IBM

SHARE Anaheim 2014
High Level Assembler Bootcamp
HLASM Listings
(15250, 15251, 15252)

Thanks

Many thanks to Sharuff Morsa (IBM HLASM Architect) for preparing these slides.

HLASM Listing: Sections

The HLASM listing is divided into:

- High Level Assembler Option Summary
- External Symbol Dictionary
- Source Statement
- Relocation Dictionary
- Ordinary Symbol and Literal Cross Reference
- Unreferenced Symbols Defined in CSECTs
- Macro and Copy Code Source Summary
- Macro and Copy Code Cross Reference
- Dsect Cross Reference
- Using Map
- General Purpose Register Cross Reference
- Diagnostic Cross Reference and Assembler Summary

High Level Assembler Option Summary

HLASM options can be specified in:

*PROCESS OVERRIDE
ASMAOPT
Invocation parm
*PROCESS
Installation defaults

High Level Assembler Option Summary ...

Our example specifies:

In source:

```
*process override(adata,mxref(full))
*process align
*process nodbcs
*process mxref(full),nolibmac
*process flag(0)
*process nofold,language(ue)
*process nora2
*process nodbcs
*process xref(full)
```

In jcl procedure - parms:

```
OPTS1='NOOBJECT,language(en),size(4meg)',
OPTS2='xref(short,unrefs)',
OPTS3='nomxref,norxref,adata,noadata'
//C    EXEC PGM=ASMA90,
//     PARM='&OPTS1,&OPTS2,&OPTS3'
```

In ASAMOPT DD

```
//ASMAOPT  DD *
* My ASMAOPTS Overides
sysparm(thisisatestsysparm)     pass this to assembler
goff                            create GOFF object code
/*
```

High Level Assembler Option Summary ...

```
                          High Level Assembler Option Summary                 (PTF R160    )   Page    1
                                                                             HLASM R6.0  2013/11/01 13.45
  Overriding ASMAOPT Parameters -
    >* My ASMAOPTS Overides
    >sysparm(thisisatestsysparm)     pass this to assembler
    >goff                            create GOFF object code
  Overriding Parameters-  NOOBJECT,language(en),size(4meg),xref(short,unrefs),nomxref,norxref,adata,noadata
  Process Statements-     override(adata,mxref(full))
                          align
                          nodbcs
                          mxref(full),nolibmac
                          flag(0)
                          nofold,language(ue)
                          nora2
                          nodbcs
                          xref(full)
```

```
** ASMA434N GOFF/XOBJECT option specified, option LIST(133) will be used
** ASMA400W Error in invocation parameter - size(4meg)
** ASMA423N Option adata, in a *PROCESS OVERRIDE statement conflicts with invocation or default option.
Option is not permitted in a *PROCESS statement and has been ignored.
** ASMA422N Option language(ue) is not valid in a *PROCESS statement.
** ASMA437N Attempt to override invocation parameter in a *PROCESS statement. Suboption full of xref
option ignored.
```

High Level Assembler Option Summary ...

```
  Options for this Assembly
 3 Overriding Parms NOADATA
        5 *Process    ALIGN
                   NOASA
                   NOBATCH
                     CODEPAGE(047C)
                   NOCOMPAT
        5 *Process  NODBCS
                   NODECK
                     DXREF
                     ESD
                   NOEXIT
        5 *Process    FLAG(0,ALIGN,NOCONT,EXLITW,NOIMPLEN,NOPAGE0,PUSH,RECORD,NOSUBSTR,USING0)
        5 *Process  NOFOLD
        2  ASMAOPT    GOFF(NOADATA)
                   NOINFO
 3 Overriding Parms    LANGUAGE(EN)
        5 *Process  NOLIBMAC
                     LINECOUNT(60)
                     LIST(133)
                     MACHINE(,NOLIST)
1 *Process Override    MXREF(FULL)
 3 Overriding Parms NOOBJECT
                     OPTABLE(UNI,NOLIST)
                   NOPCONTROL
                   NOPESTOP
                   NOPROFILE
        5 *Process  NORA2
                   NORENT
                     RLD
 3 Overriding Parms NORXREF
                     SECTALGN(8)
                     SIZE(MAX)
                     TYPECHECK(MAGNITUDE,REGISTER)
                     USING(NOLIMIT,MAP,NOWARN)
                   NOWORKFILE
 3 Overriding Parms    XREF(SHORT,UNREFS)
```

