11/14/61	10/03/00
08/16/14	08/16/14
09/01/99	08/17/14
03/18/62	06/22/15
08/17/14	11/14/61

Transforming 2-digit years to 4-digit years

You can use the new 2-digit year formats in DFSORT/VSE's OUTREC statement to identify 2-digit year data to be changed to 4-digit year data according to the century window. You can transform the date fields while sorting, merging or copying. We can show this with a simple example to transform a date field C'yyddd' while copying:

```
OPTION Y2PAST=1970        *Set 1970-2069 as the window
SORT FIELDS=(COPY)        *Copy-output is unsorted
OUTREC=(1,2,Y2C,          *Change C¢yy¢ to C¢yyyy¢ using the window
       C¢/¢,              *Insert a ¢/¢
       3,3)               *Copy C¢ddd¢
```

This will give the following result:

Input	Output
92012	1992/012
70225	1970/225
69153	2069/153
00001	2000/001
99321	1999/321
12054	2012/054

Conclusion

The DFSORT/VSE Year2000 enhancements will allow for continuing use of 2-digit year dates for sorting and merging, and help to change from using 2-digit year dates to using 4-digit year dates.

4.25 DITTO/ESA for VSE and Year2000

In general, DITTO/ESA is affected by the Year2000 as far as **creation dates** and **expiration dates** are concerned and by the date fields in panels and output listings.

4.25.1 DITTO/ESA for VSE Release 1 and Release 2

DITTO/ESA for VSE Release 2 provides support for the Year2000. It runs on VSE/ESA Version 2 and subsequent releases. Dates are displayed in full 4-digit year representation. If the operating system does not provide 4-digit date values, DITTO/ESA will apply a 50 year fixed window.

DITTO/ESA for VSE Release 1 has equivalent Year2000 support as provided by DITTO/ESA for VSE Release 2. It is provided by an enhancement PTF.

4.25.2 DITTO 3.2 for VSE and Productivity Feature for VSE

Year2000 support is added by a PTF and is based on the same technique as described for DITTO/ESA for VSE Release 2. Thus dates will be sorted correctly after applying the PTF. However, dates on panels and in title lines on printed output will continue to print or display 2-digit years.

4.26 PSF/VSE Version 2.2.0 and 2.2.1

PSF/VSE is affected by the Year2000 in the following areas:

- The AFP Accounting record has a date in the beginning of the record in the format of mm/dd/yy or dd/mm/yy.
- The current date printed on the print job separator pages is in the format mm/dd/yy or dd/mm/yy.
- The date that a resource was marked by the utility program APTRMARK displays as YY-MM-DD, when the REPORT option is used with this utility program.

The following changes are made to PSF/VSE to support Year2000.

- There is no room in the AFP Account record to expand the YY field at bytes 7 and 8 to a four bytes YYYY field. Whether the YY field represents 19YY or 20YY will be indicated in a currently reserved byte at X′2D′. The year will be 19YY if this byte contains X′00′, and it will be 20YY if this byte contains X′80′.
- The mm/dd/yy or dd/mm/yy on the print job separator pages will be expanded to mm/dd/yyyy or dd/mm/yyyy. Each field following this date field will be shifted two bytes further to the right edge of the page.
- The YY-MM-DD currently displayed by the APTRMARK utility when the REPORT option is used will be expanded to YYYY-MM-DD. Each field following this date will be shifted two bytes to the right. The process that ′marks′ the resources already accounts for Year2000. No additional changes are required to this process.

Because PSF/VSE can also run on VSE-compatible operating systems it will not use the VSE/ESA operating system facilities to determine the century information. A fixed window with a size of 86 is applied. The changes mentioned above are provided via a PTF.

4.27 NetView 2.3

NetView has been changed to cope with the Year2000 in hardware monitoring (NPDA) and session monitoring (NLDM) database records and commands. Without these changes NLDM and NPDA users may experience multiple problems after December 31, 1999. This support is provided by a PTF.

Chapter 5. Languages and the Year 2000

5.1 Overview

The languages Year2000 problem is also 2-digit year data. At the turn of the century the current year will be less than the previous year because the date field only has the last two digits of the year.

Both short-term and long-term solutions require that the application programming languages be migrated to a full 4-digit year format supporting language. The languages that support 4-digit year formats are:

1. COBOL/VSE
2. PL/I VSE
3. C/VSE
4. High Level Assembler (HLASM)

There is no 4-digit year date support with DOS/VS COBOL or VS COBOL II irrespective of whether the VSE/ESA system has been YEAR2000 enabled or not. This is also true for DOS PL/I and C/370.

The languages COBOL/VSE, PL/I VSE, and C/VSE fully support 4-digit dates, as does their prerequisite Language Environment for VSE (LE/VSE). LE/VSE provides callable date/time service routines for date conversion.

VS COBOL II programs may make calls to LE/VSE date/time service routines, although there will be no support internally for the year 2000.

DOS/VS RPG II

For DOS/VS RPG II, the date operands of UYEAR and UDATE will only return a 2-digit year.

RPG II programmers may use the newly supplied subroutine ILNY224 to convert 2-digit year date fields to 4-digit date fields, or implement their own technique as required. Please refer to the section 4.21, “DOS/VS RPG II” on page 45 for details on the provided support and 5.13, “DOS/VS RPG II Considerations” on page 92 for a usage example.

VS FORTRAN

VS FORTRAN does not provide any date/time functions, and therefore the compiler does not have a problem with 2-digit years. The compiler listing shows a 2-digit year currently. A PTF is provided to print 4-digit years on the output listings.

5.2 Language Environment/VSE Considerations

IBM Language Environment for VSE/ESA (LE/VSE) is a set of common services and language-specific routines that provide a single run-time environment for applications written in LE/VSE conforming versions of the COBOL, PL/I and C high level languages. An LE/VSE conforming language is one that adheres to the LE/VSE common interface.

Table 2 lists the LE/VSE conforming language compiler products you can use to generate applications that run with LE/VSE.

Table 2. LE/VSE-Conforming Languages

Language	LE/VSE-Conforming Language	Minimum Release
COBOL/VSE	IBM COBOL for VSE/ESA	Release 1
PL/I VSE	IBM PL/I for VSE/ESA	Release 1
C/VSE *	IBM C for VSE/ESA	Release 1
Note: Applications written in C/VSE can only run with LE/VSE Release 4.		

Any High Level Language (HLL) not listed in Table 2 is known as a non-LE/VSE-conforming or, alternatively, a pre-LE/VSE-conforming language. Some examples of non-LE/VSE-conforming languages are: C/370, DOS/VS COBOL, VS COBOL II, and DOS PL/I.

Only the following products can generate applications that run with LE/VSE:

- LE/VSE-conforming languages
- High Level Assembler (HLASM)

  LE/VSE also supports applications written in Assembler language using LE/VSE-provided macros and assembled using HLASM.

5.3 LE/VSE Date/Time Callable Services

LE/VSE includes a complete set of callable services that enhance the capabilities of High Level Languages to perform date and time calculations. You can use date and time services to read, calculate, and write values representing the date and time. LE/VSE offers unique pattern matching capabilities permitting you to process almost any date and time format contained in an input record or produced by the operating system services.

You can use date and time services to:

- Format date and time values by country code
- Format date and time values using custom formats
- Parse date values and time values
- Convert between Gregorian, Julian, Asian and Lilian formats
- Calculate days between dates
- Calculate elapsed time to the nearest millisecond
- Obtain local time and *Greenwich Mean Time* (GMT) from the system without a *supervisor call* (SVC) overhead
- Properly handle 2-digit years in the year 2000.

See Table 3 on page 53 for a list of LE/VSE time/date callable services. For a full explanation of these LE/VSE callable services refer to the manual *LE for VSE Programming Guide*.

Routine	Explanation
CEEDATE	Convert Lilian Date to Character format
CEEDTAM	Convert seconds to character timestamp
CEEDAYS	Convert date to Lilian format
CEEDYWK	Calculate day of week from Lilian date
CEEGMT	Get current Greenwich Mean Time
CEEGMTO	Get offset from Greenwich Mean time to local time
CEEISEC	Convert integers to seconds
CEELOCT	Get current local time
CEEQCEN	Query century window
CEESCEN	Set century window
CEESECI	Convert seconds to integers
CEESECS	Convert timestamp to number of seconds

Table 3. Language Environment Callable Service Routines

Definition of referenced dates:

- Lilian Date

  The Lilian date is the number of days since 14 October 1582.

- Gregorian Date

  The Gregorian date is the number of days since 31 December 1600.

5.4 The LE/VSE COUNTRY Code Option and the CEE5CTY Callable Service

LE/VSE provides a run-time option, COUNTRY, and a callable service, CEE5CTY, which set the default country code.

The country code affects the formats of date and time, the currency symbol, decimal separator, and the thousands separator.

You may need to change the format of these items to suit the standards of the country for which you are programming. You do this by using the COUNTRY option or the CEE5CTY callable service.

Therefore, when performing arithmetic on dates, particularly when manipulating dates to solve your Year2000 problem, ensure you code the date fields in the correct format for the default country code in use.

The IBM-supplied country code defaults are listed in Appendix A of *IBM Language Environment for VSE/ESA Release 4*. The installation-wide default for your installation will have been set when LE/VSE was installed. The COUNTRY run-time option and the CEE5CTY callable service are also described in this book.

The setting of the COUNTRY code run-time option also affects the format of the date and time in the options and storage reports generated by the RPTOPTS and

RPTSTG run-time options. If the default country code has a date format with a 4-digit year, the date in these reports will appear with a 4-digit year. If the default country code has a date format with a 2-digit year, the date will appear with a 2-digit year.

For example, the US country code has the date format, **MM/DD/YY**, therefore the date on the options report produced on 15 January 2000 will be **01/15/00**.

However, if you live in Germany and have set your country code to DE, which has a date format of **DD.MM.YYYY**, the date on the report will be **15.01.2000**.

Note: Changing the country code setting does not change the default settings for the picture string set by CEESETL or *setlocale()*. The country code affects only the LE/VSE national language support services, not the LE/VSE locale callable services.

5.5 COBOL Year2000 Handling

5.5.1 COBOL Solutions

COBOL/VSE is the newest VSE COBOL compiler and is a direct descendant from VS COBOL II and DOS/VS COBOL. The major differences between these compilers are:

- DOS/VS COBOL supports ANSI 68 and ANSI 74 Standard
- VS COBOL II supports ANSI 85 Standard
- COBOL/VSE supports ANSI 85 Standard with the ANSI 85 Addendum Intrinsic Functions.

To be able to make full use of 4-digit date routines and to take advantage of the LE/VSE date/time callable service routines (sliding window techniques) or the COBOL/VSE intrinsic functions that support 4-digit year, you should plan to migrate your application programs to COBOL/VSE.

The tools available to help in your migration effort are:

- COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)
- COBOL Report Writer Precompiler

CCCA/VSE is described in detail in Chapter 7, “COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)” on page 105.

If you are currently on DOS/VS COBOL, the function CURRENT-DATE will only return a 2-digit year to your program, and this will not be changed. After migration to COBOL/VSE, the intrinsic functions of:

- CURRENT-DATE
- DATE-OF-INTEGER
- DAY-OF-INTEGER
- INTEGER-OF-DATE
- INTEGER-OF-DAY

will allow you to manipulate date fields, and perform calculations using dates, with full 4-digit year capability.

If you prefer, because COBOL/VSE is an LE/VSE conforming language, you may make use of the LE/VSE century window.

The LE/VSE century window is a 100 year sliding window that is changeable at any time within a program or in an application where the current year is somewhere within the window. The callable service routines interprets the 2-digit year as a 4-digit year.

These LE/VSE callable service routines allow application programs to run in the interim without making modifications to databases and files. The advantage to this is that only application program code needs to be changed, and modifications to actual files and databases can be postponed until time permits.

VS COBOL II programs DYNAMICALLY calling LE/VSE date/time service routines will work. However, DYNAMIC calls from VS COBOL II programs to non LE/VSE date/time callable service routines are **not** supported, and STATIC calls from VS COBOL II programs to any LE/VSE callable service routines are also **not** supported.

5.5.2 DOS/VS COBOL (5746-CB1) Considerations

Programs that are written in DOS/VS COBOL do not support a 4-digit year format, nor can they use the services of LE/VSE date/time callable routines.

The date returned to the application program is in the format mm/dd/yy (month,day,year), or dd/mm/yy (day,month,year).

It would be possible to apply the fixed window technique as shown in Figure 3 on page 56 and Figure 4 on page 60 from within the DOS/VS COBOL program, or by calling a subroutine or subprogram written in High Level Assembler invoking the supplied YEAR224 macro with the sliding window to obtain century information for use within the application. However, this would mean that every program would need to be changed that references or requires manipulation of date fields for century information.

It would be more practical that the work effort involved in program alterations be directed towards migrating the DOS/VS COBOL programs to COBOL/VSE where full century information is available either by intrinsic functions or by use of LE/VSE date/time callable service routines.

In the following example of a DOS/VS COBOL program, there is a number of date fields. These date fields are stored in the format of DDMMYY or MMDDYY on the magnetic tape files and do not carry century information. The dates may or may not be required to be converted to full century/year format YYYY depending upon the logic and requirement of the application.

```
IDENTIFICATION DIVISION.
PROGRAM-ID. BANKA.                BANK RECONCILIATION
                                    BALANCE REPORT
AUTHOR. A NAME.
REMARKS.
     .
     .
*
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SPECIAL-NAMES.
    C01 is CHANNEL-1
    .
    .
    C12 is CHANNEL-12.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
     SELECT OPTIONAL BR0200-DRAWNCHQS
         ASSIGN TO SYS008-UT-3420-S-BR0200.
     SELECT OPTIONAL BR0300-PRESENTCHQS
         ASSIGN TO SYS009-UT-3420-S-BR0300.
     SELECT BALANCE-REPORT
         ASSIGN TO SYS010-UR-3203-S.
     EJECT
DATA-DIVISION.
FILE SECTION.
FD BR0200-DRAWNCHQS
   .
   .
   LABEL RECORDS ARE STANDARD.
01 IBM-CHQ-REC-1                PIC X(24).
*
FD BR0300-PRESENTCHQS
   .
   .
   LABEL RECORDS ARE STANDARD.
01 IBM-CHQ-REC-2                PIC X(24).
*
FD BALANCE-REPORT
   RECORDING MODE IS F
   LABEL RECORDS ARE OMITTED.
01 REPORT-LINE                  PIC X(133).
EJECT
WORKING-STORAGE SECTION.
77       IBM-VERSION            PIC X(4)      VALUE ¢#004¢.
77       LINE-COUNT             PIC 99        VALUE ZEROES.
77       PAGE-COUNT             PIC 999       VALUE ZEROES.
77       ACCUM-AMOUNT           PIC 9(7)V99   VALUE ZEROES.
77       COUNT-RECORDS          PIC 9         VALUE ZEROES.
77       PREV-ACCOUNT           PIC X         VALUE SPACES.
```

Figure 3 (Part 1 of 4). Batch DOS/VS COBOL Program Example

```
*
*         A
01 DETAIL-LINE.
   05    FILLER                 PIC X.
   05    FILLER                 PIC X(13).
   05    CHQ-DATE               PIC X(8).                    1
   .
   .
01 TOTAL-LINE REDEFINES DETAIL LINE.
   05    FILLER                 PIC X.
   .
   .
01 HEADINGS-1.
   .
   .
01 HEADINGS-2.
   .
   .
   05    HEADING-DATE           PIC X(8).                    2
   .
   .
01 HEADINGS-3.
   05    FILLER                 PIC X        VALUE SPACE.
   .
   .
01 HEADINGS-4.
   .
   05    FILLER                 PIC X(33)    VALUE
      ¢DATE        CHEQUE NUMBER       ¢.
   .
01 WS-PRINT-DATE.
   05    HEADING-DATE           PIC X(8).                    3
   05    PDAY                   PIC 99.
   05    FILLER                 PIC X        VALUE ¢/¢
   05    MONTH                  PIC 99.
   05    FILLER                 PIC X        VALUE ¢/¢
   05    YEAR                   PIC 99.                      4
01 WS-DATE.
   05    DAYE                   PIC 99.
   05    MUNTH                  PIC 99.
   05    YAER                   PIC 99.                      5
01 DAYTE REDEFINES WS-DATE      PIC 9(6).
*
*  RECORD DEFINITIONS OF INPUT FILES
*
01 CHQ-REC-1.
   05    REC-CODE-1             PIC X.
   .
   05    DATE-DRAWN             PIC 9(6).                    6
   .
01 CHQ-REC-2.
   05    REC-CODE-2             PIC X.
   .
   05    DATE-PAID              PIC 9(6).                    7
   .
   .
```

Figure 3 (Part 2 of 4). Batch DOS/VS COBOL Program Example

```
PROCEDURE DIVISION.
000-CONTROL SECTION.
001-CNTRL.
    PERFORM 100-DRAWN-REPORT.
    PERFORM 200-PRESENTED REPORT.
    STOP RUN.
100-DRAWN-REPORT SECTION.
101-INITIALIZE.
    OPEN INPUT  BR0200-DRAWNCHQS
         OUTPUT BALANCE-REPORT.
    .
102-WORK-CNTRL.
    .
    ADD 1 to COUNT-RECORDS.
    .
    .
    MOVE DATE-DRAWN TO WS-DATE.                              8
    MOVE DAYE TO PDAY.                                       9
    MOVE MUNTH TO MONTH.                                     10
    MOVE YAER TO YEAR.           B                 11
*                                C
    MOVE WS-PRINT-DATE TO CHQ-DATE.                          12
    .
    .
108-CLOSE-FILES.
    CLOSE BR0200-DRAWNCHQS.
    CLOSE BALANCE-REPORT.
109-DRAWNCHQS-EXIT.
    EXIT.
200-PRESENTED-REPORT-SECTION.
201-INITIALIZE.
    OPEN INPUT BR0300-PRESENTCHQS INTO CHQ-REC-2
         OUTPUT BALANCE-REPORT.
    .
    .
202-WORK-CNTRL.
    READ BR0300-PRESENTCHQS INTO CHQ-REC-2
         AT END PERFORM 400-TOTALS
                GO TO 208-CLOSE-FILES.
    .
    MOVE DATE-PAID TO WS-DATE.                                 13
    MOVE DAYE TO PDAY.                                         14
    MOVE MUNTH TO MONTH.                                       15
    MOVE YAER TO YEAR.               D             16
*                                    E
    MOVE WS-PRINT-DATE TO CHQ-DATE.                            17
    .
208-CLOSE-FILES.
    CLOSE BR0300-PRESENTCHQS.
    CLOSE BALANCE-REPORT.
209-PRESENTCHQS-EXIT.
    EXIT.
```

Figure 3 (Part 3 of 4). Batch DOS/VS COBOL Program Example

```
300-HEADINGS SECTION.
301-HEAD.
    .
    MOVE CURRENT-DATE TO HEADING-DATE
    WRITE REPORT-LINE FROM HEADINGS-1 BEFORE ADVANCING 2.
    .
309-HEADINGS-EXIT.
    EXIT.
400-TOTALS SECTION.
401-TOT.
    .
409-TOTALS-EXIT.
    EXIT.
END-OF-JOB.
```

Figure 3 (Part 4 of 4). Batch DOS/VS COBOL Program Example

1 The data field CHQ-DATE would contain the cheque date in the format of either DDMMYY or MMDDYY and may be converted to full century/ year information at the turn of the century to show the date that the cheque was drawn in the format of MMDDYYYY or DDMMYYYY according to organizational standards.

2 The HEADING-DATE to be printed on the report is created or loaded from CURRENT-DATE within the DOS/VS COBOL program. CURRENT-DATE is obtained from the partition COMREG and irrespective of the VSE system being enabled or not, the date obtained for current date will not contain century information. Therefore the heading on the report would show either DD/MM/YY or MM/DD/YY according to standard options used at IPL time. The format of the COMREG is shown in Figure 17 on page 161 and would contain the date as loaded at partition initialization time or as loaded from a // DATE card within job control.

The date occupies bytes 0-7 within the COMREG.

In the Year 2000 the YY portion of the date would be 00.

3 The explanation as shown in 2 applies to this field.

4 The YEAR portion of the structure WS-PRINT-DATE would show 00 in the Year 2000. Depending on the need within the organization, this field could be converted to full century/year format. However, changing this field would mean further changes to the application program in that the report and data structure would require redesign to cater to the additional two bytes of information. A further complication would be the dependency between data structures WS-PRINT-DATE and WS-DATE.

5 See the explanation for item 4 for an explanation of this field and the data dependencies.

6 7 Both these fields may need to be converted to full century/year format, and require redesign of the data files and the format of the report to cater for the additional bytes for century.

8 9 10 11 12 If there were changes made to any of these fields, data dependencies would perhaps cause alterations to be made to all of these fields.

13 14 15 16 17 The same applies here - any alteration would have an affect on other fields within the data structures.

5.5.2.1 Conversion Techniques for DOS/VS COBOL Programs

There is a number of methods of determining century information from within DOS/VS COBOL programs.

See Figure 4 for a coding example of using the fixed window technique. This is an example only, and has been tested for a number of fixed windows to obtain century information.

To use a sliding window within DOS/VS COBOL would require a callable program using the YEAR224 macro written in the Assembler language, or calling routines developed by Software Solution Vendors.

```
WORKING-STORAGE SECTION.
77       IBM-VERSION            PIC X(4)      VALUE ¢#004¢.
77       LINE-COUNT             PIC 99        VALUE ZEROES.
77       PAGE-COUNT             PIC 999       VALUE ZEROES.
77       ACCUM-AMOUNT           PIC 9(7)V99   VALUE ZEROES.
77       COUNT-RECORDS          PIC 9         VALUE ZEROES.
77       PREV-ACCOUNT           PIC X         VALUE SPACES.
77       YYYY                   PIC 9999      VALUE ZEROES.    F
77       TEST-YEAR              PIC 99        VALUE ZEROES.    18
77       PAST-YEAR              PIC 99        VALUE ZEROES.    19
500-DATE-CONVERT SECTION.
501-YYYY
    MOVE YYYY TO TEST-YEAR.                                    20
    MOVE 60 TO PAST-YEAR                                       21
    IF TEST-YEAR > PAST-YEAR                                   22
    THEN ADD 1900 TO TEST-YEAR                                 23
    ELSE ADD 2000 TO TEST-YEAR.                                24
    EXIT.
```

Figure 4. A Fixed Window to Determine Century in DOS/VS COBOL

18 20 This data storage structure contains the year information to be converted to full century/year notation in the format YYYY according to organizational standards.

19 21 These instructions set up the window in the range of 1960-2031. In this case the organization has decided that there will be 35 years in the past, and 64 years in the future. and that dates prior to 1/1/60 are in the 21st century.

This field is adaptable according to each organization's needs.

22 23 24 These instructions determine that if the year obtained in 20 was greater than the future window (1960 in this case) then the century would be 19, otherwise it would 20.

F 18 19 These working storage instructions could be placed into the program described in Figure 3 on page 56, at position A , and an instruction:

MOVE YEAR TO YYYY

PERFORM 500-DATE CONVERT

could be inserted in positions B C and E .

This would, in theory, convert the dates to the full format of century/year for calculation or printing purposes.

Additional coding would be required to move the resultant converted dates to the WS-PRINT-DATE structure as required. This coding is not shown here.

Running tests against this subroutine produced the following results:

TEST YEAR	FIXED WINDOW	FULL YEAR NOTATION	CURRENT-DATE:01/02/00
45	60	2045	
98	60	1998	
61	60	1961	
60	60	2060	
00	60	2000	
05	60	2005	
59	60	2059	

Figure 5. Fixed Window Test Results for DOS/VS COBOL

The fixed window could be any value as determined by the requirements of your organization. Naturally, you could have a number of different fixed windows according to each application′s requirements. In this case, you would need to provide the logic to change the fixed window value.

5.6 VS COBOL II Considerations (5668-958)

As stated earlier, VS COBOL II programs DYNAMICALLY calling LE/VSE date/time callable service routines will work. However, DYNAMIC calls from VS COBOL II programs to non date/time Language Environment callable service routines are **not** supported, and STATIC calls from VS COBOL II programs to any LE/VSE callable service routines are also **not** supported.

There is also no 4-digit support provided with COBOL II.

Programs that have been written in DOS/VS COBOL and then migrated to VS COBOL II, or programs written in VS COBOL II should be migrated to COBOL/VSE to take full advantage of Year2000 support. The effort involved in migrating from VS COBOL II to COBOL/VSE is described in the manual *COBOL for VSE/ESA Migration Guide*.

Migration from DOS/VS COBOL to VS COBOL II is not a viable migration path. After understanding the implications of remaining on a DOS/VS COBOL platform and migration away from DOS/VS COBOL is planned, the correct migration scenario is to COBOL/VSE.

5.7 COBOL/VSE Considerations (5686-068)

COBOL/VSE is IBM′s strategic COBOL compiler for the VSE/ESA operating system. COBOL/VSE is a superset of DOS/VS COBOL and VS COBOL II, with additional features such as intrinsic functions and support for a common run-time environment (LE/VSE).

COBOL/VSE is the only version of COBOL to have intrinsic function support for date/time services.

5.7.1 LE/VSE and COBOL

The COBOL environment is complex due to the number of supported products and the sharing of module and phase names. The supported products are:

- DOS/VS COBOL 1.3.1
- VS COBOL II (till end of 1997)
- COBOL/VSE (compiler only)
- LE/VSE (run time only)

The following table shows the valid combinations of COBOL products and the potential conflicts of module and phase names when multiple products are concurrently installed.

Table 4. Valid Scenarios for COBOL Compiler, Link-Edit, and Run Time

If you compile with this compiler...	And you link-edit with this run-time library...	Then you can run with this run-time library...
DOS/VS COBOL FCOB	DOS/VS COBOL ILB	DOS/VS COBOL *
		VS COBOL II *
		LE/VSE *
	VS COBOL II ILB	VS COBOL II *
		LE/VSE *
	LE/VSE ILB	LE/VSE *
VS COBOL II IGY using RES compile-time option	VS COBOL II IGZ	VS COBOL II IGZ
		LE/VSE IGZ
	LE/VSE IGZ	LE/VSE IGZ
VS COBOL II IGY using NORES compile-time option	VS COBOL II IGZ	Not required **
	LE/VSE IGZ	LE/VSE IGZ
COBOL/VSE IGY	LE/VSE IGZ	LE/VSE IGZ

Notes:

* For programs compiled by DOS/VS COBOL, there are a few library routines that can be used at run time, but this is application program dependent.

** Prior to APAR PN09126, the VS COBOL II run-time library was required for DTF build routines.

FCOB Prefix for DOS/VS COBOL compiler modules.

ILB Prefix for DOS/VS COBOL run-time modules.

IGY Prefix for VS COBOL II and COBOL/VSE compiler modules.

IGZ Prefix for VS COBOL II run-time modules and COBOL language component run-time modules of LE/VSE.

Table 4 shows that for COBOL:

- There is a possibility of module name conflict when old and new products are installed concurrently,
- It is possible to use new LE/VSE run-time with older COBOL-compiled programs before implementing the COBOL/VSE compiler.

If you are a COBOL user, there are choices available so you must decide on a migration strategy. LE/VSE provides object compatibility for programs compiled with either DOS/VS COBOL or VS COBOL II. Therefore in most cases it is

possible to migrate the run-time component first and then gradually introduce the new LE-conforming compiler.

The migration scenarios for COBOL are:

- Source
  - DOS/VS COBOL 1.3.1 to COBOL/VSE
  - VS COBOL II (CMPR2) to COBOL/VSE
  - VS COBOL II (NOCMPR2) to COBOL/VSE
- Run-Time
  - DOS/VS COBOL 1.3.1 to LE/VSE
  - VS COBOL II to LE/VSE
  - LE/VSE Release 1 to LE/VSE Release 4

Recompile and/or relinkedit?

- When must I recompile?
  - For COBOL, it depends on whether source code has been amended to conform with COBOL/VSE requirements or if LE/VSE callable services are being introduced into the program.
- When must I relink?
  - Always if recompile has been done.
  - if no recompile for COBOL, it depends....., but highly recommended.
  - Few customers retain object modules, so usually recompile is done.

5.7.2 COBOL/VSE and LE/VSE Century Windowing

COBOL/VSE Intrinsic Functions

The COBOL/VSE intrinsic functions that provide full 4-digit year support are:

CURRENT-DATE
DATE-OF-INTEGER
DAY-OF-INTEGER
INTEGER-OF-DATE
INTEGER-OF-DAY
WHEN-COMPILED

An example of obtaining the 4-digit year using COBOL intrinsic functions:

DATE-OF-INTEGER gives YYYYMMDD

DAY-OF-INTEGER gives YYYYDDD

Note: Do not confuse the COBOL intrinsic function, **WHEN-COMPILED**, with the special register, **WHEN-COMPILED**. The intrinsic function is unique to COBOL/VSE and is designed to handle 4-digit years. The special register will return only a 2-digit year, even in COBOL/VSE.

5.7.3 An Example of the Century Window using COBOL/VSE and LE/VSE

LE/VSE callable services use Lilian dates for date representation. The base for this date is 14 October 1582. Thus, in a COBOL program, to convert a string representing a date into a Lilian format, use the CEEDAYS callable service.

CALL ″CEEDAYS″ USING *input-date* PICSTR *lilian-days* FC.

This returns an integer representation of the date as the number of days since 14 October 1582, in *lilian-days*.

Then convert this number of days back to a character string using the CEEDATE callable service.

CALL ″CEEDATE″ USING *lilian-days* PICSTR *output-char-date* FC.

This returns a character representation of the date with a **4-digit** value for the year, in *output-char-date*.

The example in Figure 6 on page 65 shows the use of LE/VSE callable services for the century window feature in a COBOL/VSE program.

In this example the century window is set:

- to the current year to demonstrate date arithmetic that is used to calculate how many years are left before a particular ′due-date′ occurs;
- to 100 years ago to demonstrate the ability to calculate the age of ′customers′, given their birth date.

