

Automatic Binary Optimizer for z/OS
V2.1

用户指南



注

在使用本资料及其支持的产品前，请务必阅读第 75 页的『声明』中的常规信息。

第一版（2019 年 9 月）

此版本适用于 IBM® Automatic Binary Optimizer for z/OS® V2.1（程序编号 5697-AB2）和 IBM Automatic Binary Optimizer for z/OS Trial（程序编号 5697-TR2）以及所有后续发行版和修订版，直至在新版本中另有声明为止。确保您使用的是本产品级别的正确版本。

您可以在 www.ibm.com/shop/publications/order/ 上免费查看或下载出版物软拷贝。

© Copyright International Business Machines Corporation 2015, 2019.

目录

表.....	vii
前言.....	ix
关于本书.....	ix
缩写词汇.....	ix
如何阅读语法图.....	ix
更改汇总.....	x
如何发送您的意见.....	x
Automatic Binary Optimizer for z/OS 的辅助功能.....	xi
第 1 章概述.....	1
优势.....	1
将 ABO 与 Enterprise COBOL 一起使用.....	1
第 2 章系统需求.....	3
受支持的操作系统.....	3
目标硬件级别.....	4
第 3 章 COBOL 模块需求.....	5
符合条件的编译器.....	5
COBOL 语言功能和编译器选项支持.....	6
处理不符合条件的 CSECT.....	7
第 4 章安装并验证安装.....	9
安装 IBM Automatic Binary Optimizer for z/OS.....	9
使用“安装验证程序”(IVP) 验证安装.....	9
第 5 章优化模块.....	13
必需的 DD 语句.....	13
优化器伪指令.....	14
BOPT.....	14
IEFOPZ.....	15
优化器选项.....	17
ALLOW.....	17
ARCH.....	18
CSECT.....	18
LIST.....	21
LOG.....	21
REPLACE.....	22
SCAN.....	23
注释.....	23
行连续符.....	24
JCL 示例.....	25
使用 BOPT 指定优化.....	25
使用 IEFOPZ 指定优化.....	27
z/OS JCL REGION 和 JCL MEMLIMIT 参数的建议设置.....	28
指定要用于 ABO 消息的语言.....	29
通过 TSO、REXX 和汇编程序代码来调用 ABO.....	29
在 TSO 下进行优化.....	29

从汇编程序中启动优化器.....	34
第 6 章了解优化流程的输出.....	37
日志文件.....	37
列表转换.....	39
列表转换内容.....	39
SYSPRINT DD 和 LIST 选项.....	45
第 7 章管理优化和优化模块部署流程.....	47
优化和部署使用场景.....	47
场景 1：使用静态部署的优化流程.....	47
场景 2：使用动态部署的优化流程.....	48
场景 3：使用混合方法的优化流程.....	49
测试信息.....	50
第 8 章解决在执行优化和部署优化模块时遇到的问题.....	51
解决在优化期间发生的问题.....	51
解决在执行期间遇到的问题.....	51
优化后的 COBOL 模块大小变更.....	51
错误信息和异常终止代码差别.....	52
Application Delivery Foundation for z Systems	52
Run Time Instrumentation Profiler.....	53
附录 A JCL 示例.....	57
附录 B 返回码.....	59
附录 C 消息.....	61
BOZ1003U.....	61
BOZ1031S.....	61
BOZ1145U.....	61
BOZ1400S.....	61
BOZ1401S	61
BOZ1402S.....	62
BOZ1403S.....	62
BOZ1404S.....	62
BOZ1405S.....	62
BOZ1406S.....	62
BOZ1407S.....	62
BOZ1408S.....	63
BOZ1409S.....	63
BOZ1410I.....	63
BOZ1411S.....	63
BOZ1412S.....	63
BOZ1413S.....	63
BOZ1414S.....	64
BOZ1415S.....	64
BOZ1416S.....	64
BOZ1417S.....	64
BOZ1418S.....	64
BOZ1419S.....	64
BOZ1420S.....	65
BOZ1421S.....	65
BOZ1422S.....	65
BOZ1423S.....	65
BOZ1424S.....	65

BOZ1428U	66
BOZ1429U	66
BOZ1430U	66
BOZ1431S	66
BOZ1432S	66
BOZ1436S	67
BOZ1437S	67
BOZ1438U	67
BOZ1439U	67
BOZ1446U	67
BOZ1447U	67
BOZ1449U	68
BOZ1450U	68
BOZ1451S	68
BOZ1452S	68
BOZ1453U	68
BOZ1455W	68
BOZ1456S	69
BOZ1457S	69
BOZ1490W	69
BOZ1491W	69
BOZ1492W	70
BOZ1493S	70
BOZ1494S	70
BOZ4089I	70
BOZ4091I	70
BOZ4092I	70
BOZ4097I	71
BOZ4101W	71
BOZ4107I	71
BOZ4109I	71
BOZ4110I	71
BOZ4111I	72
BOZ4113I	72
BOZ4114I	72
BOZ4116I	72
BOZ4117I	72
BOZ4119S	72
BOZ4120S	73
BOZ4121S	73
BOZ4124I	73

声明.....	75
商标.....	75
资源列表	77
IBM Automatic Binary Optimizer for z/OS 出版物.....	77
相关出版物.....	77

1. 比较优化器与编译器的用例.....	1
2. 受支持的硬件级别.....	4
3. ABO 不支持的 COBOL 模块.....	5
4. 不符合条件的 CSECT 和发出的消息.....	7
5. 返回码和相应缺少的 LE PTF.....	11
6. 用于二进制文件优化的 DD 名称.....	13
7. 优化器选项.....	17
8. 输入模块及其包含的 CSECT.....	45
9. 输出 1：优化模块及其 CSECT.....	46
10. 输出 2：列表转换.....	46
11. 建议的分配参数.....	54
12. 建议的分配参数.....	57
13. IBM Automatic Binary Optimizer for z/OS 返回码.....	59

前言

关于本书

本书专供使用 IBM Automatic Binary Optimizer for z/OS 提高已编译的 COBOL 程序性能的 IBM COBOL 编译器客户使用。

缩写词汇

本信息中使用了某些词汇的缩写形式。下表中按字母顺序列出了最常用词汇的缩写。

所用词汇	详细格式
ABO	IBM Automatic Binary Optimizer for z/OS
CSECT	控制部分
EBCDIC	二进制编码的扩展十进制交换码
HFS	分层文件系统
JCL	作业控制语言
PDS	分区数据集
PDSE	扩展的分区数据集

对于其他一些不常见的缩写词汇，在第一次出现时会书写其详细格式。

如何阅读语法图

根据以下描述来阅读本信息中的语法图：

· 沿着线条的走向，按照从左至右、从上至下的顺序来阅读语法图。

➤➤--- 符号指示语法图的开头。

---➤ 符号指示语法图在下一行继续。

➤--- 符号指示语法图接着上一行来。

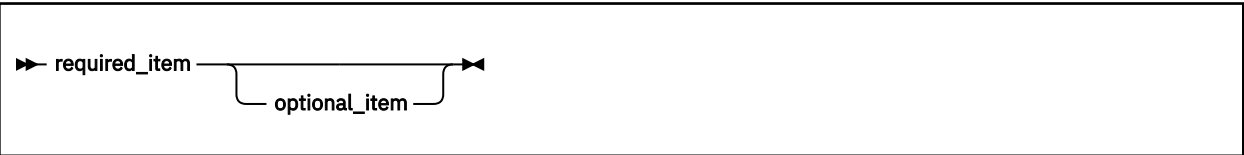
---➤< 符号指示语法图的结尾。

完整语句以外的语法单元图是以 ➤--- 符号开始并以 ---➤ 符号结束。

· 水平线（主路径）上显示必需项。



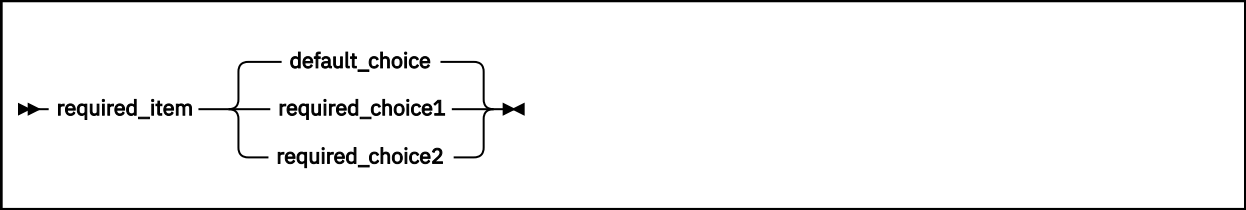
· 主路径下方显示可选项。



· 如果可以从两个或更多个项中进行选择，那么这些项将垂直显示在一个堆栈中。如果必须选择其中一个项，那么堆栈中的一个项将显示在主路径上。



如果其中一个项是缺省值，那么它将显示在主路径上方，而其余选项显示在其下方：



· 如果在主路径上方有一个返回到左侧的箭头，那么表示某个项是可重复项。



- 关键字采用全大写形式（例如，`FROM`）。必须完全按照所显示的那样拼写关键字。变量采用全小写形式（例如，`column-name`）。变量表示用户提供的名称或值。
- 如果显示了标点符号、圆括号、算术运算符或其他类似符号，那么必须将它们作为语法的一部分来输入。

更改汇总

本部分列出了此文档中针对 IBM Automatic Binary Optimizer for z/OS V2.1 所做的主要更改。在 HTML 版本中突出显示了最新技术更改，或者在 PDF 版本左页边距中用竖线 (|) 标记了最新技术更改。

V2.1

- 已将 z/OS V2.4 添加到受支持的操作系统列表中，并移除了 z/OS V2.1。（请参阅[受支持的操作系统](#)）
- 已添加 ARCH(13)，这用于生成可利用新的 IBM z15 大型机的代码。（请参阅[ARCH](#)）
- 已添加以下消息：BOZ4124I。（请参阅[消息](#)）
- 已移除以下优化器选项：HANDLERS。

如何发送您的意见

您的反馈对于帮助我们提供准确的高质量信息非常重要。如果您对本文档有任何意见，请通过以下某种方式联系我们：

- 使用联机读者意见表：<http://www.ibm.com/software/awdtools/rcf>。
- 将意见发送到以下地址：compinfo@cn.ibm.com。

请务必包含文档名称、出版物编号、产品版本以及（如果适用）您发表评论的文本的具体位置（例如，页码或章节标题）。

当您发送信息给 IBM 后，即授予 IBM 非专有权，IBM 可以按它认为适当的任何方式使用或分发您所提供的信息，而无须对您承担任何责任。

Automatic Binary Optimizer for z/OS 的辅助功能

辅助功能帮助身体有缺陷（例如行动有障碍或视力不佳）的用户顺利地使用信息技术内容。z/OS 中的辅助功能针对 Automatic Binary Optimizer for z/OS (ABO) 提供辅助功能。

辅助功能

z/OS 包含以下主要辅助功能：

- 屏幕朗读器和屏幕放大软件常用的界面
- 仅使用键盘进行的导航
- 能够定制显示属性（例如，颜色、对比度和字体大小）

z/OS 使用最新 W3C 标准 [WAI-ARIA 1.0](http://www.w3.org/TR/wai-aria/) (<http://www.w3.org/TR/wai-aria/>) 以确保符合 [US Section 508](http://www.access-board.gov/guidelines-and-standards/communications-and-it/about-the-section-508-standards/section-508-standards) (<http://www.access-board.gov/guidelines-and-standards/communications-and-it/about-the-section-508-standards/section-508-standards>) 和 [Web Content Accessibility Guidelines \(WCAG\) 2.0](http://www.w3.org/TR/WCAG20/) (<http://www.w3.org/TR/WCAG20/>)。要利用辅助功能，请使用屏幕朗读器的最新发行版以及此产品支持的最新 Web 浏览器。

[IBM Knowledge Center](http://www.ibm.com/support/knowledgecenter/en/about/releasenotes.html) 中的 ABO 联机产品文档支持辅助功能。<http://www.ibm.com/support/knowledgecenter/en/about/releasenotes.html> 描述了 IBM Knowledge Center 的辅助功能。

键盘导航

用户可以使用 TSO/E 或 ISPF 来访问 z/OS 用户界面。

用户还可使用 IBM Developer for z Systems® Enterprise Edition 来访问 z/OS 服务。

有关访问这些界面的信息，请参阅以下出版物：

- *z/OS TSO/E Primer* (http://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.ikjp100/toc.htm)
- *z/OS TSO/E User's Guide* (http://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.ikjc200/toc.htm)
- *z/OS ISPF User's Guide Volume I* (http://www.ibm.com/support/knowledgecenter/SSLTBW_2.2.0/com.ibm.zos.v2r2.f54ug00/toc.htm)
- *IBM Developer for z Systems Knowledge Center* (http://www.ibm.com/support/knowledgecenter/SSQ2R2/rdz_welcome.html?lang=en)

这些指南描述了如何使用 TSO/E 和 ISPF，包括使用键盘快捷键或功能键（PF 键）。每个指南都包括 PF 键的缺省设置，并说明了如何修改这些键的功能。

界面信息

[IBM Knowledge Center](https://www.ibm.com/support/knowledgecenter/SSERQD) (<https://www.ibm.com/support/knowledgecenter/SSERQD>) 中提供 ABO 联机产品文档，可通过标准 Web 浏览器进行查看。

PDF 文件具有有限的辅助功能支持。借助 PDF 文档，您可以使用可选字体放大、高对比显示设置，并且可仅使用键盘进行浏览。

为了让屏幕朗读器能够精确读出语法图、源代码示例以及包含句点或逗号 PICTURE 符号的文本，必须将屏幕朗读器设置为可读出所有标点符号。

辅助技术产品会使用 z/OS 中的用户界面。要获取具体的指导信息，请参阅用于访问 z/OS 界面的辅助技术产品的文档。

相关的辅助功能信息

除了标准的 IBM 帮助热线和支持 Web 站点，IBM 还建立了 TTY 电话服务，以供耳聋或有严重听力障碍的客户获取销售和支持服务：

如何发送您的意见

TTY 服务
800-IBM-3383 (800-426-3383)
(北美)

IBM 和辅助功能选项

有关 IBM 对辅助功能选项所作承诺的更多信息，请参阅 [IBM Accessibility\(www.ibm.com/able\)](http://www.ibm.com/able)。

第 1 章 概述

IBM Automatic Binary Optimizer for z/OS (ABO) 可提高已编译的 IBM COBOL 程序的性能。ABO 无需调整源代码、源代码迁移或性能选项。它采用现代优化技术来瞄准于最新的 IBM Z 大型机（包括 IBM z15），从而提高 COBOL 应用程序性能。

ABO 的输入是 COBOL 程序模块。ABO 扫描程序模块以查找符合条件的 COBOL 程序进行优化。如果满足以下所有条件，那么程序就有资格进行 ABO 优化：

- 由符合条件的 IBM COBOL 编译器生成。
- 支持所有语言功能。
- 优化验证都已成功通过。

在验证输入程序后，ABO 会处理程序模块并生成优化程序模块以瞄准于最新的 IBM Z 大型机。

需要 z/OS V2.2 或更高版本来运行 ABO 和优化程序。可以在多个 z/Architecture® 级别运行优化程序，而无需对应用程序 JCL 进行任何更改。

优势

ABO 创建的优化程序通过利用最新的 IBM Z 大型机中的功能提升了性能，同时降低了处理开销并缩短了程序执行时间。

ABO 采用最先进的 COBOL 优化技术并生成相应代码以瞄准于可提供处理能力的最新部署系统，因此能够提高性能。

早期 COBOL 编译器仅在 ARCH(0) 级别生成代码。使用 Automatic Binary Optimizer for z/OS 升级这些 ARCH(0) 级别 COBOL 应用程序，使其使用最新的 ARCH(13) z15、ARCH(12) z14、ARCH(11) z13s®/ z13® 和 ARCH(10) zEC12/zBC12 大型机。

针对早期 COBOL 程序使用 Automatic Binary Optimizer 可在硬件技术发展过程中实现多达 25 年的跳跃式发展，因而能够访问 z15、z14、z13s、z13、zEC12 和 zBC12 大型机上已有的 600 多条新的硬件指令。

将 ABO 与 Enterprise COBOL 一起使用

IBM Automatic Binary Optimizer for z/OS 和最新的 IBM Enterprise COBOL 编译器提供不同的服务但功能互补。本部分提供了有关何时使用 ABO 以及何时通过最新的 IBM Enterprise COBOL 编译器重新编译源代码以提高性能的注意事项。

要提高程序的性能，请选择以下某种方法：

- 使用 IBM Enterprise COBOL z/OS V5 或 V6 以重新编译程序源代码
- 使用 IBM Automatic Binary Optimizer for z/OS 优化不在重新编译计划中或者程序源代码不可用的已编译程序。

根据第 1 页的表 1，从它们中选择一个进行使用。

表 1. 比较优化器与编译器的用例		
用例	IBM Automatic Binary Optimizer for z/OS	IBM Enterprise COBOL 编译器
显著提高性能，无需调整源代码、迁移或选项	√	
瞄准于多个硬件级别的 z/OS V2.2 和更高版本上的内置支持	√	

表 1. 比较优化器与编译器的用例 (续)		
用例	IBM Automatic Binary Optimizer for z/OS	IBM Enterprise COBOL 编译器
互操作性和传统兼容性（例如，PDS 输入/输出，与 OS/VS COBOL 和 VS COBOL II NORES 互操作）	√	
无需降级 ARCH 设置即可适用于灾难恢复机器	√	
新 COBOL 应用程序开发或使用新的 COBOL 功能		√
维护现有 COBOL 程序		√
最大限度提高性能（需要调整源代码、迁移和选项）		√

第 2 章 系统需求

受支持的操作系统

IBM Automatic Binary Optimizer for z/OS 也可在以下操作系统上运行：

- z/OS V2.4
- z/OS V2.3
- z/OS V2.2

对于 z/OS 2.4 之前的操作系统级别，ABO 需要在安装和运行 Automatic Binary Optimizer for z/OS 的系统上应用某些 PTF。在将运行 ABO 生成的优化模块的系统上需要其他 PTF，即使未在这些系统上安装 ABO。在运行 ABO 或 ABO 生成的模块的系统上，需要 APAR/PTF (OA47689/UA90982)（仅适用于 z/OS 2.2）。

在运行 ABO 的系统上需要以下 PTF：

- z/OS V2.3
 - OA55985/UA97356（绑定程序）
- z/OS V2.2
 - OA47829/UA78084（绑定程序）
 - OA50640/UA82866（绑定程序）
 - OA47689/UA90982（IEFOPZxx SYS1.PARMLIB 支持）
 - OA55985/UA97372（绑定程序）

在运行 ABO 优化模块的系统上需要以下 PTF：

- z/OS V2.3
 - PH14705/UI64417（Language Environment Automatic Binary Optimizer 运行时引擎）
- z/OS V2.2
 - PI51546/UI33445 (Language Environment)
 - PI51802/UI32944（Language Environment CICS® 系统定义样本更新）
 - OA47689/UA90982（IEFOPZxx SYS1.PARMLIB 支持）
- PH14705/UI64419（Language Environment Automatic Binary Optimizer 运行时引擎）

如果将使用相同的系统来运行 ABO 和运行 ABO 优化模块，那么必须在此系统上安装以上针对每个 z/OS 版本列出的所有 PTF。

可用于 ABO 的可选程序：

- [Application Delivery Foundation for z Systems V3.1](#)
 - Developer for z Systems Enterprise Edition V14.1
 - Debug for z Systems V14.1
 - Fault Analyzer for z/OS V14.1
 - Application Performance Analyzer for z/OS V14.1

强烈建议安装最新 IBM Automatic Binary Optimizer 或 IBM Automatic Binary Optimizer Trial PTF。请参阅 [修订列表和新功能部件页面](#)。

目标硬件级别

IBM Automatic Binary Optimizer for z/OS 可为最新的 IBM Z 服务器生成程序模块。

Automatic Binary Optimizer for z/OS 使用与 COBOL 编译器相同的硬件编号方案。第 4 页的表 2 列出了 IBM Automatic Binary Optimizer for z/OS V2.1 支持的硬件级别。您可以使用 ARCH 选项指定希望 ABO 生成的模块所针对的硬件级别。

表 2. 受支持的硬件级别	
硬件级别	描述
10	在 z/Architecture 方式下，生成使用 2827-xxxx (IBM zEnterprise® EC12) 和 2828-xxxx (IBM zEnterprise BC12) 模型上可用指令的代码。 特别是，这些 10 级机器及其下一代产品添加了能够支持以下工具的指令： · Execution-hint 工具 · Load-and-trap 工具 · Miscellaneous-instructions-extension 工具 · Transactional-execution 工具 · 增强型十进制浮点工具（可以提高分区十进制数据项与十进制浮点数据项之间的转换效率）。编译器无需将分区十进制数据项转换为压缩十进制数据项来执行算术运算，而是直接将分区十进制数据项转换为十进制浮点数据项，并在计算完成后重新转换为分区十进制数据项。
11	在 z/Architecture 方式下，生成使用 2964-xxxx (IBM z13®) 和 2965-xxx (IBM z13s) 模型上可用指令的代码。 特别是，这些 11 级机器及其下一代产品添加了能够支持以下工具的指令： · 增强型十进制浮点工具（可以提高压缩十进制数据项与十进制浮点中间结果数据项之间的转换效率）
12	在 z/Architecture 方式下，生成使用 3906-xxx (IBM z14) 和 3907-xxx (IBM z14 ZR1) 模型上可用指令的代码。 特别是，这些 12 级机器及其下一代产品添加了能够支持向量压缩十进制工具的指令，该工具可通过在向量寄存器（而不是在内存）中存储中间结果来加速压缩十进制计算。
13	在 z/Architecture 方式下，生成使用 8561-xxx (IBM z15) 模型上可用指令的代码。 特别是，这些 13 级机器及其下一代产品添加了受以下工具支持的指令： · 向量压缩十进制增强工具 · 向量增强工具 2 · 杂项指令扩展工具 3 · 对齐向量装入/存储提示
注：ABO 可在 z/OS 级别支持的任何系统上运行。有关支持 z/OS V2.2 以及更高版本的 IBM Z 服务器的完整列表，请参阅 z/OS Server Support。	

第 3 章 COBOL 模块需求

IBM Automatic Binary Optimizer for z/OS 可优化来自绑定程序的程序模块输出和来自链接编辑器的装入模块输出。来自绑定程序的程序模块输出可以是程序对象或装入模块。链接编辑器生成的装入模块必须是绑定程序可接受的输入，以便 ABO 进行优化。

ABO 还能优化完全绑定或部分绑定的程序模块。部分绑定的模块是使用 CALL=NO 或 NCAL 选项绑定的模块，通常包含在链接库中。如果指定了 ALLOW=NOUNRESEXE 选项，那么 ABO 将不会优化部分绑定的程序模块。请参阅 [ALLOW 选项](#)，以获取更多详细信息。

ABO 不支持以下 COBOL 模块：

表 3. ABO 不支持的 COBOL 模块	
ABO 不支持的 COBOL 模块	发出的消息
绑定程序不处理的模块	BOZ4116I 后跟 BOZ1429U 例如，绑定程序将不处理具有含无效字符的 ESD 名称的装入模块。当 ABO 遇到具有无效 ESD 名称的模块时，ABO 将生成 BOZ4116I 消息，此消息包含绑定程序消息 IEW2512E 的文本。IEW2512E 的文本中包含无效 ESD 名称。
与 EDIT=NO 绑定的模块	BOZ1423S
已签名的模块	BOZ1424S
标记为不可执行的模块	BOZ1422S
包含来自预链接步骤的对象文件但未正确完成预链接步骤的模块	BOZ4116I 后跟 BOZ1419S 例如，如果未对所有对象文件执行预链接步骤，那么预链接步骤可能未正确完成。这种未正确完成的预链接步骤可能产生绑定程序和 ABO 都不会处理的装入模块。如果绑定程序不处理此模块，那么 ABO 将生成一条 BOZ4116I 消息，后跟一条 BOZ1419S 消息。如果模块包含来自多个预链接步骤的输出，那么预链接步骤也未正确完成。在此情况下，原始模块通常无法正常运行，并且 ABO 会生成同样无法正常运行的模块。

ABO 将扫描程序模块中的 CSECT，以查找符合优化条件的 CSECT。如果 CSECT 由符合条件的 COBOL 编译器生成且 ABO 支持原始 COBOL 程序中使用的所有 COBOL 功能，那么该 CSECT 符合 ABO 优化条件。

符合条件的编译器

IBM Automatic Binary Optimizer for z/OS V2.1 可优化由以下 COBOL 编译器生成的程序模块中的 CSECT：

- Enterprise COBOL for z/OS V4
- Enterprise COBOL for z/OS V3
- COBOL for OS/390® & VM V2
- COBOL for MVS™ & VM V1.2
- COBOL/370 V1.1
- VS COBOL II V1.4.0（仅限启用 LE 的模块）
- VS COBOL II V1.3.x（仅限启用 LE 的模块）

注: ABO 无法优化 CA-Optimizer 已处理的 COBOL 模块。对于这些类型的模块, 建议先使用 ABO 优化 COBOL 编译器创建的原始模块, 然后再由 CA-Optimizer 进行处理。

COBOL 语言功能和编译器选项支持

受支持的 COBOL 语言功能和编译器选项

IBM Automatic Binary Optimizer for z/OS 可以支持绝大多数的 COBOL 语言功能。

下面列出了 IBM Automatic Binary Optimizer for z/OS V2.1 中支持的关键 COBOL 功能。

- ARITH(EXTEND | COMPAT)
- CICS
- CICS HANDLE ABEND
- CICS HANDLE AID
- CICS 语言转换程序生成的 SERVICE LABEL 语句
- CMPR2
- DB2[®]
- DLL
- ENTRY
- IMS
- I/O 和调试声明式
- NOOPTIMIZE 和 OPTIMIZE(STD | FULL)
- NUMPROC(NOPFD | PFD | MIG)
- 程序分段¹
- RECURSIVE
- RENT 和 NORENT
- SORT 和 MERGE
- SQL
- SSRANGE
- TEST²
- THREAD
- TRUNC(STD | BIN | OPT)
- 用户编写的 SERVICE LABEL 语句
- XML

不受支持的 COBOL 语言功能和编译器选项

IBM Automatic Binary Optimizer for z/OS V2.1 不会优化使用以下 COBOL 功能的程序模块:

- LABEL 声明式中使用的 ACCEPT FROM SYSIPT

¹ 支持大多数程序分段情况。其余不受支持的情况是当 ABO 处理的 CSECT 的源包含独立段、更改的 GO TO 语句和 GO TO DEPENDING ON 语句时。在这种情况下, 将发出消息 B0Z1455W: unsupported feature "Program Complexity 176" found, 并跳过 CSECT。