External Symbol Dictionary

```
                                External Symbol Dictionary

Symbol   Type   Id     Address  Length   Owner Id Flags Alias-of
LISTINGB  SD 00000001
B_IDRL    ED 00000002                    00000001
B_PRV     ED 00000003                    00000001
B_TEXT    ED 00000004 00000000 00000084  00000001   08
LISTINGB  LD 00000005 00000000           00000004   08
EXTERNAL_FUNCTION
          ER 00000006                    00000001
FUNCY     ER 00000007                    00000001
listme    ER 00000008                    00000001       LISTINGZ
COMMON_DATA
          SD 00000009
B_IDRL    ED 0000000A                    00000009
B_PRV     ED 0000000B                    00000009
B_TEXT    ED 0000000C 00000000 00000018  00000009   00
COMMON_DATA
          CM 0000000D 00000000           0000000C   00
```

This section of the listing contains the external symbol dictionary information passed to the Binder

External Symbol Dictionary ...

SD Control section definition. The symbol appeared in the name field of a START, CSECT, or RSECT instruction

LD Label definition. The symbol appeared as the operand of an ENTRY statement. When you specify the GOFF assembler option, on z/OS or CMS, the assembler generates an entry type of LD for each CSECT and RSECT name.

ER External reference. The symbol appeared as the operand of an EXTRN statement, or appeared as an operand of a V-type address constant

CM Common control section definition. The symbol appeared in the name field of a COM statement.

Source Statement

This section of the listing documents source statements of the module and the resulting object code

```
                                                  360 *
00000034                                          361 continue_code_again dc 0h'0'    define a label
00000034 9836 2014                 00000014       362          lm  r3,r6,l_regs       reload registers
00000038 58B0 A038                 00000080       363          l   r11,=f'1504'       load constant
0000003C 50B0 3014                 00000014       364          st  r11,comm_country   ... and save in common
00000040 58C0 A028                 00000070       365          l   r12,pDateFormat    load address constant
00000044 07FE                                     366          br  r14                and exit program
                                                  367 *


                                                                                                      Page    4
  Active Usings: linkage_data(X'54'),R2  common_data,R3  static_data(X'3C'),R10
R-Loc    Object Code      Addr1    Addr2    Stmt  Source Statement                  HLASM R6.0  2013/11/01 15.23
00000046 0000
00000048                                          369 static_data dc 0d'0'           start of static
00000048 839697A899898788                         370 copyright   dc c'copyright IBM(UK) Ltd 2013'
00000062 0000
00000064 00000000                                 371 listingx    dc v(external_function)     declare external
00000068 E8E8E8E8D4D4C4C4                         372 dateFormat  dc cl(date_len)'YYYYMMDD'   define format
```

Page titles

Source Statement ...

```
                                                                 360 *
00000034                                                         361 continue_code_again dc 0h'0'    define a label
00000034 9836 2014                 00000014                      362          lm  r3,r6,l_regs       reload registers
00000038 58B0 A038                 00000080                      363          l   r11,=f'1504'       load constant
0000003C 50B0 3014                 00000014                      364          st  r11,comm_country   ... and save in common
00000040 58C0 A028                 00000070                      365          l   r12,pDateFormat    load address constant
00000044 07FE                                                    366          br  r14                and exit program
                                                                 367 *


                                                                                                                Page    4
  Active Usings: linkage_data(X'54'),R2  common_data,R3  static_data(X'3C'),R10
R-Loc    Object Code      Addr1    Addr2    Stmt  Source Statement                HLASM R6.0  2013/11/01 15.23
00000046 0000
00000048                                                         369 static_data dc 0d'0'            start of static
00000048 839697A899898788                                        370 copyright   dc c'copyright IBM(UK) Ltd 2013'
00000062 0000
00000064 00000000                                                371 listingx    dc v(external_function)     declare external
00000068 E8E8E8E8D4D4C4C4                                        372 dateFormat  dc cl(date_len)'YYYYMMDD'   define format
```