Clearly, the century window can be set to any year, within a hundred-year range, that is applicable for the date arithmetic required for your application.

```
ID           DIVISION.
PROGRAM-ID. COBEXMP.
ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT RUNDATA ASSIGN TO INFILE
           FILE STATUS IS RUNDATA-FS.
DATA DIVISION.
FILE SECTION.
FD RUNDATA.
01 WORK-RECORD.
   03 CUST-NAME.
      05 FIRST-NAME PIC X(10).
      05 LAST-NAME  PIC X(15).
   03 BIRTH-DATE    PIC X(8).                          1
   03 ACCOUNT-NUM   PIC 9(8).
   03 CURRENT-AGE   PIC 999.
   03 DUE-DATE      PIC X(8).                          2
   03 BENNY-FACTOR  PIC 9(3)  COMP-3.
   03 ADJUSTMENT    PIC 9(3)  COMP-3.
   03 PAYOUT        PIC S9(7) COMP-3.
   03              PIC X(20).

WORKING-STORAGE SECTION.
01 RUNDATA-FS     PIC 99.
77 EOF-IND        PIC X.
  88 EOF           VALUE †Y†.
  88 NOT-EOF       VALUE †N†.

PROCEDURE DIVISION.

OPENIT. OPEN INPUT RUNDATA.
        IF RUNDATA-FS NOT EQUAL TO 0
          DISPLAY †** ERROR ** †
                  †CAN¢T OPEN RUNDATA FILE **†
        ELSE
          SET NOT-EOF TO TRUE
          PERFORM LOOP
          CLOSE RUNDATA
        END-IF

        STOP RUN.

LOOP. PERFORM UNTIL EOF
        READ RUNDATA
             AT END SET EOF TO TRUE
             NOT AT END
                 CALL †COBSUB† USING WORK-RECORD
        END-READ

      END-PERFORM.

END PROGRAM COBEXMP.
```

Figure 6 (Part 1 of 4). COBOL Example COBEXMP

```
 ID DIVISION.
  PROGRAM-ID. COBSUB.
 DATA DIVISION.
  WORKING-STORAGE SECTION.

  01 CURR-LILIAN PIC S9(9) COMP.
  01 CURR-SECONDS COMP-2.
  01 CURR-DATE.                                          3
    05 CURR-YEAR     PIC 9(4).
    05 CURR-MONTH    PIC 99.
    05 CURR-DAY      PIC 99.
    05 CURR-TIME     PIC X(9).
  77 YEARS-LEFT      PIC 9(4).

*************************************************
* Working Storage & parameters for CEEDATE      *
*************************************************
  01 4-DIGIT-DATE    PIC X(80).                         4
  01 4-DIGIT-REDEFINED REDEFINES 4-DIGIT-DATE.
    05 YYYY PIC 9(4).
    05 MM   PIC 9(2).
    05 DD   PIC 9(2).
    05 FILLER PIC X(72).

  01 PIC4STR.
    05 PIC4STR-LENGTH PIC 9(2) COMP VALUE 8.
    05 PIC4STR-STRING PIC X(8) VALUE †YYYYMMDD†.

*************************************************
*        Working storage & Parameters for       *
*              CEEDAYS and CEESCEN              *
*************************************************

  01 WORK-DATE.                                          5
    05 WORK-DATE-LEN PIC 9(2) VALUE 8.
    05 WORK-DATE-MM  PIC X(2) .
    05 WORK-DATE-F1  PIC X VALUE †/†.
    05 WORK-DATE-DD  PIC X(2).
    05 WORK-DATE-F2  PIC X VALUE †/†.
    05 WORK-DATE-YY  PIC X(2).

  01 PICSTR.
    05 PICSTR-LENGTH PIC 9(2) COMP VALUE 8.
    05 PICSTR-STRING PIC X(50) VALUE †MM/DD/YY†.
  77 LILIAN          PIC 9(9) COMP.
  77 START-CW        PIC 9(9) COMP.
  77 FC              PIC X(12).
```

Figure 6 (Part 2 of 4). COBOL Example COBEXMP

```
  LINKAGE SECTION.
************************************************
* Linkage Section description of the record     *
* that is read by COBEXMP, and shared by COBSUB.*
* Note that BIRTH-DATE and DUE-DATE             *
* are expanded to show formats.                 *
************************************************
  01 WORK-RECORD.
     03 CUST-NAME.
       05 FIRST-NAME   PIC X(10).
       05 LAST-NAME    PIC X(15).
     03 BIRTH-DATE.
       05 BMONTH       PIC 99.
       05             PIC X.
       05 BDAY         PIC 99.
       05             PIC X.
       05 BYEAR        PIC 99.
     03 ACCOUNT-NUM    PIC 9(8).
     03 CURRENT-AGE    PIC 999.
     03 DUE-DATE.
       05 DMONTH       PIC 99.
       05             PIC X.
       05 DDAY         PIC 99.
       05             PIC X.
       05 DYEAR        PIC 99.
     03 BENNY-FACTOR   PIC 9(3)  COMP-3.
     03 ADJUSTMENT     PIC 9(3)  COMP-3.
     03 PAYOUT         PIC S9(7) COMP-3.
     03               PIC X(20).

 PROCEDURE DIVISION USING WORK-RECORD.

************************************************
* get todays date with 4-digit year.            *
************************************************
     CALL †CEELOCT† USING CURR-LILIAN , CURR-SECONDS ,   6
          CURR-DATE , FC .

************************************************
* set century window to start with the current  *
* year (current system date).                   *
************************************************
     MOVE 0 TO START-CW.
     CALL †CEESCEN† USING START-CW FC.                   7
```

Figure 6 (Part 3 of 4). COBOL Example COBEXMP

```
************************************************
* convert 2-digit due date year from input file  *
* into integer value using LE/VSE service to get *
* COBOL Lilian date                              *
************************************************
     MOVE DMONTH TO WORK-DATE-MM.                          8
     MOVE DDAY TO WORK-DATE-DD.                            8
     MOVE DYEAR TO WORK-DATE-YY.                           8
     CALL †CEEDAYS† USING WORK-DATE                        9
                          PICSTR LILIAN FC.
************************************************
* convert COBOL Lilian date into YYYYMMDD format *
************************************************
     CALL †CEEDATE† USING LILIAN , PIC4STR ,              10
          4-DIGIT-DATE , FC.

************************************************
* find out how many years until due date         *
************************************************
     COMPUTE YEARS-LEFT = YYYY - CURR-YEAR.               11

************************************************
* set century window to start 100 years ago      *
************************************************
     MOVE 100 TO START-CW.
     CALL †CEESCEN† USING START-CW FC.                    12

************************************************
* convert 2-digit birthdate year from input file *
* into integer value to get COBOL Lilian date    *
************************************************
     MOVE BMONTH TO WORK-DATE-MM.                         13
     MOVE BDAY TO WORK-DATE-DD.                           13
     MOVE BYEAR TO WORK-DATE-YY.                          13
     CALL †CEEDAYS† USING WORK-DATE                       14
                          PICSTR LILIAN FC.

************************************************
* convert COBOL Lilian date into YYYYMMDD format *
************************************************
     CALL †CEEDATE† USING LILIAN , PIC4STR ,              15
          4-DIGIT-DATE , FC.

************************************************
* find the age † even after 1999!                *
************************************************
     COMPUTE CURRENT-AGE = CURR-YEAR - YYYY.              15

     GOBACK.

 END PROGRAM COBSUB.
```

Figure 6 (Part 4 of 4). COBOL Example COBEXMP

1 2 The code in the COBOL program, COBEXMP, shows that the work records contain both a BIRTH-DATE field for the customer and a DUE-DATE field for some provided service.

The main routine of the program simply opens the input file and then reads each record in turn, calling the subroutine, COBSUB, to process the records.

3 Working storage area to hold the returned values from the callable service CEELOCT 6 to obtain the current local date and time.

4 Working storage area to hold the date, including a 4-digit value for the year, returned by CEEDATE 10 15

5 Working storage area to provide a length-prefix field to the date character string, for the calls to CEEDAYS 9 14

6 CEELOCT is called to obtain the current local date. The values returned are

CURR-LILIAN A 32-bit integer representing the current local date in Lilian format.

CURR-SECONDS A 64-bit double-floating point number representing the current local date and time as the number of seconds since 00:00:00 on 14 October 1582.

CURR-DATE A 17-byte fixed-length character string in the form YYYYMMDDHHMISS999 representing local year, month, day, hour, minute, second, and millisecond.

(The examples quoted here were run with a current date of 17 April 1996, that is CURR-DATE = 19960417142127770.)

7 CEESCEN is called to set the century window to start with the current year.

8 The DUE-DATE values are moved to the WORK-DATE field to provide the required parameter format.

9 CEEDAYS is called to convert the DUE-DATE value from the input record to an integer value Lilian date. The input date field to CEEDAYS requires a halfword length-prefixed character string indicating its format.

10 CEEDATE is called to convert the DUE-DATE value into a character string representation including a 4-digit year value.

For example, a record containing the DUE-DATE value of

- DMONTH = 01
- DDAY = 30
- DYEAR = 03

produces a date value of **20030130**

11 After this statement, the YEARS-LEFT field contains the value **7**.

12 CEESCEN is called again to set the century window to start 100 years before the current system date.

13 The BIRTH-DATE values are moved to the WORK-DATE field.

14 CEEDAYS is called again to get the Lilian date integer value.

15 CEEDATE is called again to convert the date to a 4-digit year date.

For example, a record containing the BIRTH-DATE value of

- BMONTH = 04
- BDAY = 05
- BYEAR = 09

produces a date value of **19090405**.

16 After this statement, the CURRENT-AGE field contains the value **87**.

This example is a simple, but short-term, approach to a solution to the Year2000 problem using the century window feature of LE/VSE with COBOL/VSE.

5.7.4 CEECBLDY - LE/VSE Callable Service to Convert a Date to COBOL Integer Format

The CEECBLDY callable service is provided with LE/VSE Release 4. It is primarily designed for COBOL applications, to convert a date into a format compatible with ANSI 85 Intrinsic Functions. However, you can also use it in a PL/I or C program to convert a date to a format suitable for use in a COBOL program called subsequently from the PL/I or C program.

COBOL/VSE intrinsic functions use Gregorian dates for date representation. The base used is 31 December 1600.

Use CEECBLDY to access the century window and to perform date calculations with ANSI intrinsic functions. For example, use CEECBLDY to convert a string representing a date into a COBOL integer format which is compatible with ANSI intrinsic functions. This integer is the number of days since 31 December 1600.

Call CEECBLDY only from programs that will use the returned value as the input to a COBOL intrinsic function. Do not use this returned value with other LE/VSE callable services.

5.7.5 An Example of CEECBLDY Callable Service

Figure 7 on page 71 is a sample COBOL program showing the use of CEECBLDY to convert a date to COBOL integer format.

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. CBLDAYS.
 DATA DIVISION.
 WORKING-STORAGE SECTION.
 01  CHRDATE.
     05  CHRDATE-LENGTH      PIC  9(2) BINARY.
     05  CHRDATE-STRING      PIC X(50).
 01  PICSTR.
     05  PICSTR-LENGTH       PIC S9(4) BINARY.
     05  PICSTR-STRING       PIC X(50).
 01  INTEGER                 PIC S9(9) BINARY.
 01  NEWDATE                 PIC 9(8).
 01  FC                      PIC X(12).

 PROCEDURE DIVISION.
************************************************
** Specify input date and length              **
************************************************
     MOVE †1 January 00† to CHRDATE-STRING.            1
     MOVE 25 TO CHRDATE-LENGTH.

************************************************
** Specify a picture string that describes    **
** input date, and the picture string†s length.**
************************************************
     MOVE †ZD Mmmmmmmmmmmmmz YY†
                          TO PICSTR-STRING.            2
     MOVE 23 TO PICSTR-LENGTH.

************************************************
** Call CEECBLDY to convert input date to a   **
** COBOL Integer date                         **
************************************************
     CALL †CEECBLDY† USING CHRDATE, PICSTR,            3
                           INTEGER, FC.

************************************************
** If CEECBLDY runs successfully, then compute **
** the date of the 90th day after the         **
** input date using Intrinsic Functions       **
************************************************
     IF (FC = LOW-VALUE) THEN
         COMPUTE INTEGER = INTEGER + 90
         COMPUTE NEWDATE = FUNCTION                    4
                       DATE-OF-INTEGER (INTEGER)
         DISPLAY NEWDATE † is Integer day: † INTEGER   5
     ELSE
         CONTINUE
     END-IF.

     GOBACK.
```

Figure 7. An Example of the Use of CEECBLDY

1 2 The MOVE statement 1 sets up the character string representing the date in the format specified in the PICSTR field 2 .

3 CEECBLDY is called to obtain the COBOL integer format date in the field INTEGER. This date represents the number of days since 31 December 1600.

4 Convert the COBOL integer date returned by CEECBLDY to the standard date form YYYYMMDD (Year, Month, Day) in the field NEWDATE.

5 The result of the DISPLAY statement is

20000331 is Integer day: 000145822

5.8 PL/I and the Year2000

DOS PL/I programs, as DOS/VS COBOL, will only support 2-digit dates, and will not be upgraded for Year2000 support.

A valid solution for PL/I users is to convert to PL/I VSE and use the LE/VSE callable services for date/time routines, or the built-in function DATETIME.

LE/VSE is not supported under DOS PL/I.

Please refer to the manual *PL/I for VSE/ESA Migration Guide* for information concerning migrating to the Year2000 supporting version of PL/I.

If you cannot migrate to PL/I VSE because of constraints in time or adaptability, then you could write you own version of the fixed window technique to suit your needs to convert 2-digit years to 4-digit years, or implement an externally coded High Level Assembler routine to use a sliding window with the YEAR224 macro.

5.9 DOS PL/I Considerations

In the following example of a DOS PL/I program, there is a number of date fields. These date fields are stored in the format of MMDDYY in the VSE/VSAM file and do not carry century information. The dates may or may not be required to be converted to full century/year format YYYY depending upon the logic and requirement of the application.

```
PLIPROG: PROC OPTIONS(MAIN);
DCL   INFILE   FILE RECORD SEQUENTIAL INPUT  ENV(VSAM);
DCL   PRTFILE  FILE RECORD SEQUENTIAL OUTPUT ENV(F RECSIZE(133)
               MEDIUM (SYS011,1403) CTLASA BUFFERS(2));
DCL 1 INREC,
      2 IN-KEY          CHAR  (05)     INIT (¢ ¢),
      .
      .
      .
      2 IN_BILDAT       CHAR  (06)     INIT (¢ ¢),        1
                    A
      2 IN-PAYDAT       CHAR  (06)     INIT (¢ ¢);        2

                    B
DCL 1 PRPCHG,
      .
DCL 1 PRLAD1,
      .
DCL 1 PRLAD2,
      .
DCL 1 PRTHD1,
      .
      2 PHRDATE          PIC  ¢99/99/99¢INIT (0),         3
      .
DCL  1 PRTHD2,
      .
DCL  1 PRTHD3,
      .
      2 PHFL37             CHAR (11) INIT (¢LAST BILLED¢),
      2 PHFL38             CHAR (22) INIT (¢ ¢),
      2 PHFL39             CHAR (09) INIT (¢LAST PAID¢),
      2 PHFL3C             CHAR (06) INIT (¢ ¢);
DCL  1 PRTHD4,
      .
DCL  1 PRTREC,
      .
      2 PRBIL              PIC ¢99/99/99¢ INIT (0);
      2 PRFL7              CHAR (05) INIT (¢ ¢);
      2 PRDPD              PIC ¢99/99/99¢ INIT (0);
   /*********************************************************/
   /****                                                 ****/
   /****                 WORKING STORAGE                 ****/
   /****                                                 ****/
   /*********************************************************/
```

Figure 8 (Part 1 of 3). A Sample DOS PL/I Program

```
DCL  MORE_RECORDS          BIT  (1)   INIT (¢1¢B);
.
.
DCL  PGNO                  FIXED(7,0) INIT (0);
                    C
DCL  DATE                  BUILTIN;                    4
DCL  TIME                  BUILTIN;
DCL  LOW                   BUILTIN;
DCL  SUBSTR                BUILTIN;
/*****************************************************************/
/****                                                         ****/
/****                     PROCEDURE                           ****/
/****                                                         ****/
/*****************************************************************/
                                                                 /*
        MAIN-LINE ROUTINE
        *****************                                        */
        ON ENDFILE(INFILE) MORE_RECORDS  =  OFF;
        OPEN FILE (INFILE),
             FILE (PRTFILE);
        READ FILE (INFILE) INTO (INREC);
        DO WHILE (MORE_RECORDS);
                    D
           IF LCT  >  50   THEN
              CALL PRINT_HEADING;
           CALL PRINT_DETAIL_LINE;
           READ FILE (INFILE) INTO (INREC);
        END;
        CLOSE FILE (INFILE),
              FILE (PRTFILE);
                                                                 /*
        END OF MAIN-LINE ROUTINE
        ************************
                                                                 */
PRINT_HEADING:      PROCEDURE;
/************                                                   */
       LCT        =  0;
       PGNO       =  PGNO  +  1;
       PHPGNO     =  PGNO;
       PHRDATE    =  DATE;                    5
       .
```

Figure 8 (Part 2 of 3). A Sample DOS PL/I Program

```
      WRITE FILE (PRTFILE) FROM (PRLAD1);
      END PRINT_HEADING;
/*                                                                */
PRINT_DETAIL_LINE:     PROCEDURE;
/****************                                                */
      PRKEY      =  IN_KEY;
      .
      .
      PRBIL      =  IN_BILDAT;                6
      PRDPD      =  IN_PAYDAT;                7
      WRITE FILE (PRTFILE) FROM (PRTREC);
      LCT                     =  LCT  +  1;
      PRKEY      =  SP5;
      PRNAM      =  SP30;
      PRCPN      =  SP25;
      PRAD1      =  SP25;
      PRAD1      =  IEA2;
      WRITE FILE (PRTFILE) FROM (PRTREC);
      LCT        =  LCT  +  1;
      PRAD1      =  SP25;
      PRAD1      =  IEA3;
      WRITE FILE (PRTFILE) FROM (PRTREC);
      LCT        =  LCT  +  1;
      PRAD1      =  SP25;
      PRAD1      =  IEPC;
      WRITE FILE (PRTFILE) FROM (PRTREC);
      LCT        =  LCT  +  1;
      WRITE FILE (PRTFILE) FROM (PRLAD1);
      LCT        =  LCT  +  1;
      END PRINT_DETAIL_LINE;

                  E

END PLIPROG;
```

Figure 8 (Part 3 of 3). A Sample DOS PL/I Program

1 2 6 7 The code in the DOS PL/I program has two year fields on input in YY format. These fields may be complemented with century information depending on the requirements of your organization. A sample of a routine to convert the year from 2-digit format to 4-digit format using a fixed window may be found in Figure 9 on page 76.

3 4 5 The date shown on the report would be in the format MMDDYY obtained from the built-in function. In DOS PL/I this can only be in YY format.

The field PHRDATE contains the date obtained from the built-in function DATE. Depending on organizational standards, this heading may be considered as a *cosmetic* change.

```
   PLIPROG: PROC OPTIONS(MAIN);
                    .
                    .
                    .
                    .
DCL 1 INREC,
                    .
                    .
                    .
                    .
                    .
                    .
                    .
                    .
                    .
      2 IN_BILDAT       CHAR  (06)     INIT (¢ ¢),
        3 IN_BIL_MM     CHAR  (2)      INIT (¢ ¢),
        3 IN_BIL_DD     CHAR  (2)      INIT (¢ ¢),
        3 IN_BIL_YY     CHAR  (2)      INIT (¢ ¢),
                    A

      2 IN-PAYDAT       CHAR  (06)     INIT (¢ ¢);
        3 IN_PAY_MM     CHAR  (2)      INIT (¢ ¢),
        3 IN_PAY_DD     CHAR  (2)      INIT (¢ ¢),
        3 IN_PAY_YY     CHAR  (2)      INIT (¢ ¢),

                    B
                       .
   DCL  LCT                    FIXED(7,0) INIT (88);
   DCL  PGNO                   FIXED(7,0) INIT (0);
   DCL  WINDOW                 FIXED(2,0) INIT (60);
                    C

   DCL  CENTURY                FIXED(4,0) INIT (0);
   DCL  BIL_CENTURY            FIXED(4,0) INIT (0);
   DCL  PAY_CENTURY            FIXED(4,0) INIT (0);
   DCL  DATE                   BUILTIN;
   DCL  TIME                   BUILTIN;
   DCL  LOW                    BUILTIN;
   DCL  SUBSTR                 BUILTIN;
        ON ENDFILE(INFILE) MORE_RECORDS  =  OFF;
        OPEN FILE (INFILE),
             FILE (PRTFILE);
        READ FILE (INFILE) INTO (INREC);
        DO WHILE (MORE_RECORDS);
                    D
```

Figure 9 (Part 1 of 2). DOS PL/I Fixed Window Example

```
            CENTURY = IN_BIL_YY;
            CALL CONVERT_YY;
            BIL_CENTURY = CENTURY ;
            CENTURY = IN_PAY_YY;
            CALL CONVERT_YY;
            PAY_CENTURY = CENTURY ;
            IF LCT > 50 THEN
            CALL PRINT_HEADING;
            CALL PRINT_DETAIL_LINE;
            READ FILE (INFILE) INTO (INREC);
         END;
         CLOSE FILE (INFILE),
               FILE (PRTFILE);
                     .
                     .
                     .
                     .
                     .
                      E
CONVERT_YY:     PROCEDURE;
          IF CENTURY  >  60   THEN
          CENTURY = CENTURY + 1900;
          ELSE CENTURY=CENTURY +2000;
END PLIPROG;
```

Figure 9 (Part 2 of 2). DOS PL/I Fixed Window Example

A B The fields may have been further defined into their MM, DD and YY structures for convenience. The fields IN_BIL_YY and IN_PAY_YY would be input to the conversion routine shown in E .

C The fields CENTURY, BIL_CENTURY and PAY_CENTURY have been added to working storage as work fields and for use as output in the report.

If these fields were used as output on the report, the structure PHRDATE as described in Figure 8 on page 73 would need to be changed.

D The fields CENTURY, BIL_CENTURY and PAY_CENTURY have been added to working storage as work fields and for use as output in the report.

If these fields were used as output on the report, the structure PHRDATE as described in Figure 8 on page 73 would need to be changed.

E This procedure determines whether the input date in the format of YY is greater than 60 (1960) and complements the field with the century information.

The results of running tests against this procedure produced the following results:

TEST YEAR	FIXED WINDOW	FULL YEAR NOTATION
45	60	2045
98	60	1998
61	60	1961
60	60	2060
00	60	2000
05	60	2005
59	60	2059

Figure 10. Fixed Window Test Results for DOS PL/I Procedure

5.10 LE/VSE and PL/I

As you are aware, the version of PL/I that is supported in the LE/VSE environment is PL/I VSE. DOS PL/I programs can co-exist with PL/I VSE programs and be executed in their environment as it is now, by accessing the currently supplied DOS PL/I run-time libraries, and using the DATE built-in function to obtain the date from the partition COMREG.

The date returned to the DOS PL/I program will be, of course, in either DDMMYY or MMDDYY format.

Programs migrated to PL/I VSE may use LE/VSE run-time libraries, and the built-in function DATETIME that will return the date in the format of MMDDYYYY, or may use the LE/VSE CEELOCT callable routine, which performs the same function.

5.10.1 Migrating from DOS PL/I to PL/I VSE and LE/VSE

The migration to the LE/VSE environment consists of a source migration to PL/I VSE and run-time migration to LE/VSE. There is no object compatibility between DOS PL/I and LE/VSE so all application programs must be recompiled with the LE-conforming compiler and link-edited using LE/VSE library.

Table 5. Valid Scenarios for PL/I Compiler, Link-Edit, and Run-Time

If you compile with this compiler...	**And you link-edit with this run-time library...**	**Then you can run with this run-time library...**
DOS PL/I PLI	DOS PL/I IBM	DOS PL/I IBM
PL/I VSE IEL	LE/VSE IBM	LE/VSE IBM

Note:

Although the module prefixes are shared, all module names are unique

PLI Prefix for DOS PL/I compiler modules.

IEL Prefix for PL/I VSE compiler modules.

IBM Prefix for DOS PL/I and LE/VSE PL/I run-time modules.

Table 5 on page 78 shows that for PL/I:

- There is no module name conflict when old and new products are installed concurrently,
- It is not possible to use new LE/VSE run-time with older PL/I-compiled programs before implementing the PL/I VSE compiler.

The migration scenarios for PL/I are:

- Source
  - DOS PL/I to PL/I VSE
- Run-Time
  - DOS PL/I to LE/VSE
  - LE/VSE Release 1 to LE/VSE Release 4

5.10.2 An Example of the Century Window using PL/I VSE and LE/VSE

LE/VSE callable services use Lilian dates for date representation. The base used for this date is 14 October 1582. Thus, in a PL/I program, to convert a string representing a date into a Lilian format, use the CEEDAYS callable service.

CALL CEEDAYS (*input-date*, PICSTR, *lilian-days*, FC);

This returns an integer representation of the date as the number of days since 14 October 1582, in *lilian-days*.

Then convert this number of days back to a character string using the CEEDATE callable service.

CALL CEEDATE (*lilian-days*, PICSTR, *output-char-date*, FC);

This returns a character representation of the date with a **4-digit** value for the year, in *output-char-date*.

The example in Figure 11 on page 80 shows the use of LE/VSE callable services for the century window feature in a PL/I VSE program.

In this example the century window is set:

- to the current year to demonstrate date arithmetic that is used to calculate how many years are left before a particular ′due-date′ occurs;
- to 100 years ago to demonstrate the ability to calculate the age of ′customers′, given their birth date.

Clearly, the century window can be set to any year, within a hundred-year range, that is applicable for the date arithmetic required for your application.

```
*PROCESS MACRO;
 PLIEXMP: PROC OPTIONS(MAIN);
        %INCLUDE  CEEIBMAW;
        %INCLUDE  CEEIBMCT;
        DCL EOF_IND  BIT(1) INIT(¢0¢B); /* End of File indicator */
        DCL RUNDATA FILE RECORD INPUT SEQUENTIAL  /* File        */
            ENV(RECSIZE(80));                    /* declaration */
        DCL 1 IN,                      /* Input record          */
              2 IN_CUST_NAME,                    /*  structure  */
                3 IN_FIRST_NAME CHAR(10),
                3 IN_LAST_NAME  CHAR(15),
              2 IN_BIRTH_DATE,                                 1
                3 IN_BMONTH     CHAR(2),
                3 IN_BF1        CHAR(1),
                3 IN_BDAY       CHAR(2),
                3 IN_BF2        CHAR(1),
                3 IN_BYEAR      CHAR(2),
              2 IN_ACCOUNT_NUM  CHAR(8),
              2 IN_CURRENT_AGE  CHAR(3),
              2 IN_DUE_DATE,                                   2
                3 IN_DMONTH     CHAR(2),
                3 IN_DF1        CHAR(1),
                3 IN_DDAY       CHAR(2),
                3 IN_DF2        CHAR(1),
                3 IN_DYEAR      CHAR(2),
              2 IN_BENNY_FACTOR CHAR(3),
              2 IN_ADJUSTMENT   CHAR(3),
              2 IN_PAYOUT       CHAR(7),
              2 IN_WRF          CHAR(15);
       ON ENDFILE(RUNDATA) EOF_IND = ¢1¢B;

       OPEN FILE(RUNDATA);

       READ FILE(RUNDATA) INTO(IN);

       DO WHILE (EOF_IND = 0);
         CALL PLISUB(IN);
         READ FILE(RUNDATA) INTO(IN);
         END;
```

Figure 11 (Part 1 of 4). PL/I Example PLIEXMP

```
PLISUB: PROC(WORK_RECORD);
   DCL LILIAN          INT4;                              3
   DCL SECONDS         FLOAT8;                            3
   DCL 1 GREG,                                            3
         2 CURR_YEAR   CHAR(4),
         2 CURR_MONTH  CHAR(2),
         2 CURR_DAY    CHAR(2),
         2 CURR_TIME   CHAR(9),
       GREGORN         CHAR(17) DEF GREG;
   DCL CHRDATE         VSTRING;
   DCL PICSTR          VSTRING;
   DCL PIC4STR         VSTRING;
   DCL LIL2            INT4;
   DCL STARTCW         INT4;
   DCL YYYY            CHAR(4);
   DCL FOUR_DIGIT_DATE CHAR(80);                          4
   DCL YEARS_LEFT      FIXED DECIMAL(15,0);
   DCL AGE             FIXED DECIMAL(15,0);
   DCL 01 FC           FEEDBACK;
  DCL 1 WORK_RECORD,                                      5
        2 CUST_NAME,
          3 FIRST_NAME CHAR(10),
          3 LAST_NAME  CHAR(15),
        2 BIRTH_DATE,
          3 BMONTH     CHAR(2),
          3 BF1        CHAR(1),
          3 BDAY       CHAR(2),
          3 BF2        CHAR(1),
          3 BYEAR      CHAR(2),
        2 ACCOUNT_NUM  CHAR(8),
        2 CURRENT_AGE  CHAR(3),
        2 DUE_DATE,
          3 DMONTH     CHAR(2),
          3 DF1        CHAR(1),
          3 DDAY       CHAR(2),
          3 DF2        CHAR(1),
          3 DYEAR      CHAR(2),
        2 BENNY_FACTOR CHAR(3),
        2 ADJUSTMENT   CHAR(3),
        2 PAYOUT       CHAR(7),
        2 WRF          CHAR(15);
```

Figure 11 (Part 2 of 4). PL/I Example PLIEXMP

```
/************************************************/
/* Call CEELOCT to return local time in 3 formats */
/************************************************/

 CALL CEELOCT ( LILIAN, SECONDS, GREGORN, FC );          6

/************************************************/
/* Set century window to start with the current   */
/* year current system date)                      */
/************************************************/

 STARTCW = 0;
 CALL CEESCEN ( STARTCW, FC );                           7
/************************************************/
/* Convert 2-digit due date year from input file  */
/* into integer value using LE/VSE service to get */
/* PL/I Lilian date                               */
/************************************************/

 CHRDATE = DMONTH || ¢/¢ || DDAY || ¢/¢ || DYEAR;       8
 PICSTR = ¢ZM/ZD/YY¢;
 CALL CEEDAYS ( CHRDATE, PICSTR, LILIAN, FC);            9

/************************************************/
/* Convert PL/I Lilian date into YYYYMMDD format  */
/************************************************/

 PIC4STR = ¢YYYYMMDD¢;
 CALL CEEDATE ( LILIAN, PIC4STR, FOUR_DIGIT_DATE, FC);   10

/************************************************/
/* Calculate number of years until due date       */
/************************************************/

 YYYY = SUBSTR(FOUR_DIGIT_DATE, 1, 4);
 YEARS_LEFT = YYYY - CURR_YEAR;                          11
```

Figure 11 (Part 3 of 4). PL/I Example PLIEXMP

```
/************************************************/
/* Set century window to start 100 years before   */
/* the current system date                        */
/************************************************/
 STARTCW = 100;
 CALL CEESCEN (STARTCW, FC);                              12

/************************************************/
/* Convert 2-digit birthdate year from input file */
/* into integer value using LE/VSE service to get */
/* PL/I Lilian date                               */
/************************************************/
 CHRDATE = BMONTH || ¢/¢ || BDAY || ¢/¢ || BYEAR;         13
 PICSTR = ¢ZM/ZD/YY¢;
 CALL CEEDAYS ( CHRDATE, PICSTR, LILIAN, FC);             14

/************************************************/
/* Convert PL/I Lilian date into YYYYMMDD format  */
/************************************************/
 CALL CEEDATE ( LILIAN, PIC4STR, FOUR_DIGIT_DATE, FC);    15

/************************************************/
/* Find the age - even after 1999!                */
/************************************************/
 YYYY = SUBSTR(FOUR_DIGIT_DATE, 1, 4);
 AGE = CURR_YEAR - YYYY;                                  16

