SHARE
Technology · Connections · Results

z390 and zCOBOL Portable Mainframe Assembler and COBOL with zCICS Support

Don Higgins don@higgins.net & Melvyn Maltz
Automated Software Tools Corporation

Tuesday, March 3, 2009 8:00-9:00 AM
Session Number 8194 – Hilton Salon F

Trademark Acknowledgements

- IBM Corporation
  - z/OS, HLASM, CICS, VSAM
- Microsoft Corporation
  - Windows Vista, XP, and 2000
  - Visual Express C++
- Sun Microsystems
  - J2SE, J2RE

Presentation Outline

- z390 Portable Mainframe Assembler v1.5.00
  - Assemble, link, execute HLASM compatible programs
- zCOBOL V1 Portable Mainframe COBOL (v1.5.00)
  - Compile, link, execute COBOL programs
- zCICS V7 Support by Melvyn Maltz (v1.5.00)
  - Support EXEC CICS COBOL and assembler
  - Run local and remote TN3270 CICS trans. over TCP/IP
- Demonstrations
- Questions and Answers

z390 Portable Mainframe Assembler

- z390 Open Source Java Project
- Execute HLASM compatible macro code
- Assemble HLASM compatible programs
- Link object code into z390 load modules
- Execute load modules on J2SE platforms:
  - Windows and Linux hosts
  - 24 and 31 bit AMODE/RMODE
  - 16 – 64 bit GPR/FPR, HFP/BFP/DFP
  - QSAM, VSAM, SOA, CICS, TN3270