Location counter and statement number

Source Statement ...

```
  Active Usings: linkage_data(X'54'),R2  common_data,R3  static_data(X'3C'),R10
R-Loc    Object Code      Addr1    Addr2    Stmt  Source Statement                HLASM R6.0  2013/11/01 15.23

00000074 00000000                            374 pZERO       dc a(0)
00000078 00000000                            375 ListData    dc v(FUNCY)
                                             376          extrn listingz
                                             377 listingz alias c'listme'
                                             378 *
00000080                                     379          ltorg  ,
00000080 000005E0                            380               =f'1504'
                                             381 *
                      00000084               382 static_data_end equ *
                                             383 *
                                             384          drop  r2
                                             385          drop  r3
                                             386 *
00000000              00000000 00000018      387 common_data com    ,
00000000 A8A8A8A894948484                    388 comm_date    dc cl(date_len)'yyyy....
00000008 A8A8A8A894948484                    389 comm_user    dc cl(10)'yyyymmdd'
00000012 0000
00000014 00000000                            390 comm_country dc f'0000'
                                             391 *
00000000              00000000 00000054      392 linkage_data dsect ,
00000000 C140D58194854040                    393 l_name  dc cl(10)'A Name'
0000000A A8A8A8A894948484                    394 l_date  dc cl(date_len)'yyyymmdd'
```

Location counter and statement number ...

Source Statement ...

```
                          19          print on,gen                print statements
                          20          sysstate archlvl=1          set arch level
                          21+*        THE VALUE OF SYSSTATE IS NOW SET TO ASCENV=P AMODE64=NO
                            +               L=1 OSREL=00000000
                          22          ieabrcx  DEFINE             use relative branching
          00000048        341         larl  r10,static_data
00000048                  342         using (static_data,static_data_end),r10
                          343 *
```

The column following the statement number contains one of these values:

A space ( ) indicates open source

A plus sign (+) indicates that the statement was generated as the result of macro call processing.

An unnumbered statement with a plus sign (+) is the result of open code substitution.

A minus sign (-) indicates that the statement was read by a preceding AREAD instruction.

An equals sign (=) indicates that the statement was included by a COPY instruction.

A greater-than sign (>) indicates that the statement was generated as the result of a preceding AINSERT instruction. If the statement is read by an AREAD instruction, this takes precedence and a minus sign is printed.

Source Statement ...

```
R-Loc    Object Code      Addr1    Addr2    Stmt  Source Statement
                                               21+*        THE VALUE OF SYSSTATE IS NOW SET TO ASCENV=P..
                                               22          ieabrcx  DEFINE            use relative branching
00000000 C0A0 0000 0024           00000048    341          larl  r10,static_data
                    R:A 00000048              342          using (static_data,static_data_end),r10
                                              343 *
00000006 17FF                                 344          xr  r15,r15           initialise register
                    R:2 00000000              345          using (linkage_data,l_end),r2   set using scope
                    R:3 00000000              346          using common_data,r3            set using scope
00000008 C040 0000 000A           0000001C    347          larl  r4,address_constant       address of....
0000000E D207 3000 200A 00000000  0000000A    348          mvc   comm_date,l_date   copy
00000014 9036 2014                00000014    349          stm   r3,r6,l_regs       save a copy of the reg..
00000018 A7F4 0008                00000028    350          j     continue_code      jump over constant
                                              351 *
0000001C C1C4C4D9C5E2E240                     352 address_constant dc cl12'ADDRESS'   constant value
```

Addr1 and Addr2

USING assignments

Relocation Dictionary

```
                                                  Relocation Dictionary
   Pos.Id   Rel.Id   Address  Type  Action
00000004 00000004 00000070   A 4     +
00000004 00000006 00000064   V 4    ST
00000004 00000007 00000078   V 4    ST
```

```
00000048 839697A899898788                        370 copyright   dc c'copyright IBM(UK) Lt
00000062 0000
00000064 00000000                                371 listingx    dc v(external_function)
00000068 E8E8E8E8D4D4C4C4                        372 dateFormat  dc cl(date_len)'YYYYMMDD'
00000070 00000068                                373 pDateFormat dc a(dateFormat)       po
00000074 00000000                                374 pZERO       dc a(0)                po
00000078 00000000                                375 ListData    dc v(FUNCY)            de
                                                 376          extrn listingz            de
```

Location counter 70, and ADCON symbol pDateFormat has a value of 00000068 – which is the location counter for symbol dateFormat

This section of the listing describes the relocation dictionary information passed to the Binder.