 END PLISUB;
 END PLIEXP;
```

Figure 11 (Part 4 of 4). PL/I Example PLIEXMP

1 2 The code in the PL/I program, PLIEXMP, shows that the work records contain both an IN_BIRTH_DATE field for the customer and a IN_DUE_DATE field for some provided service.

The main routine of the program simply opens the input file and then reads each record in turn, calling PLISUB to process the records.

3 Work areas to hold the returned values from the callable service CEELOCT 6 to obtain the current local date and time.

4 Work area to hold the date, including a 4-digit value for the year, returned by CEEDATE 10 15

5 Definition of the structure of the input record passed to the sub-routine, PLISUB.

6 CEELOCT is called to obtain the current local date. The values returned are:

LILIAN A 32-bit integer representing the current date in Lilian format.

SECONDS A 64-bit double-floating point number representing the current local date and time as the number of seconds since 00:00:00 on 14 October 1582.

GREGORN A 17-byte fixed-length character string in the form YYYYMMDDHHMISS999 (the Gregorian format) representing local year, month, day, hour, minute,second, and millisecond.

(The examples quoted here were run with a current date of 17 April 1996, that is GREGORN = 19960417142117770.)

7 CEESCEN is called to set the century window to start with the current year.

8 The DUE_DATE values are assigned to the CHRDATE field to provide the required parameter format as defined in the PICSTR field.

9 CEEDAYS is called to convert the DUE_DATE value from the input record to an integer value Lilian date. The input date field to CEEDAYS requires a halfword length-prefixed character string indicating it's format.

10 CEEDATE is to convert the DUE_DATE value into a character string representation including a 4-digit year value.

For example, a record containing the DUE_DATE value of

- DMONTH = 01
- DDAY = 30
- DYEAR = 03

produces a date value of **20030130**.

11 After this statement the YEARS_LEFT field contains the value **7**.

12 CEESCEN is called again to set the century window to start 100 years before the current system date.

13 The BIRTH_DATE values are assigned to the CHRDATE field.

14 CEEDAYS is called again to get the Lilian date integer value.

15 CEEDATE is called again to convert the date to a 4-digit year date.

For example, a record containing the BIRTH_DATE value of

- BMONTH = 04
- BDAY = 05
- BYEAR = 09

produces a date value of **19090405**.

16 After this statement the AGE field contains the value **87**.

This example is a simple, but short-term, approach to the Year2000 problem using the century window feature of LE/VSE with PL/I VSE.

5.10.3 The DATE and DATETIME Built-In Functions

PL/I provides two built-in functions to return the current system date and time:

DATE returns a character string of length 6, in the format **yymmdd**.

DATETIME returns a character string of length 17, in the format **yyyymmddhhmmssttt**.

DATE returns the date with only 2-digit years. Use the **DATETIME** built-in function to obtain a date with a 4-digit year.

5.11 C and the Year2000

The C/370 compiler, as DOS/VS COBOL and DOS PL/I, will only support 2-digit dates, and will not be upgraded for Year2000 support.

The only solution for C/370 users is to convert to C/VSE and use the LE/VSE callable services for date/time routines. LE/VSE is not supported under C/370.

If you cannot migrate to C/VSE, then the only alternative is to write your own fixed window routine to convert 2-digit dates to 4-digit dates, or implement an externally coded High Level Assembler routine to use a sliding window with the YEAR224 macro.

5.12 LE/VSE and C

As you are aware, the version of C that is supported in the LE/VSE environment is C/VSE. C/370 programs can co-exist with C/VSE programs and be executed in their environment as it is now, by accessing the currently supplied C/370 run-time libraries, and using the %x built-in function to obtain the local date.

The date returned to the C/370 program will be, of course, in the MMDDYY format.

Programs migrated to C/VSE will use LE/VSE run-time libraries, and use the LE/VSE CEELOCT callable routine to obtain the date/time.

5.12.1 Migrating from C/370 to C/VSE and LE/VSE

The migration to the LE/VSE environment consists of a source migration to C/VSE and run-time migration to LE/VSE. There is no object compatibility between C/370 and LE/VSE, so all applications must be recompiled with the LE-conforming compiler and link-edited using LE/VSE library.

Note: C run-time support was introduced in LE/VSE Release 4.

Table 6. Valid Scenarios for C Compiler, Link-Edit, and Run-Time		
If you compile with this compiler...	**And you link-edit with this run-time library...**	**Then you can run with this run-time library...**
C/370	C/370	C/370
C/VSE	LE/VSE[1]	LE/VSE[1]
Note: 1. LE/VSE Release 4 only The module prefixes are numerous in these products, and names are not unique between the products.		

Table 6 shows that for C:

- It is not possible to use new LE/VSE run-time with older C-compiled programs before implementing the C/VSE compiler.

Note: There are module name conflicts when old and new products are installed concurrently.

The migration scenarios for C are:

- Source
  - C/370 to C/VSE
- Run-Time
  - C/370 to LE/VSE Release 4

5.12.2 An Example of the Century Window using C/VSE and LE/VSE

LE/VSE callable services use Lilian dates for date representation. The base used for this date is 14 October 1582. Thus, in a C/VSE program, to convert a string representing a date into a Lilian format, use the CEEDAYS callable service.

CEEDAYS(*&date, &date_pic, &lil_date, &fc*);

This returns an integer representation of the date as the number of days since 14 October 1582, in *lil_date*.

Then convert this number of days back to a character string using the CEEDATE callable service.

CEEDATE(*&lil_date, &date_pic, date_out, &fc*);

This returns a character representation of the date with a **4-digit** value for the year, in *date_out*.

The code example in Figure 12 on page 87 shows code that uses the LE/VSE callable services for the century window feature in a C/VSE program.

In this example the century window is set:

- to the current year to demonstrate date arithmetic that is used to calculate how many years are left before a particular 'due-date' occurs;
- to 100 years ago to demonstrate the ability to calculate the age of 'customers', given their birth date.

Clearly, the century window can be set to any year, within a hundred-year range, that is applicable for the date arithmetic required for your application.

```
  /* ********************************************** */
  /* EDCEXMP                                        */
  /* C/VSE Routine to demonstrate the LE/VSE Century    */
  /*                 Window Feature                 */
  /* ********************************************** */

   #include <stdlib.h>
   #include <string.h>
   #include <leawi.h>
   #include <ceeedcct.h>

int main(void) {

  _FEEDBACK fc;
  _INT4    century_start;

  _INT4    lil_today;                                    1
  _FLOAT8  lil_secs;                                     1
  _CHAR17  gregorian_date;                               1

  _VSTRING due_date,due_date_pic;
  _INT4    lil_due;
  _VSTRING due4_date,due4_date_pic;
  _CHAR80  date_out1;
  _CHAR80  digit4_due_date;                              2

  _INT4    curr4_year_int;
  _INT4    due4_year_int;
  char     curr4_year[5];
  char     due4_year[5];
  _INT4    years_left;

  _VSTRING birth_date,birth_date_pic;
  _INT4    lil_birth;
  _VSTRING birth4_date,birth4_date_pic;
  _CHAR80  date_out2;
  _CHAR80  digit4_birth_date;                            2

  _INT4    birth4_year_int;
  char     birth4_year[5];
  _INT4    curr_age;
```

Figure 12 (Part 1 of 4). C/VSE Example EDCEXMP

```
/* ************************************************ */
/* Call CEELOCT to get todays date with a 4-digit year */
/* ************************************************ */

CEELOCT(&lil_today,&lil_secs,gregorian_date,&fc);              3
if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
   printf(†CEELOCT failed with message number %d\n†,
          fc.tok_msgno);
   exit(2999);
}
/* ************************************************ */
/* Call CEESCEN to set the century window to start     */
/* with the current year (current system date)         */
/* ************************************************ */

century_start = 0;

CEESCEN(&century_start,&fc);                                   4
if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
   printf(†CEESCEN failed with message number %d\n†,
          fc.tok_msgno);
   exit(2999);
}
/* ************************************************ */
/* Call CEEDAYS to convert a 2-digit year Due Date     */
/* to Lilian format                                    */
/* ************************************************ */

 strcpy(due_date.string,†01/30/03†);                           5
 due_date.length = strlen(due_date.string);
 strcpy(due_date_pic.string,†MM/DD/YY†);
 due_date_pic.length = strlen(due_date_pic.string);

 CEEDAYS(&due_date,&due_date_pic,&lil_due,&fc);                6
 if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
    printf(†CEEDAYS failed with message number %d\n†,
           fc.tok_msgno);
    exit(2999);
 }
```

Figure 12 (Part 2 of 4). C/VSE Example EDCEXMP

```
/* ************************************************ */
/* Call CEEDATE to convert the Lilian date to       */
/* MM/DD/YYYY format                                */
/* ************************************************ */

 strcpy(due4_date_pic.string,†MM/DD/YYYY†);
 due4_date_pic.length = strlen(due4_date_pic.string);

 CEEDATE(&lil_due,&due4_date_pic,digit4_due_date,&fc);          7

 if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
    printf(†CEEDATE failed with message number %d\n†,
           fc.tok_msgno);
    exit(2999);
 }
   printf(†The due date is %.80s\n†,digit4_due_date);
/* ***************************************** */
/* Calculate years left to due date          */
/* ***************************************** */

memset (curr4_year,¢\0¢,5);                                     8
memset (due4_year,¢\0¢,5);
strncpy (curr4_year,gregorian_date,4);
strncpy (due4_year,digit4_due_date+6,4);

curr4_year_int = atoi(curr4_year);
due4_year_int = atoi(due4_year);
years_left = due4_year_int - curr4_year_int;                    9
printf(†Number of years to due date is is %d\n†,years_left);
/* ************************************************ */
/* Call CEESCEN to set the century window to start  */
/* 100 years ago                                    */
/* ************************************************ */

century_start = 100;

CEESCEN(&century_start,&fc);                                    10
if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
   printf(†CEESCEN failed with message number %d\n†,
          fc.tok_msgno);
   exit(2999);
}
```

Figure 12 (Part 3 of 4). C/VSE Example EDCEXMP

```
  /* ************************************************ */
  /* Call CEEDAYS to convert a 2-digit year Birth Date   */
  /* to Lilian format                                    */
  /* ************************************************ */

   strcpy(birth_date.string,†04/05/09†);                          11
   birth_date.length = strlen(birth_date.string);
   strcpy(birth_date_pic.string,†MM/DD/YY†);
   birth_date_pic.length = strlen(birth_date_pic.string);

   CEEDAYS(&birth_date,&birth_date_pic,&lil_birth,&fc);           12
   if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
      printf(†CEEDAYS failed with message number %d\n†,
             fc.tok_msgno);
      exit(2999);
   }
  /* ************************************************ */
  /* Call CEEDATE to convert the Lilian date to          */
  /* MM/DD/YYYY format                                   */
  /* ************************************************ */

   strcpy(birth4_date_pic.string,†MM/DD/YYYY†);
   birth4_date_pic.length = strlen(birth4_date_pic.string);

   CEEDATE(&lil_birth,&birth4_date_pic,digit4_birth_date,&fc);    13

   if ( _FBCHECK ( fc , CEE000 ) != 0 ) {
      printf(†CEEDATE failed with message number %d\n†,
              fc.tok_msgno);
      exit(2999);
   }
     printf(†The birth date is %.80s\n†,digit4_birth_date);
  /* ***************************************** */
  /* Calculate age in current yyear               */
  /* ***************************************** */

  memset (curr4_year,¢\0¢,5);                                     14
  memset (birth4_year,¢\0¢,5);
  strncpy (curr4_year,gregorian_date,4);
  strncpy (birth4_year,digit4_birth_date+6,4);

  curr4_year_int = atoi(curr4_year);
  birth4_year_int = atoi(birth4_year);
  curr_age = curr4_year_int - birth4_year_int;                    15
  printf(†Age is %d\n†,curr_age);

}
```

Figure 12 (Part 4 of 4). C/VSE Example EDCEXMP

The code in the C program, EDCEXMP, shows that the work records contain both a birth date field for the customer and a due date field for some provided service.

1 Areas defined to hold the returned values from the callable service CEELOCT 3 to obtain the current local date and time.

2 Area defined to hold the date, including a 4-digit value for the year, returned by CEEDATE 7 13

3 CEELOCT is called to obtain the current local date. The values returned are:

lil_today A 32-bit integer representing the current date in Lilian format.

lil_secs A 64-bit double-floating point number representing the current local date and time as the number of seconds since 00:00:00 on 14 October 1582.

gregorian_date A 17-byte fixed-length character string in the form YYYYMMDDHHMISS999 (the Gregorian format) representing local year, month, day, hour, minute,second, and millisecond.

(The example quoted here was run with a current date of 20 August 1996, that is **gregorian_date** = 19960820142117770.)

4 CEESCEN is called to set the century window to start with the current year.

5 The due date values are set up in the **due_date** fields.

6 CEEDAYS is called to convert the **due_date** value from the input record to an integer value Lilian date. The input date field to CEEDAYS requires a halfword length-prefixed character string, **due_date_pic**. to indicate it's format.

7 CEEDATE is called to convert the due date value into a character string representation including a 4-digit year value.

For example, a record containing the **due_date** value "01/30/03", that is, 30th January 2003, produces a date value of **01/30/2003**.

8 After these statements the **years_left** field 9 contains the value **7**.

10 CEESCEN is called again to set the century window to start 100 years before the current system date.

11 The birth date values are set up in the **birth_date** fields.

12 CEEDAYS is called again to convert the **birth_date** value from the input record to an integer value Lilian date.

13 CEEDATE is called again to convert the date to a 4-digit year date.

For example, a record containing the birth_date value of "04/05/09", that is 5th April, 1909, produces a date value of **04/05/1909**.

14 After these statements the **curr_age** field 15 contains the value **87**.

This example is a simple, but short-term, approach to the Year2000 problem using the century window feature of LE/VSE with C/VSE.

5.13 DOS/VS RPG II Considerations

Operations such as TIME, UDATE and UYEAR deliver a 2-digit year.

The solution for RPG II users is to call the new subroutine, ILNY224, which supports translation of a 2-digit field to a 4-digit field. Please refer also to 4.21, “DOS/VS RPG II” on page 45.

This subroutine makes use of the YEAR224 macro and subroutine IJBY224, that are provided by VSE/ESA 1.4 and higher. The RPG programmer specifies a window between 0 and 99, which is interpreted as the forward width of a 100 year window relative to the current year. The program parameters are as follows:

Input: YEAR to be translated (two characters)

Input: forward value for the 100 years window (two characters)

Output: translated YEAR including century (four characters).

The parameter sequence and field length are mandatory. The data format expected as input and returned by ILNY224 is Zoned Decimal (F0F1). The subroutine is invoked from the RPG II main program via the CALL statement. ILNY224 follows the RPG II linkage conventions and enables the programmer to test the indicator specified in position 56 and 57 of the CALL statement specification.

In the following sample RPG II program, there are a number of fields containing dates. These date fields are stored in the format of DDMMYY or MMDDYY on the magnetic tape files and do not carry century information. The dates may or may not be required to be converted to full century/year format YYYY depending upon the logic and requirement of the application.

This program shows the coding required to call the IBM supplied access module ILNY224 which sets up and invokes the YEAR224 macro in the format YEAR224 (LINK=YES,NAME=ILNYCONV).

```
H                                                                PROGAA
FBR0200 IP F 600  24     TAPE   SYS008S
FBR0300 IP F 600  24     TAPE   SYS009S
FBALREP O  F     132 OF  PRINTERSYSLST
IBR0200 NS 01
I                                1   1 RC1
I                                2   2 ACCNT1L1
I                                3   80DRAWN1           1
I                                9  140CHQNO1
I                               15  222CHQAM1
I                               23  24 SPARE1
IBR0200 NS 02
I                                1   1 RC2
I                                2   2 ACCNT2L1
I                                3   80PAID2            2
I                                9  140CHQNO2
I                               15  222CHQAM2
I                               23  23 COMMNT
I                               24  24 SPARE2
C  L1 01           Z-ADD0        ACCTOT    92
C  L1 02           Z-ADD0        PRETOT    92
C  01     COUNT1   ADD  1        COUNT1    40 98
C  02     COUNT2   ADD  1        COUNT2    40 98
C  01     COUNT1   COMP 1                        50
C  02     COUNT2   COMP 1                        50
C  50
COROF               SETON                    99
C  99               EXCPT
C                   SETOF                    99
C  02     COMMNT   COMP ¢1¢                      11
C  02     COMMNT   COMP ¢2¢                      12
C  02     COMMNT   COMP ¢3¢                      13
C  01     COUNT3   ADD  1        COUNT3    40
C  02     COUNT4   ADD  1        COUNT4    40
C  01     CHQAM2   ADD  TOTCHQ   TOTCHQ    92
C  02     CHQAM2   ADD  TOTPRE   TOTPRE    92
C  01     CHQAM1   ADD  ACCTOT   ACCTOT
C  02     CHQAM2   ADD  PRETOT   PRETOT
C  01               MOVE DRAWN1   YRCHK     2            3
C  02               MOVE PAID2    YRCHK                  4
C                   MOVE ¢50¢     WINDOW    2            5
C                   MOVE ¢  ¢     YEAR4     4            6
C                   CALL ¢ILNY224¢                 10    7
C                   PARM          YRCHK                  8
C                   PARM          WINDOW                 9
C                   PARM          YEAR4                  10
C N10               GOTO NOERR
C* WE HAVE A PROBLEM HERE - R15 IS NOT 0 -SEE YEAR224 MACRO
C* FOR REGISTER CONVENTIONS - WE WILL TERMINATE.
C         NOERR     TAG
```

Figure 13 (Part 1 of 2). A Sample RPG II Bank Reconciliation Program

```
     OBALREP E  1  99
     O                                66 ¢MY COMPANY¢
     O       E  1  99
     O                                68 ¢BANK RECONCILIATION  DATE -¢
     O                              UDATE Y 77                      11
     O       E  1  99
     O                                77 ¢BALANCE -¢
     O       E  2  99
     O                                 8 ¢BANK A/C¢
     O                                33 ¢DATE           CHQ NO¢
     O                                96 ¢AMOUNT       COMMENT¢
     O       D  1  01
     O                              DRAWN1Z  8
     O                              CHQNO1Z 33
     O                              CHQAM12 77
     O       D  1  02
     O                              PAID2 Z  8
     O                              CHQNO2Z 33
     O                              CHQAM22 77
     O                            11        95 ¢CANCELLATION¢
     O                            12        95 ¢PAYMENT STOPPED¢
     O                            13        95 ¢DELETION¢
     O       T  1  L1
     O                            01TOTCHQ2 77
     O                            02TOTPRE2 77
     O                            01COUNT3Z 33B
     O                            02COUNT4Z 33B
     O       T  1  LR
     O                              CHQTOT2 77
     O                              PRETOTZ 96
```

Figure 13 (Part 2 of 2). A Sample RPG II Bank Reconciliation Program

1 2 These two definitions contain dates. The may or may not require conversion to full century/year formats.

3 4 These instructions set up the field for input to the module ILNY224. This field YRCHK contains the YY to be complemented with century information.

5 This field WINDOW contains the value of the forward width of the window.

6 This field YEAR4 will contain the century/year information returned from the YEAR224 macro.

7 Subroutine ILNY224 will build field YEAR4 according to YRCHK and WINDOW.

11 This field UDATE will be in the format of DD/MM/YY or MM/DD/YY as returned from COMREG. The year format of YY will only return a 2-digit year format. If desired, this date may be converted using the ILNY224 callable subprogram to full YYYY format. However, UDATE will only ever return the year in 2-digit format. The same applies to the RPG II specification UYEAR.

Chapter 6. Migration Considerations

A complete migration to a Year2000 enabled system consists of the following migration steps:

- Plan for migration
- Migrate the operating system to a Year2000 enabled system
- Test the new operating system
- Migrate your applications to make them Year2000 enabled
- Test the migrated applications
- Port the tested applications to a Year2000 enabled production system.

6.1 Planning Considerations

To accomplish a successful migration you should have a well prepared plan available to migrate your operating system and your applications.

6.1.1 Migrate the Operating System

The effort to migrate to VSE/ESA 2.2 or VSE/ESA 1.4 depends on the current level of your VSE system. Planning for a migration from a VSE/SP system is more complex than planning for an upgrade via PTF installation of a current VSE/ESA. For assistance in understanding migration paths refer to the manual *VSE/ESA Planning*.

When planning to migrate your VSE operating system, contact vendors to make sure that all your non-IBM programs support the new VSE/ESA environment to which you are migrating.

As well as consulting *VSE/ESA Planning* you should also take the following into consideration:

- If you are migrating from a very old release of VSE components such as CICS and VTAM, and back level versions of optional products, it may be necessary to order and reference intermediate release or migration guides.
- If you have a component or optional product that has been superseded by another product, you should be aware of possible functionality changes. In this case, you should consult the appropriate release or migration guide for that product.

6.1.2 Migrate the Applications

1 Plan for migration

- Determine the sequence of steps
- Review the migration procedure
- Determine the resources/time required
- Assign individuals/organizations
- Document the migration sequence
- Develop a schedule for migration to production mode.

2 Examine all data changes with date format

- Determine the source of the data in existing systems
- Determine the data that can be converted automatically
- Determine the data that must be converted manually.

3 Design bridges/interfaces

- Design bridges/interfaces to application packages
- Design bridges/interfaces to reusable applications
- Design bridges/interfaces to old systems that will coexist with new
- Design tests for the verification and validation of these bridge facilities.

4 Design procedures for manual data conversion

- Update documents/procedures used for manual data entry
- Determine checking mechanism for manual data entry
- Design new screens for new date format for manual data entry
- Update software to load data entered manually
- Check estimates for impact of new date format on data entry.

5 Design procedures for automated data conversion

- Design new software or use automated tools for automated data conversion
- Determine a checking mechanism for accuracy
- Design recovery procedures for conversion of data errors
- Estimate resources/time for automated data conversion

6 Develop the data conversion systems

- Develop subsystems to convert existing data
- Develop new subsystems for the entry of new data
- Develop bridges/interfaces to old systems that will remain in production
- Develop bridges/interfaces to applications and reusable modules
- Verify and validate the accuracy of the data conversion systems.

7 Plan the hardware installation of new systems, if needed

8 Plan for final system testing

- Determine the testing strategy
- Develop the detailed test plan and schedule
- Determine the types of tests to be conducted
- Plan the testing environment
  - Design the migration tests for systems and applications
  - Determine what testing software/tools will be used
  - Determine testing libraries to hold new programs/jobstreams
  - Install needed testing software
  - Build test libraries and test data

- Coordinate testing with development and system staff.

9 Documentation and training

- Update standard guidelines
- Update technical documentation
- Update production procedures
- Update user documentation.

6.1.3 Special Testing Considerations

When it comes to testing, the best advice is test, test, and test again. Testing is considered to be 40 to 60 percent of the entire Year2000 conversion project. Here is a list of things to consider when testing:

- Be careful setting the date ahead - data sets, passwords, user IDs, and product licenses may expire
- Use a separate system for testing
- Do not share data sets between test and production systems
- Test both the time period 1999 to 2000 and the time period after 2000
- Do not forget 2000 is a leap year - test 29 February 2000
- Handle your backup data and programs correctly
- Have a method to age the date information in your applications data to ensure thorough testing

6.2 Migrating the Operating System

The way the Year2000 changes are implemented in VSE/ESA and its subsystem allow for a smooth migration. Migrating the operating system to a Year2000 enabled system is not different from a normal release upgrade.

There are three methods that can be used to upgrade your VSE system. The first method is the Fast Service Upgrade process, the second is the Initial Installation process, the third is upgrading via PTFs. More information about these upgrade methods and the advantages and disadvantages for each of them is given in the manual *VSE/ESA Planning*.

The redbook *Migration to VSE/ESA 2.1 - Why and How* will give detailed information how to perform the migration.

It is important to notice that the FSU process does not refresh the optional products. If you choose to use the FSU method to migrate your system you must reinstall or upgrade your optional products if they are not at the correct level.

6.2.1 Fast Service Upgrade (FSU)

If you are migrating from a VSE/ESA 1.3.x. or a later version of VSE/ESA you can use the Fast Service Upgrade (FSU) dialog from the Interactive Interface to upgrade your system.

The FSU process does not update the optional programs and it is not aware of any private system modifications.

The IPL and JCL procedures as well as the LIBDEF procedure of VSE/ESA V2 have changed compared to VSE/ESA 1.3. They are not updated by the FSU process.

To be able to use the FSU method your current system must meet the following requirements:

- Your current system must be VSE/ESA 1.3.x or later
- Your supervisor mode must be MODE=ESA

  An FSU is not possible if the current supervisor mode is 370, VM or VMESA
- The predefined environment must not be 1, 2 or 5

  The environment is selected when the VSE/ESA 1.3.x was installed
- The system disks DOSRES and SYSWK1 have to be supported by your target system.
- The current system must have the standard system layout of a VSE/ESA 1.3.x system including the VSE/ESA library structure and VSE/VSAM catalogs.

6.2.2 Initial Installation

This is the method that you have to use if the requirements mentioned above under Fast Service Upgrade are not met or if the initial installation method is more convenient. The manuals *VSE/ESA Installation* and *VSE/ESA Planning* should be used to perform the initial installation.

6.2.3 PTF installation

If you are currently on VSE/ESA 1.4 or VSE/ESA 2.1 you may apply the necessary PTFs to enable your system for Year2000.

Appendix A, “Product Information for VSE/ESA 2.1.x and 1.4.x” on page 149 lists for both VSE/ESA 2.1.x and VSE/ESA 1.4.x the base and optional products. For each product there is an indication whether it is affected by the Year2000 or not. If there is a PTF available to upgrade the product, you will find the PTF number listed there.

6.3 Testing the Year2000 Enabled VSE/ESA

To be able to test your Year2000 enabled operating system you should provide for:

1. A separate, isolated test system.

   This can be:

   - A separate logical partition (LPAR). VSE/ESA can run with the TOD-clock set in the future in a Processor Resource/Systems Manager (PR/SM) LPAR. If you have a non-IBM system that has a feature similar to PR/SM you should check that it is functionally the same.
   - A virtual machine running under VM/ESA. If you are running VM/ESA V2, you can set the TOD-clock in the future for a guest operating system. To allow for the virtual TOD-clock setting add the TODENABLE parameter in the OPTIONS card in the directory entry for the guest virtual machine.

- A separate processor. This can be any type of processor that is Year2000 enabled, such as an IBM PC Server S/390 or an RS/6000 and S/390 Server-on-Board.

If you are not able to provide for a test system you have to make a complete backup of the system disks when switching from the test environment to the production environment or vice versa.

2. Disk space to allow for test data sets.

   If you cannot provide test user data sets, and you use production data sets during the test, backup the data sets before starting the test, and restore them after the tests are done.

Two types of testing should be applied after the Year2000 enabled system is installed.

1. Requirements Testing
2. Regression Testing

6.3.1 Requirements Testing

Apply requirements testing to verify that the system is performing its functions correctly in respect of Year2000 enabling. For this type of testing you have to SET the date in the future. Requirements testing includes recovery testing. You have to make sure that all recovery procedures are functioning correctly after switching to the Year 2000.

The effort required for requirements testing does not depend on the migration path. It is determined by the system environment.

Warning

If you are using security products check first that setting the date in the future will not cause immediate loss of system programmer or end user access. Be careful with file expiration dates when setting the date backwards and forwards.

Check also if there are no expiration dates for licences. Setting the date in the future can cause licences to expire.

Sharing Data

When sharing data among systems, careful consideration of the overall environment is required:

With a **Year2000-enabled** VSE/ESA system, it is safe to use expiration dates beyond 99/365, since century information is taken into account. However, if you are sharing data with a **non-enabled** system, such data is again exposed to be overwritten without warning, since that system will consider all such data as expired.

This shows, that careful testing for the year 2000 requires isolation and dedicated resources.

Testing Scenarios

The scenarios for Year2000 testing depend heavily on the system environment. Some Year2000 scenarios that are common for most installations are suggested here:

- Test the setting and display of special dates using the SET DATE command. Verify if the date is set and displayed correctly. Some interesting dates are:
  - SET DATE=2/29/1900 should fail. The year 1900 is not a leap year.
  - SET DATE=2/29/1996 should succeed. The year 1996 is a leap year.
  - SET DATE=2/29/2000 should succeed. The year 2000 is a leap year.
  - SET DATE=01/01/00 should set the system date to 01/01/2000
  - SET DATE=09/17/2024,CLOCK=23/53/48 and later dates should be rejected because they will cause a TOD-clock overflow.

    To verify if the date is correctly set you can use the DSPLY command. ′DSPLY 14′ displays 16 bytes starting at address 14. The first four bytes are the COMREG address, for example, X′00000590′. ′DSPLY 590′ shows the first 16 bytes of the COMREG. The first 11 bytes in the COMREG are JOBDATWC (JOBDATE plus century information). Its format is dd/mm/yy/cc or mm/dd/yy/cc depending on the DATE option of the STDOPT statement. Verify if the cc value is correct after each SET DATE= command. Another way to obtain the century information from the COMREG is from field SYSCENT. To display the SYSCENT value add X′64′ to the COMREG address. In our case this is X′5F4′. ′DSPLY 5F4′ will display 16 bytes where the first two bytes contain the century information in the form cc (19 or 20).
- Set the TOD-clock to test automated functions that are activated on a regular basis, for example by some scheduling products:
  - Daily
  - Weekly
  - Monthly
  - Quarterly
  - Annual
- Set the system date to 12/31/1999 and let the system run in to the Year 2000 and verify if it is functioning correctly. Terminate CICS abnormally after the century switch and test if recovery, backward and forward, is functioning correctly. Test all recovery procedures for all different types of database products.
- Set the system date after 12/31/1999 and verify that file expiration is functioning correctly. Verify that all files that are expected to expire are effectively expired.
- Set the system date to 12/31/1999 and verify that file expiration is functioning correctly. Check that all files that expire on 12/31/1999 are really expired files.
- Verify that programs that are scanning SYSLST or the hardcopy file for messages and/or dates do function correctly after the system is Year2000 enabled.

6.3.2 Regression Testing

Apply regression testing to ensure that all aspects of the system remain functionally correct and to ensure that all applications are functioning as before. To do this type of testing you don't have to IPL the system with the TOD-clock SET into the future.

The effort required to perform regression testing depends on the migration path. If the system is enabled for Year2000 by applying PTFs to a VSE/ESA 2.1 or a VSE/ESA 1.4 system, the effort to perform regression testing will be low. If a VSE/SP is migrated to a VSE/ESA 2.2, regression testing is very important.

When regression testing is completed successfully, you can migrate the production system. When the production system is Year2000 enabled it is ready to receive migrated applications.

6.4 Migrate Your Applications to Make them Year2000 Enabled

There are three basic choices to migrate applications.

1. Change the application.

   This type of solution is a short-term solution. Apply windowing techniques and/or bridge programs when choosing this type of solution.

2. Change the application and the data.

   This is a long-term solution. It is more complex to implement. Consider the use of **bridge programs** when implementing this type of solution. The use of bridge programs is discussed in Chapter 2, "The Long-Term Solution" on page 9.

3. Invest in a new application.

   This may be the only available solution if you have applications or packages where the source code is no longer available.

6.4.1 Language Considerations

When migrating to an LE/VSE environment, your 'old' languages will probably coexist with the new languages. For DOS PL/I and PL/I VSE this is not a problem, since there are no duplicate module names. But for COBOL and C there are module name conflicts when old and new products are installed concurrently. Therefore specification of the LIBDEF search chain is important. Assuming these language products are installed in the default sublibraries as follows:

Product	Sublibrary
LE/VSE Base	PRD2.SCEEBASE
LE/VSE COBOL CICS	PRD2.SCEECICS
COBOL/VSE Compiler	PRD2.PROD
VS COBOL II Compiler	PRD2.DBASE
VS COBOL II Library	PRD2.DBASE
VS COBOL II CICS	PRD2.CICSR
DOS/VS COBOL	PRD2.PROD
C/370	PRD2.PROD
C/VSE	PRD2.DBASE

Then the following LIBDEF SEARCH order is required:

- For COBOL/VSE batch (the new LE-enabled COBOL)

  For compilation, link-edit, and batch execution of COBOL/VSE programs, ensure that PRD2.SCEEBASE is followed by PRD2.PROD in the LIBDEF search chain. If PRD2.DBASE is required in the search chain, ensure it is after these sublibraries.