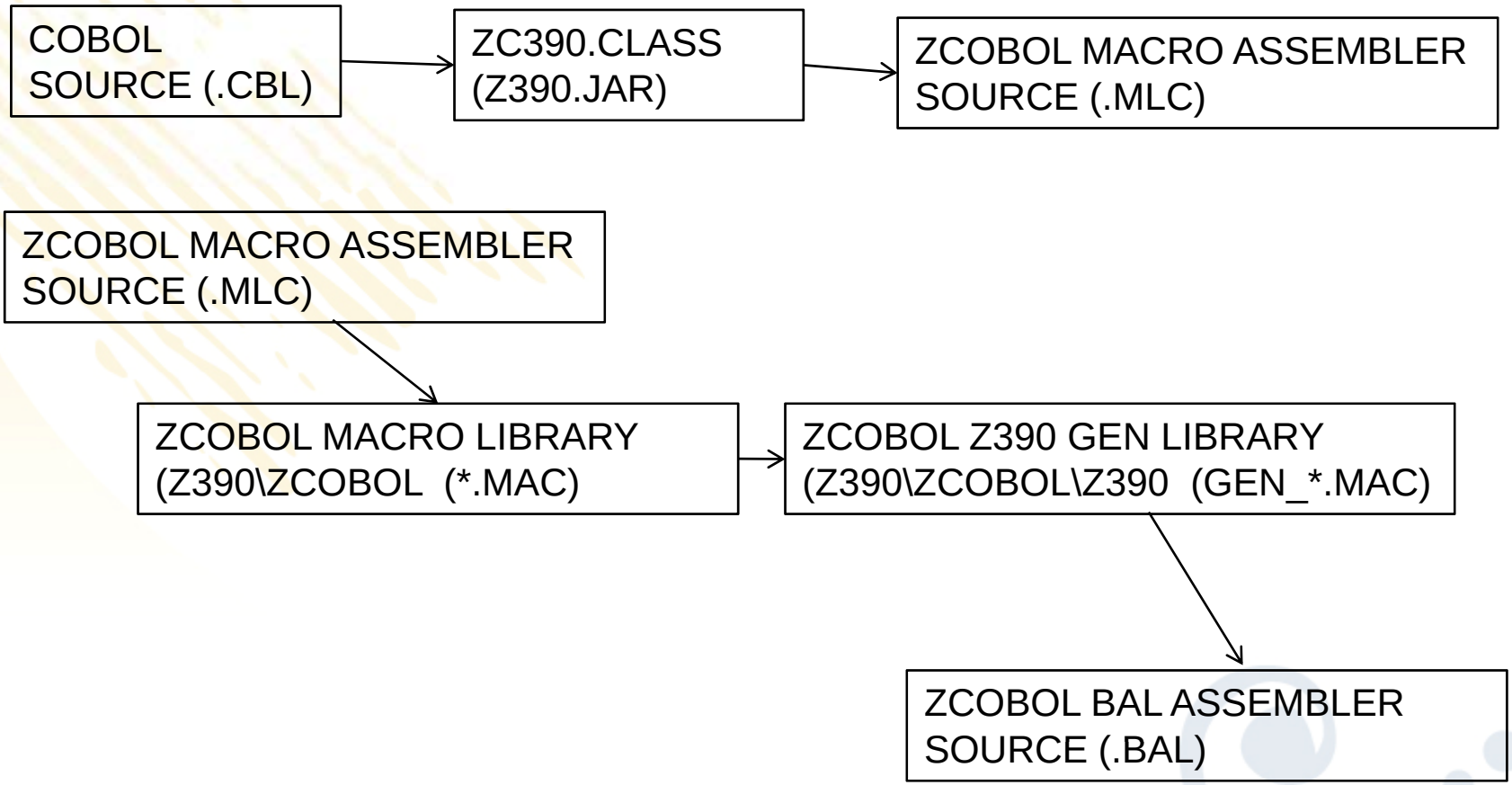
What's new in z390 since last SHARE

- ZSTRMAC structured conditional macro code
  - Supports AIF, AWHILE, ASELECT, ACASE, AEND
  - Built into mz390, translator for HLASM portability
- zcobol portable mainframe COBOL
  - Written in ZSTRMAC conditional macro assembler
  - Generates HLASM for z390 or native z9/z10 execution
  - Supports static & dynamic linking of COBOL & HLASM
- zCICS V7 support by Melvyn Maltz
  - Compile and execute EXEC CICS COBOL on z390
  - Run local and remote TN3270 CICS COBOL transactions

zCOBOL Portable Mainframe COBOL

- Compiler architecture
- Compiler examples of source code generation
- Compiler symbol table and system functions
- Compiler register allocation
- Compiler code generation
- Compiler commands
- Demo and regression test programs
- zCOBOL development priorities and RPI's
- Questions and Answers

zCOBOL Portable Mainframe COBOL



zCOBOL Compiler Architecture

- zc390.java parser CBL to MLC macro assembler
- zcobol library for all COBOL verb macros (139)
- zcobol\z390 library for all HLASM gen macros (102)
- zcobol\java for all java code gen macros (11)*
- zcobol\vce for all C++ code gen macros (11)*
- zcobol\i586 for all HLA/MASM gen macros (11)*