The entries describe the address constants in the assembled program that are affected by relocation.

This section helps you find relocatable constants in your program.

Ordinary Symbol and Literal Cross Reference

```
                                Ordinary Symbol and Literal Cross Reference
Symbol    Length    Value     Id    R Type Asm  Program   Defn References
address_constant
              12 0000001C 00000004     C  C                352  347
comm_country
               4 00000014 0000000C     F  F                390  364M
comm_date
               8 00000000 0000000C     C  C                388  348M
common_data
               1 00000000 0000000C     J                   387  346U
continue_code
               2 00000028 00000004     H  H                354  350B
continue_code_again
               2 00000034 00000004     H  H                361  357B
date_len       1 00000008 00000004 A   U                     4  372   388   394


r10            1 0000000A 00000004 A   U  GR32              12  341M  342U
r11            1 0000000B 00000004 A   U  GR32              13  363M  364
r12            1 0000000C 00000004 A   U  GR32              14  365M
r14            1 0000000E 00000004 A   U  GR32              16  366B
r15            1 0000000F 00000004 A   U  GR32              17  344M  344
```

Symbol name , length, value, Assembler type, Definition

References – no suffix, **B**ranch, **M**odified, **U**sing, **D**rop and e**X**ecution

This section of the listing concerns symbols and literals that are defined and used in the program.

Unreferenced Symbols Defined in CSECTs

```
                                Unreferenced Symbols Defined in CSECTs
Defn  Symbol
 370  copyright
 375  ListData
 371  listingx
 374  pZERO
   6  r0
   7  r1
  15  r13
```

This section of the listing shows symbols that have been defined in CSECTs but not referenced. This may help you to remove unnecessary data definitions, and reduce the size of your program.

Macro and Copy Code Source Summary

```
                          Macro and Copy Code Source Summary                                  Page    8
 Con Source                          Volume    Members                          HLASM R6.0  2013/11/01 15.23
     PRIMARY INPUT                           B         BAL       BAS       BC        BCT       BE        BH
                                             BL        BM        BNE       BNH       BNL       BNM       BNO
                                             BNP       BNZ       BO        BP        BXH       BXLE      BZ
  L2 SYS1.MACLIB                     37SY01    IEABRC    IEABRCX   SYSSTATE
```

In section 'Diagnostic Cross Reference and Assembler Summary'

```
Data Sets Allocated for this Assembly
 Con DDname   Data Set Name                                    Volume  Member
  A1 ASMAOPT  SMORSA.LISTINGC.JOB63822.D0000101.?
  P1 SYSIN    SMORSA.ASM.ASM                                   37P001  LISTINGC
  L1 SYSLIB   SMORSA.ASM.ASM                                   37P001
  L2          SYS1.MACLIB                                      37SY01
     SYSLIN   SYS13305.T152320.RA000.LISTINGC.OBJ.H01
     SYSPRINT SMORSA.LISTINGC.JOB63822.D0000102.?
```

This section of the listing shows the names of the macro libraries from which the assembler read macros or copy code members, and the names of the macros and copy code members that were read from each library.

Macro and Copy Code Cross Reference

```
                                                Macro and Copy Code Cross Reference
Macro      Con   Called By          Defn  References
B                PRIMARY INPUT       117  355
BAL                                  273
BAS                                  297
BC               PRIMARY INPUT       177  356
BCT                                  201
BE                                   121
BH                                   129
BL                                   125
BM                                   133
BNE                                  137
....
BXLE                                 225
BZ                                   173
IEABRC     L2    IEABRCX               -  22C
IEABRCX    L2    PRIMARY INPUT         -  22
SYSSTATE   L2    PRIMARY INPUT         -  20
```

```
                19          print on,gen               print statements
                20          sysstate archlvl=1         set arch level
                21+*        THE VALUE OF SYSSTATE IS NOW SET TO ASCENV=P AMODE64=N
                  +               L=1 OSREL=00000000
                22          ieabrcx  DEFINE            use relative branching
               341          larl  r10,static_data
               342          using (static_data,static_data_end),r10
```

This section of the listing shows the names of macros and copy code members and the statements where the macro or copy code member was called.