² 虽然通过 TEST 和任何子选项编译的程序可以通过 ABO 进行优化, 但 LE 不会为 ABO 生成的模块生成格式化的变量转储。LE 将改为生成此消息: <prog> was not compiled with the SYM suboption of the TEST. A formatted variable dump cannot be produced。

- CLASS
- LABEL 声明式中使用的 DISPLAY UPON SYSLST
- LABEL 声明式中使用的 DISPLAY UPON SYSPCH
- ENTER
- INVOKE
- 基于 Java 的面向对象 (OO) 语法
- RERUN

处理不符合条件的 CSECT

虽然 IBM Automatic Binary Optimizer for z/OS (ABO) 仅优化由符合条件的编译器部分中列出的编译器生成的 CSECT，但是 ABO 允许包含来自其他 COBOL 编译器和语言的 CSECT 的模块。

在开始优化之前，ABO 会检查每个 CSECT。如果满足下表中的任何条件，那么 CSECT 不符合优化条件，将发出一条消息并且将跳过该 CSECT。

表 4. 不符合条件的 CSECT 和发出的消息	
不符合条件的 CSECT	发出的消息
CSECT 名称与指定的 CSECT 过滤表达式不匹配。请参阅 CSECT 。	BOZ4113I
使用 COBOL 以外的语言生成 CSECT。例如，使用 HLASM、C/C++ 或 PL/I。	不发出任何特定消息 ³
CSECT 不是由以下某个符合条件的 COBOL 编译器进行编译。	BOZ1455W
CSECT 是由一个符合条件的 COBOL 编译器进行编译，但是其包含不受支持的 COBOL 功能部件汇总 中列出的一条或多条不受支持的 COBOL 语句。	BOZ1455W
CSECT 对于 ABO 太过复杂以致无法安全地优化和生成可正常运行的代码。	BOZ1455W
CSECT 包含任何意外数据或代码。这可包括但不限于任何缺失、损坏或其他错误字符串，ABO 依靠此类字符串来正确理解 CSECT 内容并执行正确的优化。	BOZ1455W
CSECT 先前已由 ABO 进行优化。	BOZ1455W

如果在模块中找到至少一个符合优化条件的 CSECT，那么会随优化的 CSECT 一起将任何不符合条件的 CSECT 原样复制到目标模块。

³ 优化器选项 SCAN=Y 可用于确定模块中存在的 CSECT 类型。

第 4 章 安装并验证安装

安装 IBM Automatic Binary Optimizer for z/OS

此产品随附的“程序目录”中包含有关安装 IBM Automatic Binary Optimizer for z/OS 的所有信息。

强烈建议同时安装最新的 IBM Automatic Binary Optimizer for z/OS PTF。请参阅[修订列表和新功能部件页面](#)，以获取 IBM Automatic Binary Optimizer for z/OS PTF 及 APAR 的列表。

相关参考

第 77 页的『IBM Automatic Binary Optimizer for z/OS 出版物』

使用“安装验证程序”(IVP) 验证安装

在完成 ABO 的 SMP/E 安装后，使用 ABO 安装验证程序 (IVP) BOZJIVP 来验证 ABO 是否正确安装且正常运行。

BOZJIVP 概述

ABO 安装验证程序 (IVP) BOZJIVP 位于 ABO 样本库 HLQ.SBOZJCL 中，其中，HLQ 是用于 ABO SMP/E 安装中的目标库的前缀。

在计划使用 ABO 的任何系统上以及将运行 ABO 生成的优化模块的任何系统上运行 IVP。

注：ABO 可在最低 z/OS 级别支持的任何硬件级别上运行，但是 ABO 生成的优化模块只能在 zEC12、zBC12、z13、z13s、z14、z14 ZR1 和 z15 系统上运行。请参阅第 4 页的『目标硬件级别』，以获取更多信息。在检查 IVP 结果时，请记住这些最低硬件需求。

使用 BOZJIVP

为在所选系统上继续执行 IVP 流程，请根据附带的 JCL 描述来编辑 BOZJIVP，然后进行提交。

此作业包含以下步骤：

1. LKED - 使用同一样本库中的对象 BOZOBJ1 作为输入来链接编辑原始 COBOL 程序。

注：已使用 Enterprise COBOL for z/OS V4R2 (OPT(STD) 选项生效) 编译了 BOZOBJ1 程序。为方便起见，程序源代码示例 BOZSRC1 也位于同一库中。

2. GOBEFORE - 运行原始程序。
3. VERIFY1 - 验证 z/OS 版本是否能够运行 ABO。
4. OPTIMIZE - 使用 ABO 优化原始程序。
5. VERIFY2 - 验证 IBM z 服务器类型是否能够运行 ABO 优化模块。
6. GOAFTER - 运行原始 COBOL 程序的优化版本。
7. REPORT - 报告 IVP 结果。

结果

在 IVP 成功运行时，对于每个前述步骤，您将收到返回码 0 或 4。在 REPORT 步骤完成时，将在 SYSTSRPT 输出文件和 JESMSGLG JOBLOG 中提供报告。

以下示例显示了 SYSTSRPT 输出文件中的样本 IVP 报告：

```
*** The original program start time is: 10:42:22.72
*** The original program end time is: 10:44:10.71
```

```

***
***      Optimization successful!
***
***
*** The optimized program start time is: 10:44:11.41
*** The optimized program end time is: 10:44:15.63
***
*** The elapsed time is reduced by 103.77 sec
***
***
***      Installation verification successful!
***
***

```

以下示例显示了样本 JESMSGLG JOBLOG。请注意，在 JOBLOG 和控制台中都提供 Installation verification successful! 消息。

```

10.42.22 JOB07227  HTRT01I                                CPU (Total)
Elapsed
10.42.22 JOB07227  HTRT02I  Jobname  Stepname  ProcStep  RC    I/O hh:mm:ss.th
hh:mm:ss.th
10.42.22 JOB07227  HTRT03I  BOZIVP    LKED                00    176      00.01
00.10
10.44.10 JOB07227  HTRT03I  BOZIVP    GOBEFORE              00    192      01:47.49
01:48.14
10.44.11 JOB07227  HTRT03I  BOZIVP    OPTIMIZE              00  13457      00.06
00.54
10.44.15 JOB07227  HTRT03I  BOZIVP    GOAFTER              00    205      04.21
04.37
10.44.15 JOB07227  +***      Installation verification successful!
***
10.44.15 JOB07227  HTRT03I  BOZIVP    REPORT                00     64      00.01
00.03

```

如果 VERIFY1 步骤失败，那么您将在 JOBLOG 和控制台中看到以下消息：

```

z/OS version: xx.xx is not a supported z/OS version to run AB0.

```

如果 VERIFY2 步骤失败，那么您将在 JOBLOG 和控制台中看到以下消息：

```

IBM z server: type (name)
is not a supported hardware level to run AB0 optimized modules.

```

例如：

```

IBM z server: 2817 (zEnterprise 196)
is not a supported hardware level to run AB0 optimized modules.

```

如果 OPTIMIZE 步骤失败，请验证此步骤日志文件中的消息，以查看可能缺少哪个系统或 Language Environment 组件。修复该问题，然后再次运行 BOZJIVP 作业。

如果 GOAFTER 步骤失败，请验证可能缺少哪个语言环境 PTF。如果未安装第 3 页的『受支持的操作系统』中列出的一个或多个“Language Environment Automatic Binary Optimizer 运行时引擎”PTF，那么可能发生 OC1 异常终止。如果安装了“Language Environment Automatic Binary Optimizer 运行时引擎”PTF，但它不是程序目录中列出的最新 PTF，那么将发生 U4038 异常终止，并且将显示下列其中一条消息：

```

IGZ0153S Program BOZSRC1 was compiled with a level of the compiler that requires service to be
installed on Language Environment.
IGZ0355S Program BOZSRC1 was optimized with a level of the Automatic Binary Optimizer that
requires service to be installed on Language Environment.

```

z/OS 2.2 和 2.3 上的“Language Environment Automatic Binary Optimizer 运行时引擎”PTF 将导致发出第一条消息，z/OS 2.4 上的 PTF 将导致发出第二条消息。

如果并非异常终止，而是 GOAFTER 步骤失败且返回非零返回码，那么该返回码与缺少的 Language Environment PTF 相对应，如下所示：

表 5. 返回码和相应缺少的 LE PTF		
返回码	z/OS 2.2	z/OS 2.3
17	PI84563	PI84561

安装必需的 PTF，然后再次运行 BOZJIVP 作业。

如果尝试在 ABO 不支持的系统上运行 ABO 生成的模块，那么也将发生 0C1 异常终止。请参阅第 4 页的『[目标硬件级别](#)』，以了解受支持的系统。

第 5 章 优化模块

要使用 Automatic Binary Optimizer for z/OS，请编写用于优化流程的 JCL。

使用 EXEC 语句调用 ABO

在 JCL 中使用 EXEC 作业控制语句来调用 ABO。该 EXEC 语句如下所示：

```
//OPT      EXEC PGM=BOZOPT
```

必需的 DD 语句

优化流程要求您为优化流程中的某些特殊用途指定数据集。您可以在 DD 语句中使用必需的 DD 名称定义这些数据集。第 13 页的表 6 中显示了 ABO 所使用的 DD 名称及其特征。

指定优化器伪指令 BOPT 或 IEFOPZ

使用 BOPT 或 IEFOPZ 来指引 ABO。您可以在 SYSIN DD 中包含一个或多个 BOPT 或 IEFOPZ 伪指令。有关详细信息，请参阅第 14 页的『BOPT』和第 15 页的『IEFOPZ』。

必需的 DD 语句

下表显示了 Automatic Binary Optimizer for z/OS 使用的 DD 名称。

表 6. 用于二进制文件优化的 DD 名称			
DD 名称	类型	是否必需	描述
STEPLIB	输入	是	指定包含 ABO 和从属 Language Environment® 运行时数据集的数据集的名称。
SYSIN	输入	是	指定包含优化器伪指令 BOPT 和 IEFOPZ 以及优化器选项的文件的位置。为方便起见，您可以使用 DD * 在 JCL 中指定流内文件。
OPTLOG	输出	是	指定将优化摘要信息（例如，要优化的内容以及优化的二进制文件的位置）写入到此 DD 中。SCAN 输出也会写入到此处。
SYSPRINT	输出	否（如果指定 LIST 选项） 是（如果未指定 LIST 选项）	指定列表转换的缺省位置。另请参阅第 45 页的『SYSPRINT DD 和 LIST 选项』。
OPTERR	输出	否	指定在异常情况下将优化诊断信息写入到此 DD 中。
CEEDUMP	输出	否	指定在异常情况下将优化转储信息写入到此 DD 中。

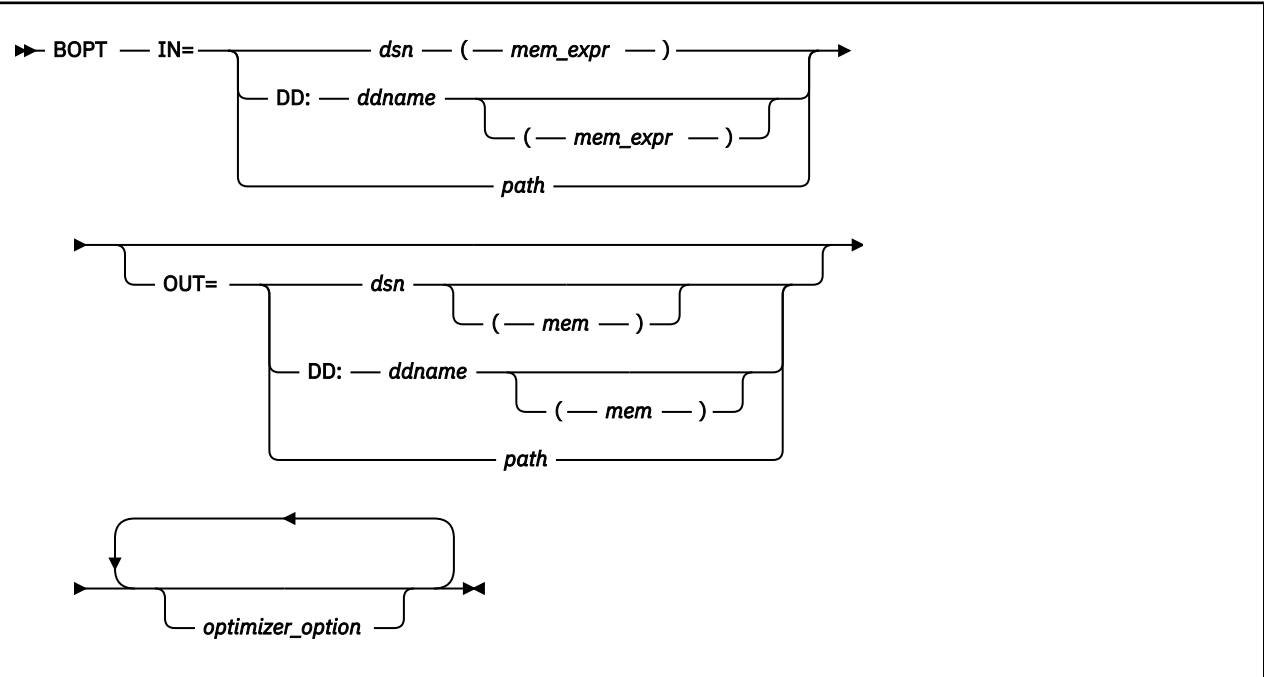
表 6. 用于二进制文件优化的 DD 名称 (续)			
DD 名称	类型	是否必需	描述
CEEOPTS	输入	否（如果想要用英语显示消息）。 是（如果想要用日语显示消息）。	指定要用于消息的语言。另请参阅第 29 页的『指定要用于 ABO 消息的语言』。

优化器伪指令

可使用 BOPT 或 IEFOPZ 来指引 ABO。

BOPT

您可以使用 BOPT 优化器伪指令来生成基于明确的输入和输出说明符的优化模块。



IN
指定一个要优化的输入模块，或者指定多个输入模块（当 *mem_expr* 说明符中指定了通配符时）。

OUT
指定一个输出模块，或者指定一个 PDS(E) 来表示一个或多个输出模块（在省略 *mem* 说明符时）。

DD:ddname
指定 DD 名称。

dsn
这是必须包含高级限定符的数据集名称。

mem
这是数据集成员名称。

mem_expr
这是可能包含表达式的数据集成员名称。仅处理其名称字符串与表达式匹配的成员。匹配不区分大小写。

正则表达式接受以下符号：

与任何字符串相匹配。

?

与任何字符相匹配。

|

可用作多个表达式的分隔符，对于多个表达式而言，其中任何表达式与字符串匹配均计为一个匹配。

!

对其后的整个表达式求反。例如：

· 要跳过名为 MEMA 的单个成员：

```
IN=DD:SYSBIN(!MEMA)
```

· 要跳过名称以 MEMB 开头的所有成员：

```
IN=DD:SYSBIN(!MEMB*)
```

· 要跳过名为 SUB1 和 SUB2 的成员：

```
IN=DD:SYSBIN(!SUB1|SUB2)
```

path

这是以斜杠 (/) 开头的完整 HFS 路径，例如，/home/user1/a.out.opt。

optimizer_option

这是优化器选项。有关可指定的优化器选项的列表，请参阅第 17 页的『优化器选项』。

注：

1. 在 SCAN 优化器选项设置为 Y 时，BOPT 上的 OUT 选项为可选选项。
2. 在 IN 选项上指定 *mem_expr* 时，将选中与 *mem_expr* 匹配的所有成员来进行优化。在指定 *mem_expr* 时，请勿在 OUT 选项上包含 *mem* 说明符。
3. 如果 OUT 选项上没有 *mem* 说明符，那么 OUT 的成员名称确定为与 IN 选项上的 *mem* 或 *mem_expr* 说明符匹配的名称。
4. IN 说明符、OUT 说明符和优化器选项可采用任意顺序。

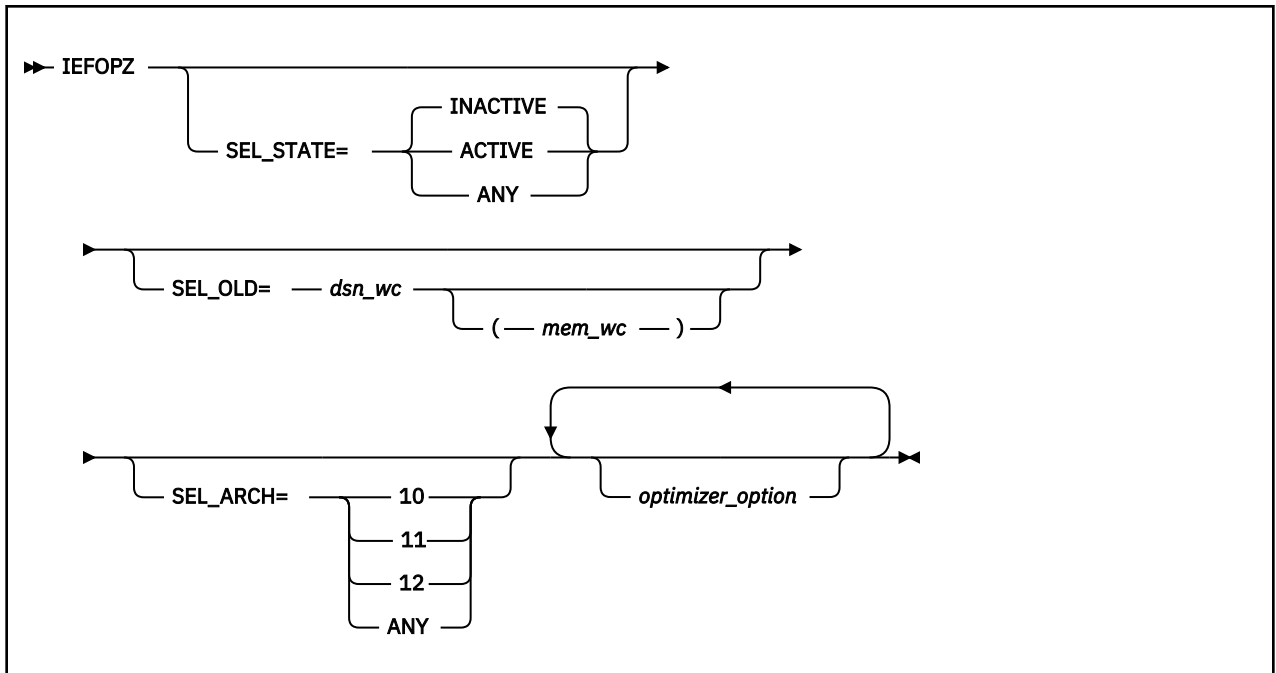
有关 BOPT 伪指令的示例，请参阅第 25 页的『JCL 示例』。对于使用 BOPT 优化器伪指令的样本场景，请参阅第 47 页的『场景 1：使用静态部署的优化流程』和第 49 页的『场景 3：使用混合方法的优化流程』。

IEFOPZ

您可以使用 IEFOPZ 优化器伪指令来基于 IEFOPZ 配置生成优化模块。

注：仅在 z/OS V2.2 和更高版本上支持 IEFOPZ。在 z/OS 2.2 上，必须应用 APAR/PTF OA47689/UA90982。

有关 IEFOPZ 配置的信息，请参阅第 48 页的『场景 2：使用动态部署的优化流程』中的步骤 2。



SEL_STATE

指示优化器优化与指定状态匹配的映射。

ANY

指示优化器优化已标记为 ACTIVE 或 INACTIVE 的映射。

ACTIVE

指示优化器仅优化已标记为 ACTIVE 的映射。

INACTIVE

指示优化器仅优化已标记为 INACTIVE 的映射。

SEL_OLD

将优化范围限制为具有与指定选择器匹配的 OLD 数据集的映射。

dsn_wc

这是可能包含通配符（使用星号 (*) 符号）的数据集名称。例如，*IN*.LOAD。

mem_wc

这是可能包含通配符（使用星号 (*) 符号）的数据集成员名称。例如，M*。

SEL_ARCH

指示优化器优化用指定体系结构标记的映射。

10

指示优化器仅优化已标记为 ARCH(10) 的映射。

11

指示优化器仅优化已标记为 ARCH(11) 的映射。

12

指示优化器仅优化已标记为 ARCH(12) 的映射。

ANY

指示优化器优化已标记为 ARCH(10)、ARCH(11) 或 ARCH(12) 的映射。

optimizer_option

这是优化器选项。有关可指定的优化器选项的列表，请参阅第 17 页的『优化器选项』。

注:

1. 映射是指 IEFOPZ 配置中的 OLD 模块与 NEW 模块的关联。
2. 缺省情况下，将在由 NEW 数据集以及 IncludeMembers 和 ExcludeMembers 配置说明符确定的体系结构级别，优化 IEFOPZ 配置的 OLD 数据集集中的所有 INACTIVE 模块。SEL_OLD= 和 SEL_ARCH= 是可用于将优化或扫描范围限制为其中一部分模块的选择器。

SEL_STATE= 是可用于将优化或扫描更改为 ACTIVE 模块或 ANY 状态模块的选择器。当 ACTIVE 模块扫描不造成风险时，ACTIVE 模块优化可能导致使用 ACTIVE 模块的程序出现问题。应谨慎使用选择器 SEL_STATE=ACTIVE 或 SEL_STATE=ANY 并应谨慎执行优化（与扫描相反）。

3. IN 说明符、OUT 说明符和优化器选项可采用任意顺序。

有关 IEFOPZ 伪指令的示例，请参阅第 25 页的『JCL 示例』。有关使用 IEFOPZ 优化器伪指令的样本场景，请参阅第 48 页的『场景 2：使用动态部署的优化流程』。

优化器选项

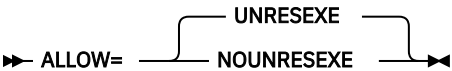
优化器选项是同时适用于 BOPT 和 IEFOPZ 优化器伪指令的 Automatic Binary Optimizer for z/OS 选项。优化器选项可放在位于一行或多行上的 SYSIN 文件的开头，或者放在 BOPT 或 IEFOPZ 优化器伪指令上。全局选项是在不包含 BOPT 或 IEFOPZ 优化器伪指令的行上指定的优化器选项。全局选项的值被称为选项的全局设置。在包含一条优化器伪指令的特定行上指定优化器选项时，该选项的值仅应用于该优化器伪指令，然后针对后续语句还原为全局设置。第 17 页的表 7 汇总了同时适用于 BOPT 和 IEFOPZ 的优化器选项。

表 7. 优化器选项		
选项	缺省值	描述
第 17 页的『ALLOW』	ALLOW=UNRESEXE	控制 ABO 将接受的程序模块的类型。
第 18 页的『ARCH』	ARCH=10	指定目标硬件级别。
第 18 页的『CSECT』	如果未指定 CSECT 选项，ABO 将处理所有符合条件的 CSECT。	允许用户将处理限制为一个或多个 CSECT。
第 21 页的『LIST』	如果未指定 LIST 选项，那么列表转换会放在 SYSPRINT DD 指定的位置中。	指定已生成的列表转换的位置。
第 21 页的『LOG』	如果未指定 LOG 选项，那么 ABO 将不会生成成员级别的日志文件。	指定要额外生成的成员级别日志文件的位置。
第 22 页的『REPLACE』	REPLACE=Y	控制是否写入到输出模块。
第 23 页的『SCAN』	SCAN=N	控制是优化还是扫描程序模块。
注: 可以在全局级别以及在 BOPT 伪指令上指定 ARCH 选项。不能在 IEFOPZ 伪指令上指定该选项。对于 IEFOPZ，可使用 SEL_ARCH 选项来选择 IEFOPZ 配置中与 SEL_ARCH 值匹配的部分以进行优化。		

ALLOW

用途

ALLOW 选项控制 Automatic Binary Optimizer for z/OS 将接受的程序模块的类型。



缺省值

ALLOW=UNRESEXE

用途

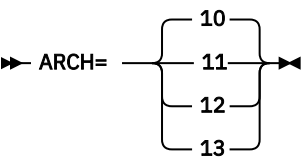
在指定 ALLOW=UNRESEXE 时，ABO 接受完全绑定的模块或部分绑定的模块。优化器接受的部分绑定的程序模块的唯一类型是使用 CALL=NO 或 NCAL 绑定程序选项链接的类型。如果完全绑定输入模块，那么将完全绑定优化的输出模块。如果部分绑定输入模块，那么将部分绑定优化模块。

在指定 ALLOW=NOUNRESEXE 时，ABO 仅接受完全绑定的程序模块（程序对象或装入模块）并且允许生成完全绑定的程序模块。如果处理部分绑定的模块，那么在指定 ALLOW=NOUNRESEXE 时，将发出消息 BOZ1494S。

ARCH

用途

ARCH 选项指定目标硬件级别。



缺省值

ARCH=10

用途

使用 ARCH 选项指定 ABO 生成的优化模块将针对的硬件级别。

使用较低 ARCH 设置生成的优化模块将在较高级别的 ARCH 系统上运行。但是，使用较高 ARCH 设置生成的优化模块无法在较低级别的 ARCH 系统上运行。

ARCH 设置	只能在 IBM Z 大型机级别上运行
ARCH=10	zEC12、zBC12、z13、z13s、z14、z14 ZR1 和 z15
ARCH=11	z13、z13s、z14、z14 ZR1 和 z15
ARCH=12	z14、z14 ZR1 和 z15
ARCH=13	z15

如果尝试了无效组合，那么该程序很可能会终止，并返回以下运行时 LE 消息：

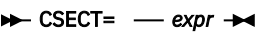
```
CEE3201S The system detected an operation exception (System Completion Code=0C1)
```

有关这些 ARCH 级别的详细信息，请参阅第 4 页的『目标硬件级别』。

CSECT

用途

CSECT 选项允许您将处理限制为 0 个或更多个 CSECT。



缺省值

缺省情况下，如果未指定 CSECT 选项，那么 ABO 将处理所有符合条件的 CSECT。

参数

expr

想要处理的 CSECT 的正则表达式。仅处理其名称字符串与表达式匹配的 CSECT。匹配不区分大小写。

用途

正则表达式接受以下符号：

与任何字符串相匹配。

?

与任何字符相匹配。

|

可用作多个表达式的分隔符，对于多个表达式而言，其中任何表达式与字符串匹配均计为一个匹配。

!