- Note once the z390 code gen macros are stabilized, they can all be copied to other target language libraries and modified to gen other source code.

zCOBOL to z390 code gen example 1

COBOL SOURCE:

```
77  CTR-1 COMP PIC S9(9)..


01  SYSTEM-DATE.
    02  SYSTEM-DD PIC 99.
    02  SYSTEM-MM PIC 99.
```

HLASM > MACROS > BAL:

```
WS 77,CTR_1,COMP,PIC,S9(9)
```

- GEN_WS
  - CTR_1 DS FL4

```
WS 01,SYSTEM_DATE
WS 02,SYSTEM_DD,PIC,99
WS 02,SYSTEM_MM,PIC,99
```

- GEN_WS
  - SYSTEM_DATE DS 0CL4
  - SYSTEM_DD DS ZL2
  - SYSTEM_MM DS ZL2

zCOBOL to z390 code gen example 2

IF CTR-1 = 2 GO TO OPT-2.

IF CTR_1,=,2

- **GEN_COMP**
  - **L R0,CTR_1**
  - **CHI R0,2**
- **GEN_BC 7,PG_IF_1**
  - **BRC 7,PG_IF_1**

GO TO,OPT_2

- **GEN_B PG_OPT_2**
  - **J PG_OPT_2**

PERIOD

- **GEN_LABEL PG_IF_1,ENDIF**
  - **PG_IF_1 DS 0H ENDIF**

zCOBOL symbol table and functions

- Global symbol table copybook zcobol\ZC_WS.CPY
  - All the COBOL verb and code generation macros share global symbol table via COPY ZC_WS
- Symbol lookup macro zcobol\ZC_SYM_FIND.MAC
  - GBLA &(ZC_IX_&SYM),&SYM_IX
  - :&SYM_IX SETA &(ZC_IX_&SYM)
- Symbol reference function zcobol\ZCGETFLD.CPY
  - Return qualified symbol name to resolve duplicates
  - Call GEN_BASE.MAC to gen WS/LK base code if any
  - Call GEN_SIX.MAC to gen subscript/index code

