IBM Mainframe Assembler for Programmers

Duration

15 days.

Participants

This course is designed for programmers already experienced in another procedural language. Non-programmers should first learn program logic. Knowledge of JCL is recommended.

Objectives

Upon successful completion of this course, you will be able to:

- Code, link and execute structured assembler programs using the fixed point binary instruction set, the packed decimal instruction set, and most of the non-privileged logical character instruction set.
- Read and interpret an assembler listing, and correct diagnostics.
- Debug program interrupt abends, such as S0C7, S0C9, S0C6, S0C1, S0CB. Troubleshooting skills are developed throughout the course.
- Convert between symbolic and explicit machine instruction code, and interpret instructions in a hexadecimal dump.
- Use DS, DC (including C, X, F, H, S, P, Z, A, and V type constants), START, CSECT, DSECT, EXTRY, EXTRN, USING, DROP, ORG, LTORG, CNOP, and END.
- Format the source listing using TITLE, EJECT, SPACE, PRINT and comments.
- Code the I/ O macros (OPEN, CLOSE, GET, PUT, DCB) for QSAM FB MOVE mode.
- Call a sub-program, pass data and access that data in the sub-program.
- Explain the use of standard entry linkage and exit linkage.
- Use appropriate assembler coding for above and below the line programs.
- Describe HLA (High Level Assembler) features.
- Describe the organization and use of the IBM manuals used in this course.

Format

Lecture and hands-on workshops.

Overview

Course topics include data definition, use of packed decimal and fixed point binary instruction sets, looping techniques, formatting and editing output reports, modular and self-documenting code, standard entry and exit linkage, writing main and sub-programs with standard linkage and data passing, DSECTs, macros for QSAM move mode I/ O, Boolean instructions for bit manipulation and testing, arithmetic for large numbers, debugging techniques, and MVS/ XA coding considerations. Students code a variety of programs and sub-programs to reinforce the lecture material.

Topic Outline

Background

- Numbering systems (hexadecimal, binary, decimal)
- Addressability
  - Absolute addresses (24 bit and 31 bit)
  - Relative addressing - base displacement concept
  - Boundaries (H, F, D)

Data Definition

- Character / packed decimal / fixed point binary
- DS and DC instructions
- Constants - C, X, B, F, H, D, P, Z
- Introduction to A and V address constants
- Using the IBM System/ 370 Reference Summary Booklet
- Redefining fields and records - ORG

Computer Listing: Overview

- Assembler listing and control blocks
- Linkage editor listing and control blocks
- SYSUDUMP layout - control blocks, registers, unformatted dump
- Locating data in the dump

Assembler Fundamentals

- Machine instruction formats (RR, RS, RX, SS, SI, S)
- Operands - explicit code, symbolic addresses, literals
- Basic machine instruction set for fixed point binary instructions
  - Data conversion - PACK, CVB, CVD, UNPK, MVZ
  - Arithmetic - A, AH, AR, S, SH, SR, M, MH, MR, D, DR
  - Moving data - LM, L, LH, IC, STM, ST, STH, STC, MVC
  - Branching - B, BR
- Comments and structured coding considerations
- Basic QSAM macros
  - OPEN, GET, PUT, CLOSE, DCB
  - Using the Data Management Macro Instructions manual
- Basic assembler instruction set - CSECT, USING, END
- Base register - USING, LR
- Entry and exit linkage - concepts and code
- Basic CLG JCL
- Manuals
  - IBM Reference Summary Booklet
  - IBM Principles of Operation
  - Assembler Language manual
  - Assembler Programmer's Guide manual
- *MACHINE EXERCISE # 1*

Topic Outline

Standard Instruction Set, Moves, Compares, Branches, Modularization

Move instructions - MVN, MVI, MVC, MVZ
Conditional Branching with fixed point binary data
Compare - CR, C, CH
Extended mnemonic instructions
Data validation techniques
Structured programming
Self documentation
Meaningful and easy to read comments and code
Meaningful symbolic names and labels
Modular programming - BAL, BALR, BR (to return)
Formatted source listing - TITLE, EJECT, SPACE, PRINT
MACHINE EXERCISE # 2

Decimal Instruction Set

Arithmetic - AP, SP, MP, DP
Moving data - ZAP
Conditional Branching - CP
MACHINE EXERCISE # 3
Truncating and rounding techniques - SRP, MVO
Editing reports - ED, EDMK
MACHINE EXERCISE # 4

Debugging

Using the System Messages and System Codes manuals
Debugging dumps - S0C7, S0CB, S0C6, S0C9, S0C1

Linking Multiple Programs

Modular programming
Address constants - A and V
Loading Addresses - LA (MVS/ XA warning)
Linkage editor
Program linkage and CALL statements
DSECTs
MACHINE EXERCISE # 5

Logical Instruction Set

And - turn bits off - NI, NC, NR, N
Or - turn bits on - OI, OC, OR, O
Exclusive or - flip bits and zero out fields- XI, XC, XR, X
Testing bits - TM
Conditional branch - extended mnemonic instructions, BCT, BCTR
MACHINE EXERCISE # 6

Topic Outline

Looping and Introduction to Table Handling

Defining one and two dimensional tables
Addressing using RX instructions and indexing
Addressing using SS instructions
Controlling loops - BCT, BCTR, BXLE, BXH
MACHINE EXERCISE # 7

Introduction to Advanced Techniques

Advanced loop control techniques
Multiple base registers
Placement of literals - LTORG
Conditional no-ops - CNOP
Data shifting - SLA, SRA, SLDA, SRDA, SLL, SRL, SLDL, SRDL
Setting program mask - SPM
Logical arithmetic (for large numbers) - AL, ALR, SL, SLR
Advanced load instructions - LTR, LCR, LPR, LNR
Translating data - TR
Testing and validating data - TRT
Executing one line subroutines - EX

Summary and Review