  ```
  for example: // LIBDEF *,SEARCH=(... ,PRD2.SCEEBASE,PRD2.PROD, ....
                                       PRD2.DBASE, ....)
  ```

- VS COBOL II for CICS (the non-LE COBOL)

  VS COBOL II programs may be run under CICS/VSE using either the LE/VSE run-time support or the VS COBOL II run-time support.

  – To run using the LE/VSE run-time support:

    1. Translate your CICS command level COBOL programs using either the COBOL2 or ANSI85 options.
    2. Specify COBOL2=NO on your CICS/VSE SIT.
    3. Place the LE/VSE sublibraries **before** the VS COBOL II sublibraries in the LIBDEF search chain of the CICS/VSE startup job (or better yet, use only the LE/VSE sublibraries).

       ```
       For example: // LIBDEF *,SEARCH=(..,PRD2.SCEECICS,
                                          PRD2.SCEEBASE)

             or     // LIBDEF *,SEARCH=(..,PRD2.SCEECICS,
                                          PRD2.SCEEBASE,..,
                                          PRD2.DBASE)
       ```

  – To run without LE/VSE run-time support:

    1. Translate your CICS command level COBOL programs using either the COBOL2 or ANSI85 options.
    2. Specify COBOL2=YES on your CICS/VSE SIT.
    3. Place the LE/VSE sublibraries **after** the VS COBOL II sublibraries in the LIBDEF search chain of the CICS/VSE startup job (or better yet, use only the VS COBOL II sublibraries).

       ```
       For example: // LIBDEF *,SEARCH=(..,PRD2.CICSR,
                                          PRD2.DBASE)
             or     // LIBDEF *,SEARCH=(..,PRD2.CICSR,
                                          PRD2.DBASE,..,
                                          PRD2.SCEECICS,
                                          PRD2.SCEEBASE)
       ```

- C/VSE

  The search order in the JCL LIBDEF statement is important when C/VSE and C/370 coexist in a VSE/ESA system. Check the LIBDEF search order to ensure that you do not compile with the wrong compiler, prelink with the wrong prelinker or link-edit with the wrong run-time library.

  Similarly, if any C/370 compiler phases or if the C/370 prelinker phase have been placed in the SVA, you must either remove these phases from the SVA or, by specifying the SDL keyword at the appropriate place in the LIBDEF PHASE statement, ensure that the SVA is searched after the sublibrary containing C/VSE or LE/VSE.

6.4.2 Migration Tools

Consider to use migration tools if they are available. One that is currently available is COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE, program number 5785-CCC). See Chapter 7, “COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)” on page 105 for more information on CCCA/VSE.

6.5 Test the Migrated Applications

To test your applications you should provide:

1. A separate test system.

   This allows for application testing with the system date set in the future.

2. Test data sets.

   This allows for application testing with database dates in the future.

As for the operating system, two testing types apply for migrated applications; requirements testing and regression testing.

6.5.1 Requirements Testing

To perform requirements testing you need a separate system that you can IPL with a date set in the future. Test data sets or test databases are required containing test data with dates in the future.

Consider the following when applying requirements testing for migrated applications:

- Make sure that adequate dates are used during the test. Meaningful test dates can be different for each application and can vary if different solution techniques are used. If you use the fixed window technique you should test close to the window boundaries and close to the switch to the Year 2000.
- If you choose the long-term solution, which means changing both the date in your programs and in the database, make sure that your application can cope with old backup data.
- If you choose the long-term solution for one or more applications, don’t forget to apply regression testing for all applications, even those that did not change.
- Try to involve the end user when performing the final acceptance test. They are the most adequate people to judge if the application performs correctly.

6.5.2 Regression Testing

To perform regression testing you don’t have to IPL with a date set in the future. There is also no need for data sets or databases containing dates in the future.

If you have to test CICS/VSE applications and there is no test system available, this type of testing can be done in a separate CICS/VSE partition using test data sets or test databases.

6.6 Port the Migrated Applications to the Production System

Before you move the migrated applications to the production system, the production system has to be enabled for Year2000. If possible, try to move one application at a time to the production system. It will make problem determination and problem source identification easier.

When all applications are moved to the production system, you have a Year2000 enabled computing system.

Chapter 7. COBOL and CICS Command Level Conversion Aid for VSE (CCCA/VSE)

7.1 What CCCA/VSE Does

CCCA/VSE is an effective tool designed to make it easier to convert old COBOL source code and copy modules to the new COBOL Standard.

As supplied, CCCA/VSE helps you to convert COBOL source from:

Source Language	Source Release	Source Number
DOS/VS COBOL	3	5746-CB1
OS/VS COBOL	2	5740-CB1
VS COBOL II	1,1.1,2,3,3.1,3.2	5668-958

to:

ANSI 85	Target Release	Target Number
VS COBOL II	3 or later	5668-958
COBOL/VSE	1 or later	5686-068

CCCA/VSE is designed to identify and convert source code incompatibility, to reduce the effort required to convert programs, and to minimize conversion errors. The conversion process can be customized by users to meet unique conversion requirements.

CCCA/VSE identifies COBOL language elements and CICS statements in the input source programs that are:

- Not supported by the target language

  or

- Supported in a different way

  then

- Converts them to the equivalent in the target language

  or

- Removes them

  or

- Flags them.

CCCA/VSE provides facilities to:

- Convert most syntax differences between DOS/VS COBOL, or VS COBOL II R1 or R2 and the current release of VS COBOL II and COBOL/VSE.
- Convert EXEC CICS commands
- Remove and/or convert the base locator for linkage (BLL) section mechanism and references
- Eliminate conflicts between user-defined names and words reserved for VS COBOL II
- Convert both source programs and copy modules

- Create conversion management reports
- Produce a statement-by-statement diagnostic listing showing the result of the conversion process for each program
- Change and/or create COBOL conversion modules

Software Requirements

The software requirements for using CCCA/VSE are:

- VSE/ESA 1.2 and subsequent releases
- VS COBOL II Library 1.3 or later, or LE/VSE V1
- A SORT program capable of being used with the COBOL SORT verb
- VS COBOL II Compiler 1.3 or later, or COBOL/VSE V1
- DL/I or SQL/DS for preprocessing if you have DL/I or SQL/DS programs.

For details on the installation of CCCA/VSE see the manual *CCCA/VSE Installation and User's Guide*.

7.2 How CCCA/VSE Works

CCCA/VSE is a combination of CICS and batch applications. You use CCCA/VSE on-line CICS panels to:

- Define the type of conversion you want
- Submit a batch job to convert your programs

The conversion consists of three phases:

Phase 1: Analyze input source

At the start of the conversion job, phase 1:

- Translates the original source program into a set of character strings known as *tokenized source*
- Extracts copy members from the appropriate copy libraries
- For each item in the tokenized source, identifies whether conversion is required, and if so, which Language Conversion Program (LCP) to use.

Phase 2: Create change requests

For each item that needs converting, phase 2:

- Loads an LCP
- Runs an LCP
- Generates change requests

Phase 3: Apply changes and generate output

Finally, phase 3:

- Applies the change requests from phase 2, creating new source programs and, if required, new copy members
- Generates the Diagnostic listing

7.3 CICS Conversion

If the original source program includes CICS statements but does not use the BLL mechanism, the conversion is the same as for a non-CICS program.

If the original source program includes CICS statements and uses the BLL mechanism, a preliminary step is run before conversion that maps the BLLs and the records to which they point. This step uses a compiler - either VS COBOL II or COBOL/VSE - to determine the length of each 01-level item and the number of BLLs needed to address them. Only the program Linkage Section is compiled.

7.4 CCCA/VSE and Invoking Language Conversion Programs (LCPs)

In conversion phase 1, CCCA/VSE reads the input source program and creates a token record for each:

- COBOL word
- Literal
- Picture character-string
- Separator
- Comment line

CCCA/VSE checks whether each COBOL word in the input source program is in the COBOL Reserved Word file or not.

The COBOL Reserved Word file lists words that may invoke LCPs, and specifies:

Word type Where a word can occur in a COBOL program
Word change The LCP (if any) to invoke when a word occurs.

If CCCA/VSE finds the word in the COBOL Reserved Word file, it adds the word type and change code to the token record.

In conversion phase 2, CCCA/VSE reads the tokenized source. When a token record is encountered that has something other than change code 999, CCCA/VSE invokes the LCP that is indicated by the code.

The invoked LCPs generate detailed change requests for converting the input source program.

In conversion phase 3, CCCA/VSE applies the change requests to the input source program.

LCPs fall into one of five categories as shown in the following tables.

LCPs that convert the following CICS statements:

Table 7. LCP Category 1

EXEC	BLL references changed to ADDRESS OF
SERVICE RELOAD	Replaced by CONTINUE
ADD(851)	BLL references changed to POINTER facilities
COMPUTE	BLL references changed to POINTER facilities
MOVE	BLL references changed to POINTER facilities
SUBTRACT	BLL references changed to POINTER facilities

LCPs that convert the following COBOL statements:

Table 8 (Page 1 of 2). LCP Category 2

ACTUAL	ACTUAL KEY clause; replaced by RELATIVE
ALPHABETIC	Changed to ALPHABETIC-UPPER
APPLY	Remove APPLY clause in I-O CONTROL paragraph
ASSIGN	Change ASSIGN clause syntax
CBL	Modify compiler options
COPY	Convert ANS 68 syntax and adds COPY information
CORR/CORRESP	Multiple MOVE changed to separate MOVEs
CURRENT-DATE	Replaced by DATE special register and reformatted or by EXEC CICS ASKTIME in a CICS program
DATE	Add hyphen if missing in DATE COMPILED and DATE WRITTEN
DATE-COMPILED	Add period after header if missing
DISP	In OPEN and CLOSE statements option deleted
ENTER	Obsolete element removed
ENVIRONMENT	Add Configuration Section header if needed; relocate it if in wrong place
EXAMINE	Change EXAMINE to INSPECT
EXHIBIT	EXHIBIT statement changed to DISPLAY
FD	Convert FD entry, check LABEL clause
FILE-LIMIT	Delete FILE-LIMIT clause
FILE-LIMITS	Delete FILE-LIMITS clause
JUST	Value literal is changed for ANS 68 syntax
JUSTIFIED	Value literal is changed for ANS 68 syntax place
LEAVE	In OPEN statement option deleted
LINE/LINES	Word removed in WRITE BEFORE/AFTER ADVANCING mnemonic
LVL88	Put 88 level value string in quotes, if missing
MEMORY	Remove MEMORY SIZE clause
MULTIPLE	For multiple reel/unit ANS 68 CLAUSE deleted
NATIVE	Add ALPHABET word in SPECIAL-NAMES
NOMINAL	Replaced by RELATIVE or clause deleted
NOTE	Change to comment
OPEN	Add FILE STATUS test for VSAM files that have had FILE STATUS clauses added
OTHERWISE	Clause of IF statement replaced by ELSE
POSITIONING	AFTER POSITIONING clause of WRITE statement replaced by AFTER ADVANCING clause
PROCEDURE	An Error Declarative Section is added for each file that is to be converted to VSAM
PROCESSING	Delete PROCESSING MODE for VSAM files that have had FILE STATUS clauses added
READ	Move NOMINAL to RECORD KEY for ISAM files
REDEFINES	Remove clause in FD
REMARKS	Change to comment
REREAD	In OPEN statement option deleted
RESERVE	Change RESERVE syntax ANS 88
REWRITE	MOVE NOMINAL KEY TO RECORD KEY for ISAM files

Table 8 (Page 2 of 2). LCP Category 2

SAME	Change SAME AREA to SAME RECORD AREA
SD	Convert SD ENTRY, LABEL clause
SEARCH	Changed to SEARCH WHEN KEY
SEEK	Statement deleted
SEQUENCE	Add ALPHABET word in SPECIAL-NAMES
SPECIAL-NAMES	Add SPECIAL NAMES
STANDARD-1	Add ALPHABET word in SPECIAL-NAMES
START	MOVE NOMINAL TO RECORD KEY
THAN	Removed after > or < relational operators
THEN	Delete THEN between statements
TIME-OF-DAY	Replaced by TIME special register or by an EXEC CICS ASKTIME in a CICS program and reformatted
TRACK-AREA	TRACK-AREA removed
TRANSFORM	Replaced by INSPECT statement
UPSI-0	Replace UPSI switch by condition name
UPSI-1	Replace UPSI switch by condition name
UPSI-2	Replace UPSI switch by condition name
UPSI-3	Replace UPSI switch by condition name
UPSI-4	Replace UPSI switch by condition name
UPSI-5	Replace UPSI switch by condition name
UPSI-6	Replace UPSI switch by condition name
UPSI-7	Replace UPSI switch by condition name
USING	START...USING KEY USING word deleted
VALUE	Remove sign if PICTURE unsigned
VALUES	Changed to VALUE if not used in level 88
WRITE	MOVE NOMINAL KEY TO RECORD KEY for ISAM files

LCPs that partially convert the following COBOL statements:

Table 9. LCP Category 3

NOT	Change abbr. relation condition ANS 68 syntax
ON	ON integer changed to IF ON integer UNTIL integer changed to IF other cases flagged

LCPs that flag COBOL statements

The flagged COBOL statements may be put into several categories:

1. Language elements from functions of the source language that are no longer supported in the target language, and have no replacement or equivalent in the target language. Therefore, a conversion cannot be performed.

Table 10. LCP Category 4 - Flagged COBOL Report Writer statements

GENERATE	Generate statement flagged
INITIATE	statement flagged
LINE-COUNTER	statement flagged
PAGE-COUNTER	statement flagged
PRINT-SWITCH	statement flagged
REPORT	statement flagged
REPORTS	statement flagged
TERMINATE	statement flagged
USE	flagged USE BEFORE REPORTING

You can avoid getting these statements flagged if you use the COBOL Report Writer Precompiler (program offering # 5798-DYR). This precompiler can permanently convert Report Writer statements to valid COBOL statements that can be compiled in COBOL/VSE, or it can be used to precompile applications containing Report Writer statements so the code will be acceptable to the COBOL/VSE compiler.

2. LCPs corresponding to information that are used internally during migration:

Table 11. LCP Category 5

ACCESS	Update file information in CONTROL file
ASCENDING	Save key-id for SEARCH...WHEN
DECLARATIVES	Check section end
DEPENDING	Save name of object of ODO
END-OF-CONVERSION-1	Add WRITE...AFTER ADVANCING section
END-OF-CONVERSION-2	Add data items in WS
END-OF-CONVERSION-3	Add data items in WS
END-OF-CONVERSION-4	Add SPECIAL NAMES
END-OF-CONVERSION-5	List data names to be checked
ENVIRONMENT	Set flag when entering Environment Division
ID	Set flag when entering ID Division
IDENTIFICATION	Set flag when entering ID Division
INDEXED	Store indexes in work file; used by the IN and the OF LCP
INPUT-OUTPUT	Set flag when entering I/O Section
LINKAGE	Set flag when entering Linkage Section
PROGRAM-ID	Update PROGRAM file
RECORD	Update key record
SECTION	Set flag when entering a section
SELECT	Update CONTROL file
WORKING-STORAGE	Set flag when entering WS section
01	Save RECORD name of FD

7.5 CCCA/VSE Conversion Example

7.5.1 Input DOS/VS COBOL Program

A batch DOS/VS COBOL program, that was error free and had been compiled successfully and placed into a production environment, was used as the input source program to the CCCA/VSE conversion routines. The source program used for testing was as follows:

```
 IDENTIFICATION DIVISION.
 PROGRAM-ID. PY06.                    1
 AUTHOR.
 DATE-WRITTEN. 4TH JULY 1977.
 DATE-COMPILED.
 REMARKS. THIS PROGRAM ACCEPTS A RANGE OF CHEQUE NUMBERS FROM
*    THE SPO IN THE FORMAT NNNNNN, NNNNNN. A LIST OF CHEQUES
*    DRAWN AND A TAPE-FILE FOR THE BANK RECONCILIATION SYSTEM
*    IS PRODUCED.
*
*
*****************************************************************
*         P R O G R A M   M O D I F I C A T I O N S             *
* PROG.   DATE   PROB NO.   COMMENT                             *
*       07/10/83           CHANGE FILENAME ON SELECT STATEMENT TO*
*                MATCH FILE NAMES IN NEW VERSION OF UDC PROGRAMS.*
*       01/10/84           ALLOW PROGRAM TO DECIDE WHETHER THE   *
*                INPUT RECORDS ARE FOR CASHFLOW OR NORMAL BANK.  *
*                IF DATA IS CASHFLOW, THEN OUTPUT VIA CASHFLOW   *
*                RECORD FORMAT WITH ADDED FIELDS TO 62 BYTES.    *
*                ALLOW INPUT OF NEW TYPES OF RECORDS TO IDENTIFY *
*                CASHFLOW PAYROLL COMPANIES. THE RECORD TYPES ARE*
*                A †CH† TYPE TO ALLOCATE THE GENLAC ACCT CODE FOR*
*                A PAYROLL COMPANY, AND A COMMENT RECORD, WITH AN*
*                †*† IN COLUMN 1 TO COMMENT THE INPUT DATA DECK. *
*****************************************************************
 ENVIRONMENT DIVISION.
 CONFIGURATION SECTION.
 SPECIAL-NAMES.
     C01 IS CHANNEL-1
     C12 IS CHANNEL-12.
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT CHEQUE-FILE
         ASSIGN TO SYS020-DA-3340-D811F
         ACCESS MODE IS SEQUENTIAL
         ORGANIZATION IS INDEXED
         RECORD KEY IS IBM-CHEQUE-KEY
         FILE STATUS IS VSAM-STATUS.
     SELECT HEADER-CARD ASSIGN TO SYS007-UR-2501-S.       2
     SELECT BANK-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.
     SELECT CASH-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.
     SELECT CHEQUE-LIST ASSIGN TO SYS010-UR-3203-S.       3
*
     EJECT
 DATA DIVISION.
 FILE SECTION.
 FD  CHEQUE-FILE
     LABEL RECORDS ARE STANDARD.                          4
*
```

Figure 14 (Part 1 of 10). Input Source Program

```
 01  IBM-CHEQUE-REC.
     03  IBM-CHEQUE-KEY          PIC X(11).
     03  FILLER                  PIC X(127).
*
 FD  HEADER-CARD
     RECORD CONTAINS 80 CHARACTERS
     LABEL RECORDS ARE OMITTED                              5
     RECORDING MODE IS F.
*
 01  IBM-HEADER-REC              PIC X(80).
*
 FD  BANK-TAPE
     RECORD CONTAINS 24 CHARACTERS
     BLOCK CONTAINS 25 RECORDS                              6
     LABEL RECORDS ARE STANDARD
     RECORDING MODE IS F.
*
 01  BANK-REC.
     03  BT-REC-CODE             PIC X.
     03  BT-ACCOUNT              PIC X.
     03  BT-DATE-DRAWN           PIC 9(6).
     03  BT-CHQE-NO              PIC 9(6).
     03  BT-AMOUNT               PIC 9(7)V99.
     03  BT-ORIGIN               PIC X.
*
 FD  CASH-TAPE
     RECORD CONTAINS 62 CHARACTERS
     BLOCK CONTAINS 10 RECORDS                              7
     LABEL RECORDS ARE STANDARD
     RECORDING MODE IS F.
*
 01  CASH-REC.
     03  CT-REC-CODE             PIC X.
     03  CT-ACCOUNT              PIC X.
     03  CT-DATE-DRAWN           PIC 9(6).
     03  CT-CHQE-NO              PIC 9(6).
     03  CT-AMOUNT               PIC 9(7)V99.
     03  CT-ORIGIN               PIC X.
     03  CT-CFLOW-ACCT1          PIC X(10).
     03  CT-CFLOW-VAL1           PIC S9(7)V99.
     03  CT-CFLOW-ACCT2          PIC X(10).
     03  CT-CFLOW-VAL2           PIC S9(7)V99.
*
 FD  CHEQUE-LIST
     RECORD CONTAINS 133 CHARACTERS                         8
     LABEL RECORDS ARE OMITTED
     RECORDING MODE IS F.
*
```

Figure 14 (Part 2 of 10). Input Source Program

```
 01  PRINT-REC                    PIC X(133).
*
     EJECT
 WORKING-STORAGE SECTION.
 77  VSAM-STATUS                   PIC X(2).
     88  OPEN-OK                   VALUE ZERO.
 77  FILE-NAME                     PIC X(10)  VALUE SPACES.
 77  WS-PREV-END                  PIC 9(6).
 77  WS-CODE                      PIC X.
 77  WS-SUB                       PIC 9(4).
 77  WS-CFLOW-LIMIT               PIC 9(4) VALUE 10.
 01  HEADER-REC.
     03  HC-TYPE                  PIC XX.
     03  HC-DATE.
         05  HC-DAY               PIC 99.
         05  HC-MONTH             PIC 99.
         05  HC-YEAR              PIC 99.
     03  HC-BANK-CODE             PIC X.
         88   VALID-BANK          VALUE ¢T¢ ¢S¢ ¢W¢ ¢Z¢.
     03  FILLER                   PIC X(71).
 01  CASHFLOW-REC REDEFINES HEADER-REC.
     03  CH-TYPE                  PIC XX.
     03  CH-COMPANY-NO            PIC X(4).
     03  CH-ACCT1                 PIC X(10).
     03  CH-ACCT2                 PIC X(10).
     03  FILLER                   PIC X(54).
 01  COMMENT-REC REDEFINES HEADER-REC.
     03  CR-TYPE                  PIC X.
     03  FILLER                   PIC X(79).
 01  CHEQUE-REC.
     03  CF-CO-NUMBER             PIC X(4).
     03  CF-EMPLOYEE-NO           PIC X(6).
     03  CF-CODE                  PIC X.
     03  CF-NAME                  PIC X(30).
     03  FILLER                   PIC X(90).
     03  CF-CHQE-VALUE            PIC S9(5)V99.
*
 01  HEADING-1.
     03  FILLER                   PIC X.
     03  FILLER                   PIC X(53)      VALUE
         ¢  PY06                                 PAYROLL CHEQUE¢.
     03  FILLER                   PIC X(38)      VALUE
         ¢ REGISTER                          DATE ¢.
```

Figure 14 (Part 3 of 10). Input Source Program

```
           03  H1-DATE.
               05  DAY1                PIC 99.
               05  FILLER              PIC X          VALUE ¢/¢.
               05  MONTH               PIC 99.
               05  FILLER              PIC X          VALUE ¢/¢.
               05  YEAR                PIC 99.
           03  FILLER                  PIC X(28)      VALUE
               ¢                       PAGE ¢.
           03  H1-PAGE                 PIC ZZ9.
           03  FILLER                  PIC XX         VALUE SPACES.
       *
        01  HEADING-2.
           03  FILLER                  PIC X.
           03  FILLER                  PIC X(44)      VALUE
               ¢  COMPANY NUMBER            EMPLOYEE       ¢.
           03  FILLER                  PIC X(44)      VALUE
               ¢                   NAME                    ¢.
           03 FILLER                  PIC X(44)      VALUE
               ¢     CHEQUE NUMBER           CHEQUE VALUE  ¢.
       *
        01  DETAIL-LINE.
           03  FILLER                  PIC X(8).
           03  DL-CO-NUMBER            PIC X(4).
           03  FILLER                  PIC X(19).
           03  DL-EMPLOYEE-NO          PIC X(6).
           03  FILLER                  PIC X(14).
           03  DL-NAME                 PIC X(30).
           03  FILLER                  PIC X(16).
           03  DL-CHQE-NO              PIC 9(6).
           03  FILLER                  PIC X(16).
           03  DL-CHQE-VALUE           PIC Z(4)9.99.
           03  FILLER                  PIC X(4).
       *
        01  CONTROL-1.
           03  FILLER                  PIC X.
           03  FILLER                  PIC X(20)      VALUE
               ¢  CONTROLS         ¢.
           03  FILLER                  PIC X(112)     VALUE SPACES.
       *
        01  CONTROL-2.
           03  FILLER                  PIC X.
           03  FILLER                  PIC X(44)      VALUE
               ¢     TOTAL NUMBER OF CHEQUES              ¢.
           03  C2-NUMBER               PIC Z(4)9.
           03  FILLER                  PIC X(83)      VALUE SPACES.
       *
        01  CONTROL-3.
           03  FILLER                  PIC X.
           03  FILLER                  PIC X(39)      VALUE
               ¢     TOTAL VALUE OF CHEQUES           ¢.
           03  C3-VALUE                PIC Z(6)9.99.
           03  FILLER                  PIC X(83)      VALUE SPACES.
       *
```

Figure 14 (Part 4 of 10). Input Source Program

```
 01  WS-DATE.
     03  DAY1                    PIC 99.
     03  MONTH                   PIC 99.
     03  YEAR                    PIC 99.
*
 01  WF-FLAGS.
     03  WF-CFLOW-BANK-IND       PIC X.
*
 01  WS-COUNTERS.
     03  WS-PAGE                 PIC 999       VALUE ZERO.
     03  WS-LINE                 PIC 99        VALUE ZERO.
     03  WS-NUMBER               PIC 9(5).
     03  WS-VALUE                PIC 9(7)V99.
 01  WS-RANGE.
     03  WS-START-RANGE          PIC 9(6).
     03  WS-START-RANGX REDEFINES WS-START-RANGE PIC X(6).
     03  WS-COMMA                PIC X.
     03  WS-END-RANGE            PIC 9(6).
     03  WS-END-RANGX REDEFINES WS-END-RANGE    PIC X(6).
 01  WS-RANGX REDEFINES WS-RANGE  PIC X(13).
*
 01  WA-CFLOW-REC-ARRAY.
     03  WA-CH-REC-DATA OCCURS 10.
         05  WA-CH-TYPE          PIC XX.
         05  WA-CH-COMPANY-NO    PIC X(4).
         05  WA-CH-ACCT1         PIC X(10).
         05  WA-CH-ACCT2         PIC X(10).
*
     EJECT
 PROCEDURE DIVISION.
 000-CONTROL SECTION.
 010-CTL.
     PERFORM 050-OPEN.
     PERFORM 150-ACCEPT-VALUES.
     PERFORM 100-WORK.
     PERFORM 650-INVALID.
     PERFORM 700-TOTALS.
     PERFORM 900-CLOSE.
     STOP RUN.
 049-CTL-EXIT.
     EXIT.
*
*
*
```

Figure 14 (Part 5 of 10). Input Source Program

```
050-OPEN SECTION.
060-OPEN.
    MOVE ¢ZA¢ TO VSAM-STATUS.
    OPEN INPUT CHEQUE-FILE
    IF  NOT OPEN-OK
        MOVE ¢PY1200¢ TO FILE-NAME
        GO TO OPEN-END.
    OPEN INPUT HEADER-CARD.
***  OPEN OUTPUT BANK-TAPE.  DELAY OPENING UNTIL CFLOW/BANK!
    OPEN OUTPUT CHEQUE-LIST.
065-SET.
    MOVE SPACES TO PRINT-REC DETAIL-LINE.
    MOVE SPACES TO WS-RANGE WA-CFLOW-REC-ARRAY.
    MOVE ZEROS TO WS-PAGE WS-LINE WS-NUMBER.
    MOVE ZEROS TO WS-VALUE WS-PREV-END.
    MOVE ¢B¢ TO WF-CFLOW-BANK-IND.
070-READ.
    READ HEADER-CARD INTO HEADER-REC
        AT END GO TO 091-ERROR.
    IF CR-TYPE = ¢*¢
        GO TO 070-READ.
    IF HC-TYPE NOT = ¢PH¢
        GO TO 090-ERROR.
    IF NOT VALID-BANK
        GO TO 090-ERROR.
    MOVE HC-DATE TO WS-DATE.
    MOVE CORRESPONDING WS-DATE TO H1-DATE.
    MOVE HC-BANK-CODE TO WS-CODE.
    MOVE ZERO TO WS-SUB.
080-READ-NEXT.
    READ HEADER-CARD INTO CASHFLOW-REC
        AT END GO TO 095-GO.
    IF CR-TYPE = ¢*¢
        GO TO 080-READ-NEXT.
    IF CH-TYPE NOT = ¢CH¢
        GO TO 090-ERROR.
    IF CH-COMPANY-NO NOT NUMERIC
        GO TO 090-ERROR.
    IF CH-ACCT1 NOT NUMERIC
        GO TO 090-ERROR.
085-STORE-CASHFLOW.
    ADD 1 TO WS-SUB.
    IF WS-SUB > 10
        DISPLAY ¢CASHFLOW TABLE EXCEEDED¢ UPON CONSOLE
        GO TO 090-ERROR.
    MOVE CASHFLOW-REC TO WA-CH-REC-DATA (WS-SUB).
    MOVE WS-SUB TO WS-CFLOW-LIMIT.
    GO TO 080-READ-NEXT.
090-ERROR.
    DISPLAY ¢INVALID HEADER - PY06¢ UPON CONSOLE.
    STOP ¢CANCEL JOB¢.
    GO TO 090-ERROR.
```

Figure 14 (Part 6 of 10). Input Source Program

```
091-ERROR.
    DISPLAY ¢NO HEADER CARD - PY06¢ UPON CONSOLE.
    STOP ¢CANCEL JOB¢.
    GO TO 091-ERROR.
095-GO.
    PERFORM 600-HEADINGS.
    PERFORM 200-READ-CHEQUE.
    IF CHEQUE-REC NOT = HIGH-VALUES
        PERFORM 260-SEARCH-COMPANY
            VARYING WS-SUB FROM 1 BY 1
            UNTIL CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
            OR WS-SUB NOT < WS-CFLOW-LIMIT
        IF CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
            MOVE ¢C¢ TO WF-CFLOW-BANK-IND.
    MOVE 1 TO WS-SUB.
    IF WF-CFLOW-BANK-IND = ¢B¢
        DISPLAY ¢BANK TAPE TO BE PRODUCED¢ UPON CONSOLE
        OPEN OUTPUT BANK-TAPE
    ELSE
        DISPLAY ¢CASH-FLOW TAPE TO BE PRODUCED¢ UPON CONSOLE
        OPEN OUTPUT CASH-TAPE.
099-OPEN-EXIT.
    EXIT.
*
    EJECT
100-WORK SECTION.
110-WORK.
    IF CHEQUE-REC = HIGH-VALUES
        GO TO 149-WORK-EXIT.
    IF WS-START-RANGE > WS-END-RANGE
        DISPLAY ¢MORE NUMBERS REQUIRED¢ UPON CONSOLE
        MOVE WS-END-RANGE TO WS-PREV-END
        MOVE SPACES TO WS-START-RANGX WS-END-RANGX WS-COMMA
        PERFORM 150-ACCEPT-VALUES
        GO TO 110-WORK.
    PERFORM 250-WRITE.
    PERFORM 500-PRINT.
    PERFORM 200-READ-CHEQUE.
    ADD 1 TO WS-START-RANGE.
    GO TO 110-WORK.
149-WORK-EXIT.
    EXIT.
    EJECT
```

Figure 14 (Part 7 of 10). Input Source Program