zCOBOL to HLASM register allocation

- R0-R3 work within single COBOL statement
- R4-R5 bases for linkage section data items
- R6-R7 bases for working storage items as required
- R8 z390 initial code base for load, then WS#2
- R9 zcobol ZCVT with function call entries
- R10 z390 zCICS support DFHTCTTE
- R11 z390 zCICS support DFHEIBLK
- R12 z390 WS#3
- R13 save area in DFHEISTG for zCICS else WS#1
- R14 return address for calls
- R15 entry address for calls

zCOBOL to HLASM code generation

- CSECT with PROGRAM-ID name starts with code to dynamically load ZC390LIB.390
- R9 set to ZC390CVT which is at ZC390LIB entry
- R13 set to DFHEISTG for CICS or WS following procedure code with standard save area.
- Procedure code is base free
  - All branches use relative instructions
  - All literal references use LARL to even length literals
  - WS and LK base registers are set as required within COBOL sentences to provide RS/RX type access.

zCOBOL Sample z390 GEN_ADD code

```
.........
AENTRY ADD_NUM_LIT
ACASE (C2A('&SYM_PIC_TYPE(&TARGET)'))
.............
AWHEN C'H'
  LH    R0,&SYM_NAME(&TARGET)
  AHI   R0,&NUM
  STH   R0,&SYM_NAME(&TARGET)
AWHEN C'G'
 AIF (K'&NUM LE 2)
  AGSI  &SYM_NAME(&TARGET),&NUM
  .........
```

zCOBOL Compile Commands

- ZC390C – compile to z390 relocatable object code
- ZC390CL – compile and link z390 390 load module
- ZC390CLG – compile, link, and execute z390 pgm
- ZCJAVCLG – compile and execute J2SE java pgm
- ZCVCECLG – compile, link, and execute C++ pgm
- ZC586CLG – compile, link, and execute MASM pgm
- Note other system software requirements (all free):
- All require J2SE and z390 installs
- ZCVCECLG requires MS Visual Express C++ install
- ZC586CLG requires HLA and MASM installs

zCOBOL Demo compile and execute

- The COBOL HELLO.CBL ”Hello World” program:
- .....
- DISPLAY ”Hello World”
- STOP RUN.
- Commands to compile HELLO.CBL in each language
  - ZC390CLG zcobol\demo\HELLO > MLC > HELLO.390
  - ZCJAVCLG zcobol\demo\HELLO > JAVA > HELLO.class
  - ZCVCECLG zcobol\demo\HELLO > CPP > HELLO.exe
  - ZC586CLG zcobol\demo\HELLO > ASM > HELLO.exe

zCOBOL Demo HLASM generated code


```
* 000400     DISPLAY 'Hello World'.
            LA    R3,ZCVT_WORKAREA
            LARL  R0,=CL12'Hello World’
            LA    R1,11
            LA    R2,C'X'
            STM   R0,R2,0(R3)  SET DISPLAY LIST ENTRY
            OI    12-4(R3),X'80'  SET VL BIT
            LR    R1,R3
            L     R15,ZCVT_DISPLAY
            BASR  R14,R15
```

zCOBOL Demo and Regression Tests

- **Demos in zcobol\demo include:**
  - **HELLO.CBL - display ”Hello World”**
  - **DATETIME.CBL- display current time and date**
  - **COPYFILE.CBL- copy line sequential file**
- **Regression tests in zcobol\test include:**
  - **TESTCMP1 – test ADD, SUBTRACT, MULTIPLY, DIVIDE**
  - **TESTFUN1 – test functions NUMERIC, etc.**
  - **TESTIF1 – test IF ELSE ENDIF**
  - **TESTISP1 - test INSPECT TALLY, REPLACING, etc.**
  - **TESTMOV1 – test MOVE including EDIT for DISPLAY**
  - **TESTPM1 – test PERFORM THRU, TIMES, VARYING**
  - **TESTSIX1 - test 2 dimensional subscripting**

zCOBOL Documentation

- **All the zCOBOL documentation is on www.zcobol.org**
  - **Demo Programs**
  - **User Guide**
  - **NIST ANSI 85 COBOL Test Suite Results**
  - **Options**
  - **Regresstion Test Programs**
  - **zCOBOLGroup – join zcobol-subscribe@yahoogroups.com**