Dsect Cross Reference

```
                                                  Dsect Cross Reference
Dsect     Length      Id       Defn
linkage_data
          00000054  FFFFFFFF     392
```

```
                                                  391 *
00000000                  00000000 00000054       392 linkage_data dsect ,                data passed in to me
00000000 C140D58194854040                         393 l_name  dc cl(10)'A Name'           users name
0000000A A8A8A8A894948484                         394 l_date  dc cl(date_len)'yyyymmdd'   users date
00000012 0000
00000014 0000000000000000                         395 l_regs  dc 4f'0,0,0,0'             return registers values
                         00000054                 396 l_end   equ *                       end
                                                  397 *
```

This section of the listing shows the names of all internal or external dummy sections defined in the program, and the number of the statement where the definition of the dummy section began.

Using Map

```
                                                    Using Map
 Stmt  -----Location-----  Action  ----------------Using-----------------
         Count      Id             Type          Value    Range     Id
  342  00000006  00000004 USING   ORDINARY    00000048 0000003C 00000004
  345  00000008  00000004 USING   ORDINARY    00000000 00000054 FFFFFFFF
  346  00000008  00000004 USING   ORDINARY    00000000 00001000 0000000C
  384  00000084  00000004 DROP
  385  00000084  00000004 DROP
```

```
Stmt  -----Location-----  Action  ------------..... Reg Max     Last Label and Using Text
        Count      Id             Type          .....     Disp    Stmt
 342  00000006  00000004 USING   ORDINARY    ..... 10 00038    365 (static_data,static_data_end),r10
 345  00000008  00000004 USING   ORDINARY    .....  2 00014    362 (linkage_data,l_end),r2
 346  00000008  00000004 USING   ORDINARY    .....  3 00014    364 common_data,r3
 384  00000084  00000004 DROP                .....  2             r2
 385  00000084  00000004 DROP                .....  3             r3
```

```
        Addr1    Addr2    Stmt  Source Statement
                00000048   341          larl  r10,static_data
  R:A 00000048             342          using (static_data,static_data_end),r10


  Active Usings: linkage_data(X'54'),R2  common_data,R3  static_data(X'3C'),R10
R-Loc    Object Code      Addr1    Addr2    Stmt  Source Statement
00000046 0000
00000048                                     369 static_data dc 0d'0'            start of
00000048 839697A899898788                    370 copyright   dc c'copyright IBM(UK) Ltd 2
00000062 0000
00000064 00000000                            371 listingx    dc v(external_function)
```

This section of the listing shows a summary of the USING, DROP, PUSH USING, and POP USING instructions used in your program.

General Purpose Register Cross Reference

```
                                 General Purpose Register Cross Reference
 Register  References (M=modified, B=branch, U=USING, D=DROP, N=index)
    0(0)    (no references identified)
    1(1)    (no references identified)
    2(2)     345U  384D
    3(3)     346U  349   362M  385D
    4(4)     347M  349   362M
    5(5)     349   362M
    6(6)     349   362M
    7(7)    (no references identified)
    8(8)    (no references identified)
    9(9)    (no references identified)
   10(A)     341M  342U
   11(B)     363M  364
   12(C)     365M
   13(D)    (no references identified)
   14(E)     366B
   15(F)     344M  344
```

Register 10 – no DROP statement?

```
  Loc    Object Code      Addr1    Addr2    Stmt  Source Statement
00000000 C0A0 0000 0024           00000048   341          larl  r10,static_data
                  R:A 00000048               342          using (static_data,static_data_end),r10
```

Register 5?

```
00000014 9036 2014               00000014    349          stm   r3,r6,l_regs

00000034 9836 2014               00000014    362          lm  r3,r6,l_regs
```

This section of the listing shows all references in the program to each of the general registers. Additional flags indicate the type of reference.