对其后的整个表达式求反。例如：

· 要跳过名为 PROGA 的单个 CSECT：

```
CSECT=!PROGA
```

· 要跳过名称以 PROGB 开头的所有 CSECT：

```
CSECT=!PROGB*
```

· 要跳过名为 SUB1 和 SUB2 的 CSECT：

```
CSECT=!SUB1|SUB2
```

注：

- 正则表达式中不允许包含空格和方括号。
- 表达式必须与整个字符串匹配。部分匹配不计为匹配。这意味着表达式 **M*2** 与字符串 **MA2** 匹配，但与字符串 **MA2A** 不匹配。
- 由于 EBCDIC 代码页中针对正则表达式中可用的特殊字符使用不同的字符编码，所以只应在以下代码页中使用此选项：
 - IBM-1047
 - IBM-37/1140
 - IBM-285/1146
 - IBM-924

示例 1

在以下示例中，不处理与通配符过滤器不匹配的 CSECT。

JCL 命令

```
BOPT IN=DD:SYSBIN(*) OUT=DD:SYSBOUT CSECT=SUB*1*
```

输出（在 OPTLOG 中）

```
...
10:46:04      Processing CSECT filter expression 'SUB*1*' on member CALLLITT
10:46:04 CSECT CALLLIT was excluded by filter - skip
10:46:04      Processing CSECT SUB01L00, in member CALLLITT
10:46:04          Optimizing CSECT SUB01L00 for zEC12
10:46:04          Succeeded in optimizing SUB01L00
10:46:04          Generating listing transform into DD:SYSPRINT
10:46:04 CSECT SUB02L00 was excluded by filter - skip
10:46:04 CSECT SUB03L00 was excluded by filter - skip
10:46:04 CSECT SUB04L00 was excluded by filter - skip
10:46:04 CSECT SUB05L00 was excluded by filter - skip
10:46:04 CSECT SUB06L00 was excluded by filter - skip
10:46:04 CSECT SUB07L00 was excluded by filter - skip
10:46:04 CSECT SUB08L00 was excluded by filter - skip
10:46:04 CSECT SUB09L00 was excluded by filter - skip
10:46:04      Processing CSECT SUB10L00, in member CALLLITT
10:46:04          Optimizing CSECT SUB10L00 for zEC12
10:46:04          Succeeded in optimizing SUB10L00
```

```
10:46:04      Generating listing transform into DD:SYSPRINT
10:46:04      Finished processing, processed 2 of 11 CSECTs in member CALLLITT
```

示例 2

以下示例显示了如何指定多个多表达式用于匹配。

JCL 命令

```
BOPT IN=DD:SYSBIN(*) OUT=DD:SYSBOUT CSECT=SUB01L00|SUB02L00
```

输出

```
...
10:49:13      Processing CSECT filter expression 'SUB01L00|SUB02L00' on member
CALLLITT
10:49:13      CSECT CALLLIT was excluded by filter - skip
10:49:13      Processing CSECT SUB01L00, in member CALLLITT
10:49:13      Optimizing CSECT SUB01L00 for zEC12
10:49:13      Succeeded in optimizing SUB01L00
10:49:13      Generating listing transform into DD:SYSPRINT
10:49:13      Processing CSECT SUB02L00, in member CALLLITT
10:49:13      Optimizing CSECT SUB02L00 for zEC12
10:49:13      Succeeded in optimizing SUB02L00
10:49:13      Generating listing transform into DD:SYSPRINT
10:49:13      CSECT SUB03L00 was excluded by filter - skip
10:49:13      CSECT SUB04L00 was excluded by filter - skip
10:49:13      CSECT SUB05L00 was excluded by filter - skip
10:49:13      CSECT SUB06L00 was excluded by filter - skip
10:49:13      CSECT SUB07L00 was excluded by filter - skip
10:49:13      CSECT SUB08L00 was excluded by filter - skip
10:49:13      CSECT SUB09L00 was excluded by filter - skip
10:49:13      CSECT SUB10L00 was excluded by filter - skip
10:49:13      Finished processing, processed 2 of 11 CSECTs in member CALLLITT
```

示例 3

在以下示例中，数据集 HLQ.IN.LOAD 中的 MEM1 具有名为 A1 和 B1 的两个 CSECT。要将 ABO 处理限制为仅 A1，请使用 CSECT=A* 过滤器，如下所示：

JCL 命令

```
BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1) CSECT=A*
```

在添加此 CSECT=A* 过滤器时，OPTLOG 如下所示：

输出

```
...
17:46:04      Processing CSECT filter expression 'A*' on member MEM1
17:46:04      Processing CSECT A1, in member MEM1
17:46:04      Optimizing CSECT A1 for zEC12
17:46:04      Succeeded in optimizing A1
17:46:04      Generating listing transform into DD:SYSPRINT
17:46:04      CSECT B1 was excluded by filter - skip
17:46:04      Finished processing, processed 1 of 2 CSECTs in member MEM1
...
```

或者，CSECT=A* 可指定为全局选项，从而应用于所有后续 BOPT 和 IEFOPZ 伪指令，除非使用特定伪指令进行覆盖：

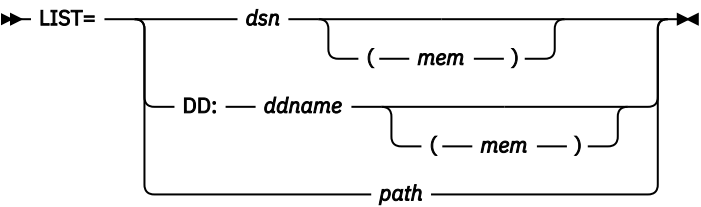
```
//SYSIN DD *
      CSECT=A*
      BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1A)
      BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1B) CSECT=B*
```

在处理后，成员 MEM1A 将包含优化的 CSECT A1 和原始 CSECT B1，并且成员 MEM1B 将包含优化的 CSECT B1 和原始 CSECT A1。

LIST

用途

LIST 选项指定已生成的列表转换的位置。



缺省值

缺省情况下，如果未指定 LIST 选项，那么列表转换会放在 SYSPRINT DD 指定的位置中。

参数

dsn

这是必须包含高级限定符的数据集名称。

mem

这是数据集成员名称。

DD:ddname

指定 DD 名称。

path

这是以斜杠 (/) 开头的完整 HFS 路径，例如， /home/user1/a.list。

用途

LIST 选项的目标可以为以下项之一：

- 顺序数据集，或者 PDSE（而非 PDS）的成员。可以向此单个顺序数据集添加多个 CSECT 优化的输出。
- PDS 或 PDSE。在优化 CSECT 时，特定于该 CSECT 的列表转换会放在 PDS 或 PDSE 的成员中，该成员名称基于 CSECT 名称（大写且截断为 8 个字符）。将覆盖该成员的内容（如果有），即使原来的内容是由优化器在先前调用中生成的，也是如此。
- HFS 路径。可以向此 HFS 文件添加多个 CSECT 优化的输出。

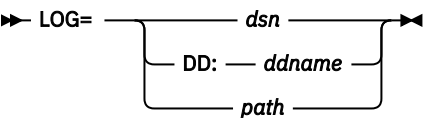
相关信息

[SYSPRINT DD](#)。

LOG

用途

LOG 选项指定要额外生成的成员级别日志文件的位置。



缺省值

缺省情况下，如果未指定 LOG 选项，那么 ABO 将不会生成成员级别日志文件。请注意，无论是否指定 LOG 选项，始终都会根据 OPTLOG DD 在相应位置中生成整个 ABO 调用的输出。

参数

dsn

这是必须包含高级限定符的数据集名称。

DD:ddname

指定 DD 名称。

path

这是以斜杠 (/) 开头的完整 HFS 路径。

用途

LOG 选项的目标可以为以下项之一：

- PDS 或 PDSE。在优化成员时，特定于该成员的日志输出将放入 PDS 或 PDSE 的某个成员中，此成员名称基于优化的成员名称（大写且截断为 8 个字符）。将覆盖该成员的内容（如果有），即使原来的内容是由 ABO 在先前调用中生成的也是如此。
- 指向目录的 HFS 路径。在优化成员时，特定于该成员的日志输出将放入指定 HFS 目录下的某个文件中，此文件名基于优化的成员名称（附加 .log）。将覆盖该成员的内容（如果有），即使原来的内容是由 ABO 在先前调用中生成的也是如此。

注：

1. LOG 选项的目标必须是未指定任何成员的 PDS 或 PDSE，或者是指向目录的 HFS 路径。如果 LOG 选项的目标指定为已指定成员的 PDS 或 PDSE、连续数据集或 HFS 文件，那么将发出一条错误消息。
2. LOG 选项的目标数据集应位于第 57 页的表 12 中 OPTLOG 的建议分配参数之后。

示例 1

```
BOPT IN=DD:SYSBIN(*) OUT=DD:SYSBOUT LOG=HLQ.LOG.OUT
```

示例 2

```
BOPT IN=DD:SYSBIN(*) OUT=DD:SYSBOUT LOG=DD:LOG
```

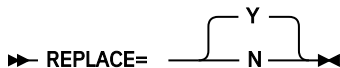
示例 3

```
BOPT IN=DD:SYSBIN(*) OUT=DD:SYSBOUT LOG=/home/user1/logdir
```

有关更多信息，请参阅第 37 页的『日志文件』。

REPLACE**用途**

控制是否写入到输出模块。

**缺省值**

REPLACE=Y

用途

当指定 REPLACE=Y 和 SCAN=N 时，不论是否存在输出模块，都会将优化模块写入到输出模块中。

- 如果不存在输出模块，那么将创建并写入到输出模块。
- 如果已存在输出模块，那么将覆盖其内容。

但是，如果不存在符合优化条件的 COBOL CSECT，那么不会向输出模块写入任何内容。

当指定 REPLACE=N 时，如果已存在输出模块，那么将绕过输入模块的优化或扫描；并且不会向输出模块写入任何内容。

您可以使用 REPLACE=N 来绕过对已优化模块的优化。例如，如果在优化了二进制文件后向原始数据集添加了新成员并且只想优化新成员，那么可以将成员通配符与 REPLACE=N 结合使用，如下所示：

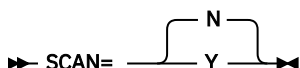
```
BOPT IN=HLQ.IN.LOAD(*) OUT=HLQ.OUT.LOAD REPLACE=N
```

在某些情况下，由于超时或者其他异常情况而导致了优化器提前终止。要解决此问题，您可以通过一系列作业以递增方式构建优化模块，而不必花时间重复执行先前作业中已完成的优化。或者，在这些情况下您可以使用 REPLACE=N。

SCAN

用途

SCAN 选项指示 ABO 是优化还是扫描程序模块



缺省值

SCAN=N

用途

在 SCAN=N 生效时，ABO 在输入程序模块上执行优化。

在 SCAN=Y 生效时，ABO 扫描输入程序模块，而非执行优化。将不生成任何输出模块。

REPLACE=N 选项将决定是否执行扫描：

- 如果在 BOPT 伪指令上指定 REPLACE=N 并且 OUT 选项上的输出模块已存在，那么将绕过对 IN 选项上模块的扫描。
- 如果在 IEFOPZ 伪指令上指定 REPLACE=N，那么在 NEW 数据集的成员已存在的情况下，将绕过对 OLD 数据集成员的扫描。

如果 BOPT 伪指令上不包含 OUT 选项，那么不论 REPLACE 选项的值如何，始终都会执行扫描。

在扫描方式下，优化器会检查输入和输出程序模块，并列出模块中的 CSECT。扫描输出将写入到 OPTLOG DD 中。

您可以使用 SCAN=Y 来测试 SYSIN 设置或者查看所显示的模块及其内容。如果由于所使用的原始编译器而使程序不符合优化条件，那么扫描输出中也会指明这一点。

有关更多信息，请参阅第 38 页的『针对扫描的日志文件』。

注释

通过以 (#) 字符开头来指定注释。

在使用 (#) 字符时，以下规则适用：

- 如果 SYSIN 文件某一行中的第一个非空字符为 (#) 字符，那么将忽略此行的其余部分。
- 如果 ABO 在 SYSIN 文件的输入行上发现 (#) 字符（其前面是一个空格），那么将忽略此行的其余部分。

示例 1

```
#BOPT IN=DD:SYSBIN OUT=SYSBOUT  
#BOPT IN=SYSBIN OUT=SYSBOUT
```

在示例 1 中，将注释掉这两个 BOPT 伪指令。在第一行上，# 字符位于第 1 列，将忽略此行的其余部分。在第二行上，第一个非空字符是 # 字符，将忽略此行的其余部分。

示例 2

```
# Optimizing all members of library  
# Note: we don't compile a member optimized earlier and found in the OUT dataset  
BOPT IN=SYSBIN(*) OUT=SYSBOUT REPLACE=N
```

示例 2 显示了如何向 SYSIN 文件添加两整行参考注释。

示例 3

```
BOPT IN=SYSBIN(*) OUT=SYSBOUT #
```

在示例 3 中，将忽略位于行尾的 # 字符。

示例 4

```
BOPT IN=SYSBIN(*) OUT=SYSBOUT #optimizing library files
```

示例 4 显示了如何在伪指令后添加注释。

行连续符

SYSIN 文件中的行连续符是通过 (+) 或 (-) 连续字符来指定，这些字符指示下一行应作为前一行的一部分来读取。虽然行连续符不是必需的，但可用于将较长的行拆分，以简化编辑或者便于其他可能存在行长度限制的工具使用。

(+) 或 (-) 连续字符必须为行上的最后一个非空格、非注释字符。使用连续字符时，以下规则适用：

- (-) 字符只能用于接续位于完整的选项、伪指令或说明符之后的行。行连续符从下一行的列 1 处开始。(-) 字符前面必须有一个或多个空格。
- (+) 字符可用于接续位于完整的选项、伪指令或说明符之后的行，也可位于选项、伪指令或说明符中间。行连续符从下一行上的首个非空格字符开始。接续完整的选项、伪指令或说明符时，(+) 字符前面必须有一个或多个空格。
- 位于行连续符前面的空格将包含在合并的行中。
- 注释字符 (#) 优先于行连续符。如果行连续符属于注释的一部分，那么它将作为此注释的一部分而被忽略，而非用于接续注释。

如果连续字符位于意外位置，那么可能会发出一条错误消息。

以下示例显示了使用 BOPT 优化器伪指令的 JCL。这些例子并非完整示例。它们旨在反映用户应该在 SYSIN 文件中指定的内容。对于基本 JCL 配置，请参阅第 57 页的『附录 A JCL 示例』。

示例 1

```
//SYSIN DD *  
ARCH=11 -  
ALLOW=UNRESEXE  
BOPT IN=HLQ.LOAD.APPXYZ1.ORIG(*) -  
OUT=HLQ.LOAD.APPXYZ1.ABO
```

ABO 按如下方式解释此示例：

```
//SYSIN DD *  
ARCH=11 ALLOW=UNRESEXE  
BOPT IN=HLQ.LOAD.APPXYZ1.ORIG(*) OUT=HLQ.LOAD.APPXYZ1.ABO
```

请注意，在所有情况下，位于连续字符前面的空格都将包含在合并的字符串中。

示例 2

```
//SYSIN DD *  
  
AR+  
CH=12 +  
  
ALLOW=UNRESEXE  
  
BOPT IN=HLQ.LOAD.APPXYZ1.OR+  
IG(*) -  
OUT=HLQ.LOAD.APPXYZ1.ABO
```

此示例说明了 (+) 字符可在下一行上任意期望的位置（首个非空格位置）接续，而非仅在列 1 处接续，还说明了如何使用 (+) 字符接续不完整的选项、伪指令或说明符。ABO 按如下方式解释此示例：

```
//SYSIN DD *  
ARCH=12 ALLOW=UNRESEXE  
BOPT IN=HLQ.LOAD.APPXYZ1.ORIG(*) OUT=HLQ.LOAD.APPXYZ1.ABO
```

示例 3

```
//SYSOPTF DD *  
# some comment + (1)
```

ARCH=12	+	#other comment	-	(2)
		ALLOW=UNRESEXE		(3)
BOPT IN=HLQ.LOAD.APPXYZ1.ORIG(*)		OUT=HLQ.LOAD.APPXYZ1.ABO		(4)

ABO 按如下方式处理此示例:

```
//SYSOPTF DD *
ARCH=12 ALLOW=UNRESEXE
BOPT IN=HLQ.LOAD.APPXYZ1.ORIG(*) OUT=HLQ.LOAD.APPXYZ1.ABO
```

将忽略行 1 上的 (+) 字符和行 2 上的 (-) 字符, 因为它们属于注释的一部分。行 2 上的 (+) 字符将作为行连续符来处理, 因为它是该行上最后一个非注释、非空格字符。

JCL 示例

在 Automatic Binary Optimizer for z/OS 中, 您可以使用这些示例中显示的作业控制语言 (JCL) 语句来处理已编译的 COBOL 模块。

使用 BOPT 指定优化

以下示例显示了使用 BOPT 优化器伪指令的 JCL。

此部分中的示例都不是完整的示例。它们旨在反映用户应该在 SYSIN 文件中指定的内容。对于基本 JCL 配置, 请参阅第 57 页的『附录 A JCL 示例』。

示例 1: 使用数据集名称指定 I/O 模块

在此示例中, 通过 BOPT 行来确定输入和输出模块。已精确指定输入和输出数据集名称, 而不是在 DD 名称中指定。

```
..
//SYSIN DD *
ARCH=12
BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1)
```

示例 2: 使用 DD 名称指定 I/O 模块

在此示例中, 通过使用 DD 名称来指定输入和输出模块。

```
..
//SYSBIN DD DSN=HLQ.IN.LOAD,DISP=SHR
//SYSBOUT DD DSN=HLQ.OUT.LOAD,DISP=SHR
//SYSIN DD *
ARCH=12
BOPT IN=DD:SYSBIN(MEM1) OUT=DD:SYSBOUT(MEM1)
```

示例 3: 使用 HFS 路径指定 I/O 模块

在此示例中, 通过使用 HFS 路径来指定输入和输出模块。必须使用以斜杠 (/) 开头的标准 HFS 路径。

```
..
//SYSIN DD *
ARCH=12
BOPT IN=/home/user1/a.out OUT=/home/user1/a.out.opt
```

示例 4: 指定输入模块并省略输出成员说明符

在此示例中, 指定输入模块并省略输出成员说明符。优化模块的输出 PDS(E) 中的成员名称将与输入 PDS(E) 中指定成员的名称相同。

```
..
//SYSBIN DD DSN=HLQ.IN.LOAD,DISP=SHR
//SYSBOUT DD DSN=HLQ.OUT.LOAD,DISP=SHR
//SYSIN DD *
```

```

ARCH=10
BOPT IN=DD:SYSBIN(MEM1) OUT=DD:SYSBOUT

```

示例 5：使用表达式指定多个输入模块

在此示例中，使用表达式指定多个输入模块。优化模块的输出 PDS(E) 中的成员名称将与输入 PDS(E) 中的对应成员名称相同。

要包含名称以 MEM 开头的模块：

```

...
//SYSBIN DD DSN=HLQ.IN.LOAD,DISP=SHR
//SYSBOUT DD DSN=HLQ.OUT.LOAD,DISP=SHR
//SYSIN DD *
ARCH=10
BOPT IN=DD:SYSBIN(MEM*) OUT=DD:SYSBOUT

```

要排除名为 MEMA 的单个模块：

```

...
//SYSBIN DD DSN=HLQ.IN.LOAD,DISP=SHR
//SYSBOUT DD DSN=HLQ.OUT.LOAD,DISP=SHR
//SYSIN DD *
ARCH=10
BOPT IN=DD:SYSBIN(!MEMA) OUT=DD:SYSBOUT

```

要排除名称以 MEMB 开头的所有成员：

```

...
//SYSBIN DD DSN=HLQ.IN.LOAD,DISP=SHR
//SYSBOUT DD DSN=HLQ.OUT.LOAD,DISP=SHR
//SYSIN DD *
ARCH=10
BOPT IN=DD:SYSBIN(!MEMB*) OUT=DD:SYSBOUT

```

要排除名为 SUB1 和 SUB2 的成员：

```

...
//SYSBIN DD DSN=HLQ.IN.LOAD,DISP=SHR
//SYSBOUT DD DSN=HLQ.OUT.LOAD,DISP=SHR
//SYSIN DD *
ARCH=10
BOPT IN=DD:SYSBIN(!SUB1|SUB2) OUT=DD:SYSBOUT

```

示例 6：使用多个 BOPT 优化器伪指令指定多个输入模块

在此示例中，使用 BOPT 优化器伪指令指定多个输入模块：

```

...
//SYSIN DD *
ARCH=10
BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1)
BOPT IN=HLQ.IN.LOAD(MEM5) OUT=HLQ.OUT.LOAD(MEM5)

```

示例 7：使用 REPLACE 选项绕过优化或扫描

在此示例中，使用 REPLACE 选项。如果关联的输出模块已存在，那么 REPLACE=N 将绕过优化或扫描。在此示例中，优化器在第一个和第三个 BOPT 伪指令中执行优化。优化器绕过第二个 BOPT 伪指令中的优化和最后一个 BOPT 伪指令（指定 REPLACE=N）中的扫描，并在第一个和第三个 BOPT 伪指令中创建了输出模块。

```

...
//SYSIN DD *
ARCH=10
BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1) REPLACE=Y
BOPT IN=HLQ.IN.LOAD(MEM1) OUT=HLQ.OUT.LOAD(MEM1) REPLACE=N
BOPT IN=HLQ.IN.LOAD(*) OUT=HLQ.OUT.LOAD REPLACE=Y
BOPT IN=HLQ.IN.LOAD(*) OUT=HLQ.OUT.LOAD SCAN=Y REPLACE=N

```

示例 8：指定全局和局部优化器选项

在本示例中，将使用全局和局部选项。ARCH=11 和 REPLACE=N 应用于第一个 BOPT 伪指令；ARCH=10 和 REPLACE=Y 应用于第二个 BOPT 伪指令。

```
...  
//SYSBOUT DD DSN=HLQ.OUTPUT.LOAD,DISP=SHR  
//SYSIN DD *  
  ARCH=10 REPLACE=N  
  BOPT IN=HLQ.IN.LOAD(MEM1) OUT=DD:SYSBOUT(MEM1) ARCH=11  
  BOPT IN=HLQ.IN.LOAD(MEM1) OUT=DD:SYSBOUT(MEM1) REPLACE=Y
```

示例 9：使用全局 SCAN 切换到扫描方式

以下示例使用全局选项 SCAN 将输入模块优化方式切换到输入模块扫描方式。在此扫描方式中，不会写入到输出模块。将会处理 OUT 选项（例如，根据正确的语法进行处理），但会忽略其他选项。有关 SCAN 选项的详细信息，请参阅第 23 页的『SCAN』。

对于第二个 BOPT 伪指令，优化器会还原为优化方式。

```
...  
//SYSBOUT DD DSN=HLQ.OUTPUT.LOAD,DISP=SHR  
//SYSIN DD *  
  ARCH=11 SCAN=Y  
  BOPT IN=HLQ.IN.LOAD(MEM1) OUT=DD:SYSBOUT(MEM1)  
  BOPT IN=HLQ.IN.LOAD(MEM1) OUT=DD:SYSBOUT(MEM1) SCAN=N
```

示例 10：指定 ALLOW 选项

此示例显示了对于第二个 BOPT 伪指令使用局部覆盖指定 ALLOW=NOUNRESEXE 的全局设置。

使用 ALLOW=NOUNRESEXE 执行第一个 (MEM1) 和第三个 (MEM3) BOPT 伪指令，这意味着仅接受完全绑定的程序模块。如果遇到部分绑定的程序模块，那么将发出 BOZ1494S 消息。

使用 ALLOW=UNRESEXE 执行第二个 (MEM2) BOPT 伪指令，因此接受完全绑定和部分绑定的程序模块作为输入。

```
...  
//SYSBOUT DD DSN=HLQ.OUTPUT.LOAD,DISP=SHR  
//SYSIN DD *  
  ARCH=11 ALLOW=NOUNRESEXE  
  BOPT IN=HLQ.IN.LOAD(MEM1) OUT=DD:SYSBOUT(MEM1)  
  BOPT IN=HLQ.IN.LOAD(MEM2) OUT=DD:SYSBOUT(MEM2) ALLOW=UNRESEXE  
  BOPT IN=HLQ.IN.LOAD(MEM3) OUT=DD:SYSBOUT(MEM3)
```

使用 IEFOPZ 指定优化

以下示例显示了使用 IEFOPZ 优化器伪指令的 JCL。

此部分中的示例都不是完整的示例。它们旨在反映用户应该在 SYSIN 文件中指定的内容。对于基本 JCL 配置，请参阅第 57 页的『附录 A JCL 示例』。

示例 1：单个 ARCH 配置

此示例显示了用于通过 IEFOPZ 伪指令运行 ABO 的最少 JCL。

在 IEFOPZ 配置仅包含一个 ARCH 级别时，您可以使用此 JCL。

```
...  
//SYSIN DD *  
  IEFOPZ
```

示例 2：多个 ARCH 配置

如果 IEFOPZ 配置包含多个 ARCH 级别，请指定独立的 IEFOPZ 伪指令行以避免发生列表文件名冲突。

```
//SYSIN DD *
IEFOPZ SEL_ARCH=11 LIST=HLQ.OUT1.ARCH11.LIST
IEFOPZ SEL_ARCH=12 LIST=HLQ.OUT1.ARCH12.LIST
```

示例 3：使用 SEL_STATE 和 SEL_ARCH 选择器限制优化

以下示例针对标记为 INACTIVE 的那些 OLDNEW 映射，生成了优化模块。与示例 2 中一样，如果映射具有多个 ARCH 级别，请使用 LIST 选项以避免发生列表文件名冲突。

```
//SYSIN DD *
IEFOPZ SEL_STATE=INACTIVE SEL_ARCH=10 LIST=HLQ.OUT.ARCH10.LIST
IEFOPZ SEL_STATE=INACTIVE SEL_ARCH=11 LIST=HLQ.OUT.ARCH11.LIST
```

示例 4：使用 SEL_OLD 选择器限制优化

以下示例针对与指定模式匹配的 OLD 数据集成员，生成了优化模块。

在第一行中，针对 HLQ.IN.LOAD 的所有成员，生成优化模块。

在第二行中，针对与 HLQ.IN.* 匹配的 OLD 数据集中与 M* 匹配的数据集成员，生成优化模块。在第二行上指定了 REPLACE=N，以避免重新优化第一行中的模块。

```
//SYSIN DD *
IEFOPZ SEL_OLD=HLQ.IN.LOAD
IEFOPZ SEL_OLD=HLQ.IN*.(M*) REPLACE=N
```

z/OS JCL REGION 和 JCL MEMLIMIT 参数的建议设置

ABO 优化方法

为生成高性能的优化模块，除 Enterprise COBOL V4 和更低版本提供的功能外，ABO 还执行高级分析并使用需要大量机器资源的代码优化方法。

此外，由于可一次性对多个已编译程序轻松调用 ABO，因此请求的处理量会导致资源总量非常高。可使用 ABO 批量优化整个数据集，而这个数据集可能包含多个模块，每个模块自身也可包含多个已编译程序 (CSECT)。因此，与一次性在单个源文件上运行的编译过程相比，ABO 一次性优化几百或几千个已编译程序所需的资源总量更高一些。