  All z390 and zCICS support documentation is on www.z390.org
  - **Download link for z390 which includes zCOBOL and zCICS**
  - **Support link to submit RPI's for fixes and enhancements**
  - **Documentation on assembler, linker, emulator, zCICS support**

zCOBOL Open Source Direction

- The zcobol user community will set direction
- Submit RPI's for fixes and enhancement requests
- Join zcobol user group for updates and Q/A
- Current major priorities are as follows:
  - NIST ANSI 85 test suite completion
  - VSAM alternate index support
  - SQL support
  - Java target language environment
  - C++ target language environment
  - Intel HLA/MASM native code language environment

z390 zCICS V7 Support by Melvyn Maltz

- **Melvyn Maltz**
  - **Worked closely with IBM on CICS**
  - **Major z390 contributions:**
    - **Documentation**
    - **Testing and debugging**
    - **UNREF utility**
  - **zCICS V7 Support with zCOBOL and VSAM Browsing**
- **Please welcome Melvyn Maltz**

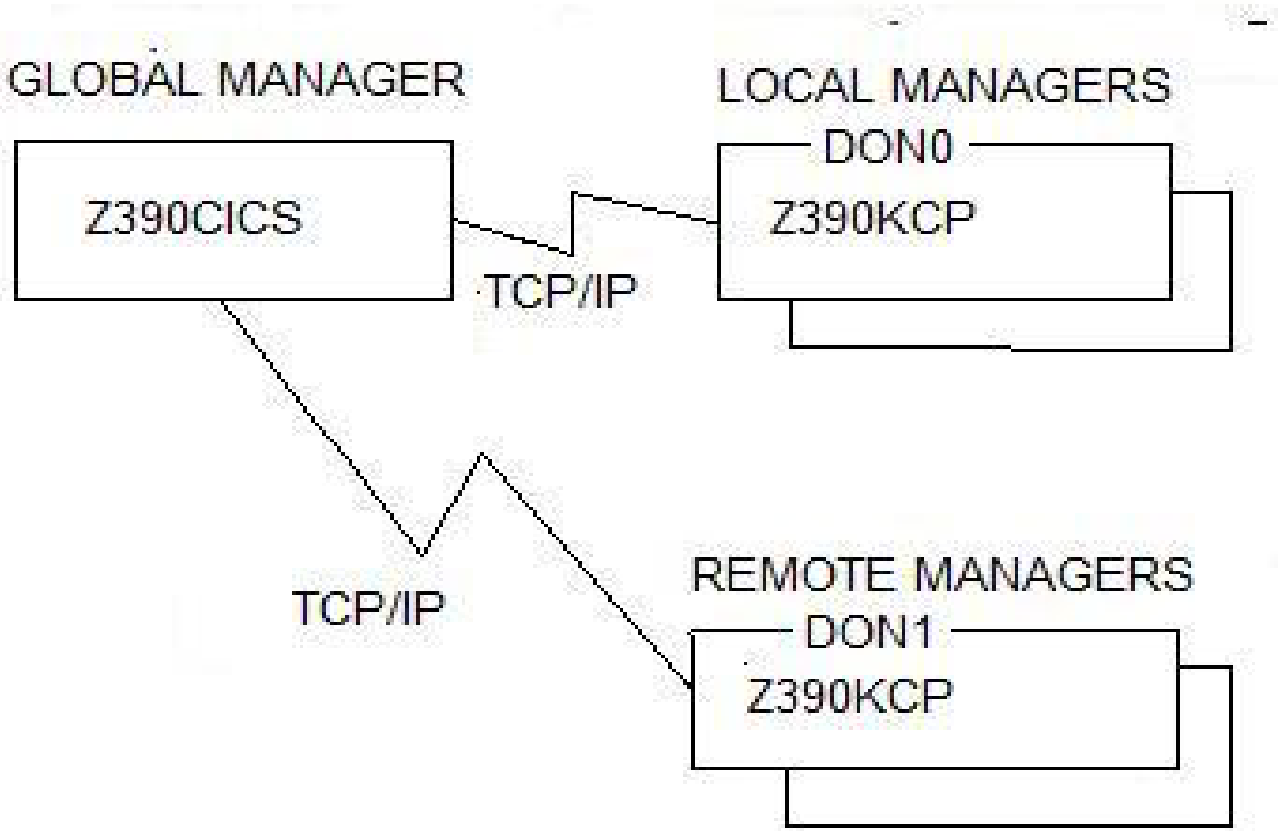
SHARE
Technology · Connections · Results