```
 150-ACCEPT-VALUES SECTION.
 160-ACCEPT.
     DISPLAY ¢TYPE IN RANGE IN FORMAT NNNNNN,NNNNNN¢
     UPON CONSOLE.
     ACCEPT WS-RANGE FROM CONSOLE.
     IF WS-START-RANGE NOT NUMERIC
         GO TO 170-ERROR.
     IF WS-END-RANGE NOT NUMERIC
         GO TO 170-ERROR.
     IF WS-COMMA NOT = ¢,¢
         GO TO 170-ERROR.
     IF WS-START-RANGE > WS-END-RANGE
         GO TO 170-ERROR.
     IF WS-START-RANGE NOT > WS-PREV-END
         DISPLAY ¢FIRST CHEQUE NUMBER LESS THAN LAST CHEQUE NO¢
         UPON CONSOLE
         DISPLAY ¢RE-ENTER VALUES¢
         UPON CONSOLE
         MOVE SPACES TO WS-RANGX
         GO TO 160-ACCEPT.
     GO TO 199-ACCEPT-EXIT.
 170-ERROR.
     DISPLAY WS-RANGE ¢ NOT ACCEPTABLE FORMAT¢ UPON CONSOLE
     MOVE SPACES TO WS-RANGX.
     GO TO 160-ACCEPT.
 199-ACCEPT-EXIT.
     EXIT.
*
     EJECT
 200-READ-CHEQUE SECTION.
 210-READ.
     READ CHEQUE-FILE INTO CHEQUE-REC
         AT END
             MOVE HIGH-VALUES TO CHEQUE-REC.
 249-READ-EXIT.
     EXIT.
*
*
*
 250-WRITE SECTION.
 251-WRITE.
     MOVE SPACES TO BANK-REC.
     MOVE ¢C¢ TO BT-REC-CODE.
     MOVE WS-CODE TO BT-ACCOUNT.
     MOVE WS-DATE TO BT-DATE-DRAWN.
     MOVE WS-START-RANGE TO BT-CHQE-NO.
     MOVE CF-CHQE-VALUE TO BT-AMOUNT.
     ADD CF-CHQE-VALUE TO WS-VALUE.
     MOVE ¢C¢ TO BT-ORIGIN.
     IF CF-CO-NUMBER NOT = WA-CH-COMPANY-NO (WS-SUB)
         PERFORM 260-SEARCH-COMPANY
             VARYING WS-SUB FROM 1 BY 1
             UNTIL CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
             OR WS-SUB NOT < WS-CFLOW-LIMIT.
```

Figure 14 (Part 8 of 10). Input Source Program

```
           PERFORM 270-WRITE-TAPE.
           ADD 1 TO WS-NUMBER.
       259-WRITE-EXIT.
           EXIT.
      *
       260-SEARCH-COMPANY SECTION.
       261-SEARCH-BEGIN.
       269-SEARCH-EXIT.
           EXIT.
      *
       270-WRITE-TAPE SECTION.
       271-WRITE-BEGIN.
           IF CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
           AND WF-CFLOW-BANK-IND = ¢C¢
               MOVE BANK-REC TO CASH-REC
               MOVE WA-CH-ACCT1 (WS-SUB) TO CT-CFLOW-ACCT1
               MOVE CF-CHQE-VALUE TO CT-CFLOW-VAL1
               WRITE CASH-REC
           ELSE
           IF CF-CO-NUMBER NOT = WA-CH-COMPANY-NO (WS-SUB)
           AND WF-CFLOW-BANK-IND = ¢B¢
               WRITE BANK-REC
           ELSE
               DISPLAY ¢COMBINED CASHFLOW AND BANK DATA¢ UPON CONSOLE
               STOP ¢CANNOT HANDLE MIXTURE - CANCEL JOB¢
               STOP RUN.
       279-WRITE-EXIT.
           EXIT.
      *
           EJECT
       500-PRINT SECTION.
       510-MOVE.
           MOVE CF-CO-NUMBER TO DL-CO-NUMBER.
           MOVE CF-EMPLOYEE-NO TO DL-EMPLOYEE-NO.
           MOVE CF-NAME TO DL-NAME.
           MOVE WS-START-RANGE TO DL-CHQE-NO.
           MOVE CF-CHQE-VALUE TO DL-CHQE-VALUE.
       520-PRINT.
           IF WS-LINE > 56
               PERFORM 600-HEADINGS.
           WRITE PRINT-REC FROM DETAIL-LINE BEFORE ADVANCING 1.
           ADD 1 TO WS-LINE.
           MOVE SPACES TO PRINT-REC DETAIL-LINE.
       549-PRINT-EXIT.
           EXIT.
      *
           EJECT
```

Figure 14 (Part 9 of 10). Input Source Program

```
 600-HEADINGS SECTION.
 610-WRYT.
     MOVE SPACES TO PRINT-REC.
     WRITE PRINT-REC BEFORE ADVANCING CHANNEL-1.
     ADD 1 TO WS-PAGE.
     MOVE WS-PAGE TO H1-PAGE.
     WRITE PRINT-REC FROM HEADING-1 BEFORE ADVANCING 2.
     WRITE PRINT-REC FROM HEADING-2 BEFORE ADVANCING 2.
     MOVE 5 TO WS-LINE.
     MOVE SPACES TO PRINT-REC.
 649-WRYT-EXIT.
     EXIT.
*
 650-INVALID SECTION.
 660-WRONG.
     IF WS-START-RANGE < WS-END-RANGE
         DISPLAY ¢TOO MANY CHEQUE NUMBERS ENTERED¢ UPON CONSOLE
         STOP ¢CANCEL JOB¢.
 699-WRONG-EXIT.
     EXIT.
     EJECT
 700-TOTALS SECTION.
 710-TOTAL.
     MOVE WS-NUMBER TO C2-NUMBER.
     MOVE WS-VALUE TO C3-VALUE.
     MOVE SPACES TO PRINT-REC.
     WRITE PRINT-REC FROM CONTROL-1 BEFORE ADVANCING 2.
     WRITE PRINT-REC FROM CONTROL-2 BEFORE ADVANCING 2.
     WRITE PRINT-REC FROM CONTROL-3 BEFORE ADVANCING 2.
     MOVE SPACES TO PRINT-REC CONTROL-1.
     MOVE SPACES TO CONTROL-2 CONTROL-3.
 749-TOTAL-EXIT.
     EXIT.
*
*
*
 900-CLOSE SECTION.
 910-CLOSE.
     CLOSE CHEQUE-FILE
           HEADER-CARD
***        BANK-TAPE
           CHEQUE-LIST.
     IF WF-CFLOW-BANK-IND = ¢C¢
         CLOSE CASH-TAPE
     ELSE
         CLOSE BANK-TAPE.
 949-CLOSE-EXIT.
     EXIT.
 END-OF-JOB.
 VSAM-OPEN-END SECTION.
 OPEN-END.
     DISPLAY FILE-NAME, ¢ VSAM OPEN ERROR¢
         UPON CONSOLE
     STOP RUN.                                          9
```

Figure 14 (Part 10 of 10). Input Source Program

7.5.2 Highlighted Statements in the Source Program

The following highlighted statements were either statements that were fully converted and/or clauses removed by the conversion routines.

1 The input source program contains author and date written information within the Identification Division of the program. The only required paragraphs for COBOL II and COBOL/VSE are the division header and the PROGRAM-ID. The other Identification Division paragraphs are optional and are treated as documentation on conversion.

2 3 The SELECT statement with the ASSIGN clause in the source program requires the ASSIGN clause to be in the format ASSIGN TO [[SYSnnn-[mfdc-]]S-]name in COBOL II and COBOL/VSE, where SSSnnn- defines the logical unit number in the range of SYS000-SYS254 assigned to the device on which the file resides, mfdc-defines the characteristics of the device, S-defines the organization, and name defines the 1-to-7 character system name for the file. The ASSIGN clauses will be amended during conversion.

4 5 6 7 8 The FD entries within the FILE SECTION for CHEQUE-FILE, HEADER-CARD, BANK-TAPE, and CHEQUE-LIST contain clauses RECORD CONTAINS, and LABEL RECORDS which are not supported under COBOL II and COBOL/VSE for unit record devices. These clauses will be removed during conversion.

9 The source program does not contain an END PROGRAM statement. An END PROGRAM statement may be generated if required for nested programs. Because of options in effect during this conversion, the END PROGRAM statement will be generated.

7.5.3 Output Source from CCCA/VSE Conversion

The following is the result of processing the above program with CCCA/VSE. The output from the conversion produced a member PY06.C in the target VSE library. The member in the source VSE library remained unchanged because of options set during JCL creation. The conversion step received a Return Code of 8 requiring a manual update of the program, and consequently no compilation or linkage edit action was taken. Refer to the diagnostic report for an explanation of the actions taken by the conversion program CCCA/VSE.

```
000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID. PY06.
000300*              PROGRAM CONVERTED BY
000400*              COBOL CONVERSION AID PO 5785-ABJ         10
000500*              CONVERSION DATE 10/30/96 08:28:18.
000600*AUTHOR.
000700*DATE-WRITTEN. 4TH JULY 1977.
000800*DATE-COMPILED.
000900*REMARKS. THIS PROGRAM ACCEPTS A RANGE OF CHEQUE NUMBERS FROM
001000*    THE SPO IN THE FORMAT NNNNNN, NNNNNN. A LIST OF CHEQUES
001100*    DRAWN AND A TAPE-FILE FOR THE BANK RECONCILIATION SYSTEM
001200*    IS PRODUCED.
001300*
001400*
001500*****************************************************************
001600*          P R O G R A M   M O D I F I C A T I O N S            *
001700* PROG.   DATE   PROB NO.   COMMENT                             *
001800*       07/10/83           CHANGE FILENAME ON SELECT STATEMENT TO*
001900*                MATCH FILE NAMES IN NEW VERSION OF UDC PROGRAMS.*
002000*       01/10/84           ALLOW PROGRAM TO DECIDE WHETHER THE   *
002100*                INPUT RECORDS ARE FOR CASHFLOW OR NORMAL BANK.  *
002200*                IF DATA IS CASHFLOW, THEN OUTPUT VIA CASHFLOW   *
002300*                RECORD FORMAT WITH ADDED FIELDS TO 62 BYTES.    *
002400*                ALLOW INPUT OF NEW TYPES OF RECORDS TO IDENTIFY *
002500*                CASHFLOW PAYROLL COMPANIES. THE RECORD TYPES ARE*
002600*                A †CH† TYPE TO ALLOCATE THE GENLAC ACCT CODE FOR*
002700*                A PAYROLL COMPANY, AND A COMMENT RECORD, WITH AN*
002800*                †*† IN COLUMN 1 TO COMMENT THE INPUT DATA DECK. *
002900*****************************************************************
003000 ENVIRONMENT DIVISION.
003100 CONFIGURATION SECTION.
003200 SPECIAL-NAMES.
003300     C01 IS CHANNEL-1
003400     C12 IS CHANNEL-12.
003500 INPUT-OUTPUT SECTION.
003600 FILE-CONTROL.
003700     SELECT CHEQUE-FILE
003800         ASSIGN TO SYS020-DA-3340-D811F
003900         ACCESS MODE IS SEQUENTIAL
004000         ORGANIZATION IS INDEXED
004100         RECORD KEY IS IBM-CHEQUE-KEY
004200         FILE STATUS IS VSAM-STATUS.
004300   SELECT HEADER-CARD ASSIGN TO SYS007-UR-2501-S-SYS007. 11
004400   SELECT BANK-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.
004500   SELECT CASH-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.
004600   SELECT CHEQUE-LIST ASSIGN TO SYS010-UR-3203-S-SYS010. 12
004700*
004800     EJECT
```

Figure 15 (Part 1 of 11). Converted Source Program

```
004900 DATA DIVISION.
005000 FILE SECTION.
005100 FD  CHEQUE-FILE.
005200*                                                          13
005300 01  IBM-CHEQUE-REC.
005400     03  IBM-CHEQUE-KEY          PIC X(11).
005500     03  FILLER                  PIC X(127).
005600*
005700 FD  HEADER-CARD                                           14
005800     RECORDING MODE IS F.
005900*
006000 01  IBM-HEADER-REC              PIC X(80).
006100*
006200 FD  BANK-TAPE
006300     BLOCK CONTAINS 25 RECORDS                            15
006400     RECORDING MODE IS F.
006500*
006600 01  BANK-REC.
006700     03  BT-REC-CODE             PIC X.
006800     03  BT-ACCOUNT              PIC X.
006900     03  BT-DATE-DRAWN           PIC 9(6).
007000     03  BT-CHQE-NO              PIC 9(6).
007100     03  BT-AMOUNT               PIC 9(7)V99.
007200     03  BT-ORIGIN               PIC X.
007300*
007400 FD  CASH-TAPE
007500     BLOCK CONTAINS 10 RECORDS                            16
007600     RECORDING MODE IS F.
007700*
007800 01  CASH-REC.
007900     03  CT-REC-CODE             PIC X.
008000     03  CT-ACCOUNT              PIC X.
008100     03  CT-DATE-DRAWN           PIC 9(6).
008200     03  CT-CHQE-NO              PIC 9(6).
008300     03  CT-AMOUNT               PIC 9(7)V99.
008400     03  CT-ORIGIN               PIC X.
008500     03  CT-CFLOW-ACCT1          PIC X(10).
008600     03  CT-CFLOW-VAL1           PIC S9(7)V99.
008700     03  CT-CFLOW-ACCT2          PIC X(10).
008800     03  CT-CFLOW-VAL2           PIC S9(7)V99.
```

Figure 15 (Part 2 of 11). Converted Source Program

```
008900*
009000 FD  CHEQUE-LIST
009100     RECORDING MODE IS F.                              17
009200*
009300 01  PRINT-REC                    PIC X(133).
009400*
009500     EJECT
009600 WORKING-STORAGE SECTION.
009700 77  VSAM-STATUS                   PIC X(2).
009800     88  OPEN-OK                   VALUE ZERO.
009900 77  FILE-NAME                     PIC X(10)  VALUE SPACES.
010000 77  WS-PREV-END                  PIC 9(6).
010100 77  WS-CODE                      PIC X.
010200 77  WS-SUB                       PIC 9(4).
010300 77  WS-CFLOW-LIMIT               PIC 9(4) VALUE 10.
010400 01  HEADER-REC.
010500     03  HC-TYPE                  PIC XX.
010600     03  HC-DATE.
010700         05  HC-DAY               PIC 99.
010800         05  HC-MONTH             PIC 99.
010900         05  HC-YEAR              PIC 99.
011000     03  HC-BANK-CODE             PIC X.
011100         88   VALID-BANK          VALUE ¢T¢ ¢S¢ ¢W¢ ¢Z¢.
011200     03  FILLER                   PIC X(71).
011300 01  CASHFLOW-REC REDEFINES HEADER-REC.
011400     03  CH-TYPE                  PIC XX.
011500     03  CH-COMPANY-NO            PIC X(4).
011600     03  CH-ACCT1                 PIC X(10).
011700     03  CH-ACCT2                 PIC X(10).
011800     03  FILLER                   PIC X(54).
011900 01  COMMENT-REC REDEFINES HEADER-REC.
012000     03  CR-TYPE                  PIC X.
012100     03  FILLER                   PIC X(79).
012200 01  CHEQUE-REC.
012300     03  CF-CO-NUMBER             PIC X(4).
012400     03  CF-EMPLOYEE-NO           PIC X(6).
012500     03  CF-CODE                  PIC X.
012600     03  CF-NAME                  PIC X(30).
012700     03  FILLER                   PIC X(90).
012800     03  CF-CHQE-VALUE            PIC S9(5)V99.
012900*
013000 01  HEADING-1.
013100     03  FILLER                   PIC X.
013200     03  FILLER                   PIC X(53)      VALUE
013300         ¢  PY06                                  PAYROLL CHEQUE¢.
013400     03  FILLER                   PIC X(38)      VALUE
013500         ¢ REGISTER                         DATE ¢.
```

Figure 15 (Part 3 of 11). Converted Source Program

```
013600     03  H1-DATE.
013700         05  DAY1                PIC 99.
013800         05  FILLER              PIC X          VALUE ¢/¢.
013900         05  MONTH               PIC 99.
014000         05  FILLER              PIC X          VALUE ¢/¢.
014100         05  YEAR                PIC 99.
014200     03  FILLER                  PIC X(28)      VALUE
014300         ¢                       PAGE ¢.
014400     03  H1-PAGE                 PIC ZZ9.
014500     03  FILLER                  PIC XX         VALUE SPACES.
014600*
014700 01  HEADING-2.
014800     03  FILLER                  PIC X.
014900     03  FILLER                  PIC X(44)      VALUE
015000         ¢  COMPANY NUMBER            EMPLOYEE       ¢.
015100     03  FILLER                  PIC X(44)      VALUE
015200         ¢                    NAME                    ¢.
015300     03 FILLER                   PIC X(44)      VALUE
015400         ¢     CHEQUE NUMBER           CHEQUE VALUE  ¢.
015500*
015600 01  DETAIL-LINE.
015700     03  FILLER                  PIC X(8).
015800     03  DL-CO-NUMBER            PIC X(4).
015900     03  FILLER                  PIC X(19).
016000     03  DL-EMPLOYEE-NO          PIC X(6).
016100     03  FILLER                  PIC X(14).
016200     03  DL-NAME                 PIC X(30).
016300     03  FILLER                  PIC X(16).
016400     03  DL-CHQE-NO              PIC 9(6).
016500     03  FILLER                  PIC X(16).
016600     03  DL-CHQE-VALUE           PIC Z(4)9.99.
016700     03  FILLER                  PIC X(4).
016800*
016900 01  CONTROL-1.
017000     03  FILLER                  PIC X.
017100     03  FILLER                  PIC X(20)      VALUE
017200         ¢  CONTROLS          ¢.
017300     03  FILLER                  PIC X(112)     VALUE SPACES.
017400*
017500 01  CONTROL-2.
017600     03  FILLER                  PIC X.
017700     03  FILLER                  PIC X(44)      VALUE
017800         ¢     TOTAL NUMBER OF CHEQUES              ¢.
017900     03  C2-NUMBER               PIC Z(4)9.
018000     03  FILLER                  PIC X(83)      VALUE SPACES.
018100*
```

Figure 15 (Part 4 of 11). Converted Source Program

```
018200 01  CONTROL-3.
018300     03  FILLER                    PIC X.
018400     03  FILLER                    PIC X(39)      VALUE
018500         ¢     TOTAL VALUE OF CHEQUES            ¢.
018600     03  C3-VALUE                  PIC Z(6)9.99.
018700     03  FILLER                    PIC X(83)      VALUE SPACES.
018800*
018900 01  WS-DATE.
019000     03  DAY1                      PIC 99.
019100     03  MONTH                     PIC 99.
019200     03  YEAR                      PIC 99.
019300*
019400 01  WF-FLAGS.
019500     03  WF-CFLOW-BANK-IND         PIC X.
019600*
019700 01  WS-COUNTERS.
019800     03  WS-PAGE                   PIC 999        VALUE ZERO.
019900     03  WS-LINE                   PIC 99         VALUE ZERO.
020000     03  WS-NUMBER                 PIC 9(5).
020100     03  WS-VALUE                  PIC 9(7)V99.
020200 01  WS-RANGE.
020300     03  WS-START-RANGE            PIC 9(6).
020400     03  WS-START-RANGX REDEFINES WS-START-RANGE PIC X(6).
020500     03  WS-COMMA                  PIC X.
020600     03  WS-END-RANGE              PIC 9(6).
020700     03  WS-END-RANGX REDEFINES WS-END-RANGE     PIC X(6).
020800 01  WS-RANGX REDEFINES WS-RANGE   PIC X(13).
020900*
021000 01  WA-CFLOW-REC-ARRAY.
021100     03  WA-CH-REC-DATA OCCURS 10.
021200         05  WA-CH-TYPE            PIC XX.
021300         05  WA-CH-COMPANY-NO      PIC X(4).
021400         05  WA-CH-ACCT1           PIC X(10).
021500         05  WA-CH-ACCT2           PIC X(10).
021600*
021700     EJECT
021800 PROCEDURE DIVISION.
021900 000-CONTROL SECTION.
022000 010-CTL.
022100     PERFORM 050-OPEN.
022200     PERFORM 150-ACCEPT-VALUES.
022300     PERFORM 100-WORK.
022400     PERFORM 650-INVALID.
022500     PERFORM 700-TOTALS.
022600     PERFORM 900-CLOSE.
022700     STOP RUN.
022800 049-CTL-EXIT.
022900     EXIT.
023000*
```

Figure 15 (Part 5 of 11). Converted Source Program

```
023100*
023200*
023300 050-OPEN SECTION.
023400 060-OPEN.
023500     MOVE ¢ZA¢ TO VSAM-STATUS.
023600     OPEN INPUT CHEQUE-FILE
023700     IF  NOT OPEN-OK
023800         MOVE ¢PY1200¢ TO FILE-NAME
023900         GO TO OPEN-END.
024000     OPEN INPUT HEADER-CARD.
024100***  OPEN OUTPUT BANK-TAPE.  DELAY OPENING UNTIL CFLOW/BANK!
024200     OPEN OUTPUT CHEQUE-LIST.
024300 065-SET.
024400     MOVE SPACES TO PRINT-REC DETAIL-LINE.
024500     MOVE SPACES TO WS-RANGE WA-CFLOW-REC-ARRAY.
024600     MOVE ZEROS TO WS-PAGE WS-LINE WS-NUMBER.
024700     MOVE ZEROS TO WS-VALUE WS-PREV-END.
024800     MOVE ¢B¢ TO WF-CFLOW-BANK-IND.
024900 070-READ.
025000     READ HEADER-CARD INTO HEADER-REC
025100         AT END GO TO 091-ERROR.
025200     IF CR-TYPE = ¢*¢
025300         GO TO 070-READ.
025400     IF HC-TYPE NOT = ¢PH¢
025500         GO TO 090-ERROR.
025600     IF NOT VALID-BANK
025700         GO TO 090-ERROR.
025800     MOVE HC-DATE TO WS-DATE.
025900     MOVE CORRESPONDING WS-DATE TO H1-DATE.
026000     MOVE HC-BANK-CODE TO WS-CODE.
026100     MOVE ZERO TO WS-SUB.
026200 080-READ-NEXT.
026300     READ HEADER-CARD INTO CASHFLOW-REC
026400         AT END GO TO 095-GO.
026500     IF CR-TYPE = ¢*¢
026600         GO TO 080-READ-NEXT.
026700     IF CH-TYPE NOT = ¢CH¢
026800         GO TO 090-ERROR.
026900     IF CH-COMPANY-NO NOT NUMERIC
027000         GO TO 090-ERROR.
027100     IF CH-ACCT1 NOT NUMERIC
027200         GO TO 090-ERROR.
027300 085-STORE-CASHFLOW.
027400     ADD 1 TO WS-SUB.
027500     IF WS-SUB > 10
027600         DISPLAY ¢CASHFLOW TABLE EXCEEDED¢ UPON CONSOLE
027700         GO TO 090-ERROR.
027800     MOVE CASHFLOW-REC TO WA-CH-REC-DATA (WS-SUB).
027900     MOVE WS-SUB TO WS-CFLOW-LIMIT.
028000     GO TO 080-READ-NEXT.
```

Figure 15 (Part 6 of 11). Converted Source Program

```
028100 090-ERROR.
028200     DISPLAY ¢INVALID HEADER - PY06¢ UPON CONSOLE.
028300     STOP ¢CANCEL JOB¢.
028400     GO TO 090-ERROR.
028500 091-ERROR.
028600     DISPLAY ¢NO HEADER CARD - PY06¢ UPON CONSOLE.
028700     STOP ¢CANCEL JOB¢.
028800     GO TO 091-ERROR.
028900 095-GO.
029000     PERFORM 600-HEADINGS.
029100     PERFORM 200-READ-CHEQUE.
029200     IF CHEQUE-REC NOT = HIGH-VALUES
029300         PERFORM 260-SEARCH-COMPANY
029400             VARYING WS-SUB FROM 1 BY 1
029500             UNTIL CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
029600             OR WS-SUB NOT < WS-CFLOW-LIMIT
029700         IF CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
029800             MOVE ¢C¢ TO WF-CFLOW-BANK-IND.
029900     MOVE 1 TO WS-SUB.
030000     IF WF-CFLOW-BANK-IND = ¢B¢
030100         DISPLAY ¢BANK TAPE TO BE PRODUCED¢ UPON CONSOLE
030200         OPEN OUTPUT BANK-TAPE
030300     ELSE
030400         DISPLAY ¢CASH-FLOW TAPE TO BE PRODUCED¢ UPON CONSOLE
030500         OPEN OUTPUT CASH-TAPE.
030600 099-OPEN-EXIT.
030700     EXIT.
030800*
030900     EJECT
031000 100-WORK SECTION.
031100 110-WORK.
031200     IF CHEQUE-REC = HIGH-VALUES
031300         GO TO 149-WORK-EXIT.
031400     IF WS-START-RANGE > WS-END-RANGE
031500         DISPLAY ¢MORE NUMBERS REQUIRED¢ UPON CONSOLE
031600         MOVE WS-END-RANGE TO WS-PREV-END
031700         MOVE SPACES TO WS-START-RANGX WS-END-RANGX WS-COMMA
031800         PERFORM 150-ACCEPT-VALUES
031900         GO TO 110-WORK.
032000     PERFORM 250-WRITE.
032100     PERFORM 500-PRINT.
032200     PERFORM 200-READ-CHEQUE.
032300     ADD 1 TO WS-START-RANGE.
032400     GO TO 110-WORK.
032500 149-WORK-EXIT.
032600     EXIT.
032700     EJECT
```

Figure 15 (Part 7 of 11). Converted Source Program

```
032800 150-ACCEPT-VALUES SECTION.
032900 160-ACCEPT.
033000     DISPLAY ¢TYPE IN RANGE IN FORMAT NNNNNN,NNNNNN¢
033100     UPON CONSOLE.
033200     ACCEPT WS-RANGE FROM CONSOLE.
033300     IF WS-START-RANGE NOT NUMERIC
033400         GO TO 170-ERROR.
033500     IF WS-END-RANGE NOT NUMERIC
033600         GO TO 170-ERROR.
033700     IF WS-COMMA NOT = ¢,¢
033800         GO TO 170-ERROR.
033900     IF WS-START-RANGE > WS-END-RANGE
034000         GO TO 170-ERROR.
034100     IF WS-START-RANGE NOT > WS-PREV-END
034200         DISPLAY ¢FIRST CHEQUE NUMBER LESS THAN LAST CHEQUE NO¢
034300         UPON CONSOLE
034400         DISPLAY ¢RE-ENTER VALUES¢
034500         UPON CONSOLE
034600         MOVE SPACES TO WS-RANGX
034700         GO TO 160-ACCEPT.
034800     GO TO 199-ACCEPT-EXIT.
034900 170-ERROR.
035000     DISPLAY WS-RANGE ¢ NOT ACCEPTABLE FORMAT¢ UPON CONSOLE
035100     MOVE SPACES TO WS-RANGX.
035200     GO TO 160-ACCEPT.
035300 199-ACCEPT-EXIT.
035400     EXIT.
035500*
035600     EJECT
035700 200-READ-CHEQUE SECTION.
035800 210-READ.
035900     READ CHEQUE-FILE INTO CHEQUE-REC
036000         AT END
036100             MOVE HIGH-VALUES TO CHEQUE-REC.
036200 249-READ-EXIT.
036300     EXIT.
036400*
036500*
036600*
036700 250-WRITE SECTION.
036800 251-WRITE.
036900     MOVE SPACES TO BANK-REC.
037000     MOVE ¢C¢ TO BT-REC-CODE.
037100     MOVE WS-CODE TO BT-ACCOUNT.
037200     MOVE WS-DATE TO BT-DATE-DRAWN.
037300     MOVE WS-START-RANGE TO BT-CHQE-NO.
037400     MOVE CF-CHQE-VALUE TO BT-AMOUNT.
037500     ADD CF-CHQE-VALUE TO WS-VALUE.
```

Figure 15 (Part 8 of 11). Converted Source Program

```
037600     MOVE ¢C¢ TO BT-ORIGIN.
037700     IF CF-CO-NUMBER NOT = WA-CH-COMPANY-NO (WS-SUB)
037800         PERFORM 260-SEARCH-COMPANY
037900             VARYING WS-SUB FROM 1 BY 1
038000             UNTIL CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
038100             OR WS-SUB NOT < WS-CFLOW-LIMIT.
038200     PERFORM 270-WRITE-TAPE.
038300     ADD 1 TO WS-NUMBER.
038400 259-WRITE-EXIT.
038500     EXIT.
038600*
038700 260-SEARCH-COMPANY SECTION.
038800 261-SEARCH-BEGIN.
038900 269-SEARCH-EXIT.
039000     EXIT.
039100*
039200 270-WRITE-TAPE SECTION.
039300 271-WRITE-BEGIN.
039400     IF CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)
039500     AND WF-CFLOW-BANK-IND = ¢C¢
039600         MOVE BANK-REC TO CASH-REC
039700         MOVE WA-CH-ACCT1 (WS-SUB) TO CT-CFLOW-ACCT1
039800         MOVE CF-CHQE-VALUE TO CT-CFLOW-VAL1
039900         WRITE CASH-REC
040000     ELSE
040100     IF CF-CO-NUMBER NOT = WA-CH-COMPANY-NO (WS-SUB)
040200     AND WF-CFLOW-BANK-IND = ¢B¢
040300         WRITE BANK-REC
040400     ELSE
040500         DISPLAY ¢COMBINED CASHFLOW AND BANK DATA¢ UPON CONSOLE
040600         STOP ¢CANNOT HANDLE MIXTURE - CANCEL JOB¢
040700         STOP RUN.
040800 279-WRITE-EXIT.
040900     EXIT.
041000*
041100     EJECT
041200 500-PRINT SECTION.
041300 510-MOVE.
041400     MOVE CF-CO-NUMBER TO DL-CO-NUMBER.
041500     MOVE CF-EMPLOYEE-NO TO DL-EMPLOYEE-NO.
041600     MOVE CF-NAME TO DL-NAME.
041700     MOVE WS-START-RANGE TO DL-CHQE-NO.
041800     MOVE CF-CHQE-VALUE TO DL-CHQE-VALUE.
041900 520-PRINT.
042000     IF WS-LINE > 56
042100         PERFORM 600-HEADINGS.
042200     WRITE PRINT-REC FROM DETAIL-LINE BEFORE ADVANCING 1.
042300     ADD 1 TO WS-LINE.
042400     MOVE SPACES TO PRINT-REC DETAIL-LINE.
042500 549-PRINT-EXIT.
042600     EXIT.
042700*
```

Figure 15 (Part 9 of 11). Converted Source Program

```
042800     EJECT
042900 600-HEADINGS SECTION.
043000 610-WRYT.
043100     MOVE SPACES TO PRINT-REC.
043200     WRITE PRINT-REC BEFORE ADVANCING CHANNEL-1.
043300     ADD 1 TO WS-PAGE.
043400     MOVE WS-PAGE TO H1-PAGE.
043500     WRITE PRINT-REC FROM HEADING-1 BEFORE ADVANCING 2.
043600     WRITE PRINT-REC FROM HEADING-2 BEFORE ADVANCING 2.
043700     MOVE 5 TO WS-LINE.
043800     MOVE SPACES TO PRINT-REC.
043900 649-WRYT-EXIT.
044000     EXIT.
044100*
044200 650-INVALID SECTION.
044300 660-WRONG.
044400     IF WS-START-RANGE < WS-END-RANGE
044500         DISPLAY ¢TOO MANY CHEQUE NUMBERS ENTERED¢ UPON CONSOLE
044600         STOP ¢CANCEL JOB¢.
044700 699-WRONG-EXIT.
044800     EXIT.
044900     EJECT
045000 700-TOTALS SECTION.
045100 710-TOTAL.
045200     MOVE WS-NUMBER TO C2-NUMBER.
045300     MOVE WS-VALUE TO C3-VALUE.
045400     MOVE SPACES TO PRINT-REC.
045500     WRITE PRINT-REC FROM CONTROL-1 BEFORE ADVANCING 2.
045600     WRITE PRINT-REC FROM CONTROL-2 BEFORE ADVANCING 2.
045700     WRITE PRINT-REC FROM CONTROL-3 BEFORE ADVANCING 2.
045800     MOVE SPACES TO PRINT-REC CONTROL-1.
045900     MOVE SPACES TO CONTROL-2 CONTROL-3.
046000 749-TOTAL-EXIT.
046100     EXIT.
046200*
046300*
046400*
046500 900-CLOSE SECTION.
046600 910-CLOSE.
046700     CLOSE CHEQUE-FILE
046800           HEADER-CARD
046900***        BANK-TAPE
047000           CHEQUE-LIST.
047100     IF WF-CFLOW-BANK-IND = ¢C¢
047200         CLOSE CASH-TAPE
047300     ELSE
047400         CLOSE BANK-TAPE.
```

Figure 15 (Part 10 of 11). Converted Source Program

```
047500 949-CLOSE-EXIT.
047600     EXIT.
047700 END-OF-JOB.
047800 VSAM-OPEN-END SECTION.
047900 OPEN-END.
048000     DISPLAY FILE-NAME, ¢ VSAM OPEN ERROR¢
048100         UPON CONSOLE
048200     STOP RUN.
048300 END PROGRAM PY06.                                    18
```

Figure 15 (Part 11 of 11). Converted Source Program

7.5.4 Highlighted Results of Conversion

The following are explanations of the conversion actions taken by CCCA/VSE.

10 During conversion the eye-catcher is inserted after the PROGRAM-ID paragraph. The eye-catcher shows that the program was converted using CCCA/VSE and shows the date and time of conversion.

11 12 The SELECT statement on the source program has been amended to comply with COBOL II and COBOL/VSE language standards for the ASSIGN clause. The new format of the SELECT statement is as shown.

13 14 15 16 17 The FD entries within the FILE SECTION have been amended to comply with the new parameters for the FD under COBOL II and COBOL/VSE rules. Extraneous parameters have been removed.

18 The options in effect during conversion requested that an END PROGRAM statement be generated. The END PROGRAM statement has been inserted into the program.

7.5.5 CCCA/VSE Diagnostic Report

Following are the job steps generated by CCCA/VSE and the diagnostic report produced. The SETPARM parameters shown are generated from the on-line panels of CCCA/VSE, or from parameters set during installation tailoring.