优化模块所需的时间和内存取决于以下因素：

- 模块中 CSECT 的数量。
- 每个 CSECT 的复杂性。复杂性同时受已编译 PROCEDURE DIVISION 语句的大小以及输入程序 DATA DIVISION 的大小的影响。

相应地设置 z/OS JCL REGION 和 JCL MEMLIMIT 参数

如第 57 页的『附录 A JCL 示例』中所示，JCL REGION 参数应设置为 0M 以便为 ABO 提供运行所需的内存。

ABO 将使用高于 2 GB BAR 的存储器来优化大型 CSECT。这意味着 z/OS MEMLIMIT 参数应设置为足够高的值以允许 ABO 处理成功完成。

0M 的建议 REGION 设置会将 MEMLIMIT 设置为 NOLIMIT。但是，JOB 或 EXEC 语句上的 MEMLIMIT 设置或出口例程 IEFUSI 可将此 NOLIMIT 值改写为较小的值。ABO 所需的存储容量将取决于要优化的 CSECT 的数量和大小。

如果将 MEMLIMIT=NOLIMIT 改写为较小的值并且此 MEMLIMIT 设置不够大，那么您可能会收到以下 ABO 消息之一：

```
BOZ1145U Insufficient memory in the compiler to continue compilation.  
BOZ1428U Insufficient memory encountered during binder API "&1": return code=&2  
         reason code=&3. Terminating optimizer.  
BOZ1449U Unhandled out of memory exception
```

优化超大型 CSECT 或大量小型 CSECT 可能需要 10GB 或更高的 MEMLIMIT 设置。如果遇到任何 ABO BOZ1145U、BOZ1428U 或 BOZ1449U 消息，请将 MEMLIMIT 设置增大到较大的值。

有关 JCL REGION 和 JCL MEMLIMIT 参数的更多信息，请参阅 [z/OS MVS Initialization and Tuning Reference](#) 和 [z/OS MVS Initialization and Tuning Guide](#)。

指定要用于 ABO 消息的语言

CEEOPTS DD 用于指定 ABO 生成的消息的语言。

缺省情况下，用英语显示消息。要指定您想要用日语显示消息，请向 JCL 添加以下代码：

```
//CEEOPTS DD *  
          NATLANG(JPN)  
/*
```

通过 TSO、REXX 和汇编程序代码来调用 ABO

本部分描述了如何通过 TSO、REXX 和汇编程序来调用 ABO。

在 TSO 下进行优化

在 TSO 下，您可以使用 TSO 命令、命令列表 (CLIST)、REXX exec 或 ISPF 以利用传统 MVS 数据集来优化 COBOL 程序。您可以使用 TSO 命令或 REXX exec 以利用 z/OS UNIX 文件来优化程序。

对于每种方法，您都需要通过执行以下步骤来分配数据集并请求优化：

1. 使用 ALLOCATE 命令分配数据集。您可以按任意顺序分配数据集。但是，在开始优化之前，必须分配所有必需的数据集。
2. 在 SYSIN 数据集中提供优化器参数。
3. 在 READY 提示符处使用以下 CALL 命令来请求优化：CALL 'hlqboz.SBOZMOD1(BOZOPT)'

您可以在 TSO 命令行上指定 ALLOCATE 和 CALL 命令；如果不使用 z/OS UNIX 文件，那么可将其包含在 CLIST 中。

您可以为所有优化器数据集分配 z/OS UNIX 文件，前提是这些数据集不是 PDS 或 PDSE 库。例如，如果 ABO 参数存储在 UNIX 文件 /u/myu/abo.parms 中，那么 ALLOCATE 语句的格式如下：

```
Allocate File(SYSIN) Path(' /u/myu/abo.parms' ) Pathopts(ORDONLY) Filedata(TEXT)
```

用于在 TSO 下进行优化的 ALLOCATE 和 CALL

以下示例说明了在 TSO 下进行优化时如何指定 ALLOCATE 和 CALL 命令。请注意，所有文件都可以是新文件或现有文件，但输入文件 SYSIN 和 SYSBIN 除外，这些输入文件在优化开始之前必须已存在。

```
[READY]  
FREE F(SYSPRINT SYSIN OPTLOG OPTERR CEEDUMP SYSBIN  
SYSBOUT)  
[READY]  
ALLOC F(SYSPRINT) DA(ABO.LISTING) SPACE(10,10) CYL NEW CATALOG RECFM(V B) LRECL(512)  
BLKSIZE(27998) DSOrg(PS)  
[READY]  
ALLOC FI(OPTLOG) DA(ABO.OPTLOG) SPACE(10,10) CYL NEW CATALOG RECFM(V B) LRECL(512)
```

```

BLKSIZE(27998) DSORG(PS)
[READY]
ALLOC FI(OPTERR) DA(ABO.OPTERR) SPACE(10,10) CYL NEW CATALOG RECFM(V B) LRECL(512)
BLKSIZE(27998) DSORG(PS)
[READY]
ALLOC FI(CEEDUMP) DA(ABO.CEEDUMP) SPACE(50,10) CYL NEW CATALOG RECFM(F B) LRECL(133)
BLKSIZE(27930) DSORG(PS)
[READY]
ALLOC FI(SYSIN) DA(ABO.SYSIN) SHR          /* supply ABO parameters within SYSIN file
*/
[READY]
ALLOC FI(SYSBIN) DA(IN.LOAD) SHR           /* supply COBOL load library to be optimized */
[READY]
ALLOC FI(SYSBOUT) DA(OUT.LOAD) SHR        /* supply load library for optimized load modules */
[READY]
CALL 'hlqboz.SBOZMOD1(BOZOPT)'
[READY]
FREE F(SYSPRINT SYSIN OPTLOG OPTERR CEEDUMP SYSBIN SYSBOUT)

```

用于在 TSO 下进行优化的 CLIST

以下示例显示了用于在 TSO 下进行优化的 CLIST：

```

PROC 1 HLQBOZ
FREE F(SYSPRINT SYSIN OPTLOG OPTERR CEEDUMP SYSBIN
SYSBOUT)
ALLOC FI(SYSPRINT) DA(ABO.LISTING) SHR
ALLOC FI(SYSIN) UNIT(SYSDA) SPACE(1,0) TRACKS NEW +
RECFM(F B) LRECL(80) BLKSIZE(800) DSORG(PS)
OPENFILE SYSIN OUTPUT
SET SYSIN = &STR(BOPT IN=DD:SYSBIN(MEMBER) OUT=DD:SYSBOUT(MEMBER))
PUTFILE SYSIN
CLOSEFILE SYSIN
ALLOC FI(OPTLOG) DA(ABO.OPTLOG) SHR
ALLOC FI(OPTERR) DA(ABO.OPTERR) SHR
ALLOC FI(CEEDUMP) DA(ABO.CEEDUMP) SHR
ALLOC FI(SYSBIN) DA(IN.LOAD) SHR
ALLOC FI(SYSBOUT) DA(OUT.LOAD) SHR
CALL '&HLQBOZ..SBOZMOD1(BOZOPT)'
FREE F(SYSIN SYSPRINT OPTLOG OPTERR CEEDUMP SYSBIN SYSBOUT)

```

用于在 TSO 下进行优化的 REXX

以下示例显示了用于在 TSO 下进行优化的 REXX：

```

/*=====>> REXX <<=====*/
Parse Arg hlqboz .                               /* get argument */
IF HLQBOZ = '' THEN DO
SAY 'HLQBOZ ARGUMENT MISSING'
EXIT
END
ADDRESS TSO
msgstat = MSG("OFF")
"FREE FILE (SYSIN SYSPRINT)"
"ALLOC FI(SYSPRINT) DA(ABO.LISTING) SHR"
"ALLOC FI(OPTLOG) DA(ABO.OPTLOG) SHR"
"ALLOC FI(OPTERR) DA(ABO.OPTERR) SHR"
"ALLOC FI(CEEDUMP) DA(ABO.CEEDUMP) SHR"
"ALLOC FI(SYSBIN) DA(IN.LOAD) SHR"
"ALLOC FI(SYSBOUT) DA(OUT.LOAD) SHR"
"ALLOC FI(SYSIN) NEW CYL SPACE(1,1) RECFM(F B)",
" LRECL(80) BLKSIZE(800) DSORG(PS)"
line.1 = 'ARCH=11'
line.2 = 'BOPT IN=DD:SYSBIN(MEMBER) OUT=DD:SYSBOUT(MEMBER) '
"EXECIO 2 DISKW SYSIN (STEM line. FINIS"
"CALL 'hlqboz.SBOZMOD1(BOZOPT)'"
"FREE FI(OPTLOG OPTERR CEEDUMP SYSPRINT SYSBIN SYSBOUT SYSIN)"
msgstat = MSG("ON")

```

用于在 TSO 批处理下进行优化并将 OPTLOG 和 SYSPRINT 输出定向到库成员的 REXX。

有时，您可能需要优化整个装入库（甚至是多个装入库）的每一个成员，而无需输入多个 BOPT 语句。

以下示例显示了用于运行 ABOMEMBS REXX（位于 SYSEXEC DD 名称引用的数据集中）的 TSO 批处理作业。ABOMEMBS 会仔细检查 SYSBIN 数据集并置，并单独优化该并置中的每一个数据集成员。

```
//OPTMEMBS EXEC PGM=IKJEFT01,PARM='ABOMEMBS',
REGION=0M
//STEPLIB DD DISP=SHR,DSN= hlqboz.SBOZMOD1
//SYSEXEC DD DISP=SHR,DSN=hlq.CLIST /* supply ABOMEMBS REXX member within SYSEXEC
library */
//SYSTSPRT DD SYSOUT=*,DCB=(LRECL=132,RECFM=FBA,BLKSIZE=1320)
//SYSTSIN DD DUMMY
//OPTERR DD SYSOUT=*
//CEEDUMP DD SYSOUT=*
//SYSBIN DD DISP=SHR,DSN=hlq.INLOAD1 /* supply one or more COBOL load libraries to be
optimized */
// DD DISP=SHR,DSN=hlq.INLOAD2
// DD DISP=SHR,DSN=hlq.INLOAD3
//SYSBOUT DD DISP=SHR,DSN=hlq.UTLOAD
//OPTLOG DD DISP=(,CATLG),DSN=hlq.OPTLOG,UNIT=3390, /* supply new or existing PDS/PDSE or
SEQ file */
// SPACE=(CYL,(5,5,20)),DSNTYPE=LIBRARY,
// DCB=(RECFM=VB,LRECL=512,BLKSIZE=0)
//SYSPRINT DD DISP=(,CATLG),DSN=hlq.SYSPRINT,UNIT=3390, /* supply new or existing PDS/PDSE or
SEQ file */
// SPACE=(CYL,(5,5,20)),DSNTYPE=LIBRARY,
// DCB=(RECFM=VB,LRECL=512,BLKSIZE=0)
//SYSIN DD *
ARCH=11
SCAN=N
# optionally put BOPTs below this line for specific library members optimization
BOPT IN=hlq.INLOAD7(A*) OUT=hlq.UTLOAD LOG=hlq.OPTLOG LIST=hlq.SYSPRINT
/*
```

对于上述 SYSBIN DD 并置中的每个成员，ABOMEMBS 会使用为该成员单独分配的 SYSIN、OPTLOG、SYSBIN 和 SYSPRINT 文件来调用 ABO。

例如，对于 hlq.INLOAD1 数据集中的成员 MEM1，它将分配 SYSBIN 文件 hlq.INLOAD1，并重新分配 OPTLOG 文件 hlq.OPTLOG(MEM1) 和 SYSPRINT 文件 hlq.SYSPRINT(MEM1)，前提是以上 JCL 中分配的 OPTLOG 和 SYSPRINT 文件是 PDS 或 PDSE 数据集。否则，它将继续使用 JCL 分配的这些文件。

如果要为 MEM1 构造 SYSIN 文件，它将使用以上 JCL SYSIN 文件中列出的所有 ABO 参数，直至第一条 BOPT 语句为止，然后为其附加内部生成的 BOPT 语句。下面是用于 MEM1 的 SYSIN 文件示例：

```
=====
ARCH=11
SCAN=N
BOPT IN=DD:SYSBIN(MEM1) OUT=DD:SYSBOUT(MEM1)
=====
```

分配所有单独的 SYSIN、OPTLOG、SYSPRINT 和 SYSBIN 文件后，会调用 ABO 来优化 MEM1。

ABOMEMBS exec 会针对 SYSBIN 并置中的每个成员重复此过程。如果在以上 JCL SYSIN 文件中存在 BOPT 语句，那么在优化 SYSBIN 并置中的所有成员后，ABOMEMBS 会将 OPTLOG 和 SYSPRINT 文件重新分配到 SYSOUT，重新分配与原始 JCL SYSIN 文件包含相同语句的 SYSIN 文件，并多次调用 ABO 以继续使用 SYSIN 文件中显式指定的参数。

此方法可用于优化所有库成员，只需将其包含在 SYSBIN 并置中而无需显式指定任何其他 BOPT，并可选择在只需优化部分库成员时包含额外的 BOPT 语句。以上 JCL 中的 JCL 卡可采用任意顺序。

ABOMEMBS 会生成以下 SYSTSPRT 输出：

```
===== Processing hlq.INLOAD1 data set =====
Member ADD10 processed, rc=4
Member CALLEE processed, rc=0
Member CALLEE1 processed, rc=0
...
===== Processing hlq.INLOAD2 data set =====
Member COBPGM processed, rc=0
Member DJSIEV85 processed, rc=0
Member IAMASM processed, rc=4
...
===== Processing hlq.INLOAD3 data set =====
Member TEMPNAM6 processed, rc=12
Member TEMPNAM7 processed, rc=0
Member TEMPNAM8 processed, rc=0
...
===== Processing SYSIN statements =====
ARCH=11
```

```

SCAN=N
BOPT IN=h1q.INLOAD7(A*) OUT=h1q.OUTLOAD LOG=h1q.OPTLOG LIST=h1q.SYSPRINT
=====
READY
END

```

下面是 ABOMEMBS REXX 源代码示例:

```

BROWSE      h1q.CLIST(ABOMEMBS)
Command ==>
***** Top of Data
/*=====> REXX <=====*/
  Parse Arg

  X = LISTDSI(STEPLIB FILE)
  If x > 0 Then Do
    Say 'Check STEPLIB allocation' ; Exit
  End
  optimizer = "''||SYSDSNAME||'(BOZOPT)''|'"
  flaglog = 0 /* assume OPTLOG is sequential */
  X = LISTDSI(OPTLOG FILE)
  IF x = 0 & SYSDSORG = 'P0' Then Do
    optlogds = SYSDSNAME ; flaglog = 1
  End
  flaglist = 0 /* assume SYSPRINT is sequential */
  X = LISTDSI(SYSPRINT FILE)
  IF x = 0 & SYSDSORG = 'P0' Then Do
    sysprtds = SYSDSNAME ; flaglist = 1
  End
  "EXECIO * DISKR SYSIN (STEM line. FINIS" /* read in SYSIN parameters into
line. array */
  n = line.0 + 1 /* assume no explicit BOPT specified. n is per-member BOPT
line number */
  Do 1 = 1 To
line.0
    PARSE UPPER VALUE line.1 WITH
line.1
    If POS('BOPT ',line.1) > 0 Then
  Do
    n = 1 ; flagbopt = 1 ; leave /* leave when first BOPT found
*/
  End
  linea.1 = line.1 /* copy all lines before first BOPT into linea. array
*/
  End
  X = LISTDSI(SYSBIN
FILE)
  If x > 0 Then
  Do
    Say 'Check SYSBIN allocation' ;
Exit
  End
  x =
outtrap(concatl.)
  lista
status
  x =
outtrap(off)
/* i2 is a line with last dsn in SYSBIN concatenation
*/
  i2 = concatl.0 - 1 /* assume that SYSBIN concatenation is a last in JCL
*/
  Do i = 1 To
concatl.0
    If substr(concatl.i,3,6) = 'SYSBIN' Then

```

```

Do
    i1 = i - 1 ; leave      /* i1 line with first dsn in concatenation
*/
End

End
k = i + 2
If k < concatl.0 Then Do
    Do i = k To concatl.0 by 2
        If substr(concatl.i,3,1) <> ' ' Then Do
            i2 = i - 3 ; leave /* i2 line with last dsn in concatenation */
        End
    End
End
Else i2 = i1                /* when SYSBIN DD with a single data set is a last JCL card
*/

msgstat = MSG("OFF")

Do i = i1 To i2 by 2      /* loop through SYSBIN concatenation */
    sysbinds = strip(concatl.i)
    Say '===== Processing 'sysbinds' data set ====='
    x=outtrap('row.')
    Address TSO "LISTDS '"sysbinds'" members"
    x=outtrap('off')
    If row.0 < 6 Then Do
        Say 'Check SYSBIN file' ; Exit
    End
    DO J = 7 TO row.0      /* loop through member list */
        PARSE VALUE row.J WITH memn alias
        "FREE FI(SYSIN)"
        "ALLOC FI(SYSIN) NEW CYL SPACE(1,1) RECFM(F B)",
        " LRECL(80) BLKSIZE(800) DSORG(PS)"
        linea.n = 'BOPT IN=DD:SYSBIN('||memn||') ',
        'OUT=DD:SYSBOUT('||memn||')'
        "EXECIO * DISKW SYSIN (STEM linea. FINIS"
        If flaglog = 1 Then Do
            "FREE FI(OPTLOG)"
            optlogm = ' '||optlogds||'('||memn||')'||' '
            "ALLOC FI(OPTLOG) DA("optlogm") SHR"
        End
        If flaglist = 1 Then Do
            "FREE FI(SYSPRINT)"
            sysprtm = ' '||sysprtds||'('||memn||')'||' '
            "ALLOC FI(SYSPRINT) DA("sysprtm") SHR"
        End
        "FREE FI(SYSBIN)"
        "ALLOC FI(SYSBIN) DA('"sysbinds"') SHR"
        "CALL "optimizer /* optimize member */
        Say 'Member 'memn' processed, rc='rc
    END /* end of member list loop */
End /* end of concatenation list loop */
If flagbopt = 1 Then Do
    "FREE FI(SYSIN)"
    "ALLOC FI(SYSIN) NEW CYL SPACE(1,1) RECFM(F B)",
    " LRECL(80) BLKSIZE(800) DSORG(PS)"
    "EXECIO * DISKW SYSIN (STEM linea. FINIS"
    "FREE FI(OPTLOG)"
    "ALLOC FI(OPTLOG) SYSOUT"
    "FREE FI(SYSPRINT)"
    "ALLOC FI(SYSPRINT) SYSOUT"
    "CALL "optimizer /* optimize with BOPTs provided in SYSIN */
    Say '===== Processing SYSIN statements ====='
    Do i = 1 To line.0
        Say line.i
    End
    Say '===== '
End
msgstat = MSG("ON")
EXIT

```

从汇编程序中启动优化器

您可以从 HLASM 程序中以编程方式调用 ABO。

在开始优化之前，请完成以下步骤：

1. 通过在汇编程序中使用动态分配或者通过在用于汇编程序调用的作业 JCL 中指定 DD 卡，以分配所有必需的数据集。必须分配以下 DD 名称：SYSPRINT、SYSBIN、SYSBOUT、SYSIN、OPTLOG、OPTERR 和 CEEDUMP。
2. 在 SYSIN 数据集中提供 ABO 参数。

您可以通过使用 LINKX 或 ATTACHX 宏从汇编程序中启动 ABO，因为这两个宏均兼容 64 位方式，而 ABO 是在 AMODE 64 中运行的。

下面是列表格式的 LINKX 宏示例：

```
symbol {LINKX} EP=BOZOPT,AMODE640K=YES,PLIST8=YES,SF=L
```

下面是执行格式的 LINKX 宏示例：

```
LINKX EP=BOZOPT,AMODE640K=YES,PARAM=(addr),PLIST8=YES,      *  
      MF=(E,#LINKX),SF=(E,#LINK2)
```

其中 # 用作前缀符号。

EP

指定 ABO 的符号名称。

PARAM

指定要从汇编程序传递到 ABO 的地址参数列表。在此示例中，*addr* 可以是任意值，因为 BOZOPT 程序会将其忽略，而直接读取 SYSIN 文件中提供的参数。

PLIST8=YES

将 LINKX 基于 PARAM 关键字所构建的参数列表的参数列表条目大小定义为每个条目参数列表 8 个字节。

AMODE640K=YES

指示系统将接受从 AMODE 24 或 AMODE 31 例程链接至 AMODE 64 目标例程的尝试。

SF=L

指定列表格式的 LINKX 宏。

ABO 完成处理后，会将返回码放入寄存器 15 中。

用于启动优化器的汇编程序

以下示例显示了来自汇编程序的 ABO 调用：

```
//JOB CARD JOB  
//*  
//ASM HCL PROC MAC='SYS1.MACLIB',MAC1='SYS1.MODGEN',U=3390,  
//          MAC2='SYS1.MACLIB'  
//*-----*  
//*          A S M H C L      H-ASSEMBLER          ***  
//*          IBM-PROCEDURE: COMPILE + LINK          ***  
//*-----*  
//ASM      EXEC PGM=ASMA90,PARM=OBJECT,REGION=0M  
//SYSLIB   DD   DSN=&MAC,DISP=SHR  
//          DD   DSN=&MAC1,DISP=SHR  
//          DD   DSN=&MAC2,DISP=SHR  
//SYSUT1   DD   DSN=&&SYSUT1,UNIT=&U,SPACE=(TRK,(60,45))  
//SYSPRINT DD   SYSOUT=*,DCB=BLKSIZE=1089  
//SYSPUNCH DD   DUMMY  
//SYSLIN   DD   DSN=&&OBJSET,UNIT=&U,SPACE=(80,(2000,500)),  
//          DISP=(MOD,PASS)
```

```

//LKED      EXEC PGM=HEWLH096,
//          PARM='XREF,LET,LIST,AC=0,FILL=NONE',
//          COND=(8,LT,ASM)
//SYSLIN    DD DSN=&&OBJSET,DISP=(OLD,DELETE)
//          DD DDNAME=SYSIN
//SYSLMOD   DD DSN=h1q.CALLABO.LOAD,DISP=SHR
//SYSUT1    DD DSN=&&SYSUT1,UNIT=(&U,SEP=(SYSLIN,SYSLMOD)),
//          SPACE=(1024,(50,20))
//SYSPRINT  DD SYSOUT=*
//          PEND
//*
//          EXEC ASMHCL
//ASM.SYSIN DD
*
          YREGS
CALLABO    CSECT
CALLABO    AMODE 31
CALLABO    RMODE ANY
          STM R14,R12,12(R13)
          USING CALLABO,R12
          LR R12,R15          LOAD BASE REGISTER
          LA R9,SAVE          POINT TO CURRENT SAVEAREA
          ST R9,8(,R13)        A(CURRENT_SA) IN OLD_SA
          ST R13,4(,R9)        A(OLD_SA) IN CURRENT_SA
          LR R13,R9           R13=A(CURRENT_SAVEAREA)
*-----*
          LLGTR R12,R12          allow 64-bit
addressability
          BAL R14,LINKOPT        invoke BOZOPT
*-----*
DONE       DS 0H
          L R13,4(,R13)
          ST R15,16(,R13)
          LM R14,R12,12(R13)
          BR R14
*-----*
LINKOPT    DS 0H
          XR R15,R15            nullify parm addr, BOZOPT ignores
it
          LLGTR R15,R15
          SAM64
          LINKX EP=BOZOPT,AMODE640K=YES,PARAM=((R15)),PLIST8=YES,      *
          MF=(E,#LINKX),SF=(E,#LINK2)
          SAM31
          B DONE
*-----*
          DS 0F
SAVE       DC XL72'00'
#LINKX    LINKX EP=BOZOPT,AMODE640K=YES,PLIST8=YES,SF=L
#LINK2    LINKX EP=BOZOPT,AMODE640K=YES,PLIST8=YES,SF=L
          LTORG
          END
//LKED.SYSIN DD *
          PAGE CALLABO
          NAME CALLABO(R)
//

```

下面是用于运行 CALLABO 程序的样本 JCL:

```

//OPT       EXEC PGM=CALLABO,REGION=0M
//STEPLIB   DD DSN=h1qboz.SBOZMOD1,DISP=SHR
//          DD DSN=h1q.CALLABO.LOAD,DISP=SHR
//OPTLOG    DD SYSOUT=*
//OPTERR    DD SYSOUT=*
//CEEDUMP   DD SYSOUT=*
//SYSPRINT  DD SYSOUT=*
//SYSBIN    DD DSN=h1q.IN.LOAD,DISP=SHR
//SYSBOUT   DD DSN=h1q.OUT.LOAD,DISP=SHR
//SYSIN     DD *
# supply ABO directives down this line
BOPT IN=DD:SYSBIN(*) OUT=DD:SYSBOUT

```


第 6 章 了解优化流程的输出

日志文件

IBM Automatic Binary Optimizer for z/OS 会生成一个日志文件，您可以通过此日志文件来确定和解决问题。

通过查看 ABO 在 OPTLOG DD 中无条件生成的日志文件，可以诊断在优化时出现的问题（请参阅第 13 页的表 6）。该日志文件包含有关优化和扫描的诊断信息。

要生成额外的成员级别的日志文件，请使用 **LOG** 选项。

针对优化的日志文件

对模块执行优化（SCAN=N 优化器选项生效）时，该日志文件包含以下信息：

- 所处理的输入模块的文件名信息
- 所优化的 CSECT 的名称
- 已优化的每个 CSECT 的列表转换的文件名信息
- 已优化的输出模块的文件名信息
- OPTLOG 的每一行的时间戳记和日期的标题
- 其他诊断信息（包括错误消息）