zCICS V7 Support for zCOBOL and z390 assembler

Melvyn Maltz
Automated Software Tools Corporation

Tuesday, March 3, 2009 8:00-9:00 AM
Session Number 8194 – Hilton Salon F

zCICS Overview



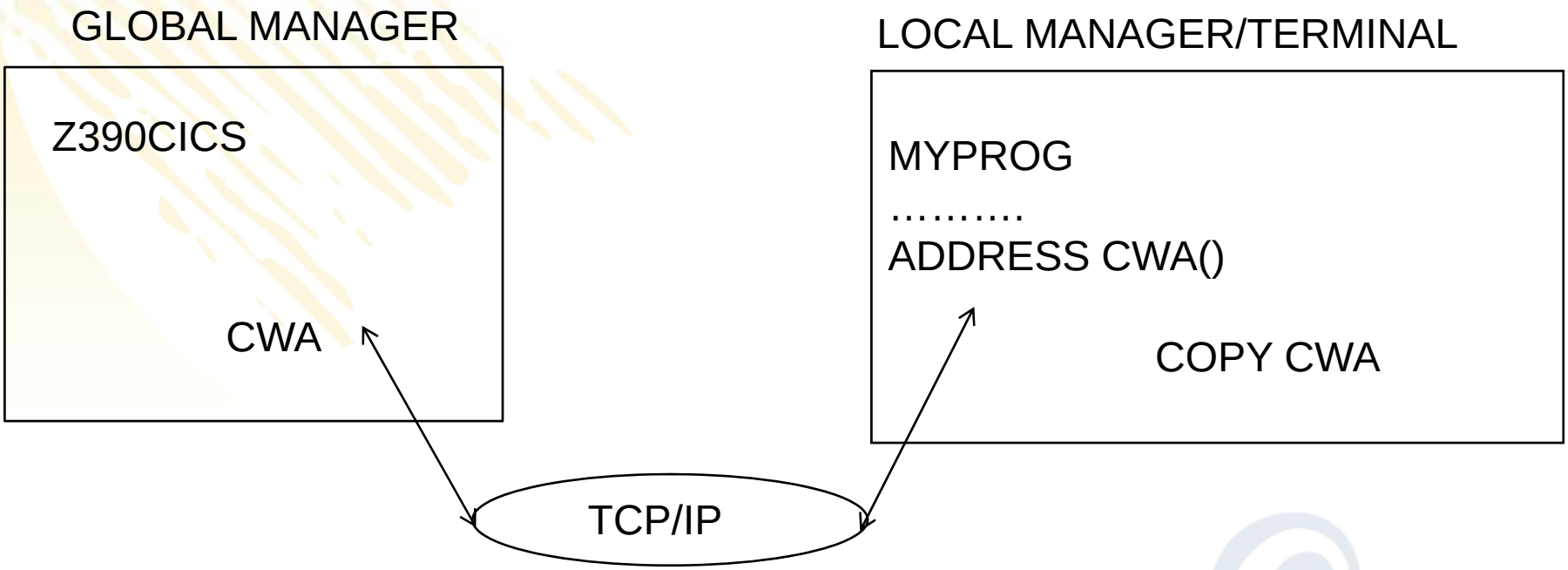
zCICS GUI Screen

zCICS supported commands

- General
  - HANDLE AID
  - HANDLE CONDITION
  - IGNORE CONDITION
  - POP HANDLE
  - PUSH HANDLE
  - ADDRESS
- TC
  - RECEIVE
  - SEND
- FC
  - READ
  - STARTBR
  - READNEXT
  - READPREV
  - ENDBR
  - RESETBR
- SC
  - FREEMAIN
  - GETMAIN
- TS
  - DELETEQ
  - READQ
  - WRITEQ
- PC
  - ABEND
  - HANDLE ABEND
  - LINK
  - LOAD
  - RELEASE
  - RETURN
  - XCTL
- IC
  - ASKTIME
  - ASKTIME ABSTIME
  - DELAY
  - FORMATTIME
  - START
  - RETRIEVE
  - CANCEL
- KC
  - ENQ
  - DEQ
- BMS
  - SEND MAP
  - RECEIVE MAP
  - SEND CONTROL
- DC
  - DUMP

zCICS CWA and ENQ/DEQ

INI CWASIZE=nnnnnnn….



EXEC CICS ENQ RESOURCE()

zCICS BMS Extensions

- **More cross-checking for Macro and execution**

  MAPFAIL now uses EIBRESP2.

  ATTRB=(ALPHA)