```
// JOB PY06                                        DATE 04/11/96
// SETPARM JOBNAME=PY06
// SETPARM ABJVSSH=¢CCCA¢          19
// SETPARM ABJVSPR=¢CCCA¢          20
// SETPARM DEST=¢(*,SYSA)¢         21
// SETPARM LST=A                   22
// SETPARM PSTCMP=Y                23
// SETPARM CTLFILE=ABJCTL          24
// SETPARM DEBUG=0                 25
// SETPARM CICS=N                  26
// SETPARM SQL=N                   27
// SETPARM LOPTION=¢*¢             28
// SETPARM INSRC=¢USERLIB.PROD¢    29
// SETPARM MEM=PY06                30
// SETPARM INCPY1=¢USERLIB.PROD¢   31
// SETPARM INCPY2=
// SETPARM INCPY3=
// SETPARM INCPY4=
// SETPARM INCPY5=
// SETPARM INCPY6=
// SETPARM OUTSRC=¢USERLIB.TEST¢   32
// SETPARM OUTCPY=¢USERLIB.TEST¢   33
// SETPARM PGMNUM=0009
// SETPARM LANGLVL=1               34
// SETPARM JOBCLAS=4               35
  ON $ABEND GOTO EEOP
  ON $CANCEL GOTO EEOP
  ON $RC > 15 CONTINUE
// SETPARM ABJLBSH=¢PRD2.PROD¢     36
// SETPARM USERCAT=¢VSESP.USER.CATALOG¢ 37
// SETPARM CBLL=¢PRD2.DBASE¢              38
// SETPARM DCBLL=¢PRD2.PROD¢              39
// SETPARM SQLL=¢PRD2.SQL340¢             40
// SETPARM SQLP=¢USERID=SQLDBA/SQLDBAPw,PREPNAME=DUMMY,COB2¢
// GOTO ABJCTL
/. ABJCTL
// SETPARM VOL1=DOSRES                    41
// SETPARM VOL2=SYSWK1                    42
// SETPARM STA1=11115                     43
// SETPARM STA2=10245                     44
// SETPARM LEN1=50                        45
// SETPARM LEN2=50                        46
// GOTO EOPPRM
/. EOPPRM
// SETPARM ONE=ONE
// SETPARM TWO=TWO
// SETPARM EOP=N
// LIBDEF PHASE,SEARCH=(PRD2.PROD,PRD2.DBASE,PRD2.SQL340,SDL)
// IF INCPY1 =¢¢ THEN
// LIBDEF SOURCE,SEARCH=(USERLIB.PROD)
*
```

Figure 16 (Part 1 of 13). CCCA/VSE Diagnostic Report

```
// EXEC ABJBWAIT,PARM=¢ENQCCCA      ¢
// DLBL IJSYSUC,¢VSESP.USER.CATALOG¢,0,VSAM
// IF CICS  = Y THEN
// GOTO PASSTEP
/. PASSTEP
// DLBL IJSYSUC,¢VSESP.USER.CATALOG¢,0,VSAM
// DLBL CONTROL,¢CCCA.CONTROL.ABJ¢,0,VSAM
// DLBL DRIVEN,¢CCCA.DRIVEN.ABJ¢,0,VSAM
// DLBL TABLE,¢CCCA.TABLE.ABJ¢,0,VSAM
// DLBL MESSAG,¢CCCA.MESSAGE.ABJ¢,0,VSAM
// DLBL CHANGE,¢CCCA.CHANGE.ABJ¢,0,VSAM
// DLBL WORK,¢CCCA.DRWORK.ABJ¢,0,VSAM
// DLBL BLL1DD,¢CCCA.BLLOUT¢,0,VSAM,RECSIZE=36,CAT=IJSYSUC,RECORDS=(2640,5280),DISP=(,KEEP)
// DLBL BLL2DD,¢CCCA.CELLOUT¢,0,VSAM,RECSIZE=63,CAT=IJSYSUC,RECORDS=(2490,5580),DISP=(,KEEP)
*
// DLBL TOKEN,¢CCCA.TOKEN.ABJ¢,0,VSAM,DISP=(,KEEP)
// DLBL TMPSRC,¢CCCA.TMPSRC¢,0,VSAM,RECSIZE=96,CAT=IJSYSUC,RECORDS=(1170,2340),DISP=(,KEEP)
// DLBL CBLCARD,¢CCCA.LITDLM¢,0,VSAM,RECSIZE=80,CAT=IJSYSUC,RECORDS=(1170,2340),DISP=(,KEEP)
// DLBL POINTER,¢CCCA.POINTER¢,0,VSAM,RECSIZE=20,CAT=IJSYSUC,RECORDS=(1395,2790),DISP=(,KEEP)
// DLBL SYSPRT1,¢CCCA.SYSPRT1¢,0,VSAM,RECSIZE=133,CAT=IJSYSUC,RECORDS=(1110,2220),DISP=(,KEEP)
// DLBL SYSPRT2,¢CCCA.SYSPRT2¢,0,VSAM,RECSIZE=133,CAT=IJSYSUC,RECORDS=(1110,3330),DISP=(,KEEP)
// DLBL SYSPRT3,¢CCCA.SYSPRT3¢,0,VSAM,RECSIZE=133,CAT=IJSYSUC,RECORDS=(1110,2220),DISP=(,KEEP)
// IF PSTCMP=N THEN
1S46I  ONE STATEMENT SKIPPED DUE TO IF CONDITION
// DLBL PROGFL,¢CCCA.P0009.PGM¢,0,VSAM,RECSIZE=80,CAT=IJSYSUC,RECORDS=1,DISP=(NEW,KEEP)
// IF PSTCMP=Y THEN
// GOTO CCNT
/. CCNT
*
  ON $ABEND GOTO PRPASS0
  ON $CANCEL GOTO PRPASS0
// SETPARM PRC=999
// EXEC PGM=ABJPASS0,SIZE=ABJPASS0,PARM=¢PY06,0,1,N,N,VSE,USERLIB.PROD,USERLIB.TEST,USERLIB.TEST)

1S55I  LAST RETURN CODE WAS 0008
// SETPARM PRC=$RC
/. PRPASS0
// IF DEBUG=0 THEN
// GOTO PRT3
/. PRT3
// DLBL IJSYSUC,¢VSESP.USER.CATALOG¢,0,VSAM
// DLBL SYSPRT,¢CCCA.SYSPRT3¢,0,VSAM,CAT=IJSYSUC,DISP=(OLD,DELETE)
// EXEC ABJPRPR
```

Figure 16 (Part 2 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

000100 IDENTIFICATION DIVISION.
000200 PROGRAM-ID. PY06.
000300*              PROGRAM CONVERTED BY
000400*              COBOL CONVERSION AID PO 5785-ABJ
000500*              CONVERSION DATE 11/04/96 16:20:36.
*OLD** AUTHOR.
000600*AUTHOR.                        47
*OLD** DATE-WRITTEN. 4TH JULY 1977.
000700*DATE-WRITTEN. 4TH JULY 1977.
*OLD** DATE-COMPILED.
000800*DATE-COMPILED.
*OLD** REMARKS. THIS PROGRAM ACCEPTS A RANGE OF CHEQUE NUMBERS FROM
000900*REMARKS. THIS PROGRAM ACCEPTS A RANGE OF CHEQUE NUMBERS FROM
001000*    THE SPO IN THE FORMAT NNNNNN, NNNNNN. A LIST OF CHEQUES
001100*    DRAWN AND A TAPE-FILE FOR THE BANK RECONCILIATION SYSTEM
001200*    IS PRODUCED.
001300*
001400*
001500*****************************************************************AO07SYSA
001600*          P R O G R A M   M O D I F I C A T I O N S             *AO07SYSA
001700* PROG.   DATE   PROB NO.   COMMENT                              *AO07SYSA
001800*       07/10/83           CHANGE FILENAME ON SELECT STATEMENT TO*RO07SYSA
001900*                MATCH FILE NAMES IN NEW VERSION OF UDC PROGRAMS.*RO07SYSA
002000*       01/10/84           ALLOW PROGRAM TO DECIDE WHETHER THE   *RO04SYSA
002100*                INPUT RECORDS ARE FOR CASHFLOW OR NORMAL BANK.  *RO04SYSA
002200*                IF DATA IS CASHFLOW, THEN OUTPUT VIA CASHFLOW   *RO04SYSA
002300*                RECORD FORMAT WITH ADDED FIELDS TO 62 BYTES.    *RO04SYSA
002400*                ALLOW INPUT OF NEW TYPES OF RECORDS TO IDENTIFY *RO01SYSA
002500*                CASHFLOW PAYROLL COMPANIES. THE RECORD TYPES ARE*RO01SYSA
002600*                A †CH† TYPE TO ALLOCATE THE GENLAC ACCT CODE FOR*RO04SYSA
002700*                A PAYROLL COMPANY, AND A COMMENT RECORD, WITH AN*RO01SYSA
002800*                †*† IN COLUMN 1 TO COMMENT THE INPUT DATA DECK. *RO01SYSA
002900*****************************************************************AO07SYSA
003000 ENVIRONMENT DIVISION.
003100 CONFIGURATION SECTION.
003200 SPECIAL-NAMES.
003300     C01 IS CHANNEL-1
003400     C12 IS CHANNEL-12.
003500 INPUT-OUTPUT SECTION.
003600 FILE-CONTROL.
003700     SELECT CHEQUE-FILE
003800         ASSIGN TO SYS020-DA-3340-D811F                          RO03SYSA
003900         ACCESS MODE IS SEQUENTIAL
004000         ORGANIZATION IS INDEXED
004100         RECORD KEY IS IBM-CHEQUE-KEY
004200         FILE STATUS IS VSAM-STATUS.      48


*OLD**     SELECT HEADER-CARD ASSIGN TO SYS007-UR-2501-S.
004300     SELECT HEADER-CARD ASSIGN TO SYS007-UR-2501-S-SYS007.
*OLD**     SELECT BANK-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.
004400     SELECT BANK-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.
*OLD**     SELECT CASH-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.         RO04SYSA
```

Figure 16 (Part 3 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

004500     SELECT CASH-TAPE ASSIGN TO SYS008-UT-3420-S-BR0200.          RO04SYSA
*OLD**     SELECT CHEQUE-LIST ASSIGN TO SYS010-UR-3203-S.
004600     SELECT CHEQUE-LIST ASSIGN TO SYS010-UR-3203-S-SYS010.
004700*
004800     EJECT
004900 DATA DIVISION.
005000 FILE SECTION.
*OLD** FD  CHEQUE-FILE
*OLD**     LABEL RECORDS ARE STANDARD.
005100 FD  CHEQUE-FILE.
005200*
005300 01  IBM-CHEQUE-REC.
005400     03  IBM-CHEQUE-KEY          PIC X(11).                      R604ANL1
005500     03  FILLER                  PIC X(127).                     R604ANL1
005600*
005700 FD  HEADER-CARD
*OLD**     RECORD CONTAINS 80 CHARACTERS
*OLD**     LABEL RECORDS ARE OMITTED
005800     RECORDING MODE IS F.
005900*
006000 01  IBM-HEADER-REC              PIC X(80).
006100*
006200 FD  BANK-TAPE
*OLD**     RECORD CONTAINS 24 CHARACTERS
006300     BLOCK CONTAINS 25 RECORDS
*OLD**     LABEL RECORDS ARE STANDARD
006400     RECORDING MODE IS F.
006500*
006600 01  BANK-REC.
006700     03  BT-REC-CODE             PIC X.
006800     03  BT-ACCOUNT              PIC X.
006900     03  BT-DATE-DRAWN           PIC 9(6).
007000     03  BT-CHQE-NO              PIC 9(6).
007100     03  BT-AMOUNT               PIC 9(7)V99.
007200     03  BT-ORIGIN               PIC X.
007300*                                                                 A004SYSA
007400 FD  CASH-TAPE                                                    A004SYSA
*OLD**     RECORD CONTAINS 62 CHARACTERS                               A004SYSA
007500     BLOCK CONTAINS 10 RECORDS                                   A004SYSA
*OLD**     LABEL RECORDS ARE STANDARD                                  A004SYSA
007600     RECORDING MODE IS F.                                        A004SYSA
007700*                                                                A004SYSA
007800 01  CASH-REC.                                                   A004SYSA
007900     03  CT-REC-CODE             PIC X.                          A004SYSA
008000     03  CT-ACCOUNT              PIC X.                          A004SYSA
008100     03  CT-DATE-DRAWN           PIC 9(6).                       A004SYSA
008200     03  CT-CHQE-NO              PIC 9(6).                       A004SYSA
008300     03  CT-AMOUNT               PIC 9(7)V99.                    A004SYSA
008400     03  CT-ORIGIN               PIC X.                          A004SYSA
008500     03  CT-CFLOW-ACCT1          PIC X(10).                      A004SYSA
008600     03  CT-CFLOW-VAL1           PIC S9(7)V99.                   A004SYSA
008700     03  CT-CFLOW-ACCT2          PIC X(10).                      A004SYSA
008800     03  CT-CFLOW-VAL2           PIC S9(7)V99.                   A004SYSA
```

Figure 16 (Part 4 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

008900*                                                                  A004SYSA
009000 FD  CHEQUE-LIST
*OLD**     RECORD CONTAINS 133 CHARACTERS
*OLD**     LABEL RECORDS ARE OMITTED
009100     RECORDING MODE IS F.
009200*
009300 01  PRINT-REC                   PIC X(133).
009400*
009500     EJECT
009600 WORKING-STORAGE SECTION.
009700 77  VSAM-STATUS                  PIC X(2).
009800     88  OPEN-OK                  VALUE ZERO.
009900 77  FILE-NAME                    PIC X(10)  VALUE SPACES.
010000 77  WS-PREV-END                 PIC 9(6).
010100 77  WS-CODE                     PIC X.
010200 77  WS-SUB                      PIC 9(4).                   R001SYSA
010300 77  WS-CFLOW-LIMIT              PIC 9(4) VALUE 10.          R004SYSA
010400 01  HEADER-REC.
010500     03  HC-TYPE                 PIC XX.
010600     03  HC-DATE.
010700         05  HC-DAY              PIC 99.
010800         05  HC-MONTH            PIC 99.
010900         05  HC-YEAR             PIC 99.
011000     03  HC-BANK-CODE            PIC X.
011100         88   VALID-BANK         VALUE ¢T¢ ¢S¢ ¢W¢ ¢Z¢.      R525SYSA
011200     03  FILLER                  PIC X(71).
011300 01  CASHFLOW-REC REDEFINES HEADER-REC.                      R001SYSA
011400     03  CH-TYPE                 PIC XX.                     R004SYSA
011500     03  CH-COMPANY-NO           PIC X(4).                   R004SYSA
011600     03  CH-ACCT1                PIC X(10).                  R004SYSA
011700     03  CH-ACCT2                PIC X(10).                  R004SYSA
011800     03  FILLER                  PIC X(54).                  R001SYSA
011900 01  COMMENT-REC REDEFINES HEADER-REC.                       R001SYSA
012000     03  CR-TYPE                 PIC X.                      R004SYSA
012100     03  FILLER                  PIC X(79).                  R001SYSA
012200 01  CHEQUE-REC.
012300     03  CF-CO-NUMBER            PIC X(4).                   R604SYSA
012400     03  CF-EMPLOYEE-NO          PIC X(6).                   R604SYSA
012500     03  CF-CODE                 PIC X.                      R604SYSA
012600     03  CF-NAME                 PIC X(30).
012700     03  FILLER                  PIC X(90).
012800     03  CF-CHQE-VALUE           PIC S9(5)V99.               R604SYSA
012900*                                                             R604SYSA
013000 01  HEADING-1.
013100     03  FILLER                  PIC X.
013200     03  FILLER                  PIC X(53)      VALUE
013300         ¢  PY06                                PAYROLL CHEQUE¢. R604SYSA
013400     03  FILLER                  PIC X(38)      VALUE
013500         ¢ REGISTER                      DATE ¢.
013600     03  H1-DATE.                                            R604SYSA
013700         05  DAY1                PIC 99.                     R604SYSA
013800         05  FILLER              PIC X          VALUE ¢/¢.   R604SYSA
013900         05  MONTH               PIC 99.                     R604SYSA
```

Figure 16 (Part 5 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

014000         05  FILLER              PIC X          VALUE ¢/¢.       R604SYSA
014100         05  YEAR                PIC 99.                         R604SYSA
014200     03  FILLER                   PIC X(28)      VALUE
014300         ¢                        PAGE ¢.
014400     03  H1-PAGE                  PIC ZZ9.
014500     03  FILLER                   PIC XX         VALUE SPACES.
014600*
014700 01  HEADING-2.
014800     03  FILLER                   PIC X.
014900     03  FILLER                   PIC X(44)      VALUE
015000         ¢  COMPANY NUMBER            EMPLOYEE       ¢.
015100     03  FILLER                   PIC X(44)      VALUE
015200         ¢                    NAME                    ¢.
015300     03 FILLER                        PIC X(44)      VALUE
015400         ¢     CHEQUE NUMBER           CHEQUE VALUE  ¢.
015500*
015600 01  DETAIL-LINE.
015700     03  FILLER                   PIC X(8).
015800     03  DL-CO-NUMBER             PIC X(4).                       R604SYSA
015900     03  FILLER                   PIC X(19).
016000     03  DL-EMPLOYEE-NO           PIC X(6).                       R604SYSA
016100     03  FILLER                   PIC X(14).
016200     03  DL-NAME                  PIC X(30).
016300     03  FILLER                   PIC X(16).
016400     03  DL-CHQE-NO               PIC 9(6).
016500     03  FILLER                   PIC X(16).
016600     03  DL-CHQE-VALUE            PIC Z(4)9.99.
016700     03  FILLER                   PIC X(4).
016800*
016900 01  CONTROL-1.
017000     03  FILLER                   PIC X.
017100     03  FILLER                   PIC X(20)      VALUE
017200         ¢  CONTROLS          ¢.
017300     03  FILLER                   PIC X(112)     VALUE SPACES.
017400*
017500 01  CONTROL-2.
017600     03  FILLER                   PIC X.
017700     03  FILLER                   PIC X(44)      VALUE
017800         ¢     TOTAL NUMBER OF CHEQUES              ¢.
017900     03  C2-NUMBER                PIC Z(4)9.
018000     03  FILLER                   PIC X(83)      VALUE SPACES.
018100*
018200 01  CONTROL-3.
018300     03  FILLER                   PIC X.
018400     03  FILLER                   PIC X(39)      VALUE
018500         ¢     TOTAL VALUE OF CHEQUES           ¢.
018600     03  C3-VALUE                 PIC Z(6)9.99.
018700     03  FILLER                   PIC X(83)      VALUE SPACES.
018800*
018900 01  WS-DATE.
019000     03  DAY1                     PIC 99.
019100     03  MONTH                    PIC 99.
019200     03  YEAR                     PIC 99.
```

Figure 16 (Part 6 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

019300*
019400 01  WF-FLAGS.                                                      RO04SYSA
019500     03  WF-CFLOW-BANK-IND       PIC X.                             RO04SYSA
019600*                                                                    A004SYSA
019700 01  WS-COUNTERS.
019800     03  WS-PAGE                 PIC 999        VALUE ZERO.
019900     03  WS-LINE                 PIC 99         VALUE ZERO.
020000     03  WS-NUMBER               PIC 9(5).
020100     03  WS-VALUE                PIC 9(7)V99.
020200 01  WS-RANGE.
020300     03  WS-START-RANGE          PIC 9(6).
020400     03  WS-START-RANGX REDEFINES WS-START-RANGE PIC X(6).
020500     03  WS-COMMA                PIC X.
020600     03  WS-END-RANGE            PIC 9(6).
020700     03  WS-END-RANGX REDEFINES WS-END-RANGE     PIC X(6).
020800 01  WS-RANGX REDEFINES WS-RANGE  PIC X(13).
020900*                                                                    A001SYSA
021000 01  WA-CFLOW-REC-ARRAY.                                            R001SYSA
021100     03  WA-CH-REC-DATA OCCURS 10.                                  RO04SYSA
021200         05  WA-CH-TYPE          PIC XX.                            RO04SYSA
021300         05  WA-CH-COMPANY-NO    PIC X(4).                          RO04SYSA
021400         05  WA-CH-ACCT1         PIC X(10).                         RO04SYSA
021500         05  WA-CH-ACCT2         PIC X(10).                         RO04SYSA
021600*
021700     EJECT
021800 PROCEDURE DIVISION.
021900 000-CONTROL SECTION.
022000 010-CTL.
022100     PERFORM 050-OPEN.
022200     PERFORM 150-ACCEPT-VALUES.
022300     PERFORM 100-WORK.
022400     PERFORM 650-INVALID.
022500     PERFORM 700-TOTALS.
022600     PERFORM 900-CLOSE.
022700     STOP RUN.
022800 049-CTL-EXIT.
022900     EXIT.
023000*
023100*
023200*
023300 050-OPEN SECTION.
023400 060-OPEN.
023500     MOVE ¢ZA¢ TO VSAM-STATUS.                                      R604ANL1
023600     OPEN INPUT CHEQUE-FILE
023700     IF  NOT OPEN-OK
023800         MOVE ¢PY1200¢ TO FILE-NAME                                 R604SYSA
023900         GO TO OPEN-END.
024000     OPEN INPUT HEADER-CARD.
024100***  OPEN OUTPUT BANK-TAPE.  DELAY OPENING UNTIL CFLOW/BANK!        RO04SYSA
024200     OPEN OUTPUT CHEQUE-LIST.
024300 065-SET.                                                          R001SYSA
024400     MOVE SPACES TO PRINT-REC DETAIL-LINE.
024500     MOVE SPACES TO WS-RANGE WA-CFLOW-REC-ARRAY.                    RO04SYSA
```

Figure 16 (Part 7 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

024600     MOVE ZEROS TO WS-PAGE WS-LINE WS-NUMBER.
024700     MOVE ZEROS TO WS-VALUE WS-PREV-END.
024800     MOVE ¢B¢ TO WF-CFLOW-BANK-IND.                                 RO04SYSA
024900 070-READ.                                                          RO01SYSA
025000     READ HEADER-CARD INTO HEADER-REC
025100         AT END GO TO 091-ERROR.                                    RO01SYSA
025200     IF CR-TYPE = ¢*¢                                               RO01SYSA
025300         GO TO 070-READ.                                            RO01SYSA
025400     IF HC-TYPE NOT = ¢PH¢
025500         GO TO 090-ERROR.
025600     IF NOT VALID-BANK
025700         GO TO 090-ERROR.
025800     MOVE HC-DATE TO WS-DATE.
025900     MOVE CORRESPONDING WS-DATE TO H1-DATE.
026000     MOVE HC-BANK-CODE TO WS-CODE.
026100     MOVE ZERO TO WS-SUB.                                           RO01SYSA
026200 080-READ-NEXT.                                                     RO01SYSA
026300     READ HEADER-CARD INTO CASHFLOW-REC                             RO04SYSA
026400         AT END GO TO 095-GO.                                       RO01SYSA
026500     IF CR-TYPE = ¢*¢                                               RO01SYSA
026600         GO TO 080-READ-NEXT.                                       RO01SYSA
026700     IF CH-TYPE NOT = ¢CH¢                                          RO04SYSA
026800         GO TO 090-ERROR.                                           RO01SYSA
026900     IF CH-COMPANY-NO NOT NUMERIC                                   RO04SYSA
027000         GO TO 090-ERROR.                                           A001SYSA
027100     IF CH-ACCT1 NOT NUMERIC                                        RO04SYSA
027200         GO TO 090-ERROR.                                           A001SYSA
027300 085-STORE-CASHFLOW.                                                RO01SYSA
027400     ADD 1 TO WS-SUB.                                               RO01SYSA
027500     IF WS-SUB > 10                                                 RO04SYSA
027600         DISPLAY ¢CASHFLOW TABLE EXCEEDED¢ UPON CONSOLE             RO04SYSA
027700         GO TO 090-ERROR.                                           RO01SYSA
027800     MOVE CASHFLOW-REC TO WA-CH-REC-DATA (WS-SUB).                  RO04SYSA
027900     MOVE WS-SUB TO WS-CFLOW-LIMIT.                                 RO04SYSA
028000     GO TO 080-READ-NEXT.                                           RO01SYSA
028100 090-ERROR.
028200     DISPLAY ¢INVALID HEADER - PY06¢ UPON CONSOLE.                  R604SYSA
028300     STOP ¢CANCEL JOB¢.                                             R604SYSA
028400     GO TO 090-ERROR.
028500 091-ERROR.                                                         A001SYSA
028600     DISPLAY ¢NO HEADER CARD - PY06¢ UPON CONSOLE.                  RO01SYSA
028700     STOP ¢CANCEL JOB¢.                                             A001SYSA
028800     GO TO 091-ERROR.                                               RO01SYSA
028900 095-GO.
029000     PERFORM 600-HEADINGS.
029100     PERFORM 200-READ-CHEQUE.
029200     IF CHEQUE-REC NOT = HIGH-VALUES                                RO04SYSA
029300         PERFORM 260-SEARCH-COMPANY                                 RO04SYSA
029400             VARYING WS-SUB FROM 1 BY 1                             RO04SYSA
029500             UNTIL CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)         RO04SYSA
029600             OR WS-SUB NOT < WS-CFLOW-LIMIT                         RO04SYSA
029700         IF CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)                RO04SYSA
029800             MOVE ¢C¢ TO WF-CFLOW-BANK-IND.                         RO04SYSA
```

Figure 16 (Part 8 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

029900     MOVE 1 TO WS-SUB.                                              R004SYSA
030000     IF WF-CFLOW-BANK-IND = ¢B¢                                     R004SYSA
030100         DISPLAY ¢BANK TAPE TO BE PRODUCED¢ UPON CONSOLE            R004SYSA
030200         OPEN OUTPUT BANK-TAPE                                      R004SYSA
030300     ELSE                                                           R004SYSA
030400         DISPLAY ¢CASH-FLOW TAPE TO BE PRODUCED¢ UPON CONSOLE       R004SYSA
030500         OPEN OUTPUT CASH-TAPE.                                     R004SYSA
030600 099-OPEN-EXIT.
030700     EXIT.
030800*
030900     EJECT
031000 100-WORK SECTION.
031100 110-WORK.
031200     IF CHEQUE-REC = HIGH-VALUES                                    R604SYSA
031300         GO TO 149-WORK-EXIT.
031400     IF WS-START-RANGE > WS-END-RANGE
031500         DISPLAY ¢MORE NUMBERS REQUIRED¢ UPON CONSOLE
031600         MOVE WS-END-RANGE TO WS-PREV-END
031700         MOVE SPACES TO WS-START-RANGX WS-END-RANGX WS-COMMA
031800         PERFORM 150-ACCEPT-VALUES
031900         GO TO 110-WORK.
032000     PERFORM 250-WRITE.
032100     PERFORM 500-PRINT.
032200     PERFORM 200-READ-CHEQUE.
032300     ADD 1 TO WS-START-RANGE.
032400     GO TO 110-WORK.
032500 149-WORK-EXIT.
032600     EXIT.
032700     EJECT
032800 150-ACCEPT-VALUES SECTION.
032900 160-ACCEPT.
033000     DISPLAY ¢TYPE IN RANGE IN FORMAT NNNNNN,NNNNNN¢                R604SYSA
033100     UPON CONSOLE.
033200     ACCEPT WS-RANGE FROM CONSOLE.
033300     IF WS-START-RANGE NOT NUMERIC
033400         GO TO 170-ERROR.
033500     IF WS-END-RANGE NOT NUMERIC
033600         GO TO 170-ERROR.
033700     IF WS-COMMA NOT = ¢,¢                                          R604SYSA
033800         GO TO 170-ERROR.
033900     IF WS-START-RANGE > WS-END-RANGE
034000         GO TO 170-ERROR.
034100     IF WS-START-RANGE NOT > WS-PREV-END
034200         DISPLAY ¢FIRST CHEQUE NUMBER LESS THAN LAST CHEQUE NO¢
034300         UPON CONSOLE
034400         DISPLAY ¢RE-ENTER VALUES$¢
034500         UPON CONSOLE
034600         MOVE SPACES TO WS-RANGX
034700         GO TO 160-ACCEPT.
034800     GO TO 199-ACCEPT-EXIT.
034900 170-ERROR.
035000     DISPLAY WS-RANGE ¢ NOT ACCEPTABLE FORMAT¢ UPON CONSOLE
035100     MOVE SPACES TO WS-RANGX.
```