示例 1：

以下日志文件显示正在优化的数据集 *HLQ.IN1.LOAD* 的成员 MEMA 和 MEMB 中的 COBOL CSECT。模块 MEMA 有两个已优化的 COBOL CSECT（名为 SUB1 和 SUB2）。模块 MEMB 有一个已优化的 COBOL CSECT（名为 PROGB）。列表转换全都放在缺省 SYSPRINT DD 中。优化模块将写入到 *HLQ.OUT1.LOAD* 数据集中。

```
5697-AB2 IBM Automatic Binary Optimizer for z/OS 2.1.0

===== Sept 24 2019 =====
10:53:16 Optimizer build level: tr_v21_binopt_20190730_1658_t54Rw7MMEem46oTHworqTQ
(Jul 30 2019 20:05:19)
10:53:16 Processing HLQ.IN1.LOAD, member MEMA
10:53:16   Processing CSECT SUB1, in member MEMA
10:53:16     Optimizing CSECT SUB1 for zEC12
10:53:16       Succeeded in optimizing SUB1
10:53:16       Generating listing transform into DD:SYSPRINT
10:53:16   Processing CSECT SUB2, in member MEMA
10:53:16     Optimizing CSECT SUB2 for zEC12
10:53:16       Succeeded in optimizing SUB2
10:53:16       Generating listing transform into DD:SYSPRINT
10:53:16   Finished processing, processed 2 of 2 CSECTs in member MEMA
10:53:16     Save HLQ.OUT1.LOAD (MEMA) succeeded
10:53:16 Processing HLQ.IN1.LOAD, member MEMB
10:53:16   Processing CSECT PROGB, in member MEMB
10:53:16     Optimizing CSECT PROGB for zEC12
10:53:16       Succeeded in optimizing PROGB
10:53:16       Generating listing transform into DD:SYSPRINT
10:53:16   Finished processing, processed 1 of 1 CSECTs in member MEMB
10:53:16     Save HLQ.OUT1.LOAD (MEMB) succeeded
10:53:16 Exiting with return code: 0
```

示例 2：

以下日志文件显示由与先前指定了 LOG=HLQ.LOG.OUT 的示例相同的 HLQ.IN1.LOAD 生成的成员级别日志文件的输出。

HLQ.LOG.OUT(MEMA) 的内容:

```
5697-AB2 IBM Automatic Binary Optimizer for z/OS 2.1.0
===== Sept 24 2019 =====
10:53:16 Optimizer build level: tr_v21_binopt_20190730_1658_t54Rw7MMEem46oTHworqTQ (Jul 30 2019 20:05:19)
10:53:16 Processing HLQ.IN1.LOAD, member MEMA
10:53:16   Processing CSECT SUB1, in member MEMA
10:53:16     Optimizing CSECT SUB1 for zEC12
10:53:16     Succeeded in optimizing SUB1
10:53:16     Generating listing transform into DD:SYSPRINT
10:53:16   Processing CSECT SUB2, in member MEMA
10:53:16     Optimizing CSECT SUB2 for zEC12
10:53:16     Succeeded in optimizing SUB2
10:53:16     Generating listing transform into DD:SYSPRINT
10:53:16   Finished processing, processed 2 of 2 CSECTs in member MEMA
10:53:16   Save HLQ.OUT1.LOAD (MEMA) succeeded
10:53:16 Exiting with return code: 0
```

HLQ.LOG.OUT(MEMB) 的内容:

```
5697-AB2 IBM Automatic Binary Optimizer for z/OS 2.1.0
===== Sept 24 2019 =====
10:53:16 Optimizer build level: tr_v21_binopt_20190730_1658_t54Rw7MMEem46oTHworqTQ (Jul 30 2019 20:05:19)
10:53:16 Processing HLQ.IN1.LOAD, member MEMB
10:53:16   Processing CSECT PROGB, in member MEMB
10:53:16     Optimizing CSECT PROGB for zEC12
10:53:16     Succeeded in optimizing PROGB
10:53:16     Generating listing transform into DD:SYSPRINT
10:53:16   Finished processing, processed 1 of 1 CSECTs in member MEMB
10:53:16   Save HLQ.OUT1.LOAD (MEMB) succeeded
10:53:16 Exiting with return code: 0
```

针对扫描的日志文件

执行扫描 (SCAN=Y 优化器选项生效) 时, 该日志文件显示以下信息:

- 所扫描的输入模块的文件名信息
- 所扫描的模块的 CSECT 的名称
- 其他诊断消息

示例:

以下日志文件显示正在扫描数据集 HLQ.MEM1.LOAD 的模块 MEMA 和 MEMB。扫描输出显示了这些模块中的所有 CSECT。针对 COBOL CSECT SUB、SUB2 和 PROGB 显示用于编译的 COBOL 编译器版本和从 CSECT 抽取的“Signature information bytes”。《COBOL 编程指南》中记录了“Signature information bytes”, 其中提供了有关已编译的程序的信息。

```
5697-AB2 IBM Automatic Binary Optimizer for z/OS 2.1.0
===== Sept 24 2019 =====
10:58:23 Optimizer build level: tr_v21_binopt_20190730_1658_t54Rw7MMEem46oTHworqTQ (Jul 30 2019 20:05:19)
10:58:23 Processing HLQ.IN1.LOAD, member MEMA
  Language ID Records:
    id 5688187    v21 m00 2015281 resident EDCOEXTS
    id 5655S7100 v42 m00 2015281 resident SUB
  Enterprise COBOL V4: start=0x10, length=4.19 (kBytes)
    Signature information bytes:
      a0487d4c 20000000 00880100 00000040
      08000000 000000 00008004 1400
    id 5655S7100 v42 m00 2015281 resident SUB2
  Enterprise COBOL V4: start=0x10d8, length=4.20 (kBytes)
    Signature information bytes:
      a0487d4c 20000000 00880100 00000040
```

```

08000000 000000 00008004 1400
id 569623400 v01 m06 2013071 resident CEESG005
id 569623400 v01 m06 2013072 resident CEEBETBL
id 569623400 v01 m06 2013072 resident CEESTART
10:58:23 Processing HLQ.IN1.LOAD, member MEMB
Language ID Records:
id 5688187 v21 m00 2015281 resident EDCOEXTS
id 5655S7100 v42 m00 2015281 resident PROGB
Enterprise COBOL V4: start=0x10, length=8.19 (kBytes)
Signature information bytes:
a0487d4c 20000000 00880100 00000040
08000000 000000 00008004 1400
id 569623400 v01 m06 2013072 resident CEESTART
10:58:23 Exiting with return code: 0

```

列表转换

列表转换描述 Automatic Binary Optimizer for z/OS 对已编译的程序模块所做的更改。它显示了如何将输入二进制文件中的指令映射到优化的指令。针对已优化的每个 CSECT，都会生成一个列表转换。列表转换旨在补充从源代码初始编译输入二进制文件时生成的编译器列表。虽然列表转换并不依赖于编译器列表，但是将两者一起使用可帮助您更好地了解 IBM Automatic Binary Optimizer for z/OS 如何转换您的二进制。

对于每个优化的 CSECT，将无条件生成列表转换；无需指定任何特殊选项或标志。缺省情况下，将在 SYSPRINT DD 中生成列表转换。

第 52 页的『Application Delivery Foundation for z Systems』也将使用列表转换。

列表转换内容

提供了列表转换以帮助诊断在执行优化程序期间遇到的问题。列表转换主要供调试工具（例如，IBM Debug for z Systems）使用。

列表转换包含以下信息：

- 优化选项的摘要
- 用输入指令点缀的优化指令
- 包含由 ABO 创建的任何新字面值的字面值池
- 包含已优化 CSECT 的相关信息的 PPA4 部分
- 由 ABO 创建的任何新堆栈符号的自动映射（也称为动态存储区域 (DSA) 映射）
- 包含要优化的 CSECT 的完整指令和编译器选项列表的输入指令部分。

优化参数的摘要

此部分包含对程序进行了优化的体系结构级别的名称，以及输入二进制文件和用于生成此二进制文件的编译器的日期和时间戳记。它还会显示输出二进制文件的日期和时间戳记。

示例：

```

Invocation Parameters:
  Architecture Level: zEC12

Input IDRL Record: 5655S7100 v42 m00 2013122
  Name: Enterprise COBOL V4
  Version: 42
  Mod Level: 00
  Compiled Date (YYYYDDD): 2013122

Output IDRL Record: 5697-AB2 v21 m00 2019256
  Name: IBM Automatic Binary Optimizer for z/OS
  Version: 21
  Mod Level: 00
  Optimized Date (YYYYDDD): 2019256

```

优化指令

列表转换中的大部分数据来自于此部分。此部分类似于各种 IBM COBOL 编译器（例如，IBM Enterprise COBOL V5 和 V6）所生成的列表的对象代码部分。已优化的每个 CSECT 都以 PROC 伪操作码开始，并且使用 CSECT 的名称作为其操作数。

示例：

(1)	(2)	(3)	(4)	(5)
000258		000000	PROC	PROGA
000258	183F	000000	LR	R3,R15
00025A	5800 3008	000000	L	R0, 8(,R3)
00025E	1E01	000000	ALR	R0,R1
000260	5500 C00C	000000	CL	R0, 12(,R12)
000264	0DF0	000000	BASR	R15,R0
000266	47D0 F00C	000000	BC	R13, 12(,R15)
00026A	58F0 C300	000000	L	R15, 768(,R12)
00026E	0DEF	000000	BASR	R14,R15
000270	181F	000000	LR	R1,R15
000272	50D0 1004	000000	ST	R13, 4(,R1)
000276	5000 104C	000000	ST	R0, 76(,R1)
00027A	D203 1000 3058	000000	MVC	0(4,R1), 88(R3)
000280	D703 1084 1084	000000	XC	132(4,R1), 132(R1)
000286	5090 105C	000000	ST	R9, 92(,R1)
00028A	18D1	000000	LR	R13,R1
00028C	41A0 D120	000000	LA	R10, 288(,R13)

以上示例显示了针对名为 PROGA 的 CSECT 生成的优化指令。按如下所示描述优化后指令的五个部分：

1. 优化指令的 CSECT 中的十六进制偏移量
2. 指令字节的十六进制表示
3. 为其生成这些优化指令的“源”指令的十六进制 CSECT 偏移量
4. 指令操作码
5. 指令操作数

“源”指令（已为其生成优化指令）使用优化指令来点缀。第 5 列中的行是优化指令，第 1 列中的行是“源”指令。在以下示例中，第一列前两行是十六进制偏移量分别为 00042C 和 000432 的 PACK 和 OI 源指令。第 5 列第 3 行是 ABO 生成的指令“CDZT”。请注意，CDZT 的“源”十六进制偏移量为 00042c，这表示是针对输入模块中的 PACK 指令生成此指令。

示例：

00042C	PACK	272(4,13),0(7,8)			
000432	OI	276(13),15			
0004C4	ED07	4000 00AA	00042C	CDZT	FP0,_WSA[0x12c] 0(8,R4),0x0
000436	PACK	280(4,13),8(7,8)			
00043C	OI	284(13),15			
0004CA	ED07	4008 10AA	000436	CDZT	FP1,_WSA[0x12c] 8(8,R4),0x0
000440	AP	272(4,13),280(4,13)			
000446	UNPK	16(7,8),272(4,13)			
0004D0	B3D2	1000	000440	ADTR	FP0,FP0,FP1
00044C	OI	23(8),240			
000450	L	2,248(0,,13)			
0004D4	ED07	4010 00A8	00044C	CZDT	FP0, 16(8,R4),0x0

字面值池

ABO 将自己创建的所有新字面值都放在代码部分的结尾处。在列表转换中，这些部分称为常量数据片段。列表中可能存在 0 个或多个常量数据片段，并且其内容非常类似于原始编译器创建的字面值池。原始字面值池保持不变并可继续使用，就如同在输入二进制文件中一样。

示例：

(1)	(2)	(3)	(4)
		L0032:	# Constant Data Snippet
000550	4040 4040 4040 4040	DC	X'4040404040404040'
000558	4040 4040 4040 4040	DC	X'4040404040404040'
000560	4040 4040 4040 4040	DC	X'4040404040404040'
000568	4040 4040 4040 0000	DC	X'4040404040400000'
000570	8000 0000 0000 0000	DC	X'8000000000000000'
000578	0000 0000 0000 0000	DC	X'0000000000000000'
000580	0000 0001 0000 0000	DC	X'0000000100000000'
000588	C9C7 E9E2 D9E3 C3C4	DC	X'C9C7E9E2D9E3C3C4'
000590	E2E8 E2D6 E4E3 4040	DC	X'E2E8E2D6E4E34040'
000598	0E00 0000 0000 0000	DC	X'0E00000000000000'

数据的四个部分的描述如下所示：

1. 字面值池中从 CSECT 起始位置到这些字节的十六进制偏移量
2. 字面值池中字节的十六进制表示
3. 用于表示字面值池开始位置的标签
4. 这些字节的汇编程序语法

其他 DSA 和 TGT 字节已分配的部分

此部分以十六进制字节数显示优化器分配的任何其他 DSA 或 TGT 字节

示例：

DSA WILL BE ALLOCATED FOR AN ADDITIONAL 000001A8 BYTES
TGT WILL BE ALLOCATED FOR AN ADDITIONAL 00000000 BYTES

PPA4 部分

PPA4 部分包含有关程序模块优化的信息。例如，它包含优化的时间和日期、代码部分的长度以及其他信息。

示例：

(1)	(2)	(3)	(4)
	PPA4: Entry Point Constants		
046668	00000000	=X'00000000'	Flags 1
04666C	00000300	=X'00000300'	Flags 2
046670	F2F0F1F6	=C'2019'	Compiled Year
046674	F0F8F2F1	=C'0913'	Compiled Date MMDD
046678	F1F2F3F0F2F3	=C'123023'	Compiled
Time HHMMSS			
04667E	F0F1F0F2F0F0	=C'010200'	Compiler
Version			
046684	0004706A	=F'290922'	Code
Length			
046688	0B020000	=X'0B020000'	Options
04668C	00000028	=X'00000028'	A(PPA4-
ListName)			
	PPA4 End		

下面描述了 PPA4 的四个部分：

1. PPA4 部分条目的 CSECT 中的十六进制偏移量
2. PPA4 部分中字节的十六进制表示
3. PPA4 中字节的汇编程序语法
4. PPA4 中数据的描述

自动映射

自动映射包含由 ABO 创建的符号的偏移量和大小（以十六进制表示）。这些偏移量相对于原始 DSA 末尾处建立的基准。自动映射不显示原始程序中的自动项或者原始编译器创建的临时项。ABO 将建立通用寄存器 (GPR) 以包含“新” DSA 的起点偏移量。将使用此新寄存器作为基准来引用新创建的所有自动项。

示例：

(1)	(2)	(3)
	*****	A U T O M A T I C M A P *****
OFFSET (HEX)	LENGTH (HEX)	NAME
0	4	_GPR0
4	4	_GPR1
8	4	_GPR2
C	4	_GPR3
10	4	_GPR4
14	4	_GPR5
18	4	_GPR6
1C	4	_GPR7
20	4	_GPR8
24	4	_GPR9
28	4	_GPR10
2C	4	_GPR11
30	4	_GPR12
34	4	_GPR13

下面描述了自动映射的三个部分：

- 1. 堆栈符号相对于新堆栈起始位置的十六进制偏移量
- 2. 符号的字节长度（以十六进制表示）
- 3. 符号的名称

输入指令

此部分包含输入模块的完整指令列表。对于优化的每个 CSECT，都会提供一个部分。这些输入指令与[优化指令部分](#)中显示的指令相同。在此部分中，显示的输入指令中穿插了对应的优化指令，所以这些并非完整的有序列表。

输入指令部分从用于编译的 COBOL 编译器版本和从 CSECT 抽取的“签名信息字节”开始。《COBOL 编程指南》中记录了“Signature information bytes”，其中提供了有关已编译的程序的信息。将对这些信息字节进行解码，并打印相应的生效编译器选项。请注意，解码的编译器选项与显示在原始编译器列表中的内容和格式可能不完全匹配。这是因为 ABO 仅根据输入 CSECT 中存在的签名信息字节而不是原始编译过程中指定的完整原始源和选项对选项进行解码。

在以下示例中，优化了名为 PROGB 的 COBOL CSECT。

(1)	(2)	(3)
*****	I N P U T	I N S T R U C T I O N S *****
id 5655S7100 v42 m00 2018005 resident PROGB		
Enterprise COBOL V4: start=0x10, length=9.15 (kBytes)		
Signature information bytes:		
a0487dcc 20000000 00880108 00000040		
08000000 00000008 00f00114 00		
Compiler Options in effect:		
ADV		
ARITH(COMPAT)		
NOAWO		
NOBLOCK0		
Compilation unit is a program.		
NOCICS		
NOCURRENCY		
Default DDNAME for OUTDD will be used		
DATA(31)		
NODATEPROC		
DBCS		
NODECK		
NODLL		

```

NODUMP
NODYNAM
NOEXPORTALL
NOFASTSRT
    INTDATE(ANSI)
NOLIB
NOMDECK
NONAME
NONUM
    NUMCLS(PRIM)
    NUMPROC(NOPFD)
    OBJ
NOOFFSET
NOOPTIMIZE
    QUOTE
    RENT
    RMODE(ANY)
    SEQUENCE
    SIZE(MAX)
    SOURCE
NOSQL
    SQLCCSID
NOSSRANGE
    TERM
    TEST(HOOK,NOSEPARATE,EJPD)
NOTHREAD
    TRUNC(STD)
NOVBREF
NOWORD
    XMLPARSE(XMLSS)
    XREF
    YEARWINDOW(1900)
    ZWB

0002BA LR 3,15
0002BC LA 0,440(0,,1)
0002C0 CL 0,12(0,,12)
0002C4 BASR 15,0
0002C6 BC 13,12(0,15)
0002CA L 15,768(0,,12)
0002CE BASR 14,15
0002D0 LR 1,15
0002D2 ST 13,4(0,,1)
0002D6 ST 0,76(0,,1)
0002DA MVC 0(4,1),88(3)
0002E0 XC 132(4,1),132(1)
0002E6 ST 9,92(0,,1)
0002EA LR 13,1
0002EC L 12,232(0,,9)
0002F0 LR 1,2
0002F2 ST 13,88(0,,13)
0002F6 L 10,36(0,,12)
0002FA L 8,300(0,,9)
0002FE STM 14,12,12(13)
000302 MVC 136(4,13),16(10)
000308 LA 2,280(0,,13)
00030C ST 2,140(0,,13)
000310 ICM 2,15,364(9)
000314 L 11,40(0,,12)
000318 BC 8,214(0,11)
00031C L 3,92(0,,9)
000320 L 15,244(0,,3)
000324 LA 1,271(0,,10)
000328 BASR 14,15
00032A A 2,0(0,,12)
00032E ST 2,364(0,,9)
000332 MVC 256(4,13),252(13)
000338 LA 2,240(0,,11)
00033C ST 2,252(0,,13)
000340 BC 15,84(0,11)
000344 MVC 252(4,13),256(13)
00034A MVC 260(4,13),248(13)
000350 L 11,40(0,,12)
000354 LA 2,268(0,,11)
000358 ST 2,248(0,,13)
00035C BC 15,0(0,11)
000360 MVC 248(4,13),260(13)
000366 TM 87(9),64
00036A L 11,40(0,,12)
00036E BC 1,306(0,11)
000372 TM 84(9),32
000376 BC 14,298(0,11)

```

00037A	OI	132(13),32
00037E	OI	87(9),64
000382	BC	15,306(0,11)
000386	OI	348(9),64
00038A	OI	132(13),1
00038E	BAS	14,316(0,,9)
000392	BAS	14,318(0,,9)
000396	AP	0(8,8),16(8,8)
00039C	ZAP	0(8,8),0(8,8)
0003A2	BAS	14,318(0,,9)
0003A6	ZAP	16(8,8),0(8,8)
0003AC	BAS	14,318(0,,9)
0003B0	CP	0(8,8),114(2,10)
0003B6	L	11,40(0,,12)
0003BA	BC	7,392(0,11)
0003BE	BAS	14,316(0,,9)
0003C2	BAS	14,318(0,,9)
0003C6	L	2,92(0,,9)
0003CA	L	15,44(0,,2)
0003CE	LA	1,259(0,,10)
0003D2	BASR	14,15
0003D4	L	11,40(0,,12)
0003D8	BC	15,414(0,11)
0003DC	BAS	14,316(0,,9)
0003E0	BAS	14,318(0,,9)
0003E4	L	2,92(0,,9)
0003E8	L	15,44(0,,2)
0003EC	LA	1,247(0,,10)
0003F0	BASR	14,15
0003F2	BAS	14,316(0,,9)
0003F6	BAS	14,318(0,,9)
0003FA	CP	16(8,8),114(2,10)
000400	L	11,40(0,,12)
000404	BC	7,466(0,11)
000408	BAS	14,316(0,,9)
00040C	BAS	14,318(0,,9)
000410	L	2,92(0,,9)
000414	L	15,44(0,,2)
000418	LA	1,235(0,,10)
00041C	BASR	14,15
00041E	L	11,40(0,,12)
000422	BC	15,488(0,11)
000426	BAS	14,316(0,,9)
00042A	BAS	14,318(0,,9)
00042E	L	2,92(0,,9)
000432	L	15,44(0,,2)
000436	LA	1,223(0,,10)
00043A	BASR	14,15
00043C	BAS	14,316(0,,9)
000440	BAS	14,318(0,,9)
000444	BAS	14,316(0,,9)
000448	L	11,40(0,,12)
00044C	BC	15,534(0,11)
000450	BAS	14,316(0,,9)
000454	TM	132(13),32
000458	BC	14,534(0,11)
00045C	L	2,92(0,,9)
000460	L	15,244(0,,2)
000464	LA	1,205(0,,10)
000468	BASR	14,15
00046A	L	2,364(0,,9)
00046E	S	2,0(0,,12)
000472	ST	2,364(0,,9)
000476	NI	348(9),191
00047A	TM	85(9),64
00047E	BC	14,572(0,11)
000482	LA	1,8(0,,0)
000486	L	2,92(0,,9)
00048A	L	15,32(0,,2)
00048E	BASR	14,15
000490	TM	84(9),40
000494	BC	7,602(0,11)
000498	L	2,296(0,,9)
00049C	LH	15,8(0,,2)
0004A0	L	13,4(0,,13)
0004A4	L	14,12(0,,13)
0004A8	LM	0,12,20(13)
0004AC	BCR	-1,14
0004AE	MVC	264(12,13),177(10)
0004B4	L	3,296(0,,9)
0004B8	LH	2,8(0,,3)
0004BC	ST	2,276(0,,13)

```
0004C0    LA      1,264(0,,13)
0004C4    L        2,92(0,,9)

*****  E N D   O F   I N P U T   I N S T R U C T I O N S   * * * * *
```

下面描述了输入指令部分的三个部分：

- 1. 原始指令的输入 CSECT 中的十六进制偏移量
- 2. 指令助记符
- 3. 指令操作数

SYSPRINT DD 和 LIST 选项

可使用 SYSPRINT DD 或 LIST 选项来指定已生成的列表转换的位置。

SYSPRINT 或 LIST 的目标可以为以下项之一：

- 顺序数据集，或者 PDSE（而非 PDS）的成员。可以按照优化顺序向此顺序数据集添加多个 CSECT 优化的输出。
- PDS 或 PDSE。在优化 CSECT 时，特定于该 CSECT 的列表转换会放在 PDS 或 PDSE 的成员中，该成员名称基于 CSECT 名称（大写且截断为 8 个字符）。将覆盖该成员的内容（如果有），即使原来的内容是由 ABO 在先前调用中生成的也是如此。
- HFS 路径。可以向此 HFS 文件添加多个 CSECT 优化的输出。

LIST 选项优先于 SYSPRINT DD。如果指定了 LIST 选项，那么它将覆盖 SYSPRINT DD。在指定 LIST 选项时，您可以省略 SYSPRINT DD 名称。

注：

ABO 列表包含详细的变换信息，因此可以变得非常大。将 SYSPRINT DD 目标指定为 SYSOUT 可能会导致 JES2 假脱机达到系统指定的行限制。当达到假脱机行限制时，JES2 会将控制权传递给安装出口例程，但 ABO 作业可能会终止也可能不会终止。虽然可以使用 JOBPARM L 选项来增大假脱机行限制，但最大 L 设置 999999 可能仍不足以容纳 ABO 列表。

为避免此问题，建议 SYSPRINT 指定 PDS、PDSE 或 HFS 位置，如本部分中所记录。

示例

以下 JCL 示例在 SYSPRINT DD 中使用 PDSE，从而将列表转换写入到 PDSE 的成员。

```
//SYSIN DD *
BOPT IN=HLQ.IN.LOAD(MOD*) OUT=HLQ.OUT.LOAD
...
//SYSPRINT DD DSN=HLQ.LIST.PDSE, DISP=SHR
```

在此示例中，输入程序模块指定为 HLQ.IN.LOAD(MOD*)，这意味着优化 HLQ.IN.LOAD 中名称以“MOD”开头的所有符合条件的成员。

输入数据集中有两个成员：MOD1 和 MOD2。这两个程序模块中提供不同的 CSECT：

表 8. 输入模块及其包含的 CSECT	
HLQ.IN.LOAD	CSECT
MOD1	PROG1A
	PROG1B
	PROG1C
MOD2	PROG2A
	PROG2B

ABO 将优化其中每个 CSECT，每次一个，并且将针对每个 CSECT 生成两个输出：

1. 优化的 CSECT
2. CSECT 的列表转换

优化的 CSECT 具有与输入 CSECT 相同的名称，并且会将优化的 CSECT 放在与输入程序模块的成员名称相同的程序模块中。但是，新的程序模块将放在名为“*HLQ.OUT.LOAD*”的新 PDSE 中。

表 9. 输出 1：优化模块及其 CSECT	
<i>HLQ.OUT.LOAD</i>	CSECT
MOD1	PROG1A
	PROG1B
	PROG1C
MOD2	PROG2A
	PROG2B

针对已优化的每个 CSECT 生成的列表将作为单独成员放在 PDSE “*HLQ.LIST.PDSE*” 中。所有这类 PDSE 成员都具有与输入 CSECT 名称相同的名称。结果是 *HLQ.LIST.PDSE* 将有 5 个成员：PROG1A、PROG1B、PROG1C、PROG2A 和 PROG2B。

表 10. 输出 2：列表转换	
<i>HLQ.LIST.PDSE</i>	
PROG1A	
PROG1B	
PROG1C	
PROG2A	
PROG2B	

第 7 章 管理优化和优化模块部署流程

在使用 ABO 时采用分阶段的迭代方法

在使用 ABO 时建议采用分阶段的迭代方法来平衡优化流程的成本与运行 ABO 生成的优化程序的优势。

例如，首先优化作为前 x% 的 CPU 时间贡献者的模块。度量使用这些 ABO 生成的模块的影响（例如，CPU 时间减少），然后针对随后 x% 的 CPU 贡献者重复此过程，直至满足性能目标。

性能度量工具和报告工具（例如，IBM Application Performance Analyzer (APA) for z/OS）可帮助确定排名前列的 CPU 贡献者。如果性能度量工具不可用，那么可使用 ABO 随附的 [RTI Profiler](#) 来帮助确定在运行应用程序时最常执行的 COBOL 模块。