Diagnostic Cross Reference and Assembler Summary

```
                                Diagnostic Cross Reference and Assembler Summary

Statements Flagged
    24(P1,24)

      1 Statement  Flagged in this Assembly        8 was Highest Severity Code
HIGH LEVEL ASSEMBLER, 5696-234, RELEASE 6.0, PTF R160
SYSTEM: z/OS 01.13.00              JOBNAME: LISTINGB    STEPNAME: B          PROCSTEP: C
```

```
R-Loc    Object Code      Addr1    Addr2    Stmt  Source Statement
0000000C 50B0 3014               00000014     22          st  r11,comm_country
00000010 58C0 F040               00000040     23          l   r12,pDateFormat
00000014 0000 0000               00000000     24          l   r7,someData
** ASMA044E Undefined symbol - r7
** ASMA029E Incorrect register specification - r7
** ASMA044E Undefined symbol - someData
** ASMA435I Record 24 in SMORSA.ASM.ASM(LISTINGB) on volume: 37P001
00000018 07FE                                    25          br  r14
                                                 26 *
```

```
Data Sets Allocated for this Assembly
 Con DDname   Data Set Name                                    Volume  Member
  A1 ASMAOPT  SMORSA.LISTINGB.JOB63825.D0000101.?
  P1 SYSIN    SMORSA.ASM.ASM                                   37P001  LISTINGB
  L1 SYSLIB   SMORSA.ASM.ASM                                   37P001
  L2          SYS1.MACLIB                                      37SY01
     SYSLIN   SYS13305.T173626.RA000.LISTINGB.OBJ.H01
     SYSPRINT SMORSA.LISTINGB.JOB63825.D0000102.?
```

This section of the listing summarises the error diagnostic messages issued during the assembly, and provides statistics about the assembly.

Diagnostic Cross Reference and Assembler Summary ...

```
                                Diagnostic Cross Reference and Assembler Summary

     No Statements Flagged in this Assembly
HIGH LEVEL ASSEMBLER, 5696-234, RELEASE 6.0, PTF R160
SYSTEM: z/OS 01.13.00              JOBNAME: LISTINGC    STEPNAME: B          PROCSTEP: C
Data Sets Allocated for this Assembly
 Con DDname   Data Set Name                                  Volume  Member
  A1 ASMAOPT  SMORSA.LISTINGC.JOB63822.D0000101.?
  P1 SYSIN    SMORSA.ASM.ASM                                 37P001  LISTINGC
  L1 SYSLIB   SMORSA.ASM.ASM                                 37P001
  L2          SYS1.MACLIB                                    37SY01
     SYSLIN   SYS13305.T152320.RA000.LISTINGC.OBJ.H01
     SYSPRINT SMORSA.LISTINGC.JOB63822.D0000102.?

   4096K allocated to Buffer Pool       Storage required     200K
     76 Primary Input Records Read         1093 Library Records Read                0 Work File Reads
      2 ASMAOPT Records Read                312 Primary Print Records Written       0 Work File Writes
     23 Object Records Written                0 ADATA Records Written
Assembly Start Time: 15.23.20 Stop Time: 15.23.20 Processor Time: 00.00.00.0044
Return Code 000
```

Where can I get help?

- z/OS V2R1 Elements and Features
  http://www.ibm.com/systems/z/os/zos/bkserv/v2r1pdf/#IEA
- HLASM Programmer's Guide (SC26-4941-06)
  http://publibz.boulder.ibm.com/epubs/pdf/asmp1021.pdf
- HLASM Language Reference (SC26-4940-06)
  http://publibz.boulder.ibm.com/epubs/pdf/asmr1021.pdf

Source: LISTINGA

```
*process override(adata,mxref(full))
*process align
*process nodbcs
*process mxref(full),nolibmac
*process flag(0)
*process nofold,language(ue)
*process nora2
*process nodbcs
*process xref(full)
* copyright IBM(UK) Ltd 2013
*
listinga csect
*
r14      equ 14,,,,gr32  define register equate
r15      equ 15,,,,gr32  define register equate
*
         using *,r15 set using
         xr  r15,r15     initialise register
         br  r14         and exit program
*
copyright dc c'copyright IBM(UK) Ltd 2013'
listingx dc  v(external_function)  declare external function
ListData dc  v(FUNCY)              declare external function
         extrn listingz            declare alias as external
listingz alias c'listme'           declare alias
*
         end listinga
```