Figure 16 (Part 9 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

035200     GO TO 160-ACCEPT.
035300 199-ACCEPT-EXIT.
035400     EXIT.
035500*
035600     EJECT
035700 200-READ-CHEQUE SECTION.
035800 210-READ.
035900     READ CHEQUE-FILE INTO CHEQUE-REC
036000         AT END
036100             MOVE HIGH-VALUES TO CHEQUE-REC.
036200 249-READ-EXIT.
036300     EXIT.
036400*
036500*
036600*
036700 250-WRITE SECTION.
036800 251-WRITE.                                                    RO01SYSA
036900     MOVE SPACES TO BANK-REC.                                  AO04SYSA
037000     MOVE ¢C¢ TO BT-REC-CODE.
037100     MOVE WS-CODE TO BT-ACCOUNT.
037200     MOVE WS-DATE TO BT-DATE-DRAWN.
037300     MOVE WS-START-RANGE TO BT-CHQE-NO.
037400     MOVE CF-CHQE-VALUE TO BT-AMOUNT.
037500     ADD CF-CHQE-VALUE TO WS-VALUE.
037600     MOVE ¢C¢ TO BT-ORIGIN.
037700     IF CF-CO-NUMBER NOT = WA-CH-COMPANY-NO (WS-SUB)           RO04SYSA
037800         PERFORM 260-SEARCH-COMPANY                            RO04SYSA
037900             VARYING WS-SUB FROM 1 BY 1                        RO04SYSA
038000             UNTIL CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)    RO04SYSA
038100             OR WS-SUB NOT < WS-CFLOW-LIMIT.                   RO04SYSA
038200     PERFORM 270-WRITE-TAPE.                                   RO04SYSA
038300     ADD 1 TO WS-NUMBER.
038400 259-WRITE-EXIT.                                               RO01SYSA
038500     EXIT.
038600*
038700 260-SEARCH-COMPANY SECTION.                                   RO01SYSA
038800 261-SEARCH-BEGIN.                                             RO04SYSA
038900 269-SEARCH-EXIT.                                              RO01SYSA
039000     EXIT.                                                     AO01SYSA
039100*                                                              AO01SYSA
039200 270-WRITE-TAPE SECTION.                                       RO04SYSA
039300 271-WRITE-BEGIN.                                              RO04SYSA
039400     IF CF-CO-NUMBER = WA-CH-COMPANY-NO (WS-SUB)               AO04SYSA
039500     AND WF-CFLOW-BANK-IND = ¢C¢                               RO04SYSA
039600         MOVE BANK-REC TO CASH-REC                             RO04SYSA
039700         MOVE WA-CH-ACCT1 (WS-SUB) TO CT-CFLOW-ACCT1           RO04SYSA
039800         MOVE CF-CHQE-VALUE TO CT-CFLOW-VAL1                   RO04SYSA
039900         WRITE CASH-REC                                        RO04SYSA
040000     ELSE                                                      RO04SYSA
040100     IF CF-CO-NUMBER NOT = WA-CH-COMPANY-NO (WS-SUB)           RO04SYSA
040200     AND WF-CFLOW-BANK-IND = ¢B¢                               RO04SYSA
040300         WRITE BANK-REC                                        RO04SYSA
040400     ELSE                                                      AO04SYSA
```

Figure 16 (Part 10 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

040500         DISPLAY ¢COMBINED CASHFLOW AND BANK DATA¢ UPON CONSOLE   R004SYSA
040600         STOP ¢CANNOT HANDLE MIXTURE - CANCEL JOB¢                R004SYSA
040700         STOP RUN.                                                R004SYSA
040800 279-WRITE-EXIT.                                                  R004SYSA
040900     EXIT.                                                        A004SYSA
041000*                                                                 A004SYSA
041100     EJECT
041200 500-PRINT SECTION.
041300 510-MOVE.
041400     MOVE CF-CO-NUMBER TO DL-CO-NUMBER.
041500     MOVE CF-EMPLOYEE-NO TO DL-EMPLOYEE-NO.
041600     MOVE CF-NAME TO DL-NAME.
041700     MOVE WS-START-RANGE TO DL-CHQE-NO.
041800     MOVE CF-CHQE-VALUE TO DL-CHQE-VALUE.
041900 520-PRINT.
042000     IF WS-LINE > 56
042100         PERFORM 600-HEADINGS.
042200     WRITE PRINT-REC FROM DETAIL-LINE BEFORE ADVANCING 1.
042300     ADD 1 TO WS-LINE.
042400     MOVE SPACES TO PRINT-REC DETAIL-LINE.
042500 549-PRINT-EXIT.
042600     EXIT.
042700*
042800     EJECT
042900 600-HEADINGS SECTION.
043000 610-WRYT.
043100     MOVE SPACES TO PRINT-REC.
043200     WRITE PRINT-REC BEFORE ADVANCING CHANNEL-1.
043300     ADD 1 TO WS-PAGE.
043400     MOVE WS-PAGE TO H1-PAGE.
043500     WRITE PRINT-REC FROM HEADING-1 BEFORE ADVANCING 2.
043600     WRITE PRINT-REC FROM HEADING-2 BEFORE ADVANCING 2.
043700     MOVE 5 TO WS-LINE.
043800     MOVE SPACES TO PRINT-REC.
043900 649-WRYT-EXIT.
044000     EXIT.
044100*
044200 650-INVALID SECTION.
044300 660-WRONG.
044400     IF WS-START-RANGE < WS-END-RANGE
044500         DISPLAY ¢TOO MANY CHEQUE NUMBERS ENTERED¢ UPON CONSOLE
044600         STOP ¢CANCEL JOB¢.                                       R604SYSA
044700 699-WRONG-EXIT.
044800     EXIT.
044900     EJECT
045000 700-TOTALS SECTION.
045100 710-TOTAL.
045200     MOVE WS-NUMBER TO C2-NUMBER.
045300     MOVE WS-VALUE TO C3-VALUE.
045400     MOVE SPACES TO PRINT-REC.
045500     WRITE PRINT-REC FROM CONTROL-1 BEFORE ADVANCING 2.
045600     WRITE PRINT-REC FROM CONTROL-2 BEFORE ADVANCING 2.
045700     WRITE PRINT-REC FROM CONTROL-3 BEFORE ADVANCING 2.
```

Figure 16 (Part 11 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
SEQNBR-A 1 B.. ... 2 ... ...  COBOL SOURCE STATEMENTS  ... 6 ... ... 7 .IDENTFCN OLD/SQ S

045800     MOVE SPACES TO PRINT-REC CONTROL-1.
045900     MOVE SPACES TO CONTROL-2 CONTROL-3.
046000 749-TOTAL-EXIT.
046100     EXIT.
046200*
046300*
046400*
046500 900-CLOSE SECTION.
046600 910-CLOSE.
046700     CLOSE CHEQUE-FILE
046800           HEADER-CARD
046900***        BANK-TAPE                                          RO04SYSA
047000           CHEQUE-LIST.
047100     IF WF-CFLOW-BANK-IND = ¢C¢                               RO04SYSA
047200         CLOSE CASH-TAPE                                      RO04SYSA
047300     ELSE                                                     RO04SYSA
047400         CLOSE BANK-TAPE.                                     RO04SYSA
047500 949-CLOSE-EXIT.
047600     EXIT.
047700 END-OF-JOB.
047800 VSAM-OPEN-END SECTION.
047900 OPEN-END.
048000     DISPLAY FILE-NAME, ¢ VSAM OPEN ERROR¢
048100         UPON CONSOLE
048200     STOP RUN.
048300 END PROGRAM PY06.
```

Figure 16 (Part 12 of 13). CCCA/VSE Diagnostic Report

```
 5785-CCC R1.0   - IBM    COBOL CONVERSION AID -       SAMPLE RUN                    PY06
CONVERSION FROM DOS/VS COBOL      TO  VS COBOL II
OPTIONS IN EFFECT :                  49
  PROCEDURE NAME CHECKING ........ YES   LANGLEVEL ..................... DOS/VS COBOL
  FLAG REPORT WRITER STMTS ....... YES   CICS ........................... NO CICS ST.
  REMOVE OBSOLETE ELEMENTS ....... YES   LINE COUNT ..................... 60
  GENERATE CALL ILBOABN0 STMT..... YES   DATE FORMAT ................... DD/MM/YY
  GENERATE END PROGRAM STMT ...... YES   RESEQUENCING .................. YES
  POST-CONVERSION COMPILE ........ YES   INCREMENT ..................... 0100
  MANUAL CHANGE FLAGGING ......... NO    RESERVED WORD SUFFIX ........... 74
  HANDLE EXEC SQL INCLUDE AS COPY. NO    GENERATE NEW PROGRAM ........... YES
  REMOVE NON 88 VALUE CLAUSE IN FS NO    GENERATE NEW COPY .............. YES
  FLAG IF FILE-STATUS (NOT) = †00† NO    REPLACE LIKE-NAMED COPY MBR .... YES
  FLAG 31-BIT ADDRESS ARITHMETIC.. NO    PRINT REFERENCE SOURCE LINE .... YES
  INCL.W-S IN CICS COMPILE OF L-S. YES   PRINT COPY MODULE .............. YES
  OPTION-13 ...................... NO    LEVEL DIAGNOSTIC ............... 00
  OPTION-14 ...................... NO    DEBUG MODE ..................... 0
  OPTION-15 ...................... NO    SQL ............................ N
HIGHEST SEVERITY MESSAGE FOR THIS CONVERSION:   08      50
       0018 MESSAGES ISSUED
       0018 MESSAGES PRINTED
SEQNBR  CPYNBR    MSGID   RC  MESSAGE TEXT
000600           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000700           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000800           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
000900           ABJ6011  00  REMARKS CHANGED TO COMMENT
004200           ABJ6252  08  DIFFERENT FILE STATUS VALUES WILL NOW BE RETURNED. *MANUAL U 51
004300           ABJ6098  00  ASSIGNMENT NAME IS CHANGED
004400           ABJ6098  00  ASSIGNMENT NAME IS CHANGED
004500           ABJ6098  00  ASSIGNMENT NAME IS CHANGED
004600           ABJ6098  00  ASSIGNMENT NAME IS CHANGED
005100           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
005800           ABJ6177  00  RECORD CLAUSE IS REMOVED
005800           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
006300           ABJ6177  00  RECORD CLAUSE IS REMOVED
006400           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
007500           ABJ6177  00  RECORD CLAUSE IS REMOVED
007600           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
009100           ABJ6177  00  RECORD CLAUSE IS REMOVED
009100           ABJ6181  00  OBSOLETE ELEMENT IS REMOVED
/. PRTCNT
// IF PRC GT 8 THEN
1S46I  ONE STATEMENT SKIPPED DUE TO IF CONDITION
// IF PRC GT 4 THEN
// GOTO EOP      TO AVOID POST-CONVERSION-COMPILE    52
/. EOP
// EXEC ABJBWAIT,PARM=¢DEQCCCA    ¢
// IF CICS = Y THEN
// GOTO ENDD
/. ENDD
EOJ PY06      MAX.RETURN CODE=0008                                    DATE 04/11/96
```

Figure 16 (Part 13 of 13). CCCA/VSE Diagnostic Report

7.5.6 Highlighted Messages in Diagnostic Report

The following are explanations of the parameters used during the conversion jobsteps, and where applicable, the conversion actions taken by CCCA/VSE.

19 The SETPARM ABJVSSH='CCCA' is obtained from the CCCA/VSE on-line panel *Environment Options*. This parameter refers to the high level qualifier of the shared VSAM files which are used by all CCCA/VSE users.

20 The SETPARM ABJVSPR='CCCA' is obtained from the CCCA/VSE on-line panel *Environment Options*. This parameter refers to the high level qualifier of the Private VSAM files which are used each by CCCA/VSE user, if required. In most cases it would be the same as shown in 19

21 The SETPARM DEST='(*,SYSA) is obtained from the CCCA/VSE on-line panel *Environment Options*. This parameter refers to the list output destination field on the on-line panel.

22 The SETPARM LST=A is obtained from the CCCA/VSE on-line panel *Environment Options*. This parameter refers to the list output class field as shown on the panel.

23 The SETPARM PSTCMP=Y is obtained from the CCCA/VSE on-line panel *Conversion Options 2*. If the parameter Compile after converting has been set to Y, the program will be compiled providing that the program conversion received a return code of less than 08.

24 The SETPARM CTLFILE=ABJCTL is obtained from the CCCA/VSE on-line panel *Control*. This parameter contains the name of the CCCA/VSE control file that has been defined in the CICS File Control Table. In this case, the name of the control file is ABJCTL.

25 The SETPARM DEBUG=0 is obtained from the CCCA/VSE on-line panel *Conversion Option Panel 1*. This parameter represents the field Conversion debug output, and in this case, conversion debug is set to none.

26 The SETPARM CICS=N is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the on-line field CICS, and in this case, CICS has been set to N.

27 The SETPARM SQL=N is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field SQL shown on the panel, and in this case, SQL has been set to N.

28 The SETPARM LOPTION='*' is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field Language level, and in this case, it will default to the source language level as set in the *Language Level* panel.

29 The SETPARM INSRC='USERLIB.PROD' is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field Program Source: Library and Sublibrary shown on the panel.

30 The SETPARM MEM=PY06 is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field Member name shown on the Conversion panel.

31 The SETPARM INCPY1='USERLIB.PROD' is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the fields Copy sublibraries entered on the panel.

32 The SETPARM OUTSRC='USERLIB.TEST' is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field Output Source: Program sublibrary.

33 The SETPARM OUTCPY='USERLIB.TEST' is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field Output Source: Copy sublibrary.

34 The SETPARM LANGLVL=1 is obtained from the CCCA/VSE on-line panel *Language Level*. This parameter represents the field Source language level shown on the panel, and in this case, it is set to 1 which represents a DOS/VS COBOL program.

35 The SETPARM JOBCLASS=4 is obtained from the CCCA/VSE on-line panel *Conversion Panel*. This parameter represents the field Job Class and refers to the VSE/POWER job class.

36 The SETPARM ABJLBSH='PRD2.PROD' is obtained from the CCCA/VSE library member ABJPPARM book cataloged into a VSE source library during tailoring of installation parameters, and represents the library/sublibrary where CCCA/VSE has been installed.

37 The SETPARM USERCAT='VSESP.USER.CATALOG' is obtained from the CCCA/VSE library member ABJPPARM book cataloged into a VSE source library during tailoring of installation parameters, and represents the name of the VSAM catalog where the shared and private files have been defined.

38 The SETPARM CBLL='PRD2.DBASE' is obtained from the CCCA/VSE library member ABJPPARM book cataloged into a VSE source library during tailoring of installation parameters, and represents the library/sublibrary where COBOL II or COBOL/VSE has been installed.

Please note that if your are converting to COBOL/VSE, the Language Environment (LE/VSE) run-time libraries must be in the permanent LIBDEF chain. For example:

// LIBDEF *, SEARCH=(PRD2.SCEEBASE,.....)

39 The SETPARM DCBLL='PRD2.PROD' is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the library/sublibrary where DOS/VS COBOL has been installed.

40 The SETPARM SQLL='PRD2.SQL340' is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the library/sublibrary where SQL/DS has been installed.

41 The SETPARM VOL1=DOSRES is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the volume identifier for the first of the required internal SAM files.

42 The SETPARM VOL2=SYSWK1 is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the volume identifier for the second of the required internal SAM files.

43 The SETPARM STA1=11115 is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the start track for the first of the required internal SAM files.

44 The SETPARM STA2=10245 is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the start track for the second of the required internal SAM files.

45 The SETPARM LEN1=50 is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the number of tracks for the first of the required internal SAM files.

46 The SETPARM LEN2=50 is obtained from the CCCA/VSE library member ABJPCNV book cataloged into a VSE source library during tailoring of installation parameters, and represents the number of tracks for the second of the required internal SAM files.

47 This shows the details of converting from old DOS/VS COBOL acceptable code to COBOL II or COBOL/VSE acceptable code. The changes that have been made have been described in the section Highlighted Results of Conversion. The same principle applies to other statements that have been modified during the conversion process.

48 51 This statement was flagged for manual update during the conversion process. DOS/VS COBOL only allowed for one data-name for validation and testing of VSAM Return Codes. COBOL II and COBOL/VSE have provided two data-name clauses for validation and testing. The application programmer is responsible for manually applying the new features within the program, and then compiling and linking. Because this statement was flagged for manual update, the conversion did not perform a post-conversion compilation.

49 This section of the report shows the conversion options that were in effect during processing. These options were described earlier, and were obtained from the CCCA/VSE panels completed prior to job submission.

50 52 This section of the report shows the highest severity for this conversion. In this case the highest level was 8, which caused the conditional JCL to skip the Post-Conversion-Compile. The reason for the severity 8 message was described in 48

Appendix A. Product Information for VSE/ESA 2.1.x and 1.4.x

In this appendix you can find the base and optional products included in VSE/ESA package 1.4 (5750-ACD) and VSE/ESA package 2.1 (5690-VSE). For each different level a table is given. Each table has five columns with the following headers:

- **Pid Number**

  In this column is the program identification number used to order the product.

- **Program Name**

  In this column are the program name and a brief description of the different components.

- **CLC**

  In this column is the component level code that uniquely defines the different components within a product.

- **Affected**

  A 'Yes' in this column means that the component is affected by the Year2000. 'No' means that the component is not affected by the Year2000.

- **PTF/APAR Number**

  In this column is the PTF or APAR number if the component is enabled for the Year2000 with a PTF. The APAR number is specified if the PTF was not yet available when the list was created. '?' means that the APAR number was not yet known when the list was created. When there is a new product or release available that provides Year2000 support, it is listed here. 'Obsolete' means that the product is no longer supported and that it has not been replaced by another product. 'No change' in this column means that there are no changes planned to the product even if it is affected by Year2000.

There are nine different tables:

1. Table 1 lists the VSE/ESA 2.1.3. base products.
2. Table 2 lists the optional products available in VSE/ESA 2.1.3.
3. Table 3 lists all optional products available in VSE/ESA 2.1.2 that are different from those listed in table 2.
4. Table 4 lists all optional products available in VSE/ESA 2.1.1 and 2.1.0 that are different from those listed in table 2.
5. Table 5 lists the VSE/ESA 1.4.2 base products.
6. Table 6 lists the optional products available in VSE/ESA 1.4.2.
7. Table 7 lists all optional products available in VSE/ESA 1.4.1 that are different from those listed in table 6.
8. Table 8 lists all optional products available in VSE/ESA 1.4.0 that are different from those listed in table 6.
9. Table 9 lists some products that are not part of the VSE/ESA packages.

Note: The product lists are from November 1996. The information available was current at that time. For updated information consult Preventive Service Planning.

An UPGRADE ID YR2000VSE exists in Preventive Service Planning (PSP). In this PSP bucket you can find Year2000 related information for VSE/ESA and VSE/ESA optional products. Refer to this bucket to obtain the latest information. If you do not have direct access to this PSP information you can call the IBM Support Center (ISC) to obtain a copy.

A.1 Product Information for VSE/ESA 2.1.3

A.1.1 VSE/ESA 2.1.3 Base Products

Table 12 (Page 1 of 2). Product Information for VSE/ESA 2.1.3 Base Products

Pid Number	Program Name	CLC	Affected	PTF/APAR number
5648-099	DITTO/ESA for VSE-1.1.0	160	Yes	UN84733 UN97461
5656-260	VSE/EREP-3.5.0	E00	Yes	No Change
5686-026	CICS/VSE Production-2.3.0	14X	Yes	UN93197
	CICS/VSE Generation-2.3.0	14V	No	
	CICS/VSE RCF-2.3.0	14W	Yes	UN96558
5686-065	ACF/VTAM-4.2.0	FE6	No	
5686-066	VSE/SP Unique code-6.1.0	15C	Yes	UN95668
	VSE/SP MRI English-6.1.0	15D	Yes	UN95669
	VSE/SP MRI Kanji-6.1.0	15E	Yes	UN95670
	VSE/SP MRI German-6.1.0	15F	Yes	UN95671
	VSE/SP MRI Spanish-6.1.0	15H	Yes	UN95672
	VSE/POWER-6.1.0	15C	Yes	UD50121
	VSE/POWER Macros-6.1.0	15G	Yes	UD50122
	VSE/AF OC/MC-6.1.0	15C	No	
	VSE/VSAM-6.1.0	15C	Yes	UD50100
	VSE/VSAM Macros-6.1.0	15G	No	

Table 12 (Page 2 of 2). Product Information for VSE/ESA 2.1.3 Base Products				
Pid Number	**Program Name**	**CLC**	**Affected**	**PTF/APAR number**
5686-066 (cont)	VSE/AF SVR BAM GDS-6.1.0	15C	Yes	UD49992 UD50018 UD50112 UD50132
	VSE/AF Macros-6.1.0	15G	Yes	UD49994 UD50019 UD50115 UD50134
	VSE/AF Gen. Feature-6.1.0	15J	Yes	UD49995 UD50020 UD50118 UD50138
	VSE/AF MSHP-6.1.0	15C	Yes	UD50120
	VSE/AF INFO/ANA-6.1.0	15C	Yes	UD49997
	VSE/AF IOCP-6.1.0	15C	No	
	VSE/AF ICCF-6.1.0	15C	Yes	UN95595
	VSE/AF Fast Copy-6.1.0	15C	Yes	UD49998 UD50139
	REXX/VSE Kernel+Interface-6.1.0	15I	No	UN95319 Funct.Enh.
	REXX/VSE Library-6.1.0	15I	No	
	OLTEP/VSE-6.1.0	15I	Yes	UD50130
	DWF/VSE-6.1.0	18R	No	
	LANRES/VSE-6.1.0	185	No	
5686-066 (cont)	VSE Workdesk Base-6.1.0	15I	Yes	UN95692
	VSE Workdesk German-6.1.0	1CJ	Yes	UN95693
	VSE Workdesk Spanish-6.1.0	1CK	Yes	UN95694
	VSE Workdesk Kanji-6.1.0	1CL	Yes	UN95695
	VSE Workdesk English-6.1.0	1CM	Yes	UN95696
5696-234	High Level Assembler-1.2.0	189	No	
5746-RC5	BTAM ES-1.1.0	I08	No	
5747-DS2	ICKDSF-1.16.0	1G0	Yes	UN88673

A.1.2 VSE/ESA 2.1.3 Optional Products

Table 13 (Page 1 of 3). Product Information for VSE/ESA 2.1.3 Optional Products				
Pid Number	**Program Name**	**CLC**	**Affected**	**PTF/APAR Number**
5648-061	QMF/VSE Base-3.2.0	17L	Yes	UN90787
	QMF/VSE Nat lang (CLCs 17M-17X)-3.2.0	17N	No	
5648-063	ACF/NCP-7.4.0	44A	No	
	ACF/NCP F-7.4.0	4EA	No	
5648-078	Vis.Gen. Host Serv. Basic + NLS-1.1.0	15K	No	
5648-109	VisualLift VSE-1.1.2	16T	No	
5648-113	AFP/Font Collection-1.1.0 (CLCs 140-15F)	140	Yes	?
5668-719	X.25 NPSI (3725)-2.1.0	B60	No	
5668-723	GDDM IVU-1.1.2	2F4	No	
5668-738	ACF/NCP-5.4.0	CH0	No	
	ACF/NCP UT 2.0-5.0 (CLCs CH1-CH5)-5.4.0	CH1	No	

Table 13 (Page 2 of 3). Product Information for VSE/ESA 2.1.3 Optional Products				
Pid Number	**Program Name**	**CLC**	**Affected**	**PTF/APAR Number**
5668-801	GDDM IMD-2.1.2	CV5	No	
5668-812	GDDM PGF-2.1.2	7H8	No	
5668-813	CSP/AD/PWS-3.3.0	A10	Yes	VisualGen
	CSP/AD-3.3.0	D66	Yes	VisualGen
5668-814	CSP/AE-3.3.0	D67	Yes	VisualGen
5668-854	ACF/NCP/VS 3725-4.3.1	D34	No	
5668-958	VS COBOL II Comp+Lib+Debug-1.4.0	40A	Yes	COBOL/VSE LE/VSE
5686-011	CICSVR-1.1.0	A10	Yes	?
5686-013	NetView FTP-1.1.1	CX8	Yes	?
5686-018	CICS-DDM/VSE-1.1.0	A10	Yes	?
5686-022	DW/370-2.1.0	A10	No	
	DW/370-Nat. Lang. (CLCs A11-A43)-2.1.0	A11	No	
	DW/DI-Nat. Lang. (CLCs A24-A44)-2.1.0	A24	No	
5686-040	PSF/VSE-2.2.1	DC0	Yes	UD50146
	CODEPAGE.B240-2.2.1	FR9	No	
	CODEPAGE.B300-2.2.1	FS0	No	
	COMPATS.U240-2.2.1	DH0	No	
	COMPATS.U300-2.2.1	DH1	No	
	ACIF/VSE-2.2.1	FW0	No	
	AFPAPI/VSE-2.2.1	FW1	No	
5686-041	VSE/DSNX-2.1.2	E62	Yes	UN85348
5686-055	NetView Base VSE-2.3.0	3II	Yes	UD49976 UD50208
	NetView NLS (CLCs 3AA-3BB)-VSE-2.3.0	3AA	No	
5686-057	GDDM/VSE Base-3.1.1	13V	Yes	UN88288 UN79821
	GDDM/VSE Nat. Lang. (13V-148)-3.1.1	13W	No	
5686-064	ACF/SSP-4.4.0	44F	No	
5686-067	Lang.Envir. Base + Nat.Lang.-1.1.0	18A	No	
5686-068	IBM COBOL 1.1.0 for VSE/ESA	18M	Yes	UN95059
	IBM COBOL for NLS Mixed Case-1.1.0	18N	Yes	UN95060
	IBM COBOL for NLS Japanese-1.1.0	18O	Yes	UN95061
5686-069	IBM PL/I for VSE/ESA-1.1.0	18P	Yes	UN97019 UN96009
5686-072	ALERT/VSE-4.9.0	S00		Vendor product
5686-073	ADSM/VSE-1.2.0	14Z	No	
5686-079	ALERT/CICS-4.9.0	S04		Vendor product
5688-022	VS COBOL II Lib.-1.4.0	40D	Yes	COBOL/VSE LE/VSE
5688-023	VS COBOL II Comp + Lib -1.4.0	40A	Yes	COBOL/VSE LE/VSE
5688-035	X.25 NPSI (3745)-3.4.0	CJ0	No	
5688-103	SQL/DS-3.5.0	F50	No	
5688-187	C/370 Compiler-2.1.0	2A2	Yes	C/VSE LE/VSE
5688-188	C/370 Runtime -2.1.0	2A3	Yes	C/VSE LE/VSE

Table 13 (Page 3 of 3). Product Information for VSE/ESA 2.1.3 Optional Products				
Pid Number	**Program Name**	**CLC**	**Affected**	**PTF/APAR Number**
5688-190	PPFA/VSE-1.1.0	A03	No	
5688-191	OGL/370 English-1.1.0	AA0	Yes	UN93115
	OGL/370 German-1.1.0	AA1	Yes	UN93116
	OGL/370 Japanese-1.1.0	AA2	Yes	UN93117
	OGL/370-1.1.0	A05	Yes	UN93034
5735-XXB	EP for ACF/NCP 3725-1.6.1	F12	No	
	EP 3745 Without NCP-1.8.0	I59	No	
	EP for ACF/NCP-1.9.0	CH6	No	
	EP for ACF/NCP-1.12.0	4A1	No	
5746-CB1	COBOL Compiler-1.3.1	E46	Yes	COBOL/VSE LE/VSE
5746-LM4	COBOL Library-1.3.1	E47	Yes	COBOL/VSE LE/VSE
5746-RG1	DOS/VS RPG II-1.3.0	O42	YES	UN95321
5746-SM3	DFSORT/VSE-3.2.0	32A	YES	PN81797
5746-XC5	OCCF-1.5.0	16V	No	
5746-XE7	VSE/ACLR-1.2.1	H06	Yes	?
5746-XXT	SDF/CICS-1.5.0	B27	No	
5746-XX1	VSE/DL/I-1.10.0	DB5	Yes	UN94805

A.2 Product Information for VSE/ESA 2.1.2

Table 14. Product Information for VSE/ESA 2.1.2				
Pid Number	**Program Name**	**CLC**	**Affected**	**PTF/APAR Number**
5648-061	QMF/VSE Base-3.1.1	190	Yes	UN90788
	QMF/VSE Nat. Lang. (CLCs 191-19F)-3.1.1	191	No	
5648-063	ACF/NCP-7.1.0	3A0	No	
	ACF/NCP F-7.1.0	3E0	No	
5666-270	DISOSS-3.4.0	I60	Yes	PN89310
5666-318	PS/CICS + NLS-1.3.1	D39	Yes	No change
5686-064	ACF/SSP-4.1.0	F41	No	
5688-103	SQL/DS-3.4.0	F40	Yes	SQL/DS 3.5

A.3 Product Information for VSE/ESA 2.1.0 and 2.1.1

Table 15. Product Information for VSE/ESA 2.1.0 and 2.1.1

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5648-040	Vis Gen Developer Basic-1.1.0	15Q	No	
5648-061	QMF/VSE Base-3.1.1	190	Yes	UN90788
	QMF/VSE Nat. Lang. (CLCs 191-19F)-3.1.1	191	No	
5648-063	ACF/NCP-7.1.0	3A0	No	
	ACF/NCP F-7.1.0	3E0	No	
5648-076	Vis Gen Work Group Basic + NLS-1.1.0	15V	No	
5648-086	Vis Gen Appl Gener incl Gen Opt-1.1.0	15N	No	
	Vis Gen Appl Gener Opt VSE-1.1.0	165	No	
5666-270	DISOSS-3.4.0	I60	Yes	PN89310
5666-318	PS/CICS + NLS-1.3.1	D39	Yes	No change
5666-329	CICSPARS VSE-1.1.1	D38	Obsolete	
5686-064	ACF/SSP-4.1.0	F41	No	
5686-075	ASF/VSE Base-3.1.0	166	No	
5688-103	SQL/DS-3.4.0	F40	No	
5736-LM4	PL/I Resident Library-1.6.0	N72	Yes	PL/I VSE LE/VSE
5736-LM5	PL/I Transient Library-1.6.0	N73	Yes	PL/I VSE LE/VSE
5736-PL1	PL/I Opt Compiler-1.6.0	N74	Yes	PL/I VSE LE/VSE
5736-PL3	PL/I Comp+Libs-1.6.0	N74	Yes	PL/I VSE LE/VSE
5746-SM3	DFSORT/VSE-3.1.0	31A	Yes	DFSORT/VSE 3.2

A.4 Product Information for VSE/ESA 1.4.2

A.4.1 VSE/ESA 1.4.2 Base Products

Table 16. Product Information for VSE/ESA 1.4.2 Base Products

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5656-092	VSE/OLTEP-1.1.0	923	Yes	UD50181
5656-260	VSE/EREP-3.5.0	E00	Yes	No Change
5666-363	ACF/VTAM-3.4.0	J90	No	
5686-026	CICS/VSE Production-2.3.0	14X	Yes	UN93197
	CICS/VSE Generation-2.3.0	14V	No	
	CICS/VSE RCF-2.3.0	14W	Yes	UN96558
5686-028	VSE/SP Unique Code-5.2.0	DB0	Yes	UN98449
	VSE/SP Unique Code Eng.-5.2.0	DB1	Yes	UN98450
	VSE/SP Unique Code Kan.-5.2.0	DB2	Yes	UN98451
	VSE/SP Unique Code Ger.-5.2.0	DB3	Yes	UN98452
	VSE/SP Unique Code Spa.-5.2.0	DB4	Yes	UN98453
	VSE/ESA DWF-6.1.0	DF0	No	
5686-032	VSE/AF IOCP-5.2.0	CE7	No	
	VSE/AF Assembler-5.2.0	DB6	No	
	VSE/AF CONT-LIB-5.2.0	DB6	Yes	UD50174
	VSE/AF ICA Utilities-5.2.0	DB6	No	
	VSE/AF Info Analysis-5.2.0	DB6	Yes	UD50176
	VSE/AF MCR/OCR-5.2.0	DB6	No	
	VSE/AF MSHP-5.2.0	DB6	Yes	UD50173
	VSE/AF Gen. Feature-5.2.0	DB7	Yes	UD50175
5686-033	VSE/POWER-5.2.0	DA9	Yes	UD50182
5686-034	VSE/FASTCOPY-2.2.0	DC2	Yes	UD50171
5686-036	VSE/ICCF-3.2.0	H22	Yes	UN98603
5686-037	VSE/VSAM-2.2.0	DA5	Yes	UD50148
	VSE/VSAM VTAM Macros-2.2.0	DA6	No	
	VSE/VSAM Space Manager-2.2.0	DA7	Yes	UD50153
	VSE/VSAM Backup/Restore-2.2.0	DA8	Yes	UD50147
5688-052	DITTO-3.2.0	CB0	Yes	PN91464
5746-RC5	BTAM ES-1.1.0	I08	No	
5747-DS2	ICKDSF-1.16.0	1G0	Yes	UN88673

A.4.2 VSE/ESA 1.4.2 Optional Products

Table 17 (Page 1 of 2). Product Information for VSE/ESA 1.4.2 Optional Products

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5648-061	QMF/VSE Base-3.2.0	17L	Yes	UN90787
	QMF/VSE Nat lang (CLCs 17M-17X)-3.2.0	17N	No	
5648-063	ACF/NCP-7.4.0	44A	No	
	ACF/NCP F-7.4.0	4EA	No	
5648-078	Vis.Gen. Host Serv. Basic + NLS-1.1.0	15K	No	
5648-109	VisualLift VSE-1.1.2	16T	No	
5648-113	AFP/Font Collection-1.1.0 (CLCs 140-15F)	140	Yes	?
5666-322	ACF/SSP-3.6.0	F65	No	
5668-719	X.25 NPSI (3725)-2.1.0	B60	No	
5668-723	GDDM IVU-1.1.2	2F4	No	
5668-738	ACF/NCP-5.4.0	CH0	No	
	ACF/NCP UT 2.0-5.0 (CLCs CH1-CH5)-5.4.0	CH1	No	
5668-801	GDDM IMD-2.1.2	CV5	No	
5668-812	GDDM PGF-2.1.2	7H8	No	
5668-813	CSP/AD/PWS-3.3.0	A10	Yes	VisualGen
	CSP/AD-3.3.0	D66	Yes	VisualGen
5668-814	CSP/AE-3.3.0	D67	Yes	VisualGen
5668-854	ACF/NCP/VS 3725-4.3.1	D34	No	
5668-958	VS COBOL II Comp + Lib + Debug-1.4.0	40A	Yes	COBOL/VSE LE/VSE
5686-011	CICSVR-1.1.0	A10	Yes	?
5686-013	NetView FTP-1.1.1	CX8	Yes	?
5686-018	CICS-DDM/VSE-1.1.0	A10	Yes	?
5686-022	DW/370-2.1.0	A10	No	
	DW/370-NLS (CLCs A11-A43)-2.1.0	A11	No	
	DW/DI-NLS (CLCs A24-A44)-2.1.0	A24	No	
5686-040	PSF/VSE-2.2.1	DC0	Yes	UD50146
	CODEPAGE.B240-2.2.1	FR9	No	
	CODEPAGE.B300-2.2.1	FS0	No	
	COMPATS.U240-2.2.1	DH0	No	
	COMPATS.U300-2.2.1	DH1	No	
	ACIF/VSE-2.2.1	FW0	No	
	AFPAPI/VSE-2.2.1	FW1	No	
5686-041	VSE/DSNX-2.1.2	E62	Yes	UN85348
5686-055	NetView Base VSE-2.3.0	3II	Yes	UD49976 UD50208
	NetView NLS (CLCs 3AA-3BB) VSE-2.3.0	3AA	No	
5686-057	GDDM/VSE-3.1.1	13V	Yes	UN88288 UN79821
	GDDM/VSE NLS (CLCs 13W-14C)-3.1.1	13W	No	

Table 17 (Page 2 of 2). Product Information for VSE/ESA 1.4.2 Optional Products

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5686-058	VSE/REXX Interface-1.1.0	DC7	No	
	VSE/REXX Kernel-1.1.0	DC7	No	
	VSE/REXX Library-1.1.0	DC7	No	
5686-064	ACF/SSP-4.4.0	44F	No	
5686-067	Lang.Envir. Base + NLS-1.1.0	18A	No	
5686-068	IBM COBOL for VSE/ESA-1.1.0	18M	Yes	UN95059
	IBM COBOL for NLS Mixed Case-1.1.0	18N	Yes	UN95060
	IBM COBOL for NLS Japanese-1.1.0	18O	Yes	UN95061
5686-069	IBM PL/I for VSE/ESA-1.1.0	18P	Yes	UN97019 UN96009
5686-072	ALERT/VSE-4.9.0	S00		Vendor product
5686-073	ADSM/VSE-1.2.0	14Z	No	
5686-079	ALERT/CICS-4.9.0	S04		Vendor product
5688-022	VS COBOL II Lib.-1.4.0	40D	Yes	COBOL/VSE LE/VSE
5688-023	VS COBOL II Comp. + Lib.-1.4.0	40A	Yes	COBOL/VSE LE/VSE
5688-035	X.25 NPSI (3745)-3.4.0	CJ0	No	
5688-052	DITTO Prod Feature-3.2.0	FJ4	Yes	PN91464
5688-103	SQL/DS-3.5.0	F50	No	
5688-187	C/370 Compiler-2.1.0	2A2	Yes	C/VSE LE/VSE
5688-188	C/370 Runtime lib-2.1.0	2A3	Yes	C/VSE LE/VSE
5688-190	PPFA/VSE-1.1.0	A03	No	
5688-191	OGL/370 English-1.1.0	AA0	Yes	UN93115
	OGL/370 German-1.1.0	AA1	Yes	UN93116
	OGL/370 Japanese-1.1.0	AA2	Yes	UN93117
	OGL/370-1.1.0	A05	Yes	UN93034
5696-234	High Level Assembler-1.2.0	189	No	
5735-XXB	EP for ACF/NCP 3725-1.6.1	F12	No	
	EP 3745 Without NCP-1.8.0	I59	No	
	EP for ACF/NCP-1.9.0	CH6	No	
	EP for ACF/NCP-1.12.0	4A1	No	
5746-CB1	COBOL Compiler-1.3.1	E46	Yes	COBOL/VSE LE/VSE
5746-LM4	COBOL Library-1.3.1	E47	Yes	COBOL/VSE LE/VSE
5746-RG1	DOS/VS RPG II-1.3.0	O42	YES	UN95321
5746-SM3	DFSORT/VSE-3.2.0	32A	YES	PN81797
5746-XC5	OCCF-1.4.1	DA1	No	
5746-XE7	VSE/ACLR-1.2.1	H06	Yes	?
5746-XXT	SDF/CICS-1.5.0	B27	No	
5746-XX1	VSE/DL/I-1.10.0	DB5	Yes	UN94805

A.5 Product Information for VSE/ESA 1.4.1

Table 18. Product Information for VSE/ESA 1.4.1

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5648-061	QMF/VSE Base-3.1.1	190	Yes	UN90788
	QMF/VSE Nat. Lang. (CLCs 191-19F)-3.1.1	191	No	
5648-063	ACF/NCP-7.1.0	3A0	No	
	ACF/NCP F-7.1.0	3E0	No	
5666-270	DISOSS-3.4.0	I60	Yes	PN89310
5666-318	PS/CICS + NLS-1.3.1	D39	Yes	No change
5686-064	ACF/SSP-4.1.0	F41	No	
5688-103	SQL/DS-3.4.0	F40	Yes	SQL/DS 3.5

A.6 Product Information for VSE/ESA 1.4.0

Table 19. Product Information for VSE/ESA 1.4.0

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5648-040	Vis Gen Developer Basic-1.1.0	15Q	No	
	Vis Gen Devel NLS (CLCs 15R-15U)-1.1.0	15R	No	
5648-061	QMF/VSE Base-3.1.1	190	Yes	UN90788
	QMF/VSE NLS (CLCs 191-19F)-3.1.1	191	No	
5648-063	ACF/NCP-7.1.0	3A0	No	
	ACF/NCP F-7.1.0	3E0	No	
5648-076	Vis Gen Work Group Basic + NLS-1.1.0	15V	No	
5648-086	Vis Gen Appl Gener incl Gen Opt-1.1.0	15N	No	
	Vis Gen Appl Gener Opt VSE-1.1.0	165	No	
5666-270	DISOSS-3.4.0	I60	Yes	PN89310
5666-318	PS/CICS + NLS-1.3.1	D39	Yes	No change
5666-329	CICSPARS VSE-1.1.1	D38	Obsolete	
5686-064	ACF/SSP-4.1.0	F41	No	
5686-075	ASF/VSE Base-3.1.0	166	No	
5688-103	SQL/DS-3.4.0	F40	Yes	SQL/DS 3.5
5736-LM4	PL/I Resident Library-1.6.0	N72	Yes	PL/I VSE LE/VSE
5736-LM5	PL/I Transient Library-1.6.0	N73	Yes	PL/I VSE LE/VSE
5736-PL1	PL/I Opt Compiler-1.6.0	N74	Yes	PL/I VSE LE/VSE
5736-PL3	PL/I Comp/Lib-1.6.0	N74	Yes	PL/I VSE LE/VSE
5746-SM3	DFSORT/VSE-3.1.0	31A	Yes	DFSORT/VSE 3.2

A.7 Optional Products not Part of a Package

Table 20. Product Information for VSE/ESA Optional Products

Pid Number	Program Name	CLC	Affected	PTF/APAR Number
5668-890	Font LIB Serv Fac-1.1.0	H18	No	
5668-985	Host Comm Fac-2.1.0	E34	No	
5684-136	SQL Master for VSE	1ES	Yes	?
5686-063	MERVA/VSE CICSBASE-3.2.0	186	No	
	MERVA/VSE CICS21-3.2.0	187	No	
	MERVA/VSE CICS22-3.2.0	188	No	
5746-AM1	Interpers V1.2.5	250	Yes	PN91198
5746-XC4	DMS/CICS/VS-1.5.0	AAE	Yes	UN68923
	DMS/CICS/VS-1.5.0	AAG	Yes	UN68925
5748-FO3	VS FORTRAN Compiler and Library-1.4.1	D41	Yes	PN91583
5748-LM3	VS FORTRAN Library-1.4.1	D42	No	
5748-XX9	DCF/VSE-1.4.0	1D5	Yes	UN89996
	DCF/VSE ATMS Feature-1.4.0	1D6	Yes	UN89998
	DCF/VSE DCF Feature-1.4.0	1D7	Yes	UN90400
5748-XXE	DLF/VSE-1.3.0	H86	Yes	UN98514
5785-CCA	CCCA/VSE-1.1.0	18W	No	
5985-DYR	Cobol Report Writer Precomp.1.4.1		Yes	?

Appendix B. Partition Communication Region (COMREG)

In a VSE/ESA system the date is usually obtained from the COMREG. Figure 17 shows the date fields available in the COMREG.