从 ABO 受益最多的程序的特征

与其他程序相比，某些已编译程序从 ABO 受益更多。掌握这些程序的关键特征还可以帮助在已编译的 COBOL 程序上分阶段使用 ABO。

ABO 只能提高原始编译器生成的代码的性能以及某些精选 Language Environment (LE) 例程的性能，但是在其他子系统（例如，CICS、Db2 和 IMS）中消耗时间时 ABO 可能无法提高性能。

可从 ABO 优化受益更多的程序的关键特征包括：

- 应用程序执行时间中有很大大一部分是用于 COBOL 代码上，而不是用于其他子系统（例如，CICS、Db2 和 IMS）上。
 - COBOL 代码将执行大量计算。例如，在程序中，COBOL 代码自身执行实际工作，而不是简单充当其他程序或子系统的“驱动程序”。
 - 在源代码级别，很可能受益的语句包括但不限于：COMPUTE、IF、MOVE、ADD、SUBTRACT、MULTIPLY、DIVIDE 和 REMAINDER。
 - 此外，ABO 还可优化某些精选的 Language Environment (LE) 例程。这些例程执行各种转换、移动和算术运算，并且包括 IGZCSH2、IGZCFPC、IGZCONV、IGZCVM0、IGZCXPR、IGZCXMU 和 IGZCXDI。ABO 通过直接在优化代码中更高效地执行这些例程的工作或者调用更高效的 LE 例程来优化这些例程。
- 注：**单独查看 COBOL 源代码时不会考虑应用程序时间实际用在哪些地方，因此，应结合分析工具（例如，APA）提供的性能报告来执行此操作。
- 应用程序中的大多数 COBOL 模块均可通过 ABO 进行优化。这意味着使用合格的 COBOL 编译器编译模块，并且模块包含 ABO 支持的语言功能。

优化和部署使用场景

本部分包含 IBM Automatic Binary Optimizer for z/OS 的三个典型使用场景。这些场景描述使用 IBM Automatic Binary Optimizer for z/OS 来提高已编译的 IBM COBOL 程序的性能的可能方法。每个场景提供逐步指示信息以支持您优化已编译的 IBM COBOL 程序。

场景 1：使用静态部署的优化流程

在此使用场景中，将使用 BOPT 伪指令在 JCL 中指定优化器的输入模块，并且对于部署，将更新用于标识包含原始模块的数据集的所有现有 JCL。

过程

要使用静态部署执行优化，请完成以下步骤：

1. 创建新的数据集。例如，为此场景创建以下数据集，其中 *HLQ* 是您定义的高级限定符。
 - *HLQ.OUT.LOAD.Z14*。使用瞄准于 z14 机器的优化二进制文件来填充此数据集。

- `HLQ.OUTPUT.LOAD.Z15`。使用瞄准于 `z15` 机器的优化二进制文件来填充此数据集。
2. 运行 ABO 以填充新的数据集。要运行优化器，请创建新的 JCL。在以 `SYSIN` 开头的流内行中，使用 `BOPT` 优化器伪指令。使用 `IN` 选项选择要优化的已编译 COBOL 程序。例如，以下 JCL 指示优化器优化 `HLQ.INPUT.LOAD` 中名称以 `M` 开头的所有成员。针对 `z14` 和 `z15` 的优化二进制文件分别放在 `HLQ.OUTPUT.LOAD.Z14` 和 `HLQ.OUTPUT.LOAD.Z15` 中。

```

...
//SYSIN      DD *
BOPT IN=HLQ.INPUT.LOAD(M*) OUT=HLQ.OUTPUT.LOAD.Z14 LIST=HLQ.OUTPUT.LIST.Z14 ARCH=12    BOPT
IN=HLQ.INPUT.LOAD(M*) OUT=HLQ.OUTPUT.LOAD.Z15 LIST=HLQ.OUTPUT.LIST.Z15 ARCH=13

```

此示例旨在反映用户应在 `SYSIN` 文件中指定的内容。对于基本 JCL 配置，请参阅第 57 页的『附录 A JCL 示例』。要获取可在静态部署场景中使用的更多样本 JCL，请参阅第 25 页的『使用 BOPT 指定优化』。

3. 要运行优化程序，请修改用于运行原始程序的 JCL。此 JCL 标识了包含原始模块的数据集。在 `STEPLIB` 设置中，对于每个目标体系结构，必须将优化模块的数据集放在原始模块的数据集之前。以下片段显示了指向优化程序二进制文件和原始程序二进制文件的 JCL 中的已修改部分。

下面是用于在 `z14` 上运行原始程序的 JCL 中的已修改部分：

```

//STEPLIB DD DSN=HLQ.OUTPUT.LOAD.Z14,DISP=SHR
//          DD DSN=HLQ.INPUT.LOAD,DISP=SHR
...

```

下面是用于在 `z15` 上运行原始程序的 JCL 中的已修改部分：

```

//STEPLIB DD DSN=HLQ.OUTPUT.LOAD.Z15,DISP=SHR
//          DD DSN=HLQ.INPUT.LOAD,DISP=SHR
...

```

场景 2：使用动态部署的优化流程

在此使用场景中，会将输入映射到 `IEFOPZxx SYS1.PARMLIB` 的成员中的输出模块，然后使用 `IEFOPZ` 优化器伪指令来指定输出模块的输入优化。在完成二进制文件优化后，运行优化程序，而不需要对曾用于运行原始程序的现有 JCL 进行任何更改。

关于此任务

`IEFOPZxx` 包含用于定义数据集优化配置的语句（提供了旧的 COBOL 库与目标新库（每个期望的体系结构级别都有一个新库）对的列表），并指定了要处理（优化）的成员。有关更多信息，请参阅 *z/OS MVS Initialization and Tuning Reference*。

过程

要使用动态部署执行二进制文件优化，请完成以下步骤：

1. 创建新的数据集。例如，为此场景创建以下数据集，其中 `HLQ` 是您定义的高级限定符。
 - `HLQ.OUTPUT.LOAD.Z14`。使用瞄准于 `z14` 机器的优化二进制文件来填充此数据集。
 - `HLQ.OUTPUT.LOAD.Z15`。使用瞄准于 `z15` 机器的优化二进制文件来填充此数据集。
2. 定义 `IEFOPZ` 配置。
 - a. 创建 `IEFOPZxx` 成员。
 - b. 对于包含要优化的已编译模块的每个旧数据集，在 `IEFOPZxx` 成员中定义一个 `OLD/NEW` 对。将 `OLD/NEW` 对标记为 `INACTIVE`，使系统不会执行任何意外的 `OLDNEW` 处理。请参阅以下示例：

```

MAXARCH(13)
CHECKALL
OWNER(IBM) MINARCH(12)
OLDNEW (
  OWNER(IBM)
  OLD (DSN(HLQ.INPUT.LOAD))
  NEW (DSN(HLQ.OUTPUT.LOAD.Z14) ARCH(12))
  NEW (DSN(HLQ.OUTPUT.LOAD.Z15) ARCH(13))
)

```

```
INCLUDEMEMBERS(M*) //Identifies to process all members beginning with M
INACTIVE
)
```

注: 可以在一个或多个 IEFOPZxx 成员中定义 OLD/NEW 对。

3. 要激活 IEFOPZ 配置, 请使用以下 z/OS MVS 系统命令:

```
SET IEFOPZ=(x1, ..., xn)
```

其中, $x_1, \dots, x_n$ 是 IEFOPZxx 成员的后缀 xx。如果在上一步骤中只创建了一个成员, 那么命令如下所示:

```
SET IEFOPZ=x1
```

注: SET 命令修改仅停留在当前 IPL 会话中。因此, 它通常用于新配置快速测试, 或在当前 IPL 会话期间覆盖一些永久定义。对于永久配置定义, 请参阅步骤 5。

4. 运行 IBM Automatic Binary Optimizer for z/OS 以填充新数据集。

要运行优化器, 请如下所示编写 JCL。在以 SYSIN 开头的流内数据中, 使用 IEFOPZ 伪指令。

```
//SYSIN DD *
IEFOPZ SEL_ARCH=12 LIST=HLQ.BOZOPT.ARCH12.LIST IEFOPZ SEL_ARCH=13
LIST=HLQ.BOZOPT.ARCH13.LIST
```

此示例旨在反映用户应在 SYSIN 文件中指定的内容。对于基本 JCL 配置, 请参阅第 57 页的『附录 A JCL 示例』。要获取可在动态部署场景中使用的更多样本 JCL, 请参阅第 27 页的『使用 IEFOPZ 指定优化』。

5. 使用 IEFOPZ 系统参数更新 IEASYSxx SYS1.PARMLIB 成员, 以便后续 IPL 正确激活所需的 IEFOPZ 配置。例如, 要使成员 IEFOPZ99 中指定的 IEFOPZ 配置在每次后续 IPL 时自动激活, 请将 IEFOPZ=99 语句放入 IEASYSxx 成员中。但是, 步骤 3 中描述的 SET 命令 (如果已发出, 如下示例所示: SET IEFOPZ=99) 将仅对当前的 IPL 会话激活期望的 IEFOPZ99 成员。
6. 将 OLD/NEW 对重新定义为 ACTIVE。如果要对 JOBLIB 和 STEPLIB 以外的任何 DD 名称执行 OLD/NEW 处理, 请使用 DDNAME 语句在 IEFOPZxx 参数库成员中定义这些名称。然后, 激活这个已更新的 IEFOPZ 配置。
7. 使用曾用于运行原始程序的现有 JCL 来运行优化程序。

相关参考

第 77 页的『相关出版物』

场景 3: 使用混合方法的优化流程

在混合方法中, 在 JCL 中明确指定要优化的输入二进制文件 (如在场景 1 中所做的那样), 但是要结合场景 2 中演示的动态部署。通过动态部署, 运行优化模块而不必更改现有的 JCL。

过程

要使用混合方法执行二进制文件优化, 请完成以下步骤:

1. 创建新的数据集。例如, 为此场景创建以下数据集, 其中 HLQ 是您定义的高级限定符。

- HLQ.OUT.LOAD.Z14。使用瞄准于 z14 机器的优化二进制文件来填充此数据集。
- HLQ.OUT.LOAD.Z15。使用瞄准于 z15 机器的优化二进制文件来填充此数据集。

2. 运行 IBM Automatic Binary Optimizer for z/OS 以填充新数据集。

要运行优化器, 请创建新的 JCL。在以 SYSIN 开头的流内数据中, 使用 BOPT 优化器伪指令来选择要优化的已编译 COBOL 模块。例如, 以下 JCL 指示优化器优化 HLQ.IN.LOAD 中以字母 M 开头的成员。针对 z14 和 z15 的优化二进制文件分别放在 HLQ.OUT.LOAD.Z14 和 HLQ.OUT.LOAD.Z15 中。

```
//SYSIN DD *
BOPT IN=HLQ.IN.LOAD(M*) OUT=HLQ.OUT.LOAD.Z14 ARCH=12 BOPT IN=HLQ.IN.LOAD(M*)
OUT=HLQ.OUT.LOAD.Z15 ARCH=13
```

此示例旨在反映用户应在 SYSIN 文件中指定的内容。对于基本 JCL 配置，请参阅第 57 页的『附录 A JCL 示例』。要获取可在混合场景中使用的更多样本 JCL，请参阅第 25 页的『使用 BOPT 指定优化』。

3. 定义 IEFOPZ 配置。

- a. 创建 IEFOPZxx 成员。
- b. 对于包含要优化的已编译模块的每个旧数据集，在 IEFOPZxx 成员中定义一个 OLD/NEW 对。将 OLD/NEW 对标记为 ACTIVE。请参阅以下示例。

```
OLDNEW (
  OLD( DSNNAME (HLQ.IN.LOAD) )
  NEW( DSNNAME (HLQ.OUT.LOAD.Z14) ARCH(12) )
  NEW( DSNNAME (HLQ.OUT.LOAD.Z15) ARCH(13) )
  INCLUDEMEMBERS(M*) //Identifies to process all members beginning with M
  ACTIVE )
```

注：可以在一个或多个 IEFOPZxx 成员中定义 OLD/NEW 对。

- c. 如果要对 JOBLIB 和 STEPLIB 以外的任何 DD 名称执行 OLDNEW 处理，请使用 DDNAME 语句在 IEFOPZxx 参数库成员中定义这些名称。然后，使用以下命令激活这个已更新的 IEFOPZ 配置：

```
SET IEFOPZ=(x1, ..., xn)
```

其中， $x_1, \dots, x_n$ 是 IEFOPZxx 成员的后缀 xx。如果在上一步骤中只创建了一个成员，那么命令如下所示：

```
SET IEFOPZ=x1
```

4. 使用曾用于运行原始程序的现有 JCL 来运行优化程序。

相关参考

第 77 页的『相关出版物』

测试信息

ABO 生成的优化模块的运行速度更快但具有与原始 COBOL 模块相同的行为，但某些独立错误消息和异常终止代码差别除外。ABO 之所以能够做到这一点，是因为其处理 COBOL 模块中的二进制代码，因此能够确保较低级别的程序逻辑仍保持相同。这意味着 ABO 用户不必执行 ABO 优化模块的完整功能的验证测试。建议执行一些有限测试，以在将 ABO 优化模块部署到生产环境之前，确保使用 ABO 优化模块的应用程序基本能够正常运行。

性能测试最好在受控环境中完成，并且使用与原始应用程序相同的输入数据以及使用包含 ABO 优化模块的应用程序。将运行尽可能少的其他应用程序的机器或 LPAR 用于性能测试能够获取可重现的稳定性能结果。比较原始应用程序与优化应用程序之间的 CPU 时间是查看性能改进的最佳方法。

第 8 章 解决在执行优化和部署优化模块时遇到的问题

解决在优化期间发生的问题

ABO 传递到 z/OS 的返回码是优化期间是否遇到问题的指示符。返回码值 0 意味着优化成功且未遇到任何问题。0 之外的返回码值指示发生某些意外或遇到问题。有关返回码的更多信息，请参阅第 59 页的『附录 B 返回码』。

ABO 会生成一些可用于诊断问题的输出文件。

以下文件可帮助诊断在优化 COBOL 程序期间遇到的问题：

- 日志文件提供已优化或扫描的内容的摘要以及错误消息（如果适用）。有关更多信息，请参阅[消息](#)。如果在优化过程中检测到问题，那么应首先查看日志文件。
- 在异常情况下，会写入到 [OPTERR DD](#) 指定的文件。如果未指定 OPTERR DD，那么会将这些消息写入到 JOBLOG。
- 某些情况下，例如，在运行 ABO 时发生程序异常，会写入到 [CEEDUMP DD](#) 指定的文件。CEEDUMP 文件由 Language Environment (LE) 生成，并且包含诸如在异常终止时执行的过程的追溯之类的信息。
- JOBLOG 包含补充错误消息（写入其他位置）的其他诊断消息，或者 JOBLOG 可以是异常情况下所遇到错误的缺省位置。

解决在执行期间遇到的问题

常见执行错误的解决方案

如果安装的“语言环境 Automatic Binary Optimizer 运行时引擎”不是可用的最新 PTF，那么 ABO 优化模块可能发生故障并显示 U4038 异常终止。在此情况下，该模块将输出以下消息之一：

```
IGZ0153S 程序 B0ZSRC1 已使用需要在“语言环境”上安装服务的编译器级别进行了编译。IGZ0355S 程序 B0ZSRC1 已使用需要在“语言环境”上安装服务的 Automatic Binary Optimizer 级别进行了优化。
```

z/OS 2.2 和 2.3 上的 PTF 将导致发出第一条消息，z/OS 2.4 上的 PTF 将导致发出第二条消息。要解决此问题，应安装程序目录中列出的最新“语言环境 Automatic Binary Optimizer 运行时引擎”PTF。也可以在[修订列表和新功能部件页面](#)上找到有关 ABO PTF 的最新信息。

如果尝试在 ABO 不支持的系统上运行 ABO 生成的模块，那么也会发生 OC1 异常终止。请参阅第 4 页的『[目标硬件级别](#)』，以了解受支持的系统。

执行错误诊断

IBM 在 Application Delivery Foundation for z Systems 中提供的问题确定工具可用于确定包含 ABO 优化模块的应用程序中执行时问题的根源。如果问题确定工具不可用，那么 ABO 生成的列表转换可帮助诊断执行时问题。

如果诊断确定是 ABO 优化模块导致执行时问题，请还原为原始 COBOL 模块并联系 IBM 服务人员以报告问题。要了解为向 IBM 报告 ABO 问题而需要收集的信息，请参阅 [ABO Mustgather](#) 页面。

优化后的 COBOL 模块大小变更

由于 ABO 为提高性能所执行的优化类型，因此优化模块的大小通常大于原始模块。

下面是导致模块大小增大的一些常见原因：

- 使用较高版本的 ARCH 指令，此类指令的长度通常为 6 字节，这不同于许多较低版本的 ARCH 指令（其长度通常为 4 字节或 2 字节）。例如：
 - 针对压缩/分区十进制运算使用十进制浮点 (DFP) 来提高性能
 - 将原始模块中的“基本定位器”指针替换为更有效但更大的长位移指令
 - 使用多个“立即移动”指令代替一个“内存中移动”指令
- ABO 中有多个优化生成的代码更多，但路径长度更短且性能更好。例如：
 - 更有效地处理已编辑的数字变量
 - 展开长距离移动和比较操作，而不是使用更短但慢很多的指令
 - 有条件地更正二进制数据的小数位精度
- 直接插入各种运行时库例程的行为会导致优化模块中生成的代码更多，但在许多情况下性能要快得多。

由于上述原因以及其他类似原因，ABO 生成的优化模块通常比原始模块更大并且需要更多的磁盘存储空间。但是，优化程序本身在运行时所使用的内存量与原始模块所使用的内存量相同。运行优化程序时，有时会观察到略高的 EXCP 计数，但这只是因为装入较大模块所需的额外 I/O 操作很少所致。

请注意，如果优化代码恰巧小于原始代码，那么优化模块的大小将保持不变。

错误信息和异常终止代码差别

在几乎所有情况下，ABO 生成的优化模块在功能上等同于对应的原始模块。但是，在某些罕见情况下，ABO 生成的模块将产生与原始模块不同的 Language Environment (LE) 运行时消息或不同的 CICS 异常终止代码。

这可能在以下情况下发生：为提高处理效率，ABO 在生成的代码中内嵌或优化大数据项除法和其他复杂运算，而非由 LE 库例程或低效率机器指令进行处理。

在非 CICS 应用程序中，ABO 生成的模块：

- 可生成定点除法异常 (CEE3209S) 消息（如果原始模块生成十进制除法异常 (CEE3211S) 或 IGZ0061S 消息）。
- 可生成十进制除法异常 (CEE3211S)（如果原始模块生成 IGZ0061S 消息）。

为参考起见，以下提供针对这些不同异常的完整 LE 运行时消息文本。

CEE3211S 系统检测到十进制除法异常（系统完成代码 = 0CB）。IGZ0061S 在程序“program-name”位移“displacement”处发生被零除。
CEE3209S 系统检测到定点除法异常（系统完成代码 = 0C9）。

对于 CICS 应用程序，“CICS ASSIGN ABCODE”返回的异常终止代码可从原始模块的“1061”更改为 ABO 所生成模块的“ASRA”。

Application Delivery Foundation for z Systems

您可以在 ABO 生成的模块上使用 Application Delivery Foundation for z Systems (ADFz)。

在 <https://www.ibm.com/ca-en/marketplace/app-delivery-foundation-on-zsystems> 查看有关 Application Delivery Foundation for z Systems 的更多信息。

以下 Application Delivery Foundation for z Systems 系列的问题确定工具可用于 ABO 生成的 COBOL 模块：

- Developer for z Systems Enterprise Edition，包含 IBM Debug for z Systems（以前称为 IBM Debug Tool (DT) for z/OS）
- Fault Analyzer for z/OS (FA)
- Application Performance Analyzer for z/OS (APA)

为了更有效地使用这些工具，您需要针对每个优化程序生成一个 LANGX 端文件。DT、FA 和 APA 利用该 LANGX 端文件来提供更好的工具体验。例如，在提供 LANGX 端文件时，Debug for z Systems 会提供源代码级别调试。如果未提供该 LANGX 端文件，那么源代码级别调试将不可用。

创建 LANGX 端文件

IPVLANGO 是 IBM Problem Determination Tools Common Component for z/OS V1.7 随附的一款新工具，与 Application Delivery Foundation for z/OS 工具共享。IPVLANGO 将原始已编译程序的 SYSDEBUG 数据集、编译器列表或 LANGX 端文件与 ABO 列表转换进行合并，以生成适合 ABO 所生成模块的新 LANGX 端文件。对 ABO 生成的模块使用 DT、FA 或 APA 时，会使用该新的 LANGX 文件。

Run Time Instrumentation Profiler

IBM Run Time Instrumentation (RTI) Profiler 是用于收集和报告批处理 z/OS 应用程序的执行时 CPU 性能特征的性能分析工具。

简介

RTI Profiler 使用 zEC12 中添加的 Runtime Instrumentation Facility，以较低的开销收集有关批处理程序 CPU 性能特征的高保真信息。

可使用 RTI Profiler 的受支持的 z/OS 版本包括：

- z/OS V2.4
- z/OS V2.3
- z/OS V2.2

概要分析应用程序中的所有 Language Environment (LE) CSECT，以全面了解整体 CPU 性能。这包括 IBM COBOL、C/C++ 和 PL/I 编译器编译的程序以及 ABO 优化的 COBOL 程序。同时还会收集并报告在任何 LE 库例程中花费的时间。

RTI Profiler 的输出是一个文本文件，其中包含每个已编译或已优化 CSECT 的详细信息以及此 CSECT 中的偏移量（运行时程序在此花费时间）。

RTI Profiler 输出可与要概要分析的程序的对应列表文件一起帮助确定程序的具体部分（向下一直到机器指令），从而有机会提高应用程序 CPU 性能。

由于生成了很详细的信息，所以 RTI Profiler 的主要用途是为 IBM 支持和开发人员提供数据来帮助进行性能调查。在此场景中，IBM 支持或开发代表请求使用 RTI Profiler，并将其输出连同来自程序的原始编译或优化的其他工件（例如，列表文件）一起发送给 IBM。

如果需要更全面且更具体的应用程序概要分析，那么建议使用概要分析工具，例如，[Application Performance Analyzer for z/OS in ADFz](#)。

系统需求和限制

只可以在 zEC12、zBC12、z13、z13s、z14、z14 ZR1 和 z15 系统上使用 RTI Profiler。存在此限制的原因是 RTI Profiler 使用了仅从 zEC12 系统开始添加的 Runtime Instrumentation Facility。

如果 z/OS 操作系统是在 z/VM® 上运行，那么 RTI Profiler 无法工作。如果是在 z/VM 访客上运行，那么会将消息 RISTART: AUTH REQUEST FAILED 输出到作业日志。

可以在包含 IMS 批处理的所有批处理应用程序以及与 Db2® 交互的批处理程序上使用 RTI Profiler。

RTI Profiler 不可用于 CICS 应用程序，并且不支持用于属于非批处理 IMS 环境的任何应用程序。

使用说明

RTI Profiler 由安装 ABO 的相同数据集中包含的 BOZBXITA 和 BOZRIDT 成员组成。

- **BOZBXITA**: 将 CEEBXITA 和相关的概要分析例程链接到应用程序的主程序。此步骤可启动和停止概要分析，以及在程序执行期间监视和管理 RTI Profiler 的缓冲区。
- **BOZRIDT**: 处理 RTI Profiler 缓冲区数据并生成文本文件报告。

在使用 RTI Profiler 之前，您必须分配 PDS 或 PDSE 数据集以保存概要分析结果。RTI Profiler 生成的报告存储在此数据集的一个成员中。在以下 JCL 示例中，此数据集名为 HLQ.SYSPROFD。

下表显示了建议的 HLQ.SYSPROFD 分配参数。

表 11. 建议的分配参数	
数据集	建议的分配参数
HLQ.SYSPROFD（作为 PDS）	Space units : CYLINDER Primary quantity: 10 Secondary quantity: 10 Directory blocks: 10 Record format: FB Record length : 80 Block size : 27920 Data set name type PDS
HLQ.SYSPROFD（作为 PDSE）	Space units : CYLINDER Primary quantity: 10 Secondary quantity: 10 Directory blocks: 10 Record format: FB Record length : 80 Block size : 27920 Data set name type LIBRARY

要使用 RTI Profiler，请执行以下步骤：

1. 在链接编辑步骤中，重新绑定现有程序以包含 **BOZBXITA**
2. 在执行步骤中，指定将保存概要分析结果的数据集的位置

在以下步骤和 JCL 示例中，\$HLQBOZ.BOZ210.SBOZMOD1 是为 ABO 选择的安装位置。

步骤 1：重新绑定现有程序以包含 **BOZBXITA**

第一步是重新绑定现有程序以包含 **BOZBXITA**，从而在运行程序时启用 RTI Profiler。

要执行此重新绑定操作，请修改包含应用程序主入口点的程序的链接编辑步骤：

- 将 \$HLQBOZ.BOZ210.SBOZMOD1 作为 SYSLIB 添加到链接编辑步骤
- 包含 **BOZBXITA** 作为链接编辑步骤的附加输入

下面是此步骤的 JCL 示例：

```
//LKED EXEC PGM=IEWL,PARM=$PARM
//SYSLIB DD DISP=SHR,DSN=$HLQBOZ.BOZ210.SBOZMOD1 <-- add $HLQBOZ.BOZ210.SBOZMOD1 as SYSLIB
//          DD DSN=$HLQCEE.SCEELKED,DISP=SHR
//          DD DSN=$HLQCEE.SCEELKEX,DISP=SHR
//LOAD DD DISP=SHR,DSN=$LOAD
//SYSLMOD DD DISP=SHR,DSN=$SYSLMOD($PROG)
//SYSPRINT DD SYSOUT=*
//SYSLIN DD *
//          INCLUDE LOAD($PROG)
//          INCLUDE SYSLIB(BOZBXITA)                                <-- add INCLUDE for BOZBXITA
//          ENTRY $PROG
//          NAME $PROG(R)
```

步骤 2：指定将保存概要分析结果的数据集的位置

第二步是指定将保存 RTI Profiler 生成的概要分析结果的数据集的位置。

在用于执行程序的现有 JCL 中：

- 将 \$HLQBOZ.BOZ210.SBOZMOD1 添加到现有 STEPLIB

· 添加 DDNAME SYSPROFD 以接收概要分析结果

下面是此步骤的 JCL 示例：

```
//GO EXEC PGM=$PROG,REGION=0M
//STEPLIB DD DSN=$SYSLMOD,DISP=SHR <-- $SYSLMOD is the output dataset from
the rebind in step 1
// DD DSN=$HLQB0Z.B0Z210.SB0ZMOD1,DISP=SHR <-- add $HLQB0Z.B0Z210.SB0ZMOD1 to STEPLIB
//SYSPROFD DD DSN=$HLQ.SYSPROFD($PROG),DISP=SHR <-- add SYSPROFD DD
```