  XINIT=FFhh

  PICIN/PICOUT supported by Assembler as an edit word

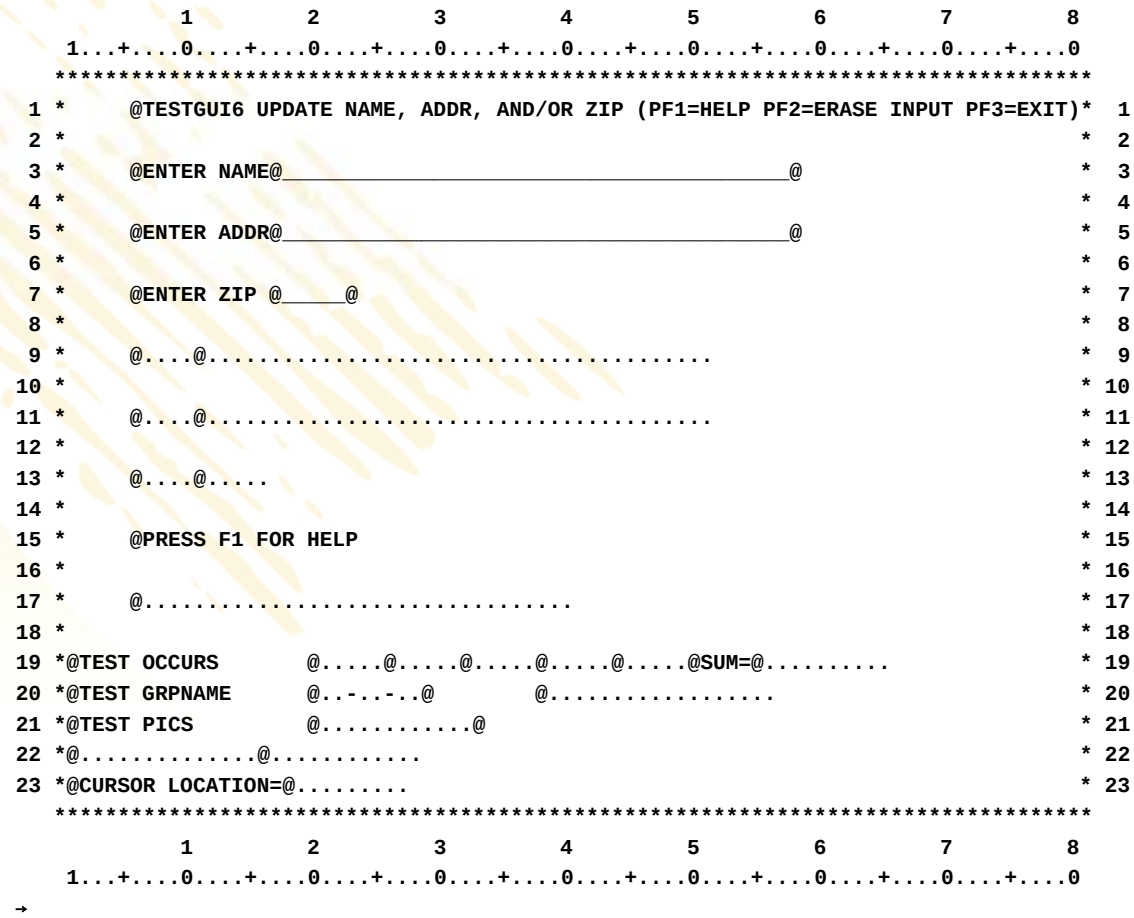
  PICOUT=5C20216B202020

  Data is 12345, displayed as *12,345

  PICOUT=5B20216B202020

  Data is 1234 , displayed as $1,234

zCICS BMS Map Layout Example



```
                1         2         3         4         5         6         7         8
       1...+....0....+....0....+....0....+....0....+....0....+....0....+....0....+....0
      ********************************************************************************
    1 *     @TESTGUI6 UPDATE NAME, ADDR, AND/OR ZIP (PF1=HELP PF2=ERASE INPUT PF3=EXIT)*  1
    2 *                                                                              *  2
    3 *     @ENTER NAME@_____________________________________________@               *  3
    4 *                                                                              *  4
    5 *     @ENTER ADDR@_____________________________________________@               *  5
    6 *                                                                              *  6
    7 *     @ENTER ZIP @_____@                                                       *  7
    8 *                                                                              *  8
    9 *     @....@.....................................                              *  9
   10 *                                                                              * 10
   11 *     @....@.....................................                              * 11
   12 *                                                                              * 12
   13 *     @....@.....                                                              * 13
   14 *                                                                              * 14
   15 *     @PRESS F1 FOR HELP                                                       * 15
   16 *                                                                              * 16
   17 *     @..............................                                          * 17
   18 *                                                                              * 18
   19 *@TEST OCCURS       @.....@.....@.....@.....@.....@SUM=@.........              * 19
   20 *@TEST GRPNAME      @..-..-..@        @..................                     * 20
   21 *@TEST PICS         @...........@                                              * 21
   22 *@..............@............                                                  * 22
   23 *@CURSOR LOCATION=@.........                                                   * 23
      ********************************************************************************
                1         2         3         4         5         6         7         8
       1...+....0....+....0....+....0....+....0....+....0....+....0....+....0....+....0
   →
```