```
000       COMREG      DSECT
000       JOBDATWC    DS    0CL11      JOB DATE WITH CENTURY
000       JOBDATE     DS    CL8        JOB DATE
008                   DS    CL3        CENTURY
00B                   DS    X          RESERVED
*         *           *     *          *
*         *           *     *          *
035       LTACT       DS    X          SYSTEM CONFIGURATION BYTE
          IJBDFORM    EQU   X¢80¢       0 DATE FORMAT DDMMYY
          MPSMSK      EQU   X¢40¢       1 MPS SUPPORTED
          DASDFPSW    EQU   X¢20¢       2 DASDFP ACTIVE
          TPMSK       EQU   X¢08¢       4 TP SUPPOTED
*         *           *     *          *
*         *           *     *          *
03B       JCSW4       DS    X          JOB DURATION INDICATOR BYTE
                            0  JOB IN PROGRESS
          OPTDUMP     EQU   X¢40¢       1  OPTION DUMP
                                        2  PAUSE AT END OF STEP
                                        3  JC OUTPUT TO SYSLST
                                       4-5 RESERVED
          DATEBIT     EQU   X¢02¢       6  // DATE PRESENT
          BATINIT     EQU   X¢01¢       7  BATCH COMMAND ISSUED
*         *           *     *          *
*         *           *     *          *
04F       SYSDATE     DS    0CL9       SYSTEM DATE
04F       MMDD        DS    XL4          MMDD OR DDMM
053       YYDDD       DS    XL5          YYDDD PORTION OF DATE
*         *           *     *          *
*         *           *     *          *
064       SYSCENT     DS    H           CENTURY OF SYSDATE
066                   DS    XL4         RESERVED
*         *           *     *           *
*         *           *     *           *
0E8       JCSW9       DS    X           JCL SWITCH 9
          IJBASIVA    EQU   X¢80¢       0 ASI PROC NAME AVAILABLE
          IJBRSTRT    EQU   X¢40¢       1 PART. IS AUTO UNBATCHED
          IJBWASDT    EQU   X¢20¢       2 DATA MODE RESET (TEMP.)
          IJBREXX     EQU   X¢10¢       3 REXX ACTIVE
          IJBUNBCH    EQU   X¢08¢       4 UNBATCH MUST BE DONE
          IJBDATEO    EQU   X¢04¢       5 DATE WITH OLD FORMAT
*         *           *     *          *
*         *           *     *          *
128                   DS    F           RESERVED
          COMREND     EQU   *
```

Figure 17. Partition Communication Region

Appendix C. Special Notices

This publication is intended to assist customer and IBM technical personnel in the conversion of their high level language programs to cope with the 4-digit date format which is required for the turn of the century. If using different languages the unique run-time platform offered by LE/VSE will alleviate this programming effort.

The information in this publication is not intended as a specification of any programming interfaces that are provided by VSE/ESA or LE/VSE. See the PUBLICATIONS section of the IBM Programming Announcement for these products for more information about what publications are considered to be product documentation.

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 may be used. Any functionally equivalent program that does not infringe any of IBM′s intellectual property rights may be used instead of the IBM product, program or service.

Information in this book was developed in conjunction with use of the equipment specified, and is limited in application to those specific hardware and software products and levels.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the IBM Director of Licensing, IBM Corporation, 500 Columbus Avenue, Thornwood, NY 10594 USA.

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, Dept. 600A, Mail Drop 1329, Somers, NY 10589 USA.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The information contained in this document has not been submitted to any formal IBM test and is distributed AS IS. The information about non-IBM (″vendor″) products in this manual has been supplied by the vendor and IBM assumes no responsibility for its accuracy or completeness. The use of this information or the implementation of any of these techniques is a customer responsibility and depends on the customer′s ability to evaluate and integrate them into the customer′s operational environment. While each item may have been reviewed by IBM for accuracy in a specific situation, there is no guarantee that the same or similar results will be obtained elsewhere. Customers attempting to adapt these techniques to their own environments do so at their own risk.

Any performance data contained in this document was determined in a controlled environment, and therefore, the results that may be obtained in other

operating environments may vary significantly. Users of this document should verify the applicable data for their specific environment.

Reference to PTF numbers that have not been released through the normal distribution process does not imply general availability. The purpose of including these reference numbers is to alert IBM customers to specific information relative to the implementation of the PTF when it becomes available to each customer according to the normal IBM PTF distribution process.

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

ACF/VTAM	AFP
APPN	C/370
CICS	CICS/VSE
Current	DATABASE 2
DB2	DFSORT
GDDM	IBM
IIN	ILE
ISSC	Language Environment
MERVA	NetView
PR/SM	Print Services Facility
Processor Resource/Systems Manager	PS/2
PSF	QMF
SKI	SQL/DS
System/390	VisualGen
VM/ESA	VSE/ESA
VTAM	XT
3090	

The following terms are trademarks of other companies:

C-bus	Corollary, Inc.
DOS	Microsoft Corporation
HP	Hewlett-Packard Company
PC Direct	Ziff Communications Company (used by IBM Corporation under license)
SX	Intel Corporation
UNIX	X/Open Company Ltd. (registered trademark in the United States and other countries)
Windows, Windows 95 logo	Microsoft Corporation
386	Intel Corporation

Appendix D. Related Publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

D.1 International Technical Support Organization Publications

For information on ordering these ITSO publications see “How To Get ITSO Redbooks” on page 167.

- *VisualGen: The Future of Your VSE Applications and How to Get There from CSP*, SG24-4545
- *Migration to VSE/ESA 2.1 - Why and How*, SG24-4773
- *Taking Advantage of IBM Language Environment for VSE/ESA*, SG24-4798

D.2 Redbooks on CD-ROMs

Redbooks are also available on CD-ROMs. **Order a subscription** and receive updates 2-4 times a year at significant savings.

CD-ROM Title	Subscription Number	Collection Kit Number
System/390 Redbooks Collection	SBOF-7201	SK2T-2177
Networking and Systems Management Redbooks Collection	SBOF-7370	SK2T-6022
Transaction Processing and Data Management Redbook	SBOF-7240	SK2T-8038
AS/400 Redbooks Collection	SBOF-7270	SK2T-2849
RISC System/6000 Redbooks Collection (HTML, BkMgr)	SBOF-7230	SK2T-8040
RISC System/6000 Redbooks Collection (PostScript)	SBOF-7205	SK2T-8041
Application Development Redbooks Collection	SBOF-7290	SK2T-8037
Personal Systems Redbooks Collection	SBOF-7250	SK2T-8042

D.3 Other Publications

These publications are also relevant as further information sources.

Language Environment for VSE/ESA Release 4

- *Installation and Customization Guide*, SC33-6682
- *Run-Time Migration Guide*, SC33-6687
- *C Run-Time Programming Guide*, SC33-6688

C for VSE/ESA

- *Installation and Customization Guide*, GC09-2422
- *Migration Guide*, SC09-2423
- *User's Guide*, SC09-2424
- *Language Reference*, SC09-2425

COBOL for VSE/ESA

- *Migration Guide*, GC26-8070

- *Installation and Customization Guide*, SC26-8071
- *Programming Guide*, SC26-8072
- *Language Reference*, SC26-8073

PL/I for VSE/ESA

- *Programming Guide*, SC26-8053
- *Language Reference*, SC26-8054
- *Migration Guide*, SC26-8056
- *Installation and Customization Guide*, SC26-8057

COBOL and CICS Command Level Conversion Aid for VSE

- *CCCA/VSE Installation and User's Guide*, SC26-8269

VSE/ESA Publications

- *VSE/ESA General Information - What's New V2.1 and V2.2*, GC33-6627
- *VSE/ESA Enhancements V2.2*, SC33-6629
- *VSE/ESA Planning V2.2*, SC33-6603
- *VSE/ESA Installation V2.1*, SC33-6604
- *VSE/ESA System Upgrade and Service V2.1*, SC33-6602
- *CICS for VSE/ESA Data Areas*, LY33-6075

Year 2000 related publications

- *VSE/ESA Software Newsletter Special Issue - VSE and Year 2000*, G225-4508
- *The Year 2000 and 2-Digit Dates: A Guide for Planning and Implementation*, GC28-1251, on the World Wide Web at *http://www.software.ibm.com/year2000/index.html*
- *VSE/ESA Home Page*, on the World Wide Web at *http://www.ibm.de/go/d00000166*

Softcopy Publications

The following collection kit contains LE/VSE and LE/VSE-conforming language product publications:

- *VSE Collection*, SK2T-0060

How To Get ITSO Redbooks

This section explains how both customers and IBM employees can find out about ITSO redbooks, CD-ROMs, workshops, and residencies. A form for ordering books and CD-ROMs is also provided.

This information was current at the time of publication, but is continually subject to change. The latest information may be found at URL http://www.redbooks.ibm.com.

How IBM Employees Can Get ITSO Redbooks

Employees may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **PUBORDER** — to order hardcopies in United States
- **GOPHER link to the Internet** - type GOPHER.WTSCPOK.ITSO.IBM.COM
- **Tools disks**

  To get LIST3820s of redbooks, type one of the following commands:

  ```
  TOOLS SENDTO EHONE4 TOOLS2 REDPRINT GET SG24xxxx PACKAGE
  TOOLS SENDTO CANVM2 TOOLS REDPRINT GET SG24xxxx PACKAGE (Canadian users only)
  ```

  To get lists of redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS REDBOOKS GET REDBOOKS CATALOG
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET ITSOCAT TXT
  TOOLS SENDTO USDIST MKTTOOLS MKTTOOLS GET LISTSERV PACKAGE
  ```

  To register for information on workshops, residencies, and redbooks:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ITSOREGI 1996
  ```

  For a list of product area specialists in the ITSO:

  ```
  TOOLS SENDTO WTSCPOK TOOLS ZDISK GET ORGCARD PACKAGE
  ```

- **Redbooks Home Page on the World Wide Web**

  http://w3.itso.ibm.com/redbooks

- **IBM Direct Publications Catalog on the World Wide Web**

  http://www.elink.ibmlink.ibm.com/pbl/pbl

  IBM employees may obtain LIST3820s of redbooks from this page.

- **REDBOOKS category on INEWS**
- **Online** — send orders to: USIB6FPL at IBMMAIL or DKIBMBSH at IBMMAIL
- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank). A category form and detailed instructions will be sent to you.

How Customers Can Get ITSO Redbooks

Customers may request ITSO deliverables (redbooks, BookManager BOOKs, and CD-ROMs) and information about redbooks, workshops, and residencies in the following ways:

- **Online Orders** (Do not send credit card information over the Internet) — send orders to:

	IBMMAIL	**Internet**
In United States:	usib6fpl at ibmmail	usib6fpl@ibmmail.com
In Canada:	caibmbkz at ibmmail	lmannix@vnet.ibm.com
Outside North America:	dkibmbsh at ibmmail	bookshop@dk.ibm.com

- **Telephone orders**

United States (toll free)	1-800-879-2755
Canada (toll free)	1-800-IBM-4YOU
Outside North America	(long distance charges apply)
(+45) 4810-1320 - Danish	(+45) 4810-1020 - German
(+45) 4810-1420 - Dutch	(+45) 4810-1620 - Italian
(+45) 4810-1540 - English	(+45) 4810-1270 - Norwegian
(+45) 4810-1670 - Finnish	(+45) 4810-1120 - Spanish
(+45) 4810-1220 - French	(+45) 4810-1170 - Swedish

- **Mail Orders** — send orders to:

IBM Publications	IBM Publications	IBM Direct Services
Publications Customer Support	144-4th Avenue, S.W.	Sortemosevej 21
P.O. Box 29570	Calgary, Alberta T2P 3N5	DK-3450 Allerød
Raleigh, NC 27626-0570	Canada	Denmark
USA		

- **Fax** — send orders to:

United States (toll free)	1-800-445-9269
Canada	1-403-267-4455
Outside North America	(+45) 48 14 2207 (long distance charge)

- **1-800-IBM-4FAX (United States)** or **(+1) 415 855 43 29 (Outside USA)** — ask for:

  Index # 4421 Abstracts of new redbooks
  Index # 4422 IBM redbooks
  Index # 4420 Redbooks for last six months

- **Direct Services** - send note to softwareshop@vnet.ibm.com
- **On the World Wide Web**

Redbooks Home Page	http://www.redbooks.ibm.com
IBM Direct Publications Catalog	http://www.elink.ibmlink.ibm.com/pbl/pbl

- **Internet Listserver**

  With an Internet E-mail address, anyone can subscribe to an IBM Announcement Listserver. To initiate the service, send an E-mail note to announce@webster.ibmlink.ibm.com with the keyword subscribe in the body of the note (leave the subject line blank).

IBM Redbook Order Form

Please send me the following:

Title	Order Number	Quantity

- **Please put me on the mailing list for updated versions of the IBM Redbook Catalog.**

First name Last name

Company

Address

City Postal code Country

Telephone number Telefax number VAT number

- Invoice to customer number _____
- Credit card number _____

Credit card expiration date Card issued to Signature

We accept American Express, Diners, Eurocard, Master Card, and Visa. Payment by credit card not available in all countries. Signature mandatory for credit card payment.

DO NOT SEND CREDIT CARD INFORMATION OVER THE INTERNET.

List of Abbreviations

ACB	Access control block
AFP	Advanced Function Printer
ANSI	American National Standards Institute
APAR	Authorized program analysis report
API	Application Program Interface
AR	Attention routine
ASC	Address space control
ASI	Automated System Initialization
ATMS	Advanced Text Management System
BAM	Basic Access Method
BSC	Binary Synchronous Communication
BTAM	Basic Telecommunications Access Method
CICS	Customer Information Control System
CMS	Conversational Monitor System
COBOL	COmmon Business Oriented Language
COMREG	Communication region
CSP	Cross-System Product
DASD	Direct Access Storage Device
DB2	DATABASE 2
DCF	Document Composition Facility
DOS	Disk Operating System
DTF	Development test facility
EC	Engineering change
EIB	Error information block
ESA	Enterprise Systems Architecture
FSU	Field serviceable unit
GDDM	Graphical Data Display Manager
I/O	Input/output
IBM	International Business Machines
ICA	Integrated Communication Adapter
IDCAMS	Program name for Access Method Services
IPL	Initial Program Load
ISAM	Indexed Sequential Access Method
ISQL	Interactive Structured Query Language
JCL	Job control language
LCP	Language conversion program
LE	Language Environment
MPS	Multiple partition support
MRI	Machine readable information
MSHP	Maintain system history program
NETVIEW	Network observation tool
NJE	Network job entry
NLDM	Network logical data manager
NLS	National language support
NPDA	Network Problem Determination Aid
NPSI	NCP Packet Switching Interface
PC	Personal computer
PL/I	Programming Language 1
PNET	Power NETworking interface
POWER	Priority Output Writers, Execution processor, and input Readers
PR/SM	Processor Resource/Systems Manager
PSB	Program status block
PSF/VSE	Print Services Facility/Virtual Storage Extended
PTF	Program temporary fix
RC	Return code
RCF	Report Controller Feature
REXX	REstructured eXtended eXecutor language
RPG	Report Program Generator
RPG II	Report Program Generator II
SAM	Sequential Access Method
SDF/CICS	Screen definition Facility/Customer Information Control System

SDL	System directory list
SMF	Source macro file
SNA	Systems Network Architecture
SQL	Structured Query Language
SQL/DS	Structured Query Language/Data System
SVA	Shared virtual area
TOD	Time of day
TP	Teleprocessing
VM	Virtual Machine
VM/ESA	Virtual Machine/Enterprise Systems Architecture
VSAM	Virtual Storage Access Method
VSE	Virtual Storage Extended
VSE/ESA	Virtual Storage Extended/Enterprise Systems Architecture
VSE/ICCF	Virtual Storage Extended/Interactive Computing and Control Facility
VSE/POWER	Virtual Storage Extended/Priority Output Writers, Execution processor, and input Readers
VSE/SP	Virtual Storage Extended/System Package
VSE/VSAM	Virtual Storage Extended/Virtual Storage Access Method
VTAM	Virtual Telecommunications Access Method

Index

Special Characters

A

B

C

D

E

F

G

I

J

L

M

N

O

P

Q

R

S

T

U

V

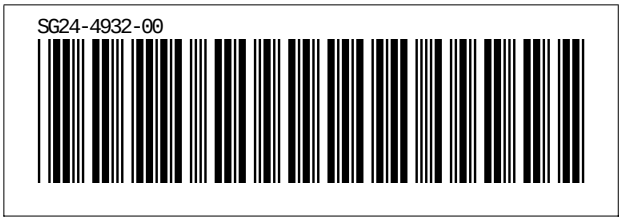
W

X

Y

Z

Printed in U.S.A.



Artwork Definitions

id	File	Page	References
ITLOGO	4932SU	i	i
ITLOGOS	4932SU	i	

Table Definitions

id	File	Page	References
PT1	4932VARS	i	150, 150, 150, 150, 151, 151, 151, 151, 151, 151, 151, 151, 151, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 152, 153, 153, 153, 153, 153, 153, 153, 153, 153, 153, 153, 153, 153, 153, 154, 154, 154, 154, 154, 154, 154, 154, 154, 154, 154, 154, 154, 154, 155, 155, 155, 155, 155, 155, 155, 155, 155, 155, 156, 156, 156, 156, 156, 156, 156, 156, 156, 156, 156, 156, 156, 156, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 157, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 158, 159, 159, 159, 159, 159, 159, 159, 159, 159, 159, 159
PT2	4932VARS	i	151, 151, 151, 152, 152, 152, 153, 153, 154, 154, 154, 156, 156, 156, 156, 156, 158, 158, 158, 158, 158, 158, 159
PT3	4932VARS	i	150, 152, 152, 155, 156, 157, 157, 159, 159
PT4	4932VARS	i	153, 153, 155, 157, 157
PT5	4932VARS	i	151
PT6	4932VARS	i	155
PT7	4932VARS	i	152, 156
PT8	4932VARS	i	155
PT9	4932VARS	i	
PT10	4932VARS	i	150
PT11	4932VARS	i	
PT12	4932VARS	i	
PT13	4932VARS	i	151
TAB0	4932VARS	i	i, i, i, 62, 78, 86
R1	4932VARS	i	62
R2	4932VARS	i	62
R3	4932VARS	i	62

Figures

id	File	Page	References
SETDAT1	4932CH4	23	1
SETDAT2	4932CH4	23	2
COBOLA	4932CH5	56	3 55, 60
COBOLB	4932CH5	60	4 55, 60
COBOLC	4932CH5	61	5
Y2CBL1	4932CH5	65	6

Headings

Index Entries

Tables

Processing Options

Runtime values:

Document fileid	SG244932 SCRIPT
Document type	USERDOC
Document style	REDBOOK
Profile	EDFPRF30
Service Level	0029
SCRIPT/VS Release	4.0.0
Date	96.12.11
Time	06:20:48
Device	3820A
Number of Passes	4
Index	YES
SYSVAR D	YES
SYSVAR G	INLINE
SYSVAR S	OFFSET
SYSVAR X	YES

Formatting values used:

Annotation	NO
Cross reference listing	YES
Cross reference head prefix only	NO
Dialog	LABEL
Duplex	YES
DVCF conditions file	(none)
DVCF value 1	(none)
DVCF value 2	(none)
DVCF value 3	(none)
DVCF value 4	(none)
DVCF value 5	(none)
DVCF value 6	(none)
DVCF value 7	(none)
DVCF value 8	(none)
DVCF value 9	(none)
Explode	NO
Figure list on new page	YES
Figure/table number separation	YES
Folio-by-chapter	NO
Head 0 body text	Part
Head 1 body text	Chapter
Head 1 appendix text	Appendix
Hyphenation	NO
Justification	NO
Language	ENGL
Layout	OFF
Leader dots	YES
Master index	(none)
Partial TOC (maximum level)	4
Partial TOC (new page after)	INLINE
Print example id's	NO
Print cross reference page numbers	YES
Process value	(none)
Punctuation move characters	.,
Read cross-reference file	(none)
Running heading/footing rule	NONE
Show index entries	NO
Table of Contents (maximum level)	3
Table list on new page	YES
Title page (draft) alignment	RIGHT
Write cross-reference file	(none)

Imbed Trace

Page 0	4932SU
Page 0	4932VARS
Page 0	REDB$BOE
Page i	REDB$ED1
Page i	4932EDNO
Page i	REDB$ED2
Page xi	4932ABST
Page xi	4932ORG
Page xi	4932ACKS
Page xii	REDB$COM
Page xii	4932MAIN
Page xii	4932CH1
Page 7	4932CH2
Page 12	4932CH3
Page 20	4932CH4
Page 50	4932CH5
Page 94	4932CH6
Page 104	4932CH7
Page 148	4932AX1
Page 159	4932AX2
Page 163	4932SPEC
Page 163	REDB$SPE
Page 164	4932TMKS
Page 164	4932BIBL
Page 165	REDB$BIB
Page 166	REDB$ORD
Page 169	4932ABRV