程序执行完毕时，概要分析结果将包含在 HLQ.SYSPROFD(\$PROG) 中。

如果未在执行步骤中添加 SYSPROFD，那么会在作业日志中生成消息 RIDATA: OPENING SYSPROFD FAILED。另外，将生成异常终止代码 ABEND=S000 U1130 REASON=00000000。

注：

1. 对于步骤 1，当重新绑定以使用 RTI Profiler 时，必须使用与原始模块相同的链接选项集。要验证是否使用了相同的链接选项集，您可以在原始模块和重新绑定模块上使用 AMBLIST。COBOL 程序不匹配最可能的原因是在重新绑定时无意中将 AMODE 从 24 更改为了 31。下面是一个 JCL 示例，说明如何设置 AMODE=24 的链接选项。

```
//LKED EXEC PGM=IEWL,PARM='LIST,MAP,AMODE=24'
```

2. 如果使用 VS COBOL II 或 ABO 优化的 VS COBOL II 编译程序来编译要进行概要分析的程序，那么可能需要额外的步骤来重新绑定模块以将引导程序例程 IGZEBST 替换为 LE 中的当前版本。如果您在步骤 1 之后未看到 SYSPROFD 位置处的任何概要分析输出，并且尝试对 VS COBOL II 或 ABO 优化的 VS COBOL II 编译程序进行概要分析，请遵循下面的步骤来替换 IGZEBST 例程并启用模块以进行 RTI 概要分析。

```
//LKED EXEC PGM=IEWL,PARM=$PARM
//SYSLIB DD DISP=SHR,DSN=$HLQB0Z.B0Z210.SB0ZMOD1 <- location of B0ZRIDT and B0ZBXITA (i.e.
ABO install location)
// DD DSN=$HLQCEE.SCEELKED,DISP=SHR
// DD DSN=$HLQCEE.SCEELKEX,DISP=SHR
//LOAD DD DISP=SHR,DSN=$LOAD <- original module that will be linked with new IGZEBST and
BOZBXITA
//SYSLMOD DD DISP=SHR,DSN=$SYSLMOD($PROG) <- output module location
//SYSPRINT DD SYSOUT=*
//SYSLIN DD *
INCLUDE SYSLIB(IGZEBST) <- new bootstrap to include from CEE
INCLUDE LOAD($PROG) <- original member from LOAD to link new parts into
REPLACE -IMMED,IGZEBST <- bootstrap member to replace
INCLUDE SYSLIB(BOZBXITA) <- RTI prof member to include
ENTRY $PROG
NAME $PROG(R)
//*
```


附录 A JCL 示例

IBM Automatic Binary Optimizer for z/OS 安装包中包含以下 JCL 示例。

```
//BOZJCLE JOB <job parameters>
//*****
//* Job Name: BOZJCLE *
//* *
//* Licensed Materials - Property of IBM *
//* 5697-AB2 *
//* Copyright IBM Corp. 2019 *
//* *
//* US government users restricted rights *
//* use, duplication or disclosure restricted *
//* by GSA ADP schedule contract with IBM Corp. *
//* *
//*****
//OPT EXEC PGM=BOZOPT,REGION=0M
//STEPLIB DD DSN=$HLQBOZ.BOZ210.SBOZMOD1,DISP=SHR
//OPTLOG DD DSN=$HLQ.BOZOUT.OPTLOG($BOZJOBID),DISP=SHR
//OPTERR DD DSN=$HLQ.BOZOUT.OPTERR($BOZJOBID),DISP=SHR
//CEEDUMP DD DSN=$HLQ.BOZOUT.CEEDUMP($BOZJOBID),DISP=SHR
//SYSPRINT DD DSN=$HLQ.BOZOUT.LISTING,DISP=SHR
//SYSBIN DD DSN=$SYSBIN,DISP=SHR
//SYSBOUT DD DSN=$SYSBOUT,DISP=SHR
//SYSIN DD *
ARCH=$ARCH
BOPT IN=DD:SYSBIN($MEMBER) OUT=DD:SYSBOUT($MEMBER)
```

在此 JCL 示例中，\$HLQBOZ.BOZ210.SBOZMOD1 是为优化器选择的安装位置。

此示例需要预先分配以下数据集：

- HLQ.BOZOUT.OPTLOG
- HLQ.BOZOUT.OPTERR
- HLQ.BOZOUT.LISTING
- HLQ.BOZOUT.CEEDUMP

您可以使用下表中建议的参数来分配这些数据集：

表 12. 建议的分配参数	
数据集	建议的分配参数
HLQ.BOZOUT.OPTLOGHLQ.BOZOUT.OPTERRHLQ.B OZOUT.LISTING	Space units: CYLS Primary quantity: 50 Secondary quantity: 50 Directory blocks: 10 Record format: VB ¹ Record length: 512 Block size: 27998 Data set name type: Library
HLQ.BOZOUT.CEEDUMP	Space units: CYLS Primary quantity: 10 Secondary quantity: 10 Directory blocks: 10 Record format: FB Record length: 133 Block size: 27930 Data set name type: Library

表 12. 建议的分配参数 (续)	
数据集	建议的分配参数
注: 1. <i>HLQ.BEZOUT.OPTLOG</i> 和 <i>HLQ.BEZOUT.OPTERR</i> 必须具有记录格式 VB 才能成功打开。对于 <i>HLQ.BEZOUT.LISTING</i>，也允许使用记录格式 FB，但如果记录长度太短，将从列表中截断信息。建议记录长度至少为 133，以确保不会发生截断。	

\$BOZJOBID 是用户为此作业选择的唯一标识。它用作 *HLQ.BEZOUT.OPTLOG*、*HLQ.BEZOUT.OPTERR* 和 *HLQ.BEZOUT.CEEDUMP* 数据集中的成员名称。\$BOZJOBID 必须是有效的成员名称。

此 JCL 样本显示了用于优化单个模块的 SYSIN DD 的定义。有关更多示例，请参阅第 25 页的『JCL 示例』和第 47 页的『优化和部署使用场景』。有关该示例中使用的 DD 名称的描述，请参阅第 13 页的『必需的 DD 语句』。

附录 B 返回码

IBM Automatic Binary Optimizer for z/OS 发出消息以提供信息、提供可能的警告或报告错误。每条消息都在第 61 页的『附录 C 消息』中记录一个“消息返回码”。在终止时，ABO 将返回码值传递到 z/OS，这是发出的所有消息的“消息返回码”值的最大值。如果未发出任何消息，那么会将返回码值 0 返回到 z/OS。

表 13. IBM Automatic Binary Optimizer for z/OS 返回码	
返回码（十进制）	描述
0	成功完成所有处理。可能已发出一条或多条参考消息。
4	成功完成，但是检测到异常条件。已发出一条或多条警告消息。
12	处理 BOPT 或 IEFOPZ 伪指令或全局选项期间检测到错误。已发出一条或多条消息。 · 如果在输入行的语法处理期间发生错误，那么将拒绝其余行并且 ABO 继续处理输入的下一行。 · 如果在处理 BOPT 或 IEFOPZ 伪指令时发生错误，那么 ABO 继续处理 BOPT 或 IEFOPZ 伪指令的下一个适用模块。如果没有其他输入模块要处理 BOPT 或 IEFOPZ 伪指令，那么伪指令处理将终止，并且 ABO 继续至输入的下一行以处理下一个伪指令或者终止（如果没有更多输入行）。
16	检测到不可恢复错误。发出一条或多条消息，并且 ABO 立即终止处理。

附录 C 消息

此部分中描述的消息将写入到 OPTLOG DD。在某些例外情况中，可能无法将消息写入到 OPTLOG DD，在此情况下，将消息写入到 OPTERR DD 而不是 JOBLOG。以下列出的每条消息具有一个“消息返回码”，用于确定返回到 z/OS 的返回码，如第 59 页的『附录 B 返回码』中所述。

此部分中的每个 ABO 消息都具有 BOZnnnnX 格式，其中 BOZ 指示消息是 ABO 消息，nnnn 是消息号，X 是严重性指示符。

严重性指示符可以是下列任何一项：I、W、E、S 或 U。

- I** 参考消息 (RC=0)
- W** 警告消息 (RC=4)
- E** 错误消息 (RC=8)
- S** 严重错误消息 (RC=12)
- U** 不可恢复的错误消息 (RC=16)

BOZ1003U 程序捕获信号 &1，退出并且返回码为 16。

说明：
优化器无法继续，因为处理期间遇到意外情况。

系统操作：
优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应：
由于先前的问题，可能发生意外问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

16

BOZ1031S 尝试打开“&1”时发生错误。

说明：
优化器无法打开 &1 指定的文件。

系统操作：
如果打开故障与一个必需的优化器 DD 相关联，例如，SYSIN DD，那么优化器立即以返回码 16 终止。否则，如果在 SYSIN 输入文件的行上（在全局选项或者 BOPT 或 IEFOPZ 伪指令中）指定文件，那么行处理将终止，并且优化器继续处理 SYSIN 输入文件的下一行。

用户响应：
确保文件名正确，并且文件已分配且具有相应的记录格式和相应的记录长度。

消息返回码

在尝试打开必需 DD 时返回 16，否则返回 12。

BOZ1145U 编译器中的内存不足无法继续编译。

说明

由于内存过低，优化器无法继续。

系统操作：
优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应：
考虑使用 JCL MEMLIMIT 或 JCL REGION 参数来增加优化器使用的内存。有关更多信息，请参阅 [z/OS MVS Initialization and Tuning Reference](#) 和 [z/OS MVS Initialization and Tuning Guide](#)。

消息返回码

16

BOZ1400S 伪指令缺少“&1”说明符。

说明

优化器遇到需要 &1 说明符的 BOPT 或 IEFOPZ 伪指令，但是缺少说明符。

系统操作：
优化器丢弃伪指令并继续处理 SYSIN 输入文件中的下一行。

用户响应：
通过添加相应的 &1 说明符纠正伪指令。

消息返回码

12

BOZ1401S 必须在行开头指定“&1”伪指令。

说明

选项位于 SYSIN 输入文件某一行的 &1 伪指令之前，或者行缺少 &1 伪指令。

系统操作:

优化器丢弃伪指令并继续处理 SYSIN 输入文件中的下一行。

用户响应:

通过在行开头指定 &1 伪指令修复行。

消息返回码

12

BOZ1402S “&1” 的说明符无效。

说明

BOPT 或 IEFOPZ 伪指令的 &1 选项包含无效的说明符。例如，“H”在选项“SCAN=H”中是无效说明符。

系统操作:

优化器丢弃具有无效说明符的伪指令并继续处理 SYSIN 输入文件中的下一行。

用户响应:

将选项中的说明符更改为有效值。

消息返回码

12

BOZ1403S 选项“&1”无效。

说明

在处理 SYSIN 输入文件时，遇到 &1，而 &1 不是优化器伪指令且 &1 不是有效选项。

系统操作:

优化器丢弃具有无效选项的行并继续处理 SYSIN 文件中的下一行。

用户响应:

使用拼写正确的伪指令或选项来更正 SYSIN 行。

消息返回码

12

BOZ1404S 只能在“&2”伪指令上指定“&1”。

说明

在伪指令上指定 &1 选项，但这不是 &2 伪指令。例如，无法在 BOPT 伪指令上指定 SEL_ARCH，因为 SEL_ARCH 仅适用于 IEFOPZ 伪指令。

系统操作:

优化器丢弃具有无效选项的行并继续处理 SYSIN 文件中的下一行。

用户响应:

通过指定适当的选项或适当的伪指令，修订包含 &1 的行。

消息返回码

12

BOZ1405S 在“&2”伪指令上不允许“&1”。

说明

&2 伪指令包含选项 &1，而此选项不适合 &2 伪指令。例如，“IN”选项不适合 IEFOPZ 伪指令且无法在此伪指令上指定。

系统操作:

优化器丢弃具有无效选项的行并继续处理 SYSIN 文件中的下一行。

用户响应:

通过指定应用于 &2 伪指令的适当选项，修复此行。

消息返回码

12

BOZ1406S “&1”的成员说明符中不支持通配符。

说明

遇到 SYSIN 行，该行具有成员说明符 (&1) 中包含通配符的“IN”选项，以及包含数据集成员说明符的“OUT”选项。在“IN”选项中使用通配符时，“OUT”选项不得包含成员说明符。

系统操作:

优化器丢弃具有无效选项的行并继续处理 SYSIN 文件中的下一行。

用户响应:

更改“IN”选项以不指定通配符或者除去“OUT”选项中的成员说明符。

消息返回码

12

BOZ1407S 在输入上使用通配符时，输出说明符“&1”无效。

说明

优化器检测到一个 BOPT 伪指令，此伪指令在“IN”选项上使用通配符说明符，且“OUT”选项指定了 USS 路径 &1。在“IN”选项上使用成员通配符时，“OUT”选项必须指定数据集而不是 USS 路径。

系统操作:

优化器丢弃具有无效“OUT”选项的 BOPT 选项并继续处理 SYSIN 文件中的下一行。

用户响应:

更改“IN”选项以不指定通配符或者更改“OUT”选项以指定数据集。

消息返回码

12

BOZ1408S 模块说明符 “&1” 是现有目录。

说明

优化器检测到在 BOPT 伪指令的 “IN” 或 “OUT” 选项上指定为模块位置的 HFS 目录 &1。在 HFS 中，模块是普通文件而不是目录。

系统操作:

优化器丢弃伪指令并继续处理 SYSIN 文件中的下一行。

用户响应:

将 “IN” 或 “OUT” 选项上的路径说明符更改为普通文件而非目录。

消息返回码

12

BOZ1409S 输出说明符 “&1” 不是现有 PDS(E)。

说明

优化器遇到以下一种情况:

- &1 作为来自 IEFOPZ 配置的 NEW 数据集
- &1 作为 SYSIN 文件中文件说明符上的 DD 名称或数据集名称，其中，文件说明符包含成员名称，
- &1 作为 LOG 选项的数据集说明符

但是，与 &1 相关联的数据集不存在或者数据集是连续数据集且非 PDS(E)。

系统操作:

在 IEFOPZ 情况下，优化器忽略 NEW 数据集并继续处理 IEFOPZ 配置。否则，优化器丢弃伪指令并继续处理 SYSIN 文件中的下一行。

用户响应:

将数据集位置更改为现有 PDS(E) 或在运行优化器前分配 PDS(E)。

消息返回码

12

BOZ1410I 无法替换输出模块 “&1”，因为 REPLACE=N 生效。

说明

在指定 REPLACE=Y 选项时，如果检测到相同名称的输出模块 (&1) 已存在，那么优化器发出此参考消息。

系统操作:

优化器绕过输入模块优化，继续处理下一个模块或下一个伪指令。

用户响应:

用户不需要执行任何操作。

消息返回码

0

BOZ1411S 从数据集说明符 “&1” 获取成员列表时出错。

说明

优化器已在处理以下之一:

- BOPT 伪指令，其中，在 “IN” 选项（包含成员通配符）上指定了 PDS(E) (&1) 且 PDS(E) 无成员
- IEFOPZ 伪指令，并且在 IEFOPZ 配置中找到一个无成员的 OLD 数据集 (&1)

系统操作:

在 BOPT 伪指令情况下，优化器丢弃伪指令并继续处理 SYSIN 文件的下一行。在 IEFOPZ 伪指令的情况下，优化器忽略 OLD 数据集并继续处理 IEFOPZ 配置的其余部分。

用户响应:

检查在 BOPT 伪指令上指定了适当的数据集或者在 IEFOPZ 配置中指定了适当的数据集。

消息返回码

12

BOZ1412S IEFOPZ 在此系统上不可用。

说明

优化器已在无 IEFOPZ 功能的 z/OS 系统上处理 IEFOPZ 伪指令。

系统操作:

优化器丢弃 IEFOPZ 伪指令并继续处理 SYSIN 文件中的下一行。

用户响应:

IEFOPZ 工具仅适用于 z/OS V2R2 以及更高版本。如果优化器在 V2R2 之前的 z/OS 系统上运行，那么更改 SYSIN 以不指定 IEFOPZ 伪指令。如果优化器在 z/OS V2R2 或更高版本上运行，那么让系统管理员安装 IEFOPZ 功能所需的相应 PTF。

消息返回码

12

BOZ1413S IEFOPZQ 系统服务问题（返回码 = “&1”，原因码 = “&2”）：&3。

说明

在处理 IEFOPZ 伪指令时，优化器遇到有关 IEFOPZ 配置的问题。&1 指定错误返回码，&2 指定用于读取配置的 IEFOPZQ 系统服务的错误原因码。&3 提供原因码的简短描述。

系统操作:

优化器丢弃 IEFOPZ 伪指令并继续处理 SYSIN 文件中的下一行。

用户响应:

向系统程序员提供此错误消息以查看错误是否有效。如果不存在与 IEFOPZ 使用相关的问题, 那么请联系 IBM 服务人员并提供此优化器消息以及任何其他 IEFOPZ 配置信息。

消息返回码

12

BOZ1414S 输入说明符 “&1” 不是现有 PDS(E)。

说明

优化器遇到以下一种情况:

- &1 作为来自 IEFOPZ 配置的 OLD 数据集
- &1 作为 SYSIN 文件中文件说明符上的 DD 名称或数据集名称, 其中, 文件说明符包含成员名称,

但是, 与 &1 相关联的数据集不存在或者数据集是连续数据集且非 PDS(E)。

系统操作:

在 IEFOPZ 情况下, 优化器忽略 OLD 数据集并继续处理 IEFOPZ 配置。否则, 优化器丢弃伪指令并继续处理 SYSIN 文件中的下一行。

用户响应:

将数据集名称更改为现有 PDS(E) 或分配 PDS(E)。

消息返回码

12

BOZ1415S 未针对 “&1” 提供 DD 定义。

说明

优化器找不到必需的优化器 DD (&1) 的 DD 定义, 或者未针对在 SYSIN 文件中使用的 DD (&1) 指定任何 DD 定义。

系统操作:

如果 DD 是针对优化器的必需 DD, 那么优化器立即终止并且返回码为 16。否则, 优化器丢弃 SYSIN 文件中包含 DD 定义的行并处理 SYSIN 的下一行。

用户响应:

错误地向 DD 提供 DD 定义。

消息返回码

16, 如果 &1 是必需的 DD, 否则为 12

BOZ1416S 未针对 PDS(E) 说明符 “&1” 指定成员名。

说明

优化器遇到指定了 PDS(E) (&1) 且需要成员名的 “IN” 或 “OUT” 选项, 但是选项上未包含任何成员。

系统操作:

优化器丢弃具有无效的 “IN” 或 “OUT” 选项的伪指令并继续处理 SYSIN 文件中的下一行。

用户响应:

更改 “IN” 或 “OUT” 选项以包含数据集成员。

消息返回码

12

BOZ1417S 文件 “&1” 不存在。

说明

无法分配输入文件 &1。两个常见原因可能导致此情况:

1. 现有输入模块 PDS(E) 的成员不存在
2. 已指定无效的 HFS 路径

系统操作:

优化器忽略处理使用无效文件规范的伪指令 (或输入模块), 并继续处理下一个输入模块或下一个伪指令

用户响应:

通过指定现有数据集成员或者纠正路径规范以指向现有 HFS 文件来纠正问题。

消息返回码

12

BOZ1418S 文件规范 “&1” 无效。

说明

文件 &1 的规范不正确。不正确的规范示例包括:

1. 成员名被指定两次:
 - 一次在 DD 定义中
 - 另一次在包含 DD 定义的优化器选项或伪指令中
2. 指定 HFS 路径, 但是路径中的目录不存在或者路径不可访问。
3. DD 名称或数据集名称的长度太长。

系统操作:

优化器忽略处理使用无效文件规范的伪指令 (或输入模块), 并继续处理下一个输入模块或下一个伪指令

用户响应:

针对文件规范 (&1) 指定适当的格式。

消息返回码

12

BOZ1419S 在输入采用程序对象格式时, 不支持 “&1” 的装入模块的输出。

说明

输入模块具有较新的程序对象格式, 但是优化模块 (&1) 以较旧的装入模块格式为目标。在输入模块是 PDSE 的成员或者是 HFS 路径中的文件但优化模块以 PDS 成员为目标或者以连续数据集为目标时, 会发生此情况。

系统操作:
优化器终止处理输入模块并继续处理下一个输入模块或下一个伪指令。

用户响应:
将优化模块的输出位置 (&1) 更正为 PDSE 的成员或 HFS 路径。

消息返回码

12

BOZ1420S 路径 “&1” 必须是绝对路径且以 “/” 开头。

说明

输入或输出文件的规范 (&1) 是 HFS 文件，但是未针对 &1 提供完整路径规范。完整或绝对路径文件规范必须以 “/” 字符开头。例如，在优化器处理输入模块、输出模块或列表转换的 HFS 规范时，可能发生此错误。

系统操作:
在指定无效路径时优化器绕过模块优化并继续处理下一个输入模块或下一个伪指令。

用户响应:
将路径 (&1) 的规范更正为绝对路径。

消息返回码

12

BOZ1421S 绑定程序 API “&1” 失败：返回码 = &2，原因码 = &3。

说明

在使用绑定程序 API (&1) 处理模块时，绑定程序 API 返回意外的返回码 (&2) 和原因码 (&3)。

系统操作:
在大多数情况下，优化器放弃输入模块处理并继续处理下一个输入模块或下一个伪指令。在某些情况下（例如，返回码 = 4，原因码 = 0x83000526），绑定程序能够从问题恢复（在此情况下将出现意外的输入）并且输入模块优化继续。

用户响应

检查绑定程序文档以获取有关原因码的信息。原因码信息可帮助确定问题的原因。例如，原因码可能指示优化的输入文件不是适当的装入模块或程序对象文件。在此情况下，纠正 JCL 或 SYSIN 文件以指定适当的输入模块。有关绑定程序 API 返回码和原因码的信息，请参阅 [z/OS MVS 程序管理：高级工具](#)。

此消息通常由 BOZ4116 绑定程序消息继续提供，该绑定程序消息可能会提供有助于您的响应的其他信息。

消息返回码

在优化器停止处理时为 12，否为 4。

BOZ1422S 无法处理模块，因为其未标记为可执行。

说明

优化器遇到未标记为可执行的输入模块。优化器需要模块标记为可执行以使优化流程成功。

系统操作:
优化器停止处理输入模块并继续处理下一个输入模块或下一个伪指令。

用户响应:
如果此问题是模块的预期行为，那么忽略消息或者更改优化器伪指令以排除模块。否则，如果问题属于意外，那么更正生成输入模块的绑定步骤，从而使从绑定生成的模块标记可执行。

消息返回码

12

BOZ1423S 无法处理模块，因为其链接 **EDIT=NO** 或者无法进行再处理。

说明

优化器遇到无法编辑的输入模块。最常见的情况是，用于生成模块的绑定步骤包含 EDIT=NO 绑定程序选项。无法编辑的模块缺少优化器所需的重要信息。

系统操作:
优化器停止处理输入模块并继续处理下一个输入模块或下一个伪指令。

用户响应:
如果此问题是模块的预期行为，那么忽略消息或者更改优化器伪指令以排除模块。否则，如果问题属于意外，那么从生成输入模块的绑定步骤中除去 EDIT=NO 选项。

消息返回码

12

BOZ1424S 无法适当地处理模块，因为程序为 **SIGNed**。

说明

优化器遇到标记为 SIGNed 的输入模块。

系统操作:
优化器不支持 SIGNed 模块且停止处理输入模块，并继续处理下一个输入模块或下一个伪指令。

用户响应:
如果此问题是模块的预期行为，那么忽略消息或者更改优化器伪指令以排除模块。否则，如果问题属于意外，那么更正用于生成模块的绑定步骤以使模块不标记为 SIGNed。

消息返回码

12

BOZ1428U	在绑定程序 API “&1” 期间遇到内存不足：返回码 = &2，原因码 = &3。终止优化器。	说明 在使用绑定程序 API (&2) 处理模块时，绑定程序检测到类型为 &1 的错误。绑定程序 API 提供了返回码 (&3) 和原因码 (&4)。
系统操作：	优化器立即终止执行并返回至操作系统（返回码 16）。	系统操作： 绑定程序立即终止，返回码为 16。
用户响应	考虑使用 JCL MEMLIMIT 或 JCL REGION 参数以增加优化流程期间使用的内存。有关更多信息，请参阅 z/OS MVS Initialization and Tuning Reference 和 z/OS MVS Initialization and Tuning Guide 。	用户响应 问题的类型 (&1) 和原因码 (&4) 可帮助指示如何诊断和修复问题的步骤。如果无法诊断问题，那么请联系 IBM 服务人员以获取帮助。有关绑定程序 API 返回码和原因码的信息，请参阅 z/OS MVS 程序管理：高级工具 。
	此消息通常由 BOZ4116 绑定程序消息继续提供，该绑定程序消息可能会提供有助于您的响应的其他信息。	此消息通常由 BOZ4116 绑定程序消息继续提供，该绑定程序消息可能会提供有助于您的响应的其他信息。
消息返回码	16	消息返回码 16
BOZ1429U	在绑定程序 API “&2” 期间遇到 “&1” I/O：返回码 = &3，原因码 = &4。终止优化器。	BOZ1431S 在绑定程序 API “&2” 期间遇到包含不受支持的功能 “&1” 的输入模块：返回码 = &3，原因码 = &4。已绕过模块。
说明	在使用绑定程序 API (&2) 处理模块时，绑定程序检测到类型为 &1 的 I/O 错误。绑定程序 API 提供了返回码 (&3) 和原因码 (&4)。	说明 在使用绑定程序 API (&2) 处理模块时，绑定程序检测到由于模块包含不受支持的功能 &1 而无法优化模块。绑定程序 API 提供了返回码 (&3) 和原因码 (&4)。
系统操作：	优化器立即终止执行并返回至操作系统（返回码 16）。	不受支持的功能的一个示例为：输入是对象模块（与之相反的是输入为装入模块或程序对象）。此问题的另一个示例是：输入模块未完全绑定且包含 “UNRESOLVED” 引用。
用户响应	I/O 问题的类型 (&1) 和原因码 (&4) 可帮助指示如何诊断和修复问题的步骤。请参阅绑定程序文档以了解原因码解释。现在，示例为优化模块的 “WRITE” (&1) 错误，因为输出 PDS(E) 或文件系统已满。绑定程序 API 信息 (&2) 或原因码 (&4) 可帮助确认或指向 “WRITE” 问题的原因。请注意，增加 PDS(E)（或文件系统）的大小可修复 “WRITE” 问题。有关绑定程序 API 返回码和原因码的信息，请参阅 z/OS MVS 程序管理：高级工具 。	系统操作：
	此消息通常由 BOZ4116 绑定程序消息继续提供，该绑定程序消息可能会提供有助于您的响应的其他信息。	用户响应 因为无法支持输入模块，所以选项为：
消息返回码	16	· 忽略消息 · 更改优化器输入以避免优化模块 · 修复问题。例如，在包含 “UNRESOLVED” 引用的模块中，更改用于生成模块的构建步骤，从而完全绑定模块
BOZ1430U	在绑定程序 API “&2” 期间遇到不可恢复的 “&1” 错误：返回码 = &3，原因码 = &4。终止优化器。	有关绑定程序 API 返回码和原因码的信息，请参阅 z/OS MVS 程序管理：高级工具 。
		此消息通常由 BOZ4116 绑定程序消息继续提供，该绑定程序消息可能会提供有助于您的响应的其他信息。
		消息返回码 12
BOZ1432S	输出模块大小已超过模块格式限制且无法保存。	