zCICS Supplied Transactions

- **Many test transactions**

- **CEMT I TERm**
- **CEMT I TRAn**
- **CEMT I FILe**
- **CEMT I SYStem**
- **CEMT I ENQueue**
- **CEBR**

- **CEMT S TER OUT**
- **CEMT P SHU**
- **CEMT P SHU IMM**

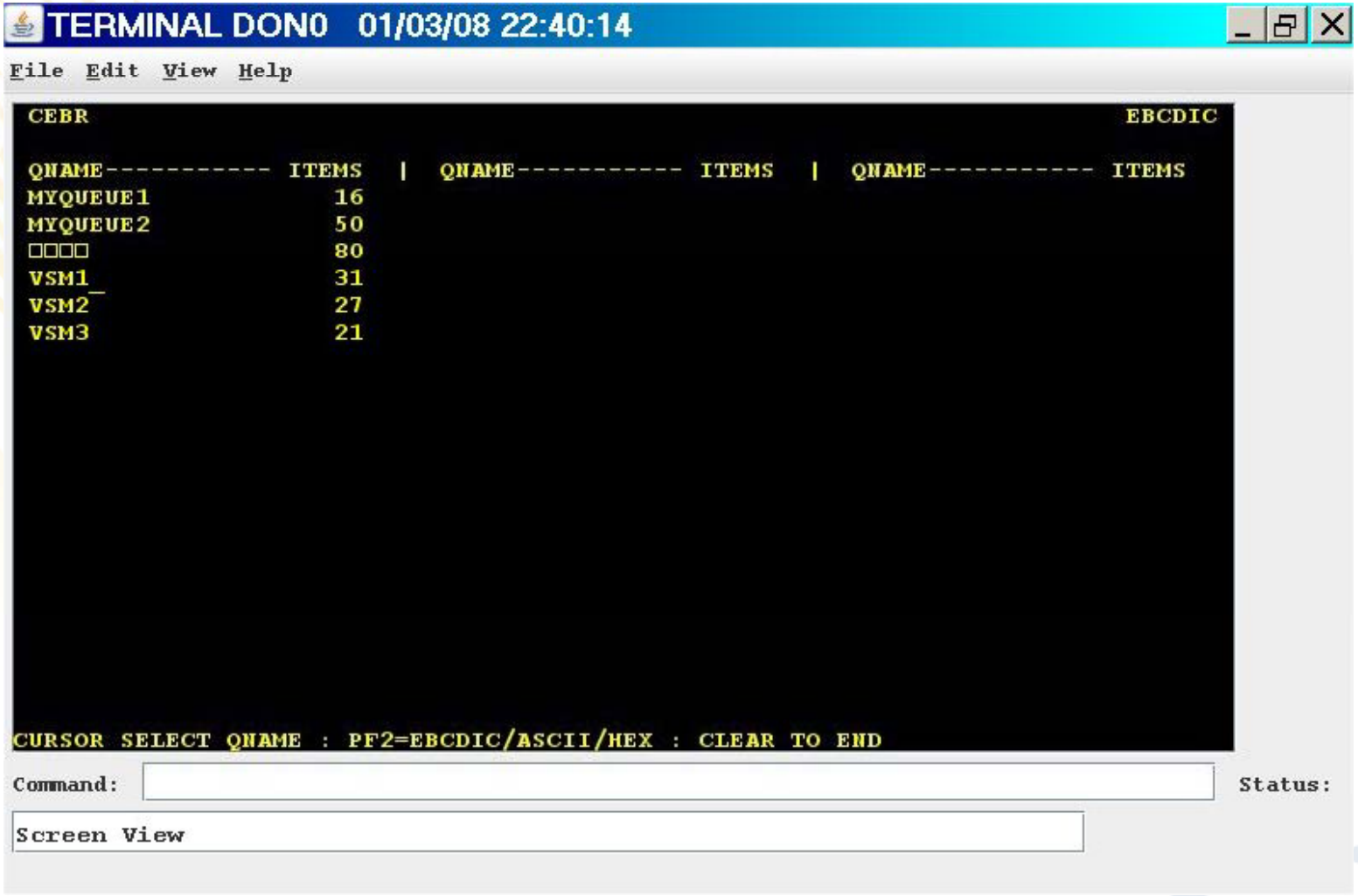
zCICS Supplied Transaction Example

CEMT I ENQ

```
RESOURCE-------------LENGTH  USE COUNT  OWNER  WAITING
```

- `MYRES4                    6          1   DON0     1`
- `MYRES5                    6          1   DON1     0`

zCICS Temporary Storage Screen A



zCICS Temporary Storage Screen B

zCICS Seq. Terminal Support (1 of 2)

- **Regression test your transactions.**

- **Run a transaction with INI parm SEQ_TERM=TRACE**
- **Run the extract program Z390SEQ to build the data streams**
- **Sequence all of your data streams**
- **Set INI parm SEQ_TERM=YES**
- **Run the simulation, you can see it happen on screen**
- **Your whole life will flash before your eyes**

zCICS Seq. Terminal Support (2 of 2)

- Regression test your transactions.
  - Run the comparator Z390CMPG, review the output
  - Refine the comparator by building an exclusion file for variable data like dates and times

zCICS Documentation (1 of 2)

- There's a lot of it.
  - None of it is meant to replace IBM's Manuals.
  - The information given refers to zCICS, its implementation, workings, extensions and command/parameter support.

zCICS Documentation (2 of 2)

- Readme
- Application Programming Guide
- Diagnosis Reference
- History
- Sequential Terminal Support
- Supplied Transactions
- System Programmer's Guide
- VSAM Guide
- Basic Mapping Support

zCICS Questions and Answers

- Can I compile and test EXEC CICS COBOL programs using z390 zCICS?
- Can I statically or dynamically link CICS COBOL and assembler modules?
- Can I use z390 SOA application generator COBOL and assembler support with CICS applications?

z390 zCOBOL zCICS Q and A Time

- **Which zCOBOL extension is highest priority?**
- **Which zCICS extension is highest priority?**
- **Which z390 extension is highest priority?**
- **How do I request a bug fix or enhancement?**
- **How do I volunteer to join the z390 project to help develop, test, and/or document z390, zCOBOL, and zCICS tools?**