Source: LISTINGB

```
* copyright IBM(UK) Ltd 2013
listingb rsect
*
date_len equ 8           length of date field
*
r0       equ 0,,,,gr32   define register equate
r1       equ 1,,,,gr32   define register equate
r2       equ 2,,,,gr32   base register for linkage
r3       equ 3,,,,gr32   base register for common
r11      equ 11,,,,gr32  define register equate
r12      equ 12,,,,gr32  define register equate
r13      equ 13,,,,gr32  define register equate
r14      equ 14,,,,gr32  define register equate
r15      equ 15,,,,gr32  base register for program
*
         using *,r15           set using scope
         xr  r15,r15           initialise register
         using (linkage_data,l_end),r2   set using scope
         using common_data,r3            set using scope
         mvc comm_date,l_date   copy
         l   r11,=f'1504'       load constant
         st  r11,comm_country   ... and save in common
         l   r12,pDateFormat    load address constant
         l   r7,someData        load into register 7
         br  r14                and exit program
*
         EJECT                  force page eject
copyright dc c'copyright IBM(UK) Ltd 2013'
listingx dc  v(external_function)  declare external function
dateFormat dc cl(date_len)'YYYYMMDD'   define format
pDateFormat dc a(dateFormat)       pointer to format constant
ListData dc  v(FUNCY)              declare external function
         extrn listingz            declare alias as external
listingz alias c'listme'           declare alias
*
         ltorg  ,                  literal pool
*
```

Source: LISTINGB ...

```
         drop  r2
         drop  r3
         drop  r15
*
common_data com   ,                 define common section
comm_date    dc cl(date_len)'yyyymmdd'   com date
comm_user    dc cl(10)'yyyymmdd'   com user
comm_country dc f'0000'            define country code
*
linkage_data dsect ,                data passed in to me
l_name  dc cl(10)'A Name'           users name
l_date  dc cl(date_len)'yyyymmdd'   users date
l_end   equ *                       end
*
         end listingb
```

Source: LISTINGC

```
* copyright IBM(UK) Ltd 2013
listingc rsect
*
date_len equ 8           length of date field
*
r0       equ 0,,,,gr32   define register equate
r1       equ 1,,,,gr32   define register equate
r2       equ 2,,,,gr32   base register for linkage
r3       equ 3,,,,gr32   base register for common
r4       equ 4,,,,gr32   address constant
r6       equ 6,,,,gr32   register
r10      equ 10,,,,gr32  base register for static data
r11      equ 11,,,,gr32  define register equate
r12      equ 12,,,,gr32  define register equate
r13      equ 13,,,,gr32  define register equate
r14      equ 14,,,,gr32  define register equate
r15      equ 15,,,,gr32  base register for program
*
         print on,gen              print statements
         sysstate archlvl=1        set arch level
         ieabrcx  DEFINE           use relative branching
         larl  r10,static_data
         using (static_data,static_data_end),r10
*
         xr  r15,r15          initialise register
         using (linkage_data,l_end),r2   set using scope
         using common_data,r3            set using scope
         larl  r4,address_constant       address of....
         mvc   comm_date,l_date   copy
         stm   r3,r6,l_regs       save a copy of the registers
         j     continue_code      jump over constant
*
address_constant dc cl12'ADDRESS'   constant value
*
continue_code dc 0h'0'          define a label
         b     continue_code_again branch over constant
*
         dc c'ADDRESS_'         another constant value
*
```

Source: LISTINGC

```
continue_code_again dc 0h'0'    define a label
         lm  r3,r6,l_regs       reload registers
         l   r11,=f'1504'       load constant
         st  r11,comm_country   ... and save in common
         l   r12,pDateFormat    load address constant
         br  r14                and exit program
*
         EJECT                  force page eject
static_data dc 0d'0'            start of static
copyright   dc c'copyright IBM(UK) Ltd 2013'
listingx    dc v(external_function)     declare external function
dateFormat  dc cl(date_len)'YYYYMMDD'   define format
pDateFormat dc a(dateFormat)        pointer to format constant
pZERO       dc a(0)                 pointer
ListData    dc v(FUNCY)             declare external function
         extrn listingz             declare alias as external
listingz alias c'listme'            declare alias
*
         ltorg  ,                   literal pool
*
static_data_end equ *
*
*
         drop  r2
         drop  r3
*
common_data com   ,                 define common section
comm_date    dc cl(date_len)'yyyymmdd'   com date
comm_user    dc cl(10)'yyyymmdd'    com user
comm_country dc f'0000'             define country code
*
linkage_data dsect ,                data passed in to me
l_name  dc cl(10)'A Name'           users name
l_date  dc cl(date_len)'yyyymmdd'   users date
l_regs  dc 4f'0,0,0,0'              return registers values
l_end   equ *                       end
*
         end listingc
```