说明

优化器尝试编写优化模块，但是遇到输出格式限制。保存到 PDS 成员（或连续数据集）的装入模块具有最严格的格式。更少的情况是遇到程序对象的格式限制（写入到 PDSE 或 HFS 路径）。

系统操作:

优化器终止输入模块优化并继续处理下一个模块或下一个伪指令。

用户响应:

如果将输出模块保存到 PDS 成员或连续数据集，那么考虑将输出位置更改为 PDSE 的成员。否则，考虑将程序拆分为多个模块。

消息返回码

12

BOZ1436S ARCH 规范无效: &1

说明

在以下某种情况下检测到无效或不受支持的体系结构规范 (&1):

- 1. 处理 SYSIN 文件时，在 ARCH 选项或 SEL_ARCH 选项中
- 2. 在处理 IEFOPZ 配置中的 NEW 数据集时

系统操作:

如果在 SYSIN 文件的行上检测到无效的规范，那么优化器丢弃具有无效选项的行并继续处理 SYSIN 文件中的下一行。如果在处理 IEFOPZ 配置的 NEW 数据集时检测到无效的规范，那么优化器忽略 NEW 数据集并继续处理 IEFOPZ 配置的其余部分。

用户响应:

通过指定优化器支持的 ARCH 级别，更正 SYSIN 文件或 IEFOPZ 配置。

消息返回码

12

BOZ1437S 未找到 BOPT 或 IEFOPZ 伪指令

说明

优化器既找不到 BOPT，也找不到 IEFOPZ 伪指令。

系统操作:

优化器将终止执行并返回至操作系统（返回码 12）。

用户响应:

请检查 JCL 是否至少包含一个 BOPT 或 IEFOPZ 伪指令。

消息返回码

12

BOZ1438U dynfree dyn 失败: 对于 DD &2, rc=&1

说明

优化器检测到尝试释放内部输入 DD 名称 (&2) 的错误，优化器在优化流程期间使用此名称。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

先前的问题可能导致此问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

16

BOZ1439U dynfree saveDyn 失败: 对于 DD &2, rc=&1

说明

优化器检测到尝试释放内部输出 DD 名称 (&2) 的错误（dynfree 服务返回了 &1），优化器在优化流程期间使用此名称。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

先前的问题可能导致此问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

16

BOZ1446U 写入 &1 时发生 I/O 错误

说明

在写入 &1 时，优化器检测到 I/O 错误，其中 &1 可以是“列表转换”或“日志文件”。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

检查 PDS(E) 或文件系统是否已满并针对 PDS(E) 分配更大的文件或者增加文件系统的大小。另外，检查是否使用适当的记录格式和记录长度分配数据集。

消息返回码

16

BOZ1447U 发生意外的 I/O 错误

说明

在执行期间，优化器检测到 I/O 错误。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

检查输出文件的定义（您应该能够排除输出模块）以确保使用适当的记录长度和记录格式并检查文件是否已满。

消息返回码

16

BOZ1449U 未处理的内存不足异常

说明

由于内存过低，优化器无法继续。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

考虑使用 JCL MEMLIMIT 或 JCL REGION 参数来增加优化器使用的内存。有关更多信息，请参阅 [z/OS MVS Initialization and Tuning Reference](#) 和 [z/OS MVS Initialization and Tuning Guide](#)。

消息返回码

16

BOZ1450U 断言失败，请检查日志以进行追溯

说明

优化器无法继续，因为处理期间遇到意外情况。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

由于先前的问题，可能发生意外问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

16

BOZ1451S **dynalloc(): DSN &1 失败，DD 为 &2，错误代码为 &3，其信息代码为 &4**

说明

在针对数据集 (&1) 分配内部 DD (&2) 时，优化器遇到错误。&3 是 MVS 动态分配函数返回的错误代码。&4 是 MVS 动态分配函数返回的信息代码。

系统操作:

优化器终止输入模块优化并继续处理下一个模块或下一个伪指令。

用户响应:

检查数据集 (&2) 存在并且可访问，并检查 JCL 不包含相同 DD (&2) 的定义。由于先前的问题，也可能发生意外的问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

12

BOZ1452S **dynalloc(): 路径 &1 失败，DD 为 &2，错误代码为 &3，其信息代码为 &4**

说明

&3 是 MVS 动态分配函数返回的错误代码。&4 是 MVS 动态分配函数返回的信息代码。

系统操作:

优化器终止输入模块优化并继续处理下一个模块或下一个伪指令。

用户响应:

检查路径可访问且可写入，并检查 JCL 不包含相同 DD (&2) 的定义。由于先前的问题，也可能发生意外的问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

12

BOZ1453U **dynalloc(): DUMMY DD &1 失败，错误代码为 &2，信息代码为 &3**

说明

在分配优化流程所需的必需 DUMMY DD (&1) 时，优化器遇到错误。&2 是 MVS 动态分配函数返回的错误代码。&3 是 MVS 动态分配函数返回的信息代码。

系统操作:

优化器立即终止执行并返回至操作系统（返回码 16）。

用户响应:

检查 JCL 不包含相同 DD (&1) 的定义。由于先前的问题，也可能发生意外的问题。请更正日志文件中报告的任何问题并重试优化流程。如果问题仍存在，请联系 IBM 服务人员以获取帮助。

消息返回码

16

BOZ1455W 找到不受支持的功能 “&1”

说明

在以下某种情况下会发出此消息：

1. 在 ABO 遇到由不适用于 ABO 的编译器构建的 COBOL CSECT（例如，已编译的 COBOL 程序）或者 CSECT 包含不受 ABO 支持的 COBOL 语言功能时。
2. 在 ABO 遇到太过复杂而难以安全优化的 CSECT 时。

在第一种情况下，ABO 在正在处理的 CSECT 中检测到其不支持的功能 “&1”。请参阅 [COBOL 模块需求](#)，以了解能够用于 ABO 的编译器以及 ABO 不支持的 COBOL 语言功能。

在第二种情况下，ABO 已确定 CSECT 太过复杂而难以安全优化，因此已跳过该 CSECT。如果可确保优化程序将使用与原始已编译程序相同的逻辑来执行，那么 ABO 将仅优化 CSECT。如果 CSECT 太复杂而使 ABO 无法提供此保证，那么 ABO 会停止优化流程并跳过此 CSECT。仍会处理模块中任何其他合格的 CSECT。

请注意，发出此消息仅供参考，并非指示 ABO 存在功能问题。

系统操作:

ABO 绕过 CSECT 优化并继续处理下一个 CSECT。

用户响应:

如果您在优化大量模块时看到针对特定不受支持的功能发出消息 BOZ1455，那么您可以打开 RFE 以指示缺少此功能将妨碍您有效地使用 ABO。

消息返回码

4

BOZ1456S “&1”不能同时是优化器输入和优化器输出。

说明

优化器不允许将数据集或文件既用作优化器的输入又用作优化器的输出。例如，如果 PDS(E) 是输入模块的源，那么无法将优化模块写入此 PDS(E)。在 &1 既用作优化器的输入位置又用作优化器的输出位置时，发出此消息。

系统操作:

优化器终止输入模块优化并继续处理下一个模块。

用户响应:

纠正 JCL 或 SYSIN 文件，从而使输出数据集与输入数据集分离。

消息返回码

12

BOZ1457S 过滤器表达式 “&1” 无效。

说明

优化器无法处理 BOPT 优化器伪指令的 *mem_expr* 参数或 CSECT 优化器选项的 *expr* 参数。表达式包含无效的语法或通配符，无法处理为已写入。

系统操作:

优化器终止优化当前伪指令，并继续处理下一个伪指令。

用户响应:

请更正表达式。请参阅第 14 页的『BOPT』的 *mem_expr* 参数和 CSECT 优化器选项的 *expr* 参数的描述。

消息返回码

12

BOZ1490W 警告：在绑定程序 API “&1” 期间遇到 AMODE/RMODE 冲突：返回码 = &2，原因码 = &3。已执行操作，并且处理将继续。

说明

在优化流程期间，绑定程序检测到与 AMODE 和 RMODE 设置的冲突。绑定程序 API (&1) 检测到此问题，对于此问题，绑定程序发出返回码 (&2) 和原因码 (&3)。您可以使用绑定程序原因码文档以及原因码 (&3) 来确定冲突的确切特性。通常，冲突已存在于输入模块中，而不是由优化流程引入的。

系统操作:

绑定程序发出此 BOZ1490 警告消息并继续优化输入模块。

用户响应:

警告消息可能是要优化的输入模块的问题指示符。修复问题可能需要修复用于生成输入模块的构建步骤。有关绑定程序 API 返回码和原因码的信息，请参阅 [z/OS MVS 程序管理：高级工具](#)。

消息返回码

4

BOZ1491W 警告：在绑定程序 API “&1” 期间，向目录添加别名时遇到问题：返回码 = &2，原因码 = &3。模块已保存，处理继续。

说明

在优化流程期间，绑定程序检测到向目录添加别名的问题。绑定程序 API (&1) 检测到此问题，对于此问题，绑定程序发出返回码 (&2) 和原因码 (&3)。在绑定程序发现存在与别名名称相同的 PDS(E) 目录成员时，无法将别名添加到 PDS(E) 目录。

系统操作:

绑定程序发出此 BOZ1491 警告消息并继续优化输入模块。

用户响应

要解决此问题，重要的是了解为何输出 PDS(E) 存在与别名名称相同的现有成员。例如：

- 请勿在 BOPT 伪指令的 IN 选项的成员说明符上指定别名。如果指定了别名，那么删除此 BOPT 伪指令并删除目标数据集中的成员。
- BOPT 伪指令的 OUT 选项上不正确的成员说明符可能导致与别名名称冲突。确保 BOPT 伪指令的 OUT 选项具有与针对 IN 选项提供的成员说明符相同的成员说明符。
- 合并来自多个输入数据集的别名可能导致与两个数据集的别名和成员名冲突。建议对于每个输入数据集使用不同的输出数据集。有关绑定程序 API 返回码和原

因码的信息，请参阅 *z/OS MVS 程序管理：高级工具*。

消息返回码

4

BOZ1492W 警告：带有导出符号的输入模块“&1”将保存到不同的指定模块“&2”。

说明

优化器在包含导出符号的 PDS(E) 中检测到输入模块 (&1)，并且已将优化模块写入到输出 PDS(E) 中指定的其他成员 (&2)。

系统操作:

绑定程序发出此 BOZ1492 警告消息并继续优化输入模块。

用户响应:

要修复问题，请更改 JCL 或 SYSIN 文件，从而使优化模块的成员名与输入模块的成员名相同。无法执行此操作可能由于成员名称更改而在查找优化模块时导致运行时问题。

消息返回码

4

BOZ1493S 遇到合并的 DD “&1”，而这对于“&2”不允许。

说明

优化器检测到并置两个或多个数据集的输入或输出 DD 定义 (&1)。&2 提供使用 DD 的上下文。例如，&2 可能指示 DD 用作输入模块位置、输出模块位置或输出列表转换位置。

系统操作:

优化器绕过包含合并的 DD 定义的伪指令，并且优化器继续处理下一个伪指令。

用户响应:

修复 JCL 以不包含输入模块、输出模块和列表转换的合并 DD 定义。

消息返回码

12

BOZ1494S 未处理模块，因为其未完全绑定。

说明

在优化流程期间，在未完全绑定的输入中遇到模块且指定了 ALLOW=NOUNRESEXE 选项。除非指定 ALLOW=UNRESEXE 选项，否则优化器不会处理未完全绑定的模块。

系统操作:

优化器终止输入模块优化并继续处理下一个模块或下一个伪指令。

用户响应:

如果目的是优化部分绑定的模块，那么请除去 ALLOW=NOUNRESEXE 选项。如果目的仅是优化完全绑定的模块，请忽略错误或者纠正 JCL 或 SYSIN 文件以仅处理完全绑定的模块。

消息返回码

12

BOZ4089I **IEFOPZ:** 未针对具有 ARCH=&3 的数据集“&2”获取 ARCH=&1 匹配。

说明

在处理 IEFOPZ 优化器伪指令时，在以下情况下优化器发出此参考消息：在 IEFOPZ 配置中找到 NEW 数据集 (&2)，此配置的 ARCH 规范 (&3) 与 IEFOPZ 优化器伪指令上指定的 SEL_ARCH 选择器 (&1) 不匹配。

系统操作:

优化器绕过 NEW 数据集并处理配置中下一个 NEW 数据集。

用户响应:

用户不需要执行任何操作。

消息返回码

0

BOZ4091I **IEFOPZ:** 未针对具有 STATE=&3 的数据集“&2”获取 STATE=&1 匹配。

说明

在处理 IEFOPZ 优化器伪指令时，在以下情况下优化器发出此参考消息：在 IEFOPZ 配置中找到 NEW 数据集 (&2)，此配置的 STATE 规范 (&3) 与 IEFOPZ 优化器伪指令上指定的 SEL_STATE 选择器 (&1) 不匹配。

系统操作:

优化器绕过 NEW 数据集并处理配置中下一个 NEW 数据集。

用户响应:

用户不需要执行任何操作。

消息返回码

0

BOZ4092I **IEFOPZ:** 未获取数据集“&2”的 DSN='&1' 匹配。

说明

在处理 IEFOPZ 优化器伪指令时，在以下情况下优化器发出此参考消息：在 IEFOPZ 配置中找到 OLD 数据集 (&2)，而此配置与 IEFOPZ 优化器伪指令上指定的 SEL_OLD 选择器值 (&1) 不匹配。

系统操作:

优化器绕过 OLD 数据集并处理配置中下一个 OLD 数据集。

用户响应:
用户不需要执行任何操作。

消息返回码

0

BOZ4097I 数据集 “&1” 中没有任何要处理的成员

说明

如果数据集中没有任何要处理的成员，那么会发出此消息。

系统操作:
优化器将继续处理下一个数据集。

用户响应:
用户不需要执行任何操作。

消息返回码

0

BOZ4101W 找不到适用的 COBOL 代码部分，返回码 4

说明

在以下情况下发出此消息：

1. 在优化器遇到装入模块但是未优化装入模块中的任何 CSECT 时（注：如果指定 REPLACE=Y 选项并且优化模块已存在，那么将不打印消息）
2. 在 “IN” 选项中具有成员通配符的 BOPT 伪指令之后，但是未优化 “IN” 数据集集中的任何模块
3. 在处理 OLD 数据集之后，但是未优化 OLD 数据集集中的任何模块
4. 在处理 IEFOPZ 伪指令之后，但是未优化任何模块

系统操作:
优化器继续处理下一个输入模块。

用户响应:
用户不需要执行任何操作。

消息返回码

4

BOZ4107I 信息：未将 IDRL 记录添加到 CSECT &1，因为装入模块格式不支持三个 IDRL。处理继续。

说明

因为 CSECT 已有 2 条 IDRL 记录而无法将 IDRL 记录（针对二进制优化器自身）添加到装入模块中的优化 CSECT 时，优化器发出此参考消息。注：PDS 中的装入模块限制为每个 CSECT 最多 2 个 IDRL（但是对于 PDSE 中的程序对象无此限制）。

系统操作:
优化器继续处理其输出装入模块。

用户响应:
用户不需要执行任何操作。

消息返回码

0

BOZ4109I 信息：向装入模块 CSECT “&1” 添加第三个 IDRL。

说明

在处理装入模块中的 CSECT (&1) 时，如果优化器添加其语言记录作为 CSECT (&1) 的第 3 个 IDRL，那么优化器发出此参考消息。请注意，需要更新绑定程序，从而使绑定程序可适当地向 CSECT 添加第 3 个 IDRL。如果未在系统上安装绑定程序更新，那么在尝试保存优化装入模块时，优化器将发出后续警告消息。

系统操作:
如果在保存模块时优化器发出警告消息，那么可能未将语言记录添加到 CSECT 并且优化器继续处理 CSECT。否则，已成功执行 CSECT 处理。

用户响应:
如果在保存模块时优化器发出警告消息，那么应联系系统程序员以安装绑定程序并再次执行优化流程。否则，用户无需执行任何操作。此消息的必需绑定程序更新位于 APAR OA50460 中。请参阅第 3 页的『受支持的操作系统』，以获取更多信息。

消息返回码

0

BOZ4110I 信息：执行第二个绑定以处理类 “&2” 中的专用部分 “&1”，指示偏移量 &4 处的 ENTRY “&3”。

说明

在以下情况下优化器发出此消息：处理具有 COBOL ENTRY 语句 (&3) 的 CSECT，存在来自专用部分 (&1) 的 ENTRY (&3) 的引用，其位于类 (&2，通常为 C_WSA) 中，偏移量为 &4。请注意，需要更新绑定程序，从而使第二个绑定正常工作。如果绑定程序未更新，优化程序可能遇到问题。

系统操作:
如果绑定程序更新可用，那么处理成功完成。但是，如果绑定程序更新不可用，第二个绑定可能显示为成功完成，但可能发生运行时错误。

用户响应:
如果在系统上安装绑定程序更新，那么无需任何操作。否则，让系统程序员安装绑定程序更新并再次执行优化流程。此消息的必需绑定程序更新位于 APAR OA50460 中。请参阅第 3 页的『受支持的操作系统』，以获取更多信息。

消息返回码

BOZ4111I 信息：对类 “&2” 中的专用部分 “&1” 执行更新，指示偏移量 &4 处的 ENTRY “&3”。

说明

在以下情况下优化器发出此消息：处理具有 COBOL ENTRY 语句 (&3) 的 CSECT，存在来自专用部分 (&1) 的 ENTRY (&3) 的引用，其位于类 (&2) 中，偏移量为 &4。请注意，需要更新绑定程序，从而使第二个绑定正常工作。如果没有绑定程序更新，那么在处理引用时优化程序将发出错误消息。

系统操作：

如果绑定程序更新不可用，那么优化器发出处理引用的错误。否则，绑定程序成功处理引用。

用户响应：

如果优化器发出处理引用的错误，那么让系统程序员安装绑定程序更新并再次执行优化流程。否则，无需任何操作。此消息的必需绑定程序更新位于 APAR OA50460 中。请参阅第 3 页的『受支持的操作系统』，以获取更多信息。

消息返回码

0

BOZ4113I **CSECT &1 was excluded by filter - skip**

说明

当由于 CSECT 优化器选项中的表达式而导致优化器排除 CSECT 时，会发出此消息。

系统操作：

优化器继续处理下一个 CSECT。

用户响应：

用户不需要执行任何操作。

消息返回码

0

BOZ4114I 信息：在 ALLOW=UNRESEXE 选项生效时处理未完全绑定的模块。

说明

在指定 ALLOW=UNRESEXE 选项时，如果在未完全绑定的输入中遇到模块，那么优化器发出此参考消息。

如果完全绑定模块，那么不会发出此消息。

此消息可用于确定优化器已处理的部分绑定的模块。

系统操作：

如果绑定程序更新不可用，那么优化器发出处理引用的错误。否则，绑定程序成功处理引用。

用户响应：

优化器处理部分绑定的模块并输出优化的部分绑定的模块。

消息返回码

0

BOZ4116I **绑定程序消息 “&1”**

说明

优化器使用绑定程序服务，某个服务可能会失败，从而阻止模块进行优化。为了响应失败的绑定程序服务，ABO 会生成诸如 BOZ1429 之类的消息并以某种方式终止处理。BOZ1429 消息可能缺少关于绑定程序服务失败原因的详细信息。例如，BOZ1429 消息指示绑定程序发现输入模块存在一些问题，但 BOZ1429 未包含绑定程序找到的精确输入问题。要提供有关失败的绑定程序服务的更多信息，优化器将捕获严重的绑定程序消息，并在 BOZ4116 参考消息的 &1 中包含绑定程序消息的文本。这意味着，当绑定程序服务失败时，ABO 通常会发出两条消息：

1. BOZ4116 消息的 &1 中包含严重绑定程序消息的文本。BOZ4116 消息后跟
2. 诸如 BOZ1429 之类的摘要消息，指示绑定程序失败的一般性质。

系统操作：

请参阅摘要消息的“系统操作”部分以确定优化器的操作。

用户响应：

请参阅摘要消息的“用户响应”部分以确定要执行的操作。BOZ4116 消息可能会提供有助于指导您的响应的信息。

消息返回码

0

BOZ4117I **成员 “&1” 已被过滤器排除 - 跳过**

说明

当由于 BOPT 伪指令的 IN 选项中的表达式而导致优化器排除模块时，会发出此消息。

系统操作：

优化器继续处理下一个模块。

用户响应：

用户不需要执行任何操作。

消息返回码

0

BOZ4119S **SYSIN 第 &1 行上指示连续。无法读取 SYSIN 第 &2 行。**

说明

在 SYSIN 的第 &1 行上指示连续，第 &1 行的最后一个非空白字符是连续字符（“+”或“-”）。在读取 SYSIN 时，优化器无法读取第 &2 行。

系统操作:
优化器将丢弃此行。

用户响应:
除去 SYSIN 的第 &1 行末尾的连续字符，或者将新的第 &2 行添加到将连续第 &1 行的 SYSIN。

消息返回码

12

BOZ4120S 不能具有多个 IN= 或 OUT= 说明符。

说明

当在 BOPT 伪指令中检测到多个 IN 或 OUT 说明符时，将发出此消息。

系统操作:
优化器继续处理下一个 BOPT 伪指令。

用户响应:
从 BOPT 伪指令中除去 IN 或 OUT 说明符，直至每个伪指令中不存在一个以上的说明符。

消息返回码

12

BOZ4121S 日志规范无效。“&1”不是目录。

说明

只有目录是 LOG 选项的有效说明符。&1 不是目录。

系统操作:
忽略 LOG 选项。

用户响应:
将 LOG 规范更正为受支持的格式，请参阅 LOG 选项以获取受支持的规范。

消息返回码

12

BOZ4124I **HANDLERS** 选项现在已不推荐使用并且不再需要；先前的缺省 **HANDLERS=Y** 行为现在始终适用。

说明

优化器选项 HANDLERS 现在已不推荐使用并且不再受支持也不再需要。

系统操作:
无论 HANDLERS 选项如何指定，优化器都将始终如先前缺省的传统 HANDLERS=Y 选项生效时一样运行。

用户响应:
用户不需要执行任何操作。

消息返回码

0

声明

本信息是为在美国国内供应的产品和服务而编写的。IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您所在区域当前可获得的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务的操作，由用户自行负责。

IBM 可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面形式将许可查询寄往：

IBM Corporation
J74/G4
555 Bailey Avenue
San Jose, CA 95141-1099
U.S.A.

有关双字节 (DBCS) 信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往：

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
3-2-12, Roppongi, Minato-ku, Tokyo 106-8711

以下段落不适用于英国或任何此类条款与当地法律不一致的国家或地区：

International Business Machines Corporation “按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。

某些国家或地区在某些交易中不允许免除明示或默示的保证。因此本条款可能不适用于您。

本信息可能包含技术方面不够准确的地方或印刷错误。

本信息将定期更改；这些更改将编入本信息的新版本中。IBM 可以随时对本出版物中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本出版物中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

商标

IBM、IBM 徽标和 ibm.com 是 International Business Machines Corp. 在全球许多管辖区域注册的商标或注册商标。其他产品和服务名称可能是 IBM 或其他公司的商标。Web 站点 [“Copyright and trademark information”](#) 部分中提供了 IBM 商标的最新列表。

资源列表

IBM Automatic Binary Optimizer for z/OS 出版物

您可以在[修订列表和新功能部件页面](#)上找到有关 IBM Automatic Optimizer for z/OS APAR 和 PTF 的最新且最完整的信息。

您可以在 [IBM Automatic Binary Optimizer for z/OS 文档库](#) 中找到以下出版物：

- 《用户指南》，SC43-5134-00
- 《程序目录》，GI13-4513-04

相关出版物

z/OS 出版物

您可以在 [z/OS Internet Library](#) 中找到以下出版物。

- *Initialization and Tuning Reference* (SA23-1380) 包含有关参数库成员 IEFOPZxx 的信息。
- *Program Management: Advanced Facilities* (SA23-1392) 包含有关绑定程序 API 返回码和原因码的信息。
- *System Management Facilities (SMF)* (SA38-0667) 包含有关 SMF 记录 90 子类型 38（用于捕获 IEFOPZ 配置）的信息。
- *System Messages, Volume 8* (SA38-0675) 包含有关消息的信息。

Enterprise COBOL for z/OS 出版物

您可以在 [Enterprise COBOL for z/OS 文档库](#) 中找到以下出版物。

- *Customization Guide* (SC27-8712) 包含可帮助您在 z/OS 下规划和定制 Enterprise COBOL 的信息。
- *Language Reference* (SC27-8713) 包含有关为 IBM COBOL 编译器编写程序所需的 COBOL 语言和参考的信息。
- *Programming Guide* (SC27-8714) 包含可帮助您编写、编译和调试程序以及类的信息和示例。
- *Migration Guide* (GC27-8715) 包含可帮助您移至最新版本的 IBM Enterprise COBOL 的信息。
- *Performance Tuning Guide* (SC27-9202) 指出了使用 IBM Enterprise COBOL for z/OS 时的关键性能优势和优化注意事项。
- *Messages and Codes* (SC27-4648) 可帮助您了解编译器和预处理器消息。

Application Delivery Foundation for z Systems 出版物

您可以在 [IBM Knowledge Center](#) 中找到以下出版物。

- *IBM Application Performance Analyzer for z/OS User's Guide* (SC27-8403) 包含可帮助识别系统约束和提高应用程序性能的信息。
- *IBM Developer for z Systems* 文档（仅限联机版本）包含有关 Integrated Development Environment (IDE) 的信息，旨在提高开发人员生产效率。
- *IBM Fault Analyzer for z/OS User's Guide and Reference* (SC19-4116) 包含有关分析和修复应用程序和系统故障的信息。



SC43-5134-00

